CSM32RV20
User Manual

1 Introduction

1.1 Feature and benefits

CSM32RV20 is a low-power MCU chip based on the RISC-V.

- Built-in RISC-V RV32IMAC Core（2.6 CoreMark/MHz）
- The working frequency is up to 32MHz
- Built-in 4kB SRAM
- The built-in 8B ALWAYS register and can save the data in the PD2
- Built-in 40kB embedded FLASH, 512B NVM, can ERASE and Program at least 100,000 times
- Built-in 2 SPI MASTER
- Built-in 1 I^2C MASTER
- Built-in 4 UART, maximum rate 1Mbps
- Built-in 2 TIMER,each TIMER supports the 4 channel complementary PWM output
- With a rapid precision of 13/14/15/16bit ADC, 1.2v high precision reference;
- Wide ADC input voltage range: 0 ~ VDD（VDD ≤ 4.8V）
- The ADC supports 11 input channels and supports up to 9 touch key
- Built-in 3 rapid comparators
- Built-in low voltage detection module
- Built-in RF detection module
- Most of the 30 GPIO, where PA supports external interrupts (with up to 16 external interrupts)
- Built-in hardware Watchdog
- Built-in 1 RTC, not working in PD2 mode
- Supports 4 low power modes, minimum power consumption less than 1 uA (Watchdog working)
- Built-in 32-bit real random number generator
- Support serial port and wireless ISP online upgrade (wireless ISP needs to be external to the Si24R1)
- Support the cjtag 2-line debugging interface
- Working voltage range: 1.8 ~ 5.5v
- Working temperature range: -40 ~ 105℃
- 4x4mm QFN32 package、TSSOP20 or QFN20 package

1.2 Pinning information

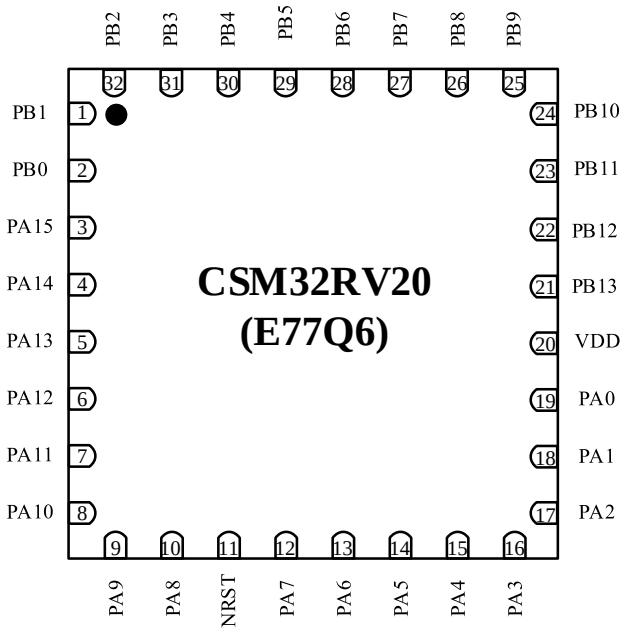


Figure1-1 Pin information（4x4 mm QFN32—E77Q6）

Table1-1 Table 1-1 Pining description（4x4 mm QFN32—E77Q6）

Pin	Name	I/O	Multiplexing function	Additional function
1	PB1	IO		-
2	PB0	IO		-
3	PA15	IO	TIMER2_CH4N/UART4_TX/ EXTI[15]	RF detection
4	PA14	IO	ADC_TRI/TIMER2_CH4/UART4_RX/ EXTI[14]	-
5	PA13	IO	TIMER2_CH3N/EXTI[13]	voltage output REFP
6	PA12	IO	TIMER2_CH3/EXTI[12]	PGA input
7	PA11	IO	TIMER2_BKIN/TIMER2_CH2N/UART3_TX/EXTI[11]	voltage output REFN
8	PA10	IO	TIMER1_BKIN/TIMER2_CH2/UART3_RX/EXTI[10]	ADC_IN9
9	PA9	IO	TIMER1_CH1/UART1_TX/TIMER2_CH1N/EXTI[9]	ADC_IN8
10	PA8	IO	SDA/UART1_RX/TIMER2_CH1/EXTI[8]	ADC_IN7
11	NRST	I	External reset, low level reset	-
12	PA7	IO	SCL/MOSI/TIMER1_CH4N/EXTI[7]	ADC_IN6
13	PA6	IO	UART1_TX/MISO/TIMER1_CH4/EXTI[6]	ADC_IN5
14	PA5	IO	UART1_RX/SCK/TIMER1_CH3N/EXTI[5]	ADC_IN4

15	PA4	IO	MOSI/TIMER1_CH1IN/TIMER1_CH3/UART2_TX/EXTI[4]	ADC_IN3
16	PA3	IO	MISO/TIMER1_CH1N/TIMER1_CH2N/UART2_RX/EXTI[3]	ADC_IN2
17	PA2	IO	SCK/TIMER1_CH1/TIMER1_CH2/EXTI[2]	-
18	PA1	IO	TMSC/SDA/TIMER1_CH1N/EXTI[1]	-
19	PA0	IO	TCKC/SCL/TIMER1_CH1/EXTI[0]	-
20	VDD	S	Power	-
21	PB13	IO	-	OSC_OUT
22	PB12	IO	-	OSC_IN
23	PB11	IO	-	COMP3-
24	PB10	IO	-	COMP3+
25	PB9	IO	-	COMP2-
26	PB8	IO	-	COMP2+
27	PB7	IO	-	COMP1-
28	PB6	IO	-	COMP1+
29	PB5	IO	-	-
30	PB4	IO	SPI2_MISO	-
31	PB3	IO	SPI2_MOSI	-
32	PB2	IO	SPI2_SCK	-
-	VSS	S	Ground,on the bottom of the package	

Note：S：Power supply pin；I：Input；O：Output；I/O：InOut；

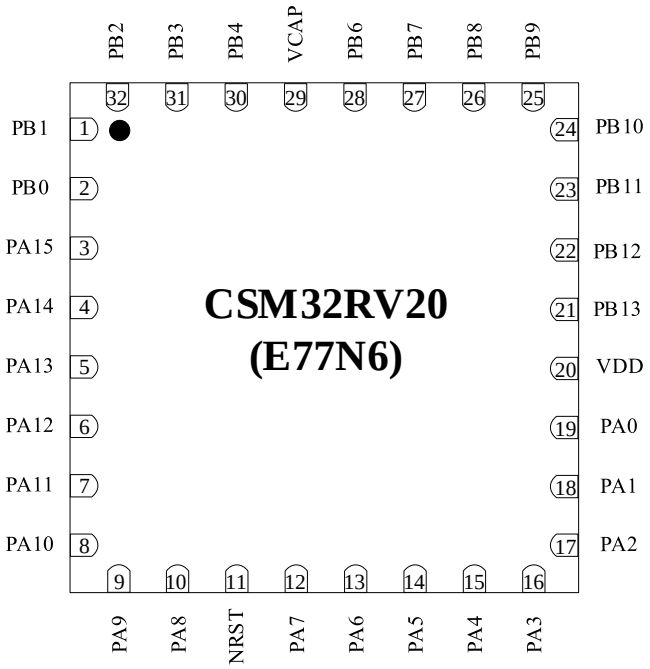


Figure1-2 Pin information（4x4 mm QFN32—E77N6）

Table1-2 Pin description（4x4 mm QFN32—E77N6）

Pin	Name	I/O	Multiplexing function	Additional function
1	PB1	IO		-
2	PB0	IO		-
3	PA15	IO	TIMER2_CH4N/UART4_TX/ EXTI[15]	RF detection
4	PA14	IO	ADC_TRI/TIMER2_CH4/UART4_RX/ EXTI[14]	-
5	PA13	IO	TIMER2_CH3N/EXTI[13]	Voltage output REFP
6	PA12	IO	TIMER2_CH3/EXTI[12]	PGA input
7	PA11	IO	TIMER2_BKIN/TIMER2_CH2N/UART3_TX/EXTI[11]	Voltage output REFN
8	PA10	IO	TIMER1_BKIN/TIMER2_CH2/UART3_RX/EXTI[10]	ADC_IN9
9	PA9	IO	TIMER1_CH1/UART1_TX/TIMER2_CH1N/EXTI[9]	ADC_IN8
10	PA8	IO	SDA/UART1_RX/TIMER2_CH1/EXTI[8]	ADC_IN7
11	NRST	I	External reset, low level reset	-
12	PA7	IO	SCL/MOSI/TIMER1_CH4N/EXTI[7]	ADC_IN6
13	PA6	IO	UART1_TX/MISO/TIMER1_CH4/EXTI[6]	ADC_IN5
14	PA5	IO	UART1_RX/SCK/TIMER1_CH3N/EXTI[5]	ADC_IN4
15	PA4	IO	MOSI/TIMER1_CH1IN/TIMER1_CH3/UART2_TX/EXTI[4]	ADC_IN3

16	PA3	IO	MISO/TIMER1_CH1N/TIMER1_CH2N/UART2_RX/EXTI[3]	ADC_IN2
17	PA2	IO	SCK/TIMER1_CH1/TIMER1_CH2/EXTI[2]	-
18	PA1	IO	TMSC/SDA/TIMER1_CH1N/EXTI[1]	-
19	PA0	IO	TCKC/SCL/TIMER1_CH1/EXTI[0]	-
20	VDD	S	Power	-
21	PB13	IO	-	OSC_OUT
22	PB12	IO	-	OSC_IN
23	PB11	IO	-	COMP3-
24	PB10	IO	-	COMP3+
25	PB9	IO	-	COMP2-
26	PB8	IO	-	COMP2+
27	PB7	IO	-	COMP1-
28	PB6	IO	-	COMP1+
29	VCAP	S	LDO power supply output (only internal circuits are used, and 1uF is required to be decouped to the ground)	-
30	PB4	IO	SPI2_MISO	-
31	PB3	IO	SPI2_MOSI	-
32	PB2	IO	SPI2_SCK	-
-	VSS	S	Ground,on the bottom of the package	

Note：S：Power supply pin；I：Input；O：Output；I/O：InOut；

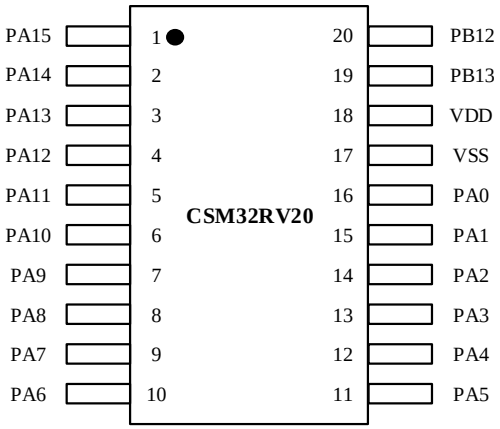


Figure1-3 Pin information（TSSOP 20）

Table1-3 Pin description（TSSOP 20）

Pin	Name	I/O	Multiplexing function	Additional function
1	PA15	IO	TIMER2_CH4N/TX4/ EXTI[15]	RF detection
2	PA14	IO	ADC_TRI/TIMER2_CH4/RX4/ EXTI[14]	-
3	PA13	IO	TIMER2_CH3N/EXTI[13]	Voltage output REFP
4	PA12	IO	TIMER2_CH3/EXTI[12]	PGA input
5	PA11	IO	TIMER2_BKIN/TIMER2_CH2N/TX3/EXTI[11]	Voltage output REFN
6	PA10	IO	TIMER1_BKIN/TIMER2_CH2/RX3/EXTI[10]	ADC_IN9
7	PA9	IO	TIMER1_CH1/TX/TIMER2_CH1N/EXTI[9]	ADC_IN8
8	PA8	IO	SDA/RX/TIMER2_CH1/EXTI[8]	ADC_IN7
9	PA7	IO	SCL/MOSI/TIMER1_CH4N/EXTI[7]	ADC_IN6
10	PA6	IO	TX1/MISO/TIMER1_CH4/EXTI[6]	ADC_IN5
11	PA5	IO	RX1/SCK/TIMER1_CH3N/EXTI[5]	ADC_IN4
12	PA4	IO	MOSI/TIMER1_CH1IN/TIMER1_CH3/TX2/EXTI[4]	ADC_IN3
13	PA3	IO	MISO/TIMER1_CH1N/TIMER1_CH2N/RX2/EXTI[3]	ADC_IN2
14	PA2	IO	SCK/TIMER1_CH1/TIMER1_CH2/EXTI[2]	-
15	PA1	IO	TMSC/SDA/TIMER1_CH1N/EXTI[1]	-
16	PA0	IO	TCKC/SCL/TIMER1_CH1/EXTI[0]	-
17	VSS	S	Ground	-
18	VDD	S	Power	-
19	PB13	O	-	OSC_OUT
20	PB12	I	-	OSC_IN

Note：S：Power supply pin；I：Input；O：Output；I/O：InOut

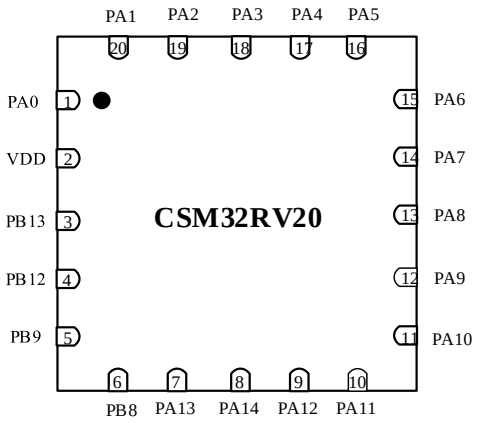


Figure1-4 Pin information（3x3 mm QFN20）

Table1-4 Pin description（3x3 mm QFN20）

Pin	Name	I/O	Multiplexing function	Additional function
1	PA0	IO	TCKC/SCL/TIMER1_CH1/EXTI[0]	-
2	VDD	S	Power	-
3	PB13	IO	-	OSC_OUT
4	PB12	IO	-	OSC_IN
5	PB9	IO	-	COMP2-
6	PB8	IO	-	COMP2+
7	PA13	IO	TIMER2_CH3N/EXTI[13]	Voltage output REFN
8	PA14	IO	ADC_TRI/TIMER2_CH4/RX4/ EXTI[14]	-
9	PA12	IO	TIMER2_CH3/EXTI[12]	PGA input
10	PA11	IO	TIMER2_BKIN/TIMER2_CH2N/TX3/EXTI[11]	Voltage output REFP
11	PA10	IO	TIMER1_BKIN/TIMER2_CH2/RX3/EXTI[10]	ADC_IN9
12	PA9	IO	TIMER1_CH1/TX/TIMER2_CH1N/EXTI[9]	ADC_IN8
13	PA8	IO	SDA/RX/TIMER2_CH1/EXTI[8]	ADC_IN7
14	PA7	IO	SCL/MOSI/TIMER1_CH4N/EXTI[7]	ADC_IN6
15	PA6	IO	TX1/MISO/TIMER1_CH4/EXTI[6]	ADC_IN5
16	PA5	IO	RX1/SCK/TIMER1_CH3N/EXTI[5]	ADC_IN4
17	PA4	IO	MOSI/TIMER1_CH1IN/TIMER1_CH3/TX2/EXTI[4]	ADC_IN3
18	PA3	IO	MISO/TIMER1_CH1N/TIMER1_CH2N/RX2/EXTI[3]	ADC_IN2
19	PA2	IO	SCK/TIMER1_CH1/TIMER1_CH2/EXTI[2]	-
20	PA1	IO	TMSC/SDA/TIMER1_CH1N/EXTI[1]	-
-	VSS	S	Ground,on the bottom of the package	

Note：S：Power supply pin；I：Input；O：Output；I/O：InOut；

Contents

2 Memory and Bus architecture

2.1 System architecture

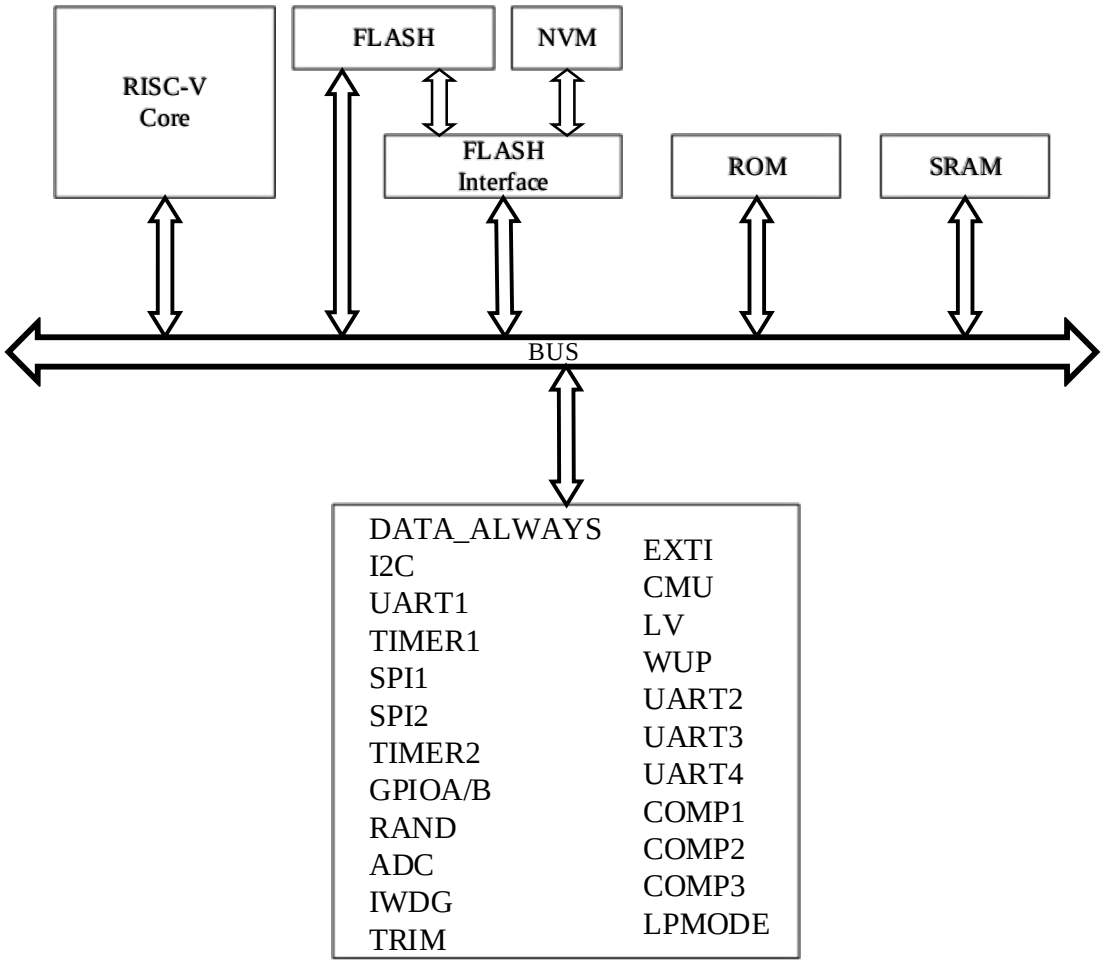


Figure2-1 System architecture

2.2 Memory map

Table2-1 Memory mapping

Base	Top	Attr	Description	Notes
0x0000_0100	0x0000_0FFF	RWX	Debug	Debug Address Space
0x0200_0000	0x01FF_FFFF	RW	CLIC	On Core Complex Devices
0x2000_0000	0x2000_9FFF	RWX	CODE	40kB FLASH Program Space
[1]		RWX	NVM	512B NVM（save the data of users）
0x2002_0000	0x2002_0FFF	RWX	DATA	4kB SRAM
0x2002_8000	0x2002_8007	RWX	DATA_ALWAYS	Save the data in PD2 mode
0x2100_0000	0x2100_17FF	RWX	CODE	Bootloader ROM
0x3000_0004	0x3000_000F	RW	I2C	Peripherals
0x3000_0010	0x3000_0017	RW	UART1	
0x3000_0018	0x3000_005F	RW	TIMER1	
0x3000_0060	0x3000_006B	RW	SPI1	
0x3000_0070	0x3000_007B	RW	SPI2	
0x3000_0098	0x3000_0103	RW	TIMER2	
0x3000_0200	0x3000_026F	RW	GPIOA/B	
0x3000_0238	0x3000_023F	RW	RANDGEN	
0x3000_0280	0x3000_0297	RW	ADC	
0x3000_02A0	0x3000_02AB	RW	IWDG	
0x3000_02C0	0x3000_02C7	RW	EXTI	
0x3000_02E0	0x3000_02F7	RW	CMU	Peripherals
0x3000_e0330	0x3000_0333	RW	LV	
0x3000_0600	0x3000_0607	RW	LPMODE	
0x3000_0610	0x3000_061B	RW	WUP	
0x3000_0700	0x3000_0707	RW	UART2	
0x3000_0800	0x3000_0807	RW	UART3	
0x3000_0900	0x3000_0907	RW	UART4	
0x3000_0B00	0x3000_0D0B	RW	COMP1/2/3	

[1] NVM can only be accessed by the operating function flash_operation(), which cannot be accessed by absolute address,more details see20.3.

2.3 Embedded SRAM

CSM32RV20 has a built-in 4KB SRAM that supports byte, half-word (16 bit), or full-word (32 bit). The start address of SRAM is 0x2002_0000.

2.4 Embedded FLASH/NVM

FLASH/NVM Key Features：

- FLASH size is 10K×32bits（40K Bytes）；
- NVM size is 128×32bits（512Bytes）；
- FLASH/NVM is origanized by sector with 512 bytes per sector；
- Support for online reading and writing, which can be used to store user programs and data；
- Support read by byte、half-word or full-words,write by byte
- Support sector-by-sector erase
- Support whole Chip Erase(erase FLASH and NVM)
- FLASH supports read/write protection

2.5 Embedded ROM

Built-in ROM is used to store boot.

2.6 DATA_ALWAYS

The DATA_ALWAYS memory can save the data in the Power Down 2 mode. Each read and write takes up 12 system clock cycles and can only be written in words. In addition, MCU is executed for single cycle, without bus delay.

address（HEX）	Reset value	Function
0x2002_8000	Arbitrary value	Special memory, only lost data under power failure
0x2002_8004	Arbitrary value	Special memory, only lost data under power failure

3 Low power mode(LPMODE)

After the system or the power supply is reset, the microcontroller is in the running state. Clock execution program code for the MCU using RCOSC in the running state by default. When the MCU does not need to continue running, you can use multiple low-power modes to save power consumption, such as waiting for an external interrupt. Based on the minimum power consumption, the fastest start time and the need of the available wake source, select the best compromise to help the user select a low power mode.

CSM32RV20 has 4 low power mode:

- Standby mode（the core is stopped, and the peripherals can still work）;
- Sleep mode（all clocks can be stopped expect 3K clock）;
- Power-down mode1（see Table 3-1）;
- Power-down mode2（see Table 3-1, IO holds the state before Power-down）;

Furthermore, in operating mode, power consumption can be reduced by:

- Reduce the system clock;
- Close the clock of the unused peripherals;
- Reasonable configuration of I / O

Table3-1 LPMODE overview

Operating mode	LPMODE	Main LDO	LPLDO	RCOSC	OSC	3K[1]	Wake-up source
Standby mode(IDLE)	2'b00	ON	ON	ON	ON	ON	Any interrupt、watch dog or reset
Sleep mode(SLEEP)	2'b01	ON	ON	OFF	OFF	ON	Any interrupt、watch dog or reset
Power-down mode1(PD1)	2'b10	OFF	ON	OFF	OFF	ON	Any interrupt、watch dog or reset
Power-down mode2(PD2)	2'b11	OFF	OFF	OFF	OFF	ON	Any interrupt、watch dog or reset

[1] 3k refers to the low speed clock within the MCU, which rate is 3kHz and cannot be closed;

3.1 LPMODE

3.1.1 Entering LPMODE

Execute WFI instruction after configuring LPMODE register, if there is no interrupt at present, MCU can enter low power mode directly. If there is an interrupt and the interrupt is enabled, the MCU can't enter the low power mode and continue to execute the program after the WFI instruction.

3.1.2 Exiting LPMODE

In order to successfully exit the low-power mode, it is necessary to enable the interrupt used for wake-up before executing the WFI instruction, and enable the corresponding clicintie (CLIC Interrupt Enable) register. After executing the WFI instruction and successfully entering the low-power mode, if the interrupt general enable has been turned on (mstate.MIE=1), when the enabled interrupt is generated, the MCU wakes up and jumps to the interrupt handler function to continue execution. If the interrupt master enable is not turned on (mstate.MIE=0), when the enabled interrupt is generated, the MCU wakes up and continues to run from the next instruction of the WFI instruction.

3.1.3 Standby mode(IDLE)

In standby mode, the MCU core stops running; any enabled interrupt, watchdog, and reset can make the MCU exit standby mode immediately. Standby mode wake-up consumes the least amount of time, but consumes more power.

Standby mode (gray part is not working module)		
RISC-V Core	GPIO	RTC
FLASH/NVM	SPI1/SPI2	TIMER1/TIMER2
RAM	I2C	WUP
DATA_ALWAYS	UART1	WDT
RCOSC	LV	COMP
OSC	RANDGEN	ADC
3K	UART2/3/4	cJTAG

3.1.4 Sleep mode(SLEEP)

In standby mode, the MCU core stops running and the high-speed clock is turned off. Arbitrarily enabled interrupts, watchdogs, and resets allow the MCU to exit sleep mode. When the enabled interrupt is generated, it takes a few to several hundred clocks, depending on the clock source, before execution can continue. The relevant peripheral can only work in 3K clock.

Note: In the sleep mode, the clock source of RTC/SPI1/SPI2/ TIMER1/TIMER2/ I2C/UART1/ADC can only be 3K.

Sleep mode (gray part is not working module)		
RISC-V 内核	GPIO	RTC
FLASH/NVM	SPI1/SPI2	TIMER1/TIMER2
RAM	I2C	WUP
DATA_ALWAYS	UART1	WDT
RCOSC	LV	COMP
OSC	RANDGEN	ADC
3K	UART2/3/4	cJTAG

3.1.5 Power-down mode1(PD1)

After entering power-down mode 1, the MCU core stops working, the high-speed clock turns off, and the FLASH enters Deep Sleep. The power consumption of the MCU is further reduced. Arbitrary interrupts, watchdogs and resets allow the MCU to exit power-down mode 1.

Before entering the power-down mode 1, the clock source of MCU must be set to 16MHz or below, and the external clock must be closed (per_en= 0).

After exiting power-down mode1(PD1), the user can configure the clock as needed.

power-down mode1 (gray part is not working module)		
RISC-V 内核	GPIO	RTC
FLASH/NVM	SPI1/SPI2	TIMER1/TIMER2
RAM	I2C	WUP
DATA_ALWAYS	UART	WDT

RCOSC	LV	COMP
OSC	RANDGEN	ADC
3K		cJTAG

3.1.6 Power-down mode2(PD2)

After entering Power Down Mode 2, the MCU core stops working, the high-speed clock and digital power supply are turned off, the FLASH enters Deep sleep, RAM and Registers data are lost, and the GPIOs are locked. This mode's power consumption is minimum.

Arbitrarily enabled interrupts, watchdogs, and resets can cause the MCU to exit Power Down Mode 2 and reset. After at least 160us, the user program starts to run again. After exiting from power-down mode 2, lprst_flag is set to 1.

Power-down mode2 (gray part is not working module)		
RISC-V 内核	GPIO	RTC
FLASH/NVM	SPI1/SPI2	TIMER1/TIMER2
RAM	I2C	WUP
DATA_ALWAYS	UART	WDT
RCOSC	LV	COMP
OSC	RANDGEN	ADC
3K		cJTAG

3.2 Low power registers

3.2.1 LPMODE register（LPMODE）

Address: 0x3000_0600

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:2	Reserved	Reserved bit
1:0	lpmode	2’b00：standby mode;　2’b01：sleep mode; 2’b10：power-down mode1;　2’b11：power-down mode 2。 Note: 1） The lpmode register also contains a set of mirror registers. When the chip receives a disturbance that causes the values of the registers and the mirror registers to be inconsistent, the chip resets and re-executes; 2） Before entering the power-down mode 1, the clock source of MCU must be set to 16MHz or below, and the external clock must be closed (per_en= 0)

3.2.2 LPMODE flag register（LPRST_FLAG）

Address：0x3000_0604

Reset value:0x0000_0000

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

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

Bits	Marking	Functional description
31:1	Reserved	Reserved bit
0	lprst_flag	The flag of the chip is woken up from power-down mode 2 and resets. 1: Reset is generated by wake-up from power-down mode 2; 0: Reset generated by other means

4 Reset and Clock Control

4.1 Reset

Most registers will be reset to the reset value when the reset occurs.

When any of the following events occurs, a system reset is generated:

1. NRST low level（external reset）；
2. Independent watchdog counting overflow（IWDG reset）；
3. Software reset（SRST）；
4. Power on reset；
5. Reset is generated by wake-up from power-down mode 2.

4.2 Software reset register（SRST）

Address: 0x3000_0360

Reset value:0x0000_0000

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

Bit	Marking	Functional description
31:1	Reserved	Reserved bit
0	srst	Write 1 to soft reset，system reset

4.3 Clock

- Automatically filter the clock jitter when the crystal vibrates;
- The clock source of Peripheral clock and MCU clock can be configured independently;
- The frequency division ratios of Peripheral clock and MCU clock can be configured independently, reducing the power consumption of the system working frequency;
- The built-in frequency divider supports 1:1 ~ 1:31, duty ratio of 50%.
- Supports no burr clock switching.

4.3.1 Functional introduction

4.3.1.1 Clock architecture

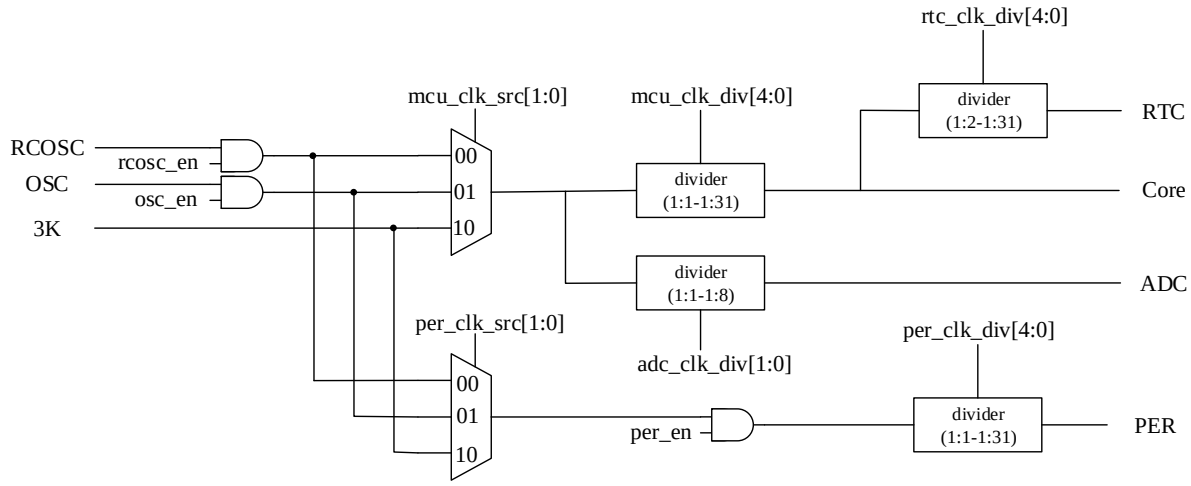


Figure4-1 clock architecture

There are three clock sources in the clock module, which are internal high speed oscillator (RCOSC), external crystal (OSC) and internal low speed clock (3K). Each clock frequency divider supports 1:1~1:31, and supports separate use of osc and rcosc clock sources to reduce power consumption. Each module's clock input supports selecting any clock source.

4.3.1.2 Independent watchdog (IWDG) clock

The clock source of internal Independent watchdog is 3K clock, which works which cannot be disabled. Once the door dog is opened, it cannot be shut down unless it is reset. The chip supports the restart of the IWDG, which is configured in the FLASH download program.

4.3.1.3 Control of clock module

- After the reset, RCOSC is selected as the current system clock by default;
- When a peripheral or MCU selects a clock source, the selected clock source cannot be disabled;
- Before switching the clock source, confirm whether the clock source to be switched is stable, otherwise you cannot complete the switch;
- The clock frequency of RTC should be less than 1/2 that of the MCU clock frequency.

Note: The mcu_clk_src is forbidden to be set to 01 when not using external OSC as the MCU clock source.

4.4 Clock control register

Base address：0x3000_02E4.

4.4.1 Peripheral clock enable contolr register（CMU_PER_EN）

Offset address:0x00

Reset value:0x0000_0001

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

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

Bit	Marking	Functional description
31:1	Reserved	Reserved bit
0	per_en	Peripheral clock enable bit, 0: disable peripheral clock 1: enable peripheral clock

4.4.2 Clock source selection（CMU_CLK_SEL）

Offset address:0x04

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:4	Reserved	Reserved bit
3:2	per_clk_src	Peripheral clock selection: 00：clock from internal high-speed oscillator RCOSC； 01：Clock from external high-speed crystal OSC

		10: Clock from internal low-speed clock 3K 11：-
1:0	mcu_clk_src	MCU clock source selection: 00：clock from internal high-speed oscillator RCOSC； 01：Clock from external high-speed crystal OSC 10: Clock from internal low-speed clock 3K 11：- Note: Setting mcu_clk_src to 01b is prohibited when external high-speed crystal OSC is not selected as the clock source of MCU.

4.4.3 Clock frequency prescaler register（CMU_CLK_DIV）

Offset address:0x08

Reset value:0x0000_0000

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

Bit	Marking	Functional description
31:15	Reserved	Reserved bit
14:10	rtc_clk_div	RTC prescaler: 0000x：RTC clock is divided by 2；0001x：RTC clock is divided by 2； 0010x：RTC clock is divided by 4；0011x：RTC clock is divided by 6； 0100x：RTC clock is divided by 8；0101x：RTC clock is divided by 10； 0110x：RTC clock is divided by 12；0111x：RTC clock is divided by 14； 1000x：RTC clock is divided by 16；1001x：RTC clock is divided by 18； 1010x：RTC clock is divided by 20；1011x：RTC clock is divided by 22； 1100x：RTC clock is divided by 24；1101x：RTC clock is divided by 26； 1110x：RTC clock is divided by 28；1111x：RTC clock is divided by 30 . Note: x is 0 or 1。
9:5	per_clk_div	Peripheral prescaler:

		00000：peripheral clock is not divided; 00001：peripheral clock is not divided; 00010：peripheral clock is divided by 2; 00011：peripheral clock is divided by 3; 00100：peripheral clock is divided by 4; 00101：peripheral clock is divided by 5; …….. 11111：peripheral clock is divided by 31
4:0	mcu_clk_div	MCU prescaler: 00000：MCU clock is not divided; 00001：MCU clock is not divided; 00010：MCU clock is divided by 2; 00011：MCU clock is divided by 3; 00100：MCU clock is divided by 4; 00101：MCU clock is divided by 5; …….. 11111：MCU clock is divided by 31

4.4.4 Clock source enable register（CLK_SRC_EN）

Offset address:0x0C

Reset value:0x0000_0002

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

Bit	Marking	Functional description
31:2	Reserved	Reserved bit
1	rcosc_en	RCOSC enable 0:RCOSC is enabled; 1: RCOSC is disabled When RCOSC is used as the internal clock input, the current clock signal input is

		not turned off even if the rcosc_en is set to 0. RCOSC is not turned off until the internal clock is switched to another clock source.
0	osc_en	crystal（OSC）enable 0：crystal is enabled；　　1：crystal is disabled

4.4.5 Clock state register（CMU_OSC_SR）

Offset address:0x10

Reset value:0x0000_0009

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								osc_st	rcosc_st	MCU_clk_st			per_clk_st		
								R	R	R			R		

Bit	Marking	Functional description
31:8	Reserved	Reserved bit
7	osc_st	OSC clock ready flag 0:OSC is not stabilized； 1:OSC has stabilized
6	rcosc_st	RCOSC clock ready flag 0:RCOSC is not stabilized； 1:RCOSC has stabilized
5:3	mcu_clk_st	The status of Clock Source which is selected to provide the clock for MCU 001：RCOSC is providing the clock for MCU 010：RCOSC is providing the clock for MCU 100：3K is providing the clock for MCU
2:0	per_clk_st	The status of Clock Source which is selected to provide the clock for peripheral 001：RCOSC is providing the clock for peripheral 010：RCOSC is providing the clock for peripheral 100：3K is providing the clock for peripheral

4.4.6 RCOSC frequency select register（RCOSC_SEL）

Address：0x3000_0E00

Reset value:0x0000_0000

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

Bit	Marking	Functional description
31:1	Reserved	Reserved bit
0	rcosc_sel	RCOSC clock frequency selection 0：RCOSC frequency is 16MHz；　1：RCOSC frequency is 32MHz

4.5 CMU register mapping

CMU register list

Base address：0x3000_02E4

Register	Offset	Description
CMU_PER_EN	0x00	Peripheral Clock Enable Control Register
CMU_CLK_SEL	0x04	Clock Source Selection Register
CMU_CLK_DIV	0x08	Clock frequency prescaler Register
CLK_SRC_EN	0x0C	Clock Source enable Register
CMU_OSC_SR	0x10	Clock State Register
RCOSC_SEL	-	RCOSC frequency select

5 General and alternate function I/Os

GPIOs are user-configurable general-purpose IOs, and each GPIO port can be independently configured for input/output, peripheral multiplexing, or analog functions. GPIOA0~15 correspond to PA0~PA15, GPIOB0~13 correspond to PB0~PB13.

The bootloader default is PA5, PA6.

5.1 GPIO Functional description

5.1.1 Main function

- Output states: pull-up and pull-down functions, open-source output, open-drain output and push-pull output;
- In power-down mode 2, the I/Os will remain the state before power-down;
- The output of the GPIO can come from the ODR register of the GPIO or the peripheral function output;
- Input state: Suspended input, pull-up and pull-down input;
- The input data is stored in the IDR register of the GPIO or peripheral data input;
- Supports analog function data transfer;
- Peripheral function interface is optional;
- You can flexibly select the corresponding input port for each GPIO;
- The IO modes are selected by the GPIO_MODER register for Input Mode, Output Mode, Alternate function Mode, and Analog Mode;

5.1.1.1 Alternate function

The external IO and internal module connection multiplexer of this chip is controlled by GPIOA_AFRH and GPIOA_AFRL registers. Write register GPIO_MODER[MODERx] = 2'b10 configures GPIOx to multiplexing function, which can flexibly map the internal module ports to the PAD.

Each IO has a splitter to connect the multiplexed functions, and the GPIO_AFRL and GPIO_AFRH registers control which function is multiplexed to, see Table 5-1 for multiplexing details.

After reset, the multiplexing controller will connect the PAD to the AF0 function

by default. UART1 supports ISP.

Table5-1 GPIO alternate function table

Pin Name	AF0	AF0_DIR	AF1	AF1_DIR	AF2	AF2_DIR	AF3	AF3_DIR
PA0	TCKC	I	SCL	IO	TIMER1_CH1	IO		
PA1	TMSC	IO	SDA	IO	TIMER1_CH1N	O		
PA2	SPI1_SCK	O	TIMER1_CH1	O	TIMER1_CH2	IO		
PA3	SPI1_MISO	I	TIMER1_CH1N	O	TIMER1_CH2N	O	UART2_RX	IO
PA4	SPI1_MOSI	O	TIMER1_CH1N	O	TIMER1_CH3	IO	UART2_TX	O
PA5	UART1_RX	I	SPI1_SCK	O	TIMER1_CH3N	O		
PA6	UART1_TX	O	SPI1_MISO	I	TIMER1_CH4	IO		
PA7	SCL	IO	SPI1_MOSI	O	TIMER1_CH4N	O		
PA8	SDA	IO	UART1_RX	IO	TIMER2_CH1	O		
PA9	TIMER1_CH1	O	UART1_TX	O	TIMER2_CH1N	O		
PA10	TIMER1_BKIN	I		O	TIMER2_CH2	O	UART3_RX	IO
PA11	TIMER2_BKIN	I		O	TIMER2_CH2N	O	UART3_TX	O
PA12	-	-		O	TIMER2_CH3	O		
PA13	-	-			TIMER2_CH3N	O		
PA14	ADC_TRI				TIMER2_CH4	IO	UART4_RX	IO

Pin Name	AF0	AF0_DIR	AF1	AF1_DIR	AF2	AF2_DIR	AF3	AF3_DIR
PA15					TIMER2_ CH4N	O	UART4 _TX	O
PB0	-	-	-	-	-	-	-	-
PB1	-	-	-	-	-	-	-	-
PB2	SPI2_ SCK	O	-	-	-	-	-	-
PB3	SPI2_ MOSI	O	-	-	-	-	-	-
PB4	SPI2_ MISO	I	-	-	-	-	-	-
PB5	-							
……..	-							
PB13	-							

Note：I:Input; O: Output; I/O: Inout.

5.1.1.2 Analog function

The GPIO function is configured by the GPIO_MODER register. When using modules such as ADC, COMP, etc. the I/O should be configured to Analog mode by setting register GPIO_MODER[MODERx] = 2’b11 to support data transmission of analog function.

Table5-2 GPIO analog function table

Pin	I/O	Analog mode （GPIO configuration）	Remark
PA3	I	ADC_IN2（ADC channel）	
PA4	I	ADC_IN3（ADC channel）	
PA5	I	ADC_IN4（ADC channel）	
PA6	I	ADC_IN5（ADC channel）	
PA7	I	ADC_IN6（ADC channel）	

Pin	I/O	Analog mode （GPIO configuration）	Remark
PA8	I	ADC_IN7（ADC channel）	
PA9	I	ADC_IN8（ADC channel） ADC external reference voltage N	
PA10	I	ADC_IN9（ADC channel） ADC external reference voltage P	
PA11	O	Voltage output REFP	You can only pull the current
PA12	I	PGA input	
PA13	O	Voltage output REFN	You can only sink the current
PA14	O	Reserved	
PA15	I	RF detection input	
PB6	I	COMP1+	
PB7	I	COMP1-	
PB8	I	COMP2+	
PB9	I	COMP2-	
PB10	I	COMP3+	
PB11	I	COMP3-	
PB12	I	XC1，OSC input	
PB13	I	XC2，OSC output	When be configured as OSC function, if there is no external crystal oscillator, XC1 can be injected into the clock from the outside. At this point, PB13 is the reverse output of PB12 and PB13 cannot be used as other functions.

Note：I: Input; O: Output; I/O: Inout.

5.1.2 Input configuration

When an IO port is configured in input mode (GPIO_MODER[MODERx] = 2'b00):

1. The output registers will be turned off;
2. The Schmidt trigger opens;
3. The pull-up/down control port will be configured according to the

GPIO_PUPDR register;

4. Each AHB cycle, the input data is updated once by the input register;
5. Each register represents the input value of an IO port.

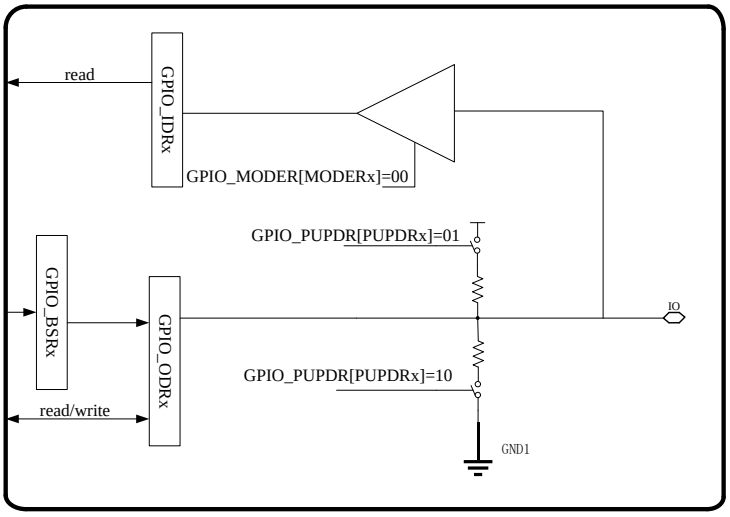


Figure5-1 GPIO input configuration schematic

When the GPIO is set to input mode, it can be used as an external interrupt. EXTI[15:0] corresponds to the interrupt of PA15 ~ 0. The settings of interrupt enable and mask and trigger mode can be set in GPIOA_LPMR and GPIOA_INTER.

Table5-3 GPIO input function

Pin	I/O	Input mode （GPIO configuration）	Remark
PA0	I	EXTI[0]	
PA1	I	EXTI[1]	
PA2	I	EXTI[2]	
PA3	I	EXTI[3]	
PA4	I	EXTI[4]	
PA5	I	EXTI[5]	
PA6	I	EXTI[6]	
PA7	I	EXTI[7]	
PA8	I	EXTI[8]	
PA9	I	EXTI[9]	
PA10	I	EXTI[10]	

Pin	I/O	Input mode （GPIO configuration）	Remark
PA11	I	EXTI[11]	
PA12	I	EXTI[12]	
PA13	I	EXTI[13]	
PA14	I	EXTI[14]	
PA15	I	EXTI[15]	

5.1.3 Output configuration

When an IO port is set to output mode (GPIO_MODER[MODERx] = 2'b01),

1. The output data register will be turned on；
2. Schmitt trigger input mode is activated；
3. The pull-up and pull-down control ports are configured according to the GPIO_PUPDR register；
4. Each AHB cycle, the IO data is refreshed once by the input register；
5. Each write cycle, the data from the output register is sent to IO；
6. When both open drain and open source outputs are turned off, it is a push-pull output.

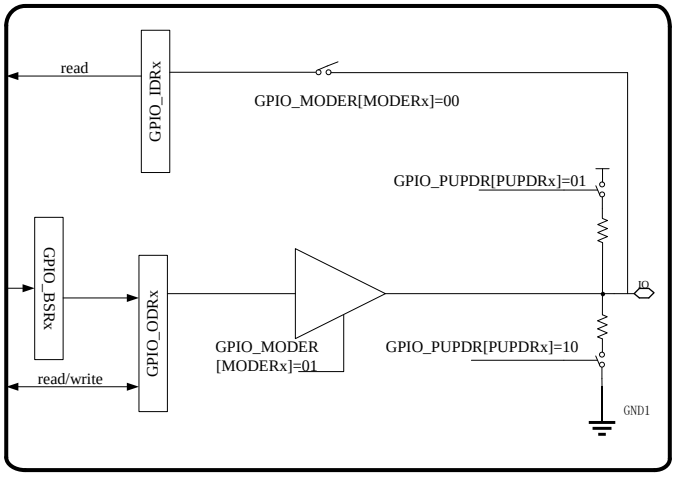


Figure5-2 GPIO output configuration schematic

5.1.4 Alternate function configuration

When an port is configured as a multiplexing function (GPIO_MODER[MODERx] = 2'b10),

1. The output data register is driven by the module port inside the chip；
2. The input and output directions of the ports are determined by control signals inside the module；
3. Schmitt trigger input mode is activated；
4. The pull-up and pull-down of a module is no longer controlled by GPIO_PUPDR, but is determined by the module to which it is connected.

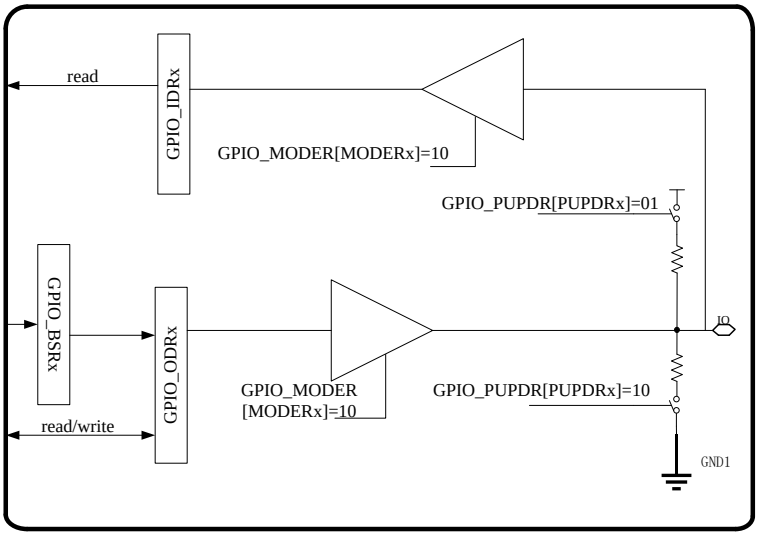


Figure5-3 GPIO alternate function configuration

5.1.5 Analog function configuration

When a IO is configured as an analog function (GPIO_MODER[MODERx] = 2'b11),

1. The output data register is turned off；
2. The Schmitt trigger will be turned off and the input data will always be 0.

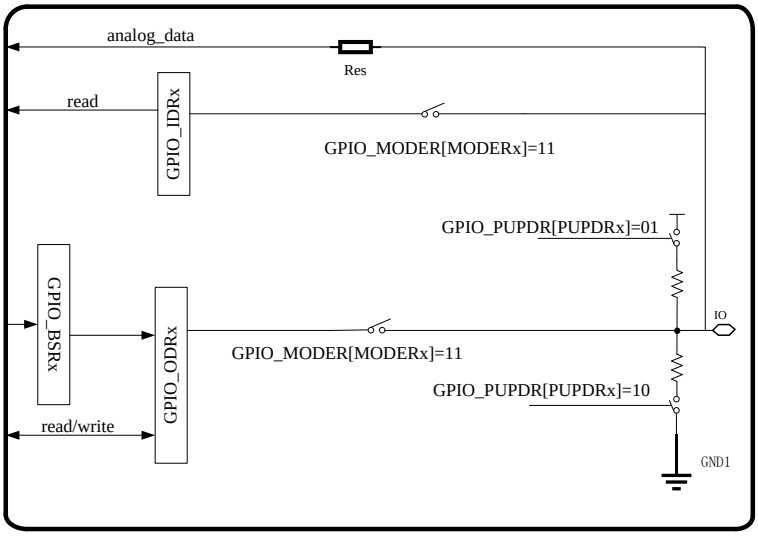


Figure5-4 GPIO analog function configuration

5.2 GPIOA register description

5.2.1 GPIOA Mode Control register（GPIOA_MODER）

Offset address:0x00

Reset value:0x3A00_000A

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MODER15		MODER14		MODER13		MODER12		MODER11		MODER10		MODER9		MODER8	
RW		RW		RW		RW		RW		RW		RW		RW	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODER7		MODER6		MODER5		MODER4		MODER3		MODER2		MODER1		MODER0	
RW		RW		RW		RW		RW		RW		RW		RW	

Bit	Marking	Functional description
31:30	MODER15	GPIOA15 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
29:28	MODER14	GPIOA14 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function

Bit	Marking	Functional description
27:26	MODER13	GPIOA13 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
25:24	MODER12	GPIOA12 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
23:22	MODER11	GPIOA11 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
21:20	MODER10	GPIOA10 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
19:18	MODER9	GPIOA9 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
17:16	MODER8	GPIOA8 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
15:14	MODER7	GPIOA7 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
13:12	MODER6	GPIOA6 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
11:10	MODER5	GPIOA5 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
9:8	MODER4	GPIOA4 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
7:6	MODER3	GPIOA3 port control bits 00：input mode；　01：output mode；

Bit	Marking	Functional description
		10：alternate function； 11：analog function
5:4	MODER2	GPIOA2 port control bits 00：input mode； 01：output mode； 10：alternate function； 11：analog function
3:2	MODER1	GPIOA1 port control bits 00：input mode； 01：output mode； 10：alternate function； 11：analog function
1:0	MODER0	GPIOA0 port control bits 00：input mode； 01：output mode； 10：alternate function； 11：analog function

5.2.2 GPIOA Output control register（GPIOA_OTYPER）

Offset address:0x04

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31	OS15	GPIOA15 open source control bit 0：open source function is off； 1：open source function turns on
30	OS14	GPIOA14 open source control bit 0：open source function is off； 1：open source function turns on
29	OS13	GPIOA13 open source control bit 0：open source function is off； 1：open source function turns on
28	OS12	GPIOA12 open source control bit 0：open source function is off； 1：open source function turns on
27	OS11	GPIOA11 open source control bit

Bit	Marking	Functional description
		0：open source function is off； 1：open source function turns on
26	OS10	GPIOA10 open source control bit 0：open source function is off； 1：open source function turns on
25	OS9	GPIOA9 open source control bit 0：open source function is off； 1：open source function turns on
24	OS8	GPIOA8 open source control bit 0：open source function is off； 1：open source function turns on
23	OS7	GPIOA7 open source control bit 0：open source function is off； 1：open source function turns on
22	OS6	GPIOA6 open source control bit 0：open source function is off； 1：open source function turns on
21	OS5	GPIOA5 open source control bit 0：open source function is off； 1：open source function turns on
20	OS4	GPIOA4 open source control bit 0：open source function is off； 1：open source function turns on
19	OS3	GPIOA3 open source control bit 0：open source function is off； 1：open source function turns on
18	OS2	GPIOA2 open source control bit 0：open source function is off； 1：open source function turns on
17	OS1	GPIOA1 open source control bit 0：open source function is off； 1：open source function turns on
16	OS0	GPIOA0 open source control bit 0：open source function is off； 1：open source function turns on
15	OD15	GPIOA15 open drain control bit 0：open drain function is off； 1：open drain function turns on
14	OD14	GPIOA14 open drain control bit 0：open drain function is off； 1：open drain function turns on.
13	OD13	GPIOA13 open drain control bit 0：open drain function is off； 1：open drain function turns on
12	OD12	GPIOA12 open drain control bit 0：open drain function is off； 1：open drain function turns on.
11	OD11	GPIOA11 open drain control bit

Bit	Marking	Functional description
		0：open drain function is off； 1：open drain function turns on
10	OD10	GPIOA10 open drain control bit 0：open drain function is off； 1：open drain function turns on.
9	OD9	GPIOA9 open drain control bit 0：open drain function is off； 1：open drain function turns on
8	OD8	GPIOA8 open drain control bit 0：open drain function is off； 1：open drain function turns on.
7	OD7	GPIOA7 open drain control bit 0：open drain function is off； 1：open drain function turns on
6	OD6	GPIOA6 open drain control bit 0：open drain function is off； 1：open drain function turns on.
5	OD5	GPIOA5 open drain control bit 0：open drain function is off； 1：open drain function turns on
4	OD4	GPIOA4 open drain control bit 0：open drain function is off； 1：open drain function turns on.
3	OD3	GPIOA3 open drain control bit 0：open drain function is off； 1：open drain function turns on
2	OD2	GPIOA2 open drain control bit 0：open drain function is off； 1：open drain function turns on.
1	OD1	GPIOA1 open drain control bit 0：open drain function is off； 1：open drain function turns on
0	OD0	GPIOA0 open drain control bit 0：open drain function is off； 1：open drain function turns on.

Note: When both the open drain and open source output functions are turned off, the IO is set to push-pull output.

5.2.3 GPIOA input mode control register（GPIOA_ITYPER）

Offset address:0x08

Reset value:0x0000_0000

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

CSx
RW

Bit	Marking	Functional description
15	CS15	Setting GPIOA15 input mode 0：Schmitt trigger mode；　1：CMOS input mode
14	CS14	Setting GPIOA14 input mode 0：Schmitt trigger mode；　1：CMOS input mode
13	CS13	Setting GPIOA13 input mode 0：Schmitt trigger mode；　1：CMOS input mode
12	CS12	Setting GPIOA12 input mode 0：Schmitt trigger mode；　1：CMOS input mode
11	CS11	Setting GPIOA11 input mode 0：Schmitt trigger mode；　1：CMOS input mode
10	CS10	Setting GPIOA10 input mode 0：Schmitt trigger mode；　1：CMOS input mode
9	CS9	Setting GPIOA9 input mode 0：Schmitt trigger mode；　1：CMOS input mode
8	CS8	Setting GPIOA8 input mode 0：Schmitt trigger mode；　1：CMOS input mode
7	CS7	Setting GPIOA7 input mode 0：Schmitt trigger mode；　1：CMOS input mode
6	CS6	Setting GPIOA6 input mode 0：Schmitt trigger mode；　1：CMOS input mode
5	CS5	Setting GPIOA5 input mode 0：Schmitt trigger mode；　1：CMOS input mode
4	CS4	Setting GPIOA4 input mode 0：Schmitt trigger mode；　1：CMOS input mode
3	CS3	Setting GPIOA3 input mode 0：Schmitt trigger mode；　1：CMOS input mode
2	CS2	Setting GPIOA2 input mode 0：Schmitt trigger mode；　1：CMOS input mode
1	CS1	Setting GPIOA1 input mode

Bit	Marking	Functional description
		0：Schmitt trigger mode； 1：CMOS input mode
0	CS0	Setting GPIOA0 input mode 0：Schmitt trigger mode； 1：CMOS input mode

5.2.4 GPIOA pull up/down control register（GPIOA_PUPDR）

Offset address:0x0C

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PUPDR15		PUPDR14		PUPDR13		PUPDR12		PUPDR11		PUPDR10		PUPDR9		PUPDR8	
RW		RW		RW		RW		RW		RW		RW		RW	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUPDR7		PUPDR6		PUPDR5		PUPDR4		PUPDR3		PUPDR2		PUPDR1		PUPDR0	
RW		RW		RW		RW		RW		RW		RW		RW	

Bit	Marking	Functional description
31:30	PUPDR15	GPIOA15 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
29:28	PUPDR14	GPIOA14 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
27:26	PUPDR13	GPIOA13 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
25:24	PUPDR12	GPIOA12 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
23:22	PUPDR11	GPIOA11 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
21:20	PUPDR10	GPIOA10 configured as pull-up or pull-down

Bit	Marking	Functional description
		00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
19:18	PUPDR9	GPIOA9 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
17:16	PUPDR8	GPIOA8 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
15:14	PUPDR7	GPIOA7 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
13:12	PUPDR6	GPIOA6 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
11:10	PUPDR5	GPIOA5 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
9:8	PUPDR4	GPIOA4 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
7:6	PUPDR3	GPIOA3 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
5:4	PUPDR2	GPIOA2 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
3:2	PUPDR1	GPIOA1 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
1:0	PUPDR0	GPIOA0 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit

5.2.5 GPIOA performance control register（GPIOA_SDR）

Offset address:0x10

Reset value:0x0000_0000

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

Bit	Marking	Functional description
31	SR15	Slew rate of GPIOA15 corresponding pad 1：high slew rate；　　0：low slew rate
30	SR14	Slew rate of GPIOA14 corresponding pad 1：high slew rate；　　0：low slew rate
29	SR13	Slew rate of GPIOA13 corresponding pad 1：high slew rate；　　0：low slew rate
28	SR12	Slew rate of GPIOA12 corresponding pad 1：high slew rate；　　0：low slew rate
27	SR11	Slew rate of GPIOA11 corresponding pad 1：high slew rate；　　0：low slew rate
26	SR10	Slew rate of GPIOA10 corresponding pad 1：high slew rate；　　0：low slew rate
25	SR9	Slew rate of GPIOA9 corresponding pad 1：high slew rate；　　0：low slew rate
24	SR8	Slew rate of GPIOA8 corresponding pad 1：high slew rate；　　0：low slew rate
23	SR7	Slew rate of GPIOA7 corresponding pad 1：high slew rate；　　0：low slew rate
22	SR6	Slew rate of GPIOA6corresponding pad 1：high slew rate；　　0：low slew rate
21	SR5	Slew rate of GPIOA5 corresponding pad

Bit	Marking	Functional description
		1：high slew rate；　　0：low slew rate
20	SR4	Slew rate of GPIOA4 corresponding pad 1：high slew rate；　　0：low slew rate
19	SR3	Slew rate of GPIOA3 corresponding pad 1：high slew rate；　　0：low slew rate
18	SR2	Slew rate of GPIOA2 corresponding pad 1：high slew rate；　　0：low slew rate
17	SR1	Slew rate of GPIOA1 corresponding pad 1：high slew rate；　　0：low slew rate
16	SR0	Slew rate of GPIOA0 corresponding pad 1：high slew rate；　　0：low slew rate
15	DR15	Driving capability of GPIOA15 corresponding pad 1：high driving capability；　　0：low driving capability
14	DR14	Driving capability of GPIOA14 corresponding pad 1：high driving capability；　　0：low driving capability
13	DR13	Driving capability of GPIOA13 corresponding pad 1：high driving capability；　　0：low driving capability
12	DR12	Driving capability of GPIOA12 corresponding pad 1：high driving capability；　　0：low driving capability
11	DR11	Driving capability of GPIOA11 corresponding pad 1：high driving capability；　　0：low driving capability
10	DR10	Driving capability of GPIOA10 corresponding pad 1：high driving capability；　　0：low driving capability
9	DR9	Driving capability of GPIOA9 corresponding pad 1：high driving capability；　　0：low driving capability
8	DR8	Driving capability of GPIOA8 corresponding pad 1：high driving capability；　　0：low driving capability
7	DR7	Driving capability of GPIOA7 corresponding pad 1：high driving capability；　　0：low driving capability
6	DR6	Driving capability of GPIOA6 corresponding pad 1：high driving capability；　　0：low driving capability
5	DR5	Driving capability of GPIOA5 corresponding pad

Bit	Marking	Functional description
		1：high driving capability；　　0：low driving capability
4	DR4	Driving capability of GPIOA4 corresponding pad 1：high driving capability；　　0：low driving capability
3	DR3	Driving capability of GPIOA3 corresponding pad 1：high driving capability；　　0：low driving capability
2	DR2	Driving capability of GPIOA2 corresponding pad 1：high driving capability；　　0：low driving capability
1	DR1	Driving capability of GPIOA1 corresponding pad 1：high driving capability；　　0：low driving capability
0	DR0	Driving capability of GPIOA0 corresponding pad 1：high driving capability；　　0：low driving capability

5.2.6 GPIOA interrupt mode register（GPIOA_LPMR）

Offset address:0x14

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LPMR15		LPMR14		LPMR13		LPMR12		LPMR11		LPMR10		LPMR9		LPMR8	
W		W		W		W		W		W		W		W	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LPMR7		LPMR6		LPMR5		LPMR4		LPMR3		LPMR2		LPMR1		LPMR0	
W		W		W		W		W		W		W		W	

Bit	Marking	Functional description
31:0	LPMRx	GPIOAx external interrupt detect control bit for PAD LPMRx[2x + 1: 2x]: MODE1, MODE0 00: high level detection;　　01: falling edge detection; 10: rising edge detection;　　11: low level detection.

5.2.7 GPIOA external interrupt collect enable register（GPIOA_INTER）

Offset address:0x18

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
15:0	INT_EN	GPIOAx external interrupt detect enable bit for PAD 0: No external interrupt input from the PAD is received; 1: Receive external interrupt input from PAD

5.2.8 GPIOA input data register（GPIOA_IDR）

Offset address:0x1C

Reset value:0x0000_0000

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
IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8
R	R	R	R	R	R	R	R

7	6	5	4	3	2	1	0
IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
R	R	R	R	R	R	R	R

Bit	Marking	Functional description
15:0	IDRx	The input value of GPIOAx

5.2.9 GPIOA output data register（GPIOA_ODR）

Offset address:0x20

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
15:0	ODRx	GPIOAx output data register, the ODRx bits can be individually set and/or reset by writing to the GPIOAx_BSR register. 1：output high level，if configured as open drain, it will need external or internal pull-up to output high level 0：output low level，if configured as open source, it will need external or internal pull-down to output low level

5.2.10 GPIOA set/reset register（GPIOA_BSR）

Offset address:0x24

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Marking	Functional description
31:16	BRx	Reset bit y of port x (y = 0...15)

Bit	Marking	Functional description
		0: No action on the corresponding ODx 1: Resets the corresponding ODx bit Note: If both BSx and BRx are set, BSx has priority.
15:0	BSx	Set bit y of port x (y = 0...15) 0: No action on the corresponding ODx bit 1: Sets the corresponding ODx bit

5.2.11 GPIOA alternate function high register（GPIOA_AFRH）

Offset address:0x28

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEH15[3:0]				AFSEH14[3:0]				AFSEH13[3:0]				AFSEH12[3:0]			
RW				RW				RW				RW			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFSEH11[3:0]				AFSEH10[3:0]				AFSEH9[3:0]				AFSEH8[3:0]			
RW				RW				RW				RW			

Bit	Marking	Functional description
31:28	AFSEH15[3:0]	Alternate function selection for port GPIOA15 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
27:24	AFSEH14[3:0]	Alternate function selection for port GPIOA14 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
23:20	AFSEH13[3:0]	Alternate function selection for port GPIOA13 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
19:16	AFSEH12[3:0]	Alternate function selection for port GPIOA12 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
15:12	AFSEH11[3:0]	Alternate function selection for port GPIOA11

Bit	Marking	Functional description
		0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
11:8	AFSEH10[3:0]	Alternate function selection for port GPIOA10 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
7:4	AFSEH9[3:0]	Alternate function selection for port GPIOA9 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
3:0	AFSEH8[3:0]	Alternate function selection for port GPIOA8 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3

5.2.12 GPIOA alternate function low register（GPIOA_AFRL）

Offset address:0x2C

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEL7[3:0]				AFSEL6[3:0]				AFSEL5[3:0]				AFSEL4[3:0]			
RW				RW				RW				RW			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFSEL3[3:0]				AFSEL2[3:0]				AFSEL1[3:0]				AFSEL0[3:0]			
RW				RW				RW				RW			

Bit	Marking	Functional description
31:28	AFSEL7[3:0]	Alternate function selection for port GPIOA7 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
27:24	AFSEL6[3:0]	Alternate function selection for port GPIOA6 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
23:20	AFSEL5[3:0]	Alternate function selection for port GPIOA5 0000：AF0； 0001：AF1；

Bit	Marking	Functional description
		0010：AF2； 0011：AF3
19:16	AFSEL4[3:0]	Alternate function selection for port GPIOA4 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
15:12	AFSEL3[3:0]	Alternate function selection for port GPIOA3 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
11:8	AFSEL2[3:0]	Alternate function selection for port GPIOA2 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
7:4	AFSEL1[3:0]	Alternate function selection for port GPIOA1 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
3:0	AFSEL0[3:0]	Alternate function selection for port GPIOA0 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3

5.3 GPIOA register mapping

GPIOA register list

Base address: 0x3000_0200

Register	Offset address	Description
GPIOA_MODER	0x00	GPIOA Mode control register
GPIOA_OTYPER	0x04	GPIOA Output control register
GPIOA_ITYPER	0x08	GPIOA input mode control register
GPIOA_PUPDR	0x0C	GPIOA pull up/down control register
GPIOA_SDR	0x10	GPIOA performance control register
GPIOA_LPMR	0x14	GPIOA interrupt mode register
GPIOA_INTER	0x18	GPIOA external interrupt collect enable control register
GPIOA_IDR	0x1C	GPIOA input data register
GPIOA_ODR	0x20	GPIOA output data register
GPIOA BSR	0x24	GPIOA set/reset register

GPIOA_AFRH	0x28	GPIOA alternate function high register
GPIOA_AFRL	0x2C	GPIOA alternate function low register

5.4 GPIOB register description

5.4.1 GPIOB Mode Control register（GPIOB_MODER）

Offset address:0x00

Reset value:0x0F00_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				MODER13		MODER12		MODER11		MODER10		MODER9		MODER8	
				RW		RW		RW		RW		RW		RW	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MODER7		MODER6		MODER5		MODER4		MODER3		MODER2		MODER1		MODER0	
RW		RW		RW		RW		RW		RW		RW		RW	

Bit	Marking	Functional description
31:28	Reserved	Reserved bit
27:26	MODER13	GPIOB13 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
25:24	MODER12	GPIOB12 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
23:22	MODER11	GPIOB11 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
21:20	MODER10	GPIOB10 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function
19:18	MODER9	GPIOB9 port control bits 00：input mode；　01：output mode； 10：alternate function；　11：analog function

Bit	Marking	Functional description
17:16	MODER8	GPIOB8 port control bits 00：input mode；01：output mode； 10：alternate function；11：analog function
15:14	MODER7	GPIOB7 port control bits 00：input mode；01：output mode； 10：alternate function；11：analog function
13:12	MODER6	GPIOB6 port control bits 00：input mode；01：output mode； 10：alternate function；11：analog function
11:10	MODER5	GPIOB5 port control bits 00：input mode；01：output mode； 10：alternate function；11：analog function
9:8	MODER4	GPIOB4 port control bits 00：input mode；01：output mode； 10：alternate function；11：analog function
7:6	MODER3	GPIOB3 port control bits 00：input mode；01：output mode； 10：alternate function；11：analog function
5:4	MODER2	GPIOB2 port control bits 00：input mode；01：output mode； 10：alternate function；11：analog function
3:2	MODER1	GPIOB1 port control bits 00：input mode；01：output mode； 10：alternate function；11：analog function
1:0	MODER0	GPIOB0 port control bits 00：input mode；01：output mode； 10：alternate function；11：analog function

5.4.2 GPIOB Output control register（GPIOB_OTYPER）

Offset address:0x04

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:30	Reserved	Reserved bit
29	OS13	GPIOB13 open source control bit 0：open source function is off；1：open source function turns on
28	OS12	GPIOB12 open source control bit 0：open source function is off；1：open source function turns on
27	OS11	GPIOB11 open source control bit 0：open source function is off；1：open source function turns on
26	OS10	GPIOB10 open source control bit 0：open source function is off；1：open source function turns on
25	OS9	GPIOB9 open source control bit 0：open source function is off；1：open source function turns on
24	OS8	GPIOB8 open source control bit 0：open source function is off；1：open source function turns on
23	OS7	GPIOB7 open source control bit 0：open source function is off；1：open source function turns on
22	OS6	GPIOB6 open source control bit 0：open source function is off；1：open source function turns on
21	OS5	GPIOB5 open source control bit 0：open source function is off；1：open source function turns on
20	OS4	GPIOB4 open source control bit 0：open source function is off；1：open source function turns on
19	OS3	GPIOB3 open source control bit 0：open source function is off；1：open source function turns on
18	OS2	GPIOB2 open source control bit 0：open source function is off；1：open source function turns on

Bit	Marking	Functional description
17	OS1	GPIOB1 open source control bit 0：open source function is off；　1：open source function turns on
16	OS0	GPIOB0 open source control bit 0：open source function is off；　1：open source function turns on
15:14	Reserved	Reserved bit
13	OD13	GPIOB13 open drain control bit 0：open drain function is off；　1：open drain function turns on
12	OD12	GPIOB12 open drain control bit 0：open drain function is off；　1：open drain function turns on.
11	OD11	GPIOB11 open drain control bit 0：open drain function is off；　1：open drain function turns on
10	OD10	GPIOB10 open drain control bit 0：open drain function is off；　1：open drain function turns on.
9	OD9	GPIOB9 open drain control bit 0：open drain function is off；　1：open drain function turns on
8	OD8	GPIOB8 open drain control bit 0：open drain function is off；　1：open drain function turns on.
7	OD7	GPIOB7 open drain control bit 0：open drain function is off；　1：open drain function turns on
6	OD6	GPIOB6 open drain control bit 0：open drain function is off；　1：open drain function turns on.
5	OD5	GPIOB5 open drain control bit 0：open drain function is off；　1：open drain function turns on
4	OD4	GPIOB4 open drain control bit 0：open drain function is off；　1：open drain function turns on.
3	OD3	GPIOB3 open drain control bit 0：open drain function is off；　1：open drain function turns on
2	OD2	GPIOB2 open drain control bit 0：open drain function is off；　1：open drain function turns on.
1	OD1	GPIOB1 open drain control bit 0：open drain function is off；　1：open drain function turns on
0	OD0	GPIOA0 open drain control bit

Bit	Marking	Functional description
		0：open drain function is off； 1：open drain function turns on.

Note: When both the open drain and open source output functions are turned off, the IO is set to push-pull output.

5.4.3 GPIOB input mode control register（GPIOB_ITYPER）

Offset address:0x08

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
15:14	Reserved	Reserved bit
13	CS13	Setting GPIOB13 input mode 0：Schmitt trigger mode； 1：CMOS input mode
12	CS12	Setting GPIOB12 input mode 0：Schmitt trigger mode； 1：CMOS input mode
11	CS11	Setting GPIOB11 input mode 0：Schmitt trigger mode； 1：CMOS input mode
10	CS10	Setting GPIOB10 input mode 0：Schmitt trigger mode； 1：CMOS input mode
9	CS9	Setting GPIOB9 input mode 0：Schmitt trigger mode； 1：CMOS input mode
8	CS8	Setting GPIOB8 input mode 0：Schmitt trigger mode； 1：CMOS input mode
7	CS7	Setting GPIOB7 input mode 0：Schmitt trigger mode； 1：CMOS input mode
6	CS6	Setting GPIOB6 input mode 0：Schmitt trigger mode； 1：CMOS input mode
5	CS5	Setting GPIOB5 input mode

Bit	Marking	Functional description
		0：Schmitt trigger mode； 1：CMOS input mode
4	CS4	Setting GPIOB4 input mode 0：Schmitt trigger mode； 1：CMOS input mode
3	CS3	Setting GPIOB3 input mode 0：Schmitt trigger mode； 1：CMOS input mode
2	CS2	Setting GPIOB2 input mode 0：Schmitt trigger mode； 1：CMOS input mode
1	CS1	Setting GPIOB1 input mode 0：Schmitt trigger mode； 1：CMOS input mode
0	CS0	Setting GPIOB0 input mode 0：Schmitt trigger mode； 1：CMOS input mode

5.4.4 GPIOB pull up/down control register（GPIOB_PUPDR）

Offset address:0x0C

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				PUPDR13		PUPDR12		PUPDR11		PUPDR10		PUPDR9		PUPDR8	
				RW		RW		RW		RW		RW		RW	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUPDR7		PUPDR6		PUPDR5		PUPDR4		PUPDR3		PUPDR2		PUPDR1		PUPDR0	
RW		RW		RW		RW		RW		RW		RW		RW	

Bit	Marking	Functional description
31:28	Reserved	Reserved bit
27:26	PUPDR13	GPIOB13 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit
25:24	PUPDR12	GPIOB12 configured as pull-up or pull-down 00：not pull up or down； 01：pull up； 10：pull down； 11：Reserved bit

Bit	Marking	Functional description
23:22	PUPDR11	GPIOB11 configured as pull-up or pull-down 00：not pull up or down；　01：pull up； 10：pull down；　11：Reserved bit
21:20	PUPDR10	GPIOB10 configured as pull-up or pull-down 00：not pull up or down；　01：pull up； 10：pull down；　11：Reserved bit
19:18	PUPDR9	GPIOB9 configured as pull-up or pull-down 00：not pull up or down；　01：pull up； 10：pull down；　11：Reserved bit
17:16	PUPDR8	GPIOB8 configured as pull-up or pull-down 00：not pull up or down；　01：pull up； 10：pull down；　11：Reserved bit
15:14	PUPDR7	GPIOB7 configured as pull-up or pull-down 00：not pull up or down；　01：pull up； 10：pull down；　11：Reserved bit
13:12	PUPDR6	GPIOB6 configured as pull-up or pull-down 00：not pull up or down；　01：pull up； 10：pull down；　11：Reserved bit
11:10	PUPDR5	GPIOB5 configured as pull-up or pull-down 00：not pull up or down；　01：pull up； 10：pull down；　11：Reserved bit
9:8	PUPDR4	GPIOB4 configured as pull-up or pull-down 00：not pull up or down；　01：pull up； 10：pull down；　11：Reserved bit
7:6	PUPDR3	GPIOB3 configured as pull-up or pull-down 00：not pull up or down；　01：pull up； 10：pull down；　11：Reserved bit
5:4	PUPDR2	GPIOB2 configured as pull-up or pull-down 00：not pull up or down；　01：pull up； 10：pull down；　11：Reserved bit
3:2	PUPDR1	GPIOB1 configured as pull-up or pull-down 00：not pull up or down；　01：pull up；

Bit	Marking	Functional description
		10：pull down；　　11：Reserved bit
1:0	PUPDR0	GPIOB0 configured as pull-up or pull-down 00：not pull up or down；　　01：pull up； 10：pull down；　　11：Reserved bit

5.4.5 GPIOB performance control register（GPIOB_SDR）

Offset address:0x10

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:30	Reserved	Reserved bit
29	SR13	Slew rate of GPIOB13 corresponding pad 1：high slew rate；　　0：low slew rate
28	SR12	Slew rate of GPIOB12 corresponding pad 1：high slew rate；　　0：low slew rate
27	SR11	Slew rate of GPIOB11 corresponding pad 1：high slew rate；　　0：low slew rate
26	SR10	Slew rate of GPIOB10 corresponding pad 1：high slew rate；　　0：low slew rate
25	SR9	Slew rate of GPIOB9 corresponding pad 1：high slew rate；　　0：low slew rate
24	SR8	Slew rate of GPIOB8 corresponding pad 1：high slew rate；　　0：low slew rate
23	SR7	Slew rate of GPIOB7 corresponding pad 1：high slew rate；　　0：low slew rate

Bit	Marking	Functional description
22	SR6	Slew rate of GPIOB6corresponding pad 1：high slew rate；　　0：low slew rate
21	SR5	Slew rate of GPIOB5 corresponding pad 1：high slew rate；　　0：low slew rate
20	SR4	Slew rate of GPIOB4 corresponding pad 1：high slew rate；　　0：low slew rate
19	SR3	Slew rate of GPIOB3 corresponding pad 1：high slew rate；　　0：low slew rate
18	SR2	Slew rate of GPIOB2 corresponding pad 1：high slew rate；　　0：low slew rate
17	SR1	Slew rate of GPIOB1 corresponding pad 1：high slew rate；　　0：low slew rate
16	SR0	Slew rate of GPIOB0 corresponding pad 1：high slew rate；　　0：low slew rate
15:14	Reserved	Reserved bit
13	DR13	Driving capability of GPIOB13 corresponding pad 1：high driving capability；　　0：low driving capability
12	DR12	Driving capability of GPIOB12 corresponding pad 1：high driving capability；　　0：low driving capability
11	DR11	Driving capability of GPIOB11 corresponding pad 1：high driving capability；　　0：low driving capability
10	DR10	Driving capability of GPIOB10 corresponding pad 1：high driving capability；　　0：low driving capability
9	DR9	Driving capability of GPIOB9 corresponding pad 1：high driving capability；　　0：low driving capability
8	DR8	Driving capability of GPIOB8 corresponding pad 1：high driving capability；　　0：low driving capability
7	DR7	Driving capability of GPIOB7 corresponding pad 1：high driving capability；　　0：low driving capability
6	DR6	Driving capability of GPIOB6 corresponding pad 1：high driving capability；　　0：low driving capability
5	DR5	Driving capability of GPIOB5 corresponding pad

Bit	Marking	Functional description
		1：high driving capability； 0：low driving capability
4	DR4	Driving capability of GPIOB4 corresponding pad 1：high driving capability； 0：low driving capability
3	DR3	Driving capability of GPIOB3 corresponding pad 1：high driving capability； 0：low driving capability
2	DR2	Driving capability of GPIOB2 corresponding pad 1：high driving capability； 0：low driving capability
1	DR1	Driving capability of GPIOB1 corresponding pad 1：high driving capability； 0：low driving capability
0	DR0	Driving capability of GPIOB0 corresponding pad 1：high driving capability； 0：low driving capability

5.4.6 GPIOB input data register（GPIOB_IDR）

Offset address:0x1C

Reset value:0x0000_0000

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		IDR13	IDR12	IDR11	IDR10	IDR9	IDR8
		R	R	R	R	R	R
7	6	5	4	3	2	1	0
IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
R	R	R	R	R	R	R	R

Bit	Marking	Functional description
13:0	IDRx	The input value of GPIOBx

5.4.7 GPIOB output data register（GPIOB_ODR）

Offset address:0x20

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
13:0	ODRx	GPIOBx output data register, the ODRx bits can be individually set and/or reset by writing to the GPIOBx_BSR register. 1：output high level，if configured as open drain, it will need external or internal pull-up to output high level 0：output low level，if configured as open source, it will need external or internal pull-down to output low level

5.4.8 GPIOB set/reset register（GPIOB_BSR）

Offset address:0x24

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
		W	W	W	W	W	W	W	W	W	W	W	W	W	W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
		W	W	W	W	W	W	W	W	W	W	W	W	W	W

Bit	Marking	Functional description
31:30	Reserved	Reserved bit
29:16	BRx	Reset bit y of port x (y = 0...13) 0: No action on the corresponding ODx 1: Resets the corresponding ODx bit Note: If both BSx and BRx are set, BSx has priority.
15:14	Reserved	Reserved bit
13:0	BSx	Set bit y of port x (y = 0...13) 0: No action on the corresponding ODx bit 1: Sets the corresponding ODx bit

5.4.9 GPIOB alternate function high register（GPIOB_AFRH）

Offset address:0x28

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								AFSEH13[3:0]				AFSEH12[3:0]			
								RW				RW			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFSEH11[3:0]				AFSEH10[3:0]				AFSEH9[3:0]				AFSEH8[3:0]			
RW				RW				RW				RW			

Bit	Marking	Functional description
31:24	Reserved	Reserved bit
23:20	AFSEH13[3:0]	Alternate function selection for port GPIOB13 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
19:16	AFSEH12[3:0]	Alternate function selection for port GPIOB12 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
15:12	AFSEH11[3:0]	Alternate function selection for port GPIOB11 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3

Bit	Marking	Functional description
11:8	AFSEH10[3:0]	Alternate function selection for port GPIOB10 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
7:4	AFSEH9[3:0]	Alternate function selection for port GPIOB9 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
3:0	AFSEH8[3:0]	Alternate function selection for port GPIOB8 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3

5.4.10 GPIOB alternate function low register（GPIOB_AFRL）

Offset address:0x2C

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFSEL7[3:0]				AFSEL6[3:0]				AFSEL5[3:0]				AFSEL4[3:0]			
RW				RW				RW				RW			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AFSEL3[3:0]				AFSEL2[3:0]				AFSEL1[3:0]				AFSEL0[3:0]			
RW				RW				RW				RW			

Bit	Marking	Functional description
31:28	AFSEL7[3:0]	Alternate function selection for port GPIOB7 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
27:24	AFSEL6[3:0]	Alternate function selection for port GPIOB6 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
23:20	AFSEL5[3:0]	Alternate function selection for port GPIOB5 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
19:16	AFSEL4[3:0]	Alternate function selection for port GPIOB4

Bit	Marking	Functional description
		0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
15:12	AFSEL3[3:0]	Alternate function selection for port GPIOB3 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
11:8	AFSEL2[3:0]	Alternate function selection for port GPIOB2 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
7:4	AFSEL1[3:0]	Alternate function selection for port GPIOB1 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
3:0	AFSEL0[3:0]	Alternate function selection for port GPIOB0 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3

5.5 GPIOB register mapping

GPIOB register list

Base address:0x3000_0240

Register	Offset address	Description
GPIOB_MODER	0x00	GPIOB Mode control register
GPIOB_OTYPER	0x04	GPIOB Output control register
GPIOB_ITYPER	0x08	GPIOB input mode control register
GPIOB_PUPDR	0x0C	GPIOB pull up/down control register
GPIOB_SDR	0x10	GPIOB performance control register
GPIOB_IDR	0x1C	GPIOB input data register
GPIOB_ODR	0x20	GPIOB output data register
GPIOB_BSR	0x24	GPIOB set/reset register
GPIOB_AFRH	0x28	GPIOB alternate function high register
GPIOB_AFRL	0x2C	GPIOB alternate function low register

6 Interrupt

6.1 Interrupt introduction

The MCU RISC-V core has a Core-Local Interrupt Controller (CLIC), whose interrupt response occupies 6 bus clock cycles. The CLIC can generate two types of interrupts defined by the standard RISC-V architecture: Machine Timer Interrupt, Machine Mode Software Interrupt and Machine Mode Interrupt. In addition to these two standard RISC-V architecture-defined interrupts, the CLIC defines 32 interrupt sources to be handled as Local Interrupts, which are used to connect to external devices and set the corresponding registers to accept input signals from the peripheral devices as interrupts. The processor can receive these 32 external device interrupts as well as the interrupts defined by the two standard RISC- V architectures mentioned above through the CLIC. Each interrupt has a corresponding Interrupt ID number, the source IDs of the above 32 interrupts are 16 ~ 47, among which 16 ~ 30, 32 ~ 35 and 40 ~ 47 have peripheral connections, and 31, 39 have no peripheral connection. More details see next table.

Table6-1 Core interrupt and Interrupt ID

ID（Dec）	Description
2 - 0	-
3	Machine Software Interrupt
6 - 4	-
7	RTC interrupt（Machine Timer Interrupt）
10 - 8	-
11	Machine External Interrupt
12	CLIC Software Interrupt
15 - 13	-
16	SPI1 Interrupt
17	SPI2 Interrupt
18	LV Interrupt
19	UART1(TX/RX) Interrupt
20	I2C waiting interrupt
21	I2C error interrupt

ID（Dec）	Description
22	Timer1 Break interrupt
23	Timer1 Updata interrupt
24	Timer1 Capture Compare interrupt
25	Timer1 Trigger and Commutation interrupt
26	Timer2 Break interrupt
27	Timer2 Updata interrupt
28	Timer2 Capture Compare interrupt
29	Timer2 Trigger and Commutation interrupt
30	ADC interrupt
31	-
32	WUP wake-up interrupt
33	UART2（TX/RX）interrupt
34	UART3（TX/RX）interrupt
35	UART4（TX/RX）interrupt
36	COMP1 interrupt
37	COMP2 interrupt
38	COMP3 interrupt
39	-
40	EXTI[0] external interrupt
41	EXTI[1] external interrupt
42	EXTI[2] external interrupt
43	EXTI[3] external interrupt
44	EXTI[4] external interrupt
45	EXTI[9:5] external interrupt
46	EXTI[15:10] external interrupt
47	RF detection interrupt

6.2 CLIC Registers

Base address：0x0280_0000

6.2.1 CLIC Interrupt Pending（clicintip）

Offset address:Interrupt ID（Hex）

Reset value:0x0000_0000

7	6	5	4	3	2	1	0
Reserved							clicintip
							RW

Bit	Marking	Functional description
7:1	Reserved	Reserved bit
0	clicintip	Indicates the pending status of the interrupt with the corresponding Interrupt ID When clicintip is set, the corresponding Interrupt ID is pending.

6.2.2 CLIC Interrupt Enable（clicintie）

Offset address:0x400 + Interrupt ID（Hex）

Reset value:0x0000_0000

7	6	5	4	3	2	1	0
Reserved							clicintie
							RW

Bit	Marking	Functional description
7:1	Reserved	Reserved bit
0	clicintie	Can be used to disable and enable the interrupt of the corresponding Interrupt ID 0: the corresponding Interrupt ID is disabled 1:the corresponding Interrupt ID is enabled

6.2.3 CLIC Interrupt Configuration（clicintcfg）

Offset address:0x800 + Interrupt ID（Hex）

Reset value:0x0000_0000

7	6	5	4	3	2	1	0
clicintcfg			Reserved				
RW							

Bit	Marking	Functional description
7:5	clicintcfg	sets the pre-emption level and priority of a given interrupt. There are a total of 2 bits in clicintcfg which specify how to encode a given interrupt's pre-emption level and/or priority. The actual number of bits which determine the preemption level is determined by nlbit in the cliccfg. If the value of nlbits in register cliccfg is less than 2, then the remaining least significant implemented bits are used to encode priorities within a given pre-emption level. If the value of nlbits in register cliccfg is set to 0, then all interrupt are treated as 255 and all 2 bits are used to set the priorities. This is shown in the following table： nlbits / Encoding / interrupt levels / priority 0 / 11111111 / 255 / configured by clicintcfg [2:1] 1 / x1111111[1] / 127 255 / configured by clicintcfg [1] 2 / xx111111[2] / 63 127 191 255 / 0 [1] x configured by clicintcfg[2]; [2] xx configured by clicintcfg[2:1]; Higher-level interrupts can preempt lower-level interrupts;In case where the interrupt level and priority are the same, the highest-numbered interrupt ID is taken first.
4:0	Reserved	Reserved bit

6.2.4 CLIC Configuration（cliccfg）

Offset address:0xC00

Reset value:0x0000_0000

7	6	5	4	3	2	1	0
Reserved	nmbits		nlbits				nvbits
	R		RW				RW

Bit	Marking	Functional description
7	Reserved	Reserved bit
6:5	nmbits	Not writable, read as 0
4:1	nlbits	Determines the number of bits in register clicintcfg that are used to encode the interrupt level and priority, see clicintcfg description.
0	nvbits	When set, selective hardware vectoring is enabled. If in CLIC Direct mode and nvbits =1, then selective interrupt vectoring is turned on. The least-significant implemented bit of clicintcfg (bit 5) controls the vectoring behavior of a given interrupt. When in CLIC Direct mode, and both nvbits and the relevant bit of clicintcfg are set to 1, then the interrupt is vectored using the vector table pointed to by the mtvt CSR. This allows some interrupts to all jump to a common base address held in mtvec CSR, while the others are vectored in hardware.

6.3 CLIC Registers Mapping

CLIC registers list

Base address：0x0200_0000

Register	Offset address	Description
msip	0x0000	Machine mode software interrupt pending register
mtimecmp	0x4000	Machine mode timer comparison register
mtime	0xBFF8	Machine mode timer register

CLIC registers list

Base address：0x0280_0000

Register	Offset address	Description
clicintip	0x000	CLIC interrupt pending register
clicintie	0x400	CLIC interrupt enable register
clicintcfg	0x800	CLIC interrupt configuration register
cliccfg	0xC00	CLIC configuration register

6.4 External interrupt（EXTI）

6.4.1 EXTI introduction

The external interrupt controller supports 17 external interrupts, each with status bits and independent trigger and mask settings.

The settings for enabling and masking external interrupt bits 1to 16 and mask and how to trigger them can be set in GPIOA_LPMR and GPIOA_INTER. EXTI[15:0] corresponds to GPIOA15 ~ 0.

Bit 17 is the interrupt of RF detection module. GPIOA15 needs to be set to analog mode to receive the RF detection signal from PA15. If the signal strength exceeds a specific value, an interrupt signal will be generated.

6.4.2 External interrupt input status register（EXTI_ISR）

Base address：0x3000_02C0

Offset address:0x00

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															exti_caw_irq
															r0_w1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
exti_isr[15:0]															
r0_w1															

Bit	Marking	Functional description
31:17	Reserved	Reserved bit
16	exti_caw_irq	Status of RF detection interrupt 1：RF interrupt has generated，write 1 can clear the interrupt
15:0	exti_isrx	Interrupt status of External Interrupt 1: Generate external interrupt for the corresponding bit, write 1 to the corresponding bit to clear the corresponding external interrupt Bit 0 registers the interrupt status of GPIOA0; Bit 1 registers the interrupt status of GPIOA1; ... Bit 15 registers the interrupt status of GPIOA15;

6.4.3 External interrupt input enable register（EXTI_IEN）

Base address:0x3000_02C0

Offset address:0x04

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:17	Reserved	Reserved bit
16	exti_ien	External RF detection interrupt input enable 1:enable RF detection interrupt
15:0	Reserved	Reserved bit

6.5 EXTI register mapping

EXTI registers list

Base address：0x3000_02C0

Register	Offset address	Description
EXTI_ISR	0x00	External interrupt input status register
EXTI_IEN	0x04	External interrupt input enable register

6.6 Interrupt operations

6.6.1 Entering or exiting an interrupt

When an interrupt occures and mstatus.MIE is 1 and the interrupt enable signal corresponding to the interrupt source is turned on:

a) The value of mstatus.MIE is copied to mcause.MPIE and then mstatus.MIE is cleared, effectively disabling interrupts;
b) The interrupt level is copied to the mcause.MPIL register;
c) The privilege mode prior to the interrupt is encoded in mstatus.MPP
d) The current PC is copied into the mepc register, and then pc is set to the value specified by mtvec as defined by the mtvec.MODE.

At this point, control is handed over to software in the interrupt handler with interrupts disabled. Interrupts can be re-enabled by explicitly setting mstatus.MIE or by executing an MRET instruction to exit the handler. When an MRET instruction is executed, the following occurs:

a) The privilege mode is set to the value encoded in mstatus.MPP.
b) When in CLIC mode, the interrupt level is set to the value encoded in mcause.MPIL.
c) The global interrupt enable, mstatus.MIE, is set to the value of mcause.MPIE.
d) The pc is set to the value of mepc.

At this point control is handed over to software. The Control and Status Registers involved in handling RISC-V interrupts are described in Section0

6.6.2 Interrupt Levels and Priorities

At any time, a hart is running in some privilege mode with some interrupt level. The hart's current interrupt level is made visible in the mintstatus register.However, the current privilege mode is not visible to software running on a hart.

The CLIC architecture supports pre-emption of up to 256 interrupt levels for each privilege mode, where higher-numbered interrupt levels can preempt lower-numbered interrupt levels. Interrupt level 0 corresponds to regular execution outside of an interrupt handler. The CLIC also supports programmable priorities within a given level which are used to prioritize among interrupts pending-and-enabled at the same interrupt level. The highest-priority interrupt at a given interrupt level is taken first. In case there are multiple pending-and-enabled interrupts at the same highest priority, the highest-numbered interrupt ID is taken first.

The number of available pre-emption levels, and priorities within each level, is determined by the number of configuration bits in the CLIC's clicintcfg register and the value of the CLIC's cliccfg.nlbits register.

6.7 Interrupt Control Status Registers

6.7.1 Machine Status Registe（mstatus）

12	11	10	9	8	7	6	5	4	3	2	1	0
MPP		Reserved			MPIE	Reserved			MIE	Reserved		
RW					RW				RW			

Bit	Marking	Functional description
12:11	MPP	Machine Previous Privilege Mode
10:8	Reserved	Reserved bit
7	MPIE	Machine Previous Interrupt Enable
6:4	Reserved	Reserved bit
3	MIE	Machine Interrupt Enable.Interrupts are enabled by setting the MIE bit, if MIE is set to 0 will not enter interrupt processing.
2:0	Reserved	Reserved bit

Interrupts are enabled by setting the MIE bit in mstatus and by enabling the desired individual interrupt in the mie register.

Note that when operating in CLIC mode, mstatus.MPP and mstatus.MPIE are accessible in the mcause register.

6.7.2 Machine Trap Vector（mtvec）

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

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

Bit	Marking	Functional description
31:2	BASE	Interrupt Vector Base Address. Requires 64-byte alignment.
1:0	MODE	MODE Sets the interrupt processing mode

		0x0：Direct mode，all synchronous exceptions and asynchronous interrupts trap set pc to BASE 0x1：Vectored mdoe, synchronous exceptions set pc to BASE, Asynchronous interrupts set pc to BASE + 4 ×mcause.EXCCODE. 0x2：CLIC Direct mdoe，all synchronous exceptions and asynchronous interrupts trap set pc to BASE； 0x3：CLIC Vectored mdoe, Asynchronous interrupts set pc to the address in the vector table located at mtvt + 4 × mcause.EXCCODE, synchronous exceptions set pc to BASE Note: In non-CLIC mode, only processing software, timers and external interrupts are supported

1) Mode Direct：When operating in direct mode all synchronous exceptions and asynchronous interrupts trap to the mtvec.BASE address. Inside the trap handler, software must read the mcause register to determine what triggered the trap；
2) Mode Vectored: While operating in vectored mode, interrupts set the pc to mtvec.BASE + 4 × exception code. For example, if a machine timer interrupt is taken, the pc is set to mtvec.BASE + 0x1C. Typically, the trap vector table is populated with jump instructions to transfer control to interrupt-specific trap handlers. In vectored interrupt mode, BASE must be 64-byte aligned.
3) Mode CLIC Direct: In CLIC Direct mode the processor jumps to the trap handler address set by mtvec. In this mode, BASE must be 64-byte aligned.
4) Mode CLIC Vectored:In this mode, the processor switches to the handler's privilege mode and sets the hardware vectoring bit mcause.MINHV, and then fetches address mtvt + 4 × mcause.EXCCODE. If the fetch is successful, the processor clears the low bit of the handler address, sets the PC to this handler address, then clears mcause.MINHV.

Synchronous exceptions always trap to mtvec.BASE in machine mode.

6.7.3 Machine Interrupt Enable（mie）

31	30	29	28	27	26	25	24	23	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				MEIE	Reserved			MTIE	Reserved			MSIE	Reserved		
				RW				RW				RW			

Individual interrupts are enabled by setting the appropriate bit in the mie register.

Bit	Marking	Functional description
31:12	Reserved	Reserved bit
11	MEIE	Machine Software Interrupt Enable
10:8	Reserved	Reserved bit
7	MTIE	Machine Timer Interrupt Enable
6:4	Reserved	Reserved bit
3	MSIE	Machine External Interrupt Enable
2:0	Reserved	Reserved bit

When in either of the CLIC modes, the mie register is hardwired to zero and individual interrupt enables are controlled by the clicintie[i].

6.7.4 Machine Interrupt Pending（mip）

31	30	29	28	27	26	25	24	23	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				MEIP	Reserved			MTIP	Reserved			MSIP	Reserved		
				RO				RO				RO			

The machine interrupt pending (mip) register indicates which interrupts are currently pending.

Bit	Marking	Functional description
31:12	Reserved	Reserved bit

11	MEIP	Machine External Interrupt Pending
10:8	Reserved	Reserved bit
7	MTIP	Machine Timer Interrupt Pending
6:4	Reserved	Reserved bit
3	MSIP	Machine External Interrupt Pending
2:0	Reserved	Reserved bit

In CLIC mode, the mip register is hardwired to zero and individual interrupt enables are controlled by the clicintip[i].

6.7.5 Machine Cause（mcause）

When a trap is taken in machine mode, mcause is written with a code indicating the event that caused the trap. When the event that caused the trap is an interrupt, the most-significant bit of mcause is set to 1, and the least-significant bits indicate the interrupt number. For example, a Machine Timer Interrupt causes mcause to be set to 0x8000_0007. mcause is also used to indicate the cause of synchronous exceptions, in which case the most-significant bit of mcause is set to 0.

In CLIC mode, mcasue is extended to record more information.

31	30	29 28	27	26 25 24	23 22 21 20 19 18 17 16
Interrupt	MINHV	MPP	MPIE	Reserved	MPIL
WARL	WLRL	WLRL	WLRL		WLRL

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

Bit	Marking	Functional description
31	Interrupt	1 if the trap was caused by an interrupt; 0 otherwise.
30	MINHV	CLIC mode only.
29:28	MPP	Previous interrupt privilege mode, same as mstatus.mpp. CLIC mode only
27	MPIE	Previous interrupt enable, same as mstatus.mpie. CLIC mode only
26:24	Reserved	Reserved bit
23:16	MPIL	Previous interrupt level. CLIC mode only.

15:10	Reserved	Reserved bit
9:0	Exception Code	A code identifying the last exception.

Interrupt Exception Codes		
interrupt	Exception Code	Description
1	0 – 2	Reserved
1	3	Machine software interrupt
1	4 – 6	Reserved
1	7	Machine timer interrupt
1	8 – 10	Reserved
1	11	Machine external interrupt
1	12	CLIC Software Interrupt Pending
1	13 – 15	Reserved
1	16	CLIC Local Interrupt 0
1	17	CLIC Local Interrupt 1
1	18 – 31	…
1	48	CLIC Local Interrupt 32
0	0	Instruction address misaligned
0	1	Instruction access fault
0	2	Illegal instruction
0	3	Breakpoint
0	4	Load address misaligned
0	5	Load access fault
0	6	Store/AMO address misaligned
0	7	Store/AMO access fault
0	8 – 10	Reserved
0	11	Environment call from M-mode
0	⩾ 12	Reserved

6.7.6 Machine Trap Vector Table（mtvt）

The mtvt register holds the base address of the trap vector table. mtvt must be 64-

byte aligned and values other than 0 in the low 6 bits of mtvt are reserved.

In CLIC mode, mcasue is extended to record more information.

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

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

Bit	Marking	Functional description
31:6	BASE	Base address of the CLIC Vector Table
5:0	Reserved	Reserved bit

6.7.7 Handler Address and Interrupt-Enable（mnxti）

The mnxti CSR can be used by software to service the next horizontal interrupt when it has greater level than the saved interrupt context (held in mcause.PIL), without incuring the full cost of an interrupt pipeline flush and context save/restore. The mnxti CSR is designed to be accessed using CSRRSI/CSRRCI instructions, where the value read is a pointer to an entry in the trap handler table and the write back updates the interrupt-enable status. A read of the mnxti CSR returns either zero, indicating there is no suitable interrupt to service, or the address of the entry in the trap handler table for software trap vectoring. If the CSR instruction that acccesses mnxti includes a write, the mstatus CSR is the one used for the read-modify-write portion of the operation, while the exception code in mcause and the mintstatus register’s mil field can also be updated with the new interrupt level.

The mnxti CSR is intended to be used inside an interrupt handler after an initial interrupt has been taken and mcause and mepc registers updated with the interrupted context and the id of the interrupt.

6.7.8 Machine Interrupt Status（mintstatus）

Mintstatus holds the active interrupt level for each supported privilege mode.

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

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

Bit	Marking	Functional description
31:24	MIL	Active Machine Mode Interrupt Level
23:0	Reserved	Reserved bit

6.7.9 Scratch register for machine trap handlers（mscratch）

In CLIC mode, mcasue is extended to record more information.

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

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

Bit	Marking	Functional description
31:0	mscratch	It is used to hold a pointer to the context space local to the machine-mode hardware thread and is exchanged with a user register at the entry point of an M-mode self-trapping handler function.

6.7.10 Machine exception program counter（mepc）

The mepc register can never hold a pc value that results in an instruction address non-alignment exception.

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

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

Bit	Marking	Functional description
31:0	mepc	Address of the instruction that generated the exception pc value

6.7.11 Machine bad address or instruction（mtval）

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

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

Bit	Marking	Functional description
31:0	mtval	Used to store the instruction or address where the exception occurred

6.8 Interrupt status register mapping

Interrupt state register list

Register	Offset address	Description
mstatus	0x300	Machine Status Register
mie	0x304	Machine Interrupt Enable Register
mtvec	0x305	Machine Trap Vector Register
mtvt	0x307	Machine Trap Vector Table Register
mepc	0x341	Machine exception PC register
mcause	0x342	Machine Cause Register
mtval	0x343	Machine bad address or instruction Register
mip	0x344	Machine Interrupt Pending Register
mnxti	0x345	Handler Address and Interrupt-Enable Register
mintstatus	0x346	Machine Interrupt Status Register
mscratch	0x340	Scratch register for machine trap handlers Register

Note: The CSR registers defined in the RISC-V architecture need to be accessed using special CSR instructions. If you need to use the CSR registers in a C/C++ program, you can only use inline assembly (CSR instructions) to operate on the CSR registers. The following is an example of calling the RISC-V's CSR read or write assembly instruction to access the CSR registers in C. The code is as follows:

```
#define read_csr(reg) ({ unsigned long __tmp; \
  asm volatile ("csrr %0, " #reg : "=r"(__tmp)); \
  __tmp; })
```

Define the above macro and call it directly in C to read the value of the CSR register, such as C "value=read_csr(mstatus)" is equivalent to reading the value of the mstatus register and assigning it to the variable value.

7 Real-time clock（RTC）

7.1 RTC introduction

The RTC is usually run at a fixed low speed clock, so it can provide time reference and it is called a real-time clock. The real-time clock is an independent timer, RTC module, which has a continuous counting counter, which can provide the function of timing under the corresponding software configuration.

The RTC is also called the Machine Timers, which is 64-bit wide and has two registers: mtime and mtimecmp. Mtime stores the count value of RTC, and mtimecmp stores the comparison value of RTC. When mtime ≥ mtimecmp, the RTC module generates a timing interrupt, also known as a Machine Timer Interrupt.

When the RTC clock source is 3K or RCOSC, the clock accuracy is determined by 3K and RCOSC.

Note: The clock frequency of the RTC must be less than the frequency after dividing the MCU core clock by 2, which can be modified by software.

7.2 Register description

Base address：0x0200_0000

7.2.1 Machine mode timer register（mtime）

Offset address:0xBFF8

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

Bit	Marking	Functional description
31:0	mtime_lo	Reflects the low 32-bit count of the current timer

Offset address:0xBFFF

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

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

Bit	Marking	Functional description
31:0	mtime_hi	Reflects the high 32-bit count of the current timer

7.2.2 Machine mode timer compare value register（mtimecmp）

Offset address:0x4000

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

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

位	Marking	Functional description
31:0	mtimecmp_lo	Reflects the low 32-bit count of the mtimecmp Timer interrupts are triggered when the memory-mapped register mtime is greater than or equal to the global timebase register mtimecmp. The timer interrupt is remain high until software rewrites the value of the mtimecmp register so that its comparison value is greater than the value in mtime, thus clearing the timer interrupt. *Note: This interrupt is enabled when MTIE is set in the mie register.*

Offset address:0x4004

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

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

位	Marking	Functional description
31:0	mtimecmp_hi	Reflects the high 32-bit count of the mtimecmp

7.3 Register Mapping

RTC register list

Base address：0x0200_0000

Register	Offset address	Description
mtime	0xBFF8	Machine Mode Timer Register
mtimecmp	0x4000	Machine Mode Timer Comparison Value Register

8 WDG

8.1 Individual watch dog(IWDG)

8.1.1 Introduction

The Independent Watchdog Dog (IWDG) is driven by a low-speed clock 3K, so it remains active even if the master clock fails. The Independent Watchdog is best suited for applications that require the watchdog to be independent of the main program, to be able to work completely independently, and to have very low requirements for time accuracy. The watchdog can be powered on or started by the user program. Powering on to start the watchdog requires seeting it during program download.

8.1.1.1 Main features

- Once the watchdog is turned on, the counter will always run;
- Under independent watchdog operating conditions, a reset signal is generated when the counter reaches 0x00000;
- In all low-power modes, the independent watchdog's counter still operates and a reset signal is generated when the counter reaches 0x00000;
- The IWDG can be turned on in two ways, by writing the IWDG register or by setting it when the FLASH downloads the program.

8.1.1.2 Functional description

Write 0xCCCC in the key value register (IWDG_KR) to start enabling the independent watchdog. At this point the counter starts counting down from its reset value 0x3FFFF. When the counter reaches the value 0x00000, a reset signal is generated (IWDG_RESET).

Whenever 0xAAAA is written in the key value register (IWDG_KR), the register is automatically reloaded. The value in (IWDG_RLR) is then reloaded into the counter, thus avoiding generating a watchdog reset.

If the main program is abnormal and can not feed the dog correctly, a system reset will result.

8.1.1.3 Register access protection

The IWDG_RLR register is write-protected. To modify the value of this register, you must first write 0x5555 to the IWDG_KR register and then wait one instruction cycle. Writing to this register with a different value will disrupt the order of operations and the register will be re-protected. A reload operation (writing 0xAAAA) will also initiate write protection.

The state register indicates whether the decrement counter is being updated.

8.1.2 IWDG register description

Base address：0x3000_02A0

8.1.2.1 Key value register（IWDG_KR）

Offset address:0x00

Reset value:0x0000_0000

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

Bit	Marking	Functional description
31:16	Reserved	Reserved bit
15:0	key	Key value (write register only, readout value 0x0000) 1. Software must write 0xAAAA at regular intervals, otherwise the watchdog will generate a reset when the counter is 0 2. Wait one instruction cycle after writing 0x5555 and then you can write the IWDG_RLR register 3. Write 0xCCCC to start watchdog working

8.1.2.2 Reload register（IWDG_RLR）

Offset address:0x04

Reset value:0x0003_FFFF

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

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

Bit	Marking	Functional description
31:18	Reserved	Reserved bit
17:0	RL	Watchdog counter reload value 1. Write-protected; 2. Used to define the reload value of the watchdog counter. Whenever the IWDG_KR register is written to the 0xAAAA, the reload value is transferred to the counter, which then counts down from this value; 3. This register can only be modified when the RVU bit in the IWDG_SR register is 0 and the value read out is valid.

8.1.2.3 Status register（IWDG_SR）

Offset address:0x08

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:2	Reserved	Reserved bit

Bit	Marking	Functional description
1	count_ens	IWDG counter operating state bit, 1: IWDG counter is counting
0	RVU	Watchdog counter reload value update 1. This bit is set to 1 by hardware to indicate that an update of the reload value is in progress; 2. This bit is cleared by hardware when the watchdog counter reload value update is complete; 3. The reload value can only be changed after the RVU bit has been cleared.

8.1.3 IWDG register mapping

Register list

Base address：0x3000_02A0

Register	Offset address	Description
IWDG_KR	0x00	IWDG Key value Register
IWDG_RLR	0x04	IWDG Reload Register
IWDG_SR	0x08	IWDG Status Register

9 Advanced-control TIMER1&TIMER2

9.1 Introduction

CSM32RV20 has two advanced timers, TIMER1 and TIMER2. They have the same function and are fully synchronized, which can be synchronized operation.

The Advanced Control Timer (TIMERx) 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, 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 external clock controller prescalers.

9.2 Main characteristics

TIMERx timer features include:

- 16-bit up, down, up/down auto-reload counter
- 16-bit programmable prescaler allowing dividing 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 modes)
  - One-pulse mode output
  - Complementary outputs with programmable dead-time
- 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
- 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
  - Break input

9.3 Block diagram

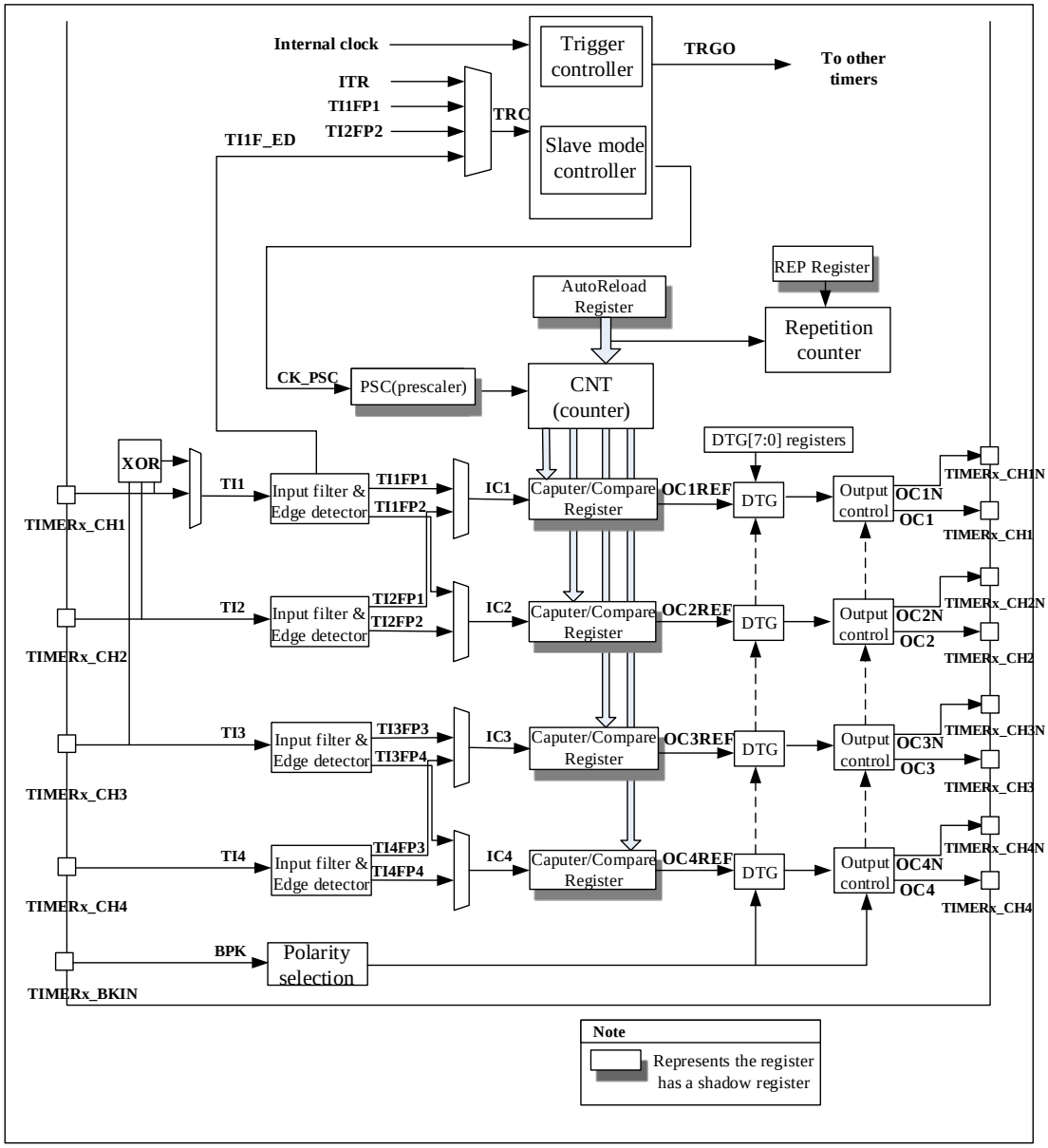


Figure9-1 Advance-control timer block diagram

9.4 Functional description

9.4.1 Time-base unit

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

register and the prescaler register can be written or read by software. This is true even when the counter is running.

The time-base unit includes:

- Counter register (TIMERx_CNT)
- Prescaler register (TIMERx_PSC)
- Auto-reload register (TIMERx_ARR)
- Repetition counter register (TIMERx_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 TIMERx_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 TIMERx_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 TIMERx_CR1 register is set (refer also to the slave mode controller description to get more details on counter enabling).

Note: The counter starts counting 1 clock cycle after setting the CEN bit in the TIMERx_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 TIMERx_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.

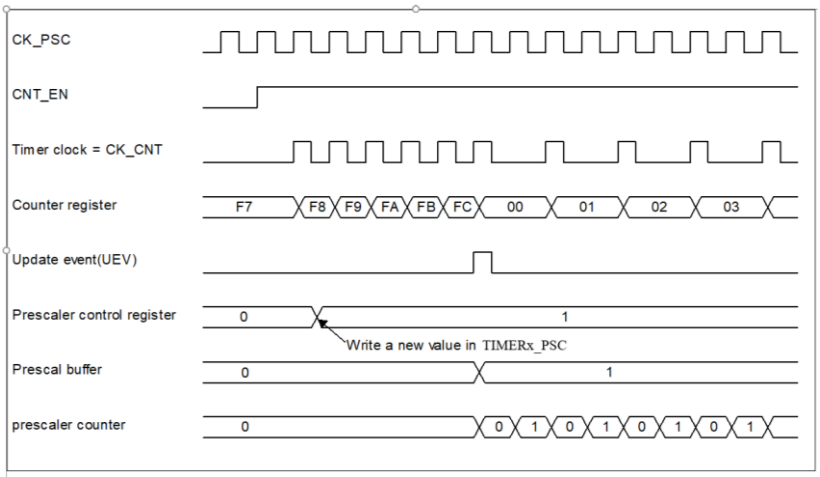


Figure9-2 Counter timing diagram with prescaler division change from 1 to 2

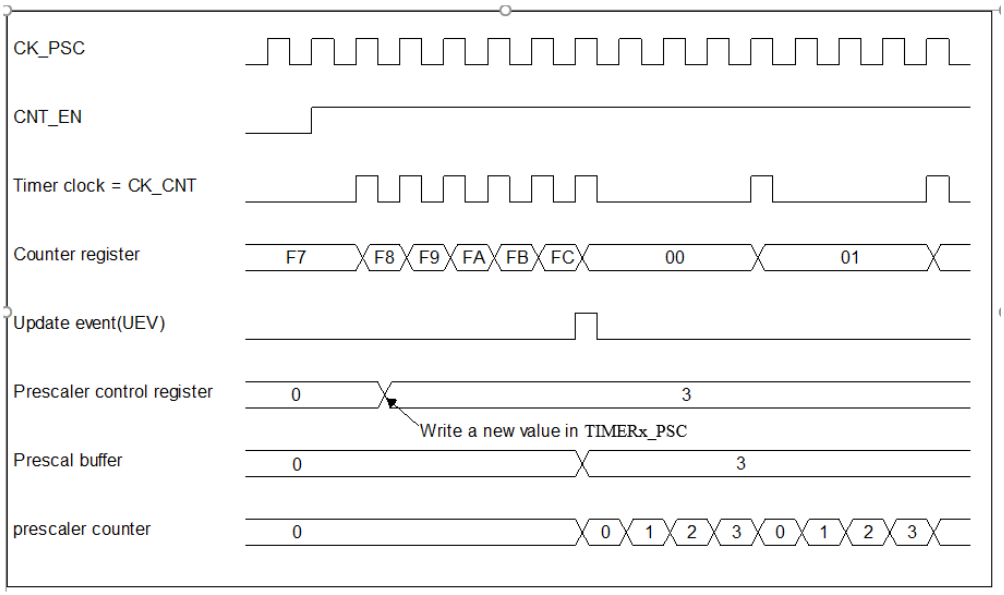


Figure9-3 Counter timing diagram with prescaler division change from 1 to 4

9.4.2 Counter modes

Upcounting mode

In upcounting mode, the counter counts from 0 to the auto-reload value (content of the TIMERx_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 (TIMERx_RCR). Else the update event is generated at each counter overflow.

Setting the UG bit in the TIMERx_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 TIMERx_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 TIMERx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (no interrupt). 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 TIMERx_SR register) is set (depending on the URS bit):

- The repetition counter is reloaded with the content of TIMERx_RCR register
- The auto-reload shadow register is updated with the preload value (TIMERx_ARR)
- The buffer of the prescaler is reloaded with the preload value (content of the TIMERx_PSC register)

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

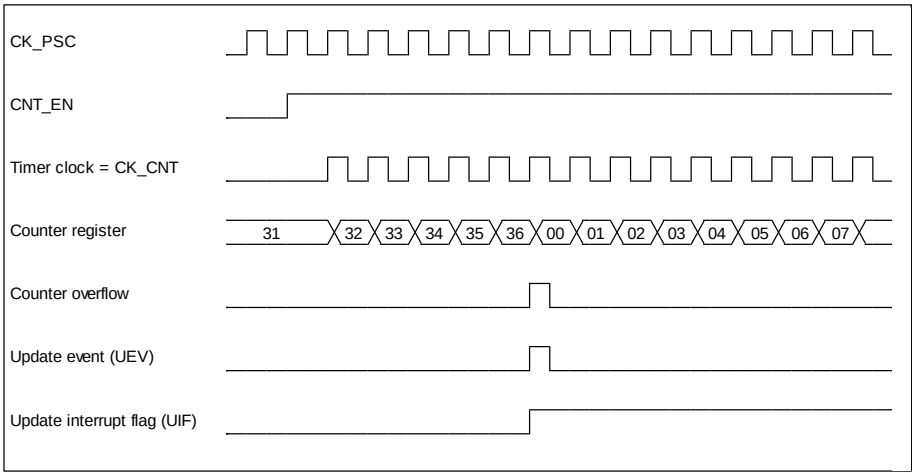


Figure9-4 Counter timing diagram, internal clock divided by 1

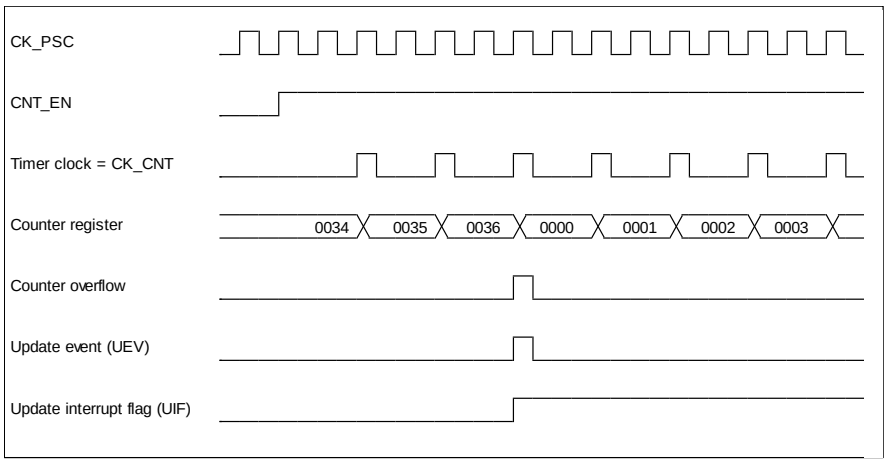


Figure9-5 Counter timing diagram, internal clock divided by 2

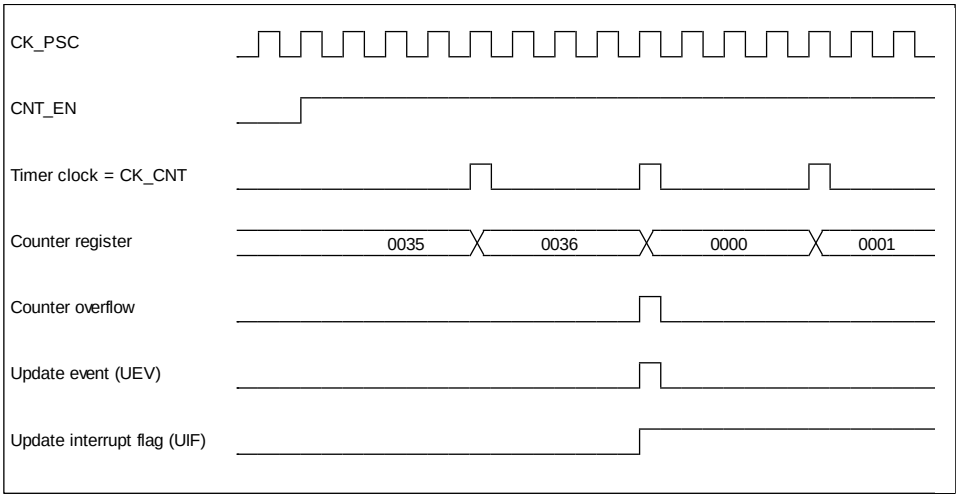


Figure9-6 Counter timing diagram, internal clock divided by 4

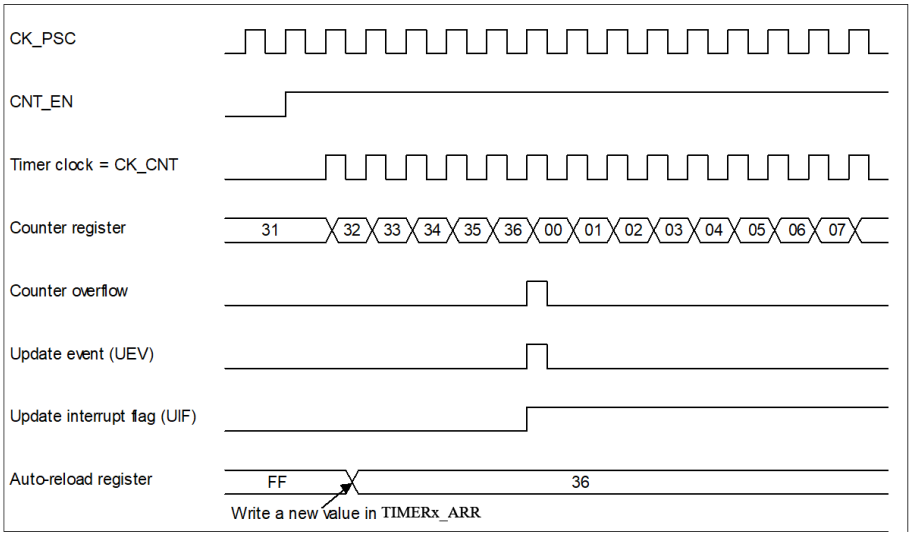


Figure9-7 Counter timing diagram, update event when ARPE=0 (TIMERx_ARR not preloaded)

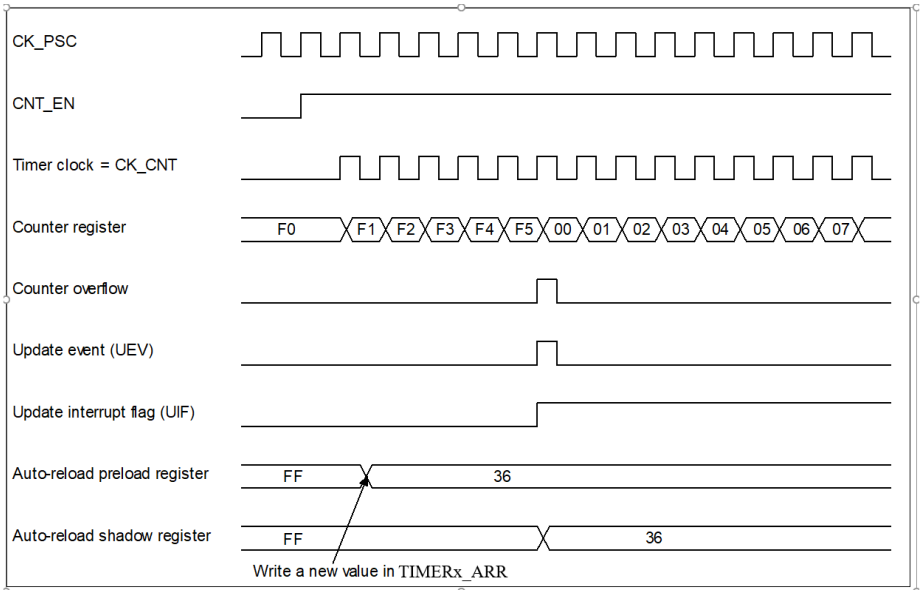


Figure9-8 Counter timing diagram,update event when ARPE=1 (TIMERx_ARR preloaded)

Downcounting mode

In downcounting mode, the counter counts from the auto-reload value (content of the TIMERx_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 (TIMERx_RCR). Else the update event is generated at each counter underflow. Setting the UG bit in the TIMERx_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 TIMERx_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 TIMERx_CR1 register is set, setting the UG bit generates an update event UEV but without setting the UIF flag (thus no interrupt). 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 TIMERx_SR register) is set (depending on the URS bit):

- The repetition counter is reloaded with the content of TIMERx_RCR register
- The auto-reload active register is updated with the preload value (content of the TIMERx_ARR register)

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 TIMERx_ARR=0x36.

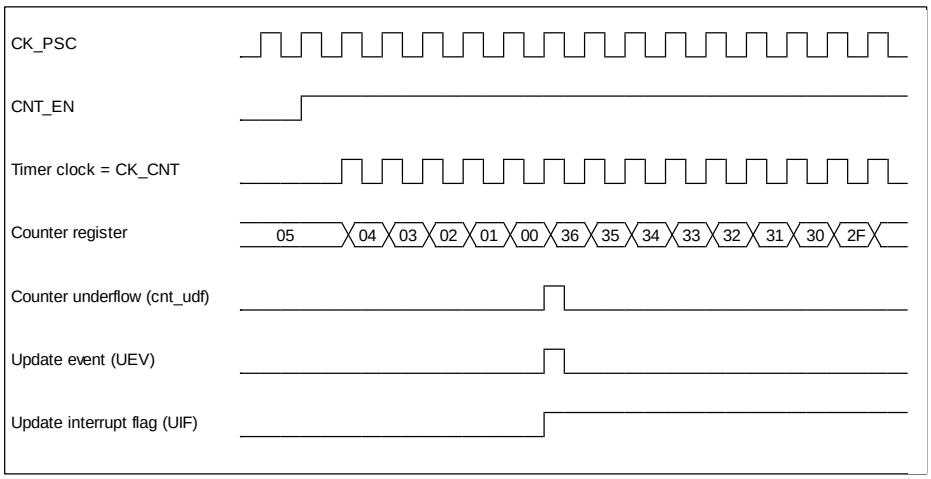


Figure9-9 Counter timing diagram, internal clock divided by 1

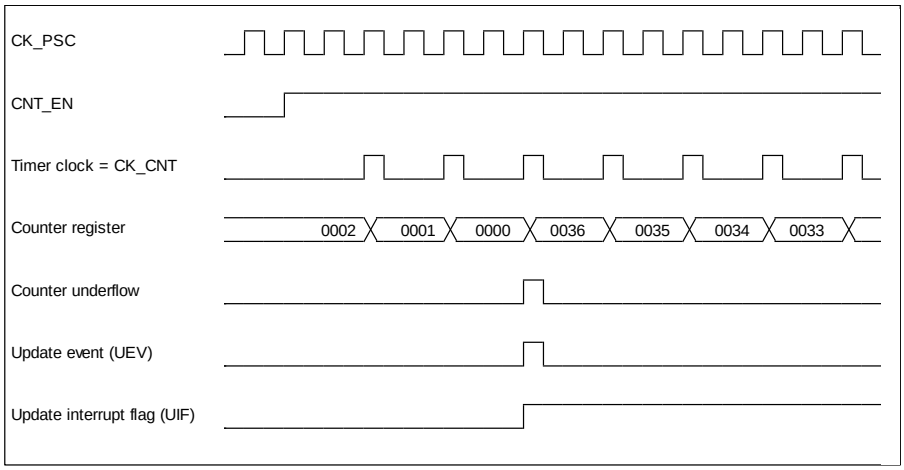


Figure9-10 Counter timing diagram, internal clock divided by 2

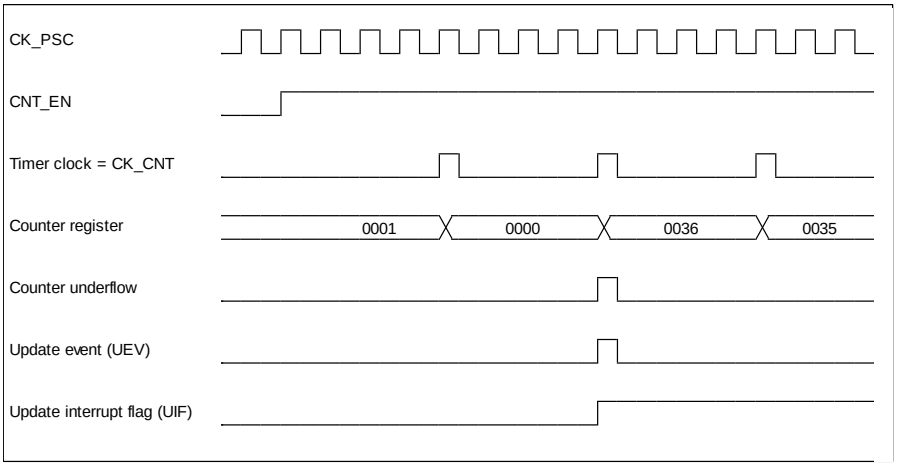


Figure9-11 Counter timing diagram, internal clock divided by 4

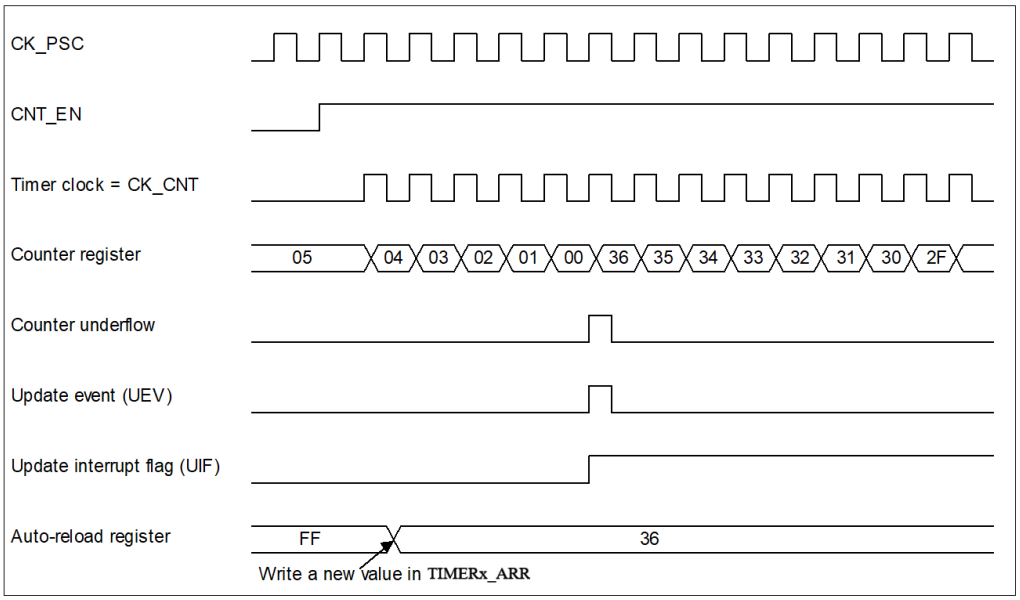


Figure9-12 Counter timing diagram, update event when ARPE=0 (TIMERx_ARR not preloaded)

Center-aligned mode (up/down counting)

In center-aligned mode, the counter counts from 0 to the auto-reload value (content of the TIMERx_ARR register) – 1, generates a counter overflow event, then counts

from the autoreload 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 TIMERx_CR1 register cannot be written.

The update event can be generated at each counter overflow and at each counter underflow or by setting the UG bit in the TIMERx_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 TIMERx_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 TIMERx_CR1 register is set, setting the UG bit generates an UEV update event but without setting the UIF flag (thus no interrupt). 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 TIMERx_SR register) is set (depending on the URS bit):

- The repetition counter is reloaded with the content of TIMERx_RCR register
- The auto-reload active register is updated with the preload value (content of the TIMERx_ARR register)

Note: If the update source is a counter overflow, the autoreload 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.

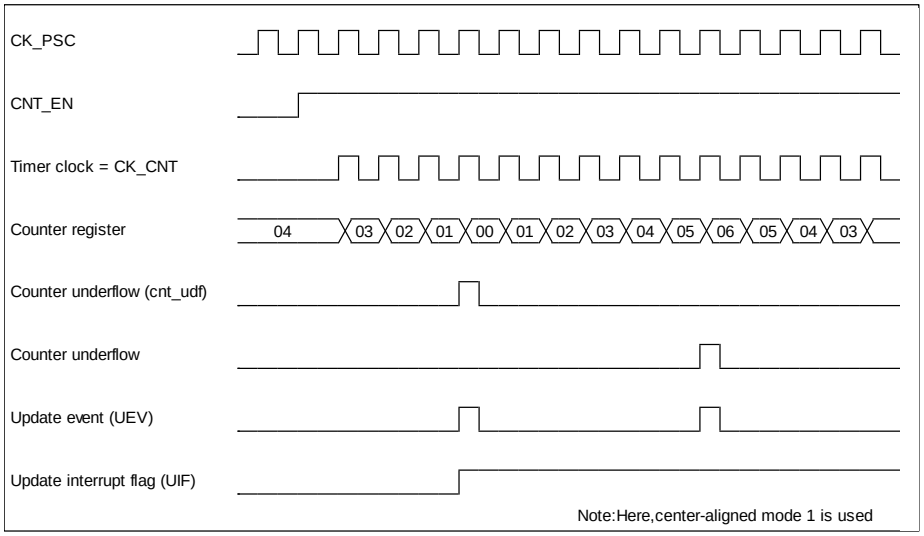


Figure9-13 Counter timing diagram, internal clock divided by 1, TIMERx_ARR = 0x6

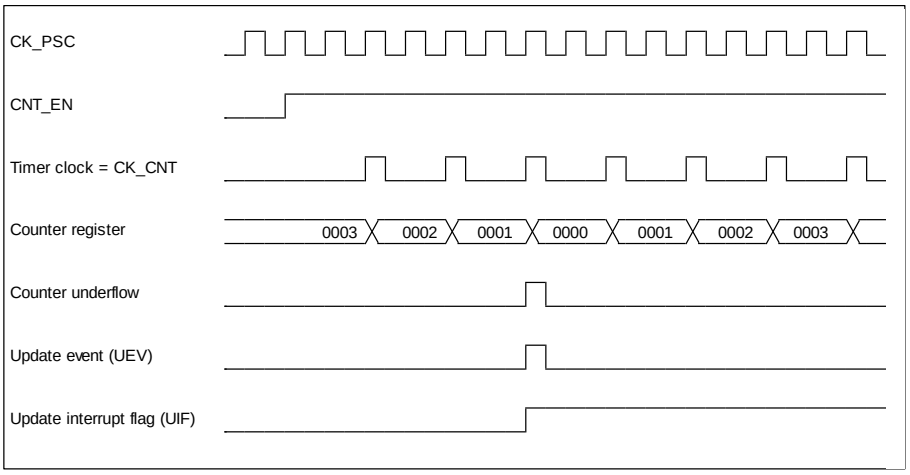


Figure9-14 Counter timing diagram, internal clock divided by 2, TIMERx_ARR = 0x6

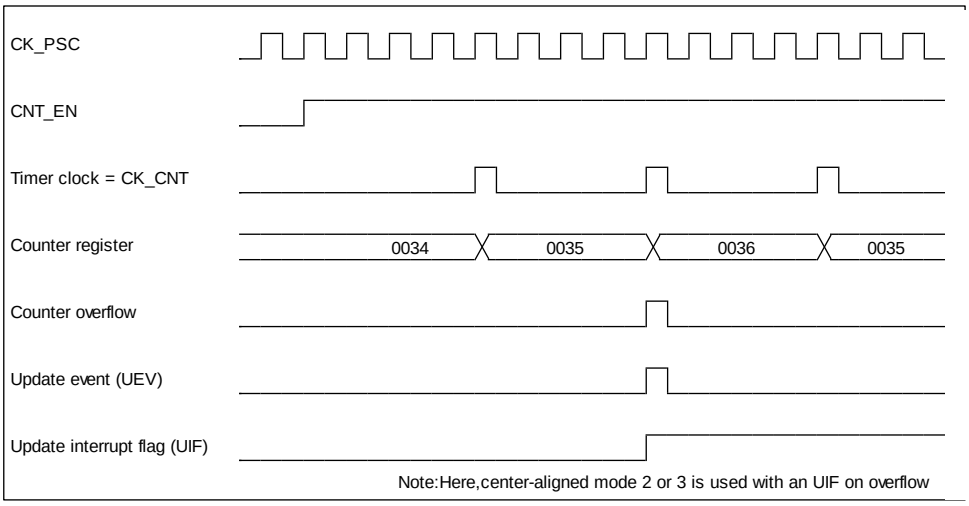


Figure9-15 Counter timing diagram, internal clock divided by 4, TIMERx_ARR = 0x6

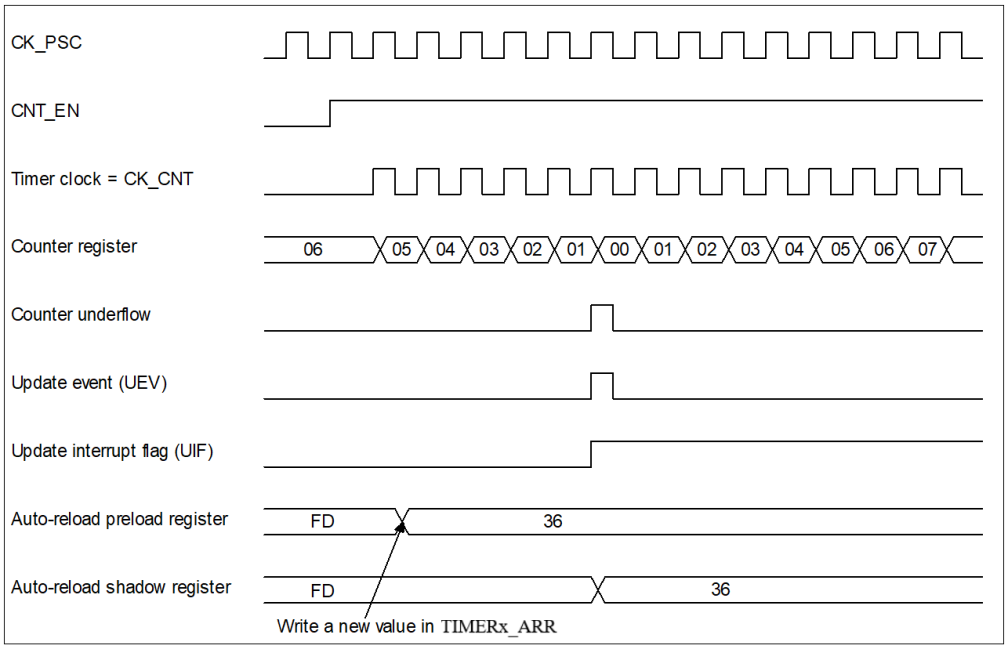


Figure9-16 Counter timing diagram, Update event with ARPE=1 (counter overflow)

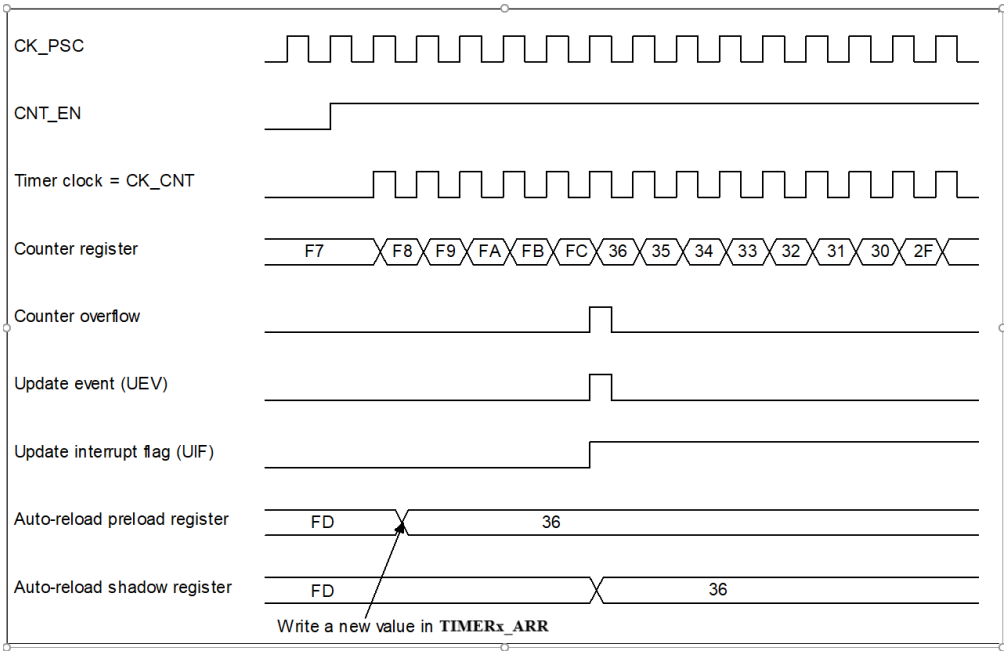


Figure9-17 Counter timing diagram, Update event with ARPE=1 (counter overflow)

9.4.3 Repetition counter

9.4.1Time-base unit describes how the update event (UEV) is generated with respect to the counter overflows/underflows. It is actually generated only when the repetition counter has reached zero. This can be useful when generating PWM signals

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

The repetition counter is decremented:

- At each counter overflow in upcounting mode
- At each counter underflow in downcounting mode
- At each counter overflow and at each counter underflow in center-aligned mode

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

The repetition counter is an auto-reload type,the repetition rate is maintained as defined by the TIMERx_RCR register value (refer to Figure 72). When the update event is generated by software (by setting the UG bit in TIMERx_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 TIMERx_RCR register

9.4.4 Clock selection

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

- Internal clock (CK_INT)
- External clock mode1: external input pin

Internal clock source (CK_INT)

If the slave mode controller is disabled (SMS=000), then the CEN, DIR (in the

TIMERx_CR1 register) and UG bits (in the TIMERx_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.

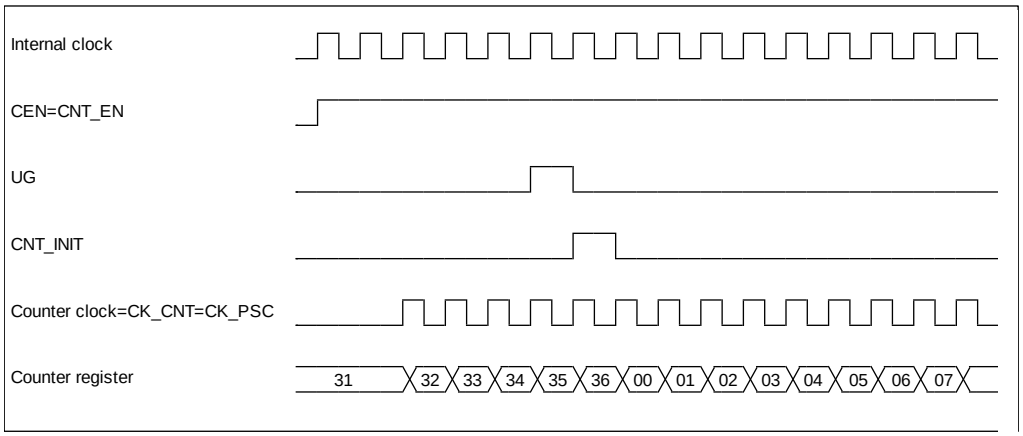


Figure9-18 Control circuit in normal mode, internal clock divided by 1

External clock source mode 1

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

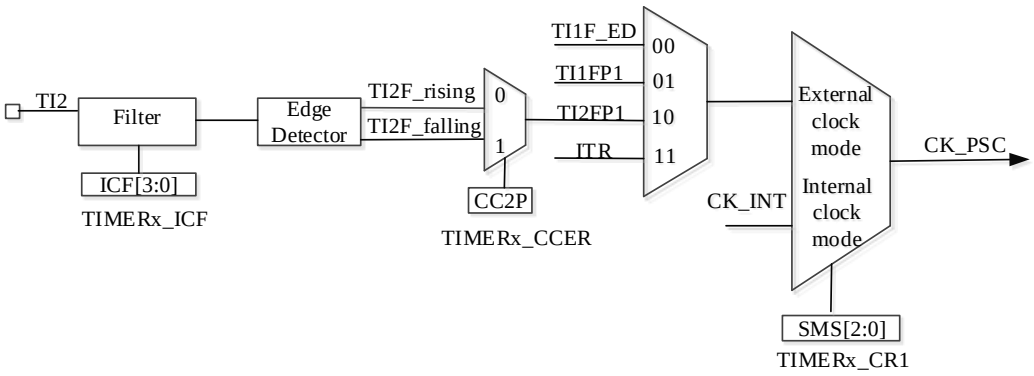


Figure9-19 TI2 external clock connection example

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 TIMERx_CCMR1 register.
2. Configure the input filter duration by writing the IC2F[3:0] bits in the TIMERx_ICF register (if no filter is needed, keep IC2F=0000).

3. Select rising edge polarity by writing CC2P=0 in the TIMERx_CCER register.
4. Configure the timer in external clock mode 1 by writing SMS=111 in the TIMERx_CR1 register.
5. Select TI2 as the trigger input source by writing TS=10 in the TIMERx_CR1 register.
6. Enable the counter by writing CEN=1 in the TIMERx_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.

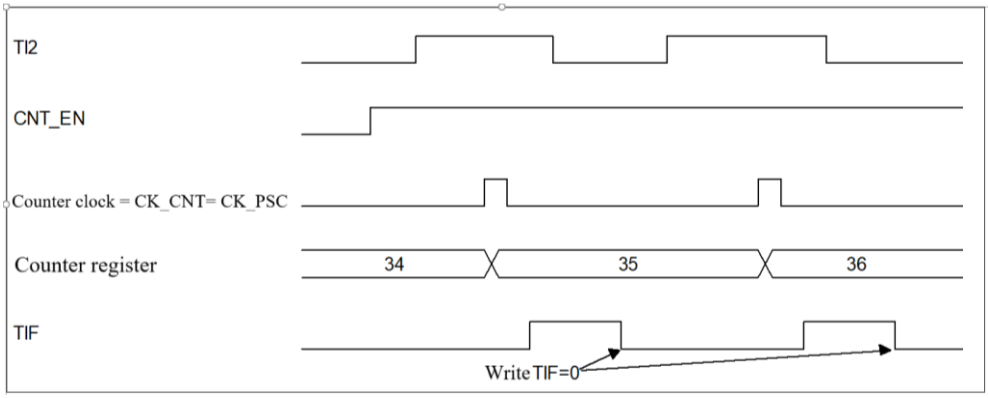


Figure9-20 Control circuit in external clock mode 1

9.4.5 Capture/compare channels

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

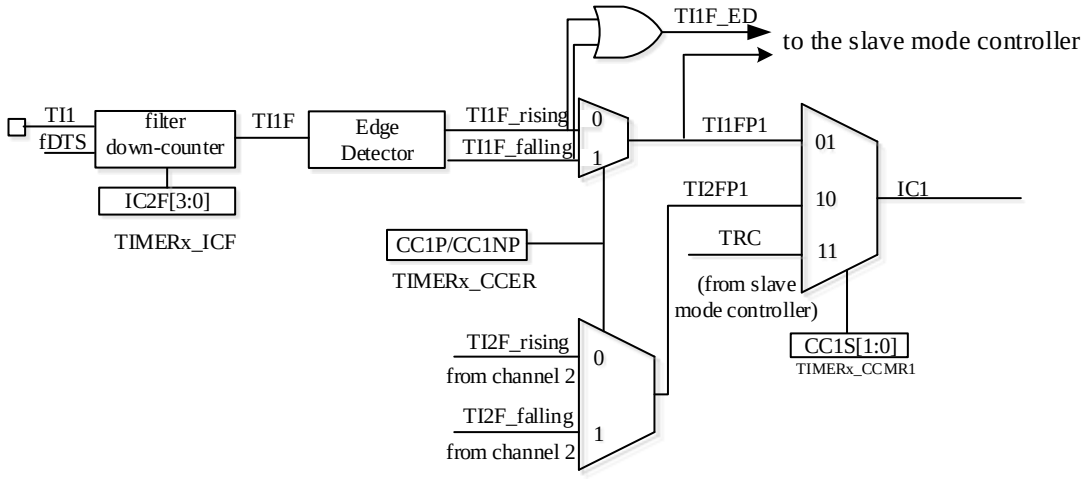


Figure9-21 Capture/compare channel (example: channel 1 input stage)

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.

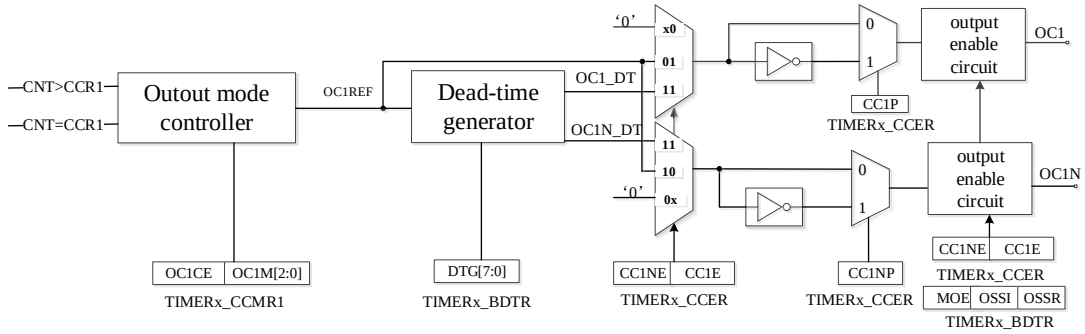


Figure9-22 Output stage of capture/compare channel

9.4.6 Input capture mode

In Input capture mode, the Capture/Compare registers (TIMERx_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 (TIMERx_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 (TIMERx_SR register) is set. CCxIF can be cleared by software by writing it to '0' or by reading the captured data stored in the TIMERx_CCRx register. CCxOF is cleared when written to '0'.

The following example shows how to capture the counter value in TIMERx_CCR1 when TI1 input rises. To do this, use the following procedure:

- Select the active input: TIMERx_CCR1 must be linked to the TI1 input, so write the CC1S bits to 01 in the TIMERx_CCMR1 register. As soon as CC1S becomes different from 00, the channel is configured in input and the TIMERx_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 TIMERx_ICF register if the input is a TIx input). Let's imagine that, when toggling, the input signal is not stable during at must 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 TIMERx_ICF register.
- Select the edge of the active transition on the TI1 channel by writing CC1P bit to 0 in the TIMERx_CCER register (rising edge in this case).
- Enable capture from the counter into the capture register by setting the CC1E bit in the TIMERx_CCER register.
- If needed, enable the related interrupt request by setting the CC1IE bit in the TIMERx_DIER register.

When an input capture occurs:

- The TIMERx_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.

IC interrupt can be generated by software by setting the corresponding CCxG bit in the TIMERx_EGR register.

9.4.7 PWM input mode

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

- Two ICx signals are mapped on the same TIx input.
- These 2 ICx signals are active on edges with opposite polarity.
- One of the two TIxFP signals is selected as trigger input and the slave mode controller is configured in reset mode. For example, user can measure the period (in TIMERx_CCR1 register) and the duty cycle (in TIMERx_CCR2 register) of the PWM applied on TI1 using the following procedure:

- Select the active input for TIMERx_CCR1: write the CC1S bits to 01 in the TIMERx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP1(used both for capture in TIMERx_CCR1 and counter clear): write the CC1P bit to '0' (active on rising edge).
- Select the active input for TIMERx_CCR2: write the CC2S bits to 10 in the TIMERx_CCMR1 register (TI1 selected).
- Select the active polarity for TI1FP2(used for capture in TIMERx_CCR2): write the CC2P bit to '1' (active on falling edge).
- Select the valid trigger input: write the TS bits to 01 in the TIMERx_CR1 register (TI1FP1 selected).
- Configure the slave mode controller in reset mode: write the SMS bits to 100 in the TIMERx_CR1 register.
- Enable the captures: write the CC1E and CC2E bits to '1' in the TIMERx_CCER register.

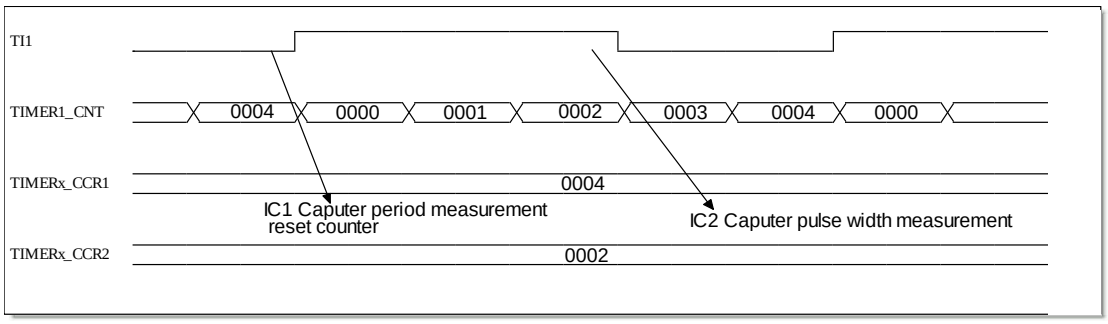


Figure9-23 PWM input mode timing

The PWM input mode can be used only with the TIMERx_CH1/TIMERx_CH2 signals due to the fact that only TI1FP1 and TI2FP2 are connected to the slave mode controller.

9.4.8 Forced output mode

In output mode (CCxS bits = 00 in the TIMERx_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 TIMERx_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 TIMERx_CCMRx register.

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

9.4.9 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 TIMERx_CCMRx register) and the output polarity (CCxP bit in the TIMERx_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 TIMERx_SR register).
- Generates an interrupt if the corresponding interrupt mask is set (CCxIE bit in the TIMERx_DIER register).

The TIMERx_CCRx registers can be programmed with or without preload

registers using the OCxPE bit in the TIMERx_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 TIMERx_ARR and TIMERx_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 TIMERx_CR1 register.

The TIMERx_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 TIMERx_CCRx shadow register is updated only at the next update event UEV). An example is given in Figure 9-24.

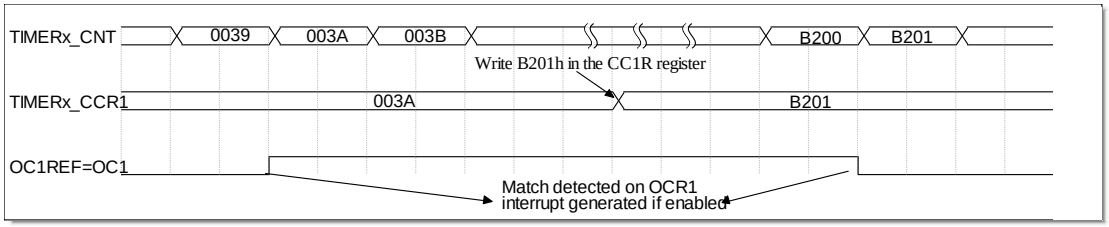


Figure9-24 Output compare mode, toggle on OC1

9.4.10 PWM mode

Pulse Width Modulation mode allows generating a signal with a frequency determined by the value of the TIMERx_ARR register and a duty cycle determined by the value of the TIMERx_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 TIMERx_CCMRx register. The corresponding preload register must be

enabled by setting the OCxPE bit in the TIMERx_CCMRx register, and eventually the auto-reload preload register (in upcounting or center-aligned modes) by setting the ARPE bit in the TIMERx_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 TIMERx_EGR register.

OCx polarity is software programmable using the CCxP bit in the TIMERx_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 (TIMERx_CCER and TIMERx_BDTR registers). Refer to the TIMERx_CCER register description for more details.

In PWM mode (1 or 2), TIMERx_CNT and TIMERx_CCRx are always compared to determine whether TIMERx_CCRx ⩽ TIMERx_CNT or TIMERx_CNT ⩽ TIMERx_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 TIMERx_CR1 register.

PWM edge-aligned mode

- Upcounting configuration

Upcounting is active when the DIR bit in the TIMERx_CR1 register is low. In the following example, we consider PWM mode 1. The reference PWM signal OCxREF is high as long as TIMERx_CNT < TIMERx_CCRx else it becomes low. If the compare value in TIMERx_CCRx is greater than the auto-reload value (in TIMERx_ARR) then OCxREF is held at ‘1’. If the compare value is 0 then OCxREF is held at ‘0’. The following figure shows some edge-aligned PWM waveforms in an example where TIMERx_ARR=8.

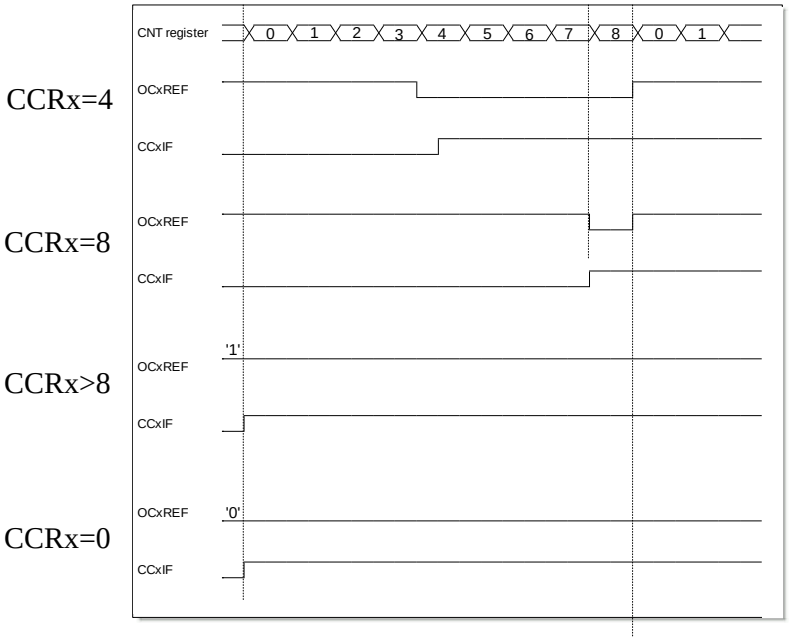


Figure9-25 Edge-aligned PWM waveforms (ARR=8)

- Downcounting configuration

Downcounting is active when DIR bit in TIMERx_CR1 register is high. In PWM mode 1, the reference signal OCxRef is low as long as TIMERx_CNT > TIMERx_CCRx else it becomes high. If the compare value in TIERMx_CCRx is greater than the auto-reload value in TIMERx_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 TIMERx_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 TIMERx_CR1 register is updated by hardware and must not be changed by software.

Figure 9-26 shows some center-aligned PWM waveforms in an example where:

- TIMERx_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 TIMERx_CR1 register

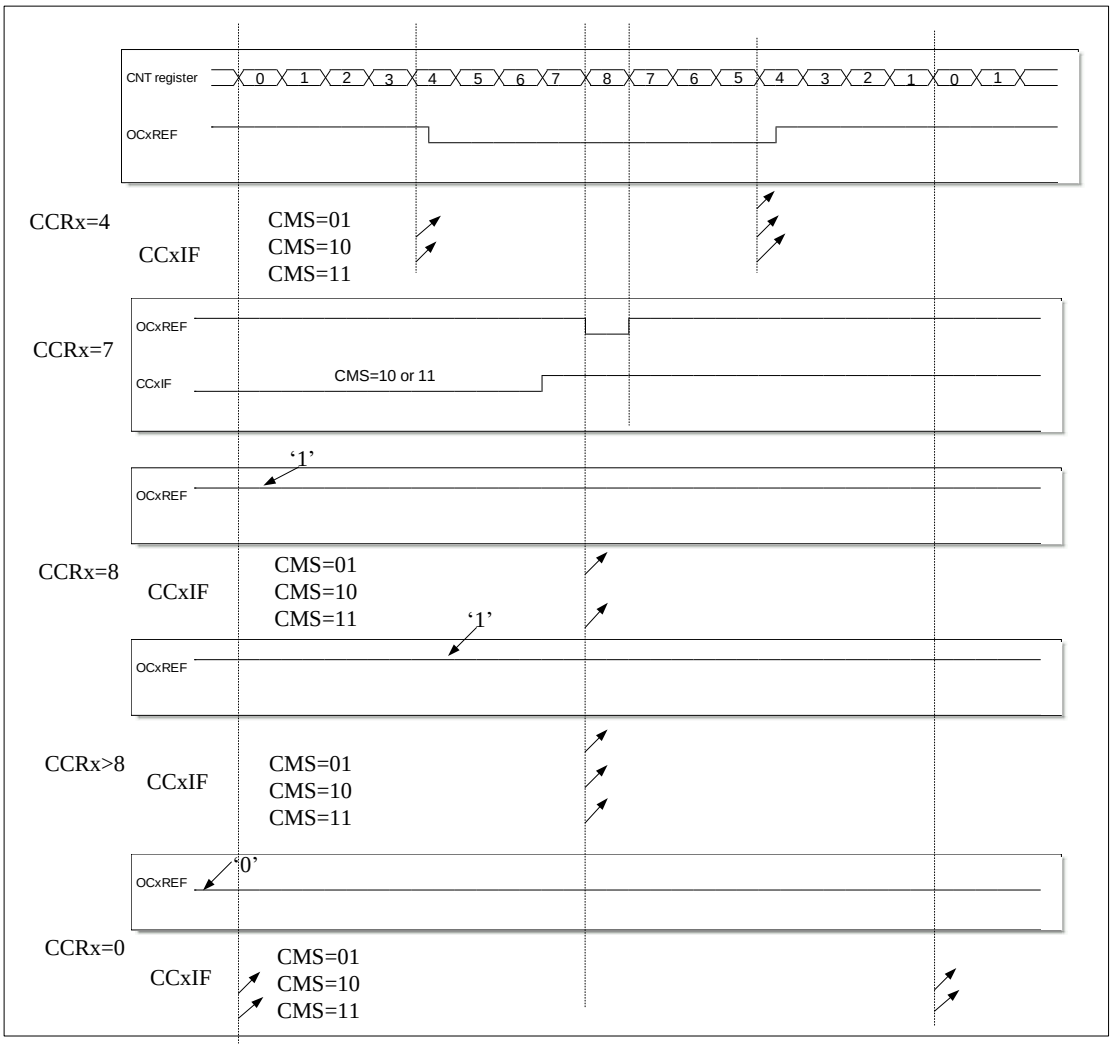


Figure9-26 Center-aligned PWM waveforms (ARR=8)

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 TIMERx_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 (TIMERx_CNT>TIMERx_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 TIMERx_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 TIMERx_EGR register) just before starting the counter and not to write the counter while it is running.

9.4.11 Complementary outputs and dead-time insertion

The advanced-control timers 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 TIMERx_CCER register.

The complementary signals OCx and OCxN are activated by a combination of several control bits: the CCxE and CCxNE bits in the TIMERx_CCER register and the MOE, OISx, OISxN, OSSI and OSSR bits in the TIMERx_BDTR. Refer to Table 9-1 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 TIMERx_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)

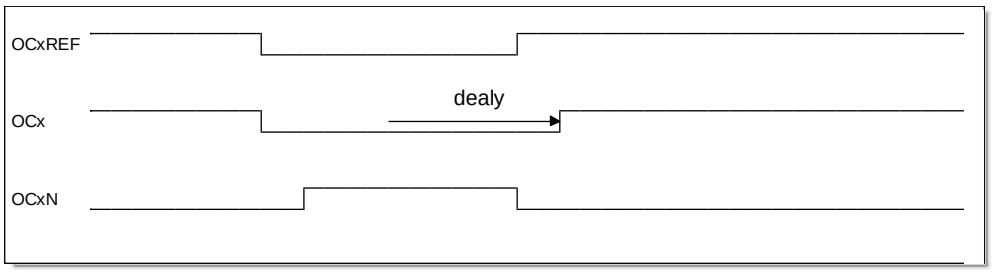


Figure9-27 Complementary output with dead-time insertion

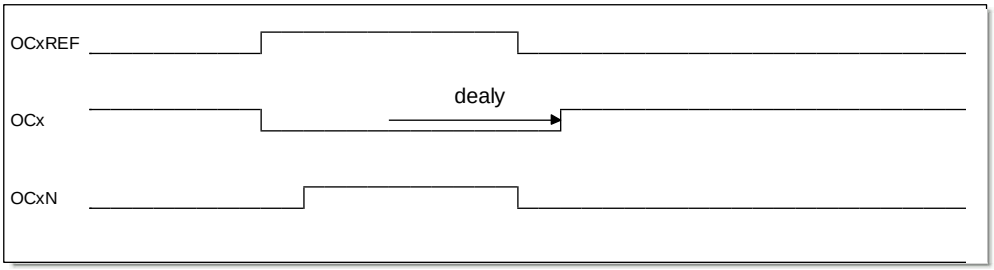


Figure9-28 Dead-time waveforms with delay greater than the negative pulse

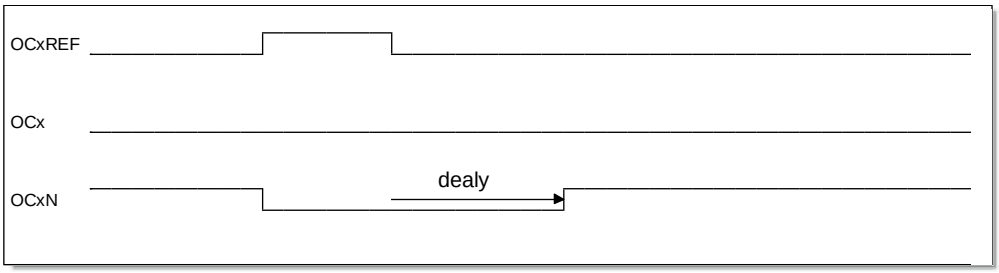


Figure9-29 Dead-time waveforms with delay greater than the positive pulse

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

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

outputs active and complementary with dead-time.

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

9.4.12 Using the break function

When using the break function, the output enable signals and inactive levels are modified according to additional control bits (MOE, OSSI and OSSR bits in the TIMERx_BDTR register, OISx and OISxN bits in the TIMERx_CR1 register). In any case, the OCx and OCxN outputs cannot be set both to active level at a given time. Refer to Table 9-1 for more details.

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

- The MOE bit is cleared asynchronously, putting the outputs in inactive state, idle state or in reset state (selected by the OSSI bit)
- Each output channel is driven with the level programmed in the OISx bit in the TIMERx_CR1 register as soon as MOE=0. If OSSI=0 then the timer releases the enable output else the enable output remains high
- When complementary outputs are used:
  - The outputs are first put in reset state inactive state (depending on the polarity).
  - If the timer clock is still present, then the dead-time generator is reactivated in order to drive the outputs with the level programmed in the OISx and OISxN bits after a dead-time. Even in this case, OCx and OCxN cannot be driven to their active level together. Note that because of the resynchronization on MOE, the dead-time duration is a bit longer than usual (around 2 ck_tim clock cycles).
  - If OSSI=0 then the timer releases the enable outputs else the enable outputs remain or become high as soon as one of the CCxE or CCxNE bits is high.
- The break status flag (BIF bit in the TIMERx_SR register) is set. An interrupt can be generated if the BIE bit in the TIMERx_DIER register is set.
- If the AOE bit in the TIMERx_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.

The break can be generated by the BKI input which has a programmable polarity and an enable bit BKE in the TIMERx_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 to freeze the configuration of several parameters (dead-time duration, OCx/OCxN polarities and state when disabled, OCxM configurations, break enable and polarity). The protection can be selected among 3 levels with the LOCK bits in the TIMERx_BDTR register. The LOCK bits can be written only once after an MCU reset.

The Figure below shows an example of behavior of the outputs in response to a break.

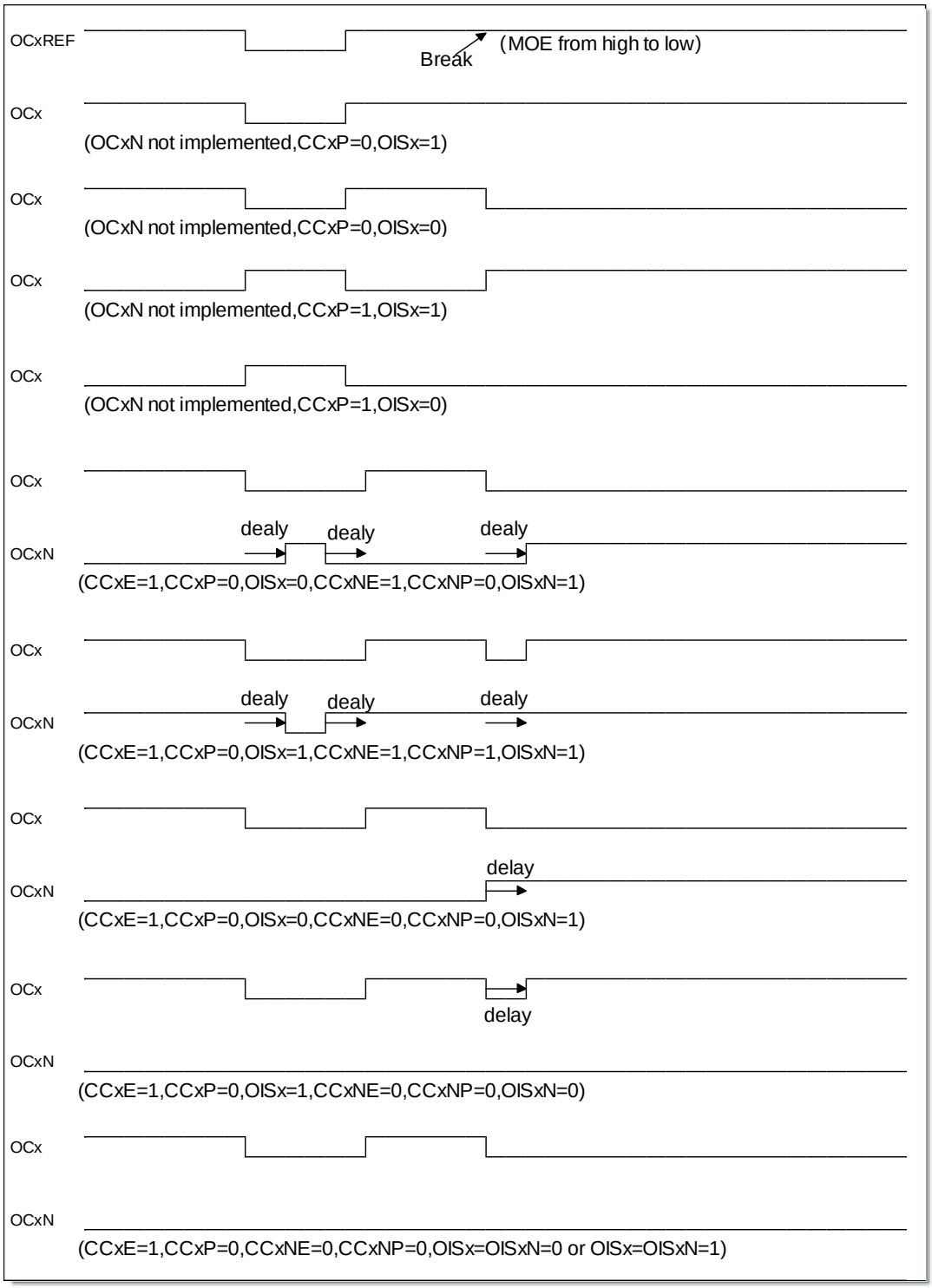


Figure9-30 Output behavior in response to a break

9.4.13 6-step PWM generation

When complementary outputs are used on a channel, preload bits are available on the OCxM, CCxE and CCxNE bits. The preload bits are transferred to the shadow bits at the COM commutation event. The user can thus program in advance the configuration for the next step and change the configuration of all the channels at the same time. COM can be generated by software by setting the COM bit in the TIMERx_EGR register or by hardware (on TRGI rising edge).A flag is set when the COM event occurs (COMIF bit in the TIMERx_SR register), which can generate an interrupt (if the COMIE bit is set in the TIMERx_DIER register).

The Figure 9-31 describes the behavior of the OCx and OCxN outputs when a COM event occurs, in 3 different examples of programmed configurations.

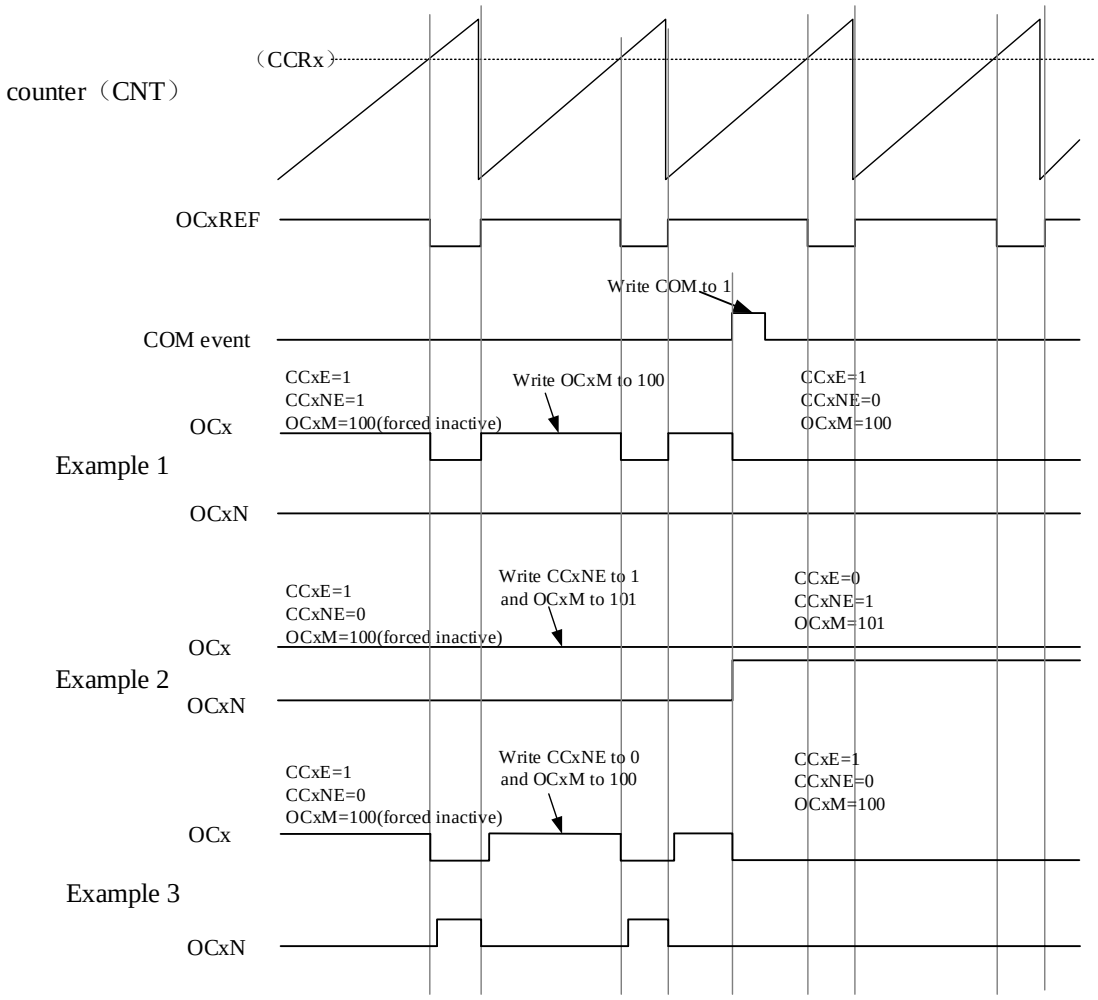


Figure9-31 6-step generation, COM example (OSSR=1)

9.4.14 One-pluse 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 TIMERx_CR1 register. This makes the counter stop automatically at the next update event UEV.A pulse can be correctly generated only if the compare value is different from the counter initial value. Before starting (when the timer is waiting for the trigger), the configuration must be:

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

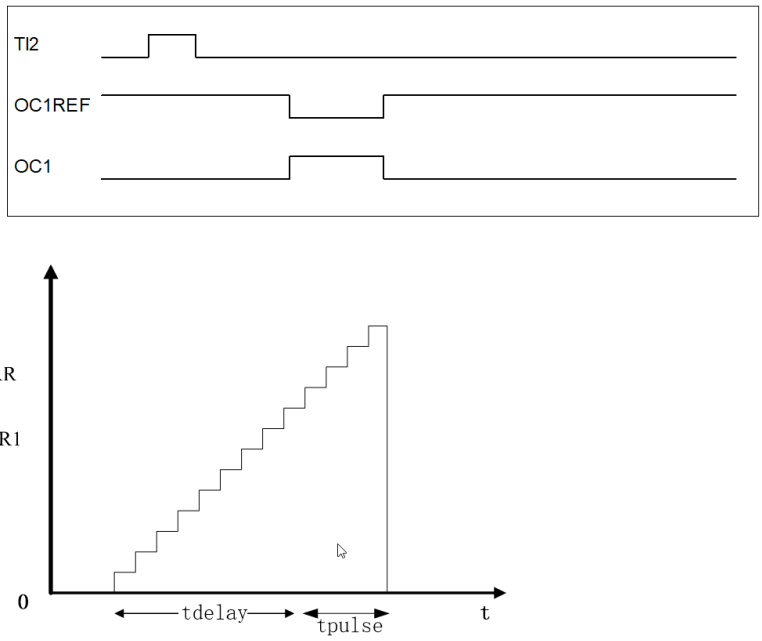


Figure9-32 Example of one pulse mode

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_{pulse} 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 TIMERx_CCMR1 register.
- TI2FP2 must detect a rising edge, write CC2P='0' in the TIMERx_CCER register.

- Configure TI2FP2 as trigger for the slave mode controller (TRGI) by writing TS=2'b10 in the TIMERx_CR1 register.
- TI2FP2 is used to start the counter by writing SMS to '110' in the TIMERx_CR1 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 TIMERx_CCR1 register.
- The t_{pulse} is defined by the difference between the auto-reload value and the compare value (TIMERx_ARR - TIMERx_CCR1).
- Let's 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 TIMERx_CCMR1 register. The user can optionally enable the preload registers by writing OC1PE='1' in the TIMERx_CCMR1 register and ARPE in the TIMERx_CR1 register. In this case the compare value must be written in the TIMERx_CCR1 register, the auto-reload value in the TIMERx_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 TIMERx_CR1 register should be low. The user only wants one pulse (Single mode), so '1' must be written in the OPM bit in the TIMERx_CR1 register to stop the counter at the next update event.

9.4.15 Timer input XOR function

The TI1S bit in the TIMERx_CR1 register, allows the input filter of channel 1 to be connected to the output of a XOR gate, combining the three input pins TIMERx_CH1, TIMERx_CH2 and TIMERx_CH3.The XOR output can be used with all the timer input functions such as trigger or input capture.

9.4.16 Interfacing with Hall sensors

The TIMER1 is used to generate PWM signals to drive the motor, and another timer is connected to the Hall sensor as an "interface timer". The "interfacing timer" captures the 3 timer input pins (TIMERx_CH1, TIMERx_CH2, and TIMERx_CH3) connected through a XOR to the TI1 input channel(selected by setting the TI1S bit in

the TIMERx_CR1 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 interface timer, capture/compare channel 1 is configured in capture mode and captures the signal TRC. The capture 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 by triggering a COM event. The TIMER1 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 another timer through the TRGO output.

Example: the user wants to change the PWM configuration of the advanced-control timer TIMER1 after a programmed delay each time a change occurs on the Hall inputs connected to one of the TIMER2.

- Configure 3 timer inputs XORed to the TI1 input channel by writing the TI1S bit in the TIMER2_CR1 register to ‘1’
- Program the time base: write the TIMER2_ARR to the max value 16’hFFFF (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 TIMER2_CCMR1 register to ‘11’. The digital filter can also be programmed 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 TIMER2_CCMR1 register
- Select OC2REF as trigger output on TRGO: write the MMS bits in the TIMER2_CR1 register to ‘101’， TS = 2’b11

In the advanced-control timer TIMER1, 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 TIMER2_CR1 register) and the COM event is controlled by the trigger input (CCUS=1 in the TIMER2_CR1 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.

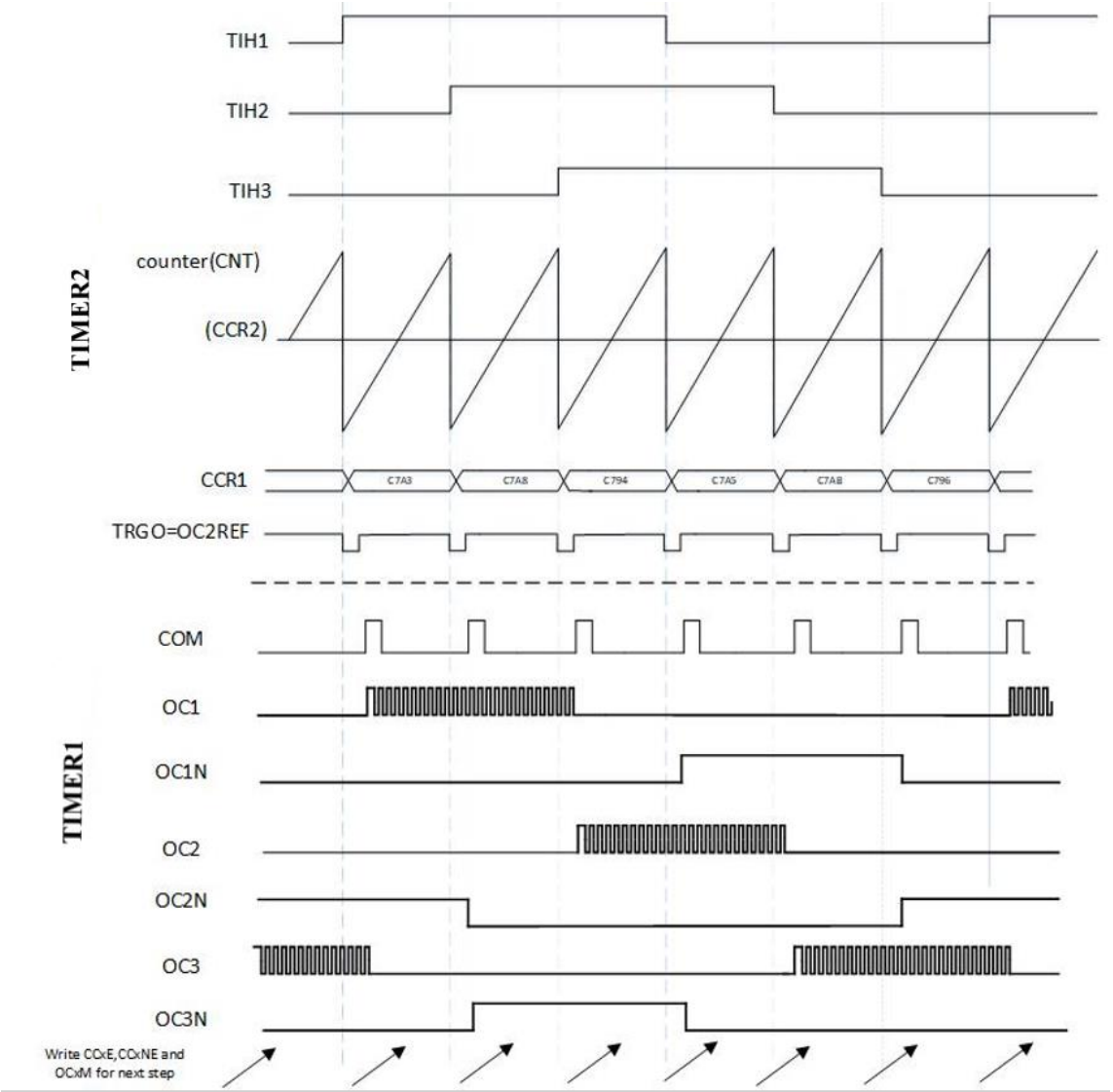


Figure9-33 Example of Hall sensor interface

9.4.17 Synchronization of timer and external trigger

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

Slave mode：reset mode

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

trigger input. Moreover, if the URS bit from the TIMERx_CR1 register is low, an update event UEV is generated. Then all the preloaded registers (TIMERx_ARR, TIMERx_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 TIMERx_CCMR1 register. Write CC1P=0 in TIMERx_CCER register to validate the polarity (and detect rising edges only).
- Configure the timer in reset mode by writing SMS=100 in TIMERx_CR1 register. Select TI1 as the input source by writing TS=2'b01 in TIMERx_CR1 register
- Start the counter by writing CEN=1 in the TIMERx_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 TIMERx_SR register).

The following figure shows this behavior when the auto-reload register TIMERx_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.

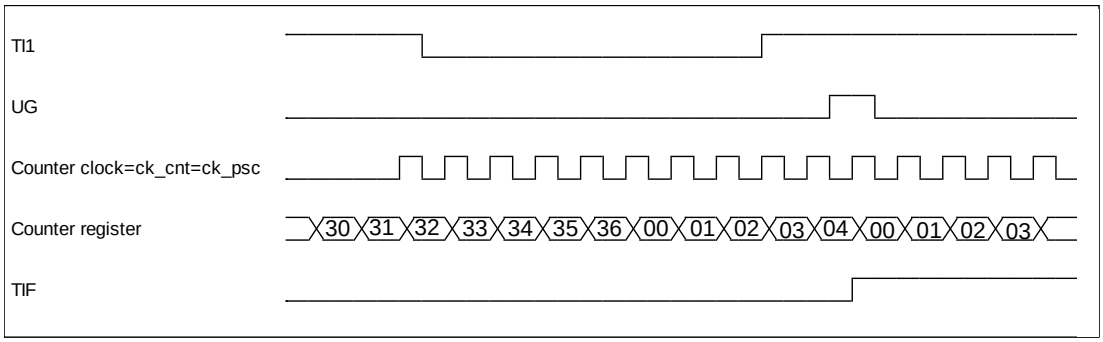


Figure9-34 Control circuit in reset mode

Slave mode: Gated mode

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

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

- Configure the channel 1 to detect low levels on TI1. Configure the input filter duration (in this example, we don't need any filter, so we keep IC1F=0000). The capture prescaler is not used for triggering, so the user does not need to configure it. The CC1S bits select the input capture source only, CC1S=01 in TIMERx_CCMR1 register. Write CC1P=1 in TIMERx_CCER register to validate the polarity (and detect low level only)
- Configure the timer in gated mode by writing SMS=101 in TIMERx_CR1 register. Select TI1 as the input source by writing TS=2'b01 in TIMERx_CR1 register.
- Enable the counter by writing CEN=1 in the TIMERx_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 TIMERx_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.

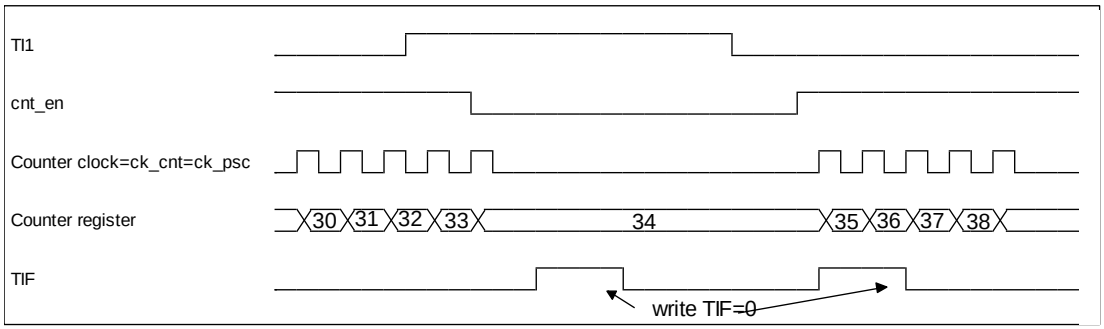


Figure9-35 Control circuit in gated mode

Slave mode: Trigger mode

The counter can start in response to an event on a selected input. In the following example, the upcounter starts in response to a rising edge on TI2 input:

- Configure the channel 2 to detect rising edges on TI2. Configure the input

filter duration (in this example, we don’t need any filter, so we keep IC2F=0000). The capture prescaler is not used for triggering, so there’s no need to configure it. The CC2S bits are configured to select the input capture source only, CC2S=01 in TIMERx_CCMR1 register. Write CC2P=1 in TIMERx_CCER register to validate the polarity (and detect low level only).

- Configure the timer in trigger mode by writing SMS=110 in TIMERx_CR1 register. Select TI2 as the input source by writing TS=2’b10 in TIMERx_CR1 register.

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

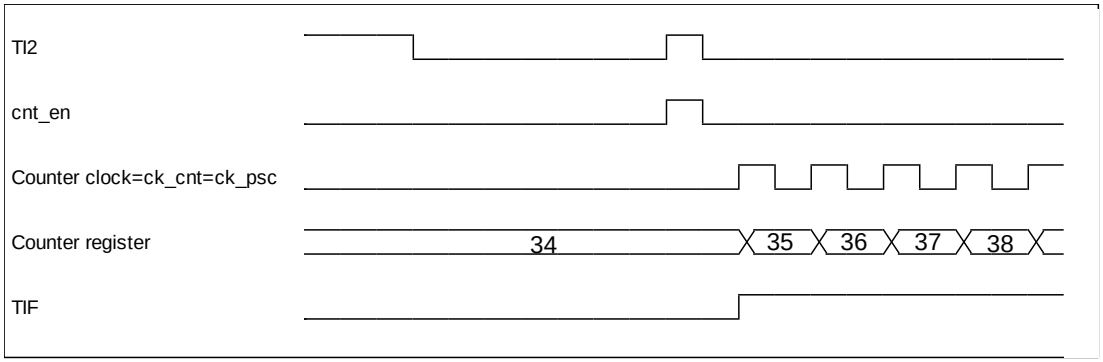


Figure9-36 Control circuit in trigger mode

9.5 TIMERx registers description

TIMER1 Base address: 0x3000_0018

TIMER2 Base address: 0x3000_0098

9.5.1 Control Register（TIMERx_CR1）

Offset address:0x00

Reset value:0x0000_0000

31	30	29	28	27	26	25	24
Reserved	MMS			Reserved		TS	
	RW					RW	

23	22	21	20	19	18	17	16
TI1S	SMS			CCUS	CCPC	OIS4N	OIS4
RW	RW			RW	RW	RW	RW

15	14	13	12	11	10	9	8
OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	CKD	
RW	RW	RW	RW	RW	RW	RW	

7	6	5	4	3	2	1	0
ARPE	CMS		DIR	OPM	URS	UDIS	CEN
RW	RW		RW	RW	RW	RW	RW

Bit	Marking	Functional description
31	Reserved	Reserved bit
30:28	MMS	100: Comparison - The OC1REF signal is used as a trigger output (TRGO); 101: Compare - The OC2REF signal is used as a trigger output (TRGO); 110: Compare - The OC3REF signal is used as a trigger output (TRGO); 111: Comparison - OC4REF signal is used as trigger output (TRGO)
27:26	Reserved	Reserved bit
25:24	TS	Trigger selection This bit-field selects the trigger input to be used to synchronize the counter 00: Edge detector for TI1, both up/down edges of TI1 are valid (TI1F_ED); 01: Filtered timer input 1 (TI1FP1); 10: Filtered Timer Input 2 (TI2FP2); 11: ITR (the selection in timer1 is the TRGO of timer2, the selection in timer2 is the TRGO of timer1)
23	TI1S	TI1 selection 0: The TIMERx_CH1 pin is connected to TI1 input 1: The TIMERx_CH1, CH2 and CH3 pins are connected to the TI1 input (XOR combination)
22:20	SMS	Slave mode selection When external signals are selected the active edge of the trigger signal (TRGI) is linked to the polarity selected on the external input. 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 edge of the selected trigger (TRGI) clock the counter *Note: The gated mode must not be used if TI1F_ED is selected as the trigger input (TS=00). Indeed, TI1F_ED outputs 1 pulse for each transition on TI1F, whereas the gated mode checks the level of the trigger signal*
19	CCUS	Capture/compare control update selection 0: When capture/compare control bits are preloaded (CCPC=1), they are updated by setting the COMG bit only 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 *Note: This bit acts only on channels that have a complementary output.*
18	CCPC	Capture/compare preload control 0: The CCxE, CCxNE and OCxM bits are not preloaded; 1: The CCxE, CCxNE, and OCxM bits are preloaded, after having been written, they are updated only when a commutation event (COM) occurs *Note: This bit acts only on channels that have a complementary output.*
17	OIS4N	Output idle state 4 (OC4N output). 0: When MOE = 0, OC4N = 0 after a dead-time; 1: When MOE = 0, OC4N = 1 after a dead-time. *Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMERx_BDTR register)*
16	OIS4	Output idle state 4 (OC4 output). 0: When MOE = 0, OC4 = 0 after a dead-time if OC4N is implemented; 1: When MOE = 0, OC4 = 1 after a dead-time if OC4N is implemented. *Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMERx_BDTR register)*
15	OIS3N	Output idle state 3 (OC3N output). 0: When MOE = 0, OC3N = 0 after a dead-time; 1: When MOE = 0, OC3N = 1 after a dead-time. *Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMERx_BDTR register).*
14	OIS3	Output idle state 3 (OC3 output).

		0: When MOE = 0, OC3 = 0 after a dead-time if OC3N is implemented; 1: When MOE = 0, OC3 = 1 after a dead-time if OC3N is implemented. *Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMERx_BDTR register).*
13	OIS2N	Output idle state 2 (OC2N output). 0: When MOE = 0, OC2N = 0 after a dead-time; 1: When MOE = 0, OC2N = 1 after a dead-time. *Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMERx_BDTR register).*
12	OIS2	Output idle state 2 (OC2 output). 0: When MOE = 0, OC2 = 0 after a dead-time if OC2N is implemented; 1: When MOE = 0, OC2 = 1 after a dead-time if OC2N is implemented. *Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMERx_BDTR register).*
11	OIS1N	Output idle state 1 (OC1N output). 0: When MOE = 0, OC1N = 0 after a dead-time; 1: When MOE = 0, OC1N = 1 after a adead-time. *Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMERx_BDTR register).*
10	OIS1	Output idle state 1 (OC1 output). 0: When MOE = 0, OC1 = 0 after a dead-time if OC1N is realized; 1: When MOE = 0, OC1 = 1 after a dead-time if OC1N is realized. *Note: This bit can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMERx_BDTR register).*
9:8	CKD	Clock division This bit-field indicates the division ratio between the timer clock (CK_INT) frequency and the digital filters (Tix). 00：$t_{DTS} = t_{CK\_INT}$； 01：$t_{DTS} = 2 \times t_{CK\_INT}$； 10：$t_{DTS} = 4 \times t_{CK\_INT}$； 11：Reserved
7	ARPE	Auto-reload preload enable 0: TIMERx_ARR register is not buffered; 1: TIMERx_ARR register is buffered

6:5	CMS	Center-aligned mode selection 00: edge-aligned mode. The counter counts up or down depending on the direction bit (DIR); 01: center-aligned mode 1. counters count up and down alternately.Output compare interrupt flags of channels configured in output (CCxS=00 in TIMERx_CCMRx register) are set only when the counter is counting down. 10: center-aligned mode 2. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMERx_CCMRx register) are set only when the counter is counting up. 11: center-aligned mode 3. The counter counts up and down alternatively. Output compare interrupt flags of channels configured in output (CCxS=00 in TIMERx_CCMRx register) are set both when the counter is counting up or down. *Note: It is not allowed to switch from edge-aligned mode to center-aligned mode as long as the counter is enabled (CEN=1)*
4	DIR	Direction 0: Counter used as upcounter 1: Counter used as downcounter *Note: This bit is read-only when the counter is configured for center-aligned mode or Encoder Mode.*
3	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)
2	URS	This bit is set and cleared by software to select the UEV event sources. 0: Any of the following events generate an update interrupt request if enabled. These events can be: – Counter overflow/underflow – Setting the UG bit – Update generation through the slave mode controller 1: Only counter overflow/underflow generates an update interrupt if enabled.
1	UDIS	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: – Counter overflow/underflow

		– Setting the UG bit – Update generation through the slave mode controller 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.
0	CEN	Counter enable 0: Counter disabled 1: Counter enabled *Note: External clock, gated mode and encoder mode can work only if the CEN bit has been previously set by software. However trigger mode can set the CEN bit automatically by hardware.*

9.5.2 Filter register（TIMERx_ICF）

Offset address:0x04

Reset value:0x0000_0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC4F[3:0]				IC3F[3:0]				IC2F[3:0]				IC1F[3:0]			
RW				RW				RW				RW			

Bit	Marking	Functional description
31:16	Reserved	Reserved bit
15:12	IC4F[3:0]	Input Capture 4 Filter
11:8	IC3F[3:0]	Input Capture 3 Filter
7:4	IC2F[3:0]	Input Capture 2 Filter
3:0	IC1F[3:0]	Input Capture 1 Filter This bit-field defines the frequency used to sample TI1 input and the length of the digital filter applied to TI1. The digital filter is made of an event counter in which N consecutive events are needed to validate a transition on the output 0000：No filter, sampled at fDTS；

Bit	Marking	Functional description
		0001：Sampling frequency $f_{SAMPLING}=f_{CK\_INT}$，N=2； 0010：Sampling frequency $f_{SAMPLING}=f_{CK\_INT}$，N=4； 0011：Sampling frequency $f_{SAMPLING}=f_{CK\_INT}$，N=8； 0100：Sampling frequency $f_{SAMPLING}=f_{DTS/2}$，N=6； 0101：Sampling frequency $f_{SAMPLING}=f_{DTS/2}$，N=8； 0110：Sampling frequency $f_{SAMPLING}=f_{DTS/4}$，N=6； 0111：Sampling frequency $f_{SAMPLING}=f_{DTS/4}$，N=8； 1000：Sampling frequency $f_{SAMPLING}=f_{DTS/8}$，N=6； 1001：Sampling frequency $f_{SAMPLING}=f_{DTS/8}$，N=8； 1010：Sampling frequency $f_{SAMPLING}=f_{DTS/16}$，N=5； 1011：Sampling frequency $f_{SAMPLING}=f_{DTS/16}$，N=6； 1100：Sampling frequency $f_{SAMPLING}=f_{DTS/16}$，N=8； 1101：Sampling frequency $f_{SAMPLING}=f_{DTS/32}$，N=5； 1110：Sampling frequency $f_{SAMPLING}=f_{DTS/32}$，N=6； 1111：Sampling frequency $f_{SAMPLING}=f_{DTS/32}$，N=8. *Note: In the current chip version, when ICxF[3:0] = 1, 2 or 3, f_{DTS} in the equation is replaced by CK_INT.*

9.5.3 Interrupt enable register（TIMERx_DIER）

Offset address:0x08

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:8	Reserved	Reserved bit
7	BIE	Break interrupt enable 0: Break interrupt disabled

Bit	Marking	Functional description
		1: Break interrupt enabled
6	TIE	Trigger interrupt enable 0: Trigger interrupt disabled 1: Trigger interrupt enabled
5	COMIE	COM interrupt enable 0: COM interrupt disabled 1: COM interrupt enabled
4	CC4IE	Capture/Compare 4 interrupt enable 0: CC4 interrupt disabled 1: CC4 interrupt enabled
3	CC3IE	Capture/Compare 3 interrupt enable 0: CC3 interrupt disabled 1: CC3 interrupt enabled
2	CC2IE	Capture/Compare 2 interrupt enable 0: CC2 interrupt disabled 1: CC2 interrupt enabled
1	CC1IE	Capture/Compare 1 interrupt enable 0: CC1 interrupt disabled 1: CC1 interrupt enabled
0	UIE	Update interrupt enable 0: Update interrupt disabled 1: Update interrupt enabled

9.5.4 Status registe（TIMERx_SR）

31	30	29	28	27	26	25	24	23	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				CC4OF	CC3OF	CC2OF	CC1OF	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:12	Reserved	Reserved bit
11	CC4OF	Capture/Compare 4 overcapture flag refer to CC1OF description
10	CC3OF	Capture/Compare 3 overcapture flag refer to CC1OF description
9	CC2OF	Capture/Compare 2 overcapture flag refer to CC1OF description
8	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 TIMERx_CCR1 register while CC1IF flag was already set.
7	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	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	COMIF	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
4	CC4IF	Capture/Compare 4 interrupt flag refer to CC1IF description

Bit	Marking	Functional description
3	CC3IF	Capture/Compare 3 interrupt flag refer to CC1IF description
2	CC2IF	Capture/Compare 2 interrupt flag refer to CC1IF description
1	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 TIMERx_CR1 register description). It is cleared by software. 0: No match. 1: The content of the counter TIMERx_CNT matches the content of the TIMERx_CCR1 register. If channel CC1 is configured as input: This bit is set by hardware on a capture. It is cleared by software or by reading the TIMERx_CCR1 register. 0: No input capture occurred 1: The counter value has been captured in TIMERx_CCR1 register (An edge has been detected on IC1 which matches the selected polarity)
0	UIF	Update interrupt flag This bit is set by hardware on an update event. It is cleared by software. 0: No update occurred. 1: Update interrupt pending. This bit is set by hardware when the registers are updated: − At overflow or underflow regarding the repetition counter value (update if repetition counter= 0) and if the UDIS=0 in the TIMERx_CR1 register. − When CNT is reinitialized by software using the UG bit in TIMERx_EGR register, if URS=0 and UDIS=0 in the TIMERx_CR1 register. − When CNT is reinitialized by a trigger event, if URS=0 and UDIS=0 in the TIMERx_CR1 register.

9.5.5 Event generation register（TIMERx_EGR）

Offset address:0x10

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31：8	Reserved	Reserved bit
7	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.
6	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 TIMERx_SR register.
5	COMG	Capture/Compare control update generation This bit can be set by software, it is automatically cleared by hardware 0: No action 1: When CCPC bit is set, it allows to update CCxE, CCxNE and OCxM bits *Note: This bit acts only on channels having a complementary output.*
4	CC4G	Capture/Compare 4 generation refer to CC1G description
3	CC3G	Capture/Compare 3 generation refer to CC1G description
2	CC2G	Capture/Compare 2 generation refer to CC1G description
1	CC1G	Capture/Compare 1 generation This bit is set by software in order to generate an event, it is automatically cleared by hardware.

Bit	Marking	Functional description
		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 TIMERx_CCR1 register. The CC1IF flag is set, the corresponding interrupt is sent if enabled. The CC1OF flag is set if the CC1IF flag was already high.
0	UG	Update generation This bit can be set by software, it is automatically cleared by hardware. 0: No action 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 (TIMERx_ARR) if DIR=1 (downcounting)

9.5.6 Capture/Compare mode register 1（TIMERx_CCMR1）

Offset address:0x14

Reset value:0x0000_0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	OC2M [2:0]			OC2PE	RES	CC2S [1:0]		RES	OC1M [2:0]			OC1PE	RES	CC1S [1:0]	
	RW			RW		RW			RW			RW		RW	

Bit	Marking	Functional description
31:15	RES	Reserved bit
14:12	OC2M[2:0]	Output compare 2 mode
11	OC2PE	Output compare 2 preload enable
10	RES	Reserved bit

9:8	CC2S[1：0]	Capture/Compare 2 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC2 channel is configured as output 01: CC2 channel is configured as input, IC2 is mapped on TI2 10: CC2 channel is configured as input, IC2 is mapped on TI1 11: CC2 channel is configured as input, IC2 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS=2'b11 in TIMERx_CR1 register. *Note: CC2S bits are writable only when the channel is OFF (CC2E = '0' in TIMERx_CCER).*
7	RES	Reserved
6:4	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. 000: Frozen - The comparison between the output compare register TIMERx_CCR1 and the counter TIMERx_CNT has no effect on the outputs. 001: Set channel 1 to active level on match. OC1REF signal is forced high when the counter TIMERx_CNT matches the capture/compare register 1(TIMERx_CCR1) 010: Set channel 1 to inactive level on match. OC1REF signal is forced low when the counter TIMERx_CNT matches the capture/compare register 1 (TIMERx_CCR1). 011: Toggle - OC1REF toggles when TIMERx_CNT=TIMERx_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 TIMERx_CNT<TIMERx_CCR1 else inactive. In downcounting, channel 1 is inactive (OC1REF='0') as long as TIMERx_CNT>TIMERx_CCR1 else active (OC1REF='1') 111: PWM mode 2 - In upcounting, channel 1 is inactive as long as TIMERx_CNT<TIMERx_CCR1 else active. In downcounting, channel 1 is active as long as TIMERx_CNT>TIMERx_CCR1 else inactive. *Note: 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMERx_BDTR register) and CC1S='00' (the channel is configured in output).*

		2: 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.
3	OC1PE	Output compare 1 preload enable 0: Preload register on TIMERx_CCR1 disabled. TIMERx_CCR1 can be written at anytime, the new value is taken in account immediately. 1: Preload register on TIMERx_CCR1 enabled. Read/Write operations access the preload register. TIMERx_CCR1 preload value is loaded in the active register at each update event. *Note: 1: These bits can not be modified as long as LOCK level 3 has been programmed (LOCK bits in TIMERx_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 TIMERx_CR1 register). Else the behavior is not guaranteed.*
2	RES	Reserved
1:0	CC1S[1:0]	Capture/Compare 1 selection This bit-field defines the direction of the channel (input/output) as well as the used input. 00: CC1 channel is configured as output 01: CC1 channel is configured as input, IC1 is mapped on TI1 10: CC1 channel is configured as input, IC1 is mapped on TI2 11: CC1 channel is configured as input, IC1 is mapped on TRC. This mode is working only if an internal trigger input is selected through the TS=2'b11 in TIMERx_CR1 register. *Note: CC1S bits are writable only when the channel is OFF (CC1E = '0' in TIMERx_CCER).*

9.5.7 Capture/Compare mode register 2（TIMERx_CCMR2）

Offset address:0x18

Reset value:0x0000_0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	OC4M [2:0]			OC4PE	RES	CC4S [1:0]		RES	OC3M [2:0]			OC3PE	RES	CC3S [1:0]	
	RW			RW		RW			RW			RW		RW	

Bit	Marking	Functional description
31:15	RES	Reserved bit
14:12	OC4M[2:0]	Output compare 4 mode
11	OC4PE	Output compare 4 preload enable
10	RES	Reserved bit
9:8	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 the TS=2’b11 in TIMERx_CR1 register. *Note: CC4S bits are writable only when the channel is OFF (CC4E = ‘0’ in TIMERx_CCER).*
7	RES	Reserved
6:4	OC3M[2:0]	Output compare 3 mode
3	OC3PE	Output compare 3 preload enable
2	RES	Reserved
1:0	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 the TS=2'b11 in TIMERx_CR1 register. *Note: CC3S bits are writable only when the channel is OFF (CC3E = '0' in TIMERx_CCER).*

9.5.8 Capture/Compare enable register（TIMERx_CCER）

Offset address:0x1C

Reset value:0x0000_0000

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
CC4NP	CC4NE	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E
RW	RW	RW	RW	RW	RW	RW	RW
7	6	5	4	3	2	1	0
CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:16	Reserved	Reserved bit
15	CC4NP	Capture/Compare 4 complementary output polarity refer to CC1NP description
14	CC4NE	Capture/Compare 4 complementary output enable refer to CC1NE description
13	CC4P	Capture/Compare 4 output polarity refer to CC1P description
12	CC4E	Capture/Compare 4 output enable refer to CC1E description
11	CC3NP	Capture/Compare 3 complementary output polarity refer to CC1NP description
10	CC3NE	Capture/Compare 3 complementary output enable refer to CC1NE description

Bit	Marking	Functional description
9	CC3P	Capture/Compare 3 output polarity refer to CC1P description
8	CC3E	Capture/Compare 3 output enable refer to CC1E description
7	CC2NP	Capture/Compare 2 complementary output polarity refer to CC1NP description
6	CC2NE	Capture/Compare 4 complementary output enable refer to CC1NE description
5	CC2P	Capture/Compare 2 output polarity refer to CC1P description
4	CC2E	Capture/Compare 3 output enable refer to CC1E description
3	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 TIMERx_BDTR register) and CC1S="00" (the channel is configured in output).*
2	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	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. *Note: This bit is not writable as soon as LOCK level 2 or 3 has been programmed (LOCK bits in TIMERx_BDTR register)*
0	CC1E	Capture/Compare 1 output enable CC1 channel configured as output: 0: Off - OC1 is not active. OC1 level is then function of MOE, OSSI, OSSR,

Bit	Marking	Functional description
		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. 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 (TIMERx_CCR1) or not. 0: Capture disabled 1: Capture enabled

Table9-1 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),
		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 OCREF+ 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，	Off-State (output enabled with inactive

					OCx = OCxREF xor CCxP OCx_EN = 1	state) OCxN = CCxNP，OCxN_EN = 1
		1	1	1	OCxREF + Polarity + dead-time OCx_EN = 1	Complementary to OCREF + Polarity + dead-time OCxN_EN = 1
0	0	x	0	0	Output Disabled (not driven by the timer) Asynchronously: OCx = CCxP，OCx_EN = 0，OCxN = CCxNP，OCxN_EN = 0； Then if the clock is present: OCx=OISx and OCxN=OISxN after a dead-time, assuming that OISx and OISxN do not correspond to OCx and OCxN both in active state.	
	0		0	1		
	0		1	0		
	0		1	1		
	1		0	0		
	1		0	1	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		1	0		
	1		1	1		

9.5.9 Count register（TIMERx_CNT）

Offset address:0x20

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:16	Reserved	Reserved bit
15:0	CNT	Counter value

9.5.10 Prescaler register（TIMERx_PSC）

Offset address:0x24

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:16	Reserved	Reserved bit
15:0	PSC	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 TIMERx_EGR register or through trigger controller when configured in “reset mode”).

9.5.11 Auto-reload register（TIMERx_ARR）

Offset address:0x28

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:16	Reserved	Reserved bit
15:0	ARR	Auto-reload value ARR is the value to be loaded in the actual auto-reload register. Refer to Section 9.4.1Time-base unit for more details about ARR update and behavior.The counter is blocked while the auto-reload value is null.

9.5.12 Repetition counter register（TIMERx_RCR）

Offset address:0x2C

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:16	Reserved	Reserved bit
7:0	REP	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 TIMERx_RCR register is not taken in account until the next repetition update event. It means in PWM mode (REP+1) corresponds to: – the number of PWM periods in edge-aligned mode – the number of half PWM period in center-aligned mode.

9.5.13 Capture/Compare register 1（TIMERx_CCR1）

Offset address:0x30

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1															
RW															

Bit	Marking	Functional description
31:16	Reserved	Reserved bit
15:0	CCR1	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 TIMERx_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 TIMERx_CNT and signaled on OC1 output. If channel CC1 is configured as input: CCR1 is the counter value transferred by the last input capture 1 event (IC1). The TIMERx_CCR1 register is read-only and cannot be programmed.

9.5.14 Capture/Compare register 2（TIMERx_CCR2）

Offset address:0x34

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
15:0	CCR2	Capture/Compare 2 value If channel CC2 is configured as output: CCR2 is the value to be loaded in the actual capture/compare 2 register (preload value). It is loaded permanently if the preload feature is not selected in the TIMERx_CCMR2 register (bit OC2PE). Else the preload value is copied in the active capture/compare 2 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMERx_CNT and signaled on OC2 output. If channel CC2 is configured as input: CCR2 is the counter value transferred by the last input capture 2 event (IC2). The TIMERx_CCR2 register is read-only and cannot be programmed.

9.5.15 Capture/Compare register 3（TIMERx_CCR3）

Offset address:0x38

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
15:0	CCR3	Capture/Compare 3 value If channel CC3 is configured as output: CCR3 is the value to be loaded in the actual capture/compare 3 register (preload value). It is loaded permanently if the preload feature is not selected in the TIMERx_CCMR3 register (bit OC3PE). Else the preload value is copied in the active capture/compare 3 register when an update event occurs. The active capture/compare register contains the value to be compared to the counter TIMERx_CNT and signaled on OC3 output. If channel CC3 is configured as input: CCR3 is the counter value transferred by the last input capture 3 event (IC3). The TIMERx_CCR3 register is read-only and cannot be programmed.

9.5.16 Capture/Compare register 4（TIMERx_CCR4）

Offset address:0x3C

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4															
RW															

Bit	Marking	Functional description
15:0	CCR4	Capture/Compare 4 value If channel CC4 is configured as output: CCR4 is the value to be loaded in the actual capture/compare 4 register (preload value). It is loaded permanently if the preload feature is not selected in the TIMERx_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 TIMERx_CNT and signaled on OC1 output. If channel CC4 is configured as input: CCR4 is the counter value transferred by the last input capture 4 event (IC4). The TIMERx_CCR4 register is read-only and cannot be programmed.

9.5.17 Break and dead-time register（TIMERx_BDTR）

Offset address:0x40

Reset value:0x0000_0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK		DTG							
RW	RW	RW	RW	RW	RW	RW		RW							

Bit	Marking	Functional description
31:16	Reserved	Reserved bit
15	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. 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 TIMERx_CCER register).
14	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) *Note: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMERx_BDTR register).*
13	BKP	Break polarity 0: Break input BRK is active low 1: Break input BRK is active high *Note: This bit can not be modified as long as LOCK level 1 has been programmed (LOCK bits in TIMERx_BDTR register).*
12	BKE	Break enable 0: Break inputs (BRK and CSS clock failure event) disabled 1: Break inputs (BRK and CSS clock failure event) enabled *Note: This bit cannot be modified when LOCK level 1 has been programmed (LOCK*

Bit	Marking	Functional description
		bits in TIMERx_BDTR register).
11	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 (Section 9.5.8 Capture/Compare enable register（TIMERx_CCER）). 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 *Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIMERx_BDTR register).*
10	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(Section 9.5.8 Capture/Compare enable register（TIMERx_CCER）)。 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) *Note: This bit can not be modified as soon as the LOCK level 2 has been programmed (LOCK bits in TIMERx_BDTR register)。*
9:8	LOCK	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 TIMERx_BDTR register, OISx and OISxN bits in TIMERx_CR1 register and BKE/BKP/AOE bits in TIMERx_BDTR register can no longer be written. 10: LOCK Level 2 = LOCK Level 1 + CC Polarity bits (CCxP/CCxNP bits in TIMERx_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 TIMERx_CCMRx registers, as long as the related channel is configured in output through the CCxS bits) can no longer be written.

Bit	Marking	Functional description
		Note: The LOCK bits can be written only once after the reset. Once the TIMERx_BDTR register has been written, their content is frozen until the next reset.
7:0	DTG	Dead-time generator setup This bit-field defines the duration of the dead-time inserted between the complementary outputs. DT correspond to this duration. DTG[7:5] = 0xx => DT = DTG[7:0] $\times T_{dtg}$， $T_{dtg} = T_{DTS}$； DTG[7:5] = 10x => DT = (64+DTG[5:0]) $\times T_{dtg}$， $T_{dtg} = 2 \times T_{DTS}$； DTG[7:5] = 110 => DT = (32+DTG[4:0]) $\times T_{dtg}$， $T_{dtg} = 8 \times T_{DTS}$； DTG[7:5] = 111 => DT = (32+DTG[4:0]) $\times T_{dtg}$， $T_{dtg} = 16 \times T_{DTS}$； Example if TDTS=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 *Note: This bit-field can not be modified as long as LOCK level 1, 2 or 3 has been programmed (LOCK bits in TIMERx_BDTR register).*

9.5.18 Timer clock enable register（TIMER_CLKEN）

Absolute address：0x3000_0100

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							timer2_clken_reg	Reserved						timer1_clken_reg	
							RW							RW	

Bit	Marking	Functional description
31:9	Reserved	Reserved bit
8	timer2_clken_reg	Timer2 control bit for the clock 0: turn off the clock of timer2 1: turn on the clock of timer2
7:1	Reserved	Reserved bit
0	timer1_clken_reg	Timer1 control bit for the clock 0: turn off the clock of timer1, 1: turn on the clock of timer1

9.6 TIM1&TIM2 register mapping

TIMER1 registers list

Base address：0x3000_0018

Register	Offset address	Description
TIMER1_CR1	0x00	Control register
TIMER1_ICF	0x04	Filter register
TIMER1_IER	0x08	Interrupt Enable Register
TIMER1_SR	0x0C	Status register
TIMER1_EGR	0x10	Event Generation Register
TIMER1_CCMR1	0x14	Capture/Compare Mode Register 1
TIMER1_CCMR2	0x18	Capture/Compare Mode Register 2
TIMER1_CCER	0x1C	Capture/Compare Enable Register
TIMER1_CNT	0x20	Count register

TIMER1_PSC	0x24	Prescaler register
TIMER1_ARR	0x28	Auto-Reload Register
TIMER1_RCR	0x2C	Repetition Counter Register
TIMER1_CCR1	0x30	Capture/Compare register 1
TIMER1_CCR2	0x34	Capture/Compare Register 2
TIMER1_CCR3	0x38	Capture/Compare Register 3
TIMER1_CCR4	0x3C	Capture/Compare Register 4
TIMER1_BDTR	0x40	Brake and dead-time Register

TIM2 registers list

Base address：0x3000_0098

Register	Offset address	Description
TIMER2_CR1	0x00	Control register
TIMER2_ICF	0x04	Filter register
TIMER2_IER	0x08	Interrupt Enable Register
TIMER2_SR	0x0C	Status register
TIMER2_EGR	0x10	Event Generation Register
TIMER2_CCMR1	0x14	Capture/Compare Mode Register 1
TIMER2_CCMR2	0x18	Capture/Compare Mode Register 2
TIMR2_CCER	0x1C	Capture/Compare Enable Register
TIMERE2_CNT	0x20	Count register
TIMER2_PSC	0x24	Prescaler register
TIMER2_ARR	0x28	Auto-Reload Register
TIMER2_RCR	0x2C	Repetition Counter Register
TIMER2_CCR1	0x30	Capture/Compare register 1
TIMER2_CCR2	0x34	Capture/Compare Register 2
TIMER2_CCR3	0x38	Capture/Compare Register 3
TIMER2_CCR4	0x3C	Capture/Compare Register 4
TIMER2_BDTR	0x40	Brake and dead-time Register
TIMER_CLKEN	0x68	Timer Clock enable register

10 WUP

10.1 Introduction

The wup module is a wakeup module, the clock source is 3k and once per wup_clk clock.When wup_data = 0 does not work and no irq generated. When wup_data is not equal to 0, an interrupt is gengrated every wup_data+1 clock cycles, and the counter reloads the value of wup_data. It can be used for the wake-up in low-power mode.

10.2 Register description

10.3 Wup data register（wup_data）

Offset address:0x00

Reset value:0x0000_0000

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

Bit	Marking	Functional description
31:16	Reserved	Reserved bit
15:0	wup_data	Wakeup data register, wup_data is actually a down counter

The wup module is a wakeup module, essentially a down counter. The clock source is 3K counts once per clock, when wup_data = 0 it does not count, otherwise it generates an interrupt every wup_data+1 clock cycles. When the interrupt is enabled, it can be used to wake up the pmu. A reset from the pmu will not reset the irq.

10.3.1 Wup interrupt enable register（wup_irq_en）

Offset address:0x04

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:1	Reserved	Reserved bit
0	wup_irq_en	Wup interrupt enableBit，1：enable，0：disable

10.3.2 Wup interrupt register（wup_irq）

Offset address:0x08

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:1	Reserved	Reserved bit
0	wup_irq	Interrupt flag bit, 1: interrupt has generated, 0: no interrupt generate, write 0 to clear interrupt. When the interrupt is enabled, it can be used to wake up in low-power mode.A reset from low-power mode will not reset irq.

10.4 Register mapping

WUP register list

Base address:0x3000_0610

Register	Offset address	Description
wup_data	0x00	WUP Data Register
wup_irq_en	0x04	WUP Interrupt enable Register
wup_irq	0x08	WUP Interrupt Register

11 Analog/digital conversion（ADC）

11.1 Introduction

The CSM32RV20 includes a fast, high-precision ADC with an integrated high-precision 1.2V reference source that supports 13/14/15/16-bit resolution and balances resolution and conversion speed. The voltage VDD required for ADC operation is greater than 2.5 V.

Note:

(1) When using ADC, it is recommend to set ADC_CCR[5] to 1, otherwise it will increase the power consumption.

(2) The resolution is initialized and the user is not allowed to change;

(3) When PGA input is selected as the channel of ADC test (adc_cha_sel= 0b101010) and the ADC is turned on(adc_on= 1), the PA11 and PA13 cannot be used as GPIO.

11.2 Functional description

11.2.1 Main features

- 13-bit resolution, requires 29 ADC clock cycles to complete a conversion
- 14-bit resolution, requires 45 ADC clock cycles to complete a conversion
- 15-bit resolution, requires 77 ADC clock cycles to complete a conversion
- 16-bit resolution, requires 141 ADC clock cycles to complete a conversion
- The interrupt is generated automatically after ADC conversion finished
- The adc clock has the same clock source as the bus clock, and supports divided by 1/2/4/8
- ADC sampling clock recommended 4 MHz, maximum 8 MHz
- Supports single-shot mode and continuous mode
- Programmable transition interval in continuous mode
- Software trigger and PA14 (ADC_TRI) trigger are supported. PA14 cannot be used as an ADC measurement channel.
- The measurable voltage range is 0 ~ VDD (VDD <4.8V)
- Support external reference, external reference high voltage terminal is less

than VDD+ 0.7v, external reference low voltage terminal is greater than -0.7v. The difference between the external reference high voltage terminal and the low voltage terminal is less than VDD 1v.

- measurement channels are optional, 9 channels of which are measured from external input of the chip inputs, 1 channel is used to measurement voltage VDD of chip and 1 channel is used to chip temperature measurement (measuring PTAT)
- Supports multiplying the voltage to be measured by 1/4
- Supports PGA input. When PGA input is selected, you can select the reference output of the ADC: REFP (PA13 is set as analog mode) and REFN(PA11 is set as analog mode) supply power for PGA input, so that you can get higher precision.It also can choose external power supply, but the accuracy is lower.

Note: REFP only can pull the current, REFN only can sink the current. The load capacity is limited, and its maximum current is 120uA.

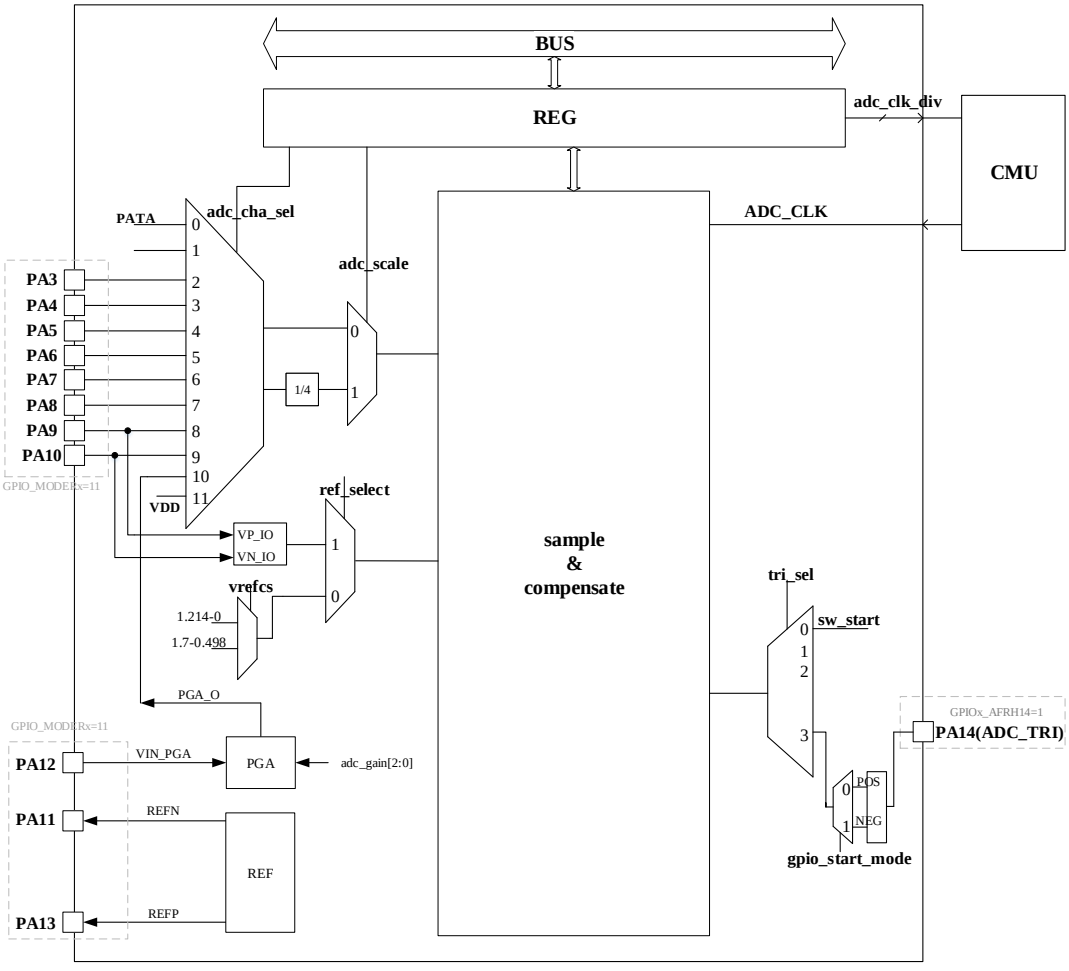


Figure11-1 ADC diagram

11.2.2 Conversion timing

After adc_on is set to 1, the ADC will be turned on and the conversion can be started. Before the ADC enters the conversion, it needs a period of stabilization time t_{STAB} .When the ADC enters the conversion state, the adc_state bit will be set. The ADC will be triggered to enter the conversion by setting sw_start or ADC_TRI (PA14). After the conversion time (sampling time set by the user program), the eoc interrupt flag bit will be set to 1 and the ADC conversion result will be stored in the ADC_DR register. Note that the signal needs to be resynchronized when transferring from the bus clock domain to the ADCCLK clock domain, thus creating a delay.

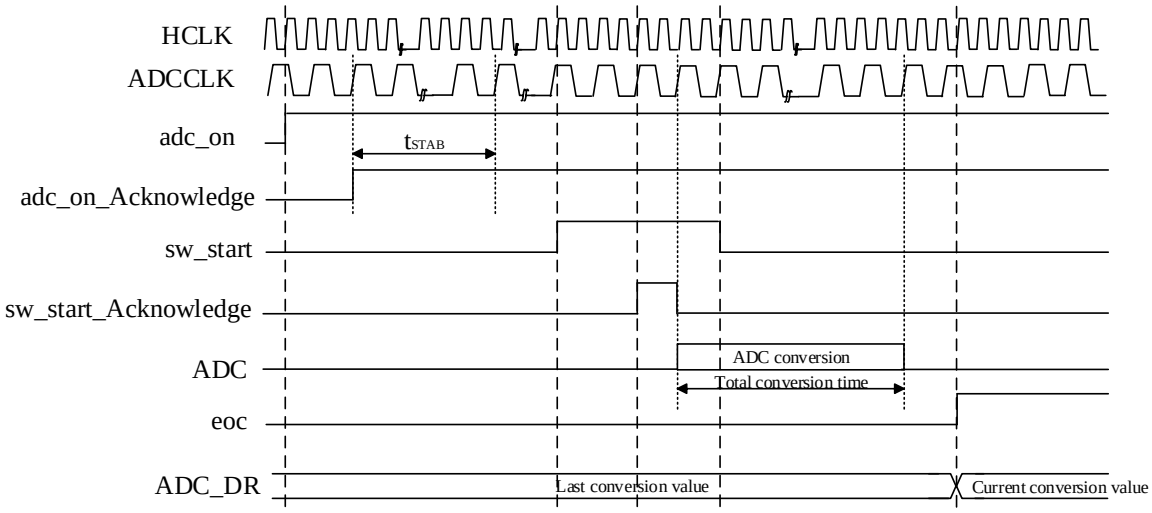


Figure11-2 ADC timing diagram

11.2.3 Internal PATA temperature change curve

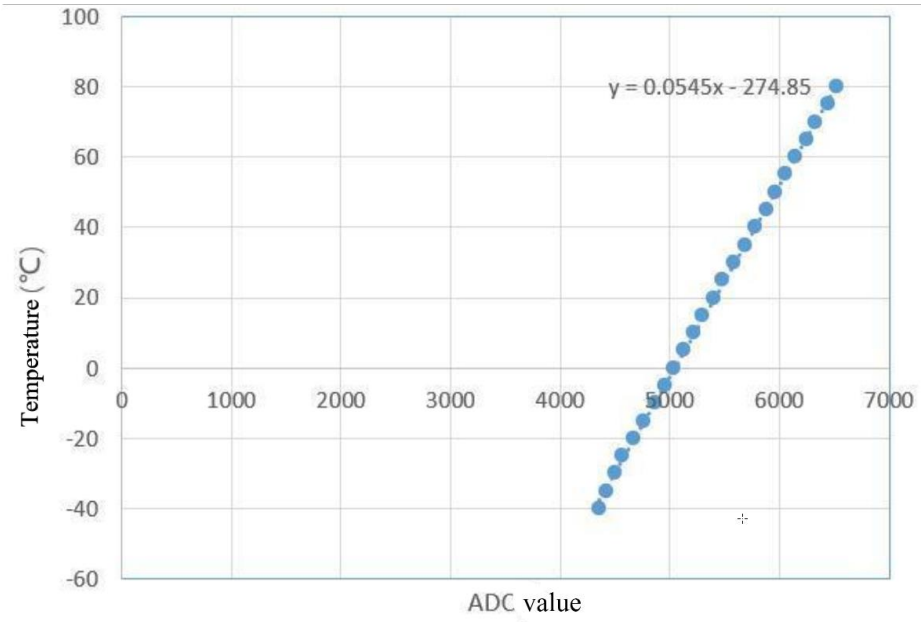


Figure11-3 ADC 值随温度变化 Figure

11.3 Register description

11.3.1 ADC Status register（ADC_ISR）

Base address:0x3000_0280

Offset address:0x00

Reset value:0x0000_0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													eoc	adc_state	adc_ready
													RC	R	R

Bit	Marking	Functional description
31:3	Reserved	Reserved bit
2	eoc	Flag bit for ADC conversion completion 0: Conversion has not been completed; 1: Conversion capture has been completed.

		The interrupt can be cleared by writing 0 to bit eoc or reading the data register
1	adc_state	ADC conversion operational status flag bit 1: ADC is converting; 0: ADC is idle
0	adc_ready	Status flag bit of ADC on 1: ADC is on; 0: ADC is off Adc_on opens 5 us. The flag adc_ready is set to 1 after the power is stabilized.

11.3.2 ADC interrupt enable register（ADC_IER）

Offset address:0x04

Reset value:0x0000_0000

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

Bit	Marking	Functional description
31:3	Reserved	Reserved bit
2	eocie	Current ADC interrupt enable bit 0: The ADC will not generate an interrupt to pass to the CLIC; 1: The ADC generates an interrupt to the CLIC when the EOC signal is set to 1.
1:0	Reserved	Reserved bit

11.3.3 ADC control register（ADC_CR）

Offset address:0x08

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:1	Reserved	Reserved bit
0	sw_start	ADC Software Trigger Control Bit One-shot mode: Write 1 to start conversion, automatically clear the zero after one conversion and generate interrupt and update ADC_DR at the same time. Write 0 to end early. Continuous Mode: Write 1 to start conversion, generate interrupt and update ADC_DR when conversion is completed and start the next conversion.Write 0 to end conversion.

11.3.4 ADC channel select register（ADC_SEL）

Offset address:0x0C

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:4	Reserved	Reserved bit
3:0	adc_cha_sel	channel select

		The GPIO needs to be configured in analog mode (GPIO_MODEx = 2'b11) in the following cases: 0000：Measurement of internal PATA voltage value, can be used for collect the chip temperature 0001：Reserved； 0010：measure channel PA3； 0011：measure channel PA4； 0100：measure channel PA5； 0101：measure channel PA6； 0110：measure channel PA7； 0111：measure channel PA8； 1000：measure channel PA9； 1001：measure channel PA10； 1010：measure channel PA12（PGA input）； Note: 1) In order to achieve higher precision, you can select the reference output of the ADC supply power for PGA input PA13(REFP) and PA11(REFN) PA11 are configured in analog mode.It also can choose external power supply, but the accuracy is lower. 2) When PGA input is selected as the channel of ADC measurement (adc_cha_sel= 0b101010) and the ADC is turned on(adc_on= 1), the PA11 and PA13 cannot be used as GPIO 1011：VDD，when select this channel,ADC_CCR[19] must be configured to 1； other：Reserved。

11.3.5 ADC Data register（ADC_DR）

Offset address:0x10

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:15	Reserved	Reserved bit
14:0	data	The data collected by the adc, when the eoc signal is high, the software can read the ADC_CFG[adc_bits_ctrl] to control the corresponding resolution data.

11.3.6 ADC general control register（ADC_CCR）

Offset address:0x14

Reset value:0x0000_0080

31	30	29	28	27	26	25	24	23	22
Reserved									

21	20	19	18	17	16	15	14	13	12
Reserved		adc_scale	pga_gain			Reserved		vrefcs	ref_select
		RW	RW					RW	RW

10	9	8	7	6	5	4	3	2	1	0
del			adc_clk_div		shd_vsample	gpio_start_mode	tri_sel		adc_mode	adc_on
RW			RW		RW	RW	RW		RW	RW

Bit	Marking	Functional description
31:22	Reserved	Reserved bit
21:20	Reserved	Must be 00
19	adc_scale	Selects the ADC internal channel gain 0: Option 1;

		1: Select 1/4 and multiply the input voltage by 1/4 to keep the detected voltage within range
18:16	pga_gain	Used to measure pga input PA12 is the input of PGA operation amplifier(PA12 needs to be configured in analog mode) 000：Operation amplifier gain is 1(by default)； 001：Operation amplifier gain is 2； 010：Operation amplifier gain is 4； 011：Operation amplifier gain is 8； 100：Operation amplifier gain is 16； 101：Operation amplifier gain is 32； 110：Operation amplifier gain is 64； 111：Operation amplifier gain is 128
15:14	Reserved	Reserved bit
13	vrefcs	Internal reference voltage source selection bit, must be written 0; The reference size is 1.2V and the measurement range is 0-1.2V (above 1.2V, use 1/4 gain, which is adc_scale=1).
12	ref_select	Select internal or external reference 1： external reference；　　0： internal reference Note: External reference high voltage terminal is less than VDD+ 0.7v, external reference low voltage terminal is greater than -0.7v. The difference between the external reference high voltage terminal and the low voltage terminal is less than VDD 1v.
11:8	delay_sel	ADC Delay between two adjacent conversions in continuous conversion mode 0000：No delay； 0001：2^0 ADC clock； 0010：2^1 ADC clock； … 1111：2^{14} ADC clock
7:6	adc_clk_div	Adc clock frequency selection 00: no frequency division;　01:2 division　10:4 division　11:8 division Note: It is recommended to set adc_clk_divat when adc_on is 0.
5	Reserved	Reserved
4	gpio_start_mode	GPIO Trigger Mode Selection

		0：GPIO rising edge trigger Single mode: rising edge triggered once; Continuous mode: rising edge trigger, falling edge end sampling. 1：GPIO falling edge trigger Single mode: falling edge triggered once; Continuous mode: falling edge trigger, rising edge end sample
3:2	tri_sel	ADC Trigger Signal Source Selection 00: ADC_CR[0] Software trigger to control ADC conversion; 01/10: Reserved; 11: GPIO-triggered ADC conversion
1	adc_mode	ADC Sampling Mode 0: Single mode 1: Continuous mode
0	adc_on	ADC Power Switch 0: off 1: on Note: When PGA input is selected as the channel of ADC measurement (adc_cha_sel= 0b101010) and the ADC is turned on(adc_on= 1), the PA11 and PA13 cannot be used as GPIO.

11.3.7 ADC resolution register（ADC_CFG）

Address：0x3000_0410

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:30	adc_bits_ctrl	ADC resolution bit 11：ADC resolution 16Bit，data 15bit； 10：ADC resolution 15Bit，data 14bit；

		01：ADC resolution 14Bit，data 13bit； 00：ADC resolution 13Bit，data 12bit
29:21	Reserved	Reserved bit
20:0	adc_data	ADC data in complementary form

11.4 Register mapping

ADC register list

Base address:0x3000_0280

Register	Offset address	Description
ADC_ISR	0x00	ADC Status Register
ADC_IER	0x04	ADC Interrupt enable Register
ADC_CR	0x08	ADC Control Register
ADC_SEL	0x0C	ADC channel select Register
ADC_DR	0x10	ADC Data Register
ADC_CCR	0x14	ADC general control Register
ADC_CFG	0x3000_0410	ADC resolution Register

12 I2C interface

12.1 Introduction

I2C (inter-integrated circuit) bus Interface serves as an interface between the microcontroller and the serial I2C bus. It provides master capability, and controls all I2C bus-specific sequencing, protocol, arbitration and timing. It supports the standard mode (Sm, up to 100 kHz) and fast mode (Fm, up to 400 kHz).

12.1.1 Main features

- Parallel-bus/I2C protocol converter
- I2C Master features:
  - Clock generation
  - Start and Stop generation
- Generation and detection of 7-bit addressing and General Call
- Supports different communication speeds:
  - Standard Speed (up to 100 kHz)
  - Fast Speed (up to 400 kHz)
- Status flags:
  - Transmitter/Receiver mode flag
  - End-of-Byte transmission flag
  - I2C busy flag
- Error flags:
  - Arbitration lost condition for master mode
  - Acknowledgment failure after address/ data transmission
  - Detection of misplaced start or stop condition
- 2 Interrupt vectors
  - 1 Interrupt for successful address/ data communication
  - 1 Interrupt for error condition

12.2 Functional description

In Master mode, the I2C interface initiates a data transfer and generates the clock signal. A serial data transfer always begins with a Start condition and ends with a Stop condition. The data transfer can only start when the bus is in the "not busy" state. During the data transfer, the data lines must remain stable as long as the clock line is high, otherwise any change on the data lines is treated as a "start" or "stop" signal. Figure 12-1 shows the defined bus states.

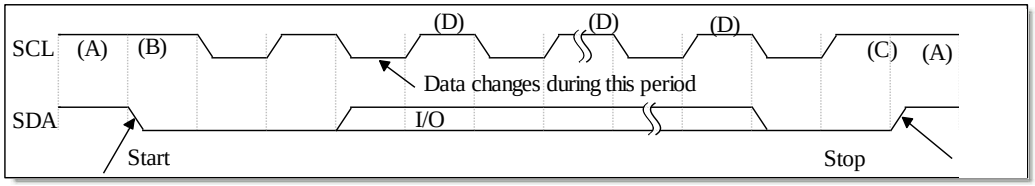


Figure12-1 I2C 2-wire series bus

I2C has the following four main operating states A, B, C and D:

(1) Bus Non-Busy State (section A): Both the data line (SDA) and the clock line (SCL) remain high in this section.

(2) Initiate data transfer (section B): When the clock line (SCL) is high, the falling edge of the data line (SDA) from high to low is considered to be a "start" signal. The other commands after the "start" signal are valid.

(3) Stop data transmission (C section): When the clock line (SCL) is high, the rising edge of the data line (SDA) from low to high is considered a "stop" signal. With the "stop" signal, all external operations are terminated.

(4) Data Valid (section D): After the "Start" signal, the data line is stable while the clock line (SCL) is high, and the state of the data line is the data to be transmitted. Changes to the data on the data line (SDA) must be accomplished while the clock line is low, with each data taking up one clock pulse. Each data transfer starts with a "start" signal and ends with a "stop" signal.

(5) Acknowledge signal: After receiving a byte of data, it is usually necessary to send an acknowledge signal. After sending a byte of data, it is usually necessary to receive an acknowledge signal. The sender releases the SDA line during the acknowledge clock pulse, and the receiver pulls the SDA line low and holds it at 0 during the high pulse of SCL, as shown in Figure 12-2. The I2C read/write controller must have generated an additional clock pulse

associated with this acknowledge bit. In a read operation, the read/write controller does not generate an acknowledge bit for the last byte of the I2C completion, but there is an end signal.

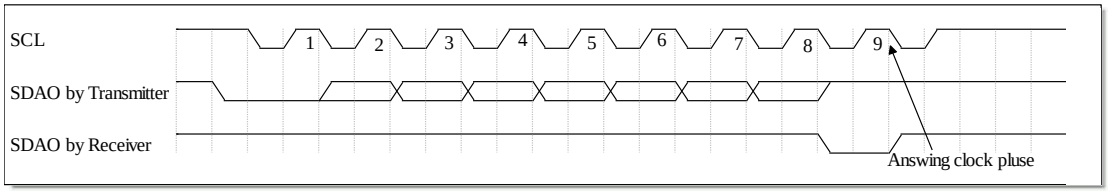


Figure12-2 Acknowledge timing diagram

Configuration steps of the Write operation：

1) With the fm bit, the frequency of data transmission is defined
2) Select whether to divided or not via I2C_CTRL[i2c_clkdiv], and I2C_CTRL[i2c_saddr] defines the slave address.
3) I2C_CTRL[i2c_r_wn]= 0 is defined as a write operation,
   I2C_CTRL[i2c_maddr] defines the memory cell address,
   I2C_CTRL[i2c_data] defines the data to be sent.

Sender: After the start condition is satisfied, serial write slave address (I2C_CTRL[i2c_saddr]), write operation (I2C_CTRL[i2c_r_wn]=0),acknowledge, memory cell address(I2C_CTRL[i2c_maddr]), acknowledge, the data to be sent (I2C_CTRL[i2c_data]), acknowledge and the stop signal from the SDA line in sequence. Refer to Figure 12-3 I2C Write Data Timing Diagram.

Each acknowledge will only continue if it receives a correct reply from the receiver, otherwise it will keep waiting for an acknowledge from the receiver.

The i2c_ready flag will be set when data transmission is complete. If the i2c_ready_en in the I2C_CTRL register is set, an interrupt will be generated.

Configuration steps of the Read operation：

1) With the fm bit, the frequency of data transmission is defined
2) Select whether to divided or not via I2C_CTRL[i2c_clkdiv], and I2C_CTRL[i2c_saddr] defines the slave address.
3) I2C_DATA[i2c_r_wn] = 0 is defined as a write operation,
   I2C_DATA[i2c_r_wn] = 1 is defined as a read operation,
   I2C_DATA[i2c_maddr] defines the memory cell address,
   and reads the data of I2C_DATA[i2c_data].

After the start condition is satisfied, serial write slave address

(I2C_CTRL[i2c_saddr]), write operation (I2C_CTRL[i2c_r_wn]=0),acknowledge, memory cell address(I2C_CTRL[i2c_maddr]), acknowledge. Restart, slave address (I2C_CTRL[i2c_saddr]), read operation (I2C_CTRL[i2c_r_wn]=1),acknowledge. Receiver replies with read data I2C_DATA[i2c_data], non-acknowledge, and finally a stop signal. Refer to Figure 12-4 and Figure 12-5.

The i2c_ready flag will be set when data reception is complete, and an interrupt will be generated if the i2c_ready_en bit in the I2C_CTRL register is set. When reading the I2C_DATA register, the I2C device returns the received data. Reading the I2C_DATA register will clear the i2c_ready bit.

During data transfer, if there is no acknowledge or other errors are generated, the i2c_error flag will be set, and an interrupt will be generated if the i2c_error_en bit in the I2C_CTRL register is set. Figure 12-2 shows the I2C acknowledge timing diagram, and Figure 12-3 and Figure 12-4 show the I2C write data and read data at specified address timing diagrams respectively.

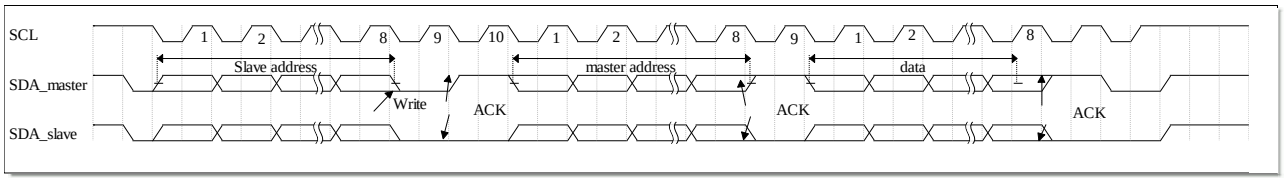


Figure12-3 I2C write data timing

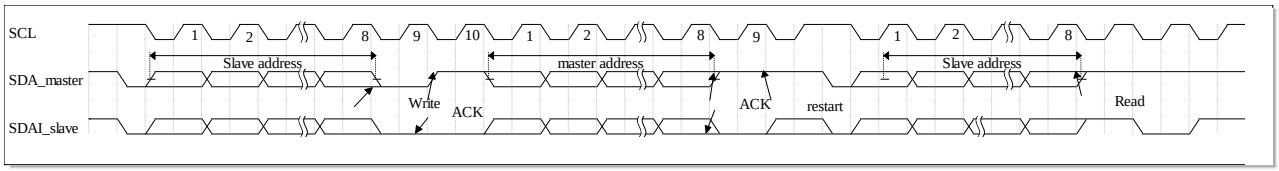


Figure12-4 I2C read data at a specified address

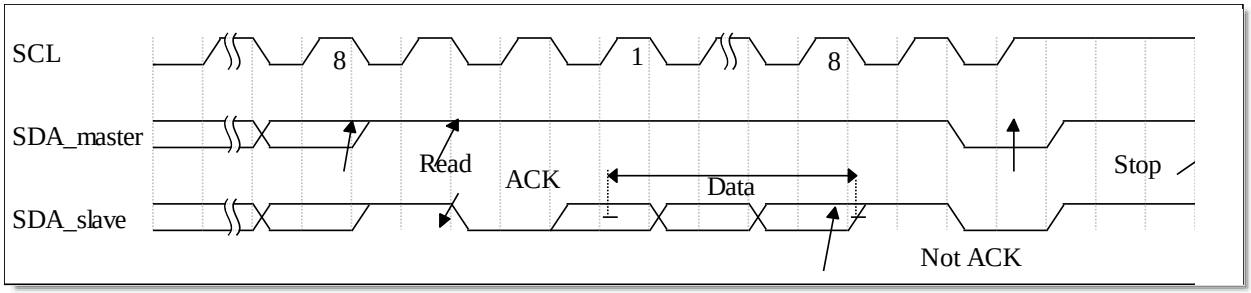


Figure12-5 I2C read data at a specified address (timing follows the previous figure)

12.3 I2C register description

Base address: 0x3000_0004

12.3.1 Status register（I2C_STATUS）

Offset address:0x00

Reset value:0x0000_0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											i2c_error	Reserved			i2c_ready
											RW				RW

Bit	Marking	Functional description
16:5	Reserved	Reserved bit
4	i2c_error	Error flag bit; 1: i2c error occurred, 0: normal operation
3:1	Reserved	Reserved bit
0	i2c_ready	Interrupt flag bit; 1: operation is completed, 0: operation is in progressing

12.3.2 Control register（I2C_CTRL）

Offset address:0x04

Reset value:0x0000_0000

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			i2c_clkdiv	Reserved		i2c_error_en	i2c_ready_en
			RW			RW	RW
7	6	5	4	3	2	1	0
fm	i2c_saddr						
RW	RW						

Bit	Marking	Functional description
31:13	Reserved	Reserved bit
12	i2c_clkdiv	0：No divided； 1：2 divided frequency
11:10	Reserved	Reserved bit
9	i2c_error_en	i2c_error interrupt enable，0: disable, 1: enable
8	i2c_ready_en	i2c_ready flag interrupt enable，0: disable，1: enable
7	fm	Frequency , 0: 100KHz 1: 400KHz
6:0	i2c_saddr	Slave address Note: The slave address of I2C is separated from the read/write bit (i2c_r_wn), users need to shift the original address left by 1 bit. For example, if the slave address is 0x50, the address written to i2c_saddr should be 0x50<<1, i.e. 0xA0.

12.3.3 DATA register（I2C_DATA）

Offset address:0x08

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															i2c_r_wn
															RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i2c_maddr								i2c_data							
RW								RW							

Bit	Marking	Functional description
31:17	Reserved	Reserved bit
16	i2c_r_wn	0: write 1: read
15:8	i2c_maddr	Memory cell address
7:0	i2c_data	i2c data

12.4 Register mapping

I2C register list

Base address:0x3000_0004

Register	Offset address	Description
I2C_STATUS	0x00	I2C status register
I2C_CTRL	0x04	I2C control register
I2C_DATA	0x08	I2C data register

13 Serial peripheral interface（SPI1）

13.1 Introduction

SPI is the abbreviation for Serial Peripheral Interface. The serial peripheral interface (SPI) allows half/ full-duplex, synchronous, serial communication with external devices. The interface can be configured as the master and in this case it provides the communication clock (SCK) to the external slave device. The interface is also capable of operating in multimaster configuration.

It may be used for a variety of purposes, including simplex synchronous transfers on two lines with a possible bidirectional data line or reliable communication using CRC checking.

13.1.1 Main features

- Full-duplex synchronous transfers on three lines
- 8-bit transfer frame format selection
- Multimaster mode capability
- 8 master mode baud rate prescalers
- Programmable clock polarity and phase
- Dedicated flags with interrupt capability
- SPI bus busy status flag

13.2 Functional description

Usually, the SPI is connected to external devices through four pins:

- MISO: Master In / Slave Out data. This pin can be used to transmit data in slave mode and receive data in master mode.
- MOSI: Master Out / Slave In data. This pin can be used to transmit data in master mode and receive data in slave mode.
- SCK: Serial Clock output for SPI masters and input for SPI slaves
- CSN: Simulated by GPIO.Slave select. This is an optional pin to select a slave device. This pin acts as a ‘chip select’ to let the SPI master communicate with slaves individually and to avoid contention on the data lines. Slave CSN inputs

can be driven by standard IO ports on the master device. The CSN pin may also be used as an output if enabled and driven low if the SPI is in master configuration.

The communication is always initiated by the master. When the master device transmits data to a slave device via the MOSI pin, the slave device responds via the MISO pin. This implies full-duplex communication with both data out and data in synchronized with the same clock signal (which is provided by the master device via the SCK pin).

Four possible timing relationships may be chosen by software, using the CPOL and CPHA bits in the SPI_CTRL register. The CPOL (clock polarity) bit controls the steady state value of the clock when no data is being transferred. This bit affects both master and slave modes. If CPOL is reset, the SCK pin has a low-level idle state. If CPOL is set, the SCK pin has a high-level idle state.

If the CPHA (clock phase) bit is set, the second edge on the SCK pin (falling edge if the CPOL bit is reset, rising edge if the CPOL bit is set) is the MSBit capture strobe. Data are latched on the occurrence of the second clock transition. If the CPHA bit is reset, the first edge on the SCK pin (rising edge if CPOL bit is set, falling edge if CPOL bit is reset) is the MSBit capture strobe. Data are latched on the occurrence of the first clock transition.

The combination of the CPOL (clock polarity) and CPHA (clock phase) bits selects the data capture clock edge. Figure13-1 to Figure13-4 show an SPI transfer with the four combinations of the CPHA and CPOL bits.

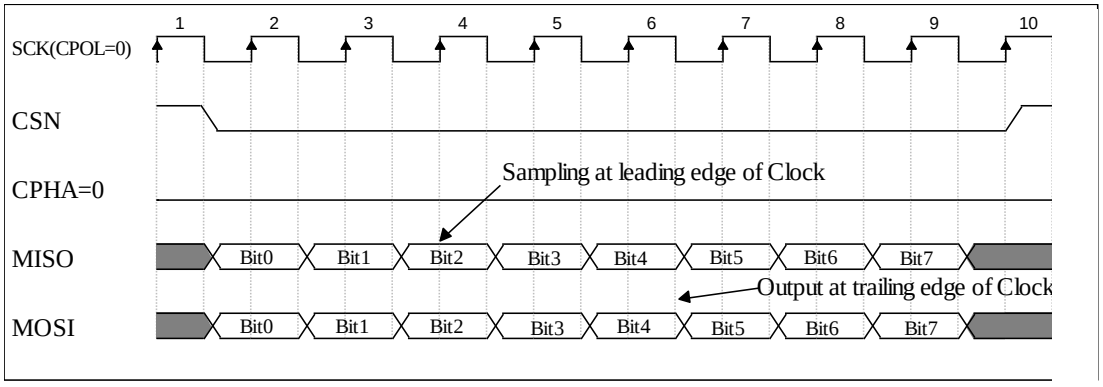


Figure13-1 spi mode0 timing

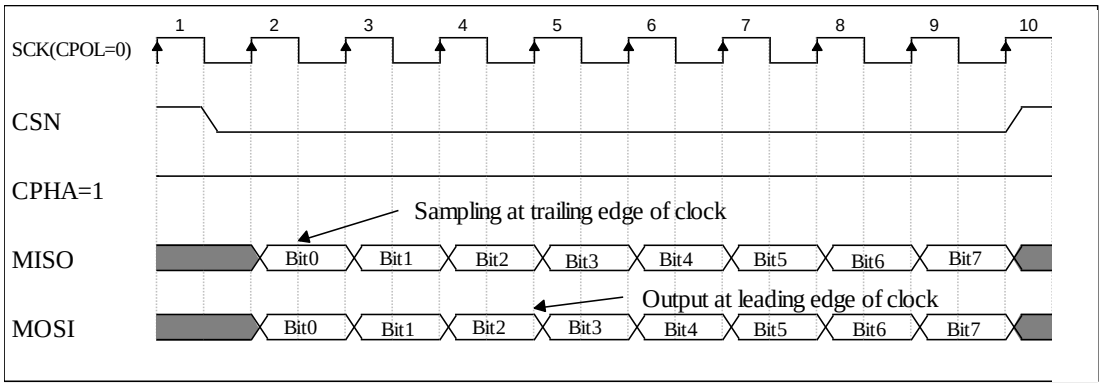


Figure13-2 spi mode1 timing

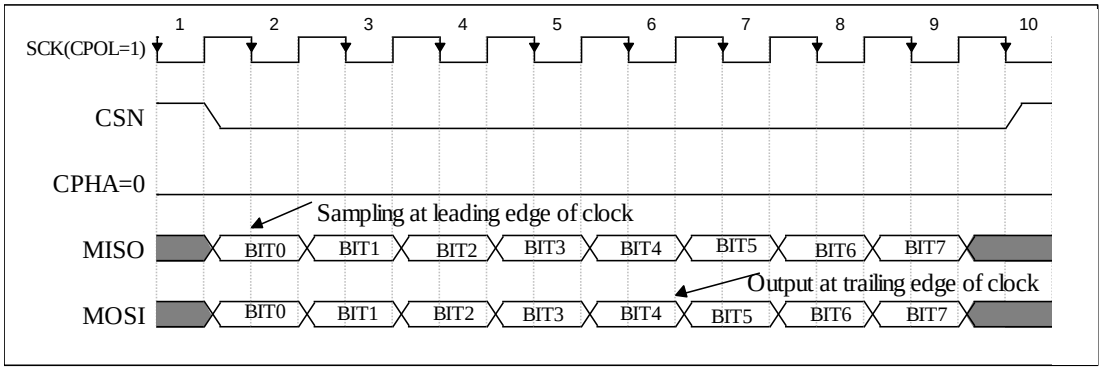


Figure13-3 spi mode 2 timing

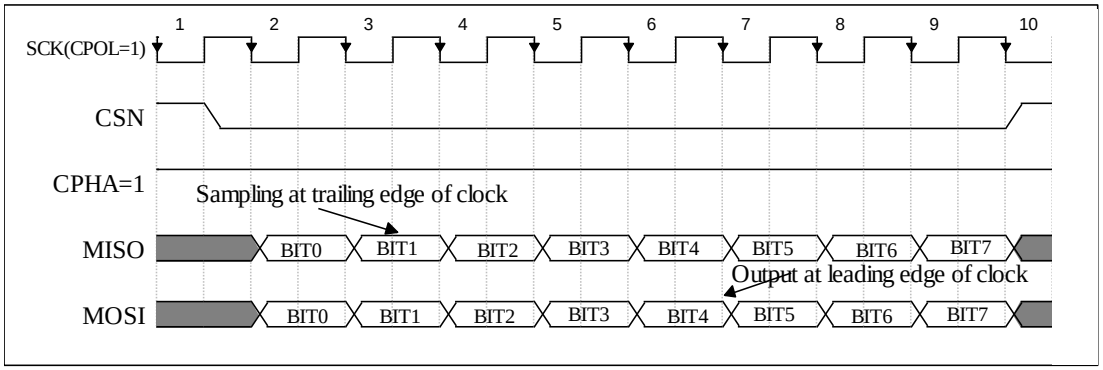


Figure13-4 spi mode3 timing

In the master configuration, the serial clock is generated on the SCK pin. MOSI is the data output pin and MISO is the data input pin.

Procedure:

1. Select the smctrl [3:0] bits to define the serial clock baud rate (see SPI_ CTRL register).
2. Select the CPOL and CPHA bits to define one of the four relationships between the data transfer and the serial clock.
3. The smctrl[5:4] bits select whether or not to turn on the SPI in SPI_CTRL

register.

Transmit sequence:

The transmit sequence begins when a byte is written in the Tx Buffer.

The data byte is parallel-loaded into the shift register (from the internal bus) during the first bit transmission and then shifted out serially to the MOSI pin MSB first.The spi_int flag is set on the transfer of data from the Tx Buffer to the shift register and an interrupt is generated if the spi_int_en bit in the SPI_CTRL register is set.

Receive sequence:

For the receiver, when data transfer is complete:

- The data in the shift register is transferred to the RX Buffer and the spi_int flag is set.
- An interrupt is generated if the spi_int_en bit is set in the SPI_CTRL register.

When the SPI_DATA register is read, the SPI peripheral returns this buffered value.

13.3 Register description

Base address: 0x3000_0060

13.3.1 Control register（SPI1_CTRL）

Offset address:0x00

Reset value:0x0000_0000

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							spi_int_en
							RW
7	6	5	4	3	2	1	0
CPOL	CPHA	smctrl5	smctrl4	smctrl3	smctrl2	smctrl1	smctrl0
RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:9	Reserved	Reserved bit
8	spi_int_en	spi1interrupt enable 0: disable，1: enable
7	CPOL	Clock polarity 0: SCK to 0 when idle 1: SCK to 1 when idle
6	CPHA	Clock phase 0: The first clock transition is the first data capture edge 1: The second clock transition is the first data capture edge
5:4	smctrl[5:4]	01：enable SPI； 00：disable SPI
3:0	smctrl[3:0]	Number of divisions from clock to SPI clock 0000：cclk/2； 0001：cclk/2； 0010：cclk/4； 0011：cclk/8； 0100：cclk/16； 0101：cclk/32； 0110：cclk/64； others：cclk/64

13.3.2 Date register（SPI1_DATA）

Offset address:0x04

Reset value:0x0000_0000

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

Bit	Marking	Functional description
31:8	Reserved	Reserved bit
7:0	spi_data	Data received or to be transmitted

13.3.3 Status register（SPI1_STATUS）

Offset address:0x08

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:5	Reserved	Reserved bit
4	spi_int	The flag bit of spi1interrupt，1:spi operation is done. Write 0 to clear the interrupt
3:1	Reserved	Reserved bit
0	spi_busy	Spi1 busy flag 1: SPI is busy in communication 0: SPI not busy

13.4 Register mapping

SPI1 register list

Base address:0x3000_0060

Register	Offset address	Description
SPI1_CTRL	0x00	SPI1 Control register
SPI1_DATA	0x04	SPI1 Data register
SPI1_STATUS	0x08	SPI1 Status register

14 Serial peripheral interface（SPI2）

14.1 Introduction

In addition to the function of SPI1, the SPI2 can also be a wireless ISP interface, and wireless ISP need to be connected to external Si24R1. In communication with Si24R1, CE and CSN are simulated by the PB0,PB1 separately.

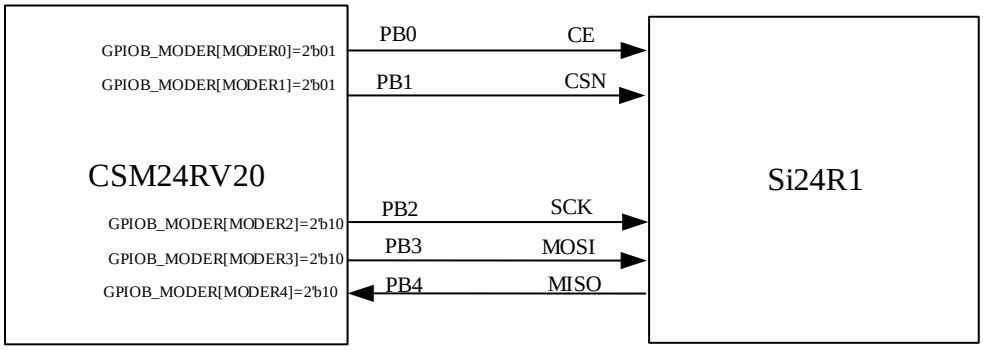


Figure14-1 SPI communication interface

14.1.1Main features

- Full-duplex synchronous transfers on three lines
- 8-bit transfer frame format selection
- Multimaster mode capability
- 8 master mode baud rate prescalers
- Programmable clock polarity and phase
- Dedicated flags with interrupt capability
- SPI bus busy status flag
- Be used as a wireless ISP interface

14.2 Functional description

Usually, the SPI is connected to external devices through four pins:

- MISO: Master In / Slave Out data. This pin can be used to transmit data in slave mode and receive data in master mode.
- MOSI: Master Out / Slave In data. This pin can be used to transmit data in master mode and receive data in slave mode.

- SCK: Serial Clock output for SPI masters and input for SPI slaves.
- CSN: Controlled by R1_CSN.

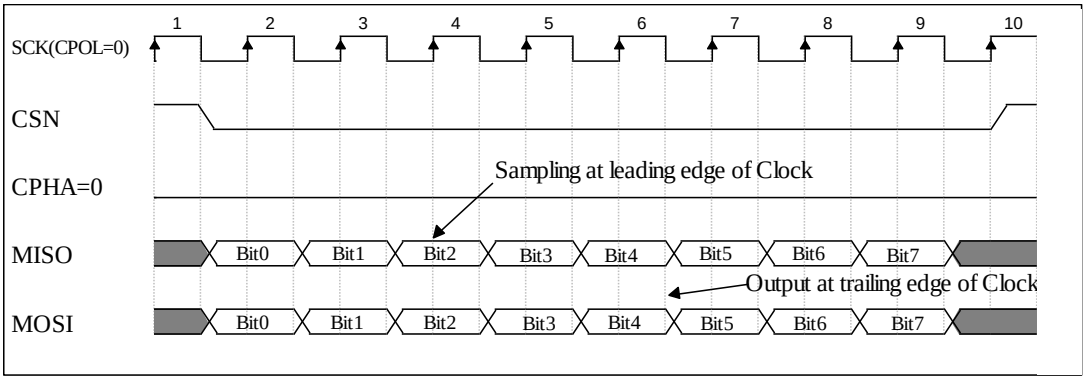


Figure14-2 spi mode0 timing

In the master configuration, the serial clock is generated on the SCK pin. MOSI is the data output pin and MISO is the data input pin.

14.3 Register description

Base address: 0x3000_0070

14.3.1 Control register（SPI2_CTRL）

Offset address:0x00

Reset value:0x0000_0000

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							spi_int_en
							RW
7	6	5	4	3	2	1	0
CPOL	CPHA	smctrl5	smctrl4	smctrl3	smctrl2	smctrl1	smctrl0
RW	RW	RW	RW	RW	RW	RW	RW

Bit	Marking	Functional description
31:9	Reserved	Reserved bit
8	spi_int_en	spi1interrupt enable 0: disable，1: enable
7	CPOL	Clock polarity 0: SCK to 0 when idle 1: SCK to 1 when idle
6	CPHA	Clock phase 0: The first clock transition is the first data capture edge 1: The second clock transition is the first data capture edge
5:4	smctrl[5:4]	01：enable SPI； 00：disable SPI
3:0	smctrl[3:0]	Number of divisions from clock to SPI clock 0000：cclk/2； 0001：cclk/2； 0010：cclk/4； 0011：cclk/8； 0100：cclk/16； 0101：cclk/32； 0110：cclk/64； others：cclk/64

14.3.2 Data register（SPI2_DATA）

Offset address:0x04

Reset value:0x0000_0000

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

Bit	Marking	Functional description
31:8	Reserved	Reserved bit
7:0	spi_data	Data received or to be transmitted

14.3.3 Status register（SPI2_STATUS）

Offset address:0x08

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:5	Reserved	Reserved bit
4	spi_int	The flag bit of spi1interrupt，1:spi operation is done. Write 0 to clear the interrupt
3:1	Reserved	Reserved bit
0	spi_busy	Spi1 busy flag 1: SPI is busy in communication 0: SPI not busy

14.4 Register mapping

SPI2 register list

Base address:0x3000_0070

Register	Offset address	Description
SPI2_CTRL	0x00	SPI2 Control register
SPI2_DATA	0x04	SPI2 Data register
SPI2_STATUS	0x08	SPI2 Status register

15 Asynchronous receiver/transmitter（UART）

15.1 Introduction

The Universal Asynchronous Transceiver (UART) provides a flexible method of exchanging full-duplex data with external devices that use the industry-standard NRZ asynchronous serial data format. The UART utilizes a fractional baud rate generator to provide a wide range of baud rate options.

The chip has four built-in UARTs, UART1 supports ISP download program.

The pin default are PA5 and PA6 in Bootload.

15.1.1 Main features

- Full-duplex, asynchronous communication
- NRZ Standard Format
- Fractional Baud Rate Generator System
- Supports baud rate adaptation
- Programmable data word length (8 or 9 bits)
- Separate transmitter and receiver enable bits
- Detecting flag
- End-of-transmission flag
- Multiprocessor communication

15.2 Functional description

The serial port is controlled by S0CON, while the actual data transferred is read or written in the S0BUF register. Transmission speed (baud rate) is selected using the uartdiv. The formula for calculating the baud rate is described in the uartdiv register.

1) Synchronous mode, fixed baud rate
2) 8-bit UART mode, variable baud rate
3) 9-bit UART mode, variable baud rate

Table15-1 Common rate uartdiv values (f_{ck} = 16Mhz, f_{ck} is the bus clock）

ID	Baud rates（Kpbs）	uartdiv
1	2.4	0x1A0B
2	9.6	0x0683
3	19.2	0x0341
4	57.6	0x0116
5	115.2	0x008B
6	230.4	0x0045
7	460.8	0x0023
8	921.6	0x0011
9	1228.8	0x000D

The serial port supports baud rate adaptation, which is realized by measuring the baud rate of the received signal on the RX pin and configuring it into the baud rate register. The usage method is as follows:

1) Configure the MCU and peripherals to use the same clock source (Set the clock source selection register (CMU_CLK_SEL) to select the clock source for the MCU and peripherals.)
2) Configure baudtrim = 1 and trim_en to write 0;
3) Configure baudtrim = 1 and trim_en to write 1;
4) RX receives the UART frame, the low level in the frame can only be 1 bit wide;
5) Wait until trim_en goes to 0 and read the result of trim_clk_result;
6) Use trim_clk_result as the baud rate setting for the uart.

15.3 UART pin mapping

	Pin	Description	Alternate configuration
UART1	PA6	UART1_TX	AF0（default）
	PA5	UART1_RX	AF0（default）
	PA9	UART1_TX	AF1
	PA8	UART1_RX	AF1
UART2	PA4	UART2_TX	AF3
	PA3	UART2_RX	AF3
UART3	PA11	UART3_TX	AF3
	PA10	UART3_RX	AF3
UART4	PA15	UART4_TX	AF3
	PA14	UART4_RX	AF3

15.4 Register description

UART1 base address: 0x3000_0010

UART2 base address: 0x3000_0700

UART3 base address: 0x3000_0800

UART4 base address: 0x3000_0900

Auto BPS base address: 0x3000_0380

15.4.1 Control register（UART_CTRL）

Offset address:0x00

Reset value:0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						tx_int_en	rx_int_en	tx_busy	Res	uartdiv					
						RW	RW	RW		RW					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
uartdiv								s0_con							
RW								RW							

Bit	Marking	Functional description
31:26	Reserved	Reserved bit
25	tx_int_en	Transmit interrupt enable bit，0：disable, 1：enable
24	rx_int_en	Receive interrupt enable bit，0：disable, 1：enable
23	tx_busy	Transmit busy，1：transmitting
22	Res	Reserved bit
21:8	uartdiv	Baud rate setting bit，21：12 for integers，11：8 for decimals Tx/ Rx baud = f_{CK}/(16 * uartdiv) ,uartdiv = uartdiv[13:4] + uart[3:0]/16； where f_{ck} is the peripheral clock.
7:0	s0_con	7:6 uart mode select 10: Mode 0 – Shift register at baud rate per_clk/12(per_clk is peripheral clock) 01: Mode 1 – 8-bit data, baud rate controlled by uartdiv; 10/11: Mode 2 – 9-bit data, baud rate controlled by uartdiv;
		bit 5: Multiprocessor communication enable
		bit 4: Serial reception enable: 1: enable 0:disable
		bit 3: transmitter bit 8. This bit is used while transmitting data through uart in Modes 2 and 3. The state of this bit corresponds with the state of the 9th transmitted bit. It is controlled by software.
		bit2: Received bit 8. This bit is used while receiving data in Modes 2 and 3. It reflects the state of the 9th received bit
		bit 1: Transmit interrupt flag. It indicates completion of a serial transmission. It is set by hardware at the end of bit 8 in mode 0 or at the beginning of a stop bit in other modes. It must be cleared by software.
		bit0: Receive interrupt flag. It is set by hardware after completion of a serial reception. It is set by hardware at the end of bit 8 in mode 0 or in the middle of a stop bit in other modes. It must be cleared by software

15.4.2 Data register（UART_DATA）

Offset address:0x04

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:8	Reserved	Reserved bit
7:0	uart_datain	Write sent data/read reveived data

15.4.3 Auto BPS configuration register（AUTOBPS_CONFIG）

Address：0x3000_0380

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:9	Reserved	Reserved bit
8	trim_en	Activate baud rate adaptation, 0: off, 1: activate This bit is automatically cleared by the hardware at the end of a trim.
7	Reserved	Reserved bit
6	baudtrim	1: Enable the baud rate adaptation function, can calculate the baud rate of the external input of the uart
5:3	Reserved	Reserved bit
2:1	uart_rx_sel	00: select the RX of UART1 01: select the RX of UART2 10: select the RX of UART3 11: select the RX of UART4 Note: The setting of uart_rx_sel is active only when baudtrim is set to 0.
0	Reserved	Reserved bit

15.4.4 Auto BPS result register（AUTOBPS_RESULT）

Address: 0x3000_0384

Reset value: 0x0000_0000

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

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

Bit	Marking	Functional description
31:14	Reserved	Reserved bit
13:0	trim_clk_result	The result of trim when trim finish

15.5 Register mapping

Uart1 Base address:0x3000_0010

Uart2 Base address:0x3000_0700

Uart3 Base address:0x3000_0800

Uart4B ase address:0x3000_0900

Register	Offset address	Description
UARTx_CTRL	0x00	UART control register
UARTx_DATA	0x04	UART data register

Auto BPS base address:0x3000_0380

Register	Offset address	Description
AUTOBPS_CONFIG	0x00	Auto BPS configuration register
AUTOBPS_RESULT	0x04	Auto BPS result register

16 Low voltage detection（LVD）

16.1 Introduction

Low voltage detection module detects power voltage. Low voltage detection interrupt is pulled up when voltage is low. If LV interrupt is enabled, interrupt can pass to CLIC and generate undervoltage interrupt.

16.2 Register description

16.2.1 Low voltage detection interrupt enable register（LVD_IRQ_EN）

Address：0x3000_0330

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:1	Reserved	Reserved bit
0	lv_irq_en	LVD interrupt enable 1:enable 0:disable

16.2.2 Low voltage detection interrupt register（LV_IRQ）

Address：0x3000_0334

Reset value:0x0000_0000

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

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

	R

Bit	Marking	Functional description
31:1	Reserved	Reserved bit
0	lv_irq	LVD interrupt flag bit 1: low voltage interrupt has generated 0: low voltage interrupt has not generated When the power voltage is higher than the threshold,this bit is cleared by hardware automatically

16.2.3 Low voltage threshold register（LV_TH）

Address：0x3000_0400

Reset value:0xFFFF_FF5E

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

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

Bit	Marking	Functional description
31:29	Reserved	Reserved bit
28:25	lv_th	low voltage threshold control bits, please refer to Chapter 0
24:0	Reserved	Reserved bit

Note：

(1) It is forbidden to modify lv_th <31:29> <24:0>, that is,the value of the register must be read first, only change the lv_th bit on the basis of the read value, and then write back

(2) When the power supply voltage fluctuates repeatedly at the comparison threshold point, in addition to low voltage interruption, exception interrupt may occur. If you need to avoid this exception interrupt, set the threshold one step higher when entering the low voltage interrupt for the first time.

16.3 Register mapping

LV register list

Base address:0x3000_0330

Register	Offset address	Description
LV_IRQ_EN	0x00	Low voltage detection interrupt enable register
LV_IRQ	0x04	Low voltage detection interrupt register
LV_TH	0x3000_0400	Low voltage threshold register

17 Random number generation（RANDGEN）

17.1 Introduction

RANDGEN is a module that generates true random numbers, and generates the corresponding flag bits after generating the random numbers. Indicates that the software goes to the corresponding space to read the corresponding data.

17.2 Register description

17.2.1 RANDGEN control register（RDGCR）

Offset address:0x38

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:2	Reserved	Reserved bit
1	data_rdy	State flag bit, 1: The random number has been generated and can be read at any time. data_rdy can only be cleared by reset.
0	rand_en	Random number generation enable bit. 1:enable 0:disable

17.2.2 RANDGEN data register（RDGDR）

Offset address:0x40

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:0	rand_data	Generated random numbers After the first time rand_en is written to 1, data_rdy is set to 1 by hardware after 32 3K clock cycles. The random number will be updated once in every 3K clock cycle as long as rand_en is 1.

17.3 Register mapping

RANDGEN register list

Base address:0x3000_0238

Register	Offset address	Description
RDGCR	0x38	RANDGEN control register
RDGDR	0x40	RANDGEN data register

18 Comparator（COMP）

18.1 Introduction

Three independent comparators are built in the CSM32RV20, and when using a comparator, the GPIO should be configured to analog mode.

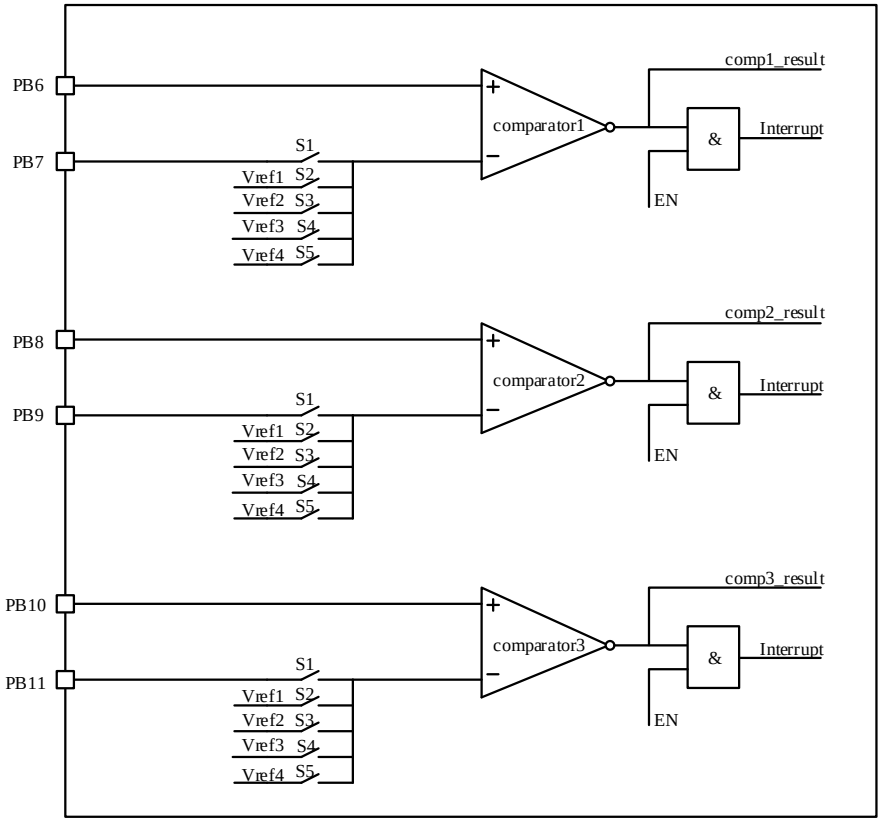


Figure18-1 Comparator block diagram

18.2 Register description

COMP1 base address: 0x3000_0B00

COMP2 base address: 0x3000_0C00

COMP3 base address: 0x3000_0D00

18.2.1 Comparator control register（COMP_CTRL）

Offset address:0x00

Reset value:0x0000_0070

31	30	29	28	27	26	25	24	23	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									ref_io_sel			ratio_sel	int_en	mode1	mode0
									RW			RW	RW	RW	RW

Bit	Marking	Functional description
31:7	Reserved	Reserved bit
6:4	ref_io_sel	Source selection and enable of negative <3><2><1>: 0 0 0 : pad access; 0 0 1 : 0.3 V access; 0 1 0 : 0.6 V access; 0 1 1 : 0.9 V access; 1 0 0 : 1.2 V access; 1 0 1: comparator off; 1 1 0: comparator off; 1 1 1: comparator off（default）;
3	ratio_sel	Rate selection 1: the maximum operate rate the comparator is 2MHz 0: the maximum operate rate the comparator is 1MHz High rate will have a higher power consumption.
2	int_en	Comparator Interrupt enable 1: enable 0:disable
1:0	mode[1:0]	The edge selection of interrupt 00：high level detection 01：falling edge detection 10：rising edge detection 11：low level detection

18.2.2 Comparator interrupt register（COMP_IRQ）

Offset address:0x04

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:1	Reserved	Reserved bit
0	irq	Comparator interrupt 1: comparator interrupt has generated, write 1 clear the interrupt

18.2.3 Comparator result registger（COMP_RESULT）

Offset address:0x08

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:1	Reserved	Reserved bit
0	result	The result of comparator 1:the result is higher than the selected voltage value of ref_io_sel

18.3 Register mapping

COMP1 register list

Base address:0x3000_0B00

Register	Offset address	Description
COMP1_CTRL	0x00	Comparator Control Register
COMP1_IRQ	0x04	Comparator Interrupt Register
COMP1_RESULT	0x08	Comparator Result Register

COMP2 register list

Base address:0x3000_0C00

Register	Offset address	Description
COMP1_CTRL	0x00	Comparator Control Register
COMP1_IRQ	0x04	Comparator Interrupt Register
COMP1_RESULT	0x08	Comparator Result Register

COMP3 register list

Base address:0x3000_0D00

Register	Offset address	Description
COMP1_CTRL	0x00	Comparator Control Register
COMP1_IRQ	0x04	Comparator Interrupt Register
COMP1_RESULT	0x08	Comparator Result Register

19 UART auto BPS（TRIM）

19.1 Introduction

The UART baud rate adaptive module calculates the baud rate of the UART by calculating the length of the RX low level. When measuring the baud rate, the RX low level width must be 1 bit.

19.2 Register description

19.2.1 Configuration register（TRIM_CLK_CFG）

Address：0x3000_0380

Reset value:0x0000_0000

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_en
							RW
7	6	5	4	3	2	1	0
Reserved	baudtrim	Reserved			uart_rx_sel		trim_clk_sel
	RW				RW		RW

Bit	Marking	Functional description
31:9	Reserved	Reserved bit
8	trim_en	Trim enable，0：off； 1：on. Trim is automatically cleared by hardware when finished
7	Reserved	Reserved bit
6	baudtrim	1：select baud rate adaptation function
5:3	Reserved	Reserved bit
2:1	uart_rx_sel	00：select the RX of UART1

		01：select the RX of UART2 10：select the RX of UART3 11：select the RX of UART4 *Note: The setting of uart_rx_sel is active when baudtrim is set to zero。*
0	trim_clk_sel	1: selecte 3k as the measured clock 0: selecte RC as the measured clock

19.2.2 Result register（TRIM_CLK_RESULT）

Address：0x3000_0384

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:14	Reserved	Reserved bit
13:0	trim_clk_result	The result of trim Measure baud rate：trim_clk_result is the baud rate

19.2.3 Flag register（TRIM_CLK_FLAG）

Address：0x3000_0388

Reset value:0x0000_0000

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

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

Bit	Marking	Functional description
31:30	Reserved	Reserved bit
0	trim_clk_flag	Trim finish flag 1：trim is finished，0：trim is in progress or no trim is in progress

19.3 Usage method

19.3.1 Baud rate adaptation

1. Configure the MCU and peripherals to use the same clock source;(Set the clock source selection register (CMU_CLK_SEL) to select the same clock source for the MCU and peripheral);
2. Configure baudtrim = 1 and trim_en to write 0;
3. Configure the test UART (uart_rx_sel);
4. Configure baudtrim = 1 and trim_en to write 1;
5. Master sends 0x7F;
6. RX receives the UART frame, the low level in the frame can only be 1 bit wide;
7. Wait until trim_en goes to 0 and read the result of trim_clk_result;
8. Use trim_clk_result as the baud rate setting for the UART;

20 FLASH/NVM burning

The CSM32RV20 is configured with 40kBytes FLASH + 512bytes NVM, starting at address 0x2000_0000, and this section describes its features, functions and operation.

20.1 FLASH/NVM Main features

- The FLASH size is 10k x 32 bits (40k bytes);
- The NVM size is 128 x 32 bits (512 bytes);
- FLASH/NVM is organized by sectors with 512 bytes per sector;
- Supports reading and writing in the application and can be used to save user data
- Supports byte, half-word (16-bit), or full-word (32-bit) reads and writes by 8 bits
- Supports sector-by-sector erase;
- Supports whole Chip Erase (erase FLASH and NVM)
- FLASH supports read/write protection

20.2 FLASH/NVM mappig

Table20-1 shows the address assignments for the FLASH/NVM memory.

Table20-1 FLASH/NVM memory address mapping

Block	Name		Memory address	Size
FLASH	Sector 0	row 0	0x2000_0000 – 0x2000_007F	128 bytes
		row 1	0x2000_0080 – 0x2000_00FF	128 bytes
		row 2	0x2000_0100 – 0x2000_017F	128 bytes
		row 3	0x2000_0180 – 0x2000_01FF	128 bytes
	Sector 1	row 0	0x2000_0200 – 0x2000_027F	128 bytes
		row 1	0x2000_0280 – 0x2000_02FF	128 bytes
		row 2	0x2000_0300 – 0x2000_037F	128 bytes
		row 3	0x2000_0380 – 0x2000_03FF	128 bytes
	…			

	Sector 78	row 0	0x2000_9C00 – 0x2000_9C7F	128 bytes
		row 1	0x2000_9C80 – 0x2000_9CFF	128 bytes
		row 2	0x2000_9D00 – 0x2000_9D7F	128 bytes
		row 3	0x2000_9D80 – 0x2000_9DFF	128 bytes
	Sector 79	row 0	0x2000_9E00 – 0x2000_9E7F	128 bytes
		row 1	0x2000_9E80 – 0x2000_9EFF	128 bytes
		row 2	0x2000_9F00 – 0x2000_9F7F	128 bytes
		row 3	0x2000_9F80 – 0x2000_9FF7	120 bytes
			0x2000_9FF8 – 0x2000_9FFF	Users are forbidden to use [1]
NVM			[2]	512 bytes

The FLASH space is used to store the user program, which can be programmed through the cJTAG or UART interface. The user program can read and write the 512-byte NVM online through the function flash_operation( ).

[1] For the user to erase the 79th sector of flash or the entire flash, you must write the data in the address 0x3000_0428 to the flash space of 0x2000_9ffc, and write the data in the address 0x3000_042c to the flash space of 0x2000_9ff8.

The NVM can only be accessed by the operating function flash_operation( ), and cannot be accessed by absolute address, more details refer to section 20.3.

20.3 FLASH/NVM operation

The function flash_operation() can operate on FLASH/NVM, and its entry address is 0x2100_0fe2. The user program can call this function by declaring the function pointer to realize the operation on NVM and FLASH.

It is recommended to turn off interrupts (MIE=0) before calling the flash_operation() function. If the interrupt is not turned off(MIE=1), when an interrupt is generated during the execution of the flash_operation()function, the MCU will enter a trap, causing the flash_operation() operation to fail.

The definition of the function in question is as follows:

```
uint8_t flash_operation(uint8_t code, uint16_t addr, uint8_t *p, uint16_t num,uint8_t clk)
```

Parameter and return value descriptions:

code: Function code, used to select read, write and erase function

addr: Used to set the start address for reading and programming, the range is 0-

0x9ffc, also used to specify the sector to be erased, the address is the first address of the sector;

*p: array pointer;

num ： 1) Number must be 1 for Erase_sector operation; 2) It is the number of bytes to be operated for Write_bytes/Read_bytes/Write_NVM/Read_NVM operation, the range is 1-128.

clk：flash clock, generally default 0

Return Value: 0, success

1) Illegal address;
2) The number of operations to be performed is greater than the number of bytes remaining;
3) The array pointer is NULL;
4) Operator code error.

Note: Check for array out-of-bounds before performing read/write operations. The function code is described in Table 20-2.

Table20-2 Function code

Operation	Code	Description
Erase_sector	1	Erase 1 specified sector, num can only be 1 in this operation.
Erase_NVM	2	Erase NVM
Erase_chip	3	Erase the entire FLASH and NVM (Users are forbidden to be used in the user program, otherwise they will erase the entire flash and cause an accident)
Write_bytes	4	Write num bytes to FLASH (address should be 4 bytes aligned, one operation can only be in one row. Inside, a sector is divided into 4 rows, each with 128 bytes)
Read_bytes	5	Read num bytes from FLASH (address should be 4-byte aligned)
Write_NVM	6	Write num bytes to the NVM (addresses should be 4-byte aligned, one operation can only be in one row.Within the NVM, the NVM is divided into 4 rows, each with 128 bytes).
Read_NVM	7	Read num bytes from NVM (address should be 4-byte aligned)

Define function pointer example:

```
#define MY_FLASH_OPEREATION_Addr 0x21000fe2
uint8_t (*my_flash_operation)(uint8_t code, uint16_t addr, uint8_t *p, uint16_t num,uint8_t clk);
my_flash_operation = MY_FLASH_OPEREATION_Addr;
```

20.4 FLASH r/w protection

The FLASH memory protects the user program from external read/write access. Setting is done through the upper computer software or IDE software.

20.5 FLASH/NVM burning

The user program can be downloaded to the CSM32RV20 in two ways:

1) Programming the upper computer software: CSM32RV20 stores the bootload program in ROM and communicates with the programming upper computer through the serial port. Users can program through the upper computer by correctly connecting the TX1/RX1 pins of the serial port

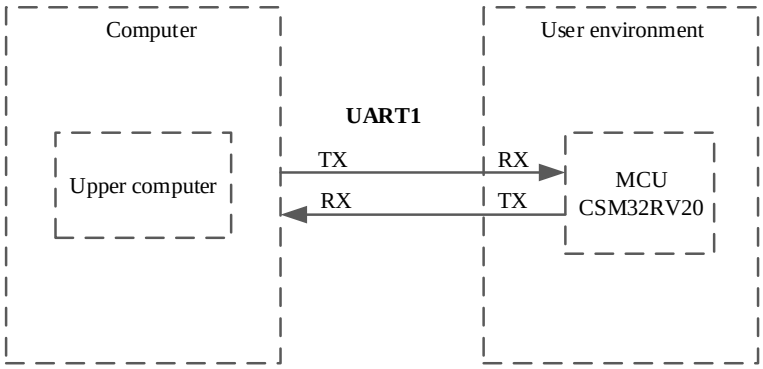


Figure20-1 MCU connection to upper computer

2) Via cJTAG interface: The CSM32RV20 is equipped with a 2-wire cJTAG interface for debugging and programming. Users can connect the debugger to the MCU's debugging interfaces: TCKC and TMSC pins, and then program via the IDE software. For the description of the cJTAG interface, please refer to the 21Debug support for more details, and the detailed operation procedures of debugging and programming are shown in the CSM Studio IDE Manual.

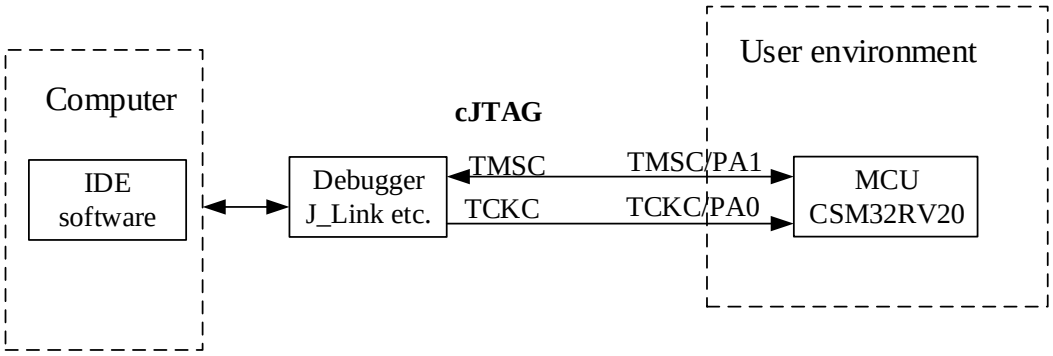


Figure20-2 MCU connection with cJTAG

21 Debug support

21.1 Introduction

The CSM32RV20 is built around a 32-bit RISC-V core that includes a JTAG debug transport module (DTM) to support testing and debugging the system using a single external industry standard 1149.1 JTAG interface. The DTM supports interactive debugging and up to 4 hardware breakpoints, and the JTAG debug interface is driven by an industry standard 2-wire cJTAG interface.

21.2 cJTAG debug interface

The CSM32RV20 Core has an embedded cJTAG adapter, which provides an interface consisting of two signals: TCKC and TMSC. Many debuggers can support both traditional JTAG and the new cJTAG. cJTAG can be used by connecting the TCKC and TMSC pins of the debugging probes to the corresponding pins of the MCU, and it supports on-line debugging and downloading. For details, please refer to the CSM Studio IDE Manual.

Table21-1 cJTAG debug interface pins

cJTAG pin name	cJTAG debug port		Pin assignment
	Type	Debugging assignment	
TMSC	IO	Data input/output	PA1
TCKC	I	clock	PA0

The two cJTAG pins are muxed to the debug interface by default, but can be muxed to the general I/O interface by the user if cJTAG is not used. More details please refer to Chapter 5 General and alternate function I/Os

22 RISC-V Core

Built-in 32-bit RISC-V core with two-stage pipeline and support for IMAC instructions, please refer to [1][2] for more information about RISC-V.

[1] A. Waterman and K. Asanovic, Eds., The RISC-V Instruction Set Manual, Volume I: User-Level ISA, Version 2.2, May 2017. [Online]. Available: https://riscv.org/specifications/

[2] The RISC-V Instruction Set Manual Volume II： Privileged Architecture Version 1.10, May 2017. [Online]. Available: https://riscv.org/specifications/

23 Chip electronic signature

The electronic signature contains VersionSize and MCUID, which can be read through the cjtag, programmatic upper computer or MCU(user program), which contains the identified data written by the factory.

23.1 Chip version ID（Version ID）

VersionID register contains the chip version information and the Flash capacity information.

Address：0x3000_0430

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

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

Bit	Marking	Functional description
31:8	version	The chip Version number
7:0	size	the size of Flash capacity 0000_0001：4k； 0000_0010：8k； 0000_0011：12k； ……. 0000_1010：40k

23.2 MCUID

MCUID is factory unique ID.

Address：0x3000_0420

Reset value:ID value

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

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

Bit	Marking	Functional description
31:0	mcuid	The ID of this mcu chip

24 Electrical parameters

24.1 Parameter conditions

All voltages are referenced to VSS unless otherwise noted.

24.1.1 Max/min value

All maximum and minimum values are guaranteed at the worst environmental temperature, supply voltage and clock frequency unless otherwise stated.

Data based on characterization results, design simulations, and/or technical characteristics are indicated in the table footnotes are not tested in production. Minimum and maximum sample testing.

24.1.2 Typical value

Typical data is based on TA = 25 ℃ and VDD = 3.3 V unless otherwise noted. It is only used as a design guide.

24.1.3 Power supply solution

CSM32RV20 has a power pin and ground(bottom pad).

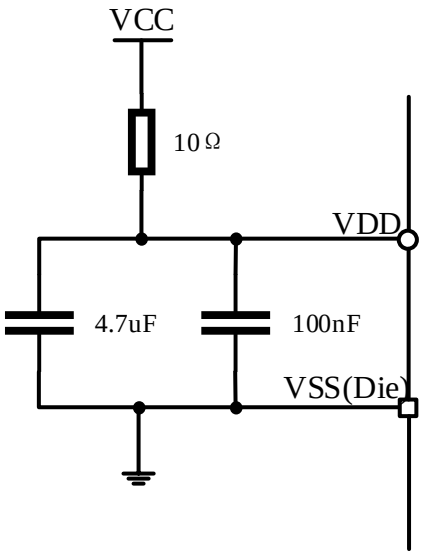


Figure24-1 Power supply scheme

24.1.4 Current consumption measurement

Current consumption measurements are shown in Figure24-2.

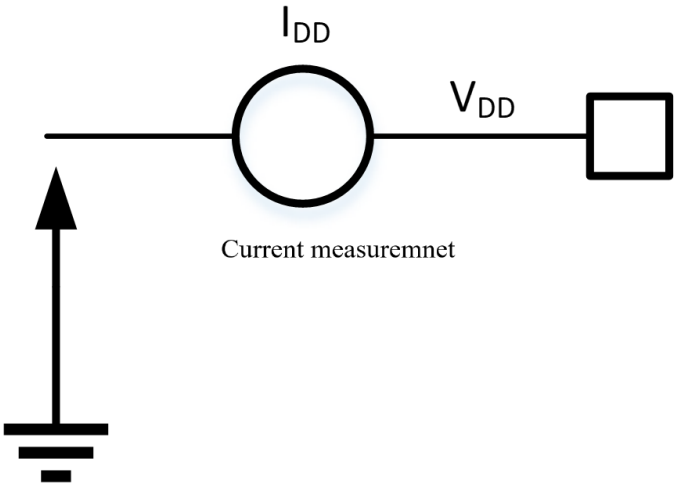


Figure24-2 Current consumption measurement

24.2 Absolute maximum ratings

Critical or exceeding the absolute maximum rating may cause the chip to operate abnormally or even be damaged.

Table24-1 Absolute maximum ratings

Symbol	Parameters	Minimum	Maximum	Unit
V_{DD}-V_{SS}(1)	External supply voltage	-0.3	5.8	V
V_{IN}(1)	Pin input voltage	-0.3	5.8	V
$V_{ESD(HBM)}$(1)	Electrostatic discharge voltage (human body model)		2000	V
T_S	Storage temperature	-55	150	℃
To	Operation temperature	-40	105	℃

Note：(1) Design parameters

24.3 Operating conditions

24.3.1 General operating conditions

Table24-2 General Operating Conditions

Symbol	Parameters	Conditions	Min	Typ	Max	Unit
f_{RC}	Internal system clock frequency	Ta=25℃，V_{DD}=3.3 V	10	16/32	34	MHz
f_{OSC}	External crystal Oscillator Clock frequency	Ta=25℃，V_{DD}=3.3 V	4	16/32	32	MHz
f_{3K}	Internal 3K clock frequency	Ta=25℃，V_{DD}=3.3 V	1.8	3	8	KHz
V_{DD}	Standard operating voltage	Using ADC	2.5	3.3	5	V
		Not using ADC	1.8	3.3	5.5	
IDD	Standard operating current	Peripheral are closed, 3K OSC and IWDG are working,16MHZ RCOSC	-	3.3	-	mA
	Standard operating current	Peripheral are closed, 3K OSC and IWDG are working,32MHZ RCOSC	-	5.7	-	mA
	Standard operating current	Peripheral are closed, 3K OSC and IWDG are working,16MHZ crystal oscillator	-	2.9	-	mA
	Standard operating current	Peripheral are closed, 3K OSC and IWDG are working,32MHZ crystal oscillator	-	4.2	-	mA
	Standby mode	Peripheral are closed, 3K OSC and IWDG are working,SRAM remains,RCOSC and OSC open,main LDO open	-	1.6	-	mA
	Sleep mode	Peripheral are closed, 3K OSC and IWDG are working,SRAM remains,RCOSC and OSC close,main LDO open	-	364	-	uA
	Power-down mode 1	Peripheral are closed, 3K OSC and IWDG are working,SRAM remains,RCOSC and OSC close,main LDO close	-	3.5	-	uA
	Power-down mode 2	Peripheral are closed, 3K OSC and IWDG are	-	1	-	uA

		working,SRAM is closed(data loss),RCOSC and OSC close,main LDO close				

Note: If there is no specification, paramerters are in Ta=25℃, VDD=3.3V.

24.3.2 Internal system clock source parameters

The frequency of RCH can be adjusted by the register, which is the low 9 bits of the absolute address 0x3000_101C.As the modifier increases, the frequency decreases by 0.4%.

Note: Except for the frequency modifier, all other bits are forbidden to be modified.

Table24-2 Internal system clock

Symbol	Parameters		Conditions	Min	Typ	Max	Unit
f_{CLK}	Frequency	-	-	10	16/32	34	MHz
TRIM	Fine-tuning step	16M	-	-	0.4	-	%
		32M	-	-	0.4	-	%
ACC	Oscillator accuracy	16M	Ta= -40℃ ~ 125℃	-	±4	-	%
			Ta= -20℃ ~ 85℃		±3		%
			Ta= 25℃		±1		%
		32M	Ta= -40℃ ~ 125℃	-	±8	-	%
			Ta= -20℃ ~ 85℃		±5		%
			Ta= 25℃		±1		%
t_{su}	Strat time	-	Ta=25℃，V_{DD}=3.3 V	-	1	-	us

24.3.3 External system clock source parameters

External clock source uses low cost crystal Oscillator: 4 ~ 32 MHz ±60 ppm.

When used, GPIOB12/GPIOB13 should be configured in analog mode. When the crystal pins are not configured for analog mode, the external crystal is turned off and the input and output pins are used as standard I/Os. High quality external ceramic capacitors in the range of 5 pF to 20 pF are recommended for crystal load capacitors. (C1 = C2 = 15 pF is typical; refer to the crystal's datasheet for actual use)

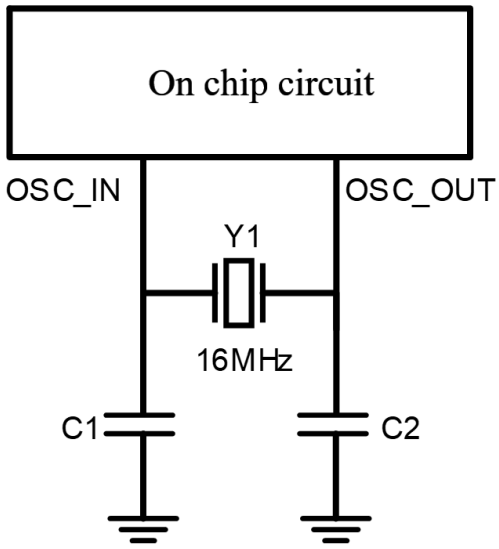


Figure24-3 Typical application when using 16 MHz crystal oscillator

Table24-3 Crystal oscillator parameter

Symbol	Parameters	V_{DD}	Conditions	Min	Typ	Max	Unit
f_{CLK}	Clock frequency		Ta=25℃，V_{DD}=3.3 V	4	16/32	32	MHz
I_{VDD}(1)	Current after crystal oscillator is stable	5.5 V	C1 = C2 = 15 pF	-	113	-	uA
		3.3 V			81		
		1.8 V			66		
t_{SU}(1)	Setup time	3.3 V	C1 = C2 = 15 pF	-	0.6	-	ms

Note：1. Design parameters

24.3.4 I/O port parameters

General input output parameters

Table24-4 I/O DC parameters

Symbol	Parameters	V_DD	Conditions	Min	Typ	Max
VIH	I/O high voltage input	5 V	-	0.7 ×VDD	-	V
		3.3 V		2.0		
		1.8 V		0.8 ×VDD		
VIL	I/O low voltage input	5 V	-	-	0.3 ×VDD	V
		3.3 V			0.8	
		1.8 V			0.2 ×VDD	
VHYS	Schmitt trigger hysteresis	5/3.3/1.8 V	-	0.1 ×VDD	-	V
I_{IH}	I/O high current input	5/3.3/1.8 V	-	-	+1	μA
I_{IL}	I/O low current input	5/3.3/1.8 V	-	-1	-	μA
VOH	I/O high voltage output	5 V	High drive Imin = 16mA Low drive Imin = 8mA	VDD-0.8		V
		3.3 V	High drive Imin = 8mA Low drive Imin = 4mA	2.4		
		1.8 V	High drive Imin = 4mA Low drive Imin = 2mA	VDD-0.45		
VOL	I/O low voltage output	5 V	High drive Imin = 16mA Low drive Imin = 8mA		0.5	V
		3.3 V	High drive Imin = 8mA Low drive Imin = 4mA		0.4	
		1.8 V	High drive Imin = 4mA Low drive Imin = 2mA		0.45	
Rpup	pull up resistor	5/3.3/1.8 V	-	20	100	KOhm
Rpdn	pull down resistor	5/3.3/1.8 V	-	20	100	KOhm
CIN	I/O capacitor input	5/3.3/1.8 V	-	-	10	pF

Note: The above are design parameters

24.3.5 ADC parameters

Symbol	Parameters	Conditions	Min	Typ	Max	Unit	Remark
VRES	Resolution	16bit		36.62		uV/LSB	
		15bit		73.24		uV/LSB	
		14bit		146.48		uV/LSB	
		13bit		292.97		uV/LSB	
TCONV	Conversion time (ADC clock 4MHz)	16bit		35.25		us	
		15bit		19.25		us	
		14bit		11.25		us	
		13bit		7.25		us	
VERR(1)	Measurement error			±3.5		mV	
INGAIN	Input channel gain		1/4	1	128		2~128 only support PA11 input
VINRANG	Input voltage range		0		VDD		VDD≤ 4.8V
fCLK	Clock frequency			4	8	MHz	

Note : 1. Input 1/4 gain，internal reference0~1.2V，VDD=3.3V，16bit resolution

24.3.6 Low voltage detection threshold

lv_th<3:0>	Voltage threshold（v）
0000	1.755
0001	1.865
0010	1.969
0011	2.077
0100	2.173
0101	2.268
0110	2.374
0111	2.49
1000	2.579
1001	2.671
1010	2.774
1011	2.879
1100	2.996
1101~1111	欠压电路关断

Note: Threshold voltage error ±50mv

25 Package information

CSM32RV20 uses 4x4 mm QFN32、TSSOP20 or 3x3mm QFN20 packages。

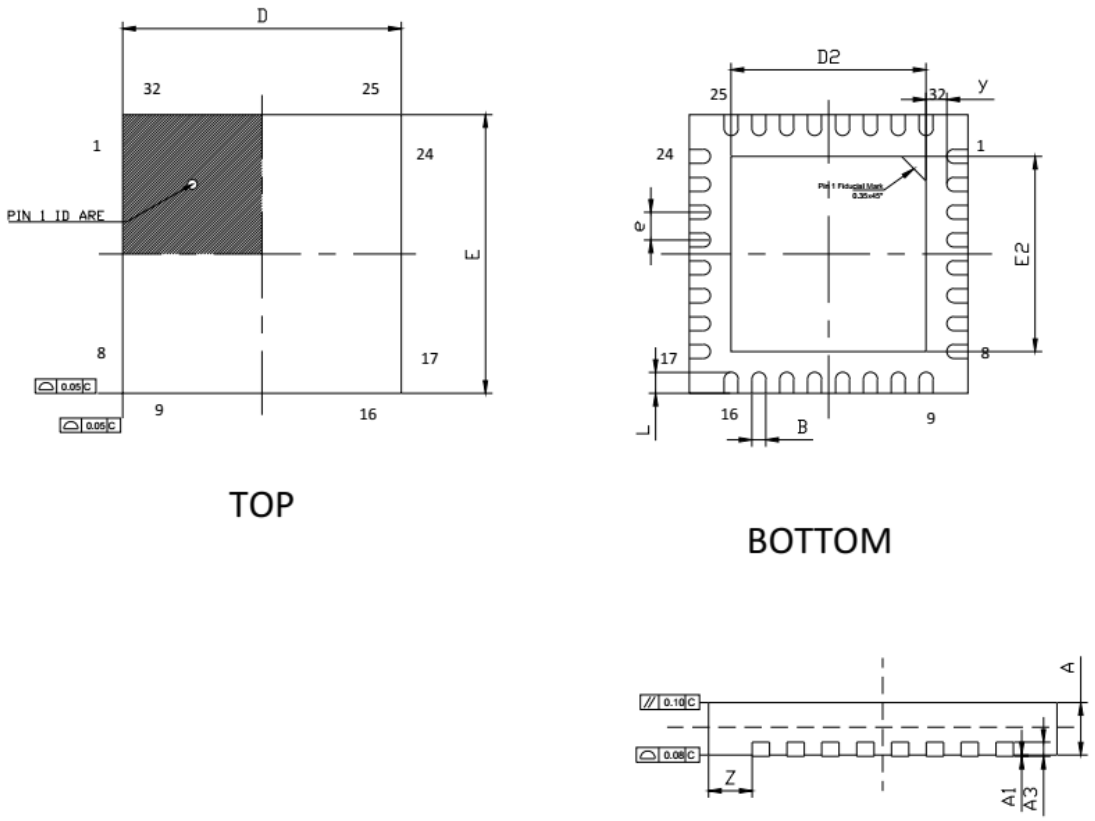


Figure25-1 QFN32 package

Size

Unit	D	E	D2	E2	A	A1	A3	B	e	K	L	y	Z
mm	4.10 (4.00) 3.90	4.10 (4.00) 3.90	2.90 (2.80) 2.70	2.90 (2.80) 2.70	0.80 (0.75) 0.70	0.05 (0.02) 0.00	0.203 REF	0.25 (0.20) 0.15	0.40 BSC	-	0.35 (0.30) 0.25	0.30 REF	0.50 REF

Note: All dimensions are in millimeters

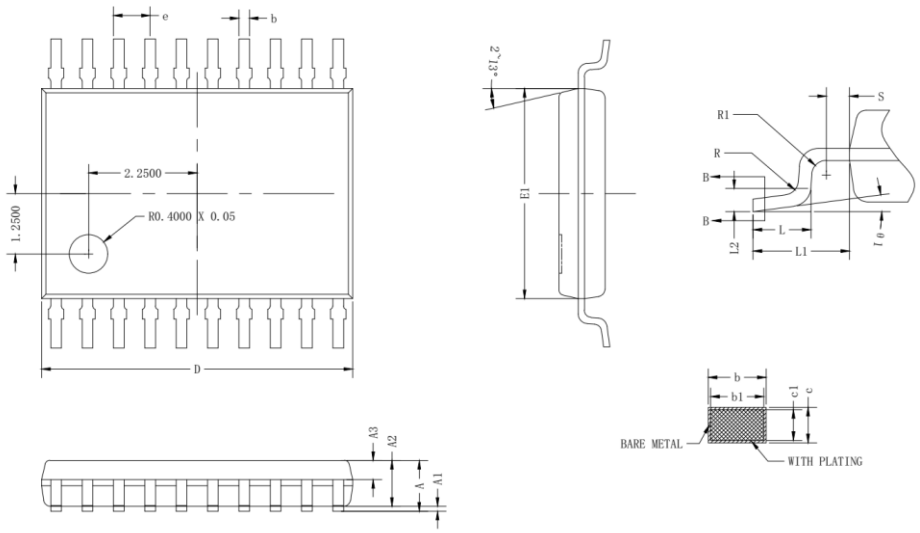
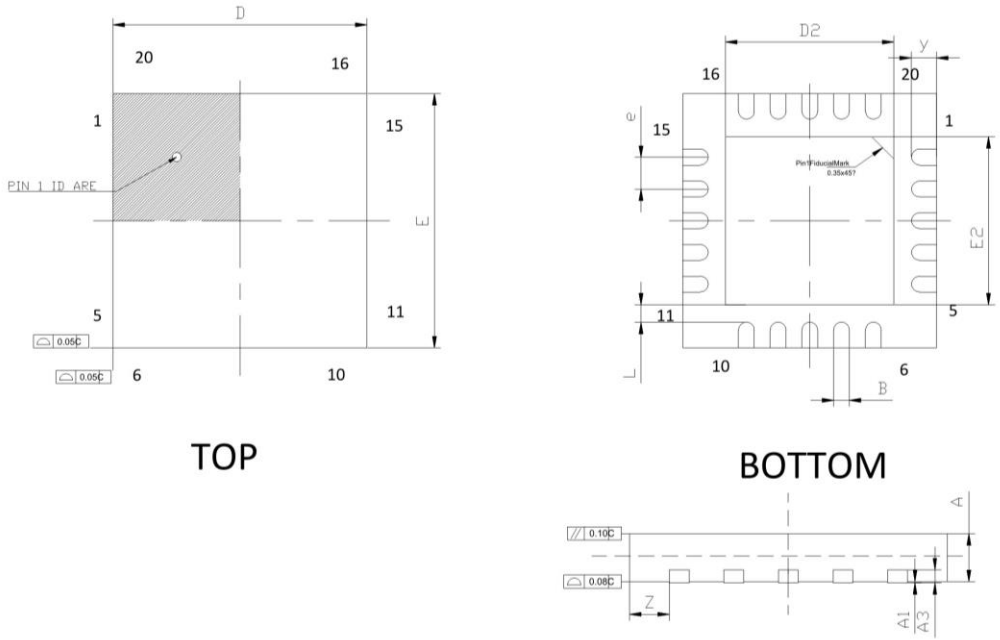


Figure25-2 TSSOP20 package

Size

Symbol	mm		
	Minimum	Typical	Maximum
A	1.00	-	1.10
A1	0.05	-	0.15
A3	0.39	-	0.40
b	0.20	-	0.2800
b1	0.2	0.22	0.24
c	0.1	-	0.19
c1	0.10	-	0.15
D	6.40	6.45	6.50
E	6.25	6.40	6.55
E1	-	4.35	4.40
L	0.50	0.60	0.7000
e	0.55	0.6500	0.75
L2	0.25NSC		
R	0.09	-	-
L1	1.0REF		
θ1	0°	-	8°
S	0.20	-	-

Note: All dimensions are in millimeters



Dimensions

Unit	D	E	D2	E2	A	A1	A3	B	e	K	L	y	Z
mm	3.025 (3.00) 2.975	3.025 (3.00) 2.975	1.65 (1.6) 1.55	1.65 (1.6) 1.55	0.80 (0.75) 0.70	0.05 (0.02) 0.00	0.203 REF	0.30 (0.25) 0.20	0.40 BSC	-	0.33 (0.28) 0.23	0.40 REF	0.655 REF

Notes

1.AllDimensionsareinMillimeters.
2.DimensionsDoNotincludeBurrs,MoldFlash,andTie-barExtrusions.

					CUSTOMER[客户]:	
					TITLE[名称]:QFN 3x3-20L	
					DWGNO.[图号]:	
					PARTNO.[零件编号]:	
SIGN	QTY	REVNO.	SIGNER	DATE	REV[版本]: 0	UNIT[单位]: mm
DRAWNBY					VIEWORIENTATION	SCALE[比例]: 1:1
CHK'DBY						PAGE: 1OF 1
APPROVEDBY						
CODE[档案编号]:						

Figure25-3 3x3mm QFN20 package

25.1 Chip screen printing style

CSM32RV20
E77Q6
623AW01
.

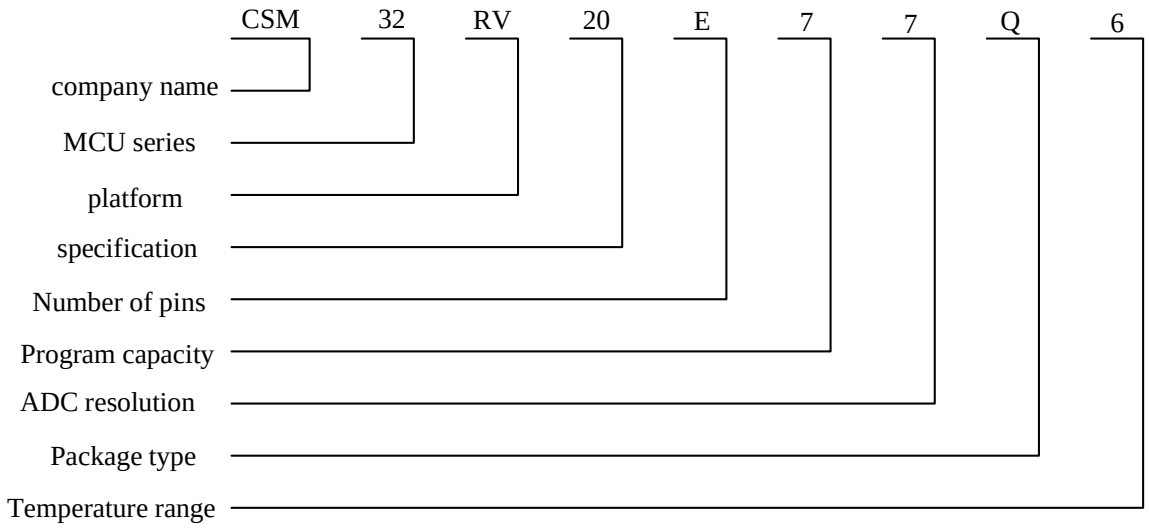
注：The MCU chip printing rules are divided into four lines, of which: the first line is the chip model number;

The second row describes the properties of the different versions;

The third row is a weekly note, a general rule for our company;

The fourth row is a dot labeled pin1 position。

25.2 Detailed description of chip screen printing letters



Pin number rules

Pin number	2	3	5	6	8	10	12	14	16
Code	1	2	3	4	5	6	7	8	A
Pin number	20	24	28	32	40	44	48	64	
Code	B	C	D	E	F	G	H	J	

Program capacity：unit kByte

Program capacity	1	2	4	8	16	32	40	64	128	256	512
Code	1	2	3	4	5	6	7	8	A	B	C

ADC：unit bits

ADC resolution	8	10	12	13	14	15	16	18	20	24
Code	1	2	3	4	5	6	7	8	A	B

Package type：

Package type	SOP	TSSOP	LQFP	QFN
Code	S	T	L	Q

Temperature range：

Temperature range	-40~85℃	-40~105℃	-40~125℃
Code	6	7	8

26 Application design-in information

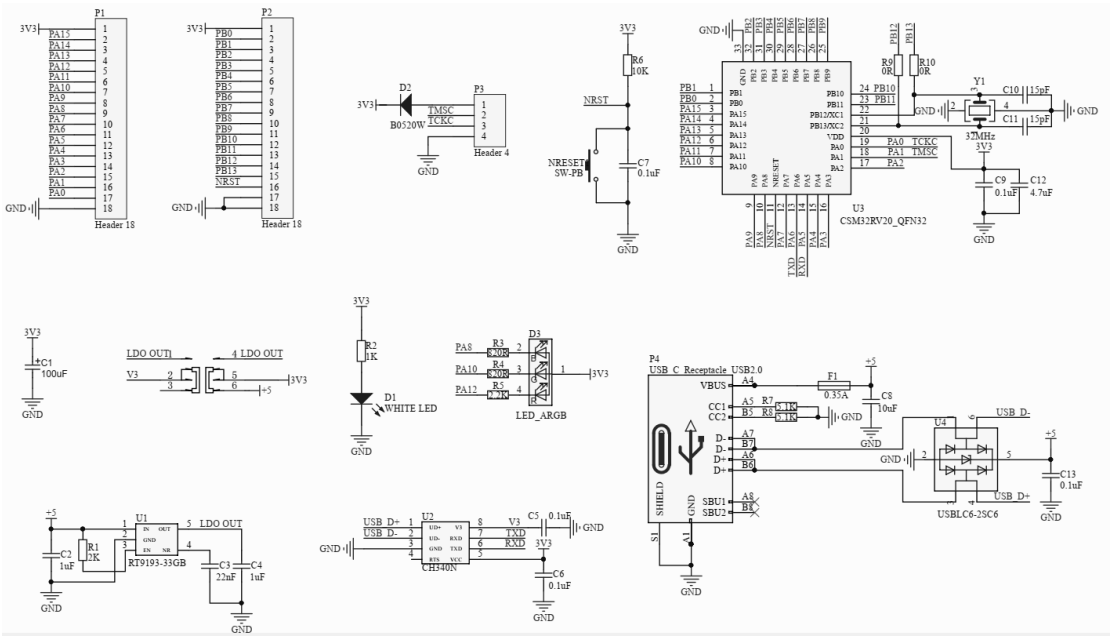


Figure26-1 Application design-in information figure（QFN32—E77N6 package）

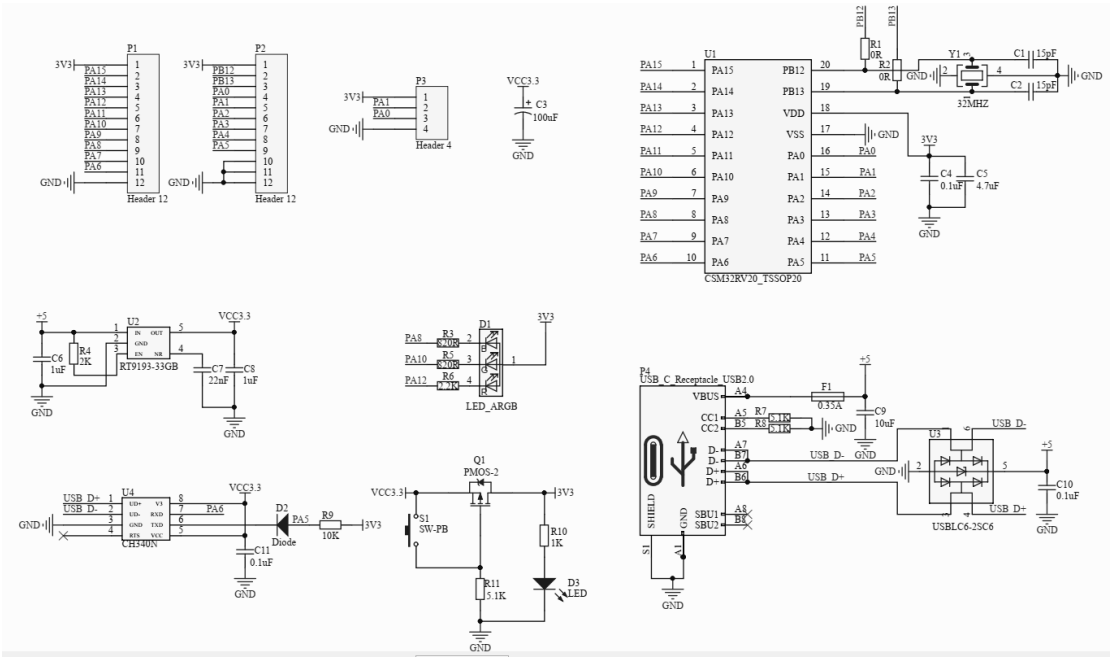


Figure26-2 Application design-in information figure（TSSOP20 package）

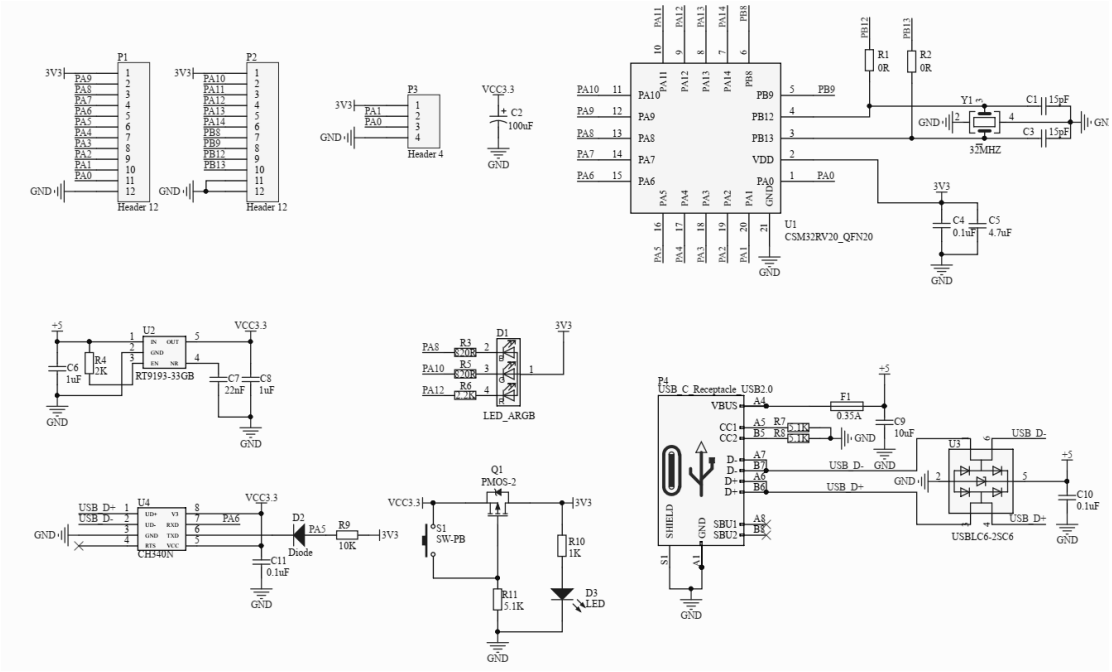


Figure26-3 Application design-in information figure（QFN20 package）

27 Revision history

Version	Modified date	Modified content
Rev1.0	2023/11/20	First draft
Rev1.1	2023/12/26	Modify pin information
Rev1.2	2024/05/11	1.Modify the Application design-in information figure of QFN32—E77N6 package 2.Add the Application design-in information figure of TSSOP20 and QFN20 package

28 Order information

Package marking

CSM32RV20 ABBCDEE

CSM32RV20:chip code

A: package date code, 5 represents year 2020

BB: week of sending out processing, 42 represents in the year A the 42th week

C: package factory code, A、HT、NJ or WA, can also abbreviated as A、H、N or W

D: test factory code, A、Z or H

EE: production batch code

Table28-1 order information

Order code	Package	Container	Minimum
CSM32RV20-Sample		Box/Tube	5
CSM32RV20 B77T6	TSSOP-20	Tape and reel	4K
CSM32RV20 B77Q6	QFN-20	Tape and reel	5K
CSM32RV20 E77Q6	QFN-32	Tape and reel	4K
CSM32RV20 E77N6	QFN-32	Tape and reel	4K

29 Technical Support and Contact information

Nanjing Zhongke Microelectronic Industry Technology Research Institute Co., Ltd
Technical Support Center
Phone: 025-68517780
Address: Room 201, Building B, Research Zone 3, Xuzhuang Software Park, Xuanwu District, Nanjing, Jiangsu, China
Website: http://www.csm-ic.com

Sales and Marketing
Phone: 13645157034, 13645157035
Email: sales@csmic.ac.cn

Technical Support
Phone: 13645157034
Email: supports@csmic.ac.cn