CPS32K11x

ARM® Cortex®-M0+ 32 位微控制器

用户参考手册

Chipsine Semiconductor

Rev0.8　2023.12

目录

1 文档约定

1.1 概述

CPS32K11x 是一款高可靠、超低功耗、车规级 32 位 MCU，符合 AEC-Q100 Grade1 标准，支持功能安全特性，可广泛应用在汽车引擎控制(电机、传感器)、暖通空调(HVAC)、车身控制(车灯、车窗、雨刷、尾门、后视镜等)、BMS、网关等领域。

CPS32K11x 采用 ARM Cortex-M0+内核，运行频率最高可达 48MHz，高达 128KB 的片内 FLASH 存储器与 20KB 的 SRAM 存储器，嵌入式 MATH 协处理器可加速电机 FOC 算法运行，大幅提升 CPU 算力。

芯片功能安全特性全面升级，FLASH 与 SRAM 都支持 ECC，支持外部晶振停振监测与系统时钟动态切换，支持 WWDG 与 IWDG，支持 11 阶阈值可配 LVD。芯片集成高速 CAN-FD、LIN、SPI*2、UART*2、LPUART、I2C*2、1-Wire 等通信外设，可满足各类车载通信应用需求。

芯片封装覆盖 LQFP80，LQFP64，LQFP48，LQFP32，QFN32。

宽工作温度范围-40~+125℃，宽工作电压 2.7~5.5V。本产品配合成熟的 KEIL & IAR 调试开发软件，支持 C 语言及汇编语言在线快速开发与调试。

1.2 寄存器相关缩写词列表

英文缩写	含义	详细描述
R/W	读/写	软件可以读写该位
RO	只读	软件只能读取该位
WO	只写	软件只能写入该位。读该位将只返回复位值
RCW0	读取/写入 0 清零	软件可以读取该位，可以通过写 0 清除该位，写 1 没有影响。
RCW1	读取/写入 1 清零	软件可以读取该位，可以通过写 1 清除该位，写 0 没有影响。
RCW	读取/写入清零	软件可以读取该位，也可以通过写入寄存器将该位清零。写入该位对其值没有影响。
RCR	读取/读取清零	软件可以读取该位，读取该位时自动清零。写入该位对其值没有影响。
RW0	读/仅可写入一次	软件仅可以写入一次该位，但可以随时读取该位。只能通过复位将该位返回到复位值。
ROW1	只读，写触发	软件可以读取该位。写入 1 时，将触发事件，但不会影响该位的值。
Res	保留	保留位，必须保持复位值。

1.3 词汇表

常见词汇缩写	英文全拼	释义
AHB	Advanced High Performance Bus	高级高性能总线
APB	Advanced Peripheral Bus	高级外设总线
CKGEN	Clock Generator	时钟产生器
ECC	Error Checking and Correction	错误检测以及纠正
HIRC	High Speed Internal Clock	高速内部振荡时钟
HXT	High speed External crystal oscillator	高速外部晶振时钟
IAP	In-Application Programming	在应用中编程
ISP	In-System Programming	在系统编程
LRU	Least Recently Used	最少最近使用
LIRC	Low Speed Internal Clock	低速内部时钟
LXT	Low speed External crystal oscillator	低速外部晶振时钟
LSB	Least Significant Bit	最低有效位
LVD	Low Voltage Detect	低电压检测
MSB	Most Significant Bit	最高有效位
NMI	Non Maskable Interrupt	不可屏蔽中断
OSC	Oscillator	振荡器
PLL	Phase Locked Loop	锁相环
POR	Power On Reset	上电复位
PMU	Power Manager Unit	系统电源管理
SRAM	Static Random-Access Memory	静态随机存取存储器
VCO	Voltage Controlled Oscillator	压控振荡器
XOSC	External Crystal Oscillator	晶体振荡器

2 产品特性

- 车规标准
  - 通过AEC-Q100检测认证，符合Grade1, ASIL-B标准
- 工作电压
  - 电压范围VDD=AVDD=2.7~5.5V
  - 环境温度Ta=-40～＋125℃
- 性能
  - 高达48MHz的ARM® Cortex- M0+内核
  - 单周期32位乘法器
  - 32位协处理器(除法器/三角函数)
- 存储器和存储器接口
  - 最高128KB的片内FLASH，支持ECC,支持1KB的cache和prefetch
  - 最高20KB的SRAM，支持ECC
- 时钟
  - 高速振荡器 (HXT) – 支持8MHz到32MHz 石英晶体振荡器；可选择低功耗或高增益振荡器
  - 低速振荡器 (LXT) – 32.768KHz 石英晶体振荡器；可选择低功耗或高增益振荡器
  - 内部高速时钟源 (HIRC) –8 MHz预校准内部基准时钟源，全温度范围误差<±2.0%@VCC=2.7~5.5V
  - 内部低速时钟源 (LIRC) –38.4KHz预校准内部基准时钟源，全温度范围误差<±3.0@VCC=2.7~5.5V
  - PLL – 输入频率4~30MHz，输出频率5~48MHz,输出倍频可以配置，锁定时间最大为50μS
  - 支持外部时钟输入，1~32MHz
- 系统外设
  - 两种低功耗模式:Sleep, DeepSleep
  - Power On Reset
  - 低压检测复位电路(LVD)
  - 带独立时钟源的看门狗(IWDG)
  - 串行线调试(SWD)接口
- 人机接口
  - 多达71个通用输入输出接口(GPIO)
  - 所有端口都是5V-IO
  - 每组GPIO都支持外部中断(IRQ)
  - 不可屏蔽中断外部输入(NMI)
- 模拟模块
  - 一个多达18个外部输入通道，12bit高速SAR ADC，最大转换速率1 MSPS。
  - 支持内部的(2V/4V)参考电压AVDD/ExtVref
  - 内部的参考电压支持全温度范围±4%的精度。
  - 温度传感器精度±5.0℃
  - 电压比较器1ch，支持轨到轨,包含6位DAC和可编程参考输入的模拟比较器(VC)
  - 一个可编程放大器PGA(放大倍率X1~X50)
- 定时器
  - 2个16位高级控制定时器(TIM1/TIM2)
  - 2个16/32位基础定时器/计数器(TIM10/TIM11)
  - 1个16位低功耗定时器(LPTIM)
  - 1个实时时钟(RTC)
  - 1个可编程计数器阵列(PCA)
  - 1个窗口看门狗(WWDG)
- 通信接口
  - 1个CAN-FD模块，兼容 CAN2.0B
  - 2个标准UART模块
  - 1个LPUART模块
  - 2个硬件LIN模块
  - 2个标准SPI模块
  - 2个标准I2C模块
  - 1个1-Wire通信
- 硬件CRC模块
- 16字节UID号
- 封装选项
  - 80引脚LQFP
  - 64引脚LQFP
  - 48引脚LQFP
  - 32引脚LQFP
  - 32引脚QFN

3 产品功能概述

3.1 32 位 Cortex®-M0+内核

ARM® Cortex®-M0+处理器是一款嵌入式 32 位 RISC 处理器,该处理器引脚数少、功耗低，能够提供满足 MCU 实现需要的低成本平台。同时提供卓越的计算性能和先进的中断系统响应。Cortex®-M0+处理器全面支持 KEIL & IAR 调试器，包含了一个硬件调试电路，支持 2 线式的 SWD 调试接口。

表 3-1 提供了本产品 Cortex®-M0+所支持的特性。

表 3-1 CPS32K11x 的 CPU 所支持的特性

MPU	支持，8 段保护区域
单周期乘法	支持
除法	支持
中断数	32 个
调试接口	SWD-DP
大小端	小端
SysTick Timer	支持

3.2 存储器(Memory)

3.2.1 FLASH 存储器

嵌入式闪存存储器，用于存放程序和数据。内建全集成 FLASH 控制器，无需外部高压输入，由全内置电路产生高压来编程，支持 ISP、IAP 功能。

- CPS32K112 系列最大支持 32KB 字节 flash，支持 ECC 功能(1bit 出错纠正，2bit 出错检测)
- CPS32K114 系列最大支持 64KB 字节 flash，支持 ECC 功能(1bit 出错纠正，2bit 出错检测)
- CPS32K116 系列最大支持 128KB 字节 flash，支持 ECC 功能(1bit 出错纠正，2bit 出错检测)

3.2.2 SRAM 存储器

内置低功耗 SRAM，用于存放程序和数据。在不同的低功耗模式下，SRAM 数据会被全部保留。该 SRAM 支持 ECC 校验算法(1bit 出错纠正，2bit 出错检测)。

- CPS32K112 系列最大支持 12KB 字节 SRAM，支持 ECC 功能(1bit 出错纠正，2bit 出错检测)
- CPS32K114 系列最大支持 16KB 字节 SRAM，支持 ECC 功能(1bit 出错纠正，2bit 出错检测)
- CPS32K116 系列最大支持 20KB 字节 SRAM，支持 ECC 功能(1bit 出错纠正，2bit 出错检测)

3.3 工作模式

该产品支持 3 种工作模式：

- 运行模式 Active：CPU 运行，周边功能模块运行。
- 休眠模式 Sleep：CPU 停止运行，周边功能模块运行。
- 深度休眠模式 DeepSleep：CPU 停止运行，高速时钟停止运行，低速时钟运行，仅需要低速时钟或者无需时钟就可以运行的模块可以继续运行。

3.4 时钟系统

1 个频率为 8MHz 的高精度内部时钟 HIRC，从 DeepSleep 模式到工作模式的唤醒时间为 5us，全电压全温度范围内的频率偏差<±2%，无需外接昂贵的高速晶体。

1 个锁相环 PLL，支持 2/3/4/5/6/7/8 等不同系数倍频，参考时钟源可选择 HIRC 或者 HXT。

1 个频率为 8~32MHz 的高速外部晶振 HXT。

1 个频率为 32.768KHz 的低速外部晶振 LXT。

1 个频率为 38.4/32.768KHz 的低速内部时钟 LIRC。

1 个时钟校准与监测模块，可以用来校准内部 HIRC/LIRC 时钟，或者监测外部晶振时钟是否正常。

3.5 复位控制器

该产品具有 9 个复位信号来源，每个复位信号可以让 CPU 重新运行，绝大多数寄存器会被重新复位，程序计数器 PC 会被复位指向复位地址。

表 3-2 中断源

编号	中断源
1	上电复位(POR)
2	外部 RSTN 端子复位
3	IWDG 复位
4	WWDG 复位
5	ARM 系统软件复位
6	低电压(LVD)复位
7	LOCKUP 复位
8	寄存器 CPURST 复位
9	寄存器 MCURST 复位

3.6 通用 IO 端口

最多可提供 71 个 GPIO 端口(80pin 封装)，其中部分 GPIO 与模拟端口复用。每个端口由独立的控制寄存器位来控制。支持边沿触发中断和电平触发中断，可从各种功耗模式下把 MCU 唤醒到工作模式。支持 Push-Pull CMOS 推挽输出、Open-Drain 开漏输出。内置上拉电阻、下拉电阻，带有施密特触发器输入滤波功能。输出驱动能力可配置，最大支持 20mA 的电流驱动能力。

支持从 NMI 端子输入，产生 NMI 异常。

3.7 定时器和看门狗

该产品包含 2 个高级控制定时器(TIM1/2)，1 个可编程计数器器阵列(PCA)，2 个基础定时器(TIM10/11)，1 个低功耗基本定时器(LPTIM)，1 个系统窗口看门狗定时器(WWDG)，1 个独立看门狗定时器(IWDG)，1 个系统嘀嗒定时器 1 个 RTC 定时器，1 个自动唤醒定时器(AWK)。

3.8 通信接口

该产品包含一个 CAN 通信模块，支持 CAN FD 和 CAN 2.0B 协议，2 个硬件 LIN 通信模块，2 个 UART 通信模块，2 个 SPI 通信模块，2 个 I2C 通信模块，1 个 1-Wire 通信模块。

3.9 蜂鸣器

该产品包含 1 个蜂鸣器模块。可以直接分频 LIRC/HXT/PCLK 时钟，来产生各种频率的输出至端口。

3.10 模拟功能

该产品包含一路高速 12bit SAR-ADC，转换速率可以达到 1Msps。

支持多大 18 路的外部通道输入+4 路内部电压监测。

一个低电压检测器(LVD),一个温度传感器(TSN)，一个电压比较器(VC)，一个可编程运算放大器(PGA)。

3.11 系统功能

该产品支持，128-bit 的唯一 ID 号(UID)，一个硬件 CRC 演算电路，一个 MATH 协处理器。

4 系统和存储器概要

4.1 系统架构图

主要的系统构成：

- 1 个 AHB 总线系统 Master：
  - Cortex®-M0+内核
- 7 个 AHB 总线 Slave
  - 内部 SRAM
  - 内部 FLASH
  - AHB to APB Bridge，包含所有 APB 接口的外设
  - GPIO 接口
  - MATH 模块
  - RCC 模块
  - CRC 等 AHB 接口模块

系统的模块框图如图 4-1 系统框图所示：

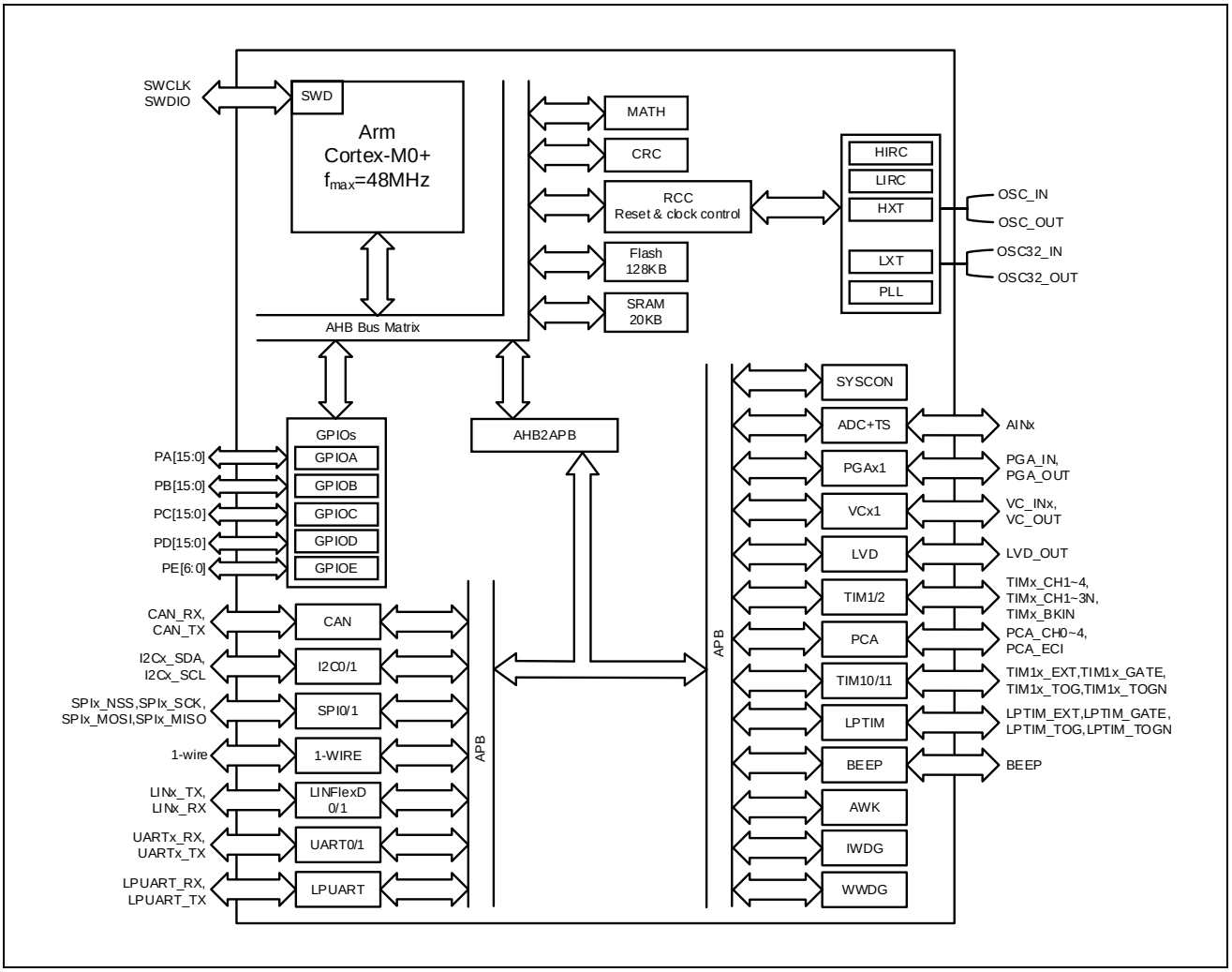


图 4-1 系统框图

4.2 存储器映射

系统的地址空间总共有 4GB，包含程序存储空间，数据存储空间，周边模块寄存器，I/O 端口等。数据使用小端点格式，就是数据的高字节保存在内存的高地址中，而数据的低字节保存在内存的低地址中。整个系统地址空间的划分如下图 4-2 存储器映射所示：

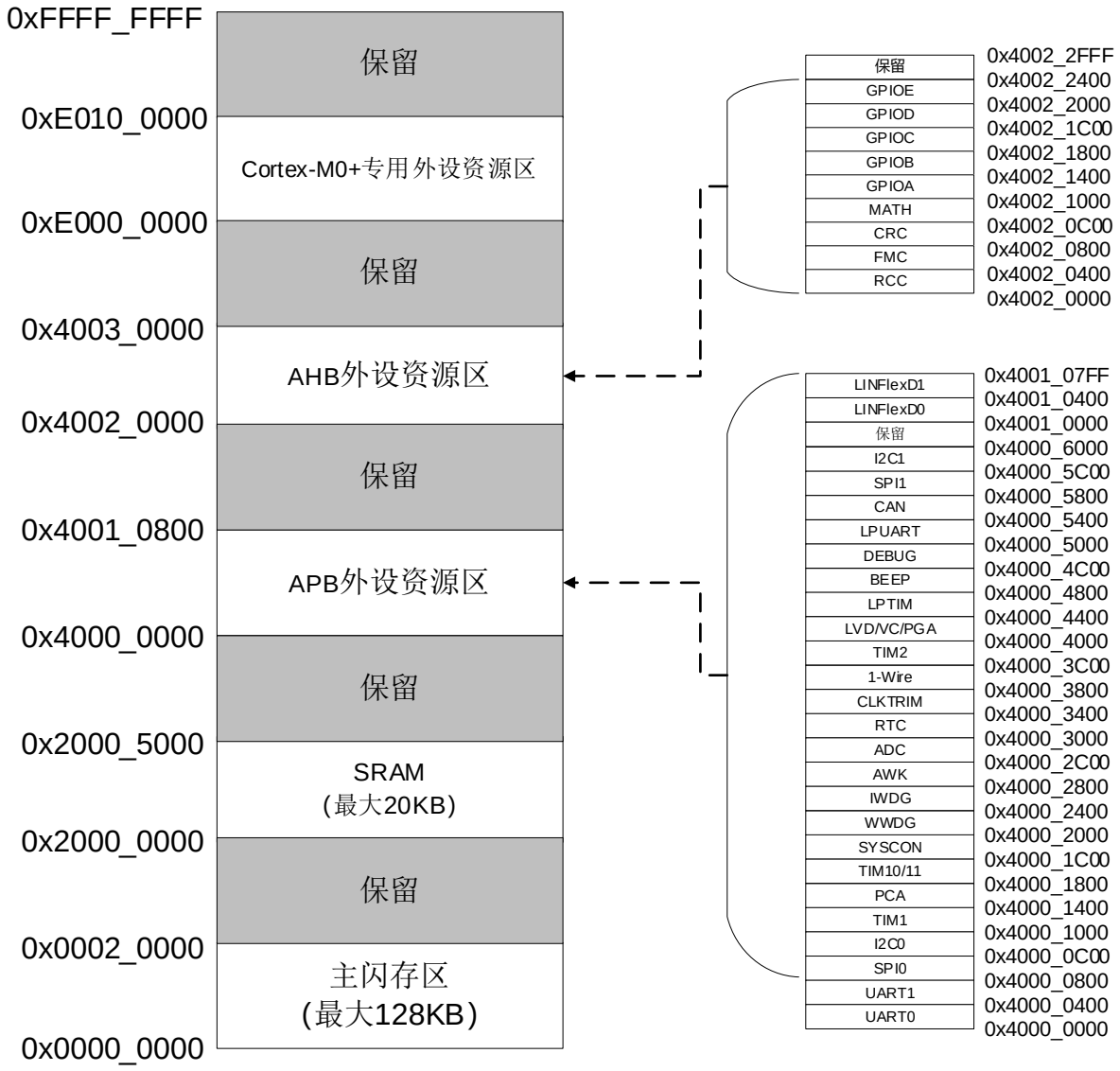


图 4-2 存储器映射

4.3 存储空间和模块地址

下面**错误!未找到引用源。**给出了 CPS32K11x 器件内部包含的各模块的地址空间和边界信息。

表 4-1 CPS32K11x 存储器映射和外设寄存器编址

总线	边界地址	空间大小(Bytes)	模块
	0xE000_0000 - 0xE00F_FFFF	1MB	Coretex-M0+ peripheral
	0x4003_0000 - 0xDFFF_FFFF		保留
AHB	0x4002_2400 - 0x4002_2FFF	1KB	保留
	0x4002_2000 - 0x4002_23FF	1KB	GPIOE
	0x4002_1C00 - 0x4002_1FFF	1KB	GPIOD
	0x4002_1800 - 0x4002_1BFF	1KB	GPIOC
	0x4002_1400 - 0x4002_17FF	1KB	GPIOB
	0x4002_1000 - 0x4002_13FF	1KB	GPIOA
	0x4002_0C00 - 0x4002_0FFF	1KB	MATH
	0x4002_0800 - 0x4002_0BFF	1KB	CRC16
	0x4002_0400 - 0x4002_07FF	1KB	FLASH
	0x4002_0000 - 0x4002_03FF	1KB	RCC
	0x4001_0800 - 0x4001_FFFF		保留
APB2	0x4001_0400 - 0x4001_07FF	1KB	LINFlexD1
	0x4001_0000 - 0x4001_03FF	1KB	LINFlexD0
	0x4000_6000 - 0x4000_FFFF		保留
APB1	0x4000_5C00 - 0x4000_5FFF	1KB	I2C1
	0x4000_5800 - 0x4000_5BFF	1KB	SPI1
	0x4000_5400 - 0x4000_57FF	1KB	CAN
	0x4000_5000 - 0x4000_53FF	1KB	LPUART
	0x4000_4C00 - 0x4000_4FFF	1KB	DEBUG
	0x4000_4800 - 0x4000_4BFF	1KB	BEEP
	0x4000_4400 - 0x4000_47FF	1KB	LPTIM
	0x4000_4000 - 0x4000_43FF	1KB	LVD/VC/OPA
	0x4000_3C00 - 0x4000_3FFF	1KB	TIM2
	0x4000_3800 - 0x4000_3BFF	1KB	1-Wire
	0x4000_3400 - 0x4000_37FF	1KB	CLKTRIM
	0x4000_3000 - 0x4000_33FF	1KB	RTC
	0x4000_2C00 - 0x4000_2FFF	1KB	ADC
	0x4000_2800 - 0x4000_2BFF	1KB	AWK
	0x4000_2400 - 0x4000_27FF	1KB	IWDT
	0x4000_2000 - 0x4000_23FF	1KB	WWDG
	0x4000_1C00 - 0x4000_1FFF	1KB	SYSCON
	0x4000_1800 - 0x4000_1BFF	1KB	TIM10/TIM11
	0x4000_1400 - 0x4000_17FF	1KB	PCA
	0x4000_1000 - 0x4000_13FF	1KB	TIM1
	0x4000_0C00 - 0x4000_0FFF	1KB	I2C0
	0x4000_0800 - 0x4000_0BFF	1KB	SPI0
	0x4000_0400 - 0x4000_07FF	1KB	UART1
	0x4000_0000 - 0x4000_03FF	1KB	UART0
AHB	0x2000_5000 - 0x3FFF_FFFF		保留
	0x2000_0000 - 0x2000_4FFF	20KB	SRAM
	0x1FFF_FA00–0x1FFF_FFFF		保留
	0x1FFF_F800–0x1FFF_F9FF	0.5KB	FLASH 用户选项字节
	0x0002_0000 - 0x1FFF_F7FF		保留
	0x0000_0000 - 0x0001_FFFF	128KB	FLASH Main Array

4.4 内嵌 SRAM

CPS32K11x 内置多达 20KB 字节的静态 SRAM，它可以以字节(8 位)、半字(16 位)或字(32 位)进行访问。该产品 SRAM 全领域支持 ECC 校验算法。

4.5 FLASH 存储器简介

FLASH 存储器有两个不同存储区域：

- 主存储块，它包括应用程序和用户数据区
- 信息块，选项字节(Option bytes)－内含硬件及存储保护用户配置选项。

FLASH 存储器接口基于 AHB 协议执行指令和数据存取。

该产品 FLASH 全领域支持 ECC 校验算法。

5 FLASH 控制器(FMC)

5.1 FMC 的简介

该产品支持最大 128KB 容量的 FLASH 存储器。共划分为 256 个页(Sector)，每个页(Sector)的容量为 512Byte。

本模块(FMC)支持对 FLASH 存储器的擦除、编程以及读取操作。此外，本模块支持对 FLASH 控制寄存器的读出/写入保护。

5.2 特性

- FLASH 存储器
- 最大支持:主存储区(128KB)
- 擦除寿命:≧100,000 次擦除操作
- 数据保存:100 年@25℃
- 读取速度:0wait@FCLK≦18MHz 1wait@18MHz＜FCLK≦36MHz 2wait@36MHz＜FCLK≦48MHz

- FLASH 控制器
- 擦除:页擦除，整片擦除，选项字节擦除
- 写:只支持 32bit 的写，写地址必须字(Word)对齐
- 读:可以按照 8bit/16bit/32bit 位宽的读取
- 选项字节加载
- 读出/写入保护
- SEC-DED ECC 校验算法，即 1bit 出错自动纠错，2bit 同时出错报警功能
- 1bit 自动纠错时，可选中断响应，2bit 同时出错时，系统 NMI 异常响应。
- 预取与 Cache 功能支持，可以显著提高 FLASH 在 wait 的情况下的 CPU 效率

5.3 FLASH 容量划分

表 5-1 片上 FLASH 容量划分

区域	序号	地址范围	容量(字节)
主存储区	页(Sector)0	0x0000 0000 – 0x0000 01FF	512Byte
	页(Sector)1	0x0000 0200 – 0x0000 03FF	512Byte
	页(Sector)2	0x0000 0400 – 0x0000 05FF	512Byte
	页(Sector)3	0x0000 0600 – 0x0000 07FF	512Byte
	…	…	…
	页(Sector)255	0x0001 FE00 – 0x0001 FFFF	512Byte
选项字节区	用户选项字节区	0x1FFF F800 – 0x1FFF F82F	64Byte
		0x1FFF F830 – 0x1FFF F9FF	448Byte(保留)

5.4 功能描述

本控制器支持对 FLASH 只支持 Word(32bits)位宽写操作(不支持按 8 位宽，16 位宽写操作)，支持对 FLASH 的 Byte(8bits)、Half-word(16bits)、Word(32bits)三种位宽读操作。需要注意的是，Byte 操作的地址必须按 Byte 对齐，Half-word 操作的目标地址必须按 Half-word 对齐(地址最低位为 0)，Word 操作的目标地址必须按 Word 对齐(地址最低两位为 0)。如果读写操作的地址没有按照位宽规定对齐，该操作无效，并且系统会进入 Hard fault 出错中断。

5.4.1 键值

键值是用来解除某些寄存器的通用键值，本产品有以下键值：

KEY1 = 0x45670123

KEY2 = 0xCDEF89AB

5.4.2 读操作

内置闪存模块可以在通用地址空间直接寻址，任何 32 位数据的读操作都能访问闪存模块的内容并得到相应的数据。

预取缓冲区

本产品包含 1KB 的 cache,在 cache 有效的情况下，如 cache 被 hit 到，那么 CPU 就从 cache 里去取值，从而实现 0 等待周期的访问，如果 cache 的访问没有被 hit 到，那么 CPU 只能从 flash 里去取值，需要根据 latency 的设定值来决定访问周期。

访问潜伏期

为了保护对 FLASH 的正确读取，必须在 FLASH 访问控制寄存器中的 LATENCY 中指定预取指控制器的速度比，这个数值等于每次访问 FLASH 后到下次访问之间所需插入的等待周期的个数。复位后，这个值默认零，也就是没有插入等待周期的状态。

5.4.3 写与擦除操作

嵌入式闪存支持在系统编程 ISP 以及在应用编程 IAP。

ISP 是在芯片安装到用户应用板上后，使用 SWD 协议或系统加载程序(ISP 程序)对 FLASH 编程，将用户代码烧录到 MCU 中。ISP 提供了一种简单高效的方法，免除了烧写芯片时的芯片装夹等问题。

与 ISP 方法不同的是，IAP(在应用编程)能够使用 MCU 支持的任何通信接口(UART，I2C，SPI 等)下载程序或者数据。IAP 允许用户在运行程序的过程中重写应用程序，前提是一部分应用程序必须预先用 ISP 的方法烧写进去。

烧写和擦除操作在整个产品工作电压范围内都可以完成。该操作有下列寄存器完成：

- 关键字寄存器(FLASH_KEYR)
- 选项字节关键字寄存器(FLASH_OPTKEYR)
- 闪存控制寄存器(FLASH_CR)
- 闪存状态寄存器(FLASH_SR)
- 闪存地址寄存器(FLASH_AR)

- 选项字节寄存器(FLASH_OBR)
- 写保护寄存器(FLASH_WRPR1)
- 写保护寄存器(FLASH_WRPR2)
- 写保护寄存器(FLASH_WRPR3)
- 写保护寄存器(FLASH_WRPR4)

只要 CPU 不去访问 FLASH 空间，进行中的 FLASH 写操作不会妨碍 CPU 的运行。也就是说，在对 FLASH 进行写/擦除操作的同时，任何对 FLASH 的访问都会令总线停顿，直到写/擦除操作完成后才会继续执行，这意味着在写/擦除 FLASH 的同时不可以对它取指和访问数据。在对 FLASH 空间做写/擦除操作时，内部 RC 振荡器(HIRC)必须处于开启状态。

5.4.3.1 解锁闪存

FPEC 说明：FLASH 编程和擦除控制器

复位后，FPEC 模块是被保护的，FLASH_CR 寄存器不允许被改写，这样可以防范意外的擦除动作。通过写入特定的序列到 FLASH_KEYR 寄存器可以打开 FPEC 模块，这个特定的序列是在 FLASH_KEYR 写入两个键值(KEY1 和 KEY2，见 5.4.1 节)；错误的操作序列都会锁死 FPEC 模块和 FLASH_CR 寄存器直至下次复位。

写入错误的键序列还会产生总线错误；总线错误发生在第一次写入的不是 KEY1，或第一次写入的是 KEY1 但第二次写入的不是 KEY2。

5.4.3.2 编程主闪存

对主闪存编程每次可以写入 32 位。当 FLASH_CR 寄存器的 PG 位为'1'时，在一个闪存地址写入一个字将启动一次编程；写入任何非一个字的数据，FPEC 都会产生总线错误。在编程过程中(BSY 位为'1')，任何读写闪存的操作都会使 CPU 暂停，直到此次闪存编程结束。

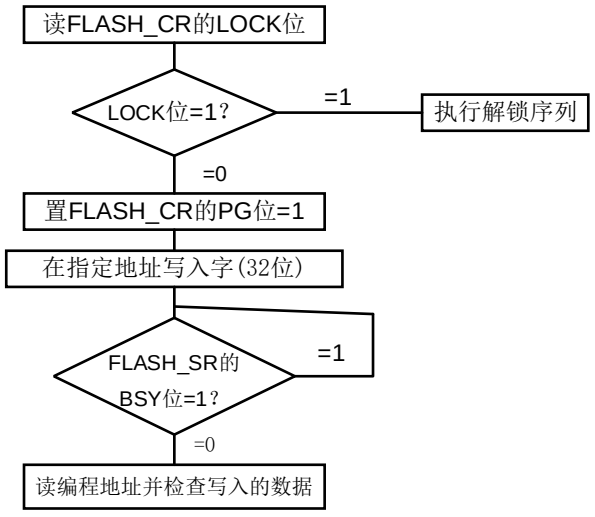


图 5-1 Main FLASH 写操作步骤

这种模式下 CPU 以标准的方式烧写闪存，FLASH_CR 寄存器的 PG 位必须置'1'。FPEC

先读出指定地址的内容并检查它是否被擦除，如未被擦除则不执行编程并在 FLASH_SR 寄存器的 PGERR 位提出警告(唯一的例外是当要烧写的数值是 0x00000000 时，0x00000000 可被正确烧入且 PGERR 位不被置位)；如果指定的地址在 FLASH_WRPR 中设定为写保护，则不执行编程并在 FLASH_SR 寄存器的 WRPRTERR 位置'1'提出警告。FLASH_SR 寄存器的 EOP 为'1'时表示编程结束。

标准的闪存编程顺序如下：

- 执行 FPEC 解锁序列；
- 检查 FLASH_SR 寄存器的 BSY 位，以确认没有其他正在进行的编程操作；
- 设置 FLASH_CR 寄存器的 PG 位为'1'；
- 在指定的地址写入要编程的字；
- 等待 BSY 位变为'0'；
- 读出写入的地址并验证数据。

注意：当 FLASH_SR 寄存器的 BSY 位为'1'时，不能对任何寄存器执行写操作。在对 FLASH 进行任何操作之前，软件必须判断 FLASH 的 BUSY 状态是否结束。

5.4.3.3 擦除主闪存

闪存可以按 Sector 擦除，也可以整片擦除。Sector 擦除闪存的任何一 Sector 都可以通过 FPEC 的 Sector 擦除功能擦除；擦除一 Sector 应遵守下述过程：

- 执行 FPEC 解锁序列
- 检查 FLASH_SR 寄存器的 BSY 位，以确认没有其他正在进行的闪存操作；
- 设置 FLASH_CR 寄存器的 PER 位为'1'；
- 用 FLASH_AR 寄存器选择要擦除的 Sector；
- 设置 FLASH_CR 寄存器的 STRT 位为'1'；
- 等待 BSY 位变为'0'；
- 读出被擦除的 Sector 并做验证。

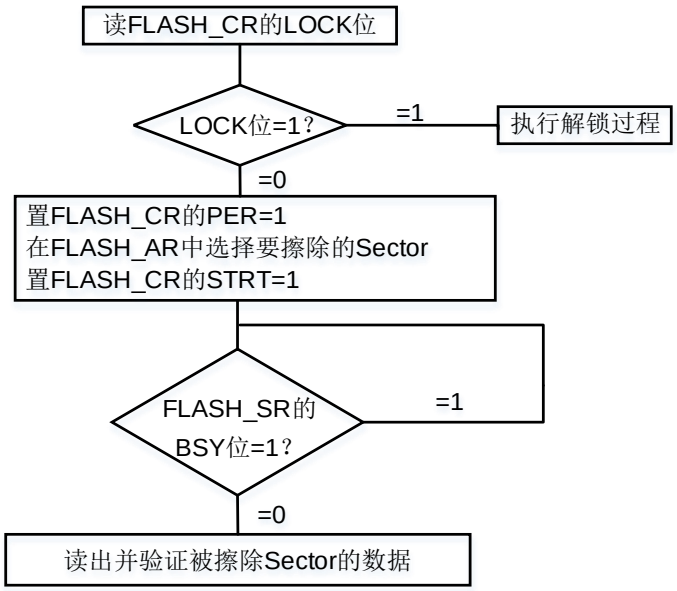


图 5-2 Main FLASH Sector 擦除操作步骤

整片擦除可以用整片擦除功能擦除所有主存储区(用户区)的闪存，信息存储区不受此操作影响。建议使用下述过程：

- 执行 FPEC 解锁序列
- 检查 FLASH_SR 寄存器的 BSY 位，以确认没有其他正在进行的闪存操作；
- 设置 FLASH_CR 寄存器的 MER 位为'1'；
- 设置 FLASH_CR 寄存器的 STRT 位为'1'；
- 等待 BSY 位变为'0'；
- 读出所有 Sector 并做验证。

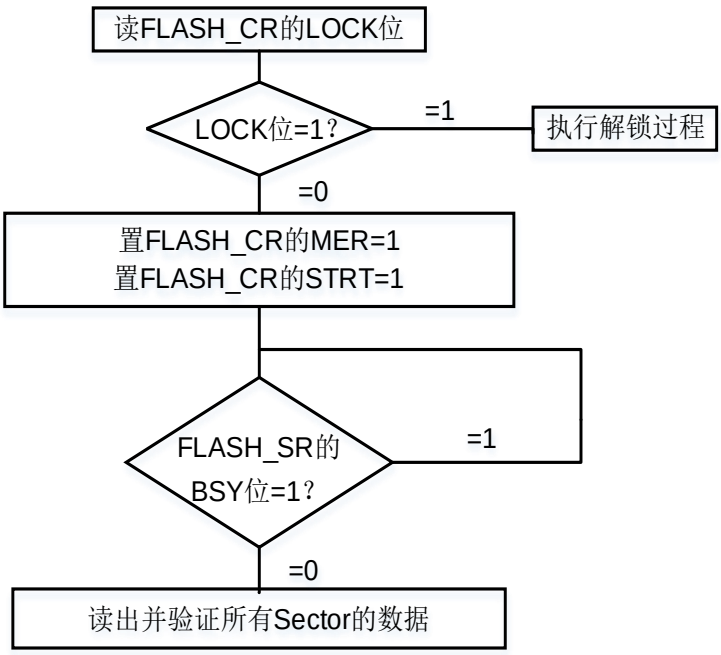


图 5-3 Main FLASH 整片擦除操作步骤

5.4.3.4 选项字节编程

对选项字节的编程与普通的用户地址不同。选项字节的数目只有 21 个字节(1 个字节作为读保护，1 个字节为配置选项，2 个字节存储用户自定义数据，16 个字节作为写保护，1 个字节为 SWD 端口配置选项)。选项字节说明见 5.4.5 章节。

对选项字节编程，除了对 FPEC 解锁外，还需对 FLASH_OPTKEYR 寄存器完成关键字写入操作，必须分别写入 KEY1 和 KEY2(见 5.4.1 节)到 FLASH_OPTKEYR 寄存器，完成该操作后，FLASH_CR 寄存器中的 OPTWRE 位会被硬件置'1'，然后就可以先置位 FLASH_CR 中的 OPTPG 位，再按字单位写目标地址。

FPEC 先读出指定地址的选项字节内容并检查它是否已经被擦除，如未被擦除则不执行编程并在 FLASH_SR 寄存器的 WRPRTERR 位提出警告。FLASH_SR 寄存器的 EOP 为'1'时表示编程结束。

选项字节编程是以字为单位，低字节为选项字节实际内容，高字节为低字节的反码。

选项字节的编程顺序如下：

- 执行 FPEC 解锁序列
- 检查 FLASH_SR 寄存器的 BSY 位，以确认没有其他正在进行的编程操作；
- 解锁 FLASH_CR 寄存器的 OPTWRE 位（写入 KEY1 和 KEY2 到 FLASH_OPTKEYR 寄存器）；
- 设置 FLASH_CR 寄存器的 OPTPG 位为‘1’；
- 写入要编程的字到指定的地址；
- 等待 BSY 位变为‘0’；
- 读出写入的地址并验证数据。

当读闪存保护选项从“保护”变为“未保护”时，在重新设置读保护选项前会自动执行一个整片擦除用户闪存的操作。如果用户要改变读保护之外的选项，则不会出现整片擦除操作。读保护选项上的这一擦除操作保护了闪存中的内容不被非法读出。

擦除过程

选项字节的擦除操作顺序(OPTERASE)如下：

- 执行 FPEC 解锁序列
- 检查 FLASH_SR 寄存器的 BSY 位，以确认没有其他正在进行的编程操作；
- 解锁 FLASH_CR 寄存器的 OPTWRE 位（写入 KEY1 和 KEY2 到 FLASH_OPTKEYR 寄存器）；
- 设置 FLASH_CR 寄存器的 OPTER 位为‘1’；
- 用 FLASH_AR 寄存器选择要擦除的 Sector；
- 设置 FLASH_CR 寄存器的 STRT 位为‘1’；
- 等待 BSY 位变为‘0’；
- 读出被擦除的选项字节并做验证。

5.4.4 存储保护

可以防范 FLASH 用户代码区的程序被非法的读出；同样可以对闪存区的 Sector 加以保护，防止在程序跑飞的情况下不被意外地改变，写保护的基本单位是 2 个 Sector。

5.4.4.1 主闪存写保护

写保护以两个 Sector 为单位(1KB)来控制，配置用户选项字节中的 WARP 位，随后的系统复位将加载新选项字节就可以使能这个保护。如果试图写入或擦除一个受保护的 Sector 区，会引起 FLASH_SR 中的 WRPRTERR 标志位被置位。

解除保护

解除写保护有下述 2 种情形：

情形 1：解除写保护，同时解除读保护：

- 使用闪存控制寄存器(FLASH_CR)的 OPTER 位擦除整个选项字节区域；
- 写入正确的 RDP 代码 0xA5，允许读访问；这个操作将强制擦除主闪存存储器；
- 进行系统复位，重装载选项字节(包含新的 WRP 字节)，写保护被解除。

使用这种方法，将解除整个主闪存模块的写保护。

情形 2：解除写保护，同时保持读保护有效。

使用闪存控制寄存器(FLASH_CR)的 OPTER 位擦除整个选项字节区域；

- 进行系统复位，重装载选项字节(包含新的 WRP 字节)；写保护被解除。
  使用这种方法，将解除除选项字节区之外的整个主闪存模块的写保护，选项字节区仍处于写保护。

5.4.4.2 主闪存读保护

这项保护是通过设置 RDP 选项字节启动的。当保护字节被写入相应的值以后，实施下述保护：

- 只允许从用户代码中对主闪存存储器的读操作(以非调试方式从主闪存存储器启动)，禁止任何外部设备将用户程序从芯片中读出。
- 所有通过 SWD 向内置 SRAM 装载代码并执行代码的功能依然有效，亦可以通过 SWD 从内置 SRAM 启动，这个功能可以用来解除读保护。当读保护的选项字节转变为存储器未保护的数值时，将会执行整片擦除过程。

当 RDP 选项字节和它的反码包含下列数值对时，闪存被置于保护状态：

表 5-2 闪存存储器保护状态

RDP 字节的值	RDP 反码的值	读保护状态
0xFF	0xFF	保护
0xA5	0x5A	未保护
任意值	非 RDP 字节的反码	保护

注意：擦除选项字节块将不会导致自动的整片擦除操作，因为擦除的结果 0xFF 相当于保护状态。

解除读保护的过程：

- 擦除整个选项字节区域，读保护码(RDP)将变为 0xFF，此时读保护仍然有效；
- 写入正确的 RDP 代码 0xA5 以解除存储器的保护，该操作将首先导致对所有用户闪存的整片擦除操作；
- 进行复位(上电复位)以重新加载选项字节(和新的 RDP 代码)，此时读保护被解除。

5.4.4.3 选项字节写保护

默认状态下，选项字节块始终是可以读且被写保护。要想对选项字节块进行写操作(编程/擦除)首先要在 OPTKEYR 中写入正确的键序列(类似解锁)，随后允许对选项字节块的写操作，FLASH_CR 寄存器的 OPTWRE 位标示允许写，清除这位将禁止写操作。

表 5-3 访问状态 vs 保护级别和执行模式

区域	保护级别	用户程序执行(主存储区执行)			调试/从 RAM 上执行		
		读	写	擦	读	写	擦
主存储区	无	可	可	可	可	可	可
	有	可	可	可	不可	不可	可
选项字节区	无	可	可	可	可	可	可
	有	可	可	可	可	可	可

5.4.5 选项字节说明

选项字节共有 21 个字节，由用户根据应用的需要配置。

在选项字节中每个 32 位的字被划分为下述格式：

表 5-4 选项字节格式

位 31~24	位 23~16	位 15~8	位 7~0
选项字节 1 的反码	选项字节 1	选项字节 0 的反码	选项字节 0

选项字节块中选项字节的组织结构如下：

表 5-5 选项字节的组织结构

地址	**[31:24]**	**[23:16]**	**[15:8]**	**[7:0]**
0x1FFF F800	nUSER	USER	nRDP	RDP
0x1FFF F804	nData1	Data1	nData0	Data0
0x1FFF F808	nWRP1	WRP1	nWRP0	WRP0
0x1FFF F80C	nWRP3	WRP3	nWRP2	WRP2
0x1FFF F810	nWRP5	WRP5	nWRP4	WRP4
0x1FFF F814	nWRP7	WRP7	nWRP6	WRP6
0x1FFF F818	保留	保留	nSWDP	SWDP
0x1FFF F81C	保留	保留	保留	保留
0x1FFF F820	nWRP9	WRP9	nWRP8	WRP8
0x1FFF F824	nWRP11	WRP11	nWRP10	WRP10
0x1FFF F828	nWRP13	WRP13	nWRP12	WRP12
0x1FFF F82C	nWRP15	WRP15	nWRP14	WRP14

表 5-6 选项字节说明

RDP：读出保护选项字节 读出保护功能帮助用户保护存在闪存中的代码。该功能由设置信息块中的一个选项字节启用。写入正确的数值(RDP=0xA5，nRDP=0x5A)到这个选项字节后，闪存被开放允许读出访问。详见表 5-3 访问状态 vs 保护级别和执行模式。 *备注：芯片出厂时，主闪存处于非读保护状态，用户可以通过 SWD 口读出闪存中的数据。*
USER：用户选项字节 这个字节用于配置下列功能： - *选择看门狗事件：硬件或软件* - *PE6/外部 reset 端子的选择*
Datax：用户自定义选项字节 两个字节的用户数据 具体功能可由用户定义，这两个地址可以使用选项字节的编程方式编程。
WRP0~WRP15：主闪存写保护选项字节，WRP0~WRP15 共 128bit，每 bit 保护 2 个 Sector（1KB） - 0：实施写保护 - 1：不实施写保护
SWDP：SWD 端口保护位 *SWDP 为非 0xFF 时，RCC_SWDIOCR 寄存器的 SWDPORT 就不能自由改写，只能由 1 改写为 0，一旦 SWDPORT 控制位被软件写 0(SWD 管脚功能无效)，则 PA4/PC4 再也无法配置成 SWD 端口功能，直至 SWDP 被擦回 0xFF。*

每次系统复位后，选项字节装载器读出信息块的数据并保存在寄存器中；每个选择位都在信息块中有它的反码位，在装载选择位时反码位用于验证选择位是否正确，如果有任何的差别，将产生一个选项字节错误标志(OPTERR)。当发生选项字节错误时，对应的选项字节被强置为 0xFF。当选项字节和它的反码均为 0xFF 时(擦除后的状态)，则关闭上述验证功能。所有的选择位(不包括它们的反码位)用于配置该寄存器，CPU 可以读选项字节(FLASH_OBR)寄存器，详见寄存器章节说明。

5.5 寄存器列表

基地址：0x40020400

偏移地址	名称	描述	默认值
0x00	FLASH_ACR	闪存访问控制寄存器	0x00000130
0x04	FLASH_KEYR	FPEC 键寄存器	0x00000000
0x08	FLASH_OPTKEYR	闪存 OPTKEY 寄存器	0x00000000
0x0C	FLASH_SR	闪存状态寄存器	0x00000000
0x10	FLASH_CR	闪存控制寄存器	0x00000180
0x14	FLASH_AR	闪存地址寄存器	0x00000000
0x18	FLASH_ECCCR	ECC 控制寄存器	0x00000000
0x1C	FLASH_OBR	选项字节寄存器	用户设定值
0x20	FLASH_WRPR1	写保护寄存器 1	用户设定值
0x24	FLASH_WRPR2	写保护寄存器 2	用户设定值
0x34	FLASH_WRPR3	写保护寄存器 3	用户设定值
0x38	FLASH_WRPR4	写保护寄存器 4	用户设定值
0x3C	FLASH_SWDP	SWD 保护寄存器	用户设定值

注：所有 FLASH 寄存器只能按 32 位方式读写。

5.6 寄存器说明

5.6.1 FLASH_ACR(闪存访问控制寄存器)

地址偏移：0x00

复位值：0x00000030

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留					CACHE_FLUSH	保留	CACHE_EN	保留					LATENCY		
					R/W		R/W						R/W		

位	标记	功能描述	复位值	读写
31:11	Res	保留	0x0	-
10	CACHE_FLUSH	cache 数据无效化， 写 1 能让 cache 无效，写 0 没有效果。 该位只能读 0。	0x0	R/W
9	Res	保留	0x0	-
8	CACHE_EN	Cache 有效 0：cache 机能无效 1：cache 机能有效	0x0	R/W
7:3	Res	保留，始终读为 5’b00110	0x06	-
2:0	LATENCY	这些位表示 SYSCLK(系统时钟)周期与闪存访问时间的比例 000：零等待状态，当 FCLK ≤ 18MHz 001：一个等待状态，当 18MHz < FCLK ≤ 36MHz 010：两个等待状态，当 36MHz < FCLK ≤ 48MHz 其他：禁止配置	0x0	R/W

注意：Cache on 的时候会一条指令读 128bit，即 4 个 word 的数，所以如果在 ECC 有效的情况下，这些数据都会进行校验，所以 ECC 有效时候的初始化，必须所有使用到的数据再加上边界的 4 个字。

Cache 无效的时候，直接写这个寄存器位 Cache_flush，接下来再设定 Cache_en 这个寄存器位。

注意：改变 latency 值的时候，需要在中断关闭的情况下，中间不插入任何其他指令，对 latency 连续两次写同样的值才有效。

5.6.2 FLASH_KEYR(FPEC 键寄存器)

地址偏移：0x04

复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:0	KEYR	FPEC 键 这些位用于输入 FPEC 的解锁键；	0x0	WO

5.6.3 FLASH_OPTKEYR(闪存 OPTKEY 寄存器)

地址偏移：0x08
复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:0	OPTKEYR	选项字节键 这些位用于输入选项字节的键以解除 OPTWRE；	0x0	WO

5.6.4 FLASH_SR(闪存状态寄存器)

地址偏移：0x0C
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留										EOP	WRPRTERR	保留	PGERR	保留	BSY
										R/W	R/W		R/W		RO

位	标记	功能描述	复位值	读写
31:6	Res	保留，必须保持为清除状态'0'；	0x0	-
5	EOP	操作结束 当闪存操作(编程/擦除)完成时，硬件设置这位为'1'，写入'1'可以清除这位状态； *注：每次成功的编程或擦除都会设置 EOP 状态；*	0x0	R/W
4	WRPRTERR	写保护错误 试图对写保护的闪存地址编程时，硬件设置这位为'1'，写入'1'可以清除这位状态；	0x0	R/W
3	Res	保留，必须保持为清除状态'0'；	0x0	-
2	PGERR	编程错误 试图对内容不是'0xFFFFFFFF'的地址编程时，硬件设置这位为'1'，写入'1'可以清除这位状态； *注：进行编程操作之前，必须先清除 FLASH_CR 寄存器的 STRT 位。*	0x0	R/W
1	Res	保留，必须保持为清除状态'0'；	0x0	-
0	BSY	忙 该位指示闪存操作正在进行。在闪存操作开始时，该位被设置为'1'，在操作结束或发生错误时该位被清除为'0'；	0x0	RO

注:如果是操作读被保护的单元，则不产生任何错误 flag,擦除和写入动作无视，读出来的数据一直是 0x00000000。

5.6.5 FLASH_CR(闪存控制寄存器)

地址偏移：0x10

复位值：0x00000180

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留			EOPIE	保留	ERRIE	OPTWRE	保留	LOCK	STRT	OPTER	OPTPG	保留	MER	PER	PG
			R/W		R/W	R/W		R/W	R/W	R/W	R/W		R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:13	Res	保留，必须保持为清除状态'0'；	0x0	-
12	EOPIE	允许操作完成中断 该位允许在 FLASH_SR 寄存器中的 EOP 位变为'1'时产生中断； 0：禁止产生中断； 1：允许产生中断；	0x0	R/W
11	Res	保留，必须保持为清除状态'0'；	0x0	-
10	ERRIE	允许错误状态中断 该位允许在发生 FPEC 错误时产生中断(当 FLASH_SR 寄存器中的 PGERR/WRPRTERR 置为'1'时)； 0：禁止产生中断； 1：允许产生中断；	0x0	R/W
9	OPTWRE	允许写选项字节 当该位为'1'时，允许对选项字节进行擦写操作。当在 FLASH_OPTKEYR 寄存器写入正确的键序列后，该位被置为'1'，软件可清除此位；	0x0	R/W
8	Res	保留	0x1	-
7	LOCK	锁 只能写'1'。当该位为'1'时表示 FPEC 和 FLASH_CR 被锁住。在检测到正确的解锁序列后，硬件清除此位为'0'。在一次不成功的解锁操作后，下次系统复位前，该位不能再被改变；	0x1	R/W
6	STRT	开始 当该位为'1'时将触发一次擦除操作。该位只可由软件置为'1'并在 BSY 变为'1'时清为'0'；	0x0	R/W
5	OPTER	擦除选项字节 0：无效； 1：开始擦除；	0x0	R/W
4	OPTPG	烧写选项字节 对选项字节编程 0：无效； 1：开始编程；	0x0	R/W
3	Res	保留	0x0	-
2	MER	全擦除 选择擦除所有用户 Sector 0：无效； 1：开始擦除；	0x0	R/W
1	PER	Sector 擦除 0：无效； 1：开始擦除；	0x0	R/W
0	PG	主程序区编程 0：无效； 1：开始编程；	0x0	R/W

5.6.6 FLASH_AR(闪存地址寄存器)

地址偏移：0x14
复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:0	FAR	闪存地址 当进行编程时选择要编程的地址，当进行 Sector 擦除时选择要擦除的 Sector。 *注意：当 FLASH_SR 中的 BSY 位为1'时，不能写这个寄存器。*	0x0	R/W

注：这些位由硬件修改为当前/最后使用的地址。在 Sector 擦除操作中，软件必须修改这个寄存器来指定要擦除的 Sector。

5.6.7 FLASH_ECCCR(ECC 控制寄存器)

地址偏移：0x18
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECCD	ECCC	保留					ECCIE	保留							ADDR_ECCERR
R/W	R/W						R/W								RO
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR_ECCERR														保留	
RO															

位	标记	功能描述	复位值	读写
31	ECCD	ECC 检测错误 0：无 ECC 检测错误 1：ECC 检测时，2bit 同时出错 该位写 1 清零，该位置 1 时，会生成 NMI	0x0	R/W
30	ECCC	ECC 校正 0：无 ECC 校正 1：ECC 检测时，发生 1bit 校正 该位写 1 清零，该位置 1 时，如果 ECCIE 设成 1，则会发生中断。	0x0	R/W
29:25	Res	保留	0x0	-
24	ECC_IE	ECC 校正中断使能 0：禁止 ECCC 中断 1：使能 ECCC 中断	0x0	R/W
23:17	Res	保留	0x0	-
16:2	ADDR_ECCERR	ECC 校正或错误，发生时的地址 在检测到 ECC 错误或者 ECC 校正的情况下，该寄存器保存出错/校正的地址。	0x0	RO
1:0	Res	保留	0x0	-

5.6.8 FLASH_OBR(选项字节寄存器)

地址偏移：0x1C
复位值：用户设定值

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA1								DATA0							
RO								RO							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RST_PORT	保留						WDG_SW	保留						RDPRT	OPTERR
R0							RO							RO	RO

位	标记	功能描述	复位值	读写
31:24	DATA1	DATA1 用户自定义数据；	用户设定值	RO
23:16	DATA0	DATA0 用户自定义数据；	用户设定值	RO
15	RST_PORT	PE6 端子，作为外部复位端子或者 GPIO 端子功能切换 0：GPIO 功能 1：外部复位功能 该位为 User 的第 7 位，对应 0x1FFFF802 的 bit7。	用户设定值	RO
14:9	Res	保留	0x3F	-
8	WDG_SW	选择看门狗事件 0：硬件看门狗； 1：软件看门狗； 该位为 User 的第 0 位，对应 0x1FFFF802 的 bit0。	用户设定值	RO
7:2	Res	保留	0x3F	-
1	RDPRT	读保护标志 当设置为 1，表示闪存存储器的读保护有效； 当设置为 0，表示闪存存储器的读保护无效	用户设定值	RO
0	OPTERR	选项字节错误标志 当该位为'1'时表示选项字节和它的反码不匹配；	用户设定值	RO

注：

OBR 对应的值分别是 DATA1(0x1FFF_F806),DATA0(0x1FFF_F804),USER(0x1FFF_F802),
RDPRT= RDP(0x1FFF_F800)

RDPRT 的逻辑是:如果 RDP 的设定值=0xA5 则清零，除此以外的其他值则置位。

OPTERR 的逻辑是所有 user option byte 领域互补 check，若所有的互补都成立，则不置位。

5.6.9 FLASH_WRPR1(闪存写保护寄存器 1)

地址偏移：0x20
复位值：用户设定值

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WRP3								WRP2							
RO								RO							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRP1								WRP0							
RO								RO							

位	标记	功能描述	复位值	读写
31:24	WRP3	写保护，对应 Sector48~63，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO
23:16	WRP2	写保护，对应 Sector32~47，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO
15:8	WRP1	写保护，对应 Sector16~31，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO
7:0	WRP0	写保护，对应 Sector0~15，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO

5.6.10 FLASH_WRPR2(闪存写保护寄存器 2)

地址偏移：0x24
复位值：用户设定值

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WRP7								WRP6							
RO								RO							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRP5								WRP4							
RO								RO							

位	标记	功能描述	复位值	读写
31:24	WRP7	写保护，对应 Sector112~127，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO
23:16	WRP6	写保护，对应 Sector96~111，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO
15:8	WRP5	写保护，对应 Sector80~95，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO
7:0	WRP4	写保护，对应 Sector64~79，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO

5.6.11 FLASH_WRPR3(闪存写保护寄存器 3)

地址偏移：0x34
复位值：用户设定值

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WRP11								WRP10							
RO								RO							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRP9								WRP8							
RO								RO							

位	标记	功能描述	复位值	读写
31:24	WRP11	写保护，对应 Sector176~191，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO
23:16	WRP10	写保护，对应 Sector160~175，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO
15:8	WRP9	写保护，对应 Sector144~159，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO
7:0	WRP8	写保护，对应 Sector128~143，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO

5.6.12 FLASH_WRPR4(闪存写保护寄存器 4)

地址偏移：0x38
复位值：用户设定值

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WRP15								WRP14							
RO								RO							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRP13								WRP12							
RO								RO							

位	标记	功能描述	复位值	读写
31:24	WRP15	写保护，对应 Sector240~255，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO
23:16	WRP14	写保护，对应 Sector224~239，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO
15:8	WRP13	写保护，对应 Sector208~223，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO
7:0	WRP12	写保护，对应 Sector192~207，每 bit 控制 2 个 Sector； 0：写保护生效； 1：写保护失效；	用户设定值	RO

5.6.13 FLASH_SWDP(SWD 保护寄存器)

地址偏移：0x3C
复位值：设定值

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

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	-
0	SWDP	SWD 端口保护，对应的保护位: 这个 bit 是设定 RCC.SWDIOCR 寄存器设定的保护位，不是直接控制 SWD 端口的。 1：用户可以自由设定 SWDIOCR 0：用户不可以自由设定 SWDIOCR，只能将 SWDPORT 位由 1 写成 0，不能由 0 改为 1。 *该位对应的值是 SWDP(0x1FFF F818)的 FLASH 值，* *若 SWDP 为 0xFF,则该位为 1;* *若 SWDP 为非 0xFF,则该位为 0;*	根据用户程序	RO

6 SRAM 控制器(ECC_SRAM)

6.1 简介

ECC_SRAM 的全称是 SRAM Error Checking and Correction，表示该 SRAM 带有自动校验和修正功能。

SRAM 出错的时候一般不会造成整个 SRAM 不能读取或是全部出错，而是整个 SRAM 中只有一个或几个位出错。ECC_SRAM 采用 SEC-DED 算法，即单 bit 纠错、两 bits 检知算法，对单 bit 错误能自动纠正，对 2bits 同时出现错误时不能自动纠正，但是系统会对这类错误响应 NMI 异常，从而保证系统的稳定性。对于 SRAM 总线上同时出现 2bits 以上的错误时，是不一定能检测到的。

6.2 功能

表 6.1 ECC_SRAM 功能表

项目	With ECC function(SEC-DED 算法)
SRAM 容量	最多 20K Bytes
SRAM 地址空间	0x2000_0000 to 0x2000_4FFF
错误纠正和检测	当检测到单 bit 错误状态时，自动纠正错误的 bit 当检测到两 bits 错误状态时，不能自动纠正错误，但是会产生异常报警。 当两 bits 以上错误同时出现时，不一定能检测到。
错误检知时的相应动作	默认 ECC 纠错功能是无效的，ECC 纠错功能有效以后，软件可以配置设定 1bit 错误自动纠正时中断响应。 发生 2bit 错误时，系统一定会发生 NMI 异常。 同时发生 2bit 以上错误时，系统不能保证检测。

- SRAM 初始化

  Reset 解除以后，ECC 纠错功能默认是无效的。通过设定 RAM_ECCEN 寄存器的 ECCEN 位可以使其功能有效，功能有效以后必须首先以 32 位写的方式，对需要访问的 SRAM 的空间进行初始化，即写入一个已知的数值。推荐写入 0x00000000。

  如果不写一个已知的数值的话，读和写未初始化的 SRAM 都会发生一个 ECC 校验错误的异常。

- 错误响应

  当使能 ECCCR.ECC_IE=1，且 ECC 校验功能检测并自动纠正 1bit 错误时，就会产生中断。中断向量号是 27,详细请参考 NVIC 章节。

  当发生该中断时，请一定按照以下流程来处理中断程序：

  - 中断标志寄存器位 ECCCR.ECCC =1 自动置 1。
  - ECC_SRAM 错误地址寄存器 ECCCR.ADDR_ECCERR 会自动填充当前产生错误的地址，程序读取该地址。

- 通过对 ECCCR.ECCC 写 1 来清除 ECCCR.ECCC=0。
- 可以退出当前中断。

当 ECC 校验功能检测 2bit 错误同时发生时，就会 NMI 异常，这个异常是无条件响应的。当发生该异常时，请参考以下流程来处理异常程序：

- 异常标志寄存器位 ECCCR.ECCD =1 自动置 1。
- ECC_SRAM 错误地址寄存器 ECCCR.ADDR_ECCERR 会自动填充当前产生错误的地址，程序读取该地址。
- 通过对 ECCCR.ECCD 写 1 来清除 ECCCR.ECCD=0。
- 通过设定 RCC_RSTCR 寄存器的 CPURST 位来产生 CPU reset，从而保证系统的安全性。

➢ 读写周期

本 SRAM 支持 Byte（8bits）、Half-word（16bits）、Word（32bits）三种读写操作。可在系统时钟频率下进行读写操作，在 ECC 纠错功能无效的时候，无须等待周期。但是在 ECC 纠错功能有效以后，Word（32bits）的读写操作还是零等待周期，Byte 和 Half-word 操作时，会产生先读再写回的动作，所以读写操作会有一定的延迟，请注意。

➢ 读写位宽

本控制器支持 Byte（8bits）、Half-word（16bits）、Word（32bits）三种位宽的读写操作。Byte 操作的地址必须按 Byte 对齐，Half-word 操作的目标地址必须按 Half-word 对齐（地址最低位为 0），Word 操作的地址必须按 Word 对齐（地址最低两位为 0）。如果读写操作的目标地址没有按照位宽规定对齐，该操作无效，并且系统会产生 Hard Fault 出错中断。非对齐的访问 Hard Fault 的管理是 CPU 内管理的。

6.3 寄存器列表

基地址：0x40001C00

偏移地址	名称	描述	默认值
0x24	RAM_ECCEN	SRAM ECC 错误检测功能使能寄存器	0x00000000
0x28	RAM_ECCCR	SRAM ECC 控制寄存器	0x00000000

注：以上寄存器的写保护都是SYSCON_UNLOCK寄存器，只有当写保护无效时(即写入有效时)，以上寄存器才能被写入。

6.4 寄存器说明

6.4.1 SRAM_ECCEN(SRAM ECC 错误检测功能使能寄存器)

地址偏移：0x24

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	KEY	只有高位写0x5A69时配置该寄存器才有效，写其它值时无效。 读这些位时，读到的是0x0000	0x0	WO
15:1	Res	保留	0x0	-
0	ECCEN	ECC 错误校验使能位 0：ECC 校验无效 1：ECC 校验有效 该位设定有效以后，要对 SRAM 进行访问之前，一定要对 SRAM 进行 32 位的写动作，写入已知的数值(推荐写入 0x0000000)。	0x0	R/W

6.4.2 SRAM_ECCCR(SRAM ECC 控制寄存器)

地址偏移：0x28

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ECCD	ECCC	保留					ECCIE	保留							
R/W	R/W						R/W								

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

位	标记	功能描述	复位值	读写
31	ECCD	ECC 检测错误 0：无 ECC 检测错误 1：ECC 检测时，2bit 同时出错 该位写 1 清零，该位由硬件置 1 时，会生成 NMI	0x0	RCW1
30	ECCC	ECC 校正 0：无 ECC 校正 1：ECC 检测时，发生 1bit 校正 该位写 1 清零，该位由硬件置 1 时，如果 ECC_IE 设成 1，则会发生中断。	0x0	RCW1
29:25	Res	保留	0x0	-
24	ECC_IE	ECC 校正中断使能 0：禁止 ECCC 中断 1：使能 ECCC 中断	0x0	R/W
23:15	Res	保留	0x0	-
14:2	ADDR_ECCERR	ECC 校正或错误，发生时的地址 在检测到 ECC 错误或者 ECC 校正的情况下，该寄存器保存出错/校正的地址。 此地址加上偏移量 0x2000_0000 就是对应的真实的 RAM 地址。	0x0	RO
1:0	Res	保留	0x0	-

7 系统复位与时钟(RCC)

7.1 复位

7.1.1 复位控制器介绍

本产品具有 9 个复位信号来源，每个复位信号都可以让 CPU 重新运行，绝大多数寄存器会被复位到复位值，程序计数器 PC 会被复位成复位地址。

- 上电复位 POR
- 外部端子复位 NRST/PE6，低电平复位
- IWDG 复位
- WWDG 复位
- 低电压复位(LVD)
- 软件复位(Cortex®-M0+SYSRESETREQ)
- 硬件复位(Cortex®-M0+LOCKUP)
- 寄存器复位(CPURST)
- MCU 复位(MCURST)

每个复位源都有一个专用的复位标志位来表示，由硬件置“1”，软件清除。除了 POR 复位标志位，其他的复位标志位都可以被 POR 来清除。

下图描述了 9 个复位的来源：

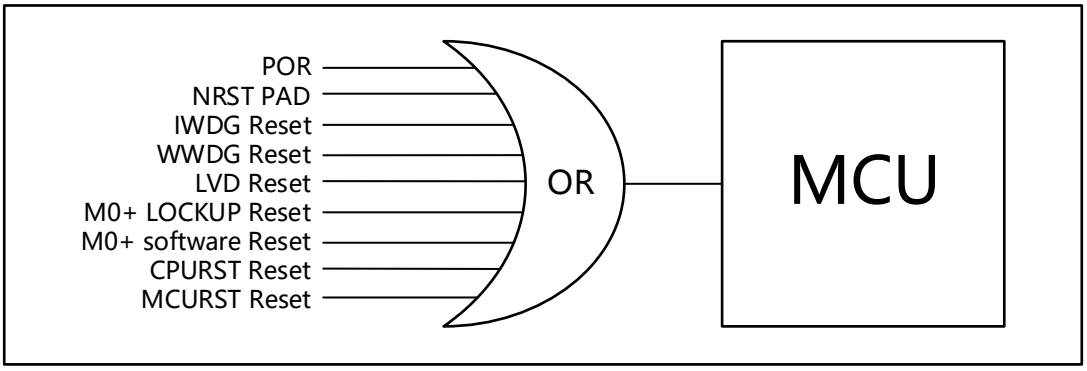


图 7-1 复位来源示意图

7.1.2 复位源

7.1.2.1 上电复位 POR

当芯片上电以及掉电时，如果电源低于一个阈值电压，内部会产生一个 POR 信号，当电源高于该阈值电压时，释放 POR 信号。POR 信号会把芯片的 SFR、控制信号复位。

7.1.2.2 外部引脚复位

把外部复位引脚拉到地电位就会产生一个系统复位。这个复位引脚内部带有上拉电阻，另外，在内部集成了一个毛刺滤波电路，系统会自动过滤小于 10us(典型值)的毛刺信号，因此，用户在使用复位引脚产生复位信号时，必须产生大于 10us 的低电平以保证芯片可以正确接收到复位信号。

7.1.2.3 IWDG 复位

独立看门狗复位，请参考独立看门狗(IWDG)一章说明。

7.1.2.4 WWDG 复位

窗口看门狗复位，请参考系统窗口看门狗(WWDG)一章说明。

7.1.2.5 LVD 低电压复位

LVD 复位，请参考低电压检测器(LVD)一章说明。

7.1.2.6 Cortex®-M0+软件复位

通过将 Cortex®-M0+系统控制寄存器 SCB_AIRCR.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。

7.1.2.7 Cortex®-M0+LOCKUP 硬件复位

当 Cortex®-M0+遇到严重的异常时，它会将自己的 PC 指针停在当前地址处，并锁死自己，并在几个时钟周期延时之后复位整个内核区域。

7.1.2.8 寄存器复位

可以通过写 RCC_RSTCR.MCURST 和 RCC_RSTCR.CPURST 来复位芯片。

7.2 系统时钟

时钟控制模块主要控制系统时钟以及外设时钟，可以配置不同的时钟源作为系统时钟、可以配置不同的系统时钟分频、可以启动或禁用外设时钟，另外为了确保高精度，内部时钟都具有校准功能。

本产品支持以下五个不同的时钟源作为系统时钟：

- 内部高速 RC 时钟 HIRC(8MHz)(默认主频)
- PLL 时钟(最高 48MHz)
- 内部低速 RC 时钟 LIRC(38.4KHz/32.768KHz)
- 外部高速晶振时钟 HXT(8-32MHz)
- LXT、HXT 可以通过管脚 PB7 从外部输入。使用外部振荡输入时，需要使能相应的振荡控制选择。外部振荡控制选择在 RCC_SYSCLKCR、RCC_LXTCR 寄存器中。

每种时钟源都可以单独的打开或关断，当它们不用时，可以关断来降低功耗。

有多个分频器可用于配置 AHB 和 APB 时钟域，AHB 和 APB 域的最大时钟频率为 48MHz。

Cortex 系统定时器(SysTick)由 AHB 时钟驱动，其可由 AHB/4 或 AHB 时钟频率直接驱动(通过 Cortex Systick 配置位来配置)。RCC 通过 AHB 时钟(HCLK)的 4 分频后作为 Cortex 系统定时器(SysTick)的外部时钟，通过对 SysTick 控制与状态寄存器 RCC_STICKCR 的设置，可选择上述时钟或 Cortex 时钟(HCLK)作为 SysTick 时钟。

需要特别注意的是：在切换系统时钟源的过程中需要按照正确的流程，具体流程参见“7.2.7 系统时钟切换”章节。

7.2.1 系统时钟树

下图为本产品的时钟树图：

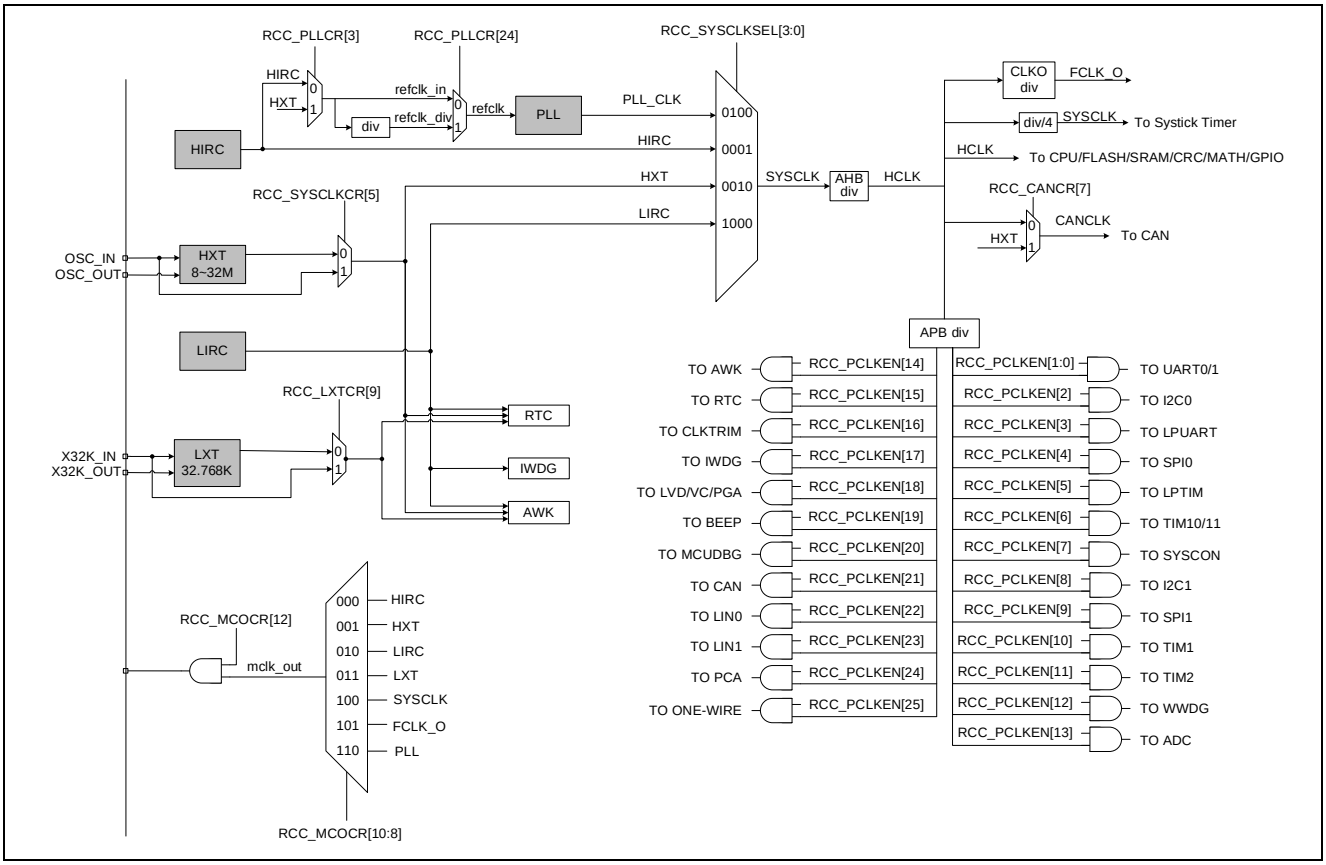


图 7-2 CPS32K11x 时钟树结构图

7.2.2 内部高速 RC 时钟 HIRC

默认的系统时钟是内部 8MHz 的高速 HIRC 时钟，在芯片上电或复位后即开始工作，出厂精度在±1%（常温@VDD=5V）。由于芯片经过焊接、贴片等环节，内部 HIRC 时钟频率可能会发生偏移导致精度超出±2%的范围（全温度范围），因此可以通过寄存 RCC_HIRCCR.TRIM 来二次校准内部高速 HIRC 时钟的频率，给出精确的 8MHz 频率值。因为内部高速时钟启动快，约 3us，为了让系统更为快速的响应外部中断，系统在从深度休眠模式被唤醒时，可以选择使用该时钟源作为系统时钟。

7.2.3 内部低速 RC 时钟 LIRC

内部低速时钟频率可配置成 38.4KHz/32.768KHz，出厂精度在±3%（常温@VDD=5V），在低速及对精度要求不高的应用场景下，可选择该时钟源作为系统时钟。同样由于芯片经过焊接、贴片等环节，内部 LIRC 时钟频率可能会发生偏移导致精度超出±3%的范围，因此可以通过寄存 RCC_LIRCCR.TRIM 来二次校准内部高速 LIRC 时钟的频率，给出精确的 38.4/32.768KHz 频率值。

7.2.4 外部高速晶振时钟 HXT

外部 8~32MHz 晶振时钟需根据用户系统需求外接一个 8~32MHz 的高速晶振。

外部晶振时钟可以选择两种输入方式：

- HXT 外部晶体/陶瓷谐振器
- HXT 用户外部时钟

为了减少时钟输出的失真和缩短启动稳定时间，晶体/陶瓷谐振器和负载电容器必须尽可能地靠近振荡器引脚。负载电容值必须根据所选择的振荡器来调整。

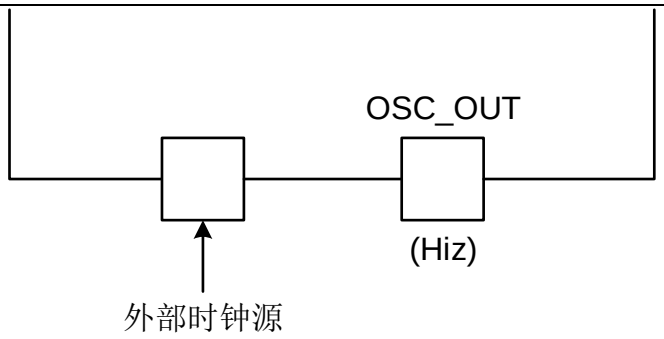
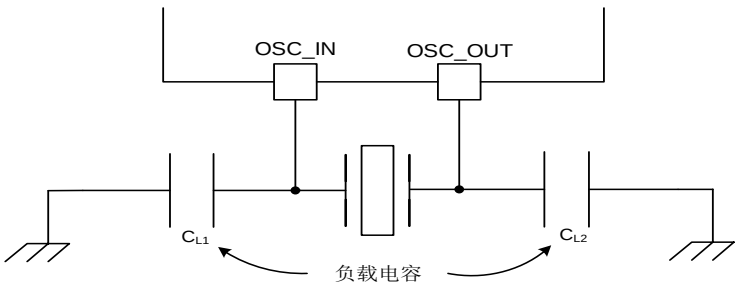
<table>
<tr><th></th><th>硬件配置</th></tr>
<tr><td>外部时钟</td><td> 
</td></tr>
<tr><td>晶体/陶瓷谐振器</td><td> 
</td></tr>
</table>

图 7-3 HXT/LXT 时钟源

7.2.5 外部低速晶振时钟 LXT

外部低速晶振时钟需外接一个 32.768KHz 的低功耗晶振，具有超高精度以及低功耗。

外部时钟源旁路请参考图 7-3 HXT/LXT 时钟源。

注：此时钟不可作为系统时钟源，只能作为 RTC 和 AWK 的时钟源。

7.2.6 系统时钟启动过程

上述四种时钟源都有一个启动稳定的时间，时钟源使能后都会等待一段稳定时间后，再把时钟输给系统使用，图 7-4 内部高速时钟启动示意图以内部高速 RC 时钟 HIRC 为例说明时钟的启动稳定过程。

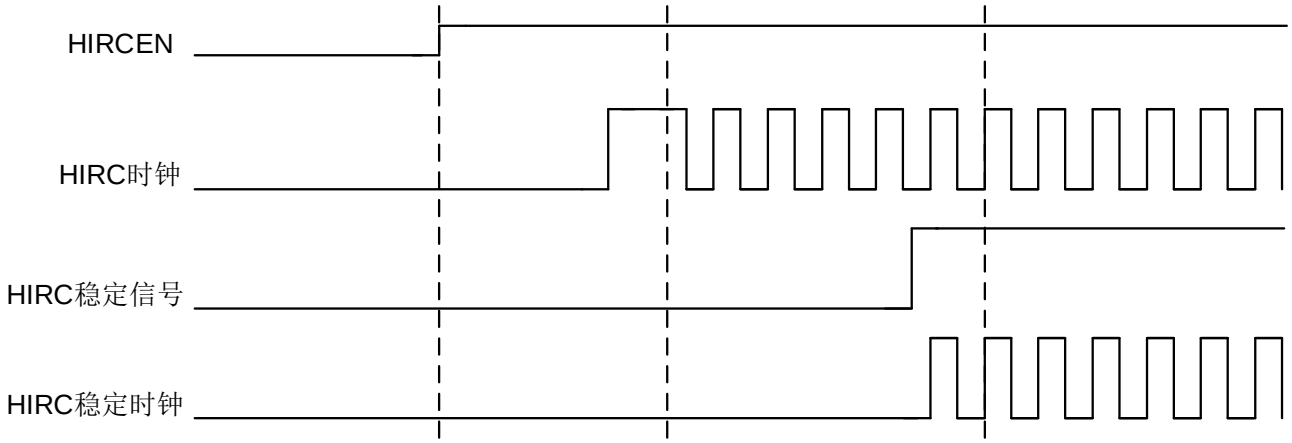


图 7-4 内部高速时钟启动示意图

芯片上电后，系统使用 8MHz 的内部高速时钟作为启动时钟，启动完成后用户可以根据自己的需要来修改高速时钟的频率以及切换时钟源。

7.2.7 系统时钟切换

时钟源的切换是由寄存器 RCC_SYSCLKSEL 来控制。当系统时钟从当前时钟切换到目标时钟时，必须按照一定的流程来实现，否则就会出现异常。

7.2.7.1 内部高速切换到外部高速

从 HIRC(内部高速时钟)切换到 HXT(外部 8~32MHz 晶振)为例，具体流程如下：

1) 通过 RCC_SYSCLKCR.HXTPORT 位来配置要切换的时钟 HXT 使用的引脚为模拟引脚，或者通过对于引脚的 GPIOx_AFR=0xF 来配置为模拟功能
2) 写寄存器 RCC_SYSCLKCR.HXTEN 位使能 HXT 时钟
3) 等待要切换的时钟 HXT 时钟稳定
4) 写寄存器 RCC_SYSCLKSEL.CLKSW 来切换时钟
5) 根据需要关闭 HIRC 时钟

注意：

使用外部高速 24MHz 晶振时，RCC_HXTCR.HXTSTARTUP 稳定时间控制位设置为 0x3，使用默认配置 0x2 稳定时间可能不够。

下图为 HIRC 到 HXT 切换时序:

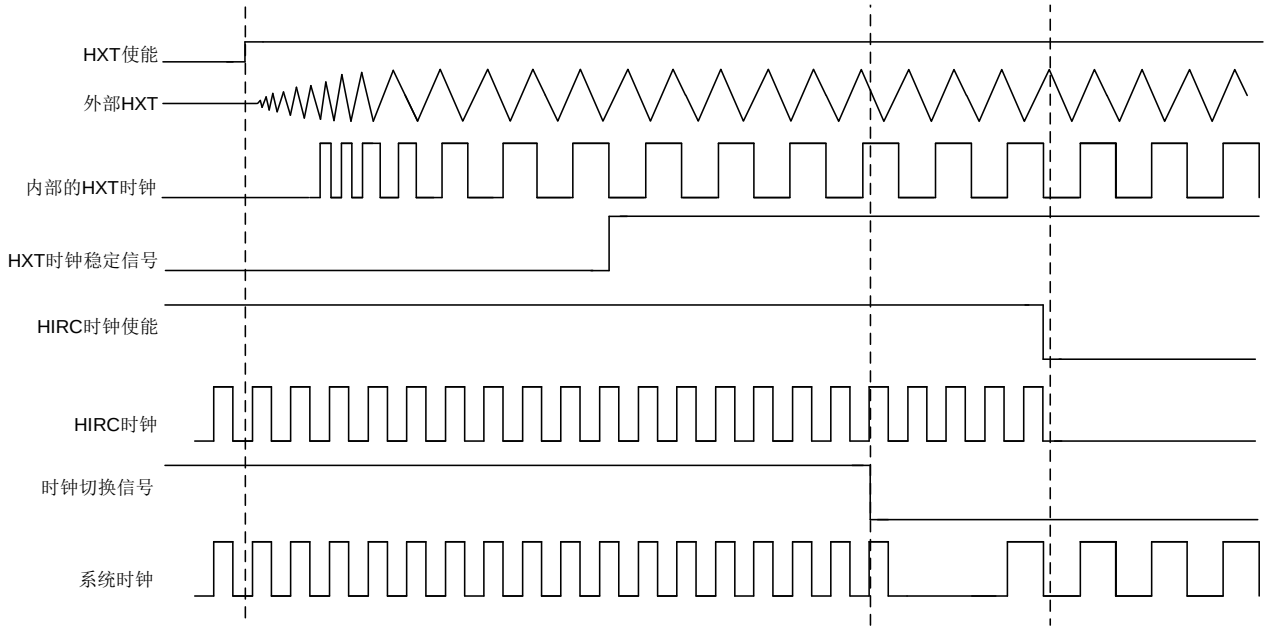


图 7-5 时钟切换示意图

7.2.7.2 内部低速切换到外部高速

从 LIRC(内部低速时钟)切换到 HXT(外部 8~32MHz 晶振)为例，具体流程如下：

1) 通过 RCC_SYSCLKCR.HXTPORT 位来配置要切换的时钟 HXT 使用的引脚为模拟引脚，或者通过对于引脚的 GPIOx_AFR=0xF 来配置为模拟功能
2) 写寄存器 RCC_SYSCLKCR.HXTEN 位使能 HXT 时钟
3) 等待要切换 HXT 时钟稳定
4) 写寄存器 RCC_SYSCLKSEL.CLKSW 来切换时钟
5) 根据需要关闭 LIRC 时钟

7.2.8 系统时钟输出

微控制器允许输出时钟信号到外部 MCO 引脚。

有如下的 7 种信号可选为 MCO 时钟输出：

- HIRC
- HXT
- LIRC
- LXT
- SYSCLK
- FCLK 以及分频输出
- PLL 时钟

MCO 时钟的选择由时钟输出控制寄存器 RCC_MCOCR.MCOSEL 位决定。

7.2.9 系统时钟安全控制

当设定 RCC_SYSCLKCR.CLKFAILEN 有效，且使能 CLKTRIM 的时钟监视功能后，如果 HXT 时钟停止，则系统时钟会切换到内部高速时钟。

具体请参考 CLKTRIM 模块的监测功能

7.2.10 IWDG 时钟

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

7.2.11 AWK 时钟

AWKCLK 时钟源可以由 HXT 分频、LXT 或 LIRC 时钟提供。

7.2.12 低功耗模式

APB 外设时钟以及部分 AHB 外设时钟可以用软件禁止。睡眠模式会停止 CPU 时钟，CPU 睡眠时，存储器接口时钟(FLASH 和 RAM 接口)会被停止。

当配置了 SYSCON_CFGR0.DBGDSLP_DIS 后，CPU 在相应的深度睡眠模式下也具有调试功能。

7.3 寄存器列表

这节详细描述了 RCC 控制模块的寄存器功能。

RCC 基址：0x40020000

表 7-1 RCC 寄存器列表和复位值

偏移地址	名称	描述	复位值
0x00	RCC_HCLKDIV	AHB 时钟分频寄存器	0x00000000
0x04	RCC_PCLKDIV	APB 时钟分频寄存器	0x00000000
0x08	RCC_HCLKEN	AHB 周边模块时钟使能寄存器	0x00000100
0x0C	RCC_PCLKEN	APB 周边模块时钟使能寄存器	0x00000000
0x10	RCC_MCOCR	时钟输出控制寄存器	0x00000000
0x14	—	保留	—
0x18	RCC_RSTCR	系统 Reset 控制寄存器	0x00000000
0x1C	RCC_RSTSR	Reset 状态寄存器	0x000000A0
0x20	RCC_SYSCLKCR	时钟源设置寄存器	0x00000001
0x24	RCC_SYSCLKSEL	系统时钟源选择寄存器	0x00000001
0x28	RCC_HIRCCR	内部高速 RC 振荡器控制寄存器	0x00001XXX
0x2C	RCC_HXTCR	外部高速晶体振荡器控制寄存器	0x00001020
0x30	RCC_LIRCCR	内部低速 RC 振荡器控制寄存器	0x00000XXX
0x34	RCC_LXTCR	外部低速晶体振荡器控制寄存器	0x00000425
0x38	RCC_IRQLATENCY	M0 IRQ 延时控制	0x00000000
0x3C	RCC_STICKCR	SysTick Timer 周期校准寄存器	0x01009C3F
0x40	RCC_SWDIOCR	管脚特殊功能选择寄存器	0x00000001
0x44	RCC_PERIPRST	周边模块复位控制寄存器	0x00000000
0x48	RCC_RTCRST	RTC 复位寄存器	0x00000000
0x4C	RCC_PLLCR	PLL 控制寄存器	0x00206041
0x50	RCC_CANCR	CAN 时钟寄存器	0x00000000
0x54~0x5C	—	保留	—
0x60	RCC_UNLOCK	寄存器写保护	0x00000000
0x64~0x33C	—	保留	—
0x340	RCC_HXTUNLOCK	外部高速晶体振荡器寄存器保护	0x00000000

7.4 寄存器说明

7.4.1 AHB 时钟分频寄存器(RCC_HCLKDIV)

地址偏移：0x00

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								AHBCKDIV							
								R/W							

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7:0	AHBCKDIV	系统 HCLK 时钟分频 0：HCLK=SYSCLK 1~255：Divide by 2×DIV (HCLK = SYSCLK/(2×AHBCKDIV)).	0x0	R/W

7.4.2 APB 时钟分频寄存器(RCC_PCLKDIV)

地址偏移：0x04

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								APBCKDIV							
								R/W							

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7:0	APBCKDIV	系统 PCLK 时钟分频 0：PCLK=HCLK 1~255：Divide by 2×DIV (PCLK = HCLK / (2×APBCKDIV)).	0x0	R/W

7.4.3 AHB 周边模块时钟使能寄存器(RCC_HCLKEN)

地址偏移：0x08

复位值：0x00000100

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

位	标记	功能描述	复位值	读写
31:9	Res	保留	0x0	–
8	FLASHCKE	FLASH 控制器模块时钟使能。关闭后 FLASH 配置寄存器不可写，FLASH 中的程序仍然可以运行。 0：关闭 1：使能；	0x1	R/W
7	Res	保留	0x0	R/W
6	GPIOECKE	GPIOE 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
5	MATHCKE	MATH 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
4	CRCCKE	CRC 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
3	GPIODCKE	GPIOD 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
2	GPIOCCKE	GPIOC 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
1	GPIOBCKE	GPIOB 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
0	GPIOACKE	GPIOA 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W

7.4.4 APB 周边模块时钟使能寄存器(RCC_PCLKEN)

地址偏移：0x0C

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留						1-WIRECKEN	PCACKEN	LIN1CKEN	LIN0CKEN	CANCKEN	DBGCKEN	BEEPCKEN	LVDVCPGACKEN	IWDGCKEN	CLKTRIMCKEN
						R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTCCKEN	AWKCKEN	ADCCKEN	WWDGCKEN	TIM2CKEN	TIM1CKEN	SPI1CKEN	I2C1CKEN	SYSCONCKEN	BASETIMCKEN	LPTIMCKEN	SPI0CKEN	LPUARTCKEN	I2C0CKEN	UART1CKEN	UART0CKEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:26	Res	保留	0x0	–
25	1-WIRECKEN	1. Wire PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
24	PCACKEN	PCA PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
23	LIN1CKEN	LIN1 PCLK 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
22	LIN0CKEN	LIN0 PCLK 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
21	CANCKEN	CAN PCLK 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
20	DBGCKEN	Debug PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
19	BEEPCKEN	BEEP PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
18	LVDVCPGACKEN	LVD_VC_PGA PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
17	IWDGCKEN	IWDG PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
16	CLKTRIMCKEN	CLKTRIM PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
15	RTCCKEN	RTC PCLK 时钟使能	0x0	R/W

		0：时钟关闭 1：时钟使能		
14	AWKCKEN	AWK PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
13	ADCCKEN	ADC PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
12	WWDGCKEN	WWDG PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
11	TIM2CKEN	TIM2 PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
10	TIM1CKEN	TIM1 PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
9	SPI1CKEN	SPI1 PCLK 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
8	I2C1CKEN	I2C1 PCLK 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
7	SYSCONCKEN	SYSCON PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
6	BASETIMCKEN	TIM10/11 PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
5	LPTIMCKEN	Low Power Timer PCLK 时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
4	SPI0CKEN	SPI0 PCLK 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
3	LPUARTCKEN	Low Power UART PCLK 寄存器配置时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
2	I2C0CKEN	I2C0 PCLK 模块时钟使能 0：时钟关闭 1：时钟使能	0x0	R/W
1	UART1CKEN	UART1 PCLK 模块时钟使能 1：时钟使能 0：时钟关闭	0x0	R/W
0	UART0CKEN	UART0 PCLK 模块时钟使能	0x0	R/W

		0：时钟关闭 1：时钟使能		

7.4.5 时钟输出控制寄存器(RCC_MCOCR)

地址偏移：0x10

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:13	Res	保留	0x0	–
12	MCOEN	MCO 输出使能 写： 0：MCO 输出禁止 1：MCO 输出使能 读： 0：MCO 未开始输出 1：MCO 开始输出 注意，读该位时是输出使能信号通过输出时钟同步后的信号。	0x0	R/W
11	Res	保留	0x0	–
10:8	MCOSEL	时钟输出源选择 000：HIRC 001：HXT 010：LIRC 011：LXT 100：SYSCLK 101：FCLK 以及分频输出 110：PLL 输出 111：Reserved，禁止设定	0x0	R/W
7:0	MCODIV	FCLK 时钟分频系数 0：FCLK 1~255：Divide by 2×MCODIV (HCLK = FCLK / (2×MCODIV)).	0x0	R/W

7.4.6 系统复位控制寄存器(RCC_RSTCR)

地址偏移：0x18

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSTKEY															
WO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSTKEY														CPU RST	MCU RST
WO														R/W	R/W

位	标记	功能描述	复位值	读写
31:2	RSTKEY	必须写入 0x156A99A6 才有效。写其他值时都无效。	0x0	WO
1	CPURST	寄存器 CPU 复位，该复位发生时，不会重新装载用户配置区中的 ISP 设定 0：Normal 1：Reset CPU	0x0	R/W
0	MCURST	寄存器 MCU 复位，该复位发生时，会重新装载用户配置区中的 ISP 设定 0：Normal 1：Reset MCU	0x0	R/W

注 1：MCU reset by write 0x55AA6699 to RSTCON；CPU reset by write 0x55AA669A to RSTCON.

注 2：只有 RCC_UNLOCK 寄存器保护解除后，才能写该寄存器。

7.4.7 系统复位状态寄存器(RCC_RSTSR)

地址偏移：0x1C

复位值：0x000000A0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留					LOL_RST	CMU_LOC	SFTRST	PADRST	LOCKUPRST	PORRST	LVDRST	IWDGRST	WWDGRST	CPURST	MCURST
					RCW0	RCW0	RCW0	RCW0	RCW0	RCW0	RCW0	RCW0	RCW0	RCW0	RCW0

位	标记	功能描述	复位值	读写
31:11	Res	保留	0x0	–
10	LOL_RST	0：无 PLL loss of lock 复位 1：有 PLL loss of lock 复位	0x0	RCW0
9	CMU_LOC	Clock monitor lose of clock 复位标志 0：无 clock monitor LOC 复位 1：有 clock monitor LOC 复位	0x0	RCW0
8	SFTRST	Cotrex-M0+ CPU 软件复位标志，需要软件清除， 0：无 Cotrex-M0+ CPU 软件复位发生 1：发生 Cotrex-M0+ CPU 软件复位	0x0	RCW0
7	PADRST	RESET 端口复位标志，能够被 POR 复位 0：无端口复位发生 1：发生端口复位	0x1	RCW0
6	LOCKUPRST	Cotrex-M0+ CPU Lockup 复位标志， 0：无 Cotrex-M0+ CPU Lockup 复位发生 1：发生 Cotrex-M0+ CPU Lockup 复位	0x0	RCW0
5	PORRST	POR 复位标志 0：POR 无复位发生 1：POR 发生复位	0x1	RCW0
4	LVDRST	LVD 复位标志 0：LVD 无复位发生 1：LVD 发生复位	0x0	RCW0
3	IWDGRST	IWDG 复位标志 0：IWDG 无复位发生 1：IWDG 发生复位	0x0	RCW0
2	WWDGRST	WWDG 复位标志 0：WWDG 无复位发 1：WWDG 发生复位	0x0	RCW0
1	CPURST	寄存器 CPU 复位标志，该复位发生时，不会重新装载 USER OPTION BYTE 中的 ISP 和 IAP 设定 0：寄存器 CPURST 无复位发生 1：寄存器 CPURST 复位发生	0x0	RCW0
0	MCURST	寄存器 MCU 复位标志，该复位发生时，会重新装载 USER OPTION BYTE 中的 ISP 和 IAP 设定 0：寄存器 MCURST 无复位发生	0x0	RCW0

		1：寄存器 MCURST 复位发生		

注：只受 POR 控制。

7.4.8 系统时钟源配置寄存器(RCC_SYSCLKCR)

地址偏移：0x20

复位值：0x00000001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
WO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WKBYHIRC	保留						CLKFAILEN	保留	HXTPORT	HXTBYP	保留		LIRCEN	HXTEN	HIRCEN
R/W							R/W		R/W	R/W			R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	KEY	只有高位写 0x5A69 时配置该寄存器才有效，写其它值时无效。	0x0	WO
15	WKBYHIRC	0：从 Deep Sleep 唤醒，system clock 来源怎么进怎么出。 1：从 Deep Sleep 唤醒时使用 HIRC 开始唤醒，硬件自动使能 HIRC，并且 system clock 自动切换到 HIRC，原时钟继续开启。	0x0	R/W
14:9	Res	保留	0x0	–
8	CLKFAILEN	时钟失效检测使能控制 0：时钟失效检测禁止。 1：时钟失效检测使能，当检测到时钟失效，自动切换系统时钟到 HIRC。	0x0	R/W
7	Res	保留	0x0	–
6	HXTPORT	OSCIN/OSCOUT 管脚配置 0：GPIO 功能模式(数字功能) 1：HXT 管脚模式(模拟功能)	0x0	R/W
5	HXTBYP	外部高速时钟输入选择 0：HXT 内部振荡模块非旁路模式，与 OSC_IN/OSC_OUT 相连 1：HXT 内部振荡模块旁路模式，HXT 从管脚 OSCIN 直接输入	0x0	R/W
4:3	Res	保留	0x0	–
2	LIRCEN	内部低速时钟 LIRC 使能信号 0：关闭 1：使能 当系统时钟选择该时钟时，不能关闭	0x0	R/W
1	HXTEN	外部 8~32MHz 晶振 HXTOSC 使能信号 0：关闭 1：使能 注意：1，使用时，与该晶振连接的两个外部端口需要设置成模拟端口；当系统进入 Deep Sleep，此高速时钟会自动关闭。	0x0	R/W

		2，当 HXT 停止检出时，此位会被硬件清 0 当系统时钟选择该时钟时，不能关闭		
0	HIRCEN	内部高速时钟 HIRC 使能信号 0：关闭 1：使能 注意：当系统进入 Deep Sleep，此高速时钟会自动关闭。 当 HXT 停止检出时，如果系统时钟选择为 HXT，且 CLKFAIL_EN 使能，HIRC_EN 会由硬件自动置 1. 当系统时钟选择该时钟时，不能关闭	0x1	R/W

注意，只有 RCC_UNLOCK 寄存器保护解除后，才能写该寄存器。

7.4.9 系统时钟源选择寄存器(RCC_SYSCLKSEL)

地址偏移：0x24

复位值：0x00000001

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

位	标记	功能描述	复位值	读写
31:16	KEY	只有高位写 0x5A69 时配置该寄存器才有效，写其它值时无效。	0x0	WO
15:4	Res	保留	0x0	–
3:0	CLKSW	System Clock Source Select. 0001：HIRC 选择 0010：HXT 选择 0100：PLL 选择 1000：LIRC 选择 其余情况均为保留 注意：当 HXT 停止检出时，如果系统时钟选择为 HXT，且 CLKFAIL_EN 使能，HIRC_EN 会由硬件自动置 1.系统时钟自动选择 HIRC。	0x1	R/W

注意：该位写受 RCC_UNLOCK 保护

7.4.10 内部高速 RC 振荡器控制寄存器(RCC_HIRCCR)

地址偏移：0x28

复位值：0x00001XXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
WO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留			HIRCRDY	保留			TRIM								
			RO				R/W								

位	标记	功能描述	复位值	读写
31:16	KEY	只有高位写 0x5A69 时配置该寄存器才有效，写其它值时无效。	0x0	WO
15:13	Res	保留	0x0	–
12	HIRCRDY	HIRC 时钟稳定标志位。 0：代表 HIRC 未稳定，不可以被内部电路使用。 1：代表 HIRC 已经稳定，可以被内部电路使用。	0x1	RO
11:9	Res	保留	0x0	–
8:0	TRIM	HIRC trim 值 从 FLASH 中预设的值加载过来，常温下 HIRC 出厂精度为±2%，用户可以调整这组值来调整 HIRC 的精度。	0xXXX	R/W

注意：只有 RCC_UNLOCK 寄存器保护解除后，才能写该寄存器。

7.4.11 外部高速晶体振荡器控制寄存器(RCC_HXTCR)

地址偏移：0x2C

复位值：0x00001020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
WO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留			HXT_FAIL_EN	保留				HXTFLT_EN	HXTRDY	HXTSTARTUP		XSEL		IBSEL	
			R/W					R/W	RO	R/W		R/W		R/W	

位	标记	功能描述	复位值	读写
31:16	KEY	只有高位写 0x5A69 时配置该寄存器才有效，写其它值时无效。	0x0	WO
15:13	Res	保留	0x0	–
12	HXT_FAIL_EN	HXT 的硬件停振检测使能 1：停振检测功能无效 0：停振检测功能有效 这一位受到 RCC_HXTLOCK 的 bit1 的写入保护	0x1	R/W
11:8	Res	保留	0x0	–
7	HXTFLT_EN	HXT 输出滤波使能 0：输出滤波无效 1：输出滤波使能	0x0	R/W
6	HXTRDY	外部 8~32MHz 晶振稳定标志位。 0：代表外部高速晶振时钟未稳定，不可以被内部电路使用。 1：代表外部高速晶振时钟已经稳定，可以被内部电路使用。	0x0	RO
5:4	HXTSTARTUP	外部 8~32MHz 晶振稳定时间选择 00：1024 个周期; 01：2048 个周期; 10：4096 个周期; 11：16384 个周期; 使用高速晶振时钟时稳定时间需要设置为 11，否则稳定时间不够，可能导致系统时钟切换时或使用高速晶振时钟深度休眠唤醒时导致系统不稳定。	0x2	R/W
3:2	XSEL	振幅控制位 00：弱 01：较弱 10：较强 11：强 注：使用晶振时，请参考厂商提供的匹配电容的大小。	0x0	R/W
1:0	IBSEL	偏置电流控制位 00：弱 01：较弱 10：较强 11：强	0x0	R/W

注意，只有 RCC_UNLOCK 寄存器保护解除后，才能写该寄存器。

7.4.12 内部低速 RC 振荡器控制寄存器(RCC_LIRCCR)

地址偏移：0x30

复位值：0x00000XXX

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留			LIRC RDY	LIRC STARTUP		保留	TRIM								
			RO	R/W			R/W								

位	标记	功能描述	复位值	读写
31:16	KEY	只有高位写 0x5A69 时配置该寄存器才有效，写其它值时无效。	0x0	WO
15:13	Res	保留	0x0	–
12	LIRCRDY	内部低速时钟稳定标志位。 0：代表内部低速未稳定，不可以被内部电路使用。 1：代表内部低速已经稳定，可以被内部电路使用。	0x0	RO
11:10	LIRCSTARTUP	内部低速时钟稳定时间选择 00：4 个周期; 01：16 个周期; 10：64 个周期; 11：256 个周期;	0x0	R/W
9	Res	保留	0x0	–
8:0	TRIM	LIRC TRIM 值 从 FLASH 中预设的值加载过来，常温下 LIRC 出厂精度为±3%，用户可以调整这组值来调整 LIRC 的精度。	0xXXX	R/W

注意：只有 RCC_UNLOCK 寄存器保护解除后，才能写该寄存器。

7.4.13 外部低速晶体振荡器控制寄存器(RCC_LXTCR)

地址偏移：0x34

复位值：0x00000425

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY												保留	LXT_CKIN_SEL	保留	
WO													R/W		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				LXTPORT	LXTAON	LXTBYP	LXTEN	LXTFLT_EN	LXTRDY	LXTSTARTUP		XSEL		IBSEL	
				R/W	R/W	R/W	R/W	R/W	RO	R/W		R/W		R/W	

位	标记	功能描述	复位值	读写
31:20	KEY	只有高位写 0x5A6 时配置该寄存器才有效，其它值无效	0x0	WO
19	Res	保留	0x0	-
18	LXT_CKIN_SEL	LXT_CKIN 输入选择 在 LXT 振荡器被旁路的情况下（LXTBYP=1） 0：LXT 时钟从 PB7 输入 1：LXT 时钟从 PA7 输入	0x0	R/W
17:12	Res	保留	0x0	–
11	LXTPORT	X32K_IN/X32K_OUT 功能选择 0：GPIO 功能(数字功能) 1：X32K 管脚模式(模拟功能) 注意：该位写受 RCC_UNLOCK 保护	0x0	R/W
10	LXTAON	LXT 只能使能，不能禁止控制 0：LXT_EN 允许禁止控制。 1：LXT_EN 只能使能，不能禁止控制。	0x1	R/W
9	LXTBYP	由软件设置和清除，该位仅在外部 32.768KHz 振荡器关闭的情况下写值。 0：LXT 振荡器非旁路模式 1：LXT 振荡器旁路模式，低速外部时钟可选择从 PB7 或 PA7 输入 注：使用外部低速振荡时需要使能低速晶体振荡的使能位 LXT_EN	0x0	R/W
8	LXTEN	外部 32.768KHz 晶振时钟 LXT 使能信号。 0：关闭 1：使能 当系统时钟选择该时钟时，不能关闭	0x0	R/W
7	LXTFLT_EN	LXT 输出滤波使能 0：输出滤波无效 1：输出滤波使能	0x0	R/W
6	LXTRDY	外部 LXT 时钟稳定标志位。 0：外部 LXT 时钟未稳定，不可以被内部电路使用。 1：外部 LXT 时钟已经稳定，可以被内部电路使用。	0x0	RO
5:4	LXTSTARTUP	外部 32.768KHz 晶振稳定时间选择 00：1024 个周期; 01：2048 个周期;	0x2	R/W

		10：4096 个周期; 11：16384 个周期; 使用高速晶振时钟时稳定时间需要设置为 11，否则稳定时间不够，可能导致系统时钟切换时或使用高速晶振时钟深度休眠唤醒时导致系统不稳定。		
3:2	XSEL	振幅控制位 00：弱 01：较弱 10：较强 11：强 注：使用晶振时，请参考厂商提供的匹配电容的大小。	0x1	R/W
1:0	IBSEL	偏置电流控制位 00：弱 01：较弱 10：较强 11：强	0x1	R/W

注意，只有 RCC_UNLOCK 寄存器保护解除后，才能写该寄存器。这个寄存器在 RTC 域，只有 POR 和 RCC_RTCRST 才能复位这个寄存器。

7.4.14 M0 IRQ 延时控制寄存器(RCC_IRQLATENCY)

地址偏移：0x38

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7:0	IRQLATENCY	Cortex-m0+处理器支持零等待状态内存的零抖动中断延迟。IRQLATENCY 指定在 NVIC 中挂起的中断和在 AHB-lite 接口上发出的该中断的 vector fetch 之间的最小循环数。	0x0	R/W

注意，只有 RCC_UNLOCK 寄存器保护解除后，才能写该寄存器。

7.4.15 SystemTick Timer 控制寄存器(RCC_STICKCR)

地址偏移：0x3C

复位值：0x01009C3F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留						NOREF	SKEW	STCALIB							
						R/W	R/W	R/W							

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

位	标记	功能描述	复位值	读写
31:26	Res	保留	0x0	–
25	NOREF	Systick 定时器是否使用外部参考时钟 0：HCLK/4 1：使用内核时钟 注意： 1)与系统寄存器 SysTick->CTRL.CLKSOURCE(0XE000E010)任意一个设置为 1 后使用内核时钟 2)使用参考时钟 HCLK/4 作为 Systick 时钟时，参考时钟频率不允许高于系统时钟 HCLK	0x0	R/W
24	SKEW	10ms STCALIB 值是否准确 0：准确 1：不准确	0x1	R/W
23:0	STCALIB	Systick 10ms 校准值，此值为使用外部参考时钟 HCLK/4(4MHz)的 10ms 校准值。	0x009C3F	R/W

7.4.16 SWDIO 端口控制寄存器(RCC_SWDIOCR)

地址偏移：0x40

复位值：0x00000001

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															SWDPORT
															R/W

位	标记	功能描述	复位值	读写
31:16	KEY	只有高位写 0x5A69 时配置该寄存器才有效，写其它值时无效。	0x0	WO
15:1	Res	保留	0x0	-
0	SWDPORT	配置 PA4 和 PC4 的管脚功能模式 0：周边模块功能模式 1：SWD 管脚功能	0x1	R/W

注意：该位写受 RCC_UNLOCK 保护，同时受 FLASH_SWDP 位保护，详细参考 5.6.13 FLASH_SWDP(SWD 保护寄存器)。

7.4.17 周边模块复位控制寄存器(RCC_PERIPRST)

地址偏移：0x44

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留	GPIOERST	MATHRST	CRCRST	GPIODRST	GPIOCRST	GPIOBRST	GPIOARST	LIN1RST	LIN0RST	CANRST	DBGRST	BEEPRST	LVD_VC_OPARST	PCARST	CLKTRIMRST
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1-WIRERST	AWKRST	ADCRST	WWDGRST	TIM2RST	TIM1RST	SPI1RST	I2C1RST	SYSCONRST	BASETIMRST	LPTIMRST	SPI0RST	LPUARTRST	I2C0RST	UART1RST	UART0RST
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31	Res	保留	0x0	-
30	GPIOERST	GPIOE 模块复位 0：正常工作 1：复位	0x0	R/W
29	MATHRST	MATH 模块复位 0：正常工作 1：复位	0x0	R/W
28	CRCRST	CRC 模块复位 0：正常工作 1：复位	0x0	R/W
27	GPIODRST	GPIOD 模块复位 0：正常工作 1：复位	0x0	R/W
26	GPIOCRST	GPIOC 模块复位 0：正常工作 1：复位	0x0	R/W
25	GPIOBRST	GPIOB 模块复位 0：正常工作 1：复位	0x0	R/W
24	GPIOARST	GPIOA 模块复位 0：正常工作 1：复位	0x0	R/W
23	LIN1RST	LIN1 模块复位 0：正常工作 1：复位	0x0	R/W
22	LIN0RST	LIN0 模块复位 0：正常工作 1：复位	0x0	R/W
21	CANRST	CAN 模块复位 0：正常工作 1：复位	0x0	R/W
20	DBGRST	MCU DEBUG 模块复位 0：正常工作	0x0	R/W

		1：复位		
19	BEEPRST	BEEP 模块复位 0：正常工作 1：复位	0x0	R/W
18	LVD_VC_OPARST	LVD_VC_OPA 模块复位 0：正常工作 1：复位	0x0	R/W
17	PCARST	PCA 模块复位 0：正常工作 1：复位	0x0	R/W
16	CLKTRIMRST	Clock TRIM 模块复位 0：正常工作 1：复位	0x0	R/W
15	1-WIRERST	2. Wire 模块复位 0：正常工作 1：复位	0x0	R/W
14	AWKRST	AWK 模块复位 0：正常工作 1：复位	0x0	R/W
13	ADCRST	ADC 模块复位 0：正常工作 1：复位	0x0	R/W
12	WWDGRST	WWDG 模块复位 0：正常工作 1：复位	0x0	R/W
11	TIM2RST	TIM2 模块复位 0：正常工作 1：复位	0x0	R/W
10	TIM1RST	TIM1 模块复位 0：正常工作 1：复位	0x0	R/W
9	SPI1RST	SPI1 模块复位 0：正常工作 1：复位	0x0	R/W
8	I2C1RST	I2C1 模块复位 0：正常工作 1：复位	0x0	R/W
7	SYSCONRST	SYSCON 模块复位 1：复位 0：正常工作	0x0	R/W
6	BASETIMRST	Base TIM10/TIM11 复位 0：正常工作 1：复位	0x0	R/W
5	LPTIMRST	Low Power Timer 复位 1：复位 0：正常工作	0x0	R/W

4	SPI0RST	SPI0 模块复位 0：正常工作 1：复位	0x0	R/W
3	LPUARTRST	LPUART 模块复位 0：正常工作 1：复位	0x0	R/W
2	I2C0RST	I2C0 模块复位 0：正常工作 1：复位	0x0	R/W
1	UART1RST	UART1 模块复位 0：正常工作 1：复位	0x0	R/W
0	UART0RST	UART0 模块复位 0：正常工作 1：复位	0x0	R/W

注意：只有 RCC_UNLOCK 寄存器保护解除后，才能写该寄存器。
外设复位的流程为，先写入 1 到对应外设的比特位，等待一定周期后（建议大于 2 个外设时钟周期）再写入 0 完成复位。

7.4.18 RTC 复位寄存器(RCC_RTCRST)

地址偏移：0x48

复位值：0x00000000

位	标记	功能描述	复位值	读写
31:16	KEY	只有高位写0x5A69时配置该寄存器才有效，写其它值时无效	0x0	WO
15:1	Res	保留	0x0	–
0	RTCRST	RTC 模块复位 0：正常工作 1：复位	0x0	R/W

注意：该位写受 RCC_UNLOCK 保护。
RTC 外设复位的流程为，先写入 1 到对应外设的比特位，等待一定周期后（建议大于 2 个 RTC 时钟周期）再写入 0 完成复位。

7.4.19 PLL 控制寄存器(RCC_PLLCR)

地址偏移：0x4C

复位值：0x00206041

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留				LOL_INTEN	LOL_ACT	LOLCHK_EN	REFCLK_PRS	LOCK	READY	STARTUP			M		
				R/W	R/W	R/W	R/W	R/W	R/W	R/W			R/W		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
M				N						OD		REFSEL	OEN	BP	PD
R/W				R/W						R/W		R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:28	Res	保留	0x0	–
27	LOL_INTEN	Loss of PLL lock中断使能 0：关闭中断 1：使能中断	0x0	R/W
26	LOL_ACT	0：interrupt模式 1：复位模式	0x0	R/W
25	LOLCHK_EN	Loss of PLL lock check enable 0：Loss of PLL lock检查无效 1：Loss of PLL lock检查有效	0x0	R/W
24	REFCLK_PRS	PLL输入参考时钟分频控制 对PLL输入参考时钟（内部8MHz或者外部晶振）进行分频，分频时钟作为PLL输入时钟 0：不分频 1：2分频	0x0	R/W
23	LOCK	PLL锁定后由硬件置“1” 0：PLL未锁定 1：PLL锁定	0x0	RO
22	READY	PLL稳定标志 0：PLL未稳定 1：PLL稳定	0x0	RO
21:19	STARTUP	PLL稳定时间选择 000： 8个PLL周期 001：16个PLL周期 010：32个PLL周期 011：64个PLL周期 100：128个PLL周期 101：256个PLL周期 110：512个PLL周期 111：1024个PLL周期	0x4	R/W
18:12	M	Loop Divider control bit 2≤M≤127	0x6	R/W
11:6	N	Pre-div control bit 1≤N≤63	0x1	R/W

5:4	OD	Output divider control bit 00：1分频 01：2分频 10：4分频 11：8分频	0x0	R/W
3	REFSEL	PLL输入时钟源选择 0：HIRC时钟 1：HXT晶振时钟	0x0	R/W
2	OEN	输出使能控制位 0：PLL输出关闭，FOUT=0 1：PLL正常输出	0x0	R/W
1	BP	Bypass Mode控制位 0：Bypass Mode无效 1：PLL处于Bypass Mode，FOUT=FIN/N	0x0	R/W
0	PD	Power Down使能控制位 0：PLL工作 1：PLL睡眠	0x1	R/W

注意：PLL 输出时钟 FOUT 与输入参考时钟 FIN 的倍频关系为：FOUT = FIN *（M/N）*（1/OD）；

N = N5 * 32 + N4 * 16 + N3 * 8 + N2 * 4 + N1 * 2 + N0 * 1 ；
M = M6 * 64 + M5 * 32 + M4 * 16 + M3 * 8 + M2 * 4 + M1 * 2 + M0 * 1
OD = $2^{2*OD[1]} * 2^{1*OD[0]}$；
FOUT 是输出时钟；
FIN 是输入频率；
N 是 Pre divider 的值（1≤N≤63）；
M 是 Loop divider 的值（2≤M≤127）；
OD 是 Post divider 的值；
FIN / N 应在比较频率范围 4 MHz~20 MHz 内；
FIN * M / N 应在 VCO 频率范围 40 MHz~60 MHz 内。

7.4.20 CAN 时钟寄存器(RCC_CANCR)

地址偏移：0x50

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7	CAN_CLK_SEL	CAN 时钟选择 0：AHB 分频时钟 1：外部振荡器时钟	0x0	R/W
6:2	Res	保留	0x0	–
1:0	CAN_TIMCLK_DIV	CAN CiA 603 time stamp 时钟分频 0：8 分频 1：16 分频 2：24 分频 3：48 分频	0x0	R/W

7.4.21 寄存器写保护控制寄存器(RCC_UNLOCK)

地址偏移：0x60

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:1	KEY	只有高位写 0x2AD5334C 时配置该寄存器才有效，写其它值时无效。	0x0	WO
0	UNLOCK	0：寄存器写保护启用，不能对受保护的寄存器写操作 1：寄存器写保护禁止，可以对受保护的寄存器写操作	0x0	R/W

注：RCC_UNLOCK 写入 0X55AA6699,解除寄存器保护

7.4.22 寄存器写保护控制寄存器(RCC_HXTUNLOCK)

地址偏移：0x340

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	KEY	只有高位写 0x5A69 时配置该寄存器才有效，写其它值时无效。	0x0	WO
15:2	Res	保留	0x0	-
1	HXT_UNLOCK	0：HXTCR 的 bit12 写入保护，不能对受保护的寄存器写操作 1：HXTCR 的 bit12 写入保护无效，可以对受保护的寄存器写操作	0x0	R/W
0	Res	保留	0x0	-

注：只有 RCC_UNLOCK 寄存器保护接触后，才能写该寄存器。

8 时钟安全(CLOCK MONITOR)

8.1 简介

时钟监测模块是专门用来校准或者监测时钟的电路。该模块分为两种模式，校准模式和监测模式。

- 校准模式：选择精准的时钟源来校准不精准的时钟，反复校准，直到被校准时钟源的频率达到精度要求。客户可以通过调整 HIRC/LIRC 的 trim 寄存器来获得比较精准的片上发振时钟。
- 监测模式：选择稳定的时钟源来监测系统工作时钟，在设定的监测周期下，监测系统工作时钟是否有失效的情况发生并产生中断或者系统复位。监测模式下又分为以下两种监测方式：

1 硬监测:外部高速晶振时钟(HXT)模块自动监测发振停止，一旦 HXT 的发振停止，就立刻能被监测到，并将系统时钟自动切换到 HIRC(内部高速 RC 时钟)。

2 软监测:该监测模式主要用于选择一个稳定的时钟源作为参考时钟，在设定的时间周期下监测系统工作时钟的异常状态。该监测功能能够检测时钟的频率是否异常，一旦检知该异常，就产生系统复位/或者产生异常。

8.2 主要特性

- 校准模式
  - 5 种待校准时钟选择
  - 6 种参考时钟选择
  - 支持中断方式
- 监测模式
  - 2 种待监测方式(硬监测和软监测)
  - 3 种待监测时钟源
  - 6 种参考时钟源
  - 支持系统复位或者中断方式

8.2.1 Clock monitor 系统构成图

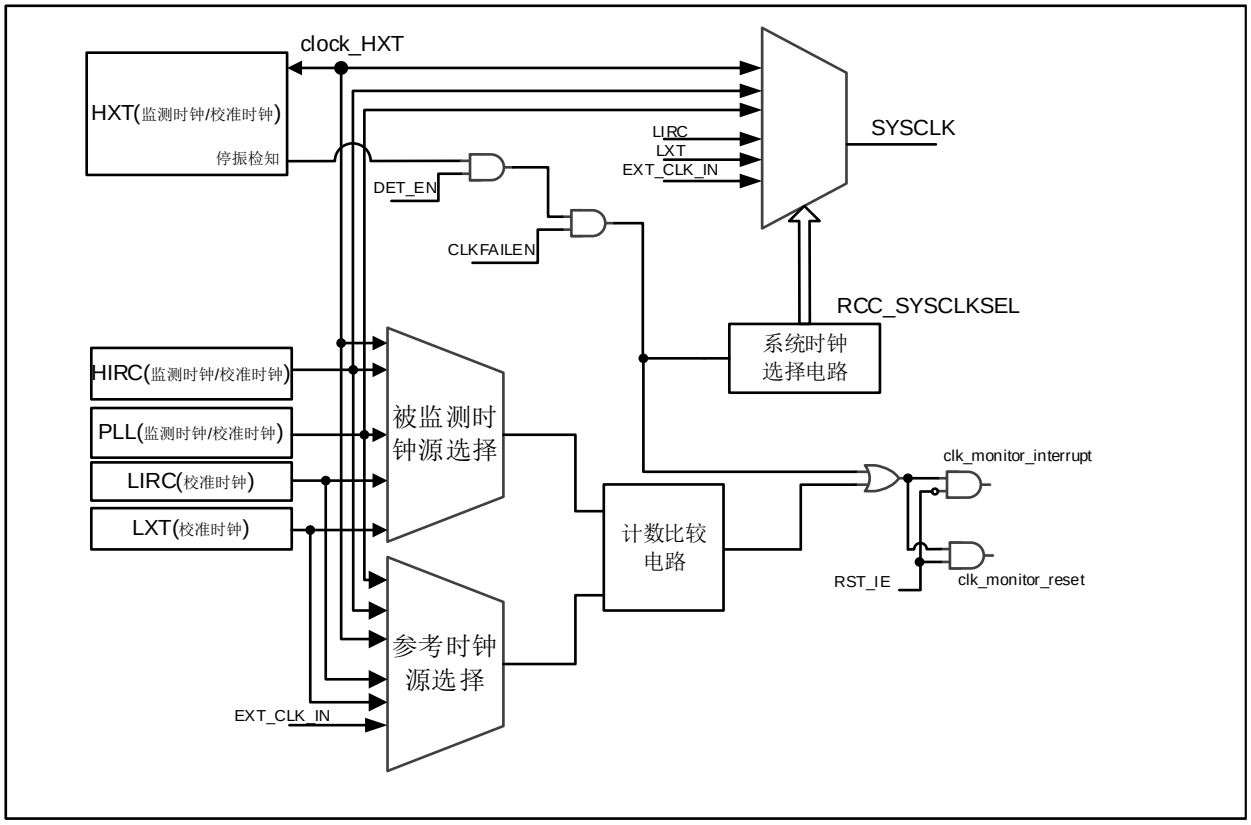


图 8-1 Clock monitor 时钟构成图

8.3 Clock monitor 功能描述

8.3.1 Clock monitor 时钟校准

8.3.1.1 时钟校准原理

在校准一个频率不精准，但是有参数设置可以调节频率的时钟（待校准时钟 CAL_CLK）时，需要一个精准的时钟作为参考时钟(REF_CLK)。设定校准时间 Ttrim，待校准时钟频率 Fcal,参考时钟频率 Fref,使用待校准时钟和参考时钟同时计数，在校准时间结束时停止计数。读取待校准时钟计数器的值 M，参考时钟计数器的值 N，

得到等式 Ttrim = M/Fcal = N/Fref

推导出 Fcal = Fref * M / N

由公式判断出，参考时钟的频率误差率越小，待校准时钟的误差率就越小。

计算出待校准时钟频率后，如果误差率超出系统要求的范围，可以调节待校准时钟的参数后，再次重复上面的步骤校准，直到待校准时钟频率的误差率满足系统的要求。

8.3.1.2 时钟校准模块设置流程

时间校准模块的待校准时钟有 5 个时钟源，参考时钟有 6 个时钟源，参考 CLKTRIM_CR 寄存器的相应设定。

如下图所示，时钟校准模块有 2 个 32 位计数器，

- 1 个是以参考时钟为时钟，可配置初值的减计数器，初值由寄存器 CLKTRIM_REFCON.RCNTVAL 来配置，计数器值可以从寄存器 CLKTRIM_REFCNT 读出，当计数器减计数到 0 时，停止计数并产生中断。
- 1 个是以待校准时钟为时钟，可配置溢出值的加计数器，溢出值由寄存器 CLKTRIM_CALCON.CCNTVAL 来配置，计数器值可以从寄存器 CLKTRIM_CALCNT 读出，当计数器加计数到溢出值时，停止计数并产生中断。

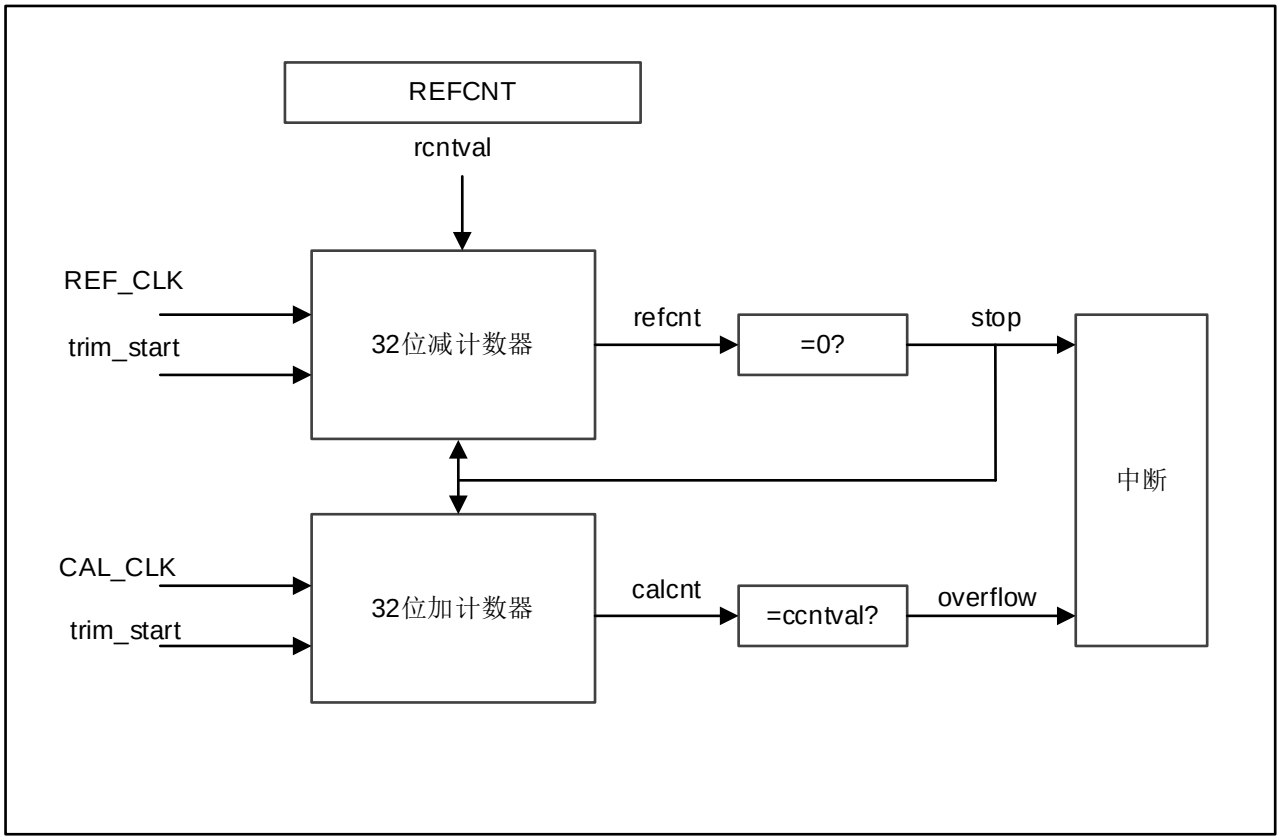


图 8-2 中断产生逻辑图

1. 设置 CLKTRIM_CR.REFCLK_SEL 寄存器选择参考时钟。
2. 设置 CLKTRIM_CR.CALCLK_SEL 寄存器选择被校准时钟。
3. 设置 CLKTRIM_REFCON.RCNTVAL 寄存器为校准时间。
4. 设置 CLKTRIM_CR.RST_IE 寄存器使能中断。(不可设成 1)
5. 设置 CLKTRIM_CR.TRIM_START 寄存器开始校准。
6. 参考时钟计数器和待校准时钟计数器开始计数。
7. 当参考时钟计数器从初始值减计数到 0 时，CLKTRIM_IFR.STOP 置 1，触发中断。
8. 中断服务子程序判断 CLKTRIM_IFR.STOP 为 1，读取寄存器 CLKTRIM_REFCNT 和 CLKTRIM_CALCNT 的值

9. 清零 CLKTRIM_CR.TRIM_START 寄存器结束单次校准。
10. 根据时钟校准原理章节提到的公式计算出待校准时钟的频率，
其中 M = 寄存器 CLKTRIM_CALCNT 的值，
N = 寄存器 CLKTRIM_REFCON.RCNTVAL 的值
当待校准时钟的频率不满足误差率要求时，调整待校准时钟参数，
重新执行步骤 5~10，直到待校准时钟满足误差率要求。

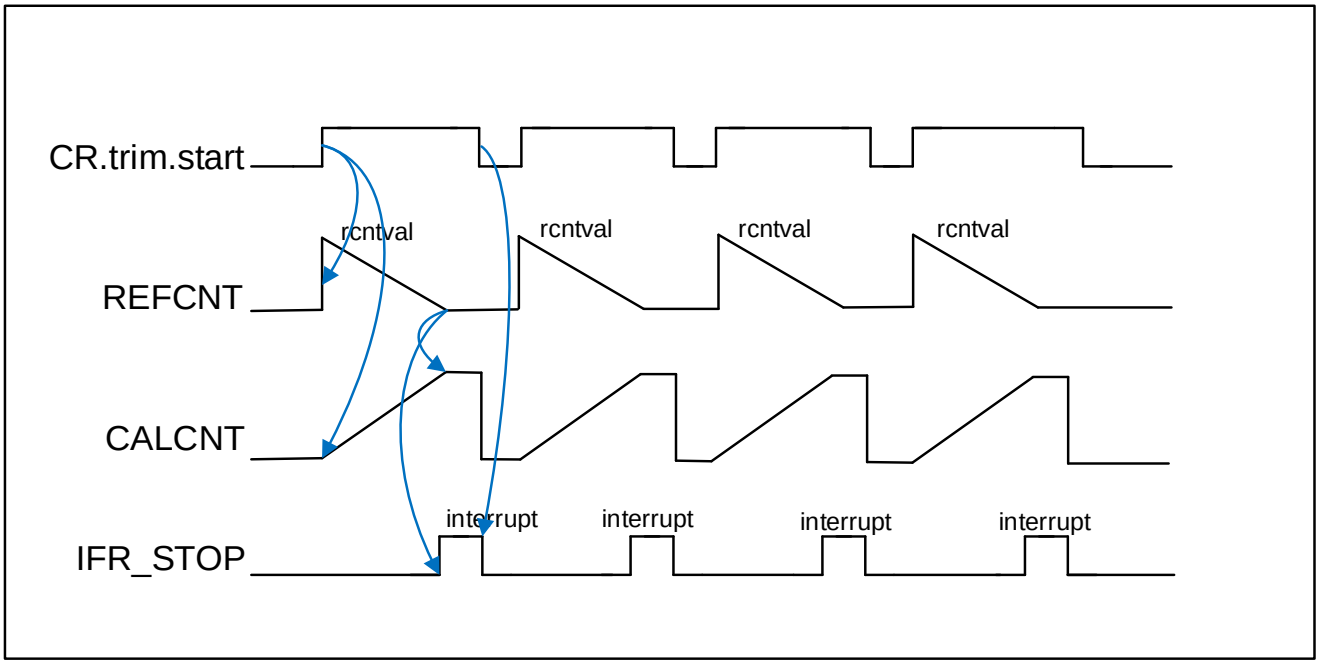


图 8-3 时钟校准波形示意图

注意：
校准模式在校准过程中有可能因为校准时间设置过长，发生待校准时钟计数器在 CLKTRIM_IFR.STOP 置 1 之前溢出的情况，CLKTRIM_IFR.CALCNT_OVF 置 1，触发中断。中断服务子程序发现 CLKTRIM_IFR.CALCNT_OVF 置 1 时，清零 CLKTRIM_CR.TRIM_START 寄存器结束校准。
这种情况下校准是无法正确进行的，必须调整校准时间，重新校准。
具体步骤是：
1.设置 CLKTRIM_REFCON.RCNTVAL 寄存器调整校准时间。
2.设置 CLKTRIM_CR.TRIM_START 寄存器重新开始校准。

8.3.2 Clock monitor 时钟监测

Clock 监测模式分为，硬件监测 HXT 和软件监测模式。硬件监测的话，只支持监测 HXT clock(不支持从外部输入时钟的检测)。
硬件监测，只需要一个 clock 动作，其他时钟源可以不动作。
软件监测，必须要使用 2 个以上的时钟同时动作，一个作为被监测时钟，一个作为参考时钟。

8.3.2.1 Clock monitor 硬监测 HXT 软件设置流程

该监测模式，直接监测的 HXT 的输出频率，该频率一旦降低到 200KHz 以下时，会产生一个中断信号，同时将系统 clock 自动切换到 HIRC 去。

该监测模式不能与以下软监测模式同时使用。

1. 设置 RCC_HXTUNLOCK 寄存器的 bit1 为 1,使其对 HXTCR 寄存器里对应位的保护无效。
2. 设置 RCC_HXTCR 寄存器的 bit12 为 0，使能 HXT 的硬件停振检测功能有效。
3. 设置 RCC_SYSCLKSEL 寄存器,将系统时钟选为 HXT。
4. 设置 RCC_SYSCLKCR 寄存器的 CLKFAILEN 为 1(有效)。
5. 设置 RCC_SYSCLKCR 寄存器的 HIRCEN 为 1(使能)。
6. 设置 CLKTRIM_CR 寄存器的 DET_EN 位为 1(监测使能)。
7. 设置 CLKTRIM_CR 寄存器的 RST_IE 位为 0(中断使能)。如果 RST_IE=1,则系统发生复位。
8. 设置对应的中断有效 NVIC。
9. 一旦 HXT 的发振频率低于 200KHz 时，会发生停振检知事件。
10. 如果 CR.RST_IE 设为 1，则发生复位，同时 RCC_RSTSR.CMU_LOC 位被置为 1。
11. 如果 CR.RST_IE 设为 0，系统时钟将自动切换到 HIRC 时钟，同时关闭 HXT 时钟
12. 进入中断子程序。

8.3.2.2 Clock monitor 软监测模式

该监测模式主要用于选择一个稳定的时钟源作为参考时钟，在设定的时间周期下监测系统工作时钟的异常状态。

在此监测模式下若监测对象是 HXT 或者 LXT 的话，一旦发生时钟异常，系统时钟会自动切换到 HIRC 时钟。

只能选择内部高速 RC 时钟(HIRC)或者由内部高速 RC 时钟产生的 PLL 时钟作为被监测时钟。

为了监测系统工作时钟(被监测时钟 CAL_CLK)是否正常工作时，需要一个稳定的时钟作为参考时钟(REF_CLK)。设定监测时间 Ttrim，被监测时钟频率 Fcal,参考时钟频率 Fref,时钟监测上限阈值和下限阈值，使用被监测时钟和参考时钟同时计数，在监测时间结束时停止计数，判断被监测时钟计数器是否处于溢出状态。如果被监测时钟计数器的值在 CLKTRIM_HTCR 和 CLKTRIM_LTCR 之间，表示在监测时间内被监测时钟工作正常，继续进行下一次监测。如果被监测时钟计数器的值高于 CLKTRIM_HTCR 的值或者低于 CLKTRIM_LTCR 的值，表示在监测时间内被监测时钟工作异常，可能被监测时钟已经停止或者频率大幅变化，产生被监测时钟工作异常中断，或者系统复位，由硬件切换系统工作时钟。

注意 1:被监测时钟和参考时钟不能是同一个时钟，并且参考时钟一定是一个稳定有效的时钟。

注意 2:因为进入 deepsleep 模式时，主振会自动停止，为了不发生意外的复位，请进去 deepsleep 模式时，一定先把 clock 监视模式停止。

8.3.2.3 Clock monitor 软监测软件设置流程

1 设置 CLKTRIM_CR.REFCLK_SEL 选择参考时钟。
2 设置 CLKTRIM_CR.CALCLK_SEL 选择被监测时钟。
3 设置 CLKTRIM_CR.CLKEN 使能被监测时钟和参考时钟。
4 设置 CLKTRIM_REFCON.RCNTVAL 寄存器来设定监测间隔时间。
5 设置 CLKTRIM_HTCR 寄存器来设定被监控时钟计数器上限阈值比较时间。
6 设置 CLKTRIM_LTCR 寄存器来设定被监控时钟计数器下限阈值比较时间。
7 设置 CLKTRIM_CR.MON_EN 寄存器使能监控功能。
8 根据需要设定 CLKTRIM_CR.RST_IE 位

- 如果被监控时钟是系统时钟，则一定设定 RST_IE=1，因为这个时候系统时钟失效，系统自动切换时钟，会死锁，只能产生复位来保证系统的稳定性。
- 如果被监测时钟不是系统时钟，则 RST_IE 可以不设为 1。

9 设置 CLKTRIM_CR.TRIM_START 开始时钟监测模块动作开始。
10 参考时钟和被监控时钟同时开始计数。
11 当参考时钟计数器计数到达监控间隔时间时，判断被监控时钟计数器是否溢出。如果溢出表示被监控时钟工作正常。如果没有溢出表示被监控时钟失效，CLKTRIM_IFR.HIRC_FAULT/PLL_FAULT 置 1，触发中断，或者复位。

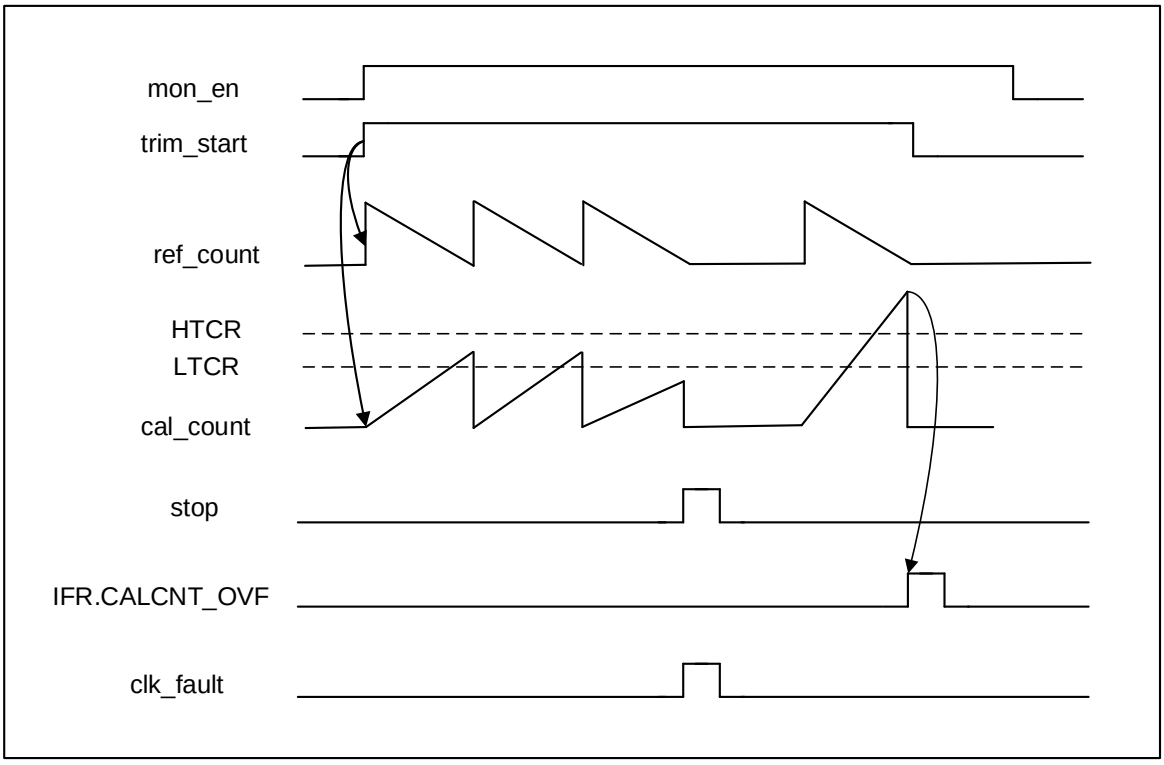


图 8-4 波形示意图

8.4 时钟安全(clock monitor)寄存器描述

基地址：0x40003400

偏移地址	名称	寄存器描述	复位值
0X00	CLKTRIM_CR	配置寄存器	0x00000000
0X04	CLKTRIM_REFCON	参考计数器初值配置寄存器	0xFFFFFFFF
0X08	CLKTRIM_REFCNT	参考计数器值寄存器	0xFFFFFFFF
0X0C	CLKTRIM_CALCNT	校准计数器值寄存器	0x00000000
0X10	CLKTRIM_IFR	中断标志位寄存器	0x00000000
0X14	CLKTRIM_ICLR	中断标志位清除寄存器	0x00000000
0x1C	CLKTRIM_HTCR	上限阈值比较寄存器	0xFFFFFFFF
0x20	CLKTRIM_LTCR	下限阈值比较寄存器	0x00000000

8.5 寄存器描述

8.5.1 配置寄存器(CLKTRIM_CR)

偏移地址：0x00

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留					DET_EN	CLKEN	RST_IE	MONEN	CALSEL			REFCLK_SEL			TRIM_START
					R/W	R/W	R/W	R/W	R/W			R/W			R/W

位	标记	功能描述	复位值	读写
31:11	Res	保留	0x0	-
10	DET_EN	HXT硬监测功能使能 0：禁止 1：硬监测功能使能	0x0	R/W
9	CLKEN	参考时钟和待监测时钟使能 0：禁止 1：时钟使能	0x0	R/W
8	RST_IE	复位/中断使能寄存器 0：发生停振检知时产生中断 1：发生停振检知时产生复位	0x0	R/W
7	MON_EN	监视模式使能寄存器 0：禁止 1：使能	0x0	R/W
6:4	CALCLK_SEL	待监测时钟源选择寄存器 000：HIRC 001：HXT 010：LIRC 011：LXT 100：PLL 以上以外：保留	0x0	R/W
3:1	REFCLK_SEL	参考时钟选择寄存器 000：HIRC 001：HXT 010：LIRC 011：LXT 100：OSC_IN 101：PLL 以上以外：保留	0x0	R/W
0	TRIM_START	软监测开始寄存器 0：停止 1：开始	0x0	R/W

注意:因为进入 deepsleep 模式时，主振会自动停止，为了不发生意外的复位/中断等动作，请进入低消费模式时，一定先把 clock 监视模式停止。

8.5.2 参考计数器初值配置寄存器(CLKTRIM_REFCON)

偏移地址：0x04

复位值：0xFFFF FFFF

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

位	标记	功能描述	复位值	读写
31:0	RCNTVAL	参考计数器值	0xFFFFFFFF	R/W

8.5.3 参考计数器值寄存器(CLKTRIM_REFCNT)

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

位	标记	功能描述	复位值	读写
31:0	REFCNT	参考计数器值 读该寄存器需要先打开时钟使能，当 TRIM_START 有效后，写入的初始值就会更新到该寄存器。 因为 REFCLK 和 PCLK 是异步的，所以在 CLK_TRIM 模块动作的时候，请不要去读该寄存器。 只有当 IFR.STOP 置 1 时，才能去读该寄存器。	0xFFFFFFFF	RO

8.5.4 校准计数器值寄存器(CLKTRIM_CALCNT)

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

位	标记	功能描述	复位值	读写
31:0	CALCNT	被校准计数器值。 因为 CALCLK 和 PCLK 是异步的，所以在 CLK_TRIM 模块动作的时候，请不要去读该寄存器，只有当 IFR.STOP 置 1 时，才能去读该寄存器。	0x0	RO

8.5.5 中断标志位寄存器(CLKTRIM_IFR)

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
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留									DETECT_FAULT	HIRC_FAULT	PLL_FAULT	HXT_FAULT	LXT_FAULT	CALCNT_OVF	STOP
									RO	RO	RO	RO	RO	RO	RO

位	标记	功能描述	复位值	读写
31:8	Res	保留位	0x0	-
7	DETECT_FAULT	HXT 硬监测停振检知标志 CLKTRIM_ICLR.DET_FAULT_CLR 写 1 清除此标志位	0x0	RO
6	HIRC_FAULT	HIRC 失效标志 CLKTRIM_ICLR.HIRC_FAULT_CLR 写 1 清除此标志位	0x0	RO
5	LIRC_FAULT	LIRC 失效标志 CLKTRIM_ICLR.LIRC_FAULT_CLR 写 1 清除此标志位	0x0	RO
4	PLL_FAULT	PLL 失效标志 CLKTRIM_ICLR.PLL_FAULT_CLR 写 1 清除此标志位	0x0	RO
3	HXT_FAULT	HXT 失效标志 CLKTRIM_ICLR.HXT_FAULT_CLR 写 1 清除此标志位	0x0	RO
2	LXT_FAULT	LXT 失效标志 CLKTRIM_ICLR.LXT_FAULT_CLR 写 1 清除此标志位	0x0	RO
1	CALCNT_OVF	校准计数器溢出标志 CLKTRIM_CR.START 写 1 清除此标志位	0x0	RO
0	STOP	参考计数器停止标志 CLKTRIM_CR.START 写 1 清除此标志位	0x0	RO

8.5.6 中断标志位清除寄存器(CLKTRIM_ICLR)

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
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留									DET_FAULT_CLR	HIRC_FAULT_CLR	PLL_FAULT_CLR	HXT_FAULT_CLR	LXT_FAULT_CLR	保留	
									WO	WO	WO	WO	WO		

位	标记	功能描述	复位值	读写
31:8	Res	保留位	0x0	-
7	DET_FAULT_CLR	清除硬监测停振检知标志，写 1 清除	0x0	WO
6	HIRC_FAULT_CLR	清除 HIRC_FAULT 失效标志，写 1 清除	0x0	WO
5	LIRC_FAULT_CLR	清除 LIRC_FAULT 失效标志，写 1 清除	0x0	WO
4	PLL_FAULT_CLR	清除 PLL_FAULT 失效标志，写 1 清除	0x0	WO
3	HXT_FAULT_CLR	清除 HXT_FAULT 失效标志，写 1 清除	0x0	WO
2	LXT_FAULT_CLR	清除 LXT_FAULT 失效标志，写 1 清除	0x0	WO
1:0	Res	保留位	0x0	-

8.5.7 上限阈值寄存器(CLKTRIM_HTCR)

偏移地址：0x1C

复位值：0xFFFF FFFF

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

位	标记	功能描述	复位值	读写
31:0	HTCR	上限阈值寄存器 如果 CALCNT 的计数值大于这个寄存器的设定值， 那么时钟监测模块会产生中断或复位请求	0x0	R/W

8.5.8 下限阈值寄存器(CLKTRIM_LTCR)

偏移地址：0x20

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:0	LTCR	下限阈值寄存器 如果 CALCNT 的计数值大于这个寄存器的设定值， 那么时钟监测模块会产生中断或复位请求	0x0	R/W

9 系统控制(SYSCON)

本产品具有一组系统配置寄存器，系统配置控制器的主要用途如下：

- LINFlexD 控制位
- 配置 GPIO 管脚的中断产生模式
- 重映射 PCA、TIM10/TIM11、TIM1、TIM2 的输入触发源
- SPI0/1 从模式的 SSN 输入重映射设定
- 系统级的 Deep Sleep 调试以及 Lockup 复位控制设定
- NMI 控制位
- PE6/NRST 外部端子输入或 GPIO 选择控制位

9.1 寄存器列表

SYSCON 基址：0x40001C00

表 9-1 SYSCON 寄存器列表和复位值

偏移地址	名称	描述	默认值
0x0	SYSCON_CFGR0	系统配置寄存器 0	0x0000 0A00
0x4	SYSCON_PORTINTCR	端子的中断模式设定	0x0000 0000
0x8	SYSCON_PORTCR	端子控制寄存器	0x0000 0000
0xC	SYSCON_PCACR	PCA 捕获通道来源选择	0x0000 0000
0x10	SYSCON_TIM1CR	TIM1 通道输入源选择	0x0000 0000
0x14	SYSCON_TIM2CR	TIM2 通道输入源选择	0x0000 0000
0x20	SYSCON_RSTCTRL	外部复位控制寄存器	0x0000 000X
0x30	SYSCON_NMICR	NMI 控制寄存器	0x0000 0000
0x34	SYSCON_NMISR	NMI 标志寄存器	0x0000 0000
0x50	SYSCON_UNLOCK	SYSCON 寄存器写保护	0x0000 0000

9.2 寄存器说明

9.2.1 系统配置寄存器 0(SYSCON_CFGR0)

地址偏移：0x00

复位值：0x00000A00

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
WO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				LIN1_RD_TYP		LIN0_RD_TYP		LIN1_STOP_ACK	LIN0_STOP_ACK	LIN1_STOP	LIN0_STOP	保留	IWDT_DSLP_STOP	DBGDSLP_DIS	LOCKUP_EN
				R/W		R/W		RO	RO	R/W	R/W		R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	KEY	仅高位写0x5A69时配置该寄存器才有效，写其它值时无效	0x0	WO
15:12	Res	保留	0x0	-
11:10	LIN1_RD_TYP	00：Byte 01：Half word 10：Word	0x2	R/W
9:8	LIN0_RD_TYP	00：Byte 01：Half word 10：Word	0x2	R/W
7	LIN1_STOP_ACK	LIN1 STOP ACK状态反馈 0：LIN1未进入STOP模式 1：LIN1已经进入STOP模式	0x0	RO
6	LIN0_STOP_ACK	LIN0 STOP ACK状态反馈 0：LIN0未进入STOP模式 1：LIN0已经进入STOP模式	0x0	RO
5	LIN1_STOP	LIN1 Stop请求控制 0：LIN 1 Stop请求无效 1：LIN 1 Stop请求有效	0x0	R/W
4	LIN0_STOP	LIN0 Stop请求控制 0：LIN 0 Stop请求无效 1：LIN 0 Stop请求有效	0x0	R/W
3	Res	保留	0x0	-
2	IWDT_DSLP_STOP	IWDT在Deepsleep模式停止计数 0：IWDT在DeepSleep模式保持计数 1：IWDT在DeepSleep模式停止计数	0x0	R/W
1	DBGDSLP_DIS	Debug 模式，进入 Deep Sleep 模式禁止控制位 0：允许在 Debug 模式进入 Deep Sleep 1：不允许在 Debug 模式进入 Deep Sleep	0x0	R/W
0	LOCKUP_EN	Cortex-M0 LookUp 功能使能 0：关闭 1：使能 注：当 Cortex-M0 读取错误的指令时，MCU 会复位，以增强系统可靠性。	0x0	R/W

注意，只有 SYSCON_UNLOCK 寄存器保护解除后，才能写该寄存器。

9.2.2 管脚 Deep Sleep 中断模式控制寄存器(SYSCON_PORTINTCR)

地址偏移：0x04

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	KEY	只有高位写0x5A69时配置该寄存器才有效，写其它值时无效。	0x0	WO
15:2	Res	保留	0x0	–
1	PADDSLPCON	0：当进入Deep Sleep后，PAD的中断产生模式自动切换到Deep Sleep中断产生模式(没有Debounce功能) 1：当进入DeepSleep后，PAD的中断产生模式不会自动切换， 由SYSCON_PORTINTSEL.INTSEL位决定中断产生模式。	0x0	R/W
0	PADINTSEL	端口中断模式选择 0：ACTIVE/Sleep中断产生模式 1：Deep Sleep中断产生模式	0x0	R/W

注意：

- 选用Deep Sleep中断产生模式时，GPIO管脚的Debounce功能需要关闭，抗干扰能力很差，一般推荐用户在芯片需要使用管脚中断唤醒Deep Sleep模式才选择此模式
- 只有SYSCON_UNLOCK寄存器保护解除后，才能写该寄存器。

9.2.3 管脚控制寄存器(SYSCON_PORTCR)

地址偏移：0x08

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留													SPI0SSN_SEL		
													R/W		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LPTIM_EXT_SEL			LPTIM_GATE_SEL			TIM11_GATE_SEL			TIM10_GATE_SEL			SPI1SSN_SEL			
R/W			R/W			R/W			R/W			R/W			

位	标记	功能描述	复位值	读写
31:19	Res	保留	0x0	–
18:16	SPI0SSN_SEL	SPI0 SLAVE模式SSN信号来源选择 000：固定高电平 001：PB0 010：PB5 011：PC13 100：PC15 101~111：固定高电平 注：从机选择GPIO口时需将GPIO配置成输入模式	0x0	R/W
15:13	LPTIM_EXT_SEL	Low Power Timer EXT输入信号来源选择 000：LPTIM_EXT 001：VC的输出 其他：保留	0x0	R/W
12:10	LPTIM_GATE_SEL	Low Power Timer门控输入信号来源选择 000：LPTIM_GATE 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：VC的输出 其他：保留	0x0	R/W
9:7	TIM11_GATE_SEL	TIM11门控输入信号来源选择 000：TIM11_GATE 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：VC的输出 其他：保留	0x0	R/W
6:4	TIM10_GATE_SEL	TIM10门控输入信号来源选择 000：TIM10_GATE 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：VC的输出 其他：保留	0x0	R/W

3:0	SPI1SSN_SEL	SPI1 SLAVE模式SSN信号来源选择 0000：固定高电平 0001：PA6 0010：PA11 0011：PB10 0100：PC5 0101：PC6 0110：PC11 0111：PD3 1000：PE0 1001~1111：固定高电平 注：从机选择GPIO口时需将GPIO配置成输入模式	0x0	R/W

9.2.4 PCA 捕获通道控制寄存器(SYSCON_PCACR)

地址偏移：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留														PCA_ECI_SEL	
														R/W	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCA_ECI_SEL	PCA_CAP4_SEL			PCA_CAP3_SEL			PCA_CAP2_SEL			PCA_CAP1_SEL			PCA_CAP0_SEL		
R/W	R/W			R/W			R/W			R/W			R/W		

位	标记	功能描述	复位值	读写
31:18	Res	保留	0x0	–
17:15	PCA_ECI_SEL	PCA ECI信号来源选择 000：PCA_ECI 001：VC输出 其他：保留	0x0	R/W
14:12	PCA_CAP4_SEL	PCA 捕获通道4信号来源选择 000：PCA_CH4 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：VC输出 其他：保留	0x0	R/W
11:9	PCA_CAP3_SEL	PCA捕获通道3信号来源选择 000：PCA_CH3 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：VC输出 其他：保留	0x0	R/W
8:6	PCA_CAP2_SEL	PCA 捕获通道2信号来源选择 000：PCA_CH2 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：VC输出 其他：保留	0x0	R/W
5:3	PCA_CAP1_SEL	PCA 捕获通道1信号来源选择 000：PCA_CH1 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：VC输出 其他：保留	0x0	R/W
2:0	PCA_CAP0_SEL	PCA 捕获通道0信号来源选择 000：PCA_CH0	0x0	R/W

		001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：VC输出 其他：保留		

9.2.5 TIM1 通道输入源选择(SYSCON_TIM1CR)

地址偏移：0x10

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留									CLKFAILBRKEN	DSLPBRKEN	TIM1BRKOUTCFG	TIM1ETR_SEL			
									R/W	R/W	R/W	R/W			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	TIM1CH4IN_SEL			保留	TIM1CH3IN_SEL			保留	TIM1CH2IN_SEL			保留	TIM1CH1IN_SEL		
	R/W				R/W				R/W				R/W		

位	标记	功能描述	复位值	读写
31:23	Res	保留	0x0	–
22	CLKFAILBRKEN	时钟失效事件产生TIM1刹车事件使能 0：无效 1：使能	0x0	R/W
21	DSLPBRKEN	进入Deep Sleep模式产生TIM1刹车事件使能 0：无效 1：使能	0x0	R/W
20	TIM1BRKOUTCFG	TIM1输出使能控制位 0：无影响 1：输出使能OcxE/OcxNE强制为0	0x0	R/W
19:16	TIM1ETR_SEL	TIM1 ETR信号来源选择 0000：固定低电平 0001：PA1 0010：PA2 0011：PA3 0100：PB4 0101：PB5 0110：PC3 0111：PC4 1000：PC5 1001：PC6 1010：PC7 1011：PD1 1100：PD2 1101：PD3 1110：PD4 1111：PD6	0x0	R/W

15	Res	保留	0x0	–
14:12	TIM1CH4IN_SEL	TIM1 CH4输入通道信号来源选择 000：TIM1_CH4 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：LIRC 101：VC的输出 其他：保留	0x0	R/W
11	Res	保留	0x0	–
10:8	TIM1CH3IN_SEL	TIM1 CH3输入通道信号来源选择 000：TIM1_CH3 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：LIRC 101：VC的输出 其他：保留	0x0	R/W
7	Res	保留	0x0	–
6:4	TIM1CH2IN_SEL	TIM1 CH2输入通道信号来源选择 000：TIM1_CH2 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：LIRC 101：VC的输出 其他：保留	0x0	R/W
3	Res	保留	0x0	–
2:0	TIM1CH1IN_SEL	TIM1 CH1输入通道信号来源选择 000：TIM1_CH1 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：LIRC 101：VC的输出 其他：保留	0x0	R/W

9.2.6 TIM2 通道输入源选择(SYSCON_TIM2CR)

地址偏移：0x14

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留									CLKFAILBRKEN	DSLPBRKEN	TIM2BRKOUTCFG	TIM2ETR_SEL			
									R/W	R/W	R/W	R/W			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	TIM2CH4IN_SEL			保留	TIM2CH3IN_SEL			保留	TIM2CH2IN_SEL			保留	TIM2CH1IN_SEL		
	R/W				R/W				R/W				R/W		

位	标记	功能描述	复位值	读写
31:23	Res	保留	0x0	–
22	CLKFAILBRKEN	时钟失效事件产生TIM2刹车事件使能 0：无效 1：使能	0x0	R/W
21	DSLPBRKEN	进入Deep Sleep模式产生TIM2刹车事件使能 0：无效 1：使能	0x0	R/W
20	TIM2BRKOUTCFG	TIM2输出使能控制位 0：无影响 1：输出使能OcxE/OcxNE强制为0	0x0	R/W
19:16	TIM2ETR_SEL	TIM2 ETR信号来源选择 0000：固定低电平 0001：PA1 0010：PA2 0011：PA3 0100：PB4 0101：PB5 0110：PC3 0111：PC4 1000：PC5 1001：PC6 1010：PC7 1011：PD1 1100：PD2 1101：PD3 1110：PD4 1111：PD6	0x0	R/W
15	Res	保留	0x0	–
14:12	TIM2CH4IN_SEL	TIM2 CH4输入通道信号来源选择 000：TIM2_CH4 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：LIRC	0x0	R/W

		101：VC的输出 其他：保留		
11	Res	保留	0x0	–
10:8	TIM2CH3IN_SEL	TIM2 CH3输入通道信号来源选择 000：TIM2_CH3 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：LIRC 101：VC的输出 其他：保留	0x0	R/W
7	Res	保留	0x0	–
6:4	TIM2CH2IN_SEL	TIM2 CH2输入通道信号来源选择 000：TIM2_CH2 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：LIRC 101：VC的输出 其他：保留	0x0	R/W
3	Res	保留	0x0	–
2:0	TIM2CH1IN_SEL	TIM2 CH1输入通道信号来源选择 000：TIM2_CH1 001：UART0_RXD 010：UART1_RXD 011：LPUART_RXD 100：LIRC 101：VC的输出 其他：保留	0x0	R/W

9.2.7 外部复位控制寄存器(SYSCON_RSTCTRL)

地址偏移：0x20

复位值：0x0000000X (X 为用户设定值)

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

位	标记	功能描述	复位值	读写
31:16	KEY	只有高位写0x5A69时配置该寄存器才有效，其它值无效	0x0	WO
15:1	Res	保留	0x0	–
0	RSTPAD_DIS	芯片外部复位控制 0：使能PE6端口的外部复位功能 1：关闭PE6端口的外部复位功能 该位的初始值从Option Byte地址(0x1FFF F802)的USER第7位加载过来，详细参考5.6.8FLASH_OBR(选项字节寄存器)。	0xX	R/W

注：只有SYSCON_UNLOCK寄存器保护接触后，才能写该寄存器。

9.2.8 NMI 中断控制寄存器(SYSCON_NMICR)

地址偏移：0x30

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
WO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留										NMI_FLTDIV		保留	NMI_FLTEN	NMI_CKSEL	NMI_EN
										R/W			R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	KEY	写0x5A69时配置该寄存器有效，其他值无效	0x0	WO
15:6	Res	保留	0x0	-
5:4	NMI_FLTDIV	NMI滤波分频 00：1分频 01：2分频 10：4分频 11：8分频	0x0	R/W
3	Res	保留	0x0	–
2	NMI_FLTEN	NMI滤波使能 0：不使能 1：使能滤波	0x0	R/W
1	NMI_CKSEL	NMI滤波时钟选择 0：PCLK 1：LIRC	0x0	R/W
0	NMI_EN	NMI使能控制 0：不使能 1：NMI使能(使能后无法关闭)	0x0	R/W

注：只有 SYSCON_UNLOCK 寄存器保护解除后，才能写该寄存器。

9.2.9 NMI 中断状态寄存器(SYSCON_NMISR)

地址偏移：0x34

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	–
0	NMI_INTF	NMI 信号中断标志，硬件置位，软件写 0 清除 0：没有 NMI 中断标志 1：有 NMI 中断标志	0x0	R/W

9.2.10 SYSCON 寄存器写保护(SYSCON_UNLOCK)

地址偏移：0x50

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:1	KEY	只有高位写0x2AD5334C时配置该寄存器才有效，写其它值时无效。	0x0	WO
0	UNLOCK	0：保护有效 1：解除保护	0x0	R/W

注：写 0x55AA6699，解除保护。

10 中断控制器(NVIC)

10.1 概述

Cortex®-M0+提供中断控制器，用于总体管理异常，称之为“嵌套向量中断控制器(NVIC)”。NVIC 和处理器内核紧密相连，可以实现低延迟的中断处理和高效地处理晚到的中断。

支持最多 32 个中断请求(IRQ)输入，以及 1 个不可屏蔽中断(NMI)输入。另外，处理器还支持多个内部异常。

本章节只对处理器的 32 个外部中断请求(中断 0 到中断 31)做详细介绍，处理器内部异常的具体情况可参考其他相关文档。详情请参考“ARM® Cortex®-M0+Technical Reference Manual”与“ARM® v6-M Architecture Reference Manual”。

10.2 特征

- 32 个可屏蔽中断通道
- 4 个可编程的优先级(使用了 2 位的中断优先级)
- 低延时的异常和中断处理
- 电源管理控制
- 系统控制寄存器的实现
- 支持嵌套和向量中断
- 动态改变优先级

10.3 中断优先级

软件可以对外部中断设置 4 级优先级。最高优先级为“0”，最低优先级为“所有用户可配置的优先级的默认值为“0”。

如果处理器已经在运行另外一个中断处理，而新中断的优先级大于正在执行的，这时就会发生抢占。正在运行的中断处理会被暂停，转而执行新的中断，这个过程通常被称为中断嵌套。新的中断执行完毕后，之前的中断处理会继续执行，并且在其结束后返回到程序线程中。

如果处理器正在运行的另外一个中断处理的优先级相同或者更高，新的中断将会等待并且进入挂起状态。挂起的中断将会一直等到当前中断等级改变，例如，当前运行的中断处理完成返回后，当前优先级降低到了比挂起中断还要小。

如果两个中断同时发生，并且它们的优先级相同，中断编号较小的中断将会首先执行。例如，如果中断 0 和中断 1 使能且具有相同的优先级，在它们同时被触发时，中断 0 会首先执行。

10.4 中断向量表

Cortex®-M0+响应中断时，处理器自动从存储器的中断向量表中取出中断服务例程(ISR)的起始地址。向量表包括复位后堆栈(MSP)的初始值以及所有异常处理的入口地址。向量号表示处理异常的先后次序。其中，中断向量的存储顺序同编号一致由于每个都是 1 个字(4 字节)，中断向量的地址为中断编号乘以 4，每个中断向量都是处理的起始地址。

表 10-1 中断向量表

中断号	优先级	优先级类型	中断源	简介	地址
	-	-	-	MSP初始值	0x0000 0000
	-3	固定	Reset	复位向量(RESET)	0x0000 0004
	-2	固定	NMI	不可屏蔽中断	0x0000 0008
	-1	固定	HardFault	所有类型的错误(Fault)	0x0000 000C
	-	-	-	保留	0x0000 000C – 0x0000 002B
	3	固定	SVCall	通用SWI指令调用的系统服务	0x0000 002C
	-	-	-	保留	0x0000 0030 – 0x0000 0037
	5	固定	PendSV	可挂起的系统服务	0x0000 0038
	6	固定	SysTick	系统嘀嗒定时器	0x0000 003C
0		可配置	GPIO_PA	GPIOA中断	0x0000 0040
1		可配置	GPIO_PB	GPIOB中断	0x0000 0044
2		可配置	GPIO_PC	GPIOC中断	0x0000 0048
3		可配置	GPIO_PD	GPIOD中断	0x0000 004C
4		可配置	FLASH	FLASH中断	0x0000 0050
5		可配置	MATH	MATH中断	0x0000 0054
6		可配置	UART0	UART0中断	0x0000 0058
7		可配置	UART1	UART1中断	0x0000 005C
8		可配置	LPUART	LPUART中断	0x0000 0060
9		可配置	CAN	CAN中断-	0x0000 0064
10		可配置	SPI0	SPI0中断	0x0000 0068
11		可配置	SPI1	SPI1中断	0x0000 006C
12		可配置	I2C0	I2C0中断	0x0000 0070
13		可配置	I2C1	I2C1中断	0x0000 006C
14		可配置	TIM10	TIM10中断	0x0000 0078
15		可配置	TIM11	TIM11中断	0x0000 007C
16		可配置	LPTIM	LPTIM中断	0x0000 0080
17		可配置	GPIO_PE	GPIOE中断	0x0000 007C
18		可配置	TIM1	TIM1中断	0x0000 0088
19		可配置	TIM2	TIM2中断	0x0000 008C
20		可配置	LIN0	LIN0中断	0x0000 0088
21		可配置	LIN1	LIN1中断	0x0000 0094
22		可配置	WWDG	WWDG中断	0x0000 0098
23		可配置	IWDG	IWDG中断	0x0000 009C
24		可配置	ADC	ADC中断	0x0000 00A0
25		可配置	LVD	LVD中断	0x0000 00A4
26		可配置	VC	VC中断	0x0000 00A8
27		可配置	SRAM/FLASH	SRAM/FLASH_ECC校验错误中断	0x0000 00A4
28		可配置	AWK/1-Wire	AWK/1-Wire中断	0x0000 00B0
29		可配置	PCA	PCA中断	0x0000 00B4
30		可配置	RTC	RTC中断	0x0000 00B8
31		可配置	CLKTRIM	CLKTRIM中断	0x0000 00BC

10.5 中断唤醒控制 WIC

当处理器利用退出休眠(Sleep-on-exit)特性或者执行 WFI 指令进入休眠之后，就会停止指令执行，并且当发生了中断请求(更高优先级)并且需要处理时，处理器就会被唤醒。

WFI行为	唤醒	ISR执行
PRIMASK清除	Y	Y
IRQ优先级>当前等级	N	N
IRQ优先级≦当前等级		
PRIMASK置位(中断禁止)		
IRQ优先级>当前等级	Y	N
IRQ优先级≦当前等级	N	N

10.5.1 NVIC 从深度休眠模式唤醒设置进入中断 ISR 设置

1. 使能深度休眠需要唤醒的中断 NVIC
2. 使能模块中断使能
3. 设置 SCR.SleepDEEP 为 1
4. 使用 WFI 指令进入深度休眠模式
5. 系统进入深度休眠模式等待中断唤醒，唤醒后执行中断服务子程序

例程：

```
SCR |= 0x00000004UL;
__asm(“nop”);
__asm(“nop”);
__asm(“nop”);
while(1)
{
    __asm(“WFI”);
}
```

10.5.2 NVIC 从深度休眠模式唤醒设置不执行中断 ISR 设置

1. 使能深度休眠需要唤醒的中断 NVIC
2. 使用 PRIMASK 寄存器屏蔽中断
3. 使能模块中断使能
4. 设置 SCR.SleepDEEP 为 1
5. 使用 WFI 指令进入深度休眠模式
6. 系统进入深度休眠模式等待中断唤醒，唤醒后执行下一条指令
7. 清除中断标志，清除中断挂起状态

10.5.3 使用退出休眠特性

退出休眠(Sleep-On-Exit)非常适合中断驱动的应用程序。当该特性使能时，只要完成异常处理并且返回到了线程模式，处理器就会进入休眠模式。利用退出休眠特性，处理器可以尽可能多的处于休眠模式。

Cortex®-M0+利用退出休眠特性进入休眠，这种情况同执行完异常退出后立即执行 WFI 的效果差不多。不过，为了下次进入异常时，不用再进行压栈操作，处理器不会执行出栈的过程。

1. 使能深度休眠需要唤醒的中断 NVIC
2. 使能模块中断使能
3. 设置 SCR.SleepDEEP 为 1
4. 设置 SCR.SleepONEXIT 为 1
5. 使用 WFI 指令进入深度休眠模式
6. 系统进入深度休眠模式等待中断唤醒，唤醒后执行中断服务子程序
7. 退出中断服务时自动进入休眠模式

例程：

```
SCR |= 0x00000004UL;
SCR |= 0x00000002UL;
__asm(“nop”);
__asm(“nop”);
__asm(“nop”);
while(1)
{
    __asm(“WFI”);
}
```

11 通用输入输出口(GPIO)

11.1 GPIO 简介

通用输入/输出口用于芯片和外部进行数据传输，共有 5 组 GPIO，GPIOA、GPIOB、GPIOC、GPIOD 和 GPIOE，可以通过配置将 GPIO 映射到对应芯片引脚。GPIO 的功能基本相同，每个引脚可以被独立配置为数字输入或者数字输出口。另外还可配置为如模拟输入，外部中断，片上外设的输入/输出等复用功能。但是在同一时刻仅有一个复用功能可以映射到引脚上。复用功能的映射是通过端口复用功能寄存器(GPIOx_AFR)控制的。

每个通用 I/O 口都有 5 个配置寄存器(GPIOx_DIRCR，GPIOx_OTYPER，GPIOx_PUPDR，GPIOx_SLEWCR 和 GPIOx_DRVCR)，2 个数据寄存器(GPIOx_IDR 和 GPIOx_ODR)，1 个输出置位寄存器(GPIOx_ODSET)，1 个输出复位寄存器(GPIOx_ODCLR)和 2 个复用功能寄存器(GPIOx_AFR1/2)。

每个端口都可以配置成内部拉高(Pull up)/拉低(Pull down)的输入/输出，浮空输入(Floating input)，推挽输出(Push-pull output)，开漏输出(open drain output)，增强驱动能力输出。芯片复位后端口为模拟模式，目的是防止芯片被异常复位时，外部器件产生异常动作。端口被配成模拟端口后，数字功能被隔离，不能输出数字“1”和“0”，CPU 读取端口的结果为“0”。

所有端口都可以提供外部中断，并且每个中断都可以配置成高电平触发、低电平触发、上升沿触发、下降沿触发或者任意边沿触发，支持边沿模式下的输入消抖。支持工作模式/睡眠模式/深度睡眠模式下产生中断。

11.2 GPIO 主要特性

- 输出状态：带有上拉或下拉的推挽输出或开漏输出
- 从数据寄存器(GPIOx_ODR)或外设(复用功能输出)输出数据
- 可配的每个 I/O 口的速度
- 输入状态：浮空、上拉/下拉、模拟输入
- 从数据寄存器(GPIOx_IDR)或外设输入数据(复用功能输出)
- 输出置位/复位寄存器(GPIOx_ODSET，GPIOx_ODCLR)对 GPIOx_ODR 寄存器具有按位写权限
- 模拟功能引脚/调试引脚/数字通用引脚/数字功能引脚复用
- 允许 GPIO 口和外设引脚的高灵活性复用

11.3 GPIO 功能描述

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：

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

每个 I/O 端口位可以自由编程，I/O 端口寄存器可按 32 位字访问。GPIOx_ODSET，GPIOx_ODCLR 寄存器允许对 GPIOx_ODR 进行按位改写操作。这种情况下，可以避免在读和更改访问之间产生中断而发生的异常。

图 111-1 给出了一个标准 IO 端口的基本结构。

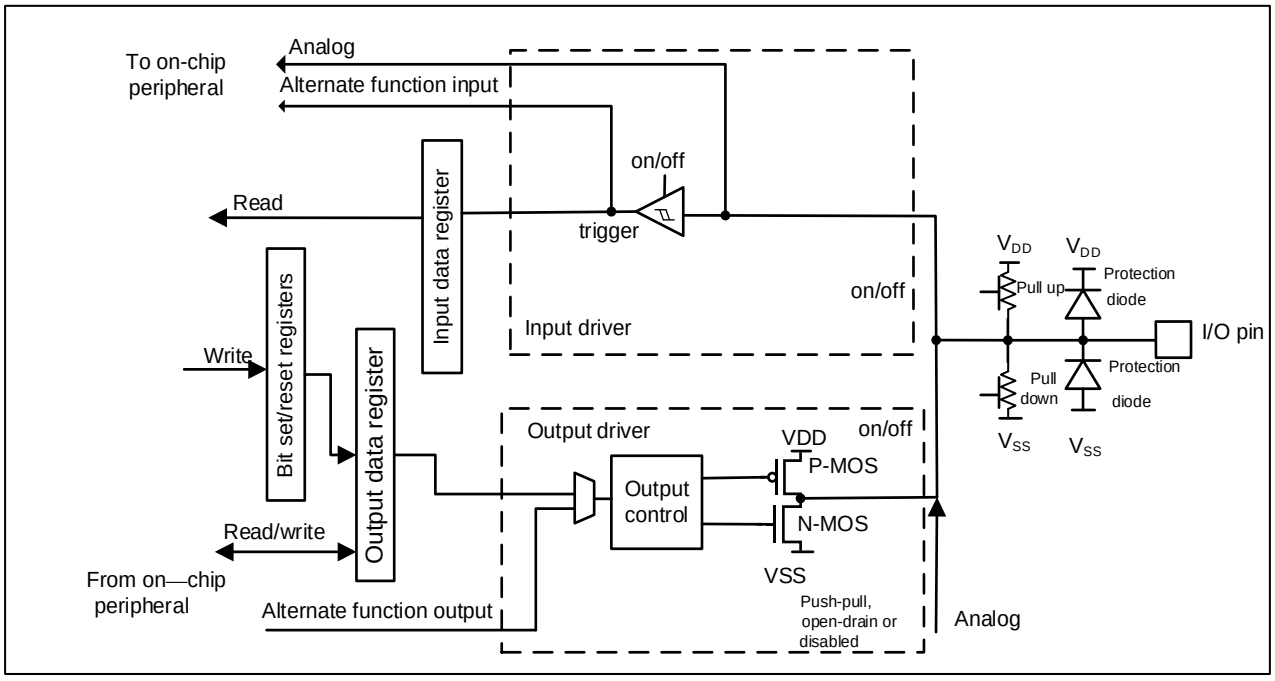


图 111-1 标准 I/O 端口的位基本结构

错误!未找到引用源。给出了端口位的可能配置

表 11-1 端口位配置表

AFR[i+3:i]	DIR[i]	OTYPCR[i]	DRVCR[i]	PUPD		IO Configuration	
0000	1	0	DR[i]	0	0	GPIO output	PP
		0		0	1	GPIO output	PP+PU
		0		1	0	GPIO output	PP+PD
		0		1	1	Reserved	
		1		0	0	GPIO output	OD
		1		0	1	GPIO output	OD+PU
		1		1	0	GPIO output	OD+PD
		1		1	1	Reserved(output OD)	
	0	x	x	0	0	Input	Floating
		x	x	0	1	Input	PU
		x	x	1	0	Input	PD
		x	x	1	1	Reserved(input floating)	
0001 ~ 1110	x	0	DR[i]	0	0	AF	PP
	x	0		0	1	AF	PP+PU
	x	0		1	0	AF	PP+PD
	x	0		1	1	Reserved	
	x	1		0	0	AF	OD
	x	1		0	1	AF	OD+PU
	x	1		1	0	AF	OD+PD
	x	1		1	1	Reserved	
1111	x	x	x	0	0	Input/output	Analog
	x	x	x	0	1	Forbidden	
	x	x	x	1	0		
	x	x	x	1	1		

注：GP = 通用，PP = 推挽输出，PU = 上拉，PD = 下拉，OD = 开漏，AF = 复用功能.

11.3.1 通用 I/O(GPIO)

复位期间和复位后，复用功能未开启且除了调试和复位引脚以外的所有 I/O 端口都被配置为模拟功能。

复位后，调试引脚被置为复用功能的上拉/下拉模式：

PC4：SWDCLK 置于下拉模式

PA4：SWDIO 置于上拉模式

复位引脚：

PE6：如果被设定为复位引脚，则置为内部上拉模式；

如果被设定为 GPIO 引脚，则置为模拟功能

当作为输出配置时，写到输出数据寄存器(GPIOx_ODR)的值输出到相应的引脚上。可以以推挽模式或开漏模式(仅低电平被驱动，高电平表现为高阻)输出。

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

所有 GPIO 引脚都有一个内部弱上拉和弱下拉电阻，它们被激活或断开依赖于 GPIOx_PUPD 寄存器的值。

注：复位解除后，PA4和PC4上拉、下拉不受GPIOx_PUPDR寄存器控制，以及PA4和PC4的复用不受GPIOx_AFR1寄存器控

11.3.2 I/O 端口控制寄存器

每个通用 I/O 口都有 5 个配置寄存器(GPIOx_DIRCR，GPIOx_OTYPER，GPIOx_PUPDR，GPIOx_SLEWCR 和 GPIOx_DRVCR)用来配置 I/O 口线。

GPIOx_DIRCR 寄存器用来选择方向(输入/输出)。GPIOx_OTYPER、GPIOx_SLEWCR 和 GPIOx_DRVCR 寄存器来选择输出类型(推挽或开漏)、转换速率和驱动能力。

GPIOx_PUPDR 寄存器用来选择上拉/下拉方式。

11.3.3 I/O 端口数据寄存器

每个 GPIO 口有两个数据寄存器：输入和输出数据寄存器(GPIOx_IDR 和 GPIOx_ODR).GPIOx_ODR 用于存储输出数据，其可进行读/写访问。从 I/O 线的输入数据存放在(GPIOx_IDR)寄存器中，该寄存器为只读寄存器。

11.3.4 I/O 数据位处理

输出置位寄存器(GPIOx_ODSET)和输出清零寄存器(GPIOx_ODCLR)寄存器，允许应用对输出数据寄存器(GPIOx_ODR)的每个位进行置位和清零操作。

当对位 GPIOx_ODSET[i]写 1 时则置位相应的 ODR[i]位。当对 GPIOx_ODCLR[i]写 1 时，则清零相应的 ODR[i]位。

对 GPIOx_ODSET，GPIOx_ODCLR 中的任意位写 0 都不会影响 GPIOx_ODR 寄存器的值。不仅可以用 GPIOx_ODSET，GPIOx_ODCLR 寄存器来改变 GPIOx_ODR 的相应

位，也可以对 GPIOx_ODR 寄存器直接进行访问。GPIOx_ODSET，GPIOx_ODCLR 寄存器提供对 GPIOx_ODR 寄存器原子按位操作处理。

GPIOx_ODR 用 GPIOx_ODSET，GPIOx_ODCLR 置位或清零的访问机制不需要软件去关闭中断来访问 GPIOx_ODR：在一个 AHB 写访问周期改变 1 位或多位数据是可能的。

11.3.5 输入配置

当 I/O 编程配置为输入时：

- 该端口的输出缓冲区禁用
- 由 GPIOx_PUPD 寄存器的值来激活上拉和下拉电阻
- 在每个 AHB 时钟周期 I/O 引脚上的数据被采样进入输入数据寄存器。
- 用对输入数据寄存器的读访问来获取同步后的输入数据

图 111-2 给出了 I/O 端口的输入配置。

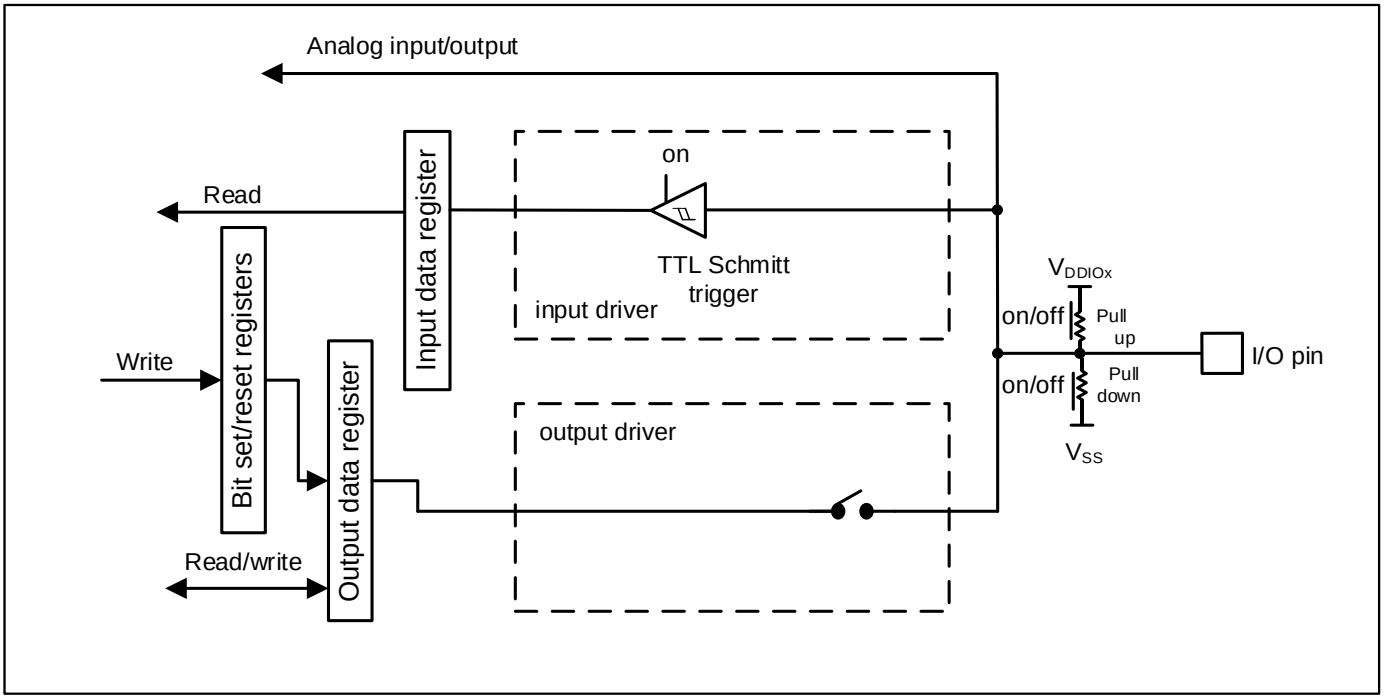


图 111-2 浮空输入/上拉/下拉配置

11.3.6 输出配置

当 I/O 口配置为输出时：

- 输出缓冲开启：
  - 开漏模式：输出寄存器上的’0’激活 N-MOS，而输出寄存器上的’1’将端口置于高阻状态(P-MOS 从不被激活)。
  - 推挽模式：输出寄存器上的’0’激活 N-MOS，而输出寄存器上的’1’将激活 P-MOS。
- 施密特触发输入被激活
- 弱上拉和弱下拉电阻是否激活取决于 GPIOx_PUPDR 寄存器的值
- 在每个 AHB 时钟周期 I/O 引脚上的数据被采样进入输入数据寄存器
- 用对输入数据寄存器的读访问来获取同步后的输入数据
- 用对输出数据寄存器的读访问来获取最后写进该寄存器的值

图 111-3 给出了 I/O 端口位的输出配置。

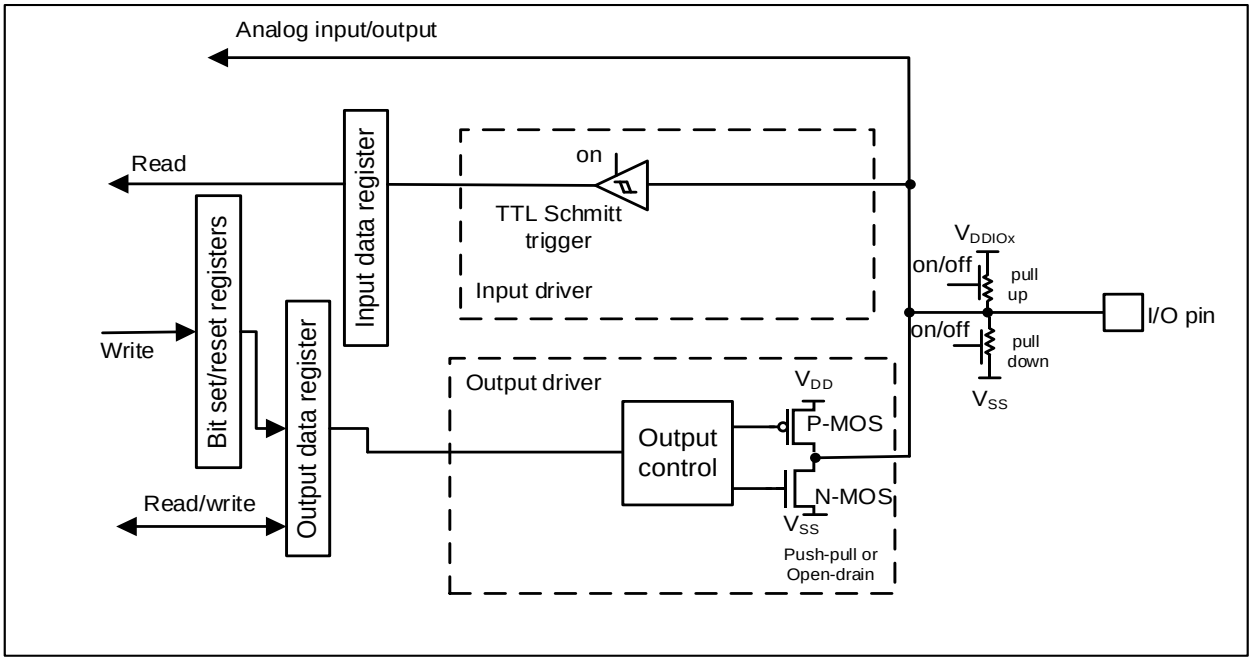


图 111-3 输出配置

11.3.7 外部中断/唤醒线

所有端口都有中断产生和唤醒能力。进入 Sleep 或者 Deep Sleep 模式后，如果检测到中断产生，WIC 模块会自动唤醒芯片。

每一个数字通用端口都可以由外部信号源产生中断，外部信号源可以是高电平/低电平/上升沿/下降沿 4 种类型的信号，当中断触发时，通过查询中断状态寄存器就可以判是哪一个端口触发了中断，通过置位中断清除寄存器就可以清除对应的中断状态标志位。

11.3.8 I/O 引脚的复用功能和重映射

器件 I/O 通过多路复用器连接到内嵌的外设模块。微观上，同一时刻仅允许外设的复用功能的一个引脚连接到一个 I/O 接口上。因此，同一根线上不能有冲突的外设引脚分配。

每个 I/O 引脚都有一个复用功能多路选择器，可通过配置 GPIOx_AFR 寄存器(从引脚 0 到引脚 7)来实现。

复位后，所有的 I/O 口都连接到 GPIO 功能。每个外设还有复用功能映射到不同的 I/O 引脚上，这种方法用于在小封装器件上优化更多的可用外设。

有关每个引脚的具体复用功能请参考：表 11-2 GPIO 口和外设引脚的复用功能映射

为了使用一个给定的 I/O 口配置，你必须按如下的原则执行：

- 调试功能：每个器件复位后，这些引脚立即配置为复用功能用来支持调用。
- GPIO：在 GPIOx_DIRCR 寄存器中配置所需的 I/O 为输出、输入。
- 外设的复用功能：
  - 连接 I/O 到所需的 AFRx，AFRx 定义在 GPIOx_AFR 寄存器中;
  - 通过对 GPIOx_PUPDR 和 GPIOx_SLEWCR，GPIOx_DRVCR 寄存器来配置相应引脚的上拉/下拉和输出速度；
  - 通过对 GPIOx_OTYPER 寄存器来配置输出类型：0 表示 push-pull，1 表示 open drain;
- 附加功能：
  - 对于 ADC/VC/PGA 等模拟 IP，配置 GPIOx_AFR 为 0xF，配置所需的 I/O 口线为模拟方式并在 ADC、VC 或 PGA 寄存器中配置所需的功能。
  - 对于附加功能振荡器，在关联的 RCC 寄存器配置相应所需的功能。

下表给出了 GPIO 口和外设引脚的复用功能映射；

表 11-2 GPIO 口和外设引脚的复用功能映射

20pin	32pin	48pin	64pin	80pin	Pin Name	DEFAULT(ALT15)	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	ALT8
		1	1	1	PD1	DISABLED	PD1	TIM1_CH1	SPI1_MOSI	LPUART_TX	TIM10_EXT	LPTIM_TOG	TIM2_CH2N	PCA_ECI	LIN0_RX
		2	2	2	PD0	DISABLED	PD0	TIM1_CH1N	SPI1_SCK	LPUART_RX	TIM10_TOG		TIM2_CH2	PCA_CH0	LIN0_TX
			3	3	PB11	DISABLED	PB11	TIM2_CH3	I2C1_SCL	LPTIM_EXT	TIM10_GATE		LPTIM_TOG	PCA_CH1	
			4	4	PB10	DISABLED	PB10	TIM2_CH2	I2C1_SDA	SPI1_SSN	TIM10_TOGN	CLK_MCO	LPTIM_TOGN	PCA_CH2	
				5	PD9	DISABLED	PD9	TIM1_CH1		CAN0_RX			LPTIM_TOG	PCA_CH3	LIN0_RX
	1	3	5	6	PB8	DISABLED	PB8	TIM1_CH2	SPI1_SCK	CAN0_TX	TIM10_GATE	LPTIM_GATE		1-wire	LIN0_TX
		4	6	7	PB9	DISABLED	PB9	TIM1_CH2N	SPI1_MISO	CAN0_RX	TIM10_TOGN				LIN0_RX
3	2	5			VCAP	VCAP									
4	3	6	7	8	VDD	VDD									
4	3	6	8	9	AVDD	AVDD									
			9	10	VCAP	VCAP									
5	4	7	10	11	AVSS	AVSS									
5	4	7	10	12	VSS	VSS									
6	5	8	11	13	PB7	OSC_IN/ OSC32K_IN	PB7		I2C0_SCL						
7	6	9	12	14	PB6	OSC_OUT/ OSC32K_OUT	PB6		I2C0_SDA						
			13	15	PD8	DISABLED	PD8	TIM1_CH4	TIM1_BKIN	UART1_RX			TIM2_BKIN	VC_OUT	
				16	PD10	DISABLED	PD10	TIM1_CH1	I2C0_SDA	UART1_TX	LPTIM_TOG				LIN1_TX
				17	PD11	DISABLED	PD11	TIM1_CH2	I2C0_SCL	UART1_RX	LPTIM_TOGN				LIN1_RX
	7	10	14	18	PA14	DISABLED	PA14	TIM1_CH3N	SPI0_MOSI			HIRC_OUT	TIM2_CH1	PCA_CH4	LIN1_TX
	8	11	15	19	PA15	DISABLED	PA15	TIM1_CH3	SPI0_SCK		TIM10_GATE	LVD_OUT	TIM2_CH2	LPTIM_TOG	LIN1_RX
		12	16	20	PB14	DISABLED	PB14	TIM2_CH4	I2C0_SDA		TIM10_GATE		TIM2_CH3	LPTIM_TOGN	
	9	13	17	21	PA8	AIN1/VC_IN3	PA8	TIM1_CH1	I2C0_SCL	CAN0_STDBY	TIM10_EXT	LIRC_OUT	TIM2_CH1N	PCA_CH0	
8	10	14	18	22	PB5	VREFEXT	PB5	TIM1_CH1N	SPI0_SSN		TIM10_TOGN	CLK_MCO	TIM2_CH2N	VC_OUT	1-wire
9	11	15	19	23	PB4	VC_IN4/ PGA_INP	PB4	TIM1_CH2	SPI0_MISO		TIM10_TOG	LPUART_RX	TIM2_CH3N	PCA_CH1	LIN0_RX
10	12	16	20	24	PC3	AIN2/VC_IN5	PC3	TIM1_CH2N	UART0_TX	CAN0_TX		LPUART_TX	TIM2_CH4		LIN0_TX
	13	17	21	25	PC2	AIN3/VC_IN6	PC2	TIM1_CH4	UART0_RX	CAN0_RX		LPUART_RX			LIN0_RX
			22	26	PD7	VC_IN7	PD7	TIM1_CH3N	UART1_TX	CAN0_STDBY	TIM10_GATE		TIM2_CH1		LIN1_TX
			23	27	PD6	VC_IN8	PD6	TIM1_CH3	UART1_RX		TIM10_EXT		TIM2_CH2		LIN1_RX
	14	18	24	28	PD5	DISABLED	PD5	TIM1_CH3N	I2C0_SDA	TIM1_BKIN	TIM10_TOGN	VC_OUT	LPTIM_EXT	PCA_CH2	
				29	PD12	DISABLED	PD12	TIM1_CH4					TIM2_CH2		
				30	PD13	DISABLED	PD13	TIM1_CH2					TIM2_CH4		
		19	25	31	PC1	AIN4	PC1	TIM1_CH3	SPI0_MOSI	I2C0_SCL	TIM11_GATE	LPTIM_GATE	TIM2_CH1	TIM2_CH4	
			26	32	PC0	AIN5	PC0	TIM1_CH1	SPI1_SCK	I2C1_SDA	TIM11_EXT	TIM2_CH3N	TIM2_CH3	PCA_CH3	
				33	PD14	DISABLED	PD14			I2C1_SCL				LPTIM_TOG	

20pin	32pin	48pin	64pin	80pin	Pin Name	DEFAULT(ALT15)	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	ALT8
				34	PD15	DISABLED	PD15			I2C1_SDA				LPTIM_TOGN	
			27	35	PB15	AIN6/VC_IN9	PB15	TIM1_CH1N			TIM11_TOG	TIM2_BKIN	TIM2_CH4		
		20	28	36	PA9	AIN7/VC_IN10	PA9	TIM2_CH3	SPI0_SCK	TIM1_BKIN	TIM11_EXT			PCA_ECI	
		21	29	37	PC15	AIN8	PC15	TIM2_CH2	SPI0_SSN	I2C1_SDA	TIM11_TOG				
		22	30	38	PC14	AIN9	PC14	TIM2_CH1	SPI0_MISO	I2C1_SCL	TIM11_TOGN				
11	15	23	31	39	PB3	AIN10/VC_IN11	PB3	TIM1_CH3	SPI0_MOSI		TIM10_EXT	LPTIM_TOG	TIM2_CH1N	PCA_CH1	LIN1_TX
12	16	24	32	40	PB2	AIN11/VC_IN12	PB2	TIM1_CH3N	SPI0_SCK		TIM10_GATE	LPTIM_TOGN	TIM2_CH2N	PCA_CH2	LIN1_RX
13	17	25	33	41	PB1	AIN12	PB1	TIM1_CH1	SPI0_MISO	CAN0_TX	UART0_TX	LPUART_TX		PCA_CH3	LIN0_TX
14	18	26	34	42	PB0	AIN13	PB0	TIM1_CH1N	SPI0_SSN	CAN0_RX	UART0_RX	LPUART_RX	LPTIM_EXT	1-wire	LIN0_RX
		27	35	43	PC9	DISABLED	PC9	TIM1_CH2	UART1_TX	LPUART_TX	TIM10_EXT	RTC_CLKOUT	TIM2_CH1	LPTIM_GATE	LIN1_TX
		28	36	44	PC8	DISABLED	PC8	TIM1_CH2N	UART1_RX		TIM10_TOGN	LPUART_RX	TIM2_CH2		LIN1_RX
15	19	29	37	45	PA7	AIN14/VC_IN13	PA7	TIM1_CH4	I2C0_SCL	TIM1_BKIN	TIM10_TOG	RTC_CLKIN		SPI0_MOSI	
			38	46	PA6	AIN15/VC_IN14	PA6	TIM1_CH1	UART0_TX	SPI0_SCK	TIM10_EXT	SPI1_SSN	TIM2_BKIN	LPTIM_TOG	
			39	47	PC10	DISABLED	PC10	TIM1_CH1N	UART0_RX		TIM10_GATE			LPTIM_TOGN	
	20	30	40	48	VSS	VSS									
	21	31	41	49	VDD	VDD									
				50	PE0	DISABLED	PE0					SPI1_SSN		PCA_CH4	
				51	PE1	DISABLED	PE1	TIM2_CH1				SPI1_MISO		PCA_CH4	
				52	PE2	DISABLED	PE2	TIM2_CH2		CAN0_RX		SPI1_MOSI			
				53	PE3	DISABLED	PE3	TIM2_CH3		CAN0_TX		SPI1_SCK		PCA_ECI	
		32	42	54	PB13	DISABLED	PB13	TIM1_CH3N	I2C0_SDA	CAN0_TX	TIM11_EXT		TIM2_CH3	PCA_CH1	
			43	55	PB12	DISABLED	PB12	TIM1_CH2N	I2C1_SCL	CAN0_RX	TIM11_GATE			PCA_CH2	
			44	56	PD4	DISABLED	PD4	TIM1_CH1N	I2C1_SDA		TIM11_TOGN		TIM1_BKIN	PCA_CH3	
	22	33	45	57	PD3	DISABLED	PD3	TIM1_CH3	SPI1_SSN	LPUART_TX	TIM11_TOG	NMI	I2C1_SCL	PCA_CH4	
		34	46	58	PD2	DISABLED	PD2	TIM1_CH2	SPI1_MISO	CAN0_STDBY	TIM11_GATE	LPUART_RX	I2C1_SDA		
16	23	35	47	59	PA3	DISABLED	PA3	TIM1_CH1	I2C1_SCL	CAN0_TX	TIM11_EXT	UART0_TX	BEEP	PCA_CH3	LIN0_TX
17	24	36	48	60	PA2	DISABLED	PA2	TIM2_CH2	I2C1_SDA	CAN0_RX	TIM11_GATE	UART0_RX	LPTIM_GATE	1-wire	LIN0_RX
18	25	37	49	61	PA1	AIN16/VC_IN1	PA1	TIM2_CH1	I2C1_SDA	UART1_TX	TIM11_TOGN	SPI1_MOSI	TIM1_CH1	1-wire	
19	26	38	50	62	PA0	AIN17/VC_IN0	PA0	TIM2_CH4	I2C1_SCL	UART1_RX	TIM11_TOG	SPI1_SCK	TIM1_CH1N		
		39	51	63	PC7	DISABLED	PC7	TIM2_CH2	SPI1_MISO	CAN0_TX	UART1_TX		TIM1_CH2	PCA_ECI	LIN1_TX
		40	52	64	PC6	DISABLED	PC6	TIM2_CH3	SPI1_SSN	CAN0_RX	UART1_RX		TIM1_CH2N	PCA_CH1	LIN1_RX
				65	PE4	DISABLED	PE4							PCA_CH2	
				66	PE5	DISABLED	PE5			CAN0_RX					
			53	67	PC11	DISABLED	PC11	TIM1_CH4	SPI1_SSN	CAN0_TX				LPTIM_TOG	LIN1_TX
			54	68	PC12	DISABLED	PC12		SPI1_MISO	CAN0_RX			LPTIM_EXT	LPTIM_TOGN	LIN1_RX
				69	VSS	VSS									
				70	VDD	VDD									

20pin	32pin	48pin	64pin	80pin	Pin Name	DEFAULT(ALT15)	ALT0	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7	ALT8
		41	55	71	PA13	DISABLED	PA13	TIM1_CH3N	SPI1_MOSI	CAN0_TX	TIM11_TOGN	LPTIM_GATE	TIM1_CH3		
		42	56	72	PA12	DISABLED	PA12	TIM1_CH2N	SPI1_SCK	CAN0_RX	TIM11_TOG	I2C0_SDA	TIM1_CH3N	TIM2_CH1	
	27	43	57	73	PA11	DISABLED	PA11	TIM1_CH1N	UART1_TX	SPI1_SSN	TIM11_EXT	I2C0_SCL	TIM1_CH2	TIM2_CH2	LIN1_TX
	28	44	58	74	PA10	DISABLED	PA10	TIM1_CH3	UART1_RX	SPI1_MISO	TIM11_GATE	I2C0_SDA	TIM1_CH2N	TIM2_CH3	LIN1_RX
			59	75	PC13	DISABLED	PC13	TIM1_CH2	SPI1_MOSI	I2C1_SCL	TIM11_TOG	SPI0_SSN		TIM2_CH1N	
			60	76	PA5	DISABLED	PA5	TIM1_CH1	SPI1_SCK	I2C1_SDA	TIM11_EXT	SPI0_MISO	BEEP	TIM2_CH2N	LIN1_TX
	29	45	61	77	PC5	DISABLED	PC5	TIM1_BKIN	SPI1_SSN	TIM1_CH3	TIM11_TOGN	RTC_CLKOUT	BEEP	TIM2_CH3N	LIN1_RX
20	30	46	62	78	PC4	SWDCLK (AIN0/VC_IN2)	PC4	TIM2_CH4	UART1_TX	TIM1_CH3N	TIM10_GATE	RTC_CLKOUT		TIM2_BKIN	LIN0_TX
1	31	47	63	79	PE6	NRST	PE6								
2	32	48	64	80	PA4	SWDIO	PA4	TIM2_CH1	SPI1_MISO	UART1_RX	TIM10_TOGN	VC_OUT			LIN0_RX

当 I/O 端口被配置为复用功能时：

- 在开漏或推挽模式下输出缓冲器可被配置
- 外设的信号驱动输出缓冲器
- 施密特触发输入被激活
- 弱上拉和弱下拉电阻是否激活取决于 GPIOx_PUPDR 寄存器的值
- 在每个 AHB 时钟周期 I/O 引脚上的数据被采样进入输入数据寄存器
- 用对输入数据寄存器的读访问来获取同步后的输入数据

图 111-4 给出了 I/O 端口位的复用功能配置。

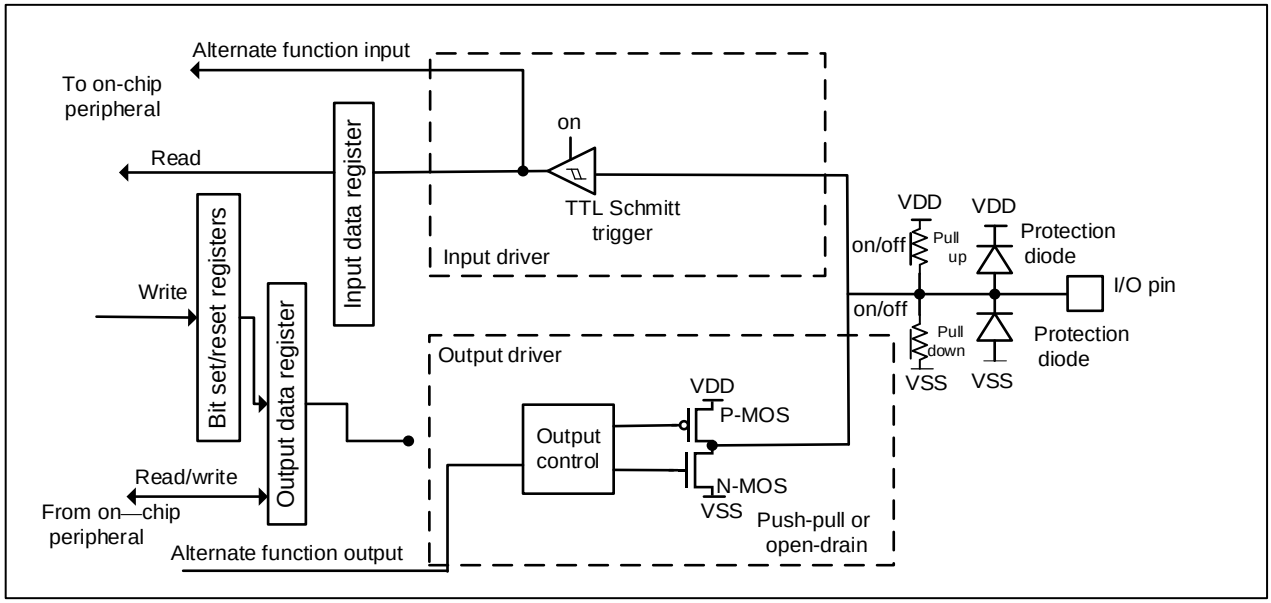


图 111-4 复用功能配置

11.3.9 模拟配置

当 I/O 端口编程为模拟配置时(GPIOx_AFR.AFR=0xF)：

- 输出缓冲器关闭
- 禁止施密特触发输入，实现了每个模拟 I/O 引脚上的零消耗。施密特触发输出值被强置为'0'
- 弱上拉和下拉电阻被禁止
- 读取输入数据寄存器时数值为 0

图 111-5 给出了 I/O 端口位的高阻抗模拟输入配置。

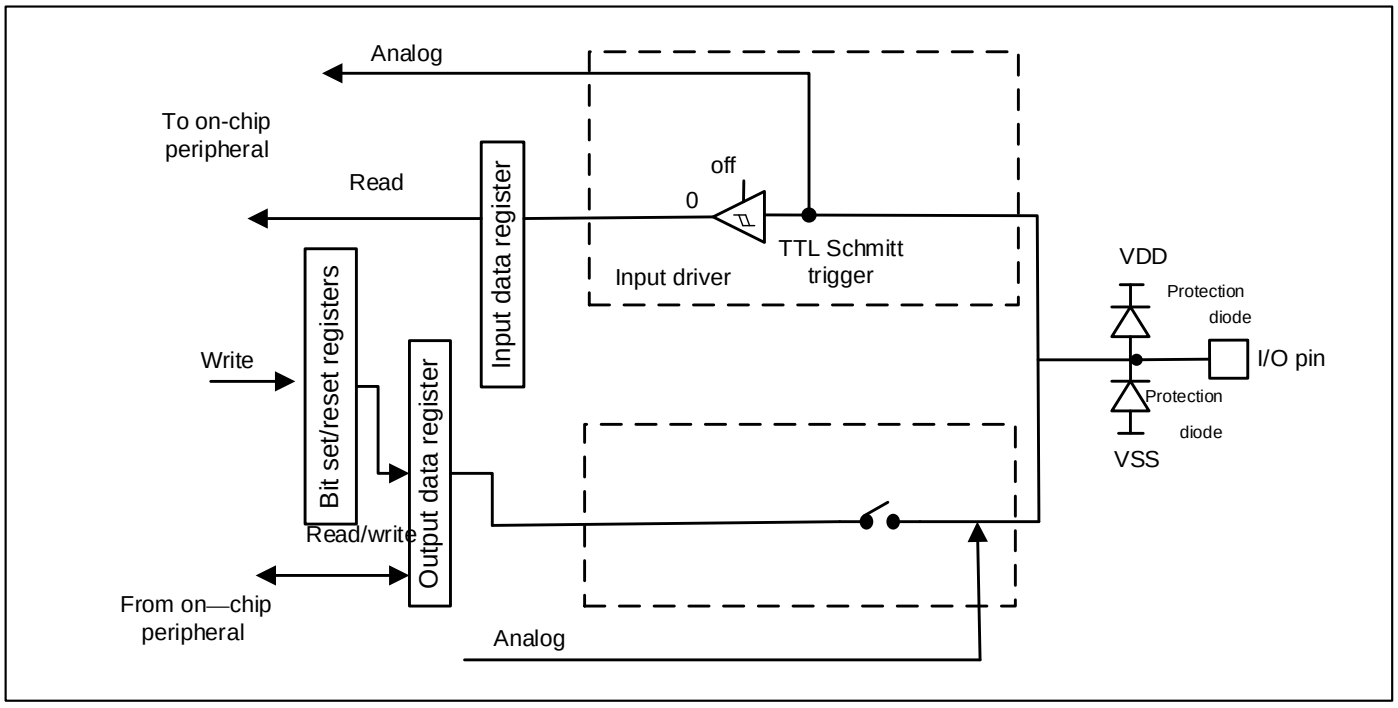


图 111-5 高阻抗模拟配置

11.3.10 HXT 或 LXT 引脚用作 GPIO

当 HXT 或 LXT 振荡器关断时(复位后的缺省状态)，相关振荡器引脚可以用做普通的 GPIO 口。

当 HXT 或 LXT 振荡器开启(设置 RCC_SYSCLKCR.HXTEN 或 RCC_LXTCR.LXTEN 位来开启)时，必须配置相应的管脚位模拟功能,振荡器控制其相关引脚且相关引脚的 GPIO 配置无效。

配置振荡器引脚为模拟功能的方法：

- 通过 RCC_SYSCLKCR.HXTPORT=1 或 RCC_LXTCR. LXTPORT=1 来配置
- 通过设定相应的引脚的 GPIOx_AFR=0xF 来配置

当振荡器配置为用户外部时钟输入方式(RCC_SYSCLKCR. HXTBYP=1 或 RCC_LXTCR. LXTBYP=1)，不需要配置相应的引脚为模拟功能，仅使用 OSC_IN 或 X32K_IN 引脚做为时钟输入处理，OSC_OUT 或 X32K_OUT 引脚仍然可配置为正常的 GPIO 引脚。

11.4 GPIO 寄存器列表

x=A、B、C、D；

GPIOx 基地址 0X40021000

GPIOx	偏移地址	描述
GPIOA	0x000	GPIOA 偏移地址
GPIOB	0x400	GPIOB 偏移地址
GPIOC	0x800	GPIOC 偏移地址
GPIOD	0xC00	GPIOD 偏移地址
GPIOE	0x1000	GPIOE 偏移地址

表 11-3 GPIOx 寄存器列表和复位值

偏移地址	名称	描述	复位值
0x00	GPIOx_DIRCR	输入输出模式寄存器	0x0000 0000
0x04	GPIOx_OTYPER	输出类型寄存器	0x0000 0000
0x08	GPIOx_ODR	输出数据寄存器	0x0000 0000
0x0C	GPIOx_IDR	输入数据寄存器	0xXXXX XXXX
0x10	GPIOx_INTEN	中断使能寄存器	0x0000 0000
0x14	GPIOx_RAWINTSR	中断原始状态寄存器，只读。 不论中断是否使能，都可以读到中断状态。	0x0000 0000
0x18	GPIOx_MSKINTSR	中断状态寄存器，只读	0x0000 0000
0x1C	GPIOx_INTCLR	中断清除寄存器	0x0000 0000
0x20	GPIOx_INTTYPCR	中断类型寄存器	0x0000 0000
0x24	GPIOx_INTPOLCR	中断类型值寄存器	0x0000 0000
0x28	GPIOx_INTANY	任意边沿触发中断寄存器	0x0000 0000
0x2C	GPIOx_ODSET	输出置位寄存器	0x0000 0000
0x30	GPIOx_ODCLR	输出清除寄存器	0x0000 0000
0x34	GPIOx_INDBEN	输入去抖动和同步使能寄存器	0x0000 0000
0x38	GPIOx_DBCLKCR	输入去抖动时钟配置寄存器	0x0000 0000
0x3C	GPIOx_PUPDR	上拉/下拉寄存器	0x0000 0000
0x40	GPIOx_SLEWCR	电压转换速率控制	0x0000 FFFF
0x44	GPIOx_DRVCR	驱动强度配置	0x0000 0000
0x48	GPIOx_AFR1	复用功能寄存器 1	0xFFFF FFFF
0x4C	GPIOx_AFR2	复用功能寄存器 2	0xFFFF FFFF
0x50	GPIOx_CS	施密特触发输入选择寄存器	0x0000 0000

0：表示逻辑值0；1：表示逻辑值1；X：表示不确定

注：复位解除后，PA4和PC4上拉、下拉不受GPIOx_PUPDR寄存器控制，以及PA4和PC4的复用不受GPIOx_AFR1寄存器控制

11.5 GPIO 寄存器说明

11.5.1 GPIO 端口方向寄存器(GPIOx_DIRCR) (x = A..E)

偏移地址：0x00

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxDIR15	PxDIR14	PxDIR13	PxDIR12	PxDIR11	PxDIR10	PxDIR9	PxDIR8	PxDIR7	PxDIR6	PxDIR5	PxDIR4	PxDIR3	PxDIR2	PxDIR1	PxDIR0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxDIR15	管脚 Px15 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
14	PxDIR14	管脚 Px14 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
13	PxDIR13	管脚 Px13 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
12	PxDIR12	管脚 Px12 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
11	PxDIR11	管脚 Px11 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
10	PxDIR10	管脚 Px10 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
9	PxDIR9	管脚 Px9 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
8	PxDIR8	管脚 Px8 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
7	PxDIR7	管脚 Px7 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W

6	PxDIR6	管脚 Px6 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
5	PxDIR5	管脚 Px5 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
4	PxDIR4	管脚 Px4 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
3	PxDIR3	管脚 Px3 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
2	PxDIR2	管脚 Px2 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
1	PxDIR1	管脚 Px1 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W
0	PxDIR0	管脚 Px0 输入/输出方向选择位 0：输入模式 1：输出模式	0x0	R/W

11.5.2 GPIO 端口输出类型寄存器(GPIOx_OTYPER) (x = A..E)

偏移地址：0x04

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxOTYP15	PxOTYP14	PxOTYP13	PxOTYP12	PxOTYP11	PxOTYP10	PxOTYP9	PxOTYP8	PxOTYP7	PxOTYP6	PxOTYP5	PxOTYP4	PxOTYP3	PxOTYP2	PxOTYP1	PxOTYP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxOTYP15	管脚 Px15 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
14	PxOTYP14	管脚 Px14 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
13	PxOTYP13	管脚 Px13 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
12	PxOTYP12	管脚 Px12 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
11	PxOTYP11	管脚 Px11 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
10	PxOTYP10	管脚 Px10 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
9	PxOTYP9	管脚 Px9 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
8	PxOTYP8	管脚 Px8 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
7	PxOTYP7	管脚 Px7 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
6	PxOTYP6	管脚 Px6 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W

5	PxOTYP5	管脚 Px5 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
4	PxOTYP4	管脚 Px4 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
3	PxOTYP3	管脚 Px3 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
2	PxOTYP2	管脚 Px2 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
1	PxOTYP1	管脚 Px1 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W
0	PxOTYP0	管脚 Px0 输出类型控制位 0：推挽输出(复位状态) 1：开漏输出	0x0	R/W

11.5.3 GPIO 端口输出数据寄存器(GPIOx_ODR) (x = A..E)

偏移地址：0x08

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxOD15	PxOD14	PxOD13	PxOD12	PxOD11	PxOD10	PxOD9	PxOD8	PxOD7	PxOD6	PxOD5	PxOD4	PxOD3	PxOD2	PxOD1	PxOD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxOD15	管脚 Px15 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
14	PxOD14	管脚 Px14 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
13	PxOD13	管脚 Px13 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
12	PxOD12	管脚 Px12 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
11	PxOD11	管脚 Px11 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
10	PxOD10	管脚 Px10 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
9	PxOD9	管脚 Px9 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
8	PxOD8	管脚 Px8 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
7	PxOD7	管脚 Px7 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
6	PxOD6	管脚 Px6 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W

5	PxOD5	管脚 Px5 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
4	PxOD4	管脚 Px4 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
3	PxOD3	管脚 Px3 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
2	PxOD2	管脚 Px2 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
1	PxOD1	管脚 Px1 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W
0	PxOD0	管脚 Px0 输出值配置位 0：输出低电平 1：输出高电平，如果为开漏输出，需要配置或外接上拉电阻	0x0	R/W

对于管脚的设置/清除，可单独对GPIOx_ODSET，GPIOx_ODCLR(x= A..D)寄存器操作来实现。

11.5.4 GPIO 端口输入数据寄存器(GPIOx_IDR) (x = A..E)

偏移地址：0x0C

复位值：0xXXXXXXXX

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxID 15	PxID 14	PxID 13	PxID 12	PxID 11	PxID 10	PxID 9	PxID 8	PxID 7	PxID 6	PxID 5	PxID 4	PxID 3	PxID 2	PxID 1	PxID0
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

位	标记	功能描述	复位值	读写
31:16	Res	保留	0xX	–
15	PxID15	管脚 Px15 输入值 0：输入低电平 1：输入高电平	0xX	RO
14	PxID14	管脚 Px14 输入值 0：输入低电平 1：输入高电平	0xX	RO
13	PxID13	管脚 Px13 输入值 0：输入低电平 1：输入高电平	0xX	RO
12	PxID12	管脚 Px12 输入值 0：输入低电平 1：输入高电平	0xX	RO
11	PxID11	管脚 Px11 输入值 0：输入低电平 1：输入高电平	0xX	RO
10	PxID10	管脚 Px10 输入值 0：输入低电平 1：输入高电平	0xX	RO
9	PxID9	管脚 Px9 输入值 0：输入低电平 1：输入高电平	0xX	RO
8	PxID8	管脚 Px8 输入值 0：输入低电平 1：输入高电平	0xX	RO
7	PxID7	管脚 Px7 输入值 0：输入低电平 1：输入高电平	0xX	RO

6	PxID6	管脚 Px6 输入值 0：输入低电平 1：输入高电平	0xX	RO
5	PxID5	管脚 Px5 输入值 0：输入低电平 1：输入高电平	0xX	RO
4	PxID4	管脚 Px4 输入值 0：输入低电平 1：输入高电平	0xX	RO
3	PxID3	管脚 Px3 输入值 0：输入低电平 1：输入高电平	0xX	RO
2	PxID2	管脚 Px2 输入值 0：输入低电平 1：输入高电平	0xX	RO
1	PxID1	管脚 Px1 输入值 0：输入低电平 1：输入高电平	0xX	RO
0	PxID0	管脚 Px0 输入值 0：输入低电平 1：输入高电平	0xX	RO

注：X 表示不定值

11.5.5 GPIO 端口中断使能寄存器(GPIOx_INTEN) (x = A..E)

偏移地址：0x10

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxIEN15	PxIEN14	PxIEN13	PxIEN12	PxIEN11	PxIEN10	PxIEN9	PxIEN8	PxIEN7	PxIEN6	PxIEN5	PxIEN4	PxIEN3	PxIEN2	PxIEN1	PxIEN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxIEN15	管脚 Px15 中断屏蔽解除位 0：Px15 管脚中断屏蔽 1：Px15 管脚中断有效	0x0	R/W
14	PxIEN14	管脚 Px14 中断屏蔽解除位 0：Px14 管脚中断屏蔽 1：Px14 管脚中断有效	0x0	R/W
13	PxIEN13	管脚 Px13 中断屏蔽解除位 0：Px13 管脚中断屏蔽 1：Px13 管脚中断有效	0x0	R/W
12	PxIEN12	管脚 Px12 中断屏蔽解除位 0：Px12 管脚中断屏蔽 1：Px12 管脚中断有效	0x0	R/W
11	PxIEN11	管脚 Px11 中断屏蔽解除位 0：Px11 管脚中断屏蔽 1：Px11 管脚中断有效	0x0	R/W
10	PxIEN10	管脚 Px10 中断屏蔽解除位 0：Px10 管脚中断屏蔽 1：Px10 管脚中断有效	0x0	R/W
9	PxIEN9	管脚 Px9 中断屏蔽解除位 0：Px9 管脚中断屏蔽 1：Px9 管脚中断有效	0x0	R/W
8	PxIEN8	管脚 Px8 中断屏蔽解除位 0：Px8 管脚中断屏蔽 1：Px8 管脚中断有效	0x0	R/W

7	PxIEN7	管脚 Px7 中断屏蔽解除位 0：Px7 管脚中断屏蔽 1：Px7 管脚中断有效	0x0	R/W
6	PxIEN6	管脚 Px6 中断屏蔽解除位 0：Px6 管脚中断屏蔽 1：Px6 管脚中断有效	0x0	R/W
5	PxIEN5	管脚 Px5 中断屏蔽解除位 0：Px5 管脚中断屏蔽 1：Px5 管脚中断有效	0x0	R/W
4	PxIEN4	管脚 Px4 中断屏蔽解除位 0：Px4 管脚中断屏蔽 1：Px4 管脚中断有效	0x0	R/W
3	PxIEN3	管脚 Px3 中断屏蔽解除位 0：Px3 管脚中断屏蔽 1：Px3 管脚中断有效	0x0	R/W
2	PxIEN2	管脚 Px2 中断屏蔽解除位 0：Px2 管脚中断屏蔽 1：Px2 管脚中断有效	0x0	R/W
1	PxIEN1	管脚 Px1 中断屏蔽解除位 0：Px1 管脚中断屏蔽 1：Px1 管脚中断有效	0x0	R/W
0	PxIEN0	管脚 Px0 中断屏蔽解除位 0：Px0 管脚中断屏蔽 1：Px0 管脚中断有效	0x0	R/W

11.5.6 GPIO 端口中断原始状态寄存器(GPIOx_RAWINTSR) (x = A..E)

偏移地址：0x14

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxRIS15	PxRIS14	PxRIS13	PxRIS12	PxRIS11	PxRIS10	PxRIS9	PxRIS8	PxRIS7	PxRIS6	PxRIS5	PxRIS4	PxRIS3	PxRIS2	PxRIS1	PxRIS0
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxRIS15	管脚 Px15 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
14	PxRIS14	管脚 Px14 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
13	PxRIS13	管脚 Px13 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
12	PxRIS12	管脚 Px12 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
11	PxRIS11	管脚 Px11 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
10	PxRIS10	管脚 Px10 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
9	PxRIS9	管脚 Px9 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
8	PxRIS8	管脚 Px8 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
7	PxRIS7	管脚 Px7 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
6	PxRIS6	管脚 Px6 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO

5	PxRIS5	管脚 Px5 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
4	PxRIS4	管脚 Px4 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
3	PxRIS3	管脚 Px3 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
2	PxRIS2	管脚 Px2 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
1	PxRIS1	管脚 Px1 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO
0	PxRIS0	管脚 Px0 中断原始状态位 0：无中断发生 1：中断发生	0x0	RO

不论中断是否使能，都可以读到中断状态。

11.5.7 GPIO 端口中断状态寄存器(GPIOx_MSKINTSR) (x = A...E)

偏移地址：0x18

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxMIS15	PxMIS14	PxMIS13	PxMIS12	PxMIS11	PxMIS10	PxMIS9	PxMIS8	PxMIS7	PxMIS6	PxMIS5	PxMIS4	PxMIS3	PxMIS2	PxMIS1	PxMIS0
RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO

这些位只读，由硬件置位，只有中断使能，才可以读到中断状态。

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxMIS15	管脚 Px15 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
14	PxMIS14	管脚 Px14 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
13	PxMIS13	管脚 Px13 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
12	PxMIS12	管脚 Px12 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
11	PxMIS11	管脚 Px11 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
10	PxMIS10	管脚 Px10 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
9	PxMIS9	管脚 Px9 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
8	PxMIS8	管脚 Px8 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
7	PxMIS7	管脚 Px7 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
6	PxMIS6	管脚 Px6 屏蔽后的中断状态位 0：无中断发生	0x0	RO

		1：中断发生		
5	PxMIS5	管脚 Px5 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
4	PxMIS4	管脚 Px4 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
3	PxMIS3	管脚 Px3 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
2	PxMIS2	管脚 Px2 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
1	PxMIS1	管脚 Px1 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO
0	PxMIS0	管脚 Px0 屏蔽后的中断状态位 0：无中断发生 1：中断发生	0x0	RO

11.5.8 GPIO 端口中断清除寄存器(GPIOx_INTCLR) (x = A..E)

偏移地址：0x1C

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxIC LR15	PxIC LR14	PxIC LR13	PxIC LR12	PxIC LR11	PxIC LR10	PxIC LR9	PxIC LR8	PxIC LR7	PxIC LR6	PxIC LR5	PxIC LR4	PxIC LR3	PxIC LR2	PxIC LR1	PxIC LR0
WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO

这些位只写，用于软件清除中断。

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxICLR15	写 1 清除管脚 Px15 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
14	PxICLR14	写 1 清除管脚 Px14 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
13	PxICLR13	写 1 清除管脚 Px13 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
12	PxICLR12	写 1 清除管脚 Px12 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
11	PxICLR11	写 1 清除管脚 Px11 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
10	PxICLR10	写 1 清除管脚 Px10 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
9	PxICLR9	写 1 清除管脚 Px9 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
8	PxICLR8	写 1 清除管脚 Px8 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
7	PxICLR7	写 1 清除管脚 Px7 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
6	PxICLR6	写 1 清除管脚 Px6 的中断状态 0：保留中断标志位	0x0	WO

		1：清除对应的中断标志位		
5	PxICLR5	写 1 清除管脚 Px5 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
4	PxICLR4	写 1 清除管脚 Px4 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
3	PxICLR3	写 1 清除管脚 Px3 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
2	PxICLR2	写 1 清除管脚 Px2 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
1	PxICLR1	写 1 清除管脚 Px1 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO
0	PxICLR0	写 1 清除管脚 Px0 的中断状态 0：保留中断标志位 1：清除对应的中断标志位	0x0	WO

11.5.9 GPIO 端口中断类型寄存器(GPIOx_INTTYPCR) (x = A..E)

偏移地址：0x20

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxITYPE15	PxITYPE14	PxITYPE13	PxITYPE12	PxITYPE11	PxITYPE10	PxITYPE9	PxITYPE8	PxITYPE7	PxITYPE6	PxITYPE5	PxITYPE4	PxITYPE3	PxITYPE2	PxITYPE1	PxITYPE0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxITYPE15	管脚 Px15 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
14	PxITYPE14	管脚 Px14 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
13	PxITYPE13	管脚 Px13 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
12	PxITYPE12	管脚 Px12 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
11	PxITYPE11	管脚 Px11 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
10	PxITYPE10	管脚 Px10 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
9	PxITYPE9	管脚 Px9 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
8	PxITYPE8	管脚 Px8 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
7	PxITYPE7	管脚 Px7 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
6	PxITYPE6	管脚 Px6 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W

5	PxITYPE5	管脚 Px5 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
4	PxITYPE4	管脚 Px4 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
3	PxITYPE3	管脚 Px3 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
2	PxITYPE2	管脚 Px2 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
1	PxITYPE1	管脚 Px1 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W
0	PxITYPE0	管脚 Px0 的中断类型配置位 0：边沿触发中断类型 1：电平触发中断类型	0x0	R/W

11.5.10 GPIO 端口中断极性寄存器(GPIOx_INTPOLCR) (x = A..E)

偏移地址：0x24

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxPOL15	PxPOL14	PxPOL13	PxPOL12	PxPOL11	PxPOL10	PxPOL9	PxPOL8	PxPOL7	PxPOL6	PxPOL5	PxPOL4	PxPOL3	PxPOL2	PxPOL1	PxPOL0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxPOL15	管脚 Px15 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
14	PxPOL14	管脚 Px14 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
13	PxPOL13	管脚 Px13 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
12	PxPOL12	管脚 Px12 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
11	PxPOL11	管脚 Px11 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
10	PxPOL10	管脚 Px10 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
9	PxPOL9	管脚 Px9 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
8	PxPOL8	管脚 Px8 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
7	PxPOL7	管脚 Px7 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
6	PxPOL6	管脚 Px6 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W

5	PxPOL5	管脚 Px5 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
4	PxPOL4	管脚 Px4 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
3	PxPOL3	管脚 Px3 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
2	PxPOL2	管脚 Px2 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
1	PxPOL1	管脚 Px1 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W
0	PxPOL0	管脚 Px0 的中断极性配置位 0：低电平或下降沿触发中断 1：高电平或上升沿触发中断	0x0	R/W

11.5.11 GPIO 端口任意边沿触发中断寄存器(GPIOx_INTANY) (x = A..E)

偏移地址：0x28

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxIANY15	PxIANY14	PxIANY13	PxIANY12	PxIANY11	PxIANY10	PxIANY9	PxIANY8	PxIANY7	PxIANY6	PxIANY5	PxIANY4	PxIANY3	PxIANY2	PxIANY1	PxIANY0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxIANY15	管脚 Px15 任意沿触发中断配置位 0：中断触发沿由 PxIVAL15 决定 1：上升/下降沿都触发中断	0x0	R/W
14	PxIANY14	管脚 Px14 任意沿触发中断配置位 0：中断触发沿由 PxIVAL14 决定 1：上升/下降沿都触发中断	0x0	R/W
13	PxIANY13	管脚 Px13 任意沿触发中断配置位 0：中断触发沿由 PxIVAL13 决定 1：上升/下降沿都触发中断	0x0	R/W
12	PxIANY12	管脚 Px12 任意沿触发中断配置位 0：中断触发沿由 PxIVAL12 决定 1：上升/下降沿都触发中断	0x0	R/W
11	PxIANY11	管脚 Px11 任意沿触发中断配置位 0：中断触发沿由 PxIVAL11 决定 1：上升/下降沿都触发中断	0x0	R/W
10	PxIANY10	管脚 Px10 任意沿触发中断配置位 0：中断触发沿由 PxIVAL10 决定 1：上升/下降沿都触发中断	0x0	R/W
9	PxIANY9	管脚 Px9 任意沿触发中断配置位 0：中断触发沿由 PxIVAL9 决定 1：上升/下降沿都触发中断	0x0	R/W
8	PxIANY8	管脚 Px8 任意沿触发中断配置位 0：中断触发沿由 PxIVAL8 决定 1：上升/下降沿都触发中断	0x0	R/W
7	PxIANY7	管脚 Px7 任意沿触发中断配置位 0：中断触发沿由 PxIVAL7 决定 1：上升/下降沿都触发中断	0x0	R/W
6	PxIANY6	管脚 Px6 任意沿触发中断配置位 0：中断触发沿由 PxIVAL6 决定 1：上升/下降沿都触发中断	0x0	R/W

5	PxIANY5	管脚 Px5 任意沿触发中断配置位 0：中断触发沿由 PxIVAL5 决定 1：上升/下降沿都触发中断	0x0	R/W
4	PxIANY4	管脚 Px4 任意沿触发中断配置位 0：中断触发沿由 PxIVAL4 决定 1：上升/下降沿都触发中断	0x0	R/W
3	PxIANY3	管脚 Px3 任意沿触发中断配置位 0：中断触发沿由 PxIVAL3 决定 1：上升/下降沿都触发中断	0x0	R/W
2	PxIANY2	管脚 Px2 任意沿触发中断配置位 0：中断触发沿由 PxIVAL2 决定 1：上升/下降沿都触发中断	0x0	R/W
1	PxIANY1	管脚 Px1 任意沿触发中断配置位 0：中断触发沿由 PxIVAL1 决定 1：上升/下降沿都触发中断	0x0	R/W
0	PxIANY0	管脚 Px0 任意沿触发中断配置位 0：中断触发沿由 PxIVAL0 决定 1：上升/下降沿都触发中断	0x0	R/W

11.5.12 GPIO 端口输出置位寄存器(GPIOx_ODSET) (x = A..E)

偏移地址：0x2C

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxODSET15	PxODSET14	PxODSET13	PxODSET12	PxODSET11	PxODSET10	PxODSET9	PxODSET8	PxODSET7	PxODSET6	PxODSET5	PxODSET4	PxODSET3	PxODSET2	PxODSET1	PxODSET0
WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO	WO

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxODSET15	管脚 Px15 输出置 1 控制位,读取时 0：输出保持 1：输出被置高	0x0	WO
14	PxODSET14	管脚 Px14 输出置 1 控制位,读取时 0：输出保持 1：输出被置高	0x0	WO
13	PxODSET13	管脚 Px13 输出置 1 控制位,读取时 0：输出保持 1：输出被置高	0x0	WO
12	PxODSET12	管脚 Px12 输出置 1 控制位,读取时 0：输出保持 1：输出被置高	0x0	WO
11	PxODSET11	管脚 Px11 输出置 1 控制位,读取时 0：输出保持 1：输出被置高	0x0	WO
10	PxODSET10	管脚 Px10 输出置 1 控制位,读取时 0：输出保持 1：输出被置高	0x0	WO
9	PxODSET9	管脚 Px9 输出置 1 控制位,读取时 0：输出保持 1：输出被置高	0x0	WO
8	PxODSET8	管脚 Px8 输出置 1 控制位,读取时 0：输出保持 1：输出被置高	0x0	WO
7	PxODSET7	管脚 Px7 输出置 1 控制位,读取时 0：输出保持 1：输出被置高	0x0	WO

6	PxODSET6	管脚 Px6 输出置 1 控制位 0：输出保持 1：输出被置高	0x0	WO
5	PxODSET5	管脚 Px5 输出置 1 控制位 0：输出保持 1：输出被置高	0x0	WO
4	PxODSET4	管脚 Px4 输出置 1 控制位 0：输出保持 1：输出被置高	0x0	WO
3	PxODSET3	管脚 Px3 输出置 1 控制位 0：输出保持 1：输出被置高	0x0	WO
2	PxODSET2	管脚 Px2 输出置 1 控制位 0：输出保持 1：输出被置高	0x0	WO
1	PxODSET1	管脚 Px1 输出置 1 控制位 0：输出保持 1：输出被置高	0x0	WO
0	PxODSET0	管脚 Px0 输出置 1 控制位 0：输出保持 1：输出被置高	0x0	WO

11.5.13 GPIO 端口输出清除寄存器(GPIOx_ODCLR) (x = A..E)

偏移地址：0x30

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxODCLR15	管脚 Px15 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
14	PxODCLR14	管脚 Px14 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
13	PxODCLR13	管脚 Px13 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
12	PxODCLR12	管脚 Px12 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
11	PxODCLR11	管脚 Px11 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
10	PxODCLR10	管脚 Px10 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
9	PxODCLR9	管脚 Px9 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
8	PxODCLR8	管脚 Px8 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
7	PxODCLR7	管脚 Px7 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO

6	PxODCLR6	管脚 Px6 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
5	PxODCLR5	管脚 Px5 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
4	PxODCLR4	管脚 Px4 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
3	PxODCLR3	管脚 Px3 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
2	PxODCLR2	管脚 Px2 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
1	PxODCLR1	管脚 Px1 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO
0	PxODCLR0	管脚 Px0 输出清 0 控制位 0：输出保持 1：复位相应的 ODRx 位	0x0	WO

注：若 ODSETx 和 ODCLRx 同时设置，ODSETx 有优先权。

11.5.14 GPIO 端口输入去抖动寄存器(GPIOx_INDBEN) (x = A..E)

偏移地址：0x34

复位值：0x00000000

11.5.14.1 GPIOx_INDBEN(x = A..D)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															SYNC_EN
															R/W

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

位	标记	功能描述	复位值	读写
31:17	Res	保留	0x0	–
16	SYNC_EN	消抖未使能时通过设定该寄存器来配置输入是否使用 2 级同步来消除亚稳态(只对中断有效)。 0：不使用两级同步 1：使用两级同步	0x0	R/W
15:0	PxDIDBn	管脚 Pxn 消抖使能配置位，如果输入信号不能被连续 2 个的去抖采样周期采样，则输入信号被视为信号抖动，而不会触发中断。 仅用于边沿触发“edge-trigger”中断，不能用于电平触发(“level trigger”)中断。电平模式时为两级同步输入 0：禁止端口消抖功能 1：使能端口消抖功能	0x0	R/W

11.5.14.2 GPIOE_INDBEN

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								SYNC_EN	PxDIDBn						
								R/W	R/W						

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7	SYNC_EN	消抖未使能时通过设定该寄存器来配置输入是否使用 2 级同步来消除亚稳态(只对中断有效)。 0：不使用两级同步 1：使用两级同步	0x0	R/W
6:0	PxDIDBn	管脚 Pxn 消抖使能配置位，如果输入信号不能被连续 2 个的去抖采样周期采样，则输入信号被视为信号抖动，而不会触发中断。 仅用于边沿触发“edge-trigger”中断，不能用于电平触发(“level trigger”)中断。电平模式时为两级同步输入 0：禁止端口消抖功能 1：使能端口消抖功能	0x0	R/W

11.5.15 GPIO 端口输入去抖动时钟配置寄存器(GPIOx_DBCLKCR) (x = A..E)

偏移地址：0x38

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:5	Res	保留	0x0	–
4	DBCLKEN	是否使能去抖动时钟 0：不使能去抖动时钟 1：使能去抖动时钟	0x0	R/W
3:0	DBCLK_DIV	去抖动采样周期选择 Debounce 时钟是 hclk 的(2^ dbclk_div)分频。 消抖采样周期选择： 0x0：消抖采样 1 个 HCLK 周期 1 次 0x1：消抖采样 2 个 HCLK 周期 1 次 0x2：消抖采样 4 个 HCLK 周期 1 次 0x3：消抖采样 8 个 HCLK 周期 1 次 0x4：消抖采样 16 个 HCLK 周期 1 次 0x5：消抖采样 32 个 HCLK 周期 1 次 0x6：消抖采样 64 个 HCLK 周期 1 次 0x7：消抖采样 128 个 HCLK 周期 1 次 0x8：消抖采样 256 个 HCLK 周期 1 次 0x9：消抖采样 512 个 HCLK 周期 1 次 0xA：消抖采样 1024 个 HCLK 周期 1 次 0xB：消抖采样 2*1024 个 HCLK 周期 1 次 0xC：消抖采样 4*1024 个 HCLK 周期 1 次 0xD：消抖采样 8*1024 个 HCLK 周期 1 次 0xE：消抖采样 16*1024 个 HCLK 周期 1 次 0xF：消抖采样 32*1024 个 HCLK 周期 1 次	0x0	R/W

11.5.16 GPIO 端口上拉/下拉寄存器(GPIOx_PUPDR) (x = A..E)

偏移地址：0x3C

复位值：0x00000000

31 30	29 28	27 26	25 24	23 22	21 20	19 18	17 16
PxPUPD15	PxPUPD14	PxPUPD13	PxPUPD12	PxPUPD11	PxPUPD10	PxPUPD9	PxPUPD8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

15 14	13 12	11 10	9 8	7 6	5 4	3 2	1 0
PxPUPD7	PxPUPD6	PxPUPD5	PxPUPD4	PxPUPD3	PxPUPD2	PxPUPD1	PxPUPD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:30	PxPUPD15	管脚 Px15 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
29:28	PxPUPD14	管脚 Px14 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
27:26	PxPUPD13	管脚 Px13 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
25:24	PxPUPD12	管脚 Px12 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
23:22	PxPUPD11	管脚 Px11 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
21:20	PxPUPD10	管脚 Px10 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
19:18	PxPUPD9	管脚 Px9 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
17:16	PxPUPD8	管脚 Px8 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W

15:14	PxPUPD7	管脚 Px7 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
13:12	PxPUPD6	管脚 Px6 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
11:10	PxPUPD5	管脚 Px5 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
9:8	PxPUPD4	管脚 Px4 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
7:6	PxPUPD3	管脚 Px3 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
5:4	PxPUPD2	管脚 Px2 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
3:2	PxPUPD1	管脚 Px1 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W
1:0	PxPUPD0	管脚 Px0 上拉/下拉配置控制位。 00：上拉,下拉禁止 01：上拉使能 10：下拉使能 11：保留	0x0	R/W

11.5.17 GPIO 端口电压转换速率配置(GPIOx_SLEWCR) (x = A..E)

偏移地址：0x40

复位值：0x0000FFFF(注：GPIOE 复位值为 0x0000007F)

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxSR15	管脚 Px15 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
14	PxSR14	管脚 Px14 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
13	PxSR13	管脚 Px13 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
12	PxSR12	管脚 Px12 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
11	PxSR11	管脚 Px11 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
10	PxSR10	管脚 Px10 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
9	PxSR9	管脚 Px9 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
8	PxSR8	管脚 Px8 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
7	PxSR7	管脚 Px7 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W

6	PxSR6	管脚 Px6 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
5	PxSR5	管脚 Px5 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
4	PxSR4	管脚 Px4 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
3	PxSR3	管脚 Px3 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
2	PxSR2	管脚 Px2 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
1	PxSR1	管脚 Px1 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W
0	PxSR0	管脚 Px0 压摆率配置控制位。 0：高压摆率 1：低压摆率	0x1	R/W

11.5.18 GPIO 端口驱动强度配置寄存器(GPIOx_DRVCR) (x = A..E)

偏移地址：0x44

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxDRV15	PxDRV14	PxDRV13	PxDRV12	PxDRV11	PxDRV10	PxDRV9	PxDRV8	PxDRV7	PxDRV6	PxDRV5	PxDRV4	PxDRV3	PxDRV2	PxDRV1	PxDRV0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxDRV15	管脚 Px15 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
14	PxDRV14	管脚 Px14 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
13	PxDRV13	管脚 Px13 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
12	PxDRV12	管脚 Px12 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
11	PxDRV11	管脚 Px11 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
10	PxDRV10	管脚 Px10 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
9	PxDRV9	管脚 Px9 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
8	PxDRV8	管脚 Px8 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
7	PxDRV7	管脚 Px7 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
6	PxDRV6	管脚 Px6 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W

5	PxDRV5	管脚 Px5 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
4	PxDRV4	管脚 Px4 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
3	PxDRV3	管脚 Px3 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
2	PxDRV2	管脚 Px2 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
1	PxDRV1	管脚 Px1 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W
0	PxDRV0	管脚 Px0 驱动强度选择控制位。 0：高驱动强度 1：低驱动强度	0x0	R/W

11.5.19 GPIO 端口复用功能寄存器(GPIOx_AFR1) (x = A..E)

偏移地址：0x48

复位值：0xFFFFFFFF (GPIOE 的复位值为 0x0FFFFFFF)

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

11.5.19.1 GPIOA 复用配置

位	标记	功能描述	复位值	读写
31:28	PAAFR7	端口PA7功能选择 0000：PA7 0001：TIM1_CH4 0010：I2C0_SCL 0011：TIM1_BKIN 0100：TIM10_TOG 0101：RTC_CLKIN 0110：保留 0111：SPI0_MOSI 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
27:24	PAAFR6	端口PA6功能选择 0000：PA6 0001：TIM1_CH1 0010：UART0_TX 0011：SPI0_SCK 0100：TIM10_EXT 0101：SPI1_SSN 0110：TIM2_BKIN 0111：LPTIM_TOG 0111~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
23:20	PAAFR5	端口PA5功能选择 0000：PA5 0001：TIM1_CH1 0010：SPI1_SCK 0011：I2C1_SDA 0100：TIM11_EXT 0101：SPI0_MISO 0110：BEEP 0111：TIM2_CH2N 1000：LIN1_TX 1001~1110：保留	0xF	R/W

		1111：模拟信号端口,高阻状态(默认功能)		
19:16	PAAFR4	端口PA4功能选择 注：PA4上电默认为SWDIO调试端口(默认功能) 0000：PA4 0001：TIM2_CH1 0010：SPI1_MISO 0011：UART1_RX 0100：TIM10_TOGN 0101：VC_OUT 0110~0111：保留 1000：LIN0_RX 1001~1110：保留 1111：模拟信号端口,高阻状态 备注：PA4端口默认为SWD调试端口SWDIO，要关闭SWDIO功能，需配置RCC_SWDIOCR寄存器。	0xF	R/W
15:12	PAAFR3	端口PA3功能选择 0000：PA3 0001：TIM1_CH1 0010：I2C1_SCL 0011：CAN0_TX 0100：TIM11_EXT 0101：UART0_TX 0110：BEEP 0111：PCA_CH3 1000：LIN0_TX 0111~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
11:8	PAAFR2	端口PA2功能选择 0000：PA2 0001：TIM2_CH2 0010：I2C1_SDA 0011：CAN0_RX 0100：TIM11_GATE 0101：UART0_RX 0110：LPTIM_GATE 0111：1-Wire 1000：LIN0_RX 0111~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W

7:4	PAAFR1	端口PA1功能选择 0000：PA1 0001：TIM2_CH1 0010：I2C1_SDA 0011：UART1_TX 0100：TIM11_TOGN 0101：SPI1_MOSI 0110：TIM1_CH1 0111：1-Wire 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
3:0	PAAFR0	端口PA0功能选择 0000：PA0 0001：TIM2_CH4 0010：I2C1_SCL 0011：UART1_RX 0100：TIM11_TOG 0101：SPI1_SCK 0110：TIM1_CH1N 0111~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W

11.5.19.2 GPIOB 复用配置

位	标记	功能描述	复位值	读写
31:28	PBAFR7	端口PB7功能选择 0000：PB7 0001：保留 0010：I2C0_SCL 0011~1110：保留 1111：模拟信号端口,高阻状态（默认功能） 备注：PB7端口支持高频晶振与低频晶振的OSC_IN/OSC32K_IN输入，具体配置参照RCC_HXTCR与RCC_LXTCR寄存器说明；	0xF	R/W
27:24	PBAFR6	端口PB6功能选择 0000：PB6 0001：保留 0010：I2C0_SDA 0011~1110：保留 1111：模拟信号端口，高阻状态（默认功能） 备注：PB7端口支持高频晶振与低频晶振的OSC_OUT/OSC32K_OUT输入，具体配置参照RCC_HXTCR与RCC_LXTCR寄存器说明；	0xF	R/W
23:20	PBAFR5	端口PB5功能选择 0000：PB5	0xF	R/W

		0001：TIM1_CH1N 0010：SPI0_SSN 0011：保留 0100：TIM10_TOGN 0101：CLK_MCO 0110：TIM2_CH2N 0111：VC_OUT 1000：1-Wire 1001~1110：保留 1111：模拟信号端口，高阻状态(默认功能)		
19:16	PBAFR4	端口PB4功能选择 0000：PB4 0001：TIM1_CH2 0010：SPI0_MISO 0011：保留 0100：TIM10_TOG 0101：LPUART_RX 0110：TIM2_CH3N 0111：PCA_CH1 1000：LIN0_RX 1001~1110：保留 1111：模拟信号端口，高阻状态(默认功能)	0xF	R/W
15:12	PBAFR3	端口PB3功能选择 0000：PB3 0001：TIM1_CH3 0010：SPI0_MOSI 0011：保留 0100：TIM10_EXT 0101：LPTIM_TOG 0110：TIM2_CH1N 0111：PCA_CH1 1000：LIN1_TX 1001~1110：保留 1111：模拟信号端口，高阻状态(默认功能)	0xF	R/W
11:8	PBAFR2	端口PB2功能选择 0000：PB2 0001：TIM1_CH3N 0010：SPI0_SCK 0011：保留 0100：TIM10_GATE 0101：LPTIM_TOGN 0110：TIM2_CH2N 0111：PCA_CH2 1000：LIN1_RX	0xF	R/W

		1001~1110：保留 1111：模拟信号端口，高阻状态(默认功能)		
7:4	PBAFR1	端口PB1功能选择 0000：PB1 0001：TIM1_CH1 0010：SPI0_MISO 0011：CAN0_TX 0100：UART0_TX 0101：LPUART_TX 0110：保留 0111：PCA_CH3 1000：LIN0_TX 1001~1110：保留 1111：模拟信号端口，高阻状态(默认功能)	0xF	R/W
3:0	PBAFR0	端口PB0功能选择 0000：PB0 0001：TIM1_CH1N 0010：SPI0_SSN 0011：CAN0_RX 0100：UART0_RX 0101：LPUART_RX 0110：LPTIM_EXT 0111：1-Wire 1000：LIN0_RX 1001~1110：保留 1111：模拟信号端口，高阻状态(默认功能)	0xF	R/W

11.5.19.3 GPIOC 复用配置

位	标记	功能描述	复位值	读写
31:28	PCAFR7	端口PC7功能选择 0000：PC7 0001：TIM2_CH2 0010：SPI1_MISO 0011：CAN0_TX 0100：UART1_TX 0101：保留 0110：TIM1_CH2 0111：PCA_ECI 1000：LIN1_TX 1001~1110：保留 1111：模拟信号端口，高阻状态（默认功能）	0xF	R/W
27:24	PCAFR6	端口PC6功能选择 0000：PC6 0001：TIM2_CH3	0xF	R/W

		0010：SPI1_SSN 0011：CAN0_RX 0100：UART1_RX 0101：保留 0110：TIM1_CH2N 0111：PCA_CH1 1000：LIN1_RX 1001~1110：保留 1111：模拟信号端口,高阻状态（默认功能）		
23:20	PCAFR5	端口PC5功能选择 0000：PC5 0001：TIM1_BKIN 0010：SPI1_SSN 0011：TIM1_CH3 0100：TIM11_TOGN 0101：RTC_CLKOUT 0110：BEEP 0111：TIM2_CH3N 1000：LIN1_RX 1000~1110：保留 1111：模拟信号端口,高阻状态（默认功能）	0xF	R/W
19:16	PCAFR4	端口PC4功能选择 注：PC4上电默认为SWDCLK调试端口(默认功能) 0000：PC4 0001：TIM2_CH4 0010：UART1_TX 0011：TIM1_CH3N 0100：TIM10_GATE 0101：RTC_CLKOUT 0110：保留 0111：TIM2_BKIN 1000：LIN0_TX 1001~1110：保留 1111：模拟信号端口,高阻状态 备注：PC4端口默认为SWD调试端口SWDCLK，要关闭SWDCLK功能，需配置RCC_SWDIOCR寄存器。	0xF	R/W
15:12	PCAFR3	端口PC3功能选择 0000：PC3 0001：TIM1_CH2N 0010：UART0_TX 0011：CAN0_TX 0100：保留 0101：LPUART_TX 0110：TIM2_CH4	0xF	R/W

		0111：保留 1000：LIN0_TX 1000~1110：保留 1111：模拟信号端口,高阻状态（默认功能）		
11:8	PCAFR2	端口PC2功能选择 0000：PC2 0001：TIM1_CH4 0010：UART0_RX 0011：CAN0_RX 0100：保留 0101：LPUART_RX 0110：保留 0111：保留 1000：LIN0_RX 1000~1110：保留 1111：模拟信号端口,高阻状态（默认功能）	0xF	R/W
7:4	PCAFR1	端口PC1功能选择 0000：PC1 0001：TIM1_CH3 0010：SPI0_MOSI 0011：I2C0_SCL 0100：TIM11_GATE 0101：LPTIM_GATE 0110：TIM2_CH1 0111：TIM2_CH4 1000~1110：保留 1111：模拟信号端口,高阻状态（默认功能）	0xF	R/W
3:0	PCAFR0	端口PC0功能选择 0000：PC0 0001：TIM1_CH1 0010：SPI1_SCK 0011：I2C1_SDA 0100：TIM11_EXT 0101：TIM2_CH3N 0110：TIM2_CH3 0111：PCA_CH3 1000~1110：保留 1111：模拟信号端口,高阻状态（默认功能）	0xF	R/W

11.5.19.4 GPIOD 复用配置

位	标记	功能描述	复位值	读写
31:28	PDAFR7	端口PD7功能选择 0000：PD7 0001：TIM1_CH3N 0010：UART1_TX 0011：CAN0_STDBY 0100：TIM10_GATE 0101：保留 0110：TIM2_CH1 0111：保留 1000：LIN1_TX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
27:24	PDAFR6	端口PD6功能选择 0000：PD6 0001：TIM1_CH3 0010：UART1_RX 0011：保留 0100：TIM10_EXT 0101：保留 0110：TIM2_CH2 0111：保留 1000：LIN1_RX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
23:20	PDAFR5	端口PD5功能选择 0000：PD5 0001：TIM1_CH3N 0010：I2C0_SDA 0011：TIM1_BKIN 0100：TIM10_TOGN 0101：VC_OUT 0110：LPTIM_EXT 0111：PCA_CH2 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
19:16	PDAFR4	端口PD4功能选择 0000：PD4 0001：TIM1_CH1N 0010：I2C1_SDA 0011：保留 0100：TIM11_TOGN 0101：保留	0xF	R/W

		0110：TIM1_BKIN 0111：PCA_CH3 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)		
15:12	PDAFR3	端口PD3功能选择 0000：PD3 0001：TIM1_CH3 0010：SPI1_SSN 0011：LPUART_TX 0100：TIM11_TOG 0101：NMI_b 0110：I2C1_SCL 0111：PCA_CH4 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
11:8	PDAFR2	端口PD2功能选择 0000：PD2 0001：TIM1_CH2 0010：SPI1_MISO 0011：CAN0_STDBY 0100：TIM11_GATE 0101：LPUART_RX 0110：I2C1_SDA 0111~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
7:4	PDAFR1	端口PD1功能选择 0000：PD1 0001：TIM1_CH1 0010：SPI1_MOSI 0011：LPUART_TX 0100：TIM10_EXT 0101：LPTIM_TOG 0110：TIM2_CH2N 0111：PCA_ECI 1000：LIN0_RX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
3:0	PDAFR0	端口PD0功能选择 0000：PD0 0001：TIM1_CH1N 0010：SPI1_SCK 0011：LPUART_RX 0100：TIM10_TOG 0101：保留	0xF	R/W

		0110：TIM2_CH2 0111：PCA_CH0 1000：LIN0_TX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)		

11.5.19.5 GPIOE 复用配置

位	标记	功能描述	复位值	读写
31:28	Res	保留	0xF	R/W
27:24	PEAFR6	端口PE6功能选择 0000：PE6(NRST) 0001~1110：保留 1111：模拟信号端口,高阻状态(默认功能) 备注：要选择NRST功能，需操作SYSCON_RSTCTRL寄存器。	0xF	R/W
23:20	PEAFR5	端口PE5功能选择 0000：PE5 0001：保留 0010：保留 0011：CAN0_RX 0100~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
19:16	PEAFR4	端口PE4功能选择 0000：PE4 0001：保留 0010：保留 0011：保留 0100：保留 0101：保留 0110：保留 0111：PCA_CH2 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
15:12	PEAFR3	端口PE3功能选择 0000：PE3 0001：TIM2_CH3 0010：保留 0011：CAN0_TX 0100：保留 0101：SPI1_SCK 0110：保留 0111：PCA_ECI 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W

11:8	PEAFR2	端口PE2功能选择 0000：PE2 0001：TIM2_CH2 0010：保留 0011：CAN0_RX 0100：保留 0101：SPI1_MOSI 0110~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
7:4	PEAFR1	端口PE1功能选择 0000：PE1 0001：TIM2_CH1 0010：保留 0011：保留 0100：保留 0101：SPI1_MISO 0110：保留 0111：PCA_CH4 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
3:0	PEAFR0	端口PE0功能选择 0000：PE0 0001：保留 0010：保留 0011：保留 0100：保留 0101：SPI1_SSN 0110：保留 0111：PCA_CH4 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W

11.5.20 GPIO 端口复用功能寄存器(GPIOx_AFR2) (x = A..D)

偏移地址：0x4C

复位值：0xFFFFFFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PxAFR15				PxAFR14				PxAFR13				PxAFR12			
R/W				R/W				R/W				R/W			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PxAFR11				PxAFR10				PxAFR9				PxAFR8			
R/W				R/W				R/W				R/W			

11.5.20.1 GPIOA 复用配置

位	标记	功能描述	复位值	读写
31:28	PAAFR15	端口PA15功能选择 0000：PA15 0001：TIM1_CH3 0010：SPI0_SCK 0011：保留 0100：TIM10_GATE 0101：LVD_OUT 0110：TIM2_CH2 0111：LPTIM_TOG 1000：LIN1_RX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
27:24	PAAFR14	端口PA14功能选择 0000：PA14 0001：TIM1_CH3N 0010：SPI0_MOSI 0011：保留 0100：保留 0101：HIRC_OUT 0110：TIM2_CH1 0111：PCA_CH4 1000：LIN1_TX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
23：20	PAAFR13	端口PA13功能选择 0000：PA13 0001：TIM1_CH3N 0010：SPI1_MOSI 0011：CAN0_TX 0100：TIM11_TOGN 0101：LPTIM_GATE 0110：TIM1_CH3	0xF	R/W

		0111~1111：保留 1111：模拟信号端口,高阻状态(默认功能)		
19:16	PAAFR12	端口PA12功能选择 0000：PA12 0001：TIM1_CH2N 0010：SPI1_SCK 0011：CAN0_RX 0100：TIM11_TOG 0101：I2C0_SDA 0110：TIM1_CH3N 0111：TIM2_CH1 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
15:12	PAAFR11	端口PA11功能选择 0000：PA11 0001：TIM1_CH1N 0010：UART1_TX 0011：SPI1_SSN 0100：TIM11_EXT 0101：I2C0_SCL 0110：TIM1_CH2 0111：TIM2_CH2 1000：LIN1_TX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
11:8	PAAFR10	端口PA10功能选择 0000：PA10 0001：TIM1_CH3 0010：UART1_RX 0011：SPI1_MISO 0100：TIM11_GATE 0101：I2C0_SDA 0110：TIM1_CH2N 0111：TIM2_CH3 1000：LIN1_RX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W

7:4	PAAFR9	端口PA9功能选择 0000：PA9 0001：TIM2_CH3 0010：SPI0_SCK 0011：TIM1_BKIN 0100：TIM11_EXT 0101：保留 0101：保留 0111：PCA_ECI 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
3:0	PAAFR8	端口PA8功能选择 0000：PA8 0001：TIM1_CH1 0010：I2C0_SCL 0011：CAN0_STDBY 0100：TIM10_EXT 0101：LIRC_OUT 0110：TIM2_CH1N 0111：PCA_CH0 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W

11.5.20.2 GPIOB 复用配置

位	标记	功能描述	复位值	读写
31:28	PBAFR15	端口PB15功能选择 0000：PB15 0001：TIM1_CH1N 0010：保留 0011：保留 0100：TIM11_TOG 0101：TIM2_BKIN 0110：TIM2_CH4 0111~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
27:24	PBAFR14	端口PB14功能选择 0000：PB14 0001：TIM2_CH4 0010：I2C0_SDA 0011：保留 0100：TIM10_GATE 0101：保留 0110：TIM2_CH3 0111：LPTIM_TOGN	0xF	R/W

		1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)		
23:20	PBAFR13	端口PB13功能选择 0000：PB13 0001：TIM1_CH3N 0010：I2C0_SDA 0011：CAN0_TX 0100：TIM11_EXT 0101：保留 0110：TIM2_CH3 0111：PCA_CH1 1000~1111：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
19:16	PBAFR12	端口PB12功能选择 0000：PB12 0001：TIM1_CH2N 0010：I2C1_SCL 0011：CAN0_RX 0100：TIM11_GATE 0101：保留 0110：保留 0111：PCA_CH2 1000~1111：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
15:12	PBAFR11	端口PB11功能选择 0000：PB11 0001：TIM2_CH3 0010：I2C1_SCL 0011：LPTIM_EXT 0100：TIM10_GATE 0101：保留 0110：LPTIM_TOG 0111：PCA_CH1 1000~1111：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
11:8	PBAFR10	端口PB10功能选择 0000：PB10 0001：TIM2_CH2 0010：I2C1_SDA 0011：SPI1_SSN 0100：TIM10_TOGN 0101：CLK_MCO 0110：LPTIM_TOGN 0111：PCA_CH2	0xF	R/W

		1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)		
7:4	PBAFR9	端口PB9功能选择 0000：PB9 0001：TIM1_CH2N 0010：SPI1_MISO 0011：CAN0_RX 0100：TIM10_TOGN 0101：保留 0110：保留 0111：保留 1000：LIN0_RX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
3:0	PBAFR8	端口PB8功能选择 0000：PB8 0001：TIM1_CH2 0010：SPI1_SCK 0011：CAN0_TX 0100：TIM10_GATE 0101：LPTIM_GATE 0110：保留 0111：1-Wire 1000：LIN0_TX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W

11.5.20.3 GPIOC 复用配置

位	标记	功能描述	复位值	读写
31:28	PCAFR15	端口PC15功能选择 0000：PC15 0001：TIM2_CH2 0010：SPI0_SSN 0011：I2C1_SDA 0100：TIM11_TOG 0101~1110：保留 1111：模拟信号端口,高阻状态（默认功能）	0xF	R/W
27:24	PCAFR14	端口PC14功能选择 0000：PC14 0001：TIM2_CH1 0010：SPI0_MISO 0011：I2C1_SCL 0100：TIM11_TOGN 0101~1110：保留 1111：模拟信号端口,高阻状态（默认功能）	0xF	R/W
23：20	PCAFR13	端口PC13功能选择 0000：PC13 0001：TIM1_CH2 0010：SPI1_MOSI 0011：I2C1_SCL 0100：TIM11_TOG 0101：SPI0_SSN 0110：保留 0111：TIM2_CH1N 1000~1110：保留 1111：模拟信号端口,高阻状态（默认功能）	0xF	R/W
19:16	PCAFR12	端口PC12功能选择 0000：PC12 0001：保留 0010：SPI1_MISO 0011：CAN0_RX 0100：保留 0101：保留 0110：LPTIM_EXT 0111：LPTIM_TOGN 1000：LIN1_RX 1001~1110：保留 1111：模拟信号端口,高阻状态（默认功能）	0xF	R/W

15:12	PCAFR11	端口PC11功能选择 0000：PC11 0001：TIM1_CH4 0010：SPI1_SSN 0011：CAN0_TX 0100：保留 0101：保留 0110：保留 0111：LPTIM_TOG 1000：LIN1_TX 1001~1110：保留 1111：模拟信号端口,高阻状态（默认功能）	0xF	R/W
11:8	PCAFR10	端口PC10功能选择 0000：PC10 0001：TIM1_CH1N 0010：UART0_RX 0011：保留 0100：TIM10_GATE 0101：保留 0110：保留 0111：LPTIM_TOGN 1000~1110：保留 1111：模拟信号端口,高阻状态（默认功能）	0xF	R/W
7:4	PCAFR9	端口PC9功能选择 0000：PC9 0001：TIM1_CH2 0010：UART1_TX 0011：LPUART_TX 0100：TIM10_EXT 0101： RTC_CLKOUT 0110：TIM2_CH1 0111：LPTIM_GATE 1000：LIN1_TX 1001~1110：保留 1111：模拟信号端口,高阻状态（默认功能）	0xF	R/W

3:0	PCAFR8	端口PC8功能选择 0000：PC8 0001：TIM1_CH2N 0010：UART1_RX 0011：保留 0100：TIM10_TOGN 0101： LPUART_RX 0110：TIM2_CH2 0111：保留 1000：LIN1_RX 1001~1110：保留 1111：模拟信号端口,高阻状态（默认功能）	0xF	R/W

11.5.20.4 GPIOD 复用配置

位	标记	功能描述	复位值	读写
31:28	PDAFR15	端口PD15功能选择 0000：PD15 0001：保留 0010：保留 0011：I2C1_SDA 0100：保留 0101：保留 0110：保留 0111：LPTIM_TOGN 1000：保留 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
27:24	PDAFR14	端口PD14功能选择 0000：PD14 0001：保留 0010：保留 0011：I2C1_SCL 0100：保留 0101：保留 0110：保留 0111：LPTIM_TOG 1000：保留 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
23:20	PDAFR13	端口PD13功能选择 0000：PD13 0001：TIM1_CH2 0010：保留	0xF	R/W

		0011：保留 0100：保留 0101：保留 0110：TIM2_CH4 0111~1110：保留 1111：模拟信号端口,高阻状态(默认功能)		
19:16	PDAFR12	端口PD12功能选择 0000：PD12 0001：TIM1_CH4 0010：保留 0011：保留 0100：保留 0101：保留 0110：TIM2_CH2 0111~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
15:12	PDAFR11	端口PD11功能选择 0000：PD11 0001：TIM1_CH2 0010：I2C0_SCL 0011：UART1_RX 0100：LPTIM_TOGN 0101：保留 0110：保留 0111：保留 1000：LIN1_RX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
11:8	PDAFR10	端口PD10功能选择 0000：PD10 0001：TIM1_CH1 0010：I2C0_SDA 0011：UART1_TX 0100：LPTIM_TOG 0101：保留 0110：保留 0111：保留 1000：LIN1_TX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W
7:4	PDAFR9	端口PD9功能选择 0000：PD9 0001：TIM1_CH1 0010：保留	0xF	R/W

		0011：CAN0_RX 0100：保留 0101：保留 0110：LPTIM_TOG 0111：PCA_CH3 1000：LIN0_RX 1001~1110：保留 1111：模拟信号端口,高阻状态(默认功能)		
3:0	PDAFR8	端口PD8功能选择 0000：PD8 0001：TIM1_CH4 0010：TIM1_BKIN 0011：UART1_RX 0100：保留 0101：保留 0110：TIM2_BKIN 0111：VC_OUT 1000~1110：保留 1111：模拟信号端口,高阻状态(默认功能)	0xF	R/W

11.5.21 GPIO 施密特触发输入选择寄存器(GPIOx_CS) (x = A..E)

偏移地址：0x50

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15	PxCS15	管脚 Px15 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
14	PxCS14	管脚 Px14 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
13	PxCS13	管脚 Px13 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
12	PxCS12	管脚 Px12 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
11	PxCS11	管脚 Px11 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
10	PxCS10	管脚 Px10 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
9	PxCS9	管脚 Px9 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
8	PxCS8	管脚 Px8 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
7	PxCS7	管脚 Px7 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
6	PxCS6	管脚 Px6 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W

5	PxCS5	管脚 Px5 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
4	PxCS4	管脚 Px4 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
3	PxCS3	管脚 Px3 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
2	PxCS2	管脚 Px2 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
1	PxCS1	管脚 Px1 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W
0	PxCS0	管脚 Px0 施密特触发输入选择 0：施密特触发输入(复位状态) 1：CMOS 输入	0x0	R/W

12 MATH 协处理器

12.1 简介

MATH 协处理器模块包含两个独立的子单元，来支持 CPU 完成数学密集型运算：一个是能完成有符号和无符号 32 位除法器的除法器单元（DIV），另一个是能进行三角函数的 CORDIC 协处理器。其中 CORDIC 单元可实现正弦 SIN、余弦 COS、反正切 ARCTAN 的运算。

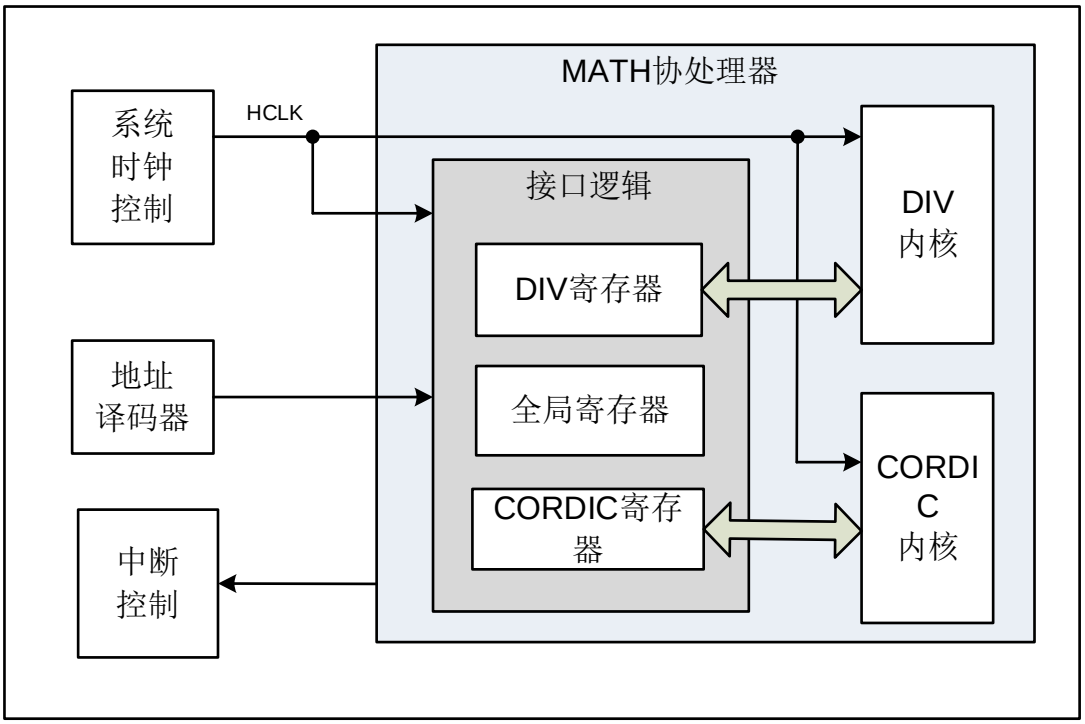


图 12-1 MATH 协处理器框图

12.2 除法器单元(DIV)

12.2.1 DIV 主要特性

DIV 支持以下特性：

- 有符号或无符号整数除法运算，有符号数为补码输入
- 32 位除数和被除数，输出 32 位的商和余数
- 8 个 HCLK 周期完成运算
- 除数为零警告标志位，除法运算结束标志位，可触发中断
- 支持写除数自动执行除法运算
- 读商和余数寄存器时自动等待运算结束

12.2.2 DIV 功能介绍

硬件除法单元包括 4 个 32 位数据寄存器，分别为被除数，除数，商和余数，可以做有符号或者无符号的 32 位除法运算。通过硬件除法控制寄存器 DIV_CON.USIGN 可以选择是有符号除法还是无符号除法。

要使用 DIV 执行一次除法运算，首先要按下述操作配置该除法：

- 通过 DIV_CON.USIGN 位选择有符号或无符号除法

- 通过 DIV_CON.STMODE 位选择启动模式

然后向 DIV_DVD 与 DIV_DVS 寄存器写入被除数与除数的值。根据启动模式不同，除法运算可以通过写 DIV_DVS 寄存器启动，也可以通过将启动位 DIV_CON.ST 置 1 来启动。如果使用 ST 位启动，则该位在下一个内核时钟周期被自动清零。除法运算启动后忙标志 DIV_ST.BSY 被置 1。

在运算结束后，结果会写入到商和余数寄存器里，并且 DIV_ST.BSY 标志被清零。

如果在结束前读商寄存器、余数寄存器，读操作会被暂停，直到结束才返回运算结果。

如果除数为零，会产生除数为零标志位 DIV_ST.DVS_ZERO。计算结束事件将除法器运算结束标志 MATH_INTST.DIV_INTF 置 1，并且向 NVIC 申请一次中断请求（如果使能了该中断），该标志只能通过软件对 MATH_INTCL.DIV_INTC 位写操作来清零。

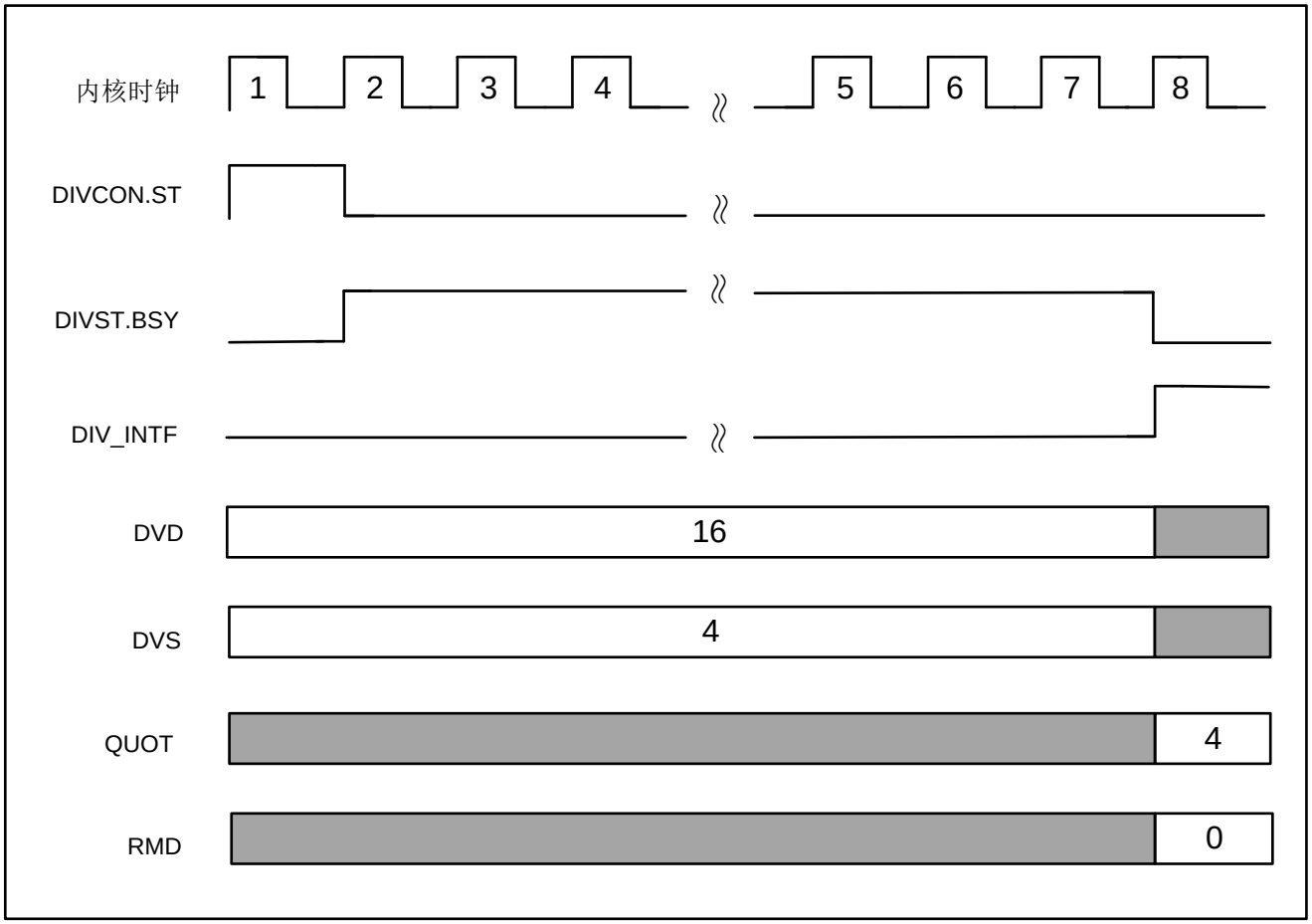


图 12-2 除法运算时序图

12.2.3 DIV 操作流程

1. 在 AHB 周边模块时钟使能寄存器 RCC_HCLKEN.MATHCKE 中打开 MATH 的时钟使能。
2. 配置寄存器 DIV_CON.USIGN 设置有符号/无符号除法运算。
3. 配置寄存器 DIV_DVD 设置被除数。
4. 配置寄存器 DIV_DVS 设置除数。
5. 除法运算开始，读寄存器 DIV_QUOT 得到商，读寄存器 DIV_RMD 得到余数；也可通过 CPU 响应中断的方式读取数据，需要提前打开 DIV 中断使能；也可通过查询运

算结束标志的方式读取数据，寄存器 MATH_INTST 的运算结束标志位 DIV_INTF 为 1 标志运算结束。

6. 当除数为零时，除数为零警告标志位 DVS_ZERO 被置起。
7. 在除法运算结束之前，读寄存器 DIV_QUOT/DIV_RMD 时， CPU 将被保持直到运算结束。

举例：计算一个无符号除法，被除数为 4278256450(0xFF01_0342)，除数为 262158(0x0004_000E)

步骤一：配置寄存器 RCC_HCLKEN.MATHCKE 为 1，打开 MATH 模块时钟使能。

步骤二：配置寄存器 DIV_CON.USIGN 为 1，选择无符号除法运算。

步骤三：配置寄存器 DIV_DVD 为 0xFF01_0342(4278256450)，即设置被除数。

步骤四：配置寄存器 DIV_DVS 为 0x0004_000E(262158)，即设置除数，计算开始。

步骤五：读寄存器 DIV_QUOT 得到商 0x0000_3FBF(16319)，读寄存器 DIV_RMD 得到余数 0x0001_86D0(100048)。

12.3 CORDIC 协处理器

12.3.1 CORDIC 主要特性

CORDIC 协处理器支持以下特性：

- 圆旋转模式：支持满范围[0,216-1]内的初始 CORDZ 数据值，代表 0~360 度，来计算 SIN、COS，输出结果范围为[-(215-1)☹215-1)]，按照 Q15 数据格式，表示[-1,1]。
- 圆向量模式：支持满范围[-(215-1)☹215-1)]的初始 CORDX、CORDY 数据值，来计算角度 ARCTAN，输出结果为[0,216-1]，代表 0~360 度。
- 每次运算需要 12 个内核 HCLK 时钟周期。
- 精度：SIN/COS 计算，精度为±0.001；ARCTAN 计算，精度为±0.05。
- 一次计算完成时，可向 CPU 申请中断。

12.3.2 CORDIC 功能介绍

- 向 CORDIC_CON.ST 位写 1 可启动一次新的计算。当一次计算正在进行时，CORDIC_CON.ST 一直为 1，表示处于忙状态。在一次计算结束时，CORDIC_CON.ST 位被硬件清零，指示运算完成的 MATH_INTST.CORDIC_INTF 被置 1。
- 需要在 CORDIC_CON.ST 为 0 且 MATH_INTST.CORDIC_INTF 为 1 期间读取结果数据。在 CORDIC_CON.ST 为 1 期间，对结果寄存器 SIN/COS/ARCTAN 的任何一次访问都会导致内核发出一个总线等待，直到 CORDIC_CON.ST 位被清零。
- 在运算结束时，指示运算完成的 MATH_INTST.CORDIC_INTF 被置 1，如果使能了 CORDIC 中断，则 MATH 可向 CPU 申请中断，MATH_INTST.CORDIC_INTF 标志必须由软件清零。
- 在一次运算正在进行期间，对 CORDIC_CON.ST 位写 1 不起作用。要启动一次新的运算，必须等到 CORDIC_CON.ST 不再有效时重新置位 CORDIC_CON.ST。

- 输入寄存器定义域

1) CORDZ 为[15：0]位宽，支持满范围[0,216 -1]内的初始 CORDZ 数据值，表示 0~360 度，用于计算 SIN(Z)、COS(Z)；
2) CORDX 为[15：0]位宽，表示四个象限的值，其中[15]为符号位，CORDX 不能为 0，用于计算 ARCTAN(Y/X)；
3) CORDY 为[15：0]位宽，表示四个象限的值，其中[15]为符号位，用于计算 ARCTAN(Y/X)；
4) CORDX 与 CORDY 为平面坐标系中半径为 32767 圆环上的任一点的 X、Y 坐标值。

- SIN/COS 计算方法

通过 CORDZ 输入角度，其中 0xFFFF 表示 360 度，0x0000 表示 0 度。
计算结果如下：
结果寄存器 COS 中，COS[15]为符号位，COS 结果为 COS/(2^15-1);
结果寄存器 SIN 中，SIN[15]为符号位，SIN 结果为 SIN/(2^15-1);

- ARCTAN 计算方法

通过 CORDX、CORDY 输入相应的数值，函数对应关系为 TANθ=CORDY/CORDX。
计算结果：θ= ARCTAN/(2^16-1) * 360。

12.3.3 CORDIC 操作流程

1. 在 AHB 周边模块时钟使能寄存器 RCC_HCLKEN.MATHCKE 中打开 MATH 的时钟使能。
2. 配置寄存器 CORDIC_CON.ROTVEC，选择计算模式。
3. 在向量模式，配置寄存器 CORDX 与 CORDY，用于计算 ARCTAN(Y/X)；在旋转模式，配置寄存器 CORDZ，用于计算 SIN(Z)、COS(Z)。
4. 向寄存器 CORDIC_CON.ST 位写 1 启动一次新的计算。
5. 计算结果读取有三种方式：（1）运算开始后，直接读取 SIN/COS 或 ARCTAN 的结果；（2）通过查询寄存器 MATH_INTST 运算结束标志位 CORDIC_INTF，CORDIC_INTF 为 1 表示计算结束，在 CORDIC_CON.ST 为 0 且 MATH_INTST.CORDIC_INTF 为 1 期间读取结果数据 SIN/COS 或 ARCTAN。（3）通过 CPU 响应中断的方式读取数据，需要提前打开 CORDIC 中断使能。
6. 在计算结束之前，读寄存器 SIN/COS 或 ARCTAN 时，CPU 将被保持直到运算结束。

举例 1：计算 SIN(Z)、COS(Z)，CORDZ 为 5461 (对应角度为 30°)。

步骤一：配置寄存器 RCC_HCLKEN. MATHCKE 为 1，打开 MATH 模块时钟使能。
步骤二：配置寄存器 CORDIC_CON.ROTVEC 为 1，选择旋转模式。
步骤三：配置寄存器 CORDZ 为 5461，即设置角度。
步骤四：配置寄存器 CORDIC_CON.ST 位为 1，启动一次新的计算。

步骤五：读取结果数据 SIN=16388(对应 0.5001),COS=28374(对应 0.8659)。

举例 2：计算 ARCTAN(Y/X)，Y 为 23170，X 为 23170。

步骤一：配置寄存器 RCC_HCLKEN. MATHCKE 为 1，打开 MATH 模块时钟使能。

步骤二：配置寄存器 CORDIC_CON.ROTVEC 为 0，选择向量模式。

步骤三：配置寄存器 CORDX 为 23170，CORDY 为 23170。

步骤四：配置寄存器 CORDIC_CON.ST 位为 1，启动一次新的计算。

步骤五：读取结果数据 ARCTAN=8191(对应 45°)。

12.4 寄存器列表

基地址：0x4002 0C00

偏移地址	名称	描述	复位值
0x00	DIV_CON	DIV控制寄存器	0x0000 0000
0x04	DIV_DVD	DIV被除数寄存器	0x0000 0000
0x08	DIV_DVS	DIV除数寄存器	0x0000 0000
0x0C	DIV_QUOT	DIV商寄存器	0x0000 0000
0x10	DIV_RMD	DIV余数寄存器	0x0000 0000
0x14	DIV_ST	DIV状态寄存器	0x0000 0000
0x18	CORDIC_CON	CORDIC控制寄存器	0x0000 0000
0x1C	CORDIC_CORDX	CORDIC X数据寄存器	0x0000 0000
0x20	CORDIC_CORDY	CORDIC Y数据寄存器	0x0000 0000
0x24	CORDIC_CORDZ	CORDIC Z数据寄存器	0x0000 0000
0x28	CORDIC_COS	CORDIC COS寄存器	0x0000 0000
0x2C	CORDIC_SIN	CORDIC SIN寄存器	0x0000 0000
0x30	CORDIC_ARCTAN	CORDIC ARCTAN寄存器	0x0000 0000
0x34	MATH_INTEN	MATH中断使能寄存器	0x0000 0000
0x38	MATH_INTCL	MATH中断清除寄存器	0x0000 0000
0x3C	MATH_INTST	MATH中断状态寄存器	0x0000 0000

12.5 寄存器说明

12.5.1 DIV 控制寄存器(DIV_CON)

地址偏移：0x00

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留													USIGN	STMODE	ST
													R/W	R/W	WO

位	标记	功能描述	复位值	读写
31:3	Res	保留	0x0	–
2	USIGN	无符号除法使能 0：选择有符号除法 1：选择无符号除法	0x0	R/W
1	STMODE	除法器启动模式选择 0：写 DVS 寄存器会自动启动一次计算 1：通过将 ST 位置 1 来启动计算 注：如果 BSY=1，启动一次新的除法运算的请求将被忽略。	0x0	R/W
0	ST	除法器运算启动位 0：无效 1：当 STMODE=1 时启动除法运算。该位在一个内核时钟周期后被硬件自动清零。	0x0	WO

12.5.2 被除数寄存器(DIV_DVD)

地址偏移：0x04

复位值：0x0000 0000

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

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

位	标记	功能描述	复位值	读写
31:0	DVD	被除数寄存器	0x0	R/W

12.5.3 除数寄存器(DIV_DVS)

地址偏移：0x08

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:0	DVS	除数寄存器	0x0	R/W

12.5.4 商寄存器(DIV_QUOT)

地址偏移：0x0C

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:0	QUOT	商寄存器	0x0	RO

12.5.5 余数寄存器(DIV_RMD)

地址偏移：0x10

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:0	RMD	余数寄存器	0x0	RO

12.5.6 DIV 状态寄存器(DIV_ST)

地址偏移：0x14

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:2	Res	保留	0x0	–
1	BSY	除法器忙指示 0：除法器不在执行运算 1：除法器仍在执行运算	0x0	RO
0	DVS_ZERO	除法器为0标志位 0：合法运算 1：除数为0，非法运算	0x0	RO

12.5.7 CORDIC 控制寄存器(CORDIC_CON)

地址偏移：0x18

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:2	Res	保留	0x0	–
1	ROTVEC	旋转向量选择 0：向量模式（默认，计算ARCTAN） 1：旋转模式,计算SIN、COS	0x0	R/W
0	ST	CORDIC运算启动位 0：写0无效 1：对该位写1启动一次运算，且在运算期间，该位保持为1，运算结束后由硬件清零。	0x0	R/W

12.5.8 CORDIC X 数据寄存器(CORDX)

地址偏移：0x1C

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	-
15:0	CORDX	ARCTAN计算输入X坐标寄存器； CORDX为16位有符号定点数，其中1bit符号位，15bit整数位，表示范围(–32767 ~ 32767)	0x0	R/W

12.5.9 CORDIC Y 数据寄存器(CORDY)

地址偏移：0x20

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	-
15:0	CORDY	ARCTAN计算输入X坐标寄存器； CORDY为16位有符号定点数，其中1bit符号位，15bit整数位，表示范围(–32767 ~ 32767)。	0x0	R/W

12.5.10 CORDIC Z 数据寄存器(CORDZ)

地址偏移：0x24

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	-
15:0	CORDZ	CORDIC SIN/COS输入角度寄存器； 表示范围（0~65535）对应（0~360°）。	0x0	R/W

12.5.11 COS 结果寄存器(COS)

地址偏移：0x28

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	-
15:0	COS	COS结果寄存器； 表示范围（-32767~32767）对应（-1~1）。	0x0	RO

12.5.12 SIN 结果寄存器(SIN)

地址偏移：0x2C

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	-
15:0	SIN	SIN结果寄存器； 表示范围（-32767~32767）对应（-1~1）。	0x0	RO

12.5.13 ARCTAN 结果寄存器(ARCTAN)

地址偏移：0x30

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	-
15:0	ARCTAN	ARCTAN结果寄存器； 表示范围为（0~65535）对应（0~360°）	0x0	RO

12.5.14 MATH 中断使能寄存器(MATH_INTEN)

地址偏移：0x34

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:2	Res	保留	0x0	–
1	CORDIC_IEN	CORDIC中断使能配置 0：禁止 1：使能	0x0	R/W
0	DIV_IEN	DIV中断使能配置 0：禁止 1：使能	0x0	R/W

12.5.15 MATH 中断清除寄存器(MATH_INTCL)

地址偏移：0x38

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:2	Res	保留	0x0	–
1	CORDIC_INTC	CORDIC中断状态清除配置 0：无作用 1：CORDIC中断状态标志位清零	0x0	WO
0	DIV_INTC	DIV中断状态清除配置 0：无作用 1：DIV中断状态标志位清零	0x0	WO

12.5.16 MATH 中断状态寄存器(MATH_INTST)

地址偏移：0x3C

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:2	Res	保留	0x0	–
1	CORDIC_INTF	CORDIC中断状态标志位 0：运算未完成 1：运算完成，如INTEN使能，则申请中断	0x0	RO
0	DIV_INTF	DIV中断状态标志位 0：运算未完成 1：运算完成，如INTEN使能，则申请中断	0x0	RO

13 循环冗余校验 CRC

13.1 概述

循环冗余校验(CRC)计算单元是根据固定的生成多项式得到任意字节数据的 CRC 计算结果。在应用中，CRC 技术主要应用于核实数据传输或者数据存储的正确性和完整性。以下示意了 CRC 算法在数据传输中的一个最典型的应用：

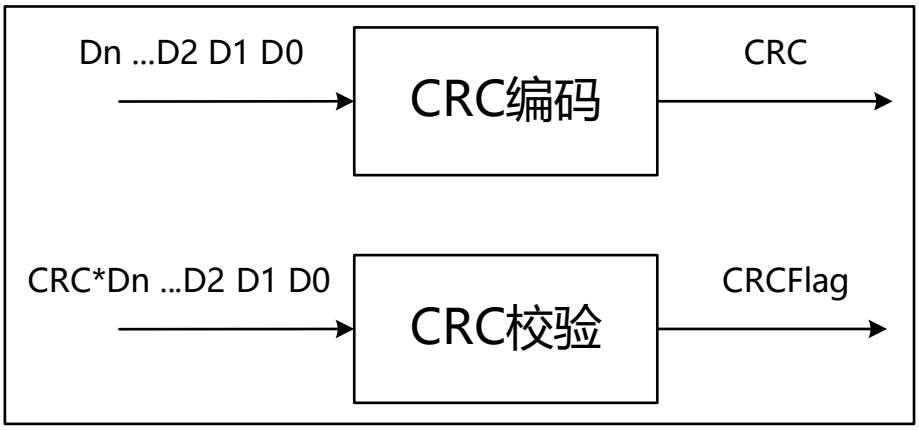


图 13-1 CRC 应用示意图

13.2 功能描述

本模块算法遵从 ISO/IEC13239 的定义，采用 16 位长度的 CRC，计算多项式为:

$$x^{16} + x^{12} + x^5 + 1$$

计算初始值为 0xFFFF。

本模块功能包括：

- CRC 编码和 CRC 校验
- 3 种位宽访问方式 8 位、16 位、32 位
- 8 位位宽下输入数据示例为 0x00,0x11,0x22,0x33,0x44,0x55,0x66,0x77
- 16 位位宽下输入数据示例为 0x1100,0x3322,0x5544,0x7766
- 32 位位宽下输入数据示例为 0x33221100,0x77665544

13.2.1 CRC 编码模式

编码模式可以对原始数据编码以计算其 CRC 值，操作流程如下所示：

Step1：向 CRC_RESULT 写入 0xFFFF，初始化 CRC 计算。

Step2：将待编码的原始数据按 8 位/16 位/32 位的组织方式，依次写入 CRC_DATA 寄存器。

Step3：读取 CRC_RESULT，即为 CRC 值。

13.2.2 CRC 检验模式

检验模式可以检验已编码的数据是否被篡改，操作流程如下所示

Step1：向 CRC_RESULT 写入 0xFFFF，初始化 CRC 计算。

Step2：将已编码的数据按 8 位/16 位/32 位的组织方式，依次写入 CRC_DATA 寄存器。

注：按 8 位组织方式写 CRC 值到 CRC_DATA 寄存器时，应先写入低 8 位，后写入高 8 位。

Step3：读取 CRC_RESULT.FLAG，以判定 CRC 校验是否成功。

13.3 寄存器列表

基地址：0x40020800

表 13-1 CRC 寄存器列表和复位值

偏移地址	名称	描述	复位值
0x04	CRC_RESULT	CRC 结果寄存器，计算完成后对该寄存器读取即获得结果。	0x00000000
0x80-0xFF	CRC_DATA	CRC 数据寄存器，用于输入需要运算的数据。	0x00000000

13.4 寄存器说明

13.4.1 CRC 结果寄存器(CRC_RESULT)

偏移地址：0x04

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:17	Res	保留	0x0	-
16	FLAG	校验结果标志；0：当前校验错误，1：当前校验正确。寄存器[16]是一个只读位，对其进行写操作不会产生影响。进行 CRC 校验时，应该在所有的数据和 16 位 CRC 编码输入数据寄存器之后读取本位，如果为 1 则表明校验成功。	0x0	RO
15:0	RESULT	本寄存器用于每次 CRC 计算结果的更新和保存。运算后，读取本寄存器将得到 16 位的 CRC 编码结果。 根据标准规定，运算完成后，16 位的 CRC 编码值是运算寄存器取反后的结果，因此本寄存器的读取将得到本寄存器当前的取反值。	0x0	R/W

注意：

- 根据标准规定，运算完成后，16 位的 CRC 编码值是运算寄存器取反后的结果，因此，对本寄存器位 15~0 的读取将得到本寄存器当前位 15~0 的取反值。
- CRC_RESULT[16]是一个只读的位，对之进行写操作不会产生影响。进行 CRC 校验运算时，应该在所有的数据和 16 位 CRC 编码输入数据寄存器后读取本位，如果为1 则表明校验成功。

13.4.2 CRC 数据寄存器(CRC_DATA)

偏移地址：0x80-0xFF

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:0	CRC_DATA	本寄存器用于输入需要运算的数据。 本寄存器的地址是一个范围(0x80-0xFF)，对该范围内的任何一个地址进行操作都会认为是对本寄存器进行操作。这样定义的目的就是为了方便软件可以用 STM 指令对本寄存器进行连续的 32 位数据写入操作，以加快运算速度。本寄存器支持 8/16/32 位的输入方式。	0x0	WO

注意：

本寄存器的地址是一个范围(8'h80-8'hFF)，对该范围内的任何一个地址进行操作都会认为是对本寄存器进行操作。这样定义的目的就是方便软件可以用 STM 指令对本寄存器进行连续的 32 位数据写入操作，以加快运算速度。同时本寄存器也支持 8/16 位的输入方式。

14 高级控制定时器(TIMx)(x=1.2)

14.1 TIMx 简介

高级控制定时器(TIMx)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。

14.2 TIMx 主要特性

TIMx 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 4 个独立通道：
  - 输入捕获
  - 输出比较
  - PWM 生成(边缘或中间对齐模式)
  - 单脉冲模式输出
- 死区时间可编程的互补输出
- 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：
  - 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

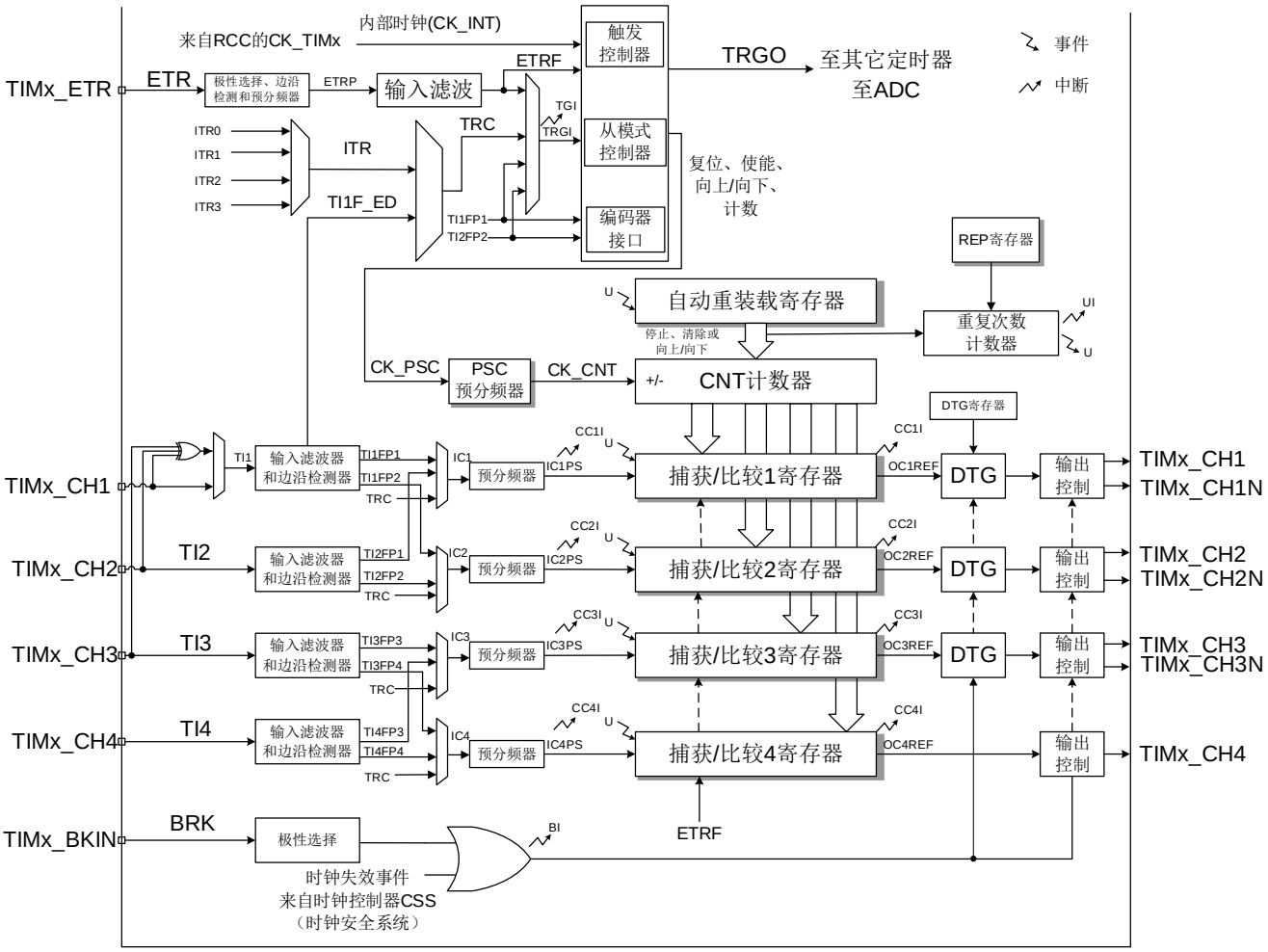


图 14-1 高级控制定时器框图

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

14.3 TIMx 功能描述

14.3.1 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(TIMx_CNT)
- 预分频器寄存器(TIMx_PSC)
- 自动装载寄存器(TIMx_ARR)
- 重复次数寄存器(TIMx_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 TIMx_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 TIMx_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。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 TIMx_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 TIMx_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

14.3.2 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。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 14-2 和图 14-3 给出了在预分频器运行时，更改计数器参数的例子。

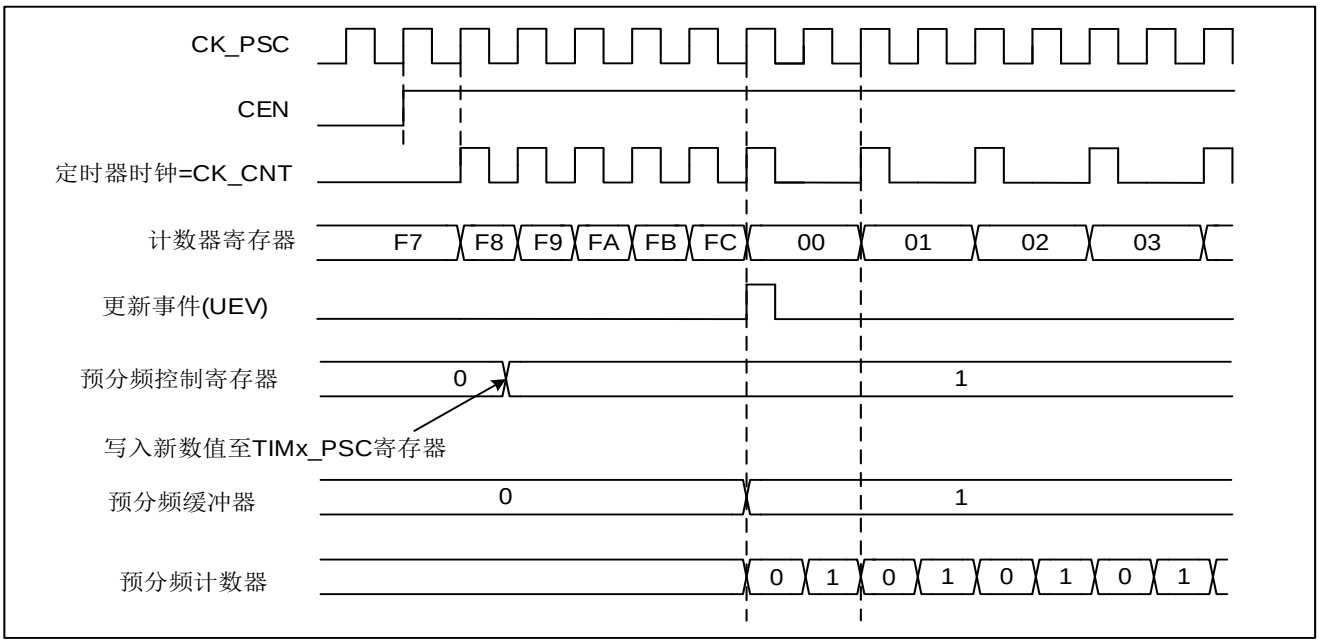


图 14-2 当预分频器的参数从 1 变到 2 时，计数器的时序图

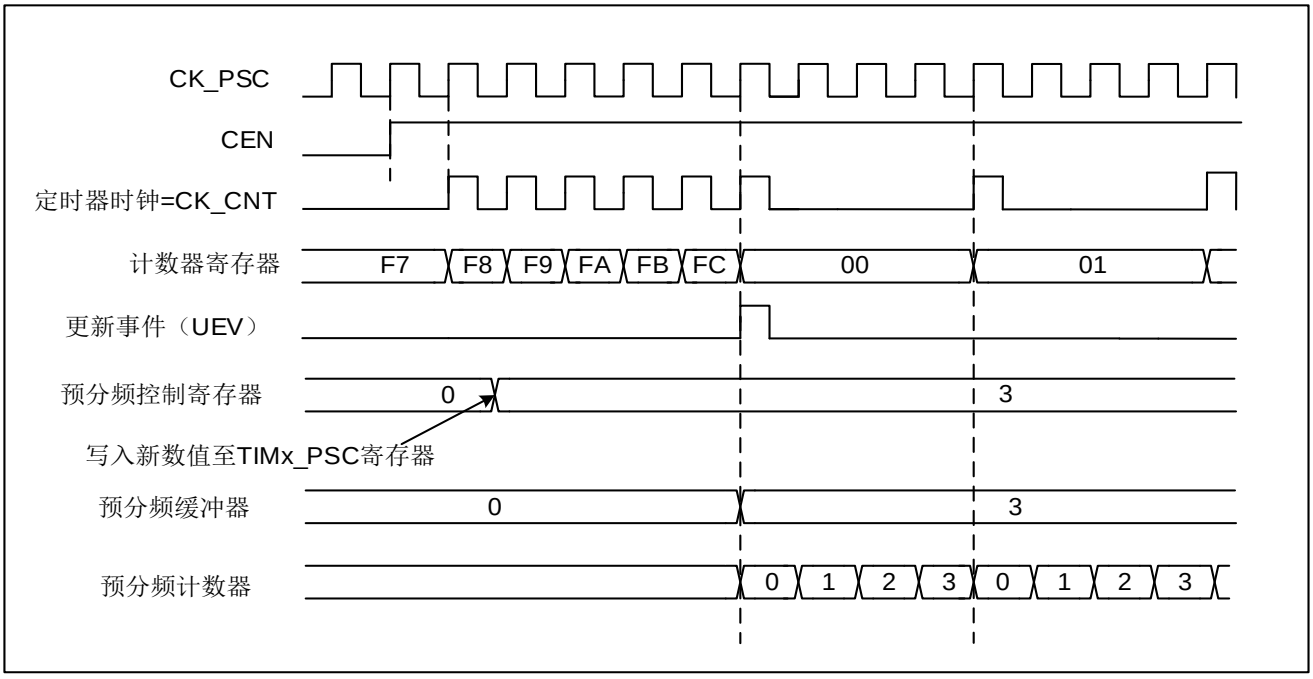


图 14-3 当预分频器的参数从 1 变到 4 时，计数器的时序图

14.3.3 计数器模式

14.3.3.1 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(TIMx_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(TIMx_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 TIMx 事件产生寄存器(TIMx_EGR)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 TIMx 控制寄存器 1(TIMx_CR1)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清 0(但预分频器的数值不变)。此外，如果设置了 TIMx 控制寄存器 1(TIMx_CR1)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)。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(TIMx_SR 寄存器中的 UIF 位)。

- 重复计数器被重新加载为 TIMx_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(TIMx_ARR)。
- 预分频器的缓冲区被置入预装载寄存器的值(TIMx_PSC 寄存器的内容)。

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

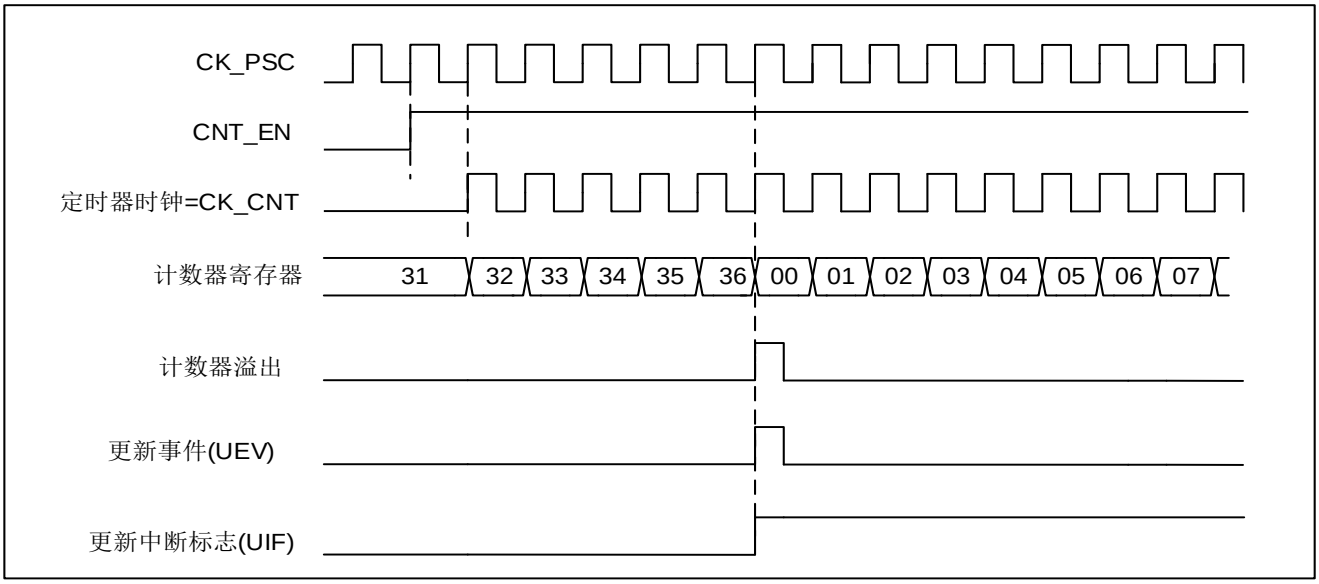


图 14-4 计数器时序图，内部时钟分频因子为 1

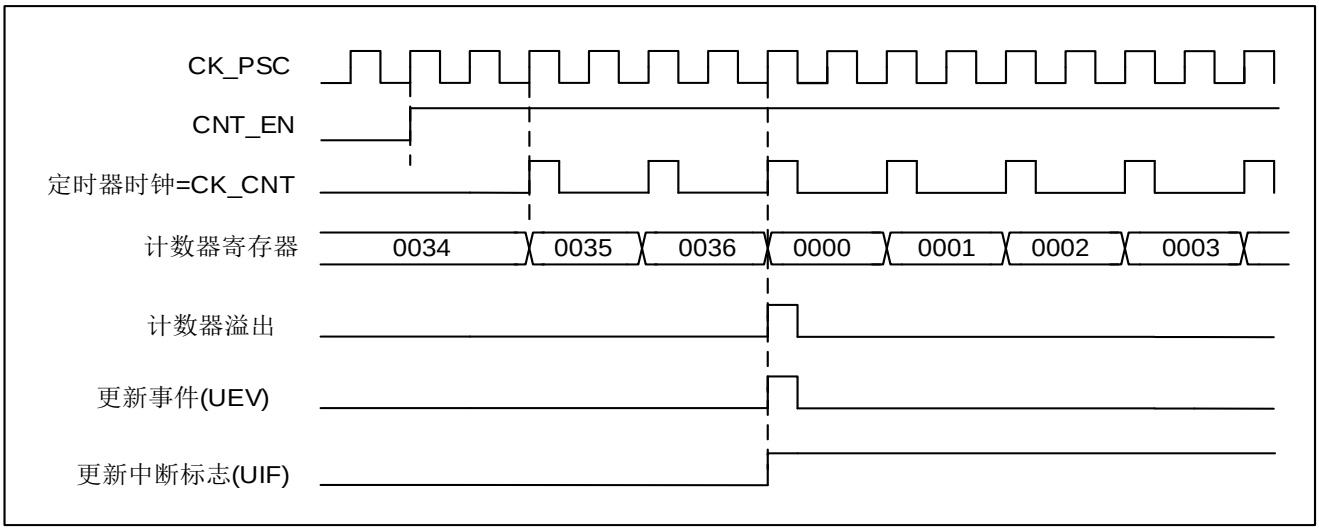


图 14-5 计数器时序图，内部时钟分频因子为 2

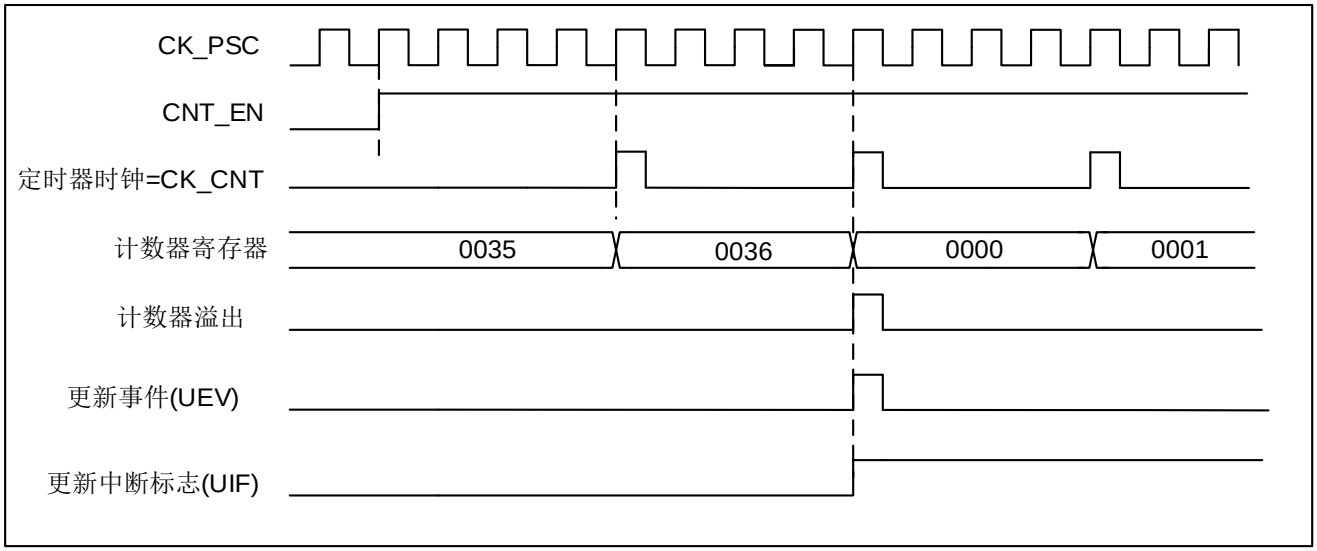


图 14-6 计数器时序图，内部时钟分频因子为 4

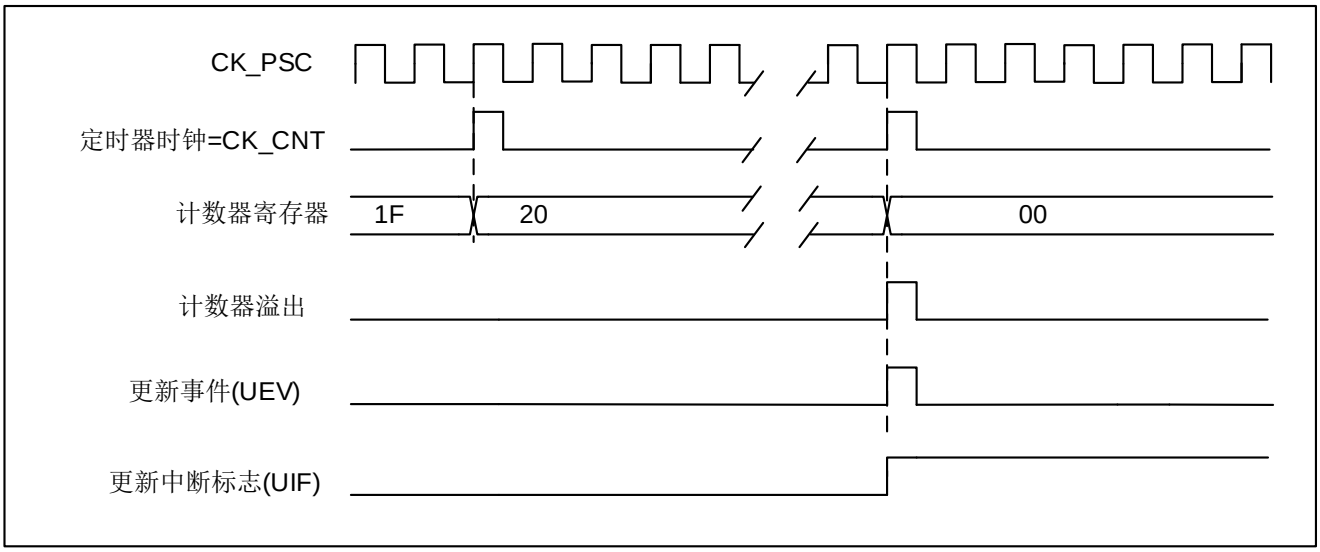


图 14-7 计数器时序图，内部时钟分频因子为 N

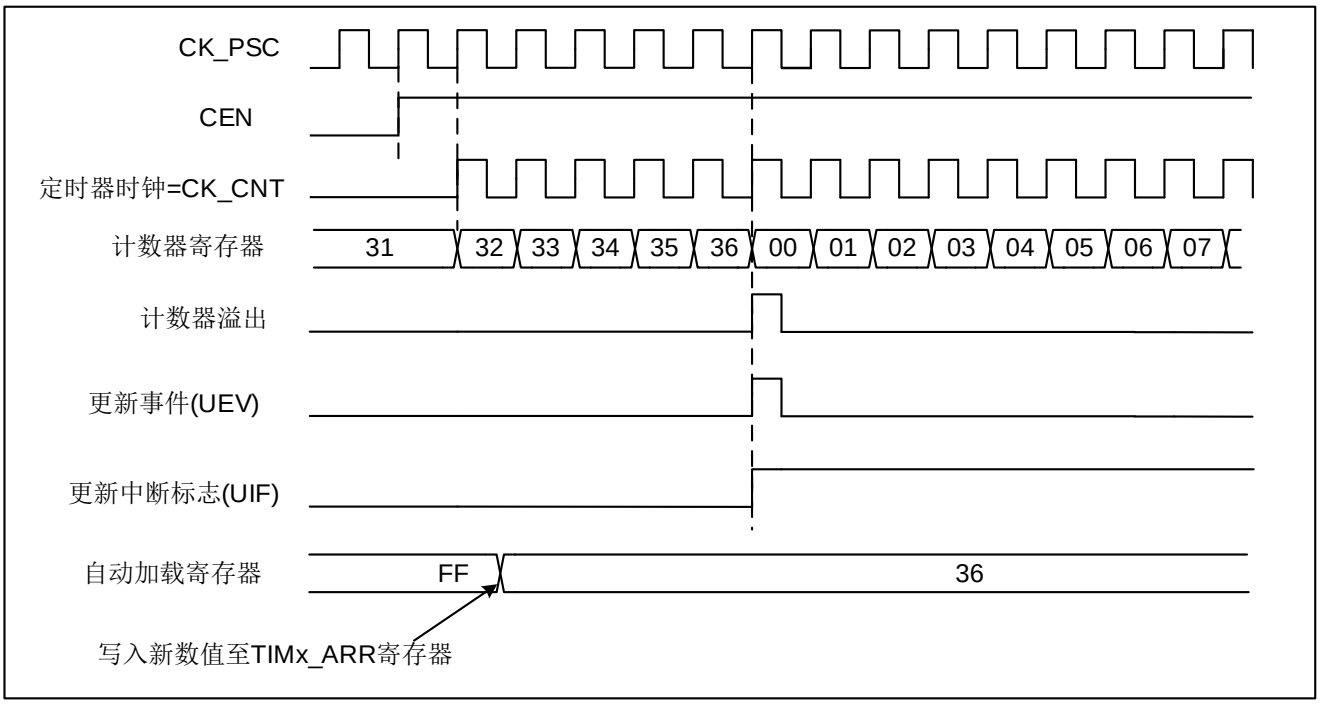


图 14-8 计数器时序图，当 ARPE=0 时的更新事件(TIMx_ARR 没有预装入)

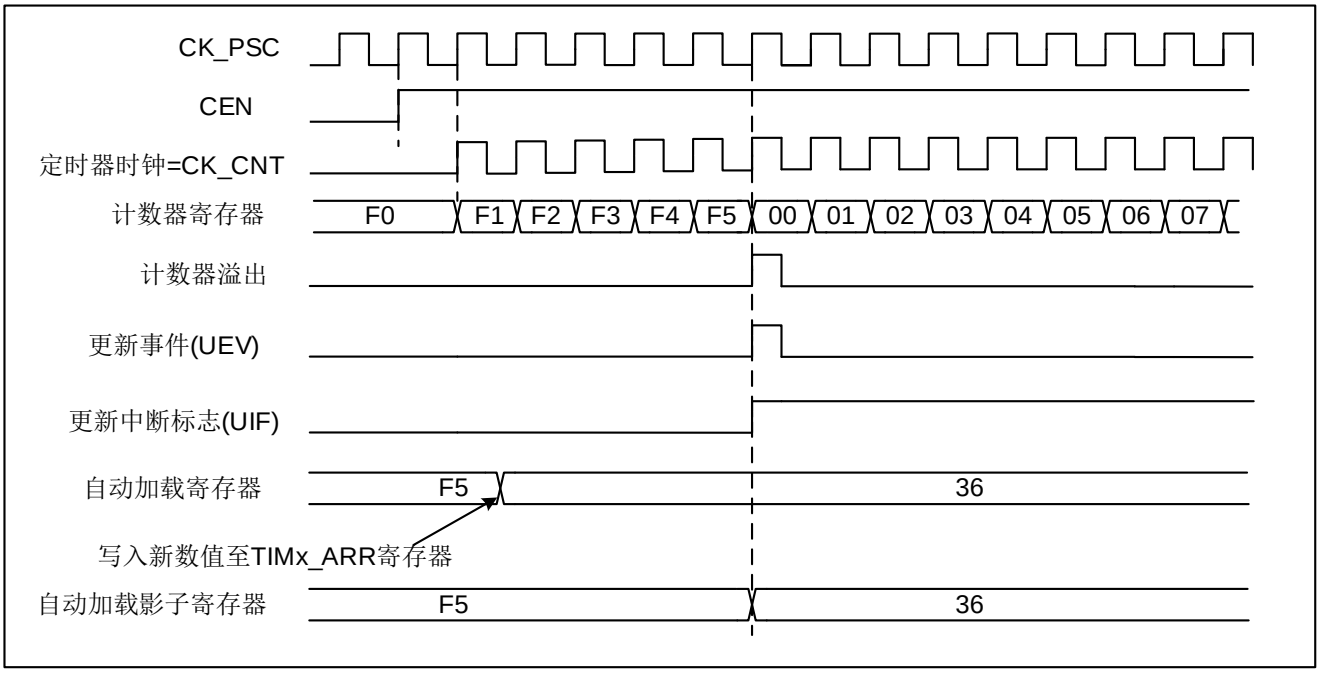


图 14-9 计数器时序图，当 ARPE=1 时的更新事件(预装入了 TIMx_ARR)

14.3.3.2 向下计数模式

在向下模式中，计数器从自动装入的值(TIMx_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(TIMx_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 TIMx_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 TIMx_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 TIMx_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)，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(TIMx_SR 寄存器中的 UIF 位)也被设置。

- 重复计数器被重置为 TIMx_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(TIMx_PSC 寄存器的值)
- 当前的自动加载寄存器被更新为预装载值(TIMx_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 TIMx_ARR=0x36 时，计数器在不同时钟频率下的操作例子。

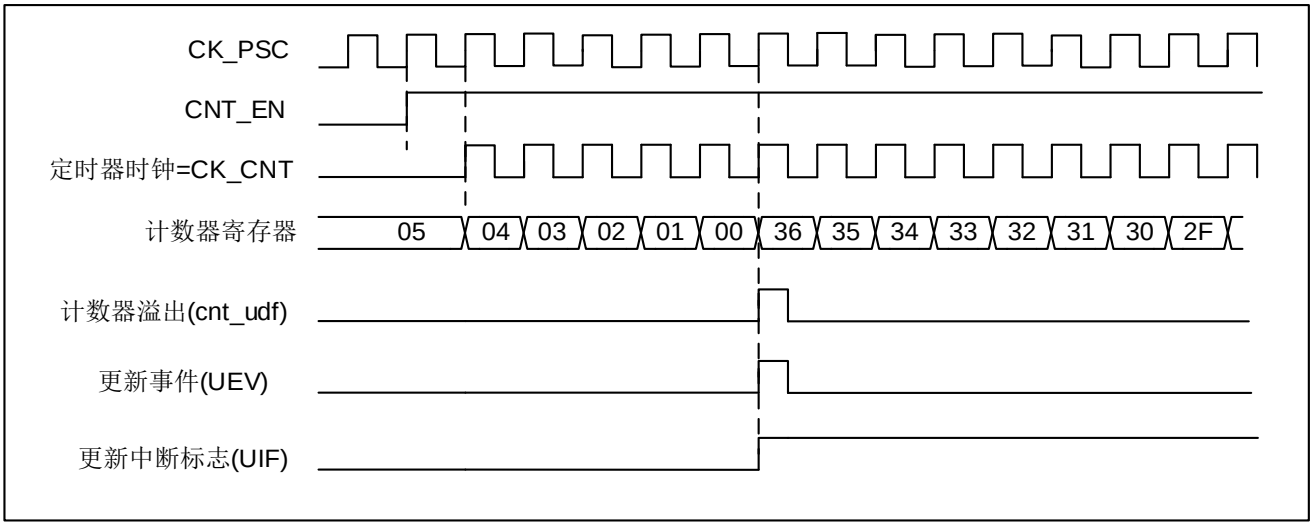


图 14-10 计数器时序图，内部时钟分频因子为 1

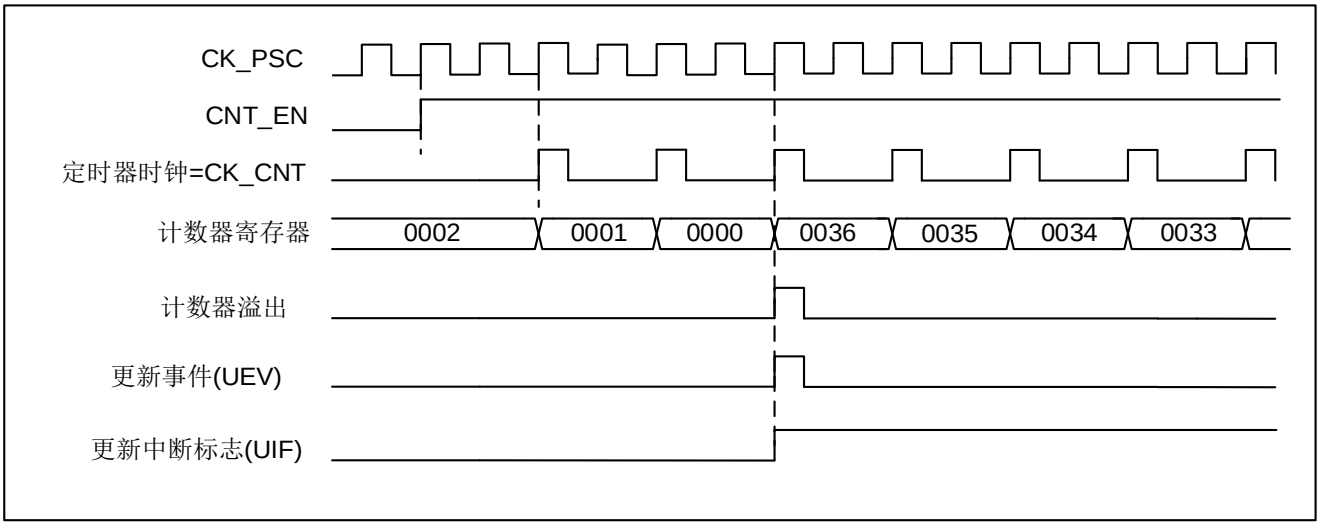


图 14-11 计数器时序图，内部时钟分频因子为 2

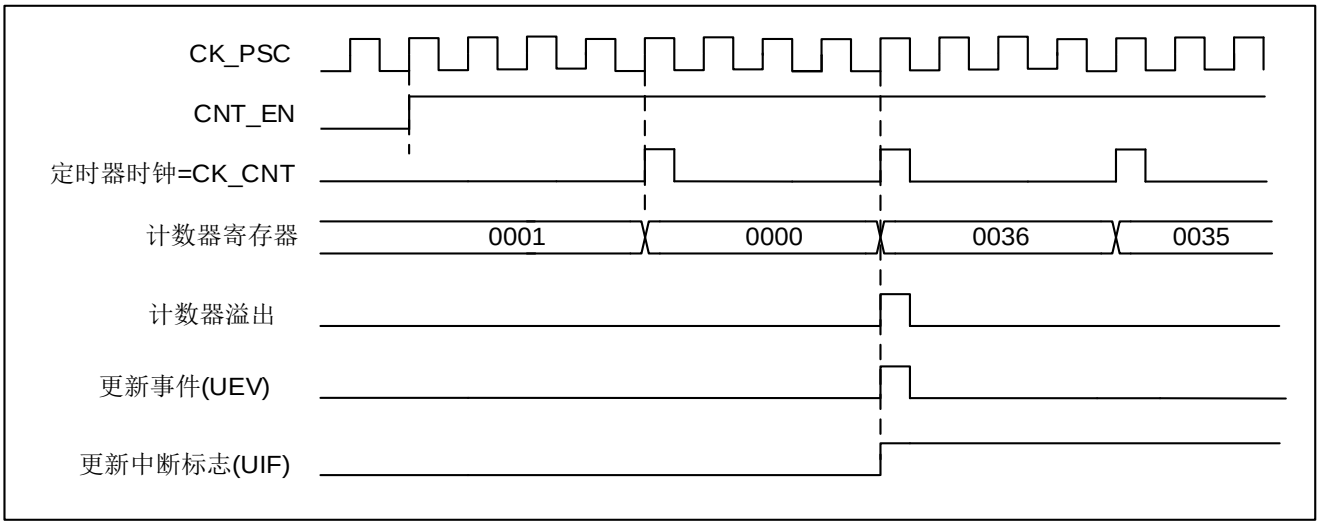


图 14-12 计数器时序图，内部时钟分频因子为 4

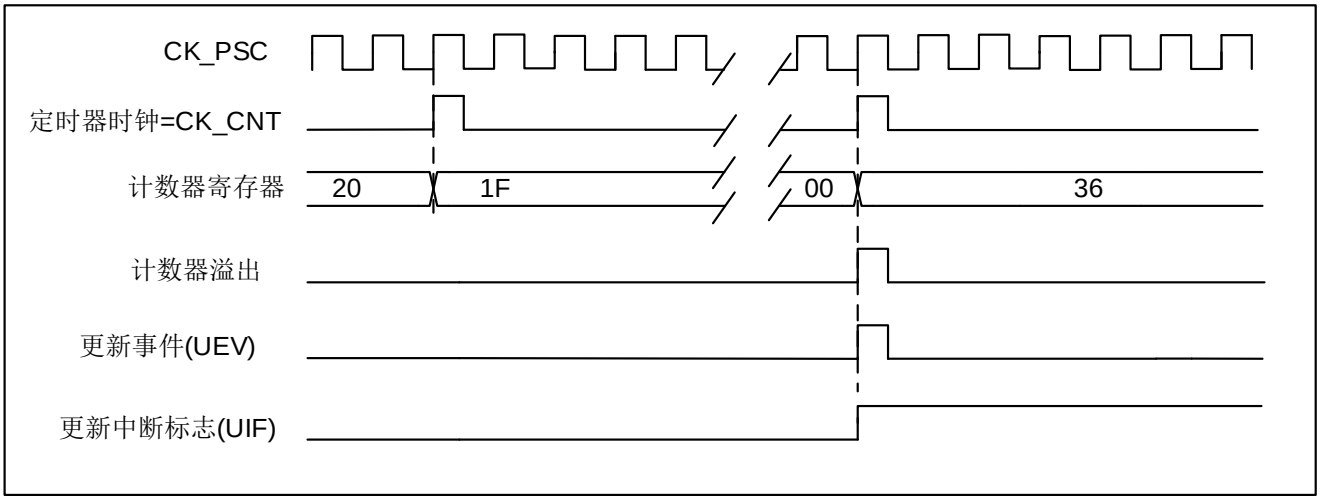


图 14-13 计数器时序图，内部时钟分频因子为 N

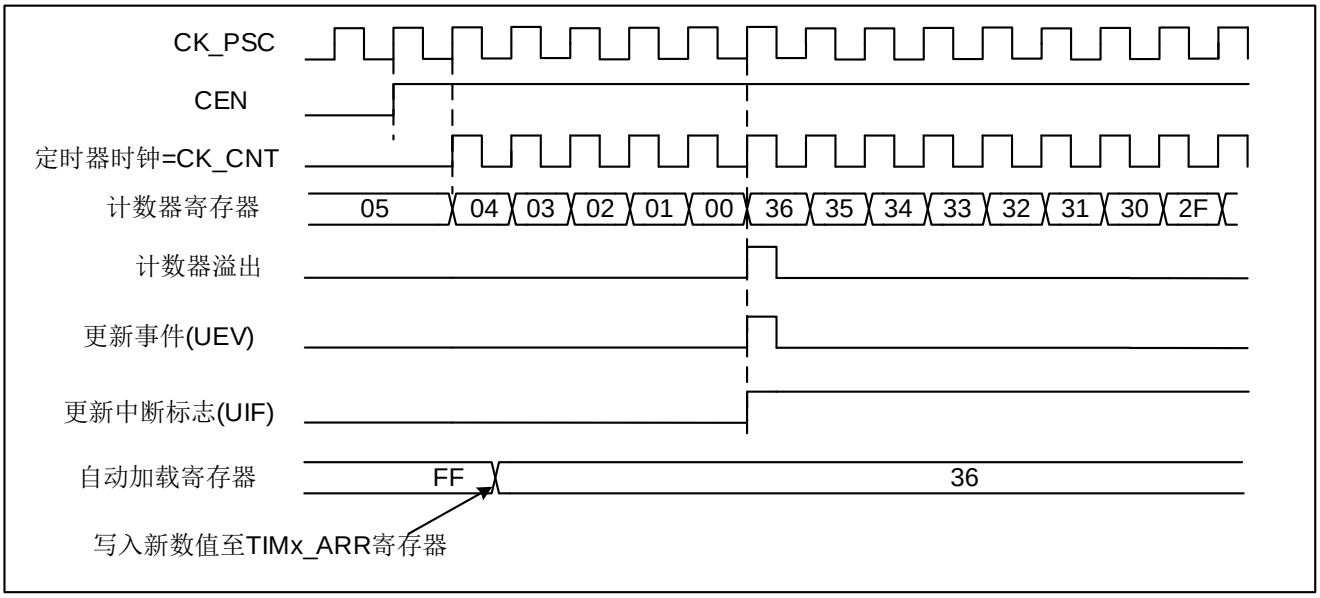


图 14-14 计数器时序图，当没有使用重复计数器时的更新事件

14.3.3.3 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(TIMx_ARR 寄存器)，产生一个计数器溢出事件，然后向下计数到 0 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 TIMx_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 TIMx_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 TIMx_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 TIMx_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)，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(TIMx_SR 寄存器中的 UIF 位)也被设置。

- 重复计数器被重置为 TIMx_RCR 寄存器中的内容
- 预分频器的缓存器被加载为预装载(TIMx_PSC 寄存器)的值
- 当前的自动加载寄存器被更新为预装载值(TIMx_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：

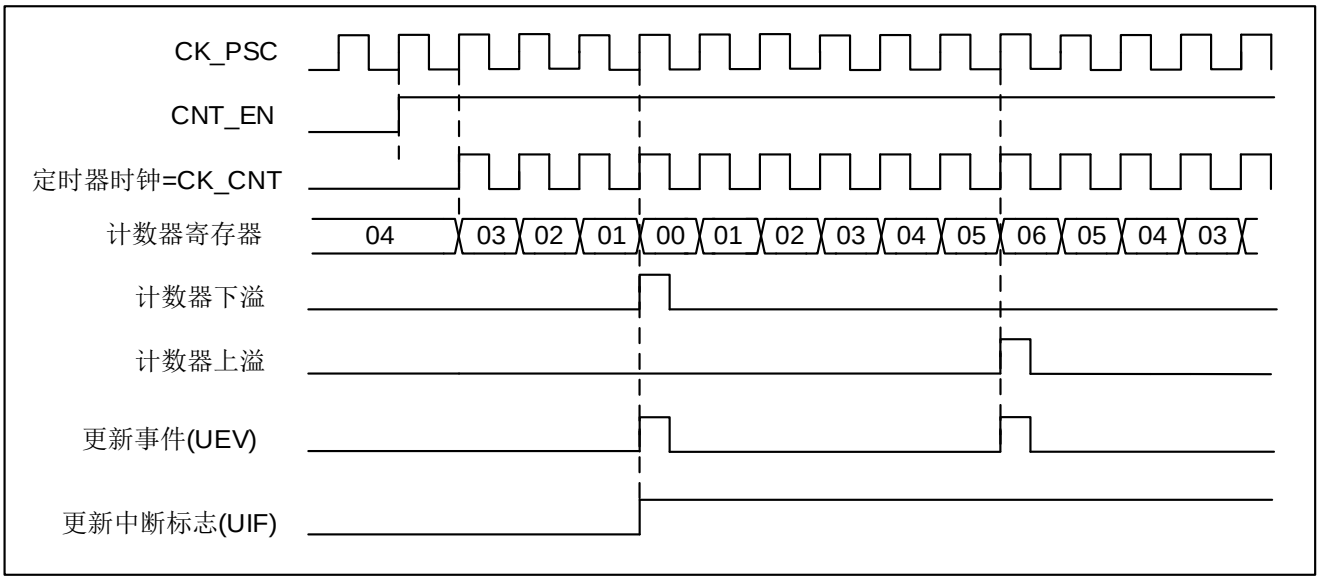


图 14-15 计数器时序图，内部时钟分频因子为 1，TIMx_ARR=0x6

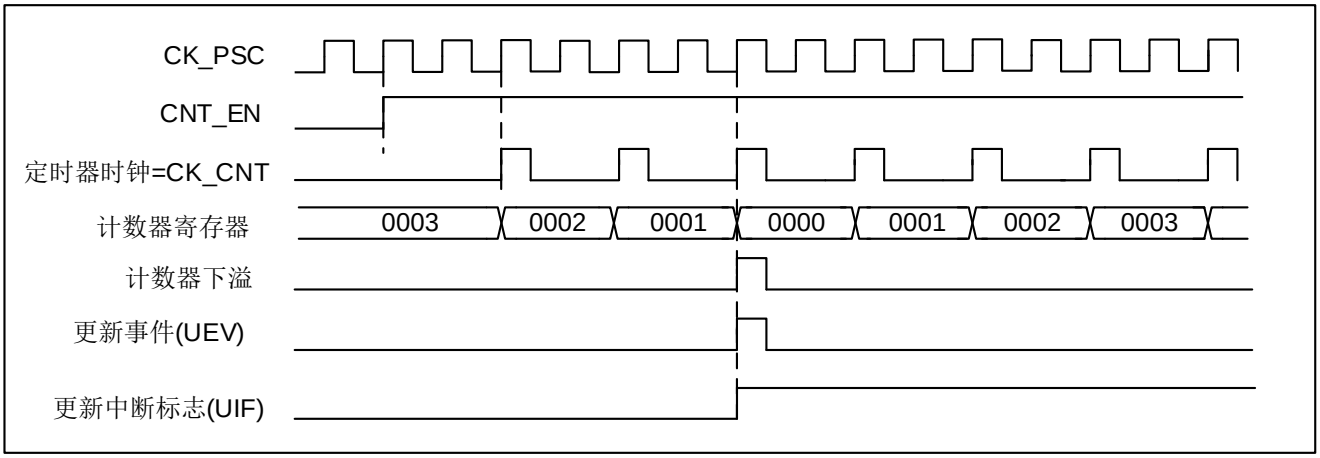


图 14-16 计数器时序图，内部时钟分频因子为 2

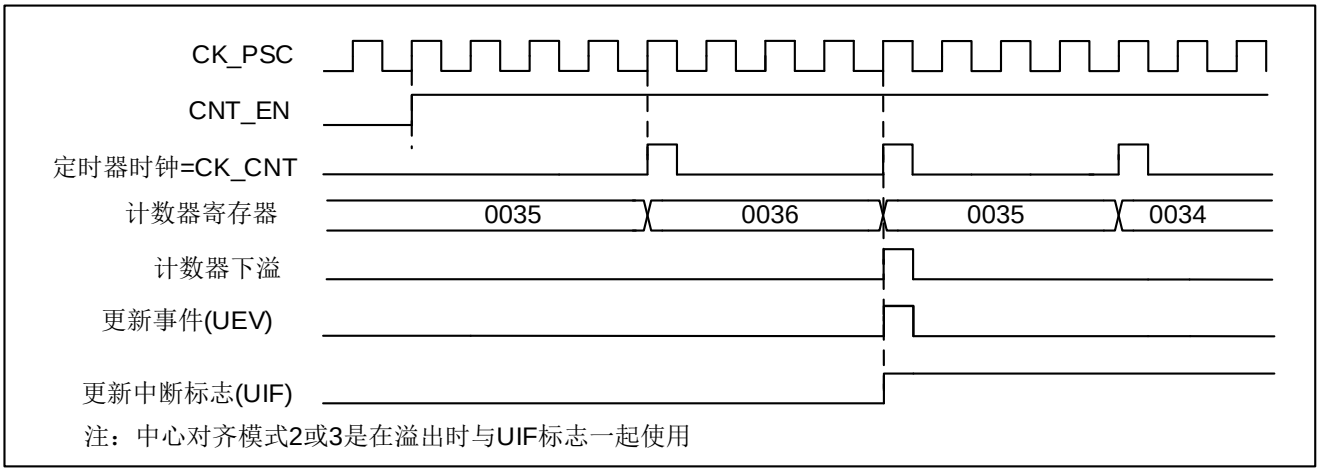


图 14-17 计数器时序图，内部时钟分频因子为 4，TIMx_ARR=0x36

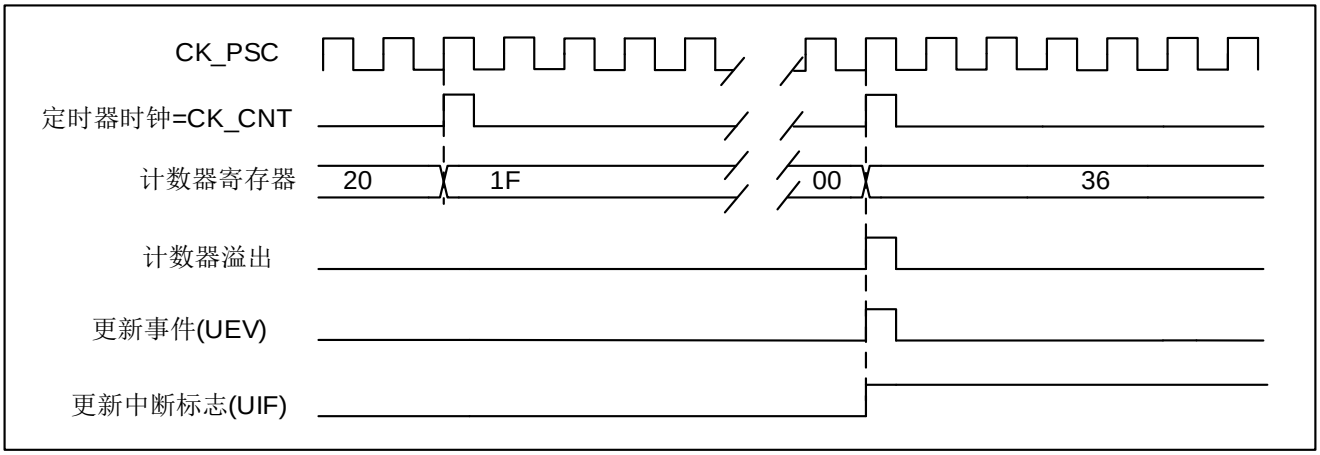


图 14-18 计数器时序图，内部时钟分频因子为 N

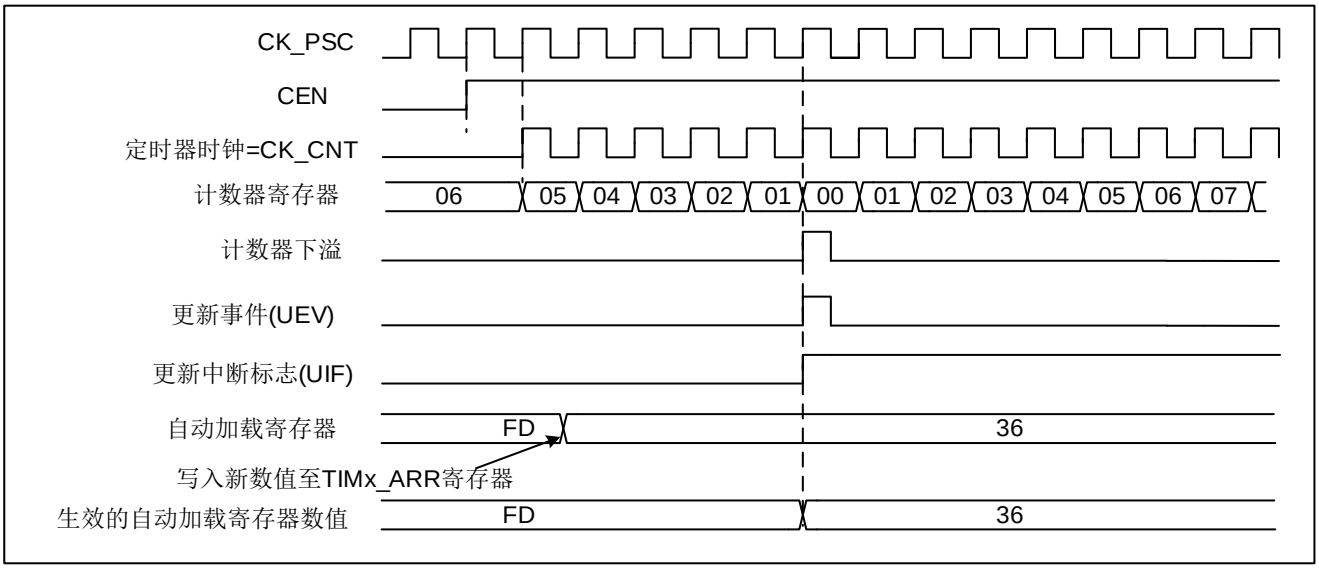


图 14-19 计数器时序图，ARPE=1 时的更新事件(计数器下溢)

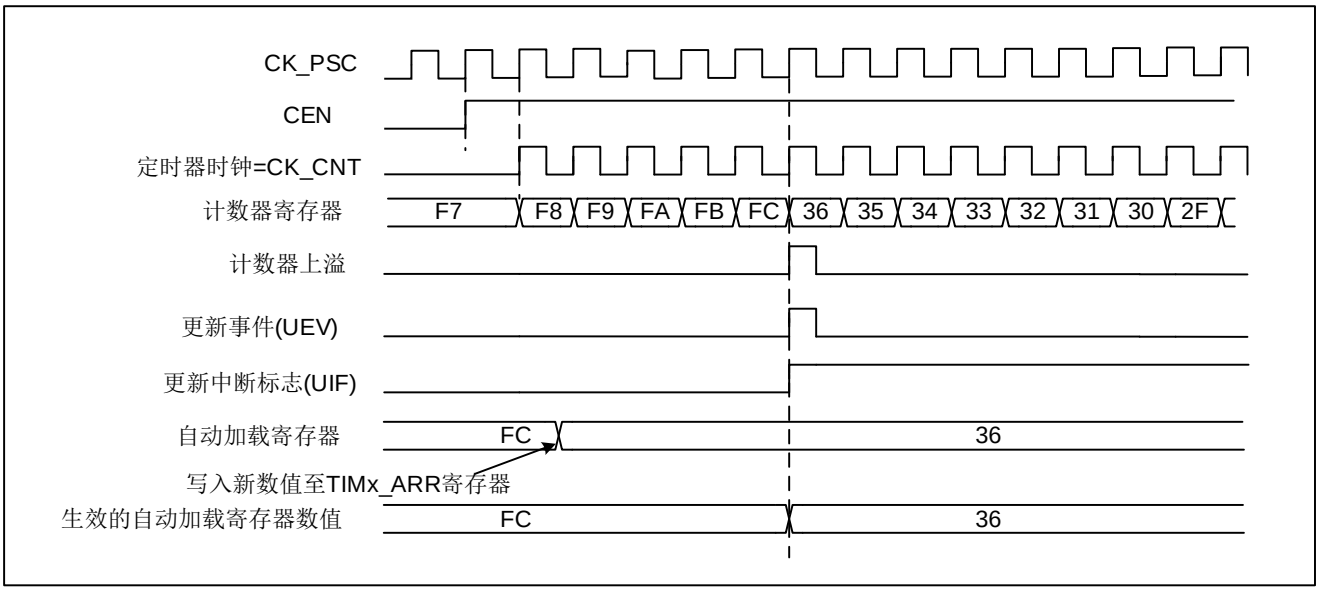


图 14-20 计数器时序图，ARPE=1 时的更新事件(计数器上溢)

14.3.4 重复计数器

“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(TIMx_ARR 自动重载入寄存器，TIMx_PSC 预装载寄存器，还有在比较模式下的捕获/比较寄存器 TIMx_CCRx)，N 是 TIMx_RCR 重复计数寄存器中的值。

重复计数器在下述任一条件成立时递减：

- 向上计数模式下每次计数器溢出时，
- 向下计数模式下每次计数器下溢时，
- 中央对齐模式下每次上溢和每次下溢时。虽然这样限制了 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 TIMx_RCR 寄存器的值定义(参看图 14-21)。当更新事件由软件产生(通过设置 TIMx_EGR 中的 UG 位)或者通过硬件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 TIMx_RCR 寄存器中的内容被重载入到重复计数器。

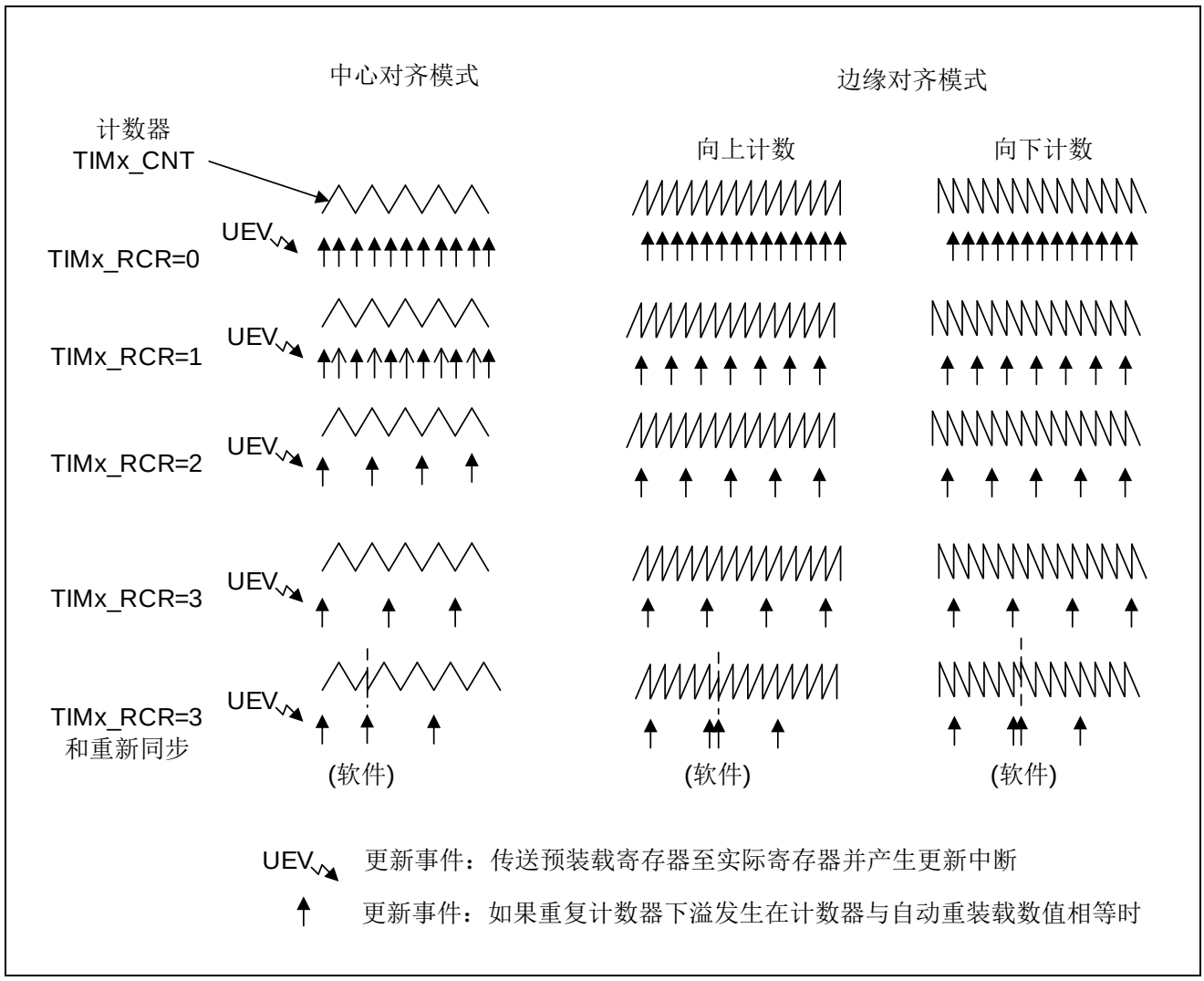


图 14-21 不同模式下更新速率的例子，及 TIMx_RCR 的寄存器设置

14.3.5 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 TIM1 而作为另一个定时器 TIM2 的预分频器。详情见 14.3.20.1。

14.3.5.1 内部时钟源(CK_INT)

如果禁止了从模式控制器(SMS=000)，则 CEN、DIR(TIMx_CR1 寄存器)和 UG 位(TIMx_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。

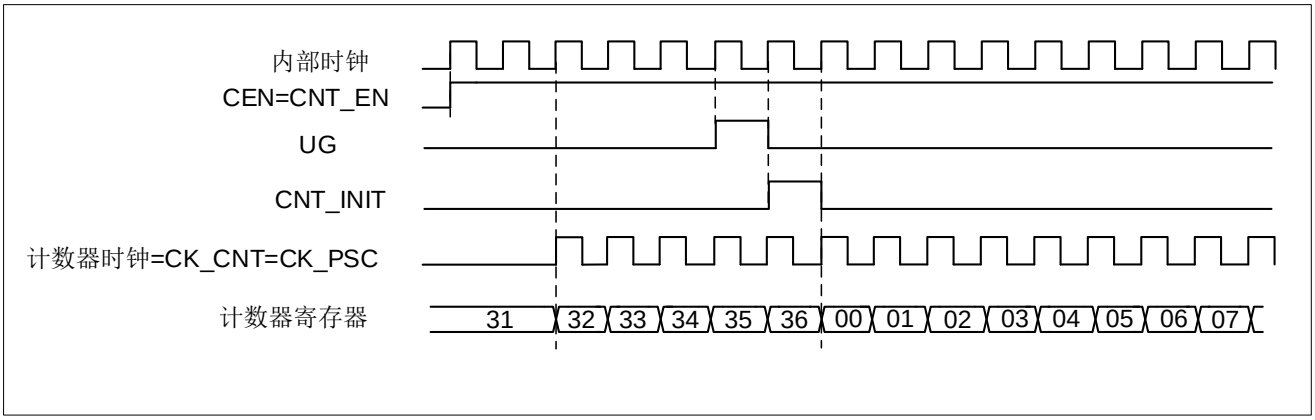


图 14-22 一般模式下的控制电路，内部时钟分频因子为 1

14.3.5.2 外部时钟源模式 1

当 TIMx_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。

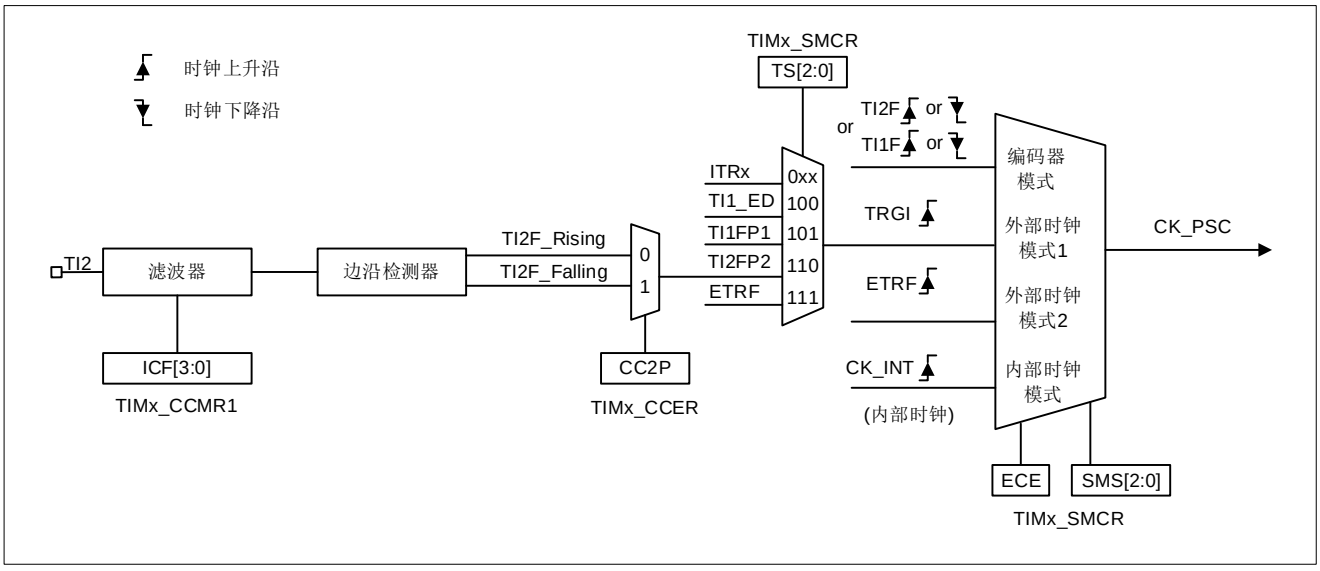


图 14-23 TI2 外部时钟连接例子

例如，要配置向上计数器在 TI2 输入端的上升沿计数，使用下列步骤：

配置 TIMx_CCMR1 寄存器 CC2S=01，配置通道 2 检测 TI2 输入的上升沿

配置 TIMx_CCMR1 寄存器的 IC2F，选择输入滤波器带宽(如果不需要滤波器，保持 IC2F=0000)

配置 TIMx_CCER 寄存器的 CC2P=0，选定上升沿极性

配置 TIMx_SMCR 寄存器的 SMS=111，选择定时器外部时钟模式 1

配置 TIMx_SMCR 寄存器中的 TS=110，选定 TI2 作为触发输入源

设置 TIMx_CR1 寄存器的 CEN=1，启动计数器

注：捕获预分频器不用作触发，所以不需要对它进行配置

当上升沿出现在 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。

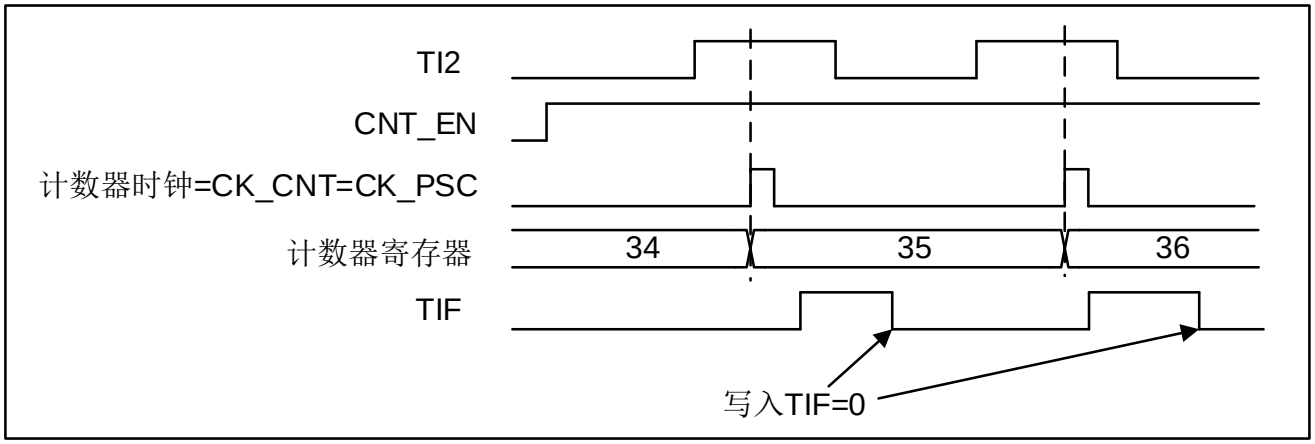


图 14-24 外部时钟模式 1 下的控制电路

14.3.5.3 外部时钟源模式 2

选定此模式的方法为：令 TIMx_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

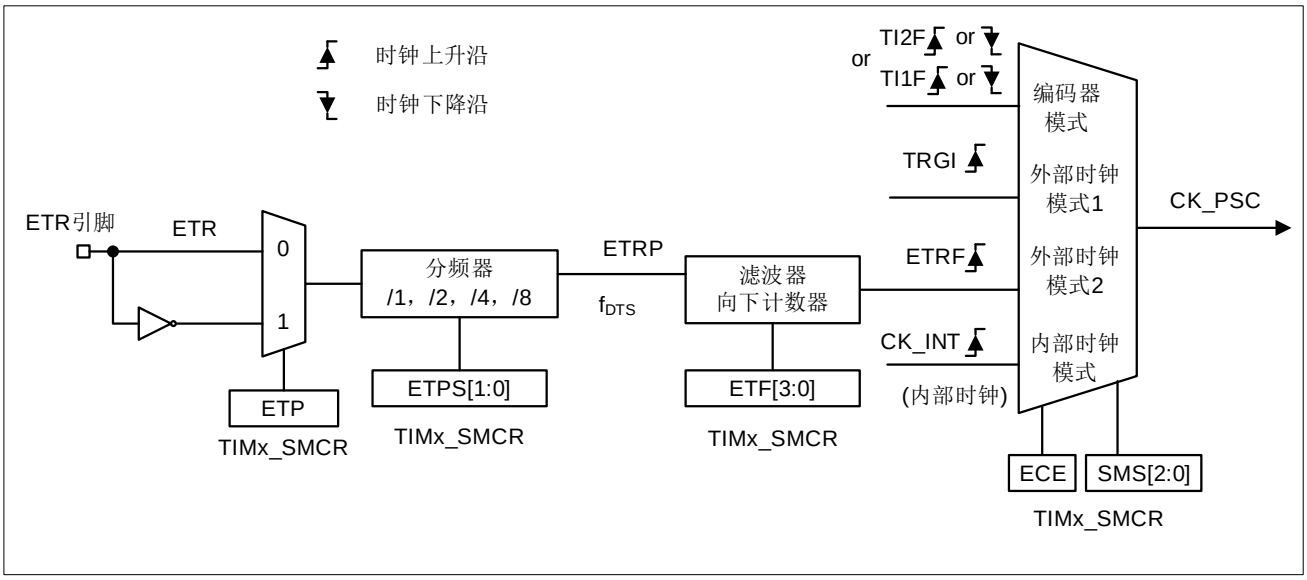


图 14-25 外部触发输入框图

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 TIMx_SMCR 寄存器中的 ETF=0000；
- 设置预分频器，置 TIMx_SMCR 寄存器中的 ETPS=01；
- 选择 ETR 的上升沿检测，置 TIMx_SMCR 寄存器中的 ETP=0；
- 开启外部时钟模式 2，写 TIMx_SMCR 寄存器中的 ECE=1；

- 启动计数器，写 TIMx_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。

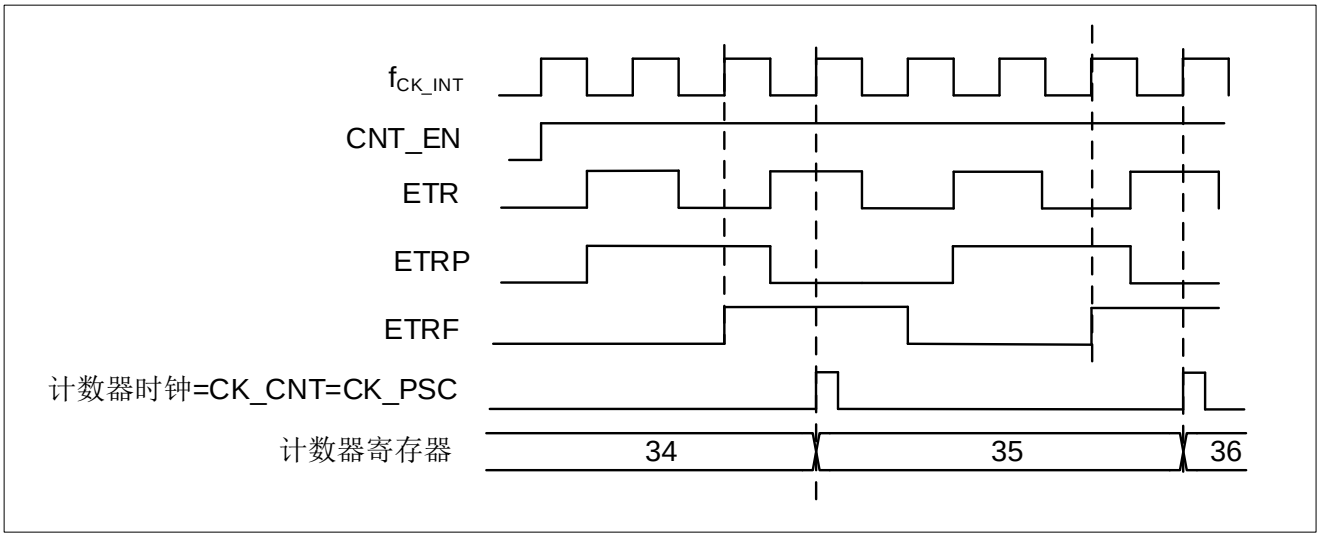


图 14-26 外部时钟模式 2 下的控制电路

14.3.6 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 14-27 至图 14-30 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)。

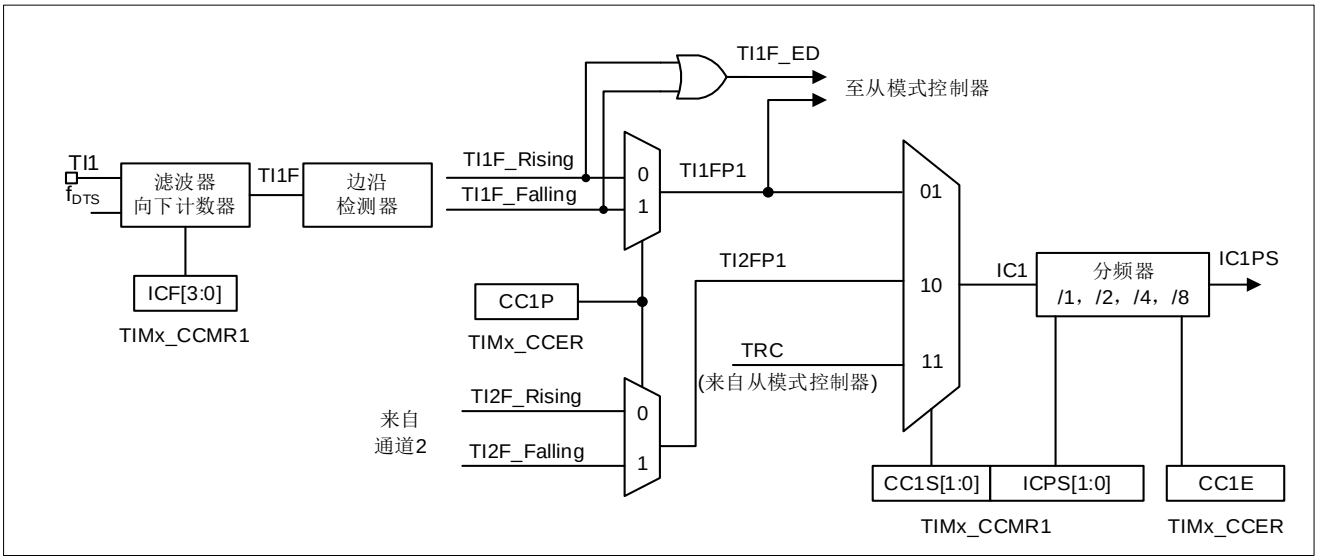


图 14-27 捕获/比较通道(如：通道 1 输入部分)

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

性。

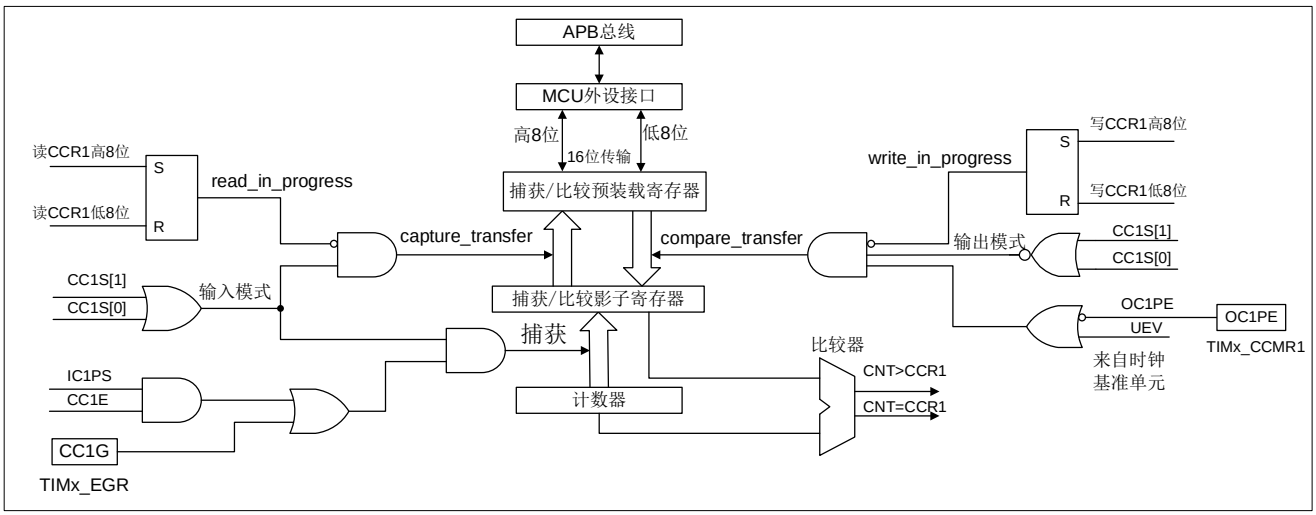


图 14-28 捕获/比较通道 1 的主电路

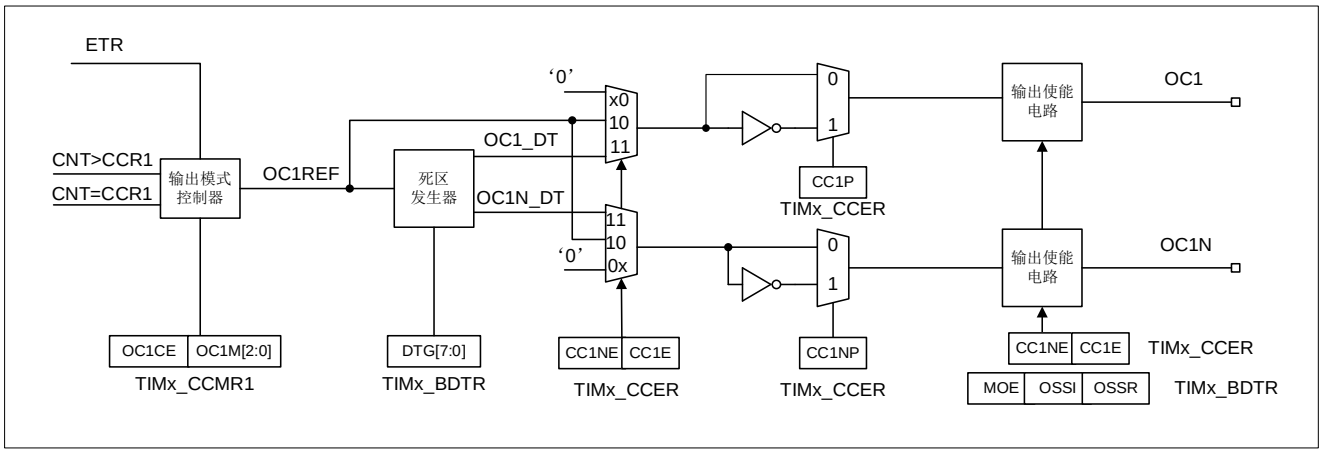


图 14-29 捕获/比较通道的输出部分(通道 1 至 3)

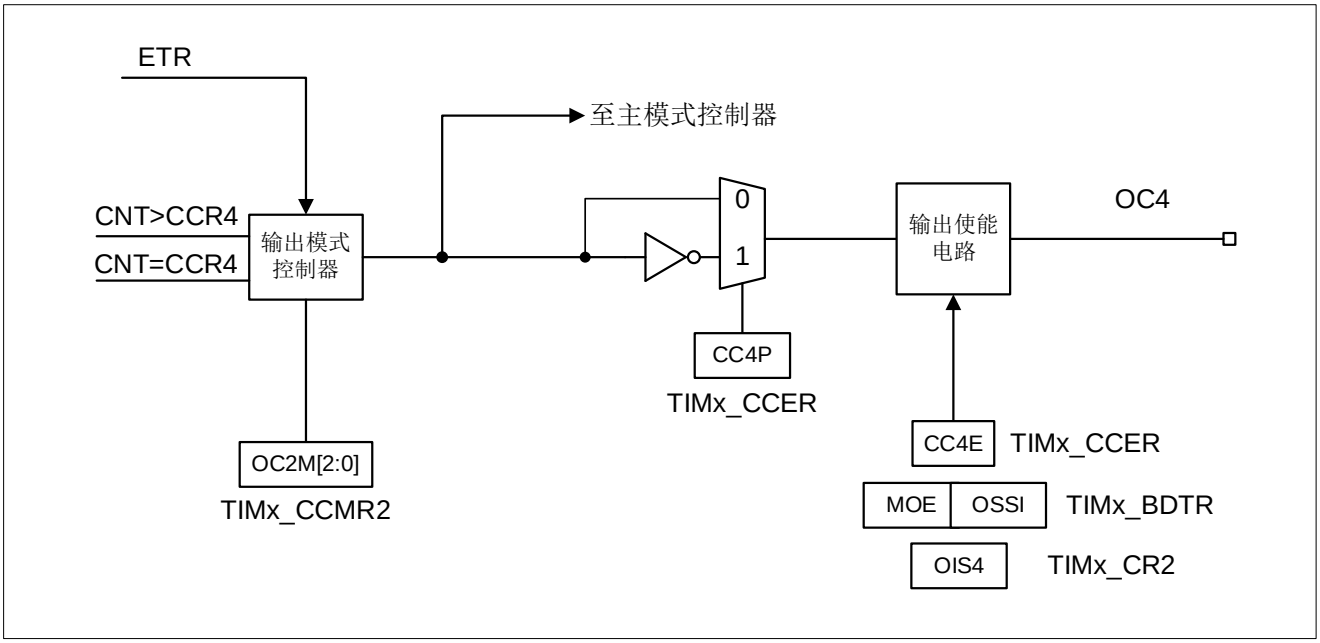


图 14-30 捕获/比较通道的输出部分(通道 4)

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。

14.3.7 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(TIMx_CCRx)中。当发生捕获事件时，相应的 CcxIF 标志(TIMx_SR 寄存器)被置 1，如果开放了中断，则将产生中断。如果发生捕获事件时 CcxIF 标志已经为高，那么重复捕获标志 CcxOF(TIMx_SR 寄存器)被置 1。写 CcxIF=0 可清除
CcxIF，或读取存储在 TIMx_CCRx 寄存器中的捕获数据也可清除 CcxIF。写 CcxOF=0 可清除 CcxOF。

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 TIMx_CCR1 寄存器中，步骤如下：

- 选择有效输入端：TIMx_CCR1 必须连接到 TI1 输入，所以写入 TIMx_CCMR1 寄存器中的 CC1S=01，只要 CC1S 不为'00'，通道被配置为输入，并且 TIMx_CCR1 寄存器变为只读。
- 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 TIMx_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 f_{DTS} 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 TIMx_CCMR1 寄存器中写入 IC1F=0011。
- 选择 TI1 通道的有效转换边沿，在 TIMx_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 TIMx_CCMR1 寄存器的 IC1PS=00)。
- 设置 TIMx_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 TIMx_DIER 寄存器中的 CC1IE 位允许相关中断请求。

当发生一个输入捕获时：

- 产生有效的电平转换时，计数器的值被传送到 TIMx_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。

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

14.3.8 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(TIMx_CCR1 寄存器)和占空比(TIMx_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 TIMx_CCR1 的有效输入：置 TIMx_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 TIMx_CCR1 中和清除计数器)：置 CC1P=0(上升沿有效)。
- 选择 TIMx_CCR2 的有效输入：置 TIMx_CCMR1 寄存器的 CC2S=10(选中 TI1)。
- 选择 TI1FP2 的有效极性(捕获数据到 TIMx_CCR2)：置 CC2P=1(下降沿有效)。
- 选择有效的触发输入信号：置 TIMx_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 TIMx_SMCR 中的 SMS=100。
- 使能捕获：置 TIMx_CCER 寄存器中 CC1E=1 且 CC2E=1。

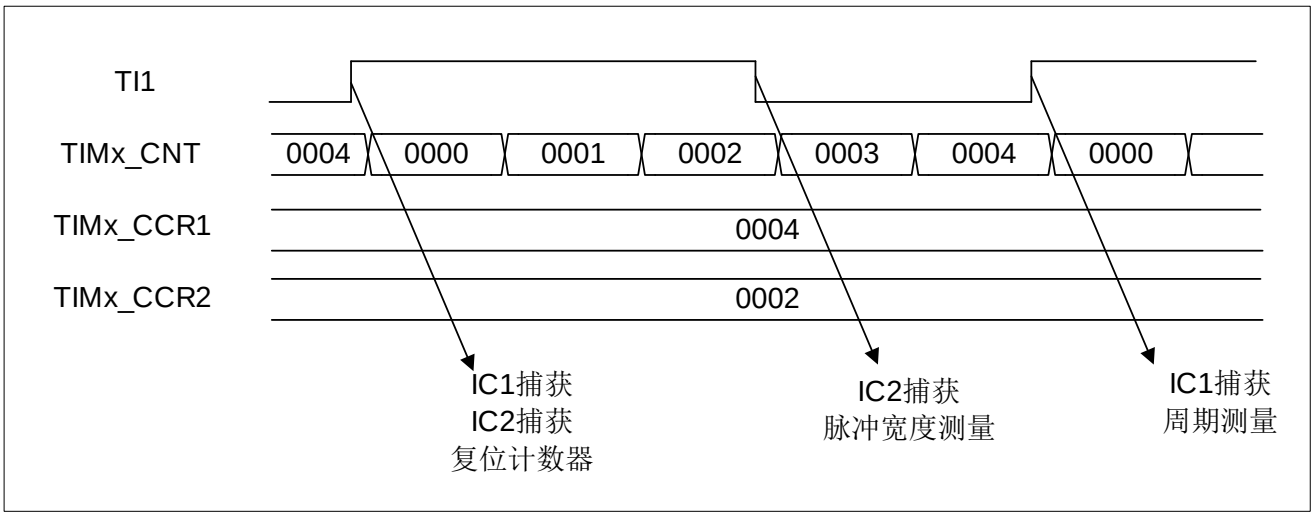


图 14-31 PWM 输入模式时序

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 TIMx_CH1/TIMx_CH2 信号。

14.3.9 强置输出模式

在输出模式(TIMx_CCMRx 寄存器中 CCxS=00)下，输出比较信号(OCxREF 和相应的 Ocx/OcxN)能够直接由软件强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 TIMx_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 TIMx_CCMRx 寄存器中的 OcxM=100，可强置 OCxREF 信号为低。

该模式下，在 TIMx_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。这将会在下面的输出比较模式一节中介绍。

14.3.10 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。当计数器与捕获/比较寄存器的内容相同时，输出比较功能做如下操作：

- 将输出比较模式(TIMx_CCMRx 寄存器中的 OcxM 位)和输出极性(TIMx_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(TIMx_SR 寄存器中的 CcxIF 位)。
- 若设置了相应的中断屏蔽(TIMx_DIER 寄存器中的 CcxIE 位)，则产生一个中断。

TIMx_CCMRx 中的 OcxPE 位选择 TIMx_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 TIMx_ARR 和 TIMx_CCRx 寄存器中。
3. 如果要产生一个中断请求，设置 CcxIE 位。
4. 选择输出模式，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 TIMx_CR1 寄存器的 CEN 位启动计数器

TIMx_CCRx 寄存器能够在任何时候通过软件进行更新以控制输出波形，条件是未使用预装载寄存器(OcxPE='0'，否则 TIMx_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。

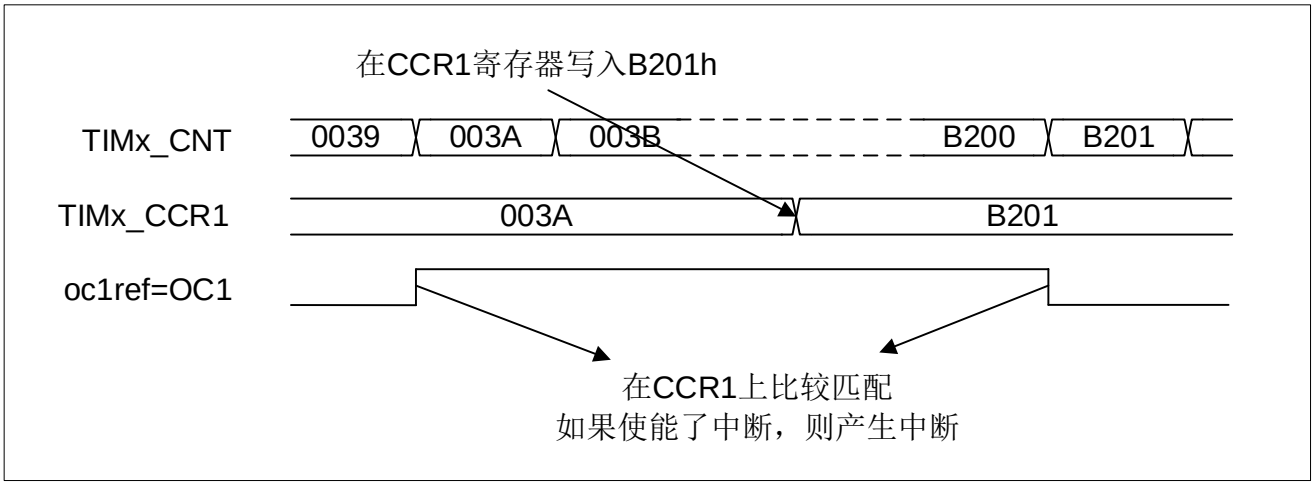


图 14-32 输出比较模式，翻转 OC1

14.3.11 PWM 模式

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

在 TIMx_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 TIMx_CCMRx 寄存器的 OcxPE 位使能相应的预装载寄存器，最后还要设置 TIMx_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 TIMx_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 TIMx_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(TIMx_CCER 和 TIMx_BDTR 寄存器中)CcxE、CcxNE、MOE、OSSI 和 OSSR 位的组合控制。详见 TIMx_CCER 寄存器的描述。

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

根据 TIMx_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。

然而为了与 OCREF_CLR 的功能(在下一个 PWM 周期之前，ETR 信号上的一个外部事件能够清除 OCxREF)一致，OCxREF 信号只能在下述条件下产生：

- 当比较的结果改变
- 当输出比较模式(TIMx_CCMRx 寄存器中的 OcxM 位)从“冻结”(无比较，OcxM=‘000’)切换到某个 PWM 模式(OcxM=‘110’或’111’)。

这样在运行中可以通过软件强置 PWM 输出。

根据 TIMx_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。

14.3.11.1 PWM 边沿对齐模式

14.3.11.1.1 向上计数配置

当 TIMx_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 TIMx_CNT<TIMx_CCRx 时，PWM 参考信号 OCxREF 为高，否则为低。如果 TIMx_CCRx 中的比较值大于自动重装载值 (TIMx_ARR)，则 OCxREF 保持为'1'。如果比较值为 0，则 OCxREF 保持为'0'。下图为 TIMx_ARR=8 时边沿对齐的 PWM 波形实例。

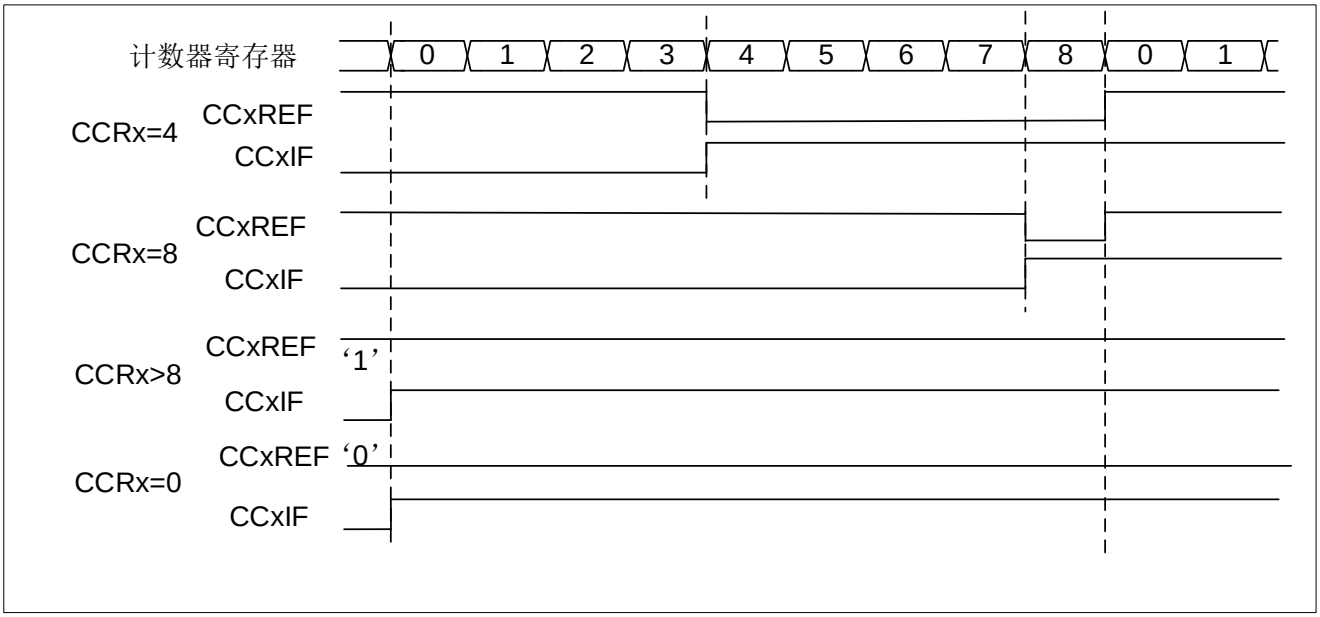


图 14-33 边沿对齐的 PWM 波形(ARR=8)

14.3.11.1.2 向下计数的配置

当 TIMx_CR1 寄存器的 DIR 位为高时执行向下计数。

在 PWM 模式 1，当 TIMx_CNT>TIMx_CCRx 时参考信号 OCxREF 为低，否则为高。如果 TIMx_CCRx 中的比较值大于 TIMx_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。

14.3.11.2 PWM 中央对齐模式

当 TIMx_CR1 寄存器中的 CMS 位不为'00'时为中央对齐模式(所有其他的配置对 OCxREF/Ocx 信号都有相同的作用)。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。TIMx_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

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

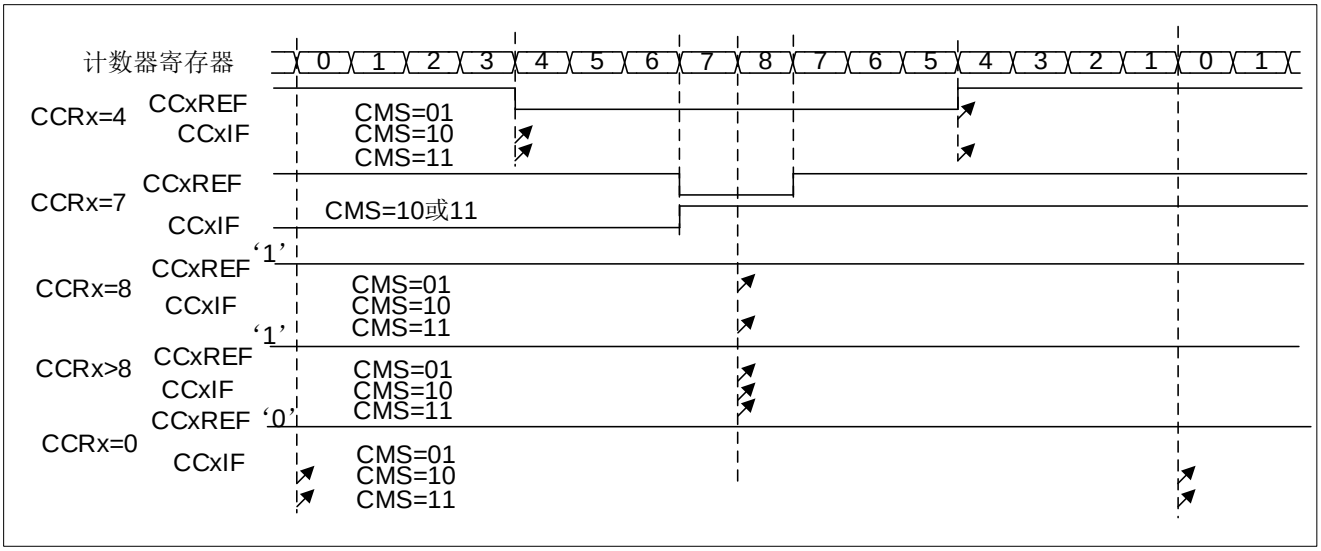


图 14-34 中央对齐的 PWM 波形(APR=8)

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 TIMx_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位。
- 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(TIMx_CNT>TIMx_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。
  - 如果将 0 或者 TIMx_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。
- 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 TIMx_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

14.3.12 互补输出和死区插入

高级控制定时器(TIMx)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 TIMx_CCER 寄存器中的 CCxP 和 CCxNP 位，可以为每一个输出独立地选择极性(主输出 Ocx 或互补输出 OcxN)。

互补信号 Ocx 和 OcxN 通过下列控制位的组合进行控制：TIMx_CCER 寄存器的 CcxE 和 CcxNE 位，TIMx_BDTR 和 TIMx_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位，详见

表 14-4 带刹车功能的互补输出通道 Ocx 和 OcxN 的控制位。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)

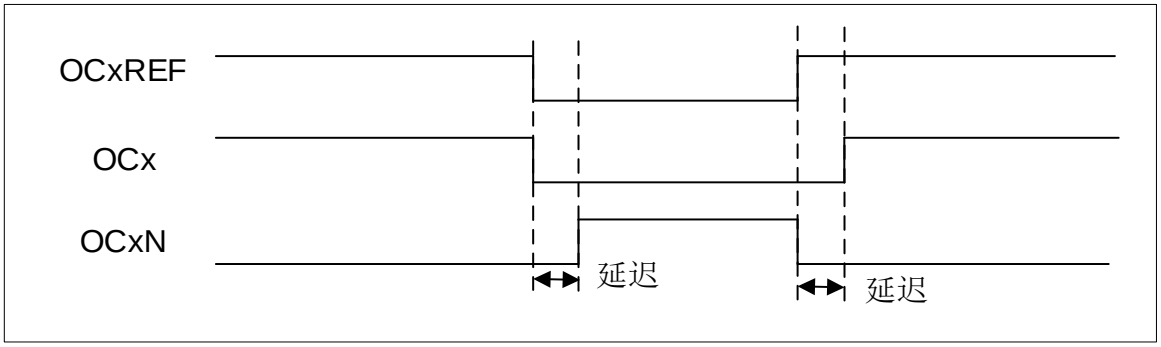


图 14-35 带死区插入的互补输出

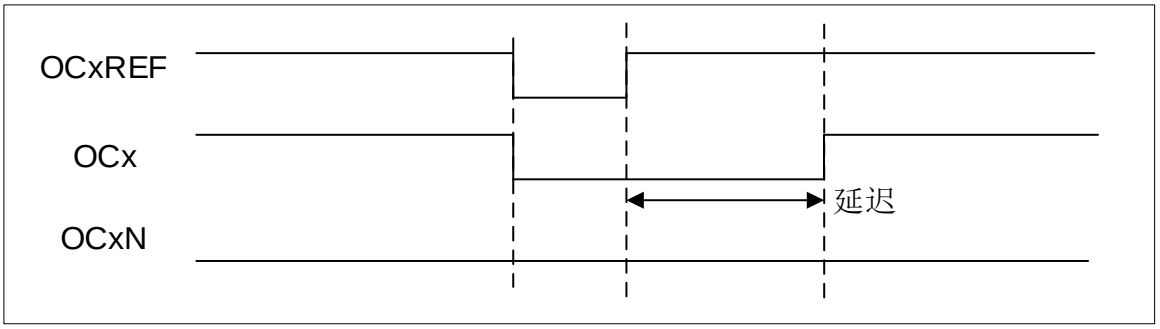


图 14-36 死区波形延迟大于负脉冲

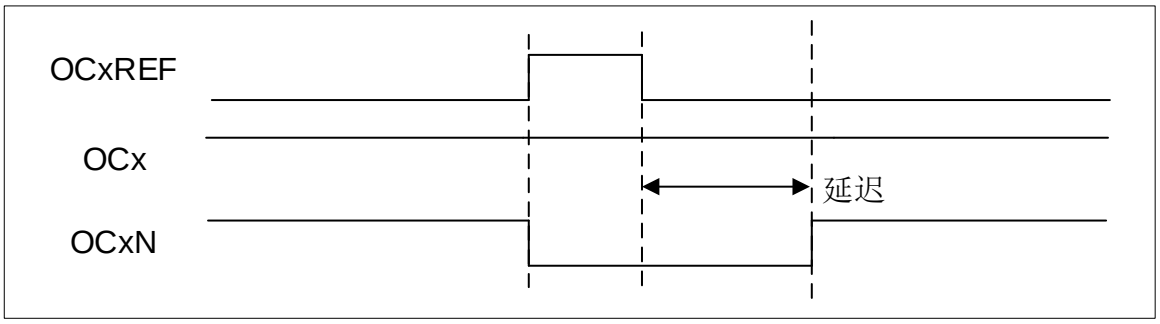


图 14-37 死区波形延迟大于正脉冲

每一个通道的死区延时都是相同的，是由 TIMx_BDTR 寄存器中的 DTG 位编程配置。详见 TIMx 刹车和死区寄存器(TIMx_BDTR)中的延时计算。

14.3.12.1 重定向 OCxREF 到 Ocx 或 OcxN

在输出模式下(强置、输出比较或 PWM)，通过配置 TIMx_CCER 寄存器的 CcxE 和 CcxNE 位，OCxREF 可以被重定向到 Ocx 或者 OcxN 的输出。

这个功能可以在互补输出处于无效电平时，在某个输出上送出一个特殊的波形(例如 PWM 或者静态有效电平)。另一个作用是，让两个输出同时处于无效电平，或处于有效电平和带死区的互补输出。

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。

14.3.13 使用刹车功能

当使用刹车功能时，依据相应的控制位(TIMx_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIMx_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。详见

表 14-4 带刹车功能的互补输出通道 Ocx 和 OcxN 的控制位。

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 TIMx_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 不可以同时被修改，应当先配置 BKP，再使能 BKE。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 TIMx_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 TIMx_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 TIMx_DIER 寄存器中的 BIE 位，当刹车状态标志(TIMx_SR 寄存器中的 BIF 位)为'1'时，则产生一个中断。
- 如果设置了 TIMx_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 TIMx_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 TIMx_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，参看 14.5.18 节 TIMx 刹车和死区寄存器(TIMx_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。

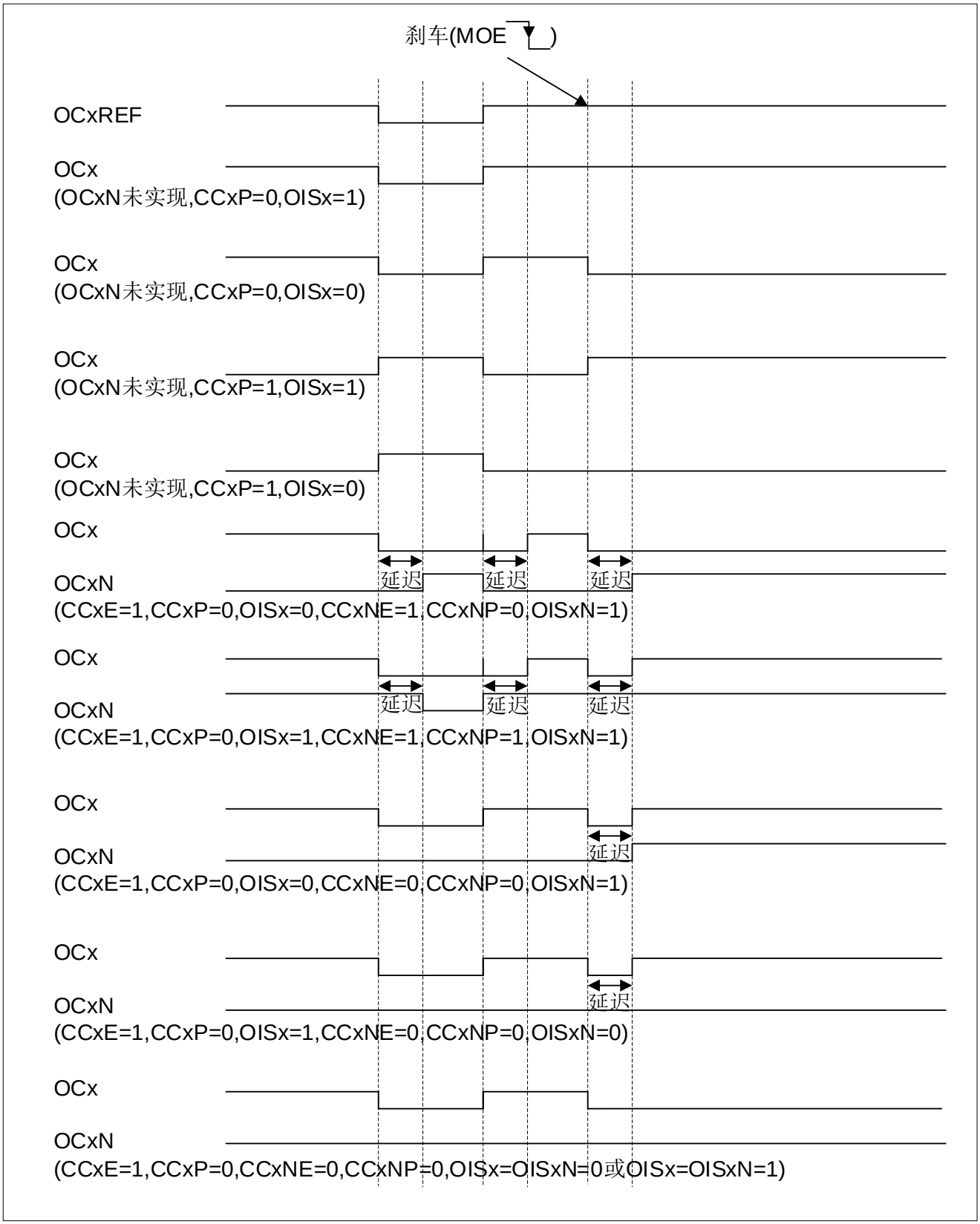


图 14-38 响应刹车的输出

14.3.14 在外部事件时清除 OCxREF 信号

对于一个给定的通道，设置 TIMx_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：TIMx_SMCR 寄存器中的 ETPS=00。
2. 必须禁止外部时钟模式 2：TIMx_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 TIMx 被置于 PWM 模式。

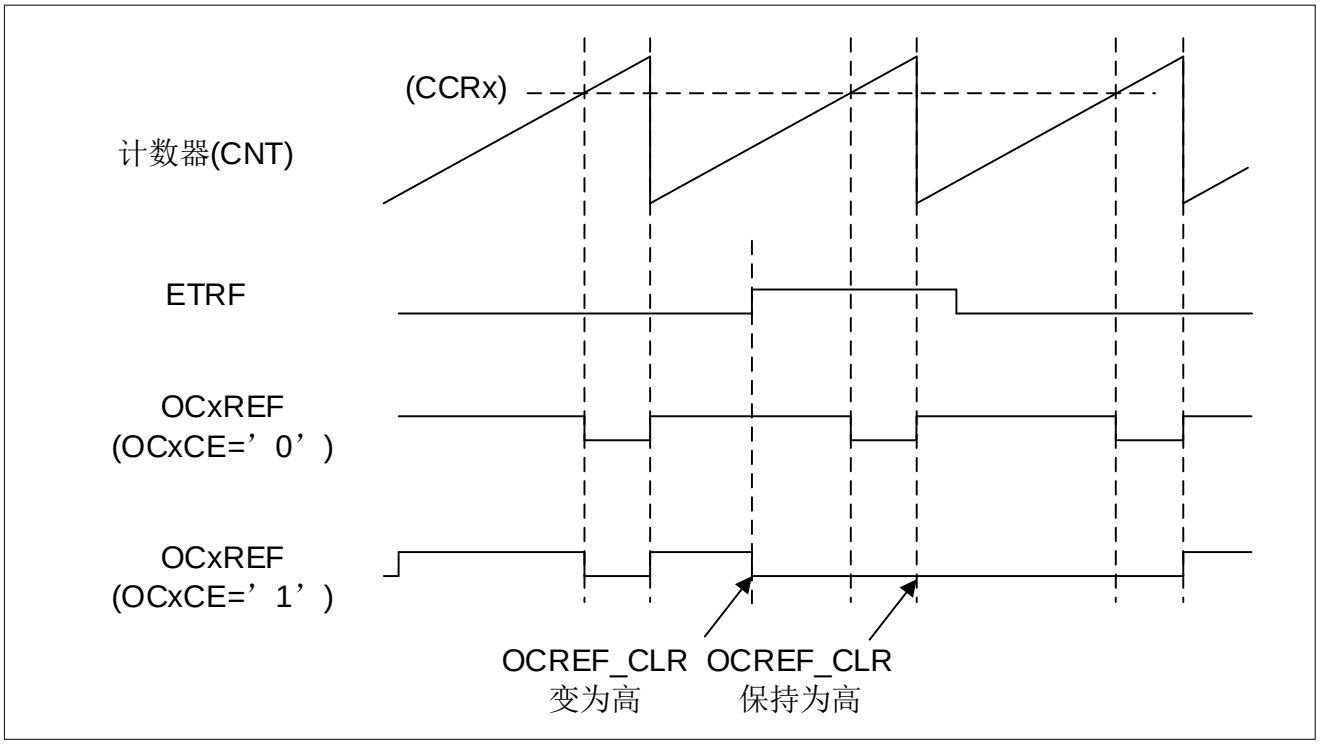


图 14-39 清除 TIMx 的 OCxREF

14.3.15 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 TIMx_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(TIMx_SR 寄存器中的 COMIF 位)，这时如果已设置了 TIMx_DIER 寄存器的 COMIE 位，则产生一个中断。

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

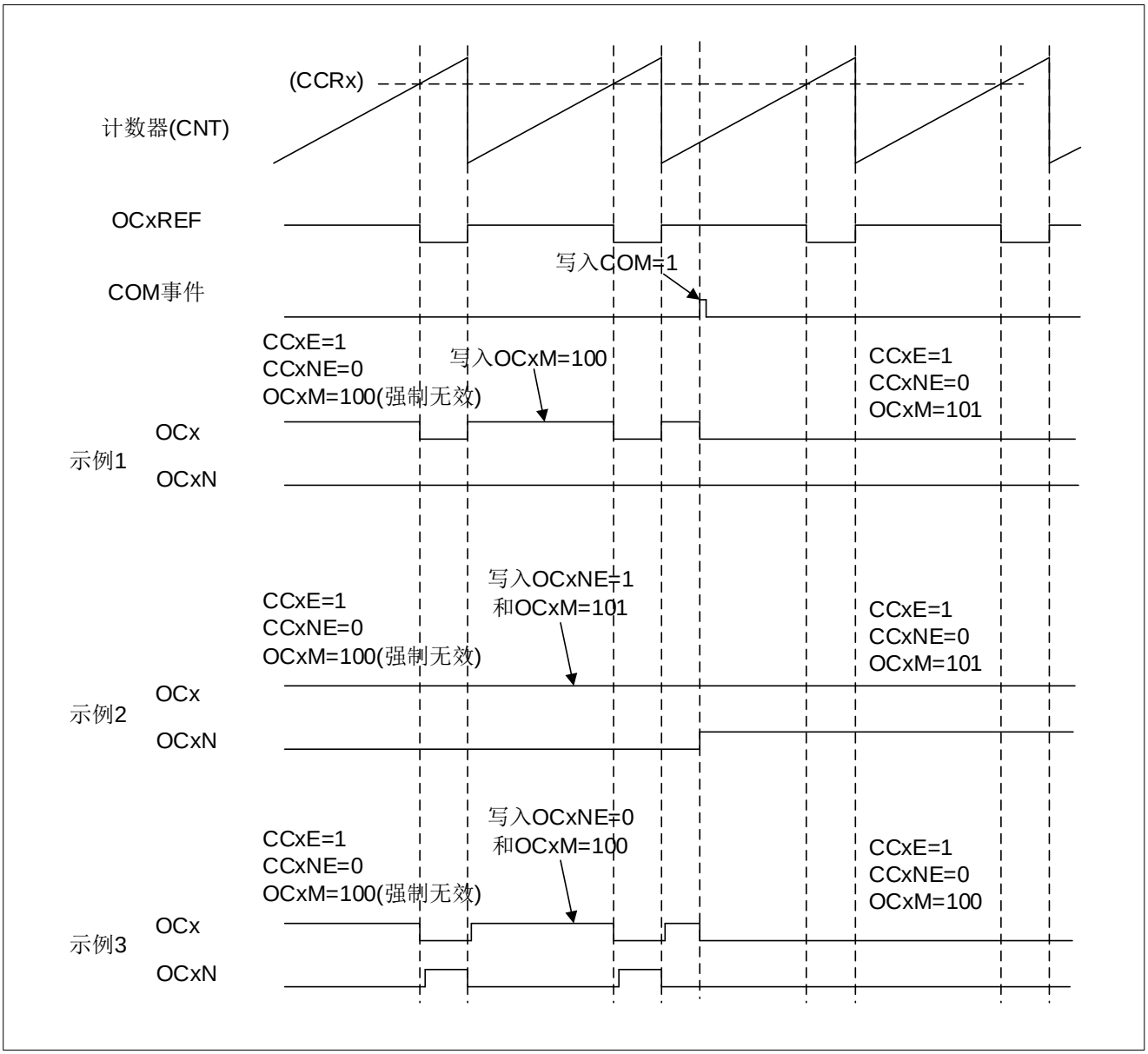


图 14-40 产生六步 PWM，使用 COM 的例子(OSSR=1)

14.3.16 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 TIMx_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。

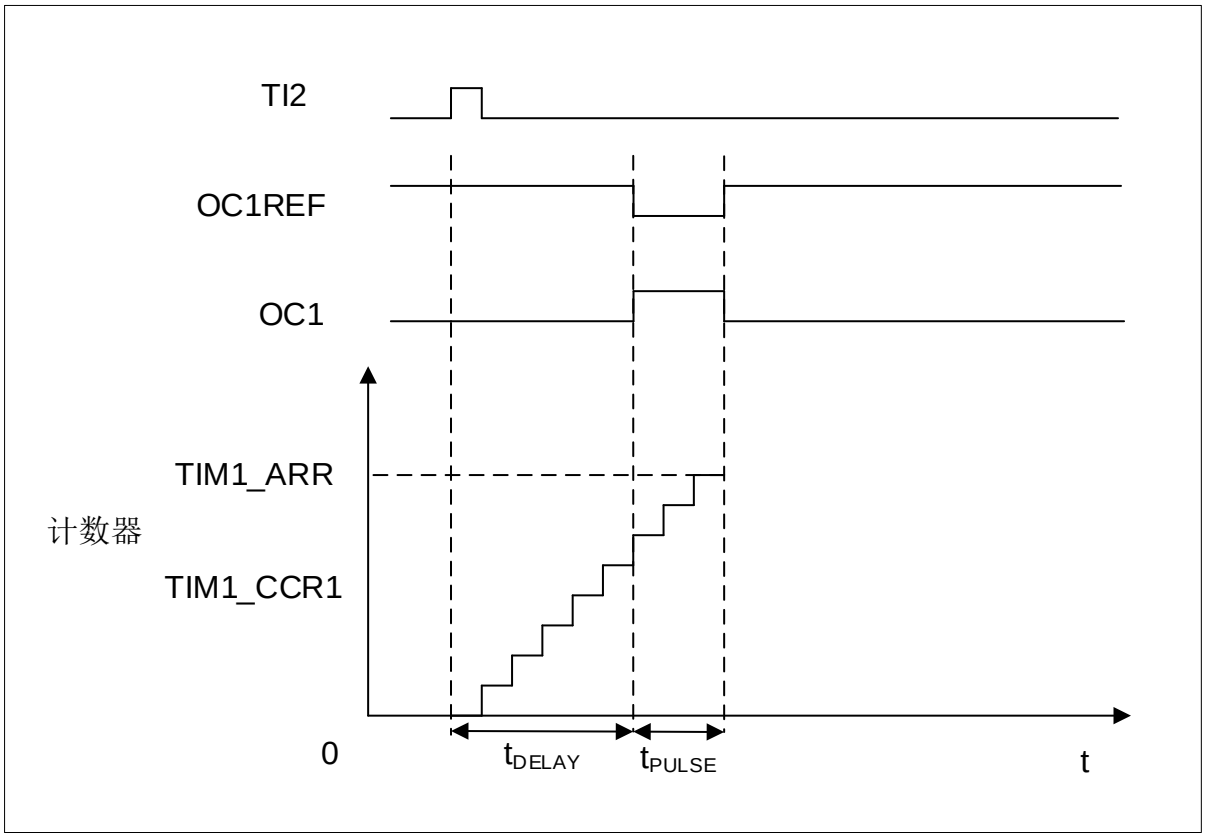


图 14-41 单脉冲模式的例子

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 TIMx_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。
- 置 TIMx_CCER 寄存器中的 CC2P=0，使 TI2FP2 能够检测上升沿。
- 置 TIMx_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。
- 置 TIMx_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。OPM 的波形由写入比较寄存器的数值决定(要考虑时钟频率和计数器预分频器)
- t_{DELAY} 由 TIMx_CCR1 寄存器中的值定义。
- t_{PULSE} 由自动装载值和比较值之间的差值定义(TIMx_ARR – TIMx_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 TIMx_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 TIMx_CCMR1 中的 OC1PE=1 和 TIMx_CR1 寄存器中的 ARPE；然后在 TIMx_CCR1 寄存器中填写比较值，在 TIMx_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，TIMx_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 TIMx_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 TIMx_CCMRx 寄存器中的 OcxFE 位；此时 OCxREF(和 Ocx)直接响应激励而不再依赖比较的结果，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。

14.3.17 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 TIMx_SMCR 寄存器中的 SMS=001；如果只在 TI1 边沿计数，则置 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。

通过设置 TIMx_CCER 寄存器中的 CC1P 和 CC2P 位，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(TIMx_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 TIMx_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 TIMx_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 TIMx_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 14-1 计数方向与编码器信号的关系

有效边沿	相对信号的电平(TIFP1 对应 TI2，TI2FP2 对应 TI1)	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
仅在 TI1 计数	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
仅在 TI2 计数	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
在 TI1 和 TI2 上计数	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大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’ (TIMx_CCMR1 寄存器，IC1FP1 映射到 TI1)
- CC2S=‘01’ (TIMx_CCMR2 寄存器，IC2FP2 映射到 TI2)
- CC1P=‘0’ (TIMx_CCER 寄存器，IC1FP1 不反相，IC1FP1=TI1)
- CC2P=‘0’ (TIMx_CCER 寄存器，IC2FP2 不反相，IC2FP2=TI2)
- SMS=‘011’ (TIMx_SMCR 寄存器，所有的输入均在上升沿和下降沿有效)
- CEN=‘1’ (TIMx_CR1 寄存器，计数器使能)

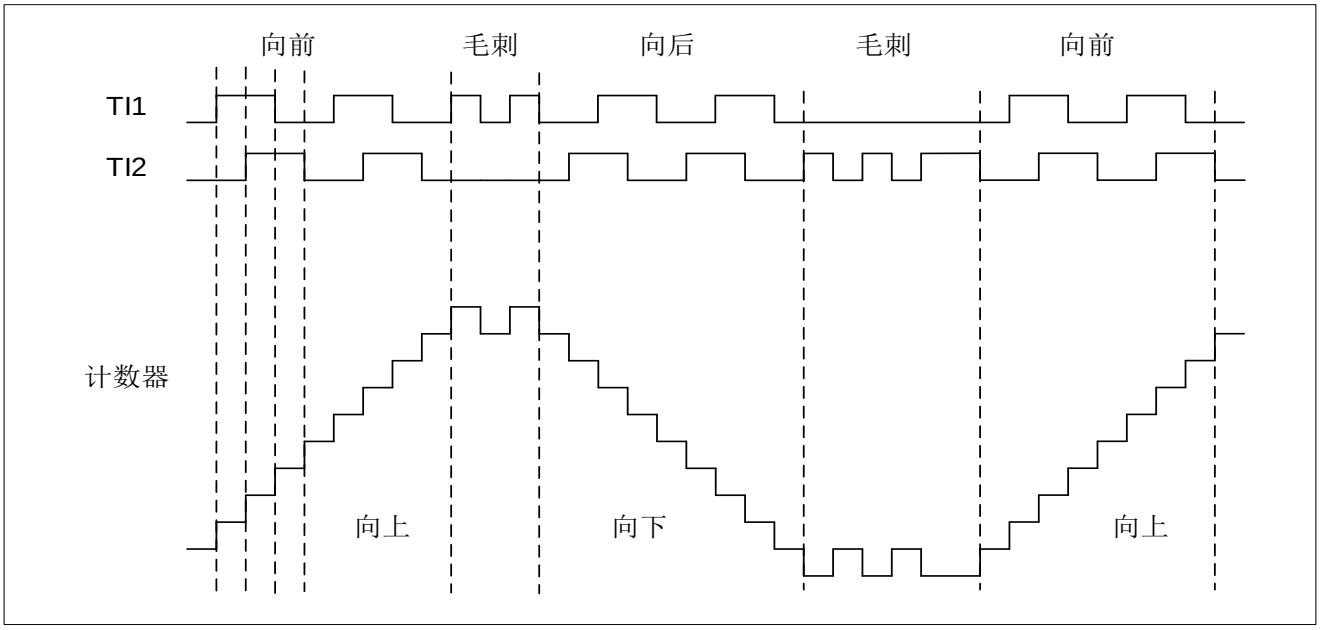


图 14-42 编码器模式下的计数器操作实例

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)

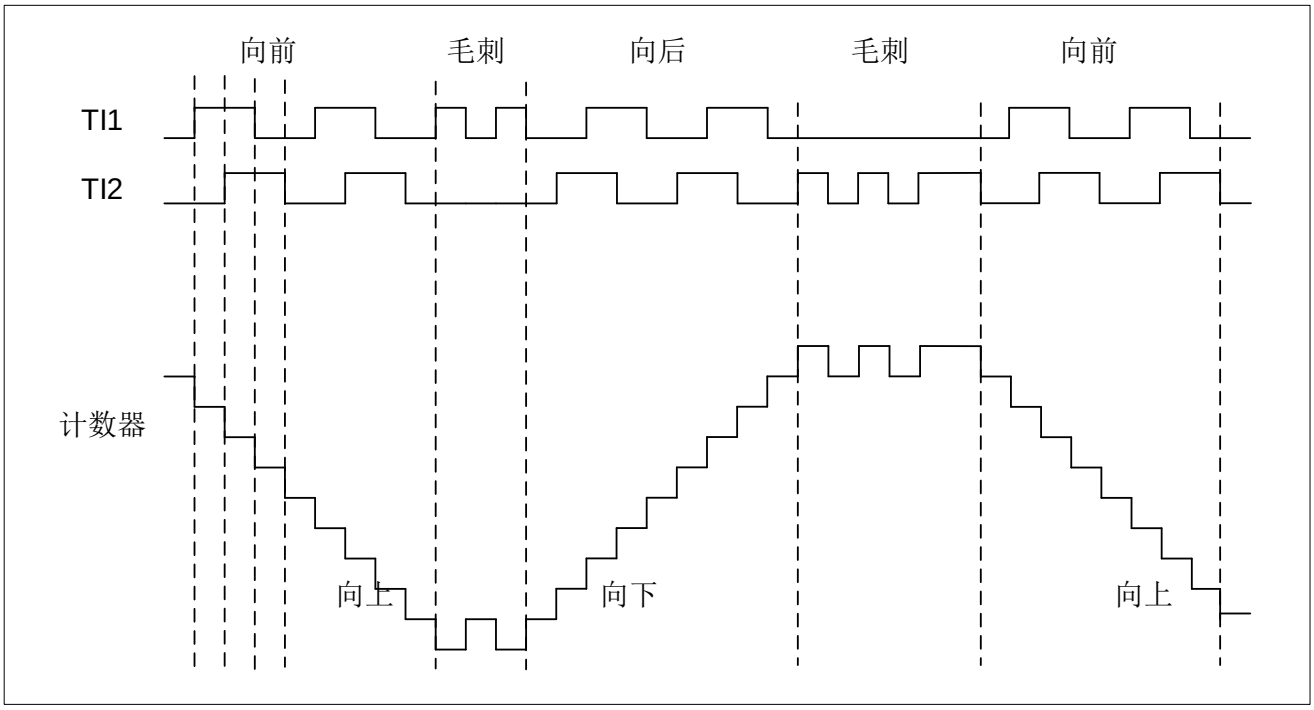


图 14-43 IC1FP1 反相的编码器接口模式实例

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)。

14.3.18 定时器输入异或功能

TIMx_CR2 寄存器中的 TI1S 位，允许通道 1 的输入滤波器连接到一个异或门的输出端，异或门的 3 个输入端为 TIMx_CH1、TIMx_CH2 和 TIMx_CH3。

异或输出能够被用于所有定时器的输入功能，如触发或输入捕获。下节 14.3.18 给出了此特性用于连接霍尔传感器的例子。

14.3.19 与霍尔传感器的接口

使用高级控制定时器(TIMx)产生 PWM 信号驱动马达时，可以用另一个通用 TIM2 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 TIMx_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 14-27)。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 TIMx 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 TIMx。

举例：霍尔输入连接到 TIMx 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 TIMx 的 PWM 配置。

- 置 TIMx_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 TIMx_ARR 为其最大值(计数器必须通过 TI1 的变化清零)。设置预分频器得到一个最大的计数器周期，它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 TIMx_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 TIMx_CCMR1 寄存器中的 OC2M=111 和 CC2S=00。
- 选择 OC2REF 作为 TRGO 上的触发输出：置 TIMx_CR2 寄存器中的 MMS=101。

在高级控制寄存器 TIMx 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(TIMx_CR2 寄存器中 CCPC=1)，同时触发输入控制 COM 事件(TIMx_CR2 寄存器中 CCUS=1)。在一次 COM 事件后，写入下一步的 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

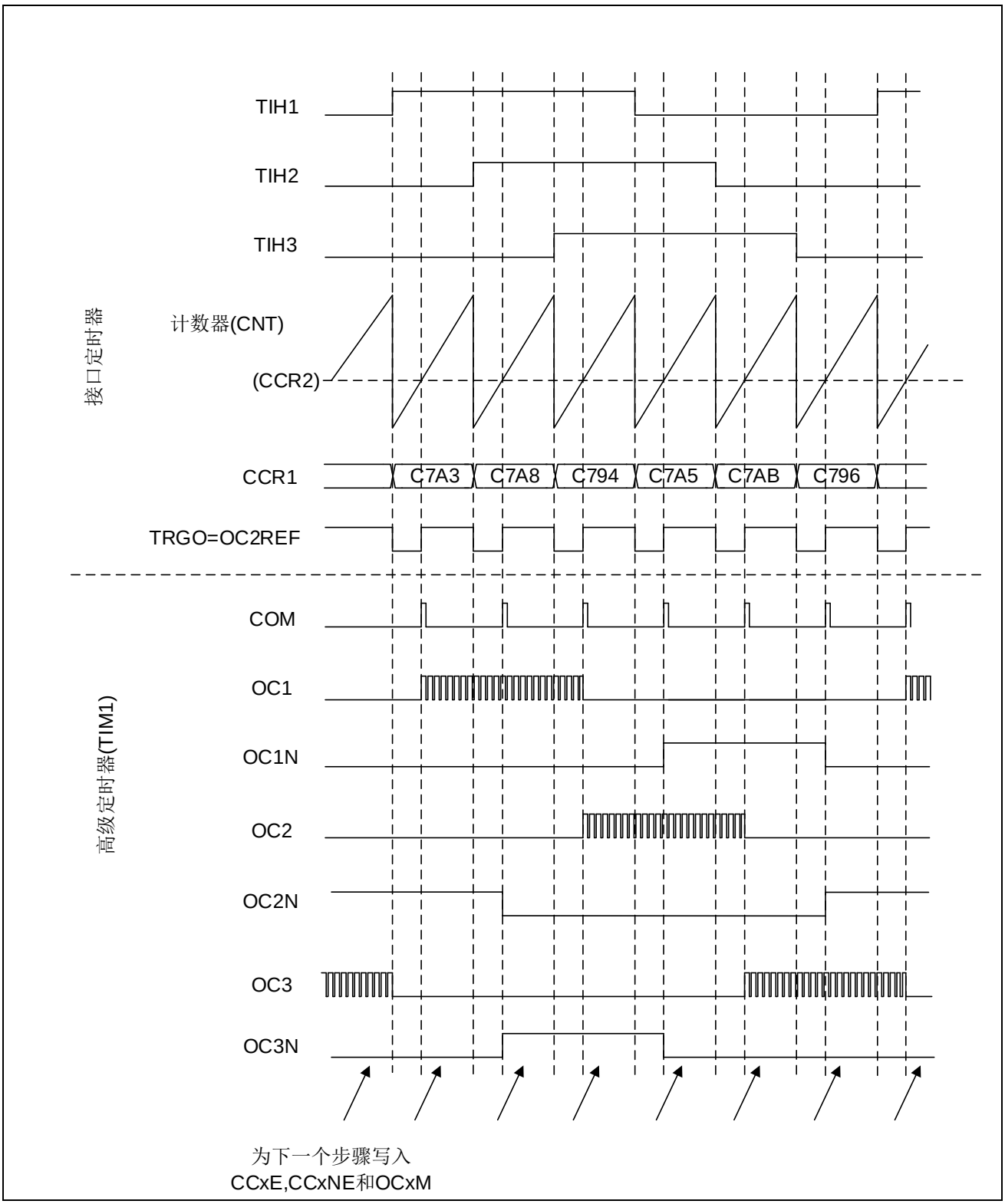


图 14-44 霍尔传感器接口的实例

14.3.20 TIMx 定时器和外部触发的同步

TIMx 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。

14.3.20.1 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 TIMx_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(TIMx_ARR，TIMx_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 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)。
- 置 TIMx_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIMx_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(TIMx_SR 寄存器中的 TIF 位)被设置，根据 TIMx_DIER 寄存器中 TIE(中断使能)位的设置，产生一个中断请求。

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

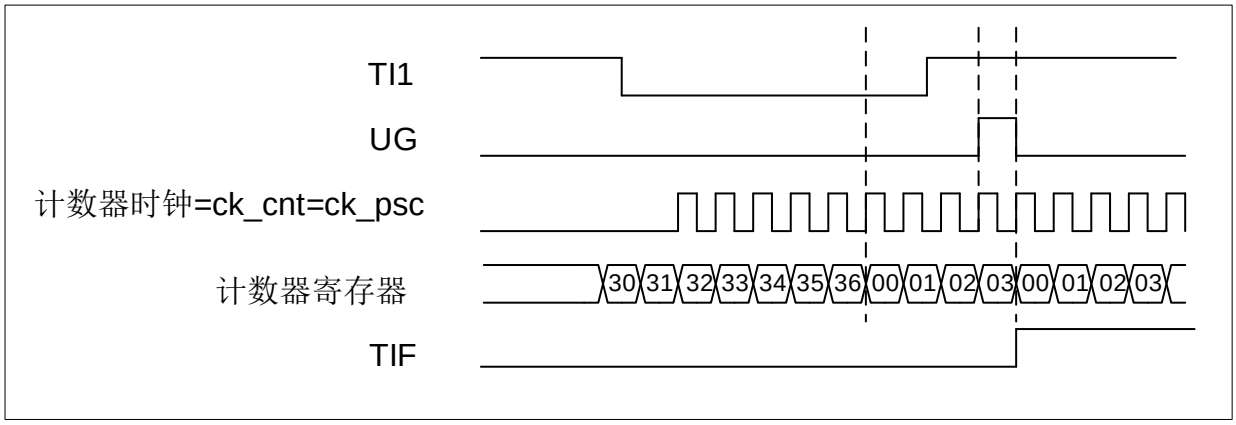


图 14-45 复位模式下的控制电路

14.3.20.2 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 TIMx_CCMR1 寄存器中 CC1S=01。置 TIMx_CCER 寄存器中 CC1P=1 以确定极性(只检测低电平)。
- 置 TIMx_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- 置 TIMx_CR1 寄存器中 CEN=1，启动计数器。在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何。

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

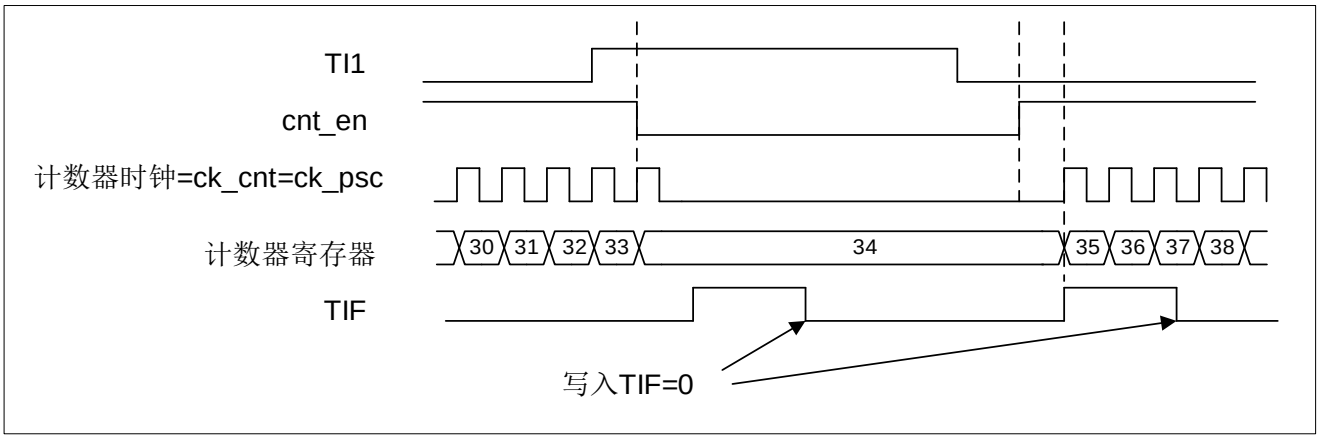


图 14-46 门控模式下的控制电路

14.3.20.3 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 TIMx_CCMR1 寄存器中 CC2S=01。置 TIMx_CCER 寄存器中 CC2P=1 以确定极性 (只检测低电平)。
- 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 TIMx_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器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。

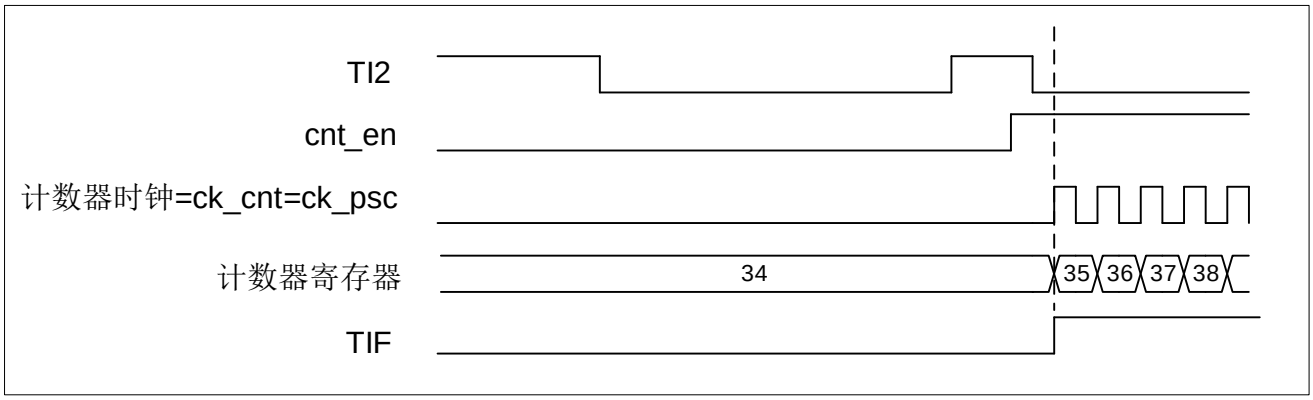


图 14-47 触发器模式下的控制电路

14.3.20.4 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 TIMx_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 TIMx_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 TIMx_CCMR1 寄存器中 CC1S=01，选择输入捕获源
   - 置 TIMx_CCER 寄存器中 CC1P=0 以确定极性(只检测上升沿)
3. 置 TIMx_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 TIMx_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被设置，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。

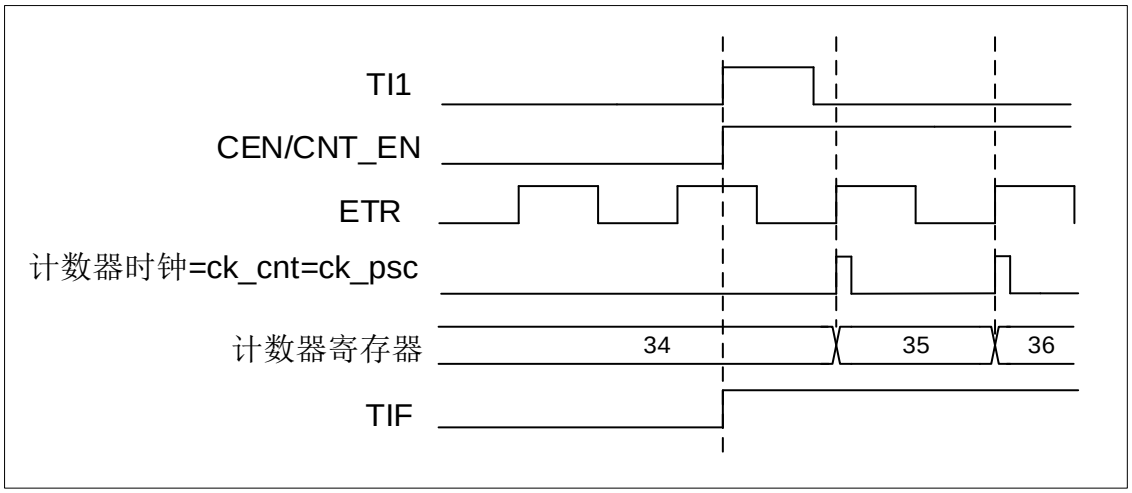


图 14-48 外部时钟模式 2＋触发模式下的控制电路

14.3.21 定时器同步

所有 TIMx 定时器在内部相连，用于定时器同步或链接。当一个定时器处于主模式时，它可以对另一个处于从模式的定时器的计数器进行复位、启动、停止或提供时钟等操作。下图显示了触发选择和主模式选择模块的概况。

14.3.21.1 使用一个定时器作为另一个定时器的预分频器

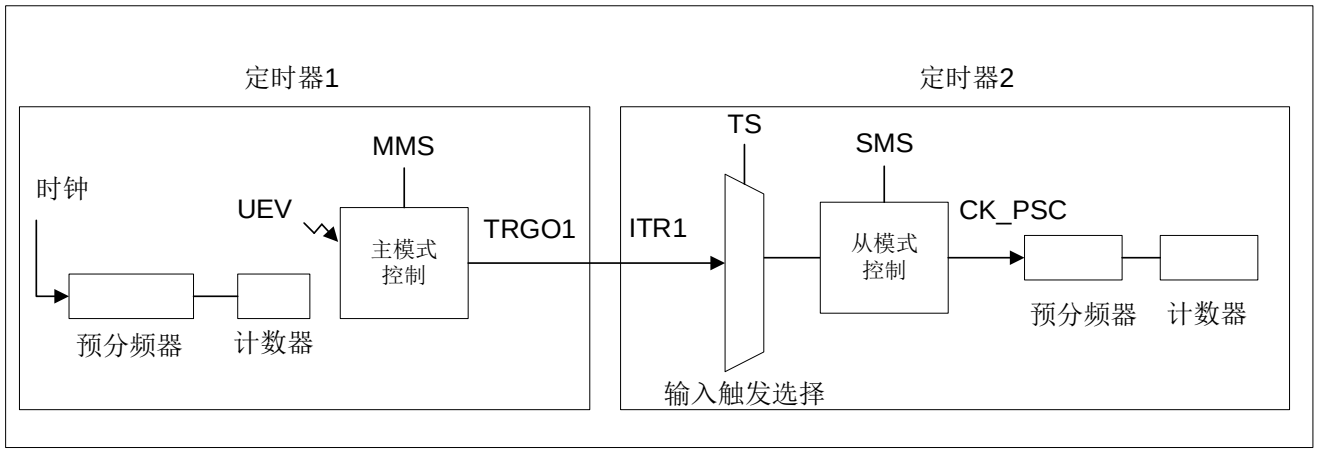


图 14-49 主/从定时器的例子

如：可以配置定时器 1 作为定时器 2 的预分频器，参考图 14-49。进行下述操作：

- 配置定时器 1 为主模式，它可以在每一个更新事件 UEV 时输出一个周期性的触发信号。在 TIM1_CR2 寄存器的 MMS=‘010’时，每当产生一个更新事件时在 TRGO1 上输出一个上升沿信号。
- 连接定时器 1 的 TRGO1 输出至定时器 2，设置 TIM2_SMCR 寄存器的 TS=‘000’，配置定时器 2 为使用 ITR1 作为内部触发的从模式。
- 然后把从模式控制器置于外部时钟模式 1(TIM2_SMCR 寄存器的 SMS=111)；这样定时器 2 即可由定时器 1 周期性的上升沿(即定时器 1 的计数器溢出)信号驱动。

- 最后，必须设置相应(TIM2_CR1 寄存器)的 CEN 位分别启动两个定时器。

注：如果 Ocx 已被选中为定时器 1 的触发输出(MMS=1xx)，它的上升沿用于驱动定时器 2 的计数器。

14.3.21.2 使用一个定时器使能另一个定时器

在这个例子中，定时器 2 的使能由定时器 1 的输出比较控制。参考**错误!未找到引用源。**的连接。只当定时器 1 的 OC1REF 为高时，定时器 2 才对分频后的内部时钟计数。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3($f_{CK\_CNT}=f_{CK\_INT}/3$)得到。

- 配置定时器 1 为主模式，送出它的输出比较参考信号(OC1REF)为触发输出(TIM1_CR2 寄存器的 MMS=100)
- 配置定时器 1 的 OC1REF 波形(TIM1_CCMR1 寄存器)
- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
- 置 TIM2_CR1 寄存器的 CEN=1 以使能定时器 2
- 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1

注：定时器 2 的时钟不与定时器 1 的时钟同步，这个模式只影响定时器 2 计数器的使能信号。

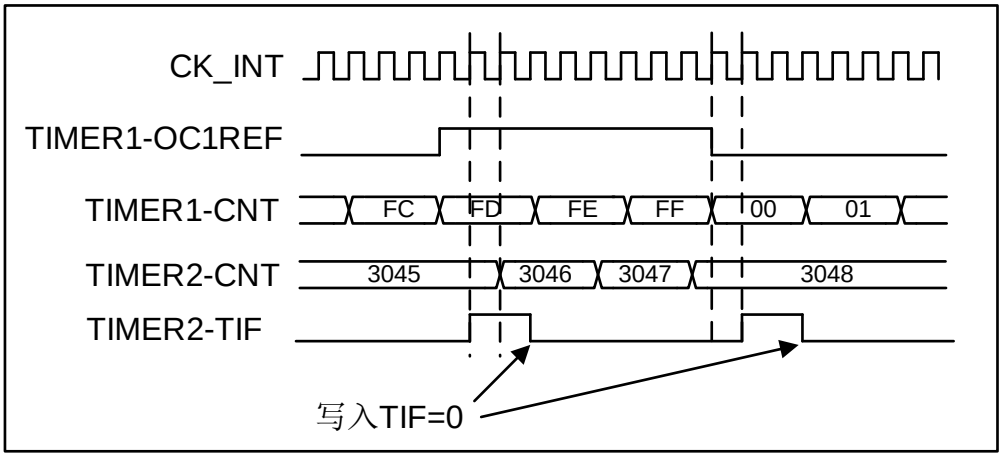


图 14-50 定时器 1 的 OC1REF 控制定时器 2

在图 14-50 的例子中，在定时器 2 启动之前，它们的计数器和预分频器未被初始化，因此它们从当前的数值开始计数。可以在启动定时器 1 之前复位 2 个定时器，使它们从给定的数值开始，即在定时器计数器中写入需要的任意数值。写 TIM2_EGR 寄存器的 UG 位即可复位定时器。

在下一个例子中，需要同步定时器 1 和定时器 2。定时器 1 是主模式并从 0 开始，定时器 2 是从模式并从 0xE7 开始；2 个定时器的预分频器系数相同。写'0'到 TIM1_CR1 的 CEN 位将禁止定时器 1，定时器 2 随即停止。

配置定时器 1 为主模式，送出输出比较 1 参考信号(OC1REF)做为触发输出(TIM1_CR2 寄存器的 MMS=100)。

- 配置定时器 1 的 OC1REF 波形(TIM1_CCMR1 寄存器)。

- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
- 置 TIM1_EGR 寄存器的 UG='1'，复位定时器 1。
- 置 TIM2_EGR 寄存器的 UG='1'，复位定时器 2。
- 写'0xE7'至定时器 2 的计数器(TIM2_CNTL)，初始化它为 0xE7。
- 置 TIM2_CR1 寄存器的 CEN='1'以使能定时器 2。
- 置 TIM1_CR1 寄存器的 CEN='1'以启动定时器 1。
- 置 TIM1_CR1 寄存器的 CEN='0'以停止定时器 1。

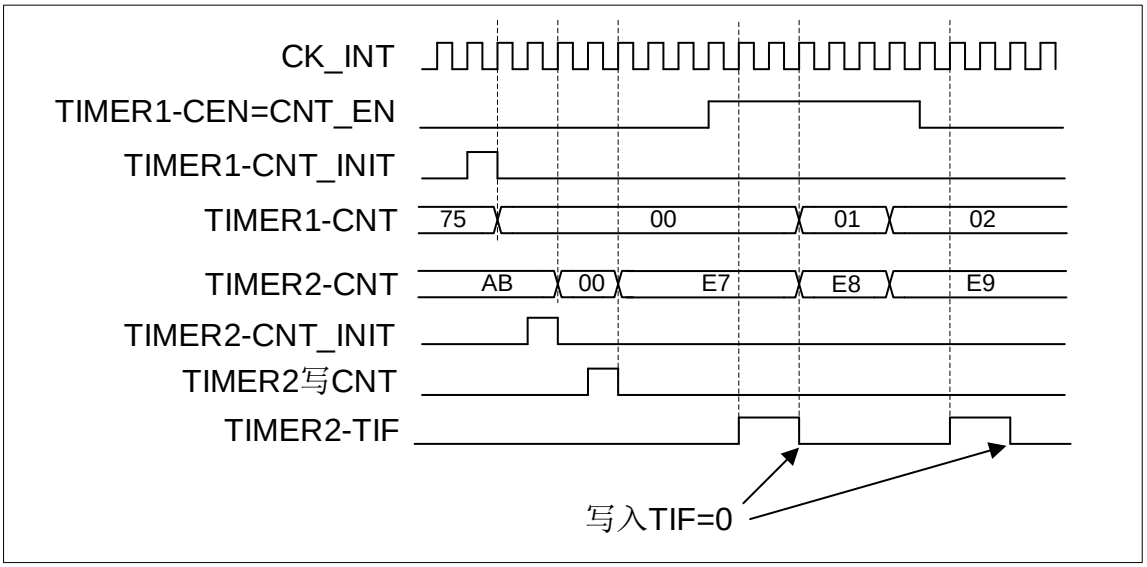


图 14-51 通过使能定时器 1 可以控制定时器 2

14.3.21.3 使用一个定时器去启动另一个定时器

在这个例子中，使用定时器 1 的更新事件使能定时器 2。参考图 14-49 的连接。一旦定时器 1 产生更新事件，定时器 2 即从它当前的数值(可以是非 0)按照分频的内部时钟开始计数。在收到触发信号时，定时器 2 的 CEN 位被自动地置'1'，同时计数器开始计数直到写'0'到 TIM2_CR1 寄存器的 CEN 位。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3($f_{CK\_CNT}=f_{CK\_INT}/3$)。

- 配置定时器 1 为主模式，送出它的更新事件(UEV)做为触发输出(TIM1_CR2 寄存器的 MMS=010)。
- 配置定时器 1 的周期(TIM1_ARR 寄存器)。
- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为触发模式(TIM2_SMCR 寄存器的 SMS=110)
- 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1。

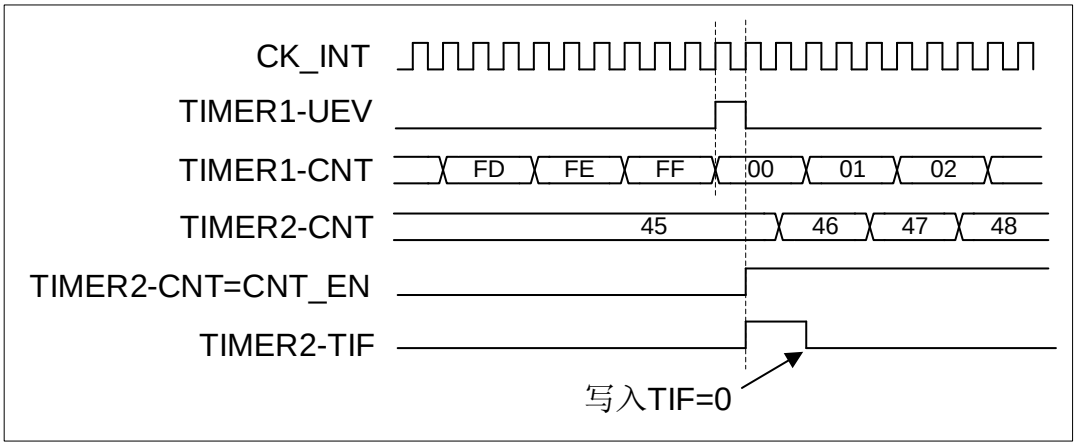


图 14-52 使用定时器 1 的更新触发定时器 2

在上一个例子中，可以在启动计数之前初始化两个计数器。图 14-53 显示在相同配置情况下，使用触发模式而不是门控模式(TIM2_SMCR 寄存器的 SMS=110)的动作。

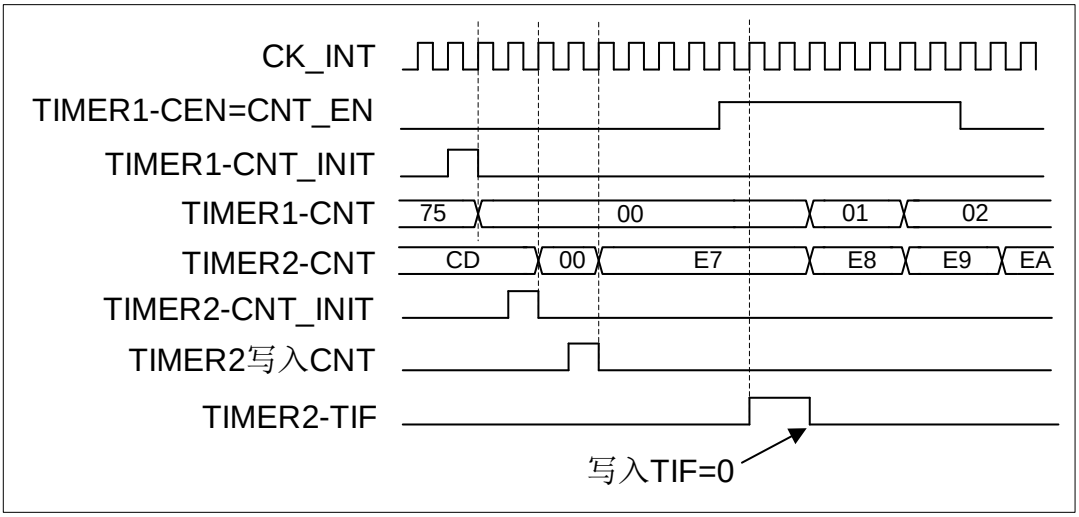


图 14-53 利用定时器 1 的使能触发定时器 2

14.3.21.4 使用一个外部触发同步地启动 2 个定时器

这个例子中当定时器 1 的 TI1 输入上升时使能定时器 1，使能定时器 1 的同时使能定时器 2，参见图 14-49。为保证计数器的对齐，定时器 1 必须配置为主/从模式(对应 TI1 为从，对应定时器 2 为主)：

- 配置定时器 1 为主模式，送出它的使能做为触发输出(TIM1_CR2 寄存器的 MMS='001')
- 配置定时器 1 为从模式，从 TI1 获得输入触发(TIM1_SMCR 寄存器的 TS='100')
- 配置定时器 1 为触发模式(TIM1_SMCR 寄存器的 SMS='110')
- 配置定时器 1 为主/从模式，TIM1_SMCR 寄存器的 MSM='1'
- 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- 配置定时器 2 为触发模式(TIM2_SMCR 寄存器的 SMS='110')

当定时器 1 的 TI1 上出现一个上升沿时，两个定时器同步地按照内部时钟开始计数，两个 TIF 标志也同时被设置。

注：在这个例子中，在启动之前两个定时器都被初始化(设置相应的 UG 位)，两个计数器都从 0 开始，但可以通过写入任意一个计数器寄存器(TIM2_CNT)在定时器间插入一个偏移。下图中能看到主/从模式下在定时器 1 的 CNT_EN 和 CK_PSC 之间有个延迟。

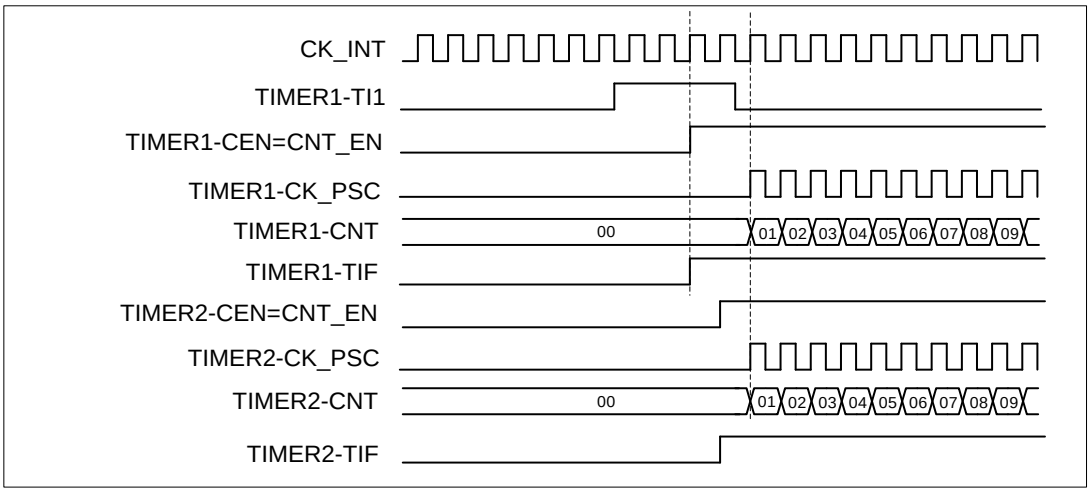


图 14-54 使用定时器 1 的 TI1 输入触发定时器 1 和定时器 2

14.3.22 调试模式

当微控制器进入调试模式时(Cortex-M0+核心停止)，根据 DBG 模块中 DBG_TIMx_STOP 的设置，TIMx 计数器可以或者继续正常操作，或者停止。

14.4 TIMx 寄存器列表

可以用字(32 位)的方式操作这些外设寄存器。

TIM1 基地址 0X40001000

TIM2 基地址 0X40003C00

表 14-2 TIMx 寄存器列表和复位值

偏移地址	名称	描述	复位值
0x00	TIMx_CR1	TIMx 控制寄存器 1	0x00000000
0x04	TIMx_CR2	TIMx 控制寄存器 2	0x00000000
0x08	TIMx_SMCR	TIMx 从模式控制寄存器	0x00000000
0x0C	TIMx_DIER	TIMx 中断使能寄存器	0x00000000
0x10	TIMx_SR	TIMx 状态寄存器	0x00000000
0x14	TIMx_EGR	TIMx 事件产生寄存器	0x00000000
0x18	TIMx_CCMR1	TIMx 捕获/比较模式寄存器 1	0x00000000
0x1C	TIMx_CCMR2	TIMx 捕获/比较模式寄存器 2	0x00000000
0x20	TIMx_CCER	TIMx 捕获/比较使能寄存器	0x00000000
0x24	TIMx_CNT	TIMx 计数器	0x00000000
0x28	TIMx_PSC	TIMx 预分频器	0x00000000
0x2C	TIMx_ARR	TIMx 自动重装载寄存器	0x00000000
0x30	TIMx_RCR	TIMx 重复计数寄存器	0x00000000
0x34	TIMx_CCR1	TIMx 捕获/比较寄存器 1	0x00000000
0x38	TIMx_CCR2	TIMx 捕获/比较寄存器 2	0x00000000
0x3C	TIMx_CCR3	TIMx 捕获/比较寄存器 3	0x00000000
0x40	TIMx_CCR4	TIMx 捕获/比较寄存器 4	0x00000000
0x44	TIMx_BDTR	TIMx 刹车和死区寄存器	0x00000000

14.5 TIMx 寄存器说明

14.5.1 TIMx 控制寄存器 1(TIMx_CR1)

偏移地址：0x00

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留						CKD		ARPE	CMS		DIR	OPM	URS	UDIS	CEN
						R/W		R/W	R/W		R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:10	Res	保留，始终读为 0。	0x0	-
9:8	CKD	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	0x0	R/W
7	ARPE	自动重装载预装载允许位 (Auto-reload preload enable) 0：TIMx_ARR 寄存器没有缓冲 1：TIMx_ARR 寄存器被装入缓冲器	0x0	R/W
6:5	CMS	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(TIMx_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(TIMx_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(TIMx_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。	0x0	R/W
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。	0x0	R/W
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	0x0	R/W
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，则下述任一事件产生更新中断：	0x0	R/W

		- 计数器溢出/下溢 - 设置 UG 位 - 从模式控制器产生的更新 1：如果使能了更新中断，则只有计数器溢出/下溢才产生更新中断。		
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。	0x0	R/W
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。	0x0	R/W

14.5.2 TIMx 控制寄存器 2(TIMx_CR2)

偏移地址：0x04

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS			保留	CCUS	保留	CCPC
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W				R/W		R/W

位	标记	功能描述	复位值	读写
31:15	Res	保留，始终读为 0。	0x0	-
14	OIS4	输出空闲状态 4(OC4 输出)。参见 OIS1 位。	0x0	R/W
13	OIS3N	输出空闲状态 3(OC3N 输出)。参见 OIS1N 位。	0x0	R/W
12	OIS3	输出空闲状态 3(OC3 输出)。参见 OIS1 位。	0x0	R/W
11	OIS2N	输出空闲状态 2(OC2N 输出)。参见 OIS1N 位。	0x0	R/W
10	OIS2	输出空闲状态 2(OC2 输出)。参见 OIS1 位。	0x0	R/W
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(TIMx_BKR 寄存器)级别 1、2 或 3 后，该位不能被修改。	0x0	R/W
8	OIS1	输出空闲状态 1(OC1 输出) (Output Idle state 1) 0：当 MOE=0 时，如果 OC1N 有效，则死区后 OC1=0 1：当 MOE=0 时，如果 OC1N 有效，则死区后 OC1=1 注：已经设置了 LOCK(TIMx_BKR 寄存器)级别 1、2 或 3 后，该位不能被修改。	0x0	R/W
7	TI1S	TI1 选择(TI1 selection) 0：TIMx_CH1 引脚连到 TI1 输入 1：TIMx_CH1、TIMx_CH2 和 TIMx_CH3 引脚经异或后连到 TI1 输入	0x0	R/W
6:4	MMS	主模式选择(Master mode selection) 这 3 位用于选择在主模式下送到从定时器的同步信息(TRGO)。可能的组合如下： 000：复位：TIMx_EGR 寄存器的 UG 位被用于作为触发输出(TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟。 001：使能：计数器使能信号 CNT_EN 被用于作为触发输出(TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式(见 TIMx_SMCR 寄存器中 MSM 位的描述)。 010：更新：更新事件被选为触发输出(TRGO 时)。例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011：捕获/比较脉冲：在发生一次捕获或一次比较成功时，当要设置	0x0	R/W

		CC1IF 标志时(即使它已经为高)，触发输出送出一个正脉冲(TRGO)。 100：比较：OC1REF 信号被用于作为触发输出(TRGO)。 101：比较：OC2REF 信号被用于作为触发输出(TRGO)。 110：比较：OC3REF 信号被用于作为触发输出(TRGO)。 111：比较：OC4REF 信号被用于作为触发输出(TRGO)。		
3	Res	保留	0x0	-
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。	0x0	R/W
1	Res	保留，始终读为 0。	0x0	-
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。	0x0	R/W

14.5.3 TIMx 从模式控制寄存器(TIMx_SMCR)

偏移地址：0x08

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS		ETF				MSM	TS			保留	SMS		
R/W	R/W	R/W		R/W				R/W	R/W				R/W		

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
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	0x0	R/W
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。	0x0	R/W
13:12	ETPS	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 TIMxCLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。 00：关闭预分频 01：ETRP 频率除以 2 10：ETRP 频率除以 4 11：ETRP 频率除以 8	0x0	R/W
11:8	ETF	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。 00xx：无滤波器，以 f_{CK_INT} 采样 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8	0x0	R/W

		1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8		
7	MSM	主/从模式 (Master/slave mode) 0：无作用 1：触发输入(TRGI)上的事件被延迟了，以允许在当前定时器(通过 TRGO)与它的从定时器间的完美同步，这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。	0x0	R/W
6:4	TS	触发选择 (Trigger selection) 这 3 位选择用于同步计数器的触发输入 000：内部触发 0(ITR0) 100：TI1 的边沿检测器(TI1F_ED) 001：内部触发 1(ITR1) 101：滤波后的定时器输入 1(TI1FP1) 010：内部触发 2(ITR2) 110：滤波后的定时器输入 2(TI2FP2) 011：内部触发 3(ITR3) 111：外部触发输入(ETRF) 更多有关 ITRx 的细节，参见表 14-3。 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。	0x0	R/W
3	Res	保留，始终读为 0。	0x0	-
2:0	SMS	从模式选择(Slave mode selection) 当选择了外部信号，触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 000：关闭从模式：如果 CEN=1，则预分频器直接由内部时钟驱动。 001：编码器模式 1：根据 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。	0x0	R/W

表 14-3 TIMx 内部触发连接

主定时器	从定时器	ITR0 (TS=000)	ITR1 (TS=001)	ITR2 (TS=010)	ITR3 (TS=011)
TIM2	TIM1	TIM2_trgo	irq_TIM10	irq_TIM11	irq_pca
TIM1	TIM2	TIM1_trgo	irq_TIM10	irq_TIM11	irq_pca

14.5.4 TIMx 中断使能寄存器(TIMx_DIER)

偏移地址：0x0C

复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:8	Res	保留，始终读为 0。	0x0	-
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	0x0	R/W
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	0x0	R/W
5	COMIE	允许 COM 中断 (COM interrupt enable) 0：禁止 COM 中断 1：允许 COM 中断	0x0	R/W
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	0x0	R/W
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	0x0	R/W
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	0x0	R/W
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	0x0	R/W
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	0x0	R/W

14.5.5 TIMx 状态寄存器(TIMx_SR)

偏移地址：0x10

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留			CC4OF	CC3OF	CC2OF	CC1OF	保留	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF
			RCW0	RCW0	RCW0	RCW0		RCW0	RCW0	RCW0	RCW0	RCW0	RCW0	RCW0	RCW0

位	标记	功能描述	复位值	读写
31:13	Res	保留，始终读为 0。	0x0	-
12	CC4OF	捕获/比较 4 重复捕获标记 (Capture/Compare 4 overcapture flag) 参见 CC1OF 描述。	0x0	RCW0
11	CC3OF	捕获/比较 3 重复捕获标记 (Capture/Compare 3 overcapture flag) 参见 CC1OF 描述。	0x0	RCW0
10	CC2OF	捕获/比较 2 重复捕获标记 (Capture/Compare 2 overcapture flag) 参见 CC1OF 描述。	0x0	RCW0
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 TIMx_CCR1 寄存器时，CC1IF 的状态已经为'1'	0x0	RCW0
8	Res	保留，始终读为 0。	0x0	-
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	0x0	RCW0
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	0x0	RCW0
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	0x0	RCW0
4	CC4IF	捕获/比较 4 中断标记 (Capture/Compare 4 interrupt flag) 参考 CC1IF 描述。	0x0	RCW0
3	CC3IF	捕获/比较 3 中断标记 (Capture/Compare 3 interrupt flag) 参考 CC1IF 描述。	0x0	RCW0
2	CC2IF	捕获/比较 2 中断标记 (Capture/Compare 2 interrupt flag)	0x0	RCW0

		参考 CC1IF 描述。		
1	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 TIMx_CR1 寄存器的 CMS 位)，它由软件清'0'。 0：无匹配发生 1：TIMx_CNT 的值与 TIMx_CCR1 的值匹配 当 TIMx_CCR1 的内容大于 TIMx_ARR 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 TIMx_CCR1 清'0'。 0：无输入捕获产生 1：计数器值已被捕获(拷贝)至 TIMx_CCR1(在 IC1 上检测到与所选极性相同的边沿)	0x0	RCW0
0	UIF	更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置'1'，它由软件清'0'。 0：无更新事件产生 1：更新中断等待响应。当寄存器被更新时该位由硬件置'1' - 若 TIMx_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 - 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当设置 TIMx_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 - 若 TIMx_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。 (参考 14.5.3 TIMx 从模式控制寄存器(TIMx_SMCR))。	0x0	RCW0

14.5.6 TIMx 事件产生寄存器(TIMx_EGR)

偏移地址:0x14

复位值:0x00000000

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

位	标记	功能描述	复位值	读写
31:8	Res	保留，始终读为 0。	0x0	-
7	BG	产生刹车事件，该位由软件置'1'，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，则产生相应的中断	0x0	WO
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：TIMx_SR 寄存器的 TIF=1，若开启对应的中断，则产生相应的中断	0x0	WO
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。	0x0	WO
4	CC4G	产生捕获/比较 4 事件 (Capture/Compare 4 generation) 参考 CC1G 描述。	0x0	WO
3	CC3G	产生捕获/比较 3 事件 (Capture/Compare 3 generation) 参考 CC1G 描述。	0x0	WO
2	CC2G	产生捕获/比较 2 事件 (Capture/Compare 2 generation) 参考 CC1G 描述。	0x0	WO
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，则产生相应的中断。 若通道 CC1 配置为输入： 当前的计数器值被捕获至 TIMx_CCR1 寄存器；设置 CC1IF=1，若开启对应的中断，则产生相应的中断。若 CC1IF 已经为 1，则设置 CC1OF=1。	0x0	WO
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	0x0	WO

		也被清'0'(但是预分频系数不变)。若在中心对称模式下或 DIR=0(向上计数)则计数器被清'0'；若 DIR=1(向下计数)则计数器取 TIMx_ARR 的值。		

14.5.7 TIMx 捕获/比较模式寄存器 1(TIMx_CCMR1)

偏移地址：0x18

复位值：0x00000000

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
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M			OC2PE	OC2FE	CC2S		OC1CE	OC1M			OC1PE	OC1FE	CC1S	
IC2F				IC2PSC				IC1F				IC1PSC			
R/W															

输出比较模式：

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
15	OC2CE	输出比较 2 清 0 使能 (Output Compare 2 clear enable)	0x0	R/W
14:12	OC2M	输出比较 2 模式 (Output Compare 2 mode)	0x0	R/W
11	OC2PE	输出比较 2 预装载使能 (Output Compare 2 preload enable)	0x0	R/W
10	OC2FE	输出比较 2 快速使能 (Output Compare 2 fast enable)	0x0	R/W
9:8	CC2S	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 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。	0x0	R/W
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。	0x0	R/W
6:4	OC1M	输出比较 1 模式 (Output Compare 1 mode) 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、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 TIMx_CCR1 与计数器 TIMx_CNT 间的比较对 OC1REF 不起作用； 001：匹配时设置通道 1 为有效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为	0x0	R/W

		高。 010：匹配时设置通道 1 为无效电平。当计数器 TIMx_CNT 的值与捕获/比较寄存器 1 (TIMx_CCR1)相同时，强制 OC1REF 为低。 011：翻转。当 TIMx_CCR1=TIMx_CNT 时，翻转 OC1REF 的电平。 100：强制为无效电平。强制 OC1REF 为低。 101：强制为有效电平。强制 OC1REF 为高。 110：PWM 模式 1，在向上计数时，一旦 TIMx_CNT<TIMx_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 TIMx_CNT>TIMx_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111：PWM 模式 2，在向上计数时，一旦 TIMx_CNT<TIMx_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 TIMx_CNT>TIMx_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。		
3	OC1PE	输出比较 1 预装载使能 (Output Compare 1 preload enable) 0：禁止 TIMx_CCR1 寄存器的预装载功能，可随时写入 TIMx_CCR1 寄存器，并且新写入的数值立即起作用。 1：开启 TIMx_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIMx_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1：一旦 LOCK 级别设为 3(TIMx_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2：仅在单脉冲模式下(TIMx_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。	0x0	R/W
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：触发输入上出现有效边沿相当于 CC1 输出上的比较匹配。随后，无论比较结果如何，OC 都设置为比较电平。采样触发输入和激活 CC1 输出的延迟时间缩短为 3 个时钟周期。仅当通道配置为 PWM1 或 PWM2 模式时，OCFE 才会起作用。	0x0	R/W
1:0	CC1S	捕获/比较 1 选择。(Capture/Compare 1 selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上；	0x0	R/W

		11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。		

输入捕获模式：

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
15:12	IC2F	输入捕获 2 滤波器 (Input capture 2 filter)	0x0	R/W
11:10	IC2PSC	输入/捕获 2 预分频器 (Input capture 2 prescaler)	0x0	R/W
9:8	CC2S	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 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(TIMx_CCER 寄存器的 CC2E=0)才是可写的。	0x0	R/W
7:4	IC1F	输入捕获 1 滤波器 (Input capture 1 filter) 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 00xx：无滤波器，以 f_{CK_INT} 采样 0100：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=6 0101：采样频率 $f_{SAMPLING}=f_{DTS}/2$，N=8 0110：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=6 0111：采样频率 $f_{SAMPLING}=f_{DTS}/4$，N=8 1000：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=6 1001：采样频率 $f_{SAMPLING}=f_{DTS}/8$，N=8 1010：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=5 1011：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=6 1100：采样频率 $f_{SAMPLING}=f_{DTS}/16$，N=8 1101：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=5 1110：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=6 1111：采样频率 $f_{SAMPLING}=f_{DTS}/32$，N=8	0x0	R/W
3:2	IC1PSC	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(TIMx_CCER 寄存器中)，则预分频器复位。 00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获； 01：每 2 个事件触发一次捕获； 10：每 4 个事件触发一次捕获； 11：每 8 个事件触发一次捕获。	0x0	R/W

1:0	CC1S	捕获/比较 1 选择 (Capture/Compare 1 Selection) 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00：CC1 通道被配置为输出； 01：CC1 通道被配置为输入，IC1 映射在 TI1 上； 10：CC1 通道被配置为输入，IC1 映射在 TI2 上； 11：CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(TIMx_CCER 寄存器的 CC1E=0)才是可写的。	0x0	R/W

14.5.8 TIMx 捕获/比较模式寄存器 2(TIMx_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
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4CE	OC4M			OC4PE	OC4FE	CC4S		OC3CE	OC3M			OC3PE	OC3FE	CC3S	
IC4F				IC4PSC				IC3F				IC3PSC			
R/W															

输出比较模式：

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
15	OC4CE	输出比较 4 清 0 使能 (Output Compare 4 clear enable)	0x0	R/W
14:12	OC4M	输出比较 4 模式 (Output Compare 4 mode)	0x0	R/W
11	OC4PE	输出比较 4 预装载使能 (Output Compare 4 preload enable)	0x0	R/W
10	OC4FE	输出比较 4 快速使能 (Output Compare 4 fast enable)	0x0	R/W
9:8	CC4S	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 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。	0x0	R/W
7	OC3CE	输出比较 3 清'0'使能 (Output Compare 3 clear enable)	0x0	R/W
6:4	OC3M	输出比较 3 模式 (Output Compare 3 mode)	0x0	R/W
3	OC3PE	输出比较 3 预装载使能 (Output Compare 3 preload enable)	0x0	R/W
2	OC3FE	输出比较 3 快速使能 (Output Compare 3 fast enable)	0x0	R/W
1:0	CC3S	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：	0x0	R/W

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

输入捕获模式：

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
15:12	IC4F	输入捕获 4 滤波器 (Input capture 4 filter)	0x0	R/W
11:10	IC4PSC	输入/捕获 4 预分频器 (Input capture 4 prescaler)	0x0	R/W
9:8	CC4S	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 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时(TIMx_CCER 寄存器的 CC4E=0)才是可写的。	0x0	R/W
7:4	IC3F	输入捕获 3 滤波器 (Input capture 3 filter)	0x0	R/W
3:2	IC3PSC	输入/捕获 3 预分频器 (Input capture 3 prescaler)	0x0	R/W
1:0	CC3S	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 TIMx_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时(TIMx_CCER 寄存器的 CC3E=0)才是可写的。	0x0	R/W

14.5.9 TIMx 捕获/比较使能寄存器(TIMx_CCER)

偏移地址：0x20

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留		CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:14	Res	保留，始终读为 0。	0x0	-
13	CC4P	输入/捕获 4 输出极性 (Capture/Compare 4 output polarity) 参考 CC1P 的描述。	0x0	R/W
12	CC4E	输入/捕获 4 输出使能 (Capture/Compare 4 output enable) 参考 CC1E 的描述。	0x0	R/W
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。	0x0	R/W
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。	0x0	R/W
9	CC3P	输入/捕获 3 输出极性 (Capture/Compare 3 output polarity) 参考 CC1P 的描述。	0x0	R/W
8	CC3E	输入/捕获 3 输出使能 (Capture/Compare 3 output enable) 参考 CC1E 的描述。	0x0	R/W
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。	0x0	R/W
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。	0x0	R/W
5	CC2P	输入/捕获 2 输出极性 (Capture/Compare 2 output polarity) 参考 CC1P 的描述。	0x0	R/W
4	CC2E	输入/捕获 2 输出使能 (Capture/Compare 2 output enable) 参考 CC1E 的描述。	0x0	R/W
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(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2 且 CC1S=00(通道配置为输出)则该位不能被修改。	0x0	R/W
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。	0x0	R/W
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；	0x0	R/W

		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(TIMx_BDTR 寄存器中的 LOCK 位)设为 3 或 2，则该位不能被修改。		
0	CC1E	输入/捕获 1 输出使能 (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 TIMx_CCR1 寄存器。 0：捕获禁止； 1：捕获使能。	0x0	R/W

表 14-4 带刹车功能的互补输出通道 Ocx 和 OcxN 的控制位

控制位					输出状态	
MOE 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) Ocx=0	输出禁止(与定时器断开) OcxN=0
		0	0	1	输出禁止(与定时器断开) Ocx=0	OCxREF + 极性， OcxN= OCxREF xor CCxNP
		0	1	0	OCxREF + 极性， Ocx= OCxREF xor CCxP	输出禁止(与定时器断开) OcxN=0
		0	1	1	OCxREF + 极性 + 死区	OCxREF 反相 + 极性 + 死区
		1	0	0	输出禁止(与定时器断开) Ocx=CCxP	输出禁止(与定时器断开) OcxN=CCxNP
		1	0	1	关闭状态(输出使能且为无效电平) Ocx=CCxP	OCxREF + 极性，OcxN= OCxREF xor CCxNP
		1	1	0	OCxREF + 极性， Ocx= OCxREF xor CCxP	关闭状态(输出使能且为无效电平) OcxN=CCxNP
		1	1	1	OCxREF + 极性 + 死区	OCxREF 反相 + 极性 + 死区
0	0	X	0	0	输出禁止(与定时器断开) 异步地：Ocx=CCxP，Ocx_EN=0，OcxN=CCxNP； 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。	
	0		0	1		
	0		1	0		
	0		1	1		
	1		0	0	关闭状态(输出使能且为无效电平) 异步地：Ocx=CCxP，Ocx_EN=1，OcxN=CCxNP； 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	1		
	1		1	0		
	1		1	1		

3. 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。

14.5.10 TIMx 计数器(TIMx_CNT)

偏移地址：0x24
复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
15:0	CNT	计数器的值(Counter value)	0x0	R/W

14.5.11 TIMx 预分频器(TIMx_PSC)

偏移地址：0x28
复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
15:0	PSC	预分频器的值(Prescaler value) 计数器的时钟频率(CK_CNT)等于 f_{CK_PSC}/(PSC+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’。	0x0	R/W

14.5.12 TIMx 自动重装载寄存器(TIMx_ARR)

偏移地址：0x2C
复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
15:0	ARR	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。	0x0	R/W

14.5.13 TIMx 重复计数寄存器(TIMx_RCR)

偏移地址：0x30
复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:8	Res	保留，始终读为 0。	0x0	-
7:0	REP	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 TIMx_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(REP+1)对应着： - 在边沿对齐模式下，PWM 周期的数目； - 在中心对称模式下，PWM 半周期的数目；	0x0	R/W

14.5.14 TIMx 捕获/比较寄存器 1(TIMx_CCR1)

偏移地址：0x34
复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
15:0	CCR1	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 TIMx_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 TIMx_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。	0x0	R/W

14.5.15 TIMx 捕获/比较寄存器 2(TIMx_CCR2)

偏移地址：0x38
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2															
R/W															

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
15:0	CCR2	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 TIMx_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 TIMx_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：	0x0	R/W

		CCR2 包含了由上一次输入捕获 2 事件(IC2)传输的计数器值。		

14.5.16 TIMx 捕获/比较寄存器 3(TIMx_CCR3)

偏移地址：0x3C
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3															
R/W															

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
15:0	CCR3	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 TIMx_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 TIMx_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。	0x0	R/W

14.5.17 TIMx 捕获/比较寄存器 4(TIMx_CCR4)

偏移地址：0x40
复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
15:0	CCR4	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 TIMx_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 TIMx_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。	0x0	R/W

14.5.18 TIMx 刹车和死区寄存器(TIMx_BDTR)

偏移地址：0x44

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK		DTG							
R/W	R/W	R/W	R/W	R/W	R/W	R/W		R/W							

注：根据锁定设置，AOE、BKP、BKE、OSSI、OSSR 和 DTG 位均可被写保护，有必要在第一次写入 TIMx_BDTR 寄存器时对它们进行配置。

位	标记	功能描述	复位值	读写
31:16	Res	保留，始终读为 0。	0x0	-
15	MOE	主输出使能 (Main output enable) 一旦刹车输入有效，该位被硬件异步清'0'。根据 AOE 位的设置值，该位可以由软件清'0'或被自动置 1。它仅对配置为输出的通道有效。 0：禁止 OC 和 OCN 输出或强制为空闲状态； 1：如果设置了相应的使能位(TIMx_CCER 寄存器的 CcxE、CcxNE 位)，则开启 OC 和 OCN 输出。 有关 OC/OCN 使能的细节，参见 TIMx 捕获/比较使能寄存器 (TIMx_CCER)。	0x0	R/W
14	AOE	自动输出使能 (Automatic output enable) 0：MOE 只能被软件置'1'； 1：MOE 能被软件置'1'或在下一个更新事件被自动置'1'(如果刹车输入无效)。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为'1'，则该位不能被修改。	0x0	R/W
13	BKP	刹车输入极性 (Break polarity) 0：刹车输入低电平有效； 1：刹车输入高电平有效。 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为'1'，则该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。	0x0	R/W
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(TIMx_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。应当先配置 BKP 位，再配置 BKE 位。	0x0	R/W
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。 参考 OC/OCN 使能的详细说明(14.5.9 节，TIMx 捕获/比较使能寄存器 (TIMx_CCER))。	0x0	R/W

		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(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。		
10	OSSI	空闲模式下“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。 参考 OC/OCN 使能的详细说明(14.5.9 节，TIMx 捕获/比较使能寄存器(TIMx_CCER))。 0：当定时器不工作时，禁止 OC/OCN 输出(OC/OCN 使能输出信号=0)； 1：当定时器不工作时，一旦 CcxE=1 或 CcxNE=1，OC/OCN 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(TIMx_BDTR 寄存器中的 LOCK 位)设为 2，则该位不能被修改。	0x0	R/W
9:8	LOCK	锁定设置 (Lock configuration) 该位为防止软件错误而提供写保护。 00：锁定关闭，寄存器无写保护； 01：锁定级别 1，不能写入 TIMx_BDTR 寄存器的 DTG、BKE、BKP、AOE 位和 TIMx_CR2 寄存器的 OISx/OISxN 位； 10：锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位(一旦相关通道通过 CCxS 位设为输出，CC 极性位是 TIMx_CCER 寄存器的 CCxP/CCxNP 位)以及 OSSR/OSSI 位； 11：锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位(一旦相关通道通过 CCxS 位设为输出，CC 控制位是 TIMx_CCMRx 寄存器的 OcxM/OcxPE 位)； 注：在系统复位后，只能写一次 LOCK 位，一旦写入 TIMx_BDTR 寄存器，则其内容冻结直至复位。	0x0	R/W
7:0	DTG	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 × Tdtg，Tdtg = T_{DTS}； DTG[7:5]=10x => DT=(64+DTG) × Tdtg，Tdtg = 2 × T_{DTS}； DTG[7:5]=110 => DT=(32+DTG) × Tdtg，Tdtg = 8 × T_{DTS}； DTG[7:5]=111 => DT=(32+DTG)× Tdtg，Tdtg = 16 × T_{DTS}； 例：若 T_{DTS} = 125ns(8MHZ)，可能的死区时间为： 0 到 15875ns，若步长时间为 125ns； 16us 到 31750ns，若步长时间为 250ns； 32us 到 63us，若步长时间为 1us； 64us 到 126us，若步长时间为 2us； 注：一旦 LOCK 级别(TIMx_BDTR 寄存器中的 LOCK 位)设为 1、2 或 3，则不能修改这些位。	0x0	R/W

15 基础定时器(TIM10,TIM11)

15.1 基础定时器简介

基础定时器包含两个定时器 TIM10/11，TIM10/11 功能完全相同。TIM10/11 是同步定时/计数器，可以作为 16/32 位自动重装载功能的定时/计数器，也可以作为 16/32 位无重载功能的定时/计数器。TIM10/11 可以对外部脉冲进行计数或者实现系统定时。

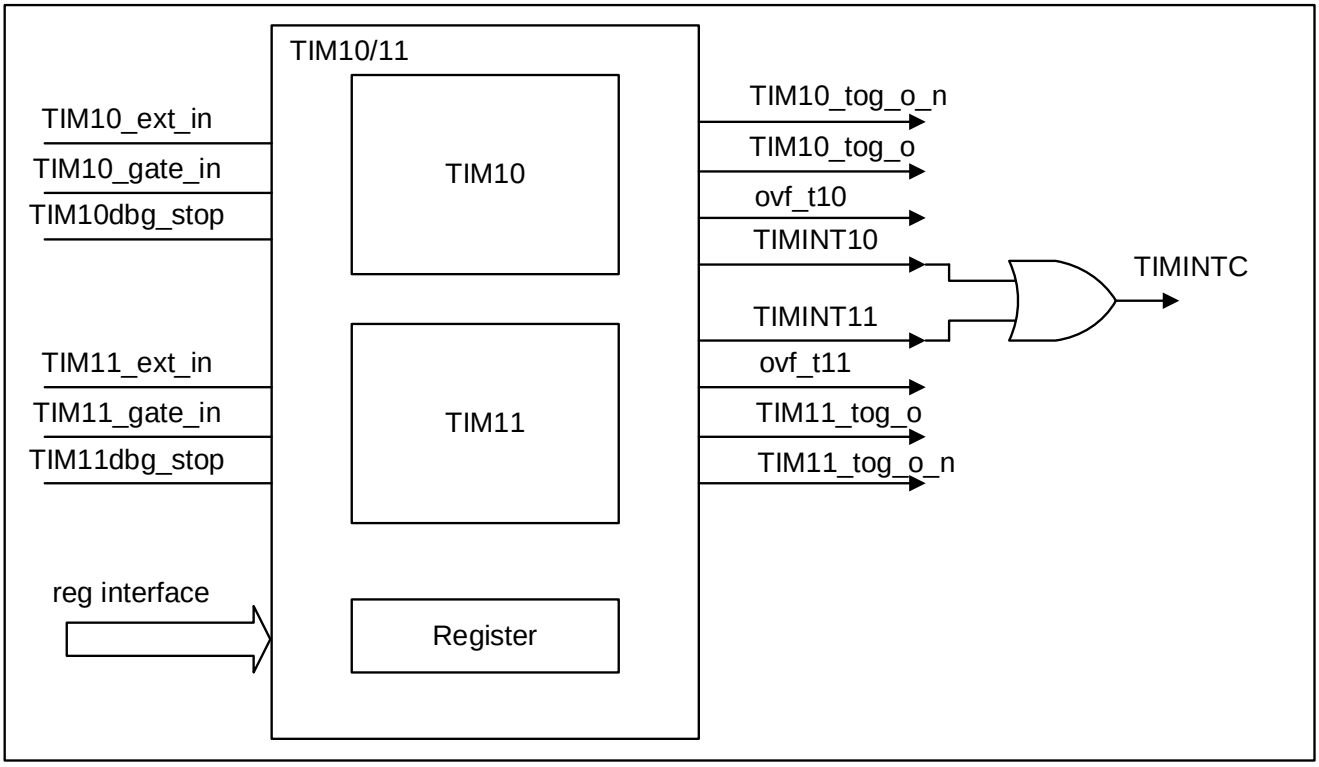


图 15-1 基础定时器框图

15.2 基础定时器功能描述

TIM10/11 每个定时/计数器都有独立的控制启动信号，以及外部输入时钟，门控信号。

TIM10/11 使用 EXT，GATE 来进行计数功能，EXT 用于计数器的外部输入时钟信号，GATE 用于有效电平计数使能信号。当门控功能使能后，当且仅当外部输入 GATE 电平有效时，计数器才会计数，否则计数器处于保持状态。门控使能使用 TIMx_CR.GATE_EN 控制。默认门控功能关闭。门控电平选择使用 TIMx_CR.GATE_P 控制。默认高电平为门控有效电平；设置 TIMx_CR.GATE_P 为 1 后，门控低电平为有效电平。

TIM10/11 使用 PCLK，GATE 来进行定时功能，PCLK 用于定时器的内部输入时钟信号，GATE 可用于有效电平定时使能信号。当门控功使能后，当且仅当外部输入 GATE 电平有效时，定时器才会计数，否则定时器处于定时计数器停止状态。门控使能使用 TIMx_CR.GATE_EN 控制。默认门控功能关闭。门控电平选择使用 TIMx_CR.GATE_P 控制。默认高电平为门控有效电平；设置为 1 后门控有效电平是低电平。定时功能可以配置预分频。TIMx_CR.TMR_PRSC 控制分频比。

TMR_PRE	000	001	010	011	100	101	110	111

分频比	1	2	4	8	16	32	64	128

TIM10/11 支持定时/计数器两种功能，可通过设置定时器控制寄存器(TIMx_CR)中 CT_SEL 进行配置。每种功能支持 2 种模式，模式 1 为 16/32 位自由计数模式，模式 2 是 16/32 位重载模式。

在模式 1 自由计数模式

计数到最大值(16 位 Max=0xFFFF,32 位最大值为 0xFFFFFFFF)溢出后产生中断，定时/计数器清零，然后继续计数；

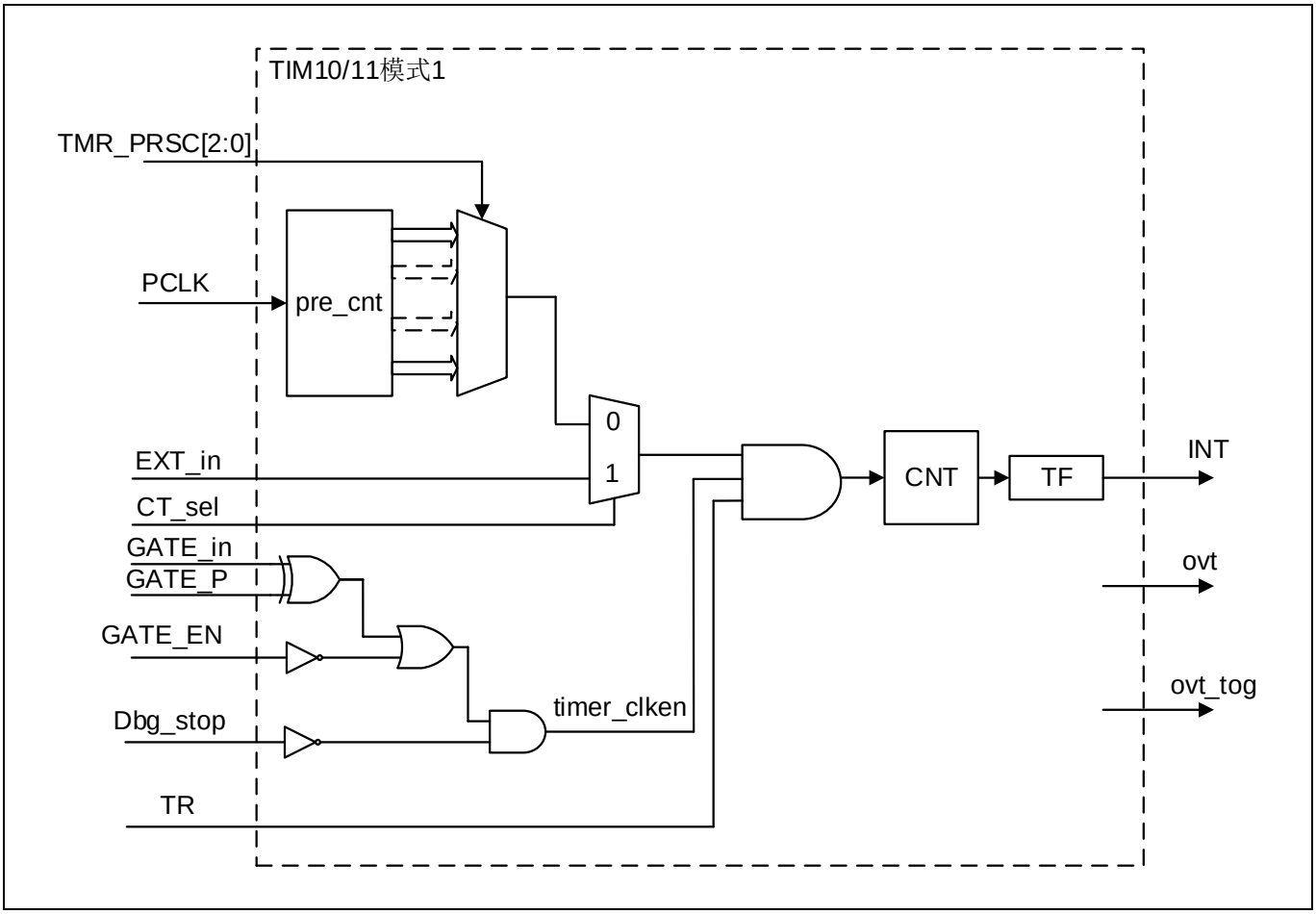


图 15-2 Timer 模式 1 框图

模式 2 重载模式

重载模式，计数到最大值后溢出，产生中断，定时/计数器的值被装载为 BGLOAD 的值，然后继续向上计数。在重载模式下，定时时间设的小的情况下需要考虑软件处理速度，否则中断会来不及处理而造成中断丢失。

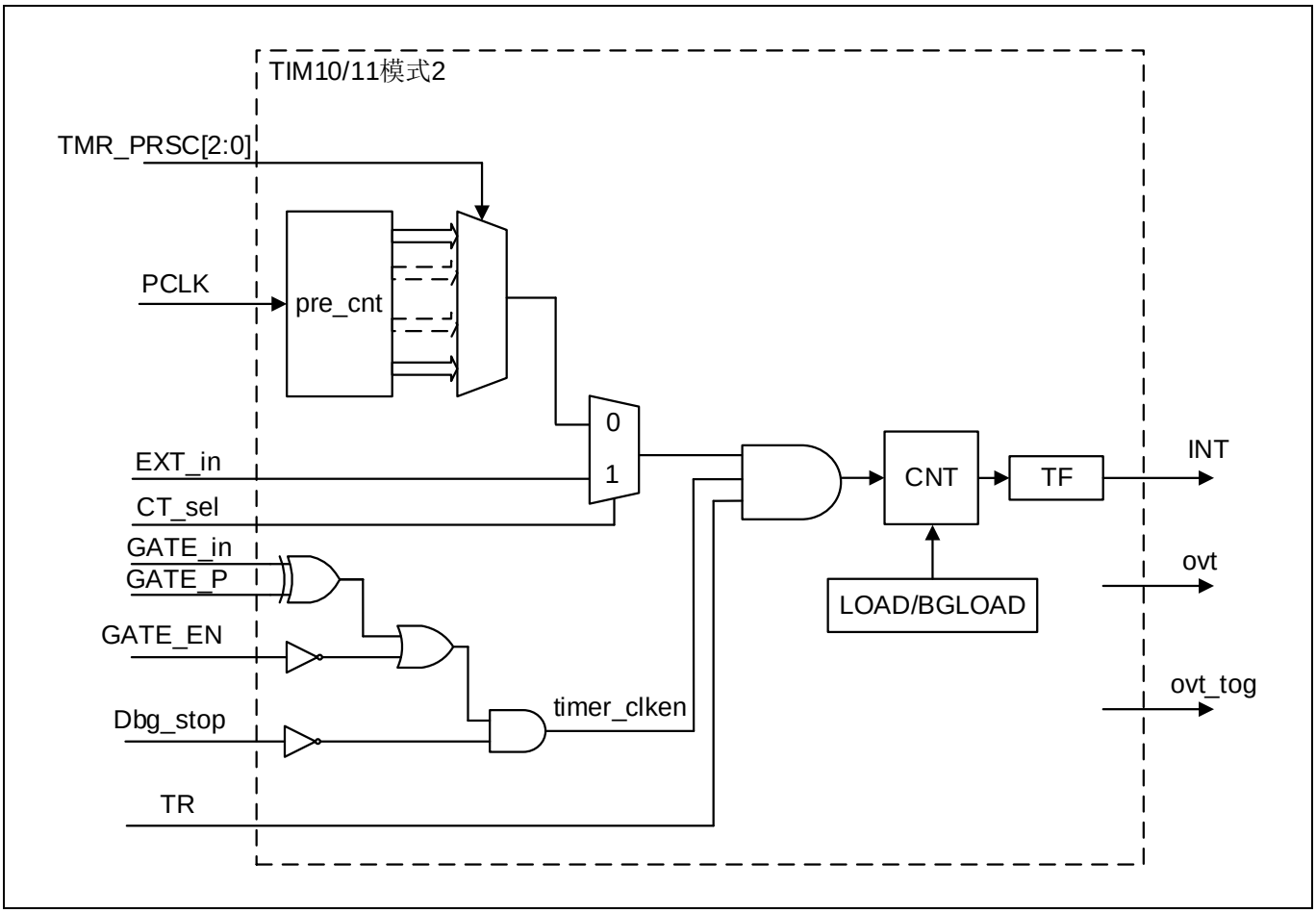


图 15-3 Timer 模式 2 框图

当设置对应定时器 TIMx_CR.TR 为 1 后定时器开始运行。模式 1，启动后从寄存器设定的初值开始计数，计数到最大后产生溢出中断，然后从 0 继续计数。模式 2，启动后从寄存器初值 CNT 开始向上计数，计数到最大值后产生中断后重载寄存器 BGLOAD 的值到计数器 CNT 中，继续向上计数。不论是自由计数模式，还是重载模式，只要是写 LOAD 值，都会立即更新定时/计数器的值，然后继续向上计数。

15.2.1 计数功能

计数功能用于测定某个事件发生的次数。在计数功能中，计数器在每个相应的输入时钟的下降沿累加一次。输入信号被内部的 PCLK 采样，因此外部输入时钟频率不能超过系统的 PCLK 时钟。计数到最大值会溢出并且产生中断。中断标志需要软件清除。

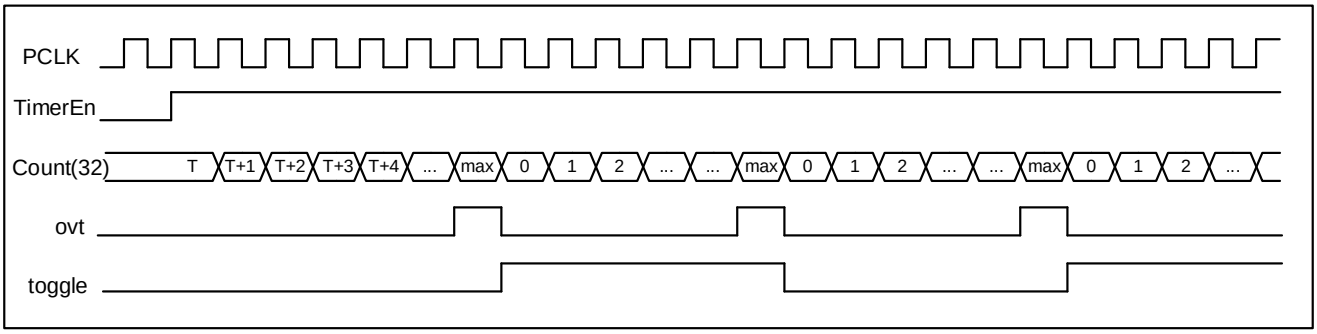


图 15-4 32 位模式 1 时序图(max=0xFFFF FFFF)

15.2.2 定时功能

定时功能用于产生间隔定时。在定时功能中，定时器有预分频，定时器在每个预分频的一个时钟累加一次，计数到最大值会溢出并且产生中断。中断标志需要软件清除。

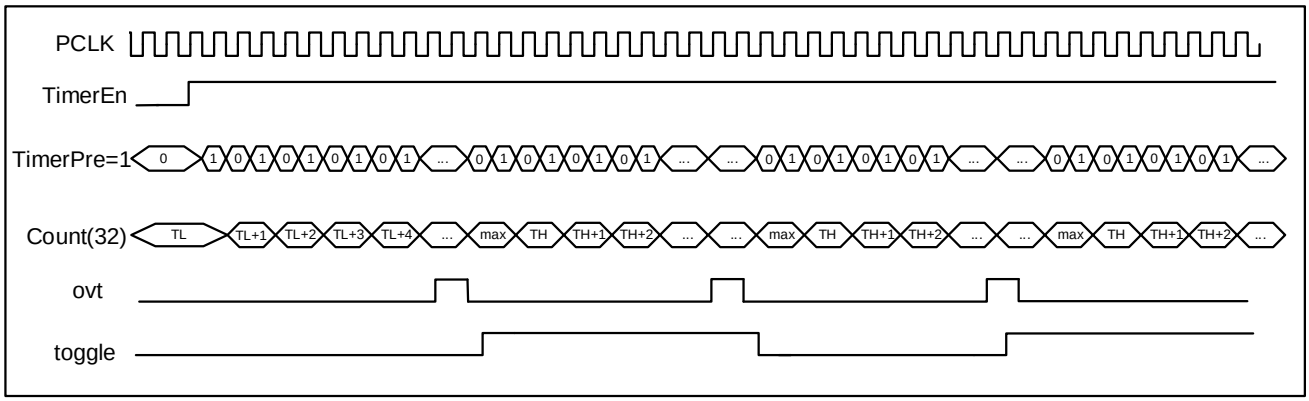


图 15-5 32 位模式 2 时序图(PCLK 二分频,max=0xFFFF FFFF)

15.2.3 Buzzer 功能

通过定时器的翻转输出功能可以实现驱动外部 Buzzer 的功能。TIMx_CR.TOG_EN 为 1 时，TOG,TOGN 输出反向。设置 TIMx_CR.TOG_EN 为 0 可以同时设置端口 TOG，TOGN 输出为 0。在计数时钟为 4M 情况下 Buzzer 输出不同频率的 Timer 重载模式配置如下：

Buzzer 频率	计数周期	计数值	重载值	CNT 初始值	LOAD 重载值
1KHz	0.5ms	2000	63536	0xF830	0xF830
2KHz	0.25ms	1000	64536	0xFC18	0xFC18

4Khz	0.125ms	500	65036	0xFE0C	0xFE0C

15.3 基础定时器互连

15.3.1 GATE 互联

GATE 输入可以从端口直接输入，还可以配置为 VC 的输出作为 GATE 信号。TIM10/11 的 GATE 都可以配置。

通过内部互联配置，可以测量 VC 比较输出的脉冲宽度，可以实现外部控制计数。

配置选择 RX 输入在 SYSCON_PORTCR 寄存器控制，VC 控制在 VC 控制寄存器控制。端口选择。VC 的输出选择优先级最高。

15.3.2 Toggle 输出互联

TIM10 的翻转输出 TIM10_tog_o 到内部模块 UART0，控制 UART0 的波特率；TIM11 的翻转输出 TIM11_tog_o 到内部模块 UART1，控制 UART1 的波特率；TIM10/11 的翻转输出还输出到端口上，可以驱动外部 Buzzer 实现蜂鸣器的控制。

15.4 基础定时器寄存器列表

x=10,11;

基础定时器基地址 0X40001800

	偏移地址	描述
TIM10	0x00	TIM0 偏移地址
TIM11	0x100	TIM1 偏移地址

表 15-1 基础定时器寄存器列表和复位值

偏移地址	名称	描述	复位值
0x00	TIMx_CR	控制寄存器	0x00000000
0x04	TIMx_LOAD	32 位立即重载寄存器	0x00000000
0x08	TIMx_CNT	读计数器寄存器，只读	0x00000000
0x0C	TIMx_RAWINTSR	读原始中断寄存器,	0x00000000
0x10	TIMx_MSKINTSR	读中断寄存器	0x00000000
0x14	TIMx_INTCLR	中断清除寄存器	0x00000000
0x18	TIMx_BGLOAD	32 位周期重载寄存器	0x00000000

15.5 基础定时器寄存器说明

15.5.1 控制寄存器(TIMx_CR)

偏移地址：0x00

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				GATE_P	GATE_EN	TOG_EN	CT_SEL	TR	MODE	INTEN	TMR_SIZE	ONESHOT	TMR_PRSC		
				R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		

位	标记	描述	复位值	读写
31:12	Res	保留位，读为 0	0x0	-
11	GATE_P	端口 GATE 极性控制，默认高电平 gate 有效，设置为 1 后低电平有效 0：高电平有效 1：低电平有效	0x0	R/W
10	GATE_EN	定时器门控 0：无门控，TR=1时定时器工作； 1：只有端口 GATE 有效并且 TR=1 时才工作；	0x0	R/W
9	TOG_EN	TOG 输出使能 0：TOG,TOGN 同时输出 0 1：TOG,TOGN 输出相位相反的信号。可供 BEEP 使用。	0x0	R/W
8	CT_SEL	计数器/定时器功能选择 0：定时器功能，定时器由 PCLK 来进行计数。 1：计数器功能，计数器由外部输入的下降沿进行计数。外部输入由 PCLK 采样，外部输入时钟频率要低于 1/2 采样时钟。	0x0	R/W
7	TR	定时器运行控制 0：定时器停止 1：定时器运行	0x0	R/W
6	MODE	定时器工作模式 0：模式 1 计数器/定时器 1：模式 2 自动重装载计数器/定时器	0x0	R/W
5	INTEN	中断使能控制，写 1 后使能中断	0x0	R/W
4	TMR_SIZE	TimerSize=0，max count value=0xFFFF; TimerSize=1，max count value=0xFFFFFFFF;	0x0	R/W
3	ONESHOT	计数器运行一次使能 0：重复模式 1：oneshot 模式	0x0	R/W
2:0	TMR_PRSC	TIM 预分频选择。 000：分频数 1； 001：分频数 2； 010：分频数 4； 011：分频数 8； 100：分频数 16； 101：分频数 32； 110：分频数 64； 111：分频数 128	0x0	R/W

15.5.2 立即重载寄存器(TIMx_LOAD)

偏移地址：0x04
复位值：0x00000000

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

位	标记	描述	复位值	读写
31:0	LOAD	立即重载寄存器 写此寄存器，会立即更新 Count 计数器的值 读 0x4，0x18，读到最近更新的 Load 或者 BGLOAD 寄存器	0x0	R/W

15.5.3 计数器寄存器(TIMx_CNT)

偏移地址：0x08
复位值：0x00000000

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

位	标记	描述	复位值	读写
31:0	CNT	计数器寄存器	0x0	RO

15.5.4 原始中断状态寄存器(TIMx_RAWINTSR)

偏移地址：0x0C
复位值：0x00000000

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

位	标记	描述	复位值	读写
31:1	Res	保留位，读为 0	0x0	-
0	RIS	中断标志，硬件置位；不论 IntEnable=0 or 1，都可以读中断	0x0	RO

15.5.5 中断标志寄存器(TIMx_MSKINTSR)

偏移地址：0x10
复位值：0x00000000

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

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

位	标记	描述	复位值	读写
31:1	Res	保留位，读为 0	0x0	-
0	TF	INTEN=1，才可以读中断寄存器	0x0	RO

15.5.6 中断清除寄存器(TIMx_INTCLR)

偏移地址：0x14
复位值：0x00000000

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

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

位	标记	描述	复位值	读写
31:1	Res	保留位，读为 0	0x0	-
0	INTCLR	中断标志清除，写 1 清除，写 0 无效	0x0	WO

15.5.7 周期重载寄存器(TIMx_BGLOAD)

偏移地址：0x018
复位值：0x00000000

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

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

位	标记	描述	复位值	读写
31:0	BGLOAD	BACKGROUND 周期重载寄存器,写此寄存器不会立即更新 Count 计数器的值。只有当 Count 值溢出时，才会重新装载 BGLOAD 的值到 Count 计数器里面。 读 0x4，0x18，读到为最近更新的 Load 或者 BGLOAD 寄存器的值。	0x0	R/W

16 低功耗定时器(LPTIM)

LPTIM 是异步 16 位定时/计数器，在系统时钟关闭后仍然可以通过内部低时钟 LIRC 或者外部低速晶体振荡时钟 LXT 计时/计数，通过中断在低功耗模式下唤醒系统。

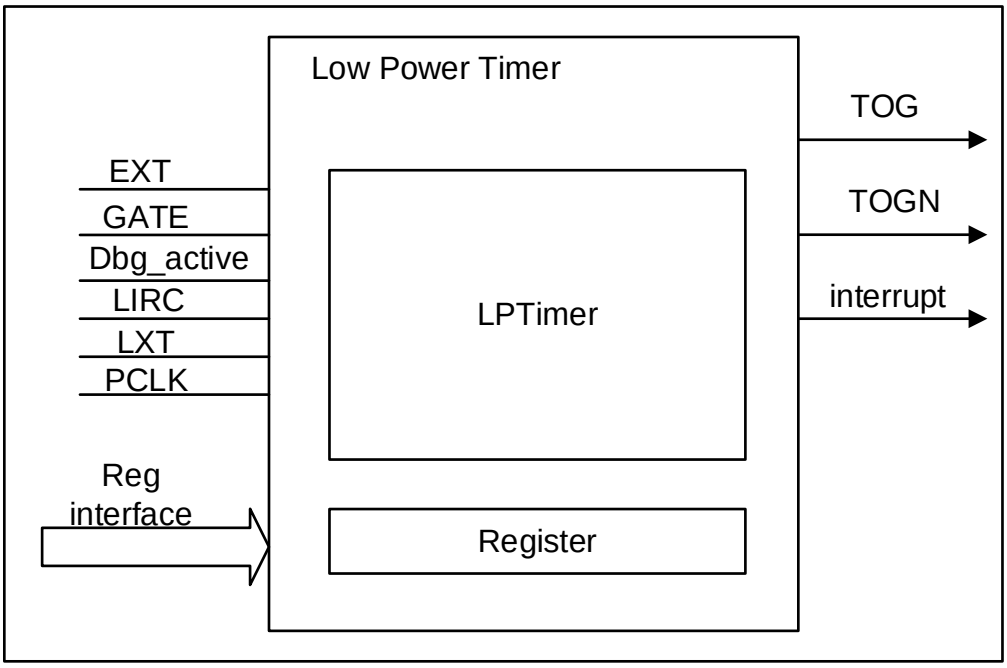


图 16-1 LPTIM 结构框图

16.1 LPTIM 功能描述

LPTIM 具有独立的控制启动信号，以及外部输入时钟，门控信号。

LPTIM 使用 EXT，GATE 来进行计数功能，EXT 用于计数器的外部输入信号，GATE 为有效电平计数使能信号。

LPTIM 的定时器支持两种工作模式，通过设置定时器控制寄存器(LPTIM_CR)中 MODE 选择工作模式。

模式 1 为 16 位自由计数模式，计数器从 LPTIM_LOAD 设定的值开始计数，溢出后计数值从 0x0000 开始重新计数。

模式 2 为 16 位重载模式，LPTIM 启动时会自动装载重载寄存器 LPTIM_LOAD 的值到计数器中，当溢出后会自动装载 LPTIM_BGLOAD 的值到计数器中。用户应当先配置一次 LPTIM_LOAD 之后只需要再配置 LPTIM_BGLOAD 的值即可。

LPTIM 可选三种时钟作为定时器时钟，通过控制寄存器 LPTIM_CR.TCK_SEL 来选择。默认选择 PCLK。时钟选择如表：

表 16-1 LPTIM 时钟选择表

TCK_SEL	00	01	10	11
定时器时钟	PCLK	PCLK	LXT	LIRC
读定时器计数值	读经过同步	无同步	读经过同步	读经过同步

当选择相应的时钟源，然后设定 TCK_EN 为 1 后可以打开计数器的计数时钟源。

当时钟源选择并打开后，设置对应定时器的 TIM_RUN 为 1 后定时器开始运行。

对于模式 1 和模式 2，如果设置 LOAD 值，任何时刻，计数值会被立即更新为 LOAD 值，计数器从 LOAD 值开始重新计数。设置 LOAD 值的优先级高于其他计数器被更新的优先级。

在模式 2，设定 BGLOAD 的值，只有在计数器发生溢出后，设定值才会被更新到计数器。

模式 1：如果没有设置 LOAD 值，计数器从 0 开始计数，计数到最大 0xFFFF 后产生溢出中断。计数器计数到最大 0xFFFF 后，计数器再次从 0 开始计数。

模式 2：如果没有设置 LOAD 值，计数器从 0 开始计数，计数到最大 0xFFFF 后产生溢出中断。计数器计数到最大 0xFFFF 后，计数值会被更新为 LPTIM_BGLOAD 的值，然后向上计数。

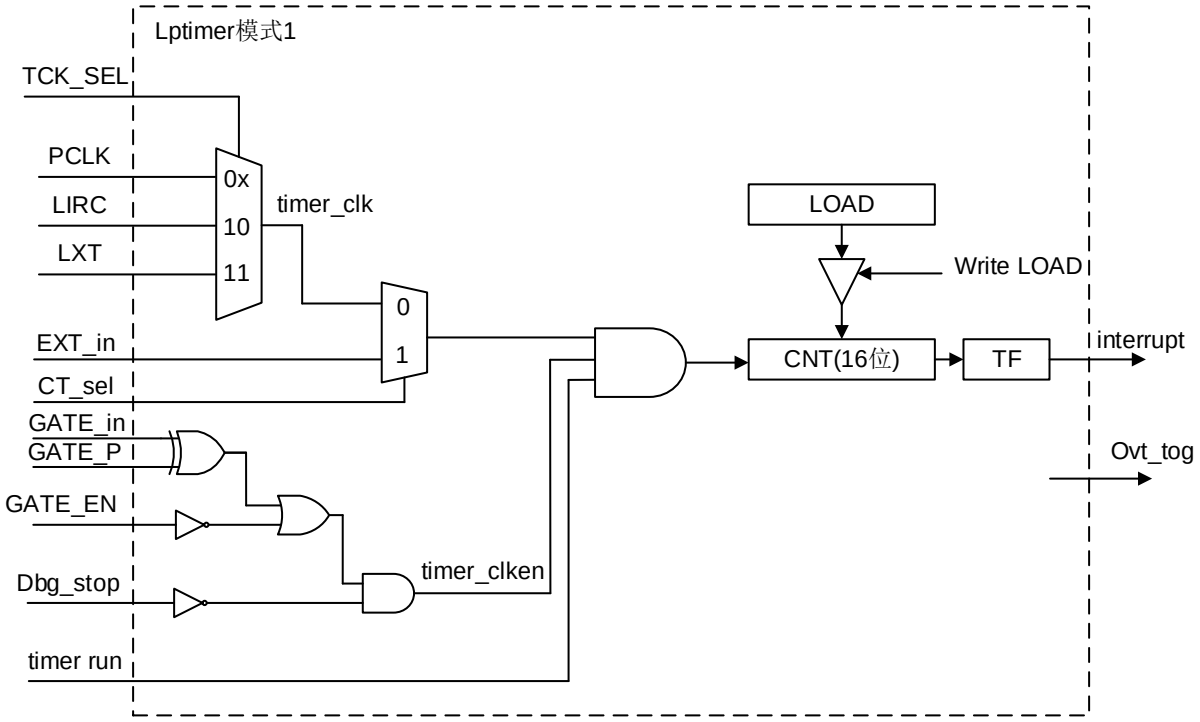


图 16-2 LPTIM 模式 1

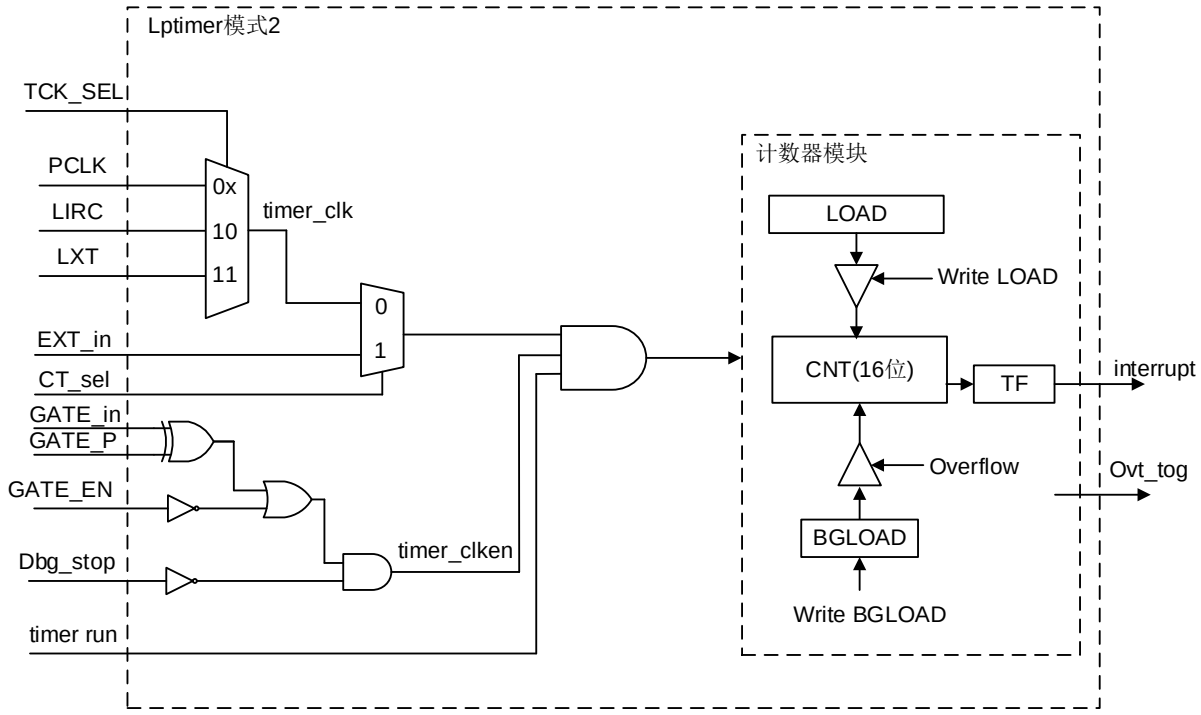


图 16-3 LPTIM 模式 2

16.1.1 计数功能

计数功能用于测定某个事件发生的次数。在计数功能中，计数器在每个相应的输入信号的下降沿累加一次。输入信号被内部的计数时钟采样，因此外部输入信号频率不能超过系统的计数时钟频率的 1/2。计数到最大值会溢出并且产生中断。

16.1.2 定时功能

定时功能用于产生间隔定时。在定时功能中，定时器一个时钟累加一次，计数到最大值会溢出并且产生中断。

16.2 LPTIM 互连

16.2.1 GATE 互联

GATE 输入可以从端口直接输入，也可以输入 UART 的 RX 信号；

通过内部互联配置，可以实现 UART 波特率的自动识别，可以测量 VC 比较输出的脉冲宽度，可以实现外部控制计数。

配置选择 RX 输入在 SYSCON_PORTCR 寄存器控制，VC 控制在 VC 控制寄存器控制。

16.2.2 EXT 互联

EXT 输入可以从端口直接输入，也配置为 VC 的输入作为 EXT 信号。

通过内部互联配置，可以测量 VC 脉冲计数。VC 输出控制寄存器在 VC 控制模块。

16.2.3 TOGGLE 输出互联

LPTIM 的翻转输出到端口上，可以驱动外部 BUZZER 实现蜂鸣器的控制。

16.3 寄存器列表

基地址：0X40004400

表 16-2 LPTIM 寄存器列表和复位值

偏移地址	名称	描述	复位值
0x00	LPTIM_CNTVAL	LPTIM 计数值只读寄存器	0x00000000
0x04	LPTIM_CR	LPTIM 控制寄存器	0x00000000
0x08	LPTIM_LOAD	LPTIM 立即重载寄存器	0x00000000
0x0C	LPTIM_INTSR	LPTIM 中断寄存器	0x00000000
0x10	LPTIM_INTCLR	LPTIM 中断清除寄存器	0x00000000
0x14	LPTIM_BGLOAD	LPTIM 周期重载寄存器	0x00000000

16.4 寄存器说明

16.4.1 LPTIM 计数值只读寄存器(LPTIM_CNTVAL)

地址偏移：0x00
复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15:0	LPT_CNT	计数值只读寄存器	0x0	RO

16.4.2 LPTIM 控制寄存器(LPTIM_CR)

地址偏移：0x04
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															WT_FLAG
															RO
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留						TCK_EN	INT_EN	GATE_POLE	GATE	TCK_SEL		TOG_EN	CT_SEL	MODE	TIMER_RUN
						R/W	R/W	R/W	R/W	R/W		R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:17	Res	保留	0x0	–
16	WT_FLAG	WT，写同步标志 0：同步完成，此时可更改 LOAD/BGLOAD 1：正在同步，此时写 LOAD/BGLOAD 无效	0x0	RO
15:10	Res	保留	0x0	–
9	TCK_EN	LPTIM 计数时钟使能 0：LPTIM 计数时钟关闭 1：LPTIM 计数时钟使能 只有计数时钟使能后才能进行 LOAD/BGLOAD 的配置	0x0	R/W
8	INT_EN	中断使能控制，写 1 后使能中断	0x0	R/W
7	GATE_P	输入 GATE 的有效极性 默认高电平 GATE 有效，设置为 1 后低电平有效	0x0	R/W
6	GATE_EN	定时器门控 0：无门控 1：有门控	0x0	R/W
5:4	TCK_SEL	LPTIM 时钟选择 00：PCLK(读经过同步) 01：PCLK(无同步) 10：LXT	0x0	R/W

		11：LIRC		
3	TOG_EN	TOG 输出使能 0：TOG,TOGN 同时输出 0 1：TOG,TOGN 输出相位相反的信号。可供 BEEP 使用。	0x0	R/W
2	CT_SEL	计数器/定时器功能选择 0：定时器功能，定时器使用 TCK_SEL 选择的时钟进行计数。 1：计数器功能，计数器使用外部输入信号的下降沿进行计数。采样时钟使用 TCK_SEL 选择的时钟，外部输入信号频率要低于 1/2 采样时钟频率。	0x0	R/W
1	MODE	定时器工作模式 0：模式 1 无重载模式 16 位计数器/定时器 1：模式 2 自动重装载 16 位计数器/定时器	0x0	R/W
0	TIM_RUN	定时器运行控制位 0：定时器停止 1：定时器运行	0x0	R/W

16.4.3 LPTIM 立即重载寄存器(LPTIM_LOAD)

地址偏移：0x08
复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15:0	LOAD	即刻重载寄存器 与 TIMER 是否运行无关，与 MODE 无关. 写 LOAD 前需要读取 LPTIM_CR.WT_FLAG，当且仅当 WT_FLAG 为 0 时，才能写入数据。写 LOAD 寄存器完成后 WT_FLAG 会变低。 写该寄存器时会立即更新计数器的值。 读该寄存器时返回最新更新到 LOAD 或者 BGLOAD 的值	0x0	R/W

注：只有计数时钟使能后才能配置 LOAD/BGLOAD 寄存器。

16.4.4 LPTIM 中断寄存器(LPTIM_INTSR)

地址偏移：0x0C
复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	–
0	INTF	中断标志： 0：未发生中断； 1：发生溢出中断；	0x0	RO

16.4.5 LPTIM 中断寄存器(LPTIM_INTCLR)

地址偏移：0x10
复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	–
0	ICLR	写 1 清除中断标志，写 0 无效	0x0	WO

16.4.6 LPTIM 周期重载寄存器(LPTIM_BGLOAD)

地址偏移：0x14

复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15:0	BGLOAD	BACKGROUND 周期重载寄存器 写 BGLOAD 前需要读取 LPTIM_CR.WT_FLAG，当且仅当 WT_FLAG 为 0 时，才能写入数据。写 BGLOAD 寄存器完成后 WT_FLAG 会变低。 写该寄存器时，当计数器溢出后，设定值会更新到计数器。 读该寄存器时返回最新更新到 LOAD 或者 BGLOAD 的值	0x0	R/W

注：只有计数时钟使能后才能配置 LOAD/BGLOAD 寄存器。

17 可编程计数阵列(PCA)

17.1 PCA 简介

PCA(可编程计数器阵列 Programmable Counter Array)支持最多 5 个 16 位的捕获/比较模块。该定时/计数器可用作为一个通用的时钟计数/事件计数器的捕获/比较功能。PCA 的每个模块都可以进行独立编程，以提供输入捕捉、输出比较或脉冲宽度调制。另外模块 4 有额外的看门狗定时器模式。

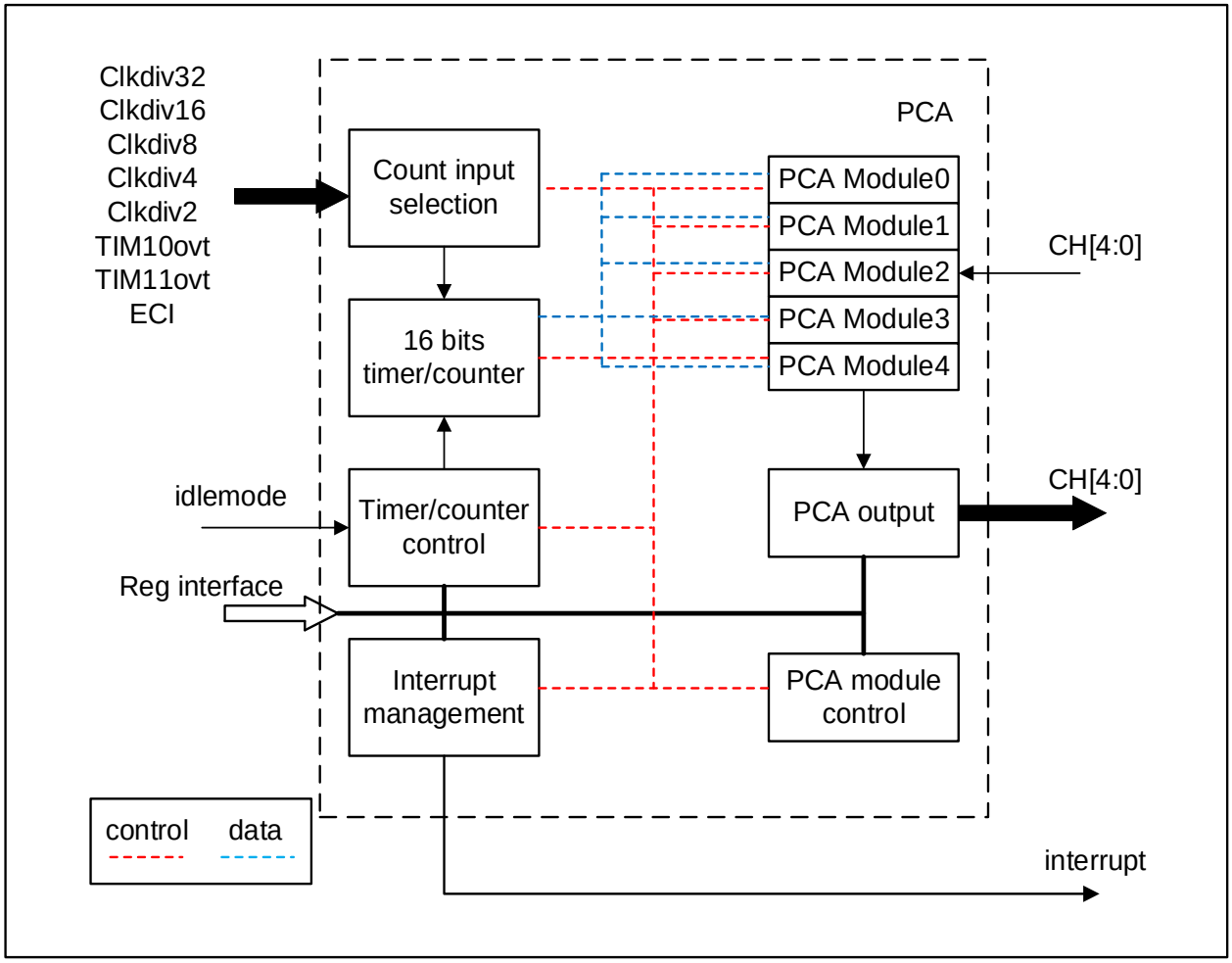


图 17-1 PCA 整体框图

17.2 PCA 功能描述

每个模块都可被配置为独立工作，有三种工作方式：边沿触发捕捉、输出比较、8 位脉宽调制。每个模块在系统控制器中都有属于自己的功能寄存器，这些寄存器用于配置模块的工作方式和与模块交换数据。每组比较/捕获模块是由一个比较/捕获寄存器组(CCAPx)，以及 1 个 16 位比较器，和各种逻辑门控制组成。寄存器组用来存储时间或次数，针对外部触发捕获条件，或内部触发比较条件。在 PWM 模式下，寄存器(CCAPxL)用来控制输出波形的占空比。每个模块都可以独立编程的操作在任何以下模式：

- 16 位捕获模式的上升沿，下降沿或任意沿触发
- 比较模式：16 位软件定时器，16 位高速输出或 8 位脉冲宽度调制
- 未启动

比较/捕获模块模式寄存器(CCAPMx)确定相应的工作模式。对于比较/捕获模块进行编程时，他们是基于共同的时间计数。定时器/计数器打开和关闭通过 CCON.CR 位即可控制 PCA 定时/计数器的运行。在一个比较/捕获模块捕获，软件定时器，高速输出，设置模块的比较/捕获标志(CCON.CCFx)，并产生 PCA 中断请求，如果相应的使能位在 CCAPMx 寄存器设置。CPU 可以在任何时候读写 CCAPx 寄存器。

17.2.1 PCA 定时/计数器

CNT 的这组特殊功能寄存器可用作为一个 16 位定时器/计数器。这是一个 16 位向上计数的计数器。如果 CMOD.CFIE 位被置“1”时，当 CNT 溢出时硬件自动设置 PCA 溢出标志(CCON.CF)并产生 PCA 中断请求。CMOD.CPS 三位选择八个信号输入到定时器/计数器。

- 系统时钟 PCLK 的 32 分频
- 系统时钟 PCLK 的 16 分频
- 系统时钟 PCLK 的 8 分频
- 系统时钟 PCLK 的 4 分频
- 系统时钟 PCLK 的 2 分频
- 定时器 0 的溢出。每次定时器 0 计数溢出后，CNT 就递增，这样提供了 PCA 的可变编程频率输入。
- 定时器 1 的溢出。每次定时器 1 计数溢出后，CNT 就递增，这样提供了 PCA 的可变编程频率输入。
- ECI。CPU 每过 4 个 PCLK 时钟周期就对 PCA ECI 进行采样，当每次采样结果从高变低时，CL 自动加 1，因此最高的 ECI 输入频率不能高于系统时钟 PCLK 的 1/8，以满足采样需求。

设置运行控制器(CCON.CR)启动 PCA 定时/计数器。当 CMOD.CIDL 置“1”后，PCA 定时器/计数器可以继续运行在空闲模式下。CPU 可以随时读取 CNT 的数值，但当计数启动后(CCON.CR=1)时，为了防止计数错误，CNT 是禁止写入的。

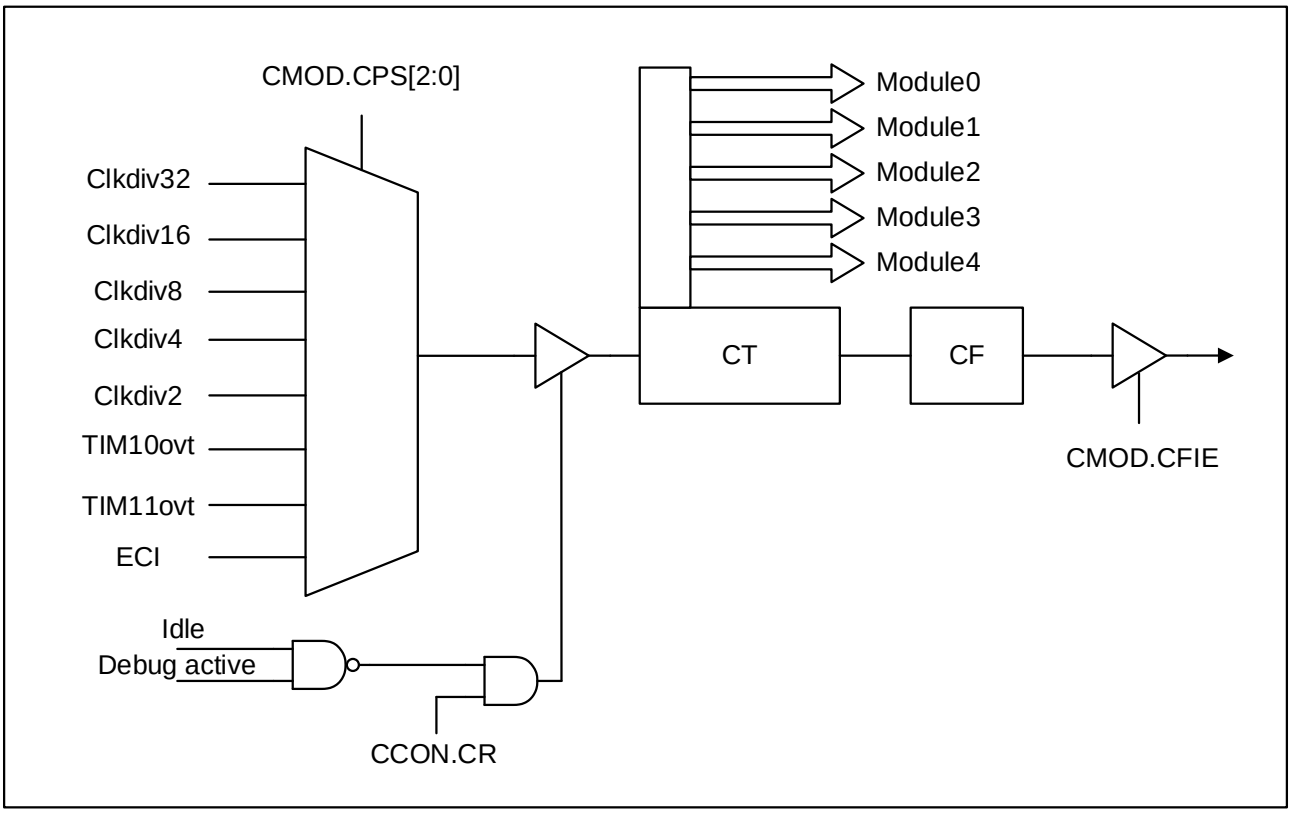


图 17-2 PCA 计数器框图

17.2.2 捕获功能

PCA 捕获模式提供了5 路 PCA 测量脉冲周期，脉冲宽度，占空比和相位差的功能。引脚上出现的电平跳变导致 PCA 捕捉 PCA 计数器/定时器的值并将其装入到对应模块的 16 位捕捉/比较寄存器(CCAPx)。CCAPMx.CAPP 以及 CCAPMx.CAPN 位用于选择触发捕捉的电平变化类型：低电平到高电平(正沿)、高电平到低电平(负沿)或任何变化(正沿或负沿)。当捕捉发生时，CCON 中的捕捉/比较标志(CCFn)被置为逻辑‘1’并产生一个中断请求(如果 CCF 中断被允许)。当 CPU 转向中断服务程序时，CCFn 位不能被硬件自动清除，用户软件写 INTCL 寄存器清除此标志位。如果 CCPMx.CAPP 以及 CCAPMx.CAPN 位都被设置为逻辑‘1’,可以通过直接读对应端口引脚的状态来确定本次捕捉是由上升沿触发还是由下降沿触发。分辨率等于定时器/计数器的时钟。输入信号必须在高电平或低电平期间至少保持 2 个时钟周期，以保证输入信号能够被硬件识别。

CPU 可以在任何时候读取或写入 CCAPx 的寄存器。捕获设置：

- 当需要在外部上升沿进行捕获，CCPMx.CAPP = “1”以及 CCAPMx.CAPN = “0”
- 当需要在外部下降沿进行捕获，CCPMx.CAPP = “0”以及 CCAPMx.CAPN = “1”
- 当需要在外部上升、下降沿进行捕获，CCPMx.CAPP = “1”以及 CCAPMx.CAPN = “1”

注意：

随后由同一模块的捕获值会覆盖现有捕获的值。为了保持捕获的值，在中断服务程序中将它保存在 RAM 里面，这个操作必须在下一次事件出现之前完成，否则就会丢失前面一次捕获采样值。

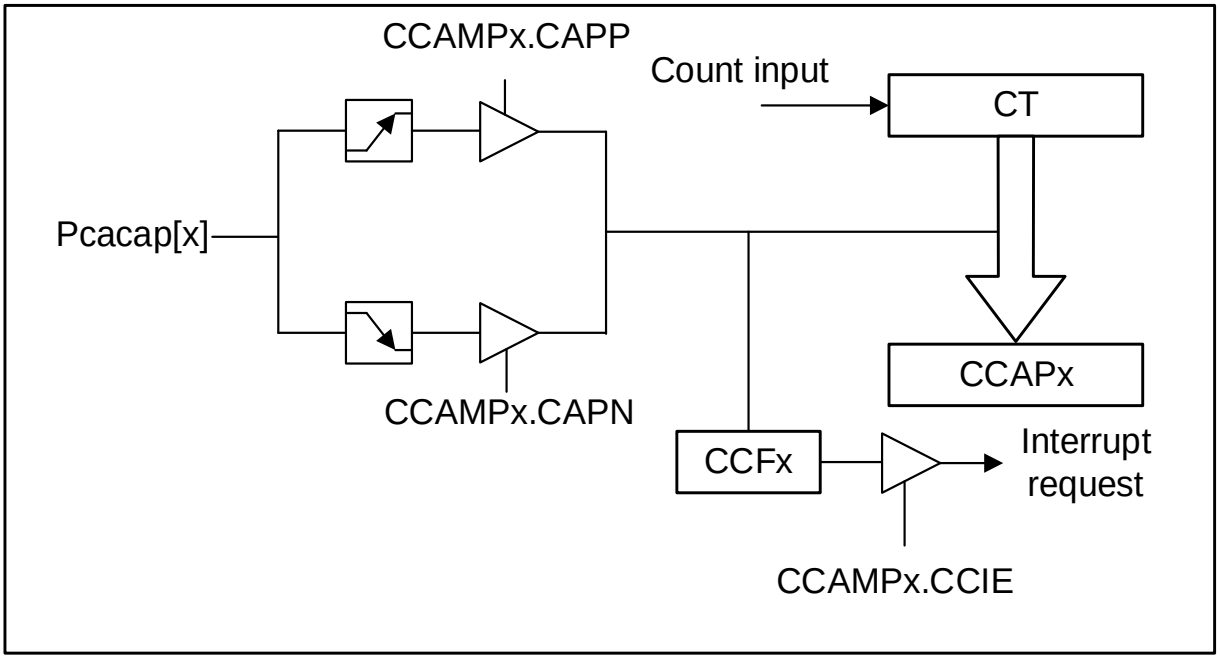


图 17-3 PCA 捕获功能框图

17.2.3 PCA 比较功能

比较功能提供如下功能，定时器，事件计数器，脉冲宽度调制。三种模式采用比较功能：16 位软件定时器模式，高速输出模式，PWM 模式。在前两个功能中，比较/捕获模块比较 16 位 PCA 定时器/计数器的值与预先加载到该模块的 CCAPx 寄存器中的 16 位值。在 PWM 模式下，PCA 模块不断地将 PCA 定时器/计数器低字节寄存器(CNT)与一个在 CCAPxL 模块寄存器8 位的值进行比较。每 4 个时钟周期比较一次，即与最快的 PCA 定时器/计数器的时钟速率相匹配。设置 CCAPMx.ECOM 位选择该模块的比较功能。若要正确使用在比较模式下的模块，请遵守以下的一般程序：

- 选择 PCA 模块的操作模式
- 选择 PCA 定时器/计数器的输入信号
- 比较值加载到模块的比较/捕获寄存器对
- 设置 PCA 定时器/计数器运行控制位
- 匹配后产生中断，清除模块的比较/捕获标志

17.2.3.1 16 位软件计数器模式

要设定一个比较/捕获模块的 16 位软件定时器模式下，需要设置 CCAPMx.ECOM 和 CCAPMx.MAT 位。一旦在 PCA 定时器/计数器和比较/捕获的寄存器(CCAPx)之间发生了匹配，这将设置模块的比较/捕获标志(CCON.CCFx)。这将产生一个中断请求，如果相应的中断使能位(CCAPMx.CCIE)设置。由于硬件不清除的比较/捕获标志中断处理时，用户必须清除软件标志。在中断服务程序中，一个新的 16 位比较值可以被写入比较/捕获的寄存器(CCAPx)。注意：在更新这些寄存器时，为了防止无效的匹配发生，用户软件应该先写 CCAPxL，后写 CCAPxH。一旦写如 CCAPxL 就会清除禁用比较功能 ECOMx 位，而

写入 CCAPxH 会同时设置的 ECOMx 位，重新启用比较功能。即当向 PCA0 的捕捉/比较寄存器写入一个 16 位数值时，应先写低字节。

17.2.3.2 高速输出模式

在高速输出方式，每当 PCA 计数器内的值与模块的 16 位捕捉/比较寄存器(CCAPx)发生匹配时，模块 PCA 的 CAP/CMP[x]引脚上的逻辑电平将发生变化。这可以提供比切换 IO 输出有更高精度，因为这个高速输出不会被中断响应而影响输出频率，靠 CPU 来切换 IO 输出的话，功耗、精度都有所欠缺。

要设定一个比较/捕获模块的高速输出模式，设置 CCAPMx.ECOM，CCAPMx.MAT 和 CCAPMx.TOG 位。PCA 定时器/计数器和比较/捕获的寄存器(CCAPx)之间的匹配切换 PCA 的 CAP/CMP[x]信号，并设置模块的比较/捕获标志(CCON.CCFx)。通过软件设置或清除 PCA 的 CAP/CMP[x]信号，用户可以选择匹配切换信号从低到高或高到低。

用户也可以选择产生一个中断请求，通过设置相应的中断使能位(CCAPMx.CCIE)当匹配发生时，即可产生中断请求。由于硬件无法清除的比较/捕获标志中断，用户必须在软件中清除这个标志位。如果用户在中断程序中不去改变比较/捕获寄存器，PCA 并重新计数比较值，如相匹配则发生下一次翻转。在中断服务程序中，一个新的 16 位比较值可以被写入比较/捕获的寄存器(CCAPx)。

注意：为了防止无效的匹配，而更新这些寄存器，用户软件应该写 CCAPxL 的首先然后 CCAPxH。写到 CCAPxL 清除禁用比较功能 ECOM 位，而写到 CCAPxH 设置的 ECOM 位，重新启用比较功能。

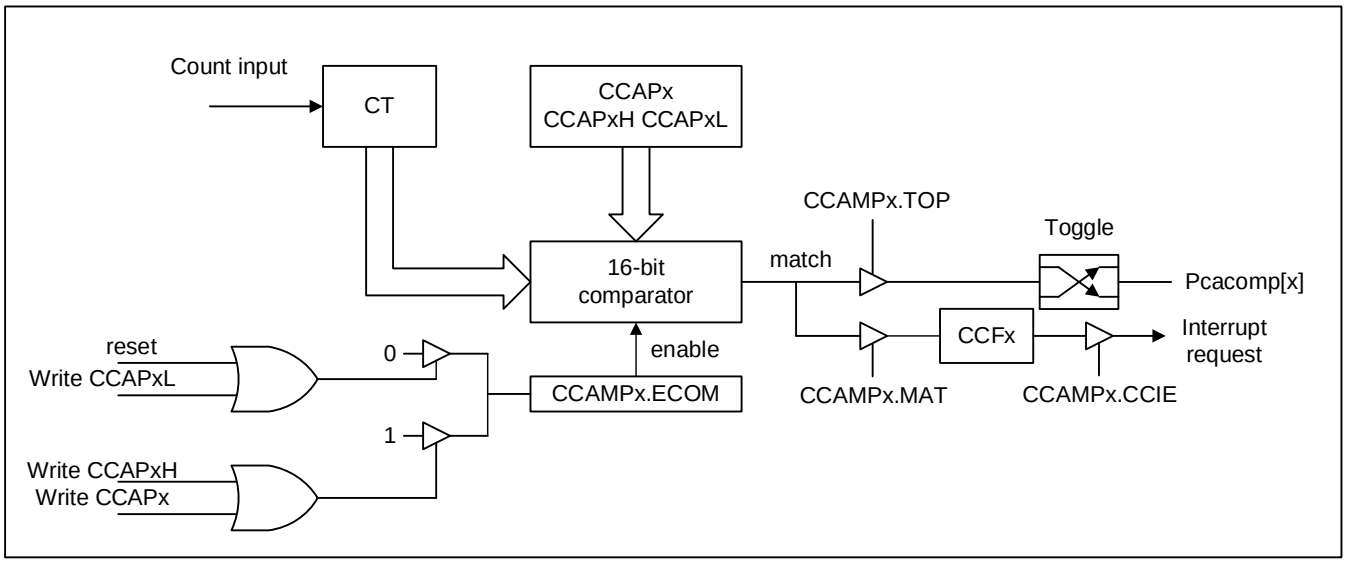


图 17-4 PCA 比较功能框图

17.2.3.3 PCA 8 位脉宽调制功能

脉宽调制是一种使用程序来控制波形占空比，周期，相位的技术。5 个 PCA 模块都可以被独立地用于在对应 PCA 的 CAP/CMP[x]引脚产生脉宽调制(PWM)输出，脉冲宽度为 8 位分辨率。PWM 输出的频率取决于 PCA 计数器/定时器的时基。使用模块的捕捉/比较寄存器 CCAPxL 来改变 PWM 输出信号的占空比。当 PCA 计数器/定时器的低字节(CL)与 CCAPxL 中的值相等时，PCA 的 CAP/CMP[x]引脚上的输出被置“1”；当 CL 中的计数值溢出时，PCA 的 CAP/CMP[x]输出被复位“0”。当计数器/定时器的低字节 CL 溢出时(从 0xFF 到 0x00)，保存在 CCAPxH 中的值被自动装入到 CCAPxL，不需软件干预。

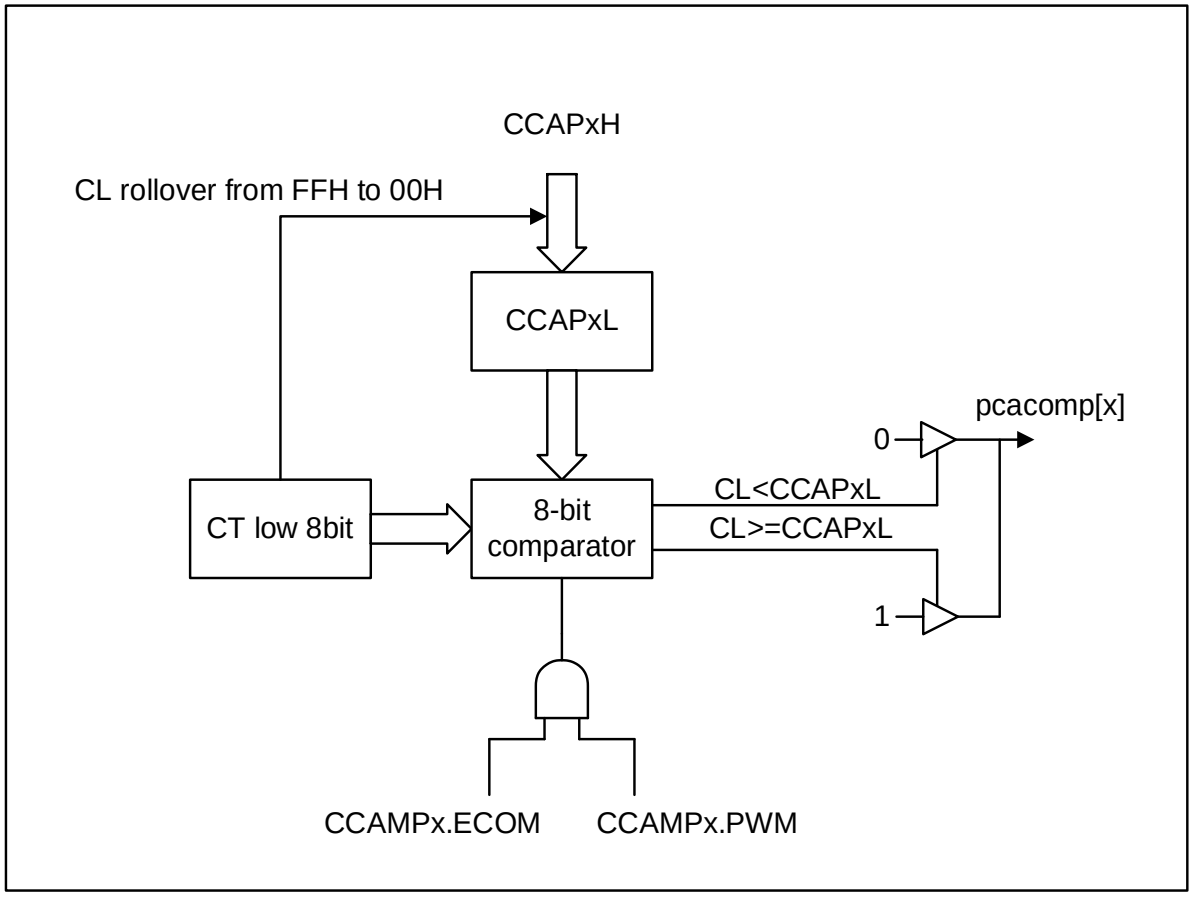


图 17-5 PCA PWM 功能框图

在这种模式下，PCA 定时器/计数器(CL)的低字节中的值是不断在低字节比较/捕获寄存器(CCAPxL)的值相比。当 CL < CCAPxL，输出波形为低。当两者匹配时(CL = CCAPxL)，输出波形去到高，直到 CL 溢出从 FFH 到 00H，结束期间仍然很高。在溢出时，在 CCAPxH 的值自动装载到 CCAPxL 内，一个新的周期的开始。

在 CCAPxL 的值决定当前波形的占空比。在 CCAPxH 的值确定下一个波形的占空比。改变 CCAPxL 中的值即可更改的脉冲宽度调制。正如图所示，8 位值在 CCAPxL 可以从 0(100％占空比)，到 255(0.4％占空比)。要改变 CCAPxL 值而不会产生毛刺，需要在高字节寄存器(CCAPxH)写入一个新值。当 CL 超过 0xFF 滚动到 0x00，这个值是由硬件自动加载到 CCAPxL。

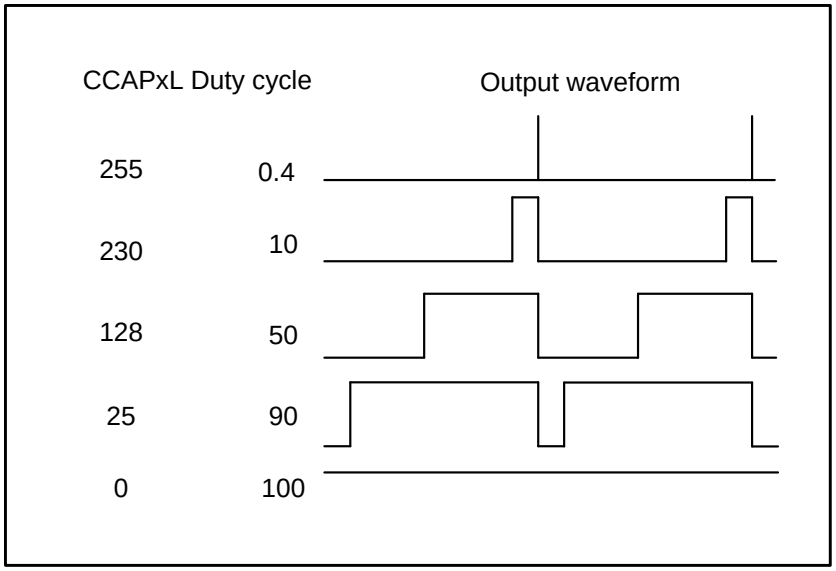


图 17-6 PCA PWM 输出波形

要设定一个比较/捕获模块在 PWM 模式下，需要设置 CCAPMx.ECOM 和 CCAPMx.PWM 位。另外 PCA 定时器/计数器由编程 CMOD.CSP 可以选择输入计数信号频率。在 CCAPxL 输入一个 8 位的值指定第一个 PWM 波形的占空比。在 CCAPxH 输入一个 8 位的值会指定第二个 PWM 波形的占空比。设置定时器/计数器运行控制位(CCON.CR)启动 PCA 定时器/计数器。

表 17-1 PCA 比较/捕获功能模块设置

ECOM	CAPP	CAPN	MAT	TOG	PWM	工作方式
X	1	0	0	0	0	用 CCPn 的正沿触发捕捉
X	0	1	0	0	0	用 CCPn 的负沿触发捕捉
X	1	1	0	0	0	用 CCPn 的跳变触发捕捉
1	0	0	1	0	0	软件定时器
1	0	0	1	1	0	高速输出
1	0	0	0	0	1	8 位脉冲宽度调制器

17.3 PCA 模块与其他模块互连及控制

17.3.1 ECI 互连

ECI 输入可以是外部通过 IO MUX 选择不同的输入端口，也可以是内部 VC 的比较的滤波输出。VC 输出控制寄存器在 VC 控制模块。

17.3.2 PCACAP0

通道 0 的捕获输入可以是：

- 外部的 IOMUX 的输入端口，外部 UART 的 RX 的 MUX 输入
- 内部的 VC 的比较滤波后的输出

  UART 选择控制在 PCA 捕获通道控制寄存器 SYSCON_PCACR 中，VC 输出控制寄存器在 VC 控制模块。

17.3.3 PCACAP[4:1]

通道 1，2，3，4 的捕获输入可以是：

- 外部的 IOMUX 的输入端口,外部 UART 的 RX 的 MUX 输入

UART 选择控制在 PCA 捕获通道控制寄存器 SYSCON_PCACR 中。

17.4 PCA 寄存器列表

基地址 0x40001400

表 17-2 PCA 寄存器列表和复位值

偏移地址	名称	描述	复位值
0x00	PCA_CR	PCA 控制寄存器	0x00000000
0x04	PCA_MOD	PCA 模式寄存器	0x00000000
0x08	PCA_CNT	PCA 计数寄存器	0x00000000
0x0C	PCA_INTCLR	PCA 中断清除寄存器	0x00000000
0x00	PCA_CCAPM0	PCA 比较/捕获模块 0 模式寄存器	0x00000000
0x14	PCA_CCAPM1	PCA 比较/捕获模块 1 模式寄存器	0x00000000
0x18	PCA_CCAPM2	PCA 比较/捕获模块 2 模式寄存器	0x00000000
0x1C	PCA_CCAPM3	PCA 比较/捕获模块 3 模式寄存器	0x00000000
0x20	PCA_CCAPM4	PCA 比较/捕获模块 4 模式寄存器	0x00000000
0x30	PCA_CCAP0L	PCA 比较/捕获模块 0 低 8 位寄存器	0x00000000
0x34	PCA_CCAP0H	PCA 比较/捕获模块 0 高 8 位寄存器	0x00000000
0x38	PCA_CCAP1L	PCA 比较/捕获模块 1 低 8 位寄存器	0x00000000
0x3C	PCA_CCAP1H	PCA 比较/捕获模块 1 高 8 位寄存器	0x00000000
0x40	PCA_CCAP2L	PCA 比较/捕获模块 2 低 8 位寄存器	0x00000000
0x44	PCA_CCAP2H	PCA 比较/捕获模块 2 高 8 位寄存器	0x00000000
0x48	PCA_CCAP3L	PCA 比较/捕获模块 3 低 8 位寄存器	0x00000000
0x4C	PCA_CCAP3H	PCA 比较/捕获模块 3 高 8 位寄存器	0x00000000
0x50	PCA_CCAP4L	PCA 比较/捕获模块 4 低 8 位寄存器	0x00000000
0x54	PCA_CCAP4H	PCA 比较/捕获模块 4 高 8 位寄存器	0x00000000
0x58	PCA_CCAPO	PCA PWM 与高速输出标志寄存器	0x00000000
0x5C	PCA_POCR	PCA 端子输出控制寄存器	0x00000000
0x60	PCA_CCAP0	PCA 比较/捕获模块 0 的 16 位寄存器	0x00000000
0x64	PCA_CCAP1	PCA 比较/捕获模块 1 的 16 位寄存器	0x00000000
0x68	PCA_CCAP2	PCA 比较/捕获模块 2 的 16 位寄存器	0x00000000
0x6C	PCA_CCAP3	PCA 比较/捕获模块 3 的 16 位寄存器	0x00000000
0x70	PCA_CCAP4	PCA 比较/捕获模块 4 的 16 位寄存器	0x00000000

17.5 寄存器说明

17.5.1 控制寄存器(PCA_CR)

偏移地址：0x00
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								CF	CR	保留	CCF4	CCF3	CCF2	CCF1	CCF0
								RO	R/W		RO	RO	RO	RO	RO

位	标记	功能描述	复位值	读写
31:8	Res	保留位	0x0	-
7	CF	PCA 计数器溢出标志: 0：无溢出； 1：发生计数器溢出； 当 PCA 计数溢出时，CF 由硬件置位; 当 CMOD 寄存器的 CFIE 位为 1，CF 标志可以产生中断；	0x0	RO
6	CR	PCA 计数器运行控制位 0：关闭 PCA 计数器计数 1：启动 PCA 计数器计数	0x0	R/W
5	Res	保留位	0x0	-
4	CCF4	PCA 计数器模块 4 比较/捕获标志位： 当出现匹配或捕获时，该位由硬件置位； 当 CCAPM4.CCIE 置位时，这个标志位会产生一个 PCA 中断； 0：无匹配或捕获； 1：匹配或捕获发生；	0x0	RO
3	CCF3	PCA 计数器模块 3 比较/捕获标志位： 当出现匹配或捕获时，该位由硬件置位； 当 CCAPM3.CCIE 置位时，这个标志位会产生一个 PCA 中断； 0：无匹配或捕获； 1：匹配或捕获发生；	0x0	RO
2	CCF2	PCA 计数器模块 2 比较/捕获标志位： 当出现匹配或捕获时，该位由硬件置位； 当 CCAPM2.CCIE 置位时，这个标志位会产生一个 PCA 中断； 0：无匹配或捕获； 1：匹配或捕获发生；	0x0	RO
1	CCF1	PCA 计数器模块 1 比较/捕获标志位： 当出现匹配或捕获时，该位由硬件置位； 当 CCAPM1.CCIE 置位时，这个标志位会产生一个 PCA 中断； 0：无匹配或捕获； 1：匹配或捕获发生；	0x0	RO
0	CCF0	PCA 计数器模块 0 比较/捕获标志位： 当出现匹配或捕获时，该位由硬件置位； 当 CCAPM0.CCIE 置位时，这个标志位会产生一个 PCA 中断； 0：无匹配或捕获；	0x0	RO

		1：匹配或捕获发生；		

17.5.2 模式寄存器(PCA_MOD)

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
保留															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								CIDL	保留			CPS			CFIE
								R/W				R/W			R/W

位	标记	功能描述	复位值	读写
31:8	Res	保留位	0x0	-
7	CIDL	空闲模式 IDLE 下，PCA 是否停止工作 0：休眠模式(Sleep)下，PCA 继续工作 1：休眠模式(Sleep)下，PCA 停止工作	0x0	R/W
6:4	Res	保留位	0x0	-
3:1	CPS	时钟分频选择及时钟源选择 000：PCLK/32 001：PCLK/16 010：PCLK/8 011：PCLK/4 100：PCLK/2 101：TIM10 溢出 110：TIM11 溢出 111：ECI 外部时钟，时钟 PCLK 四分频采样	0x0	R/W
0	CFIE	PCA 计数器中断使能控制信号 0：关闭中断 1：使能中断	0x0	R/W

17.5.3 计数寄存器(PCA_CNT)

偏移地址：0x08
复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:16	Res	保留位	0x0	-
15:0	CNT	定时器计数器的值 只有在 PCA 停止状态，CNT 才可以写入，否则写入无效	0x0	R/W

17.5.4 中断清除寄存器(PCA_INTCLR)

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
保留															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								CF	保留		CCF4	CCF3	CCF2	CCF1	CCF0
								WO			WO	WO	WO	WO	WO

位	标记	功能描述	复位值	读写
31:8	Res	保留位	0x0	-
7	CF	PCA 计数器溢出标志清除: 0：写 0 无效； 1：软件写 1 清零对应的标志;	0x0	WO
6:5	Res	保留位	0x0	-
4	CCF4	PCA 计数器模块 4 比较/捕获标志位清除: 0：写 0 无效； 1：软件写 1 清零对应的标志;	0x0	WO
3	CCF3	PCA 计数器模块 3 比较/捕获标志位清除: 0：写 0 无效； 1：软件写 1 清零对应的标志;	0x0	WO
2	CCF2	PCA 计数器模块 2 比较/捕获标志位清除: 0：写 0 无效； 1：软件写 1 清零对应的标志;	0x0	WO
1	CCF1	PCA 计数器模块 1 比较/捕获标志位清除: 0：写 0 无效； 1：软件写 1 清零对应的标志;	0x0	WO
0	CCF0	PCA 计数器模块 0 比较/捕获标志位清除: 0：写 0 无效； 1：软件写 1 清零对应的标志;	0x0	WO

17.5.5 比较捕获模式寄存器(PCA_CCAPM0~4)

偏移地址：CCAPM0：0x10；CCAPM1：0x14；
CCAPM2：0x18；CCAPM3：0x1C；
CCAPM4：0x20；

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留									ECOM	CAPP	CAPN	MAT	TOG	PWM	CCIE
									R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:7	Res	保留位	0x0	-
6	ECOM	允许比较器功能控制位： 0：禁止比较器功能； 1：允许比较器功能； 当 PCA 用于软件计数器，高速输出，PWM 模式，要置位 ECOM; 当写 CCAMPHx 或 CCAMPx 寄存器会自动置位 ECOM； 当写 CCAMPLx 寄存器会自动清除 ECOM 位；	0x0	R/W
5	CAPP	正沿捕获控制位： 0：禁止上升沿捕获 1：允许上升沿捕获；	0x0	R/W
4	CAPN	负沿捕获控制位 0：禁止下降沿捕获 1：允许下降沿捕获；	0x0	R/W
3	MAT	允许匹配控制位： 0：禁止匹配功能； 1：PCA 计数值与模块的比较/捕获寄存器的值一旦匹配，将置位 CCON 寄存的中断标志 CCFx(x=0-4)；	0x0	R/W
2	TOG	翻转控制位： 0：禁止翻转功能 1：工作在 PCA 高速输出模式，PCA 计数器的值与模块的比较/捕获寄存器的值一旦匹配，CCPx 引脚翻转	0x0	R/W
1	PWM	脉宽调制控制位： 0：禁止 PWM 脉宽调制功能 1：允许 CCPx 引脚作为 PWM 输出 只有 CCAPMx=100_0010 时，PWM 功能才有效	0x0	R/W
0	CCIE	PCA 使能中断： 0：PCA 比较/捕获功能中断禁止 1：使能比较/捕获中断	0x0	R/W

17.5.6 比较捕获数据寄存器低 8 位(PCA_CCAP0~4L)

偏移地址：CCAP0L：0x30；CCAP1L：0x38；
CCAP2L：0x40；CCAP3L：0x48；
CCAP4L：0x50；

复位值：0x0000 0000

位	标记	功能描述	复位值	读写
31:8	Res	保留位	0x0	-
7:0	CCAPxL	比较/捕获模式低 8 位寄存器: 当 PCA 模式用于比较/捕获模式时，用于保存 16 位捕获计数值的低 8 位；写 CCAPxH 寄存器会自动清除寄存器 CCAPMx 的 ECOM 位。 当 PCA 模式用于 PWM 模式时，用于控制输出占空比比较寄存器，在 PWM 模式，计数器的低 8 位的值小于 CCAPx 的值 PWM 输出低电平，否则 PWM 输出高电平。	0x0	R/W

17.5.7 比较捕获数据寄存器高 8 位(PCA_CCAP0~4H)

偏移地址：CCAP0H：0x34；CCAP1H：0x3C；
CCAP2H：0x44；CCAP3H：0x4C；
CCAP4H：0x54；

复位值：0x0000 0000

位	标记	功能描述	复位值	读写
31:8	Res	保留位	0x0	-
7:0	CCAPxH	比较/捕获模式高 8 位寄存器: 当 PCA 模式用于比较/捕获模式时,用于保存 16 位捕获计数值的高 8 位,写 CCAPxH 寄存器会自动置位寄存器 CCAPMx 的 ECOM 位。 当 PCA 模式用于 PWM 模式时，用于控制输出占空比装载寄存器，在计数器低 8 位溢出时，装载寄存器会自动更新到 PWM 比较寄存器	0x0	R/W

17.5.8 比较高速输出标志寄存器(PCA_CCAPO)

偏移地址：0x58
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留											CCAPO4	CCAPO3	CCAPO2	CCAPO1	CCAPO0
											R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:5	Res	保留位	0x0	-
4	CCAPO4	高速模式比较模块 4 的输出值 0：输出 0 1：输出 1	0x0	R/W
3	CCAPO3	高速模式比较模块 3 的输出值 0：输出 0 1：输出 1	0x0	R/W
2	CCAPO2	高速模式比较模块 2 的输出值 0：输出 0 1：输出 1	0x0	R/W
1	CCAPO1	高速模式比较模块 1 的输出值 0：输出 0 1：输出 1	0x0	R/W
0	CCAPO0	高速模式比较模块 0 的输出值 0：输出 0 1：输出 1	0x0	R/W

17.5.9 端子输出控制寄存器(PCA_POCR)

偏移地址：0x5C
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留			POINV4	POINV3	POINV2	POINV1	POINV0	保留			POE4	POE3	POE2	POE1	POE0
			R/W	R/W	R/W	R/W	R/W				R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:13	Res	保留位	0x0	-
12	POINV4	比较通道 4 的输出极性反转 0：禁止比较通道 4 输出极性反转 1：使能比较通道 4 输出极性反转	0x0	R/W
11	POINV3	比较通道 3 的输出极性反转 0：禁止比较通道 3 输出极性反转 1：使能比较通道 3 输出极性反转	0x0	R/W
10	POINV2	比较通道 2 的输出极性反转	0x0	R/W

		0：禁止比较通道 2 输出极性反转 1：使能比较通道 2 输出极性反转		
9	POINV1	比较通道 1 的输出极性反转 0：禁止比较通道 1 输出极性反转 1：使能比较通道 1 输出极性反转	0x0	R/W
8	POINV0	比较通道 0 的输出极性反转 0：禁止比较通道 0 输出极性反转 1：使能比较通道 0 输出极性反转	0x0	R/W
7:5	Res	保留位	0x0	-
4	POE4	比较通道 4 的输出使能 0：输出禁止 1：输出使能	0x0	R/W
3	POE3	比较通道 3 的输出使能 0：输出禁止 1：输出使能	0x0	R/W
2	POE2	比较通道 2 的输出使能 0：输出禁止 1：输出使能	0x0	R/W
1	POE1	比较通道 1 的输出使能 0：输出禁止 1：输出使能	0x0	R/W
0	POE0	比较通道 0 的输出使能 0：输出禁止 1：输出使能	0x0	R/W

17.5.10 比较捕获 16 寄存器(PCA_CCAP0~4)

偏移地址：CCAP0：0x60；CCAP1：0x64；
CCAP2：0x68；CCAP3：0x6C；
CCAP4：0x70；

复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留位	0x0	-
15:0	CCAPx	比较/捕获模式 16 位寄存器: 当 PCA 用于比较/捕获模式时，用于保存 16 位捕获计数值；写 CCAPx 寄存器会置位寄存器 CCAPMx 的 ECOM 位。写 CCAPX 寄存器相当于写 CCAPxL 及 CCAPxH 这两个 8 位寄存器。在比较/捕获模式下可以直接读写这个寄存器，在 PWM 模式下，使用 CCAPxL 及 CCAPxH 寄存器。	0x0	R/W

18 自唤醒定时器(AWK)

CPS32K11x 有一个专用的自唤醒定时器(AWK)，为芯片在低功耗模式下提供一个唤醒事件基准。AWK 只有在 Sleep 或 Deep Sleep 模式保持计数。当 AWK 用作唤醒定时器时，AWK 要在进入省电模式之前开启。AWK 可以配置片内 38.4KHz 时钟源 LIRC、外部 32.768KHz 时钟源 LXT、以及 HXT 分频后的时钟。注意系统时钟频率必须大于 AWK 时钟两倍以上。如果 AWK 开始计数，在设备进入 Sleep 模式时，选择的时钟源会也要保持工作。注意选择的 AWK 时钟源不会连同 AWK 的配置自动使能，用户应该手动使能选择的时钟源并等待它稳定来确保操作的成功。

AWK 配备了一个简单的 8 位自动重载向上计数定时器。它的预分频可选择从 1/2 到 1/65536，通过 DIVSEL(AWK_CR)来设置。用户填写重装载值到 AWK_RLOAD 寄存器来决定它的溢出速率。AWKEN(AWK_CR.4)置位后，当 CPU 进入 Sleep 模式时装载 AWK_RLOAD 寄存器的值到内部 8 位计数器并开始计数。当计数器溢出，AWUF(AWK_SR.0)置为 1，唤醒 CPU。

注意：未进入 Sleep 模式时，AWK 不进行计数。

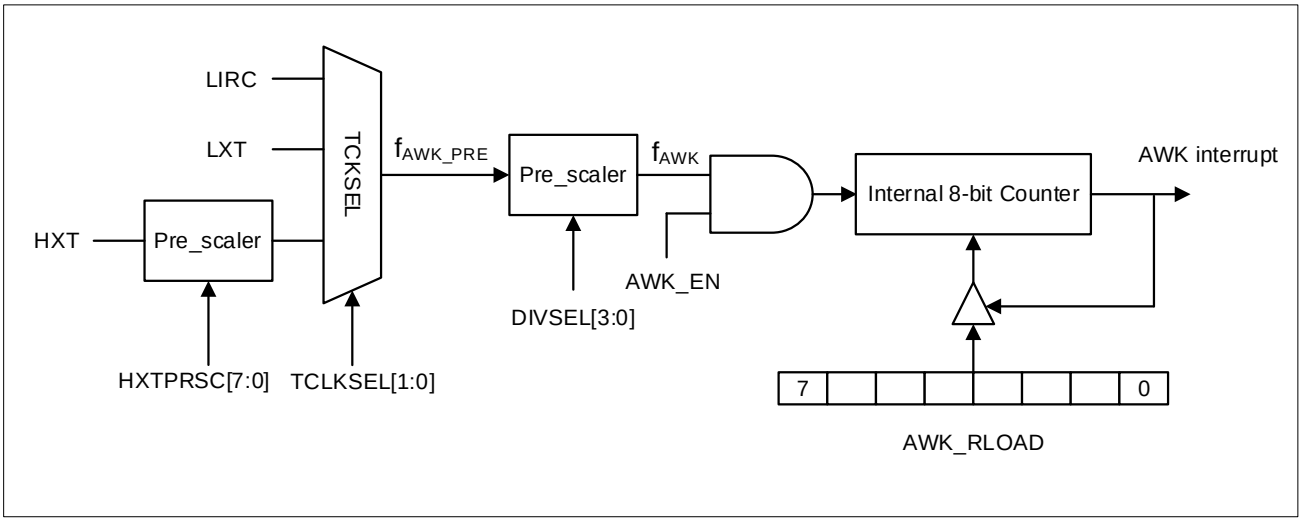


图 18-1 自唤醒定时器结构图

18.1 寄存器列表

基地址：0x40002800

表 18-1 AWK 寄存器列表和复位值

偏移地址	名称	描述	复位值
0x000	AWK_CR	自唤醒定时器控制寄存器	0x0000BC00
0x004	AWK_RLOAD	自唤醒定时器重装载数据寄存器	0x00000000
0x008	AWK_SR	自唤醒定时器状态寄存器	0x00000000
0x00C	AWK_INTCLR	自唤醒中断清除寄存器	0x00000000

18.2 寄存器说明

18.2.1 自唤醒定时器控制寄存器(AWK_CR)

地址偏移：0x00
复位值：0x0000BC00

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HXTPRSC								保留	TCLKSEL		AWK EN	DIVSEL			
R/W									R/W		R/W	R/W			

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15:8	HXTPRSC	HXT 时钟分频系数: $F_{HXT}/(N+1)$	0xBC	R/W
7	Res	保留	0x0	–
6:5	TCLKSEL	AWK 定时器时钟源选择 00：停止 01：LIRC 时钟 10：HXT 分频后的时钟 11：LXT 时钟	0x0	R/W
4	AWKEN	AWK 使能 1：使能 0：不使能	0x0	R/W
3:0	DIVSEL	定时器用时钟源选择位 0000：$F_{AWK\_PRE}/2^{\wedge 1}$ 0001：$F_{AWK\_PRE}/2^{\wedge 2}$ 0010：$F_{AWK+PRE}/2^{\wedge 3}$ … 1111：$F_{AWK\_PRE}/2^{\wedge 16}$	0x0	R/W

18.2.2 自唤醒定时器重装载数据寄存器(AWK_RLOAD)

地址偏移：0x04
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								RLDVAL							
								R/W							

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7:0	RLDVAL	计数器	0x0	R/W

18.2.3 自唤醒定时器状态寄存器(AWK_SR)

地址偏移：0x08
复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	–
0	AWUF	自动唤醒发生，硬件置 1，软件清零 0：未发生自动唤醒 1：定时器溢出，自动唤醒发生	0x0	RO

18.2.4 自唤醒中断清除寄存器(AWK_INTCLR)

地址偏移：0x0C
复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	–
0	INTCLR	自动唤醒中断清除 0：无作用 1：清除自动唤醒中断	0x0	WO

19 独立看门狗(IWDG)

19.1 概述

IWDG 的作用是为了防止软件系统在异常情况下，程序执行错误，导致系统异常工作或崩溃；而 IWDG 复位可以帮助系统自动恢复。工作原理是软件系统出错时，在固定的时间(这个时间可以配置)产生一个复位或者中断，让程序重新执行或者按照中断服务程序执行，而不至于系统崩溃。从而增加了软件系统的安全性能。

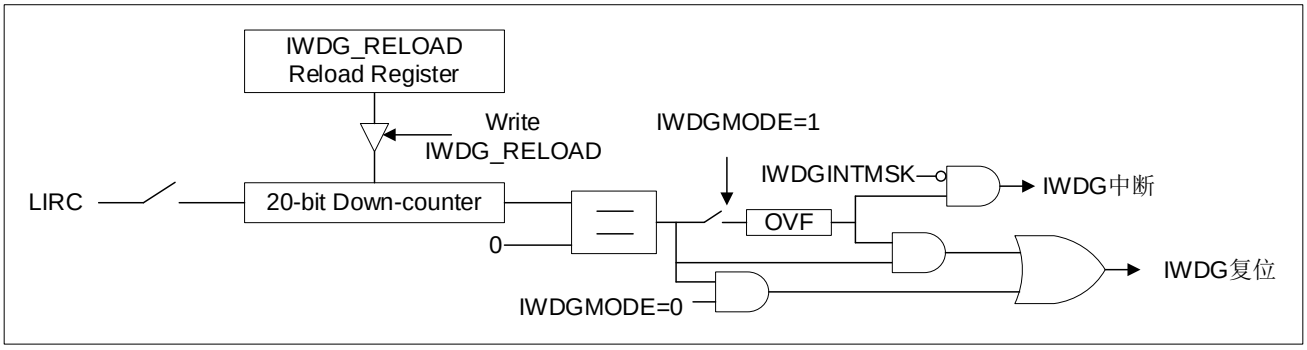


图 19-1 IWDG 整体框图

19.2 IWDG 的功能

- IWDG 模块是一个 20 位的向下计数的计数器，每一个计数时钟 LIRC 计数递减一次，计数时间可配置为 26us-27s；
- 内部低速时钟计数，IWDG 会硬件开启 LIRC 时钟；
- 当计数溢出时，支持中断和复位两种种方式；
- 可配置的计数的溢出时间；
- 启动并清零 IWDG 具有操作序列要求，增加安全性能；
- 部分寄存器写保护功能，防止程意外操作；

19.2.1 超时周期

看门狗超时周期由计数器数值决定，下表列出了它们的数值

表 19-1 看门狗超时周期(假定计数器时钟为 3KHz)

IWDG_RLOAD 寄存器配置值	超时周期
0x00000	26us
…	…
0x003FF	26.6ms
…	…
0xFFFFF	27s

19.2.2 IWDG 溢出后产生中断

在本模式下，IWDG 将按所设定的时间周期性的产生中断，在中断服务程序中需要清除 IWDG 溢出标志。

配置方法如下所示：

1. 写 0x55AA6699 到 IWDG_UNLOCK 解除 IWDG 的寄存器写保护，如果 IWDG_UNLOCK.IWDGREN 为 1 可省略本步骤。
2. 配置 IWDG_CFGR.IWDGMODE 为 1，选择中断方式。
3. 选择配置 IWDG_CFGR.IWDGINTMSK 为 0，则 CPU 响应 IWDG 的中断信号；配置为 1，则中断被屏蔽，CPU 只能通过读取 IWDG_SR.IWDGOVF 来判断是否计数溢出。
4. 配置 IWDG_RLOAD 寄存器。选择 IWDG 计数溢出时间。
5. 写非 0x55AA6699 到 IWDG_UNLOCK 开启 IWDG 的寄存器写保护。
6. 写 0x55 到 IWDG_CMDCR，启动 IWDG。
7. 如果中断产生在中断服务程序中先清除 IWDG 的寄存器保护，然后对 IWDG_INTCLR 写 1 清除中断标记。

19.2.3 IWDG 溢出后产生复位

在本模式下，IWDG 计数器溢出后会产生 Reset 信号，该信号会复位 MCU。用户需要在 IWDG 溢出前重装载 IWDG 计数器，从而避免产生 IWDG 复位。

配置方法如下：

1. 写 0x55AA6699 到 IWDG_UNLOCK 解除 IWDG 的寄存器写保护，如果 IWDG_UNLOCK.IWDGREN 为 1 可省略本步骤。
2. 配置 IWDG_CFGR.IWDGMODE 为 0，选择复位方式。
3. 配置 IWDG_RLOAD 寄存器，选择 IWDG 计数溢出时间。
4. 写非 0x55AA6699 到 IWDG_UNLOCK 开启 IWDG 的寄存器写保护。
5. 写 0x55 到 IWDG_CMDCR，启动 IWDG。
6. 在计数溢出前写 0xAA 到 IWDG_CMDCR，刷新 IWDG 计数器。

19.3 寄存器列表

基地址：0x40002400

表 19-1 IWDG 寄存器列表和复位值

偏移地址	名称	描述	默认值
0x00	IWDG_CMDCR	IWDG 控制命令寄存器	0x00000000
0x04	IWDG_CFGR	IWDG 配置寄存器	0x00000000
0x08	IWDG_RLOAD	IWDG 计数器重装载寄存器	0x000FFFFF
0x0c	IWDG_CNTVAL	IWDG 计数器值	0x000FFFFF
0x10	IWDG_SR	IWDG 中断状态寄存器	0x00000000
0x14	IWDG_INTCLR	IWDG 中断清除寄存器	0x00000000
0x18	IWDG_UNLOCK	IWDG 寄存器访问保护	0x00000000

19.4 寄存器说明

19.4.1 IWDG 控制命令寄存器(IWDG_CMDCR)

偏移地址：0x00
复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7:0	CMD	0x55：IWDG 启动命令 0xaa：IWDG 重装载刷新命令	0x0	WO

注意：在 IWDG 运行时才能写重装载命令

19.4.2 IWDG 配置寄存器(IWDG_CFGR)

地址偏移：0x04
复位值：0x00000000

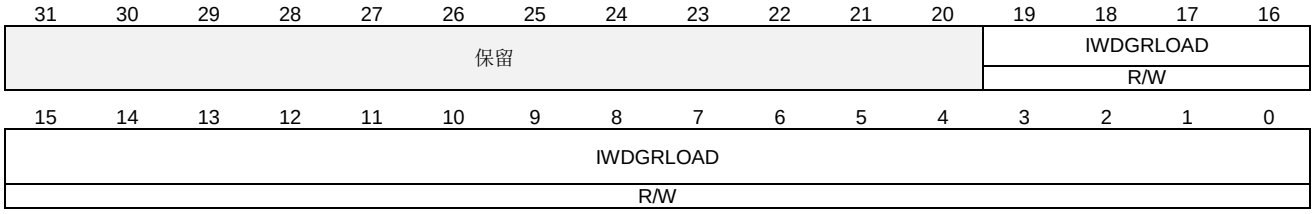
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留													IWDGRUNF	IWDGINTMS	IWDGMODE
													RO	R/W	R/W

位	标记	功能描述	复位值	读写
31:3	Res	保留	0x0	–
2	IWDGRUNF	IWDG 运行标志 0：IWDG 停止 1：IWDG 运行(即在计数))	0x0	RO
1	IWDGINTMSK	IWDG 中断屏蔽 0：中断不屏蔽(通知给 CPU) 1：中断被屏蔽	0x0	R/W
0	IWDGMODE	IWDG 计数溢出模式选择位 0：复位方式 1：中断方式，生成一个中断信号，然后重启计数器，如果中断没有在第二次超时发生之前被清除的话，则生成一个系统复位信号。 **注意：IWDG 产生复位后会复位整个系统**	0x0	R/W

19.4.3 IWDG 计数器重装载寄存器(IWDG_RLOAD)

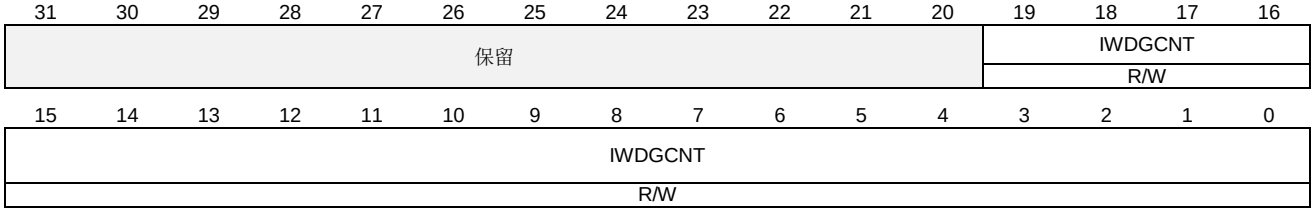
地址偏移：0x08
复位值：0x000FFFFF



位	标记	功能描述	复位值	读写
31:20	Res	保留位	0x0	-
19:0	IWDGRLOAD	IWDG 重装载寄存器	0xFFFFF	R/W

19.4.4 IWDG 计数器值寄存器(IWDG_CNTVAL)

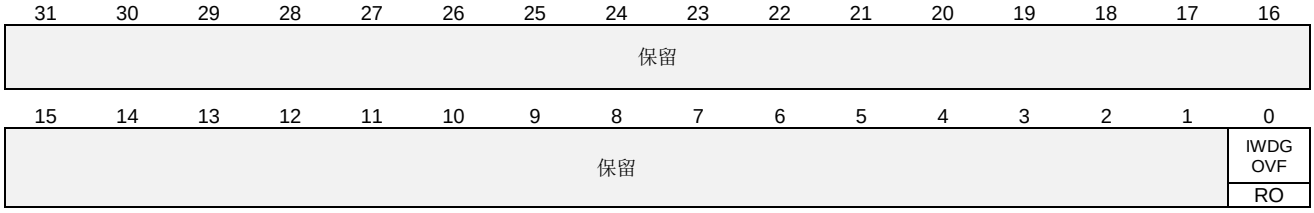
地址偏移：0x0C
复位值：0x000FFFFF



位	标记	功能描述	复位值	读写
31:20	Res	保留位	0x0	-
19:0	IWDGCNT	IWDG 计数值寄存器	0xFFFFF	RO

19.4.5 IWDG 中断状态寄存器(IWDG_SR)

地址偏移：0x10
复位值：0x00000000



位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	–
0	IWDGOVF	IWDG 溢出中断标志位 0：IWDG 无溢出中断发生 1：IWDG 有溢出中断发生 **注意：** 1. 当 IWDG 配置为复位方式，不管 IWDG 计数器是否溢出，这位都不会置高。 2. 当 IWDG 配置为中断方式，不管 IWDGINTMSK 位是否置高，只要 IWDG 计数器溢出，这位就置高。	0x0	RO

19.4.6 IWDG 中断清除寄存器(IWDG_INTCLR)

地址偏移：0x14
复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	–
0	IWDGINTCLR	IWDG 中断清除 0：无任何动作 1：清除 IWDG 中断标志	0x0	WO

19.4.7 IWDG 保护寄存器(IWDG_UNLOCK)

地址偏移：0x18
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															IWDGREN
															R/W

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	–
0	IWDGREN	IWDG 中断清除 0：不能更改 IWDG 的相关寄存器 1：可以更改 IWDG 的相关寄存器	0x0	R/W

注：写 0x55aa6699 到 IWDG_UNLOCK 可以解除 IWDG 的寄存器写保护，写任意非 0x55aa6699 值到 IWDG_UNLOCK 可以开启 IWDG 的寄存器写保护(IWDG_CFG，IWDG_RLOAD,IWDG_INTCLR)

19.5 注意

- 喂狗指令到看门狗定时器被更新需要两个看门狗计数时钟源时钟的延迟。
- 系统操作看门狗时，两次喂狗间隔需要允许有至少三个看门狗时钟。

20 系统窗口看门狗(WWDG)

20.1 概述

窗口看门狗定时器(WWDG)的目的是在一个指定的窗口周期中执行系统复位，防止软件在任何不可预知的条件下进入不可控制的状态。

20.2 特征

- 一个 8 位向下数计数器(WWDG_CNT)和一个 8 位比较值(WINCMP)使 WWDG 超时窗口周期可调；
- 支持 20 位值(PRSC)选择看门狗预分频值；
- 支持窗口计数值比较中断和计数溢出、加载计数值出错复位。

20.3 结构框图

窗口看门狗定时器框图如下：

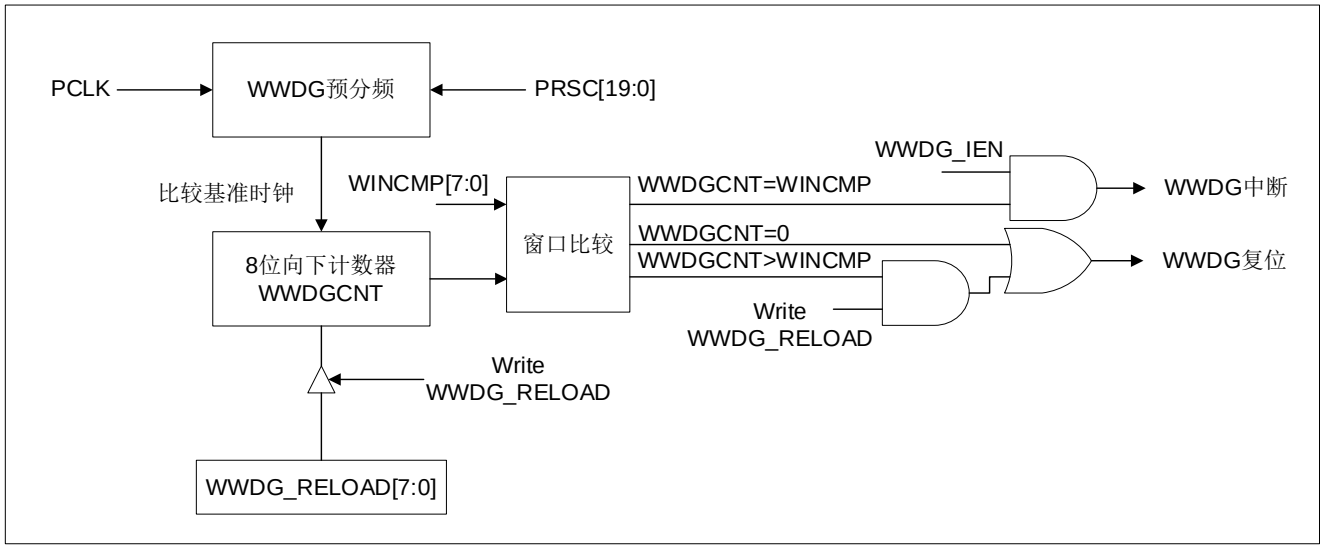


图 20-1 WWDG 结构框图

20.4 基本配置

WWDG 外设时钟源通过 WWDGCKEN(APBCLKEN[12])来使能。

1. 通过 WWDG_RLOAD 配置窗看门狗计数初始值；
2. 通过 WWDG_CR.PRSC 配置计数时钟预分频；
3. 通过 WWDG_CR.WINCMP 配置窗比较值；
4. 根据是否需要使能中断，来配置 WWDG_INTEN. WWDGIEN；
5. 通过 WWDG_CR.WINCMP 来配置窗比较值；
6. 通过 WWDG_CR.WWDGEN 写 1 来开启窗看门狗

20.5 功能描述

窗口看门狗定时器(WWDG)是一个 8 位向下计数器，该计数器带一个可选择预分频值，不同的预分频值对应不同的看门狗定时溢出时间。8 位窗口看门狗定时器的时钟源是 PCLK 时钟经分频后的时钟，看门狗的时钟源带一个可选择的 20 位预分频值，该值可通过 WWDG_CR.PRSC 位来设置选择，对应预分频值如下表。

表 20-1 窗口看门狗定时器预分频值选择

PRSC	预分频值	定时溢出周期	定时溢出间隔 PCLK=24MHz
0x00000	1	$T_{plck} * 1$	41.7ns
0x00001	2	$T_{plck} * 2$	83.4ns
0x00002	3	$T_{plck} * 3$	125.1ns
0x00003	4	$T_{plck} * 4$	166.8ns
0x00004	5	$T_{plck} * 5$	208.5ns
...	…	…	…
0x80000	524289	$T_{plck} * 524289$	21.9ms
…	…	…	…
0xFFFFF	1048576	$T_{plck} * 1048576$	43.8ms

20.5.1 窗口看门狗定时器的计数

当 WWDGEN(WWDG_CR[28])位被使能,窗口看门狗向下计数器将会从 WWDG_CNT 向下递减计数到 0，并且不能够被软件关闭。为了防止程序在非用户指定位置关闭窗口看门狗定时器，窗口看门狗定时器控制寄存器的 WWDGEN 在芯片上电或复位后仅可写一次。当 WWDGEN(WWDG_CR[28])位被软件使能以后，用户不能禁止窗口看门狗定时器(WWDG_CR[28])，修改计数器预分频周期(WWDG_CR[27:8])，或修改窗口比较值(WWDG_CR)，除非芯片复位。窗口看门狗定时器在 CPU 进入 Sleep 模式或是 DeepSleep 模式时会停止计数，CPU 被唤醒后恢复正常工作。

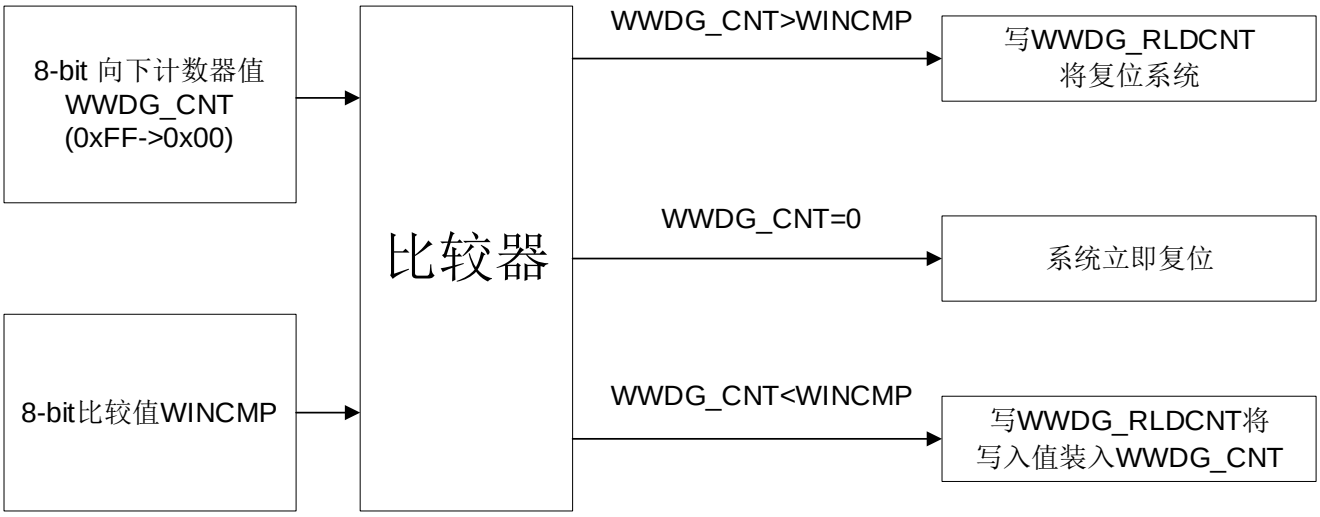


图 20-2 WWDG 复位和重载过程

20.5.2 窗口看门狗定时器比较中断

窗口看门狗定时器向下计数过程中，当窗口看门狗定时器计数值 WWDG_CNT 等于窗口比较值 WINCMP(WWDG_CR)，WWDGIF(WWDG_SR[0])会被置 1 并且 WWDGIF 可以被软件清零。如果 WWDGIEN(WWDG_INTEN[0])位被使能，当 WWDGIF 位被硬件置 1，就会产生窗口看门狗比较匹配中断。

20.5.3 窗口看门狗定时器复位系统

当 WWDG 计数器的值计到 0 时 WWDGRST(RCC_RSTSR[2])会被置 1。在 WWDG 计数器下数到 0 前，用户必须通过对 WWDG_RLOAD 写值来进行重载，从而阻止 WWDG 复位的发生。重载的动作只能在计数器的值小于或等于 WINCMP 值时进行。如果 WWDG 计数器当前值大于 WINCMP 的值，用户对 WWDG_RLOAD 寄存器写，窗口看门狗定时器复位系统信号将立刻产生，并导致芯片复位。

20.5.4 窗口看门狗定时器的窗口设置限制

当用户对 WWDG_RLOAD 寄存器写重载 WWDG 的值的时候，

Tpclk = Thclk*(2*ABPCKDIV)

设定时间间隔：

T = Tpclk*(WWDG_PRSC+1)*(WWDG_RLOAD+1)

用户可以根据需要来配置分频寄存器 WWDG_PRSC 和 WWDG_RLOAD 的值来达到想要的时间间隔。

为了保证正常工作，WWDG_RLOAD 的值要大于等于 1。

20.6 与独立看门狗定时器(IWDG)比较

20.6.1 复位条件和复位延时

IWDG 和 WWDG 通常是应用在系统跑到不可控制的状态后复位系统。IWDG 只有一个条件可以触发复位信号，WWDG 有两种条件可以触发 WWDG 产生复位信号：

WWDGCNT = 0；

WWDGCNT 大于 WINCMP 时往 WWDG_RLOAD 写入。

一旦 WWDGRST 被置 1，WWDG 将立即复位系统。

20.6.2 唤醒功能

IWDG 支持该功能并且在 DeepSleep 模式下继续工作。相比之下，WWDG 不支持该功能并且 WWDG 的计数器在 DeepSleep 模式下会停止计数。

20.7 寄存器列表

WWDG：基址：0x4000 2000

偏移地址	名称	描述	默认值
0x00	WWDG_RLOAD	窗口看门狗定时器重载计数寄存器	0x0000 00FF
0x04	WWDG_CR	窗口看门狗定时器控制寄存器	0x0800 00FF
0x08	WWDG_INTEN	窗口看门狗定时器中断使能寄存器	0x0000 0000
0x0C	WWDG_SR	窗口看门狗定时器状态寄存器	0x0000 0000
0x10	WWDG_INTCLR	窗口看门狗定时器中断清除寄存器	0x0000 0000
0x14	WWDG_CNTVAL	窗口看门狗定时器计数器值寄存器	0x0000 00FF

20.8 寄存器说明

20.8.1 窗口看门狗定时器重载计数寄存器(WWDG_RLOAD)

地址偏移：0x00
复位值：0x0000 00FF

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

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

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7:0	WWDG_RLOAD	窗口看门狗定时器重载计数寄存器。写入值大于 0	0xFF	WO

20.8.2 窗口看门狗定时器控制寄存器(WWDG_CR)

地址偏移：0x04
复位值：0x0800 00FF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留			WWDGEN	PRSC											
			R/W	R/W											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRSC								WINCMP							
R/W								R/W							

位	标记	功能描述	复位值	读写
31:29	Res	保留	0x0	–
28	WWDGEN	窗口看门狗使能位设置该位使能窗口看门狗定时器. 0：禁止窗口看门狗定时器功能 1：使能窗口看门狗定时器功能	0x0	R/W
27:8	PRSC	WWDG 预分频 Fpclk/(PRSC+1)	0x80000	R/W
7:0	WINCMP	WWDG 窗口比较寄存器设置该寄存器调整有效的重载窗口. 注：软件仅能写 WWDG_RLOAD 当 WWDG 计数器值在 0 和 WINCMP 之间。如果软件写 WWDG_RLOAD 当 WWDG 计数器值大于 WINCMP，WWDG 会产生复位信号	0xFF	R/W

注意：当 WWDGEN 设定为 1 后，该寄存器的软件配置将被禁止。

20.8.3 窗口看门狗定时器中断使能寄存器(WWDG_INTEN)

地址偏移：0x08
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															WWDGIEN
															R/W

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	–
0	WWDGIEN	WWDG 中断使能位设置该位使能窗口看门狗定时器中断功能. 0：禁止窗口看门狗定时器中断功能 1：使能窗口看门狗定时器中断功能	0x0	R/W

20.8.4 窗口看门狗定时器状态寄存器(WWDG_SR)

地址偏移：0x0C
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															WWDGIF
															R/W

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	–
0	WWDGIF	WWDG 比较匹配中断标志 0：无窗口看门狗定时器中断 1：有窗口看门狗定时器中断 当 WINCMP 和 WWDG 计数器匹配，该位置 1，软件对 WWDG_INTCLR 的 INTCLR 写 1 清 0 该位。	0x0	RO

20.8.5 窗口看门狗定时器中断清除寄存器(WWDG_INTCLR)

地址偏移：0x10
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															INTCLR
															R/W

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	–
0	INTCLR	WWDG 比较匹配中断标志清除 软件写 1 清 0 该位。	0x0	WO

20.8.6 窗口看门狗定时器计数器值寄存器(WWDG_CNTVAL)

地址偏移：0x14

复位值：0x0000 00FF

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

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

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7:0	WWDGCNT	WWDG 计数器值 该寄存器表示窗口看门狗计数器当前的值，该寄存器只读	0xFF	RO

21 蜂鸣器(BEEP)

21.1 简介

选择 LIRC 时钟，HXT 时钟(8~32MHz)或者 PCLK，可通过分频设定来产生各种频率的蜂鸣信号。

BEEP_CSR.CLKSEL 位来选择得到 f_{BEEP_PRE} 时钟，通过设置 BEEPDIV 将蜂鸣器时钟 f_{BEEP_PRE} 分频得到 f_{BEEP}。通过设置 BEEPSEL，得到 f_{BEEP_O} 蜂鸣信号。

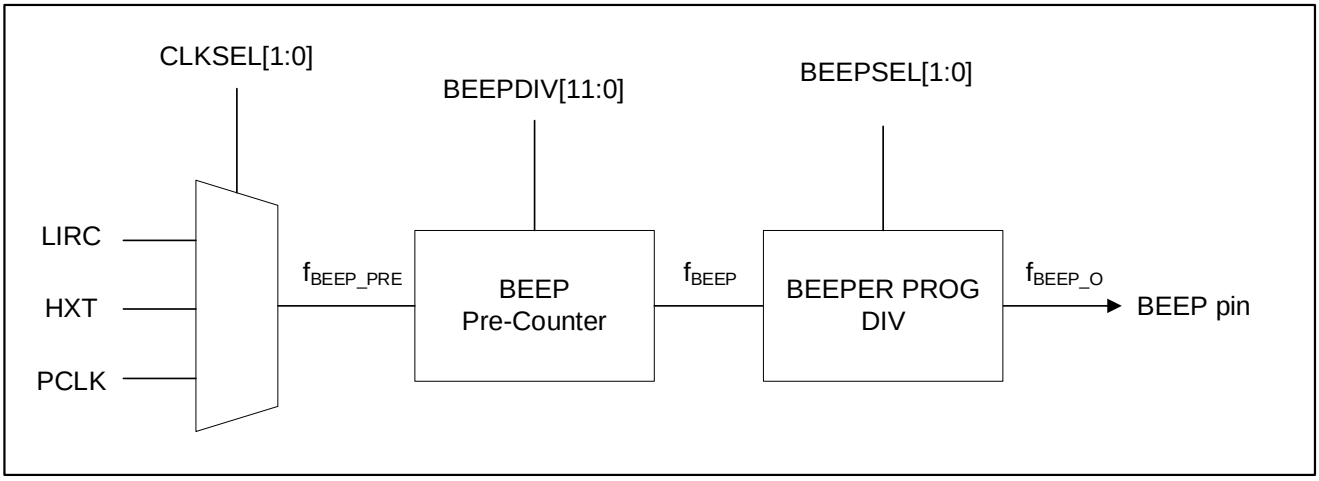


图 21-1 蜂鸣器功能图

21.2 功能描述

21.2.1 蜂鸣器操作

为了使用蜂鸣功能，按顺序执行如下的步骤：

1 根据需要的蜂鸣器频率输出值确定 BEEPDIV 的值；

2 通过写 BEEP_CSR 的 BEEPSEL 位来选择 f_{BEEP_O} 的输出频率；

3 置位 BEEP_CSR 的 BEEPEN 位来使能时钟源；

注：

- 预分频计算器仅仅在当 BEEPDIV 的值不同于复位值 0xFFF 时才开始运行。
- 在蜂鸣器运行过程中应该保持 BEEPDIV 的值不变。

21.2.2 蜂鸣器校准

该步骤可以用来校准 LIRC 时钟以便达到更标准的 f_{BEEP_O}(1kHz，2kHz 或 4kHz)频率输出。

采用如下的步骤：

1 TIM1,TIM2 或者 CLKTRIM 模块来测量内部的低速 RC 振荡器时钟(LIRC)的时钟频率

2 采用如下方法计算 BEEPDIV 的值，这里 A 和 x 是 f_{BEEP_PRE}/f_{BEEP} 的整数和小数部分值：当 x 小于或者等于 A/(1+2*A)时，BEEPDIV = A-1；否则 BEEPDIV = A

3 将 BEEPDIV 值写入到 BEEP_CSR 的 BEEPDIV 位。

21.3 寄存器列表

BEEP 基地址：0x4000 4800

表 21-1 BEEP 寄存器列表和复位值

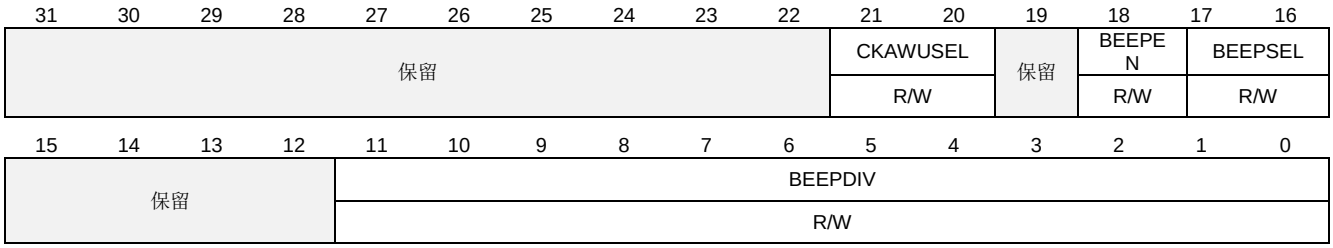
偏移地址	名称	描述	复位值
0x000	BEEP_CSR	蜂鸣器控制寄存器	0x0000 0FFF

21.4 寄存器说明

21.4.1 蜂鸣器控制/状态寄存器(BEEP_CSR)

地址偏移：0x00

复位值：0x0000 0FFF



位	标记	功能描述	复位值	读写
31:22	Res	保留	0x0	-
21:20	CLKSEL	时钟选择 00：停止 01：LIRC(38.4KHz) 10：HXT(8~32MHz) 11：PCLK	0x0	R/W
19	Res	保留	0x0	-
18	BEEPEN	使能蜂鸣	0x0	R/W
17:16	BEEPSEL	蜂鸣器输出 BEEP_O 频率选择位 00：f_{BEEP}/8 01：f_{BEEP}/4 1x：f_{BEEP}/2	0x0	R/W
15:12	Res	保留	0x0	-
11:0	BEEPDIV	蜂鸣器预分频器将时钟分频得到的信号 分频因子为 BEEPDIV+1 $f_{BEEP} = f_{BEEP\_PRE}/(BEEPDIV+1)$	0xFFF	R/W

22 实时时钟(RTC)

22.1 简介

实时时钟(RTC)是一个独立的 BCD 定时器/计数器，提供秒、分、时(12/24 小时制)、周、日、月和年的信息。

RTC 模块拥有自动唤醒功能，用于管理所有的低功耗模式。

两个 32 位寄存器以 BCD 格式存储秒、分、时(12/24 小时制)、周、日、月和年。

RTC 具有自动月份天数补偿功能，每月的天数和闰年的天数可自动调整。

使用两个 32 位寄存器存储可编程报警信息，包括秒、分、时、周、日，月和年。

对由晶体本身的频偏、温度漂移及其他原因引起的任何误差，可以利用 RTC 本身的数字校准功能进行修正。

上电复位后，所有 RTC 寄存器将被禁止访问，以防止意外的写操作。

当设备处于运行模式、低功耗模式，或复位状态(POR 复位除外)，只要电压在工作范围内，RTC 将保持正常运行。

22.2 主要特性

RTC 模块的主要特性如下：

- 日历功能，可显示秒、分、时(12/24 小时制)、周、日、月和年。
- 可进行自动闰年调整。
- 具有闹钟中断和周期中断功能。
- 数字校准电路(计数器定期修正)：0.95ppm 精准度，来自一个几秒钟的校准窗口。
- 时钟源可选：片外低速晶振(LXT)、片内低速振荡器(LIRC)、片外高速晶振(HXT)
- 1Hz 方波输出

22.3 RTC 功能描述

22.3.1 RTC 结构框图

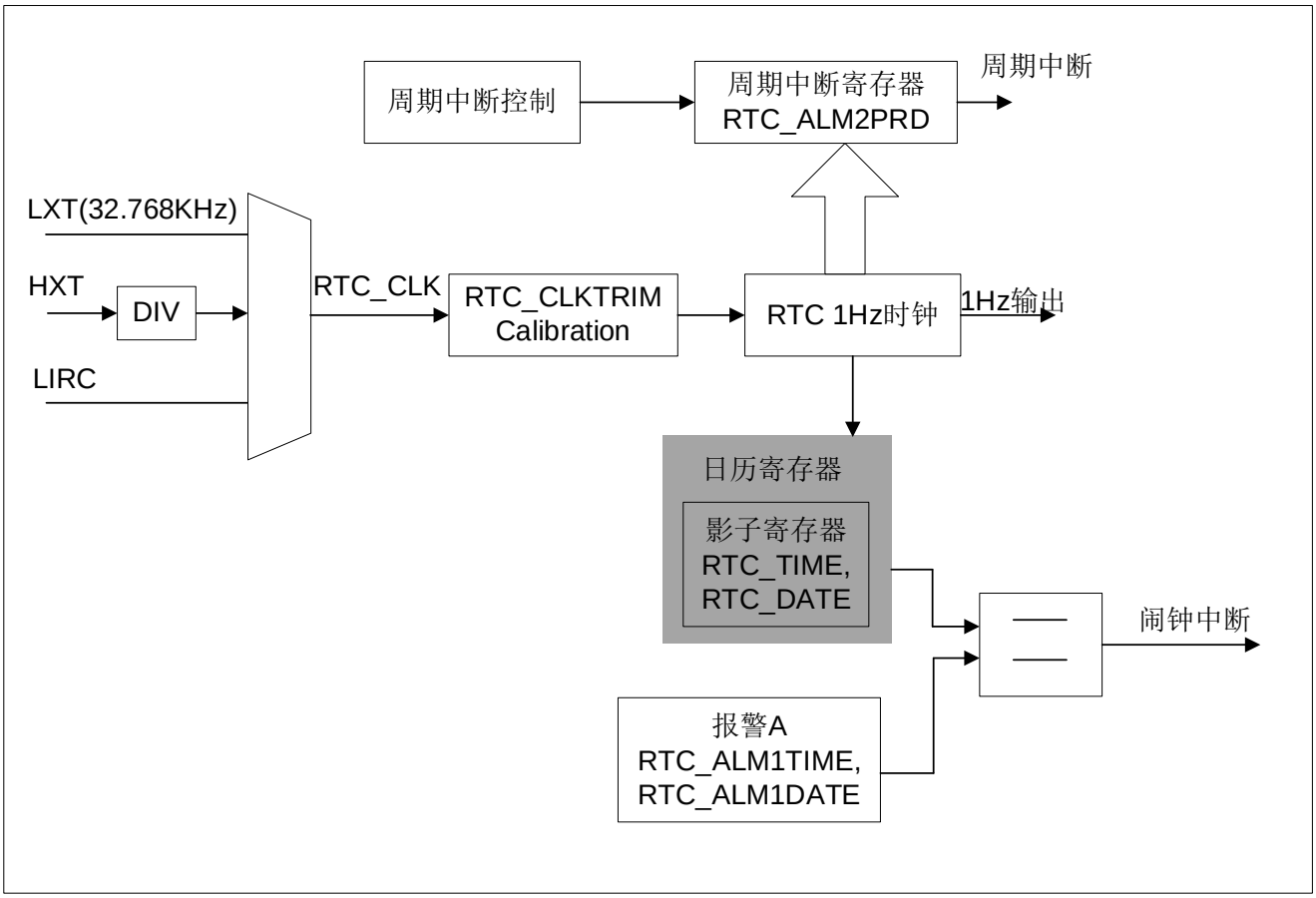


图 22-1 RTC 框图

RTC 模块包括：

- 一个闹钟报警中断
- 一个周期中断
- 校准后的 1Hz 时钟输出
- 万年历寄存器

22.3.2 RTC 时钟

由时钟控制器从以下 3 种时钟中选择 RTC 时钟源(RTC_CLK)：

- LXT 振荡器作为 RTC 时钟
- LIRC 振荡器作为 RTC 时钟
- HXT 振荡器作为 RTC 时钟

更多有关 RTC 时钟源的配置信息，请参考 第 7 章 系统复位与时钟(RCC)

22.3.3 复位过程

任何可用的系统复位源都将导致日历影子寄存器和 RTC 状态寄存器(RTC_ISR)复位至默认值。

然而，下列寄存器的复位于系统复位无任何关联，只与上电复位有关：RTC 配置寄存器(RTC_CR)、时钟控制寄存器(RTC_CLKCR)、RTC 校准寄存器(RTC_CLKTR)、ALM 寄存器(RTC_ALM1DATE/RTC_ALM1TIME/RTC_ALM2PRD)、RTC 当前日历寄存器(RTC_TIME/RTC_DATA)。

除上电复位外，发生任何系统复位时 RTC 将维持运行状态。发生上电复位后，RTC 停止运行，所有 RTC 寄存器复位至默认值。

22.3.4 寄存器的写保护

上电复位后，所有 RTC 寄存器将处于写保护状态。通过向写保护寄存器 RTC_WPR 写入指定关键字来启动 RTC 寄存器的写权限。

通过以下操作解除所有 RTC 寄存器(RTC_ISR[13:8],RTC_TAFCR 和 RTC_BKPxR 除外)的写保护。

1. 向 RTC_WPR 寄存器写入‘0xCA’；
2. 向 RTC_WPR 寄存器写入‘0x53’。

注：保护解除后，任何对该寄存器的再一次写将重新激活写保护。

22.3.5 日历初始化及配置

按照以下顺序完成时间和日期值的初始化，包括时间格式和预分频器的配置：

1. 通过 RTC_CLKCR.CKSEL 选择 RTC 计时时钟源，如果选择 HXT 时钟还要先设定预分频器。
2. 设定 RTC_CLKCR.RTCCKEN 使能 RTC 计时时钟。
3. RTC_ISR 寄存器的 WAIT 位置“1”，进入初始化模式。
4. 等待 RTC_ISR 寄存器的 WAITF 位置“1”，确保已经正式进入初始化模式。由于时钟同步的延迟，该过程大约需要 2 个 RTCCLK 时钟周期。在该模式下日历计数器暂停运行，此时可更新时间和日期计数器的值。
5. 通过 RTC_CR 寄存器的 FMT 位设置时间格式(12 小时制/24 小时制)。
6. 将初始时间和日期值加载到时间寄存器(RTC_TIME 与 RTC_DATE)。
7. 需要进行时钟误差补偿时，设定时钟补偿寄存器 RTC_CLKTRM。
8. 清除 RTC_ISR.WAIT 位的值退出初始化模式。日历计数器的实际值将会自动加载，并在 4 个 RTCCLK 时钟周期后重新启动。

在完成上述一系列初始化操作后，日历将开始计时。

22.3.6 读出计数寄存器

当 RTC_CTRL 寄存器的 BYPSHAD 控制位被清除时：

为确保在安全同步机制下正常读 RTC 日历寄存器(RTC_TIME 和 RTC_DATE)，APB 时钟频率(f_{PCLK})应至少为 RTC 时钟频率(f_{RTCCLK})的 7 倍以上。

当 APB 时钟频率低于 7 倍 RTC 时钟频率时，软件必须两次读取日历时间和日期寄存器。如果第二次读取的值与第一次读取的值相同，说明返回值是正确的。否则需再次读取。任何情况下，APB 时钟频率都必须大于 RTC 时钟频率。

日历寄存器的内容被复制到 RTC_TIME 和 RTC_DATE 影子寄存器中时，RTC_ISR 寄存器的 RSF 位被置位。复制操作每两个 RTCCLK 周期执行一次。为确保三者的值保持一致，读 RTC_TIME，锁定高阶日历影子寄存器中的值，直至 RTC_DATE 中的值被读取。为避免软件在时间间隔少于 2 个 RTCCLK 周期的情况下多次访问日历，每次读日历 RSF 位应由软件清零，软件必须等待 RSF 位被置位后才能读 RTC_TIME 和 RTC_DATE 寄存器。

从低功耗模式(待机)唤醒后，RSF 位应由软件清零，软件必须等待 RSF 位被再次置位后才能读 RTC_TIME 和 RTC_DATE 寄存器。

RSF 位应在唤醒后被清除，而不是进入低功耗模式前。

系统复位后，软件必须等待 RSF 位被置位后才能读 RTC_TIME 和 RTC_DATE 寄存器。事实上系统复位将导致影子寄存器复位至其默认值。

当 RTC_CR 寄存器的 BYPSHAD 控制位被置位时(无需考虑影子寄存器):

读日历寄存器，直接从日历计数器获取值，无需等待 RSF 位被置位。此功能在刚退出低功耗模式(停止或待机模式)时非常有用，因为影子寄存器在低功耗模式下不会自动更新。在 BYPSHAD 为“1”时，如果两次读寄存器之间出现 RTCCLK 边沿，不同寄存器中的结果可能会互不相关。此外，如果在读操作过程中遇到 RTCCLK 边沿，则某个寄存器的值可能不正确。软件必须读取所有的寄存器两次，并比较两次读取的结果，或者通过比较两组最低有效日历寄存器的结果，以检验数据是否正确且有一定关联。

22.3.7 写入计数寄存器

1. 设定 RTC_ISR.WAIT=1，停止日历寄存器计数，进入写模式；
2. 查询直到 RTC_ISR.WAITF=1；
3. 写入秒，分，时，周，日，月年计数寄存器值；
4. 设定 RTC_ISR.WAIT=0，计数器重新开始。注意，须在 1 秒内完成所有写操作；
5. 查询直到 RTC_ISR.WAITF=0。

22.3.8 闹钟设定

设置 RTC_CR 寄存器的 ALM1EN 位，启动闹钟功能。当日历中秒,分,时,日期，星期，月，年与报警寄存器 RTC_ALM1TIME 和 RTC_ALM1DATE 中设定的值匹配，ALM1_F 由硬件置 1。所有日历字段都可以通过 RTC_ALM1DATE 寄存器中的 ALMxEN 位选择为报警源。设置 RTC_CR 寄存器的 ALM1_INTEN 位，会产生报警中断。

22.3.9 校准 1HZ 输出

RTC 可选择输出校准后的 1Hz 时钟。通过 RTC1HZOE 设定输出使能，通过 RTC_CLKTRM 来设定校准值。

22.3.10 RTC 时钟校准

本产品通过每隔一个固定的时间周期屏蔽指定的 RTC 时钟周期数来补偿 RTC 时钟的频率，通过 RTC_CLKTRIM.MODE 来选择调整的时间间隔：

0b00：每 60 秒(SEC=00)校准一次

0b01：每 30 秒(SEC=00，30)校准一次

0b10：每 15 秒(SEC=00,15,30,45)校准一次

0b11：每 6 秒(SEC=00,06,12,18,24,30,36,42,48,54)校准一次

通过 RTC_CLKTRM.TRIM 来指定屏蔽的 RTC 时钟周期数。

注意 RTC_CLKTRM.TRIM 的值时有符号整数，范围为-128~+127。

22.4 RTC 中断

RTC 支持两种中断类型。闹钟中断、定周期中断。闹钟中断与定周期中断共用一个中断信号，通过标志寄存器位来区分中断源。

22.4.1 RTC 闹钟中断

1 设定 RTC_CR.ALM1EN=0，禁止闹钟功能；

2 设定时间闹钟寄存器 RTC_ALM1TIME、日期闹钟寄存器 RTC_ALM1DATE；

3 设定 RTC_CR.ALM1EN=1，闹钟许可；

4 清除中断标志位 ALM1_F；

5 设定 RTC_CR1.INTEN=1，闹钟中断许可，若当前日历时间与闹钟寄存器相等时，触发闹钟中断；

6 等待发生中断；

22.4.2 RTC 周期中断

控制寄存器 RTC_CTRL 的 ALPR_IRQ_ENA=1 时，选择的周期发生后，触发定周期唤醒中断，由于闹钟和定周期共用中断，通过标志寄存器位来区分。

1 设定 RTC_CR.ALM2_INTEN=0，禁止周期中断功能；

2 设定周期闹钟寄存器 RTC_ALM2PRD；

3 清除中断标志位 ALM2_F；

4 设定 RTC_CR1.INTEN=1，周期中断许可，选择的周期发生后，触发定周期唤醒中断；

5 等待发生中断；

22.5 RTC 寄存器列表

基地址：0x4000 3000

偏移地址	名称	描述	默认值
0x00	RTC_CR	RTC 控制寄存器	0x00000000
0x04	RTC_CLKCR	RTC 时钟控制寄存器	0x00000000
0x08	RTC_TIME	RTC 时间寄存器	0x00000000
0x0C	RTC_DATE	RTC 日期寄存器	0x00000000
0x10	RTC_ALM1TIME	RTC 时间闹钟寄存器	0x00000000
0x14	RTC_ALM1DATE	RTC 日期闹钟寄存器	0x00000000
0x18	RTC_ALM2PR	RTC 周期闹钟寄存器	0x00000000
0x1C	RTC_CLKTRM	RTC 时钟调校寄存器	0x00000000
0x20	RTC_ISR	初始化和状态寄存器	0x00000000
0x24	RTC_INTCLR	RTC 状态清除寄存器	0x00000000
0x28	RTC_WPR	RTC 写保护寄存器	0x00000000

22.6 RTC 寄存器说明

22.6.1 RTC 控制寄存器(RTC_CR)

偏移地址：0x00
复位值：0x00000000

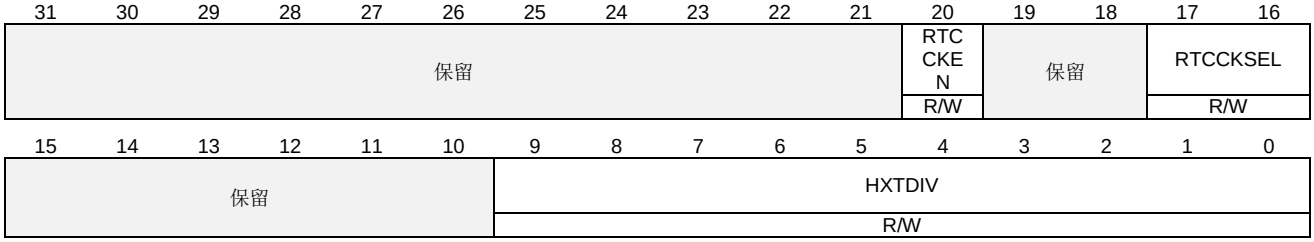
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留							START	保留	ALM1EN	ALM2_INTEN	ALM1_INTEN	保留	FMT	RTC1HZOE	BYPSHAD
							R/W		R/W	R/W	R/W		R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:9	Res	保留	0x0	-
8	START	0：停止 RTC 计数器 1：使能 RTC 计数器	0x0	R/W
7	Res	保留	0x0	-
6	ALM1EN	ALM1 闹钟功能使能 0：禁止 ALM1 闹钟功能 1：使能 ALM1 闹钟功能 注意：在 RTCCKEN=1 日历计数过程中并且 ALM1_INTEN=1 中断许可的情况下使能 ALM1_INTEN 时，为防止误动作请将系统中断关闭。使能后请将 ALM1F 标志位清除。	0x0	R/W
5	ALM2_INTEN	ALM2 周期中断使能 0：禁止 ALM2 周期中断 1：使能 ALM2 周期中断	0x0	R/W
4	ALM1_INTEN	ALM1 闹钟中断使能。 0：禁止 ALM1 闹钟中断 1：使能 ALM1 闹钟中断	0x0	R/W
3	Res	保留	0x0	-
2	FMT	时间格式 0：AM/PM 时间格式(12 小时制) 1：24 小时/天格式	0x0	R/W
1	RTC1HZOE	RTC 1Hz 输出时能 0：禁止 1：使能	0x0	R/W
0	BYPSHAD	绕过影子寄存器 0：从影子寄存器读取日历值，每两个 RTCCLK 周期更新一次； 1：直接从日历计数器读取日历值。 注：如果 APB1 时钟频率低于 RTCCLK 频率的 7 倍，BYPSHAD 必须置“1”。	0x0	R/W

22.6.2 RTC 时钟控制寄存器(RTC_CLKCR)

偏移地址：0x04
复位值：0x00000000



位	标记	功能描述	复位值	读写
31:21	Res	保留	0x0	-
20	RTCCKEN	RTC 计数时钟使能： 由软件置 1 或清 0 写： 0：RTC 时钟关闭 1：RTC 时钟开启 读： 0：RTC 时钟关闭 1：RTC 时钟开启	0x0	R/W
19:18	Res	保留	0x0	-
17:16	RTCCKSEL	RTCCKSEL：RTC 时钟源选择 由软件设置来选择 RTC 时钟源。一旦 RTC 时钟源被选定，这些位值不能被改变，除非 RTC 被复位。可通过设置 RTCRST 位来复位 RTC 域。 00：LXT 振荡器作为 RTC 时钟 01：LIRC 振荡器作为 RTC 时钟 10：F_{hxt}/(HXTDIV+1) 11：保留	0x0	R/W
15:10	Res	保留	0x0	-
9:0	HXTDIV	外部高速晶振时钟分频 0：停止 $F=F_{hxt}$/(HXTDIV)	0x0	R/W

22.6.3 RTC 时间寄存器(RTC_TIME)

偏移地址：0x08
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留					WEEK			保留		H20_PA	HOUR19				
					R/W					R/W	R/W				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	MIN							保留	SEC						
	R/W								R/W						

位	标记	功能描述	复位值	读写
31:27	Res	保留	0x0	-
26:24	WEEK	星期计数器。星期计数器值为二进制计数，计数区间从 0 到 6。(7 不被使用，除非不使用星期计数)星期和星期计数器值得对应关系由用户定义。(比如星期日=0，星期一=1……星期六=6)	0x0	R/W
23:22	Res	保留	0x0	-
21	H20_PA	这两位表示小时计数器。HOUR19 的值为 BCD 编码。时间格式是由时钟系统决定的。 12 小时制模式，H20_PA 指上午或者下午。24 小时制模式，H20_PA 决定了计数器的十位是否为 2。 12 小时制模式，当[H20_PA，HOUR19]从[1,11](11PM)数到[0,12](12AM)的时候，日期计数器增加一天。24 小时制模式，当[H20_PA，HOUR19]从[1,3](23H)数到[0,0](0H)的时候，日期计数器增加一天。	0x0	R/W
20:16	HOUR19		0x0	R/W
15	Res	保留	0x0	-
14:8	MIN	分钟计数器。分钟计数器的值为 BCD 编码，计数区间从 0 到 59。当分钟计数器从 59 数到 0 的时候，小时计数器增长 1。当这个计数器被写入时，小于一秒的时间将被忽略掉。	0x0	R/W
7	Res	保留	0x0	-
6:0	SEC	秒计数器。秒计数器的值为 BCD 编码，计数区间从 0 到 59。当秒计数器从 59 数到 0 的时候，分钟计数器增长 1。当这个计数器被写入时，小于一秒的时间将被忽略掉。	0x0	R/W

22.6.4 RTC 日期寄存器(RTC_DATE)

偏移地址：0x0C
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								YEAR							
								R/W							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CEN	保留		MONTH					保留		DAY					
R/W			R/W							R/W					

位	标记	功能描述	复位值	读写
31:24	Res	保留	0x0	-
23:16	YEAR	年计数器。年计数器代表了十进制年的十位和个位。年计数器为 BCD 编码，计数区间从 00 到 99。当年计数器从 99 数到 00 时，世纪计数器增长 1。 当世纪计数器为 0 时，04，08，…92，96 为闰年。 当世纪计数器为 1 时，00，04，08，…92，96 为闰年。	0x0	R/W
15	CEN	世纪计数器。0 代表 20 世纪，1 代表 21 世纪	0x0	R/W
14:13	Res	保留	0x0	-
12:8	MONTH	月计数器。月计数器为 BCD 编码，计数区间从 01 到 12。当年计数器从 12 数到 01 时，年计数器增长 1。	0x0	R/W
7:6	Res	保留	0x0	-
5:0	DAY	天计数器。天计数器为 BCD 编码，计数区间如下： 01 到 31：一月，三月，五月，七月，八月，十月，十二月； 01 到 30：四月，六月，九月，十一月； 01 到 29：闰年的二月 01 到 28：非闰年的二月 当年计数器从 99 数到 00 时，世纪计数器增长 1。	0x0	R/W

22.6.5 RTC 时间闹钟寄存器(RTC_ALM1TIME)

偏移地址：0x10
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留					ALWEEK			保留		ALH20_PA	ALHOUR19				
					R/W					R/W	R/W				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	ALMIN							保留	ALSEC						
	R/W								R/W						

位	标记	功能描述	复位值	读写
31:27	Res	保留	0x0	-
26:24	ALWEEK	闹钟星期设定	0x0	R/W
23:22	Res	保留	0x0	-
21	ALH20_PA	闹钟小时设定。参考小时计数寄存器。	0x0	R/W
20:16	ALHOUR19			
15	Res	保留	0x0	-
14:8	ALMIN	闹钟分钟设定	0x0	R/W
7	Res	保留	0x0	-
6:0	ALSEC	闹钟秒设定	0x0	R/W

22.6.6 RTC 日期闹钟寄存器(RTC_ALM1DATE)

偏移地址：0x14
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留	ALMYEAREN	ALMMONEN	ALMDAYEN	ALMWEEKEN	ALMHOUREN	ALMMINEN	ALMSECEN	ALYEAR							
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALCEN	保留		ALMONTH					保留	ALDAY						
R/W			R/W						R/W						

位	标记	功能描述	复位值	读写
31	Res	保留	0x0	-
30	ALMYEAREN	闹钟年设定使能	0x0	R/W
29	ALMMONEN	闹钟月设定使能	0x0	R/W
28	ALMDAYEN	闹钟日设定使能	0x0	R/W
27	ALMWEEKEN	闹钟星期设定使能	0x0	R/W
26	ALMHOUREN	闹钟小时设定使能	0x0	R/W
25	ALMMINEN	闹钟分钟设定使能	0x0	R/W
24	ALMSECEN	闹钟秒设定使能	0x0	R/W
23:16	ALYEAR	闹钟年设定	0x0	R/W

15	ALCEN	闹钟世纪设定	0x0	R/W
14:13	Res	保留	0x0	-
12:8	ALMONTH	闹钟月设定	0x0	R/W
7:6	Res	保留	0x0	-
5:0	ALDAY	闹钟日设定	0x0	R/W

22.6.7 RTC 周期闹钟寄存器(RTC_ALM2PRD)

偏移地址：0x18
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留												ALM2PR_CNT			
												R/W			

位	标记	功能描述	复位值	读写
31:4	Res	保留	0x0	-
3:0	ALM2PR_CNT	周期闹钟 2 计数周期设定 0x0：关闭周期闹钟 2 0x1：1 秒 0x2：1/2 秒 0x3：1/4 秒 0x4：1/8 秒 0x5：1/16 秒 0x6：1/32 秒 0x7：1/64 秒 0x8：1/128 秒 0x9：10 秒 0xA：30 秒 0xB：1 分钟 0xC：30 分钟 0xD：60 分钟 0xE：12 小时 0xF：24 小时	0x0	R/W

22.6.8 RTC 时钟调校寄存器(RTC_CLKTRIM)

偏移地址：0x1C
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留						TRIM_MODE		TRIM							
						R/W		R/W							

位	标记	功能描述	复位值	读写
31:10	Res	保留	0x0	-
9:8	MODE	时钟调节寄存器。决定了时钟调节的频率。 0x0：每 60 秒(SEC=00) 0x1：每 30 秒(SEC=00，30) 0x2：每 15 秒(SEC=00,15,30,45) 0x3：每 6 秒(SEC=00,06,12,18,24,30,36,42,48,54)	0x0	R/W
7:0	TRIM	时钟补偿时间寄存器。此寄存器为有符号整数。(-128~+127)	0x0	R/W

22.6.9 RTC 初始化和状态寄存器(RTC_ISR)

偏移地址：0x20
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留										ALM2_F	ALM1_F	保留	RSF	WAITF	WAIT
										RO	RO		R/W	R/W	RW

位	标记	功能描述	复位值	读写
31:6	Res	保留	0x0	-
5	ALM2_F	周期闹钟 2 中断原始状态寄存器。 当此寄存器被读出时，状态值被返回： 0：周期闹钟 2 中断没有被激活。 1：周期闹钟 2 中断被激活。	0x0	RO
4	ALM1_F	闹钟中断原始状态寄存器。 当此寄存器被读出时，状态值被返回： 0：闹钟中断没有被激活。 1：闹钟中断被激活。	0x0	RO
3	Res	保留	0x0	-
2	RSF	寄存器同步标志 每当日历寄存器中的内容复制到影子寄存器(RTC_SSRx, RTC_TRx and RTC_DRx)中时，该位由硬件置位。当处于忽略影子寄存器模式(BYPSHAD=1)下时，该位由硬件在初始化模式下清除。该位也可由软件清除。	0x0	R/W

		在初始化模式下，该位可由硬件/软件清除。 0：日历影子寄存器尚未同步； 1：日历影子寄存器已经同步。 注意不能软件写 1		
1	WAITF	0：非写入/读出状态 1：写入/读出状态 注意：WAIT 位设定是否有效标志。在写入/读出前请确认该位是否为“1”。计数过程中，在 WAIT 位清“0”后等待写入完成后该位才清“0”。	0x0	R/W
0	WAIT	0：正常计数模式 1：写入/读出模式 注意：在写入/读出时请将该位置“1”，由于计数器在连续计数，请在 1 秒的时间内完成写入/读出操作并将该位清“0”。	0x0	R/W

22.6.10 RTC 状态清除寄存器(RTC_INTCLR)

偏移地址：0x24
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留										ALM2_CLR	ALM1_CLR	保留			
										WO	WO				

位	标记	功能描述	复位值	读写
31:6	Res	保留	0x0	-
5	ALM2_CLR	周期闹钟 2 中断原始状态清除寄存器。 当此寄存器被写入时，中断原始状态被要求清除： 0：没有操作。 1：周期闹钟 2 中断原始状态被清除。	0x0	WO
4	ALM1_CLR	闹钟中断原始状态清除寄存器。 当此寄存器被写入时，中断原始状态被要求清除： 0：没有操作。 1：闹钟中断原始状态被清除。	0x0	WO
3:0	Res	保留	0x0	-

22.6.11 RTC 写保护寄存器(RTC_WPR)

偏移地址：0x28
复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	-
7:0	WPR	写入指定关键字来*启动 RTC 寄存器的写权限。* *1. 向 RTC_WPR 寄存器写入‘0xCA’；* *2. 向 RTC_WPR 寄存器写入‘0x53’。* 注：保护解除后，任何对该寄存器的再一次写将重新激活写保护。	0x0	WO

23 低功耗通用异步收发器(LPUART)

23.1 概述

本产品带有 1 个 LPUART 模块，支持半双工和全双工传输；支持 8bit、9bit 数据格式；支持 MODE0/1/2/3 四种不同传输模式；LPUART 的波特率由 LPTIM 产生，也可以由内部自动波特率发生器产生；支持多机通讯模式；支持自动地址识别；支持给定地址和广播地址，支持低功耗模式。

LPUART 为支持低功耗应用，除了原本的 PCLK 时钟外，增加了一路 SCLK 时钟，并可以控制 LPUART 工作状态。LPUART 模块内部寄存器配置逻辑工作于 PCLK 时钟域，数据收发逻辑工作于 SCLK 时钟域。当系统进入低功耗模式且后且处在 LPUART 工作状态，关闭高频 PCLK 时钟，打开低频 SCLK 时钟，LPUART 仍旧可以进行正常的数据收发。关闭工作状态则停止波特率的生成。

SCLK 时钟来源可以选择：PCLK、外部低速时钟(LXT)，内部低速时钟(LIRC)。在 LPMODE=1 时，SCLK 时钟还支持 1/2/4/8/16/32/64/128 倍的预分频。

注意，在 LPMODE=0 时，LPUART 接收的是 LPTIM 时钟的 TOGGLE 输出信号而非 OVERFLOW 信号，因此必须使能 LPTIM 的 TOGGLE 输出。

23.2 结构框图

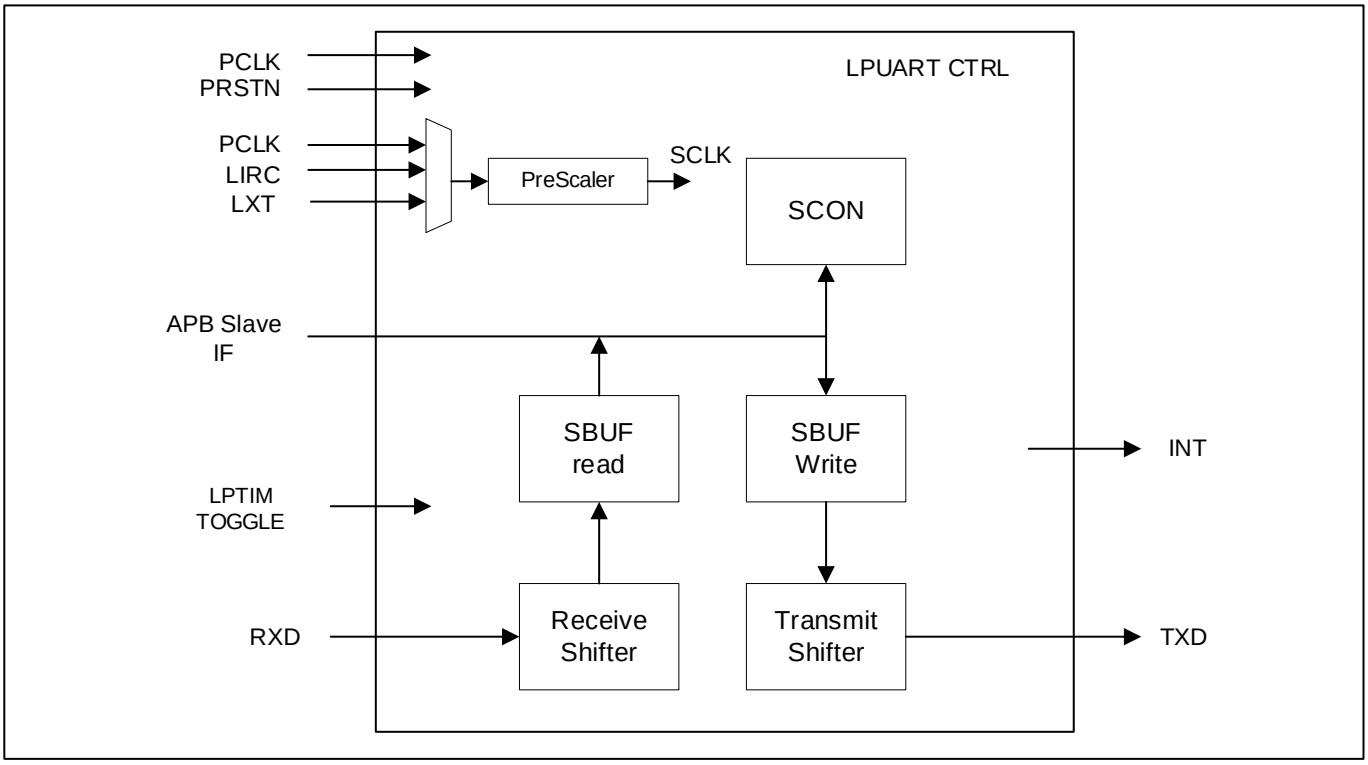


图 23-1 LPUART 结构框图

23.3 工作模式

与通用 UART(UART0/1)相比，LPUART 增加了一个 LPMODE 控制位。当该位置“1”时，只支持 MODE1/3 工作模式，并且波特率生成方式也会发生改变。具体描述请参考以下章节。

23.3.1 MODE0(同步模式，半双工)

当工作在 MODE0 时，UART 工作在同步模式，其波特率为固定的 SCLK 时钟的 1/12。UART 接收数据由 RXD 输入、UART 发送数据有 RXD 输出，RXD 此时为输入输出端口。UART 同步移位时钟由 TXD 输出，TXD 此时为输出端口。注意，本模式只能作为主机发送同步移位时钟，不可以作为从机从外部接收该时钟。该模式下，传输的数据位宽只能是 8 位的，没有起始位和结束位。

将 LPUARAT_SCON.SM0 和 LPUART_SCON.SM1 清零，可进入 MODE0 工作模式。

当 LPMODE=1 时，不支持 MODE0 工作模式。

23.3.1.1 发送数据

发送数据时，清除 LPUART_SCON.REN 位，并将数据写入 SBUF 寄存器。此时，发送数据将从 RXD 输出(低位在先，高位在后)，同步移位时钟从 TXD 输出。

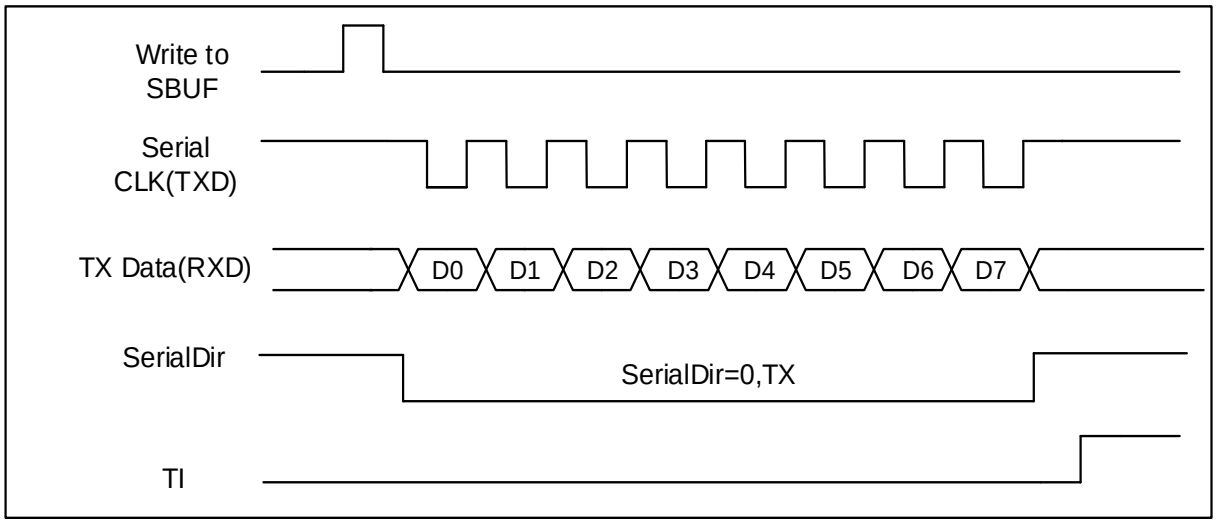


图 23-2 MODE0 发送数据

23.3.1.2 接收数据

接收数据时，将 LPUART_SCON.REN 位置 1。当接收结束，将 LPUART_INTSR.RI 位清 0，且数据可从 LPUART_SBUF 寄存器读出。此时，接收数据从 RXD 输入(低位在先，高位在后)，同步移位时钟从 TXD 输出。

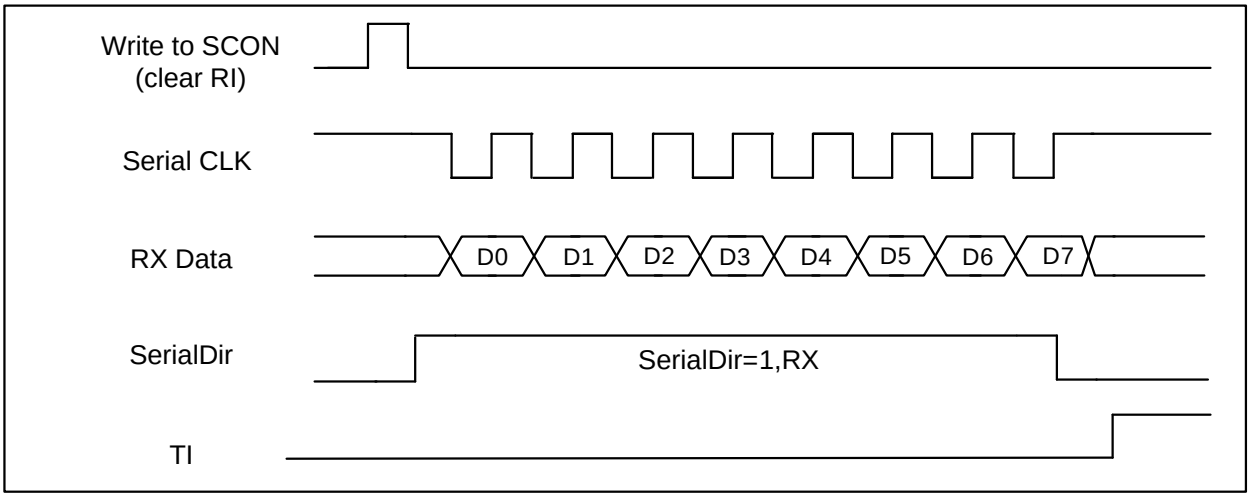


图 23-3 MODE0 接收数据

23.3.2 MODE1(异步模式，全双工)

当工作在 MODE1 时，发送数据通过 TXD 发送，接收数据通过 RXD 接收。该数据由 10 位组成：起始位“0”开始，紧接着 8 位数据位(低位在先，高位在后)，最后是结束位“1”。

将 LPUART_SCON.SM0 清 0，LPUART_SCON.SM1 置 1，可进入 MODE1 工作模式。该模式下，当 LPMODE=0 时，LPUART 的波特率可以选择由自动波特率发生器或者定时器 LPTIM 模块产生，并且是可编程的。

当 LPMODE=1 时，波特率计算方式发生改变，具体参考波特率编程章节。

23.3.2.1 发送数据

发送数据时，与 LPUART_SCON.REN 的值无关，将所发送数据写入 LPUART_SBUF 寄存器中，数据就会从 TXD 移出(低位在先，高位在后)。

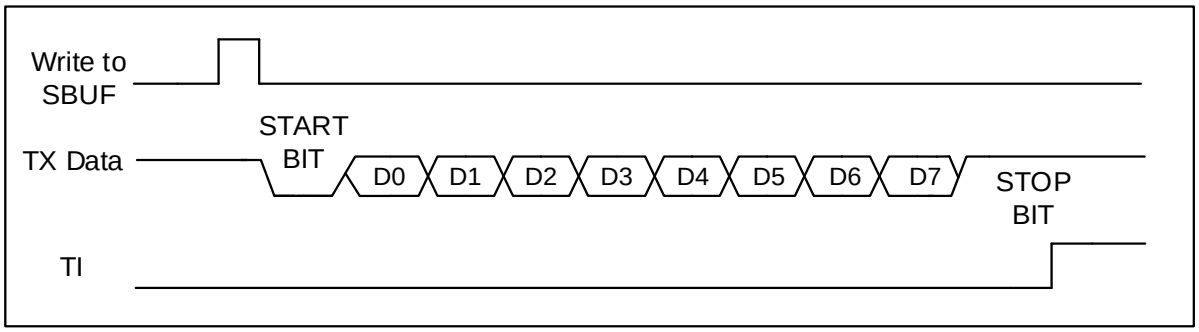


图 23-4 MODE1 发送数据

23.3.2.2 接收数据

接收数据时，需将 LPUART_SCON.REN 位置 1，开始接收 RXD 上数据(低位在先，高位在后)。当接收完毕，将 LPUART_INTSR.RI 位清 0，且可以从 LPUART_SBUF 寄存器读出。

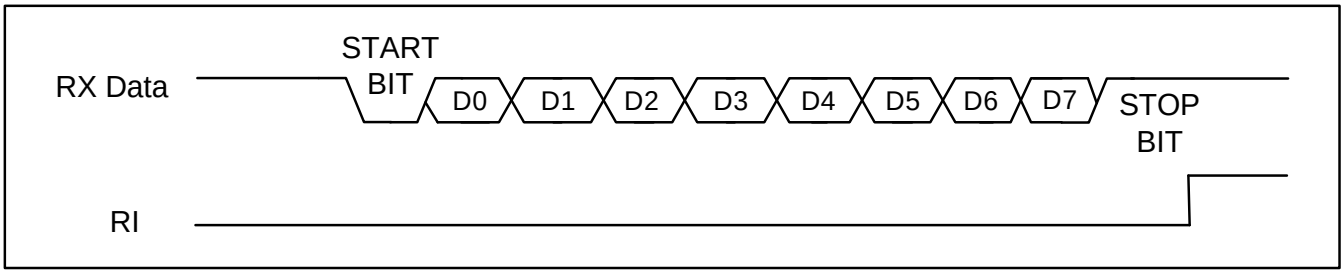


图 23-5MODE1 接收数据

23.3.3 MODE2(异步模式，全双工)

当工作在 MODE2 时，发送数据通过 TXD 发送，接收数据通过 RXD 接收。该数据由 11 位组成：起始位“0”开始，接着是 8 个数据位，1 个 TB8 位和结束位。额外的 TB8 位是用来在多机通讯环境下使用，当 TB8=1，表明所接收的是地址帧；当 TB8=0，表明所接收的是数据帧。当不需要多机通讯时，此位也可以作为奇偶校验位来使用。

将 LPUART_SCON.SM0 置 1，LPUART_SCON.SM1 清 0，可进入 MODE2 工作模式。该模式下，波特率可以独立产生，不需要外部 TIMER 产生。

当 LPMODE=1 时，不支持 MODE2 工作模式。

23.3.3.1 发送数据

发送数据时，与 LPUART_SCON.REN 的值无关，并将所发送数据写入 LPUART_SBUF 寄存器中，数据就会从 TXD 移出(低位在先，高位在后)。

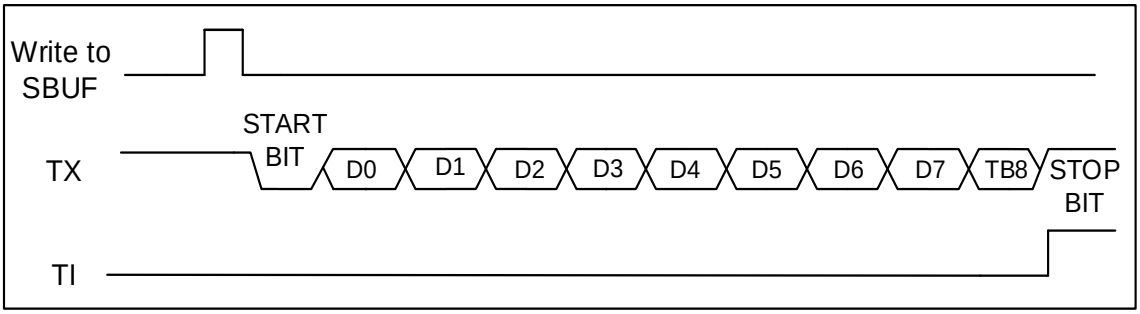


图 23-6 MODE2 发送数据

23.3.3.2 接收数据

接收数据时，需将 LPUART_SCON.REN 位置 1，开始接收 RXD 上数据(低位在先，高位在后)。当接收完毕，将 LPUART_INTSR.RI 位清 0，且可以从 LPUART_SBUF 寄存器读出。

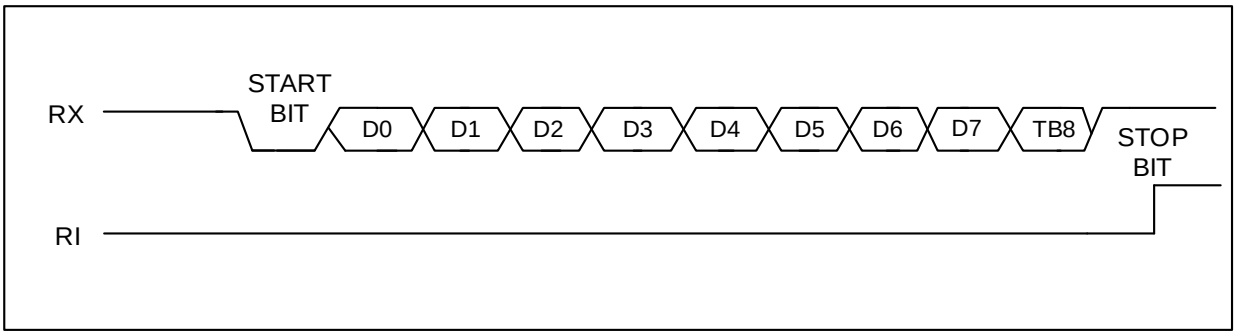


图 23-7 MODE2 接收数据

23.3.4 MODE3(异步模式，全双工)

MODE3 的数据格式，传输时序以及操作方式都与 MODE2 相同，唯一的区别是 MODE3 的波特率由 LPTIM 产生或者内部自动波特率发生器产生，而不是像 MODE2 由设备自己独立产生。MODE3 的波特率是可编程的，波特率生成方式与 MODE1 相同。

将 LPUART_SCON.SM0 置 1，LPUART_SCON.SM1 置 1，可进入 MODE3 工作模式。

当 LPMODE=1 时，支持 MODE3 工作模式。但是波特率计算方式发生改变，具体参考波特率编程章节。

23.4 波特率编程

23.4.1 MODE0

LPMODE=0

当工作在 MODE0 时，波特率被固定在 PCLK 的 1/12，不需要 LPTIM 的支持。

LPMODE=1

当 LPMODE=1 时，不支持该模式。

23.4.2 MODE1/3

LPMODE=0

当工作在 MODE1 或者 MODE3 时，波特率可以由 LPTIM 的溢出时间决定。具体公式如下图所示：

$$\text{BaudRate} = \frac{(\text{LPUART\_SCON.DBAUD} + 1) * \text{F}_{\text{sclk}}}{32 * (2^{16} - LPTIM\_BGLOAD)}$$

其中，LPUART_SCON.DBAUD 表示双倍波特率，F_{SCLK} 为 SCLK 时钟频率，LPTIM_BGLOAD 为 LPTIM 的周期装载计数值。

注意，LPTIM 必须配置为 16 位自动重载模式，立即重载寄存器(LPTIM_LOAD)和周期重载寄存器(LPTIM_BGLOAD)要写入相同的初始值。

也可以使用自身波特率生成模式

$$\text{BaudRate} = \frac{(\text{LPUART\_SCON.DBAUD} + 1) * \text{F}_{\text{sclk}}}{32 * (\text{LPUART\_BAUDCR.BRG} + 1)}$$

其中，UARTX_SCON.DBAUD 表示双倍波特率，F_{sclk} 为 SCLK 时钟频率。

LPMODE=1

当 LPMODE 设为“1”时，波特计算公式与上述公式不同，简化为：

$$\text{BaudRate} = \frac{\text{F}_{\text{sclk}}}{4 * \text{分频值}}$$

其中，F_{sclk} 为 SCLK 时钟频率，分频值请参考 LPUART_SCON.PRSC。

23.4.3 MODE2

LPMODE=0

当工作在 MODE2 时，传输时钟只能选择 PCLK，波特率被固定在如下公式所得值：

$$\text{BaudRate} = \frac{(\text{LPUART\_SCON.DBAUD} + 1) * \text{F}_{\text{pclk}}}{64}$$

其中，LPUART_SCON.DBAUD 表示双倍波特率，F_{pclk} 为 PCLK 时钟频率。

LPMODE=1

当 LPMODE=1 时，不支持该模式。

23.5 帧错误检测

MODE1/2/3 具有帧错误检测功能，硬件会自动检测接收到的帧数据是否带有效的 STOP 位。如果没有收到有效 STOP 位，则 LPUART_INTSR.FE 置 1。LPUART_INTSR.FE 位由硬件置 1，软件清 0，如果软件未及时清 0，则后续收到数据即使带有效 STOP 位，也不会把 LPUART_INTSR.FE 标志清 0。

23.6 多机通讯

MODE2/3 具有多机通讯功能，为此在其帧格式中增加了 1 位 TB8/RB8。将 LPUART_SCON.SM2 置“1”，可开启多机通讯位。当开启多机通讯位后，发送数据时，主机可以通 LPUART_SCON.TB8 来区分当前帧是地址帧(LPUART_SCON.TB8=1)还是数据帧(LPUART_SCON.TB8=0)。接收数据时，从机会忽略 RB8 位(第 9 位)为“0”的当前接收帧。当收到帧的 RB8 位(第 9 位)为“1”表明其是地址帧，从机会继续判断接收到的地址与其自身地址是否相等。如果匹配，则从机会对 LPUART_SCON.RB8 置“1”，并对 LPUART_INTSR.RI 置“1”，以表明该帧为地址帧并且地址已经匹配。从机软件看到 LPUART_SCON.RB8=1 并且 LPUART_INTSR.RI=1 后，先把 LPUART_SCON.SM2 位清“0”，然后准备接受给它的数据帧。如果地址不等，表明主机并不是寻址该从机，从机硬件保持 LPUART_SCON.RB8 和 LPUART_INTSR.RI 为“0”，软件保持 LPUART_SCON.SM2 位为“1”，从机继续处于地址监听状态。

23.7 自动地址识别

当开启多机通讯位后(LPUART_SCON.SM2 置“1”)，自动地址识别功能也将开启。该功能由硬件实现，使得从机可以检测接收到每个地址帧，如果该地址与从机地址匹配，接收端会给出 LPUART_INTSR.RI 接收标志。如果地址不匹配，则接收端不会给出任何接收标志。

如果有需要，也可以在 MODE1 下开启多机通讯位，此时 TB8 位由 STOP 位代替。当从机接收到匹配的地址帧和有效的 STOP 位时，LPUART_INTSR.RI 会被置“1”。为了支持自动地址识别，定义了广播地址和给定地址的概念。

23.8 给定地址

LPUART 设备的 LPUART_SADDR 寄存器用来表示自己的设备给定地址，LPUART_SADEN 寄存器是地址掩码，可以用来定义地址中的无关位。当 LPUART_SADEN 的某一位为“0”，表示该位地址为无关位，也就是说在地址匹配过程中，该位地址不参与地址匹配。这些无关位增加了寻址的灵活性，使得主机可以同时寻址一个或者多个从机设备。注意，如果需要给出唯一匹配地址，LPUART_SADEN 寄存器必须设为 8’HFF。

GIVENADDR = SADDR & SADEN

23.9 广播地址

广播地址是用来同时寻址所有从机设备的，一般广播地址为 8’HFF。

BoardCastAddr = SADDR | SADEN

23.10 举例

假设某从机的 LPUART_SADDR 和 LPUART_SADEN 配置如下：

SADDR：0b01101001

SADEN：0b11111011

那么其给定地址和广播地址如下：

Given：0b01101X01

BroadCast：0b11111X11

可见，主机可以用四个地址寻址到本从机，分别是：

0b01101001 和 0b01101101(given address)

0b11111011 和 0b11111111(broadcast address)。

23.11 收发端缓存

23.11.1 接收缓存

LPUART 接收端有一个帧长度(8/9BITS)的接收缓存，也就是说当一帧数据接收完毕后，接收缓存中的数据会被一直保持，直到下一帧数据的 STOP 位接收完毕后，接收缓存才会更新为新一帧数据。

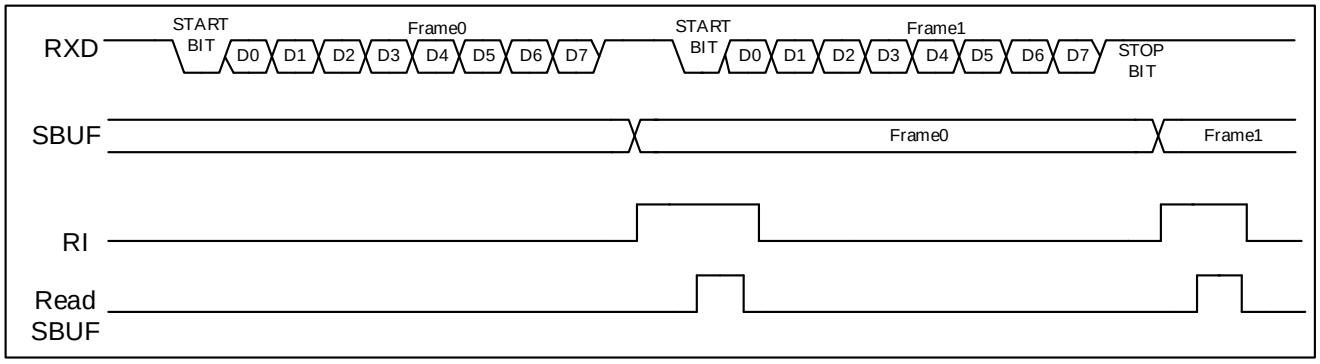


图 23-8 接收缓存

23.11.2 发送缓存

LPUART 发送端不支持发送缓存。如果在发送数据过程中，填写 LPUART_SBUF 寄存器，将会屏蔽该写操作。软件应该避免这种操作。

23.12 寄存器列表

LPUART 基地址：0x4000_5000

偏移地址	名称	描述	复位值
0x00	LPUART_SCON	控制寄存器	0x0000 E000
0x04	LPUART_SBUF	数据寄存器	0x0000 0000
0x08	LPUART_SADDR	地址寄存器	0x0000 0000
0x0C	LPUART_SADEN	地址掩码寄存器	0x0000 0000
0x10	LPUART_INTSR	中断标志位寄存器	0x0000 0008
0x14	LPUART_INTCLR	中断标志位清除寄存器	0x0000 0000
0x18	LPUART_BAUDCR	波特率控制寄存器	0x0000 0000

23.13 寄存器说明

23.13.1 LPUART 控制寄存器(LPUART_SCON)

地址偏移：0x00

复位值：0x0000E000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留														FEEN	EN
														R/W	R/W

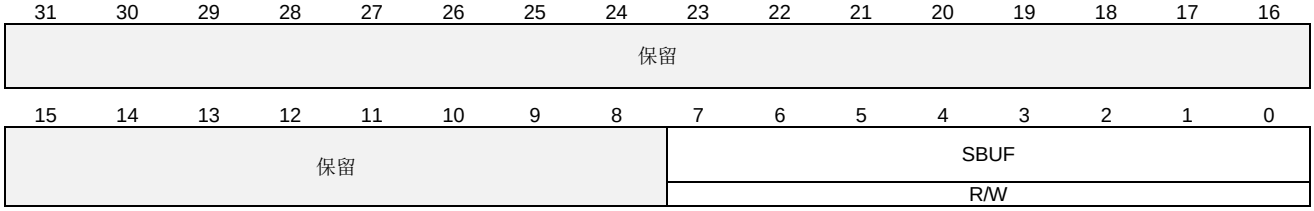
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRSC			SCLKSEL		LPMODE	DBAUD	TEEN	SM0_SM1		SM2	REN	TB8	RB8	TIEN	RIEN
R/W			R/W		R/W	R/W	R/W	R/W		R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:18	Res	保留	0x0	–
17	FEEN	接收错误帧中断使能 0：接收错误帧中断无效 1：接收错误帧中断使能	0x0	R/W
16	EN	Low-Power UART 工作使能， 0：Low-Power UART 关闭，不接收/发送数据。 1：Low-Power UART 使能，进行数据传输之前必须将该位置 1；	0x0	R/W
15:13	PRSC	传输时钟 SCLK 预分频选择； 000：DIV128； 001：DIV64； 010：DIV32； 011：DIV16； 100：DIV8； 101：DIV4； 110：DIV2； 111：DIV1； PRSC 只有当 LPMODE=1 时有效；当 LPMODE=0 时，PRSC 不会对 SCLK 预分频。	0x7	R/W
12:11	SCLKSEL	传输时钟 SCLK 选择； 0x：PCLK； 10：LXT； 11：LIRC;	0x0	R/W
10	LPMODE	低功耗模式； 0：正常工作模式； 1：低功耗工作模式	0x0	R/W
9	DBAUD	双倍波特率； 0：单倍波特率； 1：双倍波特率；	0x0	R/W
8	TEEN	发送缓存空中断使能； 0：发送缓存空中断无效； 1：发送缓存空中断使能；	0x0	R/W

<table>
<tr><td>7:6</td><td>SM0_SM1</td><td>工作模式：00：MODE0；01：MODE1；10：MODE2；11：MODE3
<table>
<tr><td>SM0</td><td>SM1</td><td>MODE</td><td>描述</td><td>波特率</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>位移寄存器</td><td>PCLK/12</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>8 位串口传输</td><td>可变波特率</td></tr>
<tr><td>1</td><td>0</td><td>2</td><td>9 位串口传输</td><td>PCLK/32,
PCLK/64</td></tr>
<tr><td>1</td><td>1</td><td>3</td><td>9 位串口传输</td><td>可变波特率</td></tr>
</table></td><td>0x0</td><td>R/W</td></tr>
<tr><td>5</td><td>SM2</td><td>多主机通讯；0：DISABLE，1：ENABLE
SM2：软件配置多机通讯以及自动地址匹配模式
1：启动多从机通讯以及地址自动匹配
0：关闭多从机通讯以及地址自动匹配
在模式 2 和模式 3 中：
如果 SM2=1，并且 REN=1，则接收机处于地址帧监测模式，可以使用接收到的第 9 位 RB8 来进行地址筛选。
RB8=1 为地址帧，通讯数据可以进入 SBUF，置位 RI，进入中断服务程序中进行地址比较；RB8=0 为数据帧，接收机忽略这些数据帧并保持 RI=0
如果 SM2=0，并且 REN=1，则接收机不使用地址监测模式，无论收到的 RB8 为 0 或 1，都直接接收并且进入 SUBF，置位 RI，RB8 在这种模式下为校验位。</td><td>0x0</td><td>R/W</td></tr>
<tr><td>4</td><td>REN</td><td>接收使能；
MODE0：0：发送，1：接收
其他：　0：发送，1：接收/发送</td><td>0x0</td><td>R/W</td></tr>
<tr><td>3</td><td>TB8</td><td>发送 TB8 位</td><td>0x0</td><td>R/W</td></tr>
<tr><td>2</td><td>RB8</td><td>接收 RB8 位</td><td>0x0</td><td>R/W</td></tr>
<tr><td>1</td><td>TIEN</td><td>发送完成中断使能；
0：发送完成中断使能无效；
1：发送完成中断使能；</td><td>0x0</td><td>R/W</td></tr>
<tr><td>0</td><td>RIEN</td><td>接收完成中断使能：
0：接收完成中断无效；
1：接收完成中断使能</td><td>0x0</td><td>R/W</td></tr>
</table>

23.13.2 LPUART 数据寄存器(LPUART_SBUF)

地址偏移：0x04
复位值：0x0000 0000



位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7:0	SBUF	发送数据时，当发送数据写入该寄存器；接收数据时，数据接收完毕后，从该寄存器中读出。	0x0	R/W

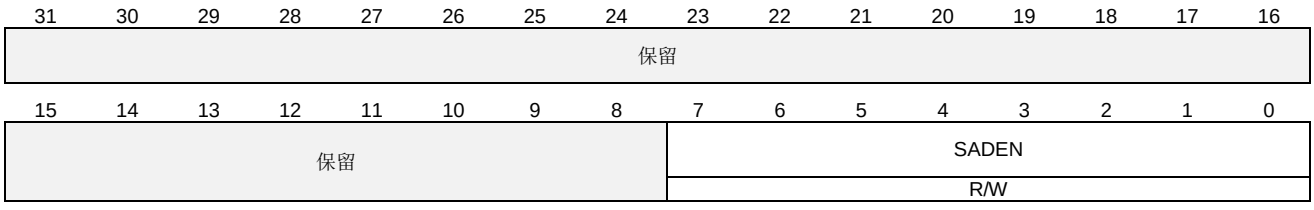
23.13.3 LPUART 地址寄存器(LPUART_SADDR)

地址偏移：0x08
复位值：0x0000 0000

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7:0	SADDR	从机设备地址寄存器	0x0	R/W

23.13.4 LPUART 地址掩码寄存器(LPUART_SADEN)

地址偏移：0x0C
复位值：0x0000 0000



位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7:0	SADEN	从机设备地址掩码寄存器	0x0	R/W

23.13.5 LPUART 标志位寄存器(LPUART_INTSR)

地址偏移：0x10
复位值： 复位值为 0X00000008

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留												TE	FE	TI	RI
												RO	RO	RO	RO

位	标记	功能描述	复位值	读写
31:4	Res	保留	0x0	–
3	TE	发送缓存空中断标志位，硬件置位，硬件清零。 注意：当该位值为“0”时，硬件自动屏蔽软件写 SBUF 操作。 1：TE 中断有效 0：TE 中断无效	0x1	RO
2	FE	接收帧错误标志位，硬件置位，软件清零 1：FE 中断有效 0：FE 中断无效	0x0	RO
1	TI	发送完成中断标志位，硬件置位，软件清零 1：TI 中断有效 0：TI 中断无效	0x0	RO
0	RI	接收完成中断标志位，硬件置位，软件清零 1：RI 中断有效 0：RI 中断无效	0x0	RO

23.13.6 LPUART 标志位清除寄存器(LPUART_INTCLR)

地址偏移：0x14
复位值：0x0000 0000

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

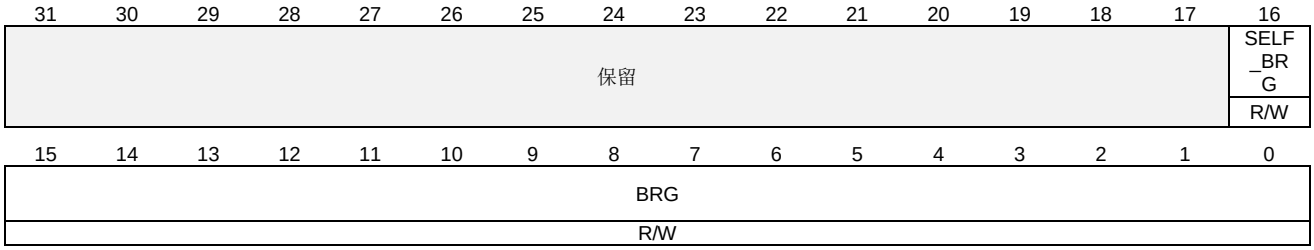
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留													FECLR	TICLR	RICLR
													WO	WO	WO

位	标记	功能描述	复位值	读写
31:3	Res	保留	0x0	–
2	FECLR	清除接收帧错误标志位；写 1 清零，写 0 无效	0x0	WO
1	TICLR	清除发送完成中断标志位；写 1 清零，写 0 无效	0x0	WO
0	RICLR	清除接收完成中断标志位；写 1 清零，写 0 无效	0x0	WO

23.13.7 LPUART 波特率控制寄存器(LPUART_BAUDCR)

地址偏移：0x18

复位值：0x0000 0000



位	标记	功能描述	复位值	读写
31:17	Res	保留	0x0	–
16	SELF_BRG	LPUART 波特率选择位： 0：LPUART 的波特率由 LPTIM 产生 1：LPUART 的波特率由(BAUD+1)*F_{sclk}/(32*(BRG+1))生成	0x0	R/W
15:0	BRG	LPUART 自动波特率生成配置位： 波特率=(BAUD+1)*F_{sclk}/(32*(BRG+1))	0x0	R/W

24 通用异步收发器(UART)

24.1 概述

本产品带有 2 个通用 UART 模块(UART0/1)，支持半双工和全双工传输；支持 8bit、9bit 数据格式；支持 Mode0/1/2/3 四种不同传输模式；UART0 的波特率可以由 TIM10 产生或者自动波特率发生器产生，UART1 的波特率可以由 TIM11 产生或者自动波特率发生器产生；支持多机通讯模式；支持自动地址识别；支持给定地址和广播地址。

通用 UART(UART0/1)只有一个时钟输入 PCLK，寄存器配置逻辑和数据收发逻辑都工作在该时钟域。

24.2 结构框图

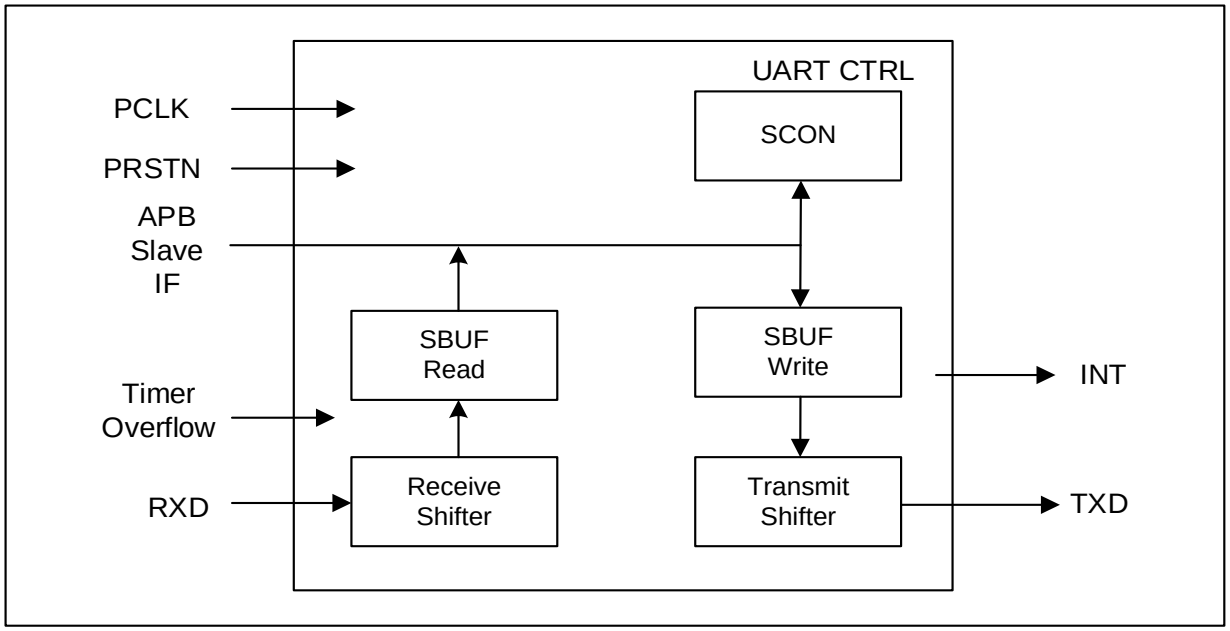


图 24-1 UART 结构图

24.3 工作模式

24.3.1 Mode0(同步模式，半双工)

当工作在 Mode0 时，UART 工作在同步模式，其波特率为固定的 PCLK 时钟的 1/12。UART 接收数据由 RXD 输入、UART 发送数据由 RXD 输出，RXD 此时为输入输出端口。UART 同步移位时钟由 TXD 输出，TXD 此时为输出端口。注意，本模式只能作为主机发送同步移位时钟，不可以作为从机从外部接收移位时钟。该模式下，传输的数据位宽只能是 8 位的，没有起始位和结束位。

将 UARTx_SCON.SM0 和 UARTx_SCON.SM1 清零，可进入 Mode0 工作模式。

24.3.1.1 发送数据

发送数据时，清除 UARTx_SCON.REN 位，并将数据写入 UARTx_SBUF 寄存器。此时，发送数据将从 RXD 输出(低位在先，高位在后)，同步移位时钟从 TXD 输出。

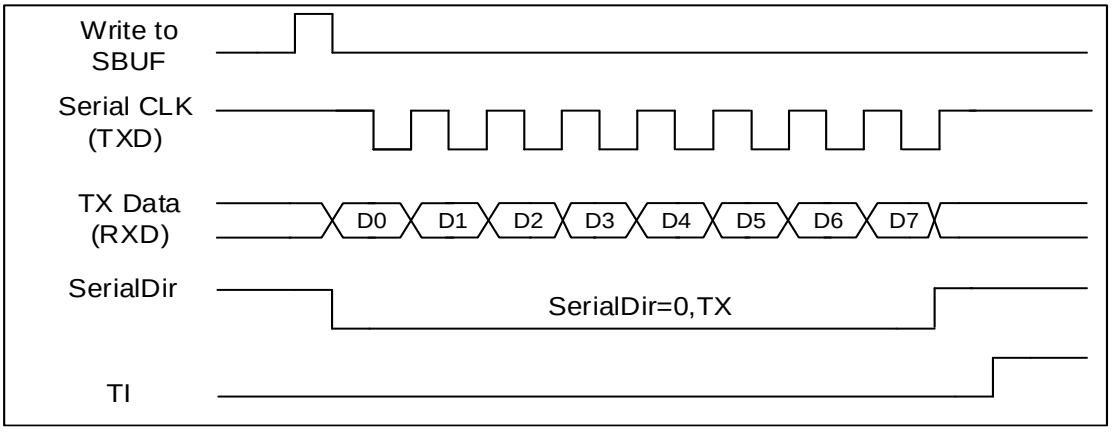


图 24-2 Mode0 发送数据

24.3.1.2 接收数据

接收数据时，将 UARTx_SCON.REN 位置 1，并将 UARTx_INTSR.RI 位清零。当接收结束，数据可从 UARTx_SBUF 寄存器读出。此时，接收数据从 RXD 输入(低位在先，高位在后)，同步移位时钟从 TXD 输出。

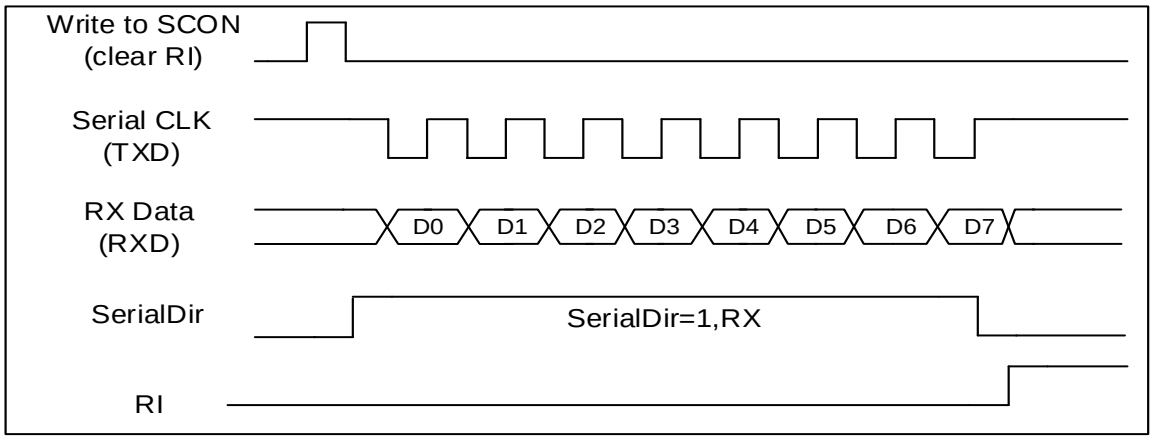


图 24-3 Mode0 接收数据

24.3.2 Mode1(异步模式，全双工)

当工作在 Mode1 时，发送数据通过 TXD 发送，接收数据通过 RXD 接收。该数据由 10 位组成：起始位“0”开始，紧接着 8 位数据位(低位在先，高位在后)，最后是结束位“1”。该模式下，波特率可以由可编程定时器模块产生，也可以由模块内部的自动波特率发生器产生。当 UARTx_BAUDCR.SELF_BRG 为 0 时，选择由定时器产生波特率时，UART0 的波特率由 TIM10 产生，UART1 的波特率由 TIM11 产生；当 UARTx_BAUDCR.SELF_BRG 置 1 时，UART0、UART1 的波特率都由各自内部的自动波特率发生器产生。波特率产生公式请参考 24.4.2 Mode1/3。

将 UARTx_SCON.SM0 清 0，UARTx_SCON.SM1 置 1，可进入 Mode1 工作模式。

24.3.2.1 发送数据

发送数据时，与 UARTx_SCON.REN 的值无关，将所发送数据写入 UARTx_SBUF 寄存器中，数据就会从 TXD 移出(低位在先，高位在后)。

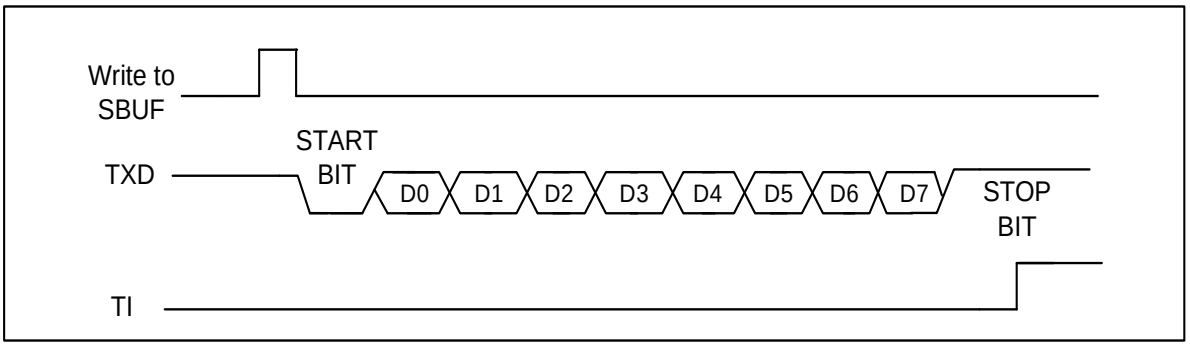


图 24-4：Mode1 发送数据

24.3.2.2 接收数据

接收数据时，需将 UARTx_SCON.REN 位置 1，并将 UARTx_INTSR.RI 位清 0。开始接收 RXD 上数据(低位在先，高位在后)，当接收完毕，可以从 UARTx_SBUF 寄存器读出。

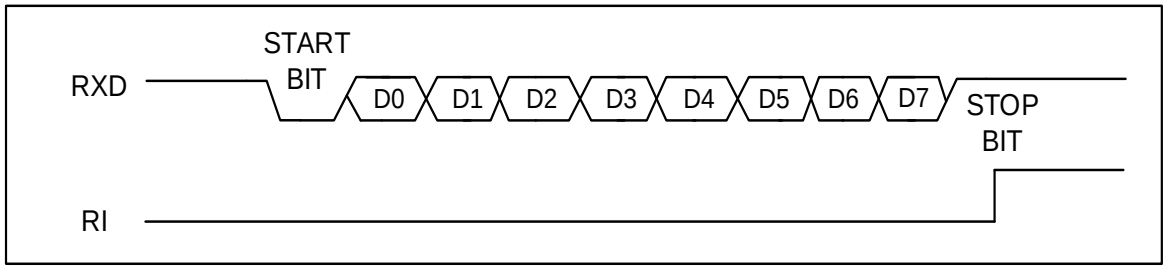


图 24-5：Mode1 接收数据

24.3.3 Mode2(异步模式，全双工)

当工作在 Mode2 时，发送数据通过 TXD 发送，接收数据通过 RXD 接收。该数据由 11 位组成：起始位“0”开始，接着是 8 个数据位，1 个 TB8 位和结束位。额外的 TB8 位是用来在多机通讯环境下使用，当 TB8=1，表明所接收的是地址帧；当 TB8=0，表明所接收的是数据帧。当不需要多机通讯时，此位也可以作为奇偶校验位来使用。该模式下，波特率可以独立产生，不需要外部定时器模块产生。

将 UARTx_SCON.SM0 置 1，UARTx_SCON.SM1 清 0，可进入 Mode2 工作模式。

24.3.3.1 发送数据

发送数据时，与 UARTx_SCON.REN 的值无关，并将所发送数据写入 UARTx_SBUF 寄存器中，数据就会从 TXD 移出(低位在先，高位在后)。

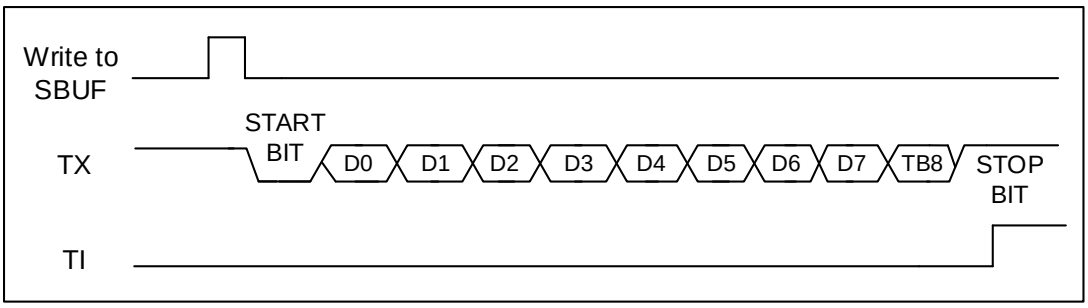


图 24-6：Mode2 发送数据

24.3.3.2 接收数据

接收数据时，需将 UARTx_SCON.REN 位置 1，并将 UARTx_INTSR.RI 位清 0。开始接收 RXD 上数据(低位在先，高位在后)，当接收完毕，可以从 UARTx_SBUF 寄存器读出。

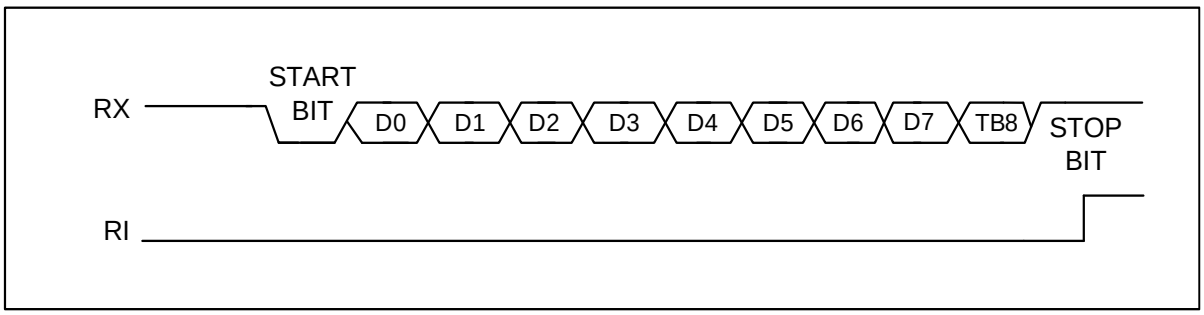


图 24-7：Mode2 接收数据

24.3.4 Mode3(异步模式，全双工)

Mode3 的数据格式，传输时序以及操作方式都与 Mode2 相同，唯一的区别是 Mode3 的波特率选择由可编程定时器产生或者内部自动波特率发生器产生，而不是像 Mode2 只由设备自己独立产生。Mode3 的波特率是可编程的，波特率生成方式与 Mode1 相同。

将 UARTx_SCON.SM0 置 1，UARTx_SCON.SM1 置 1，可进入 Mode3 工作模式。

24.4 波特率编程

24.4.1 Mode0

当工作在 Mode0 时，波特率被固定在 PCLK 的 1/12，不需要可编程定时器(TIMER)的支持。

24.4.2 Mode1/3

当工作在 Mode1 或者 Mode3 时，波特率产生公式如下图所示：

UARTx_BAUDCR.SELF_BRG=0,使用可编程定时器(TIMER)波特率模式：

$$\text{BaudRate} = \frac{(\text{UARTx\_SCON.DBAUD} + 1) * \text{F}_{\text{pclk}}}{32 * (2^{16} - TIMx\_BGLOAD)}$$

UARTx_BAUDCR.SELF_BRG=1,使用自身波特率生成模式：

$$\text{BaudRate} = \frac{(\text{UARTx\_SCON.DBAUD} + 1) * F_{\text{pclk}}}{32 * (\text{UARTx\_BAUDCR.BRG} + 1)}$$

其中，UARTx_SCON.DBAUD 表示双倍波特率，Fpclk 为 PCLK 时钟频率，TIMx_BGLOAD 为 TIMER 的周期装载计数值。注意，TIMER 必须配置为 16 位自动重载模式，立即重载寄存器(TIMx_LOAD)和周期重载寄存器(TIMx_BGLOAD)要写入相同的初始值。

24.4.3 Mode2

当工作在 Mode2 时，波特率被固定在如下公式所得值：

$$\text{BaudRate} = \frac{(\text{UARTx\_SCON.DBAUD} + 1) * \text{F}_{\text{pclk}}}{64}$$

其中，UARTx_SCON.DBAUD 表示双倍波特率，F_{pclk} 为 PCLK 时钟频率。

24.5 帧错误检测

Mode1/2/3 具有帧错误检测功能，硬件会自动检测接收到的帧数据是否带有效的 Stop 位。如果没有收到有效 Stop 位，则 UARTx_INTSR.FE 置 1。UARTx_INTSR.FE 位由硬件置 1，软件清 0，如果软件未及时清 0，则后续收到数据即使带有效 Stop 位，也不会把 UARTx_INTSR.FE 标志清 0。

24.6 多机通讯

Mode2/3 具有多机通讯功能，为此在其帧格式中增加了 1 位 TB8/RB8。将 UARTx_SCON.SM2 置“1”，可开启多机通讯位。当开启多机通讯位后，发送数据时，主机可以通过 UARTx_SCON.TB8 来区分当前帧是地址帧(UARTx_SCON.TB8=1)还是数据帧(UARTx_SCON.TB8=0)。接收数据时，从机忽略 RB8 位(第 9 位)为“0”的当前接收帧。当收到帧的 RB8 位(第 9 位)为“1”表明其是地址帧，从机继续判断接收到的地址与其自身地址是否相等。如果匹配，则从机对 UARTx_SCON.RB8 置“1”，并对 UARTx_INTSR.RI 置“1”，以表明该帧为地址帧并且地址已经匹配。从机软件看到 UARTx_SCON.RB8=1 并且 UARTx_INTSR.RI=1 后，先把 UARTx_SCON.SM2 位清“0”，然后准备接受给它的数据

帧。如果地址不等，表明主机并不是寻址该从机，从机硬件保持 UARTx_SCON.RB8 和 UARTx_INTSR.RI 为“0”，软件保持 UARTx_SCON.SM2 位为“1”，从机继续处于地址监听状态。

24.7 自动地址识别

当开启多机通讯位后(UARTx_SCON.SM2 置“1”)，自动地址识别功能也将开启。该功能由硬件实现，使得从机可以检测接收到每个地址帧，如果该地址与从机地址匹配，接收端会给出 UARTx_INTSR.RI 接收标志。如果地址不匹配，则接收端不会给出任何接收标志。如果有需要，也可以在 Mode1 下开启多机通讯位，此时 TB8 位由 Stop 位代替。当从机接收到匹配的地址帧和有效的 Stop 位时，UARTx_INTSR.RI 会被置“1”。为了支持自动地址识别，定义了广播地址和给定地址的概念。

24.8 给定地址

UART 设备的 UARTx_SADDR 寄存器用来表示自己的设备给定地址，UARTx_SADEN 寄存器是地址掩码，可以用来定义地址中的无关位。当 UARTx_SADEN 的某一位为“0”，表示该位地址为无关位，也就是说在地址匹配过程中，该位地址不参与地址匹配。这些无关位增加了寻址的灵活性，使得主机可以同时寻址一个或者多个从机设备。注意，如果需要给出唯一匹配地址，UARTx_SADEN 寄存器必须设为 8’hFF。

GivenAddr=SADDR&SADEN

24.9 广播地址

广播地址是用来同时寻址所有从机设备的，一般广播地址为 8’hFF。

BoardCastAddr = UARTx_SADDR |UARTx_SADEN

24.9.1 举例

假设某从机的 UARTx_SADDR 和 UARTx_SADEN 配置如下：

SADDR：0b01101001

SADEN：0b11111011

那么其给定地址和广播地址如下：

Given：0b01101x01

Broadcast：0b11111x11

可见，主机可以用四个地址寻址到本从机，分别是：

0b01101001 和 0b01101101(given address)

0b11111011 和 0b11111111(broadcast address)。

24.10 收发端缓存

24.10.1 接收缓存

通用 UART(UART0/1)接收端有一个帧长度(8/9bits)的接收缓存，也就是说当一帧数据接收完毕后，接收缓存中的数据会被一直保持，直到下一帧数据的 Stop 位接收完毕后，接收缓存才会更新为新一帧数据。

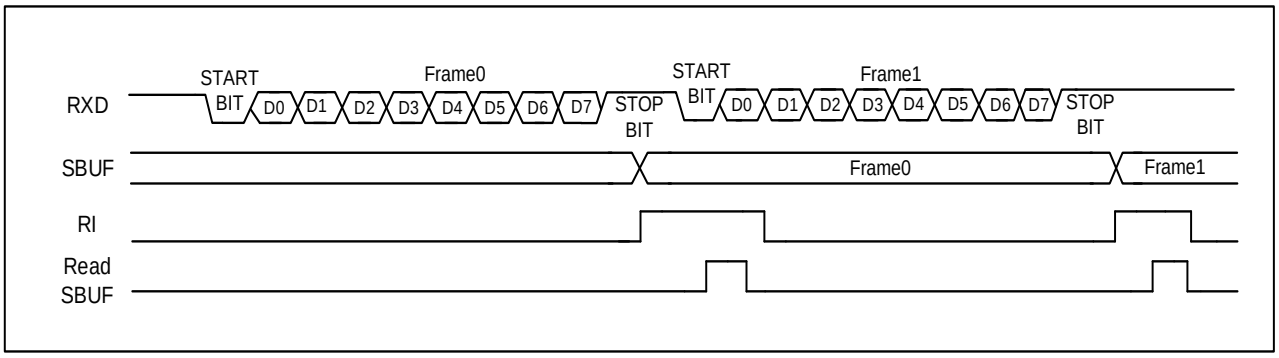


图 24-8：接收缓存

24.10.2 发送缓存

通用 UART(UART0/1)发送端不支持发送缓存。如果在发送数据过程中，填写 UARTx_SBUF 寄存器，将会破坏当前正在发送数据。软件应该避免这种操作。

24.11 IrDA 红外功能

IrDA SIR 物理层规定使用反相归零调制方案(RZI)，该方案用一个红外光脉冲代表逻辑'0'(见图 24-9IrDA 结构框图)。SIR 发送编码器对从 UART 输出的 NRZ(非归零)比特流进行调制。输出脉冲流被传送到一个外部输出驱动器和红外 LED。对于 SIRENDEC 应用，UART 最高只支持到 115.2Kbps 速率。在正常模式里，脉冲宽度规定为一个位周期的 3/16。SIR 接收解码器对来自红外接收机的归零位比特流进行解调，并将接收到的 NRZ 串行比特流输出到 UART。在空闲状态里，解码器输入通常是高(标记状态)。发送编码器输出的极性和解码器的输入相反。当解码器输入低时，检测到一个起始位。

- IrDA 是一个半双工通信协议。如果发送器忙(也就是 UART 正在送数据给 IrDA 编码器)，IrDA 接收线上的任何数据都将被 IrDA 解码器所忽略。如果接收机忙(也就是 UART 正在接收从 IrDA 解码器来的解码数据)，从 UART 的 TX 上到 IrDA 的数据将不会被 IrDA 编码。当接收数据时，应该避免发送，因为将被发送的数据可能被破坏。
- SIR 发送逻辑把'0'作为高脉冲发送，把'1'作为低电平发送；或者取反发送。脉冲的宽度规定为正常模式时位周期的 3/16(见图 24-10IrDA 收发脉冲)。
- SIR 解码器把接收到的 IrDA 信号转变成比特流后发送给 UART。
- SIR 接收逻辑把高电平状态解释为'0'，把低脉冲解释为'1'；或者取反接收。
- 发送编码器输出与解码器输入有着相反的极性。当空闲时，SIR 输出处于低状态。
- IrDA 规范要求脉冲要宽于 1.41us。脉冲宽度是可编程的。接收机端的尖峰脉冲检测电路会过对宽度小于 2 个 PSC 周期的脉冲进行过滤操作(PSC 是在 UARTx_IRDACR 中编程的预分频值)。宽度小于 1 个 PSC 周期的脉冲一定会被过滤掉，但是那些宽度大于 1 个而小于 2 个 PSC 周期的脉冲可能被接收或滤除，那些宽度大于 2 个周期的将被视为一个有效的脉冲。
- 接收机可以与一低功耗发送器通信。

24.11.1 IrDA 低功耗模式

IrDA 可以工作在正常模式，也可以工作在低功耗模式。选择低功耗模式需要把 UART_IRDACR.IRLPMODE 寄存器置 1。

- 发送器

在低功耗模式，脉冲宽度不再持续 3/16 个位周期。取而代之，脉冲的宽度是低功耗波特率时钟周期的 3 倍，该波特率的频率最小可以是 1.42MHz。通常这个值是 1.8432MHz(1.42MHz<PSC<2.12MHz)

一个低功耗模式可编程分频器把系统时钟进行分频以达到这个值。

- 接收机

低功耗模式的接收类似于正常模式的接收。为了滤除尖峰干扰脉冲，UART 应该滤除宽度短于 1 个周期的脉冲。只有持续时间大于 2 个周期的 IrDA 低功耗波特率时钟(UART_IRDACR 中的 PSC)的低电平信号才被接受为有效的信号。

注意：

1. 宽度小于 2 个大于 1 个 PSC 周期的脉冲可能会也可能不会被滤除。
2. 接收机的建立时间应该由软件管理。IrDA 物理层技术规范规定了在发送和接收之间最小要有 10ms 的延时(IrDA 是一个半双工协议)。

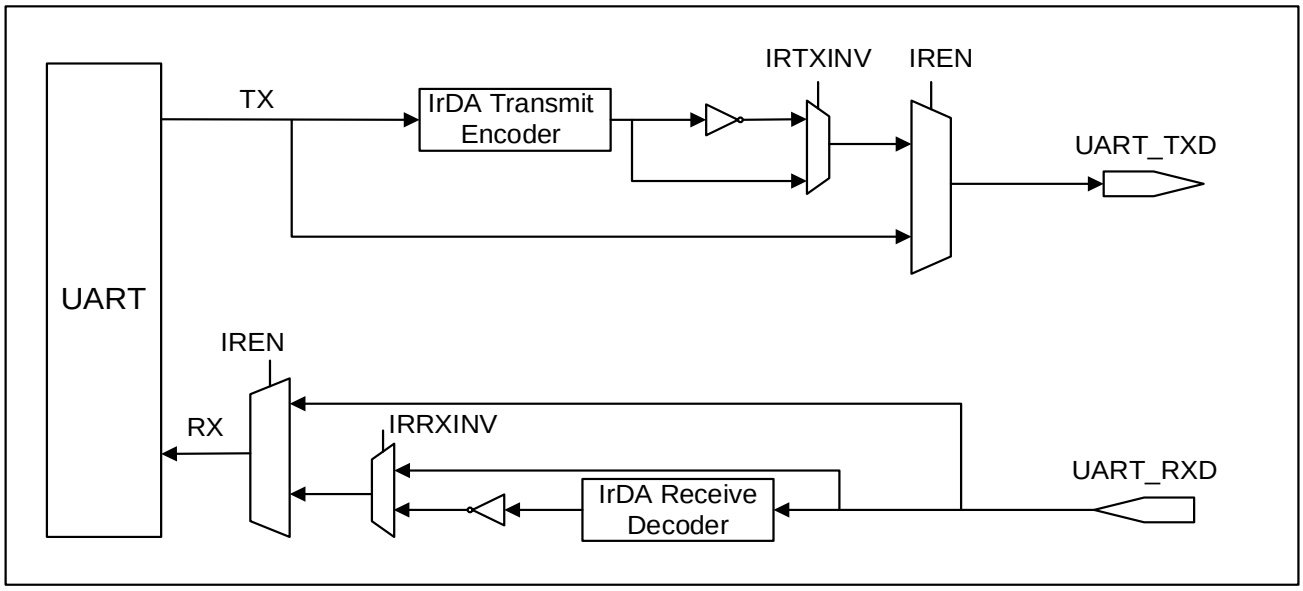


图 24-9IrDA 结构框图

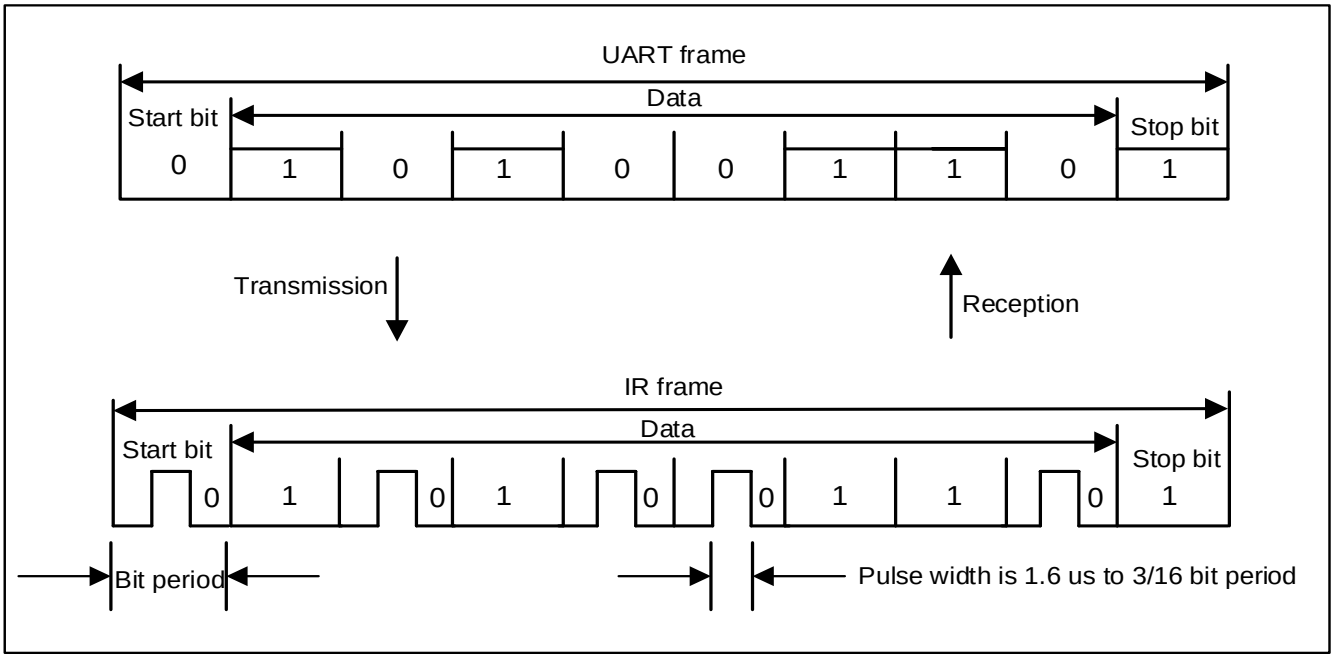


图 24-10IrDA 收发脉冲

24.12 不同波特率的分频设置

波特率	PCLK = 1 MHz					
	双波特率			单波特率		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	26	2403.85	0.16%	13	2403.85	0.16%
4800	13	4807.69	0.16%	7	4464.29	-6.99%
9600	7	8928.57	-6.99%	3	10416.67	8.51%
19200	3	20833.33	8.51%	2	15625.00	-18.62%
38400	2	31250.00	-18.62%	1	31250.00	-18.62%
57600	1	62500.00	8.51%	1	31250.00	-45.75%
76800	1	62500.00	-18.62%	0	-	-
115200	1	62500.00	-45.75%	0	-	-

波特率	PCLK =4 MHz					
	双波特率			单波特率		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	104	2403.85	0.16%	52	2403.85	0.16%
4800	52	4807.69	0.16%	26	4807.69	0.16%
9600	26	9615.38	0.16%	13	9615.38	0.16%
19200	13	19230.77	0.16%	7	17857.14	-6.99%
38400	7	35714.29	-6.99%	3	41666.67	8.51%
57600	4	62500.00	8.51%	2	62500.00	8.51%
76800	3	83333.33	8.51%	2	62500.00	-18.62%
115200	2	125000.00	8.51%	1	125000.00	8.51%

波特率	PCLK = 10 MHz					
	双波特率			单波特率		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	260	2403.85	0.16%	130	2403.85	0.16%
4800	130	4807.69	0.16%	65	4807.69	0.16%
9600	65	9615.38	0.16%	33	9469.70	-1.36%
19200	33	18939.39	-1.36%	16	19531.25	1.73%
38400	16	39062.50	1.73%	8	39062.50	1.73%
57600	11	56818.18	-1.36%	5	62500.00	8.51%
76800	8	78125.00	1.73%	4	78125.00	1.73%
115200	5	125000.00	8.51%	3	104166.67	-9.58%

波特率	PCLK = 14 MHz					
	双波特率			单波特率		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	365	2397.26	-0.11%	182	2403.85	0.16%
4800	182	4807.69	0.16%	91	4807.69	0.16%
9600	91	9615.38	0.16%	46	9510.87	-0.93%
19200	46	19021.74	-0.93%	23	19021.74	-0.93%
38400	23	38043.48	-0.93%	11	39772.73	3.57%
57600	15	58333.33	1.27%	8	54687.50	-5.06%
76800	11	79545.45	3.57%	6	72916.67	-5.06%
115200	8	109375.00	-5.06%	4	109375.00	-5.06%

波特率	PCLK = 20 MHz					
	双波特率			单波特率		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	521	2399.23	-0.03%	260	2403.85	0.16%
4800	260	4807.69	0.16%	130	4807.69	0.16%
9600	130	9615.38	0.16%	65	9615.38	0.16%
19200	65	19230.77	0.16%	33	18939.39	-1.36%
38400	33	37878.79	-1.36%	16	39062.50	1.73%
57600	22	56818.18	-1.36%	11	56818.18	-1.36%
76800	16	78125.00	1.73%	8	78125.00	1.73%
115200	11	113636.36	-1.36%	5	125000.00	8.51%

波特率	PCLK = 24 MHz					
	双波特率			单波特率		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	625	2400.00	0.00%	313	2396.17	-0.16%
4800	313	4792.33	-0.16%	156	4807.69	0.16%
9600	156	9615.38	0.16%	78	9615.38	0.16%
19200	78	19230.77	0.16%	39	19230.77	0.16%
38400	39	38461.54	0.16%	20	37500.00	-2.34%
57600	26	57692.31	0.16%	13	57692.31	0.16%
76800	20	75000.00	-2.34%	10	75000.00	-2.34%
115200	13	115384.62	0.16%	7	107142.86	-6.99%

波特率	PCLK = 2 MHz					
	双波特率			单波特率		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	52	2403.85	0.16%	26	2403.85	0.16%
4800	26	4807.69	0.16%	13	4807.69	0.16%
9600	13	9615.38	0.16%	7	8928.57	-6.99%
19200	7	17857.14	-6.99%	3	20833.33	8.51%
38400	3	41666.67	8.51%	2	31250.00	-18.62%
57600	2	62500.00	8.51%	1	62500.00	8.51%
76800	2	62500.00	-18.62%	1	62500.00	-18.62%
115200	1	125000.00	8.51%	1	62500.00	-45.75%

波特率	PCLK = 8 MHz					
	双波特率			单波特率		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	208	2403.85	0.16%	104	2403.85	0.16%
4800	104	4807.69	0.16%	52	4807.69	0.16%
9600	52	9615.38	0.16%	26	9615.38	0.16%
19200	26	19230.77	0.16%	13	19230.77	0.16%
38400	13	38461.54	0.16%	7	35714.29	-6.99%
57600	9	55555.56	-3.55%	4	62500.00	8.51%
76800	7	71428.57	-6.99%	3	83333.33	8.51%
115200	4	125000.00	8.51%	2	125000.00	8.51%

波特率	PCLK = 11.0592 MHz					
	双波特率			单波特率		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	288	2400.00	0.00%	144	2400.00	0.00%
4800	144	4800.00	0.00%	72	4800.00	0.00%
9600	72	9600.00	0.00%	36	9600.00	0.00%
19200	36	19200.00	0.00%	18	19200.00	0.00%
38400	18	38400.00	0.00%	9	38400.00	0.00%
57600	12	57600.00	0.00%	6	57600.00	0.00%
76800	9	76800.00	0.00%	5	69120.00	-10.00%
115200	6	115200.00	0.00%	3	115200.00	0.00%

波特率	PCLK = 16 MHz					
	双波特率			单波特率		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	417	2398.08	-0.08%	208	2403.85	0.16%
4800	208	4807.69	0.16%	104	4807.69	0.16%
9600	104	9615.38	0.16%	52	9615.38	0.16%
19200	52	19230.77	0.16%	26	19230.77	0.16%
38400	26	38461.54	0.16%	13	38461.54	0.16%
57600	17	58823.53	2.12%	9	55555.56	-3.55%
76800	13	76923.08	0.16%	7	71428.57	-6.99%
115200	9	111111.11	-3.55%	4	125000.00	8.51%

波特率	PCLK = 22.12 MHz					
	双波特率			单波特率		
	CNT	实际波特率	误差%	CNT	实际波特率	误差%
2400	576	2400.17	0.01%	288	2400.17	0.01%
4800	288	4800.35	0.01%	144	4800.35	0.01%
9600	144	9600.69	0.01%	72	9600.69	0.01%
19200	72	19201.39	0.01%	36	19201.39	0.01%
38400	36	38402.78	0.01%	18	38402.78	0.01%
57600	24	57604.17	0.01%	12	57604.17	0.01%
76800	18	76805.56	0.01%	9	76805.56	0.01%
115200	12	115208.33	0.01%	6	115208.33	0.01%

24.13 UART 寄存器列表

UART0 基地址：0x40000000

UART1 基地址：0x40000400

偏移地址	名称	描述	复位值
0x00	UARTx_SCON	控制寄存器	0x00000000
0x04	UARTx_SBUF	数据寄存器	0x00000000
0x08	UARTx_SADDR	地址寄存器	0x00000000
0x0C	UARTx_SADEN	地址掩码寄存器	0x00000000
0x10	UARTx_INTSR	中断标志位寄存器	0x00000008
0x14	UARTx_INTCLR	中断标志位清除寄存器	0x00000000
0x18	UARTx_BAUDCR	波特率控制寄存器	0x00000000
0x1C	UARTx_IRDACR	IrDA 控制寄存器	0x00000000

24.14 UART 寄存器说明

24.14.1 UART 控制寄存器(UARTx_SCON)

地址偏移：0x00

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				TXOVFEN	TXEEN	DBAUD	FEEN	SM0_SM1		SM2	REN	TB8	RB8	TIEN	RIEN
				R/W	R/W	R/W	R/W	R/W		R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:12	Res	保留	0x0	-
11	TXOVFEN	发送缓存满中断使能 0：发送缓存满中断无效 1：发送缓存满中断使能	0x0	R/W
10	TXEEN	发送缓存空中断使能 0：发送缓存空中断无效 1：发送缓存空中断使能	0x0	R/W
9	DBAUD	双倍波特率； 0：单倍波特率； 1：双倍波特率	0x0	R/W
8	FEEN	接收帧错误中断使能： 0：接收帧错误中断无效； 1：接收帧错误中断使能；	0x0	R/W
7:6	SM0_SM1	工作模式: 00：mode0； 01：mode1； 10：mode2； 11：mode3	0x0	R/W

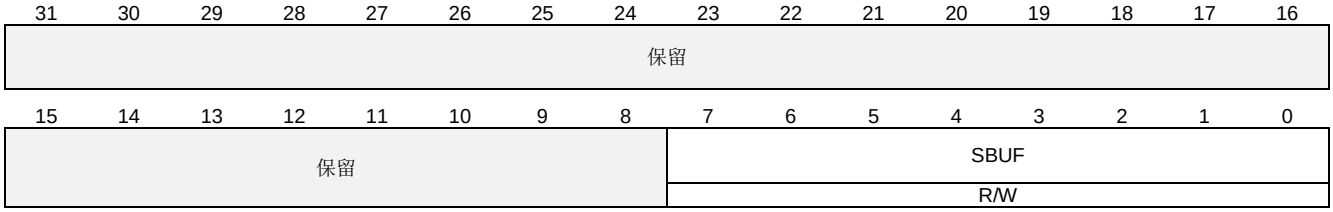
SM0	SM1	MODE	描述	波特率
0	0	0	位移寄存器	CLK/12
0	1	1	8 位串口传输	可变波特率
1	0	2	9 位串口传输	CLK/32, CLK/64
1	1	3	9 位串口传输	可变波特率

5	SM2	多主机通讯 SM2： 软件配置多机通讯以及自动地址匹配模式 1：启动多从机通讯以及地址自动匹配 0：关闭多从机通讯以及地址自动匹配 在模式 2 和模式 3 中： 如果 SM2=1，并且 REN=1，则接收机处于地址帧监测模式，可以使用接收到的第 9 位 RB8 来进行地址筛选。 RB8=1 为地址帧，通讯数据可以进入 SBUF，置位 RI，进入中断服务程序中进行地址比较；RB8=0 为数据帧，接收机忽略这	0x0	R/W

		些数据帧并保持 RI=0 如果 SM2=0，并且 REN=1，则接收机不使用地址监测模式，无论收到的 RB8 为 0 或 1，都直接接收并且进入 SUBF，置位 RI，RB8 在这种模式下为校验位。		
4	REN	接收使能； mode0： 0：发送，1：接收 其他：0：发送，1：接收/发送	0x0	R/W
3	TB8	发送 TB8 位	0x0	R/W
2	RB8	接收 RB8 位	0x0	R/W
1	TIEN	发送完成中断使能； 0：发送完成中断关闭 1：发送完成中断使能	0x0	R/W
0	RIEN	接收完成中断使能； 0：接收完成中断关闭 1：接收完成中断使能	0x0	R/W

24.14.2 UART 数据寄存器(UARTx_SBUF)

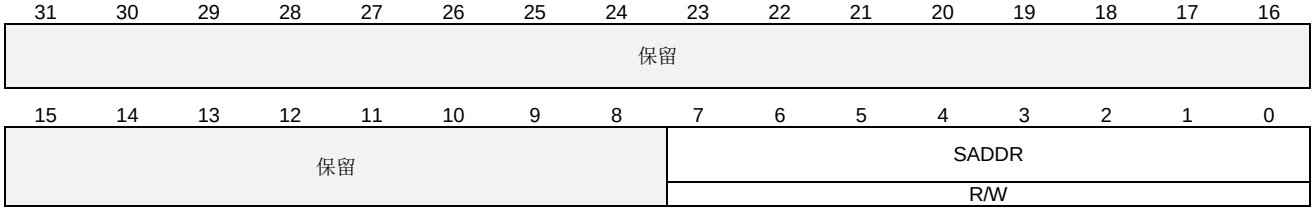
地址偏移：0x04
复位值：0x00000000



位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	-
7:0	SBUF	发送数据时，当发送数据写入该寄存器；接收数据时，数据接收完毕后，从该寄存器中读出。 注意，对该寄存器读的值实际是 RXBuffer 中的值，对该寄存器写的值实际是写到了 TXShifter 中。	0x0	R/W

24.14.3 UART 地址寄存器(UARTx_SADDR)

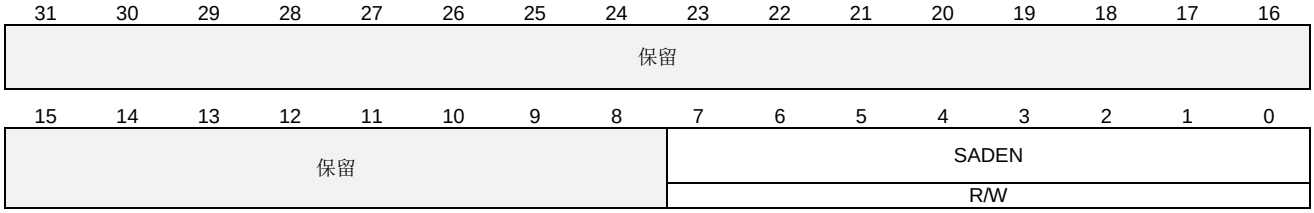
地址偏移：0x08
复位值：0x00000000



位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	-
7:0	SADDR	从机设备地址寄存器	0x0	R/W

24.14.4 UART 地址掩码寄存器(UARTx_SADEN)

地址偏移：0x0C
复位值：0x00000000



位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	-
7:0	SADEN	从机设备地址掩码寄存器	0x0	R/W

24.14.5 UART 标志位寄存器(UARTx_INTSR)

地址偏移：0x10
复位值：0x00000008

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留											TXOVF	TE	FE	TI	RI
											RO	RO	RO	RO	RO

位	标记	功能描述	复位值	读写
31:5	Res	保留	0x0	-
4	TXOVF	发送缓存满中断标志位，硬件置位，软件清零 0：TXOVF 标志无效 1：TXOVF 标志有效	0x0	RO
3	TE	发送缓存空中断标志位，硬件置位，硬件清零 0：TE 中断无效 1：TE 中断有效	0x1	RO
2	FE	接收帧错误标志位，硬件置位，软件清零 0：FE 中断无效 1：FE 中断有效 注：此位只在 UARTx_SCON.FEEN 使能的情况下有效；	0x0	RO
1	TI	发送完成中断标志位，硬件置位，软件清零 0：TI 中断无效 1：TI 中断有效 注：此位只在 UARTx_SCON.TIEN 使能的情况下有效；	0x0	RO
0	RI	接收完成中断标志位，硬件置位，软件清零 0：RI 中断无效 1：RI 中断有效 注：此位只在 UARTx_SCON.RIEN 使能的情况下有效；	0x0	RO

24.14.6 UART 标志位清除寄存器(UARTx_INTCLR)

地址偏移：0x14
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留											TXOVCLR	保留	FECLR	TICLR	RICLR
											WO		WO	WO	WO

位	标记	功能描述	复位值	读写
31:5	Res	保留	0x0	-
4	TXOVCLR	清除发送缓存满中断标志位；写 1 清零，写 0 无效	0x0	WO
3	Res	保留	0x0	-
2	FECLR	清除接收帧错误标志位；写 1 清零，写 0 无效	0x0	WO
1	TICLR	清除发送完成中断标志位；写 1 清零，写 0 无效	0x0	WO
0	RICLR	清除接收完成中断标志位；写 1 清零，写 0 无效	0x0	WO

24.14.7 UART 波特率控制寄存器(UARTx_BAUDCR)

地址偏移：0x18
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															SELF_BRG
															R/W

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

位	标记	功能描述	复位值	读写
31:17	Res	保留	0x0	-
16	SELF_BRG	UART 波特率选择位： 0：UART 的波特率由 timer 产生 1：UART 的波特率由 (BAUD+1)*F_{pclk}/(32*(BRG+1))生成 备注：UART0 由 TIM10 产生，UART1 由 TIM11 产生；	0x0	R/W
15:0	BRG	UART 自动波特率生成配置位： 波特率 = (BAUD+1)*F_{pclk}/(32*(BRG+1))	0x0	R/W

24.14.8 IrDA 控制寄存器(UARTx_IRDACR)

地址偏移：0x1C

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				IRLPMOD	IRRXINV	IRTXINV	IREN	PSC							
				R/W	R/W	R/W	R/W	R/W							

位	标记	功能描述	复位值	读写
31:12	Res	保留	0x0	-
11	IRLPMODE	Ir 低功耗模式： 0：Ir 普通模式 1：Ir 低功耗模式	0x0	R/W
10	IRRXINV	IrRXD 数据反转位 0：IrRXD 数据不反转 1：IrRXD 数据反转输出	0x0	R/W
9	IRTXINV	IrTXD 数据反转位 0：IrTXD 数据不反转 1：IrTXD 数据反转输出	0x0	R/W
8	IREN	IrDA 使能位 0：IrDA 无效 1：IrDA 使能	0x0	R/W
7:0	PSC	PSC 红外模式发送，接收模式滤波分频 对系统时钟分频已达到低功耗的频率	0x0	R/W

25 I2C 接口

25.1 I2C 简介

I2C 是双线双向的串行总线，它为设备之间数据交换提供了一种简单高效的方法。I2C 标准是一个具有冲突检测机制和仲裁机制的真正意义上的多主机总线。它能防止两个或者多个主机在同时请求控制总线时发生数据冲突。

I2C 总线控制器，能满足 I2C 总线的各种规格并支持所有与 I2C 总线通信的传输模式。

I2C 总线使用连接设备的”SCL”(串行时钟总线)和”SDA”(串行数据总线)来传送信息。数据在主机与从机之间通过 SCL 时钟线控制在 SDA 数据线上实现一个字节一个字节的同步传输，每个字节为 8 位长度，一个 SCL 时钟脉冲传输一个数据位，数据由最高位 MSB 开始传输，每个传输字节后跟随一个应答位，每个位在 SCL 为高时采样；因此，SDA 线只有在 SCL 为低时才可以改变，在 SCL 为高时 SDA 保持稳定。当 SCL 为高时，SDA 线上的跳变视为命令中断(START 或 STOP)，I2C 逻辑能自主地处理字节的传输。它能保持跟踪串行传送，而且还有一个状态寄存器(I2C_STAT)能反映 I2C 总线控制器和 I2C 总线的状态。

25.2 I2C 主要特性

I2C 控制器支持以下特性：

- 支持主机发送/接收，从机发送/接收四种工作模式
- 支持标准(100Kbps)/快速(400Kbps)/高速(1Mbps)三种工作速率
- 支持 7 位寻址功能
- 支持噪声过滤功能
- 支持广播地址
- 支持中断状态查询功能

25.3 I2C 协议描述

通常标准 I2C 传输协议包含四个部分：

1) 起始信号或重复起始信号
2) 从机地址传输和 R/W 位传输
3) 数据传输
4) 停止信号

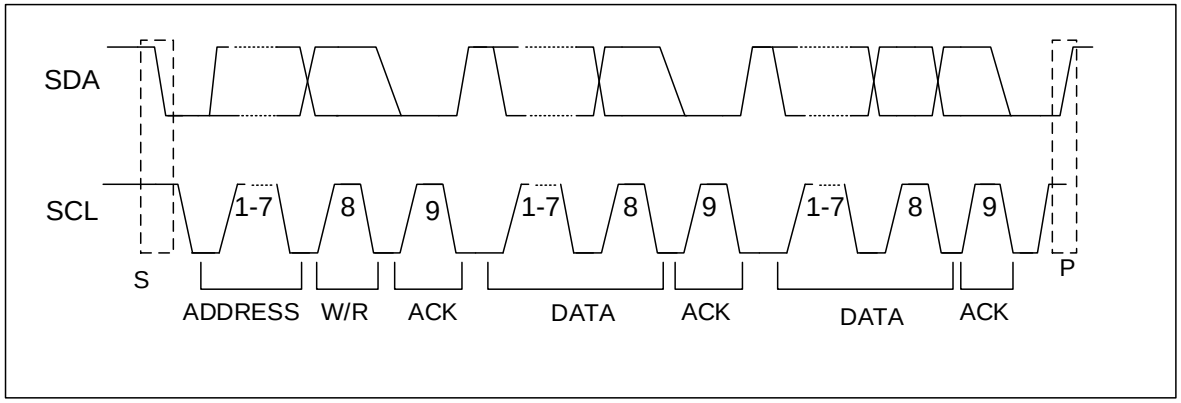


图 25-1 I2C 传输协议

25.3.1 I2C 总线上数据传输

a) 主机发出从机接收 7 位地址(一个字节)，传输方向未改变。

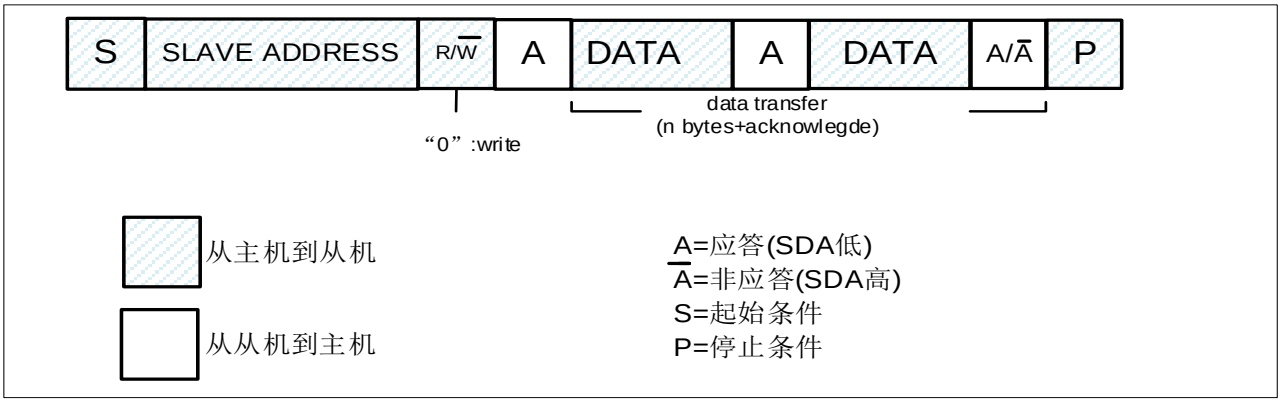


图 25-2 主机向从机传输数据

第一个字节后主机紧接着由从机读取数据(内容为从机地址)，传输方向改变。

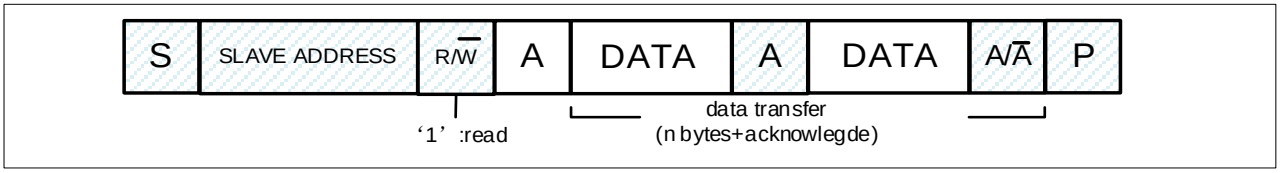


图 25-3 主机由从机读取地址

25.3.2 起始位或重复起始信号

当总线处于空闲状态下，说明没有主机对总线发起传输请求(SCL 和 SDA 线同时为高)，主机可以通过发送一个 START 信号来发起传输请求。

起始信号：通常表示为 S-bit，当 SCL 线为高时，SDA 线上信号由高至低，标示总线上产生起始信号，新的传输开始。

重复起始信号(Sr)：即在两个 START 信号之间没有 STOP 信号。主机采用这种方法与另一个从机或相同的从机以不同传输方向进行通信(例如：从写入设备到从设备读出) 而不释放总线。

STOP 信号：主机向总线发出停止信号结束数据传送。停止信号，通常用 P-bit 表示，当 SCL 线为高时，SDA 线上出现由低到高的信号，被定义为停止信号。

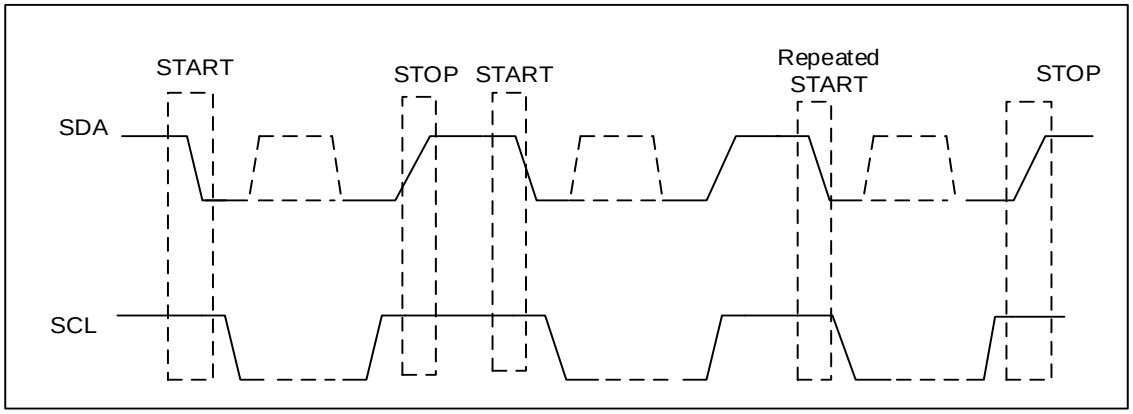


图 25-4 START 和 STOP 条件

25.3.3 从机地址传输

START 信号是从机地址时，主机立即传输数据的第一位。这是一个跟随有一个 RW 位的 7 位调用地址，RW 位控制从机的信号传输方向。系统中没有两个从机有相同的地址，只有被主机寻址的从机会通过在第 9 个 SCL 时钟周期将 SDA 置为低电平作为应答。

25.3.4 数据传输

当从机地址被成功识别，就可以根据 RW 所决定的方向，开始一字节一字节的数据传输，每个传输字节最后带一个第 9 时钟周期上的响应信号，如果从机上产生无响应信号(ACK)，主机可以产生停止信号来退出数据传输，或者产生重复起始信号开始新一轮的数据传输。

当主机作为接收器件时，发生无响应信号(ACK)，从机释放 SDA 线，使主机产生停止信号或重复起始信号。

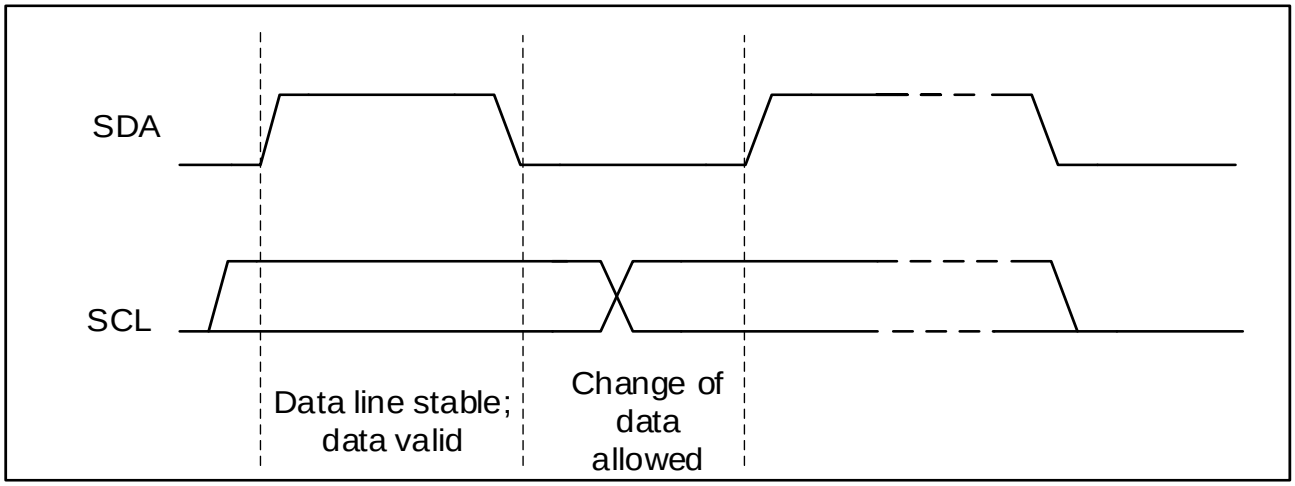


图 25-5 I2C 总线上位传输

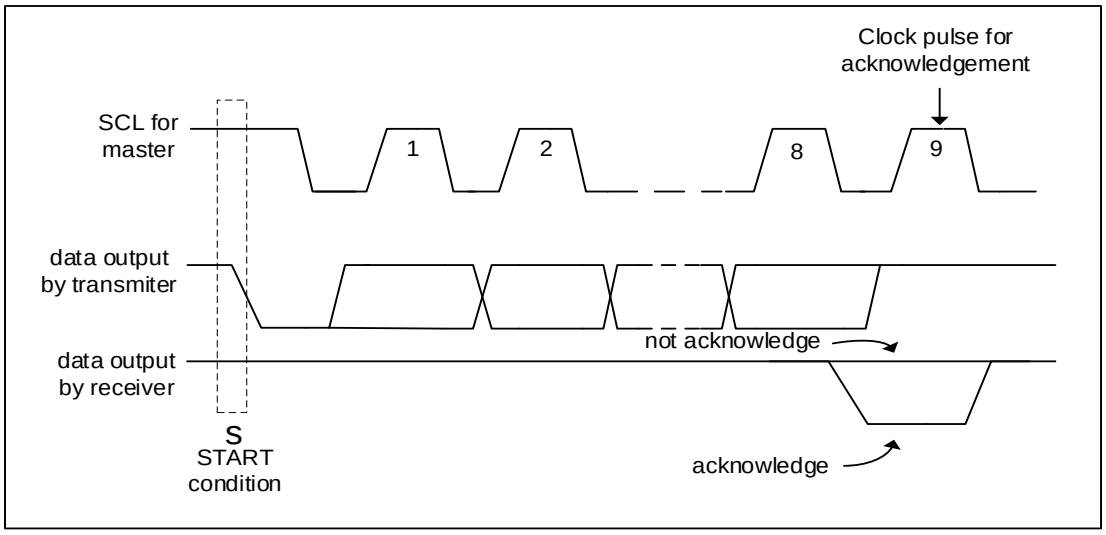


图 25-6 I2C 总线上应答信号

25.4 I2C 功能描述

I2C 总线使用双线在连接到总线“SCL”(串行时钟线)和“SDA”(串行数据线)的设备间传送信息。由于只有无方向端口，I2C 组件需要使用到引脚的漏端开路缓冲器。每个连接到总线的设备都能使用软件通过特定地址寻址。I2C 标准是一个具有冲突检测机制和仲裁机制的真正意义上的多主机总线。它能防止两个或者多个主机在同时开始传输数据时发生数据冲突。滤波逻辑可以过滤数据总线上的毛刺来保护数据的完整性。

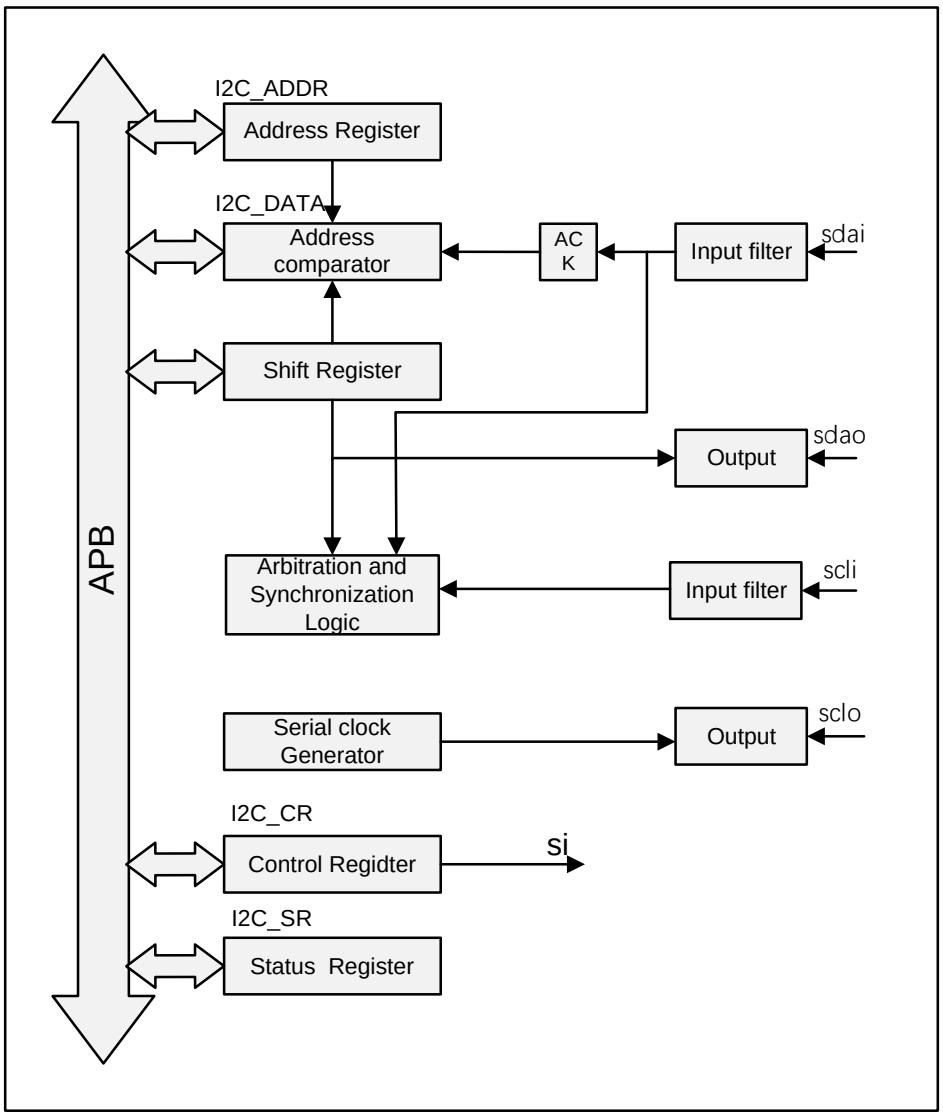


图 25-7 I2C 功能模块图

25.4.1 I2C 工作模式

I2C 组件可实现 8 位的双向数据传输，传输速率在标准模式下可达到 100Kbits/s 而在高速模式下可达 400Kbits/s，在超高速模式下可达 1Mbits/s，并且可以在以下四种模式下工作：

1) 主机发送模式：当“SCL”输出串行时钟信号时“SDA”输出串行数据。
2) 主机接收模式：当“SCL”输出串行时钟信号时串行数据通过“SDA”接收。
3) 从机接收模式：串行数据和串行时钟分别通过“SDA”和“SCL”接收。
4) 从机发送模式：当串行时钟从“SCL”口输入时串行数据通过“SDA”口发送。

25.4.2 仲裁与同步逻辑

在主发送模式中，仲裁逻辑检查每个发送的逻辑 1 是否真正出现在总线上。如果总线的另一个主机撤消了一个逻辑 1 并将 SDA 线拉低，仲裁丢失，I2C 模块立刻由主发送器变为从接收器。I2C 模块将继续输出时钟脉冲(在 SCL 上) 直至发送完当前的串行字节。

仲裁也可能在主接收模式中丢失。这种情况只在 I2C 模块正在向总线返回一个“非应答(逻辑 1)”时出现。当总线的另一个器件将信号拉低时仲裁丢失。由于它只在串行字节结束时出现，因此 I2C 模块不会再产生时钟脉冲。

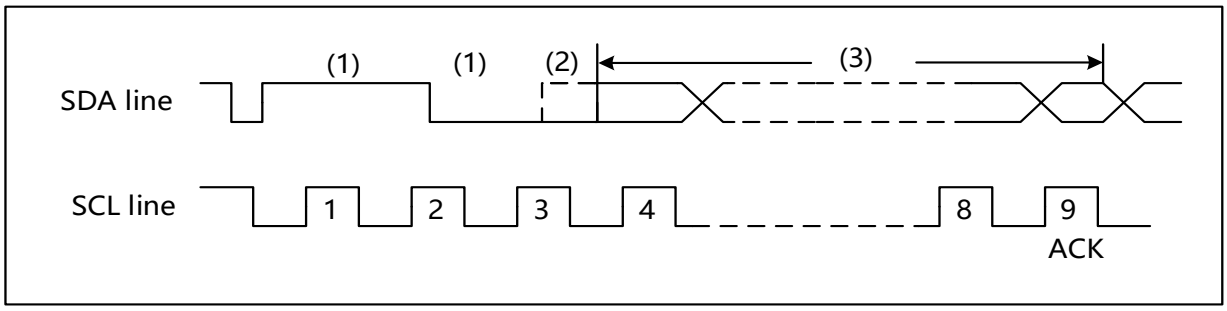


图 25-8 I2C 总线上的仲裁

1) 另一器件发送串行数据；
2) 另一器件通过拉低 SDA 先撤消了该 I2C 主机发送的一个逻辑 1，仲裁丢失，I2C 进入从接收模式；
3) 此时 I2C 处于从接收模式，但仍产生时钟脉冲，直至发送完当前字节。I2C 将不为下个字节的传输产生时钟脉冲。一旦赢得仲裁，SDA 上的数据传输由新的主机来启动。

同步逻辑使得串行时钟发生器与另一个器件 SCL 线上的时钟脉冲同步。如果 2 个或更多主器件产生时钟脉冲，则高电平时间取决于产生最短高电平时间的器件；低电平时间取决于产生最长低电平时间的器件。

25.4.3 串行时钟发生器

串行时钟发生器采用一个 8 位的计数器作为波特率发生器，SCL 信号和 pclk 信号的频率关系为 Fscl=Fpclk/8*(N+1)

下面的表格表示 pclk 为各种频率时，分频系数为 1-7 时，SCL 信号的频率值。

表 25-1 I2C 时钟信号波特率

频率 (Khz)	1	2	3	4	5	6	7
1000	62	41	31	25	20	17	15
2000	125	83	62	50	41	35	31
4000	250	166	125	100	83	71	62
6000	375	250	187	150	125	107	93
8000	500	333	250	200	166	142	125
10000	625	416	312	250	208	178	156
12000	750	500	375	300	250	214	187
14000	875	583	437	350	291	250	218
16000	1000	666	500	400	333	285	250

25.4.4 输入滤波器

输入信号与时钟信号(clk)同步，低于 3 个时钟周期的尖锋脉冲信号会被滤除。每个滤波器由 3 个触发器组成。第一个触发器用来直接锁存输入信号，并将数据载入由另外两个构成的移位寄存器中。当第二和第三个触发器的状态是”11”或”00”时，内部的滤除信号会各自被置 1 或置 0。

25.4.5 地址比较器

I2C 比较器将自己的从机地址与接收到的 7 位从机地址做比较。它可使用”I2C_ADDR”寄存器对自己的从机地址进行编程。并且会根据”I2C_ADDR”寄存器的”GC”位与首次接收到的 8 位字节与通用调用地址(0x00)相比较。如果任何一者相同，”I2C_CR”寄存器的”si”位会被置 1 并产生一个中断请求。

25.4.6 中断产生器

I2C 模块的所有四种模式都被使用时，则有 26 种可能的总线状态。当 I2C 进入 26 种状态的 25 种状态时，I2C_CR.SI 位会被硬件置 1。I2C_CR.SI 位唯一不会被置 1 的状态是:F8h，这表明没有有效的相关状态信息。I2C_CR.SI 位必须通过软件清零。为了清除

I2C_CR.SI 位，必须把 0 写入此位。若在”si”里写 1 不会改变”si”的值。为了确定中断的实际中断源，中断服务程序在清除 I2C_CR.SI 位之前，会对 I2C 状态寄存器进行查询。

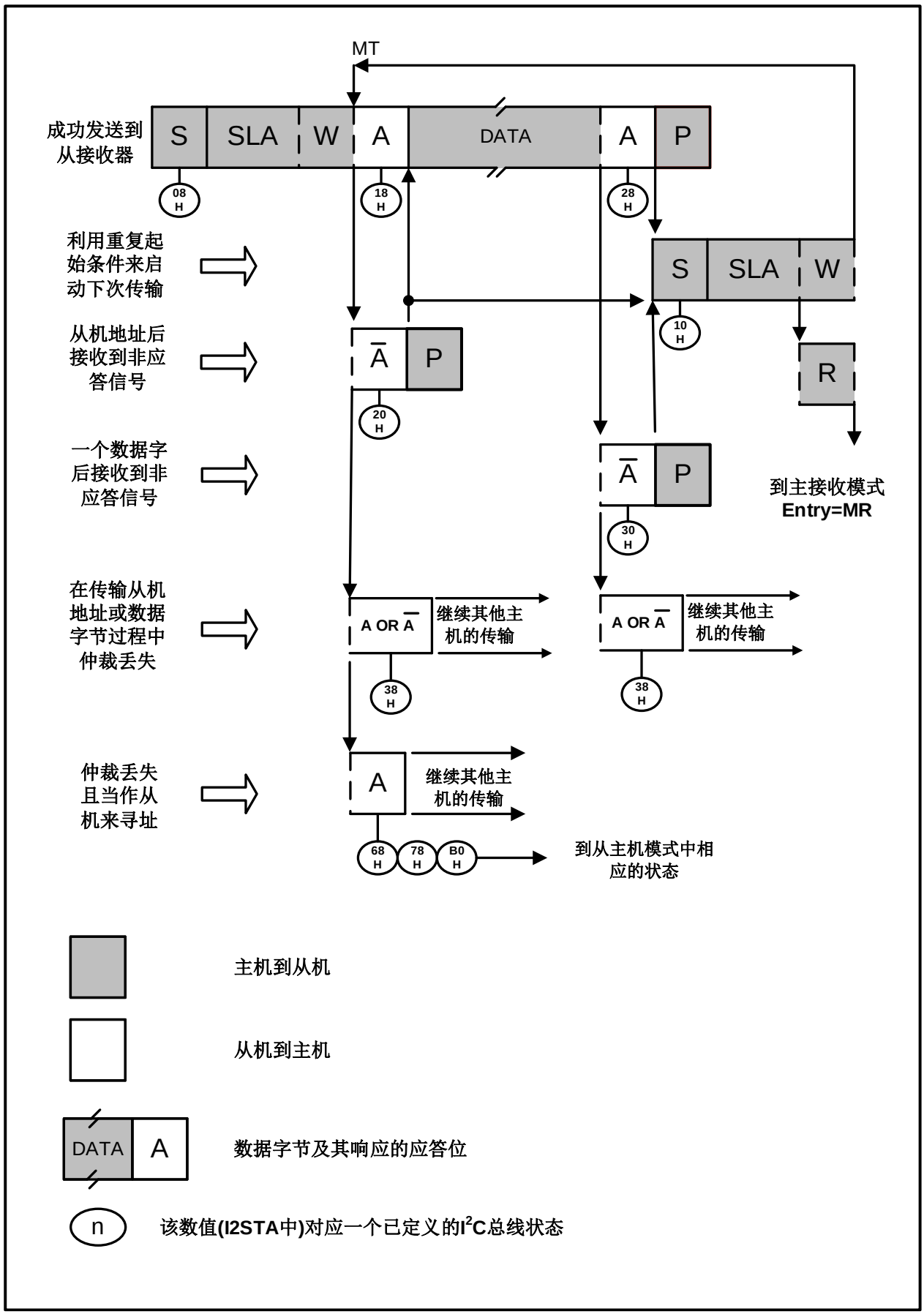
25.4.7 I2C 主机发送模式

必须将 I2C_CR.ENS 置“1”来使能 I2C 模块。如果 I2C_CR.AA 位复位，当另一个器件正变成总线主机时，I2C 模块将不会应答其自身的从机地址或通用调用地址。换句话说，如果 I2C_CR.AA 位复位，I2C 接口就不能进入从机模式。I2C_CR.STA、I2C_CR.STO 和 I2C_CR.SI 必须复位。

此时，可通过置位 I2C_CR.STA 位进入主发送模式。一旦总线空闲，I2C 逻辑会马上测试 I2C 总线并产生一个起始条件。当发送起始条件时，串行中断标志(I2C_CR.SI)置位，状态寄存器(I2C_STAT)中的状态代码为 0x08。中断服务程序利用该状态代码进入相应的状态服务程序，将从机地址和数据方向位(SLA+W)装入 I2C_DATA。I2C_CR.SI 位必须在串行传输继续之前复位。当发送完从机地址和方向位且接收到一个应答位时，串行中断标志(I2C_CR.SI))再次置位，I2C_STAT 中可能是一系列不同的状态代码。主机模式下为 0x18，0x20 或 0x38，从机模式(I2C_CR.AA=逻辑 1)为 0x68，0x78 或 0xB0。每个状态代码对应的操作在下表中详细介绍。在发送完重复起始条件(状态 0x10)后，I2C 模块通过将 SLA+R 装入 I2C_DATA 切换到主接收模式。

表 25-2I2C 主机发送模式状态表

状态代码	I2C 总线和硬件状态	应用软件响应					I2C 硬件执行的下一个动作
		读/写 I2C_DATA	写 I2C_CR				
			sta	sto	si	aa	
08H	已发送起始条件	装入 SLA+W	X	0	0	X	将发送 SLA+W，接收 ACK
10H	已发送重复起始条件	装入 SLA+W	X	0	0	X	同上
		装入 SLA+R	X	0	0	X	将发送 SLA+R，I2C 自动切换到主接收模式
18H	已发送 SLA+W 已接收 ACK	装入数据字节	0	0	0	X	将发送数据字节，将接收 ACK
		无 I2C_DATA 动作	1	0	0	X	将发送重复起始条件
		无 I2C_DATA 动作	0	1	0	X	将发送停止条件，sto 标志位复位
		无 I2C_DATA 动作	1	1	0	X	将先发送停止条件，随后发送起始条件，sto 标志位复位
20H	已发送 SLA+W 已接收非 ACK	装入数据字节	0	0	0	X	将发送数据字节，将接收 ACK
		无 I2C_DATA 动作	1	0	0	X	将发送重复起始条件
		无 I2C_DATA 动作	0	1	0	X	将发送停止条件，sto 标志位复位
		无 I2C_DATA 动作	1	1	0	X	将先发送停止条件，随后发送起始条件，sto 标志位复位
28H	已发送 I2C_DATA 中的数据，已接收 ACK	装入数据字节	0	0	0	X	将发送数据字节，将接收 ACK
		无 I2C_DATA 动作	1	0	0	X	将发送重复起始条件
		无 I2C_DATA 动作	0	1	0	X	将发送停止条件，sto 标志位复位
		无 I2C_DATA 动作	1	1	0	X	将先发送停止条件，随后发送起始条件，sto 标志位复位
30H	已发送 I2C_DATA 中的数据	装入数据字节	0	0	0	X	将发送数据字节，将接收 ACK
		无 I2C_DATA 动作	1	0	0	X	将发送重复起始条件
		无 I2C_DATA 动作	0	1	0	X	将发送停止条件，sto 标志位复位
		无 I2C_DATA 动作	1	1	0	X	将先发送停止条件，随后发送起始条件，sto 标志位复位
38H	在 SLA+R/W 或写数据字节时丢失仲裁	无 I2C_DATA 动作	0	0	0	X	I2C 总线被释放，进入不可寻址从模式
		无 I2C_DATA 动作	1	0	0	X	当 I2C 总线空闲时发送起始条件



25-9 I2C 主机发送状态图

25.4.8 I2C 主机接收模式

在主接收模式中，主机所接收的数据字节来自从发送器。按主发送模式中的方法初始化传输。当发送完起始条件后，中断服务程序必须把 7 位从机地址和数据方向位(SLA+R)装入 I2C_DATA。必须先清除 I2C_DATA__SI 位，再继续执行串行传输。当发送完从机地址和数据方向位且接收到一个应答位时，串行中断标志 I2C_DATA__SI 位再次置位，这时，I2C_STAT 中可能是一系列不同的状态代码。主机模式下为 0x40，0x48 或 0x38，从机模式(I2C_CR_AA=1)为 0x68，0x78 或 0xB0。每个状态代码对应的操作详见下表。在发送完重复起始条件(状态 0x10)后，I2C 模块通过将 SLA+W 装入 I2C_DATA 切换到主发送模式。

表 25-3 I2C 主机接收模式状态表

状态代码	I2C 总线和硬件状态	应用软件响应					I2C 硬件执行的下一个动作
		读/写 I2C_DATA	写 I2C_CR				
			sta	sto	si	aa	
08H	已发送起始条件	装入 SLA+R	X	0	0	X	将发送 SLA+R，接收 ACK
10H	已发送重复起始条件	装入 SLA+R	X	0	0	X	同上
		装入 SLA+W	X	0	0	X	将发送 SLA+W，I2C 自动切换到主发送模式
38H	在非 ACK 中丢失仲裁	无 I2C_DATA 动作	0	0	0	X	I2C 总线将被释放；进入从模式
		无 I2C_DATA 动作	1	0	0	X	当总线空闲时发起起始条件
40H	已发 SLA+R 已接收 ACK	无 I2C_DATA 动作	0	0	0	0	将接收数据字节，将返回非 ACK
		无 I2C_DATA 动作	0	0	0	1	将接收数据字节，将返回 ACK
48H	已发 SLA+R 已接收非 ACK	无 I2C_DATA 动作	1	0	0	X	将发送重复起始条件
		无 I2C_DATA 动作	0	1	0	X	将发送停止条件，sto 标志位复位
		无 I2C_DATA 动作	1	1	0	X	将先发送停止条件，随后发送起始条件，sto 标志位复位
50H	已接收数据字节，ACK 已返回	读取数据字节	0	0	0	0	将接收数据字节，将返回非 ACK
		读取数据字节	0	0	0	1	将接收数据字节，将返回 ACK
58H	已接收数据字节，非 ACK 已返回	读取数据字节	1	0	0	X	将发送重复起始条件
		读取数据字节	0	1	0	X	将发送停止条件，sto 标志位复位

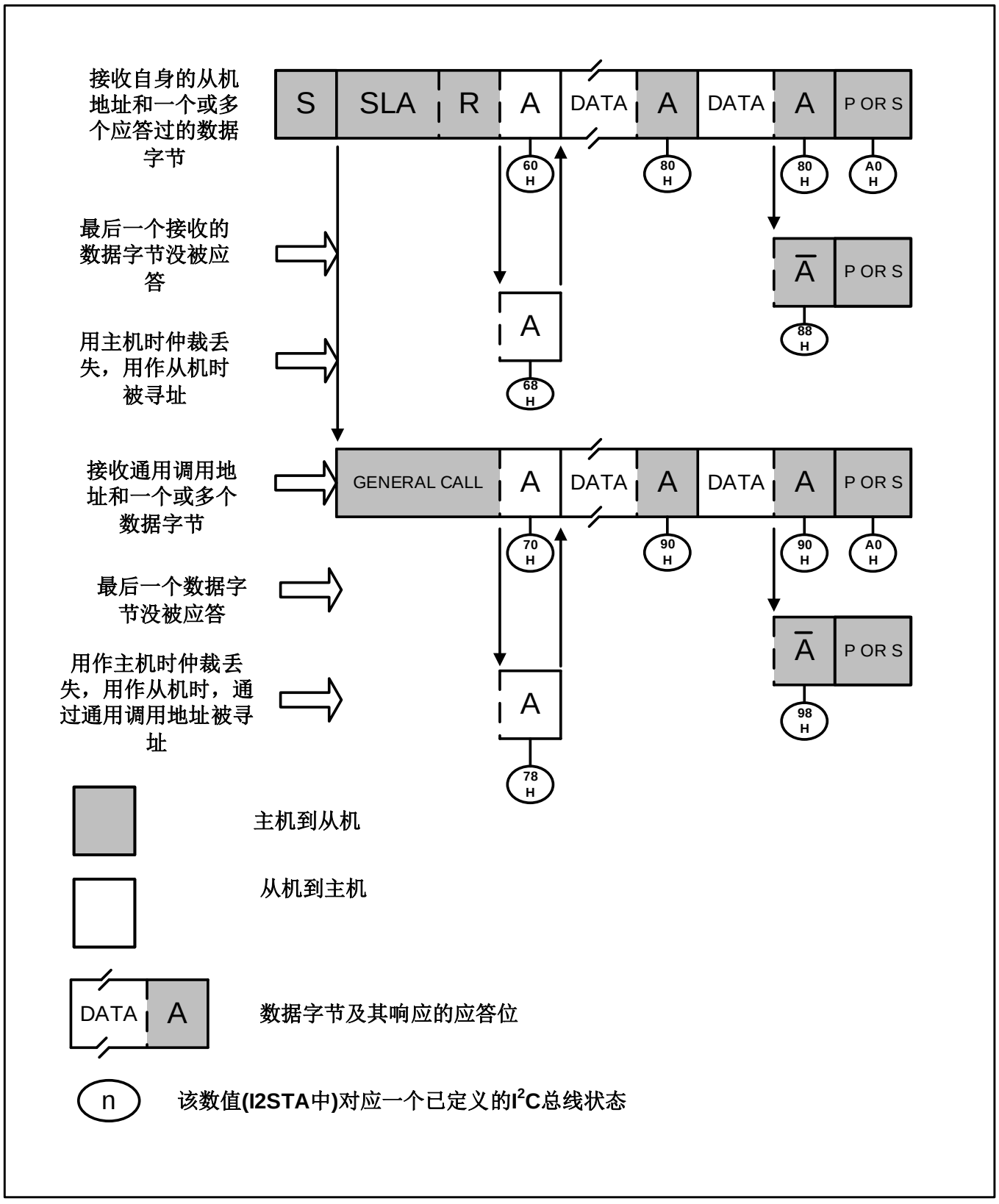


图 25-10 I2C 主机接收状态图

25.4.9 I2C 从机接收模式

在从接收模式中，从机接收的数据字节来自主发送器。高 7 位是主机寻址时 I2C 模块响应的地址。如果 LSB(I2C_ADDR.GC)被置位，I2C 模块将响应通用调用地址(0x00)；否则忽略通用调用地址。

I2C 总线速率的设置不影响从机模式中的 I2C 模块。必须置位 I2C_CR.ENS 位来使能 I2C 模块。I2C_CR.AA 位必须置位以使能 I2C 模块来应答其自身从机地址或通用调用地址。Sta，sto 和 si 必须复位。

当 I2C_ADDR 和 I2C_CR 完成初始化后，I2C 模块一直等待，直至被从机地址寻址，之后是数据方向位寻址，为了工作在从接收模式中，数据方向位必须为“0”(W)。接收完其自身的从机地址和 W 位后，串行中断标志(I2C_CR.SI)置位，可从 I2C_STAT 中读出一个有效的状态代码。该状态代码用作状态服务程序的向量。每个状态代码的对应操作见下表。如果 I2C 模块在主机模式中仲裁丢失，也可进入从接收模式(请参考状态 0x68 和 0x78 的描述)。

如果 I2C_CR.AA 位在传输过程中复位，则在接收完下一个数据字节后 I2C 模块将向 SDA 返回一个非应答(逻辑 1)。当 I2C_CR.AA 复位时，I2C 模块不响应其自身的从机地址或通用调用地址。但是，I2C 总线仍被监控，而且，地址识别可随时通过置位 I2C_CR.AA 来恢复。这就意味着 I2C_CR.AA 位可临时将 I2C 模块从 I2C 总线上分离出来。

表 25-4 I2C 从机接收模式状态表

状态代码	I2C 总线和硬件状态	应用软件响应					I2C 硬件执行的下一个动作
		读/写 I2C_DATA	写 I2C_CR				
			sta	sto	si	aa	
60H	已接收自身的 SLA+W；已接收 ACK	无 I2C_DATA 动作	X	0	0	0	将接收数据字节，将返回非 ACK
		无 I2C_DATA 动作	X	0	0	1	将接收数据字节，将返回 ACK
68H	主控 时在 SLA+R/W 丢失仲裁；已接收自身的 SLA+W；已返回 ACK；	无 I2C_DATA 动作	X	0	0	0	将接收数据字节，将返回非 ACK
		无 I2C_DATA 动作	X	0	0	1	将接收数据字节，将返回 ACK
70H	已接收通用调用地址(0x00)；已返回 ACK；	无 I2C_DATA 动作	X	0	0	0	将接收数据字节，将返回非 ACK
		无 I2C_DATA 动作	X	0	0	1	将接收数据字节，将返回 ACK
78H	主控 时在 SLA+R/W 中丢失仲裁；已接收通用调用地址；已返回 ACK；	无 I2C_DATA 动作	X	0	0	0	将接收数据字节，将返回非 ACK
		无 I2C_DATA 动作	X	0	0	1	将接收数据字节，将返回 ACK

80H	前一次寻址使用自身从地址；已接收数据字节；已返回 ACK；	无 I2C_DATA 动作	X	0	0	0	将接收数据字节，将返回非 ACK
		无 I2C_DATA 动作	X	1	0	1	将接收数据字节，将返回 ACK
88H	前一次寻址使用自身从地址；已接收数据字节；已返回非 ACK;	读取数据字节	0	0	0	0	切换到不可寻址从模式；不识别自身从地址或通用地址；
		读取数据字节	0	0	0	1	切换到不可寻址从模式；不识别自身从地址或通用地址；
		读取数据字节	1	0	0	0	切换到不可寻址从模式；不识别自身从地址或通用地址；
		读取数据字节	1	0	0	1	切换到不可寻址从模式；不识别自身从地址或通用地址；当总线空闲后发送起始条件；
90H	前一次寻址使用通用调用地址；已接收数据；已返回 ACK；	读取数据字节	1	0	0	X	将接收数据字节，将返回非 ACK
		读取数据字节	0	1	0	X	将接收数据字节，将返回 ACK
98H	前一次寻址使用通用调用地址；已接收数据；已返回非 ACK;	读取数据字节	0	0	0	0	切换到不可寻址从模式；不识别自身从地址或通用地址；
		读取数据字节	0	0	0	1	切换到不可寻址从模式；不识别自身从地址或通用地址；
		读取数据字节	1	0	0	0	切换到不可寻址从模式；不识别自身从地址或通用地址；当总线空闲后发送起始条件；
		读取数据字节	1	0	0	1	切换到不可寻址从模式；不识别自身从地址或通用地址；当总线空闲后发送起始条件；
A0H	当使用从接收/从发送模式中静态寻址时，接收到停止条件或重复起始条件	无 I2C_DATA 动作	0	0	0	0	切换到不可寻址从模式；不识别自身从地址或通用地址；
		无 I2C_DATA 动作	0	0	0	1	切换到不可寻址从模式；不识别自身从地址或通用地址；
		无 I2C_DATA 动作	1	0	0	0	切换到不可寻址从模式；不识别自身从地址或通用地址；当总线空闲后发送起始条件；
		无 I2C_DATA 动作	1	0	0	1	切换到不可寻址从模式；不识别自身从地址或通用地址；当总线空闲后发送起始条件；

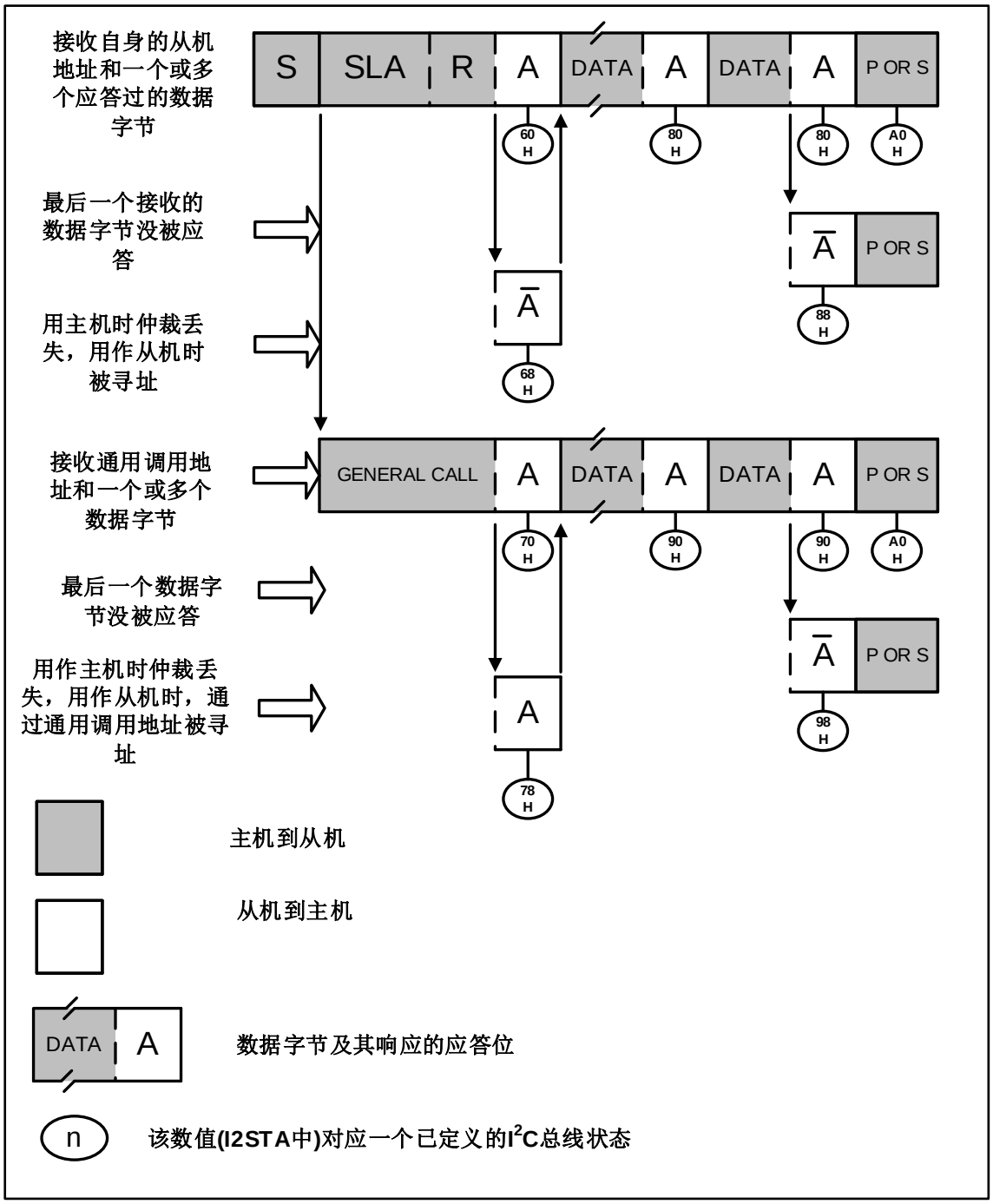


图 25-11 I2C 从机接收状态图

25.4.10 I2C 从机发送模式

在从发送模式中，向主接收器发送数据字节。数据传输按照从接收模式中的情况初始化。当初始化 I2C_ADDR 和 I2C_CR 后，I2C 模块一直等待，直至被自身的从机地址寻址，之后是数据方向位，该数据方向位必须为“1”I，以便 I2C 模块工作在从发送模式下。接收完其自身的从机地址和 R 位后，串行中断标志(I2C_CR_SI)置位，并且可从 I2C_STAT 中读取一个有效的状态代码。该状态代码用作状态服务程序的向量，每个状态代码的对应操作见下表所示。如果 I2C 模块在主机模式下时仲裁丢失，则可进入从发送模式(见状态 0xB0)。如果 I2C_CR_AA 位在传输过程中复位，则 I2C 模块将发送最后一个字节并进入状态 0xC0 或 0xC8。I2C 模块切换到非寻址的从机模式，如果继续传输，它将忽略主接收器。因此主接收器接收所有 1 作为串行数据。当 I2C_CR_AA 复位时，I2C 模块不响应其自身的从机地址或通用调用地址。但是，I2C 总线仍被监控，而且，地址识别可随时通过置位 I2C_CR_AA 来恢复。这就意味着 I2C_CR_AA 位可用来暂时将 I2C 模块从 I2C 总线上分离出来。

表 25-5 从机发送模式状态表

状态代码	I2C 总线和硬件状态	应用软件响应					I2C 硬件执行的下一个动作
		读/写 I2C_DATA	写 I2C_CR				
			sta	sto	si	aa	
A8H	已接收自身的 SLA+R；已返回 ACK	装入数据字节	X	0	0	0	将发送最后一个数据字节；将接收 ACK；
		装入数据字节	X	0	0	1	将发送一个数据字节；将接收 ACK；
B0H	当主控时在 SLA+R/W 中丢失仲裁；已接收自身 SLA+R；已返回 ACK；	装入数据字节	X	0	0	0	将发送最后一个数据字节；将接收 ACK;
		装入数据字节	X	0	0	1	将发送一个数据字节；将接收 ACK；
B8H	已发送数据；已接收 ACK；	装入数据字节	X	0	0	0	将发送最后一个数据字节；将接收 ACK；
		装入数据字节	X	0	0	1	将发送一个数据字节；将接收 ACK；
C0H	已发送数据字节；已接收非 ACK;	无 I2C_DATA 动作	0	0	0	0	切换到不可寻址从模式；不识别自身从地址或通用地址；
		无 I2C_DATA 动作	0	0	0	1	切换到不可寻址从模式；不识别自身从地址或通用地址；
		无 I2C_DATA 动作	1	0	0	0	切换到不可寻址从模式；不识别自身从地址或通用地址；当总线空闲后发送起始条件；
		无 I2C_DATA 动作	1	0	0	1	切换到不可寻址从模式；不识别自身从地址或通用地址；当总线空闲后发送起始条件；

C8H	装入的数据字节已被发送；已接收 ACK；	无 I2C_DATA 动作	0	0	0	0	切换到不可寻址从模式；不识别自身从地址或通用地址；
		无 I2C_DATA 动作	0	0	0	1	切换到不可寻址从模式；不识别自身从地址或通用地址；
		无 I2C_DATA 动作	1	0	0	0	切换到不可寻址从模式；不识别自身从地址或通用地址；当总线空闲后发送起始条件；
		无 I2C_DATA 动作	1	0	0	1	切换到不可寻址从模式；不识别自身从地址或通用地址；当总线空闲后发送起始条件；

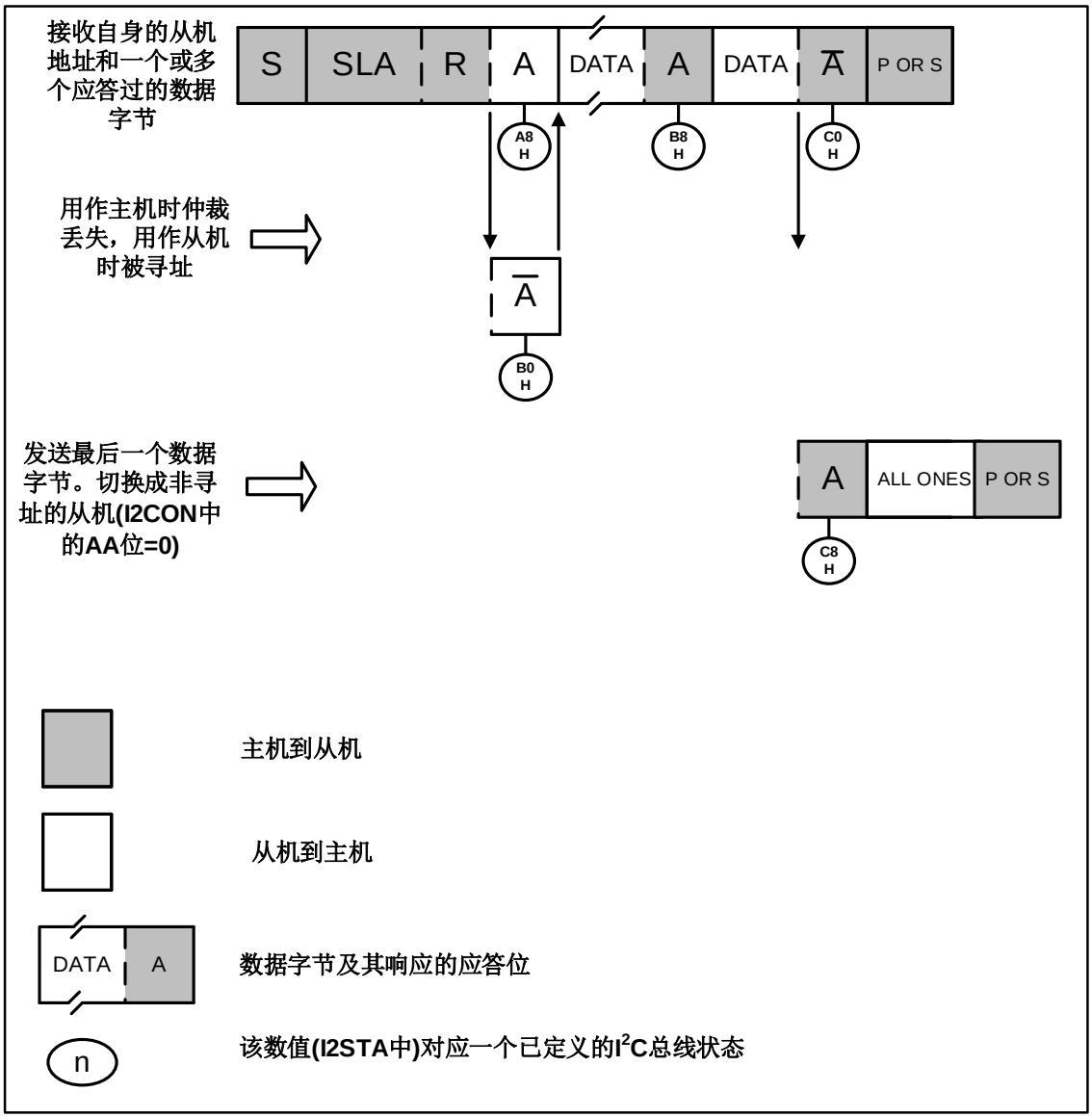


图 25-12 I2C 从机发送状态图

25.4.11 I2C 其他杂项状态

I2C_STAT=0xF8：

这个状态码表示没有任何可用的相关信息，因为串行中断标志 I2C_CR.SI 位还没有置位。这种情况在其它状态和 I2C 模块还未开始执行串行传输之间出现。

I2C_STAT=0x00：

该状态代码表示在 I2C 串行传输过程中出现了总线错误。当格式帧的非法位置上出现了起始或停止条件时总线错误产生。这些非法位置是指在串行传输过程中的地址字节、数据字节或应答位。当外部干扰影响到内部 I2C 模块信号时也会产生总线错误。总线错误出现时 I2C_CR.SI 位置位。要从总线错误中恢复，sto 标志必须置位，si 必须被清除。这使得 I2C 模块进入“非寻址的”从机模式(已定义的状态)并清除 sto 标志(I2C_CR 中的其它位不受影响)。SDA 和 SCL 线被释放(不发送停止条件)。

表 25-6 其他杂项状态表

状态代码	I2C 总线和硬件状态	应用软件响应					I2C 硬件执行的下一个动作
		读/写 I2C_DATA	写 I2C_CR				
			sta	sto	si	aa	
F8H	无可用的相关状态信息；si=0；	无 I2C_DATA 动作	无 I2C_DATA 动作				等待或执行当前传输
00H	由于非法的起始或停止条件的出现，在主机或被选中的从机将出现总线错误；当外部干扰使 I2C 进入未定义的状态时也会出	无 I2C_DATA 动作	0	1	0	X	只有在主机或被寻址的从机模式中，内部硬件受影响。一般情况下，总线被释放，I2C 模块切换到非寻址的从机模式。Sto 复位。

25.5 I2C 操作模式

25.5.1 初始化程序

将 I2C 接口初始化用作从机和/或主机的例子。

a) 将自身的从机地址装入 I2C_ADDR，使能通用调用识别(如果需要的话)

b) 使能 I2C 中断；

c) 向寄存器 I2C_CR 写入 0x44 来置位 ENS 和 AA 位，并使能从机功能。对于主机功能，可向寄存器 I2C_CR 写入 0x40。

25.5.2 端口配置程序

I2C 接口信号 SCL，SDA 映射到芯片引脚 PB6，PB7 的例子。

a) 配置 PB6，PB7 为开漏输出模式：PBOTYP6，PBOTYP7 配置为 0x1

b) 配置 PB6，PB7 的功能配置寄存器：PBAFR6，PBAFR7 配置为 0x2

c) 配置 PB6，PB7 的上拉使能配置寄存器：PBPUPD6，PBPUPD7 配置为 0x1

25.5.3 启动主机发送功能

通过建立缓冲区、指针和数据计数然后发启起始条件便可执行主发送操作。

a) 初始化主机数据计数器；

b) 建立数据将被发送到的从机地址，并且添加写位；

c) 向 I2C_CR 写入 0x20 来置位 sta 位；

d) 在主发送缓冲区内建立要发送的数据；

e) 初始化主机数据计数器来匹配正在发送的信息长度；

f) 退出。

25.5.4 启动主机接收功能

通过建立缓冲区、指针和数据计数然后发启起始条件便可执行主接收操作。

a) 初始化主机数据计数器；

b) 建立数据将被发送到的从机地址，并且添加读位；

c) 向 I2C_CR 写入 0x20 来置位 sta 位；

d) 在主接收缓冲区内建立要发送的数据；

e) 初始化主机数据计数器来匹配正在发送的信息长度；

f) 退出。

25.5.5 I2C 中断程序

确定 I2C 的状态和处理该状态的状态程序。

a) 从 I2C_STAT 中读出 I2C 的状态；

b) 使用状态值跳转到 26 个可能状态程序中的一个。

25.5.6 无指定模式状态

1) 状态：0x00 总线错误。进入非寻址的从机模式并释放总线。
   a) 向 I2C_CR 写入 0x14 来置位 sto 和 aa 位；
   b) 向 I2C_CR 写入 0xF7 来清除 si 标志；
   c) 退出。
2) 主机状态

状态 08 和 10 适用于主发送模式和主接收模式。R/W 位决定了下一个状态是在主发送模式中还是在主接收模式中。

3) 状态：0x08

已发送起始条件。即将发送从机地址＋R/W 位和接收 ACK 位。

   a) 向 I2C_DATA 写入从机地址和 R/W 位；
   b) 向 I2C_CR 写入 0x04 来置位 aa 位；
   c) 向 I2C_CR 写入 0xF7 来清除 si 标志；
   d) 建立主发送模式数据缓冲区；
   e) 建立主接收模式数据缓冲区；
   f) 初始化主机数据计数器；
   g) 退出。

4) 状态：0x10

已发送重复起始条件。即将发送从机地址＋R/W 位和接收 ACK 位。

   a) 向 I2C_DATA 写入从机地址和 R/W 位；
   b) 向 I2C_CR 写入 0x04 来置位 aa 位；
   c) 向 I2C_CR 写入 0xF7 来清除 si 标志；
   d) 建立主发送模式数据缓冲区；
   e) 建立主接收模式数据缓冲区；
   f) 初始化主机数据计数器；
   g) 退出。

25.5.7 主发送状态

1) 状态：0x18

之前状态为 8 或 10 表示已发送从机地址和写操作位，并接收了应答。即将发送第一个数据字节和接收 ACK 位。

   a) 将主发送缓冲区的第一个数据字节装入 I2C_DATA；
   b) 向 I2C_CR 写入 0x04 来置位 aa 位；
   c) 向 I2C_CR 写入 0xF7 来清除 si 标志；
   d) 主发送缓冲区指针加 1；
   e) 退出。

2) 状态：0x20 已发送从机地址和写操作位并接收了非应答。即将发送停止条件。

a) 向 I2C_CR 写入 0x14 来置位 sto 和 aa 位；
b) 向 I2C_CR 写入 0xF7 来清除 si 标志；
c) 退出。

3) 状态：0x28

已发送数据并接收了 ACK。如果发送的数据是最后一个数据字节则发送一个停止条件，否则发送下一个数据字节。

a) 主机数据计数器减 1，如果发送的不是最后一个数据字节就跳至第 e)步；
b) 向 I2C_CR 写入 0x14 来置位 sto 和 aa 位；
c) 向 I2C_CR 写入 0xF7 来清除 si 标志；
d) 退出；
e) 将主发送缓冲区的下一个数据字节装入 I2C_DATA；
f) 向 I2C_CR 写入 0x04 来置位 aa 位；
g) 向 I2C_CR 写入 0xF7 来清除 si 标志；
h) 主机发送缓冲区指针加 1；
i) 退出。

4) 状态：0x30 已发送数据并接收到非应答。即将发送停止条件；

a) 向 I2C_CR 写入 0x14 来置位 sto 和 aa 位；
b) 向 I2C_CR 写入 0xF7 来清除 si 标志；
c) 退出。

5) 状态：0x38 仲裁已在发送从机地址和写操作位或数据的过程中丢失。总线已被释放且进入非寻址的从机模式。当总线再次空闲时将发送一个新的起始条件。

a) 向 I2C_CR 写入 0x24 来置位 sta 和 aa 位；
b) 向 I2C_CR 写入 0xF7 来清除 si 标志；
c) 退出。

25.5.8 主接收状态

1) 状态：0x40

前面的状态是 08 或 10 表示已发送从机地址和读操作位，并接收到 ACK。将接收数据和返回 ACK。

a) 向 I2C_CR 写入 0x04 来置位 aa 位；
b) 向 I2C_CR 写入 0xF7 来清除 si 标志；
c) 退出。

2) 状态：0x48

已发送从机地址和读操作位，并接收到非应答。将发送停止条件。

a) 向 I2C_CR 写入 0x14 来置位 sto 和 aa 位；
b) 向 I2C_CR 写入 0xF7 来清除 si 标志；
c) 退出。

3) 状态：0x50

已接收到数据，并返回 ACK。将从 I2C_DATA 读取数据。将接收其它的数据。如果这是最后一个数据字节，则返回非应答，否则返回 ACK。

a) 读取 I2C_DATA 中的数据字节，存放到主机接收缓冲区；
b) 主机数据计数器减 1，如果不是最后一个数据字节就跳到第 e)步；
c) 向 I2C_CR 写入 0xF3 来清除 si 标志和 aa 位；
d) 退出；
e) 向 I2C_CR 写入 0x04 来置位 aa 位；
f) 向 I2C_CR 写入 0xF7 来清除 si 标志；
g) 主机接收缓冲区指针加 1；
h) 退出。

4) 状态：0x58

已接收到数据，已返回非应答。将从 I2C_DATA 中读取数据和发送停止条件。

a) 读取 I2C_DATA 中的数据字节，存放到主机接收缓冲区；
b) I2C_CR 写入 0x14 来置位 sto 和 aa 位；
c) 向 I2C_CR 写入 0xF7 来清除 si 标志；
d) 退出。

25.5.9 从接收状态

1) 状态：0x60

已接收到自身从机地址和写操作位，已返回 ACK。将接收数据和返回 ACK。

a) 向 I2C_CR 写入 0x04 来置位 aa 位；
b) 向 I2C_CR 写入 0xF7 来清除 si 标志；
c) 建立从接收模式数据缓冲区；
d) 初始化从机数据计数器；
e) 退出。

2) 状态：0x68

用作总线主机时仲裁已在传输从机地址和 R/W 位时丢失。已接收到自身从机地址和写操作位，并已返回 ACK。将接收数据和返回 ACK。当总线再次空闲后置位 sta 来重启主机模式。

a) 向 I2C_CR 写入 0x24 来置位 sta 和 aa 位；
b) 向 I2C_CR 写入 0xF7 来清除 si 标志；
c) 建立从接收模式数据缓冲区；
d) 初始化从机数据计数器；
e) 退出。

3) 状态：0x70

已接收到通用调用和返回 ACK。将接收数据和返回 ACK。

a) 向 I2C_CR 写入 0x04 来置位 aa 位；
b) 向 I2C_CR 写入 0xF7 来清除 si 标志；
c) 建立从接收模式数据缓冲区；
d) 初始化从机数据计数器；
e) 退出。

4) 状态：0x78

用作总线主机时仲裁已在传输从机地址和 R/W 位时丢失。已接收到通用调用和返回 ACK。将接收数据和返回 ACK。当总线再次空闲后置位 sta 来重启主机模式。

a) 向 I2C_CR 写入 0x24 来置位 sta 和 aa 位；
b) 向 I2C_CR 写入 0xF7 来清除 si 标志；
c) 建立从接收模式数据缓冲区；
d) 初始化从机数据计数器；
e) 退出。

5) 状态：0x80

之前寻址自身从机地址。已接收到数据并返回 ACK。将读取其它数据。

a) 读取 I2C_DATA 的数据字节，存放到从机接收缓冲区。
b) 从机数据计数器减 1，如果不是最后一个数据字节就跳到第 e)步；
c) 向 I2C_CR 写入 0xF3 来清除 si 标志和 aa 位；
d) 退出；
e) 向 I2C_CR 写入 0x04 来置位 aa 位；
f) 向 I2C_CR 写入 0xF7 来清除 si 标志；
g) 从机接收缓冲区指针加 1；
h) 退出。

6) 状态：0x88 之前寻址自身从机地址。已接收到数据并返回非应答。不会保存接收到的数据。进入非寻址的从机模式。

a) 向 I2C_CR 写入 0x04 来置位 aa 位；
b) 向 I2C_CR 写入 0xF7 来清除 si 标志；
c) 退出。

7) 状态：0x90

之前寻址通用调用地址。已接收到数据并返回 ACK。将保存接收到的数据。只接收第一个数据字节并返回 ACK。接收其它数据字节后返回非应答。

a) 读取 I2C_DATA 的数据字节，并放入从机接收缓冲区；
b) 向 I2C_CR 写入 0xF3 来清除 si 标志和 aa 位；
c) 退出。

8) 状态：0x98 之前寻址通用调用地址。已接收到数据并返回非应答。不会保存接收到的数据。进入非寻址的从机模式。

a) 向 I2C_CR 写入 0x04 来置位 aa 位；

b) 向 I2C_CR 写入 0xF7 来清除 si 标志；

c) 退出。

9) 状态：0xA0 已接收停止条件或重复起始条件，但仍作为从机寻址。不保存接收到的数据。进入非寻址的从机模式。

a) 向 I2C_CR 写入 0x04 来置位 aa 位；

b) 向 I2C_CR 写入 0xF7 来清除 si 标志；

c) 退出。

25.5.10 从发送状态

1) 状态：0xA8

已接收自身从机地址和读操作位并返回 ACK。将发送数据和接收 ACK 位。

a) 将从机发送缓冲区的第一个数据字节装入 I2C_DATA；

b) 向 I2C_CR 写入 0x04 来置位 aa 位；

c) 向 I2C_CR 写入 0xF7 来清除 si 标志；

d) 建立从发送模式数据缓冲区；

e) 从机发送缓冲区指针加 1；

f) 退出。

2) 状态：0xB0

用作总线主机时，在传输从机地址和 R/W 位时丢失仲裁。已接收自身从机地址和读操作位并返回 ACK。将发送数据和接收 ACK 位。当总线再次空闲后置位 sta 来重启主机模式。

a) 将从机发送缓冲区的第一个数据字节装入 I2C_DATA；

b) 向 I2C_CR 写入 0x24 来置位 sta 和 aa 位；

c) 向 I2C_CR 写入 0xF7 来清除 si 标志；

d) 建立从发送模式数据缓冲区；

e) 从机发送缓冲区指针加 1；

f) 退出。

3) 状态：0xB8

已发送数据并接收到 ACK。将发送数据和接收 ACK 位。

a) 将从机发送缓冲区的数据字节装入 I2C_DATA；

b) 向 I2C_CR 写入 0x04 来置位 aa 位；

c) 向 I2C_CR 写入 0xF7 来清除 si 标志；

d) 从机发送缓冲区指针加 1；

e) 退出。

4) 状态：0xC0

已发送数据并接收到非应答。进入非寻址的从机模式。

a) 向 I2C_CR 写入 0x04 来置位 aa 位；

b) 向 I2C_CR 写入 0xF7 来清除 si 标志；

c) 退出。

5) 状态：0xC8

已发送最后一个数据字节并接收到 ACK。进入非寻址的从机模式。

a) 向 I2C_CR 写入 0x04 来置位 aa 位；

b) 向 I2C_CR 写入 0xF7 来清除 si 标志；

c) 退出。

25.6 I2C 寄存器列表

I2C0 基地址：0x40000C00

I2C1 基地址：0x40005C00

表 25-7 I2C 寄存器列表

偏移地址	名称	寄存器描述	复位值
0x00	I2Cx_CR	I2C 配置寄存器。	0x00000000
0x04	I2Cx_DATA	I2C 数据寄存器。	0x00000000
0x08	I2Cx_ADDR	I2C 地址寄存器。	0x00000000
0x0C	I2Cx_SR	I2C 状态寄存器。	0x000000F8
0x10	I2Cx_TIMRUN	I2C 波特率计数器使能寄存器。	0x00000000
0x14	I2Cx_BAUDCR	I2C 波特率计数器配置寄存器。	0x00000000

25.7 I2C 寄存器说明

25.7.1 I2C 配置寄存器(I2Cx_CR)

偏移地址：0x00

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留									ENS	STA	STO	SI	AA	保留	H1M
									R/W	R/W	R/W	R/W	R/W		R/W

位	标记	功能描述	复位值	读写
31:7	Res	保留	0x0	-
6	ENS	I2C 模块使能。 0：无效 1：使能	0x0	R/W
5	STA	开始标志使能。 0：无效 1：使能	0x0	R/W
4	STO	停止标志使能。 0：无效 1：使能	0x0	R/W
3	SI	I2C 中断标志位，写 0 清除 0：无效 1：使能	0x0	R/W
2	AA	应答标志使能。 0：无效 1：使能	0x0	R/W
1	Res	保留	0x0	-
0	H1M	I2C 高速 1Mbps 模式使能。 0：无效 1：使能	0x0	R/W

25.7.2 I2C 数据寄存器(I2Cx_DATA)

偏移地址：0x04
复位值：0x00000000

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	-
7:0	I2CDAT	I2C 数据寄存器。 在 I2C 发送模式下，写发送数据到这个寄存器。在 I2C 接收模式下，从这个寄存器读取接收数据。	0x0	R/W

25.7.3 I2C 地址寄存器(I2Cx_ADDR)

偏移地址：0x08
复位值：0x00000000

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	-
7:1	I2CADR	I2C 从机模式地址。	0x0	R/W
0	GC	广播地址应答使能。 0：无效 1：使能	0x0	R/W

25.7.4 I2C 状态寄存器(I2Cx_SR)

偏移地址：0x0C
复位值：0x000000F8

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	-
7:0	I2CSR	I2C 状态寄存器。	0xF8	R/W

25.7.5 I2C 波特率计数器使能寄存器(I2Cx_TIMRUN)

偏移地址：0x10
复位值：0x00000000

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	-
0	TME	波特率计数器使能寄存器。 0：无效 1：使能	0x0	R/W

25.7.6 I2C 波特率计数器配置寄存器(I2Cx_BAUDCR)

偏移地址：0x14
复位值：0x00000000

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	-
7:0	TM	TM：波特率计数器配置值。 Fo=Fi/8*(N+1)　N=tm，N>7	0x0	R/W

26 串行外设接口(SPI)

26.1 SPI 简介

SPI(Serial Peripheral Interface)总线是一种同步串行外设接口，它可以使 MCU 与各种外围设备以串行的方向进行信息交换，SPI 接口使用 4 条线，串行时钟线(SCK)，主机输出/从机输入线(MOSI)，主机输入/从机输出线(MISO)以及低电平有效从机选择线(SSN)。

26.2 SPI 主要特性

SPI 控制器支持以下特性：

- 通过编程可以配置为主机或者从机
- 全双工通信能力
- 7 种波特率可配置
- 4 线传输方式
- 主机方式最大波特率为 1/2 系统时钟
- 从机方式最大波特率为 1/4 系统时钟
- 可配置的串行时钟极性和相位
- 支持中断方式
- 8 位的数据传输先传输高位后低位

26.3 功能描述

26.3.1 SPI 主机方式

SPI 总线上的所有数据传输都由 SPI 主器件启动，通过将主机/从机控制位 SPI_CR.MSTR 置“1”将 SPI 置于主机方式。当 SPI 处于主机方式时，使能 SPI(将 SPI_CR.SPEN 置“1”)同时向 SPI 数据寄存器 SPI_DATA 写入一个字节时，数据传输开始。SPI 主器件立即在 MOSI 线上串行移出数据，同时在 SCK 上提供串行时钟，在传输结束后 SPI_SR.SPIF 中断标志被置为“1”。如果中断被允许，将产生一个中断请求。在全双工应用中，当 SPI 主器件向从器件发送数据时，被寻址的 SPI 从器件可在 MISO 线上向主器件发送数据。因此 SPIF 标志既作为发送完成标志又作为接收数据准备好标志。处理器通过读 SPI_DATA 寄存器得到接收到的数据。

26.3.1.1 操作流程

1 端口配置；配置端口控制器，把 SCK，MISO，MOSI 信号映射到正确的引脚，并设定为正确的输入/输出状态。

2 主机模式下由寄存器 SPI_SSN.SSN 的值决定片选信号 SPI_SSN 的电平

3 SPI 波特率配置，设置 SPI_CR.SPR2，SPI_CR.SPR1，SPI_CR.SPR0

4 串行时钟配置，设置 SPI 时钟极性 SPI_CR.CPOL，时钟相位 SPI_CR.CPHA。详细见 SPI_CR 寄存器

5　主机方式配置，SPI_CR.MSTR=1

6　SPI 使能打开，SPI_CR.SPEN=1

7　从机选择，配置 SPI_SSN.SSN=0；

8　启动发送数据，要发送给从机的数据写到 SPI 数据寄存器 SPI_DATA 中

9　等待发送/接收数据完成，准备发送/接收下一个数据

以下是中断服务程序：

10　读 SPI_DATA 寄存器，也就是接收从机发送过来的数据。

注意第 10 步完成后 SPI_SR.SPIF 中断被清“0”，如果要连续不断的发送/接收数据重复第 8、9 步骤即可。

在多机通信中，SSN 引脚可以用 GPIO 代替。

26.3.1.2　时序

使用 SPI 控制寄存器 SPI_CR 中的时钟极性 CPOL 和相位 CPHA，串行时钟可以 4 种组合中选择其一。SPI_CR. CPOL 是在 SPI 总线空闲时，SCK 是处于高电平还是低电平。SPI_CR.CPHA 是选择两种时钟相位(采样数据所用的边沿)一种。主机和从机必须配置为使用相同的时钟相位和极性。波特率的设置只对主机有效，对从机的波特率设置会被忽略掉。主方式数据/时钟时序如下图：

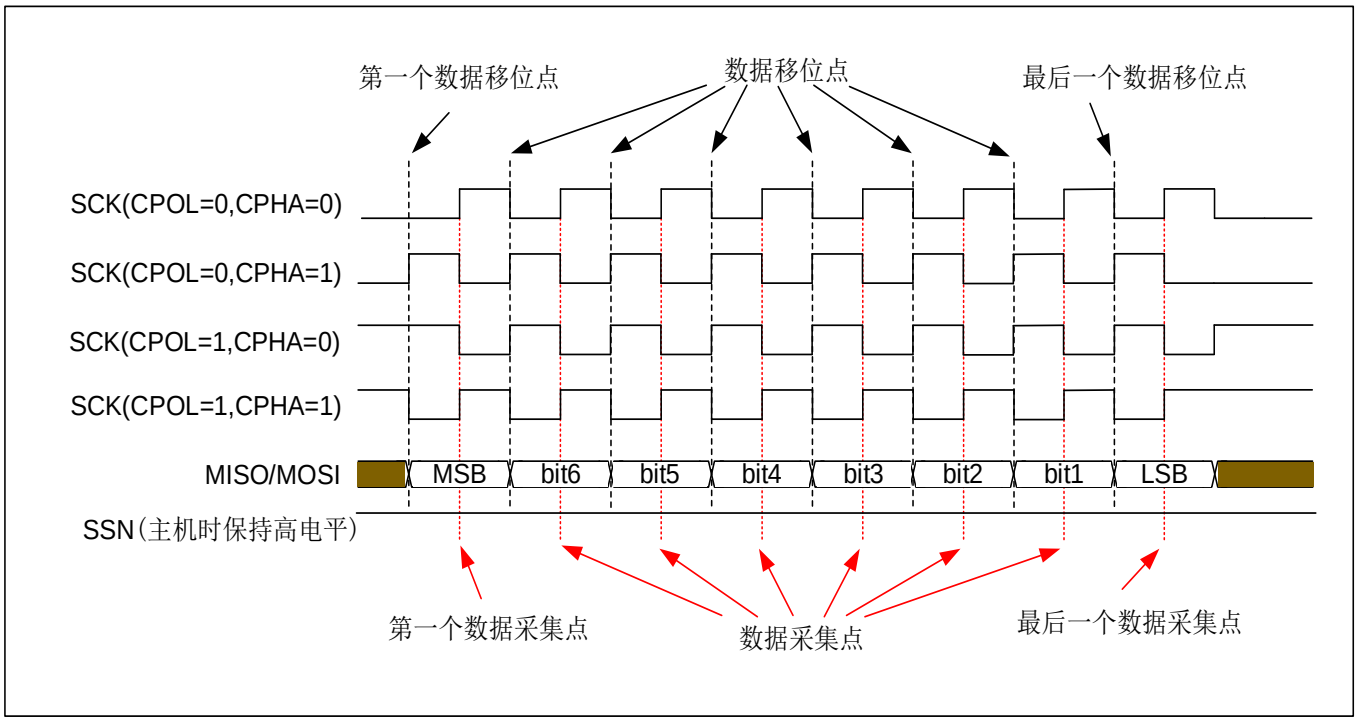


图 26-1 主机方式数据/时钟时序图

26.3.2 SPI 从机方式

当 SPI 被使能时未被配置为主器件时，它将作为 SPI 从机方式。通过将主机/从机控制位 SPI_CR.MSTR 置 0，将 SPI 置于从机方式。当 SPI 处于从机方式时，由主器件控制串行时钟(SCK)，从 MOSI 移入数据。SPI 逻辑中的计数器对 SCK 边沿计数，当 8 位数据移位完成后 SPI_SR.SPIF 标志被置 1。通过读 SPI_DATA 得到接收到的数据，读取 SPI_DATA 寄存器的同时也将清除 SPI_SR.SPIF 标志位。从器件不能启动总线发送数据功能，通过写 SPI_DATA 来预装要发送给主器件的数据，在主器件 SCK 的作用下，一位一位的移到 MISO 线上发动给主器件。

26.3.2.1 操作流程

1 端口配置；配置端口控制器，把 SCK，MISO，MOSI 信号映射到正确的引脚，并设定为正确的输入/输出状态。
2 从机模式下由管脚控制寄存器选择一个 GPIO 作为片选信号的来源，参见管脚控制寄存器(SYSCON_PORTCR)
3 SPI 波特率配置，设置 SPI_CR.SPR2，SPI_CR.SPR1，SPI_CR.SPR0
4 串行时钟配置，设置 SPI 时钟极性 SPI_CR. CPOL，时钟相 SPI_CR.CPHA。详细见 SPI_CR 寄存器
5 从机方式配置，SPI_CR.MSTR=0
6 SPI 使能打开，SPI_CR.SPEN=1
7 要发送给主机的数据写到 SPI 数据寄存器 SPI_DATA 中
8 等待发送/接收数据完成，准备发送/接收下一个数据

以下是中断服务程序：

9 读 SPI_DATA 寄存器，也就是接收主设备发来的数据，同 SPI_SR.SPIF 中断标志位被清“0”

注意：
当从机时钟相位 SPI_CR.CPHA 配置为 0 时，主机每次拉低 SPI_SSN 信号，只能向从机传输一个字节的数据。如果主机每次拉低 SPI_SSN 信号，要向从机连续传输数据时，传输完成中断发生后需要至少等待 1/2 个 SCK 周期才能进行下一次 SPI_DATA 写操作。

注意：
如果要连续不断的发送/接收数据重复第 7、8 步骤即可。

26.3.2.2 时序

从方式数据/时钟时序如下图：

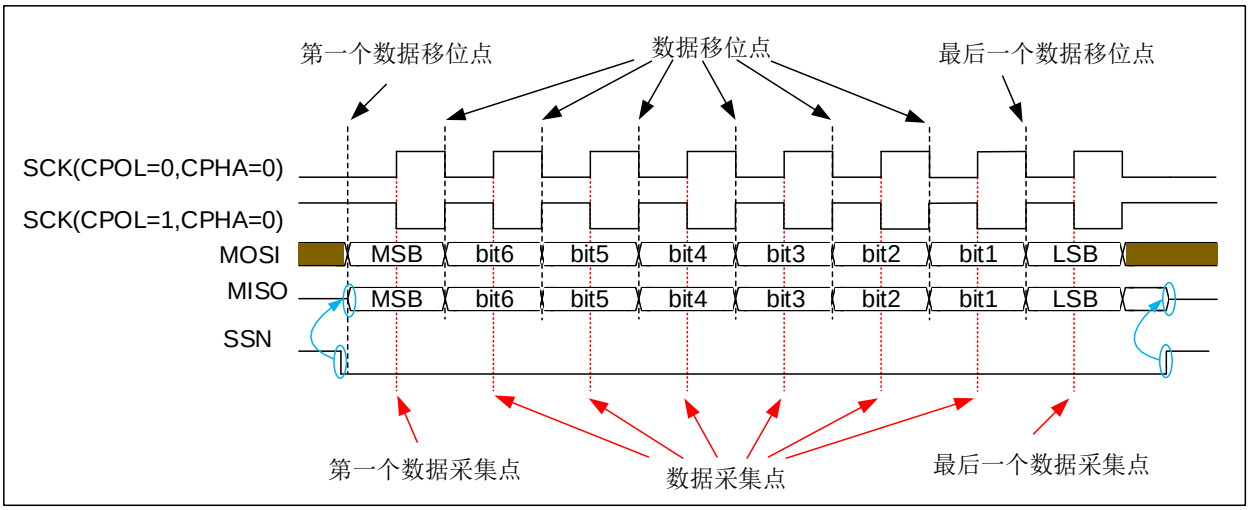


图 26-2 从机方式数据/时钟时序图(CPHA=0)

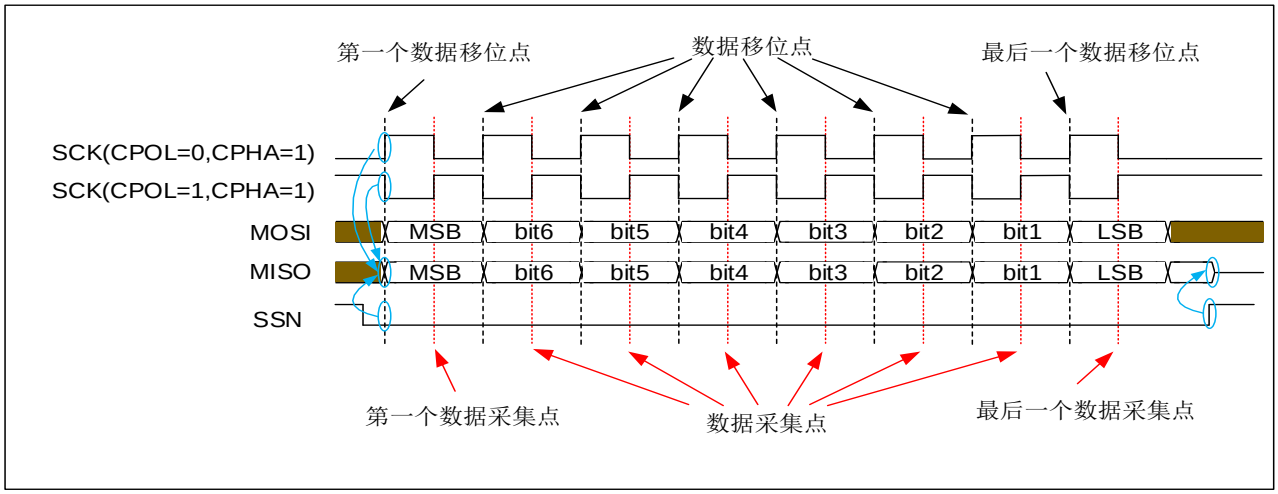


图 26-3 从机方式数据/时钟时序图(CPHA=1)

26.4 SPI 中断

如果 SPI 中断被允许（直接使能 SPI 的 NVIC 中断），SPI 传输完成中断标志位 SPI_SR.SPIF 置“1”时将产生中断；或者在 SPI 的主机模式错误中断标志位 SPI_SR.MDF 置“1”也将产生中断。

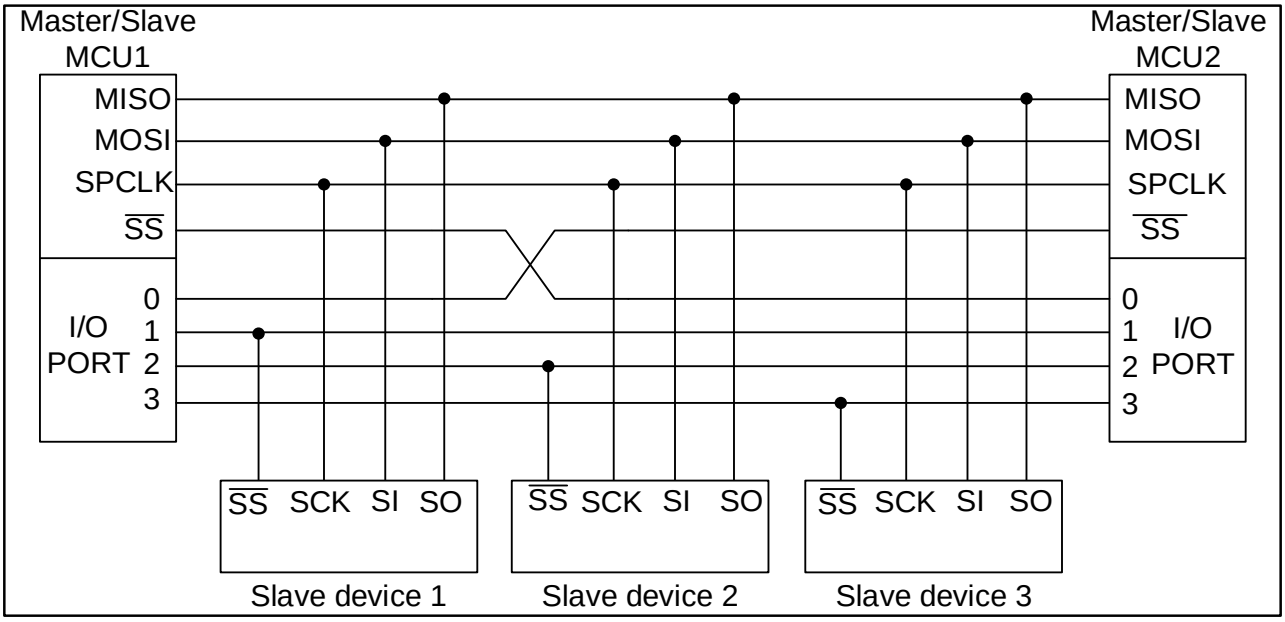
- 在每次字节传送结束，SPI 传输完成中断标志位 SPI_SR.SPIF 都会被硬件自动置“1”。
- 当 SPI 被配置为主机方式时，并且已经开始发送数据时，如果外部 SSN 输入为低电平，这时候 SSN 输入电平和 SPI 工作方式相冲突，SPI 的主机模式错误中断标志位 SPI_SR.MDF 被硬件自动置“1”。

26.5 多主机/多从机模式

本产品在 SPI 单主机单从机系统里作为主机时可以配置 SPI 片选配置寄存器 SPI_SSN，输出高低电平信号到 SPI_SSN 引脚。作为从机时，可以配置 9.2.3 管脚控制寄存器(SYSCON_PORTCR)选择一个 GPIO 引脚作为 SPI_SSN 的来源。

在多主机多从机系统里，如下图所示，必须按照相应流程配置。

当系统为单主机多从机时，可以使用 SPI_SSN 引脚作为从机 1 的片选信号，其他从机的片选信号通过 GPIO 引脚连接。当系统为多主机多从机时，所有的从机片选信号都通过 GPIO 引脚连接，主机还必须通过 GPIO 引脚和其他主机的 SPI_SSN 信号相连，来监测总线是否被占用。



26.6 SPI 寄存器列表

SPI0 基地址：0x40000800

SPI1 基地址：0x40005800

表 26-1 寄存器列表

偏移地址	名称	描述	复位值
0x00	SPIx_CR	SPI 配置寄存器	0x00000014
0x04	SPIx_SSN	SPI 片选配置寄存器	0x00000001
0x08	SPIx_SR	SPI 状态寄存器	0x00000000
0x0C	SPIx_DATA	SPI 数据寄存器	0x00000000

26.7 SPI 寄存器说明

26.7.1 SPI 配置寄存器(SPIx_CR)

偏移地址：0x00
复位值：0x00000014

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留							CNTS	SPR2	SPEN	保留	MSTR	CPOL	CPHA	SPR1	SPR0
							R/W	R/W	R/W		R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:9	Res	保留	0x0	–
8	CNTS	连续传输使能 0：关闭 1：打开	0x0	R/W
7	SPR2	波特率选择位 2,图 26-3 从机方式数据/时钟时序图(CPHA=1)	0x0	R/W
6	SPEN	模块使能： 0：无效 1：使能	0x0	R/W
5	Res	保留	0x0	–
4	MSTR	主机/从机模式选择 0：从机模式 1：主机模式	0x1	R/W
3	CPOL	时钟极性选择寄存器 0：低 1：高	0x0	R/W
2	CPHA	时钟相位选择寄存器 0：第一边沿 1：第二边沿	0x1	R/W
1	SPR1	波特率选择位 1,图 26-3 从机方式数据/时钟时序图(CPHA=1)	0x0	R/W
0	SPR0	波特率选择位 0,参考表 26-2 波特率配置表	0x0	R/W

表 26-2 波特率配置表

SPR2	SPR1	SPR0	SPI_CLK Rate
0	0	0	Fsys/2
0	0	1	Fsys/4
0	1	0	Fsys/8
0	1	1	Fsys/16
1	0	0	Fsys/32

1	0	1	Fsys/64
1	1	0	Fsys/128

注意：SPI 波特率最大不能超过 4MHz

26.7.2 SPI 片选配置寄存器(SPIx_SSN)

偏移地址：0x04
复位值：0x00000001

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															SSN
															R/W

位	标记	功能描述	复位值	读写
31:1	Res	保留	0x0	–
0	SSN	SSN 输出值,在主机模式下，软件配置 SSN 值控制 SPI_SSN 端口电平高低 1：高电平 0：低电平	0x1	R/W

注意：SPI 作从机模式用时，从机片选信号来源请参考系统控制章节中的 SYSCON_PORTCR 寄存器

26.7.3 SPI 状态寄存器(SPIx_SR)

偏移地址：0x08
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								SPIF	WCOL	SSERR	MDF	保留			
								RO	RO	RO	RO				

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7	SPIF	传输结束中断标志，包括发送完成标志及接收完成标志 读取 SPI_DATA 寄存器时，硬件清零此标志位 1：发送/接收完成 0：发送/接收完成未发生	0x0	RO
6	WCOL	写冲突标志，读取 SPI_DATA 寄存器时，硬件清零此标志位	0x0	RO
5	SSERR	从机模式 SSN 错误标志	0x0	RO
4	MDF	主机模式错误标志； 写 SPIx_CR 寄存器时，硬件清零此标志位	0x0	RO
3:0	Res	保留	0x0	–

26.7.4 SPI 数据寄存器(SPIx_DATA)

偏移地址：0x0c
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								SPIDATA							
								R/W							

位	标记	功能描述	复位值	读写
31:8	Res	保留位	0x0	–
7:0	SPIDATA	数据寄存器在发送模式，写发送值到这个寄存器；在接收模式，从这个寄存器读取接收值	0x0	R/W

27 1-Wire 接口

27.1 单总线协议(1-Wire)

主机和从机通过 1 根线进行通信，在一条总线上可挂接的从器件数量几乎不受限制。

27.1.1 特点

它采用单根信号线，既可传输时钟，又能传输数据，而且数据传输是双向的；

端口一定要配置为开漏输出。

27.1.2 优点

单总线技术具有线路简单，硬件开销少，成本低廉，便于总线扩展和维护等。

27.2 单总线通信过程

27.2.1 初始化

初始化过程 = 复位脉冲 + 从机应答脉冲。

主机通过拉低单总线 480 ~ 960 us 产生复位脉冲，然后释放总线，进入接收模式。主机释放总线时，会产生低电平跳变为高电平的上升沿，单总线器件检测到上升沿之后，延时 15 ~ 60 us，单总线器件拉低总线 60 ~ 240 us 来产生应答脉冲。主机接收到从机的应答脉冲说明单总线器件就绪，初始化过程完成。

初始化时序图如图 27-1 初始化过程中的复位与应答脉冲所示：

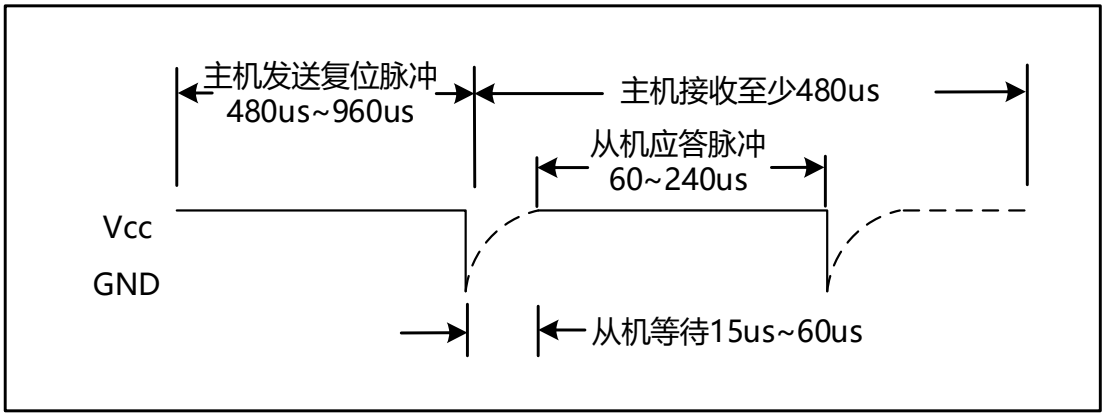


图 27-1 初始化过程中的复位与应答脉冲

27.2.2 写时间间隙

写时间隙有两种，包括写 0 的时间隙和写 1 的时间隙。

当数据线拉低后，在 15 ~ 60us 的时间窗口内对数据线进行采样。如果数据线为低电平，就是写 0，如果数据线为高电平，就是写 1。主机要产生一个写 1 时间隙，就必须把数据线拉低，在写时间隙开始后的 15 us 内允许数据线拉高。主机要产生一个写 0 时间隙，就必须把数据线拉低并保持 60 us。

写时间隙时序图如图 27-2 单总线通信协议中写时间隙时序图所示：

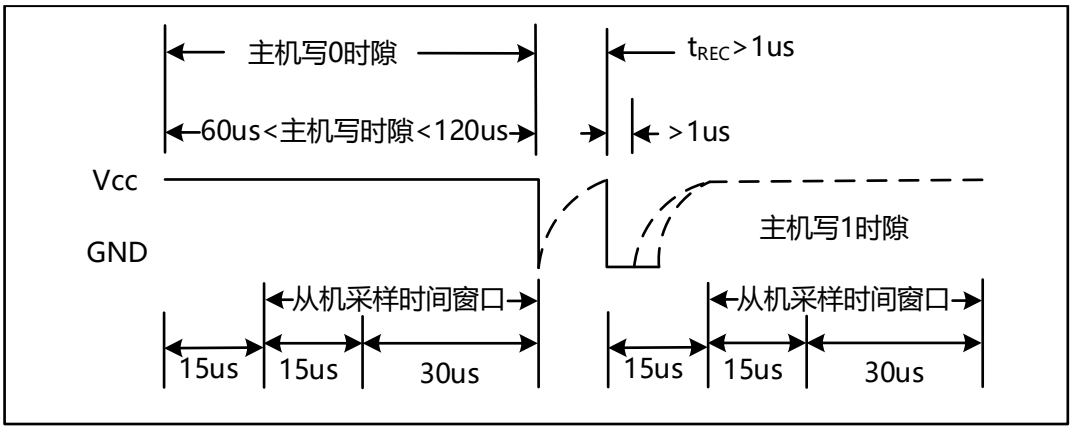


图 27-2 单总线通信协议中写时间隙时序图

27.2.3 读时间间隙

当主机把总线拉低时，并保持至少 1us 后释放总线，必须在 15us 内读取数据。

读时间隙时序图如图 27-3 单总线通信协议中读时间隙时序图所示：

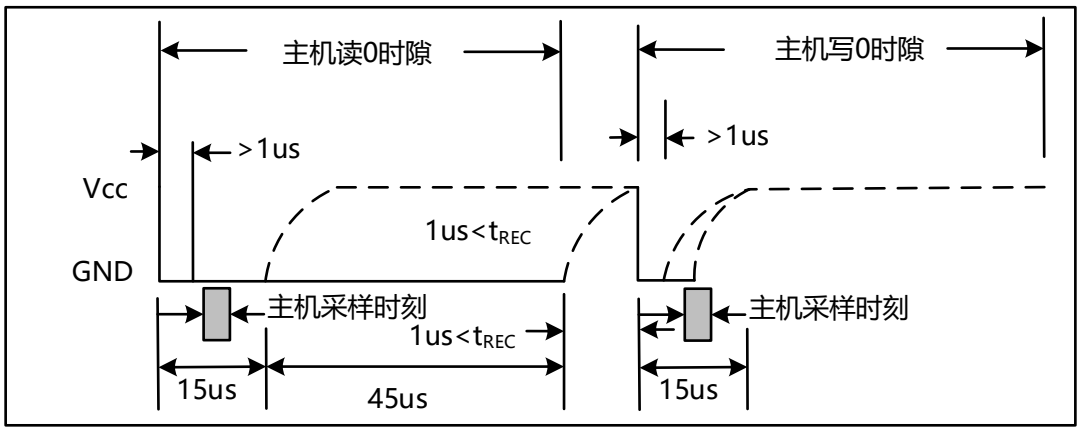


图 27-3 单总线通信协议中读时间隙时序图

27.3 配置说明

27.3.1 初始化配置说明

1) 配置 GPIO 相应的管脚复用 1-Wire 管脚，并且配置为开漏模式；
2) 配置 OWIRE_CR.CLKDIV 寄存器，设置 1-Wire 模式时钟选择;
3) 配置 1-Wire Reset 宽度控制寄存器 OWIRE_RSTCNT,设置 1-Wire 主发送复位时间(480us~960us)；
4) 配置 1-Wire 应答宽度计数寄存器 OWIRE_PRESCNT,设置 1-Wire 从应答设定计数值(60us~240us)；
5) 配置 1-Wire 中断使能寄存器 OWIRE_INTEN.INITEN 使能初始化完成中断；
6) 配置 OWIRE_CR.EN 寄存器，使能 1-Wire 模块；
7) 配置 1-Wire 总线操作命令寄存器 OWIRE_CMD,设置 Initial 指令；
8) 系统进入中断子程序，在中断子程序中配置状态清除寄存器 OWIRE_INTCLR，清除相应的中断标志；

27.3.2 读数据配置说明

1) 配置 1-Wire Bit rate 计数器 OWIRE_BITRATECNT，设置 1bit 数据宽度(15us~60us)；
2) 配置 1-Wire 主器件读/写 PULL0 驱动时间 OWIRE_DRVCNT，设置驱动时间宽度(0us~15us);
3) 配置 1-Wire 主器件读采样时间设定 OWIRE_RDSMPCNT,设置读采样时间(1us~15us);
4) 配置 1-Wire Recover Time 计数区间值 OWIRE_RECCNT，设置 RECOVER 时间为(T_{rec}>1us);
5) 配置 1-Wire 中断使能寄存器 OWIRE_INTEN.RXDONEEN 使能接收完成中断；
6) 配置 1-Wire 总线操作命令寄存器 OWIRE_CMD,设置 RX 指令；
7) 系统进入中断子程序，中断子程序中配置状态清除寄存器 OWIRE_INTCLR，清除接收完成中断 FLAG，并读 1-Wire 数据寄存器 OWIRE_DATA；

27.3.3 写数据配置说明

配置 1-Wire Bit rate 计数器 OWIRE_BITRATECNT，设置 1bit 数据宽度(15us~60us)；
配置 1-Wire 主器件读/写 PULL0 驱动时间 OWIRE_DRVCNT，设置驱动时间宽度(0us~15us);
配置 1-Wire Recover Time 计数区间值 OWIRE_RECCNT，设置 RECOVER 时间为(T_{rec} >1us);
配置 1-Wire 中断使能寄存器 OWIRE_INTEN.TXDONEEN 使能发送完成中断；
写数据到 1-Wire 数据寄存器 OWIRE_DATA；
配置 1-Wire 总线操作命令寄存器 OWIRE_CMD,设置 TX 指令；
数据发送完后系统进入中断子程序，中断子程序中配置状态清除寄存器 OWIRE_INTCLR，清除 TX 完成中断 FLAG；

27.4 寄存器列表

基地址：0x4000 3800

偏移地址	名称	描述	默认值
0x00	OWIRE_CR	1-wire 模块控制寄存器	0x00000000
0x04	OWIRE_NFCR	1-Wire 输入端子滤波控制寄存器	0x00000000
0x08	OWRIE_RSTCNT	1-Wire Master Reset pulse 宽度计数寄存器	0x00000000
0x0c	OWIRE_PRESCNT	1-Wire Device Presence Pulse 宽度计数寄存器	0x00000000
0x10	OWIRE_BITRATECNT	1-Wire Bit rate 设计计数器	0x00000000
0x14	OWIRE_DRVCNT	1-Wire 主器件读/写 PULL0 驱动时间	0x00000000
0x18	OWIRE_RDSMPCNT	1-Wire 主器件读的采样时间设定	0x00000000
0x1C	OWIRE_RECCNT	1-Wire Recover Time 计数区间值	0x00000000
0x20	OWIRE_DATA	1-Wire 数据寄存器	0x00000000
0x24	OWIRE_CMD	1-Wire 总线操作命令寄存器	0x00000000
0x28	OWIRE_INTEN	1-wire 中断使能寄存器	0x00000000
0x2C	OWIRE_SR	1-wire 状态寄存器	0x00000000
0x30	OWIRE_INTCLR	1-wire 中断状态清除寄存器	0x00000000

27.5 寄存器说明

27.5.1 1-Wire 模块控制寄存器(OWIRE_CR)

地址偏移：0x00

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								RDMODE	MSBFIRST	EN	SIZE	保留		CLKDIV	
								R/W	R/W	R/W	R/W			R/W	

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7	RDMODE	0：普通模式 1：写 0/读 0 间隙相等	0x0	R/W
6	MSBFIRST	字节发送的位模式设定 0：LSB(bit0)send/receive first 1：MSB(bit7)send/receive first 当 OWIRE_CR.SIZE=0 时，该位要设定为 0;	0x0	R/W
5	EN	4. wire 模块使能控制位 0：1-wire 模块停止 1：1-wire 模块使能	0x0	R/W
4	SIZE	数据处理位数控制位 0：单次处理 1 bit(Bit mode) 1：单次处理 8 bit(Byte mode)	0x0	R/W
3:2	Res	保留	0x0	–
1:0	CLKDIV	计数器用时钟源选择位 00：F_{pclk} 01：$F_{pclk}/2$ 10：$F_{pclk}/4$ 11：$F_{pclk}/16$	0x0	R/W

27.5.2 1-Wire 输入端子滤波控制寄存器(OWIRE_NFCR)

地址偏移：0x04
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留											NFEN	保留		NFDIV	
											R/W			R/W	

位	标记	功能描述	复位值	读写
31:5	Res	保留	0x0	–
4	NFEN	输入端子滤波使能控制位 0：滤波功能无效 1：滤波功能有效	0x0	R/W
3:2	Res	保留	0x0	–
1:0	NFDIV	输入端子滤波时钟源选择位 00：F_{pclk} 01：$F_{pclk}/2$ 10：$F_{pclk}/4$ 11：$F_{pclk}/8$	0x0	R/W

27.5.3 1-Wire RESET 宽度控制寄存器(OWIRE_RSTCNT)

地址偏移：0x08
复位值：0x0000 0000

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

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	–
15:0	RSTCNT	主发送复位时间设定计数值	0x0	R/W

27.5.4 1-Wire Presence Pulse 宽度计数寄存器(OWIRE_PRESCNT)

地址偏移：0x00C
复位值：0x0000 0000

位	标记	功能描述	复位值	读写
31:13	Res	保留	0x0	–
12:0	PRESCNT	从应答时间设定计数值	0x0	R/W

27.5.5 1-Wire Bit rate 设计计数器(OWIRE_BITRATECNT)

地址偏移：0x010
复位值：0x0000 0000

位	标记	功能描述	复位值	读写
31:12	Res	保留	0x0	–
11:0	BITRATECNT	位数据宽度时上设定计数器	0x0	R/W

27.5.6 1-Wire 主器件读/写 PULL0 驱动时间(OWIRE_DRVCNT)

地址偏移：0x014
复位值：0x0000 0000

位	标记	功能描述	复位值	读写
31:9	Res	保留	0x0	–
8:0	DRVCNT	主机读/写 PULL0 的时长设定计数器	0x0	R/W

27.5.7 1-Wire 主器件读采样时间设定(OWIRE_RDSMPCNT)

地址偏移：0x018
复位值：0x0000 0000

位	标记	功能描述	复位值	读写
31:9	Res	保留	0x0	–
8:0	RDSMPCNT	主器件读采样时间设定	0x0	R/W

27.5.8 1-Wire Recover Time 计数区间值(OWIRE_RECCNT)

地址偏移：0x01C
复位值：0x0000 0000

位	标记	功能描述	复位值	读写
31:11	Res	保留	0x0	–
10:0	RECCNT	Recover Time 计数区间值	0x0	R/W

27.5.9 1-Wire 数据寄存器(OWIRE_DATA)

地址偏移：0x020
复位值：0x0000 0000

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	–
7:0	DATA	1bit 模式：只能发送和接收 bit0 8bit 模式：能发送和接收全部 8 个 bit	0x0	R/W

27.5.10 1-Wire 总线操作命令寄存器(OWIRE_CMD)

地址偏移：0x024
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留														CMD	
														R/W	

位	标记	功能描述	复位值	读写
31:2	Res	保留	0x0	–
1:0	CMD	00：保留 01：Initial 10：TX 11：RX	0x0	R/W

27.5.11 1-Wire 中断使能寄存器(OWIRE_INTEN)

地址偏移：0x28
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留												RXDONEEN	TXDONEEN	INITDONE	ACKERREN
												R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:4	Res	保留	0x0	–
3	RXDONEEN	接收完成中断使能 0：无效 1：使能	0x0	R/W
2	TXDONEEN	发送完成中断使能 0：无效 1：使能	0x0	R/W
1	INITEN	初始化完成中断使能 0：无效 1：使能	0x0	R/W
0	ACKERREN	从机应答错误中断使能 0：无效 1：使能	0x0	R/W

27.5.12 1-Wire 状态寄存器(OWIRE_SR)

地址偏移：0x2c
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留												RXDONE	TXDONE	INITDONE	ACKERR
												RO	RO	RO	RO

位	标记	功能描述	复位值	读写
31:4	Res	保留	0x0	–
3	RXDONE	接收完成中断状态 0：接收完成未发生 1：接收完成	0x0	RO
2	TXDONE	发送完成中断标志 0：发送完成未发生 1：发送完成	0x0	RO
1	INITDONE	初始化完成标志 0：初始化完成未发生 1：初始化完成	0x0	RO
0	ACKERR	从机应答错误中断标志 0：从机应答错误未发生 1：从机应答错误发生	0x0	RO

27.5.13 1-Wire 状态清除寄存器(OWIRE_INTCLR)

地址偏移：0x2C
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留												RXDONECLR	TXDONECLR	INITDONECLR	ACKERRCLR
												WO	WO	WO	WO

位	标记	功能描述	复位值	读写
31:4	Res	保留	0x0	–
3	RXDONECLR	接收完成中断清除 写 0：无作用 写 1：清除接收完成中断	0x0	WO
2	TXDONECLR	发送完成中断标志清除 写 0：无作用 写 1：清除发送完成中断	0x0	WO
1	INITDONECLR	初始化完成中断标志清除 写 0：无作用 写 1：清除初始化完成中断	0x0	WO
0	ACKERRCLR	从机应答错误中断标志清除 写 0：无作用 写 1：清除从机应答错误中断	0x0	WO

28 LINFlexD 控制器

28.1 介绍

LINFlexD 控制器是基于主从模式的 LIN 协议总线生成的 IP 模块

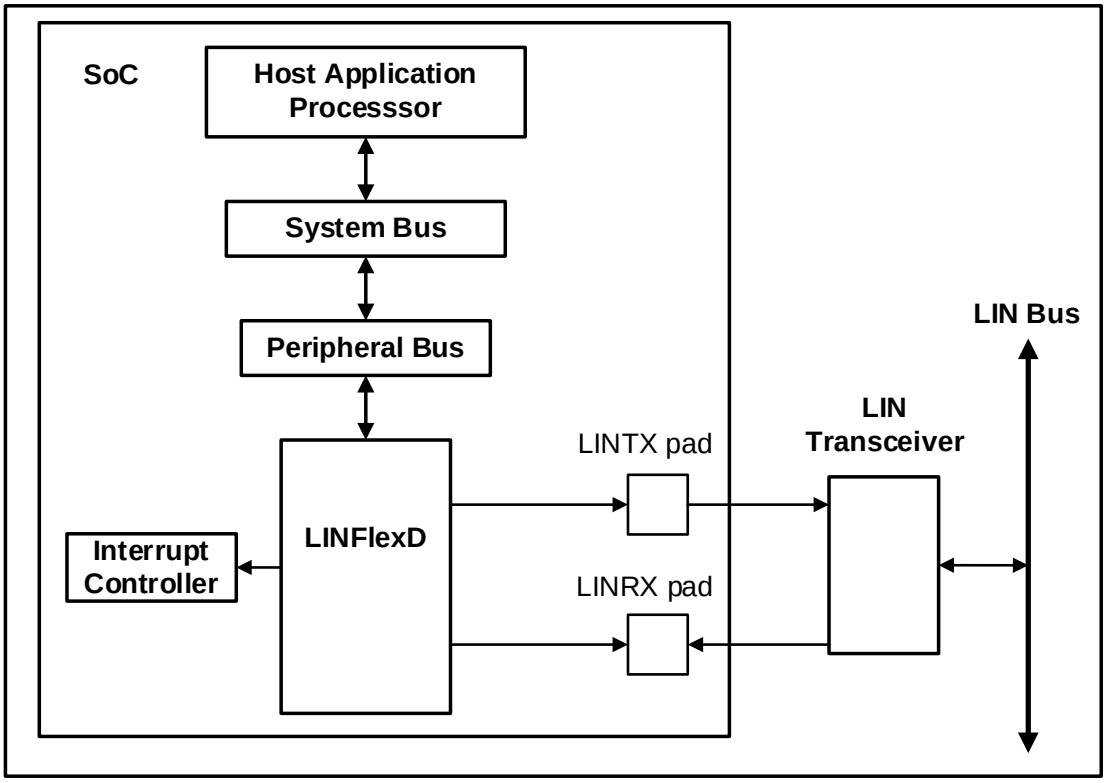


图 28-1 LINFlexD 结构框图

LINFlexD 控制器的设计目的是：用最少的 CPU 负载有效地管理大量 LIN 消息。为了减少主模式下的 CPU 负载，LINFlexD 在软件触发帧头发送后自动处理 LIN 消息，直到在发送模式下发起下一个帧头发送请求，或者直到接收模式下校验和接收完成。

LINFlexD 支持 LIN 协议版本 1.3、2.0、2.1 和 2.2，并为传输/接收数据提供了一个 8 字节的缓冲区。

LINFlexD 还支持 UART 模式，用于 8 位、9 位或 12 位数据帧的 UART 传输。

28.2 特性

5. 在 LIN 和 UART 模式中，LINFlexD 的共同特点：

- 支持小数波特率发生器；
- 3 种省电和配置寄存器锁定的工作模式：初始化、正常和睡眠；
- 测试模式：回环模式；
- 可屏蔽中断；

2、LIN 模式主要特点：

- 支持 LIN 协议版本 1.3、2.0、2.1 和 2.2；
- 最高可达 20kbit/s 的比特率(LIN 协议)；
- 主/从模式；
- 经典型和增强型校验和；
- 用于发送/接收的单个 8 字节缓冲区或 FIFO；
- 超时管理；
- 标识符过滤器；
- 支持最多 16 个可能的标识符；
- 具有自主消息处理的主模式；
- 显性电平检测的唤醒事件；
- 高级 LIN 错误检测；
- 帧头，应答和帧超时；
- 从机模式：自主帧头处理、自主发送/接收数据处理；
- 标识符过滤器，用于在从模式下自主处理消息；
- 用于计算波特率的单独时钟；

3、UART 模式主要特点：

- 全双工通信；
- 用于计算波特率的单独时钟；
- 波特率是波特时钟、LINBRR 和 LINFBRR 寄存器和函数；
- 7/8 位数据，奇偶校验；
- 1/2/3 个停止位；
- 12 位+奇偶校验接收；
- 4 字节缓冲区用于接收，4 字节缓冲区用于发送；
- 用于超时管理的 12 位计数器；
- 可达到的最大波特率是 ipg_baud_clk/4Mbit/s。

28.3 操作

28.3.1 LIN 协议

28.3.1.1 帧

一帧由主机任务提供的帧头和从机任务提供的应答组成。帧头依次包括一个同步间隔段，一个同步段和一个标识符。与标识符相关的从机任务提供应答。应答由一个数据段与校验和组成。从机任务应答接收指定对应标识符的数据并验证校验和。

LIN 帧的发送/接收如图 28-2 和图 28-3 所示。

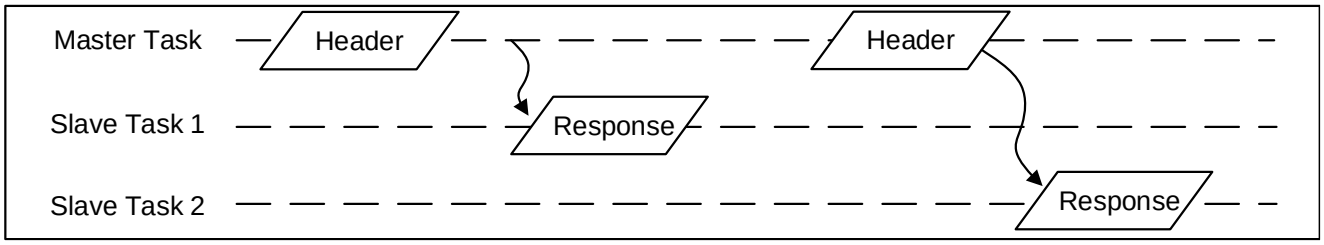


图 28-2 帧

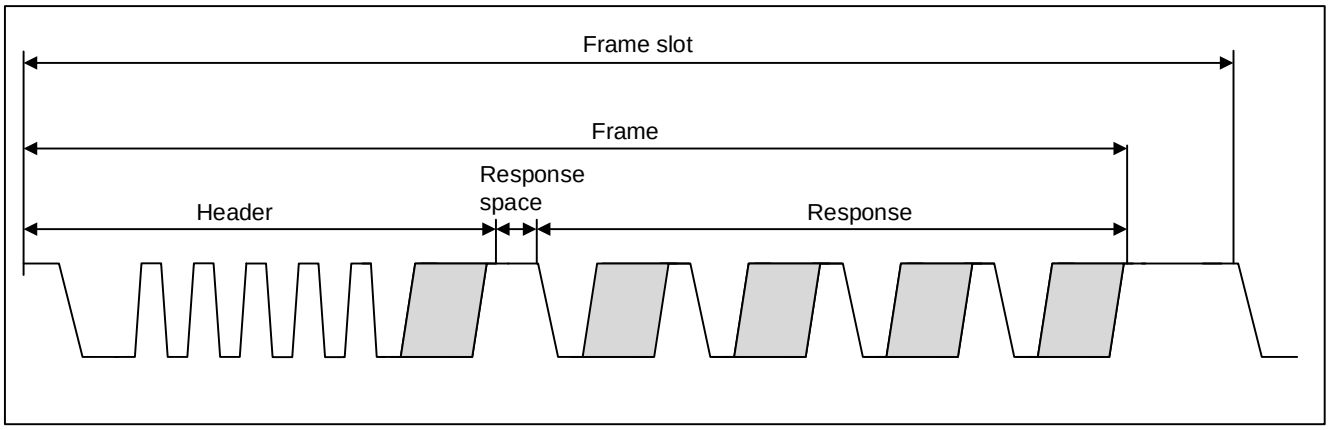


图 28-3 LIN 帧的结构

28.3.1.2 数据字节段

每个字节的传输如图 28-4 所示。数据的 LSB 先发送，MSB 最后发送。起始位被编码为数值为 0 的位(显性)，停止位被编码为数值为 1 的位(隐性)。

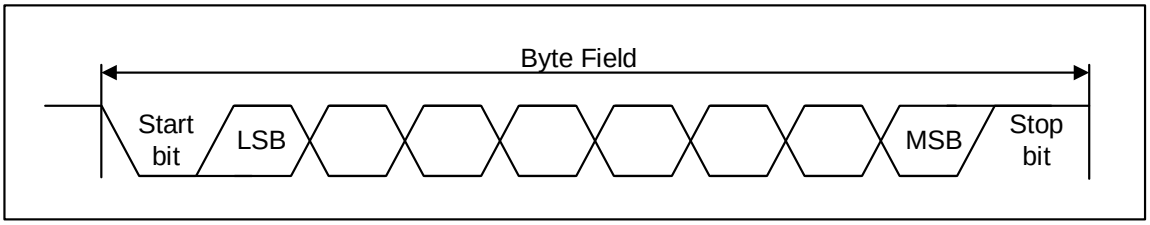


图 28-4 字节段的构成

28.3.1.3 间隔段

间隔段符表示一个新帧的开始。它是唯一不符合图 28-5 的字段。间隔段符总是由主节点产生，并且应该是至少 13 比特的显性电平，后面是间隔段间隔符，如图 28-6 所示。间隔

符的长度必须至少是一个标称比特时间。

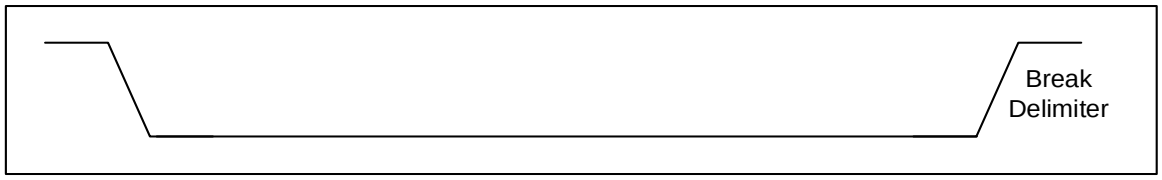


图 28-5 间隔段

28.3.1.4 同步字节段

同步字节段是一个字节字段，数据值为 0x55。

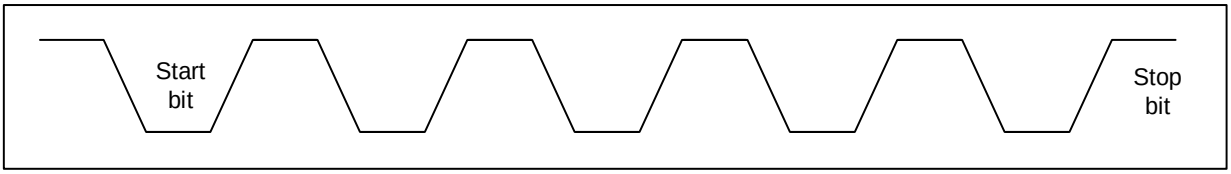


图 28-6 同步字节段

28.3.1.5 标识符

标识符字段由两个子字段组成，即标识符和标识符奇偶校验。位 0 到位 5 是标识符。位 6 和位 7 表示奇偶性：

P0 = ID0 XOR ID1 XOR ID2 XOR ID4

P1 = NOT (ID1 XOR ID3 XOR ID4 XOR ID5)

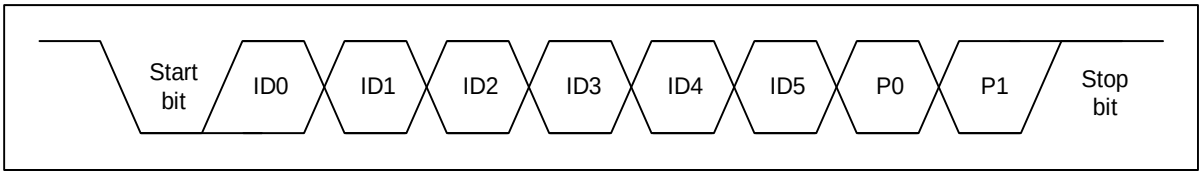


图 28-7 标识符字段

28.3.1.6 校验和

帧的最后一个字段是校验和，校验和包含所有数据字节和标识符的带位二进制之和并求反。仅对数据字节进行校验计算被称为经典型校验和，用于与 LIN1.3(及以下)从机的通信。对数据字节和受保护的标识符字节的校验计算被称为增强型校验和，用于与 LIN2.0(及更高)从机的通信。

28.3.2 LINFlexD 特性

28.3.2.1 操作模式

LINFlexD 有三种操作模式：初始化模式、正常模式和睡眠模式。硬件复位后，LINFlexD 进入休眠模式，以降低功耗。

下图为工作模式的状态图。

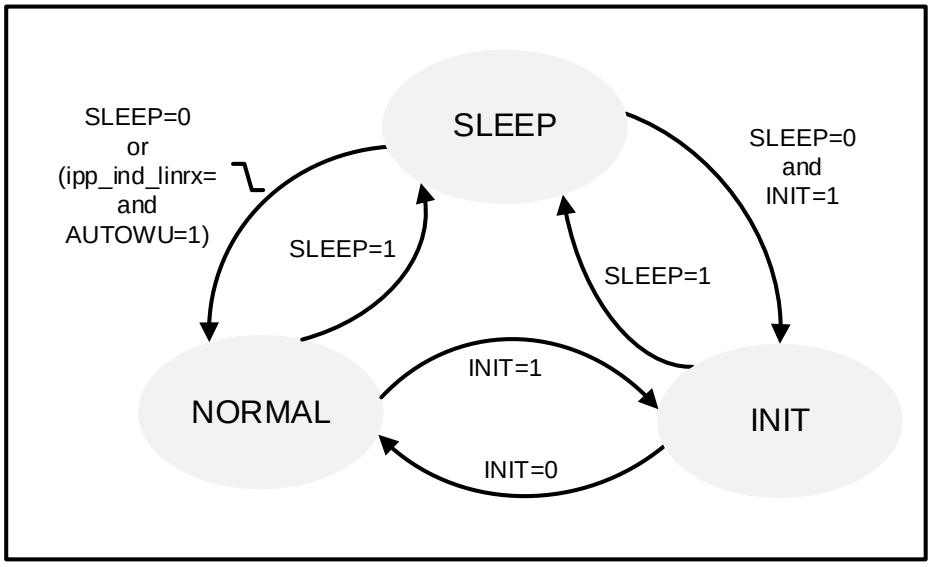


图 28-8 操作模式

初始化模式(INIT)：

进入初始化模式，需要软件设置 LINCR1.INIT 位；要退出初始化模式，需要软件复位 INIT 位。

当 LINFlexD 处于初始化模式时，LIN 总线的所有传输都会停止，LIN 总线输出 ipp_do_lintx 为高电平。如果进行总线传输时被触发进入初始化模式，则传输将中止。因此，在设置 INIT 位之前检查 LIN 状态。要初始化 LINFlexD 控制器需要：

1、设置波特率寄存器；

2、复位 UART 位；

3、选择主从模式；

4、设置校验和控制位；

5、初始化标识符列表（从模式）。

正常模式(NM)：

在初始化 LINFlexD 后，可以软件清零 INIT 位，将 LINFlexD 设置为正常模式。

睡眠模式(SM):

LINFlexD 的睡眠模式可以降低功耗。LINCR1 中设置 SLEEP 位来进入睡眠模式，且清零 SLEEP 位可退出。

如果 LIN 总线上检测到 150μs 的唤醒脉冲，LINFlexD 会从睡眠模式中唤醒。

28.3.2.2 回环模式

设置 LINCR1.BKM 位，LINFlexD 会进入回环模式。在回环模式下，LINFlexD 接收自己发送的标识符和数据，并将其分别写入 BIDR 寄存器和数据缓冲区。回环模式提供了自检功能，因此在回环模式下可以检查位传输错误。

回环模式下，LINFlexD 会忽略 ipp_ind_linrx 信号，从 Tx 输出到 Rx 输入有一个内部反馈。lpp_do_lintx 信号可以通过用户实现的 GPIO 函数从芯片级的 LINTX 引脚上断开。

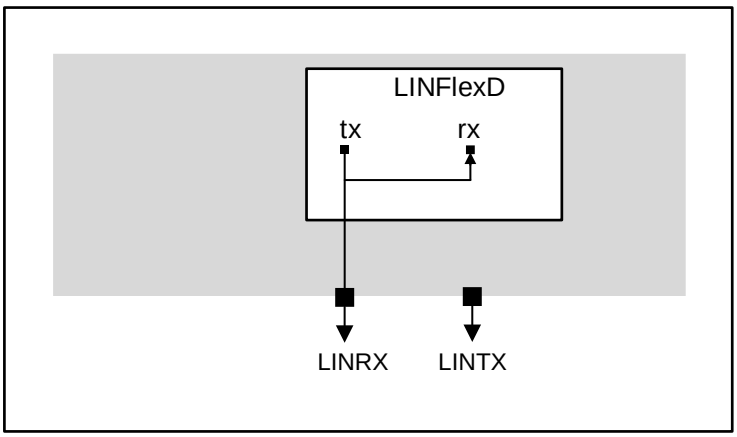


图 28-9 回环模式

28.3.2.3 主模式

LINCR1.MME 位置 1 选择主模式。

帧头传输：

根据 LIN 协议，LIN 总线上的任何通信都由主机任务发送帧头触发。主机任务节点发送帧头，从机任务节点应答。要发送帧头，要在 BIDR 寄存器中设置标识符(ID)、数据字段长度(DFL)、传输方向(DIR)和经典校验使能(CCS)，然后在 LINCR2 中设置 HTRQ 位。帧头发送开始后，在当前帧完成之前，不要修改 BIDR 寄存器。传输的 ID 也会被主节点接收并同步到 BIDR。

数据发送：

根据主节点发送的与数据对应的标志符，从机任务会在帧的应答部分发送该数据。因此，在请求帧头发送之前，软件必须向 LINFlexD 提供数据。要传输的数据存储在消息缓冲区 BDRL 和 BDRM 中，要传输的字节数取决于 BIDR.DFL。可通过软件设置 BIDR.CCS 位来配置每次数据传输的校验类型（经典型或增强型）。

应答成功发送时，LINSR.DTF 置位；出现传输错误时，LINSR.DTF 不会置位，并在 LINESR 中会显示相应的错误标志。

消息缓冲区的方向由 BIDR.DIR 位控制。发送的数据也由同一个节点接收并同步到缓冲区。

数据接收：

主节点发送带有标识符的帧头，接收到从节点发送的相应的数据。从节点接收到的数据存储在数据缓冲区中，状态存储在 LINSR。

数据接收成功，LINSR.DRF 置位；如果位传输错误，LINSR.DRF 不会置位,并且 LINESR 会显示相应的错误标志。

数据丢弃：

在帧头传输后丢弃数据，需要设置 LINCR2.DDRQ 位。

28.3.2.4 从模式

LINCR1.MME 位清零选择从模式。

发送数据：

帧头接收时，LINSR.HRF 位需置 1，并生成了一个 Rx 中断。软件必须设置：

1、在 BIDR 寄存器中读取接收到的 ID；

2、写数据到 BDRL/BDRM 寄存器；

3、在 BIDR 寄存器中设置 CCS 和 DIR 位；

4、在 BIDR.DFL 中设置传输数据字段的长度；

LINSR.HRF 只能在 LINCR2.DTRQ 置 1 后复位。数据发送请求 DTRQ 只有在 HRF 置位后才有效。只能在帧头接收后才能设置 DTRQ 位。LINSR.RXBUSY 有效后不能设置 DIR 和 DTRQ 位。

或者，通过设置 DIR 位来配置一个或多个标识符过滤器进行发送，并通过在 IFER 中设置使能位来激活。当至少一个标识符过滤器被激活用于数据发送，并且接收到的 ID 与该过滤器相匹配时，将生成一个 Tx 中断。软件可以使用 IFMI 寄存器中的索引直接指向 RAM 中相应的数据阵列，并将数据同步到 BDRM/BDRL。

使用过滤器，可以节省软件读取 BIDR 中的 ID 值并与之匹配，设置数据字段长度和校验和类型所需的处理时间。

发送的数据可由同一个节点接收并同步到缓冲区。

数据接收：

当 LINFlexD 作为数据接收方，在帧头接收时，HRF 标志位置位并产生 Rx 中断信号。软件从 BIDR 寄存器读取接收到的 ID，在接收到第一个数据字节的停止位之前指定数据字段长度。当接收到校验和时，将产生 Rx 中断信号，以便软件从 BDRM/BDRL 读取接收到的数据，并且 LINSR.RMB 将置位。

当至少一个标识符过滤器被激活并设置为接收时，仅在校验和接收之后生成 Rx 中断信号。接收 ID 时不会产生中断请求。

如果软件设置标识符过滤器，当 HRF 置位时，软件可通过设置 LINCR2.DDRQ 来丢弃数据。

注意：对于软件滤波，在接收数据之前软件必须设置校验和的类型(设置 BIDR.CCS 位)。否则，在计算校验和时，将会考虑 CCS 的上一个值。

28.3.2.5 错误

帧头错误：

帧头错误是帧头接收期间产生的错误。错误类型为：

- 同步间隔段间隔符错误(SDEF)；
- 同步字段错误(SFEF)；
- 标识符奇偶校验错误(IDPEF)；

6. 同步间隔段间隔符错误(SDEF)：

同步间隔段间隔符在至少一位时间内必须为 1；否则会因为时间比较短接收方会放弃当前帧头上的同步，然后该帧数据将会丢失，如果设置了 LINIER.HEIE 位，则会生成中断信号。

2、同步字段错误(SFEF)：

检测 SFEF 错误取决于是否启用了自动同步，这由 LINCR1.LASE 位决定。

情况 1：启用自动同步(LINCR1.LASE = 1)

当启用自动同步时，SFEF 标志表示：

(1)同步字段上的偏差误差超出了 LIN 规范，这允许从振荡器和主振荡器之间高达 14%的周期偏差;

(2)在同步字段测量期间发生溢出，导致分频器寄存器溢出;

注意：SFEF 并不表示从节点接收到了不一致的同步字段值（0x55 除外）。

同步字段上的偏差误差：

通过比较当前波特率（相对于从振荡器）与接收到的 LIN 同步段（相对于主振荡器），可以检查偏差误差。此检查基于同步段的第一个下降边和最后一个下降边之间的测量。将此周期偏差称为 D：

(1) 出现 SFEF 错误，D>为 14.0625%;

(2) 无错误，D<14.84375%；

(3) 14.0625%<D<14.84375%，那么同步场可以是一致的或不一致的，这取决于 ipp_ind_linrx 和 ipg_baud_clk 信号之间的去相位。

同步字段测量期间的溢出：

溢出检查是基于对同步字段两边之间的每个位时间的测量。基于当前波特率的比特时间，每个比特检查的时间都要足够大（超过 12 个样本）。

情况 2：禁用自动同步(LINCR1.LASE = 0)

当禁用自动同步时，同步字符将作为正常字符接收。如果接收到的字符为 0x55，则同步字段为正常。如果出现不一致的同步字段，接收方将立即退出 Sync_Field 状态，该帧将被丢弃，并且将 LINESR.SFEF 位置位。

标识符奇偶校验错误(IDPEF)：

在标识符字段中有两个奇偶校验位。奇偶校验位与硬件计算的其他六位标识符的奇偶校验进行检查，根据公式：

P0=ID0 X OR ID1 X OR ID2 X OR ID4

P1=NOT(ID1 X OR ID3 X OR ID4 X OR ID5)

ID 转移到 BIDR 寄存器后(检测到 ID 停止后)检查标识符奇偶校验。在奇偶校验不匹配时，如果 LINCR2.IOPE 置位，接收方立即退出标识符状态。

位错误:

在发送模式下，当从总线读取的值与传输的值不一致时产生位错误。如果收发器延迟小于 1 位时间与 6 个 ipg_baud_clk 周期的时间间隔，则能确保对每个位的位错误检查，例如：

20Kbit/s 时，1 位时间=50μs；

40MHZ 时，6 个 ipg_baud_clk 周期=150ns

(1 位时间)-(6 个 ipg_baud_clk 周期)=49.85μs

在同步间隔段传输期间不检查到位错误。

如果是 LINCR2. IOBE = 1，则在错误位后停止帧的传输；如果 LINCR2.IOBE=0，尽管有位错误,发送方继续发送。如果 LINIER.BEIE=1,则会生成一个中断信号。

注意：如果由于收发器延迟或总线上的错误，同步间隔段间隔符发送后，主机在在一个比特时间内如果没有检测到同步间隔段间隔符，则会生成一串误码中断。在这种情况下，在 BIDR 中 ID 不能被替换。类似地，如果在开始比特发送后的一个比特时间内发送节点没有检测到数据开始的下降沿，则会产生一串比特错误中断。在这种情况下，不能保证 BDRM/BDRL 和 LINCFR 中的数据和校验。

帧错误：

在当前接收到的字符（同步字段、标识符字段、数据字段、校验和字段）的停止位上采样到显性电平时，将标记帧错误。LINFlexD 丢弃当前帧，返回到空闲状态。生成中断信号，LINIER.FEIE 置位。

导致帧错误的字节被送到缓冲区，LINSR.DRF 不会置位。

校验和段错误：

当由硬件设置的校验和与接收到的校验和不一致时，将产生校验和错误标志。

LINFlexD 丢弃接收到的帧并返回到空闲状态。LINIER.CEIE 置位，则会产生中断信号。

溢出错误：

在消息缓冲区已满后(LINSR.RMB 置位)，下一个有效消息接收导致溢出，消息丢失。

LINFlexD 硬件通过设置 LINESR. BOF 位来表示溢出状态。哪条消息丢失取决于缓冲区锁函数控制位 LINCR1.RBLM:

RBLM=0，则缓冲区中的旧消息将被最新的消息覆盖；

RBLM=1，则丢弃最新的消息，而最先前的消息是软件可用的。在从模式下，如果在接收到下一个标识符和 RBLM=1 之前没有释放缓冲区(RMB 没有复位)，则会丢弃 ID 和数据。

超时错误：

在不完全应答（图 35）或在超时应答期间内没有应答（图 36）的情况下，就会出现超时错误。

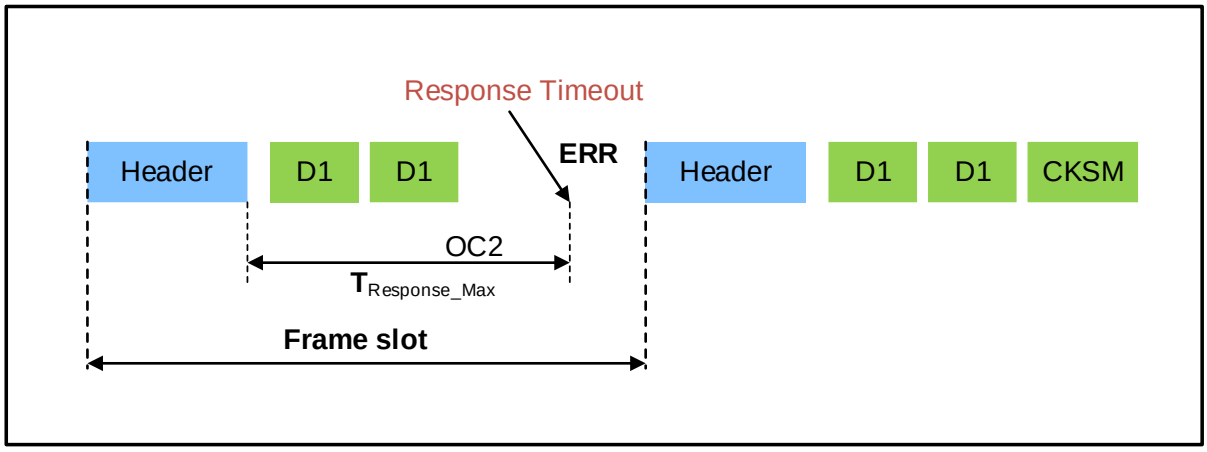


图 28-10 不完全应答（例如，校验和丢失）

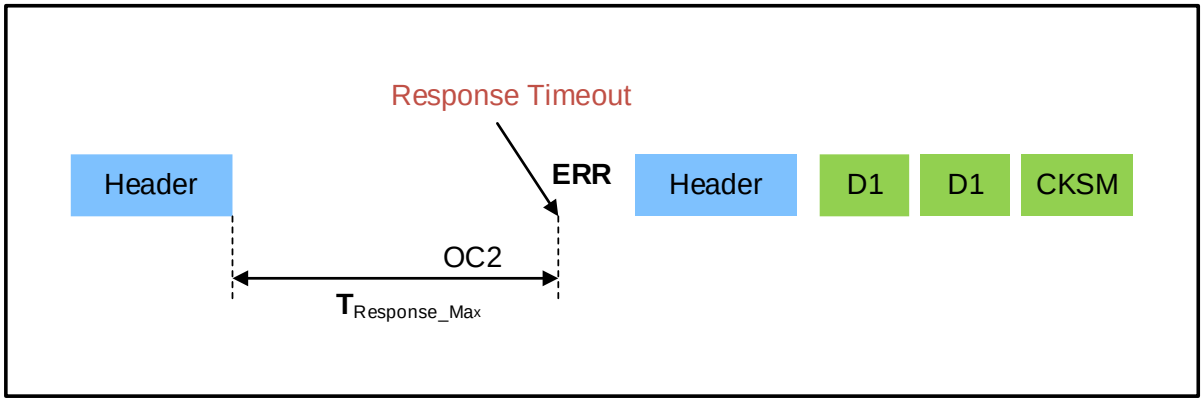


图 28-11 不应答

应答超时机制：

1、主模式：

(1) 在标识符字段的停止位的末端(nominal_time_out=1.4×((DFL+2)×10bit 时间)，OC2 被加载位 nominal_time_out+LINTCSR.CNT 的值；

此加载发生在标识符字段的结尾（而不是在数据字段的开头）。此外，这个取决于 DFL 的正确值会立即加载。OC2 不需要进一步更新。

(2)如果在此时间内根本没有应答（换句话说，没有开始接收数据），则当计数器达到 OC2 值时，将发生超时。

(3) 如果应答不完全，则当计数器达到 OC2 值时，也会发生超时。

2、从模式:

情况 1: 接收到的标识符由一个过滤器进行管理。该实现可以类似于主模式，因为 DFL 值是由硬件加载的。因此，在标识符字段的停止位的末端
(nominal_time_out=1.4×((DFL+2)×10bit 时间)，OC2 也可以加载
nominal_time_out+LINTCSR.CNT；

情况 2：接收到的标识符不由过滤器管理。在收到标识符字段后，需要通过软件更新 DFL 值，所以实现为：

-在 ID 的末尾，OC2 加载上 36（最大可能的应答空间）+LINCSR.CNT;

-在第一个数据字节的末尾，它将再次根据 DFL 重新加载;

-在重新加载之前，LINFlexD 会检查 count_val。如果计数值高于要重新加载的值，则会立即发生超时，并且不会发生重新加载;

注：nominal_time_out 值（1.4×（(DFL+2)×10bit 时间))分别与 LINTOCR.RTO 和 LINCR1.CFD 的复位值有关，，它随着 LINTOCR.RTO 和 LINCR1.CFD 值的变化而变化。这里，1.4 对应于(LINTOCR.RTO/10)，2 对应于(2-LINCR1.CFD)。

帧头超时机制：

主模式-由于帧头是由 LINFlexD 生成的，所以只有两种情况：
-总线上未发生错误，且定时正确（标称的帧头长度）；
-总线上发生错误，LINFlexD 在 LINESR 中标记（通常是位错误）；
因此，在主模式中没有帧头超时的意义，因此它被禁用了。

从模式：
-header_nominal=13+2+10+10=35Tbit
-header_max=1.4* header_nominal =49Tbit
-考虑到可能的 14%的时钟偏差，LINFlexD 的 header_max 为 49*1.14=55.86Tbit
-如果计数器在 11Tbit（间隔字符期间）之后开始，则 LINTOCR.HTO 值为 55.86-11=44.86Tbit。
LINTOCR.HTO 只能在从模式下进行编写，并且其复位值为 44。计数器在间隔持续时间和 LINOCR.OC1 加载了 LINOCR 的值后重启。

注意：超时计数器可用于检测其他超时情况。在这种情况下，LINTCSR.MODE 必须复位，输出比较值可以通过软件在 LINOCR 中进行更新。

陷入（stuck at）零超时错误：

如果一个主脉冲持续至少 100 位的时间，LINESR.SZF 置位。如果相同的主脉冲继续，后续的 SZF 设置间隔 87 位时间（而不是 100 位时间）。

噪声采样：

在接收期间，每个比特被采样 16 次，并且通过取第 8、第 9 和第 10 个采样的多数值（the majority value）来获得该比特的值。如果这三个样本中的任何一个的值与其他两个不同，则标记一个噪声错误。当 UARTCR.OSR=8，大部分第 2、3 和 4 次采样被考虑来确定噪声。如果 OSR 值小于 8，噪声检查将禁用。

标识符筛选:

在 LIN 协议中，消息的标识符不与节点的地址相关联，它与消息的内容相关联。发送方将消息发送给所有接收方。基于接收到的帧头，接收方决定是否接收或发送应答（取决于标识符值）。如果该消息不针对该节点，则应丢弃该消息。
为了满足这一要求，LINFlexD 提供了可配置的过滤器，以消除软件干预。硬件滤波节省了通过软件执行滤波所需的 CPU 资源。
在 LINFlexD 中最多有 16 个过滤器(取决于配置参数 no_of_filters)。过滤器只能在初始化

模式下进行编程。要激活过滤器，必须在 IFER 中设置相应的 FACT 位。根据对应的 IFMR.IFM 位，每个标识符可能有两种模式。

标识符列表模式：

如果是 IFMR.IFM[n]=0，过滤器 2n 和 2n+1 处于标识符列表模式。基于 IFER 值，用于发送/接收的最多过滤器数的数量是 no_of_filters。接收到的标识符必须与 IFCR2n 或 IFCR2n+1 的 ID 字段逐匹配(IFER. FACT 位中启用相应的过滤器)。

标识符掩码模式:

如果所需的过滤器数量大于 no_of_filters 值，则过滤器配置为掩码模式。如果是 IFMR.IFM[n]=1，过滤器 2n 和 2n+1 处于掩模模式。过滤器 2n 是过滤器，过滤器 2n+1 作为它的掩码。在这种情况下，对于过滤器 2n+1，IFER.FACT 位不起作用。如果掩码过滤器的 bitx 设为 1，则接收标识符的 bitx 必须与过滤器的 bitx 相匹配。

在任何一种过滤器模式下，如果标识符与过滤器编号 m 匹配，则 m+1 值将通过硬件加载到 IFMI 寄存器中。IFMI=0 表示一个不匹配的条件。

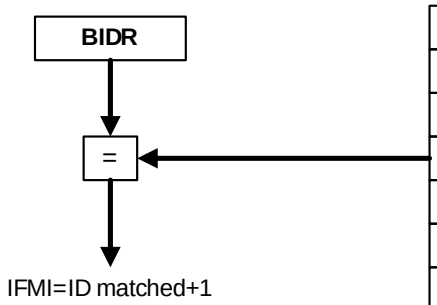
在匹配 ID 的过程中，LINFlexD 硬件将 DFL[2：0]、CCS 和 DIR 值从匹配过滤器同步到 BIDR 寄存器；自此，BIDR 寄存器只读，直到帧结束。BIDR.DIR 位决定传输方向：

1 BIDR.DIR=1，发送数据。如果 LINIER.HRIE 置位，则会生成 Tx 中断信号(ipi_int_tx)。HRIE 已设置。软件可以从 IFMI 寄存器读取过滤器匹配索引，然后将相关数据从 RAM 区域传输到 BDRM/BDRL 寄存器。在将传输数据传输到 BDRM/BDRL 寄存器后，可以设置 LINCR2.DTRQ 位启动发送。

2 BIDR.DIR=0，接收数据。如果是 LINIER.DRIE 置位，当接收到校验和并且没有校验和错误时，将生成 Rx 中断信号(ipi_int_rx)。

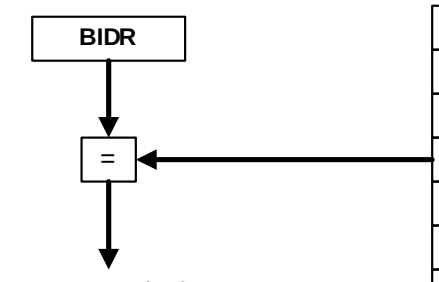
在没有过滤器匹配的情况下(IFMI=0)，如果是 LINCR1.BF=1，生成 Rx 中断信号(ipi_int_rx)。软件负责配置 BIDR 并启动传输(通过加载 BDRM/BDRL 寄存器和设置 LINCR2.DTRQ)或丢弃接收(通过设置 LINCR2.DDRQ)。

注意：如果一个标识符与两个过滤器匹配（一个在列表模式，另一个在掩码模式），则列表模式优于掩模模式。在掩码模式下，如果两个过滤器与标识符匹配，则以数字较低的过滤器为准。



IFCR0	ID5	ID4	ID3	ID2	ID1	ID0
IFCR1	ID5	ID4	ID3	ID2	ID1	ID0
IFCR2	ID5	ID4	ID3	ID2	ID1	ID0
IFCR3	ID5	ID4	ID3	ID2	ID1	ID0
IFCR4	ID5	ID4	ID3	ID2	ID1	ID0
IFCR5	ID5	ID4	ID3	ID2	ID1	ID0
IFCR6	ID5	ID4	ID3	ID2	ID1	ID0
IFCR7	ID5	ID4	ID3	ID2	ID1	ID0
...	ID5	ID4	ID3	ID2	ID1	ID0
IFCRn	ID5	ID4	ID3	ID2	ID1	ID0

图 28-12 标识符列表模式



IFCR0	ID5	ID4	ID3	ID2	ID1	ID0
IFCR1	0	0	0	1	1	1
IFCR2	ID5	ID4	ID3	ID2	ID1	ID0
IFCR3	1	1	1	1	0	0
IFCR4	ID5	ID4	ID3	ID2	ID1	ID0
IFCR5	1	1	1	1	0	1
IFCR6	ID5	ID4	ID3	ID2	ID1	ID0
IFCR7	1	1	1	1	1	0
...	...	...	...	...	...	...
IFCRn	...	...	...	...	...	...

1=Must match bits
0=Don’t care bits

$IFCR_{2n+1}$=Mask for $IFCR_{2n}$

图 28-13 标识符掩码模式

28.3.2.6 接收机中的启动检测和同步间隔符检测

接收机中有一个 10 位移位采样寄存器，它将信号数据传输给每个输入采样的下一个最低位。

一旦采样寄存器中的 1110 合格，并且预定义的验证样本中至少有 2/3 为零，就会检测到一个开始。同样，对于间隔符检测，限定样本为 0001，三个验证样本中至少有两个是 1。如果开始时的验证只有两个有效样本，则噪声标志会被置 1。

9	8	7	6	5	4	3	2	1	0	**Sample register**
—	—	—	—	—	—	—	—	—	RT1	counter=0
—	—	—	—	—	—	—	—	RT1	RT2	counter=1
—	—	—	—	—	—	—	RT1	RT2	RT3	counter=2
—	—	—	—	—	—	RT1	RT2	RT3	RT4	counter=3
—	—	—	—	—	RT1	RT2	RT3	RT4	RT5	counter=4
—	—	—	—	RT1	RT2	RT3	RT4	RT5	RT6	counter=5
—	—	—	RT1	RT2	RT3	RT4	RT5	RT6	RT7	counter=6
—	—	RT1	RT2	RT3	RT4	RT5	RT6	RT7	RT8	counter=7
—	RT1	RT2	RT3	RT4	RT5	RT6	RT7	RT8	RT9	counter=8
RT1	RT2	RT3	RT4	RT5	RT6	RT7	RT8	RT9	RT10	counter=9
RT2	RT3	RT4	RT5	RT6	RT7	RT8	RT9	RT10	RT11	counter=10
RT3	RT4	RT5	RT6	RT7	RT8	RT9	RT10	RT11	RT12	counter=11
RT4	RT5	RT6	RT7	RT8	RT9	RT10	RT11	RT12	RT13	counter=12
RT5	RT6	RT7	RT8	RT9	RT10	RT11	RT12	RT13	RT14	counter=13
RT6	RT7	RT8	RT9	RT10	RT11	RT12	RT13	RT14	RT15	counter=14
RT7	RT8	RT9	RT10	RT11	RT12	RT13	RT14	RT15	RT16	counter=15
1	1	1	0	x	0	x	0	x	0	start detect
1	1	1	0	x	0	x	0	x	1	
1	1	1	0	x	0	x	1	x	0	
1	1	1	0	x	1	x	0	x	0	
0	0	0	1	x	1	x	1	x	1	delimiter detect
0	0	0	1	x	1	x	1	x	0	
0	0	0	1	x	1	x	0	x	1	
0	0	0	1	x	0	x	1	x	1	

启动检测机制：

检测起始位的过程为：

（1）当 count=6 时，采样寄存器（移位寄存器）的状态见表 30。如果 RT1（第一个传入的采样）=0 且前三个已经收到的采样都是 1，那么它可能是一个起始位。

（2）为了确保它是一个起始位，在采样寄存器 4、2、0 中传入数据样本，对应 RT3、RT5 和 RT7，使用以下步骤进行验证。

（3）如果这 3 个样本中的大多数（2 个或更多）等于“0”，则检测到一个起始位。这三个样本 RT3、RT5 和 RT7 被称为“验证样本”，在 count=6 时被检查。

（4）样本寄存器位 9、8、7 和 6 被称为“合格样本”，在 count=6 时进行检查。

（5）在 count=9 时，如果 RT8、RT9 和 RT10 的多数值不等于“0”，则设置噪声标志。

同步间隔段间隔符检测机制：

检测同步间隔段间隔符的过程为：

（1）当 count=6 时，样本寄存器（移位寄存器）的状态见表 30。如果 RT1（第一个传入的样本）=1 和之前已经收到的三个样本都是零，那么它可能是一个间隔符位。

（2）为了确保它是一个同步间隔段间隔符，在采样寄存器 4、2、0 中传入数据样本，它们对应于 RT3、RT5 和 RT7，并使用以下步骤进行验证。

（3）如果这 3 个样本中的大多数（2 个或更多）等于“1”，则检测到一个同步间隔段间隔符位。这三个样本 RT3、RT5 和 RT7 被称为“验证样本”，在 count=6 时进行检查。

（4）样本寄存器位 9、8、7 和 6 被称为“合格样本”，在 count=6 时进行检查。

因此，一旦采样寄存器中的 1110 合格，并且在三个预定义的验证样本中至少有两个为零，就会检测到一个开始。同样，对于间隔符检测，限定样本为 0001，三个验证样本中至少有两个是 1。如果开始时的验证只有两个有效样本，则噪声标志会被置 1。

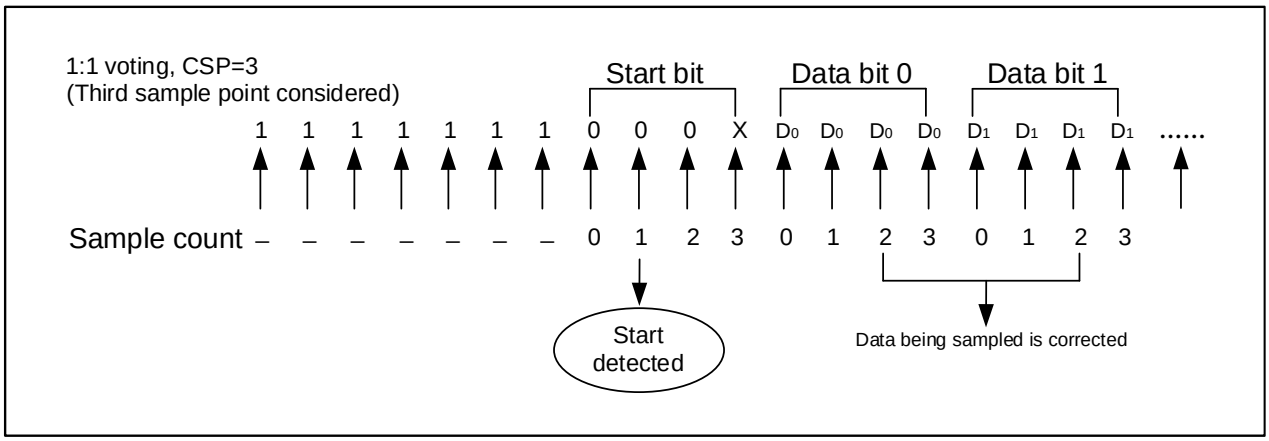


图 28-14 开始检测和取样过量采样率为 4(case 1)

注：

1)D0，D1：数据库

2)X：由于比特的转换期间的采样，值具体被采样为“0”或“1”，取决于升降时间。因此，“X”指的是一个不确定的值。

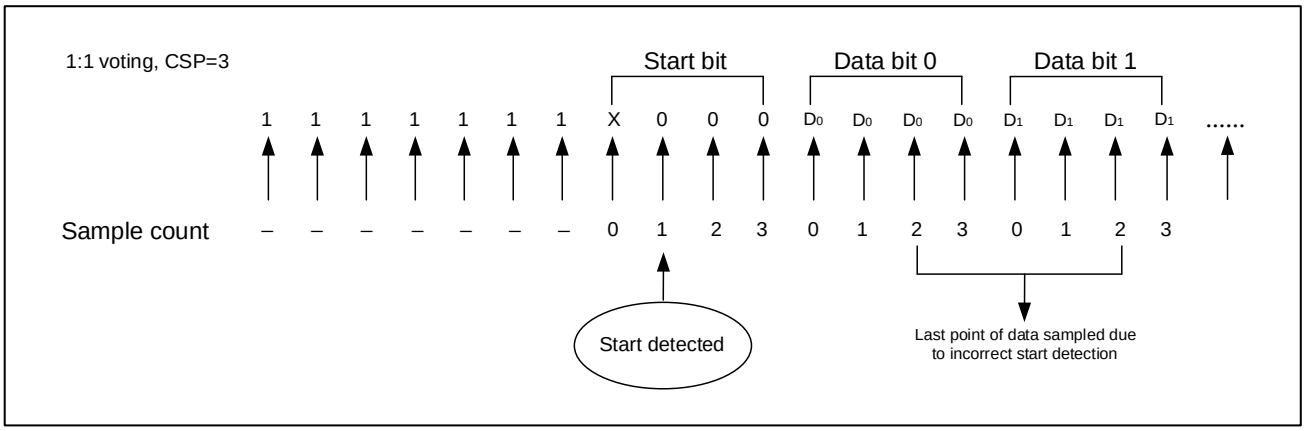


图 28-15 开始检测和取样过量采样率为 4(case 2A)

正确采样点的选择取决于外部 Rx 信号的质量。在应用开发过程中，由奇偶校验错误所指示的错误消息可以帮助找到特定应用硬件信号的最佳设置。因此，要在正确的点取样，CSP 必须改为两个；导致以下情况：

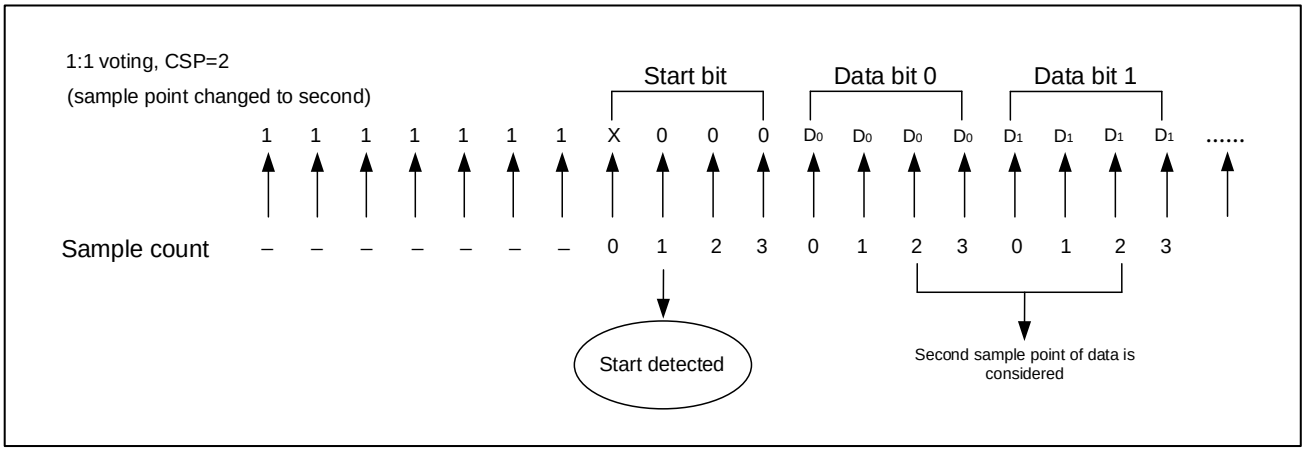


图 28-16 开始检测和取样过量采样率为 4(case 2B)

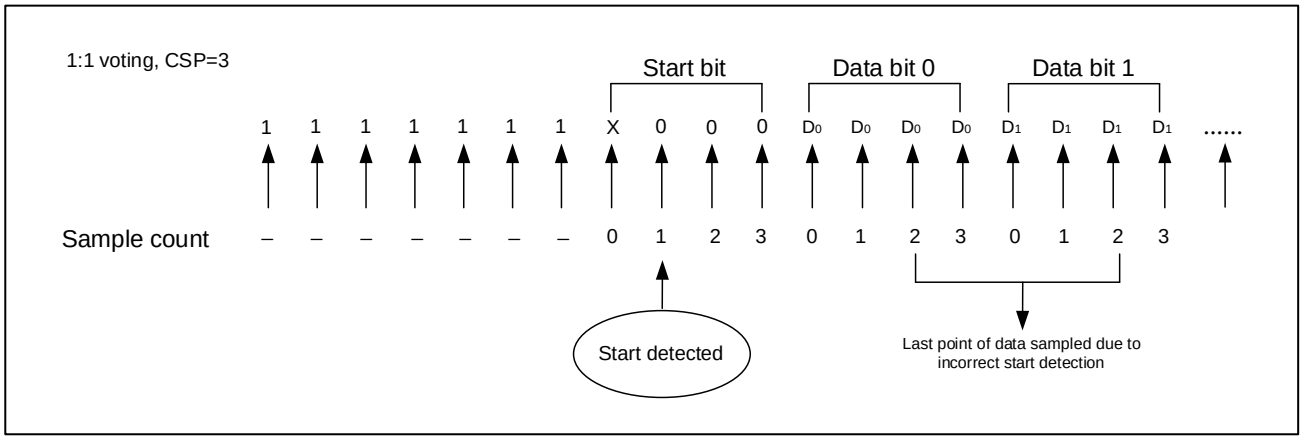


图 28-17 开始检测和取样过量采样率为 4(case 3)

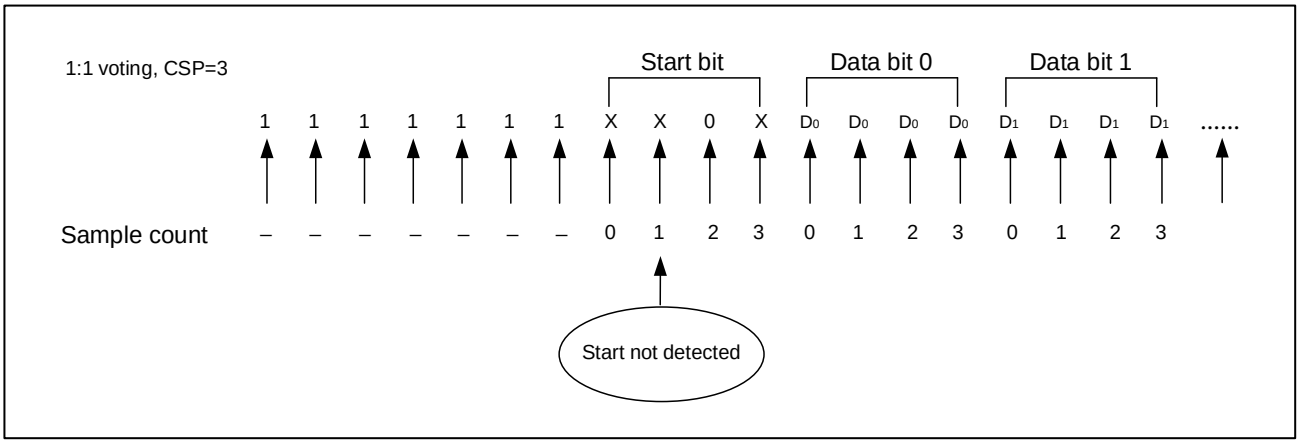


图 28-18 开始检测和取样过量采样率为 4(case 4)

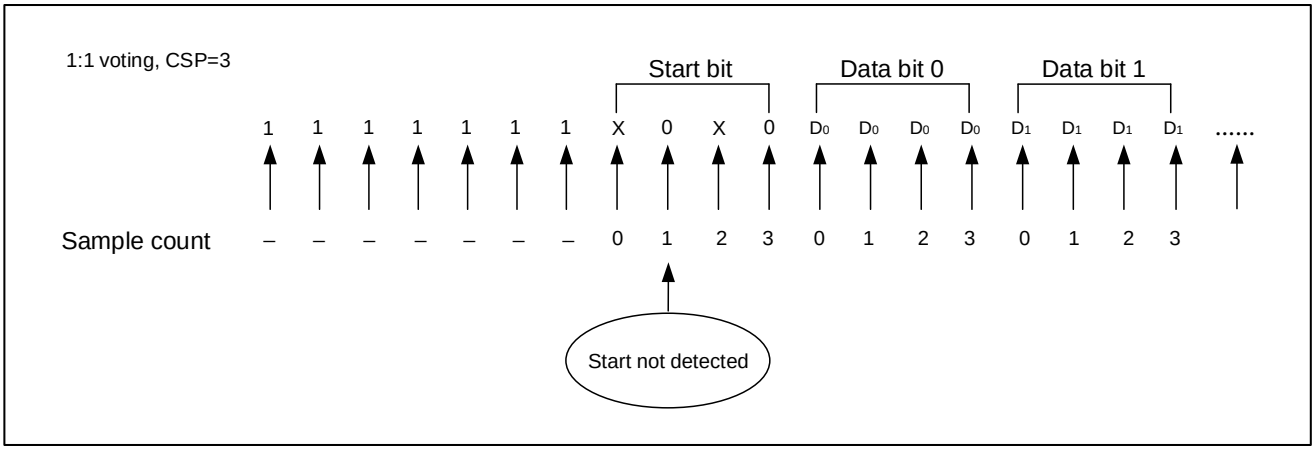


图 28-19 开始检测和取样过量采样率为 4(case 5)

28.3.2.7 波特率生成

LIN 波特率可在两个寄存器中编程：LIN 整数波特率寄存器(LINIBRR)和 LIN 小数波特率寄存器(LINFBRR)。这些寄存器只能在初始化模式下进行编程。

接收机和发送机的波特率都用以下公式计算：

（1）当 UARTCR.ROSE=1，Tx=Rx=ipg_baud_clk / (OSR×LDIV)

（2）当 UARTCR.ROSE=0，Tx=Rx=ipg_baud_clk / (16×LDIV)

其中，ipg_baud_clk 是波特时钟的频率。

LDIV 是一个无符号的定点数。整数被编码到 LINIBRR 的 20 位，分数被编码到 LINFBRR 的 4 位。

当启用精简的过采样时，LINFBRR 不应该被使用并编程为零，LDIV 只包含 LINIBRR 的整数部分。

例如：当 ROSE=0(对于 LIN 和 UART 模式)：LDIV=468.75d，ipg_baud_clk=36MHz，LINIBRR=468d，LINFBRR=12，波特率=36MHz/（16×468.75）=4.8 Kbit/s

例如：当 ROSE=1(仅用于 UART 模式)：LDIV=5d，ipg_baud_clk=80MHz，LINIBRR=5d，OSR=4，波特率=LIN_CLK/(OSR×LDIV)=80MHz/（4×5）=4 Mbit/s

28.3.2.8 自动重新同步

要根据 LIN 同步字段的测量值自动调整波特率，将前置分频值（标准波特率）写入 LINIBRR 和 LINFBRR 中，然后设置 LINCR1.LASE 以启用自动同步。

当启用自动同步时，在每个同步间隔段间隔符之后，在 ipg_baud_clk 下采样 5 个 ipp_ind_linrx 下降沿之间的时间长度。

28.3.2.9 唤醒管理

处于休眠状态的 LIN 网络中的任何节点都可以发送唤醒请求。唤醒请求为强制总线进入显性状态 250us 到 5ms。每个从节点都应该检测唤醒请求(一个长于 150us 的显性电平)，并准备在 100ms 内接收总线命令，100ms 从显性电平的结束边缘开始计算。在检测到唤醒请求时，主机节点也会被唤醒，当从机节点准备好时，开始发送帧头以查找被唤醒的原因。如果主节点在唤醒请求 150 ms 内没有发出帧头，那么发出请求的节点可以尝试发出新的唤醒请求。

在 LINFlexD 中，可以通过在 BDRL.DATA0 中写入唤醒字符并将 LINCR2.WURQ 置 1 来产生唤醒请求。LINCR2.WURQ 位置 1 后，BDRL.DATA0 中的字符被传送。

在 LINFlexD 中，可以通过两种方式检测唤醒：

（1）AUTOWU=1：在睡眠模式下检测到下降沿时，LINCR1.SLEEP 位被硬件清零，LINSR.WUF 位置 1，且如果 LINIER.WUIE = 1，则生成中断；此时 LINFlexD 处于正常模式，准备接收帧。

（2）AUTOWU=0：在检测到一个下降沿时 LINCSR.WUF 将置 1，并产生中断(如果 LINIER.WUIE = 1)。然后由软件来清除 LINCR1.SLEEP。

图 28-20 唤醒序列

28.3.3 UART MODE

UART 模式的主要功能包括：

- 全双工通信
- 8 位帧，9 位帧，13 位帧，16 位帧，17 位帧
- 偶/奇/0/1 校验
- 用户可编程过采样率，以获得高达 ipg_baud_clk/4 Mbit/s 的波特率

28.3.3.1 8 位数据帧

第 8 位可以是数据位或奇偶校验位。UARTCR.PC 可以选择偶/奇/0/1 奇偶校验。如果 7 个数据位的模 2 和为 1，则偶校验位置 1，在这种情况下，奇校验位为零。

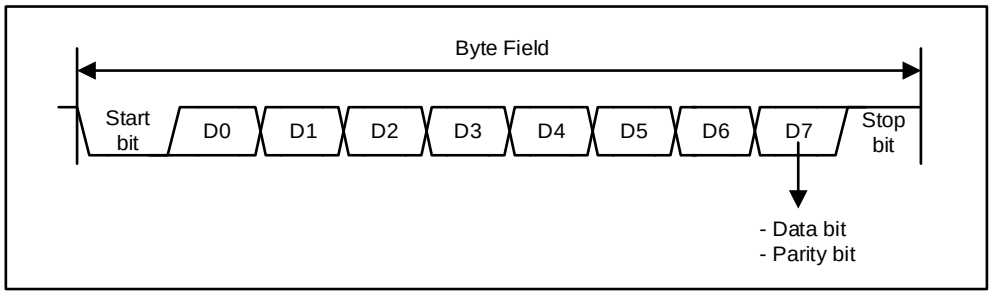


图 28-21 8 位数据帧

28.3.3.2 9 位数据帧

第 9 位是奇偶校验位。UARTCR.PC 可以选择偶/奇/0/1 奇偶校验。如果 8 个数据位的模 2 和为 1，则偶数校验位为 1。在这种情况下，奇校验位为零。奇偶校验 0 即强制逻辑值为低电平。奇偶校验 1 则强制逻辑值为高电平。

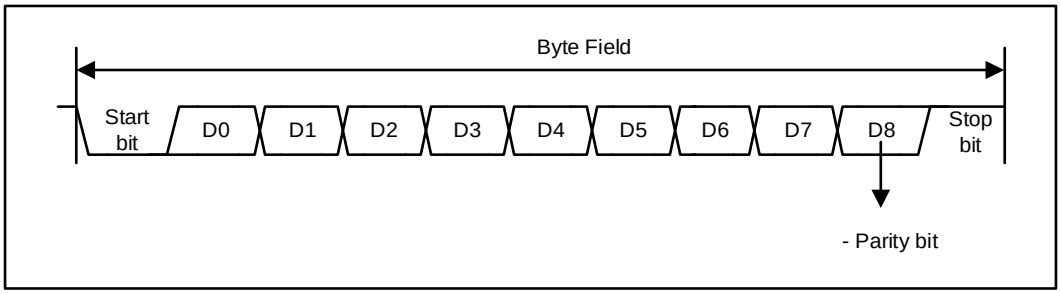


图 28-22 9 位数据帧

28.3.3.3 16 位数据帧

第 16 位可以是数据位或奇偶校验位。UARTCR.PC 可以选择偶/奇/0/1 奇偶校验。如果 15 个数据位的模 2 和为 1，则偶数校验位为 1。在这种情况下，奇校验位为零。奇偶校验 0 即强制逻辑值为低电平。奇偶校验 1 则强制逻辑值为高电平。

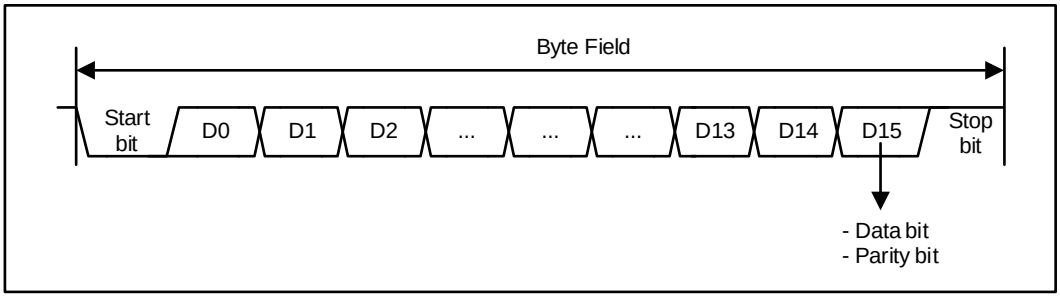


图 28-23 16 位数据帧

28.3.3.4 17 位数据帧

第 17 位为奇偶校验位。UARTCR.PC 可以选择偶/奇/0/1 奇偶校验。如果 16 个数据位的模 2 和为 1，则偶数校验位为 1。在这种情况下，奇校验位为零。奇偶校验 0 即强制逻辑值为低电平。奇偶校验 1 则强制逻辑值为高电平。

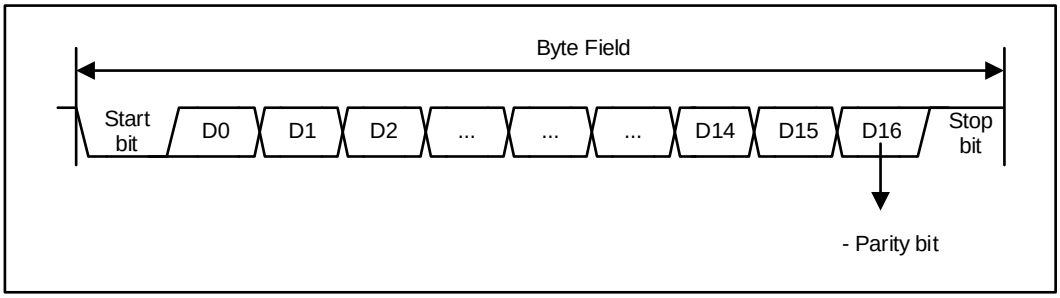


图 28-24 17 位数据帧

28.3.3.5 13 位数据帧

当 UARTCR.WLS=1，在 UART 模式下选择特殊字长。特殊字长仅允许在 FIFO 模式下接收 12 位数据+奇偶校验位。

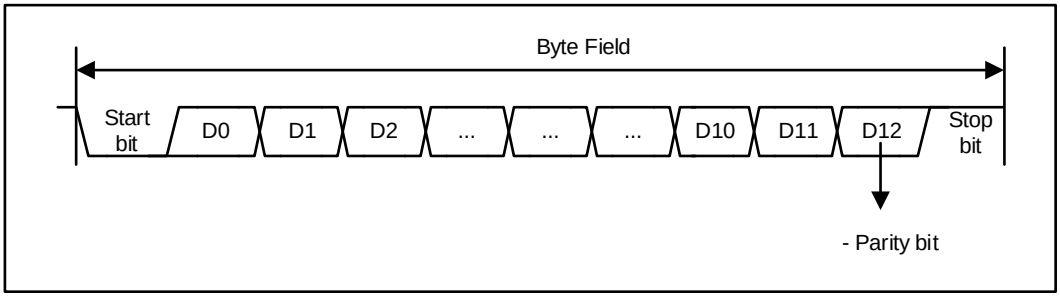


图 28-25 13 位数据帧

28.3.3.6 UART 模式下的缓冲区

8 字节缓冲区分为两部分：一部分用于接收机，一部分用于发送机，如下表所示。

表 28-2

Tx0	BDT0(BDRL.DATA0)
Tx1	BDT1(BDRL.DATA1)
Tx2	BDT2(BDRL.DATA2)
Tx3	BDT3(BDRL.DATA3)
Rx0	BDT4(BDRM.DATA4)
Rx1	BDT5(BDRM.DATA5)
Rx2	BDT6(BDRM.DATA6)
Rx3	BDT7(BDRM.DATA7)

UART 发送机

在 UART 模式下，设置 UARTCR.UART 和 UARTCR.TXEN 分别为 1 开启发送,发送在 BDR0（最低有效数据字节）编程时开始，并持续到发送的字节数/半字数等于 UARTCR.TDFL 位。

发送缓冲区为四个字节（当 UARTCR.WL1=0 时）或两个半字（当 UARTCR.WL1=1 时）。因此，最多可以发送四个字节（两个半字）。一旦发送完已编程的字节数或半字数， UARTSR.DFT 置 1。如果 UARTCR.TxEN 位在发送中途被复位，当前发送完成后不再调用其他发送。

在 FIFO 模式下，可通过设置控制位 UARTCR.TFBM 配置缓冲区

注意：如果是 UARTSR.TFF＝1，并且对 FIFO 执行写操作。传输的数据可能是错误的。

IPS 操作	寄存器	模式（UARTCR.TFBM）	数据长度（UARTCR.WL1）	外部总线操作结果
写 byte0	BDRL	FIFO	byte	ok
写 byte 1-2-3	BDRL	FIFO	byte	error
写 halfword 0-1	BDRL	FIFO	byte	error
写 word	BDRL	FIFO	byte	error
写 byte 0-1-2-3	BDRL	FIFO	half-word	error
写 halfword0	BDRL	FIFO	half-word	ok
写 halfword1	BDRL	FIFO	half-word	error
写 word	BDRL	FIFO	half-word	error
读 byte 0-1-2-3	BDRL	FIFO	byte/half-word	error
读 halfword0-1	BDRL	FIFO	byte/half-word	error
读 word	BDRL	FIFO	byte/half-word	error
写 byte 0-1-2-3	BDRL	BUFFER	byte/half-word	ok
写 halfword0-1	BDRL	BUFFER	byte/half-word	ok

写 word	BDRL	BUFFER	byte/half-word	ok
读 byte 0-1-2-3	BDRL	BUFFER	byte/half-word	ok
读 halfword0-1	BDRL	BUFFER	byte/half-word	ok
读 word	BDRL	BUFFER	byte/half-word	ok

在 UART FIFO 模式中（UARTCR.TFBM = 1），任何读操作都会造成外部总线传输错误（ipx_xft_err）

28.3.3.7 UART 接收机

一旦软件退出初始化模式（通过软件清除 LINCR1.INIT），置 UARTCR.RxEn 为 1,并且监测开始位。有一个专用的 4 字节（如果 UARTCR.WL1=0）或 2 半字（如果 UARCR.WL1=1）数据缓冲区用于接收数据。一旦接收到已编程的字节数（UARTCR.RDFL），UARTSR.DRF 标志位置 1，并且完成当前接收。只需设置 UARTCR.RxEn 为 1 即可开始接收，一旦接收到编程的字节数，接收就自动完成。在 FIFO 模式下，可通过设置控制位 UARTCR.RFBM 配置缓冲区

IPS 操作	寄存器	模式 (UARTCR.TFBM)	数据长度 (UARTCR.WL1)	外部总线操作结果
读 byte 4	BDRM	FIFO	byte	ok
读 byte 5-6-7	BDRM	FIFO	byte	error
读半字 2-3	BDRM	FIFO	byte	error
读字	BDRM	FIFO	byte	error
读字节 4-5-6-7	BDRM	FIFO	half-word	error
读半字 2	BDRM	FIFO	half-word	ok
读半字 3	BDRM	FIFO	half-word	error
读字	BDRM	FIFO	half-word	error
写字节 4-5-6-7	BDRM	FIFO	byte/half-word	error
写半字 2-3	BDRM	FIFO	byte/half-word	error
写字	BDRM	FIFO	byte/half-word	error
读字节 4-5-6-7	BDRM	BUFFER	byte/half-word	ok
读半字 2-3	BDRM	BUFFER	byte/half-word	ok
读字	BDRM	BUFFER	byte/half-word	ok
写字节 4-5-6-7	BDRM	BUFFER	byte/half-word	error
写半字 2-3	BDRM	BUFFER	byte/half-word	error

写字	BDRM	BUFFER	byte/half-word	error

注：参考寄存器 BDRL 和 BDRM 的布局，以确定 bytex 与寄存器 BDRL 与 BDRM 的数据位之间的映射。

补充说明：

- 如果事先不知道要接收多少字节。RDFL 不应预先编程。RDFLI 的复位值为零，这确保接收逐字节进行。在每个接收到的字节之后，状态机进入 Idle 状态。
- 如果 RDFL 被编程为某个值，但未接收到该字节数，则接收挂起。当发生接收超时 UARTSR.TO 被置 1。
- 如果 ipg_stop 在接收过程中被有效，则只有在接收到所有编程的数据字节数后才被响应。如果未接收到编程的数据字节数且出现接收超时，则当接收机 FSM 被重置为 Idle 状态时，ipg_stop 请求将得到响应。
- 如果在接收任意字节期间发生奇偶校验错误，则相应的 UARTSR.PE[n]位置 1。在这种情况下不产生中断。如果一个字节中发生帧错误（设置了 UARTSR.FEF），则当 LINIER.FEIE=1 时会生成一个中断。由于帧错误只有一个寄存器位，此中断有助于识别哪个字节有帧错误。
- 如果尚未从缓冲区读取上一个接收到的帧（换句话说，RMB 没有被重置），则在接收到下一个字节时，会发生溢出错误（设置了 UARTSR.BOF 为 1）。如果 LINIER.BOIE=1，则产生中断。新消息被丢弃或覆盖缓冲区中的前一消息。
- 当 LINFlexD 处于 UART FIFO 模式时系统发起 ipg_stop 请求，则软件必须设置 NCR.INIT 使接收机 FSM 进入 Idle 状态。LINFlexD 返回 ipg_stop_ack 有效。

28.3.4 中断

表 33 总结了 LINFlexD 中断事件和相应的中断启用。图 50 显示了对 LINFlexD 中断输出信号的映射中断标志。

表 28-3 中断表

中断事件	事件标志位 LINSR/LINESR/UARTSR	中断使能位 LINIER	中断向量
Stuck at zero	SZF	SZIE	error
输出比较	OCF	OCIE	error
位错误	BEF	BEIE	error
校验和错误	CEF	CEIE	error
Header error	SFEF or SDEF or IDPEF	HEIE	error
Frame error	FEF	FEIE	error
Buffer Overrun error	BOF	BOIE	error
UART 超时错误	TO	TOIE	error
LIN state	Sync Del,Sync Field,Identifier field,or Checksum Field	LSIE	Rx
唤醒	WUF	WUIE	Rx
数据接收完毕	DRF	DRIE	Rx
数据传输	DTF	DTIE	Tx
Header received	HRF	HRIE	Rx
Header received	HRF	HEIR	Tx(如果存在与 Tx 标识符的筛选

			器匹配)

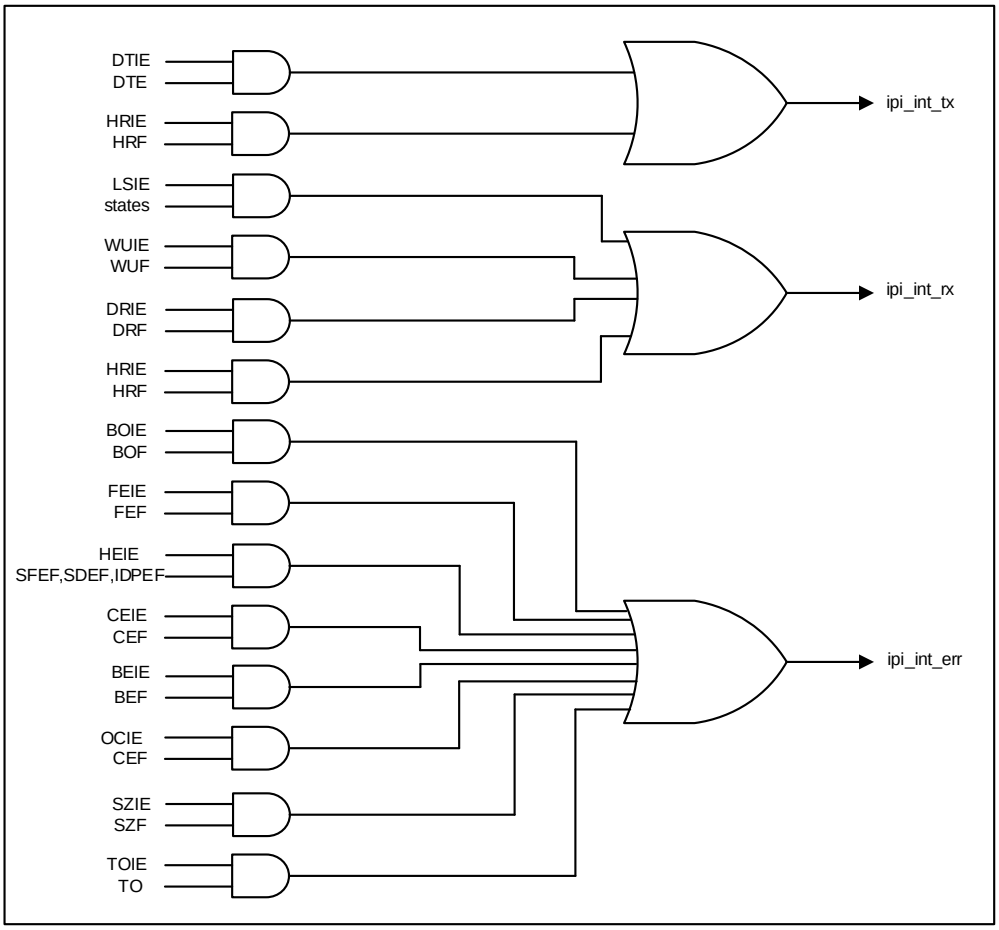


图 28-26 中断图

注意：由于 LIN 状态事件而导致的 Rx 中断将通过写入 LINSR 来清除。当启用 LSIE 时，使用 0xF 的 LINS 字段。注意：在回路模式下，Rx 中断的设置时间早于 Tx 中断。

28.3.5 LINFlexD 时钟偏差

28.3.5.1 更快的接收机偏差

如果接收机的波特率高于发送机，则结束位采样可能在最后一个发射的有效载荷位期间开始。为了确保正确的噪声和无帧误差的接收，采样点 RS8、RS9 和 RS10 必须位于传输的停止位中，如图 28-25 所示。

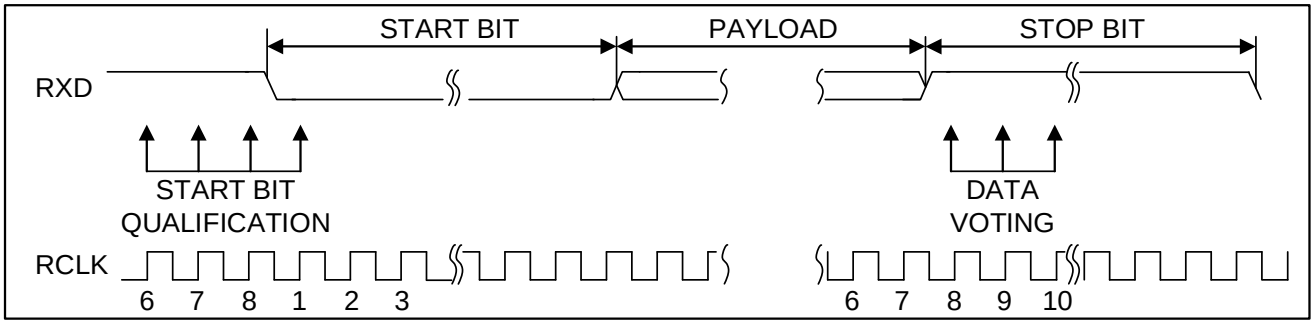


图 28-27 更快的接收机

28.3.5.2 较慢的接收机偏差

如果接收机的波特率比发送机慢，那么在传输下一个启动位时，停止位采样可能仍在运行。为了确保正确的噪声和无帧误差的接收，采样点 RS8、RS9 和 RS10 必须位于传输的停止位中，如图 28-26 所示。

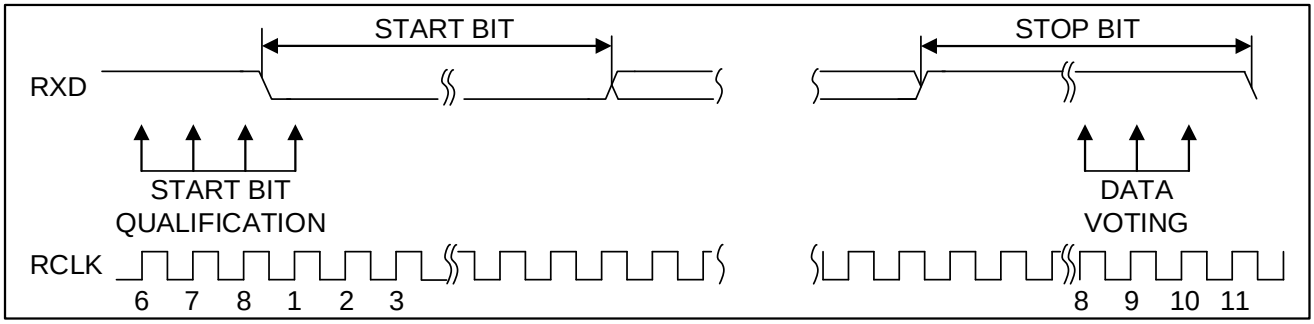


图 28-28 较慢的接收机

28.4 寄存器列表

LIN0 基地址：0x40010000
LIN1 基地址：0x40010400

表 28-4 LINFlexD 控制器寄存器列表

偏移地址	名称	描述	复位值
0x000	LINCR1	LIN 控制寄存器 1	0x00000082
0x004	LINIER	LIN 中断使能寄存器	0x00000000
0x008	LINSR	LIN 状态寄存器	0x00000040
0x00C	LINESR	LIN 错误状态寄存器	0x00000000
0x010	UARTCR	UART 模式控制寄存器	0x00000000
0x014	UARTSR	UART 模式状态寄存器	0x00000000
0x018	LINTCSR	LIN 超时特性控制状态寄存器	0x00000200
0x01C	LINOCR	LIN 输出比较寄存器	0x0000FFFF
0x020	LINTOCR	LIN 超时控制寄存器	0x00000E2C
0x024	LINFBRR	LIN 分数波特率寄存器	0x00000000
0x028	LINIBRR	LIN 整数波特率寄存器	0x00000000
0x02C	LINCFR	LIN 校验和场寄存器	0x00000000
0x030	LINCR2	LIN 控制寄存器 2	0x00006000
0x034	BIDR	缓冲区标识寄存器	0x00000000
0x038	BDRL	缓冲区有效低位数据寄存器	0x00000000
0x03C	BDRM	缓冲区有效高位数据寄存器	0x00000000
0x040	IFER	标识过滤器使能寄存器	0x00000000
0x044	IFMI	标识符过滤器匹配索引	0x00000000
0x048	IFMR	标识符过滤器模式寄存器	0x00000000
0x04C	IFCR0	识别滤波控制寄存器 0	0x00000000
0x050	IFCR1	识别滤波控制寄存器 1	0x00000000
0x054	IFCR2	识别滤波控制寄存器 2	0x00000000
0x058	IFCR3	识别滤波控制寄存器 3	0x00000000
0x05C	IFCR4	识别滤波控制寄存器 4	0x00000000
0x060	IFCR5	识别滤波控制寄存器 5	0x00000000
0x064	IFCR6	识别滤波控制寄存器 6	0x00000000
0x068	IFCR7	识别滤波控制寄存器 7	0x00000000
0x06C	IFCR8	识别滤波控制寄存器 8	0x00000000
0x070	IFCR9	识别滤波控制寄存器 9	0x00000000
0x074	IFCR10	识别滤波控制寄存器 10	0x00000000
0x078	IFCR11	识别滤波控制寄存器 11	0x00000000

0x07C	IFCR12	识别滤波控制寄存器 12	0x00000000
0x080	IFCR13	识别滤波控制寄存器 13	0x00000000
0x084	IFCR14	识别滤波控制寄存器 14	0x00000000
0x088	IFCR15	识别滤波控制寄存器 15	0x00000000
0x08C	GCR	全局控制寄存器	0x00000000
0x090	UARTPTO	UART 预设超时寄存器	0x00000FFF
0x094	UARTCTO	UART 当前超时寄存器	0x00000000

28.5 LINFlexD 寄存器说明

28.5.1 控制寄存器 1(LINCR1)

偏移地址：0x00
复位值：0x00000082

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															NLSE
															R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCD	CFD	LASE	AUTOWU	MBL				BF	保留	LBKM	MME	SSBL	RBLM	SLEEP	INIT
R/W	R/W	R/W	R/W	R/W				R/W		R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:17	Res	保留	0x0	-
16	NLSE	位错误时 LIN State 使能； 当发生位错误标志时（LINESR.BEF=1），NLSE 使能/禁止捕获 LIN 状态： 1：使能 0：禁止 注意：如果 NLSE 和 LINCR2.IOBE 都为 1 时，当 LIN 进入空闲状态时，LIN 内部的状态是不能被捕获的。当发生位错误时，LINSR.LINS 显示当前在哪个状态。	0x0	R/W
15	CCD	禁用校验和计算： 1：禁用校验和计算，LINCFR 寄存器是可读/可写的。如果启用了校验和场(CFD=0)，LINCFR 可以通过软件来发送软件计算的校验和场/CRC； 0：校验和场由硬件完成。LINCFR 寄存器是只读的； CCD 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
14	CFD	校验和段禁用： 1：在发送数据帧中没有使用校验和段； 0：在发送所指定的数据字节后，启用校验和段； CFD 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
13	LASE	LIN 自动同步使能（详见第 68 页的“自动重新同步”部分）： 1：启用自动同步； 0：禁用自动同步； LASE 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
12	AUTOWU	自动唤醒： 1：如果 LINSR.WUF=1，SLEEP 位被硬件清零； 0：SLEEP 位仅软件清零； AUTOWU 在 LIN 和 UART 模式下都有效。 AUTOWU 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
11:8	MBL	主间隔段长度—在 LIN 主模式下控制的间隔段长度： 0000b：10 位间隔段长度	0x0	R/W

		0001b：11 位间隔段长度 0010b：12 位间隔段长度 0011b：13 位间隔段长度 0100b：14 位间隔段长度 0101b：15 位间隔段长度 0110b：16 位间隔段长度 0111b：17 位间隔段长度 1000b：18 位间隔段长度 1001b：19 位间隔段长度 1010b：20 位间隔段长度 1011b：21 位间隔段长度 1100b：22 位间隔段长度 1101b：23 位间隔段长度 1110b：36 位间隔段长度 1111b：50 位间隔段长度		
7	BF	旁路过滤器-控制接收过滤器旁路功能： 1：如果没有 ID 过滤器被激活，接收输入帧；当 ID 过滤器处于激活状态时，如果接收帧不匹配任何 ID 滤波器，接收方响应； 0：如果 ID 不匹配任何 ID 过滤器或没有 ID 过滤器处于激活状态，接收机忽略传入帧； BF 可以在任何模式下读取，但只能在初始化模式下写入。	0x1	R/W
6	Res	保留	0x0	
5	LBKM	回环模式-启用或禁用回环模式（详见第 54 页的“回环模式”部分）： 1：启用回环模式； 0：禁用回环模式； LBKM 在 LIN 和 UART 模式下都有效。 LBKM 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
4	MME	主模式使能： 1：主模式； 0：从模式； MME 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
3	SSBL	从模式同步间隔段长度-设置从模式下间隔段检测的比特位数： 1：10 位间隔段长度； 0：11 位间隔段长度； SSBL 可以在任何模式下读取，但只能在初始化模式下写入	0x0	R/W
2	RBLM	接收机缓冲区锁定模式： 1：接收机缓冲区锁定防止溢出,一旦缓冲区已满，如果没有通过软件清零 LINSR.RMB 位释放缓冲区，则下一个传入消息将被丢弃; 0：接收缓冲区未被锁定,下一个传入的消息将覆盖前一个消息。 RBLM 在 LIN 模式和 UART 模式下都有效。 RBLM 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
1	SLEEP	睡眠模式请求-软件置位，请求 LINFlexD 进入睡眠模式。	0x1	R/W

		软件请零 SLEEP 位以退出休眠模式。 如果 LINCR1.AUTOWU 和 LINSR.WUF 置 1，硬件自动清零 SLEEP 位，退出休眠模式。 SLEEP 位在 LIN 和 UART 模式下都有效		
0	INIT	初始化模式请求-通知软件设置 LINFlexD 进入初始化模式。 软件清零 INIT 位(此时 SLEEP=0)，LINFlexD 进入正常模式。 INIT 在 LIN 和 UART 模式下都有效	0x0	R/W

28.5.2 LIN 使能中断寄存器(LINIER)

偏移地址：0x004
复位值：0x0000_0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIZE	OCIE	BEIE	CEIE	HEIE	保留		FEIE	BOIE	保留	WUIE	保留	TOIE	DRIE	DTIE	HRIE
R/W	R/W	R/W	R/W	R/W			R/W	R/W		R/W		R/E	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	ReS	保留	0x0	-
15	SZIE	Stuck at Zero 中断使能/Stuck at Zero： 1：使能中断；在 LINESR 或者 UARTSR 中，Stuck at Zero FLAG（SZF）置 1 将产生中断。 0：禁止中断；	0x0	R/W
14	OCIE	输出比较中断使能/Output Compare： 1：使能中断，在 LINESR 或者 UARTSR 中，输出比较标志位（OCF）置 1 时产生中断。 0：禁止中断	0x0	R/W
13	BEIE	位错误中断使能/Bit Error： 1：使能中断，当在 LINESR 中位错误标志（BEF）置 1 时产生中断； 0：禁止中断；	0x0	R/W
12	CEIE	校验和错误中断使能/Checksum Error： 1：使能中断，当在 LINESR 中校验和标志（CEF）置 1 时产生中断； 0：禁止中断；	0x0	R/W
11	HEIE	帧头错误中断使能/Header Error： 1：使能中断，当在 LINESR 中以下任何一个标志： SFEF\SDEF\IDPEF 置 1 时产生中断; 0：禁止中断；	0x0	R/W
10:9	Res	保留	0x0	-
8	FEIE	帧错误中断使能/Frame Error Interrupt Enable： 1：使能中断，当在 LINESR 或者 UARTSR 中框架错误标志（FEF）	0x0	R/W

		置 1 时产生中断； 0：禁止中断；		
7	BOE	缓冲区溢出中断使能/Buffer Overrun： 1：使能中断。在 LINESR 或 UARTSR 中缓冲区溢出标志（BOF）置 1 时产生中断。 0：禁止中断。	0x0	R/W
6	LSIE	LIN 状态中断使能/LIN State： 1：使能中断。输入以下状态时生成中断：Sync Del、Sync Field、Identifier Field、Checksum。 0：禁止中断；	0x0	R/W
5	WUIE	唤醒中断使能： 1：使能中断。在 LINESR 或 UARTSR 中唤醒标志(WUF)为 1 时产生中断。 0：禁止中断。	0x0	R/W
4	Res	保留	0x0	-
3	TOIE	超时中断使能： 1：使能中断。当 LINFlexD 处于 UART 模式并且在 UARTSR 中设置超时位（TO）时产生中断。 0：禁止中断。	0x0	R/W
2	DRIE	数据接收完成中断使能： 1：使能中断。在 LINSR 或 UARTSR 中设置数据接收标志(DRF)为 1 时产生中断。 0：禁止中断。	0x0	R/W
1	DTIE	数据发送中断使能： 1：使能中断。在 LINSR 或 UARTSR 中设置数据发送标志(DTF)为 1 时产生中断。 0：禁止中断。	0x0	R/W
0	HRIE	帧头接收中断使能： 1：使能中断。在 LINSR 中帧头接收标志(HRF)为 1 时，产生中断。 0：禁止中断。	0x0	R/W

28.5.3 LIN 状态寄存器(LINSR)

偏移地址：0x08
复位值：0x00000040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留												AUTOSYNC_COMP	RDC		
												RCW1	RO		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LINS				保留		RMB	DRBNE	RXBUSY	RDI	WUF	保留		DRF	DTF	HRF
RO						RCW1	RCW1	RO	RO	RCW1			RCW1	RCW1	RCW1

位	标记	功能描述	复位值	读写
31:20	Res	保留	0x0	-
19	AUTOSYNC_COMP	自动同步完成 AUTOSYNC_COMP 在自动同步被启用(LINCR1.LASE=1)且自动同步完成时置 1。AUTOSYNC_COMP 被置 1 后，可以读取 LINIBRR 和 LINFBRR 寄存器的内容。	0x0	RCW1
18:16	RDC	接收数据字节数 RDC 包含 LIN 模式下接收数据缓冲区的字节数。 000：1 个字节 001：2 个字节 010：3 个字节 011：4 个字节 100：5 个字节 101：6 个字节 110：7 个字节 111：8 个个字节 RDC 是一个只读字段，可用于调试目的。	0x0	RO
15:12	LINS	LIN 状态： 0000：Sleep mode：LINFlexD 处于睡眠模式以降低功耗。 0001：Init mode：LINFlexD 处于初始化模式。 0010：Idle：在以下情况，LINFlexD 会进入空闲状态： 1) LINCR1.SLEEP 和 LINCR1.INIT 由软件清除； 2) ipp_ind_linrx 收到唤醒脉冲，并且 LINCR1_AUTOWU=1； 3) 前一帧的数据的发送或接收已经完成或中止； 0011：Break： 从模式下，检测到下降沿，后面跟一个显性状态，LINFlexD 正在接收一个同步间隔段； 主模式下，正在发送 break field。 注意：在从模式下，一旦出现错误，LIN 状态的是 Idle 还是 Break，取决于 ipp_ind_linrx 检测到的最后一个比特位。如果最后检测到的位是显性位，则新状态是 Break；否则新状态是 Idle。	0x0	RO

		0100：Break Delimiter： 从模式，检测到一个有效的 break field (具体取决于 LINCR1.SSBL 的 10 或 11 位)，并且 LINFlexD 在等待一个上升沿； 主模式，break field 传输已完成，正在发送 break delimiter。 0101：Sync Field： 从模式，已检测到一个有效的 break delimiter（至少一比特时间的隐性状态），并且 LINFlexD 正在接收一个 Sync byte Field； 主模式，正在进行 Sync Field 传输。 0110：Identifler Field： 从模式，已接收到一个有效的同步字段，并且 LINFlexD 正在接收一个标识符； 主模式，正在进行标识符字段传输。 0111：Header Reception/Transmission： 从模式，已接收到一个有效的 header，并且该标识符字段在 BIDR 中可用。 主模式，header 发送完成。 1000：Data Reception/Transmission： 接收模式，正在进行接收。 发送模式，应答发送正在进行。 1001：Checksum：数据发送/接收已完成。校验和发送/接收正在进行。 在 UART 模式下，空闲、初始、睡眠和数据发送/接收状态由 LIN 状态位来反映。 注意：LINS 位域的值不会因为对它的任何写操作而改变。只有在对 LINS 写 0xF 会清除仅有 LIN 状态事件产生的 Rx 中断。		
11:10	Res	保留	0x0	-
9	RMB	释放消息缓冲区： 1：缓冲区数据已准备好由软件读取。读取完缓冲区中收到的数据后，RMB 应被软件清零； 0：缓冲区是空的。RMB 在初始化模式下由硬件清零。	0x0	RC W1
8	DRBNE	数据接收缓冲区非空标志 一旦接收到应答的第一个字节并存储在 BDRL 中（当接收缓冲区中至少有一个数据字节时），硬件就会把 DRBNE 置 1。软件应在读取所有缓冲区后清除 DRBNE； 在发生应答超时事件时，可以通过软件检查 DRBNE；	0x0	RC W1

		在初始化模式下 DRBNE 会被硬件清除。		
7	RXBUSY	接收机繁忙标志： 1：接收正在进行中； 0：接收机处于空闲。 注意：在从模式下，在帧头接收后，如果是 BIDR.DIR=0，则接收开始，RXBUSY 位被置 1。在这种情况下，软件无法设置 LINCR2.DTRQ 位。	0x0	RO
6	RDI	接收机数据输入：指示 ipp_ind_linrx 引脚的当前状态。 注意：复位释放后，RDI 会反映 ipp_ind_linrx 引脚的实际值。	0x1	RO
5	WUF	唤醒标志： 在以下情况 ipp_ind_linrx 引脚上检测到下降沿时，WUF 会被硬件设置 1： （1）LINFlexD 处于从模式且处于休眠模式； （2）LINFlexD 处于主模式且处于休眠模式或空闲状态。 WUF 在初始化模式下被硬件清零。	0x0	RC W1
4:3	Res	保留	0x0	-
2	DRF	数据接收完成标志： DRF 由硬件置 1，并表示数据接收已完成； DRF 以初始化模式由硬件清除。 注意：当发生帧错误或校验和错误时，DRF 不被设置。	0x0	RC W1
1	DTF	数据发送完成标志： DTF 由硬件置 1，并表示数据发送已完成； DTF 在初始化模式下被硬件清除。 注意：在 LIN 模式中，当出现位错误时，DTF 不被置位且 LINCR2.IOBE = 0。	0x0	RC W1
0	HRF	帧头接收标志： 帧头接收完成并满足以下条件之一时 HRF 由硬件置 1： （1）所有过滤器都是无效的，且 LINCR1.BF = 1； （2）任何过滤器都不匹配，且 LINCR1.BF = 1； （3）Tx 过滤器匹配。 HRF 在初始化模式下被复位，帧结束或帧丢弃时被硬件复位。	0x0	RC W1

LINSR 寄存器包含指示 LINFLEXD 硬件状态的状态位。

28.5.4 LIN 错误状态寄存器(LINESR)

地址偏移：0x0C

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SZF	OCF	BEF	CEF	SFEF	SDEF	IDPEF	FEF	BOF	保留						NF
RCW1	RCW1	RCW1	RCW1	RCW1	RCW1	RCW1	RCW1	RCW1							RCW1

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	-
15	SZF	零点卡顿标志位，当发生零点卡顿的超时错误时由硬件设置。	0x0	RCW1
14	OCF	输出比较标志位 主模式下，当计数器 LINTCSR.CNT 等于 LINOCR.OC2 的值时，OCF 置 1。 从模式下，当计数器 LINTCSR.CNT 等于 LINOCR.OC1 或 LINOCR.OC2 的值时，OCF 置 1。 LINTCSR.MODE=0 和 LINTCSR.IOT=1 时，OCF 置 1，LINFlexD 进入空闲状态。 当 LINTCSR.MODE=0 时，OCF 在初始化模式下被硬件清除。 当 LINTCSR.MODE=1 时，无论 LIN 状态如何，OCF 都保持其状态。	0x0	RCW1
13	BEF	位错误标志 当 LINFlexD 检测到一个比特错误时，BEF 由硬件置 1。BEF 在初始化模式下被硬件清除。	0x0	RCW1
12	CEF	校验错误标志 当收到的校验值与硬件计算的校验值不一致时，CEF 由 1 置 1。 CEF 在初始化模式下被硬件清除。 注：如果 LINCR1.CCD 或 LINCR1.CFD 被设置，CEF 就不会被设置。	0x0	RCW1
11	SFEF	同步字段错误标志 当非连续的同步段产生同步段错误时，SFEF 由硬件置 1。 SFEF 在初始化模式下被硬件清除。	0x0	RCW1
10	SDEF	间隔符错误标志 当间隔符太短（少于一个比特时间）而发生间隔符错误时，SDEF 由硬件置 1。 SDEF 在初始化模式下被硬件清除。	0x0	RCW1
9	IDPEF	ID 奇偶校验错误标志 LINFlexD 检测到 ID 奇偶性的错误，IPDEF 由硬件设置 1。IPDEF 在初始化模式下由硬件清除。	0x0	RCW1
8	FEF	帧错误标志 当发生帧错误（无效停止位）时，FEF 由硬件设置 1。FEF 在初始化模式下由硬件清除。	0x0	RCW1
7	BOF	缓冲区溢出标志 当收到一个新的字节，并且 LINSR.RMB 位没有被清除时，BOF 由硬件	0x0	RCW1

		置 1。 BOF 在初始化模式下由硬件清除。		
6:1	Res	保留	0x0	-
0	NF	噪声标志 当接收字符中检测到噪声时，NF 由硬件设置。在初始化模式下，NF 由硬件清除。	0x0	RCW1

28.5.5 UART 模式控制寄存器(UARTCR)

偏移地址：0x010
复位值： 0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MIS	CSP			OSR				ROSE	NET			DTU	SBUR		WLS
R/W	R/W			R/W				R/W	R/W			R/W	R/W		R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDFL_TFC		RDFL_RFC				RFBM	TFBM	WL1	PC1	RxEn	TxEn	PC0	PCE	WL0	UART
R/W		R/W				R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31	MIS	监测空闲状态： 1：UARTCTO 监测接收的空闲状态； 0：UARTCTO 监测接收数据的位数； MIS 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
30:28	CSP	配置采样点(i) CSP 位在降低过采样过程中选择采样点。对于指定的过采样率，CSP 可以提供以下范围的值： OSR: 4 → CSP: 2,3 OSR: 5 → CSP: 2,3,4 OSR: 6 → CSP: 3,4,5 OSR: 8 → CSP: N/A CSP 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
27:24	OSR	过采样率-当 Reduced 过采样使能时选取采样 1bit 的采样数。 OSR 所允许的值分别为 4、5、6 和 8。 当空闲状态监测使能时(MIS=1)，允许的 OSR 值分别为 4 和 8。 OSR 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
23	ROSE	降低过采样使能： 0：每个 bit 位被过采样 16 次； 1：OSR 位确定过采样率； ROSE 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
22:20	NEF	期望帧数： NEF 位设置 UART 接收模式下的期望帧数。如果 DTU 置位，则在接收到配置的帧数后，将重置 UART 超时计数器。 NEF 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W

19	DTU	禁用 UART 模式下的超时： 1：在接收到配置的数据帧数后，将禁用 UART 模式下的超时； 0：软件处理超时。 内部计数器来计数接收到的帧数。 **缓冲模式：** 当接收的字节数(rec_byte_cnt)等于 RDFL 时，计数器清零。此时，DRF 置位，num_of_frames（接收到的帧数）和 rec_byte_cnt 清零； 当 num_of_frames 等于 NEF 时，disable_timeout 置位，num_of_frames 清零。计时器重新启动，计数值清 0。 **FIFO 模式：** 当 num_of_frames 等于 NEF 时，disable_timeout 置位，num_of_frames 清零，计时器重新启动，计数值清 0。 注意： ● disable_timeout 重新启动计时器，并重新启动 timeout; ● 计数器复位意味着计数器清 0 并重新开始计数且 timeout 启用； 只有在设置了 UART 位时，DTU 才能在初始化模式下配置。	0x0	R/W
18:17	SBUR	UART 接收模式下的停止位数： 00：1 位停止位 01：2 位停止位 10：3 位停止位 11：保留 UART 发送和接收时，在 GCR.STOP 和 UARTCR.SBUR 中设置相同的停止位。UART 仅接收时，只需要设置 SBUR 位指定停止位数。 SBUR 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
16	WLS	UART 模式下的特殊字长： 1：在接收时使能 12 位+奇偶位的数据格式； 0：在接收时禁用 12 位+奇偶位的数据格式； 当 WLS=1 时，尽管在此模式下默认启用了奇偶校验，WL 和 PCE 位对 UART 接收都没有影响，而且可以使用 PC0/1 位来选择奇偶校验方式。WLS 位在 UART 接收模式下具有优先级。 WLS 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
15:13	TDFL_TFC	数据发送段长度/Tx FIFO 计数器 TDFL_TFC 有 2 个功能，取决于当前的操作模式。 UART Buffer 模式： UART buffer 模式下（TFBM=0），TDFL_TFC 定义了要发送的字节数，并可读可写： x00：1 byte x01：2 bytes x10：3 bytes x11：4 bytes bit15 为保留位	0x0	R/W

		当 UART 数据长度为 half-word（WL=10 or 11）时，TDFL_TFC 的有效值仅为 x01 和 x11。 UART FIFO 模式： UART FIFO 模式下（TFBM=1），TDFL_TFC 定义了 Tx FIFO 中的字节数，并只读： 000：空 001：1 byte 010：2 bytes 011：3 bytes 100：4 bytes 其他值：保留 当 UART 被置位时，TDFL_TFC 可以在任何模式下读取，但只能在初始化模式下写入。		
12:10	RDFL_RFC	数据接收段长度/Rx FIFO 计数器 RDFL_RFC 有 2 个功能，取决于当前的操作模式。 UART Buffer 模式： UART buffer 模式下（TFBM=0），RDFL_RFC 定义了接收的字节数，并可读可写： x00：1 byte x01：2 bytes x10：3 bytes x11：4 bytes bit12 为保留位 当 UART 数据长度为 half-word（WL=10 or 11）时，RDFL_RFC 的有效值仅为 x01 和 x11。 UART FIFO 模式： UART FIFO 模式下（TFBM=1），RDFL_RFC 定义了 Rx FIFO 中的字节数，并只读： 000：空 001：1 byte 010：2 bytes（当 WLS=1 时为 1.5byte） 011：3 bytes 100：4 bytes（当 WLS=1 时为 2 x 1.5byte） 其他值：保留 当 UART 被置位时，RDFL_RFC 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W
9	RFBM	Rx FIFO/ Buffer 模式： 1：Rx FIFO 模式使能 0：Rx Buffer 模式使能 当 UART 被置位时，RFBM 可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W

<table>
<tr><td>8</td><td>TFBM</td><td>Tx FIFO/ Buffer 模式：
1：Tx FIFO 模式使能
0：Tx Buffer 模式使能
当 UART 被置位时，TFBM 可以在任何模式下读取，但只能在初始化模式下写入</td><td>0x0</td><td>R/W</td></tr>
<tr><td>7</td><td>WL1</td><td>UART 模式下的传输字长：
<table>
<tr><th>WL1</th><th>WL0</th><th>描述</th></tr>
<tr><td>0</td><td>0</td><td>7 数据位+校验位</td></tr>
<tr><td>0</td><td>1</td><td>PCE=0 时，8 数据位；
PCE=1 时，8 数据位+校验位</td></tr>
<tr><td>1</td><td>0</td><td>15 数据位+校验位</td></tr>
<tr><td>1</td><td>1</td><td>PCE=0 时，16 数据位；
PCE=1 时，16 数据位+校验位</td></tr>
</table>
当 UART 被置位时，WL1 可以在任何模式下读取，但只能在初始化模式下才能写入。</td><td>0x0</td><td>R/W</td></tr>
<tr><td>6</td><td>PC1</td><td>设置奇偶校验：
<table>
<tr><th>PC1</th><th>PC0</th><th>描述</th></tr>
<tr><td>0</td><td>0</td><td>偶校验</td></tr>
<tr><td>0</td><td>1</td><td>奇校验</td></tr>
<tr><td>1</td><td>0</td><td>奇偶校验以逻辑 0 进行检测</td></tr>
<tr><td>1</td><td>1</td><td>奇偶校验以逻辑 1 进行检测</td></tr>
</table>
当 UART 被置位时，PC1 可以在任何模式下读取，但只能在初始化模式下才能写入。</td><td>0x0</td><td>R/W</td></tr>
<tr><td>5</td><td>RxEn</td><td>接收机使能：
1：接收机使能
0：接收机使无效
RxEn 仅在设置了 UART 位时才可写。</td><td>0x0</td><td>R/W</td></tr>
<tr><td>4</td><td>TxEn</td><td>发送器使能：
1：发送器使能
0：发送器无效
当 TxEn 置位,BDRL.DATA 位置 0 时，开始发送；
TxEn 仅在设置了 UART 位时才可写。</td><td>0x0</td><td>R/W</td></tr>
<tr><td>3</td><td>PC0</td><td>设置奇偶校验：
<table>
<tr><th>PC1</th><th>PC0</th><th>描述</th></tr>
<tr><td>0</td><td>0</td><td>偶校验</td></tr>
<tr><td>0</td><td>1</td><td>奇校验</td></tr>
<tr><td>1</td><td>0</td><td>奇偶校验以逻辑 0 进行检测</td></tr>
<tr><td>1</td><td>1</td><td>奇偶校验以逻辑 1 进行检测</td></tr>
</table>
PC0 可以在任何模式下读取，但只能在初始状态下，UART 位被设置时才能写入。</td><td>0x0</td><td>R/W</td></tr>
<tr><td>2</td><td>PCE</td><td>奇偶校验控制使能：
1：奇偶传输/检测使能
0：奇偶传输/检测禁用</td><td>0x0</td><td>R/W</td></tr>
</table>

<table>
<tr><td></td><td></td><td>当 UART 被置位时，PCE 可以在任何模式下读取，但只能在初始化模式下才能写入。</td><td></td><td></td></tr>
<tr><td>1</td><td>WL0</td><td>UART 模式下的传输字长：
<table>
<tr><th>WL1</th><th>WL0</th><th>描述</th></tr>
<tr><td>0</td><td>0</td><td>7 数据位+校验位</td></tr>
<tr><td>0</td><td>1</td><td>PCE=0 时，8 数据位；
PCE=1 时，8 数据位+校验位</td></tr>
<tr><td>1</td><td>0</td><td>15 数据位+校验位</td></tr>
<tr><td>1</td><td>1</td><td>PCE=0 时，16 数据位；
PCE=1 时，16 数据位+校验位</td></tr>
</table>
当 UART 被置位时，WL0 可以在任何模式下读取，但只能在初始化模式下才能写入。</td><td>0x0</td><td>R/W</td></tr>
<tr><td>0</td><td>UART</td><td>UART 模式使能：
1：UART 模式
0：LIN 模式
UART 可以在任何模式下读取。但只能在初始状态下写入</td><td>0x0</td><td>R/W</td></tr>
</table>

注意：

1 写 UARTCR 寄存器时，保留位需写 0；

2 在 UART 模式下，buffer 模式时 LINFlexD 不支持特殊字节长度的通信(UARTCR.WLS=1);

3 在 UART 模式下，FIFO 模式时只有当 UARTCR.WLS=11 时，LINFlexD 才支持特殊字节长度的通信(UARTCR.WLS=1);

28.5.6 UART 模式状态寄存器(UARTSR)

偏移地址：0x0014

复位值：0x0000_0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SZF	OCF	PE				RMB	FEF	BOF	RDI	WUF	RFNE	TO	DRF_RFE	DTF_TFF	NF
RCW1	RCW1	RCW1				RCW1	RCW1	RCW1	RO	RCW1	RO	RCW1	RO	RO	RCW1

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	-
15	SZF	Stuck at Zero 零状态标志 当硬件检测到 100 个显性电平时，SZF 被置为 1。如果 LINIER.SZIE=1 则会产生一个中断。 当 LINFlexD 在非初始化模式且 UARTCR.UART=1，SZF 反映了与 LINESR.SZF 相同的值。	0x0	RCW1
14	OCF	输出比较标志/Output Compare： 1：超时计数器的内容与 LINOCR 的内容相匹配。 0：没有发生输出比较事件。 当设置 OCF 时，如果为 LINIER.OCIE = 1，则产生一个中断。 当 LINFlexD 在非初始化模式且 UARTCR.UART=1，OCF 反映了与	0x0	RCW1

		LINESR.OCF 相同的值。 注意：对于超过 1 Mbit/s 的波特率，OCF 标志不可用。		
13:10	PE	奇偶校验错误标志—表示相应接收数据字节中的奇偶校验错误：1：检测到奇偶校验错误。 0：未检测到奇偶校验错误。 在收到完整的帧后，将检查奇偶校验。PE 位的使用情况取决于 UARTCR.WLS 和 UARTCR.WL1 的值： 当 UARTCR.WLS=0 和 UARTCR.WL1 = 0 时: PE[3]= 已接收到的第四个字节的奇偶校验错误标志。 PE[2] = 已接收到的第三个字节的奇偶校验错误标志。 PE[1] = 已接收到的第二个字节的奇偶校验错误标志。 PE[0] = 已接收到的第一个字节的奇偶校验错误标志。 当 UARTCR.WLS=1 或 UARTCR.WL1 = 1: PE[3]= 不使用 PE[2] = 已接收到的第三个字节的奇偶校验错误标志。 PE[1] = 不使用 PE[0] = 已接收到的第一个字节的奇偶校验错误标志。	0x0	RC W1
9	RMB	释放消息缓冲区： 1：缓冲区数据已经可以由软件读取； 0：缓冲区是未被占用的。 当 LINFlexD 在非初始化模式且 UARTCR.UART=1，RMB 与 LINSR.RMB 的值相同。	0x0	RC W1
8	FEF	帧错误标志: 当 LINFlexD 检测到帧错误（无效的停止位）时，FEF 由硬件置 1。如果为 LINIER.FEIE=1，则会产生一个中断。 当 LINFlexD 在非初始化模式且 UARTCR.UART=1，FEF 反映了与 LINESR.FEF 相同的值。	0x0	RC W1
7	BOF	FIFO/缓冲区溢出标志 在 UART 缓冲区模式下，当接收到一个新字节且未清除 RMB 位时，由硬件设置 BOF。如果是 LINCR.RBLM=1 中，新接收到的消息将被丢弃。如果是 LINCR.RBLM=0，新接收到的消息将覆盖缓冲区。 在 UART FIFO 模式下，当接收到一个新的字节并且 RxFIFO 已满时，将设置 BOF。在 UART FIFO 模式下，一旦 RxFIFO 满，无论 RBLM 位的值如何，新接收的消息将被丢弃。 当设置了 BOF 时，如果是 LINIER.BOIE=1，则会产生一个中断。 当 LINFlexD 在非初始化模式且 UARTCR.UART=0，BOF 反映了与 LINESR.BOF 相同的值。	0x0	RC W1
6	RDI	接收机数据输入—当 UARTCR.UART=1 时，表示 ipp_ind_linrx 引脚的当前状态。	0x0	RO
5	WUF	唤醒标志 在睡眠模式下，当在 ipp_ind_linrx 引脚检测到一个下降沿时 WUF 将会被硬件置 1，如果 LINIER.WUIE=1 则产生一个中断。 当 LINFlexD 在非初始化模式且 UARTCR.UART=1，WUF 反映了与	0x0	RC W1

		LINSR.WUF 相同的值。		
4	RFNE	接收 FIFO 非空 当接收 FIFO 中至少存在一个数据字节时，在 UART FIFO 模式(RFBM=1) RFNE 由硬件置 1。 RFNE 是一个用于调试目的的只读位。 RFNE 可以在发生超时事件时被软件使用。	0x0	RO
3	TO	超时 当 UART 超时发生时，TO 由硬件置 1—也就是 UARTCTO 的值等于超时的预设值(UARTPTO 寄存器设置)。如果 LINIER.TOIE = 1，则会产生一个中断。 在 UART 缓冲区和 UARTFIFO 模式下，当 UART 接收超时事件发生时，应使用 GCR.SR 位将接收机 FSM 重置为空闲状态。	0x0	RC W1
2	DRF_RFE	数据接收完成标志/RxFIFO 空标志 DRF_RFE 位的功能取决于 LINFlexD 是在 UART 缓冲模式还是 UARTFIFO 模式下。 **UART Buffer 模式:** 在 UART **Buffer** 模式(RFBM=0)中，使用 DRF_RFE 的 DRF（数据接收标志）功能。当已收到 UARTCR.RDFL 编程的字节数时，DRF 在 UART 缓冲模式由硬件置 1。如果为 LINIER.DRIE=1，则会产生一个中断。 当接收到最后一帧的配置的有效停止位数时， DRF 将置为 1。 如果帧错误出现在配置的最后一个 STOP 位中，则不管奇偶校验错误、溢出错误或帧错误，都会将 DRF 置 1。 当 LINFlexD 在非初始化模式且 UARTCR.UART=1，DRF 反映了与 LINSR.DRF 相同的值。 DRF 可以通过软件读取/清除。写入“1”将清除 DRF。 **UART FIFO 模式:** 在 UART FIFO 模式(RFBM=1)中，使用 DRF_RFE 的 RFE(RxFIFO 空)功能。当 RxFIFO 为空时，由硬件在 UART FIFO 模式下置 RFE 为 1。 RFE 是一个用于调试目的的只读位。	0x0	RC W1/ RO
1	DTF_TFF	数据发送完成标志/TxFIFO 满标志 DTF_TFF 位功能取决于 LINFlexD 是在 UART buffer 模式还是 UART FIFO 模式下。 UART Buffer Mode: 在 UART Buffer 模式(RFBM=0)下，使用 DTF_TFF 的 DTF（数据发送标志）功能。当数据发送完成时，DTF 由硬件置 1。如果 LINIER. DTIE = 1，则会产生一个中断。 当 LINFlexD 在非初始化模式且 UARTCR.UART=1，DTF 反映了与 LINSR.DTF 相同的值。 DTF 可以通过软件读取/清除。写入“1”将清除 DTF。 UART FIFO Mode : 在 UART FIFO 模式(RFBM=1)中，使用 DTF_TFF 的 TFF(TxFIFO	0x0	RC W1/ RO

		满)功能。在 UART FIFO 模式下当 TxFIFO 已满时，TFF 由硬件设置为 1。 TFF 是一个用于调试目的的只读位。		
0	NF	噪声标志位——当在接收的字符中检测到噪声时，噪声标志由硬件设置为 1。 当 LINFlexD 在非初始化模式且 UARTCR.UART=1，NF 反映了与 LINESR.NF 相同的值。 在减少过采样期间(UARTCR.ROSE=1)，只有在 UARTCR .OSR = 8 时才启用 NF 位。	0x0	RC W1

28.5.7 LIN 超时控制状态寄存器(LINTCSR)

地址偏移：0x18

复位值：0x00000200

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留					MODE	IOT	TOCE	CNT							
					R/W	R/W	R/W	RO							

位	标记	功能描述	复位值	读写
31:11	Res	保留	0x0	-
10	MODE	超时计数器模式 1：输出比较模式 0：LIN 模式 注：当 UARTCR.UART=1 时，总是将 LINTCSR.MODE 设置位'1'。在帧之间将 LINTCSR.MODE 从'1'切换到'0'之前，LINOCR 的值加载为 0xFFFF。 MODE 可以在任何模式下被读取，但只能在初始化模式下被写入。	0x0	R/W
9	IOT	超时空闲 1：LIN 状态机在超时事件中复位到空闲状态 0：LIN 状态机不会在超时事件中重置为空闲状态。 超时空闲功能仅在 MODE=0 时适用。IOT 可以在任何模式下被读取，但只能在初始化模式下被写入。	0x1	R/W
8	TOCE	超时计数器使能位 1：使能。UARTSR.OCF 标志在输出比较事件中置 1。 0：关闭。UARTSR.OCF 标志在输出比较事件中不被置 1。 TOCE 总是可以由软件在初始化模式下进行配置。如果在 LIN 模式下计数器被配置且 LINFlexD 不在初始化模式下，那么硬件会控制 TOCE。	0x0	R/W
7:0	CNT	超时计数器值 注：为了使该计数器的功能正常运行，LINIBRR 应大于等于 5。	0x0	RO

LINTCSR 寄存器包含 LIN 超时功能的控制和状态位。

注意：当 LINTCSR.MODE=0 时，发送或接收引脚上的任何动作都会导致 LIN 输出比较寄存器(LINOCR)的 8 位字段输出比较值 2(OC2)的数值发生不必要的变化。如果 LNFlexD 在 LIN

模式下被启用，且 LINTCSR.MODE 的值从’1’变为’0’.LINOCR.OC1 和 LNOCR.OC2 的旧值将被保留。因此，如果 LINFlexD 从 UART 模式重新配置为 LIN 模式，或者 LINTCSR.MODE 从’1’改为’0’，当 LIN 通信开始时就会产生一个不正确的超时异常。为了避免这种情况，将 LINTCSR.MODE 设置为’1’，在 LIN 模式下重新配置 LINFlexD，然后将 LINTCSR.MODE 重置为’0’。在帧之间将 LINTCSR.MODE 从’1’切换到’0’之前，写 0xFFFF 到 LINOCR。

28.5.8 LIN 输出比较寄存器(LINOCR)

偏移地址：0x1C
复位值：0x0000FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2								OC1							
R/W								R/W							

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	-
15:8	OC2	输出比较值 2	0xFF	R/W
7:0	OC1	输出比较值 1	0xFF	R/W

LINOCR 寄存器的值用于与 LINTCSR.CNT 比较，该寄存器只能在输出比较模式下由软件可写。

28.5.9 LIN 超时控制寄存器(LINTOCR)

偏移地址：0x20
复位值：0x00000E2C

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

位	标记	功能描述	复位值	读写
31:12	Res	保留	0x0	-
11:8	RTO	应答超时值：1 个字节的应答超时时间（以比特时间计）； 复位值为 0Eh=14，对应于 $T_{Response\_Maximum}$=1.4x $T_{Response\_Nominal}$。	0xE	R/W
7	Res	保留	0x0	-
6:0	HTO	帧头超时值 HTO 帧头超时时间（以比特时间计）。HTO 只能在从模式下写入。帧头超时编程时不考虑 11 位间隔字段。	0x2C	R/W

注意：LINTOCR 寄存器包含 LIN 模式应答和帧头超时值。

28.5.10 LIN 小数波特率寄存器(LINFBRR)

地址偏移：0x24
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留												FBR			
												R/W			

位	标记	功能描述	复位值	读写
31:4	Res	保留	0x0	-
3:0	FBR	小数波特率 0000：分频系数(LDIV) = 0 0001：分频系数(LDIV) = 1/16 0010：分频系数(LDIV) = 2/16 0011：分频系数(LDIV) = 3/16 0100：分频系数(LDIV) = 4/16 0101：分频系数(LDIV) = 5/16 0110：分频系数(LDIV) = 6/16 0111：分频系数(LDIV) = 7/16 1000：分频系数(LDIV) = 8/16 1001：分频系数(LDIV) = 9/16 1010：分频系数(LDIV) = 10/16 1011：分频系数(LDIV) = 11/16 1100：分频系数(LDIV) = 12/16 1101：分频系数(LDIV) = 13/16 1110：分频系数(LDIV) = 14/16 1111：分频系数(LDIV) = 15/16 FBR 在任何模式下可读，但仅在初始化模式下可写。	0x0	R/W

LINFBRR 设置 LIN 波特率的小数部分，如 28.3.2.7 “波特率生成”中所述。
注意：当 LINCR1.LASE=1 时，只有在设置了 LINSR.AUTOSYNC COMP 后才能读取 LINFBRR 以获得正确的值。
注意：当降低过采样使能时(UARTCR.ROSE=1)，不能使用 LINFBRR。

28.5.11 LIN 整数波特率寄存器(LINIBRR)

地址偏移：0x28
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留												IBR			
												R/W			

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

位	标记	功能描述	复位值	读写
31:20	Res	保留	0x0	-
19:0	IBR	整数波特率 IBR 位与 LINFBRR 中的小数波特率位一起，设置 LIN 波特率。 0x0：LIN 时钟禁用 0x1：尾数（LDIV） = 1 0xFFFFE：尾数（LDIV） = 1048574 0xFFFFF：尾数（LDIV） = 1048575 IBR 可以在任何模式下被读取，但只能在初始化模式下被写入	0x0	R/W

LINIBRR 设置 LIN 波特率的整数部分，如 28.3.2.7 波特率生成中所述。
注：当 LINCR1.LASE=1 时，只有在设置了 LINSR.AUTOSYNC_COMP 后才能读取 LINIBRR 以获得正确的值。

28.5.12 LIN 校验和段寄存器(LINCFR)

偏移地址：0x02C
复位值：0x0000_0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								CF							
								R/W							

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	-
7:0	CF	校验和位: 当 LINCR1.CCD=0 时，CF 位是只读的，由硬件计算。 当 LINCR1.CCD=1 时，CF 位可以用软件编写。	0x0	R/W

注意：内部校验和的值(用 ipg_baud_clk/16*LDIV 时钟)有 4 到 6 个 ipg_clk 时钟周期的延迟来反映 LINCFR。
注意：由于 LINFlexD 的两个输入时钟之间的同步结构，对这个寄存器的写操作会有 2 到 3 个 ipg_clk 的时钟周期之间延迟。

表 28-6 LINCFR 用法

1	1	读/写	无
0	0	只读	无
0	1	读/写	编程校验和
0	0	只读	计算校验和

28.5.13 LIN 控制寄存器 2(LINCR2)

地址偏移：0x30
复位值：0x00006000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBDE	IOBE	IOPE	WURQ	DDRQ	DTRQ	ABRQ	HTRQ	保留							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W								

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	-
15	TBDE	2 比特间隔符使能 1：间隔符长度为 2 比特 0：间隔符长度为 1 比特 TBDE 可以在任何模式下被读取，但只能在初始化模式下被写入	0x0	R/W
14	IOBE	位错误时进入空闲状态 1：位错会重置 LIN 状态机 0：位错不会重置 LIN 状态机 IOBE 可以在任何模式下被读取，但只能在初始化模式下被写入	0x1	R/W
13	IOPE	奇偶校验错误时进入空闲状态 1：重置状态机 0：不重置状态机 IOPE 可以在任何模式下被读取，但只能在初始化模式下被写入	0x1	R/W
12	WURQ	唤醒产生请求 设置 WURQ 会生成唤醒请求。唤醒期间发送的字符从 BDRL.DATA0 复制。当唤醒字符已发送完成后，WURQ 由硬件重置。WURQ 不能在睡眠模式下被设置。软件必须在设置 WURQ 之前退出休眠模式。发送唤醒请求时，不会检查位错误	0x0	R/W
11	DDRQ	数据丢弃请求 如果帧与节点无关，软件将 DDRQ 设置为停止数据接收。一旦 LINFlexD 忽略应答并进入 Idle 状态，DDRQ 将由硬件重置。对于 LIN 从设备，只能在 LINSR.HRF=1 时设置 DDRQ，并且 ID 由软件过滤	0x0	R/W
10	DTRQ	数据发送请求 软件在从属模式下设置 DTRQ，以请求发送存储在缓冲数据寄存器中的 LIN 数据字段。DTRQ 只能在 LINSR.HRF=1 时设置（以确保只有在 reguest 完成时才发送数据） 在只有在该请求完成后，或该请求被丢弃或发生错误的条件下，通	0x0	R/W

		过硬件清除 DTRQ。 在主模式下，当 BIDR.DIR=1 且帧头传输完成时，通过硬件设置 DTRQ。		
9	ABRQ	中止请求 ABRQ 由软件设置为中止当前发送，LINFlexD 在当前位结束时中止传输。当传输中止时，硬件将清除 ABRQ。ABRQ 可以中止唤醒请求，也可以在 UART 模式下使用。	0x0	R/W
8	HTRQ	帧头发送请求 通过软件设置 HTRQ 以请求发送 LIN 帧头。当 reguest 完成或请求中止时，硬件将清除 HTRQ。 HTRQ 在 UART 模式下不起作用。 注：在主模式下，如果同时设置 HTRQ 和 ABRQ，则 ABRQ 无效。类似地，在从模式下，接收帧头后，如果同时设置 DTRQ 和 ABRQ，则 ABRQ 不起作用。	0x0	R/W
7:0	Res	保留	0x0	-

LINCR2 寄存器包含与缓冲器操作有关的控制和状态位
注意：当访问 LINCR2 时，每个保留位都应以其原来的复位值写入。

28.5.14 缓冲区标识符寄存器(BIDR)

地址偏移：0x34
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留			DFL			DIR	CCS	保留		ID					
			R/W			R/W	R/W			R/W					

位	标记	功能描述	复位值	读写
31:13	Res	保留	0x0	-
12:10	DFL	数据字段长度——帧的应答部分的数据字节数 DFL= 数据字节数-1	0x0	R/W
9	DIR	方向- 控制数据段的传输方向 1：LINFlexD 将数据从 BDRM/BDRL 寄存器中发送出去 0：LINFlexD 接收来自 BDRM/BDRL 寄存器中复制的数据	0x0	R/W
8	CCS	经典校验和-控制应用于当前消息的校验和类型： 1：仅覆盖数据字段的经典校验和，与 LIN 规范版本 1.3 及更低版本兼容。 0：覆盖标识符和数据字段的增强型校验和，这与 LIN 规范版本 2.0 及更高版本兼容。	0x0	R/W
7:6	Res	保留	0x0	-
5:0	ID	标识符 ID—标识符字段的不含奇偶校验位标识符部分，该 ID 字段只能在主模式下写入（LINCR1.MME=1）。	0x0	R/W

BIDR 寄存器包含提供有关传输标识符的信息和其他相关信息。
注：当从模式（Tx 或 Rx）中的 ID 过滤器（使能）与接收到的 ID 匹配时，必须更新 BIDR 寄

存器的所有字段（ID、CSS，DIR，DFL）

28.5.15 缓冲区有效低位数据寄存器寄存器(BDRL)

偏移地址：0x38
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA3								DATA2							
R/W								R/W							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA1								DATA0							
R/W								R/W							

位	标记	功能描述	复位值	读写
31:24	DATA3	数据字节 3-数据字段的字节 3	0x0	R/W
23:16	DATA2	数据字节 2-数据字段的字节 2	0x0	R/W
15:8	DATA1	数据字节 1-数据字段的字节 1	0x0	R/W
7:0	DATA0	数据字节 0-数据字段的字节 0	0x0	R/W

BDRL 寄存器包含 8 字节数据缓冲区中的 0-3 个字节。

28.5.16 缓冲区有效高位数据寄存器寄存器(BDRM)

偏移地址：0x3C
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA7								DATA6							
R/W								R/W							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA5								DATA4							
R/W								R/W							

位	标记	功能描述	复位值	读写
31:24	DATA7	数据字节 7-数据字段的字节 7	0x0	R/W
23:16	DATA6	数据字节 6-数据字段的字节 6	0x0	R/W
15:8	DATA5	数据字节 5-数据字段的字节 5	0x0	R/W
7:0	DATA4	数据字节 4-数据字段的字节 4	0x0	R/W

BDRM 寄存器包含 8 字节数据缓冲区的 4-7 个字节。

28.5.17 标识过滤器使能寄存器(IFER)

偏移地址：0x40
复位值：0x00000000

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

位	标记	功能描述	复位值	读写
31:16	Res	保留	0x0	-
15:0	FACT	过滤器激活：每个 ID 过滤器有一个比特位； 软件设置 FACT[n]以激活标识符列表模式下的过滤器 n； 在标识符掩码模式下，FACT[2n+1]对相应的过滤器没有影响，因为它们作为标识符 2n 的掩码； FACT[n]可以在任何模式下被读取，但只能在初始化模式下写入。	0x0	R/W

IFER 可启用/禁用标识符过滤器，每个过滤器都有一个过滤器激活位 FACT。

28.5.18 标识符过滤器匹配索引(IFMI)

偏移地址：0x044
复位值：0x0000_0000

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

位	标记	功能描述	复位值	读写
31:5	Res	保留	0x0	-
4:0	IFMI	过滤器匹配索引-包含与接收到的标识符对应的索引。 在 filter 与 filter x 匹配时，IFMI[N:0] = x+1。 在没有匹配时，IFMI 等于 0x00。 IFMI 可用于直接用 RAM 写入或读取数据	0x0	RO

IFMI 寄存器包含与接收到的标识符对应的索引。它可以用来直接在 RAM 中读取或写数据。

28.5.19 标识符过滤器模式寄存器(IFMR)

移地址：0x048
复位值：0x0000_0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								IFM							
								R/W							

位	标记	功能描述	复位值	读写
31:8	Res	保留	0x0	-
7:0	IFM[n]	筛选模式: 0：过滤器 2n 和 2n+1 处于标识符列表模式(list mode)。 1：过滤器 2n 和 2n+1 处于掩码模式。过滤器 2n+1 是过滤器 2n 的掩模(mask)。 IFM[n]可以在任何模式下读取，但只能在初始化模式下写入。	0x0	R/W

IFMR 寄存器配置过滤器的模式。每对过滤器都有一个 IFM 位。

28.5.20 标识符过滤器控制寄存器(IFCR)

偏移地址：0x04C+(n x 0x4)
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留			DFL			DIR	CCS	保留		ID					
			R/W			R/W	R/W			R/W					

位	标记	功能描述	复位值	读写
31:13	Res	保留	0x0	-
12:10	DFL	数据字段长度——帧的应答部分的数据字节数 DFL= 数据字节数-1	0x0	R/W
9	DIR	方向- 控制数据段的传输方向 1：LINFlexD 将数据从 BDRM/BDRL 寄存器中发送出去 0：LINFlexD 接收来自 BDRM/BDRL 寄存器中复制的数据	0x0	R/W
8	CCS	经典校验和-控制应用于当前消息的校验和类型： ·1：经典校验和，仅覆盖数据字段，与 LIN 规范版本 1.3 及更低版本兼容。 ·0：增强型校验和，覆盖标识符和数据字段，这与 LIN 规范版本 2.0 及更高版本兼容。	0x0	R/W
7:6	Res	保留	0x0	R/W
5:0	ID	标识符 ID—标识符字段的不含奇偶校验位标识符部分。	0x0	R/W

注意：每个过滤器都有一个 ID 滤波控制寄存器，IFCRn 的偶数位（IFCR0，IFCR2，IFCR4

等）可以在列表模式和掩码模式下使用
IFCRn 的奇数位（IFCR1，IFCR3，IFCR5 等）在列表模式下为对应的过滤器，在 ID 掩码模式下为前置寄存器，比如，过滤器 0 和 1 被配置为掩码模式，那么 IFCR1 则作为 IFCR0 的掩码。
IFCRn 寄存器在正常模式下只读，在初始化模式下才可以被写。

28.5.21 全局控制寄存器(GCR)

偏移地址：0x8C
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留										TDFBM	RDFBM	TDLIS	RDLIS	STOP	SR
										R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:6	Res	保留	0x0	-
5	TDFBM	发送数据的第一位 MSB：控制发送数据（仅有效载荷）的第一位是各自缓冲数据寄存器的最高位还是最低位： 1：发送数据的第一位是缓冲数据寄存器的最高位(BDRM/BDRL[7]、BDRM/BDRL[15]、BDRM/BDRL[23]、BDRM/BDRL[31])； 0：发送数据的第一位是缓冲数据寄存器的最低位(BDRM/BDRL[0]、BDRM/BDRL[8]、BDRM/BDRL[16]、BDRM/BDRL[24])	0x0	R/W
4	RDFBM	接收数据第一位 MSB：控制接收数据（仅有效载荷）的第一位是否被映射到各自缓冲数据寄存器的最高位或最低位： 1：第一位的接收数据映射到缓冲数据寄存器的最高位(BDRM/BDRL[7]、BDRM/BDRL[15]、BDRM/BDRL[23]、BDRM/BDRL[31])； 0：接收到的数据的第一位被映射到缓冲区数据寄存器的最低位(BDRM/BDRL[0]、BDRM/BDRL[8]、BDRM/BDRL[16]、BDRM/BDRL[24])。	0x0	R/W
3	TDLIS	发送数据水平反转选择：控制传输数据的数据反转（仅有效载荷）： 1：传输数据被反转； 0：传输的数据不反转。	0x0	R/W
2	RDLIS	接收数据水平反转选择： 控制接收数据的数据反转（仅有效载荷）：1：接收数据的反转； 0：接收的数据不反转。	0x0	R/W
1	STOP	一个或两个停止位配置：控制所有字段（间隔符、同步段、ID、校验和、有效载荷-数据段）： 1：两个停止位。	0x0	R/W

<table>
<tr><td></td><td></td><td>0：一个停止位。</td><td></td><td></td></tr>
<tr><td>0</td><td>SR</td><td>软复位
向 SR 写入 1 可以执行 LINFlexD(FSM、FIFO 指针、计数器、定时器、状态和错误寄存器)的软复位，而不修改配置寄存器。SR 应被软件清除以执行进一步的操作（SR 未被硬件清除）；
SR 位只能在初始化模式下被软件写入。SR 的读取值始终为“0”；以下列出了通过将“1”写入 SR 来复位的寄存器字段：
<table>
<tr><td>LINSR</td><td>除 RXBUSY 和 AUTOSYNC_COMP 外的所有字段被复位</td></tr>
<tr><td>LINESR</td><td>所有字段被复位</td></tr>
<tr><td>LINTCSR</td><td>仅复位 CNT</td></tr>
<tr><td>UARTSR</td><td>除 RFE 和 TFF 外的所有字段被复位</td></tr>
<tr><td>UARTCR</td><td>只有 TFC 和 RFC 被复位</td></tr>
<tr><td>UARTCTO</td><td>所有字段被复位</td></tr>
</table></td><td>0x0</td><td>R/W</td></tr>
</table>

GCR 寄存器包含同时适用于 LIN 和 UART 模式的控制位。
GCR 寄存器在正常模式下为只读模式，并且只能在初始化模式下写入。

28.5.22 UART 预设超时寄存器(UARTPTO)

偏移地址：0x090
复位值：0x00000FFF

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				PTO											
				R/W											

位	标记	功能描述	复位值	读写
31:12	Res	保留	0x0	-
11:0	PTO	预设超时 PTO 定义了超时计数器的预设值。禁止使用零值，否则，将立即将 UARTSR[TO]状态位置 1。	0xFFF	R/W

UARTPTO 表示 UART 模式下超时寄存器的预设值，并根据要接收的数据来配置或者检测接收数据线上的空闲状态
UARTPTO 可以在任何时候被软件配置。
PTO 定义了超时计数器的预设值，禁止写 0；否则 UARTSR.TO 会被立即置位。

28.5.23 UART 当前超时寄存器(UARTCTO)

偏移地址：0x094
复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:12	Res	保留	0x0	-
11:0	CTO	当前超时计数器值： CTO 显示超时计数器当前状态的计数值。当 UARTPTO 被写入数据或者 UARTCTO=UARTPTO、或者软件/硬件复位时，UARTCTO 被复位。当 CTO 等于 UARTPTO.PTO 的值时，状态位 UARTDR.TO 被置位。	0x0	RO

注意：UARTCTO 寄存器表示 UART 模式下当前超时计数器的值。UARTCTO 和 UARTPTO 被一起用来监测 UART 模式下接收的数据位数，或者监测接收数据线上的空闲状态。
超时计数器：

1 从 0 开始计数向上计数；
2 （UARTCR.ROSE=0 时）在同步于 ipg_clk 的 ipg_baud_clk/(16*LDIV) 时钟下计数
3 （UARTCR.ROSE=1 时）在同步于 ipg_clk 的 ipg_baud_clk/(UART.OSR*LINIBRR.IBR)下计数
4 当 UARTCR.RXEN=1 时，自动使能；

因为需要同步，从内部计数值反映到 UARTCTO 的值上，大约有 4-6 到 ipg_clk 时钟的延迟。
超时计数器在以下条件下被复位：

1 当接收到帧数等于 UARTCR.NEF 时
2 UARTCTO 的值与 UARTPTO 相等时
3 UARTPTO 被写时
4 UARTSR.DRF 被置位且 UARTCR.DTU=1 时

29 控制器局域网(CAN FD)

29.1 简介

CAN（Controller Area Network）总线是一种可以在多主机情况下实现节点设备之间相互通信的总线标准。本产品搭载的 CAN 符合 CAN FD 协议并向下兼容 CAN 2.0B。

CAN 通信按帧组织。共有两种类型的帧结构：标准帧和扩展帧。对于 CAN 2.0，一帧最大传输数据为 8 个字节，而 CAN FD 一帧最多传输 64 个字节。所有 CAN 节点在总线访问方面都是相同的。

使用消息标识符完成数据寻址。在 CAN 网络中，只有一个节点必须传输具有特定标识符的消息。所有节点都接收所有消息，并且节点主控制器必须确定它是否有适当的消息标识符寻址。为了减少主控制器的负载，CAN 内核可以使用接收过滤器。这些过滤器将所有接收的消息标识符与用户可选择的位模式进行比较。仅当消息通过接收过滤器时，它才会存储在接收缓冲区中并通知主控制器。

CAN 帧标识符也用于总线仲裁。当具有较高优先级标识符的消息由另一个 CAN 节点发送时，CAN 协议控制器停止发送具有低优先级标识符的消息，并且。CAN 协议控制器自动尝试在下一个可能的发送位置重新发送已停止的消息。

29.2 主要特性

CAN 控制器支持以下特性：

- 支持 CAN 规格
  - CAN 2.0B(最多 8 个字节的有效载荷，经 Bosch 参考模型验证)
  - CAN FD

  (最多 64 字节的有效载荷，ISO 11898-1:2015 或者非 ISO Bosch)
- 可编程的比特率
  - CAN 2.0B 支持最高 1Mbit/s
  - CAN FD 支持最高 8Mbit/s(受收发器和所选择的 CAN 内核时钟频率的限制)
  - 可选择 AHB 分频时钟或者外部振荡器时钟
- 可编程波特率预分频器(1 至 1/256)
- 1 个接收缓冲区，FIFO 深度为 8
- 两个发送缓冲区
  - 主发送缓冲区(PTB) FIFO 深度为 1
  - 次发送缓冲区(STB) FIFO 深度为 16，按 FIFO 或优先级决定数据帧发送的先后顺序
- 16 个独立可编程的内部 29 位接收过滤器
- 可扩展特性
  - 单次发送模式(PTB 和 STB 都支持)
  - 监听模式
  - 回环模式(内部、外部)

- 收发器待机模式

- 扩展状态和错误报告
  - 捕获最后发生错误的类型和仲裁丢失的位置
  - 可编程错误警告限制
- 可配置的中断资源
- CiA 603 时间戳

29.3 功能描述

29.3.1 消息缓冲区

29.3.1.1 消息缓冲区的基本概念

消息缓冲区的概念如下图所示，所有缓冲区都足够大到可以存储最大长度的帧。

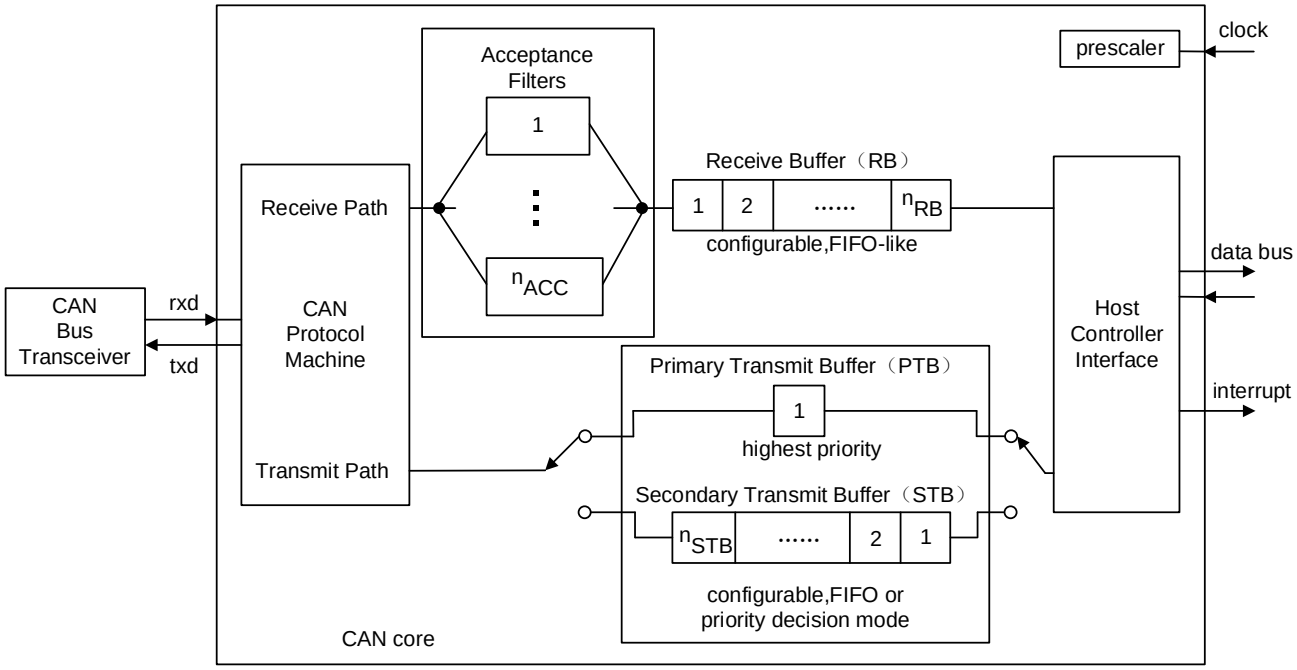


图 29-1 消息缓冲区示意图

29.3.1.2 接收缓冲区

为了减少主控制器接收帧的负载，内核使用接收过滤器。CAN 内核接收数据时， 接收过滤器会检查消息标识符。如果接收到的帧与其中一个接收过滤器的过滤条件相匹配，则它将被存储在接收缓冲区（RB）中，该接收缓冲区具有类似 FIFO 的行为。

根据可用消息缓冲区的数量，主控制器不需要立即读取传入消息。CAN 内核能够在每个接收信息上生成中断。如果用户使能了对应的“满”中断使能寄存器 RFIE 或者”几乎满”中断使能寄存器 RAFIE，当接收缓冲区(RB)“已满”或“几乎满”时，CAN 内核也会产生对应的中断。由于类似 FIFO 的行为，主控制器始终从 RB 读取最旧的消息。

表 29-1 接收缓冲区寄存器 RBUF – 标准格式（r-0）

地址	位								功能
	7	6	5	4	3	2	1	0	
RBUF	ID(7:0)								标识符
RBUF+1	-					ID(10:8)			标识符
RBUF+2	-								标识符
RBUF+3	ESI	-							标识符
RBUF+4	IDE=0	RTR	EDL (FDF)	BRS	DLC(3:0)				控制位
RBUF+5	KOER			TX	-				状态位
RBUF+6	保留								-
RBUF+7	保留								-
RBUF+8	d1(7:0)								数据字节 1
RBUF+9	d2(7:0)								数据字节 2
.	…								
RBUF+71	d64(7:0)								数据字节 64
RBUF+72	RTS(7:0)								CiA 603
.	…								.
RBUF+75	RTS(31:24)								CiA 603

表 29-2 接收缓冲区寄存器 RBUF – 扩展格式（r-0）

地址	位								功能
	7	6	5	4	3	2	1	0	
RBUF	ID(7:0)								标识符
RBUF+1	ID(15:8)								标识符
RBUF+2	ID(23:16)								标识符
RBUF+3	ESI	-		ID(28:24)					标识符
RBUF+4	IDE=1	RTR	EDL (FDF)	BRS	DLC(3:0)				控制位
RBUF+5	KOER			TX	-				状态位
RBUF+6	保留								-
RBUF+7	保留								-
RBUF+8	d1(7:0)								数据字节 1
RBUF+9	d2(7:0)								数据字节 2
.	…								
RBUF+71	d64(7:0)								数据字节 64
RBUF+72	RTS(7:0)								CiA 603
.	…								.

RBUF+75	RTS(31:24)	CiA 603

RBUF 寄存器（0x00 至 0x4f）指向的是接收缓存区 RB 中最早收到的消息。所有 RBUF 寄存器可以以任意顺序读取。

在 RBUF 中的 KOER 位和寄存器 ERRINFO 中的 KOER 意义相同。如果 RBALL=1，RBUF 中的 KOER 将变得有意义。

如果使能了回环模式，且 CAN 内核已经接收到了它自己发送的数据帧，则状态位 TX 会被置位为 1。接收时间戳 RTS 被存储在消息的后面，因此和 TTS 相比较， RTS 存储位置和 RBUF 槽有关。

29.3.1.3 发送缓冲区

出于帧发送的需要，提供了两个发送缓冲器（TB）。主 TB（PTB）优先级较高，但只能缓冲一帧。次 TB（STB）优先级较低，它可以在 FIFO 或优先模式下运行。PTB 和 STB 间的优先级是固定的，完全独立于 CAN 总线仲裁。总线仲裁是基于帧标识符的优先级来决定的。

可以命令 STB 发送一个或所有存储帧。在 FIFO 模式下，首先发送此缓冲区内最旧的帧。在优先级模式中，首先发送在该缓冲区内具有最高优先级的帧（基于帧标识符）。

无论帧标识符如何，对于 CAN 协议机器来说，位于 PTB 中的帧始终比 STB 中的帧具有更高的优先级。PTB 发送会停止并延迟 STB 发送。在成功发送 PTB 帧之后，STB 发送会自动重新启动。

如果 STB 已经启动发送且发送未完成，则 PTB 会在当前 STB 本帧发送完成后停止或者延迟 STB 发送。

在以下情况下，可以使用 PTB 发送打断 STB 发送：

1 请求 STB 发送所有存储的帧，并且在完成 STB 所有帧发送前， 主控制器请求 PTB 发送。

2 请求 STB 发送单个帧，并且在 STB 使能发送前， 主控制器请求 PTB 发送。

如果主控制器等待直到每个命令传输完成，那么它可以很容易地决定哪个缓冲区将传输下一帧。

表 29-3 发送缓冲区寄存器 TBUF – 标准格式（rw-u）

地址	位								功能
	7	6	5	4	3	2	1	0	
TBUF	ID(7:0)								标识符
TBUF+1	-					ID(10:8)			标识符
TBUF+2	-								标识符
TBUF+3	TTSEN	-							标识符
TBUF+4	IDE=0	RTR	EDL (FDF)	BRS	DLC(3:0)				控制位
TBUF+8	d1(7:0)								数据字节 1

TBUF+9	d2(7:0)	数据字节 2
.	…	
TBUF+71	d64(7:0)	数据字节 64

表 29-4 发送缓冲区寄存器 TBUF – 扩展格式（rw-u）

地址	位								功能
	7	6	5	4	3	2	1	0	
TBUF	ID(7:0)								标识符
TBUF+1	ID(15:8)								标识符
TBUF+2	ID(23:16)								标识符
TBUF+3	TTSEN	-		ID(28:24)					标识符
TBUF+4	IDE=1	RTR	EDL (FDF)	BRS	DLC(3:0)				控制位
TBUF+5	KOER			TX	-				状态位
TBUF+8	d1(7:0)								数据字节 1
TBUF+9	d2(7:0)								数据字节 2
.	…								
TBUF+71	d64(7:0)								数据字节 64

如果 TBSEL=1，TBUF 寄存器（0x50 至 0x97）指向 STB 中的下一个空消息缓冲区，否则指向 PTB。所有 TBUF 寄存器可以以任意顺序写入。对于 STB，需要设置 TSNEXT 以标记填充的缓冲区并跳转到下一个消息缓冲区。

请注意 TBUF 的寻址范围内 TBUF+5 和 TBUF + 7 的间隔，这样做为了更好的地址段对齐。可以读取和写入间隙中的存储单元，但对 CAN 协议没有意义。

TBUF 使用真正的 32 位宽存储器构建，因此写访问需要以 32 位写入的方式执行。

RBUF 和 TBUF 都包含一些独立帧控制位。对于 RBUF，这些位表示接收到的 CAN 帧相应 CAN 控制字段位的状态，而对于 TBUF，这些位为必须发送的帧选择适当的 CAN 控制字段位。

与存储每个接收帧的 RTS 相比，TTS 仅存储最后发送的帧。TTS 与实际选择的 TBUF 缓冲区无关。

表 29-5 RBUF 和 TBUF 的控制位

名称	说明
IDE	标识符扩展 0：标准帧：ID(10:0) 1：扩展帧：ID(28:0)
RTR	远程传输请求 0：数据帧 1：远程帧

	只有 CAN 2.0 帧可以是远程帧。CAN FD 没有远程帧。因此如果 TBUF 和 RBUF 中的 FDF=1，则 RRS 位(对应的 CAN2.0 帧的 RTR 位)强制为 0 当 FDF=1 时即使 RRS = 1，接收节点仍可以正确接收到 CAN FD 帧
EDL	扩展数据长度 0：CAN 2.0 帧(每帧最多 8 个字节) 1：CAN FD 帧(每帧最多 64 个)
FDF	FD 格式标示（=EDL） 0：CAN 2.0 格式 1：CAN FD 格式
BRS	位速率切换 0：整个帧为正常/低速比特率 1：切换到数据/快速比特率，因此如果 FDF=0，BRS 被强制转变为 0
ESI	错误状态指示器 这是 RBUF 的只读状态位，在 TBUF 中不可用 CAN 硬件会自动将正确的 ESI 值嵌入传输的帧中。ESI 仅包含在 CAN FD 框架中，不存在于 CAN 2.0 框架中 0：CAN 总线其它节点没有被动错误 1：CAN 总线其它节点有被动错误 对于 CAN 2.0 帧，RBUF 中的 ESI 位始终保持低 传输错误状态通过寄存器 ERRINT 中的 EPASS 位显示
TTSEN	发送时间戳启用 在 TBUF 中，对于 CiA 603 时间戳 TTS 可选择使能更新 0：不获取该帧的发送时间戳(TTS 值无效) 1：使能 TTS 更新

RBUF 和 TBUF 中的数据长度代码（DLC）定义了有效载荷的长度 – 帧中有效载荷字节的数量。

远程帧（仅用于 EDL(FDF) = 0 的 CAN 2.0 帧）总是以 0 个有效载荷字节发送，但 DLC 的内容在帧头中发送。因此，可以将一些信息编码到远程帧的 DLC 位中。但是，如果允许不同的 CAN 节点发送具有相同 ID 的远程帧，则需要注意。在这种情况下，所有发送器都需要使用相同的 DLC，否则会导致无法解决的冲突。

表 29-6 DLC 定义

DLC(二进制)	帧类型	有效字节数
0000 到 1000	CAN 2.0 和 CAN FD	0 到 8
1001 到 1111	CAN FD	~~8~~12~64
1001	CAN FD	12
1010	CAN FD	16
1011	CAN FD	20
1100	CAN FD	24
1101	CAN FD	32

1110	CAN FD	48
1111	CAN FD	64

TBUF 寄存器可读写。 因此如果有需要时主控制器可以使用 TBUF 依次逐位地准备消息。

29.3.2 总线关闭状态

CAN_CTRL0 寄存器中的状态位 BUSOFF 用来标识“总线关闭”状态。如果 CAN 节点的发送错误计数器计数超过 255，则该节点自动进入“bus off”状态。然后，在再次进入主动错误激活状态之前，它不会参与进一步的通信。如果使能 EIE，则将 BUSOFF 置 1 也会设置 EIF 中断。如果 CAN 节点通过模块(SRST_CAN)系统复位复位或者接收到 128 组连续 11 个隐性位（恢复序列）后，CAN 节点返回到主动错误状态。

在“bus off”状态下，RECNT 用于计数恢复序列，而 TECNT 保持不变。请注意，在进入“bus off”状态时，TECNT 会回滚，因此可能保持为一个较小的值。因此，建议在节点进入“bus off”状态之前使用 TECNT。

如果节点从“bus off”恢复，则 RECNT 和 TECNT 将自动重置为 0。

如果一个帧正在等待传输，但 CAN 节点已进入总线断开状态，则该帧保持挂起状态。如果一个节点在总线关闭后返回主动错误状态，那么将尝试传输挂起的帧。如果不需要，则主机控制器应中止帧。

29.3.3 接收过滤器

为了减少主控制器接收帧的负载，内核使用接收过滤器。因此，每个接收过滤器的长度是 29 位。

如果消息通过其中一个过滤器，则该消息会被接受且将其存储到接收缓存区(RB)中，如果用户使能接收中断使能寄存器 RIE，则此时 RIF 会被置位。如果该消息未被接受，则 RIF 不会被置位且 RB FIFO 指针不会增加。未被接受的消息将被丢弃并被下一条消息覆盖。任何未被接受的消息都不会覆盖已存储且有效的消息。

独立于接收过滤的结果，CAN-CTRL 内核检查总线上的每条消息，并向总线发送确认或错误帧。

接受掩码定义了要进行比较的位，而接收代码定义了适当的值。将掩码位置为 0 可以将所选择的接受码位与相应的消息标识符位进行比较。设置为 1 的掩码位被禁用以进行接收检查，这会导致接受该消息。

标识符位将与相应的接受代码位 ACODE 进行比较，如下所示：

- 标准：ID(10：0)和 ACODE(10：0)
- 扩展：ID(28：0)和 ACODE(28：0)

例如：如果 AMASK_x(0)=0 且所有其他 AMASK_x 位都为 1，那么最后一个 ID 位的值必须等于接受消息的 ACODE_x(0)的值。所有其他 ID 位会被过滤器忽略。

注意：通过设置 ACFEN_x=0 来禁用过滤器会阻止消息。与此相反，禁用 AMASK_x 中的

掩码位会禁用对此位的检查，这会导致接受消息。

仅 AMASK 和 ACODE 的定义不区分标准帧或扩展帧。如果位 AIDEE = 1，则接受 AIDE 定义的帧类型值。否则，如果 AIDE = 0，则两种类型都会被接受。

上电复位后，配置 CAN-CTRL 内核接收所有消息(设置 ACFEN_0=1 使能 Filter 0，AMASK_0 的所有位都被置为 1 同时 AIDEE 置为 0。所有其他过滤器都被禁用。Filter 0 是唯一为 AMASK/ACODE 定义复位值的过滤器，而所有其他过滤器都有未定义的复位值)。

29.3.4 消息接收

接收数据被存储在 RB 中，如图 29-3 所示。RB 可由预合成参数配置，并具有类似 FIFO 的行为。如果启用了 RIE，则每个收到有效、可被接受的消息都会设置 RIF=1。根据填充状态设置 RSTAT。当填充缓冲区的数量等于可编程值 AFWL 时，如果 RAFIE 使能，则设置 RAFIF。如果所有缓冲区都已满，则当使能 RFIE 时，置位 RFIF。

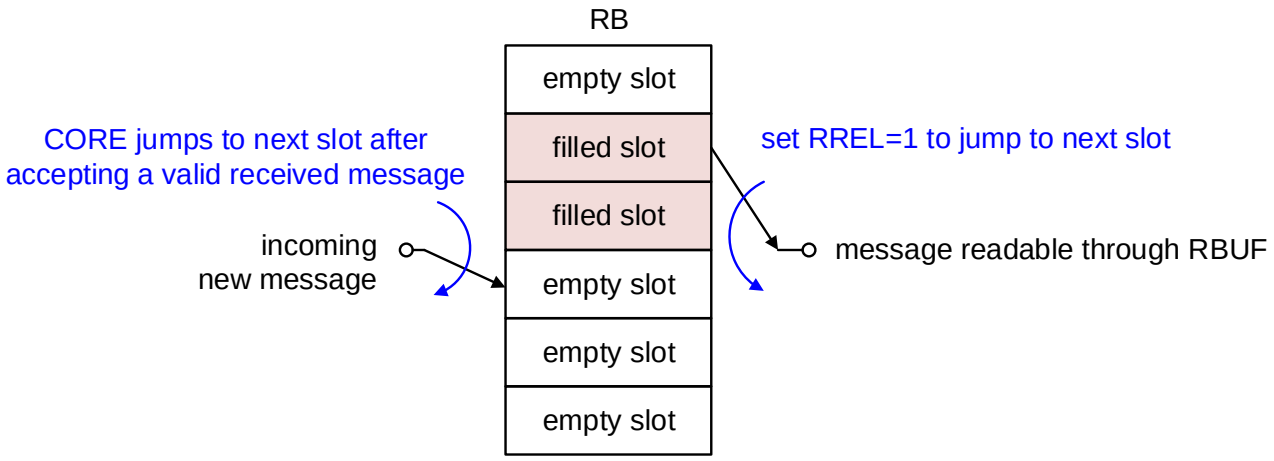


图 29-2 类似 FIFO RB 示意图(6 slots 示例)

RB 始终将包含最旧消息的消息 slot 映射到 RBUF 寄存器。CAN 2.0 消息的最大有效载荷长度为 8。每条消息的长度由 DLC 定义。因此，RB 为每条消息提供 slot，并且主控制器需要设置 RREL 以跳转到下一个 RB slot。可以按任意顺序读取实际 slot 的所有 RBUF 字节。

如果 RB 已满，则下一个传入的消息将被临时存储，直到它有效地传递（第 6 个 EOF 位）。如果 ROM 为 0，最旧的消息将被最新的消息所覆盖，或者如 ROM =1，则最新消息将被丢弃。在这两种情况下，如果使能 ROIE，则 ROIF 会被置位。如果主控制器读取最旧的消息并在新传入的消息变为有效之前设置 RREL，则不会丢失任何消息。

29.3.5 处理消息接收

如果没有验收过滤器，CAN-CTRL 内核会发出每帧的接收信号，并且主机会被要求决定它是否进行寻址，这会给主控制器带来很大的负担。

可以禁用中断并使用接受过滤器来减少主控制器的负载。一个基本操作为：如果 RIE 已启用且 CANCTRL 内核已收到有效消息，则 RIF 设置为 1。为了减少接收中断的数量，可以使用 RAFIE/RAFIF（RB 几乎满中断）或 RFIE/RFIF（RB 满中断）而不是 RIE/RIF（接收中断）。“几乎满限制”可使用 AFWL 进行编程。

RB 包含多个 RB slot，可在使用通用参数进行合成之前进行选择。读取 RB 应按如下方式进行：

2 使用 RBUF 寄存器从 RB FIFO 中读取最旧的消息；

3 用 RREL=1 释放 RB slot。这将选择下一条消息（下一个 FIFO slot），RBUF 会被自动更新；

4 重复以上动作直到 RSTAT 发出空 RB 信号。

如果 RB FIFO 已满并且新接收的消息被识别为有效（第 6 个 EOF 位），则将丢失一条消息（参见 ROM 位）。在此之前，不会丢失任何消息。应该给出足够的时间让主控制器在 RB FIFO 已满并且发生所选中断之后从 RB 读取至少一帧。为了实现这种行为，RB 包括一个（隐藏）slot。此隐藏 slot 用于接收消息，验证该消息并在溢出发生之前检查它是否与验证过滤器相匹配。

29.3.6 消息发送

开始发送之前，必须向至少一个发送缓冲区（PTB 或 STB）加载消息。如果 PTB 被锁且 TPSTAT 发信号通知 STB 的填充状态，则 TPE 会发出信号。TBUF 寄存器提供对 PTB 和 STB 的访问。推荐的编程流程如下：

1 将 TBSEL 设置为需要的值以选择 PTB 或 STB；

2 将帧写入 TBUF 寄存器；

3 对于 STB，设置 TSNEXT=1 以完成该 STBslot 的加载。

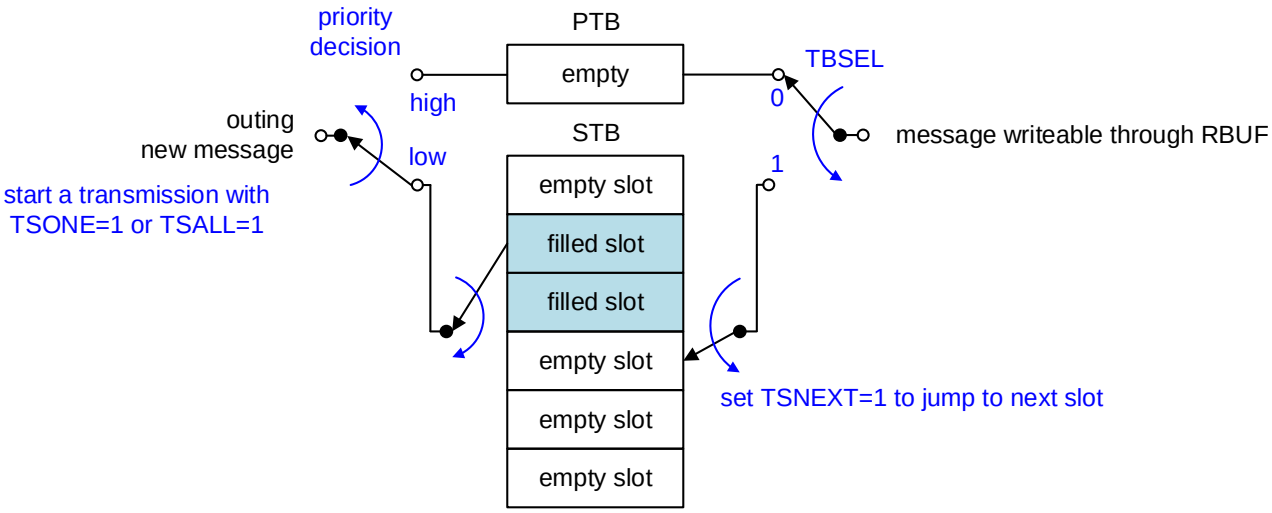


图 29-3 FIFO 模式下 PTB 及 STB 示意图（空 PTB，6 STB slots）

CAN 2.0 消息的最大有效载荷长度为 8 个字节，CAN FD 的消息最大为 64 个字节。每条

消息的长度由 DLC 定义。对于远程帧（比特 RTR），DLC 变得毫无意义，因为远程帧的数据长度始终为 0 字节。主控制器需要设置 TSNEXT 以跳转到下一个 STB slot。所有 TBUF 字节都可以按任意顺序写入。

如果 TBSEL = 0 时选择 PTB，则设置 TSNEXT = 1 是毫无意义的。在这种情况下，TSNEXT 会被自动清除并且不会有任何影响。

应将 TPE 位设置为在使用 PTB 时启动发送。要使用 STB，必须将 TSONE 设置为开始发送单个消息或 TSALL 发送所有消息。

PTB 的优先级始终比 STB 高。如果 PTB 和 STB 同时使能发送，则无论帧标识符如何，都将始终先发送 PTB 消息。如果来自 STB 的发送已经激活，则在来自 PTB 的消息在下一个可能的发送位置（下一个帧 slot）发送之前完成。在 PTB 发送完成或中止之后，CAN-CTRL 内核返回以处理来自 STB 其他暂停的消息。

当发送完成后，会设置如下发送中断：

- 对于 PTB，如果启用了 TPIE，则将设置 TPIF；
- 对于使用 TSONE 的 STB，如果一条消息已完成且已使能 TSIE，则会置位 TSIF；
- 对于使用 TSALL 的 STB，如果所有消息已完成发送且已使能 TSIE，则会置位 TSIF。换言之，如果 STB 为空，则 TSIE 会被置位。

当 STB 为空时，设置 TSONE 或 TSALL 是毫无意义的。在这种情形下，TSONE 和 TSALL 都将会自动复位。不会设置中断标志，也不会发送帧。

29.3.7 消息发送中止

由于其低优先级而无法发送缓冲区中的消息，这将长时间阻塞缓冲区。为了避免这种情况，如果尚未开始发送，则主控制器可以通过分别设置 TPA 或 TSA 来撤销发送请求。TPA 和 TSA 都提供单个中断标志：AIF。CAN 协议机器只有在不向 CAN 总线发送任何内容时才执行中止。因此，以下行为是有效的：

- 总线仲裁期间没有中止
  - 如果节点失去仲裁，则在此之后将执行中止
  - 如果节点赢得仲裁，则发送该帧
- 发送帧时没有中止
  - 如果成功发送帧，则通知主控制器发送成功。在这种情况下，不会发出中止信号。这是通过适当的中断和状态位完成的
  - 在 CAN 节点未接收到确认的发送失败之后，错误计数器递增并且将执行中止
  - 当主机使能发送所有的帧时（TSALL = 1），如果 STB 中剩余至少一帧，则完成发送的帧以及中止的帧都会通知主机

基于以上事实，中止此发送可能需要一些时间，具体取决于 CAN 通信速度和帧长度。如果执行中止，则会导致以下操作：

- TPA 释放 PTB，导致 TPE=0，在释放 PTB 之后，帧数据仍存储在 PTB 中。
- TSA 释放 STB 的单个或所有消息 slot，这取决于是使用 TSONE 还是 TSALL 来启动

发送。TSSTAT 会相应地进行更新。释放 STB 中的帧会导致丢弃帧数据，因为主机无法访问它。

不建议同时设置 TPA 和 TSA。如果主控制器仍然决定这样做，则将设置 AIF，并且如果可能的话，将中止来自 PTB 和 STB 的所有发送。如前所述，如果在可以执行中止之前完成一次发送，则这将导致信号发送成功。因此，如果使能，可以设置以下中断标志：

- AIF(设置一次，用于 PTB 和 STB 发送中止)
- TPIF + AIF
- TSIF + AIF
- TPIF + TSIF(很少，只有在主机没有立即处理 TPIF 时才会发生)
- TPIF + TSIF + AIF(很少，只有在主机没有立即处理 TPIF 和 TSIF 时才会发生)

要清除整个 STB，需要设置 TSALL 和 TSA。为了检测由于失去仲裁而无法长时间发送消息，主机可以使用 ALIF/ALIE。

29.3.8 STB 满

在向 STB 写入一条消息之后，TSNEXT=1 标记填充的缓冲 slot 并跳转到下一个空闲消息 slot。此操作后，CAN 内核会自动将 TSNEXT 复位为 0。如果最后一个消息 slot 已被填满，因此所有消息 slot 都被占用，则 TSNEXT 保持置位状态，直到新的消息 slot 空闲为止。当 TSNEXT=1 时，CAN 内核会阻止写至 TBUF。

当 slot 空闲时，CAN 内核会自动将 TSNEXT 重置为 0。如果来自 STB 的帧成功发送或者主机请求中止(TSA=1)，则 slot 会变为空闲。如果 TSALL 发送中止，则 TSNEXT 也会复位，但整个 STB 会被标记为空。

29.3.9 扩展状态及错误报告

在 CAN 总线通信时，会发生发送错误。如下特性支持检测和分析发送错误。

29.3.9.1 可编程错误警告限制

接收/发送期间的错误由 RECNT 和 TECNT 来计数。LIMIT 寄存器中的可编程错误警告限制 EWL 可由主控制器灵活配置以用于响应接收/发送错误事件。可以从 8 到 128 的 8 个错误中选择极限值：错误计数限制 = (EWL + 1) * 8.

如果在以下条件下由 EIE 使能，则将设置中断 EIF：

- 错误警告限制的边界已通过 RECENT 或 RECENT 在任一方向交叉
- BUSOFF 位已经在某一方向上变化

29.3.9.2 冲裁失利捕获

控制器能够检测仲裁段中仲裁失利的确切位位置。如果 ALIF 中断被启用，则可以通过 ALIF 中断发出此事件。如果此节点能够赢得此仲裁，则 ALC 的值会保持不变。ALC 保持

最后一次仲裁失利的值。

ALC 的值定义如下：一帧以 SOF 位开始，发送 ID 的第 1 位。第一个 ID 位的 ALC 值为 0，第二个 ID 位的 ALC 值为 1，依此类推。仲裁仅允许在仲裁域中进行。因此，ALC 的最大值为 31，是扩展帧的 RTR 位。

如果标准远程帧与扩展帧进行仲裁，则扩展帧会在 IDE 位失去仲裁，ALC 将为 12。发送标准远程帧的节点将不会注意到已经发生了的仲裁，因为该节点已经获胜。在仲裁域之外不可能获得仲裁失利，这样的事件为位错误。

29.3.9.3 错误类型

内核识别 CAN 总线上的错误，并将最后一个错误事件存储在 KOER 位中。如果使能 BEIF 中断，则可以通过 BEIF 中断发出 CAN 总线错误信号。每个新的错误事件都会覆盖之前存储的 KOER 值。因此，主控制器必须对错误事件做出快速反应。

29.3.9.4 接收所有的数据帧(RBALL)

如果 RBALL = 1，则所有接收到的数据帧都将被存储，即使有错误的数据帧也会被存储。这也适用于回环模式。仅数据帧存储在 RBUF 中，错误帧或过载帧不会被存储。

如果启用了 CiA 603 时间戳记（TIMEEN = 1）并且为 EOF 配置了时间戳记（TIMEPOS = 1），则在出现错误的情况下，将在错误帧的开始处获取时间戳。

仅当节点是帧的发送方时，大多数可能的错误才会发生。在这种情况下框架如果激活了回环模式，则仅将其存储在 RBUF 中。根据错误部分的类型，当其他部分未知时，存储在 RBUF 插槽中的帧可能有效。

29.3.10 扩展功能

29.3.10.1 单次发送

有时，不需要自动重传。因此，可以通过 TPSS 位对发送缓冲器 PTB、通过 TSSS 位对发送缓冲器 STB 分别设置发送一次消息的命令。在这种情况下，如果所选发送有效，则在发生错误或仲裁失利的情况下不会执行重传。

在立即发送成功的情况下，与正常发送没有区别。但是在发送失败的情况下， 会出现如下问题：

- 如果使能，则 TPIF 会被置位，相应的发送缓冲区 slot 会被清除
- 如果出现错误，会更新 KOER 和错误计数器。如果使能 BEIE，BEIF 会被置位，其他错误中断标志将相应地起作用
- 如果仲裁失利，当使能 ALIE 时，会置位 ALIF

因此，对于单次发送，TPIF 自身并不指示帧是否已成功发送。因此，单次发送应仅与 BEIF 和 ALIF 一起使用。

如果单次发送使用 TSALL，并且 STB 中存在多个帧，则对于每个帧，进行单次发送。不

管是否未成功发送任何帧（例如：由于应答错误），CAN-CTRL 内核前行至下一帧，并且如果 STB 为空则停止。

因此，在这种场景下，只有错误计数器指示发生了什么。这将难以评估，因为如果两个帧中的一个出现错误，则主机无法检测哪个是成功的帧。

如果总线被另一个帧占用，如果单次发送开始，则 CAN-CTRL 内核会一直等待直到总线空闲，然后尝试发送单次帧。

29.3.10.2 监听模式

LOM 提供监控 CAN 总线的能力，而不会对总线产生任何影响。

另一个应用是自动比特率检测，其中主控制器尝试不同的定时设置，直到没有错误发生。

在 LOM 中监视错误(KOER,BEIF)。

在 LOM 中，内核无法将显性位写入总线(没有有效的错误标志或过载标志，没有确认)。这是使用如下规则完成的

- 在 LOM 中，协议控制器就好像处于错误被动模式，其中仅生成隐性错误标志。只有协议机器就像处于错误被动模式一样。不触及包括状态寄存器在内的所有其他组件。
- 在 LOM 中，协议控制器不生成隐形应答。
- 不管什么错误，错误计数器保持不变。

关于 LOM 的应答：

- 如果帧由一个节点发送，则仅当至少一个不在 LOM 状态的附加节点连接到总线时，才会生成在总线上可见的应答。这样就不会出现错误，所有节点（也包括 LOM 中的节点）都会收到该帧。
- 如果在应答错误之后存在主动或被动错误标志，则 LOM 中的节点能够将其检测为应答错误。

在发送激活时，不应激活 LOM。主控制器必须注意这个问题，没有来自 CAN-CTRL 内核额外的保护措施。如果使能 LOM，则无法开始发送。

29.3.10.3 总线连接测试

要检测节点是否连接至总线，可以采用如下测试步骤：

- 发送一帧。如果节点连接到总线，则其 TX 位在其 RX 输入上是可见的；
- 如果有其他节点连接到 CAN 总线，则预期会发送成功(包括来自其他节点的确认)，不会发出任何错误信号；
- 如果该节点是唯一一个连接到 CAN 总线的节点(但总线、收发器和 CAN 内核之间的连接正常)。由于没有来自其他节点的确认信息，第一个常规错误发生在应答段中。如果使能且 KOER=“100”（应答错误），则会产生 BEIF 错误中断；
- 如果与收发器或总线间的连接断开，则在 SOF 位之后立即设置 BEIF 错误中断并且 KOER=“001”（BIT 错误）。

29.3.10.4 回环模式（LBMI 及 LBME）

CAN 内核支持两种回环模式：内部(LBMI)和外部(LBME)。两种模式都导致接收自己发送的帧，这对于自测是很有用的。

在 LBMI 中，CAN 内核与 CAN 总线断开连接，txd 输出设置为隐性。输出数据流在内部反馈到输入。在 LBMI 模式下，节点生成自应答信息以避免应答错误。

在 LBME 模式下，CAN 内核保持与收发器的连接，并且在总线上可以看到发送的帧。在收发器的帮助下，CAN 内核接收自己发送的帧。在 LBME 模式下，当 SACK=0 时节点不产生自应答信息。如果 SACK=1 时节点产生应答信息。因此，在 LBME 模式下，帧发送有两种可能的结果：

1. 另外一个节点也接收该帧，并产生确认信息。这会导致一次成功的发送和接收。
2. 没有其他节点连接到总线，这会导致应答错误。为了避免重传并增加错误计数，如果不知道其他节点是否连接到总线，建议使用 TPSS 或 TSSS。

在回环模式下，内核接收其自身的消息， 将其存储在 RBUF 中，并在使能时设置合适的发送和接收中断标志。

LBMI 可用于芯片内部和软件测试，而 LBME 可以测试收发器及其连接。

当发送处于活动状态时，不应激活两种回环模式，主控制器需要注意此问题。没有来自 CAN 内核额外的保护措施。

如果节点连接到 CAN 总线，则不能通过简单地将 LBMI 设置为 0 来完成从 LBMI 切换回正常操作，因为这可能只是在另一个 CAN 节点正在发送时发生的情况。在这种情况下，应将 RESET 位设置为 1 来切换回正常操作，这会自动将 LBMI 清零。最后，可以禁用 RESET 并且内核返回至正常操作。与此相反，LBME 每次都可以被禁用。

29.3.10.5 接收器待机模式

使用寄存器位 STBY，可以驱动输出待机信号。它可用于激活收发器的待机模式。

一旦待机模式被激活，无法进行发送，因此无法设置 TPE，TSONE 和 TSALL。另一方面，如果发送已经激活（设置了 TPE，TSONE 或 TSALL），CAN 内核不允许设置 STBY。

如果已经置位 STBY，收发器进入低功耗模式。在此模式下，无法以全速接收帧，但会监视 CAN 总线的显性状态。如果显性状态在收发器数据手册中定义的时间内有效，则收发器会将 rxd 信号拉低。如果 rxd 为低电平，CAN 内核会自动将 STBY 清零，从而禁用收发器的待机模式。这是在没有中断到主控制器的情况下完成的。

从待机模式切换到活动模式对收发器需要一些时间，因此无法成功接收初始唤醒帧。因此，最近处于待机状态的节点不会发送应答信号。如果总线上的 CAN 节点没有应答唤醒帧，这会导致唤醒帧的发送器应答错误。然后发送器将自动重复该帧。在此重复期间，收发器将返回活动模式，CAN 内核将接收此帧并将做应答。

总之，一个节点发送用于唤醒的帧。如果所有其他节点处于待机模式，发送器会收到应答错误并自动重复该帧。在重复的过程中，节点返回活动模式并将做应答。

STBY 不受 RESET 位的影响。

29.3.10.6 错误计数器复位

根据 CAN 标准，RECNT 对接收错误进行计数，TECNT 对发送错误进行计数。若发送错误太多，CAN 节点必须进入总线关闭状态，这会激活 RESET 位。取消激活该位后，RESET 不会修改错误计数器或总线关闭状态。CAN 规范定义了如何禁用总线关闭状态和减少错误计数的规则。如果只有一个临时错误导致该问题，一个好的节点将自动从所有这些问题中恢复。经典的 CAN 2.0B 规范要求这种自动行为，无需主控制器交互。

将 BUSOFF 位置 1 会复位错误计数器，从而强制节点退出总线关闭状态。这是在不设置 EIF 的情况下完成的。

29.3.11 软件复位

如果 CAN_CTRL0 寄存器中的 RESET 位被置为 1，则软件复位处于激活状态。如果 RESET = 1，则几个组件被强制置为复位状态，但 RESET 不会触及所有组件。一些组件仅对硬件复位敏感。软件和硬件复位的所有位的复位值始终相同。

表 29-7 软件复位

寄存器	复位	说明
ACFADR	否	-
ACODE_x	否	如果 RESET=1 则寄存器可写，为 0 则寄存器写锁定
ACFEN_x	否	-
AFWL	否	-
AIF	是	-
ALC	是	-
ALIE	否	-
ALIF	是	-
AMASK_x	否	如果 RESET=1 则寄存器可写，为 0 则寄存器写锁定
BEIE	否	-
BEIF	是	-
BUSOFF	否	通过设置 BUSOFF=1 可以复位 BUSOFF 及错误计数器
EIE	否	-
EIF	否	-
EPASS	否	-
EPIE	否	-
EPIF	是	-
EWARN	否	-
EWL	是	-
FD_ISO	否	如果 RESET=1 则寄存器可写，为 0 则寄存器写锁定
F_PRESC	否	如果 RESET=1 则寄存器可写，为 0 则寄存器写锁定
F_SEG_1	否	如果 RESET=1 则寄存器可写，为 0 则寄存器写锁定
F_SEG_2	否	如果 RESET=1 则寄存器可写，为 0 则寄存器写锁定
F_SJW	否	如果 RESET=1 则寄存器可写，为 0 则寄存器写锁定

KOER	是	-
LBME	是	-
LBMI	是	-
LOM	否	-
RACTIVE	是	即使接收处于活动状态，接收也会立即取消，不会生成 ACK 信息
RAFIE	否	-
RAFIF	是	-
RBALL	是	-
RBUF	是	-
RECNT	否	-
RFIE	否	-
RFIF	是	-
RIE	否	-
RIF	是	-
ROIE	否	-
ROIF	是	-
ROM	否	-
ROV	是	所有 RB slot 被标记为空
RREL	是	-
RSTAT	是	所有 RB slot 被标记为空
SACK	是	-
SELMASK	否	-
STBY	否	-
S_PRESC	否	如果 RESET=1 则寄存器可写，为 0 则寄存器写锁定
S_Seg_1	否	如果 RESET=1 则寄存器可写，为 0 则寄存器写锁定
S_Seg_2	否	如果 RESET=1 则寄存器可写，为 0 则寄存器写锁定
S_SJW	否	如果 RESET=1 则寄存器可写，为 0 则寄存器写锁定
TACTIV	是	在 RESET 时，所有发送都立即停止。如果发送有效，则会产生不完整的帧。其他节点会产生错误帧
TBSEL	是	TBUF 固定指向 PTB
TBUF	是	所有 STB slot 被标记为空，因为 TBSEL TBUF 指向 PTB
TECNT	否	-
TIMEEN	否	-
TIMEPOS	否	-
TPA	是	-
TPE	是	-
TSA	是	-
TSALL	是	-
TSMODE	否	-
TSNEXT	是	-
TSONE	是	-
TPIE	否	-
TPIF	是	-
TPSS	是	-
TSFF	是	所有 STB slot 被标记为空

TSIE	否	-
TSIF	是	-
TSSS	是	-
TSSTAT	是	所有 STB slot 被标记为空
TTS	否	-

29.3.12 CAN 位时间

CAN 2.0B 定义了高达 1Mbit/s 的数据比特率。对于 CAN FD 没有固定的限制，对于实际的系统，数据速率受所使用的收发器和 CAN-CTRL 内核可实现的时钟频率的限制，这取决于所使用的目标单元库。

CAN 内核可以编程为任意选择的数据速率，仅受相应位定时和预分频器寄存器中位设置范围的限制。

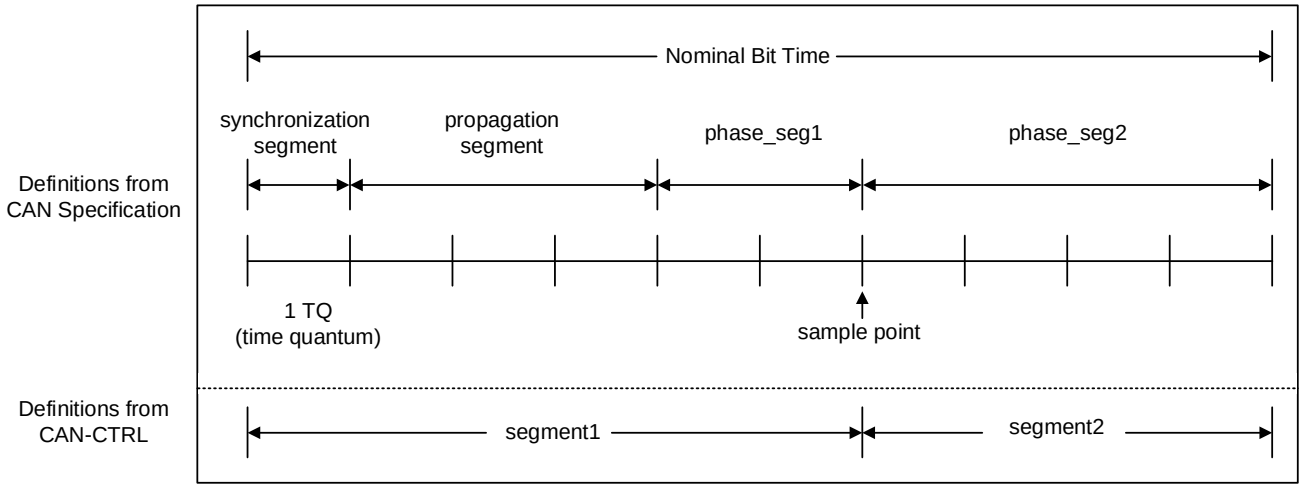


图 29-4 CAN 位时间定义

CAN 位定时 BT 由若干段（segment）组成，如图 29-4 所示。每个段由许多时间限额单位 nTQ 组成。时间限额的时长 TQ 为：

TQ = nprescaler / fCLOCK

BT = nprescaler * nTQ / fCLOCK = tSeg_1 + tSegt_2

CAN 规范要求段长度之间存在若干关系（如

所示），这会造成 tSeg_1，tSegt_2 和最大同步跳转宽度 tSJW 之间的关系。

表 29-8 CAN 时序段定义

段	说明
SYNC_SEG	同步段 = 1 TQ
PROP_SEG	传播段（Propagation Segment） [1...8] TQ CAN 2.0 比特率 CAN FD 未启用 [1...48] TQ CAN 2.0 比特率 CAN FD 已启用 [1...48] TQ CAN FD 标称比特率 [0...8] TQ CAN FD 数据比特率
PHASE_SEG1	相位缓冲段 1 [1...8] TQ CAN 2.0 比特率 CAN FD 未启用 [1...16] TQ CAN 2.0 比特率 CAN FD 已启用

	[1...16] TQ CAN FD 标称比特率 [1...8] TQ CAN FD 数据比特率
PHASE_SEG2	相位缓冲段 2 [2...8] TQ CAN 2.0 比特率 CAN FD 未启用 [2...16] TQ CAN 2.0 比特率 CAN FD 已启用 [2...16] TQ CAN FD 标称比特率 [2...8] TQ CAN FD 数据比特率
SJW	同步跳转宽度 [1...4] TQ CAN 2.0 比特率 CAN FD 未启 用 [1...16] TQ CAN 2.0 比特率 CAN FD 已启 用 [1...16] TQ CAN FD 标称比特率 [1...8] TQ CAN FD 数据比特率
IPT	信息处理时间 = [0…2] TQ PHASE_SEG2 ≥ IPT

如

所示，CAN 内核将 SYNC_SEG，PROP_SEG 和 PHASE_SEG1 等集为一组，该组的长度可使用 tSeg_1 进行配置。

列出了可用的配置范围。请注意，CAN 内核不会检查是否满足所有规则，并提供比 CAN/CAN FD 规范定义的更宽的配置范围。

表 29-9 CAN 内核时序配置

配置项	需求		
t_{Seg_1}	[2…65] [2…65] [2…17]		CAN 2.0 比特率(慢) CAN FD 标称比特率(慢) CAN FD 数据比特率(快)
t_{Seg_2}	[1…8] TQ [1…32] TQ [1…8] TQ	$t_{Seg\_1} \geq t_{Seg\_2} + 2$ $t_{Seg\_1} \geq t_{Seg\_2} + 2$ $t_{Seg\_1} \geq t_{Seg\_2} + 1$	CAN 2.0 比特率(慢) CAN FD 标称比特率(慢) CAN FD 数据比特率(快)
t_{SJW}	[1…16] TQ [1…16] TQ [1…8] TQ	$t_{Seg\_2} \geq t_{SJW}$ $t_{Seg\_2} \geq t_{SJW}$ $t_{Seg\_2} \geq t_{SJW}$	CAN 2.0 比特率(慢) CAN 2.0 标称比特率(慢) CAN 2.0 数据比特率(快)

对于 CAN 2.0 比特率标称的比特率以及 CAN FD(慢)标称比特率，配置寄存器 S_Seg_1，S_Seg_2，S_SJW 和 S_PRESC 来定义适当的段长度。寄存器 S_Seg_1，S_Seg_2，S_SJW 和 S_PRESC 对于 CAN FD(快)数据比特率是无效的。

$T_{Seg\_1} = (S\_Seg\_1 + 2) * TQ$　　$t_{Seg\_1} = (F\_Seg\_1 + 2) * TQ$

$t_{Seg\_2} = (S\_Seg\_2 + 1) * TQ$　　$t_{Seg\_2} = (F\_Seg\_2 + 1) * TQ$

$t_{SJW} = (S\_SJW + 1) * TQ$　　$t_{SJW} = (F\_SJW + 1) * TQ$

$n_{prescaler} = S\_PRESC + 1$　　$n_{prescaler} = F\_PRESC + 1$

CAN FD 内核通过设置 BRS=1 就可以从低速标称比特率切换到快速数据比特率，并在 CRC 分隔符位的采样点切回。

对于 CAN FD 节点，由于收发器的延迟导致 CAN FD 的通信可能会延迟超过一个 bit 时间，因此，无法使用原始 SP（Sample point）对正确的位值进行采样，CAN FD 规范定义了一个额外的二次采样点(SSP)，此时可以选择启用发射机延迟补偿（TDC）即

TDCEN=1 且配置 SSPOFF 寄存器，建议 SSPOFF 配置等于 tSeg_1，用户需要根据实际情况配置该寄存器。 如果不启用 TDC 发送节点可能就不能正确采样到它所发出的数据，此时发送节点就会检测到一个位错误。

初始化 CAN 内核的步骤如下：

1 设置位 RESET=1；

2 设置寄存器 S_Seg_1 and S_Seg_2；

在该示例中，总线数据速率为 1M 波特率，系统时钟为 48MHz。

所选 n_{TQ} 和 $n_{prescaler}$ 的值需适配 BT。

在该示例中，所选 nprescaler = 2，nTQ = 24，这样可以实现完全匹配：BT = 24TQ。

在时间段定义中，可以选择 t_{Seg_1} = 18TQ; t_{Seg_2} = 6TQ，对应寄存器：S_Seg_1=16，S_Seg_2 = 5；

3 加载验证码和掩码寄存器(可选)；

4 设置 S_SJW 寄存器；

当满足 $t_{Seg\_2} \geq t_{SJW}$，可以自由选择 t_{SJW} = 4，对应寄存器 S_SJW=3。

加载时钟预分频寄存器 S_PRESC：$n_{prescaler}$ = PRESC + 1，对应 S_PRESC=1；

5 设置位 RESET=0；

如上给出的顺序不是强制的。只需要在开始时设置 RESET=1，否则无法加载位定时，ACODE 和 AMASK 寄存器。配置完成后，需要将 RESET=0。然后控制器等待 8 个隐形位(帧结束)，然后恢复其正常操作。

6 继续配置中断其他配置位，并执行指令。

下面是一些适用于 CAN 网络所有节点位定时设置的示例表。

表 29-10 48MHz can_clk 的示例配置

比特率[Mbit/s]	SP[%]	预分频器	位时间[TQ]	Seg1[TQ]	Seg2[TQ]	SJW[TQ]
1	75	1	48	36	12	12
0.8	75	2	30	24	6	6
0.5	75	2	48	36	12	12
0.25	75	4	48	36	12	12
0.125	75	8	48	36	12	12
0.1	75	10	48	36	12	12

表 29-11 8MHz can_clk 的示例配置

比特率[Mbit/s]	SP[%]	预分频器	位时间[TQ]	Seg1[TQ]	Seg2[TQ]	SJW[TQ]
1	75	1	8	6	2	2
0.8	80	1	10	8	2	2
0.5	75	1	16	12	4	4
0.25	75	2	16	12	4	4
0.125	75	4	16	12	4	4
0.1	75	4	20	15	5	4

表 29-12 48MHz can fd clk 的示例配置

比特率[Mbit/s]	SP[%]	预分频器	位时间[TQ]	Seg1[TQ]	Seg2[TQ]	SJW[TQ]
8	83	1	6	5	1	1
6	75	1	8	6	2	2
4	75	1	12	9	3	3
2	75	1	24	18	6	6
1	75	2	24	18	6	6
0.1	75	20	24	18	6	6

0.05	75	40	24	18	6	6

29.3.13 时间戳

CAN 内核中的时间戳包括发送帧时间戳和接收帧时间戳。当接收数据或者发送数据时，CAN 内核复制计时器值帧时间戳存储在 RTS 和 TTS 中，用户可以读取寄存器 RTS 或者 TTS 获取时间戳值。

CAN 内核可以在有效帧的 SOF 或 EOF 位的采样点获取时间戳。可以通过配置位 TIMEPOS 进行选择采样点位置。ACK 界定符之后的七个隐性位形成 CAN/CAN FD 帧的 EOF。在许多系统中广泛使用的基于软件的时间戳依赖于接收和传输中断。因此，建议时间戳采样点配置在 EOF。

CAN 内核仅支持一个传输帧（TTS）的时间戳，但是对接收帧（RTS）有单独的时间戳。生成传输帧的时间戳可以使用 TBUF 插槽内的 TTSEN 位分别为每个帧启用或禁用。

CAN 内核内部包含计时器，时间戳机制，存储 TTS 的寄存器以及每帧存储 RTS 的内存。寄存器 TIMEEN 位启用或禁用时间戳。如果禁用，则 TTS 和 RTS 无效。

下面举例说明 CAN 内核发送时间戳功能使用步骤：

1. 初始化 CAN 内核，使能发送中断，配置其时钟源为 48MHz；
2. 设置 CAN 内核 timer clock 时钟分频 48，则 TTS 中每个计数值就是 1us；
3. 通过配置 TIMEPOS 寄存器设置采样点位置为 EOF；
4. 使能 TIMEEN 寄存器和 TTSEN 寄存器；
5. 在 CAN 内核发送完成后读取 TTS 值；

下面举例说明 CAN 内核接收时间戳功能使用步骤：

1. 初始化 CAN 内核，使能接收中断，配置其时钟源为 48MHz；
2. 设置 CAN 内核 timer clock 时钟分频 48，则 RTS 每个计数值就是 1us；
3. 通过配置 TIMEPOS 寄存器设置采样点位置为 EOF；
4. 使能 TIMEEN 寄存器；
5. 在 CAN 内核接收完成后读取 RTS 值。

29.4 CAN FD 寄存器列表

基地址：0x40005400

表 29-13 寄存器列表

偏移地址	名称	描述	复位值
0x00 ~ 0x4C	CAN_RBUF	接收缓冲区寄存器和接收时间戳，位宽 32bit×20 具体描述见表 29-1 及表 29-2	0x00000000
0x50 ~ 0x94	CAN_TBUF	发送缓冲区寄存器，位宽 32bit×18 具体描述见表 29-3 及表 29-4	0x00000000
0x98	CAN_TTS0	发送数据帧时间戳 0	0x00000000
0x9C	-	保留	-
0xA0	CAN_CTRL0	CAN 控制寄存器 0	0x00900080
0xA4	CAN_CTRL1	CAN 控制寄存器 1	0x1B0000FE
0xA8	CAN_SBITRATE	低速 CAN 通信波特率配置寄存器	0x01020203
0xAC	CAN_FBITRATE	高速 CAN 通信波特率配置寄存器	0x01020203
0xB0	CAN_ERRINFO	CAN 错误类型及错误计数寄存器	0x00000000
0xB4	CAN_ACFCTRL	接收过滤器控制寄存器	0x00010200
0xB8	CAN_ACF	接收代码 ACODE 寄存器	0x00000000
0xB8	CAN_ACF	接收掩码 AMASK 寄存器	0x00000000

29.5 CAN FD 寄存器说明

29.5.1 发送数据帧时间戳 0(CAN_TTS0)

偏移地址：0x98
复位值：0x00000000

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

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

位	标记	功能描述	复位值	读写
31:0	TTS	传输时间戳 TTS 保留最后发送帧的时间戳，用于 CiA603 时间戳。 如果 TTSEN=1，每个新的帧将覆盖 TTS，时间戳为 32 位宽	0x0	RO

29.5.2 CAN 控制寄存器 0(CAN_CTRL0)

偏移地址：0xA0
复位值：0x00900080

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SACK	ROM	ROV	RREL	RBAL	保留	RSTAT		FDISO	TSNEXT	TSMODE	保留			TSSTAT	
R/W	R/W	R/W	R/W	R/W		RO		R/W	R/W	R/W				RO	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBSEL	LOM	STBY	TPE	TPA	TSONE	TSALL	TSA	RESET	LBME	LBMI	TPSS	TSSS	RACTIVE	TACTIVE	BUSSOFF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO	RO	R/W

位	标记	功能描述	复位值	读写
31	SACK	自应答 0：没有自应答 1：当 LBME=1 有自应答	0x0	R/W
30	ROM	接收缓冲器溢出模式 如果在收到新消息时出现完整的 RBUF，则 ROM 选择以下内容： 0：最旧的消息将被覆盖 1：新消息将不会被存储	0x0	R/W
29	ROV	接收缓冲区溢出 0：不溢出 1：溢出，至少一个消息丢失 ROV 由设置 RREL=1 清除	0x0	R/W
28	RREL	接收缓冲区释放 0：不释放 1：释放，主机已经读取 RB 缓冲区 主控制器已经读取实际的 RB 缓冲区并将其释放，然后 CAN-CTRL 内核指向下一个 RB 缓冲区，RSTAT 得以更新。	0x0	R/W
27	RBALL	RB 缓冲区存储所有的数制帧	0x0	R/W

		0：正常操作 1：RB 缓冲区存储所有接收的数据帧，即使数据帧有误也会被存储，该位设置设也适用于回环模式，仅仅存储数据帧而不存储错误帧和过载帧。		
26	Res	保留位	0x0	-
25:24	RSTAT	接收缓冲区状态 00：空 01：大于空，小于几乎满（AFWL） 10：几乎满（AFWL 可编程阈值），但不满不溢出 11：满（在溢出的情况下保持设置 – 对于溢出信号，请参见 ROV）	0x0	RO
23	FDISO	CAN FD ISO 模式 0：Bosch CAN FD(non-ISO)模式 1：ISO CAN FD 模式(ISO 11898-1:2015) ISO CAN FD 模式有一个不同的 CRC 初始值以及一个额外的填位 在同一网络中不能同时混合以上两种模式 该位对 CAN2.0B 没有影响 如果 RESET=1，该位仅可写	0x1	R/W
22	TSNEXT	选择下一个次缓冲区 0：没有动作 1：填充 STB 缓冲区，选择下一个缓冲区 在将所有帧字节写入 TBUF 寄存器后，主控制器必须设置 TSNEXT 表示此缓冲区已填充。然后 CAN-CTRL 内核将 TBUF 寄存器连接到下一个缓冲区。 一旦缓冲区标记为已填充，就可以使用 TSONE 或 TSALL 启动发送。 可以在一次写访问中，将 TSNEXT，TSONE 或 TSALL 一起设置。 TSNEXT 必须由主控制器设置，并在设置后立即由 CAN-CTRL 内核自动复位。 如果 TBSEL = 0，则设置 TSNEXT 是没有意义的。在这种情况下，TSNEXT 将被忽略并自动清除且没有任何影响。如果 STB 的所有缓冲区都已填满，则 TSNEXT 将保持设置状态，直到缓冲区空闲为止。	0x0	R/W
21	TSMODE	次发送缓冲区模式 0：FIFO 模式 1：优先级决定模式 在 FIFO 模式下，帧按照它们写入 STB 的顺序发送。在优先级决策模式中，首先自动发送 STB 中具有最高优先级的帧。帧的 ID 用于优先级决定。较低的 ID 表示一个具有较高优先级的帧。 无论 ID 如何，PTB 中的帧始终具有最高优先级。只有当 STB 为空时，才能切换 TSMODE。	0x0	R/W
20:18	Res	保留	0x4	-
17:16	TSSTAT	次发送状态位 00：STB 为空	0x0	RO

		01：STB 小于或等于半满 10：STB 多于半满 11：STB 满		
15	TBSEL	发送缓冲区选择 0：PTB(高优先级缓冲区) 1：STB 选择要加载消息的发送缓冲区，使用 TBUF 寄存器进行访问。在写入 TBUF 寄存器和设置 TSNEXT 时，TBSEL 需要始终保持稳定。	0x0	R/W
14	LOM	监听（Listen Only）模式 0：禁用 1：使能 当发送处于活动状态时，不应启用 LOM。如果启用 LOM，则无法启动发送	0x0	R/W
13	STBY	收发器待机（Transceiver Standby） 模式 0：禁用 1：使能 该寄存器位连接到输出信号 standby，可用于控制收发器的待机模式。如果 TPE=1，TSONE=1 或 TSALL=1，STBY 不能被设置成 1 如果主机将 STBY 设置为 0，则收发器在主机请求新发送之前，需要等待时间。	0x0	R/W
12	TPE	主发送使能 1：发送高优先级 PTB 中的消息 0：未发送 PTB 如果设置了 TPE，则来自 PTB 的消息将在下一个可能的发送位置发送。来自 STB 的开始发送将在之前完成，但是等待新消息被延迟直到 PTB 消息已被发送。 TPE 保持设置，直到消息成功发送或使用 TPA 中止。 主控制器可以将 TPE 设置为 1 但不能将其重置为 0，这将只能使用 TPA 并中止消息。 在 CAN 内核重置该位的短时间内，主机无法设置它。 如果 RESET = 1，STBY = 1，(LOM = 1 且 LBME=0)，该位将复位为硬件复位值。	0x0	R/W
11	TPA	主发送中止 1：中止 PTB 的发送，该发送已被 TPE = 1 请求但尚未启动。(消息的数据字节保留在 PTB 中) 0：没有终止 该位必须由主控制器设置，并由 CAN~~-CTRL~~ 复位。 设置 TPA 会自动取消激活 TPE。主控制器可以将 TPA 设置为 1，但不能将其重置为 0。在 CAN~~-CTRL~~ 内核重置该位的短时间内，主机无法设置它。 如果 RESET = 1，该位将复位为硬件复位值。TPA 不应与 TPE 同时设置。	0x0	R/W
10	TSONE	次发送一帧使能 1：STB 中的一个发送使能	0x0	R/W

		在 FIFO 模式下，这是最早的消息。在优先级模式下，这是具有最高优先级的一个。 在优先级模式下，TSONE 难以处理，因为如果同时将新消息写入 STB，则不总是清楚哪个消息将被发送。 一旦总线空闲并且没有 PTB（TPE 位）的请求暂停，控制器就开始发送。 0：没有发送 STB TSONE 保持置位状态，直到消息成功发送或被 TSA 中止。 主控制器可以将 TSONE 设置为 1，但不能将其重置为 0，这只能使用 TSA 并中止消息。 在 CAN 内核重置该位的短时间内，主机无法设置它。 如果 RESET = 1，STBY = 1，(LOM = 1 且 LBME=0)，该位将复位为硬件复位值。		
9	TSALL	次发送所有帧使能 1：发送 STB 中的所有消息 一旦总线空闲并且没有 PTB（TPE 位）的请求暂停，控制器就开始发送。 0：STB 没有发送 TSALL 保持设置状态，直到所有消息都已成功发送或被 TSA 中止。 主控制器可以将 TSALL 设置为 1，但不能将其重置为 0，这只能使用 TSA 并中止消息。 在 CAN 内核重置该位的短时间内，主机无法设置它。 如果 RESET = 1，STBY = 1，(LOM = 1 且 LBME=0)，该位将复位为硬件复位值。	0x0	R/W
8	TSA	次发送中止 1：从 STB 中止已经请求但尚未启动的发送 对于 TSONE 发送，只有一帧被中止，而对于 TSALL 发送，所有帧都被中止。将释放一个或所有消息缓冲区，用于更新 TSSTAT。所有中止的消息都将丢失，因为它们不再可访问。 在优先级模式下，如果 TSONE 发送中止，此时如果同时向 STB 写入新帧，则不清楚 哪个帧将被中止。 0：不终止 该位必须由主控制器设置，并由 CAN-CTRL 复位。设置 TSA，分别自动取消 TSONE 或 TSALL。主控制器可以将 TSA 设置为 1，但不能将其重置为 0。 如果 RESET = 1，该位将复位为硬件复位值。 TSA 不应与 TSONE 或 TSALL 同时设置。	0x0	R/W
7	RESET	复位请求 1：主控制器执行 CAN 内核的本地复位 0：CAN 内核的非本地复位 若 RESET = 1，则只能修改某些寄存器（例如节点配置）。 RESET = 1 会迫使多个组件进入复位状态。 注意：RESET 切换为 0，在 11 个 CAN 位时间后，CAN 节点将参与 CAN 通信。CAN 标准（总线空闲时间）需要此延迟。 若 RESET 设置为 1 并立即设置为 0，则需要一些时间才能将 RESET 读取为 0 并变为非活动状态。原因是时钟从主机到 CAN	0x1	R/W

		控制器需要转换。根据主机和 CAN 时钟之间的关系，RESET 按需保持活动状态。		
6	LBME	回环模式，外部 0：禁止 1：使能 当发送处于活动状态时，不应启用 LBME。	0x0	R/W
5	LBMI	回环模式，内部 0：禁用 1：使能 当发送处于活动状态时，不应启用 LBMI。	0x0	R/W
4	TPSS	PTB 单次发送模式 0：禁用 1：使能	0x0	R/W
3	TSSS	STB 单次发送模式 0：禁用 1：使能	0x0	R/W
2	RACTIVE	接收有效(接收状态位) 0：没有接收活动 1：控制器当前正在接收帧	0x0	RO
1	TACTIVE	发送有效（发送状态位） 0：没有发送活动 1：控制器当前正在发送帧	0x0	RO
0	BUSOFF	总线关闭(总线状态位) 1：控制器状态为“总线关闭（bus off）” 0：控制器状态为“总线打开（bus on）” 将 1 写入 BUSOFF 将重置 TECNT 和 RECNT，这仅用于调试。	0x0	R/W

29.5.3 CAN 控制寄存器 1(CAN_CTRL1)

偏移地址：0xA4
复位值：0x1B0000FE

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
AFWL				EWL				EWARN	EPASS	EPIE	EPIF	ALIE	ALIF	BEIE	BEIF
R/W				R/W				RO	RO	R/W	R/W	R/W	R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RIF	ROIF	RFIF	RAFIF	TPIF	TSIF	EIF	AIF	RIE	ROIE	RFIE	RAFIE	TPIE	TSIE	EIE	TSFF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO	R/W	R/W	R/W	R/W	R/W	R/W	R/W	RO

位	标记	功能描述	复位值	读写
31:28	AFWL	接收缓冲区几乎满告警告限制 AFWL 定义内部警告限制 AFWL_i，其中 nRB 是可用 RB 缓冲区的数量。将 AFWL_i 与填充的 RB 缓冲区的数量进行比较，并且如果相等则触发 RAFIF。 AFWL_i 的有效范围：1…nRB. AFWL_i = 0 无意义，自动被视为 0x1。 (请注意，AFWL 适用于此规则而非 AFWL_i) AFWL_i > nRB 无意义，自动被视为 nRB AFWL_i = nRB 是有效值，但请注意 RFIF 也存在。	0x1	R/W
27:24	EWL	可编程错误警告限制值 = (EWL+1)*8。可能的限制值为 8,16,…128。 EWL 的值控制 EIF 需要使用 CDC 从主机到 CAN 时钟域发送 EWL。在发送期间，在 CDC 完成之前，EWL 寄存器位被主机写锁定几个时钟。	0xB	R/W
23	EWARN	达到错误警告限制 1：错误计数器 RECNT 或 TECNT 的一个等于或大于 EWL 0：两个计数器的值都小于 EWL	0x0	RO
22	EPASS	错误被动模式激活 0：没有激活(节点错误激活) 1：激活(节点错误被动)	0x0	RO
21	EPIE	错误被动中断使能	0x0	R/W
20	EPIF	错误被动中断标志 如果错误状态从错误激活变为错误被动或反之，并且如果启用此中断，则 EPIF 将被激活。 写 1 可以清除该标志。	0x0	R/W
19	ALIE	仲裁失利中断使能	0x0	R/W
18	ALIF	仲裁失利中断标志 写 1 可以清除该标志	0x0	R/W
17	BEIE	总线错误中断使能	0x0	R/W
16	BEIF	总线错误中断标志 总线错误类型对应 KOER。 写 1 可以清除该标志	0x0	R/W
15	RIF	接收中断标志 1：数据或远程帧已接收，并且可在接收缓冲区中使用	0x0	R/W

		0：没有帧被接收 写 1 可以清除该标志		
14	ROIF	RB 溢出中断标志 1：RB 中至少有一条接收消息被覆盖 0：没有 RB 被覆盖 如果发生溢出，ROIF 和 RFIF 都将被设置。 写 1 可以清除该标志	0x0	R/W
13	RFIF	RB 满中断标志 1：所有 RB 都已满。如果在收到下一个有效消息之前没有释放 RB，则最旧的消息将会丢失 0：RB FIFO 未满 写 1 可以清除该标志	0x0	R/W
12	RAFIF	RB 几乎满中断标志 1：填充的 RB 缓冲区数 ≥ AFWL 0：填充的 RB 缓冲区数 < AFWL 写 1 可以清除该标志	0x0	R/W
11	TPIF	主发送中断标志 1：已成功完成所请求的 PTB 发送 0：没有完成 PTB 的发送 写 1 可以清除该标志	0x0	R/W
10	TSIF	次发送中断标志 1：已成功完成所请求的 STB 发送 0：没有完成 STB 的发送 写 1 可以清除该标志	0x0	R/W
9	EIF	错误中断标志 1：错误警告限制的边界已在任一方向上穿越，或者 BUSOFF 位已在任一方向上改变 0：没有改变	0x0	R/W
8	AIF	中止中断标志 1：在设置 TPA 或 TSA 之后，已经中止了相应的消息 建议不要同时设置 TPA 和 TSA，因为两者都是源 AIF。 0：没有执行中止。 AIF 没有关联的使能寄存器。 写 1 可以清除该标志	0x0	RO
7	RIE	接收中断使能 0：禁用 1：使能	0x1	R/W
6	ROIE	RB 溢出中断使能 0：禁用 1：使能	0x1	R/W
5	RFIE	RB 满中断使能 0：禁用 1：使能	0x1	R/W
4	RAFIE	RB 几乎满中断使能 0：禁用 1：使能	0x1	R/W

3	TPIE	主发送（Transmission Primary）中断使能 0：禁用 1：使能	0x1	R/W
2	TSIE	次发送（Transmission Secondary）中断使能 0：禁用 1：使能	0x1	R/W
1	EIE	错误中断（Error Interrupt） 使能 0：禁用 1：使能，中断标志对应 EIF	0x1	R/W
0	TSFF	次发送缓冲区满标志 1：STB 填充最大数目的消息 0：STB 没有填充最大数目的消息	0x0	RO

29.5.4 低速 CAN 通信波特率配置寄存器(CAN_SBITRATE)

偏移地址：0xA8
复位值：0x01020203

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
S_PRESC								保留	S_SJW						
R/W									R/W						

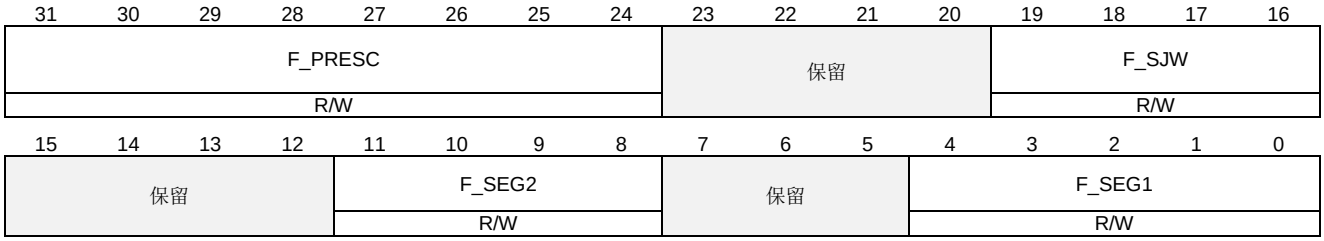
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	S_SEG2							S_SEG1							
	R/W							R/W							

位	标记	功能描述	复位值	读写
31:24	S_PRESC	预分频器 预分频器将 CAN 时钟分频以获得时间量子时钟 tq_clk。有效范围 S_PRESC=[0x00,0xff]，导致分频器值为 1 ~ 256。	0x1	R/W
23	Res	保留	0x0	-
22:16	S_SJW	同步补偿宽度 同步补偿宽度 tSJW =(S_SJW+1).TQ 是缩短或延长重新同步的位时间的最长时间，其中 TQ 是时间因子。	0xB	R/W
15	Res	保留	0x0	-
14:8	S_SEG2	位定时段 2（低速） 在采样点后定时 tSeg_2 =(S_Seg_2+ 1). TQ	0x2	R/W
7:0	S_SEG1	位定时段 1（低速） 位定时开始后，采样点被设置为 tSeg_1 =(S_Seg_1+ 2). TQ	0x3	R/W

注：CAN_SBITRATE 寄存器只能在寄存器 CAN_CTRL0.RESET 位置 1 的时候才能写入。

29.5.5 高速 CAN 通信波特率配置寄存器(CAN_FBITRATE)

偏移地址：0xAC
复位值：0x01020203



位	标记	功能描述	复位值	读写
31:24	F_PRESC	预分频器 预分频器将 CAN 时钟分频以获得时间量子时钟 tq_clk。有效范围 S_PRESC=[0x00,0xff]，导致分频器值为 1 ~ 256。	0x1	R/W
23:20	Res	保留	0x0	-
19:16	F_SJW	同步补偿宽度 同步补偿宽度 tSJW =(F_SJW+1).TQ 是缩短或延长重新同步的位时间的最长时间，其中 TQ 是时间因子。	0xB	R/W
15:12	Res	保留	0x0	-
11:8	F_SEG2	位定时段 2（低速） 在采样点后定时 tSeg_2 =(F_Seg_2+ 1).TQ	0x2	R/W
7:5	Res	保留	0x0	-
4:0	F_SEG1	位定时段 1（快速） 位定时开始后，采样点被设置为 tSeg_1 =(F_Seg_1+2).TQ	0x3	R/W

注：CAN_FBITRATE 寄存器只能在寄存器 CAN_CTRL0.RESET 位置 1 的时候才能写入。

29.5.6 CAN 错误类型及计数寄存器(CAN_ERRINFO)

偏移地址：0xB0
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TDC EN	SSPOFF							KOER			ALC				
R/W	R/W							RO			RO				

位	标记	功能描述	复位值	读写
31:24	TECNT	发送错误计数器（发送过程中的错误数） TECENT 按照 CAN 规范中的定义递增和减少 “busoff”状态下 TECNT 可以溢出，“busoff”以外 TECNT 不溢出，将 0xff = 255 作为最大值 有关 TECNT 和“总线关闭（bus off）”状态的更多详细信息请参考 29.3.2	0x0	RO
23:16	RECNT	接收错误计数器（接收过程中的错误数） RECENT 按照 CAN 规范中的定义递增和减少 RECNT 不溢出，RECNT 将 0xff = 255 作为最大值 有关 RECNT 和“总线关闭（bus off）”状态的更多详细信息请参考 29.3.2	0x0	RO
15	TDCEN	发送器延迟补偿使能 如果 TDCEN = 1，则在 BRS 处于活动状态时，将在 CAN FD 帧的数据阶段激活	0x0	R/W
14:8	SSPOFF	二次采样点偏移 传输延迟 SSPOFF 为 TDC 定义了二次采样点的时间 SSPOFF 是 TQ 的一个数	0x0	R/W
7:5	KOER	错误类型(错误代码) 000：没有错误（no error） 001：位错误（BIT ERROR） 010：形式错误（FORM ERROR） 011：填充错误（STUFF ERROR） 100：应答错误（ACKNOWLEDGEMENT ERROR） 101：校验错误（CRC ERROR） 110：其他错误（OTHER ERROR） (错误标志后收到显性电平，接收到主动错误标志太长，ACK 错误后的被动错误标志收到显性电平)。 111：未使用 KOER 会随着每个新错误而更新。因此当帧被成功地发送或接收时，它保持不变。	0x0	RO
4:0	ALC	仲裁失利捕获 仲裁失利的帧中的位所处的位置	0x0	RO

注：CAN_ERRINFO 位只能在寄存器 CAN_CTRL0.RESET 位置 1 的时候才能写入。

29.5.7 接收过滤器控制寄存器(CAN_ACFCTRL)

偏移地址：0xB4
复位值：0x00010200

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留						TIMEPOS	TIMEEN	保留		SELMASK	保留	ACFADR			
						R/W	R/W			R/W		R/W			

位	标记	功能描述	复位值	读写
31:16	ACFEN_x	接收过滤器使能 ACFEN_x（x=0~15） 0：接收过滤器禁用 1：接收过滤器使能 每个接收过滤器(AMASK/ACODE) 可以单独启用或禁用。硬件复位后，默认情况下仅启用 0 号过滤器。 禁用过滤器拒绝接收消息。如果相应的 AMASK/ACODE 配置匹配，则仅启用的过滤器可以接受消息。 要接受所有消息，必须通过设置 ACFEN_x = 1，AMASK_x = 0xff 和 ACODE_x = 0x00 来启用一个过滤器 x。这是过滤器 x = 0 硬件重置后的默认配置，而所有其他过滤器都被禁用。	0x1	R/W
15:10	Res	保留	0x0	-
9	TIMEPOS	时间戳标记位置 0：SOF 1：EOF 仅当 TIMEEN = 0 时才能更改 TIEMPOS，也可同时修改 TIMEPOS 和设置 TIMEEN=1	0x1	R/W
8	TIMEEN	时间戳标记使能 0：禁用 1：使能	0x0	R/W
7:6	Res	保留	0x0	-
5	SELMASK	选择接受掩码（Select acceptance MASK） 0：ACF_x 寄存器指向验证码，配置接收代码 ACODE 寄存器 CAN_ACF。 1：ACF_x 寄存器指向接收掩码，配置接收 AMASK 寄存器 CAN_ACF。 ACFADR 选择一个特定的接收过滤器。	0x0	R/W
4	Res	保留	0x0	-
3:0	ACFADR	接收过滤器地址 ACFADR 指向某一个接收过滤器 0-15。选择后才能使用寄存器 ACF_x 设置对应过滤器。 SELMASK 位在所选接受收过滤器的验证码和掩码之间进行选择。	0x0	R/W

注：CAN_ACFCTRL 位只能在寄存器 CAN_CTRL0.RESET 位置 1 的时候才能写入。

29.5.8 接收代码 ACODE 寄存器(CAN_ACF)

偏移地址：0xB8
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留			ACODE												
			R/W												

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

位	标记	功能描述	复位值	读写
31:29	Res	保留位	0x0	-
28:0	ACODE	接收代码（Acceptance CODE），需设置 SELMASK 等于 0 0/1：每个位与接收消息的 ID 位进行比较 ACODEx(10:0)用于标准帧，ACODE_x(28:0)用于扩展帧，只有 0 号过滤器受上电复位的影响，所有其他过滤器保持未初始化状态。	0x0	R/W

注：CAN_ACF 寄存器只能在寄存器 CAN_CTRL0.RESET 位置 1 的时候才能写入。

29.5.9 接收掩码 AMASK 寄存器(CAN_ACF)

偏移地址：0xB8
复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留	AIDEE	AIDE	AMASK												
	R/W	R/W	R/W												

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

位	标记	功能描述	复位值	读写
31	Res	保留	0x0	-
30	AIDEE	接收掩码 IDE 位检查使能，需设置 SELMASK 等于 1 1：接收过滤器接收 AIDE 定义的标准或扩展 0：接收过滤器接收标准或扩展帧 只有 0 号过滤器受上电复位的影响，所有其他过滤器保持未初始化状态。	0x0	R/W
29	AIDE	接收掩码 IDE 位的值，需设置 SELMASK 等于 1 若 AIDEE=1，则： 1：接收过滤器仅接收扩展帧 0：接收过滤器仅接收标准帧	0x0	R/W
28:0	AMASK	接收掩码（Acceptance MASK），需设置 SELMASK 等于 1 1：屏蔽对应的接收过滤位 0：匹配对应的接收过滤位 AMASK_x(10：0)被用做标准帧，AMASK_x(28：0)被用做扩展帧。禁用位导致接受该消息。因此，过滤器 0 重置后的默认配置接受收所有消息。只有 0 号过滤器受上电复位的影响，所有其他过滤器保持未初始化状态。	0x0	R/W

注：CAN_ACF 寄存器只能在寄存器 CAN_CTRL0.RESET 位置 1 的时候才能写入。

30 模拟/数字转换器(ADC)

30.1 模块简介

本芯片内部集成了一个 12 位高精度、高转换速率的逐次逼近型模数转换器(SAR ADC)模块。具有以下特性：

- 12 位转换精度；
- 转换速率最高可到 1MSPS（ADCCLK=PCLK=24MHz）；
- 22 路转换通道：18 个 IO 引脚通道、1 个 AVDD/3、1 个 AVREF(1V)、1 个温度传感器通道、1 个 PGA 通道；
- 4 种参考电压源：内部 2V、内部 4V、AVDD、外部参考 ExRef；
- ADC 的电压输入范围：0~Vref；
- 3 种转换模式：单次转换、顺序扫描连续转换、插队扫描连续转换；
- 输入通道电压阈值监测；
- ADC 转换开始条件
  - 软件启动
  - GPIO 中断触发启动
  - 其他模块触发启动
- 软件可配置 ADC 的转换速率
- 内置采样通道 Buffer，可转换高阻信号
- 支持片内外设自动触发 ADC 转换，有效降低芯片功耗并提高转换的实时性

30.2 ADC 框图

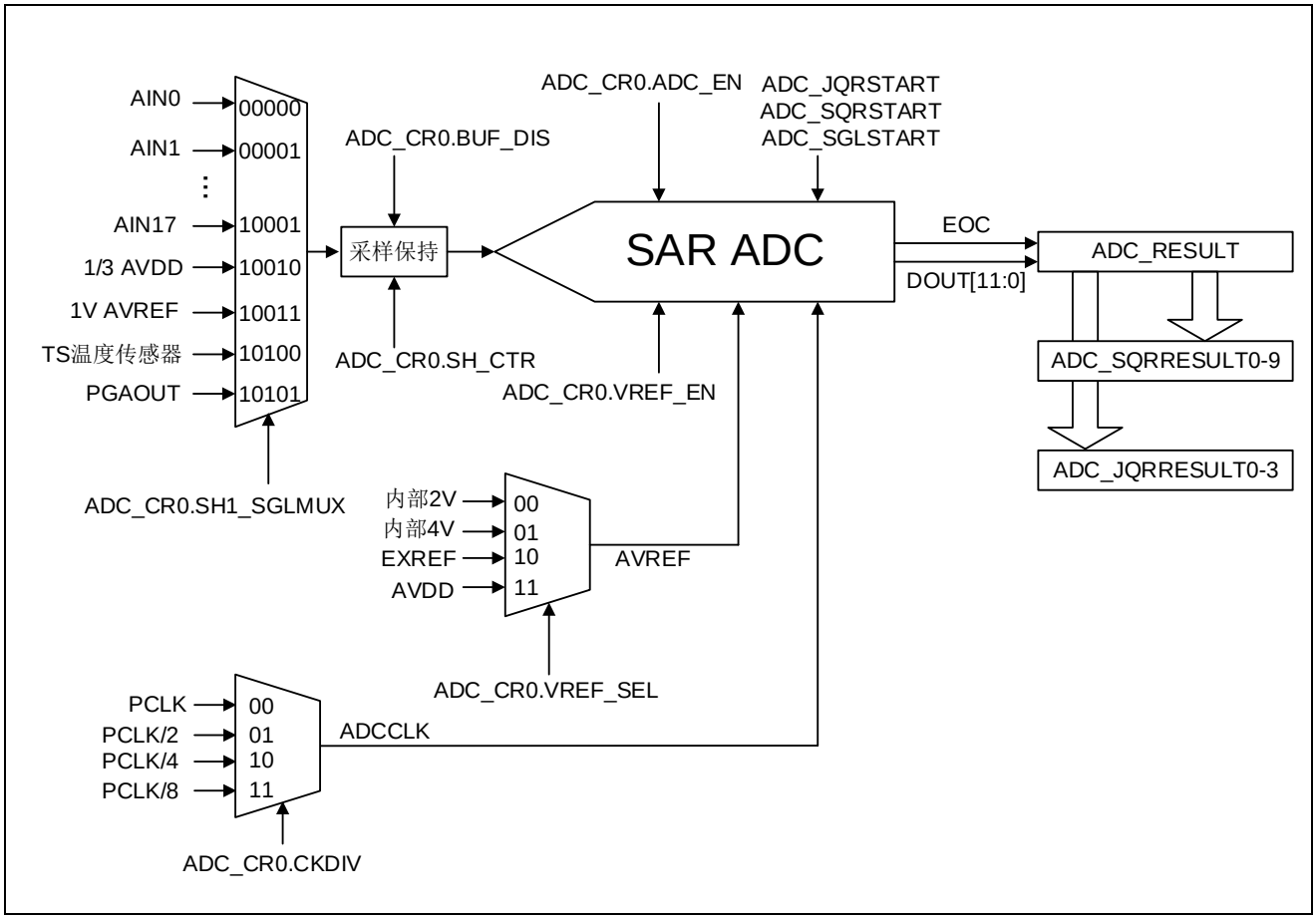


图 30-1 ADC 框图

30.3 转换时序及速度

ADC 的转换时序如下图所示：一次完整的 ADC 转换由采样过程及逐次比较过程组成。其中采样过程需要 4~11 个 AdcClk，由 ADC_CR0.SH_CTR 配置；逐次比较过程需要 13 个 ADCCLK。所以，一次 ADC 转换共需要 17~24 个 ADCCLK。

ADC 转换速度的单位为 SPS，即每秒进行多少次 ADC 转换。

ADC 转换速度的计算方法：ADCCLK 的频率/一次 ADC 转换所需要的 ADCCLK 的个数。

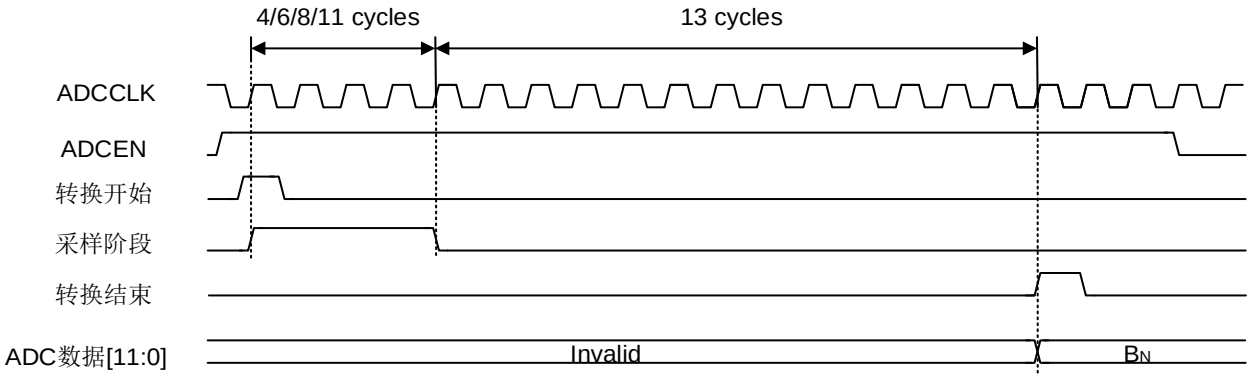


图 30-2 ADC 转换时序图

ADC 转换速度与 ADC 参考电压及 AVDD 电压相关，最高转换速度如下表所示：

ADC 参考电压	AVDD 电压	最高转换速率	最大 ADCCLK 频率
内部 2V	2.7V~5.5V	200K SPS	4MHz
内部 4V	4.4V~5.5V	200K SPS	4MHz
AVDD/EXREF	2.7V~5.5V	1M SPS	24MHz

30.4 转换模式选择

30.4.1 单次转换模式

在单次转换模式下，ADC 启动后只执行一次转换，可对所有的 22 路 ADC 通道进行转换。该模式既可通过设置 ADC_SGLSTART.SGL_START 位启动，也可通过设置 ADC_EXTTRIGGER0 的外部触发启动。一旦选定通道的 ADC 转换完成，ADC_IFR.SGL_IF 位会自动置 1，转换结果保存在 ADC_RESULT 寄存器中。

通过 ADC_SGLSTART.SGL_START 位启动 ADC 进行单次转换操作流程(不使能中断)：

Step1：配置 GPIOx_AFR1/2 相应的位，将待转换的 IO 配置为模拟端口。

Step2：设置 GPIOB_AFR1.PBAFR5 为 0xF，将 ADC 外部参考电压引脚 PB5 配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 ADC_CR0.ADC_EN 为 1，使能 ADC 模块。

Step4：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step5：设置 ADC_CR1.MODE 为 0，选择单次转换模式。

Step6：配置 ADC_CR0.VREF_SEL，选择 ADC 的参考电压。

Step7：设置 ADC_CR0.VREF_EN 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step8：配置 ADC_CR0.SH_CTR 及 ADC_CR0.CKDIV，设置 ADC 的转换速度。

Step9：配置 ADC_CR0.SH1_SGLMUX，选择待转换的通道。

Step10：设置 ADC_ICR.SGL_IC 为 0，清除 ADC_IFR.SGL_IF 标志。

Step11：设置 ADC_SGLSTART.SGL_START 为 1，启动 ADC 单次转换。

Step12：等待 ADC_IFR.SGL_IF 变为 1，读取 ADC_RESULT 寄存器以获取 ADC 转换结果。

Step13：如需对其它通道进行转换，重复执行 Step9~Step12。

Step14：设置 ADC_CR0.ADC_EN 及 ADC_CR0.VREF_EN 为 0，关闭 ADC 模块、VREF 模块。

通过外部触发启动 ADC 单次转换操作流程(使能中断)：

Step1：配置 GPIOx_AFR1/2 相应的位，将待转换的 IO 配置为模拟端口。

Step2：设置 GPIOB_AFR1.PBAFR5 为 0xF，将 ADC 外部参考电压引脚 PB5 配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 ADC_CR0.ADC_EN 为 1，使能 ADC 模块。

Step4：延时 20us，等待 ADC 及 BGR 模块启动完成。
Step5：设置 ADC_CR1.MODE 为 0，选择单次转换模式。
Step6：设置 ADC_IER.SGL_IE 为 1，使能 ADC 单次转换中断。
Step7：使能 NVIC 中断向量表中的 ADC 中断。
Step8：配置 ADC_CR0.VREF_SEL，选择 ADC 的参考电压。
Step9：设置 ADC_CR0.VREF_EN 为 1，使能 ADC 内部参考电压。
注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。
Step10：配置 ADC_CR0.SH_CTR 及 ADC_CR0.CKDIV，设置 ADC 的转换速度。
Step11：配置 ADC_CR0.SH1_SGLMUX，选择待转换的通道。
Step12：设置 ADC_ICR.SGL_IC 为 0，清除 ADC_IFR.SGL_IF 标志。
Step13：配置 ADC_EXTTRIGGER0，选择外部触发条件。
Step14：当外部触发条件触发 ADC 完成转换时，ADC 模块会产生中断。用户可在 ADC 中断服务程序中读取 ADC_RESULT 寄存器以获取 ADC 转换结果。
Step15：如需对其它通道进行转换，重复执行 Step11~Step14。
Step16：设置 ADC_CR0.ADC_EN 及 ADC_CR0.VREF_EN 为 0，关闭 ADC 模块、VREF 模块。

30.4.2 扫描转换模式

扫描转换模式分为顺序扫描转换和插队扫描转换两种模式。两种模式各自单独工作时，均可连续对多个通道进行多次转换；当两种模式同时工作时，则优先对插队扫描配置的通道进行转换。

30.4.2.1 顺序扫描转换模式

顺序扫描转换模式下，最多可进行 10 次连续转换，转换的总次数由 ADC_SQR1.SH1_CNT 进行配置；可配置对所有 22 个通道进行转换，待转换通道由 ADC_SQRx.SH1_CHxMUX 进行配置。该模式既可通过设置 ADC_SQRSTART.SQR_START 位启动也可通过设置 ADC_EXTTRIGGER0 的外部触发启动。启动转换后，ADC 模块依次转换 CHxMUX~CH0MUX 中配置的通道直到总转换次数完成。ADC 模块完成总转换次数后，ADC_IFR.SH1_SQR_IF 位会自动置 1，转换结果保存在转换通道所对应的 ADC_SQRRESULT9~ADC_SQRRESULT0 寄存器中。
下图 30-3 演示了 ADC 对 AIN0、AIN1、AIN5 进行 8 次转换的顺序扫描转换过程。其中顺序扫描转换通道 7，4，1 配置为 AIN1，转换通道 6，3，0 配置为 AIN0，转换通道 5，2 配置为 AIN5。ADC_SQRSTART.SQR_START 置 1 后，ADC 模块会依次对顺序扫描转换通道 7~0 进行转换。

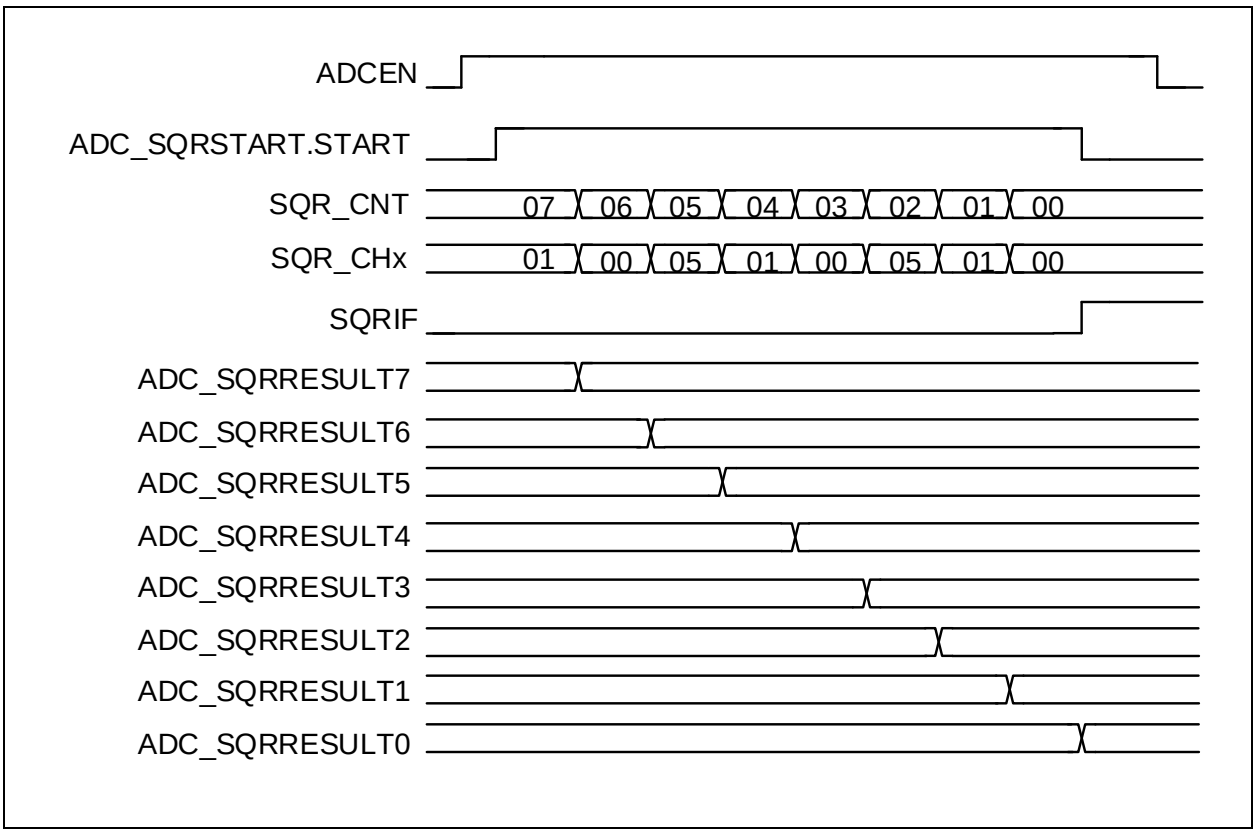


图 30-3 ADC 顺序扫描转换过程示例

通过 ADC_SQRSTART.SQR_START 位启动 ADC 进行顺序扫描转换操作流程(不使能中断)：

Step1：配置 GPIOx_AFR1/2 相应的位，将待转换的 IO 配置为模拟端口。

Step2：设置 GPIOB_AFR1.PBAFR5 为 0xf，将 ADC 外部参考电压引脚 PB5 配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 ADC_CR0.ADC_EN 为 1，使能 ADC 模块。

Step4：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step5：设置 ADC_CR1.MODE 为 1，选择扫描转换模式。

Step6：配置 ADC_CR0.VREF_SEL，选择 ADC 的参考电压。

Step7：设置 ADC_CR0.VREF_EN 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step8：配置 ADC_CR0.SH_CTR 及 ADC_CR0.CKDIV，设置 ADC 的转换速度。

Step9：配置 ADC_SQR0/1.CHxMUX，选择顺序扫描转换通道。

Step10：配置 ADC_SQR1.CNT，选择顺序扫描转换的总转换次数。

注：若 CNT=x，则 ADC 依次对通道 CHx，CHx-1，…，CH1，CH0 进行转换。

Step11：设置 ADC_ICR.SH1_SQR_IC 为 0，清除 ADC_IFR.SH1_SQR_IF 标志。

Step12：设置 ADC_SQRSTART.SQR_START 为 1，启动 ADC 顺序扫描转换。

Step13：等待 ADC_IFR.SH1_SQR_IF 变为 1，读取 ADC_SQRRESULTx~ADC_SQRRESULT0 寄存器以获取相应通道的转换结果。

Step14：如需对其它通道进行转换，重复执行 Step9~Step13。

Step15：设置 ADC_CR0.ADC_EN 及 ADC_CR0.VREF_EN 为 0，关闭 ADC 模块、

VREF 模块。

通过外部触发启动 ADC 进行顺序扫描转换操作流程(使能中断)：

Step1：配置 GPIOx_AFR1/2 相应的位，将待转换的 IO 配置为模拟端口。

Step2：设置 GPIOB_AFR1.PBAFR5 为 0xF，将 ADC 外部参考电压引脚 PB5 配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 ADC_CR0.ADC_EN 为 1，使能 ADC 模块。

Step4：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step5：设置 ADC_CR1.MODE 为 1，选择扫描转换模式。

Step6：设置 ADC_IER.SH1_SQR_IE 为 1，使能 ADC 顺序扫描转换中断。

Step7：使能 NVIC 中断向量表中的 ADC 中断。

Step8：配置 ADC_CR0.VREF_SEL，选择 ADC 的参考电压。

Step9：设置 ADC_CR0.VREF_EN 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step10：配置 ADC_CR0.SH_CTR 及 ADC_CR0.CKDIV，设置 ADC 的转换速度。

Step11：配置 ADC_SQR0/1.CHxMUX，选择顺序扫描转换通道。

Step12：配置 ADC_SQR1.CNT，选择顺序扫描转换的总转换次数。

Step13：设置 ADC_ICR.SH1_SQR_IC 为 0，清除 ADC_IFR.SH1_SQR_IF 标志。

Step14：配置 ADC_EXTTRIGGER0，选择外部触发条件。

Step15：当外部触发条件触发 ADC 完成转换时，ADC 模块会产生中断。用户可在 ADC 中断服务程序中读取 ADC_SQRRESULTx~ADC_SQRRESULT0 寄存器以获取相应通道的转换结果。

Step16：如需对其它通道进行转换，重复执行 Step11~Step15。

Step17：设置 ADC_CR0.ADC_EN 及 ADC_CR0.VREF_EN 为 0，关闭 ADC 模块、VREF 模块。

30.4.2.2 插队扫描转换模式

插队扫描转换模式下，ADC 最多可进行 4 次连续转换，转换的总次数由 ADC_JQR.SH1_CNT 进行配置；可配置对所有 22 个通道进行转换，待转换通道由 ADC_JQR.CHxMUX 进行配置。该模式既可通过设置 ADC_JQRSTART.SHx_START 位启动也可通过设置 ADC_EXTTRIGGER1 的外部触发启动。启动转换后，ADC 模块依次转换 CHxMUX~CH0Mux 中配置的通道直到总转换次数完成。ADC 模块完成总转换次数后，ADC_IFR.SHx_JQRIF 位会自动置 1，转换结果保存在转换通道所对应的 ADC_JQRRESULTx~ADC_JQRRESULT0 寄存器中。

下图演示了在单采样模式下，对 AIN0、AIN1、AIN5 进行 4 次转换的插队扫描转换的过程。其中插队扫描转换通道 3，2，1，0 分别设置为 AIN5，AIN0，AIN1，AIN5。ADC_JQRSTART.SH1_START 置 1 后，ADC 模块会依次对插队扫描转换通道 3~0 进行转换。

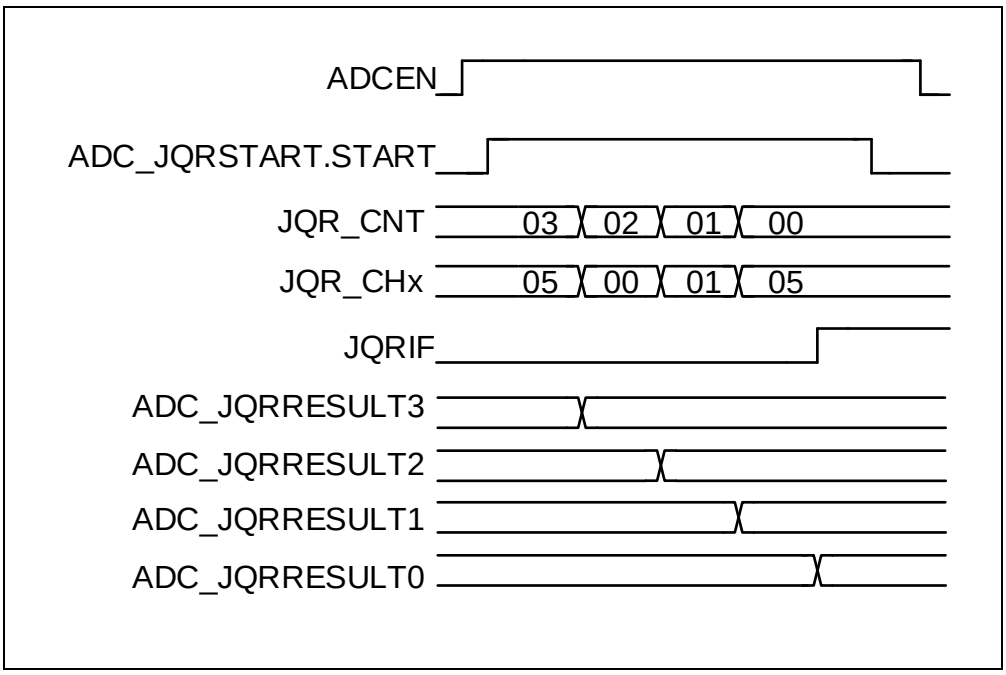


图 30-4 ADC 插队扫描转换过程示例

通过 ADC_SQRSTART.SH1_START 位启动 ADC 进行插队扫描转换操作流程(不使能中断)：

Step1：配置 GPIOx_AFR1/2 相应的位，将待转换的 IO 配置为模拟端口。

Step2：设置 GPIOB_AFR1.PBAFR5 为 0xf，将 ADC 外部参考电压引脚 PB5 配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 ADC_CR0.ADC_EN 为 1，使能 ADC 模块。

Step4：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step5：设置 ADC_CR1.MODE 为 1，选择扫描转换模式。

Step6：配置 ADC_CR0.VREF_SEL，选择 ADC 的参考电压。

Step7：设置 ADC_CR0.VREF_EN 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step8：配置 ADC_CR0.SH_CTR 及 ADC_CR0.CKDIV，设置 ADC 的转换速度。

Step9：配置 ADC_JQR.CHxMUX，选择插队扫描转换通道。

Step10：配置 ADC_JQR.CNT，选择插队扫描转换的总转换次数。

注：若 CNT=x，则 ADC 依次对通道 CHx，CHx-1，…，CH1，CH0 进行转换。

Step11：设置 ADC_ICR.SH1_JQR_IC 为 0，清除 ADC_IFR.SH1_JQR_IF 标志。

Step12：设置 ADC_JQRSTART.SH1_START 为 1，启动 ADC 序列插队扫描转换。

Step13：等待 ADC_IFR.SH1_JQR_IF 变为 1，读取 ADC_JQRRESULTx~ADC_JQRRESULT0 寄存器以获取相应通道的转换结果。

Step14：如需对其它通道进行转换，重复执行 Step9~Step13。

Step15：设置 ADC_CR0.ADC_EN 及 ADC_CR0.VREF_EN 为 0，关闭 ADC 模块、VREF 模块。

通过外部触发启动 ADC 序列进行插队扫描转换操作流程(使能中断)：

Step1：配置 GPIOx_AFR1/2 相应的位，将待转换的 IO 配置为模拟端口。

Step2：设置 GPIOB_AFR1.PBAFR5 为 0xf，将 ADC 外部参考电压引脚 PB5 配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step3：设置 ADC_CR0.ADC_EN 为 1，使能 ADC 模块。

Step4：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step5：设置 ADC_CR1.SH1_MODE 为 1，选择 ADC 扫描转换模式。

Step6：设置 ADC_IER.SH1_JQR_IE 为 1，使能 ADC 插队扫描转换中断。

Step7：使能 NVIC 中断向量表中的 ADC 中断。

Step8：配置 ADC_CR0.VREF_SEL，选择 ADC 的参考电压。

Step9：设置 ADC_CR0.VREF_EN 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step10：配置 ADC_CR0.SH_CTR 及 ADC_CR0.CKDIV，设置 ADC 的转换速度。

Step11：配置 ADC_JQR0.CHxMUX，选择插队扫描转换通道。

Step12：配置 ADC_JQR0.CNT，选择插队扫描转换的总转换次数。

Step13：设置 ADC_ICR.SH1_JQR_IC 为 0，清除 ADC_IFR.SH1_JQR_IF 标志。

Step14：配置 ADC_EXTTRIGGER0，选择外部触发条件。

Step15：当外部触发条件触发 ADC 完成转换时，ADC 模块会产生中断。用户可在 ADC 中断服务程序中读取 ADC_JQRRESULTx~ADC_JQRRESULT0 寄存器以获取相应通道的转换结果。

Step16：如需对其它通道进行转换，重复执行 Step11~Step15。

Step17：设置 ADC_CR0.ADC_EN 及 ADC_CR0.VREF_EN 为 0，关闭 ADC 模块、VREF 模块。

插队扫描转换的执行优先级高于顺序扫描转换，当顺序扫描转换正在进行时，如果插队扫描转换启动，则顺序扫描在完成当前通道的转换后暂停，等到插队扫描转换完成后，再继续执行剩下的通道转换。

下图演示了在单采样模式下，对 AIN0，AIN1，AIN5，AIN8 进行顺序扫描转换时，启动对 AIN2，AIN6 进行插队扫描转换的过程。其中顺序扫描转换通道 3，2，1，0 分别设置为 AIN1，AIN0，AIN5，AIN8，插队扫描转换通道 1，0 分别设置为 AIN6，AIN2。在 ADC 对顺序扫描进行 AIN0 转换的过程中，启动了插队扫描，顺序扫描会将当前 AIN0 的转换完成然后暂停，等到插队扫描完成 AIN6 和 AIN2 的转换后，顺序扫描再进行 AIN5 和 AIN8 的转换。

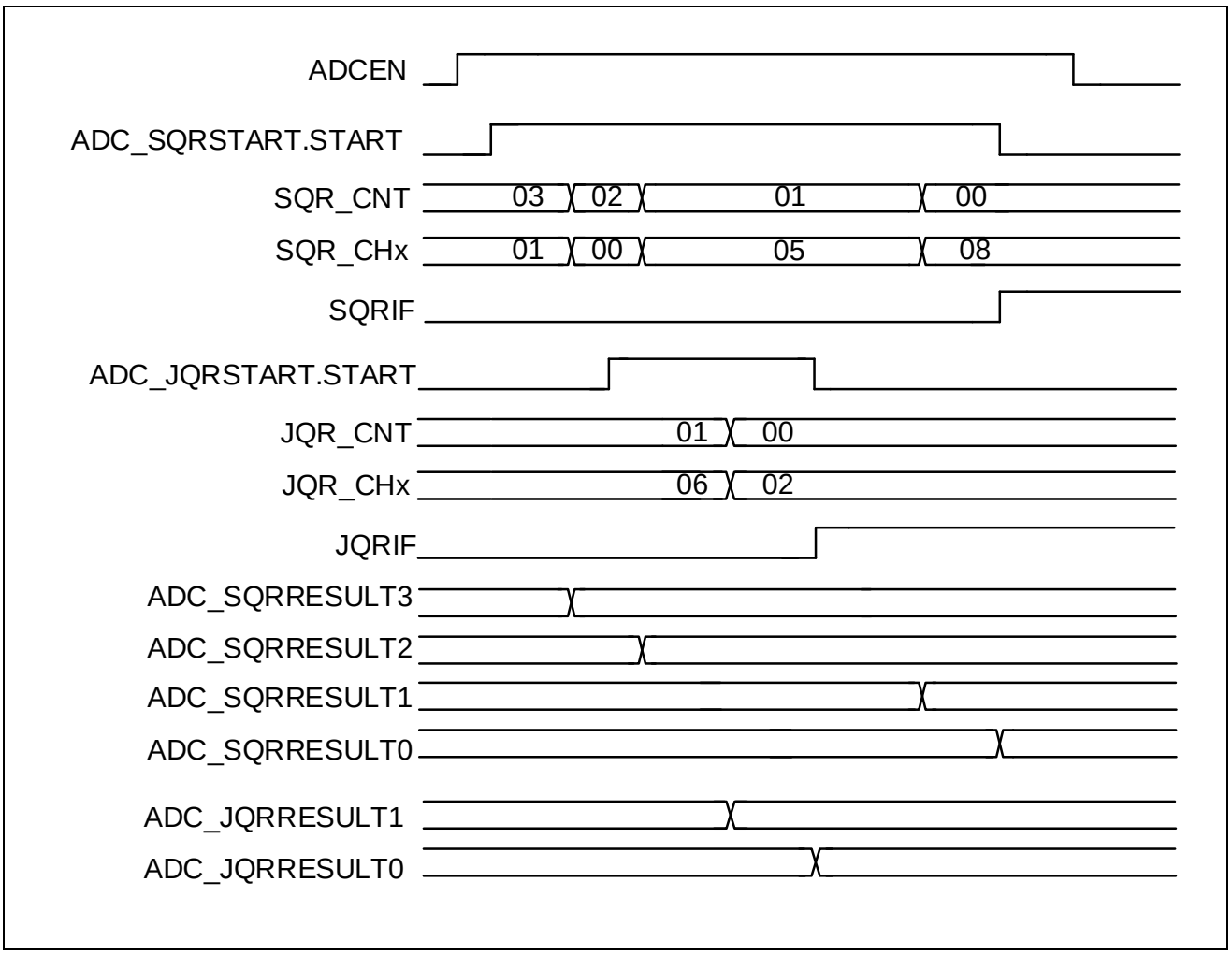


图 30-5 ADC 顺序扫描过程中进行插队扫描示例

30.5 ADC 转换结果比较

ADC 转换完成时，ADC 转换结果可以与用户设定的阈值进行比较，支持上阈值比较、下阈值比较、区间值比较。该功能需要将相应的控制位 ADC_CR1.HTCMP、ADC_CR1.LTCMP、ADC_CR1.REGCMP 置 1。该功能可实现对模拟量的自动监测，直到 ADC 转换结果符合用户预期时才产生中断申请用户程序介入。监测通道选择通过 ADC_CR1.ThCh 进行配置。

上阈值比较：当 ADC 转换结果位于[ADC_HT，4095]区间内则 ADC_IFR.HT_IF 置 1；向 ADC_ICR.HHT_IC 写入 0 则清零 ADC_IFR.HT_IF。
下阈值比较：当 ADC 转换结果位于[0,ADC_LT]区间内则 ADC_IFR.LT_IF 置 1；向 ADC_ICR.LLT_IC 写入 0 则清零 ADC_IFR.LT_IF。
区间值比较：当 ADC 转换结果位于[ADC_LT，ADC_HT]区间内则 ADC_IFR.REG_IF 置 1；向 ADC_ICR.REG_IC 写入 0 则清零 ADC_IFR.REG_IF。

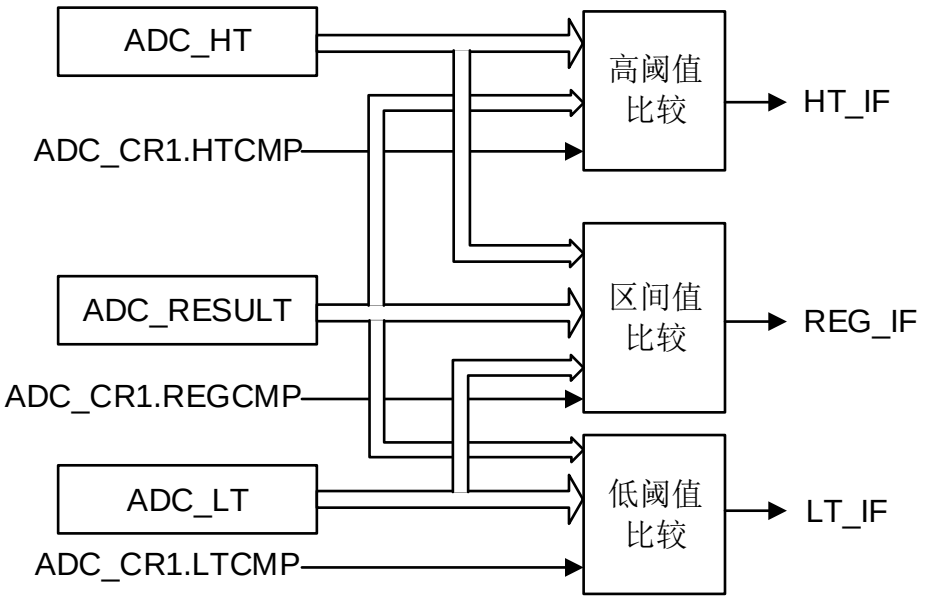


图 30-6 ADC 转换结果比较示意图

30.6 ADC 中断

ADC 中断请求如下表所示：

中断源	中断标志	中断使能
ADC 单次转换完成	ADC_IFR.SGL_IF	ADC_IER.SGL_IE
ADC 顺序扫描转换完成中断	ADC_IFR.SH1_SQR_IF	ADC_IER.SH1_SQR_IE
ADC 插队扫描转换完成中断	ADC_IFR.SH1_JQR_IF	ADC_IER.SH1_JQR_IE
ADC 转换结果位于区间值区域	ADC_IFR.REG_IF	ADC_IER.REG_IE
ADC 转换结果位于上阈值区域	ADC_IFR.HT_IF	ADC_IER.HT_IE
ADC 转换结果位于下阈值区域	ADC_IFR.LT_IF	ADC_IER.LT_IE

30.7 温度传感器简介与使用

30.7.1 TSN 简介

温度传感器的输出电压会随环境温度的改变而变化，故根据温度传感器的输出电压即可计算出相应的环境温度。当 ADC 模块的测量通道选择温度传感器的输出电压时，即可测量环境温度。

注意：因为内部 TSN 是一个弱驱动能力的电压源，所以要用 ADC 对其进行采样和转换的时候，必须保证 TSN 有足够的安定时间 20μS 以上才能被 ADC 采样。

即从 ADC_CR0.EN 有效，到启动 ADC 转换，必须经过 20μS 以上的等待。同时采样和转换速率也要满足要求。

30.7.2 温度传感器 TSN 的使用

芯片对应的环境温度的推导公式如下：

Ts=(Vs-V1)/slope + T1

Ts： 当前的 MCU 所处的温度(℃)

Vs: TSN 对应的当前温度从而输出的电压值

V1: TSN 对应的 Ta=25℃时，FT 测试时所得到的输出电压值。

Slope: 温度斜率曲线。(可以根据 Tj=135℃/-40℃所对应的值推算出来)

T1: 25(℃)

计算示例如下：

3. 通过 TSN_HIGHTEMP 和 TSN_LOWTEMP 这两个地址中保存的值来推算出当前这个 MCU 里 TSN 对应的温度斜率曲线。

Slope=(TSN_HIGHTEMP-TSN_LOWTEMP)/175

因为 TSN_HIGHTEMP 和 TSN_LOWTEMP 对应的 ADC 变换值都是在 AVDD=VDD=3.3V 的时候测定的，算出的温度斜率曲线也是 AVDD=VDD=3.3V 对应的。如果 ADC 的参考电压是其他的，需要进行换算。

2.根据当前 ADC 的参考电压和 TSN 的输出电压转换得到一个当前 TSN 输出电压的值，换算得到当前的 Vs。

3.根据 TSN_NORMTEMP 地址里保存的 25℃时，对应的 TSN 输出电压值，来推导当前电压值。

例如：TSN_HIGHTEMP 保存的值是 0xF00, TSN_LOWTEMP 保存的值是 0x100，那么当前芯片的温度斜率曲线就是 slope= (0xF00-0x100)/ (135+40) = 20.48/℃。

根据 ADC 当前 VREF 的参考电压得到一个当前 TSN 的输出电压，例如参考电压是内部 4.0V,得到 ADC 转换的结果是 0x400(1024)，从而推导出当前的 TSN 输出电压是 4.0V×(1024/4095)=1.0V

如果 TSN_NORMTEMP 里面保存的值是:0x200(512),对应的电压就是:

3.3V×(512/4095)=0.4126V

那么当前的温度就是: Ts=（1.0-0.4126）× 20.48+25 = 37℃

通过 ADC 测量环境温度操作流程：

Step1：设置 GPIOB_AFR1.PBAFR5 为 0xf，将 ADC 外部参考电压引脚 PB5 配置为模拟端口。

注：如果 ADC 参考电压不选择外部参考电压引脚，则可以略过本步骤。

Step2：设置 ADC_CR0.ADC_EN 为 1，使能 ADC 模块。

Step3：延时 20us，等待 ADC 及 BGR 模块启动完成。

Step4：设置 ADC_CR1.MODE 为 0，选择单次转换模式。

Step5：配置 ADC_CR0.VREF_SEL，选择 ADC 的参考电压。

Step6：设置 ADC_CR0.VREF_EN 为 1，使能 ADC 内部参考电压。

注：如果 ADC 参考电压不选择内部参考电压，则可以略过本步骤。

Step7：配置 ADC_CR0. SH_CTR 及 ADC_CR0.CKDIV，设置 ADC 的转换速度。

Step8：配置 ADC_CR0.SH1_SGLMUX，选择温度传感器 TS 通道。

Step9：配置 ADC_CR0.BUF_EN 为 1，使能通道输入 Buffer。

Step10：设置 ADC_ICR.SGL_IC 为 0，清除 ADC_IFR.SGL_IF 标志。

Step11：设置 ADC_SGLSTART.SGL_START 为 1，启动 ADC 单次转换。

Step12：等待 ADC_IFR.SGL_IF 变为 1，读取 ADC_RESULT 寄存器以获取 ADC 转换结果。

Step13：根据公式计算当前的环境温度。

Step14：设置 ADC_CR0.ADC_EN 及 ADC_CR0.VREF_EN 为 0，关闭 ADC 模块、VREF 模块。

30.8 寄存器列表

ADC 基地址：0x4000 2C00

偏移地址	名称	描述	复位值
0x00	ADC_CR0	ADC配置寄存器0	0x0000 F1F0
0x04	ADC_CR1	ADC配置寄存器1	0x0000 0000
0x08	ADC_SQR0	ADC顺序扫描转换通道配置寄存器0	0x0000 0000
0x0C	ADC_SQR1	ADC顺序扫描转换通道配置寄存器1	0x0000 0000
0x10	-	保留	-
0x14	ADC_JQR	ADC插队扫描转换通道配置寄存器	0x0000 0000
0x18	ADC_SQRRESULT0	ADC顺序扫描转换通道0转换结果	0x0000 0000
0x1C	ADC_SQRRESULT1	ADC顺序扫描转换通道1转换结果	0x0000 0000
0x20	ADC_SQRRESULT2	ADC顺序扫描转换通道2转换结果	0x0000 0000
0x24	ADC_SQRRESULT3	ADC顺序扫描转换通道3转换结果	0x0000 0000
0x28	ADC_SQRRESULT4	ADC顺序扫描转换通道4转换结果	0x0000 0000
0x2C	ADC_SQRRESULT5	ADC顺序扫描转换通道5转换结果	0x0000 0000
0x30	ADC_SQRRESULT6	ADC顺序扫描转换通道6转换结果	0x0000 0000
0x34	ADC_SQRRESULT7	ADC顺序扫描转换通道7转换结果	0x0000 0000
0x38	ADC_SQRRESULT8	ADC顺序扫描转换通道8转换结果	0x0000 0000
0x3C	ADC_SQRRESULT9	ADC顺序扫描转换通道9转换结果	0x0000 0000
0x40~4C	-	保留	-
0x50	ADC_JQRRESULT0	ADC插队扫描转换通道0转换结果	0x0000 0000
0x54	ADC_JQRRESULT1	ADC插队扫描转换通道1转换结果	0x0000 0000
0x58	ADC_JQRRESULT2	ADC插队扫描转换通道2转换结果	0x0000 0000
0x5C	ADC_JQRRESULT3	ADC插队扫描转换通道3转换结果	0x0000 0000
0x60	ADC_RESULT	ADC转换结果寄存器	0x0000 0000
0x64	-	保留	-
0x68	ADC_HT	ADC比较上阈值	0x0000 0FFF
0x6C	ADC_LT	ADC比较下阈值	0x0000 0000
0x70	ADC_IER	ADC中断使能寄存器	0x0000 0000
0x74	ADC_IFR	ADC中断标志寄存器	0x0000 0000
0x78	ADC_ICR	ADC中断清除寄存器	0x0000 003F
0x7C	ADC_EXTTRIGGER0	ADC单次转换或顺序扫描转换外部中断触发源配置寄存器	0x0000 0000
0x80	ADC_EXTTRIGGER1	ADC插队扫描转换外部中断触发源配置寄存器	0x0000 0000
0x84	ADC_SGLSTART	ADC单次转换启动控制寄存器	0x0000 0000
0x88	ADC_SQRSTART	ADC顺序扫描转换启动控制寄存器	0x0000 0000
0x8C	ADC_JQRSTART	ADC插队扫描转换启动控制寄存器	0x0000 0000

温度传感器基地址：0x1FFF_F050

偏移地址	名称	描述	默认值
0x00	TSN_HIGHTEMP	Tj=135℃时对应的TSN测得的温度,AVDD=VDD=3.30V	测试测得值
0x04	TSN_LOWTEMP	Tj=-40℃时对应的TSN测得的温度,AVDD=VDD=3.30V	测试测得值
0x08	TSN_NORMTEMP	Ta=25℃时对应的TSN测得的温度,AVDD=VDD=3.30V	测试测得值

30.9 ADC 寄存器说明

30.9.1 ADC 配置寄存器 0(ADC_CR0)

地址偏移：0x00

复位值：0x0000 F1F0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															VREF_EN
															R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SH_CTR		VREF_SEL		BUF_EN	保留		SH1_SGLMUX					CKDIV		保留	ADC_EN
R/W		R/W		R/W			R/W					R/W			R/W

位	标记	功能描述	复位值	读写
31:17	Res	保留，始终读为 0。	0x0	-
16	VREF_EN	ADC 内部参考电压使能(内部参考电压 4V/2V) 0：禁止内部参考电压 1：使能内部参考电压	0x0	R/W
15:14	SH_CTR	ADC 采样周期选择 00：4 个采样周期 01：6 个采样周期 10：8 个采样周期 11：11 个采样周期	0x3	R/W
13:12	VREF_SEL	ADC 参考电压选择 00：内部 2V 01：内部 4V 10：外部参考电压 VREFEXT（PB5） 11：AVDD 电压（默认）	0x3	R/W
11	BUF_DIS	ADC 输入信号放大器控制 0：打开输入 BUF，外部输入信号通过 BUF 电路后与 ADC 相连，用于高阻信号。 1：关闭输入 BUF，外部输入信号与 ADC 直接相连。 以下几种情况需要打开 BUF 功能，使用 BUF 功能时，最大速率 200k sps 1）外部驱动很弱信号 2）测量 1/3AVCC 3）测量温度传感器输出电压 4）测量 1V 基准电压	0x0	R/W
10:9	Res	保留，始终读为 0。	0x0	-
8:4	SH1_SGLMUX	ADC 转换通道选择(单次转换模式) 00000：AIN0（PC4） 00001：AIN1（PA8） 00010：AIN2（PC3） 00011：AIN3（PC2） 00100：AIN4（PC1） 00101：AIN5（PC0） 00110：AIN6（PB15） 00111：AIN7（PA9）	0x1F	R/W

		01000：AIN8（PC15） 01001：AIN9（PC14） 01010：AIN10（PB3） 01011：AIN11（PB2） 01100：AIN12（PB1） 01101：AIN13（PB0） 01110：AIN14（PA7） 01111：AIN15（PA6） 10000：AIN16（PA1） 10001：AIN17（PA0） 10010：选择 1/3 AVDD（注：ADC_CR0.BUF_EN 必须为 0） 10011：选择 1V AVREF（注：ADC_CR0.BUF_EN 必须为 0） 10100：选择 TS 温度传感器输出（注：ADC_CR0.BUF_EN 必须为 0） 10101：选择 PGA_OUT 10110~11111：保留		
3:2	CKDIV	ADC 时钟选择（注：时钟最大不能超过 24MHz） 00：PCLK 时钟 01：PCLK 时钟 2 分频 10：PCLK 时钟 4 分频 11：PCLK 时钟 8 分频	0x0	R/W
1	Res	保留，始终读为 0。	0x0	-
0	ADC_EN	ADC 使能控制 0：禁止 ADC 1：使能 ADC	0x0	R/W

30.9.2 ADC 配置寄存器 1(ADC_CR1)

地址偏移：0x04
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	REGCMP	HTCMP	LTCMP	保留			MODE	保留		THCH					ALIGN
	R/W	R/W	R/W				R/W			R/W					R/W

位	标记	功能描述	复位值	读写
31:15	Res	保留，始终读为 0。	0x0	-
14	REGCMP	ADC 区间比较控制 0：禁止区间比较 1：使能区间比较	0x0	R/W
13	HTCMP	ADC 高阈值比较控制 0：禁止高阈值比较 1：使能高阈值比较	0x0	R/W
12	LTCMP	ADC 低阈值比较控制 0：禁止低阈值比较 1：使能低阈值比较	0x0	R/W
11:9	Res	保留，始终读为 0。	0x0	-
8	MODE	ADC 转换模式选择 0：单次转换模式 1：扫描转换模式	0x0	R/W
7:6	Res	保留，始终读为 0。	0x0	-
5:1	THCH	阈值比较通道选择 00000：选择 SH1 顺序扫描转换通道 0 进行阈值比较 …… 01001：选择 SH1 顺序扫描转换通道 9 进行阈值比较 01010~01101：保留 01110：选择 SH1 插队扫描转换通道 0 进行阈值比较 …… 10001：选择 SH1 插队扫描转换通道 3 进行阈值比较 10010：选择 ADC_RESULT 进行阈值比较 10011~11111：保留	0x0	R/W
0	ALIGN	转换结果对齐控制 1：转换结果 12Bits 左对齐存储（转换结果寄存器取值范围 bit15~bit4） 0：转换结果 12Bits 右对齐存储（转换结果寄存器取值范围 bit11~bit0）	0x0	R/W

30.9.3 ADC 顺序扫描转换通道配置寄存器 0 (ADC_SQR0)

地址偏移：0x08

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留		SH1_CH5MUX					SH1_CH4MUX					SH1_CH3MUX			
		R/W					R/W					R/W			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SH1_CH2MUX					SH1_CH1MUX					SH1_CH0MUX				
	R/W					R/W					R/W				

位	标记	功能描述	复位值	读写
31:30	Res	保留，始终读为0。	0x0	-
29:25	SH1_CH5MUX	SH1顺序扫描转换通道5选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W
24:20	SH1_CH4MUX	SH1顺序扫描转换通道4选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W
19:15	SH1_CH3MUX	SH1顺序扫描转换通道3选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W
14:10	SH1_CH2MUX	SH1顺序扫描转换通道2选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W
9:5	SH1_CH1MUX	SH1顺序扫描转换通道1选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W
4:0	SH1_CH0MUX	SH1顺序扫描转换通道0选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W

30.9.4 ADC 顺序扫描转换通道配置寄存器 1(ADC_SQR1)

地址偏移：0x0C

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								SH1_CNT				SH1_CH9MUX			
								R/W				R/W			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SH1_CH8MUX					SH1_CH7MUX					SH1_CH6MUX				
	R/W					R/W					R/W				

位	标记	功能描述	复位值	读写
31:24	Res	保留，始终读为0。	0x0	-
23:20	SH1_CNT	顺序扫描转换次数 0000：0次转换 0001：1次转换 ..... 1010：10次转换 1011~1111：禁止使用	0x0	R/W
19:15	SH1_CH9MUX	SH1顺序扫描转换通道9选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W
14:10	SH1_CH8MUX	SH1顺序扫描转换通道8选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W
9:5	SH1_CH7MUX	SH1顺序扫描转换通道7选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W
4:0	SH1_CH6MUX	SH1顺序扫描转换通道6选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W

30.9.5 ADC 插队扫描转换通道配置寄存器(ADC_JQR)

地址偏移：0x14

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留									SH1_CNT			SH1_CH3MUX			
									R/W			R/W			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SH1_CH2MUX					SH1_CH1MUX						SH1_CH0MUX			
	R/W					R/W						R/W			

位	标记	功能描述	复位值	读写
31:23	Res	保留，始终读为0。	0x0	-
22:20	SH1_CNT	SH1插队扫描转换次数 000：0次转换 001：1次转换 010：2次转换 011：3次转换 100：4次转换 101~111：禁止使用	0x0	R/W
19:15	SH1_CH3MUX	SH1插队扫描转换通道3选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W
14:10	SH1_CH2MUX	SH1插队扫描转换通道2选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W
9:5	SH1_CH1MUX	SH1插队扫描转换通道1选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W
4:0	SH1_CH0MUX	SH1插队扫描转换通道0选择，设置参见ADC_CR0.SH1_SGLMUX	0x0	R/W

30.9.6 ADC 顺序扫描转换通道 X 转换结果(ADC_SQRRESULT0-9)

地址偏移：0x18/0x1C/0x20/0x24/0x28/0x2C/0x30/0x34/0x38/0x3C

复位值：0x0000 0000

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

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

位	标记	功能描述	复位值	读写
31:12	Res	保留，始终读为0。	0x0	-
11:0	RESULT	ADC顺序扫描转换通道x转换结果	0x0	RO

30.9.7 ADC 插队扫描转换通道 X 转换结果(ADC_JQRRESULT0-3)

地址偏移：0x50/0x54/0x58/0x5C
复位值：0x0000 0000

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

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

位	标记	功能描述	复位值	读写
31:12	Res	保留，始终读为0。	0x0	-
11:0	RESULT	ADC插队扫描转换通道x转换结果	0x0	RO

30.9.8 ADC 转换结果(ADC_RESULT)

地址偏移：0x60
复位值：0x0000 0000

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

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

位	标记	功能描述	复位值	读写
31:12	Res	保留，始终读为0。	0x0	-
11:0	RESULT	ADC转换结果	0x0	RO

30.9.9 ADC 比较上阈值(ADC_HT)

地址偏移：0x68
复位值：0x0000 0FFF

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				HT											
				R/W											

位	标记	功能描述	复位值	读写
31:12	Res	保留，始终读为0。	0x0	-
11:0	HT	ADC转换结果比较上阈值	0xFFF	R/W

30.9.10 ADC 比较下阈值(ADC_LT)

地址偏移：0x6C
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				LT											
				R/W											

位	标记	功能描述	复位值	读写
31:12	Res	保留，始终读为0。	0x0	-
11:0	LT	ADC转换结果比较下阈值	0x0	R/W

30.9.11 ADC 中断使能寄存器(ADC_IER)

地址偏移：0x70
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留										SH1_JQR_IE	SH1_SQR_IE	REG_IE	HT_IE	LT_IE	SGL_IE
										R/W	R/W	R/W	R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:6	Res	保留，始终读为 0。	0x0	-
5	SH1_JQR_IE	ADC 插队扫描转换完成中断使能配置 0：禁止 1：使能	0x0	R/W
4	SH1_SQR_IE	ADC 顺序扫描转换完成中断使能配置 0：禁止 1：使能	0x0	R/W
3	REG_IE	ADC 转换结果比较区间中断使能配置 0：禁止 1：使能	0x0	R/W
2	HT_IE	ADC 转换结果比较上阈值中断使能配置 0：禁止 1：使能	0x0	R/W
1	LT_IE	ADC 转换结果比较下阈值中断使能配置 0：禁止 1：使能	0x0	R/W
0	SGL_IE	ADC 单次转换完成中断使能配置 0：禁止 1：使能	0x0	R/W

30.9.12 ADC 中断标志寄存器(ADC_IFR)

地址偏移：0x74

复位值：0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留										SH1_JQR_IF	SH1_SQR_IF	REG_IF	HT_IF	LT_IF	SGL_IF
										RO	RO	RO	RO	RO	RO

位	标记	功能描述	复位值	读写
31:6	Res	保留，始终读为 0。	0x0	-
5	SH1_JQR_IF	ADC 插队扫描转换完成中断标志 0：ADC 插队扫描转换未完成 1：ADC 插队扫描转换完成	0x0	RO
4	SH1_SQR_IF	ADC 顺序扫描转换完成中断标志 0：ADC 顺序扫描转换未完成 1：ADC 顺序扫描转换完成	0x0	RO
3	REG_IF	ADC 转换结果比较区间中断标志 0：ADC 转换结果位于[ADC_LT，ADC_HT]区间外 1：ADC 转换结果位于[ADC_LT，ADC_HT]区间内	0x0	RO
2	HT_IF	ADC 转换结果比较上阈值中断标志 0：ADC 转换结果位于[ADC_HT，4095]区间外 1：ADC 转换结果位于[ADC_HT，4095]区间内	0x0	RO
1	LT_IF	ADC 转换结果比较下阈值中断标志 0：ADC 转换结果位于[0，ADC_LT]区间外 1：ADC 转换结果位于[0，ADC_LT]区间内	0x0	RO
0	SGL_IF	ADC 单次转换完成中断标志 0：ADC 单次转换未完成 1：ADC 单次转换完成	0x0	RO

注意：这些所有的中断标注都是和中断使能位与逻辑以后或到一起然后输出给同一个中断向量。如果需要精确定位具体的中断标志以及中断源的话，最好只让一个标志有效。

30.9.13 ADC 中断清除寄存器(ADC_ICR)

地址偏移：0x78

复位值：0x0000 003F

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留										SH1_JQR_IC	SH1_SQR_IC	REG_IC	HT_IC	LT_IC	SGL_IC
										RCW0	RCW0	RCW0	RCW0	RCW0	RCW0

位	标记	功能描述	复位值	读写
31:6	Res	保留，始终读为 0。	0x0	-
5	SH1_JQR_IC	ADC 插队扫描转换完成中断标志清除配置 0：写 0 清除 ADC 插队扫描转换完成标志 1：写 1 无作用	0x1	RCW0
4	SH1_SQR_IC	ADC 顺序扫描转换完成中断标志清除配置 0：写 0 清除 ADC 顺序扫描转换完成标志 1：写 1 无作用	0x1	RCW0
3	REG_IC	ADC 转换结果比较区间中断标志清除配置 0：写 0 清除 ADC 转换结果比较区间标志 1：写 1 无作用	0x1	RCW0
2	HHT_IC	ADC 转换结果比较上阈值中断标志清除配置 0：写 0 清除 ADC 转换结果比较上阈值 1：写 1 无作用	0x1	RCW0
1	LLT_IC	ADC 转换结果比较下阈值中断标志清除配置 0：写 0 清除 ADC 转换结果比较下阈值标志 1：写 1 无作用	0x1	RCW0
0	SGL_IC	ADC 单次转换完成中断标志清除配置 0：写 0 清除 ADC 单次转换完成标志 ADC_IFR.SGL_IF 1：写 1 无作用	0x1	RCW0

30.9.14 ADC 单次转换或顺序扫描转换外部中断触发源配置寄存器(ADC_EXTTRIGGER0)

地址偏移：0x7C

复位值：0x0000 0000

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

位	标记	功能描述	复位值	读写
31:5	Res	保留，始终读为0。	0x0	-
4:0	EXT_TRIGS0	ADC单次转换或顺序扫描转换外部中断触发源配置位 00000：禁用自动触发ADC转换 00001：TIM10中断，自动触发ADC转换 00010：TIM11中断，自动触发ADC转换 00011：TIM1中断，自动触发ADC转换 00100：LPTIM中断，自动触发ADC转换 00101：TIM1 TRGO，自动触发ADC转换 00110：TIM2 TRGO，自动触发ADC转换 00111：TIM2中断，自动触发ADC转换 01000：UART0中断，自动触发ADC转换 01001：UART1中断，自动触发ADC转换 01010：LPUART中断，自动触发ADC转换 01011：VC中断，自动触发ADC转换 01100：保留 01101：RTC中断，自动触发ADC转换 01110：PCA中断，自动触发ADC转换 01111：SPI0中断，自动触发ADC转换 10000：PA1中断，自动触发ADC转换 10001：PA2中断，自动触发ADC转换 10010：PA3中断，自动触发ADC转换 10011：PB4中断，自动触发ADC转换 10100：PB5中断，自动触发ADC转换 10101：PC3中断，自动触发ADC转换 10110：PC4中断，自动触发ADC转换 10111：PC5中断，自动触发ADC转换 11000：PC6中断，自动触发ADC转换 11001：PC7中断，自动触发ADC转换 11010：PD1中断，自动触发ADC转换 11011：PD2中断，自动触发ADC转换 11100：PD3中断，自动触发ADC转换 11101：PD4中断，自动触发ADC转换 11110：PD5中断，自动触发ADC转换 11111：PD6中断，自动触发ADC转换 Note: 触发ADC使用的是各中断标志位的上升沿。如果需要重复	0x0	R/W

		触发，需要清除中断标志。如果不需要进入中断服务程序，请不要使能NVIC的中断使能。		

30.9.15 ADC 插队扫描转换外部中断触发源配置寄存器(ADC_EXTTRIGGER1)

地址偏移：0x80

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留											JQ_TRIGS				
											R/W				

位	标记	功能描述	复位值	读写
31:5	Res	保留，始终读为0。	0x0	-
4:0	EXT_TRIGS1	ADC插队扫描转换外部中断触发源配置位 00000：禁用自动触发ADC转换 00001：TIM10中断，自动触发ADC转换 00010：TIM11中断，自动触发ADC转换 00011：TIM1中断，自动触发ADC转换 00100：LPTIM中断，自动触发ADC转换 00101：TIM1 TRGO，自动触发ADC转换 00110：TIM2 TRGO，自动触发ADC转换 00111：TIM2中断，自动触发ADC转换 01000：UART0中断，自动触发ADC转换 01001：UART1中断，自动触发ADC转换 01010：LPUART中断，自动触发ADC转换 01011：VC中断，自动触发ADC转换 01100：保留 01101：RTC中断，自动触发ADC转换 01110：PCA中断，自动触发ADC转换 01111：SPI中断，自动触发ADC转换 10000：PA1中断，自动触发ADC转换 10001：PA2中断，自动触发ADC转换 10010：PA3中断，自动触发ADC转换 10011：PB4中断，自动触发ADC转换 10100：PB5中断，自动触发ADC转换 10101：PC3中断，自动触发ADC转换 10110：PC4中断，自动触发ADC转换 10111：PC5中断，自动触发ADC转换 11000：PC6中断，自动触发ADC转换 11001：PC7中断，自动触发ADC转换 11010：PD1中断，自动触发ADC转换 11011：PD2中断，自动触发ADC转换 11100：PD3中断，自动触发ADC转换 11101：PD4中断，自动触发ADC转换 11110：PD5中断，自动触发ADC转换	0x0	R/W

		11111：PD6中断，自动触发ADC转换 Note: 触发ADC使用的是各中断标志位的上升沿。如果需要重复触发，需要清除中断标志。如果不需要进入中断服务程序，请不要使能NVIC的中断使能。		

30.9.16 ADC 单次转换启动控制寄存器(ADC_SGLSTART)

地址偏移：0x84

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															SGL_START
															R/W

位	标记	功能描述	复位值	读写
31:1	Res	保留，始终读为0。	0x0	-
0	SGL_START	ADC单次转换启动控制 0：停止ADC单次转换 1：启动ADC单次转换 注：该位软件写1，逻辑清零	0x0	R/W

30.9.17 ADC 顺序扫描转换启动控制寄存器(ADC_SQRSTART)

地址偏移：0x88

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															SQR_START
															R/W

位	标记	功能描述	复位值	读写
31:1	Res	保留，始终读为0。	0x0	-
0	SQR_START	ADC顺序扫描启动控制 0：停止ADC顺序扫描转换 1：启动ADC顺序扫描转换 注：该位软件写1，逻辑清零	0x0	R/W

30.9.18 ADC 插队扫描转换启动控制寄存器(ADC_JQRSTART)

地址偏移：0x8C
复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															Jqr_START
															R/W

位	标记	功能描述	复位值	读写
31:1	Res	保留，始终读为0。	0x0	-
0	JQR_START	ADC插队扫描启动控制 0：停止ADC插队扫描转换 1：启动ADC插队扫描转换 注：该位软件写1，逻辑清零	0x0	R/W

30.10 温度传感器寄存器说明

30.10.1 高温测试时对应的 TSN 测得的温度值(TSN_HIGHTEMP)

地址偏移：0x00
复位值：测试测得值

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

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

位	标记	功能描述	复位值	读写
31:12	Res	保留，始终读为1。	0x1	-
11:0	RESULT	Tj=135℃时，CP高温测试所测得的TSN输出电压值(AVDD=3.3V,VREF=3.3V)	测试测得值	RO

30.10.2 低温测试时对应的 TSN 测得的温度值(TSN_LOWTEMP)

地址偏移：0x04
复位值：测试测得值

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

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

位	标记	功能描述	复位值	读写
31:12	Res	保留，始终读为1。	0x1	-
11:0	RESULT	Tj=-40℃时，CP低温测试所测得的TSN输出电压值(AVDD=3.3V,VREF=3.3V)	测试测得值	RO

30.10.3 常温测试时对应的 TSN 测得的温度值(TSN_NORMTEMP)

地址偏移：0x08
复位值：测试测得值

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

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

位	标记	功能描述	复位值	读写
31:12	Res	保留，始终读为1。	0x1	-
11:0	RESULT	Ta=25℃时，FT常温测试所测得的TSN输出电压值(AVDD=3.3V,VREF=3.3V)	测试测得值	RO

31 低电压检测器(LVD)

31.1 LVD 简介

LVD 可用于监测工作电压，当被监测电压与 LVD 阈值的比较结果满足触发条件时，LVD 会产生中断或复位信号。中断或复位信号只能被中断或复位清零信号清除，只有当中断或复位信号被清零后，才会在产生触发条件下，再次产生中断或复位信号。

采样滤波时钟可配置，可配置为 PCLK 时钟或者 LIRC，滤波计数值可配置。

3 种触发条件：高电平，上升沿，下降沿组合。

2 种触发结果：中断，复位信号(禁止在滤波时钟选择 PCLK 时选择产生复位信号)，中断与复位信号不能同时产生。

31.2 LVD 框图

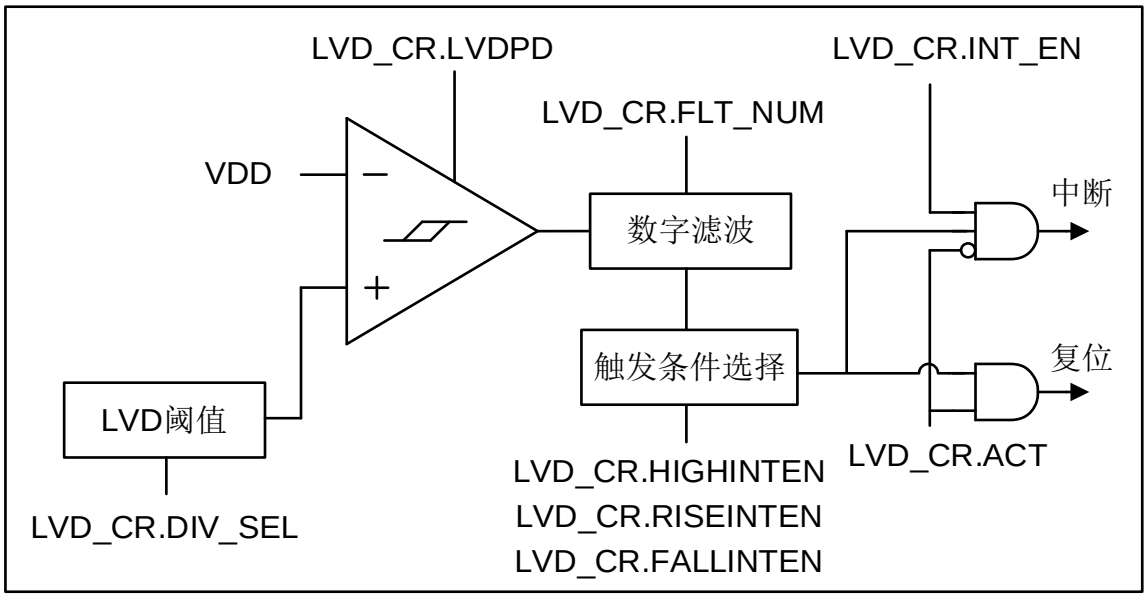


图 31-1 LVD 结构框图

31.3 数字滤波

如果芯片的工作环境恶劣，迟滞比较器的输出会出现噪声信号。使能数字滤波模块，则迟滞比较器的输出波形中脉宽小于 LVD_CR.FLT_NUM 的噪声信号都可以被滤除。禁止数字滤波模块，则数字滤波模块的输入输出信号相同。使能数字滤波模块，滤波示意如下所示：

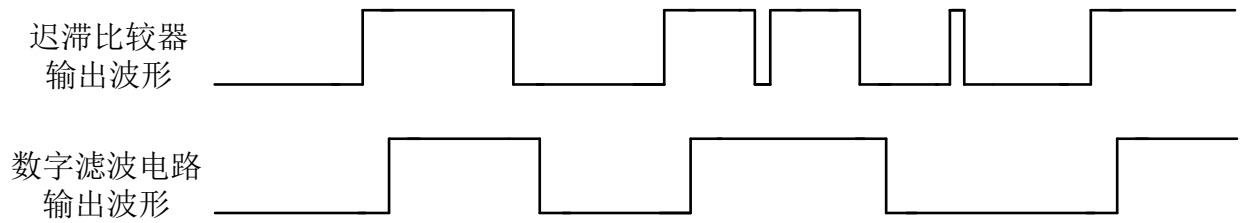


图 31-2 LVD 滤波输出

31.4 迟滞功能

LVD 内置的电压比较器具有迟滞功能，其输出信号会等到输入信号高于或低于阈值电压 20mV 后才发生翻转。迟滞功能可以增强芯片的抗干扰能力，如下图所示。

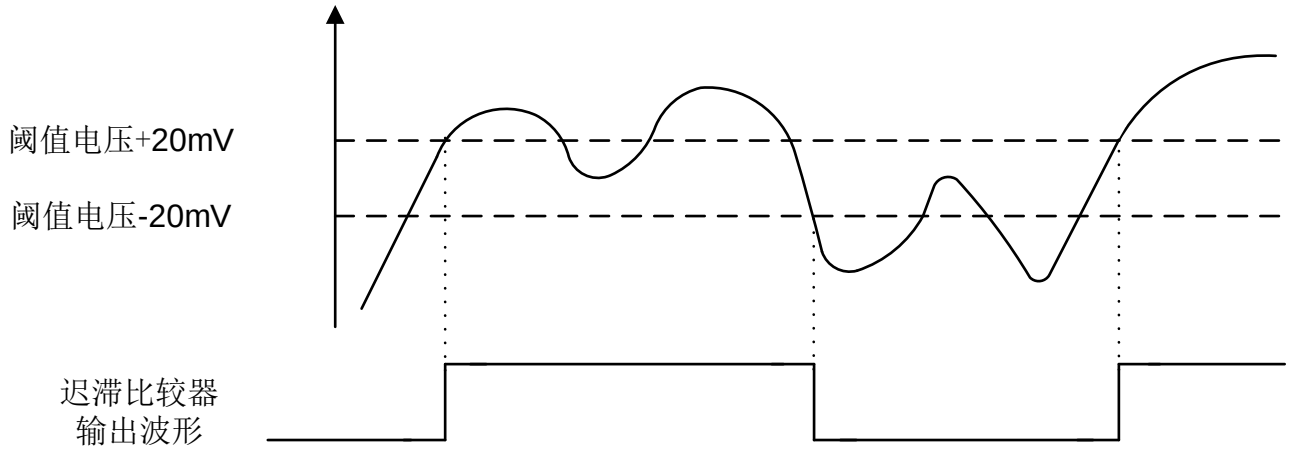


图 31-3 LVD 迟滞响应

31.5 配置示例

31.5.1 LVD 配置为低电压复位

在本模式下，监测电压低于阈值电压时复位 MCU。配置方法如下所示：

Step1：配置 LVD_CR.TH_VOLT_SEL，选择待监测的阈值电压。

Step2：配置 LVD_CR.FLT_NUM，选择 LVD 滤波采样次数。

Step3：配置 LVD_CR.FLTCLK_SEL,选择滤波时钟为 LIRC（不能选择 PCLK 时钟）。

Step4：配置 LVD_CR.FLTEN，使能 LVD 数字滤波。

Step5：设置 LVD_CR.HIGHINTEN 为 1，选择高电平触发 LVD 动作。

Step6：设置 LVD_CR.ACT 为 1，选择 LVD 动作为复位。

Step7：设置 LVD_CR.LVDEN 为 0，打开 LVD 功能。

31.5.2 LVD 配置为电压变化中断

在本模式下，监测电压高于或低于阈值电压时产生中断。配置方法如下所示：

Step1：配置 LVD_CR.TH_VOLT_SEL，选择待监测的阈值电压。

Step2：配置 LVD_CR.FLT_NUM，选择 LVD 滤波采样次数。

Step3：配置 LVD_CR.FLTCLK_SEL，选择滤波时钟。

Step4：配置 LVD_CR.FLTEN，使能 LVD 数字滤波。

Step5：设置 LVD_CR.HIGHINTEN 为 1，选择高电平触发 LVD 动作。

Step6：设置 LVD_CR.ACT 为 0，选择 LVD 动作为中断。

Step7：设置 LVD_CR.INT_EN 为 1，使能 LVD 中断。

Step8：设置 LVD_CR.LVDEN 为 0，打开 LVD 功能。

Step9：在中断服务程序中向 LVD_SR 写入 0x0 以清除中断标志。

31.6 寄存器列表

基地址：0x40004000

偏移地址	名称	描述	默认值
0x00	LVD_CR	LVD 控制寄存器	0x0000020
0x04	LVD_SR	LVD 状态寄存器	0x0000000

31.7 寄存器说明

31.7.1 LVD 控制寄存器(LVD_CR)

地址偏移：0x00
复位值：0x00000020

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INT_EN	HIGHINTEN	RISEINTEN	FALLINTEN	保留	ANA_FLT_EN	FLTCLK_SEL		FLTEN	ACT	LVDEN	保留	TH_VOLT_SEL			
R/W	R/W	R/W	R/W		R/W	R/W		R/W	R/W	R/W		R/W			

位	标记	功能描述	复位值	读写
31:16	FLT_NUM	LVD 采样滤波次数 采样时钟为 PCLK 时钟或者 LIRC，滤波计数值可配置。 采样次数达到滤波计数值时,结果一致输出。 采样计数周期 = FLT_NUM	0x0	R/W
15	INT_EN	LVD 中断使能 0：禁止 1：使能	0x0	R/W
14	HIGHINTEN	高电平触发使能 (被监测电压低于阈值电压) 0：禁止 1：使能	0x0	R/W
13	RISEINTEN	上升沿触发使能 (被监测电压从高于阈值电压变为低于阈值电压) 0：禁止 1：使能	0x0	R/W
12	FALLINTEN	下降沿触发使能 (被监测电压从低于阈值电压变为高于阈值电压) 0：禁止 1：使能	0x0	R/W
11	Res	保留	0x0	–
10	ANA_FLT_EN	LVD 模拟滤波使能 0：LVD 模拟滤波无效 1：LVD 模拟滤波使能有效	0x0	R/W
9:8	FLTCLK_SEL	滤波时钟选择 00：滤波时钟无效 01：滤波时钟选择为 PCLK(只能配置成中断模式) 10：滤波时钟选择为 LIRC（38.4KHz） 11：保留	0x0	R/W
7	FLTEN	数字滤波功能配置 0：禁止数字滤波 1：使能数字滤波	0x0	R/W
6	ACT	LVD 中断复位选择位 0：产生中断 1：产生复位	0x0	R/W
5	LVDEN	LVD 使能控制	0x1	R/W

		0：使能 LVD 功能 1：禁止 LVD 功能		
4	Res	保留	0x0	–
3:0	TH_VOLT_SEL	LVD 阈值电压选择 0000~0100：保留 0101：2.7V 0110：2.9V 0111：3.1V 1000：3.3V 1001：3.5V 1010：3.7V 1011：3.9V 1100：4.1V 1101：4.3V 1110：4.5V 1111：4.7V	0x0	R/W

31.7.2 LVD 状态寄存器(LVD_SR)

地址偏移：0x04
复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留														LVD_FLOUT	INTF
														RO	RCW0

位	标记	功能描述	复位值	读写
31:2	Res	保留	0x0	–
1	LVD_OUT	LVD输出 0：LVD输出结果为0 1：LVD输出结果为1 该标志为LVD比较器的输出经过数字滤波后的标志	0x0	RO
0	INTF	LVD 中断标志 0：未发生中断 1：发生 LVD 中断 写 0 清除中断标志，写 1 无效	0x0	RCW0

32 比较器(VC)

32.1 简介

芯片内嵌一个通用模拟电压比较器 VC，模拟电压比较器用于比较两个输入模拟电压的大小，并根据比较结果输出高/低电平。当”+”输入端电压高于”-”输入端电压时，电压比较器输出高电平；当”+”输入端电压低于”-”输入端电压时，电压比较器输出低电平。

32.2 主要特性

- 轨对轨比较器
- 可编程迟滞电压
- 可编程的速率和功耗
- 支持比较结果的滤波功能
- 输出端可以重新定向到一个 I/O 端口或多个定时器输入端，可以触发以下事件：
  - 捕获事件
  - Ocref_clr 事件(逐周期电流控制)
  - 为实现快速 PWM 关断的刹车事件
- 比较器可产生中断，并支持把 CPU 从睡眠模式和停机模式唤醒
- 支持三种软件可配置的中断触发方式：高电平触发/上升沿触发/下降沿触发；
- 支持内部 64 阶 VCC 分压(电压可选为 AVDD 或者内部参考 4V/2V)

32.3 比较器框图

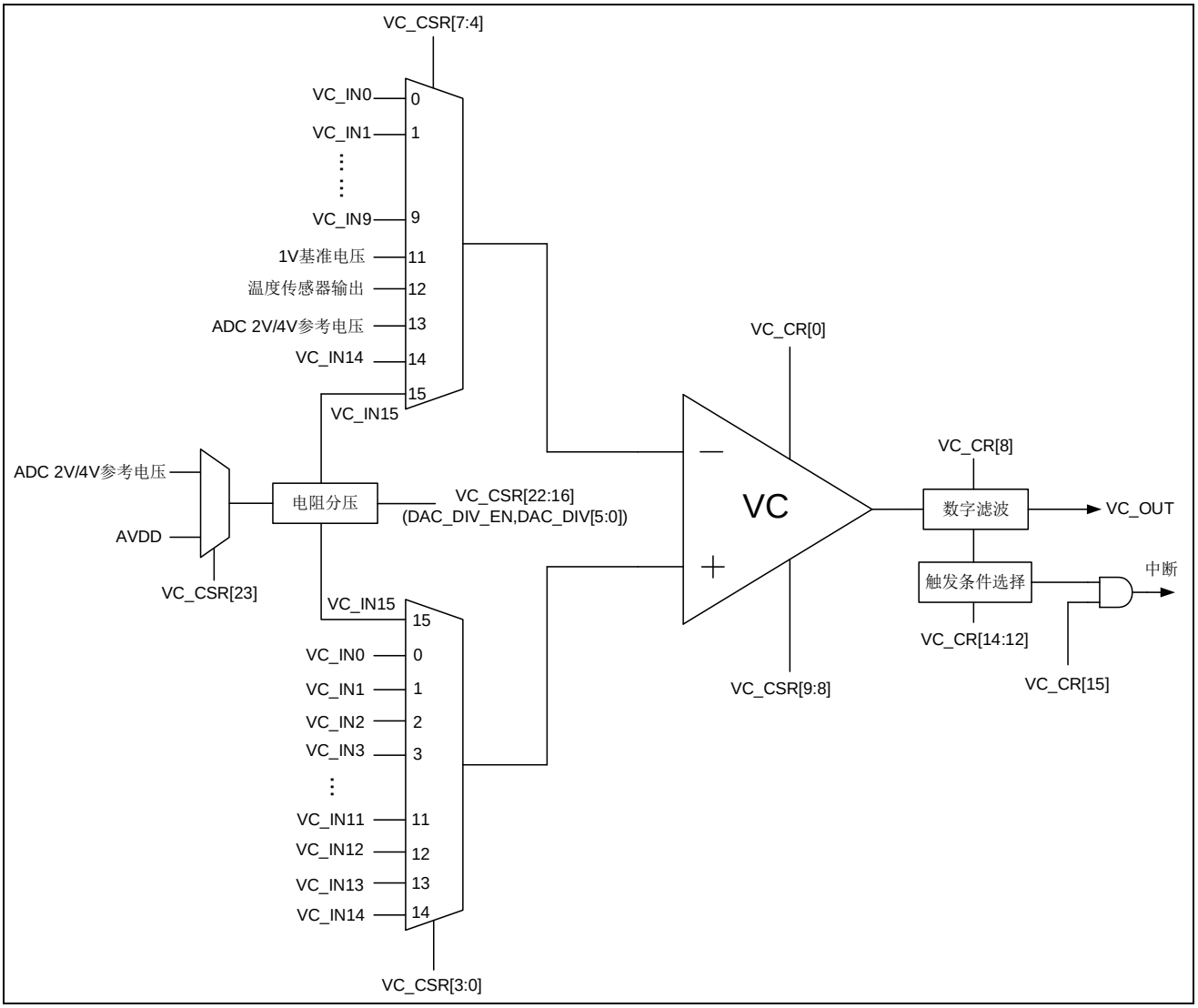


图32-1比较器结构框图

32.4 比较器开关控制

通过设置 VC_CR.VCEN 位可使比较器上电。设置 VCEN 位时，它将电压比较器从断电状态唤醒，清除 VC_CR.VCEN 位可停止比较器工作。

32.5 比较器输入和输出

作为比较器输入的 I/O 引脚必须在 GPIO 寄存器中设置为模拟模式。

比较器的输出可以内部重定向到各种定时器输入：

- 作为刹车输入，紧急关闭 PWM 信号
- 作为 OCREF_CLR，逐周期电流控制
- 作为输入捕获，测量时序

32.6 中断和唤醒

比较器的输出可以内部连接到外部中断和事件控制器。比较器能产生独立的中断或事件。详细内容可以参考手册的中断部分。

32.7 数字滤波

如果芯片的工作环境恶劣，迟滞比较器的输出会出现噪声信号。使能数字滤波模块，则迟滞比较器的输出波形中脉宽小于 VC_CR.FLT_NUM 的噪声信号都可以被滤除。禁止数字滤波模块，则数字滤波模块的输入输出信号相同。使能数字滤波模块，滤波示意如下所示：

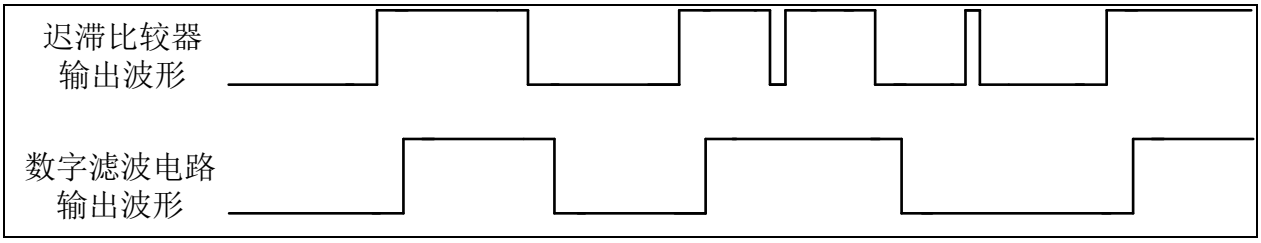


图 32-2 VC 滤波输出

32.8 迟滞功能

内置的电压比较器 VC 具有迟滞功能，其输出信号会等到输入信号高于或低于阈值电压 20mV 后才发生翻转。迟滞功能可以增强芯片的抗干扰能力，如下图所示。

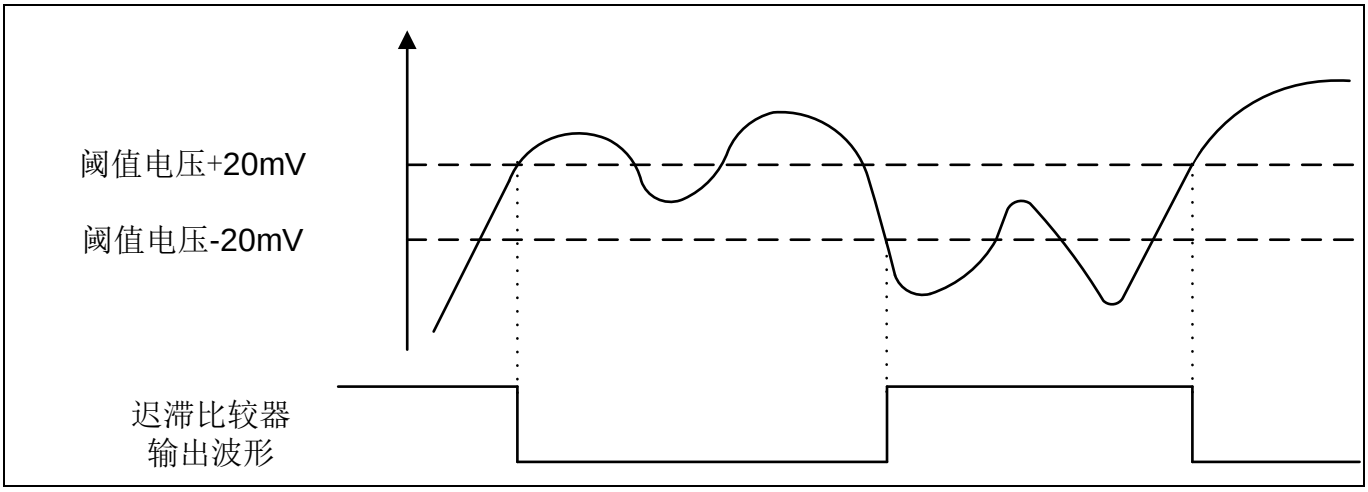


图 32-3 VC 迟滞响应

32.9 配置示例

若要使监测电压高于或低于阈值电压时产生中断。配置方法如下所示：

Step1：配置 VC_CSR.HYST，选择比较器迟滞电压。

Step2：配置 VC_CSR.NINSEL，选择“-”端待监测的电压来源。

Step3：配置 VC_CSR.PINSEL，选择“+”端待监测的电压来源。

Step4：配置 VC_CR.FLT_NUM，选择比较器滤波时间。

Step5：配置 VC_CR.FLTCLK_SEL, 选择滤波时钟。

Step6：配置 VC_CR.FLTEN，使能比较器滤波。

Step7：设置 VC_CR.HIGHINTEN、RISEINTEN、FALLINTEN，选择触发模式。

Step8：设置 VC_CR.INT_EN 为 1，使能比较器中断。

Step9：设置 VC_CR.VCEN 为 1，使能比较器。

Step10：在中断服务程序中向 VC_SR.INTF 写入 0 以清除中断标志。

32.10 寄存器列表

基地址：0x40004080

偏移地址	名称	描述	默认值
0x00	VC_CSR	VC控制和状态寄存器	0x0000000
0x04	VC_CR	VC控制寄存器	0x0000000
0x08	VC_OUTCFG	VC输出配置寄存器	0x0000000
0x0C	VC_SR	VC状态寄存器	0x0000000

32.11 寄存器说明

32.11.1 VC 电压控制和状态寄存器(VC_CSR)

地址偏移：0x00

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留								DAC_REF_SEL	DAC_DIV_EN	DAC_DIV					
								R/W	R/W	R/W					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留						HYST		NINSEL				PINSEL			
						R/W		R/W				R/W			

位	标记	功能描述	复位值	读写
31:24	Res	保留	0x0	–
23	DAC_REF_SEL	DAC_DIV参考电压Vref选择 0：AVDD 1：ADC参考电压	0x0	R/W
22	DAC_EN	6位DAC使能 0：禁止 1：使能	0x0	R/W
21:16	DAC_DIV	6位DAC配置 000000：1/64 Vref 000001：2/64 Vref 000010：3/64 Vref 000011：4/64 Vref … 111110：63/64 Vref 111111：Vref	0x0	R/W
15:10	Res	保留	0x0	-
9:8	HYST	比较器迟滞电压 00：0mV 01：10mV 10：20mV 11：30mV	0x0	R/W
7:4	NINSEL	比较器反相输入端电压选择: 0000：VC_IN0 0001：VC_IN1 0010：VC_IN2 0011：VC_IN3 0100：VC_IN4 0101：VC_IN5 0110：VC_IN6 0111：VC_IN7 1000：VC_IN8 1001：VC_IN9	0x0	R/W

		1010：保留 1011：1V基准电压 1100：温度传感器输出 1101：ADC 2V/4V内部参考电压 1110：VC_IN14 1111：分压电阻电压输出		
3:0	PINSEL	比较器同相输入端电压选择： 0000：VC_IN0 0001：VC_IN1 0010：VC_IN2 0011：VC_IN3 0100：VC_IN4 0101：VC_IN5 0110：VC_IN6 0111：VC_IN7 1000：VC_IN8 1001：VC_IN9 1010：VC_IN10 1011：VC_IN11 1100：VC_IN12 1101：VC_IN13 1110：VC_IN14 1111：分压电阻电压输出	0x0	R/W

32.11.2 VC 控制寄存器(VC_CR)

地址偏移：0x04

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INT_EN	HIGHINTEN	RISEINTEN	FALLINTEN	保留			FLTEN	保留				VC_FLTCLK_SEL		保留	VCEN
R/W	R/W	R/W	R/W				R/W					R/W			R/W

位	标记	功能描述	复位值	读写
31:16	FLT_NUM	比较器采样滤波次数 采样时钟为PCLK时钟或者LIRC，滤波计数值可配置。采样次数达到滤波计数值时，结果一致输出。 采样计数周期 = FLT_NUM	0x0	R/W
15	INT_EN	VC中断使能 0：禁止 1：使能	0x0	R/W
14	HIGHINTEN	高电平触发使能 0：禁止 1：使能	0x0	R/W
13	RISEINTEN	上升沿触发使能 0：禁止 1：使能	0x0	R/W
12	FALLINTEN	下降沿触发使能 0：禁止 1：使能	0x0	R/W
11:9	Res	保留	0x0	–
8	FLTEN	数字滤波功能配置 0：禁止数字滤波 1：使能数字滤波	0x0	R/W
7:4	Res	保留	0x0	–
3:2	FLTCLK_SEL	VC滤波时钟选择 00：滤波时钟无效 01：滤波时钟选择为PCLK 10：滤波时钟选择为LIRC 11：保留	0x0	R/W
1	Res	保留	0x0	–
0	VCEN	VC Enable 0：电压比较功能禁止 1：电压比较功能使能	0x0	R/W

32.11.3 VC 输出配置寄存器(VC_OUTCFG)

地址偏移：0x08

复位值：0x00000000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
保留													INV_PAD	TIMxBKE	TIMxCH3_EN
													R/W	R/W	R/W

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INV_TIMxCH4	TIMxCH3_EN	INV_TIMxCH3	TIMxCH2_EN	INV_TIMxCH2	TIMxCH1_EN	INV_TIMxCH1	PCAECI_EN	PCACAP0_EN	INV_PCA	LPTIMEXT_EN	LPTIM_EN	保留	TIM11_EN	TIM10_EN	INV_TIMX
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:19	Res	保留	0x0	–
18	INV_PAD	VC结果输出到PAD反向使能 0：禁止；1：使能	0x0	R/W
17	TIMxBKE	VC中断作为TIM1/TIM2刹车控制 0：禁止；1：使能	0x0	R/W
16	TIMxCH4_EN	VC结果输出到TIM1ch4/TIM2ch4使能 0：禁止；1：使能	0x0	R/W
15	INV_TIMxCH4	VC结果输出到TIM1ch4/TIM2ch4反向使能 0：禁止；1：使能	0x0	R/W
14	TIMxCH3_EN	VC结果输出到TIM1ch3/TIM2ch3使能 0：禁止；1：使能	0x0	R/W
13	INV_TIMxCH3	VC结果输出到TIM1ch3/TIM2ch3反向使能 0：禁止；1：使能	0x0	R/W
12	TIMxCH2_EN	VC结果输出到TIM1ch2/TIM2ch2使能 0：禁止；1：使能	0x0	R/W
11	INV_TIMxCH2	VC结果输出到TIM1ch2/TIM2ch2反向使能 0：禁止；1：使能	0x0	R/W
10	TIMxCH1_EN	VC结果输出到TIM1ch1/TIM2ch1使能 0：禁止；1：使能	0x0	R/W
9	INV_TIMxCH1	VC结果输出到TIM1ch1/TIM2ch1反向使能 0：禁止；1：使能	0x0	R/W
8	PCAECI_EN	VC结果输出到PCA外部时钟使能 0：禁止；1：使能	0x0	R/W
7	PCACAP0_EN	VC结果输出到PCA捕获0使能 0：禁止；1：使能	0x0	R/W
6	INV_PCA	VC结果输出到PCA反向使能 0：禁止；1：使能	0x0	R/W
5	LPTIMEXT_EN	VC结果输出到LPTIM外部时钟使能控制 0：禁止；1：使能 注：LPTIM外部时钟使能控制指代的为VC out作为LPTIM EXT_IN信号使用，同时调用此功能需要配置SYSCON_PORTCR.LPTIM_EXT_SEL	0x0	R/W
4	LPTIM_EN	VC结果输出到LPTIM门控使能	0x0	R/W

		0：禁止；1：使能		
3	Res	保留	0x0	–
2	TIM11_EN	VC结果输出到TIM11门控使能 0：禁止；1：使能	0x0	R/W
1	TIM10_EN	VC结果输出到TIM10门控使能 0：禁止；1：使能	0x0	R/W
0	INV_TIMX	VC结果输出到TIM10，TIM11，LPTIM门控反向使能 0：禁止；1：使能 注：反向使能TIM10,TIM11,LPTIM门控，需要同时开启TIM10或TIM11或LPTIM门控使能	0x0	R/W

32.11.4 VC 状态寄存器(VC_SR)

地址偏移：0x0C

复位值：0x0000 0000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留														VC_OUT	INTF
														RO	RCW0

位	标记	功能描述	复位值	读写
31:2	Res	保留	0x0	–
1	VC_OUT	VC 模块的输出值： 0：VC 输出为 0 1：VC 输出为 1	0x0	RO
0	INTF	VC 中断标志： 0：未发生中断 1：发生 VC 中断 写 0 清除中断标志，写 1 无效	0x0	RCW0

33 可编程运算放大器(PGA)

33.1 简介

芯片内部集成 1 路可编程运算放大器，增益可调范围 X1~X50，可有效放大交流信号与直流信号，PGA 输出 PGA_OUT 与 ADC 通道 21 连接，PGA 输入 PGA_INP 与 PB4 引脚相连。

33.2 运算放大器框图

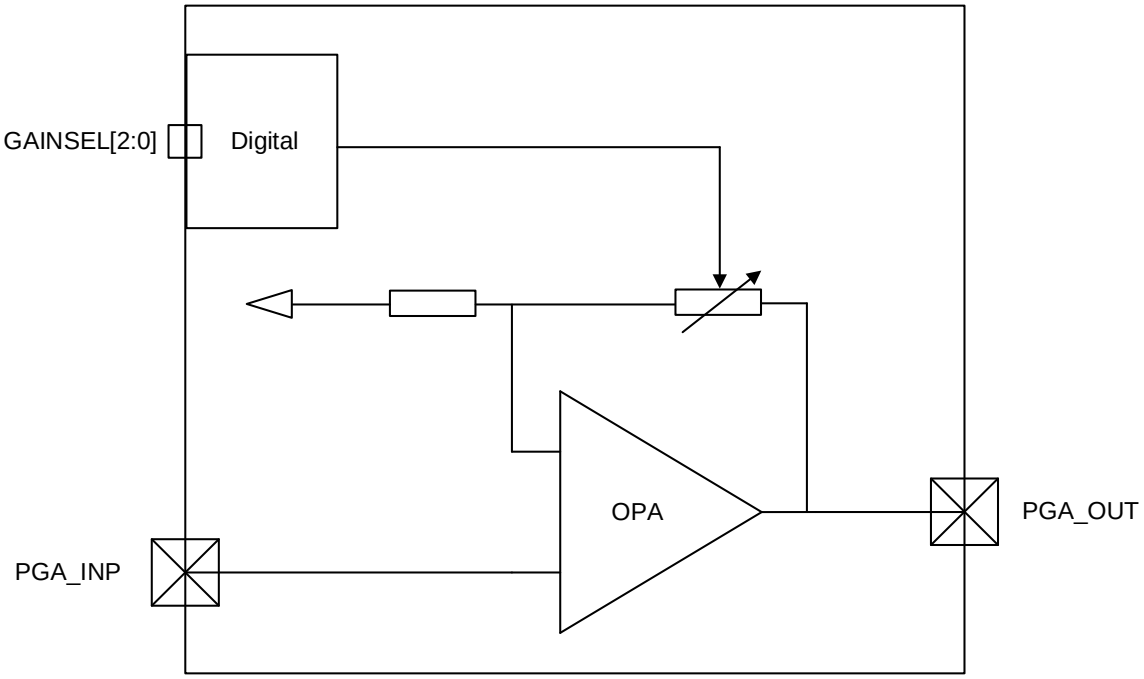


图 33-1 运算放大器框图

33.3 配置流程

1. 配置 GPIO 端口复用功能寄存器 GPIOx_AFR1 与 GPIOx_AFR2，将 PGA 输入端口对应的 IO 配置为高阻输入/模拟端口功能；
2. 配置 PGA_CR 寄存器，选择增益范围，打开 PGA 工作使能。

33.4 寄存器列表

基地址：0x40004000

偏移地址	名称	描述	默认值
0xB0	PGA_CR	可编程运算放大器控制寄存器	0x0000000

33.5 寄存器说明

33.5.1 可编程运算放大器控制寄存器(PGA_CR)

地址偏移：0xB0

复位值：0x00000001

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

位	标记	功能描述	复位值	读写
31:5	Res	保留	0x0	–
4:2	PGA_GAINSEL	PGA放大增益选择 000：1 001：10 010：20 011：30 100：40 101：50 110~111：保留	0x0	R/W
1	PGA_OEN	PGA输出使能 0：PGA输出关闭 1：PGA输出打开(PC3配置成模拟口，通过PC3端口输出)	0x0	R/W
0	PGA_DIS	PGA掉电使能 0：使能PGA 1：禁止PGA	0x1	R/W

34 Debug 支持

为了方便用户调试，本产品部分外设支持在 SWD 调试模式下控制其工作状态，具体通过寄存器 DBG_APBFZ 可进行配置。

34.1 寄存器列表

基地址：0x40004C00

表 34-1 Debug 寄存器映象和复位值

偏移地址	名称	描述	默认值
0x00	DBG_APBFZ	Debug 模式控制寄存器	0x00000000

34.2 Debug 模式控制寄存器(DBG_APBFZ)

地址偏移：0x00

复位值：0x00000000

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留				TIM2DBGSTOP	WWDGDBGSTOP	IWDGDBGSTOP	BEEPDBGSTOP	保留	RTCDBGSTOP	TIM1DBGSTOP	PCADBGSTOP	保留	LPTIMDBGSTOP	TIM11DBGSTOP	TIM10DBGSTOP
				R/W	R/W	R/W	R/W		R/W	R/W	R/W		R/W	R/W	R/W

位	标记	功能描述	复位值	读写
31:16	Key	只有高位写 0x5A69 时配置该寄存器才有效，写其它值时无效。	0x0	WO
15:12	Res	保留	0x0	-
11	TIM2DBGSTOP	TIM2 调试模式停止工作 0：调试模式下计数器仍然工作 1：调试模式下计数器停止工作	0x0	R/W
10	WWDGDBGSTOP	WWDG 调试模式停止工作 0：调试模式计数器仍然工作 1：调试模式计数器停止工作	0x0	R/W
9	IWDGDBGSTOP	IWDG 调试模式停止工作 0：调试模式下计数器仍然工作 1：调试模式下计数器停止工作	0x0	R/W
8	BEEPDBGSTOP	BEEP 调试模式停止工作 0：调试模式计数器仍然工作 1：调试模式计数器停止工作	0x0	R/W
7	AWKDBGSTOP	AWK 调试模式停止工作 0：调试模式计数器仍然工作 1：调试模式计数器停止工作	0x0	R/W
6	RTCDBGSTOP	RTC 调试模式停止工作 0：调试模式计数器仍然工作 1：调试模式计数器停止工作	0x0	R/W
5	TIM1DBGSTOP	TIM1 调试模式停止工作 0：调试模式下计数器仍然工作 1：调试模式下计数器停止工作	0x0	R/W
4	PCADBGSTOP	PCA 调试模式停止工作 0：调试模式计数器仍然工作 1：调试模式计数器停止工作	0x0	R/W
3	Res	保留	0x0	-
2	LPTIMDBGSTOP	Low Power Timer 调试模式停止工作 0：调试模式下计数器仍然工作 1：调试模式下计数器停止工作	0x0	R/W
1	TIM11DBGSTOP	TIM11 调试模式停止工作 0：调试模式下计数器仍然工作 1：调试模式下计数器停止工作	0x0	R/W
0	TIM10DBGSTOP	TIM10 调试模式停止工作 0：调试模式下计数器仍然工作 1：调试模式下计数器停止工作	0x0	R/W

35 工作模式和电源管理

CPS32K11x 的电源管理模块负责管理本产品各种工作模式之间的切换，以及控制各工作模式下的各功能模块的工作状态。本产品的工作电压(VCC)为 2.7 ~ 5.5V。本产品有如下几个工作模式：

1) 运行模式：CPU 运行，周边功能模块运行。
2) 休眠模式：CPU 停止运行，周边功能模块运行。
3) 深度休眠模式：CPU 停止运行，高速时钟停止运行。

从运行模式，通过执行软件程序，可进入其他低功耗模式。从其他各种低功模式，通过中断触发，可回到运行模式。

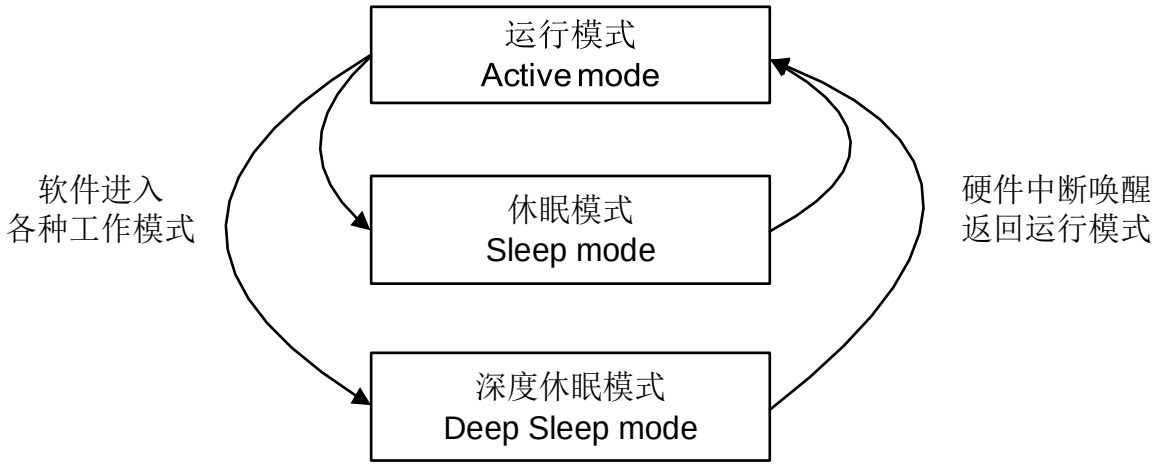


图 35-1 控制模式框图

在各种模式下，CPU 可响应的中断类型请参考**错误!未找到引用源。**。

各模式下，可以响应的复位源类型

编号	复位源	运行模式	Sleep 模式唤醒	Deep Sleep 模式唤醒
1	上电/掉电复位	Y	Y	Y
2	外部 Reset Pin 复位	Y	Y	Y
3	IWDG 复位	Y	Y	Y
4	WWDG 复位	Y	Y	N
5	系统软件复位	Y	N	N
6	欠电压(LVD)复位	Y	Y	Y
7	LOCKUP 复位	Y	N	N
8	寄存器 CPURST 复位	Y	N	N
9	寄存器 MCURST 复位	Y	N	N

注：Y 表示可以响应，N 表示不可以响应；

35.1 运行模式(Active Mode)

系统在电源上电复位后，或从各低功耗模式唤醒后，微控制器 MCU 处于运行状态，各模块的运行状态如**错误!未找到引用源。**所示。当 CPU 不需继续运行时，可以利用多种低功耗模式来节能。用户需要根据最低能耗、最快速启动时间、可用的唤醒源等条件，选定一个最佳的低功耗模式。

图 35-2 运行模式下可运行模块一览

几种降低运行模式下芯片功耗的方法：

1) 在运行模式下，通过对预分频寄存器(RCC_HCLKDIV, RCC_PCLKDIV)进行编程，可以降低任意一个系统时钟(HCLK，PCLK)的速度。进入睡眠模式前，也可以利用预分频器来降低外设的时钟。
2) 在运行模式下，关闭不使用外设的时钟(RCC_HCLKEN,RCC_PCLKEN)来减少功耗。

35.2 休眠模式(Sleep Mode)

使用 WFI 指令可以进入休眠模式，休眠模式下，CPU 停止运行，但系统时钟、NVIC 中断处理以及非 HCLK 驱动的周边功能模块仍都可以工作。

系统进入休眠状态，不会改变端口状态，在进入休眠前请根据需要更改 IO 的状态为休眠模式下的状态。

如何进入休眠模式：

通过执行 WFI 指令进入睡眠状态。根据 Cortex®-M0+系统控制寄存器中的 SleepONEXIT 位的值，有两种选项可用于选择睡眠模式进入机制：

1) ：如果 SleepONEXIT=0，当 WFI 或 WFE 被执行时，微控制器立即进入睡眠模式。
2) ：如果 SleepONEXIT=1，系统从最低优先级的中断处理程序中退出时，微控制器就立即进入睡眠模式。

如何退出休眠模式：

如果执行 WFI 指令进入睡眠模式，任意一个高优先级嵌套向量中断控制器响应的外设中断都能将系统从睡眠模式唤醒。

使用注意：

- Sleep-ON-EXIT=1，执行完中断自动进入 Sleep，程序不需要写__wfi()；
- Sleep-ON-EXIT=0，执行完中断自动进入运行模式，执行 WFI 指令后进入 Sleeping。等待后续中断触发。
- 若在中断中进入 Sleeping 后，只有优先级高于此中断的中断才能唤醒，先执行高优先级，再执行低优先级；优先级低于或等于当前中断的中断不能唤醒。

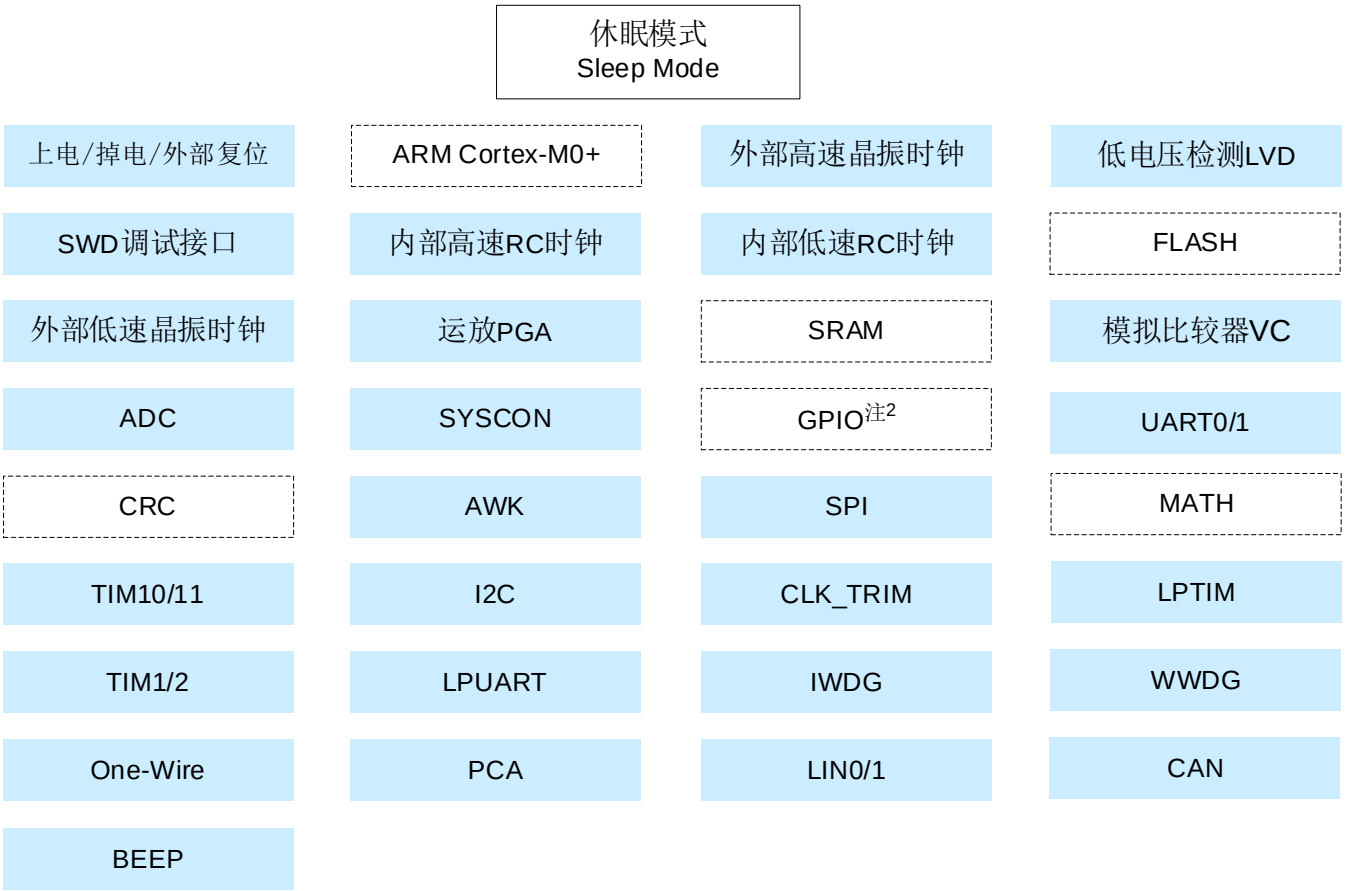


图 35-3 休眠模式下可运行模块一览

35.3 深度休眠模式(Deep Sleep Mode)

使用 SleepDEEP 配合 WFI 指令可以进入深度休眠模式，在深度休眠模式下，CPU 停止运行，高速时钟关闭，低速时钟可配置是否运行，部分低功耗的周边模块可配置为是否运行，NVIC 中断处理仍可以工作。

- 系统从高速时钟进入深度休眠模式，高速时钟自动关闭，低速时钟保持进入深度睡眠前的状态。
- 系统从低速时钟进入深度休眠模式，低速时钟保持运行，除了低功耗模块可以运行，其他模块自动关闭。
- 系统时钟切换时，所有时钟都不会自动关闭，需要根据功耗及系统需求软件关闭打开相应的时钟。
- 系统进入深度休眠状态，不会改变端口状态，在进入深度休眠前根据需要更改 IO 的状态为深度休眠模式下的状态。

如何进入深度休眠模式：

首先设置 Cortex®-M0+系统控制寄存器中的 SleepDEEP 位，通过执行 WFI 指令进入深度睡眠状态。根据 Cortex®-M0+系统控制寄存器中的 SleepONEXIT 位的值，有两种选项可用于选择深度睡眠模式进入机制：

1) ：如果 SleepONEXIT=0，当 WFI 或 WFE 被执行时，微控制器立即进入睡眠模式。

2) ：如果 SleepONEXIT =1，系统从最低优先级的中断处理程序中退出时，微控制器就立即进入睡眠模式。

如何退出深度休眠模式：

如果执行 WFI 指令进入睡眠模式，任意一个可唤醒中断(Deep Sleep 模式下可运行的周边模块中断)都能将系统从睡眠模式唤醒，可唤醒中断参考表 3-1。

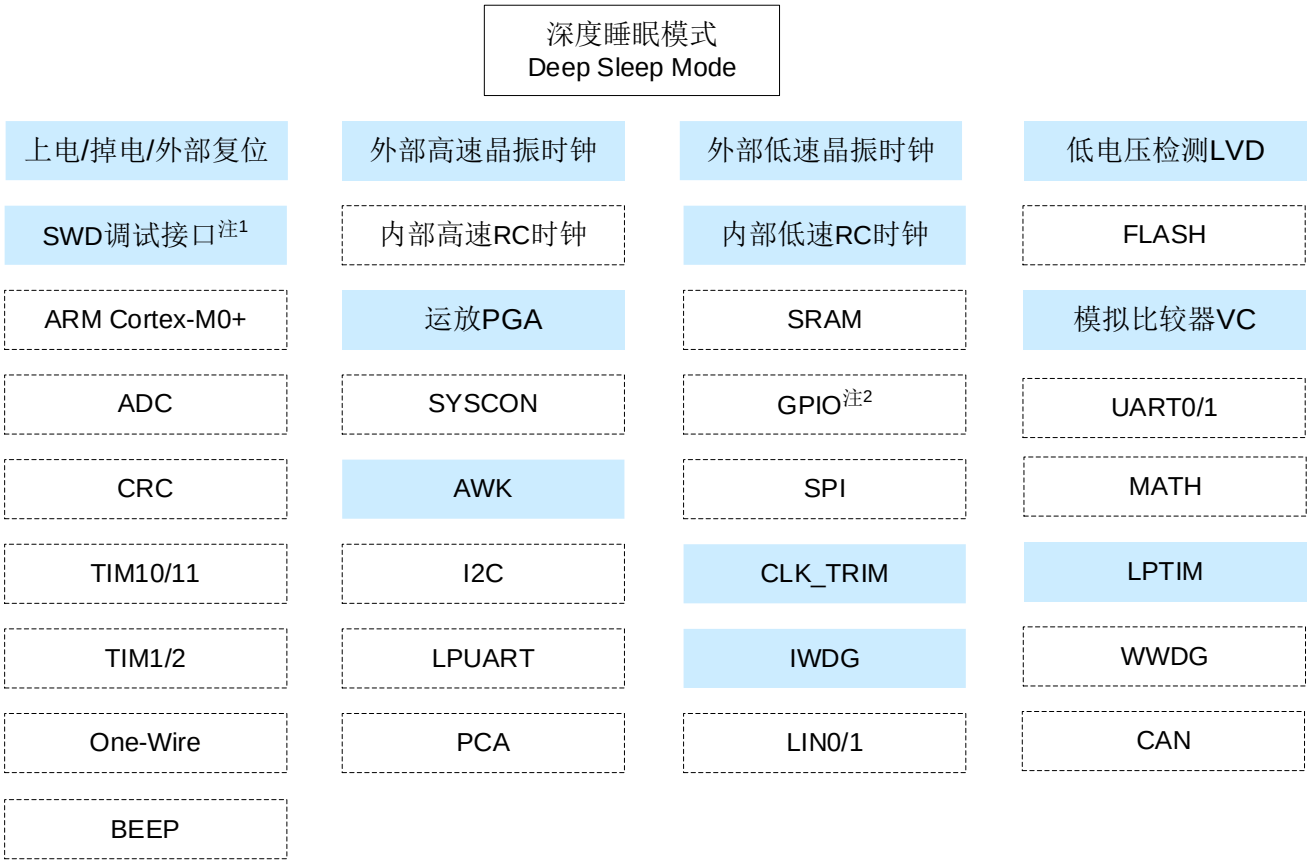


图 35-4 深度睡眠模式下可运行模块一览

注 1：在 Deep Sleep 模式，芯片重新复位后，可以通过 SWD 接口唤醒

注 2：GPIO 状态不可以改变但是可以用来作为中断唤醒源

进入深度休眠后，唤醒后系统时钟有两种选择，默认使用进入深度休眠的时钟，配置寄存器 RCC_SYSCLKCR.WKBYHIRC 为 1 后不管进入深度休眠前是什么时钟，唤醒后都使用内部高速时钟 HIRC。如果进入深度睡眠前系统使用外部晶体振荡这样设置可以加速系统唤醒。

36 唯一器件标志码(UID)

唯一器件标识码(UID)长度为 128bit，存储在 Flash 模块的存储区中，可以使用原厂提供的静态库函数对其进行读取。

36.1 寄存器列表

偏移地址	名称	描述	默认值
0x00	UID0	UID寄存器0，包含UID[31:0]	0xXXXXXXXX
0x04	UID1	UID寄存器1，包含UID[63:32]	0xXXXXXXXX
0x08	UID2	UID寄存器2，包含UID[95:64]	0xXXXXXXXX
0x0C	UID3	UID寄存器3，包含UID[127:96]	0xXXXXXXXX

36.2 寄存器说明

36.2.1 UID 寄存器 0(UID0)

地址偏移：0x00
复位值：0xXXXXXXXX(X 为出厂设定)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UID0															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UID0															
RO															

位	标记	功能描述	复位值	读写
31:0	UID0	UID寄存器0，包含UID[31:0] UID[31:0]为生产批次编号(Lot ID)	0xXXXXXXXX	RO

36.2.2 UID 寄存器 1(UID1)

地址偏移：0x04
复位值：0xXXXXXXXX(X 为出厂设定)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UID1															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UID1															
RO															

位	标记	功能描述	复位值	读写
31:0	UID1	UID寄存器1，包含UID[63:32] UID[39:32]为晶圆编号(Wafer number)(8位无符号数); UID[55:40]为晶圆横坐标(Row address); UID[63:56]为晶圆纵坐标低8位(Column address)	0xXXXXXXXX	RO

36.2.3 UID 寄存器 2(UID2)

地址偏移：0x08
复位值：0xXXXXXXXX(X 为出厂设定)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UID2															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UID2															
RO															

位	标记	功能描述	复位值	读写
31:0	UID2	UID寄存器2，包含UID[95:64] UID[71:64]为晶圆纵坐标高8位(Column address) UID[95:72]为晶圆测试日期低24位(Test date)	0xXXXXXXXX	RO

36.2.4 UID 寄存器 3(UID3)

地址偏移：0x0C
复位值：0xXXXXXXXX(X 为出厂设定)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UID3															
RO															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UID3															
RO															

位	标记	功能描述	复位值	读写
31:0	UID3	UID寄存器3，包含UID[127:96] UID[103:96]为晶圆测试日期高8位(Test date) UID[127:104]为特殊标识码(Special ID)	0xXXXXXXXX	RO

37 修订记录

版本	变更内容	修改时间
Ver0.5	初版对外发布	2023/July
Ver0.6	1. Page62, 7.4.6 RSTKEY 说明误记修正。RSTCON 为原来名字的误记。 2. Page39, 5.6.7 ECCCR 的 ADDR_ECCERR 说明的误记修正，不需要加上偏移量。 3. 产品特性中加入 20TSSOP 封装	2023/9
Ver0.7	4. Page18, 2.产品特性中，内部 VREF 精度根据 ATE 卡控结果来公开 5. Page507, ADC 配置寄存器 0(ADC_CR0)说明修改，用内部输出作为 ADC 输入通道，ADC_CR0.BUF_EN 的值误记修正	2023/12
Ver0.8	删除 TSSOP20 相关信息	2023/12/18