# RX32S11 参考手册

文档编号：RM00002

基于 Arm®Cortex®-M0 内核的 32 位专业电机微控制器

版本：V2.2

# 目录

# 表目录

# 图目录

# 1 简介

本参考手册用于帮助开发者理解并使用睿兴科技（南京）有限公司（后文简称“睿兴”或“RX”）的芯片。本文提供了 RX32S11 微控制器的存储和外设的完整信息。

相关封装和电气特性请参考相应的数据手册。

本参考手册和数据手册均可从睿兴官网 www.rxtek-icore.com 获得。

# 2 文档约定

## 2.1 寄存器缩写列表

| | |
|---|---|
| read/write(rw) | 该位可读可写。 |
| read-only(r) | 该位只读。 |
| write-only(w) | 该位只写。 |
| read/clear(rc_w1) | 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。 |

## 2.2 词汇表

Word：字，32 位长度数据。
Half-word：半字，16 位长度数据。
Byte：字节，8 位长度数据

# 3 系统和存储器概述

## 3.1 系统架构

RX32S11 集成了 Arm®Cortex®-M0 的内核，是电机专用的高性能、低功耗微控制器。

- ●一个驱动单元
  - – Cortex-M0 内核及先进高性能总线（AHB_Lite）
  - – DMA
- ●三个被动单元
  - – 内置 Flash
  - – 内置 SRAM
  - – AHB 外设包括 AHB 到 APB 的桥和 APB 外设

## 3.2 存储器结构

### 3.2.1 介绍

程序存储器，数据存储器，寄存器和 I/O 口统一编址在 4G 的线性地址空间里。

字节在存储器中以小端格式编码。一个字中编号最低的字节被认为是该字的最低有效字节，最高地址字节是最高有效字节。

### 3.2.2 存储器映像和寄存器边界地址

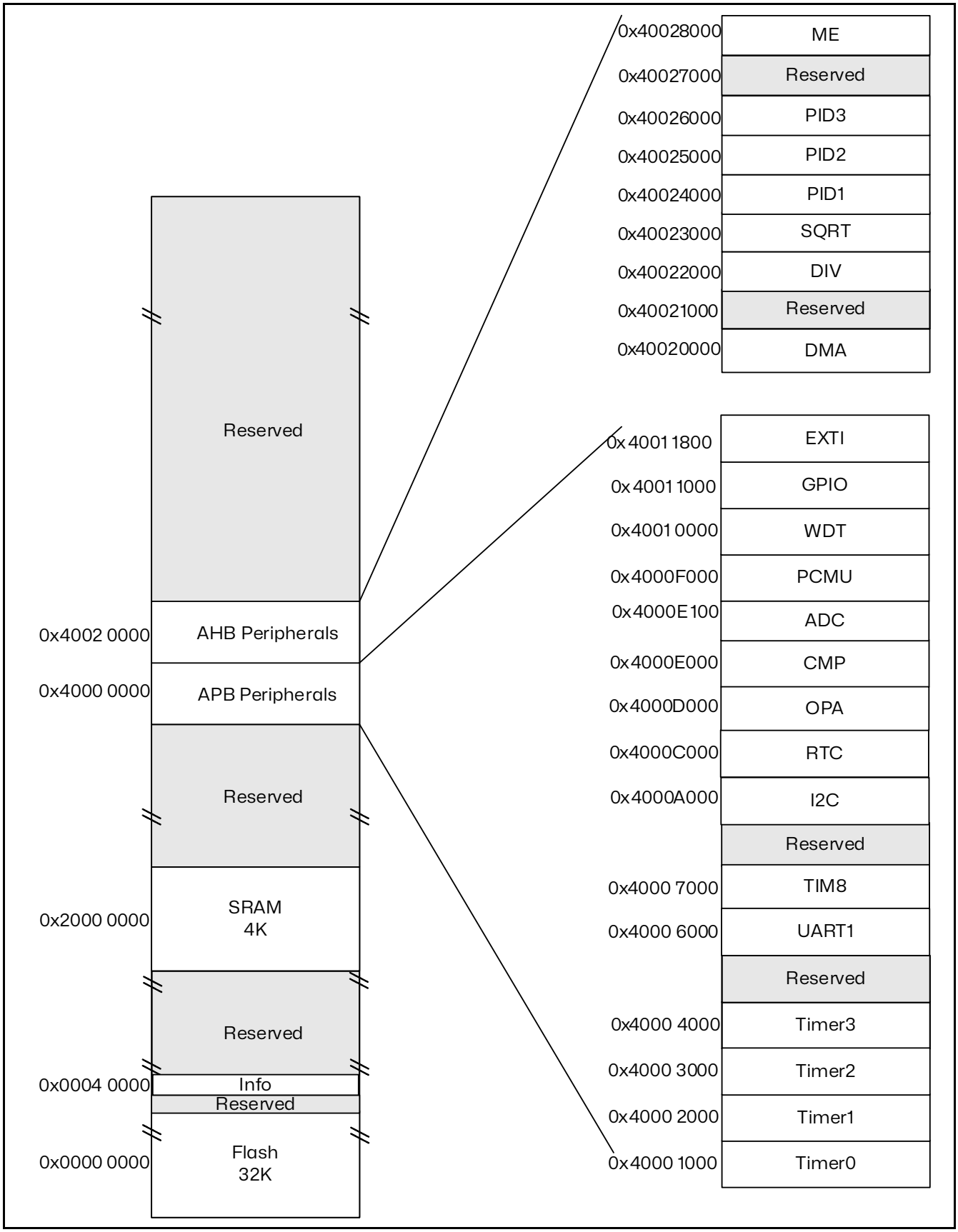



图 3.1 存储器映像

所有未分配给片上存储器和外围设备的内存映射区域都被配置为“Reserved”或者“保留”。有关可用内存和寄存器区域的详细映射，请参阅下表。

下表列出了设备中可用外设的边界地址。

表 3.1 RX32S11 存储器映像和外设边界地址

| 总线 | 边界地址 | 外设 |
|---|---|---|
| AHB | 0x40020000-0x40020FFF | DMA |
| | 0x40022000-0x40022FFF | DIV |
| | 0x40023000-0x40023FFF | SQRT |
| | 0x40024000-0x40024FFF | PID1 |
| | 0x40025000-0x40025FFF | PID2 |
| | 0x40026000-0x40026FFF | PID3 |
| | 0x40028000-0x40028FFF | ME |
| APB | 0x00000000-0x0000FFFF | Flash |
| | 0x20000000-0x20000FFF | RAM |
| | 0x40001000-0x40001FFF | TIM0 |
| | 0x40002000-0x40002FFF | TIM1 |
| | 0x40003000-0x40003FFF | TIM2 |
| | 0x40004000-0x40004FFF | TIM3 |
| | 0x40006000-0x40006FFF | UART1 |
| | 0x40007000-0x40007FFF | TIM8 |
| | 0x4000A000-0x4000AFFF | I2C |
| | 0x4000C000-0x4000CFFF | RTC |
| | 0x4000D000-0x4000DFFF | OPA |
| | 0x4000E000-0x4000E0FF | CMP |
| | 0x4000E100-0x4000EFFF | ADC |
| | 0x4000F000-0x4000FFFF | PCMU |
| | 0x40010000-0x40010FFF | WDT |
| | 0x40011000-0x400117FF | GPIO |
| | 0x40011800-0x40011FFF | EXTI |

## 3.3 内置 SRAM

RX32S11 内置总计 4KB 的 SRAM。

这些 SRAM 可以以字节（8 位）、半字节（16 字节）或字（32 位）进行访问。这些存储器可以在没有等待周期的情况下由 CPU 或者 DMA 寻址。

## 3.4 Flash 概述

Flash 有两个不同的物理区域组成：

- 主 Flash 块。用于存储应用代码和数据。包含选项字节：用于硬件和存储器保护。
- 信息块。

Flash 接口基于 AHB 协议进行指令和数据访问。Flash 寄存器具有 Flash 的操作（擦除/写入）功能。

# 4 内置 Flash 存储器（FLASH）

## 4.1 介绍

Flash 接口管理 CPU AHB-Lite 对 Flash 进行访问，具有 Flash 擦除和写入功能以及读写保护机制。Flash 接口通过一个包含指令预取的系统来加速代码的执行。

## 4.2 Flash 主要特征

Flash 接口特征：

- 总计 32KB 的 Flash。
- Flash 读取支持字节（8 位）、半字（16 位）和字（32 位）操作。
- Flash 写入支持字节（8 位）、半字（16 位）和字（32 位）操作。
- 支持页（1KB）擦除和全擦除。
- 读保护。
- 选项字节加载器。

## 4.3 Flash 功能描述

### 4.3.1 Flash 结构

Flash 包含 32 页（每页 1KB）的主存储块和信息块，如下表所示。

表 4.1 Flash 结构

| 模块 | 名称 | 地址 | 大小（字节） |
|---|---|---|---|
| 主存储块 | 页 0 | 0x0000 0000 - 0x0000 03FF | 1K |
| | 页 1 | 0x0000 0400 - 0x0000 07FF | 1K |
| | 页 2 | 0x0000 0800 - 0x0000 0BFF | 1K |
| | 页 3 | 0x0000 0C00 - 0x0000 0FFF | 1K |
| | 页 4 | 0x0000 1000 - 0x0000 13FF | 1K |
| | … | … | … |
| | 页 31 | 0x0000 7C00 - 0x0000 FFFF | 1K |
| 信息块 | INFO | 0x0004 0000 – 0x0004 07FF | 2K |

### 4.3.2 Flash 擦除和编程

Flash 支持在线编程和应用编程。

在线编程（in-circuit programming，ICP）是指使用 SWD 协议或者 bootloader 更新 Flash 的内容。

应用编程（in-application programming，IAP）是指使用芯片支持的通讯协议更新 Flash 的内容。IAP 允许用户在芯片运行过程中更新 Flash 的内容。然而要实现 IAP，需要预先使用 ICP 将部分应用代码编写进 Flash 中。

如果在 Flash 操作期间复位芯片将无法保证 Flash 的内容正确。

### 4.3.3 解锁 Flash

复位后，Flash 控制寄存器（Flash control register，CMU_FLASHCON）不可写入，这是为了防止 Flash 由于干扰产生误操作。通过以下流程可以解锁 CMU_FLASHCON。

1. 在 Flash 锁定控制位寄存器（FLASHLOCK_KEY）写入 KEY = 0x7A68。

对该寄存器写入 0x7A68 后，FLASH 被解锁，用户可以写操作 FLASH。写入非 0x7A68 数据后，FLASH 被锁定，用户禁止写操作 FLASH。

### 4.3.4 Flash 擦除流程

Flash 擦除可以分为页擦和全擦。全擦不会擦除信息区（系统存储器和选项字节）。

**页擦**

页擦流程如下：

1. 解锁 Flash 存储地址。
2. 将要擦除的页地址写入 Flash 寄存器。
3. 将 32 位指令任意数据写进需要擦除 Flash 页内的任一地址。
4. 等待 Flash 页擦除操作完成，最长 2ms。

**全擦**

全擦指令可以完全擦除主存储块，不会影响到信息块。全擦流程如下：

1. 解锁 Flash 存储地址。
2. 将要擦除的全块地址写入 Flash 寄存器。
3. 将 32 位指令任意数据写进需要擦除的 32K Flash 的任一地址。
4. 等待 Flash 全擦除操作完成，最长 10ms。

### 4.3.5 FLASH 写入流程

主 FLASH 可以一次写入一个 16 bit 的数据。写入流程如下：

1. 解锁 Flash 存储地址。
2. 将 Flash 寄存器地址位置 1
3. 将 32 位或 16 位或 8 位指令数据写进 Flash 地址中。
4. 等待 Flash 写操作完成，最长 20us。

## 4.4 选项字节功能描述

地址：0x0000 0FC0

复位值：0x0000 0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | HIIPM[3:0] | | | | sramRDflh[3:0] | | | | WDT_EN[3:0] | | | |
| | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FLASH[7:0] | | | | | | | | Reserved | | | | | ATRL | Reserved | |
| rw | rw | rw | rw | rw | rw | rw | rw | | | | | | rw | | |

Bits 31:28 **保留**，必须保持为 0

Bits 27:24 **HIIPM [3:0]**：I/O 默认态控制位

1010 或 0101：I/O 默认配置为高阻态

其他：I/O 默认配置为输入态

Bits 23:20 **sramRDflh[3:0]：**

1、sramRDflh[3:0] =1111 时，程序在 SRAM 空间运行,对 Flash main block 读取不加密

2、sramRDflh[3:0] !=1111 时，程序在 SRAM 空间运行，对 Flash main block 读取加密，读取值为固定 55555555；此时 Flash 擦写状态如下：

1） SRAM 加密打开，此时通过 SWD 口对 0-8K flash 不可页擦和写（用户程序可页擦和写 0-8K），但可进行全擦操作，可以页擦和写 9K-31K。

2） 不通过 SWD 接口，而通过用户程序，即使 SRAM 加密有打开，0-31K 也可以全擦、页擦和写。

Bits 19:16 **WDT_EN[3:0]：** Sleep 和 Hold 模式看门狗使能位

0101：看门狗在 Sleep 和 Hold 模式下关闭

其他：看门狗在 Sleep 和 Hold 模式下开启

Bits 15:8 **FLASH[7:0]：** Flash 加密位

0xFF：Flash 不加密

其他：Flash 加密

Bits 2:2 **ATRL：** 自动装载使能位

1：自动装载功能使能

0：自动装载功能屏蔽

## 4.5 Flash 寄存器

相关参考 S11 参考手册的时钟寄存器部分。

# 5 电源控制（PWR）

## 5.1 电源

RX32S11 的工作电压（VDD/VDDA）为 2.5~5 .5V。VDD 和 VDDA 必须保持一致。

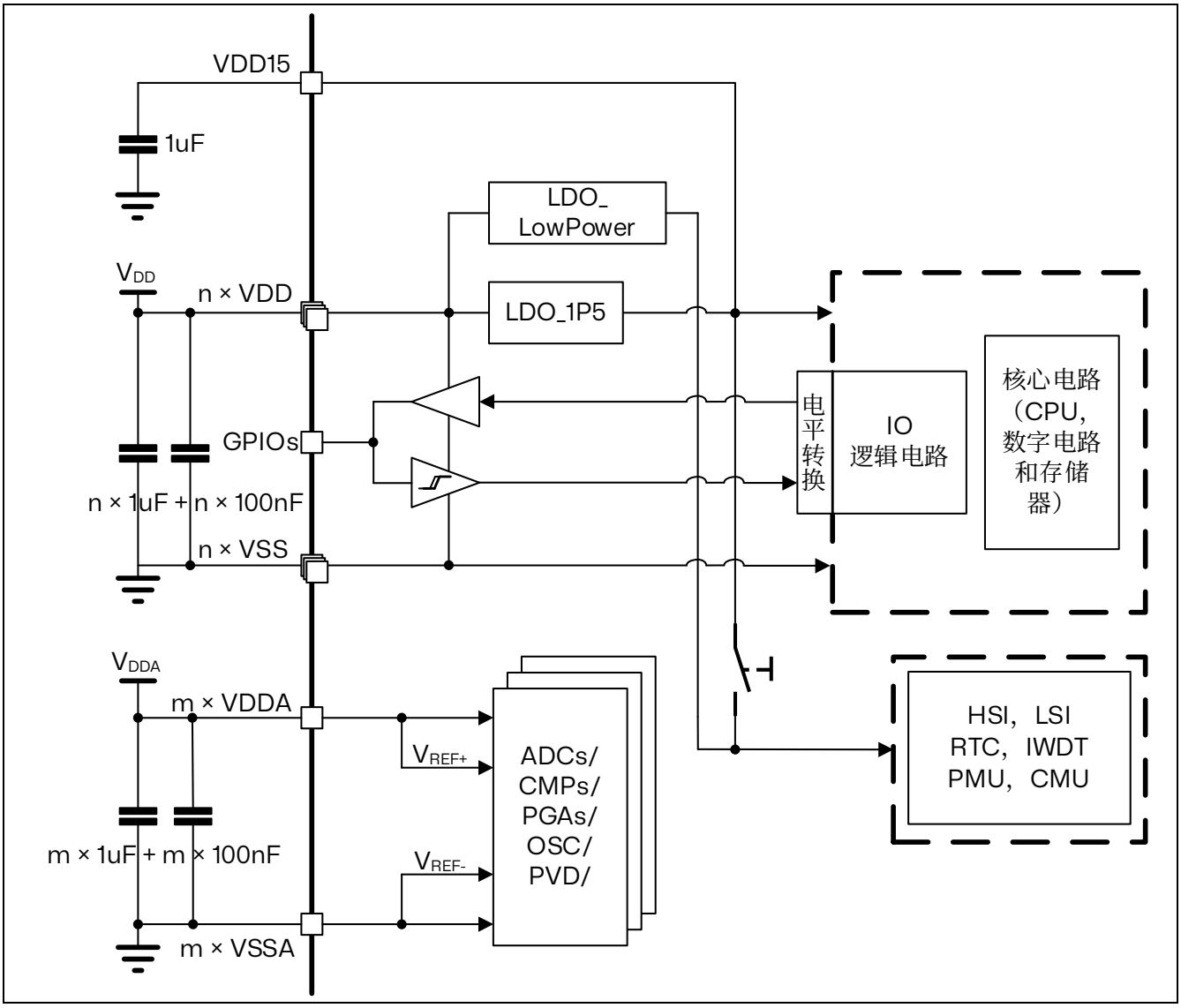



图 5.1 电源框图

## 5.2 电源管理器

### 5.2.1 上电复位（POR）和掉电复位（PDR）

RX32S11 内部有一个完整的上电复位（POR）和掉电复位（PDR）电路，当供电电压达到 2.5V 时，系统既能正常工作。

当 VDD/VDDA 低于指定的限位电压 $V_{POR}/V_{PDR}$ 时，系统保持为复位状态，无需外部复位电路。关于上电复位和掉电复位的细节请参考数据手册的电气特性部分。

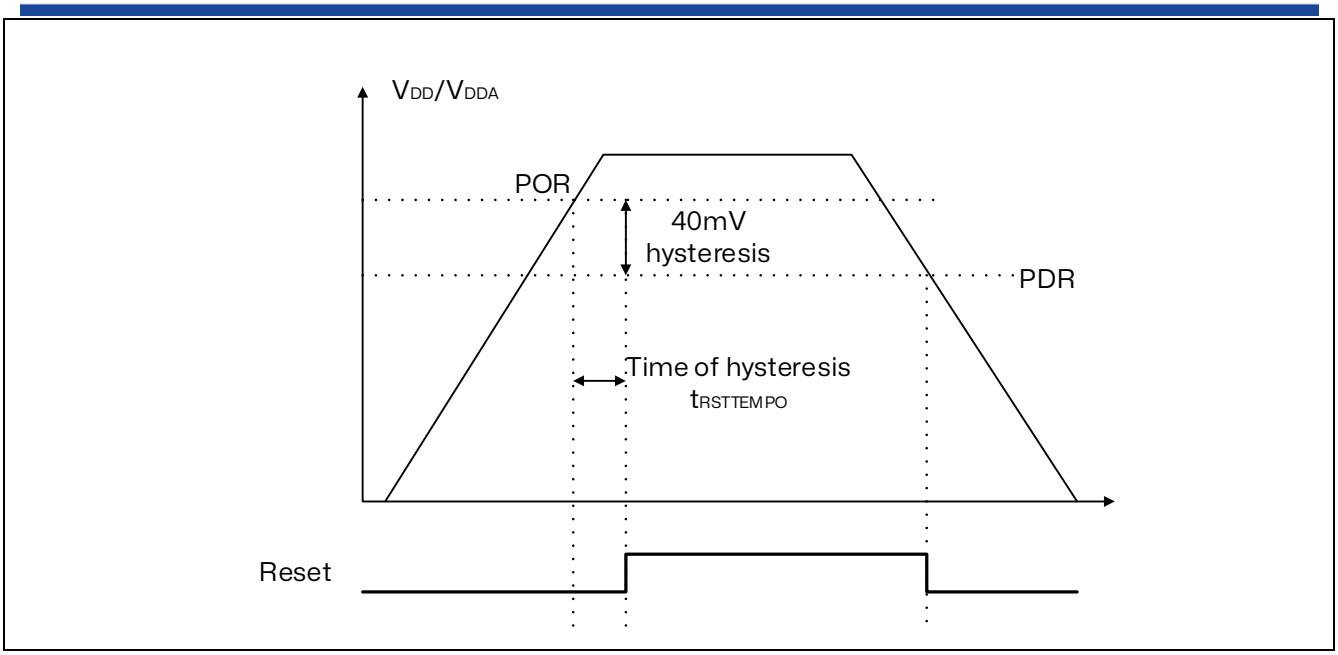



图 5.2 上电复位和掉电复位波形图

### 5.2.1 可编程电压检测器（PVD）

用户可以利用 PVD 对 VCC 电压与电源检测阈值配置寄存器（PWR_VDETCFG）中的 VCC_LVL[1:0]位进行比较来监控电源，这几位选择监控电压的阀值。通过设置中断使能寄存器（PWR_PMUIE）中的 VCCIE 来使能 VCC 检测中断。

电源状态寄存器（PWR_PMUSTA）中的 VCC_FLG 标志用来表明 VCC 是大于还是小于 PVD 的电压阀值。如果中断寄存器是使能的，当 VCC 下降到 PVD 阀值以下和（或）当 VCC 上升到 PVD 阀值之上时，就会产生 PVD 中断。同时中断标志寄存器（PWR_PMUIF）中的 VCCIF 将被置 1，该位可被软件写 0 清 0。

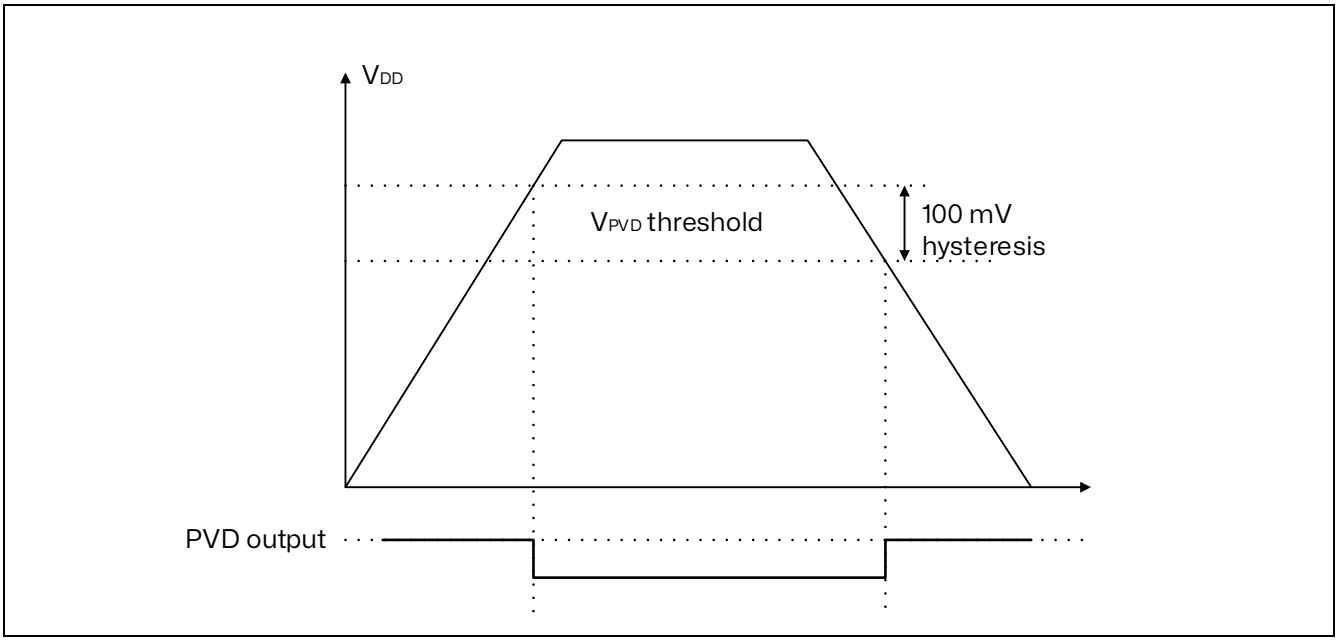



图 5.3 PVD 门限

## 5.3 低功耗模式

在系统或电源复位以后，微控制器处于运行状态。当 CPU 不需继续运行时，可以利用多种低功耗模式来节省功耗，例如等待某个外部事件时。用户需要根据最低电源消耗、最快速启动时间和可用的唤醒源等条件，选定一个最佳的低功耗模式。

RX32S11 支持两种低功耗模式，用户可以在以下模式中进行选择：

●睡眠模式：睡眠模式下，CPU 被停止且保持当前状态不执行任何操作，但仍维持着正常供电。当用户需要在该模式下达到最低功耗时，可以自行关闭大功耗 LDO（默认打开）。在该模式下 CPU 可以被任意中断唤醒。

●待机模式：待机模式下，CPU 被停止，LDO 关闭，外设停止运行。在该模式下 CPU 可以被中断或者事件唤醒。

*注意：*

（1）*例程库* V1.5 *低功耗模式对应此新版参考手册，旧版例程库* V1.4 *及以前对应旧版参考手册。*

（2）*在* Debug *模式下无法进入低功耗模式。且* NRST *复位无法退出* Debug *模式，必须掉电复位。*

表 5.1 低功耗模式一览

| 模式 | 进入 | 唤醒 | 唤醒后时钟 | 对时钟影响 |
|---|---|---|---|---|
| 睡眠 | WFI 和 SCB->SCR = 0x0000 | 任一中断 | 与进入睡眠模式前一致 | CPU 时钟关闭 |
| 待机 | WFI 和 SCB->SCR = 0x0004 | 外部中断 | HRC | 关闭除 LRC 以外所有时钟 |
| | | RTC 中断 | | |
| | | PMU 中断 | | |
| | | UART RX 中断 | | |

### 5.3.1 睡眠模式

**进入睡眠模式**

关于如何进入睡眠模式，详见后表。

**退出睡眠模式**

如果执行 WFI 进入睡眠模式，任意一个被嵌套向量控制器（NVIC）响应的外设中断都能将系统从睡眠模式中唤醒。

该模式唤醒所需的时间最短，因为没有时间损失在中断的进入和退出上。

表 5.2 睡眠模式进入和唤醒

| 进入 | 唤醒 | 唤醒延迟 |
|---|---|---|
| SLEEPDEEP = 0<br>Sleep_LDO =0<br>WFI | WKUP 引脚的上升沿、RTC 闹钟事件的上升沿、NRST 引脚上外部复位、IWDG 复位。 | 复位阶段时电压调节器的启动 |

### 5.3.2 待机模式

待机模式可以实现 MCU 的最低功耗。该模式是在 Arm®Cortex®-M0 深度休眠模式时关闭电压调节器，整个 1.5 V 供电区域会被断电。HRC 和 HSE 振荡器也被断电，SRAM 和寄存器的内容会丢失，只有待机电路维持供电。

**进入待机模式**

关于如何进入待机模式，详见后表。

**退出待机模式**

当一个外部复位（NRST 引脚）、IWDG 复位、WKUP 引脚上的上升沿或 RTC 闹钟事件的上升沿发生时，微控制器从待机模式退出。从待机唤醒后，除了电源控制/状态寄存器（PWR_PMUSTA），所有寄存器被复位。

从待机模式唤醒后的代码执行等同于复位后的执行。电源控制/状态寄存器（PWR_PMUSTA）将会指示内核由待机状态退出。

表 5.3 待机模式进入和唤醒

| 进入 | 唤醒 | 唤醒延迟 |
|---|---|---|
| SLEEPDEEP = 1<br>WFI | WKUP 引脚的上升沿、RTC 闹钟事件的上升沿、NRST 引脚上外部复位、IWDG 复位。 | 复位阶段时电压调节器的启动 |

## 5.4 低功耗模式下的自动唤醒（AWU）

RTC 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微控制器（自动唤醒模式）。RTC 提供一个可编程的时间基数，用于周期性从待机或睡眠模式下唤醒。

## 5.5 电源控制寄存器

### 5.5.1 PMU 配置寄存器（PWR_PMUCON）

地址偏移：0x00

复位值：0x0000 0013

访问：无等待，支持字，半字和字节访问

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

Bit 15:5 保留，必须保持为 0

Bit 4 **Sleep_LDO**：在 Sleep 模式下选择打开/关闭大功耗 LDO（默认打开）

0：关闭大功耗 LDO

1：打开大功耗 LDO（default）

*注：当用户需要在 Sleep 模式下达到最低功耗时，可以将该大功耗 LDO 关闭，届时芯片自动切换使用低驱动能力低功耗的 LDO*

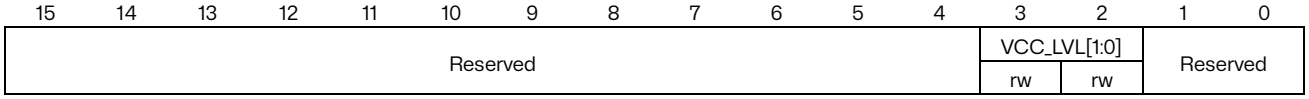
Bits 3:0 保留，必须保持为 0

### 5.5.2 电源检测阈值配置寄存器（PWR_VDETCFG）

地址偏移：0x04

复位值：0x0000 0069

访问：无等待，支持字，半字和字节访问




Bit 15:4 保留，必须保持为 0

Bit 3:2 **VCC_LVL[1:0]**：VCC_DET 检测阈值控制位

00：2.2V

01：2.8V

10：3.6V

11：4.2V

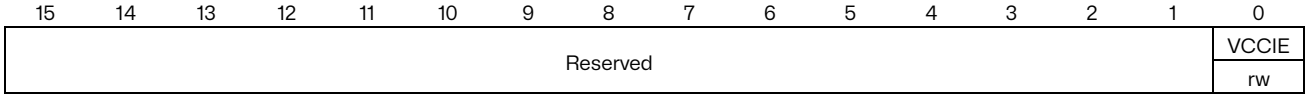
Bit 1:0 保留，必须保持为 0

### 5.5.3 PMU 中断使能寄存器（PWR_PMUIE）

地址偏移：0x0C

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问




Bit 15:1 保留，必须保持为 0

Bit 0 **VCCIE**：VCC 检测中断使能位

0：关闭

1：允许

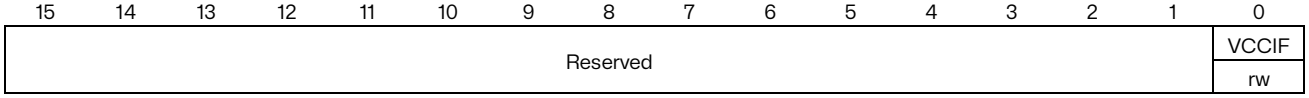
*注：需要同时使能* PMUIE *使能的中断才有效。*

### 5.5.4 PMU 中断标志寄存器（PWR_PMUIF）

地址偏移：0x10

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问




Bit 15:1 保留，必须保持为 0

Bit 0 **VCCIF**：VCC 检测中断标志位

当系统电源 VCC 电压下降到低于设定阈值或上升到高于设定阈值时，该位置 1。软件写 0 清 0。

*注：该寄存器不能被* Wake_UP *唤醒复位。*

### 5.5.5 PMU 状态寄存器（PWR_PMUSTA）

地址偏移：0x14

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

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

Bit 15:1 保留，必须保持为 0

Bit 0 **VCC_FLG**：工作电压 VCC 电压状态

0：表示 VCC 小于设定阈值（VCC_LVL[3:0]）

1：表示 VCC 大于设定阈值（VCC_LVL[3:0]）

*注：该寄存器为只读寄存器。*

### 5.5.6 唤醒标志寄存器（PWR_WAKEIF）

地址偏移：0x18

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | INT8 WKIF | INT7 WKIF | Reserved | | | | | | | RTC WKIF | Reserved | | | |
| | | r | r | | | | | | | | r | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | RX1 WKIF | RX0 WKIF | INT6 WKIF | INT5 WKIF | INT4 WKIF | INT3 WKIF | INT2 WKIF | INT1 WKIF | INT0 WKIF | Reserved | PMU WKIF |
| | | | | | r | r | r | r | r | r | r | r | r | | r |

Bit 31:30 保留，必须保持为 0

Bit 29 **INT8WKIF**：INT8 唤醒标志

INT8 唤醒发生时，此位置为 1

Bit 28 **INT7WKIF**：INT7 唤醒标志

INT7 唤醒发生时，此位置为 1

Bit 27:21 保留，必须保持为 0

Bit 20 **RTCWKIF**：RTC 唤醒标志

SLEEP/HOLD 模式下 RTC 中断发生时将会产生 RTC 唤醒，此位置为 1.（具体 RTC 唤醒源头需要查看 RTCIF 寄存器）

Bit 19:11 保留，必须保持为 0

Bit 10 **RX1WKIF**：RX1 唤醒标志

RX1 唤醒发生时，此位置为 1

Bit 9 **RX0WKIF**：RX0 唤醒标志

RX0 唤醒发生时，此位置为 1

Bit 8 **INT6WKIF**：INT6 唤醒标志

INT6 唤醒发生时，此位置为 1

Bit 7 **INT5WKIF**：INT5 唤醒标志

INT5 唤醒发生时，此位置为 1

Bit 6 **INT4WKIF**：INT4 唤醒标志

INT4 唤醒发生时，此位置为 1

Bit 5 **INT3WKIF**：INT3 唤醒标志

INT3 唤醒发生时，此位置为 1

Bit 4 **INT2WKIF**：INT2 唤醒标志

INT2 唤醒发生时，此位置为 1

Bit 3 **INT1WKIF**：INT1 唤醒标志

INT1 唤醒发生时，此位置为 1

Bit 2 **INT0WKIF**：INT0 唤醒标志

INT0 唤醒发生时，此位置为 1

Bit 1 保留，必须保持为 0

Bit 0 **PMUWKIF**：PMU 唤醒标志

SLEEP/HOLD 模式下 PMU 事件发生时将会产生 PMU 唤醒，此位置为 1.（具体 PMU 唤醒源头需要查看 PMUIF 寄存器）

# 6 复位和时钟控制

## 6.1 复位

RX32S11 支持两种复位，分别是系统复位，电源复位。

### 6.1.1 系统复位

除了时钟控制器的 CMU_RSTSR 寄存器中的复位标志位以外，系统复位将复位所有寄存器至它们的复位状态。

当发生以下任一事件时，产生一个系统复位：

1. NRST 引脚上的低电平（外部复位）
2. 独立看门狗计数终止（IWDG 复位）
3. 软件复位（SW 复位）
4. 低功耗管理复位

可通过查看 CMU_RSTSR 控制状态寄存器中的复位状态标志位识别复位事件来源。

**软件复位**

通过将 Cortex®-M0 中断应用和复位控制寄存器中的 SYSRESETREQ 位置 1，可实现软件复位。请参考 Cortex®-M0 技术参考手册获得进一步信息。

### 6.1.2 电源复位

当以下事件中之一发生时，产生电源复位：

1. 上电/掉电复位（POR/PDR 复位）
2. 从睡眠模式中返回

电源复位将复位除了备份区域外的所有寄存器。

下图中复位源将最终作用于 RESET 引脚，并在复位过程中保持低电平。复位入口矢量被固定在地址 0x0000_0004。

芯片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。

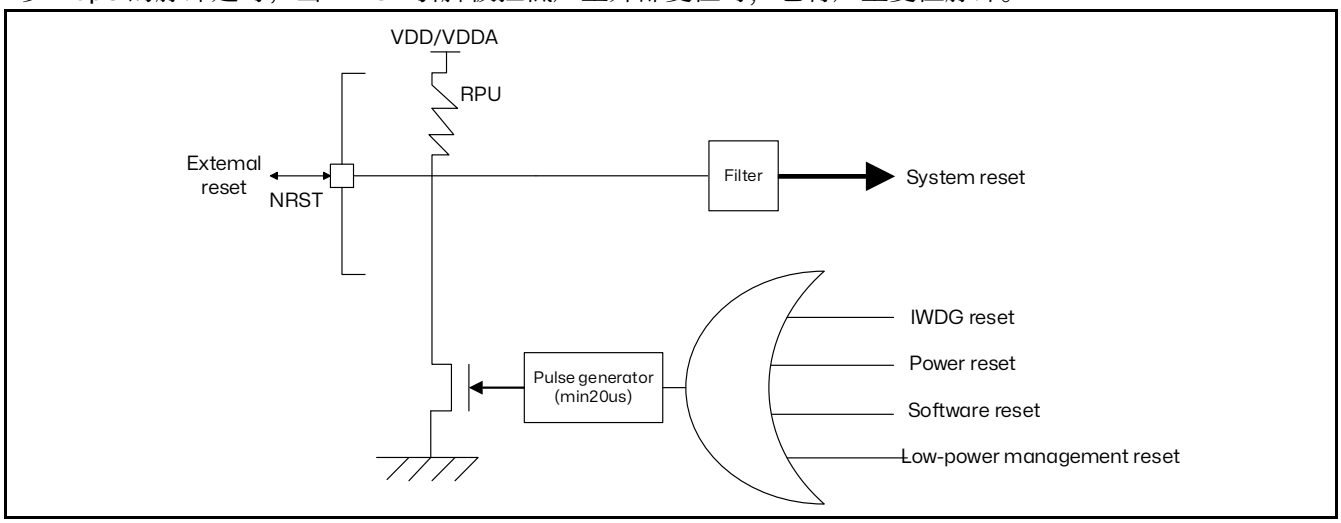



图 6.1 复位电路框图

## 6.2 时钟

2 种不同的时钟源可被用来驱动系统时钟（SYSCLK）：

- HRC
- HSE

该芯片还有 1 种二级时钟源：

- 32kHz 低速内部振荡器 LRC，可以用于驱动独立看门狗和通过程序选择驱动 RTC。RTC 用于从睡眠模式下自动唤醒系统。

当不被使用时，任一个时钟源都可被独立地启动或关闭，由此优化系统功耗。

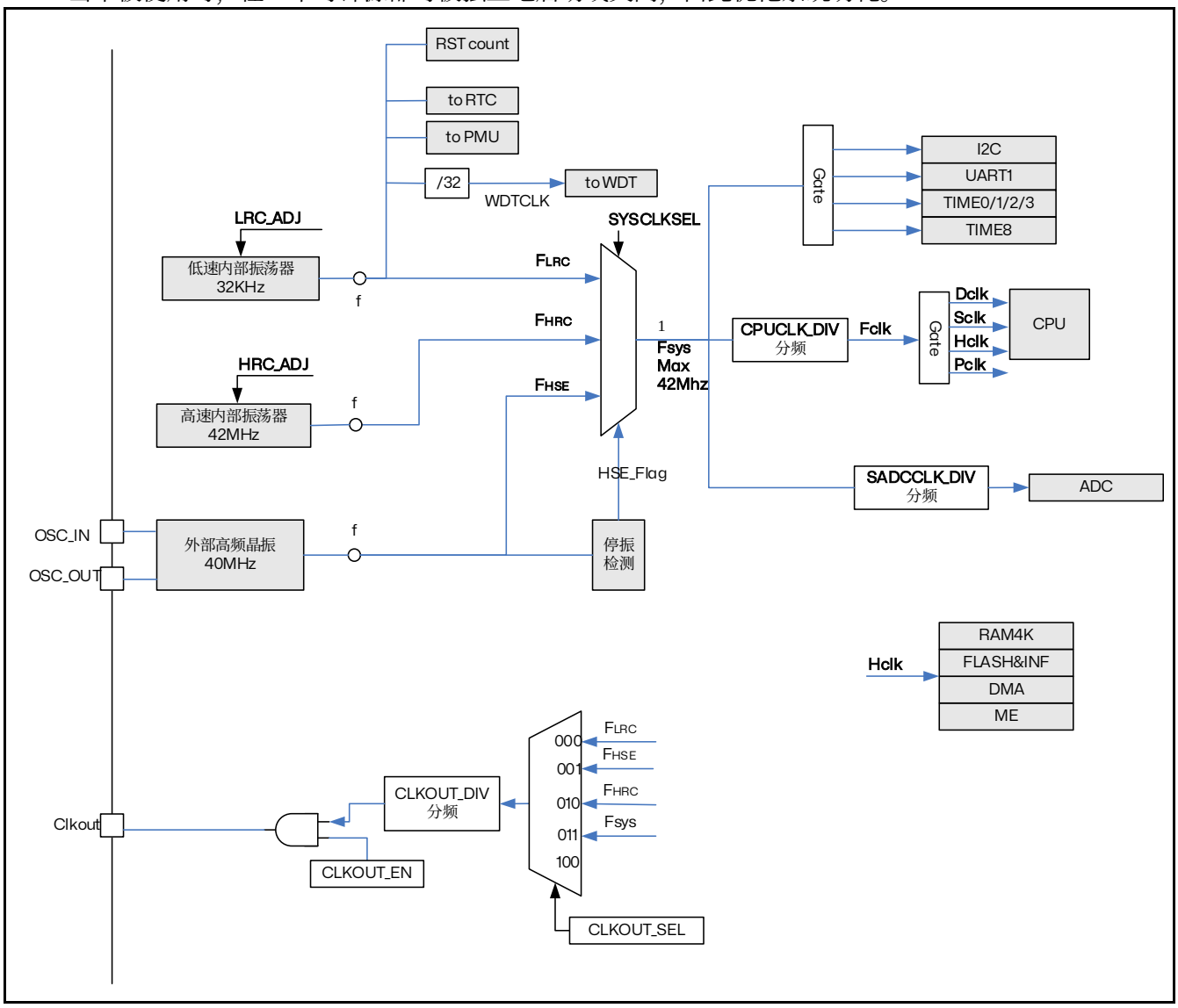



图 6.2 时钟树

用户可通过多个预分频器配置 CPU 和 ADC 域的频率，最大允许频率是 42MHz。

通过对 SysTick 控制与状态寄存器的设置，可选择 Cortex（HCLK）时钟作为 SysTick 时钟。ADC 时钟由 AHB 时钟经 1、2、3、4、5、6、7 或 8 分频后获得。

### 6.2.1 HRC 时钟

HRC 时钟信号由内部 42MHz 的 RC 振荡器产生，可作为系统时钟输入。
HRC 振荡器能够在不需要任何外部器件的条件下提供系统时钟。

### 6.2.2 LRC 时钟

LRC 担当一个低功耗时钟源的角色，它可以在睡眠模式下保持运行，为独立看门狗和自动唤醒单元提供时钟。LRC 时钟频率大约 32kHz（在 30kHz 和 60kHz 之间）。进一步信息请参考数据手册中有关电气特性部分。

### 6.2.3 系统时钟（SYSCLK）选择

系统复位后，HRC 振荡器被选为系统时钟。当时钟源被直接作为系统时钟时，不能被停止。
只有当目标时钟源准备就绪了（经过启动稳定阶段的延迟），从一个时钟源到另一个时钟源的切换才会发生。在被选择时钟源没有就绪时，系统时钟的切换不会发生。直至目标时钟源就绪，才发生切换。
在时钟选择寄存器（CMU_CLKOUTSEL）里可选择哪个时钟目前被用作系统时钟。

### 6.2.4 RTC 时钟

RTC 时钟固定为 LRC。

### 6.2.5 看门狗时钟

如果独立看门狗已经由硬件选项或软件启动，LRC 振荡器将被强制在打开状态，并且不能被关闭。在 LRC 振荡器稳定后，时钟供应给 IWDG。

### 6.2.6 时钟输出

微控制器允许输出时钟信号到外部 ClockOut 引脚。
相应的 GPIO 端口寄存器必须被配置为相应功能。以下四个时钟信号可被选作 ClockOut 时钟：

- LRC
- HSE
- HRC
- SYSCLK

时钟的选择由 CLKOUT 时钟选择寄存器（CMU_CLKOUTSEL）中的 CLKOUT_SEL[1:0]位控制。

## 6.3 CMU 寄存器

### 6.3.1 写保护寄存器（CMU_WPREG）

地址偏移：0x00
复位值：0x0000 0000
访问：0 到 2 个等待周期，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WPREG [15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **WPREG[15:0]**：写保护

1、 WPREG 写入 0xA55A，则关闭写保护功能，用户可以写操作被保护的寄存器。
2、 WPREG 写非 0xA55A，则开启写保护功能，用户禁止写操作被保护的寄存器。

0x0001：表示写保护关闭，用户可以写操作被保护的寄存器
0x0000：表示写保护开启，用户禁止写操作被保护的寄存器
*注：下列寄存器要写入值前，必须将* WPREG *先写入* 0xA55A *解锁*
SYSCLKCFG，LRCADJ，CLKADJ，CLKCTRL0，CLKCTRL1，CPUCFG，PLLCFG，FLASHCON。

### 6.3.2 系统时钟配置寄存器（CMU_SYSCLKCFG）

地址偏移：0x04
复位值：0x0000 0002
访问：无等待，支持字，半字和字节访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | WCLKEN | Reserved | | | | | SYSCLK_SEL[1:0] | |
| | | | | | | | | rw | | | | | | rw | rw |

Bit 31:8 保留，必须保持为 0

Bit 7 **WCLKEN：**时钟配置寄存器写保护位

如果用户要更改系统时钟选择，必须同时将 WCLKEN 位置 1，例如：b1xxxxxxx，才可以对系统时钟选择位 SYSCLK_SEL[1:0]进行写操作。

Bit 6:2 保留，必须保持为 0

Bit 1:0 **SYSCLK_SEL[1:0]：**系统时钟选择控制位

00：FLRC
01：FHSE
10：FHRC(Default)
11:：X

### 6.3.3 芯片状态指示寄存器（CMU_JTAGSTA）

地址偏移：0x08
复位值：0x0000 0001
访问：无等待，支持字，半字和字节访问

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

Bit 15:2 保留，必须保持为 0

Bit 1 **FLAG：**该位用于指示芯片是否处于 JTAG 调试状态

0：表示芯片处于正常运行状态
1：表示芯片处于调试状态

*注：*bit0 *的值默认为*1，*用户无须更改该位。*

### 6.3.4 LRC 时钟调整寄存器（CMU_LRCADJ）

地址偏移：0x0C

复位值：0x0000 00b5

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | LRC_ADJ[7:0] | | | | | | | |
| | | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:8 保留，必须保持为 0

Bit 7:0 **LRC_ADJ[7:0]**：LRC 输出频率调节控制位

### 6.3.5 HRC 时钟调整寄存器（CMU_HRCADJ）

地址偏移：0x10

复位值：0x0000 0040

访问：无等待，支持字，半字和字节访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | HRCADJ_TC[5:0] | | | | | | Reserved | HRCADJ_ADJ[6:0] | | | | | | |
| | | rw | rw | rw | rw | rw | rw | | rw | rw | rw | rw | rw | rw | rw |

Bit 31:14 保留，必须保持为 0

Bit 13:8 **HRCADJ_TC[5:0]**：HRC 输出频率温度参数

Bit 15:7 保留，必须保持为 0

Bit 6:0 **HRCADJ_ADJ[6:0]**：HRC 输出频率调节控制位。芯片出厂时，HRC 已经经过校准，校准之后频率为 42MHz。

### 6.3.6 时钟状态寄存器（CMU_CLKSTA）

地址偏移：0x18

复位值：0x0000 0020，除复位标志外由系统复位清除，复位标志只能由电源复位清除

访问：0 到 3 等待周期，支持字，半字和字节访问

当连续对该寄存器进行访问时，将插入等待状态。

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | HSE_LOCK | Reserved | | | HSE_FLAG | Reserved | |
| | | | | | | | | | r | | | | r | | |

Bit 15:7 保留，必须保持为 0

Bit 6 **HSE_LOCK**：HSE 时钟锁定状态标志寄存器

0：HSE 时钟锁定异常。

1：HSE 时钟锁定正常。

*注：用来指示芯片内部* PLL *的工作稳定状态，用户也可以打开* PLL *后等待* 4ms *来判断其稳定。*

Bit 5:3 保留，必须保持为 0

Bit 2 **HSE_FLAG**：外部高频 HSE 时钟 Fhse 停振标志

0：正常

1：停振

Bit 1:0 保留，必须保持为 0

### 6.3.7 系统时钟分频寄存器（CMU_SYSCLKDIV）

地址偏移：0x1C
复位值：0x0000 0001
访问：无等待周期，支持字，半字和字节访问
当连续对该寄存器进行访问时，将插入等待状态。

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | CPUCLK_DIV[2:0] | | |
| | | | | | | | | | | | | | rw | | |

Bit 15:3 保留，必须保持为 0

Bit 2:0 **CPUCLK_DIV[2:0]**：CPU 时钟分频设置

000：Fsys（Default） 001：Fsys/2
010：Fsys/4 011：Fsys/8
100：Fsys/16 101：Fsys/32
110：Fsys/64 111：Fsys/128

### 6.3.8 CLKOUT 时钟选择寄存器（CMU_CLKOUTSEL）

地址偏移：0x24
复位值：0x0000 0002
访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | CLKOUT_SEL[1:0] | |
| | | | | | | | | | | | | | | rw | rw |

Bit 15:2 保留，必须保持为 0

Bit 1:0 **CLKOUT_SEL[1:0]**：CLKOUT 时钟输出引脚配置

000：FLRC
001：FHSE
010：FHRC
011：FSYS

*注：1、用户可将芯片内部时钟源从 CLKOUT 引脚引出，以观测内部时钟。*
*2、用户可用 CLKOUTDIV 寄存器将内部时钟分频后引出，可作为外部设备的时钟源。*

### 6.3.9 CLKOUT 时钟分频寄存器（CMU_CLKOUTDIV）

地址偏移：0x28
复位值：0x0000 0000
访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | CLKOUT_DIV[3:0] | | | |
| | | | | | | | | | | | | rw | rw | rw | rw |

Bit 15:4 保留，必须保持为 0

Bit 3:0 **CLKOUT_DIV[3:0]**：CLKOUT 输出频率 $= \frac{\text{CLKOUT 选择的时钟源}}{2\times(\text{CLKOUT\_DIV[3...0]}+1)}$

### 6.3.10 内部模块使能控制寄存器 0（CMU_CLKCTRL0）

地址偏移：0x2C

复位值：0x0000 24E1

访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | OPA_EN | CMP5_EN | CMP4_EN | Reserved | | | | |
| | | | | | | | | rw | rw | rw | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | 1P5LBOR_EN | CLKOUT_EN | HSE_EN | HSE_Lock_EN | Reserved | | HSE_DET_EN | Reserved | HRC_EN | Reserved | I2C_EN | Reserved | | |
| | | rw | rw | rw | rw | | | rw | rw | rw | | rw | | | |

Bit 31:24 保留，必须保持为 0

Bit 23 **OPA_EN**：OPA 模块时钟使能

0：关闭(default)

1：开启

Bit 22 **CMP5_EN**：比较器 5 模块时钟使能

0：关闭(default)

1：开启

Bit 21 **CMP4_EN**：比较器 4 模块时钟使能

0：关闭(default)

1：开启

Bit 20:14 必须保持为 0

Bit 13 **1P5LBOR_EN**：LDO 内部 LBOR 使能位（用户不要修改这个寄存器位）

0：关闭

1：打开（default）

Bit 12 **CLKOUT_EN**：CLKOUT 使能信号

0：关闭（default）

1：打开

Bit 11 **HSE_EN**：HSE 使能信号

0：关闭（default）

1：打开

Bit 10 **HSE_Lock_EN**：HSR 强制锁定使能控制位

0：HSE 不强制锁定

1：HSE 强制锁定（default）

Bit 9:8 保留，必须保持为 0

Bit 7 **HSE_DET_EN**：HSE 时钟检测模块控制位

0：关闭 HSE 时钟停振检测单元

1：使能 HSE 时钟停振检测单元

Bit 6 保留，必须保持为 0

Bit 5 **HRC_EN**：HRC 时钟振荡器使能位

0：关闭高速振荡器时钟模块；

1：使能高速振荡器时钟模块；（default）

*注：当用户选择 FSYS 为 FHRC 时，此时不能关闭 HRC_EN，该寄存器位写入无效。此 bit 受写保护寄存器 WPREG 的控制。*

Bit 4 保留，必须保持为 0

Bit 3 **I2C_EN**：I2C 模块时钟使能位

0：关闭 I2C 模块（default）

1：使能 I2C 模块

Bit 2:0 保留，必须保持为 0

### 6.3.11 内部模块使能控制寄存器 1（CMU_CLKCTRL1）

地址偏移：0x30

复位值：0x0000 8000

访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | ADC_EN | ME_EN | TIM8_EN | Reserved | |
| | | | | | | | | | | | rw | rw | rw | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DMA_EN | SOFTWDT_EN | Reserved | | | | | | | | UART1_EN | Reserved | TMR3_EN | TMR2_EN | TMR1_EN | TMR0_EN |
| rw | rw | | | | | | | | | rw | | rw | rw | rw | rw |

Bit 31:21 保留，必须保持为 0

Bit 20 **ADC_EN**：ADC 模块使能

0：关闭(default)

1：开启

Bit 19 **ME_EN**：电机专用模块使能

0：关闭(default)

1：开启

Bit 18 **TIM8_EN**：Timer8 时钟模块使能

0：关闭(default)

1：开启

Bit 17:16 保留，必须保持为 0

Bit 15 **DMA_EN**：DMA 时钟模块使能

0：关闭(default)

1：开启

Bit 14 **SOFTWDT_EN**：调试模式下看门狗使能位

0：调试模式下看门狗关闭（default）

1：调试模式下看门狗打开

Bit 13:6 保留，必须保持为 0

Bit 5 **UART1_EN**：UART1 时钟使能位

0：关闭

1：使能

Bit 4 保留，必须保持为 0

Bit 3 **TMR3_EN**：Timer3 时钟使能

0：关闭

1：使能

Bit 2 **TMR2_EN**：Timer2 时钟使能

0：关闭

1：使能

Bit 1 **TMR1_EN**：Timer1 时钟使能

0：关闭

1：使能

Bit 0 **TMR0_EN**：Timer0 时钟使能

0：关闭

1：使能

### 6.3.12 FLASH 控制寄存器（CMU_FLASHCON）

地址偏移：0x34

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | BUSY | FOP[1:0] | |
| | | | | | | | | | | | | | r | rw | rw |

Bit 15:4 保留，必须保持为 0

Bit 15:0 **BUSY**：FLASH 忙标志位

0：表示 Flash 空闲，可以进行操作

1：表示 Flash 正在进行写/擦除操作

*注：只读状态寄存器位，写入无效。*

Bit 1:0 **FOP[1:0]**：FLASH 操作模式选择

00：处于 Flash 只读模式

01：对 STR/STRH 所指 FLASH 区执行 Flash 写操作

10：对 STR/STRH 所指 FLASH 区执行 Flash 页擦除操作

11：对 STR/STRH 所指 FLASH 区执行 Flash 全擦除操作

### 6.3.13 FLASH 锁定寄存器（CMU_FLASHLOCK）

地址偏移：0x38

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| KEY[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **KEY[15:0]**：Flash 锁定控制位

1、 对该寄存器写入 0x7A68 后，FLASH 被解锁，用户可以写操作 FLASH。

2、 写入非 0x7A68 数据后，FLASH 被锁定，用户禁止写操作 FLASH。

3、 默认为锁定状态，Flash 不可执行写/页擦除/全擦除操作。用户写入的是 0x7A68，读出值为 1；写入的是非 0x7A68,读出值为 0。

### 6.3.14 FLASH 访问周期配置寄存器（CMU_MULTFUNCFG）

地址偏移：0x204

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

*注：唤醒复位、调试和软复位、看门狗复位* BOR、EXRST、LBOR、POR *可以复位该寄存器。*

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

Bit 15:1 保留，必须保持为 0

Bit 0 **3T**：功能选择位

0：2T（Default）

1：开启

*注：开启* 3T，CPU *运行* 3T。*关闭* 3T *功能，不影响* CPU *正常运行。*

### 6.3.15 芯片版本寄存器（CMU_CHIPID）

地址偏移：0xF00

复位值：0x0000 8130

访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| REVID[15:0] | | | | | | | | | | | | | | | |
| r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CHIPID[15:0] | | | | | | | | | | | | | | | |
| r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

Bit 31:6 **REVID[15:0]**：改版编号

Bit 15:0 **CHIPID[15:0]**：芯片编号 0x8030

### 6.3.16 时钟滤波控制寄存器（CMU_FLTCTR）

地址偏移：0xF30

复位值：0x0000 0000

访问：0 到 2 个等待周期，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | HRC_CTR[1:0] | | LRC_CTR[1:0] | | HSE_CTR[1:0] | |
| | | | | | | | | | | rw | rw | rw | rw | rw | rw |

Bit 15:6 保留，必须保持为 0

Bit 5:4 **HRC_CTR[1:0]**：HRC 时钟滤波控制位

00：0.0ns（Default）

01：1.5ns

10：1.5ns

11：1.5ns

Bit 3:2 **LRC_CTR[1:0]**：LRC 时钟滤波控制位

00：0.0ns（Default）

01：1.5ns

10：3.0ns

11：4.5ns

Bit 1:0 **HSE_CTR[1:0]：** HSE 时钟滤波控制位

00：0.0ns（Default）

01：1.5ns

10：1.5ns

11：1.5ns

# 7 通用和复用功能 I/O（GPIO）

## 7.1 介绍

每个通用 I/O 有 1 个 16 位端口功能配置寄存器（GPIOx_IOCFG），1 个 16 位端口数据寄存器（GPIOx_PTDAT），1 个 32 位端口上下拉配置寄存器（GPIOx_PUPDN）和 1 个 16 位端口复位寄存器（GPIOx_PTCLR）。另外。所有 GPIO 有 1 个 16 位端口高阻态控制寄存器（GPIOx_HIIPM）和 2 个复用功能配置寄存器（GPIOx_AFCFG1 和 GPIOx_AFCFG2）。

## 7.2 GPIO 主要特征

- 输出状态：推挽或开漏 + 上拉/下拉
- 从数据寄存器（GPIOx_PTDAT）或外设（复用功能）输入输出数据
- I/O 速度可选
- 输入状态：浮空，上拉/下拉，模拟
- 模拟功能
- 复用功能选择寄存器
- 高度灵活的引脚复用允许使用 I/O 引脚作为 GPIO 或者几个外设复用功能之一

## 7.3 GPIO 功能复用

根据数据手册中列出的每个 I/O 端口的特定硬件特征，GPIO 端口的每个位可以由软件分别配置成多种模式：

- 输入浮空
- 输入上拉
- 输入下拉
- 模拟
- 上拉/下拉开漏输出
- 上拉/下拉推挽输出
- 上拉/下拉复用开漏输出
- 上拉/下拉复用推挽输出

每个 I/O 端口位可以自由编程，I/O 端口寄存器支持字、半字或字节访问。

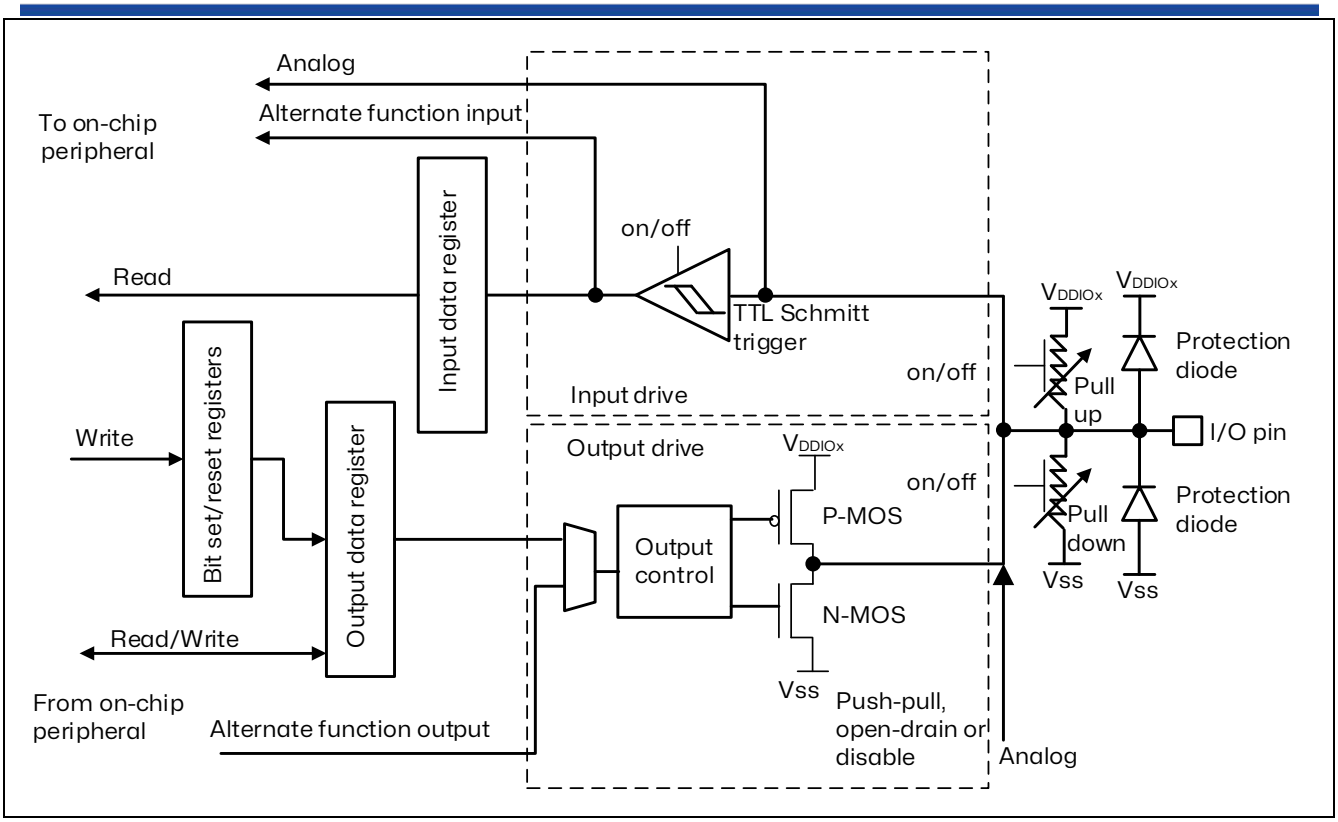



图 7.1 I/O 端口位基本结构

### 7.3.1 通用 IO（GPIO）

复位期间和刚复位后，复用功能未开启，I/O 端口被配置成浮空输入模式。

复位后，调试引脚被置于复用模式：

- PA0：SWIO 浮空
- PA1：SWCLK 浮空

当作为输出配置时，写到数据寄存器上的值（GPIOx_PTDAT）输出到相应的 I/O 引脚。

数据寄存器（GPIOx_PTDAT）在每个 APB 时钟周期捕捉 IO 引脚上的数据。

### 7.3.2 IO 引脚复用功能

芯片的 I/O 引脚通过一个多路复用器连接到外设模块，同时只能将一个外设复用功能连接到一个 I/O。因此，同一个引脚上的复用功能不会出现冲突。

每一个 I/O 引脚有一个多路复用器，可以通过 GPIOx_AFCFG1（引脚 0 到 7）GPIOx_AFCFG2（引脚 8 到 10）：

- 复位后，多路复用器选在复用功能 0（SEL0）。可以通过 GPIOx_IOCFG 寄存器将 I/O 配置成复用功能模式。
- 引脚的复用功能请参考芯片数据手册。

除了这种灵活的 I/O 复用架构之外，各外设还可以将复用功能映射到不同 I/O 引脚，这可以优化小型封装中可用外设的数量。

要将 I/O 配制成所需功能，请按照以下步骤操作：

- **调试功能：**在芯片复位后，会将这些引脚指定为专用引脚，可供调试器立即使用。
- **GPIO：**使用 GPIOx_IOCFG 寄存器将所需的 I/O 配置为 GPIO 或功能 PIN。

- **外设复用功能：**
  - 使用 GPIOx_AFCFG1 或 GPIOx_AFCFG2 将 I/O 连接到指定的复用功能上。
  - 使用 GPIOx_PTDIR，GPIOx_PUPDN 和 GPIOx_PTDAT 寄存器分别配置方向、上拉/下拉和数据输入输出。
  - 使用 GPIOx_IOCFG 寄存器将 I/O 配置为复用模式。

### 7.3.3 I/O 端口控制寄存器

每个 GPIO 端口有 2 个 32 位内存映射控制寄存器（GPIOx_AFCFG 和 GPIOx_PUPDN），可以配置最多 10 个 I/O。GPIOx_AFCFG 寄存器将 I/O 口配置到指定的复用功能上。GPIOx_PUPDN 寄存器用于选择任意 I/O 方向的上拉/下拉。

### 7.3.4 I/O 端口数据寄存器

每个 GPIO 端口有 1 个 16 位内存映射数据寄存器。GPIOx_PTDAT 存储通过 I/O 要输入输出的数据，输出数据可读可写，输入时只可读。此寄存器只在对应端口配置为 GPIO 功能或含有数字输入属性的复用功能时才有效。

### 7.3.5 I/O 复用功能输入/输出

提供了两个寄存器来为每个 I/O 选择一个可用的复用输入/输出。使用这些寄存器，用户可以根据应用程序的需要将复用功能连接到其他引脚。

使用 GPIOx_AFCFG1 和 GPIOx_AFCFG2 复用寄存器在每个 GPIO 上多路复用外设功能。因此，应用程序可以为每个 I/O 选择任意一个功能。AF 选择信号对于复用输入和复用输出是共性的，因此对于给定 I/O 的复用功能输入/输出选择单个通道。

具体 GPIO 的复用功能情况请参考数据手册。

### 7.3.6 外部中断/唤醒线

部分端口都具有外部中断功能。要使用外部中断功能，需要将端口配置为输入模式。

### 7.3.7 输入模式配置

当 I/O 端口被配置为输入时：

- 输出缓冲区被禁用
- Schmitt 触发器输入激活
- 根据 GPIOx_PUPDN 寄存器中的值，上拉和下拉电阻被激活
- 每个 APB 时钟周期，I/O 引脚上的数据被采样到的输入数据寄存器
- 输入数据寄存器提供 I/O 状态

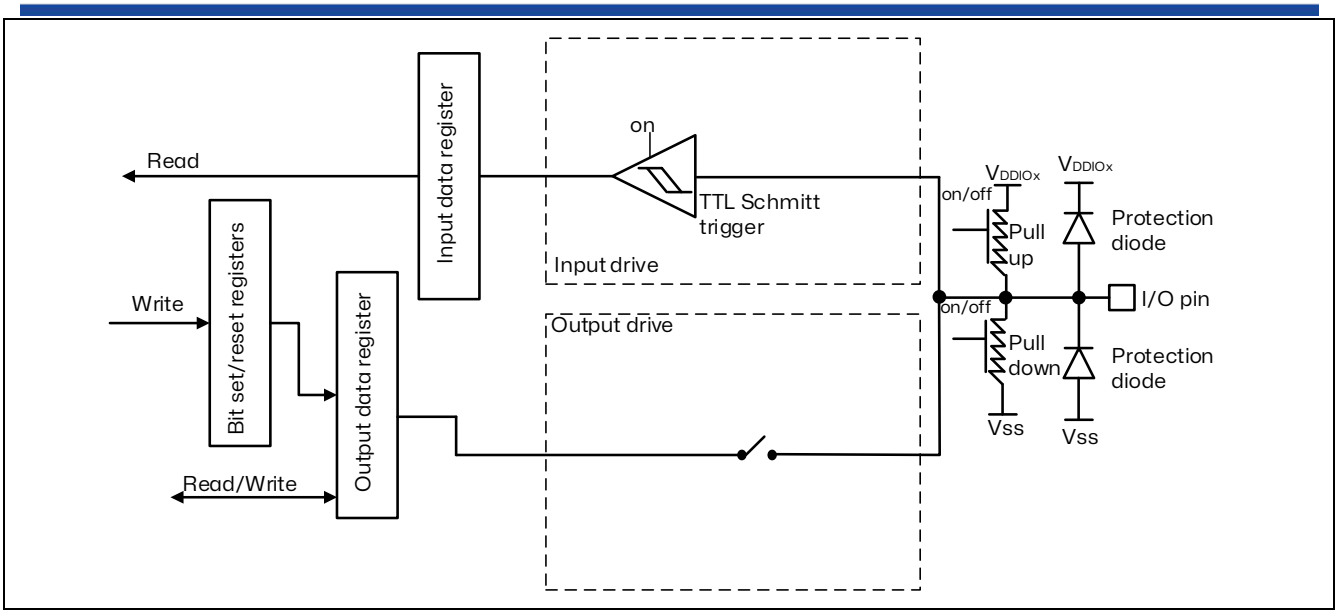



图 7.2 输入浮空/上拉/下拉结构

### 7.3.8 输出模式配置

当 I/O 端口被配置为输出时：

- 输出缓冲区已启用：
- 开漏模式：输出寄存器中的“0”开启 N-MOS，而输出寄存器中的“1”使端口处于 Hi-Z 状态（P-MOS 未被开启）
- 推挽模式：输出寄存器中的“0”开启 N-MOS，而输出寄存器中的“1”开启 P-MOS
- Schmitt 触发器输入激活
- 根据 GPIOx_PUPDN 寄存器中的值，上拉和下拉电阻被激活
- 每个 APB 时钟周期，I/O 引脚上的数据被采样到的输入数据寄存器
- 输入数据寄存器提供 I/O 状态
- 输出数据寄存器可读取最后写入的值

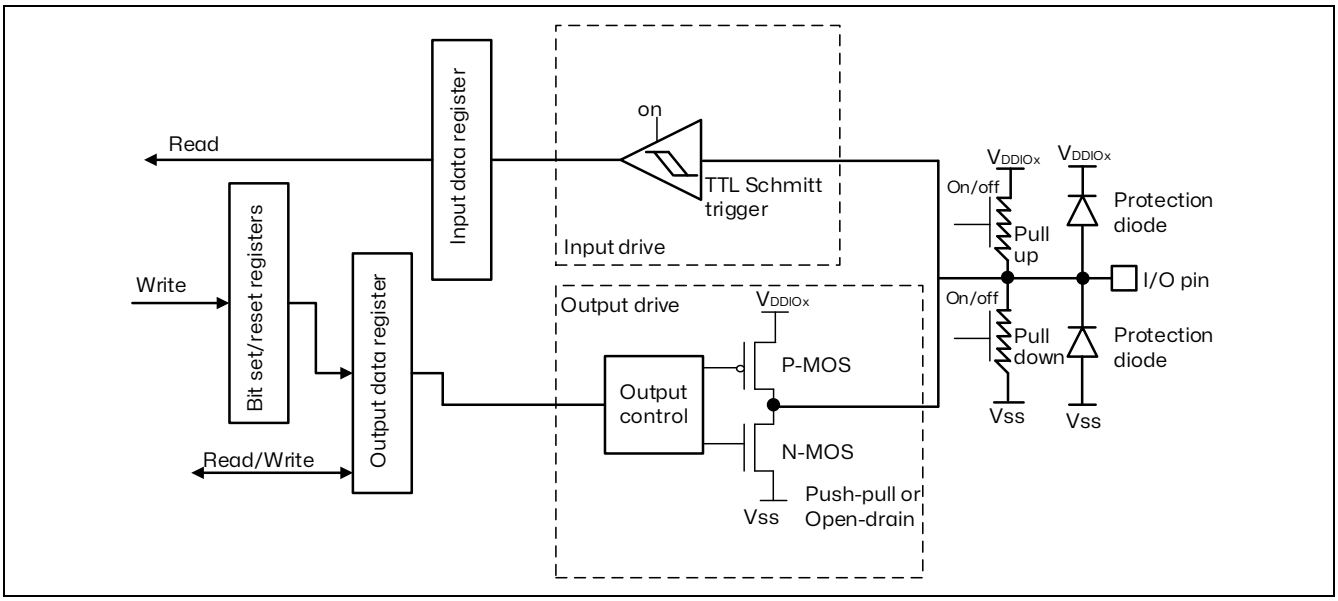



图 7.3 输出框图

### 7.3.9 复用功能配置

当 I/O 端口被配置为复用功能时：

- 输出缓冲区可以配置为开漏或推挽模式
- 输出缓冲区由来自外围设备的信号驱动（发射器使能和数据）
- Schmitt 触发器输入激活
- 根据 GPIOx_PUPDN 寄存器中的值，上拉和下拉电阻被激活
- 每个 APB 时钟周期，I/O 引脚上的数据被采样到的输入数据寄存器
- 输入数据寄存器提供 I/O 状态

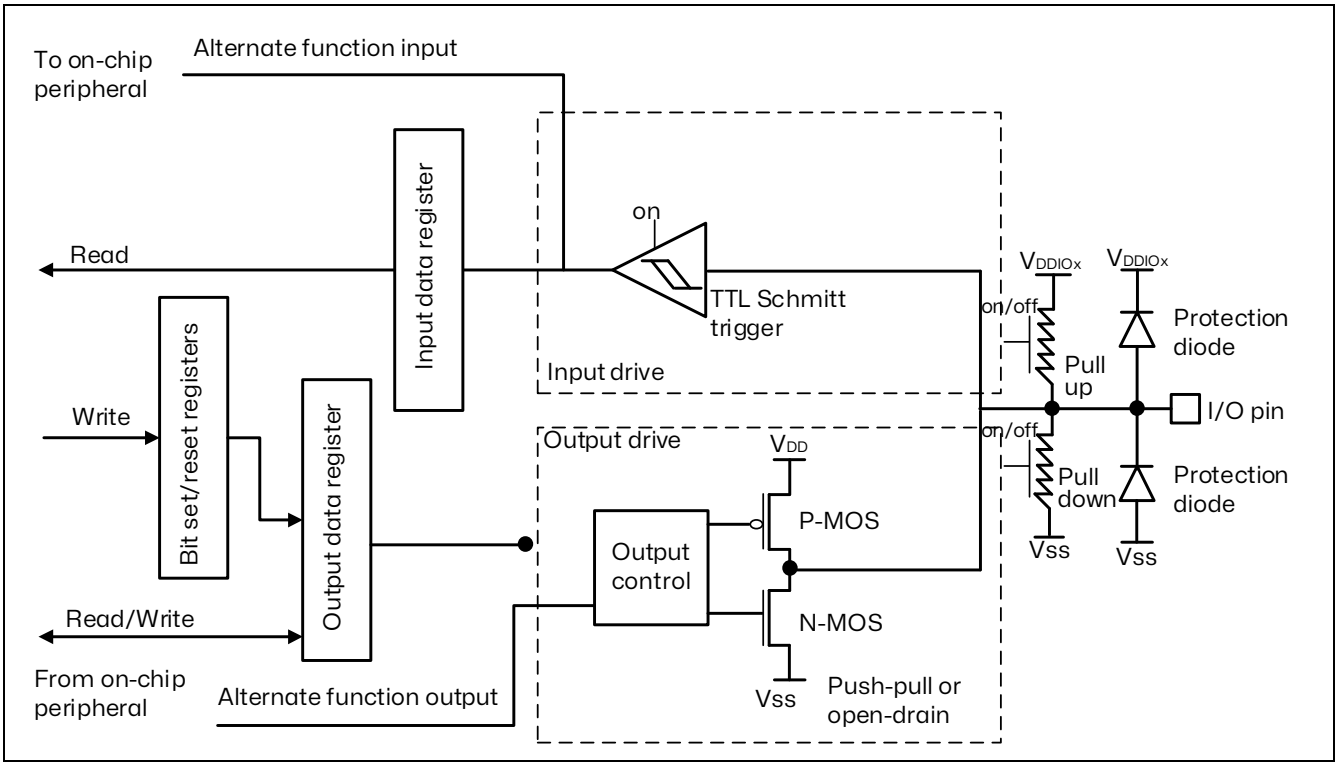



图 7.4 复用功能框图

### 7.3.10 模拟功能配置

当 I/O 端口配置为模拟配置时：

- 输出缓冲区被禁用
- Schmitt 触发输入是失活的，为 I/O 引脚的每个模拟值提供零消耗。Schmitt 触发器的输出被强制为恒定值（0）。
- 弱上拉被硬件禁用。弱下拉是可配置的。
- 输入数据寄存器读得到值固定为“0”

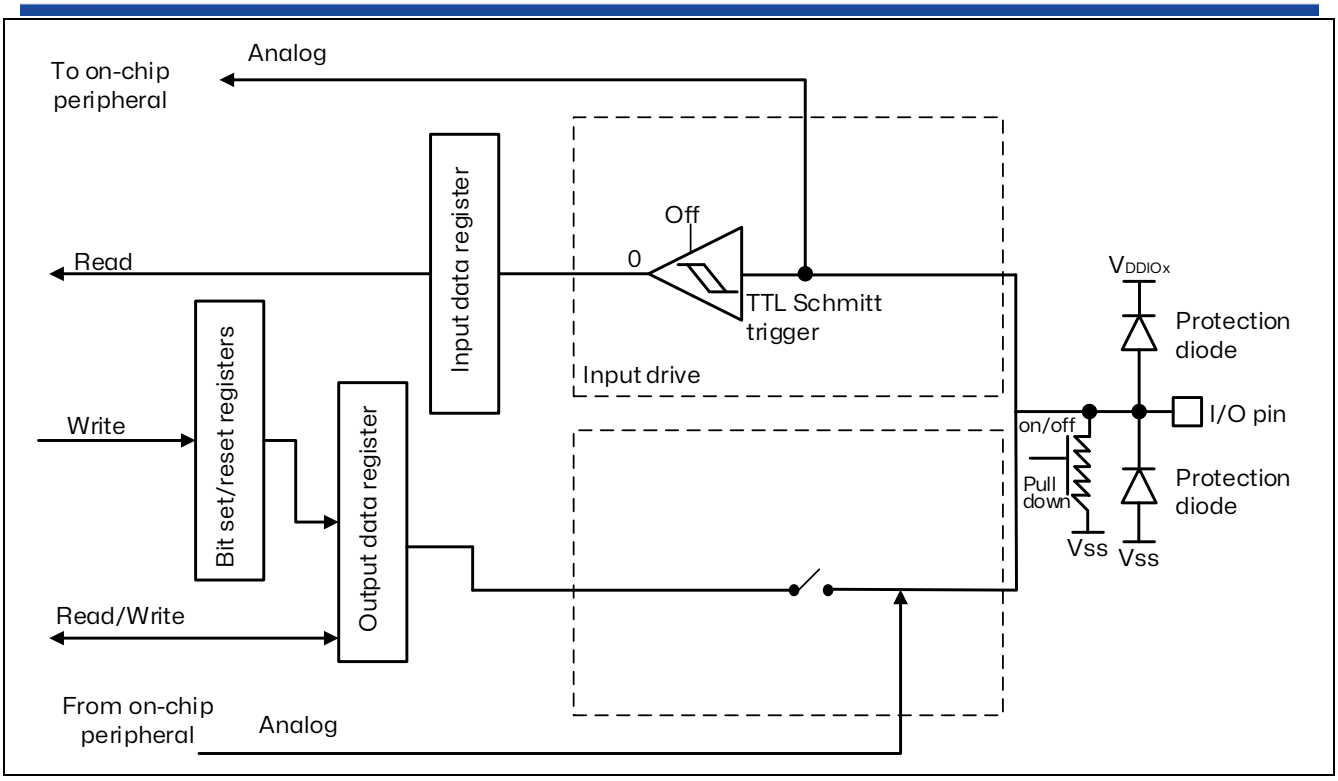



图 7.5 模拟引脚框图

## 7.4 GPIO 寄存器

### 7.4.1 端口功能配置寄存器 1（GPIOx_IOCFG）

地址偏移：0x00

复位值：0x0000 0003

访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PT15 | PT14 | PT13 | PT12 | PT11 | PT10 | PT9 | PT8 | PT7 | PT6 | PT5 | PT4 | PT3 | PT2 | PT1 | PT0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:16 保留，必须保持为复位值

Bit 15:0 **PTx：**端口 IO 功能配置位

0：对应的端口配置为 GPIO

1：对应的端口配置为功能 PIN

*注：*PA.0、PA.1 *的对应* bit *位默认为*1，*选择第一复用功能* SWD。

### 7.4.2 端口复用功能配置寄存器 1（GPIOx_AFCFG1）

地址偏移：0x04

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SEL7[1:0] | | SEL6[1:0] | | SEL5[1:0] | | SEL4[1:0] | | SEL3[1:0] | | SEL2[1:0] | | SEL1[1:0] | | SEL0[1:0] | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:16 保留，必须保持为复位值

Bit 15:0 **SELx[1:0]：**端口复用功能配置位（此寄存器只在对应端口配置为功能 PIN 时才有效。）

0：复用功能 1

1：复用功能 2

2：复用功能 3

3：复用功能 4

### 7.4.3 端口复用功能配置寄存器 2（GPIOx_AFCFG2）

地址偏移：0x30

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | SEL10[1:0] | | SEL9[1:0] | | SEL8[1:0] | |
| | | | | | | | | | | rw | rw | rw | rw | rw | rw |

Bit 31:6 保留，必须保持为复位值

Bit 5:0 **SELx[1:0]：**端口复用功能配置位（此寄存器只在对应端口配置为功能 PIN 时才有效）

0：复用功能 1

1：复用功能 2

2：复用功能 3

3：复用功能 4

### 7.4.4 端口方向配置寄存器（GPIOx_PTDIR）

地址偏移：0x08

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PT15 | PT14 | PT13 | PT12 | PT11 | PT10 | PT9 | PT8 | PT7 | PT6 | PT5 | PT4 | PT3 | PT2 | PT1 | PT0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:16 保留，必须保持为复位值

Bit 15:0 **PTx**：端口方向配置位（此寄存器只在对应端口配置为 GPIO 功能时才有效）

0：输入

1：输出

### 7.4.5 端口上下拉配置寄存器（GPIOx_PUPDN）

地址偏移：0x0C

复位值：0xFFFF FFFF（PA0 浮空，PA1 浮空）

访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PD15 | PD14 | PD13 | PD12 | PD11 | PD10 | PD9 | PD8 | PD7 | PD6 | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PT15 | PT14 | PT13 | PT12 | PT11 | PT10 | PT9 | PT8 | PT7 | PT6 | PT5 | PT4 | PT3 | PT2 | PT1 | PT0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:16 **PDx**：端口下拉配置位（此寄存器只在对应端口配置为数字输入时才有效）

0：使能下拉

1：禁止下拉（浮空），默认浮空输入

Bit 15:0 **PTx**：端口上拉配置位（此寄存器只在对应端口配置为数字输入时才有效）

0：使能上拉

1：禁止上拉（浮空），默认浮空输入

*注：数字输入含配置为输入模式的*GPIO *和具有数字输入属性的复用功能。*

### 7.4.6 端口数据寄存器（GPIOx_PTDAT）

地址偏移：0x10

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PT15 | PT14 | PT13 | PT12 | PT11 | PT10 | PT9 | PT8 | PT7 | PT6 | PT5 | PT4 | PT3 | PT2 | PT1 | PT0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:16 保留，必须保持为复位值

Bit 15:0 **PTx**：端口数据位（此寄存器只在对应端口配置为 GPIO 功能，或含有数字输入属性的复用功能时才有效）

当端口配置为输入时为读到的 IO 口状态
0：读到的为低电平
1：读到的为高电平
当端口配置为输出时
0：输出低电平
1：输出高电平
*注：端口数据寄存器 PTDAT 说明及数据读取规则：*
1）芯片引脚选择 GPIO 功能：若方向寄存器配置为输出，PTDAT 读取值为寄存器设置值，不随外部 PIN 脚电平变化而变化；若方向寄存器配置为输入，PTDAT 读取值为 pad 状态值，反映外部 PIN 脚电平变化；
2）芯片引脚选择复用数字功能：若复用为数字输出功能，PTDAT 读取值为切换成复用功能前的 GPIO 的 PTDAT 值，不随外部 PIN 脚电平变化而变化；若复用为数字输入功能，PTDAT 读取值为 pad 状态值，反映外部 PIN 脚电平变化；
3）芯片引脚选择复用模拟功能，PTDAT 相应 bit 位值，固定为 0。

### 7.4.7 端口设置寄存器（GPIOx_PTSET）

地址偏移：0x14
复位值：0x0000 0000
访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PT15 | PT14 | PT13 | PT12 | PT11 | PT10 | PT9 | PT8 | PT7 | PT6 | PT5 | PT4 | PT3 | PT2 | PT1 | PT0 |
| W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

Bit 15:0 **PTx：**端口设置位（此寄存器只在对应端口配置为 GPIO 且输出时才有效）
0：写 0 无效
1：写 1 将对应的端口输出高电平（同时更新 PTDAT 中对应的值）
*注：本寄存器只可写入。*

### 7.4.8 端口复位寄存器（GPIOx_PTCLR）

地址偏移：0x18
复位值：0x0000 0000
访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PT15 | PT14 | PT13 | PT12 | PT11 | PT10 | PT9 | PT8 | PT7 | PT6 | PT5 | PT4 | PT3 | PT2 | PT1 | PT0 |
| W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

Bit 15:0 **PTx：**端口复位位（此寄存器只在对应端口配置为 GPIO 且输出时才有效）
0：写 0 无效
1：写 1 将对应的端口输出低电平（同时更新 PTDAT 中对应的值）
*注：本寄存器只可写入。*

### 7.4.9 端口翻转寄存器（GPIOx_PTTOG）

地址偏移：0x1C

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PT15 | PT14 | PT13 | PT12 | PT11 | PT10 | PT9 | PT8 | PT7 | PT6 | PT5 | PT4 | PT3 | PT2 | PT1 | PT0 |
| W | W | W | W | W | W | W | W | W | W | W | W | W | W | W | W |

Bit 15:0 **PTx：**端口翻转位（此寄存器只在对应端口配置为 GPIO 且输出时才有效）

0：写 0 无效

1：写 1 将使对应的端口输出电平发生翻转（同时更新 PTDAT 中对应的值）

*注：本寄存器只可写入。*

### 7.4.10 端口开漏配置寄存器（GPIOx_PTOD）

地址偏移：0x20

复位值：0x0000 FFFF

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PT15 | PT14 | PT13 | PT12 | PT11 | PT10 | PT9 | PT8 | PT7 | PT6 | PT5 | PT4 | PT3 | PT2 | PT1 | PT0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **PTx：**端口开漏配置位（此寄存器只在对应端口配置为数字输出时才有效）

0：开漏功能使能（开漏输出，输出高为浮空，输出低为低）

1：开漏功能无效（推挽输出，输出高为高，输出低为低）

*注：数字输出含配置为输出模式的*GPIO *和具有数字输出属性的复用功能。*

### 7.4.11 端口高阻控制寄存器（GPIOx_HIIPM）

地址偏移：0x28

复位值：0x0000 FFFF

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| HIIPM15 | HIIPM14 | HIIPM13 | HIIPM12 | HIIPM11 | HIIPM10 | HIIPM9 | HIIPM8 | HIIPM7 | HIIPM6 | HIIPM5 | HIIPM4 | HIIPM3 | HIIPM2 | HIIPM1 | HIIPM0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **HIIPMx：**端口高阻配置位

0：高阻关闭

1：高阻开启(default)

*注：*I/O default *由自加载控制。*

*注：*PA0，PA1 *默认为*0

### 7.4.12 端口电平读取寄存器（GPIOx_IDR）

地址偏移：0x34

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IDR15 | IDR14 | IDR13 | IDR12 | IDR11 | IDR10 | IDR9 | IDR8 | IDR7 | IDR6 | IDR5 | IDR4 | IDR3 | IDR2 | IDR1 | IDR0 |
| r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

Bit 15:0 **IDRx：**读取 IO 电平状态

# 8 嵌套向量中断控制器（NVIC）

## 8.1 NVIC 主要特征

- 32 个可屏蔽中断通道
- 16 个可编程的优先等级（使用 4 位中断优先级）
- 低延迟的异常和中断处理
- 电源管理控制
- 系统控制寄存器的实现

嵌套向量中断控制器（NVIC）和处理器核的接口紧密相连，可以实现低延迟的中断处理和高效地处理晚到的中断。

## 8.2 中断和异常向量

表 8.1 RX32S11 向量表

| 位置 | 优先级 | 优先级类型 | 名称 | 说明 | 地址 |
|---|---|---|---|---|---|
| -14 | 1 | 可设置 | NMI | 硬件强制切换低频 RC 中断 | |
| -13 | 2 | 可设置 | HardFault | 故障/异常触发中断 | |
| -5 | 3 | 可设置 | SVCall | 软件触发中断 | |
| -2 | 4 | 可设置 | PendSV | 软件触发中断 | |
| -1 | 5 | 可设置 | SysTick | 系统嘀嗒定时器 | |
| 0 | 6 | 可设置 | PMU | 电源管理全局中断 | |
| 1 | 7 | 可设置 | ADC | ADC 中断 | |
| 2 | 8 | 可设置 | EXTI0 | EXTI 线 0 中断 | |
| 3 | 9 | 可设置 | EXTI1 | EXTI 线 1 中断 | |
| 4 | 10 | 可设置 | EXTI2 | EXTI 线 2 中断 | |
| 5 | 11 | 可设置 | EXTI3 | EXTI 线 3 中断 | |
| 6 | 12 | 可设置 | EXTI4 | EXTI 线 4 中断 | |
| 7 | 13 | 可设置 | EXTI5 | EXTI 线 5 中断 | |
| 8 | 14 | 可设置 | EXTI6 | EXTI 线 6 中断 | |
| 9 | 15 | 可设置 | | 保留 | |
| 10 | 16 | 可设置 | UART1 | UART1 全局中断 | |
| 11 | 17 | 可设置 | | 保留 | |
| 12 | 18 | 可设置 | | 保留 | |
| 13 | 19 | 可设置 | | 保留 | |
| 14 | 20 | 可设置 | ME | ME 全局中断 | |
| 15 | 21 | 可设置 | TMR0 | TMR0 全局中断 | |
| 16 | 22 | 可设置 | TMR1 | TMR1 全局中断 | |
| 17 | 23 | 可设置 | TMR2 | TMR2 全局中断 | |
| 18 | 24 | 可设置 | TMR3 | TMR3 全局中断 | |

| 位置 | 优先级 | 优先级类型 | 名称 | 说明 | 地址 |
|---|---|---|---|---|---|
| 19 | 25 | 可设置 | | 保留 | |
| 20 | 26 | 可设置 | RTC | RTC 全局中断 | |
| 21 | 27 | 可设置 | I2C | I2C 全局中断 | |
| 22 | 28 | 可设置 | | 保留 | |
| 23 | 29 | 可设置 | | 保留 | |
| 24 | 30 | 可设置 | | 保留 | |
| 25 | 31 | 可设置 | CMP4 | CMP4 全局中断 | |
| 26 | 32 | 可设置 | CMP5 | CMP5 全局中断 | |
| 27 | 33 | 可设置 | TIM8 | TIM8 全局中断 | |
| 28 | 34 | 可设置 | EXTI7 | EXTI 线 7 中断 | |
| 29 | 35 | 可设置 | EXTI8 | EXTI 线 8 中断 | |
| 30 | 36 | 可设置 | | 保留 | |
| 31 | 37 | 可设置 | DMA | DMA 全局中断 | |

# 9 中断和事件控制器（EXTI）

## 9.1 EXTI 简介

EXTI 主要特征如下：

- 支持 9 个中断请求
- 每个中断线都有专用的状态位
- 每个中断都有独立的使能和禁止
- 可配置上升沿和下降沿检测

## 9.2 框图

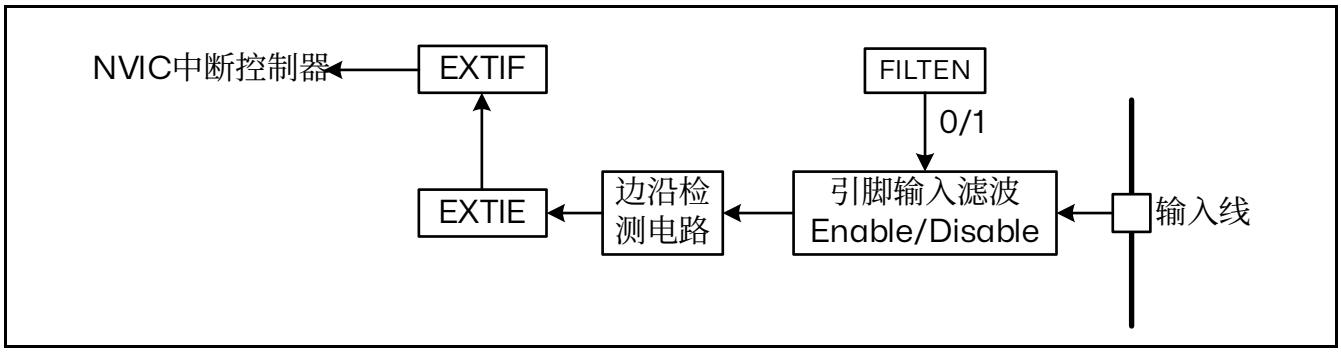



图 9.1 外部中断控制器框图

## 9.3 功能说明

外部中断控制器管理了 9 个中断线，每个中断线都对应有一个边沿检测器，可以实现输入信号的上升沿检测和下降沿的检测。EXTI 可以实现对每个中断线进行单独配置，可以单独配置成中断以及触发中断的属性。输入线可以通过寄存器设置为任意一个 GPIO，经过中断滤波使能寄存器(EXTI_FILTEN)来控制引脚输入滤波状态，边沿检测电路会根据上升沿和下降沿触发选择器(EXTI_EXTIE 和 EXTI_EXTIE2)对应位的设置来控制信号触发。输出的信号传送给中断标志位寄存器(EXTI_EXTIF 和 EXTI_EXTIF2)，由该寄存器决定是否产生中断信号，最后输出到 NVIC 中断控制器内，从而实现系统中断事件控制。

**硬件中断选择**

通过下面的过程来配置 9 个线路做为中断源：

- 将对应的 IO 口配置为 INT 复用功能
- 配置 9 个中断线的边沿触发选择位（EXTI_EXTIE 和 EXTI_EXTIE2）
- 配置所选中断线的中断滤波使能位（EXTI_FILTEN）
- 配置对应到外部中断控制器（EXTI）的 NVIC 中断通道的使能和禁止位，使得 9 个中断线中的请求可以被正确地响应。

**软件中断选择**

9 个线路可以被配置成软件中断线。下面是产生软件中断的过程：

- 配置 9 个中断边沿触发选择位（EXTI_EXTIE 和 EXTI_EXTIE2）
- 设置软件中断寄存器的请求位

## 9.4 EXTI 寄存器

### 9.4.1 外部中断边沿配置寄存器（EXTI_EXTIE）

地址偏移：0x00

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | RIE[6:0] | | | | | | | Reserved | FIE[6:0] | | | | | | |
| | rw | rw | rw | rw | rw | rw | rw | | rw | rw | rw | rw | rw | rw | rw |

Bit 15 保留，必须保持为复位值

Bit 14:8 **RIE[6:0]：**INT0-6 外部输入引脚上升沿使能

0：禁止

1：使能

*注：只有这些位置1，对应的中断标志才能置起来。*

Bit 7 保留，必须保持为复位值

Bit 6:0 **FIE[6:0]：**INT0-6 外部输入引脚下降沿使能

0：禁止

1：使能

*注：只有这些位置1，对应的中断标志才能置起来。*

### 9.4.2 外部中断标志寄存器（EXTI_EXTIF）

地址偏移：0x04

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | RIF[6:0] | | | | | | | Reserved | FIF[6:0] | | | | | | |
| | rw | rw | rw | rw | rw | rw | rw | | rw | rw | rw | rw | rw | rw | rw |

Bit 15 保留，必须保持为复位值

Bit 14:8 **RIF[6:0]：**INT 外部输入引脚上升沿中断标志

0：未产生中断

1：产生中断

Bit 7 保留，必须保持为复位值

Bit 6:0 **FIF[6:0]：**INT 外部输入引脚下降沿中断标志

0：未产生中断

1：产生中断

### 9.4.3 外部中断滤波使能寄存器（EXTI_FILTEN）

地址偏移：0x08

复位值：0x0000 01FF

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | INT8 FILTEN | INT7 FILTEN | INT6 FILTEN | INT5 FILTEN | INT4 FILTEN | INT3 FILTEN | INT2 FILTEN | INT1 FILTEN | INT0 FILTEN |
| | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:9 保留，必须保持为复位值

Bit 8 **INT8FILTEN**：INT8 50ns Filter 使能位
0：使能
1：关闭

Bit 7 **INT7FILTEN**：INT7 50ns Filter 使能位
0：使能
1：关闭

Bit 6 **INT6FILTEN**：INT6 50ns Filter 使能位
0：使能
1：关闭

Bit 5 **INT5FILTEN**：INT5 50ns Filter 使能位
0：使能
1：关闭

Bit 4 **INT4FILTEN**：INT4 50ns Filter 使能位
0：使能
1：关闭

Bit 3 **INT3FILTEN**：INT3 50ns Filter 使能位
0：使能
1：关闭

Bit 2 **INT2FILTEN**：INT2 50ns Filter 使能位
0：使能
1：关闭

Bit 1 **INT1FILTEN**：INT1 50ns Filter 使能位
0：使能
1：关闭

Bit 0 **INT0FILTEN**：INT0 50ns Filter 使能位
0：使能
1：关闭

### 9.4.4 外部中断串口数字滤波使能寄存器（EXTI_RXFILTEN）

地址偏移：0x0C

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | RXFILTEN_1 | Reserved |
| | | | | | | | | | | | | | | rw | |

Bit 15:2 保留，必须保持为复位值

Bit 1 **RXFILTEN_1：** UART RX 数字滤波使能

0：关闭

1：使能

Bit 0 保留，必须保持为复位值

### 9.4.5 外部中断边沿配置寄存器（EXTI_EXTIE2）

地址偏移：0x10

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | RIE[1:0] | | Reserved | | | | | | FIE[1:0] | |
| | | | | | | rw | rw | | | | | | | rw | rw |

Bit 15:10 保留，必须保持为复位值

Bit 9:8 **RIE[1:0]：** INT7-8 外部输入引脚上升沿中断使能

0：禁止

1：使能

*注：只有这些位置1，对应的中断标志才能置起来*

Bit 7:2 保留，必须保持为复位值

Bit 1:0 **FIE[1:0]：** INT7-8 外部输入引脚下降沿中断使能

0：禁止

1：使能

*注：只有这些位置1，对应的中断标志才能置起来*

### 9.4.6 外部中断标志寄存器 2（EXTI_EXTIF2）

地址偏移：0x14

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | RIF[1:0] | | Reserved | | | | | | FIF[1:0] | |
| | | | | | | rw | rw | | | | | | | rw | rw |

Bit 15:10 保留，必须保持为复位值

Bit 9:8 **RIF[1:0]：** INT7-8 外部输入引脚上升沿中断标志

0：未产生中断

1：产生中断

Bit 7:2 保留，必须保持为复位值

Bit 1:0 **FIF[1:0]：** INT7-8 外部输入引脚下降沿中断标志

0：未产生中断
1：产生中断

### 9.4.7 外部中断滤波选择寄存器（EXTI_FILTSEL）

地址偏移：0x1C
复位值：0x0003 FFFF
访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | INT8 FILTSEL[1:0] | |
| | | | | | | | | | | | | | | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INT7 FILTSEL[1:0] | | INT6 FILTSEL[1:0] | | INT5 FILTSEL[1:0] | | INT4 FILTSEL[1:0] | | INT3 FILTSEL[1:0] | | INT2 FILTSEL[1:0] | | INT1 FILTSEL[1:0] | | INT0 FILTSEL[1:0] | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:18 保留，必须保持为复位值

Bit 17:16 **INT8FILTSEL[1:0]：** INT8 Filter select
0X：80ns
1X：50ns

Bit 15:14 **INT7FILTSEL[1:0]：** INT7 Filter select
0X：80ns
1X：50ns

Bit 13:12 **INT6FILTSEL[1:0]：** INT6 Filter select
0X：80ns
1X：50ns

Bit 11:10 **INT5FILTSEL[1:0]：** INT5 Filter select
0X：80ns
1X：50ns

Bit 9:8 **INT4FILTSEL[1:0]：** INT4 Filter select
0X：80ns
1X：50ns

Bit 7:6 **INT3FILTSEL[1:0]：** INT3 Filter select
0X：80ns
1X：50ns

Bit 5:4 **INT2FILTSEL[1:0]：** INT2 Filter select
0X：80ns
1X：50ns

Bit 3:2 **INT1FILTSEL[1:0]：** INT1 Filter select
0X：80ns
1X：50ns

Bit 1:0 **INT0FILTSEL[1:0]：** INT0 Filter select
0X：80ns
1X：50ns

# 10 模拟数字转换器(ADC)

## 10.1 ADC 简介

12 位 ADC 是一种逐次逼近型模拟数字转换器。各通道的 A/D 转换可以单次、连续、扫描或间断模式执行。ADC 的结果可以左对齐或右对齐方式存储在 16 位数据寄存器中。

模拟看门狗特性允许应用程序检测输入电压是否超出用户定义的高/低阀值。

ADC 的输入时钟不得超过 14MHz，它是由 ADCCLK 产生。

## 10.2 ADC 主要特征

- 12 位分辨率
- 内建基准电压通道
- 单次和连续转换模式
- 从信道 0 到信道 n 的自动扫描模式
- ADC 最大时钟频率 14MHz，转换时间 1us
- 最高 15 信道，包括 11 个外部通道（ADC_IN0~10）和 4 个内部通道（PGA0~2，VBG）。
- 采样间隔可以按通道分别编程
- 1 个规则转换组，可任意设置规则序列长度和任意通道组合（序列长度最大为 8）
- 内建滑动平均滤波器专用于温度测量模块
- 触发方式可设置软件触发、RTC 触发、外部中断触发、内部 TMR&TIM8 触发
- ADC 供电要求：2.5V 到 5.5V

## 10.3 ADC 功能描述

下图为 ADC 模块框图

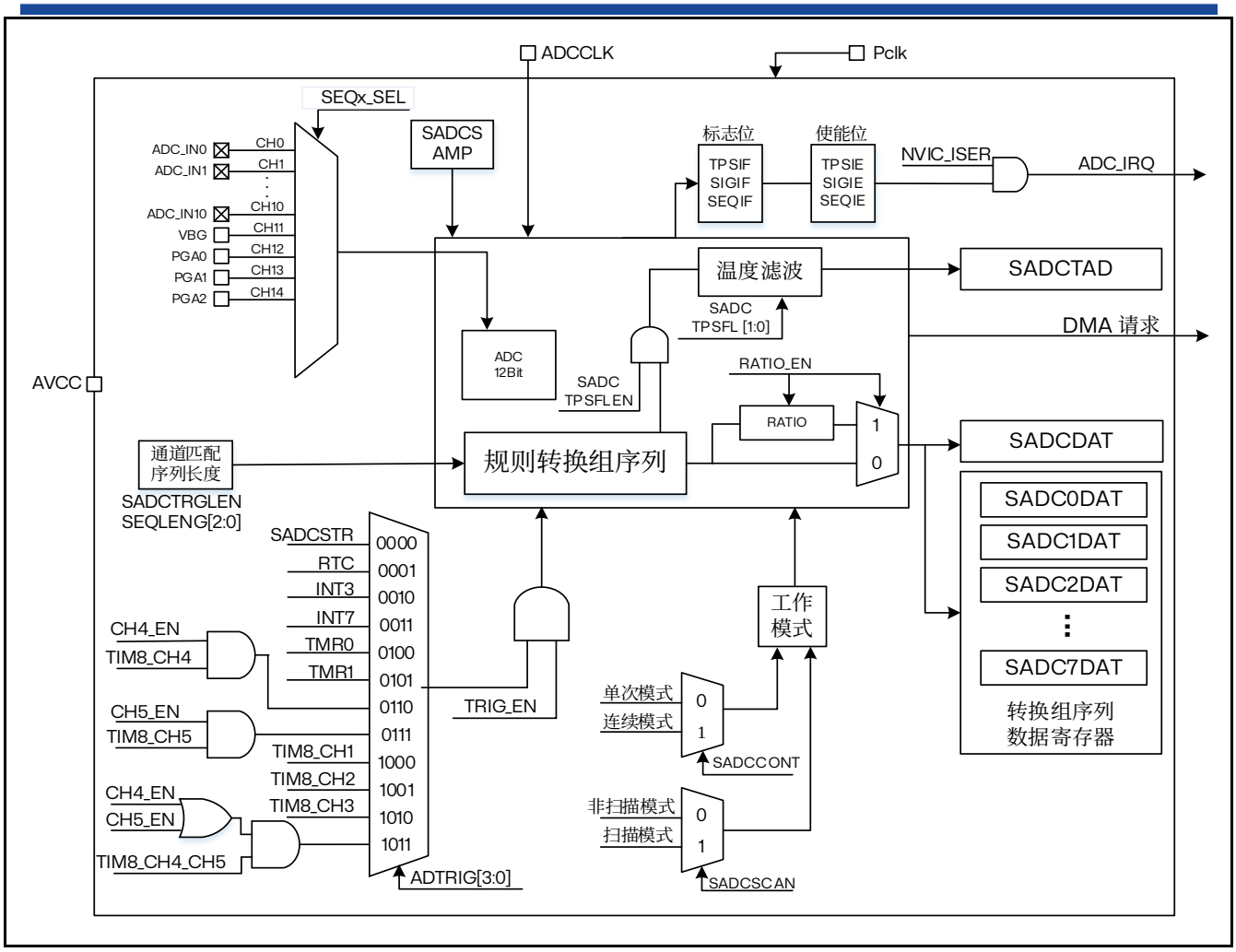



图 10.1 ADC 框图

### 10.3.1 ADC 引脚/通道

表 10.1 ADC 引脚

| 引脚 | 信号类型 | 注解 |
|---|---|---|
| VREF+ | 输入，模拟参考正极 | ADC 的正极参考电压，2.5 V ≤VREF+ ≤ VDDA |
| VDDA | 输入，模拟电源 | 等效于 VDD 的模拟电源，VDDA = VDD |
| ADCx_IN[m:0] | 模拟输入信号 | 模拟输入通道 |

表 10.2 ADC 通道

| 通道编号 | 模拟信号来源 |
|---|---|
| 通道 0 | PC4 |
| 通道 1 | PC3 |
| 通道 2 | PC1 |
| 通道 3 | PC0 |
| 通道 4 | PB8 |
| 通道 5 | PB7 |

| 通道编号 | 模拟信号来源 |
|---|---|
| 通道 6 | PB5 |
| 通道 7 | PA7 |
| 通道 8 | PA6 |
| 通道 9 | PA5 |
| 通道 10 | PA4 |
| 通道 11 | VBG |
| 通道 12 | PGA0 |
| 通道 13 | PGA1 |
| 通道 14 | PGA2 |

### 10.3.2 ADC 开关

SAR ADC 模块的开关可以控制。当设置 SADCON=1 时，将 ADC 从关闭状态唤醒，为 A/D 转换做准备；当设置 SADCON=0 时，ADC 模块处于关闭状态，降低功耗。

SADCEN 控制模块的时钟，当 SADCEN=1 时，开启模块时钟，SAR ADC 按照配置进行转换；当 SADCEN=0 时，关闭模块时钟，SAR ADC 停止转换；

### 10.3.3 ADC 时钟

由时钟控制器提供的 ADCCLK 时钟同步 PCLK。RCC 控制器为 ADC 时钟提供一个可选时钟分频。

### 10.3.4 通道选择

ADC 模块包含 1 个规则转换序列，序列长度最大为 8 个，在 ADC_IN0-10/PGA0~2/VBG 这 15 个通道任意组合选择，可向序列信道配置寄存器 SADCSEQCFG 中 SEQx_SEL[3:0],写入通道号（其中 x=0~14），从而设置序列中通道转换顺序。

### 10.3.5 单次转换模式

单次转换模式下，ADC 只执行一次转换。若配置了扫描模式，则执行规则组序列的每一次转换。在单次转换模式下，未使能扫描模式，信道序列 0 配置了 ADC_IN3，每次触发 ADC 转换只会对 ADC_IN3 通道进行一次转换，转换完成后自动结束，此时 SADCSTR 被清 0。在单次转换模式下，使能了扫描模式，设定转换组序列 0~2 为 ADC_IN3、ADC_IN7、ADC_IN5，则每次触发 ADC 转换，自动转换 ADC_IN3、ADC_IN7、ADC_IN5 通道，转换完成后自动结束，如果转换期间 SADC_TRIG_EN 被清 0，则当前序列 n 转换完成后，自动停止。下次发生触发 ADC 转换事件时，转换从序列 0 开始。

单次转换模式下，每次转换完成后：

- 转换数据储存在数据寄存器中
- 转换结束标志置 1
- 如果设置了转换完成中断，则产生转换完成中断。

然后 ADC 停止。

### 10.3.6 连续转换模式

在连续转换模式中，当前面 ADC 转换一结束马上就启动另一次转换。如果转换期间 SADC_TRIG_EN 被清 0，完成当前转换后，自动停止转换，再次发生触发事件时，转换从序列 0 开始。将 SADCCONT 写 0，完成本次规则组序列转换后也会停止。

在连续模式下，每次转换后：

- 转换数据储存在数据寄存器中
- 转换结束标志置 1
- 如果设置了转换完成中断，则产生转换完成中断
- 启动下一次转换

### 10.3.7 扫描模式

扫描模式开启时，扫描序列长度配置为 n(SADCTRGLEN.SEQLENG=n)，其中 0≤n≤7。当发生触发 ADC 事件，从规则转换组的序列 0 依次转换序列 0~n 中的通道。

扫描模式和单次模式组合时，当发生触发 ADC 事件，从规则转换组的序列 0 依次转换序列 0~n 中的通道，完成序列 n 的通道转换后自动停止。扫描模式和连续模式组合时，当发生触发 ADC 事件，从规则转换组的序列 0 依次转换序列 0~n 中的通道，完成序列 n 的通道转换后自动重新从序列 0 依次转换。

## 10.4 校准

ADC 有一个内置自校准模式。校准可大幅减小因内部电容器组的变化而造成的准精度误差。在校准期间，在每个电容器上都会计算出一个误差修正码（数字值），这个码用于消除在随后的转换中每个电容器上产生的误差。

通过设置 ADC_SADCCON 寄存器的 CAL 位启动校准。一旦校准结束，CAL 位被硬件复位，可以开始正常转换。建议在上电时执行一次 ADC 校准。校准阶段结束后，校准码储存在 ADC_SADCDAT 中。

*注意：建议在每次上电后执行一次校准。*

*启动校准前，ADC 必须处于关电状态，超过至少两个 ADC 时钟周期。*

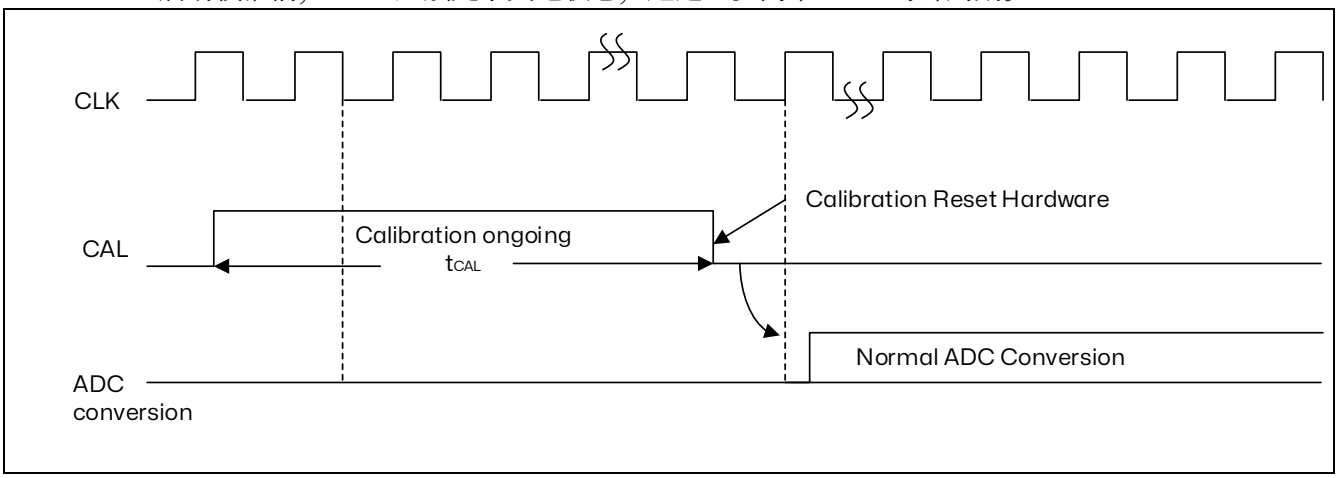



图 10.2 校准时序图

## 10.5 可编程的通道采样时间

ADC 使用若干个 ADC_CLK 周期对输入电压采样，采样周期数目可以通过 ADC_SADCSAMP 和 ADC_SADCSAMP2 寄存器中的 SAMP[3:0]位更改。每个通道可以分别用不同的时间采样。

总转换时间如下计算：

$T_{CONV}$ = 采样时间 + 12.5 个周期

例如：

当 ADCCLK = 14MHz，采样时间为 1.5 周期

$T_{CONV}$ = 1.5 + 12.5 = 14 个周期 = 1us

## 10.6 ADC 寄存器

### 10.6.1 SADC 控制寄存器（ADC_SADCCON）

地址偏移：0x00

复位值：0x0000 0402

访问：无等待，支持字，半字和字节访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SADC EN | Reserved | | | | | | | | | SADC CONT | SADC SCAN | Reserved | SADC_TRIG_EN | Reserved | SADC ON |
| rw | | | | | | | | | | rw | rw | | rw | | rw |

Bit 31:17 保留，必须保持为复位值

Bit 16 **DISCEN**：间断模式使能位：

1：间断模式使能

0：间断模式关闭

Bit 15 **SADCEN**：SADC 时钟控制位：

0：SADC 时钟关闭（default）

1：SADC 时钟开启

Bit 14:6 保留，必须保持为复位值

Bit 5 **SADCCONT**：SADC 连续转换模式控制位：

0：单次转换模式（default）

1：连续转换模式

Bit 4 **SADCSCAN**：SADC 扫描模式控制：

0：关闭扫描模式（default）

1：开启扫描模式

Bit 3 保留，必须保持为复位值

Bit 2 **SADC_TRIG_EN**： SADC 触发使能控制位：

0：触发使能关闭（default）

1：触发使能开启

Bit 1 保留，必须保持为复位值

Bit 0 **SADCON**：SADC 开关控制位：

0：SADC 关闭（default

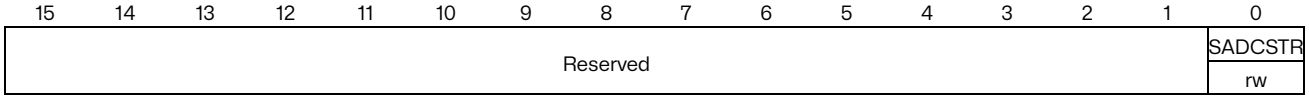
1：SADC 开启

### 10.6.2 SADC 序列转换控制寄存器（ADC_SADCSTR）

地址偏移：0x0C

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问



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

Bit 15:1 保留，必须保持为复位值

Bit 0 **SADCSTR**：SADC 序列转换触发控制位：

0：停止序列转换（default）

1：开启序列转换

*注：软件写 1，硬件自动清 0*

### 10.6.3 SADCCLK 时钟配置寄存器（ADC_SADCCLK）

地址偏移：0x04

复位值：0x0000 0002

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | SADCCLK_DIV[2:0] | | |
| | | | | | | | | | | | | | rw | | |

Bit 15:3 保留，必须保持为复位值

Bit 2:0 **SADCCLK_DIV[2:0]**：SADC 时钟分频控制位：

ADC 时钟计算公式：Fsadc= Fsys/SADCCLK_DIV

000：1 分频

001：2 分频

010：3 分频（default）

011：4 分频

100：5 分频

101：6 分频

110：7 分频

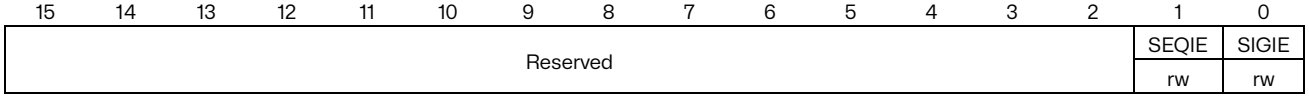
111：8 分频

### 10.6.4 SADC 中断控制寄存器（ADC_SADCIE）

地址偏移：0x10

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问



| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | SEQIE | SIGIE |
| | | | | | | | | | | | | | | rw | rw |

Bit 15:2 保留，必须保持为复位值

Bit 1 **SEQIE**：SADC 扫描序列转换完成中断使能控制位：

0：不使能序列转换完成中断（default）

1：使能序列转换完成中断

Bit 0 **SIGIE**：SADC 单次转换完成中断使能控制位：

0：不使能单次转换完成中断（default）

1：使能单次转换完成中断

*注：*（1）*单信道单次模式：只会产生* SIGIE *中断；*

（2）*多信道单次模式：每个通道转换一次，会产生* SIGIE *中断，整个序列转换完成，产生* SEQIE *中断；*

（3）*单信道连续模式：只会产生* SIGIE *中断；*

（4）*多通道连续模式：每个通道转换一次，会产生* SIGIE *中断，整个序列转换完成一次，产生一次* SEQIE *中断。*

### 10.6.5 SADC 中断标志寄存器（ADC_SADCIF）

地址偏移：0x14

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | SEQIF | SIGIF |
| | | | | | | | | | | | | | | rw | rw |

Bit 15:2 保留，必须保持为复位值

Bit 1 **SEQIF**：SADC 扫描序列转换完成中断标志位：

规则组序列转换完成标志置 1，若使能序列转换完成中断，则产生中断

Bit 0 **SIGIF**：SADC 单次转换完成中断标志位：

规则组序列中一个转换完成标志置 1，若使能单次转换完成中断，则产生中断

### 10.6.6 SADC 采样时间配置寄存器（ADC_SADCSAMP）

地址偏移：0x18

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH7_SAMP[3:0] | | | | CH6_SAMP[3:0] | | | | CH5_SAMP[3:0] | | | | CH4_SAMP[3:0] | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CH3_SAMP[3:0] | | | | CH2_SAMP[3:0] | | | | CH1_SAMP[3:0] | | | | CH0_SAMP[3:0] | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:0 **CHx_SAMP[3:0]**：模拟输入通道 x 采样时间（x=0~7）

0000：2Fsadc（default）

0001：4Fsadc

0010：8Fsadc

0011：16Fsad

0100：32Fsadc

0101：64Fsadc

0110：128Fsad

0111：256Fsadc

1xxx：X

*注：*CH0_SAMP *对应* ADC_IN0 *通道的采样时间；*CH1_SAMP *对应* ADC_IN1 *通道的采样时间；……；*

### 10.6.7 SADC 采样时间配置寄存器（ADC_SADCSAMP2）

地址偏移：0x1C

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | CH14_SAMP[3:0] | | | | CH13_SAMP[3:0] | | | | CH12_SAMP[3:0] | | | |
| | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH11_SAMP[3:0] | | | | CH10_SAMP[3:0] | | | | CH9_SAMP[3:0] | | | | CH8_SAMP[3:0] | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:0 **CHx_SAMP[3:0]：**模拟输入通道 x 采样时间（x=8~14）

0000：2Fsadc（default）

0001：4Fsadc

0010：8Fsadc

0011：16Fsad

0100：32Fsadc

0101：64Fsadc

0110：128Fsad

0111：256Fsadc

1xxx：X

### 10.6.8 SADC 触发源及扫描序列长度寄存器（ADC_SADCTRGLEN）

地址偏移：0x20

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | SEQLENG[2:0] | | | Reserved | | CH4_EN | CH5_EN | ADTRIG[3:0] | | | |
| | | | | | rw | rw | rw | | | rw | rw | rw | rw | rw | rw |

Bit 15:11 保留，必须保持为复位值

Bit 10:8 **SEQLENG[2:0]：**规则组转换扫描序列长度控制位：(设置范围为 000~111)

000：扫描序列长度为 1（default）

001：扫描序列长度为 2

...

111：扫描序列长度为 8

Bit 7:6 保留，必须保持为复位值

Bit 5 **CH4_EN：**TIM8 CH4 input enable

0：TIM8 CH4 触发关闭

1：TIM8 CH4 触发使能

Bit 4 **CH5_EN：**TIM8 CH5 input enable

0：TIM8 CH5 触发关

1：TIM8 CH5 触发使能

Bit 3:0 **ADTRIG[3:0]：**ADC 触发源选择位

0000：软件触发（SADCSTR）（default）

0001：RTC2 周期定时完成触发

0010：外部中断 INT3 触发
0011：外部中断 INT7 触发
0100：TMR0 定时触发
0101：TMR1 定时触发
0110：TIM8_CH4 定时触发
0111：TIM8_CH5 定时触发
1000：TIM8_CH1 触发
1001：TIM8_CH2 触发
1010：TIM8_CH3 触发
1011：TIM8_CH4_CH5 触发
*注：选择 000 为软件触发，屏蔽硬件触发；若选择硬件触发，则软件触发仍然有效。*

### 10.6.9 SADC 序列信道配置寄存器（ADC_SADCSEQCFG）

地址偏移：0x24
复位值：0x0000 0000
访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SEQ7_SEL[3:0] | | | | SEQ6_SEL[3:0] | | | | SEQ5_SEL[3:0] | | | | SEQ4_SEL[3:0] | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SEQ3_SEL[3:0] | | | | SEQ2_SEL[3:0] | | | | SEQ1_SEL[3:0] | | | | SEQ0_SEL[3:0] | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:0 **SEQx_SEL[3:0]：**规则转换组序列中第 x 个转换通道选择（x=0-14）

0：AD0
1：AD1
2：AD2
3：AD3
4：AD4
5：AD5
6：AD6
7：AD7
8：AD8
9：AD9
10：AD10
11：AD11
12：PGA0
13：PGA1
14：PGA2

### 10.6.10 SADC 序列数据寄存器（ADC_SADCDAT）

地址偏移：0x2C

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | SADCDAT[19:16] | | | |
| | | | | | | | | | | | | r | r | r | r |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SADCDAT[15:8] | | | | | | | | SADCDAT[7:0] | | | | | | | |
| r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

Bit 31:20 保留，必须保持为复位值

Bit 19:0 **SADCDAT[19:0]：**规则组序列数据寄存器

*注：*16bit *有符号数，只读，所有规则组序列通道的转换结果*

*实际转换值* 12 *位计算公式为*（Vin/Avcc）*4096，Bit16-bit19 *显示转换* CHANEL *值*, Bit13-bit0 *显示* 12 *位转换值得* 2 *补数，*bit14-15 *补齐符号位*

### 10.6.11 SADCx 通用数据寄存器（ADC_SADCxDAT）

地址偏移：0x30+4*x（x=0~7）

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | SADCxDAT[19:16] | | | |
| | | | | | | | | | | | | r | r | r | r |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SADCxDAT[15:8] | | | | | | | | SADCxDAT[7:0] | | | | | | | |
| r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

Bit 31:20 保留，必须保持为复位值

Bit 15:0 **SADCxDAT[19:0]：**规则组序列 x 数据寄存器

*注：*16bit *有符号数，只读，所有规则组序列通道的转换结果*

*实际转换值*12 *位计算公式为*（Vin/Avcc）*4，Bit16-bit19 *显示转换*CHANEL *值*, Bit13-bit0 *显示*12 *位转换值得*2 *补数，*bit14-15 *补齐符号位*

### 10.6.12 CALDATx 校准值寄存器（ADC_SADC_CALDATx）

地址偏移：0x74+(x*4) （x=0~4）

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SADC_CALDATx[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **SADC_CALDATx[15:0]:**

x = 0，PGA0 offset (signed)

x = 1，PGA1 offset (signed)

x = 2，PGA2 offset (signed)

x = 3，ADC offset (signed)

# 11 直接存储器访问控制器（DMA）

## 11.1 DMA 简介

直接存储器存取（DMA）用来提供在外设和存储器之间或者存储器和存储器之间的高速数据传输。无须 CPU 干预，数据可以通过 DMA 快速地移动，这就节省了 CPU 的资源来做其他操作。

一个 DMA 控制器有 2 个通道，每个通道专门用来管理来自于一个或多个外设对存储器访问的请求。还有一个仲裁器来协调各个 DMA 请求的优先权。

### 11.1.1 DMA 主要特征

- 2 个独立可配置的通道（请求）
- 每个通道都直接连接专用的硬件 DMA 请求，每个通道都同样支持软件触发。这些功能通过软件来配置。
- 在同一个 DMA 模块上，多个请求间的优先权可以通过软件编程设置（共有四级：很高、高、中等和低），优先权设置相等时由硬件决定（请求 0 优先于请求 1，依此类推）。
- 独立数据源和目标数据区的传输宽度（字节、半字、全字），模拟打包和拆包的过程。源和目标地址必须按数据传输宽度对齐。
- 支持循环的缓冲器管理。
- 每个通道都有 3 个事件标志（DMA 块传输、DMA 传输完成和 DMA 传输出错），这 3 个事件标志逻辑或成为一个单独的中断请求。
- 存储器和存储器间的传输。
- 外设和存储器、存储器和外设之间的传输。
- 可编程的数据传输数目：最大为 65535。

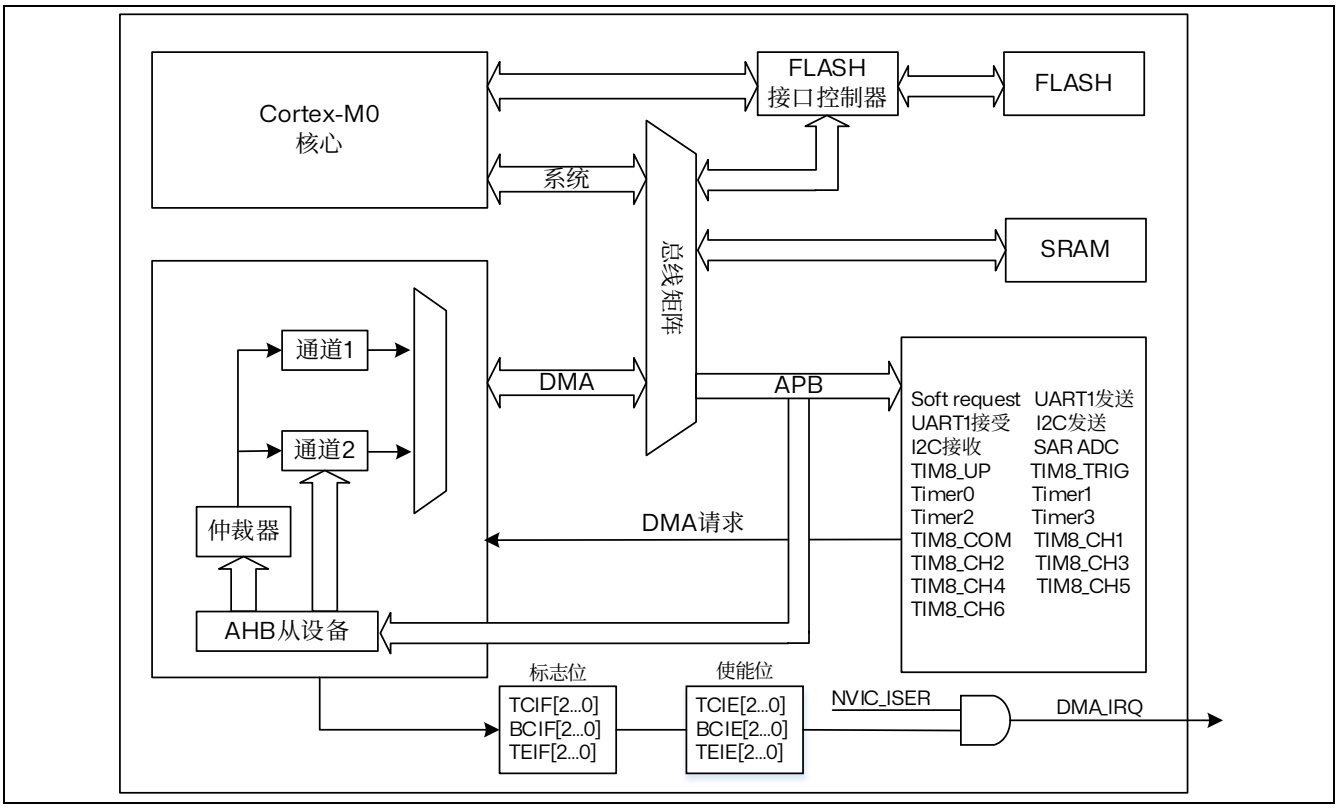



图 11.1 DMA 框图

## 11.2 功能描述

DMA 控制器和 Cortex®-M0 核心共享系统数据总线，执行直接存储器数据传输。当 CPU 和 DMA 同时访问相同的目标（RAM 或外设）时，DMA 请求会暂停 CPU 访问系统总线达若干个周期，总线仲裁器执行循环调度，以保证 CPU 至少可以得到一半的系统总线（存储器或外设）带宽。

## 11.3 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 控制器立即发送给它一个应答信号。当从 DMA 控制器得到应答信号时，外设立即释放它的请求。一旦外设释放了这个请求，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周期。

总之，每次 DMA 传送由 3 个操作组成：

- 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取数据，第一次传输时的开始地址是 DMA_CHNxSRC 或 DMA_CHNxTAR 寄存器指定的外设基地址或存储器单元。
- 存数据到外设数据寄存器或者当前外设/存储器地址寄存器指示的存储器地址，第一次传输时的开始地址是 DMA_CHNxSRC 或 DMA_CHNxTAR 寄存器指定的外设基地址或存储器单元。
- 执行一次 DMA_CHNxCNT 寄存器的递减操作，该寄存器包含未完成的操作数目。

### 11.3.1 仲裁器

仲裁器根据通道请求的优先级来启动外设/存储器的访问。

优先权管理分 2 个阶段：

- 软件：每个通道的优先权可以在 DMA_CHNxCTL 寄存器中设置，有 4 个等级：
  - 最高优先级
  - 高优先级
  - 中等优先级
  - 低优先级
- 硬件：如果 2 个请求有相同的软件优先级，则较低编号的通道比较高编号的通道有较高的优先权。举个例子，通道 2 优先于通道 4。

### 11.3.2 DMA 通道

每个通道都可以在有固定地址的外设寄存器和存储器地址之间执行 DMA 传输。DMA 传输的数据量是可编程的，最大达到 65535。包含要传输的数据项数量的寄存器，在每次传输后递减。

**可编程的数据量**

外设和存储器的传输数据量可以通过 DMA_CHNxCTL 寄存器中的 PSIZE[1:0]位编程。

**指针增量**

通过设置 DMA_CHNxCTL 寄存器中的 SOURC_INC[1:0]和 DESTIN_INC[1:0]标志位，外设和存储器的指针在每次传输后可以有选择地完成自动增量。当设置为增量模式时，下一个要传输的地址将是前一个地址加上增量值，增量值取决与所选的数据宽度为 1、2 或 4。在传输过程中，这些寄存器保持它们初始的数值，软件不能改变和读出当前正在传输的地址（它在内部的当前外设/存储器地址寄存器中）。

当通道配置为非循环模式时，传输结束后（即传输计数变为 0）将不再产生 DMA 操作。要开始新的

DMA 传输，需要在关闭 DMA 通道的情况下，在 DMA_CHNxCNT 寄存器中重新写入传输数目。

在循环模式下，最后一次传输结束时，DMA_CHNxCNT 寄存器的内容会自动地被重新加载为其初始数值，内部的当前外设/存储器地址寄存器也被重新加载为 DMA_CHNxSRC/DMA_CHNxTAR 寄存器设定的初始基地址。

**通道配置过程**

下面是配置 DMA 通道 x 的过程（x 代表通道号）：

1. 在 DMA_CHNxSRC 寄存器中设置外设寄存器的地址。发生外设数据传输请求时，这个地址将是数据传输的源或目标。
2. 在 DMA_CHNxTAR 寄存器中设置数据存储器的地址。发生外设数据传输请求时，传输的数据将从这个地址读出或写入这个地址。
3. 在 DMA_CHNxCNT 寄存器中设置要传输的数据量。在每个数据传输后，这个数值递减。
4. 在 DMA_CHNxCTL 寄存器的 PL[1:0]位中设置通道的优先级。
5. 在 DMA_CHNxCTL 寄存器中设置数据传输模式、循环模式、外设和存储器的增量模式、外设和存储器的传送位数。
6. 设置 DMA_CHNxCTL 寄存器的 CHNxEN 位，启动该通道。

一旦启动了 DMA 通道，它既可响应连到该通道上的外设的 DMA 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（如 ADC 的扫描模式）。在 DMA_CHNxCTL 寄存器中的 CYCLE 位用于开启这一功能。当启动了循环模式，数据传输的数目变为 0 时，将会自动地被恢复成配置通道时设置的初值，DMA 操作将会继续进行。

### 11.3.3 DMA 通道请求列表

**DMA 控制器**

从外设（soft request、TIMx[x=0、1、2、3、8]、SAR ADC、I2C 和 UART1）产生的请求，通过逻辑或输入到 DMA 控制器，这意味着同时只能有一个请求有效。参见下表的 DMA 通道请求列表。

外设的 DMA 请求，可以通过设置相应外设寄存器中的控制位，被独立地开启或关闭。

表 11.1 DMA 通道请求列表

| DMA_CTL[8...13] | 说明 |
|---|---|
| 0 | Soft request |
| 1 | 保留 |
| 2 | 保留 |
| 3 | UART1 发送 |
| 4 | UART1 接收 |
| 5~18 | 保留 |
| 19 | I2C 发送 |
| 20 | I2C 接收 |
| 21 | SAR ADC |
| 22 | TIM8_UP |
| 23 | TIM8_TRIG |
| 24 | Timer0 |
| 25 | Timer1 |

| | |
|---|---|
| 26 | Timer2 |
| 27 | Timer3 |
| 28 | TIM8_COM |
| 29 | TIM8_CH1 |
| 30 | TIM8_CH2 |
| 31 | TIM8_CH3 |
| 32 | TIM8_CH4 |
| 33 | TIM8_CH5 |
| 34 | TIM8_CH6 |

## 11.4 DMA 寄存器

### 11.4.1 DMA 中断使能寄存器（DMA_DMAIE）

地址偏移：0x00

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | TEIE2 | TEIE1 | TEIE0 | Reserved | BCIE2 | BCIE1 | BCIE0 | Reserved | TCIE2 | TCIE1 | TCIE0 |
| | | | | | rw | rw | rw | | rw | rw | rw | | rw | rw | rw |

Bit 15:11 保留，必须保持为复位值

Bit 10:8 **TEIE[2:0]**：通道 0/1/2 传输错误中断使能

0：禁止

1：使能

Bit 7 保留，必须保持为复位值

Bit 6:4 **BCIE[2:0]**：通道 0/1/2 块传输中断使能

0：禁止

1：使能

Bit 3 保留，必须保持为复位值

Bit 2:0 **TCIE[2:0]**：通道 0/1/2 传输结束中断使能

0：禁止

1：使能

### 11.4.2 DMA 中断标志寄存器（DMA_DMAIF）

地址偏移：0x04

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | TEIF2 | TEIF1 | TEIF0 | Reserved | BCIF2 | BCIF1 | BCIF0 | Reserved | TCIF2 | TCIF1 | TCIF0 |
| | | | | | rw | rw | rw | | rw | rw | rw | | rw | rw | rw |

Bit 15:11 保留，必须保持为复位值

Bit 10:8 **TEIF[2:0]**：通道 0/1/2 传输错误中断标志

0：未产生中断

1：产生中断

Bit 7 保留，必须保持为复位值

Bit 6:4 **BCIF[2:0]**：通道 0/1/2 块传输完成中断标志

0：未产生中断

1：产生中断

Bit 3 保留，必须保持为复位值

Bit 2:0 **TCIF[2:0]**：通道 0/1/2 传输结束中断标志

0：未产生中断

1：产生中断

### 11.4.3 DMA 状态寄存器（DMA_CHNSTA）

地址偏移：0x08

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | BUSY2 | BUSY1 | BUSY0 |
| | | | | | | | | | | | | | rw | rw | rw |

Bit 15:3 保留，必须保持为复位值

Bit 2:0 **BUSY[2:0]**：通道 0/1/2 传输 BUSY 标志

0：空闲

1：忙碌

### 11.4.4 DMA 通道控制寄存器（DMA_CHNxCTL）

地址偏移：0x0C，0x24

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | PL[1:0] | |
| | | | | | | | | | | | | | | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Soft_Trig | Channel[5:0] | | | | | | DESTIN_INC[1:0] | | SOURC_INC[1:0] | | MODE | CYCLE | PSIZE[1:0] | | DMA_CHNxEN |
| w | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:18 保留，必须保持为复位值

Bit 17:16 **PL[1:0]**：通道优先级，这些位由软件设置和清除。

00：低

01：中

10：高

11：最高

Bit 15 **Soft_Trig**：Soft request 触发位：

1：当请求源选择 Soft request 时，触发一次 DMA 硬件自动清 0

Bit 14:9 **Channel[5:0]**：触发通道选择，具体参见 DMA 通道请求列表来定。

Bit 8:7 **DESTIN_INC[1:0]**：目的地址增量模式

00：不增加

01：增加

10：数据块内循环增加

11：数据块内循环增加

Bit 6:5 **SOURC_INC[1:0]**：源地址地址增量模式

00：不增加

01：增加
10：数据块内循环增加
11：数据块内循环增加

Bit 4 **MODE**：传输模式
0：单次传输模式
1：块传输模式（块数据传输过程中不会被打断）

Bit 3 **CYCLE**：循环模式
0：不循环模式
1：循环模式

Bit 2:1 **PSIZE[2:1]**：MEMORY 的传送位数（外设的传送位数固定为 32bit）：
00：8 位
01：16 位
10：32 位
11：32bit

Bit 0 **DMA_CHNxEN**：DMA 通道使能
0：禁止
1：使能

### 11.4.5 DMA 通道源地址寄存器（DMA_CHNxSRC）

地址偏移：0x10，0x28
复位值：0x0000 0000
访问：无等待，支持字，半字和字节访问

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

Bit 31:0 **ADDR[31:0]**：数据传输源地址寄存器
*注：x 为0，1*

### 11.4.6 DMA 通道目的地址寄存器（DMA_CHNxTAR）

地址偏移：0x14，0x2C
复位值：0x0000 0000
访问：无等待，支持字，半字和字节访问

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

Bit 31:0 **ADDR[31:0]**：数据传输目的地址寄存器
*注：x 为0，1*

### 11.4.7 DMA 通道传输数量寄存器（DMA_CHNxCNT）

地址偏移：0x18，0x30

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Num 15 | Num 14 | Num 13 | Num 12 | Num 11 | Num 10 | Num 9 | Num 8 | Num 7 | Num 6 | Num 5 | Num 4 | Num 3 | Num 2 | Num 1 | Num 0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Num[15:0]：** DMA 数据传输个数设置寄存器

最大设置到 65535 个传输数据，如果用户设置的是块传输，那么该寄存器则表示用户需要传输的数据块个数。

*注：* x *为*0，1

### 11.4.8 DMA 已传输数据个数寄存器（DMA_CHNxTCCNT）

地址偏移：0x1C，0x34

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Num 15 | Num 14 | Num 13 | Num 12 | Num 11 | Num 10 | Num 9 | Num 8 | Num 7 | Num 6 | Num 5 | Num 4 | Num 3 | Num 2 | Num 1 | Num 0 |
| r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

Bit 15:0 **Num[15:0]：** 指示 DMA 已经传输完成的数据个数

如果用户设置的是块传输，那么该寄存器则表示 DMA 已经传输完成的数据块个数。

*注：* x *为*0，1

### 11.4.9 DMA 通道块传输设置寄存器（DMA_CHNxBULKNUM）

地址偏移：0x20，0x38

复位值：0x0000 0000

访问：无等待，支持字，半字和字节访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | CYCLE7 | CYCLE6 | CYCLE5 | CYCLE4 | CYCLE3 | CYCLE2 | CYCLE1 | CYCLE0 |
| | | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Num 15 | Num 14 | Num 13 | Num 12 | Num 11 | Num 10 | Num 9 | Num 8 | Num 7 | Num 6 | Num 5 | Num 4 | Num 3 | Num 2 | Num 1 | Num 0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 23:16 **CYCLE[7:0]：** 循环次数选择，当用户选择 DMA 传输为循环模式时起作用：

0：无限次循环

0x01：0xFF 次循环

Bit 15:0 **Num[15:0]：** 块传输数据个数选择，表示每一个数据块内有多少个数据：

0：65535 个

如果用户设置的是块传输，那么该寄存器则表示每一个数据块内有多少个数据

*注：* x *为*0，1

# 12 高级控制定时器（TIM8）

## 12.1 TIM8 简介

高级控制定时器（TIM8）由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。

它适合多种用途，包含测量输入信号的脉冲宽度（输入捕获），或者产生输出波形（输出比较、PWM、嵌入死区时间的互补 PWM 等）。

使用定时器预分频器和 RCC 时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。高级控制定时器（TIM8）和通用定时器（TIMx）是完全独立的，它们不共享任何资源。它们可以同步操作。

## 12.2 TIM8 主要特征

TIM8 定时器的功能包括：

- 16 位向上、向下、向上/下自动装载计数器
- 16 位可编程（可以实时修改）预分频器，计数器时钟频率的分频系数为 1～65535 之间的任意数值
- 多达 5 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成（边缘或中间对齐模式）
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 使用外部信号控制定时器和定时器互联的同步电路
- 允许在指定数目的计数器周期之后更新定时器寄存器的重复计数器
- 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态
- 如下事件发生时产生中断/DMA：
  - 更新：计数器向上溢出/向下溢出，计数器初始化（通过软件或者内部/外部触发）
  - 触发事件（计数器启动、停止、初始化或者由内部/外部触发计数）
  - 输入捕获
  - 输出比较
  - 刹车信号输入
- 支持针对定位的增量（正交）编码器和霍尔传感器电路
- 触发输入作为外部时钟或者按周期的电流管理

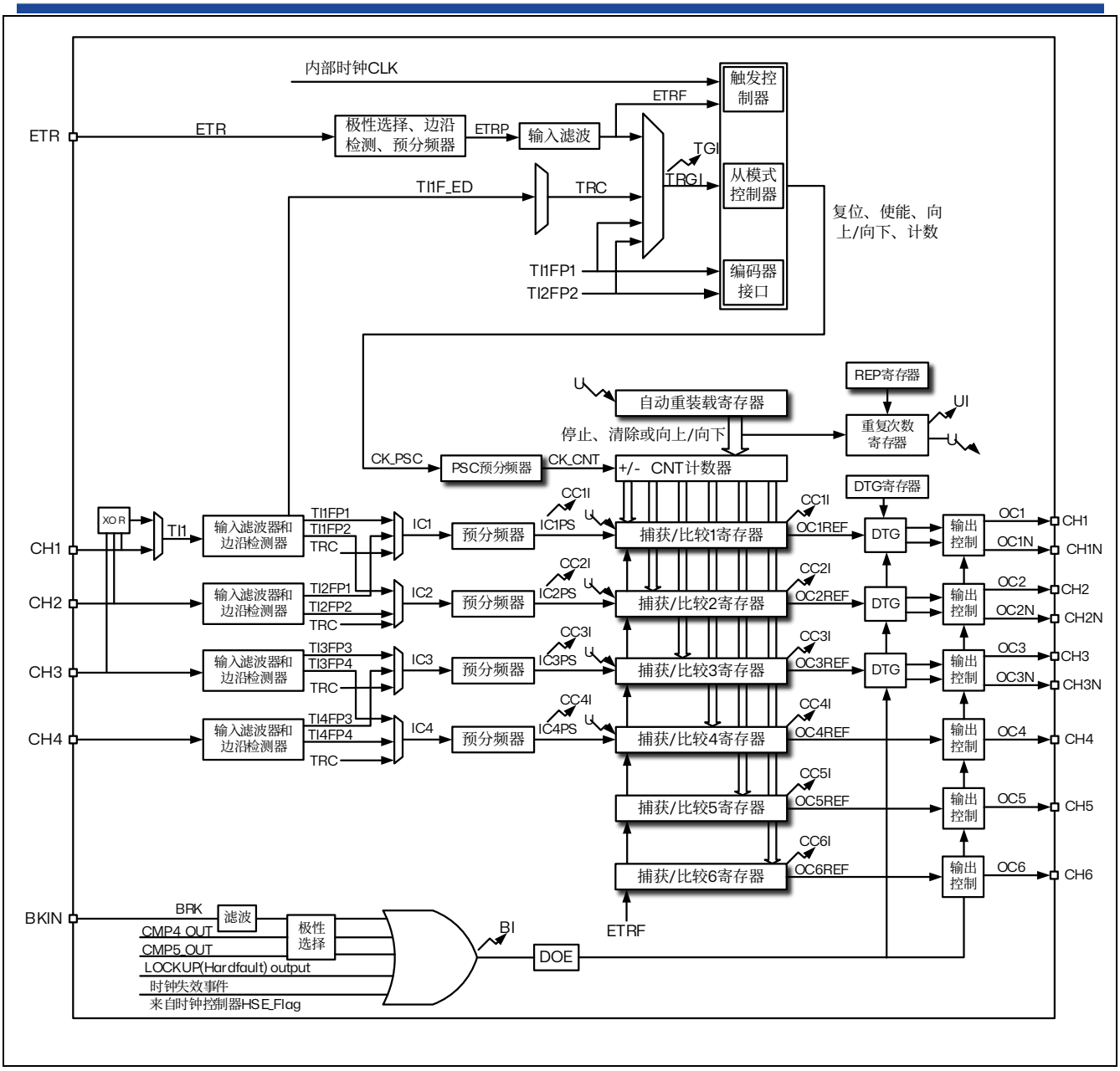



图 12.1 高级控制定时器框图

*注：根据控制位设定，在* U(更新)*事件时传送预加载寄存器的内容至工作寄存器*

*事件*

*中断和* DMA *输出*

## 12.3 TIM8 功能描述

### 12.3.1 时基单元

可编程高级控制定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。

计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单元包含：

- 计数器寄存器（TIM8_CNT）
- 预分频器寄存器（TIM8_PSC）
- 自动装载寄存器（TIM8_ARR）
- 重复次数寄存器（TIM8_RCR）

自动装载寄存器是预先装载的，写或读自动重装载寄存器将访问预装载寄存器。根据在 TIM8_CR1 寄存器中的自动装载预装载使能位（ARPE）的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件（向下计数时的下溢条件）并当 TIM8_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 TIM8_CR1 寄存器中的计数器使能位（CEN）时，CK_CNT 才有效。（更多有关使能计数器的细节，请参见控制器的从模式描述）。

注意，在设置了 TIM8_CR 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

**预分频器描述**

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个（在 TIM8_PSC 寄存器中的）16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

图 12.2 和图 12.3 给出了预分频器运行时，更爱计数器参数的例子。

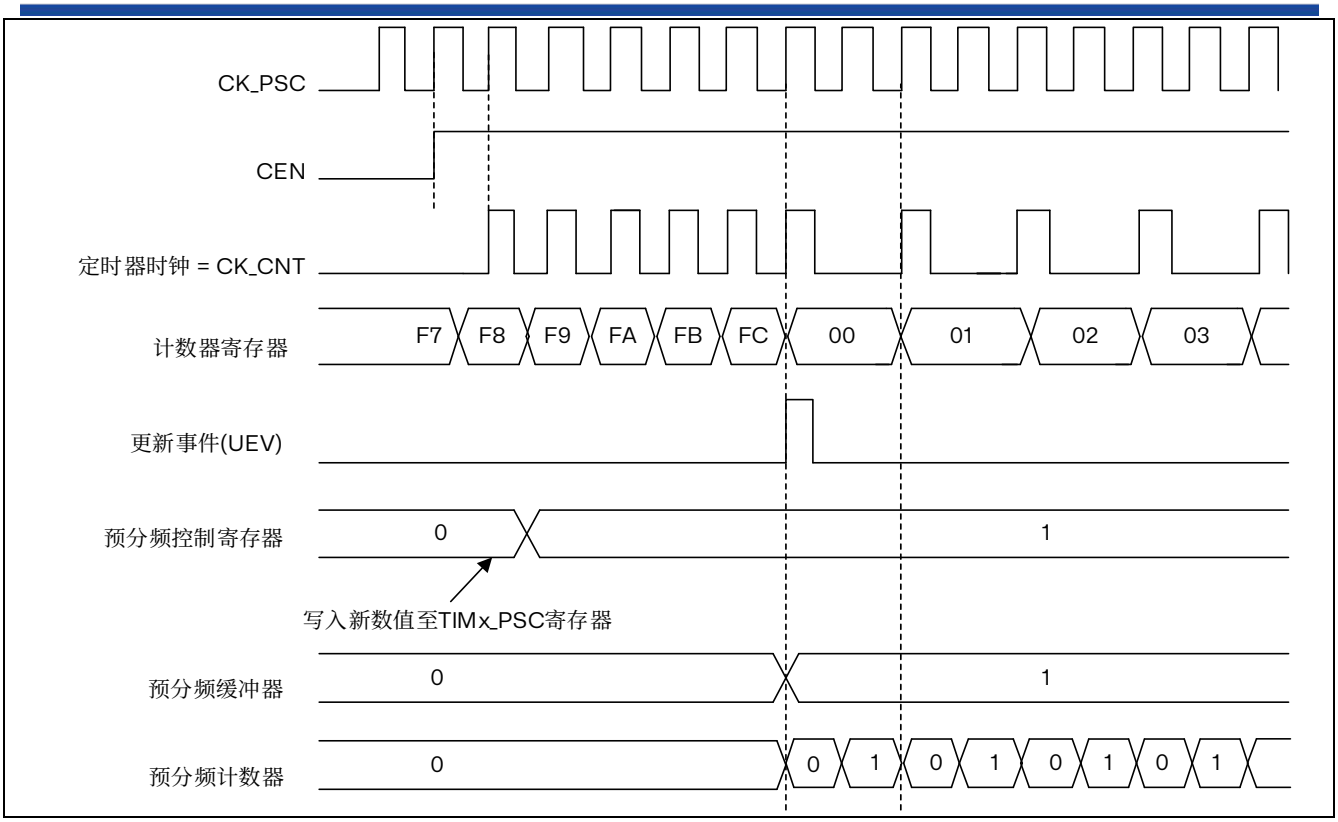



图 12.2 当预分频器的参数从 1 变成 2 时，计数器的时序图

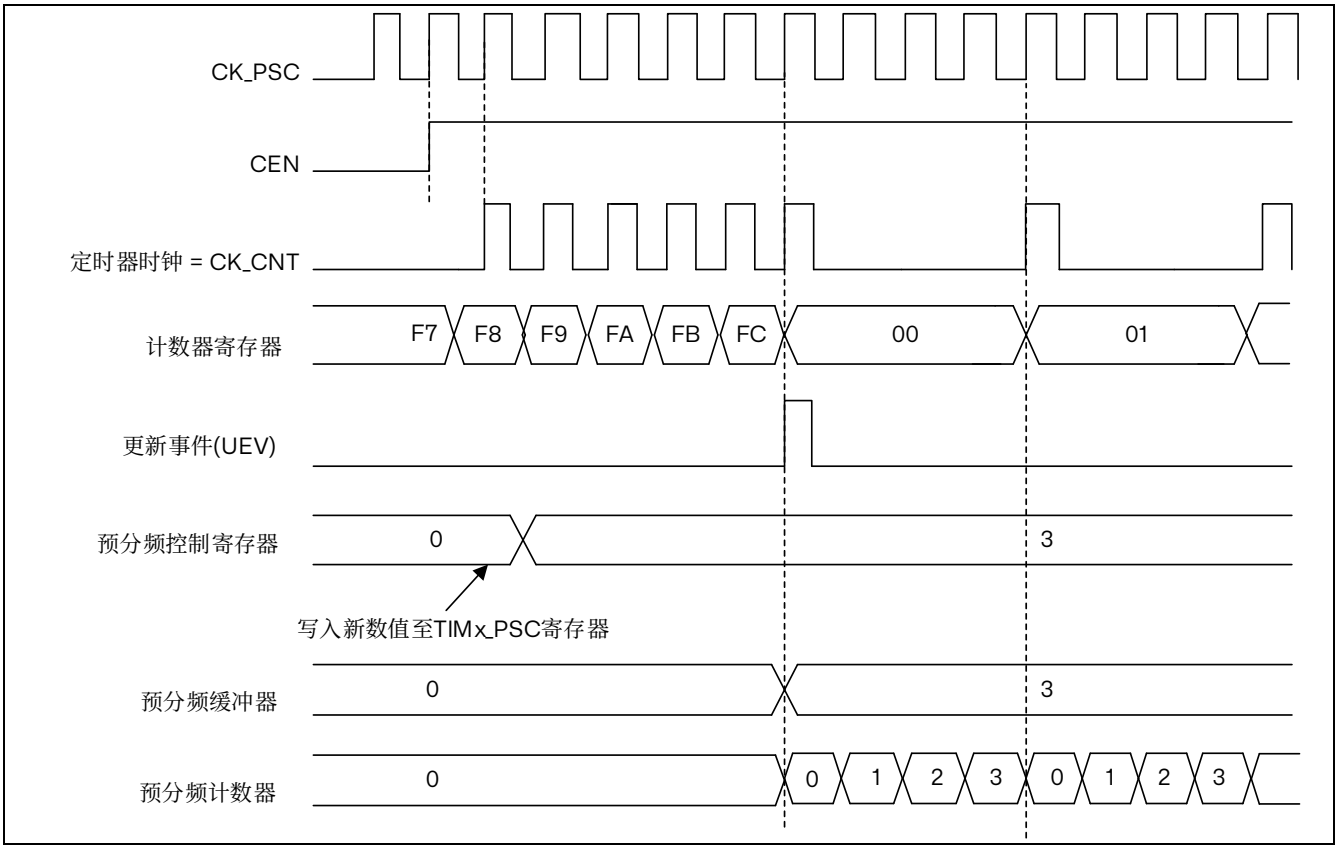



图 12.3 当预分频器的参数从 1 变成 4 时，计数器的时序图

### 12.3.2 计数器模式

#### 向上计数模式

在向上计数模式中，计数器从 0 计数到自动加载值（TIM8_ARR 计数器的内容），然后重新从 0 开始计数并且产生一个计数器溢出事件。

如果使用了重复计数器功能，在向上计数达到设置的重复计数次数（TIM8_RCR）时，产生更新事件（UEV）；否则每次计数器溢出时才产生更新事件。

在 TIM8_EGR 寄存器中（通过软件方式或者使用从模式控制器）设置 UG 位也同样可以产生一个更新事件。

设置 TIM8_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影子寄存器。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，计数器仍会被清‘0’，同时预分频器的计数也被请 0（但预分频器的数值不变）。此外，如果设置了 TIM8_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志（即不产生中断或 DMA 请求）。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，硬件同时（依据 URS 位）设置更新标志位（TIM8_SR 寄存器中的 UIF 位）。

- 重复计数器被重新加载为 TIM8_RCR 寄存器的内容。
- 自动装载影子寄存器被重新置入预装载寄存器的值（TIM8_ARR）。
- 预分频器的缓冲区被置入预装载寄存器的值（TIM8_PSC 寄存器的内容）。

下图给出一些例子，当 TIM8_ARR=0x36 时计数器在不同时钟频率下的动作。

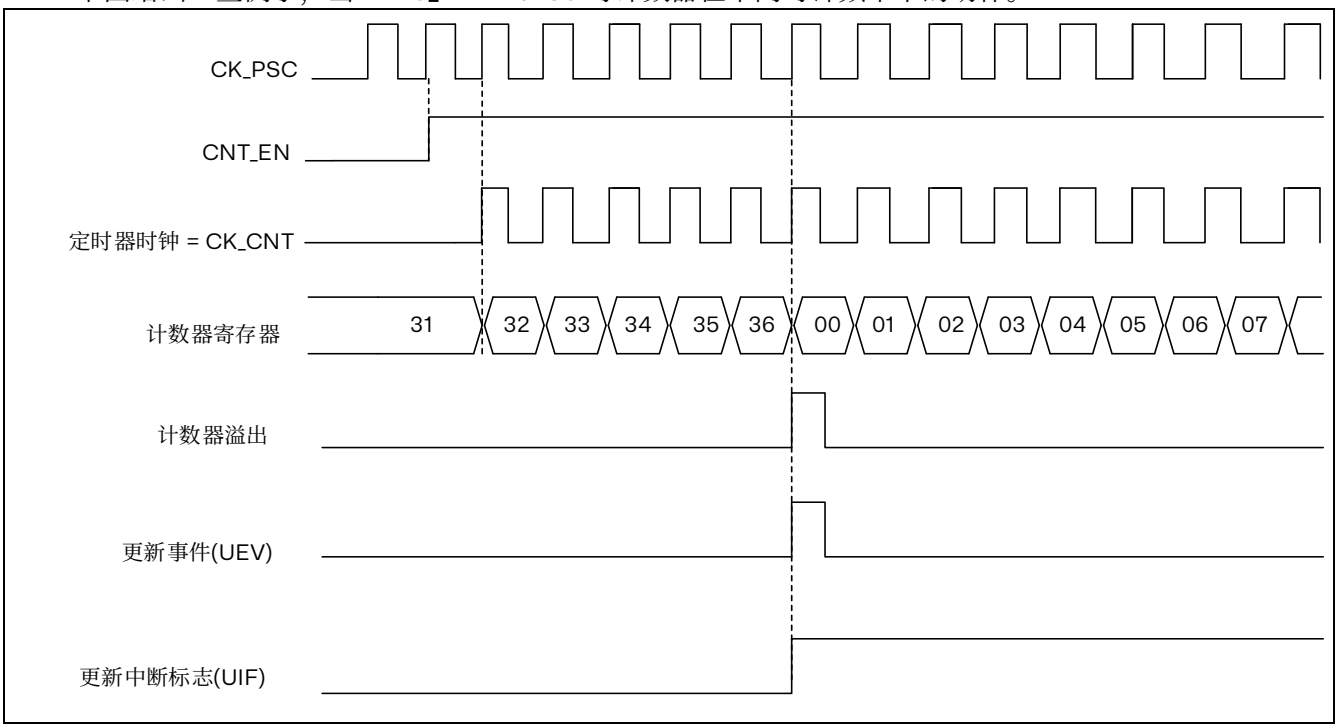



图 12.4 计数器时序图，内部时钟分频因子为 1

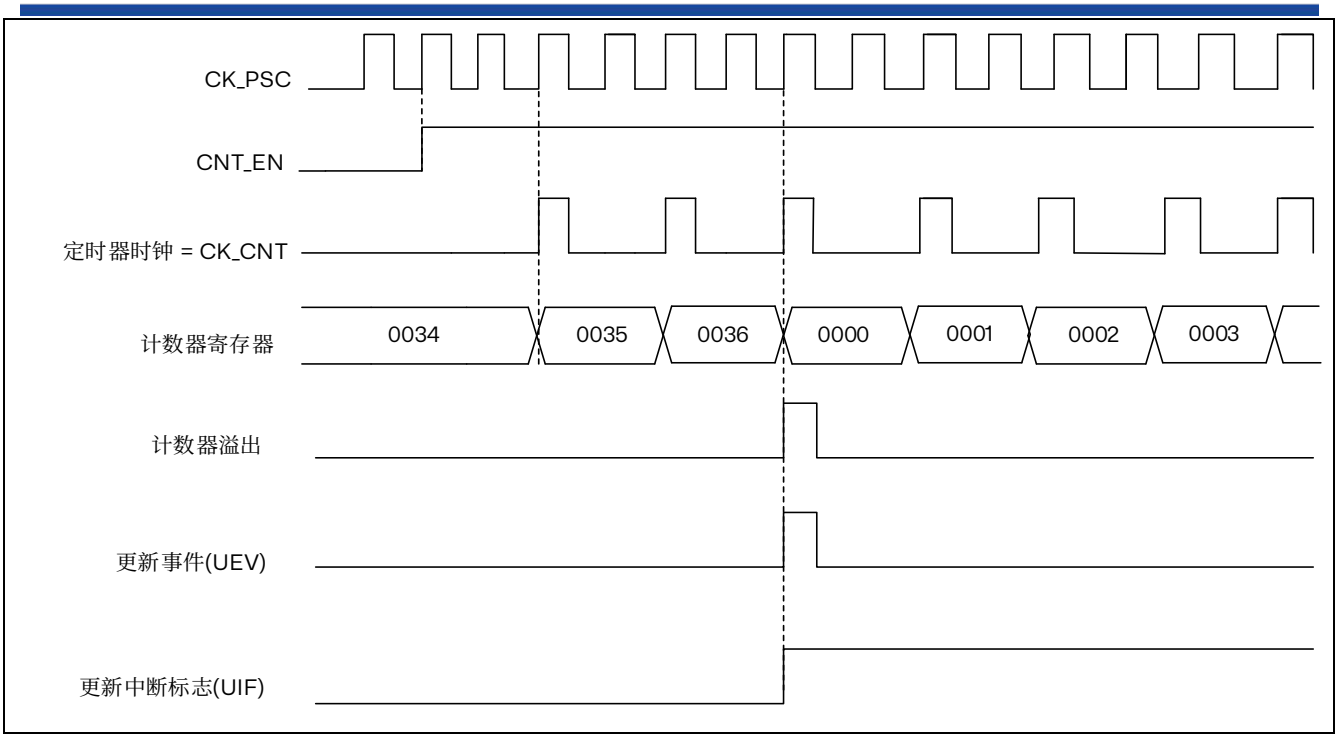



图 12.5 计数器时序图，内部时钟分频因子为 2

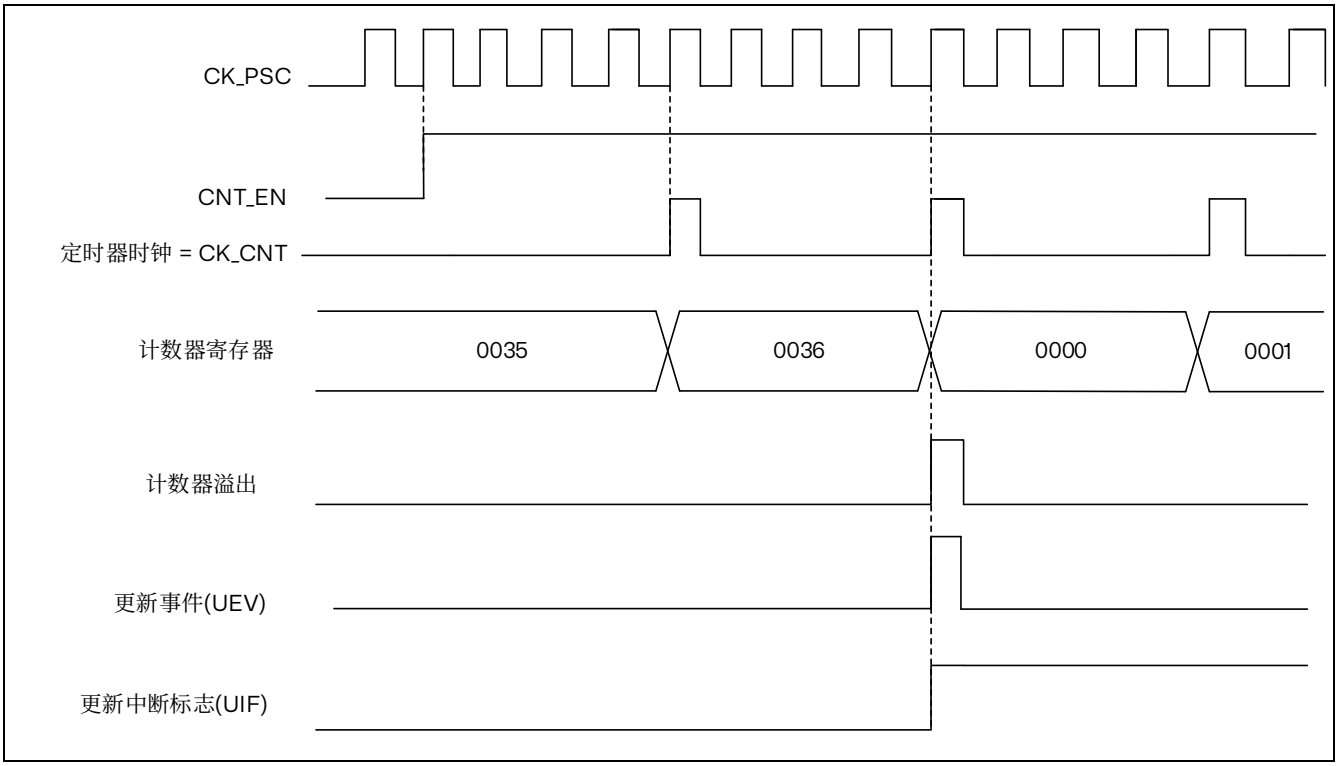



图 12.6 计数器时序图，内部时钟分频因子为 4

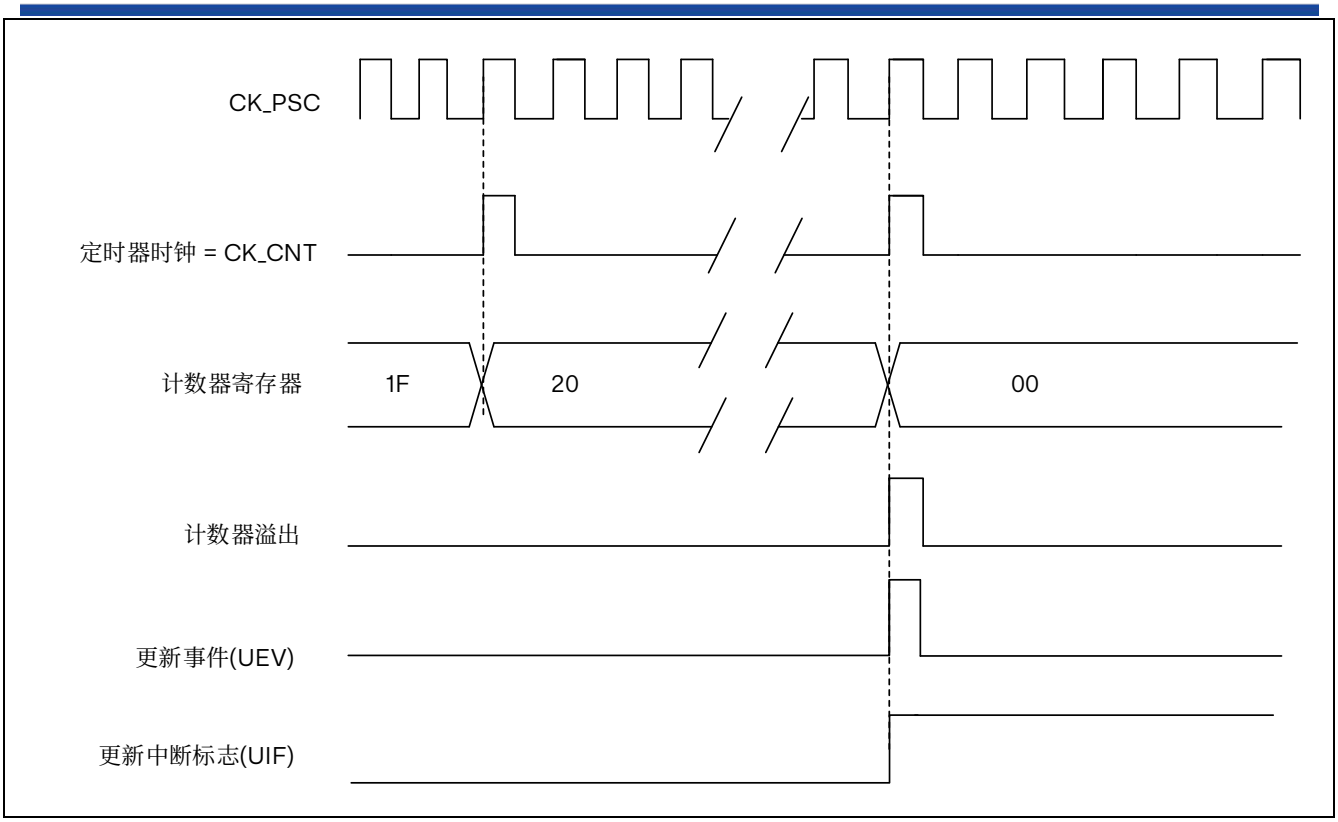



图 12.7 计数器时序图，内部时钟分频因子为 N

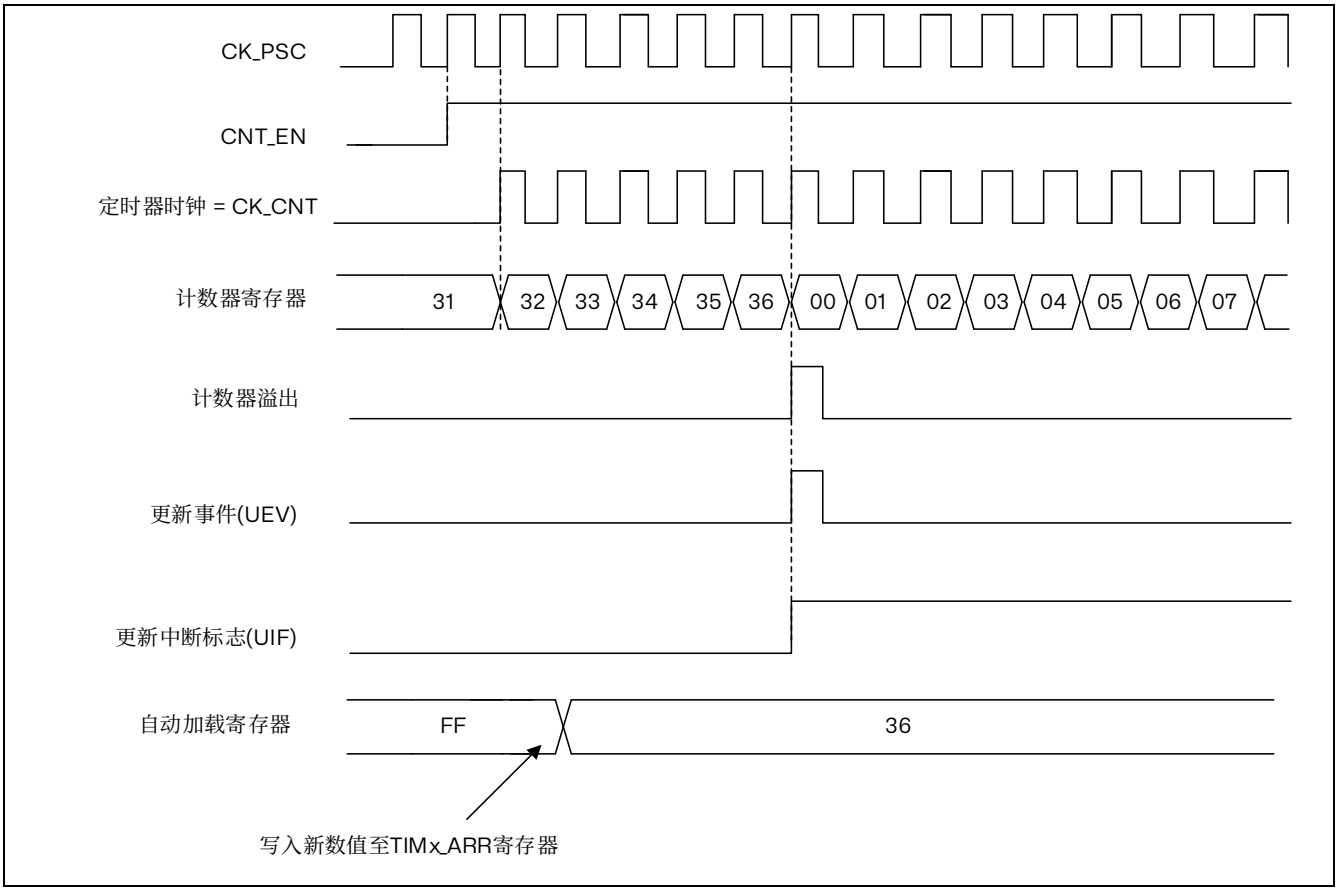



图 12.8 计数器时序图，当 ARPE=0 时的更新事件（TIM8_ARR 没有预装入）

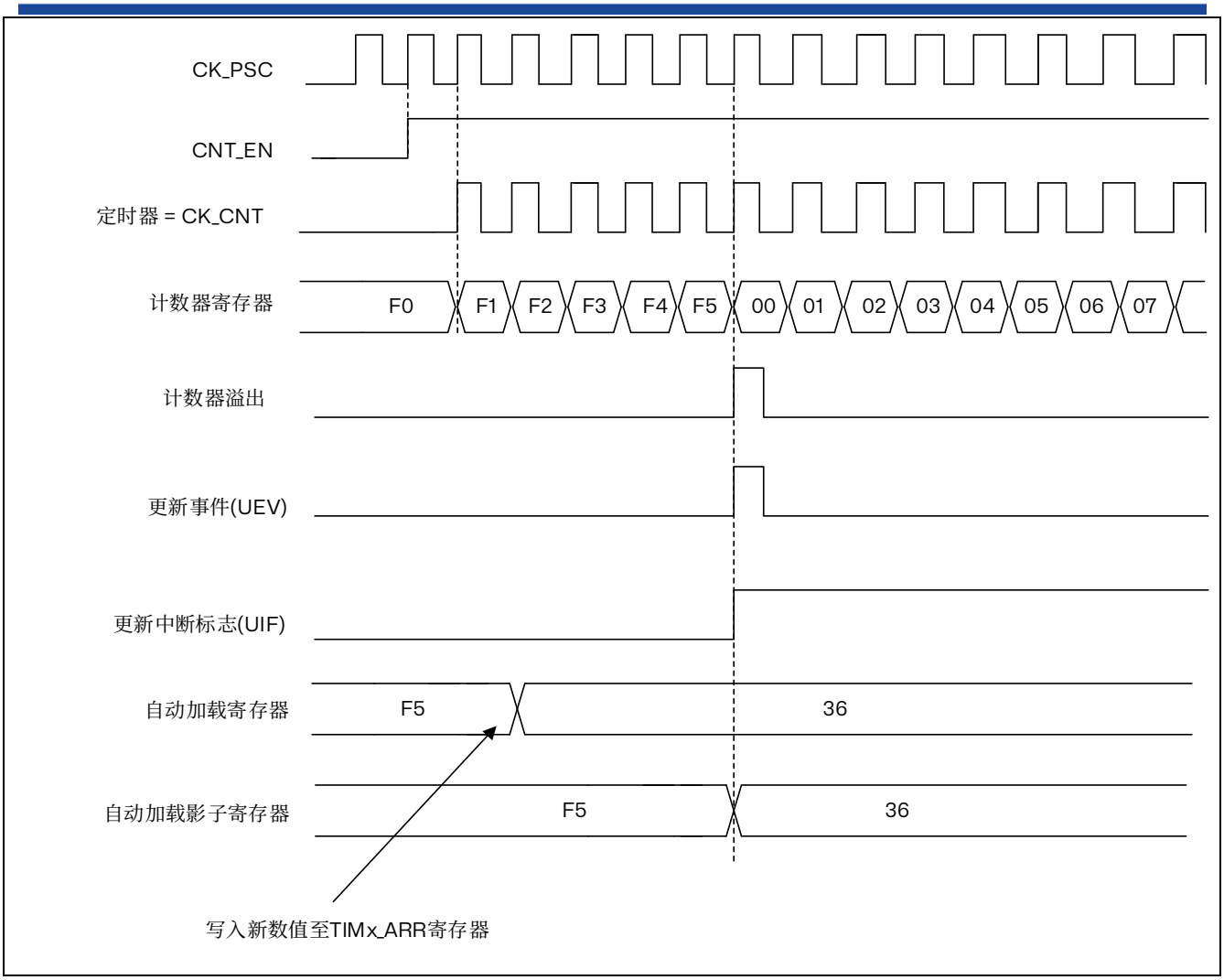



图 12.9 计数器时序图，当 ARPE=1 时的更新事件（预装入了 TIM8_ARR）

**向下计数模式**

在向下模式中，计数器从自动装入的值（TIM8_ARR 计数器的值）开始向下计数到 0，然后从自动装入的值重新开始并且产生一个计数器向下溢出事件。

如果使用了重复计数器，当向下计数重复了重复计数寄存器（TIM8_RCR）中设定的次数后，将产生更新事件（UEV），否则每次计数器下溢时才产生更新事件。

在 TIM8_EGR 寄存器中（通过软件方式或者使用从模式控制器）设置 UG 位，也同样可以产生一个更新事件。

设置 TIM8_CR1 寄存器的 UDIS 位可以禁止 UEV 事件。这样可以避免向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会从当前自动加载值重新开始计数，并且预分频器的计数器重新从 0 开始（但预分频系数不变）。

此外，如果设置了 TIM8_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断和 DMA 请求），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且（根据 URS 位的设置）更新标志位（TIM8_SR 寄存器中的 UIF 位）也被设置。

- 重复计数器被重置为 TIM8_RCR 寄存器中的内容
- 预分频器的缓存器被加载为预装载的值（TIM8_PSC 寄存器的值）。
- 当前的自动加载寄存器被更新为预装载值（TIM8_ARR 寄存器中的内容）。注：自动装载在计数器重载入之前被更新，因此下一个周期将是预期的值。

以下是一些当 TIM8_ARR=0x36 时，计数器在不同时钟频率下的操作例子。

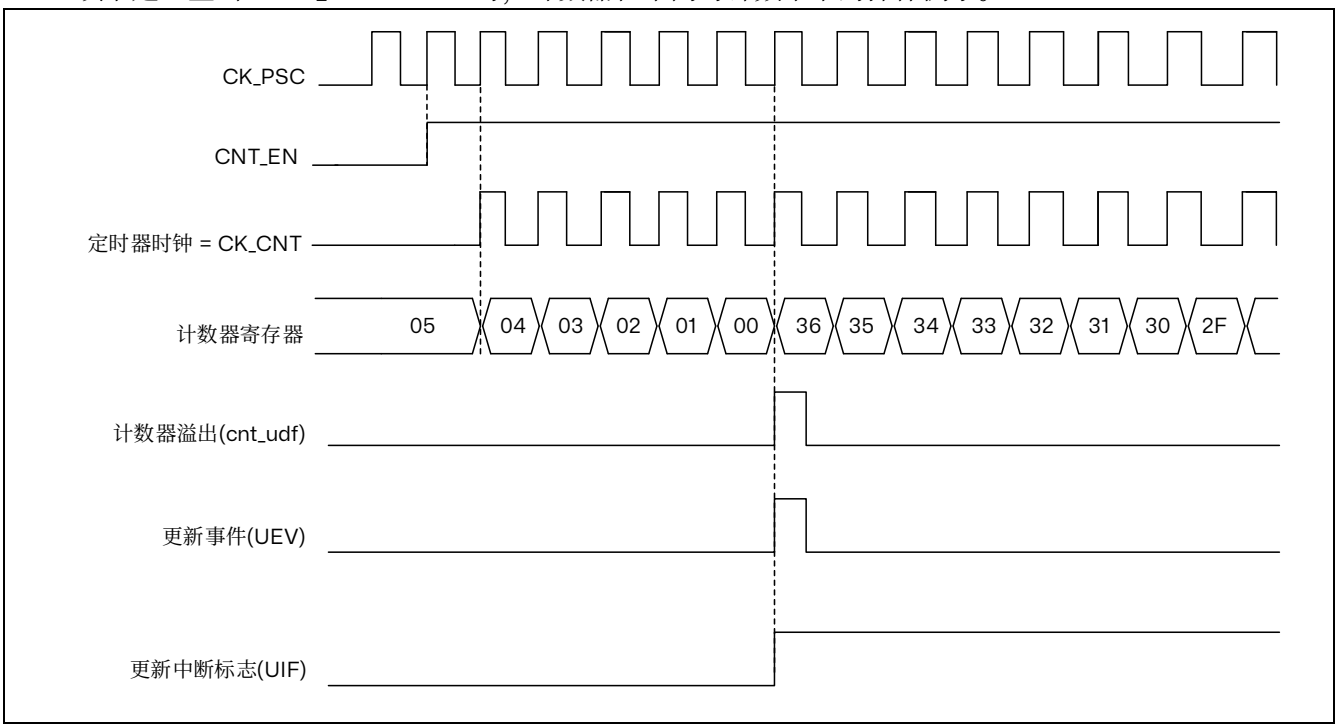



图 12.10 计数器时序图，内部时钟分频因子为 1

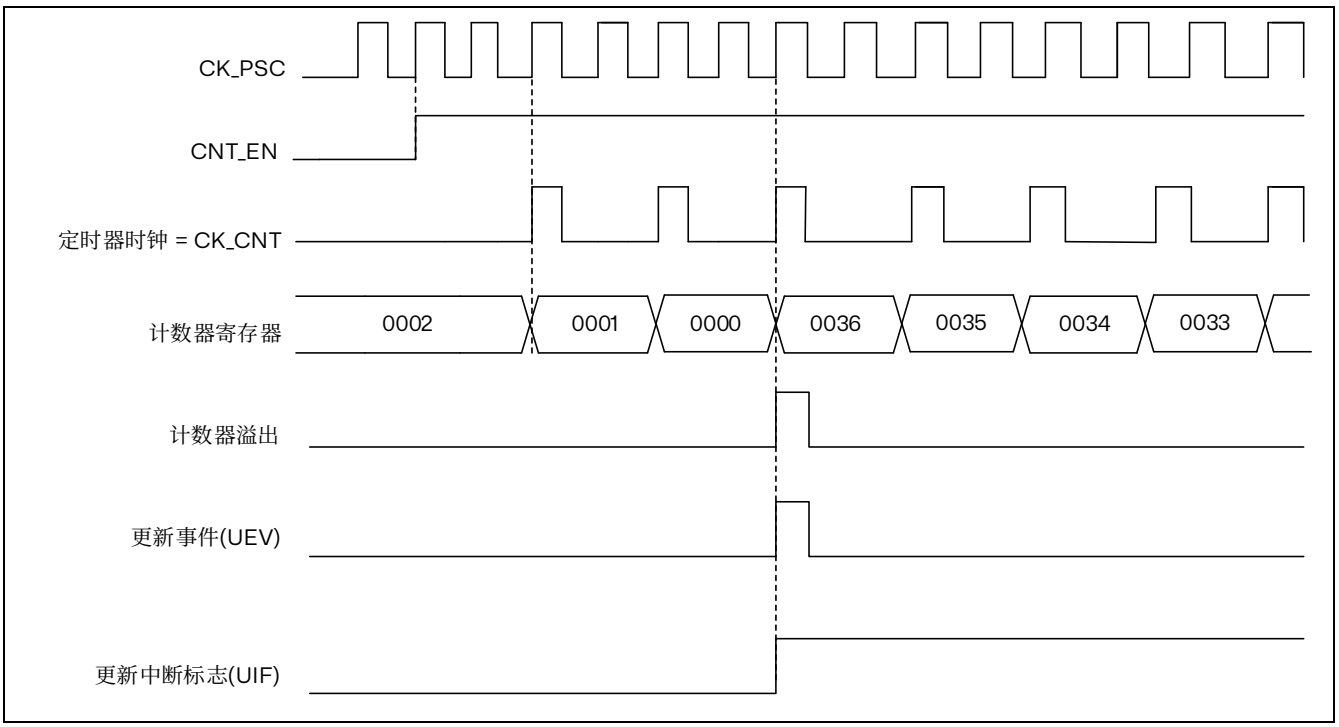



图 12.11 计数器时序图，内部时钟分频因子为 2

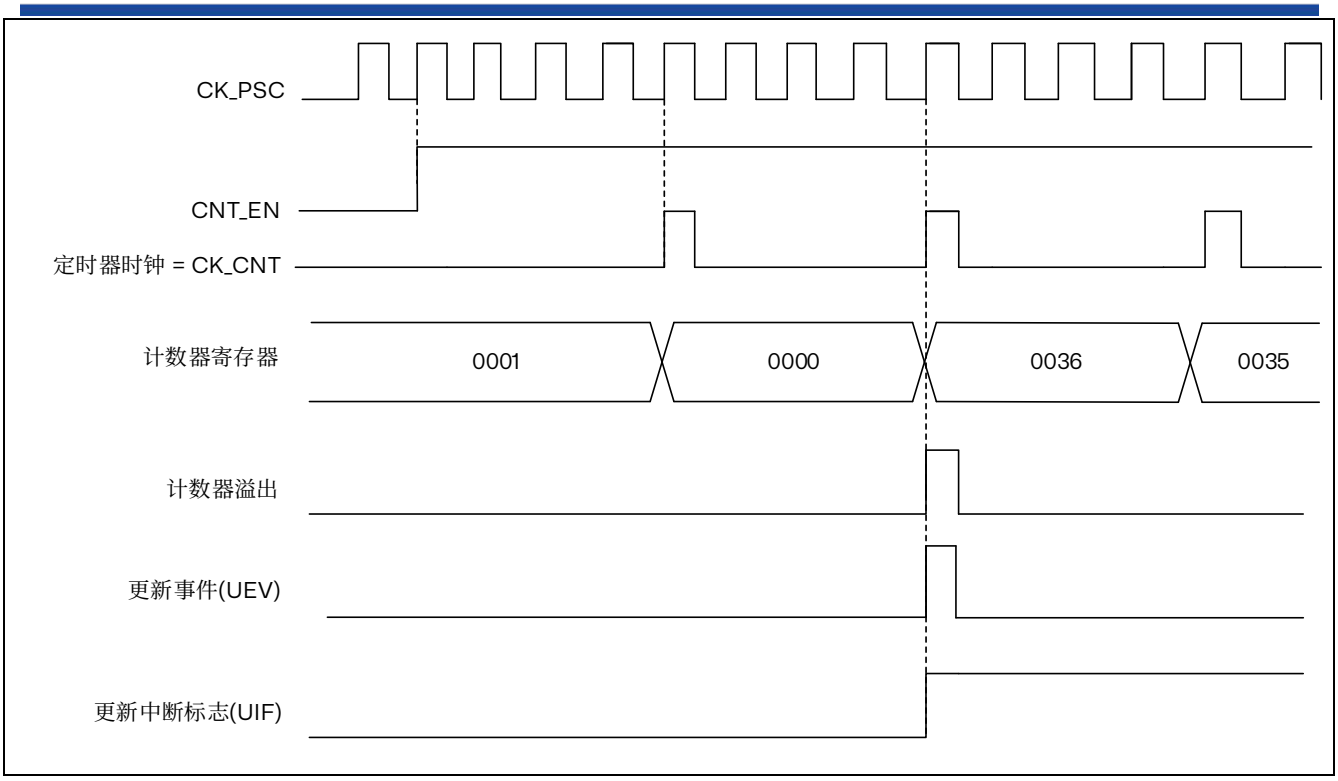



图 12.12 计数器时序图，内部时钟分频因子为 4

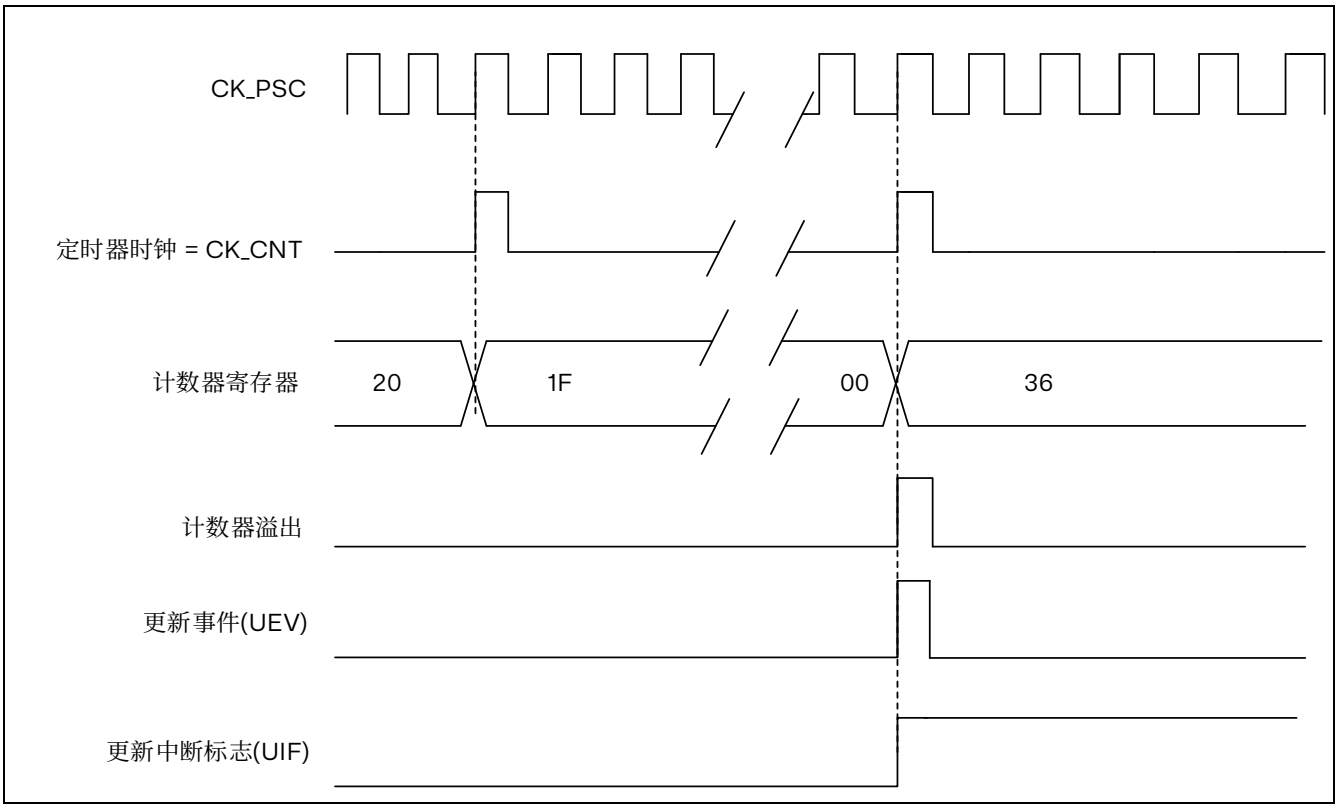



图 12.13 计数器时序图，内部时钟分频因子为 N

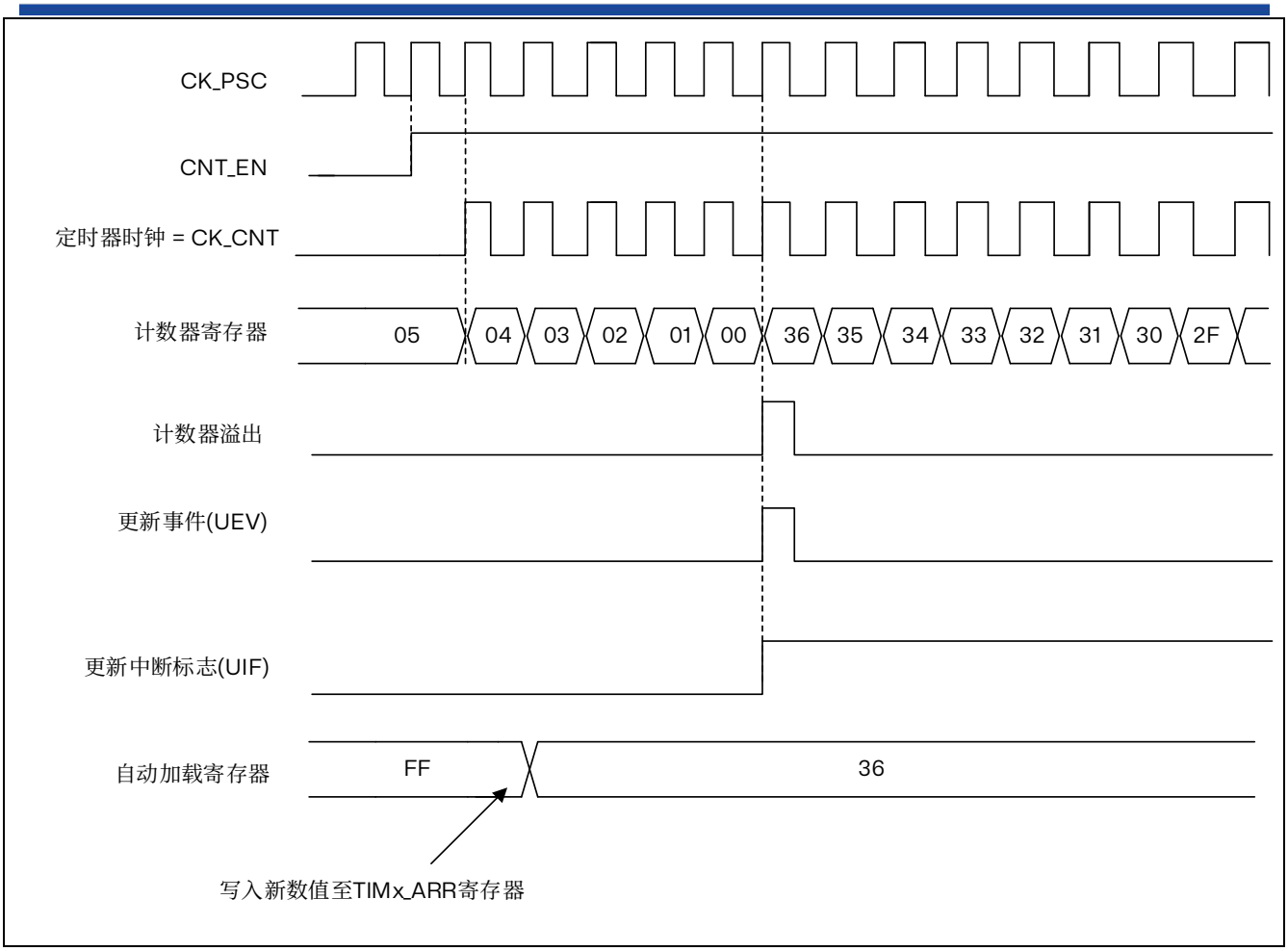



图 12.14 计数器时序图，当没有使用重复计数器时的更新事件

**中央对齐模式（向上、向下计数）**

在中央对齐模式，计数器从 0 开始计数到自动加载的值（TIM8_ARR 寄存器）−1，产生一个计数器溢出事件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。

在此模式下，不能写入 TIM8_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。

可以在每次计数上溢和每次计数下溢时产生更新事件；也可以通过（软件或者使用从模式控制器）设置 TIM8_EGR 寄存器中的 UG 位产生更新事件。然后，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

设置 TIM8_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。

此外，如果设置了 TIM8_CR1 寄存器中的 URS 位（选择更新请求），设置 UG 位将产生一个更新事件 UEV 但不设置 UIF 标志（因此不产生中断和 DMA 请求），这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且（根据 URS 位的设置）更新标志位（TIM8_SR 寄存器中的 UIF 位）也被设置。

- 重复计数器被重置为 TIM8_RCR 寄存器中的内容
- 预分频器的缓存器被加载为预装载（TIM8_PSC 寄存器）的值。

- 当前的自动加载寄存器被更新为预装载值（TIM8_ARR 寄存器中的内容）。注：如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前被更新，因此下一个周期将是预期的值（计数器被装载为新的值）。

以下是一些计数器在不同时钟频率下的操作的例子：

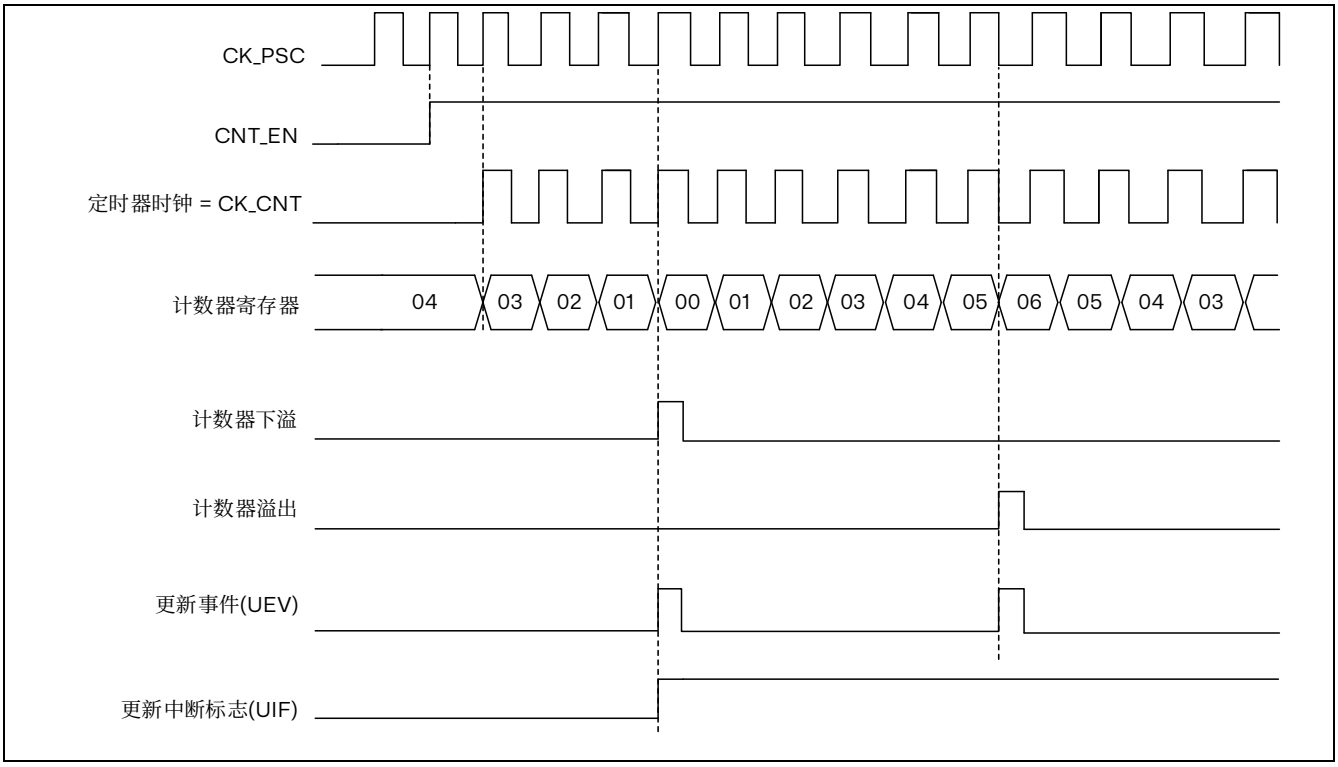



图 12.15 计数器时序图，内部时钟分频因子为 1，TIM8_ARR=0x6

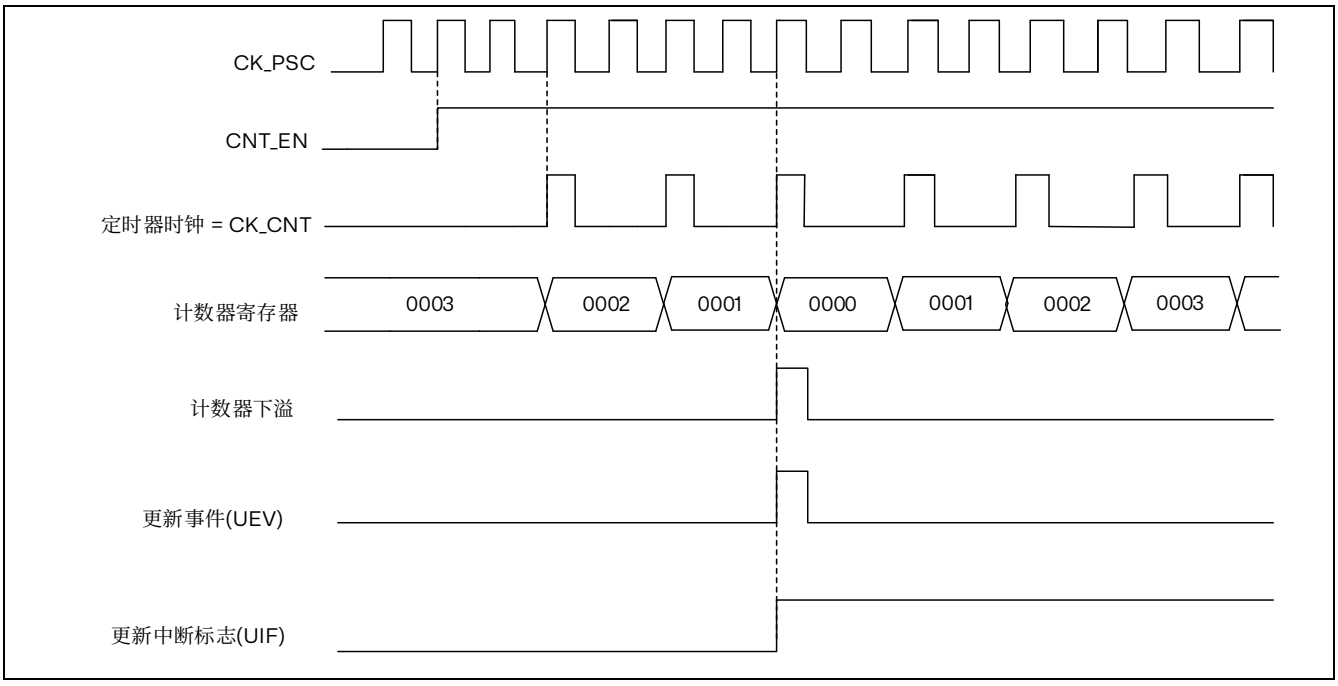



图 12.16 计数器时序图，内部时钟分频因子为 2

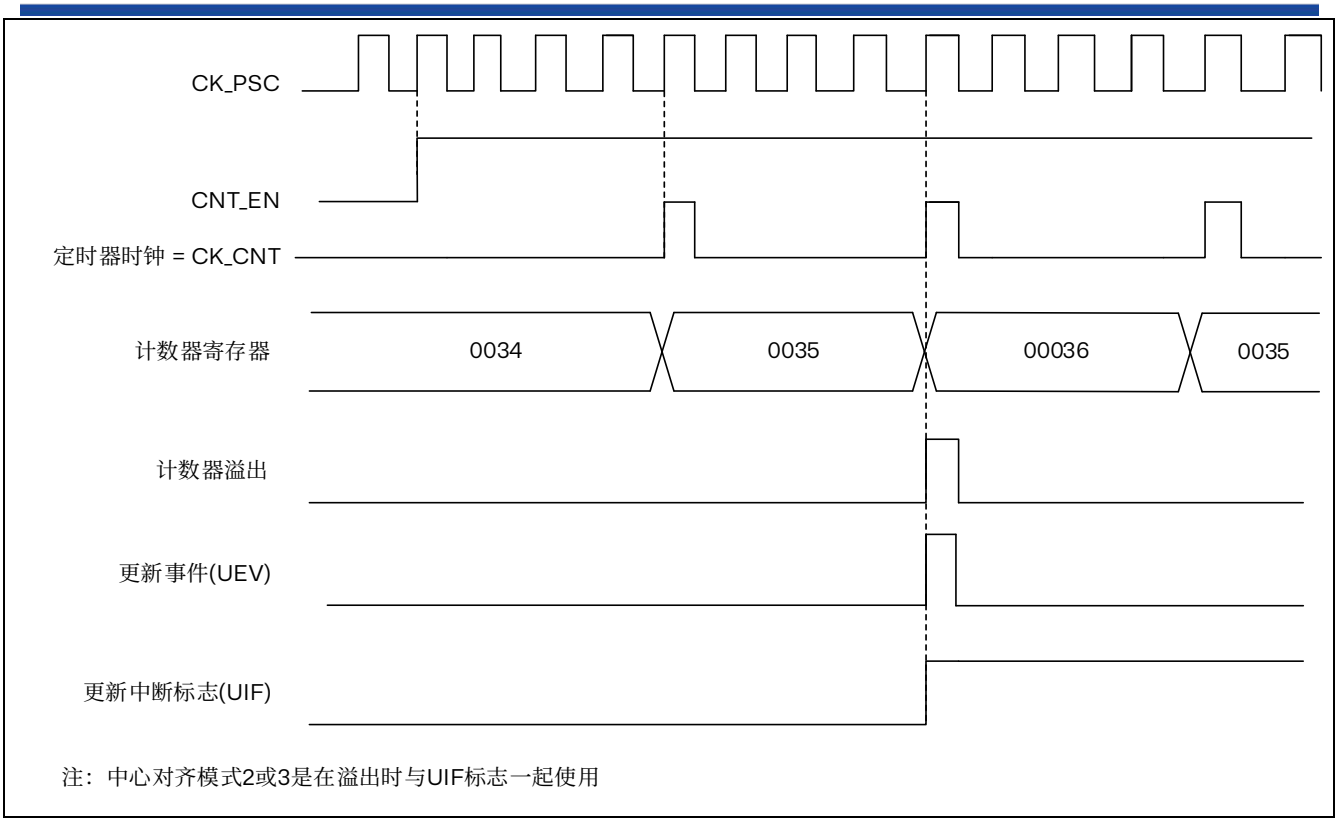



图 12.17 计数器时序图，内部时钟分频因子为 4，TIM8_ARR=0x36

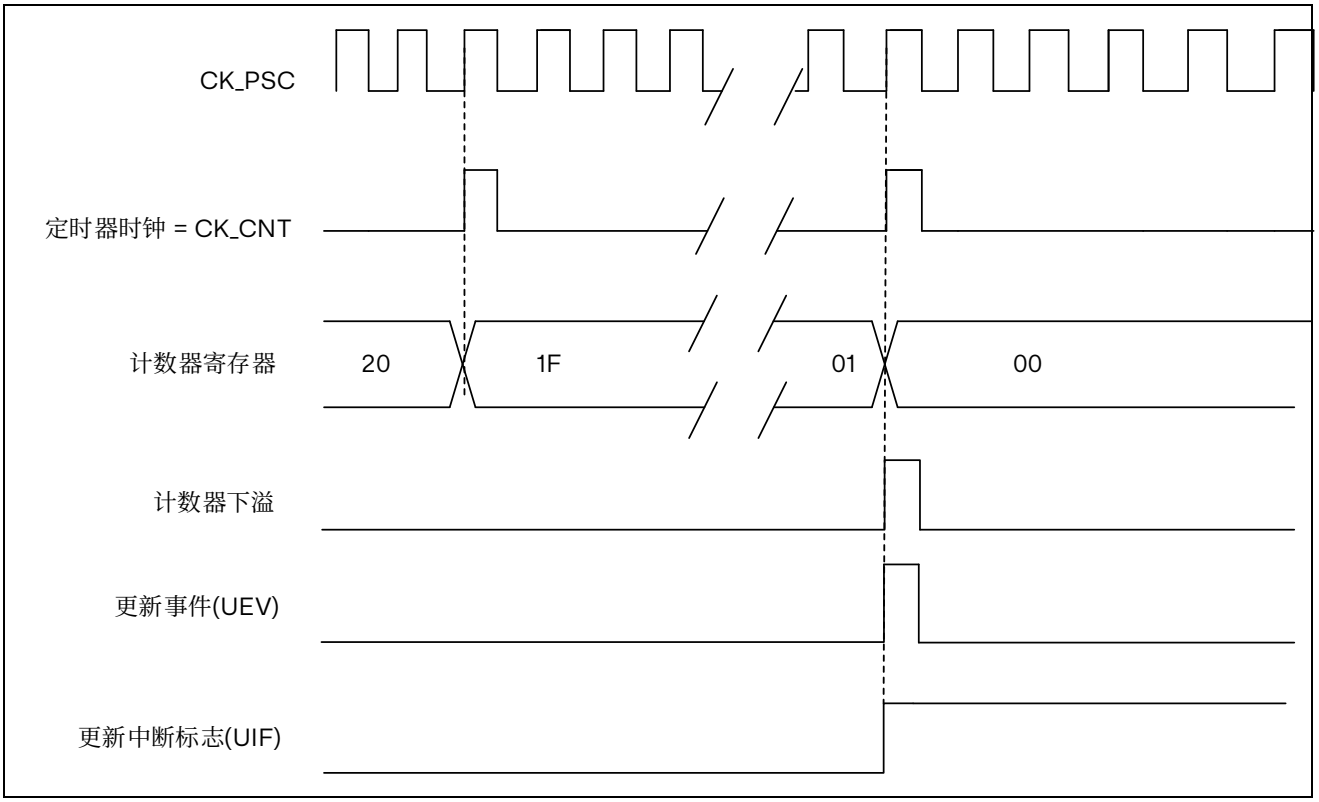



图 12.18 计数器时序图，内部时钟分频因子为 N

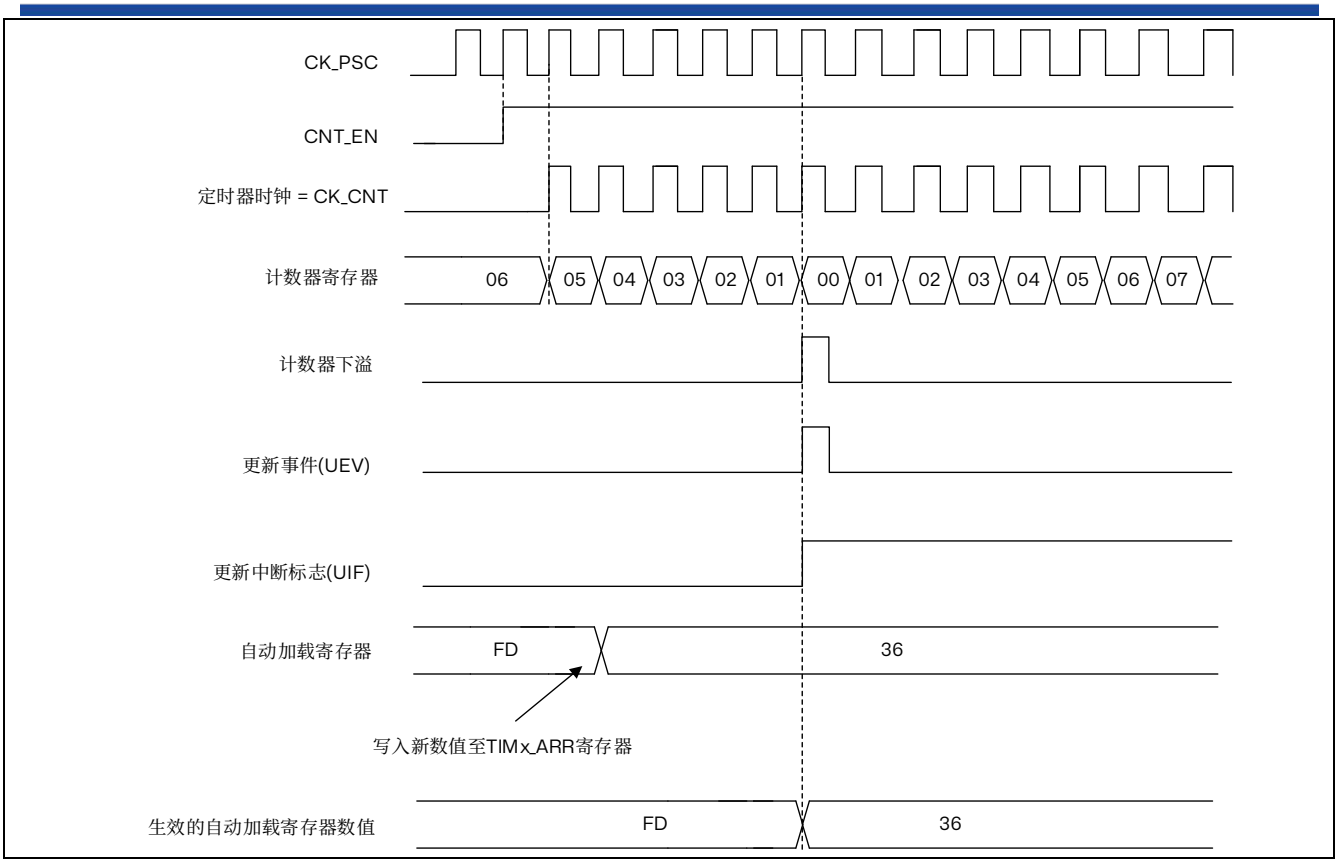



图 12.19 计数器时序图，ARPE=1 时的更新事件（计数器下溢）

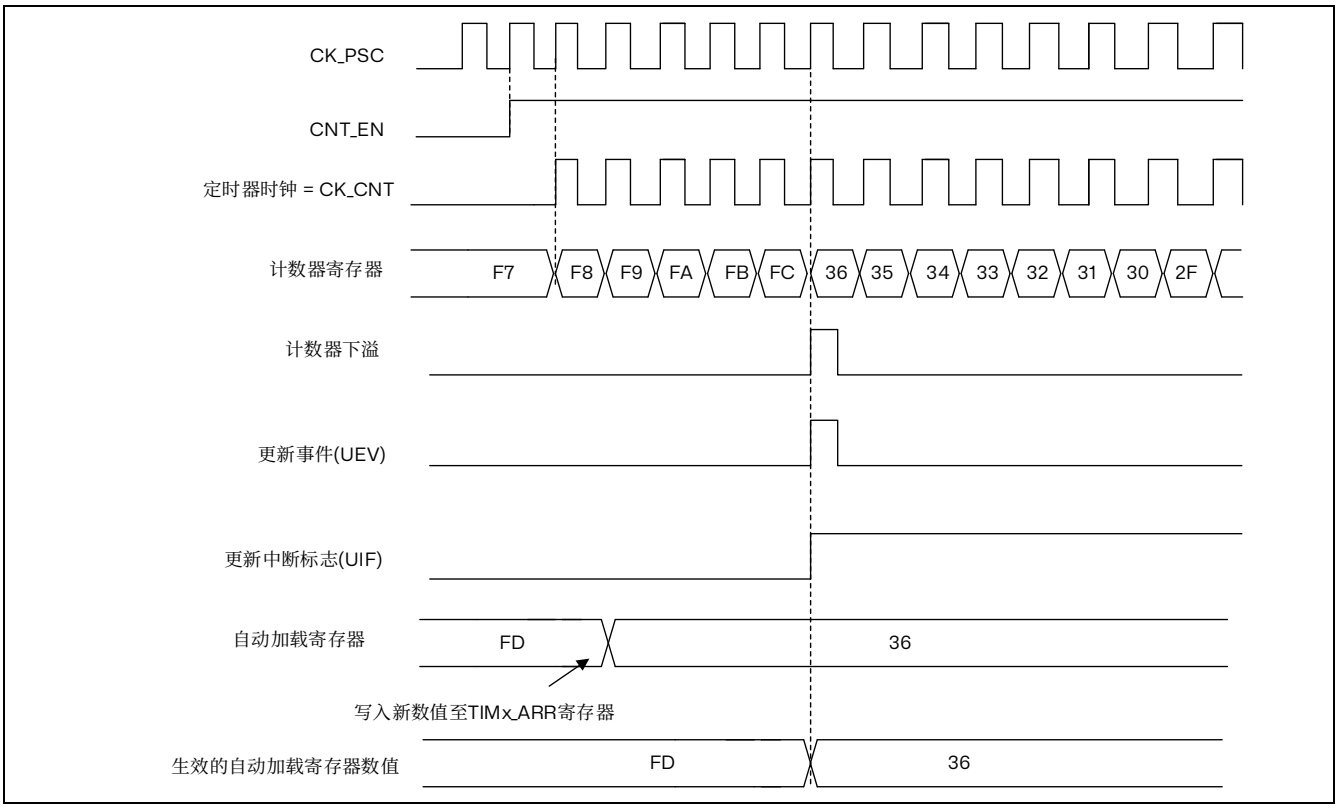



图 12.20 计数器时序图，ARPE=1 时的更新事件（计数器溢出）

### 12.3.3 重复计数器

12.3.1 节“时基单元”解释了计数器上溢/下溢时更新事件（UEV）是如何产生的，然而事实上它只能在重复计数达到 0 的时候产生。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输到影子寄存器（TIM8_ARR 自动重载入寄存器，TIM8_PSC 预装载寄存器，还有在比较模式下的捕获/比较寄存器 TIM8_CCRx），N 是 TIM8_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 PWM 的最大循环周期为 128，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。

重复计数器是自动加载的，重复速率是由 TIM8_RCR 寄存器的值定义（参看图 12.21）。当更新事件由软件产生（通过设置 TIM8_EGR 中的 UG 位）或者通过硬件的从模式控制器产生，则无论重复计数器的值是多少，立即发生更新事件，并且 TIM8_RCR 寄存器中的内容被重载入到重复计数器。

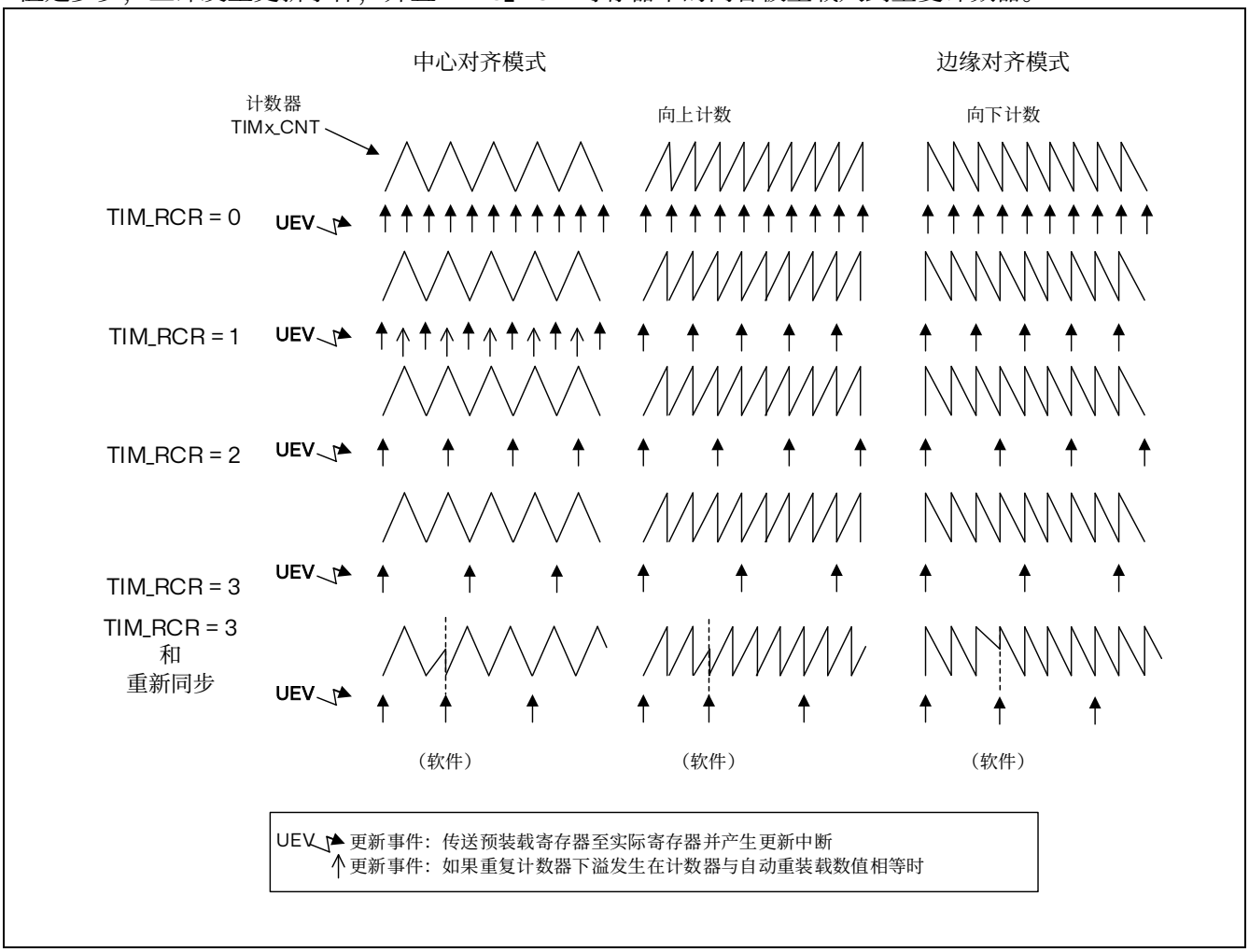



图 12.21 不同模式下更新速率的例子，及 TIM8_RCR 的寄存器设置

### 12.3.4 时钟选择

计数器时钟可由下列时钟源提供：

- 内部时钟（CK_INT）
- 外部时钟模式 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。如可以配置一个定时器 Timer1 而作为另一个定时器 Timer2 的预分频器。

**内部时钟源（CK_INT）**

如果禁止了从模式控制器（SMS=000），则 CEN、DIR（TIM8_CR1 寄存器）和 UG 位（TIM8_EGR 寄存器）是事实上的控制位，并且只能被软件修改（UG 位仍被自动清除）。只要 CEN 位被写成‘1’，预分频器的时钟就由内部时钟 CK_INT 提供。

下图显示控制电路和向上计数器在一般模式下，不带预分频器时的操作。

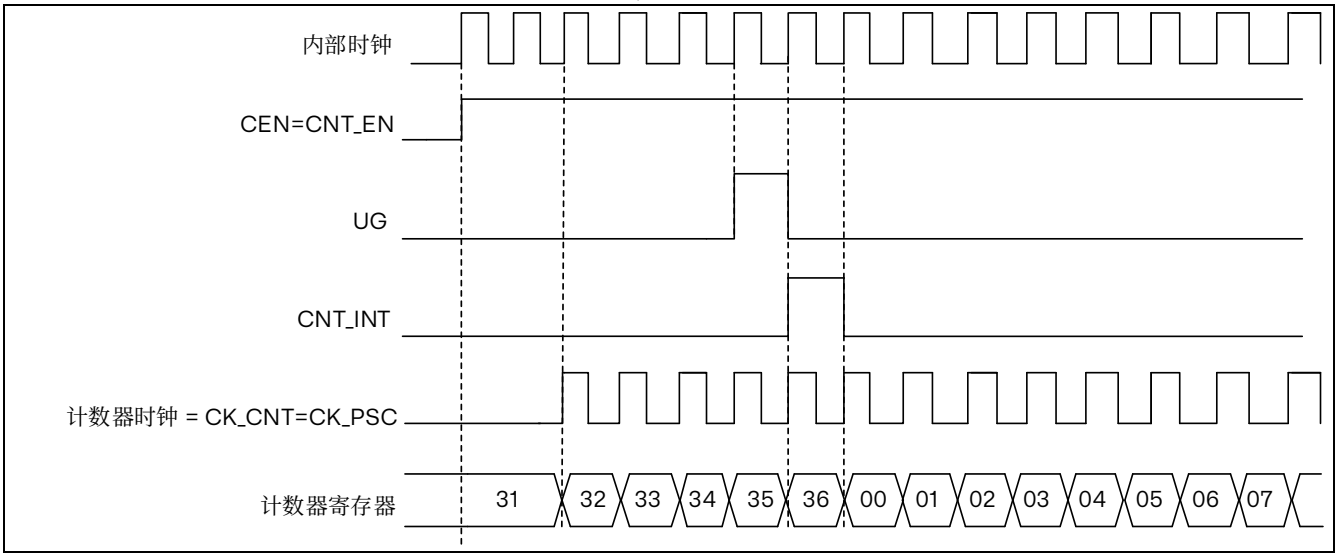



图 12.22 一般模式下的控制电路，内部时钟分频因子为 1

**外部时钟模式 1**

当 TIM8_SMCR 寄存器的 SMS=111 时，此模式被选中。计数器可以在选定输入端的每个上升沿或下降沿计数。

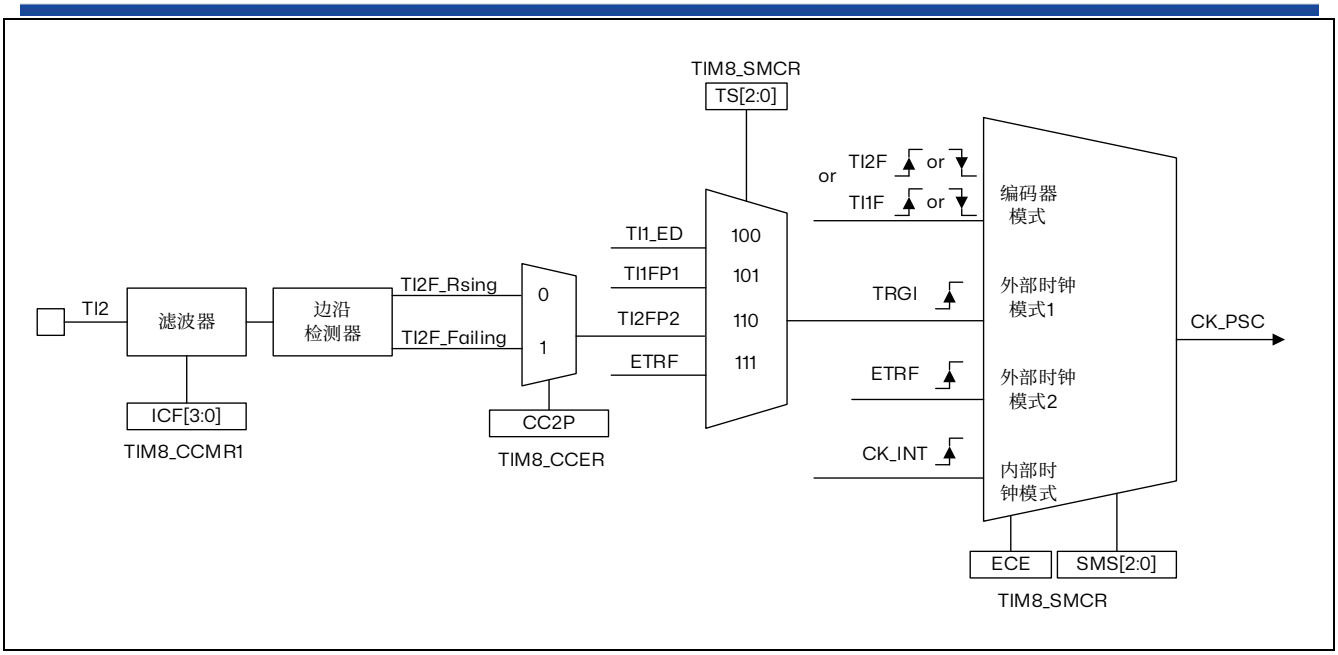



图 12.23 TI2 外部时钟连接例子

例如，要配置向上计数器在 T12 输入端的上升沿计数，使用下列步骤：
配置 TIM8_CCMR1 寄存器 CC2S=01，配置通道 2 检测 TI2 输入的上升沿
配置 TIM8_CCMR1 寄存器的 IC2F[3:0]，选择输入滤波器带宽（如果不需要滤波器，保持 IC2F=0000）
配置 TIM8_CCER 寄存器的 CC2P=0，选定上升沿极性
配置 TIM8_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1
配置 TIM8_SMCR 寄存器中的 TS=110，选定 TI2 作为触发输入源
设置 TIM8_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*
*当上升沿出现在 TI2，计数器计数一次，且 TIF 标志被设置。在 TI2 的上升沿和计数器实际时钟之间的延时，取决于在 TI2 输入端的重新同步电路。*

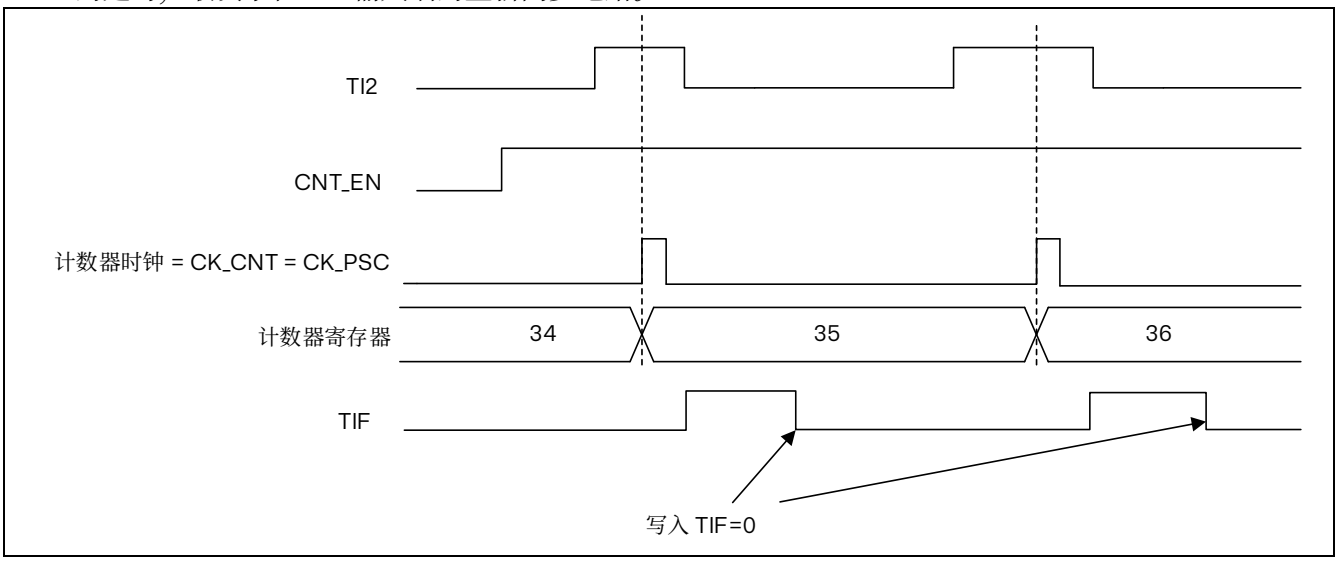



图 12.24 外部时钟模式 1 下的控制电路

**外部时钟模式 2**

选定此模式的方法为：令 TIM8_SMCR 寄存器中的 ECE=1
计数器能够在外部触发 ETR 的每一个上升沿或下降沿计数。
下图是外部触发输入的框图

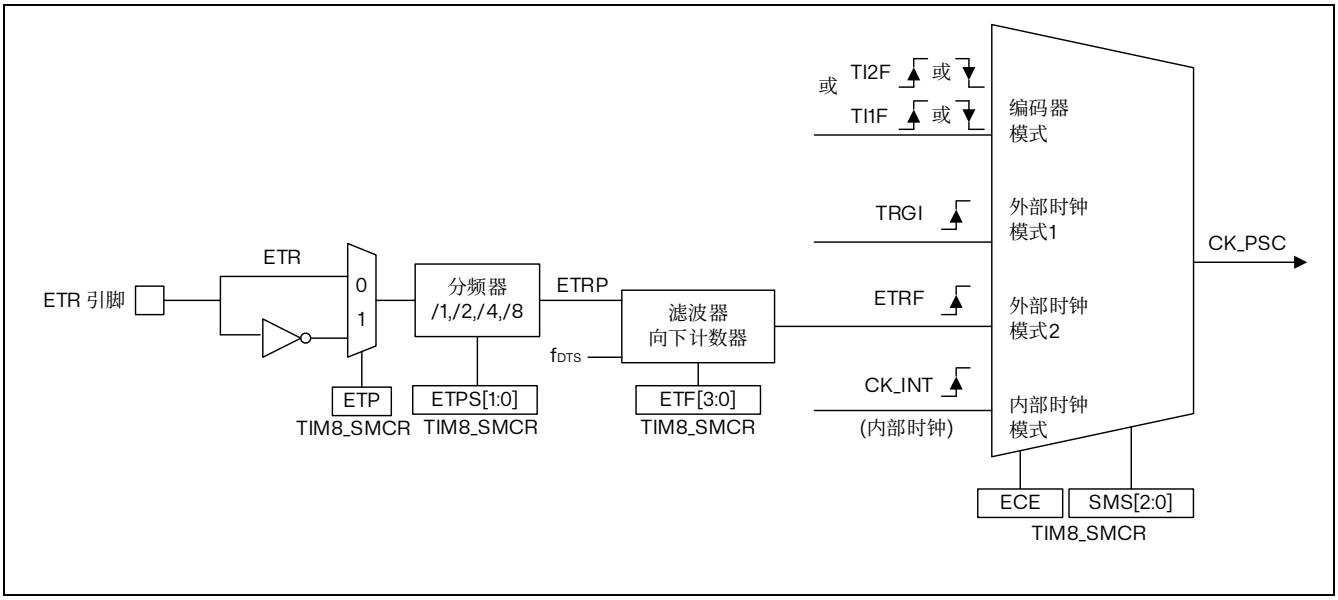



图 12.25 外部触发输入框图

例如，要配置在 ETR 下每 2 个上升沿计数一次的向上计数器，使用下列步骤：
本例中不需要滤波器，置 TIM8_SMCR 寄存器中的 ETF[3:0]=0000
设置预分频器，置 TIM8_SMCR 寄存器中的 ETPS[1:0]=01
选择 ETR 的上升沿检测，置 TIM8_SMCR 寄存器中的 ETP=0
开启外部时钟模式 2，写 TIM8_SMCR 寄存器中的 ECE=1
启动计数器，写 TIM8_CR1 寄存器中的 CEN=1
计数器在每 2 个 ETR 上升沿计数一次。
在 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。

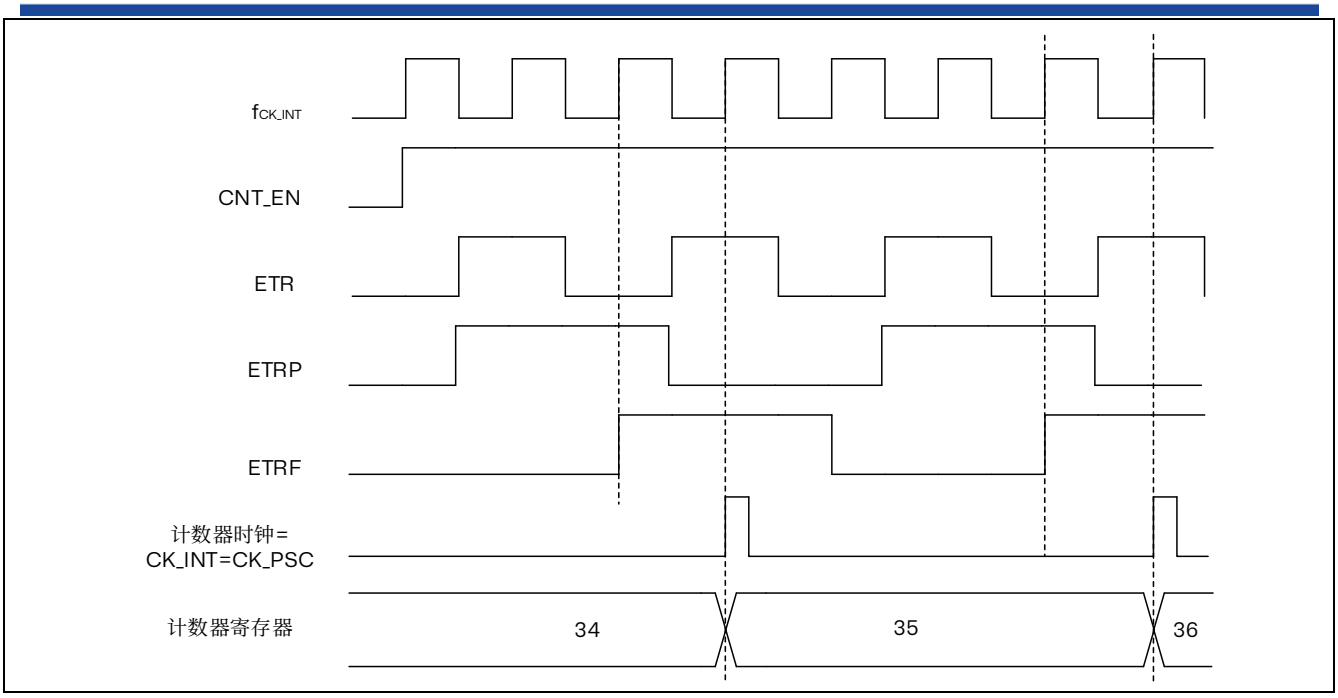



图 12.26 外部时钟模式 2 下的控制电路

### 12.3.5 捕获、比较通道

每一个捕获/比较通道都是围绕着一个捕获/比较寄存器（包含影子寄存器），包括捕获的输入部分（数字滤波、多路复用和预分频器），和输出部分（比较器和输出控制）。

图 12.27 至 12.30 是一个捕获、比较通道概览。

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

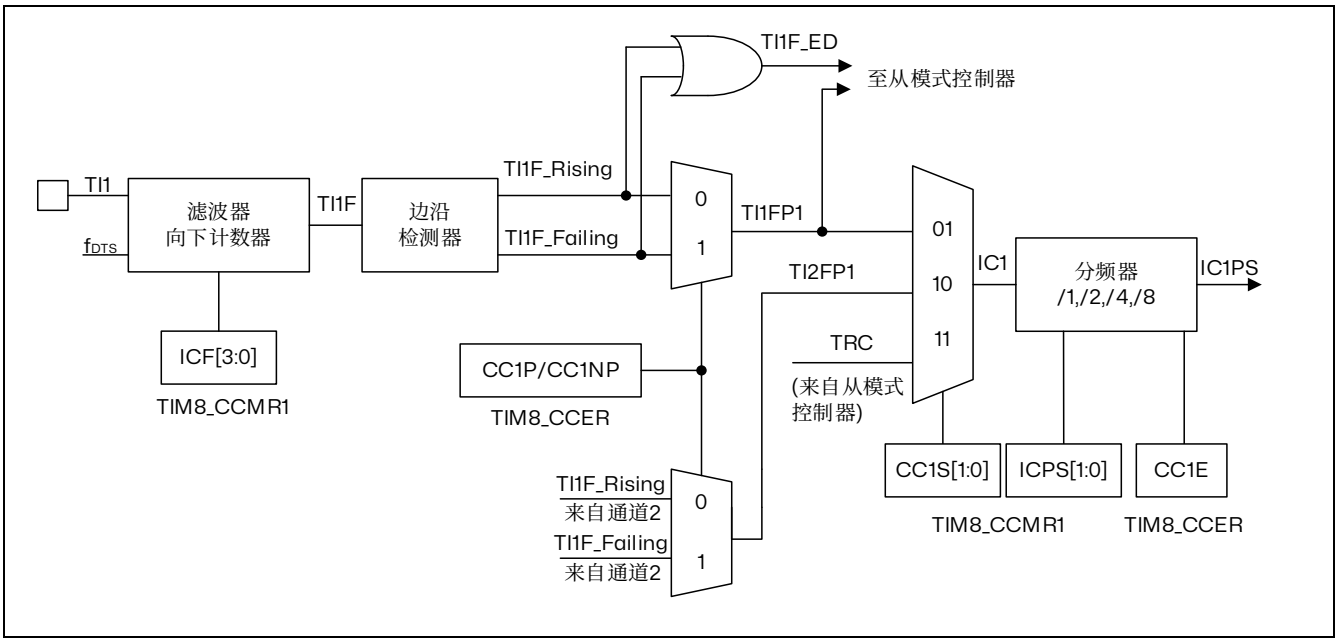



图 12.27 捕获、比较通道（如：通道一输入部分）

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

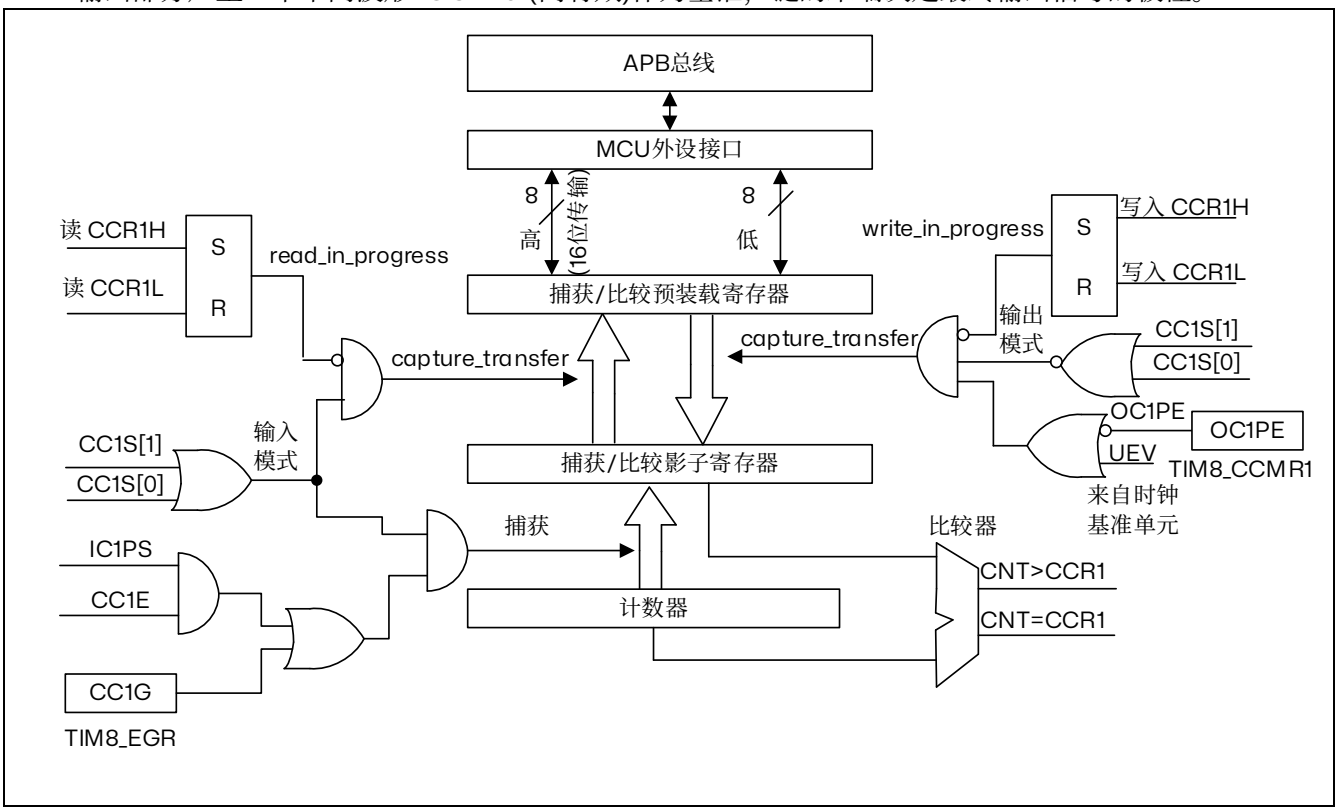



图 12.28 捕获/比较通道 1 的主电路

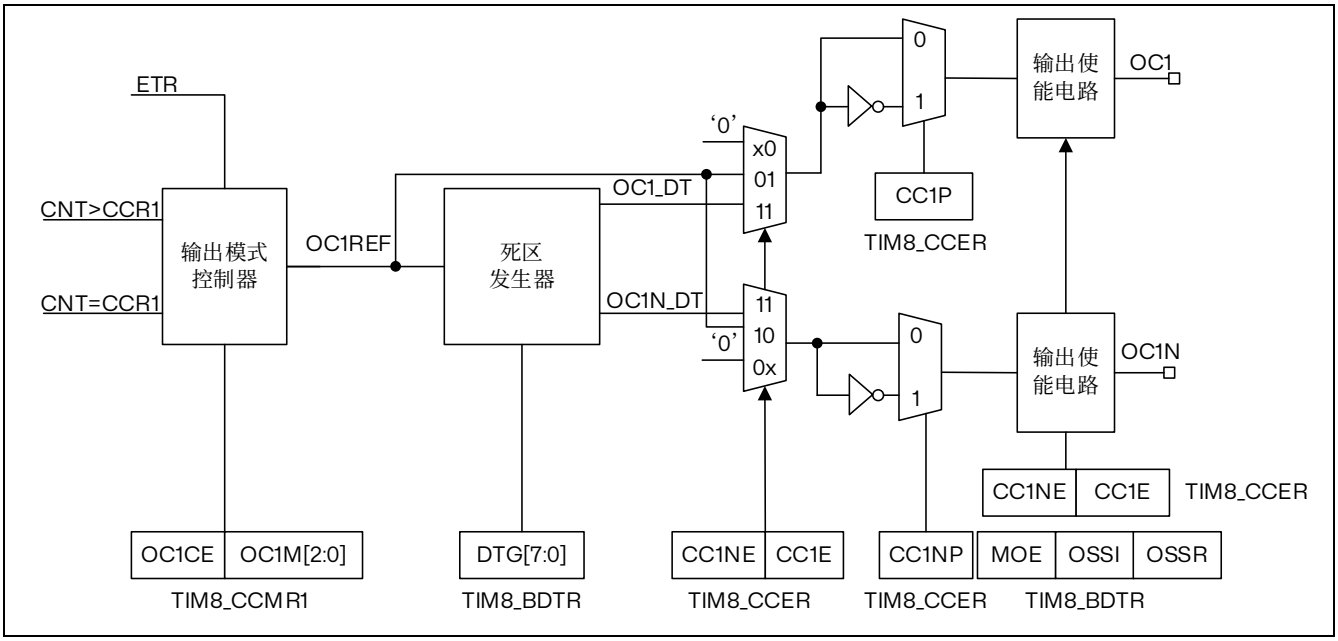



图 12.29 捕获/比较通道的输出部分（通道 1 至 3）

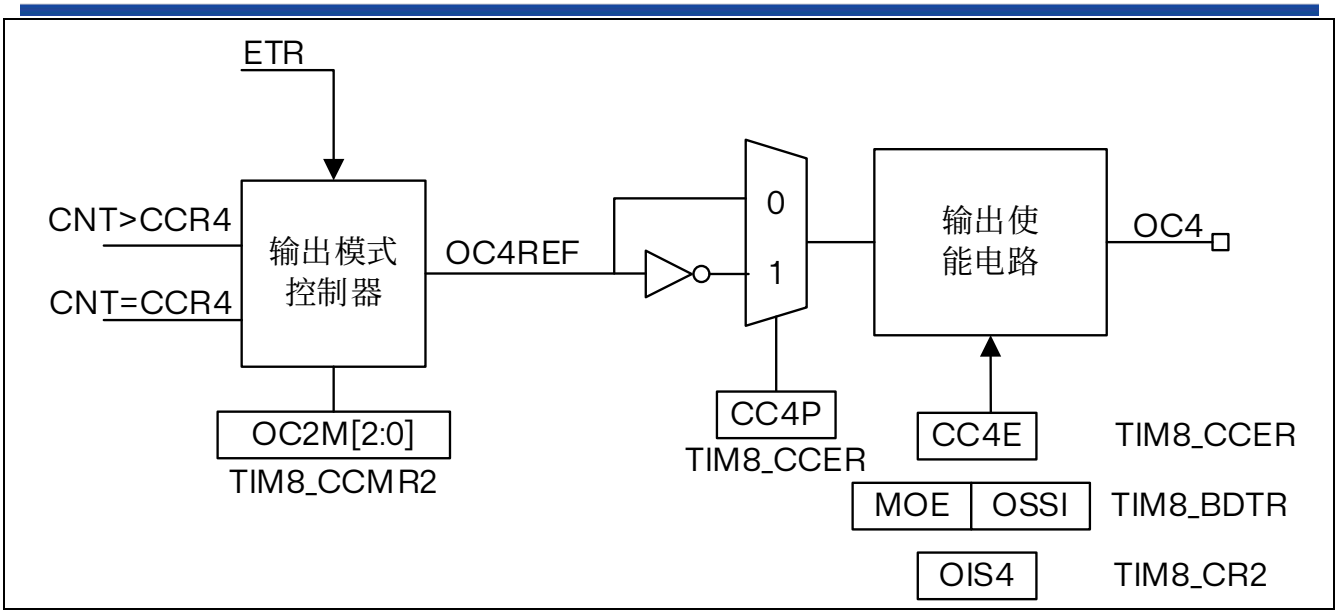



图 12.30 捕获/比较通道的输出部分（通道 4 和 5）

捕获/比较模块由一个预装载寄存器和一个影子寄存器组成。读写过程仅操作预装载寄存器。
在捕获模式下，捕获发生在影子寄存器上，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影子寄存器中，然后影子寄存器的内容和计数器进行比较。

### 12.3.6 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器（TIM8_CCRx）中。当发生捕获事件时，相应的 CCxIF 标志（TIM8_SR 寄存器）被置 1，如果开放了中断或者 DMA 操作，则将产生中断或者 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（TIM8_SR 寄存器）被置 1。写 CCxIF=0 可清除 CCxIF，或读取存储在 TIM8_CCRx 寄存器中的捕获数据也可清除 CCxIF。写 CCxOF=0 可清除 CCxOF。以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 TIM8_CCR1 寄存器中，步骤如下：

- 选择有效输入端：TIM8_CCR1 必须连接到 TI1 输入，所以写入 TIM8_CCR1 寄存器中的 CC1S=01，只要 CC1S 不为‘00’，通道被配置为输入，并且 TIM8_CCR1 寄存器变为只读。
- 根据输入信号的特点，配置输入滤波器为所需的带宽（即输入为 TIx 时，输入滤波器控制位是 TIM8_CCMRx 寄存器中的 ICxF 位）。假设输入信号在最多 5 个内部时钟周期的时间内抖动，我们须配置滤波器的带宽长于 5 个时钟周期；因此我们可以（以 fDTS 频率）连续采样 8 次，以确认在 TI1 上一次真实的边沿变换，即在 TIM8_CCMR1 寄存器中写入 IC1F=0011。
- 选择 TI1 通道的有效转换边沿，在 TIM8_CCER 寄存器中写入 CC1P=0（上升沿）。
- 配置输入预分频器。在本例中，我们希望捕获发生在每一个有效的电平转换时刻，因此预分频器被禁止（写 TIM8_CCMR1 寄存器的 IC1PS=00）。
- 设置 TIM8_CCER 寄存器的 CC1E=1，允许捕获计数器的值到捕获寄存器中。
- 如果需要，通过设置 TIM8_DIER 寄存器中的 CC1IE 位允许相关中断请求，通过设置 TIM8_DIER 寄存器中的 CC1DE 位允许 DMA 请求。当发生一个输入捕获时：
- 产生有效的电平转换时，计数器的值被传送到 TIM8_CCR1 寄存器。
- CC1IF 标志被设置（中断标志）。当发生至少 2 个连续的捕获时，而 CC1IF 未曾被清除，CC1OF 也被置 1。
- 如设置了 CC1IE 位，则会产生一个中断。

- 如设置了 CC1DE 位，则还会产生一个 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，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

*注：设置*TIM8_EGR *寄存器中相应的*CCxG *位，可以通过软件产生输入捕获中断和/或*DMA *请求。*

### 12.3.7 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：

- 两个 ICx 信号被映射至同一个 TIx 输入。
- 这 2 个 ICx 信号为边沿有效，但是极性相反。
- 其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式。例如，你需要测量输入到 TI1 上的 PWM 信号的长度（TIM8_CCR1 寄存器）和占空比（TIM8_CCR2 寄存器），具体步骤如下（取决于 CK_INT 的频率和预分频器的值）
- 选择 TIM8_CCR1 的有效输入：置 TIM8_CCMR1 寄存器的 CC1S=01（选中 TI1）。
- 选择 TI1FP1 的有效极性（用来捕获数据到 TIM8_CCR1 中和清除计数器）：置 CC1P=0（上升沿有效）。
- 选择 TIM8_CCR2 的有效输入：置 TIM8_CCMR1 寄存器的 CC2S=10（选中 TI1）。
- 选择 TI1FP2 的有效极性（捕获数据到 TIM8_CCR2）：置 CC2P=1（下降沿有效）。
- 选择有效的触发输入信号：置 TIM8_SMCR 寄存器中的 TS=101（选择 TI1FP1）。
- 配置从模式控制器为复位模式：置 TIM8_SMCR 中的 SMS=100。
- 使能捕获：置 TIM8_CCER 寄存器中 CC1E=1 且 CC2E=1。

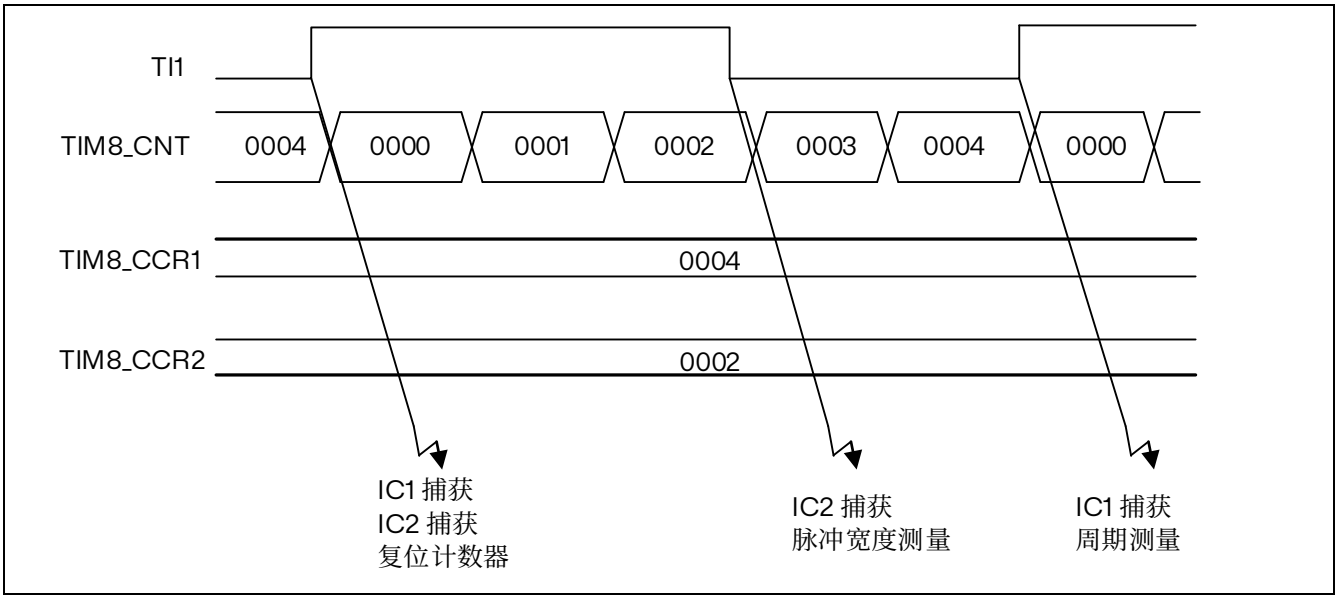



图 12.31 PWM 输入模式时序

因为只有 TI1FP1 和 TI2FP2 连到了从模式控制器，所以 PWM 输入模式只能使用 TIM8_CH1/TIM8_CH2 信号。

### 12.3.8 强制输出模式

在输出模式（TIM8_CCMRx 寄存器中 CCxS=00）下，输出比较信号（OCxREF 和相应的 OCx/OCxN）能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。置 TIM8_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号（OCxREF/OCx）为有效状

态。这样 OCxREF 被强置为高电平（OCxREF 始终为高电平有效），同时 OCx 得到 CCxP 极性相反的信号。

例如：CCxP=0（OCx 高电平有效），则 OCx 被强置为高电平。

置 TIM8_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIM8_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。这将会在下面的输出比较模式一节中介绍。

### 12.3.9 输出比较模式

此项功能是用来控制一个输出波形，或者指示一段给定的的时间已经到时。

计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

- 将输出比较模式（TIM8_CCMRx 寄存器中的 OCxM 位）和输出极性（TIM8_CCER 寄存器中的 CCxP 位）定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平（OCxM=000）、被设置成有效电平（OCxM=001）、被设置成无效电平（OCxM=010）或进行翻转（OCxM=011）。
- 设置中断状态寄存器中的标志位（TIM8_SR 寄存器中的 CCxIF 位）。
- 若设置了相应的中断屏蔽（TIM8_DIER 寄存器中的 CCxIE 位），则产生一个中断。
- 若设置了相应的使能位（TIM8_DIER 寄存器中的 CCxDE 位，TIM8_CR2 寄存器中的 CCDS 位选择 DMA 请求功能），则产生一个 DMA 请求。

TIM8_CCMRx 中的 OCxPE 位选择 TIM8_CCRx 寄存器是否需要使用预装载寄存器。

在输出比较模式下，更新事件 UEV 对 OCxREF 和 OCx 输出没有影响。

同步的精度可以达到计数器的一个计数周期。输出比较模式（在单脉冲模式下）也能用来输出一个单脉冲。

输出比较模式的配置步骤：

1. 选择计数器时钟（内部，外部，预分频器）。
2. 将相应的数据写入 TIM8_ARR 和 TIM8_CCRx 寄存器中。
3. 如果要产生一个中断请求，设置 CCxIE 位。
4. 选择输出模式，例如：
   - 要求计数器与 CCRx 匹配时翻转 OCx 的输出引脚，设置 OCxM=011
   - 置 OCxPE = 0 禁用预装载寄存器
   - 置 CCxP = 0 选择极性为高电平有效
   - 置 CCxE = 1 使能输出
5. 设置 TIM8_CR1 寄存器的 CEN 位启动计数器

TIM8_CCR 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器（OCxPE= ‘0’，否则 TIM8_CCRx 的影子寄存器只能在发生下一次更新事件时被更新）。下图给出了一个例子。

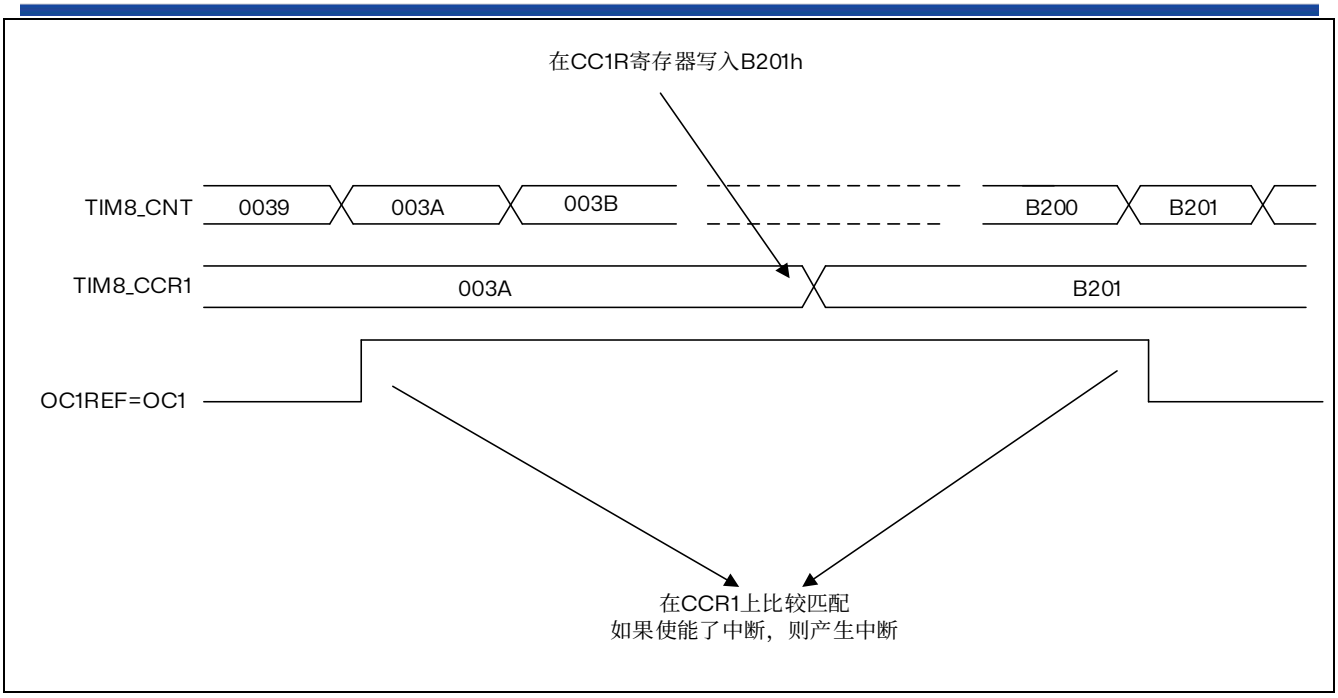



图 12.32 输出比较模式，翻转 OC1

### 12.3.10 PWM 模式

脉冲宽度调制模式可以产生一个由 TIM8_ARR 寄存器确定频率、由 TIM8_CCRx 寄存器确定占空比的信号。

在 TIM8_CCMRx 寄存器中的 OCxM 位写入'110'（PWM 模式 1）或'111'（PWM 模式 2），能够独立地设置每个 OCx 输出通道产生一路 PWM。必须通过设置 TIM8_CCMRx 寄存器的 OCxPE 位使能相应的预装载寄存器，最后还要设置 TIM8_CR1 寄存器的 ARPE 位，（在向上计数或中心对称模式中）使能自动重装载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 TIM8_EGR 寄存器中的 UG 位来初始化所有的寄存器。

OCx 的极性可以通过软件在 TIM8_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。OCx 的输出使能通过（TIM8_CCER 和 TIM8_BDTR 寄存器中）CCxE、CCxNE、MOE、OSSI 和 OSSR 位的组合控制。

在 PWM 模式（模式 1 或模式 2）下，TIM8_CNT 和 TIM8_CCRx 始终在进行比较，（依据计数器的计数方向）以确定是否符合 TIM8_CCRx≤TIM8_CNT 或者 TIM8_CNT≤TIM8_CCRx。

根据 TIM8_CR1 寄存器中 CMS 位的状态，定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

**PWM 边沿对齐模式**

- 向上计数配置

当 TIM8_CR1 寄存器中的 DIR 位为低的时候执行向上计数。下面是一个 PWM 模式 1 的例子。当 TIM8_CNTT<TIM8_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIM8_CCRx 中的比较值大于自动重装载值（TIM8_ARR），则 OCxREF 保持为 '1'。如果比较值为 0，则 OCxREF 保持为 '0'。下图为 TIM8_ARR=8 时边沿对齐的 PWM 波形实例。

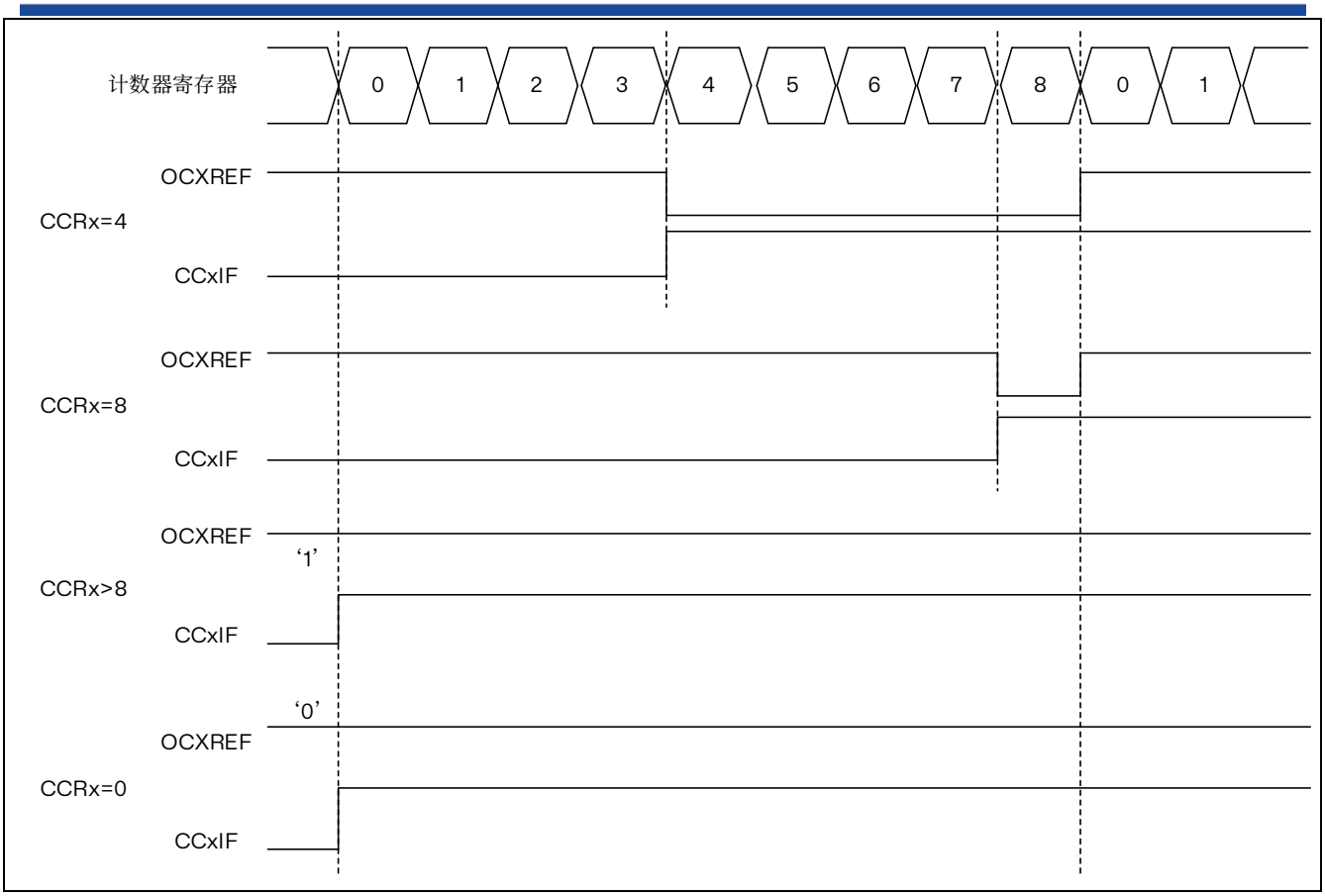



图 12.33 边沿对齐的 PWM 波形（ARR=8）

- 向下计数的配置

当 TIM8_CR1 寄存器的 DIR 位为高时执行向下计数。在 PWM 模式 1，当 TIM8_CNT>TIM8_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIM8_CCRx 中的比较值大于 TIM8_ARR 中的自动重装载值，则 OCxREF 保持为‘1’。该模式下不能产生 0%的 PWM 波形。

**PWM 中央对齐模式**

当 TIM8_CR1 寄存器中的 CMS 位不为‘00’时为中央对齐模式（所有其他的配置对 OCxREF/OCx 信号都有相同的作用）。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。TIM8_CR1 寄存器中的计数方向位（DIR）由硬件更新，不要用软件修改它。参看 12.3.2 节的中央对齐模式。下图给出了一些中央对齐的 PWM 波形的例子

- TIM8_ARR=8
- PWM 模式 1
- TIM8_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

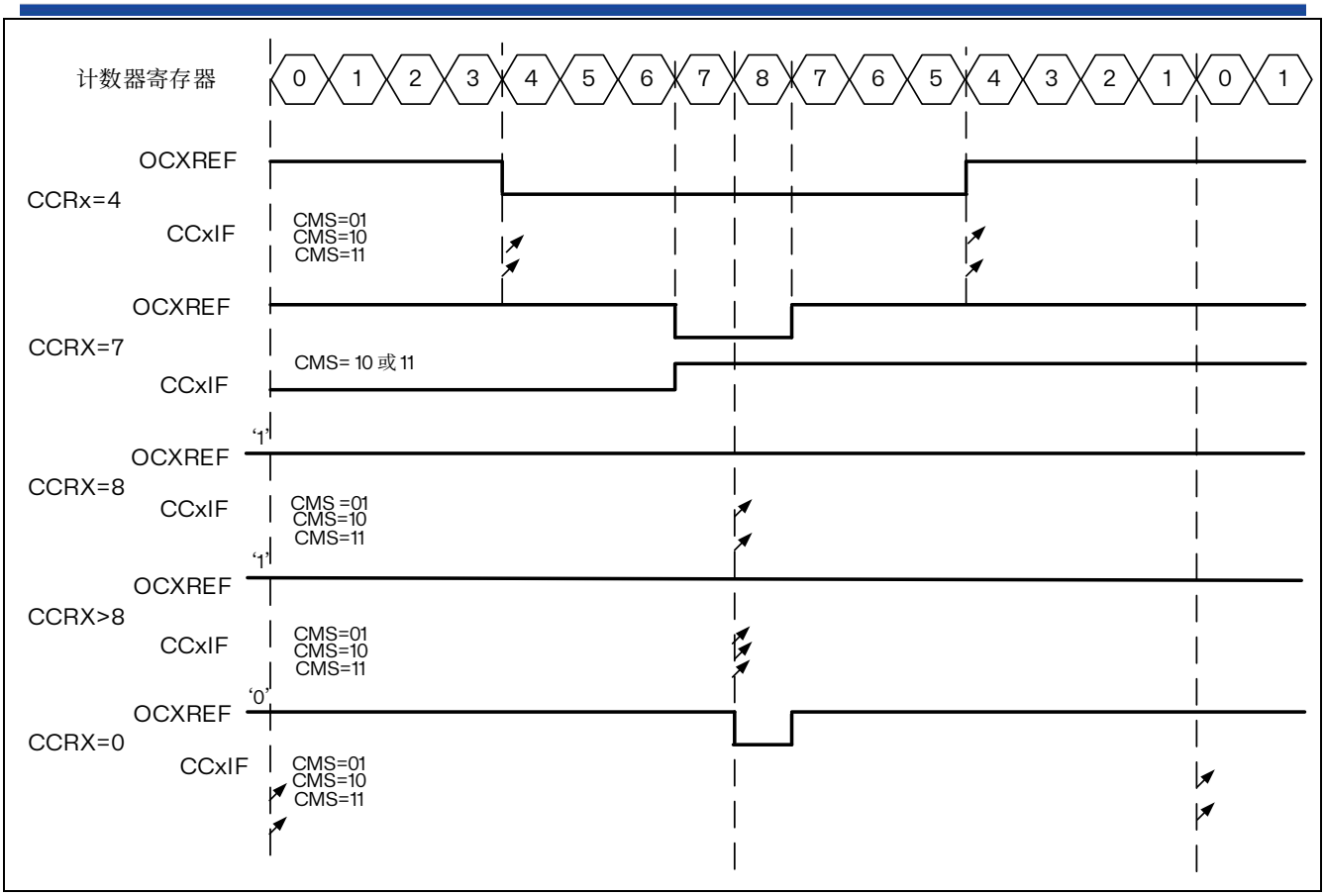



图 12.34 中央对齐的 PWM 波形（APR=8）

**使用中央对齐模式的提示：**

- 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 TIM8_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。
- 不推荐当运行在中央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加载的值（TIM8_CNT>TIM8_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 TIM8_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。
- 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新（设置 TIM8_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。

### 12.3.11 非对称 PWM 模式

**PWM 边沿对齐模式**

- 向上计数配置

当 TIM8_CR1 寄存器中的 DIR 位为低的时候执行向上计数。当 TIM8_CR1 寄存器中的 CCRxN 位为‘1’时配置位非对称模式。下面是一个 PWM 模式 1 的例子。

当 TIM8_CCRx<=TIM8_CNT<TIM8_CCRxN 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIM8_CCRx 和 TIM8_CCRxN 中的比较值大于自动重装载值（TIM8_ARR），则 OCxREF 保持为‘1’。如果 TIM8_CCRx 中的比较值大于等于 TIM8_CCRxN 中的比较值或者比较值都为 0，则 OCxREF 保持为

‘0’。下图为 TIM8_ARR=8 时边沿对齐的 PWM 波形实例。

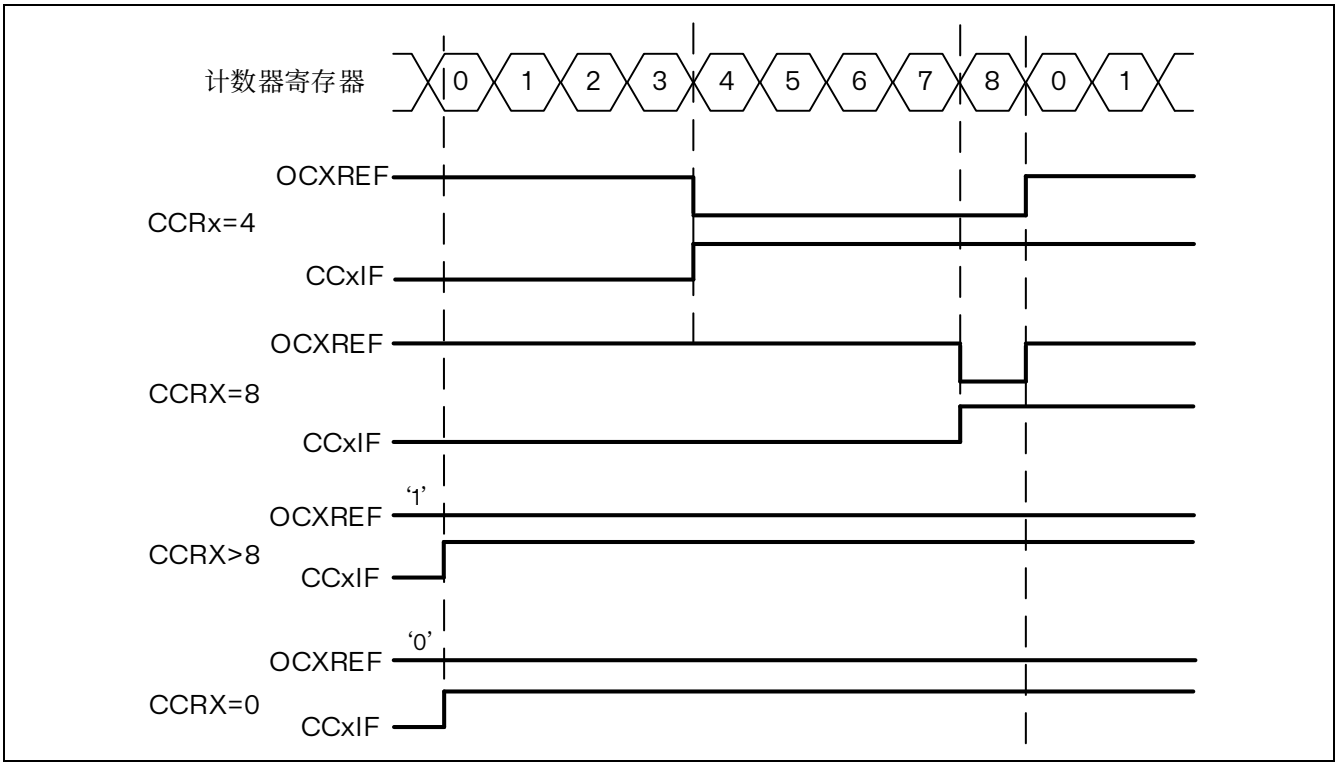



图 12.35 边沿对齐向上计数的 PWM 波形（ARR=8）

- 向下计数的配置

当 TIM8_CR1 寄存器中的 DIR 位为高的时候执行向下计数。当 TIM8_CR1 寄存器中的 CCRxN 位为‘1’时配置位非对称模式。

在 PWM 模式 1，当 TIM8_CCRxN <=TIM8_CNT<TIM8_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIM8_CCRx 和 TIM8_CCRxN 中的比较值大于自动重装载值（TIM8_ARR），则 OCxREF 保持为‘1’。该模式下不能产生 0%的 PWM 波形。

**PWM 中央对齐模式**

当 TIM8_CR1 寄存器中的 CMS 位不为‘00’时为中央对齐模式（所有其他的配置对 OCxREF/OCx 信号都有相同的作用）。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。TIM8_CR1 寄存器中的计数方向位（DIR）由硬件更新，不要用软件修改它。下图给出了一些中央对齐的 PWM 波形的例子

- TIM8_ARR=8
- PWM 模式 1
- TIM8_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

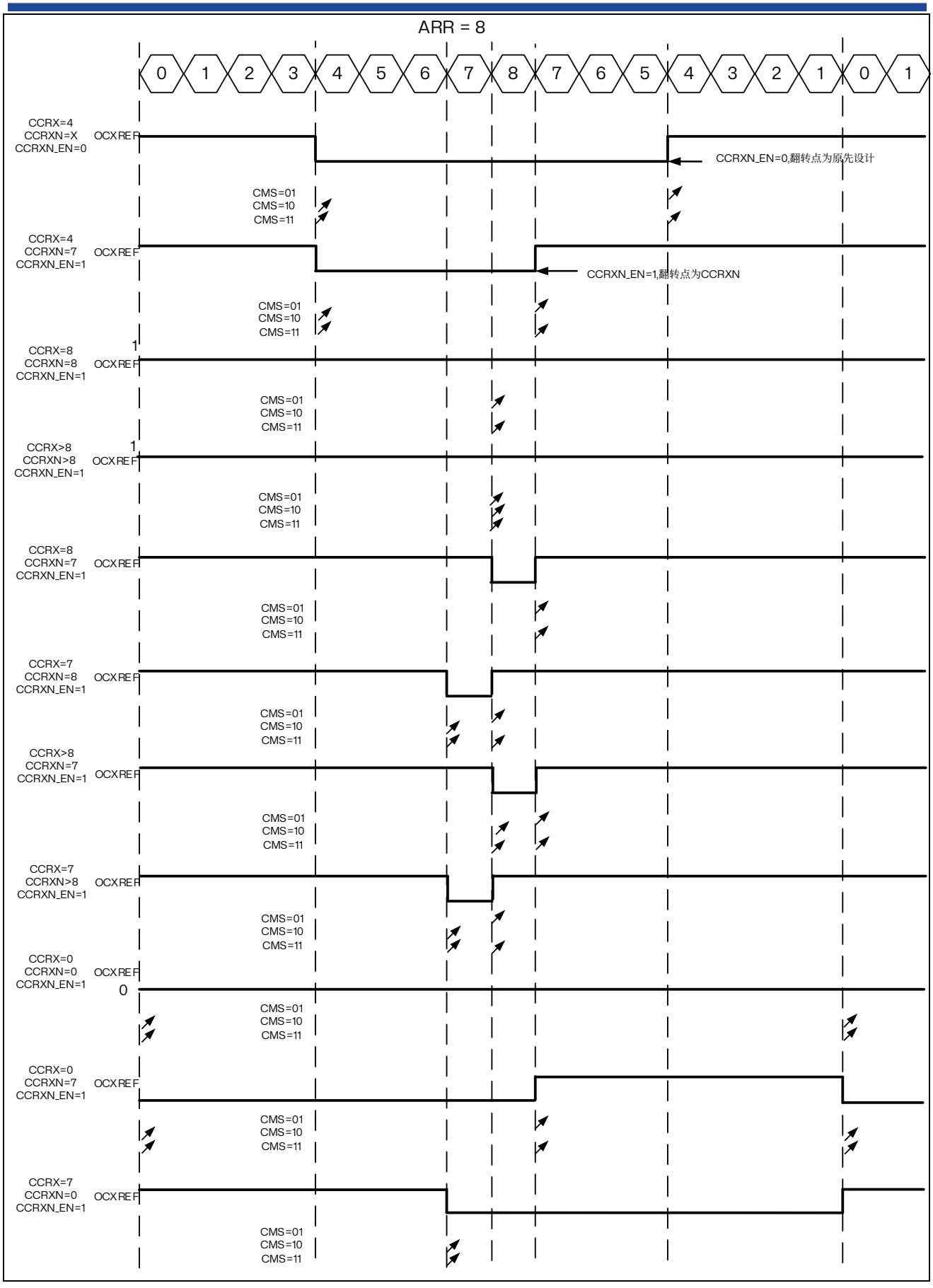



图 12.36 中央对齐的 PWM 波形（APR=8）

### 12.3.12 互补输出和死区插入

高级控制定时器（TIM8）能够输出两路互补信号，并且能够管理输出的瞬时关断和接通。这段时间通常被称为死区，用户应该根据连接的输出器件和它们的特性（电平转换的延时、电源开关的延时等）来调整死区时间。

配置 TIM8_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：TIM8_CCER 寄存器的 CCxE 和 CCxNE 位，TIM8_BDTR 和 TIM8_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，特别的是，在转换到 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 输出信号与参考信号相同，只是它的上升沿相对于参考信号的上升沿有一个延迟。
- OCxN 输出信号与参考信号相反，只是它的上升沿相对于参考信号的下降沿有一个延迟。如果延迟大于当前有效的输出宽度（OCx 或者 OCxN），则不会产生相应的脉冲。下列几张图显示了死区发生器的输出信号和当前参考信号 OCxREF 之间的关系。（假设 CCxP=0、CCxNP=0、MOE=1、CCxE=1 并且 CCxNE=1）

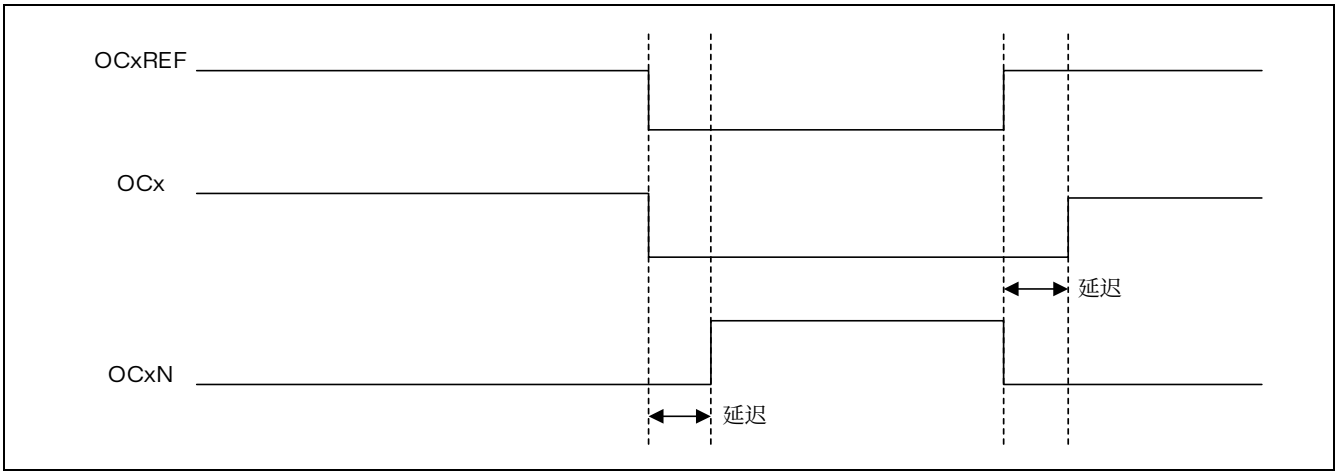



图 12.37 带死区插入的互补输出

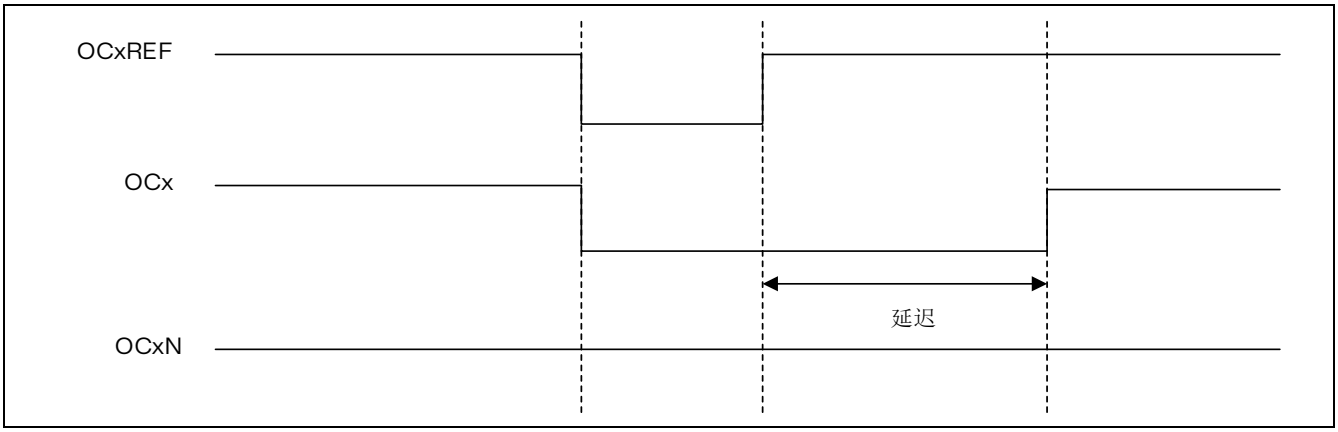



图 12.38 死区波形延迟大于负脉冲

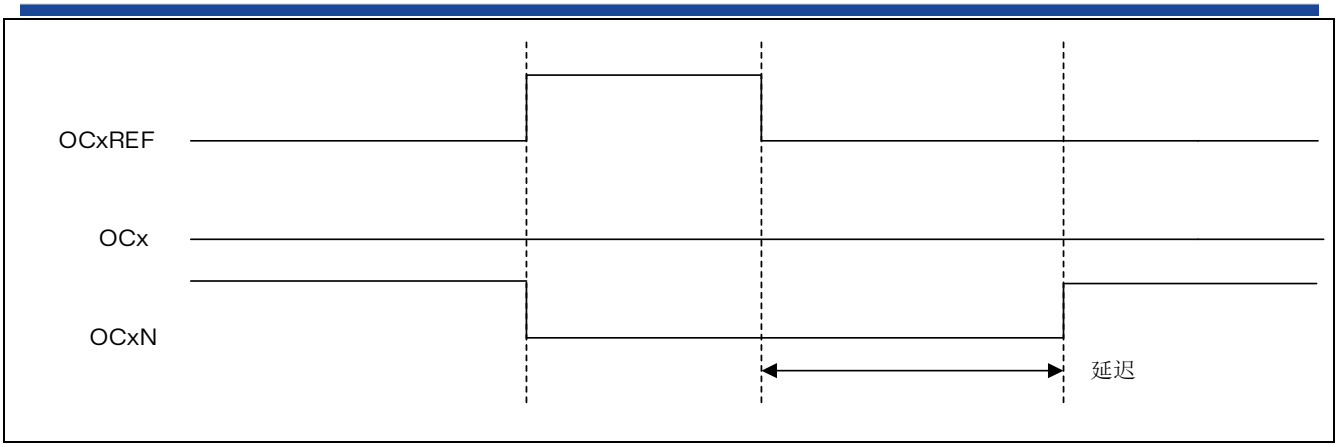



图 12.39 死区波形延迟大于正脉冲

每一个通道的死区延时都是相同的，是由 TIM8_BDTR 寄存器中的 DTG 位编程配置。

**重定向 OCxREF 到 OCx 或 OCxN**

在输出模式下（强置、输出比较或 PWM），通过配置 TIM8_CCER 寄存器的 CCxE 和 CCxNE 位，OCxREF 可以被重定向到 OCx 或者 OCxN 的输出。这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形（例如 PWM 或者静态有效电平）。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

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

### 12.3.13 使用刹车功能

当使用刹车功能时，依据相应的控制位（TIM8_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIM8_CR2 寄存器中的 OISx 和 OISxN 位），输出使能信号和无效电平都会被修改。但无论何时，OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。

刹车源既可以是刹车输入引脚又可以是一个时钟失败事件。时钟失败事件由复位时钟控制器中的时钟安全系统产生。

系统复位后，刹车电路被禁止，MOE 位为低。设置 TIM8_BDTR 寄存器中的 BKE 位可以使能刹车功能，刹车输入信号的极性可以通过配置同一个寄存器中的 BKP 位选择。BKE 和 BKP 可以同时被修改。当写入 BKE 和 BKP 位时，在真正写入之前会有 1 个 APB 时钟周期的延迟，因此需要等待一个 APB 时钟周期之后，才能正确地读回写入的位。

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 TIM8_BDTR 寄存器中）之间设置了一个再同步电路。这个再同步电路会在异步信号和同步信号之间产生延迟。特别的，如果当它为低时写 MOE=1，则读出它之前必须先插入一个延时（空指令）才能读到正确的值。这是因为写入的是异步信号而读的是同步信号。

当发生刹车时（在刹车输入端出现选定的电平），有下述动作：

- MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态（由 OSSI 位选择）。这个特性在 MCU 的振荡器关闭时依然有效。
- 一旦 MOE=0，每一个输出通道输出由 TIM8_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放使能输出，否则使能输出始终为高。

- 当使用互补输出时：
  - 输出首先被置于复位状态即无效的状态（取决于极性）。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，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。注，因为重新同步 MOE，死区时间比通常情况下长一些（大约 2 个 ck_tim 的时钟周期）。
  - 如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
- 如果设置了 TIM8_DIER 寄存器中的 BIE 位，当刹车状态标志（TIM8_SR 寄存器中的 BIF 位）为‘1’时，则产生一个中断。如果设置了 TIM8_DIER 寄存器中的 BDE 位，则产生一个 DMA 请求。
- 如果设置了 TIM8_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置‘1’；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

*注：刹车输入为电平有效。所以，当刹车输入有效时，不能同时（自动地或者通过软件）设置* MOE。*同时，状态标志* BIF *不能被清除。*

刹车由 BRK 输入产生，它的有效极性是可编程的，且由 TIM8_BDTR 寄存器中的 BKE 位开启。除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数（死区长度，OCx/OCxN 极性和被禁止的状态，OCxM 配置，刹车使能和极性）。用户可以通过 TIM8_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，参看 12.4.18 节 TIM8 刹车和死区寄存器（TIM8_BDTR）。在 MCU 复位后 LOCK 位只能被修改一次。

下图显示响应刹车的输出实例。

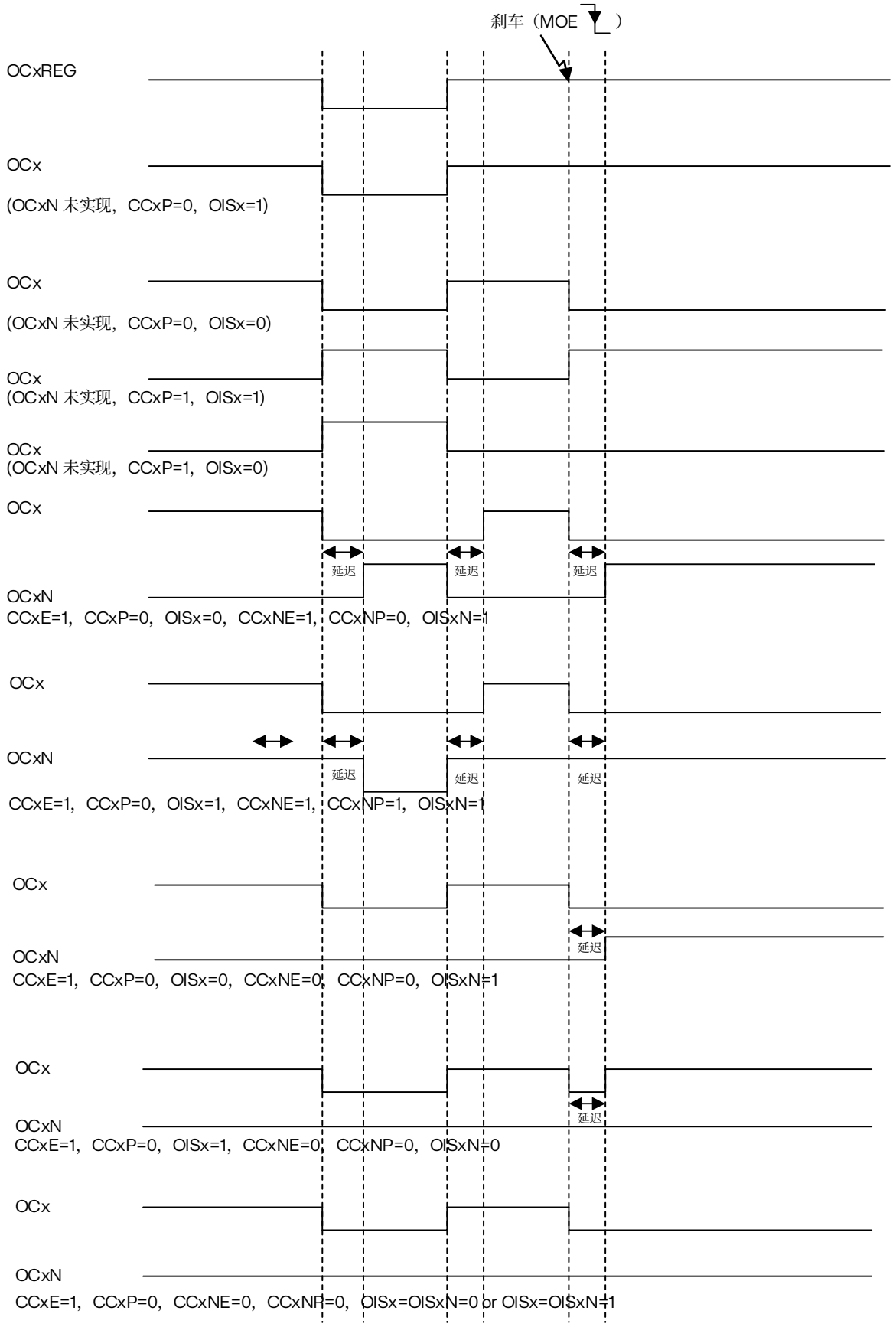



图 12.40 响应刹车的输出

### 12.3.14 在外部事件时清除 OCxREF 信号

对于一个给定的通道，设置 TIM8_CCMRx 寄存器中对应的 OCxCE 位为‘1’，能够用 ETRF 输入端的高电平把 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，而不能用于强置模式。

例如，OCxREF 信号可以联到一个比较器的输出，用于控制电流。这时，ETR 必须配置如下：

1. 外部触发预分频器必须处于关闭：TIM8_SMCR 寄存器中的 ETPS[1:0]=00。
2. 必须禁止外部时钟模式 2：TIM8_SMCR 寄存器中的 ECE=0。
3. 外部触发极性（ETP）和外部触发滤波器（ETF）可以根据需要配置。

下图显示了当 ETRF 输入变为高时，对应不同 OCxCE 的值，OCxREF 信号的动作。在这个例子中，定时器 TIM8 被置于 PWM 模式。

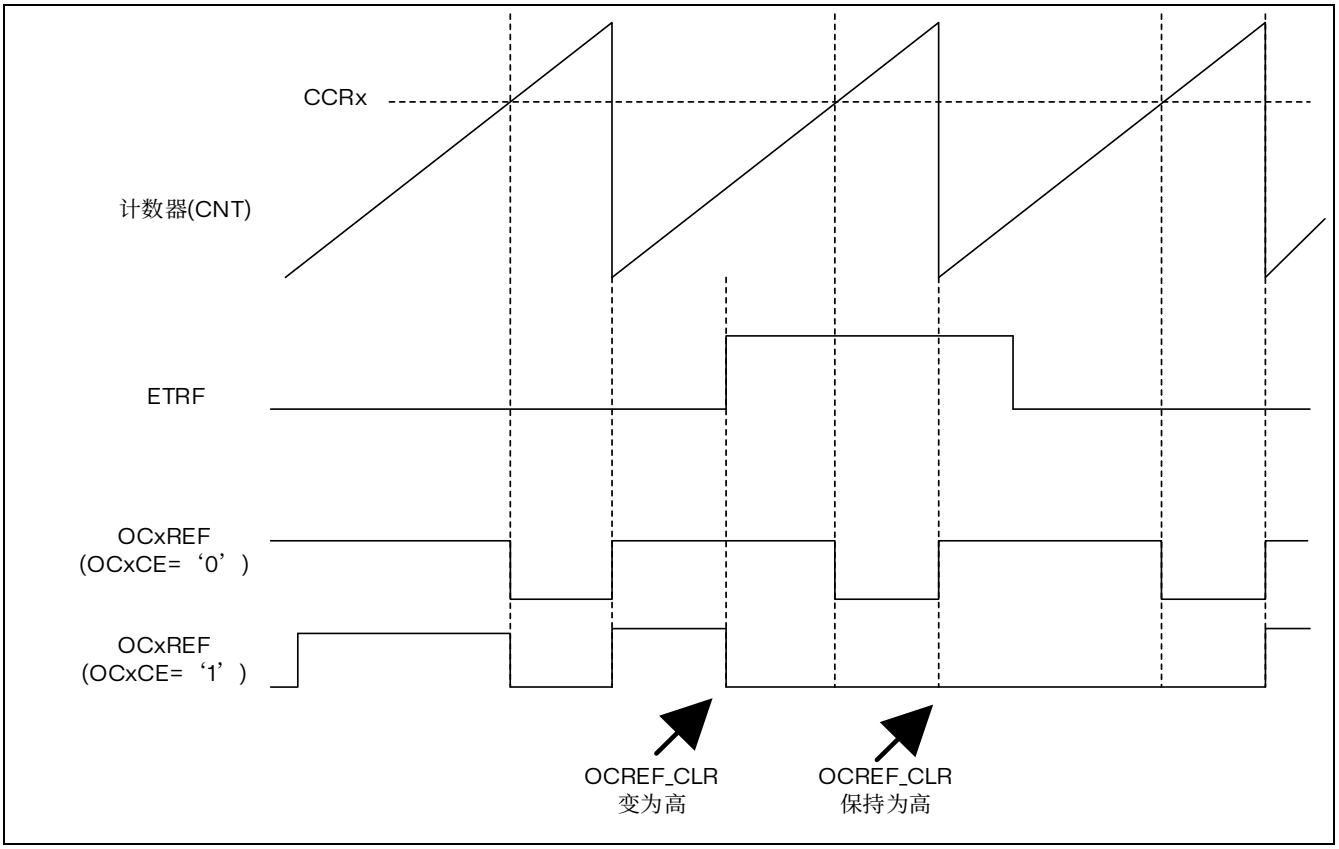



图 12.41 清除 TIM8 的 OCxREF

### 12.3.15 产生六步 PWM 输出

当在一个通道上需要互补输出时，预装载位有 OCxM、CCxE 和 CCxNE。在发生 COM 换相事件时，这些预装载位被传送到影子寄存器位。这样你就可以预先设置好下一步骤配置，并在同一个时刻同时修更改所有通道的配置。COM 可以通过设置 TIM8_EGR 寄存器的 COM 位由软件产生，或在 TRGI 上升沿由硬件产生。

当发生 COM 事件时会设置一个标志位（TIM8_SR 寄存器中的 COMIF 位），这时如果已设置了 TIM8_DIER 寄存器的 COMIE 位，则产生一个中断；如果已设置了 TIM8_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。

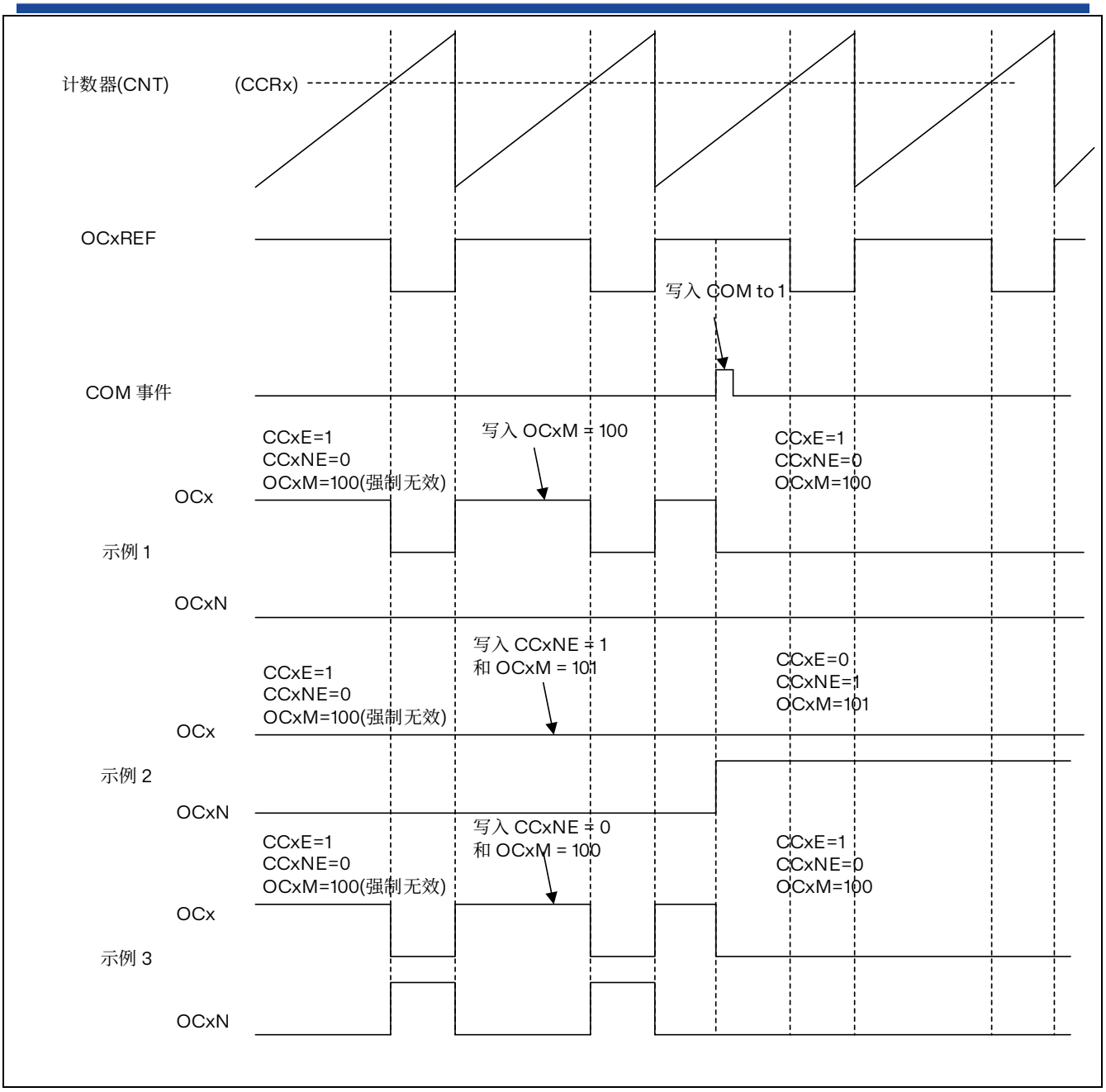



图 12.42 产生六步 PWM，使用 COM 的例子（OSSR=1）

### 12.3.16 单脉冲模式

单脉冲模式（OPM）是前述众多模式的一个特例。这种模式允许计数器响应一个激励，并在一个程序可控的延时之后产生一个脉宽可程序控制的脉冲。可以通过从模式控制器启动计数器，在输出比较模式或者 PWM 模式下产生波形。设置 TIM8_CR1 寄存器中的 OPM 位将选择单脉冲模式，这样可以让计数器自动地在产生下一个更新事件 UEV 时停止。仅当比较值与计数器的初始值不同时，才能产生一个脉冲。启动之前（当定时器正在等待触发），必须如下配置：

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

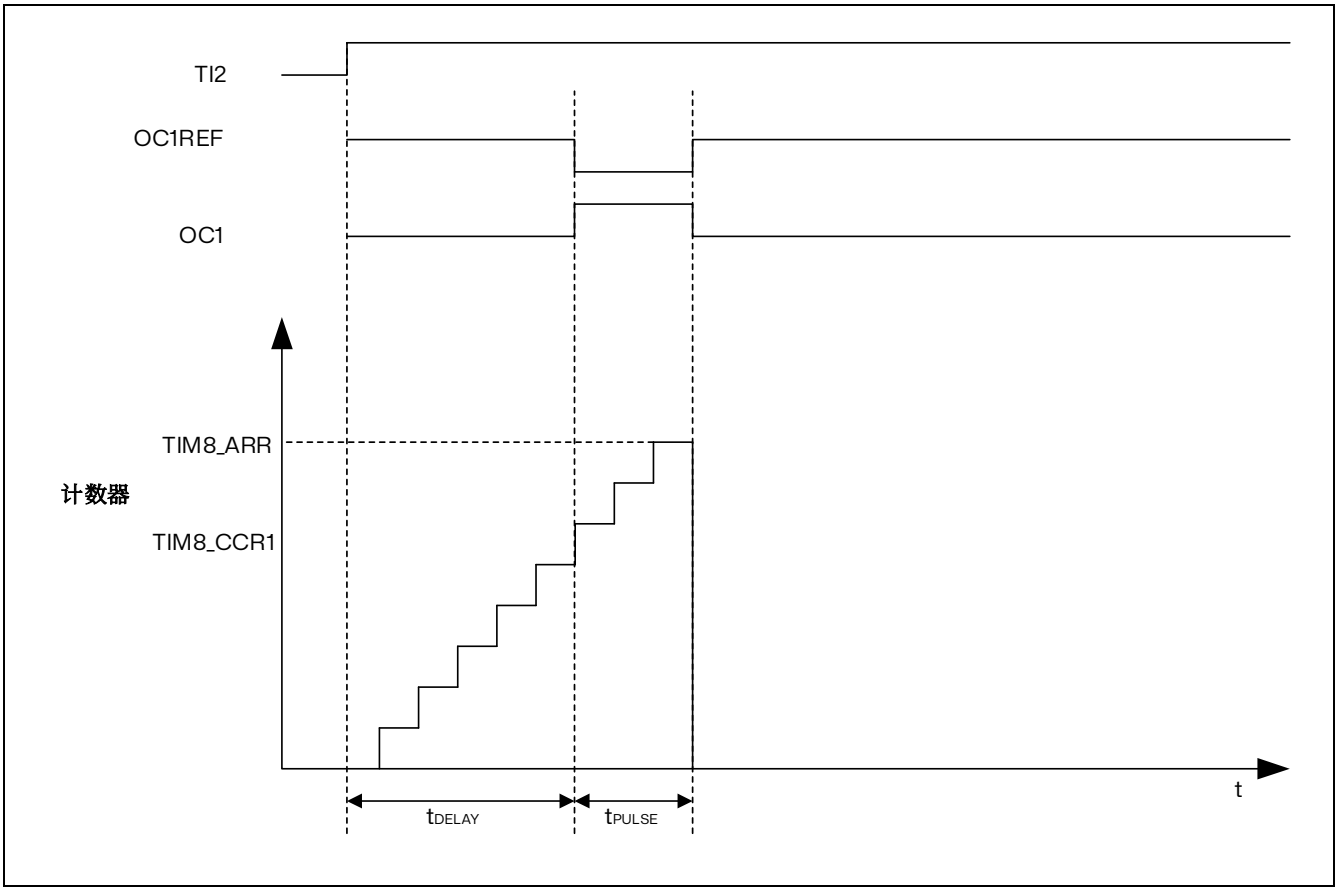



图 12.43 单脉冲模式的例子

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

假定 TI2FP2 作为触发 1:

- 置 TIM8_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。
- 置 TIM8_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。
- 置 TIM8_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发（TRGI）。
- 置 TIM8_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数值决定（要考虑时钟频率和计数器预分频器）
- $t_{DELAY}$ 由 TIM8_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（TIM8_ARR - TIM8_CCR1）。

- 假定当发生比较匹配时要产生从 0 到 1 的波形，当计数器达到预装载值时要产生一个从 1 到 0 的波形；首先要置 TIM8_CCMR1 寄存器的 OC1M=111，进入 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 TIM8_CCMR1 中的 OC1PE=1 和 TIM8_CR1 寄存器中的 ARPE；然后在 TIM8_CCR1 寄存器中填写比较值，在 TIM8_ARR 寄存器中填写自动装载值，设置 UG 位 来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，TIM8_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 TIM8_CR1 寄存器中的 OPM=1，在下一个更新事件（当计数器从自动装载值翻转到 0）时停止计数。

**特殊情况：OCx 快速使能：**

在单脉冲模式下，在 TIx 输入脚的边沿检测逻辑设置 CEN 位以启动计数器。然后计数器和比较值间的比较操作产生了输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 $t_{DELAY}$。

如果要以最小延时输出波形，可以设置 TIM8_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的结果，输出的波形与比较匹配时的波形一样。OCxFE 只在通道配置为 PWM1 和 PWM2 模式时起作用。

### 12.3.17 编码器接口模式

选择编码器接口模式的方法是：如果计数器只在 TI2 的边沿计数，则置 TIM8_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 TIM8_CCER 寄存器中的 CC1P 和 CC2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。参看表 14.1，假定计数器已经启动（TIM8_CR1 寄存器中的 CEN=1），则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号；如果没有滤波和变相，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。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 TIM8_CR1 寄存器的 DIR 位进行相应的设置。不管计数器是依靠 TI1 计数、依靠 TI2 计数或者同时依靠 TI1 和 TI2 计数，在任一输入端（TI1 或者 TI2）的跳变都会重新计算 DIR 位。

编码器接口模式基本上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 TIM8_ARR 寄存器的自动装载值之间连续计数（根据方向，或是 0 到 ARR 计数，或是 ARR 到 0 计数）。所以在开始计数之前必须配置 TIM8_ARR；同样，捕获器、比较器、预分频器、重复计数器、触发输出特性等仍工作如常。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。

表 12.1 计数方向与编码器信号的关系

| 有效边沿 | 相对信号的电平（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 |

| 有效边沿 | 相对信号的电平（TI1FP1 对应 TI2，TI2FP2 对应 TI1） | TI1FP1 信号 | | TI2FP2 信号 | |
|---|---|---|---|---|---|
| | | 上升 | 下降 | 上升 | 下降 |
| | 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，这大大增加了抗噪声干扰能力。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实例，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’（TIM8_CCMR1 寄存器，IC1FP1 映射到 TI1）
- CC2S=‘01’（TIM8_CCMR2 寄存器，IC2FP2 映射到 TI2）
- CC1P=‘0’（TIM8_CCER 寄存器，IC1FP1 不反相，IC1FP1=TI1）
- CC2P=‘0’（TIM8_CCER 寄存器，IC2FP2 不反相，IC2FP2=TI2）
- SMS=‘011’（TIM8_SMCR 寄存器，所有的输入均在上升沿和下降沿有效）
- CEN=‘1’（TIM8_CR1 寄存器，计数器使能）

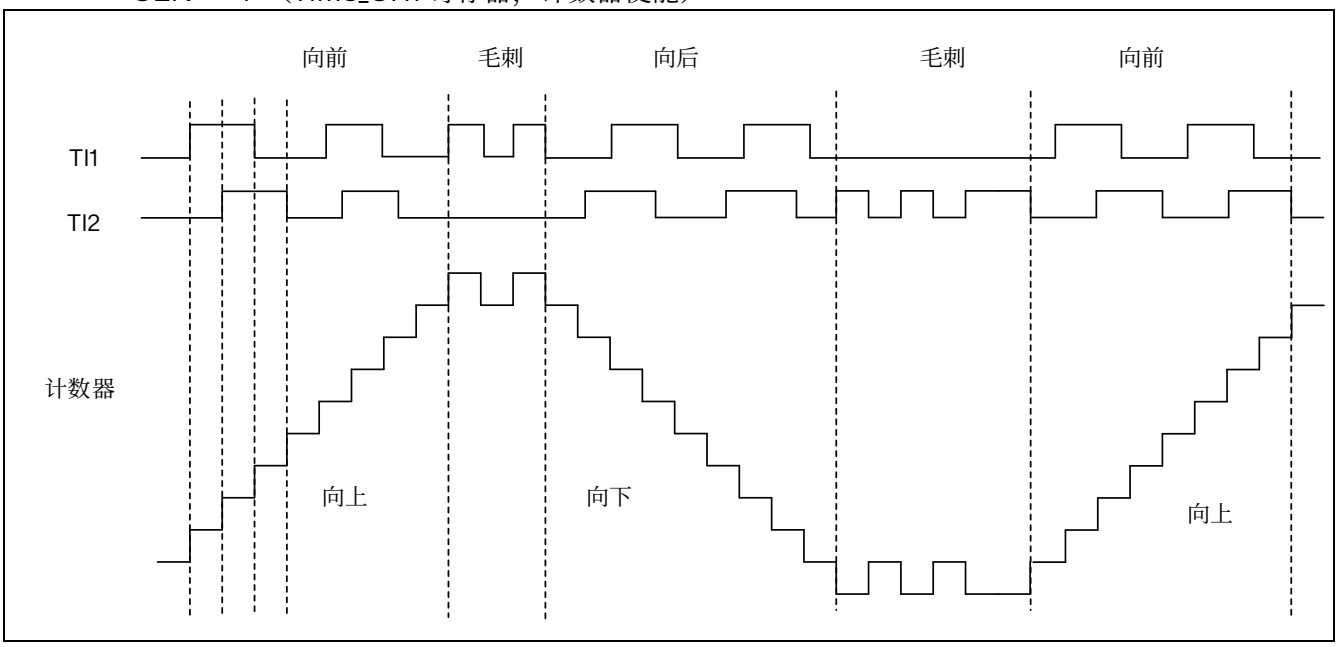



图 12.44 编码器模式下的计数器操作实例

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）

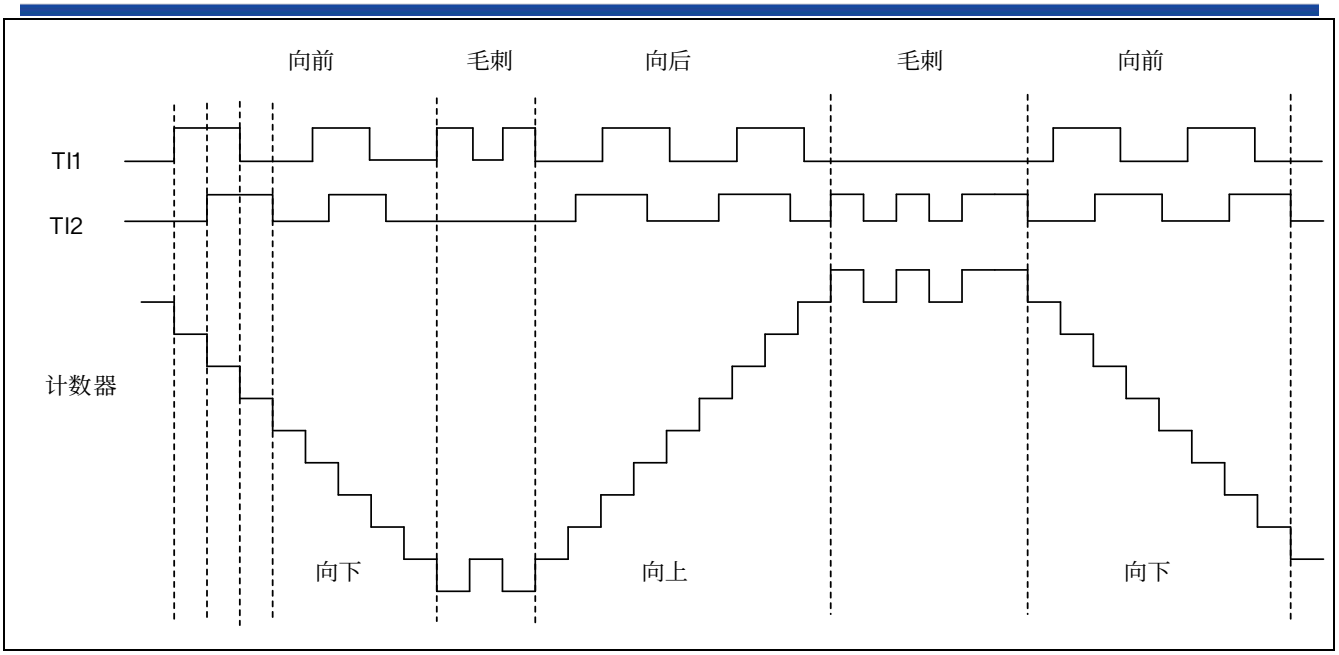



图 12.45 IC1FP1 反相的编码器接口模式实例

当定时器配置成编码器接口模式时，提供传感器当前位置的信息。使用第二个配置在捕获模式的定时器，可以测量两个编码器事件的间隔，获得动态的信息（速度，加速度，减速度）。指示机械零点的编码器输出可被用做此目的。根据两个事件间的间隔，可以按照固定的时间读出计数器。如果可能的话，你可以把计数器的值锁存到第三个输入捕获寄存器（捕获信号必须是周期的并且可以由另一个定时器产生）；也可以通过一个由实时时钟产生的 DMA 请求来读取它的值。

### 12.3.18 定时器输入异或

TIM8_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 TIM8_CH1、TIM8_CH2 和 TIM8_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。下节 13.3.18 给出了此特性用于连接霍尔传感器的例子。

### 12.3.19 与霍尔传感器接口

使用高级控制定时器（TIM1 或 TIM8）产生 PWM 信号驱动马达时，可以用另一个通用 TIM8（TIM2、TIM3、TIM4 或 TIM5）定时器作为“接口定时器”来连接霍尔传感器，见图 14.44，3 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 TIM8_CR2 寄存器中的 TI1S 位来选择），“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。每当 3 个输入之一变化时，计数器从新从 0 开始计数。这样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”上的捕获/比较通道 1 配置为捕获模式，捕获信号为 TRC（见图 76）。捕获值反映了两个输入变化间的时间延迟，给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）用于改变高级定时器 TIM1 或 TIM8 各个通道的属性，而高级控制定时器产生 PWM 信号驱动马达。因此“接口定时器”通道必须编程为在一个指定的延时（输出比较或 PWM 模式）之后产生一个正脉冲，这个脉冲通过 TRGO 输出被送到高级控制定时器 TIM1 或 TIM8。举例：霍尔输入连接到 TIM8 定时器，要求每次任一霍尔输入上发生变化之后的一个指定的时刻，改变高级控制定时器 TIM8 的 PWM 配置。

- 置 TIM8_CR2 寄存器的 TI1S 位为‘1’，配置三个定时器输入逻辑或到 TI1 输入，

- 时基编程：置 TIM8_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，它长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）：置 TIM8_CCMR1 寄存器中 CC1S=01，如果需要，还可以设置数字滤波器。
- 设置通道 2 为 PWM2 模式，并具有要求的延时：置 TIM8_CCMR1 寄存器中的 OC2M=111 和 CC2S=00。
- 选择 OC2REF 作为 TRGO 上的触发输出：置 TIM8_CR2 寄存器中的 MMS=101。

在高级控制寄存器 TIM1 中，正确的 ITR 输入必须是触发器输入，定时器被编程为产生 PWM 信号，捕获/比较控制信号为预装载的（TIM8_CR2 寄存器中 CCPC=1），同时触发输入控制 COM 事件（TIM8_CR2 寄存器中 CCUS=1）。在一次 COM 事件后，写入下一步的 PWM 控制位（CCxE、OCxM），这可以在处理 OC2REF 上升沿的中断子程序里实现。

下图显示了这个实例

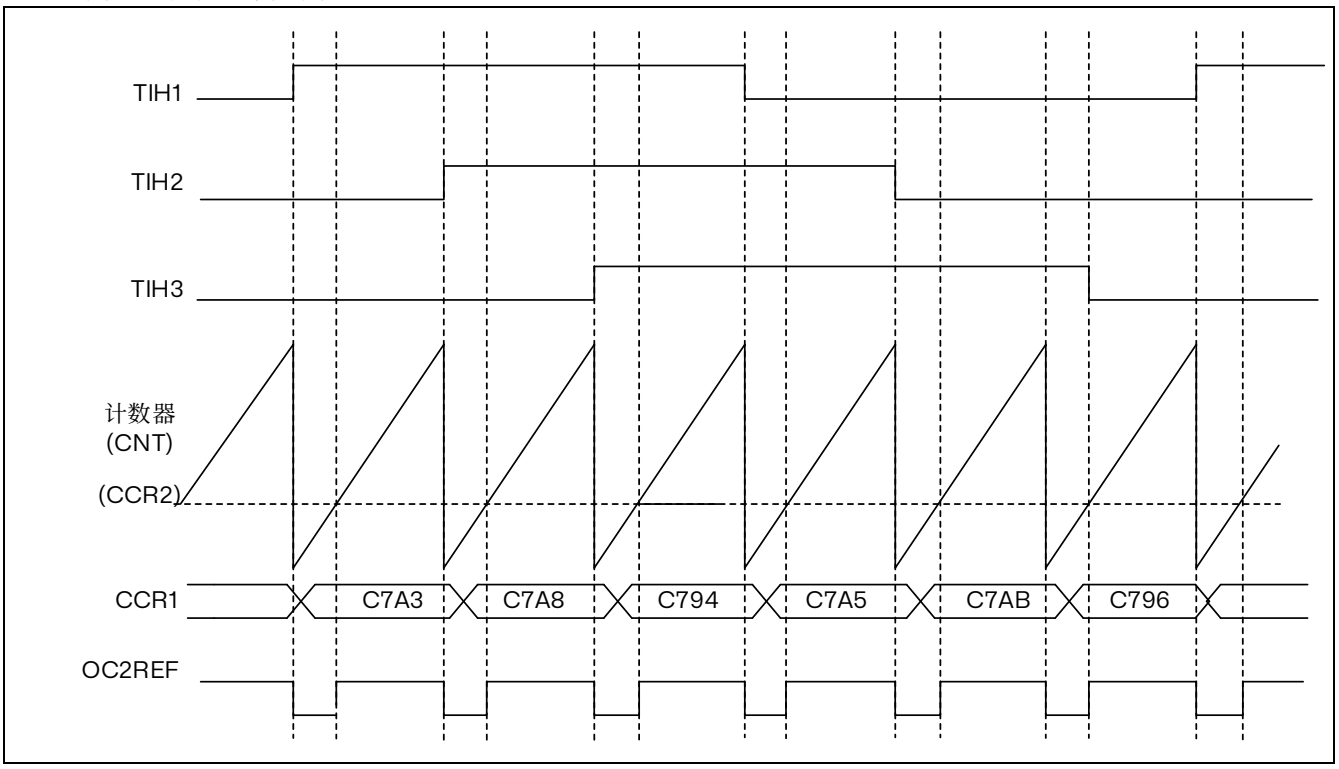



图 12.46 霍尔传感器接口的实例

### 12.3.20 TIM8 定时器和外部触发的同步

TIM8 定时器能够在多种模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

**从模式：复位模式**

在发生一个触发输入事件时，计数器和它的预分频器能够重新被初始化；同时，如果 TIM8_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器（TIM8_ARR，TIM8_CCRx）都被更新了。

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

- 配置通道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽（在本例中，不需要任何滤波器，因此保持 IC1F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕

获源，即 TIM8_CCMR1 寄存器中 CC1S=01。置 TIM8_CCER 寄存器中 CC1P=0 以确定极性（只检测上升沿）。

- 置 TIM8_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIM8_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIM8_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志（TIM8_SR 寄存器中的 TIF 位）被设置，根据 TIM8_DIER 寄存器中 TIE（中断使能）位和 TDE（DMA 使能）位的设置，产生一个中断请求或一个 DMA 请求。

下图显示当自动重装载寄存器 TIM8_ARR=0x36 时的动作。在 TI1 上升沿和计数器的实际复位之间的延时取决于 TI1 输入端的重同步电路。

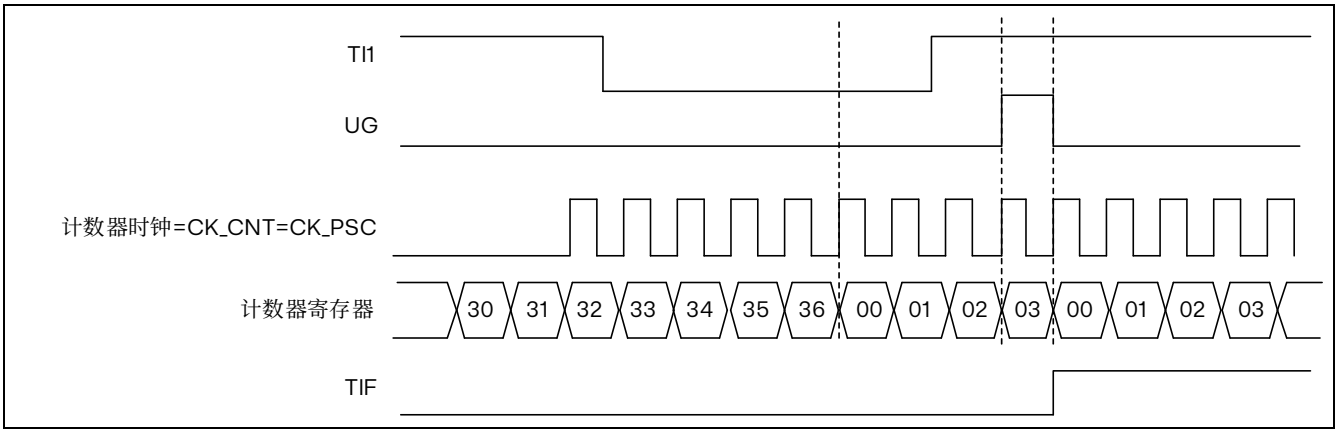



图 12.47 复位模式下的控制电路

**从模式：门控模式**

按照选中的输入端电平使能计数器。在如下的例子中，计数器只在 TI1 为低时向上计数：

- 配置通道 1 以检测 TI1 上的低电平。配置输入滤波器带宽（本例中，不需要滤波，所以保持 IC1F=0000）。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 TIM8_CCMR1 寄存器中 CC1S=01。置 TIM8_CCER 寄存器中 CC1P=1 以确定极性（只检测低电平）。
- 置 TIM8_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIM8_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIM8_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

只要 TI1 为低，计数器开始依据内部时钟计数，一旦 TI1 变高则停止计数。当计数器开始或停止时都设置 TIM8_SR 中的 TIF 标置。

TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的重同步电路。

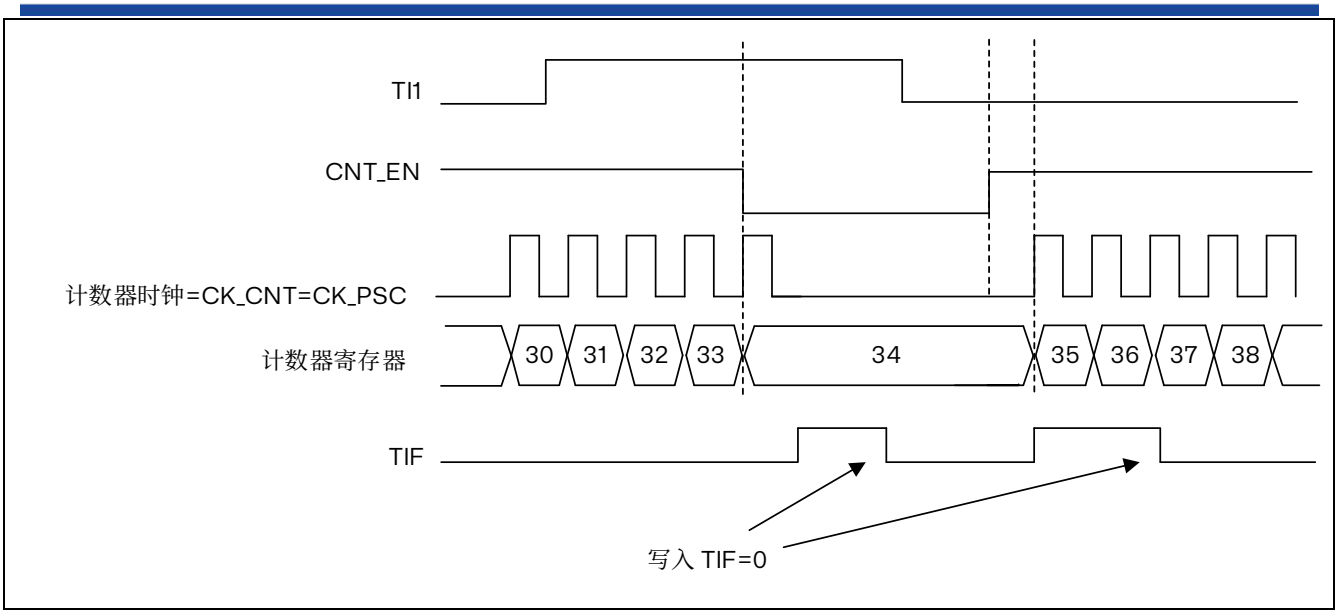



图 12.48 门控模式下的控制电路

**从模式：触发模式**

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- 配置通道 2 检测 TI2 的上升沿。配置输入滤波器带宽（本例中，不需要任何滤波器，保持 IC2F=0000）。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 TIM8_CCMR1 寄存器中 CC2S=01。置 TIM8_CCER 寄存器中 CC2P=1 以确定极性（只检测低电平）。
- 置 TIM8_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIM8_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，计数器开始在内部时钟驱动下计数，同时设置 TIF 标志。TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

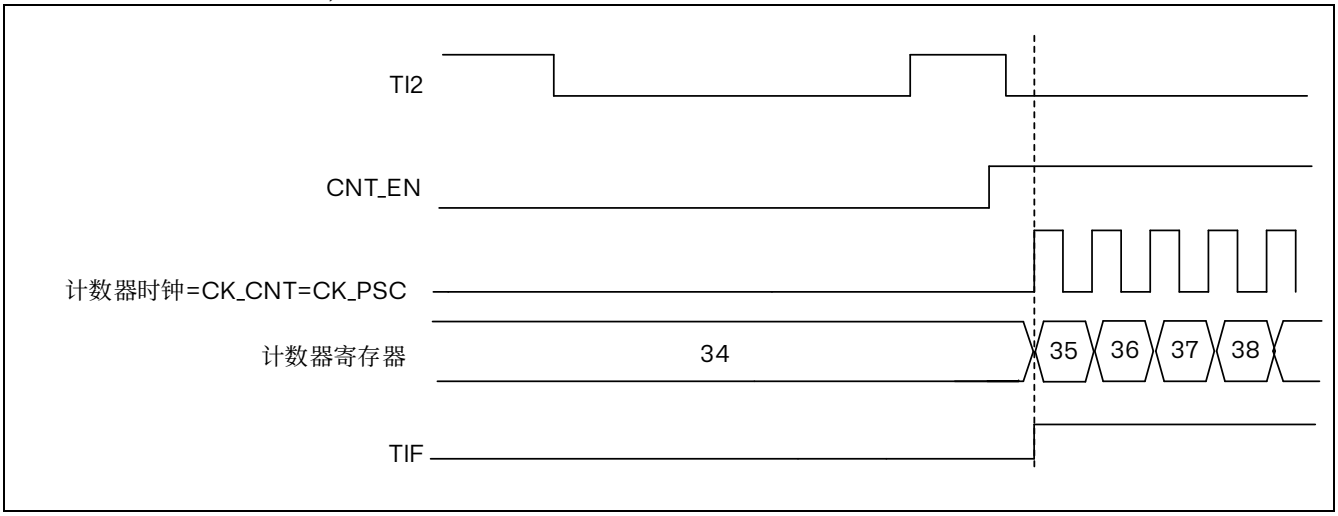



图 12.49 触发器模式下的控制电路

**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，在复位模式、门控模式或触发模式可以选择另一个输入作为触发输入。不建议使用 TIM8_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。在下面的例子中，一旦在 TI1 上出现一个上升沿，计数器即在 ETR 的每一个上升沿向上计数一次：

1. 通过 TIM8_SMCR 寄存器配置外部触发输入电路：
   - ETF=0000：没有滤波
   - ETPS=00：不用预分频器
   - ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式 2。
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
   - 置 TIM8_CCMR1 寄存器中 CC1S=01，选择输入捕获源
   - 置 TIM8_CCER 寄存器中 CC1P=0 以确定极性（只检测上升沿）
3. 置 TIM8_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 TIM8_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 标志被设置，计数器开始在 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。

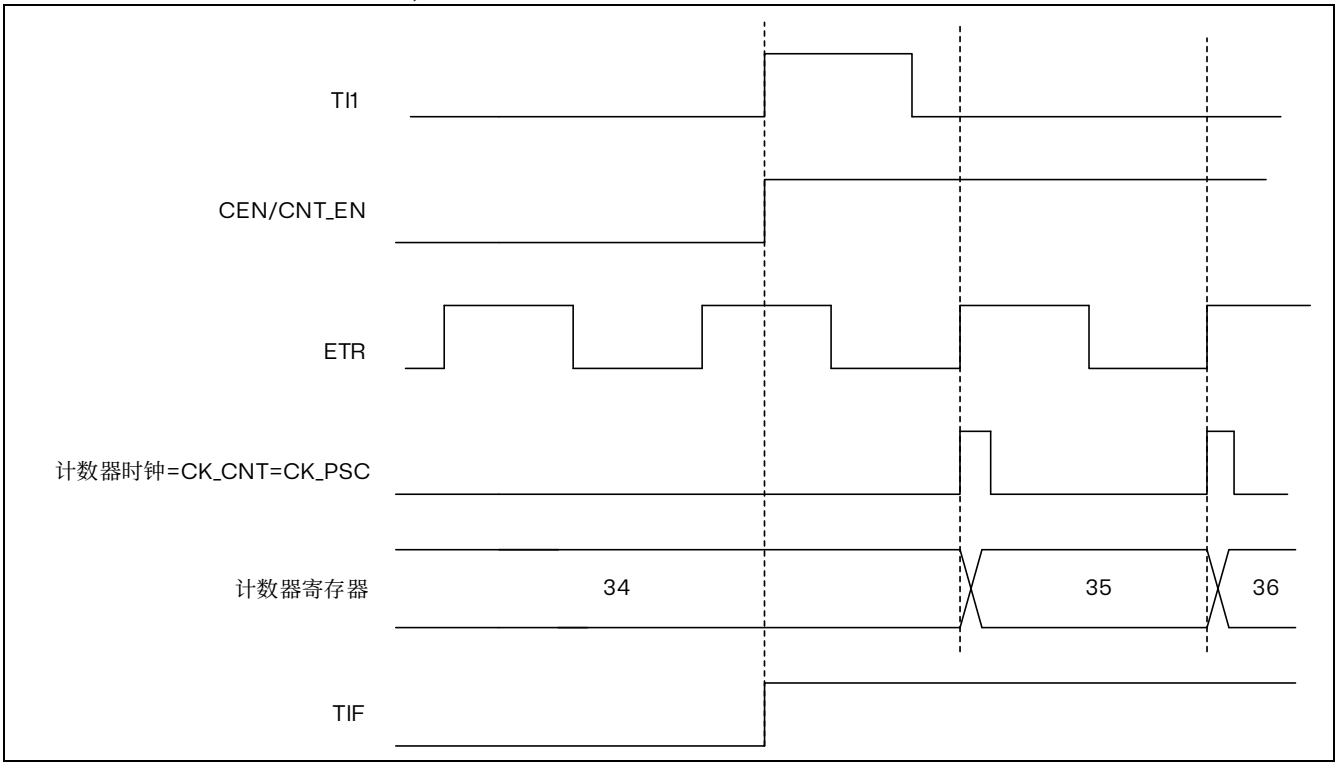



图 12.50 外部时钟模式 2＋触发模式下的控制电路

## 12.4 TIM8 寄存器

### 12.4.1 TIM8 控制寄存器 1（TIM8_CR1）

偏移地址：0x00

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 10 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | CKD | | ARPE | CMS[1:0] | | DIR | OPM | URS | UDIS | CEN |
| | | | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:10 保留，始终读为 0。

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、死区时间和由死区发生器与数字滤波器（ETR,TIx）所用 的采样时钟之间的分频比例。

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，不要使用这个配置

Bit 7 **ARPE**：自动重装载预装载允许位（Auto-reload preload enable）

0：TIM8_ARR 寄存器没有缓冲；　　1：TIM8_ARR 寄存器被装入缓冲器。

Bit 6:5 **CMS[1:0]**：选择中央对齐模式（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：中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道（TIM8_CCMRx 寄存器中 CCxS=00）的输出比较中断标志位，只在计数器向下计数时被设置。

10：中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道（TIM8_CCMRx 寄存器中 CCxS=00）的输出比较中断标志位，只在计数器向上计数时被设置。

11：中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道（TIM8_CCMRx 寄存器中 CCxS=00）的输出比较中断标志位，在计数器向上和向下计数时均被设置。

*注：在计数器开启时*（CEN=1），*不允许从边沿对齐模式转换到中央对齐模式。*

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）

软件通过该位选择 UEV 事件的源

0：如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求：

- 计数器溢出/下溢
- 设置 UG 位

– 从模式控制器产生的更新

1：如果使能更新中断或 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/下溢

– 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。（译注：更新影子寄存器）

1：禁止 UEV。不产生更新事件，影子寄存器（ARR、PSC、CCRx）保持它们的值。

如果设置了 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化。

Bit 0 **CEN**：使能计数器 （Counter enable）

0：禁止计数器；

1：使能计数器。

*注：在软件设置了*CEN *位后，外部时钟、门控模式和编码器模式才能工作。触发模式可以自动 地通过硬件设置*CEN *位。*

### 12.4.2 TIM8 控制寄存器 2（TIM8_CR2）

偏移地址：0x04

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 10 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | OIS6 |
| | | | | | | | | | | | | | | | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OIS5 | OIS4 | OIS3N | OIS3 | OIS2N | OIS2 | OIS1N | OIS1 | TI1S | Reserved | | | | CCUS | Reserved | CCPC |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | rw | | rw |

Bit 31:17 保留，始终读为 0。

Bit 16 **OIS6**：输出空闲状态 6（OC6 输出）。

Bit 15 **OIS5**：输出空闲状态 5（OC5 输出）。

Bit 14 **OIS4**：输出空闲状态 4（OC4 输出）。

Bit 13 **OIS3N**：输出空闲状态 3（OC3N 输出）。

Bit 12 **OIS3**：输出空闲状态 3（OC3 输出）。

Bit 11 **OIS2N**：输出空闲状态 2（OC2N 输出）。

Bit 10 **OIS2**：输出空闲状态 2（OC2 输出）。

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：已经设置了*LOCK *（*TIM8_BKR *寄存器）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，如果实现了 OC1N，则死区后 OC1=0；

1：当 MOE=0 时，如果实现了 OC1N，则死区后 OC1=1。

*注：已经设置了*LOCK *（*TIM8_BKR *寄存器）级别*1、2 *或*3 *后，该位不能被修改。*

Bit 7 **TI1S**：TI1 选择 （TI1 selection）

0：TIM8_CH1 引脚连到 TI1 输入；

1：TIM8_CH1、TIM8_CH2 和 TIM8_CH3 引脚经异或后连到 TI1 输入。

Bit 6:3 保留，始终读为 0。

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 COM 位更新它们；

1：如果捕获/比较控制位是预装载的（CCPC=1），可以通过设置 COM 位或 TRGI 上的一个上升沿更新它们。

*注：该位只对具有互补输出的通道起作用。*

Bit 1 保留，始终读为 0。

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，它们只在设置了 COM 位后被更新。

*注：该位只对具有互补输出的通道起作用。*

### 12.4.3 TIM8 从模式控制寄存器（TIM8_SMCR）

偏移地址：0x08

复位值：0x0000

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

Bit 31:16 保留，始终读为 0。

Bit 15 **ETP**：外部触发极性（External trigger polarity）

该位选择是用 ETR 还是 ETR 的反相来作为触发操作

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 的频率必须最多是 TIM8CLK 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。实际上，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 fDTS 采样

0001：采样频率 fSAMPLING=fCK_INT，N=2

0010：采样频率 fSAMPLING=fCK_INT，N=4

0011：采样频率 fSAMPLING=fCK_INT，N=8

0100：采样频率 fSAMPLING=fDTS/2，N=6

0101：采样频率 fSAMPLING=fDTS/2，N=8

0110：采样频率 fSAMPLING=fDTS/4，N=6

0111：采样频率 fSAMPLING=fDTS/4，N=8

1000：采样频率 fSAMPLING=fDTS/8，N=6

1001：采样频率 fSAMPLING=fDTS/8，N=8

1010：采样频率 fSAMPLING=fDTS/16，N=5

1011：采样频率 fSAMPLING=fDTS/16，N=6

1100：采样频率 fSAMPLING=fDTS/16，N=8

1101：采样频率 fSAMPLING=fDTS/32，N=5

1110：采样频率 fSAMPLING=fDTS/32，N=6

1111：采样频率 fSAMPLING=fDTS/32，N=8

Bit 7 保留，始终读为 0。

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。

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，参见表 12.2。

*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 保留，始终读为 0。

Bit 2:0 **SMS[2:0]**：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 TI1FP1 的电平，计数器在 TI2FP2 的边沿向上/下计数。

010：编码器模式 2 – 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿向上/下计数。

011：编码器模式 3 – 根据另一个信号的输入电平，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，并且产生一个更新寄存器的信号。

101：门控模式 – 当触发输入（TRGI）为高时，计数器的时钟开启。一旦触发输入变为

低，则计数器停止（但不复位）。计数器的启动和停止都是受控的。

110：触发模式 - 计数器在触发输入 TRGI 的上升沿启动（但不复位），只有计数器的启动是受控的。

111：外部时钟模式 1 - 选中的触发输入（TRGI）的上升沿驱动计数器。

注：如果 TI1F_EN 被选为触发输入（TS=100）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。

### 12.4.4 TIM8 DMA/中断使能寄存器（TIM8_DIER）

偏移地址：0x0C

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 10 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | CC6IE | Reserved |
| | | | | | | | | | | | | | | rw | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CC5IE | Reserved | | | | | | | BIE | TIE | COMIE | CC4IE | CC3IE | CC2IE | CC1IE | UIE |
| rw | | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:18 保留，始终读为 0。

Bit 17 **CC6IE**：允许捕获/比较 6 中断（Capture/Compare 6 interrupt enable）

0：禁止捕获/比较 6 中断；

1：允许捕获/比较 6 中断。

Bit 16 保留，始终读为 0。

Bit 15 **CC5IE**：允许捕获/比较 5 中断（Capture/Compare 5 interrupt enable）

0：禁止捕获/比较 5 中断；

1：允许捕获/比较 5 中断。

Bit 14:8 保留，始终读为 0。

Bit 7 **BIE**：允许刹车中断（Break interrupt enable）

0：禁止刹车中断；

1：允许刹车中断。

Bit 6 **TIE**：触发中断使能（Trigger interrupt enable）

0：禁止触发中断；

1：使能触发中断。

Bit 5 **COMIE**：允许 COM 中断（COM interrupt enable）

0：禁止 COM 中断；

1：允许 COM 中断。

Bit 4 **CC4IE**：允许捕获/比较 4 中断（Capture/Compare 4 interrupt enable）

0：禁止捕获/比较 4 中断；

1：允许捕获/比较 4 中断。

Bit 3 **CC3IE**：允许捕获/比较 3 中断（Capture/Compare 3 interrupt enable）

0：禁止捕获/比较 3 中断；

1：允许捕获/比较 3 中断。

Bit 2 **CC2IE**：允许捕获/比较 2 中断（Capture/Compare 2 interrupt enable）

0：禁止捕获/比较 2 中断；

1：允许捕获/比较 2 中断。

Bit 1 **CC1IE**：允许捕获/比较 1 中断（Capture/Compare 1 interrupt enable）

0：禁止捕获/比较 1 中断；
1：允许捕获/比较 1 中断。

Bit 0 **UIE**：允许更新中断（Update interrupt enable）
0：禁止更新中断；
1：允许更新中断。

### 12.4.5 TIM8 状态寄存器（TIM8_SR）

偏移地址：0x10
复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 10 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CC6IF | CC5IF | CC5OF | CC4OF | CC3OF | CC2OF | CC1OF | Reserved | BIF | TIF | COMIF | CC4IF | CC3IF | CC2IF | CC1IC | UIF |
| rc w0 | rc w0 | rc w0 | rc w0 | rc w0 | rc w0 | rc w0 | | rc w0 | rc w0 | rc w0 | rc w0 | rc w0 | rc w0 | rc w0 | rc w0 |

Bit 31:16 保留，始终读为 0。

Bit 15 **CC6IF**：捕获/比较 6 中断标记（Capture/Compare 6 overcapture flag）
参见 CC1IF 描述。

Bit 14 **CC5IF**：捕获/比较 5 中断标记（Capture/Compare 5 overcapture flag）
参见 CC1IF 描述。

Bit 13 **CC5OF**：捕获/比较 5 重复捕获标记（Capture/Compare 5 overcapture flag）
参见 CC1OF 描述。

Bit 12 **CC4OF**：捕获/比较 4 重复捕获标记（Capture/Compare 4 overcapture flag）
参见 CC1OF 描述。

Bit 11 **CC3OF**：捕获/比较 3 重复 捕获标记（Capture/Compare 3 overcapture flag）
参见 CC1OF 描述。

Bit 10 **CC2OF**：捕获/比较 2 重复捕获标记（Capture/Compare 2 overcapture flag）
参见 CC1OF 描述。

Bit 9 **CC1OF**：捕获/比较 1 重复捕获标记（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。写 0 可清除该位。
0：无重复捕获产生；
1：计数器的值被捕获到 TIM8_CCR1 寄存器时，CC1IF 的状态已经为‘1’。

Bit 8 保留，始终读为 0。

Bit 7 **BIF**：刹车中断标记（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**：触发器中断标记（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 中断标记（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 中断标记（Capture/Compare 4 interrupt flag）

参考 CC1IF 描述。

Bit 3 **CC3IF**：CC3IF：捕获/比较 3 中断标记（Capture/Compare 3 interrupt flag）

参考 CC1IF 描述。

Bit 2 **CC2IF**：捕获/比较 2 中断标记（Capture/Compare 2 interrupt flag）

参考 CC1IF 描述。

Bit 1 **CC1IF**：CC1IF：捕获/比较 1 中断标记（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，但在中心对称模式下除外（参考 TIM8_CR1 寄存器的 CMS 位）。它由软件清‘0’。

0：无匹配发生；

1：TIM8_CNT 的值与 TIM8_CCR1 的值匹配。

当 TIM8_CCR1 的内容大于 TIM8_APR 的内容时，在向上或向上/下计数模式时计数器溢出，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 TIM8_CCR1 清‘0’。

0：无输入捕获产生；

1：计数器值已被捕获（拷贝）至 TIM8_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 TIM8_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时（重复计数器=0 时产生更新事件）。
- 若 TIM8_CR1 寄存器的 URS=0、UDIS=0，当设置 TIM8_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。
- 若 TIM8_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。

### 12.4.6 TIM8 事件产生寄存器（TIM8_EGR）

偏移地址：0x14

复位：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Reserved | | | | | | CC6G | CC5G | BG | TG | COMG | CC4G | CC3G | CC2G | CC1G | UG |
| | | | | | | rw | rw | w | w | w | w | w | w | w | w |

Bit 31:10 保留，始终读为 0。

Bit 9 **CC6G**：产生捕获/比较 6 事件（Capture/Compare 6 generation）

参考 CC1G 描述。

Bit 8 **CC5G**：产生捕获/比较 5 事件（Capture/Compare 5 generation）

参考 CC1G 描述。

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 **TG**：产生触发事件（Trigger generation）

该位由软件置‘1’，用于产生一个触发事件，由硬件自动清‘0’。

0：无动作；

1：TIM8_SR 寄存器的 TIF=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/比较事件，产生控制更新（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**：产生捕获/比较 4 事件（Capture/Compare 4 generation）

参考 CC1G 描述。

Bit 3 **CC3G**：产生捕获/比较 3 事件（Capture/Compare 3 generation）

参考 CC1G 描述。

Bit 2 **CC2G**：产生捕获/比较 2 事件（Capture/Compare 2 generation）

参考 CC1G 描述。

Bit 1 **CC1G**：产生捕获/比较 1 事件（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 CC1 上产生一个捕获/比较事件：

**若通道 CC1 配置为输出：**

设置 CC1IF=1，若开启对应的中断和 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 TIM8_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 TIM8_ARR 的值。

### 12.4.7 TIM8 捕获/比较模式寄存器 1（TIM8_CCMR1）

偏移地址：0x18

复位值：0x0000

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。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0C2CE | 0C2M[2:0] | | | 0C2PE | 0C2FE | CC2S[1:0] | | 0CICE | 0CIM[2:0] | | | 0CIPE | 0CIFE | CCIS[1:0] | |
| IC2F[3:0] | | | | IC2PSC[1:0] | | | | IC1F[3:0] | | | | IC1PSC[1:0] | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

**输出比较模式**

Bit 31:16 保留

Bit 15 **OC2CE：**输出比较 2 清 0 使能（Output Compare 2 clear enable）

Bit 14:12 **OC2M[2:0]：**输出比较 2 模式（Output Compare 2 mode）

Bit 11 **OC2PE：**输出比较 2 预装载使能（Output Compare 2 preload enable）

Bit 10 **OC2FE：**输出比较 2 快速使能（Output Compare 2 fast enable）

Bit 9:8 **CC2S[1:0]：**捕获/比较 2 选择。（Capture/Compare 2 selection）

该位定义通道的方向（输入/输出），及输入脚的选择：

00：CC2 通道被配置为输出；

01：CC2 通道被配置为输入，IC2 映射在 TI2 上；

10：CC2 通道被配置为输入，IC2 映射在 TI1 上；

11：CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM8_SMCR 寄存器的 TS 位选择）。

*注：CC2S 仅在通道关闭时（TIM8_CCER 寄存器的 CC2E=0）才是可写的。*

Bit 7 **OC1CE：**输出比较 1 清‘0’使能（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[2:0]：**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、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。

000：冻结。输出比较寄存器 TIM8_CCR1 与计数器 TIM8_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 TIM8_CNT 的值与捕获/比较寄存器 1（TIM8_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 TIM8_CNT 的值与捕获/比较寄存器 1（TIM8_CCR1）相同时，强制 OC1REF 为低。

011：翻转。当 TIM8_CCR1=TIM8_CNT 时，翻转 OC1REF 的电平。

100：强制为无效电平。强制 OC1REF 为低。

101：强制为有效电平。强制 OC1REF 为高。

110：PWM 模式 1—在向上计数时，一旦 TIM8_CNT<TIM8_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 TIM8_CNT>TIM8_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，一旦 TIM8_CNT<TIM8_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIM8_CNT>TIM8_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（TIM8_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 TIM8_CCR1 寄存器的预装载功能，可随时写入 TIM8_CCR1 寄存器，并且新写入的数值立即起作用。

1：开启 TIM8_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIM8_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（TIM8_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（TIM8_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：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。OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。

Bit 1:0 **CC1S[1:0]：**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：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 TIM8_SMCR 寄存器的 TS 位选择）。

注：CC1S 仅在通道关闭时（TIM8_CCER 寄存器的 CC1E=0）才是可写的。

**输入捕获模式：**

Bit 31:16 保留

Bit 15:12 **IC2F[3:0]：**输入捕获 2 滤波器 （Input capture 2 filter）

Bit 11:10 **IC2PSC[1:0]：**输入/捕获 2 预分频器（Input capture 2 prescaler）

Bit 9:8 **CC2S[1:0]：**捕获/比较 2 选择（Capture/Compare 2 selection）

这 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 TIM8_SMCR 寄存器的 TS 位选择）。

*注：CC2S 仅在通道关闭时（TIM8_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 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6

0001：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=2 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8

0010：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=4 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5

0011：采样频率 $f_{SAMPLING}=f_{CK\_INT}$，N=8 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6

0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8

0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5

0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6

0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8

Bit 3:2 **IC1PSC[1:0]：**输入/捕获 1 预分频器（Input capture 1 prescaler）这 2 位定义了 CC1 输入(IC1)的预分频系数。一旦 CC1E=0(TIM8_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[1:0]：**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：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 TIM8_SMCR 寄存器的 TS 位选择）。

*注：CC1S 仅在通道关闭时（TIM8_CCER 寄存器的CC1E=0）才是可写的。*

### 12.4.8 TIM8 捕获/比较模式寄存器 2（TIM8_CCMR2）

偏移地址：0x1C

复位值：0x0000

参看以上 CCMR1 寄存器的描述

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0C4CE | 0C4M[2:0] | | | 0C4PE | 0C3FE | CC4S[1:0] | | 0C3CE | 0C3M[2:0] | | | 0C3PE | 0C3FE | CC3S[1:0] | |
| IC4F[3:0] | | | | IC4PSC[1:0] | | | | IC3F[3:0] | | | | IC3PSC[1:0] | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

**输出比较模式：**

Bit 31:16 保留

Bit 15 **OC4CE**：输出比较 4 清 0 使能（Output compare 4 clear enable）
Bit 14:12 **OC4M[2:0]**：输出比较 4 模式（Output compare 4 mode）
Bit 11 **OC4PE**：输出比较 4 预装载使能（Output compare 4 preload enable）
Bit 10 **OC4FE**：输出比较 4 快速使能（Output compare 4 fast enable）
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），及输入脚的选择：
00：CC4 通道被配置为输出；
01：CC4 通道被配置为输入，IC4 映射在 TI4 上；
10：CC4 通道被配置为输入，IC4 映射在 TI3 上；
11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM8_SMCR 寄存器的 TS 位选择）。
*注：CC4S 仅在通道关闭时（TIM8_CCER 寄存器的CC4E=0）才是可写的。*
Bit 7 **OC3CE**：输出比较 3 清 0 使能（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 预装载使能（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）
这 2 位定义通道的方向（输入/输出），及输入脚的选择：
00：CC3 通道被配置为输出；
01：CC3 通道被配置为输入，IC3 映射在 TI3 上；
10：CC3 通道被配置为输入，IC3 映射在 TI4 上；
11：CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM8_SMCR 寄存器的 TS 位选择）。
*注：CC3S 仅在通道关闭时（TIM8_CCER 寄存器的CC3E=0）才是可写的。*

**输入捕获模式：**

Bit 31:16 保留
Bit 15:12 **IC4F[3:0]**：输入捕获 4 滤波器（Input capture 4 filter）
Bit 11:10 **IC4PSC[1:0]**：输入/捕获 4 预分频器（Input capture 4 prescaler）
Bit 9:8 **CC4S[1:0]**：捕获/比较 4 选择（Capture/Compare 4 selection）
这 2 位定义通道的方向（输入/输出），及输入脚的选择：
00：CC4 通道被配置为输出；
01：CC4 通道被配置为输入，IC4 映射在 TI4 上；
10：CC4 通道被配置为输入，IC4 映射在 TI3 上；
11：CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM8_SMCR 寄存器的 TS 位选择）。
*注：CC4S 仅在通道关闭时（TIM8_CCER 寄存器的CC4E=0）才是可写的。*
Bit 7:4 **IC3F[3:0]**：输入捕获 3 滤波器（Input capture 3 filter）
Bit 3:2 **IC3PSC[1:0]**：输入/捕获 3 预分频器（Input capture 3 prescaler）
Bit 1:0 **CC3S[1:0]**：捕获/比较 3 选择（Capture/compare 3 selection）
这 2 位定义通道的方向（输入/输出），及输入脚的选择：
00：CC3 通道被配置为输出；

01：CC3 通道被配置为输入，IC3 映射在 TI3 上；
10：CC3 通道被配置为输入，IC3 映射在 TI4 上；
11：CC3 通道被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM8_SMCR 寄存器的 TS 位选择）。
*注：CC3S 仅在通道关闭时*（TIM8_CCER *寄存器的*CC3E=0）*才是可写的。*

### 12.4.9 TIM8 捕获/比较使能寄存器（TIM8_CCER）

偏移地址：0x20
复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | CC6P | CC6E |
| | | | | | | | | | | | | | | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CC5P | CC5E | CC4P | CC4E | CC3NP | CC3NE | CC3P | CC3E | CC2NP | CC2NE | CC2P | CC2E | CC1NP | CC1NE | CC1P | CC1E |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:18 保留，始终读为 0。

Bit 17 **CC6P**： 输入/捕获 6 输出极性（Capture/Compare 6 output polarity）
参考 CC1P 的描述。

Bit 16 **CC6E**： 输入/捕获 6 输出使能（Capture/Compare 6 output enable）
参考 CC1E 的描述。

Bit 15 **CC5P**： 输入/捕获 5 输出极性（Capture/Compare 5 output polarity）
参考 CC1P 的描述。

Bit 14 **CC5E**： 输入/捕获 5 输出使能（Capture/Compare 5 output enable）
参考 CC1E 的描述

Bit 13 **CC4P**： 输入/捕获 4 输出极性（Capture/Compare 4 output polarity）
参考 CC1P 的描述。

Bit 12 **CC4E**： 输入/捕获 4 输出使能（Capture/Compare 4 output enable）
参考 CC1E 的描述。

Bit 11 **CC3NP**：输入/捕获 3 互补输出极性
（Capture/Compare 3 complementary output polarity）
参考 CC1NP 的描述。

Bit 10 **CC3NE**：输入/捕获 3 互补输出使能
（Capture/Compare 3 complementary output enable）
参考 CC1NE 的描述。

Bit 9 **CC3P**： 输入/捕获 3 输出极性（Capture/Compare 3 output polarity）
参考 CC1P 的描述。

Bit 8 **CC3E**： 输入/捕获 3 输出使能（Capture/Compare 3 output enable）
参考 CC1E 的描述。

Bit 7 **CC2NP**： 输入/捕获 2 互补输出极性
（Capture/Compare 2 complementary output polarity）
参考 CC1NP 的描述。

Bit 6 **CC2NE**： 输入/捕获 2 互补输出使能
（Capture/Compare 2 complementary output enable）
参考 CC1NE 的描述。

Bit 5 **CC2P**： 输入/捕获 2 输出极性（Capture/Compare 2 output polarity）
参考 CC1P 的描述。

Bit 4 **CC2E**： 输入/捕获 2 输出使能（Capture/Compare 2 output enable）
参考 CC1E 的描述。

Bit 3 **CC1NP**：输入/捕获 1 互补输出极性
（Capture/Compare 1 complementary output polarity）
0：OC1N 高电平有效；
1：OC1N 低电平有效。
*注：一旦 LOCK 级别（TIM8_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**：输入/捕获 1 互补输出使能
（Capture/Compare 1 complementary output enable）
0：关闭－OC1N 禁止输出，因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。
1：开启－OC1N 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。

Bit 1 **CC1P**：输入/捕获 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：
该位选择是 IC1 还是 IC1 的反相信号作为触发或捕获信号。
0：不反相：捕获发生在 IC1 的上升沿；当用作外部触发器时，IC1 不反相。
1：反相：捕获发生在 IC1 的下降沿；当用作外部触发器时，IC1 反相。
*注：一旦 LOCK 级别（TIM8_BDTR 寄存器中的 LOCK 位）设为 3 或 2，则该位不能被修改。*

Bit 0 **CC1E**：输入/捕获 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，因此 OC1 的输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。
1：开启－OC1 信号输出到对应的输出引脚，其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。
CC1 通道配置为输入：
该位决定了计数器的值是否能捕获入 TIM8_CCR1 寄存器。
0：捕获禁止；
0：捕获使能。

表 12.2 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 位 | OSSI 位 | OSSR 位 | CCxE 位 | CCxNE 位 | OCx 输出状态 | OCxN 输出状态 |
| 1 | X | 0 | 0 | 0 | 输出禁止（与定时器断开）<br>OCx=0，OCx_EN=0 | 输出禁止（与定时器断开）<br>OCxN=0，OCxN_EN=0 |
| | | 0 | 0 | 1 | 输出禁止（与定时器断开）<br>OCx=0，OCx_EN=0 | OCxREF + 极性，<br>OCxN= OCxREF xor CCxNP，<br>OCxN_EN=1 |
| | | 0 | 1 | 0 | OCxREF + 极性，<br>OCx= OCxREF xor CCxP，<br>OCx_EN=1 | 输出禁止（与定时器断开）<br>OCxN=0，OCxN_EN=0 |
| | | 0 | 1 | 1 | OCxREF + 极性 + 死区，<br>OCx_EN=1 | OCxREF 反相 + 极性 + 死区，<br>OCxN_EN=1 |
| | | 1 | 0 | 0 | 输出禁止（与定时器断开）<br>OCx=CCxP，OCx_EN=0 | 输出禁止（与定时器断开）<br>OCxN=CCxNP，OCxN_EN=0 |
| | | 1 | 0 | 1 | 关闭状态<br>（输出使能且为无效电平）<br>OCx=CCxP，OCx_EN=1 | OCxREF + 极性，<br>OCxN= OCxREF xor CCxNP，<br>OCxN_EN=1 |
| | | 1 | 1 | 0 | OCxREF + 极性，<br>OCx= OCxREF xor CCxP，<br>OCx_EN=1 | 关闭状态<br>（输出使能且为无效电平）<br>OCxN=CCxNP，OCxN_EN=0 |
| | | 1 | 1 | 1 | OCxREF + 极性 + 死区，<br>OCx_EN=1 | OCxREF 反相 + 极性 + 死区，<br>OCxN_EN=1 |
| 0 | 0 | X | 0 | 0 | 输出禁止（与定时器断开） | |
| | 0 | | 0 | 1 | 异步地：OCx=CCxP，OCx_EN=0，OCxN=CCxNP，OCxN_EN=0; | |
| | 0 | | 1 | 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 | 关闭状态（输出使能且为无效电平） | |
| | 1 | | 0 | 1 | 异步地：OCx=CCxP，OCx_EN=1，OCxN=CCxNP，OCxN_EN=1; | |
| | 1 | | 1 | 0 | | |
| | 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。 | |

1、如果一个通道的 2 个输出都没有使用（CCxE = CCxNE = 0），那么 OISx，OISxN，CCxP 和 CCxNP 都必须清零。

*注：引脚连接到互补的*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 *以及*AFIO *寄存器。*

### 12.4.10 TIM8 计数器（TIM8_CNT）

偏移地址：0x24

复位值：0x0000

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

Bit 31:16 保留

Bit 15:0 **CNT[15:0]：**计数器的值（Counter value）

### 12.4.11 TIM8 预分频器（TIM8_PSC）

偏移地址：0x28

复位值：0x0000

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

Bit 31:16 保留

Bit 15:0 **PSC[15:0]：**预分频器的值（Prescaler value）

计数器的时钟频率（CK_CNT）等于 $f_{CK\_PSC}$/（PSC[15:0]+1）。

PSC 包含了每次当更新事件产生时，装入当前预分频器寄存器的值；更新事件包括计数器被 TIM_EGR 的 UG 位清‘0’或被工作在复位模式的从控制器清‘0’。

### 12.4.12 TIM8 自动重装载寄存器（TIM8_ARR）

偏移地址：0x2C

复位：0x0000

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

Bit 31:16 保留

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。

当自动重装载的值为空时，计数器不工作。

### 12.4.13 TIM8 重复计数寄存器（TIM8_RCR）

偏移地址：0x30

复位值：0x0000

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

Bit 31:8 保留，始终读为 0。

Bit 7:0 **REP[7:0]**：重复计数器的值 （Repetition counter value）

开启了预装载功能后，这些位允许用户设置比较寄存器的更新速率（即周期性地从预装载寄存器 传输到当前寄存器）；如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 TIM8_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。

这意味着在 PWM 模式中，（REP+1）对应着：

– 在边沿对齐模式下，PWM 周期的数目；

– 在中心对称模式下，PWM 半周期的数目；

### 12.4.14 TIM8 捕获/比较寄存器 1（TIM8_CCR1）

偏移地址：0x34

复位值：0x0000

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

Bit 31:16 保留

Bit 15:0 **CCR1[15:0]**：捕获/比较通道 1 的值（Capture/Compare 1 value）

**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 TIM8_CCMR1 寄存器（OC1PE 位）中未选择预装载功能，写入的数值会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 TIM8_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。

### 12.4.15 TIM8 捕获/比较寄存器 2（TIM8_CCR2）

偏移地址：0x38

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CCR2[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:16 保留

Bit 15:0 **CCR2[15:0]：**捕获/比较通道 2 的值（Capture/Compare 2 value）

**若 CC2 通道配置为输出：**

CCR2 包含了装入当前捕获/比较 2 寄存器的值（预装载值）。如果在 TIM8_CCMR2 寄存器（OC2PE 位）中未选择预装载特性，写入的数值会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 2 寄存器中。当前捕获/比较寄存器参与同计数器 TIM8_CNT 的比较，并在 OC2 端口上产生输出信号。

**若 CC2 通道配置为输入：**

CCR2 包含了由上一次输入捕获 2 事件（IC2）传输的计数器值。

### 12.4.16 TIM8 捕获/比较寄存器 3（TIM8_CCR3）

偏移地址：0x3C

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CCR3[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:16 保留

Bit 15:0 **CCR3[15:0]：**捕获/比较通道 3 的值（Capture/Compare 3 value）

**若 CC3 通道配置为输出：**

CCR3 包含了装入当前捕获/比较 3 寄存器的值（预装载值）。如果在 TIM8_CCMR3 寄存器（OC3PE 位）中未选择预装载特性，写入的数值会立即传输至当前寄 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 3 寄存器中。当前捕获/比较寄存器参与同计数器 TIM8_CNT 的比较，并在 OC3 端口上产生输出信号。

**若 CC3 通道配置为输入：**

CCR3 包含了由上一次输入捕获 3 事件（IC3）传输的计数器值。

### 12.4.17 TIM8 捕获/比较寄存器 4（TIM8_CCR4）

偏移地址：0x40

复位值：0x0000

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

Bit 31:16 保留

Bit 15:0 **CCR4[15:0]：**捕获/比较通道 4 的值（Capture/Compare 4 value）

**若 CC4 通道配置为输出：**

CCR4 包含了装入当前捕获/比较 4 寄存器的值（预装载值）。如果在 TIM8_CCMR4 寄存器（OC4PE 位）中未选择预装载特性，写入的数值会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 4 寄存器中。当前捕获/比较寄存器参与同计数器 TIM8_CNT 的比较，并在 OC4 端口上产生输出信号。

**若 CC4 通道配置为输入：**

CCR4 包含了由上一次输入捕获 4 事件（IC4）传输的计数器值。

### 12.4.18 TIM8 刹车和死区寄存器（TIM8_BDTR）

偏移地址：0x44

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | BKF[3:0] | | | | Hard fault_OE | DOE | CMP5_ OEP | CMP5_ OE | CMP4_ OEP | CMP4_ OE |
| | | | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| MOE | AOE | BKP | BKE | OSSR | OSSI | LOCK[1:0] | | DTG[7:0] | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

*注释：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* TIM8_BDTR *寄存器时对它们进行配置。*

Bit 31:26 保留

Bit 25:22 **BKF[3:0]：**这几位定义了 BRK 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：No filter, BRK acts asynchronously

0001：fSAMPLING=fCK_INT, N=2

0010：fSAMPLING=fCK_INT, N=4

0011：fSAMPLING=fCK_INT, N=8

0100：fSAMPLING=fDTS/2, N=6

0101：fSAMPLING=fDTS/2, N=8

0110：fSAMPLING=fDTS/4, N=6

0111：fSAMPLING=fDTS/4, N=8

1000：fSAMPLING=fDTS/8, N=6

1001：fSAMPLING=fDTS/8, N=8

1010：fSAMPLING=fDTS/16, N=5

1011：fSAMPLING=fDTS/16, N=6
1100：fSAMPLING=fDTS/16, N=8
1101：fSAMPLING=fDTS/32, N=5
1110：fSAMPLING=fDTS/32, N=6
1111：fSAMPLING=fDTS/32, N=8

Bit 21 **Hardfault_OE**：刹车功能 Hardfault_out 使能（Break enable）
0：禁止刹车输入
1：开启刹车输入
注：当设置了 LOCK 级别 1 时（TIM8_BDTR 寄存器中的 LOCK 位），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 20 **DOE**：直接输出
MOE 置 0 后,有效
1: 立即输出空闲状态,不等死区时间
0: 刹车输入后,等待一个死区时间后输出空闲状态

Bit 19 **CMP5_OEP**：刹车输入 CMP5_out 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(TIM8_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 **18 CMP5_OE**：刹车功能 CMP5_out 使能（Break enable）
0：禁止刹车输入
1：开启刹车输入
*注：当设置了 LOCK 级别 1 时(TIM8_BDTR 寄存器中的 LOCK 位)，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 17 **CMP4_OEP**：刹车输入 CMP4_out 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(TIM8_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 16 **CMP4_OE**：刹车功能 CMP4_out 使能（Break enable）
0：禁止刹车输入
1：开启刹车输入
*注：当设置了 LOCK 级别 1 时（IMx_BDTR 寄存器中的 LOCK 位），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 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（TIM8_CCER 寄存器的 CCxE、CCxNE 位），则开启 OC 和 OCN 输出。

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（TIM8_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（TIM8_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：**刹车功能使能（Break enable）

0：禁止刹车输入（BRK 及 CCS 时钟失效事件）；

1：开启刹车输入（BRK 及 CCS 时钟失效事件）。

注：当设置了 LOCK 级别 1 时（TIM8_BDTR 寄存器中的 LOCK 位），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。

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，首先开启 OC/OCN 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（TIM8_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 for Idle mode）

该位用于当 MOE=0 且通道设为输出时。

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 首先输出其空闲电平，然后 OC/OCN 使能输出信号=1。

*注：一旦 LOCK 级别（TIM8_BDTR 寄存器中的 LOCK 位）设为 2，则该位不能被修改。*

Bit 9:8 **LOOK[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 TIM8_BDTR 寄存器的 DTG、BKE、BKP、AOE 位和 TIM8_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 TIM8_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 TIM8_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 TIM8_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 × $T_{DTS}$；
例：若 TDTS = 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（TIM8_BDTR 寄存器中的 LOCK 位）设为 1、2 或 3，则不能修改这些位。

### 12.4.19 TIM8 捕获/比较寄存器 5（TIM8_CCR5）

偏移地址：0x50
复位值：0x0000

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CCR5[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **CCR5[15:0]：**捕获/比较通道 5 的值（Capture/Compare 5 value）
若 CC5 通道配置为输出：
CCR5 包含了装入当前捕获/比较 5 寄存器的值（预装载值）。
如果在 TIM8_CCMR4 寄存器（OC4PE 位）中未选择预装载特性，写入的数值会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 5 寄存器中。
当前捕获/比较寄存器参与同计数器 TIM8_CNT 的比较，并在 OC5 端口上产生输出信号。
若 CC5 信道配置为输入：
CCR5 包含了由上一次输入捕获 5 事件(IC5)传输的计数器值。

### 12.4.20 TIM8 捕获/比较模式寄存器 3（TIM8_CCMR3）

偏移地址：0x54
复位值：0x0000

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | OC6CE | OC6M[2:0] | | | OC6PE | OC6FE | OC5CE | OC5M[2:0] | | | OC5PE | OC5FE | CC5S | |
| | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | |

Bit 31:14 保留
Bit 13 **OC6CE：**输出比较 6 清 0 使能（Output compare 6 clear enable）
Bit 12:10 **OC6M[2:0]：**输出比较 6 模式（Output compare 6 mode）
Bit 9 **OC6PE：**输出比较 6 预装载使能（Output compare 6 preload enable）
Bit 8 **OC6FE：**输出比较 6 快速使能（Output compare 6 fast enable）
Bit 7 **OC5CE：**输出比较 5 清 0 使能（Output compare 5 clear enable）

Bit 6:4 **OC5M[2:0]：**输出比较 5 模式（Output compare 5 mode）

Bit 3 **OC5PE：**输出比较 5 预装载使能（Output compare 5 preload enable）

Bit 2 **OC5FE：**输出比较 5 快速使能（Output compare 5 fast enable）

Bit 1:0 **CC5S：**捕获/比较 5 选择（Capture/Compare 5 selection)

这 2 位定义通道的方向（输入/输出），及输入脚的选择：

00：CC5 通道被配置为输出；

01：CC5 通道被配置为输入，IC5 映射在 TI5 上；

10：CC5 通道被配置为输入，IC5 映射在 TI4 上；

11：CC5 通道被配置为输入，IC5 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：*CC5S *仅在通道关闭时*（TIMx_CCER *寄存器的*CC5E=0）*才是可写的。*

### 12.4.21 TIM8 捕获/比较寄存器 6（TIM8_CCR6）

偏移地址：0x58

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CCR6[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:16 保留

Bit 15:0 **CCR6[15:0]：**捕获/比较通道 6 的值（Capture/Compare 6 value）

若 CC6 通道配置为输出：

CCR6 包含了装入当前捕获/比较 6 寄存器的值（预装载值）。

如果在 TIM8_CCMR3 寄存器(OC6PE 位)中未选择预装载特性，写入的数值会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 6 寄存器中。

当前捕获/比较寄存器参与同计数器 TIM8_CNT 的比较，并在 OC6 端口上产生输出信号。

### 12.4.22 TIM8 捕获/比较模式寄存器（TIM8_CCR1N）

偏移地址：0x60

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CCR1N_EN | Reserved | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CCR1N[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31 **CCR1N_EN：**若 CC1 通道配置为输出：

CCR1N_EN=1：CNT 向下数比较的寄存器为 CCR1N

CCR1N_EN=0：CNT 向下数比较的寄存器为 CCR1

Bit 31:16 保留

Bit 15:0 **CCR1N[15:0]：**捕获/比较通道 1N 的值（Capture/Compare 1N value）

若 CC1 通道配置为输出：CCR1N 包含了装入当前捕获/比较 1 寄存器的值（预装载值）。

如果在 TIM8_CCMR1 寄存器（OC1PE 位）中未选择预装载功能，写入的数值会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。CCR1N_EN 必须为 1，寄存器才有效果。

### 12.4.23 TIM8 捕获/比较模式寄存器（TIM8_CCR2N）

偏移地址：0x64

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CCR2N_EN | Reserved | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CCR2N[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31 **CCR2N_EN：**若 CC2 通道配置为输出：

CCR2N_EN=1：CNT 向下数比较的寄存器为 CCR2N

CCR2N_EN=0：CNT 向下数比较的寄存器为 CCR2

Bit 31:16 保留

Bit 15:0 **CCR2N[15:0]：**捕获/比较通道 2N 的值（Capture/Compare 2N value）

若 CC2 通道配置为输出：CCR2N 包含了装入当前捕获/比较 2 寄存器的值（预装载值）。如果在 TIM8_CCMR2 寄存器（OC2PE 位）中未选择预装载功能，写入的数值会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 2 寄存器中。CCR2N_EN 必须为 1，寄存器才有效果。

### 12.4.24 TIM8 捕获/比较模式寄存器（TIM8_CCR3N）

偏移地址：0x68

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CCR3N_EN | Reserved | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CCR3N[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31 **CCR3N_EN：**若 CC3 通道配置为输出：

CCR3N_EN=1：CNT 向下数比较的寄存器为 CCR3N

CCR3N_EN=0：CNT 向下数比较的寄存器为 CCR3

Bit 31:16 保留

Bit 15:0 **CCR3N[15:0]：**捕获/比较通道 3N 的值（Capture/Compare 3N value）

若 CC3 通道配置为输出：CCR3N 包含了装入当前捕获/比较 3 寄存器的值（预装载值）。如果在 TIM8_CCMR3 寄存器（OC3PE 位）中未选择预装载功能，写入的数值会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 3 寄存器中。CCR3N_EN 必须为 1，寄存器才有效果。

# 13 通用定时器（TIMx）

## 13.1 TIMx 简介

通用定时器是一个通过可编程预分频器驱动的 16 位自动装载计数器构成。它适用于多种场合，包括测量输入信号的脉冲长度（输入捕获）或者产生输出波形（输出比较和 PWM）。使用定时器预分频器和 RCC 时钟控制器预分频器，脉冲长度和波形周期可以在几个微秒到几个毫秒间调整。每个定时器都是完全独立的，没有互相共享任何资源。它们可以一起同步操作。

## 13.2 TIMx 主要功能

通用 TIMx（TIM0、TIM1、TIM2、TIM3）定时器功能包括：

- 16 位向上、向下计数器
- 16 位可编程（可以实时修改）预分频器，计数器时钟频率的分频系数为 1～65536 之间的任意数值
- 周期定时功能
- PWM 功能
- 捕获功能
- 事件计数功能

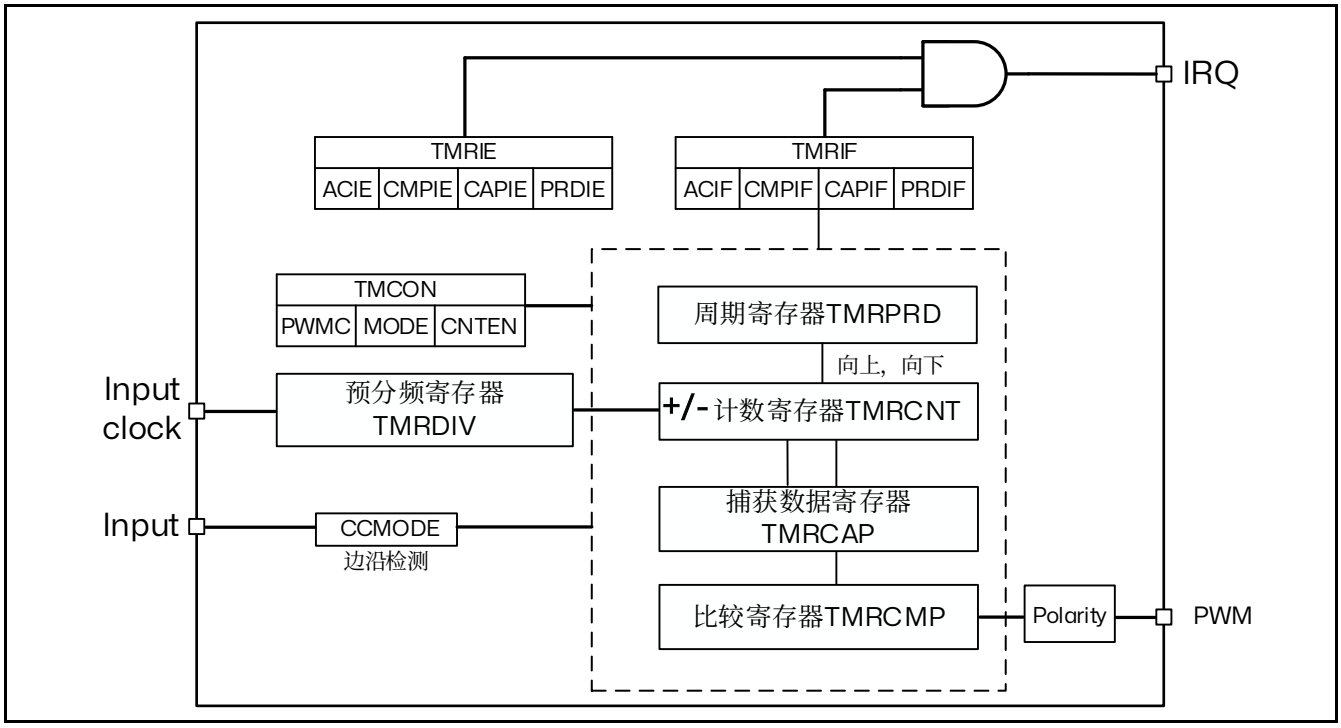



图 13.1 通用定时器框图

## 13.3 TIMx 功能描述

### 13.3.1 时基单元

可编程通用定时器的主要部分是一个 16 位计数器。这个计数器可以向上计数、向下计数或者中央计数。此计数器时钟由预分频器分频得到。

计数器和预分频器寄存器可以由软件读写，在计数器运行时仍可以读写。

时基单元包含：

- 计数器寄存器（TIMx_TMRCNT）
- 预分频器寄存器（TIMx_TMRDIV）

注：真正的计数器使能信号是在 CNTEN 的一个时钟周期后被设置。

**预分频器描述**

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个（在 TIMx_PSC 寄存器中的）16 位寄存器控制的 16 位计数器。这个控制寄存器带有缓冲器，它能够在工作时被改变。新的预分频器参数在下一次更新事件到来时被采用。图 13.2 和图 13.3 给出了在预分频器运行时，更改计数器参数的例子。

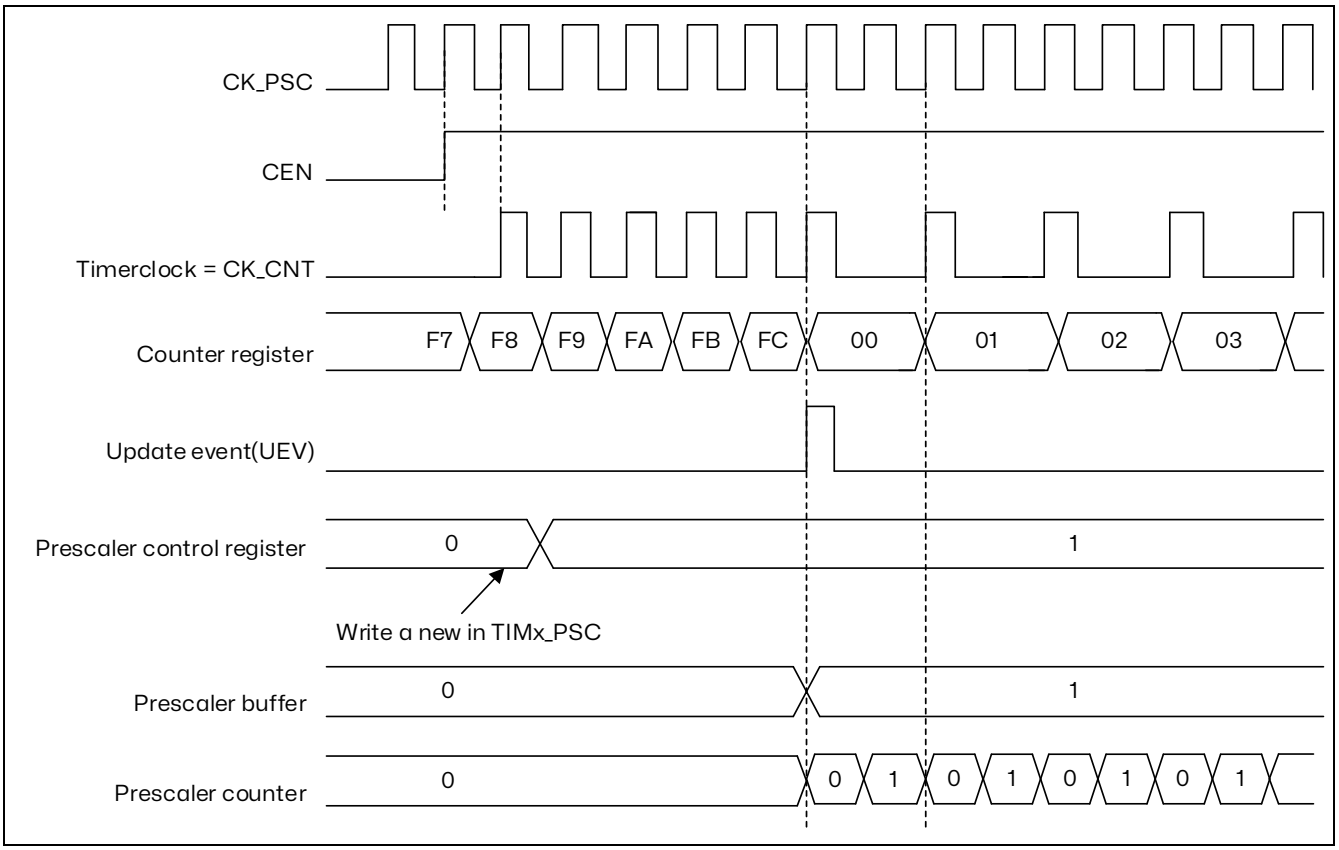



图 13.2 当预分频器的参数从 1 变到 2 时，计数器的时序图

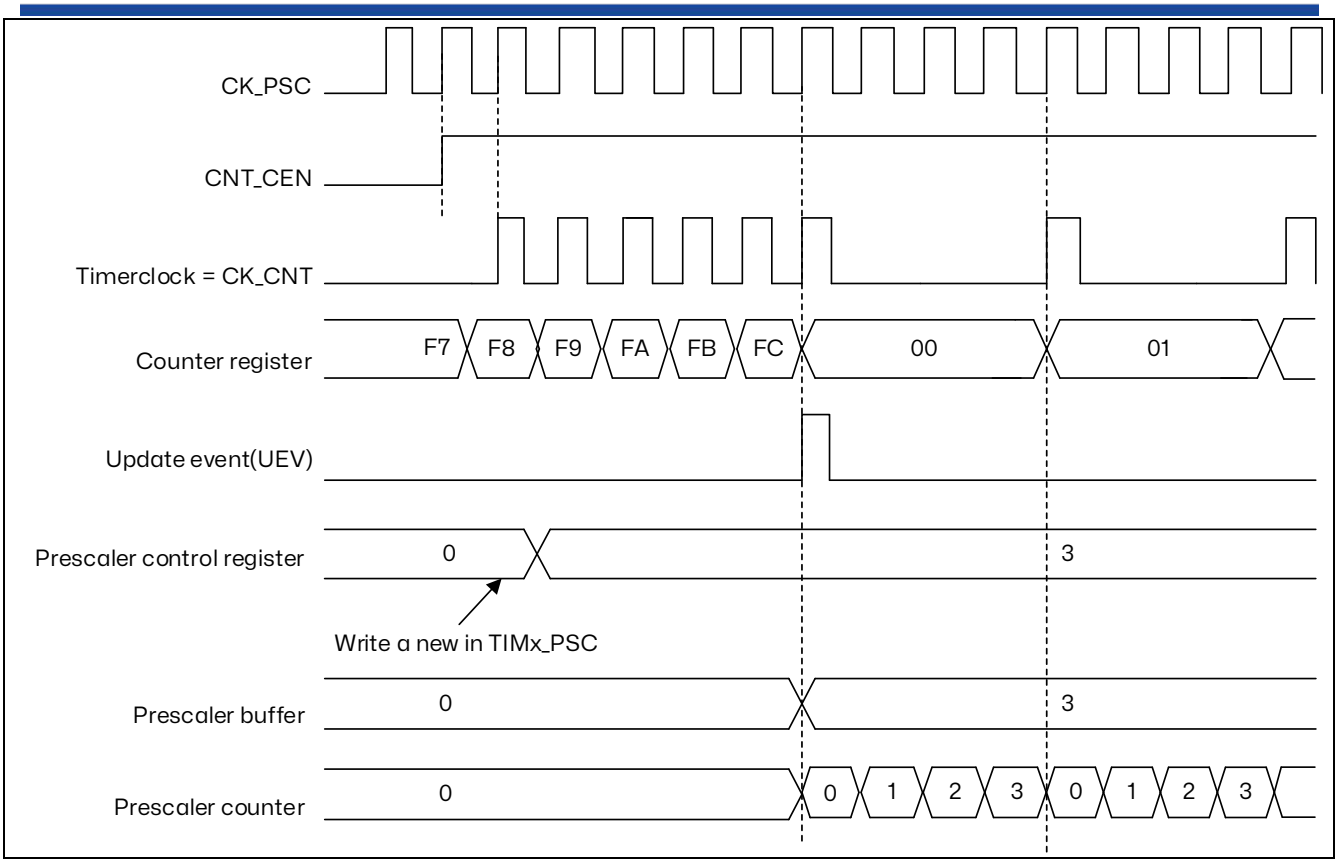



图 13.3 当预分频器的参数从 1 变到 4 时，计数器的时序图

### 13.3.2 时钟的选择

根据 SYSCLK_SEL[1:0]，可选择 LRC，HRC，HSE 作为系统时钟。时钟源为 input clock.。

### 13.3.3 周期定时模式

通用定时器包括一个 16 位计数器和周期寄存器。计数器的时钟由系统时钟（input clock）通过定时器单元内的预分频器（TMRDIV）分频得到，当使能计数器（CNTEN=1）后，定时器的计数器从 0 开始计数，当计数寄存器（TMRCNT）的值等于设定的周期寄存器（TMRPRD+1）溢出时会置位周期定时中断标志（PRDIF=1），如果使能周期定时中断（PRDIE=1），则会触发定时器周期中断，进入相应的周期中断服务程序。

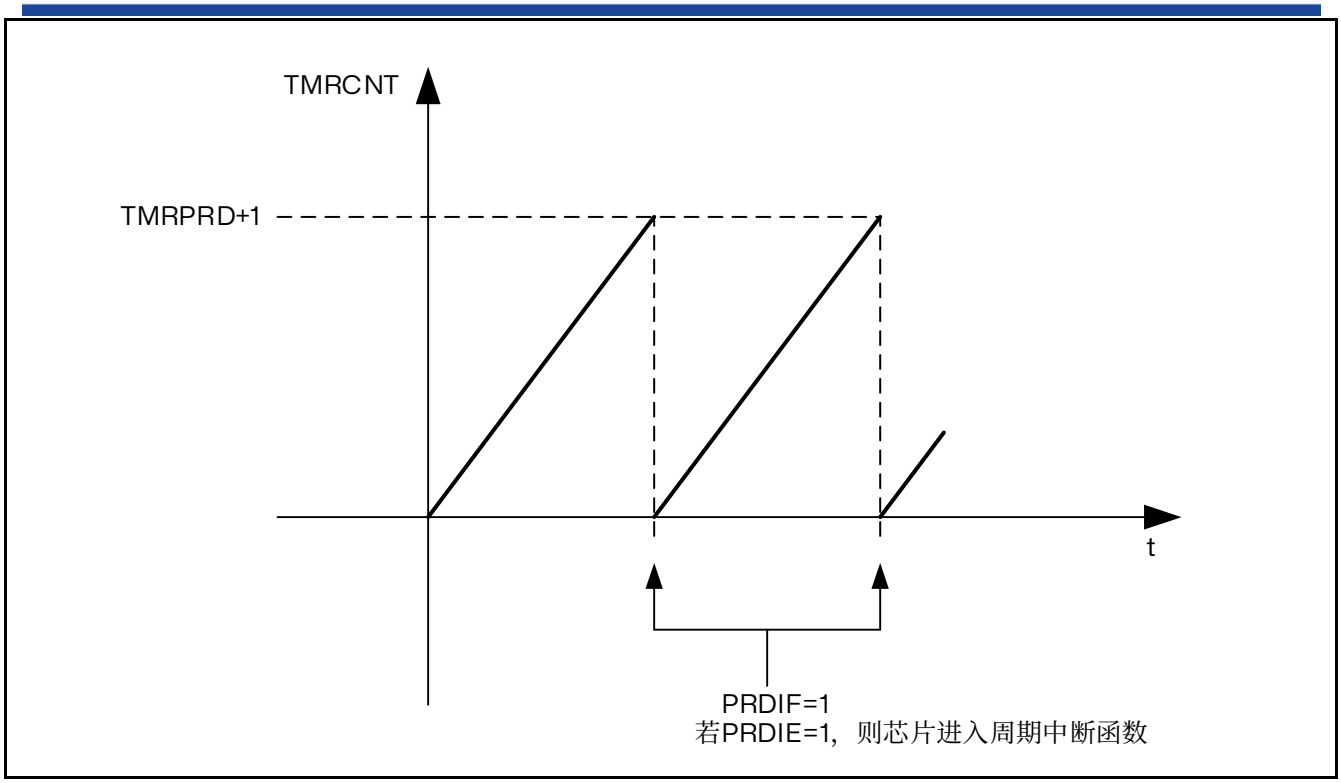



图 13.4 当计数器的值等于设定的周期寄存器溢出时，进入周期中断图

当周期定时中断标志置位后（PRDIF=1），TMRCNT 的值自动清 0，然后重新开始计数。TMRPRD 如被修改，在完成本次定时之后下一次生效。

周期定时功能仅支持向上计数。

### 13.3.4 PWM 模式

PWM 功能可通过寄存器 TMRCON.MODE[1:0]配置，同时需将对应的 GPIO 配置为 TMR 功能，配置成功后，相应的 TMR 引脚会输出 PWM 波形。PWM 的周期和占空比可通过寄存器 TMRPRD、TMRCMP 进行配置。PWM 计数方式分为向上计数、向下计数和中央计数方式。

**向上计数方式**

当使能计数器（CNTEN=1）之后，计数器开始从 0 计数，当计数寄存器（TMRCNT）的值等于设定的比较寄存器(TMRCMP+1)时，PWM 输出管脚发生电平翻转，同时置位比较中断标志（CMPIF=1）。计数器继续向上计数，当计数寄存器（TMRCNT）的值等于设定的周期寄存器（TMRPRD+1）溢出时，PWM 输出管脚再次发生电平翻转，同时置位周期定时中断标志（PRDIF=1）。当 TMRCMP>TMRPRD 时，得到 100%占空比,一直高电平，PRDIF 和 CMPIF 在 TMRCNT 数到 TMRPRD 溢出时同时置起；当 TMRCMP=0 时，得到 0%占空比,一直低电平，PRDIF 和 CMPIF 在 TMRCNT 数到 TMRPRD 溢出时同时置起。

PWM 输出波形如下图所示：

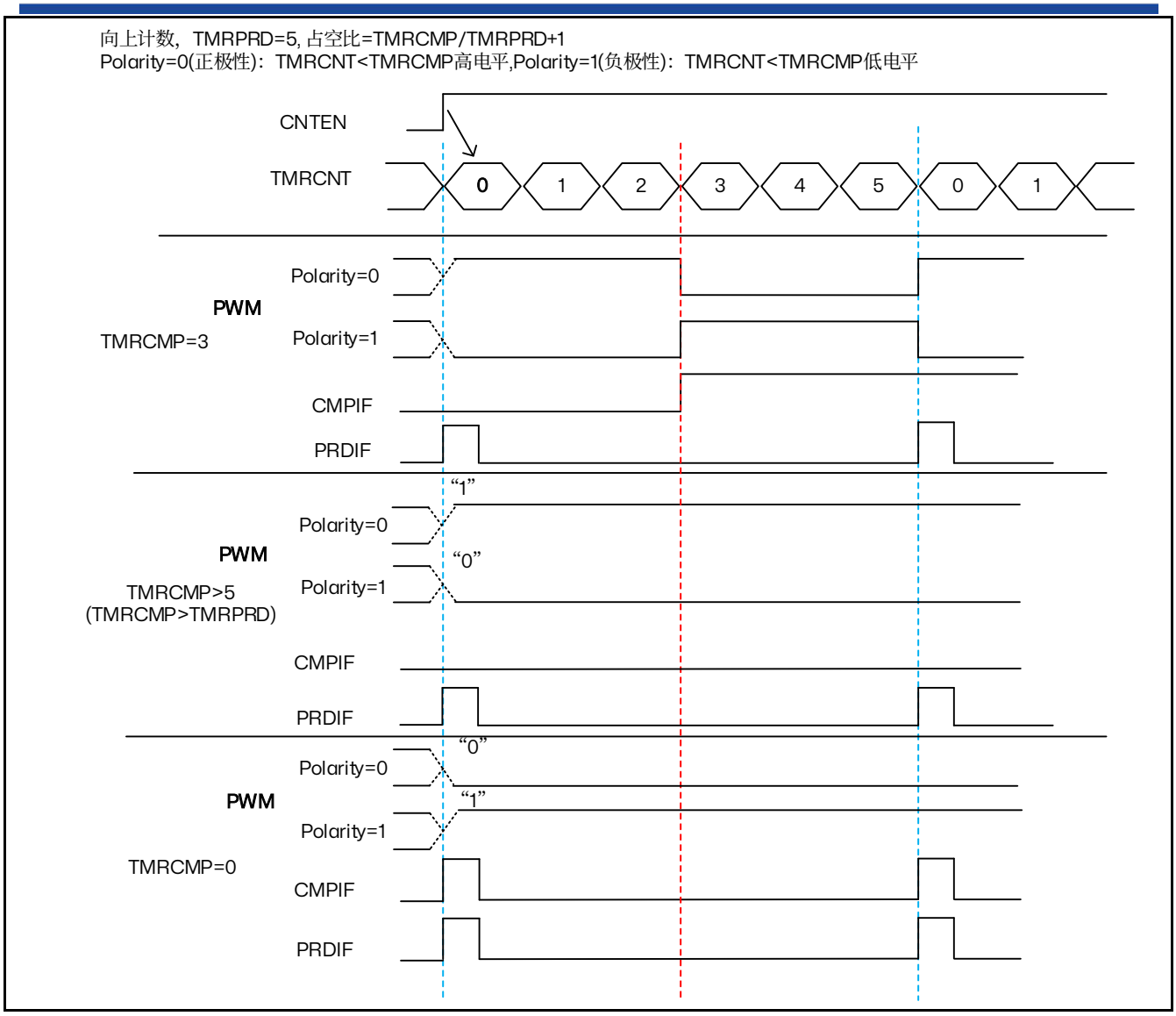



图 13.5 向上计数 PWM 输出波形图

**向下计数方式**

当使能计数器（CNTEN=1）之后，计数器开始从（TMRPRD）向下计数，当计数寄存器（TMRCNT）的值等于设定的比较寄存器（TMRCMP）时，PWM 输出管脚发生电平翻转，同时置位比较中断标志（CMPIF=1）。计数器继续向下计数，当计数寄存器（TMRCNT）的值等于 0 时，PWM 输出管脚再次发生电平翻转，同时置位周期定时中断标志（PRDIF=1）。当 TMRCMP>TMRPRD+1 时，得到 100%占空比,一直高电平，PRDIF 和 CMPIF 在 TMRCNT 数到 TMRPRD 溢出时同时置起。

PWM 输出波形如下图所示：

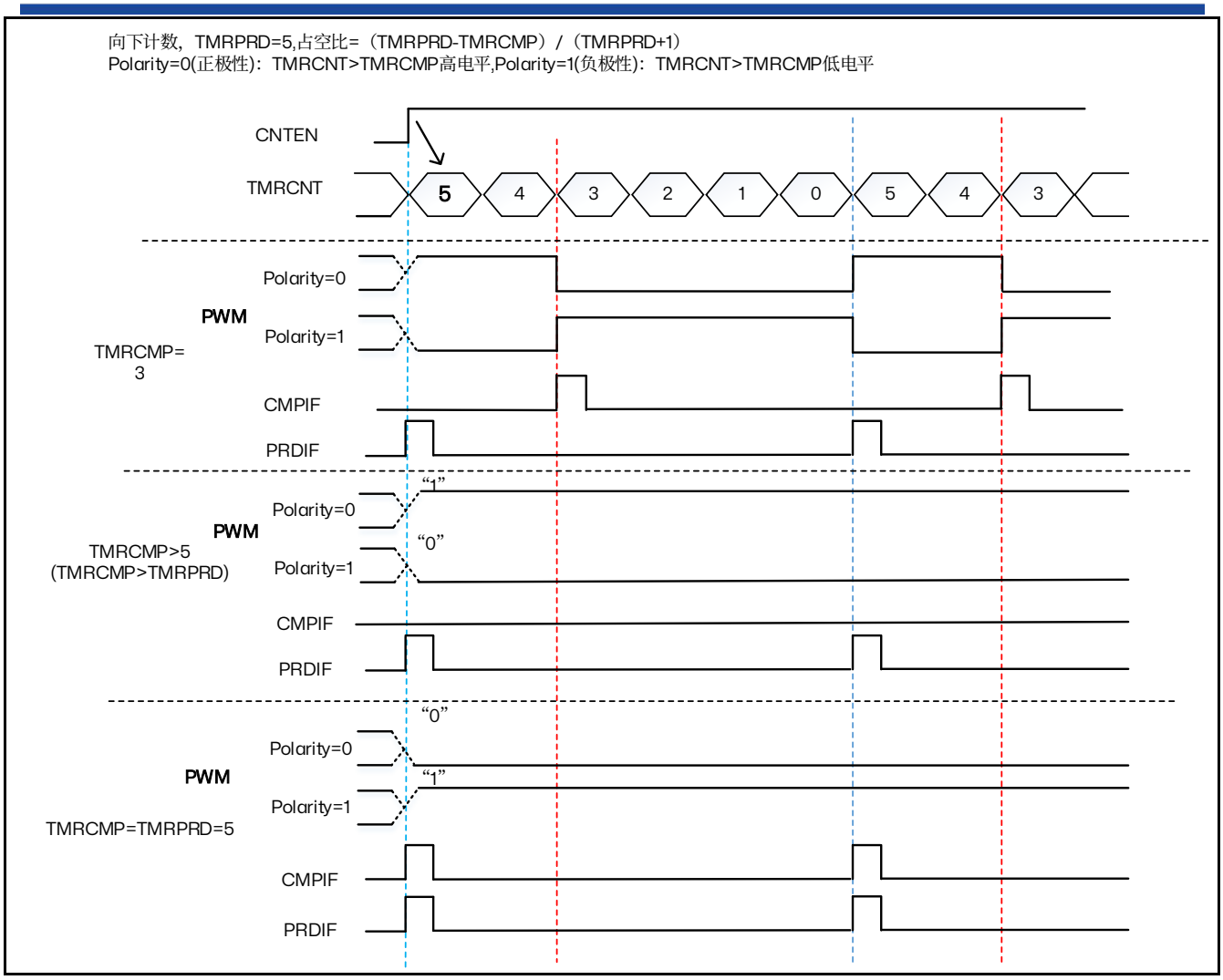



图 13.6 向下计数 PWM 输出波形图

**中央计数方式**

当计数寄存器（TMRCNT）从 0 开始向上计数，其值等于比较寄存器（TMRCMP）时，PWM 输出管脚发生电平翻转。计数器继续向上计数，当计数寄存器（TMRCNT）的值等于设定的周期寄存器（TMRPRD+1）溢出时，置位周期定时中断标志（PRDIF=1），但 PWM 输出管脚不发生电平翻转，计数器从周期寄存器（TMRPRD+1）溢出的值开始向下继续计数，当计数寄存器（TMRCNT）的值再次等于设定的比较寄存器（TMRCMP）时，PWM 输出管脚发生电平翻转。

PWM 输出波形如下图所示：

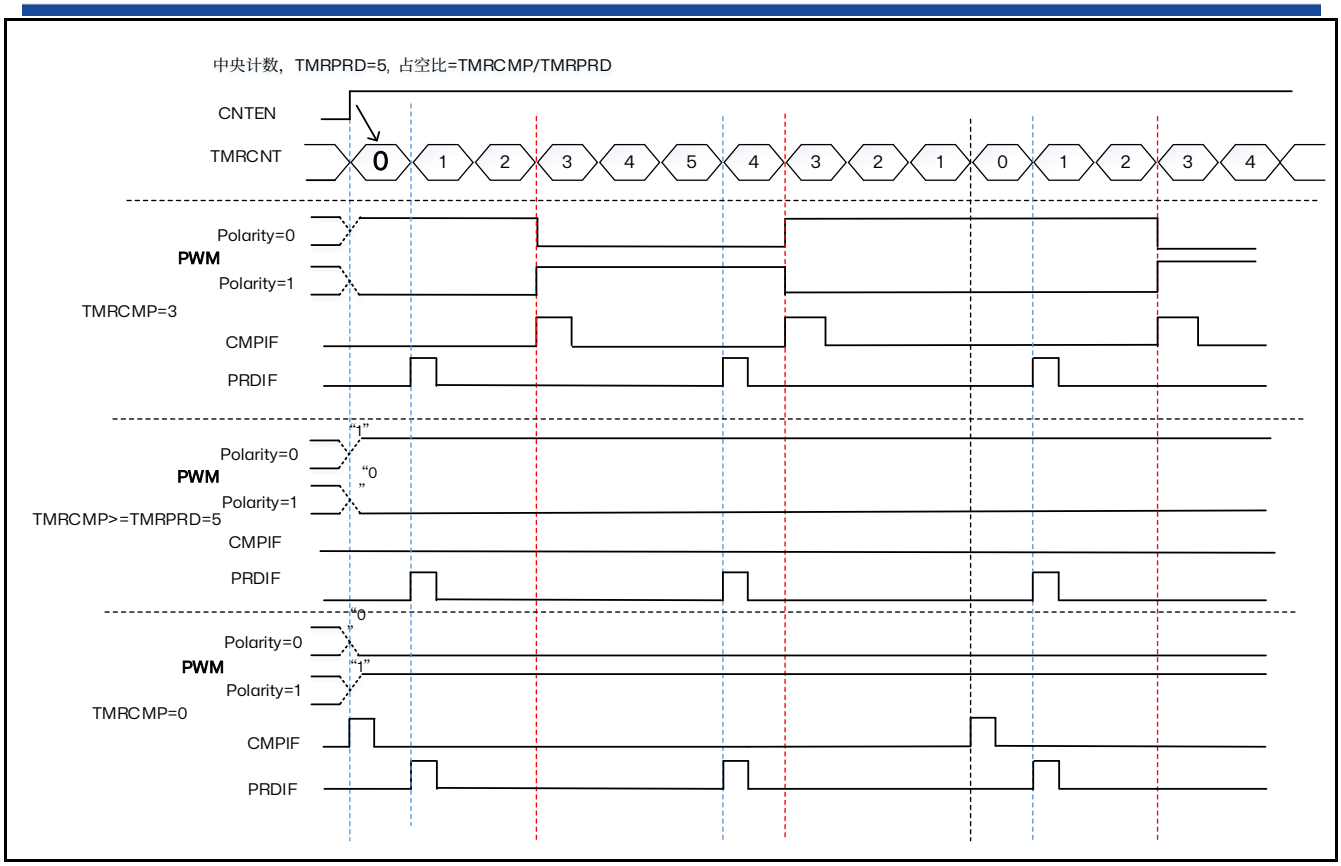



图 13.7 中央计数 PWM 输出波形图

### 13.3.5 捕获模式

在输入捕获模式下，检测到有效的沿，计数寄存器（TMRCNT）的当前值被锁定到捕获数据寄存器（TMRCAP）中。当捕获事件发生时，置位捕获中断标志（CAPIF=1），如果使能捕获中断（CAPIE=1），将 产生捕获中断，进入相应的捕获中断服务程序。

捕获过程中，如果没有检测到沿，当计数寄存器（TMRCNT）的值和设定的周期寄存器（TMRPRD+1）溢出相等时会置位周期定时中断标志（PRDIF=1），同时周期寄存器（TMRCNT）从 0 开始计数。如使能了周期定时中断（PRDIE=1），会进入相应的周期中断服务程序。

捕获功能仅支持向上计数。

功能主要相关寄存器：TMRCON，TMRCAP，TMRCNT，TMRPRD

### 13.3.6 事件计数模式

在输入事件计数模式下，检测到有效的沿，计数寄存器（TMRCNT）的当前值加 1。当计数寄存器（TMRCNT）的值等于设定的比较寄存器(TMRCMP)时，会置位事件计数中断标志（ACIF=1），同时计数寄存器（TMRCNT）自动清 0，重新开始计数。如果使能了事件计数中断（ACIE=1），将产生事件计数中断，进入相应的事件计数中断服务程序。

事件计数过程中，在 TMRCMP>TMRPRD 情况下，当计数寄存器（TMRCNT）的值等于设定的周期寄存器（TMRPRD+1）溢出时会置位周期定时中断标志（PRDIF=1），计数寄存器（TMRCNT）继续计数直到等于设定的比较寄存器(TMRCMP+1)，如果使能了周期定时中断(PRDIE=1)会进入周期中断服务程序。

单次最大计数个数为 0xFFFF，可以配合周期中断实现任意次数的组合。

比较寄存器（TMRCMP）如被修改，如果修改后的值小于当前计数寄存器（TMRCNT）的值，则立刻触发事件计数中断，同时计数寄存器（TMRCNT）清 0，重新开始计数；如果修改后的值大于当前计数寄存器（TMRCNT）的值，则继续本次计数。

在事件计数模式下，若需要修改 TMRCMP、TMRPRD，应先关闭计数器(CNTEN=0)，修改完后再开启计数器(CNTEN=1)。

事件计数功能仅支持向上计数。

### 13.3.7 中断模式

**周期定时中断**

当计数寄存器（TMRCNT）的值等于设定的周期寄存器（TMRPRD+1）溢出时，如果使能周期定时中断（PRDIE=1），则发生周期定时中断。此中断在任何功能模式下都可以产生。

**捕获中断**

当检测到外部输入信号相应沿时，如使能了捕获中断（CMPIE=1），则发生捕获中断。计数寄存器（TMRCNT）的值被锁定到捕获数据寄存器（TMRCAP）中。

**比较中断**

当计数寄存器（TMRCNT）的值等于设定的比较寄存器(TMRCMP)时,如使能了 PWM 比较中断(CMPIE=1)，则发生比较中断。

**事件计数中断**

当检测到设定次数的外部输入信号相应沿时，如使能了事件计数中断（ACIE=1），则发生事件计数中断。

## 13.4 TIMx 寄存器

### 13.4.1 定时器控制寄存器（TIMx_TMRCON）

偏移地址：0x00

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | Polarity | PWMC[1:0] | | CCMODE | MODE[1:0] | | CNTEN |
| | | | | | | | | | rw | rw | rw | rw | rw | rw | rw |

Bit 15:7 保留，必须保持为复位值

Bit 6 **Polarity**：PWM 输出电平极性选择：

0：正极性

1：负极性

Bit 5:4 **PWMC[1:0]**：PWM 工作模式选择：（PWM 计数方式）

00：向上计数

01：向下计数

1x：中央对齐

Bit 3 **CCMODE**：捕获/事件计数电平沿选择（当定时器配置为捕获/事件计数功能）：

0：上升沿

1：下降沿

Bit 2:1 **MODE**：Timer 功能选择:

00：事件计数功能

01：PWM 功能

10：捕获功能

11：周期定时功能

需将 GPIO 配置为 TMRx 功

Bit 0 **CNTEN**：计数器使能：

0：关闭

1：使能

### 13.4.2 预分频寄存器（TIMx_TMRDIV）

偏移地址：0x04

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRDIV[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **TMRDIV[15:0]**：预分频的范围在 0-65535 之间，经预分频器后的频率等于输入频率的 1/（TMRDIV[15:0] + 1）

### 13.4.3 周期寄存器（TIMx_TMRPRD）

偏移地址：0x08

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRPRD[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **TMRPRD[15:0]**：

该寄存器是一个 16 位的周期寄存器，计数的周期寄存器和 PWM 的周期寄存器都是该寄存器，在使用任何模式功能之前，需要设置周期寄存器。

$$Period=\frac{(TMPRD[15:0]+1)}{F/(TMRDIV[15:0]+1)}$$

### 13.4.4 捕获数据寄存器（TIMx_TMRCAP）

偏移地址：0x0C

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRCAP[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **TMRCAP[15:0]**：当发生捕获事件时，当前计数寄存器（TMRCNT）的值被存到该寄存器里

### 13.4.5 计数寄存器（TIMx_TMRCNT）

偏移地址：0x10

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRCNT[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **TMRCNT[15:0]**：计数器当前的计数值

### 13.4.6 比较寄存器（TIMx_TMRCMP）

偏移地址：0x14

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRCMP[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **TMRCMP[15:0]**：比较寄存器

比较寄存器有两个功能：

(1) Timer 做 PWM 功能的时候，当计数器达到（TMRCMP+1）的设定值时，PWM 输出翻转，同时置位比较中断标志（CMPIF=1），如使能了 PWM 比较中断（CMPIE=1），则发生比较中断。

(2) Timer 做事件计数功能的时候，当计数寄存器（TMRCNT）的值等于设定的比较寄存器(TMRCMP)时，会置位事件计数标志(ACIF=1)，同时计数寄存器（TMRCNT）会从 0 开始重新计数，如果使能了事件计数中断(ACIE=1)，则芯片会产生事件计数中断。

### 13.4.7 定时器中断使能寄存器（TIMx_TMRIE）

偏移地址：0x18

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | ACIE | CMPIE | CAPIE | PRDIE |
| | | | | | | | | | | | | rw | rw | rw | rw |

Bit 15:4 保留，必须保持为复位值

Bit 3 **ACIE**：事件计数中断使能

0：关闭

1：使能

Bit 2 **CMPIE**：比较中断使能

0：关闭

1：使能

Bit 1 **CAPIE**：捕获中断使能

0：关闭

1：使能

Bit 0 **PRDIE**：周期定时中断使能

0：关闭

1：使能

### 13.4.8 定时器中断标志寄存器（TIMx_TMRIF）

偏移地址：0x1C

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | ACIF | CMPIF | CAPIF | PRDIF |
| | | | | | | | | | | | | rw | rw | rw | rw |

Bit 15:4 保留，必须保持为复位值

Bit 3 **ACIF**：事件计数中断标志

0：未产生中断

1：产生中断（写 0 清 0）

Bit 2 **CMPIF**：比较中断标志

0：未产生中断

1：产生中断（写 0 清 0）

Bit 1 **CAPIF**：捕获中断标志

0：未产生中断

1：产生中断（写 0 清 0）

Bit 0 **PRDIF**：周期定时中断标志

0：未产生中断

1：产生中断（写 0 清 0）

# 14 运算放大器（PGA/OPA）

## 14.1 运算放大器简介

RX32S11 内嵌 3 个运算放大器，且都为可编程运算放大器（PGA）。

## 14.2 运算放大器特征

运算放大器具有以下特征：

- 轨对轨输入/输出
- PGA 可内部接地
- 输出连接到 I/O 或 ADC 上

## 14.3 运算放大器框图

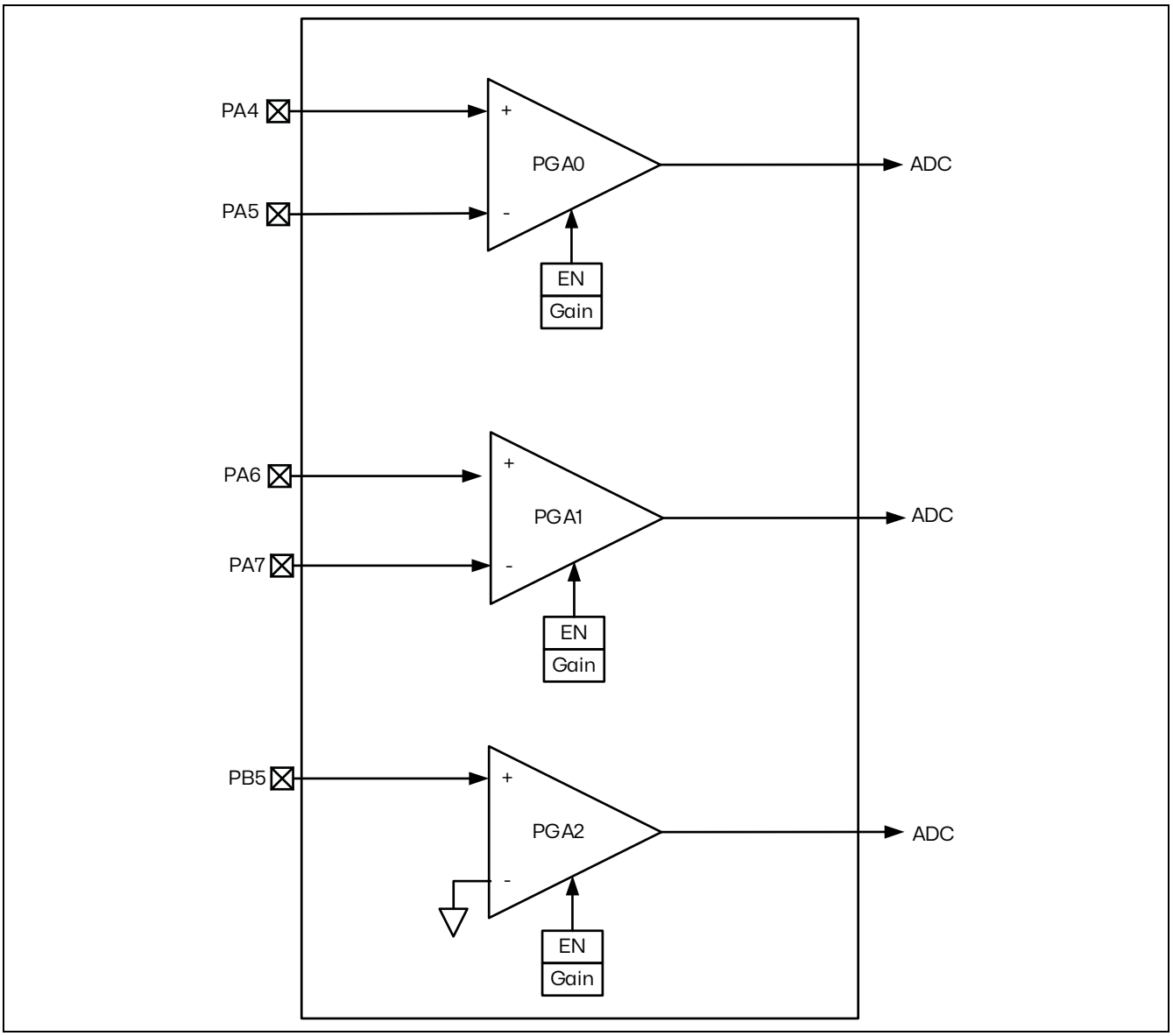



图 14.1 运算放大器框图

## 14.4 运算放大器寄存器

### 14.4.1 运算放大器控制寄存器（OPA_CR）

偏移地址：0x00

复位值：0x0000 0000

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | PGA2_Gain | | PGA2_EN | Reserved | PGA1_Gain | | PGA1_EN | Reserved | PGA0_Gain | | PGA0_EN |
| | | | | | rw | rw | rw | | rw | rw | rw | | rw | rw | rw |

Bit 31:11 保留

Bit 10:9 **PGA2_Gain**：PGA2 放大倍率选择

00b：2X

01b：4X

10b：8X

11b：保留

Bit 8 **PGA2_EN**：PGA2 使能位

1：PGA2 使能

0：PGA2 禁止

Bit 7 保留

Bit 6:5 **PGA1_Gain**：PGA1 放大倍率选择

00b：2X

01b：4X

10b：8X

11b：保留

Bit 4 **PGA1_EN**：PGA1 使能位

1：PGA1 使能

0：PGA1 禁止

Bit 3 保留

Bit **2:1 PGA0_Gain**：PGA0 放大倍率选择

00b：2X

01b：4X

10b：8X

11b：保留

Bit 0 **PGA0_EN**：PGA0 使能位

1：PGA0 使能

0：PGA0 禁止

# 15 比较器（CMP）

## 15.1 比较器简介

RX32S11 内嵌 2 个轨对轨比较器，可独立使用，也可与高级定时器或 PGA 配合使用。

## 15.2 比较器特征

比较器具有以下特征：

- 轨对轨输入/输出
- 具有迟滞功能
- 可选遮罩源

## 15.3 比较器开关控制

通过设置 CMPx_CR 寄存器的 EN 位可给 CMP 上电。设置 EN 位时，它将 CMP 从断电状态唤醒，清除 EN 位可停止比较器工作。

## 15.4 比较器输入和输出

CMP4-5 有 4 个正向输入通道可选择。CMP4 的正向输入有三个 IO 可选，当选择 CMP4_INP 时，可通过 CMP4_IOPSEL 位选择哪一个 IO 进行输入。CMP4 的反向输入有三个 IO 可选，当选择 CMP4_INM 时，可通过 CMP4_IOMSEL 位选择哪一个 IO 进行输入。CMP4-5 的正向输入包含一个外部引脚通道和 3 个 PGA 的输入。CMP4 的反向输入包含一个外部引脚通道和 CRV 电压分压值，CMP5 的反向输入只有 CRV 电压分压值。

CMP4-5 的输出可以选择 TIM8 的刹车输入。

## 15.5 比较器锁定机制

比较器能用于安全的用途，比如过流或者过热保护。在某些特定的安全需求的应用中，有必要保证比较器设置不能被无效寄存器访问或者程序计数器破坏所改变。

为了这个目的，比较器控制和状态寄存器可以设为写保护(只读)。一旦设置完成，LOCK 位必须设为 1，这导致整个 CMPx_CR2 寄存器变成只读，包括 LOCK 位在内。写保护只能被 MCU 复位所清除。

## 15.6 比较器轮询功能

比较器能用当 P 端选择 GPIO 轮询时，CMP 的 INP 端口上的信号会选择为 GPIO，并在 3 个 GPIO 之间进行周期性的轮询变化。而 INM 端的信号可以配置为不轮询状态，由 INM_SEL 和 IOM_SEL 决定；或选择为 GPIO 并跟随 P 端轮询而轮询。

当 P 端选择 PGA 轮询时，CMP 的 INP 端口上信号会选择为 PGA，并在 3 个 PGAx_P 之间进行周期性的轮询变化。而 INM 端的信号可以配置为不轮询状态，由 INM_SEL 和 IOM_SEL 决定；

CMP5 只能在 P 端选择 PGA 轮询。

当 P 端选择不轮询状态时（轮询选择位为 00b 时），N 端的轮询选择位将无效。

## 15.7 比较器框图

比较器框图如下图所示：

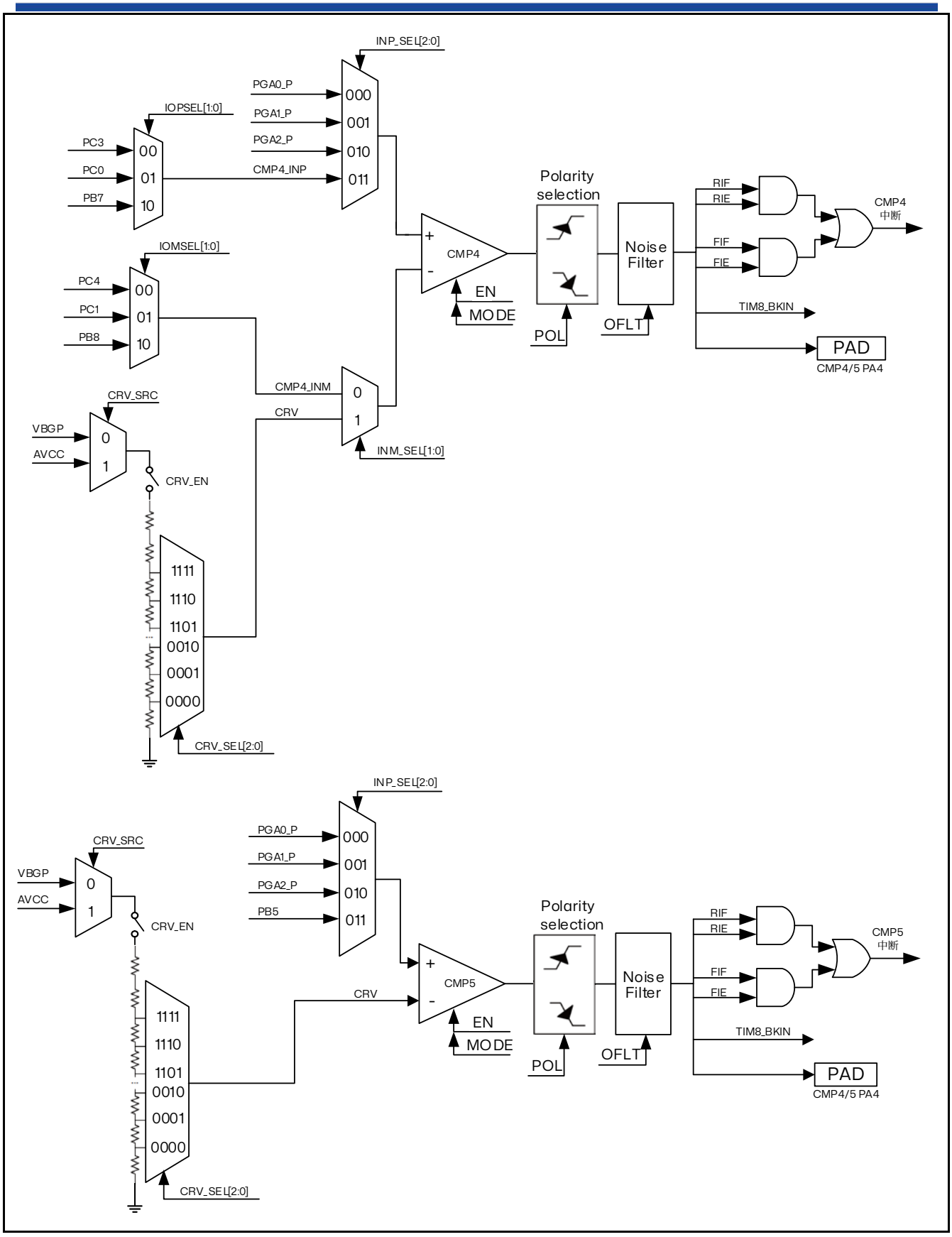

INP_SEL[2:0]
PGA0_P
PGA1_P
PGA2_P
000
001
010
011
IOPSEL[1:0]
PC3
PC0
PB7
00
01
10
CMP4_INP
IOMSEL[1:0]
PC4
PC1
PB8
00
01
10
CMP4
EN
MODE
Polarity selection
POL
Noise Filter
OFLT
RIF
RIE
FIF
FIE
CMP4 中断
TIM8_BKIN
PAD
CMP4/5 PA4
CMP4_INM
CRV
0
1
INM_SEL[1:0]
CRV_SRC
VBGP
AVCC
CRV_EN
1111
1110
1101
0010
0001
0000
CRV_SEL[2:0]
INP_SEL[2:0]
PGA0_P
PGA1_P
PGA2_P
PB5
000
001
010
011
CRV_SRC
VBGP
AVCC
CRV_EN
CRV
CMP5
EN
MODE
Polarity selection
POL
Noise Filter
OFLT
RIF
RIE
FIF
FIE
CMP5 中断
TIM8_BKIN
PAD
CMP4/5 PA4
1111
1110
1101
0010
0001
0000
CRV_SEL[2:0]

图 15.1 比较器框图

## 15.8 比较器迟滞功能

该比较器包括一个可编程迟滞，避免带有噪声输入信号的杂散输出转换。

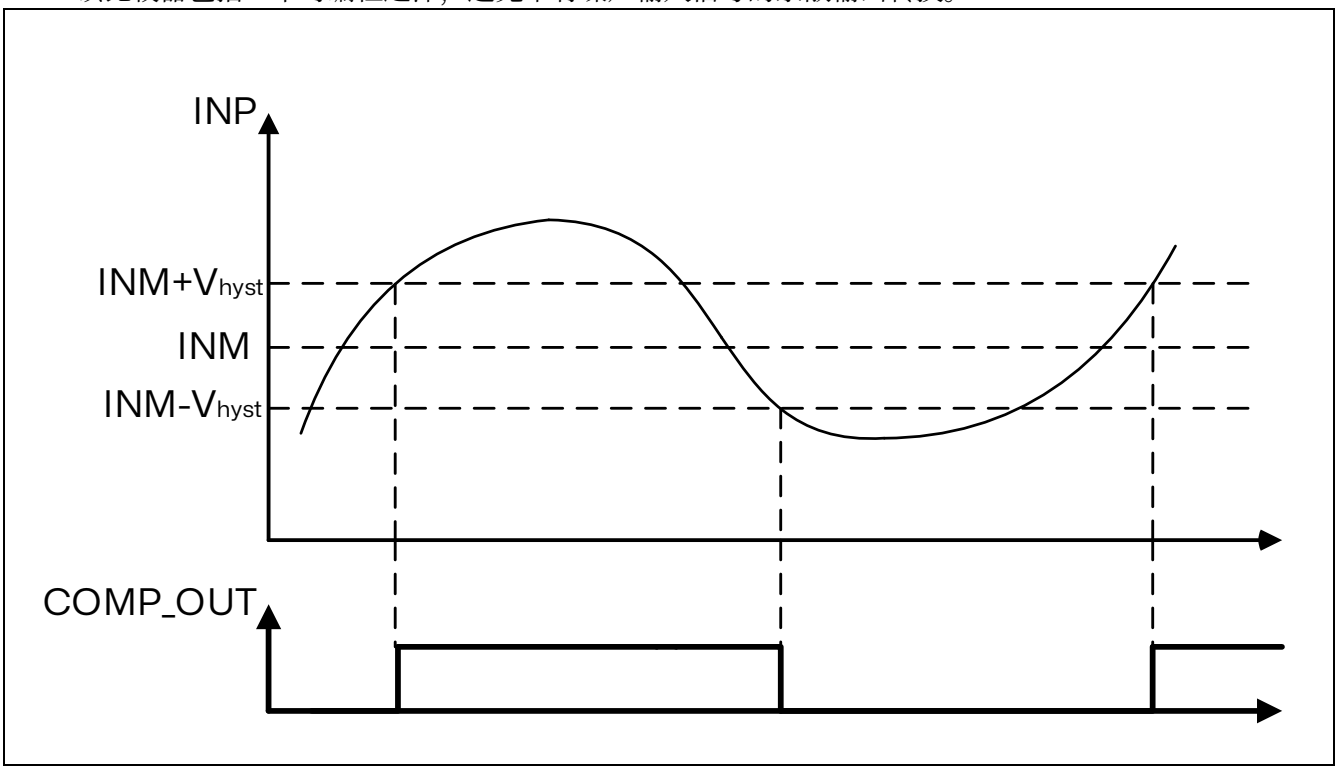



图 15.2 比较器迟滞

## 15.9 比较器消隐功能

消隐功能的目是防止在 PWM 周期开始时的短电流尖峰时电流调节跳闸。这里通过设置一个带有计时器输出比较信号的遮蔽窗来实现的。每个比较器通道有软件通过对应的 CMP_CR2 寄存器的 BLANKING [1:0]位域单独选择遮蔽源。

*注意：消隐功能只对高级定时器的刹车信号有效果*

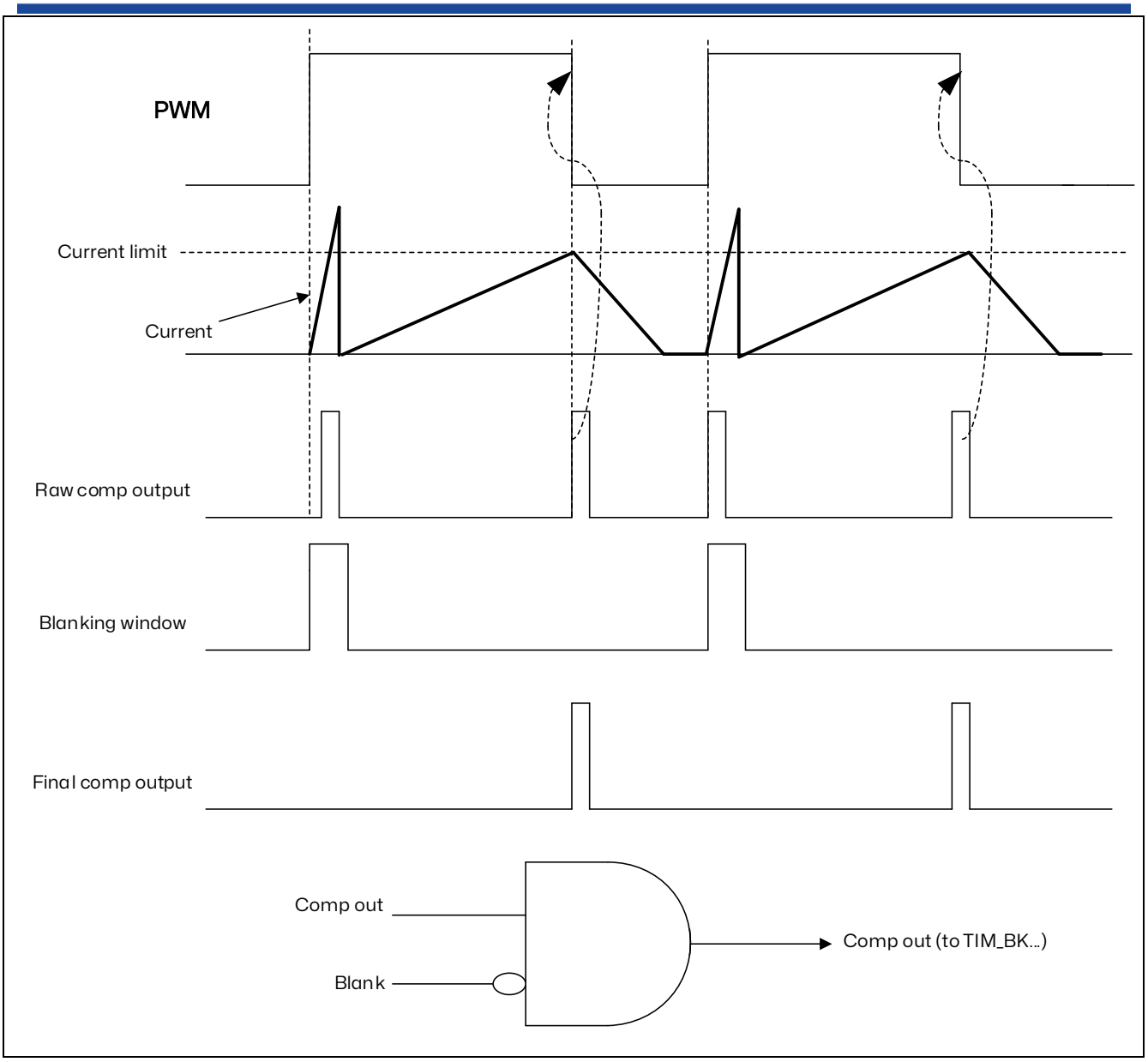



图 15.3 比较器输出遮罩

## 15.10 比较器寄存器

### 15.10.1 CMP 控制寄存器 1（CMPx_CR1）

偏移地址：0x00

复位值：0x0000 0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | Direct | FHYST | | RHYST | |
| | | | | | | | | | | | | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FIF | RIF | FIE | RIE | Reserved | OFLT | | | LOCK | OUT | Ready | Reserved | POL | MODE | | EN |
| rw | rw | rw | rw | | rw | rw | rw | rw | r | | | rw | rw | | rw |

Bit 31:21 保留，必须保持为复位值

Bit 20 **Direct**：输出引脚控制位

1：引脚输出的信号为模拟输出

0：引脚输出的信号为经过数字处理的信号

Bit 19:18 **FHYST**：下降迟滞电压

00：0mV

01：5mV

10：10mV

11：20mV

Bit 17:16 **RHYST**：上升迟滞电压

00：0mV

01：5mV

10：10mV

11：20mV

Bit 15 **FIF**：

1：下降沿中断事件发生

0：无下降沿中断事件

Bit 14 **RIF**：

1：上升沿中断事件发生

0：无上升沿中断事件

Bit 13 **FIE**：

1：下降沿中断使能

0：下降沿中断禁止

Bit 12 **RIE**：

1：上升沿中断使能

0：上升沿中断禁止

Bit 11 保留，必须保持为复位值

Bit 10:8 **OFLT**：比较器的输出滤波,连续的 PCLK 时钟比较输出不变则认为有效,否则保持不变

000b：1 个时钟周期,无滤波

001b：4 个时钟周期

010b：16 个时钟周期

011b：32 个时钟周期

100b：64 个时钟周期
101b：128 个时钟周期
110b：256 个时钟周期
111b：512 个时钟周期

Bit 7 **LOCK：**
1：CR2 只能 read
0：CR2 能 read/write
*注：*CMP4 *和*CMP5 *有效*

Bit 6 **OUT：**比较器的输出,read only
1：高输出
0：低输出
*注：*CPU *读取*OUT *值时，*OFLT *值必须设为*001b *以上*

Bit 5 **Ready：**输出有效标志
1：输出有效
0：输出无效

Bit 4 保留，必须保持为复位值

Bit 3 **POL：**输出极性
1：反相输出
0：同相输出

Bit 2:1 **MODE：**
00b:极低功率
01b:低功率
10b:中等速率
11b:高速率

Bit 0 **EN：**
1:比较器使能
0:比较器禁止

### 15.10.2 CMP 控制寄存器 2（CMPx_CR2）

偏移地址：0x04
复位值：0x0000 0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BLANKING[1:0] | | Reserved | | | | | | | | | | | | | |
| rw | rw | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | IOMSEL[1:0] | | IOPSEL[1:0] | | INP_SEL[2:0] | | | INM_SEL[1:0] | | CRV_SRC | CRV_EN | CRV_SEL[2:0] | | | |
| | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:30 **BLANKING[1:0]：**
00：无消隐
01：TIM8 OC4 被选为消隐源
10：TIM8 OC5 被选为消隐源
其他配置：保留

Bit 29:15 保留，必须保持为复位值

Bit 14:13 **IOMSEL[1:0]：**该位只在 CMP4 中且在 INM_SEL 为 0 时有效

00：CH1_INP(PC4)
01：CH2_INP(PC1)
10：CH3_INP(PB8)
*注：仅在CMP4 中有效*

Bit 12:11 **IOPSEL[1:0]：**该位只在 CMP4 中且在 INP_SEL 为 11b 时有效，以下是 CMP4 的设定
00：CH1_INP(PC3)
01：CH2_INP(PC0)
10：CH3_INP(PB7)
11：保留
*注：仅在CMP4 中有效*

Bit 10:8 **INP_SEL[2:0]：**
000：PGA0_P
001：PGA1_P
010：PGA2_P
011：CMP5_PB5，CMP4 无效

Bit 7:6 **INM_SEL[1:0]：**
00：CMP4_GPIO
01：CRV
*注：仅在CMP4 中有效，CMP5 的INM 只有CRV*

Bit 5 **CRV_SRC：**比较器外部参考电压源选择
1：AVCC
0：VBGP(1.2V)

Bit 4 **CRV_EN：**
1：比较器外部参考电压使能
0：比较器外部参考电压禁止
*注：CMP4 和CMP5 的CRV_EN 为相同讯号，*
*当CMP4 的CRV_EN 写入1，CMP5 的CRV_EN 也会变成1*

Bit 3:0 **CRV_SEL[2:0]：**比较器外部参考电压选择:

| | |
|---|---|
| 0000b：1/20 AVCC | 1000b：9/20 AVCC |
| 0001b：2/20 AVCC | 1001b：10/20 AVCC |
| 0010b：3/20 AVCC | 1010b：11/20 AVCC |
| 0011b：4/20 AVCC | 1011b：12/20 AVCC |
| 0100b：5/20 AVCC | 1100b：13/20 AVCC |
| 0101b：6/20 AVCC | 1101b：14/20 AVCC |
| 0110b：7/20 AVCC | 1110b：15/20 AVCC |
| 0111b：8/20 AVCC | 1111b：16/20 AVCC |

### 15.10.3 CMP 校正寄存器（CMPx_CAL）

偏移地址：0x08

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | CAL_NADJ[3:0] | | | | Reserved | | CAL_NEN | CAL_PEN | CAL_PADJ[3:0] | | | |
| | | | | rw | rw | rw | rw | | | rw | rw | rw | rw | rw | rw |

Bit 15:12 保留，必须保持为复位值

Bit 11:8 **CAL_NADJ[3:0]**：CMP NMOS 偏置调节

Bit 7:6 保留，必须保持为复位值

Bit 5 **CAL_NEN**：CMP NMOS 校正使能位

1：使能

0：禁止

Bit 4 **CAL_PEN**：CMP PMOS 校正使能位

1：使能

0：禁止

Bit 3:0 **CAL_PADJ[3:0]**：CMP PMOS 偏置调节

### 15.10.4 轮询控制寄存器（PLC）

偏移地址：0x0C

复位值：0x0000 0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | MATCH_DTA_CH3 | MATCH_DTA_CH2 | MATCH_DTA_CH1 | Reserved | | OFLT_CTL | MATCH_IT_EN | Reserved | | MaskT[1:0] | |
| | | | | | rw | rw | rw | | | rw | rw | | | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | PERIOD[1:0] | | Reserved | | | | CH3_EN | CH2_EN | CH1_EN | Reserved | INM_SEL | INP_SEL[1:0] | | |
| | | rw | rw | | | | | rw | rw | rw | | rw | rw | rw | |

Bit 31:27 保留，必须保持为复位值

Bit 26 **BLANKING MATCH_DTA_CH3**：CH3 输出结果匹配

0：低输出

1：高输出

Bit 25 **BLANKING MATCH_DTA_CH2**：CH2 输出结果匹配

0：低输出

1：高输出

Bit 24 **BLANKING MATCH_DTA_CH1**：CH1 输出结果匹配

0：低输出

1：高输出

Bit 23:22 保留，必须保持为复位值

Bit 21 **OFLT_CTL**：轮询滤波方案控制位

0：轮询模式下，满足 MaskT 后再执行 OFLT

1：轮询模式下，OFLT 和 MaskT 并行进行计数

Bit 20 **MATCH_IT_EN**：输出结果匹配中断使能位

0：不使能

1：使能

Bit 19:18 保留，必须保持为复位值

Bit 17:16 **MaskT[1:0]：**

00：0 个时钟周期

01：8 个时钟周期

10：16 个时钟周期

11：32 个时钟周期

Bit 15:14 保留，必须保持为复位值

Bit 13:12 **PERIOD[1:0]：轮询等待周期**

**每 n 个时钟周期切换到下一个轮询通道**

00：32 个时钟周期（mode 00 不可使用此档位）

01：64 个时钟周期

10：128 个时钟周期

11：256 个时钟周期

Bit 11:7 保留，必须保持为复位值

Bit 6 **CH3_EN：**轮询通道使能位

0：CH3 不参与轮询

1：CH3 参与轮询

Bit 5 **CH2_EN：**轮询通道使能位

0：CH2 不参与轮询

1：CH2 参与轮询

Bit 4 **CH1_EN：**轮询通道使能位

0：CH1 不参与轮询

1：CH1 参与轮询

Bit 3 保留，必须保持为复位值

Bit 2 **INM_SEL：**N 端轮询选择位

0：不轮询

1：GPIO 轮询

Bit 1:0 **INP_SEL[1:0]：**P 端轮询选择位

00：不轮询

10：GPIO 轮询

10：PGA 轮询

### 15.10.5 轮询状态寄存器（PLS）

偏移地址：0x10

复位值：0x0000 0000

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | MATCH_FLAG_CH3 | MATCH_FLAG_CH2 | MATCH_FLAG_CH1 | Reserved | | CH_STA[1:0] | | RDY | Reserved | | | | DAT_OUTP[2:0] | | |
| | | | | | | r | r | r | | | | | r | r | r |

Bit 31:15 保留，必须保持为复位值

Bit 14 **MATCH_FLAG_CH3：**CH3 输出结果匹配标志

0：不匹配

1：匹配

Bit 13 **MATCH_FLAG_CH2：**CH2 输出结果匹配标志

0：不匹配

1：匹配

Bit 12 **MATCH_FLAG_CH1：**CH1 输出结果匹配标志

0：不匹配

1：匹配

Bit 11:10 保留，必须保持为复位值

Bit 9:8 **CH_STA[1:0]：**轮询通道状态位

00：无状态

01：轮询至通道 1

10：轮询至通道 2

11：轮询至通道 3

Bit 7 **RDY：**通道轮询有效位（由硬件置起和清除）

0：无效

1：有效

Bit 6:3 保留，必须保持为复位值

Bit 2:0 **DAT_OUTP[2:0]：**通道输出位

分别对应三个正向输入时输出的结果，第 0 位对应 CH1_INP 的输入，第 1 位对应 CH2_INP 的输入，第 2 位对应 CH3_INP 的输入

0：低输出

1：高输出

# 16 电机专用协同处理器（ME）

## 16.1简介

RX32S11 提供一电机专用模块（Motor Engine），简称 ME，包含 FOC 运算所需的运算加速单元

- Clarke：将 3 轴向量转换成绝对坐标向量（α-β）
- Park：将绝对坐标向量转换成 d-q 轴向量
- RevPark：将 d-q 轴向量反转回绝对坐标向量
- SVPWM：将绝对坐标向量运算出 3 路 PWM 输出值
- DIV：除法运算加速器
- SQRT：平方根运算加速器
- PID：比例积分微分运算加速器（参考 PID 章节）

其中，坐标转换(CDTR)包含下图的方块 1，2，7
SIN COS Table 包含下图的方块 4
SVPWM 包含方块 8 和 9
SMO 包含方块 3
PID 模块提供方块 5 和 6 的硬件加速

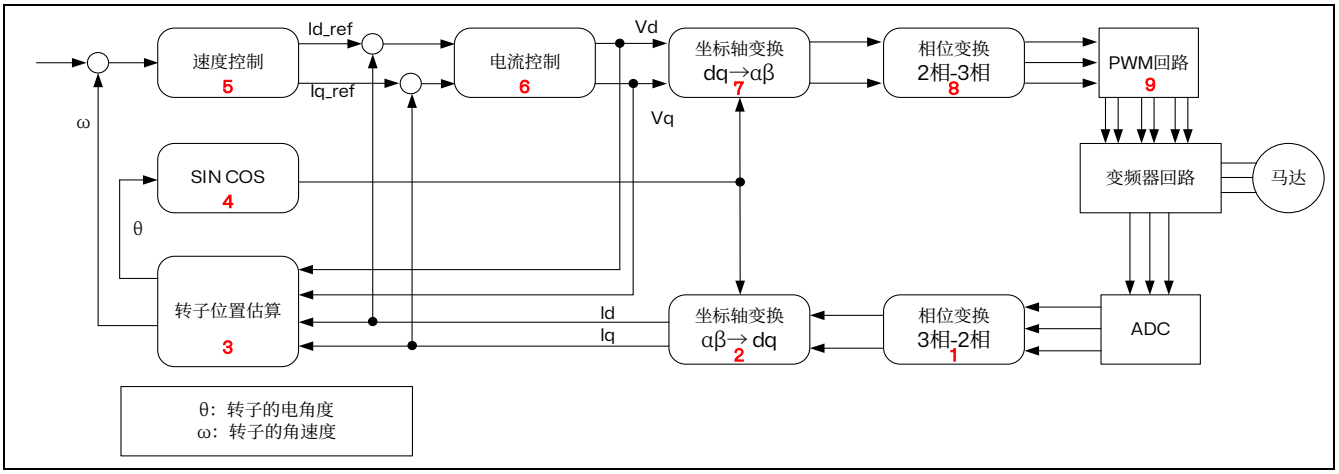



图 16.1 ME 框图

## 16.2 DIV 模块

### 16.2.1 概述

RX32S11 提供一个标准的 32bit 除法器，可以设定成有符号或无符号除法，被除数放在 DD,除数放在 DI，商数放在 DQ，余数放在 DR。

### 16.2.2 除法器的使用方法

1. 被除数写入到 DD，除数写入到 DI
2. CR 的 STR 写入 1,启动 32bit 除法运算
3. 当运算完后，可以透过 CR 的 RDY 是否为 1 来判断,软件可以写入 0 清除
4. 读取 DQ 为商，DR 为余数

注：

1. *当除法器启动运算后，在*CR.RDY=1 *之前，再次设定*CR.STR=1 *则忽略，必须等到上一笔除法运算完成才能再次启动运算*
2. *当除法运算中侦测到除数*DI *为*0，DIVF *会被置位为*1
3. CR.SN *选择有符号或无符号运算必须在启动*CR.STR *以前设置好*

### 16.2.3 DIV CR 寄存器（DIV_CR）

偏移地址：0x00

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Resverd | | | | | | | | | | | BUSY | DIVF | SN | RDY | STR |
| | | | | | | | | | | | r | rw | rw | rw | rw |

Bit 15:5 保留，始终读为 0。

Bit 4 **BUSY**:

1：DIV 运算中

0：DIV 闲置

Bit 3 **DIVF**:

1：运算错误，除数为 0，软件可清

0：运算正常

Bit 2 **SN**:

1：有符号

0：无符号

Bit 1 **RDY**:

1：除法运算完成

0：除法器闲置或运算中

Bit 0 **STR**:

1：启动除法运算，下一个 clock 硬件清除为 0

0：未启动除法运算

### 16.2.4 DIV DD 寄存器（DIV_DD）

偏移地址：0x04

复位值：0x0000 0000

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

Bit 31:0 **DD[31:0]**:

除法运算的被除数，必须在 CR.STR 设定为 1 前，将 DD 值写入。

当 CR.STR 启动后，修改 DD 值并不影响当前运算结果

### 16.2.5 DIV DI 寄存器（DIV_DI）

偏移地址：0x08

复位值：0XFFFF FFFF

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

Bit 31:0 **DI[31:0]**：

除法运算的除数，必须在 CR.STR 设定为 1 前,将 DI 值写入。

当 CR.STR 启动后，修改 DI 值并不影响当前运算结果。

*注：DI 值不能为 0*

### 16.2.6 DIV DQ 寄存器（DIV_DQ）

偏移地址：0x0C

复位值：0x0000 0000

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

Bit 31:0 **DQ[31:0]**：

除法运算的商数

### 16.2.7 DIV DR 寄存器（DIV_DR）

偏移地址：0x10

复位值：0x0000 0000

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

Bit 31:0 **DR[31:0]**：

除法运算的余数

## 16.3 SQRT 模块

### 16.3.1 概述

RX32S11 提供一个标准的开平方硬件加速器，输入寄存器 DI 为 32bit，输出寄存器 DO 为 16bit。

### 16.3.2 SQRT 使用方法

1. 将需要开平方的数值写入到 DI，自动启动开平方运算

2. 运算完成后,可以透过 CR 的 RDY 是否为 1 来判断，软件可以写 0 清除
3. 读出 DO 为开平方运算后的结果
4. 启动运算，BUSY 由硬件置 1；运算完成，BUSY 由硬件置 0；BUSY 为 1 时，DI 无法写入值。

注：

1. SQRT *必须等到上一笔运算完成才能再次启动运算*

### 16.3.3 SQRT CR 寄存器（SQRT_CR）

偏移地址：0x00

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Resverd | | | | | | | | | | | | | BUSY | RDY | Resverd |
| | | | | | | | | | | | | | r | rw | |

Bit 15:3 保留，始终读为 0。

Bit 2 **BUSY**：

1：SQRT 运算中

0：SQRT 闲置

Bit 1 **RDY**：

1：开平方运算完成

0：开平方硬件闲置或运算中

Bit 0 保留，始终读为 0。

### 16.3.4 SQRT DI 寄存器（SQRT_DI）

偏移地址：0x04

复位值：0x0000 0000

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

Bit 31:0 **DI[31:0]**：

开平方运算的输入值，写入数据后自动启动 SQRT 运算

### 16.3.5 SQRT DO 寄存器（SQRT_DO）

偏移地址：0x08

复位值：0x0000 0000

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

Bit 15:0 **DO[15:0]**：

开平方运算的输出值

## 16.4 SIN COS 表

### 16.4.1 使用方法

1. 写入 Angle 启动运算
2. 读出 Cos，Sin，IPDSin 值

## 16.5 坐标转换

### 16.5.1 使用方法

1. 设定坐标转换的通道 CDTR_CH[2:0]= =0b111
2. 写入 Ia，Ib，Ic，Cos，Sin，Vq，Vd
3. CR.CDTR_STR 写入 1 启动坐标转换运算
4. 运算是否完成,可以透过 CR.CDTR_RDY 为 1 来判断,软件可以写 0 清除
5. 转换完的数值为 Ialpha，Ibeta，Iq，Id，Valpha，Vbeta

注：

1. CDTR _CH[2:0]*的配置*

| CDTR _CH[2:0] | 项目 |
|---|---|
| 0b000 | 不启动运算 |
| 0b001 | Clarke |
| 0b010 | Park |
| 0b011 | Clarke, Park |
| 0b100 | Rev Park |
| 0b101 | Clarke, Rev Park |
| 0b110 | Park, Rev Park |
| 0b111 | Clarke, Park, Rev Park |

2. *当同时启动*Clarke，Park *的转换，必须先算完*Clarke *后才能算*Park
3. *启动运算，*CDTR_BUSY *由硬件置*1*；运算完后，*CDTR_BUSY *由硬件置*0*；*CDTR_BUSY *为*1 *时，*（Ia，Ib，Cos，Sin，Vq，Vd）*无法写入值。*

## 16.6 IPD 运算

### 16.6.1 使用方法

1. 写入 BEMFA 和 BEMFB
2. CR.IPD_STR 写入 1 启动 IPD 运算
3. 运算是否完成，可以透过 CR.IPD_RDY 为 1 来判断,软件可以写 0 清除
4. 读取 IPDTheta

注：

1. *启动运算，*IPD_BUSY *由硬件置*1*；运算完后，*IPD_BUSY *由硬件置*0*；*IPD_BUSY *为*1 *时，*（BEMFA,BEMFB）*无法写入值。*

## 16.7 SMO 运算

### 16.7.1 使用方法

1. 写入 SMOIalpha，SMOIbeta，Valpha，Vbeta
2. CR.SMO_STR 写入 1 启动 SMO 运算
3. 运算是否完成，可以透过 CR.SMO_RDY 为 1 来判断，软件可以写 0 清除
4. 读取 SMOTheta

注：

1. *启动运算，*SMO_BUSY *由硬件置*1*；运算完后，*SMO_BUSY *由硬件置*0*；*SMO_BUSY *为*1 *时，*（SMOIalpha，SOMIbeta，Valpha，Vbeta，Kslid，Kslf，KF，KG，EO）*无法写入值。*

## 16.8 SVPWM 运算

### 16.8.1 使用方法

1. 写入 Valpha，Vbeta
2. CR.SV_STR 写入 1 启动 SVPWM 运算
3. 运算是否完成,可以透过 CR.SV_RDY 为 1 来判断，软件可以写 0 清除
4. 读出 PDCH1，PDCH2，PDCH3，SVZone

注：

1. *启动运算，*SV_BUSY *由硬件置*1*；运算完后，*SV_BUSY *由硬件置*0*；*SV_BUSY *为*1 *时，*（FOVM，TRWM，LSMIN，Valpha，Vbeta）*无法写入值。*

## 16.9 中断功能

ME 中断列表如下

| 中断事件 | 事件标志 | 使能位 |
|---|---|---|
| 除法器错误事件 | ME_IFR.DIVFF | ME_IER.DIVFE |
| PID1 完成事件 | ME_IFR.PID1F | ME_IER.PID1E |
| PID2 完成事件 | ME_IFR.PID2F | ME_IER.PID2E |
| PID3 完成事件 | ME_IFR.PID3F | ME_IER.PID3E |
| 坐标转换完成事件 | ME_IFR.CDTRF | ME_IER.CDTRE |
| IPD 完成事件 | ME_IFR.IPDF | ME_IER.IPDE |
| SMO 完成事件 | ME_IFR.SMOF | ME_IER.SMOE |
| SVPWM 完成事件 | ME_IFR.SVF | ME_IER.SVE |
| FOC 运算完成事件 | ME_IFR.F4F | ME_IER.F4E |

注：

1. *当*ME_CR.F4_STR *写入*1，*依序启动*PID1，PID2，SVPWM，SMO *运算*
2. ME_IER.F4E *启动中断，当*PID1，PID2，SVPWM，SMO *都运算完成后，会产生一个中断事件*(ME_IFR.F4F)

## 16.10 ME 寄存器

### 16.10.1 ME 控制寄存器（ME_CR）

偏移地址：0x00

复位值：0x0000 0007

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | SV_FIVE | F4_STR |
| | | | | | | | | | | | | | | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SV_BUSY | SMO_BUSY | IPD_BUSY | CDTR_BUSY | SV_RDY | SV_STR | SMO_RDY | SMO_STR | IPD_RDY | IPD_STR | CDTR_RDY | CDTR_STR | SMO_CLR | CDTR_CH[2:0] | | |
| r | r | r | r | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 32:18 保留

Bit 17 **SV_FIVE**：

1：计算 Taon 不右移

2：计算 Taon 右移 1 位

Bit 16 **F4_STR**：

1：依序启动 PID1，PID2，SVPWM，SMO 运算

0：未启动

Bit 15 **SV_BUSY**：

1：SVPWM 运算中

0：SVPWM 闲置

Bit 14 **SMO_BUSY**：

1：SMO 运算中

0：SMO 闲置

Bit 13 **IPD_BUSY**：

1：IPD 运算中

0：IPD 闲置

Bit 12 **CDTR_BUSY**：

1：CDTR 运算中

0：CDTR 闲置

Bit 11 **SV_RDY**：

1：SVPWM 运算完成

0：SVPWM 硬件闲置或运算中

Bit 10 **SV_STR**：

1：启动 SVPWM 运算，下一个 clock 硬件清 0

0：未启动 SVPWM 运算

Bit 9 **SMO_RDY**：

1：SMO 运算完成

0：SMO 硬件闲置或运算中

Bit 8 **SMO_STR**：

1：启动 SMO 运算,下一个 clock 硬件清除为 0

0：未启动 SMO 运算

Bit 7 **IPD_RDY：**

1：IPD 运算完成

0：IPD 硬件闲置或运算中

Bit 6 **IPD_STR：**

1：启动 IPD 运算，下一个 clock 硬件清除为 0

0：未启动 IPD 运算

Bit 5 **CDTR_RDY：**

1：坐标转换运算完成

0：坐标转换硬件闲置或运算中

Bit 4 **CDTR_STR：**

1：启动坐标转换运算，下一个 clock 硬件清除为 0

0：未启动坐标转换运算

Bit 3 **SMO_CLR：**

1：清除 Ealpha，Ebeta，Zalpha，Zbeta，IalphaError，IbetaError，EstIalpha，EstIbeta 为 0

0：不清 0 动作

Bit 2:0 **CDTR_CH[2:0]：**坐标转换信道选择,可透过下列组合

BIT0：写入 1，坐标转换进行 Clarke 运算，反之写 0

BIT1：写入 1，坐标转换进行 Park 运算，反之写 0

BIT2：写入 1，坐标转换进行 Rev Park 运算，反之写 0

### 16.10.2 ME Ia 寄存器（ME_Ia）

偏移地址：0x04

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Ia[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Ia[15:0]：**Ia 值，Clarke 运算的输入。

### 16.10.3 ME Ib 寄存器（ME_Ib）

偏移地址：0x08

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Ib[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Ib[15:0]：**Ib 值，Clarke 运算的输入。

### 16.10.4 ME Ic 寄存器（ME_Ic）

偏移地址：0x0C

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Ic[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Ic[15:0]：**Ic 值，Clarke 运算的输入。

### 16.10.5 ME Ialpha 寄存器（ME_Ialpha）

偏移地址：0x10

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Ialpha[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Ialpha[15:0]**：Ialpha 值，Clarke 运算的输出，Park 运算的输入。

### 16.10.6 ME Ibeta 寄存器（ME_Ibeta）

偏移地址：0x14

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Ibeta[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Ibeta[15:0]**：Ibeta 值，Clarke 运算的输出，Park 运算的输入。

### 16.10.7 ME Iq 寄存器（ME_Iq）

偏移地址：0x18

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Iq[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Iq[15:0]**：Iq 值，Park 运算的输出。

### 16.10.8 ME Id 寄存器（ME_Id）

偏移地址：0x1C

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Id[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Id[15:0]**：Id 值，Park 运算的输出。

### 16.10.9 ME Vq 寄存器（ME_Vq）

偏移地址：0x20

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Vq[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Vq[15:0]**：Vq 值，Rev Park 运算的输入。

### 16.10.10 ME Vd 寄存器（ME_Vd）

偏移地址：0x24

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Vd[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Vd[15:0]**：Vd 值，Rev Park 运算的输入。

### 16.10.11 ME Valpha 寄存器（ME_Valpha）

偏移地址：0x28

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Valpha[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Valpha[15:0]**：Valpha 值，Rev Park 运算的输出，SVPWM 运算的输入。

### 16.10.12 ME Vbeta 寄存器（ME_Vbeta）

偏移地址：0x2C

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Vbeta[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Vbeta[15:0]**：Vbeta 值，Rev Park 运算的输出，SVPWM 运算的输入。

### 16.10.13 ME Angle 寄存器（ME_Angle）

偏移地址：0x30

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Angle[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Angle[15:0]**：输入角度值，范围为 0-1023，当进行坐标转换 Park 和 Rev Park 前，要先写入此寄存器。

### 16.10.14 ME Cos 寄存器（ME_Cos）

偏移地址：0x34

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Cos[15:0] | | | | | | | | | | | | | | | |
| r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

Bit 15:0 **Cos[15:0]**：根据 Angle 值，输出对应的 Cos table 值。

### 16.10.15 ME Sin 寄存器（ME_Sin）

偏移地址：0x38

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Sin[15:0] | | | | | | | | | | | | | | | |
| r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

Bit 15:0 **Sin[15:0]**：根据 Angle 值，输出对应的 Sin table 值。

### 16.10.16 ME IPDSin 寄存器（ME_IPDSin）

偏移地址：0x3C

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IPDSin[15:0] | | | | | | | | | | | | | | | |
| r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

Bit 15:0 **IPDSin[15:0]**：根据 Angle 值，输出对应的 IPD Sin 值。

### 16.10.17 ME Kslid 寄存器（ME_Kslid）

偏移地址：0x40

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Kslid[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Kslid[15:0]**：输入 Kslid 参数。

### 16.10.18 ME Kslf 寄存器（ME_Kslf）

偏移地址：0x44

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Kslf[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **Kslf[15:0]**：输入 Kslf 参数。

### 16.10.19 ME KF 寄存器（ME_KF）

偏移地址：0x48

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| KF[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **KF[15:0]**：输入 KF 参数。

### 16.10.20 ME KG 寄存器（ME_KG）

偏移地址：0x4C

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| KG[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **KG[15:0]：**输入 KG 参数。

### 16.10.21 ME E0 寄存器（ME_E0）

偏移地址：0x50

复位值：0x4000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| E0[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **E0[15:0]：**E0 参数。

### 16.10.22 ME Ealpha 寄存器（ME_Ealpha）

偏移地址：0x54

复位值：0x0000

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

Bit 31:0 **Ealpha[31:0]：**Ealpha 值。

### 16.10.23 ME Ebeta 寄存器（ME_Ebeta）

偏移地址：0x58

复位值：0x0000

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

Bit 31:0 **Ebeta[31:0]：**Ebeta 值。

### 16.10.24 ME Zalpha 寄存器（ME_Zalpha）

偏移地址：0x5C

复位值：0x0000

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

Bit 31:0 **Zalpha[31:0]：** Zalpha 值。

### 16.10.25 ME Zbeta 寄存器（ME_Zbeta）

偏移地址：0x60

复位值：0x0000

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

Bit 31:0 **Zbeta[31:0]：** Zbeta 值。

### 16.10.26 ME IalphaError 寄存器（ME_IalphaError）

偏移地址：0x64

复位值：0x0000

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

Bit 31:0 **IalphaError[31:0]：** IalphaError 值。

### 16.10.27 ME IbetaError 寄存器（ME_IbetaError）

偏移地址：0x68

复位值：0x0000

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

Bit 31:0 **IbetaError[31:0]：** IbetaError 值。

### 16.10.28 ME Estlalpha 寄存器（ME_Estlalpha）

偏移地址：0x6C

复位值：0x0000

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

Bit 31:0 **Estlalpha[31:0]：** Estlalpha 值。

### 16.10.29 ME Estlbeta 寄存器（ME_Estlbeta）

偏移地址：0x70

复位值：0x0000

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

Bit 31:0 **Estlbeta[31:0]：** Estlbeta 值。

### 16.10.30 ME SMOTheta 寄存器（ME_SMOTheta）

偏移地址：0x74

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SMOTheta[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **SMOTheta[15:0]：** SMOTheta 值。

### 16.10.31 ME SMOIalpha 寄存器（ME_SMOIalpha）

偏移地址：0x78

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SMOIalpha[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **SMOIalpha[15:0]：** SMOIalpha 值。

### 16.10.32 ME SMOIbeta 寄存器（ME_SMOIbeta）

偏移地址：0x7C

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SMOIbeta[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **SMOIbeta[15:0]：** SMOIbeta 值。

### 16.10.33 ME BEMFA 寄存器（ME_BEMFA）

偏移地址：0x80

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BEMFA[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **BEMFA[15:0]**：BEMFA 值。

### 16.10.34 ME BEMFB 寄存器（ME_BEMFB）

偏移地址：0x84

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BEMFB[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **BEMFB[15:0]**：BEMFB 值。

### 16.10.35 ME IPDTheta 寄存器（ME_IPDTheta）

偏移地址：0x88

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IPDTheta[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **IPDTheta[15:0]**：IPDTheta 值。

### 16.10.36 ME SVZone 寄存器（ME_SVZone）

偏移地址：0x90

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | SVZone[7:0] | | | | | | | |
| | | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 7:0 **SVZone [7:0]**：SVZone 值，SVPWM 运算的输出。

### 16.10.37 ME FOVM 寄存器（ME_FOVM）

偏移地址：0x94

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FOVM[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **FOVM[15:0]**：输入 PWM Full scale 值。

### 16.10.38 ME TPWM 寄存器（ME_TPWM）

偏移地址：0x98

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TPWM[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **TPWM[15:0]**：限制 PDCHx 的最大值。

### 16.10.39 ME LSMIN 寄存器（ME_LSMIN）

偏移地址：0x9C

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LSMIN[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **LSMIN[15:0]**：限制 PDCHx 的最小值。

### 16.10.40 ME PDCH1 寄存器（ME_PDCH1）

偏移地址：0xA0

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PDCH1[31:16] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PDCH1[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:0 **PDCH1[31:0]**：PwmDutyCH1 值，SVPWM 运算的输出

### 16.10.41 ME PDCH2 寄存器（ME_PDCH2）

偏移地址：0xA4

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PDCH2[31:16] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PDCH2[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:0 **PDCH2[31:0]**：PwmDutyCH2 值，SVPWM 运算的输出

### 16.10.42 ME PDCH3 寄存器（ME_PDCH3）

偏移地址：0xA8

复位值：0x0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PDCH3[31:16] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PDCH3[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 31:0 **PDCH3[31:0]**：PwmDutyCH3 值，SVPWM 运算的输出

### 16.10.43 ME IER 寄存器（ME_IER）

偏移地址：0xB0

复位值：0x0000

| 15 14 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 2 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | F4E | SVE | SMOE | IPDE | CDTRE | Reserved | PID3E | PID2E | PID1E | Reserved | DIVFE |
| | rw | rw | rw | rw | rw | | rw | rw | rw | | rw |

Bit 15:13 保留

Bit 12 **F4E**：FOC 计算中断使能

1：使能

0：禁止

Bit 11 **SVE**：SVPWM 中断使能

1：使能

0：禁止

Bit 10 **SMOE**：SMO 中断使能

1：使能

0：禁止

Bit 9 **IPDE**：IPD 中断使能

1：使能

0：禁止

Bit 8 **CDTRE**：坐标转换中断使能

1：使能

0：禁止

Bit 7 保留

Bit 6 **PID3E**：PID3 中断使能

1：使能

0：禁止

Bit 5 **PID2E**：PID2 中断使能

1：使能

0：禁止

Bit 4 **PID1E**：PID1 中断使能

1：使能

0：禁止

Bit 3:1 保留

Bit 0 **DIVFE**：DIV 错误中断使能

1：使能

0：禁止

### 16.10.44 ME IFR 寄存器（ME_IFR）

偏移地址：0xB4

复位值：0x0000

| 15 14 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 2 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | F4F | SVF | SMOF | IPDF | CDTRF | Reserved | PID3F | PID2F | PID1F | Reserved | DIVFF |
| | rw | rw | rw | rw | rw | | rw | rw | rw | | rw |

Bit 15:13 保留

Bit 12 **F4F**：FOC 计算中断旗标

1：发生 FOC 计算中断事件，写入 0 清除

0：无 FOC 中断事件

Bit 11 **SVF**：SVPWM 中断旗标

1：发生 SVPWM 计算中断事件，写入 0 清除

0：无 SVPWM 中断事件

Bit 10 **SMOF**：SMO 中断旗标

1：发生 SMO 计算中断事件，写入 0 清除

0：无 SMO 中断事件

Bit 9 **IPDF**：IPD 中断旗标

1：发生 IPD 计算中断事件，写入 0 清除

0：无 IPD 中断事件

Bit 8 **CDTRF**：坐标转换中断旗标

1：发生坐标转换计算中断事件，写入 0 清除

0：无坐标转换中断事件

Bit 7 保留

Bit 6 **PID3F**：PID3 中断旗标

1：发生 PID3 计算中断事件，写入 0 清除

0：无 PID3 中断事件

Bit 5 **PID2F**：PID2 中断旗标

1：发生 PID2 计算中断事件，写入 0 清除

0：无 PID2 中断事件

Bit 4 **PID1F**：PID1 中断旗标

1：发生 PID1 计算中断事件，写入 0 清除

0：无 PID1 中断事件

Bit 3:1 保留

Bit 0 **DIVFF**：DIV 错误中断旗标

1：发生 DIV 计算中断事件，写入 0 清除

0：无 DIV 中断事件

### 16.10.45 ME MCYC 寄存器（ME_MCYC）

偏移地址：0xC0

复位值：0x0677

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | DIV_ME[3:0] | | | | SQRT[3:0] | | | | DIV[3:0] | | | |
| | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:12 保留

Bit 11:8 **DIV_ME[3:0]**：多周期除法器(CPU)，范围 0x7-0xE

Bit 11:8 **SQRT[3:0]**：多周期平方根，范围 0x7-0xE

Bit 11:8 **DIV[3:0]**：多周期除法器(ME)，范围 0x6-0xE

# 17 PID 运算单元（PID）

## 17.1 PID 简介

PID 主要是由三个计算单元组成，比例单元(P)，积分单元(I)和微分单元(D)，主要公式如下：

$$OUT=K_P\times e(t)+ K_i\times \int_0^t e(T)dT +K_d \frac{d}{dt}e(t)$$

通过调整 Kp,Ki,Kd 的增益来调整其特性。

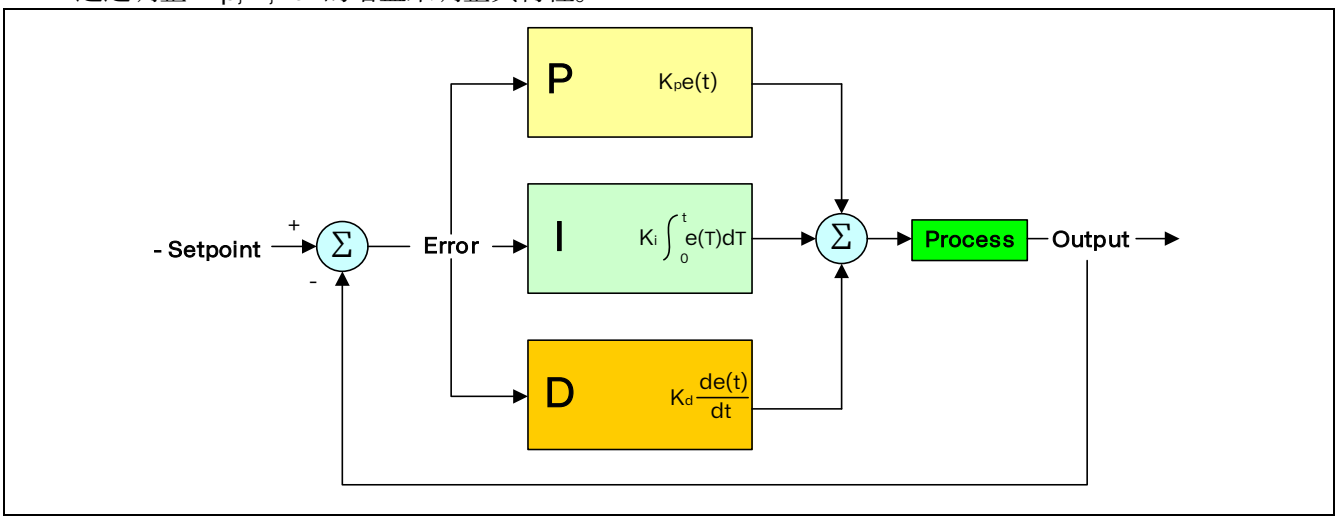



图 17.1 PID 概念图

RX32S11 提供 3 组 PID 硬件加速的配置，透过灵活的参数配置调整计算的方式。

## 17.2 PID 使用方法

1. 参考值写入到 REF，量测到反馈值写入到 FB
2. CR.STR 写入 1，启动 PID 运算
3. 运算完成后，可以透过 CR 的 RDY 是否为 1 来判断，软件可以写 0 清除
4. 读出 OUT 为 PID 运算完的结果

注：

1. *将* KPG *写入* 0 *则不计算* P *单元，* KIG *写入* 0 *则不计算* I *单元，* KDG *写入* 0
2. *启动运算，* BUSY *由硬件置*1*；运算完后，* BUSY *由硬件置*0*；* BUSY *为*1 *时，*(KPG，KIG，KDG，KIMULP，DIV，INTGLIM，OUTLIM)*无法写入值。*

## 17.3 PID 寄存器

### 17.3.1 PID CR 寄存器（PID_CR）

偏移地址：0x00

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Resverd | | | | | | | | | | | | BUSY | RDY | CLR | STR |
| | | | | | | | | | | | | r | rw | rw | rw |

Bit 15:4 保留，始终读为 0。

Bit 3 **BUSY**：

1：PID 运算中

0：PID 闲置

Bit 2 **RDY**：

1：PID 运算完成

0：PID 硬件闲置或运算中

Bit 1 **CLR**：

1：清除 INTG & ERR 为 0

0：不清除 INTG & ERR

Bit 0 **STR**：

1：启动 PID 运算,下一个 clock 硬件清除

0：未启动 PID 运算

### 17.3.2 PID REF 寄存器（PID_REF）

偏移地址：0x04

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| REF[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **REF[15:0]**：

输入的 Reference 值。ERR = REF – FB

### 17.3.3 PID FB 寄存器（PID_FB）

偏移地址：0x08

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FB[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **FB[15:0]**：

输入的 Feedback 值。ERR = REF – FB

### 17.3.4 PID OUT 寄存器（PID_OUT）

偏移地址：0x0C

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OUT[15:0] | | | | | | | | | | | | | | | |
| r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

Bit 15:0 **OUT[15:0]**：

PID 运算的结果

### 17.3.5 PID ERR 寄存器（PID_ERR）

偏移地址：0x10

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ERR[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **ERR[15:0]**：

PID 运算后保留的 Error 值。ERR = REF – FB

### 17.3.6 PID INTG 寄存器（PID_INTG）

偏移地址：0x14

复位值：0x0000 0000

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

Bit 31:0 **INTG[31:0]**：

PID 运算后保留的 Integral 值

### 17.3.7 PID KPG 寄存器（PID_KPG）

偏移地址：0x18

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| KPG[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **KPG[15:0]**：

KP Gain 值

### 17.3.8 PID KIG 寄存器（PID_KIG）

偏移地址：0x1C

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| KIG[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **KIG[15:0]**：

KI Gain 值

### 17.3.9 PID KDG 寄存器（PID_KDG）

偏移地址：0x20

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| KDG[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **KDG[15:0]**：

KD Gain 值

### 17.3.10 PID KIMULP 寄存器（PID_KIMULP）

偏移地址：0x24

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| KIMULP[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **KIMULP[15:0]**：

KI 放大倍率值（注：*放大倍率不能为*0）

### 17.3.11 PID DIV 寄存器（PID_DIV）

偏移地址：0x28

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | KDD[3:0] | | | | KID[3:0] | | | | KPD[3:0] | | | |
| | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:12 保留，始终读为 0。

Bit 11:8 **KDD[3:0]**：KP 除 2 的倍数，默认为 0

Bit 7:4 **KID[3:0]**：KI 除 2 的倍数，默认为 0

Bit 3:0 **KPD[3:0]**：KD 除 2 的倍数，默认为 0

### 17.3.12 PID INTGLIM 寄存器（PID_INTGLIM）

偏移地址：0x2C

复位值：0x0000 0000

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

Bit 31:0 **INTG[31:0]**：

限制 Integral 的上限为 INTGLIM，下限为-INTGLIM

*注：*INTGLIM *的值最大可设定为十六进制：*0x3FFFFFFF，*十进制：*1073741823

### 17.3.13 PID OUTLIM 寄存器（PID_OUTLIM）

偏移地址：0x30

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OUTLIM[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **OUTLIM[15:0]**：

限制 OUT 的上限为 OUTLIM，下限为-OUTLIM

# 18 实时时钟（RTC）

## 18.1 RTC 简介

实时时钟是一个独立的定时器。RTC 模块拥有一组连续计数的计数器，在相应软件配置下，可提供时钟日历的功能。修改计数器的值可以重新设置系统当前的时间和日期。

## 18.2 主要特性

- 可编程的预分频系数：分频系数最高为 1048576。
- 32 位的可编程计数器，可用于较长时间段的测量。
- RTC 单元提供实时时钟和一个周期性中断
- RTC 模块在 Hold 模式可被关闭，在低功耗下仍然正常运行。
- RTC 模块的时钟源为 LRC 时钟
- 1 个专门的可屏蔽中断：
  - 溢出中断，指示内部可编程计数器溢出并回转为 0 的状态。

## 18.3 功能描述

### 18.3.1 概述

RTC 核心由一组可编程计数器和 APB 总线连接组成，RTC 计数单元的时钟由低速 RC 时钟或者 2048 分频后得到时钟基准，如果开启中断就会有溢出，当 RTC2EN 被使能之后，溢出产生 RTC2IF 标志。

如果设置 CNT[15:0]＝00H，则 RTC 内部的秒表功能中断每经过 2048/Irc 的计时周期后，会置位 RTC2IF 标志，将产生中断的地址传送到 IRQ-RTC 中断向量 20 中。

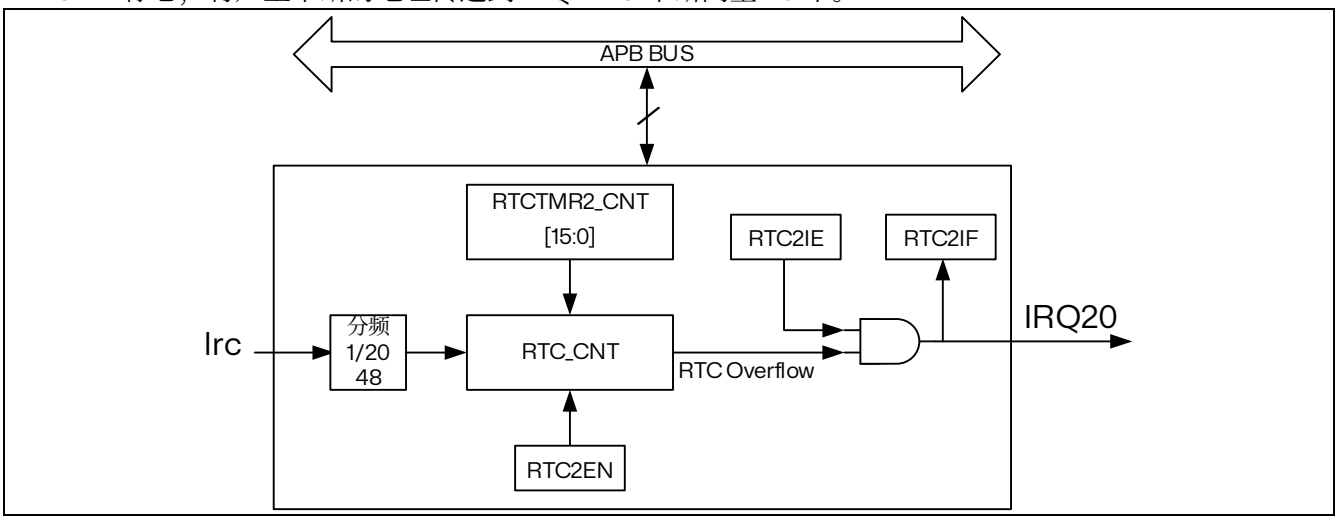



图 18.1 简化的 RTC 框图

### 18.3.2 复位过程

软件通过 APB 总线访问 RTC 的预分频值、计数器值。但是，相关的可读寄存器只在与 RTC APB 时钟进行重新同步的 RTC 时钟的上升沿被更新。RTC 标志也是如此的。

这意味着，如果 APB 接口曾经被关闭，而读操作又是在刚刚重新开启 APB 之后，则在第一次的内部

寄存器更新之前，从 APB 上读出的 RTC 寄存器数值可能被破坏了（通常读到 0）。

下述几种情况下能够发生这种情形：

- 发生系统复位或电源复位
- 系统刚从睡眠模式唤醒（参见第 5.3 节：低功耗模式）。

所有以上情况中，APB 接口被禁止时（复位、无时钟或断电）RTC 核仍保持运行状态。

*注：RTC 的APB 接口不受WFI 低功耗模式的影响。*

### 18.3.3 中断功能

RTC 提供 1 种中断源，共享 MCU 的 IRQ-RTC 中断向量，向量号 20。

RTC 的 1 种中断源由 RTCIE 控制其使能。具体的中断产生条件和中断清除步骤如下：

**RTC2IF：**RTC 定时器 2 中断标志

如设置 RTCTMR2=X，使能计数 RTC2EN 后，经过(X+1)*0.0625S 后，该标志位置位 1。

对该位写 0 清标志。

## 18.4 RTC 寄存器

### 18.4.1 RTC 控制寄存器（RTC_RTCCON）

地址偏移量：0x00

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | RTC2EN | Reserved | | | | | |
| | | | | | | | | | rw | | | | | | |

Bit 15:7 保留，必须保持为复位值

Bit 6 **RTC2EN：**RTC 定时器 2 使能位

0：RTC 定时器 2 被关闭

1：RTC 定时器 2 被使能，溢出产生 RTC2IF 标志。

Bit 5:0 保留，必须保持为复位值

### 18.4.2 RTC 中断使能寄存器（RTC_RTCIE）

偏移地址：0x04

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | RTC2IE | Reserved | | | | | |
| | | | | | | | | | rw | | | | | | |

Bit 15:7 保留，必须保持为复位值

Bit 6 **RTC2IE：**RTC 定时器 2 中断使能位

0：关闭

1：打开

Bit 5:0 保留，必须保持为复位值

### 18.4.3 RTC 中断标志寄存器（RTC_RTCIF）

偏移地址：0x08

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | RTC2IF | Reserved | | | | | |
| | | | | | | | | | rw | | | | | | |

Bit 15:7 保留，必须保持为复位值

Bit 6 **RTC2IF**：RTC 定时器 2 中断标志位

0：未产生中断

1：产生中断，写 0 清 0

Bit 5:0 保留，必须保持为复位值

### 18.4.4 RTC 定时器 2 寄存器（RTC_RTCTMR2）

偏移地址：0x14

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CNT[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **CNT[15:0]**：

CNT[15:0]用来表示一个 16BIT 的二进制的无符号整数，如果设置 CNT[15:0]＝00H，表示 RTC 内部的秒表功能中断每经过(00H+1)*2048/Irc=1*2048/Irc=2048/Irc 的计时周期后，置位 RTC2IF 标志。

*注：当定时器溢出时，如果用户没有关闭定时器，则定时器将从 0 开始重新计数。*

# 19 独立看门狗（IWDT）

## 19.1 IWDT 简介

RX32S11 内置一个独立看门狗，提供了更高的安全性、时间的精确性和使用的灵活性。独立看门狗可用来检测和解决由软件错误引起的故障。独立看门狗（IWDT）由专用的低速时钟（LRC）驱动，即使主时钟发生故障它也仍然有效。IWDT 最适合应用于那些需要看门狗作为一个在主程序之外，能够完全独立工作，并且对时间精度要求较低的场合。

## 19.2 IWDT 主要特性

- 采用硬件狗设计
- SLEEP 或者 HOLD 模式下 WDT 开启/关闭可选
- 调试模式下可关闭，且随着 CPU 停止而停止，方便调试使用

## 19.3 WDT 功能描述

看门狗由内部低频 RC 时钟驱动，当计时器计满预定时间则发出溢出脉冲，产生 WDT 复位信号。在溢出脉冲发生前将 Watchdog Timer 清零，则不会发出 WDT 复位。

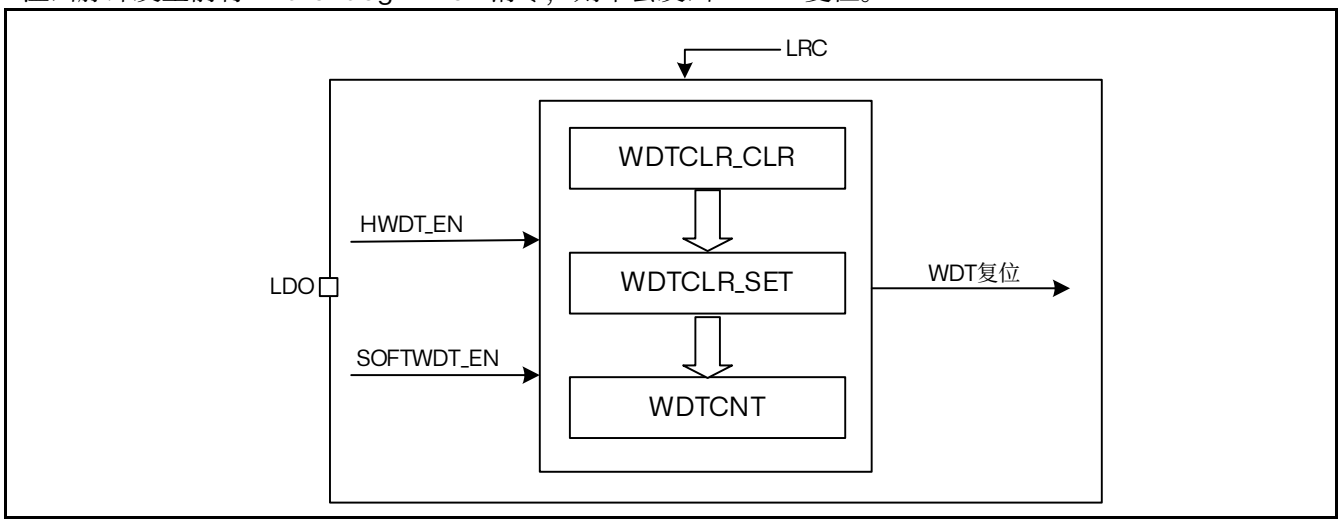



图 19.1 看门狗框图

## 19.4 工作模式

不同工作模式下，WDT 的控制状态也所有区别，具体参见下表：

| 工作模式 | SOFTWDT_EN | HWDT_EN | 功耗模式 | WDT 模块 |
|---|---|---|---|---|
| 调试模式 | 0 | X | Normal mode | 关闭 |
| | X | X | Sleep/Hold mode | 关闭 |
| | 1 | X | Normal mode | 开启，且可以随着 CPU 的停止而停止 |
| 非调试模式 | X | X | Normal mode | 开启 |
| | X | 0 | Sleep/Hold mode | 关闭 |
| | X | 1 | Sleep/Hold mode | 开启 |

注：1）X 表示无意义

2）SOFTWDT_EN 为 CLKCTRL1 寄存器的 bit14

3）HWDT_EN 由 Flash 0xFC2 地址内 WDT_EN[3:0]决定，具体参见 4.4Flash 控制功能

4）WDT 由 LRC 驱动，如果关闭了 LRC，则 WDT 也不会工作

## 19.5 看门狗寄存器

### 19.5.1 WDT 喂狗与时间配置寄存器（WDT_WDTCLR）

地址偏移量：0x04

复位值：0x0000 0040

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CLR[7:0] | | | | | | | | SET[7:0] | | | | | | | |
| w | w | w | w | w | w | w | w | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:8 **CLR[7:0]**：WDT 喂狗控制位

当该 8bit 写入 0xAA，则清狗，清除 WDT 内部计数器 WDTCNT，写入其他值无效

注：该高 8bit 只能写入，不能读取，读出值永远为 0

Bit 7:0 **SET[7:0]**：WDT 溢出时间设置：

WDT 溢出时间=64ms * (1+SET[7:0])

注：SET[7:0]为 8 位无符号数，由上面公式可以得出，最短的定时时间为 64ms，最长为 16384ms，默认为 4160ms。

### 19.5.2 WDT 计数寄存器（WDT_WDTCNT）

地址偏移量：0x08

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CNT[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **CNT[15:0]**：WDT 计数寄存器：

指示当前 WDT 内部的计数值。

# 20 I²C 接口

## 20.1 I²C 简介

I²C（芯片间）总线接口连接微控制器和串行 I²C 总线。它提供多主机功能，控制所有 I²C 总线特定的时序、协议、仲裁和定时。支持标准和快速两种模式。

根据特定设备的需要，可以使用 DMA 以减轻 CPU 的负担。

## 20.2 I²C 主要特点

- 并行总线/I²C 总线协议转换器
- 多主机功能：该模块既可做主设备也可做从设备
- I²C 主设备功能
  - 产生时钟
  - 产生起始和停止信号
- I²C 从设备功能
  - 可编程的 I²C 地址检测
  - 可响应 2 个从地址的双地址能力
  - 停止位检测
- 产生和检测 7 位位地址和广播呼叫
- 支持不同的通讯速度
  - 标准速度（高达 100kbit/s）
  - 快速（高达 400kbit/s）
- 状态标志：
  - 发送器/接收器模式标志
  - 字节发送结束标志
  - I²C 总线忙标志
- 1 个中断向量
  - 1 个中断用于地址/数据通讯成功

## 20.3 框图

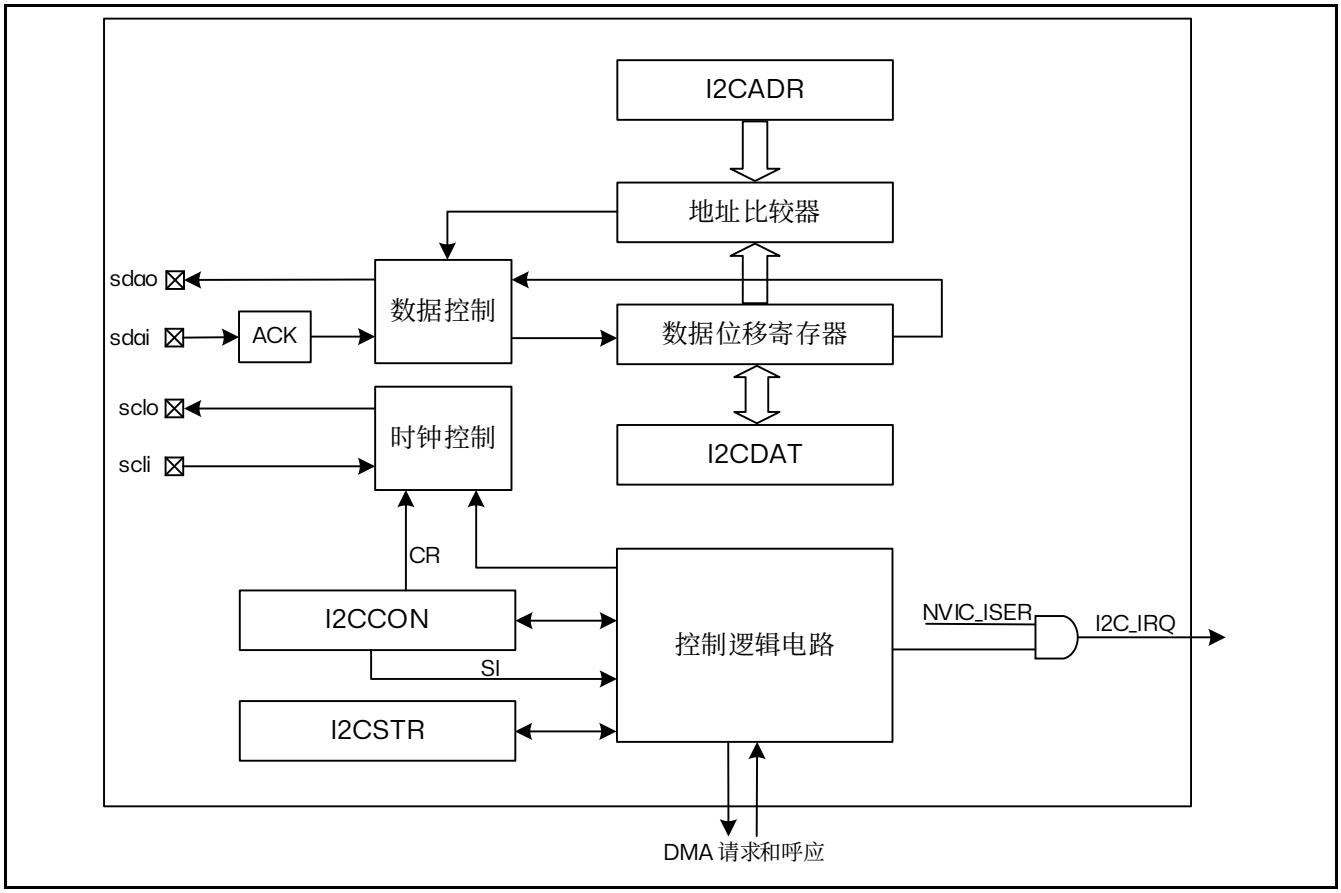



图 20.1 模块功能框图

## 20.4 I²C 功能描述

I²C 模块接收和发送数据，并将数据从串行转换成并行，或并行转换成串行。可以开启或禁止中断。接口通过数据引脚（SDA）和时钟引脚（SCL）连接到 I²C 总线。允许连接到标准（高达 100kHz）或快速（高达 400kHz）的 I²C 总线。

### 20.4.1 串行时钟生成

当 I2C 处于主机模式时，可编程的时钟发生器提供 SCL 时钟；当 I2C 处于从机模式时，时钟发生器被关闭，接收来自主机的时钟。时钟发生器的输出频率可以由寄存器 I2CCON 中的位 CR[7:0]控制,其中包含 I2CCON[0...1], I2CCON[8...14]。

### 20.4.2 中断生成

使能 ENS1，启动 I2C 模块，I2C 模块实时监测 I2C 总线状态，并根据用户设置对总线进行相应的操作及回应。当检测到总线有应用需求情况时，寄存器 I2CCON 中的标志位 SI 会被置位,并将当前应用状态写入状态寄存器 I2CSTA 中。若 I2C 中断使能打开，则产生 I2C 中断。I2C 产生中断时，寄存器 I2CCON 中的标志位 SI 会被置位。

### 20.4.3 传输模式

分别介绍 I2C 通讯的四种主要模式，并对所有可能的状态码进行了描述。下图中有如下缩写：

S：开始条件

Rs：重复开始条件

R：读控制位

W：写控制位

A：应答位

Ā：无应答位

DATA：8 位数据

P：终止条件

SLA：从机地址

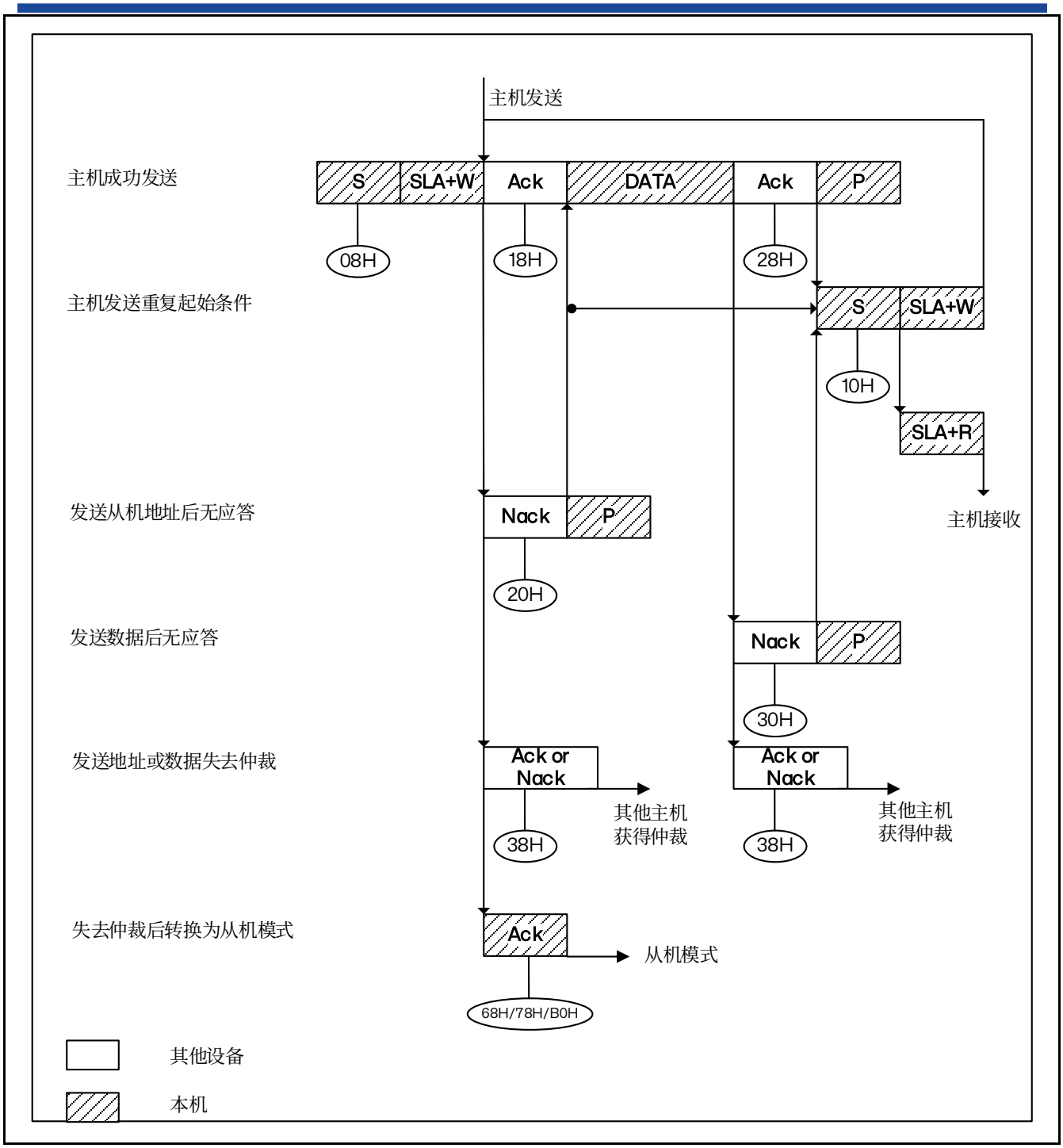
圆形用于表示中断标志已被置起。其中的数字表示当前状态寄存器 I2CSTA 中被掩去低三位的状态码。在 SI 被清除之前，I2C 通讯会暂停，应用软件必须决定是继续通讯还是终止当前传输。对每一个状态码，所需要的软件动作和随后的传输细节均有描述。

**I2C 主机发送模式：**

主机发送模式中，主机发送一系列数据到从机。一个开始条件(S)，随后一个从机地址(SLA)+写控制字(W),表示进入主机发送模式。

| 状态代码 | I2C 状态 | 应用程序配置 | | | | | I2C 硬件响应 |
|---|---|---|---|---|---|---|---|
| | | I2CDAT | I2CCON | | | | |
| | | | sta | sto | si | aa | |
| 08H | 起始条件已被发送 | 加载 SLA+W | X | 0 | 0 | X | SLA+W 将被发送<br>ACK 将被接收 |
| 10H | 重复起始条件已被发送 | 加载 SLA+W | X | 0 | 0 | X | 同上 |
| | | 加载 SLA+R | X | 0 | 0 | X | SLA+R 将被发送 I2C 将转换为“主接收器”模式 |
| 18H | SLA+W 已被发送 | 加载数据字节 | 0 | 0 | 0 | X | 数据字节将被发送；ACK 将被接收 |
| | ACK 已被接收 | 无动作 | 1 | 0 | 0 | X | 重复起始条件将被发送 |
| | | 无动作 | 0 | 1 | 0 | X | 终止条件将被发送；sto 标志将被复位 |
| | | 无动作 | 1 | 1 | 0 | X | 起始条件被发送后将再发送一个终止条件；sto 标志将被复位 |
| 20H | SLA+W 已被发送 | 加载数据字节 | 0 | 0 | 0 | X | 数据字节将被发送；ACK 将被接收 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | “not ACK” 已被接收 | 无动作 | 1 | 0 | 0 | X | 重复起始条件将被发送 |
| | | 无动作 | 0 | 1 | 0 | X | 终止条件将被发送；<br>sto 标志将被复位 |
| | | 无动作 | 1 | 1 | 0 | X | 起始条件被发送后将再发送<br>一个终止条件； sto 标志位将被复位 |
| 28H | i2cdat 的数据字节已被发送 | 加载数据字节 | 0 | 0 | 0 | X | 数据字节将被发送；<br>将发送 ACK 字节 |
| | ACK 已被接收 | 无动作 | 1 | 0 | 0 | X | 重复起始条件将被发送。 |
| | | 无动作 | 0 | 1 | 0 | X | 终止条件将被发送；<br>sto 标志将被复位 |
| | | 无动作 | 1 | 1 | 0 | X | 起始条件被发送后将再发送<br>一个终止条件； sto 标志将被复位 |
| 30H | i2cdat 的数据字节已被发送 | 数据字节 | 0 | 0 | 0 | X | 数据字节将被发送；<br>ACK 将被接收 |
| | “not ACK” 已被接收 | 无动作 | 1 | 0 | 0 | X | 重复起始条件将被发送； |
| | | 无动作 | 0 | 1 | 0 | X | 终止条件将被发送；<br>sto 标志将被复位 |
| | | 无动作 | 1 | 1 | 0 | X | 起始条件被发送后将再发送<br>一个终止条件； sto 标志将被复位 |
| 38H | SLA+R/W 或数据字节仲裁失败 | 无动作 | 0 | 0 | 0 | X | I2C 总线将被释放；<br>将进入“未寻址从机”状态； |
| | | 无动作 | 1 | 0 | 0 | X | 当总线空闲时将<br>发送一个起始条件 |

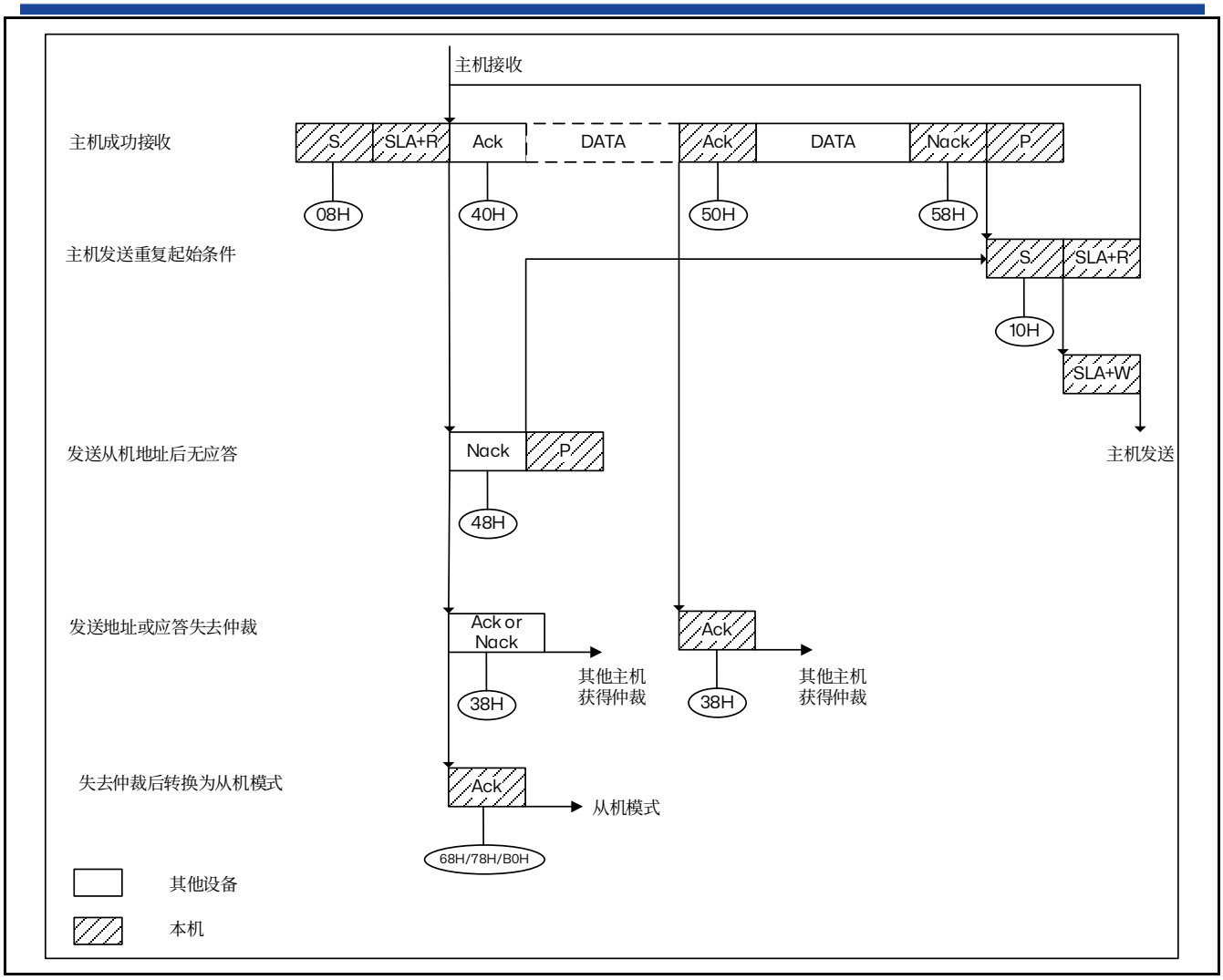



**I2C 主机接受模式：**

主机接收模式中，主机从从机接收一系列数据。一个开始条件（S），随后一个从机地址(SLA)+读控制字(R)表示进入主机接收模式。

| 状态代码 | I2C 状态 | 应用程序配置 | | | | | I2C 硬件响应 |
|---|---|---|---|---|---|---|---|
| | | I2CDAT | I2CCON | | | | |
| | | | sta | sto | si | aa | |
| 08H | 起始条件已被发送 | 加载 SLA+R | X | 0 | 0 | X | SLA+R 将被发送；ACK 将被接收 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 10H | 重复起始<br>条件已被发送 | 加载 SLA+R | X | 0 | 0 | X | 同上 |
| | | 或者加载<br>SLA+W | X | 0 | 0 | X | SLA+W 将被发送；<br>I2C 将转换为“主接收器”模式 |
| 38H | “not ACK”<br>位仲裁失败 | 无动作 | 0 | 0 | 0 | X | I2C 总线将被释放；<br>I2C 将会进入“从机”模式 |
| | | 或者无动作 | 1 | 0 | 0 | X | 当总线空闲时将发送一个起始条件 |
| 40H | SLA+R<br>已被发送 | 无动作 | 0 | 0 | 0 | 0 | 数据字节将被接收；<br>将返回“not ACK” |
| | ACK<br>已被接收 | 或者无动作 | 0 | 0 | 0 | 1 | 数据字节将被接收；<br>将返回“not ACK” |
| 48H | SLA+R<br>已被发送 | 无动作 | 1 | 0 | 0 | X | 重复起始条件将被发送 |
| | “not ACK”<br>已被接收 | 或无动作 | 0 | 1 | 0 | X | 终止条件将被发送；<br>sto 标志将被复位 |
| | | 或无动作 | 1 | 1 | 0 | X | 起始条件被发送后将再发送<br>一个终止条件；<br>sto 标志将被复位 |
| 50H | 数据字节<br>已被接收 | 读取数据字节 | 0 | 0 | 0 | 0 | 数据字节将被接收；<br>将返回“not ACK” |
| | 已返回 ACK | 或读取数据字节 | 0 | 0 | 0 | 1 | 数据字节将被接收；<br>将返回 ACK |
| 58H | 数据字节<br>将被接收 | 读取数据字节 | 1 | 0 | 0 | X | 重复起始条件将被发送 |
| | 已返回<br>“not ACK” | 或读取数据字节 | 0 | 1 | 0 | X | 终止条件将被发送；<br>sto 标志将被复位 |
| | | 或读取数据字节 | 1 | 1 | 0 | X | 起始条件被发送后将再发送<br>一个终止条件； sto 标志将被复位 |




**I2C 从机接受模式：**

从机接收模式中，从机从主机接收一系列数据。

进入从机模式前，需设置从机地址，I2CADR 中 I2CADR[7..1]位为从机地址。如果 I2CADR[0]置位，从机也将响应广播呼叫地址(00H)；否则将不响应广播呼叫地址。

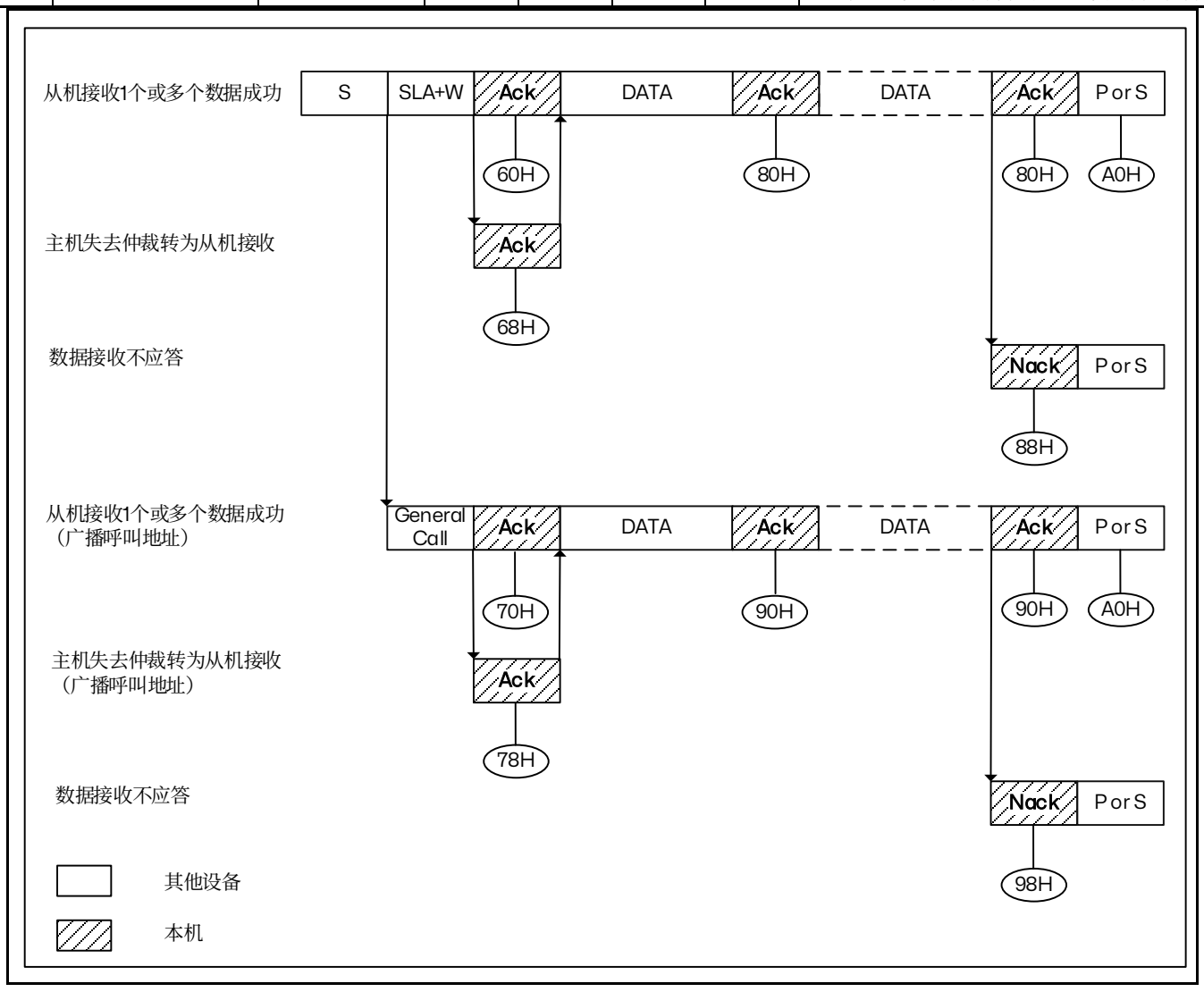
从机模式中，I2C 模块等待总线对本机地址或广播呼叫地址(如果 I2CADR[0]被置位)的寻址。如果读写数据位是'写'，则 I2C 进入从机接收模式，否则将进入从机发送模式。

地址和读写数据位接受完成后，中断标志(SI)置位，状态寄存器 I2CSTA 写入当前状态。

| 状态代码 | I2C 状态 | 应用程序配置 | | | | | I2C 硬件响应 |
|---|---|---|---|---|---|---|---|
| | | I2CDAT | I2CCON | | | | |
| | | | sta | sto | si | aa | |
| 60H | 自身的 SLA+W 已被接收 | 无动作 | 0 | 0 | 0 | 0 | 数据字节将被接收并返回“not ACK” |
| | 已返回 ACK | 或无动作 | 0 | 0 | 0 | 1 | 数据字节将被接收并返回 ACK |
| 68H | 主机 SLA+R/W 仲裁失败；自身 | 无动作 | X | 0 | 0 | 0 | 数据字节将被接收并返回“not ACK” |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | 的 SLA+W 已被接收，返回 ACK | 或无动作 | X | 0 | 0 | 1 | 数据字节将被接收<br>并返回 ACK |
| 70H | 呼叫地址（00H）已被接收；已返回 ACK | 无动作 | 0 | 0 | 0 | 0 | 数据字节将被接收<br>并返回“not ACK” |
| | | 或无动作 | 0 | 0 | 0 | 1 | 数据字节将被接收<br>并返回 ACK |
| 78H | 主机 SLA+R/W 仲裁失败；呼叫地址已被接收，返回 ACK | 无动作 | X | 0 | 0 | 0 | 数据字节将被接收<br>并返回“not ACK” |
| | | 或无动作 | X | 0 | 0 | 1 | 数据字节将被接收<br>并返回 ACK |
| 80H | 预先写入自身 SLV 地址；DATA 字节已被接收；返回 ACK | 读取数据字节 | 0 | 0 | 0 | 0 | 数据字节将被接收<br>并返回“not ACK” |
| | | 读取数据字节 | 0 | 0 | 0 | 1 | 数据字节将被接收<br>并返回 ACK |
| 88H | 预先写入自身 SLA；DATA 字节已被接收；返回“not ACK” | 读取数据字节 | 0 | 0 | 0 | 0 | 切换为“未寻址从机”模式；不识别自身从机地址或呼叫地址 |
| | | 读取数据字节 | 0 | 0 | 0 | 1 | 切换为“未寻址从机”模式；识别自身从机地址或呼叫地址 |
| | | 读取数据字节 | 1 | 0 | 0 | 0 | 切换为“未寻址从机”模式；<br>不识别自身从机地址或呼叫地址；<br>当总线空闲时将发送一个起始条件 |
| | | 读取数据字节 | 1 | 0 | 0 | 1 | 切换为“未寻址从机”模式；<br>识别自身从机地址或呼叫地址；<br>当总线空闲时将发送一个起始条件 |
| 90H | 预先写入呼叫地址；DATA 字节已被接收；返回 ACK | 读取数据字节 | 0 | 0 | 0 | 0 | 数据字节将被接收<br>并返回“not ACK” |
| | | 读取数据字节 | 0 | 0 | 0 | 1 | 数据字节将被接收<br>并返回 ACK |
| 98H | 预先写入呼叫地址；DATA 字节已被接收；返回“not ACK” | 读取数据字节 | 0 | 0 | 0 | 0 | 切换为“未寻址从机”模式；<br>不识别自身从机地址或呼叫地址 |
| | | 读取数据字节 | 0 | 0 | 0 | 1 | 切换为“未寻址从机”模式；<br>识别自身从机地址或呼叫地址 |
| | | 读取数据字节 | 1 | 0 | 0 | 0 | 切换为“未寻址从机”模式；<br>不识别自身从机地址或呼叫地址；<br>当总线空闲时将发送一个起始条件 |
| | | 读取数据字节 | 1 | 0 | 0 | 1 | 切换为“未寻址从机”模式；<br>识别自身从机地址或呼叫地址； |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | 当总线空闲时将发送一个起始条件 |
| A0H | 终止条件或重复起始条件在被配置为 SLV/REC 或 SLV/TRX 时被接收 | 无动作 | 0 | 0 | 0 | 0 | 切换为“未寻址从机”模式；<br>不识别自身从机地址或呼叫地址 |
| | | 无动作 | 0 | 0 | 0 | 1 | 切换为“未寻址从机”模式；<br>识别自身从机地址或呼叫地址 |
| | | 无动作 | 1 | 0 | 0 | 0 | 切换为“未寻址从机”模式；<br>不识别自身从机地址或呼叫地址；<br>当总线空闲时将发送一个起始条件 |
| | | 无动作 | 1 | 0 | 0 | 1 | 切换为“未寻址从机”模式；<br>识别自身从机地址或呼叫地址；<br>当总线空闲时将发送一个起始条件 |




**I2C 从机发送模式：**

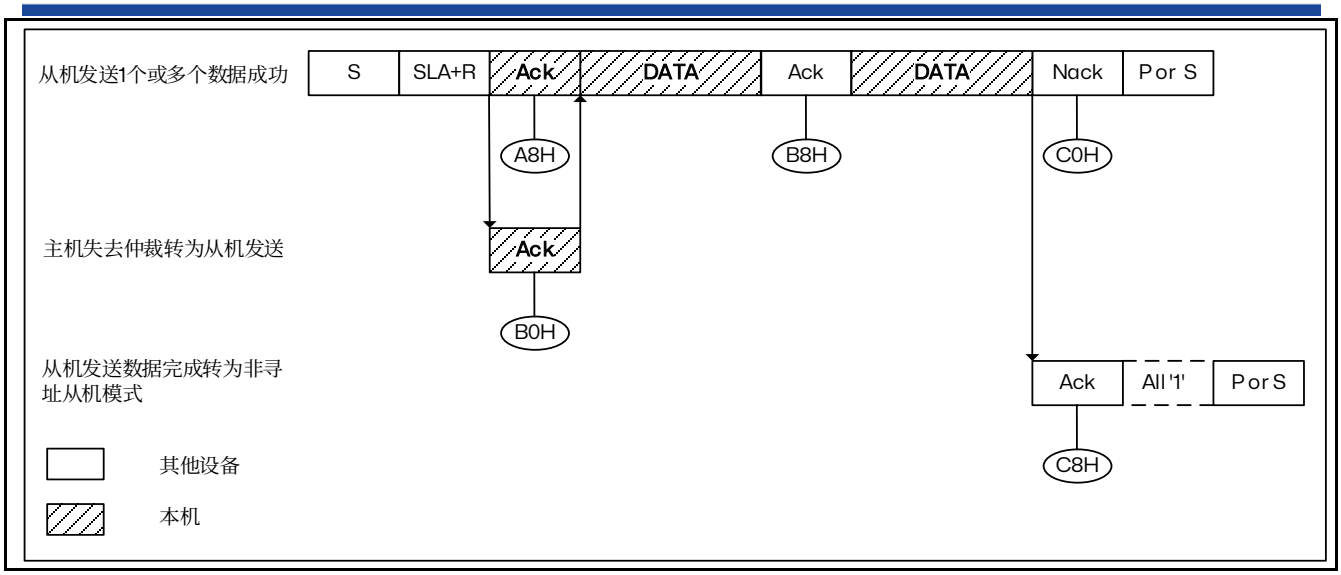
从机发送模式中，从机发送一系列数据到主机。

进入从机模式前，需设置从机地址，I2CADR 中 I2CADR[7..1]位为从机地址。如果 I2CADR[0]置位，从机也将响应广播呼叫地址(00H)；否则将不响应广播呼叫地址。

从机模式中，I2C 模块等待总线对本机地址或广播呼叫地址(如果 I2CADR[0]被置位)的寻址。如果读

写数据位是'写'，则 I2C 进入从机接收模式，否则将进入从机发送模式。地址和读写数据位接受完成后，中断标志(SI)置位，状态寄存器 I2CSTA 写入当前状态。

| 状态代码 | I2C 状态 | 应用程序配置 | | | | | I2C 硬件响应 |
|---|---|---|---|---|---|---|---|
| | | I2CDAT | I2CCON | | | | |
| | | | sta | sto | si | aa | |
| A8H | 自身 SLA+R 已被接收；返回 ACK | 加载数据字节 | 0 | 0 | 0 | 0 | 最后一个数据字节将被发送并接收 ACK |
| | | 或者加载数据字节 | 0 | 0 | 0 | 1 | 数据字节将被发送；ACK 将被接收 |
| B0H | 主机 SLA+R 仲裁失败；自身 SLA+R 已被接收；返回 ACK | 加载数据字节 | X | 0 | 0 | 0 | 最后一个数据字节将被发送并接收 ACK |
| | | 或者加载数据字节 | X | 0 | 0 | 1 | 数据字节将被发送；ACK 将被接收 |
| B8H | 数据字节已被发送；ACK 已被接收 | 加载数据字节 | 0 | 0 | 0 | 0 | 最后一个数据字节将被发送并接收 ACK |
| | | 或者加载数据字节 | 0 | 0 | 0 | 1 | 数据字节将被发送；ACK 将被接收 |
| C0H | 数据字节已被发送；"not ACK"已被接收 | 无动作 | 0 | 0 | 0 | 0 | 切换为"未寻址从机"模式；不识别自身从机地址或呼叫地址 |
| | | 无动作 | 0 | 0 | 0 | 1 | 切换为"未寻址从机"模式；识别自身从机地址或呼叫地址 |
| | | 无动作 | 1 | 0 | 0 | 0 | 切换为"未寻址从机"模式；不识别自身从机地址或呼叫地址；当总线空闲时将发送一个起始条件 |
| | | 无动作 | 1 | 0 | 0 | 1 | 切换为"未寻址从机"模式；识别自身从机地址或呼叫地址；当总线空闲时将发送一个起始条件 |
| C8H | 最后一个数据字节已被发送；ACK 已被接收 | 无动作 | 0 | 0 | 0 | 0 | 切换为"未寻址从机"模式；不识别自身从机地址或呼叫地址 |
| | | 无动作 | 0 | 0 | 0 | 1 | 切换为"未寻址从机"模式；识别自身从机地址或呼叫地址 |
| | | 无动作 | 1 | 0 | 0 | 0 | 切换为"未寻址从机"模式；不识别自身从机地址或呼叫地址；当总线空闲时将发送一个起始条件 |
| | | 无动作 | 1 | 0 | 0 | 1 | 切换为"未寻址从机"模式；识别自身从机地址或呼叫地址；当总线空闲时将发送一个起始条件 |




I2C 复合状态：

| 状态代码 | I2C 状态 | 应用程序配置 | | | | | I2C 硬件响应 |
|---|---|---|---|---|---|---|---|
| | | I2CDAT | I2CCON | | | | |
| | | | sta | sto | si | aa | |
| F8H | 没有可利用信息的相关状态；si=0 | 无动作 | 无动作 | | | | 等待或继续进行传递 |
| 00H | MST 或选择从机模式中的总线错误 | 无动作 | 0 | 1 | 0 | X | 只有当被配置为“主机”或“从机”模式时 I2C 硬件才会被触发<br>在所有情况下，总线将被释放并且 I2C 将切换到“未寻址从机”模式。sto 标志将被复位 |

### 20.4.4 模式选择

I2C 数据传输是以 8-bit 进行双向数据传输，标准模式下可达 100kbit/s 的传输速率，快速模式可达 400kbit/s 的速率。接口可以下述 4 种模式中的一种运行：

- 主机发送模式：串行数据通过 SDA 输出，串行时钟通过 SCL 输出
- 主机接收模式：串行数据通过 SDA 输入，串行时钟通过 SCL 输出
- 从机接收模式：串行数据通过 SDA 输入，串行时钟通过 SCL 输入
- 从机发送模式：串行数据通过 SDA 输出，串行时钟通过 SCL 输入

该模块默认地工作于从模式。接口在生成起始条件后自动地从从模式切换到主模式；当仲裁丢失或产生停止信号时，则从主模式切换到从模式。允许多主机功能。

**通信流**

主模式时，I²C 接口启动数据传输并产生时钟信号。串行数据传输总是以起始条件开始并以停止条件结束。起始条件和停止条件都是在主模式下由软件控制产生。

从模式时，I²C 接口能识别它自己的地址（7 位）和广播呼叫地址。软件能够控制开启或禁止广播呼叫地址的识别。

数据和地址按 8 位/字节进行传输，高位在前。跟在起始条件后的 1 个字节是地址（7 位模式为 1 个字节）。地址只在主模式发送。

在一个字节传输的 8 个时钟后的第 9 个时钟期间，接收器必须回送一个应答位（ACK）给发送器。参考下图。

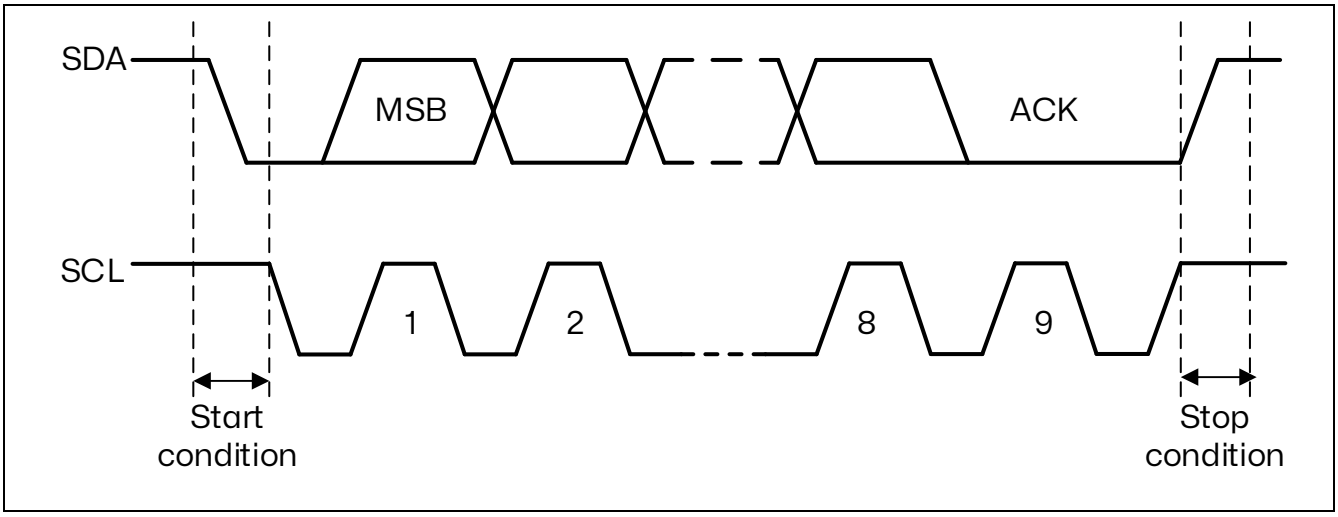



图 20.2 I²C 总线协议

## 20.5 I²C 寄存器

### 20.5.1 I2C 数据寄存器（I2C_I2CDAT）

地址偏移：0x00

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | I2CDAT[7:0] | | | | | | | |
| | | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:8 保留位，硬件强制为 0

Bit 7:0 **I2CDAT[7:0]：**寄存器 I2CDAT 是将要被传送到总线上的数据，或者是刚从总线上接收到的数据。寄存器 I2CDAT 没有设置影子寄存器，也没有双缓存，所以当 I2C 中断发生时，MCU 需要及时从它读取数据，以免数据丢失。

### 20.5.2 I2C 地址寄存器（I2C_I2CADR）

地址偏移：0x04

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | I2CADR[6:0] | | | | | | | I2CADR |
| | | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:8 保留位，硬件强制为 0

Bit 7:1 **I2CADR[6:0]：**I2C 从机地址(7 位)

Bit 0 **I2CADR：**I2CADR 呼叫地址确认位

当此位置 1 时，呼叫地址可以被识别，否则不能被识别。

### 20.5.3 I2C 控制寄存器（I2C_I2CCON）

复位地址偏移：0x08

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | CR[7:0] | | | | | | | | ENS1 | STA | STO | SI | AA | CR[1:0] | |
| | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15 保留位，硬件强制为 0

Bit 14:7 **CR[7:0]**：I2C 时钟频率控制位：I2C CLOCK=Fsys/(CR[14:7]+1)/4

Bit 6 **ENS1**：I2C 使能位

1：使能 IIC 模块

0：关闭 IIC 模块

Bit 5 **STA**：开始标志位

1：检查 IIC 总线的状态，如果空闲则生成开始信号

0：不会生成开始信号

Bit 4 **STO**：停止标志位

1：当处于主机模式，则向总线传输停止信号

0：不向总线传输停止信号

Bit 3 **SI**：中断标志位

当进入 25 种 IIC 状态之一时，SI 由硬件置位，唯一不置位的状态是“F8H”；写 0 清 0，写 1 无影响。

Bit 2 **AA**：生成应答标志位

1：应答在以下情况下被返回：接收到自身作为从机的地址；gc 被置位的情况下接收到地址呼叫；主机接收模式下一个字节接收完成；从机接收模式下一个字节接收完成

0：非应答在以下情况下被返回：主机接收模式下一个字节接收完成；从机接收模式下一个字节接收完成

Bit 1:0 **CR[1:0]**：I2C 时钟频率控制位：I2C CLOCK=Fsys/(CR[1:0]+1)/4

### 20.5.4 状态寄存器（I2C_I2CSTA）

地址偏移：0x0C

复位值：0x0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | I2CSTA[4:0] | | | | | Reserved | | |
| | | | | | | | | rw | rw | rw | rw | rw | | | |

Bit 15:8 保留位，硬件强制为 0

Bit 7:3 **I2CSTA[4:0]**：I2C 状态码

寄存器 I2CSTA 反映 I2C 模块的实时状态。这个寄存器的低三位始终为 0。总共有 26 种可能的状态。当进入 25 种状态的其中一种时，都会产生中断；唯一一种不产生中断的情况是状态 F8H。

*注：在上表中，“SLA”指从机地址，“R”指与从机地址一起传送的读/写位是读，“W”指与从机地址一起传 送的读/写位是写。*

# 21 通用异步收发器（UART）

## 21.1 UART 介绍

UART 也叫通用异步收发串口，数据通常以字符（或字节）为单位组成字符帧传送。字符帧由发送端逐帧发送，通过传输线被接收设备逐帧接收。发送端和接收端可以由各自的时钟来控制数据的发送和接收，两个时钟源彼此独立，互不同步。UART 串行通信模块可实现与外部设备的异步串行通信。

## 21.2 UART 主要特性

- 全双工通信口
- 共 1 路 UART
- 波特率可软件设置，最高波特率 115200
- 发送支持 1 个停止位或 2 个停止位
- 数据位宽支持 7 或 8 位
- 硬件自动完成奇偶校验，数据接收完成的同时判断并提示奇偶校验错误，给出标志。
- 接收/发送中断使能分别独立

## 21.3 UART 功能概述

接口通过三个引脚与其他设备连接在一起。任何 UART 双向通信至少需要两个脚：接收数据输入（RX）和发送数据输出（TX）。

RX：接收数据串行输。通过过采样技术来区别数据和噪音，从而恢复数据。

TX：发送数据输出。当发送器被禁止时，输出引脚恢复到它的 I/O 端口配置。当发送器被激活，并且不发送数据时，TX 引脚处于高电平。

- 总线在发送或接收前应处于空闲状态
- 一个起始位
- 一个数据字（7 或 8 位），最低有效位在前
- 1 或 2 个的停止位，由此表明数据帧的结束
- 一个状态寄存器（UART_UARTSTA）
- 一个串口数据缓冲寄存器（UART_SBUF）
- 一个串口波特率发生寄存器（UART_SREL），最高波特率为 115200

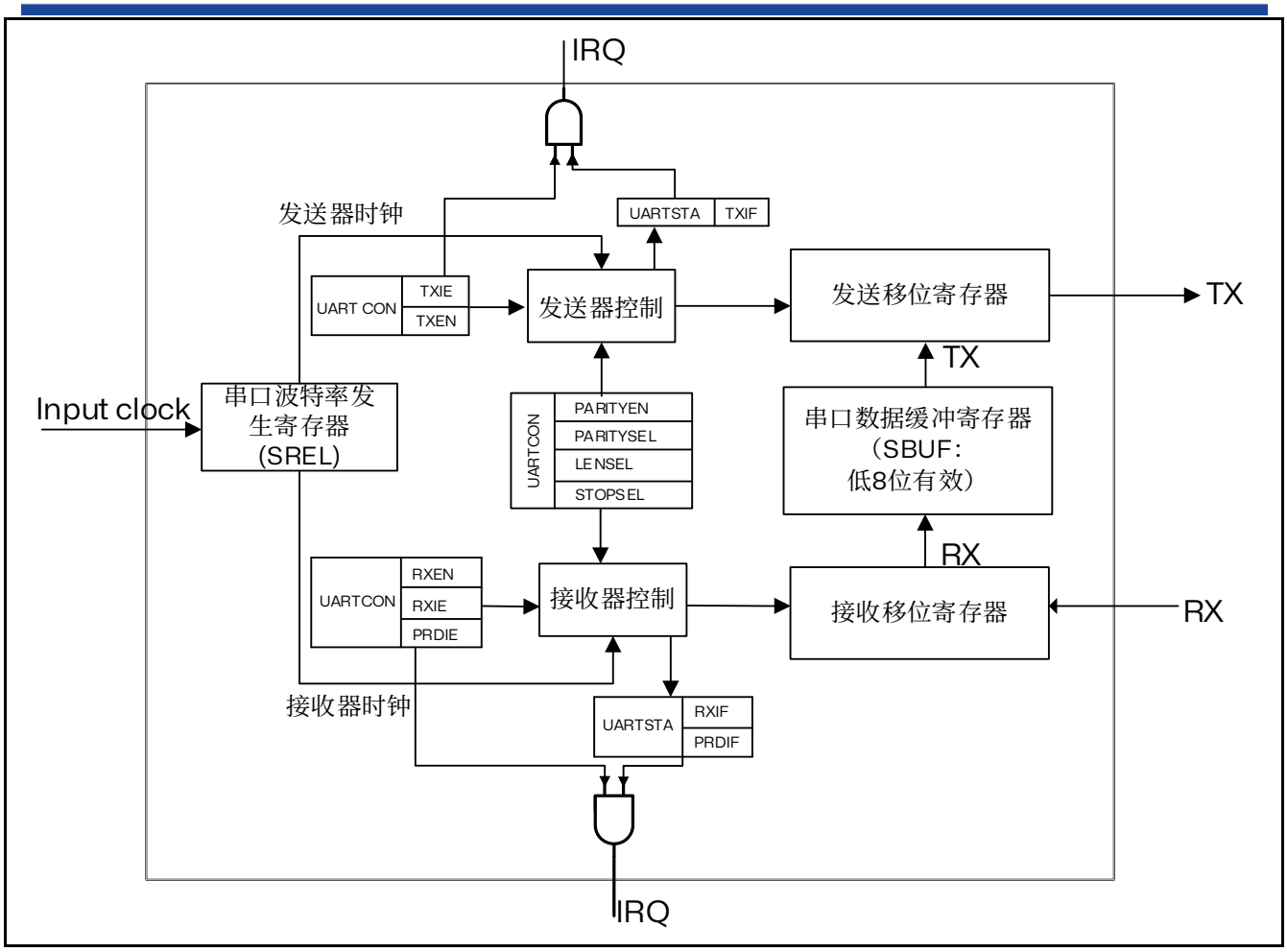



图 21.1 UART 框图

### 21.3.1 UART 特性描述

字长可以通过编程 UART_UARTCON 寄存器中的 LENSEL 位，选择成 7 或 8 位。在起始位期间，TX 脚处于低电平，在停止位期间处于高电平。

空闲符号被视为完全由‘1’组成的一个完整的数据帧，后面跟着包含了数据的下一帧的开始位（‘1’的位数也包括了停止位的位数）。

断开符号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，发送器再插入 1 或 2 个停止位（‘1’）来应答起始位。

发送和接收由一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位时，分别为其产生时钟。

随后将有每个功能块的详细说明。

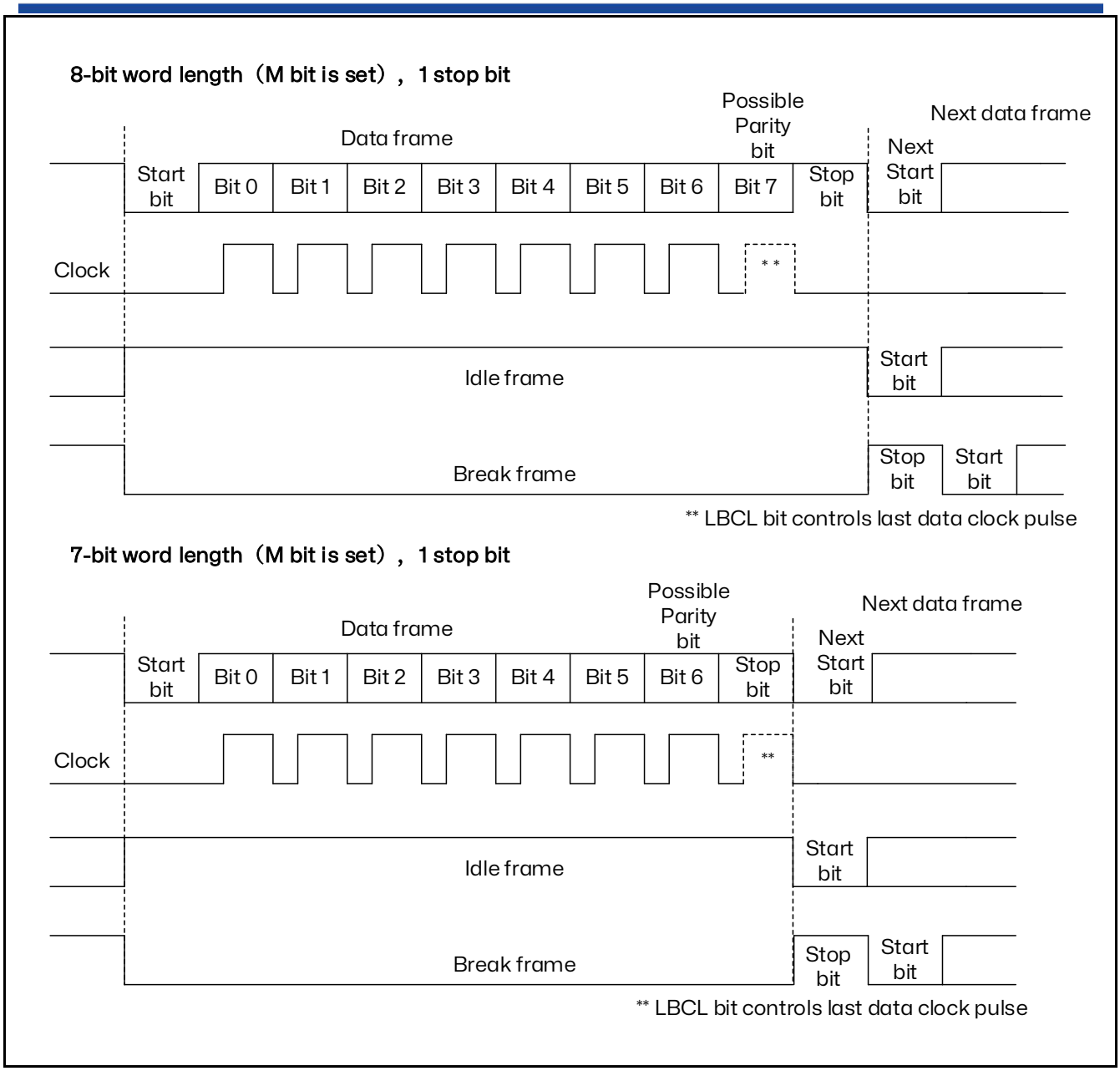



图 21.2 字长设置

### 21.3.2 发送器

发送器根据 LENSEL 位的状态发送 7 位或 8 位的数据字。当发送使能位（TXEN）被设置时，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 UART 发送期间，在 TX 引脚上首先移出数据的最低有效位。在此模式里，UART_SBUF 寄存器包含了一个内部总线和发送移位寄存器之间的缓冲器。

每个字符之前都有一个低电平的起始位；之后跟着的停止位，其数目可配置。

UART 支持多种停止位的配置：1 或 2 个停止位。

*注：1. 在数据传输期间不能复位* TXEN *位，否则将破坏* TXEN *脚上的数据，因为波特率计数器停止*

*计数。正在传输的当前数据将丢失。*

2. TXEN *位被激活后将发送一个空闲帧。*

**可配置的停止位**

随每个字符发送的停止位的位数可以通过 UART 功能配置寄存器的位 8 进行编程。

1. 1 个停止位：停止位位数的默认值。
2. 2 个停止位：可用于常规 UART 模式、单线模式以及调制解调器模式。

### 21.3.3 接收器

UART 可以根据 UART_UARTCON 的 LENSEL 位接收 7 位或 8 位的数据字。

**字符接收**

在 UART 接收期间，数据的最低有效位首先从 RX 脚移进。在此模式里，UART_SBUF 寄存器包含的缓冲器位于内部总线和接收移位寄存器之间。

配置步骤：

1. 通过在 UART_UARTCON 寄存器来激活 UART。
2. 编程 UART_UARTCON 的 LENSEL 位定义字长
3. 在 UART_UARTCON 中编写停止位的个数
4. 利用波特率寄存器 UART_SREL 选择希望的波特率。
5. 设置 UART_UARTCON 的 RXEN 位。

*注意：在接收数据时* RXEN *位不应被复位。如果* RXEN *位在接收时被清零，当前字节接收被丢失。*

**断开符号**

当接收到一个断开帧时，UART 像处理帧错误一样处理它。

**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 UART 功能配置寄存器（UART_UARTCON）的控制位来配置，在正常模式时，可以是 1 或 2 个。

1. 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。
2. 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，帧错误标志将被设置。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。

### 21.3.4 校验控制

设置 UART_UARTCON 寄存器上的 PARITYEN 位，可以使能奇偶控制（发送时生成一个奇偶位，接收时进行奇偶校验）。根据 LENSEL 位定义的帧长度，可能的 UART 帧格式列在下表中。

表 21.1 帧格式

| LENSEL 位 | PARITYEN 位 | UART 帧 |
|---|---|---|
| 0 | 0 | 丨起始位 丨8 位数据 丨 停止位 丨 |
| 0 | 1 | 丨起始位 丨7 位数据 丨 奇偶检验位 丨 停止位 丨 |
| 1 | 0 | 丨起始位 丨7 位数据 丨 停止位 丨 |
| 1 | 1 | 丨起始位 丨6 位数据 丨 奇偶检验位 丨 停止位 丨 |

*注意：在用地址标记唤醒设备时，地址的匹配只考虑到数据的*MSB *位，而不用关心校验位。(*MSB *是数 据位中最后发出的，后面紧跟校验位或者停止位)*

**偶校验：**校验位使得一帧中的 7 或 8 个 LSB 数据以及校验位中‘1’的个数为偶数。例如：数据=00110101，有 4 个‘1’，如果选择偶校验（在 UART_UARTCON 中的 PARITYSEL＝10），校验位将是‘0’。

**奇校验：**此校验位使得一帧中的 7 或 8 个 LSB 数据以及校验位中‘1’的个数为奇数。例如：数据=00110101，有 4 个‘1’，如果选择奇校验（在 UART_UARTCON 中的 PARITYSEL＝01），校验位将是‘1’。

**传输模式：**如果 UART_UARTCON 的 PARITYEN 位被置位，写进数据寄存器的数据的 MSB 位被校验位替换后发送出去（如果选择偶校验偶数个‘1’，如果选择奇校验奇数个‘1’）。如果奇偶校验失败，UART_UARTSTA 寄存器中的 PARITY 标志被置‘1’，并且如果 UART_UARTCON 寄存器的 PRDIE 在被预先设置的话，中断产生。

## 21.4 串口通讯模式说明

### 21.4.1 方式 1

方式 1 是一种标准的异步通信方式，每帧包含 9 或 10 位数据信息：1 位起始位（0），7 位数据位（低位在前），1 或 2 位停止位（1）。在这种方式中，TXD 引脚为数据发送端，RXD 引脚为数据接收端，其波形如下图所示：

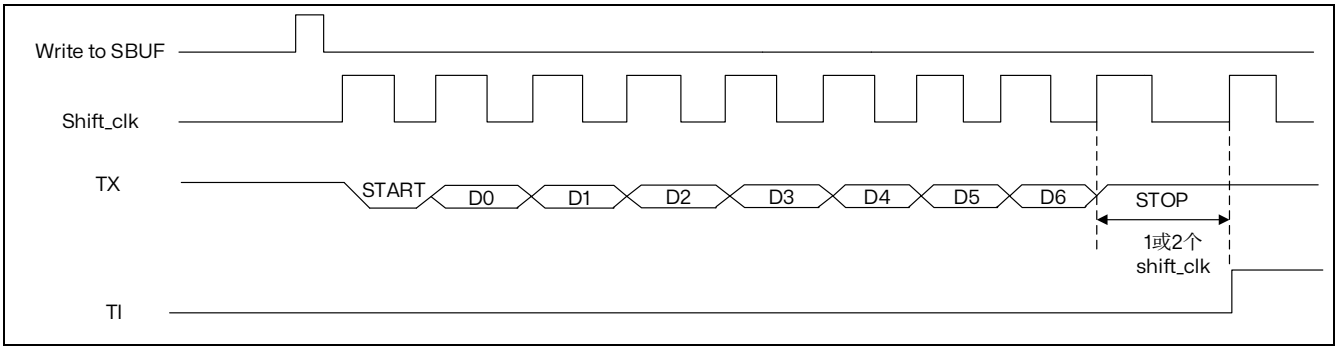



图 21.3 方式 1 时串行发送数据时序

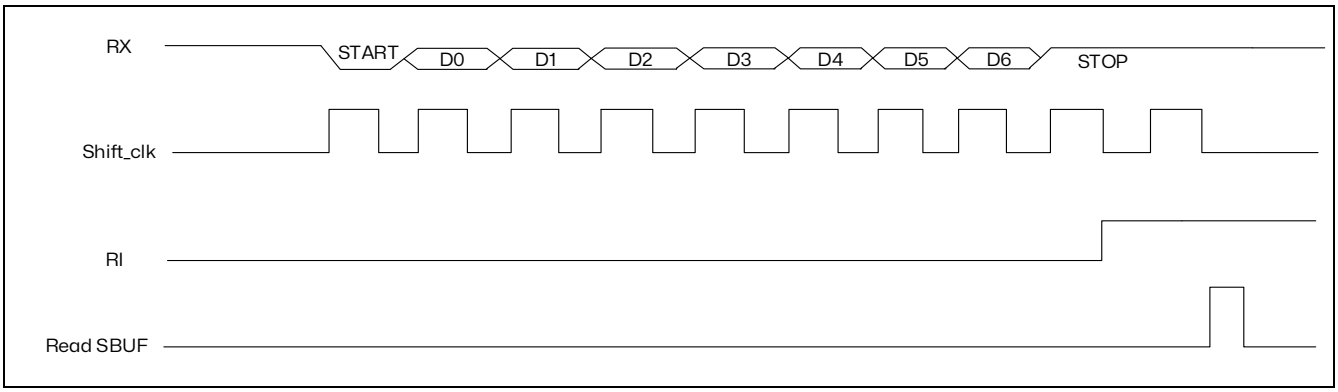



图 21.4 方式 1 时串行接收数据时序

在方式 1 中，发送状态时，当一帧中最后一个数据发送完时，发送中断标志 TI 置“1”；接收状态时，接收完最后一个数据位时，接收中断标志 RI 置 1。

### 21.4.2 方式 2

方式 2 是每帧包含 10 或 11 位数据信息：1 位起始位（0），7 位数据位（低位在前），1 位奇偶校验数据位，1 或 2 位停止位（1）。TXD 引脚为数据发送端，RXD 引脚为数据接收端，其波形如下图所示：

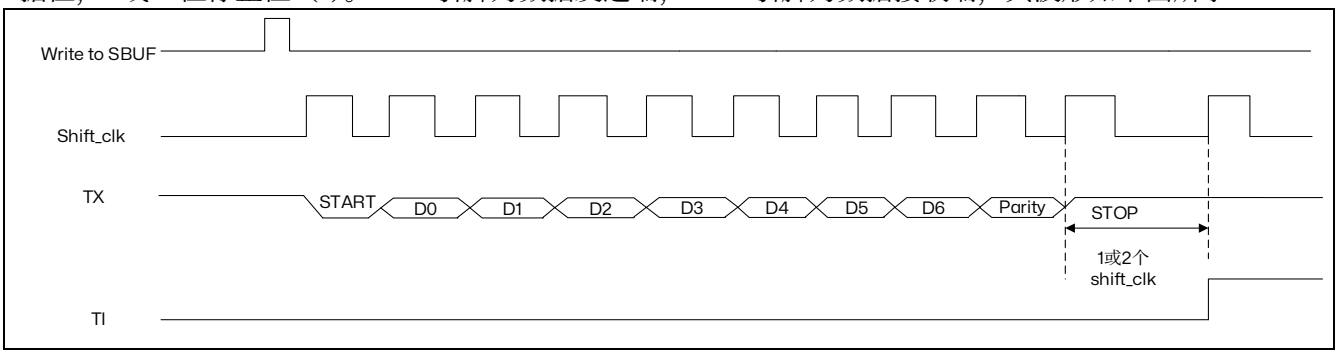



图 21.5 方式 2 时串行发送数据时序

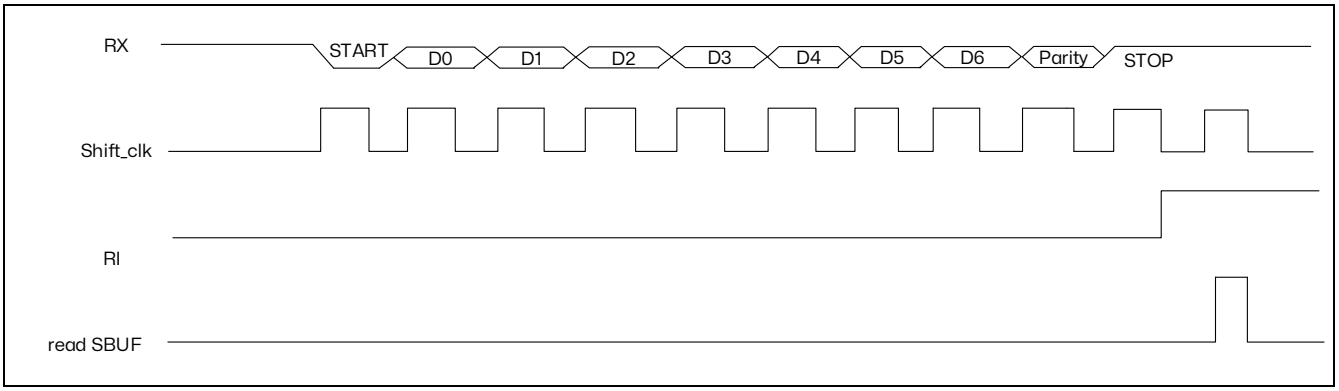



图 21.6 方式 2 时串行接收数据时序

### 21.4.3 方式 3

方式 3 是一种标准的异步通信方式，每帧包含 10 或 11 位数据信息：1 位起始位（0），8 位数据位（低位在前），1 或 2 位停止位（1）。在这种方式中，TXD 引脚为数据发送端，RXD 引脚为数据接收端，其波形如下图所示：

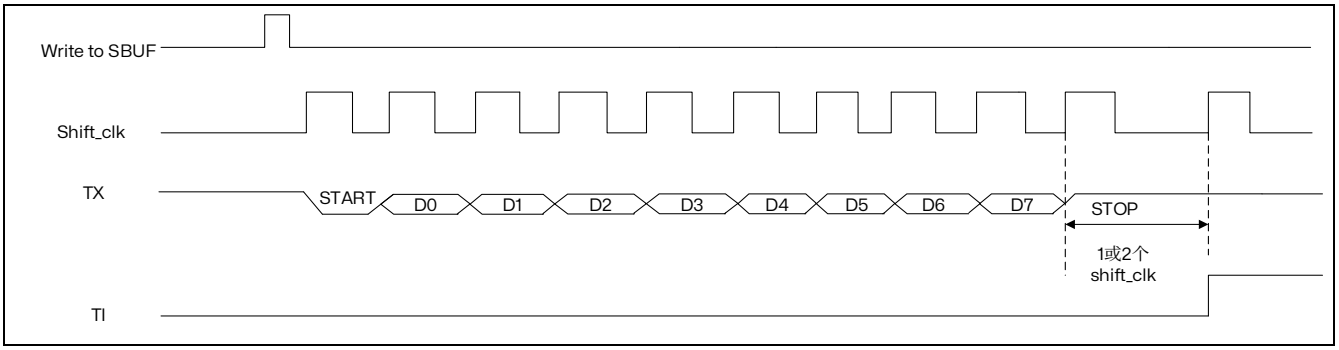



图 21.7 方式 3 时串行发送数据时序

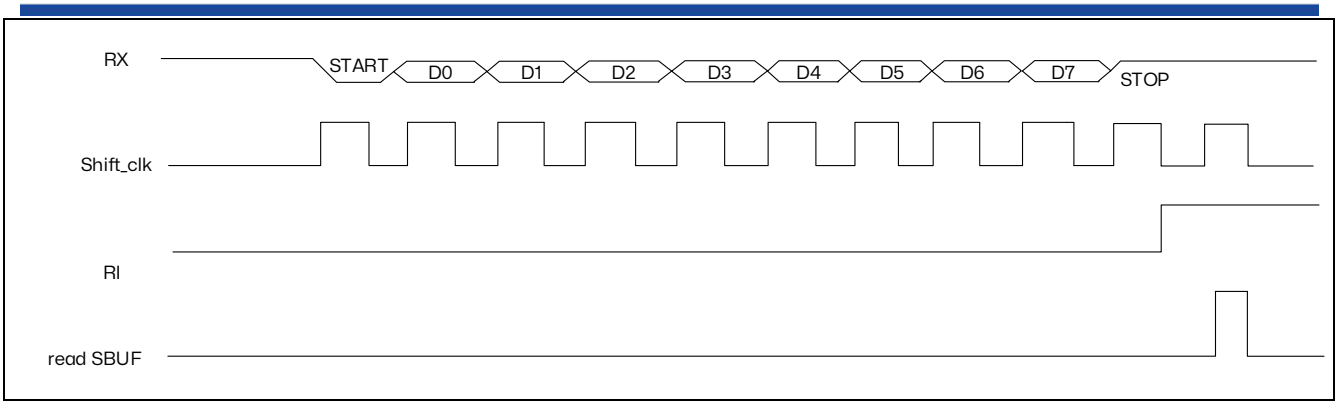



图 21.8 方式 3 时串行接收数据时序

### 21.4.4 方式 4

方式 4 是使用第 9 位数据的通信方式，每帧包含 11 或 12 位数据信息：1 位起始位（0），8 位数据位（低位在前），1 个奇偶校验或自定义数据位，1 或 2 位停止位（1）。TXD 引脚为数据发送端，RXD 引脚为数据接收端，其波形如下图所示：

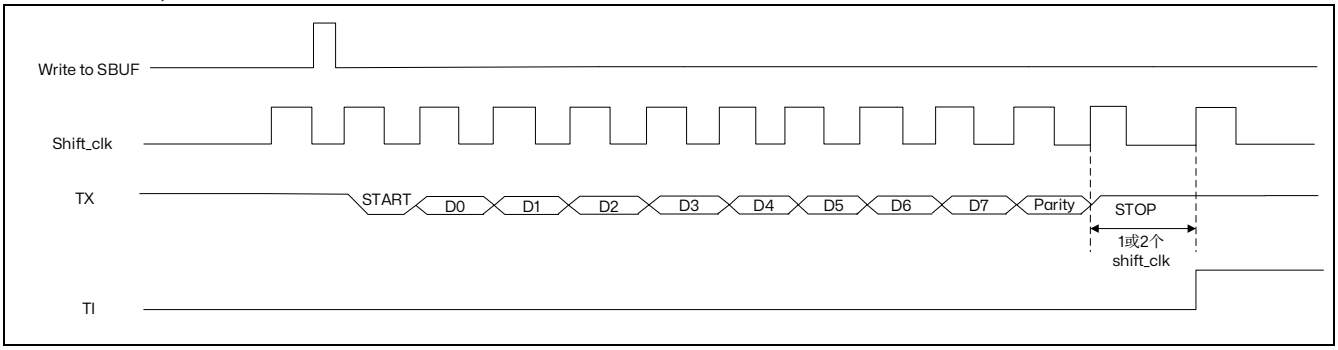



图 21.9 方式 4 时串行发送数据时序

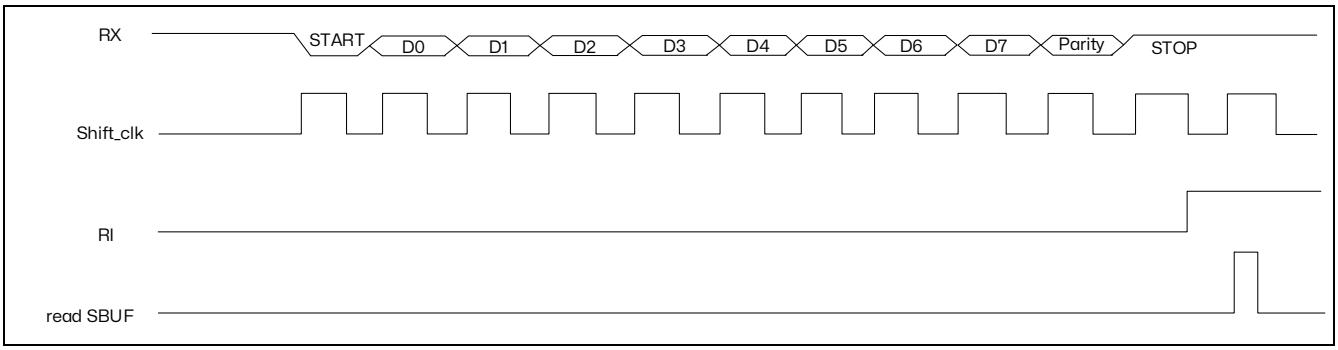



图 21.10 方式 4 时串行接收数据时序

## 21.5 UART 中断请求

表 21.2 UART 中断请求

| 中断 | 中断标志 | 使能位 |
|---|---|---|
| 串口上溢中断 | PRDIF | PRDIE |
| UART 接收中断 | RXIF | RXIE |
| UART 发送中断 | TXIF | TXIE |

## 21.6 UART 模式配置

表 21.3 UART 模式设置 (1)

| UART 模式 | UART1 |
|---|---|
| 异步模式 | √ |
| 全双工 | √ |

(1) √ = 支持, × = 不支持该应用

## 21.7 UART 寄存器

可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。

### 21.7.1 UART 功能配置寄存器（UART_UARTCON）

地址偏移：0x04

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PRDIE | Reserved | | | | | | STOPSEL | LENSEL | PARITYSEL[1:0] | | PARITYEN | RXIE | TXIE | RXEN | TXEN |
| rw | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15 **PRDIE**：串口上溢中断使能控制位：

0：上溢中断禁止

1：上溢中断使能

Bit 14:9 保留位，硬件强制为 0

Bit 8 **STOPSEL**：UART 通讯停止位长度选择位

0：1bit

1：2bit

Bit 7 **LENSEL**：UART 通讯数据长度选择位

0：8bit

1：7bit

Bit 6:5 **PARITYSEL[1:0]**：UART 奇偶校验选择位

00：固定为 0

01：奇校验

10：偶校验

11：固定为 1

Bit 4 **PARITYEN**：UART 奇偶校验使能

0：禁止

1：使能

Bit 3 **RXIE**：UART 接收中断使能位

0：禁止

1：使能

Bit 2 **TXIE**：UART 发送中断使能位

0：禁止

1：使能

Bit 1 **RXEN**：UART 接收使能位

0：禁止

1：使能

Bit 0 **TXEN**：UART 发送使能位

0：禁止

1：使能

### 21.7.2 串口波特率发生寄存器（UART_SREL）

地址偏移：0x0C

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SREL[15:0] | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:0 **SREL[15:0]**：串口波特率发生寄存器，是一个 16 位的波特率分频系数，其值可为 0~65535 之间的任一整数，最高波特率为 115200。波特率计算公式：

$$\text{波特率}=\frac{Fsys}{2\times(SREL+1)}$$

### 21.7.3 串口数据缓冲寄存器（UART_SBUF）

地址偏移：0x10

复位值：0x0000 0000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | SBUF[7:0] | | | | | | | |
| | | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw |

Bit 15:8 保留位，硬件强制为 0

Bit 7 **STOPSEL**：低 8 位有效，对寄存器 SBUF 写操作，则串口将开始向外传输发送缓存数据；对寄存器 SBUF 读操作，则串口将从串行接收缓存中读取数据。

### 21.7.4 UART 状态寄存器（UART_UARTSTA）

地址偏移：0x14

复位值：0x0000 000

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | PRDIF | PARITY | RXIF | TXIF |
| | | | | | | | | | | | | rw | rw | rw | rw |

Bit 15:4 保留位，硬件强制为 0

Bit 3 **PRDIF**：接收上溢中断标志

0：未产生上溢

1：产生上溢，也就是外部输入到 SBUF 中的数据，还未被及时读出，然后又有新的数据传输过来把老的数据覆盖了。

*注：写 0 清零，写 1 无效。*

Bit 2 **PARITY**：接收时奇偶校验的状态

0：正确

1：错误

*注：写 0 清零，写 1 无效。*

Bit 1 **RXIF**：接收中断标志

0：接收数据还未完成

1：接收数据完成，可从寄存器 SBUF 中读出

*注：写 0 清零，写 1 无效。*

Bit 1 **TXIF**：发送中断标志

0：发送未完成

1：发送完成

*注：写 0 清零，写 1 无效。*

# 22 版本历史

表 22.1 版本历史

| 日期 | 版本 | 更改内容 |
|---|---|---|
| 2023/9/13 | V2.0 | 新版 |
| 2023/11/08 | V2.1 | 1. 修改了 DMA 部分格式<br>2. HSI 修正为 HRC，LSI 修正为 LRC<br>3. 例程库 V1.5 低功耗模式对应此新版参考手册，旧版例程库 V1.4 及以前对应旧版参考手册<br>4. 更新了图 5.1 电源框图 |
| 2024/02/28 | V2.2 | 1. 修正了图 12.39，12.43 中部分笔误<br>2. 修正了比较器章节 15.4 和 15.9 中部分笔误<br>3. 修正了比较器章节 15.10 中的寄存器名称<br>4. 修正了 ADC 章节 10.6.12 中的寄存器名称<br>5. 修正了 SADCCON 寄存器的错误 |