MS32F031A6 用户手册 V1.2.0

晟矽微电 32 位 Arm Cortex-M0 内核单片机

32 位 ARM Cortex-M0 内核 FLASH 型 MCU，最高 48MHz 主频，32KB FLASH ROM，4K SRAM，12 位高速 ADC， 5 通道 DMA，1 个 16 位 7 通道高级 Timer（支持 6 路 PWM），1x32 位+4x16 位通用定时器，1 路 UART，1 路 SPI，1 路 I2C，CRC，RTC 日历，2 个 CMP，3 个 OPA，SWD 调试

主要特性

CORE

- 32 位 ARM Cortex-M0 内核
- 最高主频 48MHz

存储器

- 片上 32K 字节 FLASH 程序存储器
- 片上 4K 字节 SRAM ，支持硬件奇偶校验
- 支持代码加密保护功能

复位与电源管理

- 数字和 IO 供电：2.0～5.5V
- 模拟电源供电：VDDA=VDD~5.5V
- 上电/断电复位(POR/PDR)、
- 可编程电压监测器(PVD)
- 低功耗模式：睡眠、停止和待机模式
- VBAT 为 RTC 和备份域寄存器供电

时钟管理

- 片上高精度 16MHz RC 高速振荡器
- 片上 40KHz 低功耗 RC 低速振荡器
- 带校准功能的 32.768 kHz RTC 晶体振荡器
- 4~32MHz 晶体振荡器
- PLL 时钟，最高输出 48MHz，支持多档倍频可调

I/O

- 最多 39 个 FAST I/O，最多 25 个 I/O 支持 5V 耐受
- 所有 I/O 口可以映像到 16 个外部中断

定时器/计数器

- 1 个 16 位 7 通道高级控制定时器，支持 6 路 PWM 输出，带死区控制及紧急停止
- 1 个 32 位定时器和 1 个 16 位定时器，支持 4 路输入捕获和比较输出，支持 IR 解码
- 1 个 16 位定时器，支持 2 路输入捕获和比较输出，1 路比较反向输出，带死区控制及紧急停止
- 1 个 16 位定时器，支持输入捕获、比较输出和比较反向输出，带死区控制及紧急停止，支持 IR 解码
- 1 个 16 位定时器，支持输入捕获和比较输出
- 2 个看门狗定时器：IWDG 和 WWDG
- 1 个 24 位递减计数 SysTick 定时器

12 位高精度 ADC

- 12 位高精度逐次逼近型 ADC
- 多达 10 个外部通道
- 6 个内部通道：温度传感器，VREF，VBAT/2，3 个运算放大器输出
- 转换范围：0 ~ VDDA
- 工作电压范围：2.4V~VDDA

通讯接口

- 1 个USART 接口,支持主同步 SPI 模式,ISO7816,LIN,IrDA，自动波特率和唤醒功能
- 1 个 I2C 接口，支持快速模式+（1Mbps），低功耗唤醒
- 1 个SPI 接口，支持 4~16 位的数据格式

CRC-32 计算单元

DMA 控制器

- 支持 5 个 DMA 通道
- 支持外设：ADC、SPI、I2C、USART、TIMEx（x = 1，2，3，16，17）

RTC 日历

- 闹钟中断功能
- 从停止，待机模式中周期唤醒

2 个高性能电压比较器（CMP1/2）

- 1 个比较器正相 6 通道可选（3 外部管脚+3 内部运放输出）
- 1 个电压比较器正相 4 通道可选（1 外部管脚+3 内部运放输出）
- 反相可选择内部多档位比较电压，且有 SMT 档位选择
- 输出带数字滤波，极性选择，引发中断，与定时器产生联动效果，同时作为高级定时器的刹车输入信号
- 带有自校准功能

3 个高增益运算放大器(OPA1/2/3)

- 内部放大倍数多档位可选
- 负端与输出接口丰富，可以适配不同应用
- 带有自校准功能

96 位的芯片唯一码

串行调试(SWD)接口

工作环境温度

- -40℃~105℃

封装形式

- LQFP48/32、QFN32

1 产品简介

1.1 概述

本产品使用高性能的 ARM Cortex-M0 32 位内核，最高工作频率 48MHz，内置高速存储器，丰富的增强 I/O 端口和联接到一条 APB 总线的外设。所有型号的器件都包含 1 个 12 位的 ADC、5 个通用 16 位定时器、1 个高级 PWM 定时器、2 个电压比较器和 3 个运算放大器，还包含标准和先进的通信接口：1 个 I2C 接口和 SPI 接口、1 个 USART 接口。

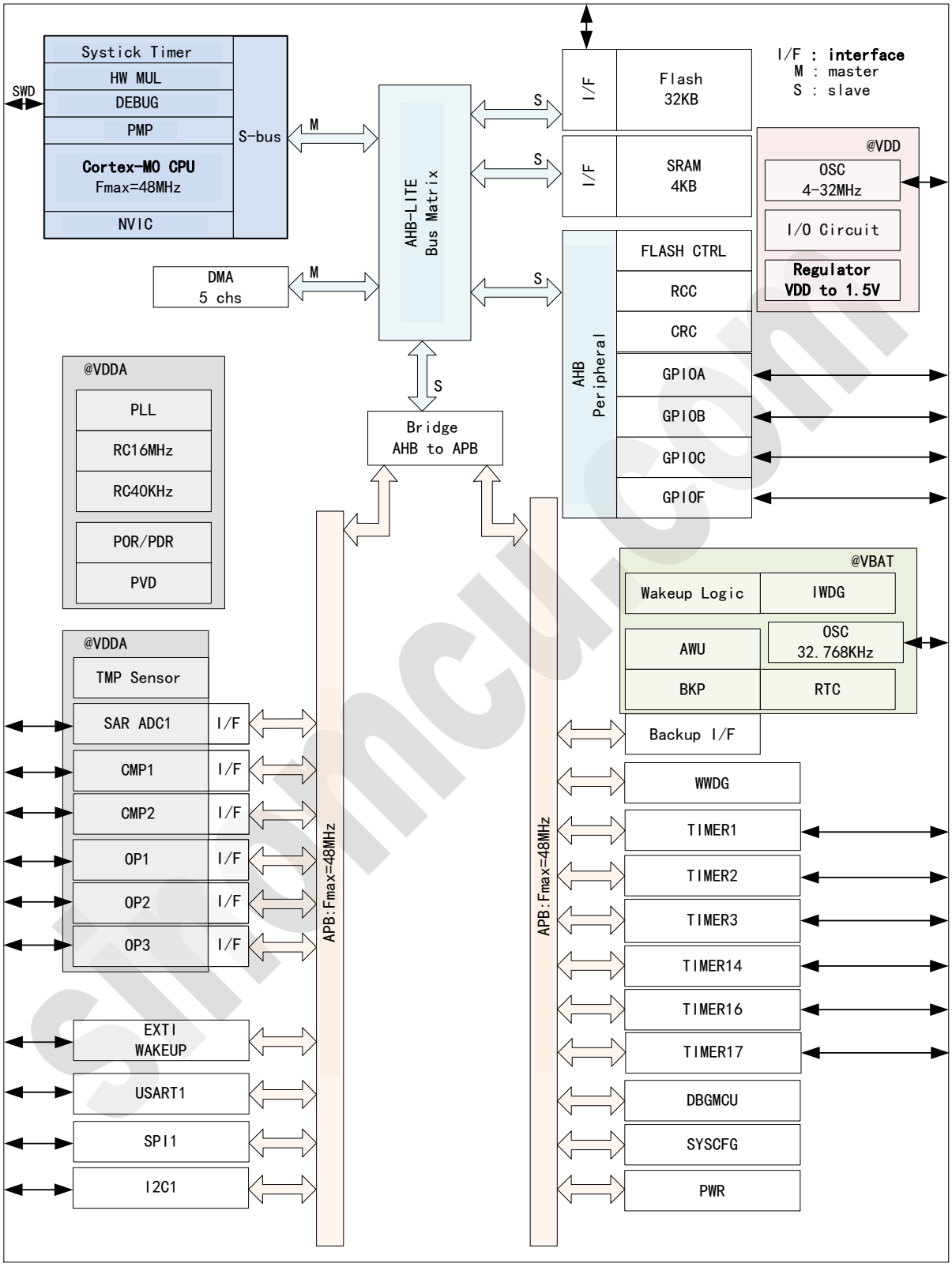
本产品供电电压为 2.0V 至 5.5V，包含-40° C 至+85° C 温度范围和-40° C 至+105° C 的扩展温度范围。一系列的省电模式保证低功耗应用的要求。

本产品提供包括 32 脚和 48 脚共 2 种不同封装形式；根据不同的封装形式，器件中的外设配置不尽相同。下面给出了该系列产品中所有外设的基本介绍。

这些丰富的外设配置，使得本产品微控制器适合于多种应用场合：

- ✧ 电机驱动和应用控制
- ✧ 医疗和手持设备
- ✧ PC 游戏外设和 GPS 平台
- ✧ 工业应用：可编程控制器(PLC)、变频器、打印机和扫描仪
- ✧ 警报系统、视频对讲、和暖气通风空调系统等

1.1 系统框图



1.2 引脚定义

引脚定义请参考对应的数据手册。

1.3 订购信息

订购信息请参考对应的数据手册。

1.4 术语和缩写

术语/缩写	全称	描述
rw	read/write	软件能读写此位
r	read-only	软件只能读此位
w	write-only	软件只能写此位，读此位将返回复位值
rc_w1	read/clear	软件可以读此位，也可以通过写‘1’清除此位，写‘0’对此位无影响。
rc_w0	read/clear	软件可以读此位，也可以通过写‘0’清除此位，写‘1’对此位无影响
rc_r	read/clear by read	软件可以读此位，读取此位自动清除此位，写‘0’对此位无影响
rs	read/set	软件可以读此位及对此位写‘1’，写‘0’对此位无影响
rt_w	read-only write trigger	软件可以读此位，对此位写‘0’或‘1’可以触发事件，但不影响此位的值。
t	toggle	软件只能通过写’1’来翻转此位，写’0’对此位无影响。
Res.	Reserved	保留位，必须保持默认值不变
SWD	Serial Wire Debug	串行线调试接口，2 线
JTAG	Joint Test Action Group	4 线仿真接口
Word		字，32 位长的数据或指令长度
Half-word		半字，16 位长的数据或指令长度
B/Byte		字节，8 位数据长度
ICP	In Circuit Programming	在电路编程，用户可以通过仿真接口直接将程序下载至芯片中
IAP	In Application Programming	在应用编程，用户可以通过程序对芯片本身进行编程
Option bytes		选项字节，用于存储 MCU 的用户配置字
AHB	Advanced High-performance Bus	先进高性能总线
APB	Advanced Peripheral Bus	先进外设总线

1.5 可用的外设

有关本产品系列的全部型号，具体的外设存在与否及数目，请查阅对应的数据手册。

2 产品器件对比

产品功能和外设配置

外围接口 \ 产品型号		MS32F031A6AOZC/YA	MS32F031A6AOZW
闪存 - K 字节		32	
SRAM - K 字节		4	
定时器	通用目的	1 (32-bit) 4 (16-bit)	1 (32-bit) 4 (16-bit)
	高级控制	1	1
通讯接口	SPI	1	1
	I2C	1	1
	USART	1	1
RTC 日历		1	1
12 位同步 ADC(通道数)		1 16channels（10 ext + 6 int）	1 16 channels（10 ext + 6 int）
电压比较器		1 (3 channel positive input) 1 (1 channel positive input)	1 (3 channel positive input) 1 (1 channel positive input)
运算放大器		3	3
GPIOs		25	39
CPU 频率		48 MHz	
工作电压		2.0 to 5.5 V	2.0 to 5.5 V
工作温度		周围环境温度：-40 to +85 ° C / -40 to +105 ° C 结温温度: -40 to + 125 ° C	
封装		LQFP32/QFN32	LQFP48

3 存储器和总线架构

3.1 系统架构

系统主要由以下几个模块组成：

2 个主单元：

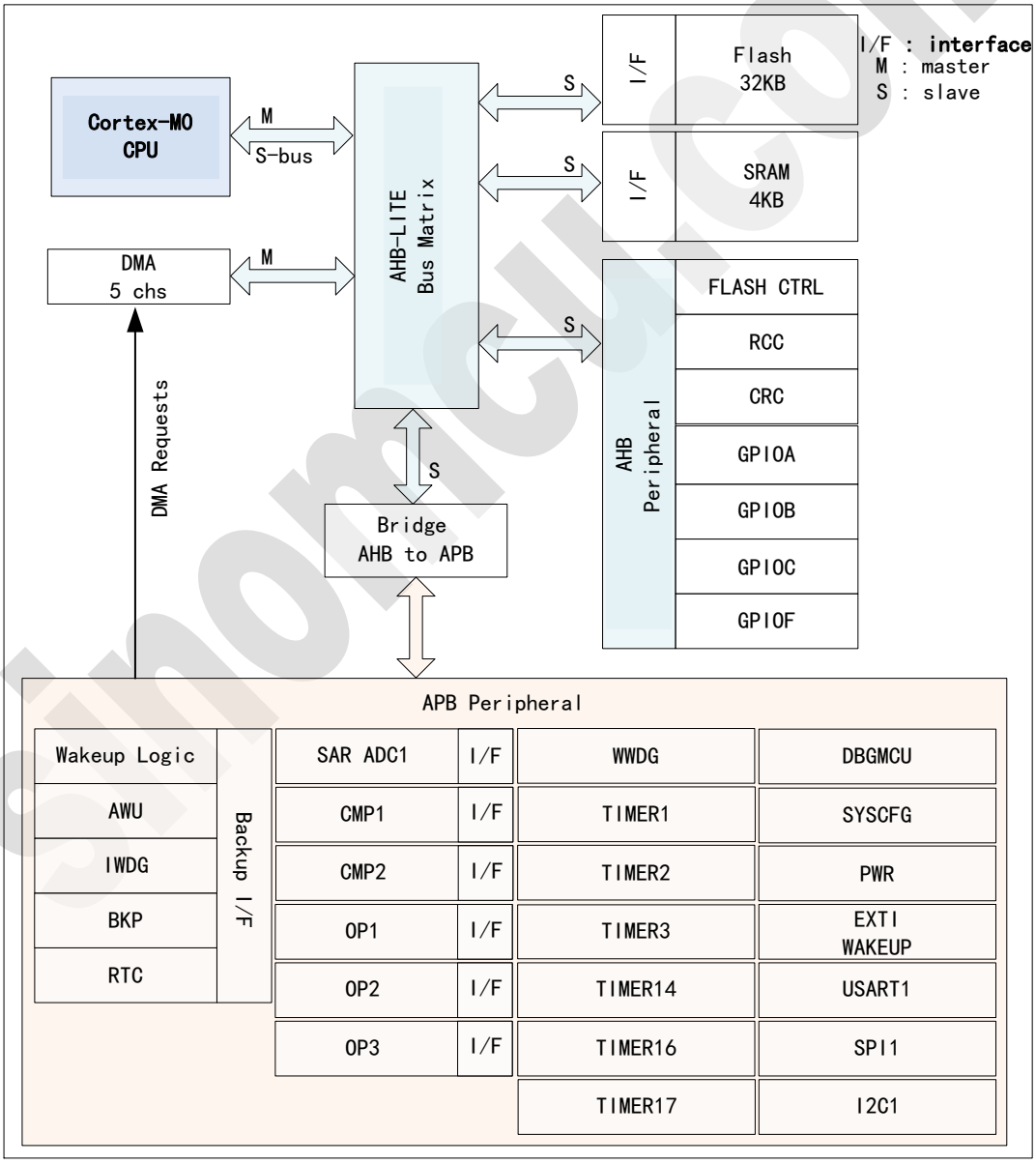
- Cortex-M0 内核及系统总线 (S-bus)
- 通用 DMA

4 个从单元：

- 内部 SRAM
- 内部闪存存储器
- AHB 到 APB 的桥，连接所有的 APB 设备
- AHB 设备总线，专门用于连接 GPIO、RCC、CRC 等模块

这些单元内部由一个多级 AHB 总线结构互联，如图所示。

系统架构框图



系统总线

此总线连接 Cortex-M0 内核的系统总线到总线矩阵，总线矩阵来协调内核和 DMA 间的总线访问控制。

DMA 总线

此总线将 DMA 的 AHB 主控接口与总线矩阵相联，总线矩阵协调 CPU 和 DMA 到 SRAM、闪存和外设的访问控制。

总线矩阵 (BusMatrix)

总线矩阵管理内核系统总线与 DMA 总线的访问仲裁，总线矩阵由 2 个主单元总线 (系统总线、DMA 总线) 及 4 个从单元总线(FLASH、SRAM、AHB2GPIO 和 AHB2APB 桥) 组成。

AHB 外设通过总线矩阵与系统总线相连，允许 DMA 访问。

AHB 到 APB 桥 (AHB2APB bridges－APB)

AHB 到 APB 桥在 AHB 与 APB 总线间提供同步连接。有关连接到桥的不同外设的地址映射请参考存储器映射章节。

在每次复位之后，所有的外设时钟都关闭(除 SRAM 及 FLASH 外)。外设使用前，用户必须打开相应的 RCC_AHBENR、RCC_APB2ENR 或 RCC_APB1ENR 寄存器中时钟使能位。

注：当对 APB 寄存器进行 8 位或者 16 位访问时，该访问会被自动转换成 32 位的访问：AHB2APB 桥会自动将 16 位或者 8 位的数据扩展以配合 32 位的位宽。

3.2 存储器

3.2.1 概述

程序存储器，数据存储器，寄存器及 I/O 口统一编址，其线性地址空间达到 4G。

数据字节以小端格式存放在存储器中，一个字里的最低地址字节被认为是该字的最低有效字节，而最高地址字节是最高有效字节。

寻址空间分成 8 块，每块 512MB。

其他所有没有分配给片上存储器和外设的存储器空间都是保留的地址空间。详细请参考存储器映像和寄存器编址章节和外设章节。

3.2.2 存储器映像及寄存器编址

存储器映像

总线	编址范围	大小	外设	备注
AHB	0xE000 0000 - 0xE00F FFFF	1MB	Cortex M0 内部外设	
	0x4800 1800 - 0x5FFF FFFF	~384 MB	Reserved	
	0x4800 1400 - 0x4800 17FF	1KB	GPIOF	
	0x4800 1000 - 0x4800 13FF	2KB	Reserved	
	0x4800 0800 - 0x4800 0BFF	1KB	GPIOC	
	0x4800 0400 - 0x4800 07FF	1KB	GPIOB	
	0x4800 0000 - 0x4800 03FF	1KB	GPIOA	
	0x4002 4400 - 0x47FF FFFF	~128 MB	Reserved	
	0x4002 3400 - 0x4002 3FFF	3 KB	Reserved	
	0x4002 3000 - 0x4002 33FF	1 KB	CRC	
	0x4002 2400 - 0x4002 2FFF	3 KB	Reserved	
	0x4002 2000 - 0x4002 23FF	1 KB	FLASH 接口	
	0x4002 1400 - 0x4002 1FFF	3 KB	Reserved	
	0x4002 1000 - 0x4002 13FF	1 KB	RCC	
	0x4002 0400 - 0x4002 0FFF	3 KB	Reserved	
	0x4002 0000 - 0x4002 03FF	1 KB	DMA	
APB	0x4001 8000 - 0x4001 FFFF	32 KB	Reserved	
	0x4001 5C00 - 0x4001 7FFF	9 KB	Reserved	
	0x4001 5800 - 0x4001 5BFF	1 KB	DBGMCU	

总线	编址范围	大小	外设	备注
	0x4001 4C00 - 0x4001 57FF	3 KB	Reserved	
	0x4001 4800 - 0x4001 4BFF	1 KB	TIM17	
	0x4001 4400 - 0x4001 47FF	1 KB	TIM16	
	0x4001 4000 - 0x4001 43FF	1 KB	Reserved	
	0x4001 3C00 - 0x4001 3FFF	1 KB	CMP&OPAMP	
	0x4001 3800 - 0x4001 3BFF	1 KB	USART1	
	0x4001 3400 - 0x4001 37FF	1 KB	Reserved	
	0x4001 3000 - 0x4001 33FF	1 KB	SPI1/I2S1	
	0x4001 2C00 - 0x4001 2FFF	1 KB	TIM1	
	0x4001 2800 - 0x4001 2BFF	1 KB	Reserved	
	0x4001 2400 - 0x4001 27FF	1 KB	ADC	
	0x4001 0800 - 0x4001 23FF	7 KB	Reserved	
	0x4001 0400 - 0x4001 07FF	1 KB	EXTI	
	0x4001 0000 - 0x4001 03FF	1 KB	SYSCFG	
	0x4000 7400 - 0x4000 FFFF	35 KB	Reserved	
	0x4000 7000 - 0x4000 73FF	1 KB	PWR	
	0x4000 5800 - 0x4000 6FFF	6 KB	Reserved	
	0x4000 5400 - 0x4000 57FF	1 KB	I2C1	
	0x4000 3400 - 0x4000 53FF	8 KB	Reserved	
	0x4000 3000 - 0x4000 33FF	1 KB	IWWDG	
	0x4000 2C00 - 0x4000 2FFF	1 KB	WWDG	
	0x4000 2800 - 0x4000 2BFF	1 KB	RTC	
	0x4000 2400 - 0x4000 27FF	1 KB	Reserved	
	0x4000 2000 - 0x4000 23FF	1 KB	TIM14	
	0x4000 0800 - 0x4000 1FFF	6 KB	Reserved	
	0x4000 0400 - 0x4000 07FF	1 KB	TIM3	
	0x4000 0000 - 0x4000 03FF	1 KB	TIM2	
	0x2000 1000 - 3FFF FFFF	~512 MB	Reserved	
SRAM	0x2000 0000 - 0x2000 0FFF	4 KB	SRAM	
	0x1FFF FC00 - 0x1FFF FFFF	1 KB	Reserved	
Info	0x1FFF F800 - 0x1FFF FBFF	1 KB	Option bytes	
	0x1FFF EC00 - 0x1FFF F7FF	3 KB	System memory	
	0x0800 8000 - 0x1FFF EBFF	~384 MB	Reserved	
Flash	0x0800 0000 - 0x0800 7FFF	32 KB	Main Flash memory	
	0x0000 8000 - 0x07FF FFFF	~128 MB	Reserved	
	0x0000 000 - 0x0000 7FFF	32 KB	主闪存存储器 ， 系统存储器或是 SRAM @依赖 BOOT 的配置	

3.2.3 SRAM

最大 4K 字节的内置静态 SRAM。它可以以字节(8 位)、半字(16 位)或字(32 位)进行访问。

CPU 及 DMA 都可用最快的系统时钟且不插入任何等待进行访问 SRAM。

奇偶校验检查 Parity check

用户可以使用用户选项字节（option byte）中的选项位 RAM_PARITY_CHECK 来启用奇偶校验。

数据总线宽度为 36 位，其中 32 位为数据，4 位用于每字节的奇偶校验位（1bit/byte）。以此来增强数据存储的鲁棒性，来适应欧洲 Class B 及 SIL 规范的数据安全要求。

当写入 SRAM 时，奇偶校验位被计算和存储。当读取时 MCU 自动校验。如果 1 bit 失败，则产生 NMI 中断。当配置寄存器 SYSCFG_CFGR2 中的 SRAM_PARITY_LOCK 控制位，此错误还可以链接到 TIM1/16/17 的 BRK_IN 输入，此 SRAM 校验错误标志（SRAM_PEF）通过 SYSCFG_CFGR2 寄存器进行访问。

注：当启用 SRAM 奇偶校验时，建议在代码开始时通过软件初始化整个 RAM 内存，以避免在读取未初始化位置时出现奇偶校验错误。

3.2.4 FLASH 概述

最大 32K 字节的内置闪存存储器，用于存放程序和数据。

FLASH 存储器有两个不同存储区域：

- ✧ 主闪存在储块（main flash block），它包括应用程序和用户数据区（若需要时 ）
- ✧ 信息块(information block)，其包含两个部分：
  - 选项字节 (Option bytes)，内含硬件及存储保护用户配置选项。
  - 系统内存 (System memory)，其包含专有 boot loader 代码。参见<嵌入式闪存>章节。

闪存接口基于 AHB 协议执行指令和数据存取。其预取缓冲的功能可加速 CPU 执行代码的速度。

3.2.5 启动配置

在启动时，通过自举引脚和自举选项控制字可以选择三种自举模式中的一种：

启动模式选择		启动模式	说明
nBOOT1 bit	BOOT0 pin		
x	0	主闪存存储器	主闪存存储器选为启动区域
1	1	系统存储器	系统存储器选为启动区域
0	1	内置 SRAM	内置 SRAM 选为启动区域

器件复位后，在复位预热的结束段锁存 BOOT0 pin 和 nBOOT1 bit 的值，用户可通过设置 nBOOT1 bit 和 BOOT0 pin 来选择启动模式。

从待机模式（standby）唤醒时，CPU 会重新采样 BOOT0 及 nBOOT1 的值，因此在有待机（standby）应用的场合需要保持启动模式的设置。

在启动延迟之后，CPU 从地址 0x0000 0000 获取堆栈顶的地址，并从启动存储器的 0x0000 0004 指示的地址开始执行代码。

根据选定的启动模式，主闪存存储器，系统存储器或 SRAM 按照以下说明访问：

- ✧ 从主闪存存储器启动： 主闪存存储器被映射到启动存储空间 (0x0000 0000)，但仍然能从原有的地址空间 (0x0800 0000) 访问。即闪存存储器的内容可从两个地址开始访问，0x0000 0000 或 0x0800 0000。
- ✧ 从系统存储器启动： 系统存储器被映射到启动空间 (0x0000 0000)，但仍然能够在它原有的地址空间 (0x1FFF EC00) 访问。
- ✧ 从内置的 SRAM 启动： SRAM 映射到启动空间 (0x0000 0000)，但仍然能够在它原有的地址空间 (0x2000 0000) 访问。

物理重映射（Physical remap）

选择了引导模式后，应用程序软件就可以修改引导模式。这个修改是通过编程 SYSCFG 配置寄存器 1 (SYSCFG_CFGR1) 中的 MEM_MODE 位来执行的。与 Cortex®M3 和 M4 不同，M0 CPU 不支持向量表的重定位。对于位于与 0x0800 0000 不同地址的应用程序代码，必须添加一些额外的代码，以便能够为应用程序中断提供服务。

一个解决方案是通过软件将矢量表重新定位到内部 SRAM：

1、从闪存 flash(映射在应用程序加载地址的基址上)复制向量表到 SRAM 的基址 0x2000 0000。

2、重新映射 SRAM 地址至 0x0000 0000，使用 SYSCFG_CFGR1 寄存器配置。

3、一旦中断发生，Cortex-M0 处理器将从 SRAM 中重新定位的矢量表中获取中断处理程序的起始地址，然后它将跳转到 Flash 中执行中断处理程序。

注：这个操作应该在应用程序的初始化阶段完成。请参考 IAP 应用例程，以了解更多细节。

内嵌的自举程序（Boot loader）

内嵌的自举程序存放在系统存储器，由原厂在生产时写入。该程序可以通过 USART1 对闪存进行重新编程。

4 嵌入式闪存（FLASH）

4.1 特性

✧ 高达 32K 字节闪存存储器
✧ 存储器结构：
- 主闪存模块（main flash block）：8K 字（8K×32 位）
- 信息模块（information block）：1K 字（1K×32 位）

闪存接口的特性为：
✧ 带预取缓冲器的读接口 （ 每字为 2×64 位 ）
✧ 选择字节加载器
✧ 闪存编程 / 擦除操作
✧ 访问 / 写保护
✧ 低功耗模式

4.2 功能描述

4.2.1 闪存结构

闪存空间由 32 位宽的存储单元组成，既可以存代码又可以存数据。主闪存块按 32 页（每页 1K 字节）或 8 扇区（每扇区 4K 字节）分块，以扇区为单位设置写保护（参见存储保护相关内容）。

Flash 模块结构

模块	Flash 存储器地址	大小（byte）	名称	描述
主存储块 Main flash block	0x0800 0000 - 0x0800 03FF	1K	页 0	扇区 0
	0x0800 0400 - 0x0800 07FF	1K	页 1	
	0x0800 0800 - 0x0800 0BFF	1K	页 2	
	0x0800 0C00 - 0x0800 0FFF	1K	页 3	
	……	……	……	……
	0x0800 7000 - 0x0800 73FF	1K	页 28	扇区 7
	0x0800 7400 - 0x0800 77FF	1K	页 29	
	0x0800 7800 - 0x0800 7BFF	1K	页 30	
	0x0800 7C00 - 0x0800 7FFF	1K	页 31	
信息块 Infomation block	0x1FFF EC00 - 0x1FFF F7FF	3K	-	系统存储器
	0x1FFF F800 - 0x1FFF F80F	16	-	Option byte
闪存接口寄存器	0x4002 2000 - 0x4002 2003	4		FLASH_ACR
	0x4002 2004 - 0x4002 2007	4		FALSH_KEYR
	0x4002 2008 - 0x4002 200B	4		FLASH_OPTKEYR
	0x4002 200C - 0x4002 200F	4		FLASH_SR
	0x4002 2010 - 0x4002 2013	4		FLASH_CR
	0x4002 2014 - 0x4002 2017	4		FLASH_AR
	0x4002 2018 - 0x4002 201B	4		保留
	0x4002 201C - 0x4002 201F	4		FLASH_OBR
	0x4002 2020 - 0x4002 2023	4		FLASH_WRPR

4.2.2 读操作

嵌入式 Flash 模块可以像普遍存储空间一样直接寻址访问。任何对 Flash 模块内容的读操作都须经过专门的判断过程。

取指令和取数据都是通过相同的 AHB 总线读取访问，通过设置 Flash 访问控制寄存器（Flash_ ACR），能够控制读读操作按照所指定的方式执行：

取指：预取指缓冲使能，可提高 CPU 运行速度

延迟：缓读等待的时钟数设置（0 或 1），以保证正确的读操作。

4.2.3 取指

Cortex-M0 通过 AHB 总线取指。预取指模块的旨在提高取指效率。

4.2.4 预取缓冲区

预取指缓冲区分 3 块，每块 8 个字节，其位宽与 Flash 相同，能够完全替代一次同样大小的读取访问。预取缓冲区的工作使得更快速的 CPU 执行成为可能，CPU 每次取一个字，下一个字在预取缓冲区中随时可用。这意味着，如果代码按照 32 位对齐，可以达到 2 倍的取指加速。

但只有 wait state number 为 1 时，预取缓冲区才会起作用。在无等待状态下（no wait state），无论预取缓冲区状态如何，性能都保持不变。可能会对功耗产生一些影响，但这很大程度上依赖于实际的应用程序代码。

4.2.5 预取控制器

预取控制器根据预取缓冲区的可用空间决定是否访问闪存。当预取缓冲区中至少有一个块空闲时，控制器才会发起读请求。

复位后，预取缓冲区的状态默认为打开。

预取缓冲区通常在初始化程序期间打开/关闭，此时的 mcu 在内部 8 MHz (HSI16M/2)振荡器上运行。

4.2.6 访问延迟

为了保护对 Flash 的正确读取，必须在 Flash 访问控制寄存器中的 LATENCY[2:0] 中指定预取指控制器的速度比，这个数值等于每次访问 Flash 后到下次访问之间所需插入的等待周期的个数。复位后，这个值默认为零，也就是没有插入等待周期的状态。

4.2.7 FLASH 的写/擦操作

嵌入式闪存支持在电路编程（ICP）以及在应用编程（IAP）。

ICP（In-circuit programming），使用 SWD 或 Bootloader 的方法对 flash 内容进行在线写/擦。

IAP（In-application programming），通过使用 MCU 支持的任意通讯接口（I/Os，USB，CAN，UART，I2C，SPI 等）下载程序或者数据。IAP 允许用户在运行程序的过程中重写应用程序，前提是一部分应用程序必须预先用 ICP 的方法烧写进去。

烧写和擦除操作支持工作全电压范围，相关操作寄存器如下：

- 密钥寄存器（FLASH_KEYR）
- 选项字节密钥寄存器 (FLASH_OPRKEYR)
- Flash 控制寄存器 (FLASH_CR)
- Flash 状态寄存器 (FLASH_SR)
- Flash 地址寄存器 (FLASH_AR)
- 选项字节寄存器 (FLASH_OBR)
- 写保护寄存器 (FLASH_WRPR)

只要 CPU 不去访问 Flash 空间，进行中的 Flash 写操作不会妨碍 CPU 的运行。也就是说，在对 Flash 进行写 / 擦除操作的同时，任何对 Flash 的访问都会令总线停顿，直到写/擦除操作完成后才会继续执行，这意味着在写/擦除 Flash 的同时不可以对它取指和访问数据。

在对 flash 空间进行写/擦除操作时，内部 RC 振荡器(HSI)必须打开。

FLASH 空间解锁

复位后，Flash 存储器默认是受保护状态的，这样可以防范意外的擦除动作。FLASH_CR 寄存器的改写具有加锁保护，只有对 FLASH_KEYR 寄存器执行解锁序列操作才能开启对 FLASH_CR 的访问权限。操作序列由下面 2 个写操作构成：

- 写密钥 1（KEY1）=0x45670123
- 写密钥 2（KEY2）=0xCDEF89AB

任何错误的顺序将会锁死 FLASH_CR 直至下次复位。

当发生密钥错误时，会产生总线错误（bus error）并触发一次硬件错误（Hardfautlt）中断。如果 KEY1 不匹配，

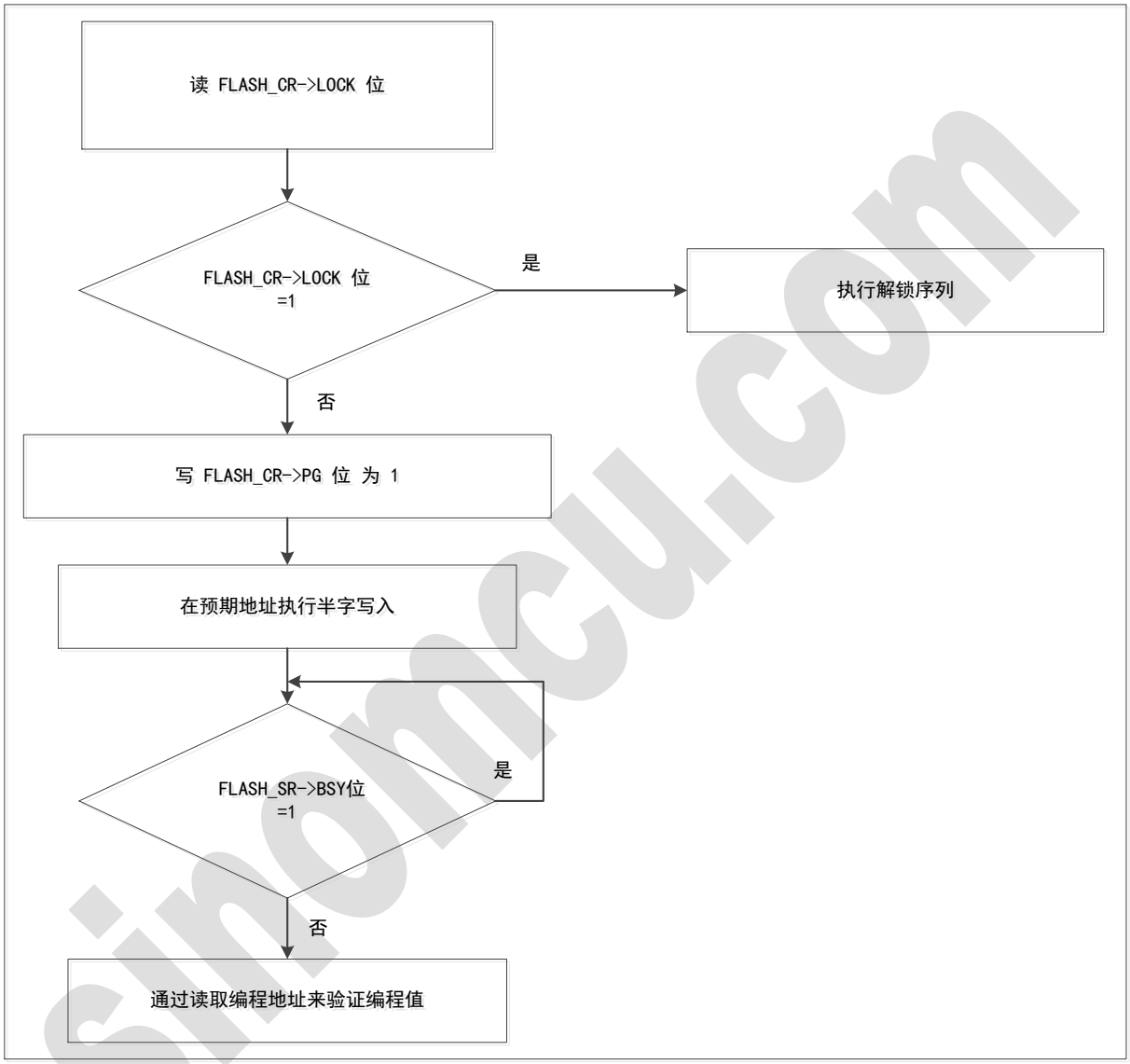
写操作后立即发生中断，如果 KEY1 正确但 KEY2 不匹配，就会在 KEY2 写操作后触发中断。

对寄存器 FLASH_CR 的 LOCK 位写 1，FLASH_CR 寄存器可以再次被用户软件锁定。

主闪存编程（Main Flash Memory Programming）

主闪存一次可以编程 16bit。当 FLASH_CR 中的 PG 位为 1 时，直接对相应的地址写一个半字（16 bit），就是一次编程操作。如果试图写其他的长度而不是半字，将引起硬件错误（Hardfault）中断。

编程流程图



Flash 存储器接口会预读一下待编程字节是否已被擦除（值是否为全 1），如果不是，那么编程操作会自动取消， 并且 FLASH_SR 寄存器的 PGERR 位被置起，提示编程错误告警。如果被编程的内容为全零（0x0000），则会例外，这时会正确编程并且不告警。

如果待编程地址所对应的 FLASH_WRPR 中的写保护位有效，同样也不会有编程动作，同时 FLASH_SR 寄存器的 WRPRTERR 位被置起，提示编程错误告警。FLASH_SR 寄存器中的 EOP 位在编程动作结束置起，提示编程结束。

主闪存标准编程流程如下：

1、检查 FLASH_SR 中的 BSY 位，以确认上次编程操作已经结束。

2、置 FLASH_CR 寄存器中的 PG 位。

3、以半字（16bit）为单位向目标地址写入数据。

4、等待 FLASH_SR 寄存器中的 BSY 位归零。

5、检查 FLASH_SR 中的 EOP 位，编程成功后 EOP 置起，软件清零。

6、读数据以校验。

注：当 FLASH_SR 中得 BSY 位为 1 的时候，这些寄存器不可访问。

FLASH 存储器擦除（Flash Memory Erase）

Flash 存储器支持页擦（Page Erase）和全擦（Mass Erase）。

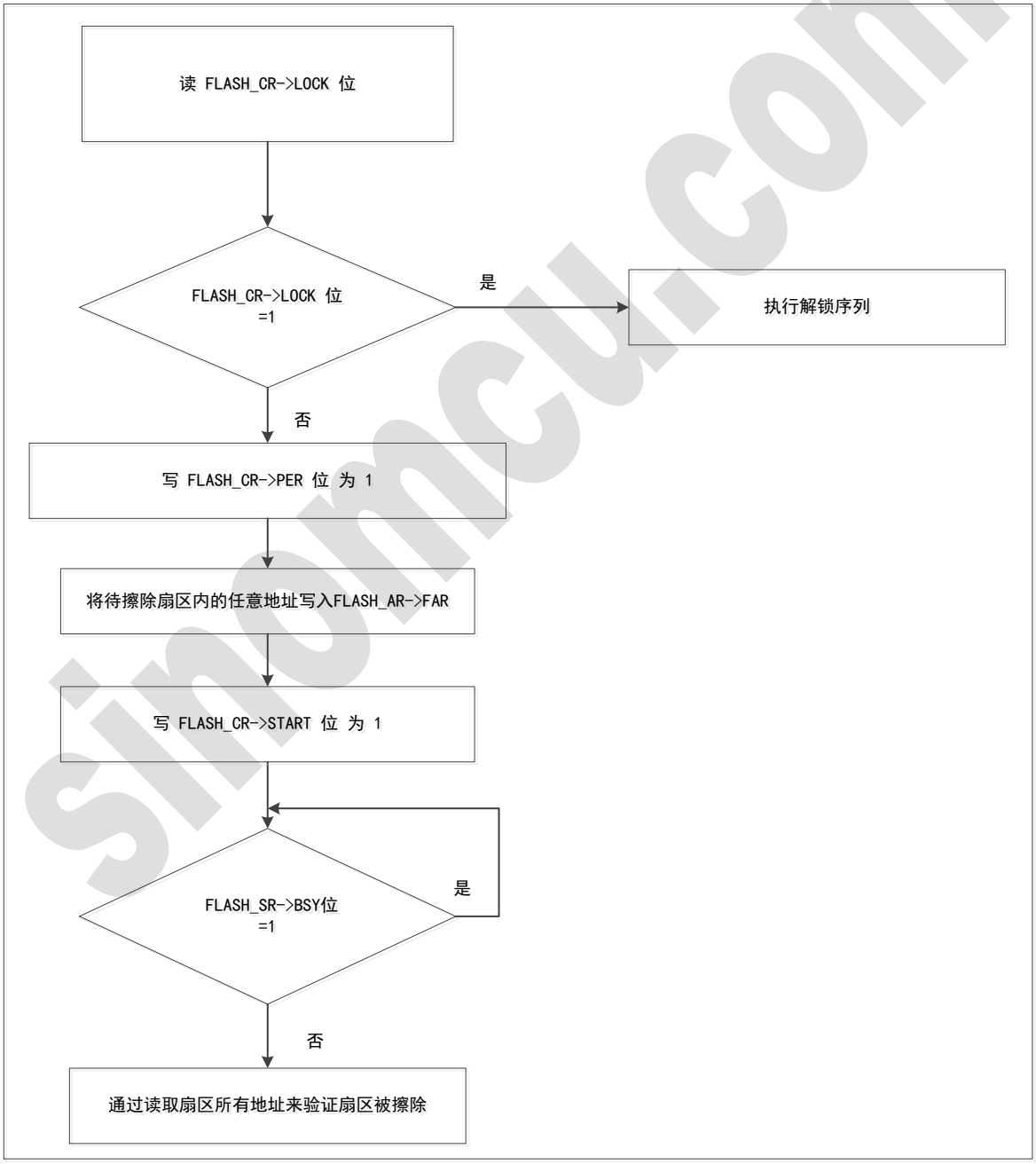
页擦（Page Erase）

页擦操作步骤如下：

1、检查 FLASH_SR 中的 BSY 位，以确认上次编程操作已经结束。
2、置 FLASH_CR 寄存器中得 PER 位为 1。
3、写 FLASH_AR 寄存器以选择待擦除的页。
4、置 FLASH_CR 寄存器中的 STRT 位为 1。
5、等待 FLASH_SR 中的 BSY 位归零。
6、检查 FLASH_SR 中的 EOP 位，编程成功后 EOP 置起，软件清零。
7、读取已擦除页以校验。

注：设置 STRT 位后，至少等待 1 个 CPU 周期，软件开始检查 BSY 位是否等于“0”。

页擦流程图



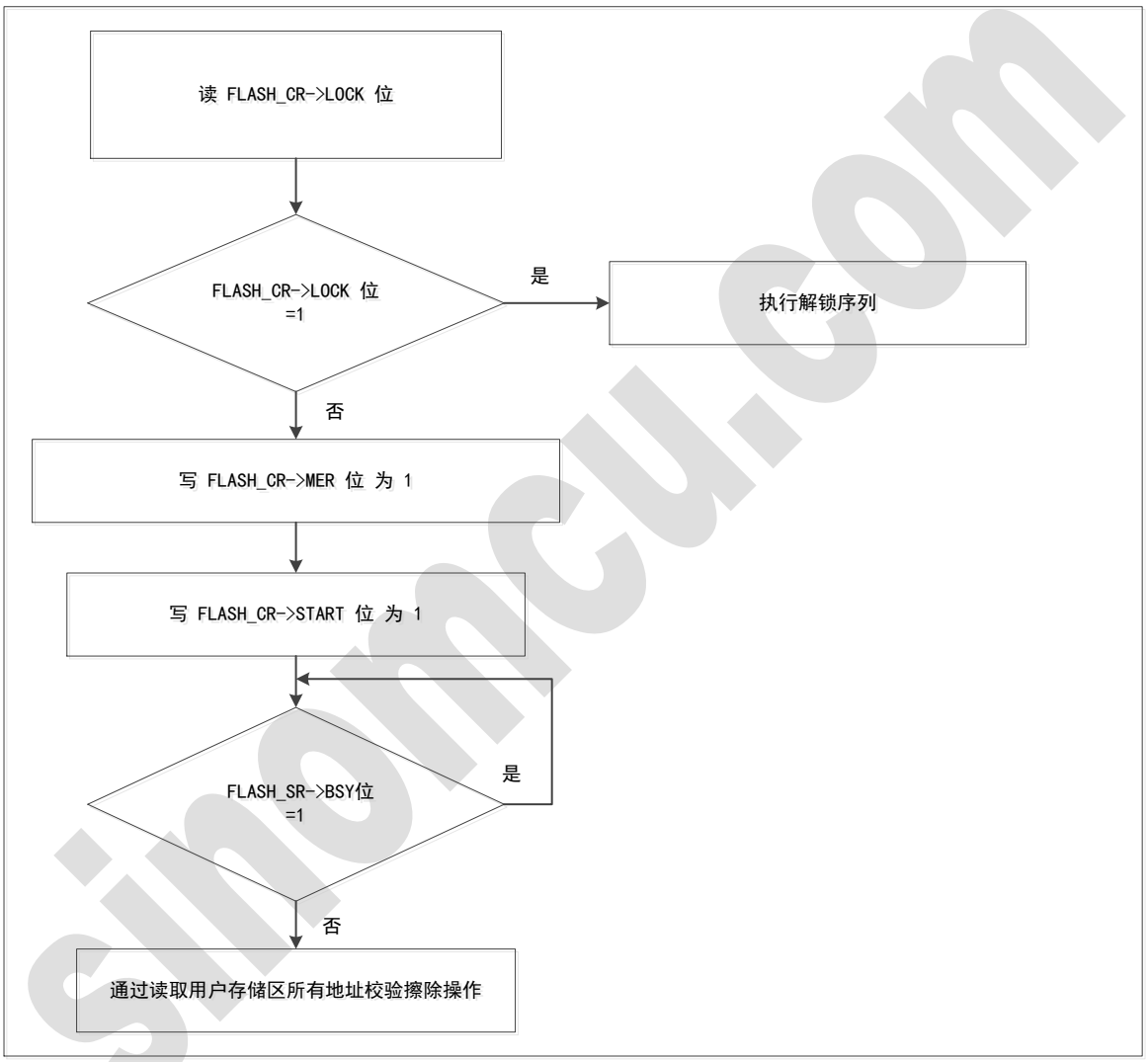
全擦（Mass Erase）

全擦命令可以用来一次擦除主闪存区（main flash block）所有页（page）；但信息块（information block）不受这个命令影响，具体操作步骤如下：

1、检查 FLASH_SR 中的 BSY 位，以确认上次编程操作已经结束。
2、置 FLASH_CR 寄存器中的 MER 位为 1。
3、置 FLASH_CR 寄存器中的 STRT 位为 1。
4、等待 FLASH_SR 中的 BSY 位归零。
5、检查 FLASH_SR 中的 EOP 位，编程成功后 EOP 置起，软件清零。
6、读取全部页并校验。

注：设置 STRT 位后，至少等待 1 个 CPU 周期，软件开始检查 BSY 位是否等于“0”。

FLASH 全擦流程



选项字节编程

选项字节的编程与常规用户地址不同，总共就是 3 个字节（1 个写保护，1 个读保护，1 个硬件配置）。解除 Flash 访问限制后，还需要针对 FLASH_OPTKEYR 寄存器完成密钥写入操作。完成该操作后，FLASH_CR 寄存器中的 OPTWRE 位会被置 1，然后就可以先置位 FLASH_ CR 中的 OPTPG 位，再按半字单位写目标地址。

选项字节编程同样会自动检查选项字节是否已经被擦除（值是否为 1），如果不是，那么编程操作会自动取消，并且 FLASH_SR 中的 WRPRTERR 位被置位，提示编程错误告警。FLASH_SR 寄存器中的 EOP 位在编程动作结束置起，提示编程结束。

在编程操作开始前，选项字节会自动填充到下个闪存地址，以保证选项字节及其反码总是对的。具体操作步骤如下：

1、检查 FLASH_SR 寄存器中的 BSY 位，以确保上次编程结束。
2、解锁 FLASH_CR 寄存器中的 OPTWRE 位。
3、置 FLASH_CR 寄存器中的 OPTPG 位为 1。
4、写数据（半字）到目标地址

5、等待 FLASH_SR 中的 BSY 位归零。
6、读取编程字节并校验
当 flash 读保护选项字节由保护状态被改成非保护状态时，会自动引发一次整片擦除。如果用户只想改写其他的字节，则不会引发整片擦除，这个机制用于保护 Flash 的内容。

选项字节擦除过程

选项字节的擦除过程如下：
1、检查 FLASH_SR 寄存器中的 BSY 位，以确保上次编程结束。
2、解锁 FLASH_CR 寄存器中的 OPTWRE 位。
3、置 FLASH_CR 寄存器中的 OPTER 位为 1。
4、置 FLASH_CR 寄存器中的 STRT 位为 1。
5、等待 FLASH_SR 中的 BSY 位归零。
6、读取擦除字节并校验

4.3 存储器保护

Flash 的用户区可以被保护，以防止被不信任代码读取。Flash 存储的每个页（pages）也可以被保护，防止程序跑飞导致的意外擦除，写保护的最小单元为 1 个扇区（sector），即 4 个页（pages）。

4.3.1 读保护

设置选项字节的 RDP 字节对应位，然后系统复位，新的 RDP 被加载，读保护被激活。
注：如果读保护被置位时，仍然通过 SWD 连着调试器，可以用 POR（上电复位）代替系统复位。
芯片支持 3 级都保护：LEVEL0(无保护)~LEVEL2(最大限度或禁止调试)

读保护级别及 RDP 对应关系

RDP 字节值	RDP 反码值	读保护级别
0xxAA	0x55	LEVEL 0
任意值 除 0xAA 和 0xCC 外	任意值（不要求互补） 除 0x55 和 0x33 外	LEVEL 1（默认）
0xCC	0x33	LEVEL 2

系统存储区不受读保护字节的影响，但该区域不允许编程和擦除操作。

LEVEL0：无保护

针对主 Flash 区域的读写和擦除操作都被允许，选项字节也全都可以操作。

LEVEL1：读保护

这是 RDP 选项字节被擦除之后的默认保护级别。对应的 RDP 值为除 0xAA 和 0xCC 以外的任意值，包括反码不正确情况。

用户模式：在用户模式下执行的代码允许对主 Flash 和选项字节做全部操作。

Debug、boot RAM 和 boot loader 模式：在调试模式下（with SWD）或运行在 boot RAM 和 boot loader 状态下，主 Flash 区和备份寄存器（RTC_BKPxR）均不允许访问。在该状态下，任何简单的读访问都会引起总线错误并触发硬件错误（Hardfault）中断。主 Flash 区同时也禁止写和擦除操作，以防范恶意程序修改代码，任何尝试擦写的操作都会引起 FLASH_SR 中的 PGERR 标志置位。

当 RPD 字节的内容被修改为 0xAA，会先执行全擦除操作，然后改为 LEVEL 0 的级别，同时备份寄存器的值也会被复位。

LEVEL 2：No debug

在这个级别上，包含 LEVEL 1 的保护功能，除此之外，Cortex-M0 的调试功能被禁止，所以 SWD 调试口、boot RAM 和 boot loader（boot from system memory）都不再有效。

在用户执行模式下，允许对主 Flash 区做全部操作，相反，对于选项字节却只能读取和写入而并不能做擦除。

此外，RDP 字节不能再改写，因此，LEVEL 2 这个保护级别永远不能清除掉，是一个不可恢复性的操作。当试图改写 RDP 字节时，FLASH_SR 寄存器中的保护错误标志 WRPRTERR 会被置位并引发一个中断。

注 1：在复位条件下，调试功能不能用。
注 2：原厂也不能对打开了 Level 2 保护功能的器件做分析处理。

保护状态和保护级别及运行模式对照表

区域	保护级别	用户代码执行			调试/从 RAM 启动/从系统区域启动		
		读	写	擦除	读	写	擦除
主 Flash 区	1	Yes	Yes	Yes	No	No	No *注 3*
	2	Yes	Yes	Yes	N/A *注 1*	N/A *注 1*	N/A *注 1*
系统存储区	1	Yes	No	No	Yes	No	No

注 2	2	Yes	No	No	NA *注 1*	N/A(1)	N/A *注 1*
选项字节	1	Yes	Yes *注 3*	Yes	Yes	Yes *注 3*	Yes
	2	Yes	Yes *注 4*	No	N/A *注 1*	N/A *注 1*	N/A *注 1*
备份寄存器	1	Yes	Yes	N/A	No	No	Yes
	2	Yes	Yes	N/A	N/A *注 1*	N/A *注 1*	N/A *注 1*

注 1：当 Level 2 保护级别使能，调试口、从 RAM 启动和从系统存储区（SYSTEM ROM）启动都被禁止。
注 2：系统存储区是唯一在任何情况下可读的区域。
注 3：当 RDP 被修改为不保护时，主 Flash 区会被擦除。
注 4：所有的选项字节中，除 RDP 字节外都能被再次编程。

改变读保护级别

改变 RDP 的值到其他值（除 0xCC 以外）就可以轻松的从 LEVEL 0 级迁移到 LEVEL 1 级别。将 RDP 写成 0xCC，就可以直接进入 LEVEL 2 级别。相反的，绕开整片擦除动作而进入 LEVEL 0（无保护）级别是不可能的。一旦 RDP 写入 0xAA，就会产生整片擦除（Mass Erase）动作。

注 1：当整片擦除命令执行时，备份寄存器（RTC_BKPxR in the RTC）也被复位。
注 2：为了确保保护级别生效，选项字节必须通过 Flash 控制寄存器中的 OBL_LAUNCH 位强制重新加载。

4.3.2 写保护

写保护以一个扇区（sector）为单位（4 pages）来控制， 配置选项字节中的 WRP[1:0] 位， 然后通过 FLASH_CR 寄存器中的 OBL_LAUNCH 位强制重新加载选项字节就可以使能这个保护。如果试图写入或擦除一个受保护的扇区，会引起 FLASH_SR 中的 WRPRTERR 标志位被置位。

写保护的解除

解除写保护有 2 个应用样例可以提供：

Case 1：在解除写保护后禁止读保护

1、使用 FLASH_CR 中的 OPTER 位擦除整个选项字节区域。
2、向 RDP 写入 0xAA 从而解除所有保护，这自然会引起整片擦除。
3、设置 FLASH_CR 中的 OBL_LAUNCH 位，重新加载选项字节(和新 WRP[1:0]位) ，写保护解除。

Case 2：在解除写保护后，读保护仍然有效；这种办法对用户使用 boot loader 进行在应用编程（IAP）时很有用。

1、使用 FLASH_CR 中的 OPTER 位擦除整个选项字节区域。
2、设置 FLASH_CR 中的 OBL_LAUNCH 位，重新加载选项字节(和新 WRP[1:0]位) ，写保护解除。

4.3.3 选项字节写保护

选项字节默认写保护开启并可读。为了对选项字节进行写/擦除操作，必须对 OPTKEYR 顺序写入密钥（类似解锁）。正确的密钥序列会引起 FLASH_CR 中的 OPTWRE 置位，表明解锁成功。同样，通过对该位清零，能够再度禁止对选项字节的写操作。

4.4 FLASH 中断

Flash 中断请求

中断事件	事件标志	使能控制位
操作结束	EOP	EOPIE
写保护错误	WRPRTERR	ERRIE
编程错误	PGERR	ERRIE

4.5 选项字节说明

本芯片支持 8 个选项字节，用户可根据需求进行配置。

4.5.1 选项字节格式

每个 32 位的选项字格式如下：

Bit[31:24]	Bit[23:16]	Bit[15:8]	Bit[7:0]
选项字节 1 反码	选项字节 1	选项字节 0 反码	选项字节 0

注：反码需要软件写入。

4.5.2 选项字节加载

每次上电复位时，选项字节加载逻辑从 Information Block 读取选项字节内容，并将数据存储到选项字节寄存器

(FLASH_OBR)和写保护寄存器(FLASH_WRPR)。在选项字节加载期间，将对选项字节和相应的反码选项字节进行互补校验，若校验失败，生成选项字节错误 (OPTERR)，对应的 option byte 被认定为 0xFF。

注：如果选项字节和它的反码选项字节都等于 0xFF(擦除态)，选项字节错误不会产生。

4.5.3 选项字节存储映射

地址	Bit[31:24]	Bit[23:16]	Bit[15:8]	Bit[7:0]
0x1FFF F800	nUSER	USER	nRDP	RDP
0x1FFF F804	nDATA1	DATA1	nDATA0	DATA0
0x1FFF F808	nWRP1	WRP1	nWRP0	WRP0

注 1：选项字节（用户 option 或读/写保护）配置，可以从上表中的存储器地址直接读取，也可以从选项字节寄存器（FLASH_OBR）读取获得。

注 2：新写入的选项字节(用户 option，读/写保护)在系统复位后不加载。要重新加载，需要 POR 或将 OBL_LAUNCH 位设置为'1'。

4.5.4 用户及读保护选项字节

User and Read Protection			地址偏移	0x1FFF F800		出厂值	0x00FF 55AA	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	nUSER							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
出厂值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	RAM_PARITY_CHECK	VDDA_MONITOR	nBOOT1	-	nRST_STDBY	nRST_STOP	WDG_SW
R/W	rw	rw	rw	rw	rw	rw	rw	rw
出厂值	1	1	1	1	1	1	1	1
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	nRDP							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
出厂值	0	1	0	1	0	1	0	1
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	RDP							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
出厂值	1	0	1	0	1	0	1	0

BIT[31:24]	nUSER	USER 反码
BIT[23:16]	USER	用户选项字节，详细功能如下，被加载到 FLASH_OBR 寄存器
BIT[23]	-	保留位。
BIT[22]	RAM_PARITY_CHECK	Ram 奇偶校验 0：使能 1：关闭
BIT[21]	VDDA_MONITOR	VDDA 电压监测 0：VDDA 电源监测关闭 1：VDDA 电源监测使能
BIT[20]	nBOOT1	连同 BOOT0 信号一起控制 BOOT 模式，详细参考 3.2.5 章节
BIT[19]	-	保留位
BIT[18]	nRST_STDBY	0：进入 STANDBY 模式产生复位 1：进入 STANDBY 模式不产生复位
BIT[17]	nRST_STOP	0：进入 STOP 模式产生复位 1：进入 STOP 模式不产生复位
BIT[16]	WDG_SW	0：硬件看门狗 1：软件看门狗
BIT[15:8]	nRDP	RDP 反码
BIT[7:0]	RDP	读保护选项字节 0xAA：LEVEL0(出厂值) 0xXX：除了 0xAA 和 0xCC 其他值， LEVEL1

		0xCC：LEVEL2 *注：读保护状态被加载到 FLASH_OBR 寄存器，读保护详细描述就参见 4.3.1 章节。*

4.5.5 用户数据选项字节

User DATA			地址偏移	0x1FFF F804		出厂值	0x00FF 00FF	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	nDATA1							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
出厂值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	DATA1							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
出厂值	1	1	1	1	1	1	1	1
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	nDATA0							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
出厂值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	DATA0							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
出厂值	1	1	1	1	1	1	1	1

BIT[31:24]	nDATA1	DATA1 反码
BIT[23:16]	DATA1	用户数据选项字节 1，被加载到 FLASH_OBR 寄存器
BIT[15:8]	nDATA0	DATA0 反码
BIT[7:0]	DATA0	用户数据选项字节 0，被加载到 FLASH_OBR 寄存器

4.5.6 写保护选项字节

通过设置写保护选项字节，可实现闪存存储器的写保护及指定扇区的写保护功能。

Write Protection			地址偏移	0x1FFF F808		出厂值	0x00FF 00FF	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	nWRP1							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
出厂值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	WRP1							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
出厂值	1	1	1	1	1	1	1	1
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	nWRP0							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
出厂值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	WRP0							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
出厂值	1	1	1	1	1	1	1	1

BIT[31:24]	nWRP1	WRP1 反码
BIT[23:16]	WRP1	FLASH 写保护选项字节 1，被加载到 FLASH_WRPR 寄存器
BIT[15:8]	nWRP0	WRP0 反码
BIT[7:0]	WRP0	FLASH 写保护选项字节 0，被加载到 FLASH_WRPR 寄存器

注：本芯片仅 WRP0 字节及 nWRP0 有效，每个 bit 控制 4K 空间的写保护；WRP1 字节及 nWRP1 为保留位，实际控制无

效。

4.6 相关寄存器

Flash 寄存器必须按 32 位的方式访问（半字节访问是不允许的）。

4.6.1 FLASH 寄存器汇总表

基址：0x4002 2000				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**FLASH_ACR**	0x0000 0000	FLASH 访问控制寄存器	4.6.2
0x04	**FLASH_KEYR**	0xXXXX XXXX	FLASH 密钥寄存器	4.6.3
0x08	**FLASH_OPTKEYR**	0xXXXX XXXX	FLASH 选项密钥寄存器	4.6.4
0x0C	**FLASH_SR**	0x0000 0000	FLASH 状态寄存器	4.6.5
0x10	**FLASH_CR**	0x0000 0000	FLASH 控制寄存器	4.6.6
0x14	**FLASH_AR**	0x0000 0000	FLASH 地址寄存器	4.6.7
0x18	RES.	-	-	-
0x1C	**FLASH_OBR**	0xXXXX XX0X	FLASH 选项字节寄存器	4.6.8
0x20	**FLASH_WRPR**	0xFFFF FFFF	FLASH 写保护寄存器	4.6.9

4.6.2 Flash 访问控制寄存器（FLASH_ACR）

FLASH_ACR			地址偏移	0x00		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	PRFTBS	PRFTBE	-	LATENCY[2:0]		
R/W	-	-	r	rw	-	rw	rw	rw
复位值	-	-	0	0	-	0	0	0

BIT[31:6]	-	保留，保持复位值
BIT[5]	PRFTBS	预取缓冲区状态 0：预取缓冲区被禁止 1：预取缓冲区被使能
BIT[4]	PRFTBE	预取缓冲区使能 0：禁止预取指 1：使能预取指
BIT[3]	-	保留，保持复位值
BIT[2:0]	LATENCY[2:0]	延迟控制，设置 SYSCLK（系统时钟）周期和 Flash 访问时间的比率关系 000：零等待，适用于 0<SYSCLK≤24MHz 001：1T 等待，适用于 0<SYSCLK≤48MHz 其他：保留

4.6.3 Flash 密钥寄存器（FLASH_KEYR）

FLASH_KEYR			地址偏移	0x04		复位值	0xXXXX XXXX	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	FKEYR[31:24]							

R/W	w	w	w	w	w	w	w	w
复位值	x	x	x	x	x	x	x	x
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	FKEYR[23:16]							
R/W	w	w	w	w	w	w	w	w
复位值	x	x	x	x	x	x	x	x
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	FKEYR[15:8]							
R/W	w	w	w	w	w	w	w	w
复位值	x	x	x	x	x	x	x	x
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	FKEYR[7:0]							
R/W	w	w	w	w	w	w	w	w
复位值	x	x	x	x	x	x	x	x

BIT[31:0]	FKEYR[31:0]	Flash 密钥，此位输入正确密钥，解锁 Flash。

注：此寄存器全部只写（W），回读固定返回 0；

4.6.4 Flash 选项密钥寄存器（FLASH_OPTKEYR）

FLASH_OPTKEYR			地址偏移	0x08		复位值	0xXXXX XXXX	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	OPTKEYR[31:24]							
R/W	w	w	w	w	w	w	w	w
复位值	x	x	x	x	x	x	x	x
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	OPTKEYR[23:16]							
R/W	w	w	w	w	w	w	w	w
复位值	x	x	x	x	x	x	x	x
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	OPTKEYR[15:8]							
R/W	w	w	w	w	w	w	w	w
复位值	x	x	x	x	x	x	x	x
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	OPTKEYR[7:0]							
R/W	w	w	w	w	w	w	w	w
复位值	x	x	x	x	x	x	x	x

BIT[31:0]	OPTKEYR[31:0]	选项字节密钥，此位输入正确密钥，解锁 OPTWRE。

注：此寄存器全部只写（W），回读固定返回 0；

4.6.5 Flash 状态寄存器（FLASH_SR）

FLASH_SR			地址偏移	0x0C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

控制位	-	-	EOP	WRPRTERR	-	PGERR	-	BSY
R/W	-	-	rc_w1	rc_w1	-	rc_w1	-	r
复位值	-	-	0	0	-	0	-	0

BIT[31:6]	-	保留，保持复位值
BIT[5]	EOP	操作结束（end of operation） 当 Flash 操作（写/擦除）完成时由硬件置 1 ，软件写 1 清零 *注 ： 用 EOP 可以判断是否每次操作都顺利完成*
BIT[4]	WRPRTERR	写保护错误标志（write protection error） 当对写保护区域的地址写操作时被硬件置 1，软件写 1 清零
BIT[3]	-	保留，保持复位值
BIT[2]	PGERR	写入错误标志（programming error） 当被编程区域的状态不为 ’0xFFFF’ 的情况下，执行写入操作时被硬件置 1 ，软件写 1 清零
BIT[1]	-	保留，保持复位值
BIT[0]	BSY	忙标志（busy） 该位标明 Flash 操作处于进行中。当开始 Flash 操作的时候被硬件置位，当操作结束时或发生错误时被硬件清零。

4.6.6 Flash 控制寄存器（FLASH_CR）

FLASH_CR			地址偏移	0x10		复位值	0x0000 0080	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	OBL_LAUNCH	EOPIE	-	ERRIE	OPTWRE	-
R/W	-	-	rw	rw	-	rw	rw	-
复位值	-	-	0	0	-	0	0	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	LOCK	STRT	OPTER	OPTPG	-	MER	PER	PG
R/W	rw	rw	rw	rw	-	rw	rw	rw
复位值	1	0	0	0	-	0	-	0

BIT[31:14]	-	保留，保持复位值
BIT[13]	OBL_LAUNCH	选项字节强制更新，当被写为 1 时，该位强制选项字节的重加载，该操作会引起系统复位。 0：无效 1：有效
BIT[12]	EOPIE	操作结束中断使能 该位使能操作结束中断，使得 FLASH_SR 中的 EOP 位变成 1 的时候产生中断请求 0：中断禁止 1：中断使能
BIT[11]	-	保留，保持复位值
BIT[10]	ERRIE	错误中断使能 该位使能操作错误中断，使得 FLASH_SR 中的 PGERR / WRPRTERR 位变成 1 的时候产生中断请求 0：中断禁止 1：中断使能
BIT[9]	OPTWRE	选项字节写使能 该位为 1 时，选项字节即允许改写。对 FLASH_OPTKEYR 寄存器写入正确的关键

		字序列就可以将它置 1。 该位可软件清零。
BIT[8]	-	保留，保持复位值
BIT[7]	LOCK	FLASH 锁定标志 只能写 1。当该位为 1 时，表明 Flash 为锁定状态。该位可以通过解锁密钥时序来清零， 当发现解锁不成功时，该位就一直为锁定为 1，除非下次复位重新操作。
BIT[6]	STRT	启动位 该位会触发一个擦除操作，仅由软件置 1，仅会在 BSY 被清零时清零。
BIT[5]	OPTER	选择选项字节擦除
BIT[4]	OPTPG	选择选项字节编程
BIT[3]	-	保留，保持复位值
BIT[2]	MER	全擦除（Mass Erase） 选择全擦除（所有用户 pages 擦除）
BIT[1]	PER	页擦除（Page Erase） 选择页擦除
BIT[0]	PG	编程 选择 FLASH 编程（写入）

4.6.7 Flash 地址寄存器（FLASH_AR）

FLASH_AR			地址偏移	0x14		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	FAR[31:24]							
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	FAR[23:16]							
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	FAR[15:8]							
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	FAR[7:0]							
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0

BIT[31:0]	FAR[31:0]	FLASH 地址 当 PG 位被选中时，选择待写入的地址，或当 PER 位被选中时，选择待擦除的页地址。

注 1：本寄存器由硬件根据当前和上次操作的地址更新。对于页擦除操作，该寄存器该由软件来更新以便匹配要擦除的页。

注 2：当 FLASH_SR 中的 BSY 位为 1 时，对这个寄存器的写访问将被阻止。

4.6.8 Flash 选项字节寄存器（FLASH_OBR）

FLASH_OBR			地址偏移	0x1C		复位值	0xXXXX XX0X	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	DATA1							
R/W	r	r	r	r	r	r	r	r
复位值	X	X	X	X	X	X	X	X
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	DATA0							
R/W	r	r	r	r	r	r	r	r
复位值	X	X	X	X	X	X	X	X
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

控制位	-	RAM_PARITY_CHECK	VDDA_MONITOR	nBOOT1	-	nRST_STDBY	nRST_STOP	WDG_SW
R/W	-	r	r	r	-	r	r	r
复位值	-	X	X	X	-	X	X	X
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	RDPRT[1:0]		OPTERR
R/W	-	-	-	-	-	r	r	r
复位值	-	-	-	-	-	X	X	X

BIT[31:24]	DATA1	DATA1
BIT[23:16]	DATA0	DATA0
BIT[15]	-	用户选项字节，保留位，保持复位值。
BIT[14]	RAM_PARITY_CHECK	用户选项字节，Ram 奇偶校验
BIT[13]	VDDA_MONITOR	用户选项字节，VDDA_MONITOR
BIT[12]	nBOOT1	用户选项字节，nBOOT1
BIT[11]	-	用户选项字节，保留位，保持复位值
BIT[10]	nRST_STDBY	用户选项字节，nRST_STDBY
BIT[9]	nRST_STOP	用户选项字节，nRST_STOP
BIT[8]	WDG_SW	用户选项字节，WDG_SW
BIT[7:3]	-	用户选项字节，保留位，保持复位值
BIT[2:1]	RDPRT[1:0]	读保护级别状态 00：读保护 LEVEL0（出厂默认配置） 01：读保护 LEVEL1 10/11：读保护 LEVEL2
BIT[0]	OPTERR	选项字节错误 此位置 1，表示加载的选项字节和反码字节不匹配；FLASH_OBR 或 FLASH_WRPR 寄存器相对应的字节被设置为 0xFF。

注：此寄存器的复位值取决于选项字节的配置值，OPTERR 位的复位值取决于复位时选项字节加载环节选项字节与反码校验结果。

4.6.9 Flash 写保护寄存器（FLASH_WRPR）

FLASH_WRPR			地址偏移	0x20		复位值	0xXXXX XXXX	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	WRP[31:24]							
R/W	r	r	r	r	r	r	r	r
复位值	X	X	X	X	X	X	X	X
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	WRP[23:16]							
R/W	r	r	r	r	r	r	r	r
复位值	X	X	X	X	X	X	X	X
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	WRP[15:8]							
R/W	r	r	r	r	r	r	r	r
复位值	X	X	X	X	X	X	X	X
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	WRP[7:0]							
R/W	r	r	r	r	r	r	r	r
复位值	X	X	X	X	X	X	X	X

BIT[31:0]	WRP[31:0]	FLASH 写保护 此寄存器包含从选项字节中加载到的写保护状态。

注：此寄存器的复位值取决于选项字节的配置值。

5 电源控制（PWR）

5.1 特性

✧ VDD：为 I/O 引脚、内部调压器、PVD、HSE 供电。

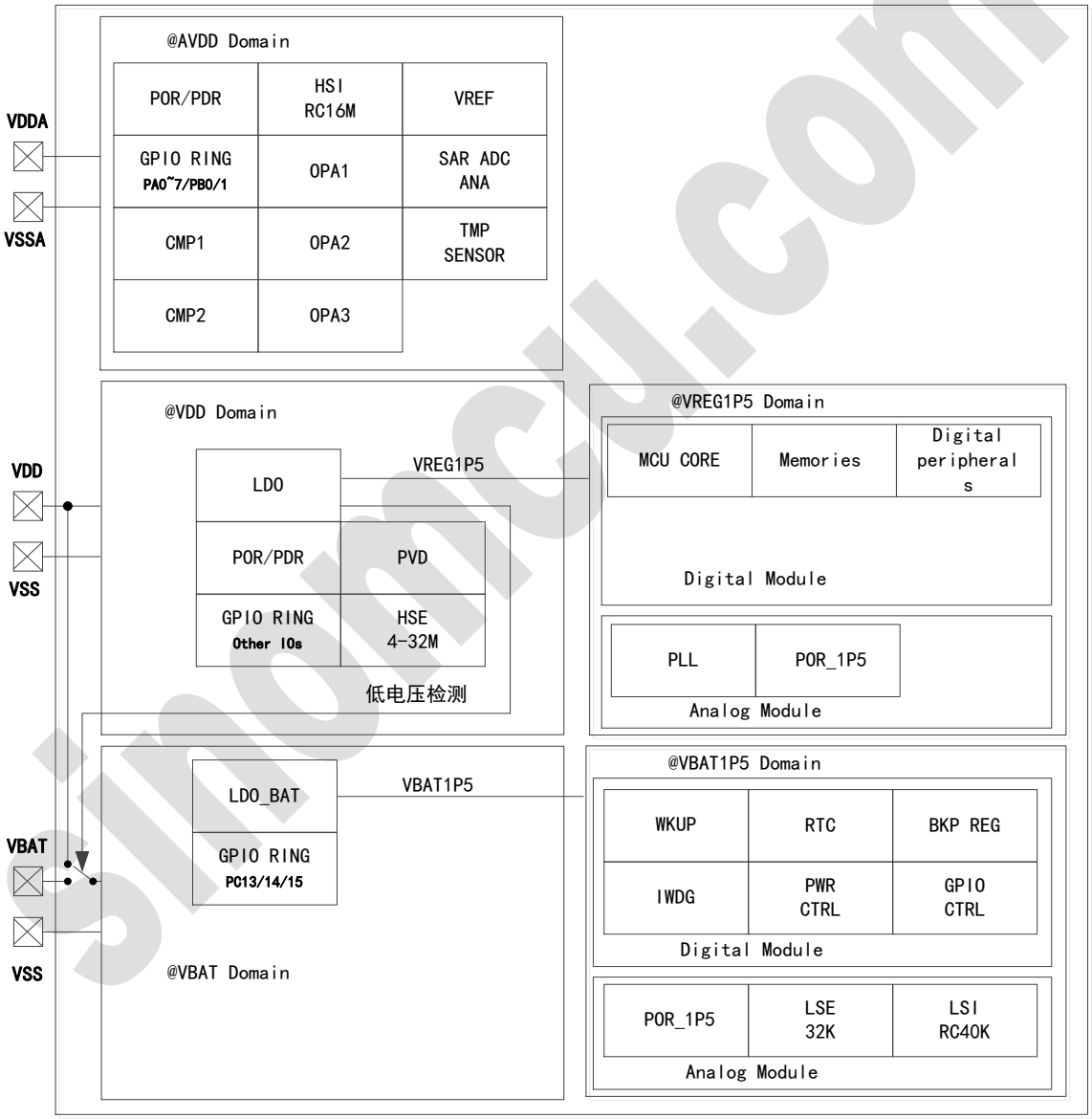
✧ VSSA，VDDA：为 ADC、复位模块、HIS、VREF、OPA、CMP 等模块供电。VDDA 和 VSSA 必须分别连接到 VDD 和 VSS。

✧ VBAT：当 VDD 不存在时，作为 RTC、32 kHz 外部时钟振荡器和备份寄存器的电源（通过电源开关供电）。

本产品内部集成了上电复位(POR)/掉电复位(PDR)电路，该电路始终处于工作状态，保证系统在供电超过 2.0V 时工作；当 VDD 低于设定的阀值(VPOR/PDR)时，置器件于复位状态，而不必使用外部复位电路。

器件中还有一个可编程电压监测器(PVD)，它监视 VDD 供电并与阀值 VPVD 比较，当 VDD 低于或高于阀值 VPVD 时产生中断，中断处理程序可以发出警告信息或将微控制器转入安全模式。PVD 功能需要通过程序开启。

5.2 电源框图



5.2.1 独立的 ADC 供电和参考电压

为了提高转换精度，ADC 使用用独立的电源供电（VDDA 和 VSSA），过滤和屏蔽来自电路板的噪声。

VDDA 供电或作为参考电压必须大于等于 VDD 电压。

当器件用单电源供电方式，则 VDDA 可通过外部滤波电路连接到 VDD，以确保 VDDA 参考电压无噪声。

当 VDDA 与 VDD 不同时，VDDA 必须大于等于 VDD 的供电电压。在通电或断电的过程中，为了保证 VDDA 和 VDD 之间安

全电位差，可在 VDD 和 VDDA 之间串接一个肖特基二极管。

5.2.2 电池备份域

当 VDD 掉电，为了保持备份寄存器的数据及保证 RTC 工作，VBAT 引脚可由独立的待机电源供电（电池或其它电源供电）。

VBAT 引脚和 VDD 经过内部电源模拟开关后给整个备份域供电，工作在备份域下的模块包括：LDO_VBAT、GPIO RING(PC13/14/15)、LSE 振荡器、LSI 振荡器、RTC、唤醒逻辑、IWDGD 等。当主电源 VDD 掉电后，VBAT 可以为 RTC 及备份域提供电源，电源开关由 PDR（PowerDownReset）控制。

注 1：在 VDD 上升阶段（tRSTTEMPO）或者检测到掉电复位（PDR）之后，VBAT 和 VDD 之间的电源开关仍会保持连接到 VBAT。

注 2：在 VDD 上升阶段，如果 VDD 在小于 tRSTTEMPO 的时间内达到稳定状态（tRSTTEMPO 请参考数据手册），且 VDD>VBAT+0.6V 时，电流可能通过 VDD 和 VBAT 之间的内部二极管注入到 VBAT。如果与 VBAT 连接的电源或者电池不能承受这样的注入电流，强烈建议在外部电源和 VBAT 引脚之间连接一个低压降二极管。

如果在应用中没有外接电池，建议 VBAT 通过一个 100nF 的陶瓷去耦电容与 VDD 外部相连。

当备份域由 VDD(内部模拟开关连到 VDD) 供电时，下述功能可用：

- PC13、PC14 和 PC15 可以作为 GPIO 口。
- PC13、PC14 和 PC15 可作为 TAMPER 引脚 ， RTC 校准时钟 ， RTC 闹钟、秒脉冲输出或 LSE （ 详见 RTC 章节)。

注： 因为模拟开关只能通过有限的电流（3mA），使用 PC13/PC14/PC15 作为输出功能是有限制的：

- *速度必须限制在 2MHz 以下@最大负载 30pF；*
- *这些 I/O 口绝对不能当作电流源 （ 如驱动 LED)。*

当备份域由 VBAT 供电时 (VDD 掉电后模拟开关连到 VBAT)，可以使用下述功能：

- PC13、PC14 和 PC15 只能用于通过 RTC 和 LSE 控制(详见 RTC 章节)。

5.2.3 电压调节器（Voltage Regulator）

芯片包含 2 个电压调节器： VDD 电压域下的 VREG1P5 和 VBAT 域下的 VBAT1P5，复位后电压调节器保持打开，VBAT1P5 任何模式都不会关闭，VREG1P5 根据应用可配置为三种不同的工作模式：

- 运行模式：调节器 VREG1P5 以全功耗模式为 1.5V 电压域（内核，内存和数字外设）提供电源。
- 停止模式：调节器 VREG1P5 以低功耗模式为 1.5V 电压域提供电源，为 VREG1P5 电压域下寄存器及 SRAM 保持数据。
- 待机模式：调节器 VREG1P5 断电，除了备份域电路外，VREG1P5 电压域下的寄存器和 SRAM 的内容全部丢失。

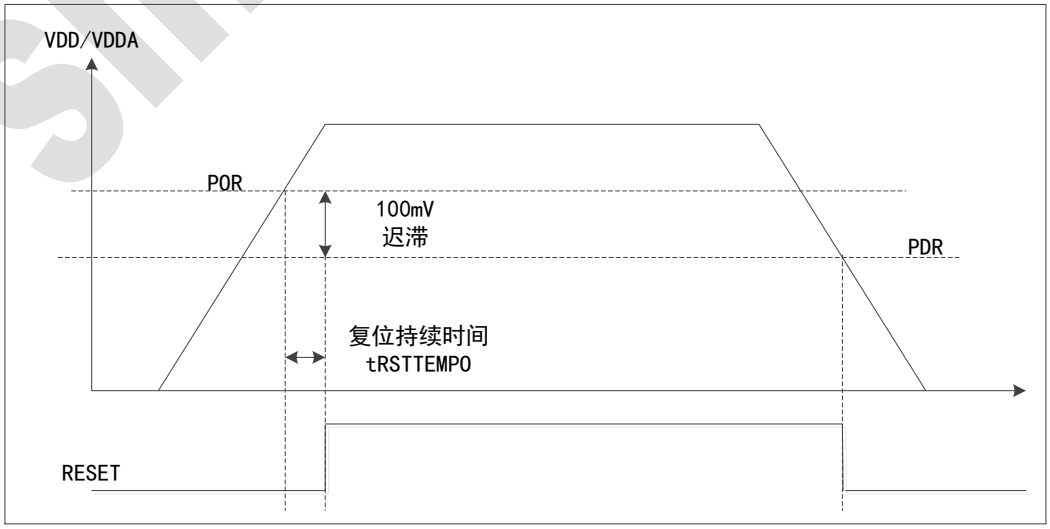
5.3 电源管理器

5.3.1 POR/PDR

芯片内置 POR(power on reset)和 PDR（power down reset）电路，一直保持有效，保证最低工作电压以上正常工作。

当被监测电源的电压低于限定门限时，器件维持复位状态，而无需外部复位电路。有关上电与掉电复位的电压门限请参见数据手册的电气特性章节。

POR/PDR 波形图



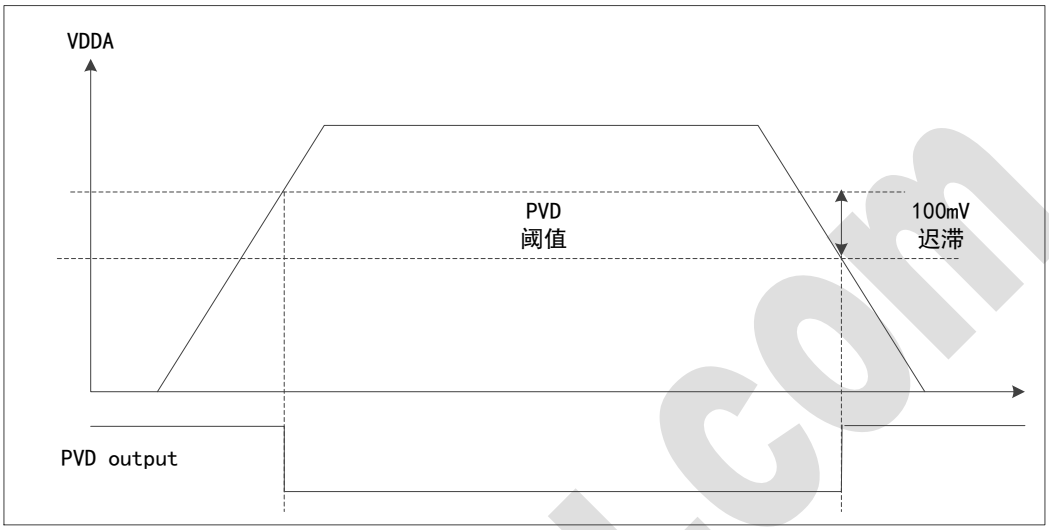
5.3.2 PVD（Programmable voltage detector）

用户可用 PVD 设置的电压阀值去监视 VDD 电源电压。

设置控制寄存器（PWR_CR)中 PLS[2:0]位，选择不同的电压阈值；通过设置 PVDE 位来使能 PVD。

电源控制/状态寄存器（PWR_CSR）中的 PVDO 标志用来表明 VDD 是高于还是低于 PVD 的电压阀值。该事件在内部连接到外部中断（EXTI）的 Line16，如果该中断被使能，该事件就会产生中断。根据 Line16 沿触发设置（上升/下降沿）就会触发对应的 PVD 中断。

PVD 波形图



5.4 低功耗模式

本产品支持低功耗模式，用户可根据最低功耗、最短启动时间和可用的唤醒事件等条件，选择最佳的低功耗模式，进而达到最佳的平衡。

本产品支持三种低功耗模式：SLEEP 模式、STOP 模式和 STANDBY 模式。

此外，运行模式（RUN mode）下，可以通过以下方式降低功耗：

- 降低系统时钟频率
- 关闭未使用外设时钟

低功耗模式

模式	进入	唤醒	对 VREG1P5 区域时钟的影响	对 VDD/VDDA 区域时钟的影响	电压调节器
睡眠（SLEEP）(SLEEP NOW or SLEEP ON EXIT)	WFI	任一中断	CPU 时钟关闭，对其它时钟及模拟时钟无影响	无	开启
	WFE	唤醒事件			
停机（STOP）	PDDS 和 LPDS 位 + SLEEPDEEP 位 + WFI 或 WFE	任一外部中断(在 EXTI 寄存器中设置) 指定通信口接收事件(USART，I2C)	所有 VREG1P5 区域时钟关闭	HSI 和 HSE 振荡器关闭	开启 可选择 Normal 或 low-power 模式(依据电源控制寄存器(PWR_CR)的设置)
待机（STANDBY）	PDDS 位+ SLEEPDEEP 位+ WFI 或 WFE	唤醒引脚上升沿，RTC 闹钟， NRST 脚外部复位， IWDG 复位			关闭

5.4.1 降低系统时钟频率

在运行模式中，系统时钟(SYSCLK，HCLK，PCLK)可通过可编程预分频寄存器降速。在进入睡眠模式前这些分频器也可用来降低外设时钟。

5.4.2 外设时钟门控

在运行模式下，可随时通过停止对外设和存储器提供时钟（HCLK 和 PCLK）来减少功耗。为了在睡眠模式下减少更多

功耗，可在执行 WFI 或 WFE 的指令前关闭外设时钟。

外设时钟门控寄存器包括：AHB 外设时钟使能寄存器（RCC_AHBENR）、APB 外设时钟使能寄存器 2(RCC_APB2ENR)、APB 外设时钟使能寄存器 1(RCC_APB1ENR)。

5.4.3 睡眠模式（SLEEP MODE）

进入睡眠模式

执行 WFI（等待中断）或 WFE（等待事件）指令可让芯片进入睡眠模式。配置 Cortex-M0 系统控制寄存器中 SLEEPONEXIT 位，有两种选项可用于选择睡眠模式的进入机制：

- ✧ SLEEP NOW：当 SLEEPONEXIT 位清 0 时，只要执行 WFI 或 WFE 指令 MCU 就立即进入睡眠模式。
- ✧ SLEEP ON EXIT：当 SLEEPONEXIT 位置 1 时，MCU 从最低优先级的中断服务程序中退出时，MCU 立即再进入睡眠模式。

退出睡眠模式

如果是执行 WFI 指令进入了睡眠模式，任意一个 NVIC 中断（interrupt）都可以唤醒 MCU，退出睡眠模式。

如果是执行 WFE 指令进入了睡眠模式，当任一事件（event）发生时，MCU 退出睡眠模式。

唤醒事件可通过以下方式产生：

- ✧ 在外设控制寄存器中使能一个中断，但不在相应 NVIC 中使能，并且在 Cortex-M0 系统控制寄存器中使能 SEVONPEND 位。当 MCU 从 WFE 中唤醒后，外设的中断挂起位和外设的 NVIC IRQ 通道挂起位（在 NVIC 中断清除挂起寄存器中）必须被清除。
- ✧ 配置一个外部或内部 EXTI 线作为事件模式。当 CPU 从 WFE 唤醒后，因为与事件 Line 对应的挂起位未被设置，不需要清除外设的中断挂起位或外设的 NVIC IRQ 通道挂起位。

因没有在中断的进入或退出上浪费时间，所以 WFE 模式唤醒所需时间最短。

SLEEP NOW 模式

SLEEP NOW 模式	说明
进入	在以下条件下执行 WFI（等待中断）或 WFE（等待事件）指令： ✧ SLEEPDEEP = 0 ✧ SLEEPONEXIT = 0 参考 Cortex-M0 系统控制寄存器。
退出	当执行 WFI 指令进入睡眠模式： 中断唤醒：参考*中断向量表* 当执行 WFE 指令进入睡眠模式： 事件唤醒：参考*事件管理*章节
唤醒延时	无

SLEEP ON EXIT 模式

SLEEP ON EXIT 模式	说明
进入	在以下条件下执行 WFI 指令： ✧ SLEEPDEEP = 0 ✧ SLEEPONEXIT = 1 参考 Cortex-M0 系统控制寄存器。
退出	中断唤醒：参考*中断向量表*
唤醒延时	无

睡眠模式中的 I/O 状态

在睡眠模式下，所有的 I/O 口状态都保持与运行模式一致。

5.4.4 停机模式（STOP MODE）

停机模式是在 Cortex-M0 的深度睡眠模式基础上结合了外设时钟控制的一种低功耗模式。在停止模式下电压调节器可运行在正常（normal）或低功耗（low power）模式。此时在 VREG1P5 供电区域的的所有时钟都被停止，PLL、HSI 和 HSE 振荡器被禁止，VREG1P5 电压域下的 SRAM 和寄存器数据被保持。

进入停止模式

关于如何进入停止模式，参见下表。

在停止模式下，通过设置电源控制寄存器（PWR_CR）的 LPDS 位使内部调节器（VREG1P5）进入低功耗模式，能够降低更多的功耗。

如果正在进行闪存编程，须等到对存储器的访问完成，系统才进入停止模式。

如果正在进行对 APB 的访问，须等到对 APB 访问完成，系统才进入停止模式。

在停机模式中 ， 可通过对独立的控制位进行编程来选择如下功能：

- 独立看门狗(IWDG)：独立看门狗可通过写看门狗密钥寄存器或硬件选项配置来启动。一旦启动了独立看门狗，看门狗会一直开启除非进行系统复位。详见*独立看门狗 (IWWDG)*章节。
- 实时时钟(RTC)：通过备份域控制寄存器 (RCC_BDCR)的 RTCEN 位来设置。
- 内部低速 RC 振荡器(LSI)：通过控制/状态寄存器(RCC_CSR)的 LSION 位来设置。
- 外部 32.768 kHz 振荡器 (LSE)：通过备份域控制寄存器(RCC_BDCR)的 LSEON 位设置。

在停机模式下，如果在进入该模式前 ADC 没有被关闭，那么仍然消耗功耗。可通过设置寄存器 ADC_CR 的 ADON 位关闭此外设。

退出停止模式

有关如何退出停机模式，详见下表。

当中断或事件唤醒 MCU 退出停机模式时，HSI 振荡器被选为系统时钟。

当电压调节器处于低功耗模式下，系统从停机模式退出时，将会有一段额外的启动延时。如果在停止模式期间保持内部调节器开启正常模式，则退出启动时间会缩短，但相应的功耗会增加。

STOP 模式进入和退出

STOP 模式	说明
进入	在以下条件下执行 WFI 或 WFE 指令： ✧ SLEEPDEEP = 1，参考 Cortex-M0 系统控制寄存器 ✧ PDDS = 0，电源控制寄存器 (PWR_CR) 通过设置 LPDS(PWR_CR)位选择电压调节器模式 *注：为了进入停机模式，所有的外部中断请求挂起位 (在挂起寄存器(EXTI_ PR)中)、所有外设中断挂起位和 RTC 闹钟标志必须清除。否则，系统会忽略 WFI 或 WFE 指令程序继续运行。* *注：如果应用程序需要在进入停止模式之前禁用外部振荡器(HSE)，系统时钟源必须首先切换到 HSI，然后清除 HSEON 位。* *注：如果在进入停止模式之前 HSEON 位保持在 1，安全系统(CSS)必须启用，检测任何外部振荡器(HSE)故障，避免进入停机模式时发生故障。*
退出	如果是执行了 WFI 批令进入了停止模式： ✧ 任一外部中断线配置为中断模式 (在 NVIC 中必须使能相应的 EXTI 中断向量). ✧ 一些特定的通信外设 (USART, I2C) 中断，设置为唤醒模式 (该外设必须配置为唤醒模式且在 NVI 中相应的中断向量必须使能)。 参见*中断向量表* 如果是执行了 WFE 指令进入了停止模式： ✧ 任一外部中断线配置为事件模式. 参考*事件管理*章节
唤醒延时	HSI 唤醒时间 + 调压器从低功耗模式唤醒的时间

停机模式中的 I/O 状态

在停机模式下，所有的 I/O 口状态都保持与运行模式一致。

注 1：USART1 和 I2C1 模块开启 stop 唤醒功能并选择 HSI8M 或 LSE，则 stop 模式 HSI 和 LSE 不会强制关闭;

注 2：STOP 模式下，LDO 可配置 normal/lowpower 模式。

5.4.5 待机模式（STANDBY MODE）

待机模式可实现系统的最低功耗。该模式是在 Cortex-M0 深度睡眠模式时关闭电压调节器 VREG1P5。整个 VREG1P5 供电区域被断电。PLL、HSI 和 HSE 振荡器也被断电。VREG1P5 电源域下的 SRAM 和寄存器内容丢失。只有备份域电路（备份寄存器、RTC、待机电路）维持供电。

注：进入停止或待机模式时，RTC、IWDG 和相应的时钟源不会停止。

进入待机模式

有关如何进入待机模式详见下表。

可以通过设置独立的控制位，选择以下待机模式的功能：

- 独立看门狗(IWDG)：独立看门狗可通过写看门狗密钥寄存器或硬件选项配置来启动。一旦启动了独立看门狗，看门狗会一直开启除非进行系统复位。详见*独立看门狗 (IWWDG)*章节。
- 实时时钟(RTC)：通过备份域控制寄存器 (RCC_BDCR)的 RTCEN 位来设置。
- 内部低速 RC 振荡器(LSI)：通过控制/状态寄存器(RCC_CSR)的 LSION 位来设置。
- 外部 32.768 kHz 振荡器 (LSE)：通过备份域控制寄存器(RCC_BDCR)的 LSEON 位设置。

退出待机模式

当有 NRST 引脚上的外部复位信号，IWDG 复位，WKUP 引脚上的上升沿或 RTC 闹钟事件发生时 MCU 退出待机模式。当退出待机模式时，除了电源控制/状态寄存器(PWR_CSR)外，所有寄存器复位。

从待机模式唤醒后的，程序执行等同于复位后的执行(采样 boot 引脚、选项字节加载、读取复位向量等)。电源控制/状态寄存器(PWR_CSR)的 SBF 状态标志指示内核由待机状态退出。

有关如何退出待机模式详见下表。

STANDBY 模式进入和退出

STOP 模式	说明
进入	在以下条件下执行 WFI 或 WFE 指令： ✧ SLEEPDEEP = 1，参考 Cortex-M0 系统控制寄存器 ✧ PDDS = 1，电源控制寄存器 (PWR_CR) ✧ WUF = 0，电源控制/状态寄存器(PWR_CSR)
退出	✧ WKUP 引脚上升沿 ✧ RTC 闹钟事件 ✧ NRST 引脚复位信号 ✧ IWDG 复位
唤醒延时	与复位相同

待机模式中的 I/O 状态

在待机模式下，除了以下引脚外，所有的 I/O 口处于高阻态。

- 复位引脚 (始终有效)
- PC13、PC14 和 PC15 被 RTC 和 LSE 模块使用
- 使能的唤醒 (WKUPx) 引脚

5.4.6 调试模式

默认情况下，当使用调试功能时，如果应用程序将 MCU 置于停机或待机模式，则调试连接将丢失。这是由于 Cortex-M0 内核已无时钟。

但是，通过设置 DBGMCU_CR 寄存器中的某些配置位，即使低功耗模式下也可以调试软件。

5.4.7 低功耗模式下的自动唤醒

RTC 闹钟可以在不需要依赖外部中断的情况下唤醒低功耗模式下的 MCU（自动唤醒模式）。RTC 提供一个可编程的时间基数，用于周期性从停止或待机模式下唤醒。通过对备份域控制寄存器（RCC_BDCR）的 RTCSEL[1:0]位的编程，RTC 时钟源中的 2 个时钟源可以选作实现此功能。

✧ **32.768kHz 低功耗外部晶振 (LSE)**

该时钟源提供了一个低功耗且精确的时间基准。(在典型情形下功耗小于 1μA)。

✧ **低功耗内部 RC 振荡器 (LSI)**

使用该时钟源，可以节省了一个 32.768kHz 晶振。但是 RC 振荡器将少许增加功耗。

使用 RTC 闹钟事件将系统从停机模式（STOP MODE）下唤醒，必须进行如下操作：

- 配置外部中断线 Line 17 为上升沿触发。
- 配置 RTC 使其可产生 RTC 闹钟事件。

如果要从待机模式（STANDBY MODE）中唤醒，无需配置外部中断线 Line 17。

5.5 相关寄存器

PWR 相关寄存器可以半字（16 BIT）或字（32 BIT）方式访问操作。

5.5.1 PWR 寄存器汇总表

基址：0x4000 7000				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**PWR_CR**	0x0000 0000	FLASH 访问控制寄存器	5.5.2
0x04	**PWR_CSR**	0x0000 0000	FLASH 状态寄存器	5.5.3

注：PWR_CR 寄存器在 STANDBY 模式唤醒后被复位；PWR_CSR 寄存器在 STANDBY 模式唤醒后不会被复位；

5.5.2 电源控制寄存器（PWR_CR）

PWR_CR			地址偏移	0x00		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	DBP
R/W	-	-	-	-	-	-	-	rw
复位值	-	-	-	-	-	-	-	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	PLS[2:0]			PVDE	CSBF	CWUF	PDDS	LPDS
R/W	rw	rw	rw	rw	rc_w1	rc_w1	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:9]	-	保留，保持复位值
BIT[8]	DBP	备份域写保护关闭 在复位后，RTC 和备份寄存器处于被保护状态以防意外写入。此位置位允许写入这些寄存器。 0：RTC 和备份寄存器写入禁止 1：RTC 和备份寄存器写入使能
BIT[7:5]	PLS[2:0]	PVD 电压阈值选择 当 PVD_LOCK（SYSCFG 配置寄存器 SYSCFG_CFGR2）被使能，则 PLS[2:0]不能再被改写。 000：2.2V 001：2.3V 010：2.4V 011：2.5V 100：2.6V 101：2.7V 110：2.8V 111：2.9V *注：详细参见数据手册电气特性章节*
BIT[4]	PVDE	PVD 使能，软件清零和置位 当 PVD_LOCK（SYSCFG 配置寄存器 SYSCFG_CFGR2）被使能，则 PVDE 不能再被改写。 0：PVD 禁止 1：PVD 使能
BIT[3]	CSBF	清零 STANDBY 标志（Clear STANDBY flag），此位读出始终为 0 0：无效 1：清除 SBF 待机标志（此位写 1）
BIT[2]	CWUF	清零唤醒标志（Clear wakeup flag），此位读出始终为 0 0：无效 1：清除 WUF 唤醒标志（此位写 1 后，2 个 sysclk 后清零 WUF）
BIT[1]	PDDS	掉电深度睡眠（power down deepsleep） 软件清零和置位，与 LPDS 协同工作 0：进入 STOP 模式@CPU Deepsleep，调压器 VREG1P5 模式由 LPDS 控制 1：进入 STANDBY 模式@CPU Deepsleep
BIT[0]	LPDS	深度睡眠下低功耗（low-power deepsleep） 软件清零和置位，与 PDDS 协同工作 0：调压器 VREG1P5 工作在正常模式 1：调压器 VREG1P5 工作在低功耗模式@STOP 模式

注：当可以在停止模式下工作的外设需要时钟时，电源控制器自动将电压调节器从低功率模式切换到正常模式，并保持在该模式，直到请求消失。

5.5.3 电源控制/状态寄存器（PWR_CSR）

PWR_CSR			地址偏移	0x04		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	EWUP2	EWUP1
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	PVDO	SBF	WUF
R/W	-	-	-	-	-	r	r	r
复位值	-	-	-	-	-	0	0	0

BIT[31:10]	-	保留，保持复位值
BIT[9:8]	EWUPx	WKUPx 引脚使能 软件清零和置位。 0：WKUPx 引脚作为通用 IO，WKUPx 引脚上的事件不能将 CPU 从 STANDBY 模式唤醒。 1:WKUPx 引脚作为唤醒脚,用于唤醒 STANDBY 模式,内部输入下拉强制打开(WKUPx 引脚上的上升沿信号唤醒 STANDBY 模式）
BIT[7:3]	-	保留，保持复位值
BIT[2]	PVDO	PVD 输出状态 此位只能硬件清零和置位，仅 PVD 使能（PVDE=1）情况下有效。 0：VDD 电压高于 PVD 设置阈值 1：VDD 电压低于 PVD 设置阈值 *注 1：STANDBY 模式下，PVD 停止工作；因此，STANDBY 唤醒或复位后，直到 PVDE 被使能，该位为 0。* *注 2：若 PVD 使能并配置，PVDO 可用于产生中断（外部中断控制器）*
BIT[1]	SBF	STANDBY 标志 当 MCU 进入 STANDBY 模式,硬件置 1;此位仅 POR/PDR 复位清零 或 CSBF 位(PWR_CR 寄存器）写 1 清零 0：设备未进入待机模式（STANDBY） 1：设备已进入待机模式
BIT[0]	WUF	唤醒标志 硬件置 1，表示设备收到唤醒事件（wakeup event）；此位系统复位清零 或 CWUF 位（PWR_CR 寄存器）写 1 清零 0：无唤醒事件发生 1：WKUPx 引脚或 RTC 闹钟产生唤醒事件 *注：当 WKUPx 引脚已经为高电平，使能 WKUPx 引脚时（设置 EWUPx 位），会检测到一个额外的唤醒事件。*

6 复位与时钟控制（RCC）

6.1 复位

本芯片支持 3 类复位：系统复位、电源复位和备份域（RTC 域）复位。

6.1.1 系统复位（System Reset）

系统复位不能复位时钟控制/状态寄存器 RCC_CSR 的复位标志和备份域的寄存器，其他寄存器都可以被系统复位。

当以下事件之一发生时，产生一个系统复位：

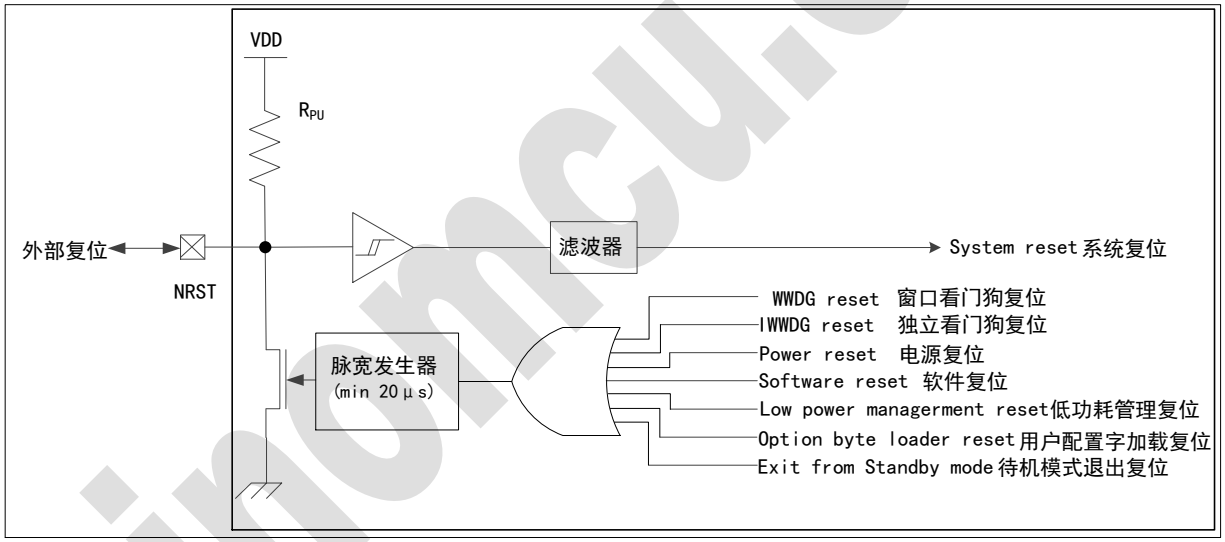
1、NRST 引脚上的低电平（外部复位）
2、窗口看门狗事件（WWDG 复位）
3、独立看门狗事件（IWWDG 复位）
4、软件复位（SW 复位）
5、低功耗管理复位
6、选项字节加载复位
7、电源复位

访问 RCC_CSR 控制/状态寄存器中的复位状态标志位可识别复位事件来源。

这些复位源经过延时扩展后作用于 NRST 引脚，并在延时时段保持低电平。复位入口向量被固定在 0x0000_0004 地址处。

内部的复位信号会在 NRST 引脚上输出，脉冲发生器保证每一个（外部或内部）复位源都能有至少 20μs 的脉冲延时；当 NRST 引脚被拉低产生外部复位时，它将产生复位脉冲。

复位电路



软件复位（software reset）

设置 SYSRESETREQ 为 1 可以产生一次软件复位，SYSRESETREQ 位在 Cortex-M0 的应用中断和复位控制寄存器（Application Interrupt and Reset Control Register）中。

详细请参见 *Cortex™-M0 technical reference manual*。

低功耗管理复位

在以下两种情况下可产生低功耗管理复位：

1、在进入待机（STANDBY）模式时产生低功耗管理复位：

将用户选项字节中的 nRST_STDBY 位置 1 将使能该复位。这时，若 MCU 进入待机模式，系统发生复位而不是进入待机模式。

2、在进入停机（STOP）模式时产生低功耗管理复位：

将用户选项字节中的 nRST_STOP 位置 1 将使能该复位。这时，若 MCU 进入停机模式，系统将被复位而不是进入停机模式。

关于用户选项字节详情，请参见 4.5 章节。

选项字节装载器复位

设置 OBL_LAUNCH 位（FLASH_CR 寄存器中）为 1，将发生选项字节装载器复位，此位用于软件重加载选项字节。

6.1.2 电源复位（POR/PDR）

当以下事件之一发生时，产生电源复位：
1、上电/掉电复位 (POR/PDR)
2、从待机模式中退出
电源复位将复位除了备份域外的所有寄存器。

6.1.3 备份域复位（RTC 域 Reset）

备份域有两个专门的复位，它们只影响备份域（RTC domain）。
当以下事件之一发生时，产生备份域复位：
1、备份域软件复位，由备份域控制寄存器 (RCC_BDCR) 的 BDRST 位触发。
2、当 VBAT 掉电低至断开连接的情况下，VDD 上电。
当以下事件之一发生时，备份寄存器发生复位：
1、发生 RTC tamper 检测事件。
2、读保护等级从 LEVEL1 修改为 LEVEL0。

6.2 时钟

本芯片包含以下时钟源:

- HSI（8M RC 振荡器时钟，由内部 HSI16M 经过 2 分频获得）
- HSI16M（内部 16M RC 振荡器时钟，用于 ADC 模块和系统时钟（2 分频））
- HSE（外部振荡器时钟）
- PLL 时钟
- LSI（40kHz 内部低速 RC 振荡器，为 IWDG 提供时钟，另可选择作为 RTC 自动唤醒时钟（从 STOP/STANDBY 模式唤醒）
- LSE（32.768kHz 外部低频晶体振荡器，用于 RTC 模块）

每个时钟源都可以独立打开/关闭，当不使用时，可以关闭以降低功耗。
其中的 HSI、HSE 和 PLL 时钟源，可作为系统时钟（SYSCLK）。
AHB 和 APB 时钟域内建独立的分频器可配置，AHB/APB 域最大时钟频率为 48MHz。
Cortex 系统定时器（SYSTICK）由 AHB 时钟驱动，其可由 AHB/8 或 AHB 时钟频率直接驱动（通过 Cortex Systick 控制/状态寄存器来配置）。
FCLK 是 Cortex-M0 的自由运行时钟，详见 *ARM Cortex™-M0 technical reference manual (TRM)*。
除以下几个外设，其他所有的外设时钟由其所在的总线时钟 (HCLK 或 PCLK) 驱动：

✧ 闪存编程接口时钟 (FLITFCLK) 固定 HSI 时钟驱动。
✧ 选项字节装载器时钟固定 HSI 时钟驱动。
✧ ADC 时钟由下列之一时钟得到 (由软件选择)：
- HSI16M 时钟，运行在最大的采样率。
- APB (PCLK) 时钟除 2 或除 4。

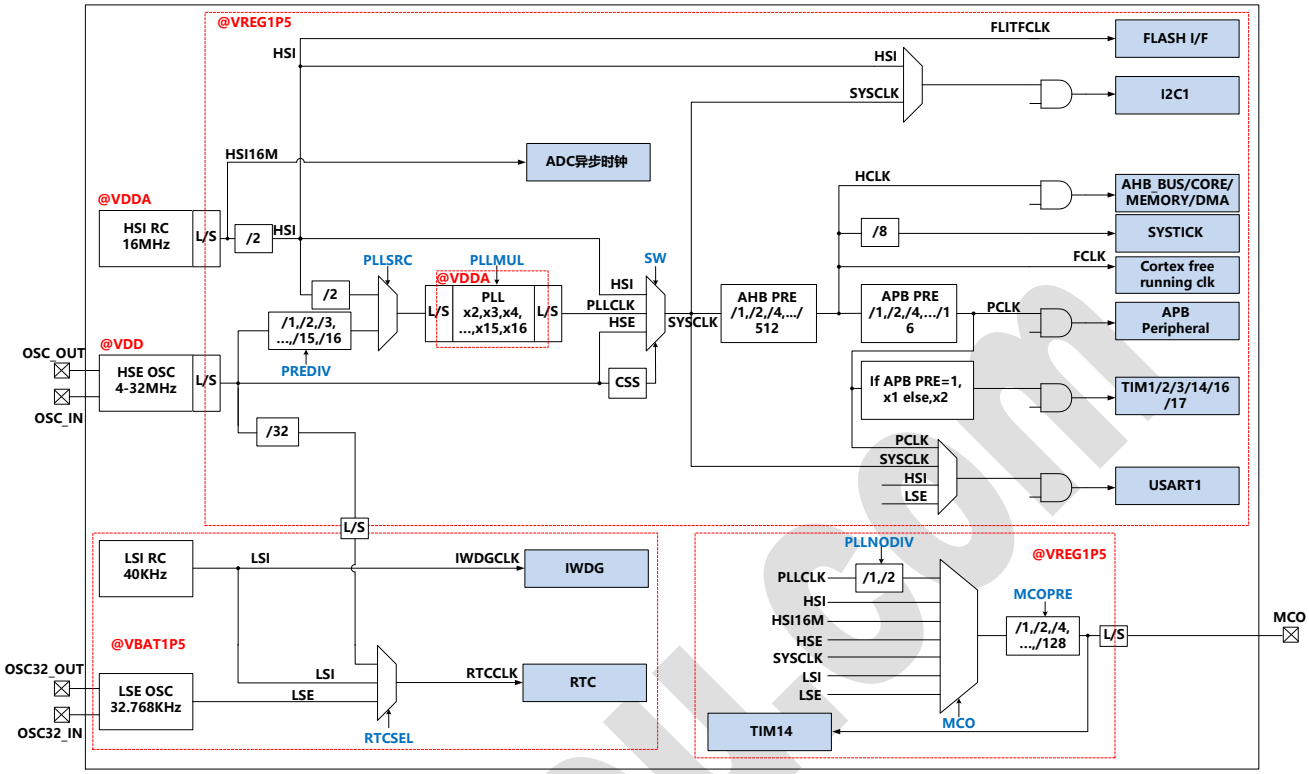
✧ USART1 的时钟为下列的时钟源之一 (由软件选择)：
- 系统时钟
- HSI 时钟
- LSE 时钟
- APB 时钟 (PCLK)

✧ I2C1 的时钟为下列的时钟源之一 (由软件选择)：
- 系统时钟
- HSI 时钟

✧ I2S1 时钟固定为系统时钟。
✧ RTC 时钟来自于 LSE、LSI 或 HSE/32。
✧ IWDG 时钟固定为 LSI 时钟。

6.2.1 时钟框图

系统时钟框图



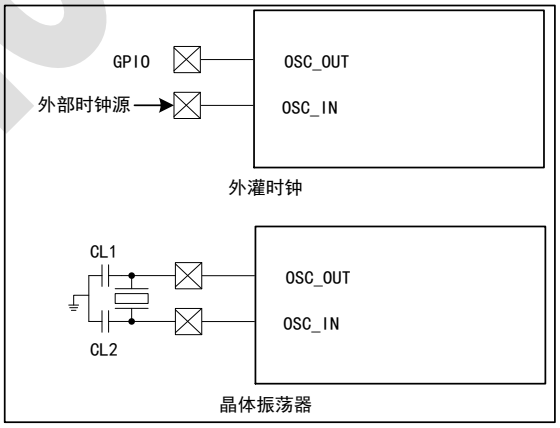
6.2.2 HSE 时钟

高速外部时钟信号（HSE）由以下两种时钟源产生：

- HSE 外部晶体谐振器
- HSE 用户外部时钟

为了减少时钟输出的失真和缩短启动稳定时间，晶体谐振器和负载电容器必须尽可能地靠近振荡器引脚。负载电容值必须匹配所选择的振荡器规定。

HSE/LSE 时钟源电路



外部晶体振荡器

内部驱动电路，支持 4~32MHz 外部晶体振荡器，可为系统时钟提供高精度时钟源。

硬件电路参见上图，详细参数信息，请参见数据手册。

在时钟控制寄存器(RCC_CR)中的 HSERDY 位用来指示高速外部振荡器是否稳定。在启动时，直到这一位被硬件置 1，该时钟才可使用。如果在时钟中断寄存器(RCC_CIR) 中允许产生中断，将会产生相应中断。

HSE 晶体可以通过设置时钟控制寄存器 (RCC_CR) 中的 HSEON 位被启动和关闭。

注：启动 HSE 过程，HSEON 使能后，内部计数器需计数 512 HSE clk。即使这样，若没有外部晶体振荡器连接至设备，

过多噪声引入到 OSC_IN 也会让振荡器开始启动，一旦启动，需要 6 个 HSE c1k 去完成关闭时序；若因为任何原因 OSC_IN 引脚振荡消失，振荡器将不能被关闭，OSC_IN 引脚被锁定不能他用并引入不必要的功耗。为了避免这种状况，强烈建议用户始终开启时钟安全系统（CSS），可以保证这种情况下也能关闭振荡器。

外灌时钟（HSE bypass）

在这个模式里，必须提供外部时钟源。它的频率最高可达 32MHz。用户可通过设置在时钟控制寄存器（RCC_CR）中的 HSEBYP 和 HSEON 位来选择这一模式。占空比为 40%～60% 外部时钟信号（方波、正弦波或三角波）必须连到 OSC_IN 引脚，此时 OSC_OUT 可作为 GPIO 使用。

6.2.3 HSI 时钟

HSI 时钟信号由内部 HSI16M RC 振荡器经过 2 分频产生，可直接作为系统时钟或在 2 分频后作为 PLL 输入。

HSI 振荡器提供低成本时钟源（不需要外部器件）。它的启动时间比 HSE 晶体振荡器短。支持出厂校准，但是，即使在校准之后它的时钟频率精度仍低于外部晶体振荡器。

校准

制造工艺决定了不同的芯片的 RC 振荡器频率有离散性，因此每颗芯片出厂都会经过精度校准（1%@25℃）。

复位后，出厂校准值会被加载至时钟控制寄存器的 HSICAL[7:0]位中（时钟控制寄存器 RCC_CR 中）。

实际应用中，不同的工作电压和环境温度，会导致 RC 振荡器频率漂移。用户可以通过修改时钟控制寄存器(RCC_CR)里的 HSITRIM[4:0]位来调整 HSI 频率。

关于 HIS 频率测量可以参见 TIM14 章节。

时钟控制寄存器（RCC_CR）中的 HSIRDY 位用来指示 HSI RC 振荡器是否稳定。在时钟启动过程中，直到这一位被硬件置 1，HSI RC 输出时钟才可以被使用。

HSI RC 可由时钟控制寄存器（RCC_CR）中的 HSION 位来启动和关闭。

如果 HSE 晶体振荡器失效，HSI 时钟会被作为备用时钟源。详细请参见时钟安全系统(CSS)章节。

此外，HSI 时钟可以通过 MCO 多路复用器输出。HSI 时钟也可以供给至 TIM14，用户进行振荡器校准和测量。

6.2.4 PLL 时钟

内部 PLL 时钟可通过 HSI 或 HSE 倍频得到，参考时钟框图。

PLL 配置（选择输入时钟，倍频因子）必须在使能 PLL 前配置好，一旦 PLL 使能，相关配置参数不能被修改。

PLL 配置流程如下：

1、设置 PLLON=0，禁用 PLL。

2、等待 PLLRDY 清 0，清 0 时 PLL 才完全停止。

3、改变 PLL 配置参数。

4、设置 PLLON=1，重新使能 PLL。

5、等待 PLLRD 置 1。

当使能时钟中断寄存器（RCC_CIR）的相应位，当 PLL 时钟就绪时，会产生一个中断。

PLL 输出频率必须设置在 16~48 MHz 范围内。

6.2.5 LSE 时钟

LSE 是一个 32.768kHz 的低速外部晶体谐振器。它为 RTC 或者其他定时功能提供一个低功耗且高精度时钟源。

LSE 可由备份域控制寄存器（RCC_BDCR）中的 LSEON 位来启动和关闭。在运行状态，设置 RCC_BDCR 中的 LSEDRV[1:0]位可修改 LSE 的驱动能力，以获得系统的鲁棒性、短启动时间及低功耗的平衡。

备份域寄存器(RCC_BDCR)中的 LSERDY 位指示 LSE 晶体振荡是否稳定。在该位被硬件置为 1 之前，LSE 的时钟信号将不会释放。如果使能时钟中断寄存器(RCC_CIR)里相应位，会产生一个中断。

注：启动 LSE 过程，LSEON 使能后，内部计数器需计数 4096 LSE c1k。即使这样，若没有外部晶体振荡器连接至设备，过多噪声引入到 OSC32_IN 也会让振荡器开始启动，一旦启动，需要 6 个 LSE c1k 去完成关闭时序；若因为任何原因 OSC32_IN 引脚振荡消失，振荡器将不能被关闭，OSC32_IN 引脚被锁定不能他用并引入不必要的功耗。恢复这种状况的唯一方式是，通过软件进行备份域（RTC 域）复位 。

外部时钟源（LSE 旁路 ）

在这个模式里，必须提供一个 32.768kHz 频率的外部时钟源。你可以通过设置在备份域控制寄存器（RCC_BDCR）里的 LSEBYP 和 LSEON 位来选择这个模式。具有 50%占空比的外部时钟信号（方波、正弦波或三角波 ）必须连到 OSC32_IN 引脚，此时 OSC32_OUT 引脚可作为 GPIO 使用，参见图表 HSE/LSE 时钟源电路。

6.2.6 LSI 时钟

LSI 为内部低功耗时钟源，可以运行在停机和待机模式下，为 IWDG 和 RTC 模块提供时钟。

LSI 时钟频率约 40kHz，具体详情请参见***数据手册***。

LSI RC 可由时钟控制/状态寄存器（RCC_CSR）中的 LSION 位来启动和关闭。

时钟控制/状态寄存器（RCC_CSR）中的 LSIRDY 位指示 LSI 振荡器是否稳定。在该位被硬件置为 1 之前，LSE 的时钟

信号将不会释放。如果使能时钟中断寄存器(RCC_CIR)里相应位，会产生一个中断。

6.2.7 系统时钟（SYSCLK）

HSI、HSE 和 PLL 时钟源，可作为系统时钟（SYSCLK）。系统复位后，默认 HSI 振荡器为系统时钟。

当时钟源被直接或通过 PLL 间接作为系统时钟时，将不能被停止。

时钟的切换只有在目标时钟源可用的情况下才能进行（即时钟经过启动延时已稳定或 PLL 已锁定）。假如系统时钟选择了未准备好的时钟源做为当前系统时钟，那么只有在目标时钟源准备好之后才真正执行切换时钟源的操作。时钟控制寄存器 (RCC_CR)中包含各个时钟的 ready（是否就绪）标志位及当前系统时钟指示位。

6.2.8 时钟安全系统（CSS）

时钟安全系统（CSS）可以通过软件被激活。一旦被激活，时钟监测器将在 HSE 振荡器启动延迟后被使能，并在 HSE 时钟关闭后关闭。

如果 HSE 时钟发生故障，HSE 振荡器被自动关闭，时钟失效事件将被送到高级定时器 (TIM1) 和通用定时器（TIM16 和 TIM17)的刹车输入(break input),并产生时钟安全中断 CSSI,以便软件进行补救处理。此 CSSI 中断连接到 Cortex-M0 的 NMI 异常中断（不可屏蔽中断）。

注意 ： 一旦 CSS 被激活，并且 HSE 时钟出现故障，CSS 中断就会发生，同时发生 NMI 中断。NMI 将被不断执行，直到 CSS 中断挂起位被清除。因此，在 NMI 的处理程序中必须设置 CSSC=1 清除 SCC 中断，CSSC 位在时钟中断寄存器 (RCC_CIR) 中。

如果 HSE 振荡器被直接或间接地作为系统时钟，(间接：HSE 被作为 PLL 输入时钟，且 PLL 时钟被作为系统时钟)，时钟故障将导致系统时钟自动切换 HSI 振荡器，同时外部 HSE 振荡器被关闭。如果 PLL 作为系统时钟且 HSE 振荡器时钟作为 PLL 的输入，发生时钟故障时，PLL 也将被关闭。

6.2.9 ADC 时钟

设置 ADC_CFG2 寄存器（ADC 配置寄存器 2），可以选择不同的 ADC 时钟。ADC 时钟包括：HSI16M RC 振荡器（异步时钟输入）和 PCLK/2(或/4)。HSI16M RC 振荡器可以软件配置成打开/关闭(auto-off mode) 模式或常开模式（打开/关闭由 ADC 接口控制）。当 APB 时钟被选为 ADC 模块时钟时， HSI16M RC 振荡器不能被 ADC 接口控制打开。

6.2.10 RTC 时钟

RTC 时钟可选择 HSE/32、LSE 或 LSI 时钟，由备份域控制寄存器(RCC_BDCR)里的 RTCSEL[1:0]位控制选择。除非备份域复位，选择后不可更改。为保证 RTC 正常运行，必须保证 PCLK 的频率大于或等于 RTCCLK 的频率。

LSE 时钟和 LSI 时钟属于备份域（RTC 域）的，但 HSE 时钟不属于备份域时钟，因此：

若 LSE 被选为 RTC 时钟：

- 只要维持 VBAT 正常供电，即使 VDD 掉电，RTC 仍会继续工作。

若 LSI 被选为 RTC 时钟：

- 只要维持 VBAT 正常供电，即使 VDD 掉电，RTC 仍会继续工作。

若 HSE/32 被选为 RTC 时钟：

- 当 VDD 掉电或内部电压调压器(VREG1P5 域的供电切断) 掉电时，RTC 处于不定的状态。

6.2.11 独立 WDT 时钟

一旦独立看门狗（IWDG）被硬件或软件启动，LSI 振荡器将强制在开启状态，且不能被关闭。LSI 振荡器稳定后，时钟供应给 IWDG。

6.2.12 时钟输出（MCO）

本芯片支持输出时钟信号到外部 MCO 引脚。对应 MCO 的 GPIO 须配置对应复用功能。

以下时钟信号之一可选为 MCO 时钟输出：

- HSI16M
- SYSCLK
- HSI
- HSE
- PLL/2 或 PLL
- LSE
- LSI

MCO 时钟的选择由时钟配置寄存器(RCC_CFGR)的 MCO[3:0]位决定。

时钟配置寄存器(RCC_CFGR)的 PLLNODIV 位，控制 PLL 时钟输入到 MCO 的二分频旁路开启/关闭。的

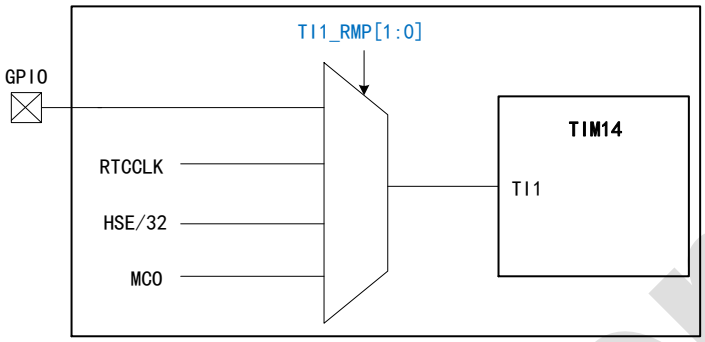
时钟配置寄存器(RCC_CFGR)的 MCOPRE[2:0]位，控制输入到 MCO 的时钟经过二进制除法器后输出，可配置

/1/2/4…/128 除频因子。

6.2.13 内外时钟测量（TIM14）

通过 TIM14 通道 1 输入捕获，可以间接测量所有板载时钟源的频率。如下图所示。

TIM14 测频电路（捕获模式）



通过配置 TIM14_OR 寄存器中的 TI1_RMP[1:0]位，可以选择不同的时钟源（GPIO 或内部时钟）接入 TIM14 的输入捕获通道。可选时钟如下：

- GPIO
- RTCCLK
- HSE/32
- MCO（参见 MCO 章节）

HSI 校准

使用 TIM14 捕获模式校准 HSI，首先通过 MCO 的多路复用器选择 LSE 时钟，并连接 MCO 至 TIM14 的输入捕获通道，设置 HSI 作为系统时钟源。对 LSE 边沿捕获（HSI 时钟计数）进而测量 HSI 频率，利用 LSE 晶体的高精度（通常几十 ppm）可以确定内部时钟频率，保证相同的精度。

基本原理就是提供一种相对测量方法（例如，HSI/LSE 比率），比率越高测量效果越好。

如果没有 LSE，为了达到更精确的校准，HSE/32 将是更好的选择。

LSI 校准

LSI 校准与 HSI 方式相近，但需要调整参考时钟。将 LSI 时钟连接至 TIM14 的输入捕获通道，设置 HSE 作为系统时钟源。对 LSI 边沿捕获（HSE 时钟计数）进而测量 LSI 频率。

基本原理就是提供一种相对测量方法（例如，HSE/LSI 比率），比率越高测量效果越好。

6.3 低功耗模式

APB 外设时钟和 DMA 时钟可以软件关闭。

睡眠模式（SLEEP mode），CPU 时钟停止，存储器接口时钟（Flash 和 RAM 接口）可以软件停止。当连接到 APB 所有外设的时钟关闭后，进入睡眠模式期间 AHB 到 APB 桥时钟被硬件关闭。

停机模式（STOP mode），VREG1P5 电压域所有时钟关闭，PLL、HSI16M（HSI)和 HSE 振荡器的时钟被关闭。

若 HSI 被选为 USART1 和 I2C1 时钟，即使在 MCU 进入停止模式，USART1 和 I2C1 仍可以打开 HSI 振荡器。当系统处于 STOP 模式且电压调节器 VREG1P5 工作在 LP 模式(低功耗模式)，来自这两个外设中任一时钟请求，电压调节器 VREG1P5 将切换至正常模式（MR mode），以便为核心逻辑提供适当的驱动电流。一旦请求被撤销，调节器回到 LP 模式，且不会唤醒 MCU。

LSE 开启后可以被选为 USART1 和 I2C1 时钟在 STOP 模式下工作，但 USART1 和 I2C1 没有打开 LSE 的能力。

待机模式（STANDBY mode），VREG1P5 电压域所有时钟关闭，PLL、HSI16M（HSI)和 HSE 振荡器的时钟被关闭。

设置 DBGMCU_CR 寄存器中的 DBG_STOP 或 DBG_STANDBY 位，CPU 的深度睡眠模式被覆盖并具有调试功能。

当中断唤醒 STOP 模式或 STANDBY 唤醒复位，HSI 振荡器默认为系统时钟。

如果正在进行闪存编程，须等到对存储器的访问完成，系统才进入深度睡眠模式（深度睡眠延后)。如果正在进行对 APB 的访问，须等到对 APB 访问完成，系统才进入深度睡眠模式。

6.4 相关寄存器

6.4.1 RCC 寄存器汇总表

基址：0x4002 1000				
偏移地址	寄存器名	复位值	功能描述	章节

0x00	RCC_CR	0x0000 XX83	时钟控制寄存器	6.4.2
0x04	RCC_CFGR	0x0000 0000	时钟配置寄存器	6.4.3
0x08	RCC_CIR	0x0000 0000	时钟中断寄存器	6.4.4
0x0C	RCC_APB2RSTR	0x0000 0000	APB 外设复位寄存器 2	6.4.5
0x10	RCC_APB1RSTR	0x0000 0000	APB 外设复位寄存器 1	6.4.6
0x14	RCC_AHBENR	0x0000 0014	AHB 外设时钟使能寄存器	6.4.7
0x18	RCC_APB2ENR	0x0000 0000	APB 外设时钟使能寄存器 2	6.4.8
0x1C	RCC_APB1ENR	0x0000 0000	APB 外设时钟使能寄存器 1	6.4.9
0x20	RCC_BDCR	0x0000 0018	备份域控制寄存器	6.4.10
0x24	RCC_CSR	0xXXX0 0000	RCC 控制/状态寄存器	6.4.11
0x28	RCC_AHBRSTR	0x0000 0000	AHB 外设复位寄存器	6.4.12
0x2C	RCC_CFGR2	0x0000 0000	时钟配置寄存器 2	6.4.13
0x30	RCC_CFGR3	0x0000 0000	时钟配置寄存器 3	6.4.14

6.4.2 时钟控制寄存器（RCC_CR）

访问：无等待，字、半字和字节访问

RCC_CR			地址偏移	0x00		复位值	0x0000 XX83	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	PLLRDY	PLLON
R/W	-	-	-	-	-	-	r	rw
复位值	-	-	-	-	-	-	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	CSSON	HSEBYP	HSERDY	HSEON
R/W	-	-	-	-	rw	rw	r	rw
复位值	-	-	-	-	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	HSICAL[7:0]							
R/W	r	r	r	r	r	r	r	r
复位值	X	X	X	X	X	X	X	X
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	HSIRDY	HSION
R/W	-	-	-	-	-	-	r	rw
复位值	1	0	0	0	0	-	1	1

BIT[31:26]	-	保留，保持复位值
BIT[25]	PLLRDY	PLL 时钟就绪标志，硬件置 1，表明 PLL 被锁定 0：PLL 未锁定 1：PLL 已锁定
BIT[24]	PLLON	PLL 使能，软件置位和清零 当进入停机或待机模式时由硬件清零。当 PLL 时钟正做为系统时钟或被选用将做为系统时钟时，该位不能被清零。 0：PLL 关闭 1：PLL 使能
BIT[23:20]	-	保留，保持复位值
BIT[19]	CSSON	时钟安全系统使能，软件置位和清零 当 CSSON=1，HSE 振荡器就绪时，时钟检测器由硬件打开，当检测到 HSE 时钟错误时清零。 0：CSS 关闭（时钟检测器关闭） 1：CSS 使能（HSE 就绪后，钟检测器打开，否则关闭）。
BIT[18]	HSEBYP	HSE 晶体振荡器旁路，软件置位和清零 使用外灌时钟时，旁路振荡器，HSEON 位须使能；此 HSEBYP 位只能在 HSEON 关闭时写入。 0：HSE 晶体振荡器不被旁路 1：HSE 晶体振荡器被旁路，外灌时钟
BIT[17]	HSERDY	HSE 时钟就绪标志，硬件置 1，表明 HSE 时钟是否稳定 当 HSEON 清零后，该位需要 6 个 HSE clk 才会清零。 0：HSE 未就绪

		1：HSE 已就绪
BIT[16]	HSEON	HSE 使能，软件置位和清零 当进入停机或待机模式时由硬件清零。当 HSE 时钟直接或间接被使用时，该位不能被清零。 0：HSE 关闭 1：HSE 使能
BIT[15:8]	HSICAL[7:0]	HSI 时钟校准 在启动时，这些位被自动初始化为出厂校准值。
BIT[7:2]	-	保留，保持复位值
BIT[1]	HSIRDY	HSI 时钟就绪标志，硬件置 1，表明 HSI 时钟是否稳定 当 HSION 清零后，该位需要 6 个 HSI clk 才会清零。 0：HSI 未就绪 1：HSI 已就绪
BIT[0]	HSION	HSI 使能，软件置位和清零 当从停机或待机模式退出或 HSE 直接/间接作为系统时钟并发生故障，该位由硬件置位启动 HSI 振荡器。当 HSI 时钟直接或间接被使用时，该位不能被清零。 0：HSI 关闭 1：HSI 使能

6.4.3 时钟配置寄存器（RCC_CFGR）

访问：0~2 个等待周期，字、半字和字节访问；其中，只有访问发生在时钟切换时插入 1~2 个等待周期。

RCC_CFGR			地址偏移	0x04		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	PLLNODIV	MCOPRE[2:0]			MCO[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	PLLMUL[3:0]				PLLXTPRE	PLLSRC[1]
R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	PPRE[2:0]		
R/W	-	-	-	-	-	rw	rw	rw
复位值	-	-	-	-	-	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	HPRE[3:0]				SWS[1:0]		SW[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31]	PLLNODIV	PLL 时钟不分频控制，软件置位和清零 0：PLL/2 连接至 MCO 1：PLL 连接至 MCO
BIT[30:28]	MCOPRE[2:0]	MCO 输出预分频，软件置位和清零 为避免产生毛刺，强烈建议在关闭 MCO 时修改此预分频器。 000：MCO/1 001：MCO/2 010：MCO/4 … 111：MCO/128
BIT[27:24]	MCO[3:0]	MCO 输出时钟源选择，软件置位和清零 0000：MCO 输出关闭，MCO 引脚无时钟输出 0001：选择 HSI16M（16MHz） 0010：选择 LSI 0011：选择 LSE 0100：选择系统时钟（SYSCLK） 0101：选择 HSI（8MHz） 0110：选择 HSE

		0011：选择 PLL/2 或 PLL（由 PLLNODIV 控制） 其他：保留
BIT[23:22]	-	保留，保持复位值
BIT[21:18]	PLLMUL[3:0]	PLL 倍频系数，软件写入 *注：只有 PLL 关闭时才可被写入，PLL 输出最大频率不能超过 48MHz* 0000：PLL 输入时钟 x2 0001：x3 0010：x4 … 1110 和 1111：x16
BIT[17]	PLLXTPRE	HSE 作为 PLL 输入分频控制，软件置位和清零 此位与时钟配置寄存器 2(RCC_CFGR2)中的 PREDIV[0]位是相同位，请参见 PREDIV[0]描述。 *注：只有 PLL 关闭时才可被写入*
BIT[16]	PLLSRC	PLL 输入时钟源，软件置位和清零 注：只有 PLL 关闭时才可被写入 0：HSI/2 作为 PLL 输入时钟 1：HSE/PREDIV 作为 PLL 输入时钟
BIT[15:11]	-	保留，保持复位值
BIT[10:8]	PPRE[2:0]	PCLK 预分频，软件置位和清零 0xx：HCLK 不分频 100：HCLK/2 101：HCLK/4 110：HCLK/8 111：HCLK/16
BIT[7:4]	HPRE[3:0]	HCLK 预分频，软件置位和清零 0xxx：SYSCLK 不分频 1000：SYSCLK/2 1001：SYSCLK/4 1010：SYSCLK/8 1011：SYSCLK/16 1100：SYSCLK/64 1101：SYSCLK/128 1110：SYSCLK/256 1111：SYSCLK/512 *注：当 AHB 时钟的预分频系数大于 1 时，必须开启预取缓冲器。详见读操作章节。*
BIT[3:2]	SWS[1:0]	系统时钟切换状态位，硬件置位或清零来指示系统时钟源。 00：HSI 作为系统时钟 01：HSE 作为系统时钟 10：PLL 作为系统时钟 11：保留
BIT[1:0]	SW[1:0]	系统时钟切换，软件置位和清零 当从停机或待机模式退出或 HSE 直接/间接作为系统时钟并发生故障，该位由硬件置位启动 HSI 振荡器并作为系统时钟（CSS 开启）。 00：HSI 作为系统时钟 01：HSE 作为系统时钟 10：PLL 作为系统时钟 11：保留

6.4.4 时钟中断寄存器（RCC_CIR）

访问：无等待周期，字、半字和字节访问

RCC_CIR			地址偏移	0x08		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16

控制位	CSSC	-	-	PLLRDYC	HSERDYC	HSIRDYC	LSERDYC	LSIRDYC
R/W	w	-	-	w	w	w	w	w
复位值	0	-	-	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	PLLRDYIE	HSERDYIE	HSIRDYIE	LSERDYIE	LSIRDYIE
R/W	-	-	-	rw	rw	rw	rw	rw
复位值	-	-	-	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CSSF	-	-	PLLRDYF	HSERDYF	HSIRDYF	LSERDYF	LSIRDYF
R/W	r	-	-	r	r	r	r	r
复位值	0	-	-	0	0	0	0	0

BIT[31:24]	-	保留，保持复位值
BIT[23]	CSSC	CSS 中断清除，软件置位 0：无效 1：清 CSSF 标志
BIT[22:21]	-	保留，保持复位值
BIT[20]	PLLRDYC	PLLRDY 中断清除，软件置位 0：无效 1：清 PLLRDYF 标志
BIT[19]	HSERDYC	HSERDY 中断清除，软件置位 0：无效 1：清 HSERDYF 标志
BIT[18]	HSIRDYC	HSIRDY 中断清除，软件置位 0：无效 1：清 HSIRDYF 标志
BIT[17]	LSERDYC	LSERDY 中断清除，软件置位 0：无效 1：清 LSERDYF 标志
BIT[16]	LSIRDYC	LSIRDY 中断清除，软件置位 0：无效 1：清 LSIRDYF 标志
BIT[15:13]	-	保留，保持复位值
BIT[12]	PLLRDYIE	PLLRDY 中断使能，软件置位和清零 0：中断关闭 1：中断使能
BIT[11]	HSERDYIE	HSERDY 中断使能，软件置位和清零 0：中断关闭 1：中断使能
BIT[10]	HSIRDYIE	HSIRDY 中断使能，软件置位和清零 0：中断关闭 1：中断使能
BIT[9]	LSERDYIE	LSERDY 中断使能，软件置位和清零 0：中断关闭 1：中断使能
BIT[8]	LSIRDYIE	LSIRDY 中断使能，软件置位和清零 0：中断关闭 1：中断使能
BIT[7]	CSSF	CSS 中断标志，软件对 CSSC 写 1 清零该位 当系统检测到 HSE 振荡器错误时由硬件置位该位 0：未发生 CSS 中断 1：发生 CSS 中断
BIT[6:5]	-	保留，保持复位值
BIT[4]	PLLRDYF	PLLRDY 中断标志，软件对 PLLRDYC 写 1 清零该位 0：未发生中断 1：发生中断
BIT[3]	HSERDYF	HSERDY 中断标志，软件对 HSERDYC 写 1 清零该位 0：未发生中断

		1：发生中断
BIT[2]	HSIRDYF	HSIRDY 中断标志，软件对 HSIRDYC 写 1 清零该位 当 HSI 振荡器时钟稳定，HSIRDYDIE=1 且 HSION =1 (RCC_CFGR2)，硬件置位该位；当 HSION=0，但 HSI 被其他外设触发启动，此位不会置位不会产生中断 0：未发生中断 1：发生中断
BIT[1]	LSERDYF	LSERDY 中断标志，软件对 LSERDYC 写 1 清零该位 0：未发生中断 1：发生中断
BIT[0]	LSIRDYF	LSIRDY 中断标志，软件对 LSIRDYC 写 1 清零该位 0：未发生中断 1：发生中断

6.4.5 APB 外设复位寄存器 2（RCC_APB2RSTR）

RCC_APB2RSTR			地址偏移	0x0C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	DBGMCU RST	-	-	-	TIM17RST	TIM16RST	-
R/W	-	rw	-	-	-	rw	rw	-
复位值	-	0	-	-	-	0	0	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	USART1RST	-	SPI1RST	TIM1RST	-	ADCRST	-
R/W	-	rw	-	rw	rw	-	rw	-
复位值	-	0	-	0	0	-	0	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	-	SYSCFGRST
R/W	-	-	-	-	-	-	-	rw
复位值	-	-	-	-	-	-	-	0

BIT[31:23]	-	保留，保持复位值
BIT[22]	DBGMCU_RST	调试 MCU 复位，软件置位和清零 0：无效 1：复位 debug MCU
BIT[21:19]	-	保留，保持复位值
BIT[18]	TIM17RST	TIM17 复位，软件置位和清零 0：无效 1：复位 TIM17
BIT[17]	TIM16RST	TIM16 复位，软件置位和清零 0：无效 1：复位 TIM16
BIT[16:15]	-	保留，保持复位值
BIT[14]	USART1RST	USART1 复位，软件置位和清零 0：无效 1：复位 USART1
BIT[13]	-	保留，保持复位值
BIT[12]	SPI1RST	SPI1 复位，软件置位和清零 0：无效 1：复位 SPI1
BIT[11]	TIM1RST	TIM1 复位，软件置位和清零 0：无效 1：复位 TIM1
BIT[10]	-	保留，保持复位值

BIT[9]	ADCRST	ADC 接口复位，软件置位和清零 0：无效 1：复位 ADC
BIT[8:1]	-	保留，保持复位值
BIT[0]	SYSCFGRST	SYSCFG 复位，软件置位和清零 0：无效 1：复位 SYSCFG

6.4.6 APB 外设复位寄存器 1（RCC_APB1RSTR）

RCC_APB1RSTR			地址偏移	0x10		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	PWRRST	-	-	-	-
R/W	-	-	-	rw	-	-	-	-
复位值	-	-	-	0	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	I2C1RST	-	-	-	-	-
R/W	-	-	rw	-	-	-	-	-
复位值	-	-	0	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	WWDGRST	-	-	TIM14RST
R/W	-	-	-	-	rw	-	-	rw
复位值	-	-	-	-	0	-	-	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	TIM3RST	TIM2RST
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	0	0

BIT[31:29]	-	保留，保持复位值
BIT[28]	PWRRST	PWR 接口复位，软件置位和清零 0：无效 1：复位 PWR
BIT[27:22]	-	保留，保持复位值
BIT[21]	I2C1RST	I2C1 复位，软件置位和清零 0：无效 1：复位 I2C1
BIT[20:12]	-	保留，保持复位值
BIT[11]	WWDGRST	窗口看门狗复位，软件置位和清零 0：无效 1：复位 WWDG
BIT[10:9]	-	保留，保持复位值
BIT[8]	TIM14RST	TIM14 复位，软件置位和清零 0：无效 1：复位 TIM14
BIT[7:2]	-	保留，保持复位值
BIT[1]	TIM3RST	TIM3 复位，软件置位和清零 0：无效 1：复位 TIM3
BIT[0]	TIM2RST	TIM2 复位，软件置位和清零 0：无效 1：复位 TIM2

6.4.7 AHB 外设时钟使能寄存器（RCC_AHBENR）

访问：无等待周期，字、半字和字节访问

RCC_AHBENR			地址偏移	0x14		复位值	0x0000 0014	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-

复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	IOPFEN	-	-	IOPCEN	IOPBEN	IOPAEN	-
R/W	-	rw	-	-	rw	rw	rw	-
复位值	-	0	-	-	0	0	0	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	CRCEN	-	FLITFEN	-	SRAMEN	-	DMAEN
R/W	-	rw	-	rw	-	rw	-	rw
复位值	-	0	-	1	-	1	-	0

BIT[31:23]	-	保留，保持复位值
BIT[22]	IOPFEN	GPIOF 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[21:20]	-	保留，保持复位值
BIT[19]	IOPCEN	GPIOC 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[18]	IOPBEN	GPIOB 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[17]	IOPAEN	GPIOA 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[16:7]	-	保留，保持复位值
BIT[6]	CRCEN	CRC 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[5]	-	保留，保持复位值
BIT[4]	FLITFEN	FLITF 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[3]	-	保留，保持复位值
BIT[2]	SRAMEN	SRAM 接口时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[1]	-	保留，保持复位值
BIT[0]	DMAEN	DMA 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启

6.4.8 APB 外设时钟使能寄存器 2（RCC_APB2ENR）

访问：字、半字和字节访问。无等待周期，除非访问发生时，APB 访问进行中，这种情况下必须插入等待直至上次的 APB 外设访问完成。

RCC_APB2ENR			地址偏移	0x18		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	DBGMCUEN	-	-	-	TIM17EN	TIM16EN	-
R/W	-	rw	-	-	-	rw	rw	-
复位值	-	0	-	-	-	0	0	-

	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	USART1EN	-	SPI1EN	TIM1EN	-	ADCEN	-
R/W	-	rw	-	rw	rw	-	rw	-
复位值	-	0	-	0	0	-	0	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	-	SYSCFGEN
R/W	-	-	-	-	-	-	-	rw
复位值	-	-	-	-	-	-	-	0

BIT[31:23]	-	保留，保持复位值
BIT[22]	DBGMCUEN	MCU 调试模块时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[21:19]	-	保留，保持复位值
BIT[18]	TIM17EN	TIM17 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[17]	TIM16EN	TIM16 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[16:15]	-	保留，保持复位值
BIT[14]	USART1EN	USART1 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[13]	-	保留，保持复位值
BIT[12]	SPI1EN	SPI1 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[11]	TIM1EN	TIM1 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[10]	-	保留，保持复位值
BIT[9]	ADCEN	ADC 接口时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[8:1]	-	保留，保持复位值
BIT[0]	SYSCFGEN	SYSCFG 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启

6.4.9 APB 外设时钟使能寄存器 1（RCC_APB1ENR）

访问：字、半字和字节访问。无等待周期，除非访问发生时，APB 访问进行中，这种情况下必须插入等待直至上次的 APB 外设访问完成。

RCC_APB1ENR			地址偏移	0x1C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	PWREN	-	-	-	-
R/W	-	-	-	rw	-	-	-	-
复位值	-	-	-	0	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	I2C1EN	-	-	-	-	-
R/W	-	-	rw	-	-	-	-	-
复位值	-	-	0	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	WWDGEN	-	-	TIM14EN
R/W	-	-	-	-	rw	-	-	rw
复位值	-	-	-	-	0	-	-	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

控制位	-	-	-	-	-	-	TIM3EN	TIM2EN
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	0	0

BIT[31:29]	-	保留，保持复位值
BIT[28]	PWREN	PWR 接口时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[27:22]	-	保留，保持复位值
BIT[21]	I2C1EN	I2C1 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[20:12]	-	保留，保持复位值
BIT[11]	WWDGEN	窗口看门狗时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[10:9]	-	保留，保持复位值
BIT[8]	TIM14EN	TIM14 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[7:2]	-	保留，保持复位值
BIT[1]	TIM3EN	TIM3 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启
BIT[0]	TIM2EN	TIM2 时钟使能，软件置位和清零 0：时钟关闭 1：时钟开启

6.4.10 备份域控制寄存器（RCC_BDCR）

访问：字、半字和字节访问。0~3 个等待周期，当连续对此寄存器进行访问，将插入等待周期。

注：备份域控制寄存器(RCC_BDCR) 的 LSEON, LSEBYP, RTCSEL 和 RTCEN 位处于备份域，这些位复位后处于写保护状态。只有在电源控制寄存器 (PWR_CR) 中的 DBP 位置为 1 后才能对这些位进行改动。这些位仅在备份域复位后才清 0，任何内部或外部复位都不会影响这些位。(参见 6.1.3 节的备份域复位 和 19 章节 RTC)

RCC_BDCR			地址偏移	0x20		复位值	0x0000 0018	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	BDRST
R/W	-	-	-	-	-	-	-	rw
复位值	-	-	-	-	-	-	-	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	RTCEN	-	-	-		-	RTCSEL[1:0]	
R/W	rw	-	-	-		-	rw	rw
复位值	0	-	-	-		-	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	LSEDRV[1:0]		LSE BYP	LSE RDY	LSEON
R/W	-	-	-	rw	rw	rw	rw	rw
复位值	-	-	-	1	1	0	0	0

BIT[31:17]	-	保留，保持复位值
BIT[16]	BDRST	备份域软件复位，软件置位和清零 0：复位未激活 1：复位整个备份域（RTC 域）
BIT[15]	RTCEN	RTC 时钟使能，软件置位和清零 0：时钟关闭

		1：时钟开启
BIT[14:10]	-	保留，保持复位值
BIT[9:8]	RTCSEL[1:0]	RTC 时钟源选择，软件置位和清零 一旦 RTC 时钟源被选定，这些位的值不能被改变， 除非备份域被复位。可通过设置 BDRST 位来复位备份域。 00：无时钟 01：LSE 作为 RTC 时钟 10：LSI 作为 RTC 时钟 11：HSE/32 作为 RTC 时钟
BIT[7:5]	-	保留，保持复位值
BIT[4:3]	LSEDRV[1:0]	LSE 振荡器驱动能力，软件置位和清零 当复位备份域时，会重装载默认值。 00：弱驱动能力 01：中低驱动能力 10：中高驱动能力 11：强驱动能力（复位默认值） *注：振荡器在晶体模式（XTAL mode）非旁路模式（bypass mode）*
BIT[2]	LSEBYP	LSE 振荡器旁路，软件置位和清零 该位仅在外部 32KHz 振荡器关闭的情况下写入。 0：LSE 振荡器未被旁路 1：LSE 振荡器被旁路
BIT[1]	LSERDY	LSE 振荡器就绪，硬件置位和清零，指示外部 32kHz 振荡器是否稳定。 当 LSEON 清零后，该位需要 6 个 LSE clk 才会清零。 0：LSE 未就绪 1：LSE 已就绪
BIT[0]	LSEON	LSE 使能，软件置位和清零 0：LSE 关闭 1：LSE 使能

6.4.11 RCC 控制/状态寄存器（RCC_CSR）

访问：字、半字和字节访问。0~3 个等待周期，当连续对此寄存器访问，将插入等待周期。

RCC_CSR			地址偏移	0x24		复位值	0xXXX0 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	LPWRRSTF	WWDGRSTF	IWDGRSTF	SFTRSTF	PORRSTF	PINRSTF	OBLRSTF	RMVF
R/W	r	r	r	r	r	r	r	rt_w
复位值	X	X	X	X	X	X	X	X
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	V15PWRRSTF	-	-	-	-	-	-	-
R/W	r	-	-	-	-	-	-	-
复位值	X	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	LSIRDY	LSION
R/W	-	-	-	-	-	-	r	rw
复位值	-	-	-	-	-	-	0	0

BIT[31]	LPWRRSTF	低功耗复位标志 低功耗管理复位发生时由硬件置 1，软件通过写 RMVF 位清除该位。 0：无低功耗管理复位发生 1：发生低功耗管理复位
BIT[30]	WWDGRSTF	窗口看门狗复位标志 窗口看门狗复位发生时由硬件置 1，软件通过写 RMVF 位清除该位。 0：无窗口看门狗复位发生

		1：发生窗口看门狗复位
BIT[29]	IWDGRSTF	独立看门狗复位标志 独立看门狗复位发生时由硬件置 1，软件通过写 RMVF 位清除该位。 0：无独立看门狗复位发生 1：发生独立看门狗复位
BIT[28]	SFTRSTF	软件复位标志 软件复位发生时由硬件置 1，软件通过写 RMVF 位清除该位。 0：无软件复位发生 1：发生软件复位
BIT[27]	PORRSTF	上电/掉电复位标志 上电/掉电复位发生时由硬件置 1，软件通过写 RMVF 位清除该位。 0：无上电/掉电复位发生 1：发生上电/掉电复位
BIT[26]	PINRSTF	NRST 引脚复位标志 NRST 引脚复位发生时由硬件置 1，软件通过写 RMVF 位清除该位。 0：无 NRST 引脚复位发生 1：发生 NRST 引脚复位
BIT[25]	OBLRSTF	选项字节加载复位标志 选项字节加载复位发生时由硬件置 1，软件通过写 RMVF 位清除该位。 0：无选项字节加载复位发生 1：发生选项字节加载复位
BIT[24]	RMVF	清除复位标志 软件写 1 来清除所有复位标志，包括 RMVF。 0：无效 1：清除复位标志
BIT[23]	V15PWRRSTF	VREG1P5 电源域复位标志 VREG1P5 域上电/掉电复位发生时由硬件置 1，软件通过写 RMVF 位清除该位。 0：无 VREG1P5 域上电/掉电复位发生 1：发生 VREG1P5 域上电/掉电复位
BIT[22:2]	-	保留，保持复位值
BIT[1]	LSIRDY	LSI 振荡器就绪，硬件置位和清零，指示 LSI 振荡器是否稳定。 当 LSION 清零后，该位需要 3 个 LSI clk 才会清零。 0：LSI 未就绪 1：LSI 已就绪
BIT[0]	LSION	LSI 使能，软件置位和清零 0：LSI 关闭 1：LSI 使能

6.4.12 AHB 外设复位寄存器（RCC_AHBRSTR）

访问：字、半字和字节访问。无等待周期。

RCC_AHBRSTR			地址偏移	0x28		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	IOPFRST	-	-	IOPCRST	IOPBRST	IOPARST	-
R/W	-	rw	-	-	rw	rw	rw	-
复位值	-	0	-	-	0	0	0	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-

BIT[31:23]	-	保留，保持复位值
BIT[22]	IOPFRST	GPIOF 复位，软件置位和清零 0：无效 1：复位 GPIOF
BIT[21:20]	-	保留，保持复位值
BIT[19]	IOPCRST	GPIOC 复位，软件置位和清零 0：无效 1：复位 GPIOC
BIT[18]	IOPBRST	GPIOB 复位，软件置位和清零 0：无效 1：复位 GPIOB
BIT[17]	IOPARST	GPIOA 复位，软件置位和清零 0：无效 1：复位 GPIOA
BIT[16:0]	-	保留，保持复位值

6.4.13 时钟配置寄存器 2（RCC_CFGR2）

访问：字、半字和字节访问。无等待周期。

RCC_CFGR			地址偏移	0x2C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	PREDIV[3:0]			
R/W	-	-	-	-	rw	rw	rw	rw
复位值	-	-	-	-	0	0	0	0

BIT[31:4]	-	保留，保持复位值
BIT[3:0]	PREDIV[3:0]	HSE 分频因子，这些位仅能在 PLL 关闭时改写 *注：bit 0 与 RCC_CFGR 的 bit 17 相同，修改 RCC_CFGR 的 bit 17 同时也会改变这里的 bit 0。* 0000：HSE 作为 PLL 的输入，不分频 0001：HSE/2 作为 PLL 输入 0010：HSE/3 作为 PLL 输入 … 1111：HSE/16 作为 PLL 输入

6.4.14 时钟配置寄存器 3（RCC_CFGR3）

访问：字、半字和字节访问。无等待周期。

RCC_CFGR3			地址偏移	0x30		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-

复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	I2C1SW	-	-	USART1SW[1:0]	
R/W	-	-	-	rw	-	-	rw	rw
复位值	-	-	-	0	-	-	0	0

BIT[31:5]	-	保留，保持复位值
BIT[4]	I2C1SW	I2C1 时钟源选择，软件置位和清零 0：HSI 时钟被选为 I2C1 时钟源（默认） 1：系统时钟(SYSCLK)被选为 I2C1 的时钟源
BIT[3:2]	-	保留，保持复位值
BIT[1:0]	USART1SW[1:0]	USART1 时钟源选择，软件置位和清零 00：PCLK 时钟被选为 USART1 时钟源（默认） 01：系统时钟(SYSCLK)被选为 USART1 的时钟源 10：LSE 时钟被选为 USART1 时钟源 11：HSI 时钟被选为 USART1 时钟源

7 通用I/O（GPIO）

7.1 概述

每组 GPIO 都 有 4 个 32 位配置寄存器(GPIOx_MODER, GPIOx_OTYPER, GPIOx_ OSPEEDR 和 GPIOx_PUPDR)，2 个 16 位数据寄存器(GPIOx_IDR 和 GPIOx_ODR)和 1 个 32 位置位/复位寄存器(GPIOx_BSRR)。GPIOA 和 GPIOB 还含有 1 个 32 位锁定寄存器(GPIOx_ LCKR)和 2 个 32 位复用功能寄存器 (GPIOx_AFRH 和 GPIOx_AFRL)。

7.2 特性

- ✧ 输出：支持推挽输出和开漏输出（带上拉/下拉控制）
- ✧ 输入：支持浮空、上拉/下拉、模拟输入
- ✧ 每个 IO 速度可选
- ✧ 支持位操作（置位/复位寄存器 GPIOx_BSRR）
- ✧ GPIOA 和 GPIOB 支持锁定配置
- ✧ 支持复用功能选择和模拟功能
- ✧ 支持 GPIO 快速翻转（2 clock）

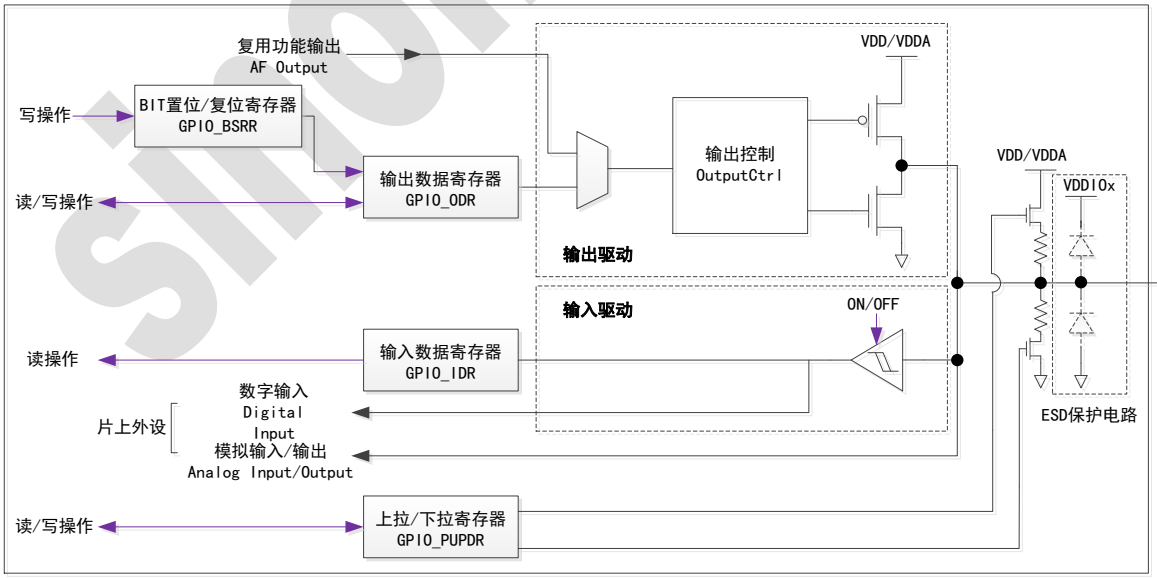
7.3 功能描述

每个 GPIO 口支持以下多种模式：

- 浮空输入
- 上拉输入
- 下拉输入
- 模拟输入
- 具有上拉或下拉能力的开漏输出
- 具有上拉或下拉能力的推挽输出
- 复用功能且具有上拉或下拉能力的推挽输出
- 复用功能且具有上拉或下拉能力的开漏输出

GPIOx_ BSRR 和 GPIOx_BRR 寄存器作用是对 GPIOx_ODR 寄存器进行读-修改-写操作，这样可以避免在读和修改操作之间发生 IRQ 带来的风险。

标准 I/O 端口基本结构框图



注： 5V 耐受类型 IO，VDDIOx 为 VDD_FT，与非 5V 耐受类型 IO 不同。

端口位配置汇总表

MODER[1:0]	OTYPER	OSPEEDR[1:0]	PUPDR[1:0]	I/O 配置

模式控制	开漏控制	IO 速度		上拉/下拉			
01	0	SPEED[1:0]		0	0	GP 输出	PP
				0	1	GP 输出	PP + PU
				1	0	GP 输出	PP + PD
				1	1	保留	
	1			0	0	GP 输出	OD
				0	1	GP 输出	OD + PU
				1	0	GP 输出	OD + PD
				1	1	保留 (GP 输出 OD)	
10	0	SPEED[1:0]		0	0	AF 复用功能	PP
				0	1	AF 复用功能	PP + PU
				1	0	AF 复用功能	PP + PD
				1	1	保留	
	1			0	0	AF 复用功能	OD
				0	1	AF 复用功能	OD + PU
				1	0	AF 复用功能	OD + PD
				1	1	保留	
00	x	x	x	0	0	输入	浮空
				0	1	输入	PU
				1	0	输入	PD
				1	1	保留 (输入浮空)	
11	x	x	x	0	0	输入/输出	模拟
				0	1	保留	
				1	0		
				1	1		

注：GP-General Purpose 通用，PP-Push Pull 推挽，PU-Pull Up 上拉，PD-Pull Down 下拉，OD-Open Drain 开漏，AF-Alternate Function 复用功能

7.3.1 通用 IO（GPIO）

复位期间和刚复位后，复用功能不开启，大多数 I/O 端口被配置为输入浮空模式。

复位后，调试引脚被置为 复用功能+上拉/下拉 模式：

- PA14：SWCLK +下拉模式
- PA13：SWDIO +上拉模式

当配置作为输出时，写到输出数据寄存器(GPIOx_ODR)的值输出到相应的 I/O 引脚上。支持推挽模式或开漏模式（仅输出低被驱动，输出高为高阻态）。

输入数据寄存器(GPIOx_IDR)在每个 AHB 时钟周期捕捉 I/O 引脚上的数据。

所有 GPIO 引脚都有一个内部弱上拉和弱下拉电阻，通 GPIOx_PUPDR 寄存器可以配置为开启或关闭。

7.3.2 I/O 复用功能映射

内置外设/模块功能通过多路复用器连接 I/O 口。同一时刻仅允许一个外设的复用功能（AF）连接到一个 I/O 口上，以防止同一 I/O 口上可用的外设功能冲突。

每个 I/O 引脚支持最多 16 个复用功能输入 （ 从 AF0 到 AF15） 的多路复用器，其可通过配置 GPIOx_AFRL 寄存器(从 pin0 到 pin7)和 GPIOx_AFRH 寄存器(从 pin 8 到 pin15) 来实现。

- 复位后，复用器选择为 AF0 功能，通过 GPIOx_MODER 寄存器设置为复用功能（AF）模式。
- 有关每个引脚的具体复用功能分配，参见芯片数据手册。

除了这种灵活的 I/O 复用结构，部分外设的复用功能还可以映射到不同的 I/O 引脚上，用于小封装器件上优化更多的可用外设。

不同的 IO 使用，按如下原则配置：

✧ 调试功能引脚：器件复位后，这些引脚立即配置为复用功能，调试器可用。

✧ GPIO：在 GPIOx_MODER 寄存器中配置所需的 I/O 口为输出、输入或模拟功能。

✧ 外设的复用功能：

- 设置连接至 I/O 所需的 AFx ，AFx 定义在 GPIOx_AFRL 或 GPIOx_AFRH 寄存器中。
- 配置 GPIOx_OTYPER、GPIOx_PUPDR 和 GPIOx_OSPEEDER 寄存器，来选择相应引脚的开漏、上拉/下拉和输出速度。
- 在 GPIOx_MODER 寄存器中配置所需的 I/O 口为复用功能模式。

✧ 附加功能：

- 对于 ADC，无论配置 GPIO 模式如何，ADC 连接都可以在 ADC 寄存器启用。建议，使用 ADC 时，在 GPIOx_MODER 寄存器中配置所需的 I/O 口为模拟模式。
- 对于附加功能如 RTC、WKUPx 和振荡器，在相关的 RTC、PWR 和 RCC 寄存器配置相应所需的功能。这些功能的优先级高于标准 GPIO 寄存器中的配置。

有关详细的 I/O 口复用功能映射，请参见器件数据手册。

7.3.3 I/O 端口控制寄存器

每组 GPIO 都有 4 个 32 位配置寄存器(GPIOx_MODER，GPIOx_OTYPER，GPIOx_ OSPEEDR 和 GPIOx_PUPDR)。

GPIOx_MODER，用来配置 IO 模式（输入、输出、复用或模拟）。

GPIOx_OTYPER，用来选择输出类型（开漏或推挽）。

GPIOx_ OSPEEDR，用来控制输出速度。

GPIOx_PUPDR，用来选择上/下拉方式。

7.3.4 I/O 端口数据寄存器

每组 GPIO 都有 2 个 16 位数据寄存器（GPIOx_IDR 和 GPIOx_ODR）。

GPIOx_IDR，从 I/O 口的输入数据存放在（GPIOx_ IDR）寄存器中，该寄存器为只读寄存器.

GPIOx_ODR，用于存储输出数据，可进行读/写访问。

7.3.5 I/O 数据位处理

每组 GPIO 都有 1 个 32 位 置位/复位寄存器(GPIOx_BSRR)。

GPIOx_BSRR，用于对输出数据寄存器（GPIOx_ODR）每个位进行置位/复位操作。其有效位宽为 GPIOx_ODR 的两倍。

对于 GPIOx_ODR 中的每一位，在 GPIOx_BSRR 中有 2 位与之对应：BS(i)和 BR(i)。对 BS(i)写 1，置位对应的 ODR(i)位；对 BR(i)写 1，复位对应的 ODR(i)位。

对 GPIOx_BSRR 的任意位写 0，无效，不影响 GPIOx_ODR 的值。若对 BS(i)和 BR(i)同时写 1，置位优先（即执行对应位的置位操作）。

通过 GPIOx_BSRR 寄存器对 GPIOx_ODR 的各个位操作，为“单次”行为，不会锁存 GPIOx_ODR 对应位。GPIOx_ODR 的位仍可以被直接访问。GPIOx_BSRR 寄存器提供一种执行原子位操作方法。

通过 GPIOx_BSRR 对 GPIOx_ODR 的位操作，不需要关闭中断，因为单 AHB 周期写访问可以实现一个或多个位修改。

7.3.6 GPIO 锁定机制

GPIOA 和 GPIOB 有 1 个 32 位锁定寄存器(GPIOx_ LCKR)。

GPIOx_ LCKR,用来冻结 GPIOA 和 GPIOB 的控制寄存器,包括:GPIOx_MODER,GPIOx_OTYPER,GPIOx_OSPEEDR,GPIOx_PUPDR,GPIOx_AFRL 和 GPIOx_AFRH。

写 GPIOx_LCKR 寄存器,必须执行一个特定的写/读序列。当正确的锁定序列作用于这个寄存器的 bit16 时，LCKR[15:0]的值用来锁定 I/O 口的配置（在写序列期间要保持 LCKR[15:0]值不变）。当锁定序列已被应用到一个端口位，该端口位不能不改变，直到下次 MCU 复位或外设复位。每个 GPIOx_LCKR 的位冻结对应控制寄存器中相应的位。

锁定序列只能按字（32 位）访问 GPIOx_ LCKR 寄存器，因为 GPIOx_LCKR 的 bit16 和 bit[15:0]必须同时设置。

详情参见 GPIO 端口配置锁定寄存器章节。

7.3.7 I/O 复用功能输入/输出

GPIOA 和 GPIOB 有 2 个 32 位复用功能寄存器（GPIOx_AFRH 和 GPIOx_AFRL)。

用户通过设置这两个寄存器，根据应用需求选择每个 GPIO 的复用功能。

详细复用功能映射，请参见产品***数据手册***。

7.3.8 外部中断/唤醒线

所有 I/O 口都可以配置为外部中断口，若要 I/O 用作外部中断 Line，对应的端口不能配置为模拟模式或用作振荡器引脚，这样输入触发器就会保持开启状态。

细节请参考外部中断与事件章节。

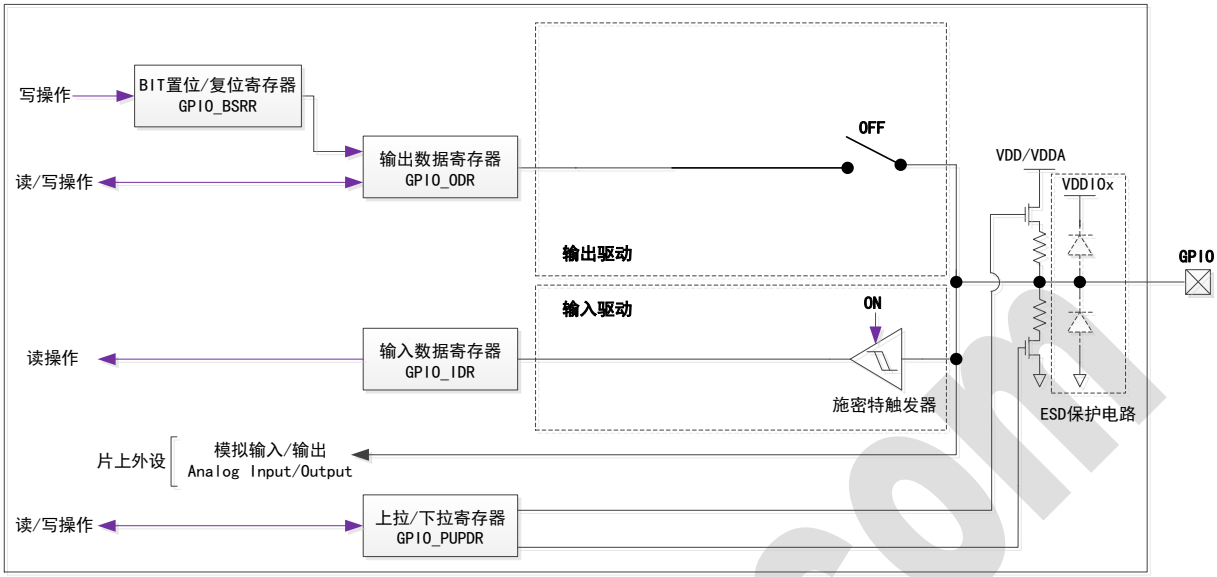
7.3.9 输入配置

当 I/O 配置为输入模式：

- 输出缓冲器关闭
- 输入施密特触发器激活
- 由 GPIOx_PUPDR 寄存器激活上拉/下拉电阻控制
- 每个 AHB 时钟周期，I/O 引脚上的数据被采样进入输入数据寄存器

- 通过读取输入数据寄存器获取 I/O 状态

输入浮空/上拉/下拉配置

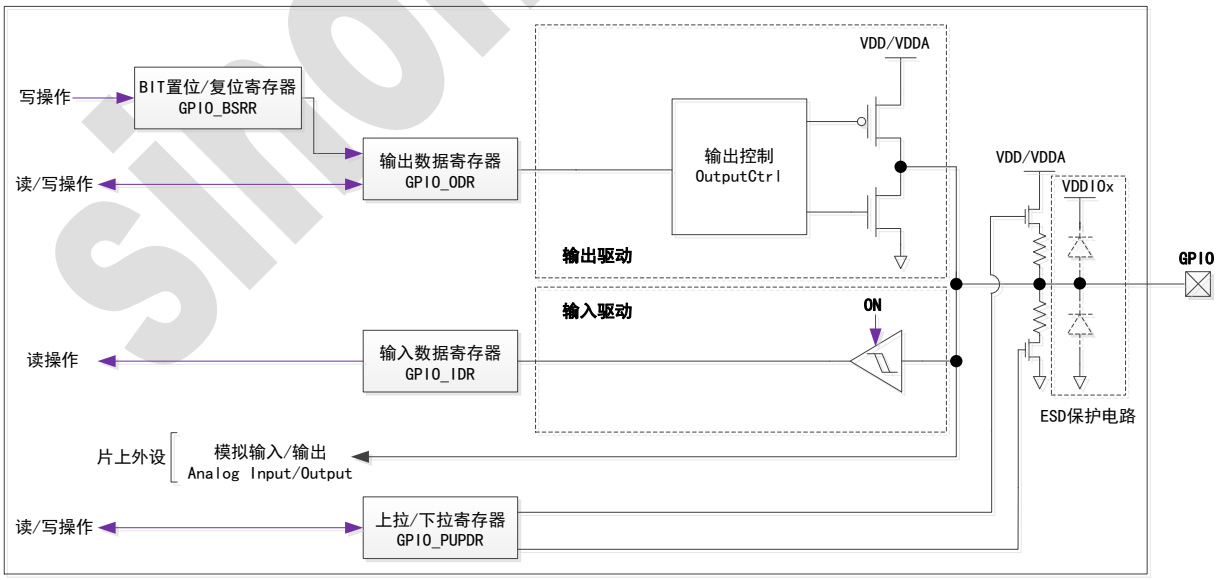


7.3.10 输出配置

当 I/O 配置为输出模式：

- 输出缓冲器开启
  开漏模式：输出数据寄存器写 0 激活输出 N-MOS（输出低），写 1 端口处于高阻态（P-MOS 不被激活）。
  推挽模式：输出数据寄存器写 0 激活输出 N-MOS（输出低），写 1 激活输出 P-MOS（输出高）。
- 输入施密特触发器激活
- 由 GPIOx_PUPDR 寄存器激活上拉/下拉电阻控制
- 每个 AHB 时钟周期，I/O 引脚上的数据被采样进入输入数据寄存器
- 通过读取输入数据寄存器获取 I/O 状态
- 通过读取输出数据寄存器获取最后写入该寄存器的值

输出配置



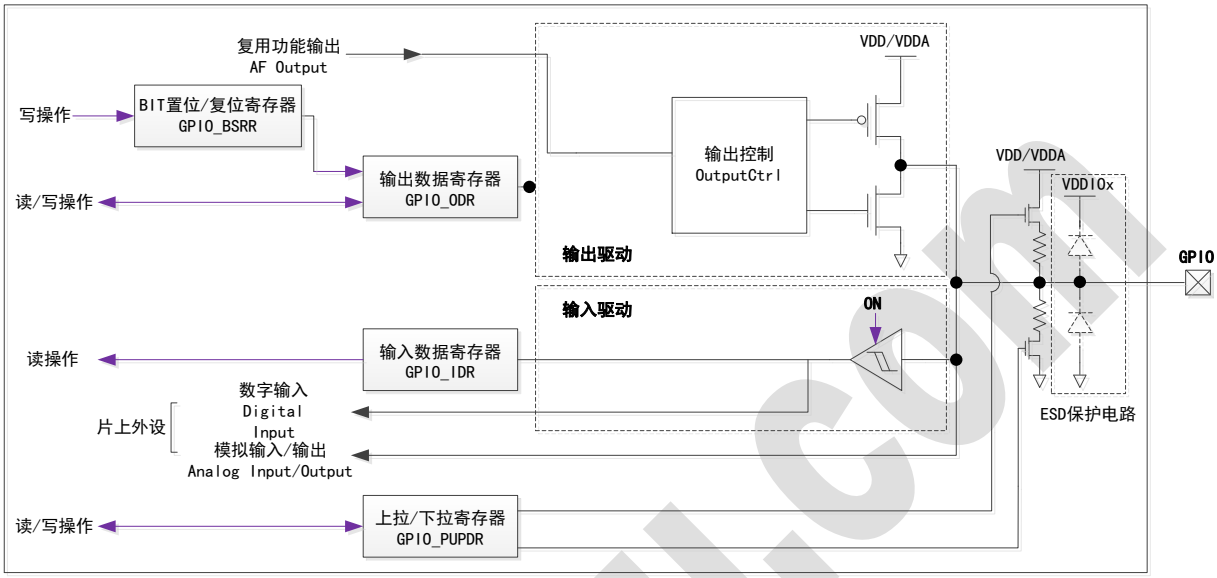
7.3.11 复用功能配置

当 I/O 配置为复用功能模式：

- 输出缓冲器可被配置为开漏或推挽

- 外设信号（发送使能或数据）控制输出缓冲器
- 输入施密特触发器激活
- 由 GPIOx_PUPDR 寄存器激活上拉/下拉电阻控制
- 每个 AHB 时钟周期，I/O 引脚上的数据被采样进入输入数据寄存器
- 通过读取输入数据寄存器获取 I/O 状态

复用功能配置

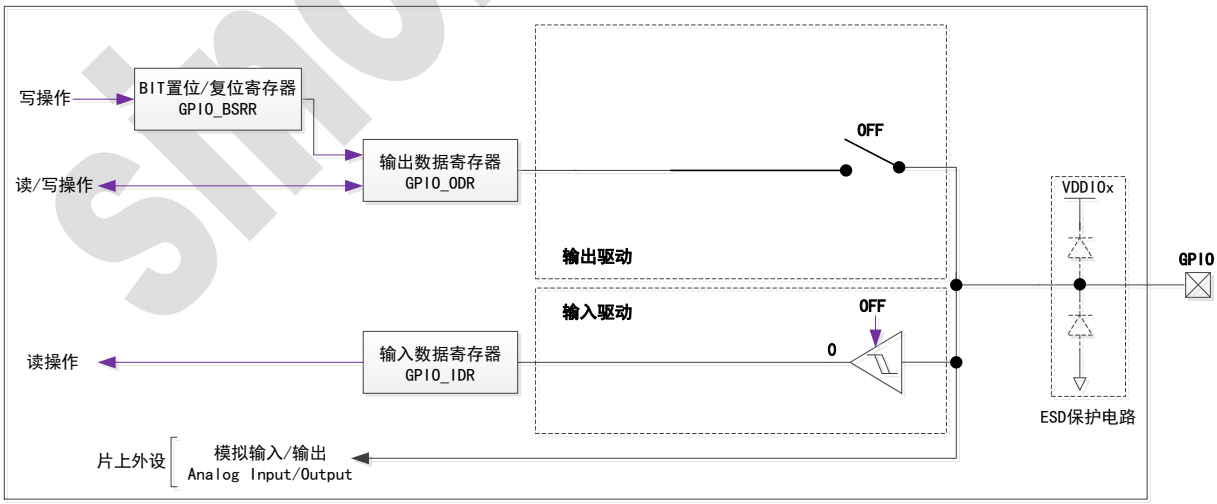


7.3.12 模拟功能配置

当 I/O 配置为模拟功能模式：

- 输出缓冲器关闭
- 输入施密特触发器禁止输入，减少模拟端口功耗，施密特输出（数字输入）固定为 0
- 弱上拉/下拉电阻硬件关闭
- 每个 AHB 时钟周期，I/O 引脚上的数据被采样进入输入数据寄存器
- 读取输入数据寄存器固定为 0

高阻/模拟配置



7.3.13 HSE 或 LSE 引脚用作 GPIO

当 HSE 或 LSE 振荡器关闭时（复位后的缺省状态），相关振荡器引脚可以用做普通的 GPIO 口。当 HSE 或 LSE 振荡器开启（在 RCC_CSR 寄存器设置 HSEON 或 LSEON 位来开启），振荡器控制其相关引脚，且相关引脚的 GPIO 配置无效。当振荡器配置为用户外部时钟方式，仅使 OSC_IN 或 OSC32_IN 引脚做为时钟输入，OSC_OUT 或 OSC32_OUT 引脚仍可以配置为普

通的 GPIO 引脚。

7.3.14 备份域下 GPIO 使用

当 VREG1P5 域断电（当器件进入待机模式）时，PC13/PC14/PC15 GPIO 功能失去。在这种情况下，若这些 GPIO 配置没有被 RTC 配置旁路，这些引脚被设为模拟输入模式。

有关 I/O 由 RTC 控制的详情，参见 RTC 功能描述章节。

7.4 相关寄存器

此外设寄存器支持字（32 位）、半字（16 位）和字节访问。

7.4.1 寄存器汇总表

基址：0x4800 0000（GPIOA），0x4800 0400（GPIOB），0x4800 0800（GPIOC），0x4800 1400（GPIOF）				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**GPIOx_MODER**	0x2800 0000 注 1 0x0000 0000	GPIO 端口模式寄存器	7.4.2
0x04	**GPIOx_OTYPER**	0x0000 0000	GPIO 端口输出类型寄存器	7.4.3
0x08	**GPIOx_OSPEEDR**	0x0C00 0000 注 2 0x0000 0000	GPIO 端口输出速度寄存器	7.4.4
0x0C	**GPIOx_PUPDR**	0x2400 0000 注 3 0x0000 0000	GPIO 端口上拉/下拉寄存器	7.4.5
0x10	**GPIOx_IDR**	0x0000 XXXX	GPIO 端口输入数据寄存器	7.4.6
0x14	**GPIOx_ODR**	0x0000 0000	GPIO 端口输出数据寄存器	7.4.7
0x18	**GPIOx_BSRR**	0x0000 0000	GPIO 端口位置位/复位寄存器	7.4.8
0x1C	**GPIOx_LCKR**	0x0000 0000	GPIO 端口配置锁定寄存器	7.4.9
0x20	**GPIOx_AFRL**	0x0000 0000	GPIO 复用功能低位寄存器	7.4.10
0x24	**GPIOx_AFRH**	0x0000 0000	GPIO 复用功能高位寄存器	7.4.11
0x28	**GPIOx_BRR**	0x0000 0000	GPIO 端口位复位寄存器	7.4.12

注 1：GPIOx_MODER，port A 复位值 0x2800 0000，其他 port 复位值 0x0000 0000。
注 2：GPIOx_OSPEEDR，port A 复位值 0x0C00 0000，其他 port 复位值 0x0000 0000。
注 3：GPIOx_PUPDR，port A 复位值 0x2400 0000，其他 port 复位值 0x0000 0000。

7.4.2 GPIO 端口模式寄存器（GPIOx_MODER）（x=A,B,C,F）

GPIOx_MODER（x=A,B,C,F）			地址偏移	0x00		复位值	0x2800 0000（GPIOA） 0x0000 0000（其他）	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	MODER15[1:0]		MODER14[1:0]		MODER13[1:0]		MODER12[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	1/0 注	0	1/0 注	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	MODER11[1:0]		MODER10[1:0]		MODER9[1:0]		MODER8[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	MODER7[1:0]		MODER6[1:0]		MODER5[1:0]		MODER4[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	MODER3[1:0]		MODER2[1:0]		MODER1[1:0]		MODER0[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

注：GPIOx_MODER，port A 复位值 0x2800 0000，其他 port 复位值 0x0000 0000。

BIT[2m+1:2m]	MODERm[1:0]	GPIOx 的模式配置位（x=A,B,C,F；m=0~15），软件置位和清零 00：输入模式 01：通用输出模式 10：复用功能模式

		11：模拟模式

7.4.3 GPIO 端口输出类型寄存器（GPIOx_OTYPER）（x=A,B,C,F）

GPIOx_OTYPER（x=A,B,C,F）			地址偏移	0x04		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	OT15	OT14	OT13	OT12	OT11	OT10	OT9	OT8
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	OT7	OT6	OT5	OT4	OT3	OT2	OT1	OT0
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	-	保留，保持复位值
BIT[15:0]	OT[15:0]	GPIOx 的输出类型配置位，软件置位和清零 0：推挽输出（复位状态） 1：开漏输出

7.4.4 GPIO 口输出速度寄存器（GPIOx_OSPEEDR）（x=A,B,C,F）

GPIOx_OSPEEDR（x=A,B,C,F）			地址偏移	0x08		复位值	0x0C00 0000（GPIOA） 0x0000 0000（其他）	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	OSPEEDR15[1:0]		OSPEEDR14[1:0]		OSPEEDR13[1:0]		OSPEEDR12[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0/1 注	0/1 注	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	OSPEEDR11[1:0]		OSPEEDR10[1:0]		OSPEEDR9[1:0]		OSPEEDR8[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	OSPEEDR7[1:0]		OSPEEDR6[1:0]		OSPEEDR5[1:0]		OSPEEDR4[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	OSPEEDR3[1:0]		OSPEEDR2[1:0]		OSPEEDR1[1:0]		OSPEEDR0[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

注：GPIOx_OSPEEDR，port A 复位值 0x0C00 0000，其他 port 复位值 0x0000 0000。

BIT[2m+1:2m]	OSPEEDRm[1:0]	GPIOx 的输出速度配置位（x=A,B,C,F；m=0~15），软件置位和清零 x0：低速 01：中速 11：高速 *注：每个速度档位对应频率、电源及负载条件，请参见产品数据手册。*

7.4.5 GPIO 口上拉/下拉寄存器（GPIOx_PUPDR）（x=A,B,C,F）

GPIOx_PUPDR（x=A,B,C,F）	地址偏移	0x0C	复位值	0x2400 0000（GPIOA） 0x0000 0000（其他）

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	PUPDR15[1:0]		PUPDR14[1:0]		PUPDR13[1:0]		PUPDR12[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0/1 注	0	0	0/1 注	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	PUPDR11[1:0]		PUPDR10[1:0]		PUPDR9[1:0]		PUPDR8[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	PUPDR7[1:0]		PUPDR6[1:0]		PUPDR5[1:0]		PUPDR4[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	PUPDR3[1:0]		PUPDR2[1:0]		PUPDR1[1:0]		PUPDR0[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

注：GPIOx_PUPDR，port A 复位值 0x2400 0000，其他 port 复位值 0x0000 0000。

BIT[2m+1:2m]	PUPDRm[1:0]	GPIOx 的上拉/下拉配置位（x=A,B,C,F；m=0~15），软件置位和清零 00：无上拉和下拉 01：上拉 10：下拉 11：保留

7.4.6 GPIO 口输入数据寄存器（GPIOx_IDR）（x=A,B,C,F）

GPIOx_IDR（x=A,B,C,F）			地址偏移	0x10		复位值	0x0000 XXXX	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8
R/W	r	r	r	r	r	r	r	r
复位值	X	X	X	X	X	X	X	X
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
R/W	r	r	r	r	r	r	r	r
复位值	X	X	X	X	X	X	X	X

BIT[31:16]	-	保留，保持复位值
BIT[15:0]	IDR[15:0]	端口输入数据位，只读 相应 I/O 口的输入值。

7.4.7 GPIO 端口输出数据寄存器（GPIOx_ODR）（x=A,B,C,F）

GPIOx_ODR（x=A,B,C,F）			地址偏移	0x14		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-

复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	-	保留，保持复位值
BIT[15:0]	ODR[15:0]	GPIOx 输出数据，软件置位和清零 *注：对于原子置位/复位，可通过对 GPIOx_BSRR 和 GPIOx_BRR 寄存器操作来实现。*

7.4.8 GPIO 端口位置位/复位寄存器（GPIOx_BSRR）（x=A,B,C,F）

GPIOx_BSRR（x=A,B,C,F）			地址偏移	0x18		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	BR[15:0]	GPIOx 复位位操作，只写，读这些位返回 0x0000 0：对相应的 ODRx 位无影响 1：复位相应的 ODRx 位 *注：若 BSx 和 BRx 同时置位，BSx（置位）有优先权。*
BIT[15:0]	BS[15:0]	GPIOx 置位位操作，只写，读这些位返回 0x0000 0：对相应的 ODRx 位无影响 1：置位相应的 ODRx 位

7.4.9 GPIO 端口配置锁定寄存器（GPIOx_LCKR）(x=A,B)

访问：锁定序列只能按字（32 位）访问，因为 GPIOx_LCKR 的 bit16 和 bit[15:0]必须同时设置。

GPIOx_LCKR（x=A,B）			地址偏移	0x1C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	LCKK
R/W	-	-	-	-	-	-	-	rw
复位值	-	-	-	-	-	-	-	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8
R/W	rw	rw	rw	rw	rw	rw	rw	rw

复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:17]	-	保留，保持复位值
BIT[16]	LCKK	锁定键 此位可随时读取，但仅能通过对锁定键特定的写/读序列被改写。 0：端口配置锁定不激活 1：端口配置锁定激活。GPIOx_LCKR 寄存器会被锁定（不能被修改），直到下一个 MCU 复位或外设复位发生。 **锁定键写序列：** - 写 LCKR[16] = ‘1’ + LCKR[15:0] - 写 LCKR[16] = ‘0’ + LCKR[15:0] - 写 LCKR[16] = ‘1’ + LCKR[15:0] - 读 LCKR - 读 LCKR[16] = ‘1’（这个读操作可选，但可以确认锁定是否激活 ） *注 1：在锁定键写序列期间， LCK[15:0]值必须保持不变。在锁定写序列中出现任何错误都会中止锁定操作。* *注 2：在第一次锁定序列操作后，任何读 LCKK 位时将返回‘1’直到下次 CPU 复位或外设复位。*
BIT[15:0]	LCK[15:0]	GPIOx 配置锁定位 这些位可以读/写，但仅在 LCKK 为‘0’时写操作。 0：端口配置未锁定 1：端口配置锁定

7.4.10 GPIO 复用功能低位寄存器（GPIOx_AFRL）(x=A,B)

GPIOx_AFRL（x=A,B）			地址偏移	0x20		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	AFRL7[3:0]				AFRL6[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	AFRL5[3:0]				AFRL4[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	AFRL3[3:0]				AFRL2[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	AFRL1[3:0]				AFRL0[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:0]	AFRLm[3:0]	GPIOx pin m 的复用功能选择 (m= 0~7)，软件置位和清零 0000：AF0 0001：AF1 0010：AF2 0011：AF3 0100：AF4 0101：AF5 0110：AF6 0111：AF7 其他：保留

7.4.11 GPIO 复用功能高位寄存器（GPIOx_AFRH）(x=A,B)

GPIOx_AFRH（x=A,B）			地址偏移	0x24		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	AFRH15[3:0]				AFRH14[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	-
复位值	0	0	0	0	0	0	0	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	AFRH13[3:0]				AFRH12[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	AFRH11[3:0]				AFRH10[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	AFRH9[3:0]				AFRH8[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:0]	AFRHm[3:0]	GPIOx pin m 的复用功能选择 (m= 8~15)，软件置位和清零 0000：AF0 0001：AF1 0010：AF2 0011：AF3 0100：AF4 0101：AF5 0110：AF6 0111：AF7 其他：保留

7.4.12 GPIO 端口复位寄存器（GPIOx_BRR）(x=A..G)

GPIOx_BRR（x=A,B,C,F）			地址偏移	0x28		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	-	保留，保持复位值
BIT[15:0]	BR[15:0]	GPIOx 复位位操作，只写，读这些位返回 0x0000 0：对相应的 ODRx 位无影响 1：复位相应的 ODRx 位 *注：与 GPIOx_BSRR->BR[15:0]位相同。*

8 系统配置控制（SYSCFG）

8.1 概述

系统配置寄存器主要功能如下：

- ✧ 在部分 IO 口上启用或禁用 I2C 快速模式+（Fast Mode Plus）。
- ✧ 重映射来自 TIM16 和 TIM17，USART1 和 ADC 的 DMA 触发源到不同的 DMA 通道。
- ✧ 重映射代码启动区的存储器。
- ✧ 管理连接到 GPIO 口的外部中断 Line。
- ✧ 管理系统的可靠性特性。

8.2 相关寄存器

8.2.1 寄存器汇总表

基址：0x4001 0000				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**SYSCFG_CFGR1**	0x0000 000X	SYSCFG 配置寄存器 1	8.2.2
0x04	-	-	-	保留
0x08	**SYSCFG_EXTICR1**	0x0000 0000	外部中断配置寄存器 1	8.2.3
0x0C	**SYSCFG_EXTICR2**	0x0000 0000	SYSCFG 外部中断配置寄存器 2	8.2.4
0x10	**SYSCFG_EXTICR3**	0x0000 0000	SYSCFG 外部中断配置寄存器 3	8.2.5
0x14	**SYSCFG_EXTICR4**	0x0000 0000	SYSCFG 外部中断配置寄存器 4	8.2.6
0x18	**SYSCFG_CFGR2**	0x0000 0000	SYSCFG 配置寄存器 2	8.2.7

8.2.2 SYSCFG 配置寄存器 1（SYSCFG_CFGR1）

此寄存器专门用于配置内存映射、DMA 请求映射以及 I2C 快速模式+控制功能。

MEM_MODE[1:0]用于选择代码启动的存储器映射，通过软件选择代码从不同的存储器启动，从而可以旁路硬件 BOOT 的控制。复位后此位值取决于实际的 BOOT 配置。

SYSCFG_CFGR1			地址偏移	0x00		复位值	0x0000 000X	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	I2C_PA10_FMP	I2C_PA9_FMP	-	I2C1_FMP	I2C_PB9_FM+	I2C_PB8_FM+	I2C_PB7_FM+	I2C_PB6_FM+
R/W	rw	rw	-	rw	rw	rw	rw	rw
复位值	0	0	-	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	TIM17_DMA_RMP	TIM16_DMA_ RMP	USART1_RX_DMA_RMP	USART1_TX_DMA_ RMP	ADC_DMA_RMP
R/W	-	-	-	rw	rw	rw	rw	rw
复位值	-	-	-	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	MEM_MODE	
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	X	X

BIT[31:24]	-	保留，保持复位值
BIT[23:22]	I2C_PAx_FMP	快速模式+（FM+）驱动能力激活位，软件置位和清零 每个位分别为 PA9 和 PA10 引脚开启 I2C FM+模式。 0：PAx 引脚设置为标准模式 1：PAx 引脚设置为快速模式+（FM+），且 I2C 速度控制被旁路（被忽略）
BIT[20]	I2C1_FMP	I2C1 超快模式（FM+）驱动能力激活位，软件置位和清零 此位与 I2C_Pxx_FMP 位是或逻辑关系。 0：FM+模式仅由 I2C_Pxx_FMP 控制

		1：GPIOx_AFR 寄存器配置选择的所有 I2C1 引脚，全部使能 FM+模式
BIT[19:16]	I2C_PBx_FMP	快速模式+（FM+）驱动能力激活位，软件置位和清零 每个位分别为 PB6、PB7、PB8 和 PB9 引脚开启 I2C FM+模式。 0：PAx 引脚设置为标准模式 1：PAx 引脚设置为快速模式+（FM+），且 I2C 速度控制被旁路（被忽略）
BIT[15:13]	-	保留，保持复位值
BIT[12]	TIM17_DMA_RMP	TIM17 DMA 请求重映射位，软件置位和清零 此位控制 TIM17 DMA 通道请求的重映射。 0：无重映射（TIM17_CH1 和 TIM17_UP DMA 请求映射在 DMA 通道 1 上） 1：重映射（TIM17_CH1 和 TIM17_UP DMA 请求映射在 DMA 通道 2 上）
BIT[11]	TIM16_DMA_RMP	TIM16 DMA 请求重映射位，软件置位和清零 此位控制 TIM16 DMA 通道请求的重映射。 0：无重映射（TIM16_CH1 和 TIM16_UP DMA 请求映射在 DMA 通道 3 上） 1：重映射（TIM16_CH1 和 TIM16_UP DMA 请求映射在 DMA 通道 4 上）
BIT[10]	USART1_RX_DMA_RMP	USART1_RX DMA 请求重映射位，软件置位和清零 此位控制 USART1_RX DMA 通道请求的重映射。 0：无重映射（USART1_RX 请求映射在 DMA 通道 3 上） 1：重映射（USART1_RX 请求映射在 DMA 通道 5 上）
BIT[9]	USART1_TX_DMA_RMP	USART1_TX DMA 请求重映射位，软件置位和清零 此位控制 USART1_TX DMA 通道请求的重映射。 0：无重映射（USART1_TX 请求映射在 DMA 通道 2 上） 1：重映射（USART1_TX 请求映射在 DMA 通道 4 上）
BIT[8]	ADC_DMA_RMP	ADC DMA 请求重映射位，软件置位和清零 此位控制 ADC DMA 通道请求的重映射。 0：无重映射（ADC DMA 请求映射在 DMA 通道 1 上） 1：重映射（ADC DMA 请求映射在 DMA 通道 2 上）
BIT[7:2]	-	保留，保持复位值
BIT[1:0]	MEM_MODE[1:0]	存储映射选择位，软件置位和清零 当复位后，此位值由实际 boot 模式配置值决定。详细请参考 3.2.5 启动配置章节。 x0：主闪存存储器映射到 0x0000 0000 01：系统存储器映射到 0x0000 0000 11：嵌入式 SRAM 映射到 0x0000 0000

8.2.3 SYSCFG 外部中断配置寄存器 1（SYSCFG_EXTICR1）

SYSCFG_EXTICR1			地址偏移	0x08		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	EXTI3[3:0]				EXTI2[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	EXTI1[3:0]				EXTI0[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	-	保留，保持复位值
BIT[15:0]	EXTIx[3:0]	EXTI x 配置位（x = 0~3），软件置位和清零，选择 EXTIx 的外部中断源。 x000：PA[x] 引脚 x001：PB[x] 引脚 x010：PC[x] 引脚

		x011：保留 x100：保留 x101：PF[x] 引脚 其它：保留

注 ： 上述提及的部分 I/O 口线在小封装的器件中可能不可用。

8.2.4 SYSCFG 外部中断配置寄存器 2（SYSCFG_EXTICR2）

SYSCFG_EXTICR2			地址偏移	0x0C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	EXTI7[3:0]				EXTI6[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	EXTI5[3:0]				EXTI4[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	-	保留，保持复位值
BIT[15:0]	EXTIx[3:0]	EXTI x 配置位（x = 4~7），软件置位和清零，选择 EXTIx 的外部中断源。 x000：PA[x] 引脚 x001：PB[x] 引脚 x010：PC[x] 引脚 x011：保留 x100：保留 x101：PF[x] 引脚 其它：保留

注 ： 上述提及的部分 I/O 口线在小封装的器件中可能不可用。

8.2.5 SYSCFG 外部中断配置寄存器 3（SYSCFG_EXTICR3）

SYSCFG_EXTICR3			地址偏移	0x10		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	EXTI11[3:0]				EXTI10[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	EXTI9[3:0]				EXTI8[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	-	保留，保持复位值
BIT[15:0]	EXTIx[3:0]	EXTI x 配置位（x = 8~11），软件置位和清零，选择 EXTIx 的外部中断源。

		x000：PA[x] 引脚 x001：PB[x] 引脚 x010：PC[x] 引脚 x011：保留 x100：保留 x101：PF[x] 引脚 其它：保留

注 ： 上述提及的部分 I/O 口线在小封装的器件中可能不可用。

8.2.6 SYSCFG 外部中断配置寄存器 4（SYSCFG_EXTICR4）

SYSCFG_EXTICR4			地址偏移	0x14		复位值	0x0000 0000
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	EXTI15[3:0]				EXTI14[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	EXTI13[3:0]				EXTI12[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	-	保留，保持复位值
BIT[15:0]	EXTIx[3:0]	EXTI x 配置位（x = 12~15），软件置位和清零，选择 EXTIx 的外部中断源。 x000：PA[x] 引脚 x001：PB[x] 引脚 x010：PC[x] 引脚 x011：保留 x100：保留 x101：PF[x] 引脚 其它：保留

注 ： 上述提及的部分 I/O 口线在小封装的器件中可能不可用。

8.2.7 SYSCFG 配置寄存器 2（SYSCFG_CFGR2）

SYSCFG_CFGR2			地址偏移	0x18		复位值	0x0000 0000
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	SRAM_PEF
R/W	-	-	-	-	-	-	-	rc_w1
复位值	-	-	-	-	-	-	-	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	PVD_LOCK	SRAM_PARITY_LOCK	LOCKUP_LOCK

R/W	-	-	-	-	-	rw	rw	rw
复位值	-	-	-	-	-	0	0	0

BIT[31:9]	-	保留，保持复位值
BIT[8]	SRAM_PEF	SRAM 校验错误标志 当 SRAM 校验错误被检测到时，硬件置位。对该位写 1 清零。 0: 无 SRAM 校验错误 1: 检测到 SRAM 校验错误
BIT[7:3]	-	保留，保持复位值
BIT[2]	PVD_LOCK	PVD 锁定使能位，软件置位，系统复位清零 可用于使能并锁定 PVD 连接到/ TIM1/16/17 的刹车 (Break) 输入 ，锁定 PVDE 和 PLS[2:0] (PWR_CR 寄存器) 0: PVD 中断信号断开与 TIM1/16/17 刹车 (Break) 输入的连接。PVDE 和 PLS[2:0]位可被应用编程。 1: PVD 中断信号连接到 TIM1/16/17 刹车 (Break) 输入 ，PVDE 和 PLS[2:0] 位为只读。
BIT[1]	SRAM_PARITY_LOCK	SRAM 校验锁定位，软件置位，系统复位清零 可用于锁定 SRAM 校验错误信号连接到 TIM1/16/17 的刹车 (Break) 输入。 0: SRAM 校验错信号断开与 TIM1/16/17 刹车 (Break) 输入的连接 1: SRAM 校验错信号连接到 TIM1/16/17 刹车 (Break) 输入
BIT[0]	LOCKUP_LOCK	Cortex-M0 LOCKUP 位使能位，软件置位，系统复位清零 可用于使能并锁定连接 Cortex-M0 LOCKUP(硬件故障)输出到 TIM1/16/17 的刹车(Break) 输入。 0: Cortex-M0 LOCKUP 输出断开与 TIM1/16/17 刹车 (Break) 输入的连接 1: Cortex-M0 LOCKUP 输出连接到 TIM1/16/17 刹车 (Break) 输入

9 中断和事件（NVIC/Systick/EXTI）

9.1 嵌套向量中断控制（NVIC）

本产品内置嵌套向量中断控制器，能够处理多达 32 个可屏蔽中断通道(不包括 16 个 Cortex-M0 的中断线)和 4 个可编程优先级。

- ✧ 紧耦合的 NVIC 能够达到低延迟的中断响应处理
- ✧ 中断向量入口地址直接进入内核
- ✧ 紧耦合的 NVIC 接口
- ✧ 允许中断的早期处理
- ✧ 处理晚到的较高优先级中断
- ✧ 支持中断末尾连锁功能
- ✧ 自动保存处理器状态
- ✧ 中断返回时自动恢复现场，无需额外指令开销

该模块以最小的中断延迟提供灵活的中断管理功能。

中断异常向量表

位置	优先级	优先级类型	名称	说明	地址
	-	-	-	保留 (Reserved)	0x0000 0000
	-3	固定	Reset	复位 (Reset)	0x0000 0004
	-2	固定	NMI	不可屏蔽中断。RCC 时钟安全系统(CSS)联接到 NMI 向量	0x0000 0008
	-1	固定	HardFault	所有类型的错误 (fault)	0x0000 000C
	-	可设置	-	保留	0x0000 0010 ~0x0000 002B
	3	可设置	SVCall	通过 SWI 指令调用的系统服务	0x0000 002C
	-	可设置	-	保留	0x0000 0030 ~0x0000 0034
	5	可设置	PendSV	可挂起的系统服务	0x0000 0038
	6	可设置	SysTick	系统嘀嗒定时器	0x0000 003C
0	7	可设置	WWDG	窗口看门狗中断	0x0000 0040
1	8	可设置	PVD	连接到 EXTI Line16 的可编程电压检测(PVD) 中断	0x0000 0044
2	9	可设置	RTC	RTC 全局中断，EXTI line17/19	0x0000 0048
3	10	可设置	FLASH	Flash 全局中断	0x0000 004C
4	11	可设置	RCC	RCC 全局中断	0x0000 0050
5	12	可设置	EXTI0_1	EXTI Line[1:0] 中断	0x0000 0054
6	13	可设置	EXTI2_3	EXTI Line[3:2] 中断	0x0000 0058
7	14	可设置	EXTI4_15	EXTI Line[15:4] 中断	0x0000 005C
8	15	可设置	-	保留	0x0000 0060
9	16	可设置	DMA_CH1	DMA 通道 1 中断	0x0000 0064
10	17	可设置	DMA_CH2_3	DMA 通道 2 和 3 中断	0x0000 0068
11	18	可设置	DMA_CH4_5	DMA 通道 4 和 5 中断	0x0000 006C
12	19	可设置	ADC	ADC 和 CMP1/2 中断 CMP1 连接至 EXTI Line21 CMP2 连接至 EXTI Line22	0x0000 0070
13	20	可设置	TIM1_BRK_UP_TRG_COM	TIM1 刹车、更新、触发、和换向中断	0x0000 0074
14	21	可设置	TIM1_CC	TIM1 捕获比较中断	0x0000 0078
15	22	可设置	TIM2	TIM2 全局中断	0x0000 007C
16	23	可设置	TIM3	TIM3 全局中断	0x0000 0080
17	24	可设置	-	保留	0x0000 0084
18	25	可设置	-	保留	0x0000 0088
19	26	可设置	TIM14	TIM14 全局中断	0x0000 008C
20	27	可设置	-	保留	0x0000 0090
21	28	可设置	TIM16	TIM16 全局中断	0x0000 0094

22	29	可设置	TIM17	TIM17 全局中断	0x0000 0098
23	30	可设置	I2C1	I2C1 全局中断（与 EXTI Line23 共用）	0x0000 009C
24	31	可设置	-	保留	0x0000 00A0
25	32	可设置	SPI1	SPI1 全局中断	0x0000 00A4
26	33	可设置	-	保留	0x0000 00A8
27	34	可设置	USART1	USART1 全局中断（与 EXTI Line25 共用）	0x0000 00AC
28	-	可设置	-	保留	0x0000 00B0
29	-	可设置	-	保留	0x0000 00B4
30	-	可设置	-	保留	0x0000 00B8
31	-	可设置	-	保留	0x0000 00BC

9.2 系统节拍定时器（Systick Timer）

这个定时器是专用于实时操作系统，也可当成一个标准的递减计数器。它具有下述特性：

- ✧ 24 位的递减计数器
- ✧ 自动重加载功能
- ✧ 当计数器为 0 时能产生一个可屏蔽系统中断
- ✧ 可编程时钟源

SysTick 校准值设置为 6000，SysTick 时钟设置为 6 MHz （fHCLK/8，fHCLK=48MHz），产生 1 ms 的参考时间基准。

9.3 外部中断与事件（EXTI）

外部中断/事件控制器（EXTI）管理外部和内部异步中断/事件，产生事件请求到 MCU 和中断控制器，产生唤醒请求到电源管理器。

外部中断/事件控制器包含 23 个边沿检测器（外部/内部事件 Line），用于产生中断/事件请求。每个外部中断 Line 都可以独立地配置它的触发事件(上升沿或下降沿或双边沿)，而内部中断 Line 有效沿固定为上升沿。

一个中断可以一直保持挂起：如果发生外部中断，一个状态寄存器被实例化并表明中断来源；一个事件通常是一个简单的脉冲，用于触发内核的唤醒(如 Cortex-M0 的 RXEV 引脚)。对于内部中断，挂起状态由生成的 IP 保证，所以不需要特定标志。

每个输入 Line 都可以独立屏蔽其中断和事件的产生，另外，内部 Line 仅在 STOP 模式下采样 。中断控制器可以通过写特定的寄存器，软件模拟外部事件（only）(与对应的硬件事件 Line 复用)。

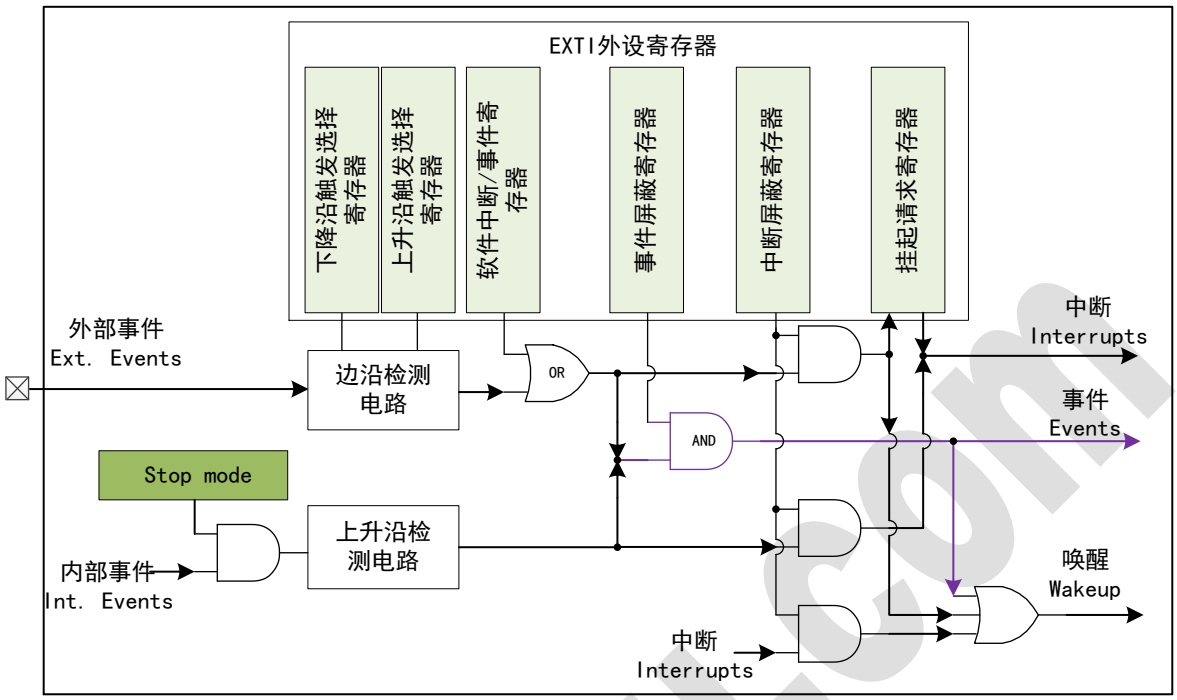
多达 39 个通用 I/O 口连接到 16 个外部中断线。

9.3.1 EXTI 特性

- ✧ 支持产生多达 23 个事件 / 中断请求；
- ✧ 作为外部或内部事件请求的每一线都可独立配置；
- ✧ 每个事件/中断线都有独立的屏蔽；
- ✧ 当系统不处于停机(STOP)模式时自动禁止内部各线；
- ✧ 独立触发外部事件/中断线；
- ✧ 每个外部中断线都有专用的状态位；
- ✧ 仿真所有的外部事件请求。

9.3.2 EXTI 框图

外部中断/事件框图



注 1：内部仅在 stop mode 有效；
注 2：软件中断/事件控制器仅对外部事件有效；

9.3.3 事件管理

芯片支持外部或内部事件唤醒内核（WFE）。唤醒事件可以通过以下方式产生：

- ✧ 外设控制寄存器中使能中断，NVIC 中不开启，并在 Cortex-MO System control 寄存器中启用 SEVONPEND 位；当 MCU 从 WFE 恢复时，EXTI 外设中断挂起位和外设 NVIC IRQ 通道挂起位(在 NVIC 中断清除挂起寄存器中)必须被清除。
- ✧ 或者，在事件模式下配置外部或内部 EXTI 线。当 CPU 从 WFE 恢复时，不需要清除外设中断挂起位或者 NVIC IRQ 通道挂起位，因为事件线对应的挂起位没有置起。

9.3.4 功能描述

对于外部中断线，要产成中断，中断线需进行配置并启用。根据预期的边沿检测，编程两个触发寄存器（trigger registers）；对中断掩码寄存器（interrupt mask register）的相应位写‘1’来启用中断请求。当所选边沿出现在外部中断线上时，就会产生一个中断请求。对应于中断线的挂起位也被置位。通过在挂起寄存器中写入’1’来复位该请求。

对于内部中断线，有效沿保持为上升沿，在中断掩码寄存器中中断默认是启用的，并且在挂起寄存器中没有对应的挂起位。

要生成事件，事件线需进行配置并启用。根据预期的边沿检测，编程两个触发寄存器（trigger registers）；对事件掩码寄存器（event mask register）的相应位写‘1’来启用事件请求。。当所选边沿出现在事件线上时，将生成一个事件脉冲。事件行对应的挂起位不会置位。

对于外部线，中断/事件请求也可以通过软件产生，在软件中断/事件寄存器（software interrupt/event register）中写入’1’来实现。

注：只有当系统处于 STOP 模式时，才能触发与内部线相关的中断或事件。如果系统仍在运行，则不会产生任何中断/事件。

硬件中断选择

要将 1 个线配置为中断源，请使用以下步骤：

- ✧ 在 EXTI_IMR 寄存器中配置相应的掩码位（mask bit）。
- ✧ 配置中断线的触发选择位(EXTI_RTSR 和 EXTI_FTSR)。
- ✧ 配置启用和掩码位，控制 NVIC IRQ 通道映射到 EXTI，以便来自 EXTI 线之一的中断可以被正确地响应。

硬件事件选择

要将 1 个线配置为事件源，请使用以下步骤：

- ✧ 在 EXTI_EMR 寄存器中配置相应的掩码位（mask bit）。

✧ 配置事件线的触发选择位(EXTI_RTSR 和 EXTI_FTSR)。

软件中断/事件选择

任何外部线都可以配置为软件中断/事件线。下面是生成软件中断的步骤：

✧ 配置相应的掩码位(EXTI_IMR, EXTI_EMR)。

✧ 置位软件中断寄存器(EXTI_SWIER)中所需的位。

9.3.5 外部和内部中断/事件线映射

所有的 GPIO 以下面方式连接到 16 个外部中断/事件线：

GPIOs 及外设	对应的 EXTI line	控制位@寄存器
PA0/PB0/PC0/PD0/PE0/PF0	EXTI0	EXTI0[3:0]@ SYSCFG_EXTICR1
PA1/PB1/PC1/PD1/PE1/PF1	EXTI1	EXTI1[3:0]@ SYSCFG_EXTICR1
……	……	……
PA15/PB15/PC15/PD15/PE15/PF15	EXTI15	EXTI1[3:0]@ SYSCFG_EXTICR4
PVD 输出	EXTI16	-
RTC Alarm 事件	EXTI17	-
保留，内部固定 low	EXTI18	-
RTC Tamper and TimeStamp 事件	EXTI19	-
保留，内部固定 low	EXTI20	-
比较器 1 输出	EXTI21	-
比较器 2 输出	EXTI22	-
内部 I2C1 唤醒事件	EXTI23	-
保留，内部固定 low	EXTI24	-
内部 USART1 唤醒事件	EXTI25	-
保留，内部固定 low	EXTI26~31	-

注：在某些产品上保留或不使用的 EXTI 线被视为内部的。

9.4 EXTI 寄存器

9.4.1 寄存器汇总表

基址：0x4001 0400				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**EXTI_IMR**	0x0FF4 0000	中断屏蔽寄存器	9.4.2
0x04	**EXTI_EMR**	0x0000 0000	事件屏蔽寄存器	9.4.3
0x08	**EXTI_RTSR**	0x0000 0000	上升沿触发选择寄存器	9.4.4
0x0C	**EXTI_FTSR**	0x0000 0000	下降沿触发选择寄存器	9.4.5
0x10	**EXTI_SWIER**	0x0000 0000	软件中断事件寄存器	9.4.6
0x14	**EXTI_PR**	0xXXXX XXXX	挂起寄存器	9.4.7

9.4.2 中断屏蔽寄存器（EXTI_IMR）

EXTI_IMR			地址偏移	0x00		复位值	0x0FF4 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	IM25	-
R/W	-	-	-	-	-	-	rw	-
复位值	-	-	-	-	-	-	1	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	IM23	IM22	IM21	-	IM19	-	IM17	IM16
R/W	rw	rw	rw	-	rw	-	rw	rw
复位值	1	1	1	-	0	-	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:0]	IMx	Line x（x=0~31）中断屏蔽 0：屏蔽来自 Line x 的中断请求 1：开启来自 Line x 的中断请求

9.4.3 事件屏蔽寄存器（EXTI_EMR）

EXTI_EMR			地址偏移	0x04		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	EM25	-
R/W	-	-	-	-	-	-	rw	-
复位值	-	-	-	-	-	-	0	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	EM23	EM22	EM21	-	EM19	-	EM17	EM16
R/W	rw	rw	rw	-	rw	-	rw	rw
复位值	0	0	0	-	0	-	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	EM15	EM14	EM13	EM12	EM11	EM10	EM9	EM8
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	EM7	EM6	EM5	EM4	EM3	EM2	EM1	EM0
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:0]	EMx	Line x（x=0~31）事件屏蔽位 0：屏蔽来自 Line x 的事件请求 1：开启来自 Line x 的事件请求

9.4.4 上升沿触发选择寄存器（EXTI_RTSR）

EXTI_RTSR			地址偏移	0x08		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	RT22	RT21	-	RT19	-	RT17	RT16
R/W	-	rw	rw	-	rw	-	rw	rw
复位值	-	0	0	-	0	-	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	RT15	RT14	RT13	RT12	RT11	RT10	RT9	RT8
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:23]	-	保留，保持复位值
BIT[22:21]	RTx	Line x 上升沿触发事件配置位（x = 21~22） 0：禁止上升沿触发（中断和事件） 1：允许上升沿触发（中断和事件）
BIT[20]	-	保留，保持复位值
BIT[19]	RTx	Line x 上升沿触发事件配置位（x = 19） 0：禁止上升沿触发（中断和事件） 1：允许上升沿触发（中断和事件）
BIT[18]	-	保留，保持复位值
BIT[17:0]	RTx	Line x 上升沿触发事件配置位（x = 0~17）

		0：禁止上升沿触发（中断和事件 ） 1：允许上升沿触发（中断和事件 ）

9.4.5 下降沿触发选择寄存器（EXTI_FTSR）

EXTI_FTSR			地址偏移	0x0C		复位值	0x0000 0000
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	FT22	FT21	-	FT19	-	FT17	FT16
R/W	-	rw	rw	-	rw	-	rw	rw
复位值	-	0	0	-	0	-	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	FT15	FT14	FT13	FT12	FT11	FT10	FT9	FT8
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	FT7	FT6	FT5	FT4	FT3	FT2	FT1	FT0
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:23]	-	保留，保持复位值
BIT[22:21]	FTx	Line x 下降沿触发事件配置位（x = 21~22） 0：禁止下降沿触发（中断和事件 ） 1：允许下降沿触发（中断和事件 ）
BIT[20]	-	保留，保持复位值
BIT[19]	FT19	Line 19 下降沿触发事件配置位 0：禁止下降沿触发（中断和事件 ） 1：允许下降沿触发（中断和事件 ）
BIT[18]	-	保留，保持复位值
BIT[17:0]	FTx	Line x 下降沿触发事件配置位（x = 0~17） 0：禁止下降沿触发（中断和事件 ） 1：允许下降沿触发（中断和事件 ）

9.4.6 软件中断事件寄存器（EXTI_SWIER）

EXTI_SWIER			地址偏移	0x10		复位值	0x0000 0000
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	SWI22	SWI21	-	SWI19	-	SWI17	SWI16
R/W	-	rw	rw	-	rw	-	rw	rw
复位值	-	0	0	-	0	-	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	SWI15	SWI14	SWI13	SWI12	SWI11	SWI10	SWI9	SWI8
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	SWI7	SWI6	SWI5	SWI4	SWI3	SWI2	SWI1	SWI0
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:23]	-	保留，保持复位值
BIT[22:21]	SWIx	Line x 的软中断位（x = 21~22）

		当该位为‘0’时，写‘1’将置位 EXTI_PR 中相应的挂起位。 如果在 EXTI_IMR 允许此 Line 产生该中断，则此时将产生一个中断请求。 通过对 EXTI_PR 的对应位写入‘1’，可以清除该位为’0’。
BIT[20]	-	保留，保持复位值
BIT[19]	SWIx	Line x 的软中断位（x = 19） 当该位为‘0’时，写‘1’将置位 EXTI_PR 中相应的挂起位。 如果在 EXTI_IMR 允许此 Line 产生该中断，则此时将产生一个中断请求。 通过对 EXTI_PR 的对应位写入‘1’，可以清除该位为’0’。
BIT[18]	-	保留，保持复位值
BIT[17:0]	SWIx	Line x 的软中断位（x = 0~17） 当该位为‘0’时，写‘1’将置位 EXTI_PR 中相应的挂起位。 如果在 EXTI_IMR 允许此 Line 产生该中断，则此时将产生一个中断请求。 通过对 EXTI_PR 的对应位写入‘1’，可以清除该位为’0’。

9.4.7 挂起寄存器（EXTI_PR）

EXTI_PR			地址偏移	0x14		复位值	0x XXXX XXXX	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	PIF22	PIF21	-	PIF19	-	PIF17	PIF16
R/W	-	rc_w1	rc_w1	-	rc_w1	-	rc_w1	rc_w1
复位值	-	x	x	-	x	-	x	x
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	PIF15	PIF14	PIF13	PIF12	PIF11	PIF10	PIF9	PIF8
R/W	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
复位值	x	x	x	x	x	x	x	x
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	PIF7	PIF6	PIF5	PIF4	PIF3	PIF2	PIF1	PIF0
R/W	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
复位值	x	x	x	x	x	x	x	x

BIT[31:23]	-	保留，保持复位值
BIT[22:21]	PIFx	Line x 挂起位 (x =21~22) 0：没有发生触发请求 1：发生了选择的触发请求 当在外部中断线上发生了选择的边沿事件，该位被置‘1’。在该位中写入‘1’可以清除它。
BIT[20]	-	保留，保持复位值
BIT[19]	PIFx	Line x 挂起位 (x =19) 0：没有发生触发请求 1：发生了选择的触发请求 当在外部中断线上发生了选择的边沿事件，该位被置‘1’。在该位中写入‘1’可以清除它。
BIT[18]	-	保留，保持复位值
BIT[17:0]	PIFx	Line x 挂起位 (x =0~17) 0：没有发生触发请求 1：发生了选择的触发请求 当在外部中断线上发生了选择的边沿事件，该位被置‘1’。在该位中写入‘1’可以清除它。

10 直接存储器访问控制器（DMA）

10.1 概述

DMA（直接存储器存取），用来提供外设和存储器之间或存储器和存储器之间的高速数据传输。无须 CPU 操作，通过 DAM 实现数据快速传输。这样可以节省 CPU 资源用于其他操作。

DMA 控制器包含 5 个通道，每个通道管理来自一个或多个外设发起的存储器访问请求；内建仲裁器来协调不同 DMA 请求的优先级。

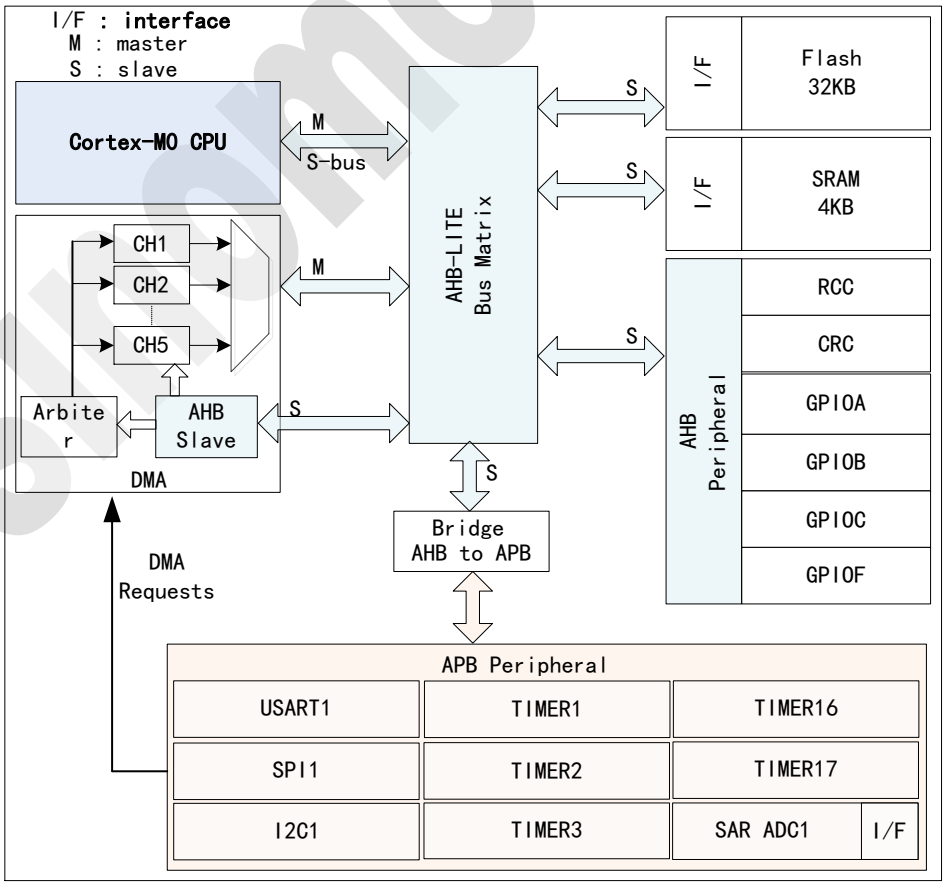
DMA 可以用于主要的外设：SPI、I2C、USART，定时器 TIMx（除了 TIM14）、ADC。

10.2 特性

- 5 个独立的可配置通道（请求）
- 每个通道都有专门的硬件 DMA 请求逻辑，同时可以由软件触发每个通道；
- 多个请求的优先级软件 4 级可设；相同优先级设置，硬件默认优先级（请求 1 优先于请求 2）
- 独立的源/目标数据宽度，模拟打包拆包；源/目标地址必须按数据传输宽度对齐
- DMA 控制器支持环形缓冲区的管理
- 每个通道都有 3 个事件标志（半传输、传输完成和传输出错），这 3 个事件标志通过或逻辑产生一个单独的中断请求。
- 支持存储器到存储区的传输
- 支持外设到存储器、存储器到外设、外设到外设的传输
- Flash、sram、APB 和 AHB 外设均可作为访问的源和目标
- 可编程的传输数量：最大为 65535

10.3 功能描述

DMA 模块框图



DMA 控制器执行 DMA 传输时，与 Cortex-M0 内核共享系统总线（System bus）。当 CPU 和 DMA 同时访问相同的目标（存储器或外设）时，DMA 请求会暂停 CPU 访问系统总线（若干总线周期），通过总线矩阵（bus matrix）的轮询调度机制，保证 CPU 获得至少一半的系统总线带宽（包括存储器和外设）。

10.3.1 DMA 处理

在发生一个事件后，外设向 DMA 控制器发送一个请求信号。DMA 控制器根据通道的优先权处理请求。当 DMA 控制器开始访问发出请求的外设时，DMA 控制器立即发送给外设一个应答信号。当外设得到应答信号时，立即释放它的请求信号。与此同时，DMA 控制器同时撤销应答信号。如果有更多的请求时，外设可以启动下一个事务。

总之 ，每次的 DMA 传输由 3 组操作组成：

- ✧ 从<外设数据寄存器>或者从当前<外设/存储器地址寄存器>指示的存储器地址加载数据；首次传输时的开始地址由 DMA_CPARx（外设基地址） 或 DMA_CMARx （存储器基地址）寄存器指定。
- ✧ 向<外设数据寄存器>或者从当前<外设/存储器地址寄存器>指示的存储器地址存储数据，首次传输时的开始地址由 DMA_CPARx（外设基地址） 或 DMA_CMARx （存储器基地址）寄存器指定。。
- ✧ 对 DMA_CNDTRx 寄存器执行一次递减操作，该寄存器存放着未完成的 DMA 操作的计数。

10.3.2 仲裁器

仲裁器根据优先级管理通道请求，并启动外设/存储器的访问序列。

优先级分为两阶管理：

软件：每个通道的优先级通过 DMA_CCRx 寄存器配置为 4 个等级：

- 最高优先级
- 高优先级
- 中优先级
- 低优先级

硬件：若 2 个请求具有相同的软件优先级，较低编号的通道具有更高的优先级。举例，通道 2 优先于通道 4。

10.3.3 DMA 通道

每个通道都支持一个固定地址的外设寄存器到存储器地址之间的 DMA 传输。传输的数据量可编程，最大可达 65535；每次传输后，保存待传输数量的计数寄存器执行一次递减操作。

可编程的传输数量

外设的传输数据量通过 DMA_CCRx 寄存器的 PSIZE 设置；

存储器的传输数据量通过 DMA_CCRx 寄存器的 MSIZE 设置。

指针递增

设置 DMA_CCRx 寄存器中的 PINC 和 MINC 位，设置外设和存储器的地址指针在每次 DMA 传输后进行自动递增。在递增模式下，地址递增的增量值取决于数据宽度设置（1、2 或 4）；首次传输时的开始地址由 DMA_CPARx（外设基地址）或 DMA_CMARx （存储器基地址）寄存器指定，传输过程中，这两个寄存器保持初始设置值，软件不能访问正在传输的地址（地址的递增在内部 buffer 地址寄存器改变）。

当通道配置为非循环传输模式时，传输结束（传输计数递减至 0）将不再接受 DMA 请求。若需要向 DMA_CNDTRx 寄存器重新载入传输数目值，需先关闭相应的 DMA 通道。

注：如果一个 DMA 通道关闭，DMA 寄存器不会复位。DMA 通道寄存器（DMA_CCRx, DMA_CPARx 和 DMA_CMARx0）保持已配置的初始值。

在循环模式下，最后一个传输结束后，DMA_CNDTRx 寄存器自动重载为初始编程的计数值。当前内部地址计数寄存器从 DMA_CPARx/DMA_CMARx 寄存器中装载基地址值。

通道配置流程

DMA 通道 x(x 为通道编号)配置流程：

1、DMA_CPARx 寄存器中设置外设寄存器地址。发生外设事件时，这个地址将是数据传输的源或目的地址。

2、DMA_CMARx 寄存器中设置存储器地址。发生外设事件时，传输的数据将从这个地址读出或写入。

3、DMA_CNDTRx 寄存器中写入需要传输的数目量，每次 DMA 传输后，该寄存器值递减。

4、DMA_CCRx 的 PL[1:0] 位中配置通道的优先级。

5、DMA_CCRx 中配置数据的传输方向、循环模式、外设和存储器的增量模式、外设和存储器的数据宽度和中断的设置。

6、DMA_CCRx 寄存器中的置位 ENABLE 位来启动该通道。

一旦启动了 DMA 通道，它既可响应连接到该道道上的外设的 DMA 请求。

当传输一半的数据后，半传输标志 (HTIF) 被置 1，当设置了允许半传输中断位 (HTIE) 时，将产生一个半传输完成的中断请求。当传输全部完成时，传输完成标志 (TCIF) 被置 1，当设置了允许传输完成中断位 (TCIE) 时，将产生

一个传输完成的中断请求。

循环模式

循环模式用于处理循环缓冲区和连续的数据传输（ 举例，ADC 的扫描模式 ）。置位 DMA_CCRx 寄存器中的 CIRC 位开启循环模式。当启动了循环模式，一组的数据传输完成时，计数寄存器将会自动地被重载成初始值（配置该通道时设置的值），DMA 请求将会持续响应。

存储器到存储器模式

在存储器到存储器模式下，DMA 通道的操作可以在没有外设请求的情况下进行。

设置 DMA_CCRx 寄存器中的 MEM2MEM 位，开启存储器到存储器模式，软件置位 DMA_CCRx 寄存器中的 EN 位，DMA 传输将立即开始。当 DMA_CNDTRx 寄存器递减为 0 时，DMA 传输结束。

存储器到存储器模式不能与循环模式同时使用。

10.3.4 数据宽度、数据对齐和数据大小端

当 PSIZE 和 MSIZE 不相同时，DMA 模块按照下表进行数据对齐。

可编程数据宽度和大小端操作（PINC=1,MINC=1）

源端宽度	目标端宽度	传输数目	源（地址/数据）	目标（地址/数据）	传输操作
8	8	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	0x0/B0 0x1/B1 0x2/B2 0x3/B3	1: 读 B0[7:0]@0x0 ， 写 B0[7:0]@0x0 2: 读 B1[7:0]@0x1 ， 写 B1[7:0]@0x1 3: 读 B2[7:0]@0x2 ， 写 B2[7:0]@0x2 4: 读 B3[7:0]@0x3 ， 写 B3[7:0]@0x3
8	16	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	0x0/00B0 0x2/00B1 0x4/00B2 0x6/00B3	1: 读 B0[7:0]@0x0 ， 写 00B0[15:0]@0x0 2: 读 B1[7:0]@0x1 ， 写 00B1[15:0]@0x2 3: 读 B2[7:0]@0x2 ， 写 00B2[15:0]@0x4 4: 读 B3[7:0]@0x3 ， 写 00B3[15:0]@0x6
8	32	4	0x0/B0 0x1/B1 0x2/B2 0x3/B3	0x0/000000B0 0x4/000000B1 0x8/000000B2 0xC/000000B3	1: 读 B0[7:0]@0x0 ， 写 000000B0[31:0]@0x0 2: 读 B1[7:0]@0x1 ， 写 000000B1[31:0]@0x4 3: 读 B2[7:0]@0x2 ， 写 000000B2[31:0]@0x8 4: 读 B3[7:0]@0x3 ， 写 000000B3[31:0]@0xC
16	8	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	0x0/B0 0x1/B2 0x2/B4 0x3/B6	1: 读 B1B0[15:0]@0x0 ， 写 B0[7:0]@0x0 2: 读 B3B2[15:0]@0x2 ， 写 B2[7:0]@0x1 3: 读 B5B4[15:0]@0x4 ， 写 B4[7:0]@0x2 4: 读 B7B6[15:0]@0x6 ， 写 B6[7:0]@0x3
16	16	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	1: 读 B1B0[15:0]@0x0 ， 写 B1B0[15:0]@0x0 2: 读 B3B2[15:0]@0x2 ， 写 B3B2[15:0]@0x2 3: 读 B5B4[15:0]@0x4 ， 写 B5B4[15:0]@0x4 4: 读 B7B6[15:0]@0x6 ， 写 B7B6[15:0]@0x6
16	32	4	0x0/B1B0 0x2/B3B2 0x4/B5B4 0x6/B7B6	0x0/0000B1B0 0x4/0000B3B2 0x8/0000B5B4 0xC/0000B7B6	1: 读 B1B0[15:0]@0x0 ， 写 0000B1B0[31:0]@0x0 2: 读 B3B2[15:0]@0x2 ， 写 0000B3B2[31:0]@0x4 3: 读 B5B4[15:0]@0x4 ， 写 0000B5B4[31:0]@0x8 4: 读 B7B6[15:0]@0x6 ， 写 0000B7B6[31:0]@0xC
32	8	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	0x0/B0 0x1/B4 0x2/B8 0x3/BC	1: 读 B3B2B1B0[31:0]@0x0 ， 写 B0[7:0]@0x0 2: 读 B7B6B5B4[31:0]@0x4 ， 写 B4[7:0]@0x1 3: 读 BBBAB9B8[31:0]@0x8 ， 写 B8[7:0]@0x2 4: 读 BFBEBDBC[31:0]@0xC ， 写 BC[7:0]@0x3
32	16	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	0x0/B1B0 0x2/B5B4 0x4/B9B8 0x6/BDBC	1: 读 B3B2B1B0[31:0]@0x0 ， 写 B1B0[15:0]@0x0 2: 读 B7B6B5B4[31:0]@0x4 ， 写 B5B4[15:0]@0x2 3: 读 BBBAB9B8[31:0]@0x8 ， 写 B9B8[15:0]@0x4 4: 读 BFBEBDBC[31:0]@0xC ， 写 BDBC[15:0]@0x6
32	32	4	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	0x0/B3B2B1B0 0x4/B7B6B5B4 0x8/BBBAB9B8 0xC/BFBEBDBC	1: 读 B3B2B1B0[31:0]@0x0 ， 写 B3B2B1B0[31:0]@0x0 2: 读 B7B6B5B4[31:0]@0x4 ， 写 B7B6B5B4[31:0]@0x4 3: 读 BBBAB9B8[31:0]@0x8 ， 写 BBBAB9B8[31:0]@0x8 4: 读 BFBEBDBC[31:0]@0xC ， 写 BFBEBDBC[31:0]@0xC

寻址一个不支持字节或半字写操作的 AHB 外设

当 DMA 开始一个 AHB 的字节或半字写操作，HWDATA[31:0]总线中，未被使用的位宽将被数据重复填充；因此当 DMA 以字节或半字写入不支持字节或半字操作的 AHB 从机外设时（当 HSIZE 不适用于外设），不会发生错误，DMA 会按照下面

举例写入 32 位 HWDATA 数据：

- ✧ 当 HSIZE=半字，写入半字“0xABCD”，DMA 将设置 HWDATA 总线为“0xABCDABCD”。
- ✧ 当 HSIZE=字节，写入字节“0xAB”，DMA 将设置 HWDATA 总线为“0xABABABAB”。

假设 AHB2APB 桥是一个 AHB 的 32 位从外设，不考虑 HSIZE 数据，它将按照下述方式把 AHB 的字节或半字操作转化为 32 位 AHB 操作：

- ✧ 一个 AHB 字节写操作，对地址 0x0（或 0x1、0x2 或 0x3）写数据“0xB0”，转换为 APB 字（32 位）写操作，对 0x0 写数据“0xB0B0B0B0”。
- ✧ 一个 AHB 半字写操作，对地址 0x0（或 0x2）写数据“0xB1B0”，转换为 APB 字（32 位）写操作，对 0x0 写数据“0xB1B0B1B0”。

举例，如果要写入 APB 备份寄存器（32 位地址对齐的 16 位寄存器），需要配置存储器数据源宽度（MSIZE）为“16 位”，外设目标数据宽度（PSIZE）为“32 位”。

10.3.5 错误管理

读写一个保留的地址区域，将会产生 DMA 传输错误。当在 DMA 读写操作时发生 DMA 传输错误时，硬件会自动地清除发生错误的通道使能位（通道配置寄存器（DMA_CCRx）的 EN 位），该通道操作被停止。此时，在 DMA_IFR 寄存器中对应该通道的传输错误中断标志位（TEIF）将被置位，如果在 DMA_CCRx 寄存器中设置了传输错误中断允许位，则将产生中断。

10.3.6 DMA 中断

每个 DMA 通道都可以在 DMA 传输过半、传输完成和传输错误时产生中断。通过设置寄存器的不同位来打开这些中断。

DMA 中断请求

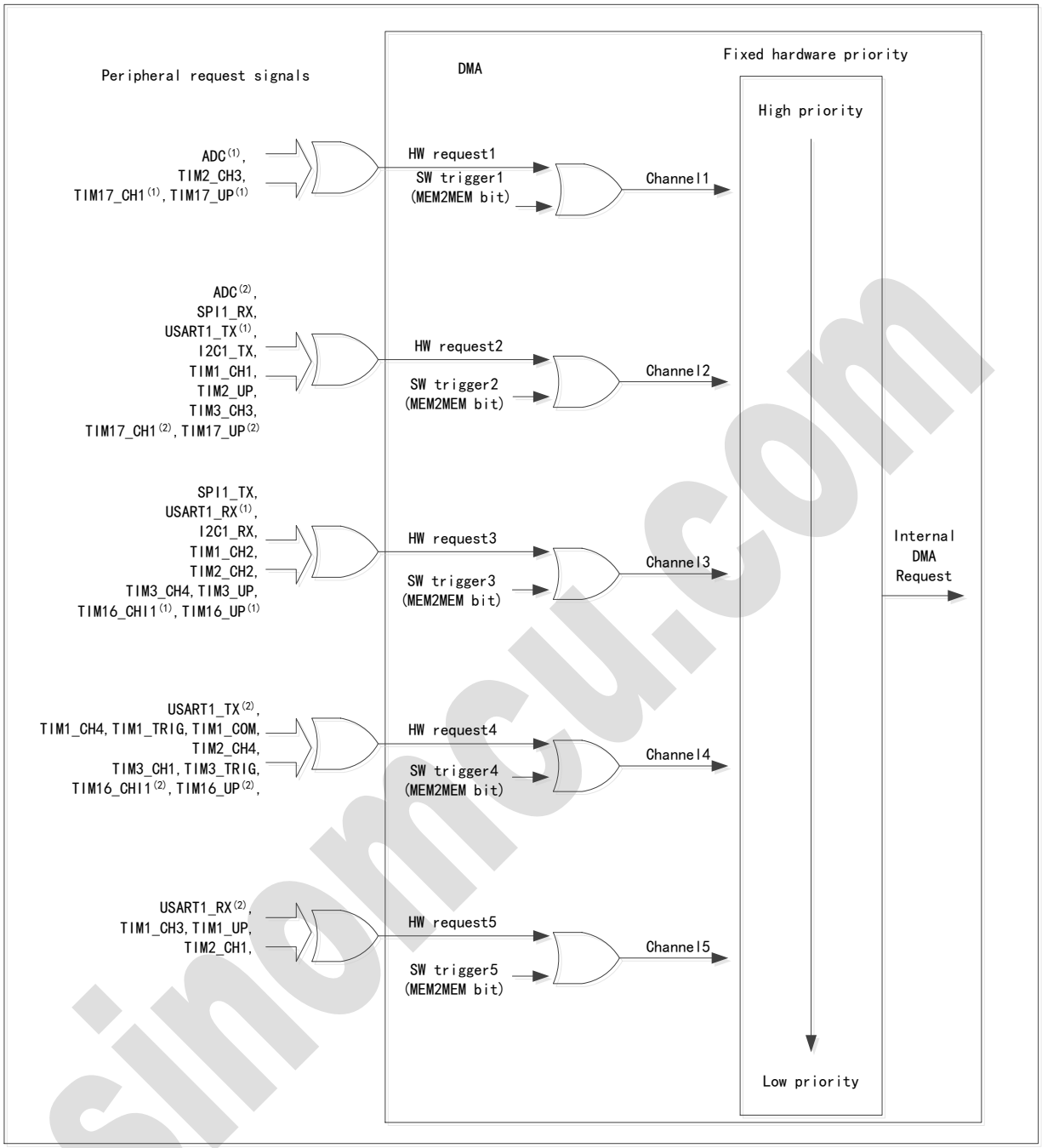
中断事件	事件标志位	中断使能控制位
传输过半	HTIF	HTIE
传输完成	TCIF	TCIE
传输错误	TEIF	TEIE

10.3.7 DMA 请求映射

来自外设（TIMx，ADC，DAC，SPI，I2C，和 USARTx）的硬件请求经过简单的逻辑或运算后给到 DMA，也就是说，一个 DMA 通道同一时刻只允许一个 DMA 请求。

外设的 DMA 请求，可以通过设置外设的对应寄存器的控制位，独立的开启和关闭。

DMA 请求映像



DMA 通道请求列表

外设	Channel 1	Channel 2	Channel 3	Channel 4	Channel 5
ADC	ADC(1)	ADC(2)			
SPI		SPI1_RX	SPI1_TX		
USART		USART1_TX(1)	USART1_RX(1)	USART1_TX(2)	USART1_RX(2)
I2C		I2C1_TX	I2C1_RX		
TIM1		TIM1_CH1	TIM1_CH2	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_CH3 TIM1_UP
TIM2	TIM2_CH3	TIM2_UP	TIM2_CH2	TIM2_CH4	TIM2_CH1
TIM3		TIM3_CH3	TIM3_CH4 TIM3_UP	TIM3_CH1 TIM3_TRIG	
TIM16			TIM16_CH1(1)	TIM16_CH1(2)	

			TIM16_UP(1)	TIM16_UP(2)	
TIM17	TIM17_CH1(1) TIM17_UP(1)	TIM17_CH1(2) TIM17_UP(2)			

注：(1) 只有在 SYSCFG_CFGR1 寄存器相应的重映象位清 0 时，该 DMA 请求映象到这个 DAM 通道。详情请参见 SYSCFG 配置寄存器 1 (SYSCFG_CFGR1)

注：(2) 只有在 SYSCFG_CFGR1 寄存器相应的重映象位置位时，该 DMA 请求映象到这个 DMA 通道。详情请参见 SYSCFG 配置寄存器 1 (SYSCFG_CFGR1)

10.4 相关寄存器

此外设寄存器仅支持字（32 位）访问。

10.4.1 寄存器汇总表

基址：0x4002 0000				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**DMA_ISR**	0x0000 0000	DMA 中断状态寄存器	10.4.2
0x04	**DMA_IFCR**	0x0000 0000	DMA 中断标志清除寄存器	10.4.3
0x08+0x14*(x-1)	**DMA_CCRx**	0x0000 0000	DMA 通道 x 配置寄存器	10.4.4
0x0C+0x14*(x-1)	**DMA_CNDTRx**	0x0000 0000	DMA 通道 x 传输数量寄存器	10.4.5
0x10+0x14*(x-1)	**DMA_CPARx**	0x0000 0000	DMA 通道 x 外设地址寄存器	10.4.6
0x14+0x14*(x-1)	**DMA_CMARx**	0x0000 0000	DMA 通道 x 存储器地址寄存器	10.4.7

注：上表中 x 为 DMA 通道号，x= 1~5。

10.4.2 DMA 中断状态寄存器（DMA_ISR）

DMA_ISR			地址偏移	0x00		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	TEIF5	HTIF5	TCIF5	GIF5
R/W	-	-	-	-	r	r	r	r
复位值	-	-	-	-	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	TEIF4	HTIF4	TCIF4	GIF4	TEIF3	HTIF3	TCIF3	GIF3
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0

BIT[31:20]	-	保留，保持复位值
BIT[19,15,11,7,3]	TEIFx	通道 x (x = 1~5) 传输错误标志 该位由硬件置位。由软件对 DMA_IFCR 寄存器的相应位写 1 清零。 0：通道 x 上无传输错误 (TE) 1：通道 x 上发生了传输错误 (TE)
BIT[18,14,10,6,2]	HTIFx	通道 x(x = 1~5) 半传输（half transfer）标志 该位由硬件置位。由软件对 DMA_IFCR 寄存器的相应位写 1 清零。 0：通道 x 上无传输一半 (HT) 的事件发生 1：通道 x 上发生了传输一半 (HT) 的事件
BIT[17,13,9,5,1]	TCIFx	通道 x (x = 1~5) 传输完成标志 该位由硬件置位。由软件对 DMA_IFCR 寄存器的相应位写 1 清零。 0：通道 x 上无传输完成 (TC) 的事件发生

		1：通道 x 上发生了传输完成(TC) 的事件
BIT[16,12,8,4,0]	GIFx	通道 x(x = 1~5) 全局中断标志 该位由硬件置位。由软件对 DMA_IFCR 寄存器的相应位写 1 清零。 0：在通道 x 上无 TE、HT 或 TC 事件发生 1：在通道 x 上发生了 TE、HT 或 TC 事件

10.4.3 DMA 中断标志清除寄存器（DMA_IFCR）

DMA_IFCR			地址偏移	0x04		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	CTEIF5	CHTIF5	CTCIF5	CGIF5
R/W	-	-	-	-	w	w	w	w
复位值	-	-	-	-	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CTEIF4	CHTIF4	CTCIF4	CGIF4	CTEIF3	CHTIF3	CTCIF3	CGIF3
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CTEIF2	CHTIF2	CTCIF2	CGIF2	CTEIF1	CHTIF1	CTCIF1	CGIF1
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0

BIT[31:20]	-	保留，保持复位值
BIT[19,15,11,7,3]	CTEIFx	通道 x (x = 1~5) 传输错误清除，该位由软件置 1 清除。 0：无效 1：在 DMA_ISR 寄存器中清除相应的 TEIF 标志
BIT[18,14,10,6,2]	CHTIFx	通道 x (x = 1~5) 传输一半标志清除，该位由软件置 1 清除。 0：无效 1：在 DMA_ISR 寄存器中清除相应的 HTIF 标志
BIT[17,13,9,5,1]	CTCIFx	通道 x (x = 1~5) 传输完成标志清除，该位由软件置 1 清除。 0：无效 1：在 DMA_ISR 寄存器中清除相应的 TCIF 标志
BIT[16,12,8,4,0]	CGIFx	通道 x (x = 1~5) 全局中断标志清除，该位由软件置 1 清除。 0：无效 1：在 DMA_ISR 寄存器中清除 GIF、TEIF、HTIF 和 TCIF 标志

10.4.4 DMA 通道 x 配置寄存器（DMA_CCRx）(x = 1~5)

DMA_CCRx			地址偏移	0x08+0x14*(x-1)		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	MEM2MEM	PL[1:0]		MSIZE[1:0]		PSIZE[1:0]	
R/W	-	rw	rw	rw	rw	rw	rw	rw
复位值	-	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:15]	-	保留，保持复位值
BIT[14]	MEM2MEM	存储器到存储器模式，软件置位和清除。 0：存储器到存储器模式关闭 1:存储器到存储器模式开启
BIT[13:12]	PL[1:0]	通道优先级，软件置位和清除。 00：低 01：中 10：高 11：最高
BIT[11:10]	MSIZE[1:0]	存储器数据宽度，软件置位和清除。 00：8 位 01：16 位 10：32 位 11：保留
BIT[9:8]	PSIZE[1:0]	外设数据宽度，软件置位和清除。 00：8 位 01：16 位 10：32 位 11：保留
BIT[7]	MINC	存储器地址递增模式，软件置位和清除。 0：存储器地址递增模式关闭 1：存储器地址递增模式开启
BIT[6]	PINC	外设地址递增模式，软件置位和清除。 0：外设地址递增模式关闭 1：外设地址递增模式开启
BIT[5]	CIRC	循环模式，软件置位和清除。 0：循环模式关闭 1：循环模式开启
BIT[4]	DIR	数据传输方向，软件置位和清除。 0：从外设读取 1：从存储器读取
BIT[3]	TEIE	传输错误中断使能，软件置位和清除。 0：禁止 TE 中断 1：允许 TE 中断
BIT[2]	HTIE	半传输中断使能，软件置位和清除。 0：禁止 HT 中断 1：允许 HT 中断
BIT[1]	TCIE	传输完成中断使能，软件置位和清除。 0：禁止 TC 中断 1：允许 TC 中断
BIT[0]	EN	通道使能，软件置位和清除。 0：通道关闭 1：通道开启

10.4.5 DMA 通道 x 传输数量寄存器（DMA_CNDTRx）（x=1~5）

DMA_CNDTRx			地址偏移	0x0C + 0x14*(x-1)		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	NDT[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw

复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	NDT[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	-	保留，保持复位值
BIT[15:0]	NDT[15:0]	传输数据的数量 传输数据的数量(0~65535)。这个寄存器只能在通道关闭(DMA_CCRx 的 EN=0) 时写入。一旦通道开启，该寄存器变为只读，其值为剩余待传输数据字节数。寄存器值在每次 DMA 传输后递减。 数据传输结束后，若循环模式关闭（DMA_CCRx 的 CIRC=0），寄存器的值保持为 0；若循环模式开启（DMA_CCRx 的 CIRC=1），寄存器的值将被自动重新加载为之前配置时的数值。 当寄存器的值为 0 时，无论通道是否开启，都不会发生任何数据传输。

10.4.6 DMA 通道 x 外设地址寄存器（DMA_CPARx）(x=1~5)

DMA_CPARx			地址偏移	0x10 + 0x14*(x-1)		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	PA[31:24]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	PA[23:16]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	PA[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	PA[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:0]	PA[31:0]	外设地址 外设数据寄存器的基地址，作为数据传输的源或目标地址。 当 PSIZE=‘01’(16 位)，忽略 PA[0]位。访问操作自动半字地址对齐。 当 PSIZE=‘10’(32 位)，忽略 PA[1:0]位。访问操作自动字地址对齐。

注：这个寄存器只能在通道关闭(DMA_CCRx 的 EN=0) 时写入。

10.4.7 DMA 通道 x 存储器地址寄存器（DMA_CMARx）(x = 1~5)

DMA_CMARx			地址偏移	0x14 + 0x14*(x-1)		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	MA[31:24]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	MA[23:16]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	MA[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

控制位	MA[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:0]	MA[31:0]	存储器地址 存储器地址，作为数据传输的源或目标地址。 当 MSIZE = ‘01’ (16 位)，忽略 MA[0]位。访问操作自动半字地址对齐。 当 MSIZE = ‘10’ (32 位)，忽略 MA[1:0]位。访问操作自动字地址对齐。

注：这个寄存器只能在通道关闭(DMA_CCRx 的 EN=0) 时写入。

11 循环冗余校验计算单元（CRC）

11.1 概述

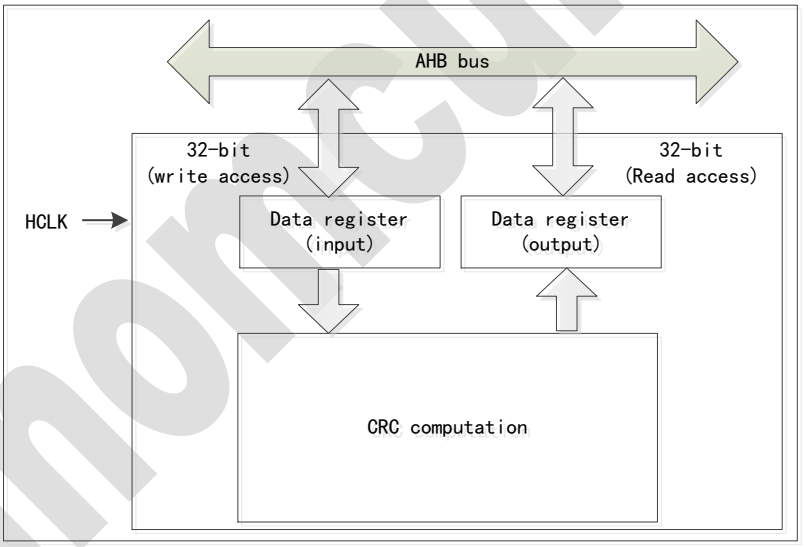
CRC（循环冗余校验）计算单元，根据特定的多项式，从一个 8 位、16 位或 32 位的数据字中产生 CRC 码。
CRC 在应用中主要来验证数据传输或存储的完整性。从功能安全标准考量，也提供了 flash 存储可靠性验证方法。
CRC 计算单元可以用于在运行期间计算软件签名，并与链接时生成的参考签名比对。

11.2 特性

- ✧ 支持 CRC-32 多项式：0x4C11DB7(以太网)
- ✧ $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^{8} + X^{7} + X^{5} + X^{4} + X^{2} + X + 1$
- ✧ 支持 8 位、16 位和 32 位数据
- ✧ CRC 初值可编程
- ✧ 输入/输出 32 位数据寄存器为同一个寄存器
- ✧ 输入缓存可以避免计算过程总线停顿
- ✧ 32 位数据计算周期：4 AHB clock（HCLK）
- ✧ 通用 8 位寄存器（可用于临时存储）
- ✧ 输入/输出数据取反可选

11.3 功能描述

CRC 模块框图



11.3.1 CRC 操作

数据寄存器

CRC 计算单元中包含一个 32 位可读/写的数据寄存器（CRC_DR）。它用来输入新的计算数据（写操作），也用来输出上次计算的结果（读操作）。

每次对该寄存器的写操作都会使得 CRC 计算单元将所写入的数据和上次计算得到的数据进行综合计算，并得到一个新的计算结果。

数据位宽和计算时间

CRC 计算单元根据写入数据的格式不同来决定是整字计算还是一个字节一个字节计算。

CRC_DR 寄存器可以按字（word）访问，也可以按半字（右对齐）或者字节（右对齐）的方式访问。其他寄存器仅支持 32-bit 位宽访问。

不同数据宽度计算持续时间：

- ✧ 32 位需要 4 个 AHB 时钟周期
- ✧ 16 位需要 2 个 AHB 时钟周期
- ✧ 8 位需要 1 个 AHB 时钟周期

连续数据写入

内建一个输入缓冲区，可以实现连续写入新数据，而无需等待当前计算持续时间结束。

数据位宽动态调整

数据位宽可以动态调整，保证计算数据块适配合适的最小的计算写入次数。 举例，一个 5 字节的 CRC 计算，会动态调整为一个字（word）计算和一个 byte 计算的组合。

输入/输出位序颠倒

输入数据高低位序可以颠倒，以适应各种不同的大小端体系。 数据位序颠倒操作可以按 8 位、16 位和 32 位进行，这个功能由 CRC_CR 寄存器中的 REV_IN[1:0] 位来选择设置。

举例 1 ：

输入数据 0x1A2B3C4D 用于 CRC 计算 ：

按字节颠倒就是：0x58D43CB2

按半字颠倒就是：0xD458B23C

按整字颠倒就是：0xB23CD458。

输出数据也可以通过设置 CRC_CR 寄存器中的 REV_OUT 位来进行位序颠倒。 该操作是按位进行颠倒的。

举例 2：

输出数据 0x11223344 转换过来就是 0x22CC4488。

复位

可以通过 CRC_CR 寄存器中的 RESET 控制位将 CRC 计算器的初值初始化为一个特定的值，这个值可以由程序指定，默认为 0xFFFFFFFF.

这个初始值可以通过 CRC_INIT 寄存器来指定。在初始化时自动更新到 CRC_DR 寄存器中。

CRC_IDR 寄存器则用来保存 CRC 计算的临时结果。这个寄存器不会受 CRC_CR 寄存器中的 RESET 位影响。

11.3.2 CRC 多项式

本芯片仅支持 CRC-32 多项式：0x4C11DB7(以太网)

$X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$

通过只读寄存器 CRC_POL，可以读到多项式值（0x4C11DB7）。

11.4 相关寄存器

11.4.1 寄存器汇总表

基址：0x4002 3000				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	CRC_DR	0xFFFF FFFF	数据寄存器	11.4.2
0x04	CRC_IDR	0x0000 0000	独立数据寄存器	11.4.3
0x08	CRC_CR	0x0000 0000	控制寄存器	11.4.4
0x0C	Res.	-	-	
0x10	CRC_INIT	0xFFFF FFFF	CRC 初值寄存器	11.4.5
0x14	CRC_POL	0x04C11DB7	CRC 多项式寄存器	11.4.6

11.4.1 数据寄存器（CRC_DR）

CRC_DR			地址偏移	0x00		复位值	0xFFFF FFFF	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	DR[31:24]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	1	1	1	1	1	1	1
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	DR[23:16]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	1	1	1	1	1	1	1
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	DR[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	1	1	1	1	1	1	1
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	DR[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	1	1	1	1	1	1	1

BIT[31:0]	DR[31:0]	数据寄存器位 该寄存器用于写入待计算的新数据，直接将其写入即可。 读取该寄存器得到的则是上次 CRC 计算的结果。 如果数据不足 32 位，右对齐读/写正确数据。

11.4.2 独立数据寄存器（CRC_IDR）

CRC_IDR			地址偏移	0x04		复位值	0x0000 0000
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	IDR[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:8]	-	保留，保持复位值
BIT[7:0]	IDR[7:0]	通用 8 位数据寄存器位 这些位用于一个字节的临时存储。 该寄存器不受 CRC_CR 寄存器中的 RESET 位所引发的复位动作的影响。

11.4.3 控制寄存器（CRC_CR）

CRC_DIR			地址偏移	0x08		复位值	0x0000 0000
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	REV_OUT	REV_IN[1:0]		-	-	-	-	RESET
R/W	rw	rw	rw	-	-	-	-	rs
复位值	0	0	0	-	-	-	-	0

BIT[31:8]	-	保留，保持复位值
BIT[7]	REV_OUT	输出数据位序颠倒，该位控制输出数据的位序颠倒 0：不颠倒 1：颠倒
BIT[6:5]	REV_IN[1:0]	输入数据位序颠倒，该位控制输入数据的位序颠倒 00：不颠倒

		01：按字节为单位颠倒 10：按半字为单位颠倒 11：按字为单位颠倒
BIT[4:1]	-	保留，保持复位值
BIT[0]	RESET	复位控制，软件置位，硬件自动清零 此位用来复位整个 CRC 计算单元，并将 CRC_INIT 寄存器中的值更新到数据寄存器中。

11.4.4 CRC 初值寄存器（CRC_INIT）

CRC_INIT			地址偏移	0x10		复位值	0xFFFF FFFF	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	CRC_INIT[31:24]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	1	1	1	1	1	1	1
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	CRC_INIT[23:16]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	1	1	1	1	1	1	1
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CRC_INIT[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	1	1	1	1	1	1	1
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CRC_INIT[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	1	1	1	1	1	1	1

BIT[31:0]	CRC_INIT[31:0]	CRC 初值 该寄存器用来设置 CRC 的初值。

11.4.5 CRC 多项式寄存器（CRC_POL）

CRC_POL			地址偏移	0x10		复位值	0xFFFF FFFF	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	POL[31:24]							
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	1	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	POL[23:16]							
R/W	r	r	r	r	r	r	r	r
复位值	1	1	0	0	0	0	0	1
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	POL[15:8]							
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	1	1	1	0	1
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	POL[7:0]							
R/W	r	r	r	r	r	r	r	r
复位值	1	0	1	1	0	1	1	1

BIT[31:0]	POL[31:0]	CRC 多项式，只读 该寄存器只读，默认值固定为 0x04C11DB7。

12 模/数转换器（ADC）

12.1 概述

本产品内嵌 1 个 12 位的逐次逼近型模拟/数字转换器(SAR ADC)，多达 10 个外部通道和 6 个内部通道（温度传感器、电压参考、 VBAT 电压测量和 3 路运算放大器输出），可以实现单次、连续、扫描或间断模式转换。数据结果可选择左对齐或右对齐存储于 16 位数据寄存器中。

ADC 可以使用 DMA 操作。

模拟看门狗功能允许非常精准地监视一路、多路或所有选中的通道，当被监视的信号超出预置的阀值时，将产生中断。

由标准定时器(TIMx)和高级控制定时器(TIM1)产生的事件，可以分别内部级联到 ADC 的触发，应用程序能使 AD 转换与时钟同步。

12.2 特性

✧ 高性能

- 12-bit/10-bit/8-bit/6-bit 分辨率可选
- ADC 转换时间 ：1.125 μs @12-bit 分辨率 ，1 μs @10-bit 分辨率 ， 在更低的分辨率下可达到更快的转换时间。
- 自校准
- 采样时间可编程
- 数据对齐
- 支持 DMA

✧ 低功耗

- 应用程序可以降低 PCLK 频率，以实现低功耗，同时保持最佳 ADC 性能。
- 等待模式：在 PLCK 低速应用中，防止 ADC 超限。

✧ 模拟输入通道

- 10 个外部模拟输入通道
- 1 个内部温度传感器通道（V_{SENSE}）
- 1 个内部电压参考通道（V_{REFINT}）
- 1 个 VBAT 供电输入检测通道
- 3 个运算放大器输出测量通道

✧ 多种启动方式

- 软件启动
- 硬件触发（TIM1、TIM2、TIM3）

✧ 转换模式

- 可转换单一通道或扫描一序列通道
- 单次模式，每次触发转换选定所有通道
- 连续模式，连续转换选定的所有通道
- 间断模式

✧ 中断产生：采样结束、转换完成、序列转换完成、模拟看门狗、溢出事件

✧ 模拟看门狗

✧ ADC 供电：2.4v~5.5v

✧ ADC 输入范围：VSSA ≤ VIN ≤ VDDA

12.3 ADC 引脚及内部信号

ADC 内部信号

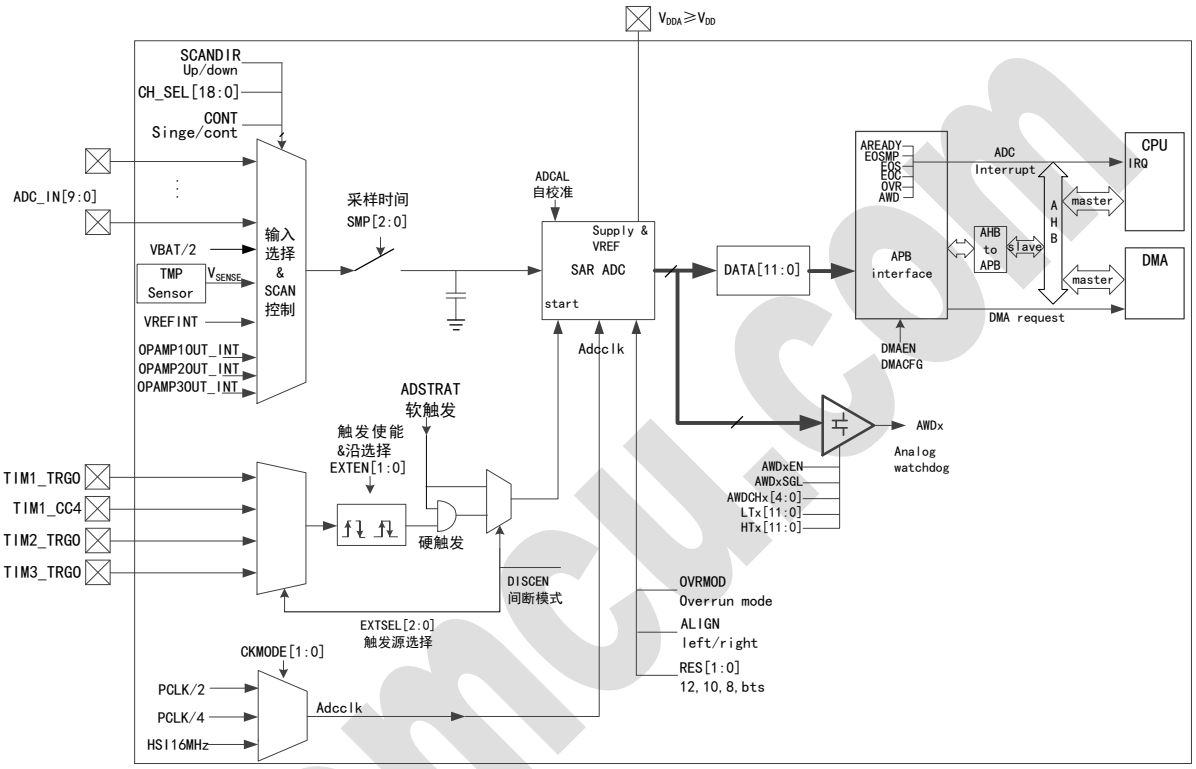
内部信号名称	信号类型	说明
TRGx	输入，数字信号	ADC 转换触发源
V_{SENSE}	输入，模拟信号	内部温度传感器输出电压
V_{REFINT}	输入，模拟信号	内部参考电压的输出
$V_{BAT}/2$	输入，模拟信号	VBAT 引脚输入电压/2
OPAMPxOUT_INT	输入，模拟信号	运算放大器输出电压

ADC 引脚

名称	信号类型	注释
V_{DDA}	输入，模拟供电电源	模拟供电电源，ADC 参考电压的正极，$V_{DDA} \geq V_{DD}$
V_{SSA}	输入，模拟电源地	模拟地，$V_{SSA} = V_{SS}$
ADC_IN[9:0]	输入，模拟信号	10 路模拟输入

12.4 功能描述

ADC 模块框图



12.4.1 ADC 校准（ADCAL）

ADC 支持校准功能，在此过程，ADC 计算一个校准因子，该因子内部作用于 ADC，在 ADC 断电后失效。在 ADC 校准期间直到校准完成，ADC 禁止使用。

在 A/D 转换前应执行校准操作，用于消除各芯片 AD 转换的偏移误差（offset error）。

设置 ADCAL=1，启动校准，且只能在 ADC 禁用（ADEN=0）时启动校准。校准期间 ADCAL 保持为 1，当完成校准后，ADCAL 位由硬件清零。此时，可以从 ADC_DR 读取校准因子（bit6~0）。

当 ADC 禁用（ADEN=0）时，内部模拟校准因子保留。当 ADC 工作条件发生变化时（VDD 变化为 ADC offset 误差的主要原因，温度变化次之），建议重新运行一个校准循环。

每次 ADC 掉电（例如，当产品进入待机模式（standby）时），校准因子会丢失。

校准软件操作流程：

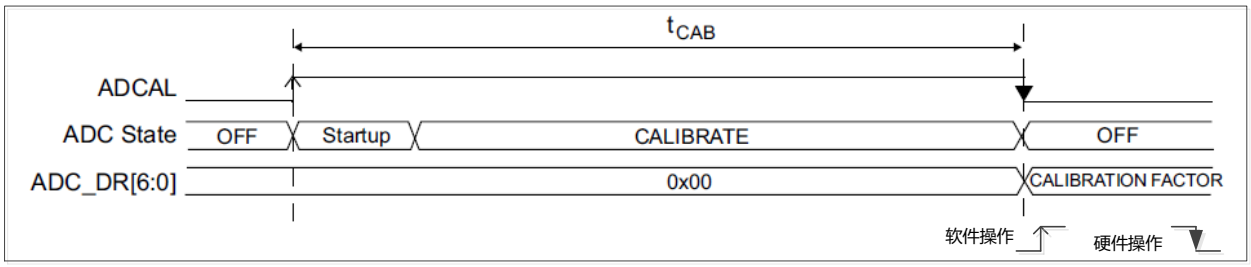
1、 确保 ADEN=0,DMAEN=0

2、 设置 ADCAL=1

3、 等待直到 ADCAL=0

4、 校准因子可从 ADC_DR 寄存器读取（bit6~0）

ADC 校准



12.4.2 ADC 开关控制（ADEN，ADDIS，ADRDY）

在 MCU 上电时，ADC 被禁用并处于掉电模式（ADEN=0）。

如下图所示，ADC 开始精确转换前，需要一个稳定时间 t_{STAB}。

两个控制位用于开启或关闭 ADC:

✧ 设置 ADEN=1 开启 ADC。当 ADC 模块准备好时，ADRDY 标志置 1。

✧ 设置 ADDIS=1 来关闭 ADC，并使 ADC 处于断电模式。当 ADC 模块全关断后，硬件自动清除 ADEN 和 ADDIS 位。

ADC 转换既可有软件设置 ADSTART=1 启动，也可以由外部触发事件启动（触发启动开启）。

使能 ADC 流程如下：

1、 清零 ADRDY（ADC_ISR 寄存器），写‘1’清零。

2、 ADC_CR 寄存器中设置 ADEN=1。

3、 等待 ADRDY=1（ADC_ISR 寄存器），并继续写入 ADEN=1（ADRDY 在 ADC 启动时间之后被置位）。如果置位 ADRDYIE 位（ADC_IER 寄存器）启用中断，这可以通过中断来处理。

关闭 ADC 流程如下：

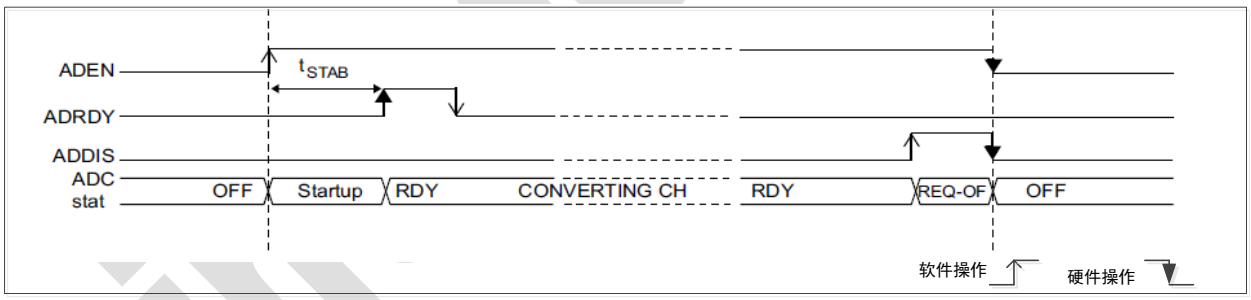
1、 检查 ADSTART=0（ADC_CR 寄存器）以确保 ADC 不在转换中。若需要，可对 ADSTP（ADC_CR 寄存器）写‘1’，停止正在进行 ADC 转换，并等待 ADSTP 被清 0。

2、 设置 ADC_CR 寄存器中的 ADDIS=1。

3、 若应用需要，等待直至 ADCEN=0（ADC_CR 寄存器），表示 ADC 模块已完全关闭（一旦 ADEN=0，ADDIS 自动清 0）。

注：当 ADCAL=1 时或 ADCAL 被硬件清零后的 4 个 ADC 时钟周期期间，ADEN 不能被置位。

开启/关闭 ADC

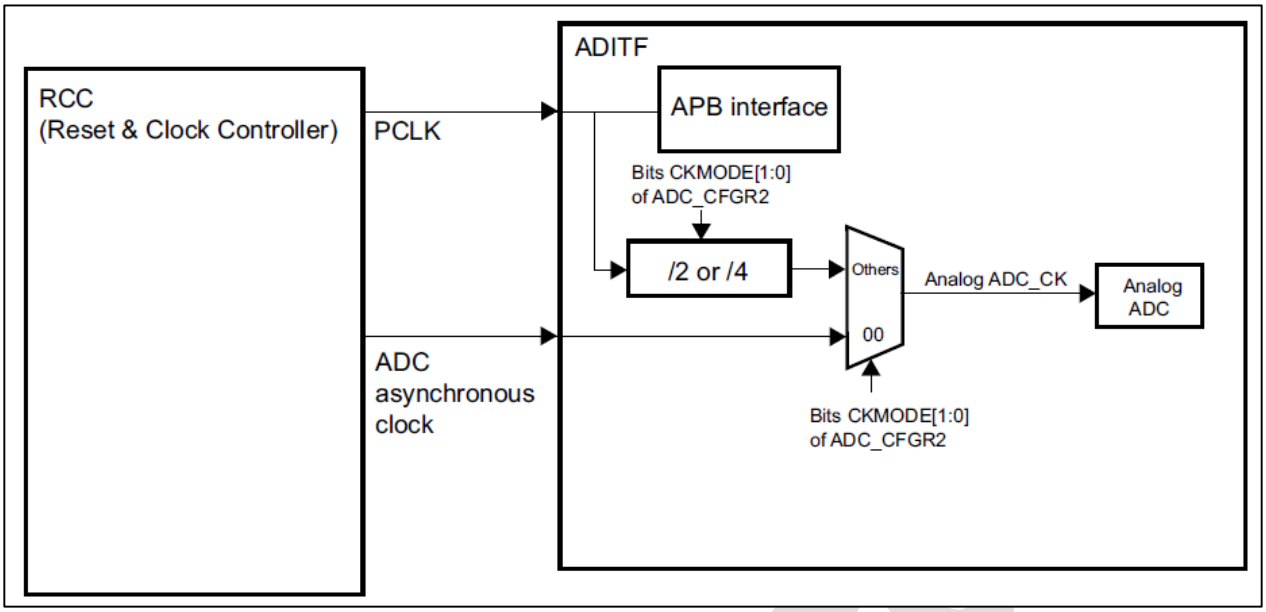


注：在自动关闭模式(AUTOFF=1)，ADC 电源的开/关阶段由硬件自动执行，ADRDY 标志不会置起。

12.4.3 ADC 时钟（CKMODE）

ADC 具有双时钟架构，ADC 时钟可以选择异步时钟，独立于 APB 时钟（PCLK）。

ADC 时钟框图



模拟 ADC 可以选择 2 种不同的时钟源（时钟使能参见 RCC 章节）：

✧ a）ADC 异步时钟（16M HIRC），设置 CKMODE[1:0]= ‘00’ 选择（ADC_CFGR2 寄存器）

✧ b）APB 时钟/2 或/4，由 ADC 总线接口派生得到，设置 CKMODE[1:0]不为 ‘00’ 选择（ADC_CFGR2 寄存器）

选项 a）优点在于，无论 APB 时钟如何选择，ADC 都可以达到最大 ADC 时钟频率。

选项 b）有点在于，避免时钟预的重同步，当 ADC 由定时器触发，且应用要求精确触发 ADC 时，对避免不确定性是非常有用的（否则，两个时钟域间的重同步将增加触发瞬间的不确定性）。

触发和开始转换之间的延迟

ADC 时钟源	CKMODE[1:0]	触发事件和开始转换之间的延迟
16MHz HIRC 时钟	00	延迟的不确定性（抖动）
PCLK / 2	01	延迟是确定的（无抖动）等于 2.75 个 ADC 时钟周期
PCLK / 4	10	延迟是确定的（无抖动）等于 2.625 个 ADC 时钟周期

12.4.4 ADC 配置

ADC_CR 寄存器中的 ADCAL 和 ADEN 位，软件必须在 ADC 禁止（ADEN 必须为 0）的情况下改写。

ADC_CR 寄存器中的 ADSTART 和 ADDIS 位，软件必须在 ADC 开启且无关闭请求挂起（即 ADEN=1 且 ADDIS=0）的情况下改写。

对于其他的控制位 （在 ADC_IER、ADC_CFGR1/2、ADC_SMPR、ADC_TR、ADC_CHSELR 和 ADC_CCR 寄存器中），软件必须在 ADC 开启（ADEN=1）且无进行的转换（ADSTART = 0）的情况下，才能进行改写。

ADC_CR 寄存器中的 ADSTP 位，软件必须在 ADC 开启且无关闭请求挂起（ADSTART=1 和 ADDIS=0）的情况下改写。

注：没有硬件保护机制，以防止上述规则禁止的写操作。若发生了禁止的写操作，ADC 将进入不确定状态。要恢复至正确操作，ADC 必须被关闭（ADEN=0,且 ADC_CR 寄存器清 0）。

12.4.5 通道选择（CHSEL，SCANDIR）

ADC 共有 16 个模拟通道：

✧ 10 个外部模拟输入，从 GPIO 引脚引入（ADC_IN0~9）

✧ 6 个内部模拟输入（温度传感器、内部参考电压、V_{BAT}通道、3 个运算放大器输出）

ADC 可以转换一个单一通道或自动扫描一个序列通道。

被转换的通道序列必须在通道选择寄存器 ADC_CHSELR 中编程选择：每个模拟输入通道有专门的一位选择位(CHSEL0~CHSEL21，其中 CHSEL10~CHSEL15 位保留)。

ADC 扫描的通道顺序由 ADC_CFGR1 中 SCANDIR 位的配置来决定：

✧ SCANDIR=0：正向扫描，从通道 0 到通道 21

✧ SCANDIR=1：反向扫描：从通道 21 到通道 0

温度传感器 ， V_{REFINT} 和 V_{BAT} 内部通道

温度传感连接到 ADC_IN16 通道，内部参考电压 V_{REFINT} 连接到 ADC1_IN17 通道。V_{BAT} 连接到 ADC1_IN18 通道。

12.4.6 可编程采样时间（SMP）

在启动数模转换之前，ADC 需要在被测电压源和内嵌采样电容间建立直接连接。则必须有足够长的采样时间，以便输入电压源对采样保持电容充电并达到输入电压的水平。

根据输入电压源的输入阻抗，编程采样时间进而调整转换速度。

输入电压采样的时钟周期个数，通过 ADC_SMPR 寄存器中的 SMP[2:0] 位来进行修改。

可编程采样时间对所有通道都通用。如有应用需求，则可在每次转换之间，软件改变和调整采样时间。

总的转换时间计算如下：

t_{CONV} = 采样时间 + 14.5 x ADC 时钟周期

举例：

当 ADC_CLK=16 MHz 且采样时间为 3.5ADC 时钟周期，

t_{CONV} = 3.5+14.5 = 18 adc clk = 1.125us

采样结束，置位 EOSMP 标志位，表明采样阶段结束。

12.4.7 单次转换模式（CONT=0）

单次转换模式下，ADC 执行一次序列转换，转换所有被选的通道。设置 ADC_CFGR1 寄存器中的 CONT=0 时，ADC 为单次转换模式。

ADC 转换可由下述两种方法启动：

- ✧ 在 ADC_CR 寄存器中，置位 ADSTART 位
- ✧ 硬件触发事件

在序列通道的转换中，每次转换完成后：

- ✧ 转换的数据结果存放到 16 位寄存器 ADC_DR 中
- ✧ EOC（转换结束标志）标志置位
- ✧ 若 EOCIE 位置位，则产生一个中断

通道序列转换完成后：

- ✧ EOSEQ（序列结束）标志置位
- ✧ 若 EOSIE 位置位，则产生一个中断

转换结束后，ADC 停止直到新的触发事件或 ADSTART 重新置位。

注：若转换单一通道，则可编程一个长度为 1 的一个转换序列。

12.4.8 连续转换模式（CONT=1）

在连续转换模式中，当软件或硬件触发事件产生，ADC 执行一个序列转换。转换所有的通道一次且自动重新开始执行相同的序列转换。设置寄存器 ADC_CFGR1 中的 CONT=1 时，ADC 选择为连续转换模式。

ADC 转换可由下述两种方法启动：

- ✧ 在 ADC_CR 寄存器中，置位 ADSTART 位
- ✧ 硬件触发事件

在序列通道的转换中，每次转换完成后：

- ✧ 转换的数据结果存放到 16 位寄存器 ADC_DR 中
- ✧ EOC（转换结束标志）标志置位
- ✧ 若 EOCIE 位置位，则产生一个中断

通道序列转换完成后：

- ✧ EOSEQ（序列结束）标志置位
- ✧ 若 EOSEQIE 位置位，则产生一个中断

一次序列转换结束后，ADC 立即重新转换相同的序列通道。

注 1：若转换单一通道，则可编程一个长度为 1 的一个转换序列。

注 2：不可能同时启用不连续模式和连续模式，禁止同时设置 DISCEN=1 和 CONT=1 位。

12.4.9 启动转换（ADSTART）

软件设置 ADSTART=1 启动 ADC 转换。

ADSTART 被置位，则转换：

- ✧ 当 EXTEN=0x0（软件触发）时，立即开始
- ✧ 当 EXTEN ≠ 0x0 时，在下一个所选择的有效沿硬件触发

ADSTART 位也用于表明目前 ADC 转换操作是否正在进行。当 ADSTART=0，表明 ADC 处于空闲时，可重新配置 ADC。

ADSTART 位可由硬件清 0：

- ✧ 单次模式+软件触发（CONT=0，EXTEN=00）
- - 在序列转换结束后（EOSEQ=1）

✧ 断续模式+软件触发（CONT=0，DISCEN=1，EXTEN=00）

- 在每次转换结束（EOC=1）

✧ 在所有情况下（CONT=x，EXTEN=xx）

- 在软件调用并执行 ADSTP 流程后（参考 12.4.11 停止当前转换 章节）

注 1：在连续模式（CONT=1）下，ADSTART 位不能由 EOSEQ 引发的硬件清除，其原因是自动重新开始序列转换。

注 2：单次转换模式 选择硬件触发（CONT=0 且 EXTEN=01），则当 EOSEQ 标志置位后，ADSTART 不会被硬件清 0 。这就避免了需要软件重新置位 ADSTART 位并保证下一次硬件触发事件不会错过。

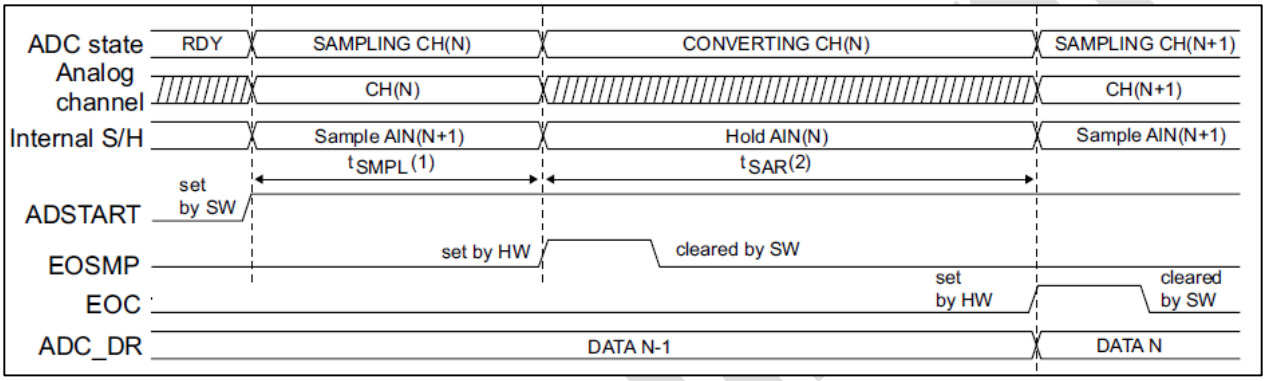
12.4.10 时序

总转换时间为配置的采样时间加上逐次逼近时间（根据数据分辨率）的总和。

$t_{ADC} = t_{SMPL} + t_{SAR} = [\ 3.5\ |_{min} + 14.5\ |_{12bit}\ ] \times t_{ADC\_CLK}$

$t_{ADC} = t_{SMPL} + t_{SAR} = 218.75\ ns\ |_{min} + 906.125ns\ |_{12bit} = 1.125\ \mu s\ |_{min}\ (for\ f_{ADC\_CLK} = 16MHz)$

AD 转换时间

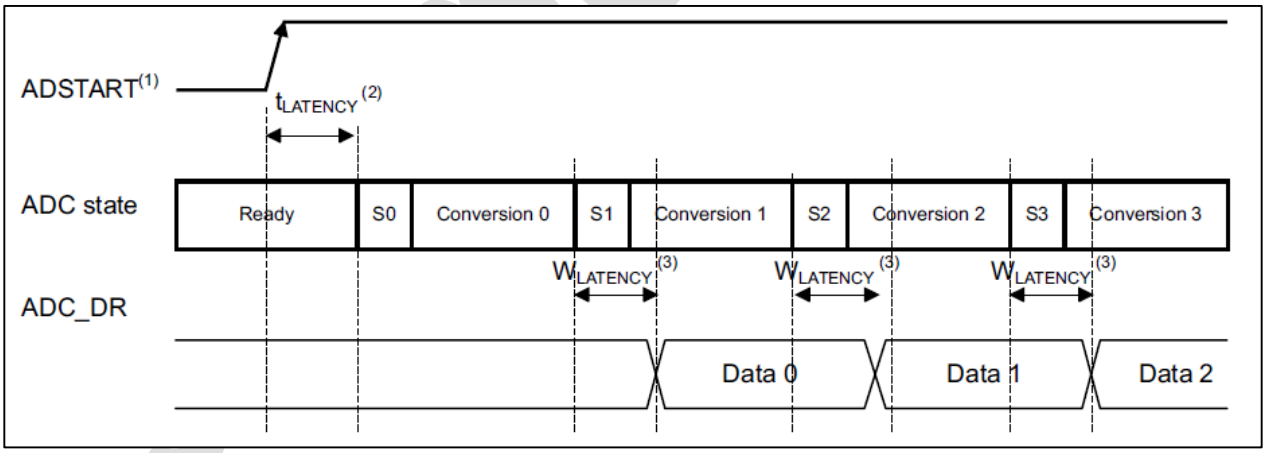


注：

（1）tSMPL 取决于 SMP[2:0]

（2）tSAR 取决于 RES[2:0]。

AD 转换时序



注 1：EXTEN=00 或 EXTEN≠00

注 2：触发延迟，详见数据手册

注 3：AD_CR 寄存器写延迟，详见数据手册

12.4.11 停止当前转换（ADSTP）

软件设置 ADC_CR 寄存器中的 ADSTP=1 可以停上当前正在进行的转换。

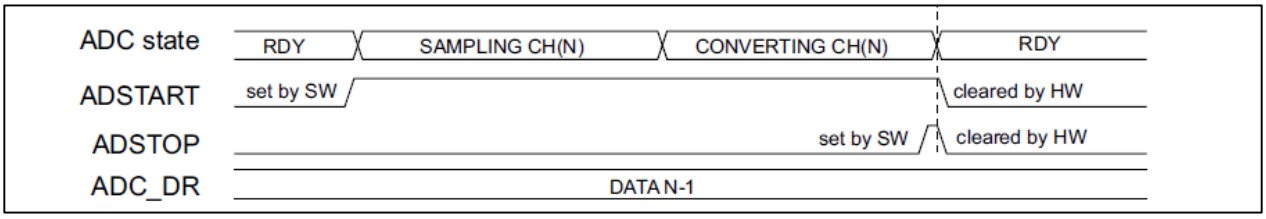
这会复位 ADC 的操作并让 ADC 进入空闲状态，为下次转换作好准备。

当 ADSTP 由软件置 1，任何当前进行的转换中止且转换结果丢弃（ADC_DR 寄存器在当前的转换不进行更新 ）。

扫描序列也被中止并复位（ 即重新启动 ADC 时会用新的序列进行转换 ）。

一旦结束该过程，ADSTP 和 ADSTART 位都由硬件清 0。软件必须等到 ADSTART=0 才开始新的转换。

停止当前的转换



12.5 外部触发转换和触发极性（EXTSEL，EXTEN）

一次转换或一个序列的转换可由软件或外部事件（ 例如：定时器捕获 ）触发。若 EXTEN[1:0] ≠ ‘0b00’，则外部事件根据所选择极性可以用于触发转换。当软件设置 ADSTART=1 时，触发选择生效。

当正在进行 ADC 转换时，任何硬件触发都会被忽略。

当 ADSTART=0 时，任何硬件触发都会忽略。

配置触发极性

EXTEN[1:0]	源
00	触发检测禁止
01	在上升沿时检测
10	在下降沿时检测
11	在上升沿及下降沿都检测

注：只有 ADC 不在转换时（ADSTART=0），才允许修改外部触发极性。

EXTSEL[2:0] 控制位用于选择可触发转换的事件（8 个事件可选）。

软件触发事件可由设置 ADC_CR 寄存器中的 ADSTART 位来产生。

外部触发源

EXTSEL[2:0]	名称	源
000	TRG0	TIM1_TRGO
001	TRG1	TIM1_CC4
010	TRG2	TIM2_TRGO
011	TRG3	TIM3_TRGO
100	TRG4	Res.
101	TRG5	Res.
110	TRG6	Res.
111	TRG7	Res.

注：只有 ADC 不在转换时（ADSTART=0），才允许修改外部触发源选择。

12.5.1 间断模式（DISCEN）

该模式由置位 ADC_CFGR1 寄存器中的 DISCEN 位来开启。

在这个模式(DISCEN=1) 下，序列中定义的的每个转换都需要一个硬件或软件的触发事件去启动。相反，DISCEN=0 时，一个硬件或软件的触发事件就可以启动定义在一个序列中的所有转换。

举例：

✧ DISCEN=1，需要转换的通道为：0，3，7，10

- 1st 触发：通道 0 被转换且一个 EOC 事件产生
- 2nd 触发：通道 3 被转换且一个 EOC 事件产生
- 3rd 触发：通道 7 被转换且一个 EOC 事件产生
- 4th 触发：通道 10 被转换且产生 EOC 和 EOSEQ 事件
- 5th 触发：通道 0 被转换且一个 EOC 事件产生
- 6th 触发：通道 3 被转换且一个 EOC 事件产生

✧ DISCEN=0，需要转换的通道为： 0，3，7，10

- 1st 触发：整个完整的序列转换：依次为通道 0，3，7 和 10。每次转换产生一个 EOC 事件，到最后一通道还产生一个 EOSEQ 事件。
- 任何触发事件都会重新开始完整的序列转换。

注 1：不可能同时启用不连续模式和连续模式:禁止同时设置 DISCEN=1 和 CONT=1 位。

12.5.2 可编程分辨率（RES）-快速转换模式

用降低 ADC 分辨率来获取更快的转换时间（t_{SAR}）是可行的。

分辨率可通过设置 ADC_ CFGR1 寄存器中的 RES[1:0]来配置，12/10/8/6 位模式可选。当应用不需要高数据精度时，可以通过降低分辨率来加快转换时间。

注：RES[1:0]位必须只在 ADEN 复位时（ADEN=0）修改。

转换结果保持为 12 位宽度，未使用的低位补 0。

分辨率与转换时间

RES[1:0] （位）	t_{SAR} （ADC clk）	t_{SAR}@16MHz （ns）	t_{SMPL}（min） （ADC clk）	t_{COV}(min)=t_{SMPL}+ t_{SAR} （ADC clk）	t_{COV} （ns）
12	14.5	906.25 ns	3.5	18	1125
10	12.5	781.125 ns	3.5	16	1000
8	10.5	656.25 ns	3.5	14	875
6	8.5	531.25 ns	3.5	12	750

12.5.3 转换结束，采样阶段结束（EOC，EOSMP 标志）

一旦在 ADC_DR 寄存器中的一个转换数据有效后，ADC 会在 ADC_ISR 寄存器中置位 EOC 标志。当 ADC_IER 中的 EOCIE 置为 1 时，则会产生一个 EOC 中断。EOC 标志由软件写 1 清零或读 ADC_DR 寄存器来清零。

同样，ADC 会在采样阶段结束时，置位 ADC_ISR 寄存器中 EOSMP 标志位。EOSMP 标志由软件写 1 清零。当在 ADC_IER 寄存器中的 EOSMPIE 置为 1 后，则会产生一个 EOSMP 中断。

此中断用于处理与转换同步，通常，模拟多路复用器可以在转换阶段的隐蔽时间内进行访问修改，这样就可以在下次采样开始前设置好多路复用器映射。

注：由于在采样结束和转换结束之间间隔时间很短，使用轮询或 WFE 指令进行处理优于中断和 WFI 指令。

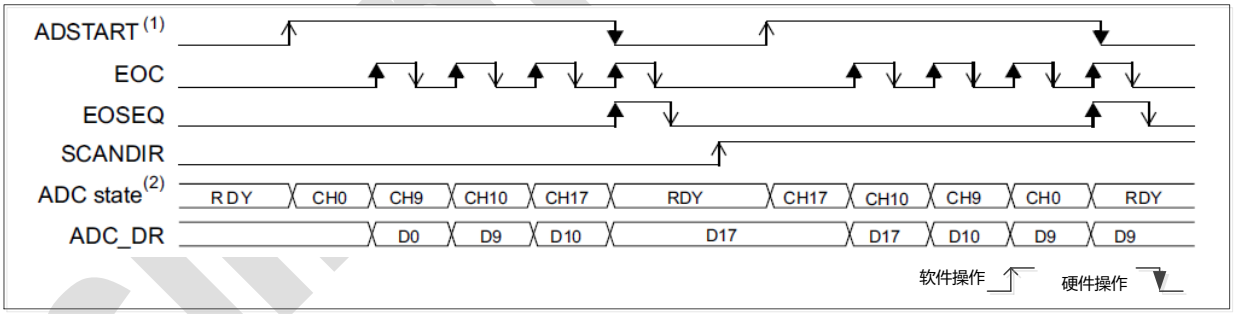
注：EOC 复位和新转换结束同时发生，则溢出标志（overrun flag）不会置位。

12.5.4 序列转换结束（EOSEQ 标志）

一旦一个转换序列的最后一个转换数据有效后（ADC_DR 寄存器中），ADC 会在 ADC_ISR 寄存器中置位 EOSEQ 标志。当 ADC_IER 中的 EOSEQIE 置为 1 时，则会产生一个 EOSEQ 中断。EOSEQ 标志由软件写 1 清零。

12.5.5 示例时序图（单次/连续模式，硬件/软件触发）

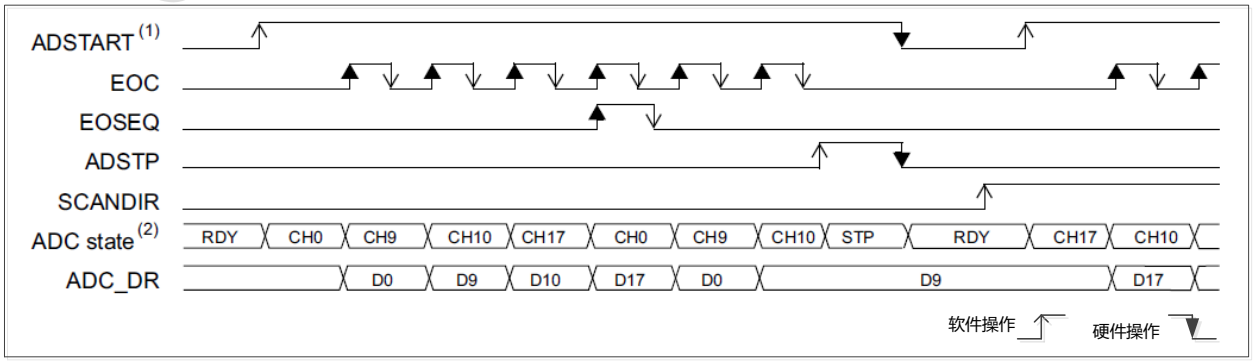
一个序列的单次转换，软件触发



注 1：EXTEN=0x0, CONT=0

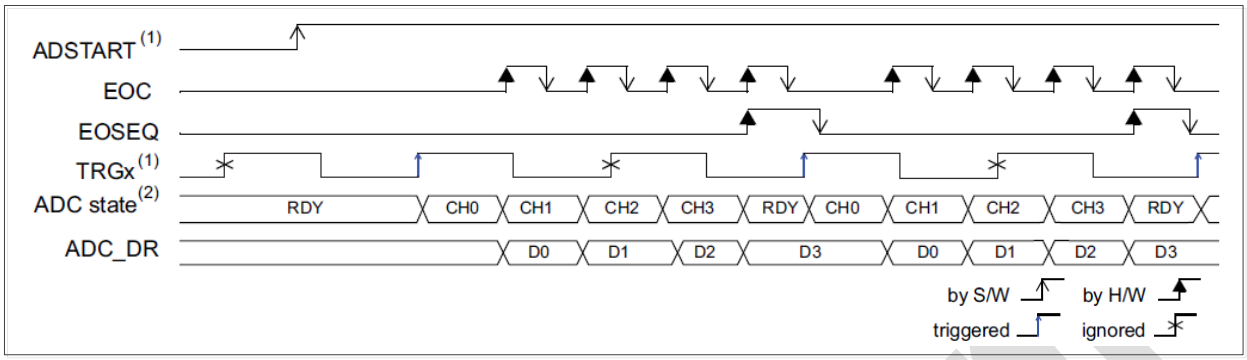
注 2：CHSEL=0x20601, WAIT=0, AUTOFF=0

一个序列的连续转换，软件触发



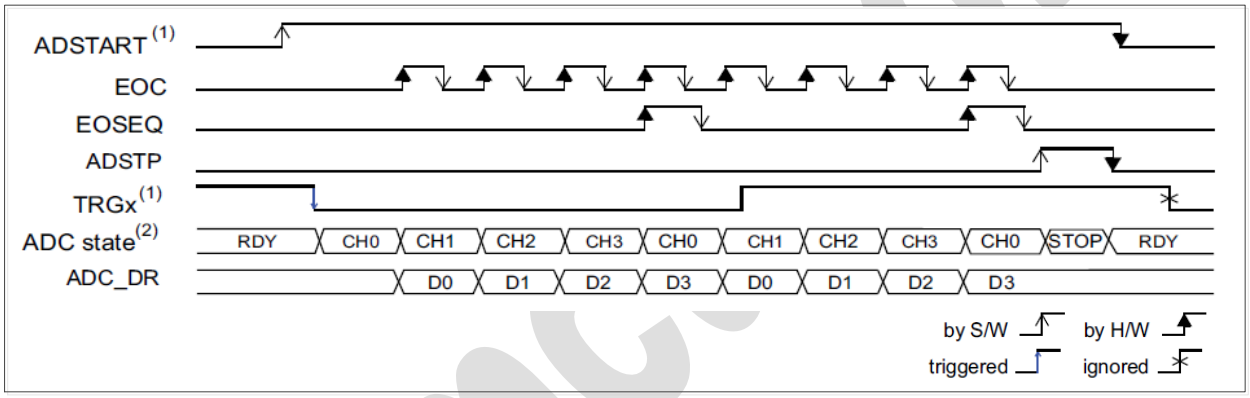
注 1：EXTEN=0x0, CONT=1
注 2：CHSEL=0x20601, WAIT=0, AUTOFF=0

一个序列的单次转换，硬件触发



注 1：EXTSEL=TRGx（over-frequency）, EXTEN=01（上升沿）, CONT=0
注 2：CHSEL=0xF, SCANDIR=0, WAIT=0, AUTOFF=0

一个序列的连续转换，硬件触发



注 1：EXTSEL=TRGx, EXTEN=10（下降沿）, CONT=1
注 2：CHSEL=0xF, SCANDIR=0, WAIT=0, AUTOFF=0

12.6 数据管理

12.6.1 数据寄存器与数据对齐（ADC_DR, ALIGN）

在每次转换结束（当 EOC 事件产生时），转换的结果数据被存放到 16 位 ADC_DR 数据寄存器中。

ADC_DR 数据格式取决于配置的数据对齐方式和分辨率。

ADC_CFGR1 寄存器中的 ALIGN 位用于选择数据存储的对齐方式，数据可选为右对齐（ALIGN=0）或左对齐(ALIGN=1)。如下图所示。

数据对齐与分辨率

ALIGN	RES	B 15	B 14	B 13	B 12	B 11	B 10	B9	B8	B7	B6	B2	B4	B3	B2	B1	B0
0	00	0x00						DR[11:0]									
	01	0x00								DR[9:0]							
	10	0x00										DR[7:0]					
	11	0x00												DR[5:0]			
1	00	DR[11:0]													0x00		
	01	DR[9:0]												0x00			
	10	DR[7:0]										0x00					
	11	0x00										DR[5:0]				0x00	

12.6.2 ADC 溢出（OVR, OVRMOD）

ADC 溢出标志（OVR）表示一个数据溢出事件，当转换好的数据未被 CPU 或 DMA 及时读取，新的转换数据已经有效时，就发生了 ADC 溢出。

当一个新的转换完成时，如果 EOC 标志仍然为‘1’，则在 ADC_ISR 寄存器中置位 OVR 标志。如果在 ADC_IER 中设置了 OVRIE 位，就会产生一个 OVR 中断。

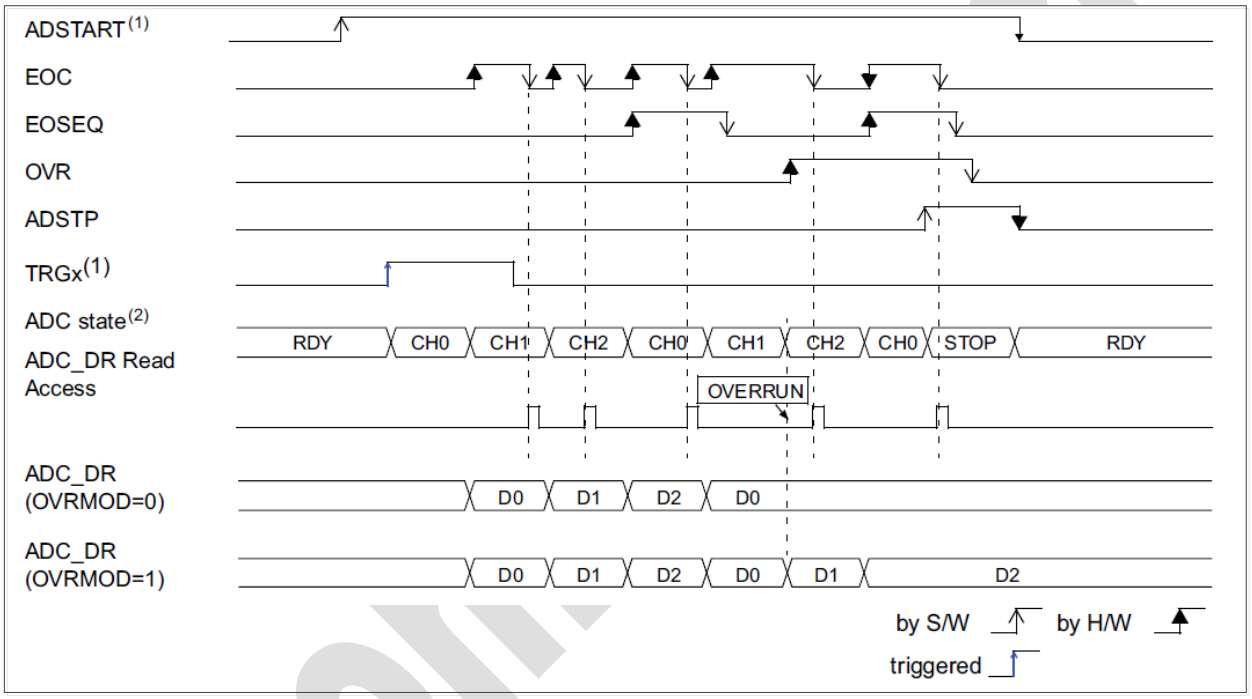
当发生溢出时，ADC 继续工作，并继续转换，除非软件停止并复位序列转换（通过设置 ADC_CR 寄存器中的 ADSTP 为 1）。

OVR 标志软件写 1 清零。

通过对 ADC_CFGR1 寄存器中的 OVRMOD 位编程，可配置当发生溢出事件时的数据是被保留还是被覆盖：

- ✧ OVRMOD=0
  - 溢出事件发生，数据寄存器(ADC_DR)的值被保留不被覆盖：老数据被保持，新的转换数据丢弃。若 OVR 保持为 1，则后续的转换会被执行但结果都被丢弃。
- ✧ OVRMOD=1
  - 数据寄存器的值被最后的转换结果覆盖，之前未读取的数据将丢失。若若 OVR 保持为 1，则后续的转换会被执行且数据寄存器（ADC_DR）一直保持最后转换结果

溢出示例（OVR）



12.6.3 管理序列转换数据（无 DMA 参与）

若 ADC 的转换足够慢，转换序列可由软件来控制。这种情况下，软件应用 EOC 标志及其关联的中断去处理每个转换数据。当每次转换结束时，ADC_ISR 寄存器中的 EOC 位置位， 此时可读 ADC_DR 寄存器的转换值。配置 ADC_CFGR1 寄存器中的 OVRMOD 位为 0，来管理过冲事件异常。

12.6.4 管理转换数据（无 DMA 和无溢出）

应用中会需要 ADC 转换一个或多个通道但并不会每次转换结束都读取结果。这种情况下，需要配置 OVRMOD=1，且 OVR 标志需要软件忽略。当 OVRMOD=1 时，溢出事件不会阻止 ADC 继续转换，ADC_DR 寄存器总是保留最新的转换数据。

12.6.5 管理转换数据（使用 DMA）

由于所有通道的转换结果数据存放到一个单一的数据寄存器中，当转换多个通道时用 DMA 方式会更高效。这样可以避免 ADC_DR 寄存器的转换结果丢失。

当 DMA 模式开启时(ADC_CFGR1 寄存器中的 DMAEN =1)，每次转换结束时都会产生一个 DMA 请求。这允许把在 ADC_DR 寄存器中的转换数据传送到软件指定的目标地址中。

注：ADC_CFGR1 寄存器中的 DMAEN 位必须在 ADC 校准之后置位。

当因 DMA 无法及时响应 DMA 传输请求而发生溢出时（OVR=1），ADC 会停止产生 DMA 请求，对应的新转换数据不会进行 DMA 传输，这意味着 DMA 传输到 RAM 区的数据都是有效的。

根据 OVRMOD 位的配置，ADC_DR 寄存器中的数据可选择为：被保留或覆盖（参见 12.6.2 ADC 溢出（OVR，OVRMOD））。

当 OVR=1 时，DMA 传输请求会被阻塞，直到软件清除 OVR 位。

配置 ADC_CFGR1 寄存器中的 DMACFG 位，选择两种 DMA 模式：

- ✧ DMA 一次模式 (one shot mode)(DMACFG=0)
  当 DMA 编程用于传输固定长度的数据时，可选用该模式。
- ✧ DMA 循环模式 (DMACFG=1)
  当 DMA 编程为循环模式或双缓冲模式时，可选用该模式。

DMA 一次模式 (DMACFG=0)

在这种模式下，ADC 在每次转换的数据有效时产生一次 DMA 请求。一旦 DMA 已达到最后一个 DMA 传输（此时产生一个 DMA_EOT 中断，参见章节：DMA 控制器）时，即使 ADC 转换已重新启动，ADC 都会停止产生 DMA 请求。

注：产生 DMA_EOT 中断时，下一次的 ADC 转换有可能已开始。

当 DMA 传输完成（DMA 控制器中配置的所有传输已经完成）：

- ✧ ADC 数据寄存器的内容冻结
- ✧ 任何进行中的转换中止且结果值丢弃
- ✧ 停止向 DMA 控制器发出新 DMA 请求。假如仍有 ADC 转换启动，这种方式可避免产生一个 ADC 溢出错误。
- ✧ ADC 扫描序列停止并复位
- ✧ DMA 停止

DMA 循环模式 (DMACFG=1)

在这种模式下，ADC 在每次转换的数据有效时产生一次 DMA 请求（即使 DMA 达到最后一个 DMA 传输）。这样允许 DMA 配置为循环模式，来处理连续模拟输入数据流。

12.7 低功耗特性

12.7.1 等待模式转换（WAIT）

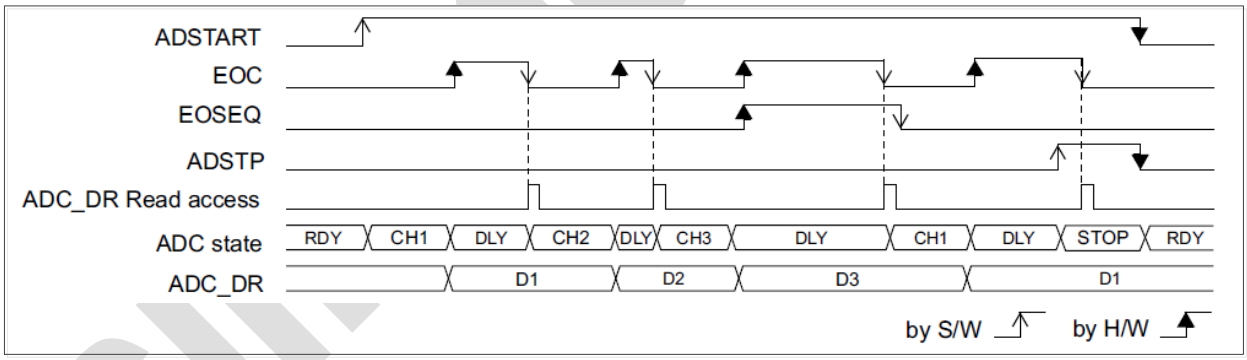
等待模式可用于在低速运行时简化软件以及优化应用程序的性能，因为低频率可能会出现 ADC 溢出的风险。

当 ADC_CFGR1 寄存器中 WAIT 位设置为 1 时，新的转换只有在先前的数据被处理后才能启动。举例，ADC_DR 寄存器数据被读取或 EOC 标志被清零。

这是一种根据系统读取数据速度自动调整 ADC 速度的方法。

注：当一个转换正在进行或处于等待时间（读访问之前），任何硬件触发都将被忽略。

等待模式（wait mode）转换（连续模式，软件触发）



注 1：EXTEN=0x0, CONT=1

注 2：CHSEL=0x3, SCANDIR=0, WAIT=1, AUTOFF=0

12.8 模拟窗口看门狗（AWDEN, AWDSGL, AWDCH, AWD_HTR/LTR, AWD）

AWD 模拟看门狗的功能由 ADC_CFGR1 寄存器中的 AWDEN 位置位来开启。 它可用于监测所选的单一通道或所有使能通道（参见下表: 模拟看门狗通道选择）保持在配置电压范围（window）内，如下图所示。

若 ADC 转换的模拟电压低于阈值下限或高于阈值上限，AWD 模拟看门狗的状态标志被置位。高低阈值在 ADC_HTR 和 ADC_LTR 寄存器中设置，低 12 位有效。若置位 ADC_IER 寄存器中的 AWDIE 位，将产生 AWD 中断。

AWD 标志位，软件写 1 清零。

若转换分辨率低于 12 位（DRES[1:0]位选择），则阈值寄存器中的低位必须保持为 0，因为内部数据比较都是按照全 12 位（左对齐）的方式进行比较的。

模拟看门狗比较操作

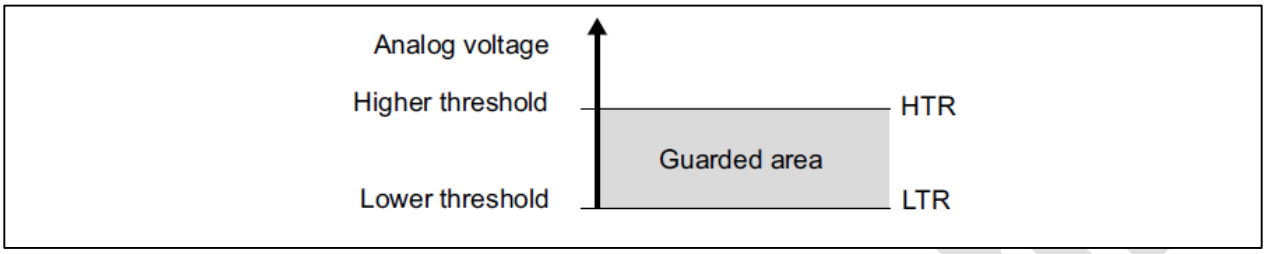
分辨率 RES[1:0]	模拟看门狗比较值		说明
	原始转换数据，左对齐	阈值	

00：12 位	DATA[11:0]	LT[11:0] ,HT[11:0]	-
01：10 位	DATA[11:2]+00B	LT[11:0] ,HT[11:0]	用户必须配置 LT1[1:0]=00B, HT1[1:0]=00B
10：8 位	DATA[11:4]+0000B	LT[11:0] ,HT[11:0]	用户必须配置 LT1[1:0]=00B, HT1[1:0]=00B
11：6 位	DATA[11:6]+000000B	LT[11:0] ,HT[11:0]	用户必须配置 LT1[1:0]=00B, HT1[1:0]=00B

注：看门狗比较，发生在对齐处理前，对原始转换数据进行比较。

下表展示了如何配置 ADC_CFGR1 寄存器中的 AWDSGL 和 AWDEN 位，在一个或多个通道上启用模拟看门狗。

模拟看门狗保护区



看门狗通道选择

由模拟看门狗监控的通道	AWDSGL 位	AWDEN 位
无	x	0
所有通道	0	1
单一通道 (1)	1	1

注：由 AWDCH[4:0] 位来选择具体通道。

12.9 温度传感器和内部参考电压

温度传感器可以用来测量器件的结温(TJ)。

温度传感器内部连接 ADC_IN16 输入通道，用于将传感器的输出电压转换为数字值。温度传感器模拟引脚的采样时间必须大于数据表中指定的最小 TS_temp 值。不使用时，可将传感器置于下电模式。

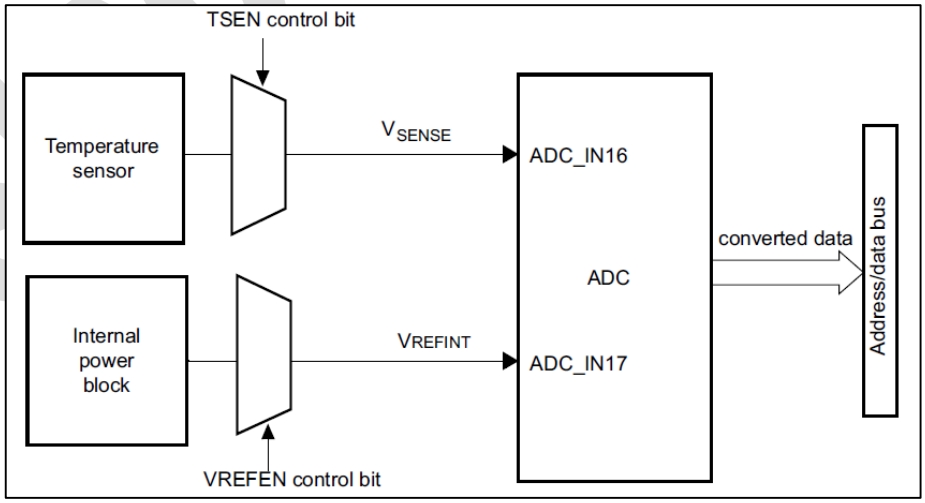
温度传感器输出电压随温度呈线性变化，但由于工艺的不同，其特性在不同芯片之间可能会有很大的变化。为了提高温度传感器的精度(特别是绝对温度测量)，在出厂测试时，会测量每颗芯片的校准值，并存储在系统内存区（system memory）。有关其他信息，请参阅特定芯片数据手册。

内部电压参考(VREFINT)提供一个稳定的(bandgap)电压输出，提供给 ADC 和模拟比较器。VREFINT 内部连接到 ADC_IN17 输入通道。VREFINT 的精确电压，在出厂测试时，为每个芯片单独测量，并存储在系统内存区（system memory）。

下图展示了温度传感器、内部电压基准和 ADC 之间的连接框图。

TSEN 位置位，ADC_IN16(温度传感器)的转换使能，VREFEN 位置位，ADC_IN17 (VREFINT)的转换使能。

温度传感器和 V_{REFINT} 通道框图



读取温度流程

1、 选择 ADC1_IN16 输入通道

2、 选择适当的采样时间，参考芯片数据手册 TS_temp 参数。

3、 ADC_CCR 寄存器中设置 TSEN 位用来唤醒从断电模式下的温度传感器，并等待其稳定时间(t_{START})。

4、 置位 ADC_CR 寄存器中的 ADSTART 位 (也可用外部触发) 来启动 ADC 转换。
5、 从 ADC_DR 寄存器中读取转换数据 VADC_Data
计算温度：**TADC_Data(℃) =(VDDA / FULL_SCALE x ADC_Data - b) / k**
6、 从 sysrom 出厂区读出 ADC_Temp1、Sens_Temp1 为出厂校准值；详细参考***数据手册***描述
计算校准点温度(芯片)：**TADC_Temp1(℃) = (VDDA_3v3 / FULL_SCALE x ADC_Temp1 - b) / k**
计算校准点温度(环境)：**TSens_Temp1(℃) = 0.0625 x Sens_Temp1**
计算温度 offset：**Toffset = TSens_Temp1 - TADC_Temp1**
7、 修正实际温度：
实际温度 **T（℃）= TADC_DATA + Toffset**
其中，
--VDDA_3v3 为出厂校准时模拟电源，电压= 3.3 v；
--VDDA 为实际应用中模拟电源，由实际应用确定
--FULL_SCALE = 4095
--ADC_Data，实际 ADC 转换值
--k 为温度传感器电压平均斜率 Avg_Slope = 0.0043v/℃，以实际***数据手册***为准；
--b 为温度传感器电压 vs 温度一次曲线常数项（截距），b=1.063 v
--ADC_Temp1、Sens_Temp1 为出厂校准值，参见***数据手册***描述
若实际应用 VDDA 为 3.3v，公式可以简化为：
实际温度 **T（℃）= ((VADC_Data-VADC_Temp1)/Avg_slope) +TSens_Temp1**

注：传感器从断电模式下唤醒时到能正确输出 V_{SENSE} 的值要有一个启动时间，另 ADC 从上电后启动也有一个启动时间，若要最小化这个延时，则需要在同一时间置位 ADEN 和 TSEN 位。

使用内部参考电压计算实际 VDDA 电压

MCU 的 V_{DDA} 电源电压可能会发生变化或不能精确知道。内部基准电压的校准数据，是在 V_{DDA}=3.3V 条件下，通过 ADC 生产测试获得，此数据可以用于评估实际 V_{DDA} 电压值变化。

实际 V_{DDA} 电压计算如下：

V_{DDA} = 3.3V x VREFINT_CAL / VREFINT_DATA

其中，

-- VREFINT_CAL 为 VREFINT 校准值

-- VREFINT_DATA 为实际 VREFTINT 通道 ADC 转换值。

通过相对电源电压的 ADC 测量计算绝对电压

ADC 的转换参考源为 V_{DDA} 电源，对一个通道进行 ADC 转换，得到的数字量化值反映通道电压和 VDDA 电压的线性关系。

应用中，若 V_{DDA} 电压已知，ADC 转换值右对齐，可以用下面公式计算绝对电压：

$V_{CHANNEL}$ = V_{DDA} / FULL_SCALE x ADC_DATA

应用中，若 V_{DDA} 电压未知，可以通过章节<使用内部参考电压计算实际 VDDA 电压>中 V_{DDA} 和 VREFINT 关系式替换，结果如下：

$V_{CHANNEL}$ = (3.3V x VREFINT_CAL x ADC_DATA) / (VREFINT_DATA x FULL_SCALE)

其中，

-- VREFINT_CAL 为 VREFINT 校准值

-- VREFINT_DATA 为实际 VREFTINT 通道 ADC 转换值。

--ADC_DATA 为通道 ADC 测量值，右对齐

--FULL_SCALE 为 ADC 转换的最大数字量化值。举例，12bit 分辨率为 2^{12}-1=4095，8bit 分辨率为 2^{8}-1=255。

注：如果 ADC 测量使用的不是 12 位右对齐的输出格式，那么在计算之前，所有参数必须首先转换为兼容格式。

12.10 电池电压监测

置位 ADC_CCR 寄存器中的 VBATEN 位，允许应用监测备份域电池电压（V_{BAT} 引脚电压）。

由于 V_{BAT} 电压可能会高于 V_{DDA} 电压，为了保证 ADC 的正确操作，V_{BAT} 引脚内部连接至桥电路进行 2 倍分压。当 VBATEN 置位，内部分压电路自动使能，V_{BAT}/2 连接至 ADC_IN18 输入通道，转换的结果为 V_{BAT} 电压的一半。

为了防止不必要的电池耗损，推荐仅在 ADC 转换需要时才开启分压桥电路。

12.11 ADC 中断

ADC 中断可由以下任一事件产生：

- ✧ ADC 上电，当 ADC 准备好（ADRDY 标志）
- ✧ 任何一次的转换结束（EOC 标志）
- ✧ 序列转换结束（EOSEQ 标志）
- ✧ 当模拟看门狗检测发生（AWD 标志）
- ✧ 当采样阶段结束发生（EOSMP 标志）

✧ 当数据溢出发生（OVR 标志）

独立的中断使能位用于灵活设置 ADC 中断。

ADC 中断

中断事件	事件标志	使能控制位
ADC 准备好（ready）	ADRDY	ADRDYIE
转换结束	EOC	EOCIE
序列转换结束	EOSEQ	EOSEQIE
模拟看门狗状态置位	AWD	AWDIE
采样阶段结束	EOSMP	EOSMPIE
溢出（overrun）	OVR	OVRIE

12.12 相关寄存器

12.12.1 寄存器汇总表

基址：0x4001 2400				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	ADC_ISR	0x0000 0000	ADC 中断和状态寄存器	12.12.2
0x04	ADC_IER	0x0000 0000	ADC 中断使能寄存器	12.12.3
0x08	ADC_CR	0x0000 0000	ADC 控制寄存器	12.12.4
0x0C	ADC_CFGR1	0x0000 0000	ADC 配置寄存器 1	12.12.5
0x10	ADC_CFGR2	0x0000 0000	ADC 配置寄存器 2	12.12.6
0x14	ADC_SMPR	0x0000 0000	ADC 采样时间寄存器	12.12.7
0x18~0x1C	Res.	-	-	
0x20	ADC_TR	0x0FFF 0000	ADC 看门狗阀值寄存器	12.12.8
0x24	Res.	-	-	
0x28	ADC_CHSELR	0x0000 0000	ADC 通道选择寄存器	12.12.9
0x2C~0x3C	Res.	-	-	
0x40	ADC_DR	0x0000 0000	ADC 数据寄存器	12.12.10
0x44~0x304	Res.	-	-	
0x308	ADC_CCR	0x0000 0000	ADC 通用配置寄存器	12.12.11

12.12.2 ADC 中断和状态寄存器（ADC_ISR）

ADC_ISR			地址偏移	0x00		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	AWD	-	-	OVER	EOSEQ	EOC	EOSMP	ADRDY
R/W	rc_w1	-	-	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
复位值	0	-	-	0	0	0	0	0

BIT[31:8]	-	保留，保持复位值
BIT[7]	AWD	模拟看门狗标志 当转换后的电压低于 ADC_TR 的 LT[11:0]或高于 ADC_TR 的 HT[11:0]设定的电压时，该位由硬件置位。软件对该位写 1 清零。 0：无模拟看门狗事件产生（或该事件标志已被软件确认并清零） 1：产生了模拟看门狗事件

BIT[6:5]	-	保留，保持复位值
BIT[4]	OVR	ADC 溢出 当溢出产生时该位由硬件置位，即新的转换结束且 EOC 已置 1 。该位软件写 1 清零。 0：无溢出事件产生（或该事件标志已被软件确认并清零） 1：溢出发生
BIT[3]	EOSEQ	序列转换结束标志 由 CHSEL 位所选的通道序列转换结束后，该位由硬件置位。软件对该位写 1 清零。 0：序列转换未完成（或该事件标志已被软件确认并清零） 1：序列转换完成
BIT[2]	EOC	转换结束标志 当每个通道新转换结束且结果有效时（存放在 ADC_DR 中），该位由硬件置位。软件对该位写 1 清零或读取 ADC_DR 寄存器来清零。 0：通道转换未结束（或该事件标志已被软件确认并清零或读 ADC_DR 寄存器清零） 1：通道转换结束
BIT[1]	EOSMP	采样结束标志 在转换期间的采样阶段结束时，该位由硬件置位。 0：不在采样结束阶段（或该事件标志已被软件确认并清零） 1：采样阶段结束
BIT[0]	ADRDY	ADC 就绪 在 ADC 使能后（ADEN=1）且 ADC 达到准备接受转换请求的状态时，该位由硬件置位。该位软件写 1 清零。 0：ADC 未准备好（或该事件标志已被软件确认并清零） 1：ADC 已准备好开始转换

注：在自动关闭模式（AUTOFF=1）上电/下电阶段由硬件自动执行，ADRDY 不会只置位。

12.12.3 ADC 中断使能寄存器（ADC_IER）

ADC_IER			地址偏移	0x04		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	AWDIE	-	-	OVERIE	EOSEQIE	EOCIE	EOSMPIE	ADRDYIE
R/W	rw	-	-	rw	rw	rw	rw	rw
复位值	0	-	-	0	0	0	0	0

BIT[31:8]	-	保留，保持复位值
BIT[7]	AWDIE	模拟看门狗中断使能，软件置位和清零 0：模拟看门狗中断禁止 1：模拟看门狗中断使能 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写该位。*
BIT[6:5]	-	保留，保持复位值
BIT[4]	OVRIE	溢出中断使能，软件置位和清零 0：溢出中断关闭 1：溢出中断开启。当 OVR 置位时产生中断。 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写该位。*
BIT[3]	EOSEQIE	序列转换结束中断使能，软件置位和清零 0：EOS 中断关闭

		1：EOS 中断开启。当 EOS 置位时产生中断。 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写该位。*
BIT[2]	EOCIE	转换结束中断使能，软件置位和清零 0：EOC 中断关闭 1：EOC 中断开启。当 EOC 置位时产生中断。 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写该位。*
BIT[1]	EOSMPIE	采样结束中断使能，软件置位和清零 0：EOSMP 中断关闭 . 1：EOSMP 中断开启。当 EOSMP 置位时产生中断。 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写该位。*
BIT[0]	ADRDYIE	ADC 就绪中断使能，软件置位和清零 0：ADRDY 中断关闭 1：ADRDY 中断开启。当 ADRDY 置位时产生中断。 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写该位。*

12.12.4 ADC 控制寄存器（ADC_CR）

ADC_CR			地址偏移	0x08		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	ADCAL	-	-	-	-	-	-	-
R/W	rs	-	-	-	-	-	-	-
复位值	0	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	ADSTP	-	ADSTART	ADDIS	ADEN
R/W	-	-	-	rs	-	rs	rs	rs
复位值	-	-	-	0	-	0	0	0

BIT[31]	ADCAL	ADC 校准 该位由软件置位来启动 ADC 校准。当校准完成后，由硬件清零。 0：校准完成。 1：写 1 开始校准 ADC，读为 1 时意味着校准进行中。 *注：只有在 ADC 关闭情况下(ADCAL=0, ADSTART=0, ADSTP=0, ADDIS=0 和 ADEN=0)，才允许软件置位 ADCAL。*
BIT[30:5]	-	保留，保持复位值
BIT[4]	ADSTP	ADC 停止转换命令 该位由软件置位来停止和放弃正在进行中的转换。 当转换停止生效时，该位由硬件清零，同时 ADC 准备好接受新的转换命令。 0：没有正在进行的 ADC 停止转换命令。 1：写 1 停止 ADC 转换，读为 1 时表明 ADSTP 命令正在执行中。 *注：只有当 ADSTART=1 和 ADDIS=0（ADC 开启且可能正在转换，且无禁止 ADC 的挂起请求 ），软件才能对该位进行置位。*
BIT[3]	-	保留，保持复位值
BIT[2]	ADSTART	ADC 开始转换命令 该位由软件置位来启动 ADC 转换。一次转换可立即启动 （ 软件触发配置 ） 或硬件触发产生（ 硬件触发配置），启动方式由 EXTEN[1:0]位的配置来决定。 该位由硬件清零： - 在单次转换模式中（CONT=0,DISCEN=0），当选择为软件触发(EXTEN=00)时：序列转换结束(EOSEQ 置位 ） 后，该位清零。 - 当执行完 ADSTP 命令后，在 ADSTP 位由硬件清零时，该位清零。 0：无进行中的 ADC 转换。

		1：写 1 开始 ADC 转换。读为 1 表明 ADC 正在进行转换。 *注：只有当 ADEN=1 和 ADDIS=0（ADC 开启且可能正在转换，且无禁止 ADC 的挂起请求 ），软件才能对该位进行置位。*
BIT[1]	ADDIS	ADC 关闭命令 该位由软件置位来禁止 ADC（ADDIS 命令）并让 ADC 处于掉电状态（关断状态）。一旦 ADC 关闭生效（ADEN 同时被硬件清零 ）后由硬件清零该位。 0：无 ADDIS 命令进行中 1：写 1 关闭 ADC。读为 1 时表明 ADDIS 命令正在执行中。 *注：只有当 ADEN=1 和 ADSTART=0（ 确保无进行中的转换）时，才允许软件置位 ADDIS 位。*
BIT[0]	ADEN	ADC 使能命令 该位由软件置位来使能 ADC。一旦 ADRDY 标志置为 1 时，表明 ADC 可供使用。执行 ADDIS 命令后，ADC 关断，并由硬件清零该位。 0：ADC 关闭（关断状态） 1：写 1 来使能 ADC *注：只有在 ADC_CR 寄存器所有位为 0（ADCAL=0, ADSTP=0, ADSTART=0, ADDIS=0 和 ADEN=0）的情况下，软件才能置位 ADEN 位。*

12.12.5 ADC 配置寄存器 1（ADC_CFGR1）

ADC_CFGR1			地址偏移	0x0C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	AWDCH[4:0]					-	-
R/W	-	rw	rw	rw	rw	rw	-	-
复位值	-	0	0	0	0	0	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	AWDEN	AWDSGL	-	-	-	-	-	DISCEN
R/W	rw	rw	-	-	-	-	-	rw
复位值	0	0	-	-	-	-	-	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	WAIT	CONT	OVRMOD	EXTEN[1:0]		-	EXTSEL[2]
R/W	-	rw	rw	rw	rw	rw	-	rw
复位值	-	0	0	0	0	0	-	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	EXTSEL[1:0]		ALIGN	RES[1:0]		SCANDIR	DMACFG	DMAEN
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31]	-	保留，保持复位值
BIT[30:26]	AWDCH[4:0]	模拟看门狗通道选择，软件置位和清零 设置模拟看门狗监视的 ADC 输入通道。 00000：输入通道 0 00001：输入通道 1 … 10010：输入通道 18 10011：输入通道 19 10100：输入通道 20 10101：输入通道 21 其它值：保留，不可使用 *注 1：被 AWDCH[4:0]位所选择的通道必须同样写入 CHSELR 寄存器。* *注 2：只有当 ADSTART=0 时(确定无进行中的转换)才允许改写这些位。*
BIT[25:24]	-	保留，保持复位值
BIT[23]	AWDEN	模拟看门狗使能位，软件置位和清零 0：模拟看门狗关闭 1：模拟看门狗开启 *注：只有当 ADSTART=0 时（确定无进行中的转换)才允许改写该位。*
BIT[22]	AWDSGL	在单一通道或所有通道使能看门狗，软件置位和清零 0：在所有通道上使能模拟看门狗 1：在单一通道上使能模拟看门狗（由 AWDCH[4:0]位指定）

		注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。
BIT[21:17]	-	保留，保持复位值
BIT[16]	DISCEN	断续模式，软件置位和清零 0：断续模式禁止 1：断续模式开启 *注：不可能同时开启断续模式和连续模式，禁止同时置位此两位（DISCEN=1, CONT=1）；在这种情况下（若 DISCEN=1, CONT=1），则 ADC 认为连续模式禁止。* *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*
BIT[15]	-	保留，保持复位值
BIT[14]	WAIT	等待转换模式，软件置位和清零 0：等待转换模式关闭 1：等待转换模式开启 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*
BIT[13]	CONT	单次/连续转换模式，软件置位和清零 若该位置位，转换为连续模式直到该位被清零。 0：单次转换模式 1：连续转换模式 *注：不可能同时开启断续模式和连续模式，禁止同时置位此两位（DISCEN=1, CONT=1）；在这种情况下（若 DISCEN=1, CONT=1），则 ADC 认为连续模式禁止。* *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*
BIT[12]	OVRMOD	溢出管理模式，软件置位和清零，来配置溢出管理 0：当检测到溢出事件时，ADC_DR 寄存器保持为老数据 1：当检测到溢出事件时，ADC_DR 寄存器用最后一次的转换数据覆盖 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*
BIT[11:10]	EXTEN[1:0]	外部触发使能和极性选择，软件置位和清零 00：硬件触发检测关闭（可由软件启动转换） 01：在上升沿进行硬件触发检测 10：在下降沿进行硬件触发检测 11：在上升和下降沿都进行硬件触发检测 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*
BIT[9]	-	保留，保持复位值
BIT[8:6]	EXTSEL[2:0]	外部触发选择 这些位用于选择触发 ADC 转换的外部事件：参考表：外部触发源 000：TRG0(TIM1_TRGO) 001：TRG1(TIM1_CC4) 010：TRG2(TIM2_TRGO) 011：TRG3(TIM3_TRGO) 100：TRG4 101：TRG5 110：TRG6 111：TRG7 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*
BIT[5]	ALIGN	数据对齐，软件置位和清零，用来选择数据的左对齐或右对齐。 0：右对齐 1：左对齐 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*
BIT[4:3]	RES[1:0]	数据分辨率，软件置位和清零，用于选择 ADC 转换的数据分辨率。 00：12 位 01：10 位 10：8 位 11：6 位 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*
BIT[2]	SCANDIR	扫描序列方向，软件置位和清零，选择通道序列中的通道扫描方向。 0：向前扫描（从 CHSEL0 到 CHSEL16） 1：向后扫描（从 CHSEL16 到 CHSEL0） *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*
BIT[1]	DMACFG	DMA 配置，软件置位和清零

		选择 DMA 模式，只有在 DMAEN=1 时起作用。 0：DMA 单次模式 1：DMA 循环模式 详情请参考<用 DMA 管理转换的数据>章节。 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*
BIT[0]	DMAEN	DMA 使能，软件置位和清零 使能 DMA 的请求。允许用 DMA 控制器来自动管理转换的结果数据。详情请参考<用 DMA 管理转换的数据>章节。 0：DMA 关闭 1：DMA 使能 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*

12.12.6 ADC 配置寄存器 2（ADC_CFGR2）

ADC_CFGR2			地址偏移	0x10		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	CKMODE[1:0]		-	-	-	-	-	-
R/W	rw	rw	-	-	-	-	-	-
复位值	0	0	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-

BIT[31:30]	CKMODE[1:0]	ADC 时钟模式，软件置位和清零，定义模拟 ADC 时钟 00：ADCCLK（异步时钟模式），产品级产生（参见 RCC 选择） 01：PCLK/2（同步时钟模式） 10：PCLK/4（同步时钟模式） 11：保留 在所有的同步时钟模式中，从定时器触发器到开始转换的延迟没有抖动。 *注：只有在 ADC 关闭情况下(ADCAL=0, ADSTART=0, ADSTP=0, ADDIS=0 和 ADEN=0)，才允许软件改写此位。*
BIT[29:0]	-	保留，保持复位值

12.12.7 ADC 采样时间寄存器（ADC_SMPR）

ADC_SMPR			地址偏移	0x14		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	SMP[2:0]		

R/W	-	-	-	-	-	rw	rw	rw
复位值	-	-	-	-	-	0	0	0

BIT[31:3]	-	保留，保持复位值
BIT[2:0]	SMP[2:0]	采样时间选择，软件改写，用于选择所有通道的采样时间。 000：3.5 ADC 时钟周期 001：7.5 ADC 时钟周期 010：13.5 ADC 时钟周期 011：28.5 ADC 时钟周期 100：41.5 ADC 时钟周期 101：55.5 ADC 时钟周期 110：71.5 ADC 时钟周期 111：239.5 ADC 时钟周期 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*

12.12.8 ADC 看门狗阀值寄存器（ADC_TR）

ADC_TR			地址偏移	0x20		复位值	0x0FFF 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	HT[11:8]			
R/W	-	-	-	-	rw	rw	rw	rw
复位值	-	-	-	-	1	1	1	1
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	HT[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	1	1	1	1	1	1	1
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	LT[11:8]			
R/W	-	-	-	-	rw	rw	rw	rw
复位值	-	-	-	-	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	LT[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:28]	-	保留，保持复位值
BIT[27:16]	HT[11:0]	模拟看门狗的高阀值，软件改写 用来定义模拟看门狗的高阀值。 详情参见章节：模拟看门狗 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*
BIT[15:12]	-	保留，保持复位值
BIT[11:0]	LT[11:0]	模拟看门狗低阀值，软件改写 用来定义模拟看门狗的低阀值。 详情参见章节：模拟看门狗 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*

12.12.9 ADC 通道选择寄存器（ADC_CHSELR）

ADC_CHSELR			地址偏移	0x28		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	CHSEL[21:19]			CHSEL[18:16]		
R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CHSEL[15:8]							

R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CHSEL[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:22]	-	保留，保持复位值
BIT[21:0]	CHSELx	通道选择（x= 0~21），软件改写，用来定义所要转换序列的通道。 0：输入通道 x 不被选为转换通道 1：输入通道 x 被选为转换通道 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写这些位。*

12.12.10 ADC 数据寄存器（ADC_DR）

ADC_DR			地址偏移	0x40		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	DATA[15:8]							
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	DATA[7:0]							
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	-	保留，保持复位值
BIT[15:0]	DATA[15:0]	转换数据，只读 保存最后转换通道的转换结果值。该数据左对齐或右对齐参见图<数据对齐与分辨率>所示。 仅在校准完成时，DATA[6:0] 值为校准因子。

12.12.11 ADC 通用配置寄存器（ADC_CCR）

ADC_CCR			地址偏移	0x308		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	VBATEN
R/W	-	-	-	-	-	-	-	rw
复位值	-	-	-	-	-	-	-	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	TSEN	VERFEN	-	-	-	-	-	-
R/W	rw	rw	-	-	-	-	-	-
复位值	0	0	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-

BIT[31:25]	-	保留，保持复位值
BIT[24]	VBATEN	VBAT 使能，软件置位和清零，打开/关闭 VBAT 通道 0：VBAT 通道关闭 1：VBAT 通道开启 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写该位。*
BIT[23]	TSEN	温度传感器使能，软件置位和清零，打开/关闭温度传感通道。 0：温度传感通道关闭 1：温度传感通道开启 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写该位。*
BIT[22]	VREFEN	VREFINT 使能，软件置位和清零，打开/关闭 VREFINT 通道。 0：VREFINT 通道关闭 1：VREFINT 通道开启 *注：只有当 ADSTART=0 时（确定无进行中的转换）才允许改写该位。*
BIT[21:0]	-	保留，保持复位值

13 高级定时器（TIM1）

13.1 概述

高级控制定时器(TIM1)支持 6 个通道的三相 PWM 发生器功能，它具有带死区插入的互补 PWM 输出，还可以被当成完整的通用定时器。四个独立的通道可以用于：

- ✧ 输入捕获
- ✧ 输出比较
- ✧ 产生 PWM(边缘或中心对齐模式)
- ✧ 单脉冲输出

配置为16位标准定时器时,它与TIMx定时器具有相同的功能。配置为16位PWM发生器时,它具有全调制能力(0~100%)。

在调试模式下，计数器可以被冻结，同时 PWM 输出被禁止，从而切断由这些输出所控制的开关。

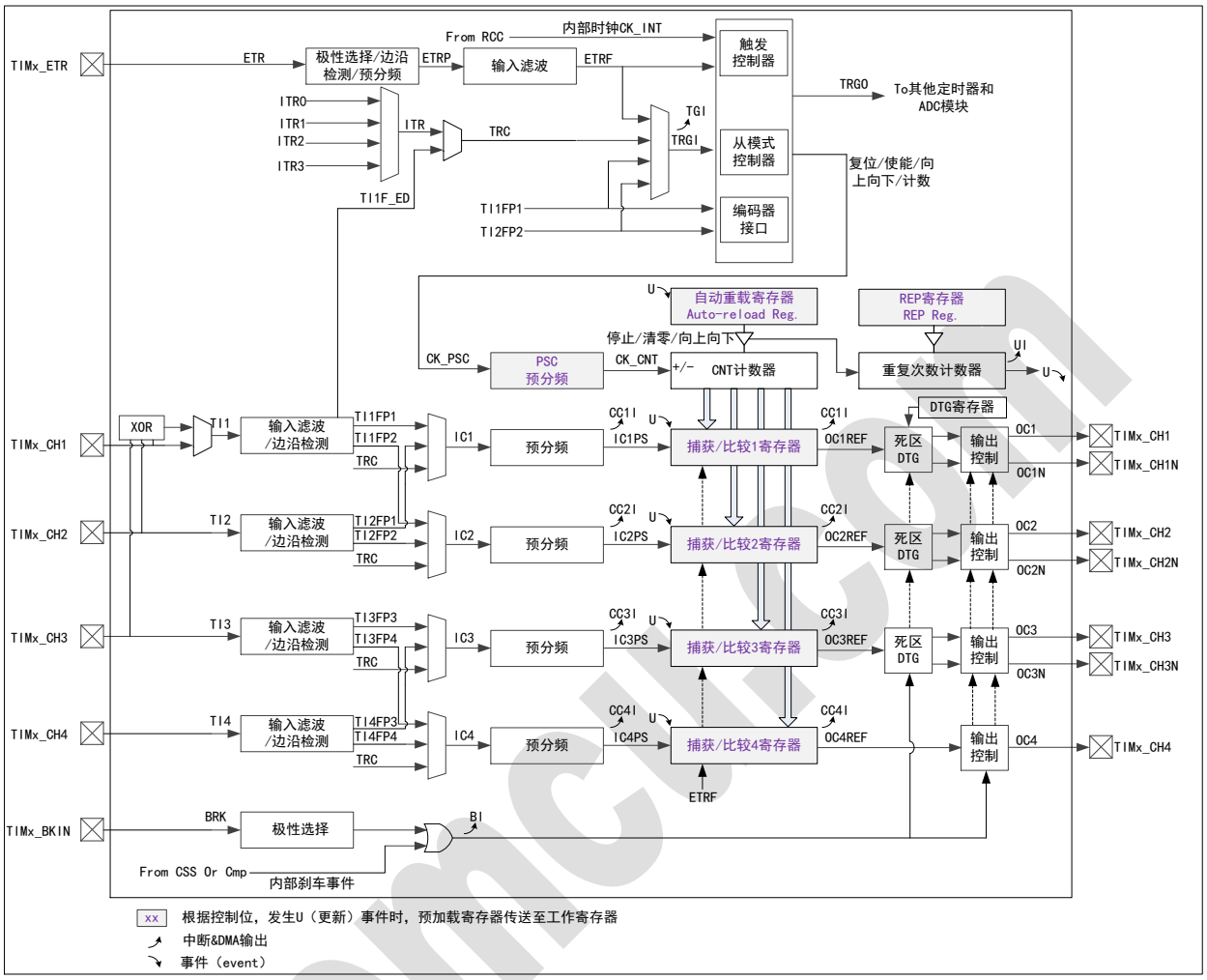
很多功能都与标准的 TIM 定时器相同，内部结构也相同，因此高级控制定时器可以通过定时器链接功能与 TIM 定时器协同操作，提供同步或事件链接功能。

13.2 特性

- ✧ 16 位向上、向下、向上/下自动装载计数器
- ✧ 16 位可编程 （ 可实时修改 ） 预分频器，支持 1～65535 之间的任意分频
- ✧ 多达 4 个独立通道，支持
- 输入捕获
- 输出比较
- PWM 生成（边沿或中心对齐）
- 单脉冲模式输出
- ✧ 互补输出，支持死区编程
- ✧ 支持外部信号同步，支持内部多个 timer 同步互联
- ✧ 重复计数器实现指定数目的计数器周期产生更新信号
- ✧ 定时器输出信号支持刹车保护
- ✧ 下述事件触发中断/DMA：
- 更新（Update）：计数器上溢/下溢，计数器初始化（通过软件或内部/外部触发）
- 触发事件（Trigger Event）：计数器启动、停止、初始化或由内部/外部触发计数
- 输入捕获（input capture）
- 输出比较（output compare）
- 刹车输入（break input）
- ✧ 支持增量式（正交）编码器和霍尔传感器定位解码功能
- ✧ 触发输入作为外部时钟或者按周期的电流管理

13.3 功能描述

TIM1 模块框图



13.3.1 时基单元

高级定时器 TIM1 主要由 16 位计数器及相关自动重载寄存器构成，计数器支持向上计数、向下计数或向上向下双向计数。计数器时钟支持预分频。

计数器寄存器、自动重载寄存器和预分频寄存器支持软件读写，即使计数器正在运行读写仍然有效。

时基单元包括：

- ✧ 计数器寄存器（TIMx_CNT）
- ✧ 预分频器寄存器（TIMx_PSC）
- ✧ 自动重载寄存器（TIMx_ARR）
- ✧ 重复次数寄存器（TIMx_RCR）

自动重载寄存器是预装载的，读写自动重载寄存器将访问预装载寄存器。设置 TIMx_CR1 寄存器中的自动重载预装载使能位（ARPE），选择预装载寄存器的内容永久传送至缓存寄存器或在每次更新事件（UEV）传送至缓存寄存器。当计数器达到溢出条件（或向下计数时的下溢条件）并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可由软件产生。有关更新事件的产生，针对每种配置后续章节会详细描述。

计数器时钟由预分频后输出 CK_CNT 驱动，需置位 TIMx_CR1 寄存器中的计数器使能位（CEN）时，CK_CNT 才有效（请参阅从模式控制器描述以获得计数器使能的更多细节）。

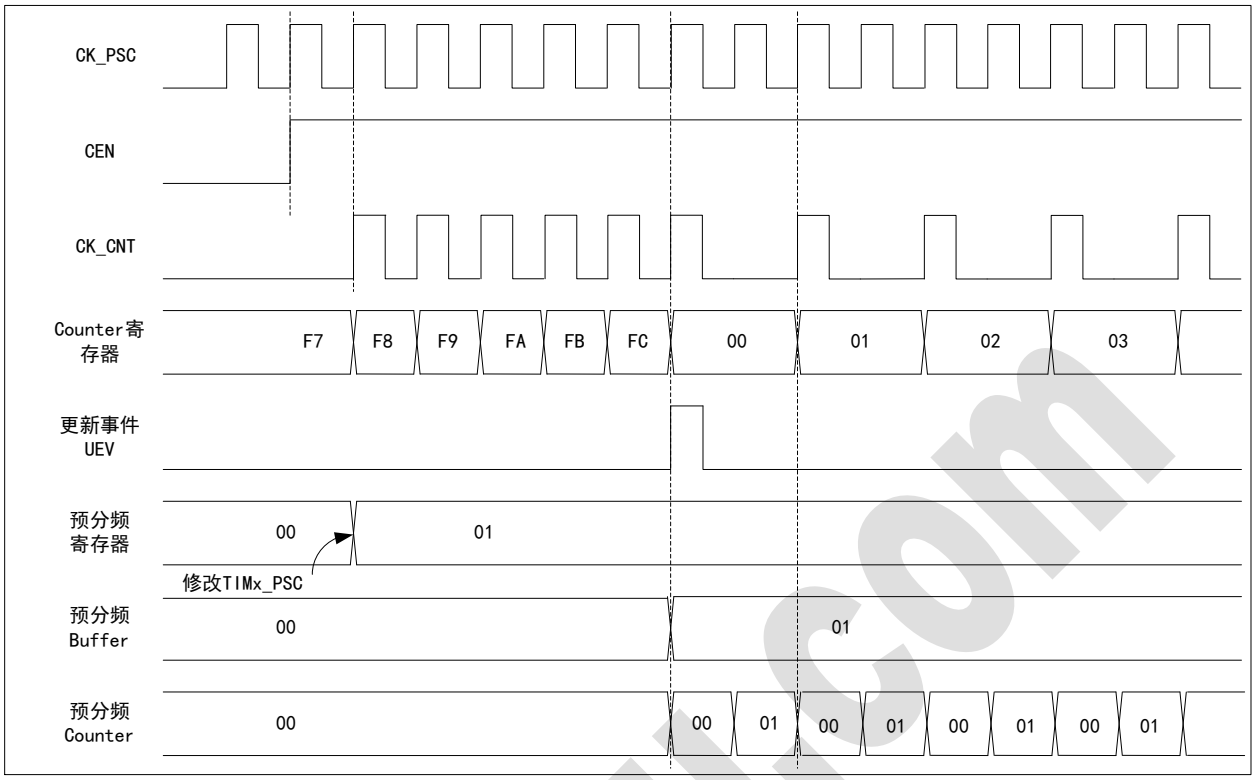
注：设置 TIMx_CR 寄存器的 CEN 位，一个时钟周期后，计数器才开始计数。

预分频

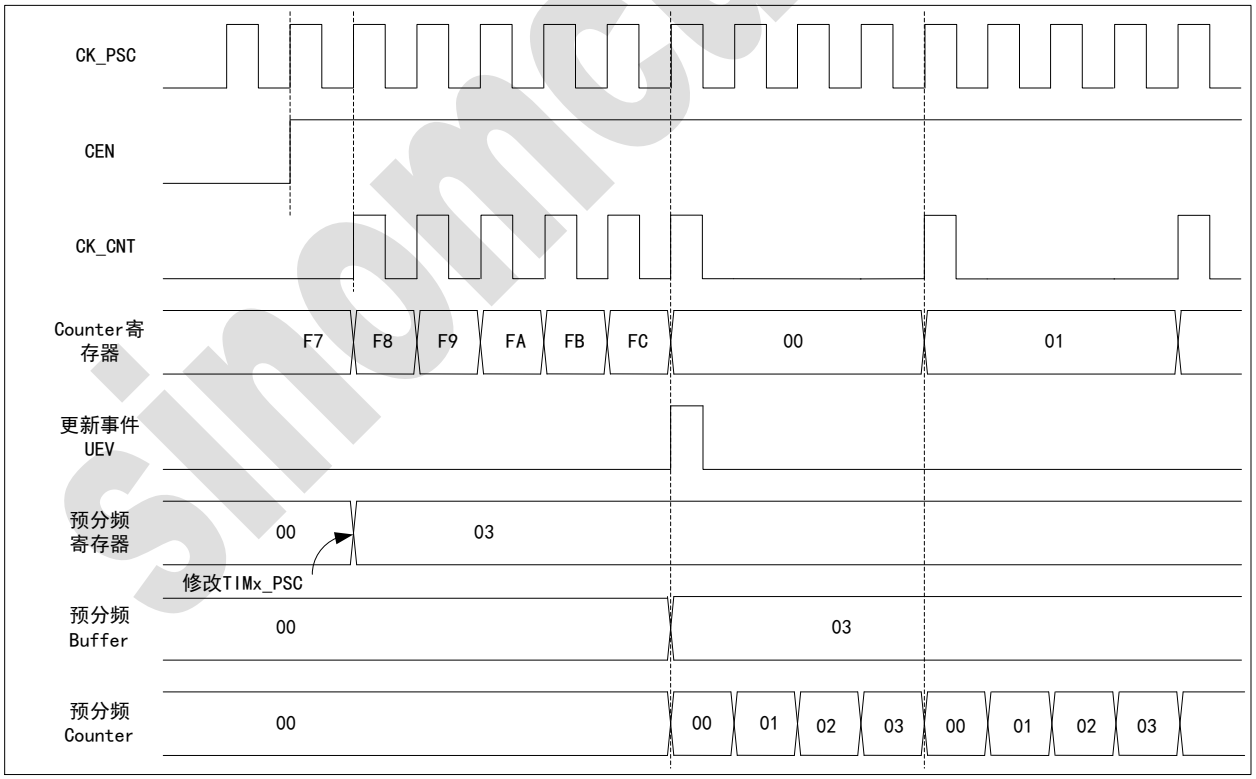
预分频器支持计数器时钟 1~65536 之间任意分频，基于 1 个 16 位的计数器，由 TIMx_PSC 寄存器中的 16 位寄存器控制。此寄存器内部带有缓存器，支持运行时修改；新修改的预分频值在下一次的更新事件（update event）发生时生效。

下图给出运行时更改预分频值，计数器行为。

计数器时序图：预分频修改，从 1 到 2



计数器时序图：预分频修改，从 1 到 4



13.3.2 计数器模式

向上计数模式

设置 TIMx_CR1 寄存器中的 CMS=00 且 DIR=0，执行计数器向上计数。

在向上计数模式中，计数器从 0 计数到自动重载值（TIMx_ARR 计数器的值 ），然后重新从 0 开始计数并且产生一个计数器溢出事件（overflow）。

如果使用了重复次数计数器（repetition counter）功能，对溢出重载次数进行向上计数，达到设置的重复次数(TIMx_RCR) 时，才产生更新事件 (UEV)；否则每次计数器溢出时都会产生更新事件。

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

通过软件置位 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新缓存寄存器。在 UDIS 位被清‘0’之前，将不产生更新事件。但是，在应该产生更新事件时，计数器会被清‘0’，同时预分频器的计数器也被清 0(但预分频器的数值不变)。

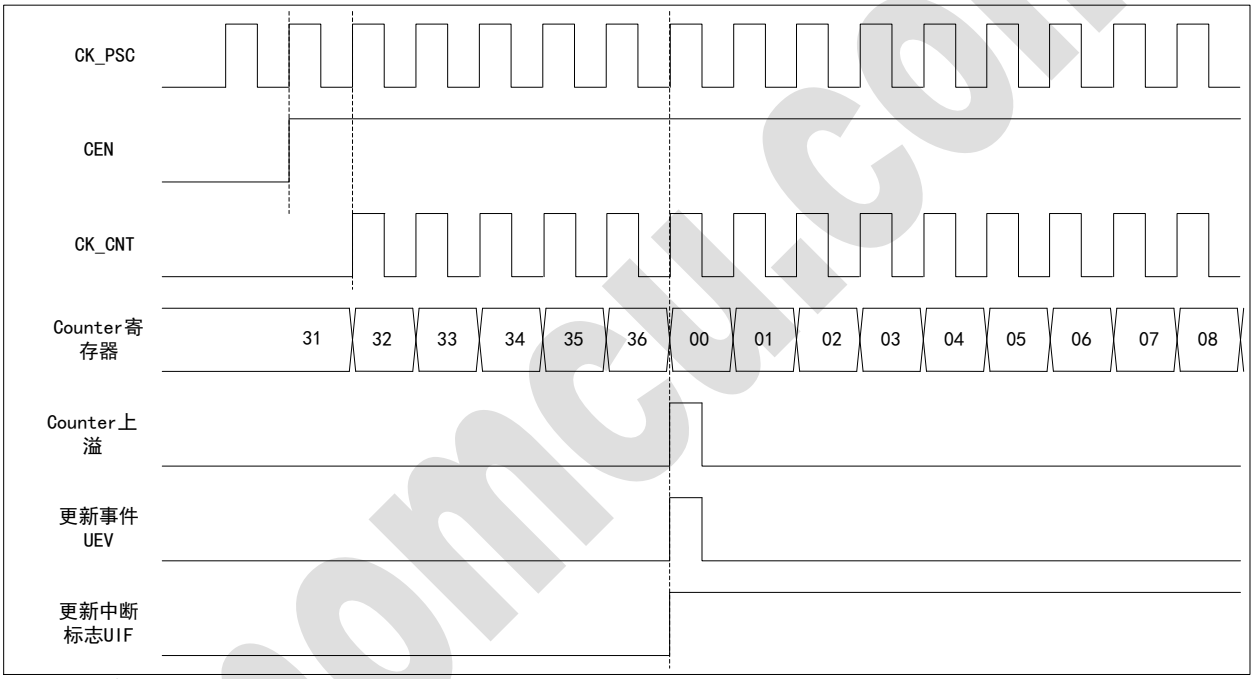
此外，如果置位 TIMx_CR1 寄存器中的 URS 位（update request selection），置位 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志（即不产生中断或 DMA 请求）。这是为了避免在发生捕获事件（capture event）清除计数器时，同时产生更新和捕获两个中断。

当发生一个更新事件时，所有的寄存器都被更新，同时（依据 URS 位）置位更新标志位（TIMx_SR 寄存器中的 UIF 位）：

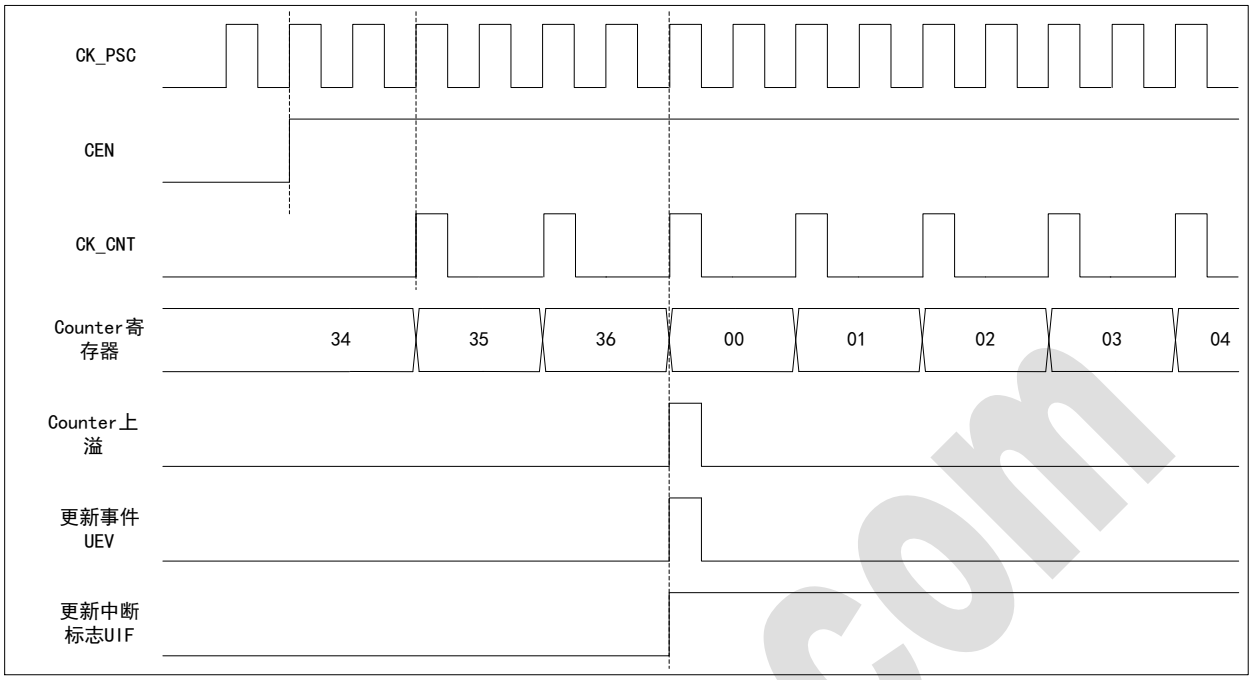
- ✧ 重复次数计数器被重新加载为 TIMx_RCR 寄存器的值
- ✧ 预分频器的缓冲区被写入预装载寄存器的值 (TIMx_PSC 寄存器的值)
- ✧ 自动重载缓存寄存器更新为预装载寄存器的值 (TIMx_ARR)

举例如下：TIMx_ARR=0x36 时，计数器在不同时钟频率下计数器行为。

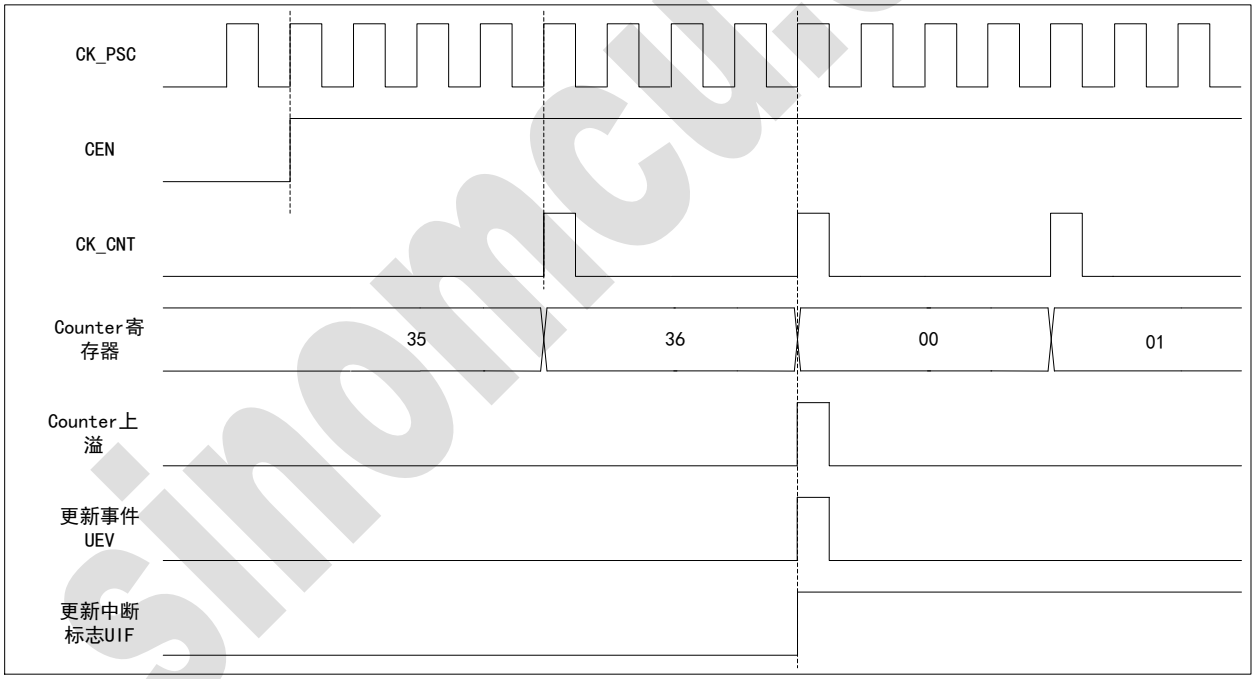
计数器时序图：预分频因子=1



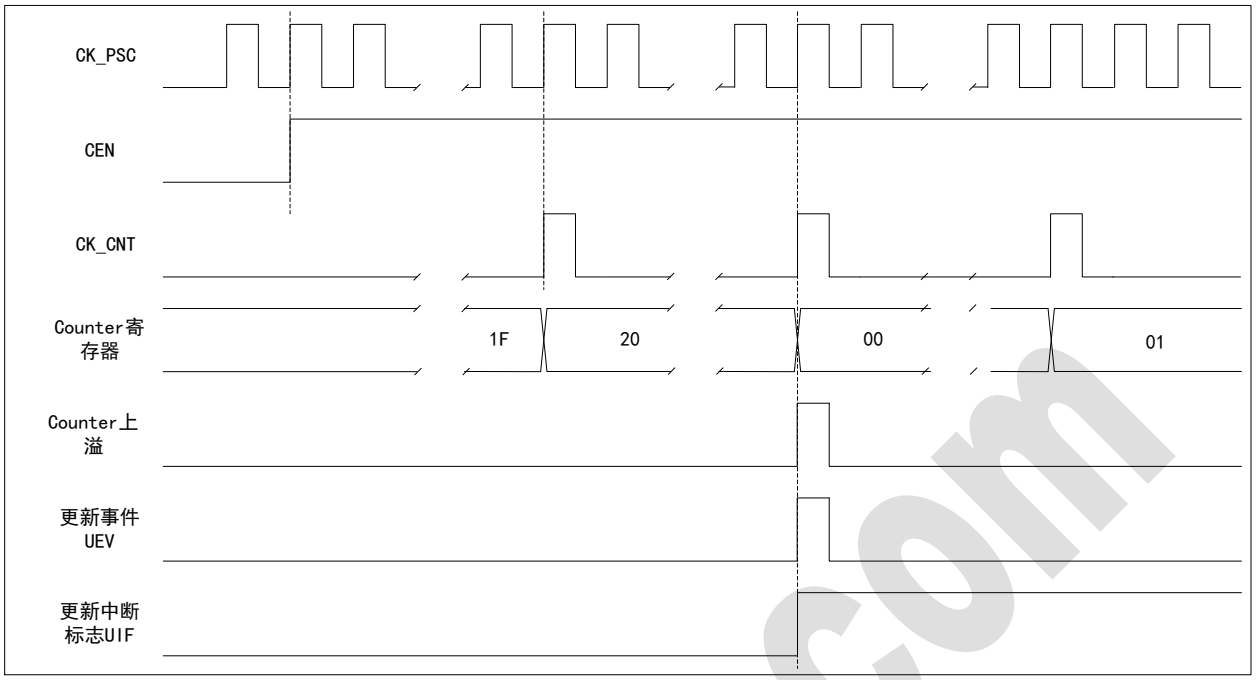
计数器时序图：预分频因子=2



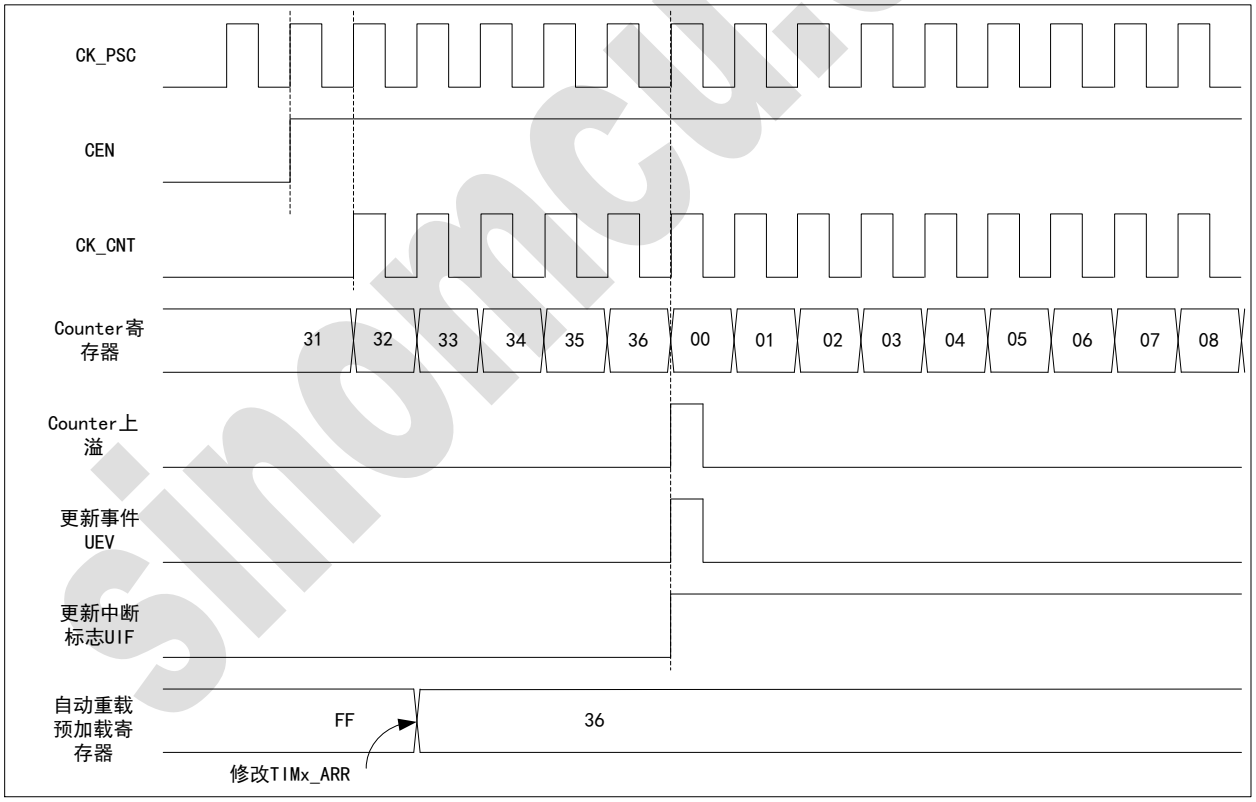
计数器时序图：预分频因子=4



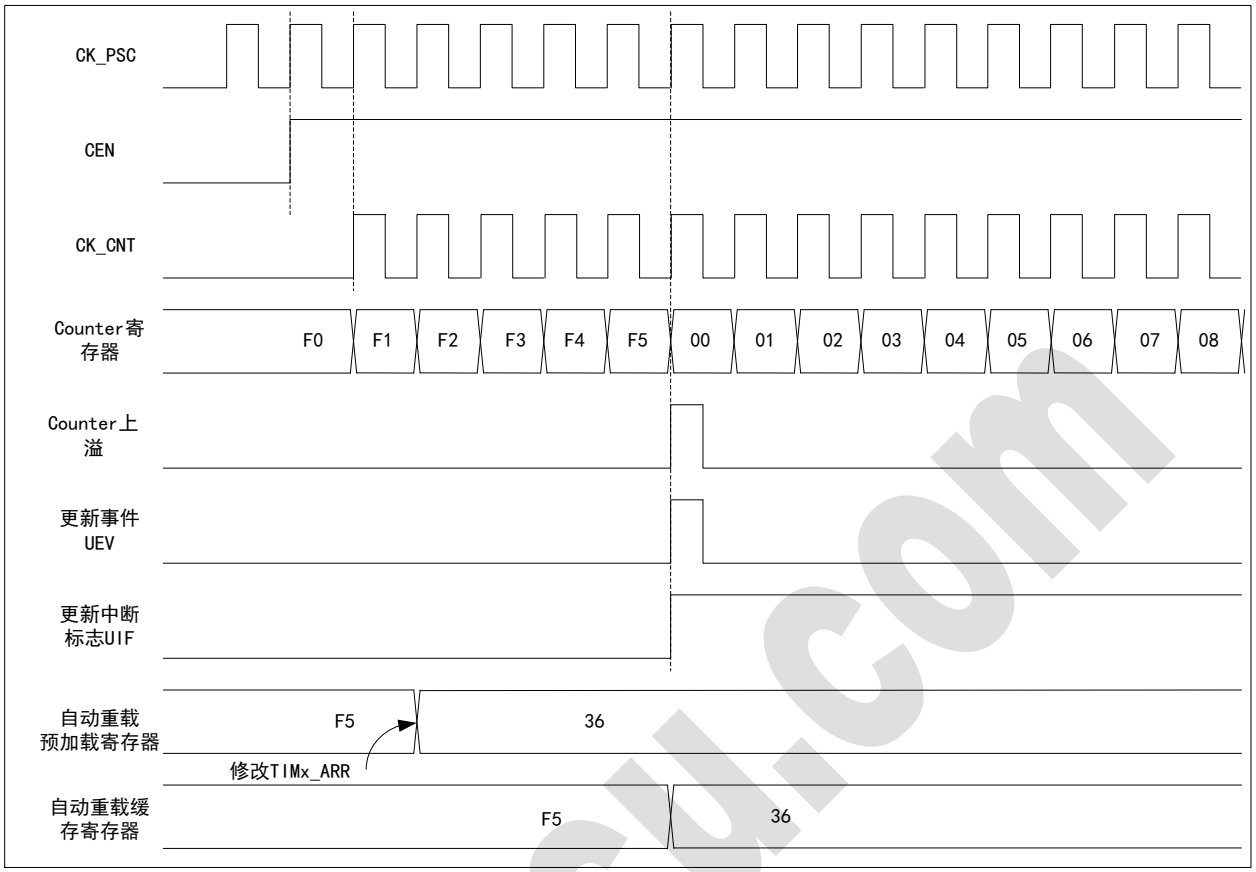
计数器时序图：预分频因子=N



计数器时序图：预分频因子=1，ARPE=0（TIMx_ARR 没有预装载）



计数器时序图：预分频因子=1，ARPE=1（TIMx_ARR 已预装载）



向下计数模式

设置 TIMx_CR1 寄存器中的 CMS=00 且 DIR=1，执行计数器向下计数。

在向下计数模式中，计数器从自动重载的值（TIMx_ARR 寄存器的值）开始向下计数到 0，然后从自动重载的值重新开始并且产生一个计数器向下溢出事件（underflow）。

如果使用了重复次数计数器（repetition counter）功能，对溢出重载次数进行向上计数，达到设置的重复次数（TIMx_RCR）时，才产生更新事件（UEV）；否则每次计数器溢出时都会产生更新事件。

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

通过软件置位 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新缓存寄存器。在 UDIS 位被清‘0’之前，将不产生更新事件。但是，在应该产生更新事件时，计数器会被清‘0’，同时预分频器的计数器也被清 0（但预分频器的数值不变）。

此外，如果置位 TIMx_CR1 寄存器中的 URS 位（update request selection），置位 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志（即不产生中断或 DMA 请求）。这是为了避免在发生捕获事件（capture event）清除计数器时，同时产生更新和捕获两个中断。

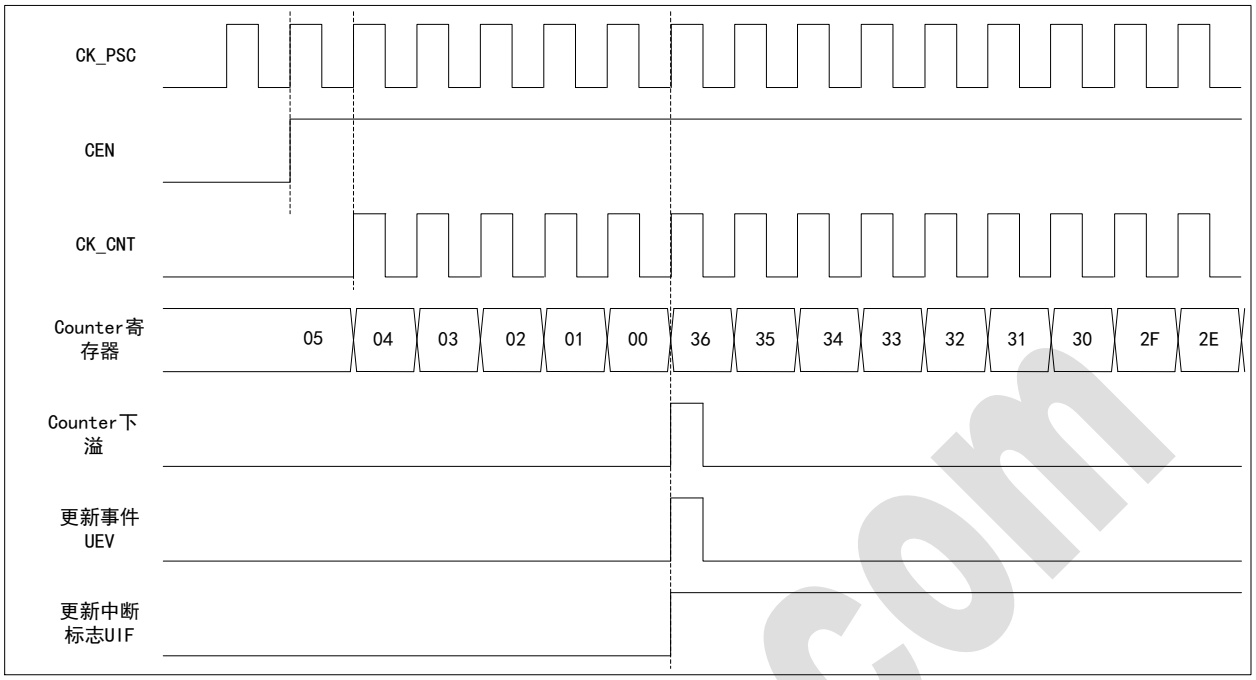
当发生一个更新事件时，所有的寄存器都被更新，同时（依据 URS 位）置位更新标志位（TIMx_SR 寄存器中的 UIF 位）：

- ✧ 重复次数计数器被重新加载为 TIMx_RCR 寄存器的值
- ✧ 预分频器的缓冲区被写入预装载寄存器的值（TIMx_PSC 寄存器的值）
- ✧ 自动重载缓存寄存器更新为预装载寄存器的值（TIMx_ARR）

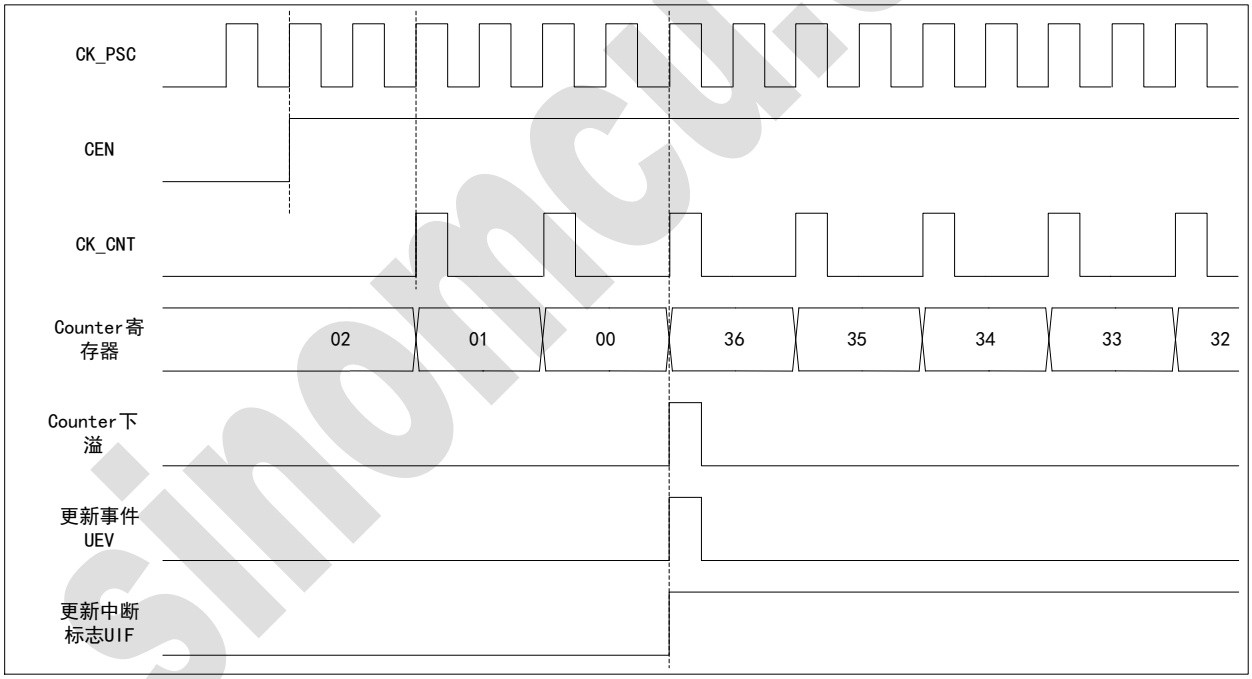
注：若在计数器重载之前自动重载值被更新，此时在下一个周期时才是预期的值。

举例如下：TIMx_ARR=0x36 时，计数器在不同时钟频率下计数器行为。

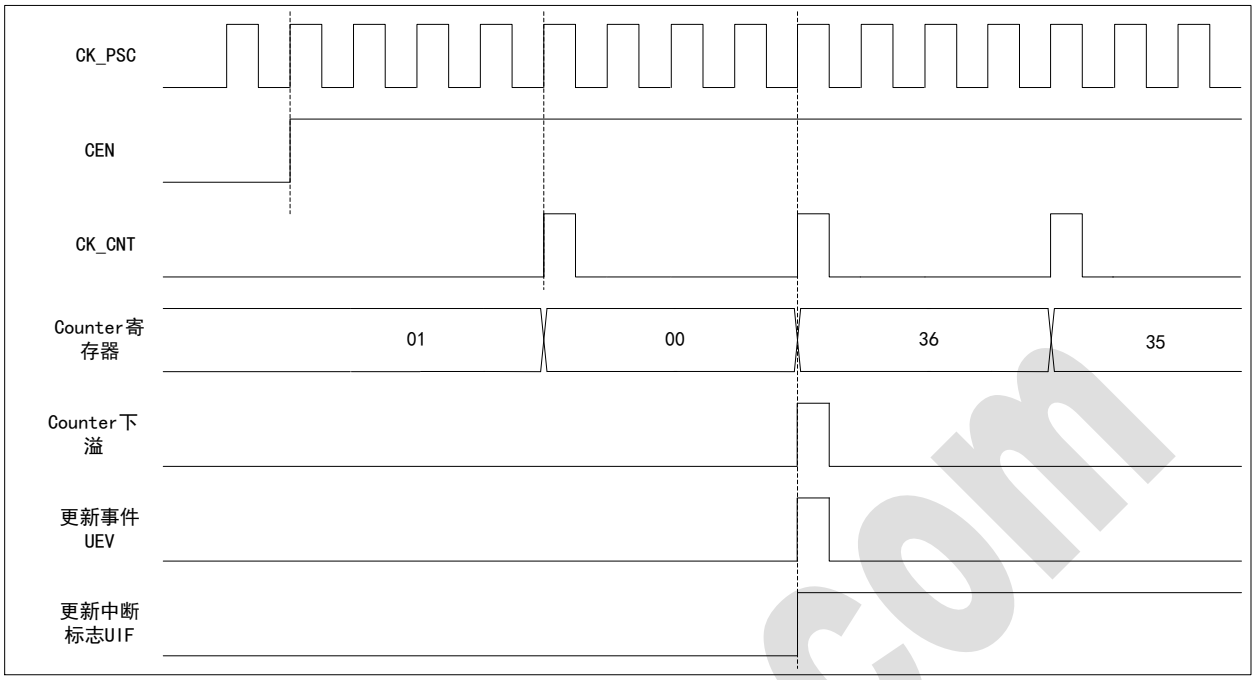
计数器时序图：预分频因子=1



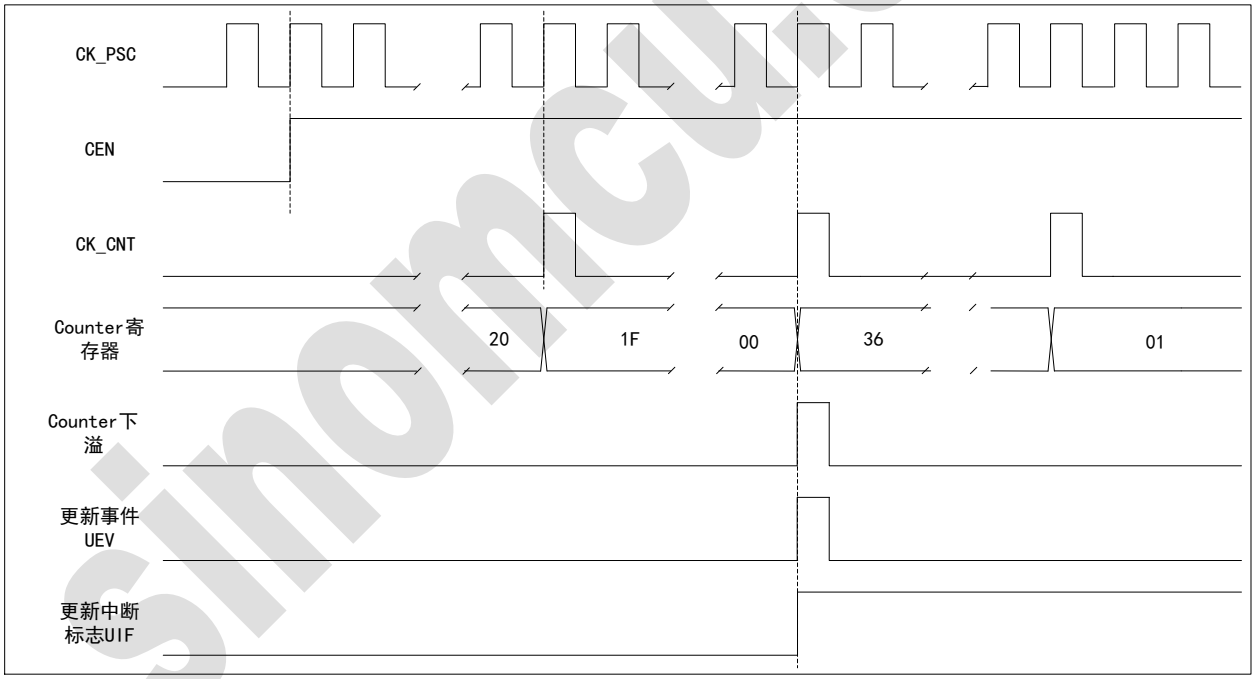
计数器时序图：预分频因子=2



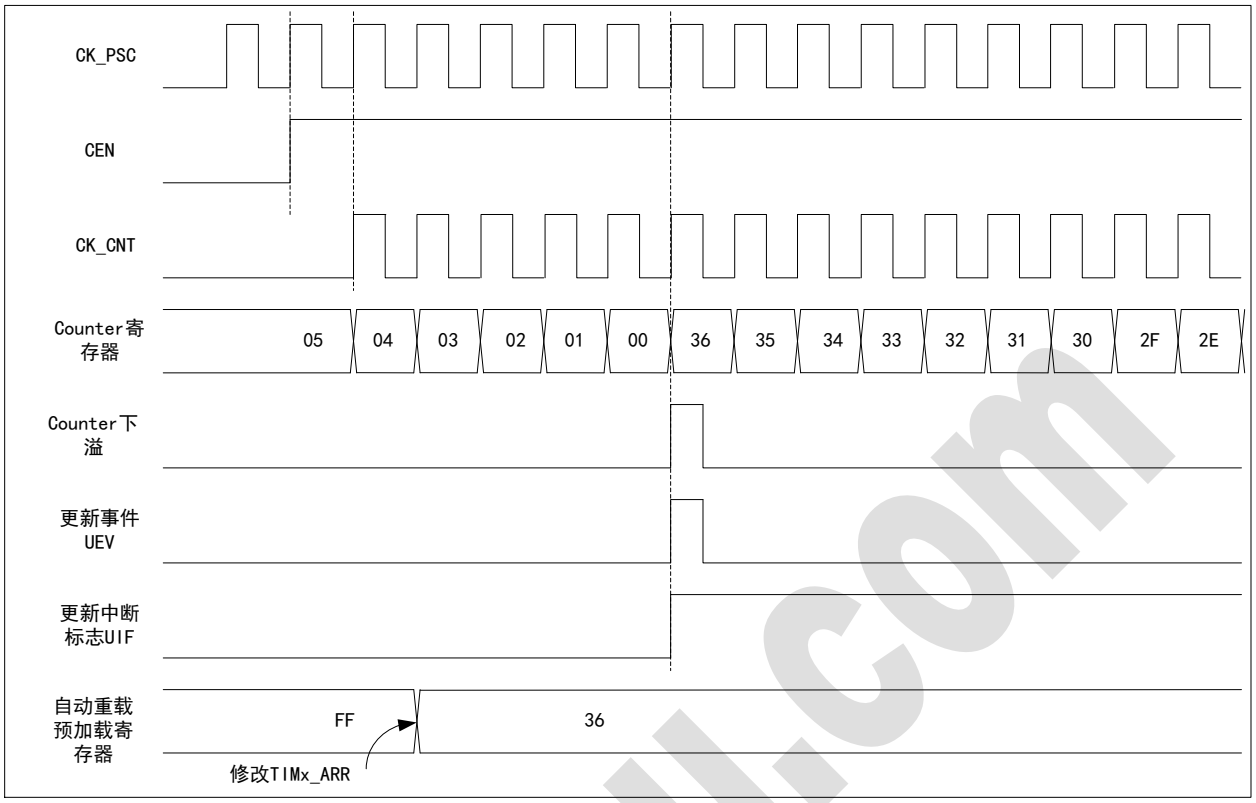
计数器时序图：预分频因子=4



计数器时序图：预分频因子=N



计数器时序图：预分频因子=1,重复次数计数器未使用



中心对齐模式（向上/向下计数）

在中央对齐模式,计数器从 0 开始计数到自动重载的值(TIMx_ARR 寄存器)－1,产生一个计数器溢出事件(overflow),然后从自动重载值向下计数到 1 并产生一个计数器下溢事件（underflow）；然后再从 0 开始重新计数。

当 TIMx_CR1 的 CMS 位不为"00"时，使能中央对齐模式。已配置为输出的通道，其输出比较中断标志在以下情况下被置位：计数器向下计数（中央对齐模式 1, CMS = "01"），计数器向上计数（中央对齐模式 2, CMS = "10"），计数器向下和向上计数（中央对齐模式 3, CMS = "11"）。

在此模式下，不能写入 TIMx_CR1 中的 DIR 方向位，它由硬件更新并指示当前的计数方向。

更新事件可以在每个计数器上溢和每个计数器下溢时产生；也可以通过(软件或者使用从模式控制器)置位 TIMx_EGR 寄存器中的 UG 位产生更新事件。在这种情况下，计数器以及预分频器的计数器将从 0 重新开始计数。

通过软件置位 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新缓存寄存器。在 UDIS 位被清‘0’之前，将不产生更新事件。**然而，计数器仍会根据当前自动重载的值，继续向上或向下计数。**

此外，如果置位 TIMx_CR1 寄存器中的 URS 位（update request selection），置位 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志（即不产生中断或 DMA 请求）。这是为了避免在发生捕获事件（capture event）清除计数器时，同时产生更新和捕获两个中断。

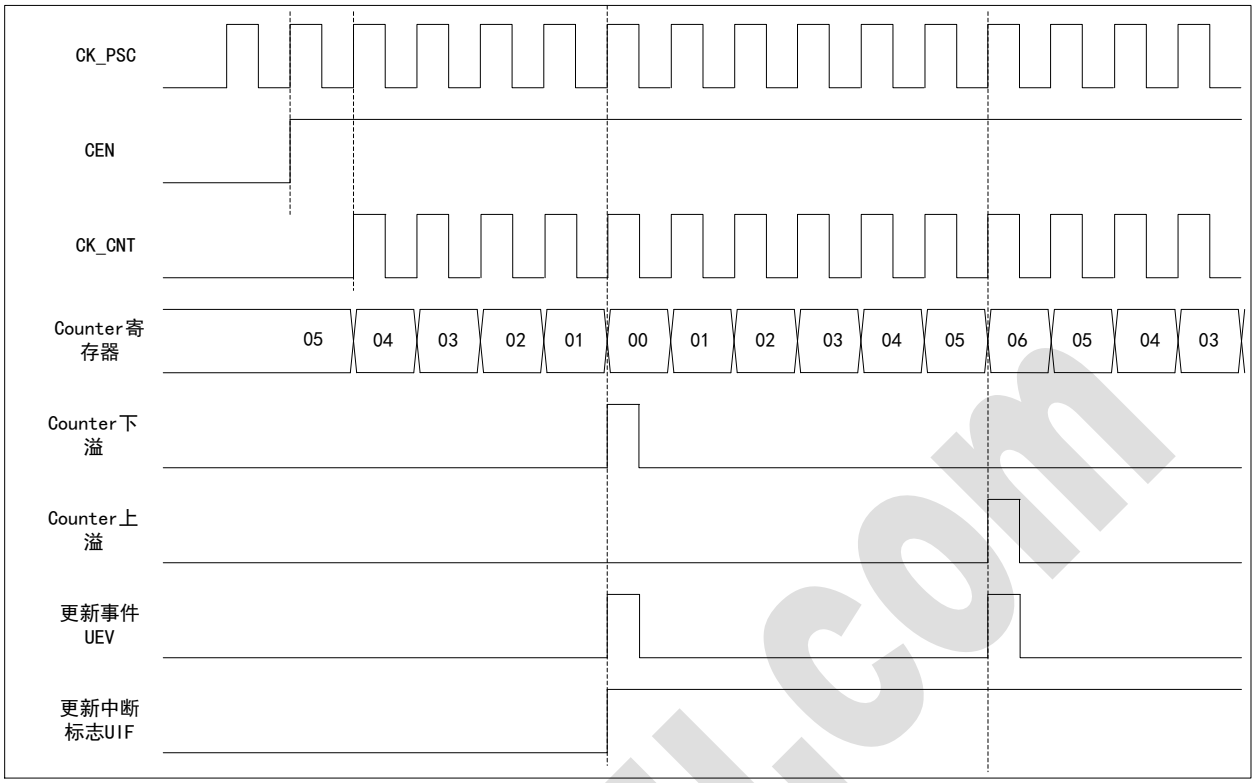
当发生一个更新事件时,所有的寄存器都被更新,同时(依据 URS 位)置位更新标志位(TIMx_SR 寄存器中的 UIF 位)：

- ✧ 重复次数计数器被重新加载为 TIMx_RCR 寄存器的值
- ✧ 预分频器的缓冲区被写入预装载寄存器的值 (TIMx_PSC 寄存器的值)
- ✧ 自动重载缓存寄存器更新为预装载寄存器的值 (TIMx_ARR)

注：若更新事件源为计数器上溢（overflow），在计数器重载之前自动重载值被更新，此时在下一个周期时才是预期的值（计数器被加载为新值）。

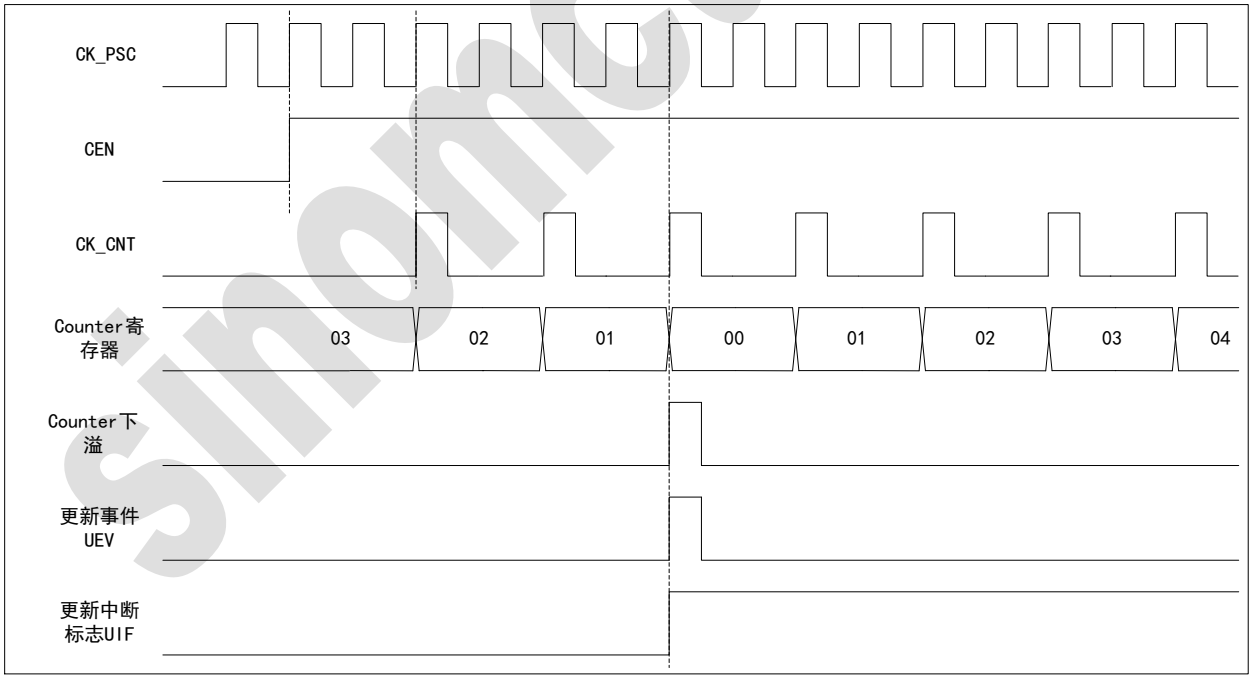
举例如下：计数器在不同时钟频率下计数器行为。

计数器时序图：预分频因子=1，TIMx_ARR=0x06

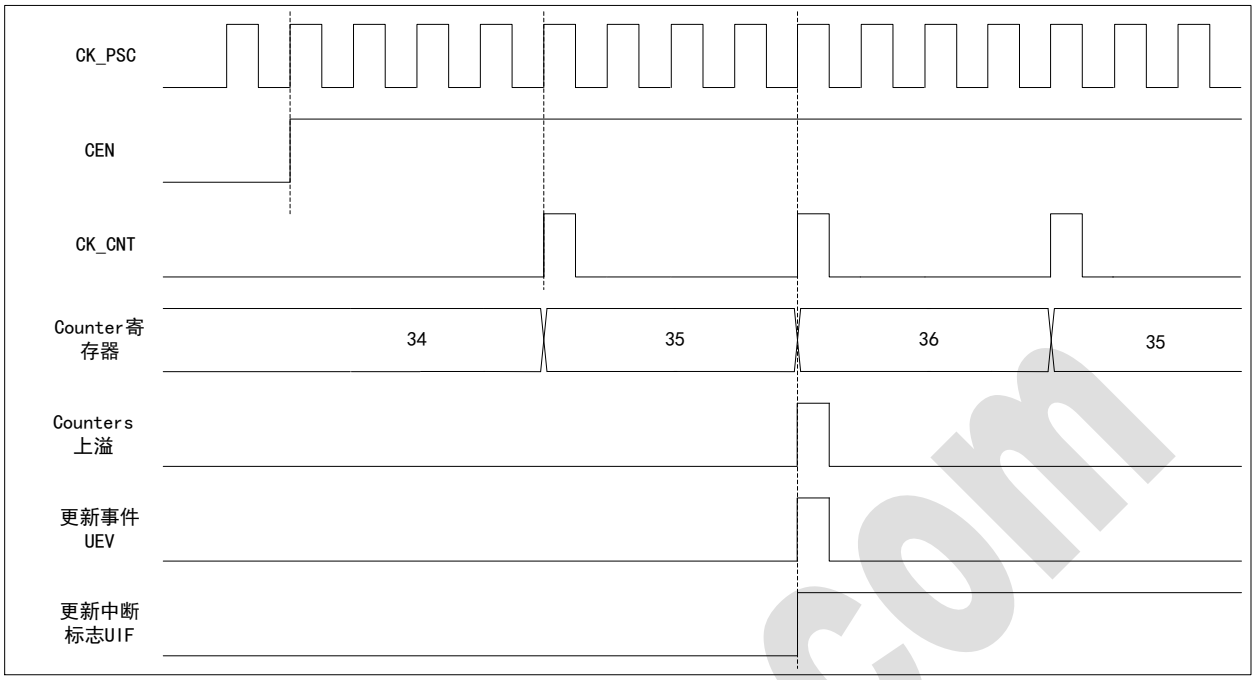


注：上图为中心对齐模式 1

计数器时序图：预分频因子=2，TIMx_ARR=0x36

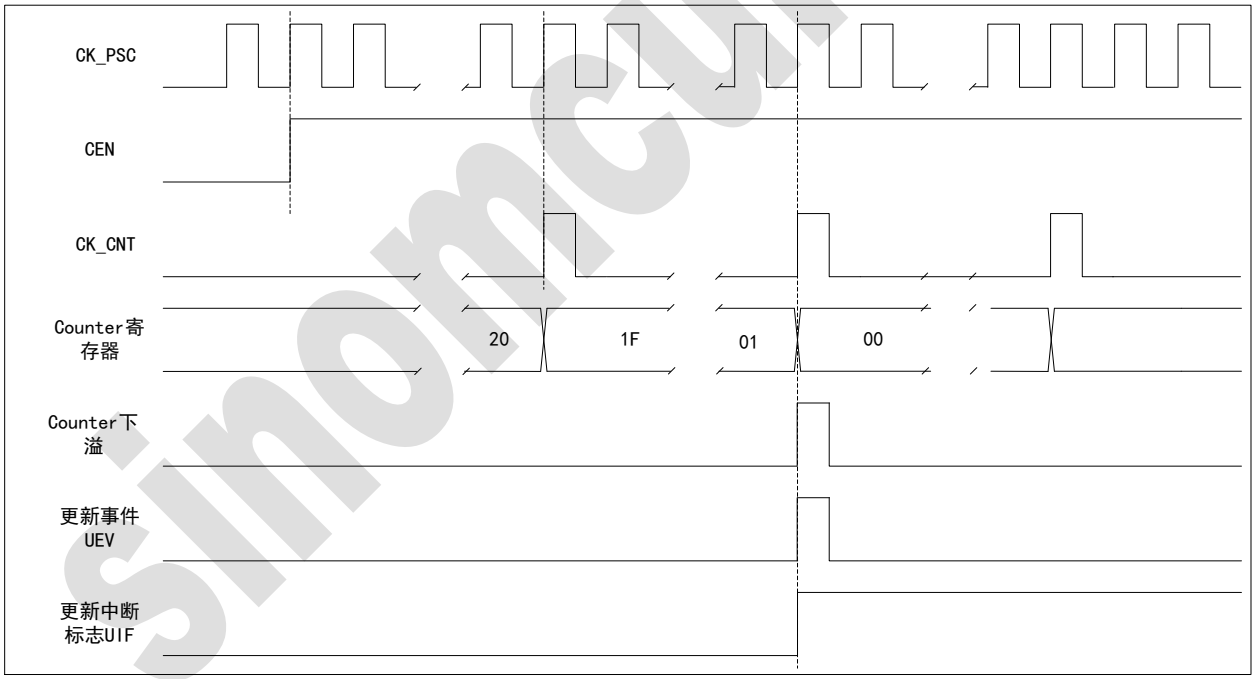


计数器时序图：预分频因子=4，TIMx_ARR=0x36

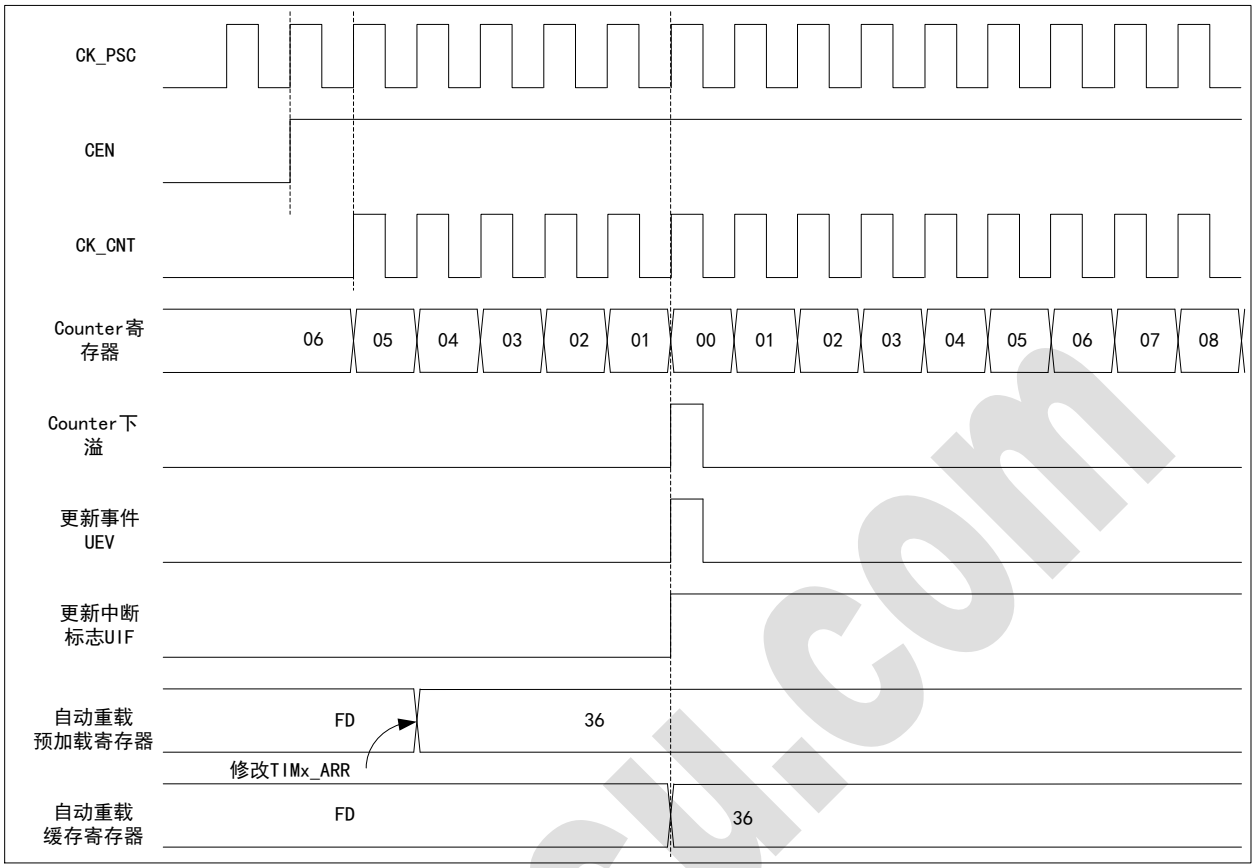


注：上图为中心对齐模式 2 或模式 3

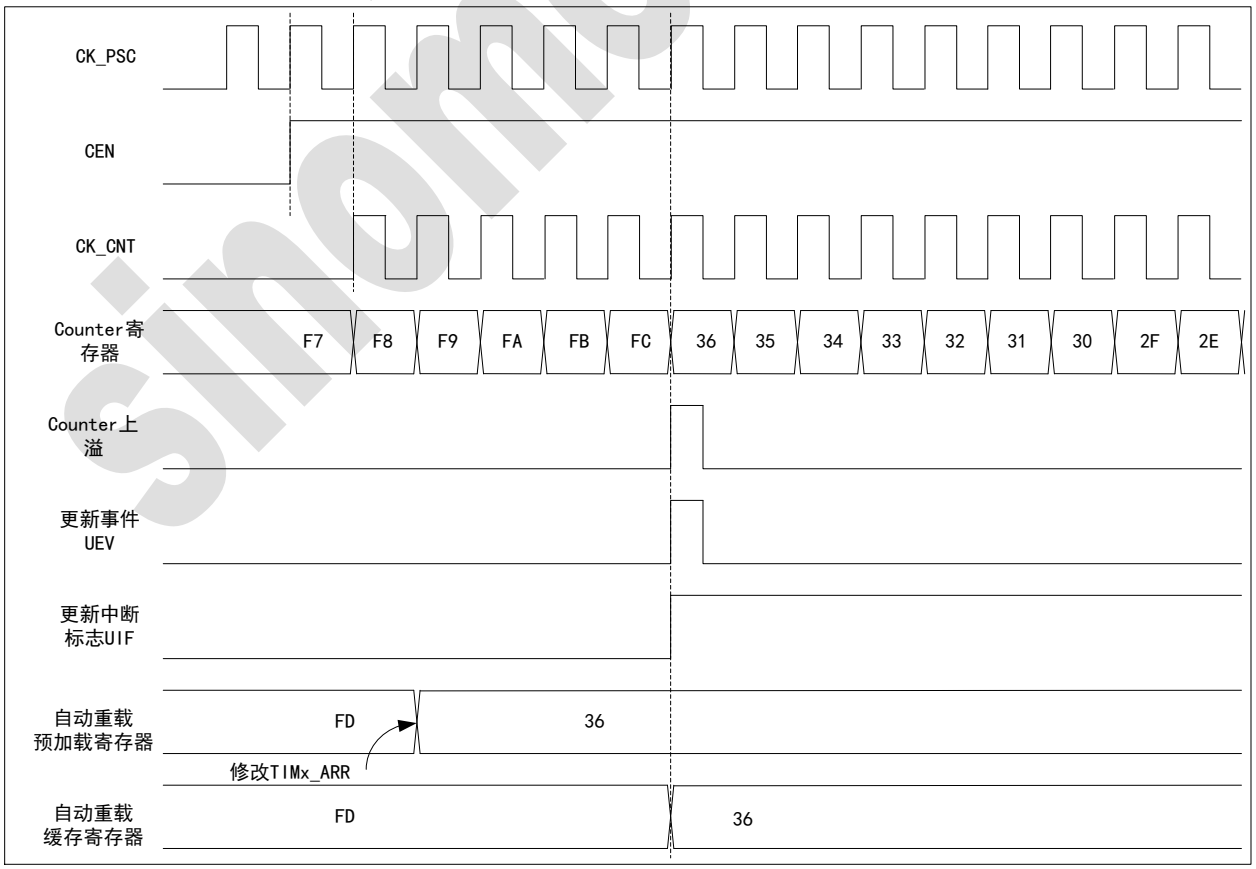
计数器时序图：预分频因子=N



计数器时序图：预分频因子=1，ARPE=1 的更新事件(计数器下溢)



计数器时序图：预分频因子=1，ARPE=1 的更新事件(计数器上溢)



13.3.3 重复次数计数器（repitition counter）

13.3.1 章节<时基单元>解释了计数器上溢/下溢时是如何产生更新事件(UEV)的，然而事实上它只能在重复计数器归 0 时产生。这个特性对产生 PWM 信号非常有用。

这意味着在每 N 次计数上溢或下溢时，数据从预装载寄存器传送至缓存寄存器(TIMx_ARR 自动重载寄存器，TIMx_PSC 预分频寄存器，还有在比较模式下的捕获/比较寄存器 TIMx_CCRx)，N 为 TIMx_RCR 重复计数寄存器中的值。

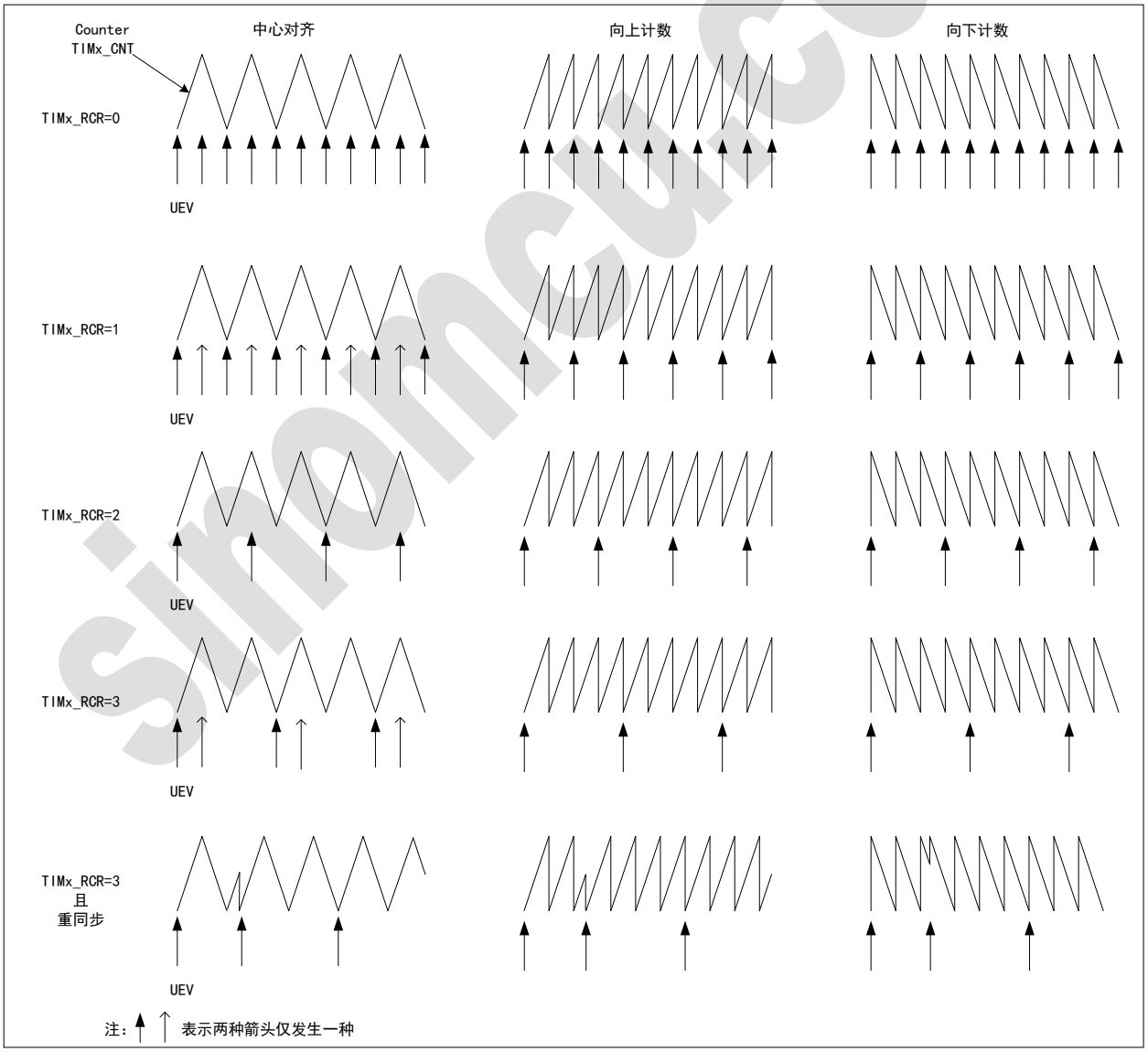
重复计数器在下述任一条件成立时递减：

- ✧ 向上计数模式下每次计数器溢出时
- ✧ 向下计数模式下每次计数器下溢时
- ✧ 中央对齐模式下每次上溢和每次下溢时。虽然这样限制了最大重复次数为 128 个 PWM 周期， 但它能够在每个 PWM 周期 2 次更新占空比。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 2xTck。

重复计数器是自动重载的，重复次数是由 TIMx_RCR 寄存器的值定义。当更新事件由软件产生（通过设置 TIMx_EGR 中的 UG 位）或者通过硬件（从模式控制器）产生，则无论重复计数器的值是多少，立即发生更新事件，并且 TIMx_RCR 寄存器中的内容被重载入到重复计数器。

在中央对齐模式下，如果 RCR 的值是奇数，更新事件只会在上溢或下溢时两者之一发生，这取决于 RCR 寄存器被写入时间和计数器启动时间。如果 RCR 在计数器启动之前被写入，则只在下溢时发生 UEV 事件。如果 RCR 在计数器启动之后被写入，则只在上溢时发生 UEV 事件。例如 RCR=3，则 UEV 事件在每第 4 个上溢或下溢产生，这取决于 RCR 被写入时间。

RCR 更新举例：不同模式及不同 TIMx_RCR 寄存器设置



注：图中 RCR=1 或 3 中，两种箭头表示有两种可能产生更新事件的情况

13.3.4 时钟源

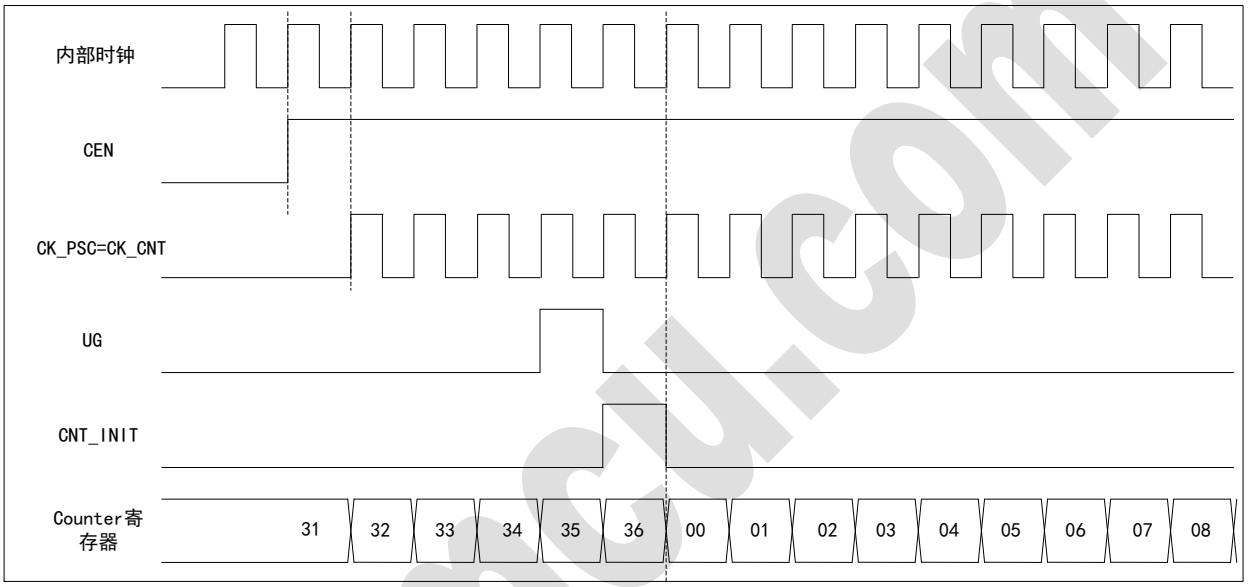
计数器时钟源包括：

- ✧ 内部时钟 CK_INT
- ✧ 外部时钟模式 1：外部输入引脚
- ✧ 外部时钟模式 2：外部触发输入 ETR
- ✧ 内部触发输入 ITRx：使用一个定时器作为另一个定时器的预分频器。举例：可以配置 Timer1 作为 Timer2 的预分频器。详情参考<定时器同步>章节。

内部时钟源（CK_INT）

如果从模式控制器禁用（SMS=000），则 CEN、DIR（TIMx_CR1 寄存器）和 UG 位（TIMx_ EGR 寄存器）为实际控制位，并且只能被软件修改（除了 UG 位保持自动被清除）。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。

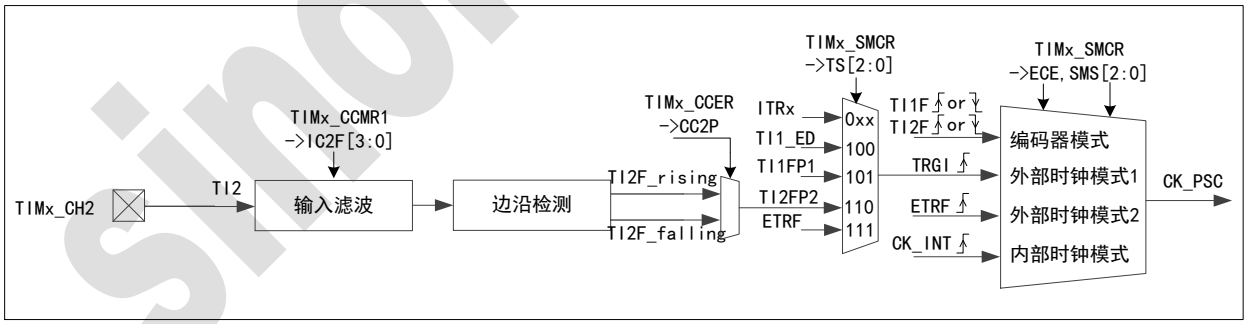
计数器时序图：内部时钟，预分频因子=1，正常模式



外部时钟源模式 1

设置 TIMx_SMCR 寄存器的 SMS=111，此模式被选中。计数器对选定输入端的每个上升沿或下降沿计数。

TI2 外部时钟源联接示例框图



举例：要配置向上计数器在 TI2 输入端的上升沿计数，步骤如下：

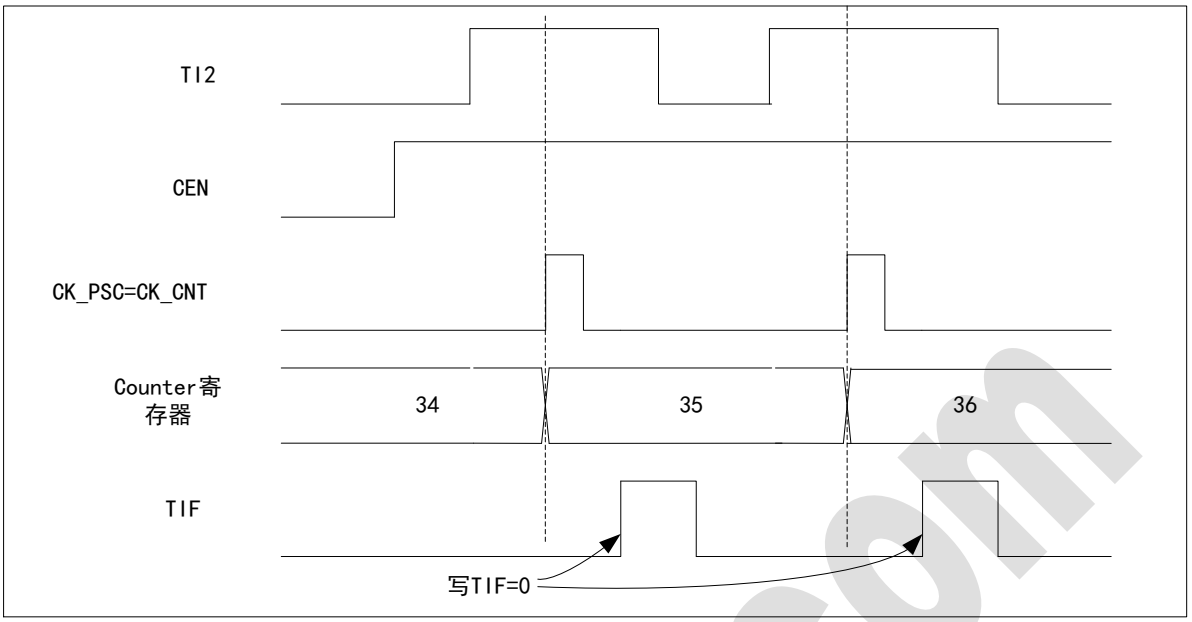
1、 写 TIMx_CCMR1 寄存器的 CC2S=01，配置通道 2 检测 TI2 输入的上升沿；
2、 写 TIMx_CCMR1 寄存器的 IC2F[3:0]，选择输入滤波器时间（如果不需要滤波器，保持 IC2F[3:0]=0000）；
3、 写 TIMx_CCER 寄存器的 CC2P=0，选定上升沿极性；
4、 写 TIMx_SMCR 寄存器的 SMS[2:0]=111，选择定时器外部时钟模式 1；
5、 写 TIMx_SMCR 寄存器中的 TS[2:0]=110，选定 TI2 作为触发输入源；
6、 写 TIMx_CR1 寄存器的 CEN=1，启动计数器。

注： 捕获预分频器不用作触发，所以不需要对它进行配置。

当发生一次 TI2 上升沿，计数器计数一次，并置位 TIF 位。

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。

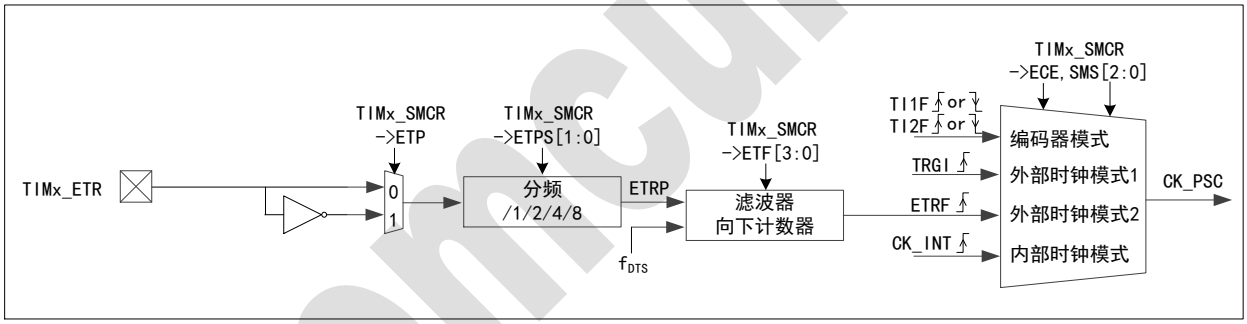
控制时序：外部时钟模式 1



外部时钟模式 2

设置 TIMx_SMCR 寄存器的 ECE=1，此模式被选中。计数器对外部触发 ETR 的每个上升沿或下降沿计数。

ETR 外部触发源联接示例框图



举例：要配置向上计数器在 ETR 的每 2 个上升沿计数一次，步骤如下：

1、 本例中不需要滤波，写 TIMx_SMCR 寄存器中的 ETF[3:0]=0000；

2、 写 TIMx_SMCR 寄存器中的 ETPS[1:0]=01，设置预分频器因子=2；

3、 写 TIMx_SMCR 寄存器中的 ETP=0，选择检测 ETR 的上升沿；

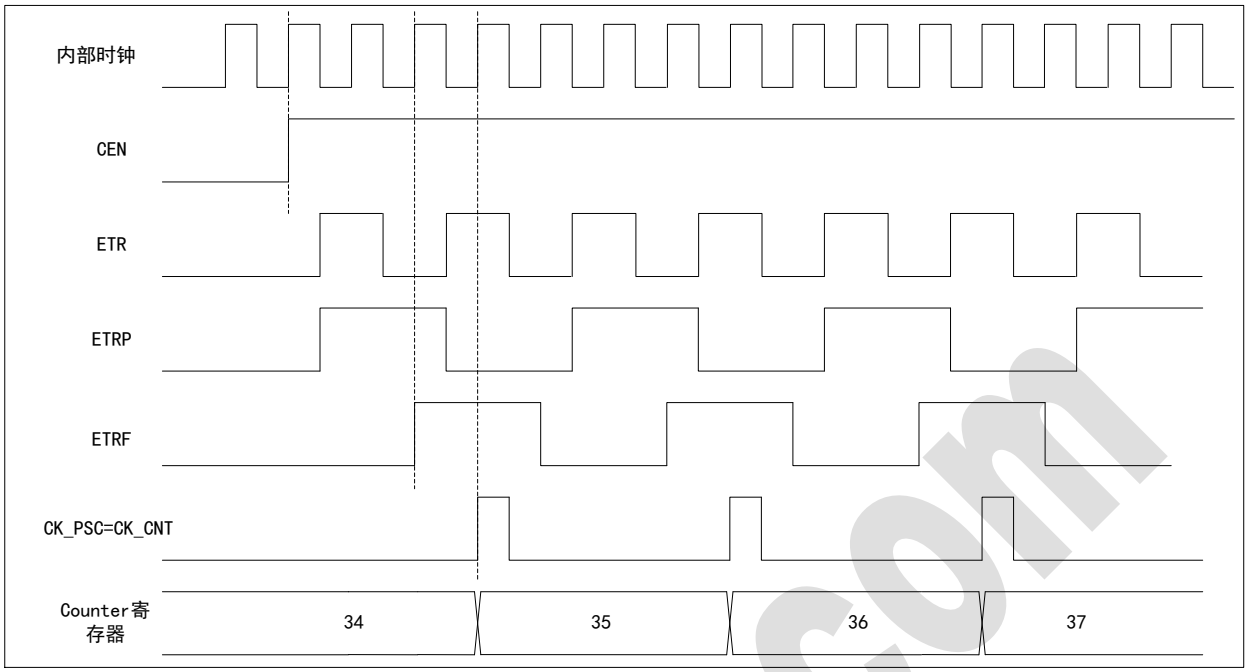
4、 写 TIMx_SMCR 寄存器中的 ECE=1，开启外部时钟模式 2；

5、 写 TIMx_CR1 寄存器中的 CEN=1，启动计数器。

计数器在每 2 个 ETR 上升沿计数一次。

ETR 的上升沿和计数器实际时钟之间的延时，取决于在 ETRP 信号端的重新同步电路。

控制时序：外部时钟模式 2



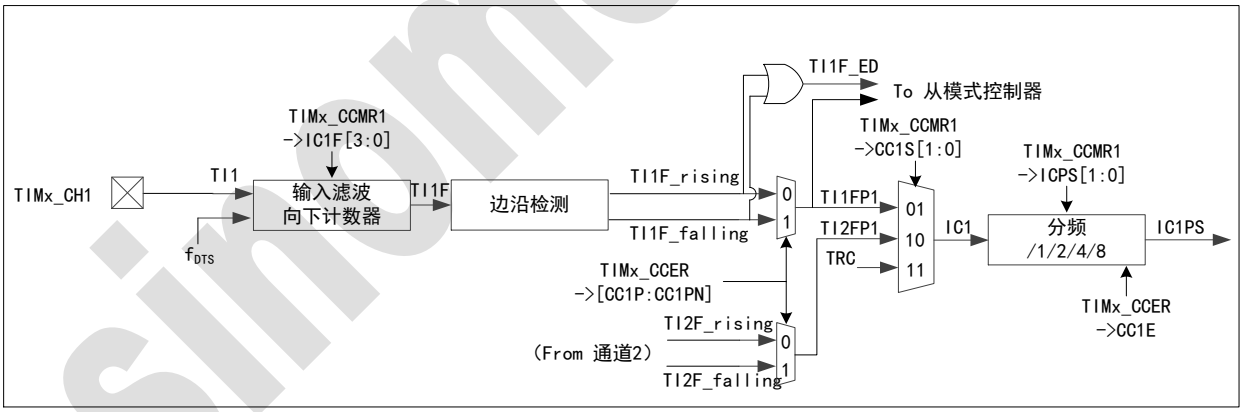
13.3.5 捕获/比较通道

每一个捕获/比较通道都是包括：一个捕获/比较寄存器（包含缓存寄存器），一个用于捕获的输入级（带数字滤波、多路复用选择和预分频器），和一个输出级（带比较器和输出控制）。

输入级

对 TIx 的输入信号进行采样，产生一个滤波后 TIxF 信号，经过带有极性选择的边沿检测器生成 TIxFPx 信号，TIxFPx 可以作为从模式控制器的触发输入或者作为捕获命令。该信号预分频后（ICxPS）进入捕获寄存器。

捕获/比较通道输入级（举例：通道 1）



输出级

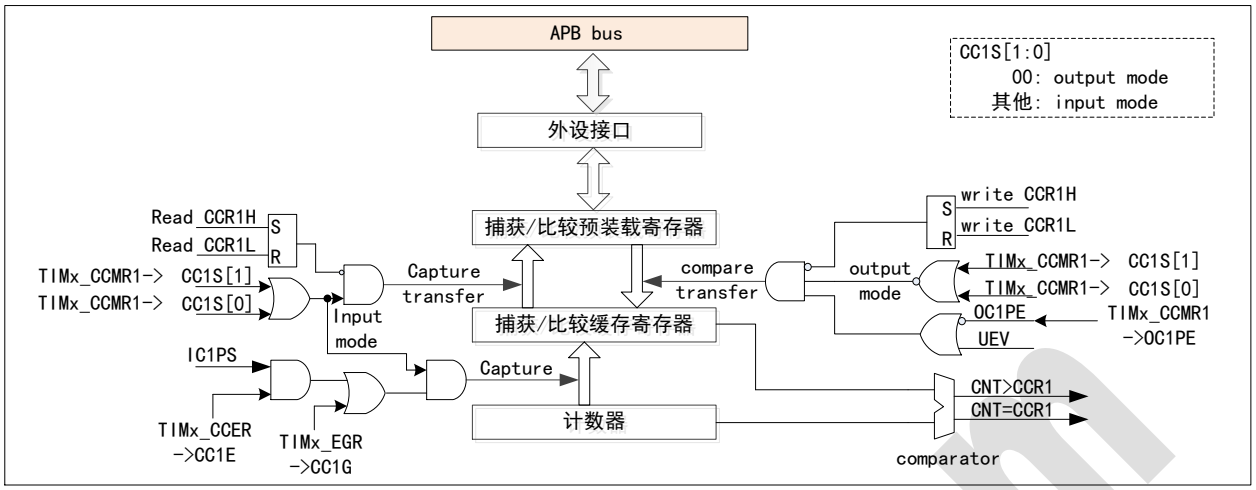
输出部分产生一个中间波形 OCxRef（高有效）作为基准，链的末端决定最终输出信号的极性。

捕获/比较模块由一个预装载寄存器（preload register）和一个缓存寄存器（shadow register）组成。读写过程仅操作预装载寄存器。

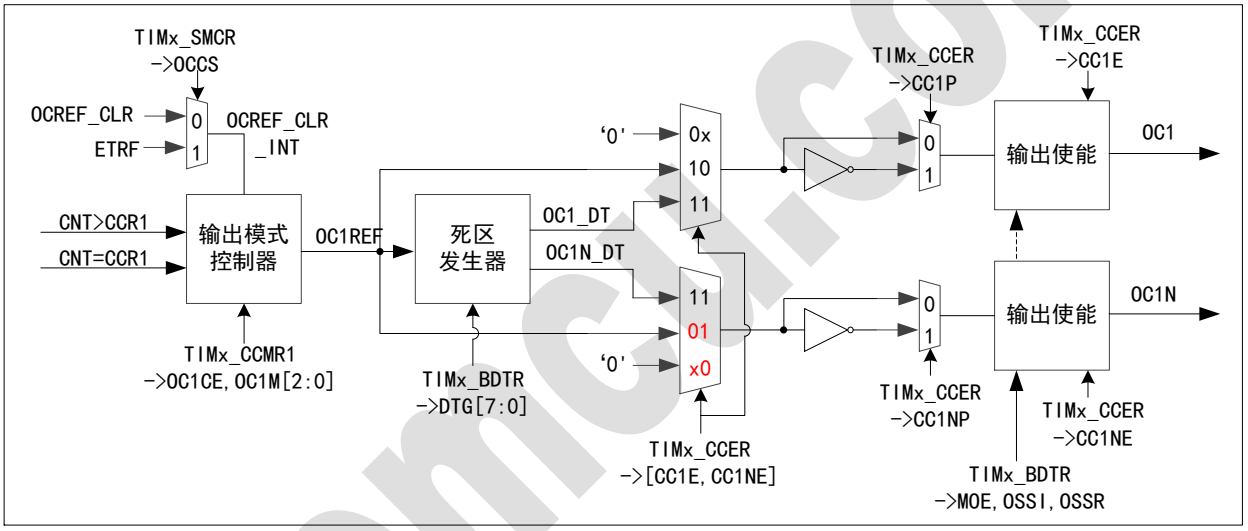
在捕获模式下，捕获发生在缓存寄存器上，然后再复制到预装载寄存器中。

在比较模式下，预装载寄存器的内容被复制到缓存寄存器中，然后缓存寄存器的内容和计数器进行比较。

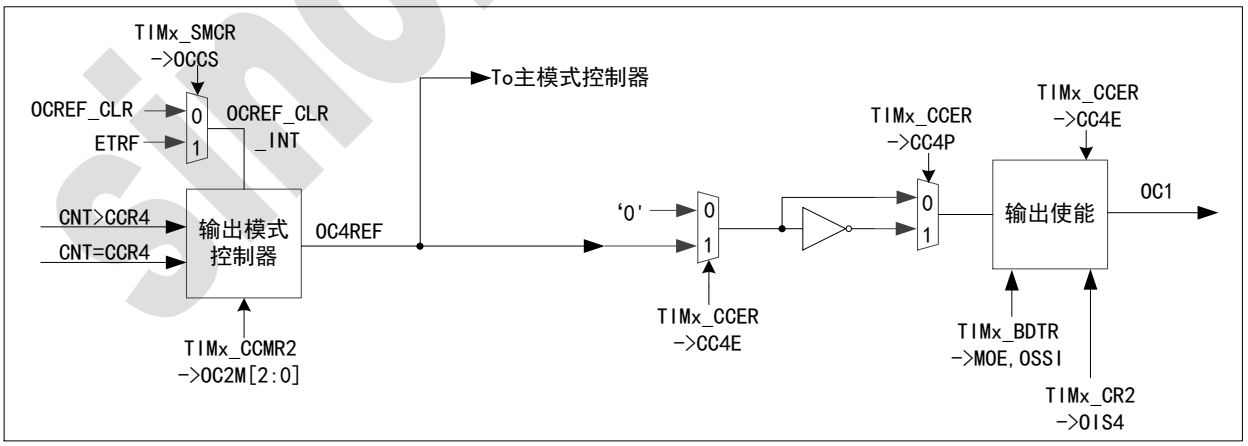
捕获/比较通道主电路



捕获/比较通道输出级（通道 1，通道 2/3 同通道 1）



捕获/比较通道输出级（通道 4）



13.3.6 输入捕获模式

在输入捕获模式下，当检测 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（TIMx_CCRx）中。当发生捕获事件时，相应 CCxIF 标志（TIMx_SR 寄存器）被置 1， 如果使能了中断或者 DMA，则产生一个中断或者一个 DMA 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（TIMx_SR 寄存器）被置 1。软件写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

举例，在 TI1 输入的上升沿时，捕获计数器的值到 TIMx_CCR1 寄存器中，操作步骤如下：

1、 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01（选中 TI1），只要 CC1S 不为‘00’，通道被配置为输入并且 TIMx_CCR1 寄存器变为只读。
2、 根据连接到计数器的输入信号，配置输入滤波器时间（输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位）。举例，当 TI1 翻转时，信号抖动最多 5 个内部时钟， 则须配置滤波器时间大于 5 个时钟周期，TIMx_CCMR1 寄存器中写入 IC1F=0011 配置采样次数 8（以 f_{DTS} 频率采样），通过连续 8 次采样以确认电平变换。
3、 选择 TI1 通道有效沿。写 TIMx_CCER 寄存器中 CC1P=0 和 CC1NP=0（本例中为上升沿）。
4、 配置输入预分频器。在本例中，设置为每个有效的电平转换时发生一次捕获，因此预分频器被禁止（写 TIMx_CCMR1 寄存器的 IC1PS=00）。
5、 写 TIMx_CCER 寄存器的 CC1E=1，允许计数器的值被捕获至捕获寄存器中。
6、 根据需要，置位 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，置位 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

✧ 有效的电平转换发生时，计数器的值被传送至 TIMx_CCR1 寄存器；
✧ CC1IF 标志被置位（中断标志）。当发生至少 2 个连续的捕获时，且 CC1IF 未被清除，CC1OF 也被置位；
✧ 若置位 CC1IE 位，则会产生一个中断。
✧ 若置位 CC1DE 位，则会产生一个 DMA 请求。

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，这是为了避免在读取标志之后和读取数据之前可能发生的捕获溢出。

注： 通过软件设置 TIMx_EGR 寄存器中相应的 CCxG 位，也可以产生输入捕获中断 和/或 DMA 请求。

13.3.7 PWM 输入模式

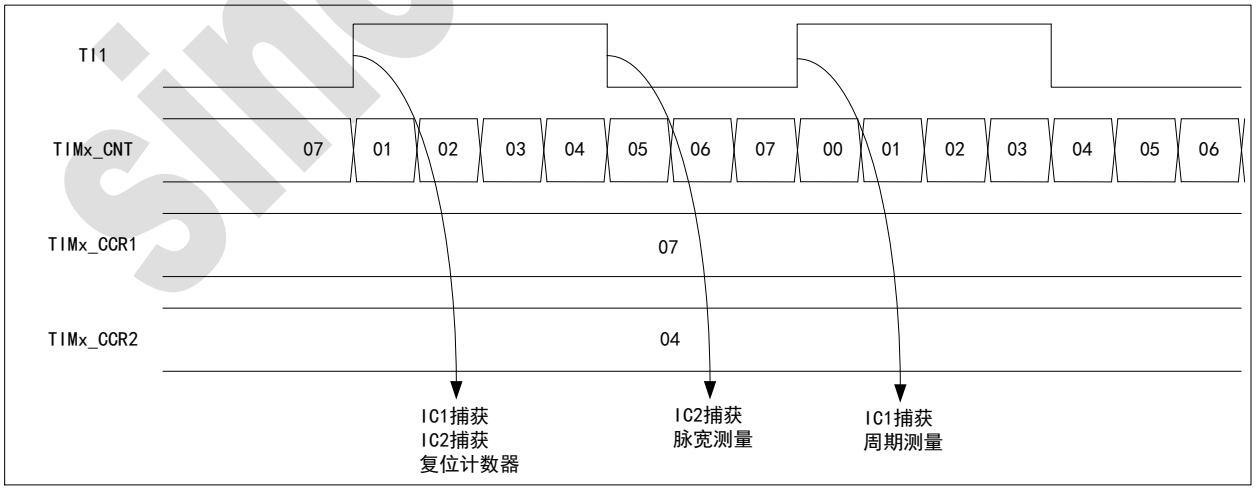
该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

✧ 一个 TIx 输入映射至两个 ICx 信号
✧ 两个 ICx 信号为边沿有效，极性相反
✧ 两个 TIxFP 信号之一被作为触发输入信号，而从模式控制器被配置成复位模式

举例，测量输入到 TI1 上的 PWM 信号的周期（TIMx_CCR1 寄存器）和占空比（TIMx_ CCR2 寄存器），操作步骤如下（取决于 CK_INT 的频率和预分频器的值）：

1、 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01（选中 TI1）。
2、 选择 TI1FP1 的有效极性（用于捕获数据到 TIMx_CCR1 中和清除计数器）：置 CC1P=0（上升沿有效）。
3、 选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
4、 选择 TI1FP2 的有效极性（用于捕获数据到 TIMx_CCR2）：置 CC2P=1（下降沿有效）。
5、 选择有效的触发输入信号：设置 TIMx_SMCR 寄存器中的 TS=101（选择 TI1FP1）。
6、 配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
7、 使能捕获功能：设置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

PWM 输入模式时序



13.3.8 强制输出模式

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

设置 TIMx_CCMRx 寄存器中相应的 OCxM=101，输出比较信号（OCxREF/OCx）强制为有效状态。这样 OCxREF 被强制高

电平(OCxREF 始终为高电平有效)，置位 TIMx_CCER 的 CCxP 位，OCx 可得到极性相反的信号。

举例，CCxP=0（OCx 高电平有效），则 OCx 被强制为高电平。

设置 TIMx_CCMRx 寄存器中相应的 OCxM=100，输出比较信号（OCxREF/OCx）强制为低电平。

该模式下，在 TIMx_CCRx 缓存寄存器和计数器之间的比较仍然在执行，相应的标志也会被置位。并会产生相应的中断和 DMA 请求。下面的输出比较模式一节中将详细介绍。

13.3.9 输出比较模式

此模式用于控制输出波形或指示一段时间到时。

当计数器与捕获/比较寄存器的内容匹配（相等）时，输出比较功能做如下操作：

- ✧ 根据输出比较模式（TIMx_CCMRx 寄存器中 OCxM 位）和输出极性（TIMx_CCER 寄存器中的 CCxP 位）的配置，输出至对应引脚。在比较匹配发生时，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）。
- ✧ 置位中断状态寄存器中的标志位（TIMx_SR 寄存器中的 CCxIF 位）。
- ✧ 若置位相应的中断使能位（TIMx_DIER 寄存器中的 CCxIE 位），则产生一个中断。
- ✧ 若置位相应的 DMA 使能位（TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位），则产生一个 DMA 请求。

设置 TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。计时分辨率为计数器的一次计数。

输出比较模式也能用来输出一个单脉冲（单脉冲模式）。

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、 根据需求，写入数据至 TIMx_ARR 和 TIMx_CCRx 寄存器中。

3、 如果要产生一个中断请求，设置 CCxIE 位。

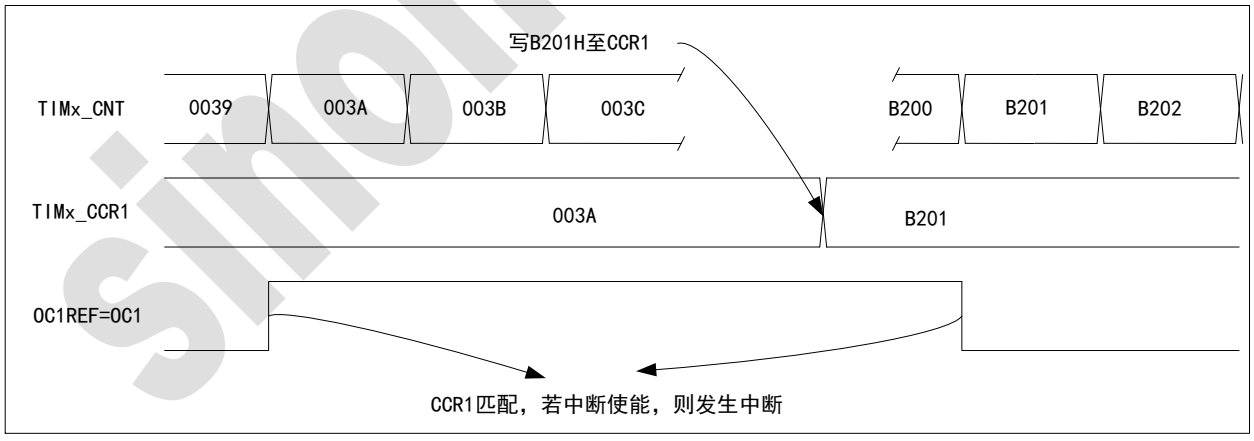
4、 选择输出模式，举例如下：

- 写 OCxM=011，计数器与 CCRx 匹配时，翻转 OCx 的输出引脚；
- 写 OCxPE=0，禁用预装载寄存器；
- 写 CCxP=0，选择极性为高电平有效；
- 写 CCxE=1，使能输出。

5、 置位 TIMx_CR1 寄存器的 CEN 位，启动计数器。

TIMx_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，前提是未启用预装载寄存器（OCxPE= ‘0’，否则 TIMx_CCRx 的缓存寄存器只在发生下次更新事件 EUV 时更新）。举例如下图。

输出比较模式，配置为：翻转 OC1



13.3.10 PWM 模式

脉冲宽度调制模式（PWM）可以产生一个 PWM 信号，由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比。

在 TIMx_CCMRx 寄存器中的 OCxM 位写入 ‘110’（PWM 模式 1）或 ‘111’（PWM 模式 2）， 可以在每个通道独立地设置 PWM 模式(每个 OCx 输出一种 PWM)。必须置位 TIMx_CCMRx 寄存器的 OCxPE 位,使能相应的预装载寄存器(preload register)，最后置位 TIMx_CR1 寄存器的 ARPE 位，使能自动重载预装载寄存器（在向上计数或中心对称模式中）。

只有发生一个更新事件时，预装载寄存器才能被传送到缓存寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。

设置 TIMx_CCER 寄存器中的 CCxP 位，配置 OCx 的极性为高电平有效或低电平有效。OCx 的输出使能通过 （TIMx_CCER 和 TIMx_BDTR 寄存器中）CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详细参考 TIMx_CCER 寄存器的描述。

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

设置 TIMx_CR1 寄存器中 CMS 位，选择 PWM 信号为边沿对齐或中央对齐。

PWM 边沿对齐模式

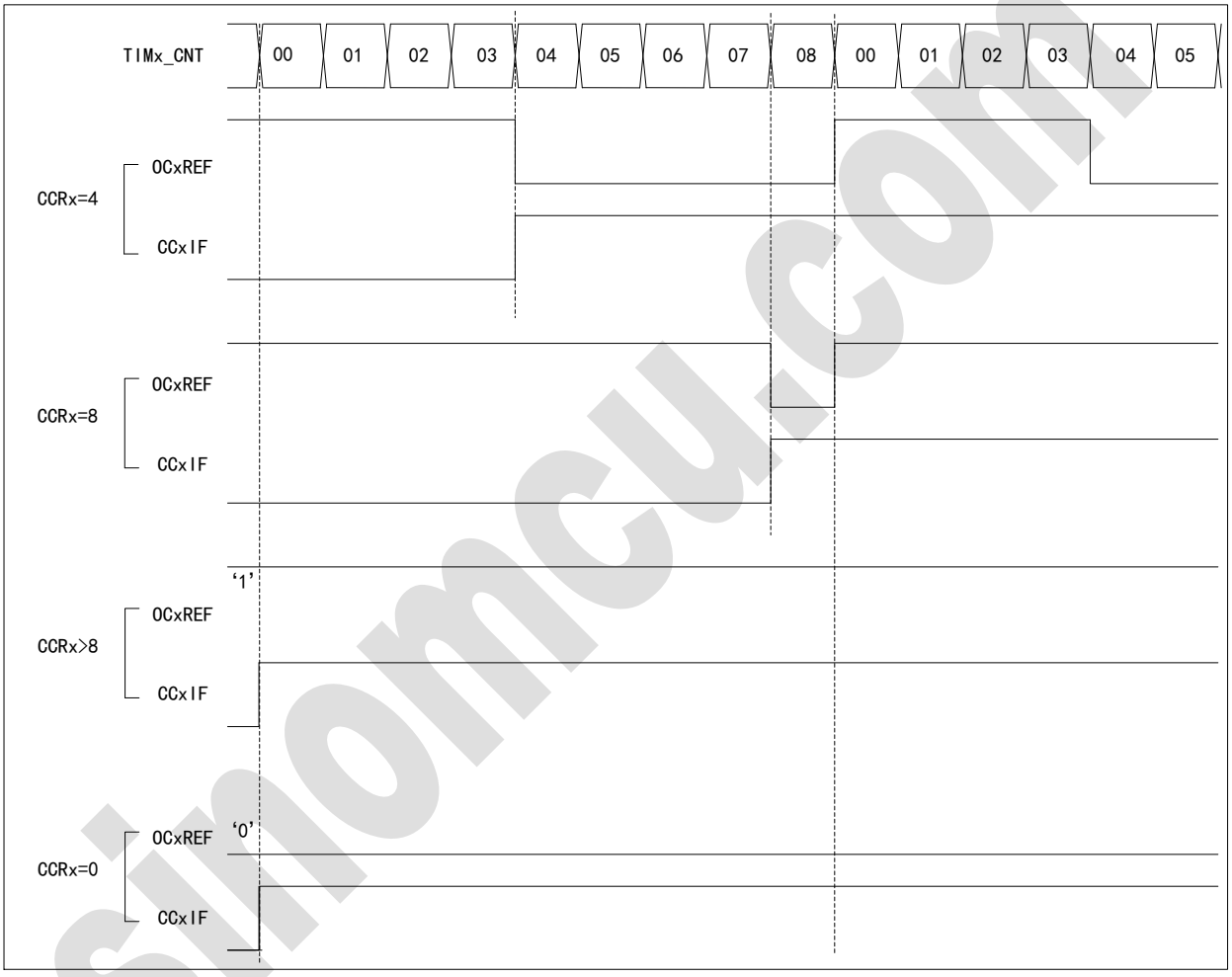
向上计数配置

设置 TIMx_CR1 寄存器中的 CMS=00 且 DIR=0，执行向上计数。参见 13.3.2 向上计数配置章节。

举例，PWM 模式 1

当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重载值(TIMx_ARR)，则 OCxREF 保持为‘1’。如果比较值为 0，则 OCxREF 保持为‘0’。 下图为 TIMx_ARR=8 时边沿对齐的 PWM 波形实例。

PWM 波形（边沿对齐，向上计数，ARR=8）



向下计数配置

设置 TIMx_CR1 寄存器中的 CMS=00 且 DIR=1，执行向下计数。参见 13.3.2 向下计数配置章节。

PWM 模式 1 下，当 TIMx_CNT>TIMx_CCRx 时，PWM 参考信号 OCxREF 为低，否则为高。如果 TIMx_CCRx 中的比较值大于自动重载值 (TIMx_ARR)，则 OCxREF 保持为‘1’。该模式下不能产生 0%的 PWM 波形。

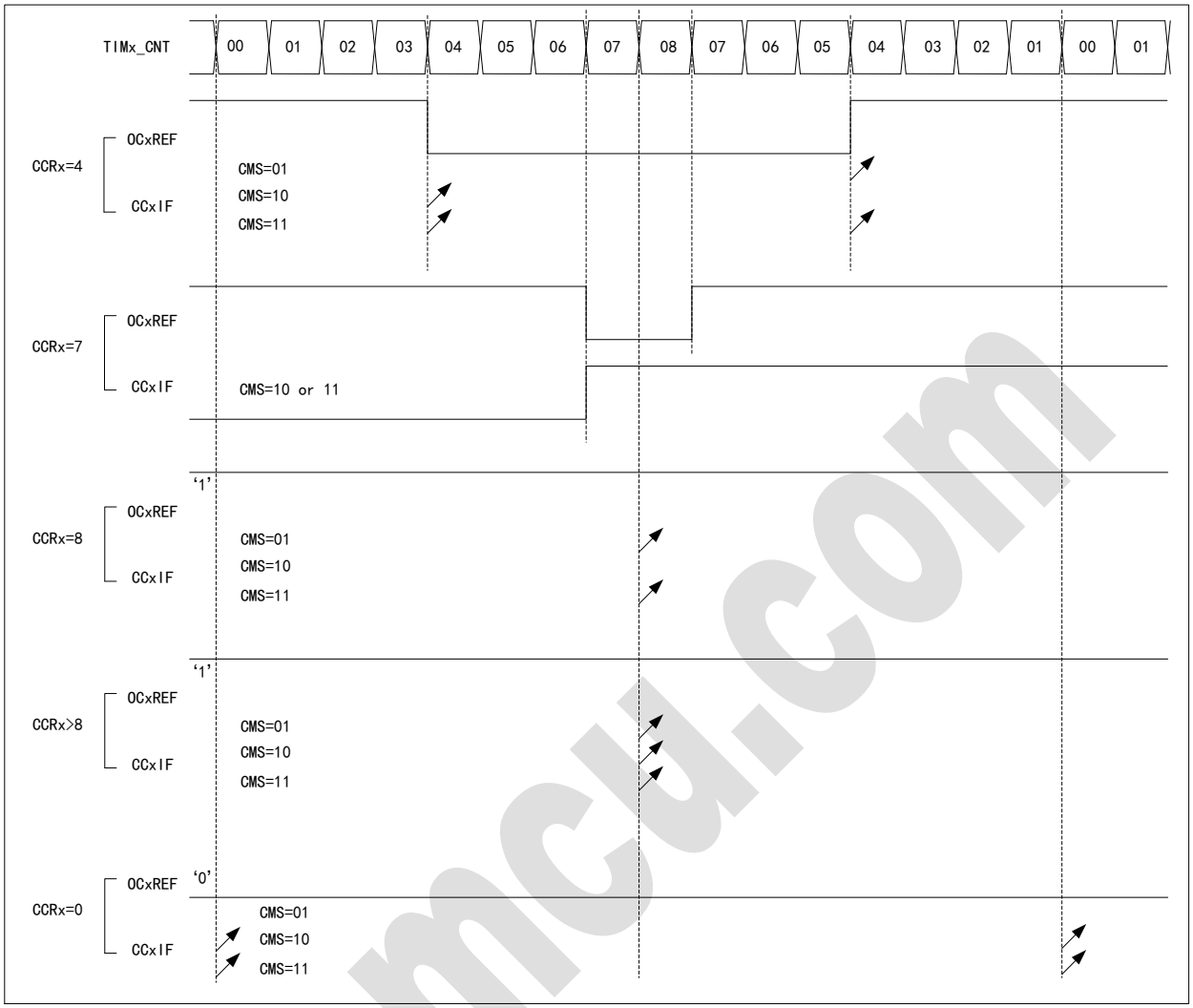
PWM 中心对齐模式

设置 TIMx_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 信号与边沿模式相同)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时、向下计数时、或在向上和向下计数时被置 1。中心对齐模式下，TIMx_CR1 寄存器中的计数方向位（DIR）由硬件更新，只读，软件不可修改。参见 13.3.2 中心对齐模式章节。

举例，PWM 中心对齐模式

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

PWM 中心对齐波形（APR=8）



注意事项：

1、 启动中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 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。

2、 不推荐当运行在中心对齐模式时改写计数器，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载的值（TIMx_CNT>TIMx_ARR），则方向不会被更新。 例如，如果计数器正在向上计数，它就会继续向上计数。
- 如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。

3、 使用中心对齐模式最可靠的方式，就是在启动计数器之前产生一个软件更新（设置 TIMx_ 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。

13.3.11 互补输出/死区插入

高级控制定时器（TIM1）能够输出两路互补信号，并且能够管理输出关断和接通的瞬时时间。

这段时间通常被称为死区，用户应该根据输出连接的器件和它们的特性（电平转换的延时、电源开关的延时等 ）来调整死区时间。

配置 TIMx_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：TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，TIMx_BDTR 和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位。详见 13.4.10 章节表格<带刹车功能的互补输出通道 OCx 和 OCxN 的控制位>。特别的是，在切换到 IDLE 状态时（MOE 下降到 0）死区被激活。

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 10 位的死区发生器。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。

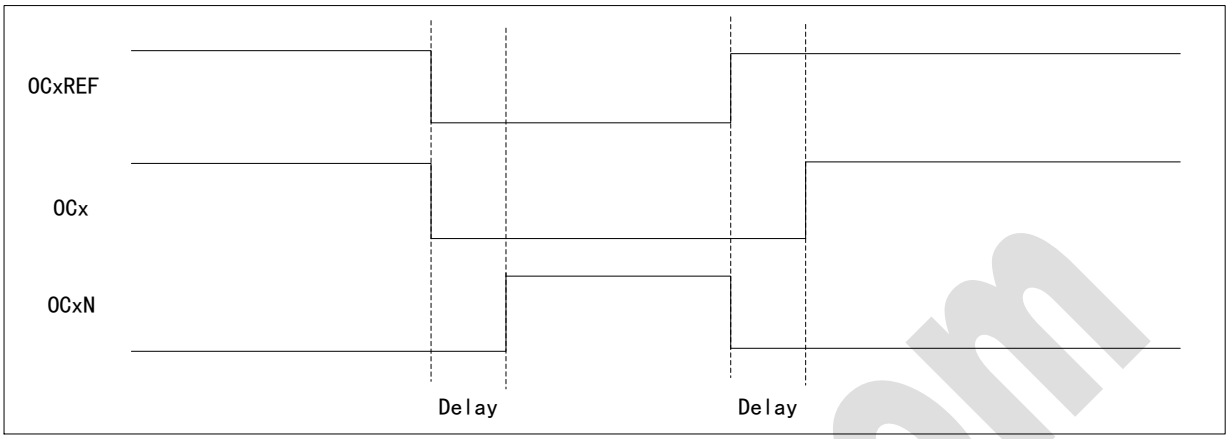
如果 OCx 和 OCxN 为高有效：

- ✧ OCx 输出信号与 OCxREF 参考信号相同，只是上升沿相对于参考信号的上升沿有一个延迟（死区）。
- ✧ OCxN 输出信号与 OCxREF 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（死区）。

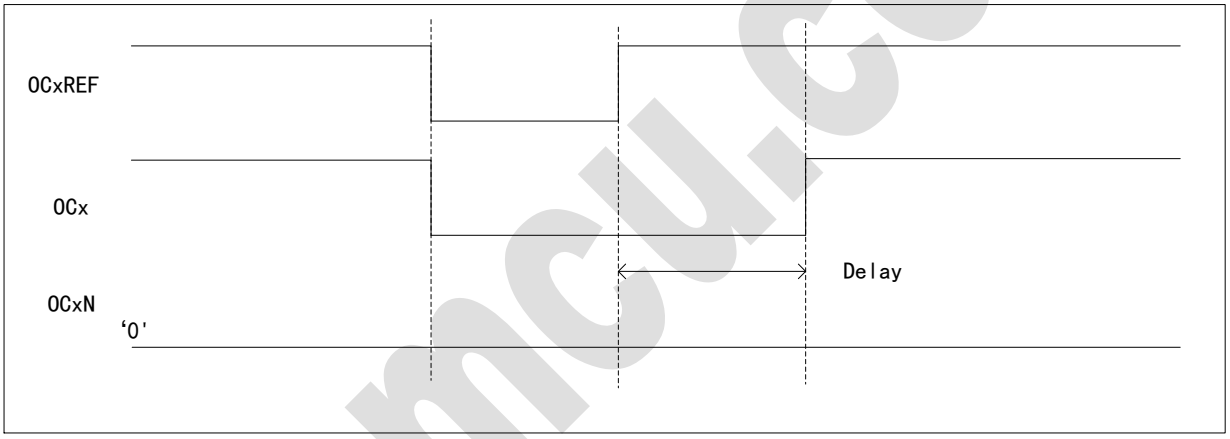
如果延迟大于当前有效的输出宽度（OCx 或者 OCxN），则不会产生相应的脉冲。

下列几张图为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 且 CCxNE=1）

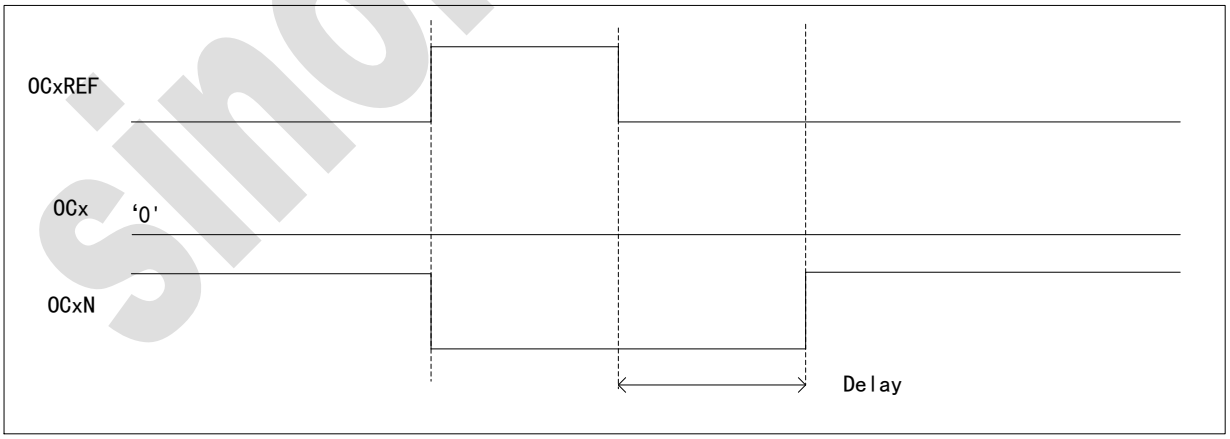
带死区插入的互补输出



死区波形：延迟大于负脉冲



死区波形：延迟大于正脉冲



每一个通道的死区延时都相同，都由 TIMx_BDTR 寄存器中的 DTG 位编程配置。具体的延时计算参见 13.4.19 章节刹车和死区寄存器（TIMx_BDTR）。

重定向 OCxREF 到 OCx 或 OCxN

在输出模式下（强制输出、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，或都处于有效电平且带死区的互补输出。

注：当只使能 OCxN（CCxE=0，CCxNE=1）时，OCxN 不会经过取反，一旦 OCxREF 有效时立即变高。例如，如果 CCxNP=0，

则 OCxN=OCxREF。
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。

13.3.12 刹车功能

当使用刹车功能时，设置额外的控制位（TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIMx_CR2 寄存器中 OISx 和 OISxN 位），输出使能信号和无效电平将会被修改。但无论何时，OCx 和 OCxN 输出电平不能在同一时间被设置都有效。详见 13.4.10 章节表格<带刹车功能的互补输出通道 OCx 和 OCxN 的控制位>。

刹车信号（BRK）源可以是连接刹车输入引脚的外部信号，也可以是以下内部信号之一：

- ✧ LOCKUP 输出
- ✧ PVD 输出
- ✧ SRAM 奇偶校验错误信号
- ✧ 时钟故障事件，由时钟安全系统（CSS）产生
- ✧ 模拟比较器输出

系统复位后，刹车电路被禁止，MOE 位为低。设置 TIMx_BDTR 寄存器中的 BKE 位可以使能刹车功能，设置同一寄存器的 BKP 位，选择刹车输入信号的极性。BKE 和 BKP 可以同时被修改。当写入 BKE 和 BKP 位时，延迟 1 个 APB 时钟周期写入生效，因此需要等待一个 APB 时钟周期之后，才能正确回读。

因为 MOE 下降沿可以是异步的，在实际信号（作用在输出端）和同步控制位（在 TIMx_BDTR 寄存器中）之间插入了一个重新同步电路。这个重新同步电路会在异步信号和同步信号之间产生延迟。特别的，如写 MOE 从 0 变为 1，则必须先插入一个延时（空指令）才能正确回读。这是因为写入是异步信号而读取是同步信号。

当发生刹车时（在刹车输入发生选定的电平），有下述动作：

- ✧ MOE 位被异步清零，将输出置于无效状态、空闲状态或者复位状态（由 OSSI 位选择）。这个特性在 MCU 的振荡器关闭时依然有效。
- ✧ 一旦 MOE=0，每一个输出通道电平由 TIMx_CR2 寄存器中的 OISx 位设定。如果 OSSI=0，则定时器释放（release）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）。这是异步操作，即使定时器没有时钟时，此功能也有效。
- 如果定时器的时钟依然存在，死区生成器将会重新生效，在 1 个死区之后根据 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。注意，因为重新同步 MOE，死区持续时间比通常情况下长一点（大约 2 个 ck_tim 的时钟周期）。
- 如果 OSSI=0，定时器释放输出使能，否则，CCxE 或 CCxNE 为高时，输出使能为高。
- ✧ 置位刹车状态标志（TIMx_SR 寄存器中的 BIF 位），如果设置了 TIMx_DIER 寄存器中的 BIE 位，则产生一个中断。
- ✧ 如果置位 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时，MOE 位被重新自动置位；例如，这可以用来进行整形。否则（AOE=0），MOE 始终保持低直到被再次置‘1’；此时， 这个特性可以用于安全防护，用户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。

注： 刹车输入为电平有效。所以，当刹车输入有效时，MOE 不能被置位（硬件自动或者软件都不可以）。同时，状态标志 BIF 不能被清除。

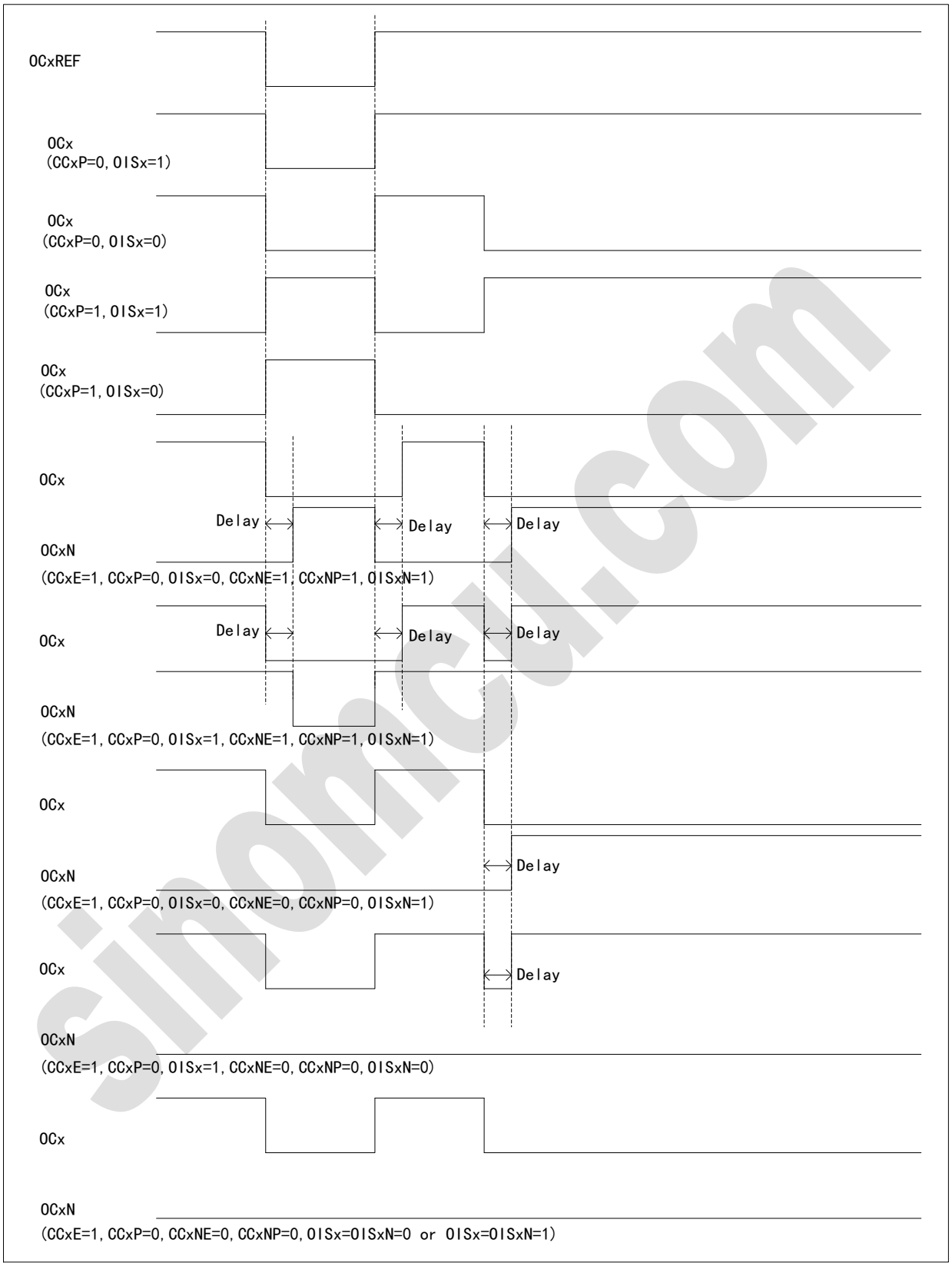
刹车可以由 BRK 输入产生，有效极性可编程，TIMx_BDTR 寄存器中的 BKE 位为使能控制。

除了刹车输入和输出管理，刹车电路中还支持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（死区时间，OCx/OCxN 极性和关闭状态，OCxM 配置，刹车使能和极性）。用户可以通过设置 TIMx_BDTR 寄存器中的 LOCK 位，选择三个保护等级中的一种，详细参见 13.4.19 节 TIM1 刹车和死区寄存器(TIMx_BDTR)。

MCU 复位后，LOCK 位只能被修改一次。

下图为刹车响应的输出实例。

刹车响应的输出



13.3.13 外部事件时清零 OCxREF 信号

设置 TIMx_CCMRx 寄存器中对应的 OCxCE 位为‘1’，指定通道的外部事件触发 OCxREF 信号清零功能开启，即当 OCxREF_CLR_INPUT 发生高电平触发 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次的更新事件 UEV。

该功能只能用于输出比较和 PWM 模式，而不能用于强制模式。

设置 TIMx_SMCR 寄存器中 OCCS 位，可以选 OCREF_CLR 输入（来自模拟比较器）和 ETRF(滤波器后的 ETR)之一作为 OCREF_CLR_INPUT 信号。

举例，OCxREF 信号可以关联到一个比较器的输出，用于控制电流（cycle by cycle）。若 ETRF 被选择，ETR 必须配置如下：

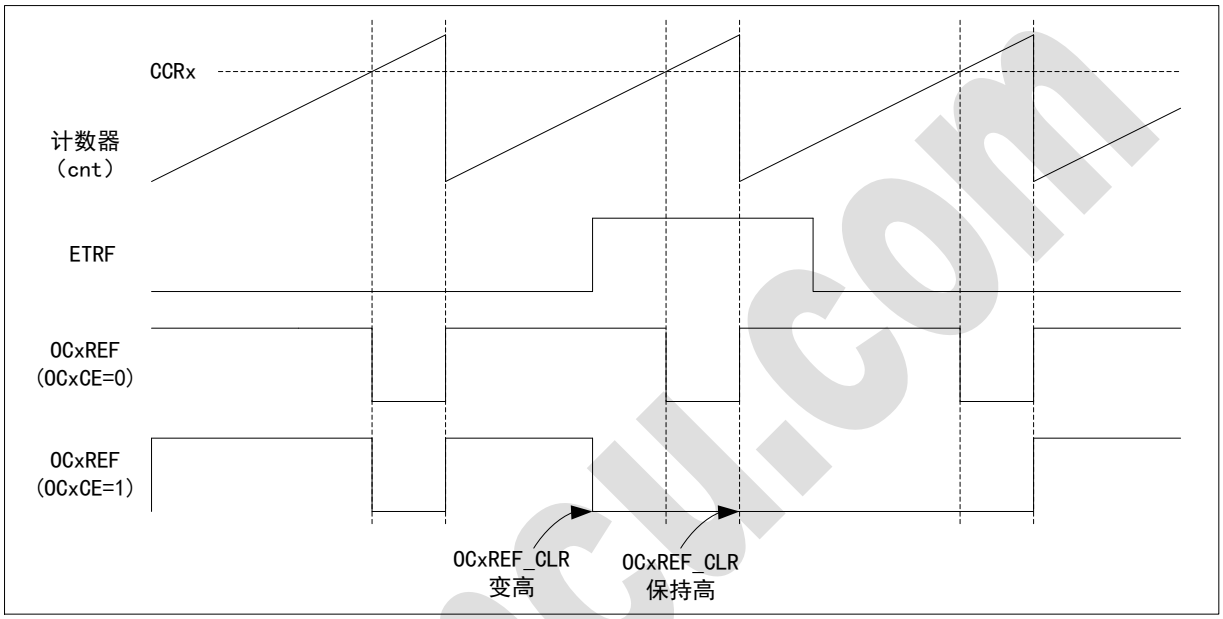
1、 ETR 预分频器必须处于关闭：TIMx_SMCR 寄存器中的 ETPS[1:0]=00。

2、 必须禁止外部时钟模式 2：TIMx_SMCR 寄存器中的 ECE=0。

3、 ETR 的极性（ETP）和滤波器（ETF）可以根据需要配置。

下图显示了当 ETRF/OCxREF_CLR 输入变为高时，对应不同 OCxCE 的值，OCxREF 信号的行为。在此例中，定时器 TIMx 被置于 PWM 模式。

清零 OCxREF 信号时序



注：在 PWM 占空比为 100%的情况时（CCRx>ARR），OCxREF 在下一次计数溢出时被再次使能。

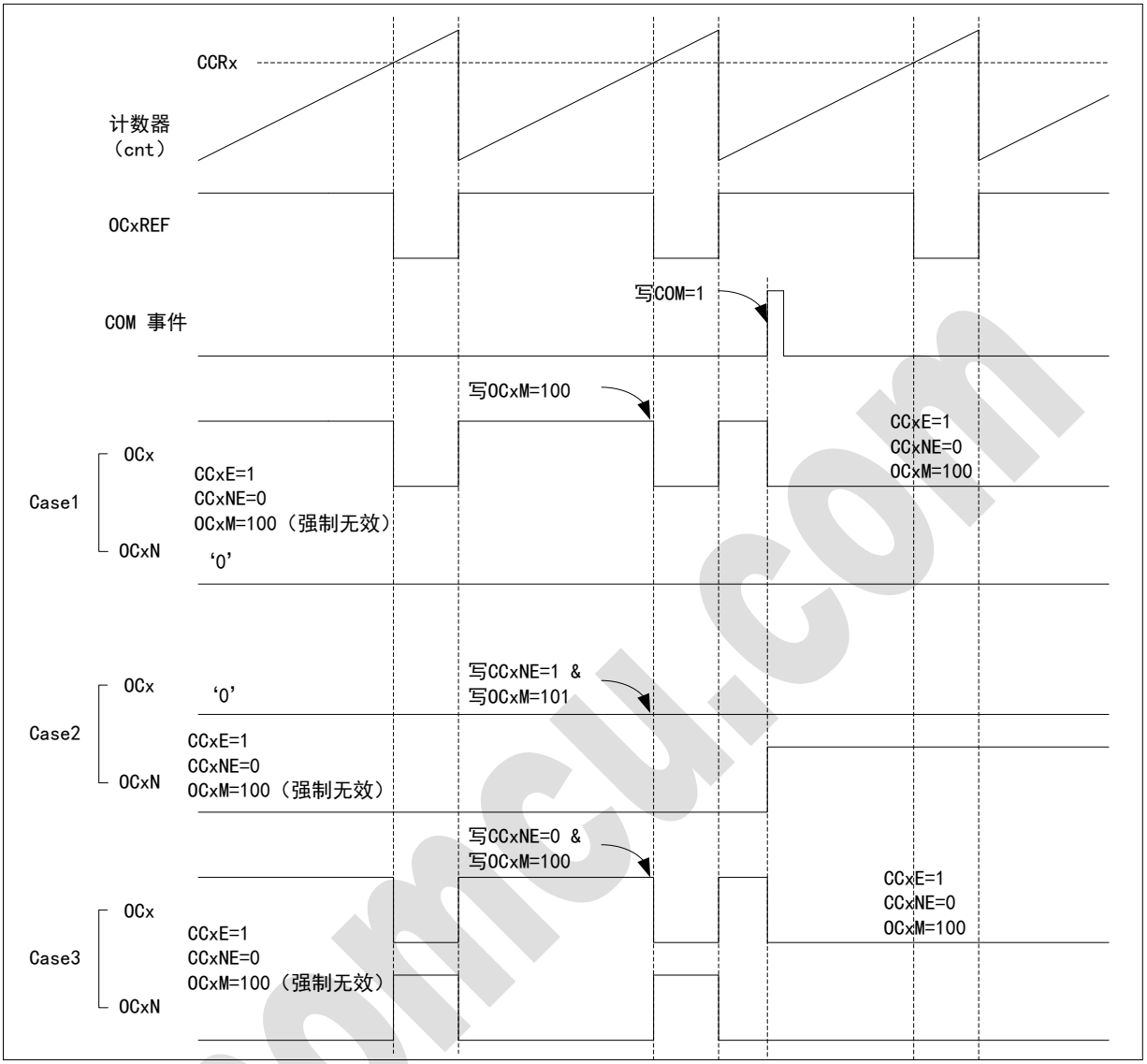
13.3.14 六步 PWM 输出产生

当在一个通道上应用了互补输出时，OCxM、CCxE 和 CCxNE 位的预装位有效。在 COM 换相事件发生时，预装载位被传送到缓存位；因而可以预先设置好下一步的配置，并在同一时刻更改所有通道的配置。COM 事件可以通过硬件(在 TRGI 的上升沿)设置或者软件修改 TIM1_EGR 寄存器中的 COM 位来产生。

当 COM 事件发生时。会置一个标志位(TIM1_SR 寄存器中的 COMIF 位)，这时如果已置位 TIM1_DIER 寄存器的 COMIE 位，则产生一个中断；或者如果已置位 TIM1_DIER 寄存器的 COMDE 位，则产生一个 DMA 请求。

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

六步 PWM 产生，COM 样例（OSSR=1）



COM 事件是专门为电机控制用的，用于同时控制所有通道的输出转换，在电机控制中同时转换所有通道的输出是十分必要的，比如无刷电机换向时，一般是三相要同时换向的，但是用户在软里设置换向时一般一次只能设置一相，这就达不到三相同时换向。这种情况下就可以启用 COM 事件，先逐个设置好每相的换向（注意：此时虽然设置了，但实际上并未生效），然后再调用 COM 事件，此时，三相将同时换向。

13.3.15 单脉冲模式

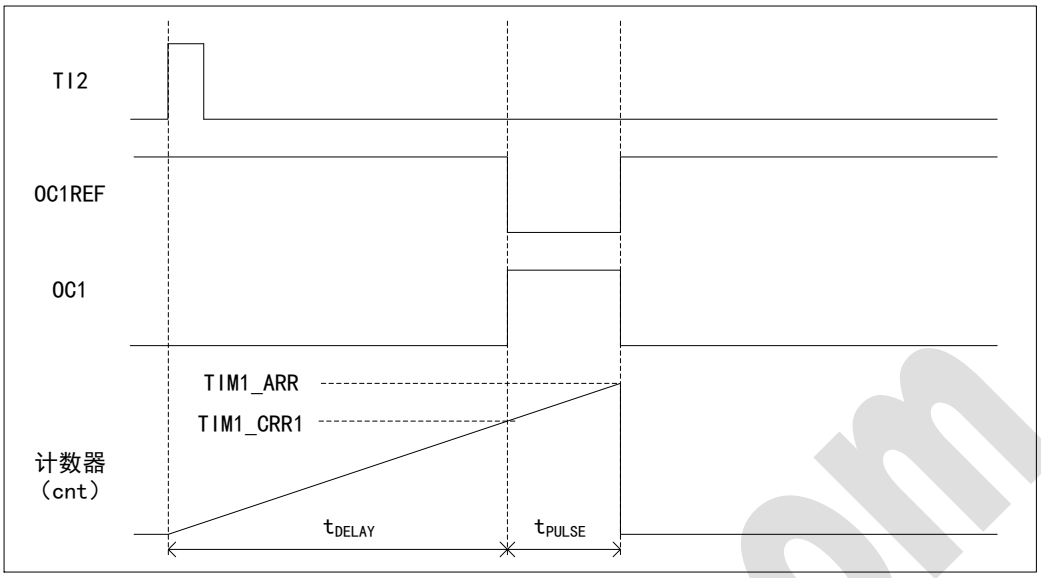
单脉冲模式（OPM）是上述模式的一个特例。这种模式下，计数器响应一个激励后启动，经过一个可编程的延时之后，产生一个脉宽编程的单脉冲。

计数器启动由从模式控制器控制，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_ CR1 寄存器中的 OPM 位将选择单脉冲模式，计数器在产生下一个更新事件 UEV 时自动停止。

想要正确产生一单脉冲，比较值与计数器的初始值必须不同。启动之前（ 此时定时器正在等待触发）， 配置必须满足：

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

单脉冲模式时序举例



举例，要求在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲，从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后。

假定 TI2FP2 作为触发 1:

✧ 配置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。

✧ 配置 TIMx_CCER 寄存器中的 CC2P=0 和 CC2NP=0，使 TI2FP2 检测上升沿。

✧ 配置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发（TRGI）。

✧ 配置 TIMx_SMCR 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（TIMx_ARR - TIMx_CCR1）。

✧ 若要求波形，比较值匹配时从 0 到 1 翻转，当计数器达到预装载值时从 1 到 0 翻转；

- 首先要设置 TIMx_CCMR1 寄存器的 OC1M=111，选择 PWM 模式 2；
- 有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；
- 在 TIMx_CCR1 寄存器中写入比较值，在 TIMx_ARR 寄存器中写入自动重载值，设置 UG 位来产生一个更新事件，
- 最后等待 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该配置为低。

设置 TIMx_CR 寄存器中的 OPM=1，在下一个更新事件（UEV，当计数器从自动装载值翻转到 0）时停止计数。当 TIMx_CR1 寄存器中的 OPM=0 时，进入重复模式。

特殊情况：OCx 快速使能：

在单脉冲模式下，TIx 输入脚的边沿检测逻辑触发 CEN 位来启动计数器。通过计数器和比较值间的比较结果驱动输出翻转。但是，这些操作需要一定的时钟周期，这样就限制了可获得的最小延时 t_{DELAY}。

如果需要以最小延时输出波形，可以置位 TIMx_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF（和 OCx）直接响应激励而不再依赖比较，翻转后电平与比较匹配时的电平一致。OCxFE 只在通道配置为 PWM 模式 1 和 PWM 模式 2 时起作用。

13.3.16 编码器接口模式

选择编码器接口模式：如果计数器只在 TI2 的边沿计数，则设置 TIMx_SMCR 寄存器中的 SMS=001；如果只在 TI1 边沿计数，则置 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。

通过设置 TIMx_CCER 寄存器中的 CC1P 和 CC2P 位，选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。CC1NP 和 CC2NP 位必须保持为低。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口，参见下表 。计数器启动（TIMx_CR1 寄存器中的 CEN=1），则计数器由 TI1FP1 或 TI2FP2 上的每次有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在经过输入滤波器和极性控制后的信号；如果没有滤波和极性反相，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。进而计数器向上或向下计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位修改。不管计数器是对 TI1 或 TI2 单独计数还是对 TI1 和 TI2 同时计数，在任一输入端（TI1 或 TI2）的跳变都会重新计算 DIR 位。

编码器接口模式相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 TIMx_ARR 寄存器的自动装载值之间连续计数（是 0 到 ARR 还是 ARR 到 0 计数，取决于计数方向）。所以在开始计数之前必须配置 TIMx_ARR；同样，捕获、比较、预分频器、重复次数计数器、触发输出这些特性仍工作如常。

编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。

在这个模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。

下表列出了所有可能的组合， 假设 TI1 和 TI2 不同时变换。

计数方向与编码器信号的关系

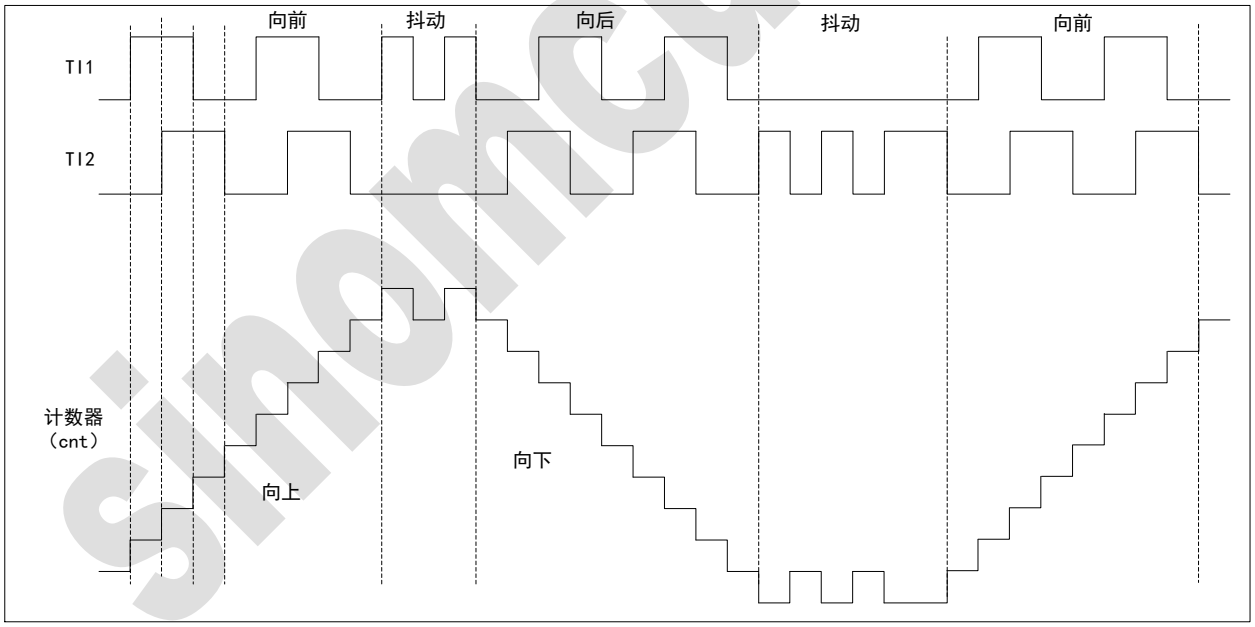
有效沿	相对信号电平 (TI1FP1 对应 TI2, TI2FP2 对应 TI1)	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
在 TI1 和 TI2 上计数	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大提高了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

下图是一个计数器操作的示例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。在这个例子中，我们假定配置如下：

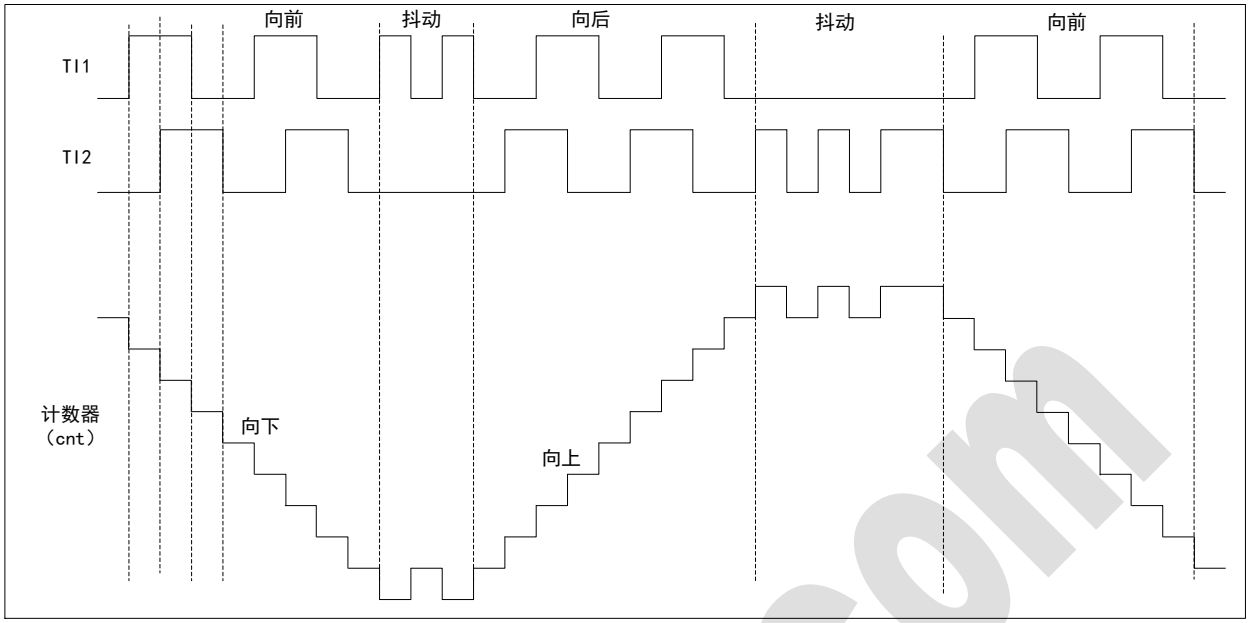
- CC1S=‘01’（TIMx_CCMR1 寄存器，TI1FP1 映射到 TI1）
- CC2S=‘01’（TIMx_CCMR2 寄存器，TI2FP2 映射到 TI2）
- CC1P=‘0’（TIMx_CCER 寄存器，TI1FP1 不反相，TI1FP1=TI1）
- CC2P=‘0’（TIMx_CCER 寄存器，TI2FP2 不反相，TI2FP2 =TI2）
- SMS=‘011’（TIMx_SMCR 寄存器，所有的输入均在上升沿和下降沿都有效）
- CEN=‘1’（TIMx_CR1 寄存器，计数器使能）

编码器模式下的计数器操作实例



下图为当 IC1FP1 极性反相时计数器的操作实例（CC1P=‘1’，其他配置与上例相同）。

TI1FP1 反相的编码器接口模式实例



当定时器配置成编码器接口模式时，用于指示传感器当前位置。配合使用第二个定时器配置为捕获模式，通过测量两个编码器事件的间隔，从而获得动态信息(速度，加速度，减速度)。编码器用来指示机械零点的输出，可用于此目的。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果有必要，可以把计数器的值锁存到第三个输入捕获寄存器 (由另一个定时器产生的捕获信号必须是周期性)；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

13.3.17 输入异或功能

TIMx_CR2 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 TIMx_CH1、TIMx_CH2 和 TIMx_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。13.3.18 章节给出了此特性用于连接霍尔传感器的例子。

13.3.18 与霍尔传感器接口

使用高级控制定时器 (TIM1) 产生 PWM 信号驱动马达，用另一个 TIMx（TIM2 或 TIM3）定时器作为“接口定时器”来连接霍尔传感器，请参见下图。3 个定时器输入脚 (CC1、CC2、CC3) 通过一个异或门连接到 TI1 输入通道(通过设置 TIMx_CR2 寄存器中的 TI1S 位来选择)，由“接口定时器”捕获这个信号。

从模式控制器被配置于复位模式，从输入是 TI1F_ED。这样，每次 3 个输入之一翻转时，计数器从 0 重新开始计数。由此产生一个由霍尔输入端的任何变化而触发的时间基准。

“接口定时器”，捕获/比较通道 1 被配置为捕获模式，捕获信号为 TRC（参见 13.3.5 章节，图：捕获/比较通道输入级）。捕获值反映了输入端两次变化之间的时间间隔，指示出了马达速度的信息。

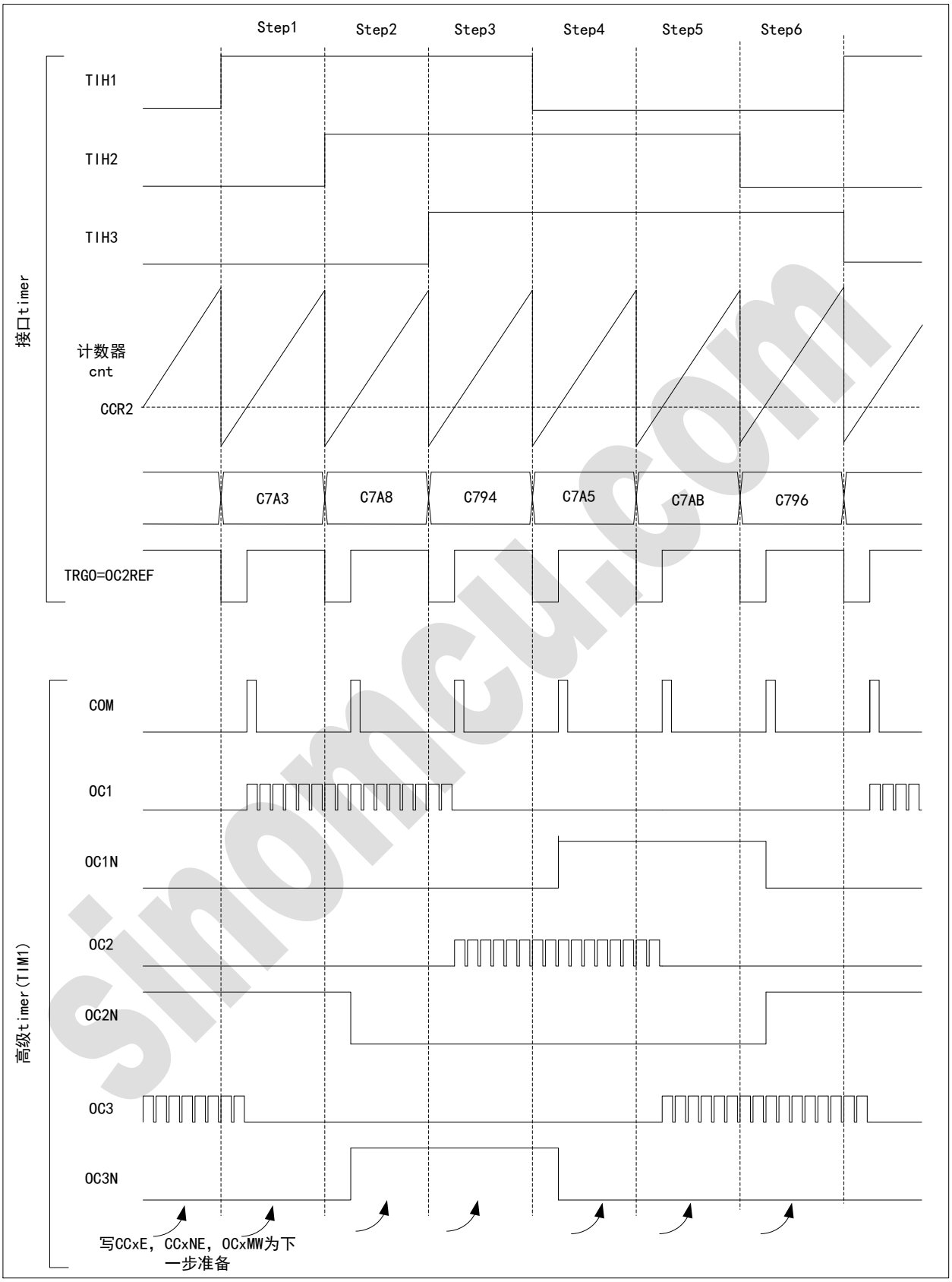
“接口定时器”可以用来在输出模式产生一个脉冲，这个脉冲可以（通过触发一个 COM 事件）用于更新高级定时器（TIM1）各个通道配置，TIM1 定时器用于产生 PWM 信号驱动马达。 为此，必须对“接口定时器”通道编程，进而得到一个经过指定延时的正脉冲（输出比较或 PWM 模式)， 这个脉冲通过 TRGO 输出被送到高级控制定时器（TIM1）。

举例，霍尔输入连接到 TIMx（TIM2 或 TIM3）定时器，要求在每次霍尔输入上发生变化之后的一个指定延迟， 改变高级控制定时器 TIM1 的 PWM 配置。

- ✧ 写 TIMx_CR2 寄存器的 TI1S 位为‘1’，配置三个定时器输入逻辑异或到 TI1 输入；
- ✧ 时基编程：写 TIMx_ARR 为其最大值 (计数器必须通过 TI1 的变化清零)。设置预分频器得到一个最大的计数器周期，它大于传感器上的两次变化的时间间隔。
- ✧ 设置通道 1 为捕获模式（选中 TRC)：写 TIMx_CCMR1 寄存器中 CC1S=01，根据需要设置数字滤波器。
- ✧ 设置通道 2 为 PWM 模式 2，带指定的延时：写 TIMx_CCMR1 寄存器中的 OC2M=111 和 CC2S=00。
- ✧ 选择 OC2REF 作为 TRGO 上的触发输出：写 TIMx_CR2 寄存器中的 MMS=101。

在高级控制寄存器 TIM1 中，选择正确的 ITR 输入作为触发器输入，定时器被编程为产生 PWM 信号， 捕获/比较控制信号配置为预装载（TIMx_CR2 寄存器中 CCPC=1)， 设置触发输入控制 COM 事件（TIMx_CR2 寄存器中 CCUS=1)。在一次 COM 事件后，写入 PWM 控制位（CCxE、OCxM)，为下一步做准备。这可以在 OC2RE 上升沿的产生中断子程序里实现。

霍尔传感器接口的实例



13.3.19 TIMx 与外部触发同步

TIMx 定时器支持多种外部触发同步模式：复位模式、门控模式和触发模式。

从模式：复位模式

在一个触发输入事件发生时，计数器和它的预分频器能够重新被初始化；此外，如果 TIMx_ CR1 寄存器的 URS 位为

低，还产生一个更新事件 UEV；然后更新所有的预装载寄存器（TIMx_ ARR，TIMx_CCRx）。

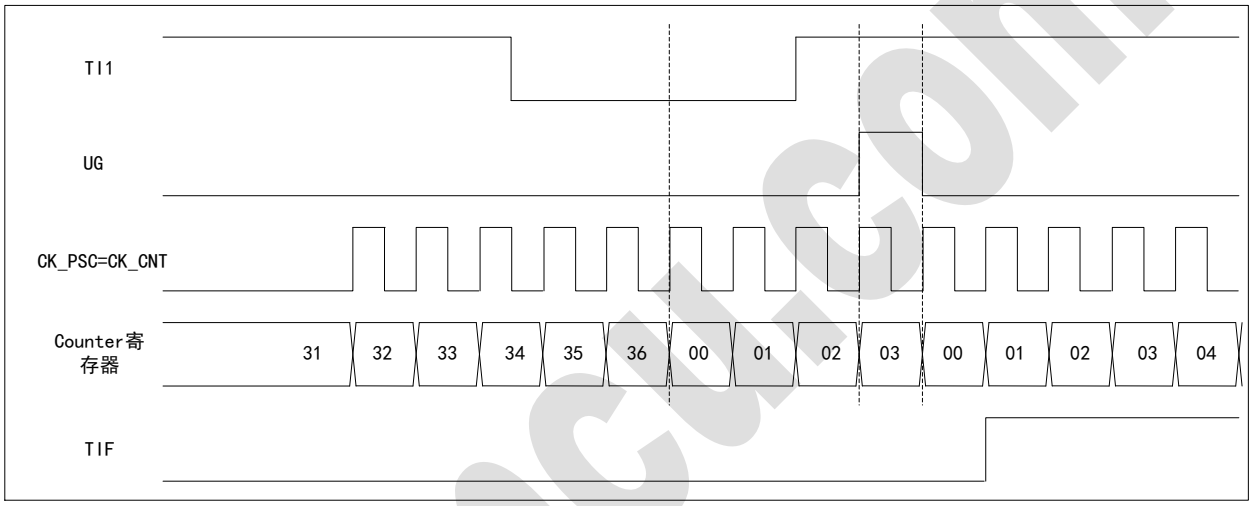
在下面的例子中，TI1 输入端的上升沿清零向上计数器：

- ✧ 配置通道 1 检测 TI1 的上升沿。配置输入滤波器时间（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。配置 TIMx_CCMR1 寄存器中 CC1S=01，只选择输入捕获源。写 TIMx_CCER 寄存器中 CC1P=0 和 CC1NP=0，以确定极性（ 只检测上升沿 ）。
- ✧ 写 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；写 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- ✧ 写 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始以内部时钟计数，然后正常运行直到 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（TIMx_SR 寄存器中的 TIF 位）被置位，若 TIMx_DIER 寄存器中 TIE 位（中断使能）和 TDE 位（DMA 使能）置位，则产生一个中断请求或一个 DMA 请求。

下图显示了自动重装载寄存器 TIMx_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。

复位模式下的控制电路



从模式：门控模式

门控模式，支持根据所选输入的电平使能计数器。

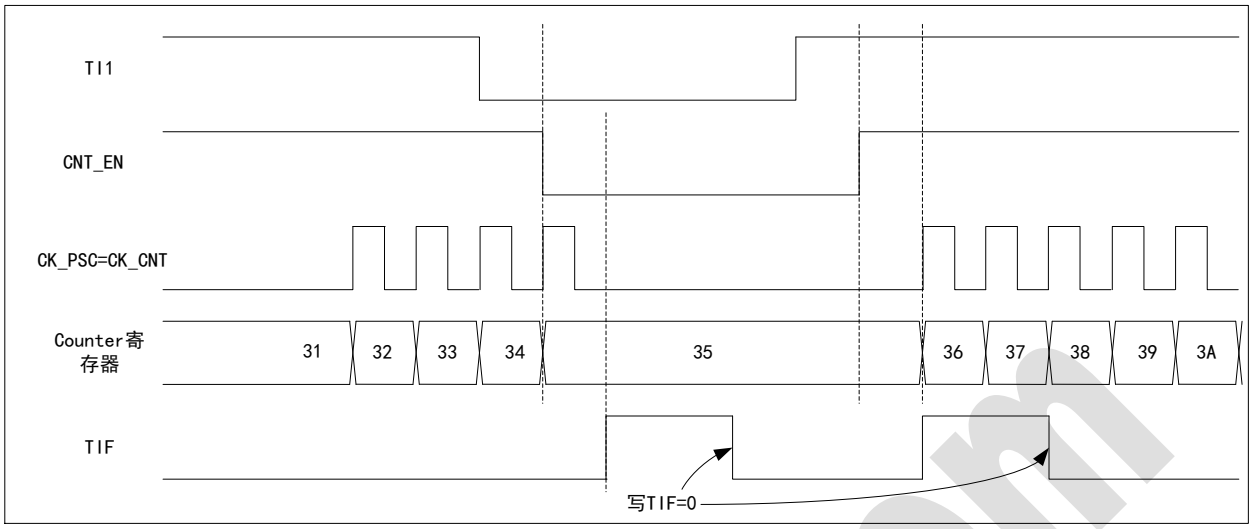
在下面的例子中，计数器只在 TI1 为低时，向上计数：

- ✧ 配置通道 1 检测 TI1 上的低电平。配置输入滤波器时间（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。配置 TIMx_CCMR1 寄存器中 CC1S=01，只选择输入捕获源。写 TIMx_CCER 寄存器中 CC1P=1 和 CC1NP=0，以确定极性（ 只检测低电平 ）。
- ✧ 写 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；写 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- ✧ 写 TIMx_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，计数器开始以内部时钟计数，一旦 TI1 变高则停止计数。计数器开始或停止时，都会置位 TIMx_SR 寄存器中的 TIF 标志。

TI1 上升沿和计数器实际停止之间的延时以及下降沿和计数器实际开始之间的延时，取决于 TI1 输入端的重同步电路。

门控模式下的控制电路



从模式：触发模式

触发模式，支持所选输入端上的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

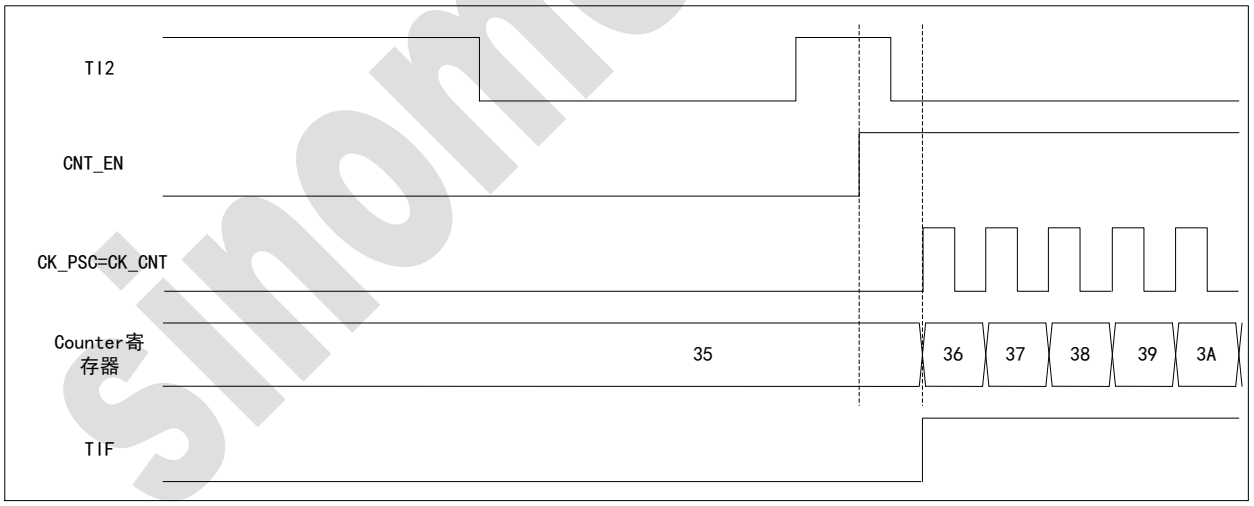
✧ 配置通道 2 检测 TI2 的上升沿。配置输入滤波器时间（本例中，不需要任何滤波器，保持 IC2F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。配置 TIMx_CCMR1 寄存器中 CC2S=01，只选择输入捕获源。写 TIMx_CCER 寄存器中 CC2P=1 和 CC2NP=0，以确定极性（只检测低电平）。

✧ 写 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；写 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器以内部时钟开始计数，同时置位 TIF 标志。

TI2 上升沿和计数器实际启动计数之间的延时，取决于 TI2 输入端的重同步电路。

触发器模式下的控制电路



从模式：外部时钟模式 2+触发模式

外部时钟模式 2 可以与另一种从模式（外部时钟模式 1 和编码器模式除外）一起使用。这时，ETR 信号被用作外部时钟的输入，可以选择另一个输入作为触发输入（在复位模式、门控模式或触发模式）。不建议使用 TIMx_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。

在下面的例子中，在 TI1 上出现一个上升沿后，计数器在 ETR 的每一个上升沿向上计数：

✧ 通过 TIMx_SMCR 寄存器配置外部触发输入电路，如下：

- ETF=0000：没有滤波
- ETPS=00：不用预分频器
- ETP=0：检测 ETR 的上升沿，
- ECE=1：使能外部时钟模式 2。

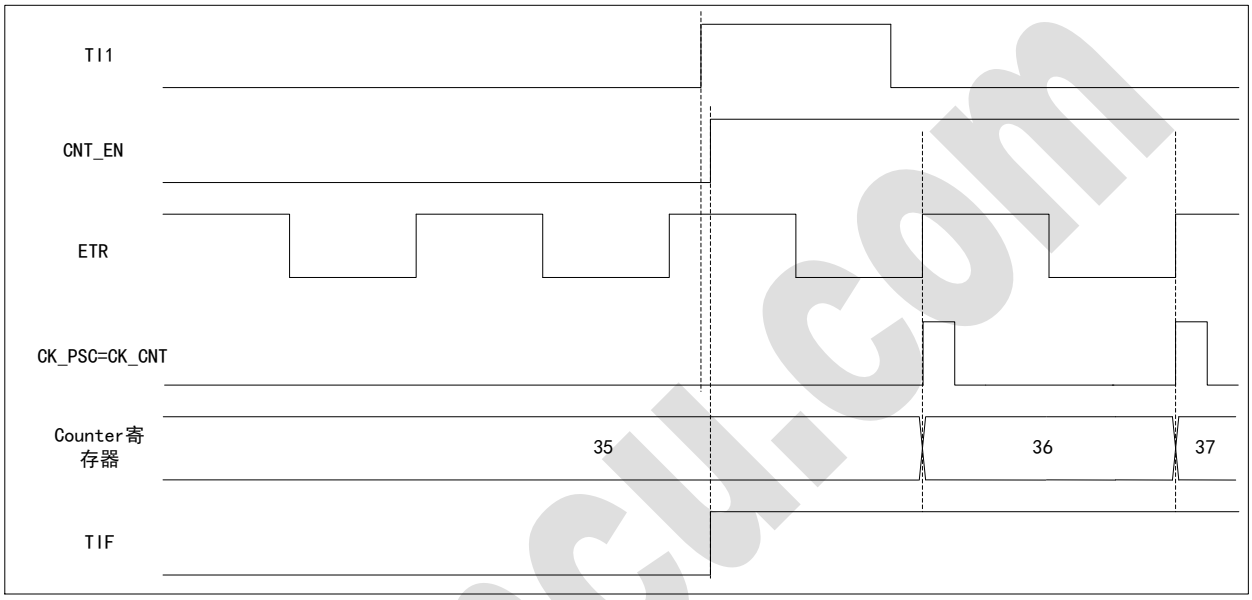
✧ 按如下配置通道 1，检测 TI 的上升沿：

- IC1F=0000：没有滤波
- 触发操作中不使用捕获预分频器，不需要配置
- 写 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源
- 写 TIMx_CCER 寄存器中 CC1P=0 和 CC1NP=0 ，以确定极性（ 只检测上升沿 ）

✧ 写 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式。写 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被置位，计数器开始以 ETR 的上升沿计数。ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

外部时钟模式 2＋触发模式下的控制电路



13.3.20 定时器同步

TIM 定时器在内部相连，用于定时器同步或链接。详见 14.3.15 章节，定时器同步。

13.3.21 调试模式

当微控制器进入调试模式时(Cortex-M0 内核停止)，根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以或者继续正常操作，或者停止。

13.4 相关寄存器

13.4.1 寄存器汇总表

基址：0x4001 2C00				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	TIM1_CR1	0x0000 0000	TIM1 控制寄存器 1	13.4.2
0x04	TIM1_CR2	0x0000 0000	TIM1 控制寄存器 2	13.4.3
0x08	TIM1_SMCR	0x0000 0000	TIM1 从模式控制寄存器	13.4.4
0x0C	TIM1_DIER	0x0000 0000	TIM1 DMA/ 中断使能寄存器	13.4.5
0x10	TIM1_SR	0x0000 0000	TIM1 状态寄存器	13.4.6
0x14	TIM1_EGR	0x0000 0000	TIM1 事件产生寄存器	13.4.7
0x18	TIM1_CCMR1	0x0000 0000	TIM1 捕获/比较模式寄存器 1	13.4.8
0x1C	TIM1_CCMR2	0x0000 0000	TIM1 捕获/比较模式寄存器 2	13.4.9
0x20	TIM1_CCER	0x0000 0000	TIM1 捕获/比较使能寄存器	13.4.10
0x24	TIM1_CNT	0x0000 0000	TIM1 计数器	13.4.11
0x28	TIM1_PSC	0x0000 0000	TIM1 预分频器	13.4.12
0x2C	TIM1_ARR	0x0000 FFFF	TIM1 自动重装载寄存器	13.4.13
0x30	TIM1_RCR	0x0000 0000	TIM1 重复计数寄存器	13.4.14
0x34	TIM1_CCR1	0x0000 0000	TIM1 捕获/比较寄存器 1	13.4.15

0x38	TIM1_CCR2	0x0000 0000	TIM1 捕获/比较寄存器 2	13.4.16
0x3C	TIM1_CCR3	0x0000 0000	TIM1 捕获/比较寄存器 3	13.4.17
0x40	TIM1_CCR4	0x0000 0000	TIM1 捕获/比较寄存器 4	13.4.18
0x44	TIM1_BDTR	0x0000 0000	TIM1 刹车和死区寄存器	13.4.19
0x48	TIM1_DCR	0x0000 0000	TIM1 DMA 控制寄存器	13.4.20
0x4C	TIM1_DMAR	0x0000 0000	TIM1 全部传输时 DMA 地址	13.4.21

13.4.2 TIM1 控制寄存器 1 (TIM1_CR1)

TIM1_CR1			地址偏移	0x00		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	CKD[1:0]	
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:10]	-	保留，保持复位值
BIT[9:8]	CKD[1:0]	时钟分频（Clock division），软件改写 CK_INT 为定时器时钟，t_{DTS} 为死区时间发生器（dead-time）和数字滤波器（ETR,TIx）采样时钟，此位定义了 t_{CK_INT} 和 t_{DTS} 的比率。 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 * t_{CK\_INT}$ 10：$t_{DTS} = 4 * t_{CK\_INT}$ 11：保留，禁止使用
BIT[7]	ARPE	自动重载预装载使能（Auto-reload preload enable） 0: TIMx_ARR 寄存器没有缓冲 1: TIMx_ARR 寄存器有缓冲
BIT[6:5]	CMS[1:0]	中央对齐模式选择（Center-aligned mode selection） 00：边沿对齐模式。计数器依据方向位（DIR）向上或向下计数。 01：中心对齐模式 1。计数器交替地向上和向下计数。输出比较中断标志位在计数器向下计数时被置位,配置（TIMx_CCMRx 寄存器中 CCxS=00）通道输出比较中断。 10：中心对齐模式 2。计数器交替地向上和向下计数。输出比较中断标志位在计数器向上计数时被置位,配置（TIMx_CCMRx 寄存器中 CCxS=00）通道输出比较中断。 11：中心对齐模式 3。计数器交替地向上和向下计数。输出比较中断标志位在计数器向上和向下计数时均被置位,配置（TIMx_CCMRx 寄存器中 CCxS=00）通道输出比较中断。 *注：计数器开启时(CEN=1)，不允许从边沿对齐模式转换到中心对齐模式。*
BIT[4]	DIR	方向（Direction） 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。*
BIT[3]	OPM	单脉冲模式（One pulse mode） 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。
BIT[2]	URS	更新请求源（Update request source） 软件置位和清零，选择 UEV 事件的源 0：下述任一事件产生更新中断和 DMA 请求（如果 DMA 使能）： - 计数器上溢/下溢 - 置位 UG 位 - 从模式（slave mode）控制器产生的更新 1：仅计数器上溢/下溢产生更新中断或 DMA 请求（如果 DMA 使能）。
BIT[1]	UDIS	禁止更新（Update disable） 软件置位和清零，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 - 置位 UG 位 - 从模式（slave mode）控制器产生的更新 具有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。如果置位 UG 位或从模式（slave mode）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。
BIT[0]	CEN	使能计数器（Counter enable） 0：禁止计数器 1：使能计数器 *注：只有软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。但触发模式可以自动地通过硬件设置 CEN 位。*

13.4.3 TIM1 控制寄存器 2（TIM1_CR2）

TIM1_CR2			地址偏移	0x04		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1
R/W	-	rw	rw	rw	rw	rw	rw	rw
复位值	-	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	TI1S	MMS[2:0]			CCDS	CCUS	-	CCPC
R/W	rw	rw	rw	rw	rw	rw	-	rw
复位值	0	0	0	0	0	0	-	0

BIT[15]	-	保留，保持复位值
BIT[14]	OIS4	输出空闲状态 4(OC4 输出） 参见 OIS1 位
BIT[13]	OIS3N	输出空闲状态 3(OC3N 输出） 参见 OIS1N 位
BIT[12]	OIS3	输出空闲状态 3(OC3 输出） 参见 OIS1 位
BIT[11]	OIS2N	输出空闲状态 2(OC2N 输出） 参见 OIS1N 位
BIT[10]	OIS2	输出空闲状态 2(OC2 输出） 参见 OIS1 位
BIT[9]	OIS1N	输出空闲状态 1(OC1N 输出）（Output Idle state 1） 0：当 MOE=0 时，死区后 OC1N=0； 1：当 MOE=0 时，死区后 OC1N=1。 *注：若设置了 LOCK(TIMx_BDTR 寄存器 LOCK 位）级别 1 、2 或 3 后，该位不能被修改。*
BIT[8]	OIS1	输出空闲状态 1(OC1 输出）（Output Idle state 1） 0：当 MOE=0 时，如果 OC1N 被使用，则死区后 OC1=0； 1：当 MOE=0 时，如果 OC1N 被使用，则死区后 OC1=1。 *注：已经设置了 LOCK(TIMx_BDTR 寄存器 LOCK 位 ）级别 1 、2 或 3 后，该位不能被修改。*
BIT[7]	TI1S	TI1 选择（TI1 selection） 0：TIMx_CH1 引脚连到 TI1 输入； 1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 引脚经异或(XOR)后连到 TI1 输入。
BIT[6:4]	MMS[2:0]	主模式选择（Master mode selection） 这 3 位用于选择在主模式下送到从定时器的同步信息（TRGO）。可能的组合如下： 000：复位 - TIMx_EGR 寄存器的 UG 位被用于作为触发输出（TRGO）。如果是触发输入产生的复位（ 从模式控制器处于复位模式 ），则 TRGO 上的信号相对实际的复位会有一个延迟。 001：使能 - 计数器使能信号 CNT_EN 被用于作为触发输出（TRGO）。可用于同时启动多个定时器或控制在一段时间内使能从定时器。在门控模式下，计数器使

		能信号是 CEN 控制位和触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式（见 TIMx_SMCR 寄存器中 MSM 位的描述）。 010：更新 - 更新事件被选为触发输入 (TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲 - 在捕获发生或比较匹配发生，CC1IF 标志被置位时（即使它已经为高），触发输出发送一个正脉冲 (TRGO)。 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)。
BIT[3]	CCDS	捕获/比较的 DMA 选择（capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx DMA 请求 1：当发生更新事件时，送出 CCx DMA 请求
BIT[2]	CCUS	捕获/比较控制更新选择（capture/compare control update selection) 0：如果捕获/比较控制位是预装载的(CCPC=1)，只能通过设置 COMG 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 COMG 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。*
BIT[1]	-	保留，保持复位值
BIT[0]	CCPC	捕获/比较预装载控制位 (capture/compare preloaded control) 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，只能在发生通信 (COM) 事件（COMG 置位或 TRGI 检测到上升沿，取决于 CCUS 位）位时被更新。 *注：该位只对具有互补输出的通道起作用。*

13.4.4 TIM1 从模式控制寄存器 (TIM1_SMCR)

TIM1_SMCR			地址偏移	0x08		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	ETP	ECE	ETPS[1:0]		ETF[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	MSM	TS[2:0]			OCCS	SMS[2:0]		
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15]	ETP	外部触发极性 (External trigger polarity) 该位选择是用 ETR 或 $\overline{ETR}$ 作为触发操作 0：ETR 不反相，高电平或上升沿有效 1：ETR 被反相，低电平或下降沿有效
BIT[14]	ECE	外部时钟使能位 (External clock enable) 该位启用外部时钟模式 2 0：禁止外部时钟模式 2； 1：使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。 *注 1：设置 ECE 位，与选择外部时钟模式 1 并将 TRGI 连到 ETRF（SMS=111 和 TS=111）具有相同功效。* *注 2：下述从模式可以与外部时钟模式 2 同时使用：复位模式，门控模式和触发模式；但是，这时 TRGI 不能连到 ETRF(TS 位不能是‘111’）。* *注 3：外部时钟模式 1 和外部时钟模式 2 同时被使能时，外部时钟的输入是 ETRF。*
BIT[13:12]	ETPS[1:0]	外部触发预分频 (External trigger prescaler) 外部触发信号 ETRP 的频率最多是 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 00：关闭预分频； 01：ETRP 频率除以 2； 10：ETRP 频率除以 4； 11：ETRP 频率除以 8。

BIT[11:8]	ETF[3:0]	外部触发滤波 (External trigger filter) 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。数字滤波器是一个事件计数器，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 $f_{SAMPLING}=f_{DTS}$ 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：请注意，当 ETF[3:0]=1/2/3 时，公式中 f_{DTS} 被 f_{CK_INT} 取代。*
BIT[7]	MSM	主/从模式 (Master/slave mode) 0：无作用 1：触发输入(TRGI)上的事件被延迟，以允许在当前定时器与它的从定时器间的完美同步(通过 TRGO)。当要求把几个定时器同步到一个单一的外部事件时，这是很有用的
BIT[6:4]	TS[2:0]	触发选择 (Trigger selection) 这 3 位选择用于同步计数器的触发输入。 000：内部触发 0 (ITR0) 001：内部触发 1 (ITR1) 010：内部触发 2 (ITR2) 011：内部触发 3 (ITR3) 100：TI1 的边沿检测器 (TI1F_ED) 101：滤波后的定时器输入 1 (TI1FP1) 110：滤波后的定时器输入 2 (TI2FP2) 111：外部触发输入 (ETRF) 更多有关 ITRx 的细节，参见下表：TIMx 内部触发的连接。 *注：这些位只能在未用到(如 SMS=000) 时被改变，以避免在改变时产生错误的边沿检测。*
BIT[3]	OCCS	OCREF 清零信号源选择（OCREF clear selection） 0：OCREF_CLR_INT 被连接至 OCREF_CLR 输入 1：OCREF_CLR_INT 被连接至 ETRF
BIT[2:0]	SMS	从模式选择 (Slave mode selection) 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相关(见输入控制寄存器和控制寄存器的说明) 000：关闭从模式 - 如果 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 - 根据 TI1FP2 的电平，计数器在 TI2FP1 的边沿向上/下计数。 010：编码器模式 2 - 根据 TI2FP1 的电平，计数器在 TI1FP2 的边沿向上/下计数。 011：编码器模式 3 - 根据其他输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 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，并且产生一次更新寄存器。 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 1：如果 TI1F_ED 被选为触发输入 (TS=100b) 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。*

		注 2：在接收主定时器事件之前，从定时器的时钟必须使能，并且在接收主定时器的触发信号过程中，不能动态改变。

表：TIMx 内部触发连接

Slave TIM	ITR0 (TS = 000)	ITR1 (TS = 001)	ITR2 (TS = 010)	ITR3 (TS = 011)
TIM1	Res.	TIM2	TIM3	TIM17

13.4.5 TIM1 DMA/ 中断使能寄存器 (TIM1_DIER)

TIM1_DIER			地址偏移	0x0C		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	TDE	COMDE	CC4DE	CC3DE	CC2DE	CC1DE	UDE
R/W	-	rw	rw	rw	rw	rw	rw	rw
复位值	-	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	BIE	TIE	COMIE	CC4IE	CC3IE	CC2IE	CC1IE	UIE
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15]	-	保留，保持复位值
BIT[14]	TDE	允许触发 DMA 请求 (Trigger DMA request enable) 0：触发 DMA 请求禁止 1：触发 DMA 请求允许
BIT[13]	COMDE	COM DMA 请求使能 0：COM DMA 请求禁止 1：COM DMA 请求允许
BIT[12]	CC4DE	捕获/比较 4 DMA 请求使能 0：CC4 DMA 请求禁止 1：CC4 DMA 请求允许
BIT[11]	CC3DE	捕获/比较 3 DMA 请求使能 0：CC3 DMA 请求禁止 1：CC3 DMA 请求允许
BIT[10]	CC2DE	捕获/比较 2 DMA 请求使能 0：CC2 DMA 请求禁止 1：CC2 DMA 请求允许
BIT[9]	CC1DE	捕获/比较 1 DMA 请求使能 0：CC1 DMA 请求禁止 1：CC1 DMA 请求允许
BIT[8]	UDE	更新 DMA 请求使能 0：更新 DMA 请求禁止 1：更新 DMA 请求允
BIT[7]	BIE	刹车中断使能 0：刹车中断禁止 1：刹车中断允
BIT[6]	TIE	触发中断使能 0：触发中断禁止 1：触发中断允许
BIT[5]	COMIE	COM 中断使能 0：COM 中断禁止 1：COM 中断允许
BIT[4]	CC4IE	捕获/比较 4 中断使能 0：CC4 中断禁止 1：CC4 中断允许
BIT[3]	CC3IE	捕获/比较 3 中断使能 0：CC3 中断禁止 1：CC3 中断允许
BIT[2]	CC2IE	捕获/比较 2 中断使能 0：CC2 中断禁止 1：CC2 中断允许

BIT[1]	CC1IE	捕获/比较 1 中断使能 0：CC1 中断禁止 1：CC1 中断允许
BIT[0]	UIE	更新中断使能 0：更新中断禁止 1：更新中断允许

13.4.6 TIM1 状态寄存器 (TIM1_SR)

TIM1_SR			地址偏移	0x10		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	CC4OF	CC3OF	CC2OF	CC1OF	-
R/W	-	-	-	rc_w0	rc_w0	rc_w0	rc_w0	-
复位值	-	-	-	0	0	0	0	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
R/W	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0
复位值	0	0	0	0	0	0	0	0

BIT[15:13]	-	保留，保持复位值
BIT[12]	CC4OF	捕获/比较 4 重复捕获标志 ，参考 CC1OF 描述
BIT[11]	CC3OF	捕获/比较 3 重复捕获标志 ，参考 CC1OF 描述
BIT[10]	CC2OF	捕获/比较 2 重复捕获标志 ，参考 CC1OF 描述
BIT[9]	CC1OF	捕获/比较 1 重复捕获标志 (Capture/Compare 1 overcapture flag) 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。 软件写 0 可清除该位。 0：无重复捕获产生 1：当 CC1IF 的状态已经为‘1’，计数器的值被捕获到 TIMx_CCR1 寄存器
BIT[8]	-	保留，保持复位值
BIT[7]	BIF	刹车中断标志 (Break interrupt flag) 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
BIT[6]	TIF	触发器中断标志 (Trigger interrupt flag) 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，或门控模式下的任一边沿 ） 时由硬件对该位置’1’。 它由软件清’0’。 0：无触发器事件产生 1：触发中断等待响应
BIT[5]	COMIF	COM 中断标志 (COM interrupt flag) 一旦产生 COM 事件 （ 当捕获/比较控制位：CCxE、CCxNE、OCxM 已被更新 ） 该位由硬件置’1’。 它由软件清’0’。 0：无 COM 事件产生 1：COM 中断等待响应
BIT[4]	CC4IF	捕获/比较 4 中断标志 ，参考 CC1IF
BIT[3]	CC3IF	捕获/比较 3 中断标志 ，参考 CC1IF
BIT[2]	CC2IF	捕获/比较 2 中断标志 ，参考 CC1IF
BIT[1]	CC1IF	捕获/比较 1 中断标志 **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 TIMx_ CR1 寄存器的 CMS 位)。 它由软件清’0’ 。 0：无匹配发生 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器上溢，或向下计数模式时的计数器下溢条件下，CC1IF 位变高

		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’，它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无输入捕获产生 1：计数器值被捕获至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)
BIT[0]	UIF	更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置’1’。它由软件清’0’。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： - 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器（repetition counter）数值上溢或下溢时 （ 重复计数器=0 时产生更新事件 ）。 - 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 进而实现软件对计数器 CNT 重新初始化时。 - 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。参考章节：TIM1 从模式控制寄存器 (TIM1_SMCR)

13.4.7 TIM1 事件产生寄存器 (TIM1_EGR)

TIM1_EGR			地址偏移	0x14		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0

BIT[15:8]	-	保留，保持复位值
BIT[7]	BG	产生刹车事件 (Break generation) 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。MOE 被清 0、BIF 标志置 1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
BIT[6]	TG	触发产生（Trigger generation） 该位由软件置‘1’，用于产生一个事件，由硬件自动清‘0’。 0：无动作 1：TIMx_SR 中 TIF 标志置 1，若开启对应的中断和 DMA，则产生相应的中断和 DMA
BIT[5]	COMG	捕获/比较控制更新产生（Capture/Compare control update generation） 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：该位只对拥有互补输出的通道有效。*
BIT[4]	CC4G	捕获/比较 4 发生参考 CC1G 描述
BIT[3]	CC3G	捕获/比较 3 发生参考 CC1G 描述
BIT[2]	CC2G	捕获/比较 2 发生参考 CC1G 描述
BIT[1]	CC1G	捕获/比较 1 发生 (Capture/Compare 1 generation) 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 1 上产生一个捕获/比较事件 **若通道 CC1 配置为输出：** 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 **若通道 CC1 配置为输入：** 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
BIT[0]	UG	产生更新事件 (Update generation) 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（寄存器）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 TIMx_ARR 的值

13.4.8 TIM1 捕获/比较模式寄存器 1（TIM1_CCMR1）

TIM1_CCMR1			地址偏移	0x18		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]	
	IC2F[3:0]				IC2PSC[1:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	IC1F[3:0]				IC1PSC[1:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

注：CCxS 位定义通道方向，可用于输入（捕获模式）或输出（比较模式）。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能，ICxx 描述了通道在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

输出比较模式：

BIT[15]	OC2CE	输出比较 2 清 0 允许
BIT[14:12]	OC2M[2:0]	输出比较 2 模式
BIT[11]	OC2PE	输出比较 2 预分频允许
BIT[10]	OC2FE	输出比较 2 快速使能
BIT[9:8]	CC2S[1:0]	捕获/比较 2 选择（capture/compare 2 selection） 该 2 位定义通道的方向（输入/输出），及输入信号的选择： 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 TIMx_SMCR 寄存器的 TS 位选择） *注：CC2S 仅在通道关闭时（TIMx_CCER 寄存器的 CC2E=0）才是可写的。*
BIT[7]	OC1CE	输出比较 1 清 0 允许（output compare 1 clear enable） 0：OC1REF 不受 ETRF 输入的影响； 1：一旦检测到 ETRF 输入高电平，清除 OC1REF=0。
BIT[6:4]	OC1M	输出比较模式 1（Output Compare 1 mode） 该 3 位定义了输出参考信号 OC1REF 的行为，而 OC1REF 决定了 OC1、OC1N 的电平。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 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用（此模式用于产生一个时基）； 001：匹配时，设置通道 1 为有效电平。当 TIMx_CNT = TIMx_CCR1 时，强制 OC1REF 为高。 010：匹配时，设置通道 1 为无效电平。当 TIMx_CNT = TIMx_CCR1 时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_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，当 TIMx_CNT<TIMx_CCR1 时，通道 1 为有效电平(OC1REF=1)，否则为无效电平(OC1REF=0)； - 在向下计数时，当 TIMx_CNT>TIMx_CCR1 时，通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2 - 在向上计数时，当 TIMx_CNT<TIMx_CCR1 时，通道 1 为无效电平，否则为有效电平； - 在向下计数时，当 TIMx_CNT>TIMx_CCR1 时，通道 1 为有效电平，否则

		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。* *注 3：在通道有互补输出时，此位被预装载。如果 TIMx_CR2 中的 CCPC 位被置位，那么只有在发生 COM 事件时，OC1M 的有效位才能从预装位得到新的值。*
BIT[3]	OC1PE	输出比较 1 预装载允许 (Output Compare 1 preload enable) 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1： 开启 TIMx_CCR1 寄存器的预装载功能， 读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 *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：仅在单脉冲模式下 (TIMx_CR1 寄存器的 OPM=1)，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。*
BIT[2]	OC1FE	输出比较 1 快速使能 (Output Compare 1 fast enable) 该位用于加快 CC 输出对触发输入事件的响应。 0：CC1 的正常操作依赖于计数器与 CCR1 的值，即使工作于触发器状态。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期 1：输入到触发器的有效沿的作用等同于发生了一次比较匹配。因此，OC 被设置为比较电平（独立于比较结果）。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
BIT[1:0]	CC1S	捕获/比较 1 选择 (Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入信号的选择： 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 (由 TIMx_SMCR 寄存器的 TS 位选择) *注：CC1S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC1E=0) 才是可写的。*

输入捕获模式：

BIT[15:12]	IC2F	输入捕获 2 滤波器
BIT[11:10]	IC2PSC	输入捕获 2 预分频
BIT[9:8]	CC2S	捕获/比较 2 选择（capture/compare 2 selection） 该 2 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 TIMx_SMCR 寄存器的 TS 位选择)。 *注：CC2S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC2E=0) 才是可写的。*
BIT[7:4]	IC1F[3:0]	输入捕获 1 滤波器 (Input capture 1 filter) 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 f_{DTS} 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：当 ICxF[3:0]=1、2 或 3 时，公式中的 fDTS 由 CK_INT 替代。*
BIT[3:2]	IC1PSC	输入捕获 1 预分频器 这 2 位定义了 CC1 输入(IC1) 的预分频系数。一旦 CC1E=0(TIMx_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
BIT[1:0]	CC1S	捕获/比较 1 选择 这 2 位定义通道的方向（输入/输出），及输入信号的选择： 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（由 TIMx_SMCR 寄存器的 TS 位选择）。 *注：CC1S 仅在通道关闭时（TIMx_CCER 寄存器的 CC3E=0）才是可写的。*

13.4.9 TIM1 捕获/比较模式寄存器 2（TIM1_CCMR2）

TIM1_CCMR2			地址偏移	0x1C		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]	
	IC4F[3:0]				IC4PSC[1:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
	IC4F[3:0]				IC3PSC[1:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

注：CCxS 位定义通道方向，可用于输入（捕获模式）或输出（比较模式）。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能，ICxx 描述了通道 在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

输出比较模式：

BIT[15]	OC4CE	输出比较 4 清 0 允许
BIT[14:12]	OC4M[2:0]	输出比较 4 模式
BIT[11]	OC4PE	输出比较 4 预分频允许
BIT[10]	OC4FE	输出比较 4 快速使能
BIT[9:8]	CC4S[1:0]	捕获/比较 4 选择（capture/compare 4 selection） 该 2 位定义通道的方向（输入/输出），及输入信号的选择： 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（由 TIMx_SMCR 寄存器的 TS 位选择） *注：CC4S 仅在通道关闭时（TIMx_CCER 寄存器的 CC4E=0）才是可写的。*
BIT[7]	OC3CE	输出比较 3 清 0 允许（output compare 3 clear enable）
BIT[6:4]	OC3M[2:0]	输出比较 3 模式（Output Compare 3 mode）
BIT[3]	OC3PE	输出比较 3 预分频允许（Output Compare 3 preload enable）
BIT[2]	OC3FE	输出比较 3 快速使能（Output Compare 3 fast enable）
BIT[1:0]	CC3S[1:0]	捕获/比较 3 选择（Capture/Compare 3 selection） 该 2 位定义通道的方向（输入/输出），及输入信号的选择： 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 TIMx_SMCR 寄存器的 TS 位选择）。 *注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0) 才是可写的。*

输入捕捉模式：

BIT[15:12]	IC4F[3:0]	输入捕获 4 滤波器
BIT[11:10]	IC4PSC	输入捕获 4 预分频
BIT[9:8]	CC4S	捕获/比较 4 选择 这 2 位定义通道的方向（输入/输出），及输入信号的选择： 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（由 TIMx_SMCR 寄存器的 TS 位选择）。 *注：CC4S 仅在通道关闭时（TIMx_CCER 寄存器的 CC4E=0）才是可写的。*
BIT[7:4]	IC3F	输入捕获 3 滤波器
BIT[3:2]	IC3PSC	输入比较 3 预分频
BIT[1:0]	CC3S	捕获/比较 3 选择 这 2 位定义通道的方向（输入/输出），及输入信号的选择： 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 TIMx_SMCR 寄存器的 TS 位选择）。 *注：CC3S 仅在通道关闭时（TIMx_CCER 寄存器的 CC3E=0）才是可写的。*

13.4.10 TIM1 捕获/比较使能寄存器（TIM1_CCER）

TIM1_CCER			地址偏移	0x20		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E
R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:14]	-	保留，保持复位值
BIT[13]	CC4P	捕获/比较 4 输出极性 **CC4 通道配置为输出：** 0：OC4 高电平有效； 1：OC4 低电平有效。 **CC4 通道配置为输入：** CC4P 位选择在触发或捕捉模式下 TI3FP4 和 TI4FP4 的有效极性。 0：非反相/上升沿 电路作用于 TIxFP4 的 上升沿（在复位、外部时钟或触发模式下的捕捉或触发操作），TIxFP4 非反相（在门控模式或编码模式）。 1：反相/下降沿 电路作用于 TIxFP4 的 下降沿（在复位、外部时钟或触发模式下的捕捉或触发操作），TIxFP4 反相（在门控模式或编码模式）。 *注：通道 4 不支持双沿选择*
BIT12]	CC4E	捕获/比较 4 输出使能，详见 CC1E
BIT[11]	CC3NP	捕获/比较 3 互补输出极性，详见 CC1NP
BIT[10]	CC3NE	捕获/比较 3 互补输出使能，详见 CC1NE
BIT[9]	CC3P	捕获/比较 3 输出极性，详见 CC1P
BIT[8]	CC3E	捕获/比较 3 输出使能，详见 CC1E
BIT[7]	CC2NP	捕获/比较 2 互补输出极性，详见 CC1NP
BIT[6]	CC2NE	捕获/比较 2 互补输出使能，详见 CC1NE
BIT[5]	CC2P	捕获/比较 2 输出极性，详见 CC1P
BIT[4]	CC2E	捕获/比较 2 输出使能，详见 CC1E

BIT[3]	CC1NP	捕获/比较 1 互补输出极性（Capture/Compare 1 complementary output polarity） **CC1 通道配置为输出** 0：OC1N 高电平有效； 1：OC1N 低电平有效。 **CC1 通道配置为输入** 该位用于和 CC1P 联合定义 TI1FP1 和 TI2FP1 的极性，详见 CC1P *注 1：在通道有互补输出时，该位被预装载。如果寄存器 TIMx_CR2 的 CCPC 被置 1 ，只有在通信事件(COM)发生时，CC1NP 的有效位才能从预装载位得到新值。* *注 2：一旦 LOCK 级别 (TIMx_BDTR 寄存器中的 LOCK 位) 设为 3 或 2 且 CC1S=00(通道配置为输出) ，则该位不能被修改。*
BIT[2]	CC1NE	捕获/比较 1 互补输出使能（Capture/Compare 1 complementary output enable） 0：关闭－OC1N 未激活，OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 这些位的功能。 1：开启－OC1N 信号输出到对应的输出引脚，OC1N 的输出依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位这些位的功能。 *注：在通道有互补输出时，该位被预装载。如果寄存器 TIMx_CR2 的 CCPC 被置 1，只有在通信事件（COM）发生时，CC1NE 的有效位才能从预装载位得到新值。*
BIT[1]	CC1P	捕获/比较 1 输出极性（Capture/Compare 1 output polarity） **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：** CC1NP/CC1P 位选择在触发或捕获模式下 TI1FP1 和 TI2FP1 的有效极性。 00：非反相/上升沿 - TIxFP1 的上升沿（在复位、外部时钟或触发模式下的捕获或触发操作） - TIxFP1 非反相（在门控模式或编码模式） 01：反相/下降沿 - TIxFP1 的下降沿（在复位、外部时钟或触发模式下的捕获或触发操作）， - TIxFP1 反相（在门控模式或编码模式） 00：保留，此配置不用 11：非反相/上升或下降沿 - TIxFP1 的上升沿和下降沿（在复位、外部时钟或触发模式下的捕获或触发操作） - TIxFP1 非反相（在门控模式） - 在编码模式下不能使用此配置 *注 1：在通道有互补输出时，本位被预装载。如果寄存器 TIMx_CR2 的 CCPC 被置 1 ，只有在通信事件（COM）发生时，CC1P 的有效位才能从预装载位得到新值。* *注：一旦 LOCK 级别 (TIMx_BDTR 寄存器中的 LOCK 位) 设为 3 或 2，则该位不能被修改。*
BIT[0]	CC1E	捕获/比较 1 输出使能(Capture/Compare 1 output enable) **CC1 通道配置为输出：** 0：关闭－OC1 未激活， OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 这些位的功能。 1：开启－OC1 信号输出到对应的输出引脚，其输出依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 这些位的功能。 **CC1 通道配置为输入：** 本位用于决定捕获的计数器值是否要装载到捕获/比较寄存器 1(TIMx_CCR1)。 0：捕获禁止 1：捕获允许 *注：在通道有互补输出时，本位被预装载。如果寄存器 TIMx_CR2 的 CCPC 被置 1 ，只有在通信事件（COM）发生时，CC1E 的有效位才能从预装载位得到新值。*

表：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(注 1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出禁止（不由定时器驱动） OCx=0, OCx_EN=0	输出禁止（不由定时器驱动） OCxN=0 OCxN_EN=0
		0	0	1	输出禁止（不由定时器驱动） 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（不由定时器驱动） OCxN=0 OCxN_EN=0
		0	1	1	OCxREF + 极性 + 死区 OCx_ EN=1	OCxREF 互补（非 OCxREF）+极性 + 死区 OCxN_EN=1
		1	0	0	输出禁止（不由定时器驱动） OCx=CCxP OCx_EN=0	输出禁止（不由定时器驱动） OCxN=CCxNP OCxN_EN=0
		1	0	1	Off-State（输出使能但无效态） 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	Off-State（输出使能但无效状态） OCxN=CCxNP OCxN_EN=1
		1	1	1	OCxREF + 极性 + 死区 OCx_ EN=1	OCxREF 互补（非 OCxREF）+极性 + 死区 OCxN_EN=1
0	0	X	0	0	输出禁止（不由定时器驱动） OCx=CCxP OCx_EN=0	输出禁止（不由定时器驱动） OCxN=CCxNP OCxN_EN=0
	0		0	1	输出禁止（不由定时器驱动）	
	0		1	0	异步：OCx=CCxP，OCx_EN=0，OCxN=CCxNP，OCxN_EN=0	
	0		1	1	若时钟存在：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		0	0	输出禁止（不由定时器驱动） OCx=CCxP OCx_EN=0	输出禁止（不由定时器驱动） OCxN=CCxNP OCxN_EN=0
	1		0	1	Off-State（输出使能但无效状态）	
	1		1	0	异步：OCx=CCxP，OCx_EN=1，OCxN=CCxNP，OCxN_EN=1	
	1		1	1	若时钟存在：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。	

注 1：当两个通道都未使用时（CCxE=CCxNE=0），OISx、OISxN、CCxP 和 CCxNP 位必须保持为 0。
注 2：连接到互补 OCx 和 OCxN 通道的外部 I/O 引脚的状态取决于 OCx 和 OCxN 通道状态和 GPIO 寄存器。

13.4.11 TIM1 计数器 (TIM1_CNT)

TIM1_CNT			地址偏移	0x24		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CNT[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CNT[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	CNT[15:0]	计数器值

13.4.12 TIM1 预分频器 (TIM1_PSC)

TIM1_PSC			地址偏移	0x28		复位值	0x0000 0000
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	PSC[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	PSC[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	PSC[15:0]	预分频值(Prescaler value) 计数器的时钟频率 (CK_CNT) = f_{CK_PSC}/(PSC[15:0]+1)。每次当更新事件产生时，PSC 的值被装入当前预分频器寄存器；更新事件包括计数器被 TIM_EGR 的 UG 位清‘0’或被配置为复位模式的触发器控制器清‘0’。

13.4.13 TIM1 自动重装载寄存器 (TIM1_ARR)

TIM1_ARR			地址偏移	0x2C		复位值	0x0000 FFFF
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	ARR[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	1	1	1	1	1	1	1
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	ARR[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	1	1	1	1	1	1	1

BIT[15:0]	ARR[15:0]	自动重装载的值 (Auto-reload value) ARR 包含了将要装载入的实际自动重装载寄存器的值。详细参考章节：14.3.1 时基单元，有关 ARR 的更新和行为。 当自动重装载的值为空时，计数器不工作。

13.4.14 TIM1 重复计数寄存器 (TIM1_RCR)

TIM1_RCR			地址偏移	0x30		复位值	0x0000 0000
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	REP[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:8]	-	保留，保持复位值
BIT[7:0]	REP[7:0]	重复计数器的值 (Repetition counter value) 预装载寄存器被使能后，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传输到当前活动寄存器)；如果允许产生更新中断，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 只有在重复更新事件 U_RC 发生时才重载 REP 值，因此对 TIMx_RCR 寄存器写入的新值只在下次重复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。

13.4.15 TIM1 捕获/比较寄存器 1 (TIM1_CCR1)

TIM1_CCR1			地址偏移	0x34		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CCR1[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CCR1[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	CCR1[15:0]	捕获/比较通道 1 的值 **若 CC1 通道配置为输出：** CCR1 是要加载到实际捕获/比较 1 寄存器的值 (预装载值)。 如果在 TIMx_CCMR1 寄存器 (OC1PE 位) 中未选择预装载功能，写入的数值会永久加载。否则只有当更新事件发生时，此预装载值才加载至内部活动的捕获/比较 1 寄存器中。 活动的捕获/比较寄存器与计数器 TIMx_CNT 进行比较,并在 OC1 端口上产生输出信号。 **若 CC1 通道配置为输入：** CCR1 是上一次输入捕获 1 事件 (IC1) 捕获的计数器值。

13.4.16 TIM1 捕获/比较寄存器 2 (TIM1_CCR2)

TIM1_CCR2			地址偏移	0x38		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CCR2[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CCR2[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	CCR2[15:0]	捕获/比较通道 2 的值 **若 CC2 通道配置为输出：** CCR2 是要加载到实际捕获/比较 2 寄存器的值 (预装载值)。 如果在 TIMx_CCMR2 寄存器 (OC2PE 位) 中未选择预装载功能，写入的数值会永久加载。否则只有当更新事件发生时，此预装载值才加载至内部活动的捕获/比较 2 寄存器中。 活动的捕获/比较寄存器与计数器 TIMx_CNT 进行比较，并在 OC2 端口上产生输出信号。 **若 CC2 通道配置为输入：** CCR2 是上一次输入捕获 2 事件 (IC2) 捕获的计数器值。

13.4.17 TIM1 捕获/比较寄存器 3 (TIM1_CCR3)

TIM1_CCR3			地址偏移	0x3C		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CCR3[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

控制位	CCR3[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	CCR3[15:0]	捕获/比较通道 3 的值 **若 CC3 通道配置为输出：** CCR3 是要加载到实际捕获/比较 3 寄存器的值 （ 预装载值 ）。 如果在 TIMx_CCMR3 寄存器 (OC3PE 位) 中未选择预装载功能，写入的数值会永久加载。否则只有当更新事件发生时，此预装载值才加载至内部活动的捕获/比较 3 寄存器中。 活动的捕获/比较寄存器与计数器 TIMx_CNT 进行比较，并在 OC3 端口上产生输出信号。 **若 CC3 通道配置为输入：** CCR3 是上一次输入捕获 3 事件 (IC3) 捕获的计数器值。

13.4.18 TIM1 捕获/比较寄存器 4 (TIM1_CCR4)

TIM1_CCR4			地址偏移	0x40		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CCR4[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CCR4[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	CCR4[15:0]	捕获/比较通道 4 的值 **若 CC4 通道配置为输出：** CCR4 是要加载到实际捕获/比较 4 寄存器的值 （ 预装载值 ）。 如果在 TIMx_CCMR4 寄存器 (OC4PE 位) 中未选择预装载功能，写入的数值会永久加载。否则只有当更新事件发生时，此预装载值才加载至内部活动的捕获/比较 4 寄存器中。 活动的捕获/比较寄存器与计数器 TIMx_CNT 进行比较,并在 OC4 端口上产生输出信号。 **若 CC4 通道配置为输入：** CCR4 是上一次输入捕获 4 事件 (IC4) 捕获的计数器值。

13.4.19 TIM1 刹车和死区寄存器 (TIM1_BDTR)

TIM1_BDTR			地址偏移	0x44		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]	
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	DTG[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

注：根据锁定设置， AOE、BKP、BKE、OSSI 、OSSR 和 DTG[7:0] 位均可被写保护，需要在第一次写入 TIMx_BDTR 寄存器时对它们进行配置。

BIT[15]	MOE	主输出使能 (Main output enable) 一旦刹车输入有效，该位被硬件异步清‘0’。根据 AOE 位的设置值，该位可以由软件置 1 或被自动置 1。它仅对配置为输出的通道有效。 0：OC 和 OCN 输出禁止 或 强制为空闲状态； 1:OC 和 OCN 输出开启,同时须设置了相应的使能位 (TIMx_CCER 寄存器的 CCxE、CCxNE 位) *注：有关 OC/OCN 使能的细节，参见章节：13.4.10 TIM1 捕获/比较使能寄*

		存器(TIMx_CCER)。
BIT[14]	AOE	自动输出使能 (Automatic output enable) 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被自动置‘1’(如果刹车输入未生效)。 *注：一旦 LOCK 级别 (TIMx_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。*
BIT[13]	BKP	刹车输入极性 (Break polarity) 0：刹车输入 BRK 低电平有效 1：刹车输入 BRK 高电平有效 *注 1：一旦 LOCK 级别 (TIMx_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。* *注 2：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。*
BIT[12]	BKE	刹车使能 (Break enable) 0: 刹车输入禁止 (BRK 和 CCS 时钟失效事件) 1; 刹车输入允许 (BRK 和 CCS 时钟失效事件) *注 1：一旦 LOCK 级别 (TIMx_BDTR 寄存器中的 LOCK 位)设为 1，则该位不能被修改。* *注 2：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。*
BIT[11]	OSSR	运行模式下“关闭状态”选择 (Off-state selection for Run mode) 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 *注：有关 OC/OCN 使能的细节，参见章节：13.4.10 TIM1 捕获/比较使能寄存器(TIMx_CCER)。* 0：当无效时，OC/OCN 禁止输出 (OC/OCN 使能输出信号 =0) 1:当无效时,一旦 CCxE=1 或 CCxNE=1,OC/OCN 使能并输出无效电平，且 OC/OCN 使能输出信号 =1。 *注：一旦 LOCK 级别 (TIMx_BDTR 寄存器中的 LOCK 位)设为’2’，则该位不能被修改。*
BIT[10]	OSSI	空闲模式下“关闭状态”选择 (Off-state selection fo Idle mode) 该位用于当 MOE=0 时且通道配置为输出。 *注：有关 OC/OCN 使能的细节，参见章节：13.4.10 TIM1 捕获/比较使能寄存器(TIMx_CCER)。* 0：当无效时，OC/OCN 输出禁止 (OC/OCN 使能输出信号 =0) 1：当无效时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 首先被强制输出空闲电平， ，OC/OCN 使能输出信号 =1 *注：一旦 LOCK 级别 (TIMx_BDTR 寄存器中的 LOCK 位)设为’2’，则该位不能被修改。*
BIT[9:8]	LOCK[1:0]	锁定设置 (Lock configuration) 该位提供一个写保护，以防止软件错误。 00：锁定关闭，寄存器无写保护 01:锁定级别 1,不能写入 TIMx_BDTR 寄存器的 DTG、BKE、BKP、AOE 位和 TIMx_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 (TIMx_CCER 寄存器的 CCxP/CCNxP 位，一旦相关通道通过 CCxS 位设为输出)，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 (TIMx_CCMRx 寄存器的 OCxM/OCxPE 位，一旦相关通道通过 CCxS 位设为输出) *注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMx_BDTR 寄存器，则其内容冻结直至复位。*
BIT[7:0]	DTG[7:0]	死区发生器设置 (Dead-time generator setup) 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}，其中 T_{dtg} = T_{DTS}; DTG[7:5]=10x => DT=(64+DTG[5:0]) × T_{dtg}，其中 T_{dtg} = 2×T_{DTS}; DTG[7:5]=110 => DT=(32+DTG[4:0]) × T_{dtg}，其中 T_{dtg} = 8 × T_{DTS}; DTG[7:5]=111 => DT=(32+DTG[4:0])× T_{dtg}，其中 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（TIMx_BDTR 寄存器中的 LOCK 位)设为 1 、2 或 3，则不能修改这些位。*

13.4.20 TIM1 DMA 控制寄存器 (TIM1_DCR)

TIM1_DCR			地址偏移	0x48		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	DBL[4:0]				
R/W	-	-	-	rw	rw	rw	rw	rw
复位值	-	-	-	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	DBA[4:0]				
R/W	-	-	-	rw	rw	rw	rw	rw
复位值	-	-	-	0	0	0	0	0

BIT[15:13]	-	保留，保持复位值
BIT[12:8]	DBL[4:0]	DMA 突发传送长度 (DMA burst length) 这 5 位定义了 DMA 传送长度 (当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次突发传送)， 00000：1 次传输 00001：2 次传输 00010：3 次传输 ..... 10001：18 次传输
BIT[7:5]	-	保留，保持复位值
BIT[4:0]	DBA[4:0]	DMA 基地址 (DMA base address) 这 5 位定义了 DMA 传送的基地址 (当对 TIMx_DMAR 寄存器进行读或写时)， DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 例如： 00000：TIMx_CR1 00001：TIMx_CR2 00010：TIMx_SMCR ..... 例：要完成如下的传输：DBL = 7 ，DBA = TIMx_CR1 此时传送从 TIMx_CR1 的地址开始，发送/读取连续 7 个寄存器。

13.4.21 TIM1 全部传输时 DMA 地址 (TIM1_DMAR)

TIM1_DMAR			地址偏移	0x4C		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	DMAB[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	DMAB[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	DMAB[15:0]	DMA 突发传送寄存器 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的访问： （TIMx_CR1 地址） +（DBA + DMA 索引）x 4， 其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1) 所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量（0~DBL），(DBL 在 TIMx_DCR 寄存器中定义)。

举一个如何使用 DMA 突发操作的例子

本例中使用定时器 DMA 的突发功能，将 CCRx 寄存器 (x = 2, 3, 4) 的内容以半字方式进行 DMA 传输，更新到 CCRx 寄存器。按如下步骤进行操作：

1、 配置相关的 DMA 通道 :

- DMA 通道外设地址为 DMAR 寄存器地址
- DMA 通道存储器地址为 RAM 缓冲区地址，包含要通过 DMA 传送到 CCRx 寄存器的数据
- 传送数据数量 = 3 (见下面的注释)
- 循环模式禁用

2、 配置 DCR 寄存器的 DBA 和 DBL 位: DBL = 3 次传送 , DBA = 0xE。

3、 使能 TIMx 更新 DMA 请求 (置位 DIER 寄存器的 UDE 位)。

4、 使能 TIMx

5、 使能 DMA 通道

注：在本例中所 CCRx 寄存器被一次性全部更新。如果需要更新 CCRx 寄存器两次，传送的数据数量应该是 6。假定，RAM 缓冲区要包含 data1,data2,data3,data4,data5 和 data6。数据按如下过程传送到 CCRx 寄存器：在第一个更新 DMA 请求时，data1 被传送到 CCR2，data2 被传送到 CCR3，data3 被传送到 CCR4；在第二个 DMA 更新中断请求时，data4 被传送到 CCR2，data5 被传送到 CCR3，data6 被传送到 CCR4。

14 通用定时器（TIM2/3）

14.1 概述

TIM2 基于一个 32 位自动重载递增/递减计数器和一个 16 位预分频。TIM3 基于一个 16 位自动重载递增/递减计数器和一个 16 位预分频。它们都具有 4 个独立通道，用于输入捕获/输出比较、PWM、单脉冲模式输出。

TIM2 和 TIM3 通用定时器可通过定时器链接功能与 TIM1 高级控制定时器协同工作，提供同步或事件链接功能。

TIM2 和 TIM3 都可生成独立的 DMA 请求。

这些定时器能够处理正交（增量）编码器信号，也能处理 1 到 3 个霍尔效应传感器的数字输出。

在调试模式下，其计数器可被冻结。

14.2 特性

✧ 16 位（TIM3）或 32 位（TIM2）向上、向下、向上/下自动装载计数器

✧ 16 位可编程 （ 可实时修改 ） 预分频器，支持 1～65535 之间的任意分频

✧ 多达 4 个独立通道，支持

- 输入捕获
- 输出比较
- PWM 生成（边沿或中心对齐）
- 单脉冲模式输出

✧ 支持外部信号同步，支持内部多个 timer 同步互联

✧ 下述事件触发中断/DMA：

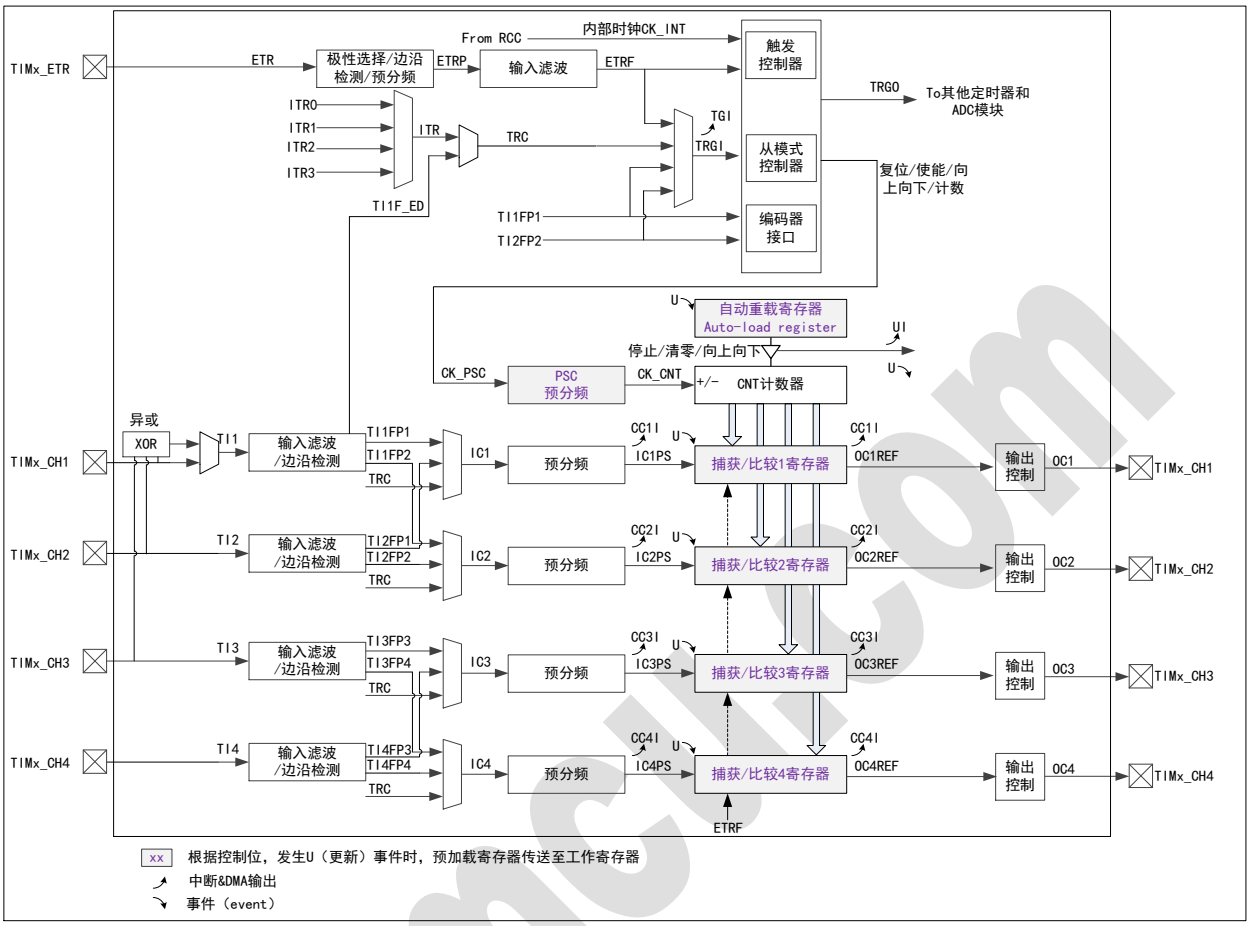
- 更新（Update）：计数器上溢/下溢，计数器初始化（通过软件或内部/外部触发）
- 触发事件（Trigger Event）：计数器启动、停止、初始化或由内部/外部触发计数
- 输入捕获（input capture）
- 输出比较（output compare）

✧ 支持增量式（正交）编码器和霍尔传感器定位解码功能

✧ 触发输入作为外部时钟或者按周期的电流管理

14.3 功能描述

TIM2/3 模块框图



14.3.1 时基单元

通用定时器 TIM2/3 主要由 32/16 位计数器及相关自动重载寄存器构成，计数器支持向上计数、向下计数或向上向下双向计数。计数器时钟支持预分频。

计数器寄存器、自动重载寄存器和预分频寄存器支持软件读写，即使计数器正在运行读写仍然有效。

时基单元包括：

- ✧ 计数器寄存器（TIMx_CNT）
- ✧ 预分频器寄存器（TIMx_PSC）
- ✧ 自动重载寄存器（TIMx_ARR）

自动重载寄存器是预装载的，读写自动重载寄存器将访问预装载寄存器。设置 TIMx_CR1 寄存器中的自动重载预装载使能位（ARPE），选择预装载寄存器的内容永久传送至缓存寄存器或在每次更新事件（UEV）传送至缓存寄存器。当计数器达到溢出条件（或向下计数时的下溢条件）并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可由软件产生。有关更新事件的产生，针对每种配置后续章节会详细描述。

计数器时钟由预分频后输出 CK_CNT 驱动，需置位 TIMx_CR1 寄存器中的计数器使能位（CEN）时，CK_CNT 才有效（请参阅从模式控制器描述以获得计数器使能的更多细节）。

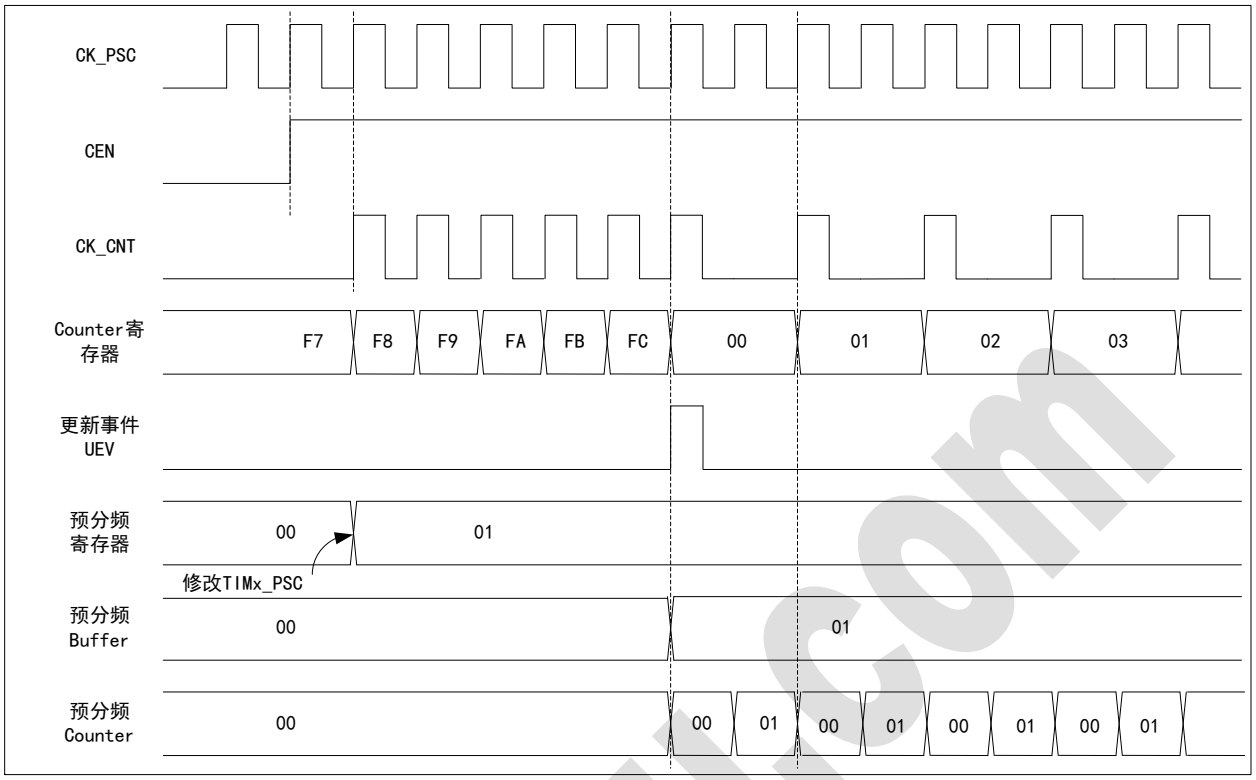
注：设置 TIMx_CR 寄存器的 CEN 位，一个时钟周期后，计数器才开始计数。

预分频

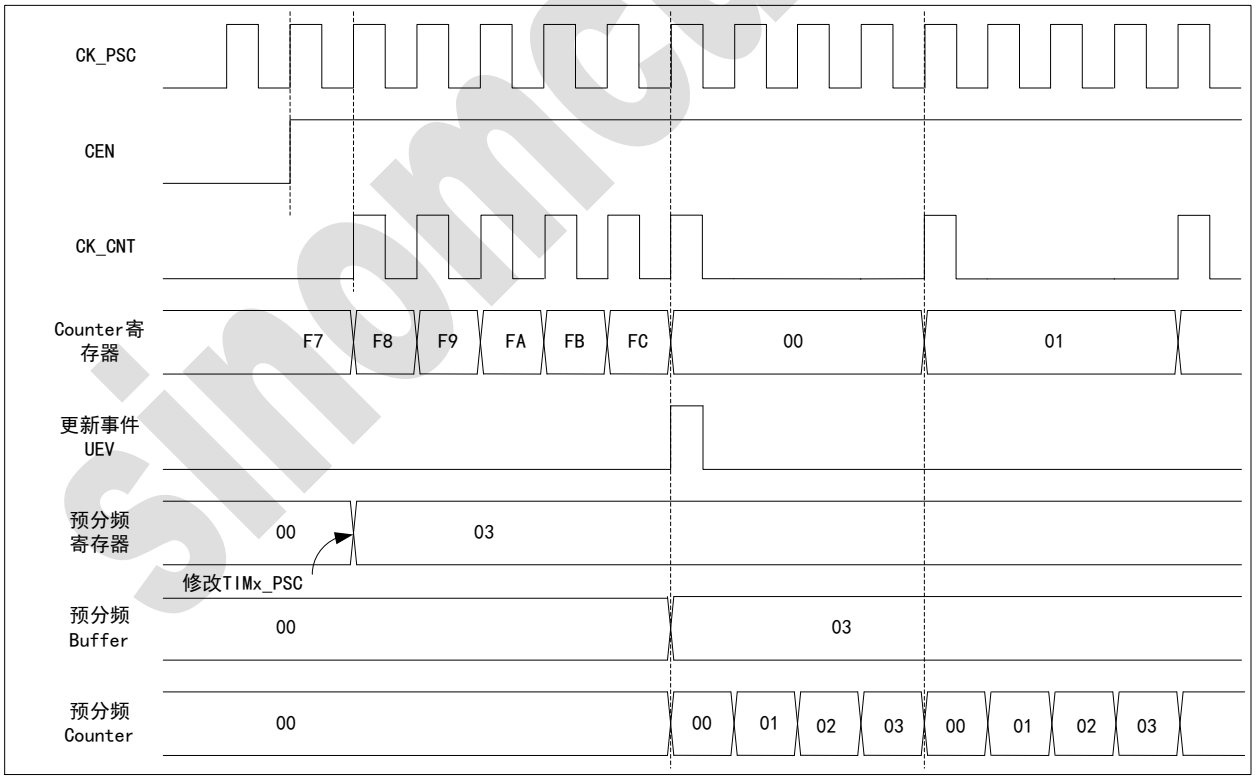
预分频器支持计数器时钟 1~65536 之间任意分频，基于 1 个 16 位的计数器，由 TIMx_PSC 寄存器中的 16 位寄存器控制。此寄存器内部带有缓存器，支持运行时修改；新修改的预分频值在下一次的更新事件（update event）发生时生效。

下图给出运行时更改预分频值，计数器行为。

计数器时序图：预分频修改，从 1 到 2



计数器时序图：预分频修改，从 1 到 4



14.3.2 计数器模式

向上计数模式

设置 TIMx_CR1 寄存器中的 CMS=00 且 DIR=0，执行计数器向上计数。

在向上计数模式中，计数器从 0 计数到自动重载值（TIMx_ARR 计数器的值 ），然后重新从 0 开始计数并且产生一个计数器溢出事件（overflow）。

每次计数器溢出时都会产生更新事件。

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

通过软件置位 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新缓存寄存器。在 UDIS 位被清‘0’之前，将不产生更新事件。但是，在应该产生更新事件时，计数器会被清‘0’，同时预分频器的计数器也被清 0(但预分频器的数值不变)。

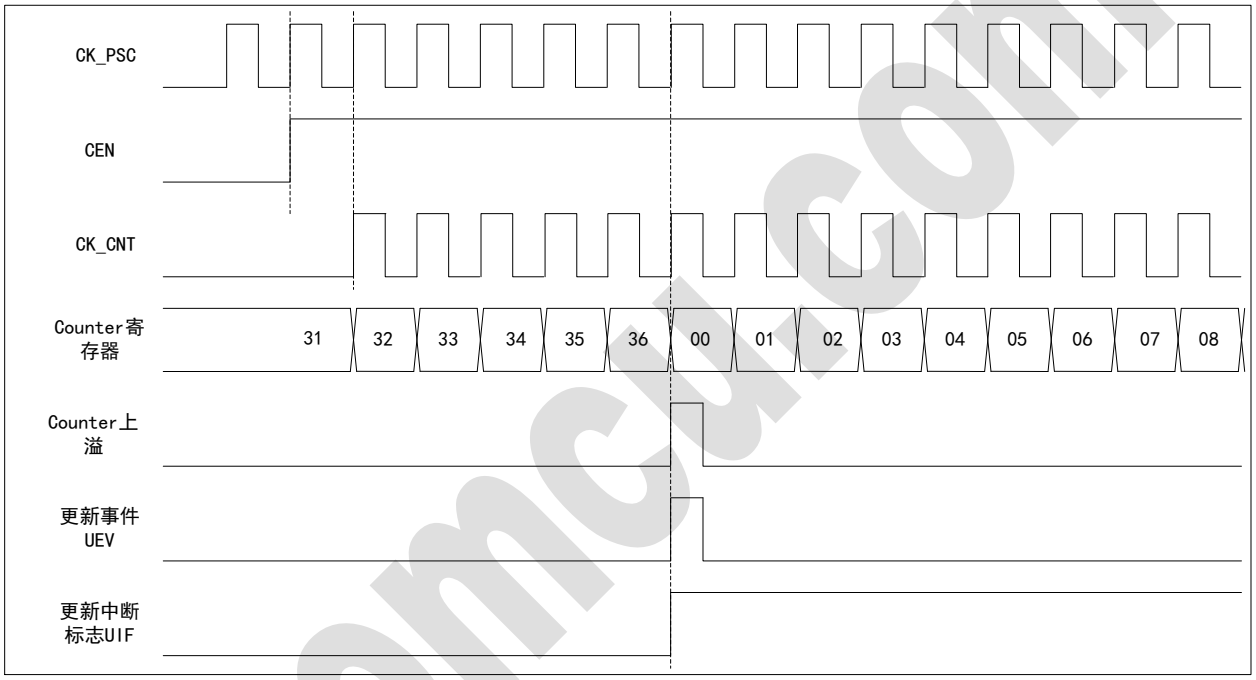
此外，如果置位 TIMx_CR1 寄存器中的 URS 位（update request selection），置位 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志（即不产生中断或 DMA 请求）。这是为了避免在发生捕获事件（capture event）清除计数器时，同时产生更新和捕获两个中断。

当发生一个更新事件时，所有的寄存器都被更新，同时（依据 URS 位）置位更新标志位（TIMx_SR 寄存器中的 UIF 位）：

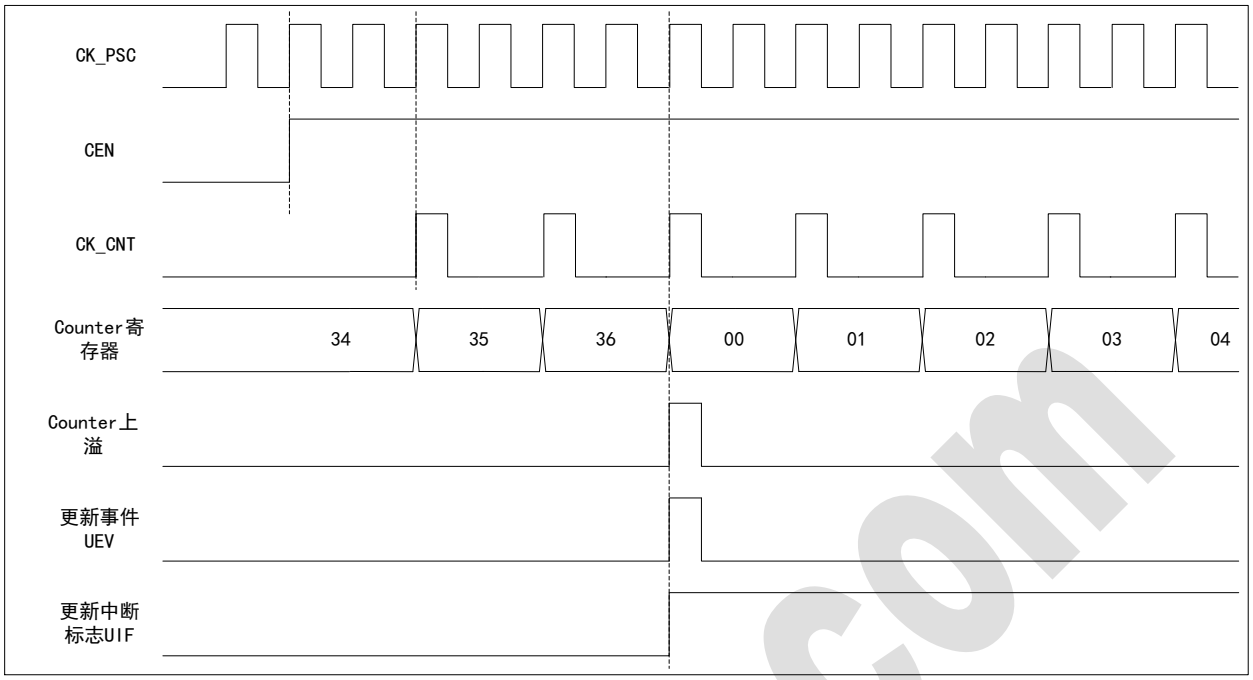
- ✧ 预分频器的缓冲区被写入预装载寄存器的值（TIMx_PSC 寄存器的值 ）
- ✧ 自动重载缓存寄存器更新为预装载寄存器的值（TIMx_ARR）

举例如下：TIMx_ARR=0x36 时，计数器在不同时钟频率下计数器行为。

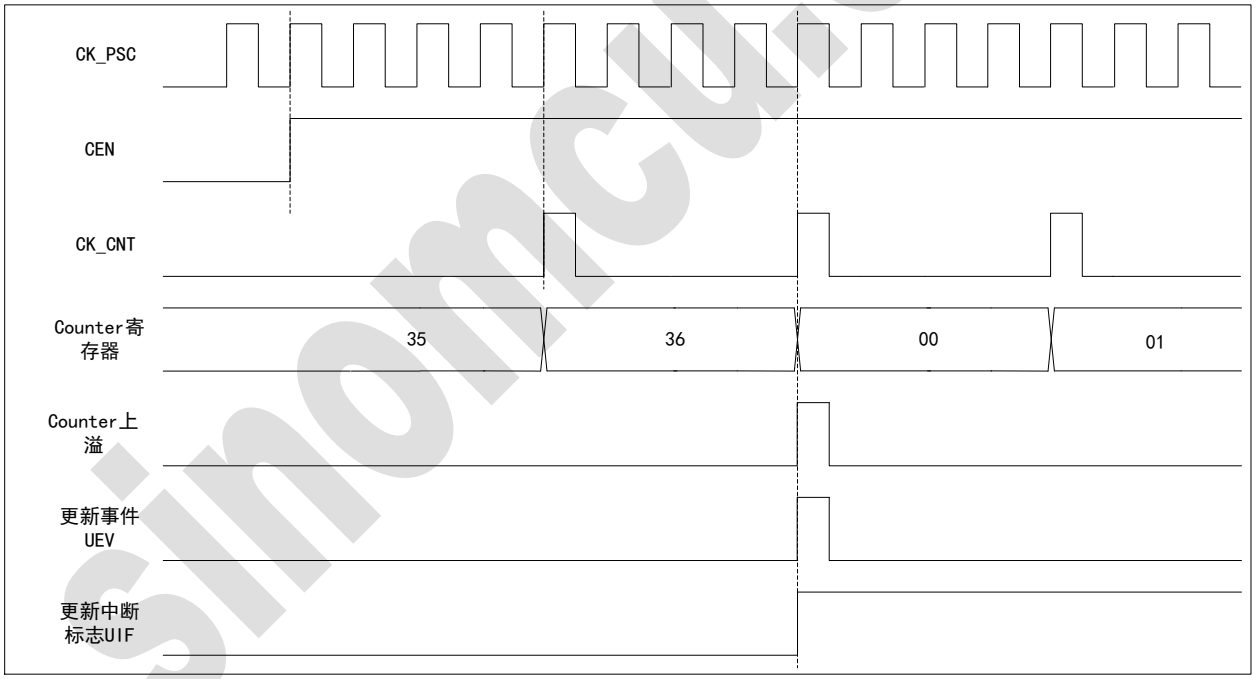
计数器时序图：预分频因子=1



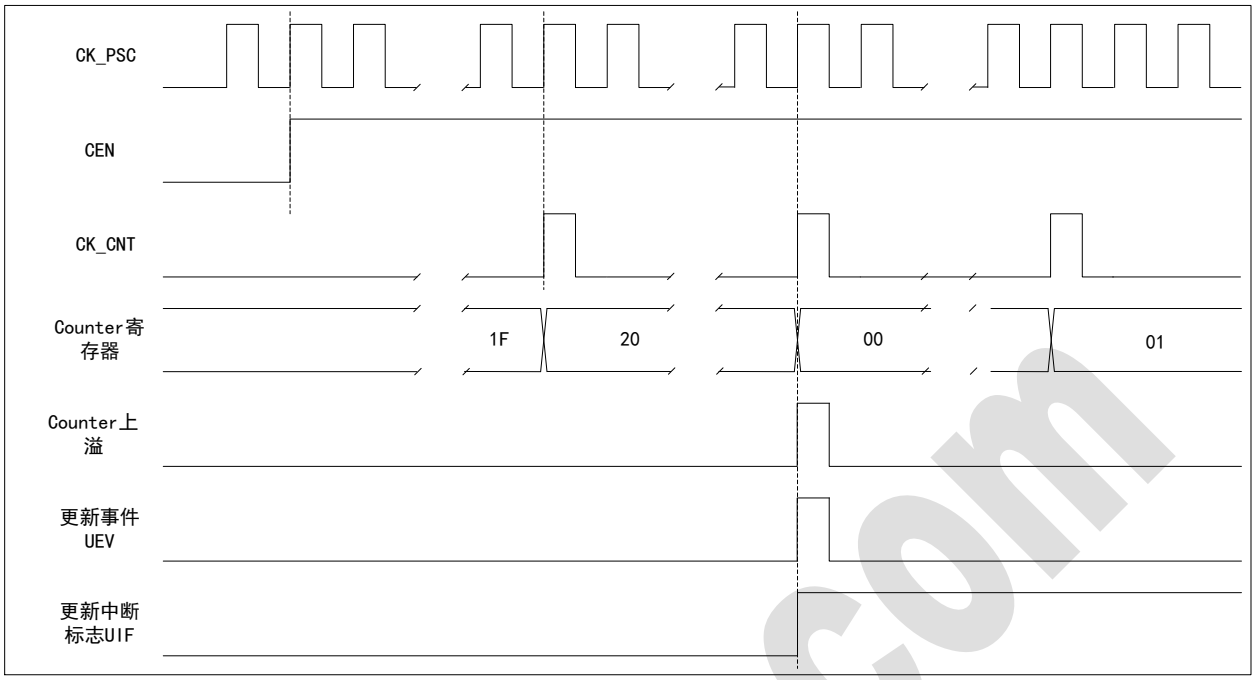
计数器时序图：预分频因子=2



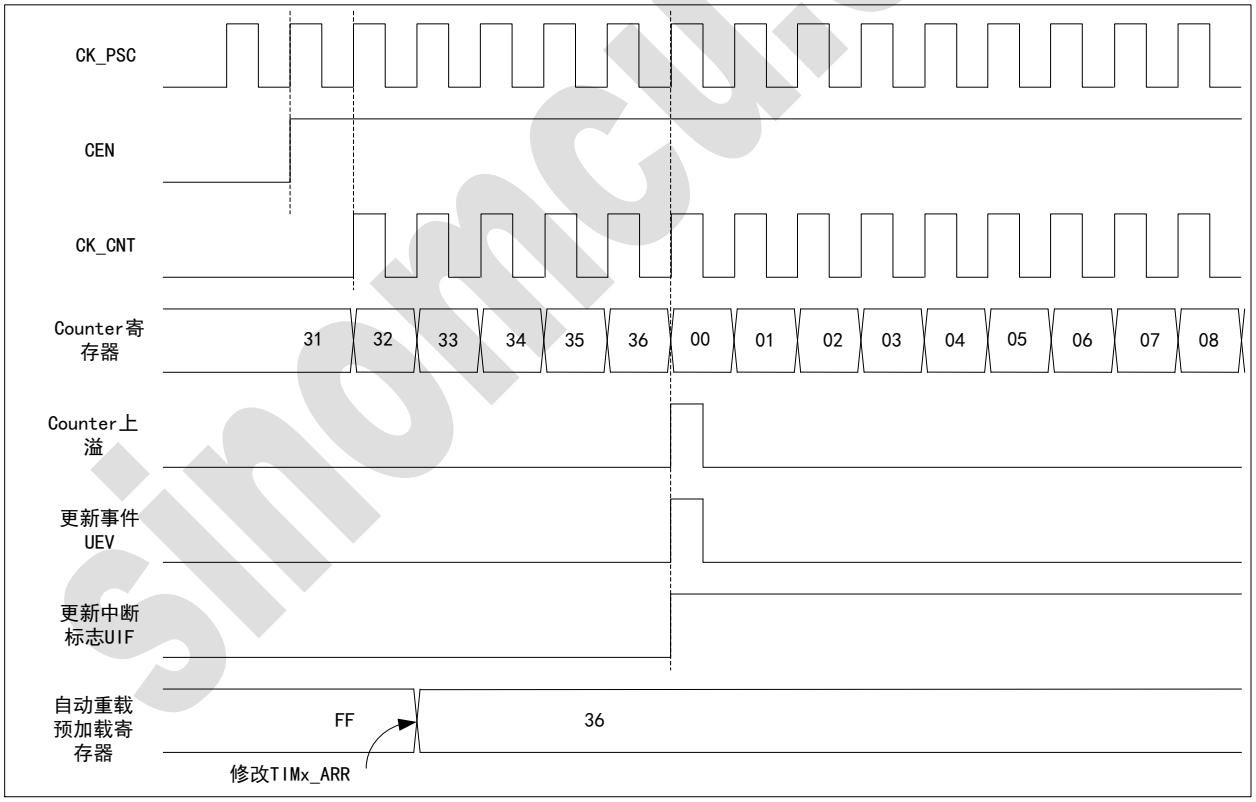
计数器时序图：预分频因子=4



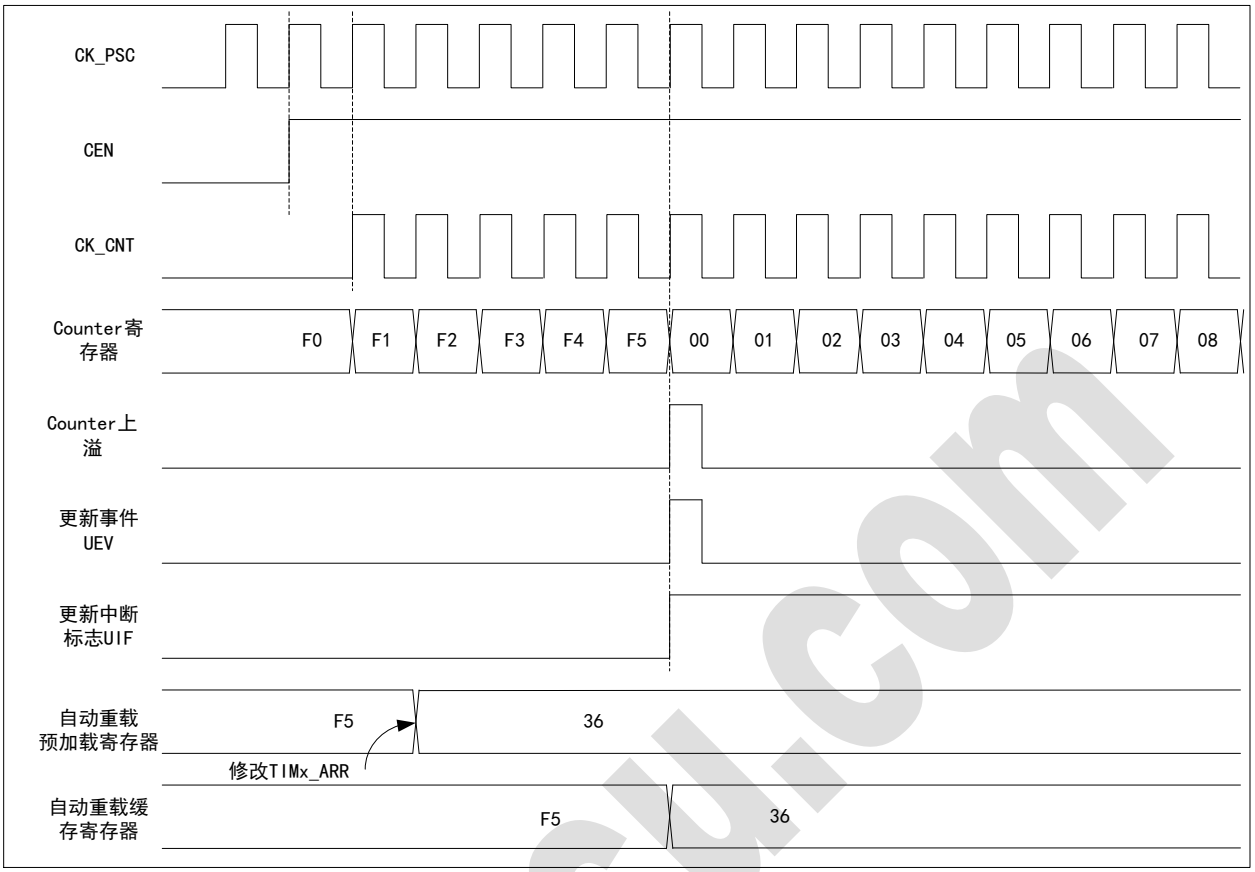
计数器时序图：预分频因子=N



计数器时序图：预分频因子=1，ARPE=0（TIMx_ARR 没有预装载）



计数器时序图：预分频因子=1，ARPE=1（TIMx_ARR 已预装载）



向下计数模式

设置 TIMx_CR1 寄存器中的 CMS=00 且 DIR=1，执行计数器向下计数。

在向下计数模式中，计数器从自动重载的值（TIMx_ARR 寄存器的值）开始向下计数到 0，然后从自动重载的值重新开始并且产生一个计数器向下溢出事件（underflow）。

每次计数器溢出时都会产生更新事件。

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

通过软件置位 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新缓存寄存器。在 UDIS 位被清‘0’之前，将不产生更新事件。但是，在应该产生更新事件时，计数器会被清‘0’，同时预分频器的计数器也被清 0（但预分频器的数值不变）。

此外，如果置位 TIMx_CR1 寄存器中的 URS 位（update request selection），置位 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志（即不产生中断或 DMA 请求）。这是为了避免在发生捕获事件（capture event）清除计数器时，同时产生更新和捕获两个中断。

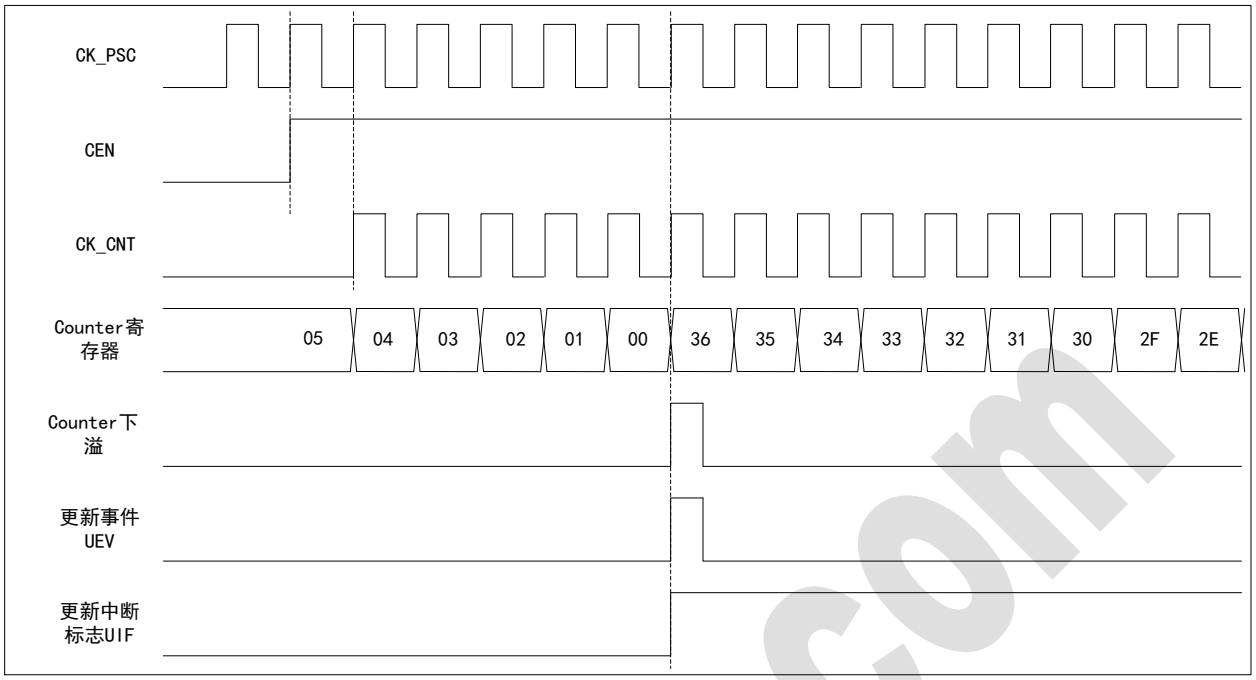
当发生一个更新事件时，所有的寄存器都被更新，同时（依据 URS 位）置位更新标志位（TIMx_SR 寄存器中的 UIF 位）：

- ✧ 预分频器的缓冲区被写入预装载寄存器的值（TIMx_PSC 寄存器的值）
- ✧ 自动重载缓存寄存器更新为预装载寄存器的值（TIMx_ARR）

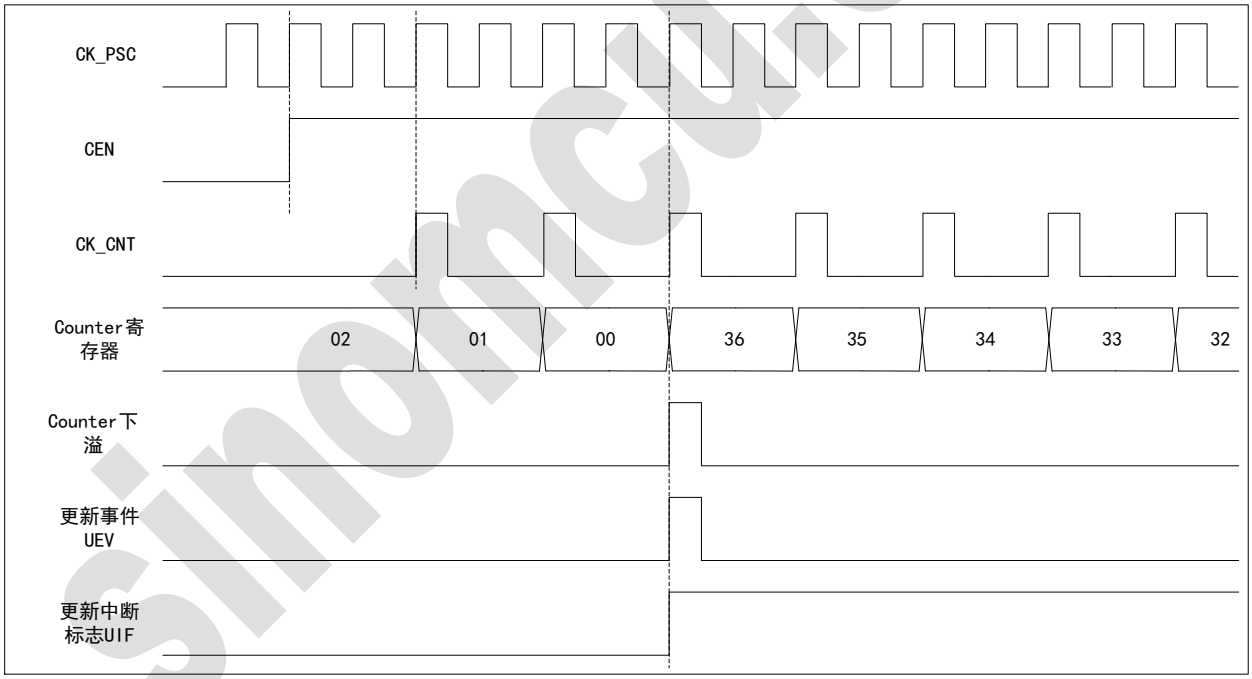
注：若在计数器重载之前自动重载值被更新，此时在下一个周期时才是预期的值。

举例如下：TIMx_ARR=0x36 时，计数器在不同时钟频率下计数器行为。

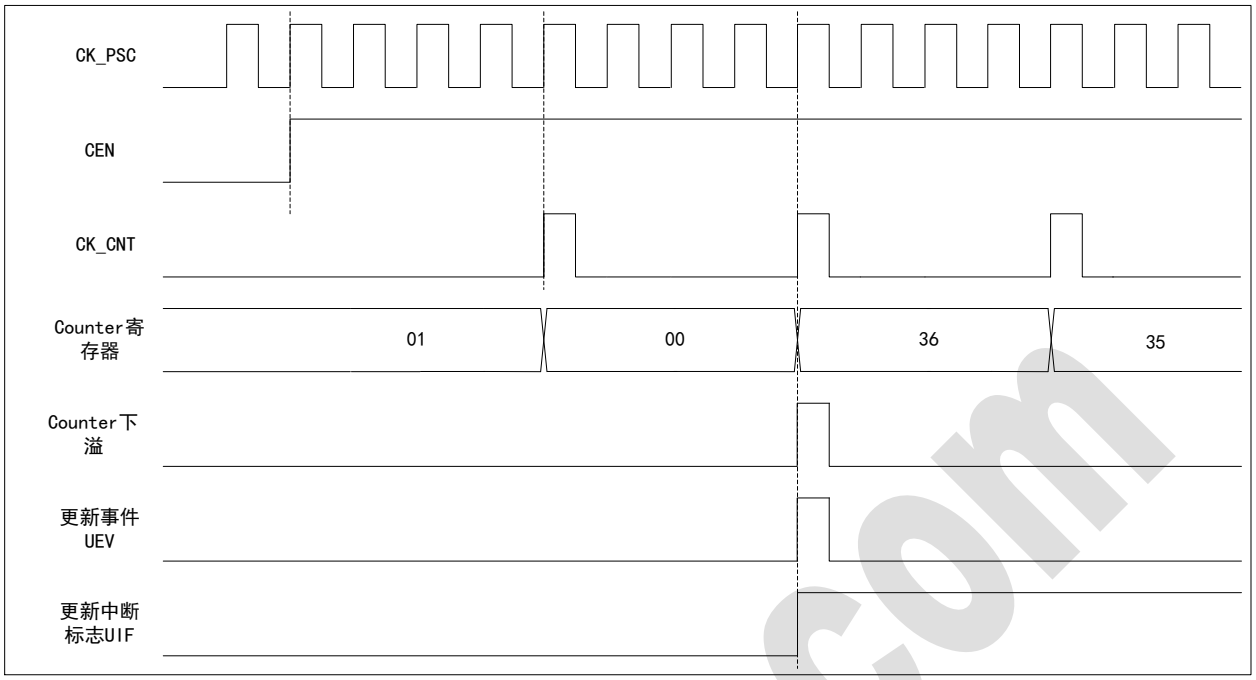
计数器时序图：预分频因子=1



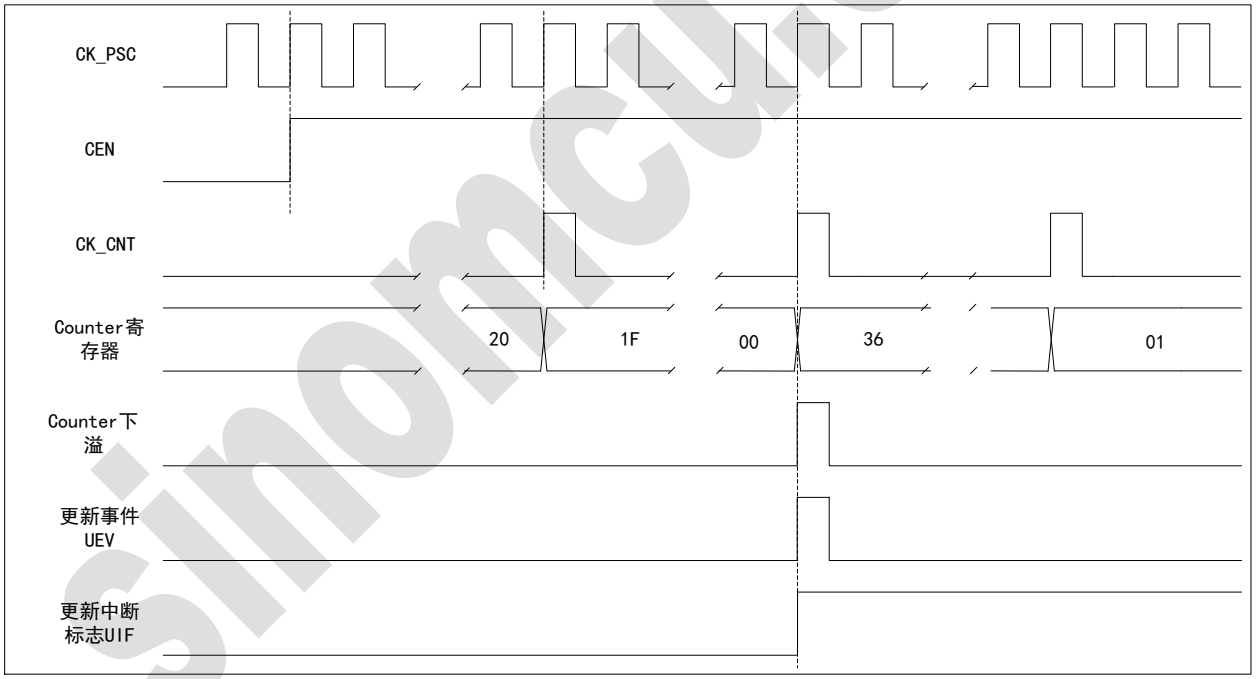
计数器时序图：预分频因子=2



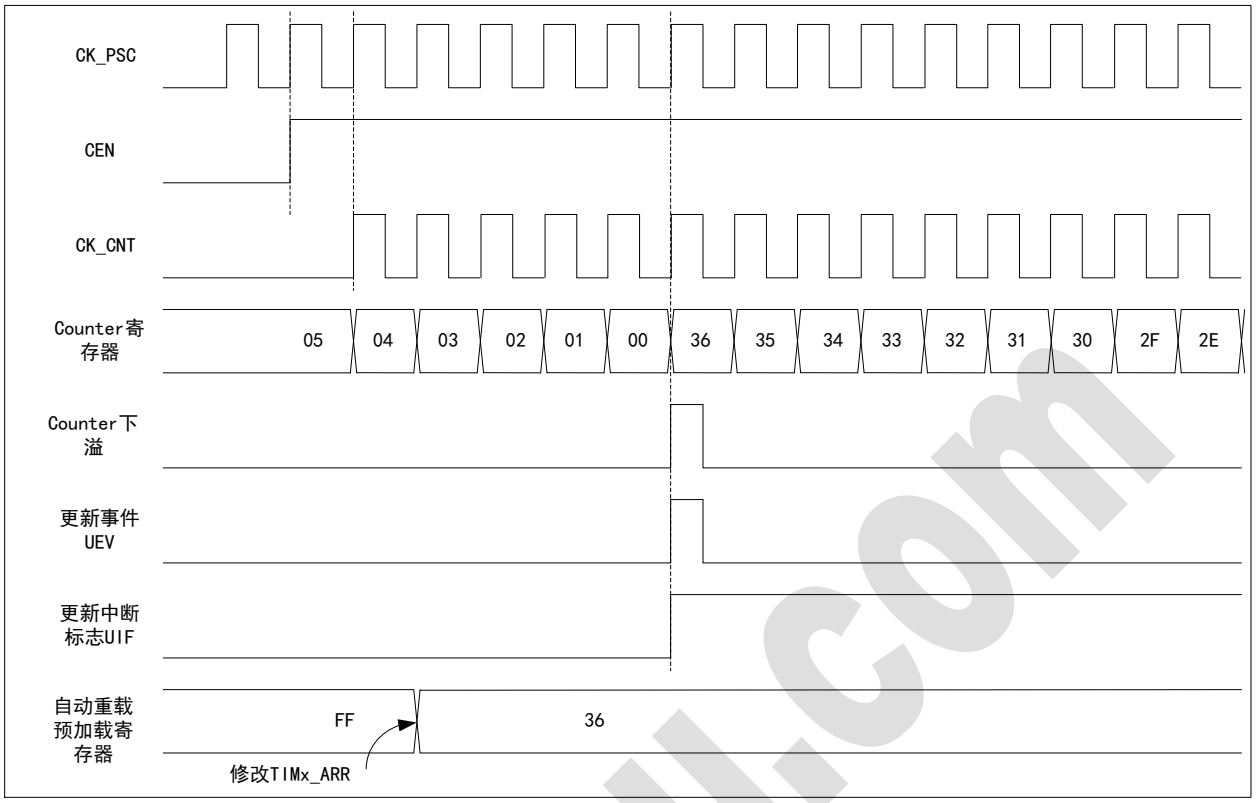
计数器时序图：预分频因子=4



计数器时序图：预分频因子=N



计数器时序图：预分频因子=1,重复次数计数器未使用



中心对齐模式（向上/向下计数）

在中央对齐模式,计数器从 0 开始计数到自动重载的值(TIMx_ARR 寄存器)－1,产生一个计数器溢出事件(overflow),然后从自动重载值向下计数到 1 并产生一个计数器下溢事件（underflow）；然后再从 0 开始重新计数。

当 TIMx_CR1 的 CMS 位不为"00"时，使能中央对齐模式。已配置为输出的通道，其输出比较中断标志在以下情况下被置位：计数器向下计数（中央对齐模式 1, CMS = "01"），计数器向上计数（中央对齐模式 2, CMS = "10"），计数器向下和向上计数（中央对齐模式 3, CMS = "11"）。

在此模式下，不能写入 TIMx_CR1 中的 DIR 方向位，它由硬件更新并指示当前的计数方向。

更新事件可以在每个计数器上溢和每个计数器下溢时产生；也可以通过(软件或者使用从模式控制器)置位 TIMx_EGR 寄存器中的 UG 位产生更新事件。在这种情况下，计数器以及预分频器的计数器将从 0 重新开始计数。

通过软件置位 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新缓存寄存器。在 UDIS 位被清‘0’之前，将不产生更新事件。**然而，计数器仍会根据当前自动重载的值，继续向上或向下计数。**

此外，如果置位 TIMx_CR1 寄存器中的 URS 位（update request selection），置位 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志（即不产生中断或 DMA 请求）。这是为了避免在发生捕获事件（capture event）清除计数器时，同时产生更新和捕获两个中断。

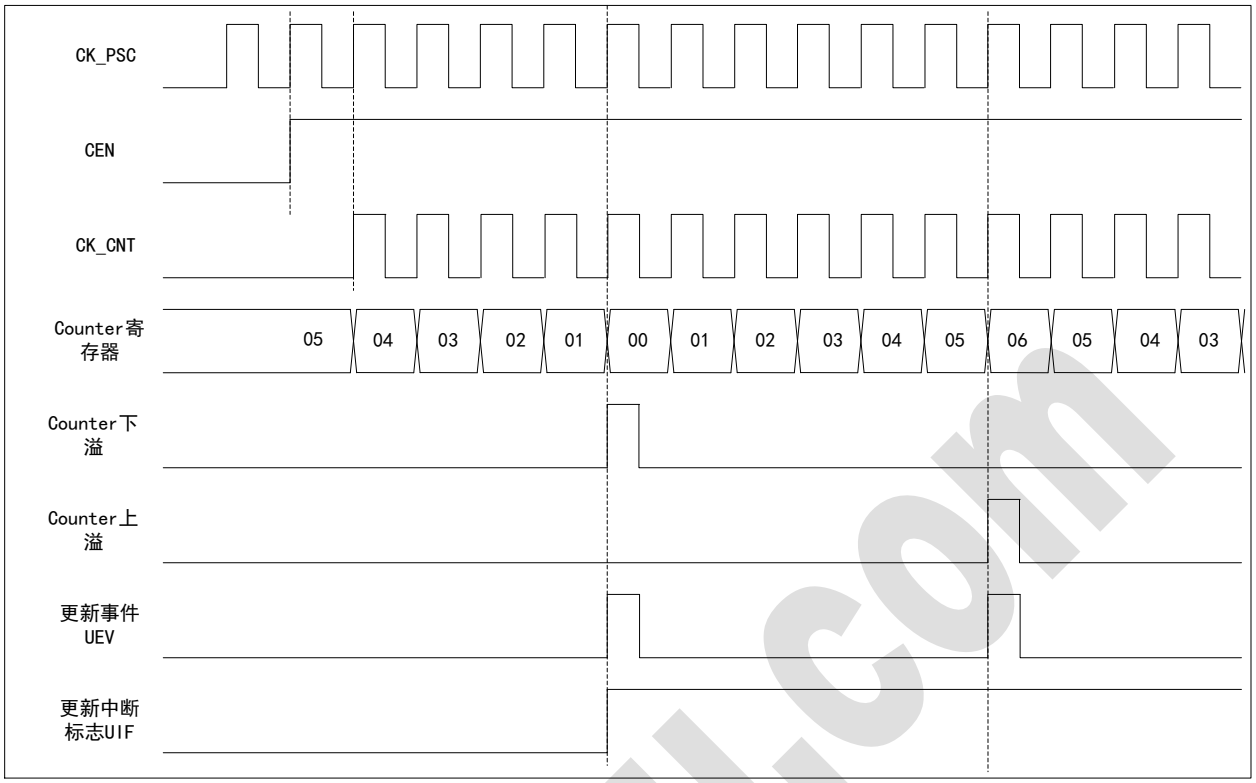
当发生一个更新事件时,所有的寄存器都被更新,同时(依据 URS 位)置位更新标志位(TIMx_SR 寄存器中的 UIF 位)：

- ✧ 预分频器的缓冲区被写入预装载寄存器的值（TIMx_PSC 寄存器的值 ）
- ✧ 自动重载缓存寄存器更新为预装载寄存器的值（TIMx_ARR）

注：若更新事件源为计数器上溢（overflow），在计数器重载之前自动重载值被更新，此时在下一个周期时才是预期的值（计数器被加载为新值）。

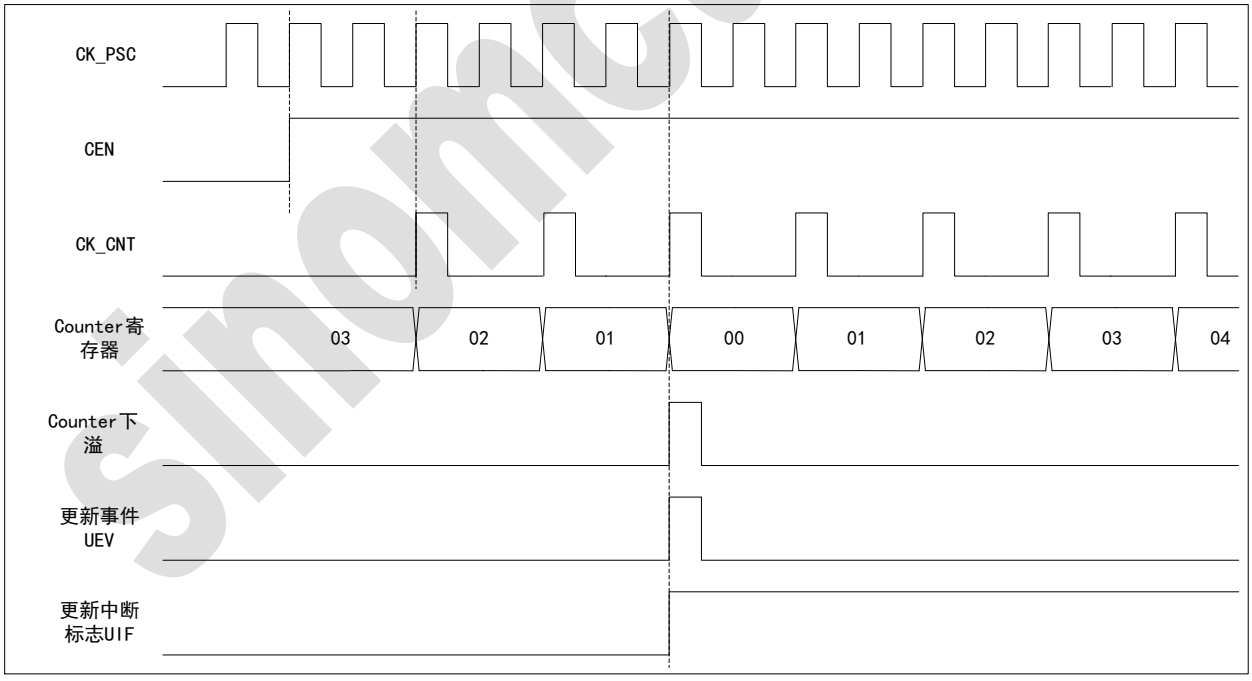
举例如下：计数器在不同时钟频率下计数器行为。

计数器时序图：预分频因子=1，TIMx_ARR=0x06

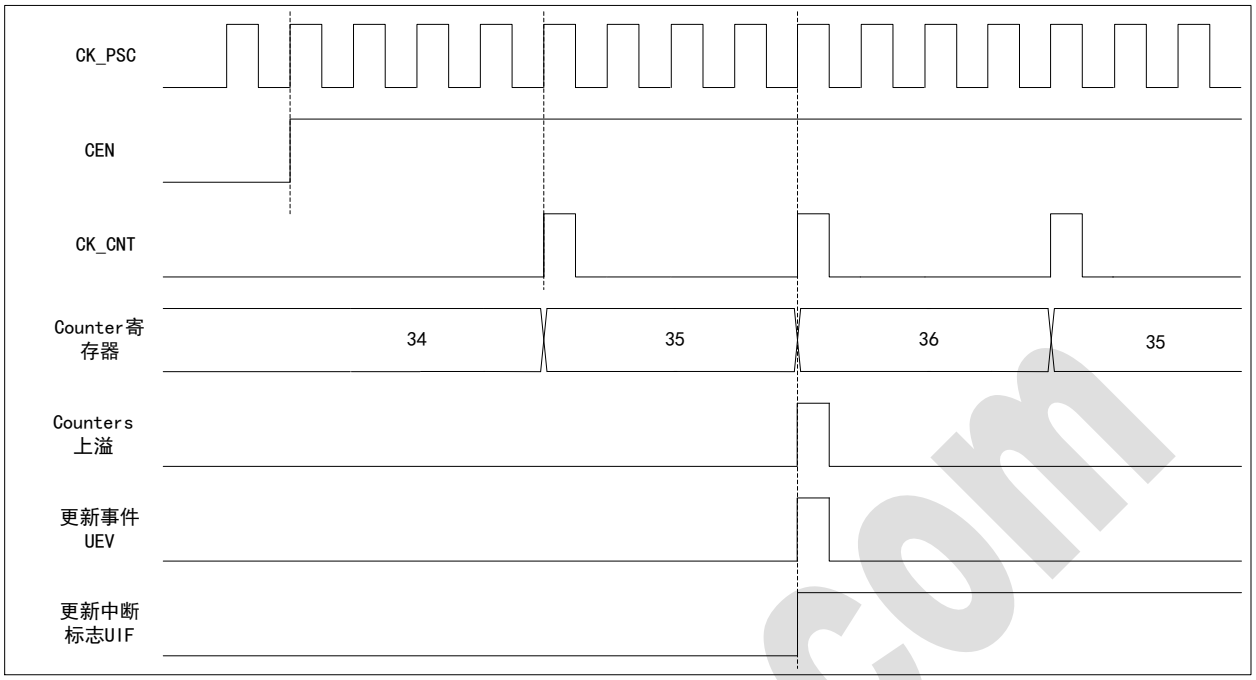


注：上图为中心对齐模式 1

计数器时序图：预分频因子=2，TIMx_ARR=0x36

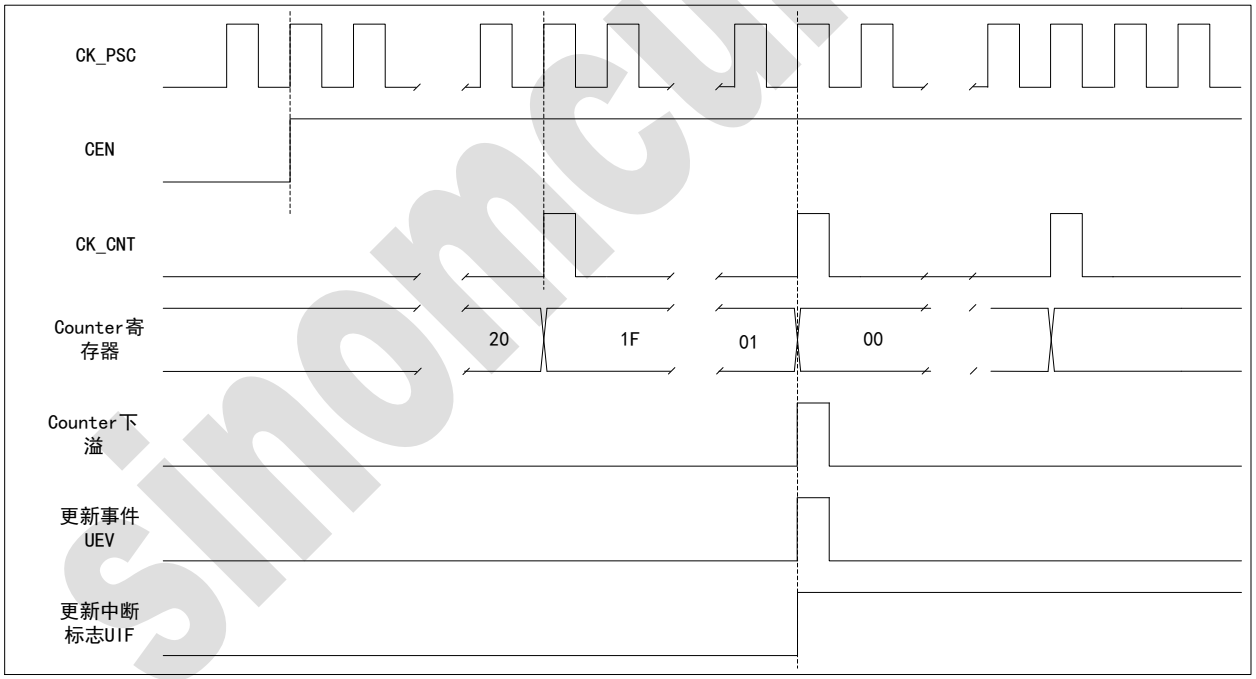


计数器时序图：预分频因子=4，TIMx_ARR=0x36

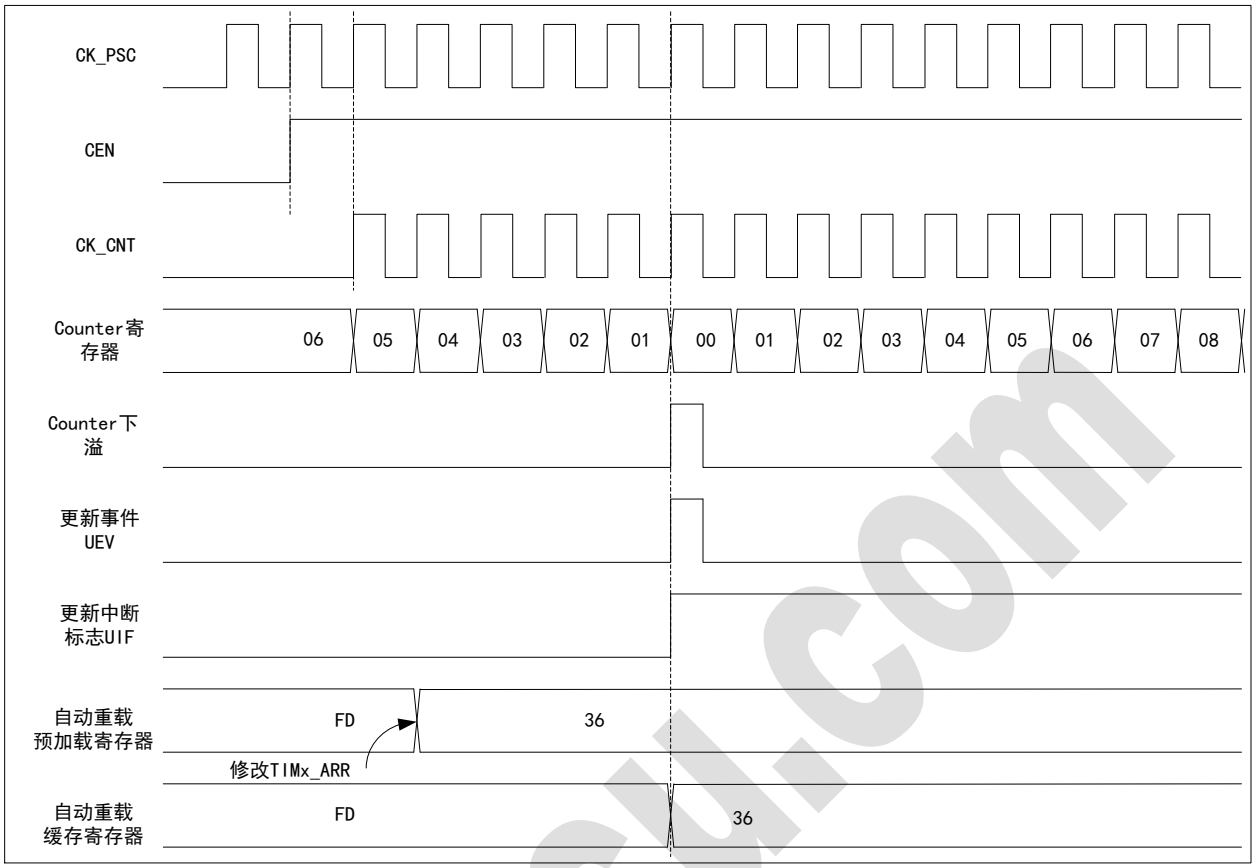


注：上图为中心对齐模式 2 或模式 3

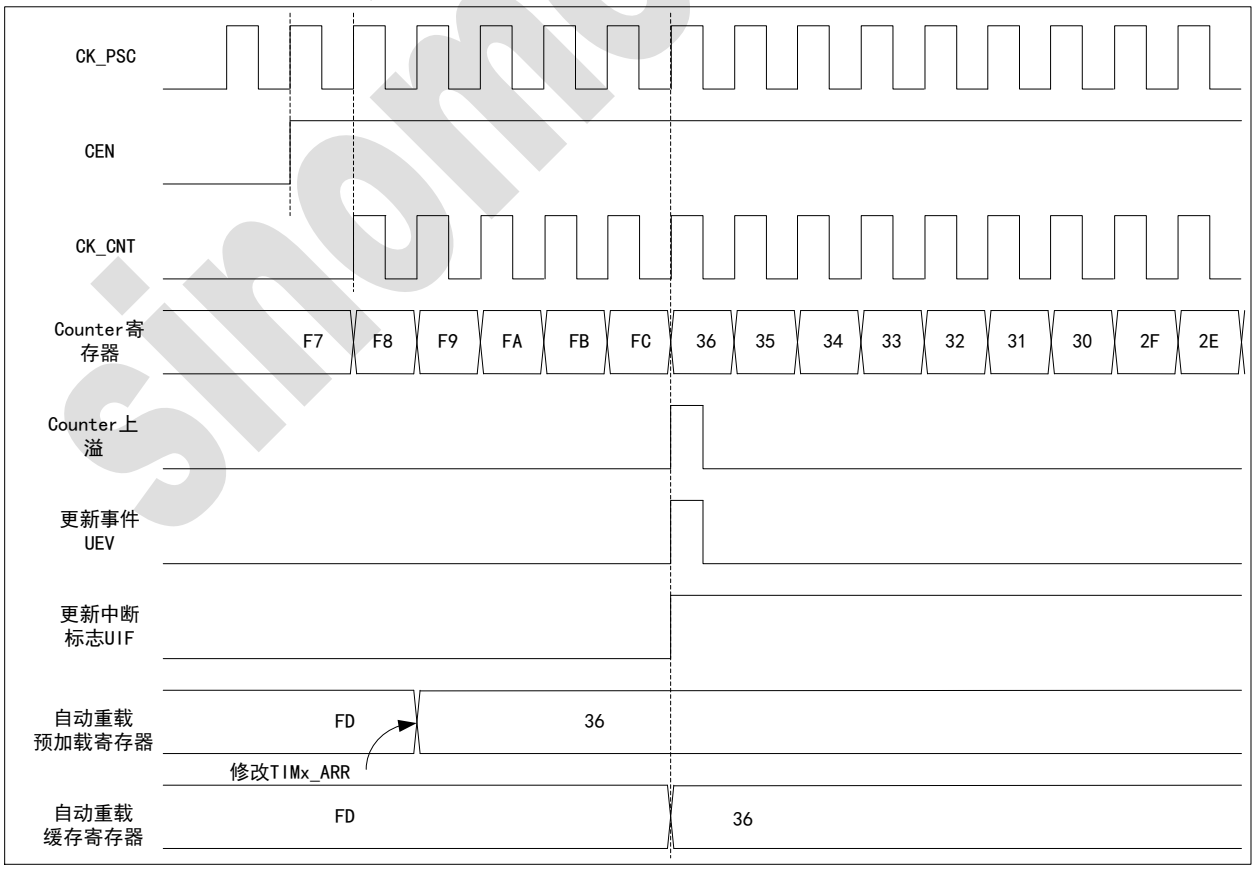
计数器时序图：预分频因子=N



计数器时序图：预分频因子=1，ARPE=1 的更新事件(计数器下溢)



计数器时序图：预分频因子=1，ARPE=1 的更新事件(计数器上溢)



14.3.3 时钟源

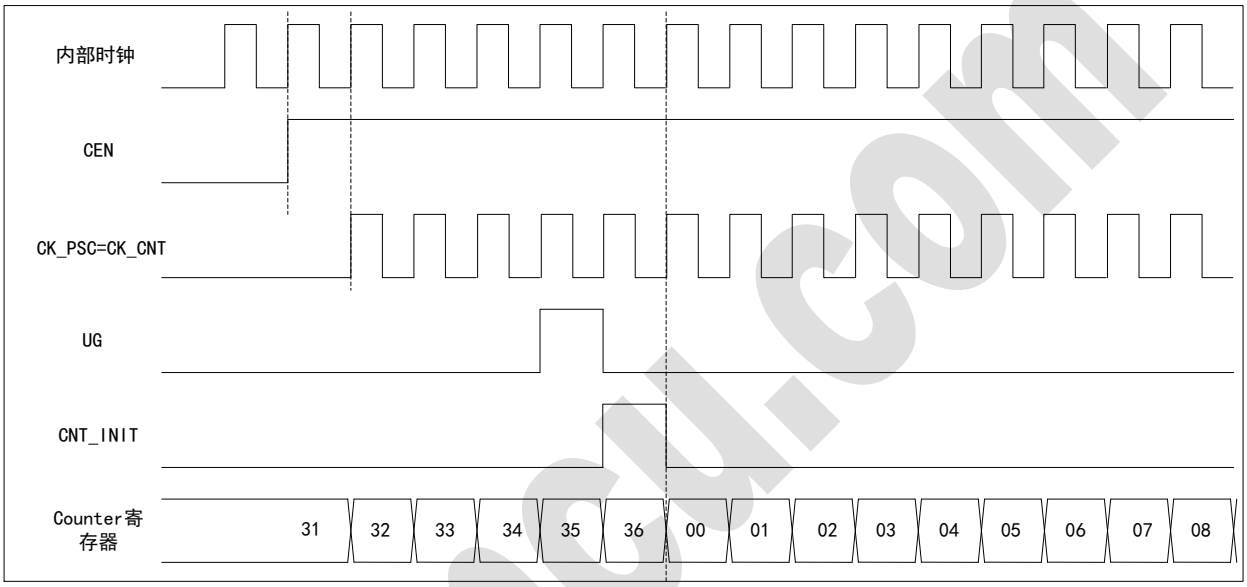
计数器时钟源包括：

- ✧ 内部时钟 CK_INT
- ✧ 外部时钟模式 1：外部输入引脚（TIx）
- ✧ 外部时钟模式 2：外部触发输入 ETR
- ✧ 内部触发输入 ITRx：使用一个定时器作为另一个定时器的预分频器。举例：可以配置 Timer1 作为 Timer2 的预分频器。详情参考<定时器同步>章节。

内部时钟源（CK_INT）

如果从模式控制器禁用（SMS=000），则 CEN、DIR（TIMx_CR1 寄存器）和 UG 位（TIMx_ EGR 寄存器）为实际控制位，并且只能被软件修改（除了 UG 位保持自动被清除）。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。

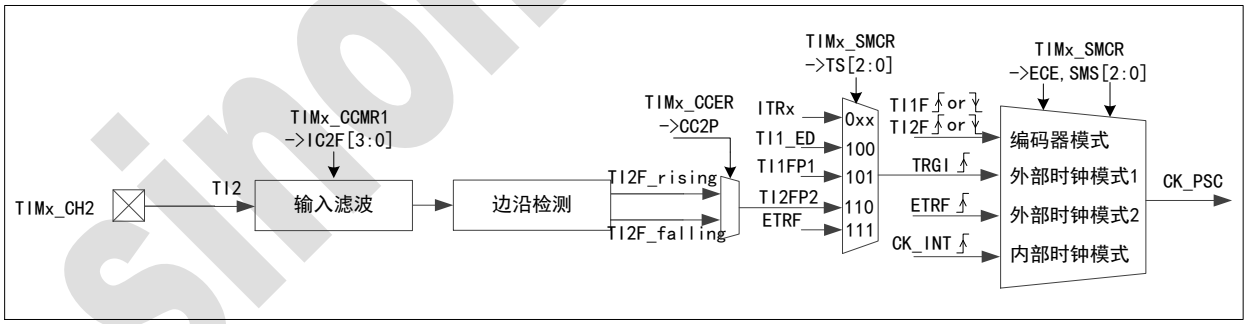
计数器时序图：内部时钟，预分频因子=1，正常模式



外部时钟源模式 1

设置 TIMx_SMCR 寄存器的 SMS=111，此模式被选中。计数器对选定输入端的每个上升沿或下降沿计数。

TI2 外部时钟源联接示例框图



举例：要配置向上计数器在 TI2 输入端的上升沿计数，步骤如下：

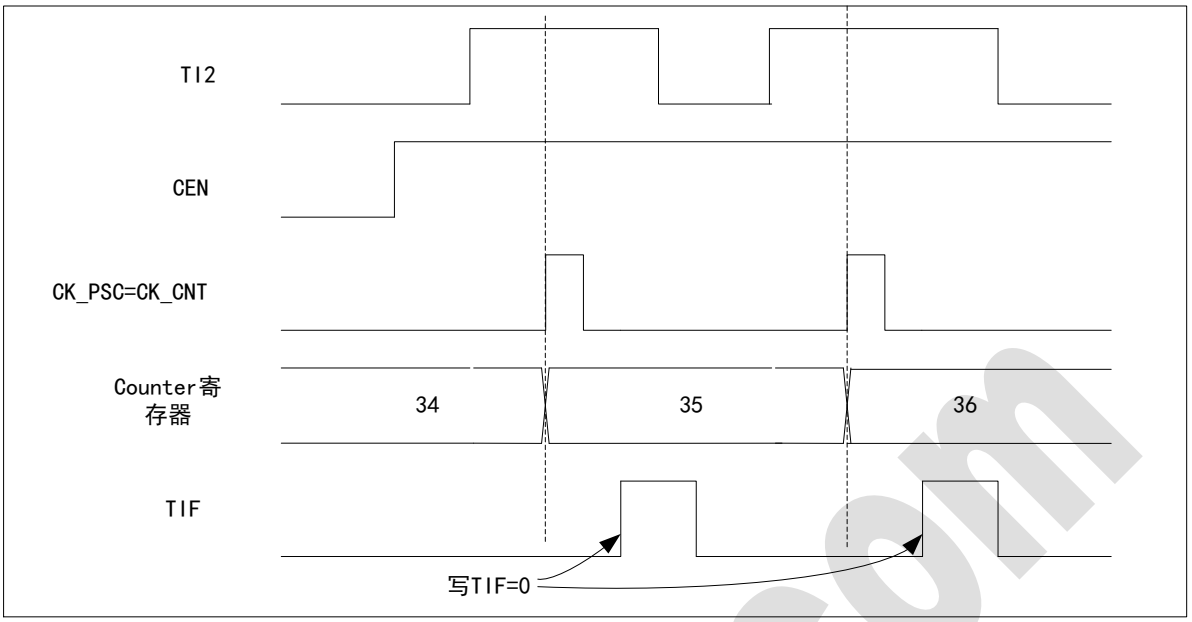
1、 写 TIMx_CCMR1 寄存器的 CC2S=01，配置通道 2 检测 TI2 输入的上升沿；
2、 写 TIMx_CCMR1 寄存器的 IC2F[3:0]，选择输入滤波器时间（如果不需要滤波器，保持 IC2F[3:0]=0000）；
3、 写 TIMx_CCER 寄存器的 CC2P=0，选定上升沿极性；
4、 写 TIMx_SMCR 寄存器的 SMS[2:0]=111，选择定时器外部时钟模式 1；
5、 写 TIMx_SMCR 寄存器中的 TS[2:0]=110，选定 TI2 作为触发输入源；
6、 写 TIMx_CR1 寄存器的 CEN=1，启动计数器。

注： 捕获预分频器不用作触发，所以不需要对它进行配置。

当发生一次 TI2 上升沿，计数器计数一次，并置位 TIF 位。

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。

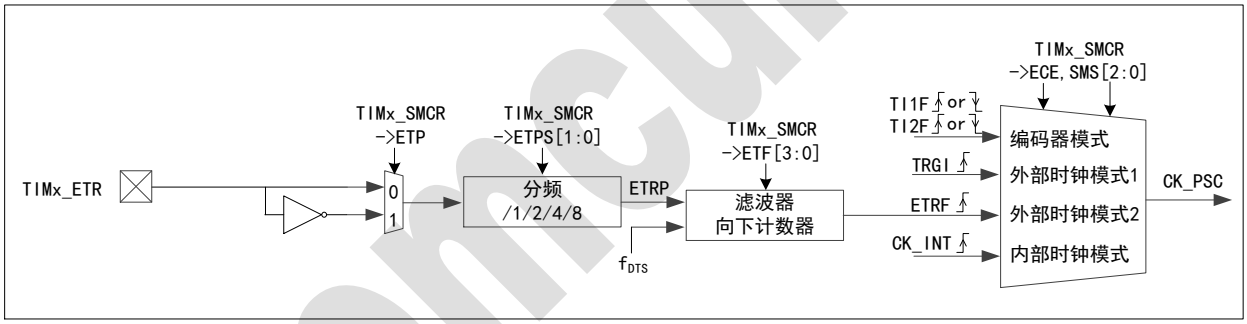
控制时序：外部时钟模式 1



外部时钟模式 2

设置 TIMx_SMCR 寄存器的 ECE=1，此模式被选中。计数器对外部触发 ETR 的每个上升沿或下降沿计数。

ETR 外部触发源联接示例框图



举例：要配置向上计数器在 ETR 的每 2 个上升沿计数一次，步骤如下：

1、 本例中不需要滤波，写 TIMx_SMCR 寄存器中的 ETF[3:0]=0000；

2、 写 TIMx_SMCR 寄存器中的 ETPS[1:0]=01，设置预分频器因子=2；

3、 写 TIMx_SMCR 寄存器中的 ETP=0，选择检测 ETR 的上升沿；

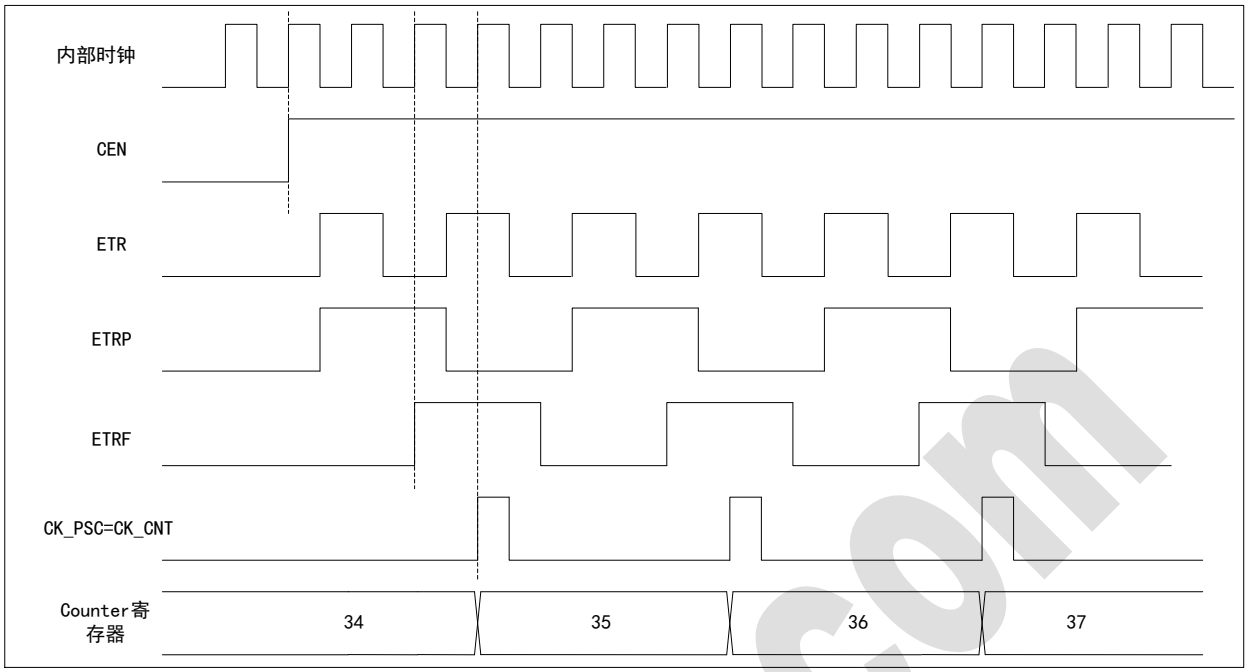
4、 写 TIMx_SMCR 寄存器中的 ECE=1，开启外部时钟模式 2；

5、 写 TIMx_CR1 寄存器中的 CEN=1，启动计数器。

计数器在每 2 个 ETR 上升沿计数一次。

ETR 的上升沿和计数器实际时钟之间的延时，取决于在 ETRP 信号端的重新同步电路。

控制时序：外部时钟模式 2



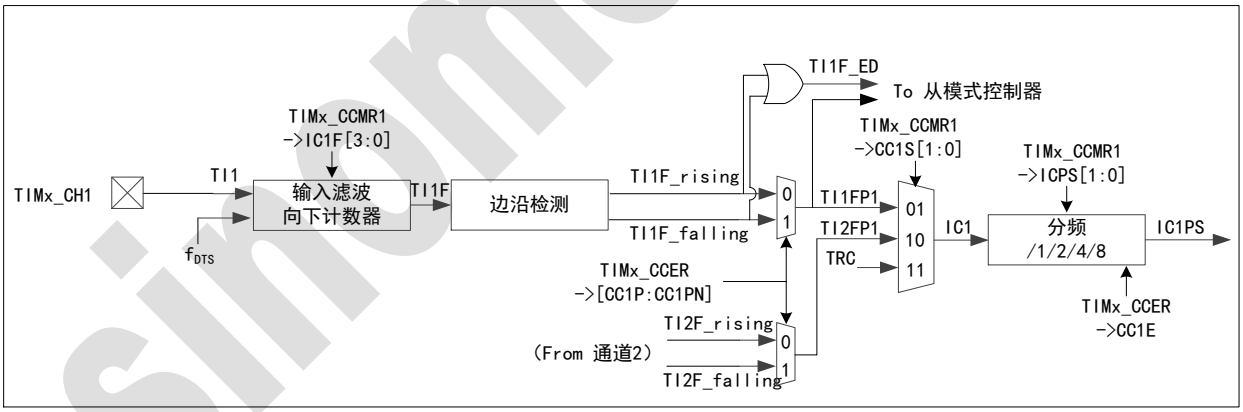
14.3.4 捕获/比较通道

每一个捕获/比较通道都是包括：一个捕获/比较寄存器（包含缓存寄存器），一个用于捕获的输入级（带数字滤波、多路复用选择和预分频器），和一个输出级（带比较器和输出控制）。

输入级

对 TIx 的输入信号进行采样，产生一个滤波后 TIxF 信号，经过带有极性选择的边沿检测器生成 TIxFPx 信号，TIxFPx 可以作为从模式控制器的触发输入或者作为捕获命令。该信号预分频后（ICxPS）进入捕获寄存器。

捕获/比较通道输入级（举例：通道 1）



输出级

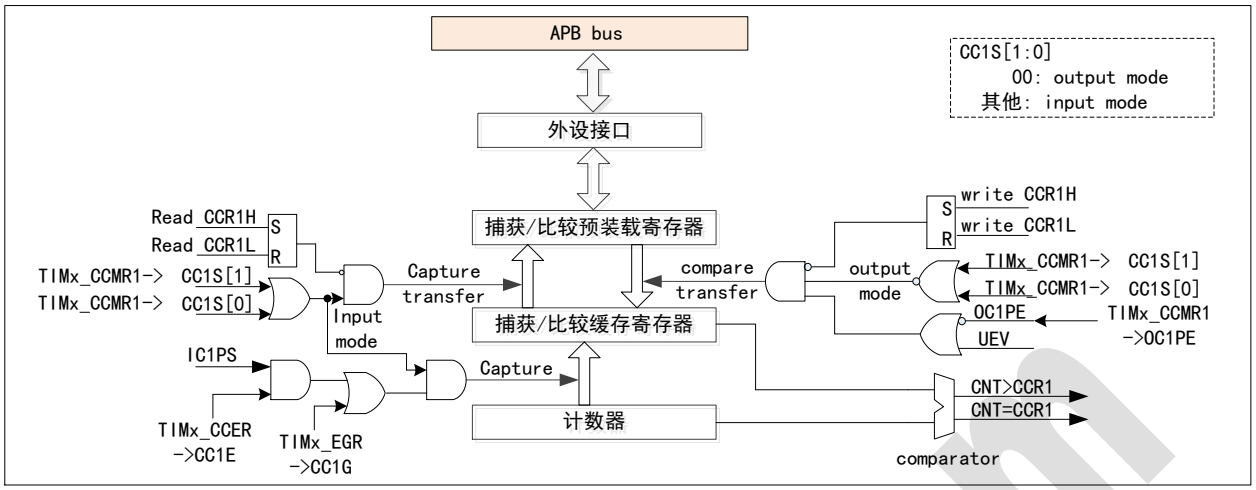
输出部分产生一个中间波形 OCxRef（高有效）作为基准，链的末端决定最终输出信号的极性。

捕获/比较模块由一个预装载寄存器（preload register）和一个缓存寄存器（shadow register）组成。读写过程仅操作预装载寄存器。

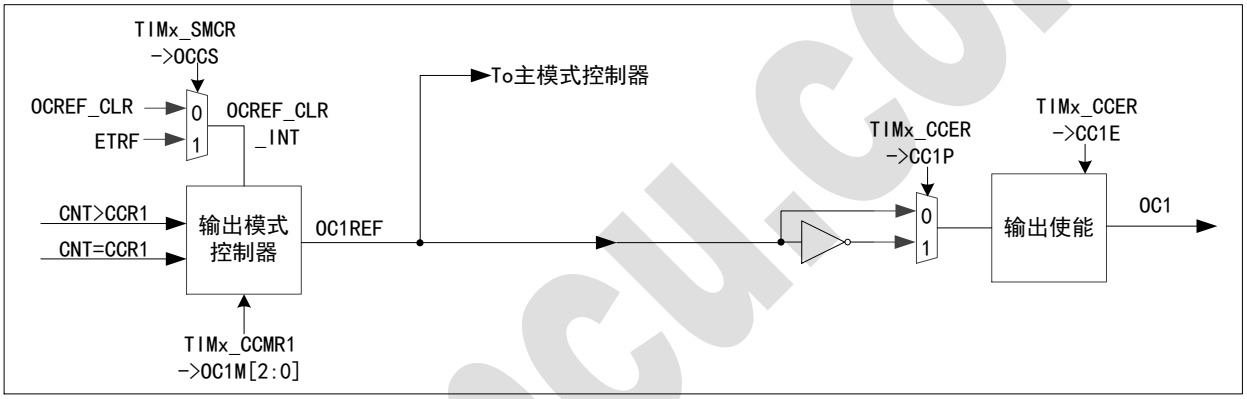
在捕获模式下，捕获发生在缓存寄存器上，然后再复制到预装载寄存器中。

在比较模式下，预装载寄存器的内容被复制到缓存寄存器中，然后缓存寄存器的内容和计数器进行比较。

捕获/比较通道主电路



捕获/比较通道输出级（通道 1，通道 2/3/4 同通道 1）



14.3.5 输入捕获模式

在输入捕获模式下，当检测 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（TIMx_CCRx）中。当发生捕获事件时，相应 CCxIF 标志（TIMx_SR 寄存器）被置 1， 如果使能了中断或者 DMA，则产生一个中断或者一个 DMA 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（TIMx_SR 寄存器）被置 1。软件写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

举例，在 TI1 输入的上升沿时，捕获计数器的值到 TIMx_CCR1 寄存器中，操作步骤如下：

1、 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01（选中 TI1），只要 CC1S 不为‘00’，通道被配置为输入并且 TIMx_CCR1 寄存器变为只读。
2、 根据连接到计数器的输入信号，配置输入滤波器时间（输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位）。举例，当 TI1 翻转时，信号抖动最多 5 个内部时钟， 则须配置滤波器时间大于 5 个时钟周期，TIMx_CCMR1 寄存器中写入 IC1F=0011 配置采样次数 8（以 f_{DTS} 频率采样），通过连续 8 次采样以确认电平变换。
3、 选择 TI1 通道有效沿。写 TIMx_CCER 寄存器中 CC1P=0 和 CC1NP=0（本例中为上升沿）。
4、 配置输入预分频器。在本例中，设置为每个有效的电平转换时发生一次捕获，因此预分频器被禁止（写 TIMx_CCMR1 寄存器的 IC1PS=00）。
5、 写 TIMx_CCER 寄存器的 CC1E=1，允许计数器的值被捕获至捕获寄存器中。
6、 根据需要，置位 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，置位 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- ✧ 有效的电平转换发生时，计数器的值被传送至 TIMx_CCR1 寄存器；
- ✧ CC1IF 标志被置位（中断标志）。当发生至少 2 个连续的捕获时，且 CC1IF 未被清除，CC1OF 也被置位；
- ✧ 若置位 CC1IE 位，则会产生一个中断。
- ✧ 若置位 CC1DE 位，则会产生一个 DMA 请求。

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，这是为了避免在读取标志之后和读取数据之前可能发生的捕获溢出。

注： 通过软件设置 TIMx_EGR 寄存器中相应的 CCxG 位，也可以产生输入捕获中断 和/或 DMA 请求。

14.3.6 PWM 输入模式

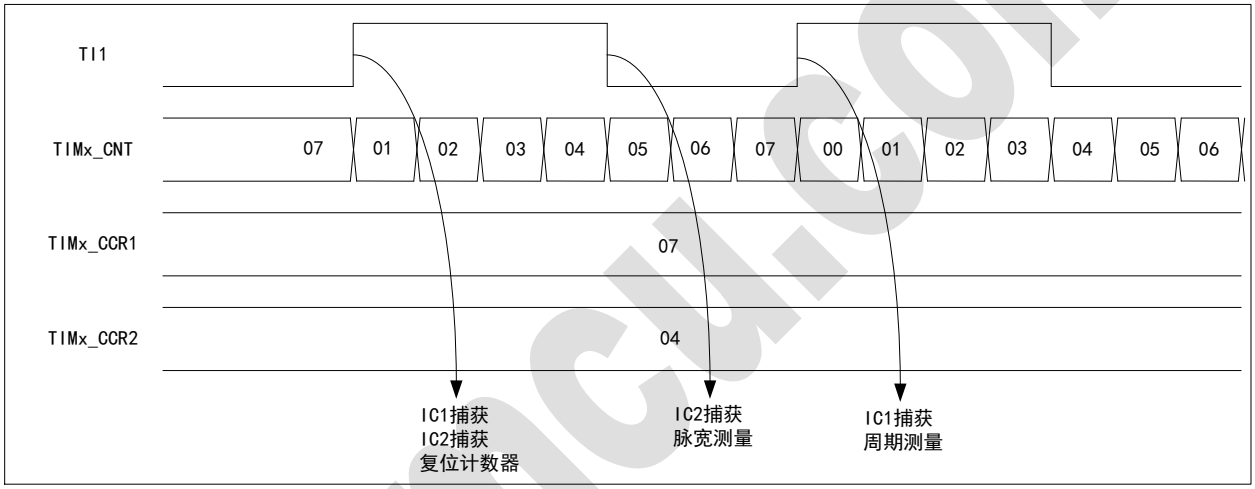
该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

- ✧ 一个 TIx 输入映射至两个 ICx 信号
- ✧ 两个 ICx 信号为边沿有效，极性相反
- ✧ 两个 TIxFP 信号之一被作为触发输入信号，而从模式控制器被配置成复位模式

举例，测量输入到 TI1 上的 PWM 信号的周期（TIMx_CCR1 寄存器）和占空比（TIMx_ CCR2 寄存器），操作步骤如下（取决于 CK_INT 的频率和预分频器的值）：

1、 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01（选中 TI1）。
2、 选择 TI1FP1 的有效极性（用于捕获数据到 TIMx_CCR1 中和清除计数器）：置 CC1P=0（上升沿有效）。
3、 选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
4、 选择 TI1FP2 的有效极性（用于捕获数据到 TIMx_CCR2）：置 CC2P=1（下降沿有效）。
5、 选择有效的触发输入信号：设置 TIMx_SMCR 寄存器中的 TS=101（选择 TI1FP1）。
6、 配置从模式控制器为复位模式：置 TIMx_SMCR 中的 SMS=100。
7、 使能捕获功能：设置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

PWM 输入模式时序



14.3.7 强制输出模式

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

设置 TIMx_CCMRx 寄存器中相应的 OCxM=101，输出比较信号（OCxREF/OCx）强制为有效状态。这样 OCxREF 被强制高电平(OCxREF 始终为高电平有效)，置位 TIMx_CCER 的 CCxP 位，OCx 可得到极性相反的信号。

举例，CCxP=0（OCx 高电平有效），则 OCx 被强制为高电平。

设置 TIMx_CCMRx 寄存器中相应的 OCxM=100，输出比较信号（OCxREF/OCx）强制为低电平。

该模式下，在 TIMx_CCRx 缓存寄存器和计数器之间的比较仍然在执行，相应的标志也会被置位。并会产生相应的中断和 DMA 请求。下面的输出比较模式一节中将详细介绍。

14.3.8 输出比较模式

此模式用于控制输出波形或指示一段时间到时。

当计数器与捕获/比较寄存器的内容匹配（相等）时，输出比较功能做如下操作：

- ✧ 根据输出比较模式（TIMx_CCMRx 寄存器中 OCxM 位）和输出极性（TIMx_CCER 寄存器中的 CCxP 位）的配置，输出至对应引脚。在比较匹配发生时，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）。
- ✧ 置位中断状态寄存器中的标志位（TIMx_SR 寄存器中的 CCxIF 位）。
- ✧ 若置位相应的中断屏蔽位（TIMx_DIER 寄存器中的 CCxIE 位），则产生一个中断。
- ✧ 若置位相应的 DMA 使能位 (TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位)，则产生一个 DMA 请求。

设置 TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。计时分辨率为计数器的一次计数。

输出比较模式也能用来输出一个单脉冲（单脉冲模式）。

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、 根据需求，写入数据至 TIMx_ARR 和 TIMx_CCRx 寄存器中。

3、 如果要产生一个中断请求，设置 CCxIE 位。

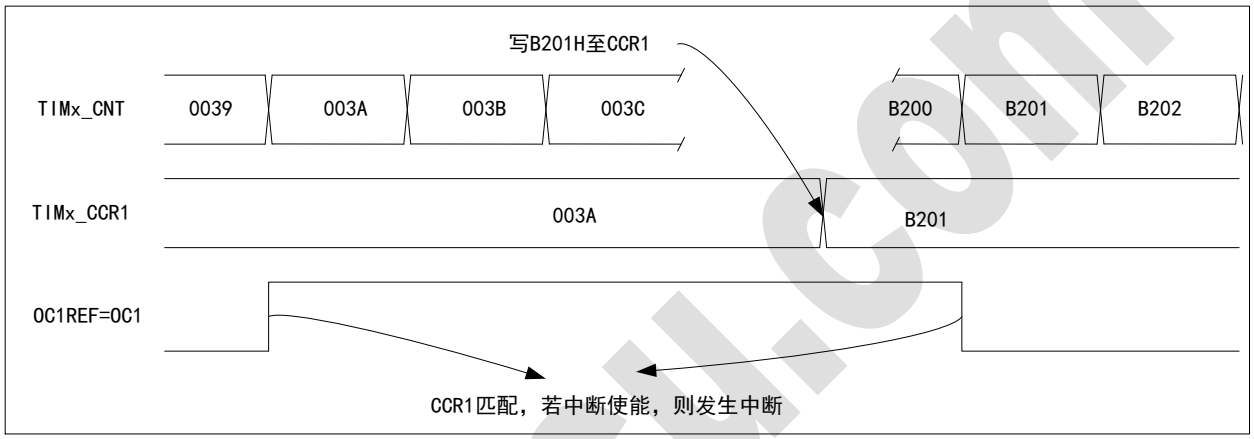
4、 选择输出模式，举例如下：

- 写 OCxM=011，计数器与 CCRx 匹配时，翻转 OCx 的输出引脚；
- 写 OCxPE=0，禁用预装载寄存器；
- 写 CCxP=0，选择极性为高电平有效；
- 写 CCxE=1，使能输出。

5、 置位 TIMx_CR1 寄存器的 CEN 位，启动计数器。

TIMx_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，前提是未启用预装载寄存器（OCxPE=‘0’，否则 TIMx_CCRx 的缓存寄存器只在发生下次更新事件 EUV 时更新）。举例如下图。

输出比较模式，配置为：翻转 OC1



14.3.9 PWM 模式

脉冲宽度调制模式（PWM）可以产生一个 PWM 信号，由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比。

在 TIMx_CCMRx 寄存器中的 OCxM 位写入‘110’（PWM 模式 1）或‘111’（PWM 模式 2），可以在每个通道独立地设置 PWM 模式（每个 OCx 输出一种 PWM）。必须置位 TIMx_CCMRx 寄存器的 OCxPE 位，使能相应的预装载寄存器（preload register），最后置位 TIMx_CR1 寄存器的 ARPE 位，使能自动重载预装载寄存器（在向上计数或中心对称模式中）。

只有发生一个更新事件时，预装载寄存器才能被传送到缓存寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。

设置 TIMx_CCER 寄存器中的 CCxP 位，配置 OCx 的极性为高电平有效或低电平有效。OCx 的输出使能通过（TIMx_CCER 寄存器中）CCxE 位控制。详细参考 TIMx_CCER 寄存器的描述。

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

然而，为了与 OCREF_CLR 的功能一致（在下一个 PWM 周期之前，ETR 信号上的一个外部事件能够清除 OCxREF），OCxREF 信号只能在下述条件下产生：

✧ 当比较的结果改变，或

✧ 当输出比较模式（TIMx_CCMRx 寄存器中的 OCxM 位）从“冻结”（无比较，OCxM=‘000’）切换到某个 PWM 模式（OCxM=‘110’或‘111’）。

设置 TIMx_CR1 寄存器中 CMS 位，选择 PWM 信号为边沿对齐或中央对齐。

PWM 边沿对齐模式

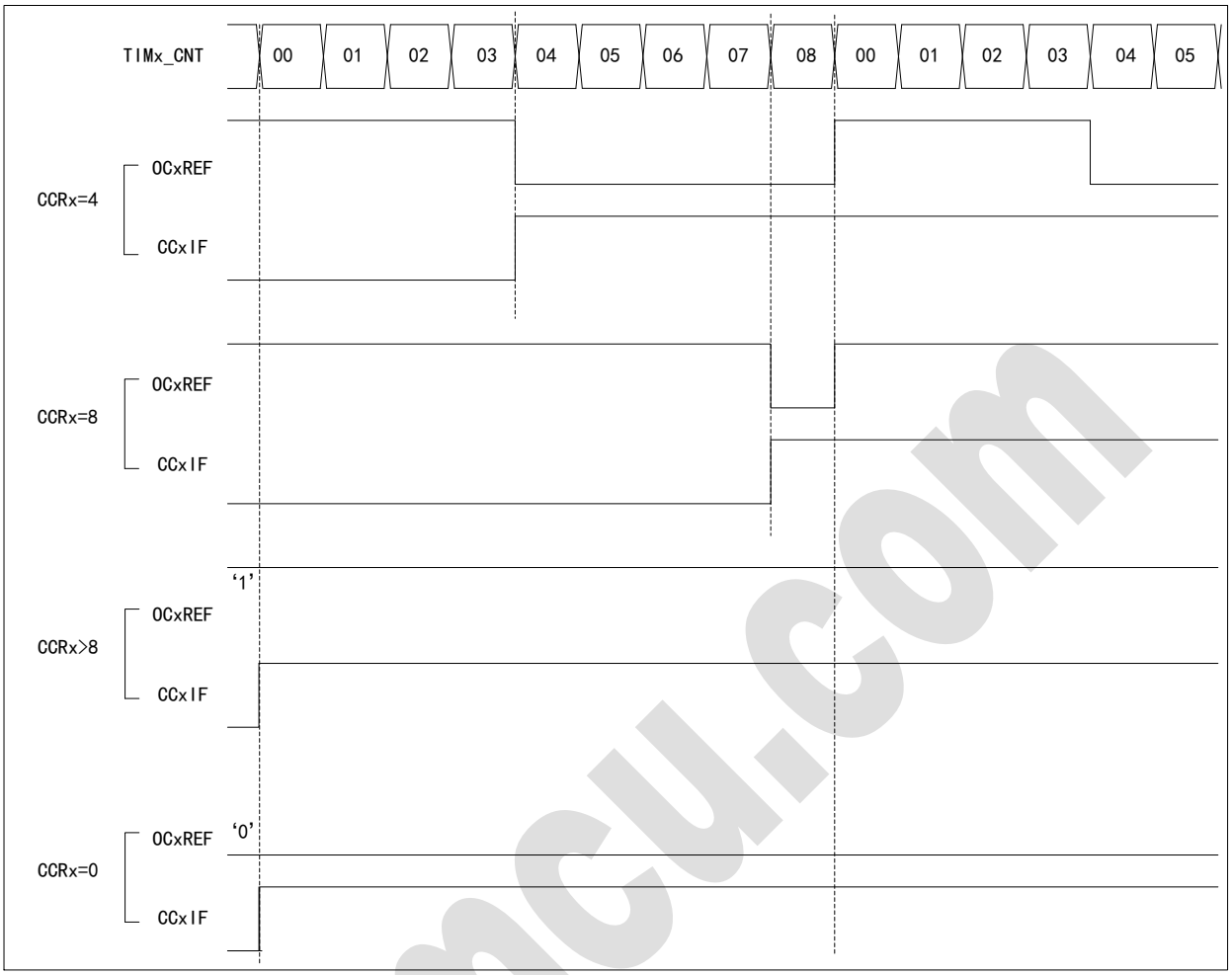
向上计数配置

设置 TIMx_CR1 寄存器中的 CMS=00 且 DIR=0，执行向上计数。参见 13.3.2 向上计数配置章节。

举例，PWM 模式 1

当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重载值（TIMx_ARR），则 OCxREF 保持为‘1’。如果比较值为 0，则 OCxREF 保持为‘0’。 下图为 TIMx_ARR=8 时边沿对齐的 PWM 波形实例。

PWM 波形（边沿对齐，向上计数，ARR=8）



向下计数配置

设置 TIMx_CR1 寄存器中的 CMS=00 且 DIR=1，执行向下计数。参见 13.3.2 向下计数配置章节。

PWM 模式 1 下，当 TIMx_CNT>TIMx_CCRx 时，PWM 参考信号 OCxREF 为低，否则为高。如果 TIMx_CCRx 中的比较值大于自动重载值（TIMx_ARR），则 OCxREF 保持为‘1’。该模式下不能产生 0%的 PWM 波形。

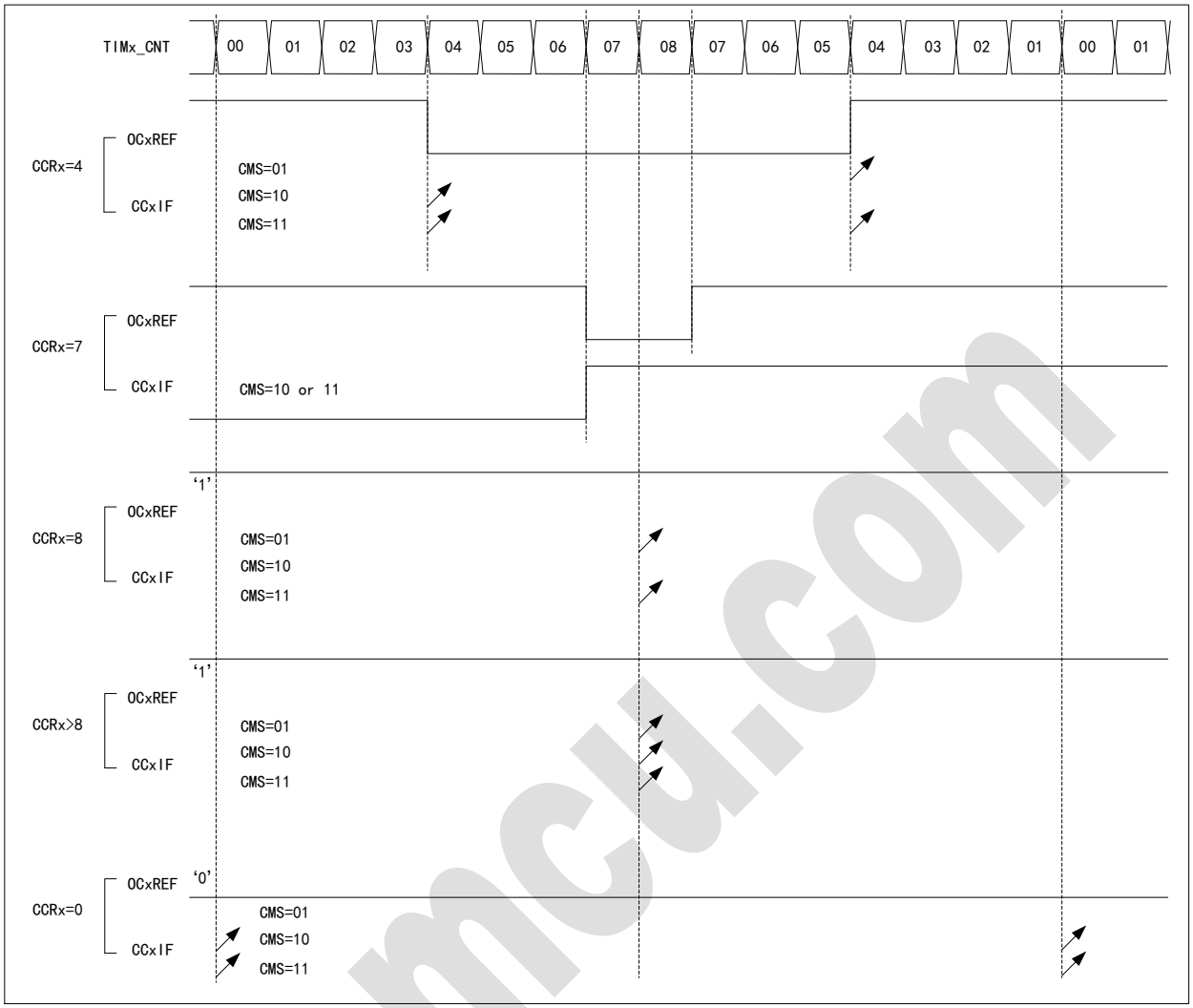
PWM 中心对齐模式

设置 TIMx_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 信号与边沿模式相同）。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时、向下计数时、或在向上和向下计数时被置 1。中心对齐模式下，TIMx_CR1 寄存器中的计数方向位（DIR）由硬件更新，只读，软件不可修改。参见 13.3.2 中心对齐模式章节。

举例，PWM 中心对齐模式

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

PWM 中心对齐波形（APR=8）



注意事项：

1、 启动中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 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。

2、 不推荐当运行在中心对齐模式时改写计数器，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载的值（TIMx_CNT>TIMx_ARR），则方向不会被更新。 例如，如果计数器正在向上计数，它就会继续向上计数。
- 如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。

3、 使用中心对齐模式最可靠的方式，就是在启动计数器之前产生一个软件更新（设置 TIMx_ 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。

14.3.10 外部事件时清零 OCxREF 信号

设置 TIMx_CCMRx 寄存器中对应的 OCxCE 位为‘1’，指定通道的外部事件触发 OCxREF 信号清零功能开启，即当 OCxREF_CLR_INPUT 发生高电平触发 OCxREF 信号拉低，OCxREF 信号将保持为低直到发生下一次的更新事件 UEV。

该功能只能用于输出比较和 PWM 模式，而不能用于强制模式。

设置 TIMx_SMCR 寄存器中 OCCS 位，可以选 OCREF_CLR 输入（来自模拟比较器）和 ETRF(滤波器后的 ETR)之一作为 OCREF_CLR_INPUT 信号。

举例，OCxREF 信号可以关联到一个比较器的输出，用于控制电流（cycle by cycle）。若 ETRF 被选择，ETR 必须配置如下：

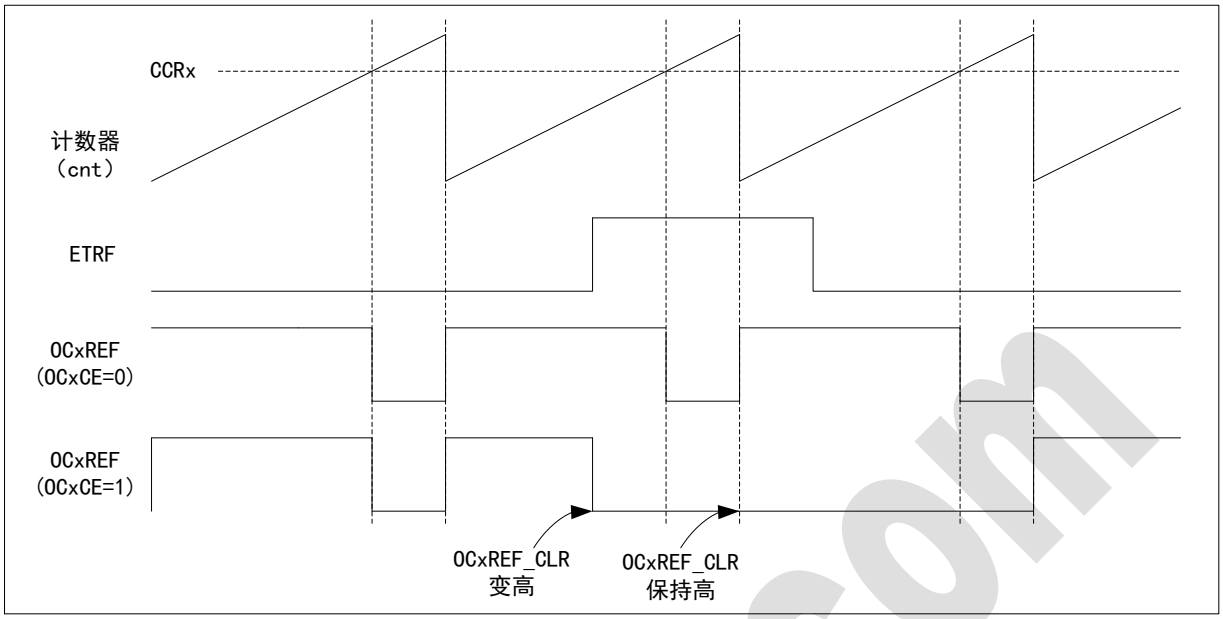
1、 ETR 预分频器必须处于关闭：TIMx_SMCR 寄存器中的 ETPS[1:0]=00。

2、 必须禁止外部时钟模式 2：TIMx_SMCR 寄存器中的 ECE=0。

3、 ETR 的极性（ETP）和滤波器（ETF）可以根据需要配置。

下图显示了当 ETRF/OCxREF_CLR 输入变为高时，对应不同 OCxCE 的值，OCxREF 信号的行为。在此例中，定时器 TIMx 被置于 PWM 模式。

清零 OCxREF 信号时序



注：在 PWM 占空比为 100%的情况时（CCRx>ARR），OCxREF 在下一次计数溢出时被再次使能。

14.3.11 单脉冲模式

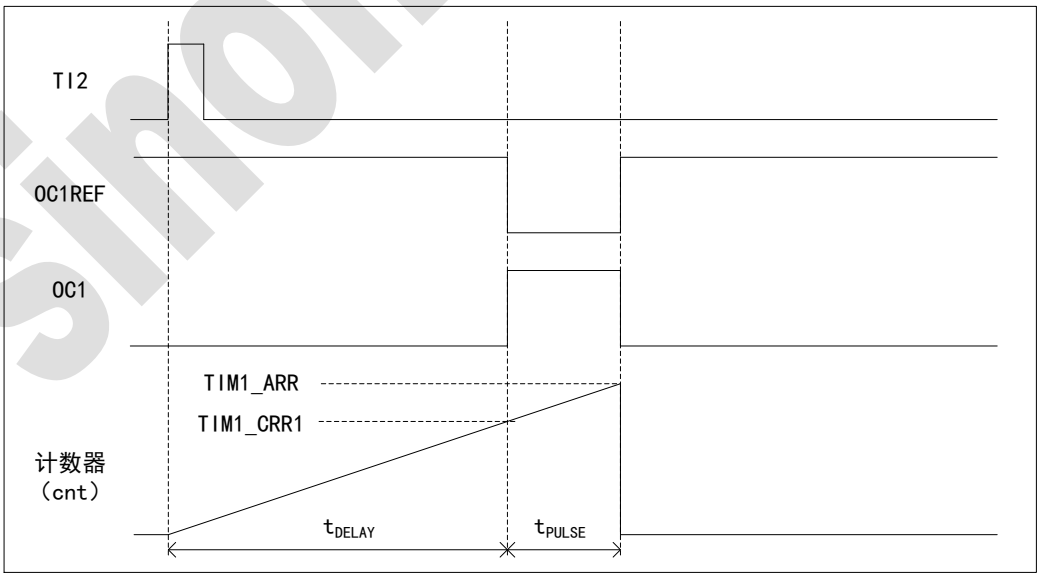
单脉冲模式（OPM）是上述模式的一个特例。这种模式下，计数器响应一个激励后启动，经过一个可编程的延时之后，产生一个脉宽编程的单脉冲。

计数器启动由从模式控制器控制，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_ CR1 寄存器中的 OPM 位将选择单脉冲模式，计数器在产生下一个更新事件 UEV 时自动停止。

想要正确产生一单脉冲，比较值与计数器的初始值必须不同。启动之前（ 此时定时器正在等待触发)， 配置必须满足：

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

单脉冲模式时序举例



举例，要求在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲，从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后。假定 TI2FP2 作为触发 1:

- ✧ 配置 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。
- ✧ 配置 TIMx_CCER 寄存器中的 CC2P=0 和 CC2NP=0，使 TI2FP2 检测上升沿。
- ✧ 配置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发（TRGI）。

✧ 配置 TIMx_SMCR 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 (TIMx_ARR - TIMx_CCR1)。

✧ 若要求波形，比较值匹配时从 0 到 1 翻转，当计数器达到预装载值时从 1 到 0 翻转；

- 首先要设置 TIMx_CCMR1 寄存器的 OC1M=111，选择 PWM 模式 2；
- 有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；
- 在 TIMx_CCR1 寄存器中写入比较值，在 TIMx_ARR 寄存器中写入自动重载值，设置 UG 位来产生一个更新事件，
- 最后等待 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，TIMx_CR1 寄存器中的 DIR 和 CMS 位应该配置为低。

设置 TIMx_CR 寄存器中的 OPM=1，在下一个更新事件（UEV，当计数器从自动装载值翻转到 0）时停止计数。当 TIMx_CR1 寄存器中的 OPM=0 时，进入重复模式。

特殊情况：OCx 快速使能：

在单脉冲模式下，TIx 输入脚的边沿检测逻辑触发 CEN 位来启动计数器。通过计数器和比较值间的比较结果驱动输出翻转。但是，这些操作需要一定的时钟周期，这样就限制了可获得的最小延时 t_{DELAY}。

如果需要以最小延时输出波形，可以置位 TIMx_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF（和 OCx）直接响应激励而不再依赖比较，翻转后电平与比较匹配时的电平一致。OCxFE 只在通道配置为 PWM 模式 1 和 PWM 模式 2 时起作用。

14.3.12 编码器接口模式

选择编码器接口模式：如果计数器只在 TI2 的边沿计数，则设置 TIMx_SMCR 寄存器中的 SMS=001；如果只在 TI1 边沿计数，则置 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。

通过设置 TIMx_CCER 寄存器中的 CC1P 和 CC2P 位，选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。CC1NP 和 CC2NP 位必须保持为低。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口，参见下表 。计数器启动 (TIMx_CR1 寄存器中的 CEN=1)，则计数器由 TI1FP1 或 TI2FP2 上的每次有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在经过输入滤波器和极性控制后的信号；如果没有滤波和极性反相，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。进而计数器向上或向下计数，同时硬件对 TIMx_CR1 寄存器的 DIR 位修改。不管计数器是对 TI1 或 TI2 单独计数还是对 TI1 和 TI2 同时计数，在任一输入端（TI1 或 TI2）的跳变都会重新计算 DIR 位。

编码器接口模式相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 TIMx_ARR 寄存器的自动装载值之间连续计数（是 0 到 ARR 还是 ARR 到 0 计数，取决于计数方向）。所以在开始计数之前必须配置 TIMx_ARR；同样，捕获、比较、预分频器、重复次数计数器、触发输出这些特性仍工作如常。

编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。

在这个模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。

下表列出了所有可能的组合， 假设 TI1 和 TI2 不同时变换。

计数方向与编码器信号的关系

有效沿	相对信号电平 (TI1FP1 对应 TI2, TI2FP2 对应 TI1)	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
在 TI1 和 TI2 上计数	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大提高了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

下图是一个计数器操作的示例，显示了计数信号的产生和方向控制。它还显示了当选择了双边沿时，输入抖动是如何被抑制的；抖动可能会在传感器的位置靠近一个转换点时产生。在这个例子中，我们假定配置如下：

✧ CC1S=‘01’（TIMx_CCMR1 寄存器，TI1FP1 映射到 TI1）

✧ CC2S=‘01’（TIMx_CCMR2 寄存器，TI2FP2 映射到 TI2）

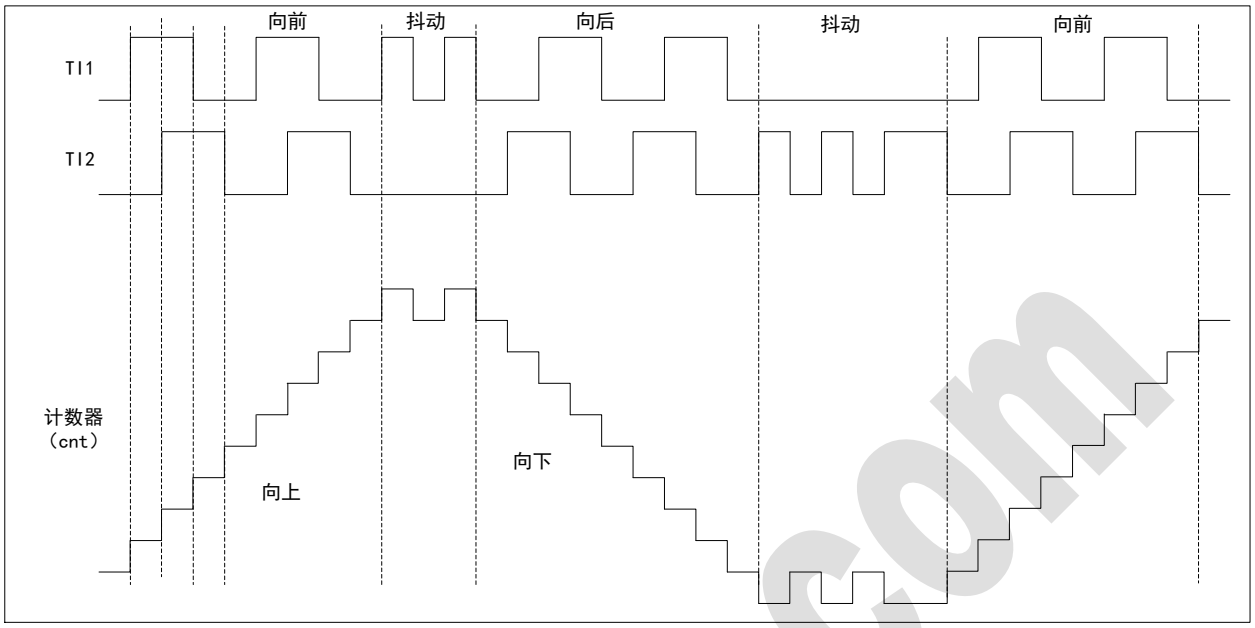
✧ CC1P=‘0’（TIMx_CCER 寄存器，TI1FP1 不反相，TI1FP1=TI1）

✧ CC2P=‘0’（TIMx_CCER 寄存器，TI2FP2 不反相，TI2FP2 =TI2）

✧ SMS=‘011’（TIMx_SMCR 寄存器，所有的输入均在上升沿和下降沿都有效）

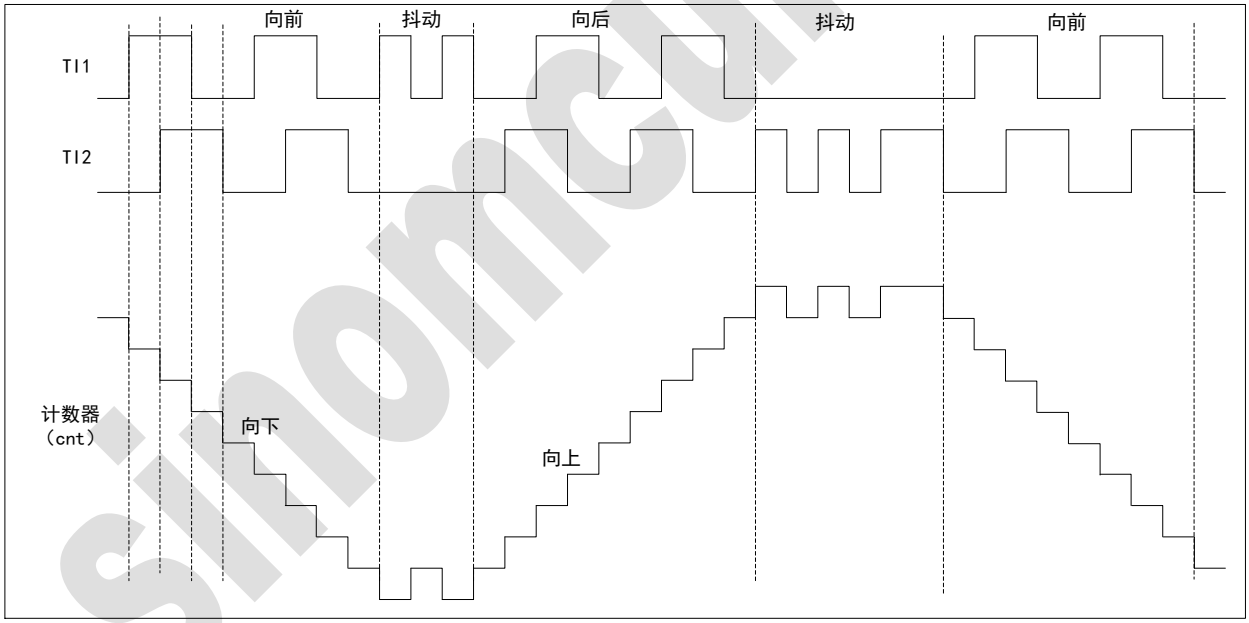
✧ CEN=‘1’（TIMx_CR1 寄存器，计数器使能）

编码器模式下的计数器操作实例



下图为当 IC1FP1 极性反相时计数器的操作实例（CC1P=‘1’，其他配置与上例相同）。

TI1FP1 反相的编码器接口模式实例



当定时器配置成编码器接口模式时，用于指示传感器当前位置。配合使用第二个定时器配置为捕获模式，通过测量两个编码器事件的间隔，从而获得动态信息（ 速度，加速度，减速度）。编码器用来指示机械零点的输出，可用于此目的。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果有必要，可以把计数器的值锁存到第三个输入捕获寄存器 （ 由另一个定时器产生的捕获信号必须是周期性 ）；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

14.3.13 输入异或功能

TIMx_CR2 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 TIMx_CH1、TIMx_CH2 和 TIMx_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。13.3.18 章节给出了此特性用于连接霍尔传感器的例子。

14.3.14 TIMx 与外部触发同步

TIMx 定时器支持多种外部触发同步模式：复位模式、门控模式和触发模式。

从模式：复位模式

在一个触发输入事件发生时，计数器和它的预分频器能够重新被初始化；此外，如果 TIMx_ CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后更新所有的预装载寄存器（TIMx_ ARR，TIMx_CCRx）。

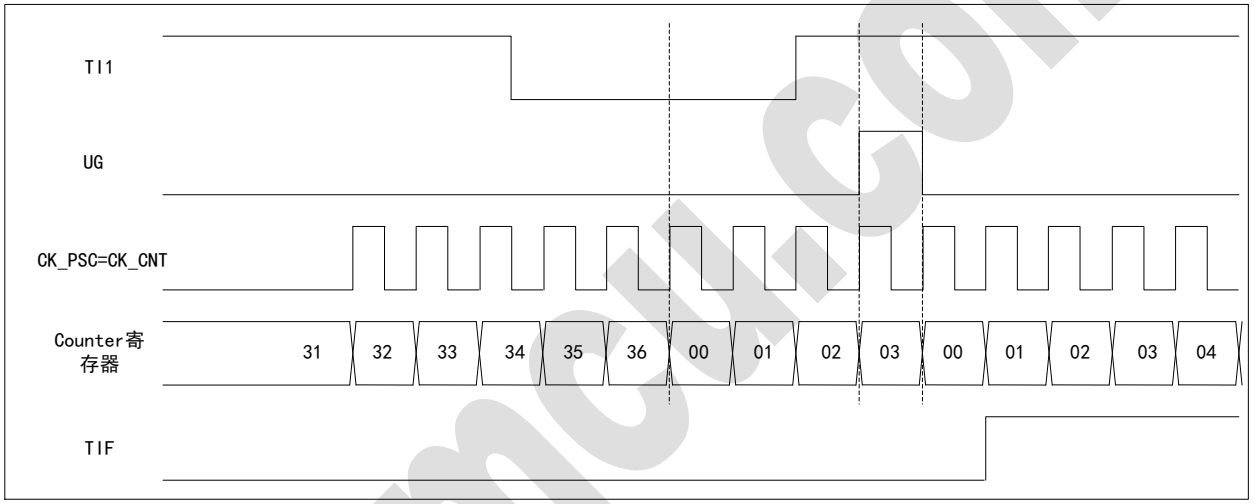
在下面的例子中，TI1 输入端的上升沿清零向上计数器：

- ✧ 配置通道 1 检测 TI1 的上升沿。配置输入滤波器时间（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。配置 TIMx_CCMR1 寄存器中 CC1S=01，只选择输入捕获源。写 TIMx_CCER 寄存器中 CC1P=0 和 CC1NP=0，以确定极性（ 只检测上升沿 ）。
- ✧ 写 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；写 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- ✧ 写 TIMx_CR1 寄存器中 CEN=1，启动计数器。

计数器开始以内部时钟计数，然后正常运行直到 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（TIMx_SR 寄存器中的 TIF 位）被置位，若 TIMx_DIER 寄存器中 TIE 位（中断使能）和 TDE 位（DMA 使能）置位，则产生一个中断请求或一个 DMA 请求。

下图显示了自动重装载寄存器 TIMx_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。

复位模式下的控制电路



从模式：门控模式

门控模式，支持根据所选输入的电平使能计数器。

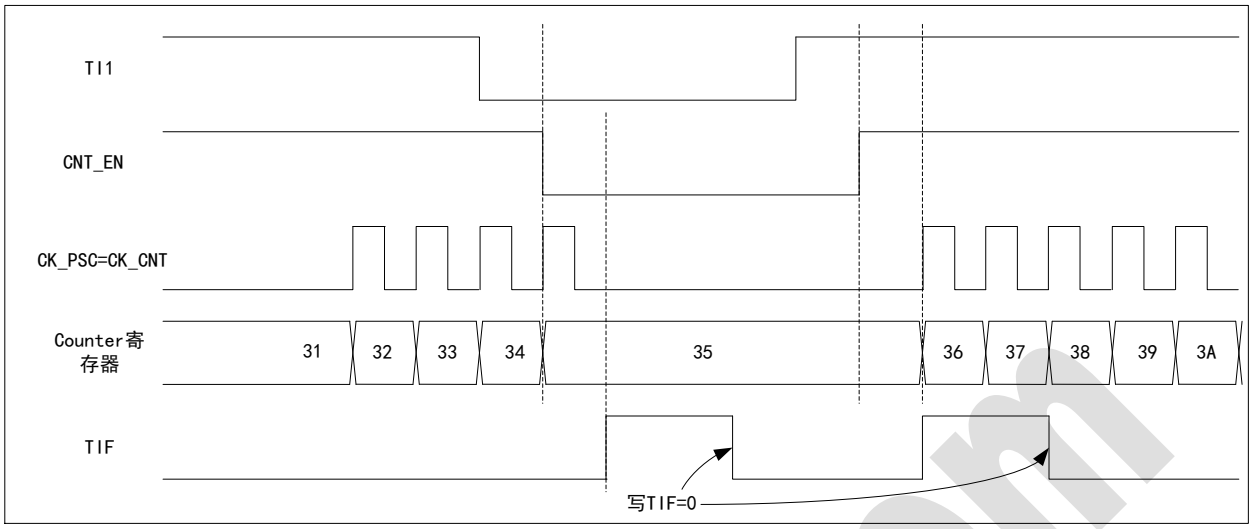
在下面的例子中，计数器只在 TI1 为低时，向上计数：

- ✧ 配置通道 1 检测 TI1 上的低电平。配置输入滤波器时间（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。配置 TIMx_CCMR1 寄存器中 CC1S=01，只选择输入捕获源。写 TIMx_CCER 寄存器中 CC1P=1 和 CC1NP=0，以确定极性（ 只检测低电平 ）。
- ✧ 写 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；写 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- ✧ 写 TIMx_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，计数器开始以内部时钟计数，一旦 TI1 变高则停止计数。计数器开始或停止时，都会置位 TIMx_SR 寄存器中的 TIF 标志。

TI1 上升沿和计数器实际停止之间的延时以及下降沿和计数器实际开始之间的延时，取决于 TI1 输入端的重同步电路。

门控模式下的控制电路



从模式：触发模式

触发模式，支持所选输入端上的事件使能计数器。

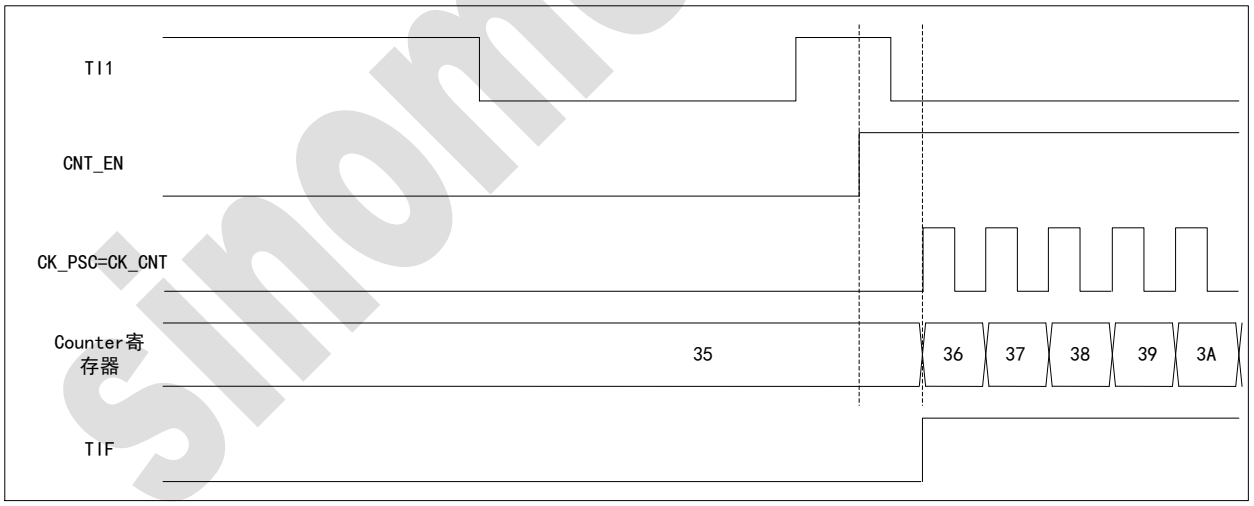
在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- ✧ 配置通道 2 检测 TI2 的上升沿。配置输入滤波器时间（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，所以不需要配置。配置 TIMx_CCMR1 寄存器中 CC2S=01，只选择输入捕获源。写 TIMx_CCER 寄存器中 CC2P=1 和 CC2NP=0，以确定极性（只检测低电平）。
- ✧ 写 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；写 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器以内部时钟开始计数，同时置位 TIF 标志。

TI2 上升沿和计数器实际启动计数之间的延时，取决于 TI2 输入端的重同步电路。

触发器模式下的控制电路



从模式：外部时钟模式 2+触发模式

外部时钟模式 2 可以与另一种从模式（外部时钟模式 1 和编码器模式除外）一起使用。这时，ETR 信号被用作外部时钟的输入，可以选择另一个输入作为触发输入（在复位模式、门控模式或触发模式）。不建议使用 TIMx_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。

在下面的例子中，在 TI1 上出现一个上升沿后，计数器在 ETR 的每一个上升沿向上计数：

- ✧ 通过 TIMx_SMCR 寄存器配置外部触发输入电路，如下：
- ETF=0000：没有滤波
- ETPS=00：不用预分频器
- ETP=0：检测 ETR 的上升沿，
- ECE=1：使能外部时钟模式 2。

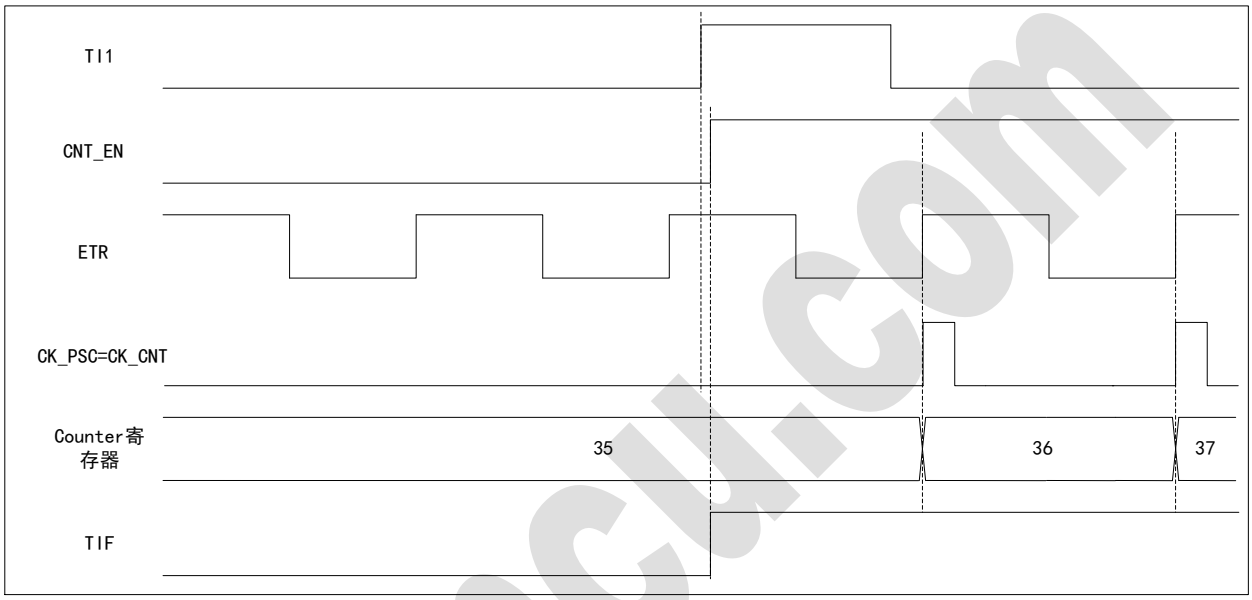
✧ 按如下配置通道 1，检测 TI 的上升沿：

- IC1F=0000：没有滤波
- 触发操作中不使用捕获预分频器，不需要配置
- 写 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源
- 写 TIMx_CCER 寄存器中 CC1P=0 和 CC1NP=0 ，以确定极性（ 只检测上升沿 ）

✧ 写 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式。写 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被置位，计数器开始以 ETR 的上升沿计数。ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

外部时钟模式 2＋触发模式下的控制电路



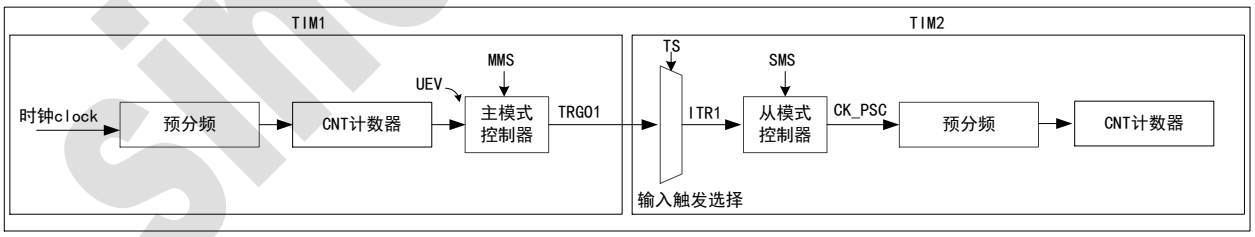
14.3.15 定时器同步

TIMx 定时器在内部相连，用于定时器同步或链接。当一个定时器处于主模式时，它可以对另一个处于从模式的定时器的计数器进行复位、启动、停止或提供时钟等操作。

使用一个定时器作为另一个定时器的预分频器

下图概述了触发选择和主从模式选择。

主/从定时器样例



举例，配置 TIM1 作为 TIM2 的预分频器。参考上图

配置步骤如下：

1、 配置定时器 1 为主模式，它可以在每一个更新事件 UEV 时输出一个周期性的触发信号。在 TIM1_CR2 寄存器的 MMS=‘010’时，每当产生一个更新事件时在 TRGO1 上输出一个上升沿信号。

2、 定时器 1 的 TRGO1 输出连接至定时器 2，设置 TIM2_SMCR 寄存器的 TS=‘000’，配置定时器 2 为使用 ITR1 作为内部触发的从模式。

3、 配置从模式控制器置于外部时钟模式 1（TIM2_SMCR 寄存器的 SMS=111）；这样定时器 2 即可由定时器 1 周期性的上升沿（即定时器 1 的计数器溢出）信号驱动。

4、 最后，置位相应（TIMx_CR1 寄存器）的 CEN 位分别启动两个定时器。

注： 如果 OCx 已被选中为定时器 1 的触发输出 (MMS=1xx)，它的上升沿将作为定时器 2 的计数器时钟源。

使用一个定时器使能另一个定时器

此例中，定时器 2 的使能由定时器 1 的输出比较控制。参考图 130 的连接。只当定时器 1 的 OC1REF 为高时，定时

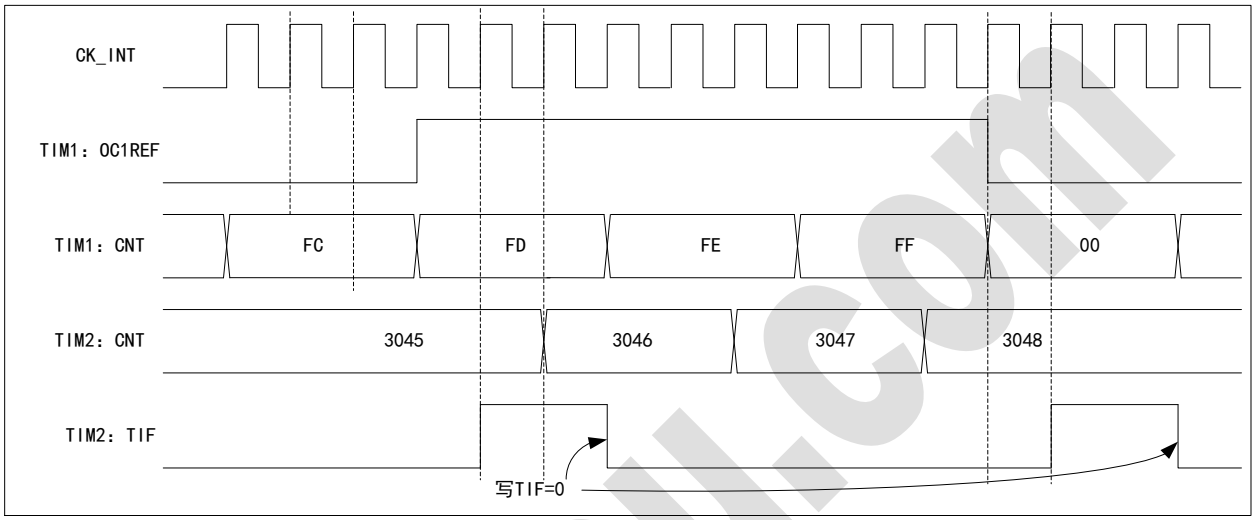
器 2 才对分频后的内部时钟计数。两个定时器的时钟频率为内部时钟 CK_INT 经过预分频器 3 分频获得，即 $f_{CK\_CNT}=f_{CK\_INT}/3$。

配置步骤如下：

1、 配置定时器 1 为主模式，送出它的输出比较参考信号（OC1REF）作为触发输出（TIM1_CR 寄存器的 MMS=100）
2、 配置定时器 1 的 OC1REF 波形（TIM1_CCMR1 寄存器）
3、 配置定时器 2 从定时器 1 获得输入触发（TIM2_SMCR 寄存器的 TS=000）
4、 配置定时器 2 为门控模式（TIM2_SMCR 寄存器的 SMS=101）
5、 置位 TIM2_CR1 寄存器的 CEN=1，使能定时器 2
6、 置位 TIM1_CR1 寄存器的 CEN=1，启动定时器 1

注：计数器 2 的时钟不与计数器 1 的时钟同步，这个模式只影响定时器 2 计数器的使能信号。

定时器 1 的 OC1REF 对定时器 2 门控



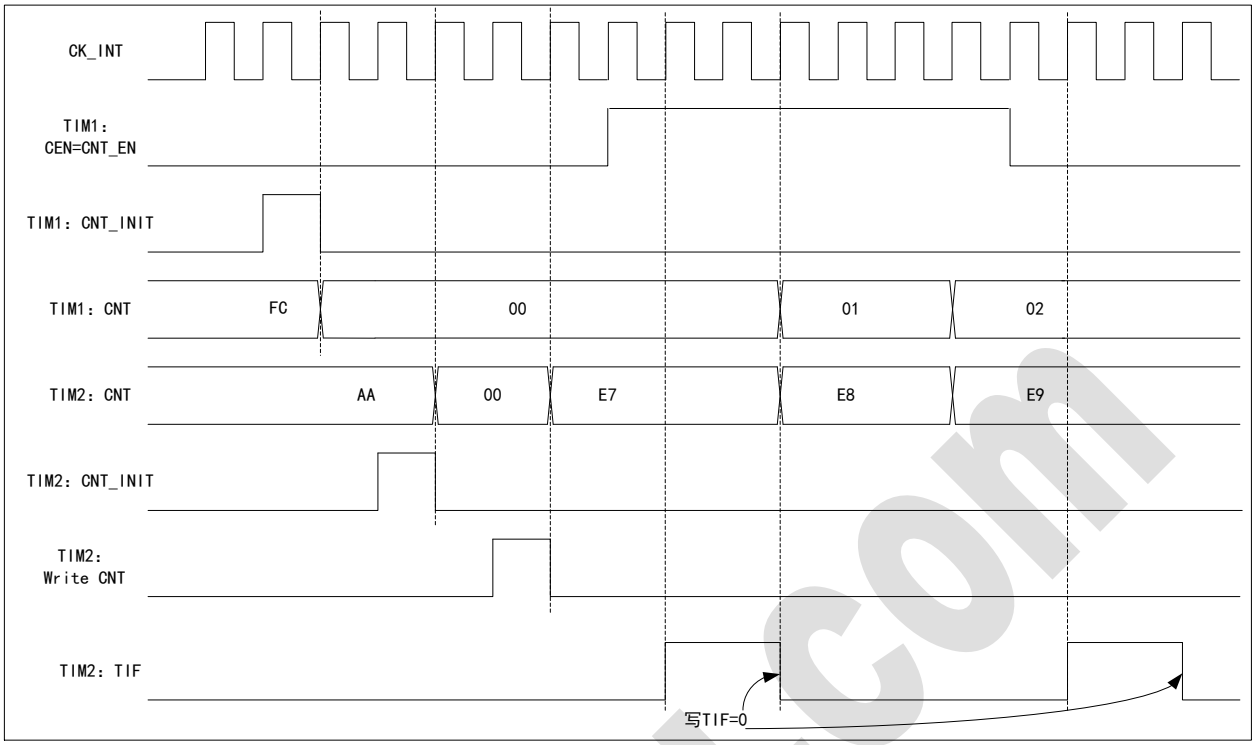
上例中，在定时器 2 启动之前，它们的计数器和预分频器未被初始化，因此它们从当前的数值开始计数。此外支持任意给定值开始启动，仅需要在启动定时器 1 之前对 2 个定时器复位，然后写入需要的的任意数值启动即可。写 TIMx_EGR 寄存器的 UG 位即可复位定时器。

下面的例子中，通过定时器 1 的使能对定时器 2 门控。定时器 1 是主模式并从 0 开始，定时器 2 是从模式并从 0xE7 开始；2 个定时器的预分频器系数相同（为 3）。写‘0’到 TIM1_CR1 的 CEN 位将禁止定时器 1，定时器 2 随即停止。

配置步骤如下：

1、 配置定时器 1 为主模式，送出它的计数器使能信号（CNT_EN）作为触发输出（TIM1_CR2 寄存器的 MMS=001）。
2、 配置定时器 1 的 OC1REF 波形（TIM1_CCMR1 寄存器）。
3、 配置定时器 2 从定时器 1 获得输入触发（TIM2_SMCR 寄存器的 TS=000）
4、 配置定时器 2 为门控模式（TIM2_SMCR 寄存器的 SMS=101）
5、 写 TIM1_EGR 寄存器的 UG=1，复位定时器 1。
6、 写 TIM2_EGR 寄存器的 UG=1，复位定时器 2。
7、 写‘0xE7’至定时器 2 的计数器（TIM2_CNTL），初始化为 0xE7。
8、 写 TIM2_CR1 寄存器的 CEN=1，使能定时器 2。
9、 写 TIM1_CR1 寄存器的 CEN=1，使能定时器 1。
10、写 TIM1_CR1 寄存器的 CEN=0，停止定时器 1。

定时器 1 的计数器使能信号对定时器 2 门控



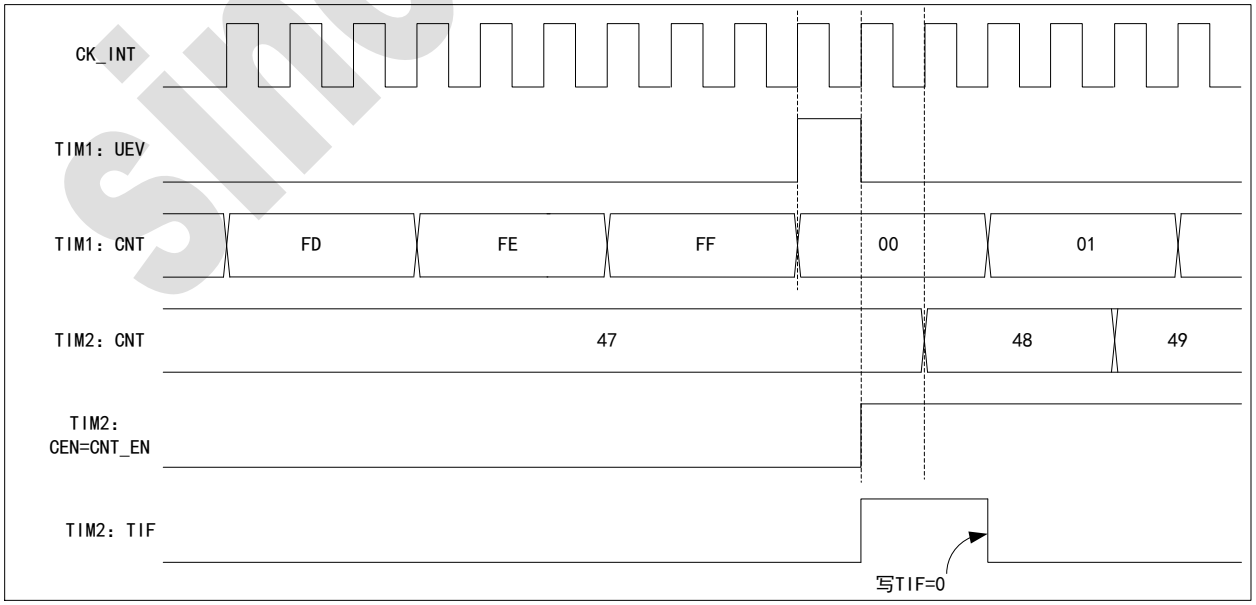
使用一个定时器启动另一个定时器

此例中，使用定时器 1 的更新事件启动定时器 2。一旦定时器 1 产生更新事件，定时器 2 即从它当前的数值（可以是非 0）以分频后的内部时钟开始计数。定时器 2 在收到触发信号时，CEN 位被自动地置‘1’，同时计数器开始计数直到 CEN 位（TIM2_CR1 寄存器）被写‘0’。两个定时器的时钟频率都被配置为内部时钟 CK_INT/3，即 $f_{CK\_CNT}=f_{CK\_INT}/3$。

配置步骤如下：

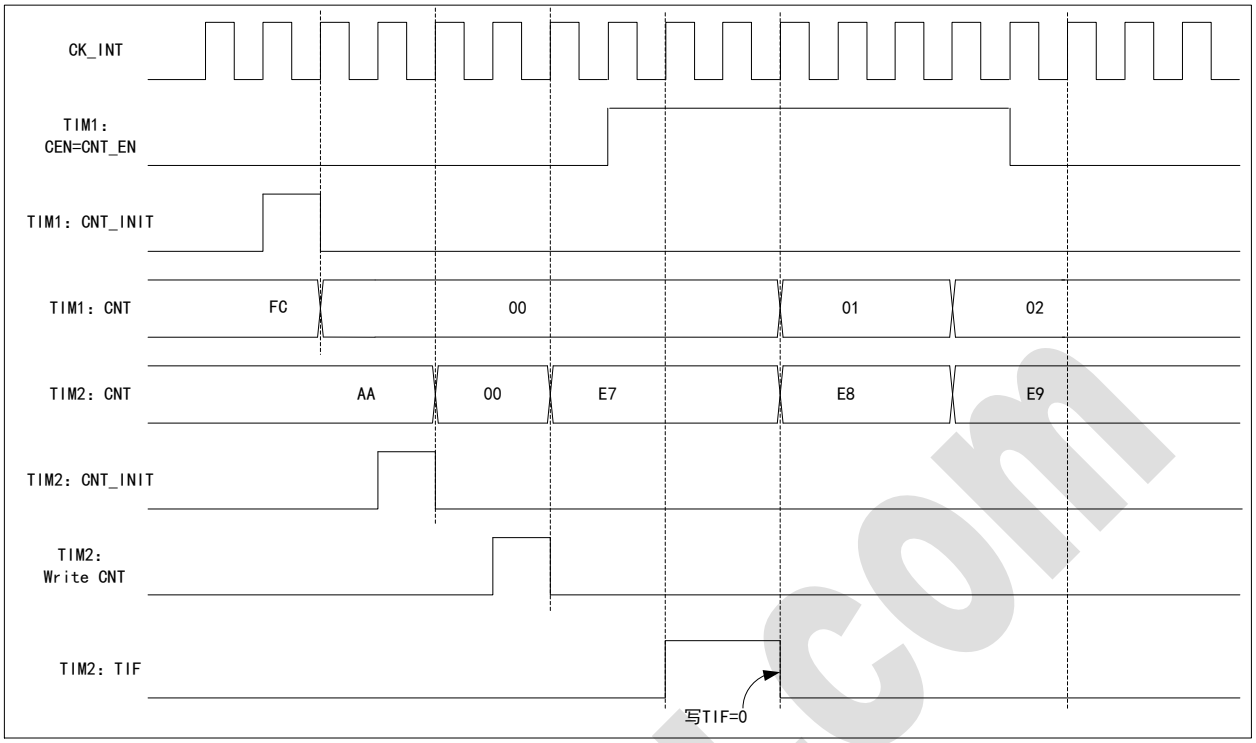
1、 配置定时器 1 为主模式，送出它的更新事件（UEV）做为触发输出（TIM1_CR2 寄存器的 MMS=010）。
2、 配置定时器 1 的周期（TIM1_ARR 寄存器）。
3、 配置定时器 2 从定时器 1 获得输入触发（TIM2_SMCR 寄存器的 TS=000）
4、 配置定时器 2 为触发模式（TIM2_SMCR 寄存器的 SMS=110）
5、 写 TIM1_CR1 寄存器的 CEN=1，启动定时器 1。

定时器 1 的更新事件对定时器 2 触发



与前面的示例一样，上例可以在启动计数之前初始化两个计数器。下图为相同配置情况下（与上图比），使用触发模式而不是门控模式（TIM2_SMCR 寄存器的 SMS=110）的动作。

定时器 1 的计数器使能信号对定时器 2 触发



使用一个外部触发同步地启动 2 个定时器

此例中，定时器 1 的 TI1 输入上升沿使能定时器 1，定时器 1 的使能触发定时器 2 使能。为保证计数器的对齐，定时器 1 必须配置为主/从模式（对于 TI1 为从，对于定时器 2 为主）：

配置步骤如下：

1、 配置定时器 1 为主模式，送出它的使能信号做为触发输出（TIM1_CR2 寄存器的 MMS=‘001’）。

2、 配置定时器 1 为从模式，从 TI1 获得输入触发（TIM1_SMCR 寄存器的 TS=‘100’）。

3、 配置定时器 1 为触发模式（TIM1_SMCR 寄存器的 SMS=‘110’）。

4、 配置定时器 1 为主/从模式，TIM1_SMCR 寄存器的 MSM=‘1’。

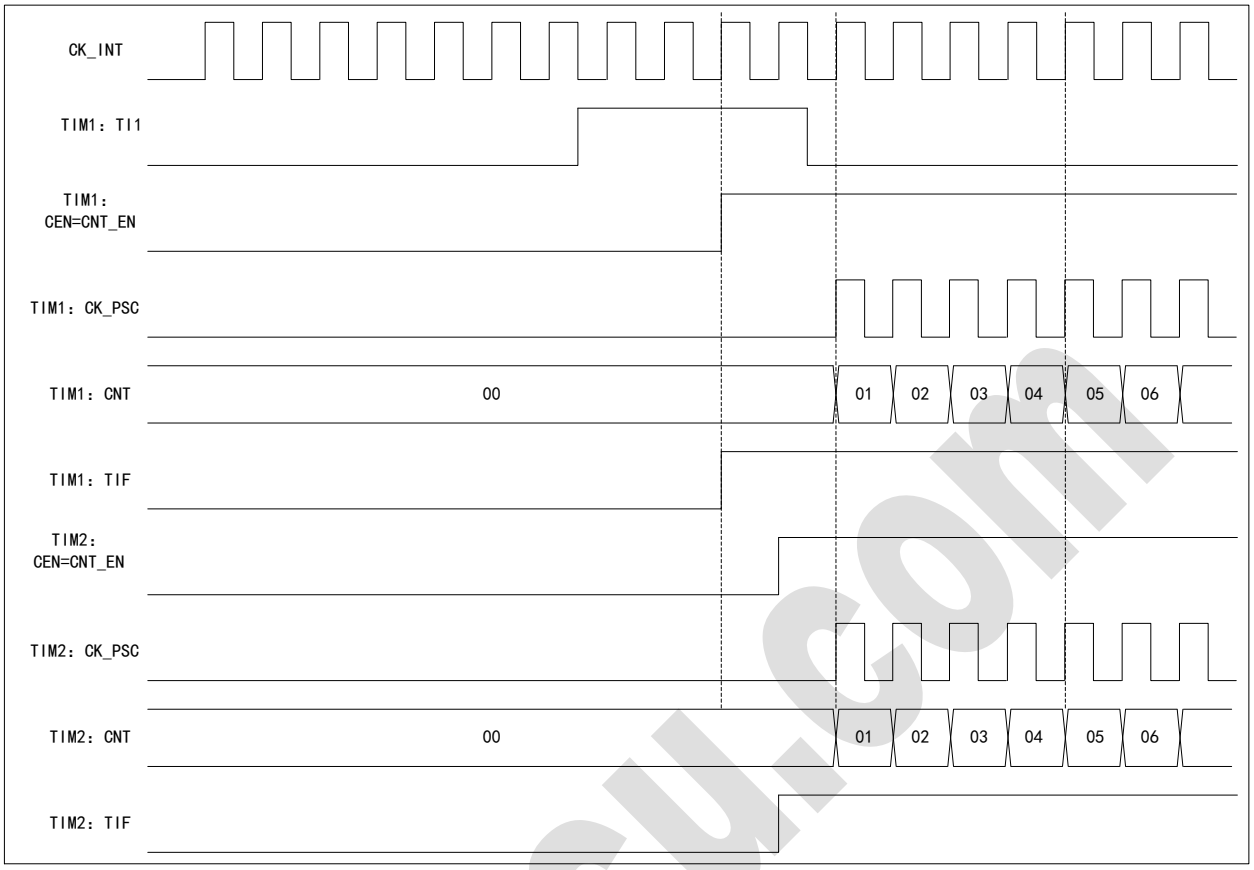
5、 配置定时器 2 从定时器 1 获得输入触发（TIM2_SMCR 寄存器的 TS=000）。

6、 配置定时器 2 为触发模式（TIM2_SMCR 寄存器的 SMS=‘110’）。

当定时器 1 的 TI1 上出现一个上升沿时，两个定时器同步地按照内部时钟开始计数，两个 TIF 标志也同时被置位。

注： 此例中，在启动之前两个定时器都被初始化（设置相应的 UG 位），两个计数器都从 0 开始，但也可以通过写入任意一个计数器寄存器（TIMx_CNT）在定时器间插入一个偏移。下图中能看到，在主/从模式下，定时器 1 的 CNT_EN 和 CK_PSC 之间插入了一个延迟。

定时器 1 的 TI1 输入触发定时器 1 和定时器 2



14.3.16 调试模式

当微控制器进入调试模式时(Cortex-M0 内核停止)，根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以或者继续正常操作，或者停止。

14.4 相关寄存器

14.4.1 寄存器汇总表

基址：0x4000 0000（TIM2）				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	TIM2_CR1	0x0000 0000	TIM2 控制寄存器 1	14.4.2
0x04	TIM2_CR2	0x0000 0000	TIM2 控制寄存器 2	14.4.3
0x08	TIM2_SMCR	0x0000 0000	TIM2 从模式控制寄存器	14.4.4
0x0C	TIM2_DIER	0x0000 0000	TIM2 DMA/ 中断使能寄存器	14.4.5
0x10	TIM2_SR	0x0000 0000	TIM2 状态寄存器	14.4.6
0x14	TIM2_EGR	0x0000 0000	TIM2 事件产生寄存器	14.4.7
0x18	TIM2_CCMR1	0x0000 0000	TIM2 捕捉 / 比较模式寄存器 1	14.4.8
0x1C	TIM2_CCMR2	0x0000 0000	TIM2 捕捉 / 比较模式寄存器 2	14.4.9
0x20	TIM2_CCER	0x0000 0000	TIM2 捕捉 / 比较使能寄存器	14.4.10
0x24	TIM2_CNT	0x0000 0000	TIM2 计数器	14.4.11
0x28	TIM2_PSC	0x0000 0000	TIM2 预分频器	14.4.12
0x2C	TIM2_ARR	0x0000 FFFF	TIM2 自动重装载寄存器	14.4.13
0x30	Res.	-	-	
0x34	TIM2_CCR1	0x0000 0000	TIM2 捕捉 / 比较寄存器 1	14.4.14
0x38	TIM2_CCR2	0x0000 0000	TIM2 捕捉 / 比较寄存器 2	14.4.15
0x3C	TIM2_CCR3	0x0000 0000	TIM2 捕捉 / 比较寄存器 3	14.4.16
0x40	TIM2_CCR4	0x0000 0000	TIM2 捕捉 / 比较寄存器 4	14.4.17
0x44	Res.	-	-	
0x48	TIM2_DCR	0x0000 0000	TIM2 DMA 控制寄存器	14.4.18
0x4C	TIM2_DMAR	0x0000 0000	TIM2 全部传输时 DMA 地址	14.4.19

基址：0x4000 0400（TIM3）				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	TIM3_CR1	0x0000 0000	TIM3 控制寄存器 1	14.4.2
0x04	TIM3_CR2	0x0000 0000	TIM3 控制寄存器 2	14.4.3
0x08	TIM3_SMCR	0x0000 0000	TIM3 从模式控制寄存器	14.4.4
0x0C	TIM3_DIER	0x0000 0000	TIM3 DMA/ 中断使能寄存器	14.4.5
0x10	TIM3_SR	0x0000 0000	TIM3 状态寄存器	14.4.6
0x14	TIM3_EGR	0x0000 0000	TIM3 事件产生寄存器	14.4.7
0x18	TIM3_CCMR1	0x0000 0000	TIM3 捕捉 / 比较模式寄存器 1	14.4.8
0x1C	TIM3_CCMR2	0x0000 0000	TIM3 捕捉 / 比较模式寄存器 2	14.4.9
0x20	TIM3_CCER	0x0000 0000	TIM3 捕捉 / 比较使能寄存器	14.4.10
0x24	TIM3_CNT	0x0000 0000	TIM3 计数器	14.4.11
0x28	TIM3_PSC	0x0000 0000	TIM3 预分频器	14.4.12
0x2C	TIM3_ARR	0x0000 FFFF	TIM3 自动重装载寄存器	14.4.13
0x30	Res.	-	-	
0x34	TIM3_CCR1	0x0000 0000	TIM3 捕捉 / 比较寄存器 1	14.4.14
0x38	TIM3_CCR2	0x0000 0000	TIM3 捕捉 / 比较寄存器 2	14.4.15
0x3C	TIM3_CCR3	0x0000 0000	TIM3 捕捉 / 比较寄存器 3	14.4.16
0x40	TIM3_CCR4	0x0000 0000	TIM3 捕捉 / 比较寄存器 4	14.4.17
0x44	Res.	-	-	
0x48	TIM3_DCR	0x0000 0000	TIM3 DMA 控制寄存器	14.4.18
0x4C	TIM3_DMAR	0x0000 0000	TIM3 全部传输时 DMA 地址	14.4.19

14.4.2 TIM2 和 TIM3 控制寄存器 1（TIM2_CR1 和 TIM3_CR1）

TIM2_CR1 / TIM3_CR1			地址偏移	0x00		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	CKD[1:0]	
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:10]	-	保留，保持复位值
BIT[9:8]	CKD[1:0]	时钟分频（Clock division），软件改写 CK_INT 为定时器时钟，t_{DTS} 为死区时间发生器（dead-time）和数字滤波器（ETR,TIx）采样时钟，此位定义了 t_{CK_INT} 和 t_{DTS} 的比率。 00：$t_{DTS} = t_{CK\_INT}$ 01：$t_{DTS} = 2 * t_{CK\_INT}$ 10：$t_{DTS} = 4 * t_{CK\_INT}$ 11：保留，禁止使用
BIT[7]	ARPE	自动重装载预装载允许位（Auto-reload preload enable） 0: TIMx_ARR 寄存器没有缓冲 1: TIMx_ARR 寄存器有缓冲
BIT[6:5]	CMS[1:0]	中央对齐模式选择 (Center-aligned mode selection) 00：边沿对齐模式。计数器依据方向位 (DIR) 向上或向下计数。 01：中心对齐模式 1。计数器交替地向上和向下计数。输出比较中断标志位在计数器向下计数时被置位，配置 (TIMx_CCMRx 寄存器中 CCxS=00) 通道输出比较中断。 10：中心对齐模式 2。计数器交替地向上和向下计数。输出比较中断标志位在计数器向上计数时被置位，配置 (TIMx_CCMRx 寄存器中 CCxS=00) 通道输出比较中断。 11：中心对齐模式 3。计数器交替地向上和向下计数。输出比较中断标志位在计数器向上和向下计数时均被置位，配置 (TIMx_CCMRx 寄存器中 CCxS=00) 通道输出比较中断。 *注：计数器开启时 (CEN=1)，不允许从边沿对齐模式转换到中心对齐模式。*

BIT[4]	DIR	方向 (Direction) 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。*
BIT[3]	OPM	单脉冲模式 (One pulse mode) 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。
BIT[2]	URS	更新请求源 (Update request source) 软件置位和清零，选择 UEV 事件的源 0：下述任一事件产生更新中断和 DMA 请求（如果 DMA 使能）： - 计数器上溢/下溢 - 置位 UG 位 - 从模式（slave mode）控制器产生的更新 1：仅计数器上溢/下溢产生更新中断或 DMA 请求（如果 DMA 使能）。 1：如果使能了更新中断或 DMA 请求，则只有计数器溢出 / 下溢才产生更新中断或 DMA 请求。
BIT[1]	UDIS	禁止更新 (Update disable) 软件置位和清零，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 - 置位 UG 位 - 从模式（slave mode）控制器产生的更新 具有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。如果置位 UG 位或从模式（slave mode）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。
BIT[0]	CEN	使能计数器 (Counter enable) 0：禁止计数器 1：使能计数器 *注：只有软件设置了 CEN 位后，外部时钟、门控模式和编码器模式才能工作。但触发模式可以自动地通过硬件设置 CEN 位。*

14.4.3 TIM2 和 TIM3 控制寄存器 2 (TIM2_CR2 和 TIM3_CR2)

TIM2_CR2 / TIM3_CR2			地址偏移	0x04		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	TI1S	MMS[2:0]			CCDS	-	-	-
R/W	rw	rw	rw	rw	rw	-	-	-
复位值	0	0	0	0	0	-	-	-

BIT[15:8]	-	保留，保持复位值
BIT[7]	TI1S	TI1 选 择 (TI1 selection) 0：TIMx_CH1 引脚连到 TI1 输入； 1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 引脚经异或(XOR)后连到 TI1 输入。
BIT[6:4]	MMS[1:0]	主模式选择 (Master mode selection) 这 3 位用于选择在主模式下送到从定时器的同步信息 (TRGO)。可能的组合如下： 000：复位 - TIMx_EGR 寄存器的 UG 位被用于作为触发输出 (TRGO)。如果是触发输入产生的复位 (从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 001：使能 - 计数器使能信号 CNT_EN 被用于作为触发输出 (TRGO)。可用于同时启动多个定时器或控制在一段时间内使能从定时器。在门控模式下，计数器使能信号是 CEN 控制位和触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式 (见 TIMx_SMCR 寄

		存器中 MSM 位的描述)。 010：更新 - 更新事件被选为触发输入 (TRGO)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：比较脉冲 - 在捕获发生或比较匹配发生，CC1IF 标志被置位时 (即使它已经为高)，触发输出发送一个正脉冲 (TRGO)。 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)。
BIT[3]	CCDS	捕获/比较的 DMA 选择 (capture/compare DMA selection) 0：当发生 CCx 事件时，送出 CCx DMA 请求 1：当发生更新事件时，送出 CCx DMA 请求
BIT[2:0]	-	保留，保持复位值

14.4.4 TIM2 和 TIM3 从模式控制寄存器 (TIM2_SMCR 和 TIM3_SMCR)

TIM1_SMCR			地址偏移	0x08		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	ETP	ECE	ETPS[1:0]		ETF[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	MSM	TS[2:0]			OCCS	SMS[2:0]		
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15]	ETP	外部触发极性 (External trigger polarity) 该位选择是用 ETR 或 $\overline{ETR}$ 作为触发操作 0：ETR 不反相，高电平或上升沿有效 1：ETR 被反相，低电平或下降沿有效
BIT[14]	ECE	外部时钟使能位 (External clock enable) 该位启用外部时钟模式 2 0：禁止外部时钟模式 2； 1：使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。 *注 1：设置 ECE 位，与选择外部时钟模式 1 并将 TRGI 连到 ETRF (SMS=111 和 TS=111) 具有相同功效。* *注 2：下述从模式可以与外部时钟模式 2 同时使用：复位模式，门控模式和触发模式；但是，这时 TRGI 不能连到 ETRF(TS 位不能是 '111') 。* *注 3：外部时钟模式 1 和外部时钟模式 2 同时被使能时，外部时钟的输入是 ETRF。*
BIT[13:12]	ETPS[1:0]	外部触发预分频 (External trigger prescaler) 外部触发信号 ETRP 的频率最多是 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 00：关闭预分频； 01：ETRP 频率除以 2； 10：ETRP 频率除以 4； 11：ETRP 频率除以 8。
BIT[11:8]	ETF[3:0]	外部触发滤波 (External trigger filter) 这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。数字滤波器是一个事件计数器，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 $f_{SAMPLING}=f_{DTS}$ 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：请注意，当 ETF[3:0]=1/2/3 时，公式中 f_{DTS} 被 f_{CK_INT} 取代。*
BIT[7]	MSM	主/从模式 (Master/slave mode) 0：无作用 1：触发输入(TRGI)上的事件被延迟，以允许在当前定时器与它的从定时器间的完美同步(通过 TRGO)。当要求把几个定时器同步到一个单一的外部事件时，这是很有用的
BIT[6:4]	TS[2:0]	触发选择 (Trigger selection) 这 3 位选择用于同步计数器的触发输入。 000：内部触发 0 (ITR0) 001：内部触发 1 (ITR1) 010：内部触发 2 (ITR2) 011：内部触发 3 (ITR3) 100：TI1 的边沿检测器 (TI1F_ED) 101：滤波后的定时器输入 1 (TI1FP1) 110：滤波后的定时器输入 2 (TI2FP2) 111：外部触发输入 (ETRF) 更多有关 ITRx 的细节，参见下表：TIM2 和 TIM3 内部触发的连接。 *注：这些位只能在未用到(如 SMS=000) 时被改变，以避免在改变时产生错误的边沿检测。*
BIT[3]	OCCS	OCREF 清零信号源选择 (OCREF clear selection) 0：OCREF_CLR_INT 被连接至 OCREF_CLR 输入 1：OCREF_CLR_INT 被连接至 ETRF
BIT[2:0]	SMS	从模式选择 (Slave mode selection) 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相关(见输入控制寄存器和控制寄存器的说明) 000：关闭从模式 – 如果 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1 – 根据 TI1FP2 的电平，计数器在 TI2FP1 的边沿向上/下计数。 010：编码器模式 2 – 根据 TI2FP1 的电平，计数器在 TI1FP2 的边沿向上/下计数。 011：编码器模式 3 – 根据其他输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 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，并且产生一次更新寄存器。 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 1：如果 TI1F_ED 被选为触发输入 (TS=100b) 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。* *注 2：在接收主定时器事件之前，从定时器的时钟必须使能，并且在接收主定时器的触发信号过程中，不能动态改变。*

表：TIM2 和 TIM3 内部触发连接

Slave TIM	ITR0 (TS = 000)	ITR1 (TS = 001)	ITR2 (TS = 010)	ITR3 (TS = 011)
TIM2	TIM1	Res.	TIM3	TIM14
TIM3	TIM1	TIM2	Res.	TIM14

14.4.5 TIM2 和 TIM3 DMA/ 中断允许寄存器 (TIM2_DIER 和 TIM3_DIER)

TIM2_DIER / TIM3_DIER			地址偏移	0x0C		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

控制位	-	TDE	-	CC4DE	CC3DE	CC2DE	CC1DE	UDE
R/W	-	rw	-	rw	rw	rw	rw	rw
复位值	-	0	-	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	TIE	-	CC4IE	CC3IE	CC2IE	CC1IE	UIE
R/W	-	rw	-	rw	rw	rw	rw	rw
复位值	-	0	-	0	0	0	0	0

BIT[15]	-	保留，保持复位值
BIT[14]	TDE	允许触发 DMA 请求 (Trigger DMA request enable) 0: 触发 DMA 请求禁止 1: 触发 DMA 请求允许
BIT[13]	-	保留，保持复位值，读取固定为 0
BIT[12]	CC4DE	捕获/比较 4 DMA 请求使能 0: CC4 DMA 请求禁止 1: CC4 DMA 请求允许
BIT[11]	CC3DE	捕获/比较 3 DMA 请求使能 0: CC3 DMA 请求禁止 1: CC3 DMA 请求允许
BIT[10]	CC2DE	捕获/比较 2 DMA 请求使能 0: CC2 DMA 请求禁止 1: CC2 DMA 请求允许
BIT[9]	CC1DE	捕获/比较 1 DMA 请求使能 0: CC1 DMA 请求禁止 1: CC1 DMA 请求允许
BIT[8]	UDE	更新 DMA 请求使能 0: 更新 DMA 请求禁止 1: 更新 DMA 请求允
BIT[7]	-	保留，保持复位值
BIT[6]	TIE	触发中断使能 0: 触发中断禁止 1: 触发中断允许
BIT[5]	-	保留，保持复位值
BIT[4]	CC4IE	捕获/比较 4 中断使能 0: CC4 中断禁止 1: CC4 中断允许
BIT[3]	CC3IE	捕获/比较 3 中断使能 0: CC3 中断禁止 1: CC3 中断允许
BIT[2]	CC2IE	捕获/比较 2 中断使能 0: CC2 中断禁止 1: CC2 中断允许
BIT[1]	CC1IE	捕获/比较 1 中断使能 0: CC1 中断禁止 1: CC1 中断允许
BIT[0]	UIE	更新中断使能 0: 更新中断禁止 1: 更新中断允许

14.4.6 TIM2 和 TIM3 状态寄存器 (TIM2_SR 和 TIM3_SR)

TIM2_SR / TIM3_SR			地址偏移	0x10		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	CC4OF	CC3OF	CC2OF	CC1OF	-
R/W	-	-	-	rc_w0	rc_w0	rc_w0	rc_w0	-
复位值	-	-	-	0	0	0	0	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	TIF	-	CC4IF	CC3IF	CC2IF	CC1IF	UIF

R/W	-	rc_w0	-	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0
复位值	-	0	-	0	0	0	0	0

BIT[15:13]	-	保留，保持复位值
BIT[12]	CC4OF	捕获/比较 4 重复捕获标志 ，参考 CC1OF 描述
BIT[11]	CC3OF	捕获/比较 3 重复捕获标志 ，参考 CC1OF 描述
BIT[10]	CC2OF	捕获/比较 2 重复捕获标志 ，参考 CC1OF 描述
BIT[9]	CC1OF	捕获/比较 1 重复捕获标志 (Capture/Compare 1 overcapture flag) 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。 软件写 0 可清除该位。 0：无重复捕获产生 1：当 CC1IF 的状态已经为‘1’，计数器的值被捕获到 TIMx_CCR1 寄存器
BIT[8:7]	-	保留，保持复位值
BIT[6]	TIF	触发器中断标志 (Trigger interrupt flag) 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，或门控模式下的任一边沿 ） 时由硬件对该位置’1’。它由软件清’0’。 0：无触发器事件产生 1：触发中断等待响应
BIT[5]	-	保留，保持复位值
BIT[4]	CC4IF	捕获/比较 4 中断标志 ，参考 CC1IF
BIT[3]	CC3IF	捕获/比较 3 中断标志 ，参考 CC1IF
BIT[2]	CC2IF	捕获/比较 2 中断标志 ，参考 CC1IF
BIT[1]	CC1IF	捕获/比较 1 中断标志 **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 TIMx_ CR1 寄存器的 CMS 位)。 它由软件清’0’。 0：无匹配发生 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的内容大于 TIMx_APR 的内容时，在向上或向上/下计数模式时计数器上溢，或向下计数模式时的计数器下溢条件下，CC1IF 位变高 **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’，它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无输入捕获产生 1：计数器值被捕获至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)
BIT[0]	UIF	更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置’1’。它由软件清’0’。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置’1’： - 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器（repetition counter）数值上溢或下溢时 （ 重复计数器=0 时产生更新事件 ）。 - 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 进而实现软件对计数器 CNT 重新初始化时。 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。 参考章节：TIM1 从模式控制寄存器 (TIM1_SMCR)

14.4.7 TIM2 和 TIM3 事件产生寄存器 (TIM2_EGR 和 TIM3_EGR)

TIM2_EGR / TIM3_EGR			地址偏移	0x14		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	TG	-	CC4G	CC3G	CC2G	CC1G	UG
R/W	-	w	-	w	w	w	w	w

复位值	-	0	-	0	0	0	0	0

BIT[15:7]	-	保留，保持复位值
BIT[6]	TG	触发产生（Trigger generation） 该位由软件置‘1’，用于产生一个事件，由硬件自动清‘0’。 0：无动作 1：TIMx_SR 中 TIF 标志置 1，若开启对应的中断和 DMA，则产生相应的中断和 DMA
BIT[5]	-	保留，保持复位值
BIT[4]	CC4G	捕获/比较 4 发生参考 CC1G 描述
BIT[3]	CC3G	捕获/比较 3 发生参考 CC1G 描述
BIT[2]	CC2G	捕获/比较 2 发生参考 CC1G 描述
BIT[1]	CC1G	捕获/比较 1 发生（Capture/Compare 1 generation） 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 1 上产生一个捕获/比较事件 **若通道 CC1 配置为输出：** 设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 **若通道 CC1 配置为输入：** 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF=1。
BIT[0]	UG	产生更新事件（Update generation） 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（寄存器）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 TIMx_ARR 的值

14.4.8 TIM2 和 TIM3 捕获/比较模式寄存器 1（TIM2_CCMR1 和 TIM3_CCMR1）

TIM2_CCMR1 / TIM3_CCMR1			地址偏移	0x18		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]	
	IC2F[3:0]				IC2PSC[1:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	IC1F[3:0]				IC1PSC[1:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

注：CCxS 位定义通道方向，可用于输入（捕获模式）或输出（比较模式）。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能，ICxx 描述了通道 在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

输出比较模式：

BIT[15]	OC2CE	输出比较 2 清 0 允许
BIT[14:12]	OC2M[2:0]	输出比较 4 模式
BIT[11]	OC2PE	输出比较 2 预分频允许
BIT[10]	OC2FE	输出比较 2 快速使能
BIT[9:8]	CC2S[1:0]	捕获/比较 2 选择（capture/compare 2 selection） 该 2 位定义通道的方向（输入/输出），及输入信号的选择： 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 TIMx_SMCR 寄存器的 TS 位选择） *注：CC2S 仅在通道关闭时（TIMx_CCER 寄存器的 CC2E=0）才是可写的。*
BIT[7]	OC1CE	输出比较 1 清 0 允许（output compare 1 clear enable）

		0: OC1REF 不受 ETRF 输入的影响； 1: 一旦检测到 ETRF 输入高电平，清除 OC1REF=0。
BIT[6:4]	OC1M	输出比较模式 1 (Output Compare 1 mode) 该 3 位定义了输出参考信号 OC1REF 的行为，而 OC1REF 决定了 OC1、OC1N 的电平。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 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用（此模式用于产生一个时基）； 001：匹配时，设置通道 1 为有效电平。当 TIMx_CNT = TIMx_CCR1 时，强制 OC1REF 为高。 010：匹配时，设置通道 1 为无效电平。当 TIMx_CNT = TIMx_CCR1 时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_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，当 TIMx_CNT<TIMx_CCR1 时，通道 1 为有效电平(OC1REF=1)，否则为无效电平(OC1REF=0)； - 在向下计数时，当 TIMx_CNT>TIMx_CCR1 时，通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2 - 在向上计数时，当 TIMx_CNT<TIMx_CCR1 时，通道 1 为无效电平，否则为有效电平； - 在向下计数时，当 TIMx_CNT>TIMx_CCR1 时，通道 1 为有效电平，否则为无效电平。 *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。*
BIT[3]	OC1PE	输出比较 1 预装载允许 (Output Compare 1 preload enable) 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1： 开启 TIMx_CCR1 寄存器的预装载功能， 读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 *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：仅在单脉冲模式下 (TIMx_CR1 寄存器的 OPM=1) ，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。*
BIT[2]	OC1FE	输出比较 1 快速使能 (Output Compare 1 fast enable) 该位用于加快 CC 输出对触发输入事件的响应。 0：CC1 的正常操作依赖于计数器与 CCR1 的值，即使工作于触发器状态。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期 1：输入到触发器的有效沿的作用等同于发生了一次比较匹配。因此，OC 被设置为比较电平（独立于比较结果）。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。 OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
BIT[1:0]	CC1S	捕获/比较 1 选择 (Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入信号的选择： 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 (由 TIMx_SMCR 寄存器的 TS 位选择) *注：CC1S 仅在通道关闭时 (TIMx_CCER 寄存器的 CC1E=0) 才是可写的。*

输入捕获模式：

BIT[15:12]	IC2F	输入捕获 2 滤波器
BIT[11:10]	IC2PSC	输入捕获 2 预分频
BIT[9:8]	CC2S	捕获/比较 2 选择（capture/compare 2 selection） 该 2 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 TIMx_SMCR 寄存器的 TS 位选择）。 *注：CC2S 仅在通道关闭时（TIMx_CCER 寄存器的 CC2E=0）才是可写的。*
BIT[7:4]	IC1F[3:0]	输入捕获 1 滤波器（Input capture 1 filter） 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 f_{DTS} 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：当 ICxF[3:0]=1、2 或 3 时，公式中的 fDTS 由 CK_INT 替代。
BIT[3:2]	IC1PSC	输入捕获 1 预分频器 这 2 位定义了 CC1 输入(IC1) 的预分频系数。一旦 CC1E=0(TIMx_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
BIT[1:0]	CC1S	捕获/比较 1 选择 这 2 位定义通道的方向（输入/输出），及输入信号的选择： 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（由 TIMx_SMCR 寄存器的 TS 位选择）。 *注：CC1S 仅在通道关闭时（TIMx_CCER 寄存器的 CC3E=0）才是可写的。*

14.4.9 TIM2 和 TIM3 捕获/比较模式寄存器 2（TIM2_CCMR2 和 TIM3_CCMR2）

TIM2_CCMR2 / TIM3_CCMR2			地址偏移		0x1C	复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	OC4CE	OC4M[2:0]			OC4PE	OC4FE	CC4S[1:0]	
	IC4F[3:0]				IC4PSC[1:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	OC3CE	OC3M[2:0]			OC3PE	OC3FE	CC3S[1:0]	
	IC4F[3:0]				IC3PSC[1:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

注：CCxS 位定义通道方向，可用于输入（捕获模式）或输出（比较模式）。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能，ICxx 描述了通道 在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

输出比较模式：

BIT[15]	OC4CE	输出比较 4 清 0 允许

BIT[14:12]	OC4M[2:0]	输出比较 4 模式
BIT[11]	OC4PE	输出比较 4 预分频允许
BIT[10]	OC4FE	输出比较 4 快速使能
BIT[9:8]	CC4S[1:0]	捕获/比较 4 选择（capture/compare 4 selection） 该 2 位定义通道的方向（输入/输出），及输入信号的选择： 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（由 TIMx_SMCR 寄存器的 TS 位选择 ） 注：CC4S 仅在通道关闭时（TIMx_CCER 寄存器的 CC4E=0）才是可写的。
BIT[7]	OC3CE	输出比较 3 清 0 允许（output compare 3 clear enable）
BIT[6:4]	OC3M	输出比较 3 模式（Output Compare 3 mode）
BIT[3]	OC3PE	输出比较 3 预分频允许（Output Compare 3 preload enable）
BIT[2]	OC3FE	输出比较 3 快速使能（Output Compare 3 fast enable）
BIT[1:0]	CC3S	捕获/比较 3 选择（Capture/Compare 3 selection） 该 2 位定义通道的方向（输入/输出），及输入信号的选择： 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 TIMx_SMCR 寄存器的 TS 位选择）。 注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0）才是可写的。

输入捕捉模式：

BIT[15:12]	IC4F	输入捕获 4 滤波器
BIT[11:10]	IC4PSC	输入捕获 4 预分频
BIT[9:8]	CC4S	捕获/比较 4 选择 这 2 位定义通道的方向（输入/输出），及输入信号的选择： 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（ 由 TIMx_SMCR 寄存器的 TS 位选择 ）。 注：CC4S 仅在通道关闭时（TIMx_CCER 寄存器的 CC4E=0）才是可写的。
BIT[7:4]	IC3F	输入捕获 3 滤波器
BIT[3:2]	IC3PSC	输入比获 3 预分频
BIT[1:0]	CC3S	捕获/比较 3 选择 这 2 位定义通道的方向（输入/输出），及输入信号的选择： 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 TIMx_SMCR 寄存器的 TS 位选择 ）。 注：CC3S 仅在通道关闭时（TIMx_CCER 寄存器的 CC3E=0）才是可写的。

14.4.10 TIM2 和 TIM3 捕获/比较使能寄存器（TIM2_CCER 和 TIM3_CCER）

TIM2_CCER/ TIM3_CCER			地址偏移	0x20		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CC4NP	-	CC4P	CC4E	CC3NP	-	CC3P	CC3E
R/W	rw	-	rw	rw	rw	-	rw	rw
复位值	0	-	0	0	0	-	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CC2NP	-	CC2P	CC2E	CC1NP	-	CC1P	CC1E
R/W	rw	-	rw	rw	rw	-	rw	rw
复位值	0	-	0	0	0	-	0	0

BIT[15]	CC4NP	捕获/比较 4 输出极性，详见 CC1NP

BIT[14]	-	保留，保持复位值
BIT[13]	CC4P	捕获/比较 4 输出极性，详见 CC1P
BIT12]	CC4E	捕获/比较 4 输出使能，详见 CC1E
BIT[11]	CC3NP	捕获/比较 3 输出极性，详见 CC1NP
BIT[10]	-	保留，保持复位值
BIT[9]	CC3P	捕获/比较 3 输出极性，详见 CC1P
BIT[8]	CC3E	捕获/比较 3 输出使能，详见 CC1E
BIT[7]	CC2NP	捕获/比较 2 输出极性，详见 CC1NP
BIT[6]	-	保留，保持复位值
BIT[5]	CC2P	捕获/比较 2 输出极性，详见 CC1P
BIT[4]	CC2E	捕获/比较 2 输出使能，详见 CC1E
BIT[3]	CC1NP	捕获/比较 1 输出极性（Capture/Compare 1 output polarity） **CC1 通道配置为输出** 无功能 **CC1 通道配置为输入** 本位用于和 CC1P 联合定义 TI1FP1 和 TI2FP1 的极性，详见 CC1P
BIT[2]	-	保留，保持复位值
BIT[1]	CC1P	捕获/比较 1 输出极性（Capture/Compare 1 output polarity） **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：** CC1NP/CC1P 位选择在触发或捕获模式下 TI1FP1 和 TI2FP1 的有效极性。 00：非反相/上升沿 - TIxFP1 的上升沿（在复位、外部时钟或触发模式下的捕获或触发操作） - TIxFP1 非反相（在门控模式或编码模式） 01：反相/下降沿 - TIxFP1 的下降沿（在复位、外部时钟或触发模式下的捕获或触发操作）， - TIxFP1 反相（在门控模式或编码模式） 00：保留，此配置不用 11：非反相/上升或下降沿 - TIxFP1 的上升沿和下降沿（在复位、外部时钟或触发模式下的捕获或触发操作） - TIxFP1 非反相（在门控模式） - 在编码模式下不能使用此配置
BIT[0]	CC1E	捕获/比较 1 输出使能(Capture/Compare 1 output enable) **CC1 通道配置为输出：** 0：关闭－OC1 未激活。 1：开启－OC1 信号输出到对应的输出引脚。 **CC1 通道配置为输入：** 本位用于决定捕获的计数器值是否要装载到捕获/比较寄存器 1(TIMx_CCR1)。 0：捕获禁止 1：捕获允许

标准 OCx 通道的输出控制位

CCxE bit	OCx 输出状态
0	输出禁止 （ 不由定时器驱动 ） OCx=0, OCx_EN=0
1	OCx = OCxREF + 极性，OCx_EN=1

注：连接到标准 OCx 通道的外部 I/O 引脚的状态取决于 OCx 通道状态和 GPIO 寄存器。

14.4.11 TIM2 和 TIM3 计数器 (TIM2_CNT 和 TIM3_CNT)

TIM2_CNT / TIM3_CNT			地址偏移	0x24		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bi25	Bit24
控制位	CNT[31:16](TIM2 only)							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	CNT[23:16](TIM2 only)							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CNT[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CNT[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	CNT[31:16]	高 16 位值（仅 TIM2 支持） 。
BIT[15:0]	CNT[15:0]	低 16 位值

14.4.12 TIM2 和 TIM3 预分频 (TIM2_PSC 和 TIM3_PSC)

TIM2_PSC / TIM3_PSC			地址偏移	0x28		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	PSC[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	PSC[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	PSC[15:0]	预分频值(Prescaler value) 计数器的时钟频率 (CK_CNT) = f_{CK_PSC}/(PSC[15:0]+1)。每次当更新事件产生时，PSC 的值被装入当前预分频器寄存器；更新事件包括计数器被 TIM_EGR 的 UG 位清‘0’或被配置为复位模式的触发器控制器清‘0’。

14.4.13 TIM2 和 TIM3 自动重装寄存器 (TIM2_ARR 和 TIM3_ARR)

TIM2_ARR / TIM3_ARR			地址偏移	0x2C		复位值	0xFFFF FFFF	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bi25	Bit24
控制位	ARR[31:16](TIM2 only)							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	ARR[23:16](TIM2 only)							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	ARR[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	ARR[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	ARR[31:16]	自动重装载高 16 位值 (仅 TIM2 支持)。
BIT[15:0]	ARR[15:0]	自动重装载的值 (Auto-reload value) ARR 包含了将要装载入的实际自动重装载寄存器的值。详细参考章节：14.3.1 时基单元，有关 ARR 的更新和行为。 当自动重装载的值为空时，计数器不工作。

14.4.14 TIM2 和 TIM3 捕获/比较寄存器 1 (TIM2_CCR1 和 TIM3_CCR1)

TIM2_CCR1 / TIM3_CCR1			地址偏移	0x34		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bi25	Bit24
控制位	CCR1[31:16](TIM2 only)							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	CCR1[23:16](TIM2 only)							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CCR1[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CCR1[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	CCR1[31:16]	捕获/比较通道 1 的值高 16 位（仅 TIM2 支持）
BIT[15:0]	CCR1[15:0]	捕获/比较通道 1 的值 **若 CC1 通道配置为输出：** CCR1 是要加载到实际捕获/比较 1 寄存器的值（预装载值）。 如果在 TIMx_CCMR1 寄存器（OC1PE 位）中未选择预装载功能，写入的数值会永久加载。否则只有当更新事件发生时，此预装载值才加载至内部活动的捕获/比较 1 寄存器中。 活动的捕获/比较寄存器与计数器 TIMx_CNT 进行比较，并在 OC1 端口上产生输出信号。 **若 CC1 通道配置为输入：** CCR1 是上一次输入捕获 1 事件（IC1）捕获的计数器值。

14.4.15 TIM2 和 TIM3 捕获/比较寄存器 2 (TIM2_CCR2 和 TIM3_CCR2)

TIM2_CCR2 / TIM3_CCR2			地址偏移	0x38		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bi25	Bit24
控制位	CCR2[31:16](TIM2 only)							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	CCR2[23:16](TIM2 only)							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CCR2[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CCR2[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	CCR2[31:16]	捕获/比较通道 2 的值高 16 位（仅 TIM2 支持）。
BIT[15:0]	CCR2[15:0]	捕获/比较通道 2 的值 **若 CC2 通道配置为输出：** CCR2 是要加载到实际捕获/比较 2 寄存器的值（预装载值）。 如果在 TIMx_CCMR2 寄存器（OC2PE 位）中未选择预装载功能，写入的数值会永久加载。否则只有当更新事件发生时，此预装载值才加载至内部活动的捕获/比较 2

		寄存器中。 活动的捕获/比较寄存器与计数器 TIMx_CNT 进行比较，并在 OC2 端口上产生输出信号。 **若 CC2 通道配置为输入：** CCR2 是上一次输入捕获 2 事件 (IC2) 捕获的计数器值。

14.4.16 TIM2 和 TIM3 捕捉 / 比较寄存器 3 (TIM2_CCR3 和 TIM3_CCR3)

TIM2_CCR3 / TIM3_CCR3			地址偏移	0x3C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bi25	Bit24
控制位	CCR3[31:16](TIM2 only)							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	CCR3[23:16](TIM2 only)							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CCR3[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CCR3[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	CCR3[31:16]	捕获/比较通道 3 的值高 16 位（仅 TIM2 支持）。
BIT[15:0]	CCR3[15:0]	捕获/比较通道 3 的值 **若 CC3 通道配置为输出：** CCR3 是要加载到实际捕获/比较 3 寄存器的值 （ 预装载值 ）。 如果在 TIMx_CCMR3 寄存器 (OC3PE 位) 中未选择预装载功能，写入的数值会永久加载。否则只有当更新事件发生时，此预装载值才加载至内部活动的捕获/比较 3 寄存器中。 活动的捕获/比较寄存器与计数器 TIMx_CNT 进行比较，并在 OC3 端口上产生输出信号。 **若 CC3 通道配置为输入：** CCR3 是上一次输入捕获 3 事件 (IC3) 捕获的计数器值。

14.4.17 TIM2 和 TIM3 捕捉 / 比较寄存器 4 (TIM2_CCR4 和 TIM3_CCR4)

TIM2_CCR4 / TIM3_CCR4			地址偏移	0x40		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	CCR4[31:16](TIM2 only)							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	CCR4[23:16](TIM2 only)							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CCR4[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CCR4[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:16]	CCR4[31:16]	捕获/比较通道 4 的值高 16 位（仅 TIM2 支持）。
BIT[15:0]	CCR4[15:0]	捕获/比较通道 4 的值 **若 CC4 通道配置为输出：** CCR4 是要加载到实际捕获/比较 4 寄存器的值 （ 预装载值 ）。 如果在 TIMx_CCMR4 寄存器 (OC4PE 位) 中未选择预装载功能，写入的数值会永久加载。否则只有当更新事件发生时，此预装载值才加载至内部活动的捕获/比较 4 寄存器中。 活动的捕获/比较寄存器与计数器 TIMx_CNT 进行比较，并在 OC4 端口上产生输出信号。 **若 CC4 通道配置为输入：** CCR4 是上一次输入捕获 4 事件 (IC4) 捕获的计数器值。

14.4.18 TIM2 和 TIM3 DMA 控制寄存器 (TIM2_DCR 和 TIM3_DCR)

TIM2_DCR / TIM3_DCR			地址偏移	0x48		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	DBL[4:0]				
R/W	-	-	-	rw	rw	rw	rw	rw
复位值	-	-	-	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	DBA[4:0]				
R/W	-	-	-	rw	rw	rw	rw	rw
复位值	-	-	-	0	0	0	0	0

BIT[15:13]	-	保留，保持复位值
BIT[12:8]	DBL[4:0]	DMA 突发传送长度 (DMA burst length) 这 5 位定义了 DMA 传送长度 （ 当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次突发传送 ）， 00000：1 次传输 00001：2 次传输 00010：3 次传输 ..... 10001：18 次传输
BIT[7:5]	-	保留，保持复位值
BIT[4:0]	DBA[4:0]	DMA 基地址 (DMA base address) 这 5 位定义了 DMA 传送的基地址 （ 当对 TIMx_DMAR 寄存器进行读或写时 ），DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 例如： 00000：TIMx_CR1 00001：TIMx_CR2 00010：TIMx_SMCR ..... 例：要完成如下的传输：DBL = 7 ，DBA = TIMx_CR1 此时传送从 TIMx_CR1 的地址开始，发送/读取连续 7 个寄存器。

14.4.19 TIM2 和 TIM3 DMA 完全传送地址寄存器 (TIM2_DMAR 和 TIM3_DMAR)

TIM2_DMAR / TIM3_DMAR			地址偏移	0x4C		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	DMAB[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	DMAB[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	DMAB[15:0]	DMA 突发传送寄存器 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的访问： （TIMx_CR1 地址） +（DBA + DMA 索引）x 4，

		其中： “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1) 所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的基地址； “DMA 索引”是由 DMA 自动控制的偏移量（0~DBL），(DBL 在 TIMx_DCR 寄存器中定义)。

举一个如何使用 DMA 突发操作的例子

本例中使用定时器 DMA 的突发功能，将 CCRx 寄存器 (x = 2, 3, 4) 的内容以半字方式进行 DMA 传输，更新到 CCRx 寄存器。按如下步骤进行操作：

1、 配置相关的 DMA 通道 :

- DMA 通道外设地址为 DMAR 寄存器地址
- DMA 通道存储器地址为 RAM 缓冲区地址，包含要通过 DMA 传送到 CCRx 寄存器的数据
- 传送数据数量 = 3 (见下面的注释)
- 循环模式禁用

2、 配置 DCR 寄存器的 DBA 和 DBL 位: DBL = 3 次传送 , DBA = 0xE。

3、 使能 TIMx 更新 DMA 请求 (置位 DIER 寄存器的 UDE 位)。

4、 使能 TIMx

5、 使能 DMA 通道

注：在本例中所 CCRx 寄存器被一次性全部更新。如果需要更新 CCRx 寄存器两次，传送的数据数量应该是 6。假定，RAM 缓冲区要包含 data1, data2, data3, data4, data5 和 data6。数据按如下过程传送到 CCRx 寄存器：在第一个更新 DMA 请求时，data1 被传送到 CCR2，data2 被传送到 CCR3，data3 被传送到 CCR4；在第二个 DMA 更新中断请求时，data4 被传送到 CCR2，data5 被传送到 CCR3，data6 被传送到 CCR4。

15 通用定时器（TIM14）

15.1 概述

该定时器基于一个 16 位自动重载递增计数器和一个 16 位预分频器。
TIM14 具有一个单通道，用于输入捕获/输出比较，PWM 或单脉冲模式输出。
在调试模式下，其计数器可被冻结。
注：TIM14 不支持触发 DMA 请求

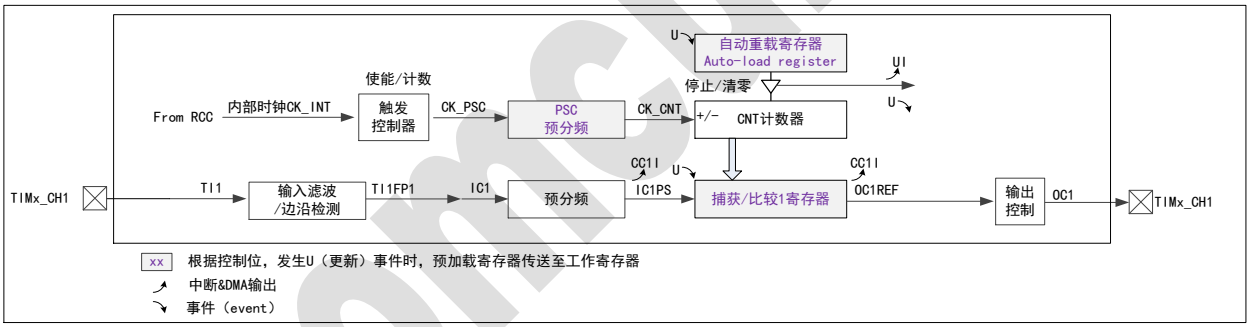
15.2 特性

✧ 16 位自动装载计数器
✧ 16 位可编程 （ 可实时修改 ） 预分频器，支持 1～65535 之间的任意分频
✧ 1 个独立通道，支持
- 输入捕获
- 输出比较
- PWM 生成（边沿对齐）

✧ 下述事件触发中断：
- 更新（Update）：计数器上溢，计数器初始化（通过软件）
- 输入捕获（input capture）
- 输出比较（output compare）

15.3 功能描述

TIM14 模块框图



15.3.1 时基单元

通用定时器 TIM14 主要由 16 位计数器及相关自动重载寄存器构成，计数器支持向上计数。计数器时钟支持预分频。计数器寄存器、自动重载寄存器和预分频寄存器支持软件读写，即使计数器正在运行读写仍然有效。
时基单元包括：
✧ 计数器寄存器（TIMx_CNT）
✧ 预分频器寄存器（TIMx_PSC）
✧ 自动重载寄存器（TIMx_ARR）

自动重载寄存器是预装载的，读写自动重载寄存器将访问预装载寄存器。设置 TIMx_CR1 寄存器中的自动重载预装载使能位（ARPE），选择预装载寄存器的内容永久传送至缓存寄存器或在每次更新事件（UEV）传送至缓存寄存器。当计数器达到溢出条件（ 或向下计数时的下溢条件） 并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可由软件产生。有关更新事件的产生，针对每种配置后续章节会详细描述。

计数器时钟由预分频后输出 CK_CNT 驱动，需置位 TIMx_CR1 寄存器中的计数器使能位（CEN）时，CK_CNT 才有效（请参阅从模式控制器描述以获得计数器使能的更多细节）。

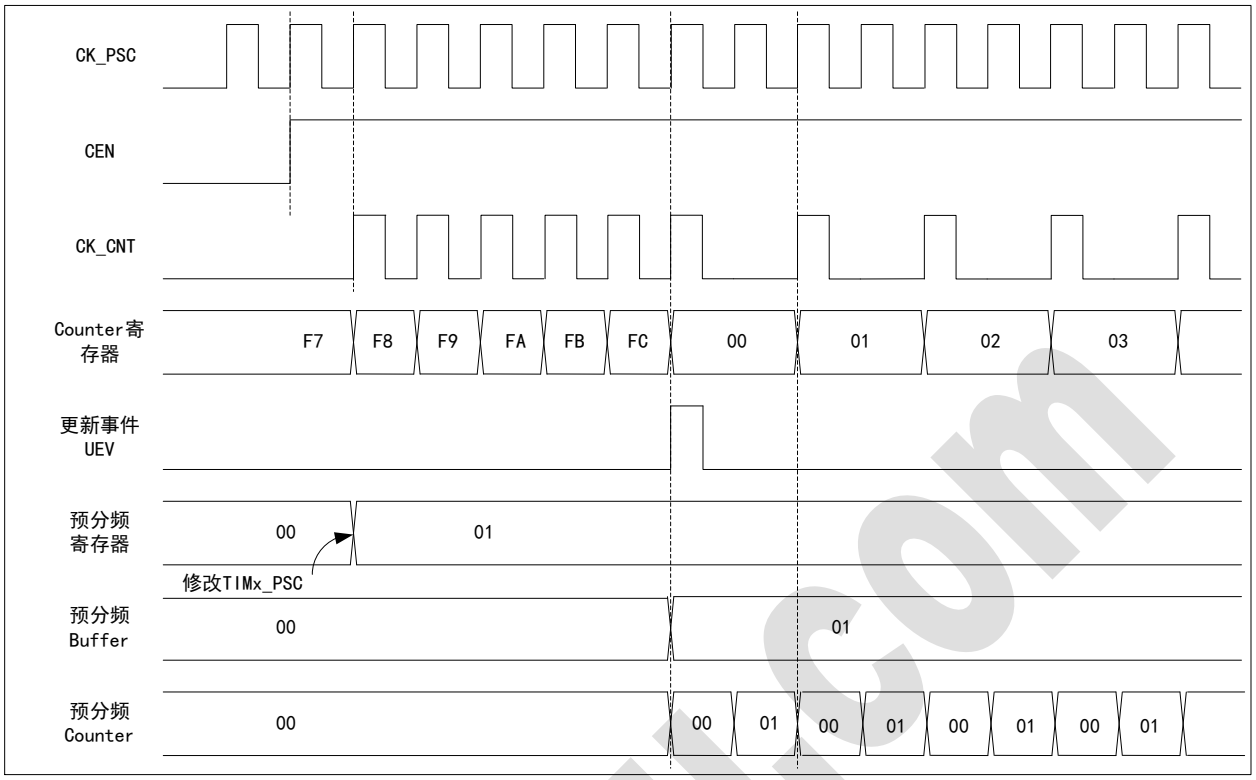
注：设置 TIMx_CR 寄存器的 CEN 位，一个时钟周期后，计数器才开始计数。

预分频

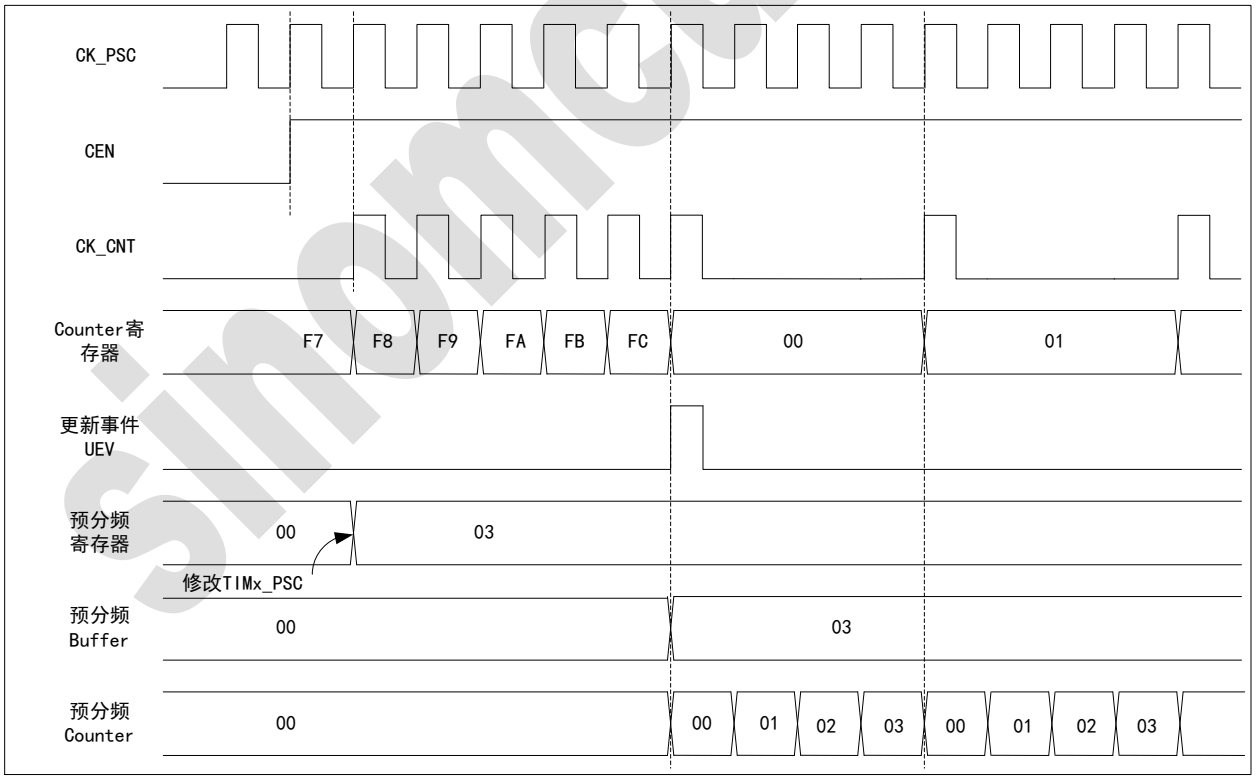
预分频器支持计数器时钟 1~65536 之间任意分频，基于 1 个 16 位的计数器，由 TIMx_PSC 寄存器中的 16 位寄存器控制。此寄存器内部带有缓存器，支持运行时修改；新修改的预分频值在下一次的更新事件（update event）发生时生效。

下图给出运行时更改预分频值，计数器行为。

计数器时序图：预分频修改，从 1 到 2



计数器时序图：预分频修改，从 1 到 4



15.3.2 计数器模式

向上计数模式

设置 TIMx_CR1 寄存器中的 CMS=00 且 DIR=0，执行计数器向上计数。

在向上计数模式中，计数器从 0 计数到自动重载值（TIMx_ARR 计数器的值 ），然后重新从 0 开始计数并且产生一个计数器溢出事件（overflow）。

每次计数器溢出时都会产生更新事件。

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

通过软件置位 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新缓存寄存器。在 UDIS 位被清‘0’之前，将不产生更新事件。但是，在应该产生更新事件时，计数器会被清‘0’，同时预分频器的计数器也被清 0(但预分频器的数值不变)。

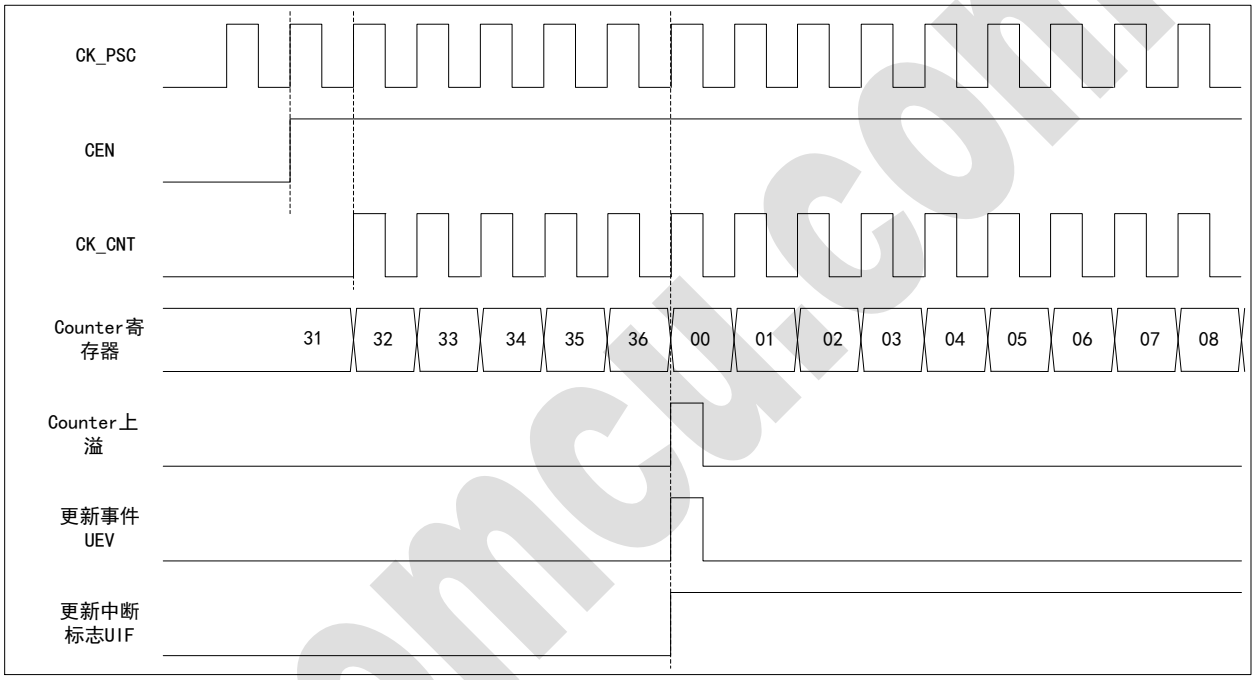
此外，如果置位 TIMx_CR1 寄存器中的 URS 位（update request selection），置位 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志（即不产生中断）。这是为了避免在发生捕获事件（capture event）清除计数器时，同时产生更新和捕获两个中断。

当发生一个更新事件时，所有的寄存器都被更新，同时（依据 URS 位）置位更新标志位（TIMx_SR 寄存器中的 UIF 位）：

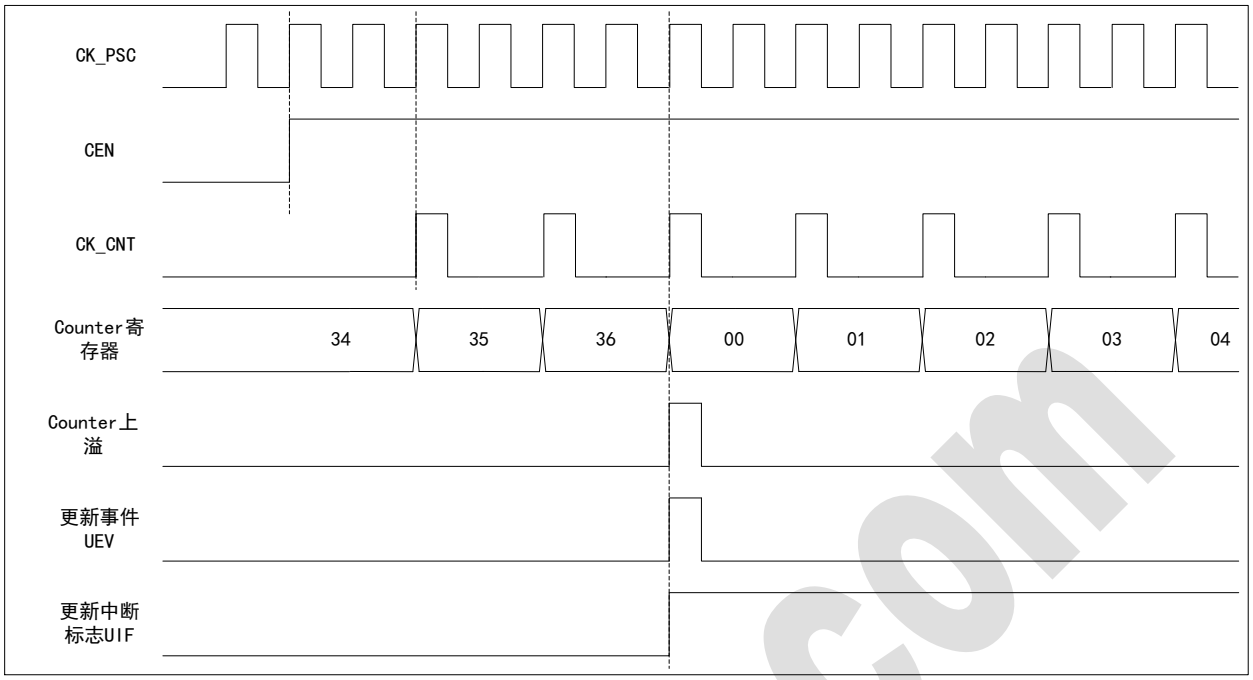
- ✧ 预分频器的缓冲区被写入预装载寄存器的值（TIMx_PSC 寄存器的值 ）
- ✧ 自动重载缓存寄存器更新为预装载寄存器的值（TIMx_ARR）

举例如下：TIMx_ARR=0x36 时，计数器在不同时钟频率下计数器行为。

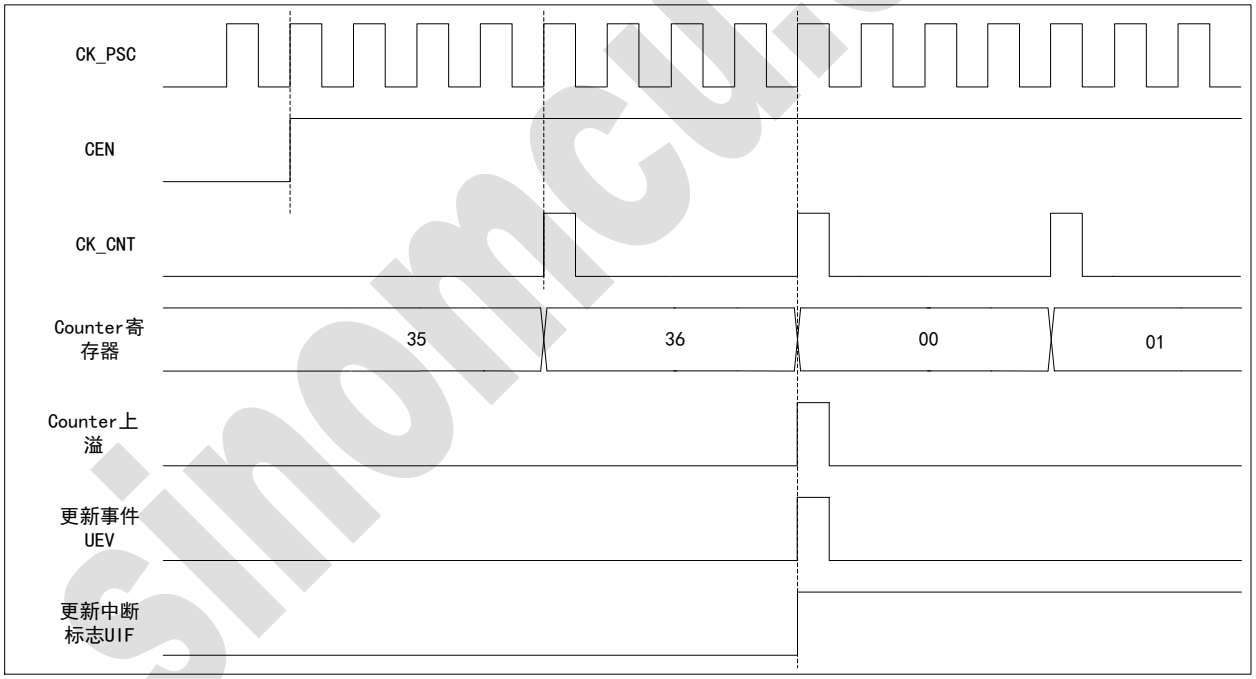
计数器时序图：预分频因子=1



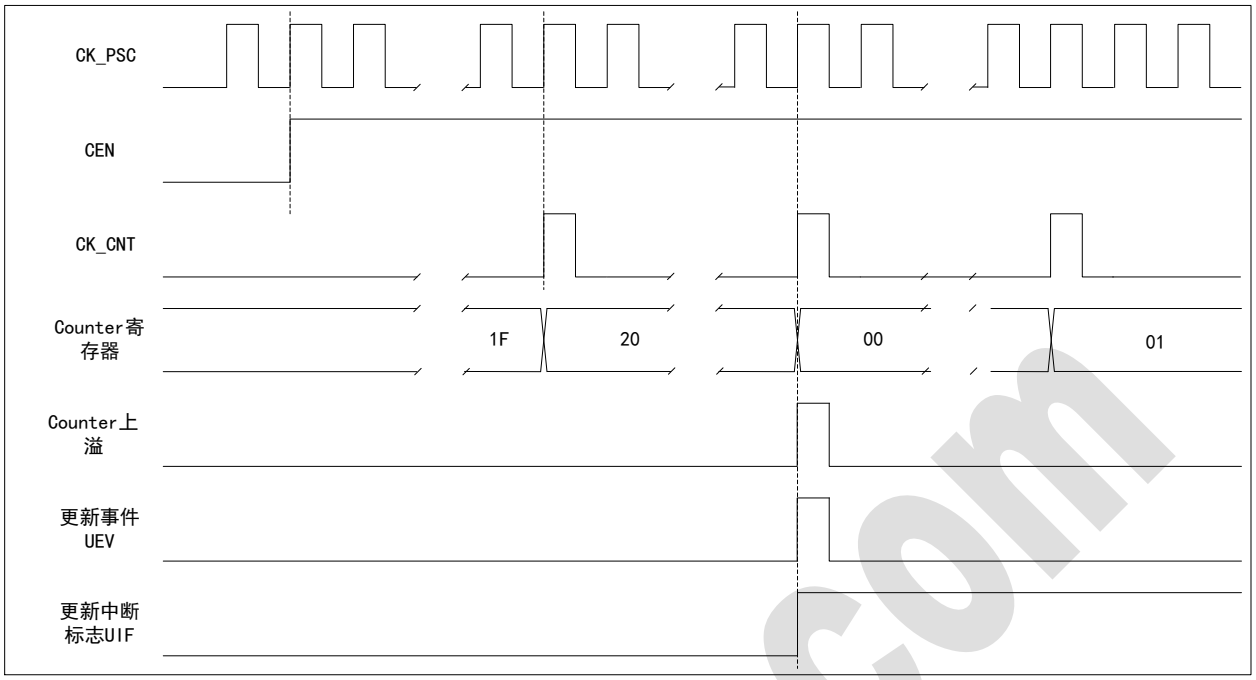
计数器时序图：预分频因子=2



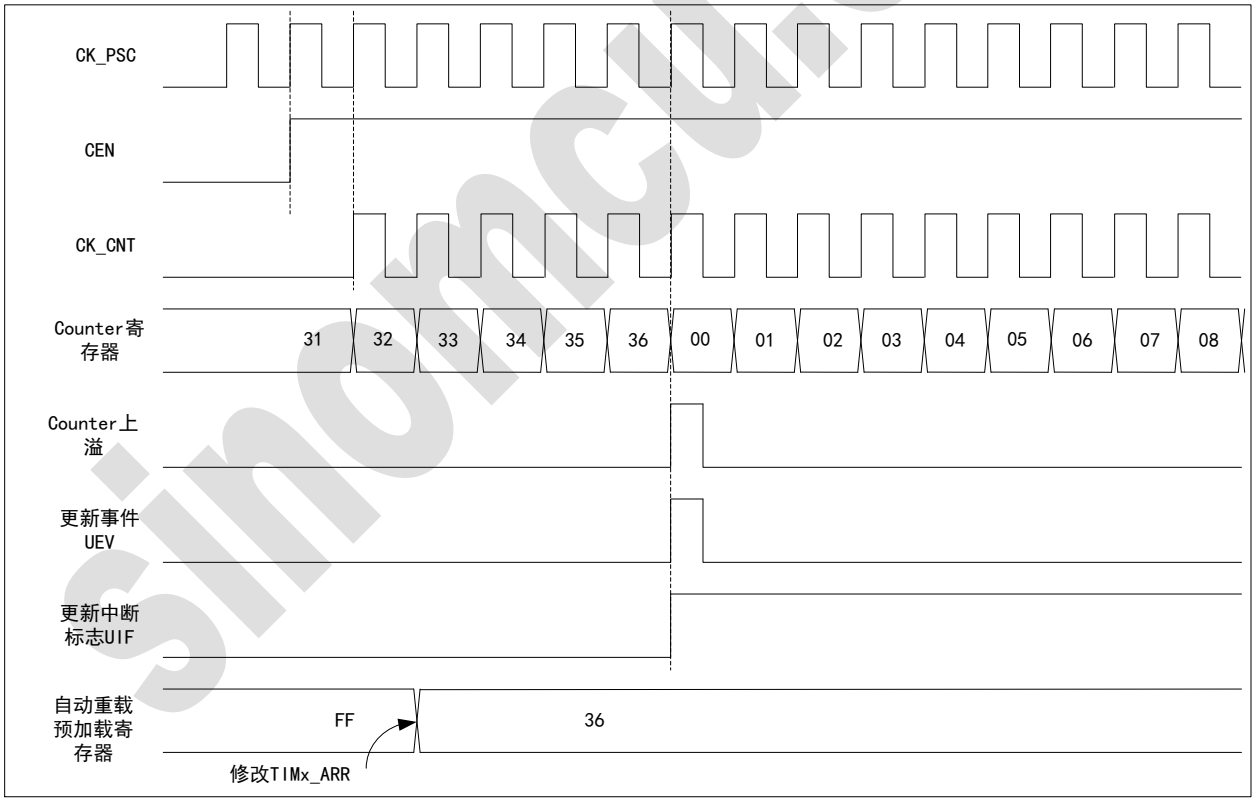
计数器时序图：预分频因子=4



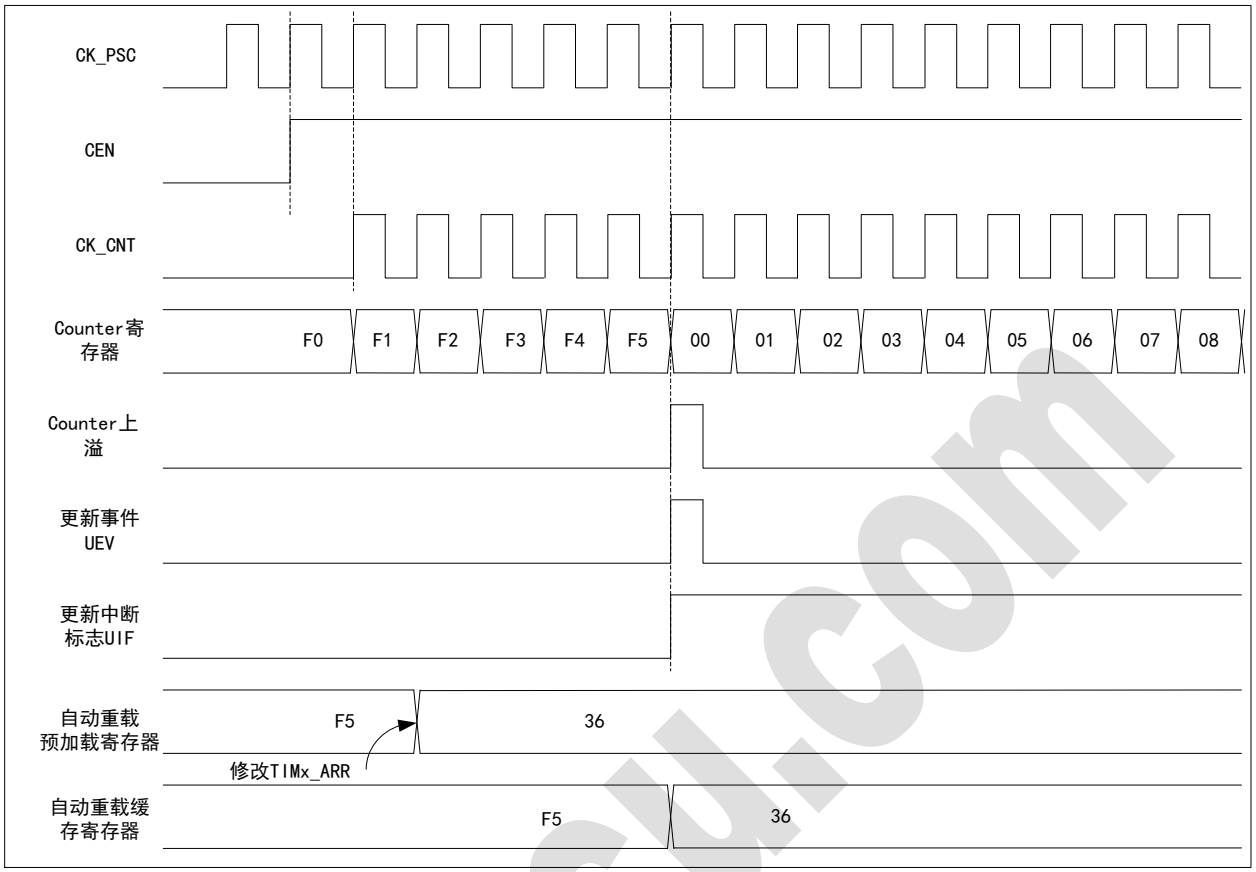
计数器时序图：预分频因子=N



计数器时序图：预分频因子=1，ARPE=0（TIMx_ARR 没有预装载）



计数器时序图：预分频因子=1，ARPE=1（TIMx_ARR 已预装载）



15.3.3 时钟源

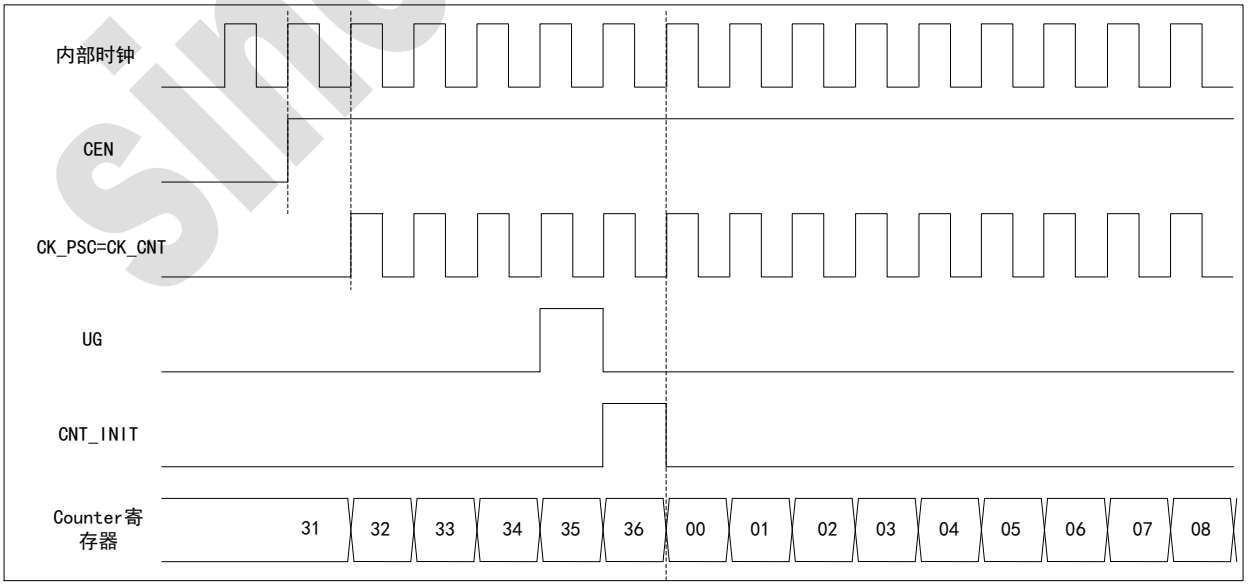
计数器时钟源包括：

✧ 内部时钟 CK_INT

内部时钟源（CK_INT）

CEN（TIMx_CR1 寄存器）和 UG 位（TIMx_ EGR 寄存器）为实际控制位，并且只能被软件修改（ 除了 UG 位保持自动被清除）。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。

计数器时序图：内部时钟，预分频因子=1，正常模式



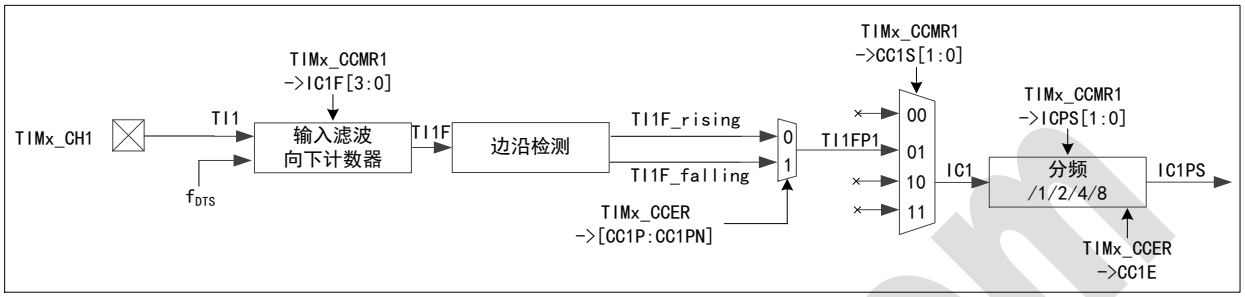
15.3.4 捕获/比较通道

每一个捕获/比较通道都是包括：一个捕获/比较寄存器（包含缓存寄存器），一个用于捕获的输入级（带数字滤波、多路复用选择和预分频器），和一个输出级（带比较器和输出控制）。

输入级

对 TIx 的输入信号进行采样，产生一个滤波后 TIxF 信号，经过带有极性选择的边沿检测器生成 TIxFPx 信号，TIxFPx 可以作为捕获命令。该信号预分频后（ICxPS）进入捕获寄存器。

捕获/比较通道输入级（举例：通道 1）



输出级

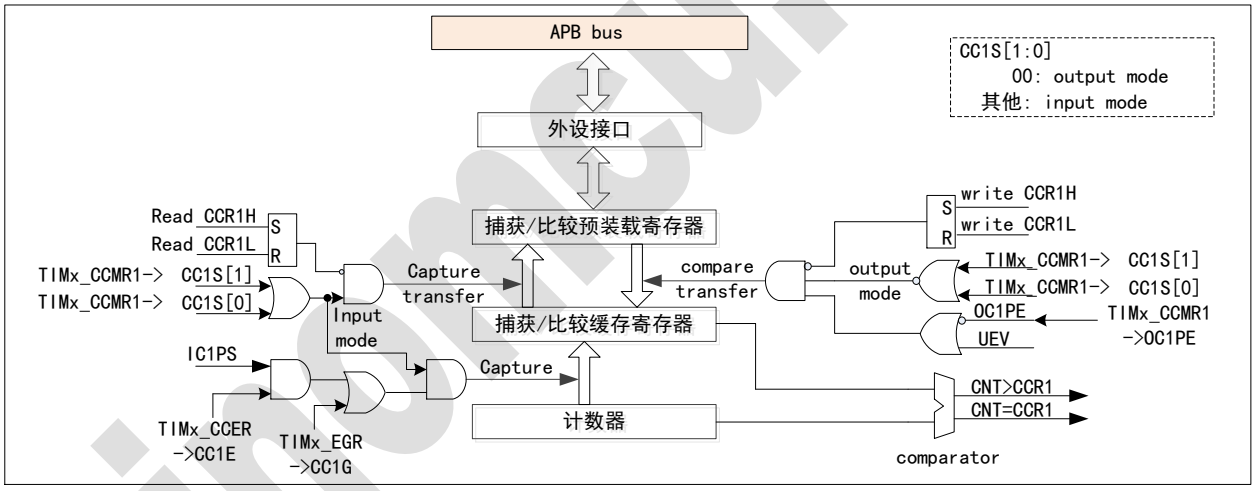
输出部分产生一个中间波形 OCxRef（ 高有效）作为基准，链的末端决定最终输出信号的极性。

捕获/比较模块由一个预装载寄存器（preload register）和一个缓存寄存器（shadow register）组成。读写过程仅操作预装载寄存器。

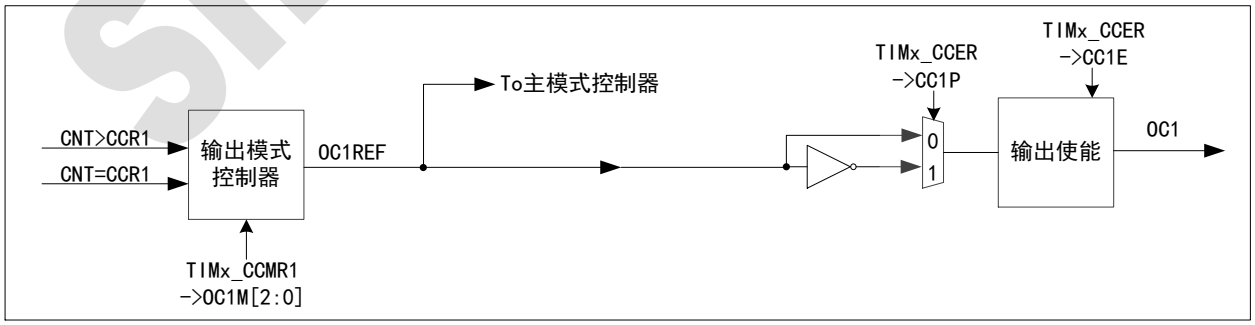
在捕获模式下，捕获发生在缓存寄存器上，然后再复制到预装载寄存器中。

在比较模式下，预装载寄存器的内容被复制到缓存寄存器中，然后缓存寄存器的内容和计数器进行比较。

捕获/比较通道主电路



捕获/比较通道输出级（通道 1）



15.3.5 输入捕获模式

在输入捕获模式下，当检测 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（TIMx_CCRx）中。当发生捕获事件时，相应 CCxIF 标志（TIMx_SR 寄存器）被置 1， 如果使能了中断，则产生一个中断请求。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（TIMx_SR 寄存器）被置 1。软件写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

举例，在 TI1 输入的上升沿时，捕获计数器的值到 TIMx_CCR1 寄存器中，操作步骤如下：

1、 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01（选中 TI1），只要 CC1S 不为‘00’，通道被配置为输入并且 TIMx_CCR1 寄存器变为只读。

2、 根据连接到计数器的输入信号，配置输入滤波器时间（输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位）。举例，当 TI1 翻转时，信号抖动最多 5 个内部时钟， 则须配置滤波器时间大于 5 个时钟周期，TIMx_CCMR1 寄存器中写入 IC1F=0011 配置采样次数 8（以 f_{DTS} 频率采样），通过连续 8 次采样以确认电平变换。

3、 选择 TI1 通道有效沿。写 TIMx_CCER 寄存器中 CC1P=0 和 CC1NP=0（本例中为上升沿）。

4、 配置输入预分频器。在本例中，设置为每个有效的电平转换时发生一次捕获，因此预分频器被禁止（写 TIMx_CCMR1 寄存器的 IC1PS=00）。

5、 写 TIMx_CCER 寄存器的 CC1E=1，允许计数器的值被捕获至捕获寄存器中。

6、 根据需要，置位 TIMx_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：

✧ 有效的电平转换发生时，计数器的值被传送至 TIMx_CCR1 寄存器；

✧ CC1IF 标志被置位（中断标志）。当发生至少 2 个连续的捕获时，且 CC1IF 未被清除，CC1OF 也被置位；

✧ 若置位 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，这是为了避免在读取标志之后和读取数据之前可能发生的捕获溢出。

注： 通过软件设置 TIMx_EGR 寄存器中相应的 CCxG 位，也可以产生输入捕获中断请求。

15.3.6 强制输出模式

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

设置 TIMx_CCMRx 寄存器中相应的 OCxM=101，输出比较信号（OCxREF/OCx）强制为有效状态。这样 OCxREF 被强制高电平(OCxREF 始终为高电平有效)，置位 TIMx_CCER 的 CCxP 位，OCx 可得到极性相反的信号。

举例，CCxP=0（OCx 高电平有效），则 OCx 被强制为高电平。

设置 TIMx_CCMRx 寄存器中相应的 OCxM=100，输出比较信号（OCxREF/OCx）强制为低电平。

该模式下，在 TIMx_CCRx 缓存寄存器和计数器之间的比较仍然在执行，相应的标志也会被置位。并会产生相应的中断请求。下面的输出比较模式一节中将详细介绍。

15.3.7 输出比较模式

此模式用于控制输出波形或指示一段时间到时。

当计数器与捕获/比较寄存器的内容匹配（相等）时，输出比较功能做如下操作：

✧ 根据输出比较模式（TIMx_CCMRx 寄存器中 OCxM 位）和输出极性（TIMx_CCER 寄存器中的 CCxP 位）的配置，输出至对应引脚。在比较匹配发生时，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）。

✧ 置位中断状态寄存器中的标志位（TIMx_SR 寄存器中的 CCxIF 位）。

✧ 若置位相应的中断屏蔽位（TIMx_DIER 寄存器中的 CCxIE 位），则产生一个中断。

设置 TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。计时分辨率为计数器的一次计数。

输出比较模式也能用来输出一个单脉冲（单脉冲模式）。

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、 根据需求，写入数据至 TIMx_ARR 和 TIMx_CCRx 寄存器中。

3、 如果要产生一个中断请求，设置 CCxIE 位。

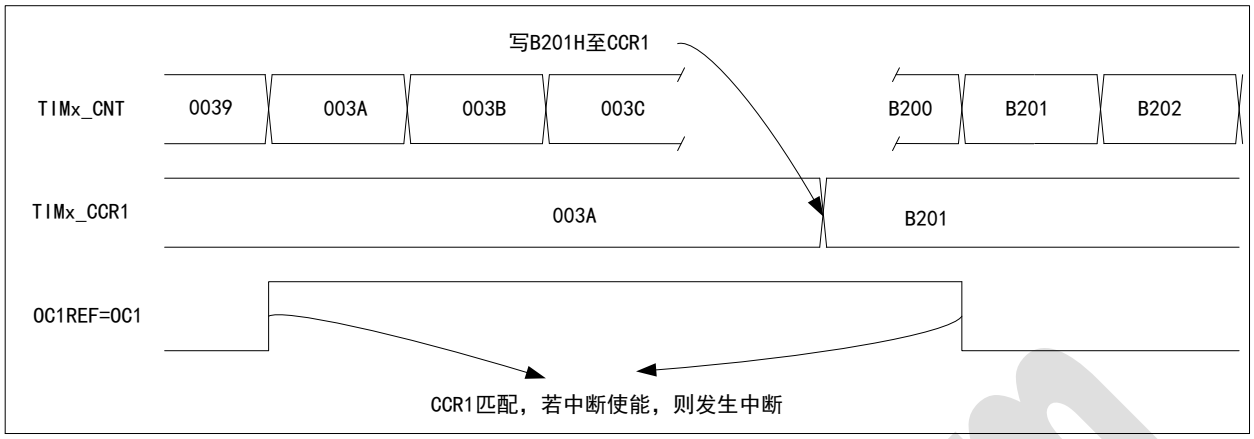
4、 选择输出模式，举例如下：

- 写 OCxM=011，计数器与 CCRx 匹配时，翻转 OCx 的输出引脚；
- 写 OCxPE=0，禁用预装载寄存器；
- 写 CCxP=0，选择极性为高电平有效；
- 写 CCxE=1，使能输出。

5、 置位 TIMx_CR1 寄存器的 CEN 位，启动计数器。

TIMx_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，前提是未启用预装载寄存器（OCxPE=‘0’，否则 TIMx_CCRx 的缓存寄存器只在发生下次更新事件 EUV 时更新）。举例如下图。

输出比较模式，配置为：翻转 OC1



15.3.8 PWM 模式

脉冲宽度调制模式（PWM）可以产生一个 PWM 信号，由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比。

在 TIMx_CCMRx 寄存器中的 OCxM 位写入‘110’（PWM 模式 1）或‘111’（PWM 模式 2），可以在每个通道独立地设置 PWM 模式（每个 OCx 输出一种 PWM）。必须置位 TIMx_CCMRx 寄存器的 OCxPE 位，使能相应的预装载寄存器（preload register），最后置位 TIMx_CR1 寄存器的 ARPE 位，使能自动重载预装载寄存器（在向上计数模式中）。

只有发生一个更新事件时，预装载寄存器才能被传送到缓存寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。

设置 TIMx_CCER 寄存器中的 CCxP 位，配置 OCx 的极性为高电平有效或低电平有效。OCx 的输出使能通过（TIMx_CCER 寄存器中）CCxE 位控制。详细参考 TIMx_CCER 寄存器的描述。

在 PWM 模式（模式 1 或模式 2）下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，以确定是否符合 TIMx_CNT ≤ TIMx_CCRx。

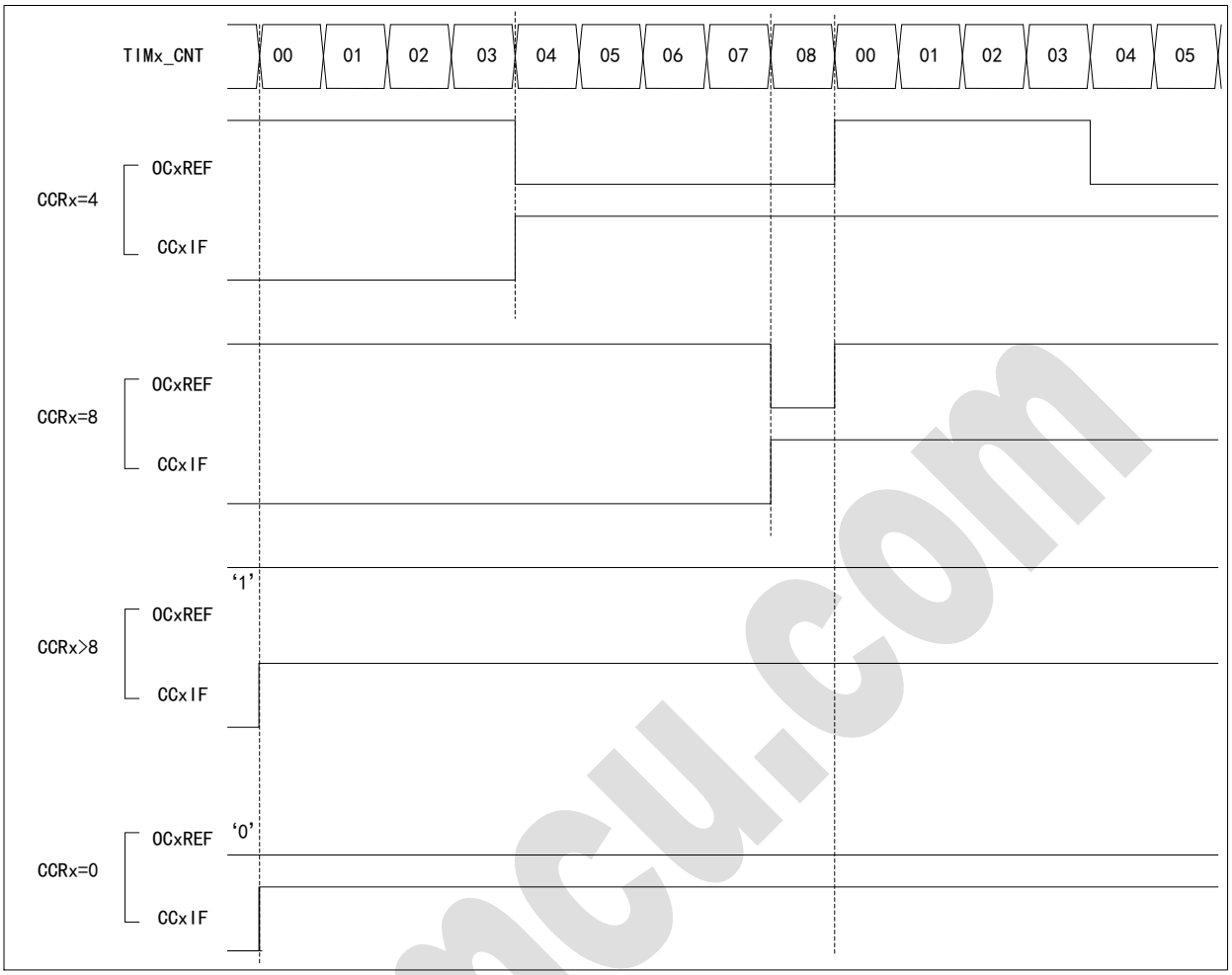
由于此计数器为向上计数，仅支持边沿对齐模式的 PWM 信号。

PWM 边沿对齐模式

举例，PWM 模式 1

当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重载值（TIMx_ARR），则 OCxREF 保持为‘1’。如果比较值为 0，则 OCxREF 保持为‘0’。 下图为 TIMx_ARR=8 时边沿对齐的 PWM 波形实例。

PWM 波形（边沿对齐，向上计数，ARR=8）



15.3.9 调试模式

当微控制器进入调试模式时(Cortex-M0 内核停止)，根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以或者继续正常操作，或者停止。

15.4 相关寄存器

15.4.1 寄存器汇总表

基址：0x4001 2000				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	TIM14_CR1	0x0000 0000	TIM14 控制寄存器 1	15.4.2
0x04~0x08	Res.	-	-	
0x0C	TIM14_DIER	0x0000 0000	TIM14 中断使能寄存器	15.4.3
0x10	TIM14_SR	0x0000 0000	TIM14 状态寄存器	15.4.4
0x14	TIM14_EGR	0x0000 0000	TIM14 事件产生寄存器	15.4.5
0x18	TIM14_CCMR1	0x0000 0000	TIM14 捕获/比较模式寄存器 1	15.4.6
0x1C	Res.	-	-	
0x20	TIM14_CCER	0x0000 0000	TIM14 捕获/比较使能寄存器	15.4.7
0x24	TIM14_CNT	0x0000 0000	TIM14 计数器	15.4.8
0x28	TIM14_PSC	0x0000 0000	TIM14 预分频器	15.4.9
0x2C	TIM14_ARR	0xFFFF FFFF	TIM14 自动重载寄存器	15.4.10
0x30	Res.	-	-	
0x34	TIM14_CCR1	0x0000 0000	TIM14 捕获/比较寄存器 1	15.4.11
0x38~0x4C	Res.	-	-	
0x50	TIM14_OR	0x0000 0000	TIM14 选项寄存器	15.4.12

15.4.2 TIM14 控制寄存器 1（TIM14_CR1）

TIM14_CR1			地址偏移	0x00		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	CKD[1:0]	
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	ARPE	-	-	-	-	URS	UDIS	CEN
R/W	rw	-	-	-	-	rw	rw	rw
复位值	0	-	-	-	-	0	0	0

BIT[15:10]	-	保留，保持复位值
BIT[9:8]	CKD[1:0]	时钟分频因子（Clock division） 这 2 位定义在定时器时钟 (CK_INT) 频率与数字滤波器(ETR,TIx) 所用的采样时钟之间的分频比例。 00：tDTS = tCK_INT 01：tDTS = 2 x tCK_INT 10：tDTS = 4 x tCK_INT 11：保留，不要使用这个配置
BIT[7]	ARPE	自动重载预装载允许位（Auto-reload preload enable） 0: TIMx_ARR 寄存器没有缓冲 1: TIMx_ARR 寄存器有缓冲
BIT[6:3]	-	保留，保持复位值
BIT[2]	URS	更新请求源 (Update request source)，软件置位和清零 软件通过该位选择中断事件（UEV）源 0：如果使能，则下述任一事件产生 UEV： - 计数器上溢 - 设置 UG 位 1：如果使能，只有计数器上溢产生 UEV 事件。
BIT[1]	UDIS	禁止更新 (Update disable)，软件置位和清零 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 - 设置 UG 位 所有缓存寄存器被装入它们的预装载值。 1：禁止 UEV。不产生更新事件 UEV，缓存寄存器 (ARR、PSC、CCRx) 保持它们的值。如果置位 UG 位则计数器和预分频器被重新初始化。
BIT[0]	CEN	使能计数器 (Counter enable) 0：禁止计数器 1：使能计数器

15.4.3 TIM14 中断使能寄存器（TIM14_DIER）

TIM14_DIER			地址偏移	0x0C		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	CC1IE	UIE
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	0	0

BIT[15:2]	-	保留，保持复位值
BIT[1]	CCIE	捕获/比较 1 中断使能 0: CC1 中断禁止

		1：CC1 中断允许
BIT[0]	UIE	更新中断使能 0：更新中断禁止 1：更新中断允许

15.4.4 TIM14 状态寄存器 (TIM14_SR)

TIM14_SR			地址偏移	0x10		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	CC1OF	-
R/W	-	-	-	-	-	-	rc_w0	-
复位值	-	-	-	-	-	-	0	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	CC1IF	UIF
R/W	-	-	-	-	-	-	rc_w0	rc_w0
复位值	-	-	-	-	-	-	0	0

BIT[15:10]	-	保留，保持复位值
BIT[9]	CC1OF	捕获/比较 1 重复捕获标志 (Capture/Compare1 overcapture flag) 仅当相应的通道被配置为输入捕获时，由硬件置‘1’。软件写‘0’清零。 0：无重复捕获产生； 1：当计数器的值被捕获到 TIMx_CCR1 寄存器时，CC1IF 的状态已经为‘1’。
BIT[8:2]	-	保留，保持复位值
BIT[1]	CC1IF	CC1IF：捕获/比较 1 中断标志 (Capture/Compare 1 interrupt flag) **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’，由软件清‘0’。 0：无匹配发生； 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的值大于 TIMx_ARR 时，在计数器上溢时 CC1IF 位变高 **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’，它由软件清’0’或通过读 TIMx_CCR1 清’0’。 0：无输入捕获产生； 1：计数器值已被捕获（拷贝）至 TIMx_CCR1（在 IC1 上检测到与所选极性相同的边沿）。
BIT[0]	UIF	更新中断标志 (Update interrupt flag) 当产生更新事件时该位由硬件置‘1’。它由软件清‘0’。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置‘1’： - 若 TIMx_CR1 寄存器的 UDIS=0，当计数溢出时； - 若 TIMx_CR1 寄存器的 UDIS=0、URS=0，通过软件写 TIMx_EGR 寄存器的 UG 位重初始化定时器时。

15.4.5 TIM14 事件产生寄存器 (TIM14_EGR)

TIM14_EGR			地址偏移	0x14		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	CC1G	UG
R/W	-	-	-	-	-	-	w	w
复位值	-	-	-	-	-	-	0	0

BIT[15:2]	-	保留，保持复位值
BIT[1]	CC1G	捕获/比较 1 产生 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：** CC1IF 被置位，若开启中断使能则产生相应的中断 **若通道 CC1 配置为输入：** 当前的计数器值捕获至 TIMx_CCR1 寄存器；CC1IF 被置位，若开启中断使能则产生相应的中断。若 CC1IF 已经为‘1’，则 CC1OF 被置位。
BIT[0]	UG	产生更新事件（Update generation） 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 ）。计数器被清‘0’。

15.4.6 TIM14 捕获/比较模式寄存器 1（TIM14_CCMR1）

此寄存器在通道配置为输入（捕获模式）或输出（比较模式），具有不同的功能。由 CCxS 位配置输入/输出，下面的控制位，OCxx 表示输出相关的控制位。ICxx 表示输入相关的控制位。

TIM14_CCMR1			地址偏移	0x18		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
	IC1F[3:0]				IC1PSC[1:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

输出比较模式

BIT[15:7]	-	保留，保持复位值
BIT[6:4]	OC1M	输出比较 1 模式（Output compare 1 mode） 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1 值。OC1REF 是高电平有效，而 OC1 的有效电平取决于 CC1P 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_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（TIMx_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 TIMx_CNT 的值与捕获/比较寄存器 1（TIMx_CCR1）相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_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 1，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平； 111：PWM 模式 2，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平； *注：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。*
BIT[3]	OC1PE	输出比较 1 预装载使能（Output compare 1 preload enable） 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即生效。 1： 开启 TIMx_CCR1 寄存器的预装载功能， 读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被传送至当前寄存器中。
BIT[2]	OC1FE	输出比较 1 快速使能（Output compare 1 fast enable） 该位用于加快 CC 输出对触发器输入事件的响应。 0：根据计数器与 CCR1 的值，CC1 正常操作，即使触发器是打开的。当触发器的输入出现一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1：输入到触发器的有效沿的作用就象发生了一次比较匹配。因此，OC 被设置为

		比较电平而与比较结果无关。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。该位只在通道被配置成 PWM1 或 PWM2 模式时起作用。
BIT[1:0]	CC1S	捕获/比较 1 选择 (Capture/Compare 1 selection) 这 2 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：保留 11：保留 *注： CC1S 仅在通道关闭时（TIMx_CCER 寄存器的 CC1E=‘ 0 ’）才是可写的。*

输入捕获模式

BIT[15:8]	-	保留，保持复位值
BIT[7:4]	IC1F	输入捕获 1 滤波器 (Input capture 1 filter) 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 f_{DTS} 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：当 ICxF[3:0]=1、2 或 3 时，公式中的 fDTS 由 CK_INT 替代。*
BIT[3:2]	IC1PSC	输入捕获 1 预分频（Input capture 1 prescaler） 这 2 位定义了 CC1 输入 (IC1) 的预分频系数。 一旦 CC1E=‘0’（TIMx_CCER 寄存器中 ），则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
BIT[1:0]	CC1S	捕获/比较 1 选择（Capture/Compare 1 selection） 这 2 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：保留 11：保留 *注：CC1S 仅在通道关闭时（TIMx_CCER 寄存器的 CC1E=‘ 0 ’）才是可写的。*

15.4.7 TIM14 捕获/比较使能寄存器 (TIM14_CCER)

TIM14_CCER			地址偏移	0x20		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	CC1NP	-	CC1P	CC1E
R/W	-	-	-	-	rw	-	rw	rw
复位值	-	-	-	-	0	-	0	0

BIT[15:4]	-	保留，保持复位值
BIT[3]	CC1NP	捕获/比较 1 输出极性 (Capture/Compare 1 complementary output Polarity) 通道 CC1 配置为输出时，CC1NP 必须操保持为清零，CC1NP＝0; 通道 CC1 配置为输入时，CC1NP 与 CC1P 联合控制 TI1FP1 的极性，参考 CC1P 描述。
BIT[2]	-	保留，保持复位值
BIT[1]	CC1P	捕获/比较 1 输出极性 .(Capture/Compare 1 output Polarity) **通道 CC1 配置为输出 ：** 0：OC1 高有效 1：OC1 低有效 **通道 CC1 配置为输入 ：** CC1P/CC1NP 用于选择作为触发或捕获的信号 TI1FP1 的极性。 00：不反相/上升沿：捕获发生在 TIxFP1 的上升沿（捕捉模式），TIxFP1 不反相 ； 01：反相/下降沿：捕获发生在 TIxFP1 的下降沿（捕捉模式），TIxFP1 反相 ； 10：保留，不使用此配置 11：不反相/上升和下降沿：捕获发生在 TIxFP1 的上升沿和下降沿（捕捉模式），TIxFP1 不反相
BIT[0]	CC1E	捕获/比较 1 输出使能 (Capture/Compare 1 output enable) **CC1 通道配置为输出：** 0：关闭，OC1 禁止输出。 1：开启，OC1 信号输出到对应的输出引脚。 **CC1 通道配置为输入：** 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止； 1：捕获使能。

标准 OCx 通道的输出控制位

CCxE 位	OCx 输出状态
0	禁止输出 (OCx=0，OCx_EN=0)
1	OCx = OCxREF + 极性，OCx_EN=1

注： 连接到标准 OCx 通道的外部 I/O 引脚状态，取决于 OCx 通道状态和 GPIO 寄存器。

15.4.8 TIM14 计数器 (TIM14_CNT)

TIM14_CNT			地址偏移	0x24		复位值	0x0000 0000
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CNT[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CNT[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	CNT[15:0]	计数器值

15.4.9 TIM14 预分频器 (TIM14_PSC)

TIM14_PSC			地址偏移	0x28		复位值	0x0000 0000
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	PSC[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	PSC[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw

复位值	0	0	0	0	0	0	0	0

BIT[15:0]	PSC[15:0]	预分频值 计数器的时钟频率 CK_CNT=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。

15.4.10 TIM14 自动重装寄存器 (TIM14_ARR)

TIM14_ARR			地址偏移	0x2C		复位值	0x0000 FFFF	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	ARR[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	ARR[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	ARR[15:0]	自动重装载的值 (Prescaler value) ARR 是将要装载入的实际自动重载寄存器的值。详细参考<时基单元>章节：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。

15.4.11 TIM14 捕获/比较寄存器 1 (TIM14_CCR1)

TIM14_CCR1			地址偏移	0x34		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CCR1[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CCR1[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	CRR1[15:0]	捕获/比较 1 的值 **若 CC1 通道配置为输出：** CCR1 是将要装入当前捕获/比较 1 寄存器的值（预装载值）。 如果在 TIMx_CCMR1 寄存器（OC1PE 位）中未选择预装载功能，写入的数值会被立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。当前捕获/比较寄存器的值同计数器 TIMx_CNT 比较，并在 OC1 端口上产生输出信号。 **若 CC1 通道配置为输入：** CCR1 为上一次输入捕获 1 事件 (IC1) 传输的计数器值。

15.4.12 TIM14 选项寄存器 (TIM14_OR)

TIM14_CR1			地址偏移	0x50		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	TI1_RMP	
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	0	0

BIT[15:2]	-	保留，保持复位值
BIT[1:0]	TI1_RMP	定时器输入 1 重映射（Timer Input 1 remap），软件清除和置位 00: TIM14 通道 1 连接到 GPIO: 参考芯片数据手册中关于复用功能映射

		（ Alternate function mapping） 01: RTC_CLK 连接到 TIM14_CH1 输入端实现校正功能 10：HSE/32 时钟 连接到 TIM14_CH1 输入端 11：MCO 连接到 TIM14_CH1 输入端，MCO 时钟源的选择由时钟配置寄存器（RCC_CFGR）的 MCO[3:0]选择。

16 通用定时器（TIM16/17）

16.1 概述

通用定时器基于一个 16 位自动重载递增计数器和一个 16 位预分频器。
它们每个都有一个单通道，用于输入捕获、输出比较、PWM 模式输出。
TIM16 和 TIM17 完全独立，不共享任何资源。
在调试模式下，其计数器可被冻结。

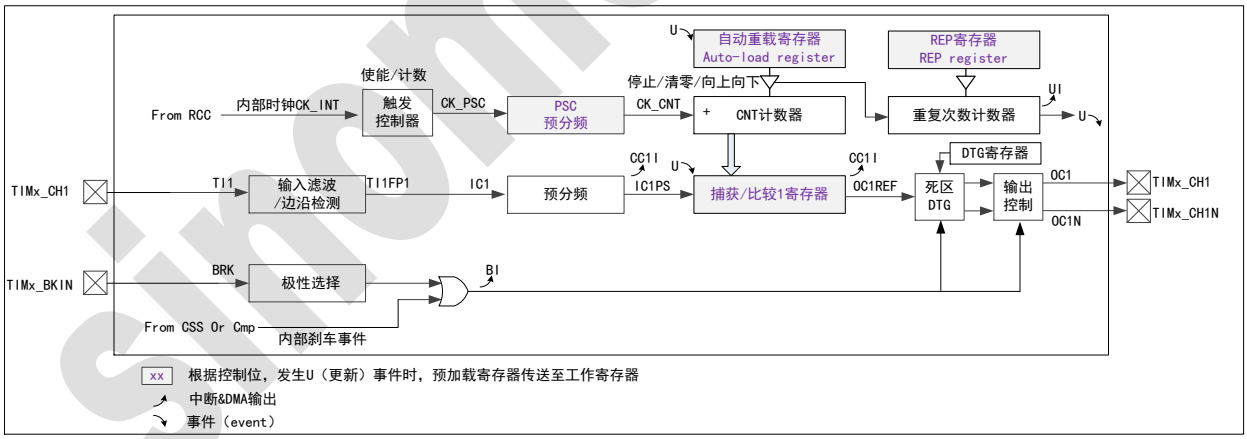
16.2 特性

✧ 16 位向上自动重载计数器
✧ 16 位可编程（ 可实时修改 ） 预分频器，支持 1～65535 之间的任意分频
✧ 2 个独立通道，支持
- 输入捕获
- 输出比较
- PWM 生成（边沿对齐）
- 单脉冲模式输出

✧ 互补输出，支持死区编程
✧ 重复计数器实现指定数目的计数器周期产生更新信号
✧ 定时器输出信号支持刹车保护
✧ 下述事件触发中断/DMA：
- 更新（Update）：计数器上溢
- 输入捕获（input capture）
- 输出比较（output compare）
- 刹车输入（break input）

16.3 功能描述

TIM16/17 模块框图



16.3.1 时基单元

高级定时器 TIM16/17 主要由 16 位计数器及相关自动重载寄存器构成，计数器支持向上计数。计数器时钟支持预分频。

计数器寄存器、自动重载寄存器和预分频寄存器支持软件读写，即使计数器正在运行读写仍然有效。

时基单元包括：

✧ 计数器寄存器（TIMx_CNT）
✧ 预分频器寄存器（TIMx_PSC）
✧ 自动重载寄存器（TIMx_ARR）
✧ 重复次数寄存器（TIMx_RCR）

自动重载寄存器是预装载的，读写自动重载寄存器将访问预装载寄存器。设置 TIMx_CR1 寄存器中的自动重载预装载使能位（ARPE），选择预装载寄存器的内容永久传送至缓存寄存器或在每次更新事件（UEV）传送至缓存寄存器。当计数

器达到溢出条件（或向下计数时的下溢条件）并当 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可由软件产生。有关更新事件的产生，针对每种配置后续章节会详细描述。

计数器时钟由预分频后输出 CK_CNT 驱动，需置位 TIMx_CR1 寄存器中的计数器使能位（CEN）时，CK_CNT 才有效（请参阅从模式控制器描述以获得计数器使能的更多细节）。

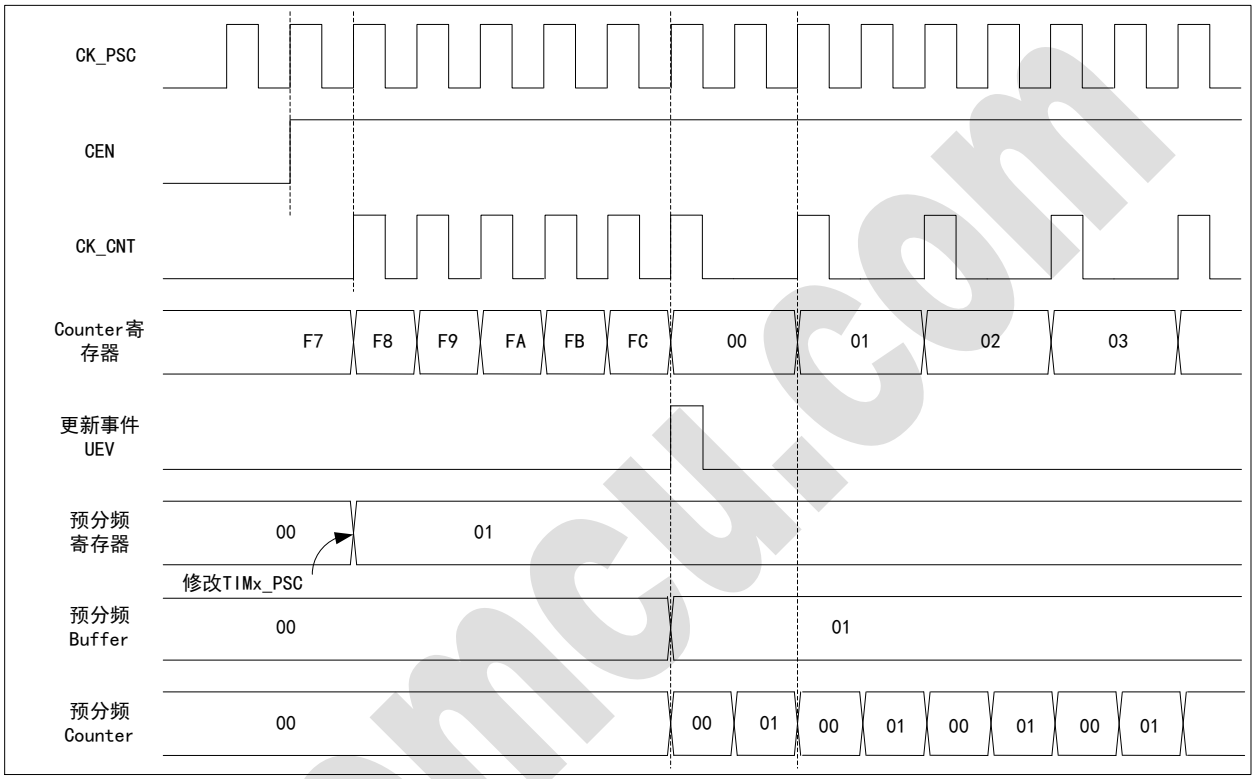
注：设置 TIMx_CR 寄存器的 CEN 位，一个时钟周期后，计数器才开始计数。

预分频

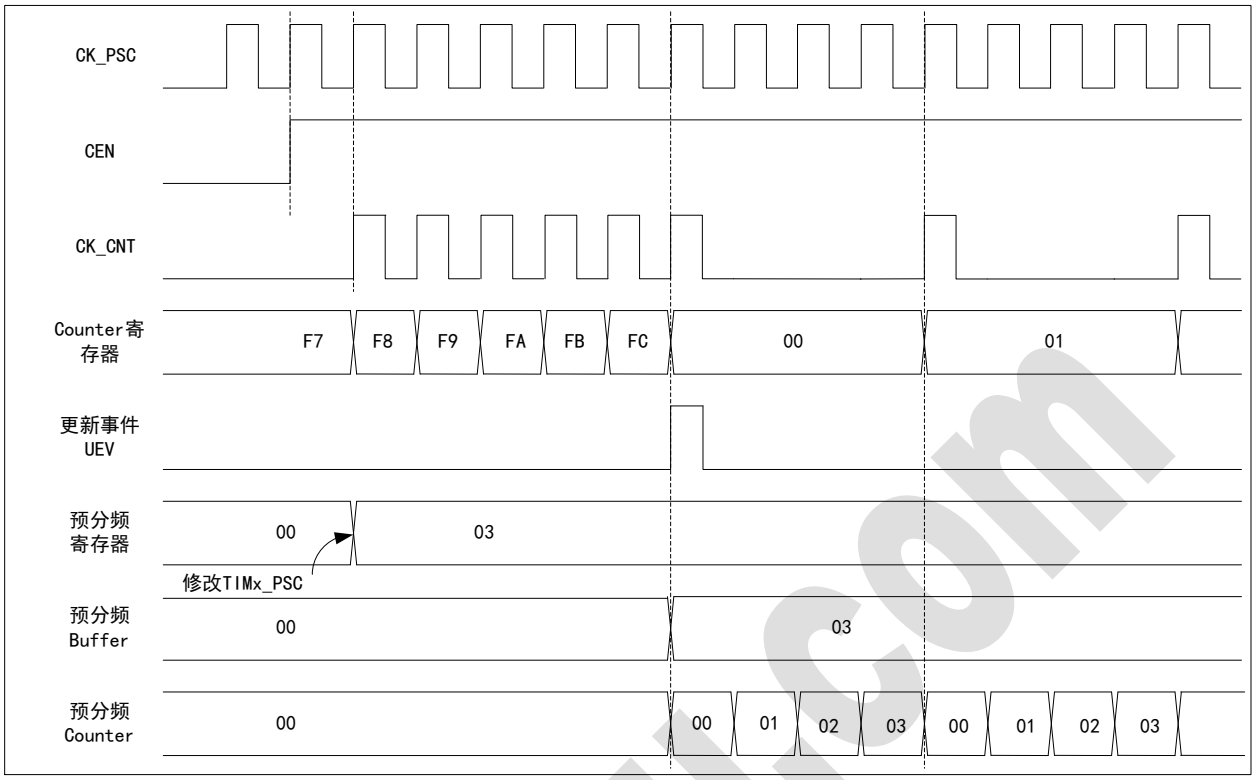
预分频器支持计数器时钟 1~65536 之间任意分频，基于 1 个 16 位的计数器，由 TIMx_PSC 寄存器中的 16 位寄存器控制。此寄存器内部带有缓存器，支持运行时修改；新修改的预分频值在下一次的更新事件（update event）发生时生效。

下图给出运行时更改预分频值，计数器行为。

计数器时序图：预分频修改，从 1 到 2



计数器时序图：预分频修改，从 1 到 4



16.3.2 计数器模式

向上计数模式

设置 TIMx_CR1 寄存器中的 CMS=00 且 DIR=0，执行计数器向上计数。

在向上计数模式中，计数器从 0 计数到自动重载值（TIMx_ARR 计数器的值 ），然后重新从 0 开始计数并且产生一个计数器溢出事件（overflow）。

如果使用了重复次数计数器（repetition counter）功能，对溢出重载次数进行向上计数，达到设置的重复次数（TIMx_RCR）时，才产生更新事件（UEV）；否则每次计数器溢出时都会产生更新事件。

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

通过软件置位 TIMx_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新缓存寄存器。在 UDIS 位被清‘0’之前，将不产生更新事件。但是，在应该产生更新事件时，计数器会被清‘0’，同时预分频器的计数器也被清 0（但预分频器的数值不变 ）。

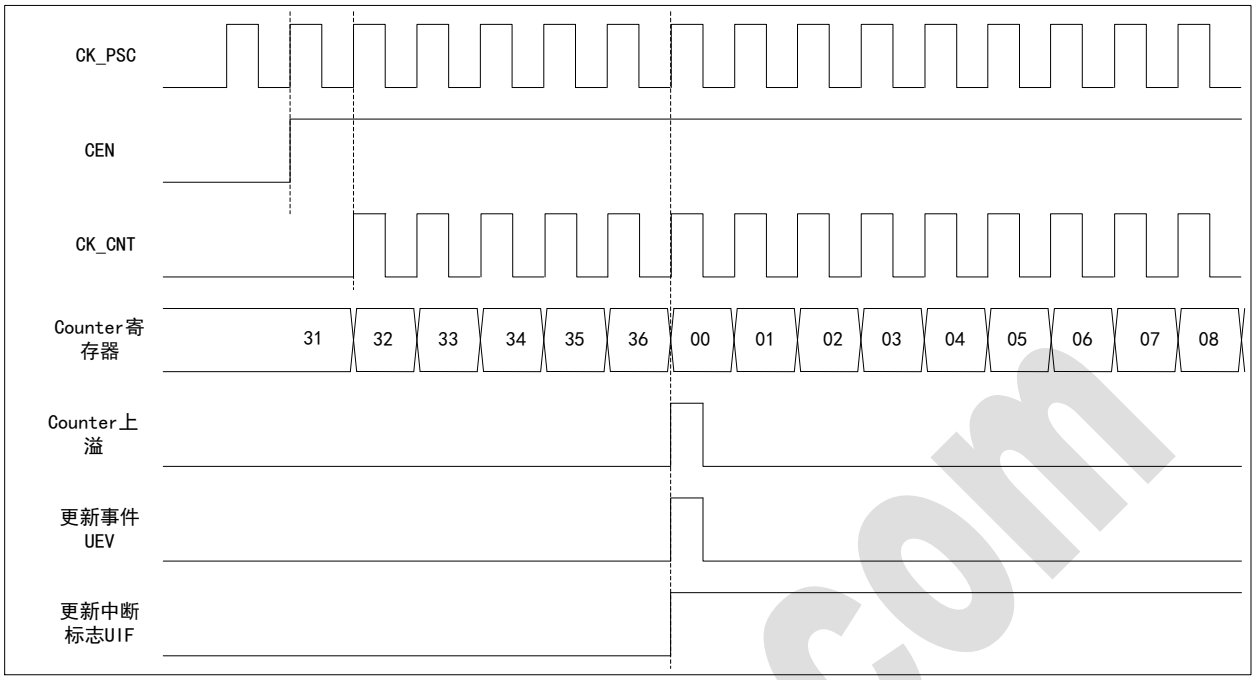
此外，如果置位 TIMx_CR1 寄存器中的 URS 位（update request selection），置位 UG 位将产生一个更新事件 UEV，但硬件不置位 UIF 标志（即不产生中断或 DMA 请求）。这是为了避免在发生捕获事件（capture event）清除计数器时，同时产生更新和捕获两个中断。

当发生一个更新事件时，所有的寄存器都被更新，同时（依据 URS 位）置位更新标志位（TIMx_SR 寄存器中的 UIF 位）：

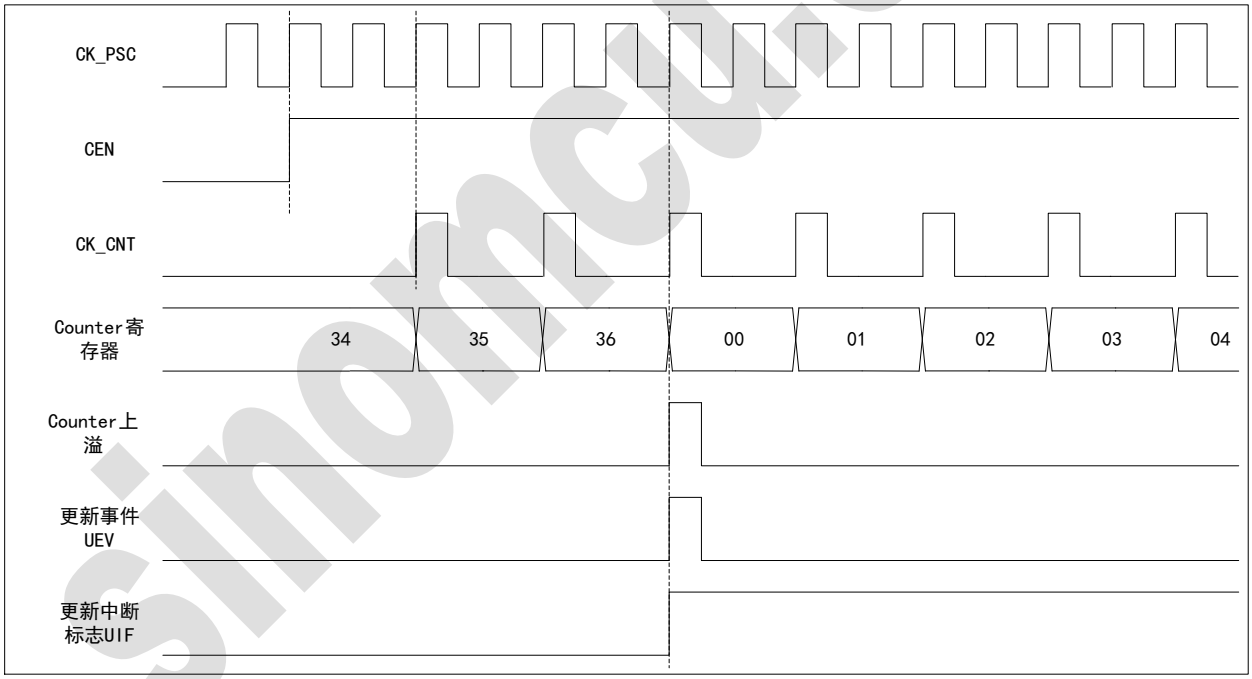
- ✧ 重复次数计数器被重新加载为 TIMx_RCR 寄存器的值
- ✧ 预分频器的缓冲区被写入预装载寄存器的值（TIMx_PSC 寄存器的值 ）
- ✧ 自动重载缓存寄存器更新为预装载寄存器的值（TIMx_ARR）

举例如下：TIMx_ARR=0x36 时，计数器在不同时钟频率下计数器行为。

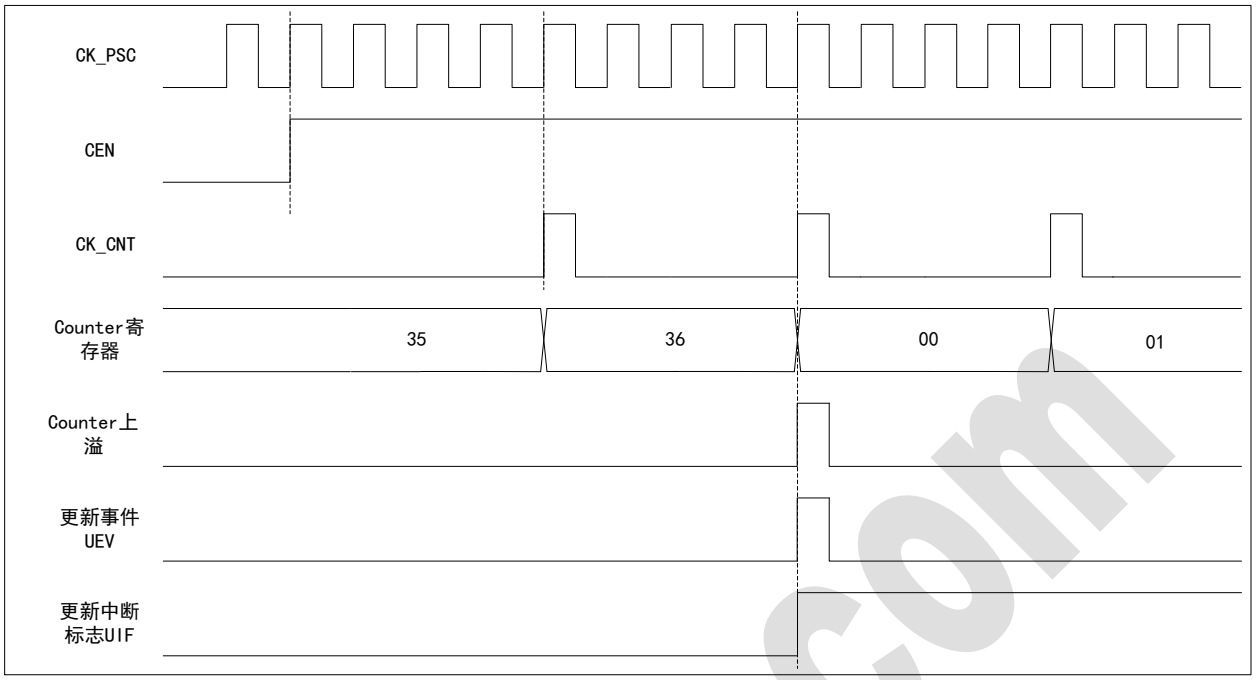
计数器时序图：预分频因子=1



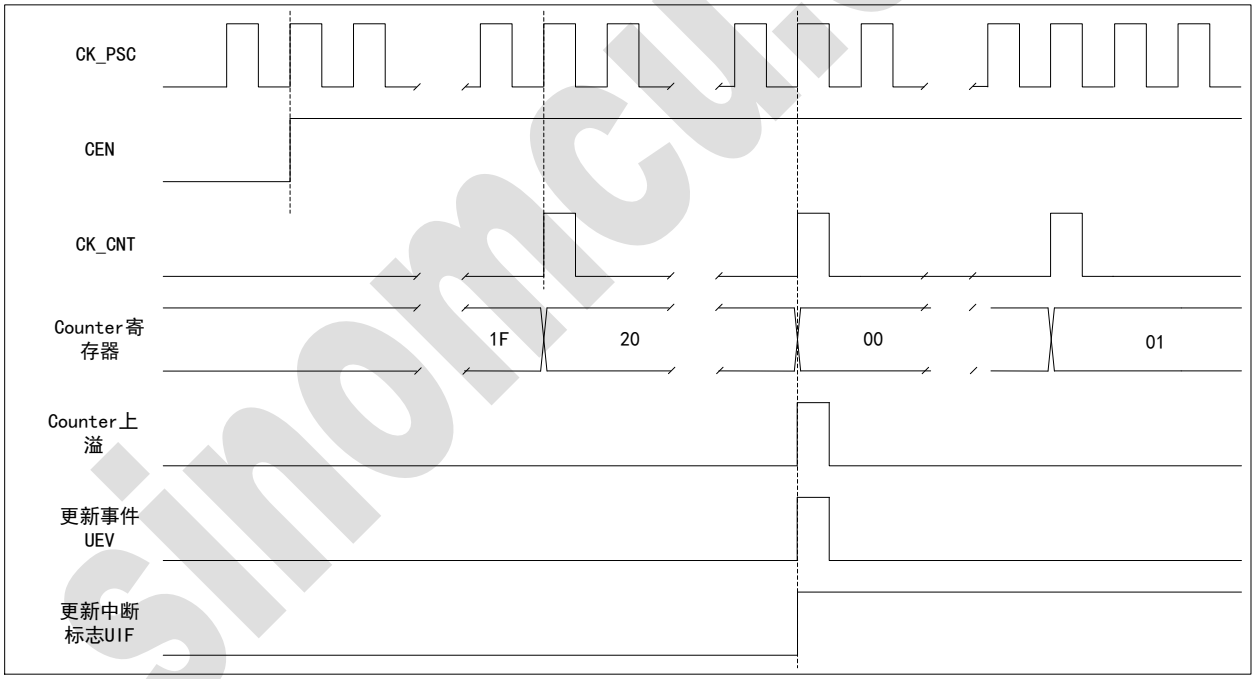
计数器时序图：预分频因子=2



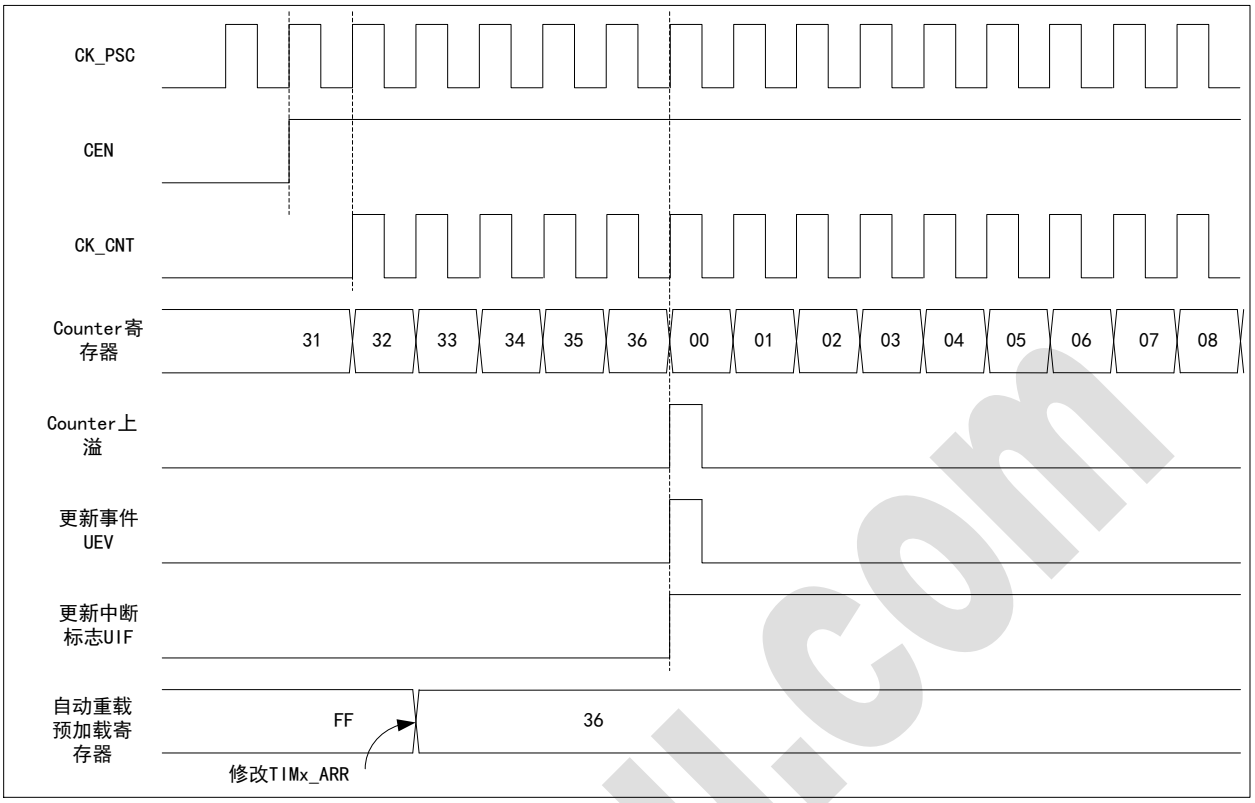
计数器时序图：预分频因子=4



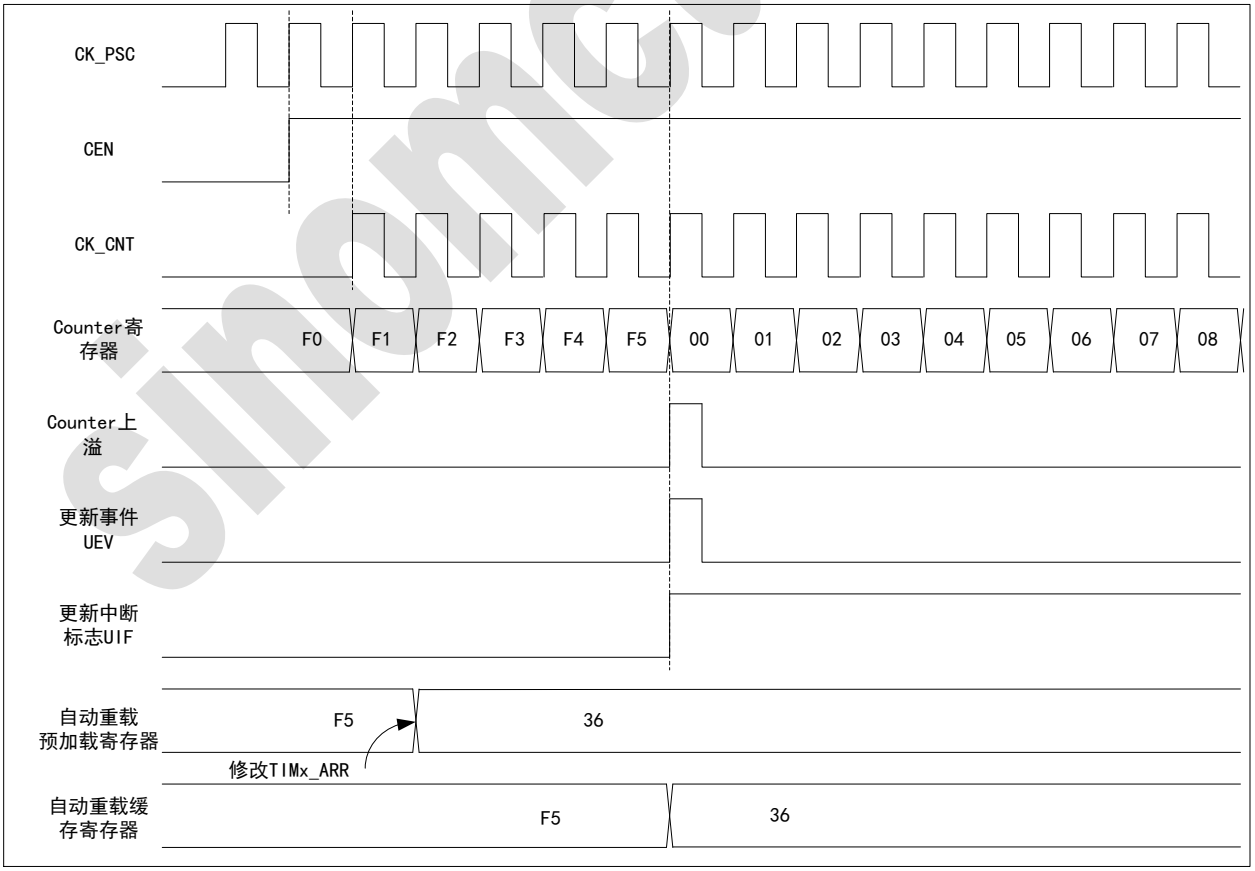
计数器时序图：预分频因子=N



计数器时序图：预分频因子=1，ARPE=0（TIMx_ARR 没有预装载）



计数器时序图：预分频因子=1，ARPE=1（TIMx_ARR 已预装载）



16.3.3 重复次数计数器（repitition counter）

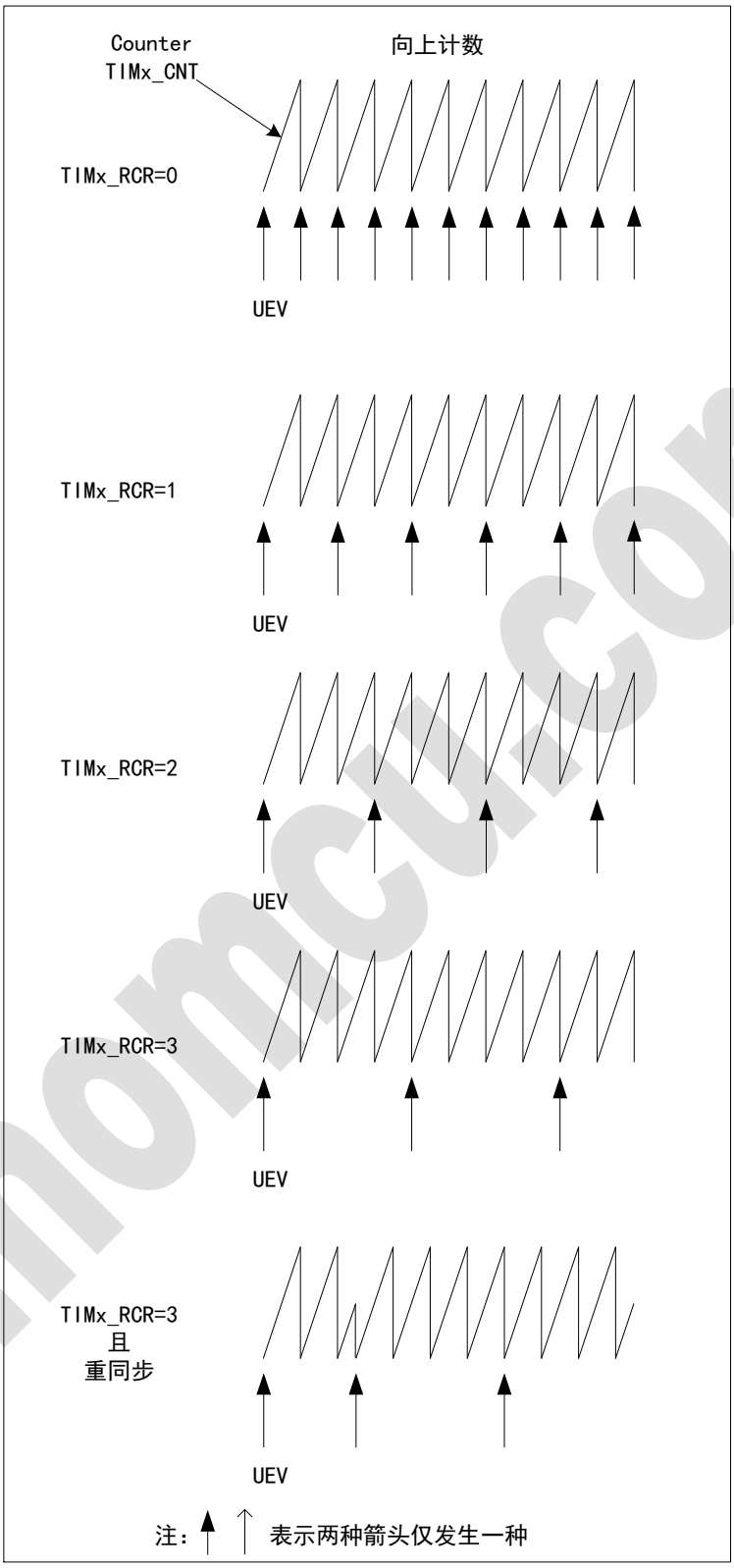
16.3.1 章节<时基单元>解释了计数器上溢时是如何产生更新事件(UEV)的，然而事实上它只能在重复次数计数器归 0 时产生。这个特性对产生 PWM 信号非常有用。

这意味着在每 N 次计数上溢时，数据从预装载寄存器传送至缓存寄存器(TIMx_ARR 自动重载寄存器，TIMx_PSC 预分频寄存器，还有在比较模式下的捕获/比较寄存器 TIMx_CCRx)，N 为 TIMx_RCR 重复计数寄存器中的值。

重复计数器在向上计数模式下每次计数器溢出时递减。

重复计数器是自动重载的，重复次数是由 TIMx_RCR 寄存器的值定义。当更新事件由软件产生（通过设置 TIMx_EGR 中的 UG 位）或者通过硬件（从模式控制器）产生，则无论重复计数器的值是多少，立即发生更新事件，并且 TIMx_RCR 寄存器中的内容被重载入到重复计数器。

RCR 更新举例：不同模式及不同 TIMx_RCR 寄存器设置



16.3.4 时钟源

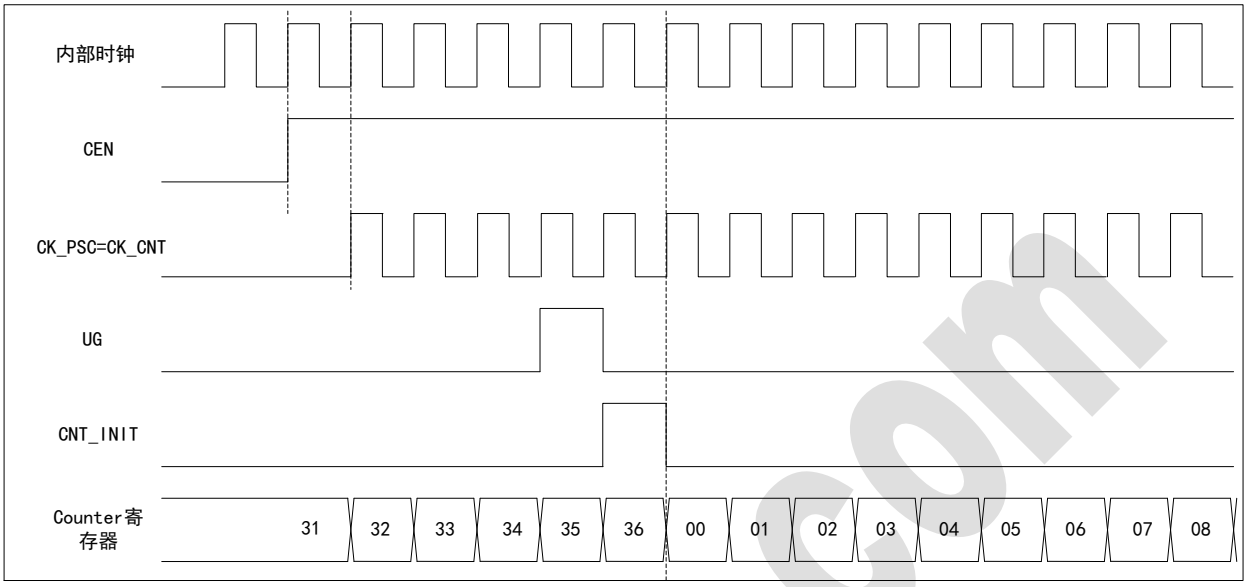
计数器时钟源包括：

✧ 内部时钟 CK_INT

内部时钟源（CK_INT）

CEN 和 UG 位（TIMx_ EGR 寄存器）为实际控制位，并且只能被软件修改（ 除了 UG 位保持自动被清除）。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。

计数器时序图：内部时钟，预分频因子=1，正常模式



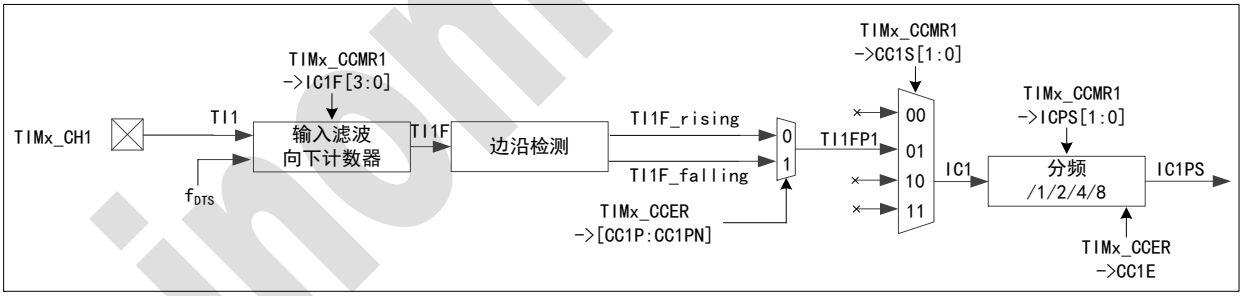
16.3.5 捕获/比较通道

每一个捕获/比较通道都是包括：一个捕获/比较寄存器（包含缓存寄存器），一个用于捕获的输入级（带数字滤波、多路复用选择和预分频器），和一个输出级（带比较器和输出控制）。

输入级

对 TIx 的输入信号进行采样，产生一个滤波后 TIxF 信号，经过带有极性选择的边沿检测器生成 TIxFPx 信号，TIxFPx 可以作为从模式控制器的触发输入或者作为捕获命令。该信号预分频后（ICxPS）进入捕获寄存器。

捕获/比较通道输入级（举例：通道 1）



输出级

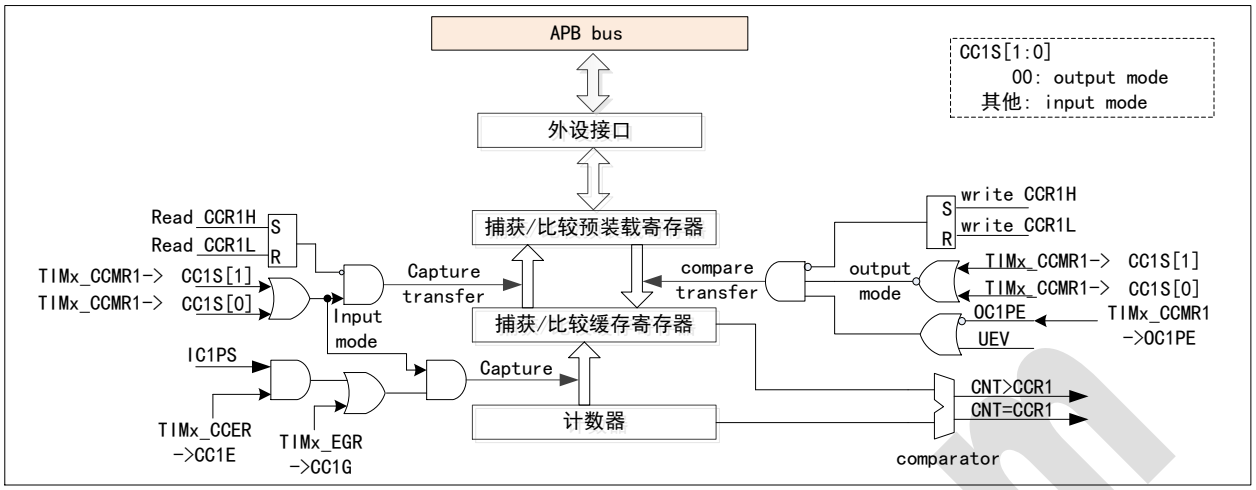
输出部分产生一个中间波形 OCxRef（ 高有效）作为基准，链的末端决定最终输出信号的极性。

捕获/比较模块由一个预装载寄存器（preload register）和一个缓存寄存器（shadow register）组成。读写过程仅操作预装载寄存器。

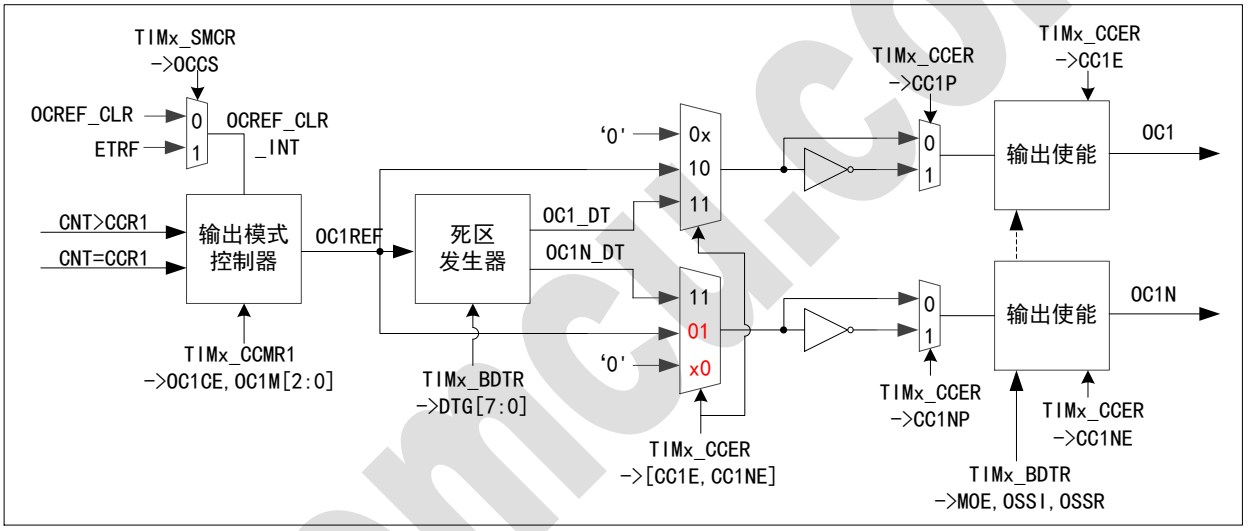
在捕获模式下，捕获发生在缓存寄存器上，然后再复制到预装载寄存器中。

在比较模式下，预装载寄存器的内容被复制到缓存寄存器中，然后缓存寄存器的内容和计数器进行比较。

捕获/比较通道主电路



捕获/比较通道输出级（通道 1，通道 2/3 同通道 1）



16.3.6 输入捕获模式

在输入捕获模式下，当检测 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（TIMx_CCRx）中。当发生捕获事件时，相应 CCxIF 标志（TIMx_SR 寄存器）被置 1， 如果使能了中断或者 DMA，则产生一个中断或者一个 DMA 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（TIMx_SR 寄存器）被置 1。软件写 CCxIF=0 可清除 CCxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。

举例，在 TI1 输入的上升沿时，捕获计数器的值到 TIMx_CCR1 寄存器中，操作步骤如下：

1、 选择 TIMx_CCR1 的有效输入：置 TIMx_CCMR1 寄存器的 CC1S=01（选中 TI1），只要 CC1S 不为‘00’，通道被配置为输入并且 TIMx_CCR1 寄存器变为只读。

2、 根据连接到计数器的输入信号，配置输入滤波器时间（输入为 TIx 时，输入滤波器控制位是 TIMx_CCMRx 寄存器中的 ICxF 位）。举例，当 TI1 翻转时，信号抖动最多 5 个内部时钟， 则须配置滤波器时间大于 5 个时钟周期，TIMx_CCMR1 寄存器中写入 IC1F=0011 配置采样次数 8（以 f_{DTS} 频率采样），通过连续 8 次采样以确认电平变换。

3、 选择 TI1 通道有效沿。写 TIMx_CCER 寄存器中 CC1P=0 和 CC1NP=0（本例中为上升沿）。

4、 配置输入预分频器。在本例中，设置为每个有效的电平转换时发生一次捕获，因此预分频器被禁止（写 TIMx_CCMR1 寄存器的 IC1PS=00）。

5、 写 TIMx_CCER 寄存器的 CC1E=1，允许计数器的值被捕获至捕获寄存器中。

6、 根据需要，置位 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求，置位 TIMx_DIER 寄存器中的 CC1DE 位允许 DMA 请求。

当发生一个输入捕获时：

- ✧ 有效的电平转换发生时，计数器的值被传送至 TIMx_CCR1 寄存器；
- ✧ CC1IF 标志被置位（中断标志）。当发生至少 2 个连续的捕获时，且 CC1IF 未被清除，CC1OF 也被置位；
- ✧ 若置位 CC1IE 位，则会产生一个中断。

✧ 若置位 CC1DE 位，则会产生一个 DMA 请求。

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，这是为了避免在读取标志之后和读取数据之前可能发生的捕获溢出。

注： 通过软件设置 TIMx_EGR 寄存器中相应的 CCxG 位，也可以产生输入捕获中断 和/或 DMA 请求。

16.3.7 强制输出模式

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

设置 TIMx_CCMRx 寄存器中相应的 OCxM=101，输出比较信号（OCxREF/OCx）强制为有效状态。这样 OCxREF 被强制高电平(OCxREF 始终为高电平有效)，置位 TIMx_CCER 的 CCxP 位，OCx 可得到极性相反的信号。

举例，CCxP=0（OCx 高电平有效），则 OCx 被强制为高电平。

设置 TIMx_CCMRx 寄存器中相应的 OCxM=100，输出比较信号（OCxREF/OCx）强制为低电平。

该模式下，在 TIMx_CCRx 缓存寄存器和计数器之间的比较仍然在执行，相应的标志也会被置位。并会产生相应的中断和 DMA 请求。下面的输出比较模式一节中将详细介绍。

16.3.8 输出比较模式

此模式用于控制输出波形或指示一段时间到时。

当计数器与捕获/比较寄存器的内容匹配（相等）时，输出比较功能做如下操作：

✧ 根据输出比较模式（TIMx_CCMRx 寄存器中 OCxM 位）和输出极性（TIMx_CCER 寄存器中的 CCxP 位）的配置，输出至对应引脚。在比较匹配发生时，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）。

✧ 置位中断状态寄存器中的标志位（TIMx_SR 寄存器中的 CCxIF 位）。

✧ 若置位相应的中断屏蔽位（TIMx_DIER 寄存器中的 CCxIE 位），则产生一个中断。

✧ 若置位相应的 DMA 使能位 (TIMx_DIER 寄存器中的 CCxDE 位，TIMx_CR2 寄存器中的 CCDS 位)，则产生一个 DMA 请求。

设置 TIMx_CCMRx 中的 OCxPE 位选择 TIMx_CCRx 寄存器是否使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。计时分辨率为计数器的一次计数。

输出比较模式也能用来输出一个单脉冲（单脉冲模式）。

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、 根据需求，写入数据至 TIMx_ARR 和 TIMx_CCRx 寄存器中。

3、 如果要产生一个中断请求，设置 CCxIE 位。

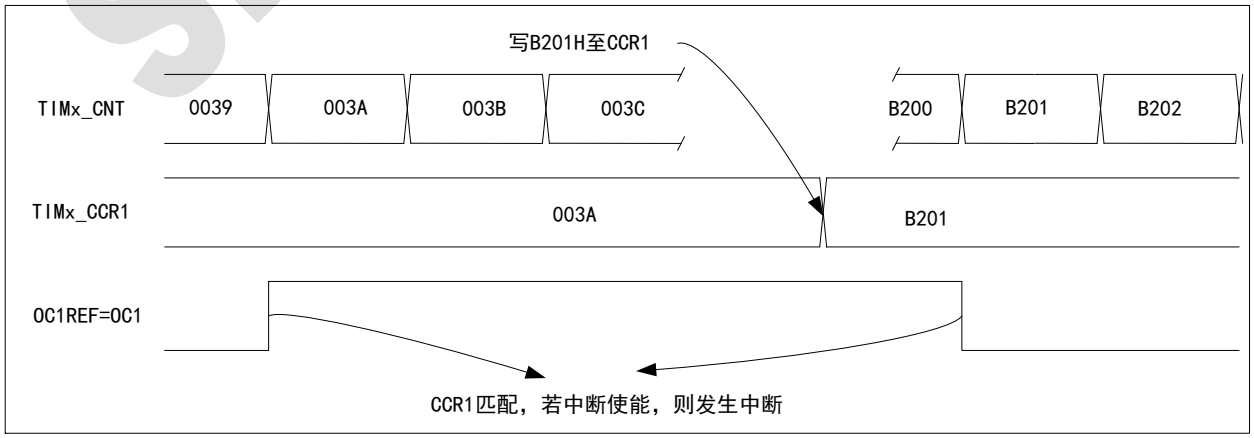
4、 选择输出模式，举例如下：

- 写 OCxM=011，计数器与 CCRx 匹配时，翻转 OCx 的输出引脚；
- 写 OCxPE=0，禁用预装载寄存器；
- 写 CCxP=0，选择极性为高电平有效；
- 写 CCxE=1，使能输出。

5、 置位 TIMx_CR1 寄存器的 CEN 位，启动计数器。

TIMx_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，前提是未启用预装载寄存器（OCxPE= ‘0’，否则 TIMx_CCRx 的缓存寄存器只在发生下次更新事件 EUV 时更新）。举例如下图。

输出比较模式，配置为：翻转 OC1



16.3.9 PWM 模式

脉冲宽度调制模式（PWM）可以产生一个 PWM 信号，由 TIMx_ARR 寄存器确定频率、由 TIMx_CCRx 寄存器确定占空比。

在 TIMx_CCMRx 寄存器中的 OCxM 位写入‘110’（PWM 模式 1）或‘111’（PWM 模式 2）， 可以在每个通道独立地设置 PWM 模式（每个 OCx 输出一种 PWM）。必须置位 TIMx_CCMRx 寄存器的 OCxPE 位，使能相应的预装载寄存器（preload register），最后置位 TIMx_CR1 寄存器的 ARPE 位，使能自动重载预装载寄存器（在向上计数模式中）。

只有发生一个更新事件时，预装载寄存器才能被传送到缓存寄存器，因此在计数器开始计数之前，必须通过设置 TIMx_EGR 寄存器中的 UG 位来初始化所有的寄存器。

设置 TIMx_CCER 寄存器中的 CCxP 位，配置 OCx 的极性为高电平有效或低电平有效。OCx 的输出使能通过 （TIMx_CCER 和 TIMx_BDTR 寄存器中）CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。详细参考 TIMx_CCER 寄存器的描述。

在 PWM 模式（模式 1 或模式 2）下，TIMx_CNT 和 TIMx_CCRx 始终在进行比较，以确定是否符合 TIMx_CNT ≤ TIMx_CCRx。

由于此计数器为向上计数，仅支持边沿对齐模式的 PWM 信号。

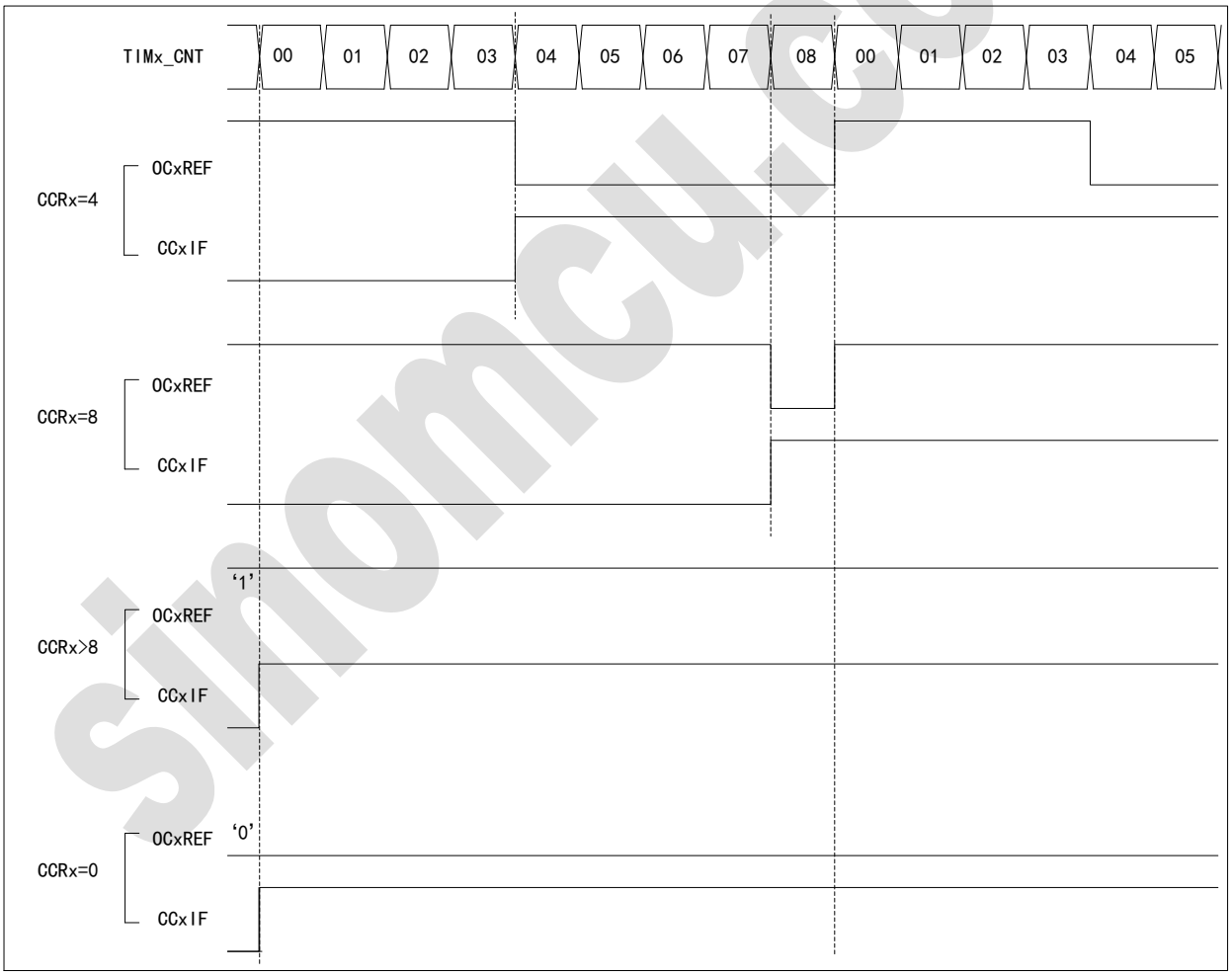
PWM 边沿对齐模式

向上计数配置

举例，PWM 模式 1

当 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重载值（TIMx_ARR），则 OCxREF 保持为‘1’。如果比较值为 0，则 OCxREF 保持为‘0’。 下图为 TIMx_ARR=8 时边沿对齐的 PWM 波形实例。

PWM 波形（边沿对齐，向上计数，ARR=8）



16.3.10 互补输出/死区插入

通用定时器（TIM16/17）能够输出两路互补信号，并且能够管理输出关断和接通的瞬时时间。

这段时间通常被称为死区，用户应该根据输出连接的器件和它们的特性（电平转换的延时、电源开关的延时等 ）来调整死区时间。

配置 TIMx_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：TIMx_CCER 寄存器的 CCxE 和 CCxNE 位，TIMx_BDTR 和 TIMx_CR2

寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位。详见 16.4.8 章节表格<带刹车功能的互补输出通道 OCx 和 OCxN 的控制位>。特别的是，在切换到 IDLE 状态时（MOE 下降到 0）死区被激活。

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 10 位的死区发生器。参考信号 OCxREF 可以产生 2 路输出 OCx 和 OCxN。

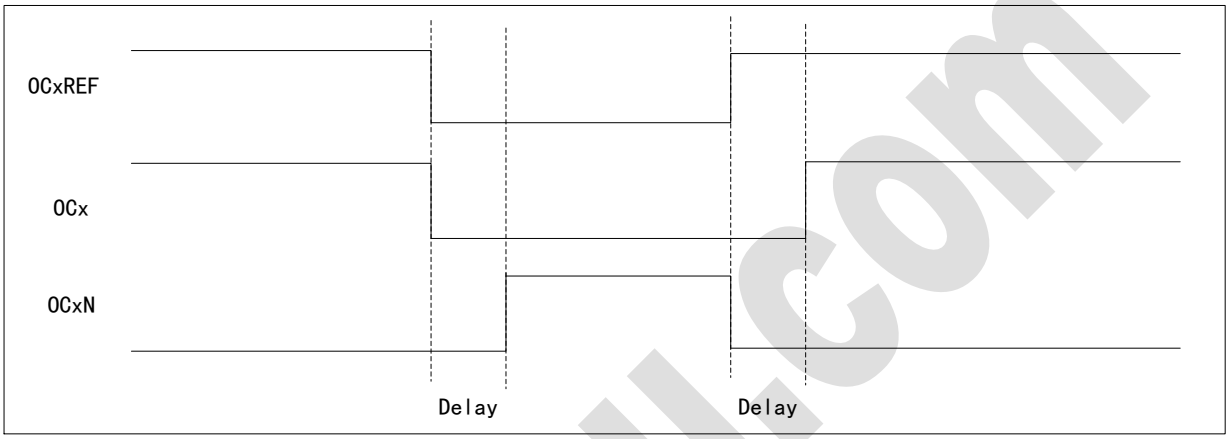
如果 OCx 和 OCxN 为高有效：

- ✧ OCx 输出信号与 OCxREF 参考信号相同，只是上升沿相对于参考信号的上升沿有一个延迟（死区）。
- ✧ OCxN 输出信号与 OCxREF 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（死区）。

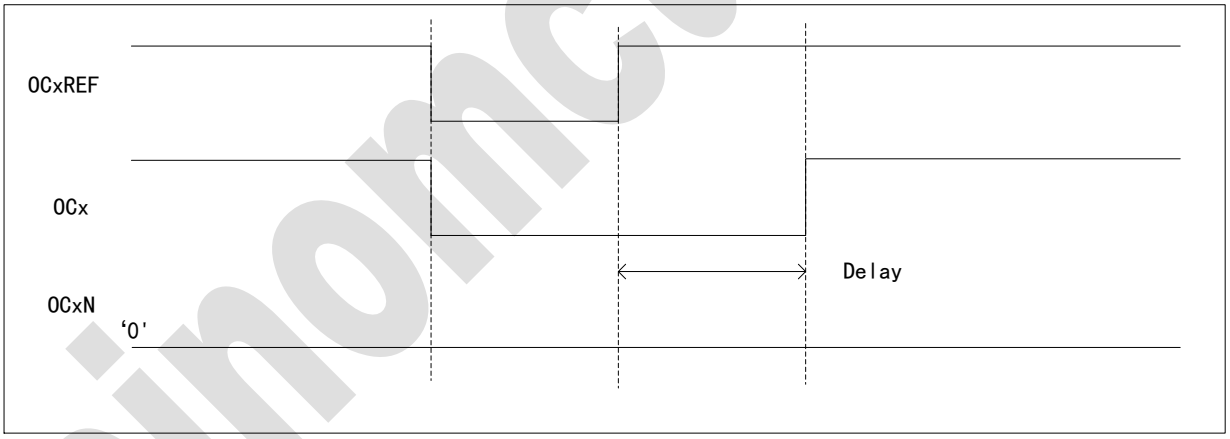
如果延迟大于当前有效的输出宽度（OCx 或者 OCxN），则不会产生相应的脉冲。

下列几张图为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 且 CCxNE=1）

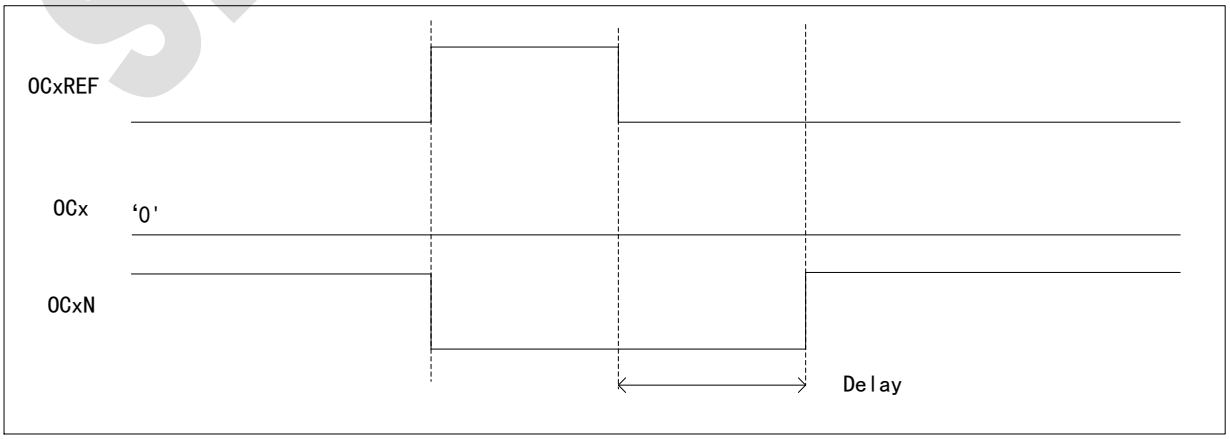
带死区插入的互补输出



死区波形：延迟大于负脉冲



死区波形：延迟大于正脉冲



每一个通道的死区延时都相同，都由 TIMx_BDTR 寄存器中的 DTG 位编程配置。具体的延时计算参见 16.4.19 章节刹

车和死区寄存器（TIMx_BDTR）。

重定向 OCxREF 到 OCx 或 OCxN

在输出模式下（强制输出、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，或都处于有效电平且带死区的互补输出。

注：当只使能 OCxN（CCxE=0, CCxNE=1）时，OCxN 不会经过取反，一旦 OCxREF 有效时立即变高。例如，如果 CCxNP=0，则 OCxN=OCxREF。

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。

16.3.11 刹车功能

当使用刹车功能时，设置额外的控制位（TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIMx_CR2 寄存器中 OISx 和 OISxN 位），输出使能信号和无效电平将会被修改。但无论何时，OCx 和 OCxN 输出电平不能在同一时间被设置都有效。详见 16.4.8 章节表格<带刹车功能的互补输出通道 OCx 和 OCxN 的控制位>。

刹车信号（BRK）源可以是连接刹车输入引脚的外部信号，也可以是以下内部信号之一：

- ✧ LOCKUP 输出
- ✧ PVD 输出
- ✧ SRAM 奇偶校验错误信号
- ✧ 时钟故障事件，由时钟安全系统（CSS）产生

注：TIM16/17 不支持比较器输出作为刹车信号

系统复位后，刹车电路被禁止，MOE 位为低。设置 TIMx_BDTR 寄存器中的 BKE 位可以使能刹车功能，设置同一寄存器的 BKP 位，选择刹车输入信号的极性。BKE 和 BKP 可以同时被修改。当写入 BKE 和 BKP 位时，延迟 1 个 APB 时钟周期写入生效，因此需要等待一个 APB 时钟周期之后，才能正确回读。

因为 MOE 下降沿可以是异步的，在实际信号（作用在输出端）和同步控制位（在 TIMx_BDTR 寄存器中）之间插入了一个重新同步电路。这个重新同步电路会在异步信号和同步信号之间产生延迟。特别的，如写 MOE 从 0 变为 1，则必须先插入一个延时（空指令）才能正确回读。这是因为写入是异步信号而读取是同步信号。

当发生刹车时（在刹车输入发生选定的电平），有下述动作：

- ✧ MOE 位被异步清零，将输出置于无效状态、空闲状态或者复位状态（由 OSSI 位选择）。这个特性在 MCU 的振荡器关闭时依然有效。
- ✧ 一旦 MOE=0，每一个输出通道电平由 TIMx_CR2 寄存器中的 OISx 位设定。如果 OSSI=0，则定时器释放（release）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 ）。这是异步操作，即使定时器没有时钟时，此功能也有效。
  - 如果定时器的时钟依然存在，死区生成器将会重新生效，在 1 个死区之后根据 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。注意，因为重新同步 MOE，死区持续时间比通常情况下长一点（大约 2 个 ck_tim 的时钟周期）。
  - 如果 OSSI=0，定时器释放输出使能，否则，CCxE 或 CCxNE 为高时，输出使能为高。
- ✧ 置位刹车状态标志（TIMx_SR 寄存器中的 BIF 位），如果设置了 TIMx_DIER 寄存器中的 BIE 位，则产生一个中断。
- ✧ 如果置位 TIMx_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时，MOE 位被重新自动置位；例如，这可以用来进行整形。否则（AOE=0），MOE 始终保持低直到被再次置‘1’；此时， 这个特性可以用于安全防护，用户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。

注： 刹车输入为电平有效。所以，当刹车输入有效时，MOE 不能被置位（硬件自动或者软件都不可以）。同时，状态标志 BIF 不能被清除。

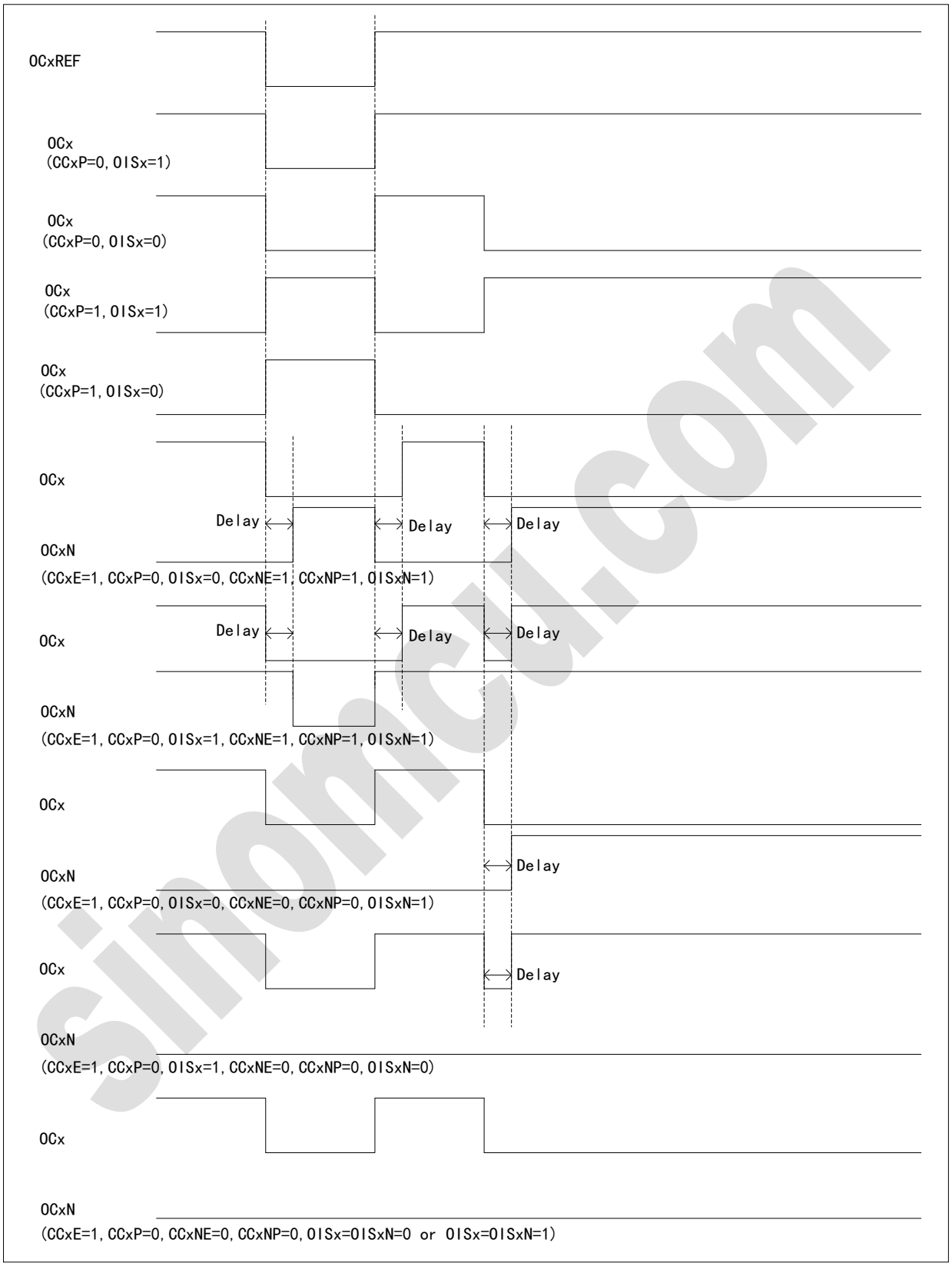
刹车可以由 BRK 输入产生，有效极性可编程，TIMx_BDTR 寄存器中的 BKE 位为使能控制。

除了刹车输入和输出管理，刹车电路中还支持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（死区时间，OCx/OCxN 极性和关闭状态，OCxM 配置，刹车使能和极性）。用户可以通过设置 TIMx_BDTR 寄存器中的 LOCK 位，选择三个保护等级中的一种，详细参见 16.4.14 节 TIM1 刹车和死区寄存器(TIMx_BDTR)。

MCU 复位后，LOCK 位只能被修改一次。

下图为刹车响应的输出实例。

刹车响应的输出



16.3.12 单脉冲模式

单脉冲模式（OPM）是上述模式的一个特例。这种模式下，计数器响应一个激励后启动，经过一个可编程的延时之后，产生一个脉宽编程的单脉冲。

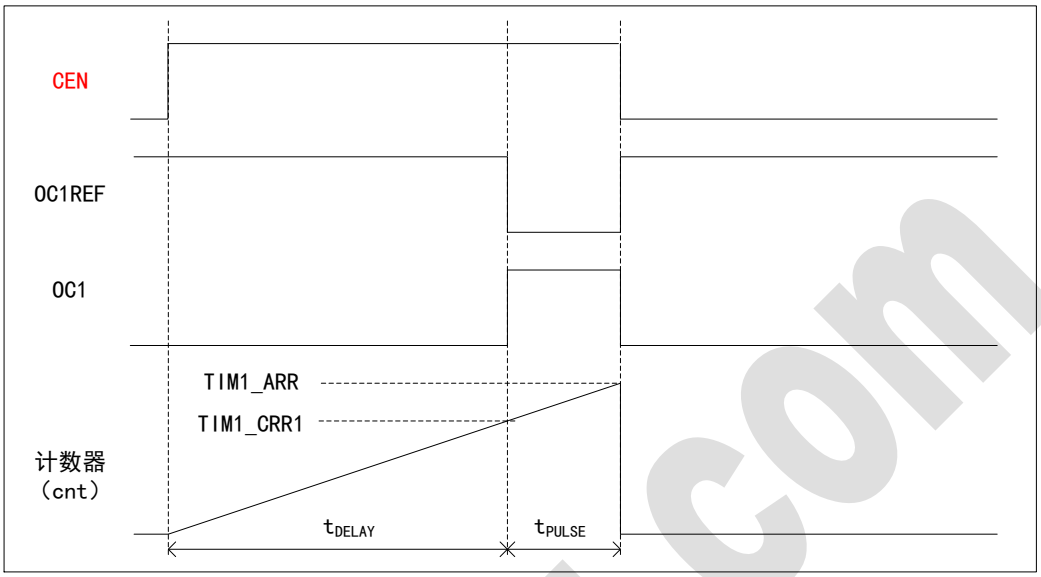
计数器启动由从模式控制器控制，在输出比较模式或者 PWM 模式下产生波形。设置 TIMx_ CR1 寄存器中的 OPM 位将

选择单脉冲模式，计数器在产生下一个更新事件 UEV 时自动停止。

想要正确产生一单脉冲，比较值与计数器的初始值必须不同。启动之前（ 此时定时器正在等待触发）， 配置必须满足：

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

单脉冲模式时序举例



举例，要求在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲，从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后。OPM 的波形由写入比较寄存器的值决定 （ 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（TIMx_ARR - TIMx_CCR1）。

✧ 若要求波形，比较值匹配时从 0 到 1 翻转，当计数器达到预装载值时从 1 到 0 翻转；

- 首先要设置 TIMx_CCMR1 寄存器的 OC1M=111，选择 PWM 模式 2；
- 有选择地使能预装载寄存器：置 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；
- 在 TIMx_CCR1 寄存器中写入比较值，在 TIMx_ARR 寄存器中写入自动重载值，设置 UG 位来产生一个更新事件，
- 最后使能 CEN 位，启动计数器。本例中，CC1P=0。

设置 TIMx_CR 寄存器中的 OPM=1，在下一个更新事件（UEV，当计数器从自动装载值翻转到 0）时停止计数。当 TIMx_CR1 寄存器中的 OPM=0 时，进入重复模式。

特殊情况：OCx 快速使能：

在单脉冲模式下，TIx 输入脚的边沿检测逻辑触发 CEN 位来启动计数器。通过计数器和比较值间的比较结果驱动输出翻转。但是，这些操作需要一定的时钟周期，这样就限制了可获得的最小延时 t_{DELAY}。

如果需要以最小延时输出波形，可以置位 TIMx_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF（和 OCx）直接响应激励而不再依赖比较，翻转后电平与比较匹配时的电平一致。OCxFE 只在通道配置为 PWM 模式 1 和 PWM 模式 2 时起作用。

16.3.13 调试模式

当微控制器进入调试模式时(Cortex-M0 内核停止)，根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以或者继续正常操作，或者停止。

16.4 相关寄存器

16.4.1 寄存器汇总表

基址：0x4001 4400（TIM16）				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	TIM16_CR1	0x0000 0000	TIM16 控制寄存器 1	16.4.2
0x04	TIM16_CR2	0x0000 0000	TIM16 控制寄存器 2	16.4.3
0x08	Res.	-	-	
0x0C	TIM16_DIER	0x0000 0000	TIM16 DMA/中断使能寄存器	16.4.4
0x10	TIM16_SR	0x0000 0000	TIM16 状态寄存器	16.4.5
0x16	TIM16_EGR	0x0000 0000	TIM16 事件产生寄存器	16.4.6

0x18	TIM16_CCMR1	0x0000 0000	TIM16 捕获/比较模式寄存器 1	16.4.7
0x1C	Res.	-	-	
0x20	TIM16_CCER	0x0000 0000	TIM16 捕获/比较使能寄存器	16.4.8
0x24	TIM16_CNT	0x0000 0000	TIM16 计数器	16.4.9
0x28	TIM16_PSC	0x0000 0000	TIM16 预分频器	16.4.10
0x2C	TIM16_ARR	0x0000 0000	TIM16 自动重载寄存器	16.4.11
0x30	TIM16_RCR	0x0000 0000	TIM16 重复计数寄存器	16.4.12
0x34	TIM16_CCR1	0x0000 0000	TIM16 捕获/比较寄存器 1	16.4.13
0x38~0x40	Res.	-	-	
0x44	TIM16_BDTR	0x0000 0000	TIM16 刹车和死区寄存器	16.4.14
0x48	TIM16_DCR	0x0000 0000	TIM16 DMA 控制寄存器	16.4.15
0x4C	TIM16_DMAR	0x0000 0000	TIM16 DMA 地址（完全传输）	16.4.16
基址：0x4001 4800（TIM17）				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	TIM17_CR1	0x0000 0000	TIM17 控制寄存器 1	16.4.2
0x04	TIM17_CR2	0x0000 0000	TIM17 控制寄存器 2	16.4.3
0x08	Res.	-	-	
0x0C	TIM17_DIER	0x0000 0000	TIM17 DMA/中断使能寄存器	16.4.4
0x10	TIM17_SR	0x0000 0000	TIM17 状态寄存器	16.4.5
0x16	TIM17_EGR	0x0000 0000	TIM17 事件产生寄存器	16.4.6
0x18	TIM17_CCMR1	0x0000 0000	TIM17 捕获/比较模式寄存器 1	16.4.7
0x1C	Res.	-	-	
0x20	TIM17_CCER	0x0000 0000	TIM17 捕获/比较使能寄存器	16.4.8
0x24	TIM17_CNT	0x0000 0000	TIM17 计数器	16.4.9
0x28	TIM17_PSC	0x0000 0000	TIM17 预分频器	16.4.10
0x2C	TIM17_ARR	0x0000 0000	TIM17 自动重装载寄存器	16.4.11
0x30	TIM17_RCR	0x0000 0000	TIM17 重复计数寄存器	16.4.12
0x34	TIM17_CCR1	0x0000 0000	TIM17 捕获/比较寄存器 1	16.4.13
0x38~0x40	Res.	-	-	
0x44	TIM17_BDTR	0x0000 0000	TIM17 刹车和死区寄存器	16.4.14
0x48	TIM17_DCR	0x0000 0000	TIM17 DMA 控制寄存器	16.4.15
0x4C	TIM17_DMAR	0x0000 0000	TIM17 DMA 地址（完全传输）	16.4.16

16.4.2 TIM16 和 TIM17 控制寄存器 1 (TIM16_CR1 and TIM17_CR1)

TIM16_CR1 / TIM17_CR1			地址偏移	0x00		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	CKD[1:0]	
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	ARPE	-	-	-	OPM	URS	UDIS	CEN
R/W	rw	-	-	-	rw	rw	rw	rw
复位值	0	-	-	-	0	0	0	0

BIT[15:10]	-	保留，保持复位值
BIT[9:8]	CKD[1:0]	时钟分频因子（Clock division） 定义在定时器时钟（CK_INT）频率与数字滤波器（ETR,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
BIT[7]	ARPE	自动重装载预装载允许位（Auto-reload preload enable） 0: TIMx_ARR 寄存器没有缓冲 1: TIMx_ARR 寄存器缓冲器有效
BIT[6:4]	-	保留，保持复位值

BIT[3]	OPM	单脉冲模式 (One pulse mode) 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
BIT[2]	URS	更新请求源 (Update request source) 软件通过该位选择 UEV 事件的源 0：如果使能了 UEV 则下述任一事件产生 UEV： - 计数器溢出 - 设置 UG 位 - 从模式控制器产生的更新 1：如果使能了 UEV 只有计数器溢出产生 UEV 事件。
BIT[1]	UDIS	禁止更新 (Update disable) 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： - 计数器溢出 - 设置 UG 位 - 从模式控制器产生的更新具有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，缓存寄存器 (ARR、PSC、CCRx) 保持它们的值。如果设置了 UG 位则计数器和预分频器被重新初始化
BIT[0]	CEN	使能计数器 (Counter enable) 0：禁止计数器 1：使能计数器 *注 1：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。* *注 2：在单脉冲模式下，当发生更新事件时，CEN 被自动清除。*

16.4.3 TIM16 和 TIM17 控制寄存器 2 (TIM16_CR2 and TIM17_CR2)

TIM16_CR2 / TIM17_CR2			地址偏移	0x04		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	OIS1N	OIS1
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	CCDS	CCUS	-	CCPC
R/W	-	-	-	-	rw	rw	-	rw
复位值	-	-	-	-	0	0	-	0

BIT[15:10]	-	保留，保持复位值
BIT[9]	OIS1N	输出空闲状态 1(OC1N 输出) (Output Idle state 1) 0：当 MOE=0 时，死区后 OC1N=0； 1：当 MOE=0 时，死区后 OC1N=1。 *注：若设置了 LOCK(TIMx_BKR 寄存器 LOCK 位）级别 1、2 或 3 后，该位不能被修改。*
BIT[8]	OIS1	输出空闲状态 1(OC1 输出) (Output Idle state 1) 0：当 MOE=0 时，如果 OC1N 被使用，则死区后 OC1=0； 1：当 MOE=0 时，如果 OC1N 被使用，则死区后 OC1=1。 *注：已经设置了 LOCK(TIMx_BKR 寄存器 LOCK 位）级别 1、2 或 3 后，该位不能被修改。*
BIT[7:4]	-	保留，保持复位值，读取为 0
BIT[3]	CCDS	捕获/比较的 DMA 选择（捕获/比较 DMA selection） 0：当发生 CCx 事件时，送出 CCx 的 DMA 请求 1：当发生更新事件时，送出 CCx 的 DMA 请求
BIT[2]	CCUS	捕获/比较控制更新选择（捕获/比较 control update selection） 0：当捕获/比较控制位被预装载时（CCPC=1），只有通过设置 COMG 位才会被更新 1：当捕获/比较控制位被预装载时（CCPC=1），在设置 COMG 位或在 TRGI 上产生上升沿时都会被更新 *注：该位只对具有互补输出的通道起作用。*

BIT[1]	-	保留，保持复位值，读取为 0
BIT[0]	CCPC	捕获/比较预装载控制位（捕获/比较 preloaded control） 0：CCxE，CCxNE 和 OCxM 位不是预装载的； 1：CCxE，CCxNE 和 OCxM 位是预装载的；这些位只能在 COM 位置位时被更新。 *注：该位只对具有互补输出的通道起作用。*

16.4.4 TIM16 和 TIM17 DMA/中断允许寄存器（TIM16_DIER 和 TIM17_DIER）

TIM16_DIER / TIM17_DIER			地址偏移	0x0C		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	CC1DE	UDE
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	BIE	-	COMIE	-	-	-	CC1IE	UIE
R/W	rw	-	rw	-	-	-	rw	rw
复位值	0	-	0	-	-	-	0	0

BIT[15:10]	-	保留，保持复位值
BIT[9]	CC1DE	捕获/比较 1 DMA 请求使能 0：CC1 DMA 请求禁止 1：CC1 DMA 请求允许
BIT[8]	UDE	更新 DMA 请求使能 0：更新 DMA 请求禁止 1：更新 DMA 请求允
BIT[7]	BIE	刹车中断使能 0：刹车中断禁止 1：刹车中断允
BIT[6]	-	保留，保持复位值
BIT[5]	COMIE	COM 中断使能 0：COM 中断禁止 1：COM 中断允许
BIT[4:2]	-	保留，保持复位值
BIT[1]	CC1IE	捕获/比较 1 中断使能 0：CC1 中断禁止 1：CC1 中断允许
BIT[0]	UIE	更新中断使能 0：更新中断禁止 1：更新中断允许

16.4.5 TIM16 和 TIM17 状态寄存器（TIM16_SR 和 TIM17_SR）

TIM16_SR / TIM17_SR			地址偏移	0x10		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	CC1OF	-
R/W	-	-	-	-	-	-	rc_w0	-
复位值	-	-	-	-	-	-	0	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	BIF	-	COMIF	-	-	-	CC1IF	UIF
R/W	rc_w0	-	rc_w0	-	-	-	rc_w0	rc_w0
复位值	0	-	0	-	-	-	0	0

BIT[15:10]	-	保留，保持复位值
BIT[9]	CC1OF	捕获/比较 1 重复捕获标志 (Capture/Compare 1 overcapture flag) 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。 软件写 0 可清除该位。 0：无重复捕获产生 1：当 CC1IF 的状态已经为‘1’，计数器的值被捕获到 TIMx_CCR1 寄存器
BIT[8]	-	保留，保持复位值

BIT[7]	BIF	刹车中断标志 (Break interrupt flag) 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
BIT[6]	-	保留，保持复位值
BIT[5]	COMIF	COM 中断标志 (COM interrupt flag) 一旦产生 COM 事件（当捕获/比较控制位：CCxE、CCxNE、OCxM 已被更新） 该位由硬件置‘1’。 软件写零清‘0’。 0：无 COM 事件产生 1：COM 中断等待响应
BIT[4:2]	-	保留，保持复位值
BIT[1]	CC1IF	捕获/比较 1 中断标志 **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。由软件清‘0’。 0：无匹配发生 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配。 当 TIMx_CCR1 的值大于 TIMx_ APR 的值时，在向上计数模式计数器溢出条件下，CC1IF 位变高 **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’，它由软件清‘0’或通过读 TIMx_CCR1 清‘0’。 0：无输入捕获产生 1：计数器值被捕获至 TIMx_CCR1（在 IC1 上检测到与所选极性相同的边沿）
BIT[0]	UIF	更新中断标志 (Update interrupt flag) 当产生更新事件时该位由硬件置‘1’。它由软件清‘0’。 0：无更新事件产生； 1：更新中断等待响应。当下述寄存器被更新时该位由硬件置‘1’： - 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢时 （ 重复计数器=0 时产生更新事件 ）。 - 若 TIMx_CR1 寄存器的 URS=0、UDIS=0， 通过软件设置 TIMx_EGR 寄存器的 UG=1，对计数器 CNT 重新初始化时。

16.4.6 TIM16 和 TIM17 事件产生寄存器 (TIM16_EGR 和 TIM17_EGR)

TIM16_EGR / TIM17_EGR			地址偏移	0x14		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	BG	-	COMG	-	-	-	CC1G	UG
R/W	w	-	w	-	-	-	w	w
复位值	0	-	0	-	-	-	0	0

BIT[15:8]	-	保留，保持复位值
BIT[7]	BG	产生刹车事件 (Break generation) 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，若开启对应的中断和 DMA，则产生相应的中断和 DMA
BIT[6]	-	保留，保持复位值
BIT[5]	COMG	捕获/比较控制更新产生（Capture/Compare control update generation） 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：该位只对拥有互补输出的通道有效。*
BIT[4:2]	-	保留，保持复位值
BIT[1]	CC1G	捕获/比较 1 发生 (Capture/Compare 1 generation)

		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 1 上产生一个捕获/比较事件 **若通道 CC1 配置为输出：** CC1IF 被置 1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。 **若通道 CC1 配置为输入：** 当前的计数器值被捕获至 TIMx_CCR1 寄存器；CC1IF 被置 1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。若 CC1IF 已经为 1，则 CC1OF 被置 1。
BIT[0]	UG	产生更新事件 (Update generation) 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（寄存器）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）。向上计数模式则计数器被初始化为‘0’

16.4.7 TIM16 和 TIM17 捕获/比较模式寄存器 1（TIM16_CCMR1 和 TIM17_CCMR1）

TIM16_CCMR1 / TIM17_CCMR1			地址偏移	0x18		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	OC1M[2:0]			OC1PE	-	CC1S[1:0]	
	IC1F[3:0]				IC1PSC[1:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

输出比较模式

BIT[15:7]	-	保留，保持复位值
BIT[6:4]	OC1M	输出比较 1 模式 (Output compare 1 mode) 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1 值。OC1REF 是高电平有效，而 OC1 的有效电平取决于 CC1P 位。 000：冻结。输出比较寄存器 TIMx_CCR1 与计数器 TIMx_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（TIMx_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 TIMx_CNT 的值与捕获/比较寄存器 1（TIMx_CCR1）相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_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 1，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为有效电平，否则为无效电平； 111：PWM 模式 2，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平； *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。*
BIT[3]	OC1PE	输出比较 1 预装载使能 (Output compare 1 preload enable) 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即生效。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被传送至当前寄存器中。 *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：仅在单脉冲模式下（TIMx_CR1 寄存器的 OPM=1），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。*

BIT[2]	-	保留，保持复位值
BIT[1:0]	CC1S	捕获/比较 1 选择 (Capture/Compare 1 selection) 这 2 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：保留 11：保留 *注：CC1S 仅在通道关闭时（TIMx_CCER 寄存器的 CC1E= 0 ）才是可写的。*

输入捕捉模式

BIT[15:8]	-	保留，保持复位值
BIT[7:4]	IC1F	输入捕获 1 滤波器 (Input capture 1 filter) 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 f_{DTS} 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：当 ICxF[3:0]=1、2 或 3 时，公式中的 fDTS 由 CK_INT 替代。*
BIT[3:2]	IC1PSC	输入捕获 1 预分频（Input capture 1 prescaler） 这 2 位定义了 CC1 输入 (IC1) 的预分频系数。 一旦 CC1E= ‘0’ (TIMx_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。
BIT[1:0]	CC1S	捕获/比较 1 选择（Capture/Compare 1 selection） 这 2 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：保留 11：保留 *注：CC1S 仅在通道关闭时（TIMx_CCER 寄存器的 CC1E=0 ）才是可写的。*

注：通道可用于输入（ 捕获模式 ）或输出（ 比较模式 ），通道的方向由相应的 CCxS 位定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能，ICxx 描述了通道 在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

16.4.8 TIM16 和 TIM17 捕获/比较使能寄存器 (TIM16_CCER 和 IM17_CCER)

TIM16_CCER / IM17_CCER			地址偏移	0x20		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	CC1NP	CC1NE	CC1P	CC1E
R/W	-	-	-	-	rw	rw	rw	rw

复位值	-	-	-	-	0	0	0	0

BIT[15:4]	-	保留，保持复位值
BIT[3]	CC1NP	捕获/比较 1 输出极性 . 0：OC1N 高有效 1：OC1N 低有效 *注：一旦 LOCK 级别设为 2 或 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。*
BIT[2]	CC1NE	捕获/比较 1 互补输出使能（Capture/Compare 1 complementary output enable） 0：关闭，OC1N 无效 1：开启，OC1N 输出到相关输出引脚的信号取决于 MOE, OSSI, OSSR, OIS1, OIS1N 和 CC1E 位
BIT[1]	CC1P	捕获/比较 1 输出极性 (Capture/Compare 1 output Polarity) **通道 CC1 配置为输出 ：** 0：OC1 高有效 1：OC1 低有效 **通道 CC1 配置为输入 ：** CC1NP/CC1P 用于选择捕获信号 TI1FP1 和 TI2FP1 的极性 00：不反相/上升沿：捕获发生在 TIxFP1 的上升沿，不反相 00：反相/下降沿：捕获发生在 TIxFP1 的下降沿，反相 10：保留，不使用此配置 11：不反相/上升和下降沿：捕获发生在 TIxFP1 的上升沿和下降沿，不反相。 *注：一旦 LOCK 级别设为 2 或 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。*
BIT[0]	CC1E	捕获/比较 1 输出使能 (Capture/Compare 1 output enable) **CC1 通道配置为输出：** 0： 关闭，OC1 禁止输出。 1： 开启，输出到对应的输出引脚的 OC1 信号取决于 MOE, OSSI, OSSR, OIS1, OIS1N 和 CC1E 位。 **CC1 通道配置为输入：** 该位决定了计数器的值是否能捕获入 TIMx_CCR1 寄存器。 0：捕获禁止； 1：捕获使能。

表：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(1)	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出禁止 （ 不由定时器驱动 ） OCx=0, OCx_EN=0	输出禁止 （ 不由定时器驱动 ） OCxN=0 OCxN_EN=0
		0	0	1	输出禁止 （ 不由定时器驱动 ） 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 （ 不由定时器驱动 ） OCxN=0 OCxN_EN=0
		0	1	1	OCxREF + 极性 + 死区 OCx_ EN=1	OCxREF 互补(非 OCxREF) +极性 + 死区 OCxN_EN=1
		1	0	0	输出禁止 （ 不由定时器驱动 ） OCx=CCxP OCx_EN=0	输出禁止 （ 不由定时器驱动 ） OCxN=CCxNP OCxN_EN=0
		1	0	1	Off-State （输出使能但无效态 ） 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	Off-State （ 输出使能但无效状态 ） OCxN=CCxNP

					OCx_EN=1	OCxN_EN=1
		1	1	1	OCxREF + 极性 + 死区 OCx_ EN=1	OCxREF 互补(非 OCxREF) +极性 + 死区 OCxN_EN=1
0	0	X	0	0	输出禁止 (不由定时器驱动) OCx=CCxP OCx_EN=0	输出禁止 (不由定时器驱动) OCxN=CCxNP OCxN_EN=0
	0		0	1	输出禁止 (不由定时器驱动)	
	0		1	0	异步 : OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0	
	0		1	1	若时钟存在：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		0	0	输出禁止 (不由定时器驱动) OCx=CCxP OCx_EN=0	输出禁止 (不由定时器驱动) OCxN=CCxNP OCxN_EN=0
	1		0	1	Off-State (输出使能但无效状态)	
	1		1	0	异步 : OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1	
	1		1	1	若时钟存在：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。	

注 1：当一个通道的两个输出都没有使用时 (CCxE = CCxNE = 0), OISx, OISxN, CCxP 和 CCxNP 必有保持清零。

注 2：连接到互补通道（OCx 和 OCxN）的外部 I/O 引脚状态，取决于 OCx 和 OCxN 通道状态和 GPIO 寄存器和 AFIO 寄存器。

16.4.9 TIM16 和 TIM17 计数器 (TIM16_CNT 和 TIM17_CNT)

TIM16_CNT / TIM17_CNT			地址偏移	0x24		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CNT[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CNT[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	CNT[15:0]	计数器值

16.4.10 TIM16 和 TIM17 预分频寄存器 (TIM16_PSC 和 TIM17_PSC)

TIM16_PSC / TIM17_PSC			地址偏移	0x28		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	PSC[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	PSC[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	PSC[15:0]	预分频值 计数器的时钟频率 CK_CNT 等于 fCK_PSC/(PSC[15:0]+1)。 PSC 包含了当更新事件(包括使用 TIMx_EGR 寄存器的 UG 或者当配置为复位模式时通过触发控制器清除计数器）产生时装入当前预分频器寄存器的值。

16.4.11 TIM16 和 TIM17 自动重载寄存器 (TIM16_ARR 和 TIM17_ARR)

TIM16_ARR / TIM17_ARR			地址偏移	0x2C		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	ARR[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	ARR[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	ARR[15:0]	自动重装载的值 (Prescaler value) 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<16.3.1 时基单元>章节：有关 ARR 的更新和动作。 当自动重载的值为空时，计数器不工作。

16.4.12 TIM16 和 TIM17 重复计数寄存器 (TIM16_RCR 和 TIM17_RCR)

TIM16_RCR / TIM17_RCR			地址偏移	0x30		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	REP[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:8]	-	保留，保持复位值
BIT[7:0]	REP[7:0]	重复计数器的值 (Repetition counter value) 预装载寄存器被使能后，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)；如果允许产生更新中断，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 U_RC 发生时才重载 REP 值， 因此对 TIMx_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。

16.4.13 TIM16 和 TIM17 捕获/比较寄存器 1 (TIM16_CCR1 和 TIM17_CCR1)

TIM16_CCR1 / TIM17_CCR1			地址偏移	0x34		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CCR1[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CCR1[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	CRR1[15:0]	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 TIMx_CCMR1 寄存器（OC1PE 位）中未选择预装载特性，写入的数值会被立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。当前捕获/比较寄存器参与同计数器 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 **若 CC1 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。

16.4.14 TIM16 和 TIM17 刹车与死区时间寄存器 (TIM16_BDTR 和 TIM17_BDTR)

TIM16_BDTR / TIM17_BDTR			地址偏移	0x44		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]	

R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	DTG[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15]	MOE	主输出使能 (Main output enable) 一旦刹车输入有效，该位被硬件异步清‘0’。根据 AOE 位的设置值，该位可以由软件清‘0’或被自动置 1。它仅对配置为输出的通道有效。 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 (TIMx_CCER 寄存器的 CCxE、CCxNE 位)，则开启 OC 和 OCN 输出。 *有关 OC/OCN 使能的细节，参见 16.4.8 章节，<TIM16 和 TIM17 捕获/比较使能寄存器(TIMx_ CCER)>。*
BIT[14]	AOE	自动输出使能 (Automatic output enable) 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被自动置’1’(如果刹车输入无效)。 *注：一旦 LOCK 级别 (TIMx_BDTR 寄存器中的 LOCK 位)设为‘1’，则该位不能被修改。*
BIT[13]	BKP	刹车输入极性 (Break polarity) 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 (TIMx_BDTR 寄存器中的 LOCK 位)设为‘1’，则该位不能被修改。* *注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。*
BIT[12]	BKE	刹车使能 (Break enable) 0: 刹车输入禁止 (BRK 和 CCS 时钟失效事件) 1; 刹车输入允许 (BRK 和 CCS 时钟失效事件) *注：一旦 LOCK 级别 (TIMx_BDTR 寄存器中的 LOCK 位)设为’1’，则该位不能被修改。* *注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。*
BIT[11]	OSSR	运行模式下“关闭状态”选择 (Off-state selection for Run mode) 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 有关 OC/OCN 使能的细节，参见 16.4.8 章节，<TIM16 和 TIM17 捕获/比较使能寄存器(TIMx_ CCER)>。 0：当定时器不工作时，禁止 OC/OCN 输出 (OC/OCN 使能输出信号 =0) 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 使能并输出无效电平，然后置 OC/OCN 使能输出信号 =1 *注：一旦 LOCK 级别 (TIMx_BDTR 寄存器中的 LOCK 位)设为‘2’，则该位不能被修改。*
BIT[10]	OSSI	运行模式下“空闲状态”选择 (Off-state selection fo Idle mode) 该位用于当 MOE=0 时通道为输出。 有关 OC/OCN 使能的细节，参见 16.4.8 章节，<TIM16 和 TIM17 捕获/比较使能寄存器(TIMx_ CCER)>。 0：当定时器不工作时，禁止 OC/OCN 输出 (OC/OCN 使能输出信号 =0 1：当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 被强制输出空闲电平，置 OC/OCN 使能输出信号 =1 *注：一旦 LOCK 级别 (TIMx_BDTR 寄存器中的 LOCK 位)设为’2’，则该位不能被修改。*
BIT[9:8]	LOCK[1:0]	锁定设置 (Lock configuration) 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 TIMx_BDTR 寄存器的 DTG、BKE、BKP、AOE 位和 TIMx_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， CC 极性位是 TIMx_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，CC 控制位是 TIMx_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 TIMx_BDTR 寄存器，则其内容冻结直至复位。*
BIT[7:0]	DTG[7:0]	死区发生器设置 (Dead-time generator setup) 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} ； 举例：若 T_{DTS}=125ns(8MHZ)，可能的死区时间为： 0~15875ns，@步长为 125ns； 16us~31750ns，@步长为 250ns； 32us~63us，@步长为 1us； 64us~126us，@步长为 2us； *注：一旦 LOCK 级别（TIMx_BDTR 寄存器中的 LOCK 位）设为 1 、2 或 3，则不能修改这些位。*

注：根据锁定设置，AOE、BKP、BKE、OSSI、OSSR 和 DTG[7:0] 位均可被写保护，有必要在第一次写入 TIMx_BDTR 寄存器时对它们进行全部配置。

16.4.15 TIM16 和 TIM17 DMA 控制寄存器 (TIM16_DCR 和 TIM17_DCR)

TIM16_DCR / TIM17_DCR			地址偏移	0x48		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	DBL[4:0]				
R/W	-	-	-	rw	rw	rw	rw	rw
复位值	-	-	-	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	DBA[4:0]				
R/W	-	-	-	rw	rw	rw	rw	rw
复位值	-	-	-	0	0	0	0	0

BIT[15:13]	-	保留，保持复位值
BIT[12:8]	DBL[4:0]	DMA 连续传送长度 (DMA burst length) 这 5 位定义了 DMA 在连续模式下的传送长度 (当对 TIMx_DMAR 寄存器进行读或写时，定时器则进行一次连续传送)， 00000：1 次传输 00001：2 次传输 00010：3 次传输 ..... 10001：18 次传输
BIT[7:5]	-	保留，保持复位值
BIT[4:0]	DBA[4:0]	DMA 基地址 (DMA base address) 这 5 位定义了 DMA 传送的基地址 (当对 TIMx_DMAR 寄存器进行读或写时)，DBA 定义为从 TIMx_CR1 寄存器所在地址开始的偏移量： 例如： 00000：TIMx_CR1 00001：TIMx_CR2 00010：TIMx_SMCR ..... 例：要完成如下的传输：DBL = 7 ，DBA = TIMx_CR1 此时传送从 TIMx_CR1 的地址开始，对连续 7 个寄存器进行 DMA 读写操作。

16.4.16 TIM16 和 TIM17 DMA 全传输地址寄存器 (TIM16_DMAR 和 TIM17_DMAR)

TIM16_DMAR / TIM17_DMAR			地址偏移	0x4C		复位值	0x0000 0000	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	DMAB[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw

复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	DMAB[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	DMAB[15:0]	DMA 并发（连续）传送寄存器 对 TIMx_DMAR 寄存器的读或写会导致对以下地址所在寄存器的访问： （TIMx_CR1 地址）+ DBA + (DMA 索引)， 其中: “TIMx_CR1 地址”是控制寄存器 1(TIMx_CR1) 所在的地址； “DBA”是 TIMx_DCR 寄存器中定义的 DMA 基地址； “DMA 索引”是由 DMA 自动控制的偏移量， 它取决于 TIMx_DCR 寄存器中定义的 DBL。

举例：如何使用 DMA 并发特性

此例中，定时器 DMA 并发特性用于更新 CCRx 寄存器(x = 2，3，4)的内容，DMA 将半字传输到 CCRx 寄存器中。

操作步骤如下：

1、 配置对应的 DMA 通道如下：

- DMA 通道外设地址是 DMAR 寄存器地址
- DMA 通道存储器地址是 RAM 中缓存区地址，包含要由 DMA 转移到 CCRx 寄存器的数据
- 传送数据数目=3（见下面注释）
- 循环模式禁用

2、 配置 DCR 寄存器如下：

DBL=3，DBA=0xE

3、 启用 TIMx 更新 DMA 请求(在 DIER 寄存器中置位 UDE 位)

4、 使能 TIMx

5、 使能 DMA 通道

注：这个例子适用于每个 CCRx 寄存器只更新一次的情况。例如，如果每个 CCRx 寄存器要更新两次，那么要传输的数据数量应该是 6。让我们以 RAM 中包含 data1、data2、data3、data4、data5 和 data6 的缓冲区为例。数据传输到 CCRx 寄存器如下所示：在第一次更新 DMA 请求中，data1 被转移到 CCR2，data2 被转移到 CCR3，data3 被转移到 CCR4，在第二次更新 DMA 请求中，data4 被转移到 CCR2，data5 被转移到 CCR3，data6 被转移到 CCR4。

17 独立看门狗（IWDG）

17.1 概述

本芯片内嵌一个独立看门狗模块，用于检测和解决某些软件故障问题，当看门狗计数器达到预定门限，将触发一个系统复位请求。

独立看门狗时钟由特定的 LSI 时钟驱动，主时钟失效情况下，仍可以保持正常工作。

IWDG 最适用于：在主程序外，看门狗作为独立进程的应用，而且对时钟精度要求较低。

17.2 特性

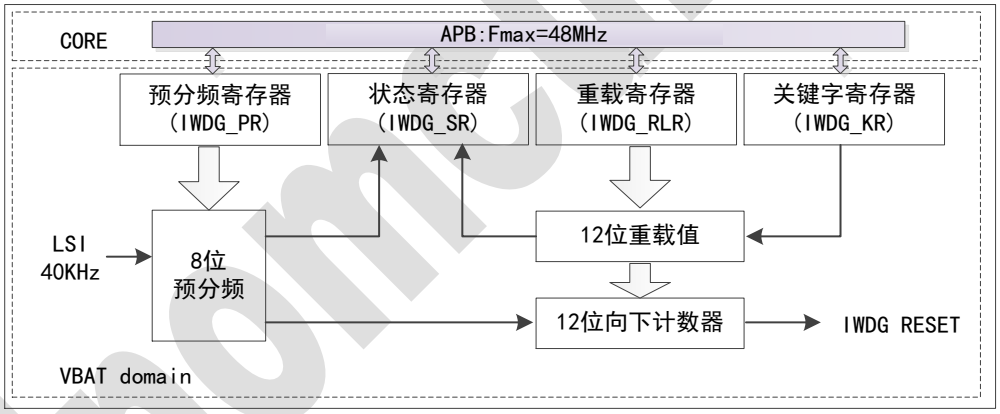
- ✧ 自由运行的向下计数器
- ✧ 由独立的低频 RC 振荡器驱动，在 standby 和 stop 模式下可以工作
- ✧ 复位条件（看门狗需开启）
- - 向下计数器计数至 0x000，产生复位
- - 向下计数器值加载值处于窗口值之外，产生复位

17.3 功能描述

当向独立看门狗的关键字寄存器(IWDG_KR)写入启动指令 0x0000CCCC 的时候，看门狗计数器开始由复位值 0xFFF 向下计数。当计数值达到 0x000 的时候由独立看门狗发出复位信号（IWDG reset）。

任何时候将关键字 0x0000AAAA 写到 IWDG_KR 寄存器中，都会使得 IWDG_RLR 寄存器中的值被重加载到看门狗计数器中，从而阻止即将发生的复位动作。

IWDG 模块框图



17.3.1 窗口功能

在 IWDG_WINR 寄存器中设置适当的窗口值，IWDG 也能够工作在窗口看门狗模式下。

如果重加载操作时，看门狗计数器的值高于窗口寄存器 (IWDG_WINR)的值，将发生复位。

IWDG_WINR 的默认值是 0x00000FFF，所以如果没有改写它，那么窗口功能选项默认是关闭的。

窗口值一旦改变，立即就会引起看门狗计数器的一次重加载动作，将其加载为 IWWDG_RLR 中所设置的值，从而一定程度上延缓目前到下次复位所需的时间周期。

窗口功能使能时 IWDG 配置：

1、 将 0x0000CCCC 写到 IWDG_KR 寄存器，使能 IWDG。
2、 向 IWDG_KR 寄存器写 0x00005555 打开寄存器访问许可。
3、 向 IWDG_PR 写 0～7 的值，以配置 IWDG 的预分频器。
4、 配置重加载寄存器 (IWDG_RLR)。
5、 等待状态寄存器 IWDG_SR 的值更新为 0x00000000。
6、 配置窗口寄存器 IWDG_WINR。 这将会引起自动将 IWDG_RLR 的值更新到看门狗计数器。

注： 当 IWDG_SR 的值为 0x00000000 时，写窗口值的动作会使 RLR 的值更新至计数器。

窗口功能关闭时 IWDG 配置：

1、 将 0x0000CCCC 写到 IWDG_KR 寄存器，使能 IWDG。
2、 向 IWDG_KR 寄存器写 0x00005555 打开寄存器访问许可。

3、 向 IWDG_PR 写 0～7 的值，以配置 IWDG 的预分频器。
4、 配置重加载寄存器 (IWDG_RLR)。
5、 等待状态寄存器 IWDG_SR 的值更新为 0x00000000。
6、 向 IWDG_KR 寄存器写 0x0000AAAA，更新 IWDG_RLR 的值到看门狗计数器中。

17.3.2 硬件看门狗

如果在 OPTION 字节中打开了"硬件看门狗"功能，那么在上电的时候看门狗就被自动打开。
若没有在下述条件下喂狗（IWDG_KR 写入正确值），那么将会产生硬件复位请求：

- 在看门狗计数器计数结束前
- 向下计数的值在窗口范围内

17.3.3 Stop 和 standby 模式

在 STOP 和 standby 模式下，一旦运行，不能被停止。

17.3.4 寄存器访问保护

默认情况下，对 IWDG_PR、IWDG_RLR 和 IWDG_WINR 的写访问操作都是受保护的。若要修改这些寄存器，必须先向 IWDG_KR 写入 0x0000 5555 解锁。 如果写入别的值，将会打破解锁状态，寄存器的访问保护重新生效。也就是说，在做重加载操作的时候（向该寄存器写入 0x0000 AAAA）就属于这种情况。

状态寄存器（IWDG_SR）可以指示预分频器的更新、看门狗计数器的重加载或窗口值的更新。

17.3.5 Debug 模式

当微控制器进入调试模式(内核停止)时，IWDG 计数器要么继续正常工作，要么停止，这取决于 DBG 模块的 DBG_IWDG_STOP 配置位。

17.4 相关寄存器

17.4.1 寄存器汇总表

基址：0x4000 3000				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**IWDG_KR**	0x0000 0000	独立看门狗关键字寄存器	17.4.2
0x04	**IWDG_PR**	0x0000 0000	独立看门狗预分频器寄存器	17.4.3
0x08	**IWDG_RLR**	0x0000 0FFF	独立看门狗重加载寄存器	17.4.4
0x10	**IWDG_WINR**	0x0000 0FFF	独立看门狗窗口寄存器	17.4.5

17.4.2 独立看门狗关键字寄存器（IWDG_KR）

IWDG_KR			地址偏移	0x00		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	KEY[15:8]-							
R/W	w							
复位值	0000 0000							
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	KEY[7:0]							
R/W	w							
复位值	0000 0000							

BIT[15:0]	KEY[15:0]	关键值（只写，读的话会是 0x0000） 这些位必须周期性的由软件写入 0xAAAA，否则当计数器向下计数到 0 的时候

		会产生硬件复位请求。 写入 0x5555 会使能对 IWDG_PR,IWDG_RLR 和 IWDG_WINR 三个寄存器的访问许可。(参见章节 17.3.4) 写入 0xCCCC 启动看门狗(除非已经由选项字节在上电的时候就启动了它)

17.4.3 独立看门狗预分频器寄存器(IWDG_PR)

IWDG_PR			地址偏移	0x00		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	PR[2:0]		
R/W	-	-	-	-	-	rw		
复位值	-	-	-	-	-	000		

注:读这个寄存器会返回 VBAT 供电区域的分频器的值。如果处于写保护状态下。这个值不一定是最新的。所以从这个地方读数据的时候要保证 IWDG_SR 寄存器中的 PVU 位为 0 才行。

BIT[2:0]	PR[2:0]	预分频器 这些位平时处于写保护状态,参见章节 17.3.4 由软件写入,用来选择对输入时钟的预分频系数。 为了能够改变预分频器的分频系数, IWDG_SR 寄存器中的 PVU 位必须先清零。 000: 4 分频 001: 8 分频 010: 16 分频 011: 32 分频 100: 64 分频 101: 128 分频 110: 256 分频 111: 256 分频

17.4.4 独立看门狗重加载寄存器(IWDG_RLR)

IWDG_RLR			地址偏移	0x08		复位值	0x0000 0FFF	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	RL[11:8]			
R/W	-	-	-	-	rw			
复位值	-	-	-	-	1111			
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	RL[7:0]							
R/W	rw							
复位值	1111 1111							

注：读这个寄存器会返回 VBAT 供电区域的分频器的值。如果处于写保护状态下。这个值不一定是最新的。所以从这个地方读数据的时候要保证 IWDG_SR 寄存器中的 RVU 位为 0 才行。

BIT[11:0]	RL[11:0]	看门狗计数器重置值 这些位平时处于写保护状态，参见章节 17.3.4 这个值是由软件来设置的，并且每次向 IWDG_KR 寄存器写入 0xAAAA 的时候，这个值会被更新到看门狗计数器中， 如果想延时长一点，这个值就该大一些。 因为看门狗计数器正是从这个值开始向下计数。 定时的长度是由这个值和预分频器的设置值来共同决定的。 参见 datasheet 中关于超时的数据。 要想改变这个重加载值，必须先保证 IWDG_SR 中的 RVU 位为 0

17.4.5 独立看门狗窗口寄存器（IWDG_WINR）

IWDG_WINR			地址偏移	0x10		复位值	0x0000 0FFF	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	WIN[11:8]			
R/W	-	-	-	-	rw			
复位值	-	-	-	-	1111			
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	WIN[7:0]							
R/W	rw							
复位值	1111 1111							

注 1：如果重加载设置、预分频设置和窗口设置都已经生效，那么再次改变它们前一定要确认相关的标志位为零，那表明他们没有处于正在被加载的过程中。更新这些设置之后却不需要等着三个标志位清零，除非马上要进入低功耗模式。

注 2：读这个寄存器会返回 VBAT 供电区域的值。如果刚刚做过写操作就读，这个值不一定是对的。所以从这个地方读数据的时候要保证 IWDG_SR 寄存器中的 WVU 位为 0 才行。

BIT[11:0]	RL[11:0]	看门狗计数器窗口值 这些位平时处于写保护状态，参见章节 20.3.3 这些位包含的是窗口值和向下计数器的比较上限。 为了阻止复位信号的产生，必须在向下计数器递减到窗口值和 0 之间的某个值时重加载它。要想改变这个窗口值，必须先保证 IWDG_SR 中的 WVU 位为 0。

18 窗口看门狗（WWDG）

18.1 概述

窗口看门狗(WWDG) 内有一个 7 位的向下计数器，并可以设置成自由运行，用来检测软件故障的发生，这些故障通常是由于外部干扰或不可预见的逻辑条件引起的，使应用程序背离正常的运行顺序。看门狗电路在程序设定的时间周期到期时产生 MCU 复位，除非程序在 T6 位被清零（递减计数 0x40 到 0x3F）之前刷新向下计数器的内容。如果 7 位向下计数器的值(在控制寄存器中)在达到窗口寄存器值之前刷新，也会产生 MCU 复位。这意味着计数器必须在一个有限的窗口内刷新。

WWDG 时钟由 APB 时钟预分频驱动，并有一个可配置的时间窗口，用以检测异常延迟和提前的应用行为。

WWDG 最适用于那些需要看门狗在一个精确的时间窗口内做出响应的应用。

18.2 特性

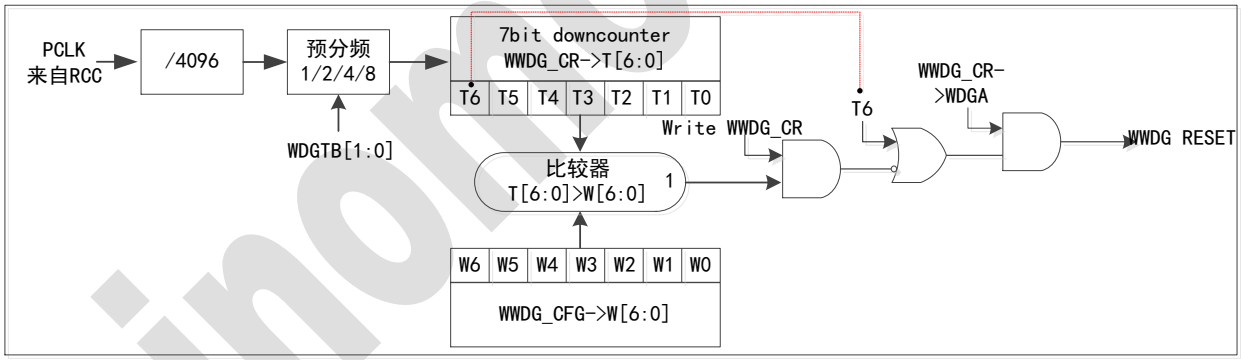
- ✧ 可编程的自由运行向下计数器
- ✧ 复位条件（看门狗需启动）
- - 向下计数器的值小于 0x40，发生复位
- - 向下计数器重载值在窗口范围外，发生复位
- ✧ 提前唤醒中断（EWI）：若看门狗启动，当向下计数器值等于 0x40 时，产生 EWI 中断

18.3 功能描述

若看门狗被启动 (WWDG_CR 寄存器中的 WDGA 位被置‘1’)， 并且当 7 位 (T[6:0]) 递减计数器从 0x40 翻转到 0x3F (T6 位清零) 时，则产生一个复位。如果软件在计数器值大于窗口寄存器中的数值时重载计数器，也将产生一个复位。

应用程序在正常运行过程中必须定期地写入 WWDG_CR 寄存器以防止 MCU 发生复位。写操作必须在计数器值小于窗口寄存器的值且大于 0x3F 时进行。储存在 WWDG_CR 寄存器中的数值必须在 0xFF 和 0xC0 之间。

WWDG 框图



18.3.1 启动看门狗

WWDG 复位后保持关闭状态，设置 WWDG_CR 寄存器的 WDGA 位开启看门狗，一旦开启，除非发生复位，否则不能被关闭。

18.3.2 控制向下计数器

WWDG 的向下计数器处于自由运行状态，即使看门狗处于关闭状态，计数器仍保持递减计数。当看门狗被启动时，T6 位必须置位，以防止产生复位。

T[5:0]位包含了看门狗产生复位前时间延时增量。定时在一个最小值和最大值之间变化，这是因为写入 WWDG_CR 寄存器时，预分频值是未知的。

配置寄存器 (WWDG_CFR)中包含窗口的上限值：要避免产生复位，递减计数器必须在其值小于窗口寄存器的数值并且大于 0x3F 时被重新装载，<超时设置章节>的看门狗时序图描述了窗口寄存器的工作过程。

注：T6 位可以用作产生一个软件复位（置位 WDGA 位，并清零 T6 位）。

18.3.3 提前唤醒中断

如果在实际复位产生之前必须进行特定的安全操作或数据记录，可以用提前唤醒中断（EWI）。

设置 WWDG_CFR 寄存器中的 EWI 位开启该中断。在复位之前当递减计数器到达 0x40 时，则产生此中断，同时可以用相应的中断服务程序(ISR) 来触发特定的行为（例如通信或数据记录）。

在某些应用中，EWI 中断可以用来管理软件系统检测和系统恢复/故障弱化，但不产生 WWDG 复位。在这种情况下，相应的中断服务程序（ISR）将重加载 WWDG 计数器，以避免 WWDG 复位，然后触发所需的操作。

EWI 中断通过将'0'写入 WWDG_SR 寄存器的 EWIF 位来清除。

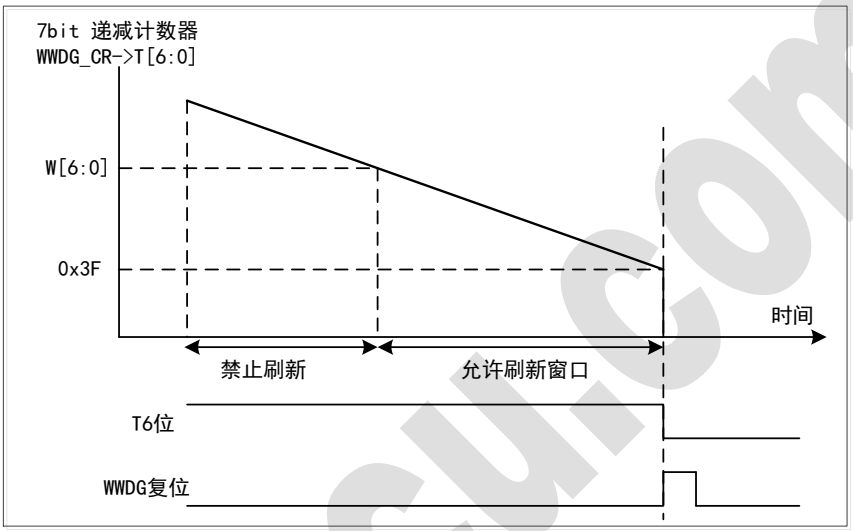
注：当提前唤醒中断（EMI）无法被服务时，例如，由于系统锁定在更高优先级任务，最终将产生 WWDG 复位。

18.3.4 超时设置

可以使用下述公式计算窗口看门狗超时时间。

注：当写入 WWDG_CR 寄存器时，必须置 T6 位为 '1' 以避免产生复位。

窗口看门狗时序图



计算超时公式如下：

$$t_{WWDG} = t_{PCLK} \times 4096 \times 2^{WDGTB[1:0]} \times (T[5:0]+1) \quad (ms)$$

其中，

t_{WWDG}：WWDG 超时时间

t_{PCLK}：APB 时钟周期，单位 ms

4096：内部除法器固定值

举例：APB 时钟 48MHz，WDGTB[1:0]为 3，T[5:0]为 63

t_{WWDG} =1/48000x4096x2^{3}x（63+1）=43.69ms

18.3.5 Debug 模式

当微控制器进入调试模式(Cortex®-M0 内核停止)时，WWDG 计数器要么继续正常工作，要么停止，这取决于 DBG 模块中的 DBG_WWDG_STOP 配置位。更多细节，请参见调试支持章节

18.4 相关寄存器

18.4.1 寄存器汇总表

基址：0x4000 3400				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**WWDG_CR**	0x0000 007F	窗口看门狗控制寄存器	18.4.2
0x04	**WWDG_CFR**	0x0000 007F	窗口看门狗配置寄存器	18.4.3
0x08	**WWDG_SR**	0x0000 0000	窗口看门狗状态寄存器	18.4.4

18.4.2 窗口看门狗控制寄存器（WWDG_CR）

WWDG_CR			地址偏移	0x00		复位值	0x0000 007F	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-

	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	WDGA	T[6:0]						
R/W	rs	rw						
复位值	0	111 1111						

BIT[7]	WDGA	激活位 此位由软件置’1’，但仅能由硬件在复位后清’0’。当 WDGA=1 时，看门狗可以产生复位。 0：禁止看门狗 1：启用看门狗
BIT[6:0]	T[6:0]	T[6:0]: 7-bit 计数器 (MSB 到 LSB) 这些位用来存储看门狗的计数器值。每 (4096x2$^{WDGTB[1:0]}$) 个 PCLK 周期减 1。当计数器值从 0x40 变为 0x3F 时 (T6 变成 0)，产生看门狗复位。

18.4.3 窗口看门狗配置寄存器（WWDG_CFR）

WWDG_CFR			地址偏移	0x04		复位值	0x0000 007F	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	EWI	WDGTB[1]
R/W	-	-	-	-	-	-	rs	rw
复位值	-	-	-	-	-	-	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	WDGTB[0]	W[6:0]						
R/W	rw	rw						
复位值	0	111 1111						

BIT[9]	EWI	提前唤醒中断 此位若置’1’，则当计数器值达到 0x40，即产生中断。 此中断只能由硬件在复位后清除。
BIT[8:7]	WDGTB[1:0]	时基 预分频器的时基可以设置如下： 00：计时器时钟：PCLK/4096/1 01：计时器时钟：PCLK/4096/2 10：计时器时钟：PCLK/4096/4 11：计时器时钟：PCLK/4096/8
BIT[6:0]	W[6:0]	7-bit 窗口值 这些位包含了用来与递减计数器进行比较用的窗口值。

18.4.4 窗口看门狗状态寄存器（WWDG_SR）

WWDG_SR			地址偏移	0x08		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-

R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	-	EWIF
R/W	-	-	-	-	-	-	-	rc_w0
复位值	-	-	-	-	-	-	-	0

BIT[1]	EWIF	提前唤醒中断标志 当计数器的值达到 0x40 时，此位由硬件置’1’。它必须通过软件写’0’来清除。对此位写’1’无效。若中断未被启用，此位也会被置’1’。

19 实时时钟（RTC）

19.1 概述

RTC 和 5 个备份寄存器通过开关供电，当 VDD 电源存在时，该开关选择 VDD 供电，否则选择由 VBAT 引脚供电。备份寄存器由 5 个 32 位寄存器组成，用于在 VDD 电源不存在时存储 20 字节的用户应用数据。备份寄存器不会在系统复位或电源复位时复位，也不会在器件从待机模式唤醒时复位。

实时时钟(RTC)是一个独立的 BCD 定时器 / 计数器。RTC 模块拥有一个具有可编程报警中断功能的时间日历时钟。

RTC 模块拥有自动唤醒功能，用于管理所有的低功耗模式。

两个 32 位寄存器以 BCD 格式存储秒、分、时（12/24 小时制）、日（星期数）、日期、月和年。亚秒值同样可用二进制存储。

RTC 具有自动月份天数补偿功能，还包括夏令时间补偿。

单独使用一个 32 位寄存器存储可编程报警相关信息，包括亚秒、秒、分、时、日、日期。

对由晶体本身的频偏、温度漂移及其他原因引起的任何误差，可以利用 RTC 本身的数字校准功能进行修正。

上电复位后，所有 RTC 寄存器将被禁止访问，以防止意外的写操作。

当设备处于运行模式、低功耗模式，或复位状态，只要电压在工作范围内， RTC 将保持正常运行。

19.2 特性

RTC 是一个独立的 BCD 定时器/计数器。其主要特性如下：

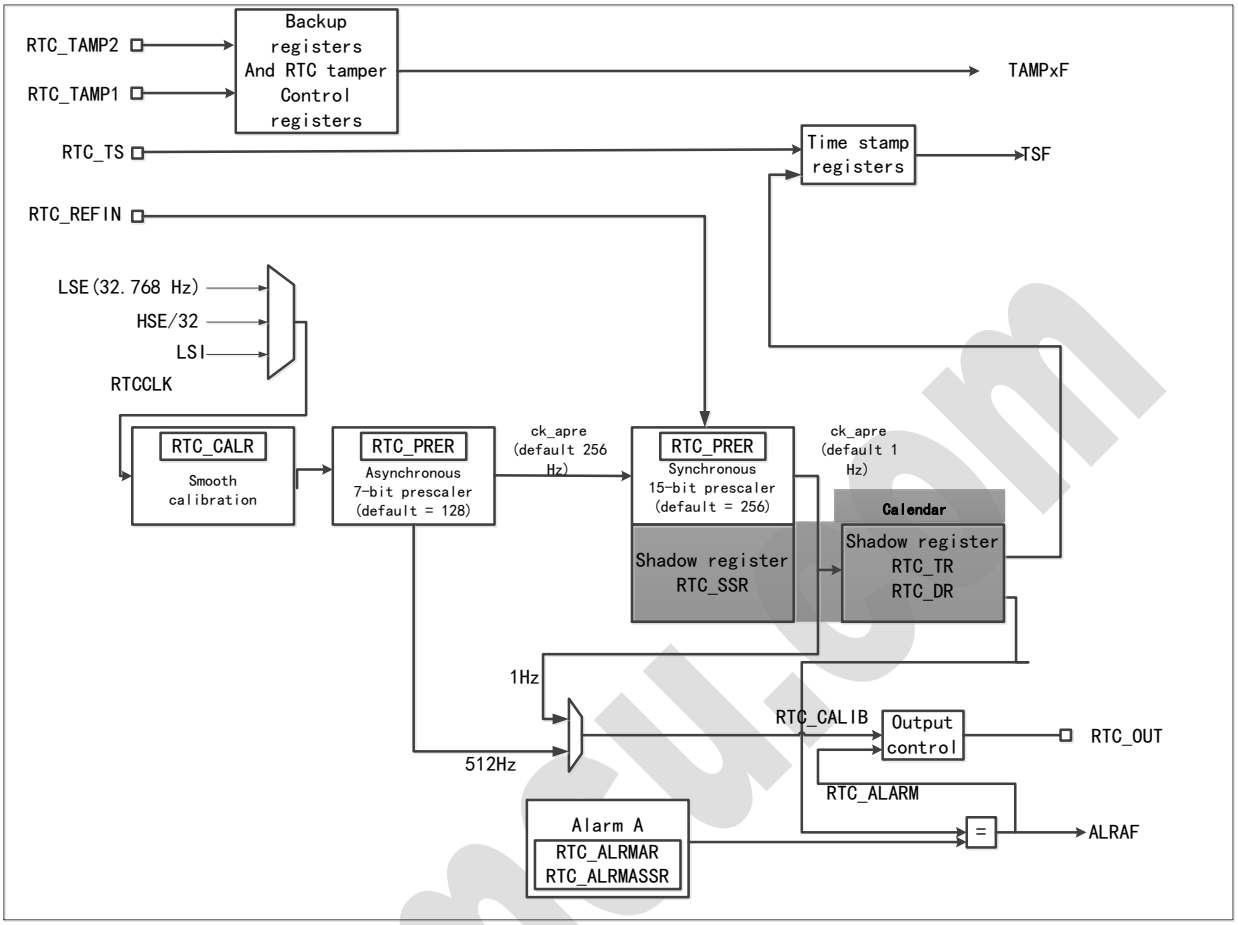
- ✧ 日历具有亚秒、秒、分、小时（12 或 24 格式）、星期几、日、月、年，格式为 BCD（二进码十进数）。
- ✧ 自动调整每月是 28、29（闰年）、30 还是 31 天。
- ✧ 可编程闹钟具有从停止和待机模式唤醒的能力。
- ✧ 可运行时纠正 1 到 32767 个 RTC 时钟脉冲。这可用于将 RTC 与主时钟同步。
- ✧ 数字校准电路具有 1 ppm 的分辨率，以补偿石英晶振的不准确性。
- ✧ 两个防篡改检测引脚具有可编程的滤波器。当检测到篡改事件时，MCU 可从停止及待机模式唤醒。
- ✧ 时间戳特性可用于保存日历内容。此功能可由时间戳引脚上的事件触发，或由篡改事件触发。当检测到时间戳事件时，MCU 可从停止及待机模式唤醒。
- ✧ 参考时钟检测：可使用更加精确的第二时钟源（50 或 60 Hz）来提高日历的精确度。

RTC 时钟源可为：

- ✧ 32.768 kHz 的外部晶振
- ✧ 谐振器或振荡器
- ✧ 内部低功耗 RC 振荡器 （典型频率为 40 kHz）
- ✧ 高速外部时钟的 32 分频

19.3 功能描述

RTC 模块框图



RTC 模块包括 ：

✧ 一个闹钟

✧ 两个侵入事件（来自 I/O）

- 侵入检测将擦除备份寄存器

✧ 一个时间戳事件输入（I/O）

✧ 侵入检测可产生时间戳事件

✧ 5 个 32bit 备份寄存器

✧ 复用功能输出 ：RTC_OUT 以下述两种形式中任意一种形式输出 ：

- RTC_CALIB: 512 Hz 或 1Hz 时钟输出 (用 32.768 kHz 的 LSE 的时候). 用 RTC_CR 寄存器中 COE 位来打开这个输出。
- RTC_ALARM: Alarm A. 这个输出用 RTC_CR 寄存器中 OSEL[1:0] 位来打开。

✧ 复用功能输入 ：

- RTC_TS ：时间戳事件
- RTC_TAMP1: 侵入事件检测 1
- RTC_TAMP2: 侵入事件检测 2
- RTC_REFIN: 50 或 60 Hz 参考时钟输入

19.3.1 由 RTC 控制的 GPIO

RTC_OUT, RTC_TS 和 RTC_TAMP1 映射到同一个引脚（PC13）。

通过 RTC_TAFCR 寄存器完成 RTC_ALARM 输出的选择， 其中 PC13VALUE 位用于选择 RTC_ALARM 输出是推挽模式或开漏模式。

当 PC13 不用作 RTC 复用功能时，可通过设置 RTC_TAFCR 中 PC13MODE 位将 PC13 强制为推挽输出模式，输出值由 PC13VALUE 位给出。此时，PC13 的推挽输出状态和值在待机模式下是可以保持的。

输出机制遵循固定的优先顺序（见下表）。

RTC pin PC13 配置

引脚配置和功能	RTC_ALARM 输出使能	RTC_CALIB 输出使能	RTC_TAMP1 输入使能	RTC_TS 输入使能	PC13MODE 位	PC13VALUE 位
RTC_ALARM output OD	1	无影响	无影响	无影响	无影响	0
RTC_ALARM output PP	1	无影响	无影响	无影响	无影响	1
RTC_CALIB output PP	0	1	无影响	无影响	无影响	无影响
RTC_TAMP1 input floating	0	0	1	0	无影响	无影响
RTC_TS and RTC_TAMP1 input floating	0	0	1	1	无影响	无影响
RTC_TS input floating	0	0	0	1	无影响	无影响
Output PP forced	0	0	0	0	1	PC13 输出数据值
Wakeup pin or Standard GPIO	0	0	0	0	0	无影响

注：OD：开漏，open drain 的缩写；PP：推挽，push-pull 的缩写。

当 PC14 和 PC15 不用作 LSE 振荡器时，可通过分别设置 RTC_TAFCR 寄存器中 PC14MODE 位和 PC15MODE 位将 PC14 和 PC15 强制为推挽输出模式， 输出值由 PC14VALUE 和 PC15VALUE 给出。此时，PC14 和 PC15 的推挽输出状态和值在待机模式下是可保持的。

输出机制遵循固定的优先顺序（见下表）

LSE pin PC14 配置

引脚配置和功能	RCC_BDCR 寄存器 LSEON 位	RCC_BDCR 寄存器 LSEBYP 位	PC14MODE 位	PC14VALUE 位
LSE oscillator	1	0	无影响	无影响
LSE bypass	1	1	无影响	无影响
Output PP forced	0	无影响	1	PC14 输出数据值
Standard GPIO	0	无影响	0	无影响

注：OD：开漏，open drain 的缩写；PP：推挽，push-pull 的缩写。

LSE pin PC15 配置

引脚配置和功能	RCC_BDCR 寄存器 LSEON 位	RCC_BDCR 寄存器 LSEBYP 位	PC15MODE 位	PC15VALUE 位
LSE oscillator	1	0	无影响	无影响
Output PP forced	0	无影响	1	PC15 输出数据值
Standard GPIO	0	无影响	0	无影响

注：OD：开漏，open drain 的缩写；PP：推挽，push-pull 的缩写。

19.3.2 时钟和预分频器

由时钟控制器从以下 3 种时钟中选择 RTC 时钟源（RTCCLK）：

- ✧ LSE 振荡器作为 RTC 时钟；
- ✧ LSI 振荡器作为 RTC 时钟；
- ✧ HSE 振荡器作为 RTC 时钟。

更多有关 RTC 时钟源的配置信息，请参考章节：复位和时钟控制（RCC）

一个可编程预分频器产生一个 1HZ 的时钟，用于更新日历。为最大限度地减少功耗，预分频器可分割成 2 个可编程预分频器。（见 RTC 框图）：

- ✧ 通过控制 RTC_PRER 寄存器中 PREDIV_A 位来配置 7 位异步预分频器。
- ✧ 通过控制 RTC_PRER 寄存器中 PREDIV_S 位来配置 15 位同步预分频器。

注 ：当所有分频器被启用时，建议将异步预分频器的值调高以最大限度地减少功率消耗

异步预分频器的分频系数设为 128，同步预分频器的分频系数设为 256，从而得到一个基于 32.768 kHz LSE 频率的 1 Hz（ck_spre）内部时钟频率。

最小分频系数为 1，最大分频系数为 222，相当于最大输入频率大约 4MHz。

fck_apre 满足：$f_{CK\_APRE} = \frac{f_{RTCCLK}}{PREDIV\_A+1}$

The ck_apre 时钟为二进制 RTC_SSR 亚秒递减计数器提供时钟。当值为 0 时，RTC_SSR 的值将被重置为 PREDIV_S 里的值。

fck_spre 满足：$f_{CK\_SPRE} = \frac{f_{RTCCLK}}{(PREDIV_S+1)\times(PREDIV_A+1)}$

19.3.3 实时时钟和日历

RTC 的日历时间寄存器和日期寄存器是通过影子寄存器访问的，该影子寄存器与 PCLK（APB clock）同步。 为避免同步等待时间，也可直接访问。

- ✧ RTC_SSR ：亚秒
- ✧ RTC_TR ：时间
- ✧ RTC_DR：日期

硬件会每两个 RTCCLK 时钟周期更新一次影子寄存器的日历值，并将 RTC_ISR 寄存器的 RSF 位置 1（见 22.6.4）。停止或待机模式下则不作这个更新，当退出上述两种模式后，影子寄存器将在最多两个 RTCCLK 周期后执行更新操作。

默认情况下，应用程序通过访问影子寄存器获取日历寄存器的内容。如需直接访问日历寄存器， 可通过设置 RTC_CR 寄存器的 BYPSHAD 控制位（默认为零）来实现。

在 BYPSHAD=0 模式下，如需读取 RTC_SSR, RTC_TR 或 RTC_DR 寄存器的内容，APB 时钟频率 (fAPB) 至少是 RTC 时钟频率 (fRTCCLK) 的 7 倍。

系统复位后，影子寄存器也将自动复位。

19.3.4 可编程闹钟

RTC 模块拥有可编程闹钟功能：Alarm A

设置 RTC_CR register 寄存器的 ALRAE 位，启动可编程闹钟功能。当日历中亚秒 ，秒，分，时，日期或星期与闹钟寄存器 RTC_ALRMASSR 和 RTC_ALRMAR 中设定的值匹配，ALRAF 由硬件置 1 。所有日历字段都可以通过 RTC_ALRMAR 寄存器的 MSKx 位和 RTC_ALRMASSR 寄存器的 MASKSSx 位独立的被选择为闹钟源。设置 RTC_CR 寄存器的 ALRAIE 位，会产生闹钟中断。

注 :当“秒”字段被选中时，为确保正常运行， RTC_PRER 寄存器中同步预分频器的分频 系数应 ≥ 3。

设置 RTC_CR 寄存器的 OSEL[0:1] 启动 Alarm A，可同步输出至 RTC_ALARM。RTC_ALARM 输出极性可由 RTC_CR 寄存器的 POL 位设置。

19.3.5 RTC 初始化及配置

访问 RTC 寄存器

RTC 寄存器是 32 位寄存器。除了 BYPSHAD=0 时对日历寄存器执行读操作外，APB 接口为其他对 RTC 寄存器的访问提供了 2 种等待状态。

RTC 寄存器的写保护

上电复位后，所有 RTC 寄存器将处于写保护状态。通过向写保护寄存器 RTC_WPR 写入指定关键字来启动 RTC 寄存器的写权限。

通过以下操作解除所有 RTC 寄存器（RTC_ISR[13:8],RTC_TAFCR 和 RTC_BKPxR 除外）的写保护。

1、向 RTC_WPR 寄存器写入‘0xCA’；

2、向 RTC_WPR 寄存器写‘0x53’。如果写入错误将重新激活写保护。

系统复位不影响写保护机制。

日历初始化及配置

按照以下顺序完成时间和日期值的初始化，包括时间格式和预分频器的配置：

1、 RTC_ISR 寄存器的 INIT 位置“1”，进入初始初始化模式。在该模式下日历计数器暂停运行， 此时可更新计数器的值。

2、 等待 RTC_ISR 寄存器的 INITF 位置“1”，确保已经正式进入初始化模式。由于时钟同步的延迟，该过程大约需要 2 个 RTCCLK 时钟周期。

3、 为了使日历计数器生成一个 1 Hz 的时钟，编写 RTC_PRER 寄存器中的分频系数。

4、 将初始时间和日期值加载到影子寄存器 (RTC_TR and RTC_DR)，通过 RTC_CR 寄存器的 FMT 位设置时间格式(12 小时制 /24 小时制）。

5、 清除 INIT 位的值退出初始化模式。日历计数器的实际值将会自动加载， 并在 4 个 RTCCLK 时钟周期后重新启动。

在完成上述一系列初始化操作后，日历将开始计时。

注 1：系统复位后，应用程序可读取 RTC_ISR 寄存器的 INITS 标志，从而检测日历是否已经被初始化。如果标志位为“0”说明“年”字段恢复成其上电复位后的默认值 0x00 ，此时日历没被初始化。

注 2：初始化后如果需要读日历，首先需要检测 RTC_ISR register 寄存器的 RSF 标志位是否为 1。

夏令时

通过 RTC_CR 寄存器的 SUB1H, ADD1H 和 BKP 位管理夏令时间。

通过设置 SUB1H 或 ADD1H 位，软件可在不通过初始化流程的情况下将日历中的时间单次增加 / 减少 1 小时。

此外，软件可通过 BKP 位记录该操作。

可编程闹钟

编程或更新可编程闹钟时，应遵循以下几点：

1、清除 RTC_CR 的 ALRAE 位禁用 Alarm A；

2、编程 Alarm A 寄存器 (RTC_ALRMASSR/RTC_ALRMAR)；

3、设置 RTC_CR 的 ALRAE 位重新启用 Alarm A。

注 :由于时钟同步的延迟，所有对 RTC_CR 寄存器的更新将在大约 2 个 RTCCLK 时钟周期后正式有效。

19.3.6 读日历寄存器

当 RTC_CR 寄存器的 BYPSHAD 控制位被清除时

为确保在安全同步机制下正常读 RTC 日历寄存器 (RTC_SSR, RTC_TR 和 RTC_DR)，APB1 时钟频率 (fPCLK) 应至少为 RTC 时钟频率 (fRTCCLK) 的 7 倍以上。

当 APB1 时钟频率低于 7 倍 RTC 时钟频率时，软件必须两次读取日历时间和日期寄存器。如果第二次读取 RTC_TR 返回的值与第一次返回值相同，说明返回值是正确的。否则需再次读取。任何情况下， APB1 时钟频率都必须大于 RTC 时钟频率。

日历寄存器的内容被复制到 RTC_TR 和 RTC_DR 影子寄存器中时，RTC_ISR 寄存器的 RSF 位被置位。复制操作每两个 RTCCLK 周期执行一次。为确保三者的值保持一致，读 RTC_ SSR 或 RTC_TR，锁定高阶日历影子寄存器中的值，直至 RTC_DR 中的值被读取。为避免软件在时间间隔少于 2 个 RTCCLK 周期的情况下多次访问日历，每次读日历 RSF 位应由软件清零，软件必须等待 RSF 位被置位后才能读 RTC_SSR，RTC_TR 和 RTC_DR 寄存器。

从低功耗模式（关机或待机）唤醒后，RSF 位应由软件清零，软件必须等待 RSF 位被再次置位后才能读 RTC_SSR，RTC_TR 和 RTC_DR 寄存器。

RSF 位应在唤醒后被清除，而不是进入低功耗模式前。

系统复位后，软件必须等待 RSF 位被置位后才能读 RTC_SSR，RTC_TR 和 RTC_DR 寄存器。事实上系统复位将导致影子寄存器复位至其默认值。

初始化（参考第 468 页：日历初始化及配置）后，软件必须等待 RSF 位被置位后才能读 RTC_SSR，RTC_TR 和 RTC_DR 寄存器。

同步（参考章节：RTC 同步）后，软件必须等待 RSF 位被置位后才能读 RTC_SSR， RTC_TR 和 RTC_DR 寄存器。

当 RTC_CR 寄存器的 BYPSHAD 控制位被置位时 （ 无需考虑影子寄存器 ）:

读日历寄存器，直接从日历计数器获取值，无需等待 RSF 位被置位。此功能在刚退出低功耗模式（停止或待机模式）时非常有用，因为影子寄存器在低功耗模式下不会自动更新。

在 BYPSHAD 为“1”时，如果两次读寄存器之间出现 RTCCLK 边沿，不同寄存器中的结果可能会互不相关。此外，如果在读操作过程中遇到 RTCCLK 边沿，则某个寄存器的值可能不正确。软件必须读取所有的寄存器两次，并比较两次读取的结果，或者通过比较两组最低有效日历寄存器的结果，以检验数据是否正确且有一定关联。

注 :BYPSHAD=1 时 ，读日历寄存器的指令的完成需另加一个额外的 APB 周期。

19.3.7 复位 RTC

任何可用的系统复位源都将导致日历影子寄存器 (RTC_SSR，RTC_TR 和 RTC_DR) 和 RTC 状态寄存器 (RTC_ISR) 复位至默认值。

然而，下列寄存器的复位于系统复位无任何关联，只与上电复位有关：RTC 当前日历寄存器、RTC 控制寄存器 (RTC_CR)、预分频器寄存器 (RTC_PRER)、RTC 校准寄存器 (RTC_ CALR)、RTC 移位寄存器 (RTC_SHIFTR)、RTC 时间戳寄存器 (RTC_TSSSR，RTC_TSTR 和 RTC_TSDR)、RTC 侵入和复用功能配置寄存器 (RTC_TAFCR)、RTC 备份寄存器 (RTC_BKPxR)、Alarm A 寄存器 (RTC_ALRMASSR/RTC_ALRMAR).

除上电复位外，任何系统复位时 RTC 将维持运行状态。发生上电复位后，RTC 停止运行，所有 RTC 寄存器复位至默认值。

19.3.8 RTC 同步

RTC 可与一个高精度远程时钟同步。在读取亚秒字段 (RTC_SSR 或 RTC_TSSSR) 后，可计算出远程时钟和 RTC 中时间的精确偏差值。RTC 可以通过 RTC_SHIFTR 寄存器用时钟移位的方式瞬间剔除这个偏差。

RTC_SSR 用于存储同步预分频器的计数器值。这允许用 1/ (PREDIV_S + 1) 秒的分辨率去衡量 RTC 所保持的实际时间。因此，这个分辨率可以通过增加同步预分频值的方式提高(PREDIV_S[14:0]. 允许的最大分辨率为 (32768 Hz 时钟时的 30.52 μs) 得自 PREDIV_S 被设为 0x7FFF 的情况。

然而，如果 PREDIV_S 增加，为了保证同步预分频器的输出为 1 Hz，PREDIV_A 必须减少。在该情况下，同步预分频器的输出频率将增加，同时动态功耗也会增加。

通过 RTC 移位控制寄存器 (RTC_SHIFTR) 可以微调 RTC。写 RTC_SHIFTR 寄存器可能移位（延迟或者提前）时钟信号，在分辨率为 1 / (PREDIV_S + 1) 秒的时候幅度可达 1 秒 . 移位操作包括将 SUBFS[14:0] 的值加到同步预分频计数器 SS[15:0]: 这将延迟时钟信号 . 如果同时 ADD1S 位是 1，结果会是加上一个整秒同时再减去不到 1 秒，所以这会提前时钟信号。

注 :在启动移位操作前，用户必须检查 SS[15] 是否为“0”以确保不会发生溢出。

写 RTC_SHIFTR 寄存器启动移位操作后，通过硬件设置 SHPF 标志，表示一个移位操作正在等待。当移位操作完成后该位被硬件清除。

注 :同步功能与参考时钟检测功能不兼容，因此当 REFCKON=1 时，固件不能写 RTC_SHIFTR。

19.3.9 RTC 参考时钟检测

RTC_REFIN 参考时钟（50Hz 或 60 Hz）与 32.768 kHz LSE 时钟比较，具有更高的精确度。RTC_REFIN 检测启用后(RTC_CR 的 REFCKON 位置“1”)，将用于补偿日历更新频率（1 Hz）的偏移。

每个 1Hz 的时钟沿会和最近的 RTC_REFIN 时钟沿（如果在给定的时间窗口中可以找到）进行比较。多数情况下两个时钟沿会恰好对齐。当发现对得不齐说明 LSE 不精确，RTC 将 1Hz 时钟移动一个最小位，从而下一个沿就能对齐。得益于这种机制使得这个万年历的精度变得和参考时钟一样。

当参考时钟停止时，日历将完全遵照 LSE 时钟进行更新。RTC 等待参考时钟，并产生一个围绕 ck_spre 边沿的检测窗口。

启动 RTC_REFIN 检测后，PREDIV_A 和 PREDIV_S 必须复位至默认值：

- PREDIV_A = 0x007F
- PREVID_S = 0x00FF

注 :RTC_REFIN 时钟检测在待机模式下禁用。

19.3.10 RTC 平滑数字校准

RTC 频率可以按大约 0.954ppm 的分辨率进行校准，范围从 -487.1 ppm 到 +488.5 ppm。 通过一系列微调（如增加 / 减少单独的 RTCCLK 脉冲）进行频率校正。这些校准会分布得相当好， 所以 RTC 会校准得很好，即便是再观察一段时间也一样。

在一个约 220 RTCCLK 脉冲周期或 32 秒（输入频率为 32768 Hz 时）内，执行平滑数字校准。平滑校准寄存器(RTC_CALR) 将明确记录在 32 秒周期中屏蔽的 RTCCLK 时钟周期的数目：

- 设置 CALM[0] 位为“1”，在 32 秒周期中一个脉冲被屏蔽。
- 设置 CALM[1] 位为“1”，另外两个周期被屏蔽。
- 设置 CALM[2] 位为“1”，另外四个周期被屏蔽。
- 以此类推，直至设置 SMC[8] 位为“1”，将有 256 个时钟被屏蔽。

最佳分辨率的情况下设置 CALM 可使 RTC 频率最多减少 487.1 ppm，通过设置 CALP 位可使频率增加 488.5 ppm。这时设置 CALP 为 1 的效果就是每 211 RTCCLK 周期插入一个额外的 RTCCLK 脉冲，也就意味着每 32 秒钟会插入 512 个时钟周期。

同时使用 CALM 和 CALP，每 32 秒可以插入 -511 到 +512 的 RTCCLK 时钟周期，换算成校准范围就是 -487.1 ppm 到 +488.5 ppm，校准步长为 0.954ppm。

下列公式用于计算有效校准频率 (FCAL)：

$F_{CAL} = F_{RTCCLK} \times [\ 1 + (CALP \times 512 - CALM) / (2^{20} + CALM - CALP \times 512)\ ]$

校准：当 PREDIV_A<3

当同步预分频器值 (RTC_PRER 寄存器的 PREDIV_A 位) 小于 3， CALP 位不能设为“1”。如果 CALP 已经被设为“1”，同时 PREDIV_A<3，则直接将 CALP 视为“0”，校准正常执行。

当 PREDIV_A<3 时执行一个校准，同步预分频器的值(PREDIV_S) 应减少，以确保所有秒均加速 8 个 RTCCLK 时钟周期，相当于每 32 秒增加了 256 个时钟周期。因此在每 32 秒周期中仅使用 CALM 位便可完成在 255 和 256 时钟脉冲间（对应 243.3 ppm - 244.1ppm 的校准范围） 有效增加。

正常情况下 RTCCLK 频率为 32768Hz，当 PREDIV_A=1（分频因子为 2），PREDIV_S 应该被设为 16379 而不是 16383（要减 4）。当 PREDIV_A=0, PREDIV_S 就应该设为 32759 而不是 32767（要减 8）。

如果 PREDIV_S 通过该方式减少，则校准输出始终的有效频率可由以下公式得出：

$F_{CAL} = F_{RTCCLK} \times [\ 1 + (256 - CALM) / (2^{20} + CALM - 256)\ ]$

在此情况下 ， 如果 RTCCLK 为 32768.00 Hz，则 CALM[7:0] 的正确值应等于 0x100 (CALM 范围的中点)。

验证 RTC 校准

通过测量 RTCCLK 的精确频率，计算正确的 CALM 和 CALP 频率以确保 RTC 精度。有一个可选的 1HZ 输出用来测量和验证 RTC 精度。

如果测量 RTC 的精确频率过程超出时间间隔， 可能会导致在测量期间内形成最多 2 个 RTCCLK 时钟周期的正误差（根据数字校准周期与测量周期的对齐方式而变）。

然而 ， 如果测量期间校准周期的长度相同，可以消除这种测量误差。在此情况下，唯一能检测到的误差是由数字校准的分辨率所产生的。

- 默认情况下，校准周期为 32 秒。

  该模式下，测量在整整 32 秒内 1HZ 输出的精确度，确保该测量精度在 0.477 ppm 内（0.5 个 RTCCLK 相对于 32 秒，受限于校准方法）。

- RTC_CALR 寄存器 CALW16 位设置为“1”，将强制使用 16 秒校准周期。

  在此情况下，在 16 秒内测量 RTC 精确度，产生的错误最大值为 0.954 ppm（0.5 个 RTCCLK 周期相对于 16

秒)。然而,因为校准解决方案的限制,长期 RTC 精确度也将减少至 0.954 ppm:当 CALW16 为“1”时,CALM[0]位将保持为“0”。

✧ RTC_CALR 寄存器 CALW8 位设置为“1”,将强制使用 8 秒校准周期。

在此情况下,在 8 秒内测量 RTC 精确度,产生的错误最大值为 1.907ppm(0.5 个 RTCCLK 周期相对于 8 秒)。长期 RTC 精确度也将变差至 1.907ppm: 当 CALW16 为“1” 时, CALM[1:0] 位将保持为“00”。

运行中重校准

当 RTC_ISR/INITF=0,校准寄存器 (RTC_CALR) 可通过以下流程实现运行中更新:

1、查询 RTC_ISR/RECALPF (重校准挂起标志);

2、如果 RTC_ISR/RECALPF 置“0”, 如果有必要通过向 RTC_CALR 写入一个新值,RECALPF 将置“1”。

3、在向 RTC_CALR 写入新值后 3 个 ck_apre 周期内,新的校准设置生效。

19.3.11 时间戳功能

RTC_CR 寄存器 TSE 位置“1”启用时间戳功能。

当 RTC_TS 引脚检测到一个时间戳事件时候,会将日历存储在时间戳寄存器中 (RTC_TSSSR, RTC_TSTR, RTC_TSDR) 。当产生时间戳事件时,RTC_ISR 寄存器的时间戳标志位 (TSF) 被置位。

通过设置 RTC_CR 寄存器的 TSIE 位,在产生时间戳事件时,同时产生一个中断。

如果检测到一个新的时间戳事件时,时间戳标志位 (TSF) 已经被置位,则时间戳溢出标志位(TSOVF) 标志为“溢出”,时间戳寄存器(RTC_TSTR 和 RTC_TSDR) 维持上一事件的内容不变。

注 1:由于同步延迟, TSF 将在时间戳事件发生后 2 个 ck_apre 周期后被置位。

注 2:TSOVF 零延迟置位。如果两个时间戳事件发生的时间间隔非常短, TSOVF 位可视为“1”,而 TSF 位仍为“0”。因此建议当 TSF 位被置位后再检测 TSOVF 位。

注意: 如果在 TSF 位被清除后立即产生一个时间戳事件, TSF 位和 TSOVF 位都将被置位。为了避免被同一时间产生的时间戳事件覆盖,应用程序不得向 RSF 位写入“0”,除非已经从该位读取到“1 ”。

(可选)一个侵入事件可导致一个时间戳被记录。参考章节:RTC 时间戳亚秒寄存器, 了解 TAMPTS 控制位的描述。

19.3.12 侵入检测

RTC_TAMPx 输入事件可设置为边沿检测或带过滤功能的电平检测。

检测可以配置为以下目的:

✧ 清除 RTC 备份寄存器

✧ 产生一个中断,能够从停止和待机模式唤醒

RTC 备份寄存器

备份寄存器(RTC_BKPxR) 处在 VDD 备份域中,当 VDD 电源被切断,他们仍由 VBAT 维持供电。当系统复位或设备在待机模式下被唤醒时,他们不会被复位。上电复位时,备份寄存器将被复位。

当产生一个侵入检测事件时,备份寄存器将被复位。(参见章节:RTC 备份寄存器(RTC_ BKPxR) 和侵入检测初始化)

侵入检测初始化

所有 RTC_TAMPx 侵入检测输入均与 RTC_ISR2 寄存器的 TAMPxF 标志相关。所有输入可通过设置 RTC_TAFCR 寄存器中相应的 TAMPxE 位为“1”来使能。

侵入检测事件可将所有备份寄存器 (RTC_BKPxR) 内容清除。

通过设置 RTC_TAFCR 寄存器的 TAMPIE 位,当检测到一个侵入检测事件时便产生一个中断。

侵入事件时间戳

设置 TAMPTS 为“1”,所有侵入事件将产生一个时间戳。这种情况下,RTC_ISR 中无论是 TSF 位还是 TSOVF 位被置 1,和正常的时间戳事件发生的效果相同。设置 TSF 或 TSOVF, 与其关联的的侵入标志寄存器 TAMPxF 将同时被设置。

侵入输入的边沿检测

如果 TAMPFLT 位为“00”,根据相关 TAMPxTRG 位,当检测到上升沿或下降沿时,RTC_ TAMPx 引脚将生成侵入检测事件。当边沿检测被选中后,RTC_TAMPx 输入的内部上拉电阻将被停用。

警告 : 为避免侵入检测事件丢失并及时地发现侵入检测事件,用于边沿检测的信号与相应 TAMPxE 位执行逻辑与操作,以便在 RTC_TAMPx 引脚启用前检测到侵入检测事件。

✧ 当 TAMPxTRG = 0:如果在启用侵入检测(通过设置 TAMPxE 位为 1)前 RTC_TAMPx 已经为高电平,一旦启用 RTC_TAMPx 输入,则会产生一个侵入事件(尽管在 TAMPxE 位置“1” 后 RTC_TAMPx 输入中并没有出现上升沿)。

✧ 当 TAMPxTRG = 1: 如果在启用侵入检测前, RTC_TAMPx 已经为低电平, 一旦启用 RTC_TAMPx,则会产生一个侵入事件(尽管在设置 TAMPxE 位后 RTC_TAMPx 输入中并没有出现下降沿)。

在一个侵入事件被检测到并清除后,RTC_TAMPx 复用功能应该被禁止。然后在再次写入备份寄存器前重新启用 RTC_TAMPx 复用功能(通过设置 TAMPxE 位为 1)。这样可以阻止软件在 RTC_TAMPx 检测出仍有侵入事件发生时对备份寄存器进行写操作。这相当于对 RTC_ TAMPx 复用功能输入进行电平检测。

注 :当电源断开时,侵入检测功能仍然有效。为避免对备份寄存器产生不必要的复位, RTC_TAMPx 复用功能对应的引脚应该在片外连接到正确的电平。

针对 RTC_TAMPx 输入的带滤波功能电平检测

将 TAMPFLT 设置为非“0”值，会启用带滤波功能的电平检测。当检测到 2 个、4 个或 8 个（根据 TAMPFLT 设置）连续样本的指定电平（TAMPxTRG 位决定）时，将产生一个侵入检测事件。

RTC_TAMPx 输入在其状态被采样前，通过 I/O 内部上拉电阻实现预充电，直至 TAMPPUDIS 设置为“1”后，停止该功能。预充电时间由 TAMPPRCH 位决定，允许 RTC_TAMPx 输入拥有更大的电容。

可通过调整 TAMPFREQ 来改变电平检测的采样频率，从而在侵入检测延迟与通过上拉电阻产生的电源消耗间进行取舍。

注 :有关上拉电阻的电气特性请参考数据手册。

19.3.13 校准时钟输出

当 RTC_CR 寄存器 COE 位置“1”，RTC_CALIB 输出上会提供一个参考时钟。

如果 RTC_CR 寄存器 COSEL 位为 0，且 PREDIV_A = 0x7F，RTC_CALIB 频率=fRTCCLK/64，对应于 32.768kHz 的 RTCCLK 时，输出频率为 512HZ。因为下降沿存在轻微抖动，所以 RTC_CALIB 占空比是不规则的。因此建议使用上升沿。

如果 COSEL=1，“PREDIV_S+1”为非“0”的 256 的倍数（例如 PREDIV_S[7:0] =0xFF），RTC_CALIB 频率等于 fRTCCLK/(256 * (PREDIV_A+1))。这样在 RTCCLK 频率为 32.768 kHz 的时候，对于的 RTC_CALIB 输出频率为 1HZ（默认 PREDIV_A = 0x7F，PREDIV_ S =0xFF）。

19.3.14 闹钟输出

RTC_CR 寄存器的 OSEL[1:0] 控制位的功能是激活报警备用功能 RTC_ALARM 输出，并选择输出功能。这些功能与 RTC_ISR 寄存器中相应标志位的内容一致。

输出极性由 RTC_CR 寄存器的 POL 控制位决定，因此当 POL 设置为“1”时，就会输出选定标志位的取反后的值。

报警复用功能输出

通过设置 RTC_TAFCR 寄存器 ALARMOUTTYPE 控制位，RTC_ALARM 引脚可被设置为开漏输出或推挽输出。

注 1：一旦 RTC_ALARM 输出启用，其优先级将高于 RTC_CALIB（COE 位无任何影响，但必须保持清除状态）。

注 2：RTC_CALIB 或 RTC_ALARM 输出启用后， RTC_OUT 脚将自动被配置为复用输出功能。

19.4 RTC 低功耗模式

RTC 低功耗模式的影响

模式	描述
睡眠模式	无影响 RTC 中断使设备退出睡眠模式。
停止模式	如果 RTC 时钟源为 LSE 或 LSI，RTC 将保持运行状态。RTC 闹钟 ，RTC 侵入事件 ，RTC 时间戳事件和 RTC 唤醒事件使设备退出停止模式。
待机模式	如果 RTC 时钟源为 LSE 或 LSI，RTC 将保持运行状态。RTC 闹钟 ，RTC 侵入事件 ，RTC 时间戳事件和 RTC 唤醒事件使设备退出待机模式。

19.5 RTC 中断

所有 RTC 中断都连接到 EXTI 控制器，参见章节：外部中断/事件控制器（EXTI）。

按下列顺序启用 RTC 中断：

1、在中断模式下设置并使能 RTC 事件对应的 NVIC 线，并选定上升沿极性。

2、设置并使能 NVIC 的 RTC IRQ 通道。

3、设置 RTC，产生 RTC 中断。

中断控制位

中断事件	事件标志	使能控制位	退出睡眠模式	退出停止模式	退出待机模式
Alarm A	ALRAF	ALRAIE	yes	yes(注 1)	yes(注 1)
RTC_TS 输入(时间戳)	TSF	TSIE	yes	yes(注 1)	yes(注 1)
RTC_TAMP1 输入检测	TAMP1F	TAMPIE	yes	yes(注 1)	yes(注 1)
RTC_TAMP2 输入检测	TAMP2F	TAMPIE	yes	yes(注 1)	yes(注 1)

注 1：仅当 RTC 时钟源为 LSE 或 LSI 时才有可能从停止和待机模式中唤醒。

19.6 相关寄存器

19.6.1 寄存器汇总表

基址：0x4000 2800				
偏移地址	寄存器名	复位值	功能描述	章节

0x00	**RTC_TR**	0x0000 0000	RTC 时间寄存器	19.6.2
0x04	**RTC_DR**	0x0000 2101	RTC 日期寄存器	19.6.3
0x08	**RTC_CR**	0x0000 0000	RTC 控制寄存器	19.6.4
0x0C	**RTC_ISR**	0x0000 0007	RTC 初始化和状态寄存器	19.6.5
0x10	**RTC_PRER**	0x007F 00FF	RTC 预分频器寄存器	19.6.6
0x14	Res.	-	-	-
0x18	Res.	-	-	-
0x1C	**RTC_ALRMAR**	0x0000 0000	RTC 闹钟寄存器	19.6.7
0x20	Res.	-	-	-
0x24	**RTC_WPR**	0x0000 0000	RTC 写保护寄存器	19.6.8
0x28	**RTC_SSR**	0x0000 0000	RTC 亚秒寄存器	19.6.9
0x2C	**RTC_SHIFTR**	0x0000 0000	RTC 移位寄存器	19.6.10
0x30	**RTC_TSTR**	0x0000 0000	RTC 时间戳事件寄存器	19.6.11
0x34	**RTC_TSDR**	0x0000 0000	RTC 时间戳日期寄存器	19.6.12
0x38	**RTC_TSSSR**	0x0000 0000	RTC 时间戳亚秒寄存器	19.6.13
0x3C	**RTC_CALR**	0x0000 0000	RTC 校准寄存器	19.6.14
0x40	**RTC_TAFCR**	0x0000 0000	RTC 侵入和复用功能配置寄存器	19.6.15
0x44	**RTC_ALRMASSR**	0x0000 0000	RTC Alarm A 亚秒寄存器	19.6.16
0x48~0x4C	Res.	-	-	-
0x50~0x60	**RTC_BKPxR, x=0~4**	0x0000 0000	RTC 备份域寄存器	19.6.17

19.6.2 RTC 时间寄存器(RTC_TR)

RTC_TR			地址偏移	0x00		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	PM	HT[1:0]		HU[3:0]			
R/W	-	rw	rw	rw	rw	rw	rw	rw
复位值	-	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	MNT[2:0]			MNU[3:0]			
R/W	-	rw	rw	rw	rw	rw	rw	rw
复位值	-	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	ST[2:0]			SU[3:0]			
R/W	-	rw	rw	rw	rw	rw	rw	rw
复位值	-	0	0	0	0	0	0	0

BIT[22]	PM	AM/PM 标记 0: AM or 24 小时制 1: PM
BIT[21:20]	HT[1:0]	时，十位值以 BCD 格式存储。
BIT[19:16]	HU[3:0]	时，个位值以 BCD 格式存储。
BIT[14:12]	MNT[2:0]	分，十位值以 BCD 格式存储。
BIT[11:8]	MNU[3:0]	分，个位值以 BCD 格式存储。
BIT[6:4]	ST[2:0]	秒，十位值以BCD 格式存储。
BIT[3:0]	SU[3:0]	秒，个位值以BCD 格式存储。

19.6.3 RTC 日期寄存器(RTC_DR)

RTC_DR			地址偏移	0x04		复位值	0x0000 2101	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-

	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	YT[3:0]				YU[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	WDU[2:0]-			MT	MU[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	1	0	0	0	0	1
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	DT[1:0]		DU[3:0]			
R/W	-	-	Rw	rw	rw	rw	rw	rw
复位值	-	-	0	0	0	0	0	1

BIT[23:20]	YT[3:0]	年，十位值以 BCD 格式存储。
BIT[19:16]	YU[3:0]	年，个位值以 BCD 格式存储。
BIT[15:13]	WDU[2:0]	星期 000：禁用 001：星期一 ... 111：星期日。
BIT[12]	MT	月，十位值以 BCD 格式存储。
BIT[11:8]	MU	月，个位值以 BCD 格式存储。
BIT[5:4]	DT[1:0]	日，十位值以 BCD 格式存储。
BIT[3:0]	DU[3:0]	日，个位值以 BCD 格式存储。

19.6.4 RTC 控制寄存器(RTC_CR)

RTC_CR			地址偏移	0x08		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	COE	OSEL[1:0]		POL	COSEL	BKP	SUB1H	ADD1H
R/W	-	rw	rw	rw	rw	rw	rw	rw
复位值	-	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	TSIE	-	-	ALRAIE	TSE	-	-	ALRAE
R/W	rw	-	-	rw	rw	-	-	rw
复位值	0	-	-	0	0	-	-	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	FMT	BYPSHAD	PEFCKON	TSEDGE	-	-	-
R/W	-	rw	rw	rw	rw-	-	-	-
复位值	-	0	0	0	0	-	-	-

BIT[23]	COE	校准输出使能。 该位使能 RTC_CALIB 输出。 0：校准输出禁用； 1：校准输出启用。
BIT[22:21]	OSEL[1:0]	输出选择。 该位用于选择 RTC_ALARM 输出关联的标志位。 00：输出禁用； 01：Alarm A 输出启用； 10：保留； 11：保留。
BIT[20]	POL	输出极性。 该位用于设置 RTC_ALARM 的输出极性。 0：根据 OSEL[1:0] 位，ALRAF 有效时，引脚为高电平；

		1：根据OSEL[1:0] 位，ALRAF 有效时，引脚为低电平
BIT[19]	COSEL	校准输出选择。 COE=1 时，该位用于选择 RTC_CALIB 的输出信号。 0：校准输出为 512 Hz； 1：校准输出为 1 Hz。 上述频率在 RTCCLK 为 32.768 kHz 和预分频器处于默认值 (PREDIV_A=127，PREDIV_S=255) 时有效。参见章节：校准时钟输出。
BIT[18]	BKP	备份。 该位可由用户写入，用于记录夏令时是否已经发生变化。
BIT[17]	SUB1H	减少1 小时(冬季时间变化)。 读该位会恒为“0”。在初始化模式之外设置该位，如果当前小时位不为 0，日历时间将减少 1 小时。如果当前小时位为 0，则该位无效。 0：无效。 1：当前时间减少 1 小时，用于校准冬季时间变化。
BIT[16]	ADD1H	增加1 小时(夏季时间变化)。 读该位会恒为“0”。在初始化模式外设置该位，日历时间会增加 1 小时。 0：无效； 1：当前时间增加1 小时，用于校准夏季时间变化。
BIT[15]	TSIE	时间戳中断使能。 0：时间戳中断禁用； 1：时间戳中断启用。
BIT[12]	ALRAIM	Alarm A 中断使能。 0：Alarm A 中断禁用； 1：Alarm A 中断启用。
BIT[11]	TSE	TSE：时间戳使能 0：时间戳禁用； 1：时间戳启用。
BIT[8]	ALRAE	Alarm A 使 能 0：Alarm A 禁用； 1：Alarm A 启用。
BIT[6]	FMT	时间格式 0：24 小时/ 天 格式； 1：AM/PM 时间格式。
BIT[5]	BYPSHAD	绕过影子寄存器 0：从影子寄存器读取日历值，每两个 RTCCLK 周期更新一次； 1：直接从日历计数器读取日历值。 *注：如果 APB1 时钟频率低于 RTCCLK 频率的 7 倍，BYPSHAD 必须置“1 ”。*
BIT[4]	REFCKON	RTC_REFIN 参考时钟检测使能(50 或60 Hz) 0：RTC_REFIN 检测禁用； 1：RTC_REFIN 检测启用。 *注：PREDIV_S 必须为 0x00FF.*
BIT[3]	TSEDGE	时间戳事件有效边沿 0：RTC_TS 输入上升沿生成一个时间戳事件； 1：RTC_TS 输入下降沿生成一个时间戳事件。 当TSEDGE 的值改变时TSE 必须为零，以避免产生不必要的TSF 值。

19.6.5 初始化和状态寄存器(RTC_ISR)

RTC_ISR			地址偏移	0x0C		复位值	0x0000 0007	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	RECALPE
R/W	-	-	-	-	-	-	-	r
复位值	-	-	-	-	-	-	-	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

控制位	-	TAMP2F	TAMP1F	TSOVF	TSF	-	-	ALRAF
R/W	-	rc_w0	rc_w0	rc_w0	rc_w0	-	-	rc_w0
复位值	-	0	0	0	0	-	-	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	INIT	INITF	RSF	-	SHPF	-	-	ALRAWF
R/W	rw	rw	rw	-	rw	-	-	r
复位值	0	0	0	-	0	-	-	1

BIT[16]	RECALPF	校准挂起标志 当软件向 RTC_CALR 做写操作时，RECALPF 状态标志位自动置“1”，表示 RTC_CALR 寄存器被封锁。当有其他新的校准设置执行时，该位回到“0”。参见“运行中重校准”。
BIT[14]	TAMP2F	RTC_TAMP2 检测标志 当在 RTC_TAMP1 输出检测到侵入事件时该标志由硬件置位。该标志由软件写入“0”后清除。
BIT[13]	TAMP1F	RTC_TAMP1 检测标志 当在 RTC_TAMP1 输出检测到侵入事件时该标志由硬件置位。该标志由软件写入“0”后清除。
BIT[12]	TSOVF	时间戳溢出标志 该标志在 TSF 已经被置 1，又产生时间戳事件时由硬件置位。由软件写入“0”后清除。建议在检查 TSF 位清除后再清除 TSOVF，否则如果在 TSF 位清除前产生一个时间戳事件，溢出可能会被忽略。
BIT[11]	TSF	时间戳标志 当产生时间戳事件时该标志由硬件置位。该标志由软件写入“0”后清除。
BIT[8]	ALRAF	Alarm A 标志 当时间 / 日期寄存器 (RTC_TR and RTC_DR) 与Alarm A 寄存器 (RTC_ALRMAR) 匹配时，该标志由硬件置位。 该标志由软件写入“0”后清除。
BIT[7]	INT	初始化模式 0：运行模式 1：在初始化模式下编程时间和日期寄存器 (RTC_TR 和 RTC_DR)，以及预分频器寄存器 (RTC_PRER)。计数器停止，直至 INIT 复位后计数器将从新值开始计数。
BIT[6]	INITF	初始化标志 该位置“1”，RTC 处在初始化状态，时间、日期和预分频器寄存器可被更新。 0：日历寄存器不可更新； 1：日历寄存器可更新。
BIT[5]	RSF	寄存器同步标志 每当日历寄存器中的内容复制到影子寄存器 (RTC_SSRx, RTC_TRx and RTC_DRx) 中时，该位由硬件置位。当移位操作被挂起 (SHPF=1) 或处于忽略影子寄存器模式(BYPSHAD=1) 下时，该位由硬件在初始化模式下清除。该位也可由软件清除。 在初始化模式下，该位可由硬件 / 软件清除。 0：日历影子寄存器尚未同步； 1：日历影子寄存器已经同步。
BIT[4]	-	保留，保持复位值
BIT[3]	SHPF	移位操作挂起 0：没有移位操作被挂起 1：有移位操作被挂起 当通过向 RTC_SHIFTR 寄存器写入产生一个移位操作时，该位立即由硬件置位。当相应的移位操作执行完毕后，该位由软件清除。直接对 SHPF 写入是无效的。
BIT[0]	ALRAWF	Alarm A 写 标 志 当RTC_CR 的ALRAE 被清“0”后，当 Alarm A 的值可以修改时，该位由硬件置位。 0：Alarm A 不可更新； 1：Alarm A 可以更新。

19.6.6 RTC 预分频器寄存器(RTC_PRER)

RTC_PRER	地址偏移	0x10	复位值	0x007F 00FF

	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	PREDIV_A[6:0]						
R/W	-	rw	rw	rw	rw	rw	rw	rw
复位值	-	1	1	1	1	1	1	1
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	PREDIV_S[14:8]						
R/W	-	rw	rw	rw	rw	rw	rw	rw
复位值	-	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	PREDIV_S[7:0]							
R/W	Rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	1	1	1	1	1	1	1

BIT[22:16]	PREDIV_A[6:0]	异步预分频器系数 ck_apre 频率 = RTCCLK 频率 /(PREDIV_A+1)
BIT[14:0]	PREDIV_S[14:0]	同步预分频器系数 ck_spre 频率 = ck_apre 频率 /(PREDIV_S+1)

19.6.7 RTC 闹钟寄存器(RTC_ALRMAR)

RTC_ALRMAR			地址偏移	0x1C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	MSK4	WDSEL	DT[1:0]		DU[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	MSK3	PM	HY[1:0]		HU[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	MSK2	MNT[2:0]			MNU[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	MSK1	ST[2:0]			SU[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31]	MSK4	Alarm A 日期屏蔽 0: 如果日期 / 星期匹配，Alarm A 置位 1: 日期/ 星期的值对Alarm A 无影
BIT[30]	WDSEL	日期选择 0: DU[3:0] 表示日期 1: DU[3:0] 表示星期数。 DT[1:0] 不起作用。
BIT[29:28]	DT[1:0]	日期，十位值。以 BCD 格式存储。
BIT[27:24]	DU[3:0]	日期，十位值。以 BCD 格式存储。
BIT[23]	MSK3	Alarm A “时”屏蔽 0: 如果“小时”匹配，Alarm A 置位 1: “小时”的值对Alarm A 无影响
BIT[22]	PM	AM/PM 标志 0: AM 或 24 小时制 1: PM
BIT[21:20]	HT[1:0]	小时，十位值。以 BCD 格式存储。
BIT[19:16]	HU[3:0]	小时，个位值。以 BCD 格式存储。位15 MSK2: Alarm A“分”屏蔽

		0：如果“分”匹配，Alarm A 置位 1：“分”的值对 Alarm A 无影响
BIT[15]	MSK2	Alarm A“分”屏蔽 0：如果“分”匹配，Alarm A 置位 1：“分”的值对 Alarm A 无影响
BIT[14:12]	MNT[2:0]	分，十位值。以 BCD 格式存储。
BIT[11:8]	MNU[3:0]	分，个位值。以 BCD 格式存储。
BIT[7]	MSK1	Alarm A“秒”屏蔽 0：如果“秒”匹配，Alarm A 置位 1：“秒”的值对 Alarm A 无影响
BIT[6:4]	ST[2:0]	ST[2:0]：秒，十位值。以BCD 格式存储。
BIT[3:0]	SU[3:0]	秒，个位值。以BCD 格式存储。

19.6.8 RTC 写保护寄存器(RTC_WPR)

RTC_WPR			地址偏移	0x24		复位值	0x0000 0000
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	KEY							
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0

BIT[7:0]	KEY	写保护键 该字节由软件写入。始终读为“0x00” 有关如何解除 RTC 寄存器写保护的信息，请参见 RTC 寄存器写保护。

19.6.9 RTC 亚秒寄存器(RTC_SSR)

RTC_SSR			地址偏移	0x28		复位值	0x0000 0000
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	SS[15:8]							
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	SS[7:0]							
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	SS	亚秒值

		SS[15:0] 是同步预分频器计数器中的值。“秒”的小数部分由下列公式给出： Second fraction = (PREDIV_S - SS) / (PREDIV_S + 1) 注：仅当执行完一个移位操作后，SS 可能大于 PREDIV_S。此时，正确的时间/日期比 RTC_TR/RTC_DR 少一秒。

19.6.10 RTC 移位控制寄存器(RTC_SHITFR)

RTC_SHIFTR			地址偏移	0x2C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	ADD1S	-	-	-	-	-	-	-
R/W	w	-	-	-	-	-	-	-
复位值	0	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	SUBFS[14:8]						
R/W	-	w	w	w	w	w	w	w
复位值	-	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	SUBFS[7:0]							
R/W	w	w	w	w	w	w	w	w
复位值	0	0	0	0	0	0	0	0

BIT[31]	ADD1S	增加一秒 0：无效 1：时钟 / 日历增加一秒。 该位只能写入且始终读为“0”。当移位操作挂起（RTC_ISR 中 SHPF=1），写操作对该位无影响。 该性质与 SUBFS 一起可用于在某次单独操作中，有效增加时钟的值， 增加值为若干分之一秒。
BIT[14:0]	SUBFS	减少若干分之一秒 该位只能写入， 如果读则恒为“0”。 当一个操作在执行时（RTC_ISR 的 SHPF=1），写该位无效。 写入 SUBFS 的值将添加到同步预分频器计数器。如果计数器是倒计时的，时钟将被延迟，延迟时间由以下公式得出： Delay (seconds) = SUBFS / (PREDIV_S + 1) 当ADD1S 功能与SUBFS 一同使用时，时钟（推进时钟）将增加若干分之一秒，具体增加值有以下公式得出： Advance (seconds) = (1 - (SUBFS / (PREDIV_S + 1))) . *注：写 SUBFS 将导致 RSF 被清除。软件持续运行直至 RSF=1 ，从而确保影子寄存器中* *相应值已与移位时间同步。*

19.6.11 RTC 时间戳事件寄存器(RTC_TSTR)

RTC_TSTR			地址偏移	0x30		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	PM	HT[1:0]		HU[3:0]			
R/W	-	r	r	r	r	r	r	r
复位值	-	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	MNT[2:0]			MNU[3:0]			
R/W	-	r	r	r	r	r	r	r

复位值	-	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	ST[2:0]			SU[3:0]			
R/W	-	r	r	r	r	r	r	r
复位值	-	0	0	0	0	0	0	0

BIT[22]	PM	AM/PM 标记 0: AM 或 24 小时制 1: PM
BIT[21:20]	HT[1:0]	小时，十位值。以 BCD 格式存储
BIT[19:16]	HU[3:0]	小时，个位值。以 BCD 格式存储 .
BIT[14:12]	MNT[2:0]	分，十位值。以 BCD 格式存储 .
BIT[11:8]	MNU[3:0]	分，个位值。以BCD 格式存储.
BIT[6:4]	ST[2:0]	秒，十位值。以BCD 格式存储.
BIT[3:0]	SU[3:0]	秒，个位值。以BCD 格式存储.

19.6.12 RTC 时间戳日期寄存器(RTC_TSDR)

RTC_TSDR			地址偏移	0x34		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	WDU[1:0]			MT	MU[3:0]			
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	DT[1:0]		DU[3:0]			
R/W	-	-	r	r	r	r	r	r
复位值	-	-	0	0	0	0	0	0

BIT[15:13]	WDU[1:0]	星期数
BIT[12]	MT	月，十位值。以BCD 格式存储
BIT[11:8]	MU[3:0]	月，个位值。以BCD 格式存储
BIT[5:4]	DT[1:0]	日期，十位值。以BCD 格式存储
BIT[3:0]	DU[3:0]	日期，个位值。以BCD 格式存储

19.6.13 RTC 时间戳亚秒寄存器(RTC_TSSSR)

RTC_TSSSR			地址偏移	0x38		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	SS[15:8]							
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

控制位	SS[7:0]							
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	SS	亚秒值 当发生时间戳事件时，SS[15:0] 是同步预分频器计数器中的值。

19.6.14 RTC 校准寄存器(RTC_CALR)

RTC_CALR			地址偏移	0x3C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CALP	CALW8	CALW16	-	-	-	-	CALM[8]
R/W	rw	rw	rw	-	-	-	-	
复位值	0	0	0	-	-	-	-	
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CALM[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15]	CALP	RTC 频率增加488.5 ppm 0: 无 RTCCLK 脉冲增加。 1: 每 211 脉冲增加一个 RTCCLK 脉冲（频率增加 488.5 ppm） 该功能与 CALM 一起使用，采用好的分辨率的同时降低日历的频率。如果输入频率为 32768 Hz，则在一个 32 秒窗口中 RTCCLK 脉冲增加数可由下列公式得出：(512 * CALP) - CALM。 参见章节：RTC 平滑数字校准。
BIT[14]	CALW8	采用8 秒校准周期 CALW8 置“1”，选择 8 秒校准周期。 *注:当 CALW8=’ 1’时， CALM[1:0] 锁定为“ 00 ” . 参见章节： RTC 平滑数字校准。*
BIT[13]	CALW16	采用16 秒校准周期 当 CALW16 置“1”，激活 16 秒校准周期。如果 CALW8=1，该位不能为“1”。 *注:当 CALW16=’ 1’时， CALM[0] 锁定为‘ 0 ’。 参见章节： RTC 平滑数字校准。*
BIT[8:0]	CALM[8:0]	校准减 减少日历频率：在 220 RTCCLK 脉冲范围内(如果输出频率为 32768 Hz，则为 32 秒)通过屏蔽 CALM 将减少日历(按 0.9537 ppm 分辨率)的频率。 要增加日历频率：此功能应与 CALP 一同使用。参见章节：RTC 滑数字校准

19.6.15 RTC 侵入和复用功能配置寄存器(RTC_TAFCR)

RTC_TAFCR			地址偏移	0x40		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	PC15MODE	PC15VALUE	PC14MODE	PC14VALUE	PC13MODE	PC13VALUE	-	-
R/W	rw	rw	rw	rw	rw	rw	-	-
复位值	0	0	0	0	0	0	-	-

	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	TAMPPUDIS	TAMPPRCH[1:0]		TAMPFL[1:0]		TAMPFREQ[2:0]		
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	TAMPTS	-	-	TAMP2TRG	TAMP2E	TAMPIE	TAMP1TRG	TAMP1E
R/W	rw	-	-	rw	rw	rw	rw	rw
复位值	0	-	-	0	0	0	0	0

BIT[23]	PC15MODE	PC15 模式 0: PC15 受 GPIO 配置寄存器控制，因此 PC15 在待机（standby）模式下浮空。 1: 若 LSE 禁用时，PC15 强制为推挽输出模式。
BIT[22]	PC15VALUE	PC15 值 若 LSE 禁用且 PC15MODE = 1，PC15VALUE 配置为 PC15 的输出值。
BIT[21]	PC14MODE	PC14 模式 0: PC14 受 GPIO 配置寄存器控制，因此 PC14 在待机（standby）模式下浮空。 1: 若 LSE 禁用时，PC14 强制为推挽输出模式。
BIT[20]	PC14VALUE	PC14 值 如果LSE 禁用且PC14MODE = 1，PC14VALUE 配置为PC14 的输出值。
BIT[19]	PC13MODE	PC13 模式 0: PC13 受 GPIO 配置寄存器控制，因此 PC13 在待机（standby）模式下浮空。 1: 若 LSE 禁用时，PC13 强制为推挽输出模式。
BIT[18]	PC13VALUE	RTC_ALARM 输出形式 / PC13 值 如果 PC13 用于输出 RTC_ALARM，PC13VALUE 为输出配置： 0: RTC_ALARM 为开漏输出； 1: RTC_ALARM 为推挽输出 如果所有 RTC 复用功能禁用 且 PC13MODE = 1，PC13VALUE 配置 PC13 的输出值。
BIT[15]	TAMPPUDIS	RTC_TAMPx 上拉禁用（RTC_TAMPx pull-up disable） 由该位决定是否所有 RTC_TAMPx 引脚在采样前进行预充电。 0: RTC_TAMPx 引脚在采样前进行预充电（使能内部上拉） 1: RTC_TAMPx 引脚的预充电功能禁用
BIT[14:13]	TAMPPRCH[1:0]	RTC_TAMPx 预充电时间（RTC_TAMPx precharge duration） 该位决定采样前上拉电阻启用时间。TAMPPRCH 适用于每个 RTC_TAMPx 输入。 00: 1 个 RTCCLK 周期 01: 2 个 RTCCLK 周期 10: 4 个 RTCCLK 周期 11: 8 个 RTCCLK 周期
BIT[12:11]	TAMPFLT[1:0]	RTC_TAMPx 滤波器计数（RTC_TAMPx filter count） 该位决定在特定电平（TAMP*TRG）下（能够激活侵入事件）的连续采样数。 TAMPFLT 适用于每个 RTC_TAMPx 输入。 00: 输入有效电平的跳变沿将激活侵入事件(RTC_TAMPx 输入无内部上拉)。 01: 2 次连续采样到有效电平，侵入事件被激活。 10: 4 次连续采样到有效电平，侵入事件被激活。 11: 8 次连续采样到有效电平，侵入事件被激活。
BIT[10:8]	TAMPFREQ[2:0]	侵入采样频率（Tamper sampling frequency） 确定 每个 RTC_TAMPx 输入的采样频率。 0x0: RTCCLK / 32768 (当 RTCCLK = 32768 Hz 时为 1 Hz) 0x1: RTCCLK / 16384 (当 RTCCLK = 32768 Hz 时为 2 Hz) 0x2: RTCCLK / 8192 (当 RTCCLK = 32768 Hz 时为 4 Hz) 0x3: RTCCLK / 4096 (当 RTCCLK = 32768 Hz 时为 8 Hz) 0x4: RTCCLK / 2048 (当 RTCCLK = 32768 Hz 时为 16 Hz) 0x5: RTCCLK / 1024 (当 RTCCLK = 32768 Hz 时为 32 Hz) 0x6: RTCCLK / 512 (当 RTCCLK = 32768 Hz 时为 64 Hz) 0x7: RTCCLK / 256 (当 RTCCLK = 32768 Hz 时为 128 Hz)
BIT[7]	TAMPTS	侵入检测事件的有效时间戳（Activate timestamp on tamper detection event） 0: 侵入检测事件不产生时间戳。 1: 侵入检测事件产生时间戳并保存。 即使 RTC_CR 中的 TSE=0，TAMPTS 依然有效。

BIT[4]	TAMP2TRG	RTC_TAMP2 输入的有效电平（Active level for RTC_TAMP2 input） **如果 TAMPFLT != 00** 0: RTC_TAMP2 输入保持低电平将触发一个侵入检测事件。 1: RTC_TAMP2 输入保持高电平将触发一个侵入检测事件。 **如果 TAMPFLT = 00** 0: RTC_TAMP2 输入上升沿将触发一个侵入检测事件。 1: RTC_TAMP2 输入下降沿将触发一个侵入检测事件。
BIT[3]	TAMP2E	RTC_TAMP2 检测使能 0: RTC_TAMP2 检测禁用 1: RTC_TAMP2 检测启用
BIT[2]	TAMPIE	Tamper 中断使能 0: Tamper 中断禁用 1: Tamper 中断启用
BIT[1]	TAMP1TRG	RTC_TAMP1 输入的有效电平 **如果 TAMPFLT != 00** 0: RTC_TAMP1 输入保持低电平将触发一个侵入检测事件。 1: RTC_TAMP1 输入保持高电平将触发一个侵入检测事件。 **如果 TAMPFLT = 00** 0: RTC_TAMP1 输入上升沿将触发一个侵入检测事件。 1: RTC_TAMP1 输入下降沿将触发一个侵入检测事件。
BIT[0]	TAMP1E	RTC_TAMP1 检测使能 0: RTC_TAMP1 检测禁用 1: RTC_TAMP1 检测启用

19.6.16 RTC Alarm A 亚秒寄存器(RTC_ALRMASSR)

RTC_ALRMASSR			地址偏移	0x44		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	MASKSS[3:0]-			
R/W	-	-	-	-	rw	rw	rw	rw
复位值	-	-	-	-	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	SS[14:8]						
R/W	-	rw	rw	rw	rw	rw	rw	rw
复位值	-	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	SS[7:0]							
rw	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[27:24]	MASKSS[3:0]	从该位起品比最明显的位。 0: Alarm A 不匹配亚秒值。报警在秒单元增加（ 假设其余字段是相互匹配的） 时置 1。 1: SS[14:1] 不参与 Alarm A 匹配。只有 SS[0] 参与匹配。 2: SS[14:2] 不参与 Alarm A 匹配。只有 SS[1:0] 参与匹配。 3: SS[14:2] 不参与 Alarm A 匹配。只有 SS[2:0] 参与匹配。 ... 12: SS[14:12] 不参与 Alarm A 匹配。只有 SS[11:0] 参与匹配。 13: SS[14:13] 不参与 Alarm A 匹配。只有 SS[12:0] 参与匹配。 14: SS[14] 不参与 Alarm A 匹配。只有 SS[13:0] 参与匹配。 15: 15 个 SS 位均需参与匹配，且需匹配成功以激活报警。 同步计数器溢出位（ 位15） 始终不参与匹配。只有在移位操作后，该位才不为“0”。
BIT[14:0]	SS[14:0]	亚秒值 比较亚秒值与同步预分频器计数器中的内容，从而判断报警是否已经被激活。 只有位 0 到 MASKSS-1 参与比较。

19.6.17 RTC 备份寄存器(RTC_BKPxR)

RTC_BKPxR			地址偏移	0x50 to 0x60		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	BKP[31:24]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	BKP[23:16]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	BKP[15:8]							
R/W	rw	rw	rw	rwr	rw	rw	rwr	w
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	BKP[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rwr	w
复位值	0	0	0	0	0	0	0	0

BIT[31:0]	BKP[31:0]	应用程序可对这些寄存器进行读或写操作。 当 VDD 电源被切断他们仍由 VBAT 维持供电，因此这些位不会被系统复位所复位。当设备在低功耗模式下运行时，这些位的内容仍然有效。 侵入检测事件发生时（即 TAMPxF=1）或闪存读出保护被禁用时，该寄存器复位

20 SPI接口

注：MS32F031A6 仅支持 SPI 协议，不支持 I2S 协议。

20.1 概述

SPI 接口支持基于 SPI 协议的通信。芯片复位后默认为 SPI 模式。

串行外设接口（SPI）协议，支持半双工，全双工和单工同步等方式与外部设备的串行通信。该接口可配置为主机模式，在这种情况下，它向外部的从属设备提供通信时钟（SCK）。 此接口也能够以多主机配置的方式操作。

SPI 具体功能配备

SPI 特性	SPI1
硬件 CRC 计算	√ *注*
RX/TX FIFO	√
NSS 脉冲模式	√

注： ✓ = 支持

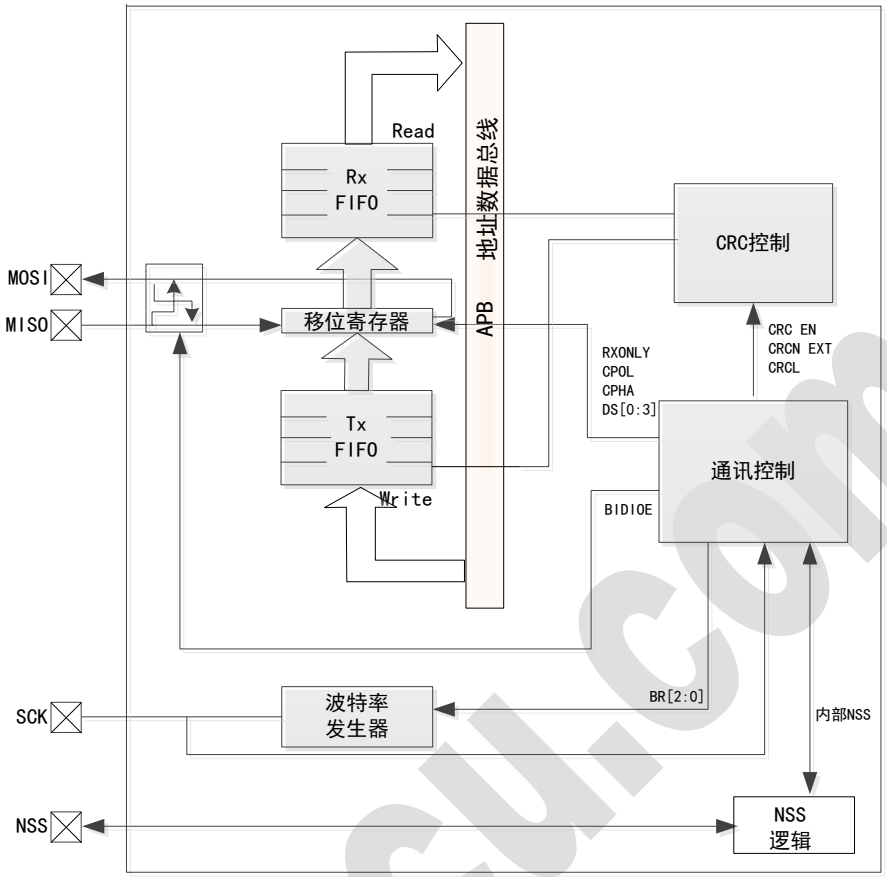
20.2 SPI 特性

- ✧ 主机或从机模式
- ✧ 三线全双工同步传输
- ✧ 两线半双工同步传输（双向数据线）
- ✧ 两线单工同步传输（单向数据线）
- ✧ 4 位~16 位数据位宽选择
- ✧ 多主机模式
- ✧ 主机模式波特率分频器 8 档可选，波特率高达 $f_{PCLK}/2$
- ✧ 从机模式频率高达 $f_{PCLK}/2$
- ✧ 主机和从机模式下都可以由硬件或软件管理 NSS：主/从模式操作的动态变化
- ✧ 可编程时钟极性和相位
- ✧ 可编程数据位序：MSB-first 或 LSB-first 移位
- ✧ 专用的发送和接收状态标志，全部支持中断触发
- ✧ SPI 总线忙状态标志
- ✧ SPI Motorola 方式支持
- ✧ 硬件 CRC 功能实现可靠的通信：
- CRC 值可以作为 TX 模式的最后一个字节传送
- 自动 CRC 错误检查最后接收到的字节
- ✧ 主模式故障、溢出标志支持中断触发
- ✧ CRC 错误标志
- ✧ 两个 32 位嵌入式 RX 和 TX FIFO，支持 DMA 功能

20.3 功能描述

SPI 支持 MCU 与外部设备的同步何异步通讯，应用中可以通过轮询状态标志或 SPI 中断来管理通讯过程。

SPI 模块框图



通常 SPI 通过 4 个引脚与外部设备相连。

- ✧ MISO： 主设备输入/ 从设备输出引脚。 一般情况下，此引脚用于在从模式下的数据发送和主模式下的数据接收。
- ✧ MOSI： 主设备输出/ 从设备输入引脚。 一般情况下，此引脚用于在从模式下的数据接收和主模式下的数据发送。
- ✧ SCK： SPI 串行时钟输出引脚，作为主设备输入，从设备的输出。
- ✧ NSS： 从机片选脚。 根据 SPI 和 NSS 设置，此引脚可用于：
- 选择一个从机来通信
- 同步数据帧或者
- 检测多个主机之间的冲突

参见章节《从机片选（NSS）引脚管理》。

SPI 总线允许一个主设备和一个或多个从设备之间的通信。 总线由至少有两条线 - 时钟信号和同步数据传输。 其他信号基于 SPI 节点和主机间的数据交换目的来添加。

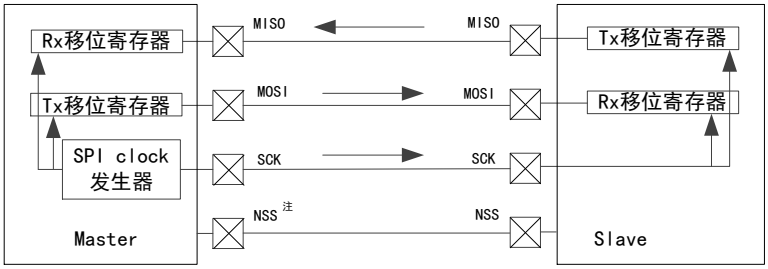
20.3.1 一个主设备和一个从机之间的通信

SPI 允许 MCU 根据目标的设备和应用程序的要求，使用不同的配置进行通信。 这些配置使用 2 或 3 线（软件 NSS 管理）或 3 或 4 线（硬件 NSS 管理）。 通信总是由主机发起的。

全双工通信

默认情况下，SPI 被配置为全双工通信。 在此配置中，主机和从机的移位寄存器通过两个单向线，MOSI 和 MISO 引脚进行连接。 在 SPI 通信时，按照主机提供的 SCK 时钟沿进行同步数据传输。 主机的数据经由 MOSI 发送到从机，从机的数据经由 MISO 线发送到主机。 当数据帧传输完成（所有位转移）时，主机和从机间的信息交换就完成了。

全双工单主 / 单从应用

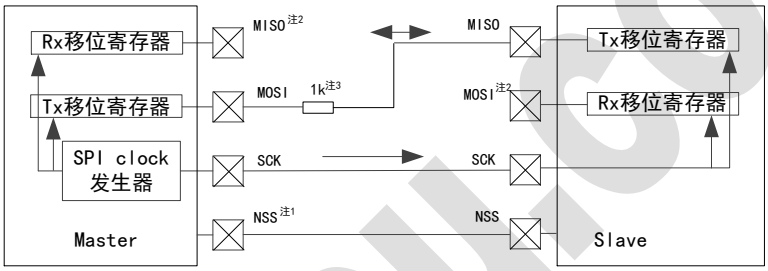


注：在这种情况下，NSS 引脚被配置为输入。

半双工通信

通过设置在 SPIx_CR1 寄存器 BIDIMODE 位，可以令的 SPI 工作在 half-duplex 模式下。在此配置中性由一个单一的跨接线连接处主机寄存器和从机。在通信期间，数据按照 SPIx_CR1 寄存器的 BDIOE 位的设定，由主机移位寄存器同步于 SCK 时钟沿传输到从机。 在此配置中， 主机的 MISO 引脚和从机的 MOSI 引脚都是自由的，可作为 GPIO 供其他应用程序使用。

半双工单主 / 单从应用



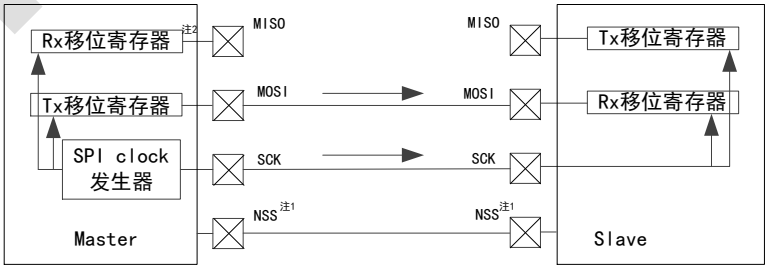
注 1：在这种情况下，NSS 引脚被配置为输入。
注 2：在此配置中，主机的 MISO 引脚和从机的 MOSI 引脚可以作为 GPIO 用。

简单通信

可以通过 SPIx_CR2 寄存器 RXONLY 位选择 SPI 工作在单工模式下，实现单发送或单接收通讯。 在此配置中，由一个单一的跨接线连接主机寄存器和从机。 其余 MISO 和 MOSI 引脚不用于通信，并可以作为标准的 GPIO 使用。

- 单发送模式（RXONLY = 0）： 配置设置和全双工相同。 应用程序必须忽略在未使用的输入引脚上捕获的信息。这个脚可以作为一个标准的 GPIO 引脚。
- 单接收模式（RXONLY = 1）： 应用程序可以设置 RXONLY 位以禁用 SPI 输出功能。在配置为从机时，MISO 输出被禁用，并且可以当作一个 GPIO 引脚来使用。 从机将从 MOSI 引脚连续接收数据，在从机选择信号处于激活状态时，它的 BSY 标志总是为 1（见 25.3.4： 从机选择（NSS）的引脚管理）。 根据数据缓冲区的配置，会出现数据接收事件。在配置为主机时，MOSI 输出被禁用，并且可以当作一个 GPIO 引脚来使用。 SPI 使能时， 时钟信号会不断产生。 要停止时钟输出，只有清除 RXONLY 位或 SPE 位，并等待直到从 MISO 引脚的输入样式的完成并填充数据缓冲区结构，这取决于其具体配置。

简单的单主 / 单从应用（主机仅发送 / 从机仅接收）

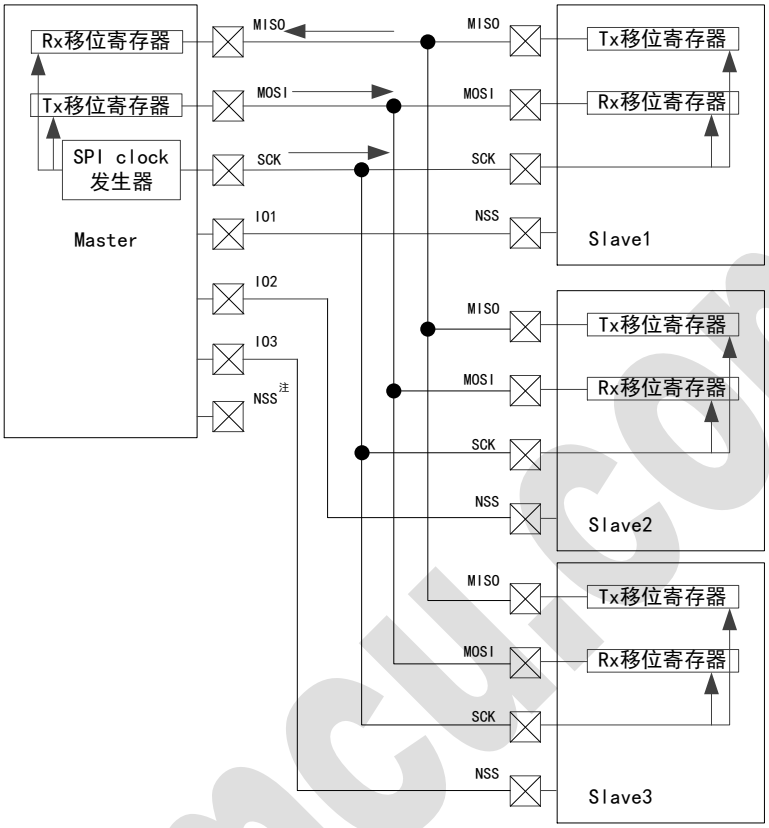


注 1：在这种情况下，NSS 引脚被配置为输入。
注 2：在移位寄存器中输入的信息被捕获，但在标准单发送模式下必须丢弃。（例如，OVF 格式标志）
注 3：在此配置中，只要是 MISO 引脚都可用作 GPIO。
注 4：简单通信模式可以被一个固定传输方向的半双工通信模式替代。

20.3.2 标准多从机通讯

在有两个或两个以上独立的从机的配置时，主机用 GPIO 引脚来管理每个从机的片选线。 主机必须通过 GPIO 拉低其中一个从机的 NSS 引脚。 一旦做到这一步，就会建立一个标准的主机和专用的通信渠道。

主机和 3 个独立的从机



注 1：由于从机的 MISO 引脚连接在一起，所有的从机都必须将他们的 MISO 引脚设置为备用功能漏极开路状态（见章节： I／O 的备用功能输入／输出）。

20.3.3 从机选择（NSS）的引脚管理

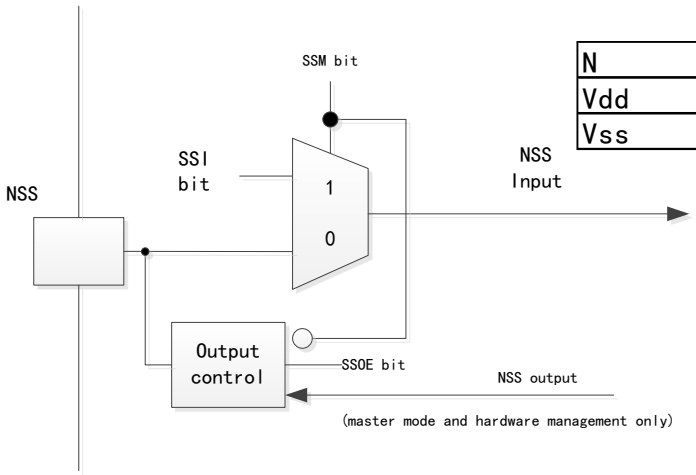
在从机模式下时，NSS 作为一个标准的“片选”输入工作，并允许主从通信。 在主机模式下， NSS 可以用来作为输入或输出。 作为输入，它可以防止多主总线冲突，作为输出，它可以驱动一个单一的从机片选信号。

可以设置 SPIx_CR1 寄存器中的 SSM 位，来选择使用硬件或软件的从机选择管理：

- ✧ 软件 NSS 管理（SSM = 1）：在此配置中，从机选择信息由内部寄存器 SPIx_CR1 的 SSI 位值来驱动。 外部 NSS 引脚可以由其他应用程序自由支配。
- ✧ 硬件 NSS 管理（SSM = 0）： 在这种情况下，有两种可能的配置。 这种设置由（寄存器 SPIx_CR1 SSOE 位）来决定 NSS 的输出。

- NSS 输出使能（SSM= 0，SSOE = 1）： 此配置仅用于当 MCU 作为主机时。 NSS 引脚由硬件管理。 在 SPI 作为主模式（SPE 的 = 1）的同时 NSS 信号被拉低，并保持低直到 SPI 被禁止（SPE = 0）。 如果 NSS 脉冲模式被打开（NSSP = 1），在连续通信间隔期间可以产生一个 NSS 脉冲。 在这种 NSS 设置中，SPI 不能支持多重主机工作状态。
- NSS 输出禁止（SSM = 0，SSOE = 0）： 如果在总线上 MCU 作为主机，那么此配置就允许多主机通讯。如果在这种模式下 NSS 引脚被拉低，SPI 进入主模式故障状态， 并自动重新配置为从机模式。在从机模式下，NSS 引脚作为标准的片选输入来工作， 当 NSS 线被拉低时，从机被选中。

硬件 / 软件从机选择管理



N	Master	Slave
Vdd	OK	Non active
Vss	Conflict	Active

20.3.4 通讯格式

在 SPI 通信中，接收和发送操作同时进行。 串行时钟（SCK）同步发送并同时采样数据线上的信息。 通讯格式取决于时钟相位，时钟极性和数据帧格式。 为了能够正常通讯，主机和从机必须遵循相同的通信格式。

时钟相位和极性控制

可以通过 SPIx_CR1 的 CPOL 和 CPHA 位用软件选择四种可能的时序关系

CPOL(时钟极性)位控制着在没有数据在发送时的空闲状态的时钟输出电平。该位既针对主机模式也针对从机模式。如果 CPOL 被清零，SCK 引脚在闲置状态输出低电平。 如果 CPOL 置 1，SCK 引脚在闲置状态输出高电平。

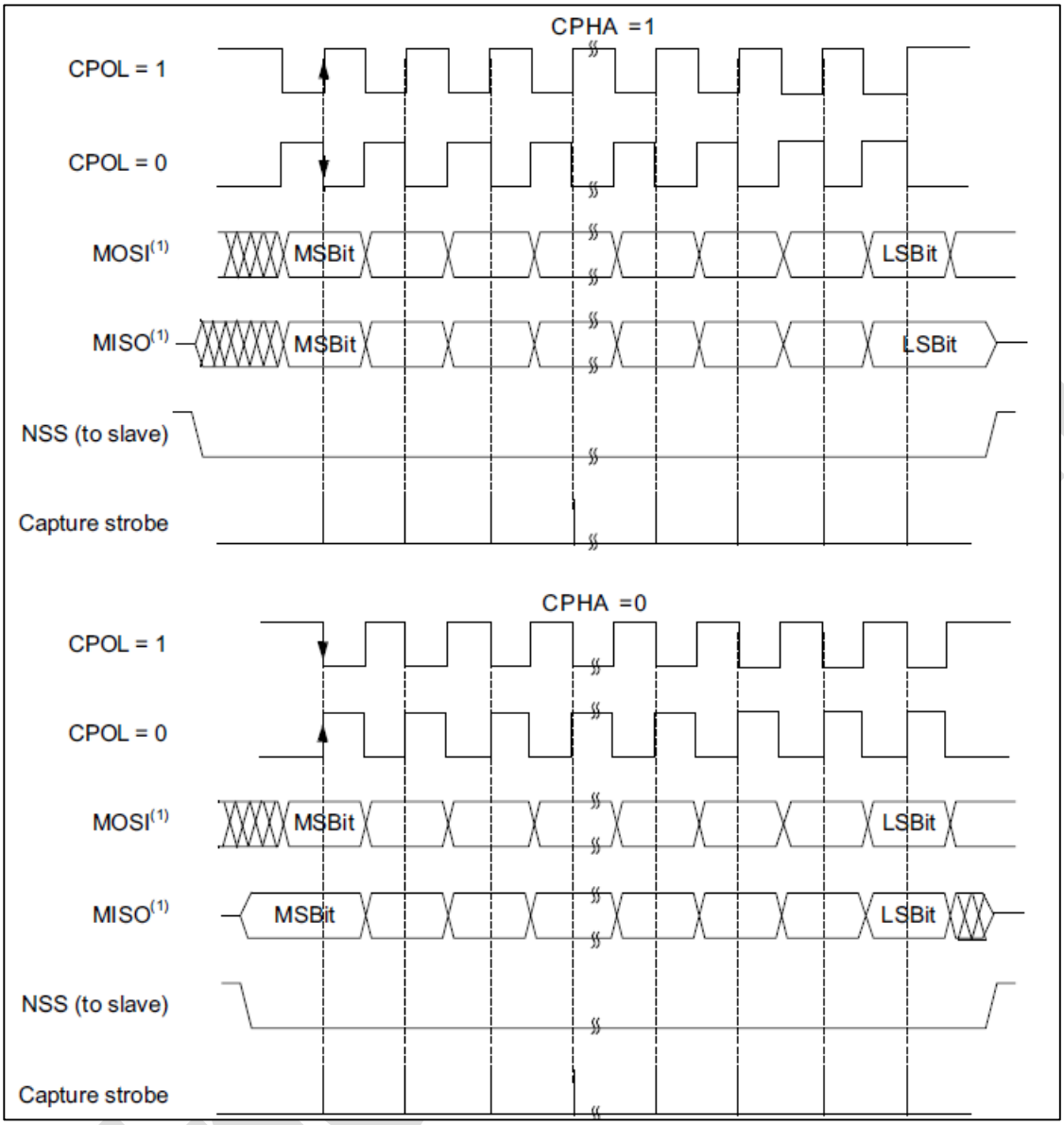
如果 CPHA 位被置 1，SCK 引脚上的第二沿对准的是第一位的捕获时机（如果 CPOL 位是 0 则为下降沿，如果 CPOL 位为 1 则为上升沿）。 每次发生这个时钟切换时，数据被锁存。 如果 CPHA 位为 0，SCK 引脚上的第个沿对准第一位的捕获时机（如果 CPOL 位为 1，则为下降沿，如果 CPOL 位为 0，则为上升沿）。 每次发生这个时钟切换时，数据被锁存。

CPOL（时钟极性）和 CPHA（时钟相位）的选择共同决定数据采集时的时钟边沿。图显示 CPHA 和 CPOL 的四种组合的 SPI 全双工传输。

注 1：改变的 CPOL / CPHA 位之前， SPI 必须通过清零 SPE 位先关闭。

注 2：在 SCK 的空闲状态，必须符合在 SPIX_CR1 寄存器中对极性的选择（如果 CPOL = 1 上拉 SCK ，如果 CPOL = 0 ， 下拉 SCK ）。

数据时钟时序图

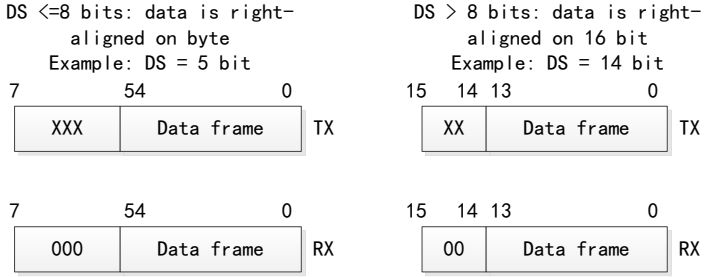


注：这些时序显示的是 SPIx_CR1 寄存器的 LSBFIRST 位为 0 且 8 位字长（DS [3:0] = 0111） 时的情形。

数据帧格式

SPI 移位寄存器可以设置 MSB 在前或 LSB 在前，取决于的 LSBFIRST 位的值。 数据字长使用 DS 位选择。它可以设定从 4 位到 16 位的长度，同时适用于发送和接收。无论选定多大字长， 对 FIFO 的读访问必须与 FRXTH 水平对齐。当访问 SPIx_DR 寄存器时，无论是一个字节（如果数据能够在一个字节中）还是一个字，数据帧总是右对齐（见下图）。通讯时，只有数据字长范围内的位会随时钟输出。

当数据长度不等于 8 位或 16 位时的数据对齐



注： 最小的数据长度为 4 位。 如果选择数据长度小于 4 位，会强制按照 8 位的数据字长。

20.3.5 SPI 的初始化

主机和从机的初始化过程几乎是相同的。 都是设置位配置寄存器 SPIx_CR1 和 SPIx_CR2：

1. 使用 BR [2:0] 位选择串行时钟的波特率（见注 1）
2. 设置 CPOL 和 CPHA 位的组合，定义数据发送和串行时钟之间的四个关系（见图注 2）
3. 配置 RXONLY，BIDIOE 和 BIDIMODE 选择传输模式（见注 3）。
4. 设置 DS 位选择用于传送的数据长度。
5. 配置 LSBFIRST 位，定义帧格式（见注 2）。
6. 根据应用的需要，设置 SSM 的，SSI 和 SSOE。 在主机模式下，内部 NSS 信号必须在完整序列期间保持为高（见章节： 从机选择引脚管理（NSS））。 在从机模式下， 内部 NSS 信号必须在完整序列期间保持为低（见注 2）。
7. 如果需要按 TI 协议，设置 FRF 位（见章节： TI 模式）。
8. 如果两个数据单位之间需要 NSS 脉冲，那么设置 NSSP 位以打开 NSS 脉冲模式。 此配置下 CHPA 位必须设置为 1（见注 2）。
9. 设置 FRXTH 位。 RXFIFO 的阈值必须和对 SPIx_DR 寄存器的读访问的字长对齐。
10. 如果使用 DMA，初始化 LDMA_TX 和 LDMA_RX 位。
11. 如果需要的 CRC，将 CRC 多项式设置为“输入”并设置 CRCEN 位。
12. 在 NSS 内部信号为高的时候设置 MSTR 位（见注 1 和第 25.3.4： 从机选择（NSS）引脚的管理）
13. 设置 SPE 位，启用 SPI（见注 3）。

注 1： 这个步骤在从模式下不需要。

注 2： 这个步骤在 TI 模式中不要求

注 3： 在任何主机单接收模式（ RXONLY = 1 或 BIDIMODE 的 = 1 & BIDIOE = 0 ），时钟在 SPI 使能后立即开始运行。

20.3.6 数据发送和接收流程

RXFIFO 和 TXFIFO

所有 SPI 数据换都通过 32 位的嵌入式 FIFO。 这使 SPI 可以连续工作，防止短数据帧时的数据断流。 每个方向都有它自己的 FIFO 称为 TXFIFO 和 RXFIFO。 这些 FIFO 被用于除了单接收 +CRC 模式外的所有的 SPI 模式（主从）（见章节：CRC 计算）。

对 FIFO 的处理会根据数据交换模式（双工，单工），数据帧格式（字长），对 FIFO 数据寄存器访问的大小（8 位或 16 位），以及是否按照数据包访问 FIFO（见章节： TI 的模式）。

对 SPIx_DR 寄存器的读访问，会返回存储在 RXFIFO 中但还未读的最老的数据。 对 SPIx_ DR 的写访问会将新的数据放在发送队列的尾部。 读访问必须总是和 SPIx_CR2 寄存器中的 FRXTH 位设置的 RXFIFO 门限对齐。 FTVL [1:0] 和 FRLVL [1:0] 位，表明两个 FIFO 目前的缓冲存储程度。

对 SPIx_DR 寄存器的读访问必须由 RXNE 事件引发。 这个时间在数据被存入 RXFIFO 并达到门槛（由 FRXTH 位定义）的时候被触发。 在 RXNE 被清除时，认为是空的 RXFIFO 已经清空。 类似的，写访问要由 TXE 事件来引发。 当 TXFIFO 的存储状况少于或等于满额的一半的时候，将触发这个事件。 否则 TXE 被清零，并且 TXFIFO 被认为是有数据。用这种方式，RXFIFO 可以存储多达 4 个数据帧，而 TXFIFO 在字长不大于 8 的时候也只可以存储多达三个数据帧。这种差异可以防止在已经有 3 个 8 位数据存在 TXFIFO 中的时候，软件试图在 16 位模式下向 TXFIFO 写入更多的数据而造成数据破坏。 TXE 和 RXNE 事件都可以通过中断查询或处理。

管理数据交换的另一种方法是使用 DMA（见章节： DMA 功能）。

如果接收到下一个数据时 RXFIFO 的是满的，将导致发生溢出事件（见章节《关于 OVR 标志的描述： 状态标志》）。溢出事件可以查询处理或中断处理。

正在执行数据传输的时候，硬件会将 BSY 位置 1 来指示这个状态。 当时钟信号连续运行时， 如果是主机模式，BSY 标志在帧与帧之间一直保持为 1，而在从机模式时，BSY 信号在帧与帧之间会有一小段时间（1 个 SPI 时钟周期）为 0。

序列处理

多个数据字节可以顺序发送来组成一个消息。 启用发送后，在主机 TXFIFO 中的数据会开始发送并连续发完。 主机会连续输出时钟信号，直至 TXFIFO 变为空，然后停下来等待新增的数据。

在单接收模式， 半双工（BIDIMODE = 1，BIDIOE = 0）， 或简单发送（BIDIMODE = 0， RXONLY = 1）模式下，只要 SPI 和单接收模式被使能，主机会立即开始接收序列。 主机连续提供时钟信号，直到主机关闭 SPI 或者关闭单接收模式之前都不会停下来。 这时主机会连续接收数据。

数据帧开始后，从机无法控制或延迟数据序列。 出于这个原因，从机必须在开始传输之前准备好数据，并总是一直保持有数据待传（存在 TXFIFO 中）。 主机必须在每个序列之间给从机保留足够的时间以准备数据。 如果可能的话，序列中的字节的数量应得到限制，以便从机完成自动的数据处理（通过使用 DMA 或 FIFO）。 主机必须提供额外的时间给从机处理数据内容。

在多从机并行的系统中，每个序列应该由 NSS 脉冲来分隔，以将每个序列对应到不同的从机。 在单从机系统中通过 NSS 控制从机就显得没那么有必要了，但这里有个脉冲还是更好， 这可以令从机和每个数据序列完成同步。 NSS 既可以由软件管理也可以由硬件管理（见第 25.3.4： 从机选择（NSS）的引脚管理）。

BSY 位被置 1 时，它标志着一个持续的传输。 这一点，结合 FTLVL [1:0] 位，可用于检查传输是否完成。 在系统进入 HALT 模式前这是很有必要的，过早的进入 Halt 模式可能导致数据破坏。 判断 BSY 位的另外一个目的是可以用关键来管理 NSS 信号。 当 RXNE 标志被置 1，它意味着对当前的传输结束了。 即最后一位刚刚采样完毕，以及完整

的数据帧存储在 RXFIFO 中。

当停止提供传输数据后，主机会完成当前数据传输。 在这种情况下，最后一个数据传输结束后时钟输出就会停止。在数据包模式中，奇数个数据传输完毕后要特别注意防止出现空的字节（见第 25.4.2： TI 模式）。 当主机处于单接收模式下，只有禁用 SPI 或单接收模式，才能停止时钟。为了收到正确数量的字节并且阻止任何无效的空数据，就得在最后一个字节正在传输的时候，把握正确的关闭接收的时机。 关闭动作必须发生在首位的采样时间和下一个字节（不要的空数据）的首位之间。

禁用 SPI 的步骤

在一帧数据正在传输的时候，或者 TXFIFO 当中有数据的时候，主机不可以通过操作 SPE 位来关闭 SPI。 如果发生这种情况，时钟信号将持续发送，直到外设被重新启用使得传输可全面完成。 如果收到了数据，但是仍然遗留在 RXFIFO 中未及读取时关闭了 SPI 的话，那么在下一次打开 SPI 开始新的传输序列之前一定要先处理历史数据。 为了防止这种情况，请确保 RXFIFO 的是空的时候再去禁用 SPI。 这个过程可以通过正确的流程来实现，或者通过专门的外设复位控制寄存器来执行软件复位命令，从而全面初始化 SPI 的寄存器（见 RCC_ APBiRSTR 中的 SPIiRST 位）。

正确的关闭流程是（除了使用单接收模式时）：

1. 等到 FTLVL [1:0] = 00（没有更多的数据传输）
2. 等待，直到的 BSY = 0（最后一个数据帧处理完毕）
3. 读取数据直到 FRLVL [1:0] = 00（读取所有接收到的数据）
4. 禁止 SPI（SPE 的 = 0）。

某些单接收模式的正确关闭步骤是：

1. 在最后一个数据的传输期间的特定时间窗口中关闭单接受模式（RXONLY = 0 或 BIDIOE= 1）
2. 等待直到 BSY = 0（最后一个数据帧处理完毕）
3. 读取数据，直到 FRLVL [1:0] = 00（读取完所有接收到的数据）
4. 禁止 SPI（SPE= 0）。

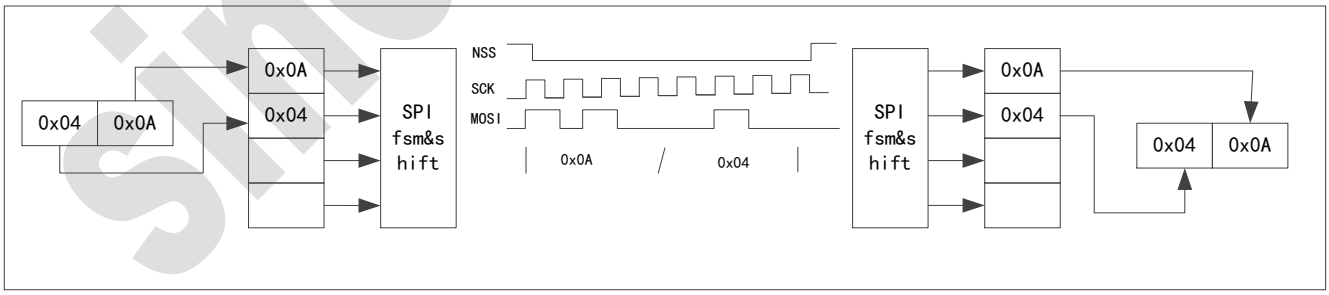
注： 如果用了打包模式，并且收到一个格式为小于或等于 8 位的奇数个数据的数据帧，当 FRLVL [1:0]= 1 时 FRXTH 必然被置 1 ，为的是产生 RXNE 事件以便去读最后的奇数数据。

数据打包

当数据帧的大小适合一个字节（小于或等于 8 位），并且对 SPIx_DR 寄存器执行任何 16 位的读写访问时，数据会自动打包在一起。 在这种情况下，可以并行处理双数据。 SPI 先操作低 8 位，然后操作高 8 位。 图 259 提供了打包方式顺序处理数据的一个例子。 在一次对 SPIx_DR 的 16 位写访问后，就会有 2 个字节的数据被发送出去。 如果 RXFIFO 的阈值设置为 16 位（FRXTH = 0），该序列则只会生成一个接收 RXNE 事件，而不是两个。 针对这种单个的 RXNE 事件的响应，接受器必须对 SPIx_DR 寄存器作一次 16 位的读访问才能够把数据全都取到。 RxFIFO 的阀值和跟进的数据访问的位宽必须保持一致，否则就会丢数据。

如果出现奇数个字节数据，那就会出现特别的问题，这是一定要解决的。 在发送端，用 8 位方式访问 SPIx_DR 将最后一个字节发出来就够了。 在接收端必须改变 Rx_FIFO 门限，以便在传送技术字节的数据帧的最后字节时能够产生 RXNE 事件。

传输和接收 FIFO 中的数据打包



使用 DMA（直接内存访问）通信

为了使 SPI 运行在其最高通讯速度，并且在方便数据寄存器读 / 写过程的同时避免溢出，SPI 需要用 DMA 支持，它实现了一个简单的请求 / 应答协议。

当在 SPIx_CR2 寄存器的 TXE 或 RXNE 位被置 1 时，可以产生一个 DMA 访问请求。 Tx 和 Rx 缓冲区的 DMA 请求是单独的。

- 在发送中，每次的 TXE 被置为 1，发出 DMA 请求。 然后 DMA 向 SPIx_DR 寄存器写数据。
- 在接收中，每次的 RXNE 被置为 1 时，发出 DMA 请求。然后 DMA 从 SPIx_DR 寄存器读数据。

当 SPI 仅用于发送数据，可以只打开 SPI TX DMA 通道。在这种情况下，溢出标志会不断置 1， 因为收到的数据不会被读取。 当 SPI 仅用于接收数据，可以只打开 SPI Rx DMA 通道。

在发送模式下，当 DMA 传输完所有要发送的数据时，DMA 控制器中 DMA_ISR 寄存器的 TCIF 标志会被置 1；监视

USART_SR 寄存器的 BSY 标志可以确认 SPI 通信是否结束。 这样可以在关闭 SPI 或进入停机模式之前避免破坏最后一次传输的数据。 软件必须先等待，直到 FTLVL [1:0] = 00，然后等 BSY = 0。

DMA 与打包传输

如果传输由 DMA 来管理（TXDMAEN 和 RXDMAEN 在 SPIx_CR2 寄存器设置），会根据 SPI TX 和 RX DMA 通道的 SPI 配置状态来自动将包装模式启用 / 禁用。如果 DMA 通道设置为按 16 位访问，而 SPI 数据的大小是小于或等于 8 位，那么打包模式会被启用。 DMA 会自动管理对 SPIx_DR 寄存器的写操作。

如果启用了打包模式，而传输的数据个数又不一定是偶数，则 LDMA_TX / LDMA_RX 位必须置 1。 而 SPI 则会认为在最后的 DMA 传输中只有一个有效的传输或接收字节（更多详情， 请见章节：数据打包。）

20.3.7 状态标志

应用程序可通过 3 个状态标志来了解 SPI 总线的全部状态。

TX 缓冲器空标志（TXE）

当 TXFIFO 中有了足够的空间来存储发送数据时，TXE 标志被置 1。 TXE 标志是连到 TXFIFO Level 的。在 TXFIFO 的的存储水平低于或等于整个 FIFO 的深度的一半的时候，这个标志会置高并且保持高。如果 SPIx_CR2 中的 TXEIE 位是 1，还会产生一个中断。如果 TXFIFO 的存储水平又高于 FIFO 深度的一半了，那这个位会自动的清零。

Rx 缓冲非空（RXNE）

RXNE 标志的设置取决于对在 SPIx_CR2 寄存器 FRXTH 位值：

- 如果 FRXTH 为 1，RXNE 会在 RXFIFO 的存储水平大于或等于 1/4（8 位）的时候被置高， 并且保持住。
- 如果 FRXTH 为 0，RXNE 会在 RXFIFO 的存储水平大于或等于 1/2（16 位）的时候被置高， 并且保持住。

如果这时 SPIx_CR2 寄存器中的 RXNEIE 位是 1，那就会产生一个中断请求。当上述条件不再成立时 RXNE 由硬件自动清零。

忙标志（BSY）

BSY 标志由硬件设置和清零（软件改写这个标志是没有用的）。

当 BSY 为 1 时，它表示 SPI 正处于数据传输过程中（SPI 总线忙）。

在某些模式下可以使用的 BSY 标志来检测传输结束，从而软件可以在进入 HALT 模式之前禁用 SPI 及其外设时钟。这就避免了破坏最后的传输字节。

BSY 标志在防止多主机系统中的写碰撞也是很有用的。 BSY 标志在下列条件之一达成时被清除：

- 当 SPI 被正确禁用时
- 当在主模式（MODF 位设置为 1）中检测到故障时
- 在主模式下，当完成了数据发送，并且没有新的数据要发送时
- 在从模式下，当两次数据传输之间间隔达到至少一个 SPI 时钟周期时

注：当下次传输可立即由主机来处理时（例如：如果主机是单接收模式或它的发送 FIFO 不为空），在主机这边的发送数据之间通讯不会中断，并且 BSY 标志仍然被保留为 1 。尽管从机不会有这个情况，ST 总是建议使用 TXE 和 RXNE 标志（而不是标志的 BSY ）来处理数据发送或接收操作。

20.3.8 错误标志

在将错误中断使能位 ERRIE 置 1 后，如果下列错误标志被置 1，就会产生 SPI 中断。

溢出标志（OVR）

当主机或从机在收到数据之后不去清除 RXNE 位，然后收到了新的数据之后，会引发溢出错误。当接收数据时 RXFIFO 中有没有足够的空间存储收到的数据时，溢出的情况也会发生。如果软件或者 DMA 没有足够时间去读取 RXFIFO 中的前面收到数据，来为后面的数据腾出足够的空间的时候就会发生这种事。在 CRC 功能打开时，接收缓冲区会被认为是一个单缓冲区， 这导致 RXFIFO 不好使了，这也会直接导致溢出事件。（见章节：CRC 计算）。

溢出的情况发生时，新收到的值不会覆盖在 RXFIFO 的前一个数据。 新收到的值将被丢弃， 随后传输的所有数据都将丢失。 在读了 SPIx_SR 寄存器后，再去读一下 SPIx_DR 寄存器， 就会清除 OVR 位。

模式故障位（MODF）

主机得知其内部 NSS 信号（NSS 引脚为硬件模式，或在 NSS 软件模式 SSI 位）被拉低时， 发生模式故障位。 这将自动设置 MODF 位。 主机模式故障对 SPI 接口的影响包括以下方面：

- MODF 位被置 1，另外如果 ERRIE 位被置 1，会产生 SPI 中断。
- SPE 位被清零。 这将阻止主机的数据输出，同时禁用 SPI 接口。
- MSTR 位被清除，从而迫使设备转到从机模式。

使用下面的软件序列，以清除 MODF 位：

1、在 MODF 位被设置后，对 SPIx_SR 寄存器做一次读或写访问。

2、然后写一下 SPIx_CR1 寄存器。

为了避免系统中的多个从机发生冲突，NSS 引脚必须在 MODF 位清零过程中拉高。 SPE 和 MSTR 位可以在这个清零过程后恢复到原来的状态。作为一种安全措施，硬件不允许在 MODF 位为 1 期间将 SPE 的 MSTR 位置 1。在从模式下，

MODF 位永远不可能被置 1，除非是以前的多主机冲突的结果。

CRC 错误（CRCERR）

在 SPIx_CR1 寄存器中的 CRCEN 位为 1 时，这个标志用来确认收到的数据的有效性。 如果接收移位寄存器中的值和接收 SPIx_RXCRCR 值不匹配，SPIx_SR 寄存器中的 CRCERR 标志会被置 1。 该标志由软件清除。

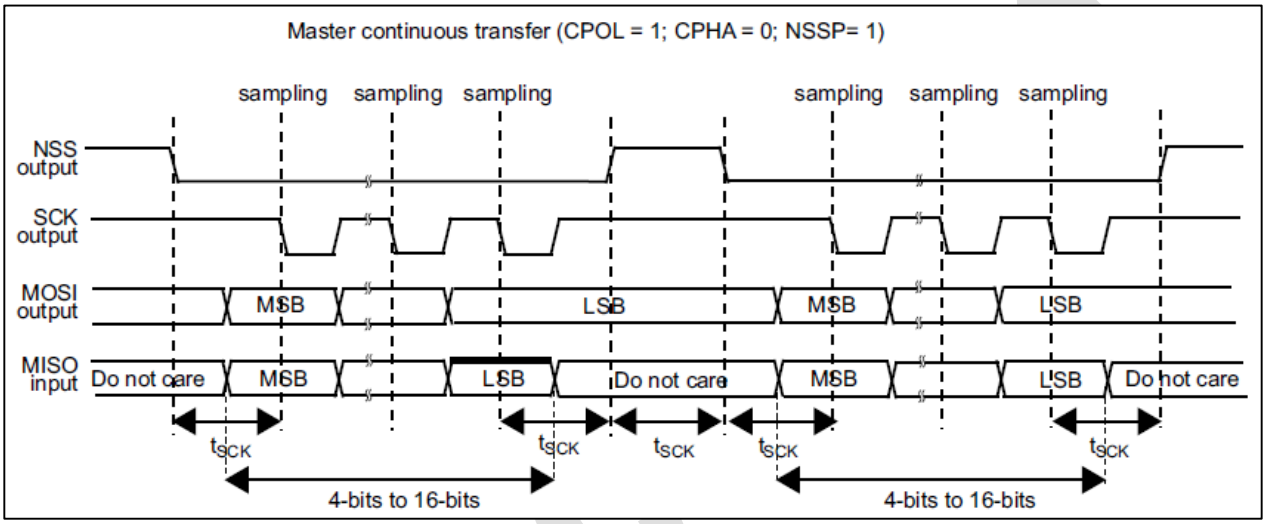
20.4 SPI 特殊功能

20.4.1 NSS 脉冲模式

在 SPIx_CR1 寄存器的 NSSP 位被置 1 时打开这个模式，只有在 SPI 接口被配置为摩托罗拉主模式，而且捕获第一个沿时（SPIx_CR1 CPHA = 0）才会生效。 这时发送时，NSS 脉冲会在两个连续的数据帧之间产生，并且 NSS 至少会在高电平保持一个时钟周期的时间。 这个模式允许从机锁存数据。 NSSP 脉冲模式为单一的主从应用而设计。

下图说明 NSSP 脉冲模式时启用时的 NSS 引脚管理。

摩托罗拉 SPI 主模式下 NSSP 脉冲的产生



注：当 CPOL = 0 时动作会类似。 这时，采样边沿是在 SCK 的上升沿，对 NSS 的采样也发生在这些沿。

20.4.2 CRC 计算

有两个独立的 CRC 计算器，分别用以检查发送和接收的数据的可靠性。 SPI 提供 CRC8 或 CRC16 计算，独立于帧的数据的位宽，它可以固定设置为 8 位或 16 位。 对于其他所有的数据位宽，CRC 无效。

CRC 原理

在使能 SPI 之前（SPE=0）通过设置 SPIx_CR1 寄存器中的 CRCEN 位，使能 CRC 计算。CRC 值由一个奇次可编程多项式按每位计算。 计算处理从采样时钟源开始执行，这个时钟沿由 SPIx_CR1 寄存器的 CPHA 和 CPOL 位定义。 在数据块传输结束的同时，计算出的 CRC 结果自动参与检查，由 CPU 或 DMA 控制传输的时候都一样。当发现 CRC 不匹配现象的时候，CRCERR 标志被置 1,表示 CRC 错误。对 CRC 计算的正确处理过程取决于 SPI 配置和所选择的传输管理。

注： 多项式的值应该是奇数。 不支持偶数值。

由 CPU 管理传输时的 CRC

在最后一个数据发送或接收完之前，SPI 的启动和传输都没什么特别的。

要在当前传输的数据之后跟上 CRC 数据，必须将 SPIx_CR1 中的 CRCNEXT 位设置为 1。 设置 CRCNEXT 位的操作必须在最后一个数据帧交易结束之前完成。 在 CRC 数据传输期间， CRC 计算会被冻结。

接收到的 CRC 数据像正常的数据字节或字存储在 RXFIFO 中。 这就是在 CRC 模式下，任何时候接受缓冲区都必须被认为是单个 16 位的缓冲区，并被用来接收单个数据的原因。

CRC 数据交换，通常需要在数据序列结束后，再占用一个或多个数据通信的时间。 当设置为 8 位的数据宽度，并做 16 位 CRC 检查时，发送完整的 CRC 数据就要两帧。

当收到最后的 CRC 数据后，会自动将收到的值和在 SPIx_RXCRC 寄存器的值进行比较。软件必须坚持 SPIx_SR 寄存器中的 CRCERR 标志以判断传输过程中数据是否被破坏。 软件对 CRCERR 标志写 0 就可清除它。

CRC 数据收到后，被存储在 RXFIFO 中，必须读 SPIx_DR 寄存器以清除 RXNE 标志。

由 DMA 管理传输时的 CRC

当 SPI 通讯功能中的 CRC 功能和 DMA 功能同时打开后，通讯尾声的 CRC 数据的发送和接收是全自动的。CRCNEXT 位不再必须由软件处理。 SPI 发送 DMA 通道计数器必须设置为不包括 CRC 数据的数量。而接受 DMA 通道计数器则要多包含一个 CRC 数据的长度，例如在 8 位数据宽度传输并使用 16 位 CRC 计算时：

DMA_RX = 数据字节数 + 2

在接收侧，接收到的 CRC 值在数据交换后依然存储在缓冲区中。 全部传输完后，如果发现有错误，SPIx_SR 寄存器中的 CRCERR 标志会被置 1。

SPIx_TXCRC 和 SPIx_RXCRC 值的重置

在 CRC 校验环节读取过 CRC 数据之后，SPIx_TXCRC 和 SPIx_RXCRC 值会自动清零。 这就允许使用 DMA 循环模式（单接收模式除外）来实现没有任何间断的传输数据（中间有几个数据块涉及 CRC 检查阶段）。

如果要在 SPI 通信过程中禁用它，必须遵循以下顺序：

- 禁止 SPI
- 清除 CRCEN 位
- 使能 CRCEN 位
- 使能 SPI

注：当 SPI 为从模式的时候，一旦设置的 CRCEN 位，CRC 计算器对 SCK 从输入时钟敏感，无论 SPE 位的值是几，都会这样。为了避免任何错误的 CRC 计算，软件必须在时钟稳定的条件下（稳定状态）启用 CRC 计算。 当 SPI 配置为从机接口时，数据阶段和 CRC 阶段之间， NSS 内部信号需要保持低。

20.5 SPI 中断

在 SPI 通信期间，中断可以由以下事件来产生：

- ✧ 发送 TXFIFO 待装填
- ✧ 在接收 RXFIFO 中有收到的数据
- ✧ 主模式故障
- ✧ 溢出错误
- ✧ CRC 错误

中断可以分别的被启用和禁用。

SPI 中断请求

中断事件	事件标志	使能控制位
发送 TXFIFO 待装填	TXE	TXEIE
在接收 RXFIFO 中有收到的数据	RXNE	RXNEIE
主模式故障	MODF	ERRIE
溢出错误	OVR	
CRC 错误	CRCERR	
TI 帧格式错误	FRE	

20.6 相关寄存器

20.6.1 寄存器汇总表

基址：0x4001 3000				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**SPI_CR1**	0x0000 0000	SPI 控制寄存器 1	20.6.2
0x04	**SPI_CR2**	0x0000 0700	SPI 控制寄存器 2	20.6.3
0x08	**SPI_SR**	0x0000 0002	SPI 状态寄存器	20.6.4
0x0C	**SPI_DR**	0x0000 0000	SPI 数据寄存器	20.6.5
0x10	**SPI_CRCPR**	0x0000 0007	SPI 的 CRC 多项式寄存器	20.6.6
0x18	**SPI_TXCRCR**	0x0000 0000	SPI 发送 CRC 寄存器	20.6.7

20.6.2 SPI 控制寄存器（SPI_CR1）

SPI_CR1			地址偏移	0x00		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	BIDIMODE	BIDIOE	CRCEN	CRCNEXT	CRCL	RXONLY	SSM	SSI
R/W	rw	rw	rw	rw	rw	rw	rw	Rw

复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	LSBFIRST	SPE	BR[2:0]			MSTR	CPOL	CPHA
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15]	BIDIMODE	双向数据模式使能（Bidirectional data mode enable） 这个位允许使用普通的单线双向数据线进行半双工通信。当双向模式激活时，保持 RXONLY 位清零。 0：选择 2 线单向数据模式 1：选择单线双向数据模式
BIT[14]	BIDIOE	双向模式输出使能（Output enable in bidirectional mode） 和 BIDIMODE 位一起决定在“单线双向”模式下数据的输出方向 0：输出禁止（只收模式）； 1：输出使能（只发模式）。 *注 1：在主模式下，用 MOSI 引脚。在从模式下，用 MISO 引脚。*
BIT[13]	CRCEN	硬件 CRC 计算使能（Hardware CRC calculation enable） 0：CRC 计算禁用 1：CRC 计算使能 *注 1：只有当 SPI 被禁止（SPE=0），该位才可以被写入。*
BIT[12]	CRCNEXT	下一个发送 CRC （Transmit CRC next） 0：下一个发送的值来自 Tx 缓冲区。 1：下一个发送的值来自 Tx CRC 寄存器。 *注 1：在 SPI_DR 寄存器写入最后一个数据后应马上写入该位。*
BIT[11]	CRCL	CRC 长度（CRC length） 由软件置位和清零，用于选择 CRC 长度。 0：8 位 CRC 长度 1：16 位 CRC 长度 *注 1：只有当 SPI 被禁止（SPE=0），该位才可以被写入。*
BIT[10]	RXONLY	只接收（Receive only mode enabled） 此位启动单工通讯，使用单线单工只接收通讯。 激活只接收模式时，保持 BIDIMODE 位清零。此位也可用于多从设备系统中，此特定的从设备不会被访问，被访问的从设备输出不会被破坏。 0：全双工（发送和接收） 1：输出禁止（只收模式）；
BIT[9]	SSM	软件从机管理（Software slave management） 当 SSM 被置位时，NSS 引脚上的电平由 SSI 位的值决定。 0：禁止软件从设备管理； 1：启用软件从设备管理
BIT[8]	SSI	内部从设备选择（Internal slave select） 此位只有当 SSM 位为 1 时才有效。此位的值被强制映射 NSS 引脚上，NSS 引脚上的 I/O 操作无效。 SSI 仅在 SPE=0 条件下可配置修改，SPE=1 时无法修改
BIT[7]	LSBFIRST	帧格式（Frame format） 0：数据发送/接收 MSB 在前； 1：数据发送/接收 LSB 在前。 *注 1：当通信在进行时不能改变该位的值。*
BIT[6]	SPE	SPI 使能 SPI enable 0：禁止 SPI 设备； 1：开启 SPI 设备。 *注 1：当关闭 SPI 设备时，请按照<禁用 SPI 步骤>操作*
BIT[5:3]	BR[2:0]	波特率控制 Baud rate control 000：保留 001：fPCLK/4 010：fPCLK/8 011：fPCLK/16 100：fPCLK/32 101：fPCLK/64 110：fPCLK/128

		111: fPCLK/256 *注 1：当通信在进行时不能改变该位的值。*
BIT[2]	MSTR	主设备选择 Master selection 0：配置为从设备 1：配置为主设备 *注 1：当通信在进行时不能改变该位的值。*
BIT[1]	CPOL	时钟极性 Clock polarity 0：空闲状态时，SCK 保持低电平； 1：空闲状态时，SCK 保持高电平。 *注 1：当通信在进行时不能改变该位的值。*
BIT[0]	CPHA	时钟相位 Clock phase 0：第一个时钟沿对准第一位数据 1：第二和时钟沿对准第一位数据 SPI 从机模式下不支持 CPOL=CPHA=1 的配置 *注 1：当通信在进行时不能改变该位的值。*

20.6.3 SPI 控制寄存器 2(SPI_CR2)

SPI_CR2			地址偏移	0x04		复位值	0x0000 0700	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	LDMA_TX	LDMA_RX	FRXTH	DS			
R/W	-	rw	rw	rw	rw	rw	rw	rw
复位值	-	0	0	0	0	1	1	1
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	TXEIE	RXNEIE	ERRIE	-	NSSP	SSOE	TXDMAEN	RXDMAEN
R/W	rw	rw	rw	-	rw	rw	rw	rw
复位值	0	0	0	-	0	0	0	0

BIT[14]	LDMA_TX	最后一次 DMA 发送 Last DMA transfer for transmission 此位是在数据打包模式中，用来定义 DMA 数据传输的总数是奇数还是偶数。它只有在 SPIx_CR2 寄存器的 TXDMAEN 位被置位，并且打包模式已开启（数据长度≤8 位和写入访问 SPIx_DR 是 16 位宽）时有意义。它必须在 SPI 被禁止（SPIx_ CR1 寄存器的 SPE = 0）时写入。 0：传输的数据数量为偶数 1：传输的数据数量为奇数 *注 1：若置位了 CRCEN 位，请按照<禁用 SPI 步骤>操作。*
BIT[13]	LDMA_RX	最后一次 DMA 接收 Last DMA transfer for reception 此位是在数据打包模式中，用来定义 DMA 数据接收的总数是奇数还是偶数。它只有在 SPIx_CR2 寄存器的 RXDMAEN 位被置位，并且打包模式已开启（数据长度 ≤ 8 位和写入访问 SPIx_DR 是 16 位宽）时有意义。它必须在 SPI 被禁止（SPIx_ CR1 寄存器的 SPE = 0）时写入。 0：传输的数据数量为偶数 1：传输的数据数量为奇数 *注 1：若置位了 CRCEN 位，请按照<禁用 SPI 步骤>操作。*
BIT[12]	FRXTH	FIFO 接收门限 FIFO reception threshold 此位用来设置触发 RXNE 事件时的 RXFIFO 的阈值。 0：如果 FIFO 的存储水平大于或等于 1/2（16 位），产生 RXNE 事件。 1：如果 FIFO 的存储水平大于或等于 1/4（8 位），产生 RXNE 事件。
BIT[11:8]	DS[3:0]	数据位宽 Data size 这些位配置 SPI 传输数据的位宽： 0000：不使用

		0001：不使用 0010：不使用 0011：4 位 0100：5 位 0101：6 位 0110：7 位 0111：8 位 1000：9 位 1001：10 位 1010：11 位 1011：12 位 1100：13 位 1101：14 位 1110：15 位 1111：16 位 *注 1：如果软件试图写一个“不使用”的值，他们会被迫赋值“ 0111 ”（ 8 位）。*
BIT[7]	TXEIE	TX 缓冲器空中断使能 Tx buffer empty interrupt enable 0：TXE 中断屏蔽 1：TXE 中断没有被屏蔽。 用于在 TXE 标志置 1 的时候产生一个中断请求。
BIT[6]	RXNEIE	RX 缓冲区非空中断使能 RX buffer not empty interrupt enable 0：RXNE 中断屏蔽 1：RXNE 中断没有被屏蔽。 用于在 RXNE 标志置 1 的时候产生一个中断请求。
BIT[5]	ERRIR	错误中断使能 Error interrupt enable 这个位控制在出现错误事件时是否产生中断。（SPI 模式中的 CRCERR、OVR、MODF，TI 模式中的 FRE， 和 I2S 模式中的 UDR、OVR、FRE） 0：错误中断屏蔽 1：错误中断使能
BIT[4]	-	保留，保持复位值
BIT[3]	NSSP	NSS 脉冲管理 NSS pulse management 该位仅在主模式下使用。 它允许 SPI 在连续传输时，两个数据传输之间产生一个 NSS 脉冲。在单个数据传输的情况下，它会在传输结束后将 NSS 脉强制为高电平。在 CPHA=1 或 FRF=1 的时候，这个位没有意义。 0：没有 NSS 脉冲 1：产生 NSS 脉冲 *注 1：该位必须在 SPI 被禁止（ SPE = 0 ）的时候写入*
BIT[2]	SSOE	SS 输出使能 SS output enable 0：在主模式下 SS 输出被禁用，SPI 接口可以工作在多主机的配置下。 1：SPI 接口启用的同时在主模式下启用 SS 输出。 SPI 接口不能在多主机环境下工作。
BIT[1]	TXDMAEN	Tx 缓冲 DMA 使能 Tx buffer DMA enable 当此位被置位，TXE 标志被置位时，产生一个 DMA 请求。 0：Tx 缓冲 DMA 禁止 1：Tx 缓冲 DMA 使能
BIT[0]	RXDMAEN	Rx 缓冲 DMA 使能 Rx buffer DMA enable 当此位被置位，RXNE 标志被置位时，产生一个 DMA 请求。 0：Rx 缓冲 DMA 禁止 1：Rx 缓冲 DMA 使能

20.6.4 SPI 状态寄存器(SPI_SR)

SPI_SR			地址偏移	0x08		复位值	0x0000 0002	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-

R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	FTLVL[1:0]		FRLVL[2:0]		-
R/W	-	-	-	r	r	r	r	-
复位值	-	-	-	0	0	0	0	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	BSY	OVR	MODF	CRCERR	-	-	TXE	RXNE
R/W	r	r	r	rc_w0	r	r	r	r
复位值	0	0	0	0	0	0	1	0

BIT[12:11]	FTLVL[1:0]	FIFO 发送水平 FIFO Transmission Level 由硬件置位或清零 00: FIFO 空 01: 1/4 FIFO 10: 1/2 FIFO 11: FIFO 满（当 FIFO 门限大于 1/2 时认为是满）
BIT[10:9]	FRLVL[1:0]	FIFO 接收水平 FIFO reception level 由硬件置位或清零 00: FIFO 空 01: 1/4 FIFO 10: 1/2 FIFO 11: FIFO 满 *注 1：这些位在 SPI 仅接收模式下且开启 CRC 计算功能时，不可用。*
BIT[8]	-	保留，保持复位值
BIT[7]	BSY	忙标志 Busy flag，由硬件置位和清除。 0: SPI 不忙 1: SPI 通信忙或 Tx 缓冲区不为空 *注：必须谨慎使用的 BSY 标志： 参见章节<SPI 状态标志>和<禁用 SPI 步骤>。*
BIT[6]	OVR	溢出标志 Overrun flag 0: 没有发生溢出 1: 发生溢出 此标志由硬件置位，由软件序列复位。
BIT[5]	MODF	模式故障 Mode fault 0: 无模式故障发生 1: 模式故障发生 此标志由硬件置位，由软件序列复位。 软件序列，请参阅章节<模式故障（MODF）>。
BIT[4]	CRCERR	CRC 错误标志 CRC error flag 0: 收到的 CRC 值和 SPIx_RXCRCR 的值是匹配的 1: 收到的 CRC 值和 SPIx_RXCRCR 值不匹配 此标志由硬件置位，由软件写 0 清零。
BIT[3:2]	-	保留，保持复位值
BIT[1]	TXE	发送缓冲区空标志 0: Tx 缓冲区非空 1: Tx 缓冲区空
BIT[0]	RXNE	接收缓冲区非空标志 0: RX 缓冲区空 1: Rx 缓冲区非空

20.6.5 SPI 数据寄存器(SPI_DR)

SPI_DR			地址偏移	0x0C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16

控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	DR[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	DR[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	DR[15:0]	数据寄存器 Data register 待发送或者已经收到的数据 数据寄存器作为 Rx 和 Tx FIFOs 的接口。 当读取数据寄存器时，RXFIFO 会被访问，而写入数据寄存器则会访问 TXFIFO（见章节：数据发送和接收流程）。 *注：数据始终是右对齐。 当写寄存器的时候，没有用到的位会被忽略，读的时候，没有用到的位会是 0。 接收门限的设置必须始终符合目前使用的读访问方式。*

20.6.6 SPI 的 CRC 多项式寄存器(SPI_CRCPR)

SPI_CRCPR			地址偏移	0x10		复位值	0x0000 0007	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CRCPOLY[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CRCPOLY[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	1	1	1

BIT[15:0]	CRCPOLY[15:0]	CRC 多项式寄存器 该寄存器包含 CRC 计算多项式。 该寄存器的复位值是（0007H）。 根据需要，可以配置成另一个多项式。 *注：多项式的值应该是奇数。 不支持偶数值*

20.6.7 SPI 发送 CRC 寄存器(SPI_TXCRCR)

SPI_TXCRCR			地址偏移	0x18		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	TxCRC[15:8]							

R/W	rw	rw	rw	Rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	TxCRC[7:0]							
R/W	rw	rwr	rw	rw	rwr	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:0]	TxCRC [15:0]	Tx CRC 寄存器 当启用 CRC 计算功能，TxCRC [7:0] 位包含根据发送的字节计算出来的 CRC 值。当 SPIx_CR1 寄存器的 CRCEN 位被写为 1 的时候，这个寄存器被复位。CRC 串行计算使用 SPI_CRCPR 中的多项式。 当数据帧格式被设置为 8 位（SPIx_CR1 中的 CRCL 位为 0）时，仅低 8 位参与计算， 并且按照 CRC8 标准进行； 当数据帧格式为 16 位(SPIx_CR1 的 CRCL 位为 1)时，寄存器中的所有 16 位都参与计算， 并且按照 CRC16 标准进行； *注 1： 在 BSY 标志为 1 的期间读这个寄存器的值，可能得到一个不正确结果。*

21 I2C接口

此模块当前版本暂不开放，有相关应用需求请联系我司技术支持人员。

22 USART接口

22.1 概述

器件内置有一个通用同步/异步收发器 （USART）。

支持工业标准 NRZ 编码，可实现 MCU 与设备的全双工异步通讯。

USART 支持同步单向通信和单线半双工通信模式以及多处理器通信模式，还支持 LIN(Local Interconnect Network) 智能卡通信（ISO7816）、 IrDA(Infrared Data Association) SIR ENDEC 及调制解调器操作（CTS/RTS）。

支持 DMA 操作，通过多缓冲设置，实现高速数据通讯。

USART 功能配备

USART 模式/功能	USART1
MODEM 所需的硬件流控制	√ *注*
用 DMA 实现连续通讯	√
多机通讯	√
同步模式	√
智能卡模式	√
单线半双工通讯	√
红外 IrDA SIR 编解码	√
LIN 模式	√
双时钟驱动和从 Stop 模式唤醒	√
接收超时中断	√
Modbus 通讯	√
自动波特率检测	√
RS485 用的驱动使能信号	√

注： √ = 支持

22.2 特性

✧ 全双工异步通信
✧ NRZ 标准格式（mark/space 校验）
✧ 可配置的 16 位或 8 位过采样方法提供速度和时钟容忍度间的灵活选择
✧ 通用可编程收发波特率可达 6Mbit/s（@时钟 48MHz，8 倍过采样）
✧ 支持双时钟域：
- STOP 模式下，USART 功能唤醒
- 方便的波特率编程，独立于 PCLK 时钟
✧ 自动波特率检测
✧ 数据字节长度可编程（8 或 9 位）
✧ 可编程数据位序，LSB 在前或 MSB 在前
✧ 停止位可编程（1 个或 2 个停止位）
✧ 支持同步模式及时钟输出
✧ 单线半双工通讯
✧ 使用 DMA 实现连续通讯
✧ 使用 DMA 将收/发字节缓冲到保留的 SRAM
✧ 接收器和发送器独立使能位
✧ 接收器和发送器独立极性控制
✧ 可配置 Tx/Rx 引脚互换
✧ 用于调制解调器和 RS485 发送器的硬件流控制
✧ 通讯控制/错误检测标志
✧ 奇偶控制
- 发送奇偶位
- 接收数据字节奇偶检查
✧ 14 个中断源和中断标志
✧ 多机通信
- 若地址不匹配，进入静默模式
✧ 静默模式唤醒（检测到线路空闲或检测到地址标记）

22.3 扩展特性

✧ LIN 主机的断开信号发送能力和 LIN 从机的断开信号检测能力
- 将 USART 硬件设置成 LIN 模式时，有 13 位的断开信号发生器和 10/11 位的断开信号检测功能
✧ IrDA SIR 编解码器，普通模式下支持 3/16 位时长
✧ 智能卡模式
- 支持 ISO/IEC7816-3 标准定义的 T=0 和 T=1 智能卡异步协议
- 0.5 和 1.5 个停止位，用于智能卡操作
✧ 支持 ModBus 通讯
- 超时功能
- CR/LF 字符识别

22.4 功能描述

任何 USART 双向通讯要求最少有两个引脚： 接收数据输入（RX）和发送数据输出（TX）。

✧ RX：接收数据输入
串行数据的输入口。 使用过采样技术来完成数据恢复，以区别输入数据和噪声。
✧ TX：数据发送输出
当发送器被禁止，输出脚回到其 I/O 口配置状态。当发送器被使能，但不发送数据时，TX 脚为高电平输出。 在单线和智能卡模式中，这个口线既用于发送数据也用于接收数据。

数据帧包含：

✧ 在发送和接收之前为空闲状态
✧ 起始位
✧ 数据字（8 或 9 位），LSB 在前
✧ 0.5/1/1.5/2 个停止位，表明帧的结束
✧ 一个波特率发生器
✧ 接收和发送数据寄存器（USART_RDR，USART_TDR）
✧ 一个波特率寄存器（USART_BRR）
✧ 一个保护时间寄存器（USART_GTPR）用于智能卡模式

同步模式和智能卡模式需要使用下面引脚：

✧ CK：时钟输出。该引脚输出发送器数据的同步时钟。同步发送与 SPI 主模式一致(起始位和停止位上没有时钟脉冲，在最后一位数据上可选择有无时钟脉冲)。同时，数据可以在 RX 上同步接收。这可以用来控制有移位寄存器的外围设备(例如 LCD 驱动程序)。时钟相位和极性是软件可编程的。在智能卡模式下，CK 输出可以为智能卡提供时钟。

下列引脚用于支持硬件流控制模式：

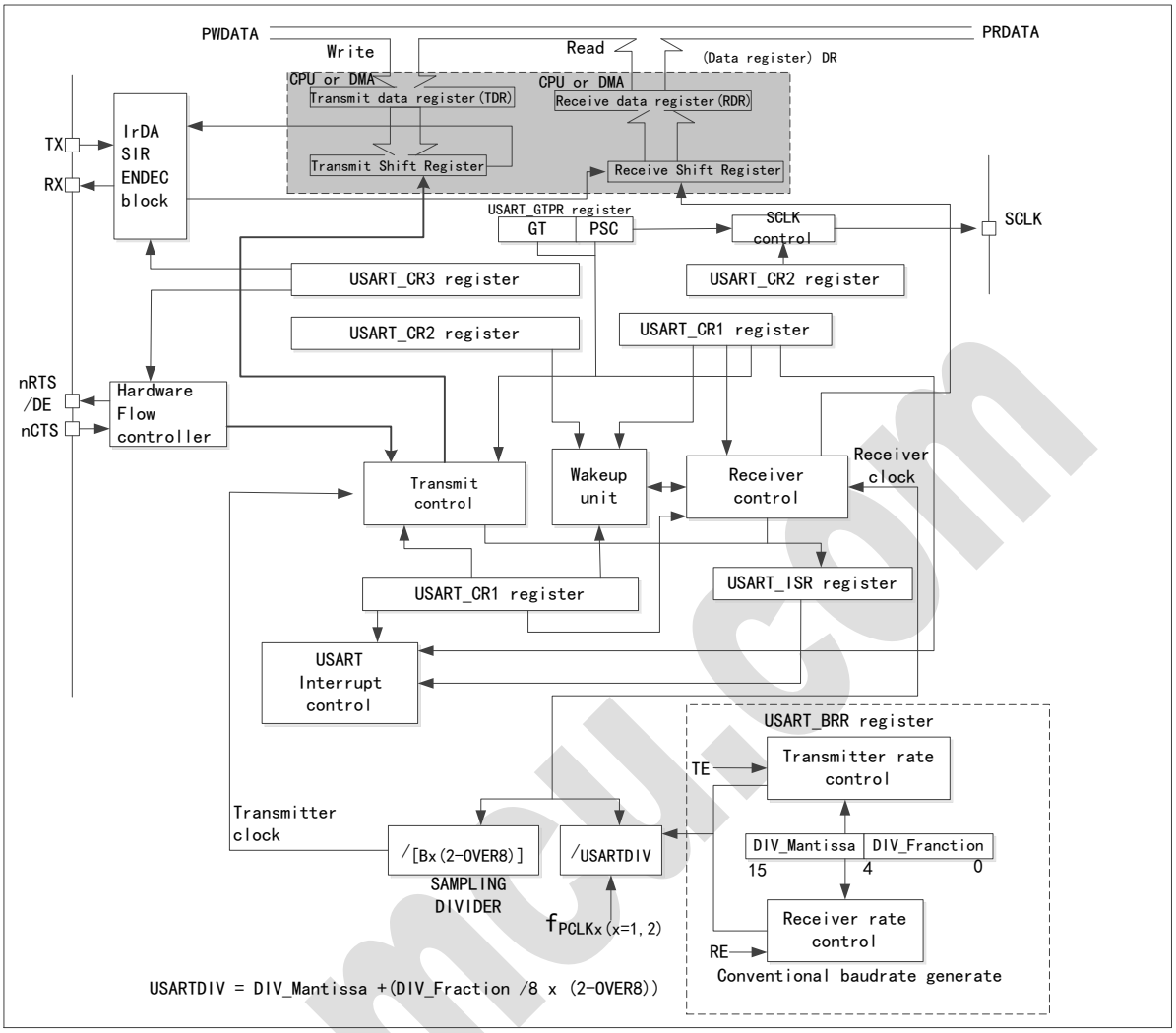
✧ CTS：(Clear To Send)发送允许，低有效，高电平作为发送阻塞信号
✧ RTS：(Request to send)，请求发送，表明 USART 已经准备好接收数据（低有效）。

下列引脚用于支持硬件流控制模式：

✧ DE：驱动使能将外部收发器的发送模式激活

注：DE 和 RT 共用一个外部引脚。

USART 模块框图



注 1：有关在 USART_BRR 寄存器中编码 USARTDIV 的详细信息，请参阅章节:USART 波特率生成。
注 2：f_{CK} 可以为 f_{LSE}, f_{HSI}, f_{SYS}。

22.4.1 USART 符号描述

配置 USART_CR1 寄存器（参见下图）中的 M[1:0] 位可选择 7 位、8 位或 9 位字长。

- M[1:0]=10B：7 位字符长度
- M[1:0]=00B：8 位字符长度
- M[1:0]=01B：9 位字符长度

注:在 7 位数据长度模式下,不支持智能卡模式、LIN 主模式和 Autobaudrate (0x7F 和 0x55 帧检测)。只有部分 USARTs 支持 7 位模式。

默认设置中，信号（RX 和 TX）在起始位都是低电平，而停止位都是高电平。

这个逻辑可以在极性控制中单独的设置为反向。

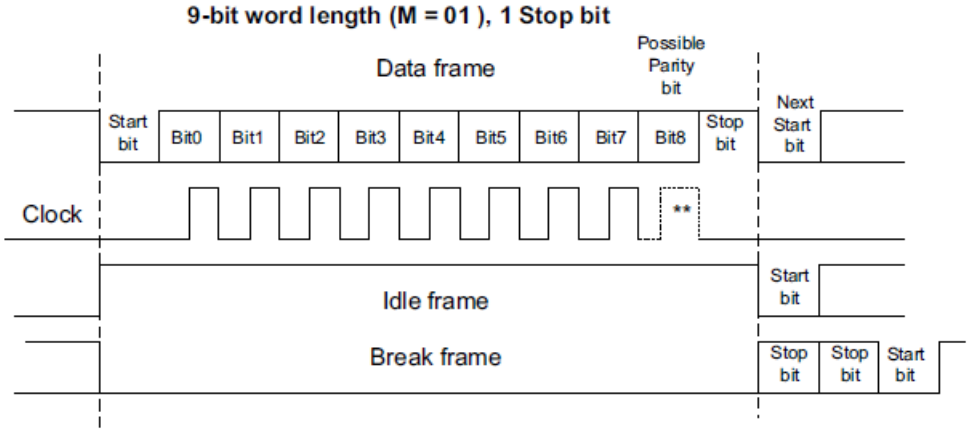
空闲帧，被视为完全由 ’1’ 组成的完整的数据帧 （’1’的位数也包括了停止位的位数 ）。

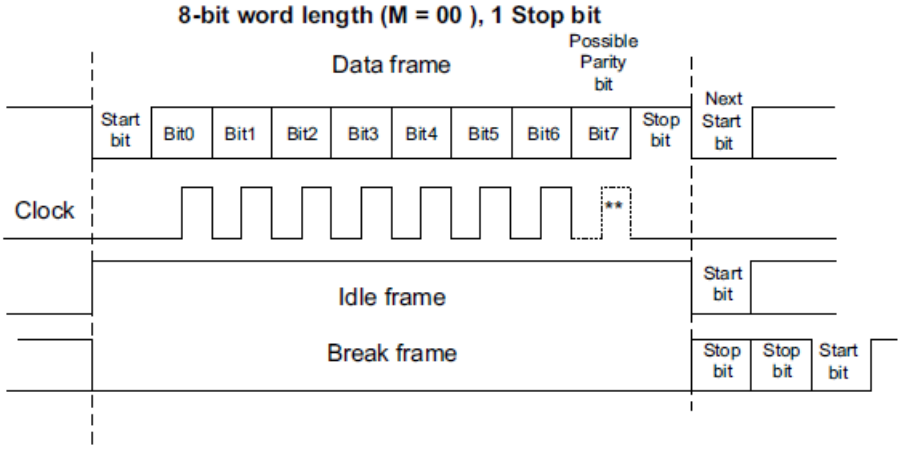
断开帧，被视为在一个帧周期内全部收到’0’（ 包括停止位期间，也是’0’）。 在断开帧结束时，发送器会再插入 2 个停止位。

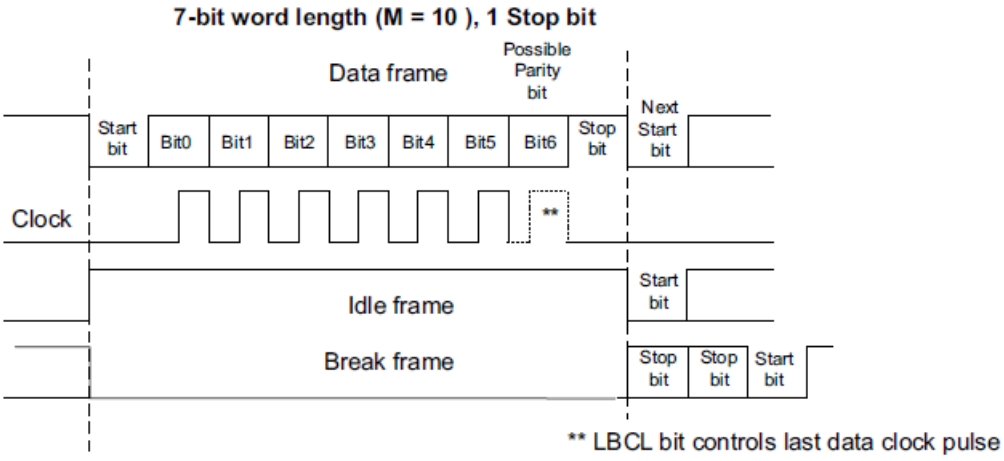
发送和接收由一个共用的波特率发生器驱动，当发送器和接收器的使能位分别置 1 时，分别为其产生时钟。

下面是每个模块的详细说明

字长编程







22.4.2 USART 发送器

发送器根据 M 位的状态发送 7 位、8 位或 9 位的数据字。

发送使能位（TE）必须置 1 以打开发送功能。 发送移位寄存器中的数据在 TX 脚上输出，相应的时钟脉冲在 CK 脚上输出。

字符发送

在 USART 发送期间，在 TX 引脚上首先移出数据的最低有效位（默认配置下）。 在此模式里，USART_TDR 寄存器充当了一个内部总线和发送移位寄存器之间的缓冲器（TDR）（见下图）。

每个字符之前都有一个低电平的起始位，之后跟着停止位，停止位的数目是可选择的。

USART 支持多种停止位的选择：0.5、1、1.5 和 2 个停止位。

注 1：在向 USART_TDR 写数据之前必须先令 TE 位为 1 。

注 2：TE 位在数据传输过程中不应复位。在传输过程中复位 TE 位将破坏 TX 引脚上的数据，因为波特率计数器将被

冻结。当前正在传输的数据将丢失。

注 3：在 TE 位被置 1 后将会发送一个空闲帧。

可配置的停止位

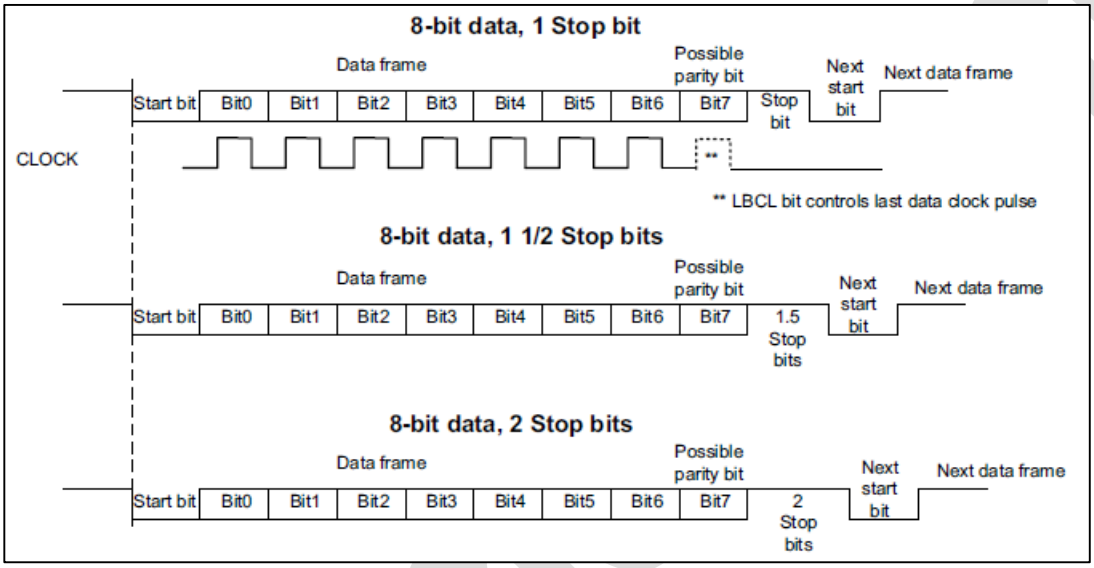
每个数据帧发送的停止位的位数可以通过控制寄存器 2（USART_CR2）的位 STOP[1:0]进行编程。

- ✧ 1 个停止位： 停止位的位数的默认值。
- ✧ 2 个停止位： 可用于常规 USART 模式、单线模式以及调制解调器（modem）模式。
- ✧ 1.5 个停止位： 在智能卡模式下发送和接收数据时使用。
- ✧ 0.5 个停止位：在智能卡模式下接收数据时使用。

空闲帧包括了停止位。

断开帧是 10 位低电平（当 M[1:0]=00 时)或者 11 位低电平（M[1:0]=01 时）或者 9 位低电平（M[1:0]=10 时），后跟 2 个停止位。 没办法传输更长的断开帧 （ 断开长度大于 9/10/11 位 ）。

可配置的停止位



配置步骤：

1、设置 USART_CR1 的 M 位来定义字长。

2、利用 USART_BRR 寄存器选择预期的波特率。

3、在 USART_CR2 中设置停止位的位数。

4、通过将 USART_CR1 寄存器的 UE 位置 1 来使能 USART。

5、如果采用多缓冲器通信，配置 USART_CR3 中的 DMA 使能位 (DMAT)。按多缓冲器通信中的描述配置 DMA 寄存器。

6、设置 USART_CR1 中的 TE 位，发送一个空闲帧作为第一次数据发送。

7、把要发送的数据写进 USART_TDR 寄存器 (此动作将清除 TXE 位)。 在只有一个缓冲器的情况下，对每个待发送的数据重复步骤 7。重复步骤 7 之前应等待 TXE 变成 1。

8、在 USART_TDR 寄存器中写入最后一个数据字后，要等待 TC=1， 它表示最后一个数据帧的传输结束。例如当关闭 USART 或需要进入 Halt 模式之前，需要确认传输结束，避免破坏最后一次传输。

单字节通信

清零 TXE 位总是通过对数据寄存器的写操作来完成的。

TXE 位由硬件来设置，它表明：

- ✧ 数据已经从 TDR 移送到移位寄存器，数据发送已经开始。
- ✧ USART_TDR 寄存器被清空
- ✧ 下一个数据可以被写进 USART_TDR 寄存器而不会覆盖先前的数据。

如果 TXEIE 位被设置，此事件将产生一个中断请求。

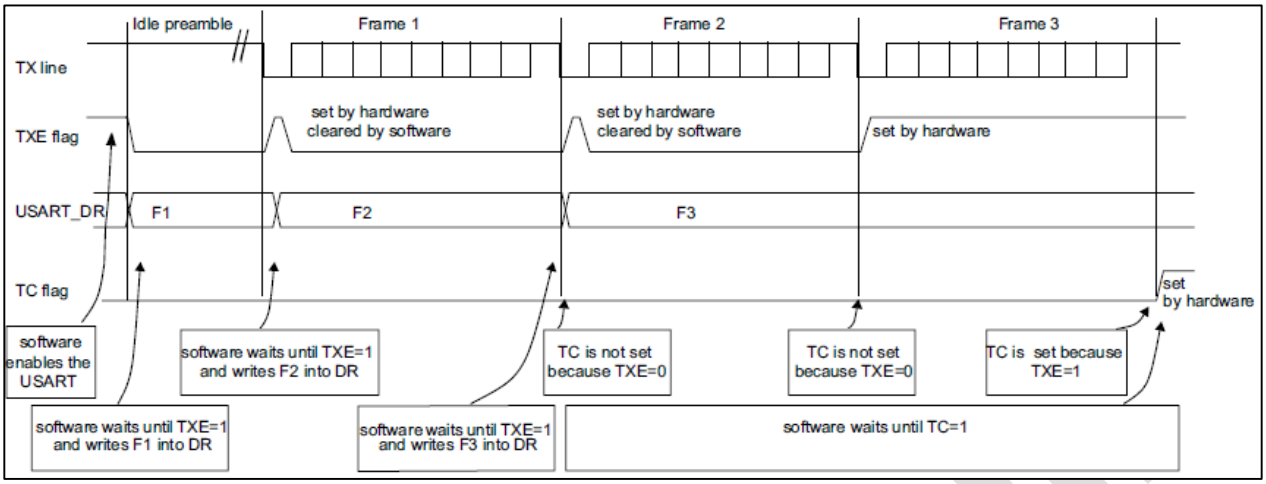
如果此时 USART 正在发送数据，对 USART_TDR 寄存器的写操作把数据存进 TDR 寄存器， 并在当前传输结束时把该数据复制进移位寄存器。

如果此时 USART 没有在发送数据，对 USART_TDR 寄存器的写操作将导致直接把数据放进移位寄存器，数据传输开始，TXE 位则立即被置起。

当一帧字节发送完成 (停止位发送后) 且 TXE 已置 1 时，TC 位会被置 1。 如果 USART_CR1 寄存器中的 TCIE 位是 1 时，则会产生中断。

在 USART_TDR 寄存器中写入了最后一个数据字后，在关闭 USART 模块之前或设置微控制器进入低功耗模式之前，必须先等待 TC=1。（见图： 发送时 TC/TXE 行为）

发送时 TC/TXE 行为



关闭 USART 的正确步骤如下：

1、清除 TE 位（如果有数据正在传输或者 USART_ 寄存器中还有待发送的数据，会在关闭动作生效前发送完）。

2、TC 位会在发送完毕后被置 1。

3、在 TC=1 后清除 UE 位。

断开符号（break characters）

向 SBKRQ 位写 1 可发送一个断开符号。断开符号的长度取决于 M[1:0] 位 。

如果写 SBKRQ =1，在完成当前数据发送后，将在 TX 线上发送一个断开符号。 SBKF 位会在写操作发生时置 1，在断开符号发送完成时 （ 在断开符号的停止位发出时 ） 被硬件清零。USART 在最后一个断开帧的结束处插入一个逻辑’1’并保持 2 位时长作为停止，以保证能识别下一帧的起始位。

如果应用程序需要在所有先前插入的数据(包括尚未传输的数据)之后发送 break 字符，则软件应该在设置 SBKRQ 位之前等待 TXE 标志置起。

空闲符号（idle characters）

将 TE 置 1 将使得 USART 在第一个数据前发送一空闲符号。

22.4.3 USART 接收器

接收器根据 USART_CR1 寄存器中 M 位的状态接收 7 位、8 位或 9 位的数据字。

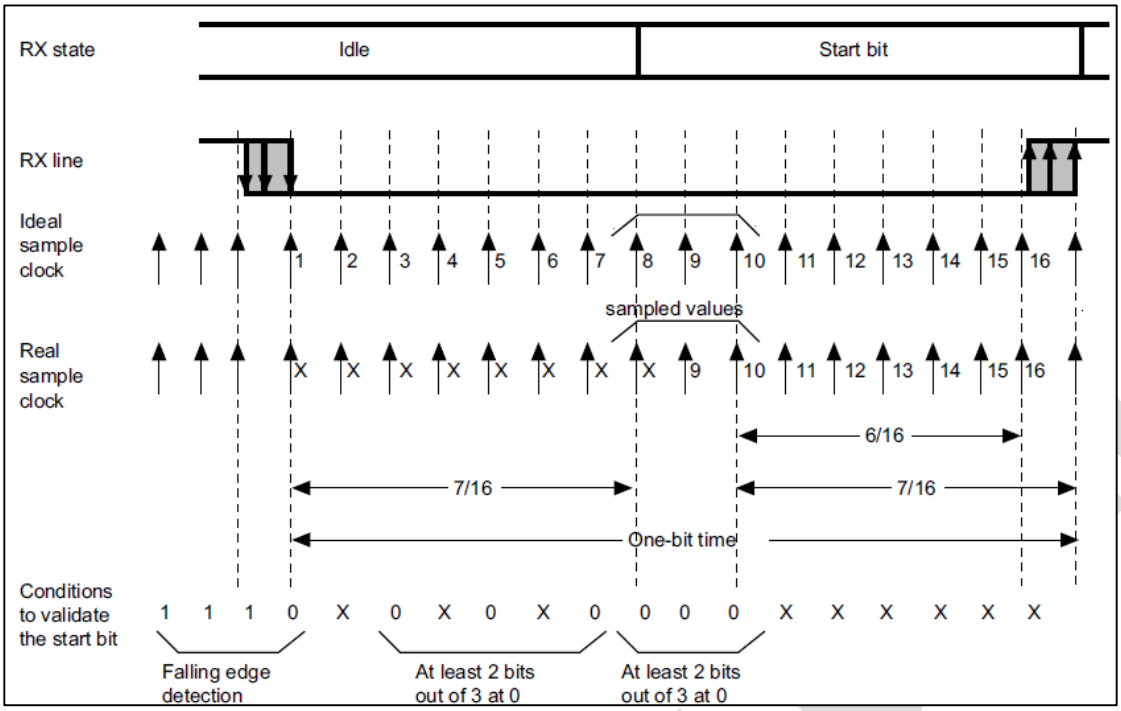
起始位侦测

过采样率设置成 16 或 8，起始位侦测序列相同。

在 USART 中，如果识别出一个特定的采样序列，那么就认为侦测到一个起始位。

该序列为：1 1 1 0 X 0 X 0 X 0 X 0 X 0 X 0。

过采样率为 16 或 8 时的起始位侦测



注：如果该序列不完整，那么接收端将退出起始位侦测并回到空闲状态(不置标志位)开始等待下降沿。

如果 3 个采样点都为’0’（ 在第 3、5、7 位的第一次采样，和在第 8、9、10 的第二次采样都为’0’），则确认收到起始位，这时 RXNE 标志会由硬件置 1，如果这时 RXNEIE=1，则会产生中断请求。

如果下面 2 个条件成立，那么起始位仍然是有效的（RXNE 标志会置 1，如果 RXNEIE=1，则会产生中断请求），但是会设置 NE 噪声标志位。

- ✧ 两组采样（第 3、5、7 位的采样点和第 8、9、10 位的采样点）中，3 个采样点有 2 次采样为‘0’；
- ✧ 两组采样中的一组（第 3、5、7 位的采样点或第 8、9、10 位的采样点），3 个采样点有 2 次采样为‘0’。

如果两个条件都不满足，则起始位检测终止，接收器返回到空闲状态(不置位标志位)。

字符接收

在 USART 接收期间，数据的最低有效位（默认情况下）首先从 RX 脚移进。在此模式里，USART_RDR 寄存器充当了一个位于内部总线和接收移位寄存器之间的缓冲器。

配置步骤：

1、设置 USART_CR1 的 M 位来定义字长。

2、利用波特率寄存器 USART_BRR 选择预期的波特率。

3、在 USART_CR2 中设置停止位的位数。

4、通过将 USART_CR1 寄存器的 UE 位置 1 来激活 USART。

5、如果采用多缓冲器通信，配置 USART_CR3 中的 DMA 使能位 (DMAR)。 按多缓冲器通信中的描述配置 DMA 寄存器。

6、将 USART_CR1 的 RE 位置 1。 这将激活接收器，使它开始寻找起始位。

当一个字符被接收到时：

- ✧ RXNE 位被置 1。 它表明移位寄存器的内容被转移到 RDR。 换句话说，数据已经被接收并且可以被读出（ 包括与之有关的错误标志 ）。
- ✧ 这时如果 RXNEIE 位是 1，将会引起中断请求。
- ✧ 在接收期间如果检测到帧错误，噪音或溢出错误，错误标志将被置起。 PE 标志也会和 RXNE 一起被置 1。
- ✧ 在多缓冲器通信时，RXNE 在每个字节接收后被置起，并由 DMA 对数据寄存器的读操作而清零。
- ✧ 在单缓冲器模式里，由软件读 USART_RDR 寄存器完成对 RXNE 位清除。 RXNE 标志也可以通过对 USART_RQR 寄存器中的 RXFRQ 位写 1 来清除。 RXNE 位必须在下一字符接收结束前被清零，以避免溢出错误。

断开符号

当接收到一个断开帧时，USART 会像处理帧错误一样处理它。

空闲符号

当一空闲帧被检测到时，其处理步骤和接收到普通数据帧一样，但如果 IDLEIE 位为 1 将产生一个中断请求。

溢出错误

如果 RXNE 还没有被复位，而又接收到了一个字符，则发生溢出错误。 数据只有当 RXNE 位被清零后才能从移位寄存器转移到 RDR 寄存器。

RXNE 标记是接收到每个字节后被置 1 的。 如果下一个数据已被收到或先前 DMA 请求还没被服务时，RXNE 标志仍是置起的，也会产生溢出错误。 当溢出错误产生时：

- ORE 位被置 1。
- RDR 内容将不会丢失。 读 USART_RDR 寄存器仍能得到先前的数据。
- 移位寄存器中以前的内容将被覆盖。 随后接收到的数据都将丢失。
- 如果 RXNEIE 位被置为 1 或 EIE 位被置为 1，会产生中断。
- ORE 位可以通过将 ICR 寄存器中的 ORECF 位置 1 的方式清除。

注：当 ORE 位置 1 时，表明至少有 1 个数据已经丢失。有两种可能性：

- *如果 RXNE=1，尚一个有效数据还在接收寄存器 RDR 上，可以被读出。*
- *如果 RXNE=0，这意味着上一个有效数据已经被读走，RDR 已经没有东西可读。 当上一个有效数据在 RDR 中被读取的同时又接收到新的（ 也就是丢失的 ）数据时，此种情况是可能发生的。*

选择时钟源和适当的过采样方法

时钟源的选择要通过时钟控制系统（参见章节：复位和时钟控制系统（RCC））。必须在使能 USART 之前（将 UE 位置 1）从而打开 USART 的时钟源。

时钟源的选择需要依据两个标准：

- 在低功耗模式下使用串口的可能性
- 通讯速度

时钟源的频率是 f_{CK}。

当使用双时钟域支持 Stop 模式唤醒的功能时，时钟源可以是下列时钟源中的一个：PCLK (default), LSE, HSI 或 SYSCLK.。

除此之外，USART 的时钟源是 PCLK。

选择 LSE、HSI 作为时钟源，可以使得 USART 在 MCU 处于低功耗状态的时候还能接收数据。基于唤醒模式选择和接收到的数据，USART 会将 MCU 唤醒，并由软件或者 DMA 将收到的数据从 USART_RDR 寄存器中读走。

如果用其它的时钟源，必须先将其打开以保证 USART 通讯。

通讯速度范围（特别是最大通讯速度）是由所选的时钟源来决定的。

接收器根据用户设定的过采样率来执行数据恢复，从而将接收数据和噪声区分开来。这需要在最大通讯速度和噪声抗扰度及对波特率偏差的敏感度之间做取舍。

通过 USART_CR1 寄存器中的 OVER8 位来选择过采样率是波特率时钟的 8 倍或者是 16 倍。

根据应用：

- 选择 8 倍过采样以实现较高速度（高至 $f_{CK}/8$）。 这种情况下最大的接收器允许的波特率偏差要小一些（参见章节 USART 波特率对时钟偏差的容忍）
- 选择 16 倍过采样率（OVER8=0）可提高时钟偏差容忍度。 这种情况下，最高通讯速度就会被限制在 $f_{CK}/16$，其中，f_{CK} 就是时钟源的频率。

USART_CR3 中的 ONEBIT 位用来选择判断逻辑电平的方法。 有两种选择：

- 根据接收数据位中央的三个采样结果进行多数票决。 这种情况下，当发现三个参与票决的采样结果不等时，NF 位会被置 1。
- 根据接收数据位中央的单个采样结果来直接裁决

根据应用：

- 在噪声干扰环境中应该选择三个采样多数票决的方式，可以排除检测到噪声的数据， 因为那说明在数据采样的时候有干扰。
- 在不担心噪声的情况下，应选择单采样裁决（ONEBIT=1），可以提高接收器对时钟偏差的容忍度（参见章节：USART 波特率对时钟偏差的容忍度） 这种情况下 NF 位永远都不会置 1。

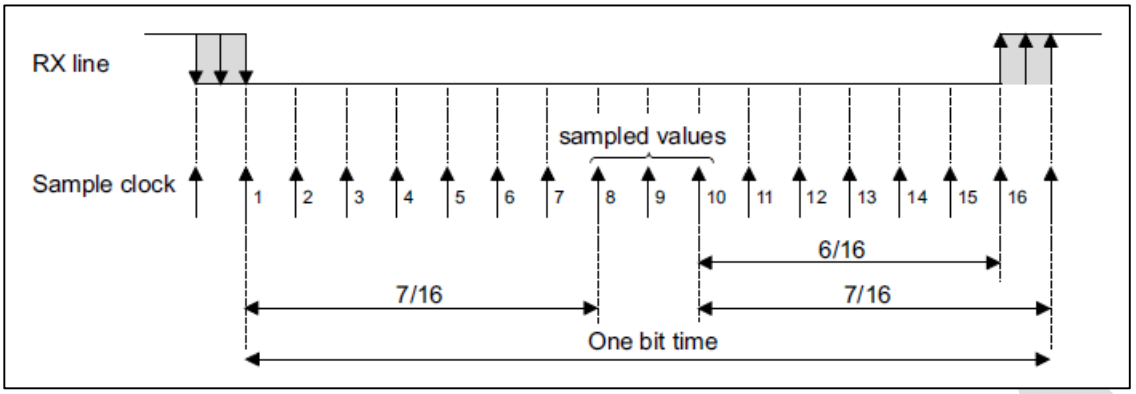
当一帧数据中检测到噪声时：

- 在 RXNE 位的上升沿，NF 位会被置 1.
- 有问题的数据仍会从移位寄存器转移到 USART_RDR 寄存器。
- 单字节通讯的时候不会产生中断。 然而与这一位同时上升的 RXNE 是会产生中断的。 在多缓冲区通讯的情况下，只要 USART_CR3 中的 EIE 位是高就会产生中断。

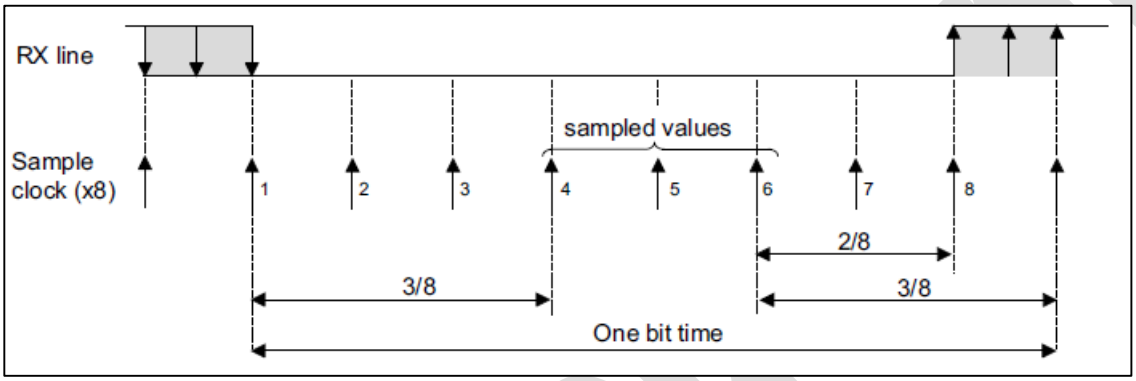
将 ICR 寄存器中的 NFCF 位置 1 就可以清除 NF 标志。

注：在智能卡、 IrDA 和 LIN 模式下，不能用 8 倍过采样的方式。在那些模式下，OVER8 位会由硬件强制为 0 。

过采样率为 16 的时候的数据采样



过采样率为 8 的时候的数据采样



从采样数据中检测噪声

采样值	NF 状态	接收到的位值
000	0	0
001	1	0
010	1	0
011	1	1
100	1	0
101	1	1
110	1	1
111	0	1

帧错误

当以下情况发生时检测到帧错误：由于没有同步上或大量噪音的原因，停止位没有在预期的时间上接收和识别出来。当帧错误被检测到时：

- ✧ FE 位被硬件置 1
- ✧ 有问题的数据仍会从移位寄存器转移到 USART_RDR 寄存器。
- ✧ 单字节通讯的时候不会产生中断。 然而与这一位同时上升的 RXNE 是会产生中断的。 在多缓冲区通讯的情况下，只要 USART_CR3 中的 EIE 位是高就会产生中断。

将 USART_ICR 寄存器中的 FECF 置 1 就可以清除 FE 标志。

接收期间的可配置的停止位

被接收的停止位的个数可以通过 USART_CR2 的控制位来配置，在正常模式时，可以是 1 或 2 个；在智能卡模式里可能是 0.5 或 1.5 个。

- ✧ 0.5 个停止位(智能卡接收模式)：0.5 停止位不做采样。因此，当选择 0.5 停止位时，不会检测到帧错误和中断帧。
- ✧ 1 个停止位： 对 1 个停止位的采样在第 8，第 9 和第 10 采样点上进行。

✧ 1.5 个停止位（智能卡模式）：当以智能卡模式发送时，器件必须检查数据是否被正确的发送出去。 所以接收器功能块必须被激活(USART_CR1 寄存器中的 RE =1)，并且在停止位的发送期间采样数据线上的信号。 如果出现校验错误，智能卡会在发送方采样 NACK 信号时，即总线上停止位对应的时间内时，拉低数据线，以此表示出现了帧错误。FE 在 1.5 个停止位结束时和 RXNE 一起被置 1。 对 1.5 个停止位的采样是在第 16，第 17 和第 18 采样点进行的(停止位开始后的 1 波特时钟周期)。1.5 个的停止位可以被分成 2 部分：一个是 0.5 个时钟周期，期间不做任何事情 。随后是 1 个时钟周期的停止位，在这段时间的中点处采样。 参见章节：USART 智能卡模式

✧ 2 个停止位： 对 2 个停止位的采样是在第一停止位的第 8，第 9 和第 10 个采样点完成的。如果第一个停止位期间检测到一个帧错误，帧错误标志将被置 1。 第二个停止位不再检查帧错误。 在第一个停止位结束时 RXNE 标志将被设置。

22.4.4 USART 波特率产生

接收器和发送器的波特率相同，在 USART_BRR 寄存器中设置。

公式 1： 标准 USART（包括 SPI 模式）的波特率（OVER8=0/1）

- 若过采样为 16，等式为

  Tx/Rx Baud = f_{CK} / (USARTDIV)

- 若过采样为 16，等式为

  Tx/Rx Baud = 2 x f_{CK} / (USARTDIV)

公式 2： 智能卡模式、LIN 模式和 IrDA 模式的波特率（OVER8=0）

智能卡模式、LIN 模式和 IrDA 模式，过采样仅支持 16，等式为

Tx/Rx Baud = f_{CK} / (USARTDIV)

USARTDIV 是一个无符号的定点数。值设置在 USART_BRR 寄存器。

✧ 当 OVER8=0 时，BRR = USARTDIV

✧ 当 OVER8=1 时，

- BRR[2:0] = USARTDIV 右移 1 位；
- BRR[3]必须为 0；
- BRR[15:4] = USARTDIV[15:4]。

注 1：在写入 USART_BRR 之后，波特率计数器会立即被波特率寄存器的新值替换。因此，不要在通信过程中改变波特率寄存器的数值。

注 2：当过采样为 16 或 8 时，USARTDIV 的值必须大于或等于 16。

如何从 USART_BRR 寄存器值得到 USARTDIV

例 1：获得 9600 波特率@f_{CK}=8MHz

当过采样为 16：

USARTDIV = 8000000/9600

BRR = USARTDIV = 833d = 0341h

当过采样为 8：

USARTDIV = 2*8000000/9600 = 1666.66（1667d=683h）

BRR[3:0] = USARTDIV[3:0]>>1= 3h>>1 = 1h

BRR = 681h

例 2：获得 921600 波特率@f_{CK}=48MHz

当过采样为 16：

USARTDIV = 48000000/921600

BRR = USARTDIV = 52d = 34h

当过采样为 8：

USARTDIV = 2*48000000/921600 = 104（104d=68h）

BRR[3:0] = USARTDIV[3:0]>>1= 8h>>1 = 4h

BRR = 64h

波特率时的误差计算（f_{CK}=48MHz，过采样率为 16/8）

	过采样率=16（OVER8=0）			过采样率=8（OVER8=1）		
目标波特率	实际波特率	BRR	%Error 注1	实际波特率	BRR	%Error 注1
2400	2400	0x4E20	0	2400	0x9C40	0
9600	9600	0x1388	0	9600	0x2710	0
19200	19200	0x9C4	0	19200	0x1384	0
38400	38400	0x4E2	0	38400	0x9C2	0
57600	57.62K	0x341	0.03	57.59K	0x681	0.02

115200	115.11K	0x1A1	0.08	115.25K	0x340	0.04
230400	230.76K	0xD0	0.16	230.21K	0x1A0	0.08
460800	461.54K	0x68	0.16	461.54K	0xD0	0.16
921600	923.07K	0x34	0.16	923.07K	0x64	0.16
2M	2M	0x18	0	2M	0x30	0
3M	3M	0x10	0	3M	0x20	0
4M	N.A	N.A	N.A	4M	0x14	0
5M	N.A	N.A	N.A	5052.63K	0x11	1.05
6M	N.A	N.A	N.A	6M	0x10	0

注 1：% Error = (实际波特率 - 目标波特率) / 目标波特率
注 2：CPU 的时钟频率越低，则某一特定波特率的精度也越低。 可以达到的波特率上限可以由这组数据得到。

22.4.5 USART 波特率对时钟偏差的容忍

只有当总的时钟偏差小于 USART 接收器的容忍度，USART 异步接收器才能正常地工作。 影响这些变化的因素有：

- ✧ DTRA：由于发送器误差而产生的变化 （ 包括发送器端振荡器的变化 ）
- ✧ DQUANT：接收器端波特率取整所产生的误差
- ✧ DREC：接收器端振荡器的变化
- ✧ DTCL：由于传输线路产生的变化 (通常是由于收发器在由低变高的转换时序与由高变低转换时序之间的不对称性所造成)。

DTRA + DQUANT + DREC + DTCL + DTU < USART 接收器的容忍度

其中，DWU 是使用停止模式唤醒时由于采样点偏差引起的误差。

当 M[1:0] = 01：DWU = $t_{WUUSART}$ / (11 * Tbit)

当 M[1:0] = 00：DWU = $t_{WUUSART}$ / (10 * Tbit)

当 M[1:0] = 10：DWU = $t_{WUUSART}$ / (9 * Tbit)

$t_{WUUSART}$ 是从检测到唤醒事件到时钟(由外设请求)和调节器（LDO）准备就绪之间的时间。在本产品中，$t_{WUUSART}$ 对应于数据表中提供的 t_{WUSTOP} 值。

根据以下选择，USART 接收器可以在下面表格中指定的最大耐受偏差范围内正确接收数据：

- ✧ 由 USART_CR1 寄存器的 M[1:0] 位定义 9 位、10 位 或 11 位字符长度。
- ✧ USART_CR1 寄存器的 OVER8 位定义，过采样率 8 位或 16 位。
- ✧ USART_BRR 寄存器的比特 BRR[3:0]等于或不等于 0000。
- ✧ USART_CR3 寄存器的 ONEBIT 位定义的，是 1 位采样还是 3 位采样。

串口接收容忍度（当 BRR[3:0]= 0000b）

M[1:0]	OVER8 = 0		OVER8 = 1	
	ONEBIT = 0	ONEBIT = 1	ONEBIT = 0	ONEBIT = 1
00	3.75%	4.375%	2.50%	3.75%
01	3.41%	3.97%	2.27%	3.41%
10	4.16%	4.86%	2.77%	4.16%

串口接收容忍度（当 BRR[3:0] != 0000b）

M[1:0]	OVER8 = 0		OVER8 = 1	
	ONEBIT = 0	ONEBIT = 1	ONEBIT = 0	ONEBIT = 1
00	3.33%	3.88%	2%	3%
01	3.03%	3.53%	1.82%	2.73%
10	3.7%	4.31%	2.22%	3.33%

注：表格中指定的数据在特殊情况下可能会略有不同，当接收到的帧包含一些恰好是 10 位持续时间的空闲帧时，M bits = 00 (当 M bits =01 时 11 位位长，当 M bits = 10 时 9 位位长)。

22.4.6 USART 自动波特率检测

USART 可以根据接收到的一个字符来检测和自动设置 USART_BRR 寄存器的值。

自动波特率检测在两种情况下很有用：

- ✧ 通讯速度预先不可知的情况下
- ✧ 系统使用低精度时钟源，需要在不测量时钟偏差的条件下纠正波特率的时候。

时钟源的频率必须和预期的波特率保持兼容（如果只支持模式 0 和 1，则必须选择过采样 16，波特率介于 f_{CK}/65535 和 f_{CK}/16 之间。否则，支持 4 种模式：过采样 16 时，波特率在 f_{CK}/65535 和 f_{CK}/16 之间;当过采样 8 时，波特率在 f_{CK}/65535 和 f_{CK}/8 之间）。

在打开自动波特率检测之前，必须先选择自动波特率检测方式。根据不同的字符模型对应不同的模式。

通过 USART_CR2 寄存器的 ABRMOD[1:0] 位进行选择。在这些自动波特率检测方式下，在同步数据接收过程中，多次测量波特率，并将每次测量的波特率与前一次进行比较。

具体是：

- Mode 0：以 1 开头的任何字符。在这种情况下，USART 测量起始位长度(下降沿到上升沿)。
- Mode 1：以 10xx 开头的任何字符。 这种情况下，USART 将测量起始位和第一个数据位的长度。 测量下降沿到下降沿的时长，在缓慢斜坡信号时可以确保更好的测量精度。
- Mode 2：一个 0x7F 字符帧（可以是 0x7F@LSB 在前，或 0xFE@MSB 在前）。这种情况下，波特率首先在起始位(BR.start)结束时更新，然后在第 6 位结束时更新(基于从下降沿到下降沿的测量:BR.6)。BIT0~BIT6，在 BR.start 处采样，而字符的其他位在 BR.6 处采样。
- Mode 3：一个 0x55 字符帧。在这种情况下，波特率首先在起始位(BR.start)结束时更新，然后在第 0 位结束时更新(基于从下降沿到下降沿所做的测量:BR.0)，最后在第 6 位结束时更新。BIT0 在 BR.start 处采样，BIT1~BIT6 在 BR.0 处采样，字符的其他位在 BR.6 处采样。
- 同时，对 RX 线上的每个中间的翻转（电平转换）进行检查，如果 RX 线上的转换与接收器不充分同步，将产生错误。接收器是基于 BIT0 计算自动波特率。

在打开波特率自动检测之前，USART_BRR 寄存器必须先初始化为一个不为零的波特率值。

将 USART_CR2 寄存器中的 ABREN 位置 1，打开自动波特率检测功能。 然后 USART 就在 RX 线上等待第一个字符过来。自动波特率操作结束后，USART_ISR 寄存器中的 ABRF 标志会被硬件置 1。

如果线路噪声严重，不能保证得到的波特率是准确的。这时 BRR 值被破坏且 ABRE 错误标志会被置 1。如果通讯速度不兼容自动波特率检测范围（位长度不在 16~ 65536 个时钟周期之间@过采样率=16，或位长度不在 8~ 65536 个时钟周期之间@过采样率=8）时也会发生这种情况。

RXNE 中断会在操作结束时产生。

后续的任何时候，都可以将 ABRF 标志清零，以重启自动波特率检测。

注：如果在自动波特率操作时，禁用 USART (UE=0)，则可能导致 BRR 值损坏。

22.4.7 USART 多机通信

多机通讯中，下面这些位需要清零：

- USART_CR2 寄存器中的 LINEN 位
- USART_CR3 寄存器中的 HDSEL、IREN 和 SCEN 位

可以将多个 USART 连接成一个网络来实现多机通讯。例如，一个 USART 设备可以是主机，它 TX 输出和其他 USART 的 RX 输入相连接；其他 USART 作为从设备，他们各自的 TX 输出‘逻辑与’在一起，并且和主设备的 RX 输入相连接。

在多机配置中，我们通常希望只有被寻址的接收者才被激活，来接收预期的数据，这样就可以减少未被寻址的接收器的参与带来多余的 USART 服务开销。

未被寻址的设备可启用其静默功能进入静默模式。通过 USART_CR1 寄存器的 MME 位置 1，启动静默模式功能。

在静默模式：

- 任何接收状态位都不会被置 1。
- 所有接收中断被禁止。
- USART_CR1 寄存器中的 RWU 位被置 1。 RWU 可以被硬件自动控制或在一定条件下由软件通过 USART_RQR 寄存器的 MMRQ 位写入。

根据 USART_CR1 寄存器中的 WAKE 位状态，USART 可以使用一下两种方式之一进入或退出静默模式：

- 如果 WAKE 位为 0：进行空闲总线检测（Idle line detection）。
- 如果 WAKE 位为 1：进行地址标记检测（Address mark detection）。

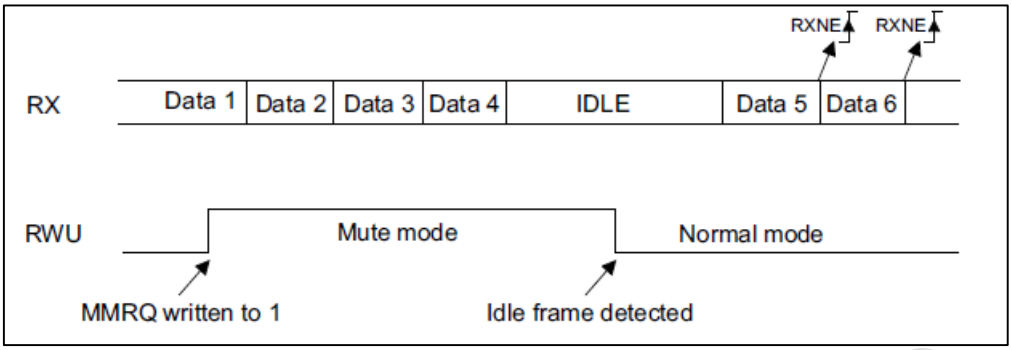
空闲总线检测（Idle line detection）（WAKE=0）

当 MMRQ 位被置 1，且 RWU 被自动置 1 时，USART 进入静默模式。

当检测到一个空闲帧的时候会唤醒。然后 RWU 位被硬件清零但 USART_ISR 寄存器中的 IDEL 位没有被置位。

下图给出了利用空闲总线检测来唤醒和进入静默模式的一个举例。

静默模式（使用空闲总线检测）



注 1：如果在空闲字符已经过去之后才置位 MMRQ，将不会退出静默模式（RWU 没被置 1）。

注 2：如果在线路空闲期间激活 USART，在一个空闲帧时长之后才会检测到空闲状态（不仅仅是在收到一个数据帧之后）。

4-bit/7-bit 地址标记检测（Address mark detection）（WAKE=1）

在这个模式里，如果 MSB 是 1，该字节被认为是地址，否则被认为是数据。 在一个地址字节中，目标接收器的地址被放在 4-bit 或 7-bit 位域中。设置 ADDM7 位来选择用 4 位地址还是用 7 位地址。接收器将这 4-bit 或 7-bit 地址同它自己地址做比较，接收器的地址在 USART_CR2 寄存器的 ADD 位设置。

注：在 7-bit 和 9-bit 数据模式下，地址检测分别按 6 位和 8 位地址（ADD[5:0]和 ADD[7:0]）操作。

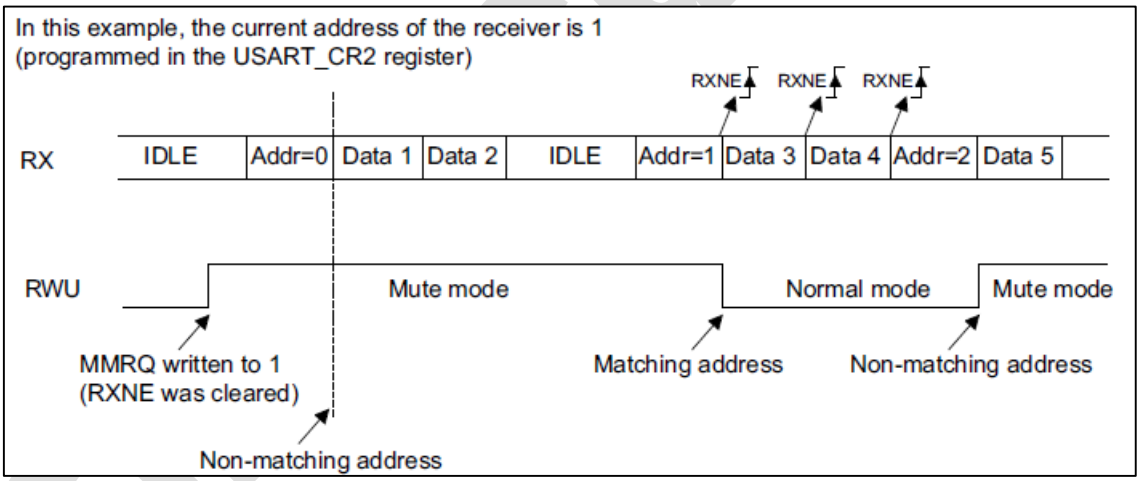
如果接收到的地址字节与它的编程地址不匹配时，USART 进入静默模式。 此时，硬件将 RWU 位置 1。 当 USART 进到静默模式，此地址字节不会置位 RXNE 标志，也不会产生中断或发出 DMA 请求。

将 MMRQ 写 1 也会令 USART 进入静默模式。 这时 RWU 位也被自动置 1。

如果接收到的字节与它的编程地址匹配，USART 将退出静默模式。 然后 RWU 位被清零，后续的字节会被正常接收。自 RWU 位被清零之后，RXNE 位会因为地址字节的接收而被置 1。

下图给出了利用地址标记检测来退出静默模式的一个举例。

静默模式（使用地址标记检测）



22.4.8 MODBUS 通信

USART 提供对 Modbus/RTU 和 Modbus/ASCII 协议实现的基本支持。 Modbus/RTU 是一个半双工、块传输协议。协议的控制部分（地址识别 address recognition，块完整性控制 block integrity control 和命令解释 command interpretation）必须在软件中实现。

USART 提供对块尾（end of a block）检测的基本支持，无需软件开销和其他资源。

Modbus/RTU

这个模式下，块尾一般公认为一个超过 2 个字符长度的静默阶段（idle line）。 此功能通过一个可设置的超时长度功能来实现。

超时功能和相应的中断必须开启，通过 USART_CR2 寄存器中的 RTOEN 位和 USART_CR1 寄存器中的 RTOIE 位。 RTO 寄存器中要写入一个与 2 个字符超时长度相当的值(例如，长度为 22 个位长)。当接收线保持空闲阶段达到这个长度时，在最后一个停止位被收到之后，会产生一个中断，表示当前的块接收已经完毕。

Modbus/ASCII

在这个模式，块尾被公认为一个特定字符串（CR/LF）。USART 用字符匹配功能实现这个机制。

将 LF 的 ASCII 码写到 ADD[7:0] 区域，然后打开字符匹配中断（CMIE=1），那么软件就会在收到 LF 字符后得到

提示，并能够在 DMA 缓冲区检查 CR/LF 字符。

22.4.9 USART 检验控制

设置 USART_CR1 寄存器上的 PCE 位，可以使能校验控制（发送时生成一个校验位，接收时进行校验检查）。 根据 M 位定义的帧长度，USART 帧格式罗列在下表。

帧格式

M 位	PCE 位	USART 帧[注 1]
00	0	Start bit+8-bit data+stop bit
00	1	Start bit+7-bit data+parity bit+stop bit
01	0	Start bit+9-bit data +stop bit
01	1	Start bit+8-bit data+parity bit+stop bit
10	0	Start bit+7-bit data +stop bit
10	1	Start bit+6-bit data+parity bit+stop bit

注 1：在数据寄存器中，parity bit 总是占据 MSB 位置(第 9 位、第 8 位或第 7 位，取决于 M 位值)

偶校验

计算校验位，使得一帧中的 LSB 数据位加上校验位中‘1’的总个数为偶数。LSB 数据位可以是 6、7 或 8 位，由 M 位控制。

例如： 数据=00110101，有 4 个‘1’，如果选择偶校验（在 USART_CR1 中的 PS ＝ 0），校验位将是‘0’。

奇校验

计算校验位，使得一帧中的 LSB 数据位加上校验位中‘1’的总个数为奇数。LSB 数据位可以是 6、7 或 8 位，由 M 位控制。

例如： 数据=00110101，有 4 个‘1’，如果选择奇校验（在 USART_CR1 中的 PS ＝ 1），校验位将是‘1’。

接收时的校验检查

如果校验检查失败，USART_ISR 寄存器中的 PE 标志会被置 1，如果 USART_CR1 寄存器中的 PEIE 为 1，将产生相应中断。软件对 USART_ICR 寄存器的 PECF 位写 1，可以清除 PE 标志。

发送时的校验生成

如果 USART_CR1 的 PCE 位被置 1，写进数据寄存器的数据的 MSB 位被校验位替换后发送出去（ 如果选择偶校验（PS=0）偶数个‘1’，如果选择奇校验（PS=1）奇数个‘1’）。

22.4.10 USART LIN（local interconnection network）模式

本节内容仅在 LIN 模式支持时有效。请参见 USART 具体功能配备。

LIN 模式是通过设置 USART_CR2 寄存器的 LINEN 位选择。 在 LIN 模式下，下列这些位必须保持为 0：

- ✧ USART_CR2 寄存器的 CLKEN 、STOP[1:0]位
- ✧ USART_CR3 寄存器的 SCEN，HDSEL 和 IREN

LIN 发送

章节：<USART 发送器>的操作过程同样适用于 LIN 的主机发送。 和常规的 USART 发送相同，但包含下列区别：

- ✧ 清零 M 位以配置 8 位字长
- ✧ 置位 LINEN 位进入 LIN 模式。 这时，置位 SBKRQ 发送 13 个‘0’位作为断开符。 然后发 2 位‘1’，以允许对下一个起始位的检测。

LIN 接收

当 LIN 模式被使能时，断开符检测电路被激活。 该检测完全独立于 USART 接收器，不管是在总线空闲时还是在发送某数据帧期间，断开符都可以被检测到。

一旦接收器被激活（USART_CR1 的 RE=1），电路就开始监测 RX 上的起始信号。 监测起始位的方法同搜索断开符号或数据相同。当起始位被检测到后，电路对每个接下来的位进行采样，和数据位一样（在每个位的第 8，9，10 个过采样点上进行采样）。如果 10 个（当 USART_CR2 的 LBDL = 0）或 11 个（当 USART_CR2 的 LBDL = 1）连续位都是‘0’，并且又跟着一个定界符（delimiter character），USART_SR 的 LBDF 标志就会被置 1。 如果 LBDIE 位为 1，还会产生中断。在确认断开符前，要检查定界符，因为它表示 RX 线已经回到高电平。

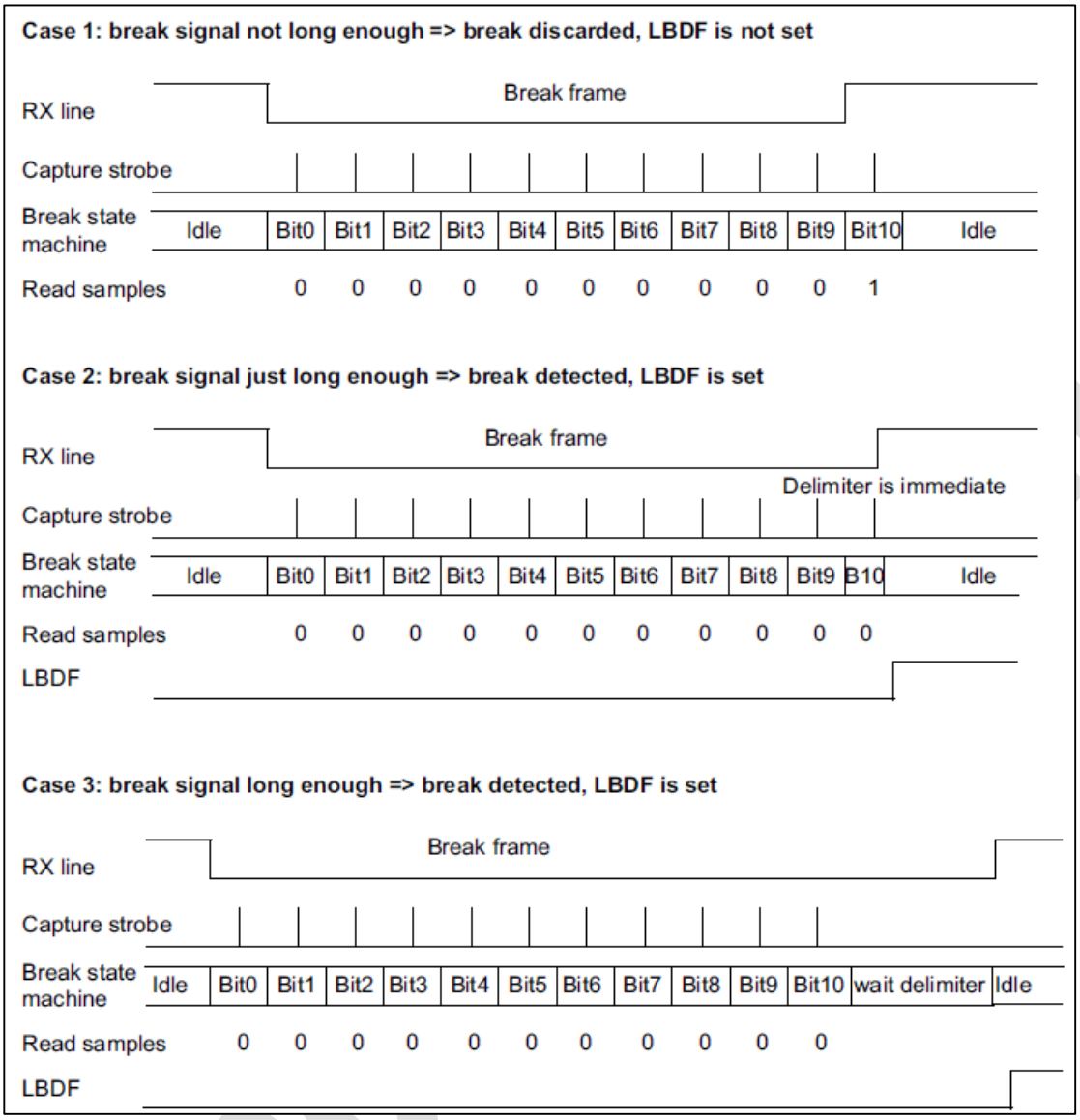
如果在第 10 或 11 个采样点之前采样到了‘1’，断开符检测电路取消当前检测，并重新寻找起始位。

如果 LIN 模式被禁止（LINEN=0），接收器将继续作为正常的 USART 工作，而不考虑检测断开符。

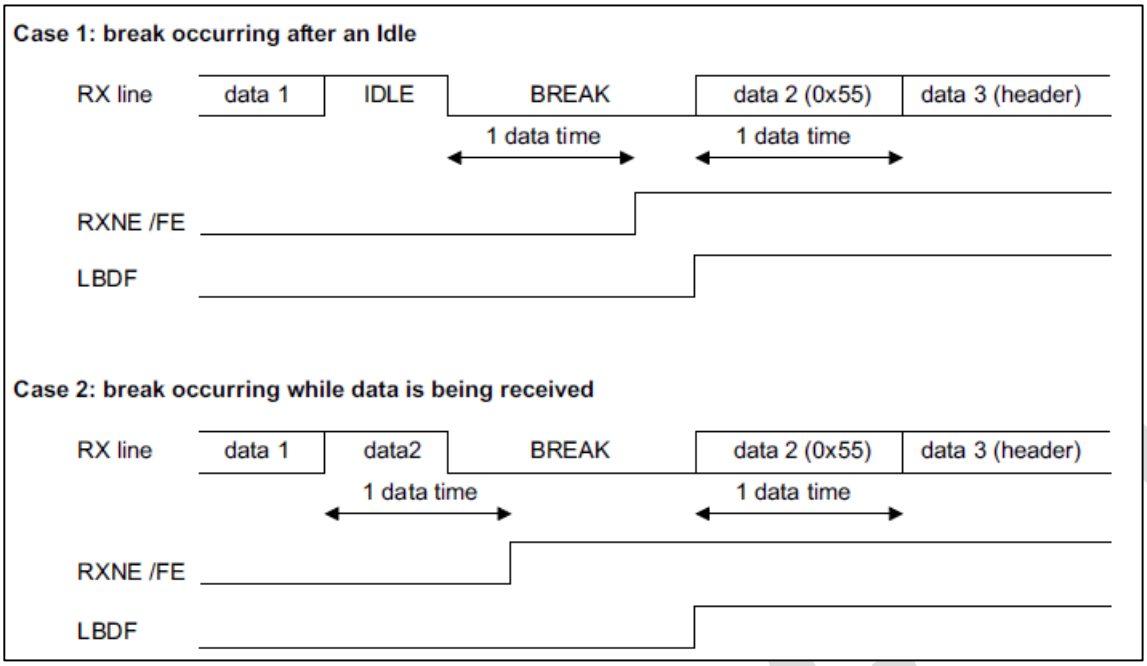
如果 LIN 模式被激活(LINEN=1)，只要一发生帧错误（例如：停止位检测到‘0’，这种情况出现在断开符接收时），接收器就停止，直到断开符检测电路接收到一个‘1’（断开符没有完成)或一个定界符（已经检测到一个完整的断开符）。

下图为断开符检测器状态机和断开标志的行为。

LIN 模式下的断开检测（11-bit 断开长度，LBDL=1）



LIN 模式下的断开检测 vs 帧错误检测



22.4.11 USART 同步模式

通过在 USART_CR2 寄存器的 CLKEN 位写 1 来选择同步模式。在同步模式下，下列位必须保持为 0：

- ✧ USART_CR2 寄存器中的 LINEN 位
- ✧ USART_CR3 寄存器中的 SCEN,HDSEL 和 IREN 位

此模式下，USART 可用于在主模式下控制双向同步串行通信。CK 脚是 USART 发送器时钟输出。在起始位和停止位期间，CK 脚上没有时钟脉冲。USART_CR2 寄存器中 LBCL 位的状态决定了在最后一个有效数据位（address mark）期间产生或不产生时钟脉冲。 USART_CR2 寄存器的 CPOL 位用于选择时钟极性，USART_CR2 寄存器上的 CPHA 位用于选择外部时钟的相位（见下图）。

在总线空闲期间、实际数据到来之前以及发送断开符期间，外部 CK 时钟不被激活。

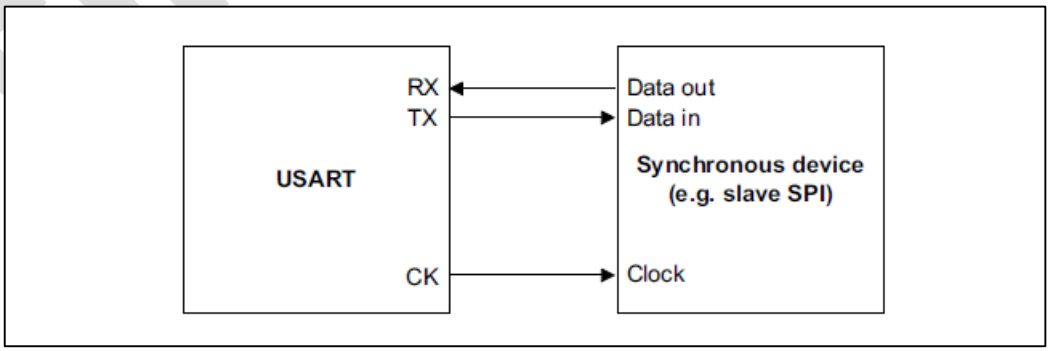
同步模式下，USART 发送器和异步模式下工作相同。但是因为 CK 与 TX 同步（根据 CPOL 和 CPHA），所以 TX 上的数据是随 CK 同步发出的。

同步模式下，USART 接收器工作方式与异步模式不同。 当 RE=1，数据在 CK 上采样（根据 CPOL 和 CPHA 决定在上升沿还是下降沿），不需要任何的过采样。 但必须考虑建立时间和持续时间（ 取决于波特率：1/16 位时间 ）。

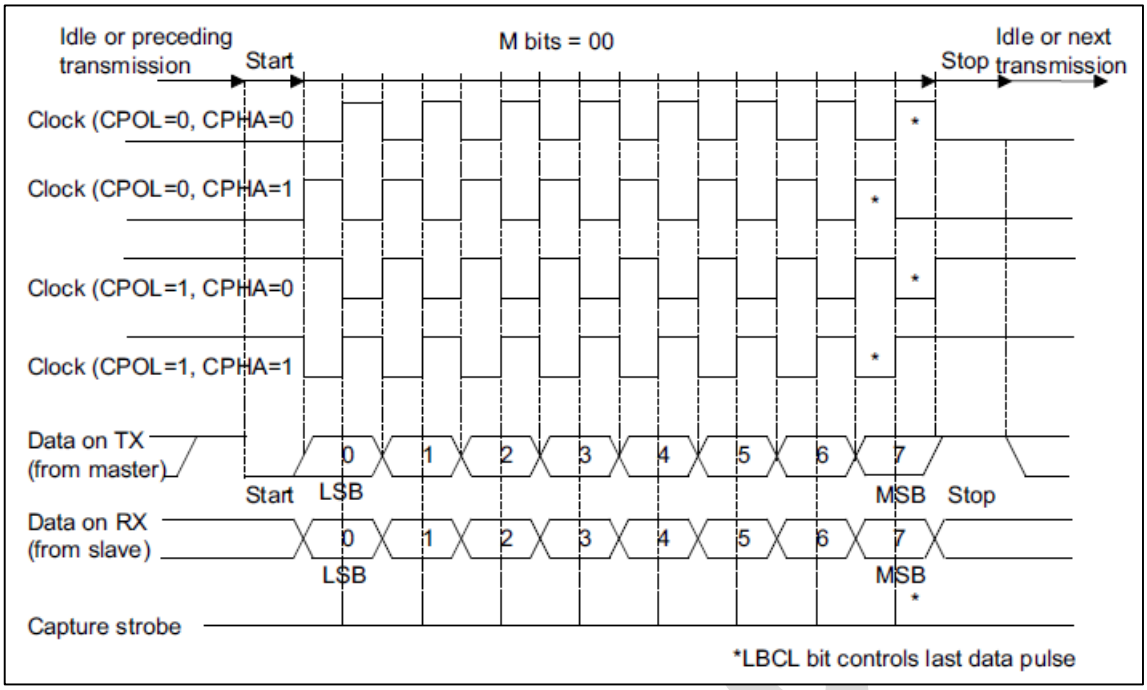
注 1：CK 脚同 TX 脚是协同工作的。因而，只有在使能了发送器 (TE=1)，并且发送数据时(写入数据至 USART_DR 寄存器)才提供时钟。这意味着在没有发送数据时是不可能接收一个同步数据的。

注 2：需要在发送器和接收器都被禁止时（UE=0）去修改 LBCL,CPOL 和 CPHA 位的配置，以保证时钟脉冲功能的正确性。

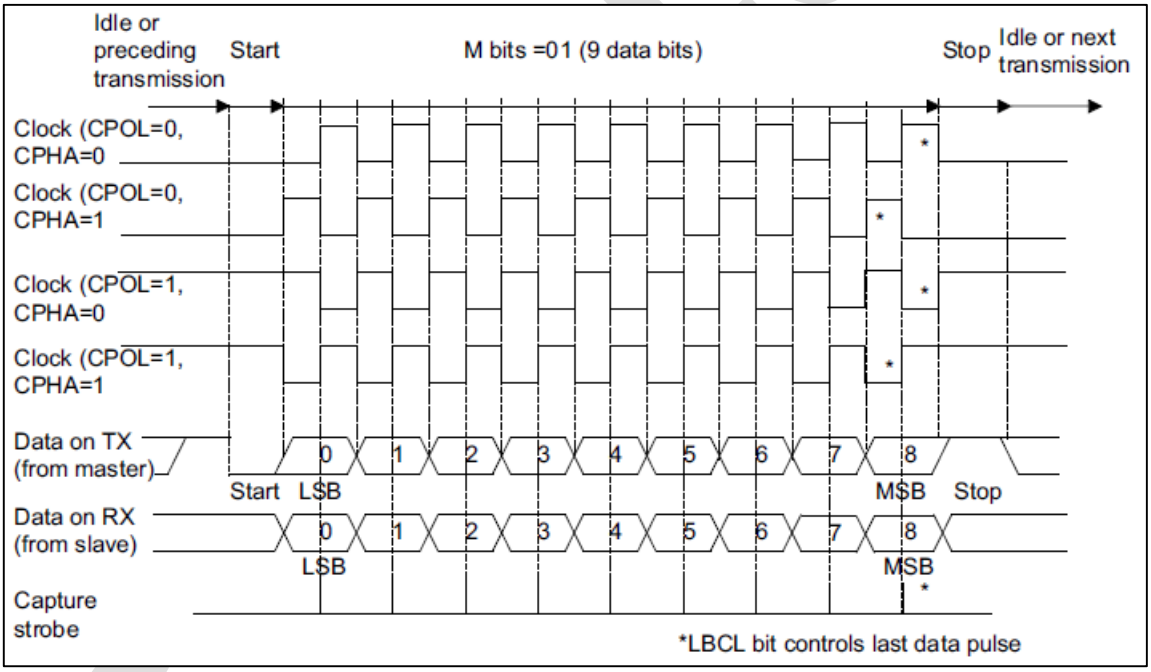
USART 同步传输举例



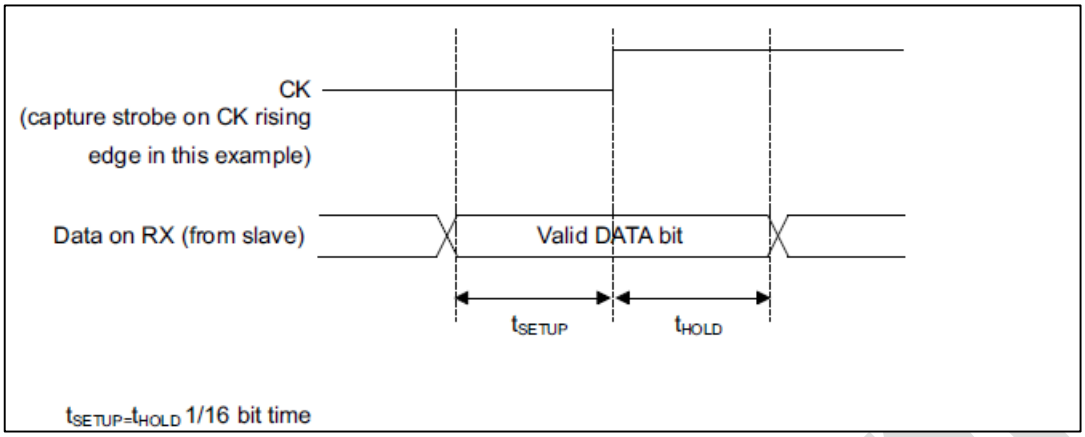
USART 数据时钟时序图（M 位=00b）



USART 数据时钟时序图（M 位=01b）



RX 数据建立/保持时间



注：CK 功能在智能卡模式下有所不同，详情参见章节：智能卡模式。

22.4.12 USART 单线半双工通信（single-wire Half-duplex）

单线半双工模式通过设置 USART_CR3 寄存器的 HDSEL 位选择。在此模式下，下列位必须保持为 0：

- ✧ USART_CR2 寄存器的 LINEN 和 CLKEN 位
- ✧ USART_CR3 寄存器的 SCEN 和 IREN 位

USART 可以配置成符合单线半双工协议。在单线半双工模式下，TX 和 RX 引脚在芯片在内部是连在一起。使用控制位(USART_CR3 中的 HDSEL 位) 选择半双工（Half-duplex）和全双工（Full- duplex）通信。

当 HDSEL 为‘1’时：

- ✧ TX 和 RX 引脚在芯片在内部连在一起
- ✧ RX 引脚不再被使用
- ✧ 当没有数据传输时，TX 总是被释放。因此，它在空闲状态或接收状态时表现为一个标准 I/O 口。这就意味该 I/O 必须配置为复用功能开漏且外部上拉。

除此以外，通信协议与正常 USART 模式类似。线上的任何冲突必须由软件来管理（ 例如，通过使用一个中央仲裁器)。 特别的是，当 TE 位被置位时，只要数据一写到数据寄存器上，发送从不会被硬件所阻碍。

22.4.13 USART 智能卡模式

本节内容仅在智能卡模式支持时有效。请参见 USART 具体功能配备。

设置 USART_CR3 寄存器的 SCEN 位，选择智能卡模式。在智能卡模式下，下列位必须保持为 0：

- ✧ USART_CR2 寄存器中的 LINEN 位
- ✧ USART_CR3 寄存器中的 HDSEL 和 IREN 位

此外，CLKEN 位应该被置 1，以便提供时钟给智能卡。

该接口符合 ISO7816-3 标准，支持智能卡异步协议。T=0 （ 字符模式） 和 T=1 （块模式） 都支持。

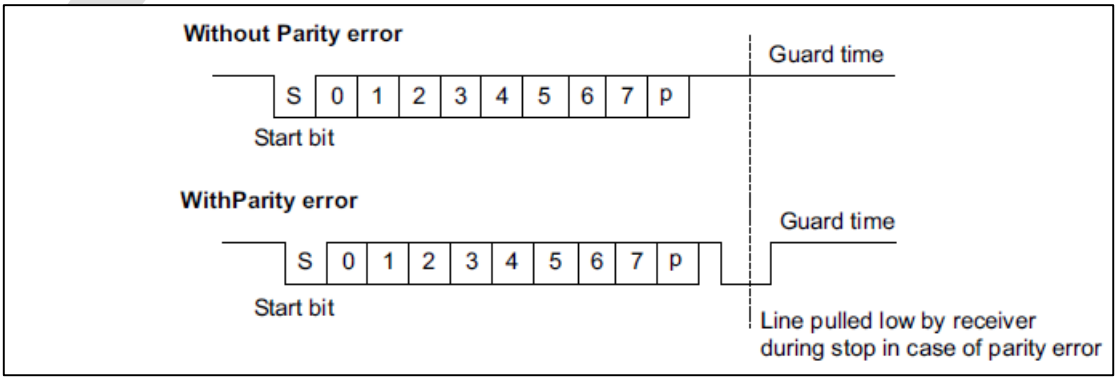
USART 应该被设置为：

- ✧ 8 位数据位加校验位： 此时 USART_CR1 寄存器中 M=01、PCE=1
- ✧ 1.5 个停止位： 即 USART_CR2 寄存器的 STOP=11；也可以选择 0.5 个停止位接收

在 T=0（字符）模式中，保护时间（guard time）内，校验错误在字符发送完毕后被提出。

下图所示为在数据线上有校验错误和没有校验错误时的情形。

ISO7816-3 异步协议



当连接到智能卡时，USART 的 TX 输出脚和智能卡通过同一根双向数据线进行通讯。 所以 TX 引脚必须配置成开漏状态。

智能卡模式实现了单线半双工通讯协议。

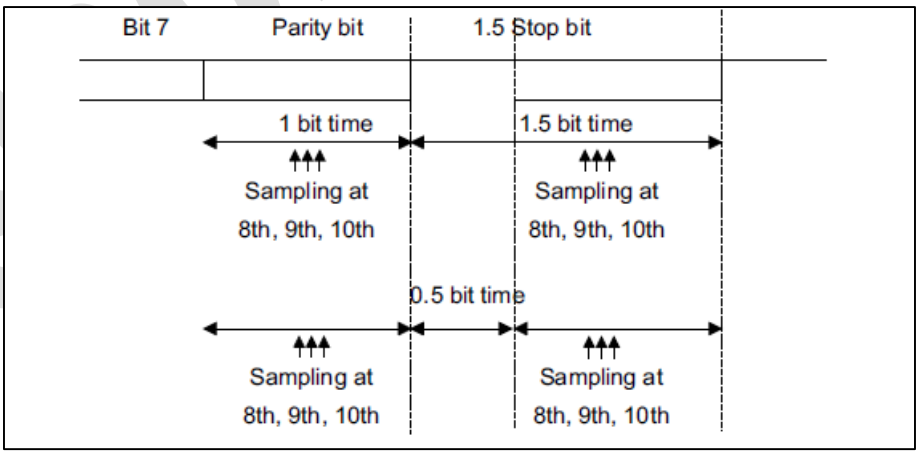
- 发送数据延后最少 1/2 个波特率时钟周期后从发送移位寄存器中移出。正常操作模式下，一个完整发送移位寄存器的内容在下一个波特时钟沿开始移出。在智能卡模式下，则会加入一个 确定的 1/2 波特时钟的延迟。
- 发送时，如果智能卡检测到一个校验错误，它会用拉低数据线的方式（NACK）将这个情况提示给 USART。这个 NACK 信号（拉低传输线 1 个波特时钟）将在发送方（配置为 1.5 个停止位）引起一个帧错误。USART 能够捕获这个情况并依照协议自动进行重发。重试的次数可以在 SCARCNT 区域进行设置。如果 USART 连续收到 NACK 信号，并重试达到设定的次数，USART 会停止发送并提示帧错误。
- 智能卡发送自动重试：USART 在收到 NACK 信号后，会等待 2.5 个波特周期，然后再给出重发字符的起始位。最后一个重发字符被顺利送达后，TC 位就立即置 1 了，这里没有保护时间（guard time）。 如果软件想要再重发，这里必须确保最少 2 个波特周期的延时，这是标准规定的。
- 如果使用 1.5 个停止位，并且在一帧数据的接收过程中检出校验错误，传输线会在一帧传输完毕后被拉低 1 个波特周期。这是在向智能卡表示，传给 USART 的数据不完全正确。如果 NACK 控制位为 1，校验错误会导致接收器（USART）发一个 NACK 出来，否则就 NACK 不发（在 T=1 的模式下）。 如果接收字符不正确，RXNE 及接收 DMA 请求不会被激活。按照协议规定，智能卡必须再发一个相同的字符。如果达到 SCARCNT 设定的重试次数而收到的字符还是错的，USART 将停止发送 NACK，而是提示一个校验错误。
- 智能卡接收自动重试:如果 USART 向智能卡发了 NACK 但智能卡却没有重发字符回应,则 BUSY 标志保持置位。
- 在发送过程中，USART 会在两次成功的发送字符之间插入一个保护时间 guardtime（具体长度在保护时间寄存器中设定）。 保护时间长度是从上一个字符的停止位开始计算的，GT[7:0] 位必须按照所须的 CGT（Character Guard Time，7816-3 标准要求的保护时间长度）进行编程，最少是 12（相当于 1 个字符的时间）。
- 可以调整保护时间寄存器来延迟 TC 标志的置位。正常操作中，TC 标志是在发送移位寄存器为空，且也没有进一步的发送请求后被置位。而在智能卡模式下，发送移位寄存器为空的事件只会触发保护时间计数器，令其计数直到所设定的保护时间。在这个过程中，TC 会强制为 0。当保护时间计数器达到所设定的值，TC 才会被置 1。
- 对 TC 标志的撤销不受智能卡模式的影响。
- 如果在发送过程的最后检出帧错误（就是从接收方传来一个 NACK），这个 NACK 是不会被发送方的接收模块当作起始位来检测的。根据 ISO 协议，接收到的 NACK 的持续时间可以是 1 或 2 个波特时钟周期。
- 在接收端，如果检测到一个校验错误，并且已经发送 NACK，接收器也不会把 NACK 当作起始位。

注 1：断开字符在智能卡模式中没有意义。一个带帧错误的 0x00 数据将被当成数据而不是断开符号。

注 2：当改变 TE 位时，不会有空闲帧发出。空闲帧（在其它模式中有定义）在 ISO 协议中是没有的。

下图显示 USART 如何采样 NACK 信号，本例中，USART 配置成 1.5 个停止位来发送数据。USART 的接收部分使能，用来检查数据完整性以及 NACK 信号。

校验错误检测（使用 1.5 位停止位）



USART 可以通过 CK 脚向智能卡提供时钟。 智能卡模式中，CK 和通讯无关，只是通过一个 5 位的预分频器从内部外设时钟源派生得到。这个分频系数在 USART_GTPR 寄存器中设置。CK 频率可以设置在 $f_{CK}/2$ 到 $f_{CK}/62$ 之间 ，其中 f_{CK} 指外设输入时钟。

块模式 （T=1）

在块模式（T=1）中，校验错误是要通过将 USART_CR3 寄存器的 NACK 位清零来关闭。

当请求从智能卡读取数据时，在块模式下，软件必须置位 USART_CR2 寄存器的 的 RTOEN 位启动接收超时功能，并将

RTOR 寄存器 RTO 位设为（BWT（block wait time）-11）的值。 如果达到超时时间，还没有收到智能卡的回应，将置位 RTOF 标志并会引起超时中断（若 USART_CR1 寄存器的 RTOIE 位置 1）。如果超时到期之前收到第一个字节，则会引起 RXNE 中断。

注：块模式下，即便是用 USART 的 DMA 模式从智能卡读取数据，也必须使能 RXEN 中断。同时，也只能在第一个字节接收之后使能 DMA。

在收到第一个字符之后（RXNE 中断），必须将 RTOR 寄存器的 RTO 位设置成（CWT（character wait time）-11）的值，这是为了允许在两个连续的字符之间自动检测最大等待时间。这个时间以波特时间作为单位。 如果智能卡在前一个字符发送结束后到设定的 CWT 周期之间没有发送新的字符，USART 会通过 RTOF 标志提示给软件，如果 RTOIE 位为 1，则会引起中断。

注 1：RTO 计数器开始计数：

- *在 STOP=00 情况下 ，从停止位结束开始计数；*
- *在 STOP=10 情况下，从第二个停止位结束开始计数；*
- *在 STOP=11 情况下，从停止位开始后 1 位持续事件开始计数；*
- *在 STOP=01 情况下，从停止位开始开始计数；*

注 2：按照智能卡协议的定义， BWT/CWT 值是从最后一个字符的起始位（start bit）开始计算的。所以 RTO 位必须为（BWT-11）或者（CWT-11），从而将各自的字符长度也算在内。

块长度计数器用于统计 USART 收到的全部字符的个数。 这个计数器在 USART 开始发送的时候（TXE=0）自动清零。块长度信息位于智能卡发出数据块的第三字节（序言部分 prologue field）。这个值必须写入到 USART_RTOR 寄存器的 BLEN 域。 当使用 DMA 模式，从块的开始前，这个寄存器必须设定为最小值（0x0）。 为了得到这个值，在收到第 4 个字节的时候会引起一个中断。 软件必须从接收缓冲区中读取到这个 LEN 域（第三字节）。

在中断驱动接收模式，块的长度可以由软件或编程 BLEN 位检查。 在块开始前，BLEN 可以设为最大值（0xFF）。 实际值则要在收到第三字节之后被写到寄存器中。

如果块格式使用 LRC 纵向冗余校验（1 结尾字节 epilogue byte），BLEN 就等于 LEN。如果块格式使用 CRC 循环冗余校验（2 个结尾字节），BLEN 就等于 LEN+1。 整个块长度（包括序言 prologue，校验 epilogue 和信息 information 区域）等于 BLEN+4。块结束将通过 EOBF 标志及相应中断（EOBIE 位被置 1 时）提示给软件。

在块长度出错的情况下，块结束则会引起 RTO 中断（CWT 字符等待时间溢出）。

注：错误校验码（ LRC/CRC ）的验算需要由软件来完成。

正向和反向约定

智能卡协议定义了两种约定： 正向约定和反向约定。

正约定定义如下： 低位在前，逻辑位为‘1’相当于传输线高电平(H)，采用偶校验。 要使用这个转换方式，下列控制位必须设置： MSBFIRST=0, DATAINV=0 （ 默认值 ）。

反向约定定义如下： 高位在前，逻辑位为‘1’相当于传输线低电平(L)，采用奇校验。 要使用这个转换方式，下列控制位必须设置： MSBFIRST=1, DATAINV=1。

注：当逻辑数据为反向状态时（0=H ，1=L），校验位也是采用同样的逻辑（0 是 H，1 是 L）。

为了确认智能卡的约定方式，智能卡将发送初始化字符 TS ，作为 ATR（应答复位 Answer To Reset）帧的第一个字符。TS 的两种可能的模式为：LHHL LLL LLH 和 LHHL HHH LLH。

- ✧ (H) LHHL LLL LLH 将启用反向约定：状态 L 对应数据 1，高位在前（MSB first）。 用反向约定解码时，此传达的字节等于 ’3F’。
- ✧ (H) LHHL HHH LLH 将启用正向约定：状态 H 对应数据 1，低位在前（LSB first）。用正向解码时，此传达的字节等于 ’3B’。

当在 bit 2 ~ bit10 之间的 9 个时刻中有偶数个 1 时，字符校验正确。

由于 USART 并不知道智能卡用哪种约定模式，所以需要能够识别任一约定并且照着执行。模式识别不是由硬件完成，而是通过软件实现。此外，假设 USART 被配置为正向约定（默认），而智能卡的应答是按反向约定，TS= LHHL LLL LLH => USART 收到的字符会是‘03’并且校验规则是奇校验。

因此，有两个方法可以用来确认 TS 的约定模式：

方法 1：USART 被设置为标准智能卡模式/正向约定。这时，TS 样式确认产生出一个校验错误中断并发送一个错误信号给智能卡。

- ✧ 校验错误中断提示软件，智能卡在正向约定模式下没有给出正确的应答。软件来重新设置 USART 为反向约定。
- ✧ 收到错误信号后，卡会再次发送一个同样 TS 字符，由于 USART 已经被重新设置过，这次就会被正确接收。

或者，在校验错误中断处理中，软件可以对 USART 重新设置，向智能卡发出新的复位命令，然后重新等待 TS。

方法 2：USART 被设置为 9 位字长/无校验的模式，无 BIT 反向。 在这个情况下能够收到的两种 TS 样式如下：

(H) LHHL LLL LLH = 0x103 -> 选为反向约定

(H) LHHL HHH LLH = 0x13B -> 选为正向约定

软件根据两种约定模式检查接收到的字符，若其中任何一个匹配，将 USART 设置成正确的模式来适应下一个字符的接收。

如果两个都不识别，那就向智能卡发送一个复位命令以重启新一轮会话。

22.4.14 USART IrDA SIR ENDEC 模块

本节内容仅在 IrDA 模式支持时有效。请参见 USART 具体功能配备。

通过设置 USART_CR3 寄存器的 IREN 位选择 IrDA 模式。在 IrDA 模式下，下列位必须保持为 0：

- ✧ USART_CR2 寄存器的 LINEN, STOP 和 CLKEN 位，
- ✧ USART_CR3 寄存器的 SCEN 和 HDSEL 位。

IrDA SIR 物理层规定使用反向归零（RZI）调制方案，该方案用逻辑‘0’表示一个红外光脉冲。（见下图）。

SIR 发送编码器对 USART 发出的非归零（NRZ）比特流进行调制。输出脉冲流用来驱动一个外部的输出驱动器和红外 LED。USART 的 SIR ENDEC 最高只支持到 115.2Kbps 速率。在正常模式下， 脉冲宽度规定为一个位周期的 3/16。

SIR 接收解码器对来自红外接收器的归零位比特流进行解调，并将接收到的 NRZ 串行比特流输出到 USART。 在空闲状态，解码器输入通常是高（标记状态 marking state）。发送编码器输出的极性和解码器的输入极性相反。当解码器输入低时，检测到一个起始位。

- ✧ IrDA 是一个半双工通信协议。如果发送器忙（当 USART 正在送数据给 IrDA 编码器）， IrDA 接收线上的任何数据将被 IrDA 解码器忽视。如果接收器忙（当 USART 正在接收从 IrDA 解码器来的解码数据），从 USART 到 IrDA 的 TX 上的数据将不会被 IrDA 编码。当接收数据时，应该避免发送，因为将被发送的数据可能被破坏。
- ✧ SIR 发送逻辑把‘0’作为高脉冲发送，把’1’作为低电平发送。 脉冲的宽度规定为正常模式时位周期（bit period）的 3/16（见下图）。
- ✧ SIR 解码器把 IrDA 兼容的接收信号转变成给 USART 的比特流。
- ✧ SIR 接收逻辑把高电平状态解释为‘1’，把低脉冲解释为‘0’。
- ✧ 发送编码器输出与解码器输入有着相反的极性。当空闲时，SIR 输出处于低电平状态。
- ✧ IrDA 规范要求脉冲宽度要大于 1.41us。脉冲宽度是可编程的。接收器端的尖峰脉冲检测逻辑滤除宽度小于 2 个 PSC 周期的脉冲（PSC 是寄存器 USART_GTPR 中编程的预分频值 ）。 宽度小于 1 个 PSC 周期的脉冲一定被滤除掉，但是那些宽度大于 1 而小于 2 个周期的脉冲可能被接收或滤除，那些宽度大于 2 个周期的将被视为一个有效的脉冲。当 PSC=0 时，IrDA 编码器/解码器不工作。
- ✧ 接收器可以与一个低功耗发送器通信。
- ✧ 在 IrDA 模式里，USART_CR2 寄存器上的 STOP 位必须配置成‘1 个停止位’。

IrDA 低功耗模式

发送器

在低功耗模式，脉冲宽度不再保持 3/16 个位周期。 而是低功耗波特率的 3 倍，波特率最小可以是 1.42MHz。

通常这个值是 1.8432MHz（1.42 MHz < PSC < 2.12 MHz）。一个低功耗模式可编程分频器把系统时钟进行分频以达到这个值。

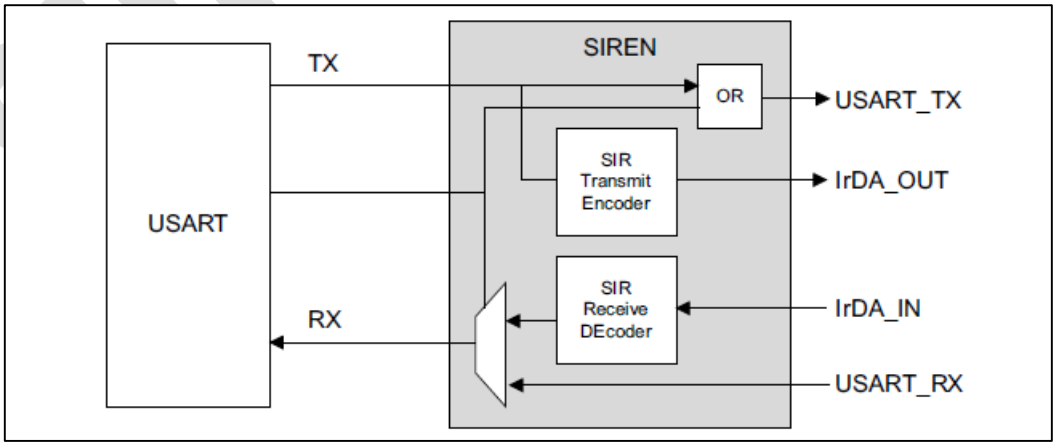
接收器

低功耗模式的接收类似于正常模式的接收。 为了滤除尖峰干扰脉冲，USART 应该滤除宽度短于 1 个 PSC 的脉冲。只有持续时间大于 2 个周期（IrDA 低功耗波特率时钟，USART_GTPR 中的 PSC）的低电平信号才被接受为有效信号。

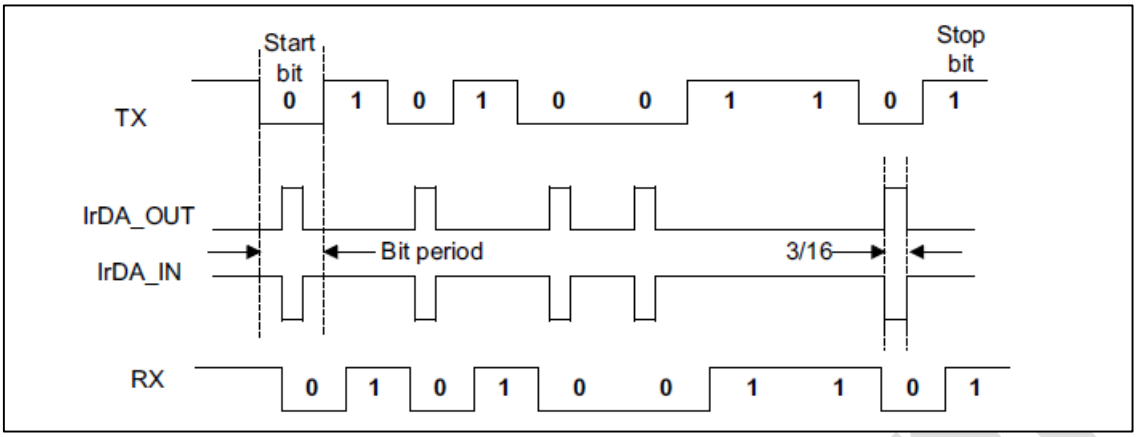
注 1：宽度小于 2 且大于 1 个 PSC 周期的脉冲可能会被滤除也可能不会。

注 2：接收器的建立时间应该由软件管理。 IrDA 物理层规范规定了在发送和接收之间最小要有 10ms 的延时（IrDA 是一个半双工协议 ）。

IrDA SIR ENDEC 框图



IrDA 数据调制(3/16)-正常模式



22.4.15 DMA 连续通信

USART 可以利用 DMA 连续通信。Rx 缓冲器和 Tx 缓冲器的 DMA 请求是分别产生的。

注：请参见章节：USART 具体功能配备，以确定是否支持 DMA 模式。如果所用产品无 DMA 功能，应按章节：USART 发送器或章节：USART 接收器 里所描述的方法使用 USART 。在 USART_ISR 寄存器里，可以清零 TXE/RXNE 标志来实现连续通信。

利用 DMA 发送

使用 DMA 进行发送，可以通过设置 USART_CR3 寄存器上的 DMAT 位激活。只要设置了 TXE 位，数据就会从使用 DMA 外设配置的 SRAM 区域（参见章节:直接存储器访问 (DMA) 控制器）加载到 USART_TDR 寄存器。

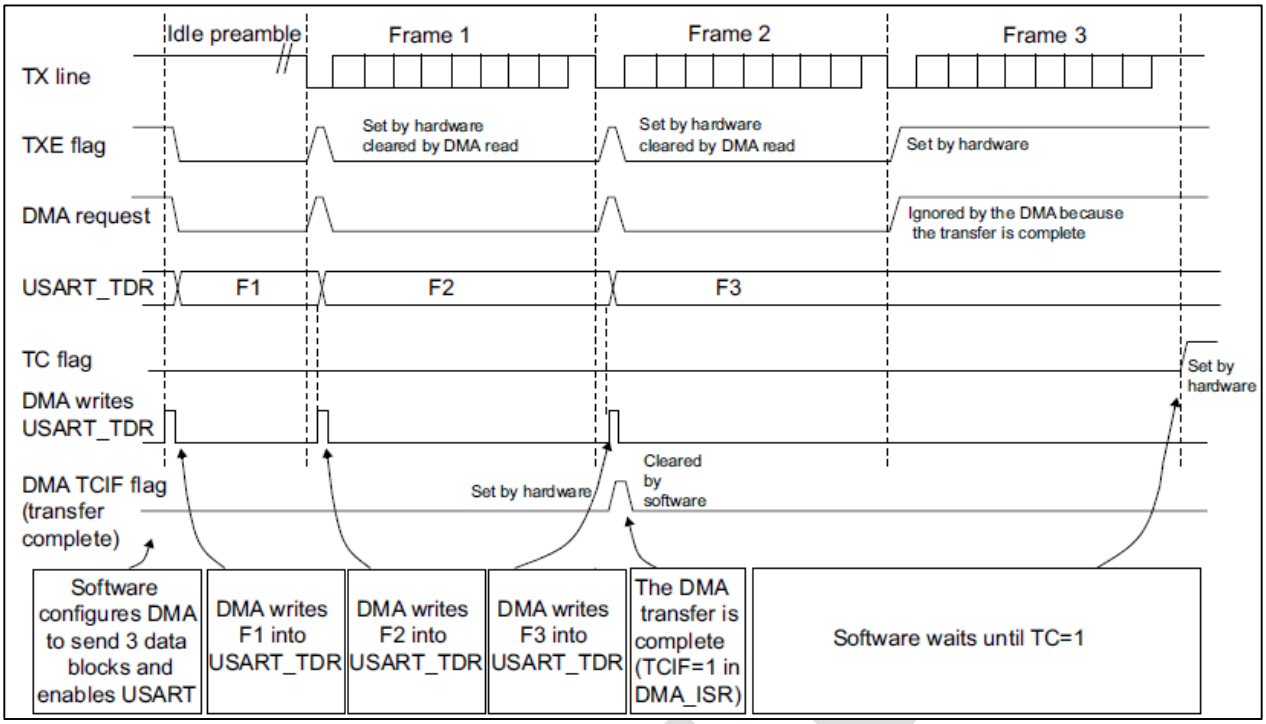
设置一个 DMA 通道给 USART 发送，要使用下列步骤（x 指通道号）：

1、 在 DMA 控制寄存器上将 USART_TDR 寄存器地址配置成 DMA 传输的目的地址。在每个 TXE 事件后，数据将被传送到这个地址。
2、 在 DMA 控制寄存器上将内存地址配置成 DMA 传输的源地址。在每个 TXE 事件后，将从此存储器区读出数据并传送到 USART_TDR 寄存器。
3、 在 DMA 控制寄存器中配置要传输的总的字节数。
4、 在 DMA 寄存器上配置通道优先级。
5、 根据应用程序需求，配置在传输完成一半或全部完成时产生 DMA 中断。
6、 将 USART_ICR 寄存器的 TCCF 位置 1 以清除 USART_ISR 寄存器的 TC 标志。
7、 在 DMA 寄存器上激活该通道。

当 DMA 控制器指定的数据量传输完成时，DMA 控制器在该 DMA 通道的中断向量上产生一中断。

在发送模式下，当 DMA 传输完所有要发送的数据时（DMA_ISR 寄存器的 TCIF 标志置 1）；监视 USART_SR 寄存器的 TC 标志可以确认 USART 通信是否结束。 这是必需的，可以避免在关闭 USART 或进入停机模式之前破坏最后一次传输的数据。软件必须等待直到 TC =1。TC 标志在全部数据发送期间会是 0，并且在最后一帧数据发送结束后由硬件置 1。

利用 DMA 发送



利用 DMA 接收

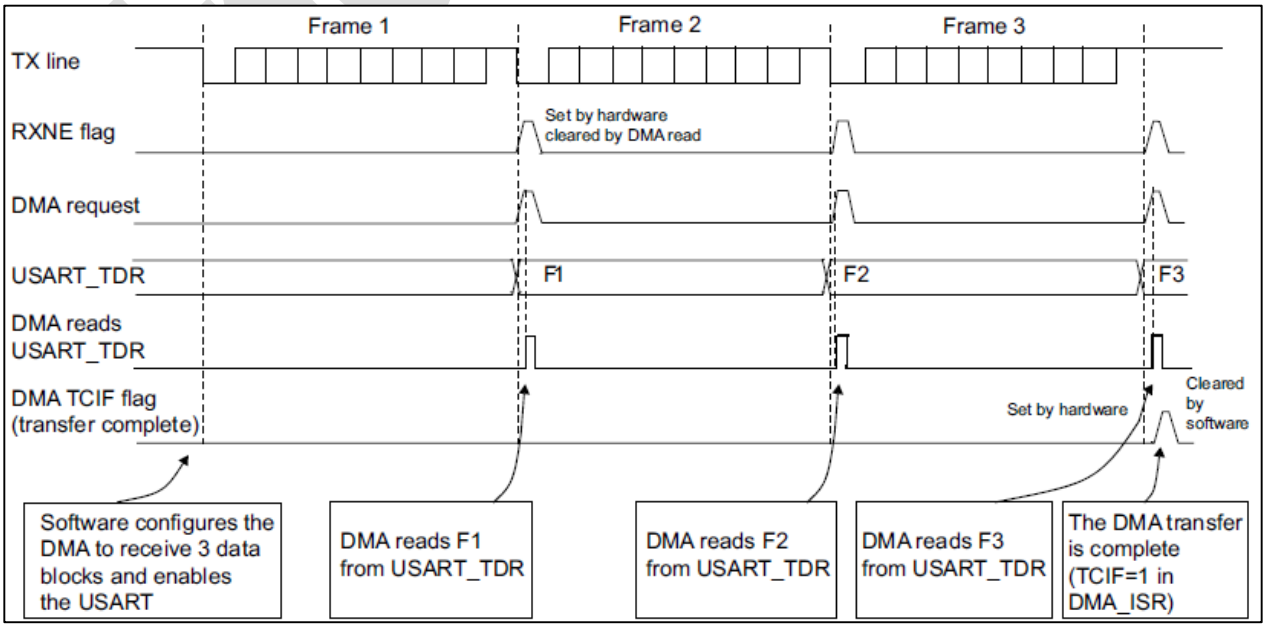
使用 DMA 进行接收,可以通过设置 USART_CR3 寄存器上的 DMAR 位激活。当收到一个字节数据时,从 USART_RDR 寄存器取出来的数据会被转移到 DMA 外设中指向的 SRAM 区域（参见章节：直接存储器访问(DMA)控制器）。

将一个 DMA 通道设置给 USART 接收，要按照下列步骤：

1、 在 DMA 控制寄存器上将 USART_RDR 配置成 DMA 传输的源地址。在每个 RXNE 事件后，数据将从这个地址取走。
2、 在 DMA 控制寄存器上将内存地址配置成 DMA 传输的目标地址。 在每个 RXNE 事件后， 数据将从 USART_RDR 取出发往这个目标地址。
3、 在 DMA 控制寄存器中配置要传输的总的字节数。
4、 在 DMA 寄存器上配置通道优先级。
5、 根据应用程序需求，配置在传输完成一半或全部完成时产生 DMA 中断。
6、 在 DMA 寄存器上激活该通道。

当 DMA 控制器指定的数据量传输完成时，DMA 控制器在该 DMA 通道的中断向量上产生一中断。

利用 DMA 接收



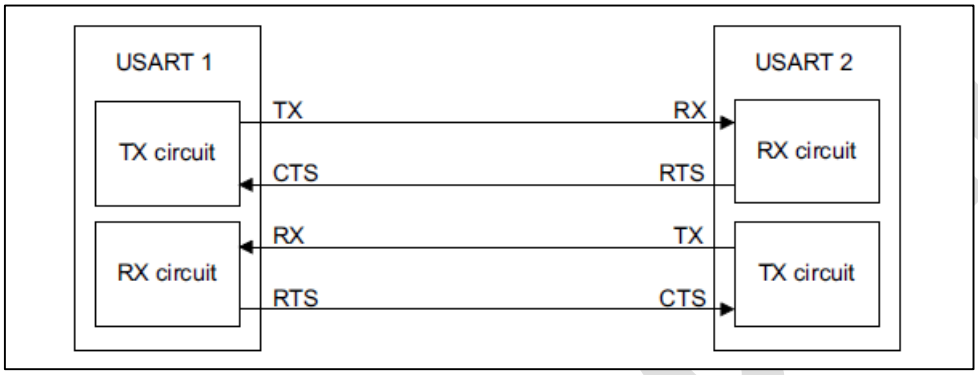
多缓冲器通信中的错误标志和中断产生

在多缓冲器通信的情况下，通信期间如果发生任何错误，会在当前字节传输后将错误标志置 1。如果中断使能位被置 1，将产生中断。 在单个字节接收的情况下，帧错误（framing error）、溢出错误（overrun）和噪音标志(noise flag)和 RXNE 一起被置位，有一个独立的错误标志中断使能位（USART_CR3 寄存器的 EIE 位）；如果此 EIE 被置位，当任一错误发生时，在当前字节传输结束后，产生中断。

22.4.16 RS232 硬件流控制和 RS485 驱动

利用 CTS 输入和 RTS 输出可以控制 2 个设备间的串行数据流。下图显示了这种模式下如何连接两个设备：

2 个 USART 之间的硬件流控制



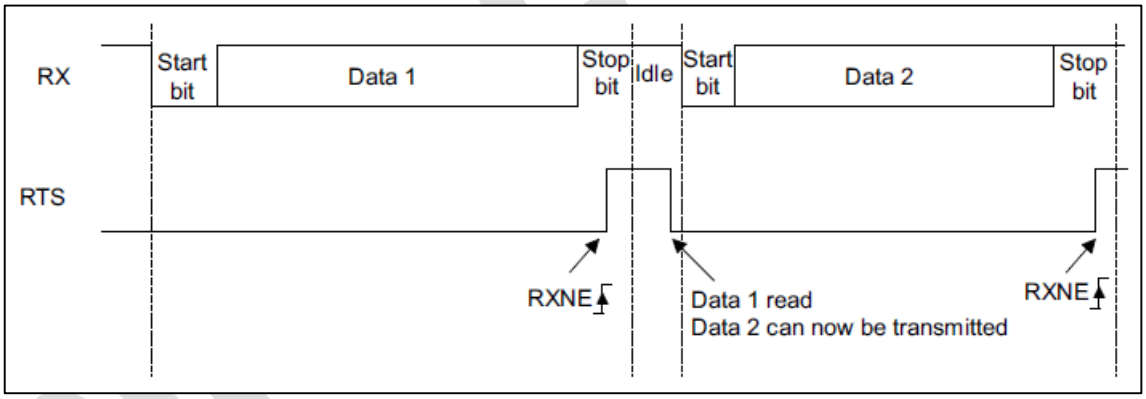
通过将 UASRT_CR3 中的 RTSE 和 CTSE 置 1，可以分别独立地使能 RTS 和 CTS 流控制。

RS232 RTS 流控制

如果 RTS 流控制被使能 (RTSE=1)，只要 USART 接收器准备好接收新的数据，RTS 就变成有效 (拉低电平)。 当接收寄存器内的数据为满时，RTS 被释放（高电平），表明希望在当前帧结束时停止数据传输。

下图是一个启用 RTS 流控制的通信的例子。

RS232 RTS 流控制



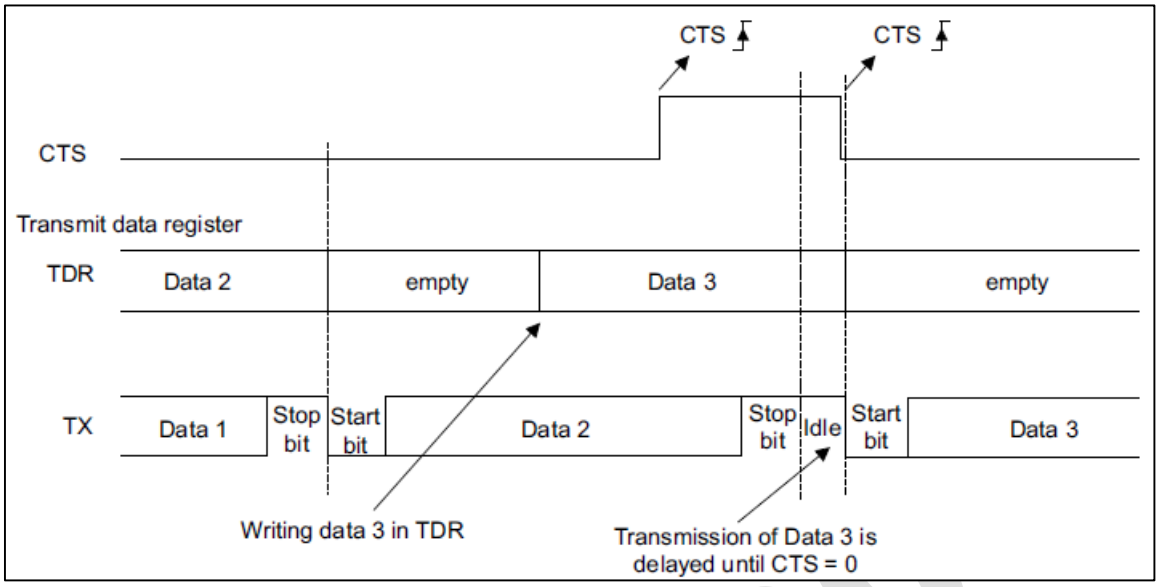
RS232 CTS 流控制

如果 CTS 流控制被使能 (CTSE=1)，发送器在发送下一帧前检查 CTS 输入。 如果 CTS 有效 (拉低电平)，则下一个数据被发送 (假设那个数据是准备发送的，也就是如果 TXE=0)，否则下一帧数据不被发出去。 若 CTS 在传输期间被变成无效，当前的传输完成后才停止发送。

当 CTSE=1 时，只要 CTS 输入翻转状态，硬件就自动设置 CTSIF 状态位。它表明接收器是否准备好进行通信。 如果设置了 USART_CT3 寄存器的 CTSIE 位，则产生中断。

下图是一个启用 CTS 流控制的通信的例子。

RS232 CTS 流控制



注： 为了保证正确操作，CTS 必须在当前字符的结束前至少 3 个 USART 时钟周期做动作（使有效）。另外，需要注意的是，对于小于 2 个 PCLK 周期的脉冲，CTSCF 标志可能不会置 1 。

RS485 驱动使能

驱动使能功能通过 USART_CR3 控制寄存器的 DEM 位置 1 来打开。 它允许用户通过 DE（驱动使能）信号来激活外部收发器的控制端。提前时间（assertion time）的意思是驱动使能信号和第一个字节的起始位之间的时间间隔。 这个时间可以在 USART_CR1 控制寄存器的 DEAT[4:0] 域中设置。滞后时间（de-assertion time）是一个发送消息的最后一个字节的停止位和释放 DE 信号之间的时间间隔。这个时间可以在 USART_CR1 控制寄存器的 DEDT[4:0] 域中设置。DE 信号的极性则可以通过 USART_CR3 控制寄存器中的 DEP 位进行选择。

在 USART 中，DEAT 和 DEDT 用采样时间单位表示(1/8 或 1/16 比特持续时间，取决于过采样率)。

22.4.17 STOP 模式唤醒

本节内容仅在 STOP 模式唤醒功能支持时有效。请参见 USART 具体功能配备。

当 USART 时钟被设置为 HSI 或 LSE 时，并置位 UESM 位就可以使用 USART 将 MCU 从 Stop 模式唤醒。 时钟设置参考章节：复位和时钟控制系统（RCC）。

✧ USART 时钟源为 HSI

如果在停止模式下，HSI 时钟被关闭，当 USART 接收线上的下降沿被检测到时，USART 请求打开 HSI 时钟。然后 HSI 时钟用于帧接收。

- 如果确认唤醒事件，则 MCU 从低功耗模式唤醒，数据接收正常。
- 如果唤醒事件未被验证，则 HSI 时钟再次关闭，MCU 不被唤醒，保持在低功耗模式，释放时钟请求。

✧ USART 时钟源为 LSE

除了 LSE 时钟在 stop 模式下时开启的，其他与时钟源为 HSI 相同。如果 USART 不请求 LSE 时钟，LSE 不会传递到 USART，LSE 不是被关闭，是通过门控以避免无用的消耗。

可以用标准的 RXNE 中断来唤醒 STOP 模式。这时，在进入 Stop 模式之前 RXNEIE 位必须先设置为 1。

另外需要在 WUS 位指定这个中断。

必须在进入 Stop 模式之前先要置位 USART_CR1 中的 UESM 位，否则唤不醒。

当检测到唤醒事件，WUF 表示会被硬件置 1，若 WUFIE 位置 1，将会产生中断。

注 1：在进入 Stop 模式之前，软件必须检查 USART 没有正在发数据，这可以通过 USART_SR 寄存器中的 BUSY 标志来判断。

注 2：在收到唤醒事件时，WUF 标志肯定会被置 1，这不受 MCU 所处的工作模式影响，无论是在 Stop 模式还是在活动模式，都可以。

注 3：当刚刚初始化好接收器的时候就进入 Stop 模式时，需要检查 REACK 位以确定 USART 确实已经被打开了。

注 4：当用 DMA 来作接收时，在进入 Stop 模式前必须先禁止掉，在唤醒之后在重新使能它都行。

Stop 模式下使用静默功能

如果在进入 Stop 模式之前，USART 已经处于静默状态下：

✧ 不可以使用基于空闲检测退出静默的方式，因为空闲检测在 Stop 模式下不工作。

✧ 如果使用地址匹配的方式退出静默状态，唤醒源也就只能是地址匹配事件。如果在进入 Stop 模式时设置了 RXNE

标志位，则 USART 接口在地址匹配时保持静音模式，并在 Stop 模式下唤醒。

✧ 如果 USART 被设置为起始位检测就唤醒 MCU，那么 WUF 标志会置 1，但 RXNE 标志就不会置位。

确定允许从停止模式正确唤醒的最大 USART 波特率（USART 时钟源是 HSI 时钟）

允许从停止模式正确唤醒的最大波特率取决于：

✧ 数据手册中的 $t_{WUUSART}$。

✧ USART 接收器容忍，参见 22.4.5 USART 波特率对时钟偏差的容忍。

举例：OVER8 = 0, M bits = 01, ONEBIT = 0, BRR[3:0] = 0000。

此条件下，根据 表格：<串口接收容忍度（当 BRR[3:0]= 0000b）>，USART 接收器容忍为 3.41%。

DTRA + DQUANT + DREC + DTCL + DWU < USART 接收器容忍

DWU max = $t_{WUUSART}$ / (11 x Tbit Min)

Tbit Min = $t_{WUUSART}$ / (11 x DWU max)

如果我们考虑理想情况，参数 DTRA、DQUANT、DREC 和 DTCL 为 0%，DWU max 为 3.41%；在实际中，我们至少需要考虑 HSI 的不准确性。

假设 HSI 误差 = 1%，$t_{WUUSART}$ = 3 μs(值为示例，正确值请参考芯片数据手册)：

DWU max = 3.41% - 1% = 2.41%

Tbit min = 3 μs / (11 x 2.41%) = 11.32 μs

此条件下，允许从停止模式正确唤醒的最大波特率为 1/11.32 μs = 88.36Kbaud。

22.5 USART 低功耗模式

模式	描述
SLEEP	无影响，USART 中断唤醒 SLEEP 模式
STOP	当 UESM 位被设置并且 USART 时钟被设置为 HSI 或 LSE 时，USART 能够从 STOP 模式唤醒 MCU。 MCU 从 STOP 模式唤醒可以使用任一标准 RXNE 或 WUF 中断。
STANDBY	USART 将被关闭，当设备退出 STANDBY 模式时，必须重新初始化。

22.6 USART 中断

中断事件	事件标志	使能控制位
发送数据寄存器空	TXE	TXEIE
CTS 中断	CTSIF	CTSIE
发送完成	TC	TCIE
接收数据寄存器非空（数据 ready 可读）	RXNE	RXNEIE
溢出错误检测	ORE	
空闲线检测	IDLE	IDLEIE
奇偶错误	PE	PEIE
LIN 断开	LBDF	LBDIE
噪声标志，溢出错误和多缓冲区通讯中的帧错误	NF or ORE or FE	EIE
字符匹配	CMF	CMIE
接收超时错误	RTOF	RTOIE
发现块尾	EOBF	EOBIE
从 Stop 模式唤醒	WUF(注 1)	WUFIE

注 1：WUF 中断只在 Stop 模式中有用。

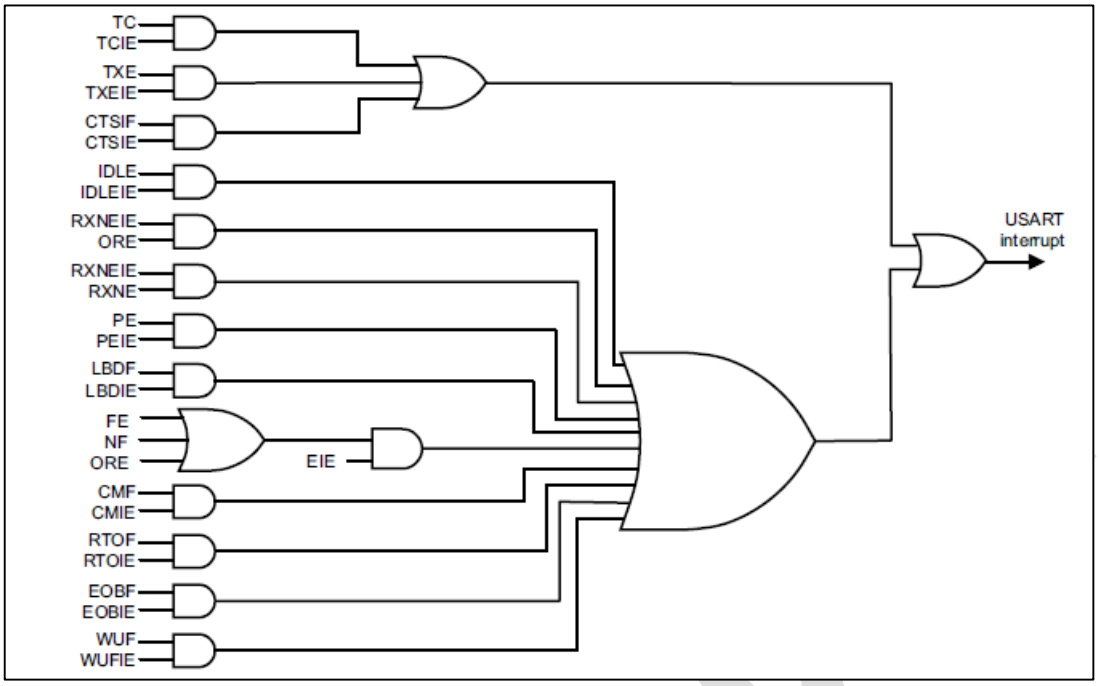
USART 中断时间全部连接到同一个中断向量 （ 见下图)

✧ 发送期间： 发送完成，CTS，发送数据寄存器空或帧错误（智能卡模式下）中断。

✧ 接收期间： 空闲线检测，溢出错误，接收数据寄存器非空，校验错误，LIN 断开检测， 噪声标志（仅在多缓冲区通讯时），帧错误（仅在多缓冲区通讯），字符匹配，等等。

如果设置了相应的使能控制位，这些事件都可以引起中断。

USART 中断映射图



22.7 相关寄存器

22.7.1 寄存器汇总表

基址：0x4001 3800				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**USART_CR1**	0x0000 0000	控制寄存器 1	22.7.2
0x04	**USART_CR2**	0x0000 0000	控制寄存器 2	22.7.3
0x08	**USART_CR3**	0x0000 0000	控制寄存器 3	22.7.4
0x0C	**USART_BRR**	0x0000 0000	波特率寄存器	22.7.5
0x10	**USART_GTPR**	0x0000 0000	保护时间和预分频寄存器	22.7.6
0x14	**USART_RTOR**	0x0000 0000	接收超时寄存器	22.7.7
0x18	**USART_RQR**	0x0000 0000	请求寄存器	22.7.8
0x1C	**USART_ISR**	0x0000 0C00	中断和状态寄存器	22.7.9
0x20	**USART_ICR**	0x0000 0000	中断标志清除寄存器	22.7.10
0x24	**USART_RDR**	0xXXXX XXXX	数据接受寄存器	22.7.11
0x28	**USART_TDR**	0xXXXX XXXX	数据发送寄存器	22.7.12

22.7.2 控制寄存器 1(USART_CR1)

USART_CR1			地址偏移	0x00		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	EOBIE	RTOIE	DEAT[4:3]	
R/W	-	-	-	-	rw	rw	rw	rw
复位值	-	-	-	-	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	DEAT[2:0]			DEDT[4:0]				
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	OVER8	CMIE	MME	M0	WAKE	PCE	PS	PEIE
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	UESM	UE

R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:28]	-	保留位，保持复位值
BIT[27]	EOBIE	块尾（end of block）中断使能，软件置 1 和清零。 0：中断禁止 1：当 USART_ISR 寄存器中的 EOBF 标志被置 1 时引发 USART 中断 *注：如果 USART 不支持智能卡模式，该位保留并由硬件强制为零。参见章节：USART 功能配备*
BIT[26]	RTOIE	接收超时（receiver timeout）中断使能，软件置 1 和清零。 0：中断禁止 1：当 USART_ISR 寄存器中的 RTOF 标志被置 1 时引发 USART 中断。 *注： 如果 USART 不支持接收超时功能，该位保留并由硬件强制为零。参见章节：USART 功能配备。*
BIT[25:21]	DEAT[4:0]	驱动使能提前时间（Driver Enable assertion time） 这个 5 位值定义 DE（驱动器使能）信号激活和第一个发送字节的起始位的时间间隔。它以采样时间为单位（1/8 或者 1/16 位时间，由过采样率决定） 这个位域只能在 USART 未被使能的时候（UE=0）改写。 *注： 如果 USART 不支持驱动器使能功能，该位保留，且必须保持为零。*
BIT[20:16]	DEDT[4:0]	驱动使能滞后时间（Driver Enable de-assertion time） 这个 5 位值定义了在传输的消息中，从最后一个停止位结束到 DE(驱动使能)信号去激活之间的时间。它以采样时间为单位（1/8 或者 1/16 位时间，由过采样率决定） 如果 USART_TDR 寄存器在 DEDT 时间内被改写，新的数据只会在 DEDT 和 DEAT 时间都结束之后才会被发送。 此位只能在 USART 禁用时（UE=0）改写。 *注：如果 USART 不支持驱动器使能功能，这个位保留，且必须保持为零。*
BIT[15]	OVER8	过采样模式（Oversampling mode） 0：16 倍过采样 1：8 倍过采样 此位只能在 USART 禁用时（UE=0）改写。 *注：在 LIN，IrDA 和智能卡模式中，这个位必须保持为零。*
BIT[14]	CMIE	字符匹配（Character match）中断使能，由软件置 1 和清零 0：中断禁止 1：当 USART_ISR 寄存器中的 CMF 标志被置 1 时引发 USART 中断
BIT[13]	MME	静默模式使能（Mute mode enable），由软件置 1 和清零。 这个位开启 USART 的静默模式功能。当置为 1 时，USART 可以在活动模式和静默模式之间切换，和 WAKE 位的定义一样。 0：接收器保持在活动模式（active mode） 1：接收器可以在活动模式和静默模式间切换。
BIT[12]	MO	字长（word length） 由软件置 1 和清零。 0：1 个起始位，8 个数据位，n 个停止位 1：1 个起始位，9 个数据位，n 个停止位 此位只能在 USART 禁用时（UE=0）改写。
BIT[11]	WAKE	接收器唤醒方式（Receiver wakeup method） 这个位决定 USART 从静默模式唤醒的方式。 由软件置 1 和清零。 0：空闲线（idle line） 1：地址标记（address mark） 此位只能在 USART 禁用时（UE=0）改写。
BIT[10]	PCE	校验控制使能（Parity control enable），由软件置 1 和清零。 这个位选择硬件校验控制（产生和检测）功能。当校验控制被打开，计算好的校验位被插入到最高位 MSB（M=01 时是第 9 位，M=00 时是第 8 位，M=10 时是第 7 位），并检测接收数据的校验位。一旦此位（PCE）被置 1，在当前字节（包括收发）之后激活校验控制。 0：校验控制禁止 1：校验控制使能 此位只能在 USART 禁用时（UE=0）改写。

BIT[9]	PS	校验选择（Parity selection） 当校验生成和检测开启（PCE=1），这个位选择使用奇校验还是偶校验。由软件置 1 和清零。 校验方式会在当前字节结束后生效。 0：偶校验 1：奇校验 此位只能在 USART 禁用时（UE=0）改写。
BIT[8]	PEIE	PE 中断使能，由软件置 1 和清零 0：中断禁止 1：在 USART_ISR 寄存器中的 PE 被置 1 的时候会产生 USART 中断。
BIT[7]	TXEIE	TXE 中断使能，由软件置 1 和清零 0：中断禁止 1：在 USART_ISR 寄存器中的 TXE 被置 1 的时候会产生 USART 中断
BIT[6]	TCIE	TC 中断使能，由软件置 1 和清零 0：中断禁止 1：在 USART_ISR 寄存器中的 TC 位被置 1 的时候会产生 USART 中断
BIT[5]	RXNEIE	RXNE 中断使能，由软件置 1 和清零 0：中断禁止 1：在 USART_ISR 寄存器中的 ORE 或者 RXNE 被置 1 的时候会产生 USART 中断。
BIT[4]	IDLEIE	空闲（IDLE）中断使能，由软件置 1 和清零 0：中断禁止 1：在 USART_ISR 寄存器中的 IDLE 位被置 1 的时候会产生 USART 中断
BIT[3]	TE	发送器使能（Transmitter enable） 这个位打开发送器。由软件置 1 和清零。 0：发送器关闭 1：发送器打开 *注 1：在发送期间，TE 位上的一个’0’脉冲（’0’后面跟一个 ’1’）会引起当前字发送完后跟着再发送一个线路空闲（idle line）信息出去，但在智能卡模式中则没有这个功能。要产生一个空闲字符，TE 位不可以立即被写 1 。为了确保所需的持续时间，软件可以轮询 USART_ISR 寄存器中的 TEACK 位。* *注 2：在智能卡模式下，当 TE 被置为 1 后，在开始发送前会有 1 个 bit 的延迟时间。*
BIT[2]	RE	接收器使能（Receiver enable） 这个位打开接收器。由软件置 1 和清零。 0：接收器关闭 1：接收器打开并开始侦测起始位
BIT[1]	UESM	stop 模式下 USART 使能（USART enable in Stop mode） 当这个位为 0，USART 不能将 MCU 从 Stop 模式下唤醒。 当这个位为 1，USART 可以将 MCU 从 Stop 模式下唤醒，条件是 USART 时钟选择 HSI 或 LSE（在 RCC 模块）。由软件置 1 和清零。 0：USART 不能从 Stop 模式中唤醒 MCU。 1：USART 可以从 Stop 模式中唤醒 MCU。 当这个功能被打开，USART 的时钟源必须为 HSI 或者是 LSE（参见 RCC 章节） *注 1：推荐在进入 Stop 模式前将这个 UESM 位置 1，并在退出 Stop 模式后将它清零。* *注 2：如果 USART 不支持从 Stop 模式唤醒功能，该位为保留位并由硬件强制为零。参见章节：USART 功能配备*
BIT[0]	UE	USART 使能 当这个位被清零，USART 的预分频器和输出都立即停止，并且当前的操作也被取消。USART 的配置会被保留，但 USART_ISR 中所有的状态标志都会被复位。此位由软件置 1 和清零。 0：USART 预分频器和输出关闭，低功耗模式 1：USART 开启 *注 1：为了进入低功耗模式而不在线路上产生错误，需要在清零 UE 位之前先清零 TE 位，并等待 USART_ISR 中的 TC 位被置 1 。* *注 2：在 UE=0 的时候，DMA 请求也同时被复位，所以在清零 UE 位之前 DMA 通道也必须先关闭。*

22.7.3 控制寄存器 2(USART_CR2)

USART_CR2			地址偏移	0x04		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	ADD[7:4]				ADD[3:0]			
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	RTONE	ABRMOD[1:0]		ABREN	MSBFIRST	DATAINV	TXINV	RXINV
R/W	rw	rw	rw	rw	rw	Rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	SWAP	LINEN	STOP[1:0]		CLKEN	CPOL	CPHA	LBCL
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	LBDIE	LBDL	ADDM7	-	-	-	-
R/W	-	rw	rw	rw	-	-	-	-
复位值	-	0	0	0	-	-	-	-

BIT[31:28]	ADD[7:4]	USART 节点的地址 这个位给出 USART 节点的地址或等待确认的字符码。 这用于静音模式或停止模式期间的多处理器通信，用于 7 位地址标记检测唤醒。发送器发出的字符的最高位（MSB）应该为 1。 也用在正常接收过程中的字符检测中，这时不打开静默状态(例如，在 ModBus 协议下对块尾的检测)。 这个时候，整个收到的 8 位字节与 ADD[7:0] 进行比较，如果匹配，CMF 标志被硬件置起。 这个位只能在接收器被关闭（RE=0）或者在 USART 被关闭的时候（UE=0）才能改写。
BIT[27:24]	ADD[3:0]	USART 节点的地址 这用于静音模式或停止模式期间的多处理器通信，用于地址标记检测唤醒。 这个位域只能在接收器被关闭（RE=0）或者在 USART 被关闭的时候（UE=0）才能改写。
BIT[23]	RTONE	接收超时使能（Receiver timeout enable） 由软件置 1 和清零 0：接收器超时检测功能关闭 1：接收器超时检测功能开启 当这个功能开启，只要 RX 线发现空闲(不是接收)达到 RTOR 寄存器(receiver timeout register）中设置的时间长度后，USART_ISR 寄存器中的 RTOF 标志会被硬件置 1。 *注： 如果 USART 不支持接收超时功能，该位为保留位并由硬件强制为零。参见章节：USART 功能配备。*
BIT[22:21]	ABRMOD[1:0]	自动波特率模式（Auto baud rate mode） 由软件置 1 和清零。 00：对起始位的测量被用来检测波特率。(接收数据必须以单个的‘1’作为开头，帧格式为 Start 1 0 xxxxxx） 01：使用下降沿对下降沿的测量。(接收数据必须以单个的‘1’作为开头，帧格式为 Start 1 0 xxxxxx） 10：保留 11：保留 这个位域只能在 ABREN=0 或者 USART 被禁止时（UE=0）改写。 *注 1：如果 DATAINV=1 和/或 MSBFIRST=1，要求和线路上使用相同的样式，(例如，0xAA 对于 MSBFIRST)* *注 2：如果 USART 不支持自动波特率功能，该位为保留位并由硬件强制为零。请参见章节：USART 功能配备。*
BIT[20]	ABREN	自动波特率使能（Auto baud rate enable） 由软件置 1 和清零 0：自动波特率检测被禁止。 1：自动波特率检测被打开

		注：如果 USART 不支持自动波特率功能，该位为保留位并由硬件强制为零。请参见章节：USART 功能配备。
BIT[19]	MSBFIRST	高位在前（Most significant bit first） 由软件置 1 和清零 0：数据发送和接收，起始位+数据位 LSB 在前（bit0）。 1：数据发送和接收，起始位+数据位 MSB 在前（bit7/8/9 取决 M[1:0]）。 这个位只能在 USART 禁止时（UE=0）改写。
BIT[18]	DATAINV	按位反向（Binary data inversion） 由软件置 1 和清零 0：数据寄存器中的逻辑数据在发送和接收时，采用正/直接逻辑。(1=H, 0=L) 1：数据寄存器中的逻辑数据在发送和接收时，采用负/反向逻辑。(1=L, 0=H). 校验位也一样反向。 这个位只能在 USART 禁止时（UE=0）改写。
BIT[17]	TXINV	TX 脚有效电平反向，由软件置 1 和清零 0：TX 脚信号工作于标准逻辑电平 (VDD =1/idle, Gnd=0/mark) 1：TX 脚信号被反向。 ((VDD =0/mark, Gnd=1/idle). 这可以用于 TX 线上带有外部反相器应用。 这个位只能在 USART 禁止时（UE=0）改写。
BIT[16]	RXINV	RX 引脚有效电平反向，由软件置 1 和清零 0：RX 脚信号工作于标准逻辑电平 (VDD =1/idle, Gnd=0/mark) 1：RX 脚信号被反向。 (VDD =0/mark, Gnd=1/idle)。这可以用于 RX 线上带有外部反相器应用。 这个位只能在 USART 禁止时（UE=0）改写。
BIT[15]	SWAP	交换 TX/RX 引脚，由软件置 1 和清零 0：TX/RX 引脚按照标准引脚分配来使用 1：TX 和 RX 的引脚功能交换。 这用于和其它 USART 口进行交叉互联的时候。 这个位只能在 USART 禁止时（UE=0）改写。
BIT[14]	LINEN	LIN 模式，由软件置 1 和清零 0：LIN 模式禁止 1：LIN 模式使能 LIN 模式打开后，可以具备发送 LIN 同步断开符（13 个低 bit）的功能，用 USART_ CR1 寄存器的 SBKRQ 位实现，同时还具备 LIN 同步断开信号的检测功能。 这个位只能在 USART 禁止时（UE=0）改写。 *注：如果 USART 不支持 LIN 模式，该位为保留位并由硬件强制为零。请参见章节：USART 功能配备。*
BIT[13:12]	STOP[1:0]	停止位 这些位用来配置停止位的个数 00：1 个停止位： 01：保留 10：2 个停止位 11：1.5 个停止位 这个位只能在 USART 禁止时（UE=0）改写。
BIT[11]	CLKEN	时钟使能 这个位用来打开 CK 引脚的功能 0：CK 引脚被禁止 1:：CK 引脚被使能 这个位只能在 USART 禁止时（UE=0）改写。 *注 1：如果同步模式和智能卡模式都不被支持，这个位保留并由硬件强制为 0 。请参见章节：USART 功能配备。* *注 2：CLKEN = 1 时（CK 一直有效），为了保证提供给智能卡 CK 时钟的正确性，无论 UE 位是否已使能，必须遵循以下步骤:* - *UE=0* - *SCEN=1* - *GTPR 配置（如果需要配置 PSC，建议单次访问 USART_ GTPR 寄存器一并配置 PSC 和 GT ）* - *CLKEN=1* - *UE=1*

BIT[10]	CPOL	时钟极性（Clock polarity） 这个位允许用户选择同步模式下 CK 引脚上的时钟输出的极性。它连同 CPHA 位一起决定所需要的时钟/数据时序关系。 0：在没有数据传输的时候保持低电平 1：在没有数据传输的时候保持高电平 这个位只能在 USART 禁止时（UE=0）改写。 *注：如果同步模式不被支持，这个位保留并由硬件强制为 0 。*
BIT[9]	CPHA	时钟相位（Clock phase） 这个位允许用户选择同步模式下 CK 引脚上的时钟输出的相位。它连同 CPOL 位一起决定所需要的时钟/数据时序关系。 0：第一个时钟沿对准第一位数据 1：第二和时钟沿对准第一位数据 这个位只能在 USART 禁止时（UE=0）改写。 *注：如果同步模式不被支持，这个位保留并由硬件强制为 0 。*
BIT[8]	LBCL	末位时钟脉冲（Last bit clock pulse） 这个位用来选择在同步模式下，CK 脚上传输最后一个位（MSB）的时候是否必须输出时钟脉冲。 0：CK 脚上在传输末位数据的时候不输出时钟脉冲。 1：CK 脚上在传输末位数据的时候输出时钟脉冲。 这个位只能在 USART 禁止时（UE=0）改写。 *注 1：末位是第 7、8 位还是第 9 位，取决于 USART_CR1 寄存器中的 M 位的设置。* *注 2：如果同步模式不被支持，这个位保留并由硬件强制为 0 。*
BIT[7]	-	保留，保持复位值
BIT[6]	LBDIE	LIN 断开信号监测中断使能（LIN break detection interrupt enable） 断开中断屏蔽（利用断开分隔符来检测） 0：中断禁止 1：在 USART_ISR 寄存器中的 LBDF 被置 1 的时候会产生 USART 中断 *注：如果 LIN 模式不被支持，这个位保留并由硬件强制为 0 。*
BIT[5]	LBDL	LIN 断开监测长度（LIN break detection length） 这个位用来选择使用 11 位还是 10 位断开检测。 0：10 位断开检测 1：11 位断开检测 这个位只能在 USART 禁止时（UE=0）改写。 *注： 如果 LIN 模式不被支持，这个位保留并由硬件强制为 0 。*
BIT[4]	ADDM7	7 位地址监测或 4 位地址监测(7-bit Address Detection/4-bit Address Detection) 这个位用来选择使用 4 位地址检测还是 7 位地址检测。 0：4 位地址检测 1：7 位地址检测（在 8 位数据模式下） 这个位只能在 USART 禁止时（UE=0）改写。 *注：在 7 位和 9 位数据模式下，地址检测分别按 6 位和 8 位地址(ADD[5:0]和 ADD[7:0]）操作。*

22.7.4 控制寄存器 3(USART_CR3)

USART_CR3			地址偏移	0x08		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	WUFIE	WUS		SCARCNT[2:0]			-
R/W	-	rw	rw	rw	rw	rw	rw	-
复位值	-	0	0	0	0	0	0	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	DEP	DEM	DDRE	OVRDIS	ONEBIT	CTSIE	CTSE	RTSE
R/W	rw	rw	rw	rw	rw	rw	rw	Rw
复位值	0	0	0	0	0	0	0	0

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	DMAT	DMAR	SCEN	NACK	HDSEL	IRLP	IREN	EIE
R/W	rw	rw	v	v	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[22]	WUFIE	从 Stop 模式唤醒中断使能，能软件置一和清零 0：中断禁止 1：在 USART_ISR 寄存器中的 WUF 被置 1 的时候会产生 USART 中断 注： 1. WUFIE 位必须在进入 Stop 模式之前设置。 2. WUF 中断只在 Stop 模式中有用。 3. 如果 USART 不支持从 Stop 模式唤醒功能，该位为保留位并由硬件强制为零。
BIT[21:20]	WUS[1:0]	从 Stop 模式唤醒中断标志选择这个位域制定激活 WUF 标志的事件。 00：在发生地址匹配世家的时候激活 WUF 01：保留 10：WUF 在检测到起始位的时候激活 11：WUF 在得到接受数据寄存器非空事件时激活 这个位域只能在 USART 未被使能的时候（UE=0）改写。 注： 如果 USART 不支持从 Stop 模式唤醒功能，该位为保留位并由硬件强制为零。
BIT[19:17]	SCARCNT[2:0]	智能卡模式重试计数器 这个位域指定智能卡模式中接收和发送的重试次数。 在发送模式下，它指定的是在产生发送错误（FE=1）之前自动重试发送的次数。 在接收模式下，它指定的是在产生接收错误事件前（RXNE 和 PE=1）所作的错误接收的尝试的次数。 这个位域只能在 USART 未被使能的时候（UE=0）改写。 当 USART 被使能，这个位域只允许写成 0x0，这是为了避免盲目的自动重发数据。0x0：重发功能关闭 – 在发送模式下不进行自动重发操作。 0x1 to 0x7：自动重传的尝试次数（在提示错误之前） 注： 如果智能卡模式不被支持，这个位保留并由硬件强制为 0 。
BIT[15]	DEP	驱动使能输出脚的极性选择 0：DE 信号高有效 1：DE 信号低有效 注： 如果串口不支持驱动器使能功能，这个位保留，且必须保持为零。 这个位域只能在 USART 未被使能的时候（UE=0）改写。
BIT[14]	DEM	驱动器使能模式 它允许用户通过 DE（驱动使能）信号来激活外部收发器的控制端。0：DE 功能被禁止 1：DE 功能被打开。 DE 信号在 RTS 脚输出。 注： 如果串口不支持驱动器使能功能，这个位保留，且必须保持为零。 这个位域只能在 USART 未被使能的时候（UE=0）改写。
BIT[13]	DDRE	在接收错误的时候禁止 DMA 0：在发生接收错误的时候不禁止 DMA。 相应的错误标志被置 1，但 RXNE 仍保持零以阻止数据溢出覆盖。 作为结果，不会发起 DMA 请求，所以错误的数据不会被传输（因为没有 DMA 请求），但下一个正确的数据会被传送。（用于智能卡模式）1：在发生接收错误之后，DMA 被关闭。相应的错误标志会随着 RXNE 的置 1 而被置 1。 DMA 请求会被屏蔽，直到相关的错误标志被清零。 这意味着软件必须先禁止 DMA 请求（DMAR=0）或者在清零错误标志前先清零 RXNE 标志。 这个位域只能在 USART 未被使能的时候（UE=0）改写。 注： 接收错误指的是： 校验错，帧错误或噪声错误。
BIT[12]	OVRDIS	溢出检测禁止，这个位用于禁止对接收溢出现象的检测 0：当之前接收到的数据没有在新接收到数据之前读走时，会引起溢出错误标志 ORE 被硬件置 1。 1：溢出检测功能关闭。 如果在新的接收数据到来时，RXNE 标志仍然是 1，但 ORE 标志还不是 1 时，新的数据会将 USART_RDR 中以前的内容覆盖掉。这个位域只能在 USART 未被使能的时候（UE=0）改写。 注： 这个控制位允许读取数据的时候检查通讯数据流。

BIT[11]	ONEBIT	单次采样方式使能 这个位允许用户选择采样方式。 当选择单次采样方式的时候，噪声监测标志（NF） 就被禁止了。 0：三次采样方式 1：单次采样方式 这个位域只能在 USART 未被使能的时候（UE=0）改写。
BIT[10]	CTSIE	CTS 中断使能 0：中断禁止 1：在 USART_ISR 寄存器中的 CTSIF 被置 1 的时候会产生 USART 中断 注： 如果硬件流控制不被支持，这个位保留并由硬件强制为 0 。
BIT[9]	CTSE	CTSE 功能使能 0：关闭 CTS 硬件流控 1：打开 CTS 硬件流控，只有在 nCTS 输入上收到有效信号（被拉低）时才会发送数据。如果在数据传送时 nCTS 输入变为无效，会在数据发送完成后再停。如果写数据到发送寄存器的时候，nCTS 处于无效状态，那么这个数据的发送会被推迟直到 nCTS 信号变成有效。 这个位域只能在 USART 未被使能的时候（UE=0）改写。 注： 如果硬件流控制不被支持，这个位保留并由硬件强制为 0 。 请参见表：USART 具体功能配备。
BIT[8]	RTSE	RTS 使能 0：关闭 RTS 硬件流控 1：RTS 输出使能，只会在有接收空间的时候才请求下一个数据。当前数据发送完成后，发送操作就需要暂停下来。如果可以接收数据了，将 nRTS 输出置为有效（拉低）。这个位域只能在 USART 未被使能的时候（UE=0）改写。 注： 如果硬件流控制不被支持，这个位保留并由硬件强制为 0 。 请参见表：USART 具体功能配备。
BIT[7]	DMAT	DMA 发送使能，该位由软件置一和清零 1：为发送数据使能 DMA 模式 0：关闭发送 DMA 模式
BIT[6]	DMAR	DMA 接收使能，该位由软件置一和清零 1：为接收数据使能 DMA 模式 0：关闭接收 DMA 模
BIT[5]	SCEN	智能卡模式使能，该位用于打开只能卡模式 0：关闭智能卡模式 1：打开智能卡模式 这个位域只能在 USART 未被使能的时候（UE=0）改写。 注： 如果 USART 不支持智能卡模式，该位为保留为并由硬件强制为零。
BIT[4]	NACK	智能卡 NACK 发送使能 0：出现校验错误的时候不发送 NACK 1：出现校验错误的时候，发送 NACK 这个位域只能在 USART 未被使能的时候（UE=0）改写。 注： 如果 USART 不支持智能卡模式，该位为保留位并由硬件强制为零。
BIT[3]	HDSEL	半双工选择 选择单线半双工模式 0：不选择半双工模式 1：选择半双工模式 这个位域只能在 USART 未被使能的时候（UE=0）改写。
BIT[2]	IRLP	IrDA 低功耗模式选择 用来选择 IrDA 的普通模式还是低功耗模式 0：正常模式 1：低功耗模式 这个位域只能在 USART 未被使能的时候（UE=0）改写。 注： 如果 IrDA 模式不被支持，这个位保留并由硬件强制为 0 。
BIT[1]	IREN	红外模式使能，由软件置一和清零 0：不使能红外模式 1：使能红外模式 这个位域只能在 USART 未被使能的时候（UE=0）改写。 注： 如果 IrDA 模式不被支持，这个位保留并由硬件强制为 0 。 请参见表：USART 具体功能配备。

BIT[0]	EIE	错误中断使能 在允许帧错误，溢出错误或噪声错误产生中断请求时要打开这个开关。 0：中断禁止 1：当 USART_ISR 寄存器中的 FE=1 或 ORE=1 或 NF=1 时，会产生中断。

22.7.5 波特率寄存器(USART_BRR)

USART_BRR			地址偏移	0x0C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	BRR[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	BRR[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:4]	DIV Mantissa[11:0]	UASRTDIV 的整数部分 这 12 位定义 USART 分频器触发因子的整数部分
BIT[3:0]	DIV Mantissa[3:0]	USARTDIV 分配器触发因子的小数部分 这 4 位定义 USART 分频器触发因子的小数部分，当 OVER 8 = 1 时，DIV_Fraction3 位是没有用的，并且必须保持为 0.

22.7.6 保护时间和预分频器寄存器(USART_GTPR)

USART_GTPR			地址偏移	0x10		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	GT[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	PSC[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[15:8]	GT[7:0]	保护时间值 这个位域用来设置保护时间长度，以波特时钟为单位。 在智能卡模式下要用到。 在保护时间过去之后才会设置发送完成标志。这个位域只能在 USART 未被使能的时候（UE=0）改写。 注： 如果智能卡模式不被支持，这个位保留并由硬件强制为 0。
BIT[7:0]	PSC[7:0]	预分频器值 -在红外低功耗和正常模式下:PSC[7:0]=IrDA 正常和低功耗模式波特率对 USART

		的时钟源进行分频以获得低功耗模式下的频率： 时钟源按寄存器给定的值来分频（8 位有效）： 0000 0000：保留-不要写这种无聊的值 0000 0001：1 分频 0000 0010：2 分频 ... - 智能卡模式： PSC[4:0]：预分频器值 用于设定对 USART 时钟源的分频系数，得到智能卡时钟。按寄存器中的值（低 5 位有效）乘以 2 得到的值作为分频系数来分频： 0 0000：保留 - 不要写这种无聊的值 0 0001：2 分频 0 0010：4 分频 0 0011：6 分频 ... 这个位域只能在 USART 未被使能的时候（UE=0）改写。 注： 如果智能卡模式不被支持，这个位保留并由硬件强制为 0 。

22.7.7 接收超时寄存器(USART_RTOR)

USART_RTOR			地址偏移	0x14		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	BLEN[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	RTO[23:16]							
R/W	rw	rw	w	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	RTO[15:8]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	RTO[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	0	0	0	0	0	0	0	0

BIT[31:24]	BLEN[7:0]	块长度 这个位域给出了智能卡模式 T=1 的接收时的块长度。 这个值等于 信息块字符数 +结束部分（1-LEC/2-CRC）-1 。例如： BLEN = 0-> 0 个信息字符 + LEC BLEN = 1-> 0 个信息字符 + LEC BLEN = 255 -> 254 个信息字符 +CRC（ 总共 256 个字符） 智能卡模式中， 当 TXE=0，会导致块长度计数器清零。 这个位域也可以在其它模式中使用。 这时，块长度计数器在 RE=0 的时候清零和 /或 在 EOBCF 位被写 1 的时候清零。 注： 这个值可以在块接收开始之后再去设置（用于需要从块的序言部分提取块长度的情形）。每个块接收的时候只能设置一次。
BIT[23:0]	RTO[23:0]	接收超时值 这个位域给出了接收超时的设置，单位是波特时钟的时长。 标准模式下，最后一个字节接收后，在整个 RTO 值规定的时长内，再也没有检测到新的起始位的时候，RTOF 标志会被硬件置 1。 在智能卡模式中，这个值被用来实现 CWT 和 BWT。 详细信息参见智能卡相关章节。这里，超时测量时从最后一个字节的其实位开始算的。 注：对于每个接收字符只能改写一次这个值。 注：RTOR 可以边运行边改写，如果新值小于等于已经产生的超时计数，无非就是马上导致 RTOF 标志被置 1 。

22.7.8 请求寄存器(USART_RQR)

USART_RQR			地址偏移	0x18		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	TXFRQ	RXFRQ	MMRQ	SBKRQ	ABRRQ
R/W	-	-	-	w	w	w	w	w
复位值	-	-	-	0	0	0	0	0

BIT[4]	TXFRQ	发送数据清空请求 对这个位写 1 会直接令 TX 标志被硬件置 1 这里允许取消数据发送。 这个位只能在智能卡模式下使用，当数据由于所发生的错误导致还没发出去，USART_ISR 寄存器的 FE 标志是 1 的时候用。 如果 USART 不支持智能卡模式，该位为保留位并由硬件强制为零。
BIT[3]	RXFRQ	接收数据清空请求 对这个位写 1 会直接令 RXNE 标志被硬件清零 这里允许直接丢弃还没有读的接收数据，免得被提示溢出错误。
BIT[2]	MMRQ	静默模式请求 对这个位写 1 将导致 USART 进入静默模式，同时置位 RWU 标志。
BIT[1]	SBKRQ	请求发送断开字符 对这个位写 1 会令 SBKF 标志置 1，并在发送状态机可用的时候向线路发出一个断开字符。
BIT[0]	ABRRQ	自动波特率检测请求 向这个位写 1 会清除 USART_ISR 中的 ABRF 标志，并在下一次数据接收的时候开始一次自动波特率测量。 注： 如果 USART 不支持自动波特率功能，该位为保留位并由硬件强制为零。

22.7.9 中断和状态寄存器(USART_ISR)

USART_ISR			地址偏移	0x1C		复位值	0x0000 0C00	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	REACK	TEACK	WUF	RWU	SBKF	CMF	BUSY
R/W	-	r	r	r	r	r	r	r
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	ABRF	ABRE	-	EOBF	RTOF	CTS	CTSIF	LBDF
R/W	r	r	-	r	r	r	r	r
复位值	0	0	-	0	1	1	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	TXE	TC	RXNE	IDLE	ORE	NF	FE	PE
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0

BIT[22]	REACK	接收使能通知标志 这个位有硬件控制，当接收使能信号被 USART 读到的时候，会给出一个回执。这可以在进入 Stop 模式之前，用来确认 USART 是不是已经准备好接收了。 注： 如果 USART 不支持从 Stop 模式唤醒功能，该位为保留位并由硬件强制为零。
BIT[21]	TEACK	发送使能通知标志 这个位有硬件控制，当发送使能信号被 USART 读到的时候，会给出一个回执。这可以在写 USART_CR1 寄存器的 TE=0 以产生一个空闲帧请求，跟着又将 TE 写成 1 的时候，用于保证 TE=0 的最小周期的时候用。
BIT[20]	WUF	从 Stop 模式唤醒标志 当检测到唤醒事件时由硬件置 1。 具体时间有 WUS 位域来定义。 由软件向 USART_ICR 寄存器的 PECF 位写 1，可以清除这个标志。 如果 USART_CR3 寄存器的 WUF=1，会产生中断请求。 注： 1. 当 UESM 被清零， WUF 标志也会被清零。 2. WUF 中断只在 Stop 模式中有用。 3. 如果 USART 不支持从 Stop 模式唤醒功能，该位为保留位并由硬件强制为零。
BIT[19]	RWU	接收器从静默模式唤醒 这个位表示 USART 处于静默模式时，当唤醒和静默模式切换时，由硬件清零和置 1。静默模式控制顺序（地址还是空闲）用 USART_CR1 寄存器的 WAKE 位来选择。如果选择由空闲信号唤醒，那这个位就只能由软件置 1 了，方法是向 USART_RQR 寄存器置 1。 0: 接收器处于活动模式 1: 接收器处于静默模式 注： 如果 USART 不支持从 Stop 模式唤醒功能，该位为保留位并由硬件强制为零。
BIT[18]	SBKF	断开信号发送标志 这个位表示当要求发送一个断开字符时， 由软件置 1，方法是向 USART_CR3 寄存器的 SBKRQ 位写 1。 当断开字符的停止位传出后，由硬件自动清零。 0: 没发送断开字符 1: 断开字符将会被发
BIT[17]	CMF	字符匹配标志 当收到一个字符和 ADD[7:0] 中设置的内容相同时，由硬件置 1。 由软件向 USART_ ICR 寄存器的 CMCF 位写 1，可以清除这个标志。如果 USART_CR1 寄存器中的 CMIE 位是 1，就会产生中断请求。 : 没发现字符匹配 1: 有发现字符匹
BIT[16]	BUSY	忙标志 由硬件置 1 和清零。 当 RX 线在通讯时（成功检测到起始位），这个位被硬件置 1。接收结束后（不管成功与否）会由硬件清零。 0: USART 处于空闲（没接收） 1: 正在接收数据
BIT[15]	ABRF	自动波特率检测标志 当自动波特率功能打开时（RXNE 也被置 1，如果 RXNEIE=1 产生中断）或者当自动波特率操作不成功时（ABRE，RXNE 和 FE 也都被置 1 的时候），这个位被硬件置 1。 开始一轮新的波特率检测（向 USART_RQR 寄存器的 ABRQ 写 1）的时候会被清除。 注： 如果 USART 不支持自动波特率功能，该位为保留位并由硬件强制为零。
BIT[14]	ABRE	自动波特率检测错误 在波特率测量失败的时候（超量程或者字符比较失败）由硬件置一。 由软件清零，方法是向 USART_CR3 寄存器的 ABRRQ 位写 1。 注： 如果 USART 不支持自动波特率功能，该位为保留位并由硬件强制为零。
BIT[12]	EOBF	块结束标志 当一个完整的块被接收时由硬件置 1（例如 T=1 智能卡模式中）。当收到的字节数（从块开始，包括序言部分）大于等于 BLEN+4 的时候完成检测。 如果 USART_CR2 寄存器的 EOBIE=1，会产生中断请求。 由软件向 USART_ICR 寄存器的 EOBCF 位写 1，可以清除这个标志。 0: 还没到块结束

		1：块结束（足够字节数）已经到了 注： 如果智能卡模式不被支持，这个位保留并由硬件强制为 0 。
BIT[11]	RTOF	接收超时标志 如果没有通讯的时间长度达到了 RTOR 寄存器中设定的超时值，这个位会被硬件置 1。 由软件向 USART_ICR 寄存器的 RTOCF 位写 1，可以清除这个标志。如果 USART_CR2 寄存器的 RTOIE=1，会产生中断请求。 在智能卡模式中，这个超时相当于 CWT 或 BWT 计时。 0：尚未超时 1：已经达到超时 注： 1. 如果 RTOR 寄存器中设定的时间点正好位于两个字符中， RTOF 不会被置 1 。而如果这个时间达到了设定值 +2 个采样周期（ 2/16 或 2/8，取决于过采样方式的设定），RTOF 标志会被置 1 。 2.在 RE=0 的时候还是会进行超时计数，但 RTOF 只会在 RE=1 的时候置 1 。如果在 RE 被置 1 的时候，超时早就过了， RTOF 会马上被置 1 。 3. 如果 USART 不支持接收超时功能，该位为保留位并由硬件强制为零。
BIT[10]	CTS	CTS 标志 该位由硬件置 1 和清零 这是 nCTS 输入脚状态的反向拷贝。 0：nCTS 线是高 1：nCTS 线是低 注： 如果硬件流控制不被支持，这个位保留并由硬件强制为 0 。
BIT[9]	CTSIF	CTS 中断标志 如果 CTSE 位为 1，那么当 nCTS 输入状态变化的时候，由硬件置 1。 由软件向 USART_ICR 寄存器的 CTSCF 位写 1，可以清除这个标志。如果 USART_CR3 寄存器的 CTSIE=1，会产生中断请求。0：nCTS 线的状态无变化 1：nCTS 线的状态有变化 注： 如果硬件流控制不被支持，这个位保留并由硬件强制为 0 。
BIT[8]	LBDF	LIN 断开检测 当检测到 LIN 断开字符时由硬件置 1。由软件向 USART_ICR 寄存器的 LBDCF 位写 1， 可以清除这个标志。如果 USART_CR2 寄存器的 LBDIE=1，会产生中断请求。 0：没检测到 LIN 断开字符 1：检测到 LIN 断开字符 注： 如果 USART 不支持 LIN 模式，该位为保留位并由硬件强制为零。
BIT[7]	TXE	发送数据寄存器空 当 USART_TDR 寄存器中的值被取到移位寄存器的同时，这个位被硬件置 1。 再向 USART_TDR 寄存器写数据就会同时清掉这个位。 TXE 标志还可以用其它方式清除，例如向 USART_RQR 寄存器的 TXFRQ 位写 1，为了丢弃数据（仅于智能卡模式 T=0，传送失败的情形）。如果 USART_CR1 寄存器中的 TXEIE 位被置起时，则会产生中断。 0：没有数据被传到移位寄存器 1：有数据被传到移位寄存器，发送数据寄存器为空。 注： 这个位在单缓冲区发送的时候会用到。
BIT[6]	TC	发送完成 在 TXE 为 1 的条件下， 当数据发送完成的时候， 这个位会被硬件置 1。 如果 USART_CR1 寄存器中的 TCIE 位是 1，就会产生中断请求。向 USART_TDR 寄存器再次写入数据，或者向 USART_ICR 寄存器的 TCCF 位写 1，都可以清除这个标志。如果 USART_CR1 寄存器中的 TCIE 位是 1，就会产生中断请求。 0：发送未完成 1：发送完成 注： 如果 TE 位被清零，并且没有在发送数据，那 TC 位会立即被置 1 。
BIT[5]	RXNE	接收数据寄存器非空 当接收移位寄存器的内容被传递到 USART_RDR 寄存器中时，这个位被硬件置 1。读取 USART_TDR 寄存器的数据就会同时清掉这个位。 RXNE 标志也可以通过对 USART_RQR 寄存器中的 RXFRQ 位写 1 来清除。如果 USART_CR1 寄存器中的 RXNEIE 位是 1，就会产生中断请求。 0：没收到数据 1：收到的数据已经可读
BIT[4]	IDLE	空闲线检测 当检测到线路空闲时由硬件置 1。如果 USART_CR1 寄存器中的 IDLEIE 位是 1，

		就会产生中断请求。 由软件向 USART_ICR 寄存器的 IDLECF 位写 1，可以清除这个标志。 0：没有检测到线路空闲 1：检测到线路空闲 注： 1. 除非 RXNE 位被置 1 ，否则 IDLE 位不会再次置 1（例如 ： 一个新的线路空闲信号来临）。 2. 如果静默模式被使能了（ MME=1 ），只要 USART 没有处于静默状态（ RWU=0 ），那么 IDLE 会被置 1 ，而无论在 WAKE 位上用什么方式设置的静默功能。如果 RWU=1，那 IDLE 就不会被置 1 了。
BIT[3]	ORE	溢出错误 在 RXNE=1 的的条件下（也就是上次数据还没有读走），串口接收寄存器又接收好了一个字节的数据并准备往 RDR 寄存器去转移的时候，会由硬件将这个位置 1。由软件向 USART_ICR 寄存器的 ORECF 位写 1，可以清除这个标志。 如果 USART_CR1 寄存器中的 RXNEIE 位或 EIE 位是 1，就会产生中断请求。 0：没有溢出错误 1：检测到溢出错误 注： 1. 当这个位被置 1 ， RDR 寄存器中的数据不会丢，但移位寄存器中的（那个新的）数据就会蒸发掉了。如果在多缓冲区通讯时 EIE 位是 1，并且 ORE 标志被置 1 的话，就会同步引起一个中断请求。 2. 如果 USART_CR3 寄存器中的 OVRDIS 位是 1，那么这个位就会被长期的强制为零（没有了溢出检测功能）。
BIT[2]	NF	噪声检测标志 当收帧的时候检测到噪声，这一位会有硬件置一。由软件向 USART_ICR 寄存器的 NFCF 位写 1，可以清除这个标志。 0：没有检测到噪声 1：检测到噪声 注： 1. 这一位不会产生中断，因为和它同时置 1 的 RXNE 能够产生中断。如果在多缓冲区通讯时 EIE 位是 1，并且 NF 标志被置 1 的话，也会同步引起一个中断请求。 2. 当线路不会受到噪声干扰，可以将这个噪声检测功能关闭，从而降低 USART 对时钟偏差的敏感度，具体做法是将 ONEBIT 位写成 1。（参见章节 24.5.5 ）：USART 对时钟偏差的容忍度）
BIT[1]	FE	帧错误 当一个不同步现象、强噪声或一个断开符号被检测到的时候，这个位有硬件置 1。由软件向 USART_ICR 寄存器的 FECF 位写 1，可以清除这个标志。 在智能卡模式中发送数据时，当重发尝试的次数达到上限，由没有收到成功的回应（卡一直响应 NACK）的时候，这个位也会被硬件置 1。 如果 USART_CR1 寄存器中的 EIE 位是 1，会产生中断请求。 0：没有检测到帧错误 1：有检测到帧错误或者有收到断开字
BIT[0]	PE	校验错误标志 当在接收数据的时候发现校验错误，这个位会由硬件置 1。 由软件向 USART_ICR 寄存器的 PECF 位写 1，可以清除这个标志。如果 USART_CR1 寄存器中的 PEIE 位是 1，会产生中断请求。 0：没有校验错误 1：有校验错误

22.7.10 中断标志清除寄存器(USART_ICR)

USART_ICR			地址偏移	0x20		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	WUCF	-	-	CMCF	-
R/W	-	-	-	rc_w1	-	-	rc_w1	-
复位值	-	-	-	0	-	-	0	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8

控制位	-	-	-	EOBCF	RTOCF	-	CTSCF	LBDCF
R/W	-	-	-	rc_w1	rc_w1	-	rc_w1	rc_w1
复位值	-	-	-	0	0	-	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	TCCF	-	IDLECF	ORECF	NCF	FECF	PECF
R/W	-	rc_w1	-	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
复位值	-	0	-	0	0	0	0	0

BIT[20]	WUCF	从 Stop 模式唤醒的标志的清除 对这个位写 1，会清除 USART_ISR 寄存器中的 WUF 标志位。 注： 如果 USART 不支持从 Stop 模式唤醒功能，该位为保留位并由硬件强制为零。
BIT[17]	CMCF	字符匹配标志的清除 对这个位写 1，会清除 USART_ISR 寄存器中的 CMF 标志位
BIT[12]	EOBCF	块结束标志的清除 对这个位写 1，会清除 USART_ISR 寄存器中的 EOBF 标志位。 注： 如果 USART 不支持智能卡模式，该位为保留位并由硬件强制为零。 请参见表：USART 具体功能配备。
BIT[11]	RTOCF	接收超时标志的清除 对这个位写 1，会清除 USART_ISR 寄存器中的 RTOF 标志位。 注： 如果 USART 不支持接收超时功能，该位为保留位并由硬件强制为零。 请参见表：USART 具体功能配备。
BIT[9]	CTSCF	CTS 标志的清除 对这个位写 1，会清除 USART_ISR 寄存器中的 CTSIF 标志位。 注： 如果硬件流控制不被支持，这个位保留并由硬件强制为 0 。 请参见表：USART 具体功能配备。
BIT[8]	LBDCF	LIN 断开检测标志的清除 对这个位写 1，会清除 USART_ISR 寄存器中的 LBDF 标志位。 注： 如果 LIN 模式不被支持，这个位保留并由硬件强制为 0 。 请参见表：USART 具体功能配备。
BIT[6]	TCCF	发送完成标志的清除 对这个位写 1，会清除 USART_ISR 寄存器中的 TC 标志
BIT[4]	IDLECF	线路空闲检测标志的清除 对这个位写 1，会清除 USART_ISR 寄存器中的 IDLE 标志位。
BIT[3]	ORECF	溢出错误标志的清除 对这个位写 1，会清除 USART_ISR 寄存器中的 ORE 标志位。
BIT[2]	NCF	噪声检测标志的清除 对这个位写 1，会清除 USART_ISR 寄存器中的 NF 标志位。
BIT[1]	FECF	帧错误标志的清除 对这个位写 1，会清除 USART_ISR 寄存器中的 FE 标志位。
BIT[0]	PECF	校验错误标志的清除 对这个位写 1，会清除 USART_ISR 寄存器中的 PE 标志位。

22.7.11 数据接收寄存器(USART_RDR)

USART_RDR			地址偏移	0x24		复位值	0xXXXX XXXX	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	RDR[8]
R/W	-	-	-	-	-	-	-	rw
复位值	-	-	-	-	-	-	-	x

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	RDR[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	r
复位值	x	x	x	x	x	x	x	x

BIT[8:0]	RDR[8:0]	接收数据的值 包含所收到的字节。 RDR 寄存器提供输入移位寄存器和内部总线间的并行接口（见模块框图）。当接收数据时打开了校验位，读这个寄存器得到的最高位是校验位。

22.7.12 数据发送寄存器(USART_TDR)

USART_TDR			地址偏移	0x28		复位值	0xXXXX XXXX	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	TDR[8]
R/W	-	-	-	-	-	-	-	rw
复位值	-	-	-	-	-	-	-	x
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	TDR[7:0]							
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	x	x	x	x	x	x	x	x

BIT[8:0]	TDR[8:0]	发送数据的值 用于写入要发送的数据字节。 TDR 寄存器提供发送移位寄存器和内部总线间的并行接口（见模块框图）。 当发送的时候设置了校验功能（USART_CR1 中的 PCE=1），向最高位（位 7 还是位 8 取决于设置的字长）写入的信息是无效的，因为它总是要被校验位代替了之后再去发送的。 *注： 这个寄存器只能在 TXE=1 的时候才能做写操作。*

23 模拟电压比较器（CMP）

23.1 概述

本产品内嵌 1 个正相 6 通道（3 外部+3 内部通道）切换的比较器(CMP1)和 1 个正相 4 通道（1 外部+3 内部通道）的比较器(CMP2)，每个比较器都有反相端内部比较电压可选功能，施密特窗口档位选择，输出信号滤波及极性改变功能。同时最终的输出信号可以反馈到芯片管脚，内部中断，同时也能接入 TIMx 与其产生联动，或是作为保护信号接入 PWM 刹车功能。

23.2 特性

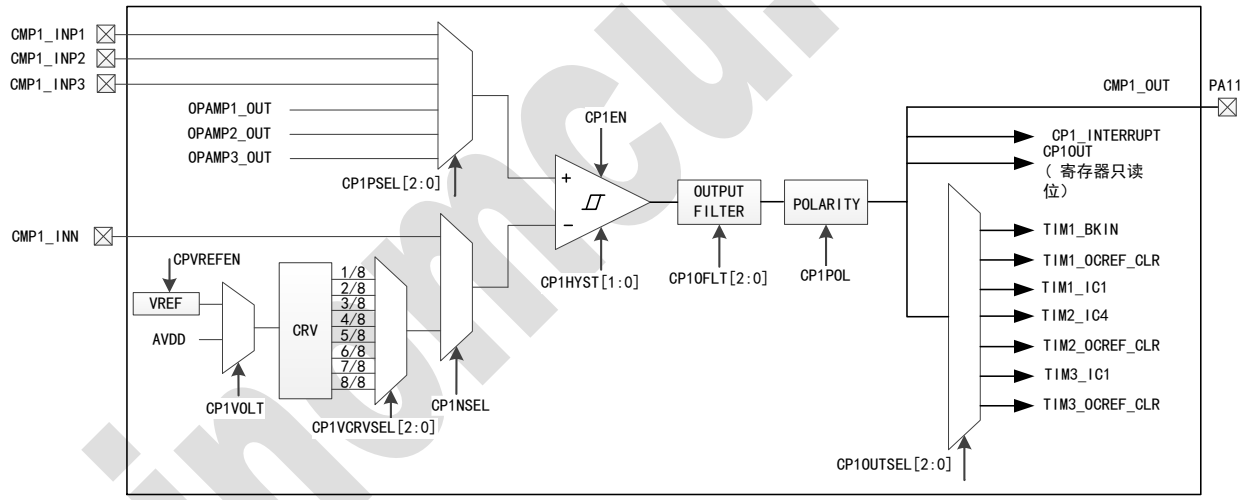
- ✧ 比较器 1 支持正相 6 通道选择
- ✧ 比较器 2 支持正相 4 通道选择
- ✧ 支持反相多级电压可选
- ✧ 支持施密特窗口档位配置
- ✧ 支持输出滤波和极性控制
- ✧ 输出信号连接至 TIMx_BKN 等信号进行保护控制（PWM 刹车）

23.3 功能描述

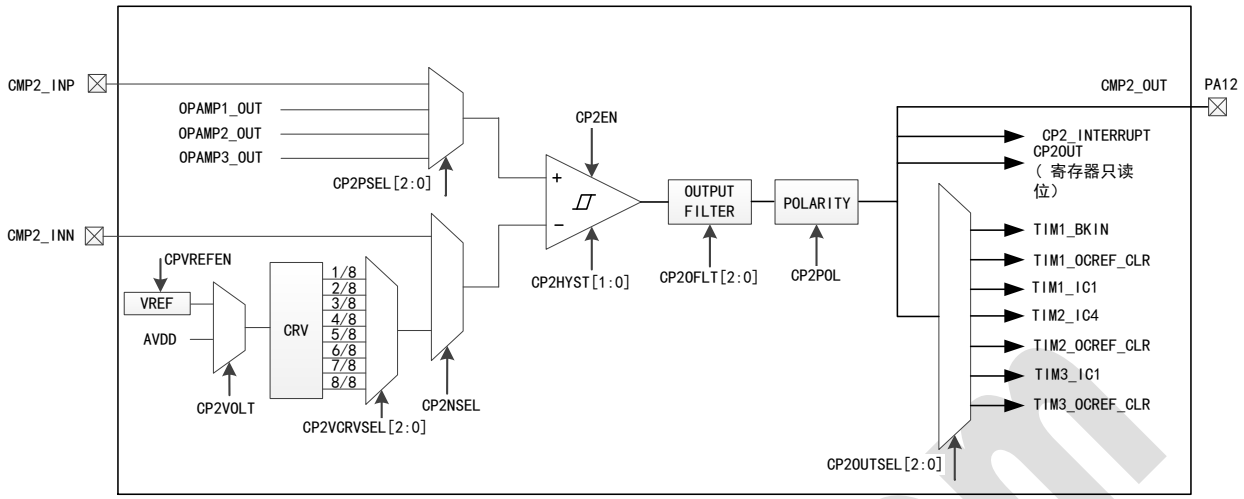
2 路比较器，以下为单个比较器的系统框图，CMP_INTERRUPT 由 ARM 外部中断单元统一控制，CMPx_OUT 引脚由 GPIO 复用功能单元统一控制。

注：在电机控制应用中，建议比较器 1 作为三相反电动势比较，比较器 2 作为过流保护比较。

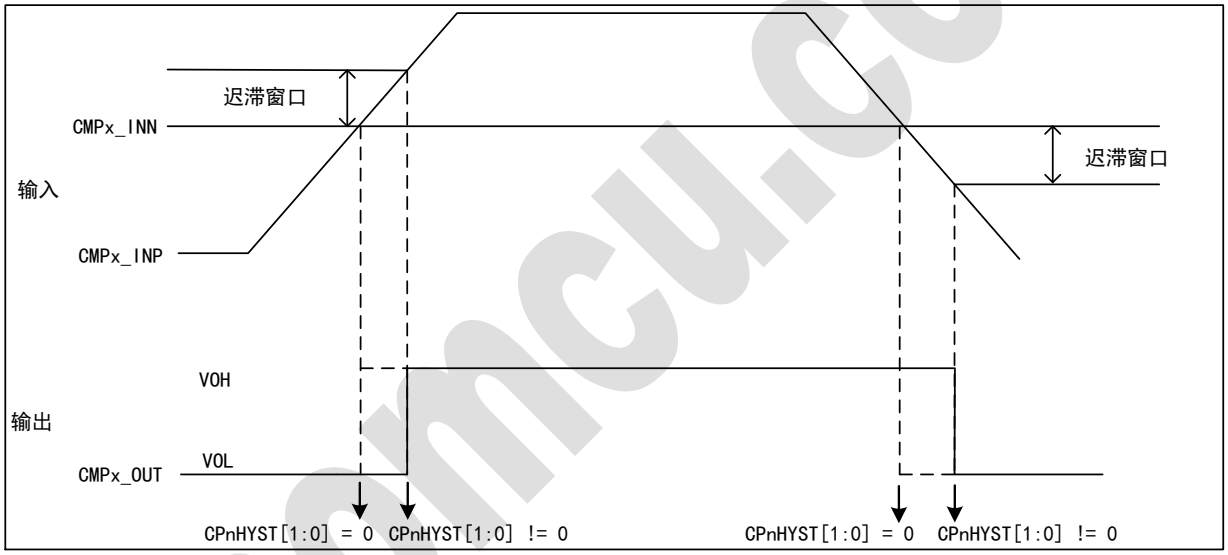
比较器 1 模块框图



比较器 2 模块框图



比较器施密特窗口及输出电压关系



23.3.1 管脚映射

实际管脚	比较器 1 对应模拟端口	复用类型
PA0	CMP1_INP1	模拟复用功能
PA1	CMP1_INP2	模拟复用功能
PA2	CMP1_INP3	模拟复用功能
PA3	CMP1_INN	模拟复用功能
PA11	CMP1_OUT	数字复用功能 AF7

实际管脚	比较器 2 对应模拟端口	复用类型
PB5	CMP2_INP	模拟复用功能
PB4	CMP2_INN	模拟复用功能
PA12	CMP2_OUT	数字复用功能 AF7

23.4 相关寄存器

23.4.1 寄存器汇总表

基址：0x4001 3C00				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**CMP_CP1CR**	0x0000 0180	比较器 1 控制寄存器	23.4.2

0x04	**CMP_CP2CR**	0x0000 0180	比较器 2 控制寄存器	23.4.3
0x08	**CMP_CPANA**	0x0000 0000	比较器模拟控制寄存器	23.4.4
0x0C	**CMP_CP1CAL**	0x0000 2020	比较器 1 校准寄存器	23.4.5
0x10	**CMP_CP2CAL**	0x0000 2020	比较器 2 校准寄存器	23.4.6

23.4.2 比较器 1 控制寄存器(CMP_CP1CR)

CMP_CP1CR			地址偏移	0x00		复位值	0x0000 0180	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	CP1LCK	CP1OUT	-	-	-	-	-	-
R/W	rw	r	-	-	-	-	-	-
复位值	0	0	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	CP1OUTSEL[2:0]		
R/W	-	-	-	-	-	rw	rw	rw
复位值	-	-	-	-	-	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CP1POL	-	CP1OFLT[2:0]			-	CP1VREFSEL[2:1]	
R/W	rw	-	rw	rw	rw	-	rw	rw
复位值	0	-	0	0	0	-	0	1
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CP1VREFSEL[0]	CP1NSEL	CP1PSEL[2:0]			CP1HYST[1:0]		CP1EN
R/W	rw	rw	rw	rw	rw	rw	rw	rw
复位值	1	0	0	0	0	0	0	0

BIT[31]	CP1LCK	比较器 1 寄存器锁定控制位 该位软件只能置“1”不能清“0”，只有系统复位后才能清“0”。当置“1”时令比较器 1 所有控制位变为只读。 0：比较器 1 寄存器的所有控制位维持自己的读写属性（默认） 1：比较器 1 寄存器的所有控制位为只读
BIT[30]	CP1OUT	比较器 1 数字输出值 该位只读，反映应比较器 1 输出状态。 0：输出低（默认） 1：输出高
BIT[18:16]	CP1OUTSEL[2:0]	比较器 1 内部输出功能选择位 这些位用来选择比较器 1 输出方向对应的功能。 000：定时器 1 的刹车输入（TIM1_BKIN）（默认） 001：定时器 1 OCREF CLEAR 输入（TIM1_OCREF_CLR） 010：定时器 1 输入捕获 1 通道（TIM1_IC1） 011：定时器 2 输入捕获 4 通道（TIM2_IC4） 100：定时器 2 OCREF CLEAR 输入（TIM2_OCREF_CLR） 101：定时器 3 输入捕获 1 通道（TIM3_IC1） 110：定时器 3 OCREF CLEAR 输入（TIM3_OCREF_CLR） 111：保留（客户不可配置成该值） *注：此位与“CP2CR”中的“CP2OUTSEL[2:0]”选择一致时，优先 CP1CR 的功能效果*
BIT[15]	CP1POL	**CP1POL** - 比较器 1 输出极性控制位 该位控制比较器 1 的输出极性。 0：保持输出（默认） 1：取反输出
BIT[13:11]	CP1OFLT	比较器 1 输出滤波周期选择位 这几位控制比较器 1 的输出滤波，达到连续的模块时钟周期后比较器输出不变则认为是有效的结果，否则保持不变，使用芯片的系统时钟。 000：1 时钟周期（Pclk），无滤波（默认） 001：4 时钟周期 010：16 时钟周期 011：32 时钟周期 100：64 时钟周期

		101：128 时钟周期 110：256 时钟周期 111：512 时钟周期
BIT[9:7]	CP1VREFSEL[2:0]	比较器 1 内部参考电压选择位 这几位控制比较器 1 参考电压的档位选择。 000：1/8*参考电压 CRV 001：2/8*参考电压 CRV 010：3/8*参考电压 CRV 011：4/8*参考电压 CRV（默认） 100：5/8*参考电压 CRV 101：6/8*参考电压 CRV 110：7/8*参考电压 CRV 111：8/8*参考电压 CRV
BIT[6]	CP1NSEL	比较器 1 反相输入选择位 该位用于选择连接到比较器 1 反相输入的信号源。 0：CP1N 比较器外部反相管脚（默认） 1：内部参考电压
BIT[5:3]	CP1PSEL[2:0]	比较器 1 正相输入选择位 这几位用于选择连接到比较器 1 正相输入的信号源。 000：CMP1_INP1 比较器外部正相管脚 1（默认） 001：CMP1_INP2 比较器外部正相管脚 2 010：CMP1_INP3 比较器外部正相管脚 3 011：OPAMP1_OUT 运算放大器 1 输出电压 100：OPAMP2_OUT 运算放大器 2 输出电压 101：OPAMP3_OUT 运算放大器 3 输出电压 110：保留（此位禁用） 111：保留（此位禁用）
BIT[2:1]	CP1HYST[1:0]	比较器 1 迟滞功能选择位 这几位控制比较器 1 的迟滞电压。 00：0mV（默认） 01：15mV 10：30mV 11：90mV
BIT[0]	CP1EN	比较器 1 模块工作使能位 该位控制比较器 1 模块是打开/关闭。 0：关闭比较器 1（默认） 1：打开比较器 1 *注：此位与“CP1CAL”中的“CP1CALEN”逻辑互斥，不能同时为“1”；即当“CP1CALEN”为“1”此时对“CP1EN”写“1”无效，当“CP1EN”为“1”此时对“CP1CALEN”写“1”无效*

23.4.3 比较器 2 控制寄存器(CMP_CP2CR)

CMP_CP2CR			地址偏移	0x04		复位值	0x0000 0180	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	CP2LCK	CP2OUT	-	-	-	-	-	-
R/W	rw	r	-	-	-	-	-	-
复位值	0	0	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	CP2OUTSEL[2:0]		
R/W	-	-	-	-	-	rw	rw	rw
复位值	-	-	-	-	-	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	CP2POL	-	CP2OFLT[2:0]			-	CP2VREFSEL[2:1]	
R/W	rw	-	rw	rw	rw	-	rw	rw
复位值	0	-	0	0	0	-	0	1
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	CP2VREFSEL[0]	CP2NSEL	-	CP2PSEL[1:0]		CP2HYST[1:0]		CP2EN

R/W	rw	rw	-	rw	rw	rw	rw	rw
复位值	1	0	-	0	0	0	0	0

BIT[31]	CP2LCK	比较器 2 寄存器锁定控制位 该位软件只能置“1”不能清“0”，只有系统复位后才能清“0”。当置“1”时令比较器 2 所有控制位变为只读。 0：比较器 2 寄存器的所有控制位维持自己的读写属性（默认） 1：比较器 2 寄存器的所有控制位为只读
BIT[30]	CP2OUT	比较器 2 数字输出值 该位只读，反映应比较器 2 输出状态。 0：输出低（默认） 1：输出高
BIT[18:16]	CP2OUTSEL[2:0]	比较器 2 内部输出功能选择位 这些位用来选择比较器 2 输出方向对应的功能。 000：定时器 1 的刹车输入（TIM1_BKIN）（默认） 001：定时器 1 OCREF CLEAR 输入（TIM1_OCREF_CLR） 010：定时器 1 输入捕获 1 通道（TIM1_IC1） 011：定时器 2 输入捕获 4 通道（TIM2_IC4） 100：定时器 2 OCREF CLEAR 输入（TIM2_OCREF_CLR） 101：定时器 3 输入捕获 1 通道（TIM3_IC1） 110：定时器 3 OCREF CLEAR 输入（TIM3_OCREF_CLR） 111：保留（客户不可配置成该值） *注：此位与“CP1CR”中的“CP1OUTSEL[2:0]”选择一致时，优先 CP1CR 的功能效果*
BIT[15]	CP2POL	**CP2POL** - 比较器 2 输出极性控制位 该位控制比较器 2 的输出极性。 0：保持输出（默认） 1：取反输出
BIT[13:11]	CP2OFLT	比较器 2 输出滤波周期选择位 这几位控制比较器 2 的输出滤波，达到连续的模块时钟周期后比较器输出不变则认为是有效的结果，否则保持不变，使用芯片的系统时钟。 000：1 时钟周期（Pclk），无滤波（默认） 001：4 时钟周期 010：16 时钟周期 011：32 时钟周期 100：64 时钟周期 101：128 时钟周期 110：256 时钟周期 111：512 时钟周期
BIT[9:7]	CP2VREFSEL[2:0]	比较器 2 内部参考电压选择位 这几位控制比较器 2 参考电压的档位选择。 000：1/8*参考电压 CRV 001：2/8*参考电压 CRV 010：3/8*参考电压 CRV 011：4/8*参考电压 CRV（默认） 100：5/8*参考电压 CRV 101：6/8*参考电压 CRV 110：7/8*参考电压 CRV 111：8/8*参考电压 CRV
BIT[6]	CP2NSEL	比较器 2 反相输入选择位 该位用于选择连接到比较器 2 反相输入的信号源。 0：CMP2_INN 比较器外部反相管脚（默认） 1：内部参考电压
BIT[4:3]	CP2PSEL[2:0]	比较器 2 正相输入选择位 这几位用于选择连接到比较器 2 正相输入的信号源。 00：CMP2_INP 比较器外部正相管脚（默认） 01：OPAMP1_OUT 运算放大器 1 输出电压 10：OPAMP2_OUT 运算放大器 2 输出电压 11：OPAMP3_OUT 运算放大器 3 输出电压

BIT[2:1]	CP2HYST[1:0]	比较器 2 迟滞功能选择位 这几位控制比较器 2 的迟滞电压。 00：0mV（默认） 01：15mV 10：30mV 11：90mV
BIT[0]	CP2EN	比较器 2 模块工作使能位 该位控制比较器 2 模块是打开/关闭。 0：关闭比较器 2（默认） 1：打开比较器 2 *注：此位与“CP2CAL”中的“CP2CALEN”逻辑互斥，不能同时为“1”；即当“CP2CALEN”为“1”此时对“CP2EN”写“1”无效，当“CP2EN”为“1”此时对“CP2CALEN”写“1”无效*

23.4.4 比较器模拟控制寄存器(CMP_CPANA)

CMP_CPANA			地址偏移	0x08		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	CP2VOLT	CP1VOLT	CPVREFEN
R/W	-	-	-	-	-	rw	rw	rw
复位值	-	-	-	-	-	0	0	0

BIT[2]	CP2VOLT	比较器 2 参考电压选择位 该位控制比较器 2 的参考电压的来源（VREF=2V，VDDA 为模拟电源上的输入电压）。 0：选择外部 AVDD 作为参考电压 CRV（默认） 1：选择内部 VREF 作为参考电压 CRV
BIT[1]	CP1VOLT	比较器 1 参考电压选择位 该位控制比较器 1 的参考电压的来源（VREF=2V，VDDA 为模拟电源上的输入电压）。 0：选择外部 AVDD 作为参考电压 CRV（默认） 1：选择内部 VREF 作为参考电压 CRV
BIT[0]	CPVREFEN	比较器内部参考电压使能位 该位控制比较器的内部参考电压是打开/关闭，打开参考电压后需要等待 100us 等待模拟信号建立稳定。 0：关闭内部参考电压 VREF（默认） 1：打开内部参考电压 VREF

23.4.5 比较器 1 校准寄存器(CMP_CP1CAL)

CMP_CP1CAL			地址偏移	0x0C		复位值	0x0000 2020	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	CP1CALEN	CP1SYNC	-	-	-	-	-	-
R/W	rw	r	-	-	-	-	-	-
复位值	0	0	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-

R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	CP1CALDATP					
R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	1	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	CP1CALDATN					
R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	1	0	0	0	0	0

BIT[31]	CP1CALEN	比较器 1 自动校准功能使能位 该位控制比较器 1 自动校准功能。由软件置“1”时自动启动校准，同时表示比较器 1 正在校准中；当完成时硬件自动清“0”表示校准完成并停止。 0：比较器 1 自动校准功能停止/完成（默认） 1：比较器 1 自动校准功能启动/进行中 *注 1：此位与“CPCR”中的“CPAMP1EN”逻辑互斥，不能同时为“1”；即当“CP1CALEN”为“1”此时对“CPAMP1EN”写“1”无效，当“CPAMP1EN”为“1”此时对“CP1CALEN”写“1”无效* *注 2：校准时需要打开 HSI 或是 HSE 的时钟，校准时间用时约 3ms（按照 8MHz HSI 时钟）*
BIT[30]	CP1SYNC	比较器 1 校准值同步状态标志位 该位表示比较器 1 校准值的同步标志。当对“CP1CALDATP”或“CP1CALDATN”写入数值后，由硬件自动置“1”时，此时表示比较器 1 正在将寄存器数值同步到内部电路中；当完成时硬件自动清“0”表示同步完成。 0：比较器 1 校准值完成 1：比较器 1 校准值正在同步中
BIT[13:8]	CP1CALDATP[5:0]	比较器 1 正端校验结果数据值 这几位用于显示自动校验结果，用户可根据数据内容进行微调。 **BIT5** 代表符号位： 0：表示增加电压 1：表示减少电压 **BIT[4:0]** 代表数据档位： 1：LSB 表示一格模拟设计电压档位（1LSB 约 0.5mv） *注：只有“CP1CALEN”为 0 时，可以对此区域的内容进行修改；当“CP1CALEN”为 1 时，此区域无法写入任何内容。*
BIT[5:0]	CP1CALDATN[5:0]	比较器 1 负端校验结果数据值 这几位用于显示自动校验结果，用户可根据数据内容进行微调。 **BIT5** 代表符号位： 0：表示增加电压 1：表示减少电压 **BIT[4:0]** 代表数据档位： 1：LSB 表示一格模拟设计电压档位（1LSB 约 0.5mv） *注：只有“CP1CALEN”为 0 时，可以对此区域的内容进行修改；当“CP1CALEN”为 1 时，此区域无法写入任何内容。*

23.4.6 比较器 2 校准寄存器(CMP_CP2CAL)

CMP_CP2CAL			地址偏移	0x10		复位值	0x0000 2020	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	CP2CALEN	CP2SYNC	-	-	-	-	-	-
R/W	rw	r	-	-	-	-	-	-
复位值	0	0	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	CP2CALDATP					

R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	1	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	CP2CALDATN					
R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	1	0	0	0	0	0

BIT[31]	CP2CALEN	比较器 2 自动校准功能使能位 该位控制比较器 2 自动校准功能。由软件置“1”时自动启动校准，同时表示比较器 2 正在校准中；当完成时硬件自动清“0”表示校准完成并停止。 0：比较器 2 自动校准功能停止/完成（默认） 1：比较器 2 自动校准功能启动/进行中 *注 1：此位与“CPCR”中的“CPAMP2EN”逻辑互斥，不能同时为“1”；即当“CP2CALEN”为“1”此时对“CPAMP2EN”写“1”无效，当“CPAMP2EN”为“1”此时对“CP2CALEN”写“1”无效* *注 2：校准时需要打开 HSI 或是 HSE 的时钟，校准时间用时约 3ms（按照 8MHz HSI 时钟）*
BIT[30]	CP2SYNC	比较器 2 校准值同步状态标志位 该位表示比较器 2 校准值的同步标志。当对“CP2CALDATP”或“CP2CALDATN”写入数值后，由硬件自动置“1”时，此时表示比较器 1 正在将寄存器数值同步到内部电路中；当完成时硬件自动清“0”表示同步完成。 0：比较器 2 校准值完成 1：比较器 2 校准值正在同步中
BIT[13:8]	CP2CALDATP[5:0]	比较器 2 正端校验结果数据值 这几位用于显示自动校验结果，用户可根据数据内容进行微调。 **BIT5** 代表符号位： 0：表示增加电压 1：表示减少电压 **BIT[4:0]** 代表数据档位： 1：LSB 表示一格模拟设计电压档位（1LSB 约 0.5mv） *注：只有“CP2CALEN”为 0 时，可以对此区域的内容进行修改；当“CP2CALEN”为 1 时，此区域无法写入任何内容。*
BIT[5:0]	CP2CALDATN[5:0]	比较器 2 负端校验结果数据值 这几位用于显示自动校验结果，用户可根据数据内容进行微调。 **BIT5** 代表符号位： 0：表示增加电压 1：表示减少电压 **BIT[4:0]** 代表数据档位： 1：LSB 表示一格模拟设计电压档位（1LSB 约 0.5mv） *注：只有“CP2CALEN”为 0 时，可以对此区域的内容进行修改；当“CP2CALEN”为 1 时，此区域无法写入任何内容。*

24 运算放大器（OPAMP）

24.1 概述

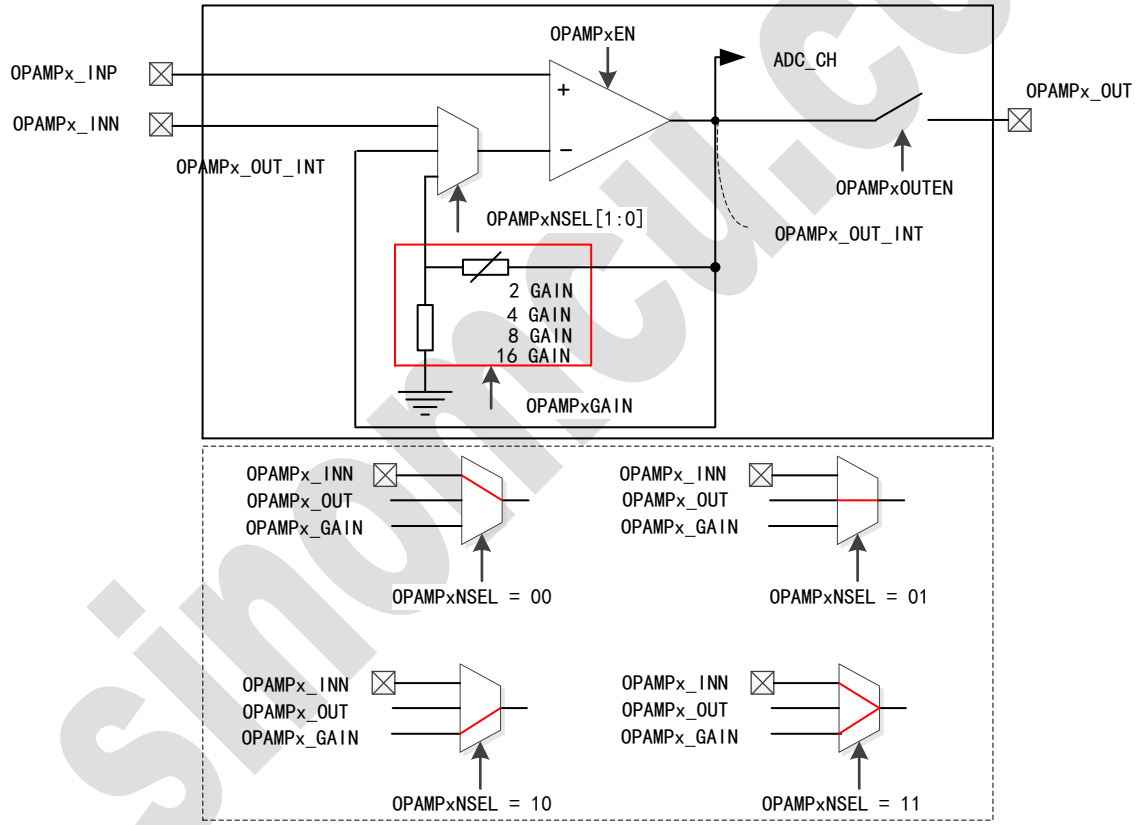
本产品内嵌 3 个高性能运算放大器（OPAMP1、2、3），每个运算放大器都有内部固定倍数放大功能，施密特窗口档位选择（2、4、8、16 倍），反相端与输出口可以灵活选择接线方式，配合外部电路达到一些应用效果。输出信号可以接入 ADC 端口进行数据采样，也可以接入比较正端作为比较源头，同时也能输出到芯片管脚作为其它使用。

24.2 特性

- ✧ 内嵌 3 路运算放大器
- ✧ 支持内部固定放大倍数选择
- ✧ 支持施密特窗口档位选择
- ✧ 反相输入通道灵活选择
- ✧ 输出可接入 ADC 输入通道和比较器正相输入通道

24.3 功能描述

运算放大器模块框图



注：OPAMPxNSEL=11B 模式保留，不推荐用户使用。

24.3.1 引脚映射

实际管脚	运算放大器 1 对应模拟端口	复用类型
PB5	OPAMP1_INP	模拟复用功能
PB6	OPAMP1_INN	模拟复用功能
PB7	OPAMP1_OUT	模拟复用功能

实际管脚	运算放大器 2 对应模拟端口	复用类型
PA4	OPAMP2_INP	模拟复用功能
PA5	OPAMP2_INN	模拟复用功能
PA6	OPAMP2_OUT	模拟复用功能

实际管脚	运算放大器 3 对应模拟端口	复用类型
PC15	OPAMP3_INP	模拟复用功能
PC14	OPAMP3_INN	模拟复用功能
PC13	OPAMP3_OUT	模拟复用功能

24.3.2 与 ADC 模块连接

运算放大器的输出信号需要内部对接 ADC 通道进行采样

ADC 内部输入通道 19~21，分别对应 OPAMP1~3，关系如下：

- ADC channel 19 -> OPAMP1 内部模拟输出
- ADC channel 20 -> OPAMP2 内部模拟输出
- ADC channel 21 -> OPAMP3 内部模拟输出

通过 ADC_CFGR1 的控制位“AWDCH[4:0]”选择对应的 OPAMPx 通道：

- 10011：由模拟看门狗监控的 ADC 模拟输入通道 19
- 10100：由模拟看门狗监控的 ADC 模拟输入通道 20
- 10101：由模拟看门狗监控的 ADC 模拟输入通道 21

ADC_CHSELR 的 CHSELx（x=19~21）位：用于选择转换序列中 OPAMPx 对应通道是否为转换通道。

- bit19：CHSEL19
- bit20：CHSEL20
- bit21：CHSEL21

详细 ADC 配置请参见 ADC 章节。

24.4 相关寄存器

24.4.1 寄存器汇总表

基址：0x4001 3C00				
偏移地址	寄存器名	复位值	功能描述	章节
0x14	**OPA_OPCR**	0x0000 0000	运算放大器控制寄存器	24.4.2
0x18	**OPA_OP1CAL**	0x0000 2020	运算放大器 1 校准寄存器	24.4.3
0x1C	**OPA_OP2CAL**	0x0000 2020	运算放大器 2 校准寄存器	24.4.4
0x20	**OPA_OP3CAL**	0x0000 2020	运算放大器 3 校准寄存器	24.4.5

24.4.2 运算放大器控制寄存器(OPA_OPCR)

OPA_OPCR			地址偏移	0x14		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	OPAMPLCK	-	-	-	-	-	-	-
R/W	rw	-	-	-	-	-	-	-
复位值	0	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	OPAMP3OUTEN	-	OPAMP3GAIN[1:0]		-	OPAMP3NSEL[1:0]		OPAMP3EN
R/W	rw	-	rw	rw	-	rw	rw	rw
复位值	0	-	0	0	-	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	OPAMP2OUTEN	-	OPAMP2GAIN[1:0]		-	OPAMP2NSEL[1:0]		OPAMP2EN
R/W	rw	-	rw	rw	-	rw	rw	-
复位值	0	-	0	0	-	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	OPAMP1OUTEN	-	OPAMP1GAIN[1:0]		-	OPAMP1NSEL[1:0]		OPAMP1EN
R/W	rw	-	rw	rw	-	rw	rw	rw
复位值	0	-	0	0	-	0	0	0

BIT[31]	OPAMPLOCK	运算放大器寄存器锁定控制位

		该位软件只能置“1”不能清“0”，只有系统复位后才能清“0”。当置“1”时令所有运算放大器控制位变为只读。 0：所有运算放大器寄存器的控制位维持自己的读写属性（默认） 1：所有运算放大器寄存器的控制位为只读
BIT[23]	OPAMP3OUTEN	运算放大器 3 输出端口使能位 该位控制运算放大器 3 的端口输出是否有效。 0：运算放大器 3 的输出端口不连接到 GPIO 1：运算放大器 3 的输出端口连接到 GPIO *注 1：运算放大器的输出端口是否连接到 GPIO，不仅使能该位，同时对应 GPIO 还要选择为“模拟功能”* *注 2：使用者请注意，如果对应其它模拟功能也打开（如：ADC 通道），该管脚的 GPIO 会同时具备运放输出功能和 ADC 采样功能*
BIT[21:20]	OPAMP3GAIN[1:0]	运算放大器 3 增益倍数选择位 这几位控制运算放大器 3 的增益倍数。 00：2 GAIN（默认） 01：4 GAIN 10：8 GAIN 11：16 GAIN
BIT[18:17]	OPAMP3NSEL[1:0]	运算放大器 3 反相输入选择位 这几位用于选择连接到运算放大器 3 反相输入的信号源。 00：OPAMP3_INN 运算放大器 3 反相输入（默认） 01：OPAMP3_OUT 运算放大器 3 输出 10：OPAMP3 的 GAIN 增益倍数 11：保留
BIT[16]	OPAMP3EN	运算放大器 3 模块工作使能位 该位控制运算放大器 3 模块是打开/关闭。 0：关闭运算放大器 3（默认） 1：打开运算放大器 3 *注：此位与“OP3CAL”中的“OP3CALEN”逻辑互斥，不能同时为“1”；即当“OP3CALEN”为“1”此时对“OPAMP3EN”写“1”无效，当“OPAMP3EN”为“1”此时对“OP3CALEN”写“1”无效*
BIT[15]	OPAMP2OUTEN	运算放大器 2 输出端口使能位 该位控制运算放大器 2 的端口输出是否有效。 0：运算放大器 2 的输出端口不连接到 GPIO 1：运算放大器 2 的输出端口连接到 GPIO *注 1：运算放大器的输出端口是否连接到 GPIO，不仅使能该位，同时对应 GPIO 还要选择为“模拟功能”* *注 2：使用者请注意，如果对应其它模拟功能也打开（如：ADC 通道），该管脚的 GPIO 会同时具备运放输出功能和 ADC 采样功能*
BIT[13:12]	OPAMP2GAIN[1:0]	运算放大器 2 增益倍数选择位 这几位控制运算放大器 2 的增益倍数。 00：2 GAIN（默认） 01：4 GAIN 10：8 GAIN 11：16 GAIN
BIT[10:9]	OPAMP2NSEL[1:0]	运算放大器 2 反相输入选择位 这几位用于选择连接到运算放大器 2 正相输入的信号源。 00：OPAMP2_INN 运算放大器 2 反相输入（默认） 01：OPAMP2_OUT 运算放大器 2 输出 10：OPAMP2 的 GAIN 增益倍数 11：OPAMP2_INN 运算放大器 2 反相输入与 GAIN 增益倍数短接再输入
BIT[8]	OPAMP2EN	运算放大器 2 模块工作使能位 该位控制运算放大器 2 模块是打开/关闭。 0：关闭运算放大器 2（默认） 1：打开运算放大器 2 *注：此位与“OP2CAL”中的“OP2CALEN”逻辑互斥，不能同时为“1”；即当“OP2CALEN”为“1”此时对“OPAMP2EN”写“1”无效，当“OPAMP2EN”为“1”此时对“OP2CALEN”写“1”无效*
BIT[7]	OPAMP1OUTEN	运算放大器 1 输出端口使能位

		该位控制运算放大器 1 的端口输出是否有效。 0：运算放大器 1 的输出端口不连接到 GPIO 1：运算放大器 1 的输出端口连接到 GPIO *注 1：运算放大器的输出端口是否连接到 GPIO，不仅使能该位，同时对应 GPIO 还要选择为“模拟功能”* *注 2：使用者请注意，如果对应其它模拟功能也打开（如：ADC 通道），该管脚的 GPIO 会同时具备运放输出功能和 ADC 采样功能*
BIT[5:4]	OPAMP1GAIN[1:0]	运算放大器 1 增益倍数选择位 这几位控制运算放大器 1 的增益倍数。 00：2 GAIN（默认） 01：4 GAIN 10：8 GAIN 11：16 GAIN
BIT[2:1]	OPAMP1NSEL[1:0]	运算放大器 1 反相输入选择位 这几位用于选择连接到运算放大器 1 正相输入的信号源。 00：OPAMP1_INN 运算放大器 1 反相输入（默认） 01：OPAMP1_OUT 运算放大器 1 输出 10：OPAMP1 的 GAIN 增益倍数 11：OPAMP1_INN 运算放大器 1 反相输入与 GAIN 增益倍数短接再输入
BIT[0]	OPAMP1EN	运算放大器 1 模块工作使能位 该位控制运算放大器 1 模块是打开/关闭。 0：关闭运算放大器 1（默认） 1：打开运算放大器 1 *注：此位与“OP1CAL”中的“OP1CALEN”逻辑互斥，不能同时为“1”；即当“OP1CALEN”为“1”此时对“OPAMP1EN”写“1”无效，当“OPAMP1EN”为“1”此时对“OP1CALEN”写“1”无效*

24.4.3 运算放大器 1 校准寄存器(OPA_OP1CAL)

OPA_OP1CAL			地址偏移	0x18		复位值	0x0000 2020	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	OP1CALEN	OP1SYNC	-	-	-	-	-	-
R/W	rw	r	-	-	-	-	-	-
复位值	0	0	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	OP1CALDATP					
R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	1	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	OP1CALDATN					
R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	1	0	0	0	0	0

注：该寄存器的读写属性会受“OPCR”的“OPAMPLCK”置“1”的影响变成只读属性

BIT[31]	OP1CALEN	运算放大器 1 自动校准功能使能位 该位控制运算放大器 1 自动校准功能。由软件置“1”时自动启动校准，同时表示运算放大器 1 正在校准中；当完成时硬件自动清“0”表示校准完成并停止。 0：运算放大器 1 自动校准功能停止/完成（默认） 1：运算放大器 1 自动校准功能启动/进行中 *注 1：此位与“OPCR”中的“OPAMP1EN”逻辑互斥，不能同时为“1”；即当“OP1CALEN”为“1”此时对“OPAMP1EN”写“1”无效，当“OPAMP1EN”为“1”此时对“OP1CALEN”写“1”无效* *注 2：校准时需要打开 HSI 或是 HSE 的时钟，校准时间用时约 3ms（按照 8MHz HSI 时钟）*

BIT[30]	OP1SYNC	运算放大器 1 校准值同步状态标志位 该位表示运算放大器 1 校准值的同步标志。当对“OP1CALDATP”或“OP1CALDATN”写入数值后，由硬件自动置“1”时，此时表示运算放大器 1 正在将寄存器数值同步到内部电路中；当完成时硬件自动清“0”表示同步完成。 0：运算放大器 1 校准值完成 1：运算放大器 1 校准值正在同步中
BIT[13:8]	OP1CALDATP[5:0]	运算放大器 1 正端校验结果数据值 这几位用于显示自动校验结果，用户可根据数据内容进行微调。 **BIT5** 代表符号位： 0：表示增加电压 1：表示减少电压 **BIT[4:0]** 代表数据档位： 1：LSB 表示一格模拟设计电压档位 *注：只有“OP1CALEN”为 0 时，可以对此区域的内容进行修改；当“OP1CALEN”为 1 时，此区域无法写入任何内容。*
BIT[5:0]	OP1CALDATN[5:0]	运算放大器 1 负端校验结果数据值 这几位用于显示自动校验结果，用户可根据数据内容进行微调。 **BIT5** 代表符号位： 0：表示增加电压 1：表示减少电压 **BIT[4:0]** 代表数据档位： 1：LSB 表示一格模拟设计电压档位 *注：只有“OP1CALEN”为 0 时，可以对此区域的内容进行修改；当“OP1CALEN”为 1 时，此区域无法写入任何内容。*

24.4.4 运算放大器 2 校准寄存器(OPA_OP2CAL)

OPA_OP2CAL			地址偏移	0x1C		复位值	0x0000 2020	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	OP2CALEN	OP2SYNC	-	-	-	-	-	-
R/W	rw	r	-	-	-	-	-	-
复位值	0	0	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	OP2CALDATP					
R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	1	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	OP2CALDATN					
R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	1	0	0	0	0	0

BIT[31]	OP2CALEN	运算放大器 2 自动校准功能使能位 该位控制运算放大器 2 自动校准功能。由软件置“1”时自动启动校准，同时表示运算放大器 2 正在校准中；当完成时硬件自动清“0”表示校准完成并停止。 0：运算放大器 2 自动校准功能停止/完成（默认） 1：运算放大器 2 自动校准功能启动/进行中 *注 1：此位与“OPCR”中的“OPAMP2EN”逻辑互斥，不能同时为“1”；即当“OP2CALEN”为“1”此时对“OPAMP2EN”写“1”无效，当“OPAMP2EN”为“1”此时对“OP2CALEN”写“1”无效* *注 2:校准时需要打开 HSI 或是 HSE 的时钟,校准时间用时约 3ms(按照 8MHz HSI 时钟)*
BIT[30]	OP2SYNC	运算放大器 2 校准值同步状态标志位 该位表示运算放大器 2 校准值的同步标志。当对“OP2CALDATP”或“OP2CALDATN”写入数值后，由硬件自动置“1”时，此时表示运算放大器 1 正在将寄存器数值同步到内部电路中；当完成时硬件自动清“0”表示同步完成。

		0：运算放大器 2 校准值完成 1：运算放大器 2 校准值正在同步中
BIT[13:8]	OP2CALDATP[5:0]	运算放大器 2 正端校验结果数据值 这几位用于显示自动校验结果，用户可根据数据内容进行微调。 **BIT5** 代表符号位： 0：表示增加电压 1：表示减少电压 **BIT[4:0]** 代表数据档位： 1：LSB 表示一格模拟设计电压档位 *注：只有“OP2CALEN”为 0 时，可以对此区域的内容进行修改；当“OP2CALEN”为 1 时，此区域无法写入任何内容。*
BIT[5:0]	OP2CALDATN[5:0]	运算放大器 2 负端校验结果数据值 这几位用于显示自动校验结果，用户可根据数据内容进行微调。 **BIT5** 代表符号位： 0：表示增加电压 1：表示减少电压 **BIT[4:0]** 代表数据档位： 1：LSB 表示一格模拟设计电压档位 *注：只有“OP2CALEN”为 0 时，可以对此区域的内容进行修改；当“OP2CALEN”为 1 时，此区域无法写入任何内容。*

24.4.5 运算放大器 3 校准寄存器（OPA_OP3CAL）

OPA_OP3CAL			地址偏移	0x20		复位值	0x0000 2020	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	OP3CALEN	OP3SYNC	-	-	-	-	-	-
R/W	rw	r	-	-	-	-	-	-
复位值	0	0	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	OP3CALDATP					
R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	1	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	OP3CALDATN					
R/W	-	-	rw	rw	rw	rw	rw	rw
复位值	-	-	1	0	0	0	0	0

注：该寄存器的读写属性会受“OPCR”的“OPAMPLCK”置“1”的影响变成只读属性

BIT[31]	OP3CALEN	运算放大器 3 自动校准功能使能位 该位控制运算放大器 3 自动校准功能。由软件置“1”时自动启动校准，同时表示运算放大器 3 正在校准中；当完成时硬件自动清“0”表示校准完成并停止。 0：运算放大器 3 自动校准功能停止/完成（默认） 1：运算放大器 3 自动校准功能启动/进行中 *注 1：此位与“OPCR”中的“OPAMP3EN”逻辑互斥，不能同时为“1”；即当“OP3CALEN”为“1”此时对“OPAMP3EN”写“1”无效，当“OPAMP3EN”为“1”此时对“OP3CALEN”写“1”无效* *注 2：校准时需要打开 HSI 或是 HSE 的时钟，校准时间用时约 3ms（按照 8MHz HSI 时钟）*
BIT[30]	OP3SYNC	运算放大器 3 校准值同步状态标志位 该位表示运算放大器 3 校准值的同步标志。当对“OP3CALDATP”或“OP3CALDATN”写入数值后，由硬件自动置“1”时，此时表示运算放大器 1 正在将寄存器数值同步到内部电路中；当完成时硬件自动清“0”表示同步完成。 0：运算放大器 3 校准值完成 1：运算放大器 3 校准值正在同步中

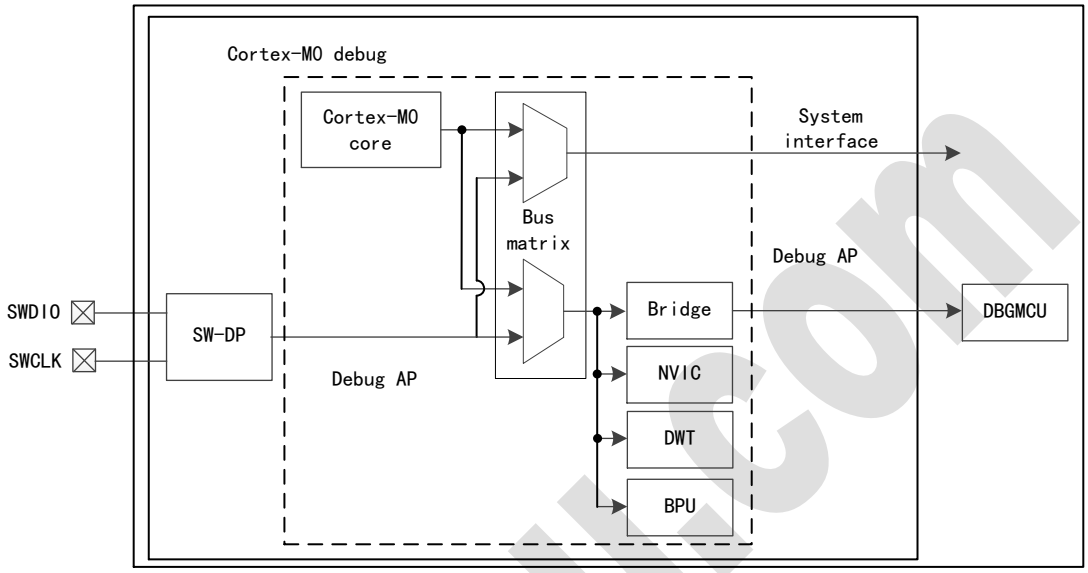
BIT[13:8]	OP3CALDATP[5:0]	运算放大器 3 正端校验结果数据值 这几位用于显示自动校验结果，用户可根据数据内容进行微调。 **BIT5** 代表符号位： 0：表示增加电压 1：表示减少电压 **BIT[4:0]** 代表数据档位： 1：LSB 表示一格模拟设计电压档位 *注：只有“OP3CALEN”为 0 时，可以对此区域的内容进行修改；当“OP3CALEN”为 1 时，此区域无法写入任何内容。*
BIT[5:0]	OP3CALDATN[5:0]	运算放大器 3 负端校验结果数据值 这几位用于显示自动校验结果，用户可根据数据内容进行微调。 **BIT5** 代表符号位： 0：表示增加电压 1：表示减少电压 **BIT[4:0]** 代表数据档位： 1：LSB 表示一格模拟设计电压档位 *注：只有“OP3CALEN”为 0 时，可以对此区域的内容进行修改；当“OP3CALEN”为 1 时，此区域无法写入任何内容。*

25 调试支持（DBG）

25.1 概述

本器件基于 Cortex-M0 内核，内嵌高级调试功能。支持内核在取值（breakpoint）和访问数据（watchpoint）时停止。内核停止时，可查询内核状态和系统外部状态。完成查询，内核和系统可恢复，程序继续执行。

DEBUG 框图



SW-DP：串行调试接口

BPU：断点单元

DWT：数据观察点触发

注：有关 ARM Cortex®-M0 内核调试功能的更多信息，请参阅 the Cortex®-M0 Technical Reference Manual。

相关文档参考：

<Cortex®-M0 Technical Reference Manual (TRM)>

<ARM Debug Interface V5>

<ARM CoreSight Design Kit revision r1p1 Technical Reference Manual>

25.2 引脚分布和调试引脚

SW debug 端口

SW-DP 引脚名	I/O 类型	说明	引脚分配
SWDIO	I/O	串行数据输入/输出	PA13
SWCLK	I	串行时钟	PA14

复位后（SYSRESETn 或 PORRESETn），SW-DP 端口被指定为 SW 功能，调试器主机可以立即使用这些引脚。

另外，MCU 也提供了禁用 SWD 端口的操作，这样可以释放相关引脚为通用 IO（GPIO）。详细禁用操作信息，请参考<7.3.2 I/O 复用功能映射>。

SWD 引脚内部上拉/下拉

保证 SWD 输入引脚非浮空状态，内嵌上拉/下拉：

SWDIO：内部上拉

SWCLK：内部下拉

一旦 SWD 引脚被用户代码释放，GPIO 控制器将取得控制权，IO 口恢复到复位态：

SWDIO：输入内部上拉

SWCLK：输入内部下拉

注：内嵌电阻可以消除片外增加电阻的需求。

25.3 ID 码和锁定机制

芯片包含多个 ID 码，用户可通过 debug 接口或应用程序访问。

25.3.1 寄存器汇总表

MCU 内置 MCU CHIPID 寄存器，包含芯片代号和版本号信息。

基址：0x4001 0000				
偏移地址	寄存器名	复位值	功能描述	章节
0xFC	**CHIPID**	0xA80X0000	CHIPID	
基址：0x4001 5800				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**DBGMCU_IDCODE**		MCU 设备 ID CODE	

25.3.2 CHIPID 寄存器

CHIPID			地址偏移	0x00		复位值	0xA80X0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	DEV_ID[11:4]							
R/W	r	r	r	r	r	r	r	r
复位值	1	0	1	0	1	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	DEV_ID[3:0]				REV_ID[3:0]			
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	x	x	x	x
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	res.							
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	res.				res.			
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0

BIT[31:20]	DEV_ID[11:0]	0xA80
BIT[19:16]	REV_ID[3:0]	包含版本信息
BIT[15:0]	-	保留，保持复位值

25.3.3 DBGMCU_IDCODE 寄存器

DBGMCU_IDCODE			地址偏移	0x00		复位值	0xXXXX XXXX	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	REV_ID[31:24]							
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	1	0	0	0	0
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	REV_ID[23:16]							
R/W	r	r	r	r	r	r	r	r
复位值	0	0	0	0	0	0	0	0
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	Res.	Res.	Res.	Res.	DEV_ID[11:8]			
R/W	r	r	r	r	r	r	r	r
复位值	0	1	1	0	0	1	0	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	DEV_ID[7:0]							
R/W	r	r	r	r	r	r	r	r
复位值	0	1	0	0	0	1	0	0

BIT[31:16]	REV_ID[31:16]	0x1000，只读
BIT[15:12]	-	保留，保持复位值
BIT[11:0]	REV_ID[11:0]	0x444，只读

25.3.4 Cortex SW IDCODE

CPU 有一个 SW IDCODE，通过 SW-DP 访问，包含 SW-DP 的 partnumber 和 JEDEC-106 ID 标识码。在 SW 通讯连接时作为 connect 连接判断及 JTAG/SW 区分。

25.3.5 CPUID

CPU 内置一个 CPUID 寄存器，用于确定 CPU 的类型和版本。

25.3.6 ID 编码汇总

ID 名称	地址	值
CHIPID	0x4001 00FC	0x0000A80x（x=0,1,2…）
DBGMCU_IDCODE	0x4001 5800	0x10006444
CPUID	0xE000 ED00	0x410CC200 (r0p0)
SW IDCODE	DPnAP=1，地址 0b00，read	0x0BB11477

注:CPUID 和 SW IDCODE 是 cortex-m0 内核设计包含的,详情请参考<Cortex®-M0 Technical Reference Manual (TRM)>。

25.4 SWD 接口

详情请参考<Cortex®-M0 Technical Reference Manual (TRM)>。

25.5 MCU 调试模块（MCUDBG）

MCU 调试模块协助调试器提供以下功能：

- ✧ 低功耗模式
- ✧ 在断点时提供定时器，看门狗和 I2C 的时钟控制

25.5.1 低功耗模式的调试支持

通过使用 WFI 和 WFE 可以进入低功耗模式。

MCU 支持多种低功耗模式，分别可以关闭 CPU 时钟，或降低 CPU 的能耗。

内核不允许在调试期间关闭 FCLK 或 HCLK。这些时钟对于调试操作是必要的，因此在调试期间，它们必须工作。MCU 使用一种特殊的方式，允许用户在低功耗模式下调试代码。

为实现这一功能，调试器必须先设置一些配置寄存器来改变低功耗模式的行为：

- 在 sleep 模式下，FCLK 和 HCLK 保持工作状态，因此，这种模式不会对标准的调试特性施加任何限制。
- 在 stop/standby 模式下，调试器必须先置位 DBG_STOP 位。这将在 stop/standby 模式下激活内部 RC 振荡器，为 FCLK 和 HCLK 提供时钟。

25.5.2 Timer、看门狗、I2C 的调试支持

在产生断点时，有必要根据定时器和看门狗的不同用途选择计数器的工作模式：

- 在产生断点时，计数器继续计数。这在输出 PWM 控制电机时常常要用到。
- 在产生断点时，计数器停止计数。这对于看门狗的计数器是必需的。

对于 I2C，用户可以选择在断点期间阻止 SMBUS 超时。

25.5.3 调试寄存器汇总表

基址：0x4001 5800				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**DBGMCU_IDCODE**	0x10006444	DBGMCU IDCODE 寄存器	25.3.3
0x04	**DBGMCU_CR**	0x0000 0000	DBGMCU 控制寄存器	25.5.4
0x08	**DBGMCU_APB1_FZ**	0x0000 0000	DBGMCU APB1 冻结寄存器	25.5.5
0x0C	**DBGMCU_APB2_FZ**	0x0000 0000	DBGMCU APB2 冻结寄存器	25.5.6

25.5.4 DBGMCU 控制寄存器（DBGMCU_CR）

此寄存器 PORREST 异步复位（不支持系统复位），系统复位状态下，调试器可以写该寄存器。

若调试器主机不支持这些特性，还可以通过用户软件写该寄存器。

仅支持 32-bit 访问

DBGMCU_CR			地址偏移	0x04		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-

R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	DBG_STANDBY	DBG_STOP	-
R/W	-	-	-	-	-	rw	rw	-
复位值	-	-	-	-	-	0	0	-

BIT[31:3]	-	保留，保持复位值
BIT[2]	DBG_STANDBY	调试 standby 模式（Debug standby mode） 0:（FCLK 关,HCLK 关）整个数字电路部分都断电。从软件的角度看，退出STANDBY模式与复位是一样的（除了一些状态位指示了微控制器刚从STANDBY 状态退出） 1：（FCLK 开，HCLK 开）数字电路部分不下电，RC振荡器保持开启，FCLK 和HCLK 时钟由内部RC振荡器提供时钟。另外，微控制器通过产生系统复位来退出STANDBY模式，和复位是一样的。
BIT[1]	DBG_STOP	调试停机模式（Debug Stop mode） 0：（FCLK 关，HCLK 关）在停机模式时，时钟控制器禁止一切时钟（包括HCLK和FCLK）。当从STOP模式退出时，时钟的配置和复位之后的配置一样（微控制器由8MHz的内部振荡器（HSI）提供时钟）。因此，软件必须重新配置时钟控制系统启动PLL，晶振等。 1：（FCLK 开，HCLK 开）在停机模式时，RC振荡器保持开启，FCLK 和HCLK时钟由内部RC振荡器提供。退出STOP模式，软件必须重新配置时钟控制系统启动PLL，晶振等（与DBG_STOP=0相同）。
BIT[0]	-	保留，保持复位值

25.5.5 DBGMCU_APB1 冻结寄存器

此寄存器 PORREST 异步复位（不支持系统复位），系统复位状态下，调试器可以写该寄存器。

仅支持 32-bit 访问

DBGMCU_APB1			地址偏移	0x08		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	DBG_I2C1_SMBUS_TIMEOUT	-	-	-	-	-
R/W	-	-	rw	-	-	-	-	-
复位值	-	-	0	-	-	-	-	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-	DBG_IWDG_STOP	DBG_WWDG_STOP	DBG_RTC_STOP	_	DBG_TIM14_STOP
R/W	-	-	-	rw	rw	rw	-	rw
复位值	-	-	-	0	0	0	-	0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

控制位	-	-	-	-	-	-	DBG_TIM3_STOP	DBG_TIM2_STOP
R/W	-	-	-	-	-	-	rw	rw
复位值	-	-	-	-	-	-	0	0

BIT[31:22]	-	保留，保持复位值
BIT[21]	DBG_I2C1_SMBUS_TIMEOUT	调试模式下 CPU 停止，SMBUS 超时模式停止（SMBUS timeout mode stopped when core is halted） 0：同 normal 模式 1：SMBUS 超时被冻结
BIT[20:13]	-	保留，保持复位值
BIT[12]	DBG_IWDG_STOP	调试模式下 CPU 停止，独立看门狗停止（Debug IWDG stopped when core is halted） 0：内核停止，独立看门狗继续工作 1：内核停止，独立看门狗停止
BIT[11]	DBG_WWDG_STOP	调试模式下 CPU 停止，窗口看门狗停止（Debug WWDG stopped when core is halted） 0：内核停止，窗口看门狗继续工作 1：内核停止，窗口看门狗停止
BIT[10]	DBG_RTC_STOP	调试模式下 CPU 停止,RTC 计数器停止(Debug RTC stopped when core is halted） 0：内核停止，RTC 计数器继续工作 1：内核停止，RTC 计数器停止
BIT[9]	-	保留，保持复位值
BIT[8]	DBG_TIM14_STOP	调试模式下 CPU 停止，TIM14 计数器停止（Debug TIM14 stopped when core is halted） 0：内核停止，TIM14 计数器继续工作 1：内核停止，TIM14 计数器停止
BIT[7:2]	-	保留，保持复位值
BIT[1]	DBG_TIM3_STOP	调试模式下 CPU 停止，TIM3 计数器停止（Debug TIM3 stopped when core is halted） 0：内核停止，TIM3 计数器继续工作 1：内核停止，TIM3 计数器停止
BIT[0]	DBG_TIM2_STOP	调试模式下 CPU 停止，TIM2 计数器停止（Debug TIM2 stopped when core is halted） 0：内核停止，TIM2 计数器继续工作 1：内核停止，TIM2 计数器停止

25.5.6 DBGMCU_APB2 冻结寄存器

此寄存器 PORREST 异步复位（不支持系统复位），系统复位状态下，调试器可以写该寄存器。

仅支持 32-bit 访问

DBGMCU_APB2			地址偏移	0x0C		复位值	0x0000 0000	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	-	-	-	-	-	DBG_TIM17_STOP	DBG_TIM16_STOP	-
R/W	-	-	-	-	-	rw	rw	-
复位值	-	-	-	-	-	0	0	-
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	-	-	-		DBG_TIM1_STOP	-	-	-
R/W	-	-	-		rw	-	-	-
复位值	-	-	-		0	-	-	-
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	-	-	-	-	-	-	-	-

R/W	-	-	-	-	-	-	-	-
复位值	-	-	-	-	-	-	-	-

BIT[31:19]	-	保留，保持复位值
BIT[18]	DBG_TIM17_STOP	调试模式下 CPU 停止，TIM17 计数器停止（Debug TIM17 stopped when core is halted） 0：内核停止，TIM17 计数器继续工作 1：内核停止，TIM17 计数器停止
BIT[17]	DBG_TIM16_STOP	调试模式下 CPU 停止，TIM16 计数器停止（Debug TIM16 stopped when core is halted） 0：内核停止，TIM16 计数器继续工作 1：内核停止，TIM16 计数器停止
BIT[16:12]	-	保留，保持复位值
BIT[11]	DBG_TIM1_STOP	调试模式下 CPU 停止，TIM1 计数器停止（Debug TIM1 stopped when core is halted） 0：内核停止，TIM1 计数器继续工作 1：内核停止，TIM1 计数器停止
BIT[10:0]	-	保留，保持复位值

26 器件电子签名（UID）

设备的电子签名存储在闪存模块的系统内存区，可以通过调试接口读取，也可以通过 CPU 读取。它包含工厂编程的识别和校准数据，允许用户固件或其他外部设备读取，用以自动匹配微控制器的特性。

26.1 UID 器件唯一码

UID 用于：

- 用作序列号（举例，USB 字符串序列号或其他终端应用）
- 用作安全密钥，使用此唯一码结合软件加密算法，可以提高代码安全性
- 用作激活自举过程的安全检测机制。

UID 为 96 位的唯一硬件标识符，由芯片出厂时确定。用户不可改写。

26.1.1 UID 寄存器汇总表

基址：0x1FFF F7AC				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**UID_DR1**	0xXXXX XXXX	UID 寄存器 1	26.1.2
0x04	**UID_DR2**	0xXXXX XXXX	UID 寄存器 2	26.1.2
0x08	**UID_DR3**	0xXXXX XXXX	UID 寄存器 3	26.1.2

26.1.2 UID 寄存器 1（UID_DR1）

UID_DR1			地址偏移	0x00		复位值	0xXXXX XXXX	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	UID[31:24]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	UID[23:16]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	UID[15:8]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	UID[7:0]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x

BIT[31:0]	UID[31:0]	X and Y coordinates on the wafer(BCD 码格式)

26.1.3 UID 寄存器 2（UID_DR2）

UID_DR2			地址偏移	0x04		复位值	0xXXXX XXXX	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	UID[63:56]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	UID[55:48]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	UID[47:40]							
R/W	r	r	r	r	r	r	r	r

复位值	x	x	x	x	x	x	x	x
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	UID[39:32]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x

BIT[31:8]	UID[63:40]	Lot number(ASCII 码格式)
BIT[7:0]	UID[39:32]	Wafer number(8-bit 无符号数)

26.1.4 UID 寄存器 3（UID_DR3）

UID_DR3			地址偏移	0x08		复位值	0xXXXX XXXX	
	Bit31	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
控制位	UID[95:88]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x
	Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
控制位	UID[87:80]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	UID[79:72]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	UID[71:64]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x

BIT[31:8]	UID[95:64]	Lot number(ASCII 码格式)

26.2 存储区大小数据寄存器

基址：0x1FFF F7CC				
偏移地址	寄存器名	复位值	功能描述	章节
0x00	**FLS_DR1**	0xXXXX XXXX	Flash 空间大小数据寄存器	26.2.1

26.2.1 Flash 空间大小数据寄存器(FLS_DR1)

FLS_DR1			地址偏移	0x00		复位值	0xXXXX XXXX	
	Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
控制位	FLASH_SIZE[15:8]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
控制位	FLASH_SIZE[7:0]							
R/W	r	r	r	r	r	r	r	r
复位值	x	x	x	x	x	x	x	x

BIT[15:0]	FLASH_SIZE [15:0]	Flash 空间大小 这个位字段表示设备 Flash 内存的大小，以 kbytes 为单位 举例：0x020 为 32kbytes，0x040 为 64kbytes

27 版本修改记录

版本号	修订者	修订日期	修订内容
V1.0.0	晟矽微电	2020-10-7	初版，预发布版本 1、 当前版本 IIC 和 SPI/I2S 章节部分功能不可用，具体请联系晟矽微电获取使用 demo 样例； 2、 当前版本 ADC 仅支持硬件触发，具体请联系晟矽微电获取使用 demo 样例；
V1.0.1	晟矽微电	2021-12-3	1、 EXTI 章节 完善事件管理/功能描述章节； 去除寄存器相关无效位； 2、 CMP 和 OPA 章节，修改寄存器命名 3、 UID 章节，补充 falsh 空间大小寄存器，位于 system rom 区 4、 勘误
V1.0.2	晟矽微电	2022-1-6	勘误 1、 修改系统框图，删除 DIV 指令，M0 内核仅支持硬件 32bit 乘法指令
V1.1.0	晟矽微电	2022-4-21	修改文档样式
V1.2.0	晟矽微电	2022-10-18	1、 修改勘误 去除各模块章节，芯片不支持的功能描述，及相关寄存器控制位； ADC 章节更新温度测量计算公式
V1.2.1	晟矽微电	2023-10-25	1、 修改勘误；

28 免责声明

本资料内容为晟矽微电（以下简称“我公司”）版权所有。

我公司将力求本资料的内容做到准确无误，但同时保留在不通知用户的情况下，对本资料内容的修改权。如您需要获得最新的资料，请及时联系我公司。

我公司将尽最大努力为您提供高品质、高稳定性的产品。尽管如此，由于一般半导体器件的电气敏感性及易受到外部物理损伤等固有特性存在，所以难免造成半导体器件出现故障或失效的可能。当您使我公司产品时，有责任按照本资料以及相连资料中提到的“规则”来设计一个稳定及安全可靠的系统环境。另外，在遇到超规格（本资料中未描述到内容）的使用，请您提前咨询我公司，以免因我公司产品在一些特殊设备中或者特殊环境下的使用，导致财产损失、人员伤亡等严重后果的发生。因超规格的使用、未经咨询授权我公司产品的使用，我公司对此不承担任何法律责任。