CSM32RV20
芯片手册

1 简介

CSM32RV20 是一款基于 RISC-V 核的低功耗 MCU 芯片。

- 内置 RISC-V RV32IMAC 内核（2.6 CoreMark/MHz）；
- 最高 32MHz 工作频率；
- 内置 4kB 的 SRAM；
- 内置 8B 的 ALWAYS 寄存器，能在掉电模式 2 下保存数据；
- 内置 4~40kB 的嵌入式 FLASH，512B 的 NVM，至少能擦写 10 000 次；
- 内置 2 个 SPI MASTER；
- 内置 1 个 I2C MASTER；
- 内置 4 个 UART 支持最高 1Mbps；
- 内置 2 个 TIMER，每个 TIMER 支持 4 路互补 PWM 输出；
- 内置 1 个快速的高精度 13/14/15/16bit ADC，集成 1.2V 高精度基准；
- 宽 ADC 输入电压范围：0 ~ VDD（VDD ≤ 4.8V）；
- ADC 支持 11 个输入通道，最多支持 9 个触摸按键；
- 内置 3 个快速比较器；
- 内置低压检测模块；
- 内置 RF 检测模块；
- 最多支持 30 个 GPIO 和 16 个外部中断；
- 内置硬件看门狗；
- 内置 1 个 RTC，在掉电模式 2 下不工作；
- 支持 4 种低功耗模式，最低功耗小于 1uA（看门狗工作）；
- 内置 32 位真随机数发生器；
- 支持串口和无线 ISP 在线升级（无线 ISP 需外接 Si24R1）；
- 支持 cJTAG 2 线调试接口；
- 工作电压范围：1.8 ~ 5.5V；
- 支持 4x4mm QFN32、TSSOP20 和 3x3mm QFN20 封装。

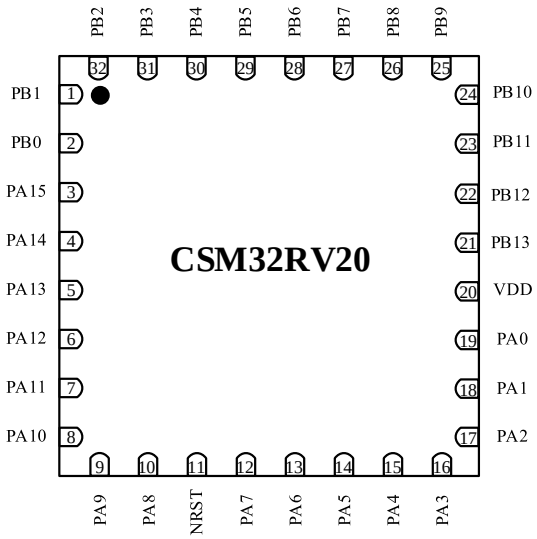


图 1-1 管脚信息图（4x4 mm QFN32）

表 1-1 管脚信息表（4x4 mm QFN32）

序号	端口	I/O	复用功能	额外功能
1	PB1	IO		-
2	PB0	IO		-
3	PA15	IO	TIM2_CH4N/TX4/ EXTI[15]	RF 检测
4	PA14	IO	ADC_TRI/TIM2_CH4/RX4/ EXTI[14]	-
5	PA13	IO	TIM2_CH3N/EXTI[13]	电压输出 REFN
6	PA12	IO	TIM2_CH3/EXTI[12]	PGA 输入
7	PA11	IO	TIM2_BKIN/TIM2_CH2N/TX3/EXTI[11]	电压输出 REFP
8	PA10	IO	TIM1_BKIN/TIM2_CH2/RX3/EXTI[10]	ADC_IN9
9	PA9	IO	TIM1_CH1/TX/TIM2_CH1N/EXTI[9]	ADC_IN8
10	PA8	IO	SDA/RX/TIM2_CH1/EXTI[8]	ADC_IN7
11	NRST	I	外部复位，低电平复位	-
12	PA7	IO	SCL/MOSI/TIM1_CH4N/EXTI[7]	ADC_IN6
13	PA6	IO	TX1/MISO/TIM1_CH4/EXTI[6]	ADC_IN5
14	PA5	IO	RX1/SCK/TIM1_CH3N/EXTI[5]	ADC_IN4
15	PA4	IO	MOSI/TIM1_CH1IN/TIM1_CH3/TX2/EXTI[4]	ADC_IN3
16	PA3	IO	MISO/TIM1_CH1N/TIM1_CH2N/RX2/EXTI[3]	ADC_IN2
17	PA2	IO	SCK/TIM1_CH1/TIM1_CH2/EXTI[2]	-
18	PA1	IO	TMSC/SDA/TIM1_CH1N/EXTI[1]	-
19	PA0	IO	TCKC/SCL/TIM1_CH1/EXTI[0]	-
20	VDD	S	电源	-
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	-

注：S：电源供电引脚；I：输入；O：输出；I/O：输入/输出；
地接在封装外壳的底部金属片上，必须接地。

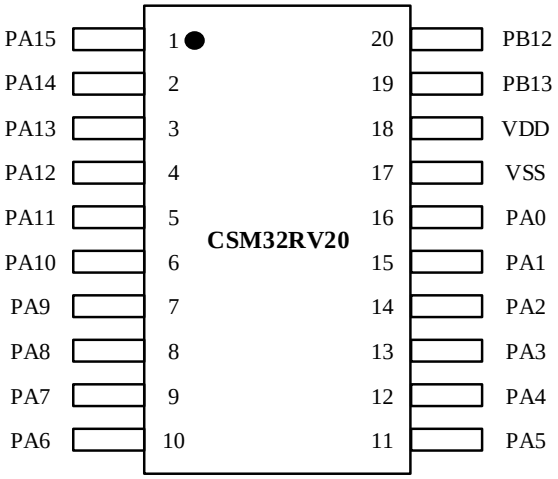


图 1-2 管脚信息图（TSSOP 20）

表 1-2 管脚信息表（TSSOP 20）

序号	端口	I/O	复用功能	额外功能
1	PA15	IO	TIM2_CH4N/TX4/ EXTI[15]	RF 检测
2	PA14	IO	ADC_TRI/TIM2_CH4/RX4/ EXTI[14]	-
3	PA13	IO	TIM2_CH3N/EXTI[13]	电压输出 REFP
4	PA12	IO	TIM2_CH3/EXTI[12]	PGA 输入
5	PA11	IO	TIM2_BKIN/TIM2_CH2N/TX3/EXTI[11]	电压输出 REFN
6	PA10	IO	TIM1_BKIN/TIM2_CH2/RX3/EXTI[10]	ADC_IN9
7	PA9	IO	TIM1_CH1/TX/TIM2_CH1N/EXTI[9]	ADC_IN8
8	PA8	IO	SDA/RX/TIM2_CH1/EXTI[8]	ADC_IN7
9	PA7	IO	SCL/MOSI/TIM1_CH4N/EXTI[7]	ADC_IN6
10	PA6	IO	TX1/MISO/TIM1_CH4/EXTI[6]	ADC_IN5
11	PA5	IO	RX1/SCK/TIM1_CH3N/EXTI[5]	ADC_IN4
12	PA4	IO	MOSI/TIM1_CH1IN/TIM1_CH3/TX2/EXTI[4]	ADC_IN3
13	PA3	IO	MISO/TIM1_CH1N/TIM1_CH2N/RX2/EXTI[3]	ADC_IN2
14	PA2	IO	SCK/TIM1_CH1/TIM1_CH2/EXTI[2]	-
15	PA1	IO	TMSC/SDA/TIM1_CH1N/EXTI[1]	-
16	PA0	IO	TCKC/SCL/TIM1_CH1/EXTI[0]	-
17	VSS	S	地	-
18	VDD	S	电源	-
19	PB13	O	-	OSC_OUT
20	PB12	I	-	OSC_IN

注：S：电源供电引脚；I：输入；O：输出；I/O：输入/输出；

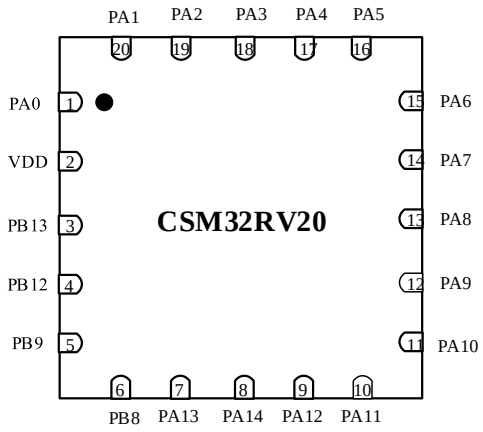


图 1-3 管脚信息图（3x3 mm QFN20）

表 1-3 管脚信息表（3x3 mm QFN20）

序号	端口	I/O	复用功能	额外功能
1	PA0	IO	TCKC/SCL/TIM1_CH1/EXTI[0]	-
2	VDD	S	电源	-
3	PB13	IO	-	OSC_OUT
4	PB12	IO	-	OSC_IN
5	PB9	IO	-	COMP2-
6	PB8	IO	-	COMP2+
7	PA13	IO	TIM2_CH3N/EXTI[13]	电压输出 REFN
8	PA14	IO	ADC_TRI/TIM2_CH4/RX4/ EXTI[14]	-
9	PA12	IO	TIM2_CH3/EXTI[12]	PGA 输入
10	PA11	IO	TIM2_BKIN/TIM2_CH2N/TX3/EXTI[11]	电压输出 REFP
11	PA10	IO	TIM1_BKIN/TIM2_CH2/RX3/EXTI[10]	ADC_IN9
12	PA9	IO	TIM1_CH1/TX/TIM2_CH1N/EXTI[9]	ADC_IN8
13	PA8	IO	SDA/RX/TIM2_CH1/EXTI[8]	ADC_IN7
14	PA7	IO	SCL/MOSI/TIM1_CH4N/EXTI[7]	ADC_IN6
15	PA6	IO	TX1/MISO/TIM1_CH4/EXTI[6]	ADC_IN5
16	PA5	IO	RX1/SCK/TIM1_CH3N/EXTI[5]	ADC_IN4
17	PA4	IO	MOSI/TIM1_CH1IN/TIM1_CH3/TX2/EXTI[4]	ADC_IN3
18	PA3	IO	MISO/TIM1_CH1N/TIM1_CH2N/RX2/EXTI[3]	ADC_IN2
19	PA2	IO	SCK/TIM1_CH1/TIM1_CH2/EXTI[2]	-
20	PA1	IO	TMSC/SDA/TIM1_CH1N/EXTI[1]	-

注：S：电源供电引脚；I：输入；O：输出；I/O：输入/输出；
地接在封装外壳的底部金属片上，必须接地。

目　录

2 存储器和总线架构

2.1 系统架构

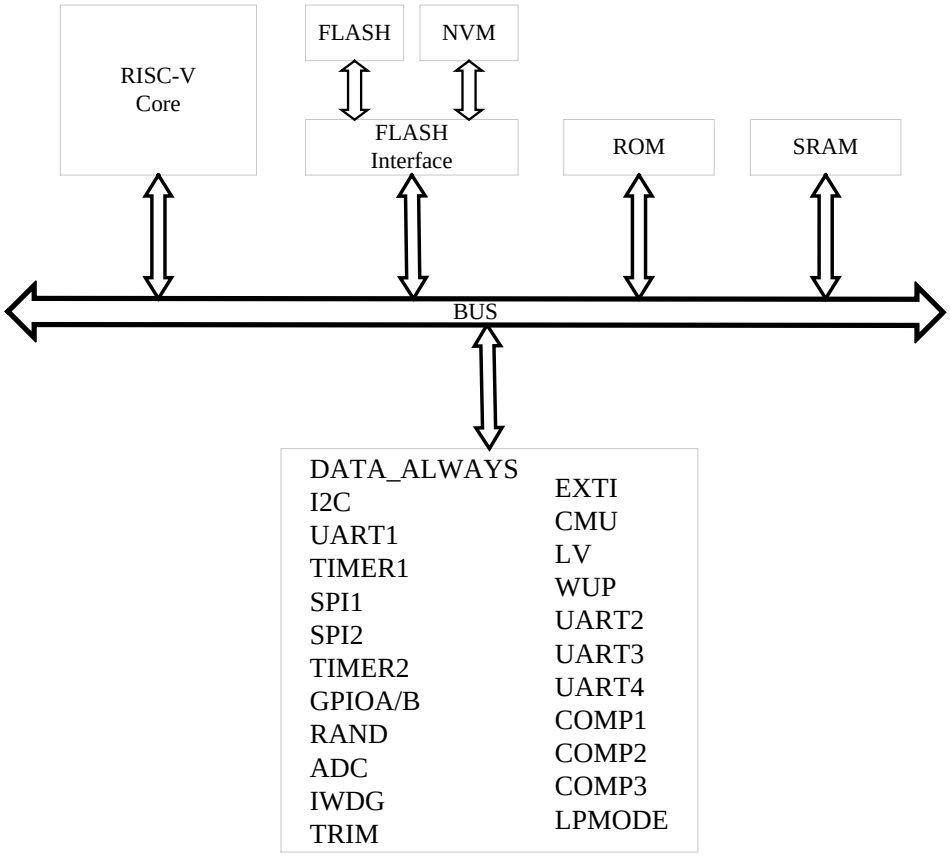


图 2-1 系统架构

2.2 存储器映射

表 2-1 存储器映射

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
0x2001_0000	0x2001_03FF	RWX	NVM	512B NVM（可以保存用户数据）
0x2002_0000	0x2002_0FFF	RWX	DATA	4kB SRAM
0x2002_8000	0x2002_8007	RWX	DATA_ALWAYS	掉电模式 2 下保存数据

Base	Top	Attr	Description	Notes
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	

2.3 嵌入式 SRAM

芯片内置了一个 4K 字节的 SRAM。支持字节、半字（16 位）或全字（32 位）访问。SRAM 的起始地址是 0x2002_0000。

2.4 嵌入式 FLASH

FLASH 主要特性：

- 10K×32 位（40K 字节）主存储空间；
- 每个扇区 512 字节；
- 1 个 NVM，512 字节；
- 支持在线读写，可用来保存用户数据；
- 支持字节、半字（16 位）或全字（32 位）读，按 8 位写；
- FLASH 读/编程/擦除操作；
- 读写保护。

2.5 嵌入式 ROM

芯片内置的 ROM，用于存储引导程序。

2.6 DATA_ALWAYS

DATA_ALWAYS 存储器能在掉电模式 2 下保存数据。每次读写占用 12 个系统时钟周期，且只能按字写。除此之外，MCU 都为单周期执行，无总线延迟。

地址（HEX）	复位值	功能
0x2002_8000	任意值	专用存储器，只有断电才会丢失数据
0x2002_8004	任意值	专用存储器，只有断电才会丢失数据

3 低功耗模式

在系统或电源复位以后，微控制器处于运行状态。运行状态下默认使用RCOSC 为 MCU 提供时钟执行程序代码。当 MCU 不需继续运行时，可以利用多个低功耗模式来节省功耗，例如等待某个外部中断时。根据最低电源消耗，最快速启动时间和可用唤醒源的需求，选取一个最佳的折中方案来帮助用户选定一个低功耗模式。

CSM32RV20 有四种低功耗模式：

- 待机模式（内核停止，外设仍可运行）；
- 睡眠模式（除 3K 时钟外，所有的时钟都可以停止）；
- 掉电模式 1（见表 3-1）；
- 掉电模式 2（见表 3-1，IO 保持掉电前状态）。

此外，在运行模式下，可以通过以下方式降低功耗：

- 降低系统时钟；
- 关闭未被使用的外设的时钟；
- 合理配置 I/O。

表 3-1 低功耗模式一览表

工作模式	LPMODE	LDO	LPLDO	RCOSC	OSC	3K[1]	唤醒
待机模式	2’b00	ON	ON	ON	ON	ON	任意中断、看门狗与复位
睡眠模式	2’b01	ON	ON	OFF	OFF	ON	任意中断、看门狗与复位
掉电模式 1	2’b10	OFF	ON	OFF	OFF	ON	任意中断、看门狗与复位
掉电模式 2	2’b11	OFF	OFF	OFF	OFF	ON	任意中断、看门狗与复位

[1] 3K 指的是在 MCU 内部低速时钟，其频率为 3kHz，且无法关闭；

3.1 低功耗模式的进入和退出

3.1.1 进入低功耗模式

配置 LPMODE 寄存器后执行 WFI 指令，如果当前没有中断，可以直接进入低功耗模式。如果当前有中断，且中断使能，MCU 无法进入低功耗模式，继续执行 WFI 指令之后的程序。

3.1.2 退出低功耗模式

为了成功退出低功耗模式，需要在执行 WFI 指令之前使能用于唤醒的中断，

并且使能对应的 clicintie（CLIC Interrupt Enable）寄存器。当执行 WFI 指令并成功进入低功耗模式后，如果已经打开了中断总使能（mstatus.MIE=1），当使能的中断产生后，MCU 唤醒跳到中断处理函数处继续执行。如果没有打开中断总使能（mstatus.MIE=0），当使能的中断产生后，MCU 唤醒，从 WFI 指令的下一条指令继续运行。

待机模式唤醒消耗的时间最少，但功耗较大。

进入睡眠模式后，OSC 和 RCOSC 振荡电路均被关闭，能够进一步降低功耗。当使能的中断产生后，根据时钟来源的不同，需要几个到几百个时钟后，才可以继续执行。

进入掉电模式 2 后，OSC 和 RCOSC 振荡电路均被关闭，数字电源关闭，FLASH 进入深度待机模式，此模式功耗最小。当使能的中断产生后，芯片立即被唤醒至少要过 160us 后，芯片复位，用户程序重新开始运行。

在掉电模式 2 下，RAM 和寄存器的值会丢失，DATA_ALWAYS 存储器的值能保留。外部复位引脚和看门狗复位可以退出低功耗模式，LPMODE 复位值为 0，芯片重新运行。

3.2 低功耗寄存器

3.2.1 低功耗模式寄存器（LPMODE）

地址：0x3000_0600

复位值：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	

位	标记	功能描述
31:2	Reserved	保留位
1:0	lpmode	2’b00：待机模式；　　2’b01：睡眠模式； 2’b10：掉电模式 1；　　2’b11：掉电模式 2。 注：LPMODE 寄存器还包含一组镜像寄存器。当芯片收到干扰导致寄存器和镜像寄存器的值不一致时，芯片会复位，重新执行

3.2.2 低功耗标志寄存器（LPRST_FLAG）

地址：0x3000_0604

复位值：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

位	标记	功能描述
31:1	Reserved	保留位
0	lprst_flag	芯片从掉电模式 2 被唤醒，重新复位标志位。 1：复位由掉电模式 2 唤醒产生； 0：复位由其他方式产生

4 复位和时钟控制

4.1 复位

复位将所有寄存器置为它们的复位状态。

当发生以下任一事件时，产生一个系统复位：

1. NRST 引脚上的低电平（外部复位）；
2. 独立看门狗计数溢出（IWDG 复位）；
3. 软件复位（SRST）；
4. 芯片上电复位；
5. 进入低功耗模式中的掉电模式 2 之后唤醒的复位。

4.2 软复位寄存器（SRST）

地址：0x3000_0360

复位值：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

位	标记	功能描述
31:1	Reserved	保留位
0	srst	写 1 软复位，系统复位

4.3 时钟

- 自动过滤晶振起振时的时钟抖动；
- 外设时钟和 MCU 时钟可以独立配置时钟源；
- 外设时钟与 MCU 时钟可以独立配置分频系数，降低系统工作频率节约功耗；
- 内置分频器支持 1:1 ~ 1:31，占空比为 50%；
- 支持无毛刺时钟切换。

4.3.1 功能介绍

4.3.1.1 时钟架构

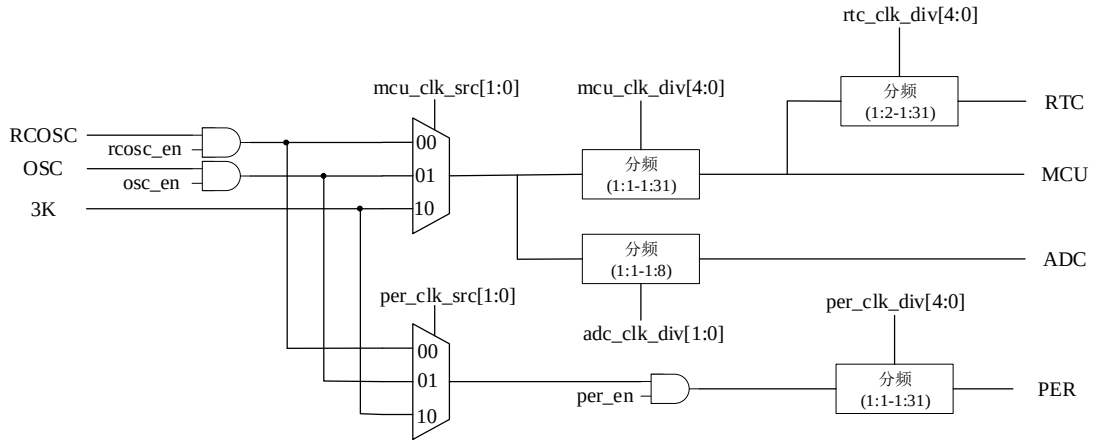


图 4-1 时钟架构

Clock 模块一共有三个时钟源，分别是内部高速振荡器（RCOSC），外部晶振（OSC）和内部低速时钟（3K）。每个时钟都支持 1:1～1:31 分频，并支持单独把 OSC 和 RCOSC 时钟源切断用以降低功耗。每一个模块的时钟输入都支持选择任意一个时钟源。

4.3.1.2 看门狗时钟

内部独立工作的看门狗时钟来自于（3K）时钟，这个时钟不能被关闭。一旦看门狗被打开，除非复位，不能被关闭；芯片支持上电复位启动看门狗，在 FLASH 下载程序时配置。

4.3.1.3 时钟模块的控制

- 在复位之后，默认选择 RCOSC 为当前的系统时钟；
- 当外设或者 MCU 选中某个时钟源时，被选时钟源不能被关闭；
- 切换时钟源时应该首先确认需要被切换的时钟源是否已稳定，否则无法完成切换；
- RTC 的时钟频率应该小于 MCU 时钟频率的 1/2。

4.4 时钟控制寄存器

基地址：0x3000_02E4.

4.4.1 外设使能寄存器（CMU_PER_EN）

偏移地址：0x00
复位值：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

位	标记	功能描述
31:1	Reserved	保留位
0	per_en	外设时钟使能位，0：关闭外设时钟　1：使能外设时钟

4.4.2 时钟源选择寄存器（CMU_CLK_SEL）

偏移地址：0x04
复位值：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	

位	标记	功能描述
31:4	Reserved	保留位
3:2	per_clk_src	外设时钟选择: 00：外设时钟源来自内部高速时钟 RCOSC； 01：外设时钟源来自外部高速晶振 OSC； 10：外设时钟来自内部低速时钟 3K； 11：-
1:0	mcu_clk_src	MCU 时钟来源选择 00：MCU 时钟来自于内部高速时钟 RCOSC； 01：MCU 时钟来自于外部高速晶振 OSC； 10：MCU 时钟来自于内部低速时钟 3K； 11：-

4.4.3 时钟分频寄存器（CMU_CLK_DIV）

偏移地址：0x08

复位值：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				

位	标记	功能描述
31:15	Reserved	保留位
14:10	rtc_clk_div	RTC 时钟分频系数： 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 注：x 为 0 或 1。
9：5	per_clk_div	外设时钟分频系数： 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 时钟分频系数 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 时钟源开关寄存器（CLK_SRC_EN）

偏移地址：0x0C
复位值：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

位	标记	功能描述
31:2	Reserved	保留位
1	rcosc_en	RCOSC 开关 0：关闭 RCOSC；　　1：打开 RCOSC。 当 RCOSC 作为系统内部时钟输入的时候，即使 rcosc_en 配置成 0 也不会关闭当前的时钟信号输入，直到内部时钟被切换成其他时钟源之后，rcosc 才会被关闭
0	osc_en	晶振（OSC）开关 0：晶振关闭；　　1：晶振打开

4.4.5 时钟状态寄存器（CMU_OSC_SR）

偏移地址：0x10
复位值：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		

位	标记	功能描述
31:8	Reserved	保留位
7	osc_st	OSC 的工作状态 0：OSC 时钟未稳定；1：OSC 时钟已稳定
6	rcosc_st	RCOSC 的工作状态 0：RCOSC 时钟未稳定；1：RCOSC 时钟已稳定
5:3	mcu_clk_st	MCU 的时钟来源选择状态寄存器 001：RCOSC 在为 MCU 提供时钟； 010：OSC 在为 MCU 提供时钟； 100：内部 3K 时钟为 MCU 提供时钟
2:0	per_clk_st	外设的时钟来源选择状态寄存器 001：RCOSC 在为外设提供时钟；

位	标记	功能描述
		010：OSC 在为外设提供时钟； 100：内部 3K 时钟为外设提供时钟

4.4.6 RCOSC 频率选择（RCOSC_SEL）

地址：0x3000_0E00

复位值：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

位	标记	功能描述
31:1	Reserved	保留位
0	rcosc_sel	RCOSC 时钟频率选择， 0：RCOSC 频率 16MHz；　1：RCOSC 频率 32MHz

4.5 CMU 寄存器映射

CMU 寄存器列表

基地址：0x3000_02E4

寄存器	偏移量	寄描述
CMU_PER_EN	0x00	外设时钟使能控制寄存器
CMU_CLK_SEL	0x04	时钟来源选择
CMU_CLK_DIV	0x08	时钟分频比
CLK_SRC_EN	0x0C	时钟源使能
CMU_OSC_SR	0x10	OSC 时钟状态寄存器
RCOSC_SEL	-	RCOSC 频率选择

5 通用和复用功能 I/O

GPIO 是用户可配置的通用 IO，每一个 GPIO 口都可以独立配置成输入输出、外设复用功能或模拟功能。GPIOA0~15 对应 PA0~PA15，GPIOB0~13 对应 PB0~PB13。

5.1 GPIO 功能描述

5.1.1 主要功能

- 输出状态：上拉、下拉功能，开源输出，开漏输出和推挽输出；
- 掉电模式 2 下，IO 会保持掉电前的状态不变；
- GPIO 的输出可以来自于 GPIO 的 ODR 寄存器或者外设功能输出；
- 输入状态：悬空输入，上拉、下拉输入；
- 输入的数据会被存入 GPIO 的 IDR 寄存器或者外设数据输入；
- 支持模拟功能数据传输；
- 外设功能接口可选；
- 可以灵活的为每一个 GPIO 选择对应的输入口；
- IO 模式由 GPIO_MODER 寄存器选择输入模式、输出模式、复用模式和模拟模式。

5.1.1.1 复用功能

本芯片的外部 IO 和内部模块的连接复用器被 GPIOA_AFRH 和 GPIOA_AFRL 寄存器控制，写寄存器 GPIO_MODER[MODERx] = 2’b10 将 GPIOx 配置成复用功能，可以灵活的将内部模块的端口映射到 PAD 上。

每一个 IO 都有一个拆分器把复用功能连通，通过 GPIO_AFRL 和 GPIO_AFRH 两个寄存器来控制具体复用到哪个功能，复用详情见表 5-1。

在复位之后，复用控制器会默认把 PAD 连接到 AF0 功能上，串口 1 支持 ISP。

表 5-1 GPIO 复用功能表

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_	O		

Pin Name	AF0	AF0_DIR	AF1	AF1_DIR	AF2	AF2_DIR	AF3	AF3_DIR
					CH1N			
PA2	SPI1_SCK	O	TIM1_CH1	O	TIMER1_CH2	IO		
PA3	SPI1_MISO	I	TIM1_CH1N	O	TIMER1_CH2N	O	UART2 RX2	IO
PA4	SPI1_MOSI	O	TIM1_CH1N	O	TIMER1_CH3	IO	UART2 TX2	O
PA5	UART1 RX1	I	SPI1_SCK	O	TIMER1_CH3N	O		
PA6	UART1 TX1	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 RX3	IO
PA11	TIMER2_BKIN	I		O	TIMER2_CH2N	O	UART3 TX3	O
PA12	-	-		O	TIMER2_CH3	O		
PA13	-	-			TIMER2_CH3N	O		
PA14	ADC_TRI				TIMER2_CH4	IO	UART4 RX4	IO
PA15					TIMER2_CH4N	O	UART4 TX4	O
PB0	-	-	-	-	-	-	-	-
PB1	-	-	-	-	-	-	-	-
PB2	SPI2_SCK	O	-	-	-	-	-	-
PB3	SPI2_MOSI	O	-	-	-	-	-	-
PB4	SPI2_MISO	I	-	-	-	-	-	-

Pin Name	AF0	AF0_DIR	AF1	AF1_DIR	AF2	AF2_DIR	AF3	AF3_DIR
PB5	-							
……..	-							
PB13	-							

注：I：输入；O：输出；I/O：输入/输出；

5.1.1.2 模拟功能

IO 功能由 GPIO_MODER 寄存器配置。使用 ADC、COMP 等模块时需要配置 IO 为模拟模式 GPIO_MODER[MODERx] = 2’b11，支持模拟功能数据传输。

表 5-2 GPIO 模拟功能表

引脚	I/O	模拟模式 （GPIO 配置）	备注
PA3	I	ADC_IN2（ADC 通道）	
PA4	I	ADC_IN3（ADC 通道）	
PA5	I	ADC_IN4（ADC 通道）	
PA6	I	ADC_IN5（ADC 通道）	
PA7	I	ADC_IN6（ADC 通道）	
PA8	I	ADC_IN7（ADC 通道）	
PA9	I	ADC_IN8（ADC 通道） ADC 外部基准高电压端	
PA10	I	ADC_IN9（ADC 通道） ADC 外部基准低电压端	
PA11	O	电压输出 REFP	仅能拉电流
PA12	I	PGA 输入	
PA13	O	电压输出 REFN	仅能灌电流
PA14	O	保留	
PA15	I	RF 检测输入	
PB6	I	COMP1+	
PB7	I	COMP1-	
PB8	I	COMP2+	
PB9	I	COMP2-	
PB10	I	COMP3+	
PB11	I	COMP3-	
PB12	I	XC1，OSC 输入	

引脚	I/O	模拟模式 （GPIO 配置）	备注
PB13	I	XC2，OSC 输出	

注：I：输入；O：输出；I/O：输入/输出；

5.1.2 输入配置

当一个 IO 口被配置成输入模式（GPIO_MODER[MODERx] = 2’b00）时，

1. 输出寄存器会被关闭；
2. 施密特触发器打开；
3. 上拉下拉控制口会根据 GPIO_PUPDR 寄存器进行配置；
4. 每个 AHB 周期，输入的数据会被输入寄存器刷新一次；
5. 每个寄存器代表一个 IO 口的输入值。

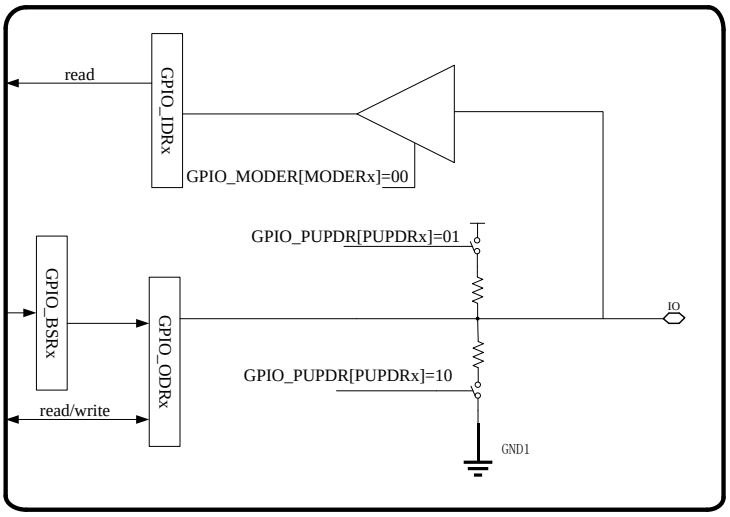


图 5-1 GPIO 输入配置原理图

当 GPIO 被设置为输入模式时，可被用作外部中断。EXTI[15:0]对应 GPIOA15 ~ 0。中断的使能和屏蔽及触发方式的设置可以在 GPIOA_LPMR 和 GPIOA_INTER 中设置。

表 5-3 GPIO 输入功能表

引脚	I/O	输入模式 （GPIO 配置）	备注
PA0	I	EXTI[0]	
PA1	I	EXTI[1]	
PA2	I	EXTI[2]	
PA3	I	EXTI[3]	
PA4	I	EXTI[4]	

引脚	I/O	输入模式 （GPIO 配置）	备注
PA5	I	EXTI[5]	
PA6	I	EXTI[6]	
PA7	I	EXTI[7]	
PA8	I	EXTI[8]	
PA9	I	EXTI[9]	
PA10	I	EXTI[10]	
PA11	I	EXTI[11]	
PA12	I	EXTI[12]	
PA13	I	EXTI[13]	
PA14	I	EXTI[14]	
PA15	I	EXTI[15]	

5.1.3 输出配置

当一个 IO 口被设置成输出模式（GPIO_MODER[MODERx] = 2’b01）时，

1. 输出数据寄存器会被打开；
2. 施密特触发器输入模式打开；
3. 上拉、下拉控制口会根据 GPIO_PUPDR 寄存器进行配置；
4. 每个 AHB 周期，IO 端口的数据会被输入寄存器刷新一次；
5. 每个写周期，都会把输出寄存器里面的数据送到 IO 口上；
6. 当开漏和开源输出都关闭时，为推挽输出。

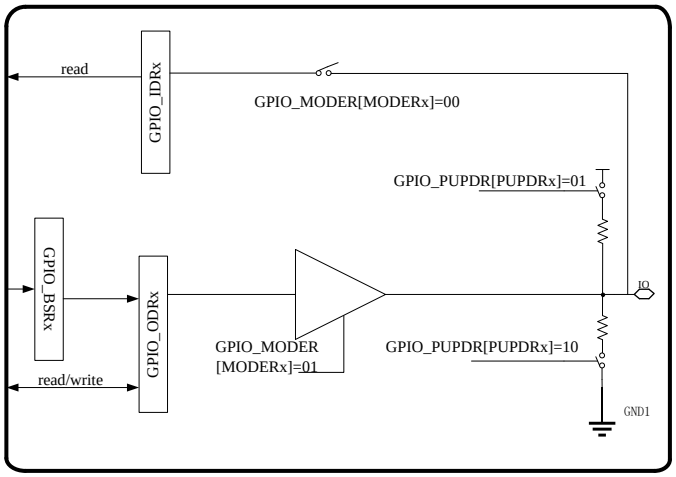


图 5-2 GPIO 输出配置原理图

5.1.4 复用功能配置

当配置成复用功能（GPIO_MODER[MODERx] = 2’b10）时，

1. 出数据寄存器被芯片内部的模块端口驱动；
2. 口的输入输出方向被模块内部的控制信号决定；
3. 施密特触发器输入模式被激活；
4. 模块的上拉、下拉不再受 GPIO_PUPDR 控制，而是受到与之相连的模块决定。

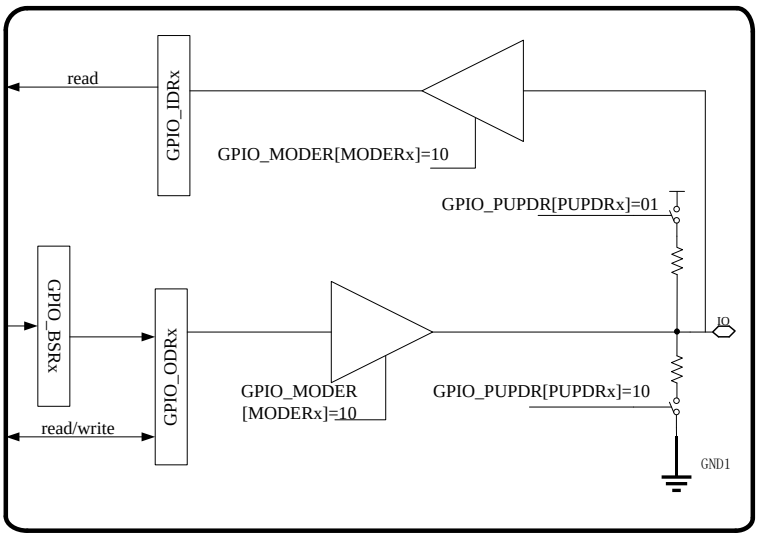


图 5-3 GPIO 复用功能配置原理图

5.1.5 模拟功能配置

当配置成模拟功能时（GPIO_MODER[MODERx] = 2’b11）

1. 输出数据寄存器会被关闭；
2. 施密特触发器会被关闭，输入数据总为 0。

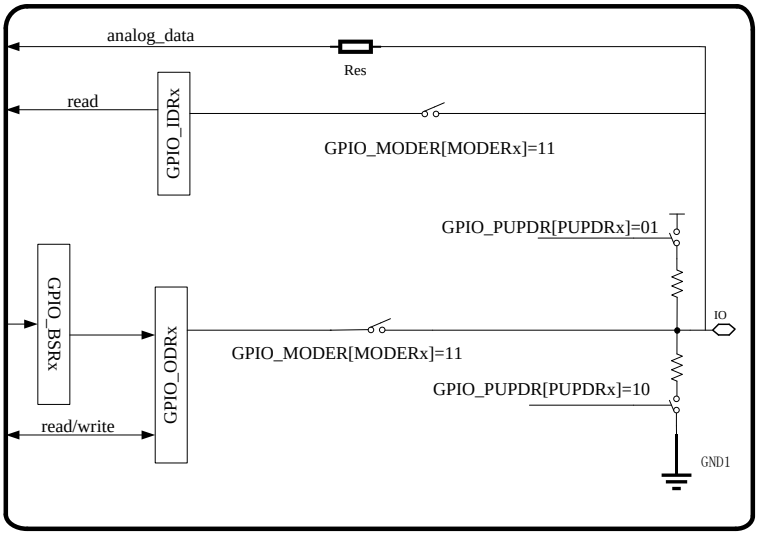


图 5-4 GPIO 模拟功能配置原理图

5.2 GPIOA 寄存器描述

5.2.1 GPIOA 模式控制寄存器（GPIOA_MODER）

偏移地址：0x00
复位值：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

位	标记	功能描述
31:30	MODER15	GPIOA15 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能。
29:28	MODER14	GPIOA14 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能。
27:26	MODER13	GPIOA13 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能。
25:24	MODER12	GPIOA12 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能。
23:22	MODER11	GPIOA11 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能。
21:20	MODER10	GPIOA10 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能。
19:18	MODER9	GPIOA9 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能。
17:16	MODER8	GPIOA8 端口控制位

位	标记	功能描述
		00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能。
15:14	MODER7	GPIOA7 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能。
13:12	MODER6	GPIOA6 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能。
11:10	MODER5	GPIOA5 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能。
9:8	MODER4	GPIOA4 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能。
7:6	MODER3	GPIOA3 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能。
5:4	MODER2	GPIOA2 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能。
3:2	MODER1	GPIOA1 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能。
1:0	MODER0	GPIOA0 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能。

5.2.2　GPIOA 输出控制寄存器（GPIOA_OTYPER）

偏移地址：0x04
复位值：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															

位	标记	功能描述
31	OS15	GPIOA15 开源控制位 0：开源功能关闭；　　1：开源功能打开
30	OS14	GPIOA14 开源控制位 0：开源功能关闭；　　1：开源功能打开
29	OS13	GPIOA13 开源控制位 0：开源功能关闭；　　1：开源功能打开
28	OS12	GPIOA12 开源控制位 0：开源功能关闭；　　1：开源功能打开
27	OS11	GPIOA11 开源控制位 0：开源功能关闭；　　1：开源功能打开
26	OS10	GPIOA10 开源控制位 0：开源功能关闭；　　1：开源功能打开
25	OS9	GPIOA9 开源控制位 0：开源功能关闭；　　1：开源功能打开
24	OS8	GPIOA8 开源控制位 0：开源功能关闭；　　1：开源功能打开
23	OS7	GPIOA7 开源控制位 0：开源功能关闭；　　1：开源功能打开
22	OS6	GPIOA6 开源控制位 0：开源功能关闭；　　1：开源功能打开
21	OS5	GPIOA5 开源控制位 0：开源功能关闭；　　1：开源功能打开
20	OS4	GPIOA4 开源控制位 0：开源功能关闭；　　1：开源功能打开
19	OS3	GPIOA3 开源控制位 0：开源功能关闭；　　1：开源功能打开
18	OS2	GPIOA2 开源控制位 0：开源功能关闭；　　1：开源功能打开
17	OS1	GPIOA1 开源控制位 0：开源功能关闭；　　1：开源功能打开
16	OS0	GPIOA0 开源控制位 0：开源功能关闭；　　1：开源功能打开
15	OD15	GPIOA15 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
14	OD14	GPIOA14 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开。
13	OD13	GPIOA13 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
12	OD12	GPIOA12 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
11	OD11	GPIOA11 开漏控制位

位	标记	功能描述
		0：开漏功能关闭；　　1：开漏功能打开
10	OD10	GPIOA10 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
9	OD9	GPIOA9 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
8	OD8	GPIOA8 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
7	OD7	GPIOA7 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
6	OD6	GPIOA6 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
5	OD5	GPIOA5 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
4	OD4	GPIOA4 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
3	OD3	GPIOA3 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
2	OD2	GPIOA2 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
1	OD1	GPIOA1 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
0	OD0	GPIOA0 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开

注：当开漏和开源输出功能都关闭时，IO 被设置为推挽输出。

5.2.3　GPIOA 输入模式控制寄存器（GPIOA_ITYPER）

偏移地址：0x08
复位值：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															

位	标记	功能描述
15	CS15	设置 GPIOA15 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
14	CS14	设置 GPIOA14 的输入模式

位	标记	功能描述
		0：施密特触发器模式； 1：CMOS 输入模式
13	CS13	设置 GPIOA13 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
12	CS12	设置 GPIOA12 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
11	CS11	设置 GPIOA11 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
10	CS10	设置 GPIOA10 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
9	CS9	设置 GPIOA9 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
8	CS8	设置 GPIOA8 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
7	CS7	设置 GPIOA7 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
6	CS6	设置 GPIOA6 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
5	CS5	设置 GPIOA5 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
4	CS4	设置 GPIOA4 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
3	CS3	设置 GPIOA3 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
2	CS2	设置 GPIOA2 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
1	CS1	设置 GPIOA1 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式
0	CS0	设置 GPIOA0 的输入模式 0：施密特触发器模式； 1：CMOS 输入模式

5.2.4 GPIOA 上下拉控制寄存器（GPIOA_PUPDR）

偏移地址：0x0C
复位值：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

位	标记	功能描述
31:30	PUPDR15	GPIOA15 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
29:28	PUPDR14	GPIOA14 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
27:26	PUPDR13	GPIOA13 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
25:24	PUPDR12	GPIOA12 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
23:22	PUPDR11	GPIOA11 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
21:20	PUPDR10	GPIOA10 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
19:18	PUPDR9	GPIOA9 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
17:16	PUPDR8	GPIOA8 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
15:14	PUPDR7	GPIOA7 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
13:12	PUPDR6	GPIOA6 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
11:10	PUPDR5	GPIOA5 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
9:8	PUPDR4	GPIOA4 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
7:6	PUPDR3	GPIOA3 端口配置成上拉还是下拉

位	标记	功能描述
		00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
5:4	PUPDR2	GPIOA2 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
3:2	PUPDR1	GPIOA1 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
1:0	PUPDR0	GPIOA0 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位

5.2.5　GPIOA 性能控制寄存器（GPIOA_SDR）

偏移地址：0x10
复位值：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															

位	标记	功能描述
31	SR15	GPIOA15 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
30	SR14	GPIOA14 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
29	SR13	GPIOA13 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
28	SR12	GPIOA12 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
27	SR11	GPIOA11 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
26	SR10	GPIOA10 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
25	SR9	GPIOA9 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率

位	标记	功能描述
24	SR8	GPIOA8 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
23	SR7	GPIOA7 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
22	SR6	GPIOA6 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
21	SR5	GPIOA5 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
20	SR4	GPIOA4 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
19	SR3	GPIOA3 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
18	SR2	GPIOA2 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
17	SR1	GPIOA1 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
16	SR0	GPIOA0 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
15	DR15	GPIOA15 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
14	DR14	GPIOA14 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
13	DR13	GPIOA13 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
12	DR12	GPIOA12 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
11	DR11	GPIOA11 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
10	DR10	GPIOA10 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
9	DR9	GPIOA9 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
8	DR8	GPIOA8 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
7	DR7	GPIOA7 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
6	DR6	GPIOA6 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
5	DR5	GPIOA5 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
4	DR4	GPIOA4 对应的 PAD 的驱动能力

位	标记	功能描述
		1：高驱动能力；　　0：低驱动能力
3	DR3	GPIOA3 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
2	DR2	GPIOA2 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
1	DR1	GPIOA1 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力
0	DR0	GPIOA0 对应的 PAD 的驱动能力 1：高驱动能力；　　0：低驱动能力

5.2.6　GPIOA 中断模式寄存（GPIOA_LPMR）

偏移地址：0x14
复位值：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	

位	标记	功能描述
31:0	LPMRx	GPIOAx 对应 PAD 的外部中断检测控制位 LPMRx[2x + 1: 2x]：MODE1，MODE0 00：高电平检测；　　01：下降沿检测； 10：上升沿检测；　　11：低电平检测

5.2.7　GPIOA 外部中断采集使能寄存器（GPIOA_INTER）

偏移地址：0x18
复位值：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															

位	标记	功能描述
15:0	INT_EN	GPIOAx 对应 PAD 的外部中断检测使能位 0：不接收来自 PAD 的外部中断输入； 1：接收来自 PAD 的外部中断输入

5.2.8 GPIOA 输入数据寄存器（GPIOA_IDR）

偏移地址：0x1C
复位值：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

位	标记	功能描述
15:0	IDRx	GPIOAx 的接收数据输入

5.2.9 GPIOA 输出数据寄存器（GPIOA_ODR）

偏移地址：0x20
复位值：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															

位	标记	功能描述
15:0	ODRx	GPIOAx 的接收数据输出控制位

5.2.10 GPIOA 写使能控制寄存器（GPIOA_BSR）

偏移地址：0x24
复位值：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

位	标记	功能描述
31:16	BRx	清除端口 x 的位 y（y = 0…15） 这些位只能写入并只能以字（16 位）的形式操作 0：对于对应的 ODRy 位不产生影响； 1：清除对应的 ODRy 位为 0。 注：如果同时设置了 BSy 和 BRy 的对应位，BSy 位起作用
15:0	BSx	设置端口 x 的位 y（y = 0…15） 这些位只能写入并只能以字（16 位）的形式操作 0：对于对应的 ODRy 位不产生影响； 1：设置对应的 ODRy 位为 1

5.2.11 GPIOA 复用控制寄存器高位（GPIOA_AFRH）

偏移地址：0x28
复位值：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			

位	标记	功能描述
31:28	AFSEH15[3:0]	对应的 GPIOA15 的复用功能选择寄存器

位	标记	功能描述
		0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
27:24	AFSEH14[3:0]	对应的 GPIOA14 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
23:20	AFSEH13[3:0]	对应的 GPIOA13 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
19:16	AFSEH12[3:0]	对应的 GPIOA12 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
15:12	AFSEH11[3:0]	对应的 GPIOA11 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
11:8	AFSEH10[3:0]	对应的 GPIOA10 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
7:4	AFSEH9[3:0]	对应的 GPIOA9 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
3:0	AFSEH8[3:0]	对应的 GPIOA8 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3

5.2.12 GPIOA 复用控制寄存器低位（GPIOA_AFRL）

偏移地址：0x2C
复位值：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

位	标记	功能描述
31:28	AFSEL7[3:0]	对应的 GPIOA7 的复用功能选择寄存器 0000：AF0； 0001：AF1；

位	标记	功能描述
		0010：AF2； 0011：AF3
27:24	AFSEL6[3:0]	对应的 GPIOA6 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
23:20	AFSEL5[3:0]	对应的 GPIOA5 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
19:16	AFSEL4[3:0]	对应的 GPIOA4 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
15:12	AFSEL3[3:0]	对应的 GPIOA3 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
11:8	AFSEL2[3:0]	对应的 GPIOA2 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
7:4	AFSEL1[3:0]	对应的 GPIOA1 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3
3:0	AFSEL0[3:0]	对应的 GPIOA0 的复用功能选择寄存器 0000：AF0； 0001：AF1； 0010：AF2； 0011：AF3

5.3 GPIOA 寄存器映射

GPIOA 寄存器列表

基地址：0x3000_0200

寄存器	偏移地址	寄存器描述
GPIOA_MODER	0x00	GPIOA 模式控制寄存器
GPIOA_OTYPER	0x04	GPIOA 输出控制寄存器
GPIOA_ITYPER	0x08	GPIOA 输入控制寄存器
GPIOA_PUPDR	0x0C	GPIOA 上拉下拉控制寄存器
GPIOA_SDR	0x10	GPIOA 性能控制寄存器
GPIOA_LPMR	0x14	GPIOA 外部中断检测控制寄存器
GPIOA_INTER	0x18	GPIOA 外部中断使能控制寄存器
GPIOA_IDR	0x1C	GPIOA 输入数据寄存器
GPIOA_ODR	0x20	GPIOA 输出数据寄存器
GPIOA BSR	0x24	GPIOA 写使能控制寄存器
GPIOA_AFRH	0x28	GPIOA 复用控制寄存器高位

GPIOA_AFRL	0x2C	GPIOA 复用控制寄存器低位

5.4 GPIOB 寄存器描述

5.4.1 GPIOB 模式控制寄存器（GPIOB_MODER）

偏移地址：0x00
复位值：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	

位	标记	功能描述
31:28	Reserved	保留位
27:26	MODER13	GPIOB13 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能
25:24	MODER12	GPIOB12 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能
23:22	MODER11	GPIOB11 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能
21:20	MODER10	GPIOB10 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能
19:18	MODER9	GPIOB9 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能
17:16	MODER8	GPIOB8 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能
15:14	MODER7	GPIOB7 端口控制位 00：输入模式；　01：输出模式； 10：复用功能；　11：模拟功能

位	标记	功能描述
13:12	MODER6	GPIOB6 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能
11:10	MODER5	GPIOB5 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能
9:8	MODER4	GPIOB4 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能
7:6	MODER3	GPIOB3 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能
5:4	MODER2	GPIOB2 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能
3:2	MODER1	GPIOB1 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能
1:0	MODER0	GPIOB0 端口控制位 00：输入模式；　　01：输出模式； 10：复用功能；　　11：模拟功能

5.4.2　GPIOB 输出控制寄存器（GPIOB_OTYPER）

偏移地址：0x04
复位值：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													

位	标记	功能描述
31:30	Reserved	保留位
29	OS13	GPIOB13 开源控制位 0：开源功能关闭；　　1：开源功能打开
28	OS12	GPIOB12 开源控制位

位	标记	功能描述
		0：开源功能关闭； 1：开源功能打开
27	OS11	GPIOB11 开源控制位 0：开源功能关闭； 1：开源功能打开
26	OS10	GPIOB10 开源控制位 0：开源功能关闭； 1：开源功能打开
25	OS9	GPIOB9 开源控制位 0：开源功能关闭； 1：开源功能打开
24	OS8	GPIOB8 开源控制位 0：开源功能关闭； 1：开源功能打开
23	OS7	GPIOB7 开源控制位 0：开源功能关闭； 1：开源功能打开
22	OS6	GPIOB6 开源控制位 0：开源功能关闭； 1：开源功能打开
21	OS5	GPIOB5 开源控制位 0：开源功能关闭； 1：开源功能打开
20	OS4	GPIOB4 开源控制位 0：开源功能关闭； 1：开源功能打开
19	OS3	GPIOB3 开源控制位 0：开源功能关闭； 1：开源功能打开
18	OS2	GPIOB2 开源控制位 0：开源功能关闭； 1：开源功能打开
17	OS1	GPIOB1 开源控制位 0：开源功能关闭； 1：开源功能打开
16	OS0	GPIOB0 开源控制位 0：开源功能关闭； 1：开源功能打开
15:14	Reserved	保留位
13	OD13	GPIOB13 开漏控制位 0：开漏功能关闭； 1：开漏功能打开
12	OD12	GPIOB12 开漏控制位 0：开漏功能关闭； 1：开漏功能打开
11	OD11	GPIOB11 开漏控制位 0：开漏功能关闭； 1：开漏功能打开
10	OD10	GPIOB10 开漏控制位 0：开漏功能关闭； 1：开漏功能打开
9	OD9	GPIOB9 开漏控制位 0：开漏功能关闭； 1：开漏功能打开
8	OD8	GPIOB8 开漏控制位 0：开漏功能关闭； 1：开漏功能打开
7	OD7	GPIOB7 开漏控制位 0：开漏功能关闭； 1：开漏功能打开
6	OD6	GPIOB6 开漏控制位

位	标记	功能描述
		0：开漏功能关闭；　　1：开漏功能打开
5	OD5	GPIOB5 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
4	OD4	GPIOB4 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
3	OD3	GPIOB3 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
2	OD2	GPIOB2 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
1	OD1	GPIOB1 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开
0	OD0	GPIOB0 开漏控制位 0：开漏功能关闭；　　1：开漏功能打开

注：当开漏和开源输出功能都关闭时，IO 被设置为推挽输出。

5.4.3　GPIOB 输入模式控制寄存器（GPIOB_ITYPER）

偏移地址：0x08
复位值：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													

位	标记	功能描述
15:14	Reserved	保留位
13	CS13	设置 GPIOB13 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
12	CS12	设置 GPIOB12 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
11	CS11	设置 GPIOB11 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
10	CS10	设置 GPIOB10 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
9	CS9	设置 GPIOB9 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
8	CS8	设置 GPIOB8 的输入模式

位	标记	功能描述
		0：施密特触发器模式；　　1：CMOS 输入模式
7	CS7	设置 GPIOB7 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
6	CS6	设置 GPIOB6 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
5	CS5	设置 GPIOB5 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
4	CS4	设置 GPIOB4 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
3	CS3	设置 GPIOB3 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
2	CS2	设置 GPIOB2 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
1	CS1	设置 GPIOB1 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式
0	CS0	设置 GPIOB0 的输入模式 0：施密特触发器模式；　　1：CMOS 输入模式

5.4.4　GPIOB 上下拉控制寄存器（GPIOB_PUPDR）

偏移地址：0x0C
复位值：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	

位	标记	功能描述
31:28	Reserved	保留位
27:26	PUPDR13	GPIOB13 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
25:24	PUPDR12	GPIOB12 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉；

位	标记	功能描述
		10：下拉；　　11：保留位
23:22	PUPDR11	GPIOB11 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
21:20	PUPDR10	GPIOB10 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
19:18	PUPDR9	GPIOB9 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
17:16	PUPDR8	GPIOB8 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
15:14	PUPDR7	GPIOB7 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
13:12	PUPDR6	GPIOB6 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
11:10	PUPDR5	GPIOB5 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
9:8	PUPDR4	GPIOB4 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
7:6	PUPDR3	GPIOB3 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
5:4	PUPDR2	GPIOB2 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
3:2	PUPDR1	GPIOB1 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位
1:0	PUPDR0	GPIOB0 端口配置成上拉还是下拉 00：不上拉也不下拉；　　01：上拉； 10：下拉；　　11：保留位

5.4.5 GPIOB 性能控制寄存器（GPIOB_SDR）

偏移地址：0x10
复位值：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													

位	标记	功能描述
31:30	Reserved	保留位
29	SR13	GPIOB13 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
28	SR12	GPIOB12 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
27	SR11	GPIOB11 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
26	SR10	GPIOB10 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
25	SR9	GPIOB9 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
24	SR8	GPIOB8 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
23	SR7	GPIOB7 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
22	SR6	GPIOB6 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
21	SR5	GPIOB5 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
20	SR4	GPIOB4 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
19	SR3	GPIOB3 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
18	SR2	GPIOB2 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
17	SR1	GPIOB1 对应的 PAD 的压摆率 1：快速压摆率；　　0：低速压摆率
16	SR0	GPIOB0 对应的 PAD 的压摆率

位	标记	功能描述
		1：快速压摆率； 0：低速压摆率
15:14	Reserved	保留位
13	DR13	GPIOB13 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
12	DR12	GPIOB12 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
11	DR11	GPIOB11 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
10	DR10	GPIOB10 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
9	DR9	GPIOB9 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
8	DR8	GPIOB8 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
7	DR7	GPIOB7 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
6	DR6	GPIOB6 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
5	DR5	GPIOB5 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
4	DR4	GPIOB4 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
3	DR3	GPIOB3 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
2	DR2	GPIOB2 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
1	DR1	GPIOB1 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力
0	DR0	GPIOB0 对应的 PAD 的驱动能力 1：高驱动能力； 0：低驱动能力

5.4.6 GPIOB 输入数据寄存器（GPIOB_IDR）

偏移地址：0x1C
复位值：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

位	标记	功能描述
13:0	IDRx	GPIOBx 的接收数据输入

5.4.7 GPIOB 输出数据寄存器（GPIOB_ODR）

偏移地址：0x20
复位值：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							

位	标记	功能描述
13:0	ODRx	GPIOBx 的接收数据输出控制位

5.4.8 GPIOB 读写使能控制寄存器（GPIOB_BSR）

偏移地址：0x24
复位值：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

位	标记	功能描述
31:30	Reserved	保留位
29:16	BRx	清除端口 x 的位 y（y = 0…15） 这些位只能写入并只能以字（16 位）的形式操作。 0：对于对应的 ODRy 位不产生影响； 1：清除对应的 ODRy 位为 0。 注：如果同时设置了 BSy 和 BRy 的对应位，BSy 位起作用
15:14	Reserved	保留位
13:0	BSx	设置端口 x 的位 y（y = 0…15） 这些位只能写入并只能以字（16 位）的形式操作。 0：对于对应的 ODRy 位不产生影响； 1：设置对应的 ODRy 位为 1

5.4.9 GPIOB 复用控制寄存器高位（GPIOB_AFRH）

偏移地址：0x28
复位值：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			

位	标记	功能描述
31:24	Reserved	保留位
23:20	AFSEH13[3:0]	对应的 GPIOB13 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
19:16	AFSEH12[3:0]	对应的 GPIOB12 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
15:12	AFSEH11[3:0]	对应的 GPIOB11 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
11:8	AFSEH10[3:0]	对应的 GPIOB10 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3

位	标记	功能描述
7:4	AFSEH9[3:0]	对应的 GPIOB9 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
3:0	AFSEH8[3:0]	对应的 GPIOB8 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3

5.4.10 GPIOB 复用控制寄存器低位（GPIOB_AFRL）

偏移地址：0x2C
复位值：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			

位	标记	功能描述
31:28	AFSEL7[3:0]	对应的 GPIOB7 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
27:24	AFSEL6[3:0]	对应的 GPIOB6 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
23:20	AFSEL5[3:0]	对应的 GPIOB5 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
19:16	AFSEL4[3:0]	对应的 GPIOB4 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
15:12	AFSEL3[3:0]	对应的 GPIOB3 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
11:8	AFSEL2[3:0]	对应的 GPIOB2 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
7:4	AFSEL1[3:0]	对应的 GPIOB1 的复用功能选择寄存器

位	标记	功能描述
		0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3
3:0	AFSEL0[3:0]	对应的 GPIOB0 的复用功能选择寄存器 0000：AF0；　　0001：AF1； 0010：AF2；　　0011：AF3

5.5 GPIOB 寄存器映射

GPIOB 寄存器列表

基地址：0x3000_0240

寄存器	偏移地址	描述
GPIOB_MODER	0x00	GPIOB 模式控制寄存器
GPIOB_OTYPER	0x04	GPIOB 输出控制寄存器
GPIOB_ITYPER	0x08	GPIOB 输入控制寄存器
GPIOB_PUPDR	0x0C	GPIOB 上拉下拉控制寄存器
GPIOB_SDR	0x10	GPIOB 性能控制寄存器
GPIOB_IDR	0x1C	GPIOB 输入数据寄存器
GPIOB_ODR	0x20	GPIOB 输出数据寄存器
GPIOB_BSR	0x24	GPIOB 写使能控制寄存器
GPIOB_AFRH	0x28	GPIOB 复用控制寄存器高位
GPIOB_AFRL	0x2C	GPIOB 复用控制寄存器低位

6 中断

6.1 中断简介

MCU RISC-V 核有一个局部中断控制器 CLIC（Core-Local Interrupt Controller）。CLIC 可以产生机器模式计时器中断（Machine Timer Interrupt）、机器模式软件中断（Machine Software Interrupt）两个标准 RISC-V 架构定义的中断类型。除了这两个标准 RISC-V 架构定义的中断，CLIC 还定义了 32 个中断源作为本地中断/局部中断（Local Interrupt）来处理，用于连接到外部设备，设置相应寄存器即可接受外设的输入信号作为中断。处理器可以通过 CLIC 接收到这 32 个外部设备的中断以及上述两个标准 RISC-V 架构定义的中断。每个中断都有相应的 Interrupt ID 号，上述 32 个中断源 ID 为 16 ~ 47，其中 16 ~3 0，32 ~ 35 与 40 ~ 47 有外设连接，31、39 无外设连接。Interrupt ID 号可参见下面表格。

表 6-1 内核中断及其 Interrupt ID

ID（十进制）	说明
2 - 0	-
3	机器模式软件中断（Machine Software Interrupt）
6 - 4	-
7	机器模式计时器中断（Machine Timer Interrupt）
10 - 8	-
11	机器模式外部中断（Machine External Interrupt）
12	CLIC 软件中断
15 - 13	-
16	SPI1 中断
17	SPI2 中断
18	LV 中断
19	UART1 收发中断（TX/RX）
20	I2C 等待中断
21	I2C 错误中断
22	Timer1 Break 中断
23	Timer1 Updata 中断
24	Timer1 Capture Compare 中断
25	Timer1 Trigger and Commutation 中断
26	Timer2 Break 中断
27	Timer2 Updata 中断
28	Timer2 Capture Compare 中断

ID（十进制）	说明
29	Timer2 Trigger and Commutation 中断
30	ADC 中断
31	-
32	WUP 唤醒中断
33	UART2（TX/RX）中断
34	UART3（TX/RX）中断
35	UART4（TX/RX）中断
36	比较器 1 中断
37	比较器 2 中断
38	比较器 3 中断
39	-
40	EXTI[0]外部中断
41	EXTI[1]外部中断
42	EXTI[2]外部中断
43	EXTI[3]外部中断
44	EXTI[4]外部中断
45	EXTI[9:5]外部中断
46	EXTI[15:10]外部中断
47	RF 检测中断

6.2 CLIC 寄存器

基址：0x0280_0000

6.2.1 CLIC 中断等待寄存器（clicintip）

偏移地址：Interrupt ID（十六进制）
复位值：0x0000_0000

7	6	5	4	3	2	1	0
Reserved							clicintip
							RW

位	标记	功能描述
7:1	Reserved	保留位
0	clicintip	表明相应 Interrupt ID 的中断的等待状态 若置 1，说明当前有相应 Interrupt ID 的中断正在等待

6.2.2 CLIC 中断使能寄存器（clicintie）

偏移地址：0x400 + Interrupt ID（十六进制）
复位值：0x0000_0000

7	6	5	4	3	2	1	0
Reserved							clicintie
							RW

位	标记	功能描述
7:1	Reserved	保留位
0	clicintie	可用于屏蔽与使能相应 Interrupt ID 的中断 写 0 屏蔽中断；写 1 使能中断

6.2.3 CLIC 中断配置寄存器（clicintcfg）

偏移地址：0x800 + Interrupt ID（十六进制）
复位值：0x0000_0000

7	6	5	4	3	2	1	0
clicintcfg			Reserved				
RW							

位	标记	功能描述
7:5	clicintcfg	设置相应 Interrupt ID 中断的抢占级别（pre-emption level）与优先级（priority）。clicintcfg 中总共有两位可以指定如何编码给定中断的抢占级别与优先级。确定抢占级别的实际位数是由 CLIC 配置寄存器 cliccfg 中的 nlbit 位决定的，如果寄存器 cliccfg 中的 nlbits 的值小于 2，则剩余最小有效实现位以给定的抢占级别编码优先级。如果将寄存器 cliccfg 中的 nlbits 的值设置为 0，则所有的中断的级别（level）为 15，并且 2 位都被用于设置优先级。
4:0	Reserved	保留位

6.2.4 CLIC 配置寄存器（cliccfg）

偏移地址：0xC00
复位值：0x0000_0000

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

位	标记	功能描述
7	Reserved	保留位
6:5	nmbits	不可写，读为 0
4:1	nlbits	设置 nlbits 可以确定寄存器 clicintcfg 中用于级别的位数与用于优先级的位数。CLIC 可最多支持 256 个抢占级别，需要 8 位去编码所有 16 个级别
0	nvbits	当 mvbits 被置位，选择硬件向量使能。在 CLIC Direct 模式下，nvbits 允许选定的中断向量化。在 CLIC Direct 模式下，若 nvbits = 1，则启动选择中断向量化。clicintcfg 最小有效实现位（位 5）控制对应中断的向量行为。在 CLIC Direct 模式下，nvbits 与 clicintcfg 的相关位会被置 1，则中断会按照 mtvt CSR 中断向量表所说明的向量化。这允许一些中断跳转到 mtvec CSR 保存的公共基地址，其他中断则被硬件向量化

6.3 CLIC 寄存器映射

CLIC 寄存器列表
基地址：0x0200_0000

寄存器	偏移地址	描述
msip	0x0000	机器模式软件中断等待寄存器
mtimecmp	0x4000	机器模式计时器比较值寄存器
mtime	0xBFF8	机器模式计时器寄存器

CLIC 寄存器列表
基地址：0x0280_0000

寄存器	偏移地址	描述
clicintip	0x000	CLIC 中断等待寄存器
clicintie	0x400	CLIC 中断使能寄存器
clicintcfg	0x800	CLIC 中断配置寄存器
cliccfg	0xC00	CLIC 配置寄存器

6.4 外部中断（EXTI）

6.4.1 EXTI 介绍

外部中断控制器支持 17 个外部中断，每个中断均设有状态位，每个中断都有独立的触发和屏蔽设置。

外部中断第 1 位到第 16 位中断的使能和屏蔽及触发方式的设置可以在 GPIOA_LPMR 和 GPIOA_INTER 中设置。EXTI[15:0]对应 GPIOA15 ~ 0。

第 17 位为 RF 检波模块 RF 检测的中断，需要 GPIOA15 设置成模拟模式，从 PA15 接收检测射频信号，信号强度超过特定值，会产生中断信号，若此中断使能则该中断会传给 MCU。

6.4.2 外部中断输入状态寄存器（EXTI_ISR）

基址：0x3000_02C0
偏移地址：0x00
复位值：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															

位	标记	功能描述
31:17	Reserved	保留位
16	exti_caw_irq	RF 检测中断状态，1：产生 RF 中断，写 1 可清除中断
15:0	exti_isrx	外部中断中断状态 1：产生相应位的外部中断，对相应位写 1 可以清除相应的外部中断 第 0 位寄存 GPIOA0 的中断状态； 第 1 位寄存 GPIOA1 的中断状态； … 第 15 位寄存 GPIOA15 的中断状态；

6.4.3 外部中断输入使能寄存器（EXTI_IEN）

基址：0x3000_02C0
偏移地址：0x04
复位值：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															

位	标记	功能描述
31:17	Reserved	保留位
16	exti_ien	外部 RF 检测中断输入使能，1：使能 RF 检测中断
15:0	Reserved	保留位

6.5 EXTI 寄存器映射

EXTI 寄存器列表
基地址：0x3000_02C0

寄存器	偏移地址	描述
EXTI_ISR	0x00	外部中断输入状态寄存器
EXTI_IEN	0x04	外部中断输入使能寄存器

6.6 中断操作

6.6.1 中断的进入和退出

当产生中断时：

a) mstatus.MIE 的值被复制到 mcause.MPIE，mstatus.MIE 被清零，有效阻隔中断；

b) 在 CLIC 模式中，中断等级被复制到 mcause.MPIL 寄存器；

c) 根据 mstatus.MPP 判断特权模式；

d) 当前 PC 被复制到 mepc 寄存器，之后 pc 变为 mtvec.MODE 设定的值。

此时，如果没有使能中断，将由软件控制。可以通过写 mstatus.MIE 重新使能中断或者执行 MRET 指令退出中断处理。当执行了 MRET 指令后：

a) 特权模式按照 mstatus.MPP 设置；

b) 在 CLIC 模式中，中断等级按照 mcause.MPIL 设置；

c) mstatus.MIE 按照 mcause.MPIE 设置；

d) PC 按照 mepc 设置。

此时，由软件控制。CSR 相关的寄存器在 6.8 章节中描述。

6.6.2 中断等级和优先级

在任何时候，hart 都以具有中断级别的特权模式运行。hart 的当前中断等级可在 mintstatus 寄存器查看。然而，当前特权模式对 hart 上运行的软件是不可见的。

在每个特权模式下，CLIC 架构支持最多 256 个中断等级。其中，较高的中断级可以抢占较低的中断级别。中断等级 0 对应于在中断处理程序之外的常规指令。CLIC 还支持给已定的中断等级编程优先级，用于在同一中断级别上对挂起和启用的中断进行优先排序。在一个给定的中断级别上，优先级最高的中断被优先处理。如果有多个挂起并启用的中断具有相同的最高优先级，ID 号最高的中断被优先处理。

抢占等级（pre-emption levels）和每个等级的优先级数目可以通过 clicintcfg 寄存器和 cliccfg.nlbits 寄存器配置。

6.7 中断控制状态寄存器

6.7.1 机器状态寄存器（mstatus）

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

位	标记	功能描述
12:11	MPP	预特权模式寄存器
10:8	Reserved	保留位
7	MPIE	预中断使能寄存器
6:4	Reserved	保留位
3	MIE	中断使能寄存器，中断的总开关，设置为 0 不会进入中断处理
2:0	Reserved	保留位

通过设置 mstatus.MIE 可以使能中断总开关，通过设置机器中断使能（mie）可以设置每个中断的独立开关。

注意：在 CLIC 模式下，mstatus.MPP 和 mstatus.MPIE 可以通过 mcause 寄存器操作。

6.7.2 机器异常向量寄存器（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

位	标记	功能描述
31:2	BASE	中断向量基址 在 CLINT 直接模式下操作需要 4 字节对齐； 在 CLINT 矢量模式下操作需要 128 字节对齐； 在 CLIC 模式下运行需要至少 64 字节对齐
1:0	MODE	MODE 设置中断进程的模式 0x00：直接（Direct）模式，所有异步中断和同步异常设置 PC 为 BASE； 0x01：向量（Vectored）模式，异常设置 pc 为 BASE，中断设置 pc 为 BASE + 4 × mcause.EXCCODE； 0x02：CLIC 直接模式（CLIC Direct），所有中断和异常设置 PC 为 BASE； 0x03：CLIC 向量模式（CLIC Vectored），异步中断设置 PC 为 mtvt + mcause.EXCCODE × 4； 注：在非 CLIC 模式中，仅支持处理软件、计时器和外部中断

1) 直接（Direct）模式：此模式下，所有同步异常和异步中断使用 mtvec.BASE 地址。在中断处理过程中，软件需要读 mcause 寄存器来确定中断的来源；
2) 向量（Vectored）模式：此模式下，PC 指针被设置为 mtvec.BASE + 4 × exception code。例如，当机器 timer 中断发生后，PC 指针被设置为 mtvec.BASE + 0x1C。一般来说，中断向量表由跳转指令占据，到专门的中断处理处执行。此模式下 BASE 必须是 64 字节对齐；
3) CLIC 直接模式（CLIC Direct）：此模式下，处理器跳转到 mtevc 设置的地址处执行。此模式下 BASE 必须是 64 字节对齐；
4) CLIC 向量模式（CLIC Vectored）：此模式下，处理器转换到特权模式并设置 mcause.MINHV。之后做取址操作，地址为 mtvt + 4 × mcause.EXCCODE。如果取址成功，处理器会清除 handler address 低位，并将 PC 设置为这个 handler address。之后会清除 mcause.MINHV。

同步例外（exception）总是陷入 mtvec.BASE 在机器模式中。

6.7.3 机器模式中断使能寄存器（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			

MIE 寄存器独立设置每个中断是否使能

位	标记	功能描述
31:12	Reserved	保留位
11	MEIE	机器模式软件中断使能
10:8	Reserved	保留位
7	MTIE	机器模式计时器中断使能
6:4	Reserved	保留位
3	MSIE	机器模式外部中断使能
2:0	Reserved	保留位

在 CLIC 模式中，mie 寄存器被硬件置 0，并且独立的中断使能由 clicintip[i] 控制。

6.7.4 机器模式中断等待寄存器（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 RO	Reserved			MTIP RO	Reserved			MSIP RO	Reserved		

MIP 寄存器独立设显示哪个中断在等待

位	标记	功能描述
31:12	Reserved	保留位
11	MEIE	机器模式软件中断等待
10:8	Reserved	保留位
7	MTIE	机器模式计时器中断等待
6:4	Reserved	保留位
3	MSIE	机器模式外部中断等待
2:0	Reserved	保留位

在 CLIC 模式中，mip 寄存器被硬件置 0，并且独立的中断使能由 clicintip[i] 控制。

6.7.5 机器模式异常原因寄存器（mcause）

mcause 寄存器显示中断的来源。当中断产生时，mcause 最高位变为 1，低

位显示中断的 ID。比如机器 timer 中断会使 mcause 被置为 0x8000_0007。mcasue 也被用于显示同步异常来源，此时最高位被置为 0。

在 CLIC 模式，mcasue 显示更多信息。

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	

位	标记	功能描述
31	Interrupt	1：trap 由中断产生　　0：其它
30	MINHV	只支持 CLIC 模式
29:28	MPP	预中断特权模式，和 mstatus.mpp 相同，只支持 CLIC 模式
27	MPIE	预中断使能，和 mstatus.mpie 相同，只支持 CLIC 模式
26:24	Reserved	保留位
23:16	MPIL	预中断等级，只支持 CLIC 模式
15:10	Reserved	保留位
9:0	Exception Code	显示上次异常代码

中断异常代码		
interrupt	异常代码	描述
1	0 – 2	保留
1	3	机器模式软件中断使能
1	4 – 6	保留
1	7	机器模式计时器中断使能
1	8 – 10	保留
1	11	机器模式外部中断使能
1	12	CLIC 软件中断使能
1	13 – 15	保留
1	16	CLIC 本地中断 0
1	17	CLIC 本地中断 1
1	18 – 31	…
1	48	CLIC 本地中断 32
0	0	指令地址未对齐

0	1	指令存取错误
0	2	非法指令
0	3	断点
0	4	装载地址未对齐
0	5	装载错误
0	6	Store/AMO 地址未对齐
0	7	Store/AMO 存取错误
0	8 – 10	保留
0	11	Environment call from M-mode
0	≥ 12	保留

6.7.6 机器模式异常向量表（mtvt）

mtvt 寄存器保存着用于 CLIC 向量中断的机器异常向量基址。mtvt 允许重定位向量表。BASE 必须是 64 字节对齐。

在 CLIC 模式，mcasue 显示更多信息。

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															

位	标记	功能描述
31:6	BASE	CLIC 向量表的基址
5:0	Reserved	保留位

6.7.7 中断入口地址和中断使能（mnxti）

mnxti 可以被软件用来处理下一个等级超过已保存（保存在 mcause.PIL 里）中断，且不需要完整耗费的中断的关断和内容存取。CSRRSI/CSRRCI 指令可以操作 mnxti 寄存器。读这个寄存器返回 mtvt 的地址，如果返回值是 0，说明没有合适的中断需要处理。当写这个寄存器时，mcause 的异常编码寄存器和 mintstatus

的 mil 寄存器将会更新为新的寄存器等级。

在中断处理中，mnxti 寄存器一般在初始化 mcause 和 mepc 寄存器之后使用。

6.7.8 机器模式中断状态寄存器（mintstatus）

Mintstatus 为每个支持的特权模式保存中断级别。

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															

位	标记	功能描述
31:24	MIL	机器模式中断等级
23:0	Reserved	保留位

6.7.9 机器模式 Scratch 寄存器（mscratch）

在 CLIC 模式，mcasue 显示更多信息。

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															

位	标记	功能描述
31:0	mscratch	被用于保存一个指向机器模式硬件线程本地的上下文空间的指针，并在一个 M-mode 自陷处理函数入口处，与一个用户寄存器进行交换

6.7.10 机器异常 PC 寄存器（mepc）

Mepc 寄存器永远不能保存一个导致指令地址非对齐异常的 pc 值。

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															

位	标记	功能描述
31:0	mepc	产生异常的指令地址 pc 值

6.7.11 机器模式异常值寄存器（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															

位	标记	功能描述
31:0	mtval	用于存储发生异常的指令和地址

6.8 中断状态寄存器映射

中断状态寄存器列表

寄存器	偏移地址	描述
mstatus	0x300	机器状态寄存器
mie	0x304	机器中断使能寄存器
mtvec	0x305	机器异常入口基地址寄存器
mtvt	0x307	机器模式异常向量表
mepc	0x341	机器模式异常 PC 寄存器
mcause	0x342	机器模式异常原因寄存器
mtval	0x343	机器模式异常值寄存器
mip	0x344	机器模式中断等待寄存器
mnxti	0x345	中断入口地址和中断使能
mintstatus	0x346	机器模式中断状态寄存器
mscratch	0x340	机器模式 Scratch 寄存器

注：RISC-V 架构中的定义的 CSR 寄存器需要使用特殊的 CSR 指令进行访问，如果需要在 C/C++程序中使用 CSR 寄存器，只能采用内嵌汇编（CSR 指令）的方式，才能对 CSR 寄存器进行操作。以下是在 C 语言中调用 RISC-V 的 CSR 读或者写汇编指令访问 CSR 寄存器的实例，代码如下：

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

定义以上宏，在 C 语言中直接调用此宏即相当于读取指 CSR 寄存器的值，譬如 C 语言“value=read_csr(mstatus)”即相当于读取 mstatus 寄存器的值将其赋值给变量 value 中。

7 实时时钟（RTC）

7.1 RTC 介绍

RTC 通常以固定不变的低速时钟运行，因此可以提供时间参考，所以被称为实时时钟。实时时钟是一个独立的定时器，RTC 模块拥有连续计数的计数器，在相应软件配置下，可提供计时的功能。

MCU 的主要时钟输入为：时钟（clock）、rtc_toggle。时钟用来驱动总线和调试接口。通常，rtc_toggle 连接到实时时钟或作为平台的基本时钟输入频率。时钟输入频率之间的关系如下：clock > (2 × rtc_toggle)。

MCU 有一个计时器，该计时器有两个寄存器，mtime 和 mtimecmp。Mtime 寄存器用于反映当前计时器的计数值，mtimecmp 用于设置计时器比较值。在 MCU 核中，采用 rtc_toggle 输入作为这个实时计数器。rtc_toggle 可以通过在 mtime 寄存器中确定实时计数器计数值来确定。rtc_toggle 用于计时时，频率不需要很高。rtc_toggle 必须严格地以上述公式描述的频率范围运行，即它必须严格地小于时钟频率的一半。此外，rtc_toggle 的频率必须保持不变，并且必须可以通过软件设置来修改这个频率。

当 RTC 时钟来源为 3K 或 RCOSC 时，时钟精度由 3K 和 RCOSC 决定。

7.2 寄存器说明

基址：0x0200_0000

7.2.1 机器模式计时器寄存器（mtime）

偏移地址：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															

位	标记	功能描述
31:0	mtime_lo	反映当前计时器的低 32 位计数值

偏移地址：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															

位	标记	功能描述
31:0	mtime_hi	反映当前计时器的高 32 位计数值

7.2.2 机器模式计时器比较值寄存器（mtimecmp）

偏移地址：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	标记	功能描述
31:0	mtimecmp_lo	配置计时器的比较值低 32 位 当 mtime 中的计数值大于或者等于 mtimecmp 中设置的比较值时，计时器便会产生计时器中断。计时器中断会一直拉高，直到软件重新写 mtimecmp 寄存器的值，使得其比较值大于 mtime 中的值，从而将计时器中断清除。 注：当 mie 寄存器中的 MTIE 被设置才会使能此中断。

偏移地址：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	标记	功能描述
31:0	mtimecmp_hi	配置计时器的比较值高 32 位

7.3 寄存器映射

RTC 寄存器列表

基地址：0x0200_0000

寄存器	偏移地址	描述
mtime	0xBFF8	机器模式计时器寄存器
mtimecmp	0x4000	机器模式计时器比较值寄存器

8 看门狗

8.1 独立看门狗

8.1.1 简介

独立的看门狗（IWDG）是由低速时钟 3K 来驱动的，因此即使主时钟出现故障，它也会保持活动状态。独立看门狗最适合应用于需要看门狗独立于主程序之外，能够完全独立工作，并且对时间精度要求很低的场合。看门狗可以上电启动或用户程序启动，上电启动看门狗需要在下载程序时设置。

8.1.1.1 主要特征

- 一旦开启看门狗，计数器会一直运行；
- 在独立看门狗工作条件下，计数器计到 0x00000 时，产生复位信号；
- 在所有低功耗模式下，独立看门狗的计数器仍会工作，计数器计到 0x00000 时，会产生复位信号；
- 可通过写 IWDG 寄存器或者 FLASH 下载程序时设置两种方式打开 IWDG。

8.1.1.2 功能描述

在键值寄存器（IWDG_KR）中写入 0xCCCC，开始启用独立看门狗。此时计数器开始从其复位值 0x3FFFF 递减计数。当计数器计数到尾值 0x00000 时，会产生一个复位信号（IWDG_RESET）。

无论何时，只要在键值寄存器（IWDG_KR）中写入 0xAAAA，自动重装载寄存器（IWDG_RLR）中的值就会被重新加载到计数器，从而避免产生看门狗复位。

如果主程序异常，无法正确喂狗，会导致系统复位。

8.1.1.3 寄存器访问保护

IWDG_RLR 寄存器具有写保护功能。要修改这个寄存器的值，必须先向 IWDG_KR 寄存器中写入 0x5555，然后等待一个指令周期。以不同的值写入这个寄存器将会打乱操作顺序，寄存器将重新被保护。重装载操作（即写入 0xAAAA）也会启动写保护功能。

状态寄存器指示递减计数器是否正在被更新。

8.1.2 IWDG 寄存器描述

基址：0x3000_02A0

8.1.2.1 键值寄存器（IWDG_KR）

偏移地址：0x00
复位值：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															

位	标记	功能描述
31:16	Reserved	保留位
15:0	key	键值（只写寄存器，读出值为 0x0000） 1.软件必须以一定的间隔写入 0xAAAA，否则，当计数器为 0 时，看门狗会产生复位； 2.写入 0x5555 之后等待一个指令周期，然后可以写 IWDG_RLR 寄存器； 3.写入 0xCCCC 启动看门狗工作

8.1.2.2 重装载寄存器（IWDG_RLR）

偏移地址：0x04
复位值：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															

位	标记	功能描述
31:18	Reserved	保留位
17:0	RL	看门狗计数器重装载值（Watchdog counter reload value） 1.具有写保护功能； 2.用于定义看门狗计数器的重装载值，每当向 IWDG_KR 寄存器写入 0xAAAA 时，重装载值会被传送到计数器中，随后计数器从这个值开始递减计数； 3.只有当 IWDG_SR 寄存器中的 RVU 位为 0 时，才能对此寄存器进行修改，读出的值才有效。

8.1.2.3 状态寄存器（IWDG_SR）

偏移地址：0x08
复位值：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

位	标记	功能描述
31:2	Reserved	保留位
1	count_ens	IWDG 计数器工作状态位，1：IWDG 计数器正在计数中
0	RVU	看门狗计数器重装载值更新（Watchdog counter reload value update） 1.此位由硬件置 1 来表示重装载值的更新正在进行中； 2.当看门狗计数器重装载值更新完成后，此位会被硬件清零； 3.重装载值只有在 RVU 位被清零后才可被更改。

8.1.3 IWDG 寄存器映射

寄存器列表
基地址：0x3000_02A0

寄存器	偏移地址	描述
IWDG_KR	0x00	IWDG 键值寄存器
IWDG_RLR	0x04	IWDG 重装载寄存器
IWDG_SR	0x08	IWDG 状态寄存器

9 高级定时器（TIMER1&TIMER2）

9.1 简介

CSM32RV20 拥有两个高级定时器 TIMER1 和 TIMER2。二者的功能相同并且是完全同步的，可以同步操作。

高级控制定时器（TIMERx）由一个 16 位的自动装载计数器组成，它由一个可编程预分频器驱动。

它适合多种用途，包含测量输入信号的脉冲宽度（输入捕获），或者产生输出波形（输出比较，PWM，嵌入死区时间的互补 PWM 等）。使用定时器预分频器和外设时钟分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。

9.2 主要特性

TIMERx 定时器的功能包括：

- 16 位向上，向下，向上/向下自动装载计数器；
- 16 位可编程预分频器，计数器时钟频率的分频系数为 1 ~ 65535 之间的任意数值；
- 4 个独立通道：
  - 输入捕获；
  - 输出比较；
  - PWM 生成（边缘或中间对齐模式）；
  - 单脉冲模式输出；
  - 死区时间可编程的互补输出。
- 在指定数目的计数器周期之后更新定时器寄存器；
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态；
- 如下列事件发生，则产生中断：
  - 更新：计数器向上溢出/向下溢出，计数器初始化（通过软件或者内部/外部触发）；
  - 触发事件（计数器启动，停止，初始化或者由内部/外部触发计数）；
  - 输入捕获；
  - 输出比较；
  - 刹车信号输入。

9.3 框图

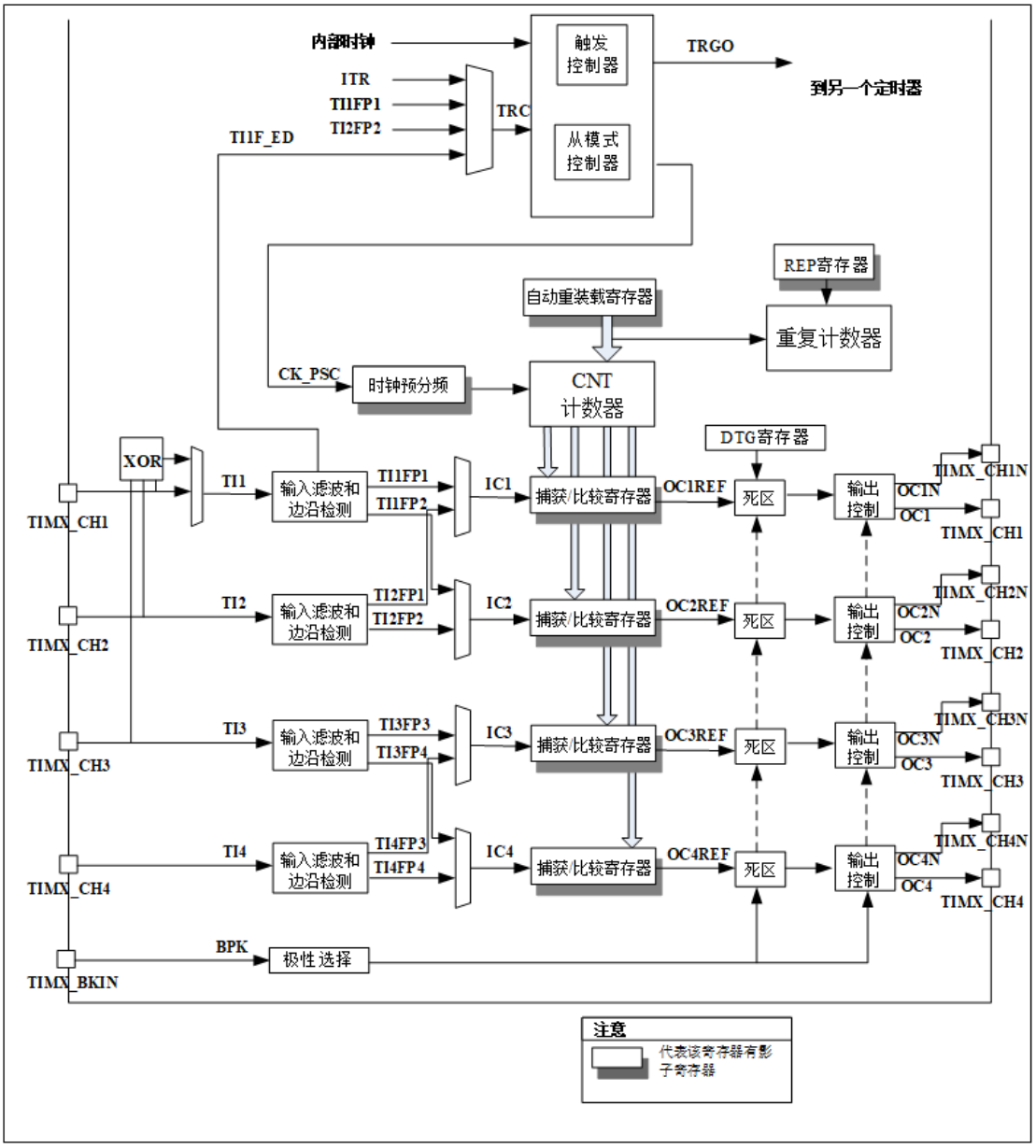


图 9-1 高级定时器框图

9.4 功能描述

9.4.1 时基单元

可编程高级控制定时器的主要部分是一个 16 位计数器和与其相关的自动装

载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行，读写仍然有效。

时基单元包含：

- 计数器寄存器（TIMERx_CNT）；
- 预分频器寄存器（TIMERx_PSC）；
- 自动装载寄存器（TIMERx_ARR）；
- 周期计数寄存器（TIMERx_RCR）。

自动装载寄存器是预先装载的。写或读自动重装载寄存器将访问预装载寄存器。根据在 TIMERx_CR1 寄存器中的自动预装载使能位（ARPE）的设置，预装载寄存器的内容被永久地或在每次产生 UEV 更新事件时传送到影子寄存器。当计数器达到溢出条件（向下计数时的下溢条件）并当 TIMERx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生，随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIMERx_CR1 寄存器中的计数器使能位（CEN）时，CK_CNT 才有效（有关更多的计数器使能的细节，请参见控制器的从模式描述）。

注：真正的计数器使能信号 CNT_EN 是在 CEN 的一个时钟周期后被设置。

预分频器描述

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个（在 TIMERx_PSC 寄存器中的）16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在工作时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

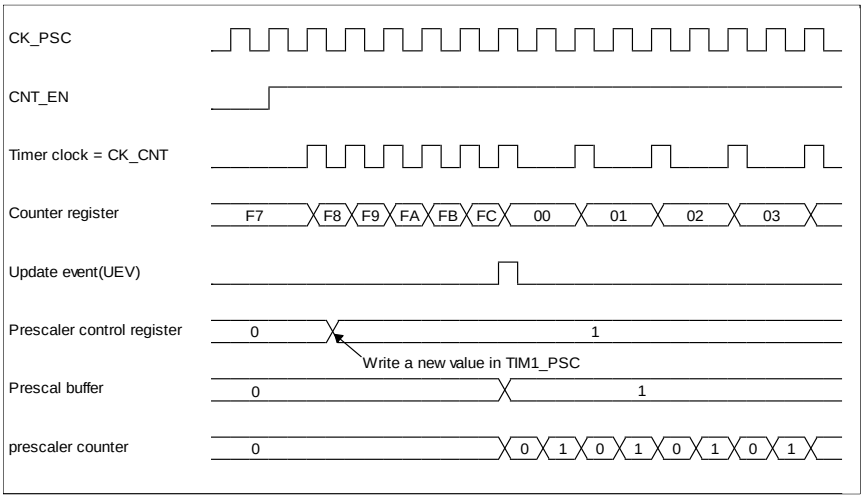


图 9-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

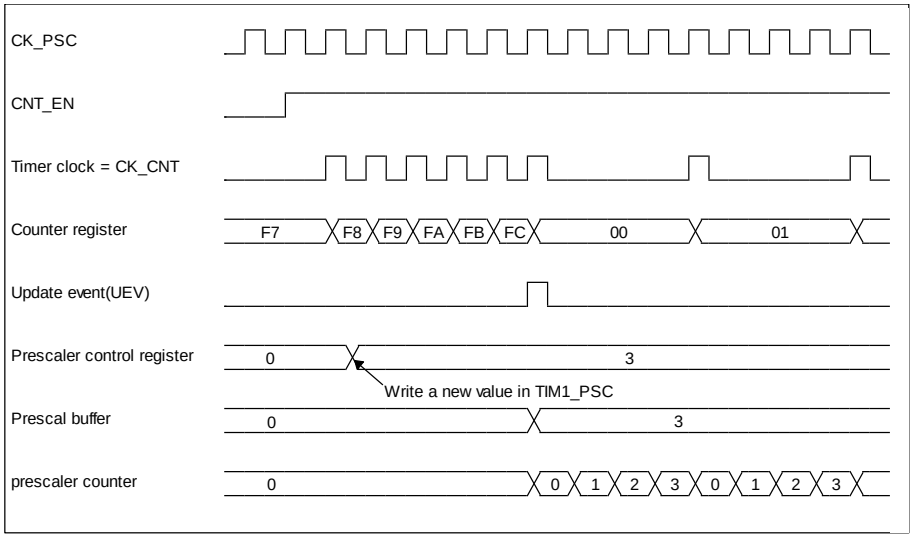


图 9-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

9.4.2 计数器模式

向上计数模式

在向上计数模式中，计数器从 0 计数到自动加载值（TIMERx_ARR 计数器的内容），然后重新从 0 开始计数并且产生一个计数器溢出事件。

如果使用了周期计数功能，在向上计数达到设置的周期计数次数（TIMERx_RCR）时，将产生更新事件（UEV）；否则每次直到计数器溢出才会产生更新事件。

在 TIMERx_EGR 寄存器中设置 UG 位（通过软件方式或者使用从模式控制器）也同样可以产生一个更新事件。

通过软件置 TIM1_CR1 寄存器中的 UDIS 位为 1，将选择禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清成 0 之前，将没有更新事件产生。即使这样，在应该产生更新事件时，计数器仍会被清 0，同时预分频器的计数也被清 0（但预分频器的数值不变）。此外，如果 TIMERx_CR1 寄存器中的 URS 位（更新请求选择）被置 0，设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志（即不产生中断）。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件，所有的寄存器都被更新，硬件同时（依据 URS 位）

设置更新标志位（TIMERx_SR 寄存器中的 UIF 位）。

- 周期计数器被重新加载为 TIMERx_RCR 寄存器的内容；
- 自动装载影子寄存器被重新置入预装载寄存器的值（TIMERx_ARR）；
- 预分频器的缓冲区被置入预分频器寄存器的值（TIMERx_PSC 寄存器的内容）。

下图给出一些例子，当 TIMER1_ARR = 0x36 时计数器在不同时钟频率下的动作。

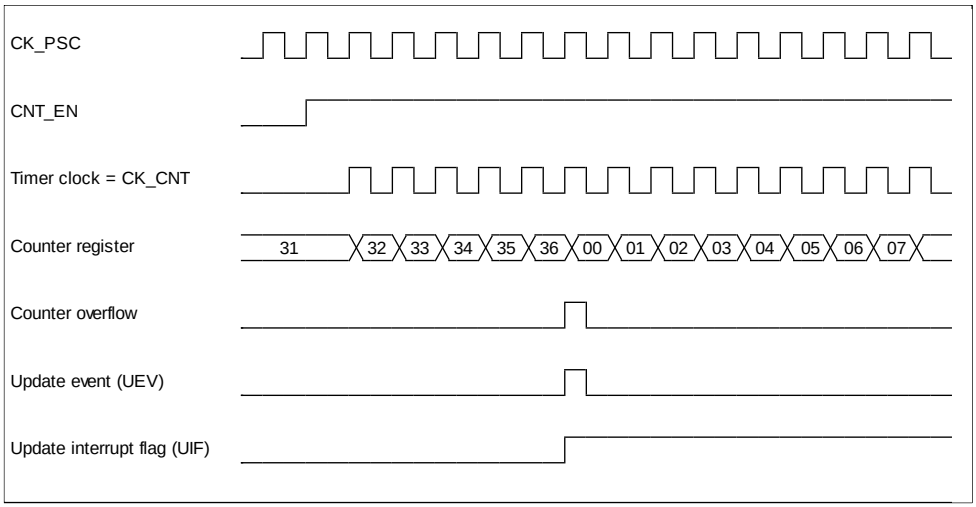


图 9-4 计数器时序图，内部时钟分频因子为 1

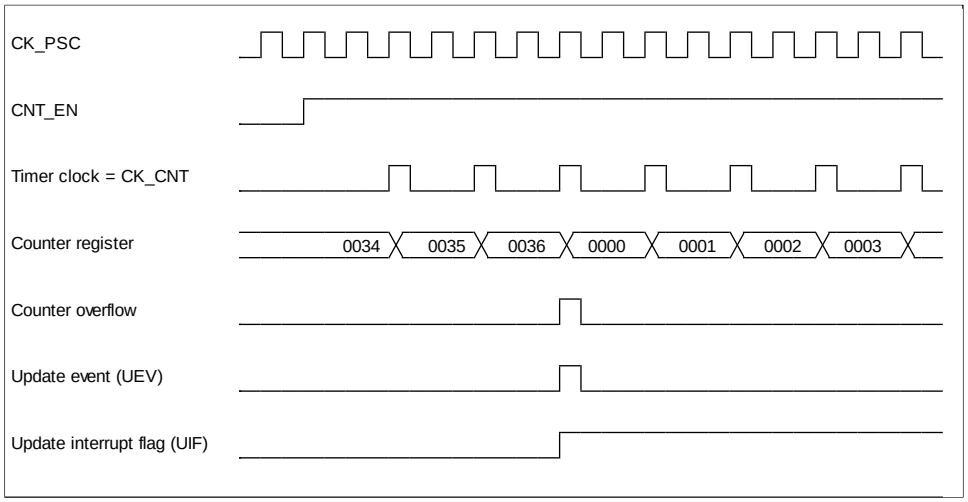


图 9-5 计数器时序图，内部时钟分频因子为 2

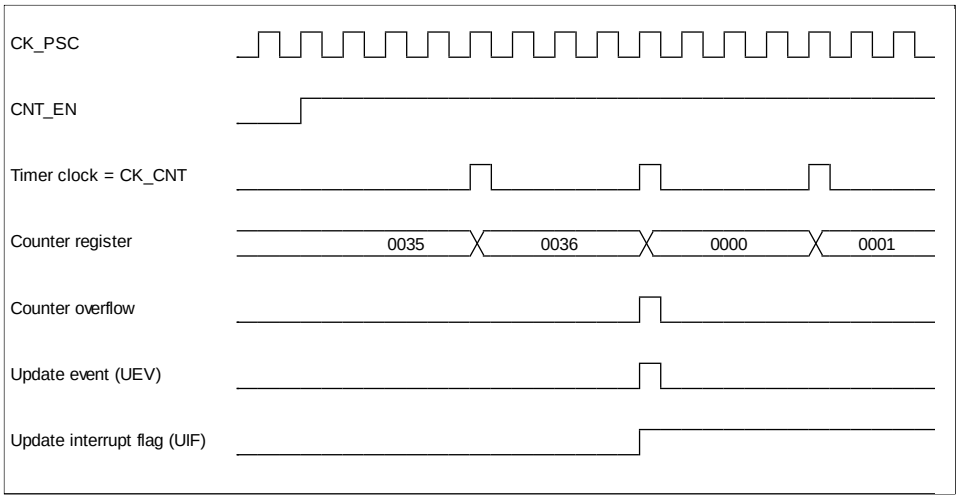


图 9-6 计数器时序图，内部时钟分频因子为 4

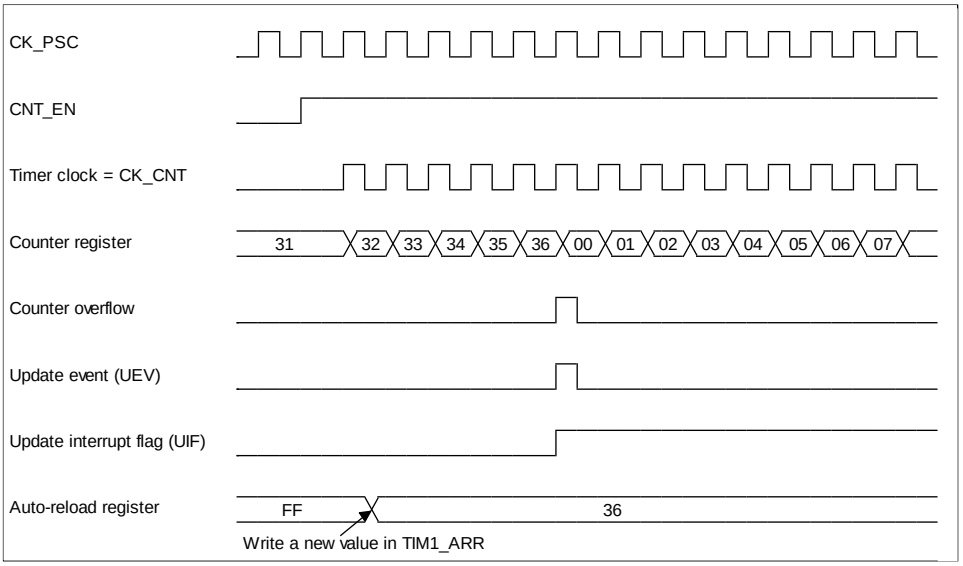


图 9-7 计数器时序图，当 ARPE = 0 时的更新事件（TIMER1_ARR 没有预装入）

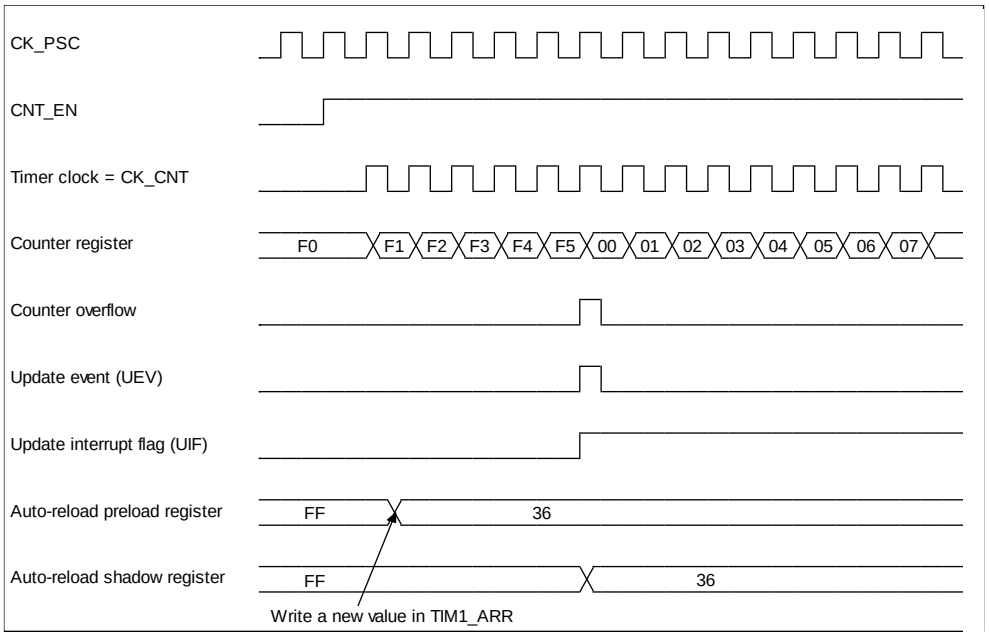


图 9-8 计数器时序图，当 ARPE = 1 时的更新事件（预装入了 TIMER1_ARR）

向下计数模式

在向下模式中，计数器从自动装入的值（TIMERx_ARR 计数器的值）开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数器向下溢出事件。

如果使用周期计数器，当向下计数重复了周期计数寄存器（TIMERx_RCR）中设定的次数后，将产生更新事件（UEV），否则每次计数器下溢时就会产生更新事件。在 TIMERx_EGR 寄存器中设置 UG 位（通过软件方式或者使用从模式控制器）也同样可以产生一个更新事件。UEV 事件可以通过软件设置 TIMERx_CR1 寄存器中的 UDIS 位被禁止。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。这样 UDIS 位被写成 0 之前不会产生更新事件。然而，计数器仍会从当前自动加载值重新开始计数，并且预分频器的计数器重新从 0 开始（但预分频器的速率不能被修改）。此外，如果设置了 TIMERx_CR1 寄存器中的 URS 位（更新请求选择）为 0，设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。当发生更新事件时，所有的寄存器都被更新，并且（根据 URS 位的设置）更新标志位（TIM1_SR 寄存器中的 UIF 位）也被设置。

- 周期计数器被重置为 TIMERx_RCR 寄存器中的内容；
- 当前的自动加载寄存器被更新为预装载值（TIMERx_ARR 寄存器中的内容）。

注：自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。

以下的图显示一些当 TIMER1_ARR = 0x36 时计数器在不同时钟频率下的操作例子。

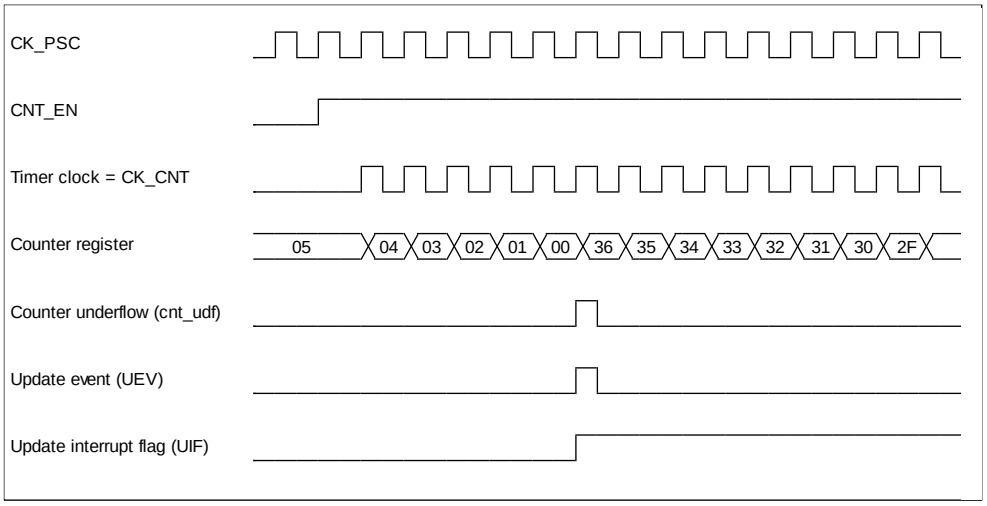


图 9-9 计数器时序图，内部时钟分频因子为 1

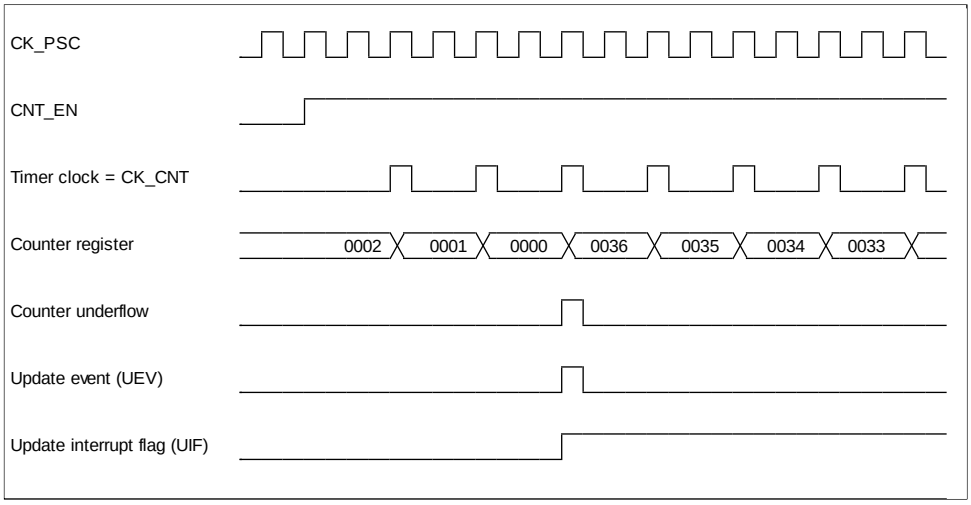


图 9-10 计数器时序图，内部时钟分频因子为 2

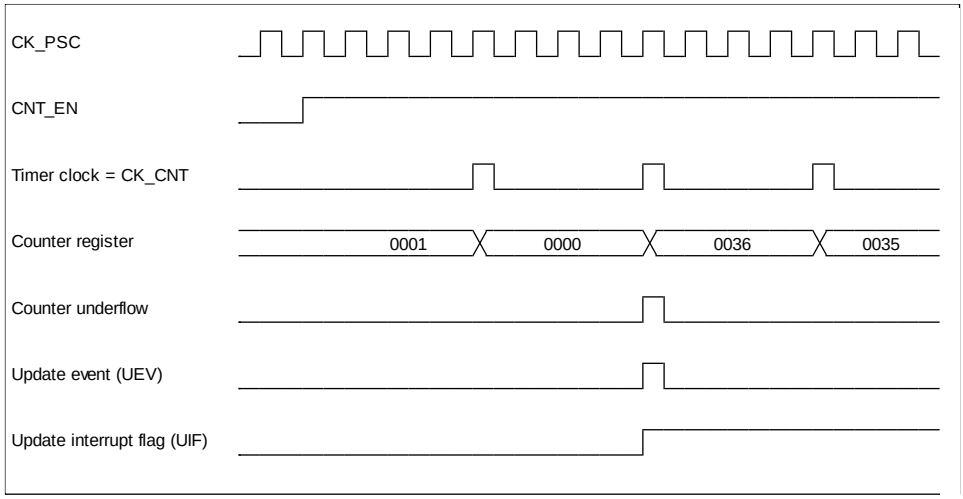


图 9-11 计数器时序图，内部时钟分频因子为 4

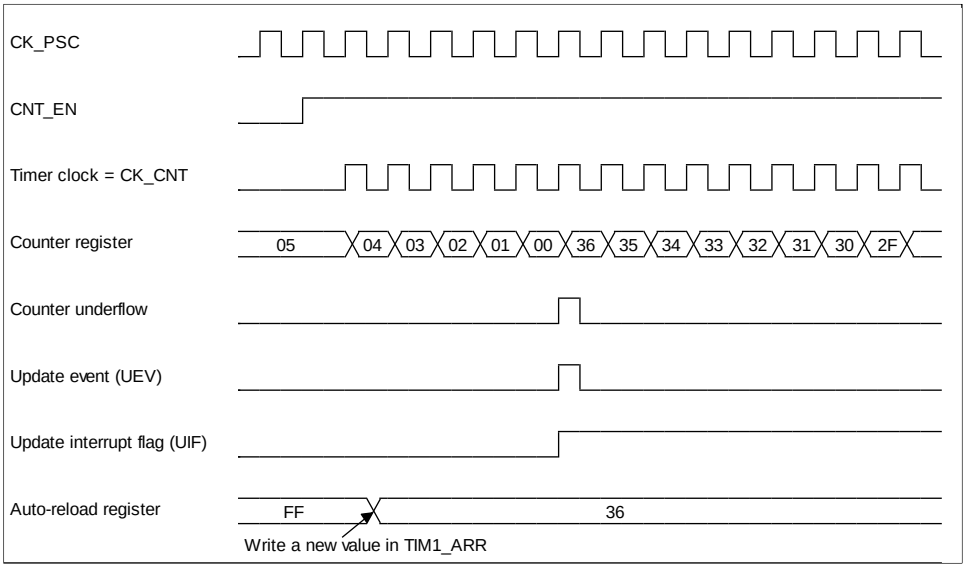


图 9-12 计数器时序图，当 ARPE = 0 时的更新事件（TIMER1_ARR 没有预装入）

中央对齐模式（向上/向下计数）

在中央对齐模式中，计数器从 0 开始计数到自动加载的值（TIMERx_ARR 寄存器的内容）- 1，产生一个计数器溢出事件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。在这个模式下，不能写入 TIMERx_CR1 中的 DIR 方向位。

更新事件可以产生在每一次计数上溢和每一次计数下溢，也可以通过（软件或者使用从模式控制器）设置 TIMERx_EGR 寄存器中的 UG 位来产生更新事件。

此时，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。UEV 事件可以通过软件设置 TIMERx_CR1 寄存器中的 UDIS 位被禁止。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器，这样 UDIS 位被写成 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。

此外，如果设置了 TIMERx_CR1 寄存器中的 URS 位（更新请求选择），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且（根据 URS 位的设置）更新标志位（TIMERx_SR 寄存器中的 UIF 位）也被设置。

- 周期计数器被重置为 TIMERx_RCR 寄存器中的内容；
- 当前的自动加载寄存器被更新为预装载值（TIMERx_ARR 寄存器中的内容）。

注：如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前被更新，因此下一个周期将是预期的值（计数器被装载为新的值）。

以下的图显示一些计数器在不同时钟频率下的操作的例子：

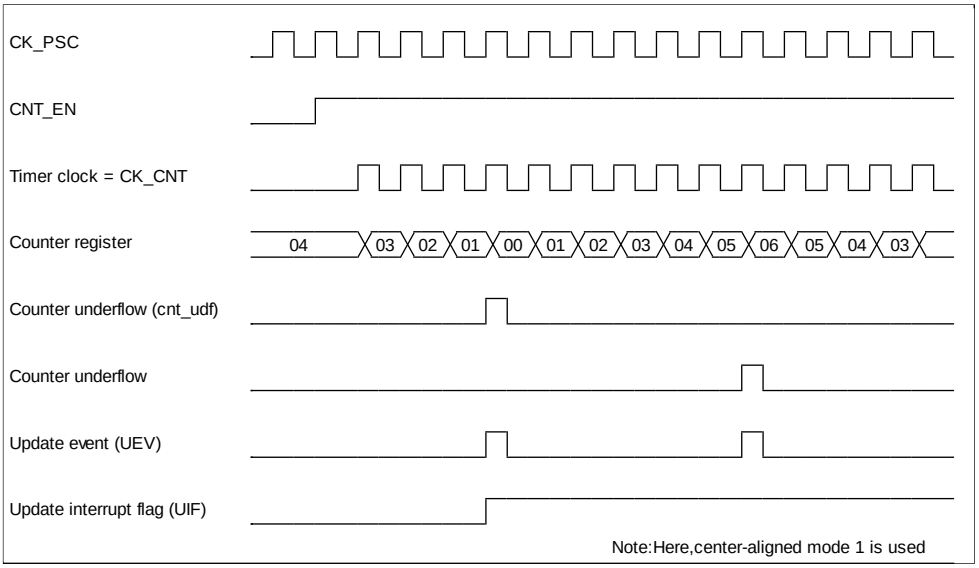


图 9-13 计数器时序图，内部时钟分频因子为 1，TIMER1_ARR = 0x06

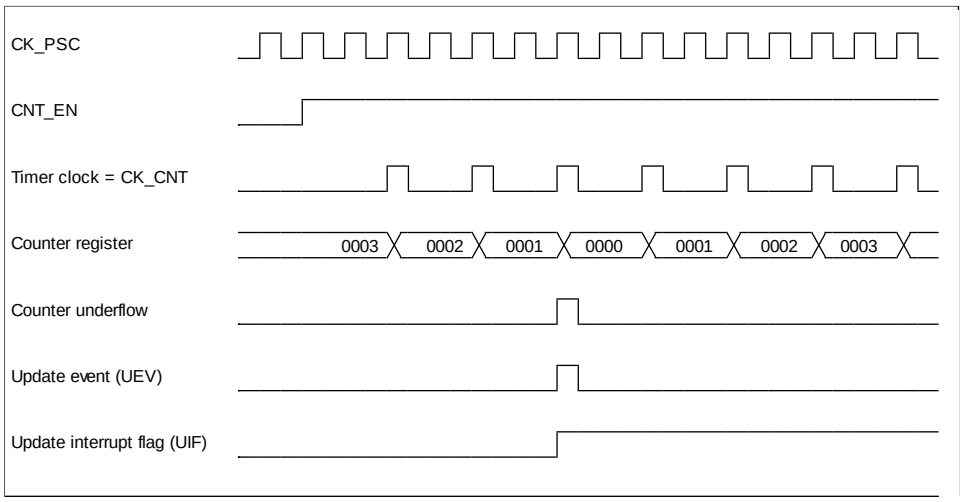


图 9-14 计数器时序图，内部时钟分频因子为 2，TIMER1_ARR = 0x06

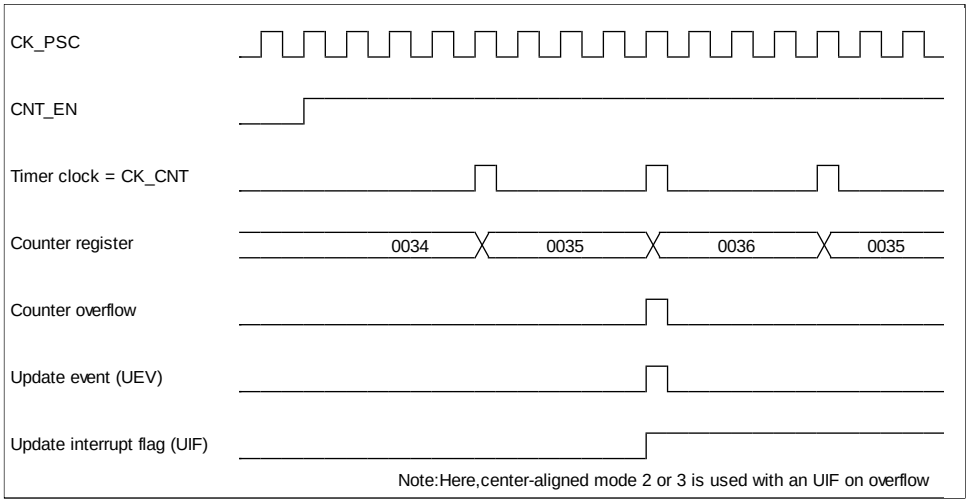


图 9-15 计数器时序图，内部时钟分频因子为 4，TIMER1_ARR = 0x36

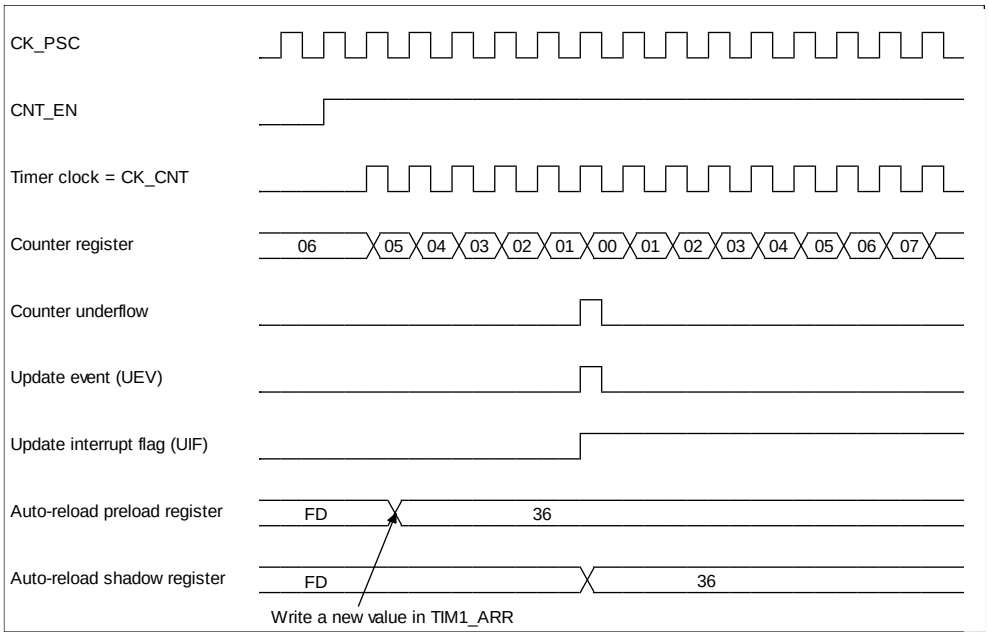


图 9-16 计数器时序图，ARPE = 1 时的更新事件（计数器下溢）

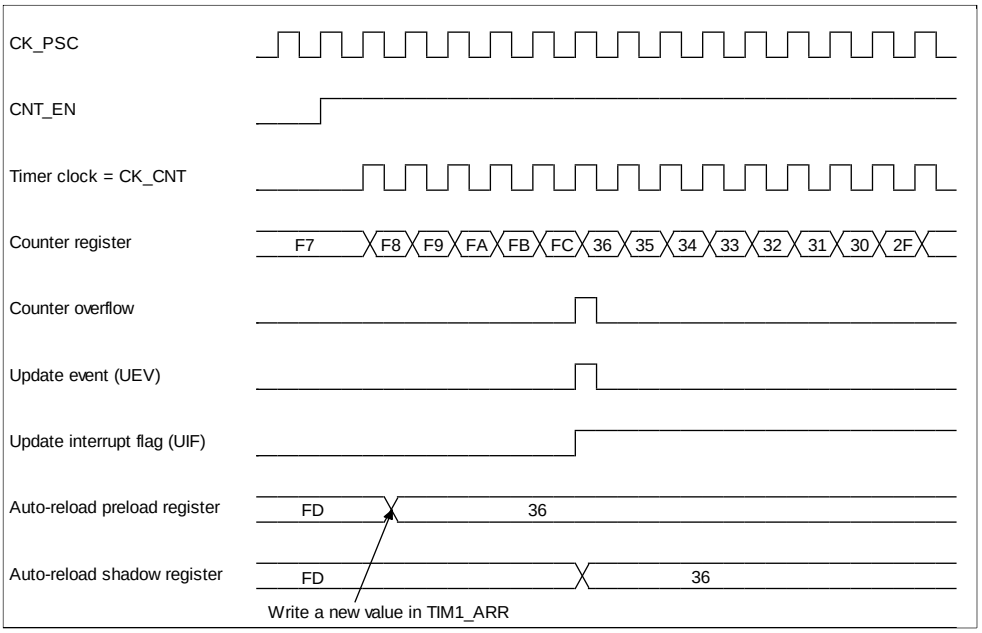


图 9-17 计数器时序图，ARPE = 1 时的更新事件（计数器溢出）

9.4.3 重复向下计数器

章节 9.4.1 时基单元解释了计数器上溢/下溢时更新事件（UEV）是如何产生的，然而事实上更新事件只能在重复计数器达到 0 的时候才会产生。这对于产生

PWM 信号非常有用。

这意味着在每发生 N + 1 次计数上溢或下溢时，数据从预装载寄存器传输到影子寄存器（TIMERx_ARR 自动重载入寄存器，TIMERx_PSC 预装载寄存器，还有在比较模式下的捕获/比较寄存器），N 是 TIMERx_RCR 周期计数寄存器中的值。

重复向下计数器在下述任一条件成立时递减：

- 向上计数模式下每次计数器上溢；
- 向下计数模式下每次计数器下溢；
- 中央对齐模式下每次计数器上溢和下溢。

虽然这样将 PWM 的最大循环周期限制为 128，但它能够在每个 PWM 周期更新 2 次占空比。在中央对齐模式下，因为波形是对称的，如果每个 PWM 周期中仅刷新一次比较寄存器，则最大的分辨率为 $2\times T_{ck}$。

重复向下计数器是自动加载的，重复速率由 TIMERx_RCR 寄存器的值定义。当更新事件由软件产生（通过设置 TIMERx_EGR 中的 UG 位）或者通过硬件的从模式控制器产生时，无论重复向下计数器的值是多少，都会立即发生更新事件，并且 TIMERx_RCR 寄存器中的内容被重载入到重复向下计数器中。

9.4.4 时钟选择

计数器时钟可由下列时钟源提供：

- 内部时钟（CK_INT）；
- 外部时钟模式：外部输入脚。

内部时钟源（CK_INT）

如果从模式控制器被禁止（SMS = 000），则 CEN、DIR（TIMERx_CR1 寄存器中）和 UG 位（TIMERx_EGR 寄存器中）成为实际上的控制位并且只能被软件修改（UG 位除外，该位仍会被自动清除）。一旦 CEN 位被写成 1，预分频器的时钟就由内部时钟 CK_INT 提供。

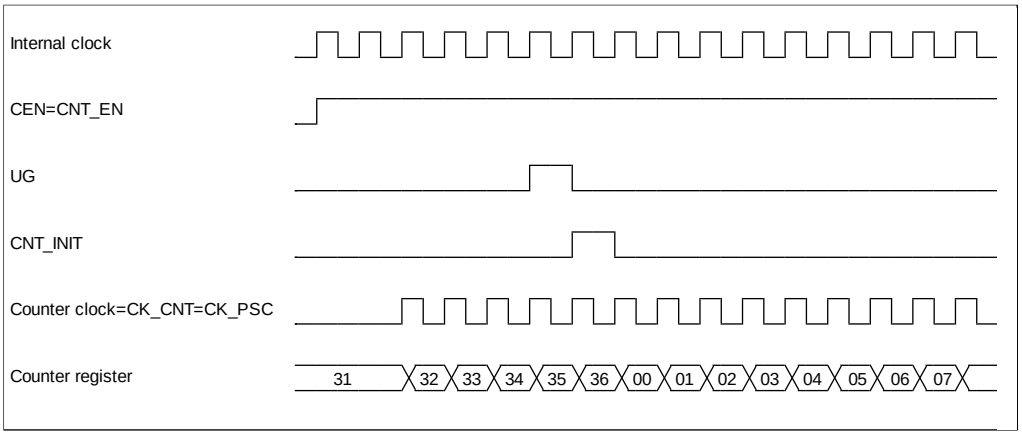


图 9-18 一般模式下的控制电路，内部时钟分频因子为 1

外部时钟源模式

当 TIMERx_CR1 寄存器中的 SMS = 111 时，此模式被选中。计数器可以在选定输入的上每个上升沿或下降沿计数。

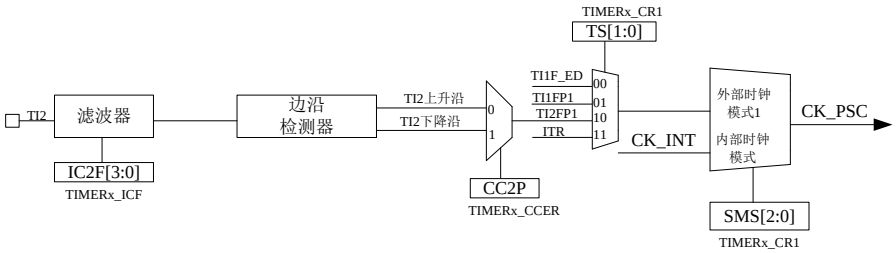


图 9-19 TI2 外部时钟连接例子

例如，要配置向上计数器在 T12 输入端的上升沿计数，使用下列步骤：

1. 配置 TIMERx_CCMR1 寄存器 CC2S = 01，配置通道 2 检测 TI2 输入的上升沿；
2. 配置 TIMERx_ICF 寄存器的 IC2F[3:0]，选择输入滤波器带宽（如果不需要滤波器，保持 IC2F = 0000）；
3. 配置 TIMERx_CCER 寄存器的 CC2P = 0，选定上升沿极性；
4. 配置 TIMERx_CR1 寄存器的 SMS = 111，选择定时器外部时钟模式 1；
5. 配置 TIMERx_CR1 寄存器中的 TS = 10，选定 TI2 作为触发输入源；
6. 设置 TIMERx_CR1 寄存器的 CEN = 1，启动计数器。

当 TI2 上出现上升沿，计数器计数一次，且 TIF 标志被设置。在 TI2 的上升沿和计数器实际时钟之间的延时取决于在 TI2 输入端的重新同步电路。

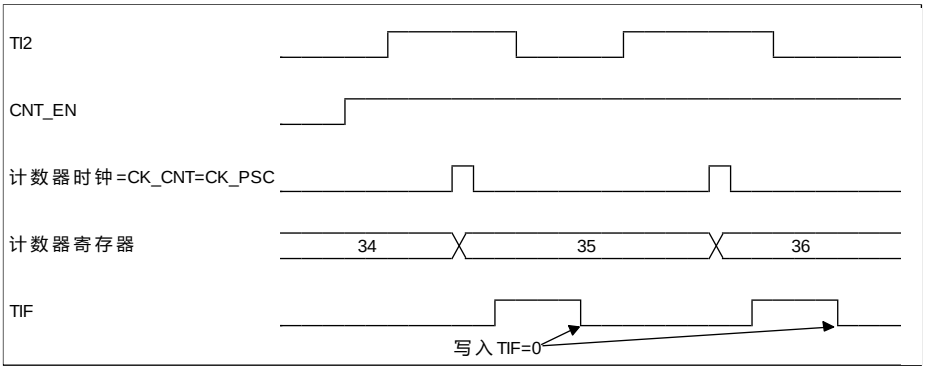


图 9-20 外部时钟模式 1 下的控制电路

9.4.5 捕获/比较通道

每一个捕获/比较通道是围绕着一个捕获/比较寄存器（包含影子寄存器）构成的，包括捕获的输入部分（包含数字滤波和多路复用）和输出部分（包含比较器和输出控制）。

输入部分对相应的 TIx 输入信号采样，并产生一个滤波后的信号 TIxF。然后，一个带极性选择的边缘监测器产生一个信号（TIxFPx），它可以作为从模式控制器的输入触发或者作为捕获控制。该信号先进行预分频（ICxPS）再进入到捕获寄存器。

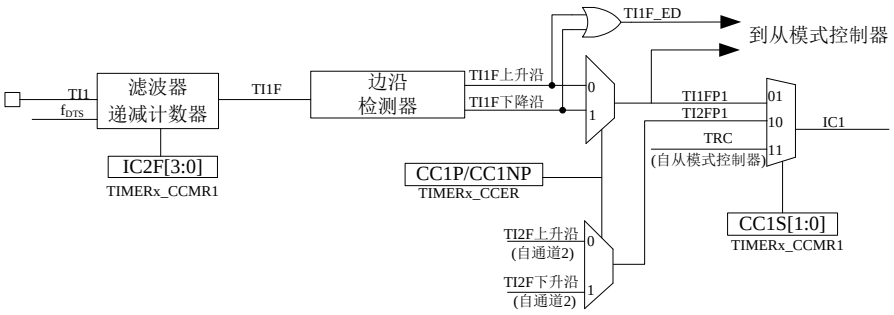


图 9-21 捕获/比较通道（通道 1 输入部分）

输出部分产生一个中间波形 OCxRef（高有效）作为基准，链的末端决定最终输出信号的极性。

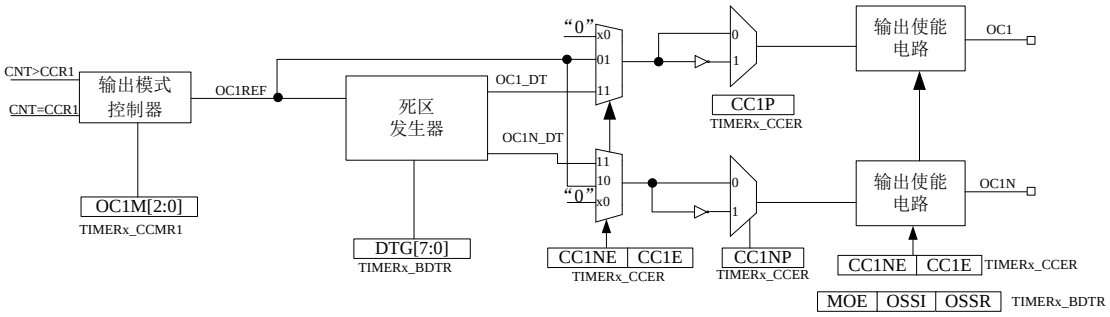


图 9-22 捕获/比较通道的输出部分

9.4.6 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器（TIMERx_CCRx）中。当发生捕获事件时，相应的 CCxIF 标志（TIMERx_SR 寄存器）被置 1，如果开放了中断操作，则将产生中断请求。如果发生捕获事件时 CCxIF 标志已经为高，那么重复捕获标志 CCxOF（TIMERx_SR 寄存器）被置 1。写 CCxIF = 0 可清除 CCxIF，或读取存储在 TIMERx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF = 0 可清除 CCxOF。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIMERx_CCR1 寄存器中，步骤如下：

- 选择有效输入端：TIMERx_CCMR1 必须连接到 TI1 输入，所以写入 TIMERx_CCMR1 寄存器中的 CC1S = 01，只要 CC1S 不为 00，通道就会被配置为输入，并且 TIMERx_CCMR1 寄存器变为只读；
- 根据输入信号的特点，配置输入滤波器为所需的带宽（即输入为 TIx 时，输入滤波器控制位是 TIMERx_ICF 寄存器中的 ICxF 位）。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以（以 fDTS 频率）连续采样 8 次，以确认在 TI1 上一次进行了真实的边沿变换，即在 TIMERx_ICF 寄存器中写入 IC1F = 0011；
- 选择 TI1 通道的有效转换边沿，在 TIMERx_CCER 寄存器中写入 CC1P = 0（上升沿）；
- 设置 TIMERx_CCER 寄存器的 CC1E = 1，允许捕获计数器的值被传送到捕获寄存器中；
- 如果需要，通过设置 TIMERx_DIER 寄存器中的 CC1IE 位允许相关中断请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIMERx_CCR1 寄存器；
- CC1IF 标志被设置（中断标志），当发生至少 2 个连续的捕获，且 CC1IF 未曾被清除时，CC1OF 也被置 1；
- 如设置了 CC1IE 位，则会产生一个中断。

为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

设置 TIMERx_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断请求。

9.4.7 PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

- 两个 ICx 信号被映射至同一个 TIx 输入；
- 这两个 ICx 信号为边沿有效，但是极性相反；
- 其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。例如，你需要测量输入到 TI1 上的 PWM 信号的长度（TIMERx_CCR1 寄存器）和占空比（TIMERx_CCR2 寄存器），具体步骤如下：

— 选择 TIMERx_CCR1 的有效输入：置 TIMERx_CCMR1 寄存器的 CC1S = 01（选中 TI1）；

— 选择 TI1FP1 的有效极性（用来捕获数据到 TIMERx_CCR1 中和清除计数器）：置 CC1P = 0（上升沿有效）；

— 选择 TIMERx_CCR2 的有效输入：置 TIMERx_CCMR1 寄存器的 CC2S = 10（选中 TI1）；

— 选择 TI1FP2 的有效极性（捕获数据到 TIMERx_CCR2）：置 CC2P = 1（下降沿有效）；

— 选择有效的触发输入信号：置 TIMERx_CR1 寄存器中的 TS = 2’b01（选择 TI1FP1）；

— 配置从模式控制器为复位模式：置 TIMERx_CR1 中的 SMS = 100；

— 使能捕获：置 TIMERx_CCER 寄存器中 CC1E = 1 且 CC2E = 1。

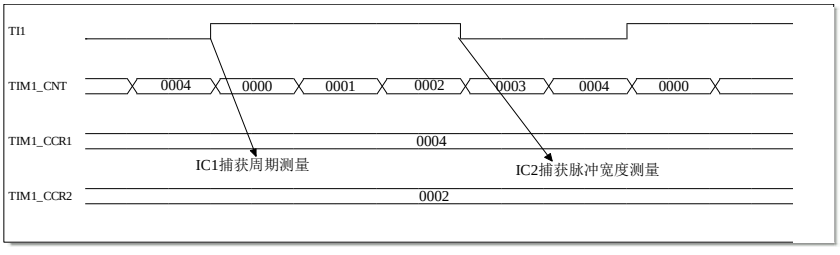


图 9-23 PWM 输入模式时序

因为只有 TI1FP1 和 TI2FP2 连到了从模式控制器，所以 PWM 输入模式只能使用 TIMERx_CH1/TIMERx_CH2 信号。

9.4.8 强制输出模式

在输出模式（TIMERx_CCMRx 寄存器中 CCxS = 00）下，输出比较信号（OCxREF 和相应的 OCx/OCxN）能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。

置 TIMERx_CCMRx 寄存器中相应的 OCxM = 101，即可强置输出比较信号（OCxREF/OCx）为有效状态。这样 OCxREF 被强置为高电平（OCxREF 始终为高电平有效），同时 OCx 得到 CCxP 极性相反的信号。

例如：CCxP = 0（OCx 高电平有效），则 OCx 被强置为高电平。置 TIMERx_CCMRx 寄存器中的 OCxM = 100，可强置 OCxREF 信号为低。

该模式下，在 TIMERx_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断请求。这将会在下面的输出比较模式一节中介绍。

9.4.9 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的时间已经到时。

当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

- 将输出比较模式（TIMERx_CCMRx 寄存器中的 OCxM 位）和输出极性（TIMERx_CCER 寄存器中的 CCxP 位）定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平（OCxM = 000）、被设置成有效电平（OCxM = 001）、被设置成无效电平（OCxM = 010）或进行翻转（OCxM = 011）；
- 设置中断状态寄存器中的标志位（TIMERx_SR 寄存器中的 CCxIF 位）；
- 若使能了相应的中断位（TIMERx_DIER 寄存器中的 CCxIE 位），则产生一个中断。

TIMERx_CCMRx 中的 OCxPE 位选择 TIMERx_CCRx 寄存器是否需要使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。同步的精度可以达到计数器的一个计数周期。输出比较模式（在单脉冲模式下）也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1. 选择计数器时钟（内部，外部，预分频器）；
2. 将相应的数据写入 TIMERx_ARR 和 TIMERx_CCRx 寄存器中；

3. 如果要产生一个中断请求，设置 CCxIE 位；

4. 选择输出模式，例如：

— 要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM = 011；

— 置 OCxPE = 0 禁用预装载寄存器；

— 置 CCxP = 0 选择极性为高电平有效；

— 置 CCxE = 1 使能输出。

5. 设置 TIMERx_CR1 寄存器的 CEN 位启动计数器 TIMERx_CCRx 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器（OCxPE = 0，否则 TIMERx_CCRx 的影子寄存器只能在发生下一次更新事件时被更新）。下图给出了一个例子。

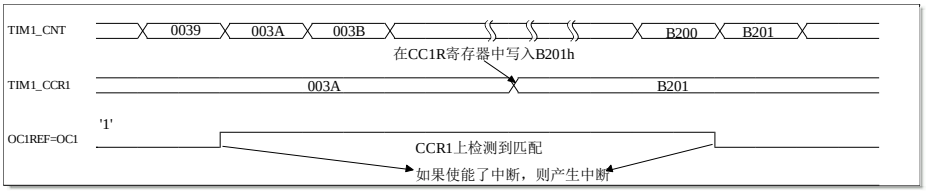


图 9-24 输出比较模式，翻转 OC1

9.4.10 PWM 模式

脉冲宽度调制模式可以产生一个由 TIMERx_ARR 寄存器确定频率、由 TIMERx_CCRx 寄存器确定占空比的信号。

在 TIMERx_CCMRx 寄存器中的 OCxM 位写入 110（PWM 模式 1）或 111（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIMERx_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIMERx_CR1 寄存器的 ARPE 位，（在向上计数或中心对称模式中）使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIMERx_EGR 寄存器中的 UG 位来初始化所有的寄存器。

OCx 的极性可以通过软件在 TIMERx_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过（TIMERx_CCER 和 TIMERx_BDTR 寄存器中）CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详见 TIMERx_CCER 寄存器的描述。

在 PWM 模式（模式 1 或模式 2）下，TIMERx_CNT 和 TIMERx_CCRx 始终在进行比较，（依据计数器的计数方向）以确定是否符合 TIMERx_CCRx ≤

TIMERx_CNT 或者 TIMERx_CNT ≤ TIMERx_CCRx。根据 TIMERx_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

PWM 边沿对齐模式

- 向上计数配置

当 TIMERx_CR1 寄存器中的 DIR 位为低的时候执行向上计数。下面是一个 PWM 模式 1 的例子。当 TIMERx_CNT < TIMERx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMERx_CCRx 中的比较值大于自动重装载值（TIMERx_ARR），则 OCxREF 保持为 1。如果比较值为 0，则 OCxREF 保持为 0。下图为 TIMERx_ARR=8 时边沿对齐的 PWM 波形实例。

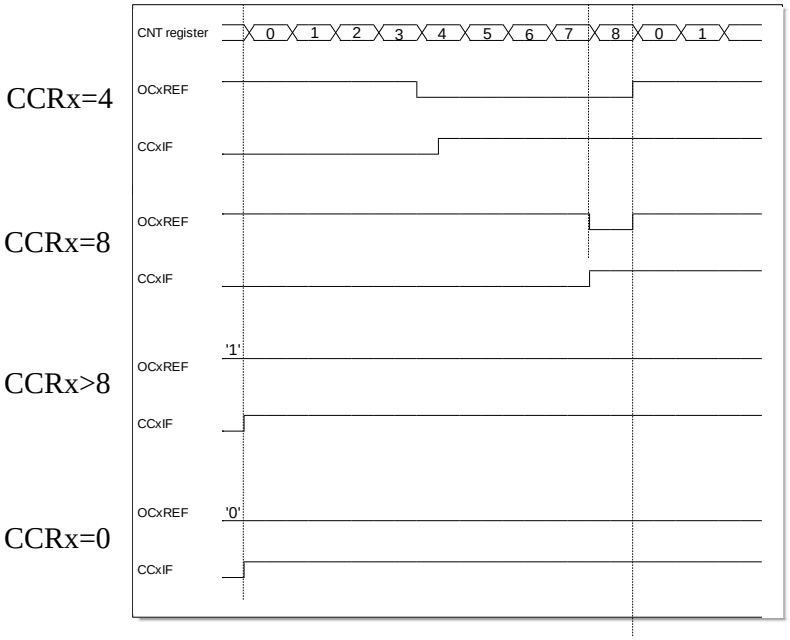


图 9-25 边沿对齐的 PWM 波形（ARR=8）

- 向下计数配置

当 TIMERx_CR1 寄存器的 DIR 位为高时执行向下计数。在 PWM 模式 1 下，当 TIMERx_CNT > TIMERx_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIMERx_CCRx 中的比较值大于 TIMERx_ARR 中的自动重装载值，则 OCxREF 保持为 1。该模式下不能产生 0%的 PWM 波形。

PWM 中央对齐模式

当 TIMERx_CR1 寄存器中的 CMS 位不为 00 时为中央对齐模式（所有其他的配置对 OCxREF/OCx 信号都有相同的作用）。根据不同的 CMS 位设置，比较标志可以在计数器向上计数、在计数器向下计数、或在计数器向上和向下计数时被置 1。注意不要用软件修改 TIMERx_CR1 寄存器中的计数方向位（DIR）。

下图给出了一些中央对齐的 PWM 波形的例子

- TIMx_ARR = 8；
- PWM 模式 1；
- TIMx_CR1 寄存器的 CMS = 01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

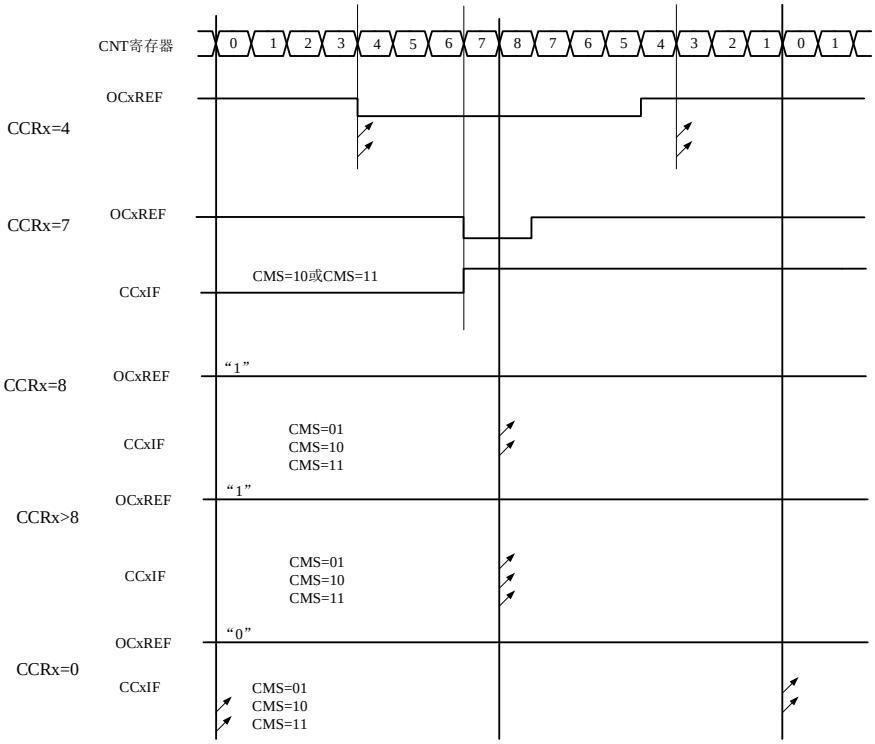


图 9-26 中央对齐的 PWM 波形（APR=8）

使用中央对齐模式的提示：

- 进入中央对齐模式时，使用当前的向上/向下计数配置；即，计数器的计数方向取决于 TIMx_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位；
- 在中央对齐模式下运行时，不推荐对计数器进行改写，因为这会产生不可预知的结

果。特别地：

— 如果写入计数器的值大于自动重加载的值（TIMERx_CNT > TIMERx_ARR），则方向不会被更新。即，向上计数的计数器在该情况下会继续向上计数，其他同理；

— 如果将 0 或者 TIMERx_ARR 的值写入计数器，方向会被更新，但不会产生更新事件 UEV。

● 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新（设置 TIMERx_EGR 位中的 UG 位），并且不要在计数进行过程中修改计数器的值。

9.4.11 互补输出和死区插入

高级控制定时器能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性（电平转换的延时、电源开关的延时等）来调整死区时间。配置 TIMERx_CCER 寄存器中的 CCxP 和 CCxNP 位，可以为每一个输出独立地选择极性（主输出 OCx 或互补输出 OCxN）。

互补信号 OCx 和 OCxN 通过下列控制位的组合进行控制：TIMERx_CCER 寄存器的 CCxE 和 CCxNE 位，TIMERx_BDTR 和 TIMERx_CR1 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，详见表 9-1。特别地，在转换到 IDLE 状态时（MOE 下降到 0）时，死区会被激活。

同时设置 CCxE 和 CCxNE 位将插入死区，如果存在刹车电路，则还要设置 MOE 位。每一个通道都有一个 8 位的死区发生器。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。如果 OCx 和 OCxN 为高位有效：

● OCx 输出信号与参考信号相同，只是它的上升沿相对于参考信号的上升沿有一个延迟；

● OCxN 输出信号与参考信号相反，只是它的上升沿相对于参考信号的下降沿有一个延迟。如果延迟大于当前有效的输出宽度（OCx 或者 OCxN），则不会产生相应的脉冲。下列几张图显示了死区发生器的输出信号和当前参考信号 OCxREF 之间的关系。（假设 CCxP = 0、CCxNP = 0、MOE = 1、CCxE = 1 并且 CCxNE = 1）。

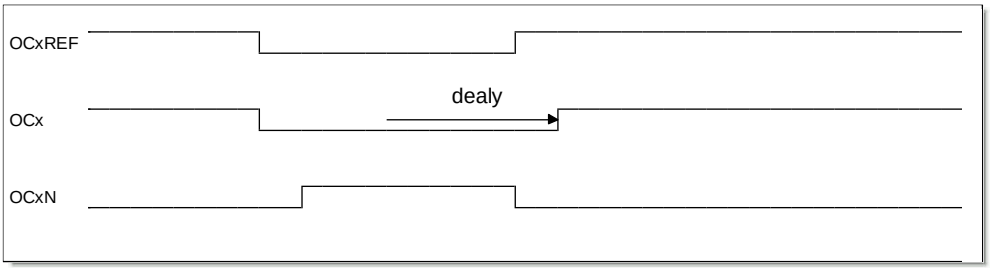


图 9-27 带死区插入的互补输出

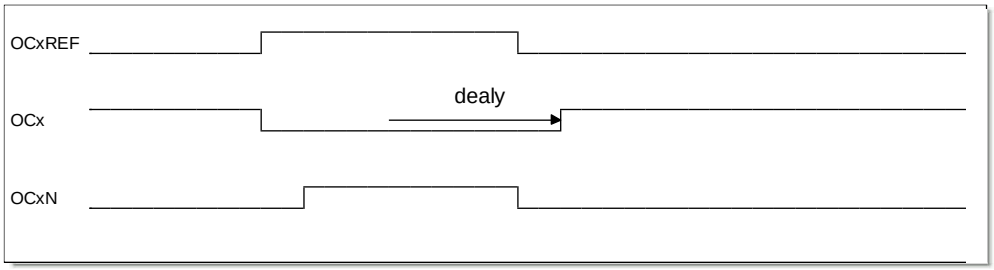


图 9-28 死区波形延迟大于负脉冲

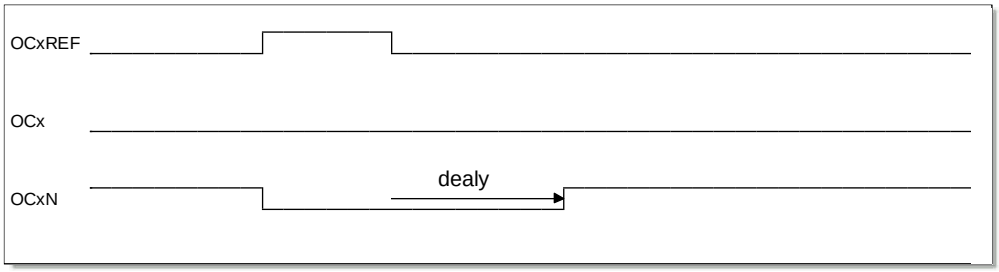


图 9-29 死区波形延迟大于正脉冲

每一个通道的死区延时都是相同的，是由 TIMERx_BDTR 寄存器中的 DTG 位编程配置的。

重定向 OCxREF 到 OCx 或 OCxN

在输出模式下（强置、输出比较或 PWM），通过配置 TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形（例如 PWM 或者静态有效电平）。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出状态。

注：当只使能 OCxN（CCxE = 0，CCxNE = 1）时，OCx 和 OCxN 不互补相。当 OCxREF 有效时 OCxN 立即变高。例如，如果 CCxNP = 0，则 OCxN = OCxREF。另一方面，当 OCx 和 OCxN 都被使能时（CCxE = CCxNE = 1），当 OCxREF 为高时 OCx 有效；而 OCxN 相反，当 OCxREF 低时 OCxN 变为有效。

9.4.12 刹车功能

当使用刹车功能时，依据相应的控制位（TIMERx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIMERx_CR1 寄存器中的 OISx 和 OISxN 位），输出使能信

号和无效电平都会被修改。但无论何时，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。详见表 9-1。

当发生刹车时（在刹车输入端出现选定的电平），有下述动作：

- MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态（由 OSSI 位选择）；
- 一旦 MOE = 0，每一个输出通道输出由 TIMx_CR1 寄存器中的 OISx 位设定的电平决定。如果 OSSI = 0，则定时器释放使能输出，否则使能输出始终为高；
- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态（取决于极性）；
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。注：因为重新同步 MOE，死区时间比通常情况下长一些（大约 2 个 ck 的时钟周期）；
  - 如果 OSSI = 0，定时器释放使能输出，否则保持使能输出；或一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
- 如果设置了 TIMERx_DIER 寄存器中的 BIE 位，当刹车状态标志（TIMx_SR 寄存器中的 BIF 位）为 1 时，则产生一个中断；
- 如果设置了 TIMERx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；这可以用来进行整形。否则，MOE 始终保持低直到被再次置 1；这个特性也可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

刹车由 BKI 输入产生，它的有效极性是可编程的，且由 TIMERx_BDTR 寄存器中的 BKE 位开启。除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数（死区长度，OCx/OCxN 极性和被禁止的状态，OCxM 配置，刹车使能和极性）。用户可以通过 TIMERx_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种。在 MCU 复位后 LOCK 位只能被修改一次。

下图显示响应刹车的输出实例。

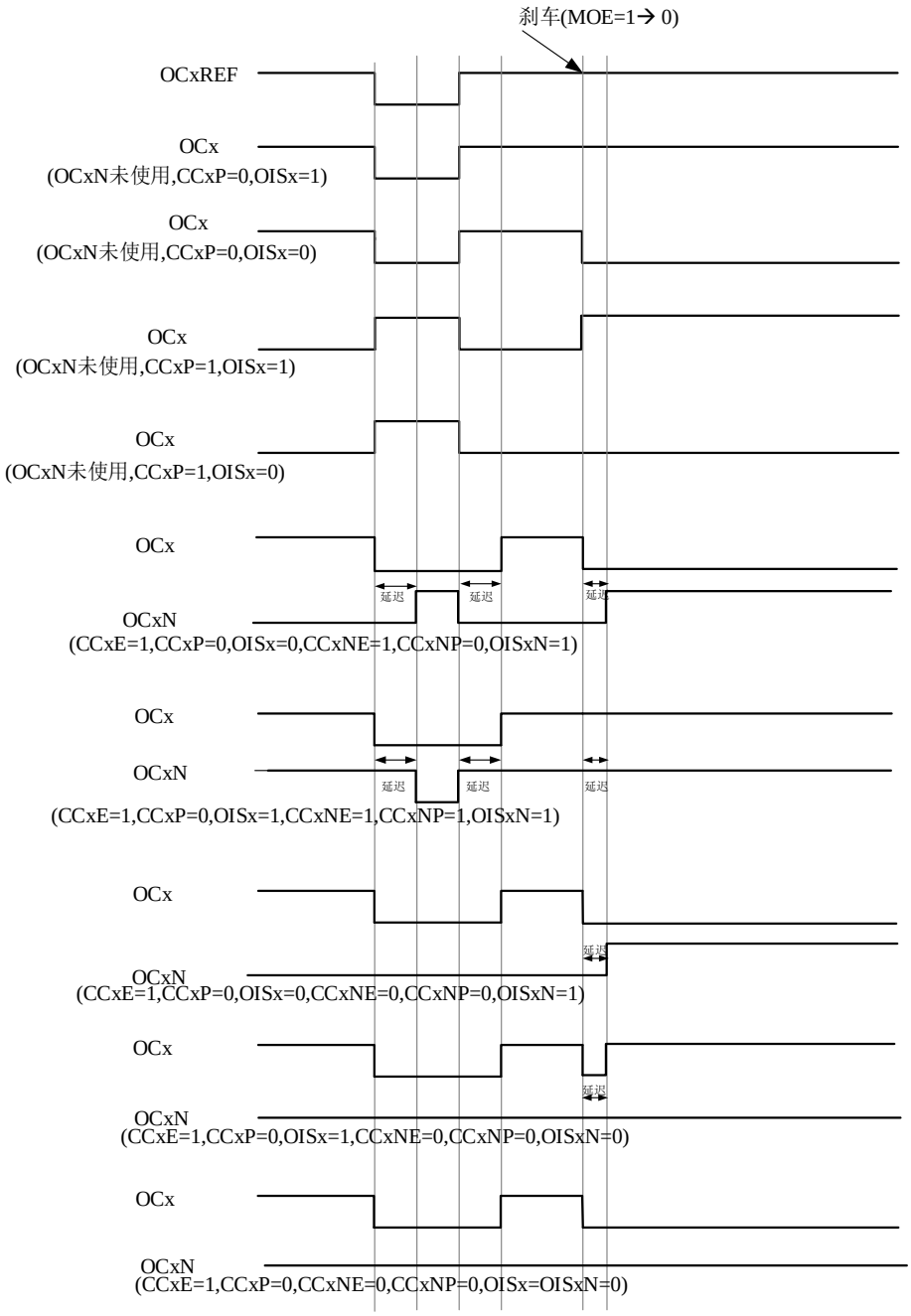


图 9-30 响应刹车的输出

9.4.13 六步 PWM 产生

当在一个通道上需要互补输出时，预装载位有 OCxM、CCxE 和 CCxNE。在发生 COM 换相事件时，这些预装载位被传送到影子寄存器位。用户可以预先设置好下一步骤的配置，并在同一个时刻同时修改所有通道的配置。COM 可以通过设置 TIMERx_EGR 寄存器的 COM 位由软件产生，或在 TRGI 上升沿由硬件

产生。当发生 COM 事件时会设置一个标志位（TIMERx_SR 寄存器中的 COMIF 位），这时如果已设置了 TIMERx_DIER 寄存器的 COMIE 位，则产生一个中断。

下图显示当发生 COM 事件时，三种不同配置下 OCx 和 OCxN 输出。

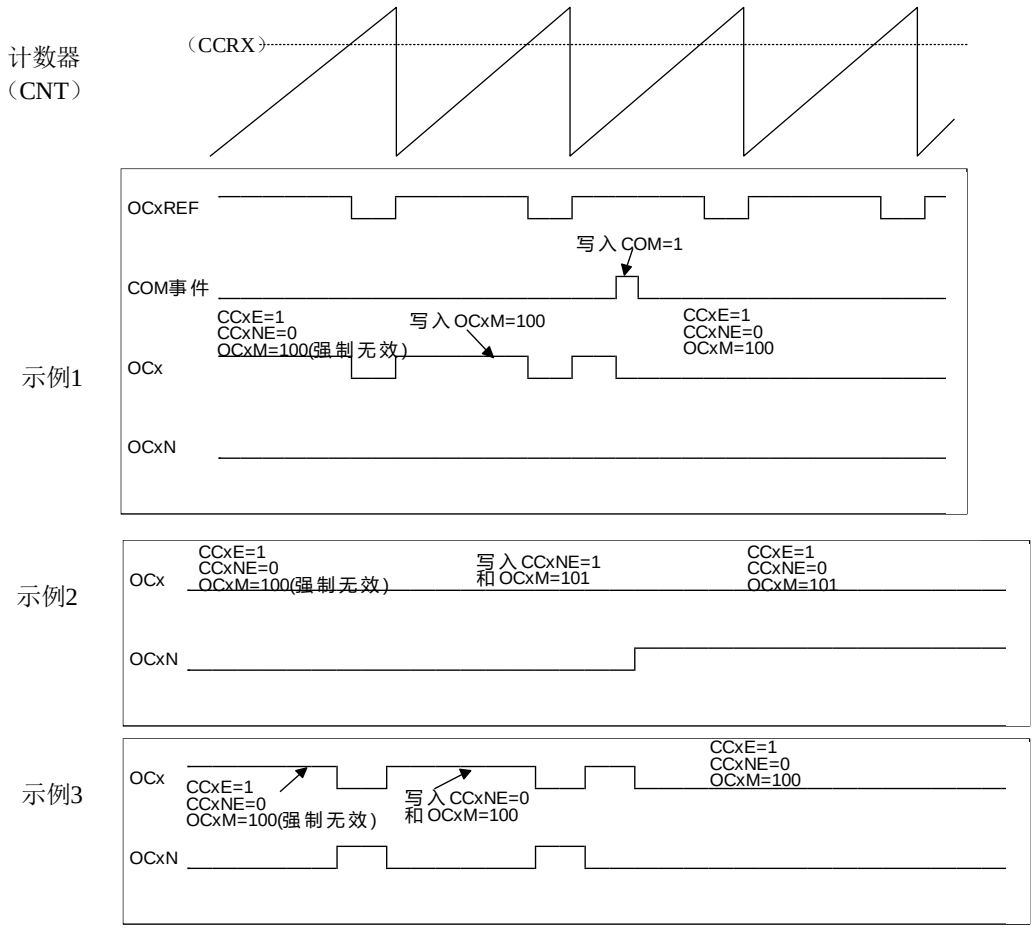


图 9-31 产生六步 PWM，使用 COM 的例子（OSSR=1）

9.4.14 单脉冲模式

单脉冲模式（OPM）是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可被程序控制的脉冲。

可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIMERx_CR1 寄存器中的 OPM 位应选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前（当定时器正在等待触发），必须如下配置：

- 向上计数方式：计数器 CNT < CCRx ≤ ARR（特别地，CCRx > 0）；

- 向下计数方式：计数器 CNT > CCRx。

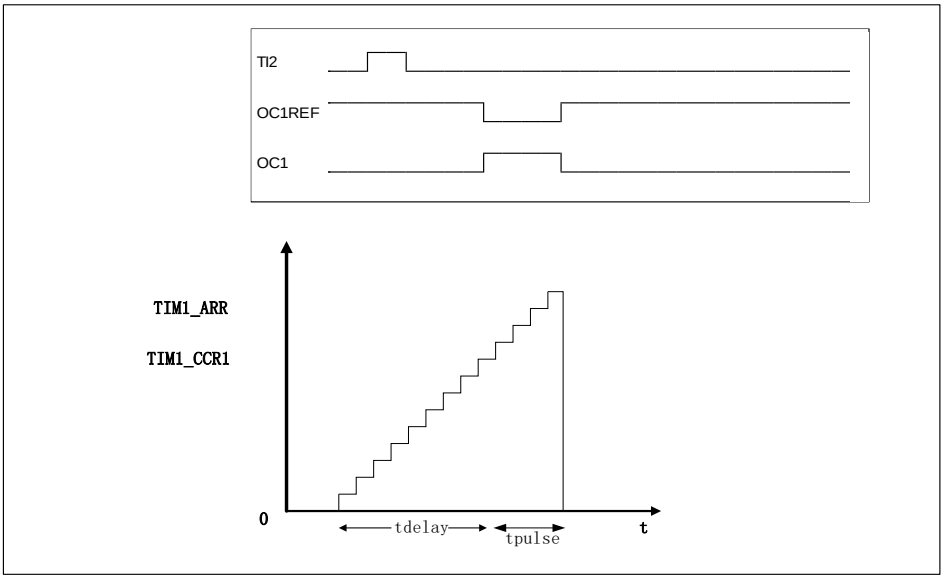


图 9-32 单脉冲模式的例子

例如，你需要在从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后，在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲。

假定 TI2FP2 作为触发 1：

- 置 TIMERx_CCMR1 寄存器中的 CC2S = 01，把 TI2FP2 映像到 TI2；
- 置 TIMERx_CCER 寄存器中的 CC2P = 0，使 TI2FP2 能够检测上升沿；
- 置 TIMERx_CR1 寄存器中的 TS = 2’b10，TI2FP2 作为从模式控制器的触发（TRGI）
- 置 TIMERx_CR1 寄存器中的 SMS = 110（触发模式），TI2FP2 被用来启动计数器，OPM 的波形由写入比较寄存器的数值决定（要考虑时钟频率和计数器预分频器）；
- t_{DELAY} 由 TIMx_CCR1 寄存器中的值定义；
- t_{PULSE} 由自动装载值和比较值之间的差值定义（TIMERx_ARR -TIMERx_CCR1）；
- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；首先要置 TIMERx_CCMR1 寄存器的 OC1M = 111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIMERx_CCMR1 中的 OC1PE = 1 和 TIMERx_CR1 寄存器中的 ARPE；然后在 TIMERx_CCR1 寄存器中填写比较值，在 TIMERx_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P = 0。

在这个例子中，TIMERx_CR1 寄存器中的 DIR 和 CMS 位应该置低。因为只需要一个脉冲，所以必须设置 TIMERx_CR1 寄存器中的 OPM = 1，在下一个更新事件时停止计数。

9.4.15 定时器输入异或功能

TIMERx_CR1 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TIMERx_CH1、TIMERx_CH2 和 TIMERx_CH3。异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。

9.4.16 与霍尔传感器的接口

使用定时器 TIMER1 产生 PWM 信号来驱动马达，另一个定时器作为“接口定时器”与霍尔传感器相连。3 个定时器输入引脚（TIMERx_CH1、TIMERx_CH2 和 TIMERx_CH3）通过异或门连接到 TI1 输入通道（通过将 TIMERx_CR1 寄存器中的 TI1S 位置 1 来选择），并由接口定时器进行捕获。

从模式控制器配置为复位模式；从输入为 TI1F_ED。因而，每次 3 个输入中有一个输入变化时，计数器会从 0 开始重新计数。这样将产生由霍尔输入的任何变化而触发的时基。

在接口定时器上，捕获/比较通道 1 配置为捕获模式，捕获信号 TRC。捕获值对应于输入上两次变化的间隔时间，可提供与电机转速相关的信息。

接口定时器可用于在输出模式下产生脉冲，以通过触发 COM 事件更改定时器各个通道的配置。TIMER1 定时器用于生成电机驱动 PWM 信号。为此，必须对接口定时器通道进行编程，以便在编程的延迟过后产生正脉冲（在输出比较或 PWM 模式中）。该脉冲通过 TRGO 输出发送到另一个定时器。

示例：霍尔输入与一个 TIMER2 定时器相连接，要求每次任一霍尔输入上发生变化后的一个指定时刻，改变定时器 TIMER1 的 PWM 配置。

- 置 TIMER2_CR1 寄存器的 TI1S 位为 1，使 3 个定时器输入经过异或运算后进入 TI1 输入通道；
- 时基编程：向 TIMER2_ARR 写入其最大值 16’hffff（计数器必须通过 TI1 的变化清零）。设置预分频器，以得到最大计数器周期，该周期长于传感器上两次变化的间隔时间；
- 将通道 1 编程为捕获模式（选择 TRC）：配置 TIMER2_CCMR1 寄存器的 CC1S = 2’b11。如果需要，还可以编程数字滤波器；
- 将通道 2 编程为 PWM2 模式，并具有所需延迟：置 TIMERx_CCMR1 寄存器的 OC2M = 3’b111，CC2S = 2’b00；
- 选择 OC2REF 作为 TRGO 上的触发输出：置 TIMER2_CR1 寄存器的 MMS = 3’b101，TS = 2’b11。

在定时器 TIMER1 中，必须选择正确的 ITR 输入作为触发输入，定时器编

程为可产生 PWM 信号，捕获/比较控制信号进行预装载（TIMER2_CR1 寄存器的 CCPC = 1），并且 COM 事件由触发输入控制（TIMER2_CR1 寄存器中 CCUS = 1）。发生 COM 事件后，在 PWM 控制位（CCxE、OCxM）中写入下一步的配置，此操作可在由 OC2REF 上升沿产生的中断子程序中完成。

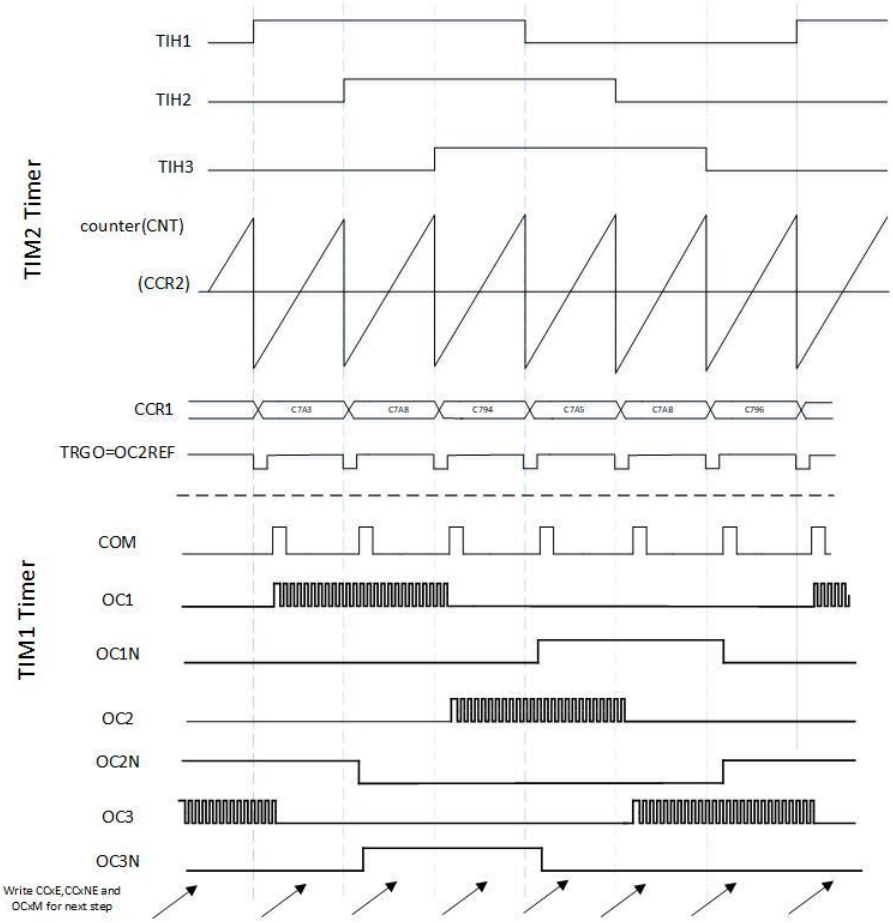


图 9-33 霍尔传感器接口的例子

9.4.17 定时器和外部触发的同步

TIMERx 定时器能够在多种模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

从模式：复位模式

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIMERx_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然

后所有的预装载寄存器（TIMERx_ARR，TIMERx_CCRx）都会被更新。

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽（在本例中，不需要任何滤波器，因此保持 IC1F = 0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 TIMERx_CCMR1 寄存器中 CC1S = 01。置 TIMERx_CCER 寄存器中 CC1P = 0 以确定极性（只检测上升沿）；
- 置 TIMERx_CR1 寄存器中 SMS = 100，配置定时器为复位模式；置 TIMERx_CR1 寄存器中 TS = 2’b01，选择 TI1 作为输入源；
- 置 TIMERx_CR1 寄存器中 CEN = 1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志（TIMERx_SR 寄存器中的 TIF 位）被设置。

下图显示当自动重装载寄存器 TIMERx_ARR = 0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

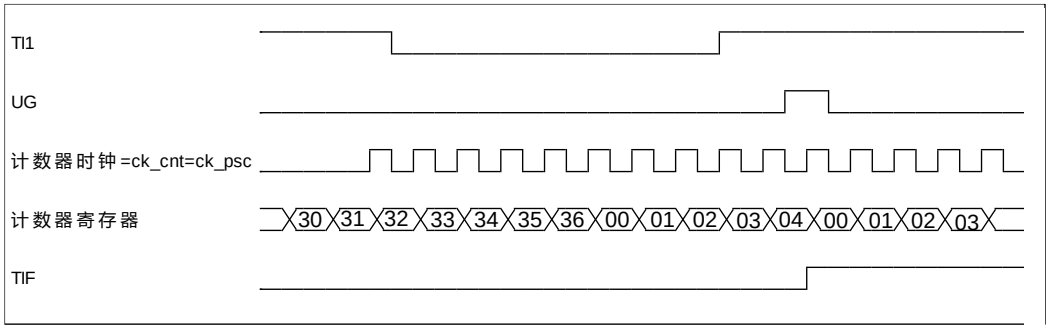


图 9-34 复位模式下的控制电路

从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽（本例中，不需要滤波，所以保持 IC1F = 0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIMERx_CCMR1 寄存器中 CC1S = 01。置 TIMERx_CCER 寄存器中 CC1P = 1 以确定极性（只检测低电平）；
- 置 TIMERx_CR1 寄存器中 SMS = 101，配置定时器为门控模式；置 TIMERx_CR1 寄存器中 TS = 2’b01，选择 TI1 作为输入源；
- 置 TIMERx_CR1 寄存器中 CEN = 1，启动计数器。在门控模式下，如果 CEN = 0，

则计数器不能启动，不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时都设置 TIMERx_SR 中的 TIF 标置。TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的重同步电路。

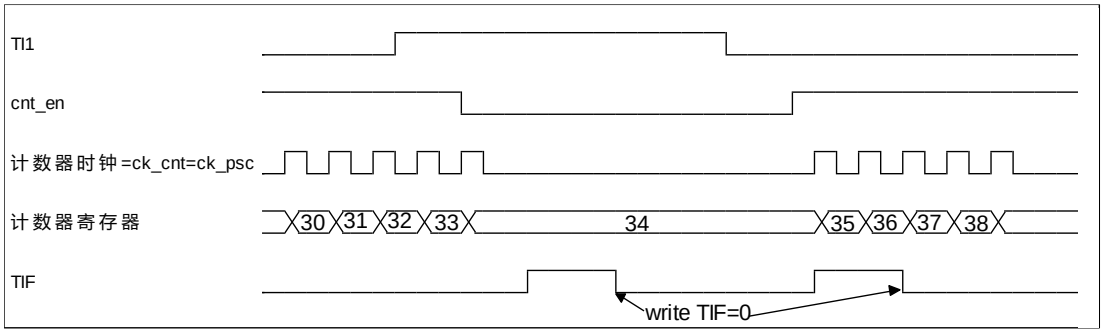


图 9-35 门控模式下的控制电路

从模式：触发模式

输入端上选中的事件使能计数器。在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- 配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽（本例中，不需要任何滤波器，保持 IC2F = 0000）。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIMERx_CCMR1 寄存器中 CC2S = 01。置 TIMERx_CCER 寄存器中 CC2P = 1 以确定极性（只检测低电平）；
- 置 TIMERx_CR1 寄存器中 SMS = 110，配置定时器为触发模式；置 TIMERx_CR1 寄存器中 TS = 2’b10，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。

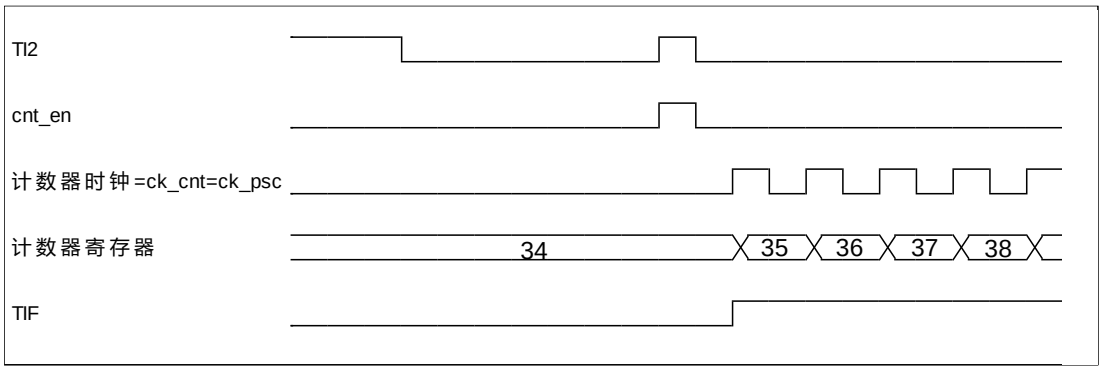


图 9-36 触发器模式下的控制电路

9.5 TIMERx 寄存器描述

TIM1 基址：0x3000_0018

TIM2 基址：0x3000_0098

9.5.1 控制寄存器（TIMERx_CR1）

偏移地址：0x00

复位值：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

位	标记	功能描述
31	Reserved	保留位
30:28	MMS	100：比较 – OC1REF 信号被用于作为触发输出（TRGO）； 101：比较 – OC2REF 信号被用于作为触发输出（TRGO）； 110：比较 – OC3REF 信号被用于作为触发输出（TRGO）； 111：比较 – OC4REF 信号被用于作为触发输出（TRGO）
27:26	Reserved	保留位
25:24	TS	触发选择，选择用于同步计数器的触发输入。 00：TI1 的边沿检测器，TI1 的上/降沿均有效（TI1F_ED）； 01：滤波后的定时器输入 1（TI1FP1）； 10：滤波后的定时器输入 2（TI2FP2）； 11：ITR（timer1 中选择的是 timer2 的 TRGO，timer2 中选择的是 timer1 的 TRGO）
23	TI1S	TI1S：TI1 选择 0：TIM1_CH1 管脚连到 TI1 输入；

		1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 管脚经异或后连到 TI1 输入
22:20	SMS	当选择了外部信号，触发信号（TRGI）的有效边沿与选中的外部输入极性关。 100：复位模式：选中的触发输入（TRGI）的上升沿重新初始化计数器，并且产生一个更新寄存器的信号； 101：门控模式：当触发输入（TRGI）为高时，计数器的时钟开启。一旦触发输入变为低，则计数器停止（但不复位）。计数器的启动和停止都是受控的； 110：触发模式 – 计数器在触发输入 TRGI 的上升沿启动（但不复位），只有计数器的启动是受控的； 111：外部时钟模式 1 – 选中的触发输入（TRGI）的上升沿驱动计数器 *注：如果 TI1F_ED 被选为触发输入（TS = 00）时，不要使用门控模式。这是因为 TI1F_ED 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。*
19	CCUS	捕获/比较控制更新选择 0：如果捕获/比较控制位是预装载的（CCPC=1），只能通过设置 COM 位更新它们； 1：如果捕获/比较控制位是预装载的（CCPC=1），可以通过设置 COM 位或 TRGI 上的一个上升沿更新它们。 *注：该位只对具有互补输出的通道起作用。*
18	CCPC	捕获/比较预装载控制位 0：CCxE，CCxNE 和 OCxM 位不是预装载的； 1：CCxE，CCxNE 和 OCxM 位是预装载的；设置该位后，它们只在设置了 COM 位后被更新。 *注：该位只对具有互补输出的通道起作用。*
17	OIS4N	输出空闲状态 1（OC4N 输出）。 0：当 MOE = 0 时，死区后 OC4N = 0； 1：当 MOE = 0 时，死区后 OC4N = 1。 *注：已经设置了 LOCK（TIMER1_BKR 寄存器）级别 1、2 或 3 后，该位不能被修改。*
16	OIS4	输出空闲状态 1（OC4 输出）。 0：当 MOE = 0 时，如果实现了 OC4N，则死区后 OC4 = 0； 1：当 MOE = 0 时，如果实现了 OC4N，则死区后 OC4 = 1。 *注：已经设置了 LOCK（TIMER1_BKR 寄存器）级别 1、2 或 3 后，该位不能被修改。*
15	OIS3N	输出空闲状态 1（OC3N 输出）。 0：当 MOE = 0 时，死区后 OC3N = 0； 1：当 MOE = 0 时，死区后 OC3N = 1。 *注：已经设置了 LOCK（TIMER1_BKR 寄存器）级别 1、2 或 3 后，该位不能被修改。*
14	OIS3	输出空闲状态 1（OC3 输出）。 0：当 MOE = 0 时，如果实现了 OC3N，则死区后 OC3 = 0；

		1：当 MOE = 0 时，如果实现了 OC3N，则死区后 OC3 = 1。 *注：已经设置了 LOCK（TIMER1_BKR 寄存器）级别 1、2 或 3 后，该位不能被修改。*
13	OIS2N	输出空闲状态 1（OC2N 输出）。 0：当 MOE = 0 时，死区后 OC2N = 0； 1：当 MOE = 0 时，死区后 OC2N = 1。 *注：已经设置了 LOCK（TIMER1_BKR 寄存器）级别 1、2 或 3 后，该位不能被修改。*
12	OIS2	输出空闲状态 1（OC2 输出）。 0：当 MOE = 0 时，如果实现了 OC2N，则死区后 OC2 = 0； 1：当 MOE = 0 时，如果实现了 OC2N，则死区后 OC2 = 1。 *注：已经设置了 LOCK（TIMER1_BKR 寄存器）级别 1、2 或 3 后，该位不能被修改。*
11	OIS1N	输出空闲状态 1（OC1N 输出）。 0：当 MOE = 0 时，死区后 OC1N = 0； 1：当 MOE = 0 时，死区后 OC1N = 1。 *注：已经设置了 LOCK（TIMER1_BKR 寄存器）级别 1、2 或 3 后，该位不能被修改。*
10	OIS1	输出空闲状态 1（OC1 输出）。 0：当 MOE = 0 时，如果实现了 OC1N，则死区后 OC1 = 0； 1：当 MOE = 0 时，如果实现了 OC1N，则死区后 OC1 = 1。 *注：已经设置了 LOCK（TIMER1_BKR 寄存器）级别 1、2 或 3 后，该位不能被修改。*
9:8	CKD	时钟分频因子 这 2 位定义在定时器时钟（CK_INT）频率与数字滤波器（Tix）使用的采样频率之间的分频比例。 00：$t_{DTS} = t_{CK\_INT}$； 01：$t_{DTS} = 2 \times t_{CK\_INT}$； 10：$t_{DTS} = 4 \times t_{CK\_INT}$； 11：保留
7	ARPE	自动重装载预装载允许位 0：TIMERx_ARR 寄存器没有缓冲； 1：TIMERx_ARR 寄存器被装入缓冲器
6:5	CMS	选择中央对齐模式 00：边沿对齐模式。计数器依据方向位（DIR）向上或向下计数； 01：中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道（TIMERx_CCMRx 寄存器中 CCxS=00）的输出比较中断标志位，只在计数器向下计数时被设置； 10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道（TIMERx_CCMRx 寄存器中 CCxS=00）的输出比较中断标志位，只在计数器向上计数时被设置； 11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道

		（TIMERx_CCMRx 寄存器中 CCxS = 00）的输出比较中断标志位，在计数器向上和向下计数时均被设置； *注：在计数器开启时（CEN=1），不允许从边沿对齐模式转换到中央对齐模式。*
4	DIR	方向 0：计数器向上计数； 1：计数器向下计数。 *注：当计数器配置为中央对齐模式或编码器模式时，该位为只读。*
3	OPM	单脉冲模式 0：在发生更新事件时，计数器不停止； 1：在发生下一次更新事件（清除 CEN 位）时，计数器停止
2	URS	更新请求源 软件通过该位选择 UEV 事件的源 0：如果允许产生更新中断，则下述任一事件产生一个更新中断： – 计数器溢出/下溢 – 设置 UG 位 – 从模式控制器产生的更新； 1：如果允许产生更新中断，则只有计数器溢出/下溢产生一个更新中断
1	UDIS	禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0：允许 UEV。更新（UEV）事件由下述任一事件产生： – 计数器上溢/下溢 – 设置 UG 位 – 从模式控制器产生的更新 被缓存的寄存器被装入它们的预装载值； 1：禁止 UEV。不产生更新事件，影子寄存器（ARR，PSC，CCRx）保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化
0	CEN	允许计数器 0：禁止计数器； 1：开启计数器。 *注：在软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。在单脉冲模式下，当发生更新事件时，CEN 被自动清除。*

9.5.2 滤波寄存器（TIMERx_ICF）

偏移地址：0x04

复位值：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			

位	标记	功能描述
31:16	Reserved	保留位
15:12	IC4F[3:0]	输入捕获 4 滤波器
11:8	IC3F[3:0]	输入捕获 3 滤波器
7:4	IC2F[3:0]	输入捕获 2 滤波器
3:0	IC1F[3:0]	输入捕获 1 滤波器 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成， 它记录到 N 个事件后会产生一个输出的跳变： 0000：无滤波器，以 fDTS 采样； 0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2； 0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4； 0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8； 0100：采样频率 $f_{SAMPLING}=f_{DTS/2}$，N=6； 0101：采样频率 $f_{SAMPLING}=f_{DTS/2}$，N=8； 0110：采样频率 $f_{SAMPLING}=f_{DTS/4}$，N=6； 0111：采样频率 $f_{SAMPLING}=f_{DTS/4}$，N=8； 1000：采样频率 $f_{SAMPLING}=f_{DTS/8}$，N=6； 1001：采样频率 $f_{SAMPLING}=f_{DTS/8}$，N=8； 1010：采样频率 $f_{SAMPLING}=f_{DTS/16}$，N=5； 1011：采样频率 $f_{SAMPLING}=f_{DTS/16}$，N=6； 1100：采样频率 $f_{SAMPLING}=f_{DTS/16}$，N=8； 1101：采样频率 $f_{SAMPLING}=f_{DTS/32}$，N=5； 1110：采样频率 $f_{SAMPLING}=f_{DTS/32}$，N=6； 1111：采样频率 $f_{SAMPLING}=f_{DTS/32}$，N=8。 *注：在现在的芯片版本中，当 ICxF[3:0]=1，2 或 3 时，公式中的 f_{DTS} 由 CK_INT 替代。*

9.5.3 中断使能寄存器（TIMERx_DIER）

偏移地址：0x08

复位值：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	

位	标记	功能描述
31:8	Reserved	保留位
7	BIE	允许刹车中断 0：禁止刹车中断； 1：允许刹车中断
6	TIE	允许触发中断 0：禁止触发中断； 1：允许触发中断
5	COMIE	允许 COM 中断 0：禁止 COM 中断； 1：允许 COM 中断
4	CC4IE	允许捕获/比较 4 中断 0：禁止捕获/比较 4 中断； 1：允许捕获/比较 4 中断
3	CC3IE	允许捕获/比较 3 中断 0：禁止捕获/比较 3 中断； 1：允许捕获/比较 3 中断
2	CC2IE	允许捕获/比较 2 中断 0：禁止捕获/比较 2 中断； 1：允许捕获/比较 2 中断
1	CC1IE	允许捕获/比较 1 中断 0：禁止捕获/比较 1 中断； 1：允许捕获/比较 1 中断
0	UIE	允许更新中断 0：禁止更新中断； 1：允许更新中断

9.5.4 状态寄存器（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

位	标记	功能描述
31:12	Reserved	保留位
11	CC4OF	捕获/比较 4 过捕获标记 参见 CC1OF 描述
10	CC3OF	捕获/比较 3 过捕获标记 参见 CC1OF 描述
9	CC2OF	捕获/比较 2 过捕获标记 参见 CC1OF 描述
8	CC1OF	捕获/比较 1 过捕获标记 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。写 0 可清除该位。 0：无过捕获产生； 1：CC1IF 置 1 时，计数器的值已经被捕获到 TIMERx_CCR1 寄存器。
7	BIF	刹车中断标记 一旦刹车输入有效，由硬件对该位置 1。如果刹车输入无效，则该位可由软件清 0。 0：无刹车事件产生； 1：刹车输入上检测到有效电平
6	TIF	触发器中断标记 当发生触发事件（当从模式控制器处于除门控模式外的其它模式时，在 TRGI 输入端检测到有效边沿，或或门控模式下的任一边沿）时由硬件对该位置 1。该位由软件清 0。 0：无触发器事件产生； 1：触发器中断等待响应
5	COMIF	COM 中断标记 一旦产生 COM 事件（当 CCxE、CCxNE、OCxM 已被更新），该位由硬件置 1。该位由软件清 0。 0：无 COM 事件产生； 1：COM 中断等待响应
4	CC4IF	捕获/比较 4 中断标记 参考 CC1IF 描述
3	CC3IF	捕获/比较 3 中断标记 参考 CC1IF 描述
2	CC2IF	捕获/比较 2 中断标记 参考 CC1IF 描述
1	CC1IF	捕获/比较 1 中断标记 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外（参考 TIMER1_CR1 寄存器的 CMS 位）。该位由软件清 0。 0：无匹配发生；

位	标记	功能描述
		1：TIM1_CNT 的值与 TIM1_CCR1 的值匹配。 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置 1，该位由软件清 0 或通过 TIMERx_CCR1 清 0。 0：无输入捕获产生； 1：输入捕获产生并且计数器值已装入 TIMERx_CCR1（在 IC1 上检测到与所选极性相同的边沿）
0	UIF	更新中断标记 当产生更新事件时该位由硬件置 1。该位由软件清 0。 0：无更新事件产生； 1：更新事件等待响应。当寄存器被更新时该位由硬件置 1： – 若 TIMERx_CR1 寄存器的 UDIS = 0，当 REP_CNT = 0 时产生更新事件（重复向下计数器上溢或下溢时）； – 若 TIMERx_CR1 寄存器的 UDIS = 0、URS = 0，当 TIMERx_EGR 寄存器的 UG=1 时产生更新事件（软件对 CNT 重新初始化）； – 若 TIMERx_CR1 寄存器的 UDIS = 0、URS = 0，当 CNT 被触发事件重初始化时产生更新事件

9.5.5 事件产生寄存器（TIMERx_EGR）

偏移地址：0x10

复位值：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	

位	标记	功能描述
31：8	Reserved	保留位
7	BG	产生刹车事件 该位由软件置 1，用于产生一个刹车事件，由硬件自动清 0。 0：无动作； 1：产生一个刹车事件。此时 MOE = 0、BIF =1。
6	TG	产生触发事件 该位由软件置 1，用于产生一个触发事件，由硬件自动清 0. 0：无动作； 1：产生一个触发事件。此时 TIF =1。

5	COMG	捕获/比较事件，产生控制更新 该位由软件置 1，由硬件自动清 0。 0：无动作； 1：当 CCPC = 1，允许更新 CCxE、CCxNE、OCxM 位。 注：该位只对有互补输出的通道有效
4	CC4G	产生捕获/比较 4 事件 参考 CC1G 描述
3	CC3G	产生捕获/比较 3 事件 参考 CC1G 描述
2	CC2G	产生捕获/比较 2 事件 参考 CC1G 描述
1	CC1G	产生捕获/比较 1 事件 该位由软件置 1，用于产生一个捕获/比较事件，由硬件自动清 0。 0：无动作； 1：在通道 CC1 上产生一个捕获/比较事件： 若通道 CC1 配置为输出： 设置 CC1IF=1，若开启对应的中断，则产生相应的中断。 若通道 CC1 配置为输入： 当前的计数器值捕获至 TIMERx_CCR1 寄存器，设置 CC1IF = 1，若开启对应的中断，则产生相应的中断。若 CC1IF 已经为 1，则设置 CC1OF = 1
0	UG	产生更新事件 该位由软件置 1，由硬件自动清 0。 0：无动作； 1：重新初始化计数器，并产生一个更新事件。注意预分频器的计数器也被清 0（但是预分频系数不变）。若在中心对称模式下或 DIR = 0（向上计数）则计数器被清 0，若 DIR = 1（向下计数）则计数器取 TIMERx_ARR 的值。

9.5.6 捕获/比较模式寄存器 1（TIMERx_CCMR1）

偏移地址：0x14

复位值：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	

位	标记	功能描述

31:15	RES	保留位
14:12	OC2M[2：0]	输出比较 2 模式
11	OC2PE	输出比较 2 预装载使能
10	RES	保留位
9:8	CC2S[1：0]	捕获/比较 2 选择。 该位定义通道的方向（输入/输出），及输入脚的选择： 00：CC2 通道被配置为输出； 01：CC2 通道被配置为输入，IC2 映射在 TI2 上； 10：CC2 通道被配置为输入，IC2 映射在 TI1 上； 11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIMERx_CR1 寄存器的 TS = 2’b11）。 *注：CC2S 仅在通道关闭时（TIMERx_CCER 寄存器的 CC2E = 0）才是可写的。*
7	RES	保留位
6:4	OC1M[2：0]	输出比较 1 模式 该位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000：冻结。输出比较寄存器 TIMERx_CCR1 与计数器 TIMERx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1（TIMERx_CCR1）相同时，强制 OC1REF 为高； 010：匹配时设置通道 1 为无效电平。当计数器 TIMERx_CNT 的值与捕获/比较寄存器 1（TIMERx_CCR1）相同时，强制 OC1REF 为低； 011：翻转。当 TIMERx_CCR1 = TIMERx_CNT 时，翻转 OC1REF 的电平； 100：强制为无效电平。强制 OC1REF 为低； 101：强制为有效电平。强制 OC1REF 为高； 110：PWM 模式 1－在向上计数时，一旦 TIMERx_CNT < TIMERx_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIM1_CNT > TIM1_CCR1，通道 1 变为无效电平（OC1REF = 0），否则变为有效电平（OC1_REF = 1）； 111：PWM 模式 2－在向上计数时，一旦 TIMERx_CNT < TIMERx_CCR1，通道 1 变为无效电平，否则变为有效电平；在向下计数时，一旦 TIMERx_CNT > TIMERx_CCR1，通道 1 变为有效电平，否则变为无效电平。 *注 1：一旦 LOCK 级别设为 3（TIMx_BDTR 寄存器中的 LOCK 位）并且 CC1S = 00（该通道配置成输出）则该位不能被修改。* *注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。*
3	OC1PE	输出比较 1 预装载使能 0：禁止 TIMERx_CCR1 寄存器的预装载功能，可随时写 TIMERx_CCR1 寄存器，且新值马上起作用；

		1：开启 TIMERx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMERx_CCR1 的预装载值在更新事件到来时被载入当前寄存器中。 *注 1：一旦 LOCK 级别设为 3（TIMERx_BDTR 寄存器中的 LOCK 位）并且 CC1S = 00（该通道配置成输出）则该位不能被修改。* *注 2：仅在单脉冲模式下，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定*
2	RES	保留位
1:0	CC1S[1:0]	捕获/比较 1 选择。 该位定义通道的方向（输入/输出），及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（TIMERx_CR1 寄存器的 TS = 2’b11）。 *注：CC1S 仅在通道关闭时（TIMERx_CCER 寄存器的 CC1E = 0）才是可写的。*

9.5.7 捕获/比较模式寄存器 2（TIMERx_CCMR2）

偏移地址：0x18

复位值：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	

位	标记	功能描述
31:15	RES	保留位
14:12	OC4M[2:0]	输出比较 4 模式
11	OC4PE	输出比较 4 预装载使能
10	RES	保留位
9:8	CC4S[1:0]	捕获/比较 4 选择。 该位定义通道的方向（输入/输出），及输入脚的选择： 00：CC4 通道被配置为输出； 01：CC4 通道被配置为输入，IC4 映射在 TI4 上； 10：CC4 通道被配置为输入，IC4 映射在 TI3 上； 11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触

		发器输入被选中时（TIMERx_CR1 寄存器的 TS = 2’b11）。 *注：CC4S 仅在通道关闭时（TIMERx_CCER 寄存器的 CC4E = 0）才是可写的*
7	RES	保留位
6:4	OC3M[2:0]	输出比较 3 模式
3	OC3PE	输出比较 3 预装载使能
2	RES	保留位
1:0	CC3S[1:0]	捕获/比较 3 选择。 这 2 位定义通道的方向（输入/输出），及输入脚的选择： 00：CC3 通道被配置为输出； 01：CC3 通道被配置为输入，IC3 映射在 TI3 上； 10：CC3 通道被配置为输入，IC3 映射在 TI4 上； 11：CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIMERx_CR1 寄存器的 TS = 2’b11）。 *注：CC3S 仅在通道关闭时（TIMERx_CCER 寄存器的 CC3E = 0）才是可写的*

9.5.8 捕获/比较使能寄存器（TIMERx_CCER）

偏移地址：0x1C

复位值：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

位	标记	功能描述
31:16	Reserved	保留位
15	CC4NP	输入/捕获 4 互补输出极性。参考 CC1NP 的描述。
14	CC4NE	输入/捕获 4 互补输出使能。参考 CC1NE 的描述。
13	CC4P	输入/捕获 4 输出极性。参考 CC1P 的描述。
12	CC4E	输入/捕获 4 输出使能。参考 CC1E 的描述。
11	CC3NP	输入/捕获 3 互补输出极性。参考 CC1NP 的描述。
10	CC3NE	输入/捕获 3 互补输出使能。参考 CC1NE 的描述。

位	标记	功能描述
9	CC3P	输入/捕获 3 输出极性。参考 CC1P 的描述。
8	CC3E	输入/捕获 3 输出使能。参考 CC1E 的描述。
7	CC2NP	输入/捕获 2 互补输出极性。参考 CC1NP 的描述。
6	CC2NE	输入/捕获 2 互补输出使能。参考 CC1NE 的描述。
5	CC2P	输入/捕获 2 输出极性。参考 CC1P 的描述。
4	CC2E	输入/捕获 2 输出使能。参考 CC1E 的描述。
3	CC1NP	输入/捕获 1 互补输出极性 0：OC1N 高电平有效； 1：OC1N 低电平有效。 *注：一旦 LOCK 级别（TIMERx_BDTR 寄存器中的 LCCK 位）设为 3 或 2 且 CC1S = 00（通道配置为输出）则该位不能被修改。*
2	CC1NE	输入/捕获 1 互补输出使能 0：关闭－OC1N 禁止输出，因此 OC1N 的输出电平依赖于 MOE，OSSI，OSSR，OIS1，OIS1N，CC1E 位的值。 1：开启－OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE，OSSI，OSSR，OIS1，OIS1N，CC1E 位的值。
1	CC1P	输入/捕获 1 输出极性 CC1 通道配置为输出： 0：OC1 高电平有效；　　1：OC1 低电平有效。 CC1 通道配置为输入： 该位选择是 IC1 还是 IC1 的反相信号作为触发或捕获信号。 0：不反相－发生在 IC1 的上升沿；当用作外部触发器时，IC1 不反相。 1：反相－捕获发生在 IC1 的下降沿；当用作外部触发器时，IC1 反相。 *注：一旦 LOCK 级别（TIMERx_BDTR 寄存器中的 LCCK 位）设为 3 或 2，则该位不能被修改*
0	CC1E	输入/捕获 1 输出使能 CC1 通道配置为输出： 0：关闭－OC1 禁止输出，因此 OC1 的输出电平依赖于 MOE，OSSI，OSSR，OIS1，OIS1N，CC1NE 位的值； 1：开启－OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE，OSSI，OSSR，OIS1，OIS1N，CC1NE 位的值。 CC1 通道配置为输入： 该位决定了计数器的值是否能捕获入 TIMERx_CCR1 寄存器。 0：捕获禁止；　1：捕获使能

表 9-1 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位					输出状态	
MOE	OSSI	OSSR	CCxE	CCxNE	OCx 输出状态	OCxN 输出状态
1	x	0	0	0	输出禁止（与定时器断开） OCx = 0，OCx_EN = 0	输出禁止（与定时器断开）

		0	0	1	输出禁止（与定时器断开） OCx = 0，OCx_EN = 0	OCxREF + 极性， OCxN = OCxREF xor CCxNP OCxN_EN = 1
		0	1	0	OCxREF + 极性， OCx = OCxREF xor CCxP OCx_EN = 1	输出禁止（与定时器断开) OCxN = 0，OCxN_EN = 0
		0	1	1	OCxREF + 极性 + 死区， OCx_EN=1	OCxREF 反相 + 极性 + 死区 OCxN_EN = 1
		1	0	0	输出禁止（与定时器断开） OCx = CCxP，OCx_EN = 0	输出禁止（与定时器断开） OCxN = CCxNP，OCxN_EN = 0
		1	0	1	关闭状态（输出使能且为无效电平） OCx = CCxP，OCx_EN = 1	OCxREF + 极性， OCxN = OCxREF xor CCxNP OCxN_EN = 1
		1	1	0	OCxREF + 极性， OCx = OCxREF xor CCxP OCx_EN = 1	关闭状态（输出使能且为无效电平） OCxN = CCxNP，OCxN_EN = 1
		1	1	1	OCxREF + 极性 + 死区 OCx_EN = 1	OCxREF 反相 + 极性 + 死区 OCxN_EN = 1
0	0	x	0	0	输出禁止（与定时器断开） 异步地：OCx = CCxP，OCx_EN = 0，OCxN = CCxNP， OCxN_EN = 0；若存在时钟：经过一个死区时间后 OCx = OISx， OCxN = OISxN，假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平	
	0		0	1		
	0		1	0		
	0		1	1		
	1		0	0		
	1		0	1	关闭状态（输出使能且为无效电平） 异步地：OCx = CCxP，OCx_EN = 1，OCxN = CCxNP， OCxN_EN = 1；若存在时钟：经过一个死区时间后 OCx = OISx， OCxN = OISxN，假设 OISx 与 OISxN 并不都对应 OCx 和 OCxN 的有效电平	
	1		1	0		
	1		1	1		

9.5.9 计数寄存器（TIMERx_CNT）

偏移地址：0x20

复位值：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															

位	标记	功能描述
31:16	Reserved	保留位
15:0	CNT	计数器的值

9.5.10 分频寄存器（TIMERx_PSC）

偏移地址：0x24

复位值：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															

位	标记	功能描述
31:16	Reserved	保留位
15:0	PSC	预分频器的值 计数器的时钟频率（CK_CNT）等于 f_{CK_PSC} /（PSC[15:0] + 1）。 PSC 包含了当更新事件产生时装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清 0 或被工作在复位模式的从控制器清 0。

9.5.11 自动重装载寄存器（TIMERx_ARR）

偏移地址：0x28

复位值：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															

位	标记	功能描述
31:16	Reserved	保留位
15:0	ARR	自动重装载的值 ARR 包含了将要装载入实际的自动重装载寄存器的值。 详细参考 9.4.1：时基单元有关 ARR 的更新和动作。 当自动重装载的值为空时，计数器不工作。

9.5.12 重复计数寄存器（TIMERx_RCR）

偏移地址：0x2C

复位值：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							

位	标记	功能描述
31:8	Reserved	保留位
7:0	REP	周期计数器的值 开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率（即周期性地从预装载寄存器传输到当前寄存器）；如允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器 REP_CNT 达到 0，会产生一个更新事件并且计数器 REP_CNT 重新从 REP 值开始计数。由于 REP_CNT 只有在周期更新事件 UEV 发生时才重载 REP 值，因此对 TIMERx_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，（REP+1）对应着： — 在边沿对齐模式下，PWM 周期的数目； — 在中心对称模式下，PWM 半周期的数目。

9.5.13 捕获/比较寄存器 1（TIMERx_CCR1）

偏移地址：0x30

复位值：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															

位	标记	功能描述
31:16	Reserved	保留位
15:0	CCR1	捕获/比较 1 的值 若 CC1 通道配置为输出： CCR1 包含了装入当前捕获/比较 1 寄存器的值（预装载值）。 如果在 TIMERx_CCMR1 寄存器（OC1PE 位）中未选择预装载特性，其始终装入当前寄存器中。否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 1 寄存器中。当前捕获/比较寄存器包含了与计数器 TIMERx_CNT 比较的值，并且在 OC1 端口上输出信号。 若 CC1S 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件（IC1）传输的计数器值。

9.5.14 捕获/比较寄存器 2（TIMERx_CCR2）

偏移地址：0x34

复位值：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															

位	标记	功能描述
15:0	CCR2	捕获/比较 2 的值 若 CC2 通道配置为输出： CCR2 包含了装入当前捕获/比较 2 寄存器的值（预装载值）。 如果在 TIMERx_CCMR2 寄存器（OC2PE 位）中未选择预装载特性，其始

		终装入当前寄存器中。否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 2 寄存器中。当前捕获/比较寄存器包含了与计数器 TIMERx_CNT 比较的值，并且在 OC 端口上输出信号。 若 CC2S 通道配置为输入： CCR2 包含了由上一次输入捕获 2 事件（IC2）传输的计数器值。

9.5.15 捕获/比较寄存器 3（TIMERx_CCR3）

偏移地址：0x38

复位值：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															

位	标记	功能描述
15:0	CCR3	捕获/比较 3 的值 若 CC3 通道配置为输出： CCR3 包含了装入当前捕获/比较 3 寄存器的值（预装载值）。 如果在 TIMERx_CCMR3 寄存器（OC3PE 位）中未选择预装载特性，其始终装入当前寄存器中。否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 3 寄存器中。当前捕获/比较寄存器包含了与计数器 TIMERx_CNT 比较的值，并且在 OC 端口上输出信号。 若 CC3S 通道配置为输入： CCR3 包含了由上一次输入捕获 3 事件（IC3）传输的计数器值。

9.5.16 捕获/比较寄存器 4（TIMERx_CCR4）

偏移地址：0x3C

复位值：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															

位	标记	功能描述
15:0	CCR4	捕获/比较 4 的值 若 CC4 通道配置为输出： CCR4 包含了装入当前捕获/比较 4 寄存器的值（预装载值）。 如果在 TIMERx_CCMR4 寄存器（OC4PE 位）中未选择预装载特性，其始终装入当前寄存器中。否则，只有当更新事件发生时，此预装载值才装入当前捕获/比较 3 寄存器中。当前捕获/比较寄存器包含了与计数器 TIMERx_CNT 比较的值，并且在 OC 端口上输出信号。 若 CC4 通道配置为输入： CCR4 包含了由上一次输入捕获 4 事件（IC4）传输的计数器值。

9.5.17 刹车和死区寄存器（TIMERx_BDTR）

偏移地址：0x40

复位值：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							

位	标记	功能描述
31:16	Reserved	保留位
15	MOE	主输出使能 一旦刹车输入有效，该位被硬件异步清 0。根据 AOE 位的值，可由软件清 0 或自动置 1。它仅对配置为输出通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态； 1：如果设置了相应的使能位（TIMERx_CCER 寄存器的 CCxE、CCxNE 位），则开启 OC 和 OCN 输出
14	AOE	自动输出使能 0：MOE 只能被软件置 1； 1：MOE 能被软件置 1 或在下一个更新事件自动置 1（如果刹车输入无效）。 *注：一旦 LOCK 级别（TIMERx_BDTR 寄存器中的 LOCK 位）设为 1，则该位不能被修改。*
13	BKP	刹车输入极性 0：刹车输入低电平有效； 1：刹车输入高电平有效。 *注：一旦 LOCK 级别（TIMERx_BDTR 寄存器中的 LOCK 位）设为 1，则*

位	标记	功能描述
		该位不能被修改。
12	BKE	刹车功能使能 0：禁止刹车输入（BRK 及 BRK_ACTH）； 1：开启刹车输入（BRK 及 BRK_ACTH）。 *注：一旦 LOCK 级别（TIMERx_BDTR 寄存器中的 LOCK 位）设为 1，则该位不能被修改。*
11	OSSR	运行模式下“关闭状态”选择 （该位用于当 MOE = 1 时配置为输出模式且具有互补输出的通道。没有互补输出的定时器中不存在 OSSR 位。） 有关详细参考 OC/OCN 使能的详细说明（9.5.8 节，捕获/比较使能寄存器（TIMERx_CCER））。 0：当定时器不工作时，禁止 OCx/OCxN 输出（OCx/OCxN 使能输出信号等于 0）； 1：当定时器不工作时，一旦 CCxE = 1 或 CCxNE = 1，开启 OCx/OCxN 输出并输出无效电平。OCx/OCxN 使能输出信号等于 1。 *注：一旦 LOCK 级别（TIMERx_BDTR 寄存器中的 LOCK 位）设为 2，则该位不能被修改。*
10	OSSI	空闲模式下“态”选择 该位用于当 MOE = 0 且通道设为输出时。 参考 OCx/OCxN 使能的详细说明（12.5.9 节，捕获/比较使能寄存器（TIMERx_CCER））。 0：当定时器不工作时，禁止 OCx/OCxN 输出（OCx/OCxN 使能输出信号等于 0）； 1：当定时器不工作时，一旦 CCxE = 1 或 CCxNE = 1，OCx/OCxN 首先输出其空闲电平。OC/OCN 使能输出信号等于 1。 *注：一旦 LOCK 级别（TIMERx_BDTR 寄存器中的 LOCK 位）设为 2，则该位不能被修改。*
9:8	LOCK	锁定设置 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01：锁定级别 1，不能写入 TIMERx_BDTR 寄存器的 DTG/BKE/BKP/AOE 位、TIMERx_CR2 寄存器的 OISx/OISxN 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位（一旦相关通道通过 CCxS 位设为输出，TIMERx_CCER 寄存器的 CCxP/CCNxP 位）以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位（一旦相关通道通过 CCxS 位设为输出，TIMERx_CCMRx 寄存器的 OCxM/OCxPE 位）。 *注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMERx_BDTR 寄存器，则其内容冻结直至复位。*
7:0	DTG	死区发生器设置

位	标记	功能描述
		这些位定义了插入互补输出之间的死区持续时间。假设 DT 表示其持续时间： DTG[7:5] = 0xx => DT = DTG[7:0] × T_{dtg}，T_{dtg} = T_{DTS}； DTG[7:5] = 10x => DT = (64+DTG[5:0])× T_{dtg}，T_{dtg} = 2 × T_{DTS}； DTG[7:5] = 110 => DT = (32+DTG[4:0]) × T_{dtg}，T_{dtg} = 8 × T_{DTS}； DTG[7:5] = 111 => DT = (32+DTG[4:0]) × T_{dtg}，T_{dtg} = 16 × T_{DTS}； 例：若 T_{DTS} = 125ns (8MHZ)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns； 16us 到 31750ns，若步长时间为 250ns； 32us 到 63us，若步长时间为 1us； 64us 到 126us，若步长时间为 2us。 *注：一旦 LOCK 级别（TIMERx_BDTR 寄存器中的 LOCK 位）设为 1、2 或 3，则这些位不能被修改。*

9.5.18 Timer 时钟使能寄存器（TIMER_CLKEN）

绝对地址：0x3000_0100

复位值：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	

位	标记	功能描述
31:9	Reserved	保留位
8	timer2_clken_reg	timer2 时钟的控制位 0：关闭 timer2 的时钟，1：开启 timer2 的时钟
7:1	Reserved	保留位
0	timer1_clken_reg	timer1 时钟的控制位 0：关闭 timer1 的时钟，1：开启 timer1 的时钟

9.6 TIM1&TIM2 寄存器映射

TIMER1 寄存器列表

基地址：0x3000_0018

寄存器	偏移地址	描述
TIMER1_CR1	0x00	控制寄存器

TIMER1_ICF	0x04	滤波寄存器
TIMER1_IER	0x08	中断使能寄存器
TIMER1_SR	0x0C	状态寄存器
TIMER1_EGR	0x10	事件产生寄存器
TIMER1_CCMR1	0x14	捕获/比较模式寄存器 1
TIMER1_CCMR2	0x18	捕获/比较模式寄存器 2
TIMER1_CCER	0x1C	捕获/比较使能寄存器
TIMER1_CNT	0x20	计数寄存器
TIMER1_PSC	0x24	预分频寄存器
TIMER1_ARR	0x28	自动重装载寄存器
TIMER1_RCR	0x2C	重复向下计数器寄存器
TIMER1_CCR1	0x30	捕获/比较寄存器 1
TIMER1_CCR2	0x34	捕获/比较寄存器 2
TIMER1_CCR3	0x38	捕获/比较寄存器 3
TIMER1_CCR4	0x3C	捕获/比较寄存器 4
TIMER1_BDTR	0x40	刹车/死区寄存器

TIM2 寄存器列表

基地址：0x3000_0098

寄存器	偏移地址	描述
TIMER2_CR1	0x00	控制寄存器
TIMER2_ICF	0x04	滤波寄存器
TIMER2_IER	0x08	中断使能寄存器
TIMER2_SR	0x0C	状态寄存器
TIMER2_EGR	0x10	事件产生寄存器
TIMER2_CCMR1	0x14	捕获/比较模式寄存器 1
TIMER2_CCMR2	0x18	捕获/比较模式寄存器 2
TIMR2_CCER	0x1C	捕获/比较使能寄存器
TIMERE2_CNT	0x20	计数寄存器
TIMER2_PSC	0x24	预分频寄存器
TIMER2_ARR	0x28	自动重装载寄存器
TIMER2_RCR	0x2C	重复向下计数器寄存器
TIMER2_CCR1	0x30	捕获/比较寄存器 1
TIMER2_CCR2	0x34	捕获/比较寄存器 2
TIMER2_CCR3	0x38	捕获/比较寄存器 3
TIMER2_CCR4	0x3C	捕获/比较寄存器 4
TIMER2_BDTR	0x40	刹车/死区寄存器
TIMER_CLKEN	0x68	时钟使能寄存器

10 自动唤醒（WUP）

10.1 简介

WUP 模块是唤醒模块，时钟来源 3K，每隔 0.3ms 计数一次，当 wup_data = 0 时，不工作，无 irq 产生。wup_data 不等于 0 时每隔（wup_data+1）个时钟周期产生一个 irq，计数器重新装载 wup_data 的值。可以用于低功耗模式的唤醒等。

10.2 寄存器描述

10.2.1 wup 数据寄存器（wup_data）

偏移地址：0x00

复位值：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															

位	标记	功能描述
31:16	Reserved	保留位
15:0	wup_data	唤醒数据寄存器，wup_data 实际上是一个向下计数器

wup 模块是唤醒模块，实质是一个向下计数器。时钟来源 3k，每隔 0.3ms 计数一次，当 wup_data = 0 不计数，否则每隔（wup_data+1）个时钟周期产生一个中断。当使能了中断后，可以用于 pmu 的唤醒。来自 pmu 的复位不会使 irq 复位。

10.2.2 wup 中断使能寄存器（wup_irq_en）

偏移地址：0x04

复位值：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

位	标记	功能描述
31:1	Reserved	保留位
0	wup_irq_en	wup 中断使能位，1：使能，0：不使能。

10.2.3 wup 中断寄存器（wup_irq）

偏移地址：0x08

复位值：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

位	标记	功能描述
31:1	Reserved	保留位
0	wup_irq	中断标志位，1：产生中断，0：没有中断，写 0 清除中断。 当使能了中断后可以用于 pmu 的唤醒，来自 pmu 的复位不会使 irq 复位

10.3 寄存器映射

WUP 寄存器列表

基地址：0x3000_0610

寄存器	偏移地址	描述
wup_data	0x00	WUP 数据寄存器
wup_irq_en	0x04	WUP 中断使能寄存器
wup_irq	0x08	WUP 中断寄存器

11 模拟/数字转换（ADC）

11.1 简介

CSM32RV20 内置了 1 个快速、高精度 ADC，内部集成高精度 1.2 V 基准源，支持 13/14/15/16 位分辨率，在分辨率和转换速度之间得到平衡。ADC 工作时，VDD 电压要求大于 2.5 V。

注：1）推荐用户使用 ADC 时，将 ADC_CCR[5]写 1，否则会增大功耗；

2）分辨率出厂初始化，用户不可更改。

11.2 功能描述

- 分辨率为 13 位，需 29 个 ADC 时钟周期完成一次转换
- 分辨率为 14 位，需 45 个 ADC 时钟周期完成一次转换
- 分辨率为 15 位，需 77 个 ADC 时钟周期完成一次转换
- 分辨率为 16 位，需 141 个 ADC 时钟周期完成一次转换
- ADC 转换完成之后自动产生中断
- ADC 时钟与总线时钟具有相同的时钟源，支持 1/2/4/8 分频
- ADC 采样时钟推荐 4 MHz，最高不超过 8 MHz
- 支持单次模式和连续模式
- 连续模式下转换间隔可编程
- 支持软件触发和 GPIO 触发
- 可测量电压范围为 0 ~ VDD（VDD < 4.8 V）
- 支持外部基准
- 11 个测量通道可选，最多支持 9 个触摸按键
- 支持待测量电压乘以 1/4

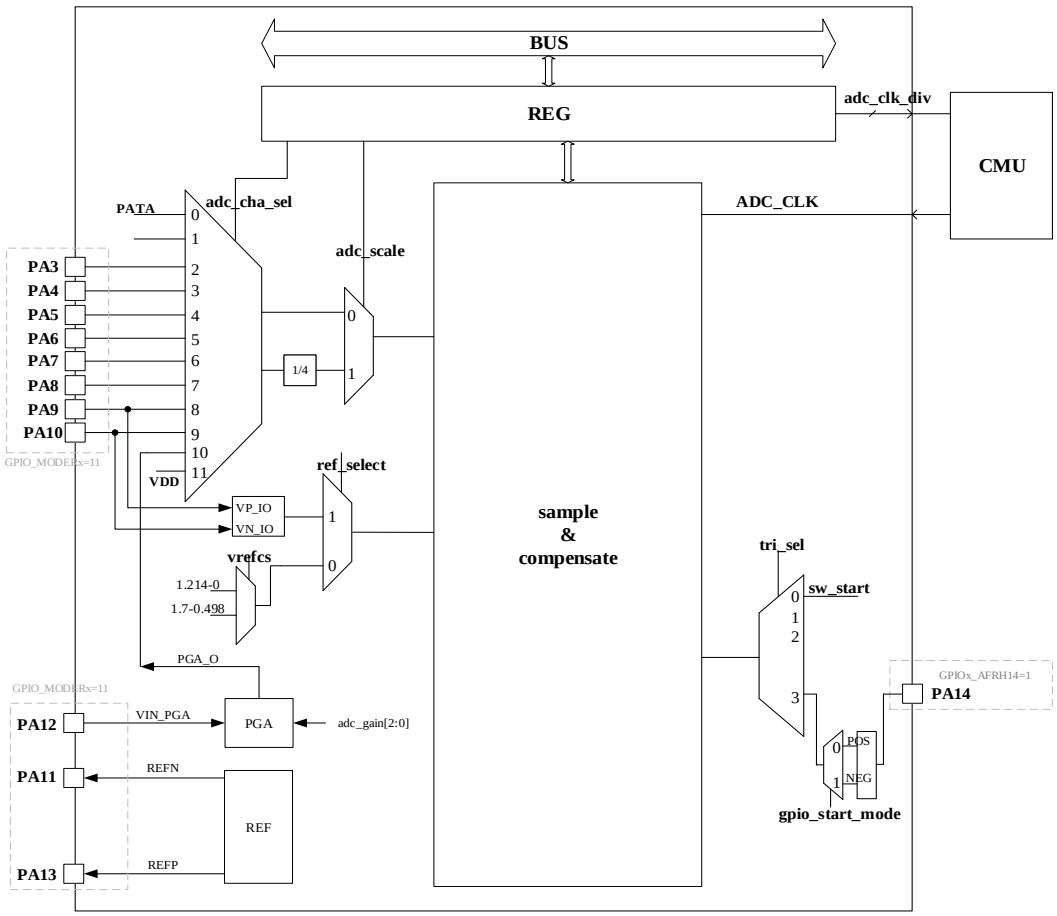


图 11-1　ADC 简图

adc_on置1后ADC开启，可以开始转换；在ADC在进入转换前，需要一段稳定时间t_{STAB}。当ADC进入转换状态时，adc_state位将置位；sw_start位写1将触发ADC进入转换。转换时间（用户程序设定的采样时间）结束后，eoc中断标志位拉高，ADC的转换结果将存储在ADC_DR寄存器中。注意信号从HCLK时钟域传输到ADCCLK时钟域时需要重新同步，从而产生延时。

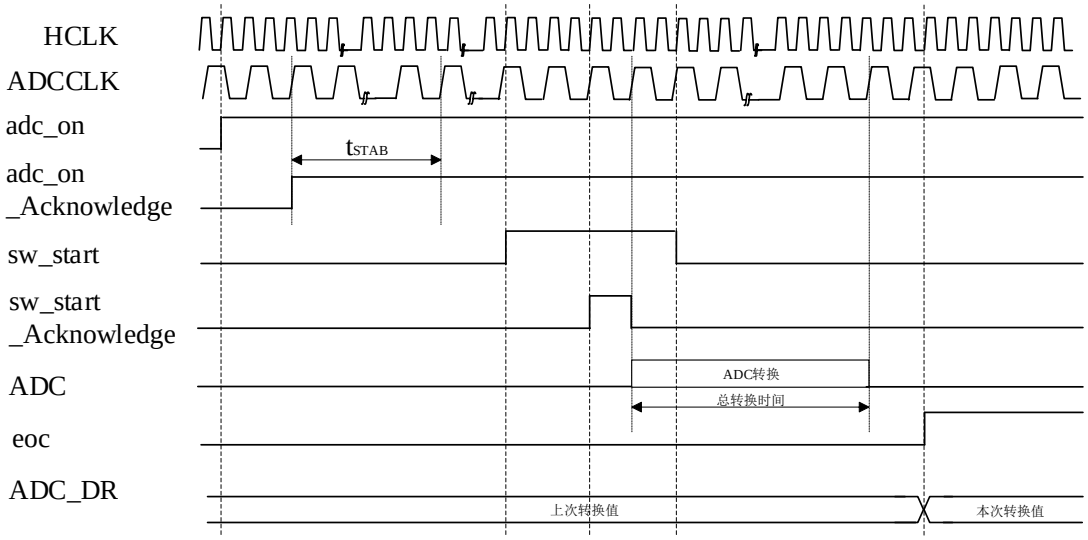


图 11-2 ADC 时序图

11.3 寄存器描述

11.3.1 ADC 状态寄存器（ADC_ISR）

基地址：0x3000_0280

偏移地址：0x00

复位值：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

位	标记	功能描述
31:3	Reserved	保留位
2	eoc	ADC 转换工作是否完成的标志位 0：转换尚未完成； 1：转换采集已完成。 对其写 0 清除中断，读数据寄存器也可清除中断
1	adc_state	ADC 转换的工作状态标志位 1：ADC 正在转换；　　0：ADC 处于空闲状态
0	adc_ready	ADC 开启状态标志位 1：ADC 处于开启状态；　0：ADC 处于关闭状态。 adc_on 开启 5 us，电源稳定后 adc_ready 拉高

11.3.2 ADC 中断控制寄存器（ADC_IER）

偏移地址：0x04

复位值：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		

位	标记	功能描述
31:3	Reserved	保留位
2	eocie	当前 ADC 的中断使能位 0：ADC 不会产生中断传给 CLIC； 1：ADC 会在 EOC 信号被拉高的时候产生中断传给 CLIC
1:0	Reserved	保留位

11.3.3 ADC 控制寄存器（ADC_CR）

偏移地址：0x08

复位值：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

位	标记	功能描述
31:1	Reserved	保留位
0	sw_start	ADC 软件触发控制位 单次模式：写 1 开始转换，1 次转换完成后自动清零同时产生中断和更新 ADC_DR，写 0 提前结束。 连续模式：写 1 开始转换，转换完成后产生中断和更新 ADC_DR 并开始下一次转换，写 0 结束转换。

11.3.4 ADC 通道选择寄存器（ADC_SEL）

偏移地址：0x0C

复位值：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			

位	标记	功能描述

31:4	Reserved	保留位
3:0	adc_cha_sel	通道选择 下列情况 GPIO 需配置模拟模式（GPIO_MODEx = 2'b11） 0000：测量内部 PTAT 电压值，可用于芯片温度采集； 0001：保留； 0010：测量通道 PA3； 0011：测量通道 PA4； 0100：测量通道 PA5； 0101：测量通道 PA6； 0110：测量通道 PA7； 0111：测量通道 PA8； 1000：测量通道 PA9； 1001：测量通道 PA10； 1010：测量外部热敏电阻 NTC 中 PGA 输入，需配置 PA11/PA12/PA13； 1011：VDD，选择此通道时，ADC_CCR[19]必须配置为 1； 其他：保留。

11.3.5 ADC 数据寄存器（ADC_DR）

偏移地址：0x10

复位值：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														

位	标记	功能描述
31:15	Reserved	保留位
14:0	data	ADC 采集到的数据，当 eoc 信号被拉高的时候，软件可以从这个寄存器中读取 ADC_CFG[adc_bits_ctrl]控制相应分辨率的数据。

11.3.6 ADC 通用控制寄存器（ADC_CCR）

偏移地址：0x14

复位值：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
del			adc_clk_div		shd_vsample	gpio_start_mode	tri_sel	adc_mode	adc_on
RW			RW		RW	RW	RW	RW	RW

位	标记	功能描述
31:22	Reserved	保留位
21:20	Reserved	必须为 00
19	adc_scale	选择 ADC 内部通道增益 0：选择 1； 1：选择 1/4，输入电压乘以 1/4，使检测电压在量程范围内
18:16	pga_gain	用于测量 PGA 输入 PGA 运算放大器输入为 PA12（需要设置成模拟模式） 000：运算放大器增益为 1（默认）； 001：运算放大器增益为 2； 010：运算放大器增益为 4； 011：运算放大器增益为 8； 100：运算放大器增益为 16； 101：运算放大器增益为 32； 110：运算放大器增益为 64； 111：运算放大器增益为 128
15:14	Reserved	保留位
13	vrefcs	选择内部基准电压来源 1：基准为 1.214 V ~ 0；　　0：基准为 1.7 V ~ 0.498 V
12	ref_select	选择内部或者外部基准 1：外部基准；　　0：　内部基准
11:8	delay_sel	ADC 连续转换模式下相邻两次转换之间的延迟 0000：不延迟； 0001：2^0 个 ADC clock； 0010：2^1 个 ADC clock； … 1111：2^{14} 个 ADC clock
7:6	adc_clk_div	ADC 时钟分频选择位 00：不分频；　　01：2 分频；　　10：4 分频；　　11：8 分频。 注：推荐在 adc_on 为 0 时设置 adc_clk_div。

5	Reserved	保留
4	gpio_start_mode	gpio 触发模式选择 0：GPIO 上升沿触发 单次模式：上升沿触发一次； 连续模式：上升沿触发，下降沿结束采样。 1：GPIO 下降沿触发 单次模式：下降沿触发一次； 连续模式：下降沿触发，上升沿结束采样
3:2	tri_sel	ADC 触发信号来源选择 00：ADC_CR[0]软件触发控制 ADC 转换； 01/10：保留； 11：GPIO 触发 ADC 转换
1	adc_mode	ADC 采样模式 0：单次模式　　1：连续模式
0	adc_on	ADC 电源开关 0：关闭 ADC　　1：开启 ADC

11.3.7 ADC 分辨率寄存器（ADC_CFG）

地址：0x3000_0410

复位值：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															

位	标记	功能描述
31:30	adc_bits_ctrl	ADC 分辨率控制位 11：ADC 分辨率 16 位，数据 15 位； 10：ADC 分辨率 15 位，数据 14 位； 01：ADC 分辨率 14 位，数据 13 位； 00：ADC 分辨率 13 位，数据 12 位
29:21	Reserved	保留位
20:0	adc_data	ADC 数据，以补码形式存放

11.4 寄存器映射

ADC 寄存器列表

基地址：0x3000_0280

寄存器	偏移地址	描述
ADC_ISR	0x00	ADC 状态寄存器
ADC_IER	0x04	ADC 中断使能寄存器
ADC_CR	0x08	ADC 控制寄存器
ADC_SEL	0x0C	ADC 通道选择寄存器
ADC_DR	0x10	ADC 数据寄存器
ADC_CCR	0x14	ADC 通用控制寄存器
ADC_CFG	0x3000_0410	ADC 分辨率寄存器

12 I2C 接口

12.1 介绍

I2C 总线接口是单片机与串行 I2C 总线之间的接口。它提供主设备功能，并控制所有 I2C 总线特定的排序、协议、仲裁和定时。它支持标准模式（达到 100 kHz）和快速模式（达到 400 kHz）。

12.1.1 主要特点

- 并行总线/I2C 总线协议转换器
- I2C 主设备功能
  - 产生时钟
  - 产生起始和停止信号
- 产生和检测 7 位地址和广播呼叫
- 支持不同的通讯速度
  - 标准速度(高至 100 kHz)
  - 快速(高至 400 kHz)
- 状态标志：
  - 发送器/接收器模式标志
  - 字节发送结束标志
  - I2C 总线忙标志
- 错误标志
  - 主模式时的仲裁丢失
  - 地址/数据传输后的应答(ACK)错误
  - 检测到起始和停止错位
- 2 个中断向量
  - 1 个中断用于地址/数据通讯成功
  - 1 个中断用于出错

12.2功能描述

主模式时，I2C 接口启动数据传输并产生时钟信号。串行数据传输总是以起始条件开始和停止条件结束。只有在总线处于“非忙”状态时，数据传输才能开

始。在数据传输期间，只要时钟线为高电平，数据线都必须保持稳定，否则数据线上的任何变化都被当作“启动”或“停止”信号。图 12-1 是被定义的总线状态。

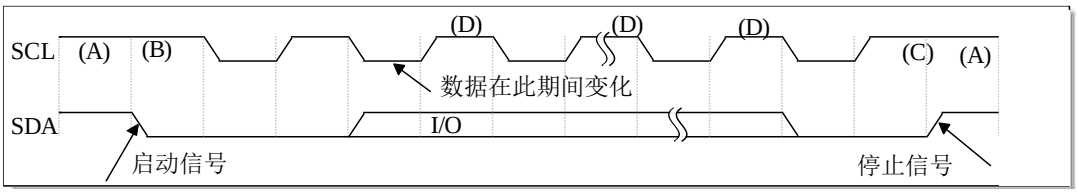


图 12-1　I2C 二线制串行总线

I2C 主要有以下 A，B，C，D 四段的工作状态：

（1）总线非忙状态（A 段）：该段内的数据线（SDA）和时钟线（SCL）都保持高电平。

（2）启动数据传输（B 段）：当时钟线（SCL）为高电平状态时，数据线（SDA）由高电平变为低电平的下降沿被认为是“启动”信号。只有出现“启动”信号后，其它的命令才有效。

（3）停止数据传输（C 段）：当时钟线（SCL）为高电平状态时，数据线（SDA）由低电平变为高电平的上升沿被认为是“停止”信号。随着“停止”信号的出现，所有的外部操作都结束。

（4）数据有效（D 段）：在出现“启动”信号后，在时钟线（SCL）为高电平状态时，数据线是稳定的，这时数据线的状态就是要传送的数据。数据线（SDA）上数据的改变必须在时钟线为低电平期间完成，每位数据占用一个时钟脉冲。每个数据传输都是由“启动”信号开始，结束于“停止”信号。

（5）应答信号：在接收到一个字节的数据后，通常需要发出一个应答信号。 在发出一个字节的数据后，通常需要接收一个应答信号。发送方在应答时钟脉冲期间释放 SDA 线，接收方拉低 SDA 线，并且在 SCL 的高脉冲期间保持为 0，如图 12-2 所示。I2C 读写控制器必须有产生一个与这个应答位相联系的额外的时钟脉冲。在读操作中，读写控制器对 I2C 完成的最后一个字节不产生应答位，但是会有一个结束信号。

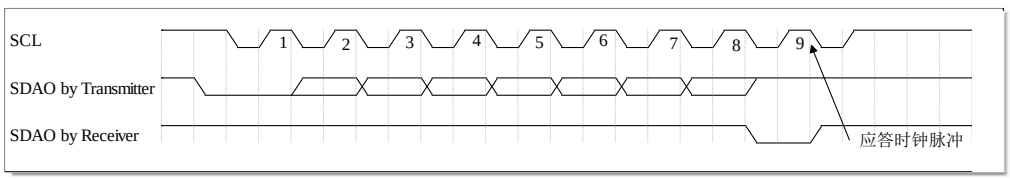


图 12-2　I2C 总线上应答时序图

写操作配置步骤：

1) 通过 fm 位，定义数据传输的频率。

2) 通过 I2C_CTRL[i2c_clkdiv]选择是否分频，I2C_CTRL[i2c_saddr]定义从机地址。

3) I2C_CTRL[i2c_r_wn] = 0 被定义为写操作；I2C_CTRL[i2c_maddr]定义存储单元地址，I2C_CTRL[i2c_data]定义要发送的数据。

发送方：满足启动条件后，从 SDA 线先后串行写入 I2C_CTRL[i2c_saddr], I2C_CTRL[i2c_r_wn] = 0 写操作，应答，I2C_CTRL[i2c_maddr], 应答和 I2C_CTRL[i2c_data]、应答，最后是停止信号。参考图 12-3 I2C 写数据时序图。

每次应答只有收到接收方正确回复应答信号才会继续往下执行，否则一直在等待接收方的应答。

数据发送完成后 i2c_busy 标志将被置位，如果设置 I2C_CTRL 寄存器中的 i2c_ready_en 位，将产生中断。

读操作配置步骤：

1) 通过 fm 位，定义数据传输的频率。

2) 通过 I2C_CTRL[i2c_clkdiv] 选择是否分频，I2C_CTRL[i2c_saddr] 定义从机地址。

3) I2C_CTRL[i2c_r_wn] = 0 被定义为写操作，I2C_CTRL[i2c_r_wn] = 1 被定义为读操作；I2C_CTRL[i2c_maddr]定义存储单元地址，读取 I2C_CTRL[i2c_data]数据。

满足启动条件后，从 SDA 线先后串行写入 I2C_CTRL[i2c_saddr]，I2C_CTRL[i2c_r_wn] = 0 写操作，应答信号，I2C_CTRL[i2c_maddr]，应答。重新启动，I2C_CTRL[i2c_saddr]，I2C_CTRL[i2c_r_wn] = 1 读操作，应答。接收方回复读取的数据 I2C_CTRL[i2c_data]，非应答，最后是停止信号。参考图 12-4 和图 12-5。

数据接收完成后 i2c_busy 标志将被置位，如果设置 I2C_CTRL 寄存器中的 i2c_ready_en 位，将产生中断。读 I2C_DATA 寄存器时，I2C 设备返回接收到的数据。读 I2C_DATA 寄存器将清除 i2c_ready 位。

在数据传输过程中，若没有应答或产生其他错误，i2c_error 标志将被置位，如果设置 I2C_CTRL 寄存器中的 i2c_error_en 位，将产生中断。图 12-2 是 I2C 应答时序图，图 12-3 和图 12-4 分别是 I2C 的写数据和读指定地址数据的时序图。

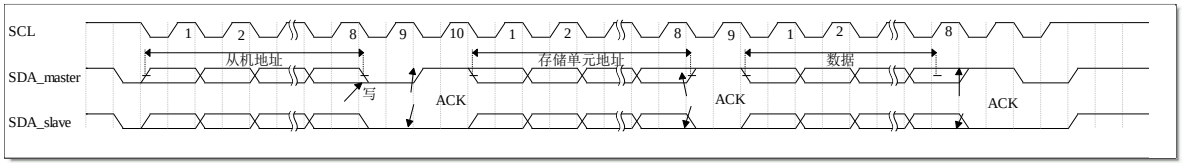


图 12-3　I2C 写数据时序图

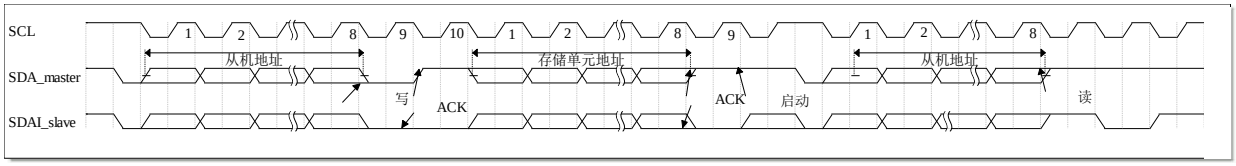


图 12-4　I2C 读指定地址数据的时序图 1

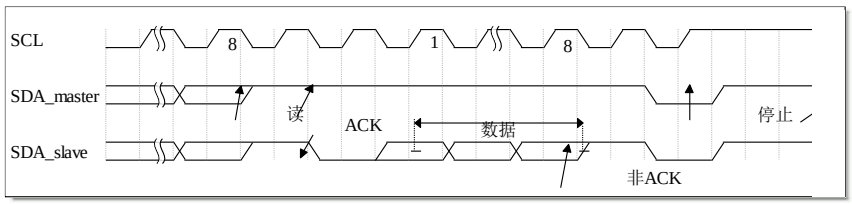


图 12-5　I2C 读指定地址数据的时序图 2（时序接着上图）

12.3 I2C 寄存器描述

基址：0x3000_0004

12.3.1 状态寄存器（I2C_STATUS）

偏移地址：0x00

复位值：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

位	标记	功能描述
16:5	Reserved	保留位
4	i2c_error	错误标志位；1：i2c 发生错误，0：正常工作
3:1	Reserved	保留位
0	i2c_ready	中断标志位；1：操作完成，0：正在进行操作

12.3.2 控制寄存器（I2C_CTRL）

偏移地址：0x04

复位值：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						

位	标记	功能描述
31:13	Reserved	保留位
12	i2c_clkdiv	0：不分频； 1：二分频
11:10	Reserved	保留位
9	i2c_error_en	i2c_error 标志位使能端，0：关闭中断，1：开启中断
8	i2c_ready_en	i2c_ready 标志位使能端，0：关闭中断，1：开启中断
7	fm	时钟频率，0：100KHZ 1：400KHZ
6:0	i2c_saddr	从机地址

12.3.3 数据寄存器（I2C_DATA）

偏移地址：0x08

复位值：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							

位	标记	功能描述
31:17	Reserved	保留位
16	i2c_r_wn	0：写操作； 1：读操作
15:8	i2c_maddr	存储单元地址
7:0	i2c_data	i2c 数据

12.4 寄存器映射

I2C 寄存器列表

基地址：0x3000_0004

寄存器	偏移地址	描述
I2C_STATUS	0x00	I2C 状态寄存器
I2C_CTRL	0x04	I2C 控制寄存器
I2C_DATA	0x08	I2C 数据寄存器

13 串行外设接口（SPI1）

13.1 简介

SPI，是 Serial Peripheral Interface 的缩写，顾名思义就是串行外围设备接口。串行外设接口（SPI）允许芯片与外部设备以半/全双工、同步、串行方式通信。此接口仅支持主模式，这种工作模式下，它要为外部从设备提供通信时钟（SCK）。接口还能以多主配置方式工作。

它可用于多种用途，包括可选第三根双向数据线的双线单工同步传输，或使用 CRC 校验的可靠通信。

13.1.1 主要特征

- 3 线全双工同步传输
- 8 位传输帧格式
- 支持多主模式
- 8 个主模式波特率预分频系数
- 可编程的时钟极性和相位
- 可编程的数据顺序，MSB 在前
- 可触发中断的专用发送和接收标志
- SPI 总线忙状态标志

13.2 功能描述

通常，SPI 通过 4 个引脚和外部设备相连。

- MISO：主入/从出数据口。此脚可以被用来在从模式中发送数据，在主模式中接收数据。
- MOSI：主出/从入数据口。此脚可以用来在主模式时发送数据，在从模式时接收数据。
- SCK：SPI 主设备输出串行时钟，SPI 从设备输入串行时钟。
- CSN：由 GPIO 模拟。选择主/从模式的可选引脚。SPI 主设备和从设备分别通信时，该引脚起到依次片选各个从设备的作用，以避免发生数据线冲突。从设备的 CSN 输入可以由主设备上的标准 I/O 端口驱动。如果使能 SPI，则 SPI 工作在主设备，

CSN 引脚用作输出，并输出低电平。

通信总是由主设备发起。主设备通过 MOSI 脚把数据发送给从设备，从设备通过 MISO 引脚回传数据。这意味全双工通信的数据输出和数据输入是用同一个时钟信号同步的；时钟信号由主设备通过 SCK 脚提供。

使用 SPI_CTRL 寄存器的 CPOL 和 CPHA 位，组合成四种可能的时序关系。CPOL（时钟极性）位控制在没有数据传输时时钟的空闲状态电平，此位对主模式和从模式下的设备都有效。如果 CPOL 被复位，SCK 引脚在空闲状态保持低电平；如果 CPOL 被置位，SCK 引脚在空闲状态保持高电平。

如果 CPHA（时钟相位）位被置位，SCK 时钟的第二个边沿（CPOL 位为 0 时就是下降沿，CPOL 位为 1 时就是上升沿）进行数据位的采样。数据在第一个时钟边沿被锁存。如果 CPHA 位被复位，SCK 时钟的第一边沿（CPOL 位为 0 时就是下降沿，CPOL 位为 1 时就是上升沿）进行数据位采样。数据在第二个时钟边沿被锁存。

CPOL 时钟极性和 CPHA 时钟相位的组合选择数据捕捉的时钟边沿。图 13-1、图 13-4 显示了 SPI 传输的 4 种 CPHA 和 CPOL 位组合。

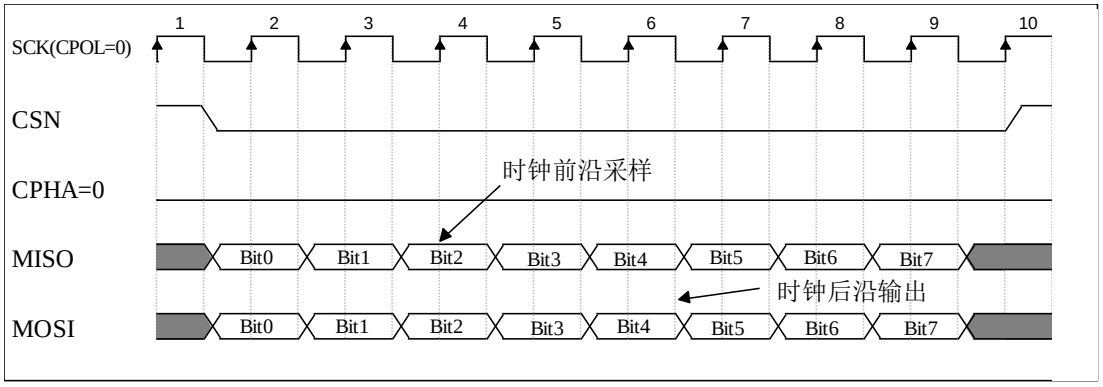


图 13-1　spi 模式 0 时序图

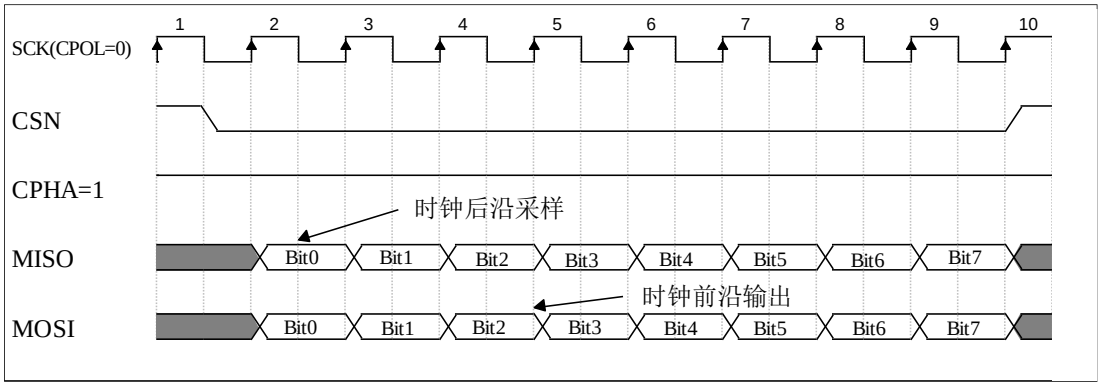


图 13-2　spi 模式 1 时序图

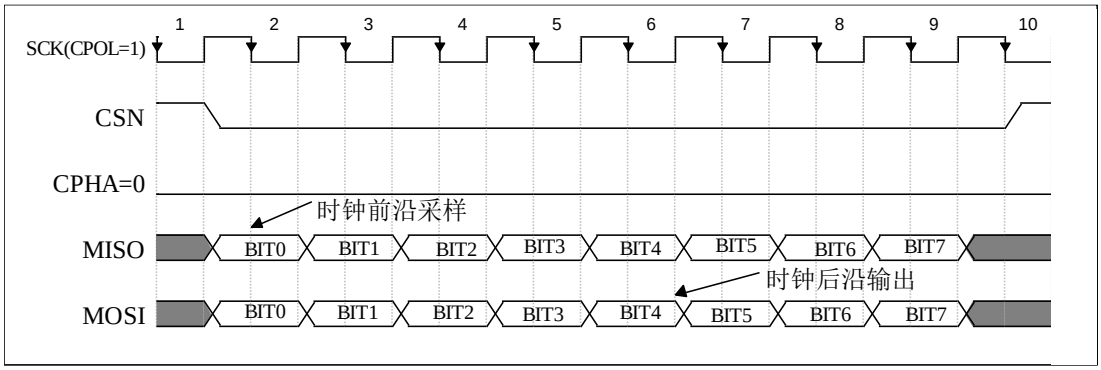


图 13-3　spi 模式 2 时序图

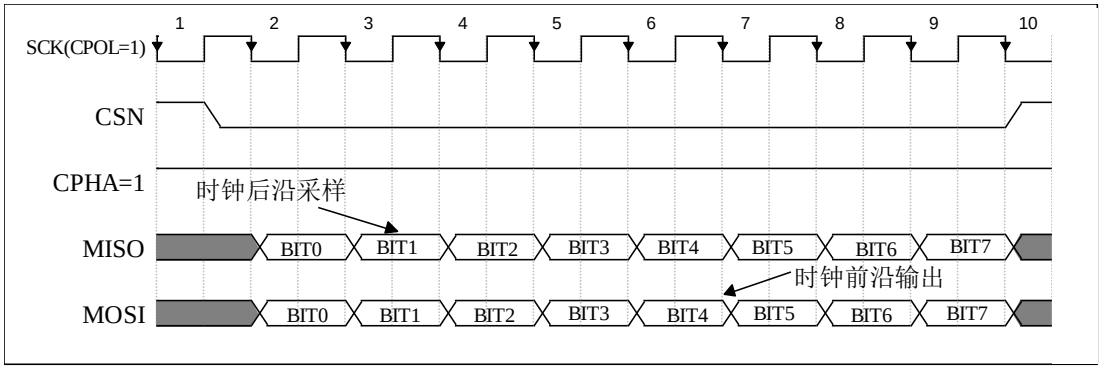


图 13-4　spi 模式 3 时序图

在主配置时，串行时钟在 SCK 脚产生。MOSI 脚是数据输出，而 MISO 脚是数据输入。

配置例程：

1. 通过 SPI_CTRL[smctrl 3:smctrl 0]定义串行时钟波特率。
2. 选择 CPOL 和 CPHA 位，定义数据传输和串行时钟间的相位关系。
3. 通过 SPI_CTRL[smctrl 5:smctrl 4]位选择是否开启 SPI。

数据发送过程：

当一字节写进发送缓冲器时，发送过程开始。

在发送第一个数据位时，数据字被并行地传入移位寄存器，而后串行地移出到 MOSI 脚上；MSB 在先。数据从发送缓冲器传输到移位寄存器时 spi_int 标志将被置位，如果设置 SPI_CTRL[spi_int_en] = 1，将产生中断。

数据接收过程：

对于接收器来说，当数据传输完成时：

- 移位寄存器里的数据传送到接收缓冲器，并且 spi_int 标志被置位。
- 如果设置 SPI_CTRL[spi_int_en] = 1，则产生中断。

读 SPI_DATA 寄存器时，SPI 设备返回接收到的数据字。

13.3 寄存器描述

基址：0x3000_0060

13.3.1 控制寄存器（SPI1_CTRL）

偏移地址：0x00

复位值：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

位	标记	功能描述
15:9	Reserved	保留位
8	spi_int_en	spi1 中断使能。0：关闭中断，1：开启中断
7	CPOL	时钟极性 0：SCK 默认低电平　1：　SCK 默认高电平
6	CPHA	时钟相位 0：时钟前沿采样，时钟后延输出 1：时钟前沿输出，时钟后延采样
5:4	smctrl[5:4]	01：使能 SPI；　00：关闭 SPI
3:0	smctrl[3:0]	从时钟到 SPI 时钟的分频数 0000：cclk/2； 0001：cclk/2； 0010：cclk/4； 0011：cclk/8； 0100：cclk/16； 0101：cclk/32； 0110：cclk/64； 其它：cclk/64

13.3.2 数据寄存器（SPI1_DATA）

偏移地址：0x04

复位值：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							

位	标记	功能描述
31:8	Reserved	保留位
7:0	spi_data	SPI 数据写发送的数据/读接收的数据

13.3.3 状态寄存器（SPI1_STATUS）

偏移地址：0x08

复位值：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

位	标记	功能描述
31:5	Reserved	保留位
4	spi_int	spi1 中断标志位，1：spi 操作完成。写 0 清中断
3:1	Reserved	保留位
0	spi_busy	spi1 工作标志位，1：正在进行操作；　0：操作完成或未操作

13.4 寄存器映射

SPI1 寄存器列表

基地址：0x3000_0060

寄存器	偏移地址	寄存器描述
SPI1_CTRL	0x00	SPI1 控制寄存器

SPI1_DATA	0x04	SPI1 数据寄存器
SPI1_STATUS	0x08	SPI1 状态寄存器

14 串行外设接口（SPI2）

14.1 简介

SPI2 除了具有 SPI1 的功能以外，还能作为无线 ISP 接口，无线 ISP 需外接 Si24R1。和 Si24R1 通信时，CE 和 CSN 分别由 PB0、PB1 模拟。

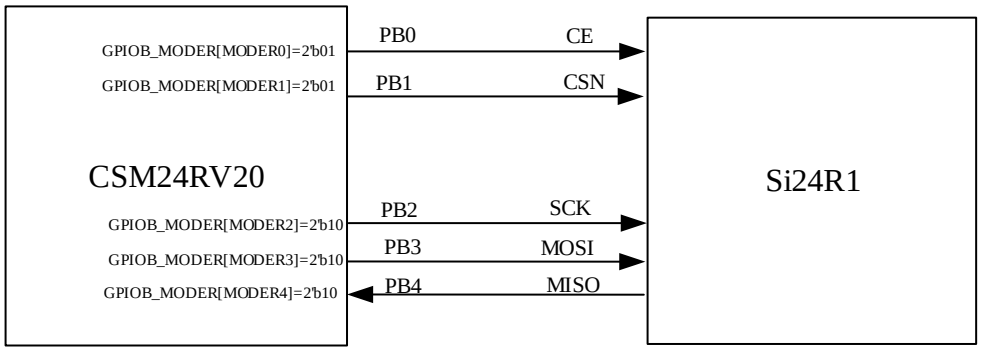


图 14-1　SPI 通信接口

14.1.1 主要特征

- 3 线全双工同步传输
- 8 位传输帧格式
- 支持多主模式
- 8 个主模式波特率预分频系数（最大为 fPCLK/2）
- 可编程的时钟极性和相位
- 可编程的数据顺序，MSB 在前
- 可触发中断的专用发送和接收标志
- SPI 总线忙状态标志
- 作为无线 ISP 接口

14.2 功能描述

通常，SPI 通过 4 个引脚和外部设备相连。

- MISO：主入/从出数据口。此脚可以被用来在从模式中发送数据，在主模式中接收数据。
- MOSI：主出/从入数据口。此脚可以用来在主模式时发送数据，在从模式时接收数据。
- SCK：SPI 主设备输出串行时钟，SPI 从设备输入串行时钟。

- CSN：由 R1_CSN 控制

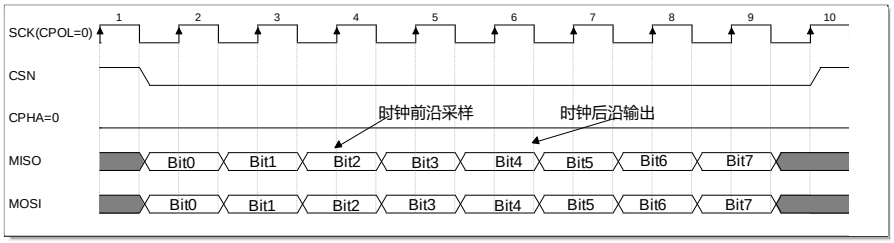


图 14-2　spi 模式 0 时序图

在主配置时，串行时钟在 SCK 脚产生。MOSI 脚是数据输出，而 MISO 脚是数据输入。

14.3 寄存器描述

基址：0x3000_0070

14.3.1 控制寄存器（SPI2_CTRL）

偏移地址：0x00

复位值：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

位	标记	功能描述
31:9	Reserved	保留位
8	spi_int_en	Spi2 中断使能。0：关闭中断；　1：开启中断
7	CPOL	时钟极性 0：SCK 默认低电平；　1：　SCK 默认高电平
6	CPHA	时钟相位 0：时钟前沿采样，时钟后延输出； 1：时钟前沿输出，时钟后延采样
5:4	smctrl[5:4]	01：使能 SPI；　00：关闭 SPI

3:0	smctrl[3:0]	从时钟到 SPI 时钟的分频数 0000：cclk/2； 0001：cclk/2； 0010：cclk/4； 0011：cclk/8； 0100：cclk/16； 0101：cclk/32； 0110：cclk/64； 其它：cclk/64

14.3.2 数据寄存器（SPI2_DATA）

偏移地址：0x04

复位值：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							

位	标记	功能描述
31:8	Reserved	保留位
7:0	spi_data	SPI 数据写发送的数据/读接收的数据

14.3.3 状态寄存器（SPI2_STATUS）

偏移地址：0x08

复位值：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

位	标记	功能描述
31:5	Reserved	保留位
4	spi_int	spi2 中断标志位，1：spi 操作完成。写 0 清中断
3:1	Reserved	保留位
0	spi_busy	spi2 工作标志位，1：正在进行操作　0：操作完成或未操作

14.4 寄存器映射

SPI2 寄存器列表

基地址：0x3000_0070

寄存器	偏移量	描述
SPI2_CTRL	0x00	SPI2 控制寄存器
SPI2_DATA	0x04	SPI2 数据寄存器
SPI2_STATUS	0x08	SPI2 状态寄存器

15 异步收发器（UART）

15.1 简介

通用异步收发器（UART）提供了一种灵活的方法来与使用工业标准 NRZ 异步串行数据格式的外部设备之间进行全双工数据交换。UART 利用分数波特率发生器提供宽范围的波特率选择。

芯片内置四个 UART，UART1 支持 ISP 下载程序。

15.1.1 主要特性

- 全双工的，异步通信
- NRZ 标准格式
- 分数波特率发生器系统
- 支持波特率自适应
- 可编程数据字长度（8 位或 9 位）
- 单线半双工通信
- 单独的发送器和接收器使能位
- 检测标志
- 传输结束标志
- 多处理器通信

15.2 功能描述

串口由 S0CON 控制，而实际传输的数据则在 S0BUF 寄存器中读取或写入。传输速度（波特率）是通过配置 uartdiv 来选择的。

1) 同步模式，固定波特率。
2) 8 位 UART 数据模式，波特率可变。
3) 9 位 UART 数据模式，波特率可变。

表 15-1 常用速率 uartdiv 值对应表（f_{CK} = 16Mhz，fck 为总线时钟）

序号	波特率（Kpbs）	uartdiv
1	2.4	0x1A0B
2	9.6	0x0683
3	19.2	0x0341

序号	波特率（Kpbs）	uartdiv
4	57.6	0x0116
5	115.2	0x008B
6	230.4	0x0045
7	460.8	0x0023
8	921.6	0x0011
9	1228.8	0x000D

串口支持波特率自适应，通过测出 RX 引脚上接收信号的波特率并将其配置到波特率寄存器中实现。使用方法如下：

1) 配置 MCU 和外设使用同一个时钟来源；
   （设置时钟源选择寄存器（CMU_CLK_SEL），选择 MCU 和外设的时钟源。）
2) 配置 baudtrim = 1，trim_en 写 0；
3) 配置 baudtrim = 1，trim_en 写 1；
4) RX 接收 UART 帧，帧中的低电平只能是 1 位宽；
5) 等到 trim_en 变为 0，读出 trim_clk_result 的结果；
6) 使用 trim_clk_result 作为 uart 的波特率设置。

15.3 UART 的引脚映射

	引脚	描述	复用配置
UART1	PA6	TX1	AF0（默认）
	PA5	RX1	AF0（默认）
UART2	PA4	TX2	AF3
	PA3	RX2	AF3
UART3	PA11	TX3	AF3
	PA10	RX3	AF3
UART4	PA15	TX4	AF3
	PA14	RX4	AF3

15.4 寄存器描述

UART1 基址：0x3000_0010
UART2 基址：0x3000_0700
UART3 基址：0x3000_0800
UART4 基址：0x3000_0900
波特率自适应基址：0x3000_0380

15.4.1 控制寄存器（UART_CTRL）

偏移地址：0x00

复位值：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							

位	标记	功能描述
31:26	Reserved	保留位
25	tx_int_en	发送中断使能位，0：关闭中断，1：开启中断
24	rx_int_en	接收中断使能位，0：关闭中断，1：开启中断
23	tx_busy	发射忙，1：正在发射
22	Res	保留位
21:8	uartdiv	波特率设置位，21：12 表示整数，11：8 表示小数， Tx/ Rx baud = fCK/(16 * uartdiv) ，uartdiv = uartdiv[13:4] + uart[3:0]/16；
7:0	s0_con	7：6：uart 模式选择 00：模式 0，移位寄存器，波特率是 cclk/12； 01：模式 1，8 位数据，波特率由 uartdiv 控制； 10/11：模式 2，9 位数据，波特率由 uartdiv 控制
		5：多处理器通信使能端
		4：串口接收使能，0：关闭接收；　1：打开接收
		3：发送数据 bit8：在模式 2 和模式 3，传输 9 位数据时，对应于发送数据第 9 位的状态，由软件控制
		2：接收数据 bit8：在模式 2 和模式 3，传输 9 位数据时，对应于接收数据第 9 位的状态
		1：发送中断标志，标志着串口发送数据的完成。在模式 0 的第 8 位数据结束时或者在其他模式中停止位开始前被用硬件置 1，该位必须由软件写 1 清除
		0：接收中断标志，在完成接收数据后由硬件置 1。在模式 0 的第 8 位数据结束时或者在其他模式中停止位开始前被用硬件置 1，该位由软件写 1 清除，不清除接收中断不能继续接收。

15.4.2 数据寄存器（UART_DATA）

偏移地址：0x04

复位值：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							

位	标记	功能描述
31:8	Reserved	保留位
7:0	uart_datain	写发送数据/读接收到的数据

15.4.3 波特率自适应配置寄存器（AUTOBPS_CONFIG）

地址：0x3000_0380

复位值：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		

位	标记	功能描述
31:9	Reserved	保留位
8	trim_en	启动波特率自适应，0：关闭，1：启动。trim 结束后硬件自动清零
7	Reserved	保留位
6	baudtrim	1：使能波特率适应功能，能够计算 uart 外部输入的波特率
5:3	Reserved	保留位
2:1	uart_rx_sel	00：选择的时钟 uart1 的 rx； 01：选择的时钟 uart2 的 rx； 10：选择的时钟 uart3 的 rx； 11：选择的时钟 uart4 的 rx。 注意：baudtrim 设置为 0 时，uart_rx_sel 的设置才生效
0	Reserved	保留位

15.4.4 波特率自适应结果寄存器（AUTOBPS_RESULT）

地址：0x3000_0384

复位值：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													

位	标记	功能描述
31:14	Reserved	保留位
13:0	trim_clk_result	trim 结束后显示为结果

15.5 寄存器映射

Uart1 基地址：0x3000_0010

Uart2 基地址：0x3000_0700

Uart3 基地址：0x3000_0800

Uart4 基地址：0x3000_0900

寄存器	偏移地址	描述
UARTx_CTRL	0x00	UART 控制寄存器
UARTx_DATA	0x04	UART 数据寄存器

波特率自适应寄存器基地址：0x3000_0380

寄存器	偏移地址	描述
AUTOBPS_CONFIG	0x00	波特率自适应配置寄存器
AUTOBPS_RESULT	0x04	波特率自适应结果寄存器

16 低压检测（LV）

16.1 简介

低压检测模块检测电源电压，电压较低时，低压检测中断变高，如果使能了中断，中断可以传递到 CLIC 产生欠压中断。

16.2 寄存器描述

16.2.1 低压检测中断使能寄存器（LV_IRQ_EN）

地址：0x3000_0330

复位值：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

位	标记	功能描述
31:1	Reserved	保留位
0	lv_irq_en	低压检测中断使能，1：使能，0：不使能

16.2.2 低压检测中断寄存器（LV_IRQ）

地址：0x3000_0334

复位值：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

位	标记	功能描述
31:1	Reserved	保留位
0	lv_irq	低压检测中标志位，1：产生低压中断，0：未产生中断。当电源电压高于设定阈值时，硬件自动清除该中断标志。

16.2.3 低压阈值寄存器（LV_TH）

地址：0x3000_0400

复位值：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															

位	标记	功能描述
31:29	Reserved	保留位
28:25	lv_th	低压阈值控制字参考第 24 章 24.3.6 节
24:0	Reserved	保留位

注：

(1) lv_th <31:29> <24:0> 禁止修改，即必须先读寄存器的值，在读出值的基础上只改变 lv_th 位，然后写回。

(2) 电源电压在比较阈值点反复波动时，除了会产生低压中断还可能会产生异常中断。如果需要避免该异常中断，在第一次进入低压中断时将上表阈值调高一档。

16.3 寄存器映射

LV 寄存器列表

基地址：0x3000_0330

寄存器	偏移地址	描述
LV_IRQ_EN	0x00	低压检测中断使能寄存器
LV_IRQ	0x04	低压检测中断寄存器
LV_TH	0x3000_0400	低压阈值寄存器

17 随机数生成模块（RANDGEN）

17.1 简介

RANDGEN 是一个生成真随机数的模块，生成之后会产生对应的标志位。指示软件去对应的空间读取对应的数据。

17.2 寄存器描述

17.2.1 随机数生成控制寄存器（RDGCR）

偏移地址：0x38

复位值：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

位	标记	功能描述
31:2	Reserved	保留位
1	data_rdy	状态标志位，1：随机数已经生成完毕，可以随时读取随机数。 data_rdy 只能通过复位清零。
0	rand_en	随机数生成使能位，1：使能；　0：不使能

17.2.2 随机数生成数据寄存器（RDGDR）

偏移地址：0x40

复位值：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															

位	标记	功能描述
31:0	rand_data	生成的随机数 第一次将 rand_en 写 1 之后，32 个 3K 时钟周期以后硬件将 data_rdy 置 1，以后只要 rand_en 为 1，在每个 3K 时钟周期都会更新一次随机数

17.3 寄存器映射

RANDGEN 寄存器列表

基地址：0x3000_0238

寄存器	偏移地址	寄存器描述
RDGCR	0x38	随机数生成控制寄存器
RDGDR	0x40	随机数数据寄存器

18 比较器（COMP）

18.1 简介

芯片内置了三个独立比较器，当使用比较器的时候需要将相应的 GPIO 口配置为模拟模式。

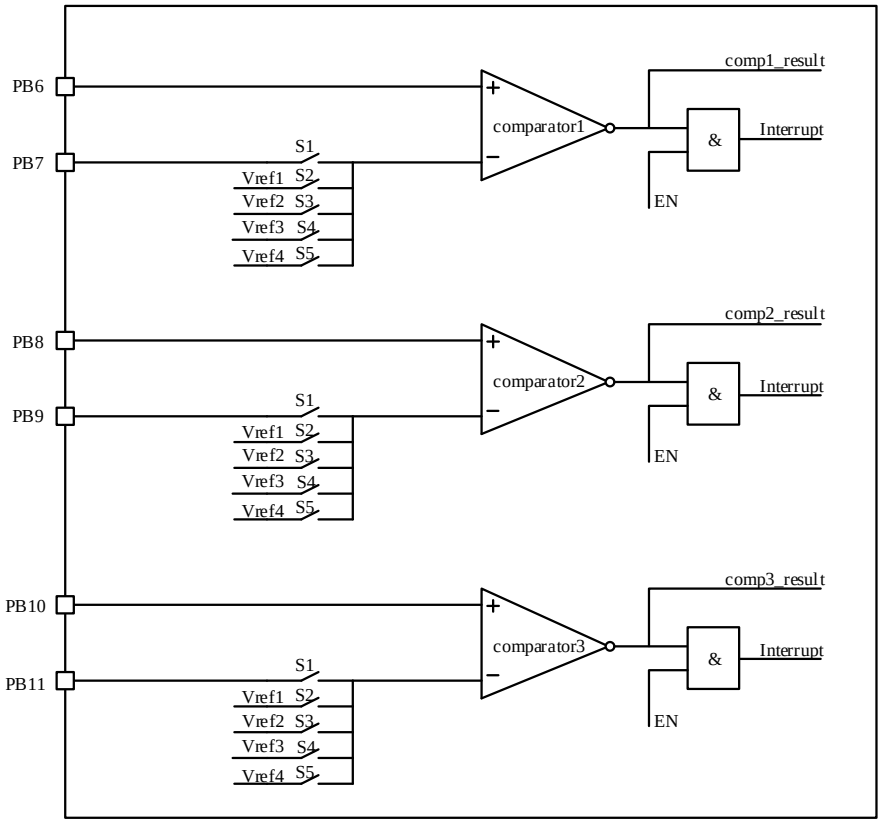


图 18-1 比较器框图

18.2 寄存器描述

COMP1 基址：0x3000_0B00
COMP2 基址：0x3000_0C00
COMP3 基址：0x3000_0D00

18.2.1 比较器控制寄存器（COMP_CTRL）

偏移地址：0x00
复位值：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

位	标记	功能描述
31:7	Reserved	保留位
6:4	ref_io_sel	负端来源选择和使能 <3><2><1>： 0 0 0 pad 接入； 0 0 1 0.3 V 接入； 0 1 0 0.6 V 接入； 0 1 1 0.9 V 接入； 1 0 0 1.2 V 接入； 1 0 1 比较器关断； 1 1 0 比较器关断； 1 1 1 比较器关断（默认）；
3	ratio_sel	速率选择 1：比较器的最高工作速度为 2MHz； 速率选择 0：比较器的最高工作速度为 1MHz； 高速时比较器会有更高的功耗。
2	int_en	比较器中断使能，1：使能； 0：不使能
1:0	mode[1:0]	中断边沿选择 00：高电平检测； 01：下降沿检测； 10：上升沿检测； 11：低电平检测；

18.2.2 比较器中断寄存器（COMP_IRQ）

偏移地址：0x04

复位值：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

位	标记	功能描述
31:1	Reserved	保留位
0	irq	比较器中断，1：产生比较器中断，写 1 清除中断

18.2.3 比较器结果寄存器（COMP_RESULT）

偏移地址：0x08

复位值：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

位	标记	功能描述
31:1	Reserved	保留位
0	result	比较器结果，1：高于 ref_io_sel 选择电压

18.3 寄存器映射

COMP1 寄存器列表

基地址：0x3000_0B00

寄存器	偏移地址	描述
COMP1_CTRL	0x00	比较器控制寄存器
COMP1_IRQ	0x04	比较器中断寄存器
COMP1_RESULT	0x08	比较器结果寄存器

COMP2 寄存器列表

基地址：0x3000_0C00

寄存器	偏移地址	描述
COMP1_CTRL	0x00	比较器控制寄存器
COMP1_IRQ	0x04	比较器中断寄存器
COMP1_RESULT	0x08	比较器结果寄存器

COMP3 寄存器列表

基地址：0x3000_0D00

寄存器	偏移地址	描述
COMP1_CTRL	0x00	比较器控制寄存器
COMP1_IRQ	0x04	比较器中断寄存器
COMP1_RESULT	0x08	比较器结果寄存器

19 UART 波特率自适应（TRIM）

19.1 简介

UART 波特率自适应模块通过计算 RX 低电平长度来计算 UART 的波特率。在测量波特率时，RX 的低电平宽度必须为 1 bit。

19.2 寄存器描述

19.2.1 配置寄存器（TRIM_CLK_CFG）

地址：0x3000_0380

复位值：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

位	标记	功能描述
31:9	Reserved	保留位
8	trim_en	调使能，0：关闭； 1：打开。trim 结束后硬件自动清零
7	Reserved	保留位
6	baudtrim	1：选择波特率适应功能
5:3	Reserved	保留位
2:1	uart_rx_sel	00：选择的时钟 UART1 的 rx 01：选择的时钟 UART2 的 rx 10：选择的时钟 UART3 的 rx 11：选择的时钟 UART4 的 rx *注意：baudtrim 设置为 0 时，uart_rx_sel 的设置才生效。*
0	trim_clk_sel	1：选择被测的时钟为 3k，0：选择被测的时钟为 RC

19.2.2 结果寄存器（TRIM_CLK_RESULT）

地址：0x3000_0384

复位值：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													

位	标记	功能描述
31:14	Reserved	保留位
13:0	trim_clk_result	trim 结束后显示为结果 测波特率：trim_clk_result 即为波特率

19.2.3 标志寄存器（TRIM_CLK_FLAG）

地址：0x3000_0388

复位值：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

位	标记	功能描述
31:30	Reserved	保留位
0	trim_clk_flag	trim 结束后标志 1：trim 已结束，0：正在进行 trim 或者没有进行 trim

19.3 使用方法

19.3.1 波特率自适应

1. 配置 MCU 和外设使用同一个时钟来源；

（设置时钟源选择寄存器（CMU_CLK_SEL），选择 MCU 和外设的时钟源相同）

2. 配置 baudtrim = 1，trim_en 写 0；
3. 配置测试串口号（uart_rx_sel）；
4. 配置 baudtrim = 1，trim_en 写 1；
5. 主机发送 0x7F；
6. RX 接收 UART 帧，帧中低电平只能是 1 位宽；
7. 等到 trim_en 变为 0，读出 trim_clk_result 的结果；
8. 使用 trim_clk_result 作为 UART 的波特率设置。

20 FLASH/NVM 烧录

CSM32RV20 配置了 40kBytes FLASH + 512bytes NVM，起始地址为 0x2000_0000，本部分介绍了其特性、功能和操作。

20.1 FLASH/NVM 主要特性

- 10K×32 位（40K 字节）主存储空间
- 每个扇区 512 字节
- 1 个 NVM，512 字节，用户可操作
- 按 32 位读，按 8 位写
- FLASH/NVM 编程/擦除操作
- 读写保护

20.2 FLASH/NVM 映射

表 20-1 展示了 FLASH/NVM 存储器的地址分配。

表 20-1 FLASH/NVM 存储器地址映射

块	名字		存储地址	大小
FLASH	扇区 0	行 0	0x2000_0000 – 0x2000_00FF	256 字节
		行 1	0x2000_0100 – 0x2000_01FF	256 字节
	扇区 1	行 0	0x2000_0200 – 0x2000_02FF	256 字节
		行 1	0x2000_0300 – 0x2000_03FF	256 字节
	扇区 2		0x2000_0400 – 0x2000_05FF	512 字节
	扇区 3		0x2000_0600 – 0x2000_07FF	512 字节
	扇区 4		0x2000_0800 – 0x2000_09FF	512 字节
	•		•	•
	•		•	•
	•		•	•
	扇区 79		0x2000_9E00 – 0x2000_9FF7	504 字节
			0x2000_9FF8 – 0x2000_9FFF	保留
NVM			0x2001_0000 – 0x2001_01FF	512 字节

其中，主存储空间用于存储用户程序，可以通过 cJTAG 或者 UART 接口对其编程。用户程序可以在线读写 512 字节 NVM。

20.2.1 NVM 操作

CSM32RV20 配备了 512 字节的 NVM 供用户存储自定义数据，提供了扇区的用户程序操作函数：flash_operation()，函数的入口地址为 0x2100_0fe2，用户程序可通过声明函数指针调用该函数，实现对 NVM 和主扇区操作，关于函数的定义如下：

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

参数和返回值描述：

code：功能码，用于选择读，写，擦除功能；

addr：读/编程的起始地址，0-0x9ffc；

*p：数组指针；

num ：待操作字节数，1-128

clk：flash 时钟，一般默认 0

返回值：0，成功；

1) 非法地址；
2) 待操作数大于剩余字节数；
3) 数组指针为 NULL；
4) 操作码错误。

注：执行读/写操作前需对数组越界检查。

功能码 code 的描述如表 20-2 所示。

表 20-2 功能码 code 描述

操作	代码	描述
Erase_sector	1	擦除指定 n 个主扇区
Erase_NVM	2	擦除 NVM
Erase_chip	3	擦除整个芯片（用户禁止使用，会擦掉整个主扇区，造成死机）
Write_bytes	4	写 n 字节到 flash（地址要 4 字节对齐，一次操作只能在一个 row 内，一个扇区分为 4 个 row，每个 row 有 128 字节）
Read_bytes	5	从主扇区读 n 字节（地址要 4 字节对齐）
Write_NVM	6	写 n 字节到 NVM（地址要 4 字节对齐，一次操作只能在一个 row 内，NVM 分为 4 个 row，每个 row 有 128 字节）
Read_NVM	7	从 NVM 读 n 字节（地址要 4 字节对齐）

定义函数指针示例：

```
#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;
```

20.2.2 FLASH 读写保护

FLASH 存储器能够保护用户程序防止外部读写访问。通过上位机软件或 IDE 软件设置。

20.3 FLASH/NVM 烧录

用可以通过两种方法下载用户程序至 CSM32RV20：

1) 通过编程上位机软件：CSM32RV20 于 ROM 中存放了引导程序，通过串口与编程上位机通信。用户正确连接串口的 TX1/RX1 引脚后即可通过上位机进行编程；

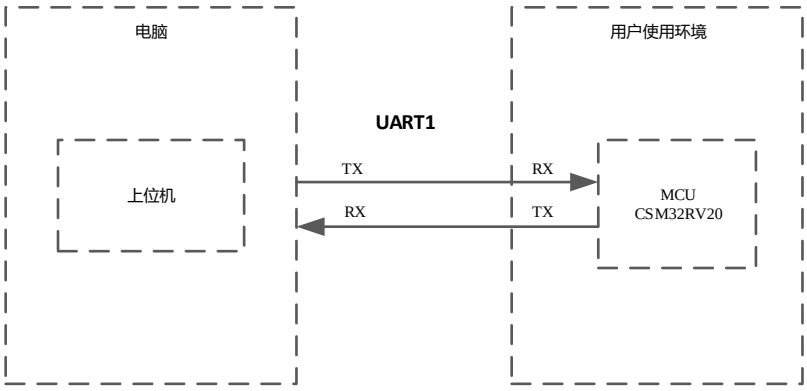


图 20-1 MCU 与上位机连接

2) 通过 cJTAG 接口：CSM32RV20 配备了 2-线 cJTAG 接口用于调试和编程，用户连接调试器和 MCU 的调试接口：TCKC 和 TMSC 引脚，通过 IDE 软件进行编程，关于 cJTAG 接口的描述详见 21 Debug 支持，调试和编程的详细操作流程详见 CSM Studio IDE Manual。

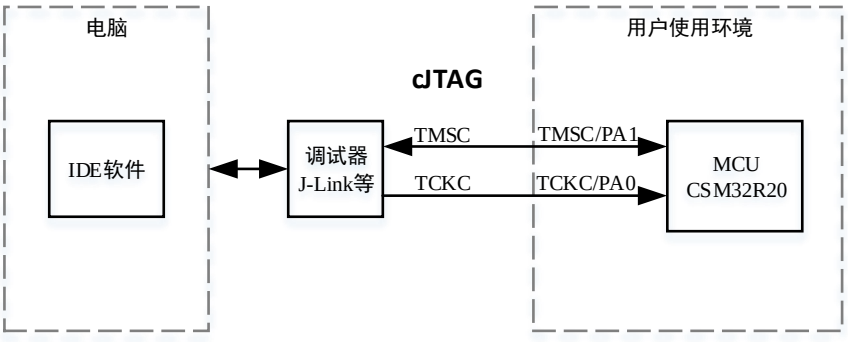


图 20-2　MCU 与 cJTAG 连接

21 Debug 支持

21.1 概述

CSM32RV20 围绕 32 位 RISC-V 内核构建，该内核包含 JTAG 调试传输模块（debug transport module，DTM），支持使用单个外部工业标准 1149.1 JTAG 接口测试和调试系统。DTM 支持交互式调试和最多 4 个硬件断点。JTAG 调试接口由一个工业标准 2-线 cJTAG 接口驱动。

21.2 cJTAG 调试接口

CSM32RV20 内核内嵌了 cJTAG 适配器，提供了由两个信号组成的接口：TCKC 和 TMSC。许多调试器可以同时支持传统的 JTAG 和新的 cJTAG。使用时将调试探针的 TCKC 和 TMSC 引脚与 MCU 相对应的引脚相连，支持在线调试和下载，具体操作详见 CSM Studio IDE Manual。

表 21-1 cJTAG 调试接口引脚

cJTAG 引脚名	cJTAG 调试端口		引脚分配
	类型	调试分配	
TMSC	IO	数据输入/输出	PA1
TCKC	I	时钟	PA0

cJTAG 的 2 个引脚默认复用为调试接口，若不使用 cJTAG 则用户可以将其复用为通用 I/O，详见 5 通用和复用功能 I/O。

22 RISC-V 内核

内置 32 位 RISC-V 核，采用两级流水线，支持 IMAC 指令。RISC-V 相关请参照[1][2]。

[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 芯片电子签名

电子签名包含 VersionSize 和 MCUID，可以通过 cJTAG、编程上位机或 MCU（用户程序）读取，包含出厂编写的识别数据。

23.1 芯片版本号（VersionSize）

VersionID 寄存器包含芯片版本信息和 FLASH 容量信息。

地址：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							

位	标记	功能描述
31:8	version	本芯片的版本号
7:0	size	本芯片中 FLASH 容量的大小 0000_0001：4k； 0000_0010：8k； 0000_0011：12k； ……. 0000_1010：40k

23.2 MCUID

MCUID 为出厂唯一 ID。

地址：0x3000_0420

复位值：ID 值

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

位	标记	功能描述
31:0	mcuid	本颗 mcu 芯片的 ID 号

24 电气参数

24.1 参数条件

除非特别说明，所有电压均以 V_{SS} 为参考。

24.1.1 最大和最小值

除非特别说明，所有最大值和最小值在最坏的环境温度、供电电压和时钟频率下得到保证。

基于特性结果、设计模拟和/或技术特性的数据在表脚注中指明，未在生产中进行测试。最小值和最大值样本测试。

24.1.2 典型值

除非另有说明，典型数据基于 T_A = 25 摄氏度，V_{DD} = 3.3 V。仅作为设计指南。

24.1.3 电源供电方案

CSM32RV20 有一个电源引脚和地（底部焊盘）。

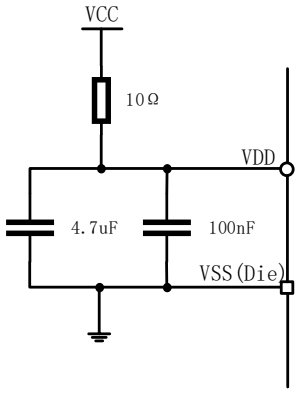


图 24-1 电源供电方案

24.1.4 电流消耗测量

电流消耗测量如图 24-2 所示。

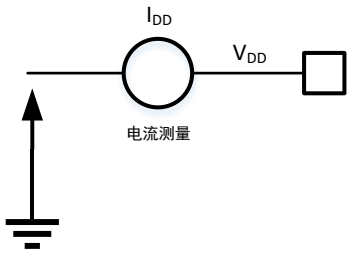


图 24-2 电流消耗测量

24.2 绝对最大额定值

临界或超过绝对最大额定值将可能导致芯片工作异常甚至损坏。

表 24-1 绝对最大额定值表

符号	参数	最小值	最大值	单位
V_{DD}-V_{SS}(1)	外部主供电电压	-0.3	5.8	V
V_{IN}(1)	引脚输入电压	-0.3	5.8	V
$V_{ESD(HBM)}$(1)	静电放电电压（人体模型）		2000	V
T_S	存储温度	-55	150	℃

注：1. 设计参数

24.3 操作条件

24.3.1 一般操作条件

表 24-2 一般操作条件

符号	参数	条件	最小	标准	最大	单位
f_{RC}	内部系统时钟频率	Ta=25℃，V_{DD}=3.3 V	10	16/32	34	MHz
f_{OSC}	外部晶振时钟频率	Ta=25℃，V_{DD}=3.3 V	4	16/32	32	MHz
f_{3K}	内部 3K 时钟频率	Ta=25℃，V_{DD}=3.3 V	1.8	3	8	KHz
V_{DD}	标准操作电压	使用 ADC	2.5	3.3	5	V
		未使用 ADC	1.8	3.3	5.5	
IDD	标准工作电流	外设关闭，3K OSC 和看门狗工作，16MHZ RCOSC	-	3.3	-	mA
	标准工作电流	外设关闭，3K OSC 和看门狗工作，32MHZ RCOSC	-	5.7	-	mA
	标准工作电流	外设关闭，3K OSC 和看门狗工作，16MHZ 晶振	-	2.9	-	mA
	标准工作电流	外设关闭，3K OSC 和看门狗工作，32MHZ 晶振	-	4.2	-	mA

	待机模式	外设关闭，3K OSC 和看门狗工作，SRAM 保持	-	1.6	-	mA
	睡眠模式	外设关闭，3K OSC 和看门狗工作，SRAM 保持	-	364	-	uA
	掉电模式 1	外设关闭，3K OSC 和看门狗工作，SRAM 保持	-	3.5	-	uA
	掉电模式 2	外设关闭，3K OSC 和看门狗工作	-	1	-	uA

注：1. 没有特别说明，Ta=25 ℃,VDD=3.3V 条件下参数。

24.3.2 内部系统时钟源参数

RCOSC 的频率能够通过寄存器修调,寄存器为绝对地址 0x3000_0404 下的低 9 位.随着修调字的递增,频率递减,递减幅度为 0.4% .

注意:除了频率修调字,其他位禁止修改.

表 24-2 内部系统时钟参数

符号	参数		条件	最小	标准	最大	单位
f_{CLK}	频率	-	-	10	16/32	34	MHz
TRIM	微调步进	16M	-	-	0.4	-	%
		32M	-	-	0.4	-	%
ACC	振荡器精度	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}	起振时间	-	Ta=25℃，V_{DD}=3.3 V	-	1	-	us

24.3.3 外部时钟源参数

外部时钟源使用低成本晶振：4 ~ 32 MHz ± 60 ppm。

使用时，GPIOB12/13 应配置为模拟功能；晶振引脚在未设置为模拟功能时，外部晶振被关闭且输入输出引脚用作标准 I/O。

晶振负载电容推荐高质量外部陶瓷电容，容值在 5 pF 至 20 pF 范围内（C1 = C2 = 15 pF 为典型值，实际使用请参考晶振的数据手册）。

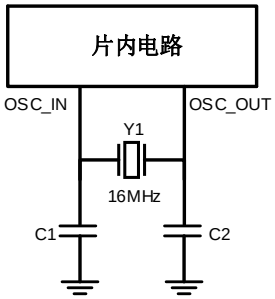


图 24-3 使用 16 MHz 晶振时典型应用图

表 24-3 晶振参数表

符号	参数	V_{DD}	条件	最小	标准	最大	单位
f_{CLK}	时钟频率		Ta=25℃，V_{DD}=3.3 V	4	16/32	32	MHz
I_{VDD}(1)	晶振稳定后电流	5.5 V	C1 = C2 = 15 pF	-	113	-	uA
		3.3 V			81		
		1.8 V			66		
t_{SU}(1)	起振时间	3.3 V	C1 = C2 = 15 pF	-	0.6	-	ms

注：1.为设计参数

24.3.4 I/O 端口参数

通用输入输出参数

表 24-4 I/O 直流参数

符号	参数	V_{DD}	条件	最小	最大	单位
V_{IH}	I/O 输入高电压	5 V	-	0.7 × V_{DD}	-	V
		3.3 V		2.0		
		1.8 V		0.8 × V_{DD}		
V_{IL}	I/O 输入低电压	5 V	-	-	0.3 × V_{DD}	V
		3.3 V			0.3× V_{DD}	
		1.8 V			0.2 × V_{DD}	
V_{HYS}	施密特触发器迟滞	5/3.3/1.8 V	-	0.1 × V_{DD}	-	V
I_{IH}	I/O 输入高电流	5/3.3/1.8 V	-	-	+1	μA
I_{IL}	I/O 输入低电流	5/3.3/1.8 V	-	-1	-	μA
V_{OH}	I/O 输出高电压	5 V	高驱动 I*min* = 16mA 低驱动 I*min* = 8mA	V_{DD}-0.8		V
		3.3 V	高驱动 I*min* = 8mA 低驱动 I*min* = 4mA	2.4		
		1.8 V	高驱动 I*min* = 4mA 低驱动 I*min* = 2mA	V_{DD}-0.45		
V_{OL}	I/O 输出低电压	5 V	高驱动 I*min* = 16mA 低驱动 I*min* = 8mA		0.5	V
		3.3 V	高驱动 I*min* = 8mA		0.4	

符号	参数	V_{DD}	条件	最小	最大	单位
			低驱动 I*min* = 4mA			
		1.8 V	高驱动 I*min* = 4mA 低驱动 I*min* = 2mA		0.45	
R_{pup}	上拉电阻	5/3.3/1.8 V	-	20	100	KOhm
R_{pdn}	下拉电阻	5/3.3/1.8 V	-	20	100	KOhm
C_{IN}	I/O 输入电容	5/3.3/1.8 V	-	-	10	pF

注：以上为设计参数

24.3.5 ADC 参数

符号	参数	条件	最小	标准	最大	单位	备注
VRES	分辨率	16bit		36.62		uV/LSB	
		15bit		73.24		uV/LSB	
		14bit		146.48		uV/LSB	
		13bit		292.97		uV/LSB	
TCONV	转换时间(ADC 时钟 4MHz)	16bit		35.25		us	
		15bit		19.25		us	
		14bit		11.25		us	
		13bit		7.25		us	
VERR(1)	测量误差			±3.5		mV	
INGAIN	输入通道增益		1/4	1	128		2~128 仅支持 PA11 输入
VINRANG	输入电压范围		0		VDD		VDD≤ 4.8V
fCLK	时钟频率			4	8	MHz	

注 1. 输入 1/4 增益，内部基准 0~1.2V，V_{DD}=3.3V，16 位分辨率

24.3.6 低压检测阈值

lv_th<3:0>	阈值电压（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

注：阈值电压误差±50mv

25 封装信息

CSM32RV20 采用 4x4 mm QFN32、TSSOP20 或者 3x3mm QFN20 封装。

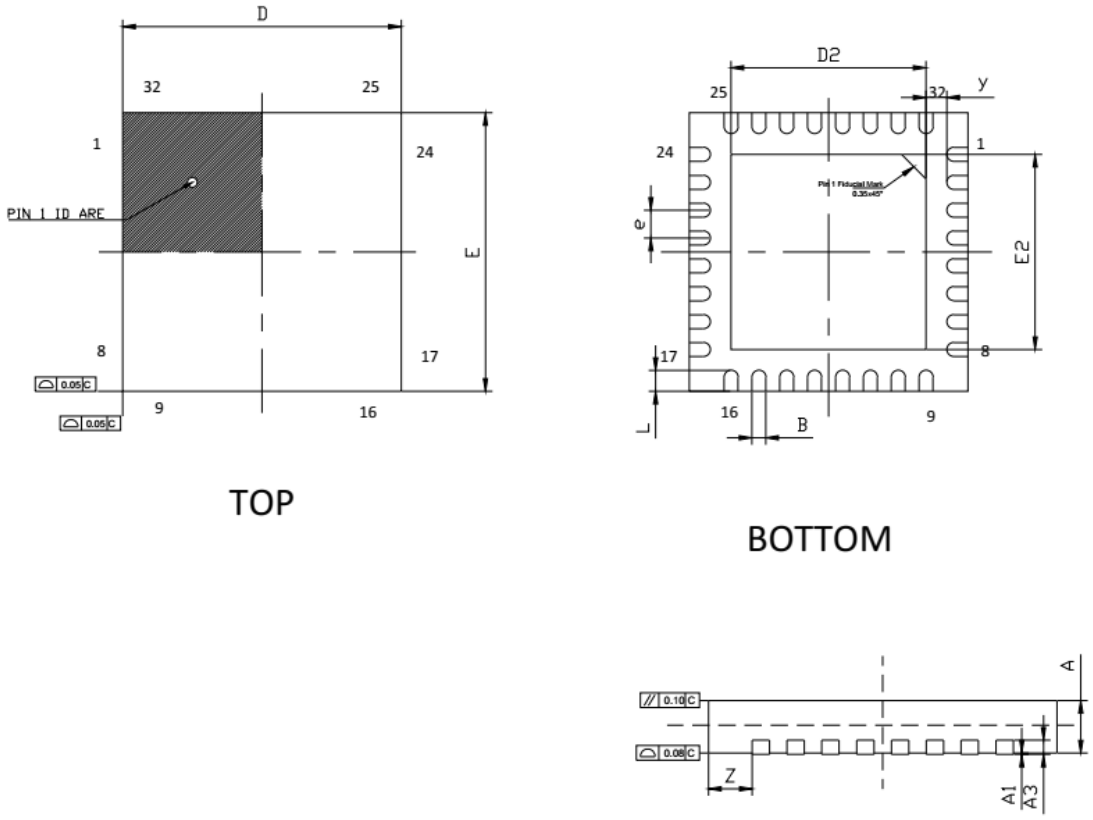


图 25-1 QFN32 封装

尺寸

单位	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

注：所有尺寸均为毫米

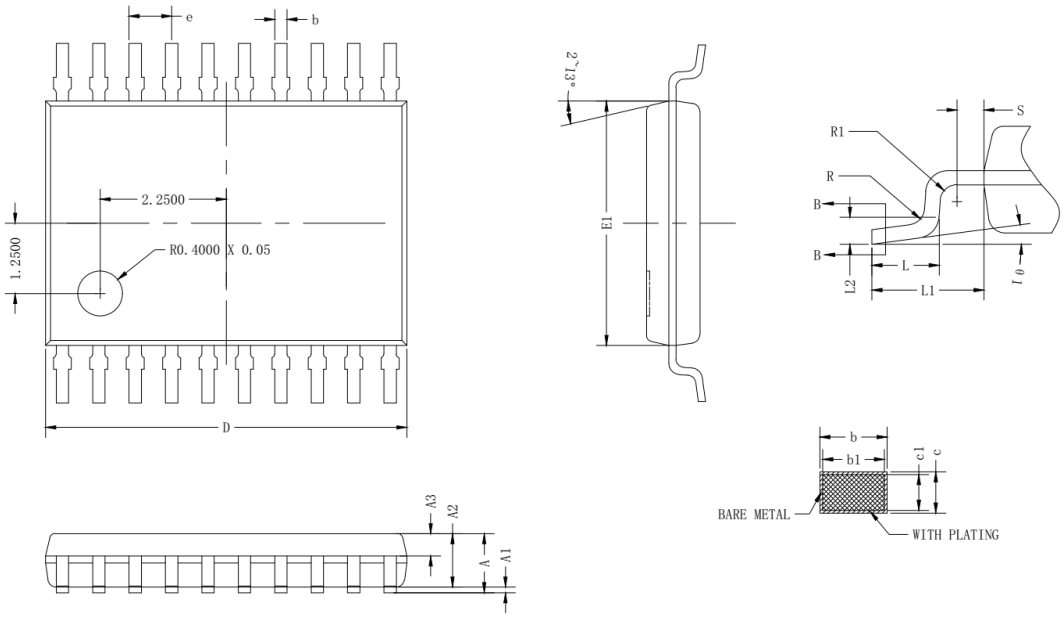
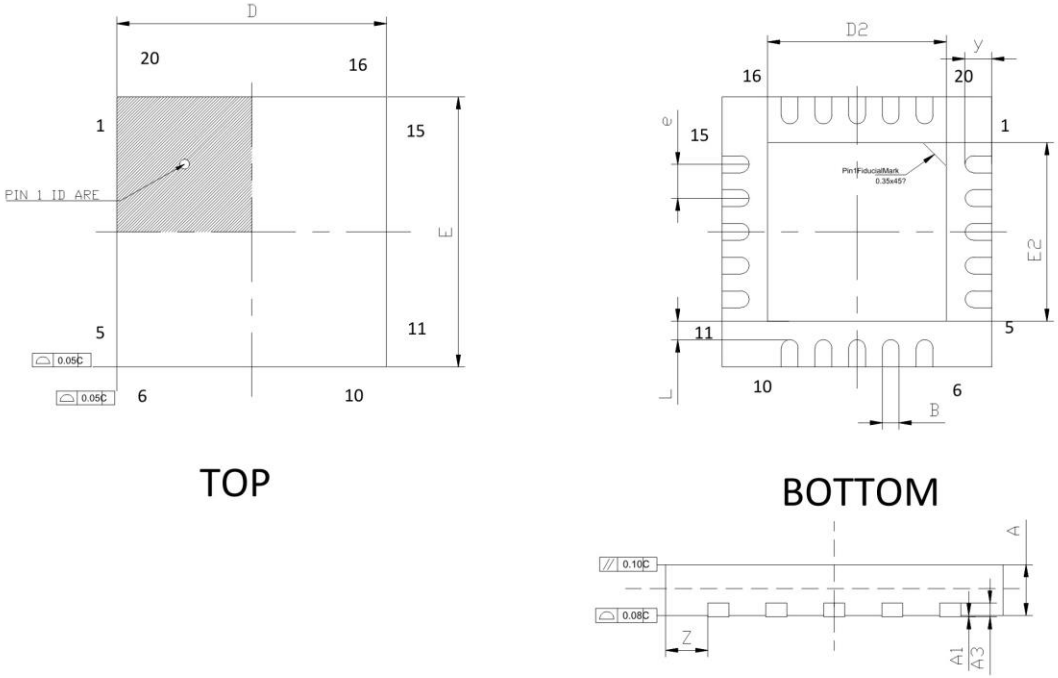


图 25-2 TSSOP20 封装

尺寸

符号	mm		
	最小	典型	最大
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	-	-

注：所有尺寸均为毫米



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]:						

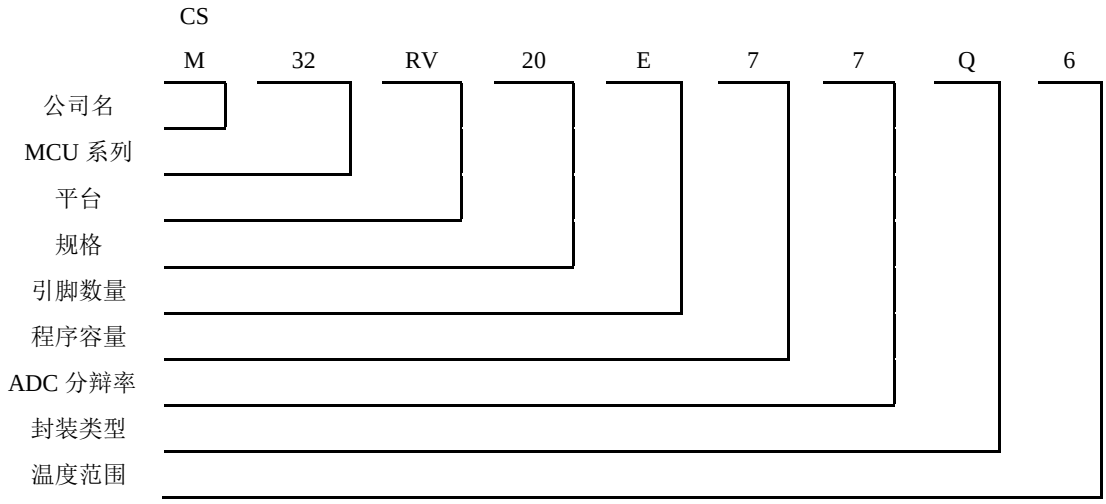
图 25-3 3x3mm QFN20 封装

25.1 芯片丝印样式

CSM32RV20 E77Q6 623AW01 .

注：MCU 芯片打印规则共分四行，其中：
第一行为芯片型号；
第二行为不同版本属性的说明；
第三行为周记，为我公司通用规则；
第四行为标注 pin1 位置的圆点。

25.2 芯片丝印字母的详细说明



引脚数量规则

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

程序容量：单位 kByte

程序容量	1	2	4	8	16	32	40	64	128	256	512
代号	1	2	3	4	5	6	7	8	A	B	C

ADC：单位 bits

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

封装类型：

封装类型	SOP	TSSOP	LQFP	QFN
代号	S	T	L	Q

温度范围：

温度范围	-40~85℃	-40~105℃	-40~125℃
代号	6	7	8

26 版本信息

版本	修订日期	修订内容摘要
Rev1.0	2021/9/24	初稿
Rev1.1	2021/11/16	修改独立看门狗 IWDG_RLR 寄存器的写描述：在向 IWDG_KR 寄存器中写入 0x5555 之后必须等待一个指令周期才能写 IWG_RLR 寄存器。
Rev1.2	2021/12/3	修改联系方式
Rev1.3	2022/01/13	1） 补充 ADC 通道描述； 2） 更新 NVM 操作描述。
Rev1.4	2022/03/01	增加 3x3mm QFN20 封装。
Rev1.5	2022/04/07	1） 修改表 1-1/1-2/1-3 中的 PA8 描述错误； 2） 修改图 14-1 标注错误。
Rev1.6	2022/04/07	更新图 24-1。
Rev1.7	2022/04/12	修改时钟分频寄存器 rtc_clk_div 描述。
Rev1.8	2022/04/27	1） 修改表 24-4 中 V_{IL} 描述； 2） 修改表 20-1 中 FLASH 地址描述。

27 技术支持与联系方式

南京中科微电子有限公司 技术支持中心

电话：025-68517780

地址：南京市玄武区徐庄软件园研发三区 B 栋 201

销售

手机：18961759481

邮箱：sales@csmic.ac.cn

技术支持

手机：13645157034

邮箱：supports@csmic.ac.cn

网址：http://www.csm-ic.com