保密级别：外部使用

UM324xF 用户手册

版本：V1.9.1

广芯微电子（广州）股份有限公司

http://www.unicmicro.com/

条款协议

目录

表目录

图目录

1 文档约定

1.1 寄存器相关缩写词列表

寄存器说明中使用以下缩写词：

可读可写（R/W）	软件可以读写该位
只读（R）	软件只能读取该位
只写（W）	软件只能写入该位，软件读返回复位值
可读，写 1 清零（R/W1C）	软件可以读取该位，写 1 清除该位，写 0 无效
可读，写 0 清除（R/W0C）	软件可以读取该位，写 0 清除该位，写 1 无效

1.2 词汇表

本节简要介绍本文档中所用首字母缩略词和缩写词的定义：

- 在本文档中，将具有 FPU 的 Cortex-M4 内核称为 Cortex-M4F。
- CPU 内核集成了两个调试端口：
  - JTAG 调试端口 (JTAG-DP) 提供基于联合测试工作组 (JTAG) 协议的 5 引脚标准接口。
  - SWD 调试端口 (SWD-DP) 提供基于串行线调试 (SWD) 协议的 2 引脚（时钟和数据）接口。
- 字：32 位数据/指令。
- 半字：16 位数据/指令。
- 字节：8 位数据。
- 双字：64 位数据。
- IAP（在应用中编程）：IAP 是指可以在用户程序运行期间对微控制器的–Flash–进行重新编程。
- ICP（在线编程）：ICP 是指可以在器件安装于用户应用电路板上时使用–JTAG–协议、SWD 协议或自举程序对微控制器的–Flash–进行编程。
- I-Code：此总线用于将 CPU 内核的指令总线连接到 Flash 指令接口。通过此总线可执行预取操作。
- D-Code：此总线用于将 CPU 的 D-Code–总线（数据加载和调试访问）连接到 Flash 数据接口。
- 选项字节：存储于 Flash 中的产品配置位。
- OBL：选项字节加载器。
- AHB：高级高性能总线。
- CPU：指 Cortex-M4F 内核。

2 产品简介

2.1 系统概述

UM324xF 系列芯片是高性能低功耗的 ARM® Cortex® -M4 核的通用微处理器芯片。其中，M4 处理器核支持 DSP 指令集，有浮点运算单元（FPU）、内存保护单元（MPU）。系统最高工作时钟频率 204MHz（240MHz @boost mode）。内置 FLASH 最大容量为 512KB，SRAM 最大容量为 160KB，DMA SRAM 容量为 32KB。支持 1.8 ~ 3.6V 宽电压供电，适宜在-40℃至 105℃（工业级）的温度下工作。

UM324xF 系列芯片带有丰富外设，包括 1 个 USB FS Device 接口、1 个 SDIO/SD/eMMC 接口、1 个 10/100M/1000M 以太网接口、1 个外部高速存储接口（EMC）、2 个 12 位的高速 ADC、2 个 12 位的 DAC、内置温度传感器、3 个比较器、2 个运算放大器、2 个 USART、6 个 UART、4 个 SPI 接口、3 个 I2C 接口、2 个 I2S 接口、2 个看门狗定时器、2 个 CAN 总线接口、1 个 QSPI 接口、14 个计数器/定时器（高级控制器和通用定时器）、1 个低功耗串口（LPUART）、2 个低功耗计时器（LPTIMx）、1 个 32 位 RTC 时钟及计数器、高达 82 通道的通用输入输出口、支持数字摄像头接口（DCMI）和 I8080 LCD 显示接口、同时集成硬件 Cordic 模块（支持 sin、cos、arctan、平方根、乘除法等）、内建加解密引擎（包含 AES、SHA 等）、1 个可产生随机 Key 的随机数产生器（RNG）。

适用场景：

- 电机应用
- 二维码识别
- 智能门锁
- 网关
- 打印机
- Sensor Hub
- 物联网应用等

2.2 主要特性

- **高性能的 ARM 架构 Cortex™-M4F 处理器**
  - 主频高达 204MHz（240MHz @boost mode）
  - 内存保护单元（MPU）
  - 16 级中断优先级
  - IEEE 754 兼容浮点运算单元（FPU）
  - DSP 指令扩展，带硬件除法器和 32 位单周期硬件乘法器
  - 512KB Flash 和 160KB SRAM+4KB 备份 SRAM+32KB SRAM2
  - 2 个 DMA 控制器，总共 16 通道
  - 支持 Serial Wire Debug (SWD)和 JTAG 在线调试
- **内嵌系统引导程序（bootloader）**
  - 支持 Flash 存储器 ISP 和 IAP
- **灵活的时钟单元**
  - 外部晶振的输入范围 1MHz 到 48MHz
  - 内部 48MHz 高速振荡器
  - 内部低速 32kHz 振荡器
  - 内置 PLL（支持整数，小数，扩频）
  - 外部晶体支持 32.768kHz，支持 32K 实时时钟(RTC)
- **模拟外设**
  - 2 个 12 位逐次型 ADC 转换器，总共 16 个外部通道，支持单端和差分输入
  - 2 个 12 位 1Msps DAC 转换器，带 Buffer
  - 3 个模拟比较器(ACMP)
  - 2 个运算放大器(OPA)，支持 PGA 和比较器功能
  - 内置参考电压源（VREF）
  - 内置温度传感器(TS)
- **硬件加速运算协处理器**
  - Cordic 加速器：支持$m \cdot \sin\theta$、$m \cdot \cos\theta$、atan2(y,x)、$\sqrt{x^2+y^2}$、$y \cdot x$、y/x、$\sinh w$、$\cosh w$、$\tanh^{-1}(y/x)$、ln(x)、$\sqrt{x}$ 等函数运算
- **丰富的通信接口**
  - 6 个 UART 接口（最高支持 10.5Mbps），支持 IrDA
  - 2 个 USART 接口，支持 UART/SPI/IrDA/LIN
  - 1 个低功耗串口（LPUART）
  - 3 个 I2C 接口(最高支持 1 Mbps)，支持 SMBus

- 2 个 I2S 接口（支持 I2S/PCM 格式）
- 4 个通用的 SPI 接口
- 1 个 QSPI 接口
- 2 个 CAN 工业总线，支持 CAN2.0B
- 1 个 48MHz 的 SDIO/SD/eMMC 接口
- 1 个 10M/100M/1000M 以太网 RGMII/RMII/MII 接口
- 1 个全速 USB20 Device 接口（带 PHY）

- **外部存储接口**
  - 1 个 EMC 高速存储控制器，支持 SRAM/Nor Flash 接口
- **加密引擎**
  - 支持 AES 128/256
  - 支持 SHA 256
  - 一个随机数产生器（RNG）能产生随机 Key
- **定时器**
  - 2 个 32 位高级定时器(eQCT™：TIM0 和 TIM7)，32bit 自动重载计数器及一个可编程预分频器。支持捕获、输出比较、PWM 输出（带死区插入的互补 PWM）。每个高级定时器支持 4 路 PWM 输出，总共 8 路 PWM 输出。
  - 10 个 32 位通用定时器（TIM1~4, TIM8~13），每个支持 4 路 PWM 输出，总共 40 路 PWM 输出。
  - 2 个 32 位普通定时器（TIM5 和 TIM6）
  - 2 个 16 位低功耗计时器（LPTIM0~1），总共支持 2 路 PWM 输出
  - 1 个 24 位的 Systick 定时器
- **实时时钟（RTC）**
  - 支持硬件万年历（秒、分钟、小时、星期几、日期、月份和年份显示），支持校准功能
- **通用输入/输出端口**
  - 高达 82 个通用输入/输出管脚
  - 58 个管脚支持 5V 耐压
- **视频接口**
  - 支持 I8080 接口的 TFT LCD 显示控制器
  - 支持数字摄像头接口（DCMI 接口）
- **电源管理**
  - 多种省电模式：睡眠模式（Sleep）、停止模式（Stop）、待机模式（Standby）、深度待机模式（DeepStandby）
  - 支持欠压检测（BOR）和低电压检测（LVD），两组检测点可分别产生欠压中断和强制复位

- 支持上电复位（POR）和掉电复位（PDR）
- 集成的电源管理单元（PMU）

- **电气参数**
  - 工作电压：1.8 ~ 3.6V
  - 工作温度：-40 ~ +105°C
- **安全**
  - 防抄板设计，防止 eFlash 中程序被盗取
  - code CRC16-CCITT/CRC32 数据校验算法硬件加速
  - 128 位全球唯一芯片序列号 ID

3 存储器和总线架构

3.1 系统架构

主系统由 32 位多层 AHB 总线矩阵构成，可实现以下部分的互连：

- **8 条主控总线：**
  - Cortex™-M4F 内核 I 总线、D 总线和 S 总线
  - DMA0 存储器总线
  - DMA1 存储器总线
  - SDIO 的 DMA 总线
  - 以太网的 DMA 总线
  - DCMI 控制器 DMA 总线
- **12 条被控总线：**
  - FLASH NVR
  - 内部 Flash I Code 总线
  - 内部 Flash D Code 总线
  - 主要内部 SRAM0 (128 KB)
  - 辅助内部 SRAM1 (32 KB)
  - DMA SRAM2 (32 KB)
  - AHB0 外设
  - AHB1 外设
  - AHB2APB1 外设
  - AHB2APB2 外设
  - EMC 外设
  - QSPI

3.2 总线架构图

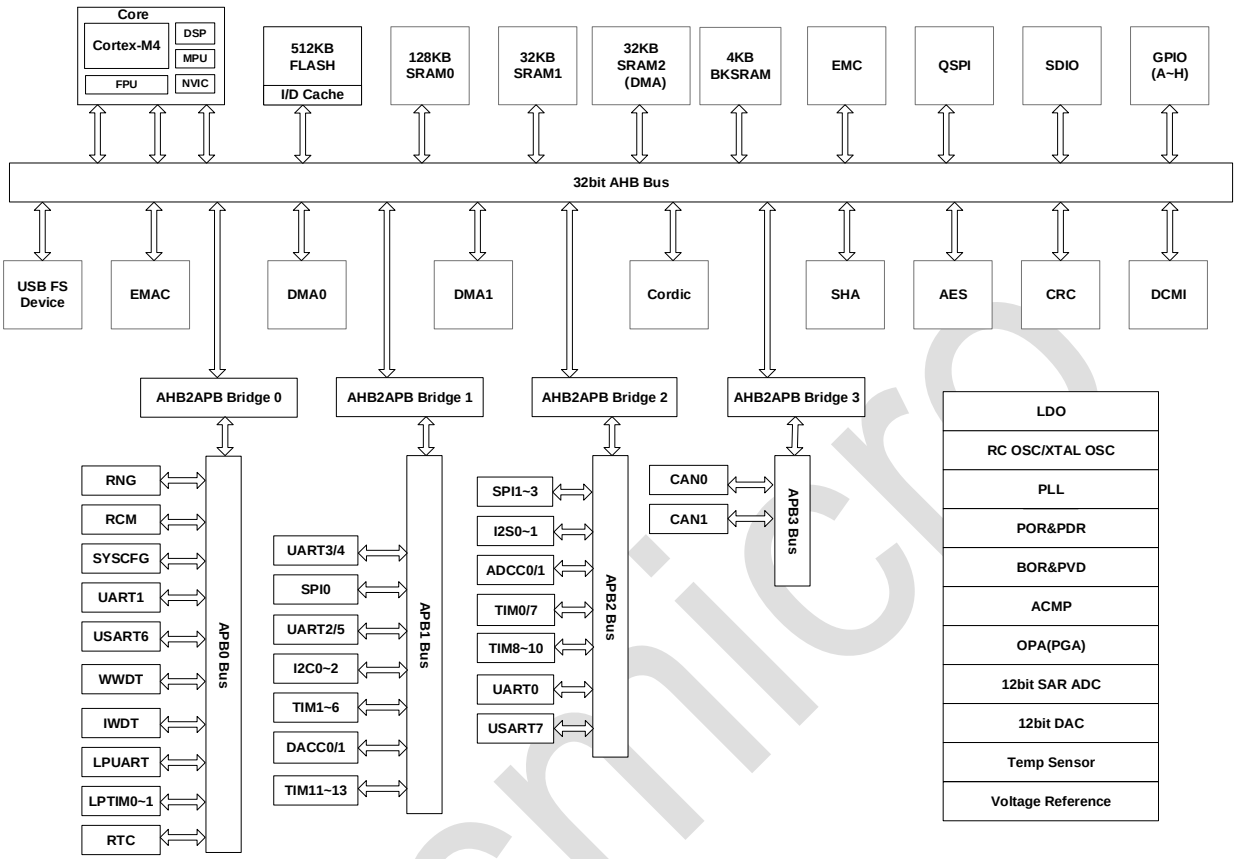


图 3-1：总线架构图

3.3 存储器映射（Memory Mapping）

程序存储器、数据存储器、寄存器和 I/O 端口排列在同一个顺序的 4GB 地址空间内。

各字节按小端格式在存储器中编码。字中编号最低的字节被视为该字的最低有效字节，而编号最高的字节被视为最高有效字节。

有关外设寄存器映射的详细信息，请参见相关章节。

未分配给片上存储器和外设的所有存储区域均视为“保留区”。

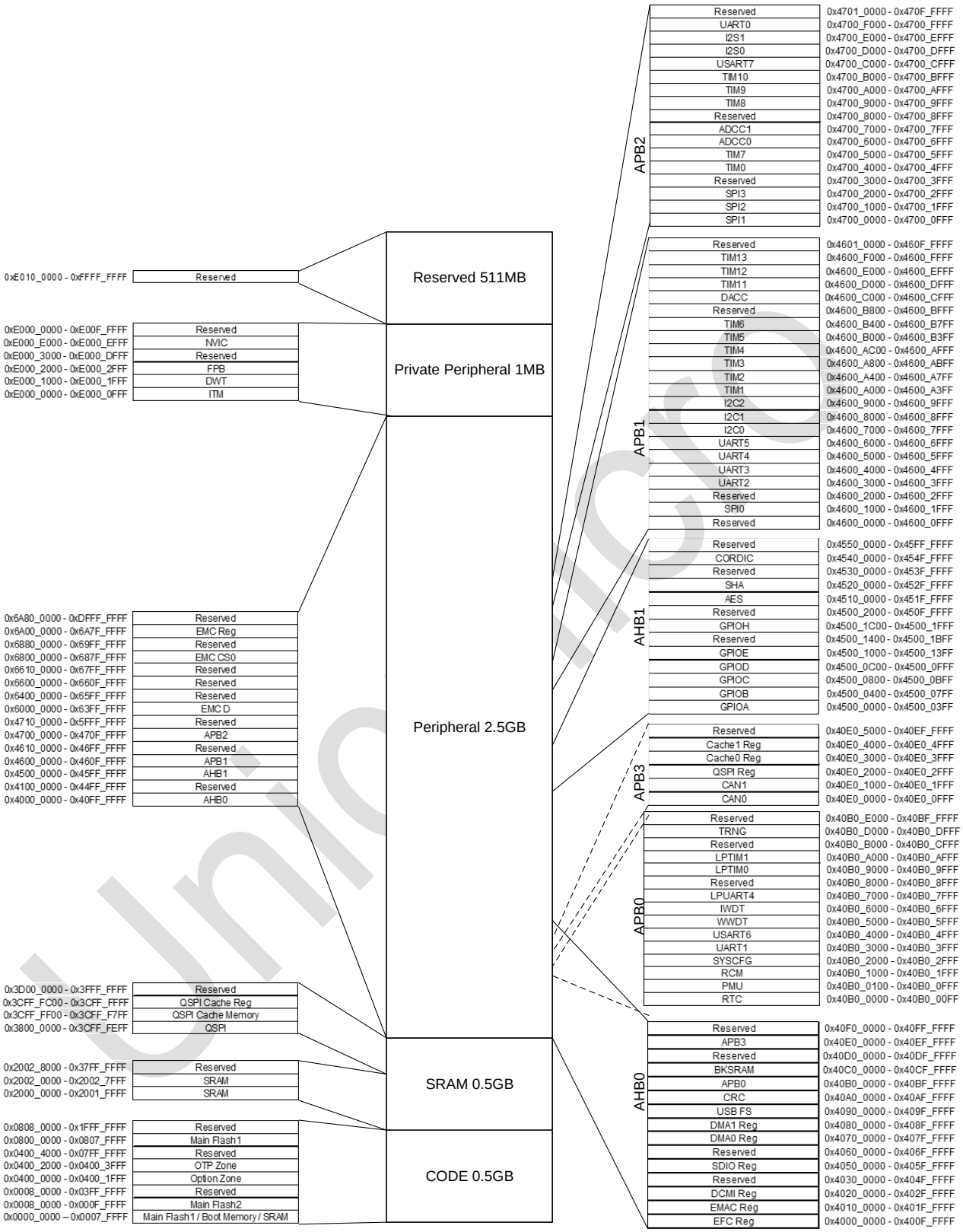


图 3-2：存储器地址映射图

表 3-1：存储器重映射地址

Block/Module	Address range	Size
Main FLASH1	0x0000_0000 ~ 0x0007_FFFF 0x0800_0000 ~ 0x0807_FFFF	512KB
SRAM0	0x2000_0000 ~ 0x2001_FFFF	128KB
SRAM1	0x2002_0000 ~ 0x2002_7FFF	32KB
SRAM2（DMA）	0x2002_8000 ~ 0x2002_FFFF	32KB
EMC_D	0x6000_0000 ~ 0x63FF_FFFF	16MB
EMC_CS0	0x6800_0000 ~ 0x687F_FFFF	8MB
EMC_Config	0x6A00_0000 ~ 0x6A7F_FFFF	8MB
QSPI	0x0008_0000 ~ 0x03FF_FFFF 0x3800_0000 ~ 0x3CFF_FFFF	16MB
QSPICACHE_DATA_MEM	0x3CFF_F000 ~ 0x3CFF_F7FF	-
QSPICACHE_TAG_MEM	0x3CFF_F800 ~ 0x3CFF_FBFF	-
QSPICACHE_REG	0x3CFF_FC00 ~ 0x3CFF_FFFF	-

表 3-2：存储器映射地址表

Block / Module		Address range
AHB0 Peripheral	EFC	0x4000_0000 ~ 0x400F_FFFF
	EMAC	0x4010_0000 ~ 0x401F_FFFF
	DCMI	0x4020_0000 ~ 0x402F_FFFF
	SDIO	0x4050_0000 ~ 0x405F_FFFF
	DMA0	0x4070_0000 ~ 0x407F_FFFF
	DMA1	0x4080_0000 ~ 0x408F_FFFF
	USB FS Device	0x4090_0000 ~ 0x409F_FFFF
	CRC	0x40A0_0000 ~ 0x40AF_FFFF
	BKSRAM	0x40C0_0000 ~ 0x40CF_FFFF
AHB1 Peripheral	GPIOA	0x4500_0000 ~ 0x4500_03FF
	GPIOB	0x4500_0400 ~ 0x4500_07FF
	GPIOC	0x4500_0800 ~ 0x4500_0BFF
	GPIOD	0x4500_0C00 ~ 0x4500_0FFF
	GPIOE	0x4500_1000 ~ 0x4500_13FF
	GPIOH	0x4500_1C00 ~ 0x4500_1FFF
	AES	0x4510_0000 ~ 0x451F_FFFF
	SHA	0x4520_0000 ~ 0x452F_FFFF
	CORDIC	0x4540_0000 ~ 0x454F_FFFF
AHB2APB0 Bridge	RTC	0x40B0_0000 ~ 0x40B0_00FF
	PMU	0x40B0_0100 ~ 0x40B0_0FFF
	RCM	0x40B0_1000 ~ 0x40B0_1FFF
	SYSCFG	0x40B0_2000 ~ 0x40B0_2FFF
	UART1	0x40B0_3000 ~ 0x40B0_3FFF
	USART6	0x40B0_4000 ~ 0x40B0_4FFF
	WWDT	0x40B0_5000 ~ 0x40B0_5FFF
	IWDT	0x40B0_6000 ~ 0x40B0_6FFF

Block / Module		Address range
	LPUART	0x40B0_7000 ~ 0x40B0_7FFF
	LPTIM0	0x40B0_9000 ~ 0x40B0_9FFF
	LPTIM1	0x40B0_A000 ~ 0x40B0_AFFF
	RNG	0x40B0_D000 ~ 0x40B0_DFFF
AHB2APB1 Bridge	SPI0	0x4600_1000 ~ 0x4600_1FFF
	UART2	0x4600_3000 ~ 0x4600_3FFF
	UART3	0x4600_4000 ~ 0x4600_4FFF
	UART4	0x4600_5000 ~ 0x4600_5FFF
	UART5	0x4600_6000 ~ 0x4600_6FFF
	I2C0	0x4600_7000 ~ 0x4600_7FFF
	I2C1	0x4600_8000 ~ 0x4600_8FFF
	I2C2	0x4600_9000 ~ 0x4600_9FFF
	TIM1~4	0x4600_A000 ~ 0x4600_AFFF
	TIM5~6	0x4600_B000 ~ 0x4600_BFFF
	DAC	0x4600_C000 ~ 0x4600_CFFF
	TIM11	0x4600_D000 ~ 0x4600_DFFF
	TIM12	0x4600_E000 ~ 0x4600_EFFF
	TIM13	0x4600_F000 ~ 0x4600_FFFF
AHB2APB2 Bridge	SPI1	0x4700_0000 ~ 0x4700_0FFF
	SPI2	0x4700_1000 ~ 0x4700_1FFF
	SPI3	0x4700_2000 ~ 0x4700_2FFF
	TIM0	0x4700_4000 ~ 0x4700_4FFF
	TIM7	0x4700_5000 ~ 0x4700_5FFF
	ADC0	0x4700_6000 ~ 0x4700_6FFF
	ADC1	0x4700_7000 ~ 0x4700_7FFF
	TIM8	0x4700_9000 ~ 0x4700_9FFF
	TIM9	0x4700_A000 ~ 0x4700_AFFF
	TIM10	0x4700_B000 ~ 0x4700_BFFF
	USART7	0x4700_C000 ~ 0x4700_CFFF
	I2S0	0x4700_D000 ~ 0x4700_DFFF
	I2S1	0x4700_E000 ~ 0x4700_EFFF
	UART0	0x4700_F000 ~ 0x4700_FFFF
AHB2APB3 Bridge	CAN0	0x40E0_0000 ~ 0x40E0_0FFF
	CAN1	0x40E0_1000 ~ 0x40E0_1FFF
	QSPI	0x40E0_2000 ~ 0x40E0_2FFF
	EFC ICache	0x40E0_3000 ~ 0x40E0_3FFF
	EFC DCache	0x40E0_4000 ~ 0x40E0_4FFF

4 存储系统（Memory system）

4.1 FLASH

4.1.1 概述

提供最大 512KB 容量的 FLASH，其中 Flash 控制器（EFC）用于在 CPU 的控制下完成对 Eflash 储存器的读、写、擦等操作。

4.1.2 主要特性

- 8/16/32 位 Flash 读，32 位 Flash 编程
- 片内 512KB 容量 FLASH，总共 64 个 page，每个 page 是 8KByte
- 支持 page 擦除和 chip 擦除操作流程
- Flash 控制器支持连续编程，在向一个 512 字节对齐的储存空间编程数据时可以节省时间
- 读等待时间可以配置
- 支持擦写保护功能
- 支持自动锁总线功能
- 支持编程后自动回读校验
- 具有一个 OTP 储存区
- 支持指令和数据 Cache

4.1.3 功能描述

Flash 控制器可以完成对 Flash 的读、写、擦除等操作。

Flash 控制器要求在>1MHz 频率下进行擦写。用户需要根据实际运行频率配置 EFC_TIME 寄存器。

可以对 Flash 编程 32 位宽的数据，编程前需要解锁操作。

Flash 控制器支持连续编程，在向一个 512 字节对齐的储存空间中编程数据时可以节省时间。进行连续编程需要在此 Flash 以外的存储器上运行程序。

控制器具有擦除校验功能，可以在页擦除和批量擦除模式下自动检查 Flash 中的数据位是否全部变为 1。

4.1.3.1 FLASH 块 1 的储存器地址映射

图 4-1：EFC 存储器地址映射

4.1.3.2 用户选项区域

地址 0x0400_0000 ~ 0x0400_1FFF 为用户选项页，其中部分地址具有特殊功能，如下表所示：

表 4-1：EFC 特殊功能地址说明

名称	地址	位宽	内容
WRITE_PROTECT & IWDT_HW_START	0x04001FA0	32bit	禁止对 flash main 区指定编号区间(0~63)的页进行擦写。 Bit[31:24]：A5：激活写保护功能；其它：不激活写保护功能。 Bit[23:14]：保留。 Bit[13:8]：指定保护页结束编号。 Bit[7:6]：保留。 Bit[5:0]：指定保护页起始编号。
DEBUG_DISABLE	0x04001FB0	8bit	禁止 JTAG 或 SWD 调试接口操作： 0xAB：禁止调试口，JTAG 将无法识别 其它：不禁止调试口操作

4.1.3.3 一次性编程（OTP）区域

表 4-2：EFC 一次性编程（OTP）区域

编号	OTP 储存区域地址	大小	OTP 锁定控制地址	备注
0	0x04002000 ~ 0x040020FF	256 bytes	0x04003F80	-

编号	OTP 储存区域地址	大小	OTP 锁定控制地址	备注
1	0x04002100 ~ 0x040021FF	256 bytes	0x04003F84	-
2	0x04002200 ~ 0x040022FF	256 bytes	0x04003F88	-
3	0x04002300 ~ 0x040023FF	256 bytes	0x04003F8C	-
4	0x04002400 ~ 0x040024FF	256 bytes	0x04003F90	-
5	0x04002500 ~ 0x040025FF	256 bytes	0x04003F94	-
6	0x04002600 ~ 0x040026FF	256 bytes	0x04003F98	-
7	0x04002700 ~ 0x040027FF	256 bytes	0x04003F9C	-
8	0x04002800 ~ 0x040028FF	256 bytes	0x04003FA0	-
9	0x04002900 ~ 0x040029FF	256 bytes	0x04003FA4	-
10	0x04002A00 ~ 0x04002AFF	256 bytes	0x04003FA8	-
11	0x04002B00 ~ 0x04002BFF	256 bytes	0x04003FAC	-
12	0x04002C00 ~ 0x04002CFF	256 bytes	0x04003FB0	-
13	0x04002D00 ~ 0x04002DFF	256 bytes	0x04003FB4	-
14	0x04002E00 ~ 0x04002EFF	256 bytes	0x04003FB8	-
15	0x04002F00 ~ 0x04002FFF	256 bytes	0x04003FBC	-
16	0x04003000 ~ 0x040030FF	256 bytes	0x04003FC0	-
17	0x04003100 ~ 0x040031FF	256 bytes	0x04003FC4	-
18	0x04003200 ~ 0x040032FF	256 bytes	0x04003FC8	-
19	0x04003300 ~ 0x040033FF	256 bytes	0x04003FCC	-
20	0x04003400 ~ 0x040034FF	256 bytes	0x04003FD0	保留区域
21	0x04003500 ~ 0x040035FF	256 bytes	0x04003FD4	保留区域
22	0x04003600 ~ 0x040036FF	256 bytes	0x04003FD8	保留区域
23	0x04003700 ~ 0x040037FF	256 bytes	0x04003FDC	保留区域
24	0x04003800 ~ 0x040038FF	256 bytes	0x04003FE0	-
25	0x04003900 ~ 0x040039FF	256 bytes	0x04003FE4	-
26	0x04003A00 ~ 0x04003AFF	256 bytes	0x04003FE8	-
27	0x04003B00 ~ 0x04003BFF	256 bytes	0x04003FEC	保留区域
28	0x04003C00 ~ 0x04003CFF	256 bytes	0x04003FF0	保留区域
29	0x04003D00 ~ 0x04003DFF	256 bytes	0x04003FF4	保留区域
30	0x04003E00 ~ 0x04003EFF	256 bytes	0x04003FF8	-
31	0x04003F00 ~ 0x04003F7F	128 bytes	-	保留区域
32	0x04003F80 ~ 0x04003FFF	128 bytes	-	OTP 锁定控制地址

注：

- 将锁定控制地址的第一个字节编程为 8’h00 即可锁定对应的储存空间，使其无法编程。
- OTP 区域不可擦除。

4.1.4 寄存器描述

寄存器基地址：0x4000_0000

寄存器列表如下：

表 4-3：EFC 寄存器列表

偏移地址	名称	描述
0x00	EFC_CTRL	控制寄存器
0x04	EFC_TIME	时序寄存器
0x08	EFC_SEC	安全操作寄存器
0x0C	EFC_STATUS	状态寄存器
0x10	EFC_INTSTATUS	中断状态寄存器

0x14	EFC_INTEN	中断使能寄存器
0x18	EFC_LPCR	低功耗控制寄存器
0x1C	EFC_ADDRREC	最后操作地址寄存器

以下各节详细介绍寄存器。

4.1.4.1 控制寄存器 (EFC_CTRL)

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5	ERASEVE	R/W	0x0	擦除校验使能位： 1：使能擦除校验 0：不使能擦除校验
4	PROGVE	R/W	0x0	编程校验使能位（仅在使用单次编程时有效）： 1：使能编程校验 0：不使能编程校验
3	MERASES	R/W	0x0	批量擦除模式选择位（擦除 main 区全部页）： 1：选择批量擦除模式 0：不选择批量擦除模式
2	PERASES	R/W	0x0	页擦除模式选择位： 1：选择页擦除模式 0：不选择页擦除模式
1	CONPROGS	R/W	0x0	连续编程模式选择位： 1：选择连续编程模式 0：不选择连续编程模式
0	PROGS	R/W	0x0	单次编程模式选择位： 1：选择单次编程模式 0：不选择单次编程模式

注：第 0 ~ 3 位只能选用其中一个。

4.1.4.2 时序寄存器 (EFC_TIME)

偏移地址：0x04

复位值：0x0000 6018

位	名称	属性	复位值	描述
31:24	REGWE	W	0x0	此域写 A5 时此寄存器可写入。 请一次对此寄存器写入 0xA50XXXXX
23:18	RSV	-	-	保留
17:16	RECYC	R/W	0x0	读取完成后到下一次读取的额外等待时间

位	名称	属性	复位值	描述
15:12	RWAITCYC	R/W	6	读等待周期设置位，参考下方表格，可以偏大
11:9	RSV	-	-	保留
8:0	FREQ	R/W	24	擦写时间标尺，请填入（EFC 运行频率-1），单位 MHz。 最小值为 0，对应擦写时要求的最低频率 1MHz。 （耐压-10% ~ +30%）

表 4-4：VDD=1.1V 下 Read wait cycle 与频率的关系

频率/MHz	Read wait cycle
< 50	0
50 ~ 100	1
100 ~ 150	2
150 ~ 200	3
200~250	4
250~300	5
…	…

4.1.4.3 安全操作寄存器（EFC_SEC)

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	UNLOCK	R/W	0x0	对 Flash 编程/擦除时，需要对此寄存器进行解锁操作。 对此寄存器写入 0x55AAAA55 解除锁定，解锁后读此寄存器可以读到 1。 写入其他值会重新锁定。 开始编程/擦除时会重新锁定。

4.1.4.4 状态寄存器（EFC_STATUS)

偏移地址：0x0C

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	VDDS	R	0x0	VDDH 电压状态（LVD）指示位： 0：正常状态 1：低电压警告
7:6	RSV	-	-	保留
5:4	LPS	R	0x0	低功耗模式状态指示位： 11：正在掉电模式 10：正在睡眠模式 01：低电压运行 00：高电压运行

位	名称	属性	复位值	描述
3	RSV	-	-	保留
2	CONPROGRDY	R/W	0x0	Flash 连续编程状态指示位；对此位写 0 可以退出连续编程模式 1：正处在等待下一个连续编程的状态 0：未处在等待下一个连续编程的状态
1	RSV	-	-	保留
0	RDYS	R	0x1	Flash 状态指示位： 1：Flash 状态空闲 0：Flash 状态忙

4.1.4.5 中断状态寄存器 (EFC_INTSTATUS)

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3	EVFS	R/W1C	0x0	擦除校验失败状态位： 写 1 清除该位；如果中断允许，则产生中断。 1：擦除校验失败，Flash 中的数据不是全为 1 0：未出现擦除校验失败
2	PVFS	R/W1C	0x0	写入校验失败状态位： 写 1 清除该位；如果中断允许，则产生中断。 1：写入校验失败，Flash 中的数据与写入数据不符 0：未出现写入校验失败
1	VDDLS	R/W1C	0x0	VDDH 电压过低（LVD）中断状态位： 写 1 清除该位；如果中断允许，则产生中断。 1：发生过 VDDH 电压过低警告（LVD） 0：正常状态
0	OPDS	R/W1C	0x0	编程或擦除完成中断状态位 写 1 清除该位；如果中断允许，则产生中断。 1：发生了编程/擦除完成 0：未发生编程/擦除完成

4.1.4.6 中断使能寄存器 (EFC_INTEN)

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3	EVFE	R/W	0x0	擦除校验失败中断使能位： 1：使能中断 0：不使能中断

位	名称	属性	复位值	描述
2	PVFE	R/W	0x0	编程校验失败中断使能位： 1：使能中断 0：不使能中断
1	VDDLE	R/W	0x0	VDDH 电压（LVD）过低中断使能位： 1：使能中断 0：不使能中断
0	OPDE	R/W	0x0	编程或擦除完成中断使能位： 1：使能中断 0：不使能中断

4.1.4.7 低功耗控制寄存器（EFC_LPCR)

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:24	REGWE	W	0x0	此域写 A5 时此寄存器可写入。 请一次对此寄存器写入 0xA500000X
23:1	RSV	-	-	保留
0	MODES	R/W	0x0	系统进入低功耗模式时 Flash 的行为选择位： 1：进入掉电模式； 0：进入睡眠模式 注：STOP 模式，2 种模式都可以选择。 STANDBY 待机模式，建议设置 1。

4.1.4.8 最后操作地址寄存器（EFC_ADDRREC)

偏移地址：0x1C

复位值：0x0C07 FFFF

位	名称	属性	复位值	描述
31:21	RSV	-	-	保留
20:0	ADDR	R	0xC07FFFF	记录最近一次完成擦除或编程的地址

4.1.5 使用流程

4.1.5.1 时序设定

在使用 Flash 时，无论进行何种操作，都需要首先设定 EFC_TIME 寄存器令控制值处于正确范围。

1. 根据系统频率，确认 FREQ 域需要填入的数值（系统频率-1），使用编程、擦除、低功耗模式功

能前需要该域数值正确（使用低功耗模式不要求系统频率大于 1MHz，低于 1MHz 频率下填入 0 即可）。

2. 根据系统频率，确认 RWAITCYC(Read_Wait_Cycle)域需要填入的数值，不能小于要求的数值；当需要提高系统频率时，应当先修改 EFC_TIME 寄存器再提高系统频率；当需要降低系统频率时，应当先降低系统频率再修改 EFC_TIME 寄存器。
3. 一次性对 EFC_TIME 寄存器进行写入，需要配合向 REGWE(Reg_Wr_Enable)域写特征值。

4.1.5.2 单次编程

1. 操作__set_PRIMASK()，关闭总中断。
2. 配置 EFC_CTRL 寄存器，选择单次编程模式，可以选择打开编程校验。
3. 操作 EFC_SEC 寄存器解锁，一次写入 0x55AAAA55。
4. 对需要编程的地址写入编程数据。
5. 等待 EFC_STATUS[0]空闲。
6. 还原 EFC_CTRL 寄存器为默认值。
7. 操作__set_PRIMASK()，开启总中断。
8. 操作 EFC_SEC 寄存器锁定 flash，一次写入 0x55AA0000。

4.1.5.3 连续编程

在 512bytes 范围内可以连续进行多次编程，控制程序运行在 Flash 以外的存储器中时才能真正地加快速度。

1. 配置 EFC_CTRL 寄存器，选择连续编程模式。
2. 操作 EFC_SEC 寄存器解锁，一次写入 0x55AAAA55。
3. 对需要编程的地址写入编程数据。
4. 根据需要，重复 2 和 3 中的操作。
5. 完成最后一次编程后，将 0 写入 EFC_STATUS 寄存器以结束连续编程，禁止在连续编程模式下等待。
6. 还原 EFC_CTRL 寄存器为默认值。
7. 操作 EFC_SEC 寄存器锁定 flash，一次写入 0x55AA0000。

4.1.5.4 页擦除

1. 操作__set_PRIMASK()，关闭总中断。
2. 配置 EFC_CTRL 寄存器，选择页擦除模式。
3. 操作 EFC_SEC 寄存器解锁，一次写入 0x55AAAA55。

4. 对需要擦除的页中的任意地址写任意数据。
5. 等待 EFC_STATUS[0]空闲。
6. 还原 EFC_CTRL 寄存器为默认值。
7. 操作 EFC_SEC 寄存器锁定 flash，一次写入 0x55AA0000。
8. 操作__set_PRIMASK()，开启总中断。

4.1.5.5 批量擦除

批量擦除会擦除 Flash Main 区域所有的页。

1. 操作__set_PRIMASK()，关闭总中断。
2. 配置 EFC_CTRL 寄存器，选择批量擦除模式。
3. 操作 EFC_SEC 寄存器解锁，一次写入 0x55AAAA55。
4. 对 Main 区域中的任意地址写任意数据。
5. 等待 EFC_STATUS[0]空闲。
6. 还原 EFC_CTRL 寄存器为默认值。
7. 操作 EFC_SEC 寄存器锁定 flash，一次写入 0x55AA0000。
8. 操作__set_PRIMASK()，开启总中断。

4.2 SRAM

4.2.1 概述

SRAM 主要用于代码运行，存放程序执行过程中的变量和数据或堆栈，容量最大为 160KB（其中 128KB 容量的 SRAM0，支持 0 等待访问），最大地址范围为：0x2000_0000~0x2002_7FFF。

4.2.2 主要特性

- 支持字节、半字、字的读写访问
- 支持 DMA 方式

4.3 CACHE

缓存控制器可以缓存从 Flash 中取出的数据，加速下一次读取以达到更高的系统性能。此 CACHE 分为指令缓冲器 ICACHE 和数据缓冲器 DCACHE，ICACHE 大小为 4kB，DCACHE 大小为 256byte。缓存控制器可以缓存从 Flash 中取出的数据，加速下一次读取；可选择是否开启预取

Cache Line，缓存开启时会自动初始化缓存区域。

4.3.1 寄存器描述

指令和数据 Cache 控制的寄存器如下：

ICache（指令 Cache）寄存器基地址：0x40E0_3000

DCache（数据 Cache）寄存器基地址：0x40E0_4000

ICache 和 DCache 的寄存器列表如下：

表 4-5：Cache 寄存器列表

偏移地址	名称	描述
0x00	CACHE_CTRL	控制寄存器
0x04	CACHE_STATUS	状态寄存器
0x14	CACHE_HITCNT	命中统计寄存器
0x18	CACHE_MISSCNT	未命中统计寄存器

以下各节详细介绍寄存器。

4.3.1.1 控制寄存器 (CACHE_CTRL)

偏移地址：0x00

复位值：0x0000 0040

位	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6	STATIS_EN	R/W	0x1	统计计数器开关： 0：关闭 1：开启
5	PREFETCH	R/W	0x0	预取开关： 0：关闭 1：开启
4:1	RSV	-	-	保留
0	CACHE_EN	R/W	0x0	缓存总体开关： 0：关闭 1：开启

4.3.1.2 状态寄存器 (CACHE_STATUS)

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	INV_STATUS	R	0x0	指示正在初始化缓存

位	名称	属性	复位值	描述
1:0	CACHE_STATUS	R	00	缓存开关状态： 00：关闭 01：正在开启 10：开启 11：正在关闭

4.3.1.3 命中统计寄存器（CACHE_HITCNT)

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	HIT_CNT	R/W	0x0	统计缓存命中次数

4.3.1.4 未命中统计寄存器（CACHE_MISSCNT)

偏移地址：0x18

复位值：0x0000_0000

位	名称	属性	复位值	描述
31:0	MISS_CNT	R/W	0x0	统计缓存未命中次数

4.3.2 使用流程

4.3.2.1 开启缓存

1. （可选）在缓存未开启的状态下配置预取开关（CACHE_CTRL[5]），注意开启预取可能会对效率产生影响。
2. 开启缓存开关（CACHE_CTRL[0]）。
3. 查询缓存开关状态（CACHE_STATUS[1:0]），直至显示为已开启。

4.3.2.2 更新 FLASH 内容

在开启缓存后，如果更新了 FLASH 中储存的内容，会产生不一致，需要重启缓存以清空缓存空间。

1. 关闭缓存开关（CACHE_CTRL[0]）。
2. 查询缓存开关状态（CACHE_STATUS[1:0]），直至显示为已关闭。
3. 开启缓存开关（CACHE_CTRL[0]）。
4. 查询缓存开关状态（CACHE_STATUS[1:0]），直至显示为已开启。

5 电源管理单元（PMU）

5.1 芯片供电

芯片单电源供电和 VBAT 备份电源，外部提供供电电源（1.8~3.6V），内置 LDO 产生内部数字电路工作电压。

系统供电方案如下图所示。

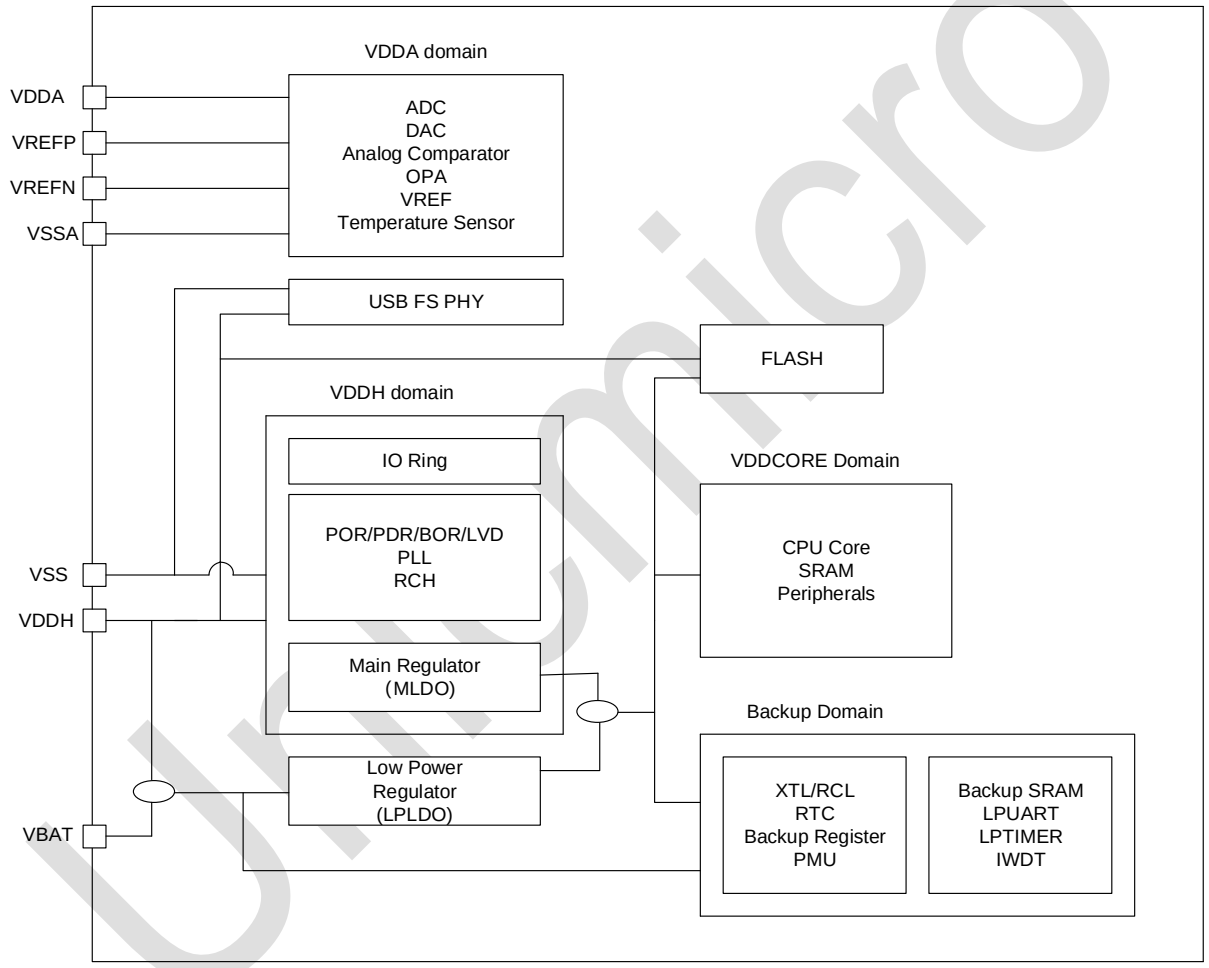


图 5-1：系统供电方案框图

5.2 内部 PMU 模块功能

5.2.1 内部 PMU 模块简介

芯片基于应用场景提供不同能耗模式来降低整体功耗。

- 工作模式（Run Mode）
- 睡眠模式（Sleep Mode）

- 时钟停止模式（Stop Mode）
- 待机模式（Standby Mode）
- 深度待机模式（DeepStandby Mode）

低功耗模式和唤醒信息如下表所示：

表 5-1：低功耗模式表

Power Modes	说明	进入方式	唤醒源
工作模式 (Run Mode)	所有电源上电，高速时钟工作	-	-
睡眠模式 (Sleep Mode)	所有电源上电，M4 核高速时钟关掉，其他保持	1. PMU_STANDBY_EN=0, PMU_MODE[1:0]=00，PMU_BKSRAMOFF=0 2. EFC_Sys_Mode=0 3. WFI/WFE	1. 任何 GPIO 中断唤醒 2. 任何外设中断唤醒 3. 复位唤醒（RESETN, IWDT 复位）
停止模式 (Stop Mode)	所有电源上电，高速时钟关掉，32K 低速时钟在	1. PMU_STANDBY_EN=0, PMU_MODE[1:0]=01，PMU_BKSRAMOFF=0 2. EFC_Sys_Mode=0 3. M4 核的 SLEEPDEEP =1 4. WFI 或 WFE	1. 任何 GPIO 中断唤醒 2. RTC 闹钟中断，RTC 时间戳和侵入中断唤醒 3. IWDT 复位或者中断唤醒 4. 外部 LPUART 唤醒 5. LPTIM0~1 定时唤醒
待机模式 0 (Standby0 Mode)	CORE domain 断电，BBU domain 一直有电，32K 低速时钟运行。BKSRAM/IWDT/LPTIM/LPUART 模块有电	1. PMU_STANDBY_EN=1, PMU_MODE[1:0]=10，PMU_BKSRAMOFF=0 2. EFC_Sys_Mode=1 3. M4 核的 SLEEPDEEP =1 4. WFI 或 WFE	1. 外部管脚 PA0，PA2，PC0，PC2，PC3 唤醒 2. RTC 闹钟中断，RTC 时间戳和侵入中断唤醒 3. LPTIM0~1 定时唤醒 4. LPUART（只有 PC2 管脚唤醒） 5. IWDT（复位和中断都支持）功能唤醒 6. 复位唤醒（RESETN）
待机模式 1 (Standby1 Mode)	CORE domain 断电，BBU domain 一直有电，32K 低速时钟运行。BKSRAM/IWDT/LPTIM/LPUART 模块掉电	1. PMU_STANDBY_EN=1, PMU_MODE[1:0]=10，PMU_BKSRAMOFF=1 2. EFC_Sys_Mode=1 3. 设置 M4 核的 SLEEPDEEP =1 4. WFI 或 WFE	1. 外部管脚 PA0，PA2，PC0，PC2，PC3，PC13 唤醒 2. RTC 闹钟中断，RTC 时间戳和侵入中断唤醒 3. 复位唤醒（RESETN）
深度待机模式 0 (DeepStandby0 Mode)	CORE domain 断电，BBU domain 一直有电，32K 低速时钟不在。BKSRAM/IWDT/LPTIM/LPUART 有电	1. PMU_STANDBY_EN=1, PMU_MODE[1:0]=11，PMU_BKSRAMOFF=0 2. EFC_Sys_Mode=1 3. 设置 M4 核的 SLEEPDEEP =1 4. WFI 或 WFE	1. 外部管脚 PA0，PA2，PC0，PC2，PC3，PC13 唤醒 2. 复位唤醒（RESETN）

Power Modes	说明	进入方式	唤醒源
深度待机模式 1 (DeepStandby1 Mode）	CORE domain 断电，BBU domain 一直有电，32K 低速时钟不在。BKSRAM/IWDT/LPTIM/LPUART 掉电	1. PMU_STANDBY_EN=1, PMU_MODE[1:0]=11，PMU_BKSRAMOFF=1 2. EFC_Sys_Mode=1 3. 设置 M4 核的 SLEEPDEEP =1 4. WFI 或 WFE	1. 外部管脚 PA0，PA2，PC0，PC2，PC3，PC13 唤醒 2. 复位唤醒（RESETN）
掉电模式 (Poweroff Mode）	所有电源断电	外部 VDDH&VBAT 掉电	上电

注：

- BBU domain 包括 RTC，备份寄存器和 PMU 逻辑。
- Sleep Mode 和 STOP Mode 支持 IO 保持。
- Standby* Mode 和 DeepStandby* Mode，不支持 IO 保持，除了外部输入唤醒的 IO，其他 IO 状态为 High-Impedance 高阻态。

5.2.2 工作模式（Run mode）

在工作模式下，根据应用场景不同，也可以降低工作频率或者关掉一些不用的外设来达到降低功耗的目的。

5.2.3 睡眠模式（Sleep mode）

在低功耗模式下，CPU 核停止工作，保留中断处理功能。其它外设等模块时钟和复位可由软件设置。低功耗模式由 CPU 核特定指令 WFI 进入，唤醒由中断触发。

5.2.4 停止模式（Stop mode）

在停止模式下，系统 RCH/XTH/PLL 停止工作（RCL/XTL 低频时钟正常工作），保留中断处理功能。其它外设等模块时钟和复位可由软件设置。低功耗模式由 CPU 核特定指令 WFI 进入，唤醒由中断触发。

5.2.5 待机模式（Standby mode）

待机模式 RTC 一直工作，LPUART/LPTIM0~1 可根据配置工作或者不工作，BKSRAM 根据配置可以断电或者不断电。

5.2.6 深度待机模式（DeepStandby mode）

深度待机模式内部低速时钟停止工作，备份寄存器保持，BKSRAM 根据配置可以断电或者不断电保持。

5.3 寄存器描述

注意：如果系统选用 XTL 或者 RCL 作为系统时钟，此 PMU 的寄存器不可访问。

寄存器基地址：0x40B0_0100

PMU 寄存器列表如下：

表 5-2：PMU 寄存器列表

偏移地址	名称	描述
0x00	PMU_MR	PMU 模式寄存器
0x04	PMU_PDWKCR	掉电唤醒控制寄存器
0x08	PMU_PUSTCR	POWERUP 稳定时间配置寄存器
0x0C	PMU_VDCR	PDR/BOR/LVD 配置寄存器
0x14	PMU_SASR	系统辅助状态寄存器
0x1C	PMU_XTLCR	XTL 时钟配置寄存器
0x20	PMU_RCLCR	RCL 时钟配置寄存器
0x24	PMU_FCCR	功能时钟控制寄存器
0x28	PMU_FRCR	功能复位控制寄存器
0x50	PMU_CPR	配置保护寄存器

5.3.1 PMU 模式寄存器（PMU_MR）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:9	RSV	R	0	保留
8	VOLTAGE_SEL	R/W	0	MLDO 输出的电压选择： 0：选择 1.1V 输出（采用 1.1V 的 trimming 值） 1：选择 0.9V 输出（采用 0.9V 的 trimming 值）
7:5	RSV	R	0	保留
4	STDBY_EN	R/W	0	待机模式（Standby mode）有效控制寄存器： 0：待机模式无效 1：待机模式有效

位	名称	属性	复位值	描述
3	STOP_CLK_SEL	R/W	0	STOP MODE（停止模式）时钟选择控制寄存器： 0：进入 STOP MODE，先切换到 RCH 时钟，唤醒还是 RCH 时钟工作 1：保持 STOP MODE 前的工作时钟不变（比如 PLL 时钟）
2	BKSRAMOFF	R/W	0	BKSRAM 等逻辑断电控制寄存器： 0：BKSRAM、IWDT、LPUART、LPTIM 0~1 不断电 1：BKSRAM、IWDT、LPUART、LPTIM 0~1 断电
1:0	PMU_MODE	R/W	0	PMU 模式寄存器： 2'b00：正常工作模式（Run mode） 2'b01：停止模式（Stop mode） 2'b10：待机模式（Standby 模式/掉电模式） 2'b11：深度待机模式（DeepStandby 模式/深度掉电模式）

5.3.2 掉电唤醒控制寄存器（PMU_PDWKCR）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31	IO_WK_CLR	W	-	清除 PA0/PA2/PC0/PC2/PC3/PC13 唤醒状态位。 注：清除唤醒状态位前，需要先让唤醒使能位为无效，然后再写 1 清除所有唤醒状态标志位。
30	RESETN_WK_FLAG	R	0	外部 RESETN 复位唤醒状态位（只有 Standby*&DeepStandby*模式下有效）： 0：不是 RESETN 管脚唤醒 1：RESETN 管脚唤醒(单管脚唤醒情况下)，或者可能是 RESETN 管脚唤醒（多管脚同时唤醒情况下） 注：此寄存器位被 bit[31]写 1 清除。
29	PC13_WK_FLAG	R	0	PC13 唤醒状态位（只有 Standby*&DeepStandby*模式下有效）： 0：不是 PC13 管脚唤醒 1：PC13 管脚唤醒(单管脚唤醒情况下)，或者可能是 PC13 管脚唤醒（多管脚同时唤醒情况下） 注：此寄存器位被 bit[31]写 1 清除。

位	名称	属性	复位值	描述
28	PC3_WK_FLAG	R	0	PC3 唤醒状态位（只有 Standby*&Deepstandby*模式下有效）： 0：不是 PC3 管脚唤醒 1：PC3 管脚唤醒(单管脚唤醒情况下)，或者可能是 PC3 管脚唤醒（多管脚同时唤醒情况下） 注：此寄存器位被 bit[31]写 1 清除。
27	PC2_WK_FLAG	R	0	PC2 唤醒状态位（只有 Standby*&DeepStandby*模式下有效）： 0：不是 PC2 管脚唤醒 1：PC2 管脚唤醒(单管脚唤醒情况下)，或者可能是 PC2 管脚唤醒（多管脚同时唤醒情况下） 注：此寄存器位被 bit[31]写 1 清除。
26	PC0_WK_FLAG	R	0	PC0 唤醒状态位（只有 Standby*&DeepStandby*模式下有效）： 0：不是 PC0 管脚唤醒 1：PC0 管脚唤醒(单管脚唤醒情况下)，或者可能是 PC0 管脚唤醒（多管脚同时唤醒情况下） 注：此寄存器位被 bit[31]写 1 清除。
25	PA2_WK_FLAG	R	0	PA2 唤醒状态位（只有 Standby*&DeepStandby*模式下有效）： 0：不是 PA2 管脚唤醒 1：PA2 管脚唤醒(单管脚唤醒情况下)，或者可能是 PA2 管脚唤醒（多管脚同时唤醒情况下） 注：此寄存器位被 bit[31]写 1 清除。
24	PA0_WK_FLAG	R	0	PA0 唤醒状态位（只有 Standby*&DeepStandby*模式下有效） 0：不是 PA0 管脚唤醒 1：PA0 管脚唤醒(单管脚唤醒情况下)，或者可能是 PA0 管脚唤醒（多管脚同时唤醒情况下） 注：此寄存器位被 bit[31]写 1 清除。
23:22	RSV	R	-	保留
21	PC13_WKEG	R/W	0	PC13 管脚唤醒边沿选择寄存器： 0：上升沿触发 1：下降沿触发
20	PC3_WKEG	R/W	0	PC3 管脚唤醒边沿选择寄存器： 0：上升沿触发 1：下降沿触发
19	PC2_WKEG	R/W	0	PC2 管脚唤醒边沿选择寄存器： 0：上升沿触发 1：下降沿触发
18	PC0_WKEG	R/W	0	PC0 管脚唤醒边沿选择寄存器： 0：上升沿触发 1：下降沿触发

位	名称	属性	复位值	描述
17	PA2_WKEG	R/W	0	PA2 管脚唤醒边沿选择寄存器： 0：上升沿触发 1：下降沿触发
16	PA0_WKEG	R/W	0	PA0 管脚唤醒边沿选择寄存器： 0：上升沿触发 1：下降沿触发
15:13	RSV	R	0	保留
12	PC13_WKE	R/W	0	PC13 中断唤醒使能位： 0：唤醒事件无效 1：唤醒事件有效
11	PC3_WKE	R/W	0	PC3 中断唤醒使能位： 0：唤醒事件无效 1：唤醒事件有效
10	PC2_WKE	R/W	0	PC2 中断唤醒使能位： 0：唤醒事件无效 1：唤醒事件有效
9	PC0_WKE	R/W	0	PC0 中断唤醒使能位： 0：唤醒事件无效 1：唤醒事件有效
8	PA2_WKE	R/W	0	PA2 中断唤醒使能位： 0：唤醒事件无效 1：唤醒事件有效
7	LPTIM1_WKE	R/W	0	LPTIM1 中断唤醒使能位： 0：唤醒事件无效 1：唤醒事件有效
6	LPTIM0_WKE	R/W	0	LPTIM0 中断唤醒使能位： 0：唤醒事件无效 1：唤醒事件有效
5	LPUART_WKE	RW	0	LPUART 中断唤醒使能位： 0：唤醒事件无效 1：唤醒事件有效
4	IWDT_WKE	R/W	0	IWDT 中断唤醒使能位： 0：唤醒事件无效 1：唤醒事件有效
3	RTC_TAMP_WKE	R/W	0	RTC TAMPER 中断唤醒使能位： 0：唤醒事件无效 1：唤醒事件有效 注意:深度待机模式下是 PC13 唤醒
2	RTC_ALARM_WKE	R/W	0	RTC ALARM 中断唤醒使能位： 0：唤醒事件无效 1：唤醒事件有效
1	PA0_WKE	R/W	0	PA0 管脚复位唤醒使能位： 0：唤醒事件无效 1：唤醒事件有效
0	RSTN_WKE	R/W	0	外部管脚复位唤醒使能位（下降沿唤醒）： 0：唤醒事件无效 1：唤醒事件有效

5.3.3 POWERUP 稳定时间配置寄存器（PMU_PUSTCR）

偏移地址：0x08

复位值：0x0000 007F

位	名称	属性	复位值	描述
31:7	RSV	R	0	保留
6:0	CNT_VALUE	R/W	7'h7F	MLDO 稳定时间设置: 默认为 128 个 32K 时钟周期（4ms 左右），实际应用可以设置为 3 或者更小。

5.3.4 PDR/BOR/LVD 配置寄存器（PMU_VDCR）

偏移地址：0x0C

复位值：0x0000 A701

位	名称	属性	复位值	描述
31:18	RSV	R	0	保留
17	LVD_INT_EN	R/W	0	LVD 中断使能寄存器： 0：关闭 1：使能
16	LVD_RST_EN	R/W	0	LVD 复位使能寄存器： 0：关闭 1：使能
15:12	LVDS	R/W	4'hA	LVD settings 参考表 5-3
11:8	BORS	R/W	4'h7	BOR settings 参考表 5-4
7:6	PDRS	R/W	2'b00	PDR settings 参考表 5-5
5	RSV	R	0	保留
4	LVD_FILTER_EN	R/W	0	LVD 滤波使能位： 0：禁止滤波 1：使能滤波（2 个 32K 周期）
3	RSV	R/W	0	保留
2	LVD_EN	R/W	0	LVD 低电压检测使能寄存器： 0：不打开 1：打开
1	BOR_EN	R/W	0	BOR 复位控制寄存器： 0：不打开 BOR 检测，不产生复位 1：打开 BOR 检测，可产生复位 注意：使用中此位必须设置为 1，打开 BOR 检测复位功能。
0	PDR_EN	R/W	1	掉电复位控制寄存器： 0：不打开掉电检测，不产生复位 1：打开掉电检测，可产生复位 注意：工作模式此位需要一直为 1。修改为 0，只有测试模式才可能用。

表 5-3：LVD（VDT）Settings

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Operation Current	-	-	-	40	-	nA
LVD trigger point	V_{LVD}	LVDS[3:0] = 0000	-	1.59	-	V
		LVDS[3:0] = 0001	-	1.68	-	V
		LVDS[3:0] = 0010	-	1.78	-	V
		LVDS[3:0] = 0011	-	1.88	-	V
		LVDS[3:0] = 0100	-	1.98	-	V
		LVDS[3:0] = 0101	-	2.08	-	V
		LVDS[3:0] = 0110	-	2.18	-	V
		LVDS[3:0] = 0111	-	2.28	-	V
		LVDS[3:0] = 1000	-	2.38	-	V
		LVDS[3:0] = 1001	-	2.48	-	V
		LVDS[3:0] = 1010	-	2.58	-	V
		LVDS[3:0] = 1011	-	2.67	-	V
		LVDS[3:0] = 1100	-	2.77	-	V
		LVDS[3:0] = 1101	-	2.87	-	V
		LVDS[3:0] = 1110	-	2.97	-	V
		LVDS[3:0] = 1111	-	3.07	-	V

表 5-4：BOR Settings

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Operation Current	-	-	-	40	-	nA
BOR trigger point	V_{BOR}	BORS[3:0] = 0000	-	1.56	-	V
		BORS[3:0] = 0001	-	1.66	-	V
		BORS[3:0] = 0010	-	1.75	-	V
		BORS[3:0] = 0011	-	1.85	-	V
		BORS[3:0] = 0100	-	1.95	-	V
		BORS[3:0] = 0101	-	2.05	-	V
		BORS[3:0] = 0110	-	2.14	-	V
		BORS[3:0] = 0111	-	2.24	-	V
		BORS[3:0] = 1000	-	2.34	-	V
		BORS[3:0] = 1001	-	2.43	-	V
		BORS[3:0] = 1010	-	2.53	-	V
		BORS[3:0] = 1011	-	2.63	-	V
		BORS[3:0] = 1100	-	2.73	-	V
		BORS[3:0] = 1101	-	2.83	-	V
		BORS[3:0] = 1110	-	2.92	-	V
		BORS[3:0] = 1111	-	3.02	-	V

表 5-5：PDR Settings

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Operation Current	-	-	-	40	-	nA
PDR Rising trigger point	V_{PDR_R}	PDRS[1:0] = 00	-	1.57	-	V
		PDRS[1:0] = 01	-	1.77	-	V
		PDRS[1:0] = 10	-	1.87	-	V
		PDRS[1:0] = 11	-	1.97	-	V
PDR Falling trigger point	V_{PDR_F}	PDRS[1:0] = 00	-	1.49	-	V
		PDRS[1:0] = 01	-	1.67	-	V
		PDRS[1:0] = 10	-	1.77	-	V
		PDRS[1:0] = 11	-	1.87	-	V
PDR trigger point hysteresis voltage	V_{PDR_HY}	PDRS[1:0] = 00	-	0.08	-	V
		PDRS[1:0] = 01	-	0.1	-	V

Parameter	Symbol	Condition	Min	Typ	Max	Unit
		PDRS[1:0] = 10	-	0.1	-	V
		PDRS[1:0] = 11	-	0.1	-	V

5.3.5 系统辅助状态寄存器（PMU_SASR）

偏移地址：0x14

复位值：0x0000 0234

位	名称	属性	复位值	描述
31:14	RSV	R	0	保留
13	LVD_FLAG	R/W	0	LVD 复位状态标志位： 0：未发生复位 1：发生复位 注意：此位被 RST_FLAG_CLR 位写 1 清除。只有 LVD 复位 PMU 时有用。
12	BOR_FLAG	R/W	0	BOR 复位状态标志位： 0：未发生复位 1：发生复位 注意：此位被 RST_FLAG_CLR 位写 1 清除。只有 BOR 复位 PMU 时有用
11	RSV	R	0	保留
10	XTL_RSTN_FLAG	R	0	XTL 时钟异常状态标志位： 0：未发生异常 1：发生异常 注意： 1）此位被 RST_FLAG_CLR 位写 1 清除 2）XTL 检测电路功能，建议是 XTL 作为 BBU 模块时钟时才打开。发生复位后时钟自动切换到 RCL。另外此状态会作为中断信号送到 CPU 核，和 RTC TAMPER 中断共用一个中断编号
9	CORE_PDN_FLAG	R/W	1	CORE 掉电复位状态标志位： 0：未发生复位 1：发生复位 注意：此位被 RST_FLAG_CLR 位写 1 清除
8	PDR_FLAG	R/W	0	PDR 复位状态标志位： 0：未发生复位 1：发生复位 注意：此位被 RST_FLAG_CLR 位写 1 清除
7	RST_FLAG_CLR	W	0	清除复位标志位： 0：不清除 1：清除复位标识位
6	XTL_DT_EN	R/W	0	XTL 时钟检测电路使能控制位： 0：不使能检测功能 1：XTL 时钟检测功能使能
5	LVD_PMU_EN	R/W	1	LVD 复位信号是否可以复位 PMU 状态机： 0：不能复位 PMU 状态机 1：可以复位 PMU 状态机

位	名称	属性	复位值	描述
4	BOR_PMU_EN	R/W	1	BOR 复位信号是否可以复位 PMU 状态机： 0：不能复位 PMU 状态机 1：可以复位 PMU 状态机
3	EFC_LOW_VDD_EN	R/W	0	低电压警告有效控制位。 0：低电压警告无效 1：低电压警告使能
2:1	RSV	R/W	2'b10	保留
0	BAT_SEL	R/W	0	外部辅助电池（Voltage Battery）状态寄存器： 0：没有外部辅助电池供电（只有一个主电源给所有的系统芯片供电） 1：存在外部辅助电池供电（系统芯片有 2 个外部供电电源）。 如果是外部 VDDH 异常掉电，此位 PDR_EN 建议配置为 1。如果是芯片主动进入低功耗模式，然后 VDDH 掉电，PDR_EN 建议设置为 0。

5.3.6 XTL 配置寄存器（PMU_XTLCR）

偏移地址：0x1C

复位值：0x0000 9240

位	名称	属性	复位值	描述
31:17	RSV	R	0	保留
15:13	ISET	R/W	3'b100	Oscillator current
12:10	RSET	R/W	3'b100	Bias resistance
9:7	GSET	R/W	3'b100	Gain of oscillator
6:4	FBRSET	R/W	3'b100	Feedback resistance
3:2	RSV	R	0	保留
1	XTL_EN	R/W	0	XTL 使能寄存器: 0：不使能 1：使能
0	LSCLK_SEL	R/W	0	低速时钟选择寄存器: 0：选择 RCL 1：选择 XTL

5.3.7 RCL 配置寄存器（PMU_RCLCR）

偏移地址：0x20

复位值：0xXXXX XXXX

位	名称	属性	复位值	描述
31:1	RSV	-	0xXXXX XXXX	保留
0	RCL_PD	R/W	0	RCL 时钟关闭寄存器： 0：RCL 打开 1：RCL 关闭（不建议关闭） 注：RCL 也受其他逻辑控制。深度待机（deep standby）模式下，自动关闭 RCL。有外部唤醒时自动打开 RCL。

5.3.8 功能时钟控制寄存器（PMU_FCCR）

偏移地址：0x24

复位值：0x0000 004F

位	名称	属性	复位值	描述
31:7	RSV	R	0	保留
6	RTCEN	R/W	1	RTC 控制器时钟使能： 0：关闭时钟 1：使能时钟
5:4	RSV	R	0	保留
3	LPTIM1EN	R/W	1	LPTIM1 控制器时钟使能： 0：关闭时钟 1：使能时钟
2	LPTIM0EN	R/W	1	LPTIM0 控制器时钟使能： 0：关闭时钟 1：使能时钟
1	IWDTEN	R/W	1	IWDT 控制器时钟使能： 0：关闭时钟 1：使能时钟
0	LPUARTEN	R/W	1	LPUART 控制器时钟使能： 0：关闭时钟 1：使能时钟

5.3.9 功能复位控制寄存器（PMU_FRCR）

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:7	RSV	R	0	保留
6	RTCRST	R/W	0	RTC 控制器复位使能： 0：不复位 1：复位有效
5:4	RSV	R	0	保留

位	名称	属性	复位值	描述
3	LPTIM1RST	R/W	0	LPTIM1 控制器复位使能： 0：不复位 1：复位有效
2	LPTIM0RST	R/W	0	LPTIM0 控制器复位使能： 0：不复位 1：复位有效
1	IWDTRST	R/W	0	IWDT 控制器复位使能： 0：不复位 1：复位有效
0	LPUARTRST	R/W	0	LPUART 控制器复位使能： 0：不复位 1：复位有效

5.3.10 配置保护寄存器（PMU_CPR）

偏移地址：0x50

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	R	0	保留
15:0	CONFIGEN	R/W	0	配置保护寄存器： 写 16'hABCD，使能 PMU 寄存器写操作。只有使能后才能对 PMU 相关配置寄存器进行配置写操作。 写 16'h459E，关闭 PMU 寄存器写操作。

6 复位和时钟模块（RCM）

6.1 时钟单元（Clock Logic）

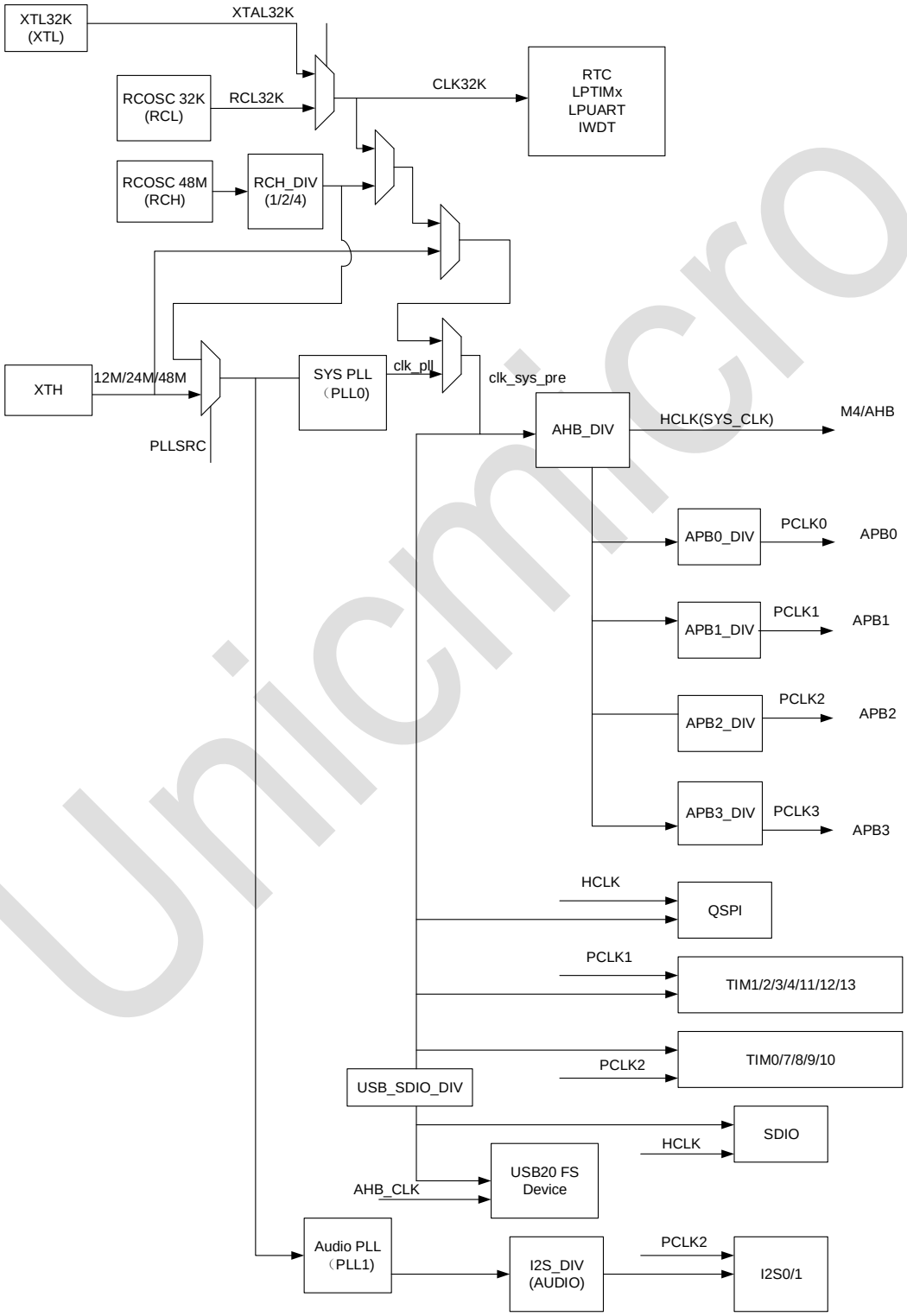


图 6-1：时钟单元

系统存在 4 个时钟源：

- 一个频率为 48MHz 的高精度内部时钟 RCH
- 一个频率为 32KHz 的内部时钟 RCL
- 一个频率为 32.768KHz 的外部晶体时钟 XTL
- 支持外部晶体时钟 XTH

内置 2 个 PLL，提供系统时钟、音频时钟和 USB 时钟。

时钟控制单元提供了一系列频率的时钟功能，包括一个内部 48M RC 振荡器时钟(RCH)、一个外部高速晶体振荡器时钟(XTH)、一个内部 32K RC 振荡器时钟(RCL)、一个外部低速晶体振荡器时钟(XTL)、两个锁相环(PLL)、一个 XTH 时钟监视器、时钟预分频器、时钟多路复用器和时钟门控电路。

AHB、APB、Cortex™-M4 源自系统时钟(SYS_CLK)，系统时钟的时钟源可以选择 XTH、RCH、PLL 或低频 32KHz 的 RCL/XTL 时钟。独立看门狗定时器有采用低频的时钟源（RCL 或者 XTL），实时时钟(RTC)使用 RCL 或 XTL 作为时钟源。

注：

- 如需要支持音频，可配置 PLL 小数分频。
- 对 EMC 要求严格应用（电机等），可打开 PLL 扩频功能。

6.2 复位单元（Reset Logic）

芯片复位控制包括二种控制方式：

- 电源复位
- 系统复位

6.2.1 电源复位

电源复位又称之为冷复位，其复位除了备份域的所有系统。系统复位将复位除了备份域之外的其余部分，包括处理器内核和外设 IP。备份域复位将复位备份域区域。复位能够被外部信号、内部事件和复位发生器触发。

当发生以下任一事件时，将产生电源复位：

- 上电/掉电复位（POR/PDR 复位）
- 欠压复位（BOR 复位）
- 从待机模式中返回后由内部复位发生器产生

电源复位作用于除备份域以外的寄存器。电源复位为低电平有效，当内部 LDO 电源基准准备好提供 Core 电压（比如 1.1V）时，电源复位电平将变为无效。复位入口向量被固定在存储器映射的地址 0xXXXX_0004。

6.2.2 系统复位

当发生以下任一事件时，将产生系统复位：

- 上电复位（POR）
- 外部引脚复位（RESETN）
- M4 窗口看门狗计数终止（WWDT_RSTN）
- 独立看门狗计数终止（IWDT_RSTN）
- Cortex™-M4 的中断应用和复位控制寄存器中的 SYSRESETREQ 位置‘1’

系统复位将复位除了备份域之外的其余部分，包括处理器内核和外设 IP。

本产品有 7 个复位信号来源，均为低电平有效，每个复位信号都可以让 CPU 重新运行，绝大多数寄存器会被重新复位，程序计数器（PC）会复位指向地址 0x00000000。

表 6-1：系统里复位信号来源

复位信号名称	有效电平	说明
RESETN	低	全局硬件复位引脚，复位整个芯片(除 BBU 等寄存器)。
POR_RSTN	低	1.1V 上电复位，复位整个芯片。
BOR_RSTN	低	掉电复位，复位整个芯片。
SOFT_RSTN	低	全局软复位控制寄存器，复位系统。
IWDT_RSTN	低	看门狗定时器复位，复位系统。
WWDT_RSTN	低	窗口看门狗定时器复位，复位系统。
Block reset signals	低	复位单个模块。

表 6-2：复位方式

复位方式	产生条件
上电和掉电复位（POR&PDR）	V_{DDH}（1.8~3.6V）上电和内部 CORE 电压上电
RESETN 引脚复位	RESETN 外部管脚输入低电平
欠压复位（BOR）	V_{DDH} 电压降至低于 VBOR 电压
低电压检测复位（LVD）	V_{DDH} 电压降至低于 LVD 电压
窗口看门狗复位（WWDT）	-
独立看门狗复位（IWDT）	-
掉电唤醒复位	通过设置掉电模式产生的复位，内核在掉电唤醒事件发生后从复位状态唤醒
软件复位	-
外部高速振荡器异常停振复位	外部高速振荡器异常停振时产生的复位

6.3 寄存器描述

寄存器基地址：0x40B0_1000

表 6-3：RCM 寄存器列表

偏移地址	名称	描述
0x000	RCM_CR0	时钟控制寄存器 0
0x008	RCM_PLL0CFGR0	PLL0 配置寄存器 0
0x00C	RCM_PLL0CFGR1	PLL0 配置寄存器 1
0x010	RCM_PLL0CFGR2	PLL0 配置寄存器 2
0x014	RCM_PLL1CFGR0	PLL1 配置寄存器 0
0x018	RCM_PLL1CFGR1	PLL1 配置寄存器 1
0x01C	RCM_PLL1CFGR2	PLL1 配置寄存器 2
0x020	RCM_PLLTSR	PLL 稳定时间设置寄存器
0x030	RCM_CFGR0	时钟配置寄存器 0
0x034	RCM_CFGR1	时钟配置寄存器 1
0x040	RCM_CIFR	时钟中断标志寄存器
0x044	RCM_CIER	时钟中断使能寄存器
0x060	RCM_AHBCKENR	AHB 外围模块时钟使能寄存器
0x064	RCM_AHB0CKENR	AHB0 外围模块时钟使能寄存器
0x068	RCM_AHB1CKENR	AHB1 外围模块时钟使能寄存器
0x06C	RCM_APB0CKENR	APB0 外围模块时钟使能寄存器
0x070	RCM_APB1CKENR	APB1 外围模块时钟使能寄存器
0x074	RCM_APB2CKENR	APB2 外围模块时钟使能寄存器
0x078	RCM_APB3CKENR	APB3 外围模块时钟使能寄存器
0x07C	RCM_AHBRSTR	AHB 外围模块复位控制寄存器
0x080	RCM_AHB0RSTR	AHB0 外围模块复位控制寄存器
0x084	RCM_AHB1RSTR	AHB1 外围模块复位控制寄存器
0x088	RCM_APB0RSTR	APB0 外围模块复位控制寄存器
0x08C	RCM_APB1RSTR	APB1 外围模块复位控制寄存器
0x090	RCM_APB2RSTR	APB2 外围模块复位控制寄存器
0x094	RCM_APB3RSTR	APB3 外围模块复位控制寄存器
0x098	RCM_SOFTRSTR	软件复位寄存器
0x0E0	RCM_RFR	复位标识寄存器
0x0E4	RCM_EXRSTFER	外部复位管脚滤波使能寄存器
0x0F0	RCM_RCMPR	RCM 寄存器配置写保护寄存器

6.3.1 时钟控制寄存器 0（RCM_CR0）

地址：0x000

复位值：0x0020 0421

位	名称	属性	复位值	描述
31:29	RSV	R	0	保留

位	名称	属性	复位值	描述
28	PLLSRC	R/W	0	PLL0/PLL1 输入时钟源选择： 0：选择 RCH 作为 PLL0/PLL1 输入时钟源 1：选择 XTH 作为 PLL0/PLL1 输入时钟源 注：只有 PLL0/PLL1 不工作时才能配置。
27	PLL1STB	R	0	PLL1（Audio 音频）稳定标志位： 0：不稳定 1：稳定
26	PLL1EN	R/W	0	PLL1（Audio 音频）使能信号： 0：关闭 1：使能
25	PLL0STB	R	0	PLL0 稳定标志位： 0：不稳定 1：稳定
24	PLL0EN	R/W	0	PLL0 使能信号： 0：关闭 1：使能
23:22	XTH_SF	R/W	0	XTH 振荡频率选择（SF0, SF1）： 2'b00：1MHz~4MHz 2'b01：4.1MHz~12MHz 2'b10：12.1MHz~24MHz 2'b11：24.1MHz~48MHz
21:20	XTHSS	R/W	2'b10	外部高速时钟 XTH 稳定时间选择： 2'b00：4096 个周期 2'b01：16384 个周期 2'b10：32768 个周期 2'b11：65535 个周期
19	RSV	R	0	保留
18	XTH_BYP	R/W	0	高速晶体振荡器 XTH 旁路选择信号： 1：不需要外部晶体，直接从管脚输入时钟 0：从 XTH 时钟出来
17	XTH_STB	R	0	高速晶体振荡器 XTH 稳定标志位： 1：稳定 0：未稳定
16	XTH_EN	R/W	0	高速晶体振荡器（XTH）时钟使能： 0：关闭 1：使能
15:14	XTH_SFRB	R/W	0	XTH 反馈电阻阻值选择。 电阻配置信号，对应关系如下： 00：500kΩ 01：100kΩ 1X：HiZ
13:11	RSV	R	-	保留
10	RCH_STB	R	1	内部高速时钟 RCH 稳定标志位： 1：代表 RCH 已经稳定，可以被内部电路使用 0：代表 RCH 未稳定，不可以被内部电路使用

位	名称	属性	复位值	描述
9:8	RCHSS	R/W	2’b00	内部高速时钟 RCH 稳定时间选择： 11：256 个周期 10：64 个周期 01：16 个周期 00：4 个周期 注：STOP mode 可将此值改小，加快唤醒时间。
7:6	RSV	R	0	保留
5:4	XTLSS	R/W	2’h2	外部低速时钟 XTL 稳定时间选择： 2'b00：1024 个周期 2'b01：4096 个周期 2'b10：16384 个周期 2'b11：32768 个周期
3	XTH_RST_INT_EN	R/W	0	XTH 时钟异常复位或中断使能位： 0：禁止 XTH 时钟异常产生复位或中断 1：容许时钟异常产生复位或者中断 注：此寄存器不会被 XTH 时钟异常复位掉。需写 0 清除此寄存器的值，清除 XTH 停振中断。
2	XTH_STOP_SEL	R/W	0	XTH 时钟异常状态选择位： 0：时钟异常产生中断 1：时钟异常产生复位 注：此寄存器不会被 XTH 时钟异常复位掉。需写 0 清除此寄存器的值。
1	XTH_MEN	R/W	0	XTH 时钟监视使能： 0：禁止 1：使能
0	RCH_EN	R/W	1	内部高速时钟 RCH 使能信号： 0：关闭 1：使能 注：当系统进入 standby 或者 DeepStandby 时，RCH 时钟会自动关闭。

6.3.2 PLL0 配置寄存器 0（RCM_PLL0CFGR0）

地址：0x008

复位值：0x0001 0388

位	名称	属性	复位值	描述
31:21	SSRATE	R/W	0	扩频斜率设置
20:15	PLL0_DM	R/W	6'h2	PLL0_DM 信号
14:5	PLL0_DN	R/W	10'h1C	PLL0_DN 信号：写入值需要大于等于 16.
4:2	PLL0_DP	R/W	3'h2	PLL0_DP 信号

位	名称	属性	复位值	描述
1	PLL0_BYP	R/W	0	PLL0 Bypass 控制信号： 0：时钟从 PLL VCO 来 1：绕开 PLL VCO 时钟，从 REF CLK 来
0	PLL0_CFGEN	R/W	0	PLL0 配置使能信号（有参数改变）： 0：无参数改变 1：有配置改变 注：PLL 参数配置有改变后，使能此位，等待 PLL 稳定时间才能切换 PLL 时钟。

注：

1. 对于整数分频，PLL0 输出时钟的公式如下：

   $f_{CLKO} = f_{refclk} * PLL0\_DN / (PLL0\_DM * PLL0\_DP)$

   注：$f_{refclk} * PLL0\_DN/PLL0\_DM$ 频率需大于 300MHz。

2. 对于小数分频，PLL0 输出时钟的公式如下：

   $f_{CLKO} = f_{refclk} * (PLL0\_DN + PLLI0\_FRAC / 2^{24}) / (PLL0\_DM * PLL0\_DP)$

   注：$f_{refclk} * PLL0\_DN/PLL0\_DM$ 频率需大于 300MHz。

6.3.3 PLL0 配置寄存器 1（RCM_PLL0CFGR1）

地址：0x00C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:26	RSV	R	0	保留
25:24	SSC_MODE	R/W	0	PLL0 扩频模式选择寄存器: 2'b00：下扩频 2'b01：中扩频 2'b10：上扩频 2'b11：保留
23:0	SLOPE	R/W	0	PLL0 扩频 Slope 系数

6.3.4 PLL0 配置寄存器 2（RCM_PLL0CFGR2）

地址：0x010

复位值：0x0800 0000

位	名称	属性	复位值	描述
31:30	RSV	R	0	保留
29:26	COUNT	R/W	4'h2	PLL 控制信号 COUNT[0]：dither 使能位（设置 1，dither 功能有效） COUNT[1]：PLL 小数部分逻辑复位控制寄存器（1：小数部分逻辑复位，0：小数部分逻辑复位释放）。小数模式和扩频模式，此位写 0。 COUNT[3:2]：保留

位	名称	属性	复位值	描述
25:24	MODE	R/W	0	操作模式选择： 2'b00：整数模式 2'b01：小数模式 2'b10：扩频模式 2'b11：保留
23:0	FRAC	R/W	0	PLL0 小数分频系数

6.3.5 PLL1 配置寄存器 0（RCM_PLL1CFGR0）

地址：0x014

复位值：0x0001 0390

位	名称	属性	复位值	描述
31:21	SSRATE	R/W	0	扩频斜率设置
20:15	PLL1_DM	R/W	6'h2	PLL1_DM 信号
14:5	PLL1_DN	R/W	10'h1C	PLL1_DN 信号：写入值需要大于等于 16。
4:2	PLL1_DP	R/W	3'h4	PLL1_DP 信号
1	PLL1_BYP	R/W	0	PLL1 Bypass 控制信号： 0：时钟从 PLL VCO 来 1：绕开 PLL VCO 时钟，从 REF CLK 来。
0	PLL1_CFGEN	R/W	0	PLL1 配置使能信号（有参数改变）： 0：无参数改变 1：有配置改变 注：PLL 参数配置有改变后，使能此位，等待 PLL 稳定时间才能切换 PLL 时钟

注：对于整数分频，PLL1 输出时钟的公式如下：

$f_{CLKO}=f_{refclk}$ * PLL1_DN/(PLL1_DM * PLL1_DP)

6.3.6 PLL1 配置寄存器 1（RCM_PLL1CFGR1）

地址：0x018

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:26	RSV	R	0	保留
25:24	SSC_MODE	R/W	0	PLL1 扩频模式选择寄存器： 2'b00：下扩频 2'b01：中扩频 2'b10：上扩频 2'b11：保留
23:0	SLOPE	R/W	0	PLL1 扩频 Slope 系数

6.3.7 PLL1 配置寄存器 2（RCM_PLL1CFGR2）

地址：0x01C

复位值：0x0800 0000

位	名称	属性	复位值	描述
31:30	RSV	R	0	保留
29:26	COUNT	R/W	4'h2	PLL 控制信号 COUNT[0]：dither 使能位（设置 1，dither 功能有效） COUNT[1]：PLL 小数部分逻辑复位控制寄存器（1：小数部分逻辑复位，0：小数部分逻辑复位释放）。小数模式和扩频模式，此位写 0。 COUNT[3:2]：保留
25:24	MODE	R/W	0	操作模式选择： 2'b00：整数模式 2'b01：小数模式 2'b10：扩频模式 2'b11：保留
23:0	FRAC	R/W	0	PLL1 小数分频系数

6.3.8 PLL 稳定时间设置寄存器（RCM_PLLTSR）

地址：0x020

复位值：0x0000 2EE0

位	名称	属性	复位值	描述
31:17	RSV	R	0	保留
16	PLL_LT_EN	R/W	0	PLL 输出时钟： 0：软件等待 PLL0STB 或者 PLL1STB 后才将系统时钟切换到 PLL（软件等待） 1：PLL 稳定后才能输出时钟应用于系统 注：如果 STOP MODE 系统时钟选择 PLL 时钟，同时希望唤醒后保持 PLL 时钟，此位需要设置为 1。
15:0	PLL_LT	R/W	16'h2EE0	PLL Lock 时钟设置寄存器： 16'h2EE0 (16'd12000)对应 PLL LOCK 时间 0.5ms（采用 24MHz 时钟计数）

6.3.9 时钟配置寄存器 0（RCM_CFGR0）

地址：0x030

复位值：0x0900 8040

位	名称	属性	复位值	描述
31:28	I2S_DIV	R/W	0	I2S_MCLK 时钟分频系数（基于 Audio PLL）： 4'b0xxx：不分频 4'b1000：2 分频 4'b1001：4 分频 4'b1010：8 分频 4'b1011：16 分频 4'b1100：32 分频 4'b1101：64 分频 4'b1110：128 分频 4'b1111：256 分频
27:24	APB3_DIV	R/W	4'b1001	APB3 时钟分频系数（基于 AHB 时钟）： 4'b0xxx：不分频 4'b1000：2 分频 4'b1001：4 分频 4'b1010：8 分频 4'b1011：16 分频 4'b1100：32 分频 4'b1101：64 分频 4'b1110：128 分频 4'b1111：256 分频
23:20	APB2_DIV	R/W	0	APB2 时钟分频系数（基于 AHB 时钟）： 4'b0xxx：不分频 4'b1000：2 分频 4'b1001：4 分频 4'b1010：8 分频 4'b1011：16 分频 4'b1100：32 分频 4'b1101：64 分频 4'b1110：128 分频 4'b1111：256 分频
19:16	APB1_DIV	R/W	0	APB1 时钟分频系数（基于 AHB 时钟）： 4'b0xxx：不分频 4'b1000：2 分频 4'b1001：4 分频 4'b1010：8 分频 4'b1011：16 分频 4'b1100：32 分频 4'b1101：64 分频 4'b1110：128 分频 4'b1111：256 分频
15:12	APB0_DIV	R/W	4'b1000	APB0 时钟分频系数（基于 AHB 时钟）： 4'b0xxx：不分频 4'b1000：2 分频 4'b1001：4 分频 4'b1010：8 分频 4'b1011：16 分频 4'b1100：32 分频 4'b1101：64 分频 4'b1110：128 分频 4'b1111：256 分频

位	名称	属性	复位值	描述
11:8	AHB_DIV	R/W	0	AHB 时钟分频系数（基于 clk_sys_pre 时钟）： 4'b0xxx：不分频 4'b1000：2 分频 4'b1001：4 分频 4'b1010：8 分频 4'b1011：16 分频 4'b1100：32 分频 4'b1101：64 分频 4'b1110：128 分频 4'b1111：256 分频
7:6	RCH_DIV	R/W	2'b01	内部高速时钟 RCH 分频系数： 上电预读时 RCH（48MHz）用固定的 4 分频，产生 12MHz 时钟。 2'b00：1 分频 2'b01：2 分频 2'b10：3 分频 2'b11：4 分频 注：正常工作是 24MHz 时钟。
5	I2S_SRC	R/W	0	I2S 时钟源选择： 0：PLL1 (AUDIO PLL)时钟输出作为 I2S 时钟 1：外部管脚 I2S_MCLK 输入时钟作为 I2S 时钟
4	RSV	R	0	保留
3:2	SYS_SWS	R	2'b00	系统时钟切换状态： 2'b00：选择 RCH 时钟作为系统时钟 2'b01：选择 XTH 时钟作为系统时钟 2'b10：选择 PLL0 输出时钟作为系统时钟 2'b11：选择 32kHz 低频时钟（RCL 或 XTL）作为系统时钟 注：存在 XTH 时钟没有，切换到 RCH 的情况
1:0	SYS_SW	R/W	2'b00	系统时钟切换： 2'b00：选择 RCH 时钟作为系统时钟 2'b01：选择 XTH 时钟作为系统时钟 2'b10：选择 PLL0 输出时钟作为系统时钟 2'b11：选择 32kHz 低频时钟（RCL 或 XTL）作为系统时钟

6.3.10 时钟配置寄存器 1（RCM_CFGR1）

地址：0x034

复位值：0xFF06 2002

位	名称	属性	复位值	描述
31:28	USART7_DIV	R/W	4’hF	USART7 分频时钟系数（基于 PCLK2 时钟）： 4’b0000：保留 4’b0001：2 分频 …… 4’b1111：16 分频

位	名称	属性	复位值	描述
27:24	USART6_DIV	R/W	4’hF	USART6 分频时钟系数（基于 PCLK0 时钟）： 4’b0000：保留 4’b0001：2 分频 …… 4’b1111：16 分频
23	USART7_CLK_SEL	R/W	0	USART7 slow clock 时钟源选择： 0：选择 RCL/XTL 低速时钟 1：选择 pclk2 时钟
22	USART6_CLK_SEL	R/W	0	USART6 slow clock 时钟源选择： 0：选择 RCL/XTL 低速时钟 1：选择 pclk0 时钟
21	TIM_CLK_SEL2	R/W	0	TIM0/7/8/9/10 的时钟源选择： 0：TIM0/7/8/9/10 时钟源选择 apb2_clk 1：TIM0/7/8/9/10 时钟源选择 SYSPLL。 注意：如果设置为 1, apb2_clk=HCLK=SYSPLL/2， 比如 SYSPLL 时钟是 336MHz,则 apb2_clk=HCLK=168MHz,TIM0/7/8/9/10 的输入时钟是 336MHz
20	TIM_CLK_SEL1	R/W	0	TIM1/2/3/4/11/12/13 的时钟源选择.TIM5/6 时钟源一直为 apb1_clk： 0：TIM1/2/3/4/11/12/13 时钟源选择 apb1_clk. 1：TIM1/2/3/4/11/12/13 时钟源选择 SYSPLL。 注意：如果设置为 1, apb1_clk=HCLK=SYSPLL/2， 比如 SYSPLL 时钟是 336MHz,则 apb1_clk=HCLK=168MHz,TIM1/2/3/4/11/12/13 的输入时钟是 336MHz（TIM5/6 除外）
19	RSV	R	0	保留
18:16	USB_SDIO_DIV （48M_DIV）	R/W	3'b110	USB 和 SDIO 模块工作时钟的分频系数设置（如用 USB 时，必须配置合适的分频系数，产生 48MHz 时钟）： 3'b000：不分频 3'b001：2 分频 3'b010：3 分频 3'b011：4 分频 3'b100：5 分频 3'b101：6 分频 3'b110：7 分频 3'b111：8 分频 注：用于 USB，SDIO 模块
15	RSV	R	0	保留

位	名称	属性	复位值	描述
14:12	MCO1_DIV	R/W	3'b010	MCO1 分配系数： 3'b000：不分频 3'b001：2 分频 3'b010：4 分频 3'b011：8 分频 3'b100：16 分频 3'b101：32 分频 3'b110：64 分频 3'b111：128 分频
11	RSV	R	0	保留
10:8	MCO1	R/W	3'b000	MCO1 时钟输出： 3'b000：选择 RCH_DIV 时钟（比如 24MHz）输出到 MCO1 引脚（PC9） 3'b001：选择 XTH 时钟输出到 MCO1 引脚(PC9) 3'b010：选择低速时钟（RCL 或者 XTL）时钟输出到 MCO1 引脚(PC9) 3'b011：选择 XTL 时钟输出到 MCO1 引脚(PC9) 3'b100：选择 PLL0 时钟输出到 MCO1 引脚(PC9) 3'b101：选择 PLL1 时钟输出到 MCO1 引脚(PC9) 3'b110：保留 3'b111：选择 AHB 时钟输出到 MCO1 引脚(PC9)
7	RSV	R	0	保留
6:4	MCO0_DIV	R/W	3'b000	MCO0 分配系数： 3'b000：不分频 3'b001：2 分频 3'b010：4 分频 3'b011：8 分频 3'b100：16 分频 3'b101：32 分频 3'b110：64 分频 3'b111：128 分频
3	RSV	R	0	保留

位	名称	属性	复位值	描述
2:0	MCO0	R/W	3'b010	MCO0 时钟输出： 3'b000：选择 RCH_DIV 时钟（比如 24MHz）输出到 MCO0 引脚(PA8) 3'b001：选择 XTH 时钟输出到 MCO0 引脚(PA8) 3'b010：选择低速时钟（RCL 或者 XTL）时钟输出到 MCO0 引脚(PA8) 3'b011：选择 XTL 时钟输出到 MCO0 引脚(PA8) 3'b100：选择 PLL0 时钟输出到 MCO0 引脚(PA8) 3'b101：选择 PLL1 时钟输出到 MCO0 引脚(PA8) 3'b110：保留 3'b111：选择 AHB 时钟输出到 MCO0 引脚(PA8)

6.3.11 时钟中断标志寄存器（RCM_CIFR）

地址：0x040

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	R	0	保留
7	CSSF	R/W	0	时钟安全系统标志位.写 1 清除中断。 0：XTH 时钟正常 1：XTH 时钟输出故障
6	RSV	R	0	保留
5	PLL1RDYF	R/W	0	PLL1 时钟锁定中断标志位。写 1 清除中断。 0：PLL1 输出时钟未锁定 1：PLL1 输出时钟锁定
4	PLL0RDYF	R/W	0	PLL0 时钟锁定中断标志位。写 1 清除中断。 0：PLL0 输出时钟未锁定 1：PLL0 输出时钟锁定
3	HSERDYF	R/W	0	高速外部晶体振荡器时钟 XTH 就绪中断标志位。写 1 清除中断。 0：XTH 时钟未就绪 1：XTH 时钟就绪
2	RSV	R	0	保留
1	LSERDYF	R/W	0	低速外部晶体振荡器时钟 XTL 就绪中断标志位。写 1 清除中断。 0：XTL 时钟未就绪 1：XTL 时钟就绪
0	RSV	R	0	保留

6.3.12 时钟中断使能寄存器（RCM_CIER）

地址：0x044

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	R	0	保留
7	CSSIE	R/W	0	时钟安全系统（XTH 时钟故障）中断使能位： 0：中断禁止 1：中断使能
6	RSV	R	0	保留
5	PLL1RDYIE	R/W	0	PLL1 时钟锁定中断使能位： 0：中断禁止 1：中断使能
4	PLL0RDYIE	R/W	0	PLL0 时钟锁定中断使能位： 0：中断禁止 1：中断使能
3	HSERDYIE	R/W	0	高速外部晶体振荡器时钟 XTH 就绪中断使能位： 0：中断禁止 1：中断使能
2	RSV	R	0	保留
1	LSERDYIE	R/W	0	低速外部晶体振荡器时钟 XTL 就绪中断使能位： 0：中断禁止 1：中断使能
0	RSV	R	0	保留

6.3.13 AHB 外围时钟使能寄存器（RCM_AHBCKENR）

地址：0x060

复位值：0x0001 0700

位	名称	属性	复位值	描述
31:18	RSV	R	0	保留
17	EMCEN	R/W	0	EMC 控制器时钟使能： 0：关闭时钟 1：使能时钟
16	QSPIEN	R/W	1	QSPI 控制器时钟使能： 0：关闭时钟 1：使能时钟
15:11	RSV	R	0	保留
10	RSV	R/W	1	保留
9	SRAM1EN	R/W	1	SRAM1 控制器时钟使能： 0：关闭时钟 1：使能时钟
8	SRAM0EN	R/W	1	SRAM0 控制器时钟使能： 0：关闭时钟 1：使能时钟
7:0	RSV	R	0	保留

6.3.14 AHB0 外围时钟使能寄存器（RCM_AHB0CKENR）

地址：0x064

复位值：0x0000 0011

位	名称	属性	复位值	描述
31:13	RSV	R	0	保留
12	DCMIEN	R/W	0	DCMI 控制器时钟使能： 0：关闭时钟 1：使能时钟
11	RSV	R	0	保留
10	EMACEN	R/W	0	EMAC 控制器时钟使能： 0：关闭时钟 1：使能时钟
9	RSV	R	0	保留
8	SDIOEN	R/W	0	SDIO 控制器时钟使能： 0：关闭时钟 1：使能时钟
7	RSV	R	0	保留
6	DMA1EN	R/W	0	DMA1 控制器时钟使能： 0：关闭时钟 1：使能时钟
5	DMA0EN	R/W	0	DMA0 控制器时钟使能： 0：关闭时钟 1：使能时钟
4	CRCEN	R/W	1	CRC 控制器时钟使能： 0：关闭时钟 1：使能时钟
3:1	RSV	R	0	保留
0	USBEN	R/W	1	USB (Device)控制器时钟使能： 0：关闭时钟 1：使能时钟

6.3.15 AHB1 外围时钟使能寄存器（RCM_AHB1CKENR）

地址：0x068

复位值：0x0000 01FF

位	名称	属性	复位值	描述
31:16	RSV	R	0	保留
15	CORDICEN	R/W	0	CORDIC 控制器时钟使能： 0：关闭时钟 1：使能时钟
14	RSV	R	0	保留
13	SHAEN	R/W	0	SHA 控制器时钟使能： 0：关闭时钟 1：使能时钟

位	名称	属性	复位值	描述
12	AESEN	R/W	0	AES 控制器时钟使能： 0：关闭时钟 1：使能时钟
11:8	RSV	R	0	保留
7	GPIOHEN	R/W	1	GPIOH 控制器时钟使能： 0：关闭时钟 1：使能时钟
6:5	RSV	R	0	保留
4	GPIOEEN	R/W	1	GPIOE 控制器时钟使能： 0：关闭时钟 1：使能时钟
3	GPIODEN	R/W	1	GPIOD 控制器时钟使能： 0：关闭时钟 1：使能时钟
2	GPIOCEN	R/W	1	GPIOC 控制器时钟使能： 0：关闭时钟 1：使能时钟
1	GPIOBEN	R/W	1	GPIOB 控制器时钟使能： 0：关闭时钟 1：使能时钟
0	GPIOAEN	R/W	1	GPIOA 控制器时钟使能： 0：关闭时钟 1：使能时钟

6.3.16 APB0 外围时钟使能寄存器（RCM_APB0CKENR）

地址：0x06C

复位值：0x0000 0400

位	名称	属性	复位值	描述
31:13	RSV	R	0	保留
12	USART6EN	R/W	0	USART6 控制器时钟使能： 0：关闭时钟 1：使能时钟
11	UART1EN	R/W	0	UART1 控制器时钟使能： 0：关闭时钟 1：使能时钟
10	WWDTEN	R/W	1	WWDT 控制器时钟使能： 0：关闭时钟 1：使能时钟
9:0	RSV	R	0	保留

6.3.17 APB1 外围时钟使能寄存器（RCM_APB1CKENR）

地址：0x070

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:21	RSV	R	0	保留
20	SPI0EN	R/W	0	SPI0 控制器时钟使能： 0：关闭时钟 1：使能时钟
19	RSV	R	0	保留
18	I2C2EN	R/W	0	I2C2 控制器时钟使能： 0：关闭时钟 1：使能时钟
17	I2C1EN	R/W	0	I2C1 控制器时钟使能： 0：关闭时钟 1：使能时钟
16	I2C0EN	R/W	0	I2C0 控制器时钟使能： 0：关闭时钟 1：使能时钟
15	RSV	R/W	0	保留
14	RSV	R	0	保留
13	DACEN	R/W	0	DAC0/1 控制器时钟使能： 0：关闭时钟 1：使能时钟
12	TIM13EN	R/W	0	TIM13EN 控制器时钟使能： 0：关闭时钟 1：使能时钟
11	TIM12EN	R/W	0	TIM12 控制器时钟使能： 0：关闭时钟 1：使能时钟
10	TIM11EN	R/W	0	TIM11EN 控制器时钟使能： 0：关闭时钟 1：使能时钟
9	TIM6EN	R/W	0	TIM6 控制器时钟使能： 0：关闭时钟 1：使能时钟
8	TIM5EN	R/W	0	TIM5 控制器时钟使能： 0：关闭时钟 1：使能时钟
7	TIM4EN	R/W	0	TIM4 控制器时钟使能： 0：关闭时钟 1：使能时钟
6	TIM3EN	R/W	0	TIM3 控制器时钟使能 0：关闭时钟 1：使能时钟
5	TIM2EN	R/W	0	TIM2 控制器时钟使能： 0：关闭时钟 1：使能时钟
4	TIM1EN	R/W	0	TIM1 控制器时钟使能： 0：关闭时钟 1：使能时钟

位	名称	属性	复位值	描述
3	UART5EN	R/W	0	UART5 控制器时钟使能： 0：关闭时钟 1：使能时钟
2	UART4EN	R/W	0	UART4 控制器时钟使能： 0：关闭时钟 1：使能时钟
1	UART3EN	R/W	0	UART3 控制器时钟使能： 0：关闭时钟 1：使能时钟
0	UART2EN	R/W	0	UART2 控制器时钟使能： 0：关闭时钟 1：使能时钟

6.3.18 APB2 外围时钟使能寄存器（RCM_APB2CKENR）

地址：0x074

复位值：0x0000 8000

位	名称	属性	复位值	描述
31:16	RSV	R	0	保留
15	UART0EN	R/W	1	UART0 控制器时钟使能： 0：关闭时钟 1：使能时钟
14	USART7EN	R/W	0	USART7 控制器时钟使能： 0：关闭时钟 1：使能时钟
13	TIM10EN	R/W	0	TIM10 控制器时钟使能： 0：关闭时钟 1：使能时钟
12	TIM9EN	R/W	0	TIM9 控制器时钟使能： 0：关闭时钟 1：使能时钟
11	TIM8EN	R/W	0	TIM8 控制器时钟使能： 0：关闭时钟 1：使能时钟
10	TIM7EN	R/W	0	TIM7 控制器时钟使能： 0：关闭时钟 1：使能时钟
9	TIM0EN	R/W	0	TIM0 控制器时钟使能： 0：关闭时钟 1：使能时钟
8	SPI3EN	R/W	0	SPI3 控制器时钟使能： 0：关闭时钟 1：使能时钟
7	SPI2EN	R/W	0	SPI2 控制器时钟使能： 0：关闭时钟 1：使能时钟

位	名称	属性	复位值	描述
6	SPI1EN	R/W	0	SPI1 控制器时钟使能： 0：关闭时钟 1：使能时钟
5	ADC1EN	R/W	0	ADC1 控制器时钟使能： 0：关闭时钟 1：使能时钟
4	ADC0EN	R/W	0	ADC0 控制器时钟使能： 0：关闭时钟 1：使能时钟
3:2	RSV	R	0	保留
1	I2S1EN	R/W	0	I2S1 控制器时钟使能： 0：关闭时钟 1：使能时钟
0	I2S0EN	R/W	0	I2S0 控制器时钟使能： 0：关闭时钟 1：使能时钟

6.3.19 APB3 外围时钟使能寄存器（RCM_APB3CKENR）

地址：0x078

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:2	RSV	R	0	保留
1	CAN1EN	R/W	0	CAN1 控制器时钟使能： 0：关闭时钟 1：使能时钟
0	CAN0EN	R/W	0	CAN0 控制器时钟使能： 0：关闭时钟 1：使能时钟

6.3.20 AHB 外围复位使能寄存器（RCM_AHBRSTR）

地址：0x07C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:21	RSV	R	0	保留
20	LOCKUP_EN	R/W	0	M4 核 LOCKUP 复位使能： 0：不使能 1：使能（发生 lockup 复位会复位系统） 注：此位只会被 POR 和掉电复位影响，其他复位不会复位掉此位的值
19:18	RSV	R	0	保留

位	名称	属性	复位值	描述
17	EMCRST	R/W	0	EMC 控制器复位使能： 0：不复位 1：复位有效
16	QSPIRST	R/W	0	QSPI 控制器复位使能： 0：不复位 1：复位有效
15:10	RSV	R	0	保留
9	SRAM1RST	R/W	0	SRAM1 控制器复位使能： 0：不复位 1：复位有效
8	SRAM0RST	R/W	0	SRAM0 控制器复位使能： 0：不复位 1：复位有效
7:0	RSV	R	0	保留

6.3.21 AHB0 外围复位使能寄存器（RCM_AHB0RSTR）

地址：0x080

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:13	RSV	R	0	保留
12	DCMIRST	R/W	0	DCMI 控制器复位使能： 0：不复位 1：复位有效
11	RSV	R	0	保留
10	EMACRST	R/W	0	EMAC 控制器复位使能： 0：不复位 1：复位有效
9	RSV	R	0	保留
8	SDIORST	R/W	0	SDIO 控制器复位使能： 0：不复位 1：复位有效
7	RSV	R/W	0	保留
6	DMA1RST	R/W	0	DMA1 控制器复位使能： 0：不复位 1：复位有效
5	DMA0RST	R/W	0	DMA0 控制器复位使能： 0：不复位 1：复位有效
4	CRCRST	R/W	0	CRC 控制器复位使能： 0：不复位 1：复位有效
3:1	RSV	R	0	保留
0	USBRST	R/W	0	USB 控制器复位使能： 0：不复位 1：复位有效

6.3.22 AHB1 外围复位使能寄存器（RCM_AHB1RSTR）

地址：0x084

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	R	0	保留
15	CORDICRST	R/W	0	CORDIC 控制器复位使能： 0：不复位 1：复位有效
14	RSV	R	0	保留
13	SHARST	R/W	0	SHA 控制器复位使能： 0：不复位 1：复位有效
12	AESRST	R/W	0	AES 控制器复位使能： 0：不复位 1：复位有效
11:8	RSV	R	0	保留
7	GPIOHRST	R/W	0	GPIOH 控制器复位使能： 0：不复位 1：复位有效 注：应用中保持默认 0
6:5	RSV	R	0	保留
4	GPIOERST	R/W	0	GPIOE 控制器复位使能： 0：不复位 1：复位有效 注：应用中保持默认 0
3	GPIODRST	R/W	0	GPIOD 控制器复位使能： 0：不复位 1：复位有效 注：应用中保持默认 0
2	GPIOCRST	R/W	0	GPIOC 控制器复位使能： 0：不复位 1：复位有效 注：应用中保持默认 0
1	GPIOBRST	R/W	0	GPIOB 控制器复位使能： 0：不复位 1：复位有效 注：应用中保持默认 0
0	GPIOARST	R/W	0	GPIOA 控制器复位使能： 0：不复位 1：复位有效 注：应用中保持默认 0

6.3.23 APB0 外围复位使能寄存器（RCM_APB0RSTR）

地址：0x088

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:13	RSV	R	0	保留
12	USART6RST	R/W	0	USART6 控制器复位使能： 0：不复位 1：复位有效
11	UART1RST	R/W	0	UART1 控制器复位使能： 0：不复位 1：复位有效
10	WWDTRST	R/W	0	WWDT 控制器复位使能： 0：不复位 1：复位有效
9:0	RSV	R	0	保留

6.3.24 APB1 外围复位使能寄存器（RCM_APB1RSTR）

地址：0x08C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:21	RSV	R	0	保留
20	SPI0RST	R/W	0	SPI0 控制器复位使能： 0：不复位 1：复位有效
19	RSV	R	0	保留
18	I2C2RST	R/W	0	I2C2 控制器复位使能： 0：不复位 1：复位有效
17	I2C1RST	R/W	0	I2C1 控制器复位使能： 0：不复位 1：复位有效
16	I2C0RST	R/W	0	I2C0 控制器复位使能： 0：不复位 1：复位有效
15	RSV	R/W	0	保留
14	RSV	R	0	保留
13	DACRST	R/W	0	DAC 控制器复位使能： 0：不复位 1：复位有效
12	TIM13RST	R/W	0	TIM13EN 控制器复位使能： 0：不复位 1：复位有效
11	TIM12RST	R/W	0	TIM12 控制器复位使能： 0：不复位 1：复位有效
10	TIM11RST	R/W	0	TIM11EN 控制器复位使能： 0：不复位 1：复位有效

位	名称	属性	复位值	描述
9	TIM6RST	R/W	0	TIM6 控制器复位使能： 0：不复位 1：复位有效
8	TIM5RST	R/W	0	TIM5 控制器复位使能： 0：不复位 1：复位有效
7	TIM4RST	R/W	0	TIM4 控制器复位使能： 0：不复位 1：复位有效
6	TIM3RST	R/W	0	TIM3 控制器复位使能： 0：不复位 1：复位有效
5	TIM2RST	R/W	0	TIM2 控制器复位使能 0：不复位 1：复位有效
4	TIM1RST	R/W	0	TIM1 控制器复位使能： 0：不复位 1：复位有效
3	UART5RST	R/W	0	UART5 控制器复位使能： 0：不复位 1：复位有效
2	UART4RST	R/W	0	UART4 控制器复位使能： 0：不复位 1：复位有效
1	UART3RST	R/W	0	UART3 控制器复位使能： 0：不复位 1：复位有效
0	UART2RST	R/W	0	UART2 控制器复位使能： 0：不复位 1：复位有效

6.3.25 APB2 外围复位使能寄存器（RCM_APB2RSTR）

地址：0x090

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	R	0	保留
15	UART0RST	R/W	0	UART0 控制器复位使能： 0：不复位 1：复位有效
14	USART7RST	R/W	0	USART7 控制器复位使能： 0：不复位 1：复位有效
13	TIM10RST	R/W	0	TIM10 控制器复位使能： 0：不复位 1：复位有效

位	名称	属性	复位值	描述
12	TIM9RST	R/W	0	TIM9 控制器复位使能： 0：不复位 1：复位有效
11	TIM8RST	R/W	0	TIM8 控制器复位使能： 0：不复位 1：复位有效
10	TIM7RST	R/W	0	TIM7 控制器复位使能： 0：不复位 1：复位有效
9	TIM0RST	R/W	0	TIM0 控制器复位使能： 0：不复位 1：复位有效
8	SPI3RST	R/W	0	SPI3 控制器复位使能： 0：不复位 1：复位有效
7	SPI2RST	R/W	0	SPI2 控制器复位使能： 0：不复位 1：复位有效
6	SPI1RST	R/W	0	SPI1 控制器复位使能： 0：不复位 1：复位有效
5	ADC1RST	R/W	0	ADC1 控制器复位使能： 0：不复位 1：复位有效
4	ADC0RST	R/W	0	ADC0 控制器复位使能： 0：不复位 1：复位有效
3:2	RSV	R	0	保留
1	I2S1RST	R/W	0	I2S1 控制器复位使能： 0：不复位 1：复位有效
0	I2S0RST	R/W	0	I2S0 控制器复位使能： 0：不复位 1：复位有效

6.3.26 APB3 外围复位使能寄存器（RCM_APB3RSTR）

地址：0x094

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:2	RSV	R	0	保留
1	CAN1RST	R/W	0	CAN1 控制器复位使能： 0：不复位 1：复位有效
0	CAN0RST	R/W	0	CAN0 控制器复位使能： 0：不复位 1：复位有效

6.3.27 软件复位寄存器（RCM_SOFTRSTR）

地址：0x098

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:0	SOFTRST	R/W	1	软件复位寄存器： 写入 32'hA5A5_4321，会产生一次软件复位，复位 CPU 及 AHB/APB 总线上的所有 IP。并且 eFlash 地址重新映射（remap 为 1）。 读取时：0：系统即将进行软件复位 1：系统不进行软件复位 写入时：32'hA5A5_4321：产生一次软件复位 其他值：不产生软件复位 注：一般不使用

6.3.28 复位标识寄存器（RCM_RFR）

地址：0x0E0

复位值：0x0000 0003

位	名称	属性	复位值	描述
31:17	RSV	R	0	保留
16	RSTM	W	0	清除复位标志位： 0：不清除 1：清除复位标识位
15:11	RSV	R	0	保留
10	LOCKUPRSTF	R	0	LOCKUP 复位状态标志位： 0：未发生复位 1：发生复位
9	LVDRSTF	R	0	LVD 复位状态标志位： 0：未发生复位 1：发生复位
8	SYSSOFTRSTF	R	0	系统软件复位状态标志位： 0：未发生复位 1：发生复位
7	BORRSTF	R	0	BOR 复位状态标志位： 0：未发生复位 1：发生复位
6	XTHRSTF	R	0	XTH 时钟停止服务状态标志位： 0：未发生复位 1：发生复位
5	WWDTRSTF	R	0	窗口看门狗复位状态标志位： 0：未发生复位 1：发生复位
4	IWDTRSTF	R	0	独立看门狗复位状态标志位： 0：未发生复位 1：发生复位

位	名称	属性	复位值	描述
3	SOFTRSTF	R	0	M4 核软件复位状态标志位： 0：未发生复位 1：发生复位
2	PINRSTF	R	0	外部管脚 RESETN 复位状态标志位： 0：未发生复位 1：发生复位
1	PORRSTF	R	1	上电 POR01 复位状态标志位： 0：未发生复位 1：发生复位
0	RSV	R	1	保留

注：这个寄存器只能 POR 复位。

6.3.29 外部复位滤波控制寄存器（RCM_EXRSTFER）

地址：0x0E4

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	R	0	保留
0	EXT_FILTER_EN	R/W	0	外部复位滤波使能位： 1：外部复位滤波使能 0：外部复位滤波禁止

6.3.30 RCM 配置保护寄存器（RCM_RCMPR）

地址：0x0F0

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	RCMPR	R/W	0	RCM 寄存器写保护控制寄存器： 给此寄存器写 0xA5A5_5A5A，启动 RCM 寄存器的写使能；给此寄存器写其他值，关闭它们的写使能。对 RCM 的时钟复位配置寄存器配置完后，建议写入其他值，以关闭写使能，保护配置好的各个 RCM 寄存器值。 读取时：0：写使能关闭 1：写使能打开 写入时：32'hA5A5_5A5A：写使能打开 其他值：写使能关闭

7 通用 I/O（GPIO）

7.1 概述

GPIO 包含通用数据输入输出接口，这些管脚可以与其他功能管脚共享，这取决于芯片的配置。通过这些数据接口，可以配置任意数目的管脚作为中断信号输入。

7.2 主要特性

- 所有输入/输出引脚方向都可以通过软件进行配置
- 每个输入引脚可配置成边沿或电平方式触发中断
- 支持输入滤波

7.3 功能描述

7.3.1 GPIO 输入输出

产生输出信号，同步输入信号。

7.3.2 GPIO 中断产生

捕获输入信号，产生中断。当关闭滤波时，可以在没有时钟的情况下触发电平检测中断。

7.4 寄存器描述

GPIOA 寄存器基地址：0x4500_0000

GPIOB 寄存器基地址：0x4500_0400

GPIOC 寄存器基地址：0x4500_0800

GPIOD 寄存器基地址：0x4500_0C00

GPIOE 寄存器基地址：0x4500_1000

GPIOH 寄存器基地址：0x4500_1C00

寄存器列表如下：

表 7-1：GPIO 寄存器列表

偏移地址	名称	描述
0x00	GPIOx_MODE	功能模式寄存器

偏移地址	名称	描述
0x04	GPIO_SET	输出置位寄存器
0x08	GPIO_CLR	输出清零寄存器
0x0C	GPIO_ODATA	输出数据寄存器
0x10	GPIO_IDATA	输入数据寄存器
0x14	GPIO_IEN	中断使能寄存器
0x18	GPIO_IS	中断触发模式寄存器
0x1C	GPIO_IBE	中断边沿触发模式寄存器
0x20	GPIO_IEV	中断触发电平设置寄存器
0x24	GPIO_IC	中断清除寄存器
0x28	GPIO_RIS	原始中断状态寄存器
0x2C	GPIO_MIS	屏蔽后中断状态寄存器
0x30	GPIO_DBEN	滤波使能寄存器
0x34	GPIO_DBL	滤波长度寄存器
0x38	GPIO_LOCK	配置锁定寄存器
0x3C	GPIO_IM	输入类型寄存器
0x40	GPIO_PULL	上下拉寄存器
0x48	GPIO_SR	驱动速率寄存器
0x4C	GPIO_DS	驱动能力寄存器
0x50	GPIO_AFL	复用功能选择低位寄存器
0x54	GPIO_AFH	复用功能选择高位寄存器

以下各节详细介绍寄存器。

7.4.1 功能模式寄存器 (GPIOx_MODE)

偏移地址：0x00

GPIOA 的复位值是：0xABFF FFFF

GPIOB 的复位值是：0xFFFF FE8F

GPIOC/GPIOD/GPIOE/GPIOH 的复位值：0xFFFF FFFF

位	名称	属性	复位值	描述
31:0	MODE	R/W	0xFFFF FFFF (GPIOA,GPIOB 除外)	32 位寄存器，每 2 位对应 1 个 IO PAD；选择 GPIO 的功能： 00：通用输入模式 01：通用输出模式 10：复用功能模式 11：模拟模式

7.4.2 输出置位寄存器（GPIO_SET)

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	SET	W	0x0	16 位寄存器，每 1 位对应 1 个 IO： 0：无效操作 1：当 I/O 为输出时，对此位写 1 将 I/O 置位。

7.4.3 输出清零寄存器（GPIO_CLR)

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	CLR	W	0x0	16 位寄存器，每 1 位对应 1 个 IO： 0：无效操作 1：当 I/O 为输出时，对此位写 1 将 I/O 清零。

7.4.4 输出数据寄存器（GPIO_ODATA)

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	ODATA	R/W	0x0	16 位寄存器，每 1 位对应 1 个 IO。 当 GPIO 方向为输出有效，写操作会直接写至外部引脚；读操作可获得外部引脚输出值。

7.4.5 输入数据寄存器（GPIO_IDATA)

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留

位	名称	属性	复位值	描述
15:0	IDATA	R	0x0	16 位寄存器，每 1 位对应 1 个 IO 当 GPIO 方向为输入有效，读取可获得外部引脚实际值。

7.4.6 中断使能寄存器（GPIO_IEN)

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	IEN	R/W	0x0	16 位寄存器，每 1 位对应 1 个 IO： 0：禁止相应引脚中断 1：使能相应引脚中断

7.4.7 中断触发模式寄存器（GPIO_IS)

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	IS	R/W	0x0	16 位寄存器，每 1 位对应 1 个 IO： 0：边沿检测 1：电平检测

7.4.8 中断边沿触发设置寄存器（GPIO_IBE)

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	IBE	R/W	0x0	16 位寄存器，每 1 位对应 1 个 IO： 0：单边沿触发 1：双边沿触发

7.4.9 中断触发电平设置寄存器（GPIO_IEV)

偏移地址：0x20

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	IEV	R/W	0x0	16 位寄存器，每 1 位对应 1 个 IO： 0：下降沿/低电平触发 1：上升沿/高电平触发

7.4.10 中断清除寄存器（GPIO_IC)

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	IC	W	0x0	16 位寄存器，每 1 位对应 1 个 IO： 0：无效操作 1：写入 1 清除对应引脚中断

7.4.11 原始中断状态寄存器（GPIO_RIS)

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	RIS	R	0x0	16 位寄存器，每 1 位对应 1 个 IO： 0：对应引脚无中断挂起: 1：对应引脚有中断挂起

7.4.12 屏蔽后中断状态寄存器（GPIO_MIS)

偏移地址：0x2C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	MIS	R	0x0	16 位寄存器，每 1 位对应 1 个 IO： 0：对应引脚无中断输出到系统 1：对应引脚有中断输出到系统

7.4.13 滤波使能寄存器（GPIO_DBEN)

偏移地址：0x30

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	DBEN	R/W	0x0	16 位寄存器，每 1 位对应 1 个 IO： 0：不使能输入滤波 1：使能输入滤波 注：使用时，每次使能 1 个 IO 进行输入滤波，不能同时使能多个 IO 进行输入滤波。

7.4.14 滤波长度寄存器（GPIO_DBL)

偏移地址：0x34

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	DBL	R/W	0x0	32 位寄存器，滤波周期数。

7.4.15 配置锁定寄存器（GPIO_LOCK)

偏移地址：0x38

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	LOCK	R/W	0x0	16 位寄存器，每 1 位对应 1 个 IO；设为 1 即可锁定对应的 IO 的配置，直至下次复位。 锁定范围为 GPIO_MODE、GPIO_IM、GPIO_PULL、GPIO_SR、GPIO_DS、GPIO_AFH、GPIO_AFL 寄存器

7.4.16 输入类型寄存器（GPIO_IM)

偏移地址：0x3C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留

位	名称	属性	复位值	描述
15:0	IM	R/W	0x0	16 位寄存器，每 1 位对应 1 个 IO，选择 GPIO 输入类型： 0：CMOS 1：Schmitt trigger

7.4.17 上下拉寄存器（GPIO_PULL）

偏移地址：0x40

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	PE	R/W	0x0	16 位寄存器，每 1 位对应 1 个 IO，选择使能上下拉： 0：不使能 1：使能
15:0	PS	R/W	0x0	16 位寄存器，每 1 位对应 1 个 IO，选择拉动方向： 0：下拉 1：上拉

7.4.18 驱动速率寄存器（GPIO_SR）

偏移地址：0x48

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	SR	R/W	0x0	16 位寄存器，每 1 位对应 1 个 IO，选择 GPIO 驱动变化速率： 0：高速率 1：低速率

7.4.19 驱动能力寄存器（GPIO_DS）

偏移地址：0x4C

复位值：0x5555 5555

位	名称	属性	复位值	描述
31:0	DS	R/W	0x5555_5555	32 位寄存器，每 2 位对应 1 个 IO，选择 GPIO 输出驱动能力： 00：2mA 01：4mA 10：8mA 11：12mA

7.4.20 复用功能选择低位寄存器 (GPIO_AFL)

偏移地址：0x50

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	AFL	R/W	0x0	32 位寄存器，每 4 位对应 1 个 IO(0~7)，选择 GPIO 复用功能： 0000：复用功能 0 0001：复用功能 1 0010：复用功能 2 … 1111：复用功能 15 (对应管脚复用表的 AF0~AF15)

7.4.21 复用功能选择高位寄存器 (GPIO_AFH)

偏移地址：0x54

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	AFH	R/W	0x0	32 位寄存器，每 4 位对应 1 个 IO(8~15)，选择 GPIO 复用功能： 0000：复用功能 0 0001：复用功能 1 0010：复用功能 2 … 1111：复用功能 15 (对应管脚复用表的 AF0~AF15)

7.5 使用流程

7.5.1 输入

1. RCM 模块中使能 GPIOx 时钟。
2. 配置 GPIOx_MODE 寄存器，选择 GPIO 为通用输入功能。

3. 使用 GPIO_IDATA 来获取输入引脚电平。

7.5.2 输出

1. RCM 模块中使能 GPIOx 时钟。
2. 配置 GPIOx_MODE 寄存器，选择 GPIO 为通用输出功能。
3. 使用 GPIO_SET/GPIO_CLR 或 GPIO_ODATA 来设置输出电平。

7.5.3 中断触发模式

1. RCM 模块中使能 GPIOx 时钟。
2. 设置 GPIOx_MODE 为输入。
3. 使用 GPIO_IC 清除中断以避免异常。
4. 配置寄存器 GPIO_IS，选择是边沿/电平触发类型。
5. 在边沿触发方式下，配置寄存器 GPIO_IBE，确定是单边触发还是双边触发。
6. 在单边沿触发方式下，配置寄存器 GPIO_IEV，确定是哪种边沿触发类型。
7. 在电平触发方式下，配置寄存器 GPIO_IEV，确定是哪种电平触发类型。
8. 使用 GPIO_IC 清除中断。
9. 配置寄存器 GPIO_IEN 使能相应位中断。

8 中断控制器（NVIC）

ARM Cortex-M4 处理器和嵌套式矢量型中断控制器（NVIC）在处理（handler）模式下对所有异常进行优先级区分以及处理。当异常发生时，系统自动将当前处理器工作状态压栈，在执行完中断服务子程序（ISR）后自动将其出栈。

取向量是和当前工作态压栈并行进行的，从而提高了中断入口效率。处理器支持咬尾中断，可实现背靠背中断，大大削减了反复切换工作态所带来的开销。

所有的中断类型如下表所示：

表 8-1：Cotrex-M4 核的中断源端口描述

IRQ No.	外设中断	外设中断说明	向量地址
0	WWDT	窗口看门狗中断	0x0000_0040
1	LVD	LVD 中断	0x0000_0044
2	RTC_TAMPER	RTC 浸入和时间戳中断	0x0000_0048
3	-	保留	-
4	EFC	EFC 中断	0x0000_0050
5	RCM	时钟复位中断	0x0000_0054
6	EXTI0	EXTI[0]线中断	0x0000_0058
7	EXTI1	EXTI[1]线中断	0x0000_005C
8	EXTI2	EXTI[2]线中断	0x0000_0060
9	EXTI3	EXTI[3]线中断	0x0000_0064
10	EXTI4	EXTI[4]线中断	0x0000_0068
11	DMA0_CH0	DMA 控制器 0 通道 0 全局中断	0x0000_006C
12	DMA0_CH1	DMA 控制器 0 通道 1 全局中断	0x0000_0070
13	DMA0_CH2	DMA 控制器 0 通道 2 全局中断	0x0000_0074
14	DMA0_CH3	DMA 控制器 0 通道 3 全局中断	0x0000_0078
15	DMA0_CH4	DMA 控制器 0 通道 4 全局中断	0x0000_007C
16	DMA0_CH5	DMA 控制器 0 通道 5 全局中断	0x0000_0080
17	DMA0_CH6	DMA 控制器 0 通道 6 全局中断	0x0000_0084
18	DMA0_CH7	DMA 控制器 0 通道 7 全局中断	0x0000_0088
19	ADC0	ADC0 全局中断	0x0000_008C
20	ADC1	ADC1 全局中断	0x0000_0090
21	CAN0	CAN0 全局中断	0x0000_0094
22	CAN1	CAN1 全局中断	0x0000_0098
23	EXTI9~5	EXTI[9:5]线中断	0x0000_009C
24	TIM0_BRK_TIM8	TIM0 中止中断和 TIM8 全局中断	0x0000_00A0
25	TIM0_UP_TIM9	TIM0 更新中断和 TIM9 全局中断	0x0000_00A4
26	TIM0_TRG_COM_TIM10	TIM0 触发与通道换相中断和 TIM10 全局中断	0x0000_00A8
27	TIM0_CC	TIM0 捕获比较中断	0x0000_00AC
28	TIM1	TIM1 全局中断	0x0000_00B0

IRQ No.	外设中断	外设中断说明	向量地址
29	TIM2	TIM2 全局中断	0x0000_00B4
30	TIM3	TIM3 全局中断	0x0000_00B8
31	I2C0	I2C0 全局中断	0x0000_00BC
32	I2C1	I2C1 全局中断	0x0000_00C0
34~33	-	保留	-
35	SPI0	SPI0 全局中断	0x0000_00CC
36	SPI1	SPI1 全局中断	0x0000_00D0
37	UART0	UART0 全局中断	0x0000_00D4
38	UART1	UART1 全局中断	0x0000_00D8
39	UART2	UART2 全局中断	0x0000_00DC
40	EXTI15~10	EXTI[15:10]线中断	0x0000_00E0
41	RTC_ALARM	RTC 闹钟中断	0x0000_00E4
42	-	保留	-
43	TIM7_BRK_TIM11	TIM7 中止中断和 TIM11 全局中断	0x0000_00EC
44	TIM7_UP_TIM12	TIM7 更新中断和 TIM12 全局中断	0x0000_00F0
45	TIM7_TRG_COM_TIM13	TIM7 触发与通道换相中断和 TIM13 全局中断	0x0000_00F4
46	TIM7_CC	TIM7 捕获比较中断	0x0000_00F8
47	SDIO	SDIO 全局中断	0x0000_00FC
49~48	-	保留	-
50	TIM4	TIM4 全局中断	0x0000_0108
51	SPI2	SPI2 全局中断	0x0000_010C
52	UART3	UART3 全局中断	0x0000_0110
53	UART4	UART4 全局中断	0x0000_0114
54	TIM5	TIM5 全局中断	0x0000_0118
55	TIM6	TIM6 全局中断	0x0000_011C
56	DMA1_CH0	DMA 控制器 1 通道 0 全局中断	0x0000_0120
57	DMA1_CH1	DMA 控制器 1 通道 1 全局中断	0x0000_0124
58	DMA1_CH2	DMA 控制器 1 通道 2 全局中断	0x0000_0128
59	DMA1_CH3	DMA 控制器 1 通道 3 全局中断	0x0000_012C
60	DMA1_CH4	DMA 控制器 1 通道 4 全局中断	0x0000_0130
61	EMAC	EMAC 全局中断	0x0000_0134
62	SPI3	SPI3 全局中断	0x0000_0138
63	-	保留	-
64	TS	TS 温度传感器中断	0x0000_0140
65	OPA0	OPA0 中断	0x0000_0144
66	OPA1	OPA1 中断	0x0000_0148
67	DAC	DAC 控制器全局中断	0x0000_014C
68	DMA1_CH5	DMA 控制器 1 通道 5 全局中断	0x0000_0150
69	DMA1_CH6	DMA 控制器 1 通道 6 全局中断	0x0000_0154
70	DMA1_CH7	DMA 控制器 1 通道 7 全局中断	0x0000_0158

IRQ No.	外设中断	外设中断说明	向量地址
71	UART5	UART5 全局中断	0x0000_015C
72	I2C2	I2C2 全局中断	0x0000_0160
73	USB0 CONTROLLER	USB0 CONTROLLER 全局中断	0x0000_0164
75~74	-	保留	-
76	USART6	USART6 全局中断	0x0000_0170
77	USART7	USART7 全局中断	0x0000_0174
78	DCMI	DCMI 全局中断	0x0000_0178
79	AES	AES 全局中断	0x0000_017C
80	SHA	SHA 全局中断	0x0000_0180
81	FPU	FPU 全局中断	0x0000_0184
82	ACMP0	ACMP0 模拟比较器中断	0x0000_0188
83	ACMP1	ACMP1 模拟比较器中断	0x0000_018C
84	ACMP2	ACMP2 模拟比较器中断	0x0000_0190
85	I2S0	I2S0 全局中断	0x0000_0194
86	I2S1	I2S1 全局中断	0x0000_0198
89~87	-	保留	-
90	QSPI	QSPI 全局中断	0x0000_01A8
91	-	保留	-
92	-	保留	-
93	IWDT	IWDT 全局中断	0x0000_01B4
106~94	-	保留	-
107	LPUART	LPUART 全局中断	0x0000_01EC
108	-	保留	-
109	LPTIM0	LPTIM0 全局中断	0x0000_01F4
110	LPTIM1	LPTIM1 全局中断	0x0000_01F8
118~111	-	保留	-

9 系统配置控制器（SYSCFG）

寄存器基地址：0x40B0_2000

寄存器列表如下：

表 9-1：SYSCFG 寄存器列表

偏移地址	名称	描述
0x04	SYSCFG_EMACMR	EMAC 模式控制寄存器
0x08	SYSCFG_ADCETSR	ADC 外部触发选择寄存器
0x0C	SYSCFG_TIMCFGR	TIM 刹车控制寄存器
0x10	SYSCFG_EXTICR0	外部中断配置寄存器 0
0x14	SYSCFG_EXTICR1	外部中断配置寄存器 1
0x18	SYSCFG_EXTICR2	外部中断配置寄存器 2
0x1C	SYSCFG_EXTICR3	外部中断配置寄存器 3
0x20	SYSCFG_EXTIWR	外部中断唤醒配置寄存器
0x24	SYSCFG_MISCCR	MISC 控制寄存器
0x28	SYSCFG_SCRVR	Systick 计数参考值寄存器

9.1 寄存器描述

9.1.1 EMAC 模式控制寄存器（SYSCFG_EMACMR）

地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:3	RSV	R	0	保留
2:0	EMACMR	R/W	0	EMAC 模式控制寄存器（与外部 MAC PHY 连接）： 3'b000：MII 3'b001：RGMII 3'b010：保留 3'b011：保留 3'b100：RMII 3'b101：保留 3'b110：保留 3'b111：保留

9.1.2 ADC 外部触发选择寄存器（SYSCFG_ADCETSR）

地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:7	RSV	R	0	保留

位	名称	属性	复位值	描述
6:4	INJ_TRIG_SEL	R/W	0	ADC 注入模式下外部触发源选择： 000：EXTI15 001：EXTI9 010：EXTI5 011：LPTIM0_OUT 100：LPTIM1_OUT 101：保留 110：保留 111：保留
3	RSV	R	0	保留
2:0	RGL_TRIG_SEL	R/W	0	ADC 常规模式下外部触发源选择： 000：EXTI11 001：EXTI10 010：EXTI6 011：LPTIM0_OUT 100：LPTIM1_OUT 101：保留 110：保留 111：保留

9.1.3 TIM 刹车控制寄存器（SYSCFG_TIMCFGR）

地址：0x0C

复位值：0x0000 0003

位	名称	属性	复位值	描述
31:5	RSV	R	0	保留
4	TIMx_DEBUG_STOP	R/W	0	TIMx 调试控制信号： 0：无效 1：有效
3:2	RSV	R	0	保留
1	TIM7_BRKR	R/W	1	TIM7 刹车模式控制寄存器： 1：刹车时 OCx 和 OCxn 均为 0 0：刹车时 OCx 和 OCxn 为 OCxp 和 OCxnp
0	TIM0_BRKR	R/W	1	TIM0 刹车模式控制寄存器： 1：刹车时 OCx 和 OCxn 均为 0 0：刹车时 OCx 和 OCxn 为 OCxp 和 OCxnp

9.1.4 外部中断配置寄存器 0（SYSCFG_EXTICR0）

地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:28	RSV	R	0	保留
27	EXTI3_WK_FLAG	R	0	STOP 模式下，EXTI3 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
26	EXTI2_WK_FLAG	R	0	STOP 模式下，EXTI2 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
25	EXTI1_WK_FLAG	R	0	STOP 模式下，EXTI1 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
24	EXTI0_WK_FLAG	R	0	STOP 模式下，EXTI0 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
23	EXTI3_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI3 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
22	EXTI2_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI2 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
21	EXTI1_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI1 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
20	EXTI0_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI0 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
19	EXTI3_WKEN	R/W	0	STOP 模式下，EXTI3 唤醒使能位： 0：不使能 1：使能
18	EXTI2_WKEN	R/W	0	STOP 模式下，EXTI2 唤醒使能位： 0：不使能 1：使能
17	EXTI1_WKEN	R/W	0	STOP 模式下，EXTI1 唤醒使能位： 0：不使能 1：使能
16	EXTI0_WKEN	R/W	0	STOP 模式下，EXTI0 唤醒使能位： 0：不使能 1：使能
15:12	EXTI3	R/W	0	外部中断配置寄存器： 4'b0000：PA[3]引脚 4'b0001：PB[3]引脚 4'b0010：PC[3]引脚 4'b0011：PD[3]引脚 4'b0100：PE[3]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留

位	名称	属性	复位值	描述
11:8	EXTI2	R/W	0	外部中断配置寄存器： 4'b0000：PA[2]引脚 4'b0001：PB[2]引脚 4'b0010：PC[2]引脚 4'b0011：PD[2]引脚 4'b0100：PE[2]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留
7:4	EXTI1	R/W	0	外部中断配置寄存器： 4'b0000：PA[1]引脚 4'b0001：PB[1]引脚 4'b0010：PC[1]引脚 4'b0011：PD[1]引脚 4'b0100：PE[1]引脚 4'b0101：保留 4'b0110：保留 4'b0111：PH[1]引脚 4'b1000：保留
3:0	EXTI0	R/W	0	外部中断配置寄存器： 4'b0000：PA[0]引脚 4'b0001：PB[0]引脚 4'b0010：PC[0]引脚 4'b0011：PD[0]引脚 4'b0100：PE[0]引脚 4'b0101：保留 4'b0110：保留 4'b0111：PH[0]引脚 4'b1000：保留

9.1.5 外部中断配置寄存器 1（SYSCFG_EXTICR1）

地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:28	RSV	R	0	保留
27	EXTI7_WK_FLAG	R	0	STOP 模式下，EXTI7 唤醒源状态： 0：不是此位唤醒 1：此位唤醒（
26	EXTI6_WK_FLAG	R	0	STOP 模式下，EXTI6 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
25	EXTI5_WK_FLAG	R	0	STOP 模式下，EXTI5 唤醒源状态： 0：不是此位唤醒 1：此位唤醒

位	名称	属性	复位值	描述
24	EXTI4_WK_FLAG	R	0	STOP 模式下，EXTI4 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
23	EXTI7_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI7 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
22	EXTI6_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI6 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
21	EXTI5_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI5 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
20	EXTI4_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI4 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
19	EXTI7_WKEN	R/W	0	STOP 模式下，EXTI7 唤醒使能位： 0：不使能 1：使能
18	EXTI6_WKEN	R/W	0	STOP 模式下，EXTI6 唤醒使能位： 0：不使能 1：使能
17	EXTI5_WKEN	R/W	0	STOP 模式下，EXTI5 唤醒使能位： 0：不使能 1：使能
16	EXTI4_WKEN	R/W	0	STOP 模式下，EXTI4 唤醒使能位： 0：不使能 1：使能
15:12	EXTI7	R/W	0	外部中断配置寄存器： 4'b0000：PA[7]引脚 4'b0001：PB[7]引脚 4'b0010：PC[7]引脚 4'b0011：PD[7]引脚 4'b0100：PE[7]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留
11:8	EXTI6	R/W	0	外部中断配置寄存器： 4'b0000：PA[6]引脚 4'b0001：PB[6]引脚 4'b0010：PC[6]引脚 4'b0011：PD[6]引脚 4'b0100：PE[6]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留

位	名称	属性	复位值	描述
7:4	EXTI5	R/W	0	外部中断配置寄存器： 4'b0000：PA[5]引脚 4'b0001：PB[5]引脚 4'b0010：PC[5]引脚 4'b0011：PD[5]引脚 4'b0100：PE[5]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留
3:0	EXTI4	R/W	0	外部中断配置寄存器： 4'b0000：PA[4]引脚 4'b0001：PB[4]引脚 4'b0010：PC[4]引脚 4'b0011：PD[4]引脚 4'b0100：PE[4]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留

9.1.6 外部中断配置寄存器 2（SYSCFG_EXTICR2）

地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:28	RSV	R	-	保留
27	EXTI11_WK_FLAG	R	0	STOP 模式下，EXTI11 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
26	EXTI10_WK_FLAG	R	0	STOP 模式下，EXTI10 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
25	EXTI9_WK_FLAG	R	0	STOP 模式下，EXTI9 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
24	EXTI8_WK_FLAG	R	0	STOP 模式下，EXTI8 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
23	EXTI11_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI11 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
22	EXTI10_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI10 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒

位	名称	属性	复位值	描述
21	EXTI9_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI9 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
20	EXTI8_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI8 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
19	EXTI11_WKEN	R/W	0	STOP 模式下，EXTI11 唤醒使能位： 0：不使能 1：使能
18	EXTI10_WKEN	R/W	0	STOP 模式下，EXTI10 唤醒使能位： 0：不使能 1：使能
17	EXTI9_WKEN	R/W	0	STOP 模式下，EXTI9 唤醒使能位： 0：不使能 1：使能
16	EXTI8_WKEN	R/W	0	STOP 模式下，EXTI8 唤醒使能位： 0：不使能 1：使能
15:12	EXTI11	R/W	0	外部中断配置寄存器： 4'b0000：PA[11]引脚 4'b0001：PB[11]引脚 4'b0010：PC[11]引脚 4'b0011：PD[11]引脚 4'b0100：PE[11]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留
11:8	EXTI10	R/W	0	外部中断配置寄存器： 4'b0000：PA[10]引脚 4'b0001：PB[10]引脚 4'b0010：PC[10]引脚 4'b0011：PD[10]引脚 4'b0100：PE[10]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留
7:4	EXTI9	R/W	0	外部中断配置寄存器： 4'b0000：PA[9]引脚 4'b0001：PB[9]引脚 4'b0010：PC[9]引脚 4'b0011：PD[9]引脚 4'b0100：PE[9]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留

位	名称	属性	复位值	描述
3:0	EXTI8	R/W	0	外部中断配置寄存器： 4'b0000：PA[8]引脚 4'b0001：PB[8]引脚 4'b0010：PC[8]引脚 4'b0011：PD[8]引脚 4'b0100：PE[8]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留

9.1.7 外部中断配置寄存器 3（SYSCFG_EXTICR3）

地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:28	RSV	R	0	保留
27	EXTI15_WK_FLAG	R	0	STOP 模式下，EXTI15 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
26	EXTI14_WK_FLAG	R	0	STOP 模式下，EXTI14 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
25	EXTI13_WK_FLAG	R	0	STOP 模式下，EXTI13 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
24	EXTI12_WK_FLAG	R	0	STOP 模式下，EXTI12 唤醒源状态： 0：不是此位唤醒 1：此位唤醒
23	EXTI15_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI15 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
22	EXTI14_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI14 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
21	EXTI13_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI13 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒
20	EXTI12_WK_EDGE_SEL	R/W	0	STOP 模式下，EXTI12 唤醒边沿选择： 0：上升沿唤醒 1：下降沿唤醒

位	名称	属性	复位值	描述
19	EXTI15_WKEN	R/W	0	STOP 模式下，EXTI15 唤醒使能位： 0：不使能 1：使能
18	EXTI14_WKEN	R/W	0	STOP 模式下，EXTI14 唤醒使能位： 0：不使能 1：使能
17	EXTI13_WKEN	R/W	0	STOP 模式下，EXTI13 唤醒使能位： 0：不使能 1：使能
16	EXTI12_WKEN	R/W	0	STOP 模式下，EXTI12 唤醒使能位： 0：不使能 1：使能
15:12	EXTI15	R/W	0	外部中断配置寄存器： 4'b0000：PA[15]引脚 4'b0001：PB[15]引脚 4'b0010：PC[15]引脚 4'b0011：PD[15]引脚 4'b0100：PE[15]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留
11:8	EXTI14	R/W	0	外部中断配置寄存器： 4'b0000：PA[14]引脚 4'b0001：PB[14]引脚 4'b0010：PC[14]引脚 4'b0011：PD[14]引脚 4'b0100：PE[14]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留
7:4	EXTI13	R/W	0	外部中断配置寄存器： 4'b0000：PA[13]引脚 4'b0001：PB[13]引脚 4'b0010：PC[13]引脚 4'b0011：PD[13]引脚 4'b0100：PE[13]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留

位	名称	属性	复位值	描述
3:0	EXTI12	R/W	0	外部中断配置寄存器： 4'b0000：PA[12]引脚 4'b0001：PB[12]引脚 4'b0010：PC[12]引脚 4'b0011：PD[12]引脚 4'b0100：PE[12]引脚 4'b0101：保留 4'b0110：保留 4'b0111：保留 4'b1000：保留

9.1.8 外部中断唤醒配置寄存器（SYSCFG_EXTIWR）

地址：0x20

复位值：0x0002 0000

位	名称	属性	复位值	描述
31:18	RSV	R	0	保留
17	INT_SEL	R/W	1	中断响应方式： 0：保留 1：多组 GPIO 里选择一个当作中断 注：此位不用修改，必须为 1
16	IESEL	R/W	0	端口中断模式选择寄存器： 1：STOP 模式 0：RUN 模式 注： ● 当系统处于 RUN 模式下时，系统时钟不会被关闭，可以将 IESEL 设置为 0，此时触发端口中断的外部信号源将经过系统时钟同步之后产生中断信号，可以滤除外部信号源的毛刺。 ● 当系统即将进入 STOP 模式下时，系统时钟将被关闭，可以将 IESEL 设置为 1，此时触发端口中断的外部信号源直接产生中断信号，不能滤除外部信号源的毛刺。
15	EXTI15_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI15(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作。
14	EXTI14_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI14(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作

位	名称	属性	复位值	描述
13	EXTI13_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI13(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作
12	EXTI12_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI12(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作
11	EXTI11_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI11(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作
10	EXTI10_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI10(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作
9	EXTI9_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI9(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作
8	EXTI8_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI8(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作
7	EXTI7_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI7(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作
6	EXTI6_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI6(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作
5	EXTI5_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI5(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作

位	名称	属性	复位值	描述
4	EXTI4_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI4(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作
3	EXTI3_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI3(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作
2	EXTI2_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI2(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作
1	EXTI1_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI1(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作
0	EXTI0_CLR	R/W	0	当寄存器中的 IESEL 位为 1，且唤醒源为 EXTI0(INT_SEL=1)时，在系统唤醒后，此位写 1，清除 GPIO 唤醒中断。 1：系统唤醒后，清除 GPIO 的唤醒中断 0：无操作

9.1.9 MISC 控制寄存器（SYSCFG_MISCCR）

地址：0x24

复位值：0x0000 0030

位	名称	属性	复位值	描述
31:6	RSV	R	0	保留
5	EFC_VOL_PD_EN	R/W	1	EFC power down ready 有效信号（standby mode）： 0：无效 1：有效 注：一般不修改
4	EFC_CLK_PD_EN	R/W	1	EFC clock stop ready 有效信号（stop mode）： 0：无效 1：有效 注：一般不修改
3	RSV	R	0	保留
2	LVD_INT_EN	R/W	0	LVD 中断有效控制位： 0：无效 1：LVD 中断有效
1	RSV	R		保留

位	名称	属性	复位值	描述
0	NMIEN	R/W	0	NMI 中断控制寄存器： 0：NMI 中断无效 1：NMI 中断有效（LVD 中断）

9.1.10 SysTick 计数参考值寄存器（SYSCFG_SCRVR）

地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:24	RSV	R	0	保留
23:0	STCALIB	R/W	0	SysTick 计数参考值。

10 循环冗余校验计算单元（CRC）

10.1 概述

CRC 控制器集成了 CRC32 和 CRC16 的功能，可以以多种多项式进行循环冗余计算。

10.2 主要特性

- 支持以下多项式：

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

  $x^{16} + x^{15} + x^2 + 1$

  $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1$
- 支持输入字节、半字、字型数据
- 支持输入数据按字、半字、字节倒序、初始值倒序、结果值倒序

10.3 功能描述

10.3.1 CRC 运算

可以进行 CRC 计算，兼容多种格式。

10.3.2 常见 CRC 格式

表 10-1：常见 CRC 格式

名称	多项式	初始值	输入数据按字节倒序	输出数据倒序	结果异或值
CRC-16 / CCITT	1021	0	Y	Y	0
CRC-16 / CCITT-FALSE	1021	FFFF	N	N	0
CRC-16 / X25	1021	FFFF	Y	Y	FFFF
CRC-16 / XMODEM	1021	0	N	N	0
CRC-16 / IBM	8005	0	Y	Y	0
CRC-16 / MAXIM	8005	0	Y	Y	FFFF
CRC-16 / USB	8005	FFFF	Y	Y	FFFF
CRC-16 / MODBUS	8005	FFFF	Y	Y	0
CRC-32	04C11DB7	FFFFFFFF	Y	Y	FFFFFFFF
CRC-32 / MPEG-2	04C11DB7	FFFFFFFF	N	N	0

注：本模块未包含对结果进行异或计算的功能，如有需要请在取出结果后再另外计算。

10.4 寄存器描述

寄存器基地址：0x40A0_0000

寄存器列表如下：

表 10-2：CRC 寄存器列表

偏移地址	名称	描述
0x00	CRC_DATA	数据寄存器
0x04	CRC_CFG	设置寄存器
0x08	CRC_INIT	初始值寄存器

10.4.1 数据寄存器（CRC_DATA)

偏移地址：0x00

复位值：0x0000_0000

位	名称	属性	复位值	描述
31:0	DATA	R/W	0x0	写入数据进行计算，读出数据获得结果。

10.4.2 设置寄存器（CRC_CFG)

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
30:9	RSV	-	-	保留
8	INIT_REV	R/W	0x0	初始值格式： 0：初始值不倒序 1：初始值倒序
7	DOUT_REV	R/W	0x0	输出数据格式： 0：输出数据不倒序 1：输出数据倒序
6	RSV	-	-	保留
5	DIN_REV	R/W	0x0	输入数据格式： 0：输入数据不倒序 1：输入数据倒序
4:3	WIDTH_DIN	R/W	0x0	选择输入数据宽度： 00：8bit 01：16bit 10/11：32bit

位	名称	属性	复位值	描述
2:1	POL	R/W	0x0	选择 CRC 运算多项式： 00：选择 CRC16-1021 $(x^{16} + x^{12} + x^5 + 1)$； 01：选择 CRC16-8005 $(x^{16} + x^{15} + x^2 + 1)$； 10/11：选择 CRC32-04C11DB7 $(x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x^1 + 1)$
0	RESET	W	0x0	对此位写 1，会清除计算结果并载入初始值，此位会自动清除

10.4.3 初始值寄存器（CRC_INIT)

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	INIT	R/W	0x0	设定初始值。

10.5 使用流程

1. 配置 CRC_INIT 寄存器，确定初始值。
2. 配置 CRC_CFG 寄存器，选择 CRC 多项式、数据宽度、数据倒序情况，并载入初始值。
3. 向 CRC_DATA 寄存器写入数据，可连续写入数据。
4. 从 CRC_DATA 寄存器取出 CRC 运算结果。

11 DMA 控制器（DMA）

11.1 概述

DMA 可以协助 CPU 进行数据搬运的工作，减轻 CPU 的工作负担并提升系统效率。

11.2 主要特性

- 可以控制多个模块之间的数据传输
- 支持 Memory to Memory 模式、Memory to Peripheral 模式、Peripheral to Memory 模式、Peripheral to Peripheral 模式
- 内部含有 8 个 DMA 通道
- 数据传输的位宽可设、传输的 Block 长度可设
- 每个通道含有大小为 4x32 的 FIFO
- Block 最大长度可设为 4095
- 支持源地址不变传输、递增传输
- 支持目的地址的不变传输、递增传输

11.3 功能描述

11.3.1 流控制

DMA 支持由 DMA 本身作为流控制器的传输。对于传输源和目标，支持以下几种模式：

- 从储存器到储存器
- 从储存器到外设
- 从外设到储存器
- 从外设到外设

11.3.2 握手信号

当传输源或目标被设定为外设时，需要指定该外设使用的握手信号编号。当握手信号有效时，DMA 才会发起传输。

除了由外设硬件产生握手信号外，也可以用软件写入 DMA 寄存器的方式来产生握手信号。

DMA 握手信号分配如下：

表 11-1：DMA0 握手信号表

	握手信号0	握手信号1	握手信号2	握手信号3	握手信号4	握手信号5	握手信号6	握手信号7	握手信号8	握手信号9	握手信号10	握手信号11	握手信号12	握手信号13
外设请求	SPI2_RX	TIM1_UP	SPI2_RX	SPI1_RX	SPI1_TX	SPI2_TX	I2C0_TX	SPI2_TX	QSPI_RX	QSPI_TX	TIM11_CH4	TIM11_UP	TIM11_TRIG	TIM11_CH1
	I2C0_RX	TIM1_CH3	TIM6_UP	TIM3_CH2	TIM6_UP	I2C0_RX	TIM3_UP	I2C0_RX	TIM11_CH2	TIM11_CH3	TIM12_CH3	TIM12_CH4	TIM12_UP	TIM12_TRIG
	TIM3_CH1	UART2_RX	I2S1_RX	I2S0_RX	I2S0_TX	I2S1_TX	TIM1_CH2	TIM3_CH3	TIM12_CH1	TIM12_CH2	TIM13_CH2	TIM13_CH3	TIM13_CH4	TIM13_UP
	I2S1_RX	TIM4_CH4	I2C2_RX	UART2_TX	I2C2_TX	TIM1_CH1	TIM1_CH4	TIM1_CH4	TIM13_TRIG	TIM13_CH1	USART6_RX	USART6_TX		
	UART4_RX	TIM4_TRIG	UART3_RX	TIM4_TRIG	UART3_TX	UART1_RX	UART1_TX	TIM1_UP	TIM3_CH4	TIM3_TRIG	TIM1_TRIG			
	TIM4_CH3	TIM5_UP	TIM2_CH4	TIM4_CH4	TIM2_TRIG	TIM2_CH2	TIM4_UP	UART4_TX						
	TIM4_UP		TIM2_UP	I2C1_RX	TIM2_CH1	DAC0	DAC1	TIM2_CH3						
			TIM4_CH1		TIM4_CH2									
			I2C1_RX		UART2_TX									

表 11-2：DMA1 握手信号表

	握手信号0	握手信号1	握手信号2	握手信号3	握手信号4	握手信号5	握手信号6	握手信号7	握手信号8	握手信号9	握手信号10	握手信号11	握手信号12	握手信号13
外设请求	ADC0	DCMI	TIM7_CH1	ADC1	ADC0	QSPI_TX	TIM0_CH1	DCMI	SPI3_RX	SPI3_TX	TIM8_CH1	TIM8_CH2	CORDIC_IN	CORDIC_OUT
	SPI0_RX	UART5_RX	TIM7_CH2	SPI0_TX	TIM0_CH4	AES_OUT	TIM0_CH2	SHA_IN	TIM8_UP	TIM8_TRIG	TIM9_TRIG	TIM9_CH1	TIM8_CH3	TIM8_CH4
	TIM0_TRIG	TIM0_CH1	TIM7_CH3	TIM0_CH1	TIM0_TRIG	SPI0_TX	TIM0_CH3	UART0_TX	TIM9_CH4	TIM9_UP	TIM10_UP	TIM10_TRIG	TIM9_CH2	TIM9_CH3
		TIM7_UP	ADC1	TIM0_CH1	TIM0_COM	UART0_RX	QSPI_RX	UART5_TX	TIM10_CH3	TIM10_CH4	USART7_RX	USART7_TX	TIM10_CH1	TIM10_CH2
			SPI0_RX	TIM7_CH2	TIM7_CH3	TIM0_UP	UART5_TX	TIM7_CH4						
			UART0_RX					TIM7_TRIG						
			UART5_RX					TIM7_COM						
			TIM0_CH2											

注：

- 注意同一个握手信号每次只能从多个外设中选中一个来响应，不能同时选中多个外设来触发 DMA。
- 握手信号（外设）和 DMA 的 8 个通道，可根据应用自由配置。
- 握手信号是固定的，只能从表里相应的外设设置好，对应上。
- DMA 的 8 个通道，可以软件自由配置，适应不同的外设应用。

11.4 寄存器描述

DMA0 寄存器基地址：0x4070_0000

DMA1 寄存器基地址：0x4080_0000

寄存器列表如下：

表 11-3：DMA 寄存器列表

偏移地址	名称	说明
0x00	DMA_SAR0	通道 0 源地址寄存器
0x08	DMA_DAR0	通道 0 目的地址寄存器
0x18	DMA_CTL0	通道 0 控制寄存器
0x40	DMA_CFG0	通道 0 设置寄存器
0x58	DMA_SAR1	通道 1 源地址寄存器
0x60	DMA_DAR1	通道 1 目的地址寄存器
0x70	DMA_CTL1	通道 1 控制寄存器
0x98	DMA_CFG1	通道 1 设置寄存器
0xB0	DMA_SAR2	通道 2 源地址寄存器
0xB8	DMA_DAR2	通道 2 目的地址寄存器
0xC8	DMA_CTL2	通道 2 控制寄存器
0xF0	DMA_CFG2	通道 2 设置寄存器
0x108	DMA_SAR3	通道 3 源地址寄存器

偏移地址	名称	说明
0x110	DMA_DAR3	通道 3 目的地址寄存器
0x120	DMA_CTL3	通道 3 控制寄存器
0x148	DMA_CFG3	通道 3 设置寄存器
0x160	DMA_SAR4	通道 4 源地址寄存器
0x168	DMA_DAR4	通道 4 目的地址寄存器
0x178	DMA_CTL4	通道 4 控制寄存器
0x1A0	DMA_CFG4	通道 4 设置寄存器
0x1B8	DMA_SAR5	通道 5 源地址寄存器
0x1C0	DMA_DAR5	通道 5 目的地址寄存器
0x1D0	DMA_CTL5	通道 5 控制寄存器
0x1F8	DMA_CFG5	通道 5 设置寄存器
0x210	DMA_SAR6	通道 6 源地址寄存器
0x218	DMA_DAR6	通道 6 目的地址寄存器
0x228	DMA_CTL6	通道 6 控制寄存器
0x250	DMA_CFG6	通道 6 设置寄存器
0x268	DMA_SAR7	通道 7 源地址寄存器
0x270	DMA_DAR7	通道 7 目的地址寄存器
0x280	DMA_CTL7	通道 7 控制寄存器
0x2A8	DMA_CFG7	通道 7 设置寄存器
0x2C0	DMA_RAWTFR	原始传输中断寄存器
0x2C8	DMA_RAWBLOCK	原始 Block 传输中断寄存器
0x2D0	DMA_RAWSRCTRAN	原始源传输中断寄存器
0x2D8	DMA_RAWDSTTRAN	原始目标传输中断寄存器
0x2E0	DMA_RAWERR	原始错误中断寄存器
0x2E8	DMA_STATUSTFR	传输中断状态寄存器
0x2F0	DMA_STATUSBLOCK	Block 传输中断状态寄存器
0x2F8	DMA_STATUSSRCTRAN	源传输中断状态寄存器
0x300	DMA_STATUSDSTTRAN	目标传输中断状态寄存器
0x308	DMA_STATUSERR	错误中断状态寄存器
0x310	DMA_MASKTFR	传输中断屏蔽寄存器
0x318	DMA_MASKBLOCK	Block 传输中断屏蔽寄存器
0x320	DMA_MASKSRCTRAN	源传输中断屏蔽寄存器
0x328	DMA_MASKDSTTRAN	目标传输中断屏蔽寄存器
0x330	DMA_MASKERR	错误中断屏蔽寄存器
0x338	DMA_CLEARTFR	传输中断清除寄存器
0x340	DMA_CLEARBLOCK	Block 传输中断清除寄存器
0x348	DMA_CLEARSRCTRAN	源传输中断清除寄存器
0x350	DMA_CLEARDSTTRAN	目标中断清除寄存器
0x358	DMA_CLEARERR	错误中断清除寄存器
0x360	DMA_STATUSINT	中断状态寄存器
0x368	DMA_REQSRCREG	源传输 Req 信号软件握手寄存器
0x370	DMA_REQDSTREG	源传输 Req 信号软件握手寄存器

偏移地址	名称	说明
0x378	DMA_SGLREQSRCREG	源传输 Single 信号软件握手寄存器
0x380	DMA_SGLREQDSTREG	目标传输 Single 信号软件握手寄存器
0x388	DMA_LSTSRCREG	源传输 Last 信号软件握手寄存器
0x390	DMA_LSTDSTREG	目标传输 Last 信号软件握手寄存器
0x398	DMA_CFGREG	DMA 模块使能寄存器
0x3A0	DMA_CHENREG	通道使能寄存器

11.4.1 源地址寄存器（DMA_SARx）

偏移地址：0x00/0x58/0xB0/0x108/0x160/0x1B8/0x210/0x268，x=0~7

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	SAR	R/W	0x0	DMA 传输源地址，会自动更新

注：本寄存器必须在通道禁止时（DMA_CHENREG 的 CH_EN=0）进行设定。

11.4.2 目的地址寄存器（DMA_DARx）

偏移地址：0x08/0x60/0xB8/0x110/0x168/0x1C0/0x218/0x270，x=0~7

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	DAR	R/W	0x0	DMA 传输目的地址，会自动更新

注：本寄存器必须在通道禁止时（DMA_CHENREG 的 CH_EN=0）进行设定。

11.4.3 控制寄存器（DMA_CTLx）

偏移地址：0x18/0x70/0xC8/0x120/0x178/0x1D0/0x228/0x280，x=0~7

复位值：0x0030 4801

位	名称	属性	复位值	描述
31:22	RSV	-	-	保留
21:20	TT_FC	R/W	0x3	传输类型与流控制： 0x0：储存器到储存器 0x1：储存器到外设 0x2：外设到储存器 0x3：外设到外设
19:17	RSV	-	-	保留

位	名称	属性	复位值	描述
16:14	SRC_MSIZE	R/W	0x1	源传输 Burst 长度，每次源外设 Req 握手信号可用时读取数据个数： 0x0：1 0x1：4 0x2：8 0x3：16 其他：保留
13:11	DST_MSIZE	R/W	0x1	目标传输 Burst 长度，每次目标外设 Req 握手信号可用时写入数据个数： 0x0：1 0x1：4 0x2：8 0x3：16 其他：保留
10:9	SINC	R/W	0x0	源地址递增： 0x0：递增 0x1：递减 0x2/0x3：不变
8:7	DINC	R/W	0x0	目标地址递增： 0x0：递增 0x1：递减 0x2/0x3：不变
6	RSV	-	-	保留
5:4	SRC_TR_WIDTH	R/W	0x0	源传输数据宽度： 0x0：8bits 0x1：16bits 0x2：32bits
3	RSV	-	-	保留
2:1	DST_TR_WIDTH	R/W	0x0	目标传输数据宽度： 0x0：8bits 0x1：16bits 0x2：32bits
0	INT_EN	R/W	0x1	中断使能： 0x0：不使能 0x1：使能

注：本寄存器必须在通道禁止时（DMA_CHENREG 的 CH_EN=0）进行设定。

11.4.4 控制寄存器（DMA_CTLHx）

偏移地址：0x1C/0x74/0xCC/0x124/0x17C/0x1D4/0x22C/0x284，x=0~7

复位值：0x0000 0002

位	名称	属性	复位值	描述
31:12	RSV	-	-	保留

位	名称	属性	复位值	描述
11:0	BLOCK_TS	R/W	0x2	Block 传输源数据个数，单位是源传输的宽度，总的传输量由源传输决定，目的传输的次数根据源传输和目的传输的位宽自动变化。一旦传输开始，回读值都是已从源读取到的数据的总个数。

注：本寄存器必须在通道禁止时（DMA_CHENREG 的 CH_EN=0）进行设定。

11.4.5 设置寄存器（DMA_CFGx）

偏移地址：0x40/0x98/0xF0/0x148/0x1A0/0x1F8/0x250/0x2A8，x=0~7

复位值：0x0000_0e00 + 0x20 * x（x=0~7）

位	名称	属性	复位值	描述
31	RELOAD_DST	R/W	0x0	自动重启目标传输： 0x0：关闭 0x1：开启：当一次 Block 传输结束时，将 DARx 重设为初始值，并开始一次新的 Block 传输
30	RELOAD_SRC	R/W	0x0	自动重启源传输： 0x0：关闭 0x1：开启：当一次 Block 传输结束时，将 SARx 重设为初始值，并开始一次新的 Block 传输
29:20	RSV	-	-	保留
19	SRC_HS_POL	R/W	0x0	请使用 0x0：握手信号高有效
18	DST_HS_POL	R/W	0x0	请使用 0x0：握手信号高有效
17:12	RSV	-	-	保留
11	HS_SEL_SRC	R/W	0x1	源传输握手信号选择： 0x0：硬件握手 0x1：软件握手
10	HS_SEL_DST	R/W	0x1	目标传输握手信号选择： 0x0：硬件握手 0x1：软件握手
9	FIFO_EMPTY	R	0x1	指示该通道的 FIFO 是否为空： 0x0：FIFO 非空 0x1：FIFO 空
8	CH_SUSP	R/W	0x0	暂停该通道的传输： 0x0：正常传输 0x1：暂停传输
7:5	CH_PRIOR	R/W	各通道不同	指定通道优先级，0 为最低
4:0	RSV	-	-	保留

注：本寄存器必须在通道禁止时（DMA_CHENREG 的 CH_EN=0）进行设定。

11.4.6 设置寄存器（DMA_CFGHx）

偏移地址：0x44/0x9C/0xF4/0x14C/0x1A4/0x1FC/0x254/0x2AC，x=0~7

复位值：0x0000 0004

位	名称	属性	复位值	描述
31:15	RSV	-	-	保留
14:11	DEST_PER	R/W	0x0	目标外设握手信号编号： 在 TT_FC 中指定目标为储存器时此项被忽略
10:7	SRC_PER	R/W	0x0	源外设握手信号编号： 在 TT_FC 中指定源为储存器时此项被忽略
6:5	RSV	-	-	保留
4:2	PROTCTL	R/W	0x1	驱动 HPROT[3:1]
1	FIFO_MODE	R/W	0x0	指定需要多少可用数据/空间起一次 Burst 传输： 0x0：只要有一个可用数据/空间就会发起传输 0x1：可用数据大于或等于一半 FIFO 深度时发起对目标的传输，空间大于或等于一半 FIFO 深度时发起源传输；在一次 Burst 或 Block 传输的末尾时例外
0	FCMODE	R/W	0x0	流控模式/数据预读： 0x0：开启预读，源数据传输可用时读取 0x1：关闭预读，目标传输完成前不进行源数据传输

注：本寄存器必须在通道禁止时（DMA_CHENREG 的 CH_EN=0）进行设定。

11.4.7 原始传输中断寄存器（DMA_RAWTFR）

偏移地址：0x2C0

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	RAWTFR	R/W	0x0	bit x 为通道 x 的原始 TFR 中断状态。 该中断会在该通道的完成所有传输时触发。 注：正常使用时不建议直接写入这个寄存器。

11.4.8 原始 Block 传输中断寄存器（DMA_RAWBLOCK）

偏移地址：0x2C8

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	RAWBLOCK	R/W	0x0	bit x 为通道 x 的原始 BLOCK 中断状态。 该中断会在该通道完成一个 Block 传输时触发。 注：正常使用时不建议直接写入这个寄存器。

11.4.9 原始源传输中断寄存器（DMA_RAWSRCTRAN）

偏移地址：0x2D0

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	RAWSRCTRAN	R/W	0x0	bit x 为通道 x 的原始 SRCTRAN 中断状态 该中断会在该通道完成一个响应源外设握手信号的 Burst/Single 传输时触发。 注：正常使用时不建议直接写入这个寄存器

11.4.10 原始目标传输中断寄存器（DMA_RAWDSTTRAN）

偏移地址：0x2D8

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	RAWDSTTRAN	R/W	0x0	bit x 为通道 x 的原始 DSTTRAN 中断状态。 该中断会在该通道完成一个响应目标外设握手信号的 Burst/Single 传输时触发。 注：正常使用时不建议直接写入这个寄存器

11.4.11 原始错误中断寄存器（DMA_RAWERR）

偏移地址：0x2E0

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	RAWERR	R/W	0x0	bit x 为通道 x 的原始 Err 中断状态。 该中断会在该通道在传输时收到 HRSP 上的 ERROR 响应时触发，并会导致传输被取消和该通道被关闭。 注：正常使用时不建议直接写入这个寄存器

11.4.12 传输中断状态寄存器（DMA_STATUSTFR）

偏移地址：0x2E8

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	STATUSTFR	R	0x0	bit x 为通道 x 的 TFR 中断输出状态，该中断会在该通道完成所有传输时触发，如果该中断被屏蔽则该寄存器不会被置 1。

注：禁止对本寄存器进行写操作。

11.4.13 Block 传输中断状态寄存器（DMA_STATUSBLOCK）

偏移地址：0x2F0

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	STATUSBLOCK	R	0x0	bit x 为通道 x 的 Block 中断输出状态。 该中断会在该通道完成一个 Block 传输时触发 如果该中断被屏蔽则该寄存器不会被置 1

注：禁止对本寄存器进行写操作。

11.4.14 源传输中断状态寄存器（DMA_STATUSSRCTRAN）

偏移地址：0x2F8

复位值：0x00000000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	STATUSSRCTRAN	R	0x0	bit x 为通道 x 的 SRCTRAN 中断输出状态。 该中断会在该通道完成一个响应源外设握手信号的 Burst/Single 传输时触发。 如果该中断被屏蔽则该寄存器不会被置 1。

注：禁止对本寄存器进行写操作。

11.4.15 目标传输中断状态寄存器（DMA_STATUSDSTTRAN）

偏移地址：0x300

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	STATUSDSTTRAN	R	0x0	bit x 为通道 x 的 DSTTRAN 中断输出状态。 该中断会在该通道完成一个响应目标外设握手信号的 Burst/Single 传输时触发。 如果该中断被屏蔽则该寄存器不会被置 1。

注：禁止对本寄存器进行写操作。

11.4.16 错误中断状态寄存器（DMA_STATUSERR）

偏移地址：0x308

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	STATUSERR	R	0x0	bit x 为通道 x 的 Err 中断输出状态。 该中断会在该通道在传输时收到 HRSP 上的 ERROR 响应时触发，并会导致传输被取消和该通道被关闭。 如果该中断被屏蔽则该寄存器不会被置 1。

注：禁止对本寄存器进行写操作。

11.4.17 传输中断屏蔽寄存器（DMA_MASKTFR）

偏移地址：0x310

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	MASKTFR_WE	W	0x0	bit (x+8)为通道 x 的写入激活位： 0x0：不写入 0x1：写入 当次有效，读取为 0
7:0	MASKTFR	R/W	0x0	bit x 为通道 x 的 TFR 中断屏蔽状态： 0x0：屏蔽 0x1：不屏蔽

使用方法举例：

- 要将 bit 0 置 1，请对该寄存器写入 0x101，bit 1 不受影响。
- 要将 bit 1 置 0，请对该寄存器写入 0x200，bit 0 不受影响。
- 要将 bit 0 和 bit 1 置 1，请对该寄存器写入 0x303。

11.4.18 Block 传输中断屏蔽寄存器（DMA_MASKBLOCK）

偏移地址：0x318

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	MASKBLOCK_WE	W	0x0	bit (x+8)为通道 x 的写入激活位： 0x0：不写入 0x1：写入 当次有效，读取为 0

位	名称	属性	复位值	描述
7:0	MASKBLOCK	R/W	0x0	bit x 为通道 x 的 Block 中断屏蔽状态： 0x0：屏蔽 0x1：不屏蔽

11.4.19 源传输中断屏蔽寄存器（DMA_MASKSRCTRAN）

偏移地址：0x320

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	MASKSRCTRAN_WE	W	0x0	bit (x+8)为通道 x 的写入激活位： 0x0：不写入 0x1：写入 当次有效，读取为 0
7:0	MASKSRCTRAN	R/W	0x0	bit x 为通道 x 的 SRCTRAN 中断屏蔽状态： 0x0：屏蔽 0x1：不屏蔽

11.4.20 目标传输中断屏蔽寄存器（DMA_MASKDSTTRAN）

偏移地址：0x328

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	MASKDSTTRAN_WE	W	0x0	bit (x+8)为通道 x 的写入激活位： 0x0：不写入 0x1：写入 当次有效，读取为 0
7:0	MASKDSTTRAN	R/W	0x0	bit x 为通道 x 的 DSTTRAN 中断屏蔽状态： 0x0：屏蔽 0x1：不屏蔽

11.4.21 错误中断屏蔽寄存器（DMA_MASKERR）

偏移地址：0x330

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留

位	名称	属性	复位值	描述
15:8	MASKERR_WE	W	0x0	bit (x+8)为通道 x 的写入激活位： 0x0：不写入 0x1：写入 当次有效，读取为 0
7:0	MASKERR	R/W	0x0	bit x 为通道 x 的 Err 中断屏蔽状态： 0x0：屏蔽 0x1：不屏蔽

11.4.22 传输中断清除寄存器（DMA_CLEARTFR）

偏移地址：0x338

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CLEARTFR	W	0x0	对 bit x 写入 1 可以清除通道 x 的 Tfr 中断

注：禁止对本寄存器进行读操作。

11.4.23 Block 传输中断清除寄存器（DMA_CLEARBLOCK）

偏移地址：0x340

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CLEARBLOCK	W	0x0	对 bit x 写入 1 可以清除通道 x 的 Block 中断

注：禁止对本寄存器进行读操作。

11.4.24 源传输中断清除寄存器（DMA_CLEARSRCTRAN）

偏移地址：0x348

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CLEARSRCTRAN	W	0x0	对 bit x 写入 1 可以清除通道 x 的 SRCTRAN 中断

注：禁止对本寄存器进行读操作。

11.4.25 目标中断清除寄存器（DMA_CLEARDSTTRAN）

偏移地址：0x350

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CLEARDSTTRAN	W	0x0	对 bit x 写入 1 可以清除通道 x 的 DSTTRAN 中断

注：禁止对本寄存器进行读操作。

11.4.26 错误中断清除寄存器（DMA_CLEARERR）

偏移地址：0x358

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CLEARERR	W	0x0	对 bit x 写入 1 可以清除通道 x 的 Err 中断

注：禁止对本寄存器进行读操作。

11.4.27 中断状态寄存器（DMA_STATUSINT）

偏移地址：0x360

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	ERR	R	0x0	STATUSERR 各个 bit 或运算的结果
3	DSTTRAN	R	0x0	STATUSDSTTRAN 各个 bit 或运算的结果
2	SRCTRAN	R	0x0	STATUSSRCTRAN 各个 bit 或运算的结果
1	BLOCK	R	0x0	STATUSBLOCK 各个 bit 或运算的结果
0	TFR	R	0x0	STATUSTFR 各个 bit 或运算的结果

注：禁止对本寄存器进行写操作。

11.4.28 源传输 Req 信号软件握手寄存器（DMA_REQSRCREG）

偏移地址：0x368

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	SRC_REQ_WE	W	0x0	bit (x+8)为通道 x 的写入激活位： 0x0：不写入 0x1：写入 当次有效，读取为 0
7:0	SRC_REQ	R/W	0x0	bit x 为通道 x 的源传输 Req 握手信号

11.4.29 源传输 Req 信号软件握手寄存器（DMA_REQDSTREG）

偏移地址：0x370

复位值：0x0000_0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	DST_REQ_WE	W	0x0	bit (x+8)为通道 x 的写入激活位： 0x0：不写入 0x1：写入 当次有效，读取为 0
7:0	DST_REQ	R/W	0x0	bit x 为通道 x 的目标传输 Req 握手信号

11.4.30 源传输 Single 信号软件握手寄存器（DMA_SGLREQSRCREG）

偏移地址：0x378

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	SRC_SGLREQ_WE	W	0x0	bit (x+8)为通道 x 的写入激活位： 0x0：不写入 0x1：写入 当次有效，读取为 0
7:0	SRC_SGLREQ	R/W	0x0	bit x 为通道 x 的源传输 Single 握手信号

11.4.31 目标传输 Single 信号软件握手寄存器（DMA_SGLREQDSTREG）

偏移地址：0x380

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	DST_SGLREQ_WE	W	0x0	bit (x+8)为通道 x 的写入激活位： 0x0：不写入 0x1：写入 当次有效，读取为 0
7:0	DST_SGLREQ	R/W	0x0	bit x 为通道 x 的目标传输 Single 握手信号

11.4.32 源传输 Last 信号软件握手寄存器（DMA_LSTSRCREG）

偏移地址: 0x388

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	LSTSRC_WE	W	0x0	bit (x+8)为通道 x 的写入激活位： 0x0：不写入 0x1：写入 当次有效，读取为 0
7:0	LSTSRC	R/W	0x0	bit x 为通道 x 的源传输 Last 握手信号

11.4.33 目标传输 Last 信号软件握手寄存器（DMA_LSTDSTREG）

偏移地址：0x390

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	LSTDST_WE	W	0x0	bit (x+8)为通道 x 的写入激活位： 0x0：不写入 0x1：写入 当次有效，读取为 0
7:0	LSTDST	R/W	0x0	bit x 为通道 x 的目标传输 Last 握手信号

11.4.34 DMA 模块使能寄存器（DMA_CFGREG）

偏移地址：0x398

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	DMA_EN	R/W	0x0	DMA 使能： 0x0：DMA 模块功能不使能 0x1：DMA 模块功能使能，要在使能通道之前使能此位

11.4.35 通道使能寄存器（DMA_CHENREG）

偏移地址：0x3A0

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留

位	名称	属性	复位值	描述
15:8	CH_EN_WE	W	0x0	bit (x+8)为通道 x 的写入激活位： 0x0：不写入 0x1：写入 当次有效，读取为 0
7:0	CH_EN	R/W	0x0	bit x 为通道 x 的使能位 传输完成后此位会自动置 0

使用方法举例：

- 要将 bit 0 置 1，请对该寄存器写入 0x101，bit 1 不受影响。
- 要将 bit 1 置 0，请对该寄存器写入 0x200，bit 0 不受影响。
- 要将 bit 0 和 bit 1 置 1，请对该寄存器写入 0x303。

11.5 使用流程

11.5.1 基本硬件流控使用方法

1. 配置 DMA_SARx 寄存器，指定源地址。
2. 配置 DMA_DARx 寄存器，指定目标地址。
3. 配置 DMA_CTLx 寄存器：
   A. 有需要时使能中断。
   B. 设置传输数据宽度和大小。
   C. 选择流控类型，注意 SRAM 没有握手信号，在 TT_FC 中将 SRAM 认定为储存器。
   D. 选择地址是否递增/递减。
   E. 选择 burst 传输长度。
4. 配置 DMA_CFGx 寄存器：
   A. 选择通道优先级。
   B. 设置是否自动重启传输。
   C. 选定握手信号。
   D. 将 HS_SEL 域设为 0。
5. 有需要时配置 DMA_MASKBLOCK 寄存器，开启对应通道的中断。
6. 配置 DMA_CFGREG 寄存器使能 DMA。
7. 配置 DMA_CHENREG 寄存器使能通道。
8. 等待中断或查询 DMA_CHENREG。
9. 清除中断。

11.5.2 软件流控使用方法

1. 配置 DMA_SARx 寄存器，指定源地址。
2. 配置 DMA_DARx 寄存器，指定目标地址。
3. 配置 DMA_CTLx 寄存器：
   A. 有需要时使能中断。
   B. 设置传输数据宽度和大小。
   C. 选择流控类型。
   D. 选择地址是否递增/递减。
   E. 选择 burst 传输长度。
4. 配置 DMA_CFGx 寄存器：
   A. 选择通道优先级。
   B. 将 HS_SEL 域设为 1。
5. 有需要时配置 DMA_MASKBLOCK 寄存器，开启对应通道的中断。
6. 配置 DMA_CFGREG 寄存器使能 DMA。
7. 配置 DMA_CHENREG 寄存器使能通道。
8. 软件写入软件握手寄存器，触发传输。
9. 等待中断或查询 DMA_CHENREG。
10. 清除中断。

12 数字摄像头接口（DCMI）

12.1 DCMI 概述

DCMI 可以接收来自摄像头的 8/10/12/14 位数据流，支持多种格式和 JPEG 传输。

12.2 主要特性

- 支持 8/10/12/14 位 DVP 接口
- 支持硬件/内嵌码同步
- 支持连续/单帧模式
- 支持窗口剪裁功能
- 支持内置 DMA 传输数据

12.3 管脚说明

表 12-1：DCMI 管脚说明

功能管脚	复用管脚	方向	功能描述
DCMI_PIXCLK	PA6	Input	数字摄像头时钟信号输入
DCMI_HSYNC	PA4	Input	数字摄像头行同步信号输入
DCMI_VSYNC	PB7	Input	数字摄像头列同步信号输入
DCMI_D0	PA9,PC6	Input	数字摄像头数据信号输入
DCMI_D1	PA10,PC7	Input	数字摄像头数据信号输入
DCMI_D2	PC8,PE0	Input	数字摄像头数据信号输入
DCMI_D3	PC9,PE1	Input	数字摄像头数据信号输入
DCMI_D4	PC11,PE4	Input	数字摄像头数据信号输入
DCMI_D5	PB6	Input	数字摄像头数据信号输入
DCMI_D6	PB8,PE5	Input	数字摄像头数据信号输入
DCMI_D7	PB9,PE6	Input	数字摄像头数据信号输入
DCMI_D8	PC10	Input	数字摄像头数据信号输入
DCMI_D9	PC12	Input	数字摄像头数据信号输入
DCMI_D10	PB5	Input	数字摄像头数据信号输入
DCMI_D11	PD2	Input	数字摄像头数据信号输入
DCMI_D12	PD0	Input	数字摄像头数据信号输入
DCMI_D13	PD1	Input	数字摄像头数据信号输入

12.4 功能描述

12.4.1 捕获数据

12.4.1.1 8 位数据

当采集数据宽度设定为 8 位时，每 4 个采集到的数据会填充 1 个字的储存空间，当采集到 1 个字的数据或检测到行结束（非 JPEG 模式）或帧结束时，会将一个字的数据写入 FIFO。数据储存格式如下表所示：

表 12-2：8 位数据存储格式

位	31:24	23:16	15:8	7:0
数据	Data4	Data3	Data2	Data1

12.4.1.2 10 位数据

当采集数据宽度设定为 10 位时，每 2 个采集到的数据会填充 1 个字的储存空间，当采集到 1 个字的数据或检测到行结束（非 JPEG 模式）或帧结束时，会将一个字的数据写入 FIFO。数据储存格式如下表所示：

表 12-3：10 位数据存储格式

位	31:26	25:16	15:10	9:0
数据	6’b0	Data2	6’b0	Data1

12.4.1.3 12 位数据

当采集数据宽度设定为 12 位时，每 2 个采集到的数据会填充 1 个字的储存空间，当采集到 1 个字的数据或检测到行结束（非 JPEG 模式）或帧结束时，会将一个字的数据写入 FIFO。数据储存格式如下表所示：

表 12-4：12 位数据存储格式

位	31:28	27:16	15:12	11:0
数据	4’b0	Data2	4’b0	Data1

12.4.1.4 14 位数据

当采集数据宽度设定为 14 位时，每 2 个采集到的数据会填充 1 个字的储存空间，当采集到 1 个字的数据或检测到行结束（非 JPEG 模式）或帧结束时，会将一个字的数据写入 FIFO。数据储存格式如下表所示：

表 12-5：14 位数据存储格式

位	31:30	29:16	15:14	13:0
数据	2’b0	Data2	2’b0	Data1

12.4.2 硬件同步模式

在硬件同步模式下，DCMI 使用 Vsync 和 Hsync 信号来控制信号采集，当 Vsync 和 Hsync 均无效时 DCMI 会接收数据传输，Hsync 有效表示已完成一行的传输，Vsync 有效表示已完成一帧的传输。Vsync、Hsync 和像素时钟的极性可编程。

12.4.3 JPEG 模式

JPEG 模式开启时，禁止使用窗口剪裁和内嵌码同步功能，Hsync 仅用于数据使能不会触发行结束写入 FIFO。

12.4.4 内嵌码同步模式

内嵌码同步模式开启时，使用数据流中包含的特定值来代替 Vsync 和 Hsync 信号，该模式仅在选择 8 位数据输入时可用。

在 DCMI_ESC 寄存器和 DCMI_ESCMASK 寄存器中设定检测的特征码，当数据流中出现 FF 00 00 XX，XX 与特征码相同时即可触发同步，开始或停止采集数据。

DCMI_ESCMASK 寄存器可以设定屏蔽特征码检测中的某些位，只要数据中未屏蔽的位符合 DCMI_ESC 寄存器中设定的值，即可触发同步。例如将 DCMI_ESC 中的 FS 设定为 AA，将 DCMI_ESCMASK 中的 FSM 设定为 F0，只要数据流中出现 FF 00 00 AX（X=0~F），即可触发帧开始同步。

12.4.5 窗口剪裁

通过设定 DCMI_CROPSTART 寄存器和 DCMI_CROPSIZE 寄存器，可以将图像采集范围限定在一个窗口区域内部。

12.4.6 单帧捕获

通过使用单帧捕获模式，可以使 DCMI 在完成一帧的捕获后自动关闭捕获。

12.5 寄存器

寄存器基地址：0x4020_0000

寄存器列表如下：

表 12-6：DCMI 寄存器列表

偏移地址	名称	描述
0x00	DCMI_CTRL	控制寄存器
0x04	DCMI_STATUS	状态寄存器
0x08	DCMI_INTRAW	原始中断寄存器
0x0C	DCMI_INTEN	中断使能寄存器
0x10	DCMI_INTMASKED	中断状态寄存器
0x14	DCMI_INTCLR	中断清除寄存器
0x18	DCMI_ESC	内嵌码寄存器
0x1C	DCMI_ESCMASK	内嵌码屏蔽寄存器
0x20	DCMI_CROPSTART	剪裁窗口起点寄存器
0x24	DCMI_CROPSIZE	剪裁窗口尺寸寄存器
0x28	DCMI_DATA	数据寄存器
0x2C	DCMI_MSTCTRL	数据输出控制寄存器
0x30	DCMI_MSTADR	数据输出地址寄存器

12.5.1 控制寄存器 (DCMI_CTRL)

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	INPUT_OFFSET	R/W	0x0	选择当输入数据宽度为 8 位时，输入信号的来源： 0：输入信号来自 D0~D7 端口 1：输入信号来自 D2~D9 端口
14	DCMI_EN	R/W	0x0	DCMI 开关： 0：关闭 1：开启
13:12	RSV	-	-	保留
11:10	WIDTH	R/W	0x0	选择数据输入宽度： 0：8bit 1：10bit 2：12bit 3：14bit
9:8	FRAME_RATE	R/W	0x0	选择帧率控制： 0：采集所有帧 1：采集一半的帧 2/3：采集 1/4 的帧
7	VSYNC_POL	R/W	0x0	选择 VSYNC 垂直同步极性（此位指示数据在并行接口上无效时 VSYNC 引脚的电平）： 0：低电平时消隐（VSYNC 低电平有效，不传输数据） 1：高电平时消隐(VSYNC 高电平有效，不传输数据)

位	名称	属性	复位值	描述
6	HSYNC_POL	R/W	0x0	选择 HSYNC 水平同步极性（此位指示数据在并行接口上无效时 HSYNC 引脚的电平）： 0：低电平时消隐（HSYNC 低电平有效，不传输数据） 1：高电平时消隐（HSYNC 高电平有效，不传输数据）
5	CLK_POL	R/W	0x0	选择时钟极性（此位来配置像素时钟的捕获沿）： 0：下降沿有效 1：上升沿有效
4	ESC_EN	R/W	0x0	选择硬件或内嵌码同步： 0：硬件同步 1：内嵌码同步
3	JPEG_EN	R/W	0x0	选择开启 JPEG 模式： 0：不开启 JPEG 模式 1：开启 JPEG 模式，会停用帧率减少、内嵌码同步和窗口剪裁功能
2	CROP_EN	R/W	0x0	选择开启窗口剪裁： 0：不开启窗口剪裁 1：开启窗口剪裁
1	SINGLE_FRAME	R/W	0x0	选择开启单帧捕获： 0：不开启单帧捕获 1：开启单帧捕获，完成一帧的捕获后自动关闭捕获
0	CAPTURE_EN	R/W	0x0	选择开启捕获： 0：关闭捕获 1：开启捕获

12.5.2 状态寄存器 (DCMI_STATUS)

偏移地址：0x04

复位值：0x0000 0003

位	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	FIFO_EMPTY	R	0x0	指示 FIFO 状态： 0：空 1：非空
1	VSYNC_STATUS	R	0x1	指示垂直同步状态： 0：传输数据 1：消隐
0	HSYNC_STATUS	R	0x1	指示水平同步状态： 0：传输数据 1：消隐

12.5.3 原始中断寄存器 (DCMI_INTRAW)

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5	MST_ERR	R	0x0	输出数据发生传输错误时置位
4	HSYNC	R	0x0	Hsync 变为消隐时置位
3	VSYNC	R	0x0	Vsync 变为消隐时置位
2	ESC_ERR	R	0x0	接收到错误的内嵌码时置位
1	OVERFLOW	R	0x0	FIFO 溢出时置位
0	FRAME_END	R	0x0	完成一帧的捕获时置位

12.5.4 中断使能寄存器 (DCMI_INTEN)

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5	MST_ERR_EN	R/W	0x0	使能数据传输错误中断
4	HSYNC_EN	R/W	0x0	使能 Hsync 中断
3	VSYNC_EN	R/W	0x0	使能 Vsync 中断
2	ESC_ERR_EN	R/W	0x0	使能内嵌码错误中断
1	OVERFLOW_EN	R/W	0x0	使能 FIFO 溢出中断
0	FRAME_END_EN	R/W	0x0	使能帧捕获完成中断

12.5.5 中断状态寄存器 (DCMI_INTMASKED)

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5	MST_ERR_MASKED	R	0x0	数据传输错误中断状态
4	HSYNC_MASKED	R	0x0	Hsync 中断状态
3	VSYNC_MASKED	R	0x0	Vsync 中断状态
2	ESC_ERR_MASKED	R	0x0	内嵌码错误中断状态
1	OVERFLOW_MASKED	R	0x0	FIFO 溢出中断状态
0	FRAME_END_MASKED	R	0x0	帧捕获完成中断状态

12.5.6 中断清除寄存器（DCMI_INTCLR)

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5	MST_ERR_CLR	W	0x0	对此位写 1 清除传输错误中断
4	HSYNC_CLR	W	0x0	对此位写 1 清除 Hsync 中断
3	VSYNC_CLR	W	0x0	对此位写 1 清除 Vsync 中断
2	ESC_ERR_CLR	W	0x0	对此位写 1 清除内嵌码错误中断
1	OVERFLOW_CLR	W	0x0	对此位写 1 清除 FIFO 溢出中断
0	FRAME_END_CLR	W	0x0	对此位写 1 清除帧捕获完成中断

12.5.7 内嵌码寄存器（DCMI_ESC)

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:24	FE	R/W	0x0	设定用于检测帧结束的内嵌码
23:16	LE	R/W	0x0	设定用于检测行结束的内嵌码
15:8	LS	R/W	0x0	设定用于检测行开始的内嵌码
7:0	FS	R/W	0x0	设定用于检测帧开始的内嵌码

12.5.8 内嵌码屏蔽寄存器（DCMI_ESCMASK)

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:24	FEM	R/W	0x0	设定用于检测帧结束的内嵌码检测屏蔽位： 设定为 0 的位将不会检测 设定为 1 的位会参与检测
23:16	LEM	R/W	0x0	设定用于检测行结束的内嵌码检测屏蔽位： 设定为 0 的位将不会检测 设定为 1 的位会参与检测
15:8	LSM	R/W	0x0	设定用于检测行开始的内嵌码检测屏蔽位： 设定为 0 的位将不会检测 设定为 1 的位会参与检测
7:0	FSM	R/W	0x0	设定用于检测帧开始的内嵌码检测屏蔽位： 设定为 0 的位将不会检测 设定为 1 的位会参与检测

12.5.9 剪裁窗口起点寄存器 (DCMI_CROPSTART)

偏移地址：0x20

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:29	RSV	-	-	保留
28:16	VERTICAL_ STARTPOINT	R/W	0x0	设定在开始捕获前跳过的行数
15:14	RSV	-	-	保留
13:0	HORIZONAL_ STARTPOINT	R/W	0x0	设定在每行开始捕获前跳过的像素时钟个数

12.5.10 剪裁窗口尺寸寄存器 (DCMI_CROPSIZE)

偏移地址：0x24

复位值：0x0000_0000

位	名称	属性	复位值	描述
31:29	RSV	-	-	保留
28:16	VERTICAL_SIZE	R/W	0x0	设定每帧捕获的行数
15:14	RSV	-	-	保留
13:0	HORIZONAL_SIZE	R/W	0x0	设定每行捕获的像素时钟个数

12.5.11 数据寄存器 (DCMI_DATA)

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	DATA	R	0x0	读取此寄存器以从 FIFO 中取出数据

12.5.12 数据输出控制寄存器 (DCMI_MSTCTRL)

偏移地址：0x2C

复位值：0x0000 0016

位	名称	属性	复位值	描述
31:6	RSV	-	-	保留

位	名称	属性	复位值	描述
5	ADRRST	R/W	0x0	设定是否在一帧传输完成之后自动重设传输地址： 0：不重设 1：一帧传输完成后传输地址自动重设为最初的设定地址
4:3	WIDTH	R/W	0x2	设定数据传输宽度： 0：8bit 1：16bit 2/3：32bit
2:1	ADRINCR	R/W	0x3	设定是否使能地址自增： 0/1：不使能，数据输出到同一个地址 2：使能，数据输出到递减的地址 3：使能，数据输出到递增的地址
0	MST_EN	R/W	0x0	设定是否使能数据自动输出： 0：不使能 1：使能

12.5.13 数据输出地址寄存器 (DCMI_MSTADR)

偏移地址：0x30

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	MST_ADR	R/W	0x0	设定数据输出目的地址，注意要与数据传输宽度对齐

12.6 使用流程

12.6.1 使用 DCMI 捕获数据并保存数据到储存器

1. 连接外部摄像头。
2. 配置 IO 相应管脚为 I2C_SCL 和 I2C_SDA，并检查 I2C 总线状态，使之处于空闲状态。
3. 配置 IO 相应管脚分别为 camera pdn，camera rst。
4. 配置 IO 相应管脚分别为 DCMI PCLK，DCMI VSYNC，DCMI HSYNC，DCMI D0–D13 复用 DCMI 功能。
5. camera pdn=0，低功耗关闭；通过设置 camera RST 硬件复位。注：需要根据所选型的摄像头数据手册进行配置，此处以 ov2640 为例，下同。
6. 通过 I2C 对摄像头进行软复位。读取摄像头 id。
7. 通过 I2C 总线配置摄像头为 uxga，rgb565 模式，具体参数配置参考示例代码。

8. 通过 I2C 总线配置摄像头屏幕选取大小为 320width*240heigh。
9. 通过 I2C 总线配置摄像头时钟频率。
10. 使能 DCMI 时钟，释放 DCMI 复位。
11. 使能 dcmi frameend 中断，使能 DCMI 硬中断。当 dcmi_frameend 中断触发时，表示 DCMI 从外部 camera 采集过来的一帧数据已完整保存到目的存储区域。
12. 配置 DCMI 接收数据宽度为 8bit，配置输入信号为 d0~d7。注：此步需参考硬件设计及传输数据的模式。
13. 配置 DCMI Vsync 为低，DCMI Hsync 为低，DCMI PCLK 为高，使能硬件同步。
14. 设置裁剪窗口起点及尺寸，DCMI_CROPSTART 和 DCMI_CROPSIZE，应小于等于摄像头屏幕选取尺寸。
15. 开启窗口裁剪 crop_en = 1。
16. 配置采集帧率为所有，使能连续捕获，关闭 jpeg 接收。
17. 设置 DCMI 数据输出目的地址 mst_adr 为目的存储区合法地址。
18. 设置一帧传输完成后地址自动重设；DCMI 输出数据传输宽度为 8bit；设置输出地址自动加 1。注：数据宽度需要根据实际应用做配置。
19. 使能数据自动输出 mst_en = 1。
20. DCMI 使能，DCMI 捕获使能，DCMI 将捕获数据存储到指定存储区。

12.6.2 使用 DCMI 捕获数据并直接传输至目的地址

1. 连接外部摄像头。
2. 配置 IO 相应管脚为 I2C_SCL 和 I2C_SDA，并检查 I2C 总线状态，使之处于空闲状态。
3. 配置 IO 相应管脚分别为 camera pdn，camera rst。
4. 配置 IO 相应管脚分别为 DCMI PCLK，DCMI VSYNC，DCMI HSYNC，DCMI D0–D13 复用 DCMI 功能。
5. camera pdn=0， 低功耗关闭；通过设置 camera rst 硬件复位。注：需要根据所选型的摄像头数据手册进行配置，此处以 ov2640 为例，下同。
6. 通过 I2C 对摄像头进行软复位。读取摄像头 id。
7. 通过 I2C 总线配置摄像头为 uxga，rgb565 模式，具体参数配置参考示例代码。
8. 通过 I2C 总线配置摄像头屏幕选取大小为 320width*240heigh。
9. 通过 I2C 总线配置摄像头时钟频率。
10. 使能 DCMI 时钟，释放 DCMI 复位。
11. 配置 DCMI 接收数据宽度为 8bit，配置输入信号为 d0~d7。注：此步需参考硬件设计及传输数据的模式。
12. 配置 DCMI Vsync 为低，DCMI Hsync 为低，DCMI PCLK 为高，使能硬件同步。

13. 设置裁剪窗口起点及尺寸，DCMI_CROPSTART 和 DCMI_CROPSIZE，应小于等于摄像头屏幕选取尺寸。
14. 开启窗口裁剪 crop_en = 1。
15. 配置采集帧率为所有，使能连续捕获，关闭 jpeg 接收。
16. 设置 DCMI 数据输出目的地址 mst_adr 为目的外设地址，例如 emc i80 数据输入寄存器地址。
17. DCMI 输出数据传输宽度为 16bit；设置输出地址不变。注：数据宽度需要根据实际应用做配置。
18. 使能数据自动输出 mst_en = 1。
19. DCMI 使能，DCMI 捕获使能。DCMI 将捕获数据传输到指定外设地址。

12.6.3 使用 DCMI 捕获数据通过 DMA 传输至目的地址

1. 连接外部摄像头。
2. 配置 IO 相应管脚为 I2C_SCL 和 I2C_SDA，并检查 I2C 总线状态，使之处于空闲状态。
3. 配置 IO 相应管脚分别为 camera pdn，camera rst。
4. 配置 IO 相应管脚分别为 DCMI PCLK，DCMI VSYNC，DCMI HSYNC，DCMI D0–D13 复用 DCMI 功能。
5. camera pdn=0， 低功耗关闭；通过设置 camera rst 硬件复位。注：需要根据所选型的摄像头数据手册进行配置，此处以 ov2640 为例，下同。
6. 通过 I2C 对摄像头进行软复位。读取摄像头 id。
7. 通过 I2C 总线配置摄像头为 uxga，rgb565 模式，具体参数配置参考示例代码。
8. 通过 I2C 总线配置摄像头屏幕选取大小为 320width*240heigh。
9. 通过 I2C 总线配置摄像头时钟频率。
10. 使能 DCMI 时钟，释放 DCMI 复位。
11. 配置 DCMI 接收数据宽度为 8bit，配置输入信号为 d0~d7。注：此步需参考硬件设计及传输数据的模式。
12. 配置 DCMI Vsync 为低，DCMI Hsync 为低，DCMI HCLK 为高，使能硬件同步，。
13. 设置裁剪窗口起点及尺寸，DCMI_CROPSTART 和 DCMI_CROPSIZE，应小于等于摄像头屏幕选取尺寸。
14. 开启窗口裁剪 crop_en = 1。
15. 配置采集帧率为所有，使能连续捕获，关闭 jpeg 接收。
16. 设置 DCMI 数据输出目的地址 mst_adr 为目的外设地址，例如 emc i80 数据输入寄存器地址。
17. DCMI 输出数据传输宽度为 16bit；设置输出地址不变。注：数据宽度需要根据实际应用做配置。
18. 关闭数据自动输出 mst_en = 0。此模式下输出地址重设，输出数据宽度，输出地址等配置无效。DMA 方式默认内部采用 32bit 数据宽度传输。
19. DCMI 使能，DCMI 捕获使能。

20. 使能 DMA1 时钟，释放 DMA1 复位。举例：DMA 搬运 DCMI 数据到 EMC。
21. 初始化 DMA 及通道配置，例如 DMA1 通道 1，peripheral 到 memory ，源地址不变，目的地址不变。源、目的 DMA burst 长度均为 1。源数据传输宽度为 32bit，目地数据传输宽度为 16bit。源、目的均为硬件同步。源握手信号为 signal1，目的握手信号为任意值（memory）。
22. 使能 DMA 模块。使能 DMA1 中断，挂接中断处理函数。
23. 启动 DMA1 传输数据，本例：DMA1 通道 1，源地址为 DCMI_DATA 寄存器，目的地址为 EMC_DATA 寄存器，配置传输数据长度为一行像素总字节数/4，DMA 配置详见 DMA 章节介绍或示例代码。
24. 每行数据传输完毕后进入 DMA 中断，进行行计数，启动下一行 DMA 数据传输。当一帧传输完毕后，清行计数，此时采集的一帧数据已完整传输至目的外设，启动新一帧数据的第一次 DMA 传输。

13 外部存储控制器（EMC）

13.1 EMC 概述

模块支持 SRAM，NOR FLASH，I80 接口功能。存储器接口控制器 EMC(External Memory Controller)负责控制信息的传输，该传输是在内部局部总线和外部存储器模块（同步或者异步）之间进行的。同时此模块也可以作为 TFT-LCD 控制器使用，用来支持 8080 接口的 TFT-LCD。

13.2 主要特性

- 支持片外 Sram、Norflash 扩展
- 支持 8080 TFT-LCD 控制
- 支持写保护功能
- 支持外部接口 8/16 可配
- 读写操作等待时间可配

13.3 管脚说明

表 13-1：EMC 管脚说明

功能管脚	复用管脚	方向	功能描述
EX_NCE（csn0)	PD7	Output	片选信号
EX_RS(rs)	PD6	Output	数据和命令选择
EX_WR(wen)	PD5	Output	写有效
EX_RD(oen)	PD4	Output	读有效
EX_BA[1]（EBn[1])	PE1	Output	高 8 位数据有效位控制
EX_BA[0]（EBn[0])	PE0	Output	低 8 位数据有效位控制
EX_RST	-	Output	用一个 GPIO 来控制
EX_D0	PD14	Input/Output	数据输入输出
EX_D1	PD15	Input/Output	数据输入输出
EX_D2	PD0	Input/Output	数据输入输出
EX_D3	PD1	Input/Output	数据输入输出
EX_D4	PE7	Input/Output	数据输入输出
EX_D5	PE8	Input/Output	数据输入输出
EX_D6	PE9	Input/Output	数据输入输出
EX_D7	PE10	Input/Output	数据输入输出
EX_D8	PE11	Input/Output	数据输入输出

功能管脚	复用管脚	方向	功能描述
EX_D9	PE12	Input/Output	数据输入输出
EX_D10	PE13	Input/Output	数据输入输出
EX_D11	PE14	Input/Output	数据输入输出
EX_D12	PE15	Input/Output	数据输入输出
EX_D13	PD8	Input/Output	数据输入输出
EX_D14	PD9	Input/Output	数据输入输出
EX_D15	PD10	Input/Output	数据输入输出
MEM_A0	PB10	Output	地址线
MEM_A1	PB11	Output	地址线
MEM_A2	PB12	Output	地址线
MEM_A3	PB13	Output	地址线
MEM_A4	PC0	Output	地址线
MEM_A5	PC1	Output	地址线
MEM_A6	PC2	Output	地址线
MEM_A7	PC3	Output	地址线
MEM_A8	PC4	Output	地址线
MEM_A9	PC5	Output	地址线
MEM_A10	PE2	Output	地址线
MEM_A11	PE3	Output	地址线
MEM_A12	PE4	Output	地址线
MEM_A13	PE5	Output	地址线
MEM_A14	PE6	Output	地址线
MEM_A15	PD3	Output	地址线
MEM_A16	PD11	Output	地址线
MEM_A17	PD12	Output	地址线
MEM_A18	PD13	Output	地址线

13.4 功能描述

13.4.1 片选

EMC_CSCR0 寄存器中的 CSen 信号对应着一个外设的片选使能。设置寄存器内的 CSen 不意味着外部片选信号 CSnx 为低，只有当寄存器内部 CSen 为 1，并且对相应的外设地址区域进行读写操作时，外部 CSnx 信号才会有效。

13.4.2 EBn 信号时序

对于 CSnx 写操作来说，EBn 信号的建立和取消时间可以灵活地进行配置。然而对于读操作来说，仅仅建立时间被配置。读操作的取消时间总是在上升沿末尾。

写操作时序图如下图所示：

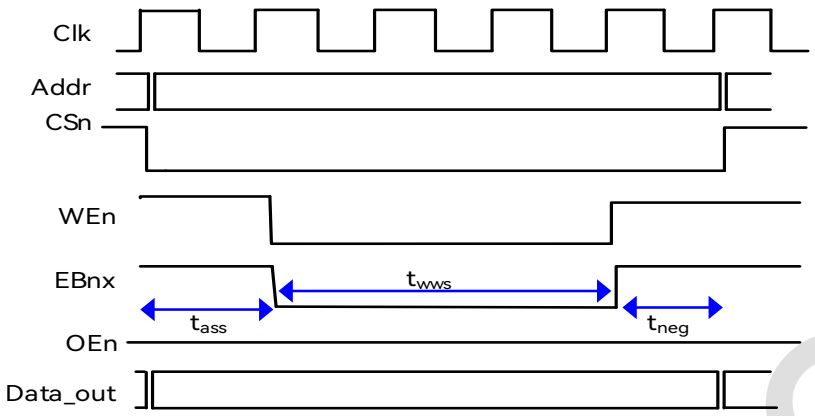


图 13-1：片外存储控制器写操作时序图

读操作时序图如下图所示：

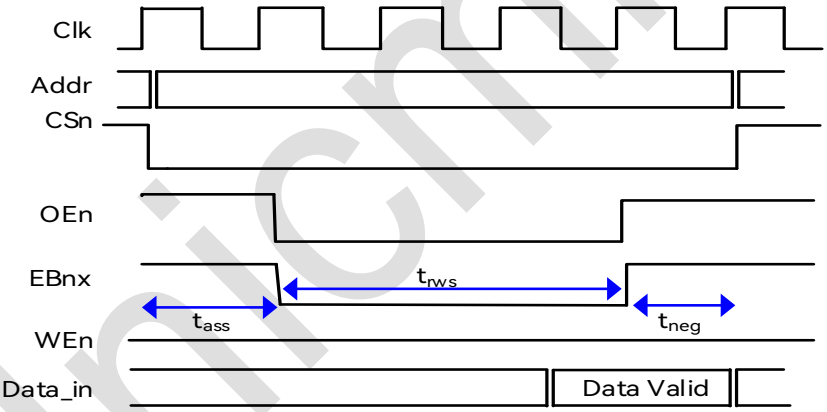


图 13-2：片外存储控制器读操作时序图

片外存储器读写时序，对应的时间如下表所示：

表 13-2：片外存储器读写时序对应时间

Wats/Rats/Aats	Tass	Wnts	Tneg	Wws/Rws	Twws/Trws
000	0	000	0	0000	1
001	1	001	1	0001	2
010	2	010	2	0010	3
011	3	011	3	0011	4
100	4	100	4	0100	5
101	5	101	5	0101	6
110	6	110	6	0110	7

Wats/Rats/Aats	Tass	Wnts	Tneg	Wws/Rws	Twws/Trws
111	7	111	7	0111	8
-	-	-	-	1000	9
				1001	10
				1010	11
				1011	12
				1100	13
				1101	14
				1110	15
				1111	16

注：对于 TFT 读情景，Tneg 有效，即时序中含有 Tneg 时间。

13.4.3 字节使能管脚

EMC_CSCR0 寄存器中的 PS 根据外部连接的外设的数据位宽来设置。EMC 控制器支持外部数据为 8/16bit 的外设，CPU 可以以高于外设数据的位宽的访问方式进行访问。例如外设数据宽度为 8bit，当 CPU 发起一次 32bit 写命令时，EMC 控制器内部会将 32bit 的命令分成四次独立的写操作，从而达到写 32bit 的目的。当 CPU 发起低于外设最小规定数据宽度的访问时，会发生错误，软件应避免发起这样的操作。

下表为 EBn 与外设数据线之间的对应关系。

表 13-3：EBn 与外设数据线对应关系

外设支持的最小访问数据位宽（单位：bit）	外设的数据位宽（单位：bit）	EBnx	外部数据接口信号
8	16	EBnx[1]	Data_in/Data_out[15:8]
		EBnx[0]	Data_in/Data_out[7:0]
16	16	EBnx[1]	-
		EBnx[0]	Data_in/Data_out[15:0]
8	8	EBnx[1]	-
		EBnx[0]	Data_in/Data_out[7:0]

13.4.4 TFT-LCD 控制

EMC 模块支持 8080 TFT-LCD 控制。8080 TFT-LCD 时序接口如下图所示：

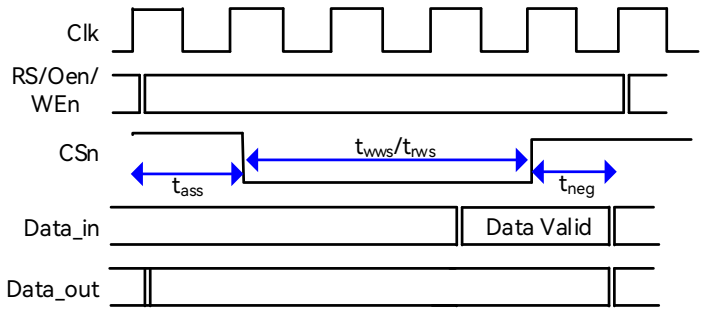


图 13-3：I8080 TFT-LCD 时序接口

EMC 模块中 RS 用来选择是命令操作还是数据传输操作，其根据此次操作的寄存器为 EMC_CMD 或 EMC_DATA 来决定。

注意，对于 TFT 读情景，Tneg 有效，即读时序中含有 Tneg 时间。

13.5 寄存器描述

EMC 寄存器基地址：0x6A00_0000

寄存器列表如下：

表 13-4：EMC 寄存器列表

偏移地址	名称	描述
0x00	EMC_CSCR0	片选控制寄存器 0
0x10	EMC_TFTS	选择 TFT 选择接口寄存器
0x14	EMC_CMD	TFT 模式下 CMD 通道寄存器
0x18	EMC_DATA	TFT 模式下 DATA 通道寄存器
0x1C	EMC_SEG0	区域划分寄存器 0

区域地址首地址：0x6800_0000

表 13-5：片选区域首地址

偏移地址	名称	描述
0x800000*x	UM_SEG_ADDRx	片选 x 区域首地址，x 取值 0~3

以下各节详细介绍寄存器。

13.5.1 片选控制寄存器（EMC_CSCR0）

偏移地址: 0x00

复位值：0x7770 2FF0

位	名称	属性	复位值	描述
31	RSV	-	-	保留

位	名称	属性	复位值	描述
30:28	WATS	R/W	0x7	写信号有效时间选择： 定义为CSn信号有效到EBn信号有效之间的时间间隔
27	RSV	-	-	保留
26:24	WNTS	R/W	0x7	写信号无效时间选择： 定义为EBn信号无效到CSn信号无效之间的时间间隔
23	RSV	-	-	保留
22:20	RATS	R/W	0x7	读信号有效时间选择： 定义为CSn信号有效到EBn/OEn信号有效之间的时间间隔
19:15	RSV	-	-	保留
14	RO	R/W	0	Ro位用来限制对相应片选下的地址域进行写访问： 1：只有读访问被允许 0：读写访问都被允许 当访问一个片选的存储空间时，片选逻辑将把Ro位同内部read/write信号相比较。如果片选逻辑检测到一个违例事件，访问将被忽视 1：只允许读访问；片选逻辑将忽视写访问 0：读写访问都被允许 此位在TFT模式下不起作用
13	PS	R/W	0x1	外部端口位宽选择： 0：16位端口 1：8位端口
12	RSV	-	-	保留
11:8	WWS	R/W	0xF	Wws字段决定着写等待状态的时钟周期个数
7:4	RWS	R/W	0xF	Rws字段决定着读等待状态的时钟周期个数
3:1	RSV	-	-	保留
0	CSEN	R/W	0x0	CSen位用来使能片选逻辑： 当片选被禁止时，外部片选信号将会是高电平 1：使能片选 0：禁止片选

13.5.2 TFT 模式通道寄存器（EMC_TFTS）

偏移地址： 0x10

复位值：0x0000 0003

位	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1:0	TFTS	R/W	3	TFT模式选择： 00：CSn0控制的外设为TFT模式，用于8080 TFT-LCD控制 01：CSn0控制的外设为SRAM模式，用于外部拓展SRAM 10：CSn0控制的外设为NORFlash模式，用于外部拓展NORFlash

13.5.3 TFT 命令通道寄存器（EMC_CMD）

偏移地址: 0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CMD	W	0x0	如果外设选择为TFT模式，向此寄存器中读写数据，数据将被理解为Cmd的内容，会发起TFT Cmd操作。

13.5.4 TFT 数据通道寄存器（EMC_DATA）

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	默认值	描述
31:0	DATA	R/W	0x0	如果外设选择为TFT模式，向此寄存器中读写数据，数据将被理解为数据的内容，会发起TFT Data操作。

13.5.5 区域划分寄存器（EMC_SEG0）

偏移地址：0x1C

复位值：0x0FFF 8000

位	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:15	SIZE	R/W	0x1FFF	CSen0对应的存储区域（SEG0）的大小设置位： SEG0（容量：byte）=（Size + 1）* 1K
14:0	START	R/W	0	CSen0对应的存储区域（SEG0）的起始地址，地址以1Kbyte对齐。即此处地址为1Kbyte为单位，每一个SEG最小单位为1Kbyte

13.6 使用流程

13.6.1 片外存储器读操作

1. 配置 IO 管脚分别为 EMC CSn，WEn，RS，OEn，D0~D15，A0~A18，DQM0，DQM1 复用 EMC 功能。
2. 使能 EMC 时钟，释放 EMC 复位。
3. 配置 EMC CSN0 为非 TFT 模式。
4. 设置 EMC_CSCR0 寄存器的 bit[30:28]，bit[26:24]，bit[22:20]，bit[11:8]，bit[7:4]，设置值可参

考默认值或根据所使用外设参数配置。

5. 设置 EMC_CSCR0[14]为只读/读写允许。外部端口位宽设置为 16bit 或 8bit，需根据外设参数设置。
6. 获取片选存储区域首地址 UM_SEG_ADDR0。
7. 设置区域划分寄存器 EMC_SEG0。
8. 使能 CSN0。
9. 对相应的合法片选存储区域发起读操作。

13.6.2 片外存储器写操作

1. 配置 IO 管脚分别为 EMC CSn，WEn，RS，OEn，D0~D15，A0~A18，DQM0，DQM1 复用 EMC 功能。
2. 使能 EMC 时钟，释放 EMC 复位。
3. 配置 EMC CSN0 为非 TFT 模式。
4. 设置 EMC_CSCR0 寄存器的 bit[30:28]，bit[26:24]，bit[22:20]，bit[11:8]，bit[7:4]，设置值可参考默认值或根据所使用外设参数配置。
5. 设置 EMC_CSCR0[14]为读写允许。外部端口位宽设置为 16bit 或 8bit，需根据外设参数设置。
6. 获取片选存储区域首地址 UM_SEG_ADDR0。
7. 设置区域划分寄存器 EMC_SEG0。
8. 使能 CSN0。
9. 对相应的合法片选存储区域发起写操作。

13.6.3 TFT-LCD 操作

1. 配置 IO 管脚分别为 EMC CSn，WEn，RS，OEn，D0~D15 复用 EMC 功能。
2. 使能 EMC 时钟，释放 EMC 复位。
3. 配置 EMC CSN0 为 TFT 模式。
4. 设置 EMC_CSCR0 寄存器的 bit[30:28]，bit[26:24]，bit[22:20]，bit[11:8]，bit[7:4]，设置值可参考默认值或根据所使用外设参数配置。
5. 设置 EMC_CSCR0[14]为读写允许。外部端口位宽设置为 16bit 或 8bit，需根据外设参数设置。
6. 使能 CSN0。
7. 向 EMC_CMD 写入数据发起对外设命令传输。
10. 向 EMC_DATA 读写数据发起对外设数据传输。

14 硬件加速运算协处理器（Cordic）

14.1 概述

Cordic 控制器可以使用 Cordic 算法计算$m \cdot \sin\theta$、$m \cdot \cos\theta$、atan2(y,x)、$\sqrt{x^2+y^2}$、$y \cdot x$、y/x、$\sinh w$、$\cosh w$、$\tanh^{-1}(y/x)$、ln(x)、$\sqrt{x}$ 等函数。

14.2 主要特性

- 内部使用 24 位精度计算
- AHB 接口可以输入输出 32 或 16 位数据
- 支持 DMA 方式传输
- 输入输出数据地址可配置

14.3 功能描述

14.3.1 数据格式与输入输出

本模块内部采用 24 位数据进行运算，数据输入输出可以选择 32 位或 16 位格式来进行数据传输。本节数据格式范围适用于所有功能模式。

当选用 32 位格式时，高 24 位为输入输出的数据，低 8 位无用；如果带符号，可在系统中使用带符号的 32 位型来操作。数据的小数点位于第[23]位与第[22]位之间，如果数据带符号，整个数据的表示范围为 [-1, (2^{23}-1)/2^{23}]，也就是[0x80000000,0x7FFFFFFF]；如果数据不带符号，表示范围为 [0, (2^{24}-1)/2^{23}]，也就是[0x0,0xFFFFFF00]。以下为数据转换举例说明【适用于正负数】:

输入数据 x[DEC]	计算公式	转换后数据[DEC]	转换后数据[HEX]
0.25	x*2^{31}	536870912	0x20000000

当选用 16 位格式时，16 位数据为内部数据的高 16 位，输入数据的低 8 位用 0 填充；如果带符号，可在系统中使用带符号的 16 位型来操作。如果数据带符号，整个数据的表示范围为 [-1, (2^{15}-1)/2^{15})，也就是[0x8000,0x7FFF]；如果数据不带符号，表示范围为 [0, (2^{16}-1)/2^{15}]，也就是[0x0,0xFFFF]。

输入数据 x[DEC]	计算公式	转换后数据[DEC]	转换后数据[HEX]
0.25	x*2^{15}	8192	0x00002000

进行运算时如果需要输入 2 个数据，当 2 个数据均被写入后，运算自动开始；如果只需要 1 个数据，写入 1 个数据就会使运算自动开始。

14.3.2 功能模式 0：$m \cdot \sin\theta$ / $m \cdot \cos\theta$

当选用功能模式 0：$m \cdot \sin\theta$ / $m \cdot \cos\theta$时，输入输出数据如下所示：

表 14-1：Cordic 控制器功能模式 0 输入输出表

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	θ	m	$m \cdot \sin\theta$	$m \cdot \cos\theta$
符号位	True	True	True	True
备注	单位为 $\pi\, rad$	可用 0x7FFFFF(XX)(32 位模式)或 0x7FFF(16 位模式)表示 1，此时不会计算乘法	-	-

相关计算：$\tan\theta = \sin\theta \div \cos\theta$

注：

1. 符号位：表示数据是否支持符号位，数据带符号位时最高位表示符号位。

   True：支持符号位，False：不支持符号位。

2. 数据输入范围请参考 14.3.1 数据格式与输入输出章节。

3. 以上注释适用于所有功能模式。

14.3.3 功能模式 1：atan2(y,x) / $\sqrt{(x^2+y^2)}$

当选用功能模式 1：atan2(y,x) / $\sqrt{x^2+y^2}$时，输入输出数据如下所示：

表 14-2：Cordic 控制器功能模式 1 输入输出表

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	y	x	atan2(y, x)	$\sqrt{x^2+y^2}$
符号位	True	True	True	False
备注	-	-	单位为 $\pi\, rad$	Bit[23]表示小数点前是 0 或 1，不表示符号，表示范围为[0,$\sqrt{2}$]

相关计算：$\tan^{-1} x = \mathrm{atan2}(x \cdot 2^{-n}, 2^{-n})$

14.3.4 功能模式 2：$y \cdot x$

当选用功能模式 2：$y \cdot x$时，输入输出数据如下所示：

表 14-3：Cordic 控制器功能模式 2 输入输出表

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	y	x	$y \cdot x$	-
符号位	True	True	True	-

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
备注	-	-	-	-

14.3.5 功能模式 3：y/x

当选用功能模式 3：y/x时，输入输出数据如下所示：

表 14-4：Cordic 控制器功能模式 3 输入输出表

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	y	x	y/x	-
符号位	True	True	True	-
备注	要求 $\|y\| \le \|x\|$		-	-

14.3.6 功能模式 4：$\sinh w$ / $\cosh w$

当选用功能模式 4：$\sinh w$ / $\cosh w$时，输入输出数据如下所示：

表 14-5：Cordic 控制器功能模式 4 输入输出表

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	$2^{-1} \cdot w$	-	$2^{-1} \cdot \sinh w$	$2^{-1} \cdot \cosh w$
符号位	True	-	True	False
备注	$w \in [-1.1181, 1.1181]$	-	范围 [0, 1.366]	范围 [0, 1.693]

相关计算：$e^w = \sinh w + \cosh w$

14.3.7 功能模式 5：$\tanh^{-1}(y/x)$

当选用功能模式 5：$\tanh^{-1}(y/x)$时，输入输出数据如下所示：

表 14-6：Cordic 控制器功能模式 5 输入输出表

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	y	x	$2^{-1} \cdot \theta$	-
符号位	True	False	True	-
备注	可以从$y \in [-1,1)$，$x \in [0,2)$的范围内取点(y,x)，也可以输入 $y = 2^{-n} \cdot \tanh\theta$，$x = 2^{-n}$，要求 $(y/x) \in [-0.8069, 0.8069]$		-	-

14.3.8 功能模式 6：ln(x)

当选用功能模式 6：ln(x)时，输入输出数据格式根据输入数据的范围变化，需要在控制寄存器中输入数值范围 SCALE，如下所示：

当$x \in [0.1069, 1)$时，令 SCALE=0

表 14-7：Cordic 控制器功能模式 6 输入输出表 1

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	x	-	$2^{-2} \cdot \ln x$	-
符号位	False	-	True	-
备注	$x \in [0.1069, 1)$	-	-	-

当$x \in (1, 3)$时，令 SCALE=1

表 14-8：Cordic 控制器功能模式 6 输入输出表 2

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	$2^{-1} \cdot x$	--	$2^{-2} \cdot \ln x$	-
符号位	False	--	True	-
备注	$x \in (1, 3)$	-	-	-

当$x \in [3, 7)$时，令 SCALE=2

表 14-9：Cordic 控制器功能模式 6 输入输出表 3

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	$2^{-2} \cdot x$	-	$2^{-2} \cdot \ln x$	-
符号位	False	-	True	-
备注	$x \in [3,7)$	-	-	-

当$x \in [7, 9.35]$时，令 SCALE=3

表 14-10：Cordic 控制器功能模式 6 输入输出表 4

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	$2^{-3} \cdot x$	-	$2^{-2} \cdot \ln x$	-
符号位	False	-	True	-
备注	$x \in [7, 9.35]$	-	-	-

相关计算：$w^t = e^{t \ln w}$

14.3.9 功能模式 7：Sqr(x)

当选用功能模式 7：$\sqrt{x}$时，也就是 Sqr(x)，输入输出数据格式根据输入数据的范围变化，需要在控制寄存器中输入数值范围 SCALE，如下所示：

当$x \in [0.1069, 1)$时，令 SCALE=0

表 14-11：Cordic 控制器功能模式 7 输入输出表 1

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	x	-	$\sqrt{x}$	-
符号位	False	-	False	-
备注	$x \in [0.1069, 1)$	-	-	-

当$x \in (1, 3)$时，令 SCALE=1

表 14-12：Cordic 控制器功能模式 7 输入输出表 2

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	$2^{-1} \cdot x$	-	$2^{-1} \cdot \sqrt{x}$	-
符号位	False	-	False	-
备注	$x \in (1,3)$	-	-	-

当$x \in [3,7)$时，令 SCALE=2

表 14-13：Cordic 控制器功能模式 7 输入输出表 3

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	$2^{-2} \cdot x$	-	$2^{-2} \cdot \sqrt{x}$	-
符号位	False	-	False	-
备注	$x \in [3,7)$	-	-	-

当$x \in [7,9.35]$时，令 SCALE=3

表 14-14：Cordic 控制器功能模式 7 输入输出表 4

	输入数据 1	输入数据 2	输出数据 1	输出数据 2
内容	$2^{-3} \cdot x$	-	$2^{-3} \cdot \sqrt{x}$	-
符号位	False	-	False	-
备注	$x \in [7,9.35]$	-	-	-

14.4 寄存器描述

寄存器基地址：0x4540_0000

寄存器列表如下：

表 14-15：Cordic 控制器寄存器列表

偏移地址	名称	描述
0x00	CORDIC_CTRL	控制寄存器
0x04	CORDIC_DIN1	输入寄存器 1
0x08	CORDIC_DIN2	输入寄存器 2
0x0C	CORDIC_DOUT1	输出寄存器 1
0x10	CORDIC_DOUT2	输出寄存器 2

以下各节详细介绍寄存器。

14.4.1 控制寄存器（CORDIC_CTRL)

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31	DATA_READY	R	0x0	0：当前无可用的运算结果 1：当前有可用的运算结果
30:22	RSV	-	-	保留
21:20	SCALE	R/W	0x0	ln(x)和$\sqrt{x}$计算参数,由x的数值确定
19:18	RSV	-	-	保留
17	DMA_OUT	R/W	0x0	输出数据 DMA 使能信号 0：关闭输出数据 DMA 1：开启输出数据 DMA
16	DMA_IN	R/W	0x0	输入数据 DMA 使能信号 0：关闭输入数据 DMA 1：开启输入数据 DMA
15:14	RSV	-	-	保留
13	MERGE_OUT	R/W	0x0	输出数据合并，仅当输出数据采用 16 位总线模式时可用 0：输出数据分两次输出 1：输出数据寄存器 1 的低 16 位读出第 1 个结果，高 16 位读出第 2 个结果
12	MERGE_IN	R/W	0x0	输入数据合并，仅当输入数据采用 16 位总线模式时可用： 0：输入数据分两次输入 1：输入数据寄存器 1 的低 16 位写入第 1 个参数，高 16 位写入第 2 个参数
11	ADDR_OUT	R/W	0x0	输出地址模式： 0：从两个输出数据寄存器分别读出数据 1：可将两个数据按顺序从输出数据寄存器 1 读出
10	ADDR_IN	R/W	0x0	输入地址模式： 0：需要将两个参数分别写入两个数据输入寄存器 1：可将两个参数按顺序写入输入数据寄存器 1
9	WIDTH_OUT	R/W	0x0	输出数据传输模式： 0：输出数据采用 32 位总线传输模式 1：输出数据采用 16 位总线传输模式
8	WIDTH_IN	R/W	0x0	输入数据传输模式： 0：输入数据采用 32 位总线传输模式 1：输入数据采用 16 位总线传输模式
7:4	ITERATION	R/W	12	（运算中的迭代次数/2） 次数越多越精确，耗时越长， 可用范围为[4, 12]
3	RSV	-	-	保留

位	名称	属性	复位值	描述
2:0	MODE	R/W	0x0	功能模式 0：选定模式 0： $\boldsymbol{m}\cdot\sin\boldsymbol{\theta}$ / $\boldsymbol{m}\cdot\cos\boldsymbol{\theta}$ 1：选定模式 1： atan2(y,x) / $\sqrt{x^2+y^2}$ 2：选定模式 2： $\boldsymbol{y}\cdot\boldsymbol{x}$ 3：选定模式 3： $\boldsymbol{y}/\boldsymbol{x}$ 4：选定模式 4： $\sinh w$ / $\cosh w$ 5：选定模式 5： $\tanh^{-1}(y/x)$ 6：选定模式 6： ln($\boldsymbol{x}$) 7：选定模式 7： $\sqrt{x}$

14.4.2 数据输入寄存器 1 (CORDIC_DIN1)

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	DIN1	W	0x0	将输入数据写入到此寄存器

14.4.3 数据输入寄存器 2 (CORDIC_DIN2)

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	DIN2	W	0x0	将输入数据写入到此寄存器

14.4.4 数据输出寄存器 1 (CORDIC_DOUT1)

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	DOUT1	R	0x0	从此寄存器读出运算结果

14.4.5 数据输出寄存器 2 (CORDIC_DOUT2)

偏移地址: 0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	DOUT2	R	0x0	从此寄存器读出运算结果

14.5 使用流程

1. 使能 CORDIC 模块时钟和释放复位。
2. 设置输入输出地址格式，数据格式等。
3. 设置运算次数（次数越高运算结果越精确）。
4. 设置运算方式、设置输入输出数据传输方式。
5. 根据所设置的运算方式输入合适的数据到 CORDIC_DIN1 和 CORDIC_DIN2。
6. 等待 CORDIC_CTRL[31]置位。
7. 读取两个输出数据 CORDIC_DOUT1 和 CORDIC_DOUT2。

14.6 计算示例

本节相关计算背景为 CORDIC_CTRL 默认配置下：【其他计算模式可参考此模式计算输入输出】

默认配置：输入输出数据格式：32bit

输入输出数据：分两个输入寄存器输入

数据传输方式：CPU

计算功能模式：功能模式 0

输入数据 1 转换为硬件规定格式输入计算方式：0.25[DEC] =0.25* 2^{31}[DEC]

=536,870,912 [DEC]

=0x20000000[HEX]

输入数据 2 转换为硬件规定格式输入计算方式：0.00390625[DEC] =0.00390625*2^{31} [DEC]

= 8388608[DEC]

= 0x00800000[HEX]

将以上计算结果分别输入 CORDIC_DIN1 和 CORDIC_DIN2，硬件即可开始运算，我们可以从 CORDIC_DOUT1 和 CORDIC_DOUT2 获取到计算结果。

验证计算结果：

输出数据 1 理论计算结果：m*sinθ = 0.00390625*（sin (0.25*π）) [DEC]

= 0.00390625*0.7071067811865 [DEC]

= 0.00276213586400995 12671907982[DEC] *2^{31}

= 5,931,641.601515722055569 [DEC]

= 5A8279[HEX]

输出数据 2 理论计算结果：m*cosθ = 0.00390625*（cos (0.25*π）) [DEC]

= 0.00390625*0.7071067811865 [DEC]

= 0.00276213586400995 [DEC]

= 5,931,641.601515722055569[DEC] *2^{31}

= 5A8279[HEX]

硬件自动计算结果为：0x5a8300【误差：135，误差主要来源于小数点】

15 高级加解密算法加速器（AES）

15.1 概述

AES 模块可以实现 FIPS PUB 197 中定义的 AES-128 或 AES-256 算法。

15.2 主要特性

- 支持 128 位和 256 位密钥长度
- 支持 CBC、ECB、CTR、CCM、CMAC 和 GCM 模式
- 支持加密和解密
- 按照 AES 规格规定，所有数据为大端对齐
- 密钥和信息使用分离的储存接口，密钥使用低容量储存，输入输出数据使用 FIFO 与接口交互
- 数据输入输出采用 DMA 进行操作
- 可用即时密钥拓展，无需额外储存空间
- 对于一个信息，仅支持基于分组的加解密，不支持信息切换
- 支持 3 密钥模式

15.3 功能描述

15.3.1 加解密运算

密钥加载之后，可以通过使能 AES_GO 控制位启动运算核心，它会自动读取数据进行运算。运算完成后会使能 AES_DONE 位并自动将数据输出。

15.4 寄存器描述

寄存器基地址：0x4510_0000

寄存器列表如下：

表 15-1：高级加密算法加速器 AES 寄存器列表

偏移地址	名称	描述
0x00	AES_MSGCFG	信息控制寄存器
0x04	AES_CTXCFG	密钥控制寄存器
0x08	AES_MSGTOTALBYTES	信息长度寄存器
0x0C	AES_MSGAADBYTES	附加信息长度寄存器

偏移地址	名称	描述
0x10	AES_GCMAADINFO	GCM 模式附加信息长度寄存器
0x14	AES_CTXKEYSEL	密钥选择寄存器
0x20	AES_CTXKEY0	密钥寄存器 0
0x24	AES_CTXKEY1	密钥寄存器 1
0x28	AES_CTXKEY2	密钥寄存器 2
0x2C	AES_CTXKEY3	密钥寄存器 3
0x30	AES_CTXKEY4	密钥寄存器 4
0x34	AES_CTXKEY5	密钥寄存器 5
0x38	AES_CTXKEY6	密钥寄存器 6
0x3C	AES_CTXKEY7	密钥寄存器 7
0x40	AES_CTXCBCKEY0	CBC 模式密钥寄存器 0
0x44	AES_CTXCBCKEY1	CBC 模式密钥寄存器 1
0x48	AES_CTXCBCKEY2	CBC 模式密钥寄存器 2
0x4C	AES_CTXCBCKEY3	CBC 模式密钥寄存器 3
0x50	AES_CTXCTR0	CTR 模式算子寄存器 0
0x54	AES_CTXCTR1	CTR 模式算子寄存器 1
0x58	AES_CTXCTR2	CTR 模式算子寄存器 2
0x5C	AES_CTXCTR3	CTR 模式算子寄存器 3
0x60	AES_CTXIV0	初始向量寄存器 0
0x64	AES_CTXIV1	初始向量寄存器 1
0x68	AES_CTXIV2	初始向量寄存器 2
0x6C	AES_CTXIV3	初始向量寄存器 3
0x70	AES_CTXMAC0	消息验证码寄存器 0
0x74	AES_CTXMAC1	消息验证码寄存器 1
0x78	AES_CTXMAC2	消息验证码寄存器 2
0x7C	AES_CTXMAC3	消息验证码寄存器 3
0x80	AES_INGRESSFIFO	输入 FIFO 寄存器
0x84	AES_INGFSTATUS	输入 FIFO 状态寄存器
0x88	AES_ENGRESSFIFO	输出 FIFO 寄存器
0x8C	AES_ENGFSTATUS	输出 FIFO 状态寄存器
0x90	AES_DMAINGLEN	DMA 输入长度寄存器
0x94	AES_INGDBCFG	DMA 输入爆发传输设置寄存器
0x98	AES_DMAENGLEN	DMA 输出长度寄存器
0x9C	AES_ENGDBCFG	DMA 输出爆发传输设置寄存器
0xA0	AES_DONESTATUS	完成状态寄存器
0xA4	AES_INGDMADONE	DMA 输入完成状态寄存器
0xA8	AES_ENGDMADONE	DMA 输出完成状态寄存器

以下各节详细介绍寄存器。

15.4.1 信息控制寄存器（AES_MSGCFG)

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:14	RSV	-	-	保留
13:12	KEY_SIZE	R/W	0x0	设定密钥位宽： 0：128bit 2：256bit 1 和 3：保留
11:8	MAC_LEN	R/W	0x0	设定消息验证码或初始向量长度（仅用于 CCM、CMAC、XCBC 模式）
7:4	ALG_MODE	R/W	0x0	选择模式： 0：ECB 1：CBC 2：CTR 3：CCM 4：CMAC 5：GCM
3	DIR	R/W	0x0	选择加解密方向： 0：解密 1：加密
2	MSG_BEGIN	R/W	0x0	指示当前操作包含信息的第一个字节。将会在操作之前自动取得需要的密钥。
1	MSG_END	R/W	0x0	指示当前操作包含信息的最后一个字节。如果操作结果是消息验证码或初始向量，将会在操作完成后自动储存到密钥储存空间。
0	AES_GO	R/W	0x0	启动加解密操作。开始操作后此位和其它控制位在 AES_DONE 置位前需要保持不变。

15.4.2 密钥控制寄存器（AES_CTXCFG)

偏移地址：0x04

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:4	CTX_INDEX	R/W	0x0	上下文页编号
3	INV_KEY_STR	R/W	0x0	储存上一轮密钥到密钥储存空间
2	INV_KET_RET	R/W	0x0	取出上一轮密钥
1	CTX_STR	R/W	0x0	把当前密钥储存到密钥储存空间
0	CTX_RET	R/W	0x1	取出当前密钥

15.4.3 信息长度寄存器 (AES_MSGTOTALBYTES)

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:0	TOTAL_BYTES	R/W	0x0	设定信息的总长度，包括所有分组，不包括附加信息。 在 CCM 和 GCM 模式中需要此项。

15.4.4 附加信息长度寄存器 (AES_MSGAADBYTES)

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	AAD_LEN	R/W	0x0	设定当前操作中附加信息的长度。 在 CCM 和 GCM 模式中需要此项。
15:0	MSG_LEN	R/W	0x0	设定当前操作中信息的长度，包括信息和附加信息。

15.4.5 GCM 模式附加信息长度寄存器 (AES_GCMAADINFO)

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:0	AAD_LEN_TOT	R/W	0x0	设定附加信息的总长度，包括所有分组。 在 GCM 模式中需要此项。

15.4.6 密钥选择寄存器 (AES_CTXKEYSEL)

偏移地址：0x14

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:3	RSV	-	-	保留

位	名称	属性	复位值	描述
2:0	KEY_SEL	R/W	0x1	选择密钥来源： 1：CPU 设置的密钥（寄存器） 2：OTP KEY1 4：OTP KEY2

15.4.7 密钥寄存器 0 (AES_CTXKEY0)

偏移地址：0x20

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	KEY_0	W	0x0	密钥[31:0] 注：ECB 模式秘钥寄存器

15.4.8 密钥寄存器 1 (AES_CTXKEY1)

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	KEY_1	W	0x0	密钥[63:32]

15.4.9 密钥寄存器 2 (AES_CTXKEY2)

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	KEY_2	W	0x0	密钥[95:64]

15.4.10 密钥寄存器 3 (AES_CTXKEY3)

偏移地址: 0x2C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	KEY_3	W	0x0	密钥[127:96]

15.4.11 密钥寄存器 4 (AES_CTXKEY4)

偏移地址：0x30

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	KEY_4	W	0x0	密钥[159:128]

15.4.12 密钥寄存器 5 (AES_CTXKEY5)

偏移地址：0x34

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	KEY_5	W	0x0	密钥[191:160]

15.4.13 密钥寄存器 6 (AES_CTXKEY6)

偏移地址：0x38

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	KEY_6	W	0x0	密钥[223:192]

15.4.14 密钥寄存器 7 (AES_CTXKEY7)

偏移地址：0x3C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	KEY_7	W	0x0	密钥[255:224]

15.4.15 CBC 模式密钥寄存器 0 (AES_CTXCBCKEY0)

偏移地址：0x40

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CBC_KEY_0	W	0x0	CBC 密钥[31:0] (XCBC-3K K2)

15.4.16 CBC 模式密钥寄存器 1 (AES_CTXCBCKEY1)

偏移地址：0x44

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CBC_KEY_1	W	0x0	CBC 密钥[63:32] (XCBC-3K K2)

15.4.17 CBC 模式密钥寄存器 2 (AES_CTXCBCKEY2)

偏移地址：0x48

复位值：0x0000_0000

位	名称	属性	复位值	描述
31:0	CBC_KEY_2	W	0x0	CBC 密钥[95:64] (XCBC-3K K2)

15.4.18 CBC 模式密钥寄存器 3 (AES_CTXCBCKEY3)

偏移地址：0x4C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CBC_KEY_3	W	0x0	CBC 密钥[127:96] (XCBC-3K K2)

15.4.19 CTR 模式算子寄存器 0 (AES_CTXCTR0)

偏移地址：0x50

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CCM_CTR_0	W	0x0	CTR/CCM 算子[31:0] (XCBC-3K K1)

15.4.20 CTR 模式算子寄存器 1 (AES_CTXCTR1)

偏移地址：0x54

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CCM_CTR_1	W	0x0	CTR/CCM 算子[63:32] (XCBC-3K K1)

15.4.21 CTR 模式算子寄存器 2 (AES_CTXCTR2)

偏移地址：0x58

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CCM_CTR_2	W	0x0	CTR/CCM 算子[95:64] (XCBC-3K K1)

15.4.22 CTR 模式算子寄存器 3 (AES_CTXCTR3)

偏移地址：0x5C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CCM_CTR_3	W	0x0	CTR/CCM 算子[127:96] (XCBC-3K K1)

15.4.23 初始向量寄存器 0 (AES_CTXIV0)

偏移地址：0x60

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	IV_0	W	0x0	初始向量[31:0]

15.4.24 初始向量寄存器 1 (AES_CTXIV1)

偏移地址：0x64

复位值：0x0000_0000

位	名称	属性	复位值	描述
31:0	IV_1	W	0x0	初始向量[63:32]

15.4.25 初始向量寄存器 2 (AES_CTXIV2)

偏移地址：0x68

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	IV_2	W	0x2	初始向量[95:64]

15.4.26 初始向量寄存器 3 (AES_CTXIV3)

偏移地址：0x6C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	IV_3	W	0x0	初始向量[127:96]

15.4.27 消息验证码寄存器 0 (AES_CTXMAC0)

偏移地址：0x70

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	MAC_0	W	0x0	CMAC/XCBC/CCM 消息验证码[31:0]

15.4.28 消息验证码寄存器 1 (AES_CTXMAC1)

偏移地址：0x74

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	MAC_1	W	0x0	CMAC/XCBC/CCM 消息验证码[63:32]

15.4.29 消息验证码寄存器 2 (AES_CTXMAC2)

偏移地址：0x78

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	MAC_2	W	0x0	CMAC/XCBC/CCM 消息验证码[95:64]

15.4.30 消息验证码寄存器 3 (AES_CTXMAC3)

偏移地址：0x7C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	MAC_3	W	0x0	CMAC/XCBC/CCM 消息验证码[127:96]

15.4.31 输入 FIFO(AES_INGRESSFIFO)

偏移地址：0x80

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	INGRESS_FIFO	W	0x0	AES 输入数据 FIFO 入口

15.4.32 输入 FIFO 状态寄存器 (AES_INGFSTATUS)

偏移地址：0x84

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	ING_FULL	R	0x0	指示输入数据 FIFO 满
0	ING_EMPTY	R	0x1	指示输入数据 FIFO 空

15.4.33 输出 FIFO(AES_ENGRESSFIFO)

偏移地址：0x88

复位值：不确定

位	名称	属性	复位值	描述
31:0	ENGRESS_FIFO	R	-	AES 输出数据 FIFO

15.4.34 输出 FIFO 状态寄存器 (AES_ENGFSTATUS)

偏移地址：0x8C

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	ENG_FULL	R	0x0	指示输出数据 FIFO 满
0	ENG_EMPTY	R	0x1	指示输出数据 FIFO 空

15.4.35 DMA 输入长度寄存器（AES_DMAINGLEN)

偏移地址：0x90

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	INGRESS_DMA_DATA_LENGTH	R/W	0x0	设定 DMA 向 AES 输入数据的总长度，用于生成握手信号。 即使使用 CPU 而不是 DMA 来传输数据，此项依然需要。

15.4.36 DMA 输入爆发传输设置寄存器（AES_INGDBCFG)

偏移地址：0x94

复位值：0xC000 0012

位	名称	属性	复位值	描述
31	PLAINTEXT_REQ	R/W	0x1	此位设定当前 AES 模式是否需要传入数据
30	DMA_EN	R/W	0x1	此位设定输入数据由 DMA 还是 CPU 传入： 0：使用 CPU 1：使用 DMA
29:6	RSV	-	-	保留
5:3	DST_MSIZE	R/W	0x2	DMA 的目标传输爆发传输长度： 0：1 1：4 2：8 3：16 4：32 5：64 6：128 7：256
2:0	DST_TR_WIDTH	R/W	0x2	DMA 的目标传输宽度： 0：8bit 1：16bit 2：32bit

15.4.37 DMA 输出长度寄存器 (AES_DMAENGLEN)

偏移地址：0x98

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	ENGRESS_DMA_DATA_LENGTH	R/W	0x0	设定 DMA 从 AES 输出数据的总长度，用于生成握手信号。

15.4.38 DMA 输出爆发传输设置寄存器 (AES_ENGDBCFG)

偏移地址：0x9C

复位值：0xC000 0012

位	名称	属性	复位值	描述
31	CIPERTEXT_REQ	R/W	0x1	此位设定当前 AES 模式是否需要传出数据
30	DMA_EN	R/W	0x1	此位设定输入数据由 DMA 还是 CPU 传入： 0：使用 CPU 1：使用 DMA
29:6	RSV	-	-	保留
5:3	SRC_MSIZE	R/W	0x2	DMA 的源传输爆发传输长度： 0：1 1：4 2：8 3：16 4：32 5：64 6：128 7：256
2:0	SRC_TR_WIDTH	R/W	0x2	DMA 的源传输宽度： 0：8 bits 1：16 bits 2：32 bits

15.4.39 完成状态寄存器 (AES_DONESTATUS)

偏移地址：0xA0

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	MAC_VAILD	R	0x0	指示 MAC 验证是否成功。 此位仅在 AES_DONE 置位后有效， AES_GO 置位后自动清零。

位	名称	属性	复位值	描述
0	AES_DONE	R	0x0	指示操作已经完成。 AES_GO 置位后自动清零。

15.4.40 DMA 输入完成状态寄存器 (AES_INGDMADONE)

偏移地址：0xA4

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	DMA_INGRESS_DONE	R	0x0	指示 DMA 输入已经完成传输。 读取清零。

15.4.41 DMA 输出完成状态寄存器 (AES_ENGDMADONE)

偏移地址：0xA8

复位值：0x0000 0000

	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	DMA_ENGRESS_DONE	R	0x0	指示 DMA 输出已经完成传输。 读取清零。

15.5 使用流程

15.5.1 AES 加密编程模型

1. 配置 AES_CTXKEYSEL 寄存器，并填充 AES_CTXKEYx 寄存器。
2. 输入数据到 AES_INGRESSFIFO 和从 AES_ENGRESSFIFO 取出数据。
3. 配置 AES_MSGCFG 设定参数，配置方向为加密，最后置位 AES_MSGCFG[0]。
4. 当 AES 模块完成当前分组的操作后，AES_DONE 位会置位并引起中断。
5. 查询或者中断判断完成标志。
6. 如果当前分组不是最后分组，重复步骤 2 和 3。
7. 如果当前分组是最后分组，且使用了能产生消息验证码的加密模式，从 AES_CTXMAC 取出消息验证码。

15.5.2 AES 解密编程模型

1. 配置 AES_CTXKEYSEL 寄存器，并填充 AES_CTXREGx 寄存器。
2. 配置 DMA 输入数据到 AES_INGRESSFIFO 和从 AES_ENGRESSFIFO 取出数据。
3. 配置 AES_MSGCFG 设定参数，配置方向为解密，最后置位 AES_MSGCFG[0]。
4. 当 AES 模块完成当前分组的操作后，AES_DONE 位会置位并引起中断。
5. 查询或者中断判断完成标志。
6. 如果使用了能产生消息验证码的加密模式，从 AES_CTXMAC 取出消息验证码。

16 安全散列算法加速器（SHA）

16.1 散列概述

SHA 模块可以实现 FIPS PUB 180-4 中定义的 SHA-256 算法。

16.2 主要特性

- 支持 256 位 ICV 长度
- 需要外部储存中的信息使用小端对齐且对齐到双字(4 字节对齐)
- 使用 32x32bit 的输入数据 FIFO
- 数据输入可采用 DMA 进行操作

16.3 功能描述

16.3.1 加解密运算

使能 SHA_GO 控制位启动运算核心，输入数据即可开始计算。运算完成后会使能 SHA_DONE 位。

16.3.2 OTP 数值比较

当 SHA 运算结果与 OTP ICV 数值相同时，仅能通过 SHA 模块得知 ICV 吻合，无法读出具体 ICV 数值。

要使用 OTP 比较功能，请将 OTP ICV 以小端对齐的方式写入到 OTP 储存区域，SHA-256 格式的 OTP ICV 请写入到 OTP 储存区域的[383:128]位，并对 OTP 储存区域的[127:0]写入 0。

16.4 寄存器描述

SHA 寄存器基地址：0x4520_0000

寄存器列表如下：

表 16-1：SHA 寄存器列表

偏移地址	名称	描述
0x00	SHA_CMD	控制命令寄存器
0x04	SHA_MODE	位数模式寄存器

偏移地址	名称	描述
0x08	SHA_MSGCFG	数据设定寄存器
0x18	SHA_MSGTOTALBYTES	数据长度寄存器
0x1C	SHA_DATAIN	数据输入寄存器
0x20	SHA_ICV00	ICV 读取寄存器 00
0x24	SHA_ICV01	ICV 读取寄存器 01
0x28	SHA_ICV02	ICV 读取寄存器 02
0x2C	SHA_ICV03	ICV 读取寄存器 03
0x30	SHA_ICV04	ICV 读取寄存器 04
0x34	SHA_ICV05	ICV 读取寄存器 05
0x38	SHA_ICV06	ICV 读取寄存器 06
0x3C	SHA_ICV07	ICV 读取寄存器 07
0x40	SHA_ICV08	ICV 读取寄存器 08
0x44	SHA_ICV09	ICV 读取寄存器 09
0x48	SHA_ICV10	ICV 读取寄存器 10
0x4C	SHA_ICV11	ICV 读取寄存器 11
0x50	SHA_VERIRESULT	校验结果寄存器
0x54	SHA_IRQEN	中断使能寄存器
0x58	SHA_IRQCLR	中断清除寄存器
0x5C	SHA_IRQSTATUS	中断状态寄存器
0x60	SHA_DMABURSTSIZE	DMA Burst 长度寄存器

以下各节详细介绍寄存器。

16.4.1 控制命令寄存器 (SHA_CMD)

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	SHA_DONE	R	0x0	指示完成当前段落的运算
0	SHA_GO	R/W	0x0	写 1 启动运算，完成运算后自动清零，写 0 无效。 读取可知目前是否在运算。

16.4.2 位数模式寄存器 (SHA_MODE)

偏移地址：0x04

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1:0	MODE	R/W	0x1	选择运算模式： 1：SHA-256 2：SHA-384 其它值无效。注：当设置为其它非法值，并触发 SHA 计算后，需要复位 SHA 加速器并且设置为有效值才能再次计算正确。

16.4.3 数据设定寄存器（SHA_MSGCFG)

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:27	RSV	-	-	保留
26:12	NUM_BYTES	R/W	0x0	设定当前段落的长度，如果 MSG_END 未使能，此域必须设定为运算块大小的倍数。 对于 SHA-256，运算块大小是 64Bytes; 对于 SHA-384，运算块大小是 128Bytes
11:2	RSV	-	-	保留
1	MSG_BEGIN	R/W	0x0	指示目前段落是整段信息的第一个段落
0	MSG_END	R/W	0x0	指示目前段落是整段信息的最后一个段落

16.4.4 数据长度寄存器（SHA_MSGTOTALBYTES)

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:26	RSV	-	-	保留
25:0	TOTAL_BYTES	R/W	0x0	整个信息的长度，仅在 MSG_END 使能时有效

16.4.5 数据输入寄存器（SHA_DATAIN)

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	DATA_IN	R/W	0x0	将需要运算的数据输入到这里

16.4.6 ICV 读取寄存器（SHA_ICVn)

偏移地址：0x20~0x4C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	ICV	R	0x0	当运算结果与 OTP ICV 不符时，可从这些寄存器读取结果。 对于 SHA-256，SHA_ICV 04~11 储存了运算结果； 对于 SHA-384，SHA_ICV 00~11 储存了运算结果； SHA_ICV 00 储存了 ICV[31:0]； SHA_ICV 01 储存了 ICV[63:32]； SHA_ICV 02 储存了 ICV[95:64]； SHA_ICV 03 储存了 ICV[127:96]； SHA_ICV 04 储存了 ICV[159:128]； SHA_ICV 05 储存了 ICV[191:160]； SHA_ICV 06 储存了 ICV[223:192]； SHA_ICV 07 储存了 ICV[255:224]； SHA_ICV 08 储存了 ICV[287:256]； SHA_ICV 09 储存了 ICV[319:288]； SHA_ICV 10 储存了 ICV[351:320]； SHA_ICV 11 储存了 ICV[383:352]。 注：SHA-256 的结果[255:0]存放在 SHA_ICV [128:383]

16.4.7 校验结果寄存器（SHA_VERIRESULT)

偏移地址：0x50

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	VERI_VLD	R	0x0	指示进行了 ICV 校验。 由 SHA_DONE+SHA_MSG_END 触发，通过使能 SHA_GO 清除。
0	VERI_FAIL	R	0x0	指示 ICV 是否符合 OTP ICV 值： 0：符合 1：不符合

16.4.8 中断使能寄存器（SHA_IRQEN)

偏移地址：0x54

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	IRQ_EN	R/W	0x0	选择使能运算完成中断： 0：不使能 1：使能

16.4.9 中断清除寄存器（SHA_IRQCLR)

偏移地址：0x58

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	IRQ_CLR	W	0x0	写 1 清除中断

16.4.10 中断状态寄存器（SHA_IRQSTATUS)

偏移地址：0x5C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	IRQ_STATUS	R	0x0	指示中断状态： 0：未发生中断 1：发生中断

16.4.11 DMA Burst 长度寄存器（SHA_DMABURSTSIZE)

偏移地址：0x60

复位值：0x0000 0003

位	名称	属性	复位值	描述
31:3	RSV	-	-	保留

位	名称	属性	复位值	描述
2:0	DMA_BURST_SIZE	R/W	0x3	设定 DMA burst 长度： 0x0：1 0x1：4 0x2：8 0x3：16 0x4：32 需要与 DMA 模块中设定的保持一致。比如 DMA burst 4，如果数据操作单位是 byte，就是 4 个 byte 操作，如果数据操作单位是 32bit 数据，就是 4 个 32bit 数据。

16.5 使用流程

16.5.1 SHA 运算应用(通过 DMA 输入报文)

1. 使能 SHA 时钟，释放 SHA 复位。
2. 使能 DMA1 时钟，释放 DMA1 复位。
3. 初始化 DMA 及通道配置，例如 DMA1 通道 2，memory 到 peripheral，源地址自动增 1，目的地址不变。源、目的 DMA burst 长度均为 1。源数据传输宽度为 32bit，目地数据传输宽度为 32bit。源、目的均为硬件同步。源握手信号为任意值(memory)，目的握手信号为 signal7。
4. 使能 DMA 模块。
5. 配置 SHA_MODE[1:0]，选择 SHA-256 或 SHA-384 模式。
6. 配置 SHA_MSGCFG[1]和 SHA_MSGCFG[0]，报文长度小于 64byte(SHA-256)或 128byte(SHA-384)为单 segment，BEGIN 和 END 均设置为 1。长度超过单 segment，根据首 segment，中间 segment，末 segment 分别设置。
7. 配置 SHA_MSGCFG[26:12]，为当前 segment 的长度。
8. 配置 SHA_MSGTOTALBYTES 寄存器，为整段报文的长度。
9. 获取待加密明文。例如：可将待加密明文存储到 sram 的合法地址空间。
10. 使能 SHA_CMD 寄存器，SHA_GO = 1 启动一次 SHA 计算。
11. 启动 DMA1 传输数据到 SHA，例如：DMA1 通道 2，目的地址 SHA_DATA_IN(地址不自动增加)，源地址为明文首地址(地址自动增加)，配置传输数据长度(根据 DMA1 源数据宽度配置，若源数据宽度为 32bit，则当前 segment 的长度应除 32)，DMA 配置见 DMA 章节或示例代码。
12. 等待 SHA_DONE 置位。
13. 检查加密结果寄存器 SHA_ICVn，SHA-256 对应 SHA_ICV04-11 储存运算结果，SHA-384 对

应 SHA_ICV00-11 储存运算结果。

14. 事先将加密结果储存在 OTP SHA ICV 中(详见 EFC 章节介绍或示例代码)，可检查校验结果寄存器 SHA_VERI_RESULT 是否符合预期。
15. 如有多 segment 信息段落，重复 6~12 步。

16.5.2 SHA 运算应用(通过 CPU 输入报文)

1. 使能 SHA 时钟，释放 SHA 复位。
2. 配置 SHA_MODE[1:0]，选择 SHA-256 或 SHA-384 模式。
3. 配置 SHA_MSGCFG[1]和 SHA_MSGCFG[0]，报文长度小于 64byte(SHA-256)或 128byte(SHA-384)为单 segment，BEGIN 和 END 均设置为 1。长度超过单 segment，根据首 segment，中间 segment，末 segment 分别设置。
4. 配置 SHA_MSGCFG[26:12]，为当前 segment 的长度。
5. 配置 SHA_MSGTOTALBYTES 寄存器，为整段报文的长度。
6. 获取待加密明文。例如：可将待加密明文存储到 sram 的合法地址空间。
7. 使能 SHA_CMD 寄存器，SHA_GO = 1 启动一次 SHA 计算。
8. 通过 CPU 将明文按照小端模式拼接为 32bit 字循环放入到 SHA_DATAIN 寄存器。
9. 等待 SHA_DONE 置位。
10. 检查加密结果寄存器 SHA_ICVn，SHA-256 对应 SHA_ICV04-11 储存运算结果，SHA-384 对应 SHA_ICV00-11 储存运算结果。
11. 事先将加密结果储存在 OTP SHA ICV 中(详见 EFC 章节介绍或示例代码)，可检查校验结果寄存器 SHA_VERIRESULT 是否符合预期。
12. 如有多 segment 信息段落，重复 3~9 步。

17 随机数发生器（RNG）

17.1 概述

RNG 控制器可以产生随机数。

17.2 主要特性

可以利用随机数种子产生随机数

17.3 功能描述

17.3.1 随机数发生器

可以利用随机数种子产生随机数。

17.4 寄存器描述

寄存器基地址：0x40B0_D000

寄存器列表如下：

表 17-1：RNG 寄存器列表

偏移地址	名称	描述
0x00	RNG_DATA	数据寄存器
0x04	RNG_SEED	种子寄存器
0x08	RNG_CR	控制寄存器

17.4.1 数据寄存器 (RNG_DATA)

偏移地址：0x00

复位值：0xXXXX XXXX（读出的数据是随机的）

位	名称	属性	复位值	描述
31:0	DATA	R/W	0xXXXX XXXX	读取数据获取随机数

17.4.2 种子寄存器 (RNG_SEED)

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	SEED	W	0x0	保留

17.4.3 控制寄存器 (RNG_CR)

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	EN	R/W	0x0	开关随机数种子随时间自动变化： 1：开启 0：关闭

17.5 使用流程

1. 配置 RNG_CR 寄存器，开启随机数发生器。
2. 配置 RNG_SEED 寄存器，写入一个种子。
3. 从 RNG_DATA 寄存器读取随机数，可连续读取。

18 高级控制定时器（TIM0 & TIM7）

18.1 概述

包含一个 32bit 自动重载计数器及一个可编程预分频器。

可以支持多种应用，包括如捕获、输出比较、带死区插入的互补 PWM。

18.2 主要特性

- 32bit 向上、向下、双向自动重载计数器
- 16bit 可编程预分频器，支持实时调整计数时钟分频
- 4 个独立通道可用于输入捕获、输出比较、PWM、单脉冲输出
- 可编程死区插入的互补输出
- 支持与其他定时器级联
- 重复计数器，支持定时器多个循环后更新状态
- 刹车引脚输入，刹车信号滤波和极性选择，刹车信号组合配置
- 支持在以下事件发生时产生中断或 DMA 事件
  - 计数器上/下溢出，计数器初始化（软件或硬件 trigger）
  - Trigger 事件（计数器启动、停止、初始化、内外部触发）
  - 输入捕获
  - 输出比较
  - 刹车输入
- 支持增量正交编码器和霍尔传感器
- 支持外部时钟和触发输入

18.3 系统框图

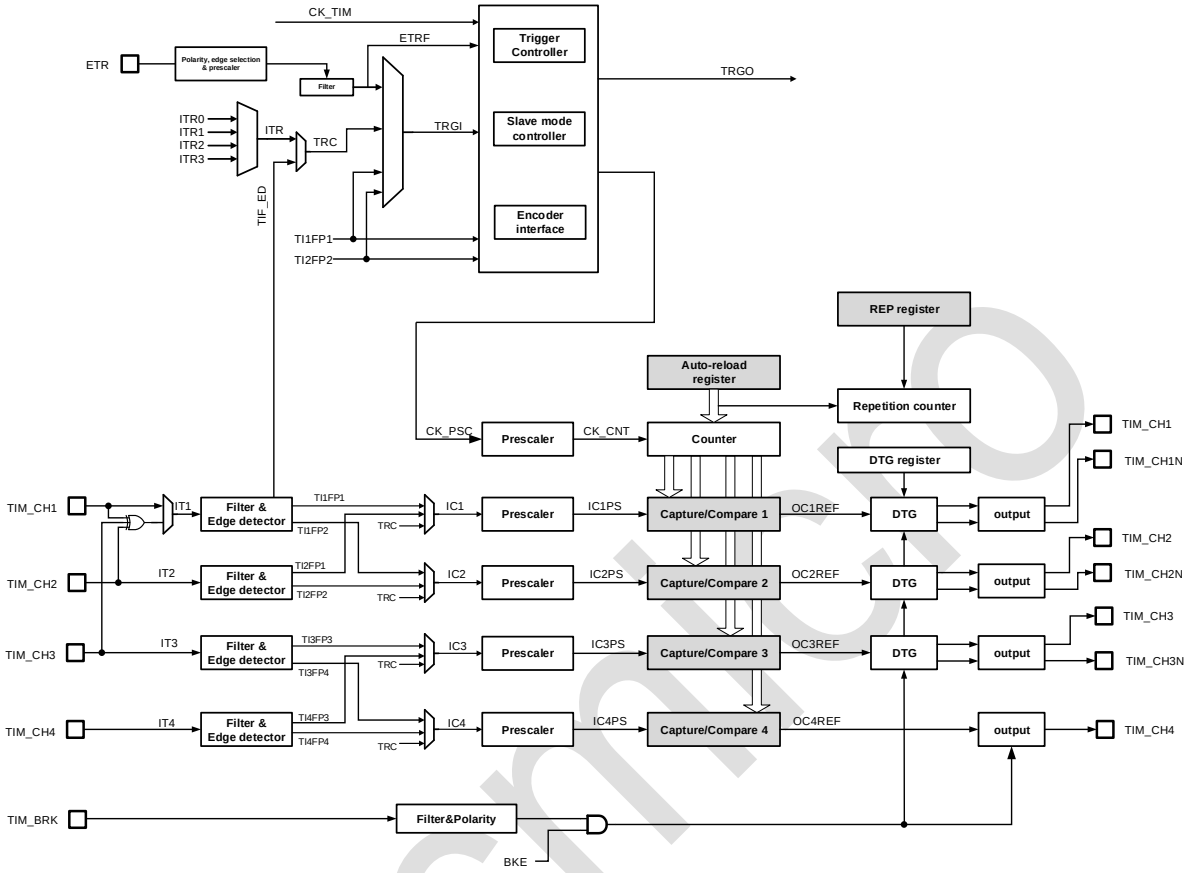


图 18-1：TIM0&TIM7 系统框图

18.4 管脚说明

表 18-1：TIM0&TIM7 管脚说明

功能管脚	复用管脚	方向	功能描述
TIM0_BKIN	PA6,PB12,PE15	Input	刹车输入
TIM0_ETR	PA12,PE7	Input	外部触发输入
TIM0_CH1	PA8,PE9	Input/Output	通道输入捕获或者PWM输出
TIM0_CH1N	PA7,PB13,PE8	Output	PWM输出反向
TIM0_CH2	PA9,PE11	Input/Output	通道输入捕获或者PWM输出
TIM0_CH2N	PB0,PB14,PE10	Output	PWM输出反向
TIM0_CH3	PA10,PA4,PE13	Input/Output	通道输入捕获或者PWM输出
TIM0_CH3N	PB1,PB15,PE12	Output	PWM输出反向
TIM0_CH4	PA11,PA5,PE14	Input/Output	通道输入捕获或者PWM输出
TIM7_BKIN	PA6	Input	刹车输入
TIM7_ETR	PA0	Input	外部触发输入
TIM7_CH1	PC6	Input/Output	通道输入捕获或者PWM输出

功能管脚	复用管脚	方向	功能描述
TIM7_CH1N	PA5,PA7	Output	PWM输出反向
TIM7_CH2	PC7	Input/Output	通道输入捕获或者PWM输出
TIM7_CH2N	PB0,PB14	Output	PWM输出反向
TIM7_CH3	PC8	Input/Output	通道输入捕获或者PWM输出
TIM7_CH3N	PB1,PB15	Output	PWM输出反向
TIM7_CH4	PC9	Input/Output	通道输入捕获或者PWM输出

18.5 定时器内部互联

表 18-2：定时器内部互联

TIMx Interconnect								
Source (TIMx Trigger Output)	Inputs (TIMx Trigger Input)							
TIM0_TRGO	-	TIM1_ITR0	TIM2_ITR0	TIM3_ITR0	-	TIM7_ITR0	-	-
TIM1_TRGO	TIM0_ITR1	-	TIM2_ITR1	TIM3_ITR1	TIM4_ITR0	TIM7_ITR1	TIM8_ITR0	-
TIM2_TRGO	TIM0_ITR2	TIM1_ITR2	-	TIM3_ITR2	TIM4_ITR1	-	TIM8_ITR1	-
TIM3_TRGO	TIM0_ITR3	TIM1_ITR3	TIM2_ITR3	-	TIM4_ITR2	TIM7_ITR2	-	TIM11_ITR0
TIM4_TRGO	TIM0_ITR0	-	TIM2_ITR2	-	-	TIM7_ITR3	-	TIM11_ITR1
TIM7_TRGO	-	TIM1_ITR1	-	TIM3_ITR3	TIM4_ITR3	-	-	-
TIM8_TRGO	-	-	-	-	-	-	-	-
TIM9_TRGO	-	-	-	-	-	-	TIM8_ITR2	-
TIM10_TRGO	-	-	-	-	-	-	TIM8_ITR3	-
TIM11_TRGO	-	-	-	-	-	-	-	-
TIM12_TRGO	-	-	-	-	-	-	-	TIM11_ITR2
TIM13_TRGO	-	-	-	-	-	-	-	TIM11_ITR3

18.6 功能描述

18.6.1 定时单元

定时单元由一个 32 位计数器和自动重载寄存器组成。计数器可以向上、向下或双向计数。计数时钟可以通过 16 位预分频器对时钟进行分频后得到。

计数器、自动重载寄存器预分频寄存器都可以由软件改写或读取，即使在计数器正在运行时也是如此。

定时单元包含如下寄存器：

- 计数器（TIM_CNT）
- 预分频寄存器（TIM_PSC）
- 自动重载寄存器（TIM_ARR）
- 重复计数寄存器（TIM_RCR）

ARR 包含预装载功能，该功能通过 ARPE（Auto Reload Preload Enable）寄存器控制。当 ARPE=0 时，对 ARR 寄存器执行写入，写入数据将直接传入到影子寄存器；当 ARPE=1 时，对 TIM_ARR 寄存器执行写入的数据在 update event（TIM_CNT 上溢出或者下溢出）发生时，传送到影子寄存器。软件也可以通过寄存器操作主动触发 ARR 更新（UEV）。

TIM_CNT 工作时钟由 TIM_PSC 产生的分频时钟驱动，只有在计数器使能寄存器（CEN）置位时，CNT 才开始计数。当 CNT=ARR 时，本轮计数结束，发送 update event。

TIM_PSC 是一个同步预分频器，能够对时钟进行 1~65536 分频。PSC 寄存器同样被缓存，改写 PSC 实际不改写影子寄存器，只有当新的 update event 到来时，才会从 PSC 更新至影子寄存器。因此在 CNT 计数过程中，软件可以实时改写 PSC，而新的预分频比将在下一更新事件发生时被采用。

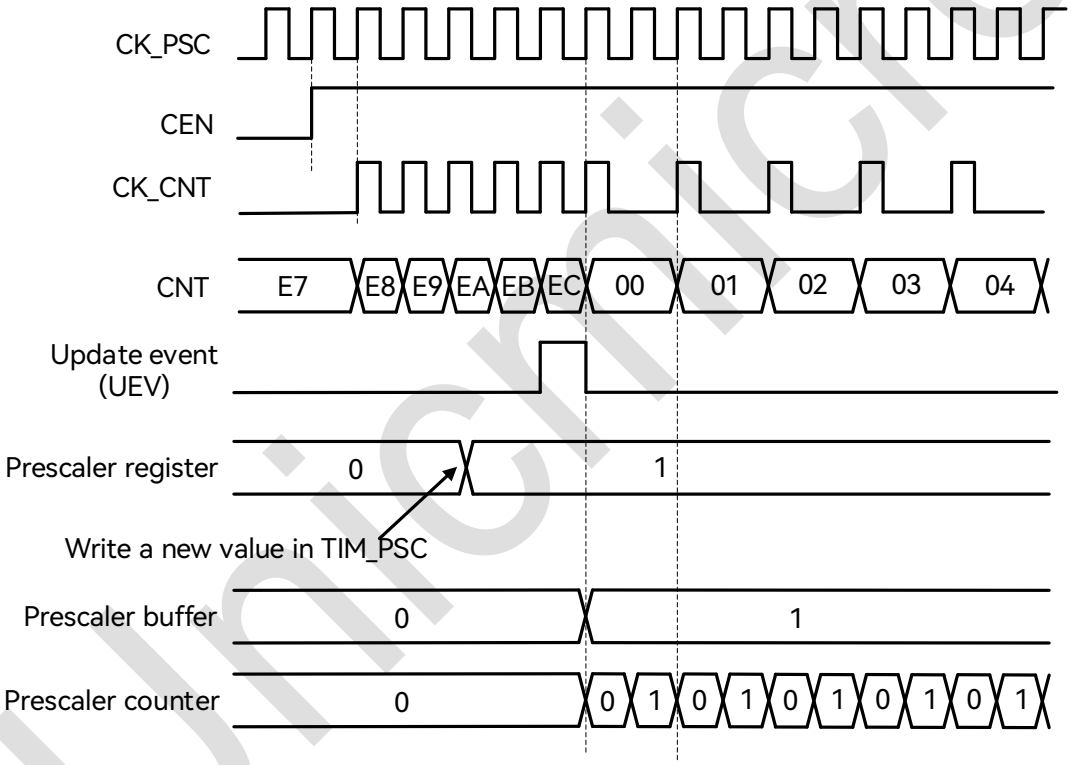


图 18-2：预分频从 1 变为 2 的波形图

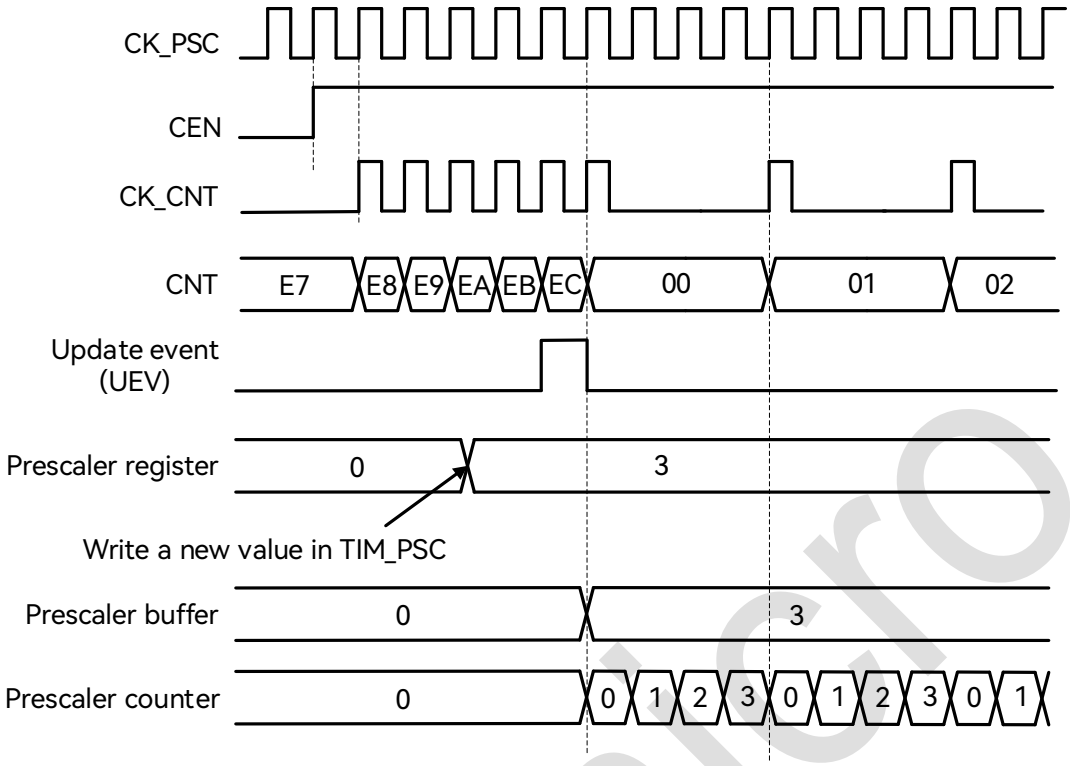


图 18-3：预分频从 1 变为 4 的波形图

18.6.2 定时器工作模式

定时器支持向上计数、向下计数和中心计数模式。

18.6.2.1 向上计数

此模式中，计数器使能后从 0 开始计数，直到 CNT=ARR，产生溢出事件，然后重新从 0 开始计数。

如果使能了重复计数功能，则计数器按照 RCR 的定义重复上述过程若干次（RCR+1），才会产生溢出事件。

软件可以通过设置 UG 寄存器直接触发 update event，此时 CNT 和预分频计数器自动清零。设置 UG 寄存器是否触发 UIF（Update Interrupt Flag）中断标志置位由 URS 寄存器的设置决定。

通过设置 UDIS 寄存器可以禁止 update event，这样可以避免将 preload 寄存器中的值更新到工作寄存器中。

当 update event 发生时，以下寄存器被更新，并且 UIF 置位：

- RCR影子寄存器被更新为TIM_RCR内容
- ARR影子寄存器被更新为TIM_ARR内容
- PSC影子寄存器被更新为TIM_PSC内容

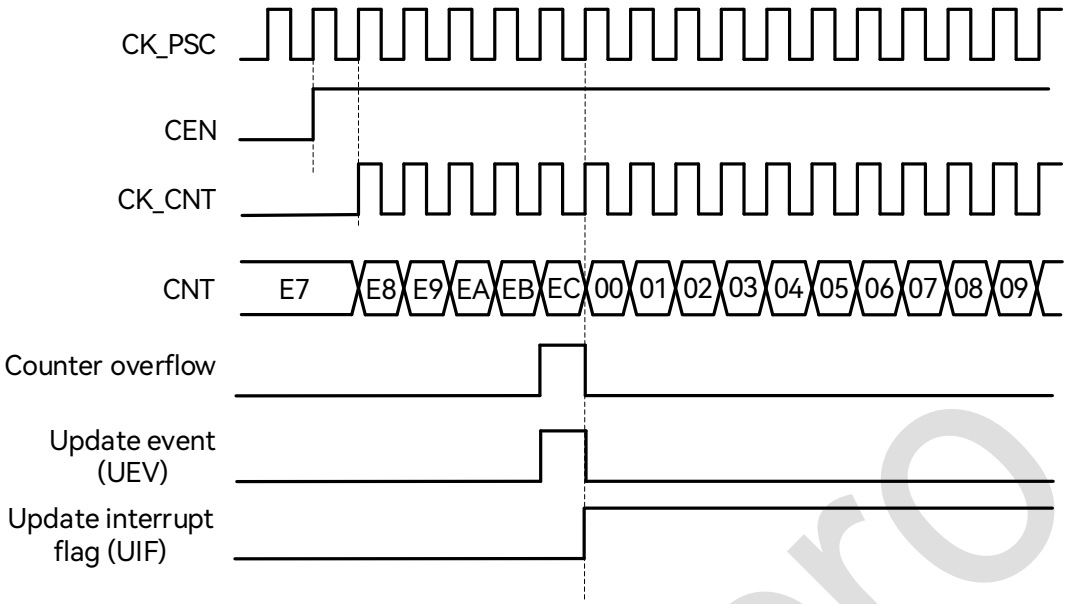


图 18-4：向上计数波形，内部时钟不分频图

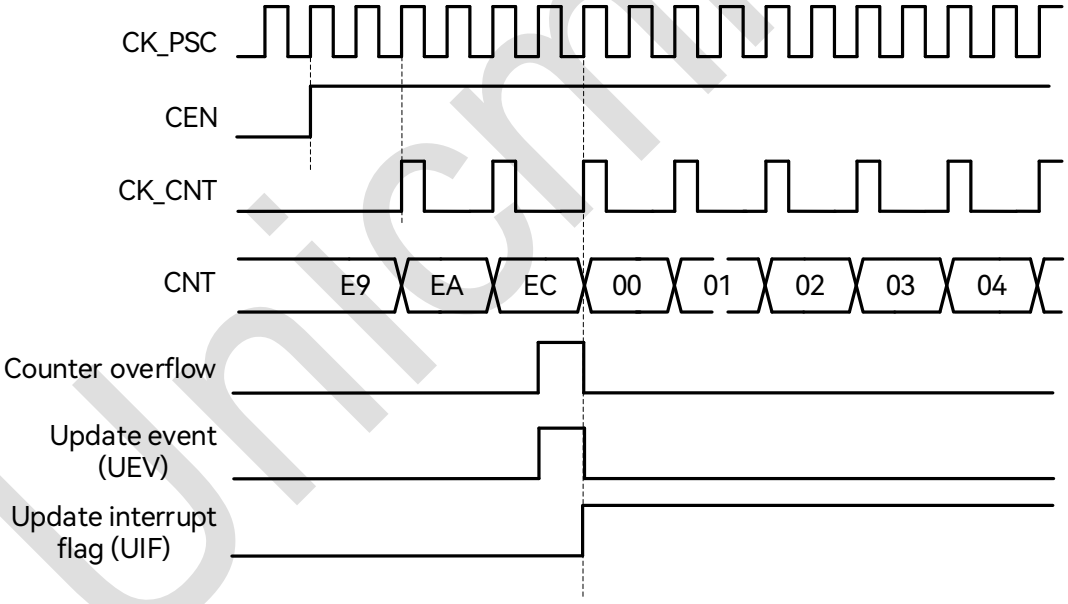


图 18-5：向上计数波形，内部时钟 2 分频图

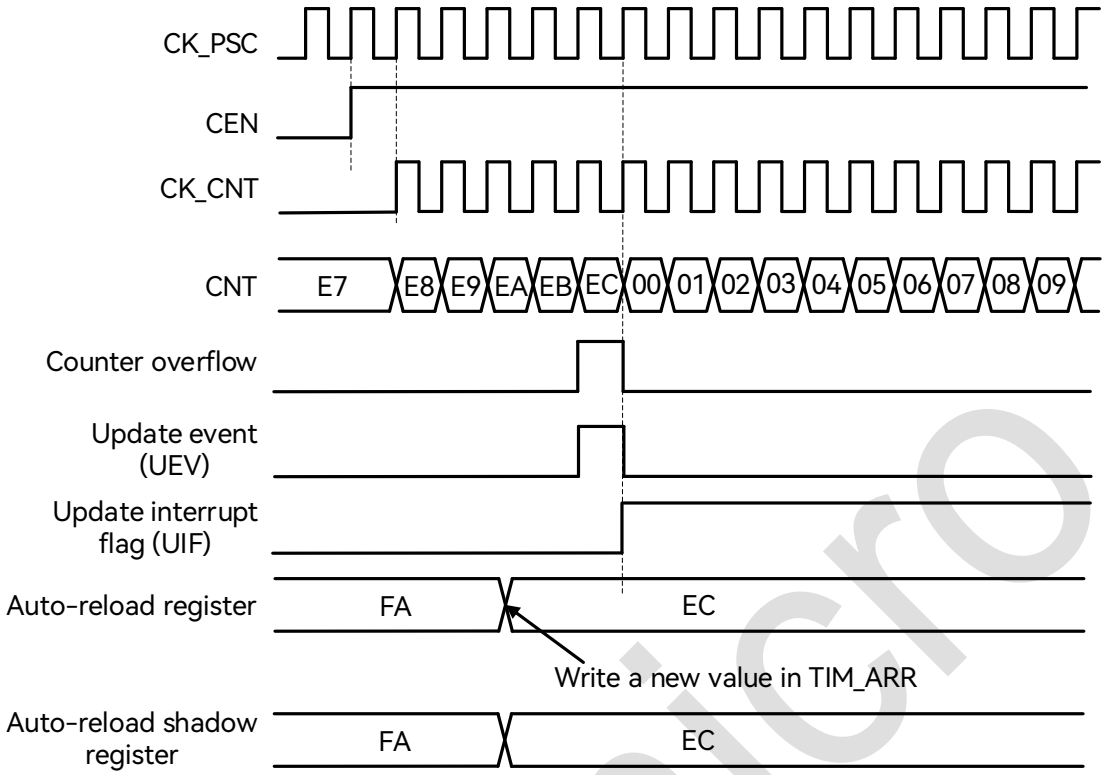


图 18-6：ARPE=0（TIM_ARR 没有预装载）时的更新事件图

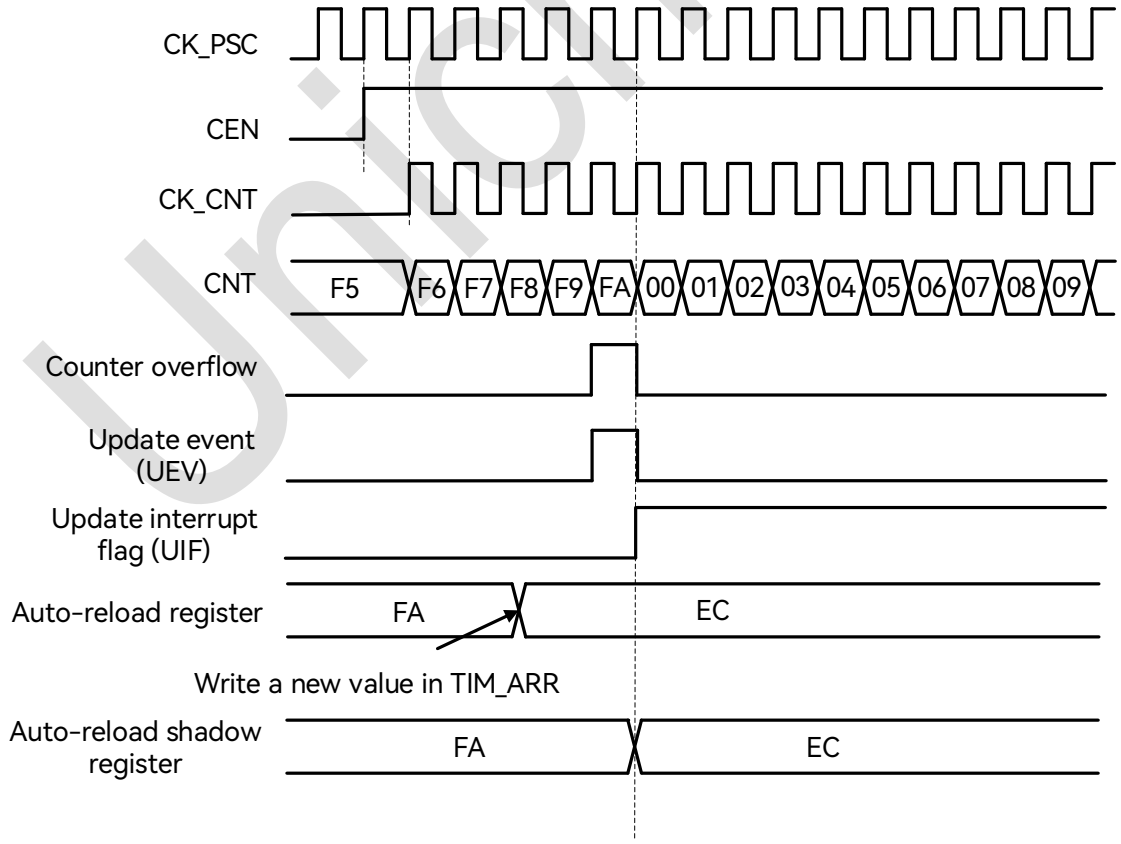


图 18-7：ARPE=1（TIM_ARR 预装载）时的更新事件图

18.6.2.2 向下计数

向下计数模式中，计数器从 ARR 值开始递减，到 0 后产生下溢出事件，并且重新从 ARR 开始计数。

如果使能了重复计数功能，则计数器按照 RCR 的定义重复上述过程若干次（RCR+1），才会产生溢出事件。

软件可以通过设置 UG 寄存器直接触发 update event，此时 CNT 和预分频计数器自动清零。设置 UG 寄存器是否触发 UIF（Update Interrupt Flag）中断标志置位由 URS 寄存器的设置决定。

通过设置 UDIS 寄存器可以禁止 update event，这样可以避免将 preload 寄存器中的值更新到工作寄存器中。

当 update event 发生时，以下寄存器被更新，并且 UIF 置位：

- RCR影子寄存器被更新为TIM_RCR内容
- ARR影子寄存器被更新为TIM_ARR内容
- PSC影子寄存器被更新为TIM_PSC内容

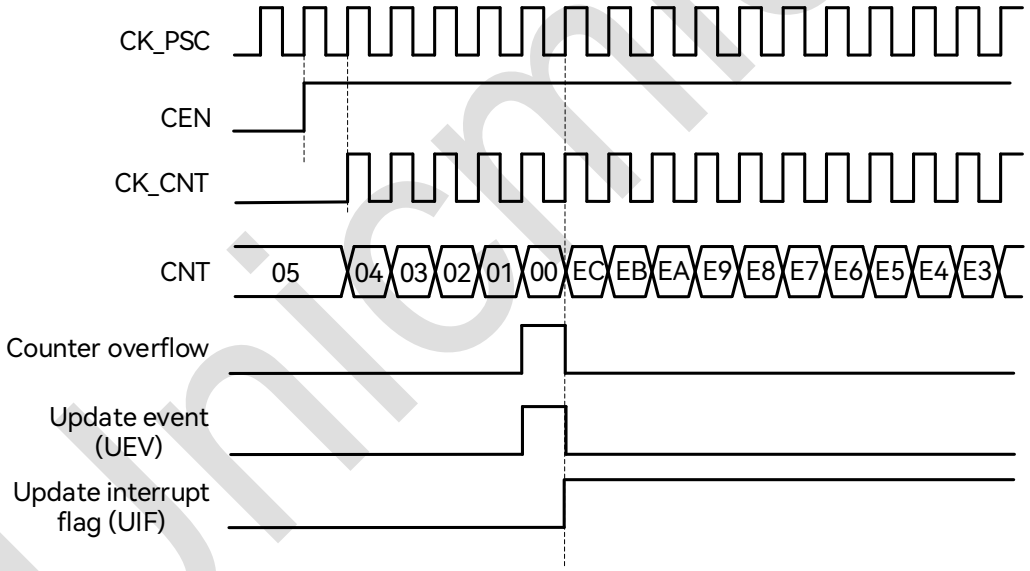


图 18-8：向下计数，内部时钟不分频图

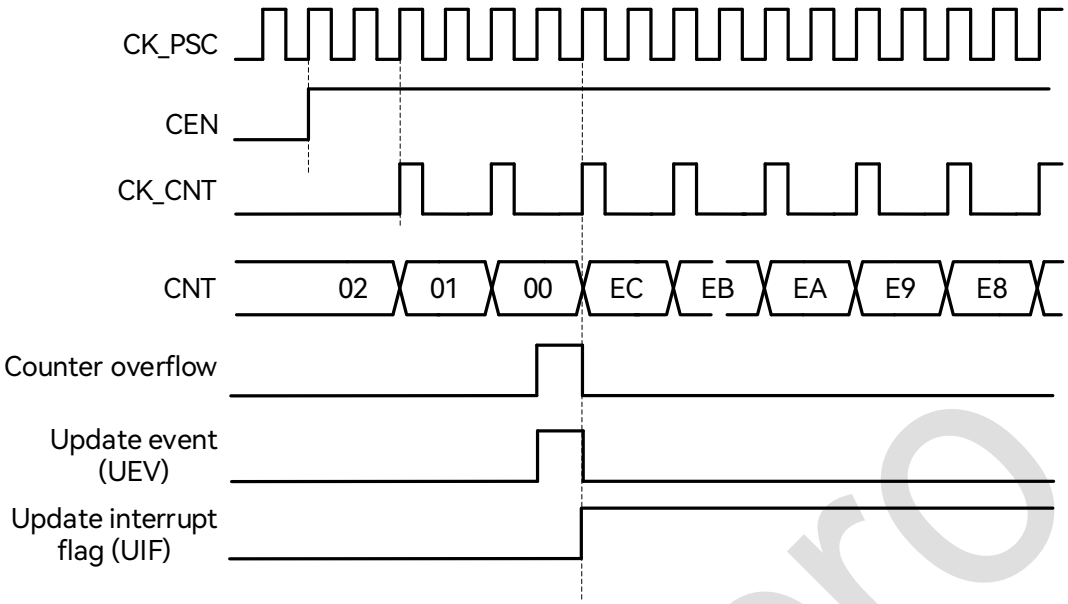


图 18-9：向下计数，内部时钟 2 分频图

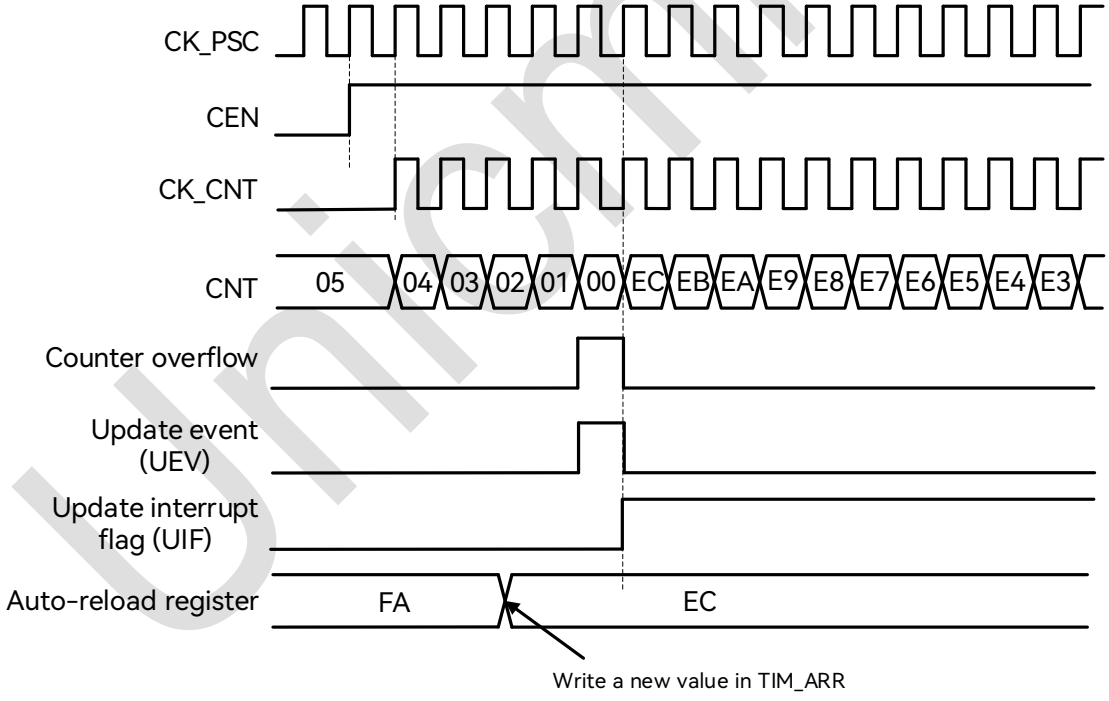


图 18-10：向下计数，内部时钟 2 分频图

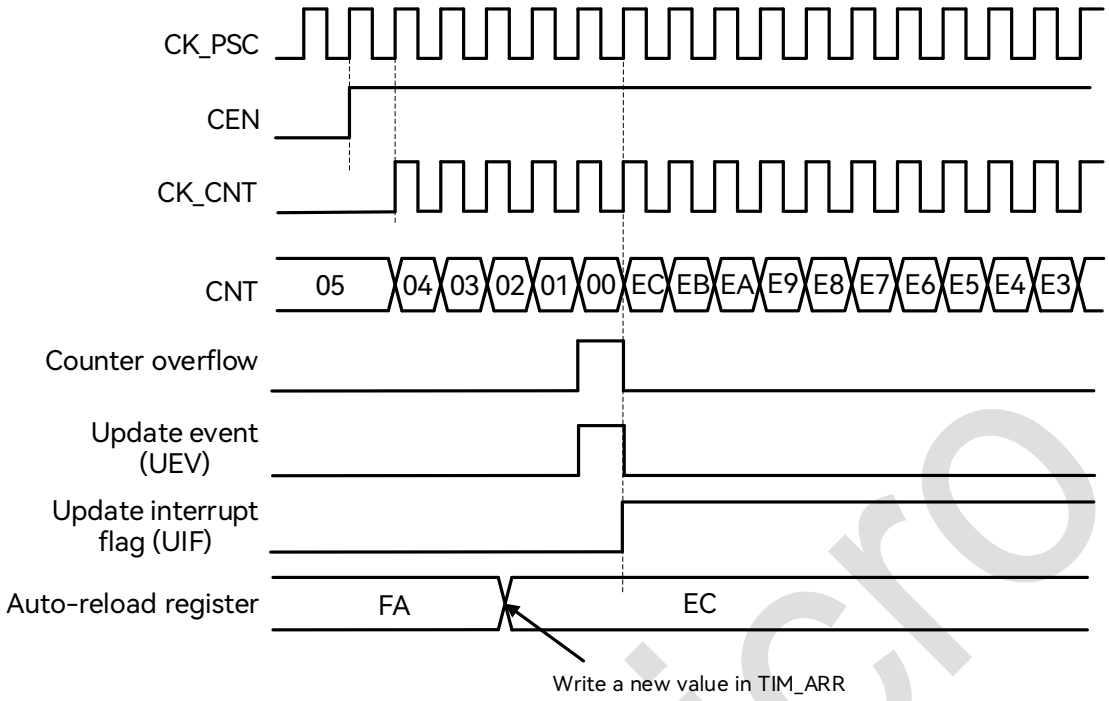


图 18-11：向下计数，不使用重复计数时的更新事件图

18.6.2.3 中心对齐计数

在中心对齐模式下，计数器从 0 开始向上计数，到 ARR-1 产生上溢出事件，然后从 ARR 开始向下计数到 1，产生下溢出事件，再从 0 重新开始向上计数。

CMS[1:0]用于使能中心对齐模式，并选择中心对齐模式下的输出比较工作方式。当 CMS!=00 时为中心对齐计数，当 CMS=01 时，输出比较功能仅在向下计数时有效，当 CMS=10 时，输出比较功能仅在向上计数时有效，当 CMS=11 时，输出比较功能在上下计数时都有效。

中心对齐模式下，DIR 寄存器无法由软件改写，而是随着计数方向变化硬件自动更新，表示当前计数方向。

计数器在 overflow 和 underflow 的事件上都会更新 ARR、PSC 和 RCR 的影子寄存器。

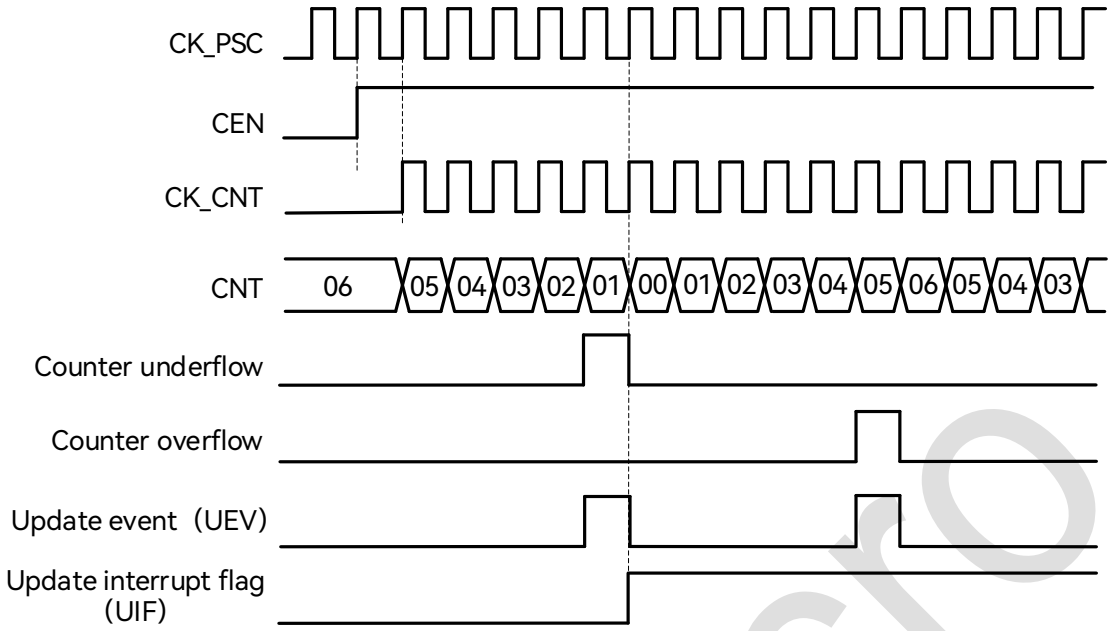


图 18-12：中心对齐计数器时序图，TIM_PCS=0，TIM_ARR=0x6

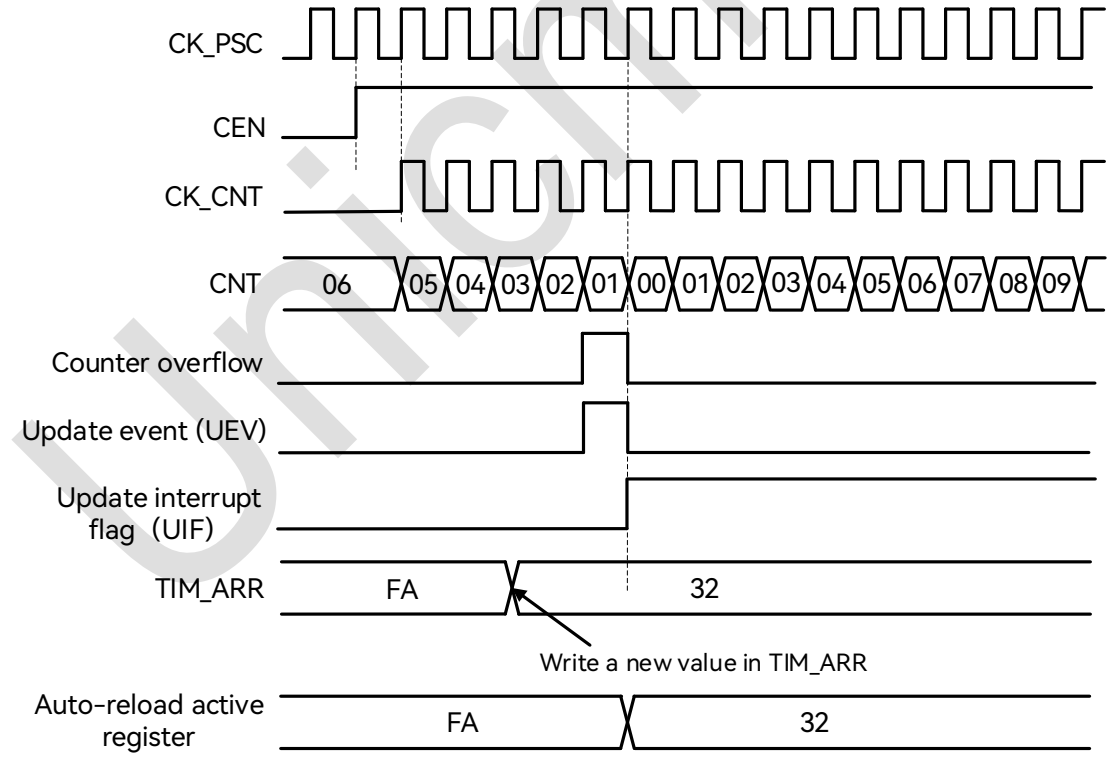


图 18-13：计数器时序图，ARPE=1 时的更新事件（计数器下溢）

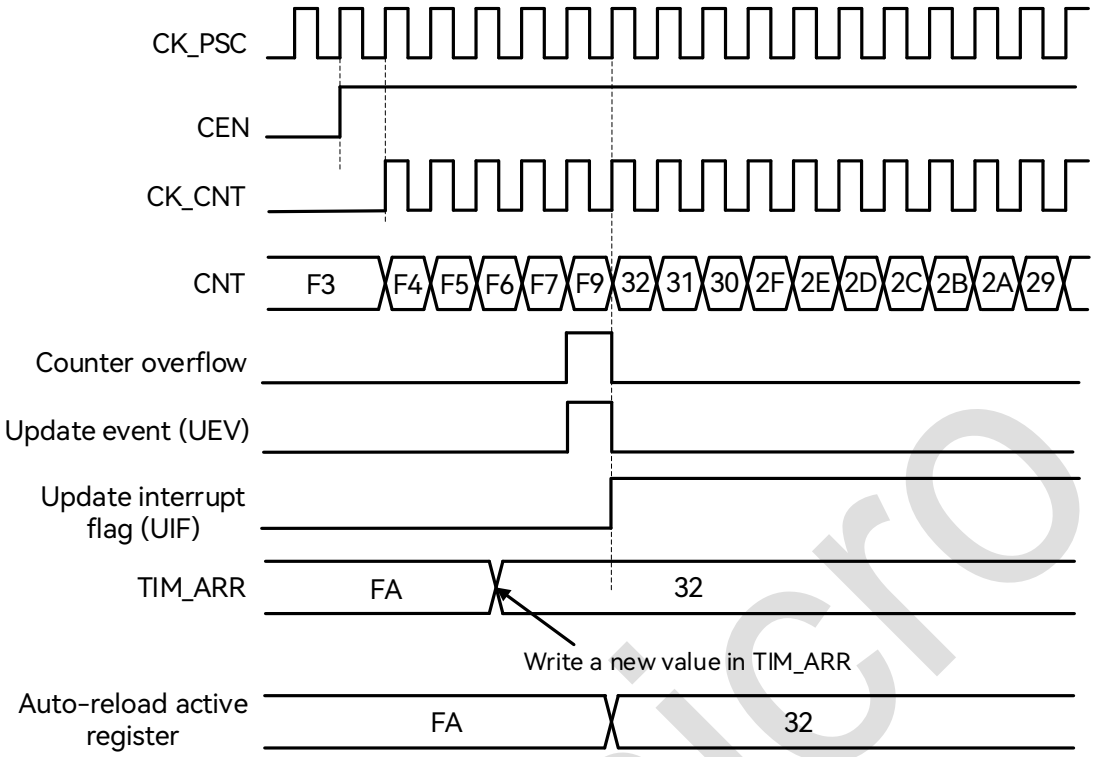


图 18-14：计数器时序图，ARPE=1 时的更新事件（计数器溢出）

18.6.3 重复计数器

Update event 在计数器 overflow 或 underflow，并且重复计数器为 0 的情况下产生。这意味着 ARR、PSC、CCR（比较/捕获寄存器，输出比较模式下）的 preload 寄存器会在 N+1 次 overflow 或 underflow 之后，才将数据传输给影子寄存器，其中 N 是 RCR 寄存器值。

重复计数器在以下情况下递减：

- 向上计数模式下发生上溢出
- 向下计数模式下发生下溢出
- 中心计数模式下每次上溢出或者下溢出

注意，当 update event 由软件或 slave mode controller 触发时，更新事件会立即发生，而不管当前 RCR 是什么值，同时重复计数器也会被立即更新为 RCR 的值。

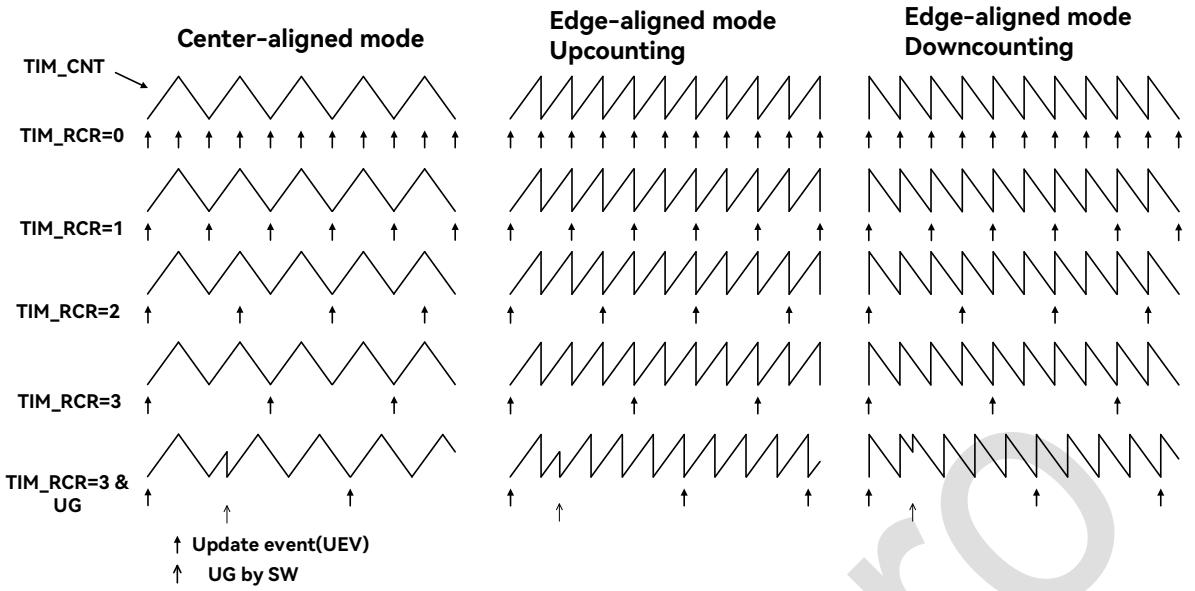


图 18-15：不同模式下更新速率的例子，及 TIM_RCR 的寄存器设置图

18.6.4 Preload 寄存器

以下功能寄存器支持 preload 功能：

- 自动重载寄存器 TIM_ARR
- 预分频寄存器 TIM_PSC（不可关闭 preload 功能）
- 通道控制寄存器 TIM_CCR
- CCxE 和 CCxNE 控制寄存器
- OCxM 控制寄存器

以上寄存器，除了 PSC 之外，都可以由软件选择使能或者禁止 preload 功能。

具备 preload 功能的寄存器，包含两组物理实体：

- Shadow register（影子寄存器）：实际定时器正在使用的寄存器
- Preload register（预装载寄存器）：软件可以访问的寄存器

当禁止 preload 时，具备 preload 功能的寄存器特性如下：

- Preload 寄存器可以实时由软件访问、改写
- Shadow 寄存器与 Preload 寄存器同步更新

如果使能了 preload，则：

- 所有软件操作访问的是 preload 寄存器
- 当 update event 发生时，所有 preload 寄存器内容将同步被转移到对应的 shadow 寄存器

18.6.5 计数器工作时钟

计数器可以使用如下时钟工作：

- Timerx_clk——内部时钟模式
- 外部引脚输入时钟（TIx）——外部时钟模式 1
- 外部引脚触发输入（ETR）——外部时钟模式 2
- 内部触发（ITRx）——使用一个 timer 的触发输出（TRGO）作为计数时钟

18.6.5.1 内部时钟模式

内部时钟模式下，禁止从机模式（SMS=000），CEN、DIR、UG 等寄存器位都是软件控制。软件操作 UG 寄存器后，update 信号经过 CLK_PSC 同步后，计数器值将被重新初始化。

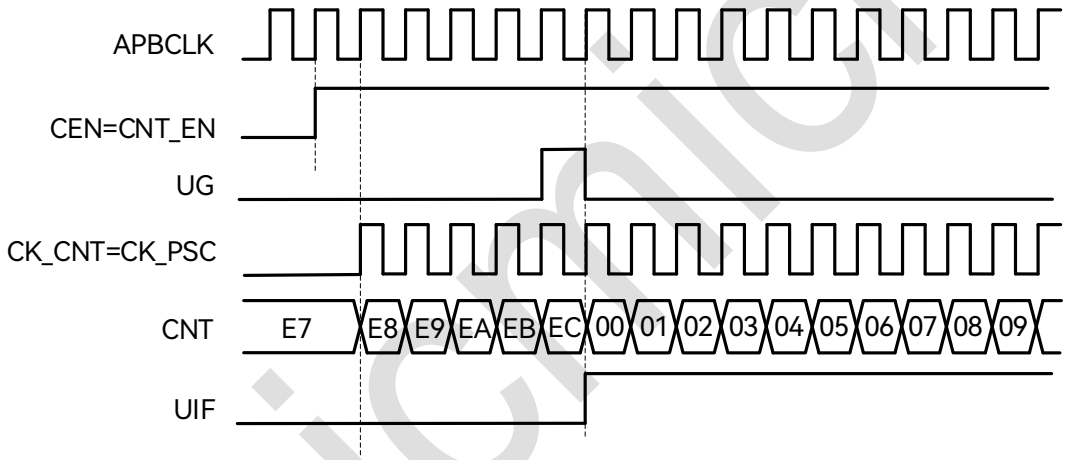


图 18-16：内部时钟源模式，时钟分频因子为 1 时序图

18.6.5.2 外部时钟模式 1

此模式下直接使用外部引脚输入信号作为计数时钟，配置 SMS=111，计数边沿可以配置为上升或下降沿。

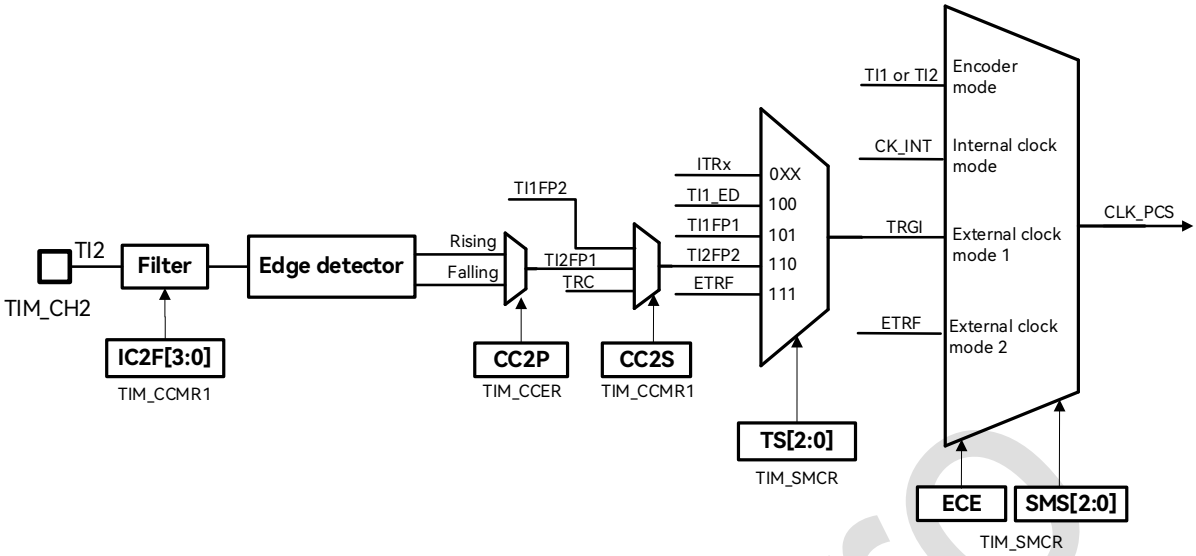


图 18-17：外部时钟连接图

外部输入信号在触发计数器计数前，会先经过内部时钟的同步过程，同时输入信号的有效沿会触发 TIF 标志。

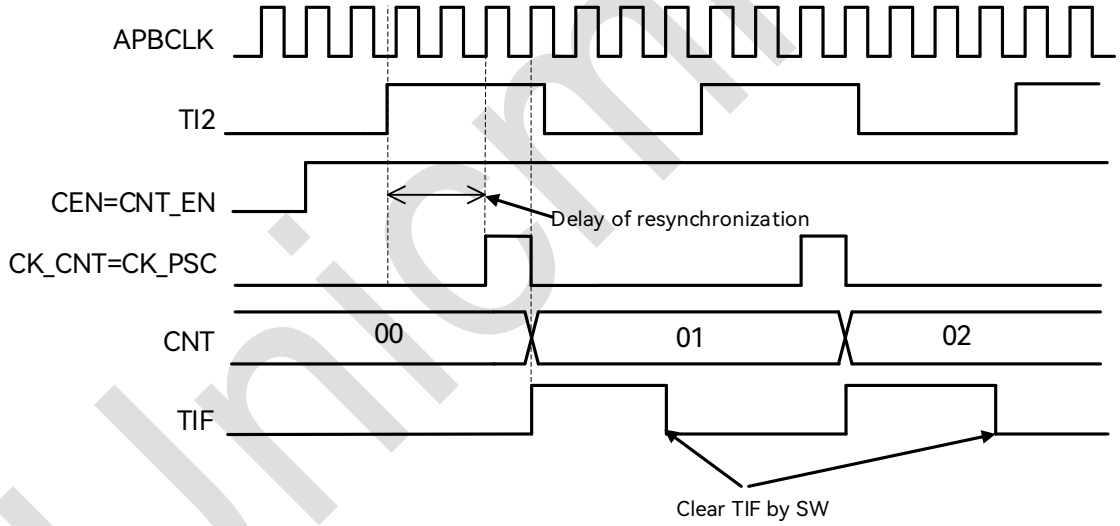


图 18-18：外部时钟模式 1 下的时序图

使用外部时钟计数时，仍然要使能 TIM 的内部时钟（timerx_clk），因为 TIM 要使用 timerx_clk 来对外部输入时钟进行同步和滤波。在外部时钟模式 1 下，外部输入时钟首先经过滤波和边沿选择，得到有效的计数沿，作为有效工作时钟（CLK_PSC）输入给预分频模块。

外部时钟同步采用简单的 2 级触发器结构，因此为了避免亚稳态，要求外部输入时钟宽度至少大于 2 个 timerx_clk 周期。

此模式下只有通道 1 和 2 的输入可以用做时钟输入，所需配置如下：

- 在GPIO模块中，配置相应管脚为TIM_CH2功能
- 关闭通道使能，配置TIM_CCER[4]=0，确保之后通道配置成功

- 选择输入通道，配置TIM_CCMR1[9:8]=01, IC2映射到TI2
- 选择计数有效沿，配置TIM_CCER[5]=0，选择上沿或者下沿
- 配置输入滤波时间，配置TIM_CCMR1.IC2F[3:0]（IC2F=0000，不进行输入滤波）
- 使能外部时钟模式1，配置TIM_SMCR[2:0]=111
- 选择触发输入源，配置TIM_SMCR[6:4]=110，选定TI2作为触发输入源
- 打开通道使能，配置TIM_CCER[4]=1
- 使能计数器，配置TIM_CR1[0]=1

下图是一个典型的外部时钟计数模式 1 的示例：

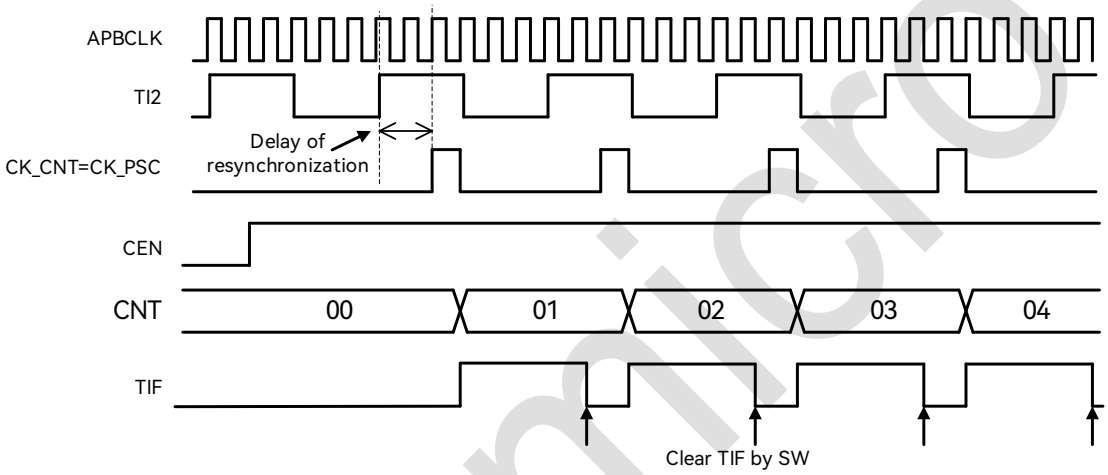


图 18-19：外部时钟模式 1 下的时序图

18.6.5.3 外部时钟模式 2

此模式下使用 TIM_ETR 管脚输入信号的上升沿或下降沿（不支持双沿）来计数。

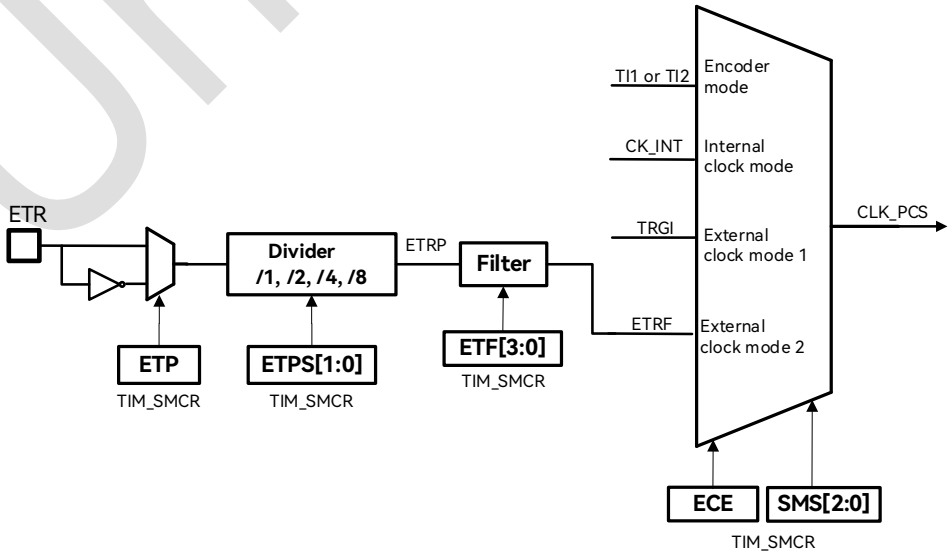


图 18-20：外部触发输入框图

下图是使用 ETR 二分频后的上升沿进行计数，其中实际计数发生时间因为内部时钟的同步过程而延迟于 ETR 输入上升沿。

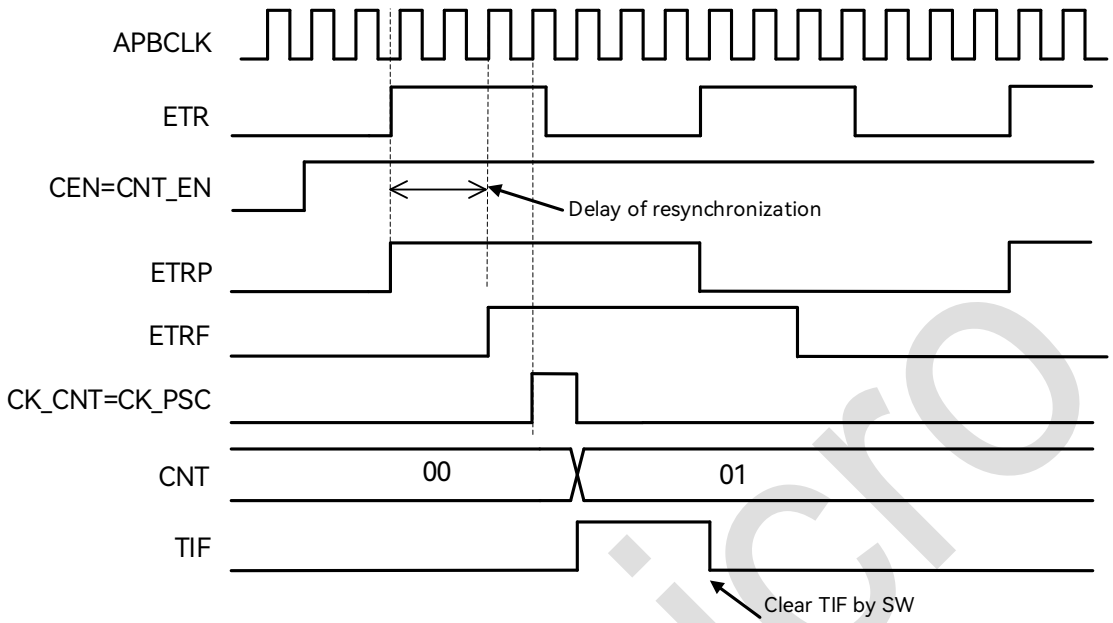


图 18-21：外部时钟模式 2 下的时序 1 图

与外部时钟模式 1 的主要差别是，ETR 输入直接被分频后再进行滤波，产生 CK_PSC 时钟，这意味着可以支持 ETR 输入频率高于 timerx_clk 的应用场景，这种情况下，需要首先对 ETR 输入进行预分频，再用于驱动计数器。

此模式所需配置如下：

- 在 GPIO 模块中，配置相应管脚为 TIM_ETR 功能
- 设置 ETP 进行沿选择，TIM_SMCR[15]=0
- 设置 ETR 分频比，配置 TIM_SMCR.ETPS[1:0]=01
- 配置输入滤波时间，TIM_SMCR.ETF[3:0]=0000
- 置位 ECE 寄存器，使能外部时钟模式 2,TIM_SMCR[14]=1，TIM_SMCR[2:0]=000
- 使能计数器，配置 TIM_CR1[0]=1

下图是一个典型的外部时钟模式 2 的示例：

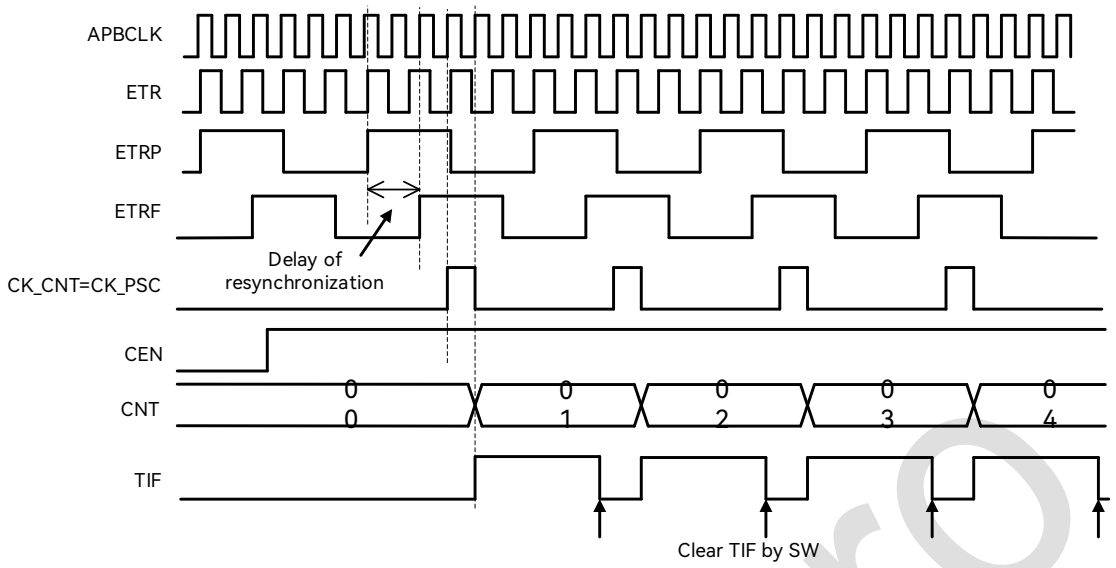


图 18-22：外部时钟模式 2 下的时序 2 图

在使用外部时钟模式 2 时，仍可以将 TIM 配置为 slave 模式：比如使用 ETR 输入计数，同时使用另一个 Timer 的 TRGO 作为触发信号，当触发事件到来时，复位计数器重新开始计数。

18.6.6 内部触发信号（ITRx）

TIM 支持 4 个 ITR 输入，可用于计数触发或者内部信号捕获。当用于内部信号捕获时，需要将 TS 配置为 000~011 用于选择 ITR0~ITR3，并将 CCxS 配置为 11，即将 TRC（4 个 ITRx 输入选择后的信号）选为捕获信号。

每个 ITR 输入支持 4 个内部信号扩展，由 ITRxSEL 寄存器配置。

18.6.7 捕获/比较通道

TIM 包含 4 个捕获/比较通道，每个通道由一个捕获比较寄存器（TIM_CCR）（包含影子寄存器）、一个捕获输入级、一个比较输出级组成。

输入级电路会采样 TIx 输入并产生滤波后的信号 TIxF，然后边沿检测和极性选择产生对应的 TIxFPx 信号，此信号可作为计数触发或者待捕获信号，并且在被捕获前经过预分频。

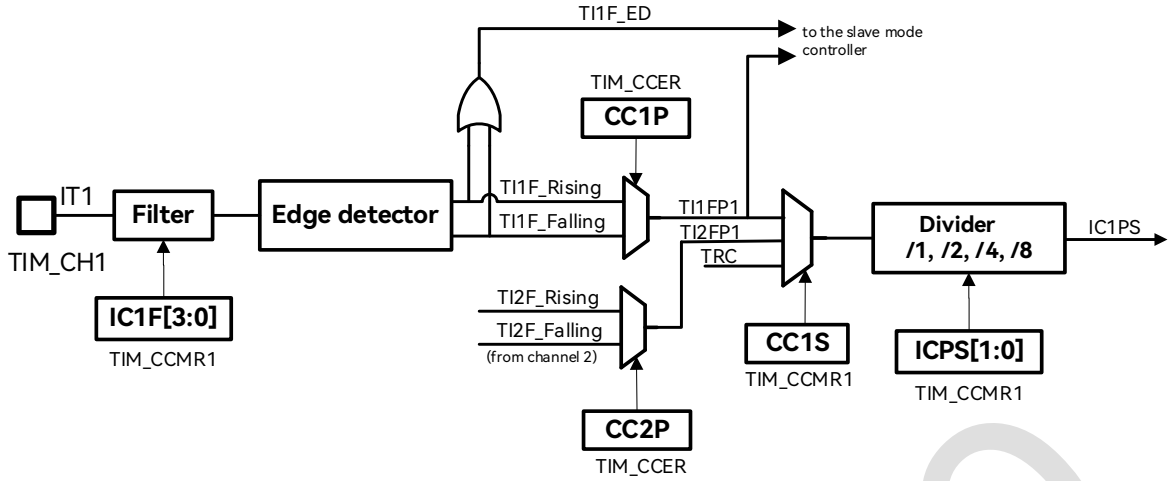


图 18-23：捕获/比较通道（通道 1 输入部分）图

输出级电路会产生一个输出基准信号 OCxREF，此信号固定为高电平有效，作为最终输出电路的参考输入。其中通道 1~3 支持互补输出和死区插入，通道 4 则比较简单，不支持互补输出。

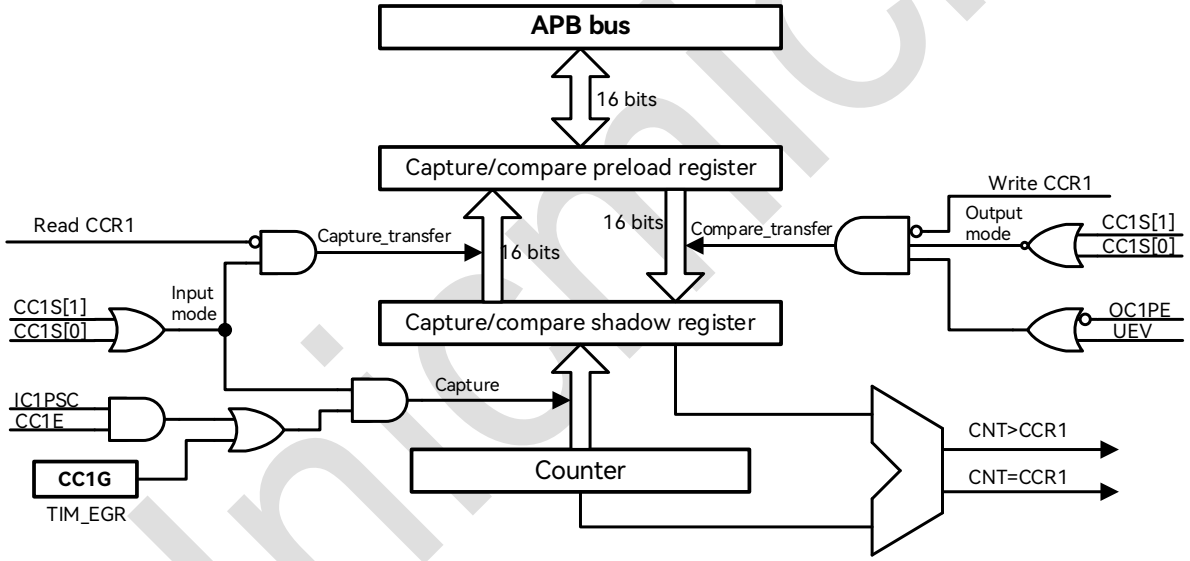


图 18-24：捕获/比较通道 1 的主电路图

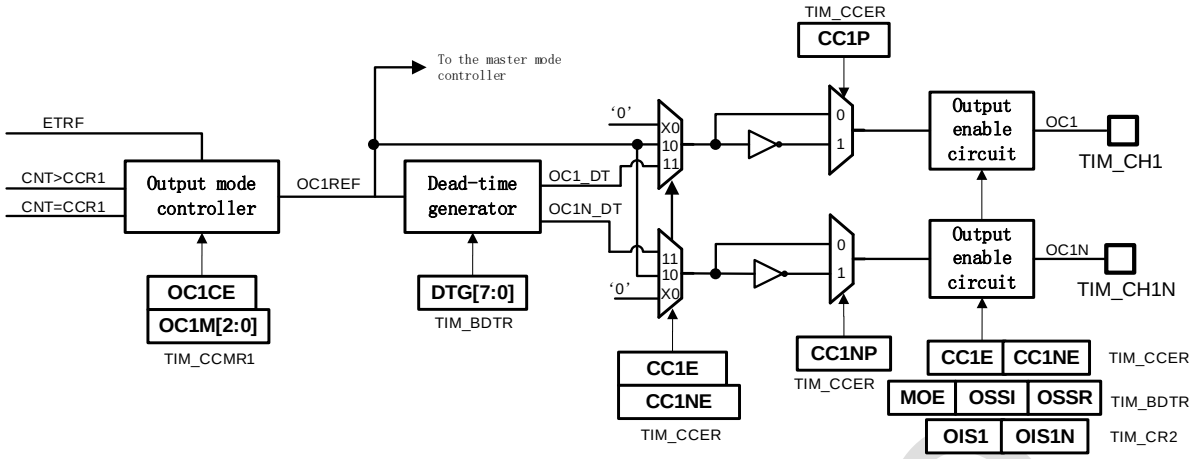


图 18-25：捕获/比较通道的输出部分（通道 1 至 3）图

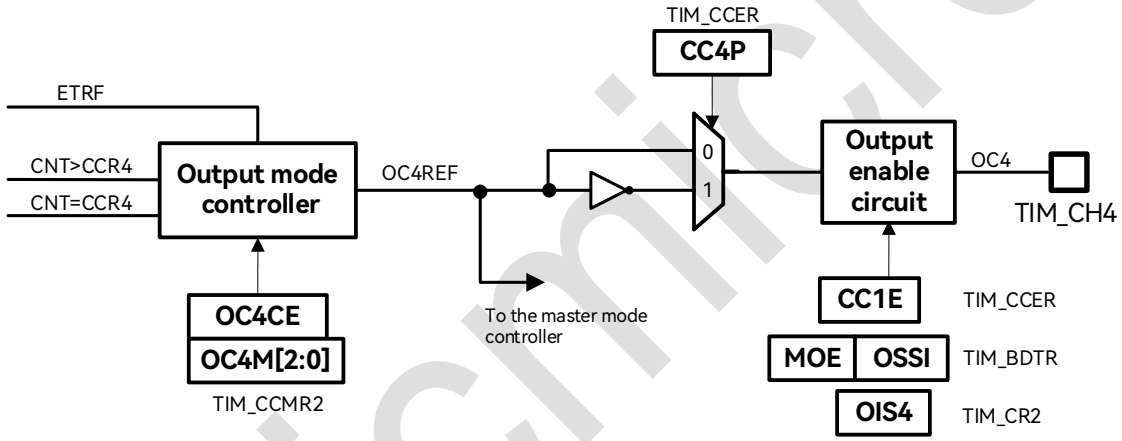


图 18-26：捕获/比较通道的输出部分（通道 4）图

捕获/比较寄存器（CCR）包含 preload 寄存器和 shadow 寄存器，软件读写总是访问 preload 寄存器。在捕获模式下，捕获值保存在 shadow 寄存器中并复制到 preload 寄存器。在比较模式下，preload 寄存器的值被拷贝到 shadow 寄存器用来与计数器比较。

18.6.8 输入捕获模式

当 ICx 信号上出现预期的电平变换，将触发一次 capture，当前计数器值被锁存进 CCR，与此同时，CCxIF 中断标志置位，并且可以触发对应的中断或者 DMA 请求。如果一个捕获事件在 CCxIF 为高的情况下出现，则捕获数据冲突标志（CCxOF, Over-Capture）置位（CCR 中上次捕获值被覆盖）。CCxIF 可以由软件清零，或者通过读取 CCR 寄存器自动清零。CCxOF 标志通过软件写 1 清零。

通过两个或更多通道配合，可以实现 PWM 信号的输入捕获。比如要计算一个输入信号的周期和占空比，可以将此信号从 TI1 引脚输入，芯片内部将滤波后的信号取上升沿得到 TI1FP1，将滤波后的信号取下降沿得到 TI1FP2，将 TI1FP1 输入给捕获通道 1，将 TI1FP2 输入给捕获通道 2，

即可实现通道 1 对输入信号上升沿捕获，同时通道 2 对输入信号下降沿捕获；捕获中断定期发生后，软件通过 CCR1 和 CCR2 寄存器的值，即可计算输入信号的周期和占空比。

实现在 TI1 输入的上升沿捕获计数器的值到 TIM_CCR1 寄存器，配置步骤如下：

1. 在 GPIO 模块中，配置相应管脚为 TIM_CH1 功能。
2. 关闭通道使能，配置 TIM_CCER[0]=0，确保之后通道配置成功。
3. 选择输入通道，配置 TIM_CCMR1[1:0]=01, IC1 映射到 TI1。
4. 选择计数有效沿，配置 TIM_CCER[1]，选择上沿或者下沿。
5. 配置输入滤波时间，配置 TIM_CCMR1.IC1F[3:0]。
6. 配置输入预分频器，配置 TIM_CCMR1.IC1PS[1:0]。
7. 打开通道使能，配置 TIM_CCER[0]=1。

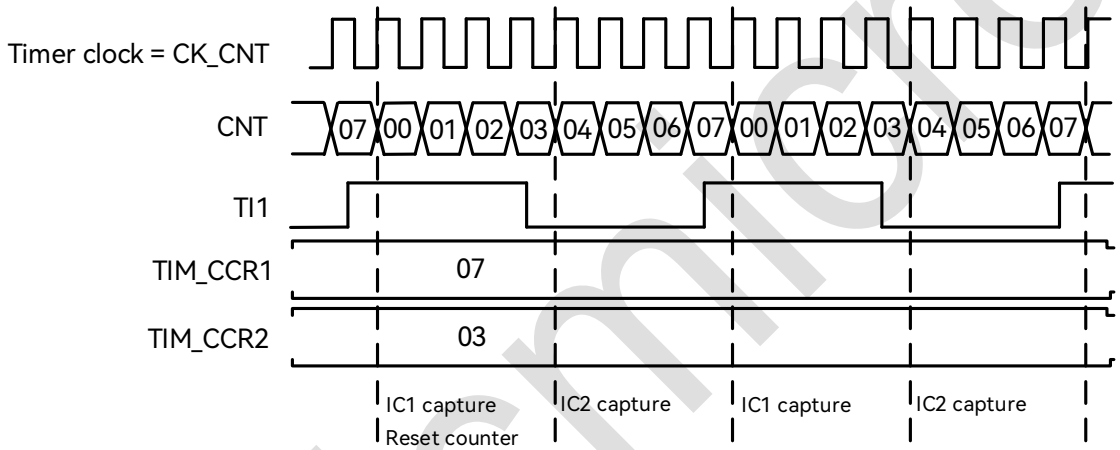


图 18-27：PWM 输入捕获模式时序图

若想实现 PWM 输入捕获功能，需进行如下设置：

1. 在 GPIO 模块中，配置相应管脚为 TIM_CH1 功能。
2. 关闭通道使能，配置 TIM_CCER[0]=0，TIM_CCER[4]=0 确保之后通道配置成功。
3. 选择输入通道，两个通道 IC1，IC2 被映射到同一个 TI1 输入口，配置 TIM_CCMR1[1:0]=01，TIM_CCMR1[9:8]=10。
4. 选择计数有效沿，两个通道 IC1,IC2 有效沿极性相反，配置 TIM_CCER[1]=0，TIM_CCER[5]=1。
5. 配置输入滤波时间，配置 TIM_CCMR1.IC1F[3:0]，TIM_CCMR1.IC2F[3:0]。
6. 配置输入预分频器，配置 TIM_CCMR1.IC1PS[1:0] ，TIM_CCMR1.IC2PS[1:0]。
7. 选择触发输入信号，配置 TIM_SMCR.TS[2:0]=101。
8. 设定从模式控制器为复位模式，配置 TIM_SMCR.SMS[2:0]=100。
9. 打开通道使能，配置 TIM_CCER[0]=1，TIM_CCER[4]=1。

18.6.9 软件 Force 输出

在比较输出模式下，软件可以直接将 OCxREF force 成特定电平，而独立于 CCR 和计数器的比较结果。

软件通过写 OCxM=101 寄存器，可以直接将 OCxREF 强制为有效（OCxREF 固定为高有效），通过写 OCxM=100 可以直接将 OCxREF 强制为无效（低电平）。但是软件 force 操作不会取消比较过程，CCR 和计数器的比较还会一直进行。

18.6.10 输出比较模式

输出比较模式下，当 CCR 与计数器值相等，OCxREF 可以被置位成有效、无效、或电平翻转。同时，中断标志也会置位，DMA 请求可以发送。

输出比较也可以被用于输出一个特定宽度的脉冲信号（单次输出）。

使用步骤：

1. 选择计数时钟（内部、外部、预分频等）。
2. 向ARR和CCR寄存器写入期望数据。
3. 根据需要设置中断使能和DMA使能。
4. 选择输出模式。
5. 使能计数器。

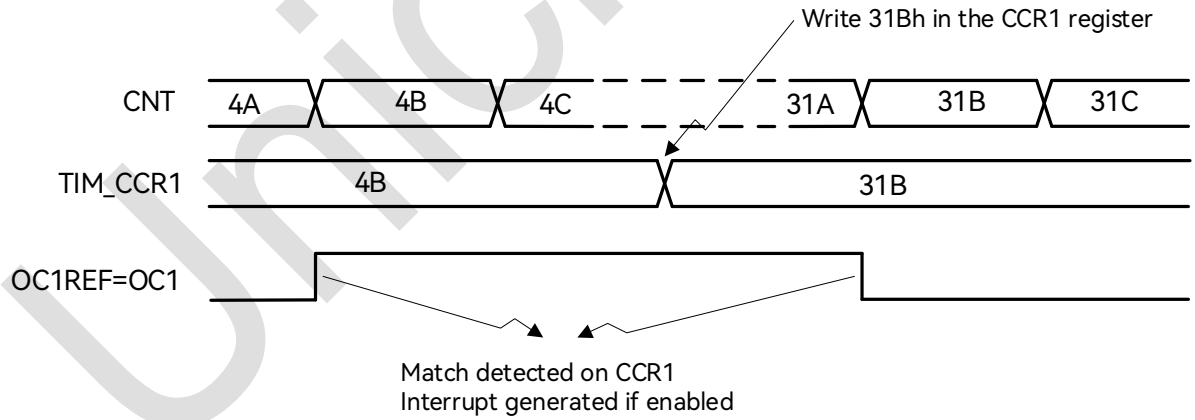


图 18-28：输出比较模式，翻转 OC1 时序图

在不使能 preload 的情况下，软件可以随时改写 CCR 寄存器实现对输出波形的实时控制。如果使能了 preload，则 CCR shadow 寄存器仅在下一次 update event 发生时更新为 preload 寄存器的内容。

18.6.11 PWM 输出

PWM 模式可以输出脉宽调制信号，其周期由 ARR 寄存器决定，占空比由 CCR 寄存器决定。

输出信号的极性可以由 CCxP 寄存器配置。PWM 模式工作中，CNT 和 CCR 实时比较。由于计数器支持边缘对齐和中央对齐计数模式，PWM 输出也支持边缘对齐和中央对齐模式。

18.6.11.1 PWM 边缘对齐模式

在向上计数的情况下，配置为 PWM 模式 1 时，OCxREF 信号在 CNT<CCR 时为高电平，否则为低电平。如果 CCR 值大于 ARR 值，则 OCxREF 被固定为 1；如果 CCR 为 0 则 OCxREF 被固定为 0。

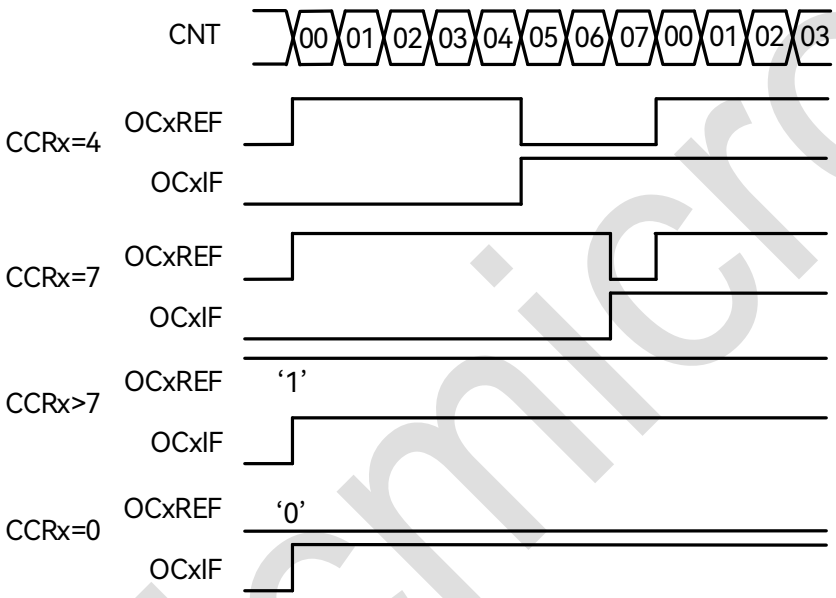


图 18-29：边沿对齐的 PWM 波形（ARR=7）图

在向下计数时，OCxREF 电平高低定义与向上计数时相同。

18.6.11.2 PWM 中央对齐模式

OCxREF 电平定义与边缘对齐模式相同。下图是一个示例：

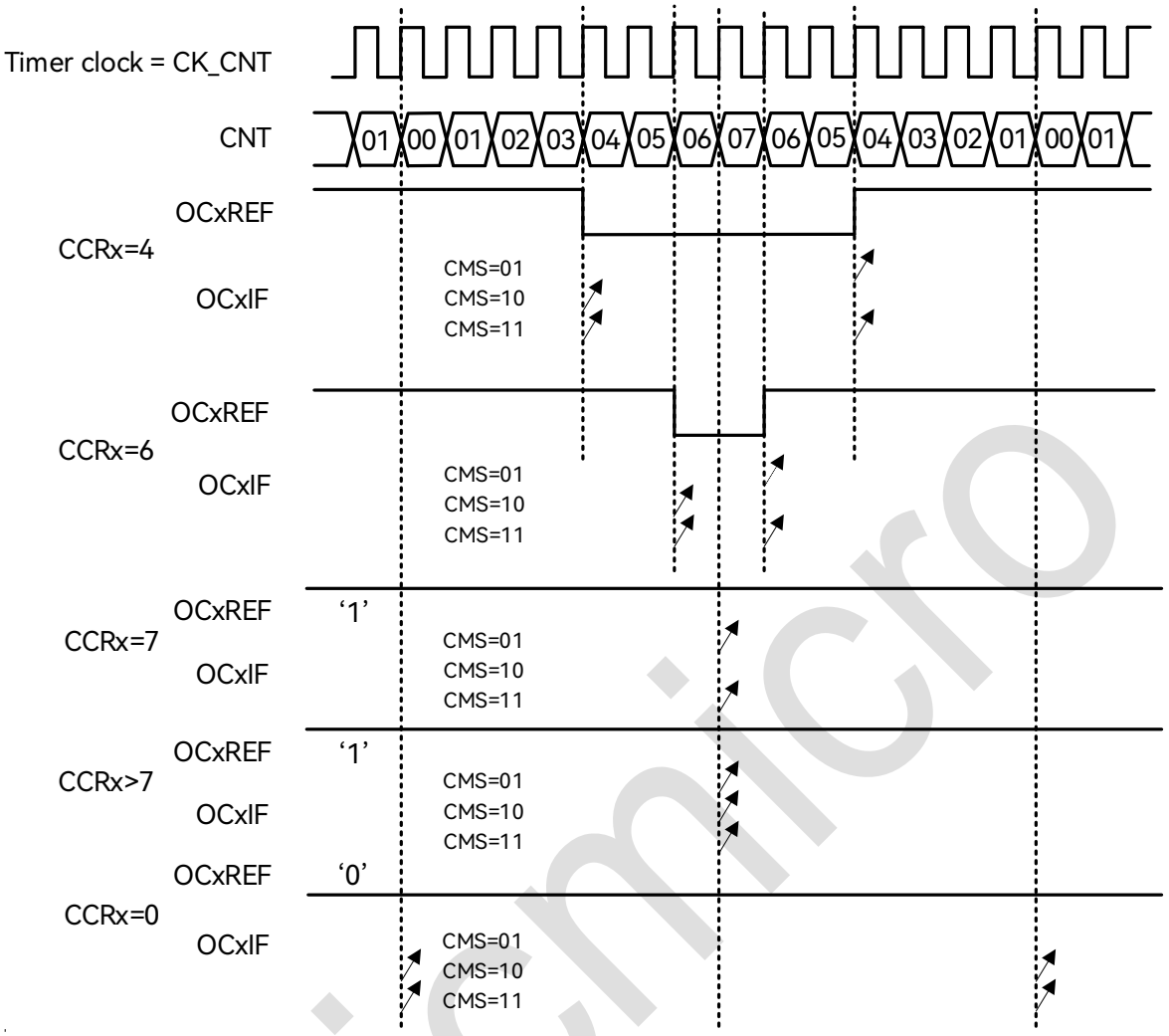


图 18-30：中央对齐的 PWM 波形（ARR=7）图

当启动中央对齐计数时，一开始的计数方向是由 DIR 寄存器决定的；随后在计数过程中，DIR 寄存器的状态由硬件直接控制。安全起见，建议用户程序在启动计数器之前，通过 UG 寄存器做一次 update，并且在计数过程中不要改写计数器。

18.6.12 互补输出和死区插入

TIM 的通道 1~3 支持互补输出和死区插入。DTG[7:0]寄存器用于设置死区时间（对所有通道同时有效）。输出信号 OCx 与参考信号 OCxREF 同相，OCxN 与参考信号反相；OCx 的上升沿是 OCxREF 上升沿的 delay，OCxN 的上升沿是 OCxREF 下降沿的 delay。

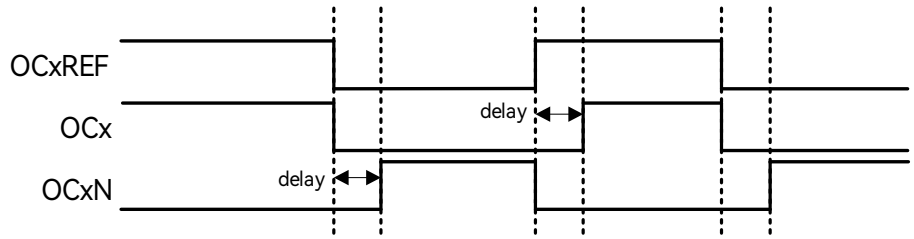


图 18-31：带死区插入的互补输出时序图

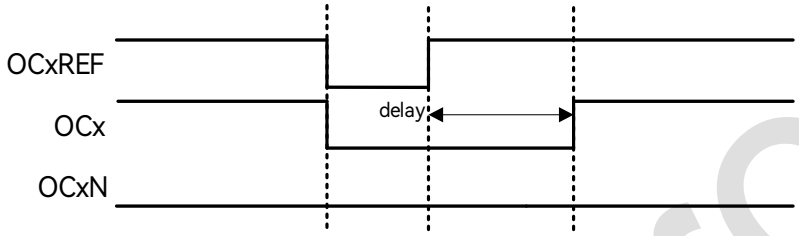


图 18-32：死区波形延迟大于负脉冲时序图

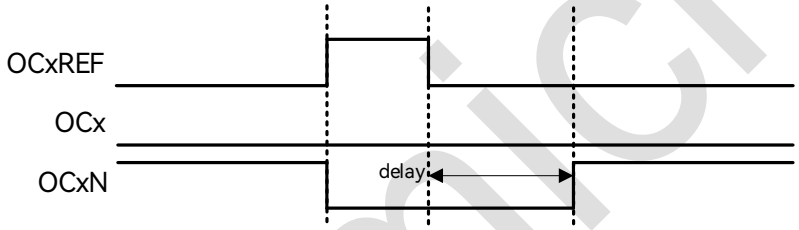


图 18-33：死区波形延迟大于正脉冲时序图

18.6.13 刹车功能

刹车功能可以使用外部刹车信号或时钟故障信号激活。

当一个刹车事件发生时：

- 输出使能寄存器被异步清零，可以通过OSSI寄存器选择输出被强制为inactive/idle/reset状态。
- 每个输出通道被驱动为OISx寄存器定义的电平。
- 当互补输出使能时，输出被异步置位成inactive和reset状态，死区插入电路开始工作，在死区时间后驱动输出为OISx和OISxN定义的电平。
- 刹车标志寄存器置位，根据配置可以触发中断或DMA。
- 如果使能了自动输出（AOE=1），输出使能位（MOE）将在下一个update event发生时被自动置位；否则MOE将保持为0直到被软件重新置位。

注意 BRK 信号是电平有效的，因此在 BRK 保持有效的情况下，无法使能 MOE，同时刹车标志 BIF 也无法清除。

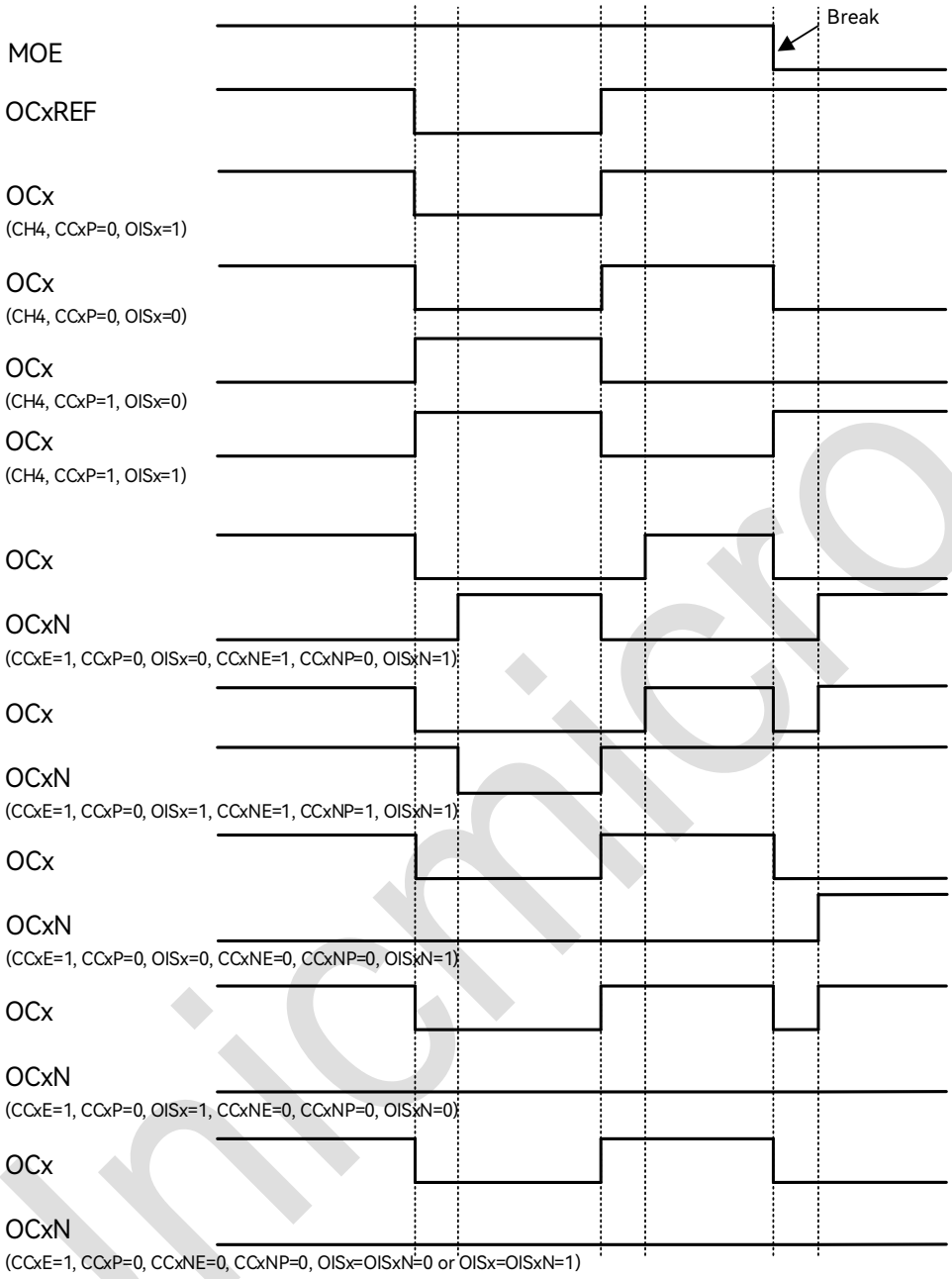


图 18-34：响应刹车的输出时序图

18.6.14 6-step PWM 输出

当某个通道使用互补输出时，OCxM, CCxE, CCxNE 寄存器支持 preload 功能，preload 寄存器的值在换相（COM）事件发生时被装载到 shadow 寄存器中。用户因此可以预先设置下一步配置，并在 COM 事件发生时同步更新所有通道。COM 事件可以由软件写 TIM_EGR 中的 COM 位触发，或者由 TRGI 上升沿硬件触发。

当 COM 事件发生时，换相标志寄存器置位，并且可以产生中断或 DMA 请求。

下图是一个 6 步换相控制的例子，当 COM 事件发生时，三个例子显示不同配置下的输出变化。

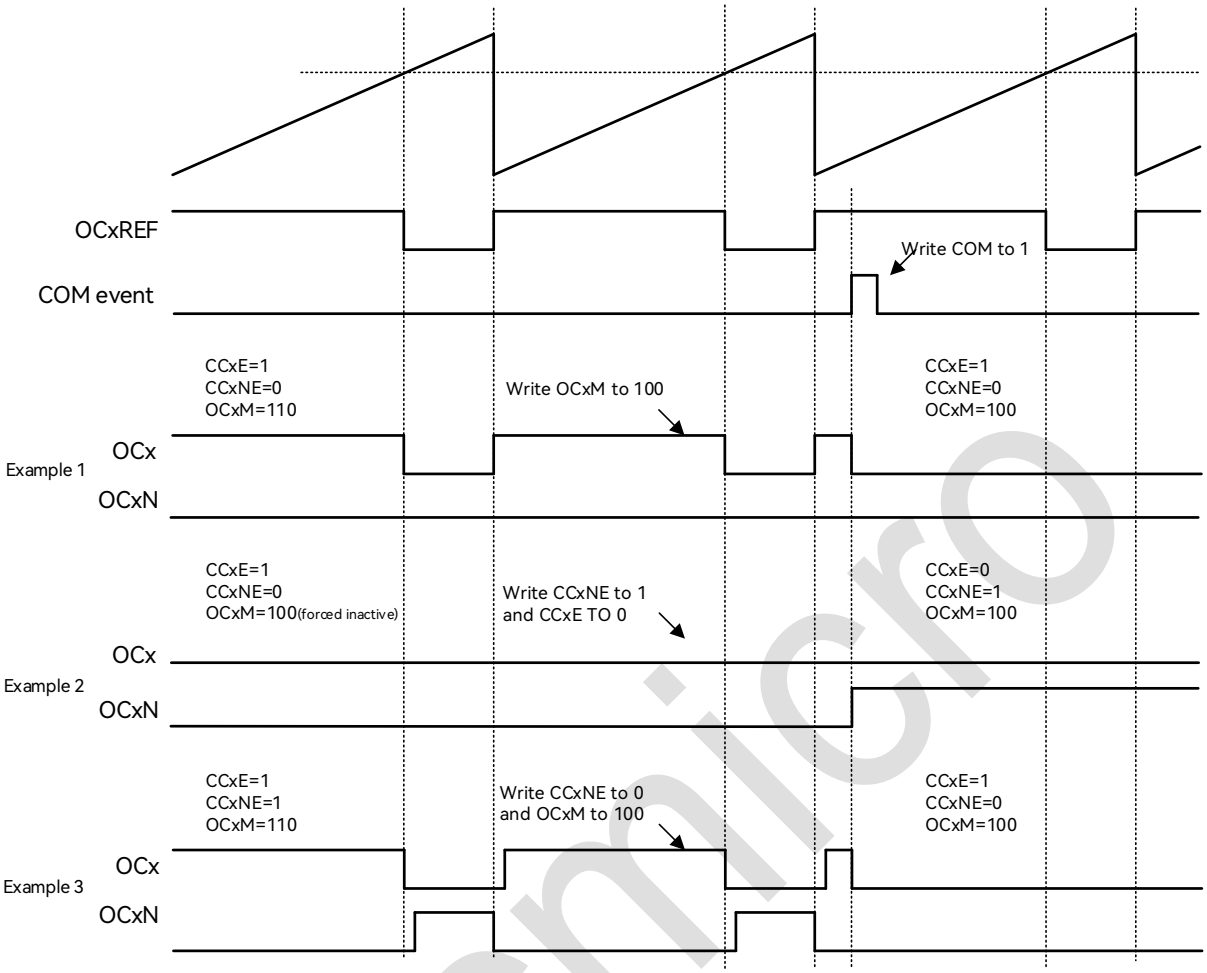


图 18-35：产生六步 PWM，使用 COM 的例子（OSSR=1)）时序图

18.6.15 单脉冲输出

单脉冲输出是比较输出模式的特殊情况，允许用户在某个事件发生后，经过可编程的延迟，输出一个可编程宽度的脉冲信号。

与其他输出模式不同的是，在下一次 update event 到来时，计数器会自动停止。只有当 CCR 和计数器初值不同时，脉冲才有可能正确输出。在向上计数时，要求 CNT<CCR<=ARR，在向下计数时，要求 CNT>CCR

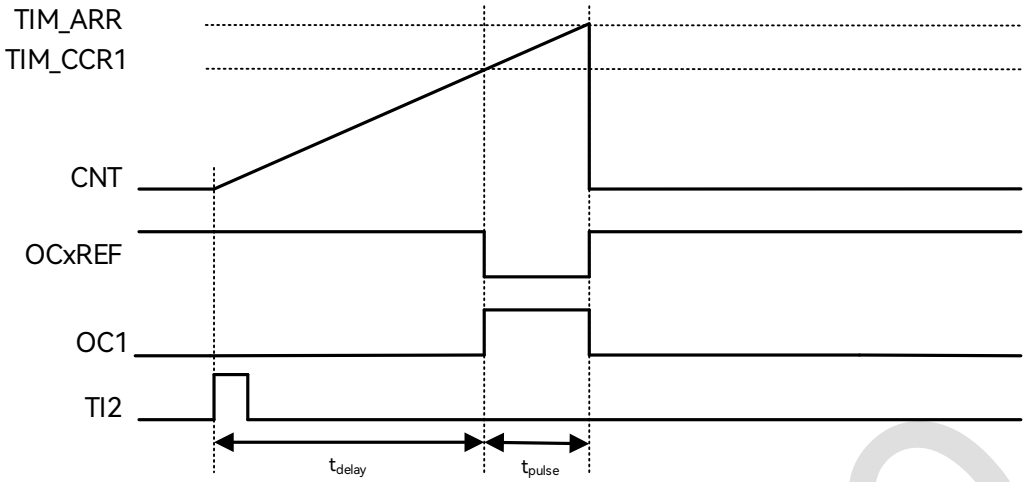


图 18-36：单脉冲模式时序图

上图是以 TI2 输入为计数器触发信号，计数值等于 CCR 后 OCxREF 输出低电平，计数到 ARR 后 OCxREF 回到高电平，并且计数器回滚到 0，停止计数。

实现上述功能 TI2 作为输入触发的配置如下：

1. 在 GPIO 模块中，配置相应管脚为 TIM_CH2 功能。
2. 关闭通道使能，配置 TIM_CCER[4]=0，确保之后通道配置成功。
3. 选择输入通道，配置 TIM_CCMR1[9:8]=01。
4. 选择计数有效沿，配置 TIM_CCER[5]=0。
5. 选择触发输入信号，配置 TIM_SMCR.TS[2:0]=110，TI2FP2 作为 TRGI。
6. 设定从模式控制器为触发模式，配置 TIM_SMCR.SMS[2:0]=110,TI2FP2 用来启动计数器。
7. 打开通道使能，配置 TIM_CCER[4]=1。

实现上述功能 OC1 作为输出的配置如下：

1. 在 GPIO 模块中，配置相应管脚为 TIM_CH1 功能。
2. 关闭通道使能，配置 TIM_CCER[0]=0，确保之后通道配置成功。
3. 输出通道，配置 TIM_CCMR1[1:0]=00。
4. 选择计数有效沿，配置 TIM_ CCMR1[6:4]=111,PWM 模式 2。
5. 打开通道使能，配置 TIM_CCER[0]=1。

OPM 波形产生时基的特殊设置：

1. TIM_CCR1 的值决定了 Tdelay。
2. TIM_ARR 和 TIM_CCR1 的差值决定了 Tpulse（TIM_ARR-TIM_CCR1）。
3. 设置为单脉冲模式，配置 TIM_CR1[3]=1。

18.6.16 外部事件清除 OCxREF

OCxREF 的有效状态未高电平，通过对外部 ETR 引脚施加高电平，可以直接拉低 OCxREF，直到下一次 update event。此功能仅在输出比较和 PWM 模式下有效，无法在软件 force 模式下起作用。使能此功能需要将 OCxCE 置 1。

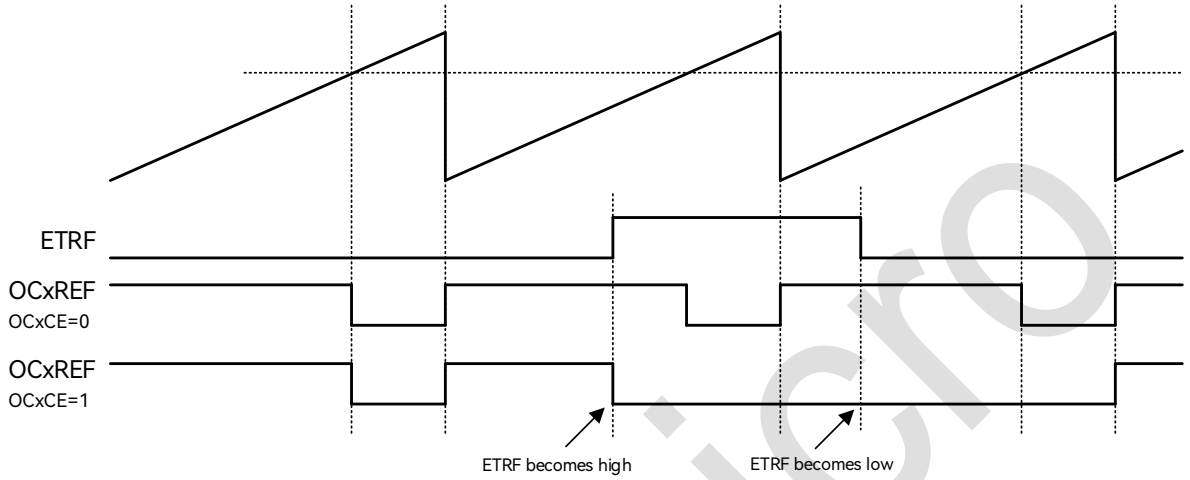


图 18-37：ETR 信号清除 TIM 的 OCxREF 时序图

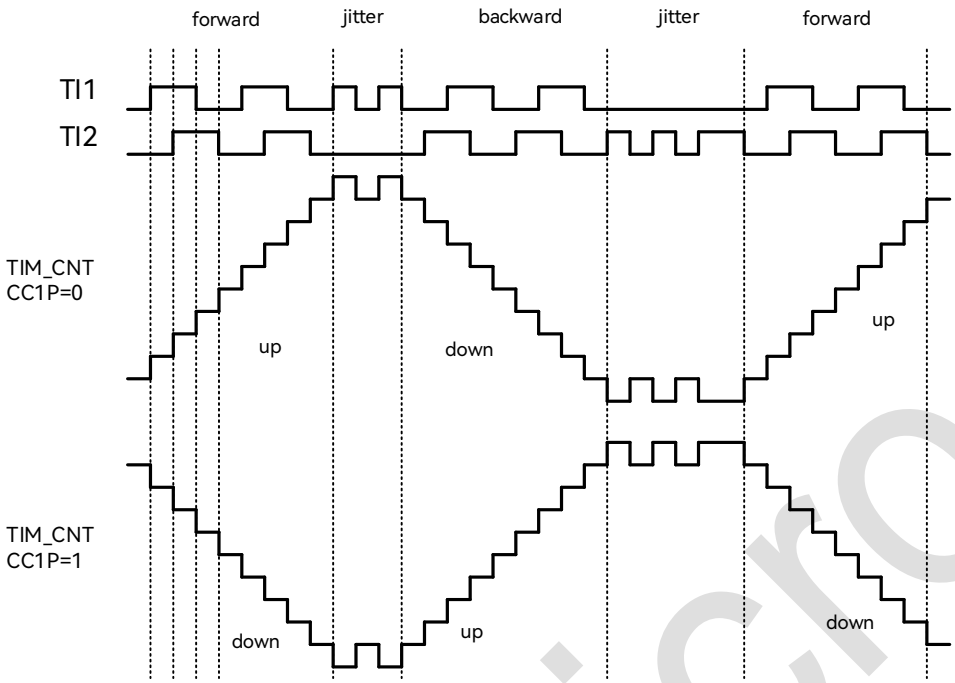
18.6.17 编码器接口模式（encoder interface）

编码器接口模式涉及到两个外部输入信号，TIM 根据其中一个信号的边沿相对于另一个信号的电平来决定递增还是递减计数值。下表是计数方式与两路输入信号之间的关系：

表 18-3：encoder interface 计数方式

有效沿	对应信号的电平（TI1 对应 TI2，TI2 对应 TI1）	TI1 信号		TI2 信号	
		上升	下降	上升	下降
仅在 TI1 处计数	高	递减	递增	不计数	不计数
	低	递增	递减	不计数	不计数
仅在 TI2 处计数	高	不计数	不计数	递增	递减
	低	不计数	不计数	递减	递增
在 TI1 和 TI2 处均计数	高	递减	递增	递增	递减
	低	递增	递减	递减	递增

比如在计数器以 TI1 信号为时钟计数时，如果 TI1 上升沿采样到 TI2 为高电平，则计数器递减；如果 TI1 下降沿采样到 TI2 为高电平，则计数器递增。



Example of counter operation in encoder interface mode

图 18-38：编码器模式下的计数器操作实例图

编码模式输入通道需进行如下设置：

1. 在GPIO模块中，配置相应管脚为TIM_CH1，TIM_CH2功能。
2. 关闭通道使能，配置TIM_CCER[0]=0，TIM_CCER[4]=0，确保之后通道配置成功。
3. 选择输入通道，配置TIM_CCMR1[1:0]=01，TIM_CCMR1[9:8]=01。
4. 选择计数有效沿，配置TIM_CCER[1]=0，TIM_CCER[5]=0。
5. 设定从模式控制器为编码模式3，配置TIM_SMCR.SMS[2:0]=011 。
6. 打开通道使能，配置TIM_CCER[0]=1，TIM_CCER[4]=1。

18.6.18 TIM 从机模式

TIM 作为 slave 时（外部事件触发），可配置为三种工作模式：复位模式、门控模式、触发模式。

18.6.18.1 复位模式

此模式下，外部输入的事件将导致 TIM 内部所有 preload 寄存器重新初始化，CNT 回到 0 开始计数。以下图为例，计数器正常计数，外部 TI1 输入上升沿时，触发计数器清零，重新开始计数。

下图例中的配置如下：

1. 在GPIO模块中，配置相应管脚为TIM_CH1功能。

2. 关闭通道使能，配置TIM_CCER[0]=0确保之后通道配置成功。
3. 选择输入通道，配置TIM_CCMR1[1:0]=01。
4. 选择计数有效沿，配置TIM_CCER[1]=0。
5. 选择触发输入信号，配置TIM_SMCR.TS[2:0]=101，TI1FP1作为TRGI。
6. 设定从模式控制器为复位模式，配置TIM_SMCR.SMS[2:0]=100。
7. 打开通道使能，配置TIM_CCER[0]=1。
8. 使能计数器，配置TIM_CR1[0]=1。

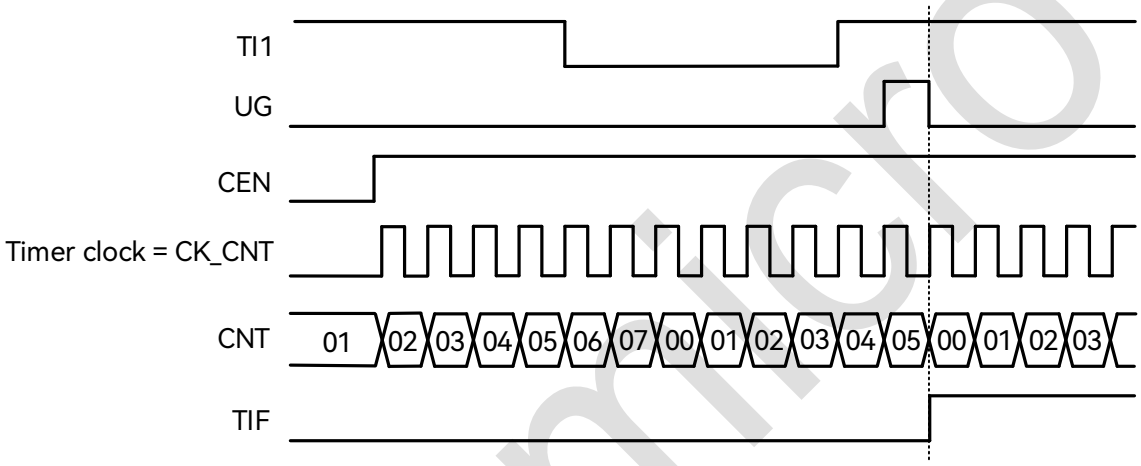


图 18-39：复位模式下的时序图

18.6.18.2 门控模式

此模式下，计数器仅在输入信号为特定电平时工作。电平变换导致计数器开始或停止计数时，都会触发中断标志。

下图例中的配置如下：

1. 在GPIO模块中，配置相应管脚为TIM_CH1功能。
2. 关闭通道使能，配置TIM_CCER[0]=0确保之后通道配置成功。
3. 选择输入通道，配置TIM_CCMR1[1:0]=01。
4. 选择计数有效沿，配置TIM_CCER[1]=0。
5. 选择触发输入信号，配置TIM_SMCR.TS[2:0]=101，TI1FP1作为TRGI。
6. 设定从模式控制器为门控模式，配置TIM_SMCR.SMS[2:0]=101。
7. 打开通道使能，配置TIM_CCER[0]=1。
8. 使能计数器，配置TIM_CR1[0]=1。

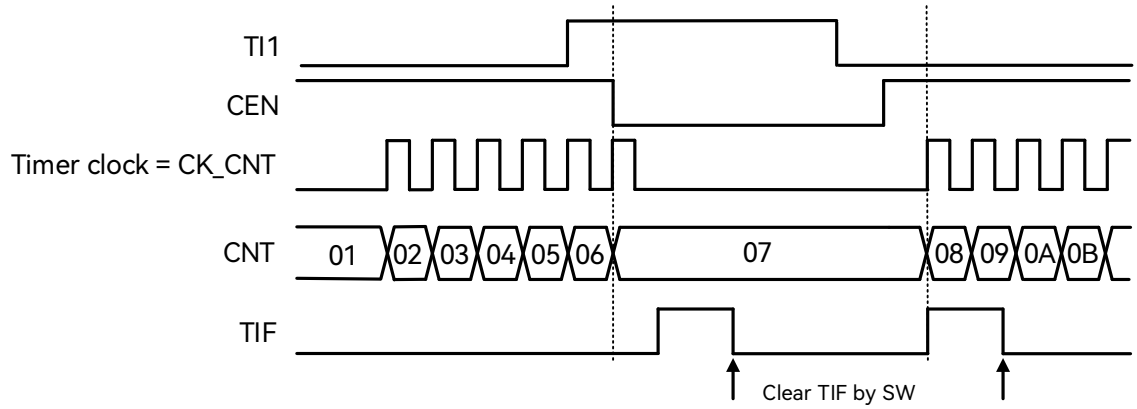


图 18-40：门控模式下的时序图

18.6.18.3 触发模式

计数器在外部输入的某个事件到来后才开始计数。

下图例中的配置如下：

1. 在GPIO模块中，配置相应管脚为TIM_CH1功能。
2. 关闭通道使能，配置TIM_CCER[0]=0确保之后通道配置成功。
3. 选择输入通道，配置TIM_CCMR1[1:0]=01。
4. 选择计数有效沿，配置TIM_CCER[1]=0。
5. 选择触发输入信号，配置TIM_SMCR.TS[2:0]=101，TI1FP1作为TRGI。
6. 设定从模式控制器为触发模式，配置TIM_SMCR.SMS[2:0]=110。
7. 打开通道使能，配置TIM_CCER[0]=1。

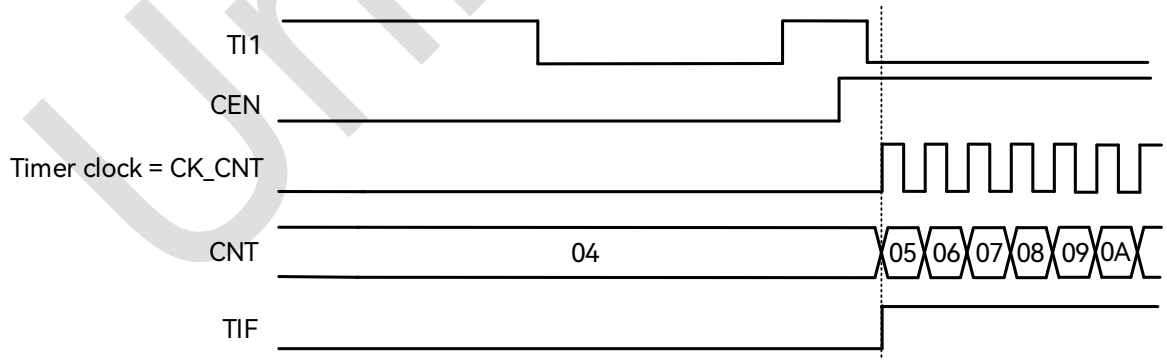


图 18-41：触发器模式下的时序图

18.6.18.4 外部事件触发的外部时钟计数模式

可以将 ETR 设置为计数时钟，同时使用另一个外部输入作为计数器启动触发信号。比如在检测到 TI1 的上升沿之后，计数器开始以 ETR 输入的上升沿计数。

下图例中的配置如下：

1. 在GPIO模块中，配置相应管脚为TIM_CH1，TIM_ETR功能。
2. 设置ETP进行沿选择，TIM_SMCR[15]=0。
3. 设置ETR分频比，配置TIM_SMCR.ETPS[1:0]=01。
4. 配置输入滤波时间，TIM_SMCR.ETF[3:0]=0000。
5. 置位ECE寄存器，使能外部时钟模式2,TIM_SMCR[14]=1。
6. 关闭通道使能，配置TIM_CCER[0]=0确保之后通道配置成功。
7. 选择输入通道，配置TIM_CCMR1[1:0]=01。
8. 选择计数有效沿，配置TIM_CCER[1]=0。
9. 选择触发输入信号，配置TIM_SMCR.TS[2:0]=101，TI1FP1作为TRGI。
10. 设定从模式控制器为触发模式，配置TIM_SMCR.SMS[2:0]=110。
11. 打开通道使能，配置TIM_CCER[0]=1。

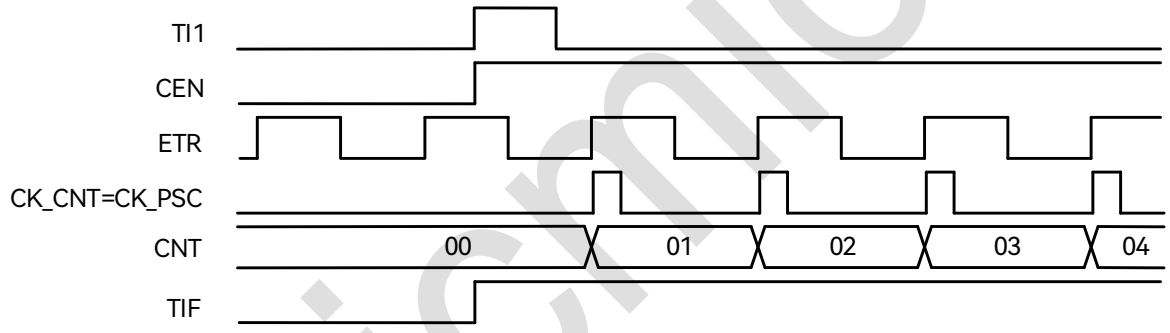


图 18-42：外部时钟模式 2＋触发模式下的时序图

18.6.19 定时器同步

定时器之间可以通过触发事件级联起来，实现定时器同步或级联。

定时器可以使用 4 个内部触发输入。

定时器的触发信号输出则可以接到其他定时器的内部触发输入上。

18.6.20 DMA 访问

TIM 支持 7 种 DMA 请求，分别为 4 个 CC 通道请求、外部触发请求、Update 事件请求和 COM 触发请求。

其中每个 CC 通道各自产生一个 DMA 请求，在捕获模式下用于将 CCRx 中的内容传输给 RAM，在比较模式下则用于将 RAM 中的数据写入 CCRx；CC 通道的 DMA 请求可以配置为单次传输或 Burst 传输（CCxBURSTEN），单次传输仅访问 CCRx 寄存器，Burst 传输则根据 DCR 寄

存器配置对特定的一组寄存器进行访问。

此外，外部触发事件、软件触发事件和 COM 事件也可以产生 DMA 请求，当这些请求发生时，会启动 DMA Burst 传输，向 TIM 内部 1 个或多个寄存器写入数据，或者从 TIM 读取 1 个或多个寄存器值。

表 18-4：TIM 支持 7 种 DMA 请求

DMA 请求	CCxBURSTEN	DMA.CHxCTRL.DIR	DMA 访问对象	一次传输长度
TIM_CH1	0	0	Read CCR1	1
		1	Write CCR1	
	1	0	Read DMAR	DBL
		1	Write DMAR	
TIM_CH2	0	0	Read CCR2	1
		1	Write CCR2	
	1	0	Read DMAR	DBL
		1	Write DMAR	
TIM_CH3	0	0	Read CCR3	1
		1	Write CCR3	
	1	0	Read DMAR	DBL
		1	Write DMAR	
TIM_CH4	0	0	Read CCR4	1
		1	Write CCR4	
	1	0	Read DMAR	DBL
		1	Write DMAR	
TIM_TRIG	N/A	0	Read DMAR	DBL
		1	Write DMAR	
TIM_UEV	N/A	0	Read DMAR	DBL
		1	Write DMAR	
TIM_COM	N/A	0	Read DMAR	DBL
		1	Write DMAR	

18.6.21 MA Burst

TIM 支持 DMA 和 DMA-Burst 访问，可以配置 TIM 在特定事件发生时触发 DMA 请求，可以将 CCR 中的捕获结果写入 RAM，或者从 RAM 中将一个或多个寄存器内容写入 TIM 的 preload 寄存器中。

DMA-Burst 支持一个事件触发连续多次 DMA 请求，主要作用是在事件发生后连续更新多个寄存器的内容，因此可以实现动态实时调整输出波形等功能。

DMA 控制器需将外设目标地址指向一个虚拟寄存器 TIM_DMAR。在特定的定时器事件发生时，TIM 会连续发射多个 DMA 请求。每个 DMA 对 TIM_DMAR 的写操作都会被 TIM 重新定向到实际的功能寄存器上。

DBL 寄存器用于设置 DMA burst 长度，DBA 寄存器用于设置 DMA 访问 TIM 内部的基地址（相对于 TIM_CR 的 offset）。

DMA-Burst 模式下，DMA 所有访问都要指向 DMAR 虚拟寄存器，由 TIM 自动根据访问来累加内部 offset 地址。DBA 寄存器用于指定 TIM 内部首次 DMA 传输的目标地址，而 DBL 用于指定 Burst 长度。

18.6.22 输入异或功能

通道 1~3 的输入信号可以被异或起来之后，接入到通道 1 的滤波和边沿电路输入，用于通道 1 的输入捕获或者触发。

TIM_CR2 寄存器的 TI1S 位用于选择通道 1 的输入是否来自于三个通道输入的异或。

18.6.23 Debug 模式

当 CPU 进入 debug 模式后，定时器可以停止或继续工作，其行为由芯片系统中的寄存器定义。

Debug 时当定时器被停止后，其输出会被禁止（MOE 清零），根据寄存器配置，此时的输出信号可以被 force 成 inactive 或由 GPIO 模块控制。

18.7 寄存器描述

TIM0 寄存器基地址：0x4700_4000

TIM7 寄存器基地址：0x4700_5000

寄存器列表如下：

表 18-5：TIM0 & TIM7 高级控制定时器寄存器表

偏移地址	名称	描述
0x00	TIM_CR1	控制寄存器 1
0x04	TIM_CR2	控制寄存器 2
0x08	TIM_SMCR	从机模式控制寄存器
0x0C	TIM_DIER	DMA 和中断使能寄存器
0x10	TIM_SR	状态寄存器
0x14	TIM_EGR	事件产生寄存器
0x18	TIM_CCMR1	捕获/比较寄存器 1
0x1C	TIM_CCMR2	捕获/比较寄存器 2
0x20	TIM_CCER	捕获/比较使能寄存器
0x24	TIM_CNT	计数值寄存器
0x28	TIM_PSC	预分频寄存器
0x2C	TIM_ARR	自动重载寄存器

偏移地址	名称	描述
0x30	TIM_RCR	重复计数寄存器
0x34	TIM_CCR1	捕获/比较寄存器 1
0x38	TIM_CCR2	捕获/比较寄存器 2
0x3C	TIM_CCR3	捕获/比较寄存器 3
0x40	TIM_CCR4	捕获/比较寄存器 4
0x44	TIM_BDTR	刹车和死区控制寄存器
0x48	TIM_DCR	DMA 控制寄存器
0x4C	TIM_DMAR	DMA 访问寄存器

以下各节详细介绍寄存器。

18.7.1 控制寄存器 1 (TIM_CR1)

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9:8	CKD	R/W	0x0	Dead time 和数字滤波时钟频率分频寄存器（相对 CK_INT 的分频比）： 00：t_{DTS}=tCK_INT 01：t_{DTS}=2*tCK_INT 10：t_{DTS}=4*tCK_INT 11：保留，禁止使用
7	APRE	R/W	0x0	Auto-reload 预装载使能： 0：ARR 寄存器不使能 preload 1：ARR 寄存器使能 preload
6:5	CMS	R/W	0x0	计数器对齐模式选择： 00：边沿对齐模式 01：中央对齐模式 1，输出比较中断标志仅在计数器向下计数的过程中置位 10：中央对齐模式 2，输出比较中断标志仅在计数器向上计数的过程中置位 11：中央对齐模式 3，输出比较中断标志在计数器向上向下计数的过程中都会置位
4	DIR	R/W	0x0	计数方向寄存器： 0：向上计数 1：向下计数 注意：当定时器配置为中央计数模式或编码器模式时，此寄存器只读
3	OPM	R/W	0x0	单脉冲输出模式：： 0：Update Event 发生时计数器不停止 1：Update Event 发生时计数器停止（自动清零 CEN）

位	名称	属性	复位值	描述
2	URS	R/W	0x0	更新请求选择： 0：以下事件能够产生 update 中断或 DMA 请求 ● 计数器上溢出或下溢出 ● 软件置位 UG 寄存器 ● 从机控制器产生 update 1：仅计数器上溢出或下溢出会产生 update 中断或 DMA 请求
1	UDIS	R/W	0x0	禁止 update： 0：使能 update 事件。以下事件发生时产生 update 事件 ● 计数器上溢出或下溢出 ● 软件置位 UG 寄存器 ● 从机控制器产生 update 1：禁止 update 事件，不更新 shadow 寄存器。当 UG 置位或从机控制器收到硬件 reset 时重新初始化计数器和预分频器。
0	CEN	R/W	0x0	计数器使能： 0：计数器关闭 1：计数器使能 注意：外部触发模式可以自动置位 CEN

18.7.2 控制寄存器 2（TIM_CR2）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:15	RSV	-	-	保留
14	OIS4	R/W	0x0	定义 OC4 的输出 IDLE 状态： 0：当 MOE=0 时（如果使能了互补输出，需经过 dead time 后），OC4=0 1：当 MOE=0 时（如果使能了互补输出，需经过 dead time 后），OC4=1
13	OIS3N	R/W	0x0	定义 OC3N 的输出 IDLE 状态： 0：当 MOE=0 时，经过 dead time 后，OC3N=0 1：当 MOE=0 时，经过 dead time 后，OC3N=1【对于 OC1~3N，还需要（CCxP||CCxNP）=1 来使 OCxN=1】
12	OIS3	R/W	0x0	定义 OC3 的输出 IDLE 状态： 0：当 MOE=0 时（如果使能了互补输出，需经过 dead time 后），OC3=0 1：当 MOE=0 时（如果使能了互补输出，需经过 dead time 后），OC3=1 【对于 OC1~3，还需要（CCxP||CCxNP）来使 OCx=1】

位	名称	属性	复位值	描述
11	OIS2N	R/W	0x0	定义 OC2N 的输出 IDLE 状态： 0：当 MOE=0 时，经过 dead time 后，OC2N=0 1：当 MOE=0 时，经过 dead time 后，OC2N=1【对于 OC1~3N，还需要（CCxP||CCxNP）=1 来使 OCxN=1】
10	OIS2	R/W	0x0	定义 OC2 的输出 IDLE 状态： 0：当 MOE=0 时（如果使能了互补输出，需经过 dead time 后），OC2=0 1：当 MOE=0 时（如果使能了互补输出，需经过 dead time 后），OC2=1 【对于 OC1~3，还需要（CCxP||CCxNP）来使 OCx=1】
9	OIS1N	R/W	0x0	定义 OC1N 的输出 IDLE 状态： 0：当 MOE=0 时，经过 dead time 后，OC1N=0 1：当 MOE=0 时，经过 dead time 后，OC1N=1【对于 OC1~3N，还需要（CCxP||CCxNP）=1 来使 OCxN=1】
8	OIS1	R/W	0x0	定义 OC1 的输出 IDLE 状态： 0：当 MOE=0 时（如果使能了互补输出，需经过 dead time 后），OC1=0 1：当 MOE=0 时（如果使能了互补输出，需经过 dead time 后），OC1=1 【对于 OC1~3，还需要（CCxP||CCxNP）来使 OCx=1】
7	TI1S	R/W	0x0	选择 T1 输入： 0：T1 输入来自 CH1 引脚 1：T1 输入来自 CH1、CH2、CH3 引脚异或组合
6:4	MMS	R/W	0x0	主机模式选择，选择 TRGO 触发的方式： 000：复位：EGR 寄存器中的 UG 位产生 TRGO 001：使能：TRGO 由计数器使能信号产生，包括 CEN 位和外部触发 010：更新：更新事件产生 TRGO 011：比较脉冲：发生输入捕获或比较事件，使 CC1F 置 1 时，产生 TRGO 100：比较：OC1REF 产生 TRGO 101：比较：OC2REF 产生 TRGO 110：比较：OC3REF 产生 TRGO 111：比较：OC4REF 产生 TRGO
3	CCDS	R/W	0x0	捕获/比较 DMA 选择： 0：发生 CCx 事件时产生 CCxDMA 请求； 1：发生更新事件时产生 CCxDMA 请求
2	CCUS	R/W	0x0	捕获/比较控制更新选择，选择通过何种方式更新预装载的捕获/比较控制位： 0：仅通过将 COMG 置 1 来更新 1：可通过将 COMG 置 1 或者 TRGI 信号来更新

位	名称	属性	复位值	描述
1	RSV	-	-	保留
0	CCPC	R/W	0x0	捕获/比较预装载控制： 0：CCxE, CCxNE, OCxM 寄存器不使能 preload 1：CCxE, CCxNE, OCxM 寄存器使能 preload，使能此项后，这些寄存器仅在发生换向事件（COM）时更新。 注意：此位仅在拥有互补输出功能的通道上有效

18.7.3 从机模式控制寄存器（TIM_SMCR）

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	ETP	R/W	0x0	外部触发信号极性配置： 0：高电平或上升沿有效 1：低电平或下降沿有效
14	ECE	R/W	0x0	外部时钟使能： 0：关闭外部时钟模式 2 1：使能外部时钟模式 2，计数器时钟为 ETRF 有效沿
13:12	ETPS	R/W	0x0	外部触发信号预分频寄存器： 外部触发信号 ETRP 的频率最多只能是 TIM 工作时钟的 1/4，当输入信号频率较高时，可以使用预分频。 00：不分频 01：2 分频 10：4 分频 11：8 分频
11:8	ETF	R/W	0x0	外部触发信号滤波时钟和长度选择： 0000：无滤波 0001：$f_{SAMPLING}=f_{CK\_INT}$, N=2 0010：$f_{SAMPLING}=f_{CK\_INT}$, N=4 0011：$f_{SAMPLING}=f_{CK\_INT}$, N=8 0100：$f_{SAMPLING}=f_{DTS}/2$, N=6 0101：$f_{SAMPLING}=f_{DTS}/2$, N=8 0110：$f_{SAMPLING}=f_{DTS}/4$, N=6 0111：$f_{SAMPLING}=f_{DTS}/4$, N=8 1000：$f_{SAMPLING}=f_{DTS}/8$, N=6 1001：$f_{SAMPLING}=f_{DTS}/8$, N=8 1010：$f_{SAMPLING}=f_{DTS}/16$, N=5 1011：$f_{SAMPLING}=f_{DTS}/16$, N=6 1100：$f_{SAMPLING}=f_{DTS}/16$, N=8 1101：$f_{SAMPLING}=f_{DTS}/32$, N=5

位	名称	属性	复位值	描述
				1110：$f_{SAMPLING}=f_{DTS}/32$, N=6 1111：$f_{SAMPLING}=f_{DTS}/32$, N=8
7	MSM	R/W	0x0	主机从机模式选择： 0：无动作 1：TRGI 触发的动作被延迟，以便于通过 TRGO 将当前定时器和从机定时器完美同步起来
6:4	TS	R/W	0x0	触发选择，用于选择同步计数器的触发源： 000：内部触发信号（ITR0） 001：内部触发信号（ITR1） 010：内部触发信号（ITR2） 011：内部触发信号（ITR3） 100：TI1 边沿检测（TI1F_ED） 101：滤波后 TI1（TI1FP1） 110：滤波后 TI2（TI2FP2） 111：外部触发输入（ETRF） 注意：仅当 SMS=000 即禁止从机模式的情况下，可以改写 TS 寄存器
3	RSV	-	-	保留
2:0	SMS	R/W	0x0	从机模式选择： 000：从机模式禁止；CEN 使能后预分频电路时钟源来自内部时钟 001：Encoder 模式 1；计数器使用 TI2FP1 边沿，根据 TI1FP2 电平高低来计数 010：Encoder 模式 2；计数器使用 TI1FP2 边沿，根据 TI2FP1 电平高低来计数 011：Encoder 模式 3；计数器同时使用 TI1FP1 和 TI2FP2 边沿，根据其他输入信号电平来计数 100：复位模式；TRGI 上升沿初始化计数器，并触发寄存器 update 101：闸门模式；TRGI 为高电平时，计数时钟使能，TRGI 为低电平时，计数时钟停止 110：触发模式；TRGI 上升沿触发计数器开始计数（不会复位计数器） 111：外部时钟模式 1；TRGI 上升沿直接驱动计数器

18.7.4 DMA 和中断使能寄存器（TIM_DIER）

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:20	RSV	-	-	保留

位	名称	属性	复位值	描述
19	CC4OF_DISABLE	R/W	0x0	选择禁用 CC4OF 中断： 0：不禁用 1：禁用
18	CC3OF_DISABLE	R/W	0x0	选择禁用 CC3OF 中断： 0：不禁用 1：禁用
17	CC2OF_DISABLE	R/W	0x0	选择禁用 CC2OF 中断： 0：不禁用 1：禁用
16	CC1OF_DISABLE	R/W	0x0	选择禁用 CC1OF 中断： 0：不禁用 1：禁用
15	RSV	-	-	保留
14	TDE	R/W	0x0	外部触发 DMA 请求使能： 0：从机模式下，禁止外部触发事件产生 DMA 请求 1：从机模式下，允许外部触发事件产生 DMA 请求（可用于自动更新 preload 寄存器）
13	COMDE	R/W	0x0	COM 事件 DMA 请求使能： 0：COM 事件发生时，禁止产生 DMA 请求 1：COM 事件发生时，允许产生 DMA 请求
12	CC4DE	R/W	0x0	捕获比较通道 4 的 DMA 请求使能： 0：禁止 CC4 DMA 请求 1：允许 CC4 DMA 请求
11	CC3DE	R/W	0x0	捕获比较通道 3 的 DMA 请求使能： 0：禁止 CC3 DMA 请求 1：允许 CC3 DMA 请求
10	CC2DE	R/W	0x0	捕获比较通道 2 的 DMA 请求使能： 0：禁止 CC2 DMA 请求 1：允许 CC2 DMA 请求
9	CC1DE	R/W	0x0	捕获比较通道 1 的 DMA 请求使能： 0：禁止 CC1 DMA 请求 1：允许 CC1 DMA 请求
8	UDE	R/W	0x0	Update Event DMA 请求使能： 0：Update Event 发生时，禁止产生 DMA 请求 1：Update Event 发生时，允许产生 DMA 请求
7	BIE	R/W	0x0	刹车事件中断使能： 0：禁止刹车事件中断 1：允许刹车事件中断
6	TIE	R/W	0x0	触发事件中断使能： 0：禁止触发事件中断 1：允许触发事件中断

位	名称	属性	复位值	描述
5	COMIE	R/W	0x0	COM 事件中断使能： 0：禁止 COM 事件中断 1：允许 COM 事件中断
4	CC4IE	R/W	0x0	捕获/比较通道 4 中断使能： 0：禁止捕获/比较 4 中断 1：允许捕获/比较 4 中断
3	CC3IE	R/W	0x0	捕获/比较通道 3 中断使能： 0：禁止捕获/比较 3 中断 1：允许捕获/比较 3 中断
2	CC2IE	R/W	0x0	捕获/比较通道 2 中断使能： 0：禁止捕获/比较 2 中断 1：允许捕获/比较 2 中断
1	CC1IE	R/W	0x0	捕获/比较通道 1 中断使能： 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	R/W	0x0	Update 事件中断使能： 0：禁止 Update 事件中断 1：允许 Update 事件中断

18.7.5 状态寄存器（TIM_SR）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12	CC4OF	R/W0C	0x0	捕获/比较通道 4 的 Overcapture 中断 此比特仅在对应通道设置为输入捕获模式的情况下有效。硬件置位，软件“0”清零。 0：无 overcapture 事件 1：在 CC4IF 标志为 1 的情况下发生新的捕获
11	CC3OF	R/W0C	0x0	捕获/比较通道 3 的 Overcapture 中断 此比特仅在对应通道设置为输入捕获模式的情况下有效。硬件置位，软件“0”清零。 0：无 overcapture 事件 1：在 CC3IF 标志为 1 的情况下发生新的捕获
10	CC2OF	R/W0C	0x0	捕获/比较通道 2 的 Overcapture 中断 此比特仅在对应通道设置为输入捕获模式的情况下有效。硬件置位，软件“0”清零。 0：无 overcapture 事件 1：在 CC2IF 标志为 1 的情况下发生新的捕获
9	CC1OF	R/W0C	0x0	捕获/比较通道 1 的 Overcapture 中断 此比特仅在对应通道设置为输入捕获模式的情况下有效。硬件置位，软件“0”清零。 0：无 overcapture 事件 1：在 CC1IF 标志为 1 的情况下发生新的捕获
8	RSV	-	-	保留

位	名称	属性	复位值	描述
7	BIF	R/W0C	0x0	刹车事件中断标志，硬件置位，软件写 0 清零
6	TIF	R/W0C	0x0	触发事件中断标志，硬件置位，软件写 0 清零
5	COMIF	R/W0C	0x0	COM 事件中断标志，硬件置位，软件写 0 清零
4	CC4IF	R/W0C	0x0	捕获/比较通道 4 中断标志 如果 CC4 通道配置为输出：CC4IF 在计数值等于比较值时置位，软件写 0 清零。 如果 CC4 通道配置为输入：发生捕获事件时置位，软件写 0 清零，或者软件读 TIM_CCR4 自动清零。
3	CC3IF	R/W0C	0x0	捕获/比较通道 3 中断标志 如果 CC3 通道配置为输出：CC3IF 在计数值等于比较值时置位，软件写 0 清零。 如果 CC3 通道配置为输入：发生捕获事件时置位，软件写 0 清零，或者软件读 TIM_CCR3 自动清零。
2	CC2IF	R/W0C	0x0	捕获/比较通道 2 中断标志 如果 CC2 通道配置为输出：CC2IF 在计数值等于比较值时置位，软件写 0 清零。 如果 CC2 通道配置为输入：发生捕获事件时置位，软件写 0 清零，或者软件读 TIM_CCR2 自动清零。
1	CC1IF	R/W0C	0x0	捕获/比较通道 1 中断标志 如果 CC1 通道配置为输出：CC1IF 在计数值等于比较值时置位，软件写 0 清零。 如果 CC1 通道配置为输入：发生捕获事件时置位，软件写 0 清零，或者软件读 TIM_CCR1 自动清零。
0	UIF	R/W0C	0x0	Update 事件中断标志，硬件置位，软件写 0 清零。 当以下事件发生时，UIF 置位，并更新 shadow 寄存器。 ● 重复计数器=0，并且 UDIS=0 的情况下，计数器发生溢出 ● URS=0 且 UDIS=0 的情况下，软件置位 UG 寄存器初始化计数器 ● URS=0 且 UDIS=0 的情况下，触发事件初始化计数器

18.7.6 事件产生寄存器（TIM_EGR）

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留

位	名称	属性	复位值	描述
7	BG	W	0x0	软件刹车，软件置位此寄存器产生刹车事件，硬件自动清零
6	TG	W	0x0	软件触发，软件置位此寄存器产生触发事件，硬件自动清零
5	COMG	W	0x0	软件 COM 事件，硬件置位，软件写 1 清零
4	CC4G	W	0x0	捕获/比较通道 4 软件触发 如果 CC4 通道配置为输出：CC4G 置位，在使能的情况下可以产生相应的中断和 DMA 请求 如果 CC4 通道配置为输入：当前计数值被捕获到 TIM_CCR4 寄存器，CC4G 置位，在使能的情况下可以产生相应的中断和 DMA 请求
3	CC3G	W	0x0	捕获/比较通道 3 软件触发 如果 CC3 通道配置为输出：CC3G 置位，在使能的情况下可以产生相应的中断和 DMA 请求 如果 CC3 通道配置为输入：当前计数值被捕获到 TIM_CCR3 寄存器，CC3G 置位，在使能的情况下可以产生相应的中断和 DMA 请求
2	CC2G	W	0x0	捕获/比较通道 2 软件触发 如果 CC2 通道配置为输出：CC2G 置位，在使能的情况下可以产生相应的中断和 DMA 请求 如果 CC2 通道配置为输入：当前计数值被捕获到 TIM_CCR2 寄存器，CC2G 置位，在使能的情况下可以产生相应的中断和 DMA 请求
1	CC1G	W	0x0	捕获/比较通道 1 软件触发 如果 CC1 通道配置为输出：CC1G 置位，在使能的情况下可以产生相应的中断和 DMA 请求 如果 CC1 通道配置为输入：当前计数值被捕获到 TIM_CCR1 寄存器，CC1G 置位，在使能的情况下可以产生相应的中断和 DMA 请求
0	UG	W	0x0	软件 Update 事件，软件置位此寄存器产生 Update 事件，硬件自动清零。 软件置位 UG 时会重新初始化计数器并更新 shadow 寄存器，预分频计数器被清零。

18.7.7 捕获/比较寄存器 1（TIM_CCMR1）

偏移地址：0x18

复位值：0x0000 0000

注：此寄存器在输出比较和输入捕获配置下复用为两组不同功能。

- 输出比较模式

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	OC2CE	R/W	0x0	输出比较 2 清零使能 0：OC2REF 不受 ETRF 影响 1：检测到 ETRF 高电平时，自动清零 OC2REF
14:12	OC2M	R/W	0x0	输出比较 2 模式配置 此寄存器定义 OC2REF 信号的行为： 000：输出比较寄存器 CCR2 和计数器 CNT 的比较结果不会影响输出 001：CCR2=CNT 时（后边沿），将 OC2REF 置高 010：CCR2=CNT 时（后边沿），将 OC2REF 置低 011：CCR2=CNT 时（后边沿），翻转 OC2REF 100：OC2REF 固定为低（inactive） 101：OC2REF 固定为高（active） 110：PWM 模式 1： 在向上计数时，OC2REF 在 CNT<CCR2 时置高，否则置低。 在向下计数时，OC2REF 在 CNT>=CCR2 时置低，否则置高。 111：PWM 模式 2 在向上计数时，OC2REF 在 CNT<CCR2 时置低，否则置高。 在向下计数时，OC2REF 在 CNT>=CCR2 时置高，否则置低
11	OC2PE	R/W	0x0	输出比较 2 预装载使能 0：CCR2 preload 寄存器无效，CCR2 可以直接写入 1：CCR2 preload 寄存器有效，针对 CCR2 的读写操作都是访问 preload 寄存器，当 update event 发生时才将 preload 寄存器的内容转移到 shadow 寄存器中
10	OC2FE	R/W	0x0	输出比较 2 快速使能 0：关闭快速使能，trigger 输入不会影响比较输出 1：打开快速使能，trigger 输入会立即将 OC2REF 改变为比较值匹配时的输出，而不管当前实际比较情况 此功能仅在当前通道配置为 PWM1 或 PWM2 模式时有效
9:8	CC2S	R/W	0x0	捕获/比较 2 通道选择： 00：CC2 通道配置为输出 01：CC2 通道配置为输入，IC2 映射到 TI2 10：CC2 通道配置为输入，IC2 映射到 TI1 11：CC2 通道配置为输入，IC2 映射到 TRC 注意：CC2S 仅在通道关闭时（CC2E=0）可以写
7	OC1CE	R/W	0x0	输出比较 1 清零使能： 0：OC1REF 不受 ETRF 影响 1：检测到 ETRF 高电平时，自动清零 OC1REF

位	名称	属性	复位值	描述
6:4	OC1M	R/W	0x0	输出比较 1 模式配置，此寄存器定义 OC1REF 信号的行为： 000：输出比较寄存器 CCR1 和计数器 CNT 的比较结果不会影响输出 001：CCR1=CNT 时（后边沿），将 OC1REF 置高 010：CCR1=CNT 时（后边沿），将 OC1REF 置低 011：CCR1=CNT 时（后边沿），翻转 OC1REF 100：OC1REF 固定为低（inactive） 101：OC1REF 固定为高（active） 110：PWM 模式 1： 在向上计数时，OC1REF 在 CNT<CCR1 时置高，否则置低。 在向下计数时，OC1REF 在 CNT>=CCR1 时置低，否则置高。 111：PWM 模式 2 在向上计数时，OC1REF 在 CNT<CCR1 时置低，否则置高。 在向下计数时，OC1REF 在 CNT>=CCR1 时置高，否则置低。
3	OC1PE	R/W	0x0	输出比较 1 预装载使能： 0：CCR1 preload 寄存器无效，CCR1 可以直接写入 1：CCR1 preload 寄存器有效，针对 CCR1 的读写操作都是访问 preload 寄存器，当 update event 发生时才将 preload 寄存器的内容转移到 shadow 寄存器中
2	OC1FE	R/W	0x0	输出比较 1 快速使能： 0：关闭快速使能，trigger 输入不会影响比较输出 1：打开快速使能，trigger 输入会立即将 OC1REF 改变为比较值匹配时的输出，而不管当前实际比较情况此功能仅在当前通道配置为 PWM1 或 PWM2 模式时有效
1:0	CC1S	R/W	0x0	捕获/比较 1 通道选择： 00：CC1 通道配置为输出 01：CC1 通道配置为输入，IC1 映射到 TI1 10：CC1 通道配置为输入，IC1 映射到 TI2 11：CC1 通道配置为输入，IC1 映射到 TRC 注意：CC1S 仅在通道关闭时（CC1E=0）可以写

- 输入捕获模式

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:12	IC2F	R/W	0x0	输入捕获 2 滤波
11:10	IC2PSC	R/W	0x0	输入捕获 2 预分频

位	名称	属性	复位值	描述
9:8	CC2S	R/W	0x0	捕获/比较 2 通道选择： 00：CC2 通道配置为输出 01：CC2 通道配置为输入，IC2 映射到 TI2 10：CC2 通道配置为输入，IC2 映射到 TI1 11：CC2 通道配置为输入，IC2 映射到 TRC 注意：CC2S 仅在通道关闭时（CC2E=0）可以写
7:4	IC1F	R/W	0x0	输入捕获 1 滤波： 此寄存器定义 TI1 的采样频率和滤波长度 0000：无滤波，使用 fDTS 采样 0001：$f_{SAMPLING} = f_{CK\_INT}$, N = 2 0010：$f_{SAMPLING} = f_{CK\_INT}$, N = 4 0011：$f_{SAMPLING} = f_{CK\_INT}$, N = 8 0100：$f_{SAMPLING} = f_{DTS} / 2$, N = 6 0101：$f_{SAMPLING} = f_{DTS} / 2$, N = 8 0110：$f_{SAMPLING} = f_{DTS} / 4$, N = 6 0111：$f_{SAMPLING} = f_{DTS} / 4$, N = 8 1000：$f_{SAMPLING} = f_{DTS} / 8$, N = 6 1001：$f_{SAMPLING} = f_{DTS} / 8$, N = 8 1010：$f_{SAMPLING} = f_{DTS} / 16$, N = 5 1011：$f_{SAMPLING} = f_{DTS} / 16$, N = 6 1100：$f_{SAMPLING} = f_{DTS} / 16$, N = 8 1101：$f_{SAMPLING} = f_{DTS} / 32$, N = 5 1110：$f_{SAMPLING} = f_{DTS} / 32$, N = 6 1111：$f_{SAMPLING} = f_{DTS} / 32$, N = 8
3:2	IC1PSC	R/W	0x0	输入捕获 1 预分频： 00：无分频 01：每 2 个事件输入产生一次捕获 10：每 4 个事件输入产生一次捕获 11：每 8 个事件输入产生一次捕获 IC1PSC 寄存器在 CC1E=0 时复位
1:0	CC1S	R/W	0x0	捕获/比较 1 通道选择： 00：CC1 通道配置为输出 01：CC1 通道配置为输入，IC1 映射到 TI1 10：CC1 通道配置为输入，IC1 映射到 TI2 11：CC1 通道配置为输入，IC1 映射到 TRC 注意：CC1S 仅在通道关闭时（CC1E=0）可以写

18.7.8 捕获/比较寄存器 2（TIM_CCMR2）

偏移地址：0x1C

复位值：0x0000 0000

注：此寄存器在输出比较和输入捕获配置下复用为两组不同功能。

- 输出比较模式

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	OC4CE	R/W	0x0	输出比较 4 清零使能： 0：OC4REF 不受 ETRF 影响 1：检测到 ETRF 高电平时，自动清零 OC4REF
14:12	OC4M	R/W	0x0	输出比较 4 模式配置。 此寄存器定义 OC4REF 信号的行为： 000：输出比较寄存器 CCR4 和计数器 CNT 的比较结果不会影响输出： 001：CCR4=CNT 时，将 OC4REF 置高 010：CCR4=CNT 时，将 OC4REF 置低 011：CCR4=CNT 时，翻转 OC4REF 100：OC4REF 固定为低（inactive） 101：OC4REF 固定为高（active） 110：PWM 模式 1 在向上计数时，OC4REF 在 CNT<CCR4 时置高，否则置低。 在向下计数时，OC4REF 在 CNT>CCR4 时置低，否则置高。 111：PWM 模式 2 在向上计数时，OC4REF 在 CNT<CCR4 时置低，否则置高。 在向下计数时，OC4REF 在 CNT>CCR4 时置高，否则置低。
11	OC4PE	R/W	0x0	输出比较 4 预装载使能： 0：CCR4 preload 寄存器无效，CCR4 可以直接写入 1：CCR4 preload 寄存器有效，针对 CCR4 的读写操作都是访问 preload 寄存器，当 update event 发生时才将 preload 寄存器的内容转移到 shadow 寄存器中
10	OC4FE	R/W	0x0	输出比较 4 快速使能： 0：关闭快速使能，trigger 输入不会影响比较输出 1：打开快速使能，trigger 输入会立即将 OC4REF 改变为比较值匹配时的输出，而不管当前实际比较情况 此功能仅在当前通道配置为 PWM1 或 PWM2 模式时有效

位	名称	属性	复位值	描述
9:8	CC4S	R/W	0x0	捕获/比较 4 通道选择： 00：CC4 通道配置为输出 01：CC4 通道配置为输入，IC4 映射到 TI4 10：CC4 通道配置为输入，IC4 映射到 TI3 11：CC4 通道配置为输入，IC4 映射到 TRC 注意：CC4S 仅在通道关闭时（CC4E=0）可以写
7	OC3CE	R/W	0x0	输出比较 3 清零使能： 0：OC3REF 不受 ETRF 影响 1：检测到 ETRF 高电平时，自动清零 OC3REF
6:4	OC3M	R/W	0x0	输出比较 3 模式配置，此寄存器定义 OC3REF 信号的行为： 000：输出比较寄存器 CCR3 和计数器 CNT 的比较结果不会影响输出 001：CCR3=CNT 时，将 OC3REF 置高 010：CCR3=CNT 时，将 OC3REF 置低 011：CCR3=CNT 时，翻转 OC3REF 100：OC3REF 固定为低（inactive） 101：OC3REF 固定为高（active） 110：PWM 模式 1 在向上计数时，OC3REF 在 CNT<CCR3 时置高，否则置低。 在向下计数时，OC3REF 在 CNT>CCR3 时置低，否则置高。 111：PWM 模式 2 在向上计数时，OC3REF 在 CNT<CCR3 时置低，否则置高。 在向下计数时，OC3REF 在 CNT>CCR3 时置高，否则置低。
3	OC3PE	R/W	0x0	输出比较 3 预装载使能： 0：CCR3 preload 寄存器无效，CCR3 可以直接写入 1：CCR3 preload 寄存器有效，针对 CCR3 的读写操作都是访问 preload 寄存器，当 update event 发生时才将 preload 寄存器的内容转移到 shadow 寄存器中
2	OC3FE	R/W	0x0	输出比较 3 快速使能： 0：关闭快速使能，trigger 输入不会影响比较输出 1：打开快速使能，trigger 输入会立即将 OC3REF 改变为比较值匹配时的输出，而不管当前实际比较情况 此功能仅在当前通道配置为 PWM1 或 PWM2 模式时有效

位	名称	属性	复位值	描述
1:0	CC3S	R/W	0x0	捕获/比较 3 通道选择： 00：CC3 通道配置为输出 01：CC3 通道配置为输入，IC3 映射到 TI3 10：CC3 通道配置为输入，IC3 映射到 TI4 11：CC3 通道配置为输入，IC3 映射到 TRC 注意：CC3S 仅在通道关闭时（CC3E=0）可以写

- 输入捕获模式

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:12	IC4F	R/W	0x0	输入捕获 4 滤波。 此寄存器定义 TI4 的采样频率和滤波长度： 0000：无滤波，使用 fDTS 采样 0001：$f_{SAMPLING}=f_{CK\_INT}$, N=2 0010：$f_{SAMPLING}=f_{CK\_INT}$, N=4 0011：$f_{SAMPLING}=f_{CK\_INT}$, N=8 0100：$f_{SAMPLING}=f_{DTS}/2$, N=6 0101：$f_{SAMPLING}=f_{DTS}/2$, N=8 0110：$f_{SAMPLING}=f_{DTS}/4$, N=6 0111：$f_{SAMPLING}=f_{DTS}/4$, N=8 1000：$f_{SAMPLING}=f_{DTS}/8$, N=6 1001：$f_{SAMPLING}=f_{DTS}/8$, N=8 1010：$f_{SAMPLING}=f_{DTS}/16$, N=5 1011：$f_{SAMPLING}=f_{DTS}/16$, N=6 1100：$f_{SAMPLING}=f_{DTS}/16$, N=8 1101：$f_{SAMPLING}=f_{DTS}/32$, N=5 1110：$f_{SAMPLING}=f_{DTS}/32$, N=6 1111：$f_{SAMPLING}=f_{DTS}/32$, N=8
11:10	IC4PSC	R/W	0x0	输入捕获 4 预分频： 00：无分频 01：每 2 个事件输入产生一次捕获 10：每 4 个事件输入产生一次捕获 11：每 8 个事件输入产生一次捕获 IC4PSC 寄存器在 CC4E=0 时复位
9:8	CC4S	R/W	0x0	捕获/比较 4 通道选择： 00：CC4 通道配置为输出 01：CC4 通道配置为输入，IC4 映射到 TI4 10：CC4 通道配置为输入，IC4 映射到 TI3 11：CC4 通道配置为输入，IC4 映射到 TRC 注意：CC4S 仅在通道关闭时（CC4E=0）可以写

位	名称	属性	复位值	描述
7:4	IC3F	R/W	0x0	输入捕获 3 滤波： 此寄存器定义 TI3 的采样频率和滤波长度 0000：无滤波，使用 fDTS 采样 0001：$f_{SAMPLING}=f_{CK\_INT}$, N=2 0010：$f_{SAMPLING}=f_{CK\_INT}$, N=4 0011：$f_{SAMPLING}=f_{CK\_INT}$, N=8 0100：$f_{SAMPLING}=f_{DTS}/2$, N=6 0101：$f_{SAMPLING}=f_{DTS}/2$, N=8 0110：$f_{SAMPLING}=f_{DTS}/4$, N=6 0111：$f_{SAMPLING}=f_{DTS}/4$, N=8 1000：$f_{SAMPLING}=f_{DTS}/8$, N=6 1001：$f_{SAMPLING}=f_{DTS}/8$, N=8 1010：$f_{SAMPLING}=f_{DTS}/16$, N=5 1011：$f_{SAMPLING}=f_{DTS}/16$, N=6 1100：$f_{SAMPLING}=f_{DTS}/16$, N=8 1101：$f_{SAMPLING}=f_{DTS}/32$, N=5 1110：$f_{SAMPLING}=f_{DTS}/32$, N=6 1111：$f_{SAMPLING}=f_{DTS}/32$, N=8
3:2	IC3PSC	R/W	0x0	输入捕获 3 预分频： 00：无分频 01：每 2 个事件输入产生一次捕获 10：每 4 个事件输入产生一次捕获 11：每 8 个事件输入产生一次捕获 IC3PSC 寄存器在 CC3E=0 时复位
1:0	CC3S	R/W	0x0	捕获/比较 3 通道选择： 00：CC3 通道配置为输出 01：CC3 通道配置为输入，IC3 映射到 TI3 10：CC3 通道配置为输入，IC3 映射到 TI4 11：CC3 通道配置为输入，IC3 映射到 TRC 注意：CC1S 仅在通道关闭时（CC1E=0）可以写

18.7.9 捕获/比较使能寄存器（TIM_CCER）

偏移地址：0x20

复位值：0x0000 0000

	名称	属性	复位值	描述
31:14	RSV	-	-	保留
13	CC4P	R/W	0x0	捕获/比较 4 输出极性： CC4 通道配置为输出时 0：OC4 为 OC4REF 1：OC4 为 OC4REF 反转 CC4 通道配置为输入时 0：非取反模式–捕获在 IC4 的上升沿进行 1：取反模式–捕获在 IC4 的下降沿进行

	名称	属性	复位值	描述
12	CC4E	R/W	0x0	捕获/比较 4 输出使能： CC4 通道配置为输出时 0：OC4 不 active 1：OC4 active CC4 通道配置为输入时 0：关闭捕获功能 1：使能捕获功能
11	CC3NP	R/W	0x0	捕获/比较 3 互补输出极性： **CC3 通道配置为输出时** 0：OC3N 为 OC3REF 反转 1：OC3N 为 OC3REF **CC3 通道配置为输入时** 与 CC3P 配合使用
10	CC3NE	R/W	0x0	捕获/比较 3 互补输出使能： 0：关闭：OC3N 未激活 1：开启：输出 OC3N 信号
9	CC3P	R/W	0x0	捕获/比较 3 输出极性： **CC3 通道配置为输出时** 0：OC3 为 OC3REF 1：OC3 为 OC3REF 反转 **CC3 通道配置为输入时** 与 CC3NP 配合，{CC3NP,CC3P}选择 IC3 的极性： 00：非取反模式–捕获在 IC3 的上升沿进行 01：取反模式–捕获在 IC3 的下降沿进行 10：保留 11：非取反模式–捕获在 IC3 的上升沿和下降沿进行，不能在编码器模式下选择此模式
8	CC3E	R/W	0x0	捕获/比较 3 输出使能，参考 CC1E
7	CC2NP	R/W	0x0	捕获/比较 2 互补输出极性： CC2 通道配置为输出时 0：OC2N 为 OC2REF 反转 1：OC2N 为 OC2REF CC2 通道配置为输入时 与 CC2P 配合使用
6	CC2NE	R/W	0x0	捕获/比较 2 互补输出使能 0：关闭：OC2N 未激活。 1：开启：输出 OC2N 信号
5	CC2P	R/W	0x0	捕获/比较 2 输出极性： CC2 通道配置为输出时 0：OC2 为 OC2REF 1：OC2 为 OC2REF 反转 CC2 通道配置为输入时 与 CC2NP 配合，{CC2NP,CC2P}选择 IC2 的极性： 00：非取反模式–捕获在 IC2 的上升沿进行 01：取反模式–捕获在 IC2 的下降沿进行 10：保留 11：非取反模式–捕获在 IC2 的上升沿和下降沿进行，不能在编码器模式下选择此模式

	名称	属性	复位值	描述
4	CC2E	R/W	0x0	捕获/比较 2 输出使能： CC2 通道配置为输出时 0：OC2 不 active 1：OC2 active CC2 通道配置为输入时 0：关闭捕获功能 1：使能捕获功能
3	CC1NP	R/W	0x0	捕获/比较 1 互补输出极性： CC1 通道配置为输出时 0：OC1N 为 OC1REF 反转 1：OC1N 为 OC1REF CC1 通道配置为输入时 与 CC1P 配合使用，选择 IC1/IC2 的极性
2	CC1NE	R/W	0x0	捕获/比较 1 互补输出使能： 0：关闭：OC1N 未激活 1：开启：输出 OC1N 信号
1	CC1P	R/W	0x0	捕获/比较 1 输出极性： CC1 通道配置为输出时 0：OC1 为 OC1REF 1：OC1 为 OC1REF 反转 CC1 通道配置为输入时 与 CC1NP 配合，{CC1NP,CC1P}选择 IC1 的极性： 00：非取反模式–捕获在 IC1 的上升沿进行 01：取反模式–捕获在 IC1 的下降沿进行 10：保留 11：非取反模式–捕获在 IC1 的上升沿和下降沿进行，不能在编码器模式下选择此模式
0	CC1E	R/W	0x0	捕获/比较 1 输出使能： CC1 通道配置为输出时 0：OC1 不 active 1：OC1 active CC1 通道配置为输入时 0：关闭捕获功能 1：使能捕获功能

如下表所示，其中 MOE 为定时器总输出使能位，OSSI 为 IDLE 状态（MOE=0）下的 off_state 选择位，OSSR 为 RUN 状态（MOE=1）下的 off_state 选择位 Off-state：

表 18-6：控制寄存器和互补输出通道的状态对应表

控制寄存器					输出状态	
MOE	**OSSI**	**OSSR**	**CCxE**	**CCxNE**	**OCx 输出状态**	**OCxN 输出状态**
1	X	0	0	0	输出关闭（不由 TIM 驱动） OCx=0,OCx_EN=0	输出关闭（不由 TIM 驱动） OCxN=0,OCxN_EN=0
		0	0	1	输出关闭（不由 TIM 驱动） OCx=0,OCx_EN=0	OCxREF + Polarity OCxN=OCxREF xor CCxNP, OCxN_EN=1

控制寄存器					输出状态	
MOE	**OSSI**	**OSSR**	**CCxE**	**CCxNE**	**OCx 输出状态**	**OCxN 输出状态**
		0	1	0	OCxREF + Polarity OCx=OCxREF xor CCxP, OCx_EN=1	Output disabled (not driven by the timer) OCxN=0, OCxN_EN=0
		0	1	1	OCREF + Polarity + dead-time OCx_EN=1	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN=1
		1	0	0	Output disabled (not driven by the timer) OCx=CCxP, OCx_EN=0	Output disabled (not driven by the timer) OCxN=CCxNP, OCxN_EN=0
		1	0	1	Off-state (output enabled with inactive state) OCx=CCxP OCx_EN=1	OCxREF + Polarity OCxN=OCxREF xor CCxNP OCxN_EN=1
		1	1	0	OCxREF + Polarity OCx=OCxREF xor CCxP OCx_EN=1	Off-state (output enabled with inactive state) OCxN=CCxNP, OCxN_EN=1
		1	1	1	OCREF + Polarity + dead-time OCx_EN=1	Complementary to OCREF (not OCREF) + Polarity + dead-time OCxN_EN=1
0	0	X	0	0	输出关闭（不由 TIM 驱动） OCx=CCxP, OCx_EN=0	输出关闭（不由 TIM 驱动） OCxN=CCxNP, OCxN_EN=0
	0		0	1	输出关闭（不由 TIM 驱动） 如果无时钟：OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0 如果有时钟：经过死区时间后 OCx=OISx, OCxN=OISxN	
	0		1	0		
	0		1	1		
	1		0	0	输出关闭（不由 TIM 驱动） OCx=CCxP, OCx_EN=0	输出关闭（不由 TIM 驱动） OCxN=CCxNP, OCxN_EN=0
	1		0	1	Off-state（输出使能，inactive 输出） 如果无时钟：OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1 如果有时钟：经过死区时间后 OCx=OISx, OCxN=OISxN	

18.7.10 计数值寄存器（TIM_CNT）

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CNT	R/W	0x0	计数器值

18.7.11 预分频寄存器（TIM_PSC）

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	PSC	R/W	0x0	计数器时钟（CK_CNT）预分频值： $f_{CK\_CNT}=f_{CK\_PSC}/(PSC[15:0]+1)$ 这是一个 preload 寄存器，在 update 事件发生时其内容被载入 shadow 寄存器

18.7.12 自动重载寄存器（TIM_ARR）

偏移地址：0x2C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	ARR	R/W	0x0	计数溢出时的自动重载值。 这是一个 preload 寄存器，在 update 事件发生时其内容被载入 shadow 寄存器

18.7.13 重复计数寄存器（TIM_RCR）

偏移地址：0x30

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	REP	R/W	0x0	重复计数值。REP 不为 0 时，每次 update 条件发生时 REP 递减，当 REP=0 时触发 update 事件

18.7.14 捕获/比较寄存器 1（TIM_CCR1）

偏移地址：0x34

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CCR1	R/W	0x0	捕获/比较通道 1 寄存器： **如果通道 1 配置为输出：** 这是一个 preload 寄存器，其内容被载入 shadow 寄存器后用于与计数器比较产生 OC1 输出 **如果通道 1 配置为输入：** CCR1 保存最近一次输入捕获事件发生时的计数器值，此时 CCR1 为只读

18.7.15 捕获/比较寄存器 2（TIM_CCR2）

偏移地址：0x38

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CCR2	R/W	0x0	捕获/比较通道 2 寄存器： **如果通道 2 配置为输出：** 这是一个 preload 寄存器，其内容被载入 shadow 寄存器后用于与计数器比较产生 OC2 输出 **如果通道 2 配置为输入：** CCR2 保存最近一次输入捕获事件发生时的计数器值，此时 CCR2 为只读

18.7.16 捕获/比较寄存器 3（TIM_CCR3）

偏移地址：0x3C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CCR3	R/W	0x0	捕获/比较通道 3 寄存器： **如果通道 3 配置为输出：** 这是一个 preload 寄存器，其内容被载入 shadow 寄存器后用于与计数器比较产生 OC3 输出 **如果通道 3 配置为输入：** CCR3 保存最近一次输入捕获事件发生时的计数器值，此时 CCR3 为只读

18.7.17 捕获/比较寄存器 4（TIM_CCR4）

偏移地址：0x40

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CCR4	R/W	0x0	捕获/比较通道 4 寄存器： **如果通道 4 配置为输出：** 这是一个 preload 寄存器，其内容被载入 shadow 寄存器后用于与计数器比较产生 OC4 输出 **如果通道 4 配置为输入：** CCR4 保存最近一次输入捕获事件发生时的计数器值，此时 CCR4 为只读

18.7.18 刹车和死区控制寄存器（TIM_BDTR）

偏移地址：0x44

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	MOE	R/W	0x0	输出使能主控： 此寄存器控制所有通道的输出使能，每个通道独立的输出使能还需要 CCxE 和 CCxNE 来控制。MOE 由软件置位，或者在 AOE=1 的情况下硬件触发自动置位。当刹车输入有效时，MOE 被硬件异步清零。 0：关闭 OC 和 OCN 输出，具体 IO 输出状态由 OSSI 决定 1：使能 OC 和 OCN 输出（仍需各个通道的 CCxE 和 CCxNE 状态来决定是否输出）
14	AOE	R/W	0x0	自动输出使能： 0：MOE 仅能由软件置位 1：MOE 可以软件置位，或者由 update 事件自动置位
13	BKP	R/W	0x0	刹车极性： 0：刹车输入为低电平有效 1：刹车输入为高电平有效
12	BKE	R/W	0x0	刹车使能： 0：禁止刹车输入 1：允许刹车输入
11	OSSR	R/W	0x0	运行状态（MOE=1）下的输出关闭状态（CCx(N)E=0 的通道）： 仅在 MOE=1 的情况下，针对使能了互补输出的通道有效。 0：输出通道不使能时，OC 和 OCN 不驱动 GPIO，OCxN 总是驱动 IO 1：输出通道不使能时，OC 和 OCN 驱动 GPIO 为“无效状态”（指 CR2 中的 OIS） *（如果 cfg_timx_break_ossi0_disout=1/TIMCFGR[0]/[1]=0，会没有效果，并且会直接使 OC4 不驱动）

位	名称	属性	复位值	描述
10	OSSI	R/W	0x0	IDLE 状态（MOE=0）下的输出关闭状态（CCx(N)E=0 的通道）： 仅在 MOE=0 的情况下，针对输出通道有效。 0：输出通道不使能时，OC 和 OCN 不驱动 GPIO。 ● 如果 cfg_timx_break_ossi0_disout=0/TIMCFGR[0]/[1][12]=1，会一直驱动 ● OCxN 总是驱动 IO 1：输出通道不使能时，OC 和 OCN 先驱动空闲状态，待死区时间结束后，驱动“无效状态”（指 CR2 中的 OIS）。 *(MOE=0 会使 OCx(N) =（|CCx(N)P）&&OISx(N)，OC4=OIS4)
9:8	LOCK	R/W	0x0	寄存器写保护配置： 00：无写保护 01：保护等级 1 – DTG, OISx, OISxN, BKE, BKP, AOE 不能改写 10：保护等级 2 –在等级 1 基础上，CCxP, CCxNP, OSSR, OSSI 不能改写 11：保护等级 3 –在等级 2 基础上，OCxM, OcxPE 在相应通道配置为输出时不能改写 注意：LOCK 寄存器在被写入任意值之后无法再改写，写保护后的寄存器只有在 TIM 模块被复位后才能重新写入。
7:0	DTG	R/W	0x0	死区时间插入，用于配置互补输出插入的死区时间长度 DTG[7:5]=0xx：DT=DTG[7:0] * t_{DTS} DTG[7:5]=10x：DT=(64+DTG[5:0]) * 2 * t_{DTS} DTG[7:5]=110：DT=(32+DTG[4:0]) * 8 * t_{DTS} DTG[7:5]=111：DT=(32+DTG[4:0]) * 16 * t_{DTS}

18.7.19 DMA 控制寄存器（TIM_DCR）

偏移地址：0x48

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12:8	DBL	R/W	0x0	DMA Burst 长度： 对 TIM_DMAR 寄存器的读写将触发 burst DMA 操作，burst 长度为 1~18 00000：长度=1 00001：长度=2 …… 10001：长度=18 其他：无效值，禁止写入
7:5	RSV	-	-	保留

位	名称	属性	复位值	描述
4:0	DBA	R/W	0x0	DMA 基地址，定义指向寄存器的偏移地址： 00000：TIM_CR1 00001：TIM_CR2 00010：TIM_SMCR …… 注意：当 DBA+DBL 超出了 TIM 寄存器地址范围，则实际 burst 传输到 TIM 最高寄存器地址后自动停止，即 burst 长度会缩短。

18.7.20 DMA 访问寄存器（TIM_DMAR）

偏移地址：0x4C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	DMAR	R/W	0x0	DMA burst 访问寄存器 在使用 DMA burst 传输时，将 DMA 通道外设地址设置到 TIM_DMAR，对此寄存器的访问会指向 TIM_DCR 中指定的寄存器，TIM 会根据 DBL 的值产生多次 DMA 请求

18.8 使用流程

18.8.1 定时计数模式

1. RCM 模块中使能 TIMx 时钟。
2. 配置 TIM_CR1[4]，设置计数方向。
3. 配置 TIM_CR1[7]为 1，使能 Auto-reload 预装载。
4. 配置 TIM_PSC[15:0]，设置预分频值。
5. 配置 TIM_ARR[31:0]，设置自动重载值。
6. 配置 TIM_RCR[7:0]为 0，不进行重复计数。
7. 配置 TIM_CR1[2]为 1，设置仅计数器上溢出或下溢出会产生 update 中断或 DMA 请求。
8. 配置 TIM_CR1[1]为 0，使能 update 事件。
9. 配置 TIM_EGR[0]为 1，软件置位 UG 时会重新初始化计数器并更新 shadow 寄存器，预分频计数器被清零。
10. 配置 TIM_CR1[0]为 1，使能计数器。
11. 配置 TIM_DIER[0]为 1，允许 Update 事件中断。

18.8.2 PWM 模式

1. 配置 GPIO 复用，RCM 模块中使能 TIMx 时钟。
2. 配置 TIM_CR1[4]，设置计数方向。
3. 配置 TIM_CR1[7]为 1，使能 Auto-reload 预装载。
4. 配置 TIM_PSC[15:0]，设置预分频值。
5. 配置 TIM_ARR[31:0]，设置自动重载值。
6. 配置 TIM_RCR[7:0]为 0，不进行重复计数。
7. 根据输出通道配置 TIM_CCMRx[1:0/9:8]为 0，设置通道 x 为输出。
8. 配置 TIM_CCMRx 寄存器的 OCxM[6:4/14:12]，设置为 PWM 模式 1//2。
9. 配置 TIM_CCER[1/3/5/7/9/11/13]，设置输出极性。
10. 配置 TIM_CCER[0/4/8/12]为 1，通道 x 输出使能。
11. 配置 TIM_BDTR[15]为 1，该位为输出使能主控，使能 OC 和 OCN 输出。
12. 配置 TIM_CR1[2]为 1，设置仅计数器上溢出或下溢出会产生 update 中断或 DMA 请求。
13. 配置 TIM_CR1[1]为 0，使能 update 事件。
14. 配置 TIM_EGR[0]为 1，软件置位 TIM_EGR[0]时会重新初始化计数器并更新 shadow 寄存器，预分频计数器被清零。
15. 配置 TIM_CR1[0]为 1，使能计数器。
16. 配置 TIM_DIER[0]为 1，允许 Update 事件中断。
17. 配置 TIM_CCRx[31:0]，设置通道 x 的比较值。

18.8.3 输入捕获模式

1. 配置 GPIO 复用，RCM 模块中使能 TIMx 时钟。
2. 配置 TIM_CR1[4]，设置计数方向。
3. 配置 TIM_CR1[7]为 1，使能 Auto-reload 预装载。
4. 配置 TIM_PSC[15:0]，设置预分频值。
5. 配置 TIM_ARR[31:0]，设置自动重载值。
6. 配置 TIM_RCR[7:0]为 0，不进行重复计数。
7. 配置 TIM_CCMRx[1:0/9:8]，设置 CCx 通道为输入，并更根据需求映射。
8. 配置 TIM_CCER[1/3/5/7/9/11/13]，设置捕获极性。
9. 配置 TIM_CCMRx[7:4/15:12]，设置采样频率和滤波长度，一般设置为 0 即可。
10. 配置 TIM_CCMRx[3:2/11:10]，设置输入捕获预分频。
11. 配置 TIM_CCER[0/4/8/12]为 1，使能捕获功能。
12. 配置 TIM_EGR[0]为 1，软件置位 TIM_EGR[0]时会重新初始化计数器并更新 shadow 寄存器，

预分频计数器被清零。

13. 配置 TIM_CR1[0]为 1，使能计数器。
14. 配置 TIM_DIER[1/2/3/4]为 1，允许通道 x 捕获中断。

18.8.4 互补输出和死区插入

以 168MHz 举例计算的周期时间：t_{DTS} = 5.95 ns

- DT = (0~127) * 5.95 = 0~755.65 ns，DTG[7:5]=0xx：DT=DTG[7:0] * t_{DTS}。
- DT = (64 + (0~63)) * 2 * 5.95 = 761.6~1511.3 ns，DTG[7:5]=10x：DT=(64+DTG[5:0]) * 2 * t_{DTS}。
- DT = (32 + (0~31)) * 8 * 5.95 = 1523.2~2998.8 ns，DTG[7:5]=110：DT=(32+DTG[4:0]) * 8 * t_{DTS}。
- DT = (32 + (0~31)) * 16 * 5.95 = 3046.4~5997.6 ns，DTG[7:5]=111：DT=(32+DTG[4:0]) * 16 * t_{DTS}。

在初始化 PWM 模式前，补充以下配置：

1. 配置 TIM_BDTR[7:0]，设置互补输出的死区时间长度；
2. 配置 TIM_CCER[2/6/10]为 1，使能互补输出；

18.8.5 刹车功能

在初始化 PWM 模式前，补充以下配置：

1. 配置 TIM_BDTR[11]，设置运行状态下的输出关闭状态。
2. 配置 TIM_BDTR[10]，设置空闲状态下的输出关闭状态。
3. 配置 TIM_BDTR[13]，设置刹车极性。
4. 配置 TIM_CCER[1/5/9/13]，设置 OCx 的输出极性。
5. 配置 TIM_CCER[3/7/11]，设置 OCxN 的输出极性。
6. 配置 TIM_CR2[8/10/12/14]，设置 OCx 的空闲输出状态。
7. 配置 TIM_CR2[9/11/13]，设置 OCxN 的空闲输出状态。
8. 配置 TIM_BDTR[14]，设置 TIM_BDTR [15]置位方式。
9. 配置 TIM_BDTR[12]为 1，允许刹车输入。

18.8.6 编码器接口模式

1. 配置 GPIO 复用，RCM 模块中使能 TIMx 时钟。
2. 配置 TIM_CR1[4]，设置计数方向。
3. 配置 TIM_CR1[7]为 1，使能 Auto-reload 预装载。

4. 配置 TIM_PSC[15:0]，设置预分频值。
5. 配置 TIM_ARR[31:0]，设置自动重载值。
6. 配置 TIM_RCR[7:0]为 0，不进行重复计数。
7. 配置 TIM_CCMR1[1:0]为 1，设置 CC1 通道为输入，IC1 映射到 TI1。
8. 配置 TIM_CCMR1[9:8]为 1，设置 CC2 通道为输入，IC2 映射到 TI2。
9. 配置 TIM_CCER[1]和 TIM_CCER[3]，设置捕获极性。
10. 配置 TIM_CCER[1]和 TIM_CCER[7]，设置捕获极性。
11. 配置 TIM_CCMR1[7:4]，设置采样频率和滤波长度，一般设置为 0 即可。
12. 配置 TIM_CCMR1[15:12]，设置采样频率和滤波长度，一般设置为 0 即可。
13. 配置 TIM_SMCR[2:0]，设置 Encoder 模式 1/2/3。
14. 配置 TIM_CCER[0]为 1，使能通道 1 捕获功能。
15. 配置 TIM_CCER[4]为 1，使能通道 2 捕获功能。
16. 配置 TIM_EGR[0]为 1，软件置位 TIM_EGR[0]时会重新初始化计数器并更新 shadow 寄存器，预分频计数器被清零。
17. 配置 TIM_CR1[0]为 1，使能计数器。
18. 配置 TIM_DIER[1]为 1，允许通道 1 捕获中断。

18.8.7 DMA 模式

- **输入捕获模式下，TIMx 的通道捕获值通过 DMA 传输到 SRAM：**

  1. 在输入捕获模式中软件置位 TIM_EGR[0]和使能计数器前，补充以下配置。
  2. 配置 TIM_DCR[12:8]，设置 DMA Burst 长度。
  3. 配置 TIM_DCR[4:0]，设置 DMA 基地址，一般该处的基地址选择相应捕获通道对应的捕获/比较寄存器。
  4. 配置 TIM_DIER[9/10/11/12]为 1，允许 CCx DMA 请求。
  5. 配置 TIM_CR2[3]为 0，发生 CCx 事件时产生 CCxDMA 请求。
  6. DMA 控制器配置请参见“11 DMA 控制器（DMA）”章节。
  7. 开启 DMA 传输后，当通道发生捕获时，DMA 将基地址存放的值传到 SRAM 中。

- **输出比较模式下，SRAM 中的值通过 DMA 传输到 TIMx 的比较寄存器：**

  1. 在 PWM 模式中软件置位 TIM_EGR[0]和使能计数器前，补充以下配置。
  2. 配置 TIM_DCR[12:8]，设置 DMA Burst 长度。
  3. 配置 TIM_DCR[4:0]，设置 DMA 基地址，一般该处的基地址选择相应比较通道对应的捕获/比较寄存器。
  4. 配置 TIM_DIER[9/10/11/12]为 1，允许 CCx DMA 请求。

5. 配置 TIM_CR2[3]为 0，发生 CCx 事件时产生 CCxDMA 请求。
6. DMA 控制器配置请参见“11 DMA 控制器（DMA）”章节。
7. 开启 DMA 传输后，当计数器计数值等于比较值时，DMA 将 SRAM 中的值传到基地址。

19 通用定时器（TIM1~TIM4 & TIM8~TIM13）

19.1 概述

包含一个 32bit 自动重载计数器及一个可编程预分频器。可以支持多种应用，包括如捕获、输出比较、PWM。

19.2 主要特性

- 32bit 向上、向下、双向自动重载计数器
- 16bit 可编程预分频器，支持实时调整计数时钟分频
- 4 个独立通道可用于输入捕获、输出比较、PWM、单脉冲输出
- 支持与其他定时器级联
- 支持在以下事件发生时产生中断或 DMA 事件
  - 计数器上/下溢出，计数器初始化（软件或硬件 trigger）
  - Trigger 事件（计数器启动、停止、初始化、内外部触发）
  - 输入捕获
  - 输出比较
- 支持增量正交编码器和霍尔传感器
- 支持外部时钟和触发输入

19.3 系统框图

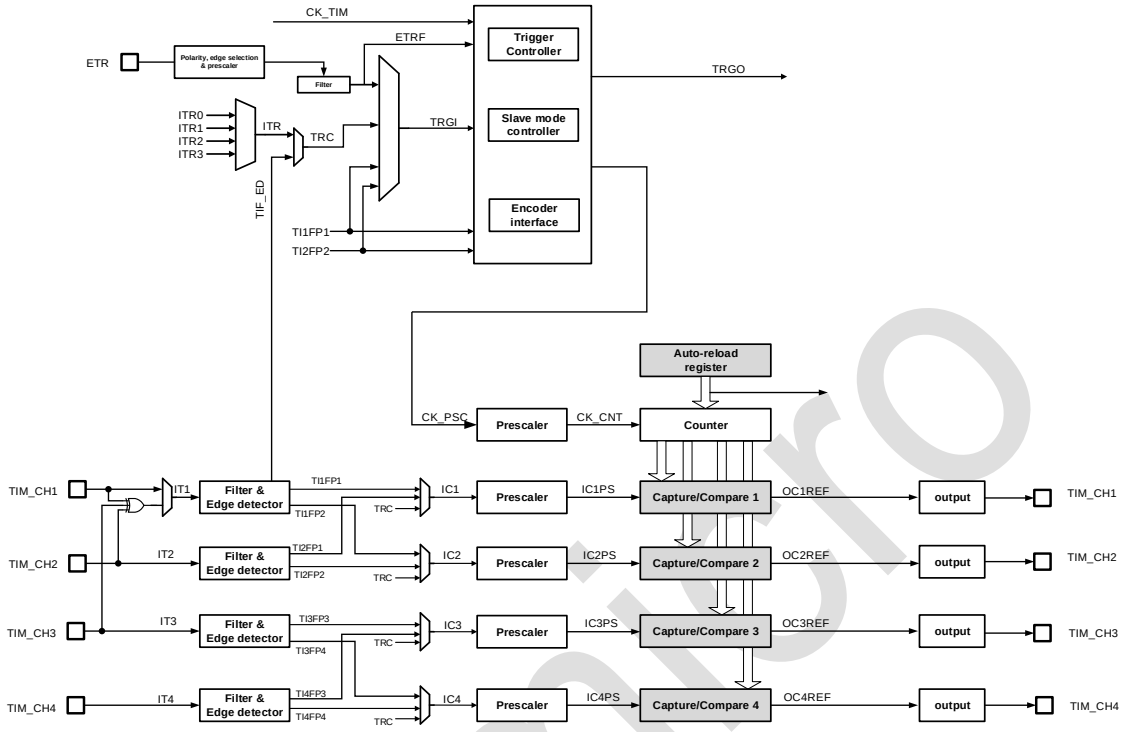


图 19-1：TIM1~TIM4 & TIM8~TIM13 系统框图

19.4 管脚说明

表 19-1：TIM1~TIM4 & TIM8~TIM13 管脚说明

功能管脚	复用管脚	方向	功能描述
TIM1_CH1_ETR	PA0,PA5,PA15	Input/Output	作为外部触发输入 TIM1_ETR 功能，或者通道 1 输入捕获或 PWM 输出功能 TIM1_CH1。TIM1_ETR 和 TIM1_CH1 不能同时使用。
TIM1_CH2	PA1,PB3	Input/Output	通道输入捕获或者 PWM 输出
TIM1_CH3	PA2,PB10	Input/Output	通道输入捕获或者 PWM 输出
TIM1_CH4	PA3,PB11	Input/Output	通道输入捕获或者 PWM 输出
TIM2_ETR	PD2	Input	外部触发输入
TIM2_CH1	PA6,PB4,PC6	Input/Output	通道输入捕获或者 PWM 输出
TIM2_CH2	PA7,PB5,PC7	Input/Output	通道输入捕获或者 PWM 输出
TIM2_CH3	PB0,PC8	Input/Output	通道输入捕获或者 PWM 输出
TIM2_CH4	PB1,PC9	Input/Output	通道输入捕获或者 PWM 输出
TIM3_ETR	PE0	Input	外部触发输入
TIM3_CH1	PB6,PD12	Input/Output	通道输入捕获或者 PWM 输出
TIM3_CH2	PB7,PD13	Input/Output	通道输入捕获或者 PWM 输出

功能管脚	复用管脚	方向	功能描述
TIM3_CH3	PB8,PD14	Input/Output	通道输入捕获或者 PWM 输出
TIM3_CH4	PB9,PD15	Input/Output	通道输入捕获或者 PWM 输出
TIM4_CH1	PA0,PC0	Input/Output	通道输入捕获或者 PWM 输出
TIM4_CH2	PA1,PC1	Input/Output	通道输入捕获或者 PWM 输出
TIM4_CH3	PA2,PC2	Input/Output	通道输入捕获或者 PWM 输出
TIM4_CH4	PA3,PC3	Input/Output	通道输入捕获或者 PWM 输出
TIM8_CH1	PA2,PE5	Input/Output	通道输入捕获或者 PWM 输出
TIM8_CH2	PA3,PE6	Input/Output	通道输入捕获或者 PWM 输出
TIM8_CH3	PE3	Input/Output	通道输入捕获或者 PWM 输出
TIM8_CH4	PE4	Input/Output	通道输入捕获或者 PWM 输出
TIM9_CH1	PB8,PC4	Input/Output	通道输入捕获或者 PWM 输出
TIM9_CH2	PA15,PC5	Input/Output	通道输入捕获或者 PWM 输出
TIM9_CH3	PB2,PC11	Input/Output	通道输入捕获或者 PWM 输出
TIM9_CH4	PC10	Input/Output	通道输入捕获或者 PWM 输出
TIM10_CH1	PB9,PD5	Input/Output	通道输入捕获或者 PWM 输出
TIM10_CH2	PD2	Input/Output	通道输入捕获或者 PWM 输出
TIM10_CH3	PD3,PD6	Input/Output	通道输入捕获或者 PWM 输出
TIM10_CH4	PD4	Input/Output	通道输入捕获或者 PWM 输出
TIM11_CH1	PB14	Input/Output	通道输入捕获或者 PWM 输出
TIM11_CH2	PB15	Input/Output	通道输入捕获或者 PWM 输出
TIM11_CH3	PB12	Input/Output	通道输入捕获或者 PWM 输出
TIM11_CH4	PB13	Input/Output	通道输入捕获或者 PWM 输出
TIM12_CH1	PA6,PD7,PB2	Input/Output	通道输入捕获或者 PWM 输出
TIM12_CH2	PA5,PD8,PB3	Input/Output	通道输入捕获或者 PWM 输出
TIM12_CH3	PB0,PD9	Input/Output	通道输入捕获或者 PWM 输出
TIM12_CH4	PB1,PD10	Input/Output	通道输入捕获或者 PWM 输出
TIM13_CH1	PA7,PE12	Input/Output	通道输入捕获或者 PWM 输出
TIM13_CH2	PE7	Input/Output	通道输入捕获或者 PWM 输出
TIM13_CH3	PE8	Input/Output	通道输入捕获或者 PWM 输出
TIM13_CH4	PE10	Input/Output	通道输入捕获或者 PWM 输出

19.5 功能描述

19.5.1 定时单元

定时单元由一个 32 位计数器和自动重载寄存器组成。计数器可以向上、向下或双向计数。计数时钟可以通过 16 位预分频器对时钟进行分频后得到。

计数器、自动重载寄存器预分频寄存器都可以由软件改写或读取，即使在计数器正在运行时也是如此。

定时单元包含如下寄存器：

- 计数器（TIM_CNT）
- 预分频寄存器（TIM_PSC）
- 自动重载寄存器（TIM_ARR）

ARR 包含预装载功能，该功能通过 ARPE（Auto Reload Preload Enable）寄存器控制。当 ARPE=0 时，对 ARR 寄存器执行写入，写入数据将直接传入到影子寄存器；当 ARPE=1 时，对 ARR 寄存器执行写入的数据在 update event（TIM_CNT 上溢出或者下溢出）发生时，传送到影子寄存器。软件也可以通过寄存器操作主动触发 ARR 更新（UEV）。

TIM_CNT 工作时钟由 TIM_PSC 产生的分频时钟驱动，只有在计数器使能寄存器（CEN）置位时，CNT 才开始计数。当 CNT=ARR 时，本轮计数结束，发送 update event。

TIM_PSC 是一个同步预分频器，能够对时钟进行 1~65536 分频。PSC 寄存器同样被缓存，改写 PSC 实际不改写影子寄存器，只有当新的 update event 到来时，才会从 PSC 更新至影子寄存器。因此在 CNT 计数过程中，软件可以实时改写 PSC，而新的预分频比将在下一更新事件发生时被采用。

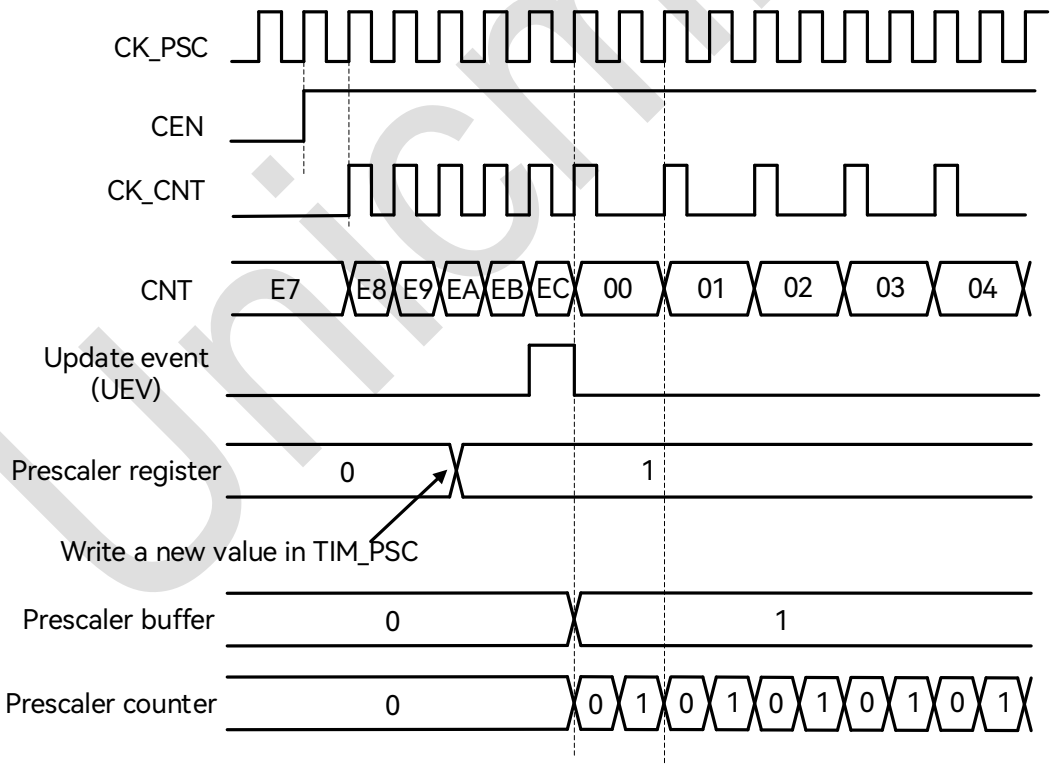


图 19-2：预分频从 1 变为 2 的波形

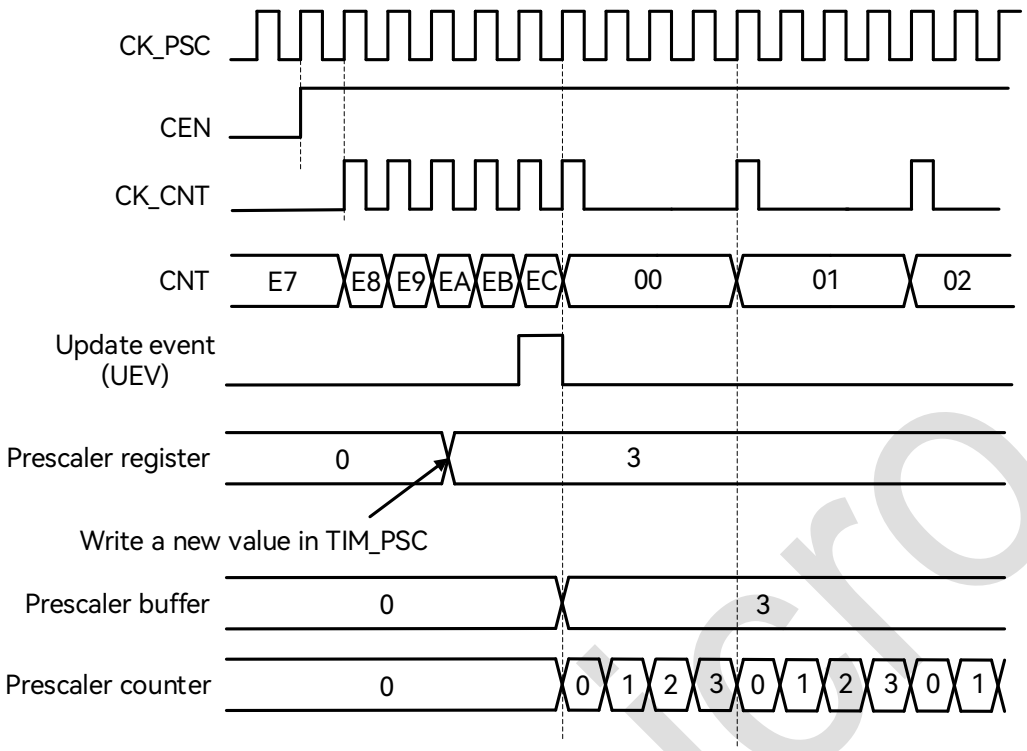


图 19-3：预分频从 1 变为 4 的波形

19.5.2 定时器工作模式

定时器支持向上计数、向下计数和中心计数模式。

19.5.2.1 向上计数

此模式中，计数器使能后从 0 开始计数，直到 CNT=ARR，产生溢出事件，然后重新从 0 开始计数。

如果使能了重复计数功能，则计数器按照 RCR 的定义重复上述过程若干次（RCR+1），才会产生溢出事件。

软件可以通过设置 UG 寄存器直接触发 update event，此时 CNT 和预分频计数器自动清零。设置 UG 寄存器是否触发 UIF（Update Interrupt Flag）中断标志置位由 URS 寄存器的设置决定。

通过设置 UDIS 寄存器可以禁止 update event，这样可以避免将 preload 寄存器中的值更新到工作寄存器中。

当 update event 发生时，以下寄存器被更新，并且 UIF 置位：

- ARR影子寄存器被更新为TIM_ARR内容
- PSC影子寄存器被更新为TIM_PSC内容

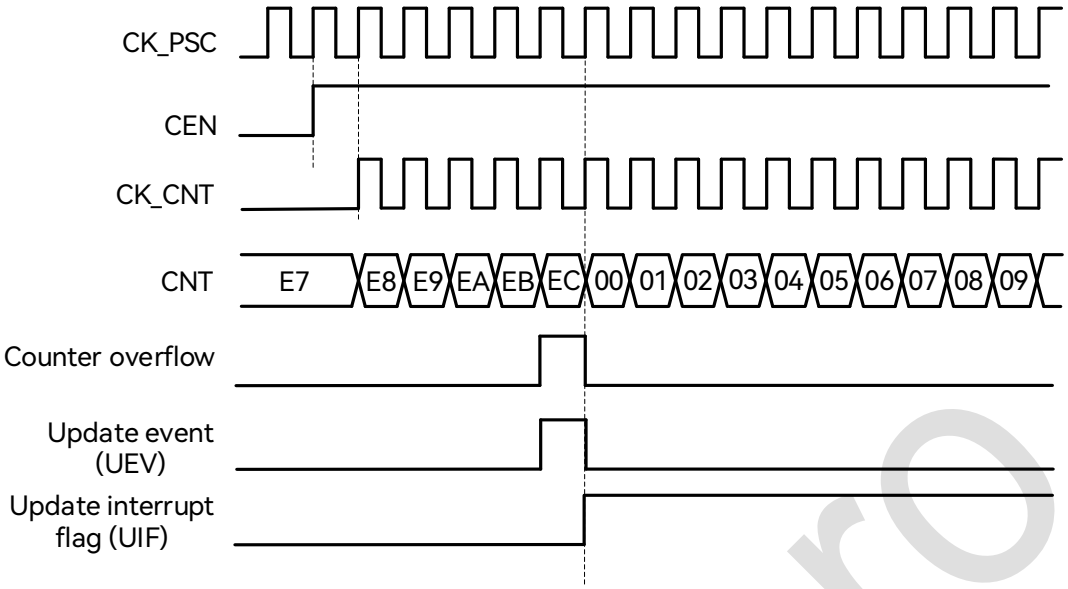


图 19-4：向上计数波形，内部时钟不分频

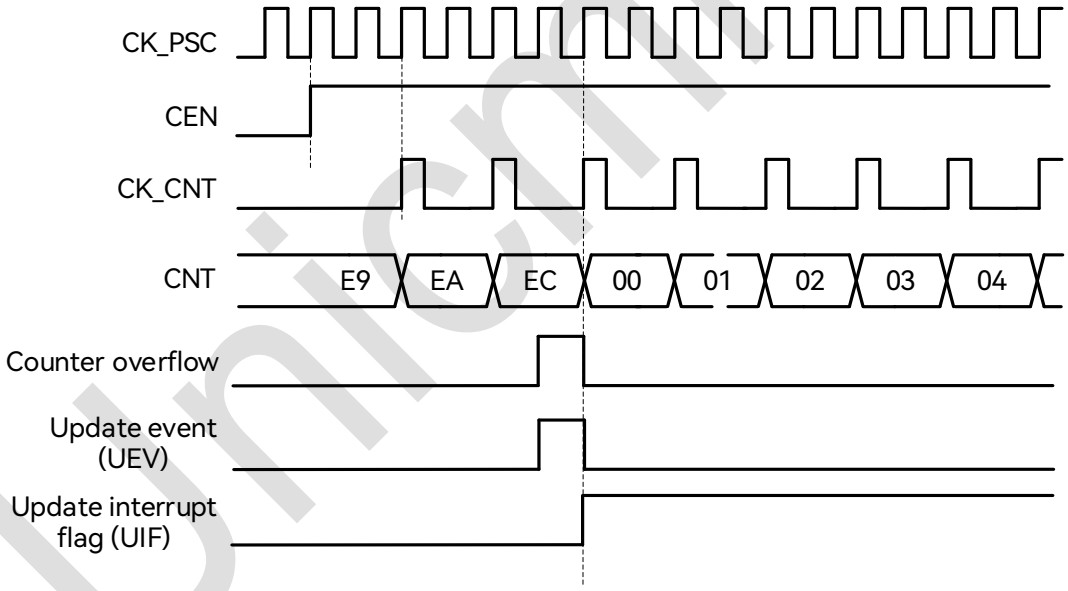


图 19-5：向上计数波形，内部时钟 2 分频

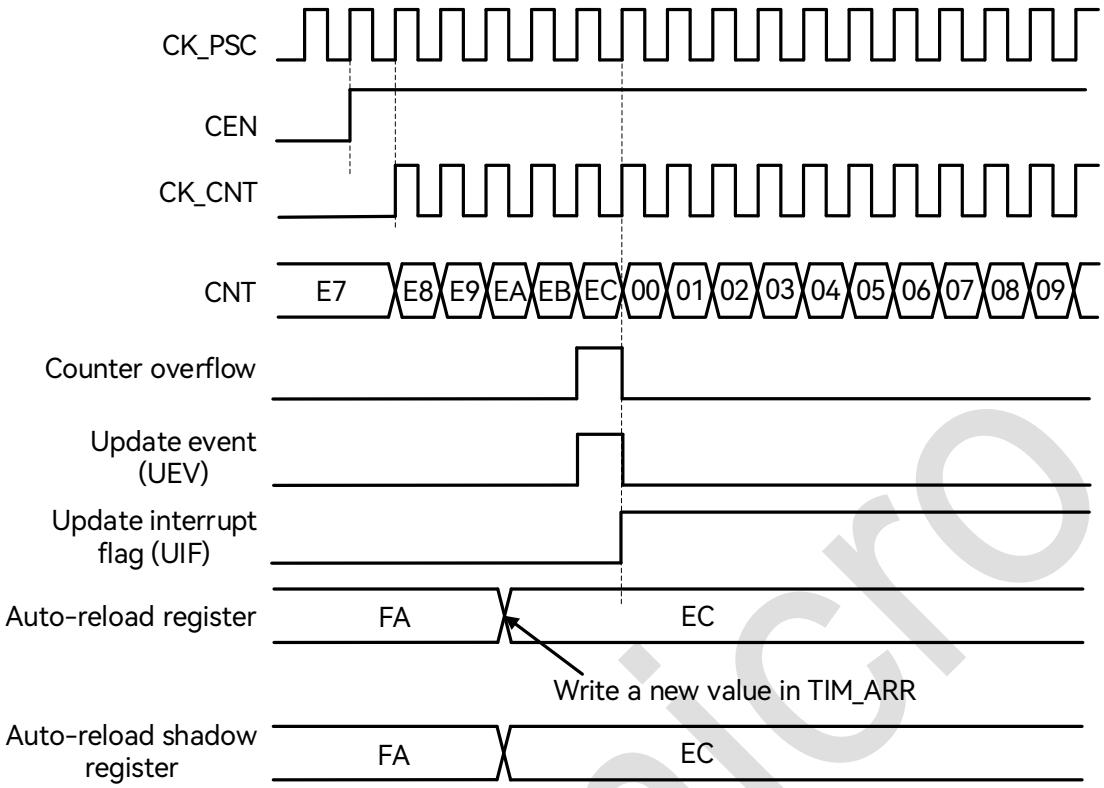


图 19-6：ARPE=0（TIM_ARR 没有预装载）时的更新事件

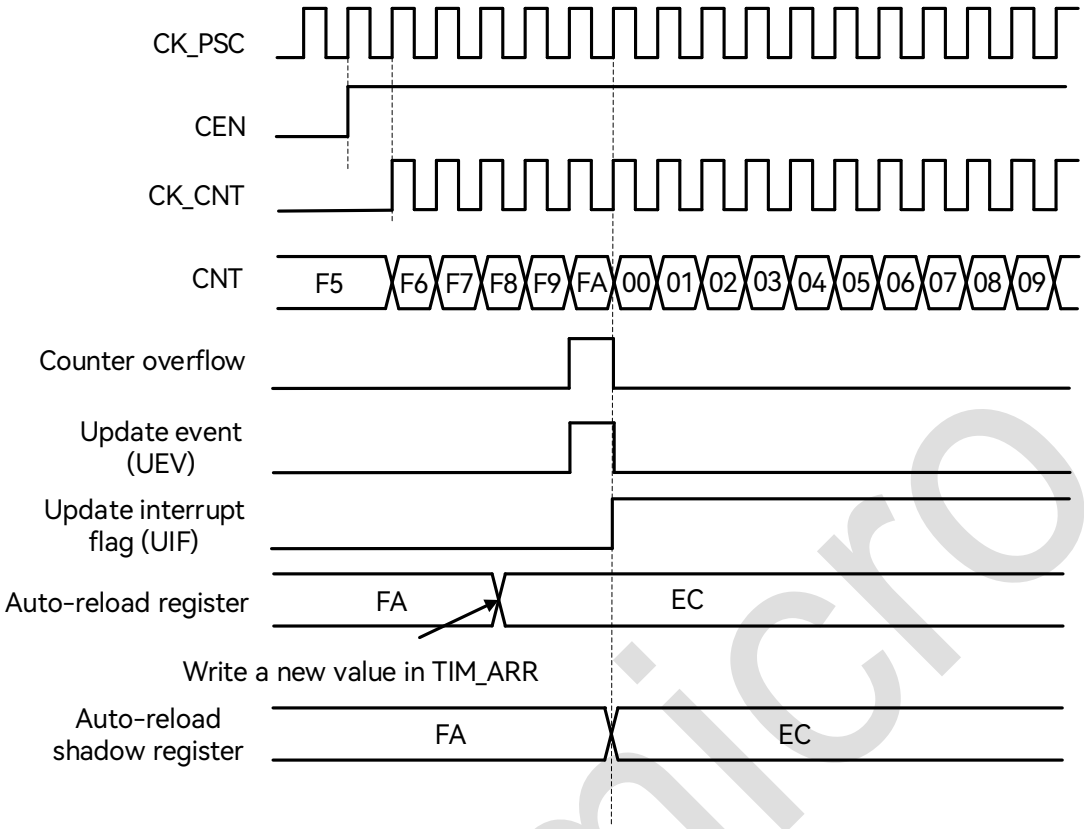


图 19-7：ARPE=1（TIM_ARR 预装载）时的更新事件

19.5.2.2 向下计数

向下计数模式中，计数器从 ARR 值开始递减，到 0 后产生下溢出事件，并且重新从 ARR 开始计数。

如果使能了重复计数功能，则计数器按照 RCR 的定义重复上述过程若干次（RCR+1），才会产生溢出事件。

软件可以通过设置 UG 寄存器直接触发 update event，此时 CNT 和预分频计数器自动清零。设置 UG 寄存器是否触发 UIF（Update Interrupt Flag）中断标志置位由 URS 寄存器的设置决定。

通过设置 UDIS 寄存器可以禁止 update event，这样可以避免将 preload 寄存器中的值更新到工作寄存器中。

当 update event 发生时，以下寄存器被更新，并且 UIF 置位：

- ARR影子寄存器被更新为TIM_ARR内容
- PSC影子寄存器被更新为TIM_PSC内容

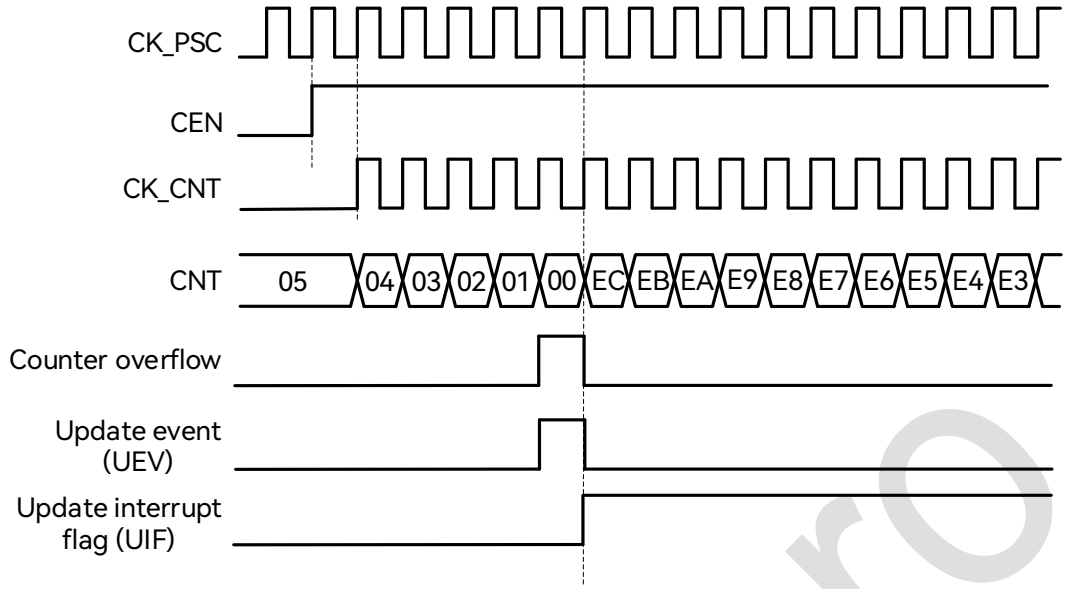


图 19-8：向下计数，内部时钟不分频

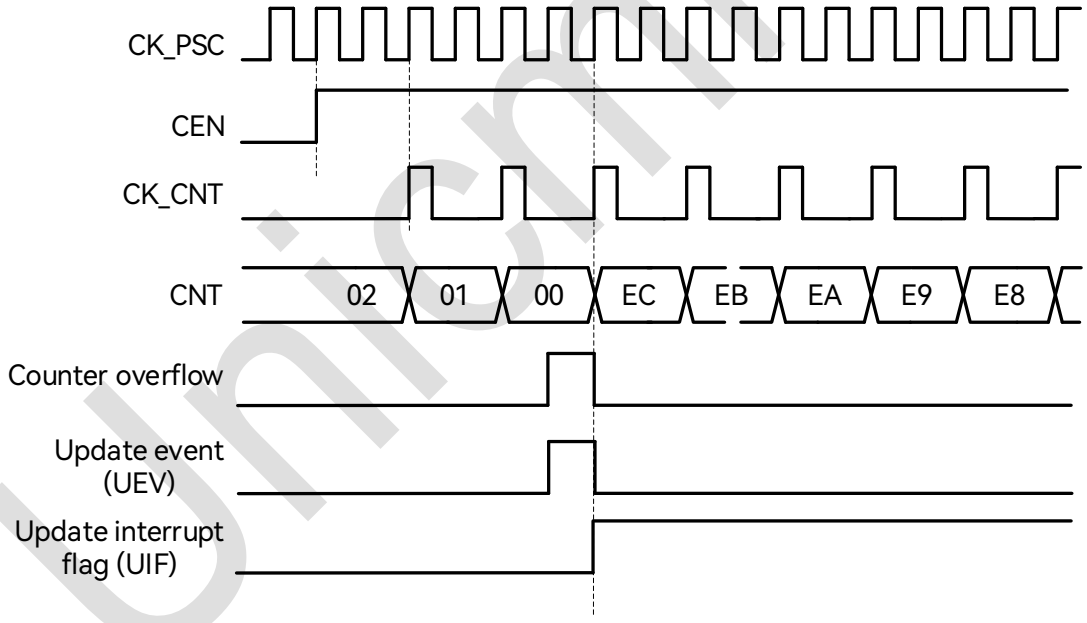


图 19-9：向下计数，内部时钟 2 分频

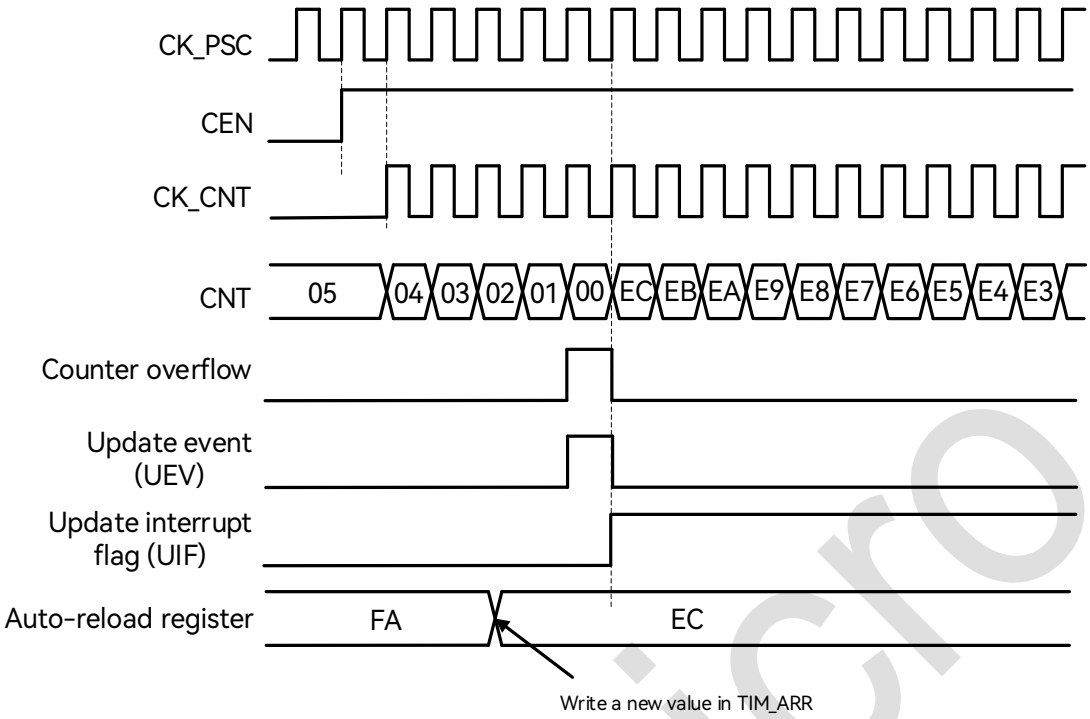


图 19-10：向下计数，内部时钟 2 分频

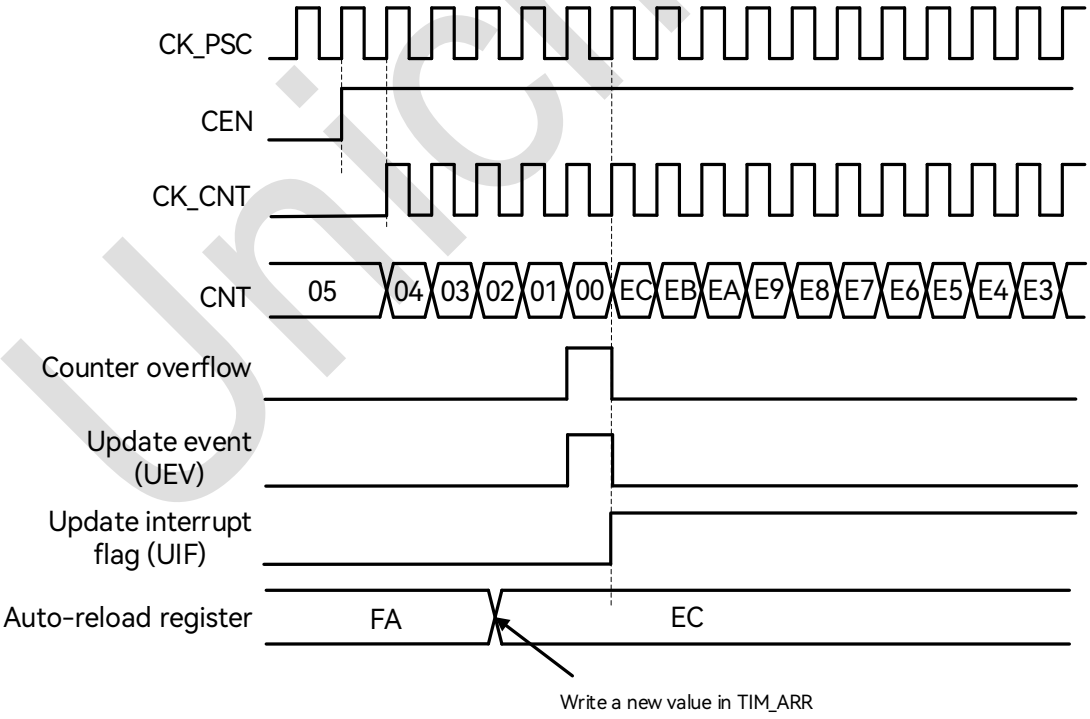


图 19-11：向下计数，不使用重复计数时的更新事件

19.5.2.3 中心对齐计数

在中心对齐模式下，计数器从 0 开始向上计数，到 ARR-1 产生上溢出事件，然后从 ARR 开始向下计数到 1，产生下溢出事件，再从 0 重新开始向上计数。

CMS[1:0]寄存器用于使能中心对齐模式，并选择中心对齐模式下的输出比较工作方式。当 CMS!=00 时为中心对齐计数，当 CMS=01 时，输出比较功能仅在向下计数时有效，当 CMS=10 时，输出比较功能仅在向上计数时有效，当 CMS=11 时，输出比较功能在上下计数时都有效。

中心对齐模式下，DIR 寄存器无法由软件改写，而是随着计数方向变化硬件自动更新，表示当前计数方向。

计数器在 overflow 和 underflow 的事件上都会更新 ARR、PSC 的影子寄存器。

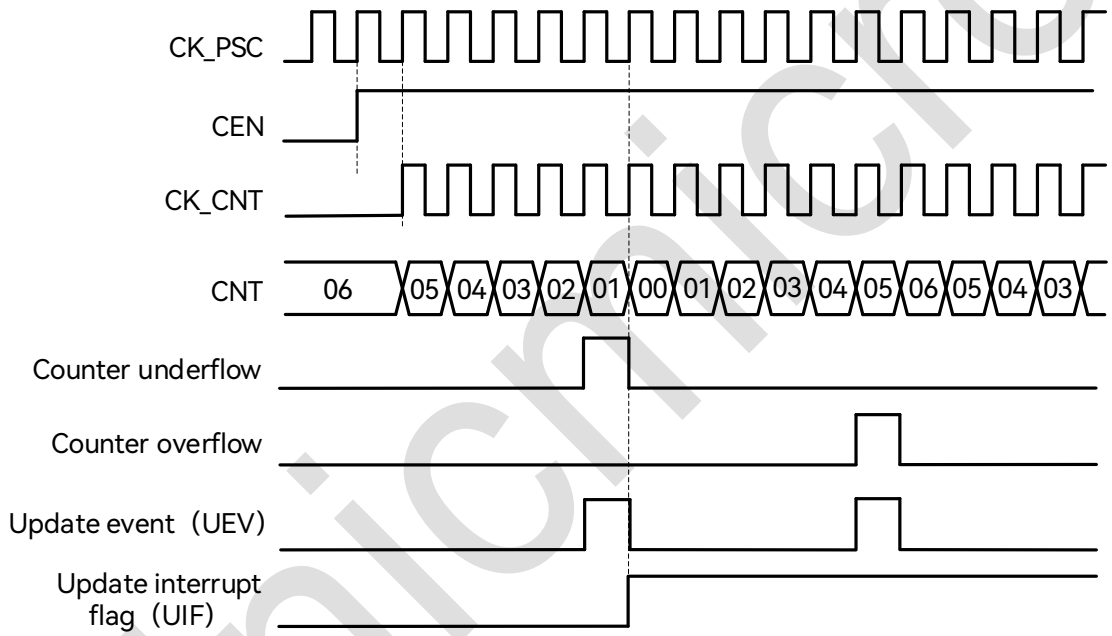


图 19-12：中心对齐计数器时序图，TIM_PCS=0，TIM_ARR=0x6

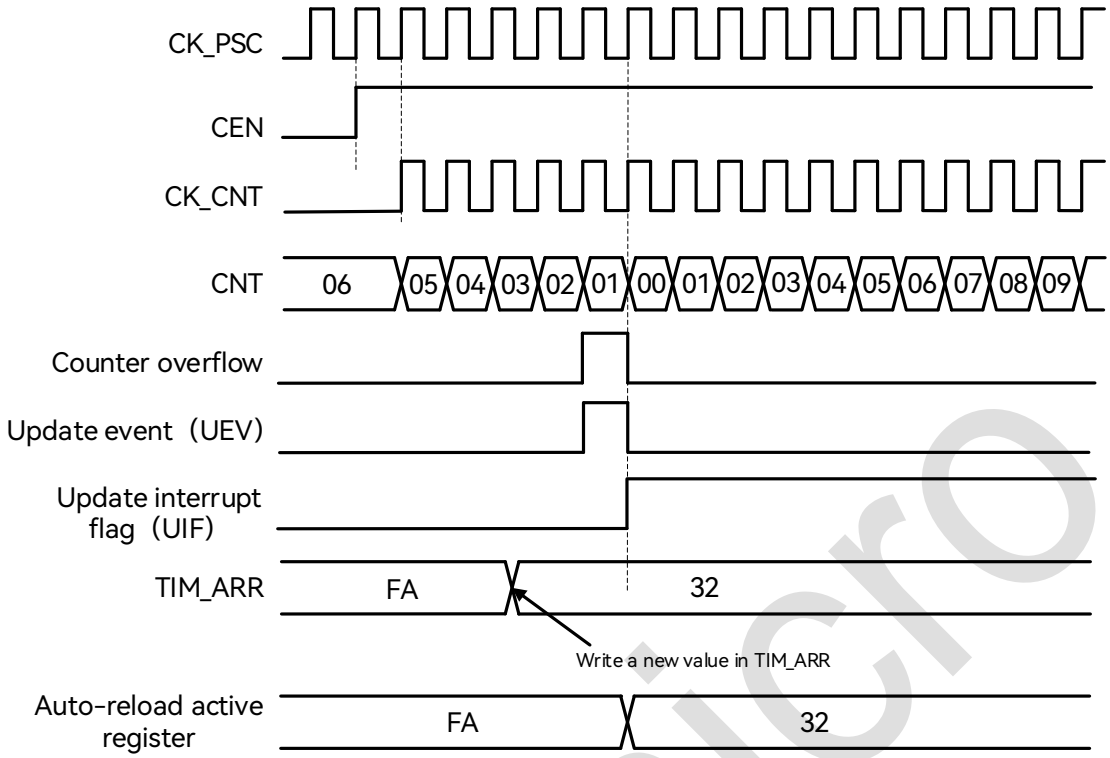


图 19-13：计数器时序图，ARPE=1 时的更新事件(计数器下溢)

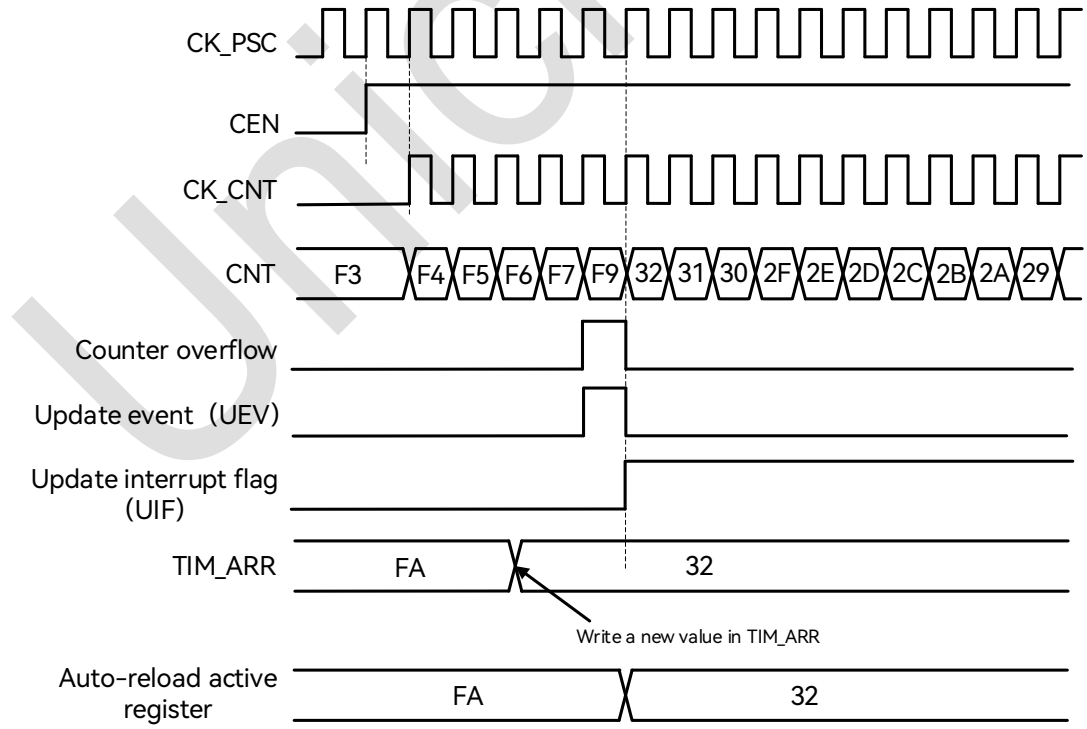


图 19-14：计数器时序图，ARPE=1 时的更新事件(计数器溢出)

19.5.3 Preload 寄存器

- 以下功能寄存器支持preload功能
  - 自动重载寄存器TIM_ARR
  - 预分频寄存器TIM_PSC（不可关闭preload功能）
  - 通道控制寄存器TIM_CCR
  - CCxE和CCxNE控制寄存器
  - OCxM控制寄存器

  以上寄存器，除了 PSC 之外，都可以由软件选择使能或者禁止 preload 功能。
- 具备preload功能的寄存器，包含两组物理实体
  - Shadow register（影子寄存器）：实际定时器正在使用的寄存器
  - Preload register（预装载寄存器）：软件可以访问的寄存器
- 当禁止preload时，具备preload功能的寄存器特性如下
  - Preload寄存器可以实时由软件访问、改写
  - Shadow寄存器与Preload寄存器同步更新
- 如果使能了preload，则
  - 所有软件操作访问的是preload寄存器
  - 当update event发生时，所有preload寄存器内容将同步被转移到对应的shadow寄存器

19.5.4 计数器工作时钟

计数器可以使用如下时钟工作：

- Timerx_clk——内部时钟模式
- 外部引脚输入时钟（TIx）——外部时钟模式1
- 外部引脚触发输入（ETR）——外部时钟模式2
- 内部触发（ITRx）——使用一个timer的触发输出（TRGO）作为计数时钟

19.5.4.1 内部时钟模式

内部时钟模式下，禁止从机模式（SMS=000），CEN、DIR、UG 等寄存器位都是软件控制。

软件操作 UG 寄存器后，update 信号经过 CLK_PSC 同步后，计数器值将被重新初始化。

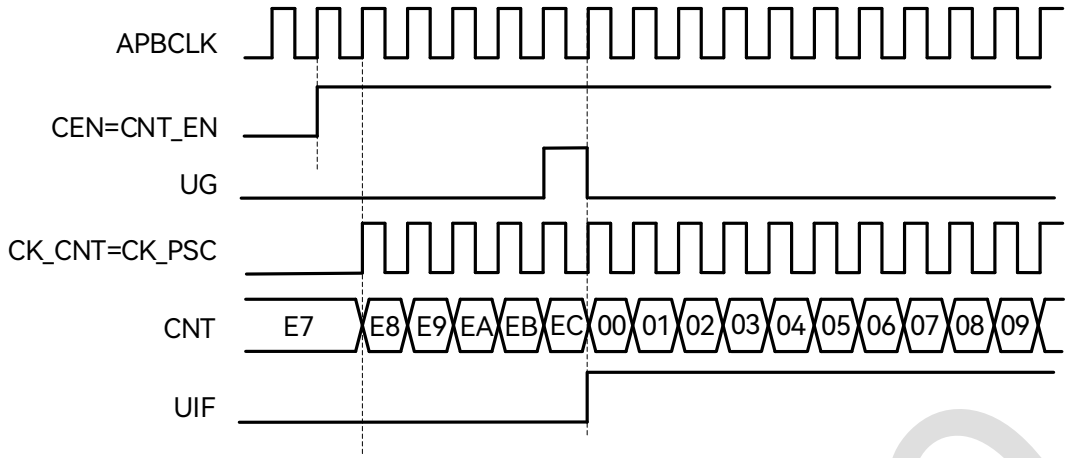


图 19-15：内部时钟源模式，时钟分频因子为 1

19.5.4.2 外部时钟模式 1

此模式下直接使用外部引脚输入信号作为计数时钟，配置 SMS=111，计数边沿可以配置为上升或下降沿。

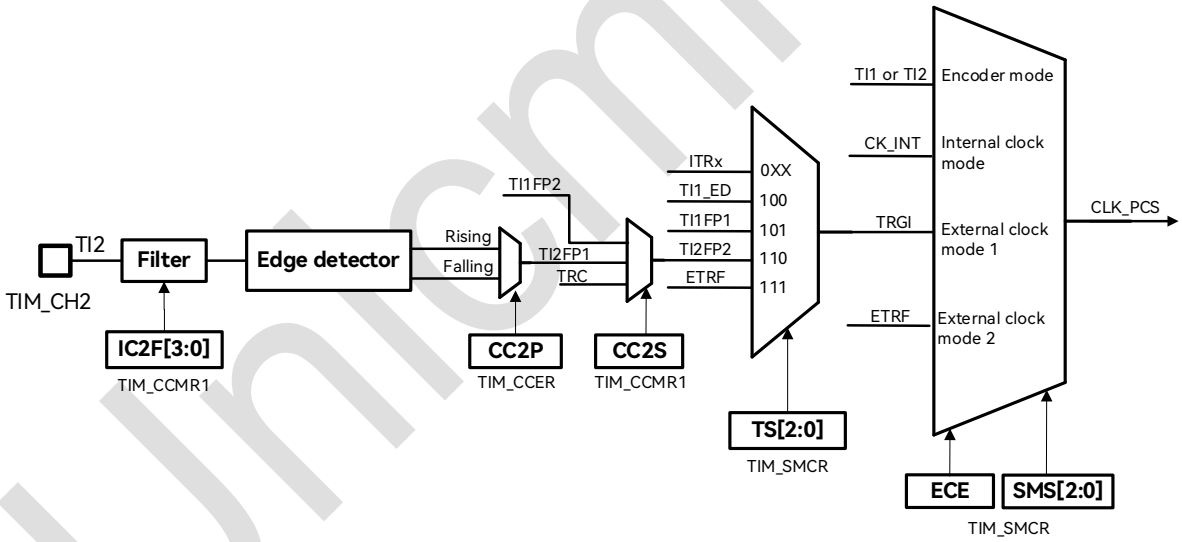


图 19-16：外部时钟连接例子

外部输入信号在触发计数器计数前，会先经过内部时钟的同步过程，同时输入信号的有效沿会触发 TIF 标志。

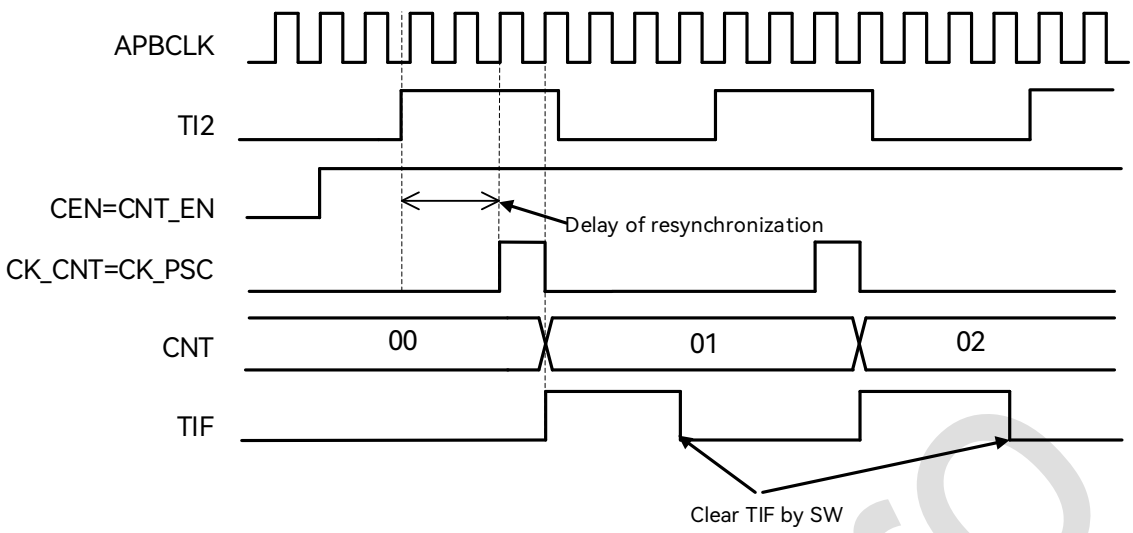


图 19-17：外部时钟模式 1 下的时序

使用外部时钟计数时，仍然要使能 TIM 的内部时钟（timerx_clk），因为 TIM 要使用 timerx_clk 来对外部输入时钟进行同步和滤波。在外部时钟模式 1 下，外部输入时钟首先经过滤波和边沿选择，得到有效的计数沿，作为有效工作时钟（CLK_PSC）输入给预分频模块。

外部时钟同步采用简单的 2 级触发器结构，因此为了避免亚稳态，要求外部输入时钟宽度至少大于 2 个 timerx_clk 周期。

此模式下只有通道 1 和 2 的输入可以用做时钟输入，所需配置如下：

1. 在GPIO模块中，配置相应管脚为TIM_CH2功能。
2. 关闭通道使能，配置TIM_CCER[4]=0，确保之后通道配置成功。
3. 选择输入通道，配置TIM_CCMR1[9:8]=01, IC2映射到TI2。
4. 选择计数有效沿，配置TIM_CCER[5]=0，选择上沿或者下沿。
5. 配置输入滤波时间，配置TIM_CCMR1.IC2F[3:0](IC2F=0000，不进行输入滤波)。
6. 使能外部时钟模式1，配置TIM_SMCR[2:0]=111。
7. 选择触发输入源，配置TIM_SMCR[6:4]=110,选定TI2作为触发输入源。
8. 打开通道使能，配置TIM_CCER[4]=1。
9. 使能计数器，配置TIM_CR1[0]=1。

下图是一个典型的外部时钟计数模式 1 的示例：

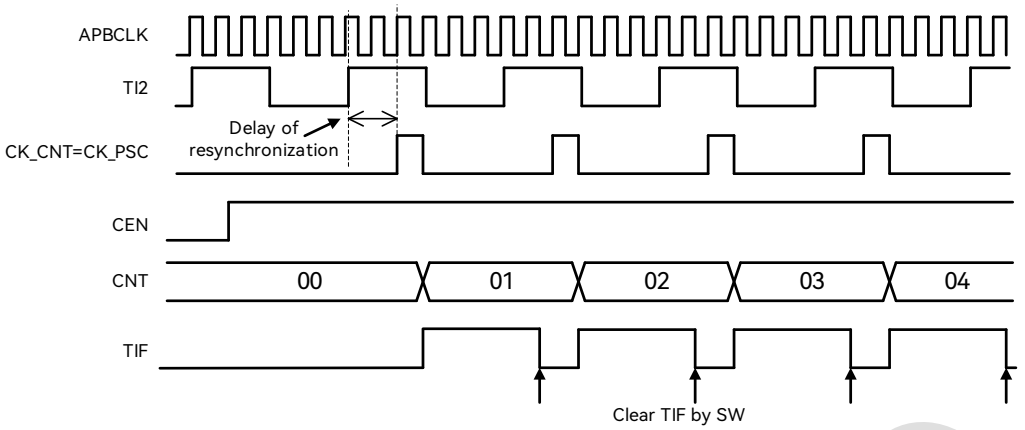


图 19-18：外部时钟模式 1 下的时序

19.5.4.3 外部时钟模式 2

此模式下使用 TIM_ETR 管脚输入信号的上升沿或下降沿（不支持双沿）来计数。

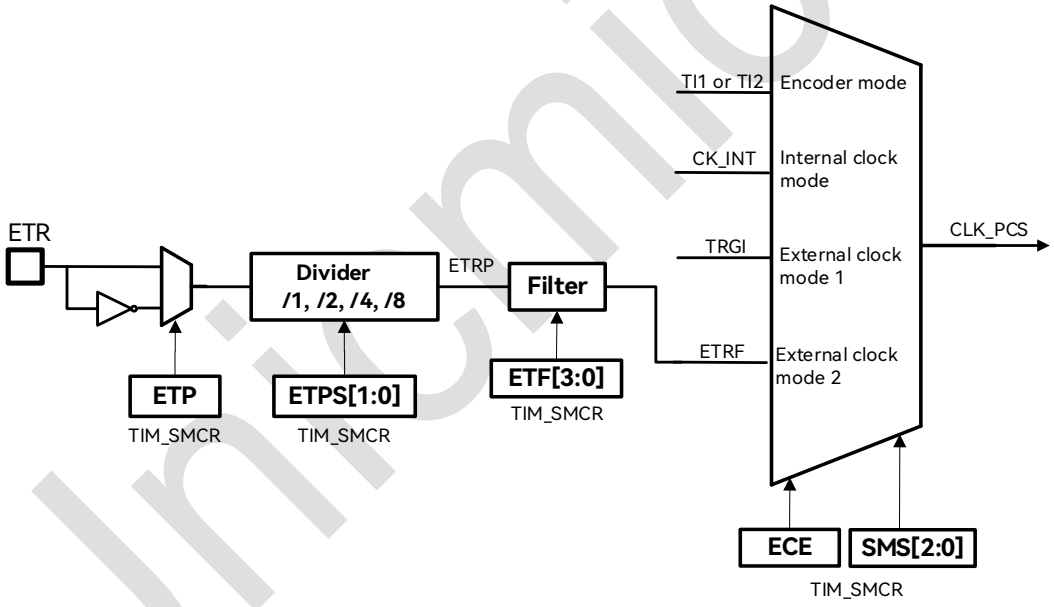


图 19-19：外部触发输入框图

下图是使用 ETR 二分频后的上升沿进行计数，其中实际计数发生时间因为内部时钟的同步过程而延迟于 ETR 输入上升沿。

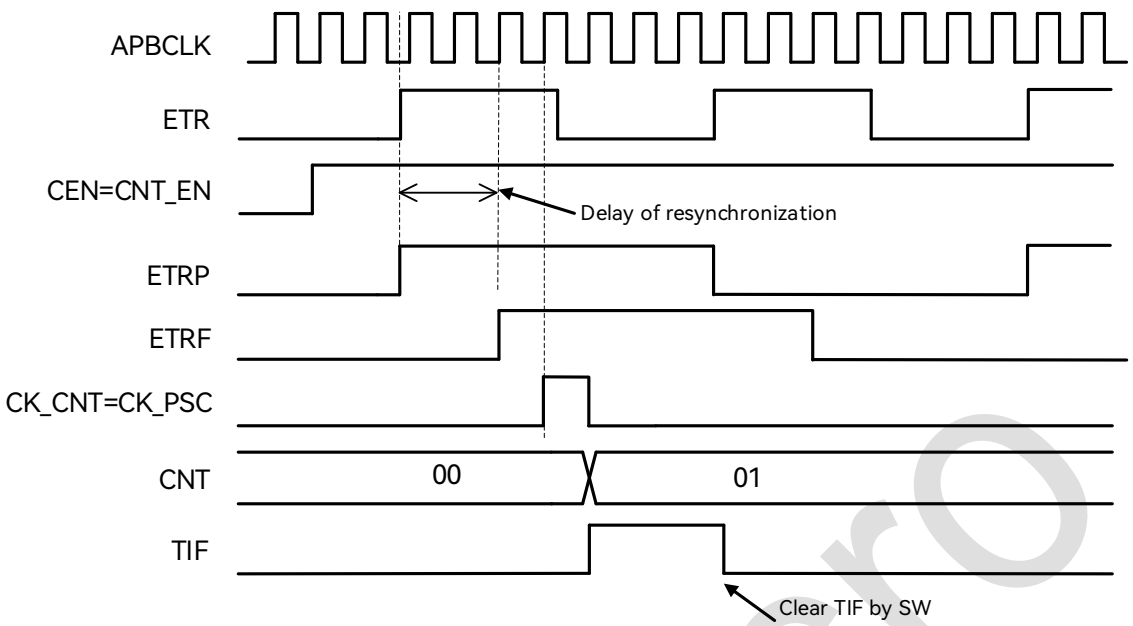


图 19-20：外部时钟模式 2 下的时序 1

与外部时钟模式 1 的主要差别是，ETR 输入直接被分频后再进行滤波，产生 CK_PSC 时钟，这意味着可以支持 ETR 输入频率高于 timerx_clk 的应用场景，这种情况下，需要首先对 ETR 输入进行预分频，再用于驱动计数器。

此模式所需配置如下：

1. 在GPIO模块中，配置相应管脚为TIM_ETR功能。
2. 设置ETP进行沿选择，TIM_SMCR[15]=0。
3. 设置ETR分频比，配置TIM_SMCR.ETPS[1:0]=01。
4. 配置输入滤波时间，TIM_SMCR.ETF[3:0]=0000。
5. 置位ECE寄存器，使能外部时钟模式2,TIM_SMCR[14]=1，TIM_SMCR[2:0]=000。
6. 使能计数器，配置TIM_CR1[0]=1。

下图是一个典型的外部时钟模式 2 的示例：

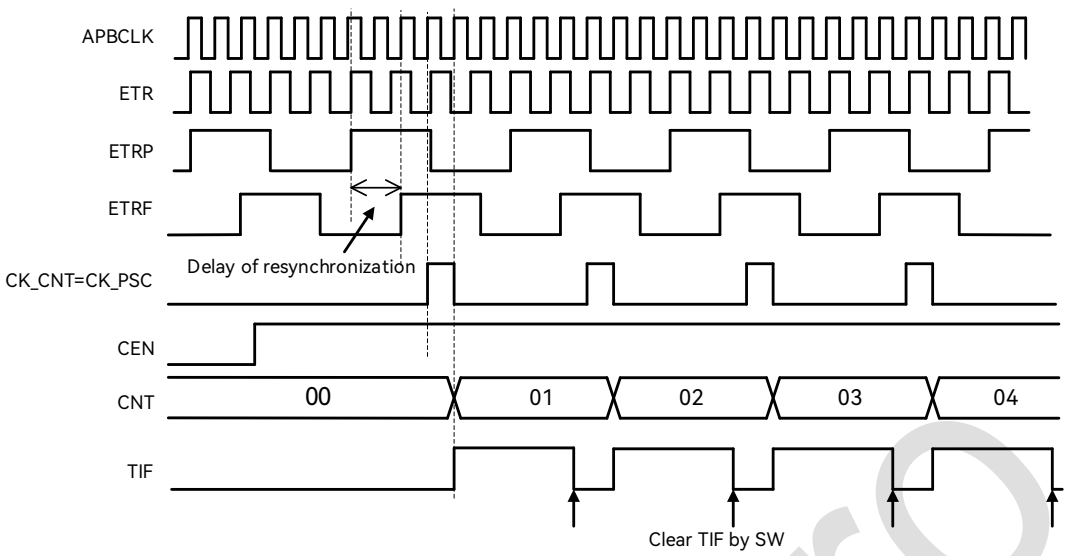


图 19-21：外部时钟模式 2 下的时序 2

在使用外部时钟模式 2 时，仍可以将 TIM 配置为 slave 模式：比如使用 ETR 输入计数，同时使用另一个 Timer 的 TRGO 作为触发信号，当触发事件到来时，复位计数器重新开始计数。

19.5.5 内部触发信号（ITRx）

TIM 支持 4 个 ITR 输入，可用于计数触发或者内部信号捕获。当用于内部信号捕获时，需要将 TS 配置为 000~011 用于选择 ITR0~ITR3，并将 CCxS 配置为 11，即将 TRC 选为捕获信号。

每个 ITR 输入支持 4 个内部信号扩展，由 ITRxSEL 寄存器配置。

19.5.6 捕获/比较通道

TIM 包含 4 个捕获/比较通道，每个通道由一个捕获比较寄存器（CCR）（包含影子寄存器）、一个捕获输入级、一个比较输出级组成。

输入级电路会采样 TIx 输入并产生滤波后的信号 TIxF，然后边沿检测和极性选择产生对应的 TIxFPx 信号，此信号可作为计数触发或者待捕获信号，并且在被捕获前经过预分频。

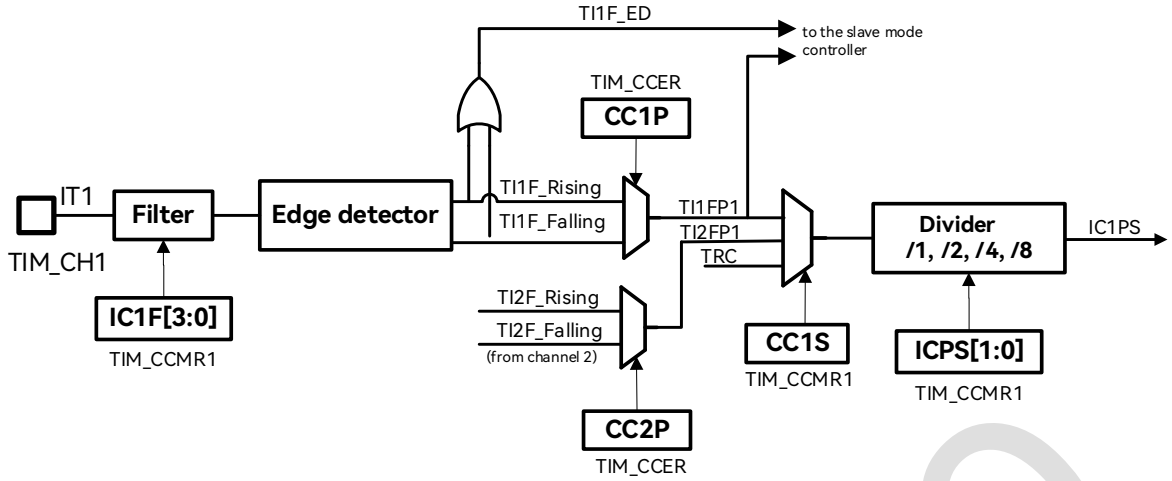


图 19-22：捕获/比较通道（通道 1 输入部分）

输出级电路会产生一个输出基准信号 OCxREF，此信号固定为高电平有效，作为最终输出电路的参考输入。其中通道 1~3 支持互补输出和死区插入，通道 4 则比较简单，不支持互补输出。

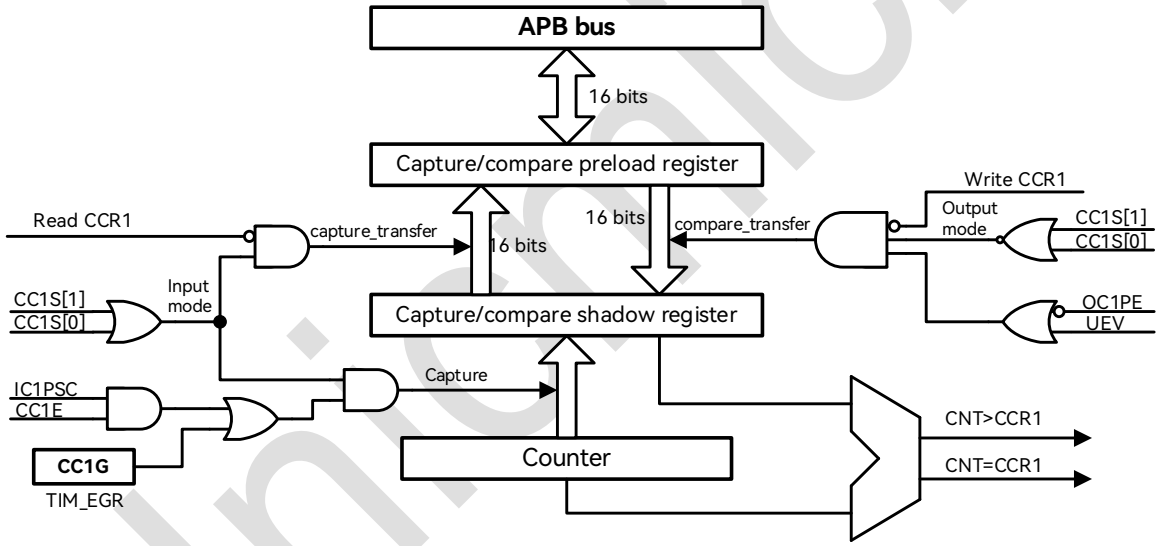


图 19-23：捕获/比较通道 1 的主电路

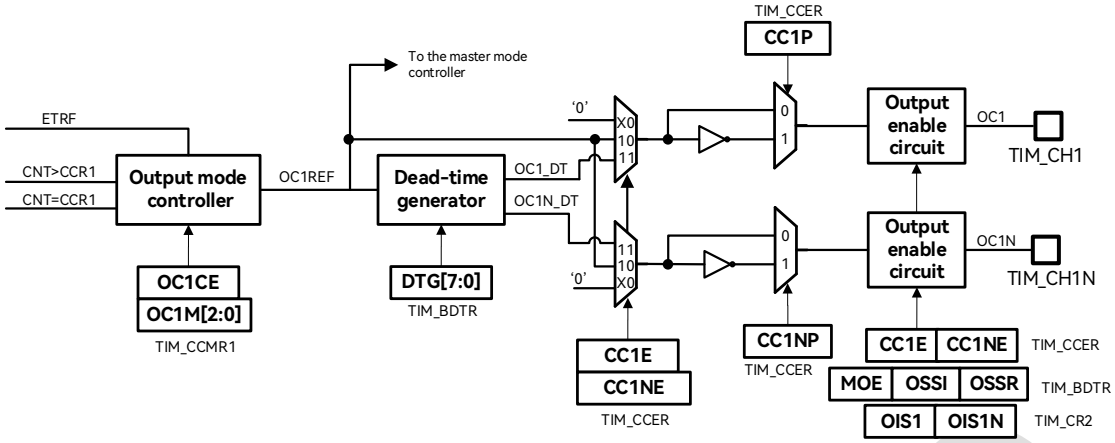


图 19-24：捕获/比较通道的输出部分（通道 1 至 3）

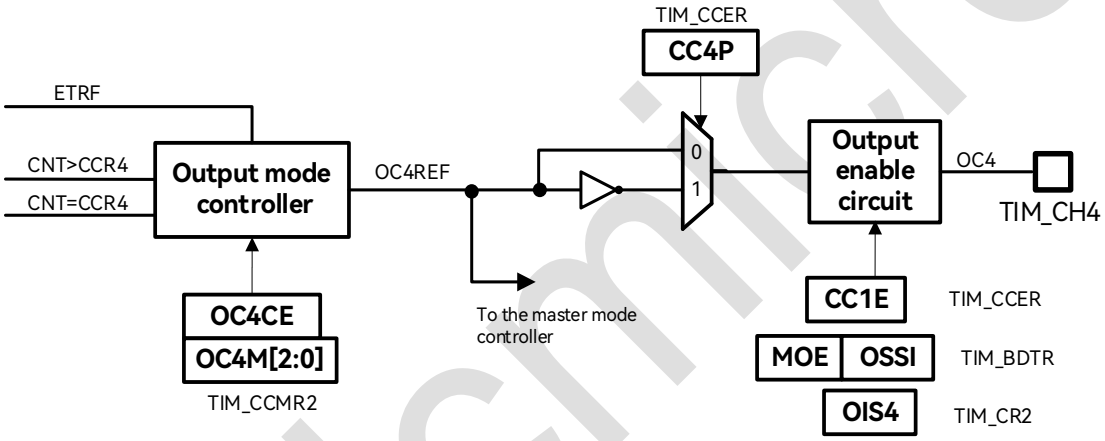


图 19-25：捕获/比较通道的输出部分（通道 4）

捕获/比较寄存器（CCR）包含 preload 寄存器和 shadow 寄存器，软件读写总是访问 preload 寄存器。在捕获模式下，捕获值保存在 shadow 寄存器中并复制到 preload 寄存器。在比较模式下，preload 寄存器的值被拷贝到 shadow 寄存器用来与计数器比较。

19.5.7 输入捕获模式

当 ICx 信号上出现预期的电平变换，将触发一次 capture，当前计数器值被锁存进 CCR，与此同时，CCxIF 中断标志置位，并且可以触发对应的中断或者 DMA 请求。如果一个捕获事件在 CCxIF 为高的情况下出现，则捕获数据冲突标志（CCxOF, Over-Capture）置位（CCR 中上次捕获值被覆盖）。CCxIF 可以由软件清零，或者通过读取 CCR 寄存器自动清零。CCxOF 标志通过软件写 1 清零。

通过两个或更多通道配合，可以实现 PWM 信号的输入捕获。比如要计算一个输入信号的周期和占空比，可以将此信号从 TI1 引脚输入，芯片内部将滤波后的信号取上升沿得到 TI1FP1，将滤波后的信号取下降沿得到 TI1FP2，将 TI1FP1 输入给捕获通道 1，将 TI1FP2 输入给捕获通道 2，

即可实现通道 1 对输入信号上升沿捕获，同时通道 2 对输入信号下降沿捕获；捕获中断定期发生后，软件通过 CCR1 和 CCR2 寄存器的值，即可计算输入信号的周期和占空比。

实现在 TI1 输入的上升沿捕获计数器的值到 TIM_CCR1 寄存器，配置步骤如下：

1. 在GPIO模块中，配置相应管脚为TIM_CH1功能。
2. 关闭通道使能，配置TIM_CCER[0]=0，确保之后通道配置成功。
3. 选择输入通道，配置TIM_CCMR1[1:0]=01, IC1映射到TI1。
4. 选择计数有效沿，配置TIM_CCER[1]，选择上沿或者下沿。
5. 配置输入滤波时间，配置TIM_CCMR1.IC1F[3:0]。
6. 配置输入预分频器，配置TIM_CCMR1.IC1PS[1:0]。
7. 打开通道使能，配置TIM_CCER[0]=1。

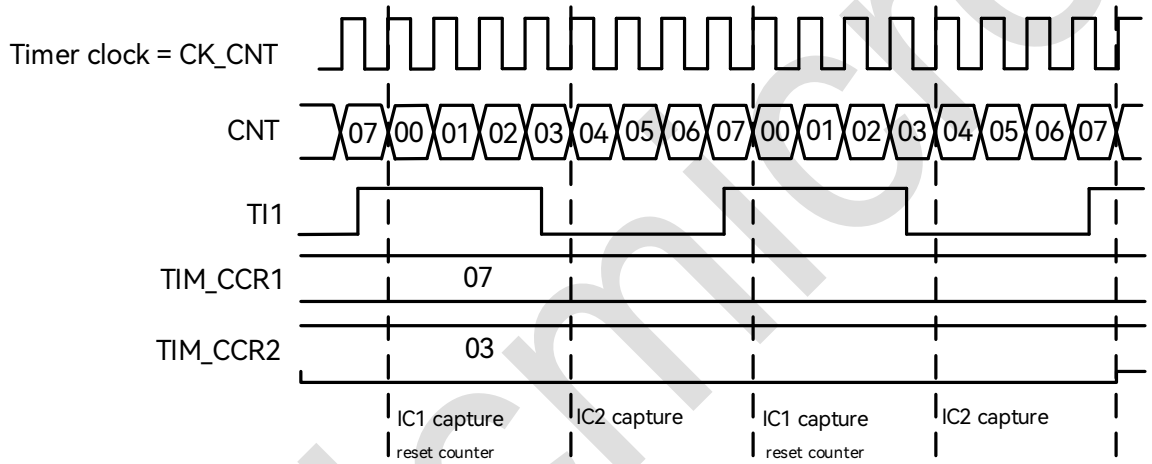


图 19-26：PWM 输入捕获模式时序

若想实现 PWM 输入捕获功能，需进行如下设置：

1. 在GPIO模块中，配置相应管脚为TIM_CH1功能。
2. 关闭通道使能，配置TIM_CCER[0]=0，TIM_CCER[4]=0确保之后通道配置成功。
3. 选择输入通道，两个通道IC1,IC2被映射到同一个TI1输入口，配置TIM_CCMR1[1:0]=01, TIM_CCMR1[9:8]=10。
4. 选择计数有效沿，两个通道IC1,IC2有效沿极性相反，配置TIM_CCER[1]=0，TIM_CCER[5]=1。
5. 配置输入滤波时间，配置TIM_CCMR1.IC1F[3:0]，TIM_CCMR1.IC2F[3:0]。
6. 配置输入预分频器，配置TIM_CCMR1.IC1PS[1:0] ，TIM_CCMR1.IC2PS[1:0]。
7. 选择触发输入信号，配置TIM_SMCR[6:4][2:0]=101。
8. 设定从模式控制器为复位模式，配置TIM_SMCR.SMS[2:0]=100。
9. 打开通道使能，配置TIM_CCER[0]=1，TIM_CCER[4]=1。

19.5.8 软件 Force 输出

在比较输出模式下，软件可以直接将 OCxREF force 成特定电平，而独立于 CCR 和计数器的比较结果。

软件通过写 OCxM=101 寄存器，可以直接将 OCxREF 强制为有效（OCxREF 固定为高有效），通过写 OCxM=100 可以直接将 OCxREF 强制为无效（低电平）。但是软件 force 操作不会取消比较过程，CCR 和计数器的比较还会一直进行。

19.5.9 输出比较模式

输出比较模式下，当 CCR 与计数器值相等，OCxREF 可以被置位成有效、无效、或电平翻转。同时，中断标志也会置位，DMA 请求可以发送（改写配置寄存器？）。

输出比较也可以被用于输出一个特定宽度的脉冲信号（单次输出）。

使用步骤：

1. 选择计数时钟（内部、外部、预分频等）。
2. 向ARR和CCR寄存器写入期望数据。
3. 根据需要设置中断使能和DMA使能。
4. 选择输出模式。
5. 使能计数器。

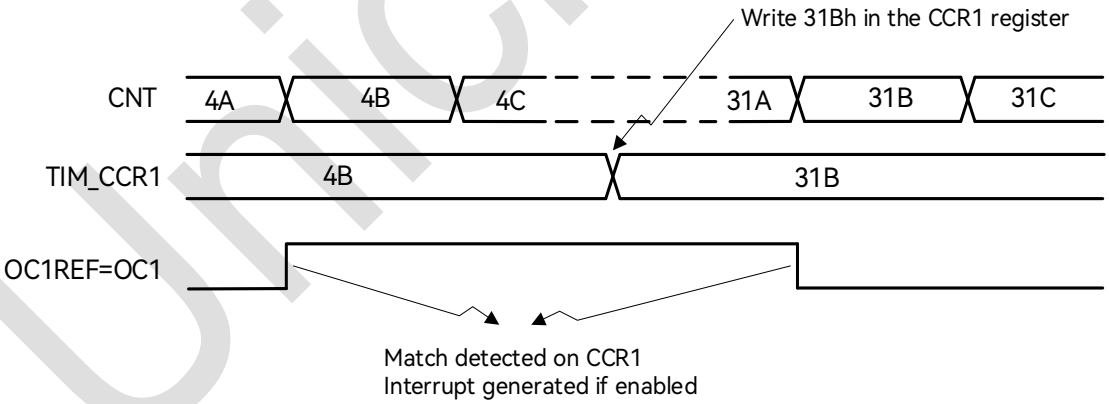


图 19-27：输出比较模式，翻转 OC1

在不使能 preload 的情况下，软件可以随时改写 CCR 寄存器实现对输出波形的实时控制。如果使能了 preload，则 CCR shadow 寄存器仅在下一次 update event 发生时更新为 preload 寄存器的内容。

19.5.10 PWM 输出

PWM 模式可以输出脉宽调制信号，其周期由 ARR 寄存器决定，占空比由 CCR 寄存器决定。

输出信号的极性可以由 CCxP 寄存器配置。PWM 模式工作中，CNT 和 CCR 实时比较。由于计数器支持边缘对齐和中央对齐计数模式，PWM 输出也支持边缘对齐和中央对齐模式。

19.5.10.1 PWM 边缘对齐模式

在向上计数的情况下，配置为 PWM 模式 1 时，OCxREF 信号在 CNT<CCR 时为高电平，否则为低电平。如果 CCR 值大于 ARR 值，则 OCxREF 被固定为 1；如果 CCR 为 0 则 OCxREF 被固定为 0。

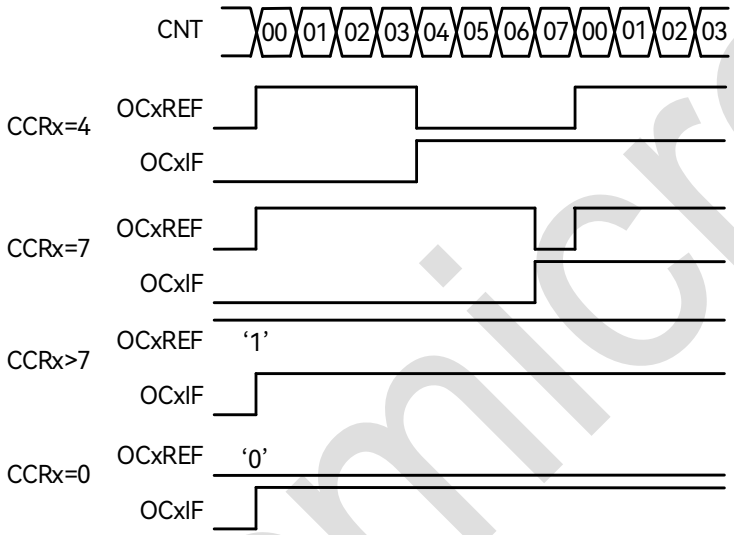


图 19-28：边沿对齐的 PWM 波形（ARR=7）

在向下计数时，OCxREF 电平高低定义与向上计数时相同。

19.5.10.2 PWM 中央对齐模式

OCxREF 电平定义与边缘对齐模式相同。下图是一个示例：

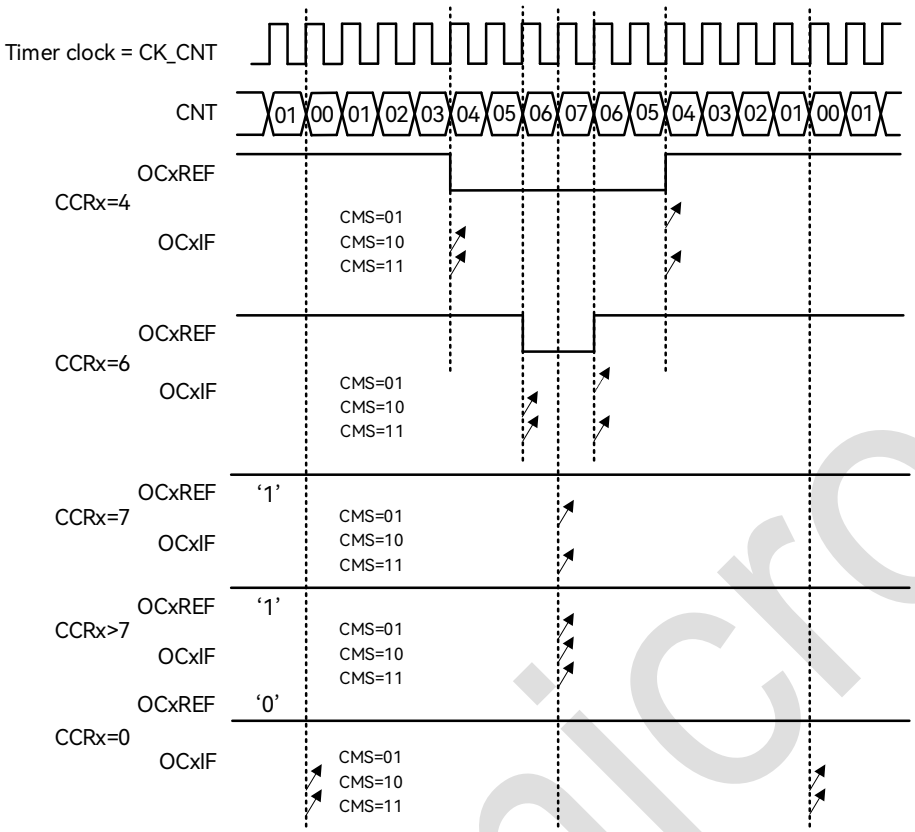


图 19-29：中央对齐的 PWM 波形（ARR=7）

当启动中央对齐计数时，一开始的计数方向是由 DIR 寄存器决定的；随后在计数过程中，DIR 寄存器的状态由硬件直接控制。安全起见，建议用户程序在启动计数器之前，通过 UG 寄存器做一次 update，并且在计数过程中不要改写计数器。

19.5.11 单脉冲输出

单脉冲输出是比较输出模式的特殊情况，允许用户在某个事件发生后，经过可编程的延迟，输出一个可编程宽度的脉冲信号。

与其他输出模式不同的是，在下一次 update event 到来时，计数器会自动停止。只有当 CCR 和计数器初值不同时，脉冲才有可能正确输出。在向上计数时，要求 CNT<CCR<=ARR，在向下计数时，要求 CNT>CCR

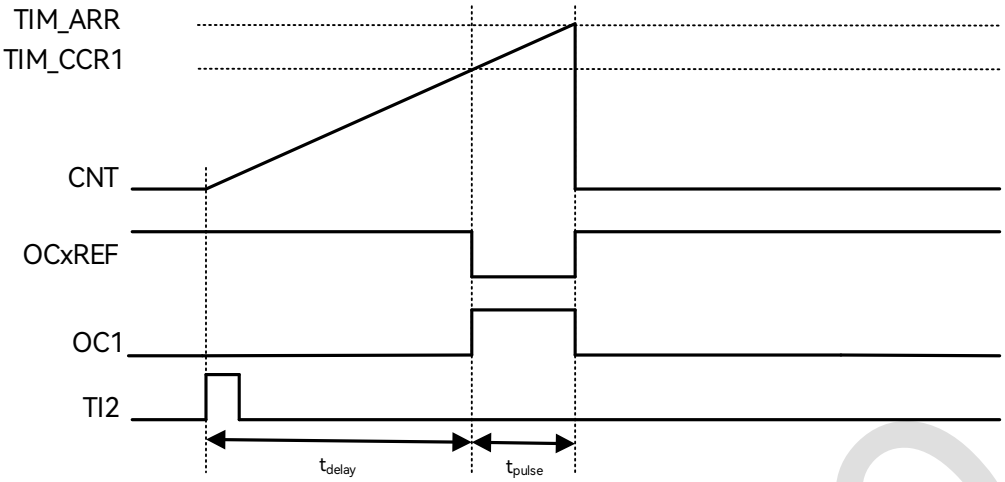


图 19-30：单脉冲模式的例子

上图是以 TI2 输入为计数器触发信号，计数值等于 CCR 后 OCxREF 输出低电平，计数到 ARR 后 OCxREF 回到高电平，并且计数器回滚到 0，停止计数。

- 实现上述功能 TI2 作为输入触发的配置如下：
  1. 在GPIO模块中，配置相应管脚为TIM_CH2功能
  2. 关闭通道使能，配置TIM_CCER[4]=0，确保之后通道配置成功
  3. 选择输入通道，配置TIM_CCMR1[9:8]=01
  4. 选择计数有效沿，配置TIM_CCER[5]=0
  5. 选择触发输入信号，配置TIM_SMCR.TS[2:0]=110，TI2FP2作为TRGI
  6. 设定从模式控制器为触发模式，配置TIM_SMCR.SMS[2:0]=110,TI2FP2用来启动计数器
  7. 打开通道使能，配置TIM_CCER[4]=1
- 实现上述功能 OC1 作为输出的配置如下：
  1. 在GPIO模块中，配置相应管脚为TIM_CH1功能
  2. 关闭通道使能，配置TIM_CCER[0]=0，确保之后通道配置成功
  3. 输出通道，配置TIM_CCMR1[1:0]=00
  4. 选择计数有效沿，配置TIM_ CCMR1.OC1M=111,PWM模式2
  5. 打开通道使能，配置TIM_CCER[0]=1
- OPM 波形产生时基的特殊设置：
  1. TIM_CCR1的值决定了Tdelay
  2. TIM_ARR和TIM_CCR1的差值决定了Tpulse（TIM_ARR-TIM_CCR1）
  3. 设置为单脉冲模式，配置TIM_CR1.OPM=1

19.5.12 外部事件清除 OCxREF

OCxREF 的有效状态未高电平，通过对外部 ETR 引脚施加高电平，可以直接拉低 OCxREF，直到下一次 update event。此功能仅在输出比较和 PWM 模式下有效，无法在软件 force 模式下起作用。使能此功能需要将 OCxCE 置 1。

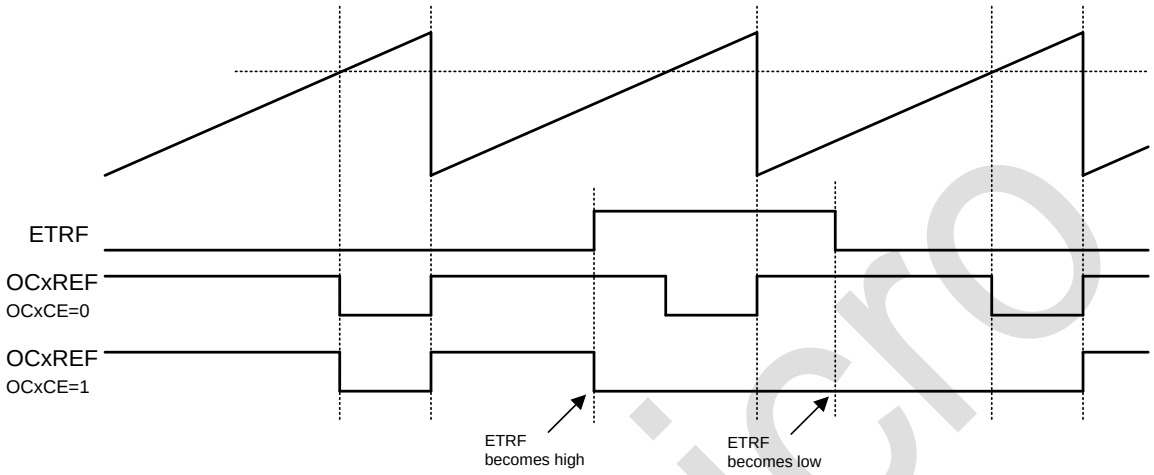


图 19-31：ETR 信号清除 TIM 的 OCxREF

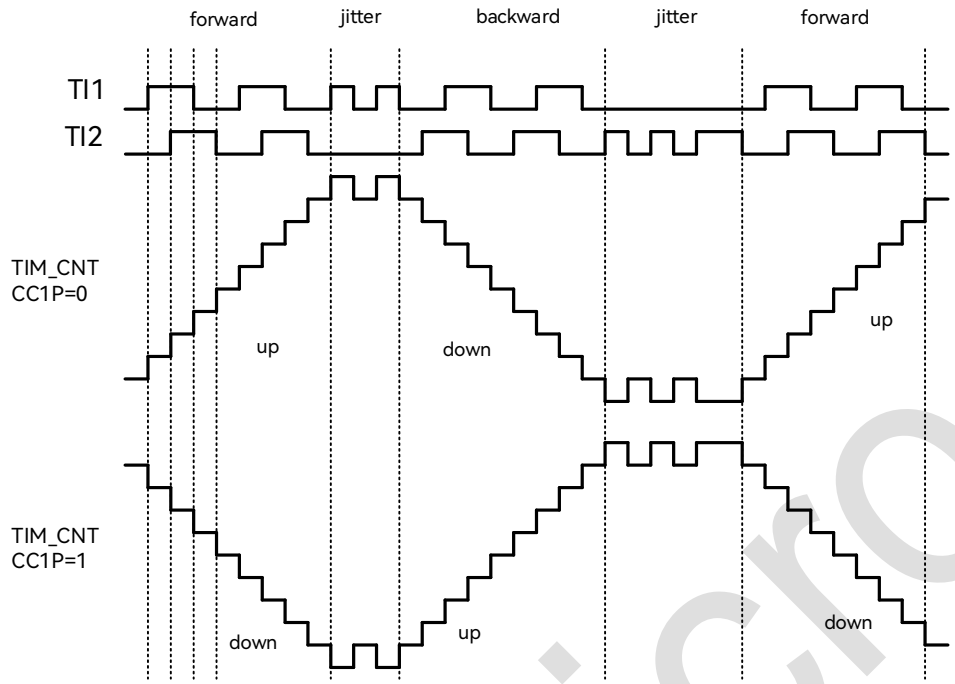
19.5.13 编码器接口模式（encoder interface）

编码器接口模式涉及到两个外部输入信号，TIM 根据其中一个信号的边沿相对于另一个信号的电平来决定递增还是递减计数值。下表是计数方式与两路输入信号之间的关系：

表 19-2：encoder interface 计数方式

有效沿	对应信号的电平（TI1 对应 TI2，TI2 对应 TI1）	TI1 信号		TI2 信号	
		上升	下降	上升	下降
仅在 TI1 处计数	高	递减	递增	不计数	不计数
	低	递增	递减	不计数	不计数
仅在 TI2 处计数	高	不计数	不计数	递增	递减
	低	不计数	不计数	递减	递增
在 TI1 和 TI2 处均计数	高	递减	递增	递增	递减
	低	递增	递减	递减	递增

比如在计数器以 TI1 信号为时钟计数时，如果 TI1 上升沿采样到 TI2 为高电平，则计数器递减；如果 TI1 下降沿采样到 TI2 为高电平，则计数器递增。



Example of counter operation in encoder interface mode

图 19-32：编码器模式下的计数器操作实例

编码模式输入通道需进行如下设置：

1. 在GPIO模块中，配置相应管脚为TIM_CH1，TIM_CH2功能。
2. 关闭通道使能，配置TIM_CCER[0]=0，TIM_CCER[4]=0，确保之后通道配置成功。
3. 选择输入通道，配置TIM_CCMR1[1:0]=01，TIM_CCMR1[9:8]=01。
4. 选择计数有效沿，配置TIM_CCER[1]=0，TIM_CCER[5]=0。
5. 设定从模式控制器为编码模式3，配置TIM_SMCR.SMS[2:0]=011 。
6. 打开通道使能，配置TIM_CCER[0]=1，TIM_CCER[4]=1。

19.5.14 TIM 从机模式

TIM 作为 slave 时（外部事件触发），可配置为三种工作模式：复位模式、门控模式、触发模式。

19.5.14.1 复位模式

此模式下，外部输入的事件将导致 TIM 内部所有 preload 寄存器重新初始化，CNT 回到 0 开始计数。以下图为例，计数器正常计数，外部 TI1 输入上升沿时，触发计数器清零，重新开始计数。

下图例中的配置如下：

1. 在GPIO模块中，配置相应管脚为TIM_CH1功能。

2. 关闭通道使能，配置TIM_CCER[0]=0确保之后通道配置成功。
3. 选择输入通道，配置TIM_CCMR1[1:0]=01。
4. 选择计数有效沿，配置TIM_CCER[1]=0。
5. 选择触发输入信号，配置TIM_SMCR.TS[2:0]=101，TI1FP1作为TRGI。
6. 设定从模式控制器为复位模式，配置TIM_SMCR.SMS[2:0]=100。
7. 打开通道使能，配置TIM_CCER[0]=1。
8. 使能计数器，配置TIM_CR1[0]=1。

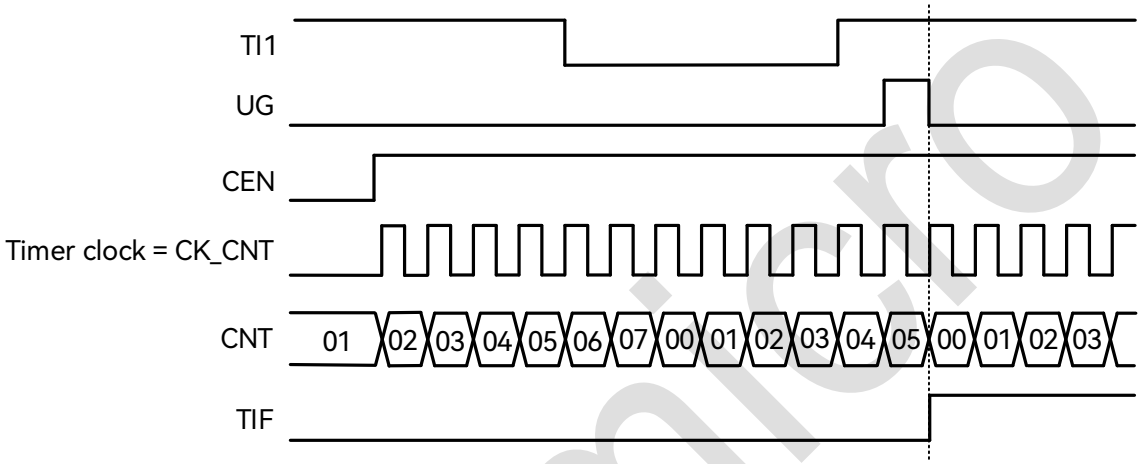


图 19-33：复位模式下的时序

19.5.14.2 门控模式

此模式下，计数器仅在输入信号为特定电平时工作。电平变换导致计数器开始或停止计数时，都会触发中断标志。

下图例中的配置如下：

1. 在GPIO模块中，配置相应管脚为TIM_CH1功能。
2. 关闭通道使能，配置TIM_CCER[0]=0确保之后通道配置成功。
3. 选择输入通道，配置TIM_CCMR1[1:0]=01。
4. 选择计数有效沿，配置TIM_CCER[1]=0。
5. 选择触发输入信号，配置TIM_SMCR.TS[2:0]=101，TI1FP1作为TRGI。
6. 设定从模式控制器为门控模式，配置TIM_SMCR.SMS[2:0]=101。
7. 打开通道使能，配置TIM_CCER[0]=1。
8. 使能计数器，配置TIM_CR1[0]=1。

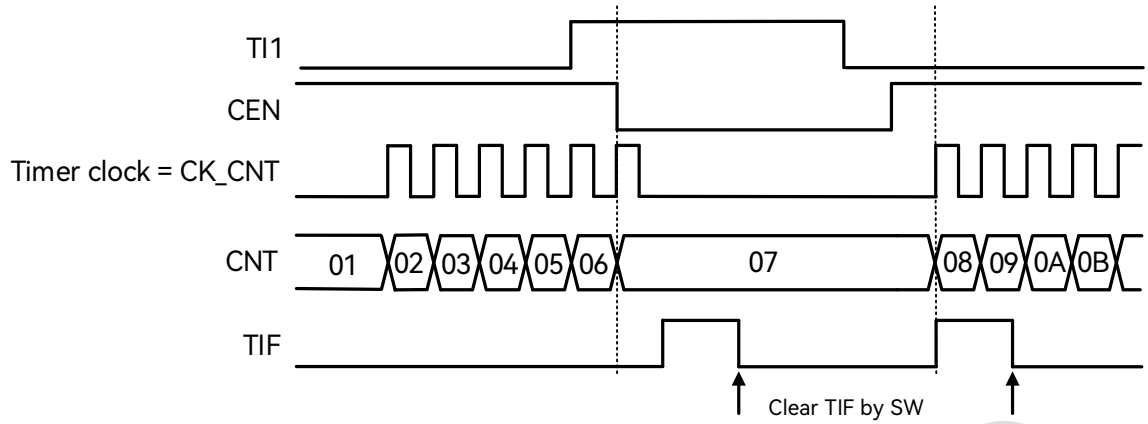


图 19-34：门控模式下的时序

19.5.14.3 触发模式

计数器在外部输入的某个事件到来后才开始计数。

下图例中的配置如下：

1. 在GPIO模块中，配置相应管脚为TIM_CH1功能。
2. 关闭通道使能，配置TIM_CCER[0]=0确保之后通道配置成功。
3. 选择输入通道，配置TIM_CCMR1[1:0]=01。
4. 选择计数有效沿，配置TIM_CCER[1]=0。
5. 选择触发输入信号，配置TIM_SMCR.TS[2:0]=101，TI1FP1作为TRGI。
6. 设定从模式控制器为触发模式，配置TIM_SMCR.SMS[2:0]=110。
7. 打开通道使能，配置TIM_CCER[0]=1。

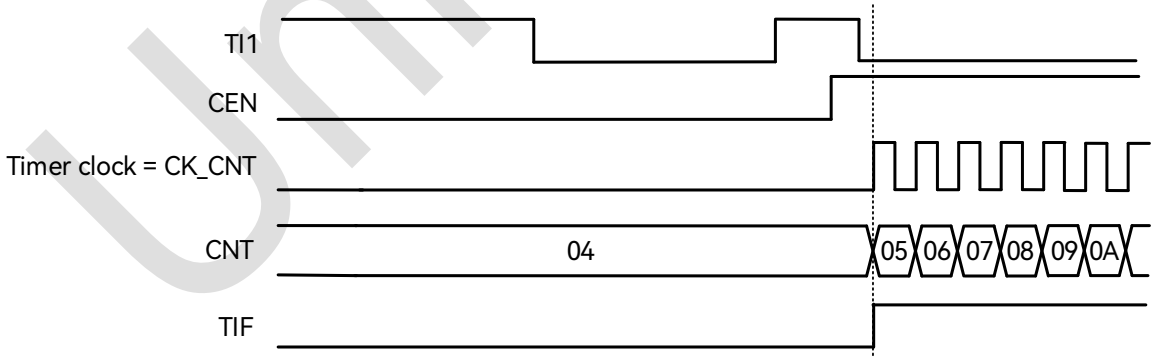


图 19-35：触发器模式下的时序

19.5.14.4 外部事件触发的外部时钟计数模式

可以将 ETR 设置为计数时钟，同时使用另一个外部输入作为计数器启动触发信号。比如在检测到 TI1 的上升沿之后，计数器开始以 ETR 输入的上升沿计数。

下图例中的配置如下：

1. 在GPIO模块中，配置相应管脚为TIM_CH1，TIM_ETR功能。
2. 设置ETP进行沿选择，TIM_SMCR[15]=0。
3. 设置ETR分频比，配置TIM_SMCR.ETPS[1:0]=01。
4. 配置输入滤波时间，TIM_SMCR.ETF[3:0]=0000。
5. 置位ECE寄存器，使能外部时钟模式2,TIM_SMCR[14]=1。
6. 关闭通道使能，配置TIM_CCER[0]=0确保之后通道配置成功。
7. 选择输入通道，配置TIM_CCMR1[1:0]=01。
8. 选择计数有效沿，配置TIM_CCER[1]=0。
9. 选择触发输入信号，配置TIM_SMCR.TS[2:0]=101，TI1FP1作为TRGI。
10. 设定从模式控制器为触发模式，配置TIM_SMCR.SMS[2:0]=110。
11. 打开通道使能，配置TIM_CCER[0]=1。

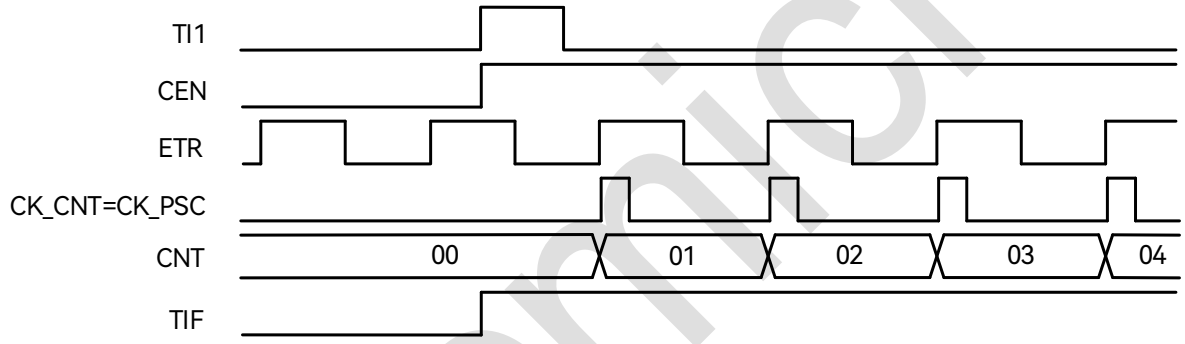


图 19-36：外部时钟模式 2＋触发模式下的时序

19.5.15 定时器同步

定时器之间可以通过触发事件级联起来，实现定时器同步或级联。定时器可以使用 4 个内部触发输入。定时器的触发信号输出则可以接到其他定时器的内部触发输入上。

19.5.16 DMA 访问

TIM 支持 7 种 DMA 请求，分别为 4 个 CC 通道请求、外部触发请求、用户软件触发请求和 COM 触发请求。

其中每个 CC 通道各自产生一个 DMA 请求，在捕获模式下用于将 CCRx 中的内容传输给 RAM，在比较模式下则用于将 RAM 中的数据写入 CCRx；CC 通道的 DMA 请求可以配置为单次传输或 Burst 传输（CCxBURSTEN），单次传输仅访问 CCRx 寄存器，Burst 传输则根据 DCR 寄存器配置对特定的一组寄存器进行访问。

此外，外部触发事件、软件触发事件和 COM 事件也可以产生 DMA 请求，当这些请求发生时，会启动 DMA Burst 传输，向 TIM 内部 1 个或多个寄存器写入数据，或者从 TIM 读取 1 个或多

个寄存器值。

表 19-3：TIM 支持 7 种 DMA 请求列表

DMA 请求	CCxBURSTEN	DMA.CHxCTRL.DIR	DMA 访问对象	一次传输长度
TIM_CH1	0	0	Read CCR1	1
		1	Write CCR1	
	1	0	Read DMAR	DBL
		1	Write DMAR	
TIM_CH2	0	0	Read CCR2	1
		1	Write CCR2	
	1	0	Read DMAR	DBL
		1	Write DMAR	
TIM_CH3	0	0	Read CCR3	1
		1	Write CCR3	
	1	0	Read DMAR	DBL
		1	Write DMAR	
TIM_CH4	0	0	Read CCR4	1
		1	Write CCR4	
	1	0	Read DMAR	DBL
		1	Write DMAR	
TIM_TRIG	N/A	0	Read DMAR	DBL
		1	Write DMAR	
TIM_UEV	N/A	0	Read DMAR	DBL
		1	Write DMAR	
TIM_COM	N/A	0	Read DMAR	DBL
		1	Write DMAR	

19.5.17 DMA Burst

TIM 支持 DMA 和 DMA-Burst 访问，可以配置 TIM 在特定事件发生时触发 DMA 请求，可以将 CCR 中的捕获结果写入 RAM，或者从 RAM 中将一个或多个寄存器内容写入 TIM 的 preload 寄存器中。

DMA-Burst 支持一个事件触发连续多次 DMA 请求，主要作用是在事件发生后连续更新多个寄存器的内容，因此可以实现动态实时调整输出波形等功能。

DMA 控制器需将外设目标地址指向一个虚拟寄存器 TIM_DMAR。在特定的定时器事件发生时，TIM 会连续发射多个 DMA 请求。每个 DMA 对 TIM_DMAR 的写操作都会被 TIM 重新定向到实际的功能寄存器上。

DBL 寄存器用于设置 DMA burst 长度，DBA 寄存器用于设置 DMA 访问 TIM 内部的基地址（相对于 TIM_CR 的 offset）。

DMA-Burst 模式下，DMA 所有访问都要指向 DMAR 虚拟寄存器，由 TIM 自动根据访问来累加内部 offset 地址。DBA 寄存器用于指定 TIM 内部首次 DMA 传输的目标地址，而 DBL 用于指定

Burst 长度。

19.5.18 输入异或功能

通道 1~3 的输入信号可以被异或起来之后，接入到通道 1 的滤波和边沿电路输入，用于通道 1 的输入捕获或者触发。

TIM_CR2 寄存器的 TI1S 位用于选择通道 1 的输入是否来自于三个通道输入的异或。

19.5.19 Debug 模式

当 CPU 进入 debug 模式后，定时器可以停止或继续工作，其行为由芯片系统中的寄存器定义。

Debug 时当定时器被停止后，其输出会被禁止（MOE 清零），根据寄存器配置，此时的输出信号可以被 force 成 inactive 或由 GPIO 模块控制。

19.6 寄存器描述

TIM1 寄存器基地址：0x4600_A000

TIM2 寄存器基地址：0x4600_A400

TIM3 寄存器基地址：0x4600_A800

TIM4 寄存器基地址：0x4600_AC00

TIM8 寄存器基地址：0x4700_9000

TIM9 寄存器基地址：0x4700_A000

TIM10 寄存器基地址：0x4700_B000

TIM11 寄存器基地址：0x4600_D000

TIM12 寄存器基地址：0x4600_E000

TIM13 寄存器基地址：0x4600_F000

寄存器列表如下：

表 19-4：通用定时器（TIM1~TIM4 & TIM8~TIM13）寄存器列表

偏移地址	名称	描述
0x00	TIM_CR1	控制寄存器 1
0x04	TIM_CR2	控制寄存器 2
0x08	TIM_SMCR	从机模式控制寄存器
0x0C	TIM_DIER	DMA 和中断使能寄存器
0x10	TIM_SR	状态寄存器
0x14	TIM_EGR	事件产生寄存器
0x18	TIM_CCMR1	捕获/比较寄存器 1

偏移地址	名称	描述
0x1C	TIM_CCMR2	捕获/比较寄存器 2
0x20	TIM_CCER	捕获/比较使能寄存器
0x24	TIM_CNT	计数值寄存器
0x28	TIM_PSC	预分频寄存器
0x2C	TIM_ARR	自动重载寄存器
0x34	TIM_CCR1	捕获/比较寄存器 1
0x38	TIM_CCR2	捕获/比较寄存器 2
0x3C	TIM_CCR3	捕获/比较寄存器 3
0x40	TIM_CCR4	捕获/比较寄存器 4
0x48	TIM_DCR	DMA 控制寄存器
0x4C	TIM_DMAR	DMA 访问寄存器

以下各节详细介绍寄存器。

19.6.1 控制寄存器 1 (TIM_CR1)

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9:8	CKD	R/W	0x0	Dead time 和数字滤波时钟频率分频寄存器（相对 CK_INT 的分频比）： 00：t_{DTS}=tCK_INT 01：t_{DTS}=2*tCK_INT 10：t_{DTS}=4*tCK_INT 11：保留，禁止使用
7	APRE	R/W	0x0	Auto-reload 预装载使能： 0：ARR 寄存器不使能 preload 1：ARR 寄存器使能 preload
6:5	CMS	R/W	0x0	计数器对齐模式选择： 00：边沿对齐模式 01：中央对齐模式 1，输出比较中断标志仅在计数器向下计数的过程中置位 10：中央对齐模式 2，输出比较中断标志仅在计数器向上计数的过程中置位 11：中央对齐模式 3，输出比较中断标志在计数器向上向下计数的过程中都会置位
4	DIR	R/W	0x0	计数方向寄存器： 0：向上计数 1：向下计数 注意：当定时器配置为中央计数模式或编码器模式时，此寄存器只读

位	名称	属性	复位值	描述
3	OPM	R/W	0x0	单脉冲输出模式： 0：Update Event 发生时计数器不停止 1：Update Event 发生时计数器停止（自动清零 CEN）
2	URS	R/W	0x0	更新请求选择： 0：以下事件能够产生 update 中断或 DMA 请求： ● 计数器上溢出或下溢出 ● 软件置位 UG 寄存器 ● 从机控制器产生 update 1：仅计数器上溢出或下溢出会产生 update 中断或 DMA 请求
1	UDIS	R/W	0x0	禁止 update 0：使能 update 事件；以下事件发生时产生 update 事件： ● 计数器上溢出或下溢出 ● 软件置位 UG 寄存器 ● 从机控制器产生 update 1：禁止 update 事件，不更新 shadow 寄存器。当 UG 置位或从机控制器收到硬件 reset 时重新初始化计数器和预分频器。
0	CEN	R/W	0x0	计数器使能： 0：计数器关闭 1：计数器使能 注意：外部触发模式可以自动置位 CEN

19.6.2 控制寄存器 2（TIM_CR2）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	TI1S	R/W	0x0	选择 T1 输入： 0：T1 输入来自 CH1 引脚 1：T1 输入来自 CH1、CH2、CH3 引脚异或组合
6:4	MMS	R/W	0x0	主机模式选择，选择 TRGO 触发的方式： 000：复位：EGR 寄存器中的 UG 位产生 TRGO 001：使能：TRGO 由计数器使能信号产生，包括 CEN 位和外部触发 010：更新：更新事件产生 TRGO 011：比较脉冲：发生输入捕获或比较事件，使 CC1F 置 1 时，产生 TRGO 100：比较：OC1REF 产生 TRGO 101：比较：OC2REF 产生 TRGO

位	名称	属性	复位值	描述
				110：比较：OC3REF 产生 TRGO 111：比较：OC4REF 产生 TRGO
3	CCDS	R/W	0x0	捕获/比较 DMA 选择： 0：发生 CCx 事件时产生 CCxDMA 请求 1：发生更新事件时产生 CCxDMA 请求
2:0	RSV	-	-	保留

19.6.3 从机模式控制寄存器（TIM_SMCR）

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	ETP	R/W	0x0	外部触发信号极性配置： 0：高电平或上升沿有效 1：低电平或下降沿有效
14	ECE	R/W	0x0	外部时钟使能： 0：关闭外部时钟模式 2 1：使能外部时钟模式 2，计数器时钟为 ETRF 有效沿
13:12	ETPS	R/W	0x0	外部触发信号预分频寄存器： 外部触发信号 ETRP 的频率最多只能是 TIM 工作时钟的 1/4，当输入信号频率较高时，可以使用预分频。 00：不分频 01：2 分频 10：4 分频 11：8 分频
11:8	ETF	R/W	0x0	外部触发信号滤波时钟和长度选择： 0000：无滤波 0001：$f_{SAMPLING}=f_{CK\_INT}$, N=2 0010：$f_{SAMPLING}=f_{CK\_INT}$, N=4 0011：$f_{SAMPLING}=f_{CK\_INT}$, N=8 0100：$f_{SAMPLING}=f_{DTS}/2$, N=6 0101：$f_{SAMPLING}=f_{DTS}/2$, N=8 0110：$f_{SAMPLING}=f_{DTS}/4$, N=6 0111：$f_{SAMPLING}=f_{DTS}/4$, N=8 1000：$f_{SAMPLING}=f_{DTS}/8$, N=6 1001：$f_{SAMPLING}=f_{DTS}/8$, N=8 1010：$f_{SAMPLING}=f_{DTS}/16$, N=5 1011：$f_{SAMPLING}=f_{DTS}/16$, N=6 1100：$f_{SAMPLING}=f_{DTS}/16$, N=8 1101：$f_{SAMPLING}=f_{DTS}/32$, N=5 1110：$f_{SAMPLING}=f_{DTS}/32$, N=6 1111：$f_{SAMPLING}=f_{DTS}$ /32, N=8
7	MSM	R/W	0x0	主机从机模式选择：

位	名称	属性	复位值	描述
				0：无动作 1：TRGI 触发的动作被延迟，以便于通过 TRGO 将当前定时器和从机定时器完美同步起来
6:4	TS	R/W	0x0	触发选择，用于选择同步计数器的触发源： 000：内部触发信号（ITR0） 001：内部触发信号（ITR1） 010：内部触发信号（ITR2） 011：内部触发信号（ITR3） 100：TI1 边沿检测（TI1F_ED） 101：滤波后 TI1（TI1FP1） 110：滤波后 TI2（TI2FP2） 111：外部触发输入（ETRF） 注：仅当 SMS=000 即禁止从机模式的情况下，可以改写 TS 寄存器
3	RSV	-	-	保留
2:0	SMS	R/W	0x0	从机模式选择： 000：从机模式禁止；CEN 使能后预分频电路时钟源来自内部时钟 001：Encoder 模式 1；计数器使用 TI2FP1 边沿，根据 TI1FP2 电平高低来计数 010：Encoder 模式 2；计数器使用 TI1FP2 边沿，根据 TI2FP1 电平高低来计数 011：Encoder 模式 3；计数器同时使用 TI1FP1 和 TI2FP2 边沿，根据其他输入信号电平来计数 100：复位模式；TRGI 上升沿初始化计数器，并触发寄存器 update 101：闸门模式；TRGI 为高电平时，计数时钟使能，TRGI 为低电平时，计数时钟停止 110：触发模式；TRGI 上升沿触发计数器开始计数（不会复位计数器） 111：外部时钟模式 1；TRGI 上升沿直接驱动计数器

19.6.4 DMA 和中断使能寄存器（TIM_DIER）

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:20	RSV	-	-	保留
19	CC4OF_DISABLE	R/W	0x0	选择禁用 CC4OF 中断： 0：不禁用 1：禁用
18	CC3OF_DISABLE	R/W	0x0	选择禁用 CC3OF 中断： 0：不禁用 1：禁用

位	名称	属性	复位值	描述
17	CC2OF_DISABLE	R/W	0x0	选择禁用 CC2OF 中断： 0：不禁用 1：禁用
16	CC1OF_DISABLE	R/W	0x0	选择禁用 CC1OF 中断： 0：不禁用 1：禁用
15	RSV	-	-	保留
14	TDE	R/W	0x0	外部触发 DMA 请求使能： 0：从机模式下，禁止外部触发事件产生 DMA 请求 1：从机模式下，允许外部触发事件产生 DMA 请求（可用于自动更新 preload 寄存器）
13	RSV	-	-	保留
12	CC4DE	R/W	0x0	捕获比较通道 4 的 DMA 请求使能： 0：禁止 CC4 DMA 请求 1：允许 CC4 DMA 请求
11	CC3DE	R/W	0x0	捕获比较通道 3 的 DMA 请求使能： 0：禁止 CC3 DMA 请求 1：允许 CC3 DMA 请求
10	CC2DE	R/W	0x0	捕获比较通道 2 的 DMA 请求使能： 0：禁止 CC2 DMA 请求 1：允许 CC2 DMA 请求
9	CC1DE	R/W	0x0	捕获比较通道 1 的 DMA 请求使能： 0：禁止 CC1 DMA 请求 1：允许 CC1 DMA 请求
8	UDE	R/W	0x0	Update Event DMA 请求使能： 0：Update Event 发生时，禁止产生 DMA 请求 1：Update Event 发生时，允许产生 DMA 请求
7	RSV	-	-	保留
6	TIE	R/W	0x0	触发事件中断使能： 0：禁止触发事件中断 1：允许触发事件中断
5	RSV	-	-	保留
4	CC4IE	R/W	0x0	捕获/比较通道 4 中断使能： 0：禁止捕获/比较 4 中断 1：允许捕获/比较 4 中断
3	CC3IE	R/W	0x0	捕获/比较通道 3 中断使能： 0：禁止捕获/比较 3 中断 1：允许捕获/比较 3 中断
2	CC2IE	R/W	0x0	捕获/比较通道 2 中断使能： 0：禁止捕获/比较 2 中断 1：允许捕获/比较 2 中断
1	CC1IE	R/W	0x0	捕获/比较通道 1 中断使能： 0：禁止捕获/比较 1 中断 1：允许捕获/比较 1 中断
0	UIE	R/W	0x0	Update 事件中断使能： 0：禁止 Update 事件中断 1：允许 Update 事件中断

19.6.5 状态寄存器（TIM_SR）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12	CC4OF	R/W0C	0x0	捕获/比较通道 4 的 Overcapture 中断： 此寄存器仅在对应通道设置为输入捕获模式的情况下有效。硬件置位，软件写 0 清零。 0：无 overcapture 事件 1：在 CC4IF 标志为 1 的情况下发生新的捕获
11	CC3OF	R/W0C	0x0	捕获/比较通道 3 的 Overcapture 中断： 此寄存器仅在对应通道设置为输入捕获模式的情况下有效。硬件置位，软件写 0 清零。 0：无 overcapture 事件 1：在 CC3IF 标志为 1 的情况下发生新的捕获
10	CC2OF	R/W0C	0x0	捕获/比较通道 2 的 Overcapture 中断： 此寄存器仅在对应通道设置为输入捕获模式的情况下有效。硬件置位，软件写 0 清零。 0：无 overcapture 事件 1：在 CC2IF 标志为 1 的情况下发生新的捕获
9	CC1OF	R/W0C	0x0	捕获/比较通道 1 的 Overcapture 中断： 此寄存器仅在对应通道设置为输入捕获模式的情况下有效。硬件置位，软件写 0 清零。 0：无 overcapture 事件 1：在 CC1IF 标志为 1 的情况下发生新的捕获
8:7	RSV	-	-	保留
6	TIF	R/W0C	0x0	触发事件中断标志，硬件置位，软件写 0 清零
5	RSV	-	-	保留
4	CC4IF	R/W0C	0x0	捕获/比较通道 4 中断标志： 如果 CC4 通道配置为输出：CC4IF 在计数值等于比较值时置位，软件写 0 清零。 如果 CC4 通道配置为输入：发生捕获事件时置位，软件写 0 清零，或者软件读 TIM_CCR4 自动清零。
3	CC3IF	R/W0C	0x0	捕获/比较通道 3 中断标志： 如果 CC3 通道配置为输出：CC3IF 在计数值等于比较值时置位，软件写 0 清零。 如果 CC3 通道配置为输入：发生捕获事件时置位，软件写 0 清零，或者软件读 TIM_CCR3 自动清零。
2	CC2IF	R/W0C	0x0	捕获/比较通道 2 中断标志： 如果 CC2 通道配置为输出：CC2IF 在计数值等于比较值时置位，软件写 0 清零。 如果 CC2 通道配置为输入：发生捕获事件时置位，软件写 0 清零，或者软件读 TIM_CCR2 自动清零。
1	CC1IF	R/W0C	0x0	捕获/比较通道 1 中断标志： 如果 CC1 通道配置为输出：CC1IF 在计数值等于比较值时置位，软件写 0 清零。 如果 CC1 通道配置为输入：发生捕获事件时置位，软件写 0 清零，或者软件读 TIM_CCR1 自动清零。

位	名称	属性	复位值	描述
0	UIF	R/W0C	0x0	Update 事件中断标志，硬件置位，软件写 0 清零。 当以下事件发生时，UIF 置位，并更新 shadow 寄存器 ● 重复计数器=0，并且 UDIS=0 的情况下，计数器发生溢出 ● URS=0 且 UDIS=0 的情况下，软件置位 UG 寄存器初始化计数器 ● URS=0 且 UDIS=0 的情况下，触发事件初始化计数器

19.6.6 事件产生寄存器（TIM_EGR）

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6	TG	W	0x0	软件触发，软件置位此寄存器产生触发事件，硬件自动清零
5	RSV	-	-	保留
4	CC4G	W	0x0	捕获/比较通道 4 软件触发： 如果 CC4 通道配置为输出：CC4G 置位，在使能的情况下可以产生相应的中断和 DMA 请求 如果 CC4 通道配置为输入：当前计数值被捕获到 TIM_CCR4 寄存器，CC4G 置位，在使能的情况下可以产生相应的中断和 DMA 请求
3	CC3G	W	0x0	捕获/比较通道 3 软件触发： 如果 CC3 通道配置为输出：CC3G 置位，在使能的情况下可以产生相应的中断和 DMA 请求 如果 CC3 通道配置为输入：当前计数值被捕获到 TIM_CCR3 寄存器，CC3G 置位，在使能的情况下可以产生相应的中断和 DMA 请求
2	CC2G	W	0x0	捕获/比较通道 2 软件触发： 如果 CC2 通道配置为输出：CC2G 置位，在使能的情况下可以产生相应的中断和 DMA 请求 如果 CC2 通道配置为输入：当前计数值被捕获到 TIM_CCR2 寄存器，CC2G 置位，在使能的情况下可以产生相应的中断和 DMA 请求

位	名称	属性	复位值	描述
1	CC1G	W	0x0	捕获/比较通道 1 软件触发： 如果 CC1 通道配置为输出：CC1G 置位，在使能的情况下可以产生相应的中断和 DMA 请求 如果 CC1 通道配置为输入：当前计数值被捕获到 TIM_CCR1 寄存器，CC1G 置位，在使能的情况下可以产生相应的中断和 DMA 请求
0	UG	W	0x0	软件 Update 事件，软件置位此寄存器产生 Update 事件，硬件自动清零 软件置位 UG 时会重新初始化计数器并更新 shadow 寄存器，预分频计数器被清零。

19.6.7 捕获/比较寄存器 1（TIM_CCMR1）

偏移地址：0x18

复位值：0x0000 0000

此寄存器在输出比较和输入捕获配置下复用为两组不同功能。

- 输出比较模式

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	OC2CE	R/W	0x0	输出比较 2 清零使能，参考 OC1CE
14:12	OC2M	R/W	0x0	输出比较 2 模式配置，参考 OC1M
11	OC2PE	R/W	0x0	输出比较 2 预装载使能，参考 OC1PE
10	OC2FE	R/W	0x0	输出比较 2 快速使能，参考 OC1FE
9:8	CC2S	R/W	0x0	捕获/比较 2 通道选择： 00：CC2 通道配置为输出 01：CC2 通道配置为输入，IC2 映射到 TI2 10：CC2 通道配置为输入，IC2 映射到 TI1 11：CC2 通道配置为输入，IC2 映射到 TRC 注：CC2S 仅在通道关闭时（CC2E=0）可以写
7	OC1CE	R/W	0x0	输出比较 1 清零使能： 0：OC1REF 不受 ETRF 影响 1：检测到 ETRF 高电平时，自动清零 OC1REF

位	名称	属性	复位值	描述
6:4	OC1M	R/W	0x0	输出比较 1 模式配置，此寄存器定义 OC1REF 信号的行为： 000：输出比较寄存器 CCR1 和计数器 CNT 的比较结果不会影响输出 001：CCR1=CNT 时（后边沿），将 OC1REF 置高 010：CCR1=CNT 时（后边沿），将 OC1REF 置低 011：CCR1=CNT 时（后边沿），翻转 OC1REF 100：OC1REF 固定为低（inactive） 101：OC1REF 固定为高（active） 110：PWM 模式 1 –在向上计数时，OC1REF 在 CNT<CCR1 时置高，否则置低；在向下计数时，OC1REF 在 CNT>=CCR1 时置低，否则置高 111：PWM 模式 2 –在向上计数时，OC1REF 在 CNT<CCR1 时置低，否则置高；在向下计数时，OC1REF 在 CNT>=CCR1 时置高，否则置低
3	OC1PE	R/W	0x0	输出比较 1 预装载使能： 0：CCR1 preload 寄存器无效，CCR1 可以直接写入 1：CCR1 preload 寄存器有效，针对 CCR1 的读写操作都是访问 preload 寄存器，当 update event 发生时才将 preload 寄存器的内容转移到 shadow 寄存器中
2	OC1FE	R/W	0x0	输出比较 1 快速使能： 0：关闭快速使能，trigger 输入不会影响比较输出 1：打开快速使能，trigger 输入会立即将 OC1REF 改变为比较值匹配时的输出，而不管当前实际比较情况 此功能仅在当前通道配置为 PWM1 或 PWM2 模式时有效
1:0	CC1S	R/W	0x0	捕获/比较 1 通道选择： 00：CC1 通道配置为输出 01：CC1 通道配置为输入，IC1 映射到 TI1 10：CC1 通道配置为输入，IC1 映射到 TI2 11：CC1 通道配置为输入，IC1 映射到 TRC 注意：CC1S 仅在通道关闭时（CC1E=0）可以写

- 输入捕获模式

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:12	IC2F	R/W	0x0	输入捕获 2 滤波

位	名称	属性	复位值	描述
11:10	IC2PSC	R/W	0x0	输入捕获 2 预分频
9:8	CC2S	R/W	0x0	捕获/比较 2 通道选择： 00：CC2 通道配置为输出 01：CC2 通道配置为输入，IC2 映射到 TI2 10：CC2 通道配置为输入，IC2 映射到 TI1 11：CC2 通道配置为输入，IC2 映射到 TRC 注：CC2S 仅在通道关闭时（CC2E=0）可以写
7:4	IC1F	R/W	0x0	输入捕获 1 滤波。 此寄存器定义 TI1 的采样频率和滤波长度： 0000：无滤波，使用 fDTS 采样 0001：fSAMPLING=fCK_INT, N=2 0010：fSAMPLING=fCK_INT, N=4 0011：fSAMPLING=fCK_INT, N=8 0100：fSAMPLING=fDTS/2, N=6 0101：fSAMPLING=fDTS/2, N=8 0110：fSAMPLING=fDTS/4, N=6 0111：fSAMPLING=fDTS/4, N=8 1000：fSAMPLING=fDTS/8, N=6 1001：fSAMPLING=fDTS/8, N=8 1010：fSAMPLING=fDTS/16, N=5 1011：fSAMPLING=fDTS/16, N=6 1100：fSAMPLING=fDTS/16, N=8 1101：fSAMPLING=fDTS/32, N=5 1110：fSAMPLING=fDTS/32, N=6 1111：fSAMPLING=fDTS/32, N=8
3:2	IC1PSC	R/W	0x0	输入捕获 1 预分频： 00：无分频 01：每 2 个事件输入产生一次捕获 10：每 4 个事件输入产生一次捕获 11：每 8 个事件输入产生一次捕获 IC1PSC 寄存器在 CC1E=0 时复位
1:0	CC1S	R/W	0x0	捕获/比较 1 通道选择： 00：CC1 通道配置为输出 01：CC1 通道配置为输入，IC1 映射到 TI1 10：CC1 通道配置为输入，IC1 映射到 TI2 11：CC1 通道配置为输入，IC1 映射到 TRC 注：CC1S 仅在通道关闭时（CC1E=0）可以写

19.6.8 捕获/比较寄存器 2（TIM_CCMR2）

偏移地址：0x1C

复位值：0x0000 0000

此寄存器在输出比较和输入捕获配置下复用为两组不同功能。

- 输出比较模式

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	OC4CE	R/W	0x0	输出比较 4 清零使能： 0：OC4REF 不受 ETRF 影响 1：检测到 ETRF 高电平时，自动清零 OC4REF
14:12	OC4M	R/W	0x0	输出比较 4 模式配置： 此寄存器定义 OC4REF 信号的行为 000：输出比较寄存器 CCR4 和计数器 CNT 的比较结果不会影响输出 001：CCR4=CNT 时，将 OC4REF 置高 010：CCR4=CNT 时，将 OC4REF 置低 011：CCR4=CNT 时，翻转 OC4REF 100：OC4REF 固定为低（inactive） 101：OC4REF 固定为高（active） 110：PWM 模式 1 ● 在向上计数时，OC4REF 在 CNT<CCR4 时置高，否则置低； ● 在向下计数时，OC4REF 在 CNT>CCR4 时置低，否则置高 111：PWM 模式 2 ● 在向上计数时，OC4REF 在 CNT<CCR4 置低，否则置高； ● 在向下计数时，OC4REF 在 CNT>CCR4 时置高，否则置低
11	OC4PE	R/W	0x0	输出比较 4 预装载使能： 0：CCR4 preload 寄存器无效，CCR4 可以直接写入 1：CCR4 preload 寄存器有效，针对 CCR4 的读写操作都是访问 preload 寄存器，当 update event 发生时才将 preload 寄存器的内容转移到 shadow 寄存器中
10	OC4FE	R/W	0x0	输出比较 4 快速使能： 0：关闭快速使能，trigger 输入不会影响比较输出 1：打开快速使能，trigger 输入会立即将 OC4REF 改变为比较值匹配时的输出，而不管当前实际比较情况 此功能仅在当前通道配置为 PWM1 或 PWM2 模式时有效
9:8	CC4S	R/W	0x0	捕获/比较 4 通道选择： 00：CC4 通道配置为输出 01：CC4 通道配置为输入，IC4 映射到 TI4 10：CC4 通道配置为输入，IC4 映射到 TI3 11：CC4 通道配置为输入，IC4 映射到 TRC 注：CC4S 仅在通道关闭时（CC4E=0）可以写

位	名称	属性	复位值	描述
7	OC3CE	R/W	0x0	输出比较 3 清零使能： 0：OC3REF 不受 ETRF 影响 1：检测到 ETRF 高电平时，自动清零 OC3REF
6:4	OC3M	R/W	0x0	输出比较 3 模式配置，此寄存器定义 OC3REF 信号的行为： 000：输出比较寄存器 CCR3 和计数器 CNT 的比较结果不会影响输出 001：CCR3=CNT 时，将 OC3REF 置高 010：CCR3=CNT 时，将 OC3REF 置低 011：CCR3=CNT 时，翻转 OC3REF 100：OC3REF 固定为低（inactive） 101：OC3REF 固定为高（active） 110：PWM 模式 1 –在向上计数时，OC3REF 在 CNT<CCR3 时置高，否则置低；在向下计数时，OC3REF 在 CNT>CCR3 时置低，否则置高 111：PWM 模式 2 –在向上计数时，OC3REF 在 CNT<CCR3 时置低，否则置高；在向下计数时，OC3REF 在 CNT>CCR3 时置高，否则置低
3	OC3PE	R/W	0x0	输出比较 3 预装载使能： 0：CCR3 preload 寄存器无效，CCR3 可以直接写入 1：CCR3 preload 寄存器有效，针对 CCR3 的读写操作都是访问 preload 寄存器，当 update event 发生时才将 preload 寄存器的内容转移到 shadow 寄存器中
2	OC3FE	R/W	0x0	输出比较 3 快速使能： 0：关闭快速使能，trigger 输入不会影响比较输出 1：打开快速使能，trigger 输入会立即将 OC3REF 改变为比较值匹配时的输出，而不管当前实际比较情况 此功能仅在当前通道配置为 PWM1 或 PWM2 模式时有效
1:0	CC3S	R/W	0x0	捕获/比较 3 通道选择： 00：CC3 通道配置为输出 01：CC3 通道配置为输入，IC3 映射到 TI3 10：CC3 通道配置为输入，IC3 映射到 TI4 11：CC3 通道配置为输入，IC3 映射到 TRC 注：CC3S 仅在通道关闭时（CC3E=0）可以写

- 输入捕获模式

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留

位	名称	属性	复位值	描述
15:12	IC4F	R/W	0x0	输入捕获 4 滤波。 此寄存器定义 TI4 的采样频率和滤波长度： 0000：无滤波，使用 fDTS 采样 0001：fSAMPLING=fCK_INT, N=2 0010：fSAMPLING=fCK_INT, N=4 0011：fSAMPLING=fCK_INT, N=8 0100：fSAMPLING=fDTS/2, N=6 0101：fSAMPLING=fDTS/2, N=8 0110：fSAMPLING=fDTS/4, N=6 0111：fSAMPLING=fDTS/4, N=8 1000：fSAMPLING=fDTS/8, N=6 1001：fSAMPLING=fDTS/8, N=8 1010：fSAMPLING=fDTS/16, N=5 1011：fSAMPLING=fDTS/16, N=6 1100：fSAMPLING=fDTS/16, N=8 1101：fSAMPLING=fDTS/32, N=5 1110：fSAMPLING=fDTS/32, N=6 1111：fSAMPLING=fDTS/32, N=8
11:10	IC4PSC	R/W	0x0	输入捕获 4 预分频： 00：无分频 01：每 2 个事件输入产生一次捕获 10：每 4 个事件输入产生一次捕获 11：每 8 个事件输入产生一次捕获 IC4PSC 寄存器在 CC4E=0 时复位
9:8	CC4S	R/W	0x0	捕获/比较 4 通道选择： 00：CC4 通道配置为输出 01：CC4 通道配置为输入，IC4 映射到 TI4 10：CC4 通道配置为输入，IC4 映射到 TI3 11：CC4 通道配置为输入，IC4 映射到 TRC 注：CC4S 仅在通道关闭时（CC4E=0）可以写
7:4	IC3F	R/W	0x0	输入捕获 3 滤波。 此寄存器定义 TI3 的采样频率和滤波长度： 0000：无滤波，使用 fDTS 采样 0001：fSAMPLING=fCK_INT, N=2 0010：fSAMPLING=fCK_INT, N=4 0011：fSAMPLING=fCK_INT, N=8 0100：fSAMPLING=fDTS/2, N=6 0101：fSAMPLING=fDTS/2, N=8 0110：fSAMPLING=fDTS/4, N=6 0111：fSAMPLING=fDTS/4, N=8 1000：fSAMPLING=fDTS/8, N=6 1001：fSAMPLING=fDTS/8, N=8 1010：fSAMPLING=fDTS/16, N=5 1011：fSAMPLING=fDTS/16, N=6 1100：fSAMPLING=fDTS/16, N=8 1101：fSAMPLING=fDTS/32, N=5 1110：fSAMPLING=fDTS/32, N=6 1111：fSAMPLING=fDTS/32, N=8

位	名称	属性	复位值	描述
3:2	IC3PSC	R/W	0x0	输入捕获 3 预分频： 00：无分频 01：每 2 个事件输入产生一次捕获 10：每 4 个事件输入产生一次捕获 11：每 8 个事件输入产生一次捕获 IC3PSC 寄存器在 CC3E=0 时复位
1:0	CC3S	R/W	0x0	捕获/比较 3 通道选择： 00：CC3 通道配置为输出 01：CC3 通道配置为输入，IC3 映射到 TI3 10：CC3 通道配置为输入，IC3 映射到 TI4 11：CC3 通道配置为输入，IC3 映射到 TRC 注：CC3S 仅在通道关闭时（CC3E=0）可以写

19.6.9 捕获/比较使能寄存器（TIM_CCER）

偏移地址：0x20

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:14	RSV	-	-	保留
13	CC4P	R/W	0x0	捕获/比较 4 输出极性： CC4 通道配置为输出时 0：OC4 为 OC4REF 1：OC4 为 OC4REF 反转 CC4 通道配置为输入时 0：非取反模式–捕获在 IC4 的上升沿进行 1：取反模式–捕获在 IC4 的下降沿进行
12	CC4E	R/W	0x0	捕获/比较 4 输出使能： CC4 通道配置为输出时 0：OC4 不 active 1：OC4 active CC4 通道配置为输入时 0：关闭捕获功能 1：使能捕获功能
11:10	RSV	-	-	保留
9	CC3P	R/W	0x0	捕获/比较 3 输出极性： CC3 通道配置为输出时 0：OC3 为 OC3REF 1：OC3 为 OC3REF 反转 CC3 通道配置为输入时 0：非取反模式–捕获在 IC3 的上升沿进行 1：取反模式–捕获在 IC3 的下降沿进行

位	名称	属性	复位值	描述
8	CC3E	R/W	0x0	捕获/比较 3 输出使能： CC3 通道配置为输出时 0：OC3 不 active 1：OC3 active CC3 通道配置为输入时 0：关闭捕获功能 1：使能捕获功能
7:6	RSV	-	-	保留
5	CC2P	R/W	0x0	捕获/比较 2 输出极性： CC2 通道配置为输出时 0：OC2 为 OC2REF 1：OC2 为 OC2REF 反转 CC2 通道配置为输入时 0：非取反模式–捕获在 IC2 的上升沿进行 1：取反模式–捕获在 IC2 的下降沿进行
4	CC2E	R/W	0x0	捕获/比较 2 输出使能： CC2 通道配置为输出时 0：OC2 不 active 1：OC2 active CC2 通道配置为输入时 0：关闭捕获功能 1：使能捕获功能
3:2	RSV	-	-	保留
1	CC1P	R/W	0x0	捕获/比较 1 输出极性： CC1 通道配置为输出时 0：OC1 为 OC1REF 1：OC1 为 OC1REF 反转 CC1 通道配置为输入时 0：非取反模式–捕获在 IC1 的上升沿进行 1：取反模式–捕获在 IC1 的下降沿进行
0	CC1E	R/W	0x0	捕获/比较 1 输出使能： CC1 通道配置为输出时 0：OC1 不 active 1：OC1 active CC1 通道配置为输入时 0：关闭捕获功能 1：使能捕获功能

19.6.10 计数值寄存器（TIM_CNT）

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CNT	R/W	0x0	计数器值

19.6.11 预分频寄存器（TIM_PSC）

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	PSC	R/W	0x0	计数器时钟（CK_CNT）预分频值： fCK_CNT=fCK_PSC/(PSC[15:0]+1) 这是一个 preload 寄存器，在 update 事件发生时其内容被载入 shadow 寄存器

19.6.12 自动重载寄存器（TIM_ARR）

偏移地址：0x2C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	ARR	R/W	0x0	计数溢出时的自动重载值： 这是一个 preload 寄存器，在 update 事件发生时其内容被载入 shadow 寄存器

19.6.13 捕获/比较寄存器 1（TIM_CCR1）

偏移地址：0x34

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CCR1	R/W	0x0	捕获/比较通道 1 寄存器： 如果通道 1 配置为输出： 这是一个 preload 寄存器，其内容被载入 shadow 寄存器后用于与计数器比较产生 OC1 输出 如果通道 1 配置为输入： CCR1 保存最近一次输入捕获事件发生时的计数器值，此时 CCR1 为只读

19.6.14 捕获/比较寄存器 2（TIM_CCR2）

偏移地址：0x38

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CCR2	R/W	0x0	捕获/比较通道 2 寄存器： **如果通道 2 配置为输出：** 这是一个 preload 寄存器，其内容被载入 shadow 寄存器后用于与计数器比较产生 OC2 输出 **如果通道 2 配置为输入：** CCR2 保存最近一次输入捕获事件发生时的计数器值，此时 CCR2 为只读

19.6.15 捕获/比较寄存器 3（TIM_CCR3）

偏移地址：0x3C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CCR3	R/W	0x0	捕获/比较通道 3 寄存器： **如果通道 3 配置为输出：** 这是一个 preload 寄存器，其内容被载入 shadow 寄存器后用于与计数器比较产生 OC3 输出 **如果通道 3 配置为输入：** CCR3 保存最近一次输入捕获事件发生时的计数器值，此时 CCR3 为只读

19.6.16 捕获/比较寄存器 4（TIM_CCR4）

偏移地址：0x40

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CCR4	R/W	0x0	捕获/比较通道 4 寄存器： **如果通道 4 配置为输出：** 这是一个 preload 寄存器，其内容被载入 shadow 寄存器后用于与计数器比较产生 OC4 输出 **如果通道 4 配置为输入：** CCR4 保存最近一次输入捕获事件发生时的计数器值，此时 CCR4 为只读

19.6.17 DMA 控制寄存器（TIM_DCR）

偏移地址：0x48

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12:8	DBL	R/W	0x0	DMA Burst 长度。 对 TIM_DMAR 寄存器的读写将触发 burst DMA 操作，burst 长度为 1~18： 00000：长度=1 00001：长度=2 00010：长度=3 …… 10001：长度=18 其他：无效值，禁止写入
7:5	RSV	-	-	保留
4:0	DBA	R/W	0x0	DMA 基地址，定义指向寄存器的偏移地址： 00000：TIM_CR1 00001：TIM_CR2 00010：TIM_SMCR …… 注：当 DBA+DBL 超出了 TIM 寄存器地址范围，则实际 burst 传输到 TIM 最高寄存器地址后自动停止，即 burst 长度会缩短。

19.6.18 DMA 访问寄存器（TIM_DMAR）

偏移地址：0x4C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	DMAR	R/W	0x0	DMA burst 访问寄存器： 在使用 DMA burst 传输时，将 DMA 通道外设地址设置到 TIM_DMAR，对此寄存器的访问会指向 TIM_DCR 中指定的寄存器，TIM 会根据 DBL 的值产生多次 DMA 请求

19.7 使用流程

19.7.1 定时计数模式

1. RCM 模块中使能 TIMx 时钟。
2. 配置 TIM_CR1[4]，设置计数方向。
3. 配置 TIM_CR1[7]为 1，使能 Auto-reload 预装载。
4. 配置 TIM_PSC[15:0]，设置预分频值。
5. 配置 TIM_ARR[31:0]，设置自动重载值。
6. 配置 TIM_CR1[2]为 1，设置仅计数器上溢出或下溢出会产生 update 中断或 DMA 请求。

7. 配置 TIM_CR1[1]为 0，使能 update 事件。
8. 配置 TIM_EGR[0]为 1，软件置位 TIM_EGR[0]时会重新初始化计数器并更新 shadow 寄存器，预分频计数器被清零。
9. 配置 TIM_CR1[0]为 1，使能计数器。
10. 配置 TIM_DIER[0]为 1，允许 Update 事件中断。

19.7.2 PWM 模式

1. 配置 GPIO 复用，RCM 模块中使能 TIMx 时钟。
2. 配置 TIM_CR1[4]，设置计数方向。
3. 配置 TIM_CR1[7]为 1，使能 Auto-reload 预装载。
4. 配置 TIM_PSC[15:0]，设置预分频值。
5. 配置 TIM_ARR[31:0]，设置自动重载值。
6. 根据输出通道配置 TIM_CCMRx[1:0/9:8]为 0，设置通道 x 为输出。
7. 配置 TIM_CCMRx[6:4/14:12]，设置为 PWM 模式 1//2。
8. 配置 TIM_CCER[1/5/9/13]，设置输出极性。
9. 配置 TIM_CCER[0/4/8/12]为 1，通道 x 输出使能。
10. 配置 TIM_CR1[2]为 1，设置仅计数器上溢出或下溢出会产生 update 中断或 DMA 请求。
11. 配置 TIM_CR1[1]为 0，使能 update 事件。
12. 配置 TIM_EGR[0]为 1，软件置位 TIM_EGR[0]时会重新初始化计数器并更新 shadow 寄存器，预分频计数器被清零。
13. 配置 TIM_CR1[0]为 1，使能计数器。
14. 配置 TIM_DIER[0]为 1，允许 Update 事件中断。
15. 配置 TIM_CCRx[31:0]，设置通道 x 的比较值。

19.7.3 输入捕获模式

1. 配置 GPIO 复用，RCM 模块中使能 TIMx 时钟。
2. 配置 TIM_CR1[4]，设置计数方向。
3. 配置 TIM_CR1[7]为 1，使能 Auto-reload 预装载。
4. 配置 TIM_PSC[15:0]，设置预分频值。
5. 配置 TIM_ARR[31:0]，设置自动重载值。
6. 配置 TIM_CCMRx[1:0/9:8]，设置 CCx 通道为输入，并更根据需求映射。
7. 配置 TIM_CCER[1/5/9/13]，设置捕获极性。
8. 配置 TIM_CCMRx[7:4/15:12]，设置采样频率和滤波长度，一般设置为 0 即可。

9. 配置 TIM_CCMRx[3:2/11:10]，设置输入捕获预分频。
10. 配置 TIM_CCER[0/4/8/12]为 1，使能捕获功能。
11. 配置 TIM_EGR[0]为 1，软件置位 TIM_EGR[0]时会重新初始化计数器并更新 shadow 寄存器，预分频计数器被清零。
12. 配置 TIM_CR1[0]为 1，使能计数器。
13. 配置 TIM_DIER[1/2/3/4]为 1，允许通道 x 捕获中断。

19.7.4 编码器接口模式

1. 配置 GPIO 复用，RCM 模块中使能 TIMx 时钟。
2. 配置 TIM_CR[4]，设置计数方向。
3. 配置 TIM_CR1[7]为 1，使能 Auto-reload 预装载。
4. 配置 TIM_PSC[15:0]，设置预分频值。
5. 配置 TIM_ARR[31:0]，设置自动重载值。
6. 配置 TIM_CCMR1[1:0]为 1，设置 CC1 通道为输入，IC1 映射到 TI1。
7. 配置 TIM_CCMR1[9:8]为 1，设置 CC2 通道为输入，IC2 映射到 TI2。
8. 配置 TIM_CCER[1]，设置捕获极性。
9. 配置 TIM_CCER[5]，设置捕获极性。
10. 配置 TIM_CCMR1[7:4]，设置采样频率和滤波长度，一般设置为 0 即可。
11. 配置 TIM_CCMR1[15:12]，设置采样频率和滤波长度，一般设置为 0 即可。
12. 配置 TIM_SMCR[2:0]，设置 Encoder 模式 1/2/3。
13. 配置 TIM_CCER[0]为 1，使能通道 1 捕获功能。
14. 配置 TIM_CCER[4]为 1，使能通道 2 捕获功能。
15. 配置 TIM_EGR[0]为 1，软件置位 TIM_EGR[0]时会重新初始化计数器并更新 shadow 寄存器，预分频计数器被清零。
16. 配置 TIM_CR1[0]为 1，使能计数器。
17. 配置 TIM_DIER[1]为 1，允许通道 1 捕获中断。

19.7.5 DMA 模式

输入捕获模式下，TIMx 的通道捕获值通过 DMA 传输到 SRAM：

1. 在输入捕获模式中软件置位 TIM_EGR[0]和使能计数器前，补充以下配置。
2. 配置 TIM_DCR[12:8]，设置 DMA Burst 长度。
3. 配置 TIM_DCR[4:0]，设置 DMA 基地址，一般该处的基地址选择相应捕获通道对应的捕获/比较寄存器。

4. 配置 TIM_DIER[9/10/11/12]为 1，允许 CCx DMA 请求。
5. 配置 TIM_CR2[3]为 0，发生 CCx 事件时产生 CCxDMA 请求。
6. DMA 控制器配置详细请看第 11 章节。
7. 开启 DMA 传输后，当通道发生捕获时，DMA 将基地址存放的值传到 SRAM 中。

输出比较模式下，SRAM 中的值通过 DMA 传输到 TIMx 的比较寄存器：

1. 在 PWM 模式中软件置位 TIM_EGR[0]和使能计数器前，补充以下配置。
2. 配置 TIM_DCR[12:8]，设置 DMA Burst 长度。
3. 配置 TIM_DCR[4:0]，设置 DMA 基地址，一般该处的基地址选择相应比较通道对应的捕获/比较寄存器。
4. 配置 TIM_DIER[9/10/11/12]为 1，允许 CCx DMA 请求。
5. 配置 TIM_CR2[3]为 0，发生 CCx 事件时产生 CCxDMA 请求。
6. DMA 控制器配置详细请看第 11 章节。
7. 开启 DMA 传输后，当计数器计数值等于比较值时，DMA 将 SRAM 中的值传到基地址。

20 基本定时器（TIM5 & TIM6）

20.1 概述

包含一个 32bit 自动重载计数器及一个可编程预分频器。

20.2 主要特性

- 32bit 向上自动重载计数器
- 16bit 可编程预分频器，支持实时调整计数时钟分频
- 支持在以下事件发生时产生中断
  - 计数器溢出，计数器初始化（软件或硬件 trigger）

20.3 功能描述

20.3.1 定时单元

定时单元由一个 32 位计数器和自动重载寄存器组成。计数时钟可以通过 16 位预分频器对时钟进行分频后得到。

计数器、自动重载寄存器预分频寄存器都可以由软件改写或读取，即使在计数器正在运行时也是如此。

定时单元包含如下寄存器：

- 计数器（TIM_CNT）
- 预分频寄存器（TIM_PSC）
- 自动重载寄存器（TIM_ARR）

ARR 包含预装载功能，该功能通过 ARPE（Auto Reload Preload Enable）寄存器控制。当 ARPE=0 时，对 ARR 寄存器执行写入，写入数据将直接传入到影子寄存器；当 ARPE=1 时，对 ARR 寄存器执行写入的数据在 update event（TIM_CNT 上溢出或者下溢出）发生时，传送到影子寄存器。软件也可以通过寄存器操作主动触发 ARR 更新（UEV）。

TIM_CNT 工作时钟由 TIM_PSC 产生的分频时钟驱动，只有在计数器使能寄存器（CEN）置位时，CNT 才开始计数。当 CNT=ARR 时，本轮计数结束，发送 update event。

TIM_PSC 是一个同步预分频器，能够对时钟进行 1~65536 分频。PSC 寄存器同样被缓存，改写 PSC 实际不改写影子寄存器，只有当新的 update event 到来时，才会从 PSC 更新至影子寄

存器。因此在 CNT 计数过程中，软件可以实时改写 PSC，而新的预分频比将在下一更新事件发生时被采用。

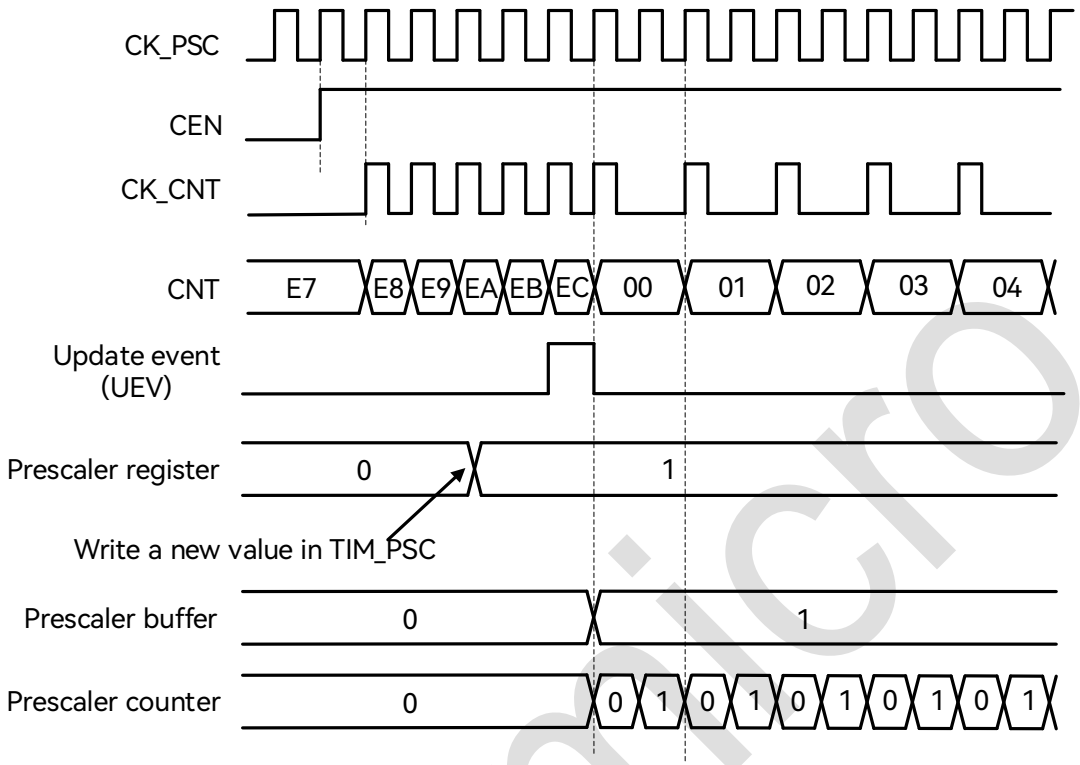


图 20-1：预分频从 1 变为 2 的波形

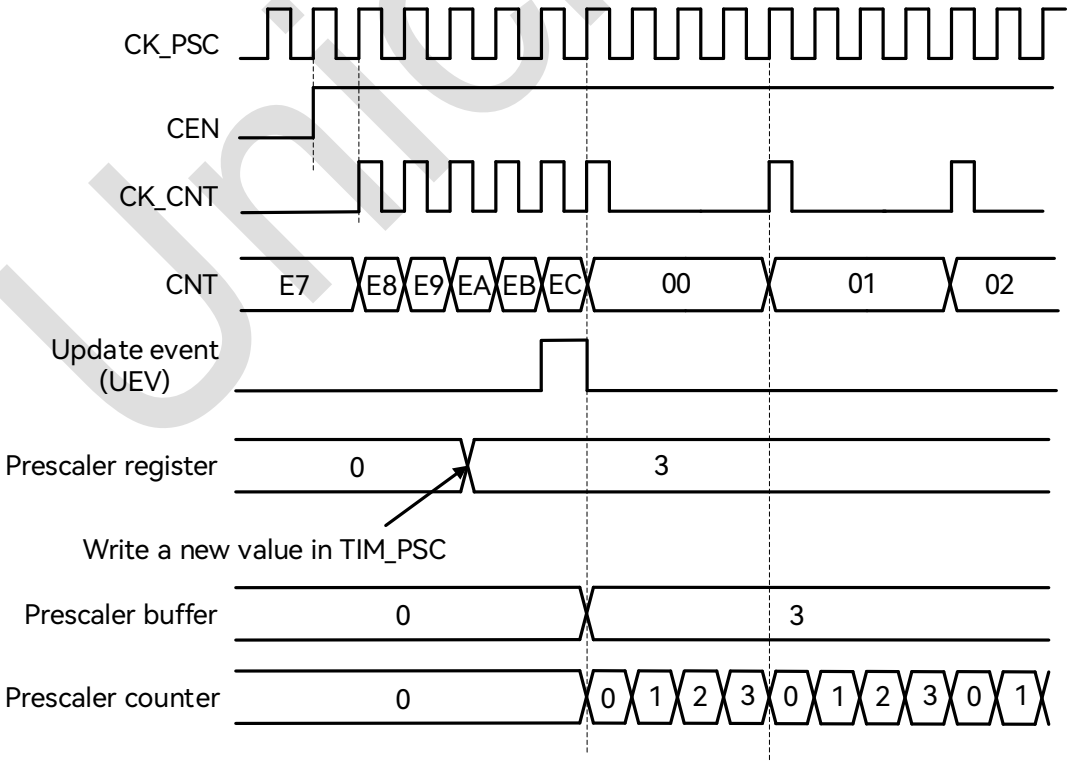


图 20-2：预分频从 1 变为 4 的波形

20.3.2 定时器工作模式

定时器支持向上计数计数模式。

此模式中，计数器使能后从 0 开始计数，直到 CNT=ARR，产生溢出事件，然后重新从 0 开始计数。

如果使能了重复计数功能，则计数器按照 RCR 的定义重复上述过程若干次（RCR+1），才会产生溢出事件。

软件可以通过设置 UG 寄存器直接触发 update event，此时 CNT 和预分频计数器自动清零。设置 UG 寄存器是否触发 UIF（Update Interrupt Flag）中断标志置位由 URS 寄存器的设置决定。

通过设置 UDIS 寄存器可以禁止 update event，这样可以避免将 preload 寄存器中的值更新到工作寄存器中。

当 update event 发生时，以下寄存器被更新，并且 UIF 置位：

- ARR 影子寄存器被更新为 TIM_ARR 内容
- PSC 影子寄存器被更新为 TIM_PSC 内容

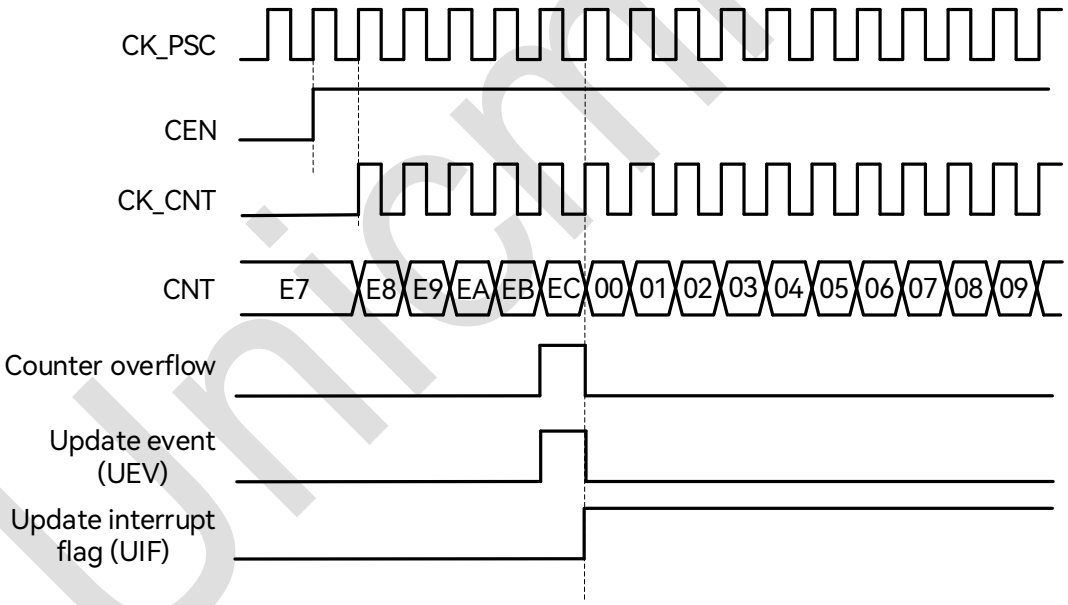


图 20-3：向上计数波形，内部时钟不分频

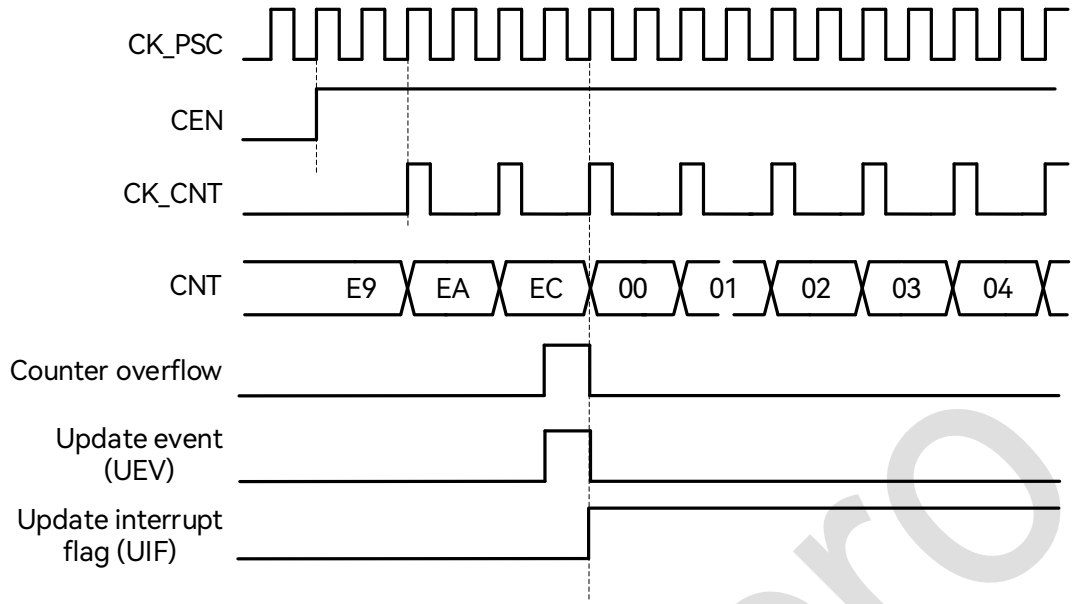


图 20-4：向上计数波形，内部时钟 2 分频

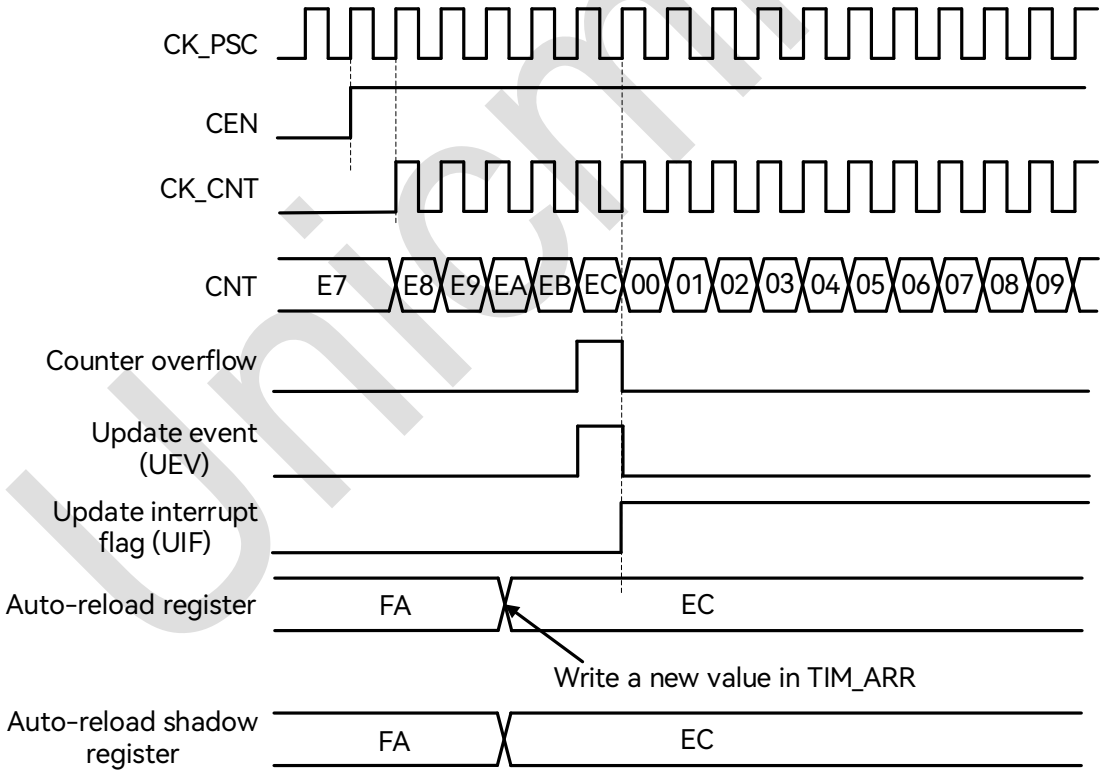


图 20-5：ARPE=0（TIM_ARR 没有预装载）时的更新事件

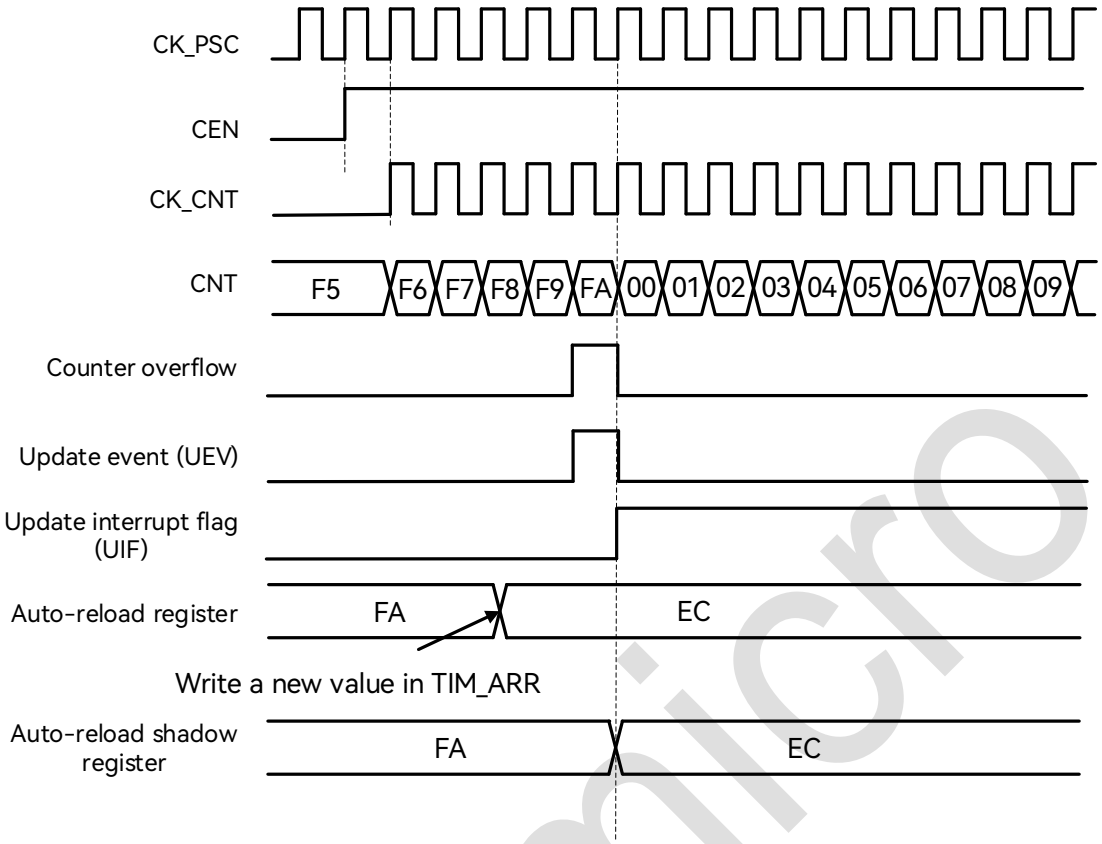


图 20-6：ARPE=1（TIM_ARR 预装载）时的更新事件

20.3.3 Preload 寄存器

- 以下功能寄存器支持 preload 功能：
  - 自动重载寄存器 TIM_ARR
  - 预分频寄存器 TIM_PSC（不可关闭 preload 功能）
  - 通道控制寄存器 TIM_CCR
  - CCxE 和 CCxNE 控制寄存器
  - OCxM 控制寄存器

  以上寄存器，除了 PSC 之外，都可以由软件选择使能或者禁止 preload 功能。
- 具备 preload 功能的寄存器，包含两组物理实体：
  - Shadow register（影子寄存器）：实际定时器正在使用的寄存器
  - Preload register（预装载寄存器）：软件可以访问的寄存器
- 当禁止 preload 时，具备 preload 功能的寄存器特性如下：
  - Preload 寄存器可以实时由软件访问、改写
  - Shadow 寄存器与 Preload 寄存器同步更新
- 如果使能了 preload，则：

- 所有软件操作访问的是 preload 寄存器
- 当 update event 发生时，所有 preload 寄存器内容将同步被转移到对应的 shadow 寄存器

20.3.4 计数器工作时钟

计数器可以使用如下时钟工作：

Timerx_clk——内部时钟模式

20.3.4.1 内部时钟模式

内部时钟模式下，禁止从机模式（SMS=000），CEN、DIR、UG 等寄存器位都是软件控制。软件操作 UG 寄存器后，update 信号经过 CLK_PSC 同步后，计数器值将被重新初始化。

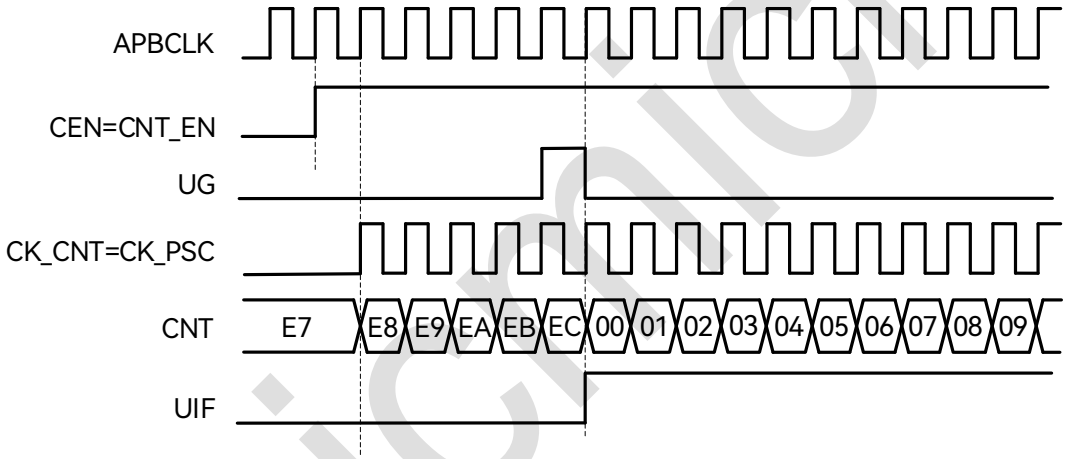


图 20-7：内部时钟源模式，时钟分频因子为 1

20.3.5 Debug 模式

当 CPU 进入 debug 模式后，定时器可以停止或继续工作，其行为由芯片系统中的寄存器定义。

Debug 时当定时器被停止后，其输出会被禁止（MOE 清零），根据寄存器配置，此时的输出信号可以被 force 成 inactive 或由 GPIO 模块控制。

20.4 寄存器描述

TIM5 寄存器基地址：0x4600_B000

TIM6 寄存器基地址：0x4600_B400

寄存器列表如下：

表 20-1：TIM5 & TIM6 寄存器列表

偏移地址	名称	描述
0x00	TIM_CR1	控制寄存器 1
0x04	TIM_CR2	控制寄存器 2
0x0C	TIM_DIER	DMA 和中断使能寄存器
0x10	TIM_SR	状态寄存器
0x14	TIM_EGR	事件产生寄存器
0x24	TIM_CNT	计数值寄存器
0x28	TIM_PSC	预分频寄存器
0x2C	TIM_ARR	自动重载寄存器

以下各节详细介绍寄存器。

20.4.1 控制寄存器 1 (TIM_CR1)

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	APRE	R/W	0x0	Auto-reload 预装载使能： 0：ARR 寄存器不使能 preload 1：ARR 寄存器使能 preload
6:4	RSV	-	-	保留
3	OPM	R/W	0x0	单脉冲输出模式： 0：Update Event 发生时计数器不停止 1：Update Event 发生时计数器停止（自动清零 CEN）
2	URS	R/W	0x0	更新请求选择： 0：以下事件能够产生 update 中断或 DMA 请求 ● 计数器上溢出或下溢出 ● 软件置位 UG 寄存器 ● 从机控制器产生 update 1：仅计数器上溢出或下溢出会产生 update 中断或 DMA 请求
1	UDIS	R/W	0x0	禁止 update： 0：使能 update 事件；以下事件发生时产生 update 事件 ● 计数器上溢出或下溢出 ● 软件置位 UG 寄存器 1：禁止 update 事件，不更新 shadow 寄存器。当 UG 置位或从机控制器收到硬件 reset 时重新初始化计数器和预分频器。

位	名称	属性	复位值	描述
0	CEN	R/W	0x0	计数器使能： 0：计数器关闭 1：计数器使能 注意：外部触发模式可以自动置位 CEN

20.4.2 控制寄存器 2（TIM_CR2）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6:4	MMS	R/W	0x0	主机模式选择，选择 TRGO 触发的方式： 000：复位：EGR 寄存器中的 UG 位产生 TRGO 001：使能：TRGO 由计数器使能信号产生，包括 CEN 位和外部触发 010：更新：更新事件产生 TRGO 011：比较脉冲：发生输入捕获或比较事件，使 CC1F 置 1 时，产生 TRGO 100：比较：OC1REF 产生 TRGO 101：比较：OC2REF 产生 TRGO 110：比较：OC3REF 产生 TRGO 111：比较：OC4REF 产生 TRGO
3:0	RSV	-	-	保留

20.4.3 DMA 和中断使能寄存器（TIM_DIER）

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	UDE	R/W	0x0	Update Event DMA 请求使能： 0：Update Event 发生时，禁止产生 DMA 请求 1：Update Event 发生时，允许产生 DMA 请求
7:1	RSV	-	-	保留
0	UIE	R/W	0x0	Update 事件中断使能： 0：禁止 Update 事件中断 1：允许 Update 事件中断

20.4.4 状态寄存器（TIM_SR）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	UIF	R/W0C	0x0	Update 事件中断标志，硬件置位，软件写 0 清零。 当以下事件发生时，UIF 置位，并更新 shadow 寄存器： ● 重复计数器=0，并且 UDIS=0 的情况下，计数器发生溢出 ● URS=0 且 UDIS=0 的情况下，软件置位 UG 寄存器初始化计数器 ● URS=0 且 UDIS=0 的情况下，触发事件初始化计数器

20.4.5 事件产生寄存器（TIM_EGR）

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	UG	W	0x0	软件 Update 事件，软件置位此寄存器产生 Update 事件，硬件自动清零 软件置位 UG 时会重新初始化计数器并更新 shadow 寄存器，预分频计数器被清零。

20.4.6 计数值寄存器（TIM_CNT）

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CNT	R/W	0x0	计数器值

20.4.7 预分频寄存器（TIM_PSC）

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	PSC	R/W	0x0	计数器时钟（CK_CNT）预分频值： fCK_CNT=fCK_PSC/(PSC[15:0]+1) 这是一个 preload 寄存器，在 update 事件发生时其内容被载入 shadow 寄存器

20.4.8 自动重载寄存器（TIM_ARR）

偏移地址：0x2C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	ARR	R/W	0x0	计数溢出时的自动重载值： 这是一个 preload 寄存器，在 update 事件发生时其内容被载入 shadow 寄存器

20.5 使用流程

20.5.1 启动计数器

1. RCM 模块中使能 TIMx 时钟。
2. 配置 TIM_ARR[31:0]，设置自动重载值。
3. 配置 TIM_PSC[15:0]，设置预分频值。
4. 配置 TIM_EGR[0]为 1，产生软件 Update 事件更新 shadow 寄存器。
5. 配置 TIM_SR[0]为 0，清除 Update 事件中断标志。
6. 配置 TIM_CR1[0]为 1，使能计数器。

21 低功耗定时器/计数器（LPTIM）

21.1 概述

LPTIM 是运行在 Always-On 电源域下的 16 位低功耗定时器/计数器模块。通过选择合适的工作时钟，LPTIM 在各种低功耗模式下保持运行，并且只消耗很少的电量。LPTIM 可以在没有内部时钟的条件下实现低功耗模式下的外部脉冲计数功能。此外，与外部输入的触发信号结合，可以实现低功耗超时唤醒功能。

本芯片有 2 个 LPTIM(LPTIM0/LPTIM1)。LPTIM 0/1 在 sleep、stop 模式下，可以通过中断唤醒。进入 Standby0 模式后，LPTIM0 计数溢出中断、比较匹配中断、外部脉冲计数中断可以唤醒（LPTIM0 产生的所有中断，都可以从 standby0 唤醒）。而 LPTIM1 在 standby0 下只支持内部计数溢出中断唤醒和比较匹配中断唤醒，外部脉冲中断不可以唤醒。

21.2 主要特性

- 16 位递增计数器
- 3 位异步时钟分频器，8 种分频系数（1、2、4、8、16、32、64、128）
- 可选工作时钟源：
  - 内部时钟源：LSCLK（RCL 或者 XTL）、CLK1HZ、APB0 时钟（PCLK0）
  - 外部时钟源：LPTIMx_IN
- 16 位比较寄存器
- 16 位目标寄存器
- 软件/硬件触发
- 输入极性选择
- 无时钟外部脉冲计数
- 外部触发的低功耗超时唤醒
- 定时唤醒
- 支持 16 位 PWM

21.3 系统框图

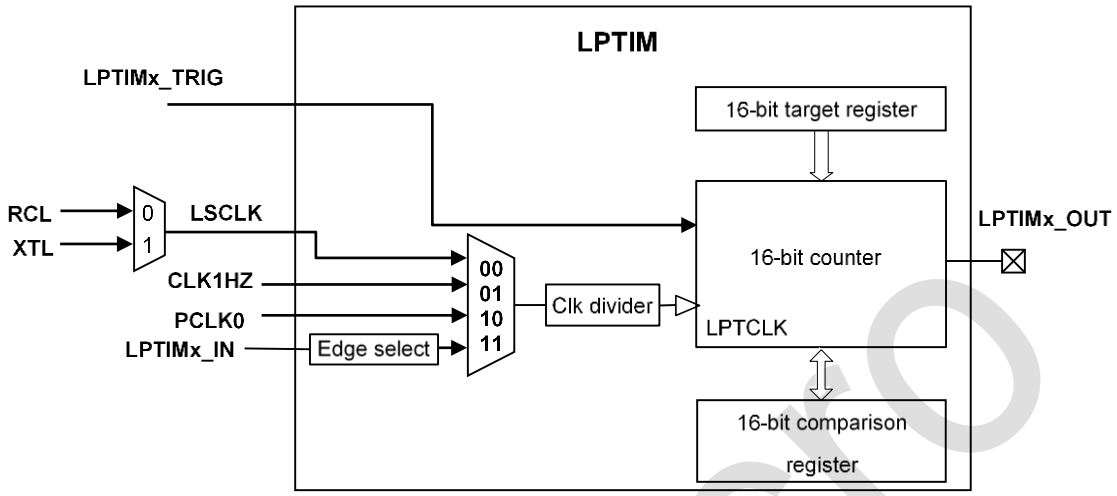


图 21-1：系统框图

21.4 管脚说明

表 21-1：LPTIM 管脚说明

功能管脚	复用管脚	方向	功能描述
LPTIM0_IN	PC0	Input	LPTimer 信号输入（用于外部异步脉冲计数模式）
LPTIM0_TRIG	PC3	Input	LPTimer 时钟输入（输入捕获）
LPTIM0_OUT	PC1	Output	LPTimer 信号输出（PWM)
LPTIM1_IN	PD10	Input	LPTimer 信号输入（用于外部异步脉冲计数模式）
LPTIM1_TRIG	PD11	Input	LPTimer 时钟输入（输入捕获）
LPTIM1_OUT	PE4	Output	LPTimer 信号输出（PWM)

注：待机模式（Standby0 Mode），LPTIM0 可以通过 PC0、PC3 管脚唤醒系统。LPTIM1 只能内部计时唤醒，不能 PD10/PD11 外部管脚唤醒。

21.5 功能描述

21.5.1 普通定时器

LPTIM 可以使用内部或外部时钟进行工作，使能后有两个计数时钟周期的同步过程，然后开始工作。

21.5.2 Trigger 脉冲触发计数

使用内部时钟工作，采样外部输入的异步 Trigger 信号，可以对 Trigger 信号的上升、下降、双边沿计数。当脉冲数量达到设定的比较值或溢出后，相应的中断标志位会置 1。使能前后有两个计数时钟周期的同步过程。

21.5.3 外部异步脉冲计数

直接使用外部输入脉冲作为工作时钟，极性可配置为上升沿或下降沿，使能后立即开始工作无需同步过程。

21.5.4 Timeout 模式

使用内部或外部时钟工作，采样外部输入的异步 Trigger 信号。首次采样到 Trigger 信号后启动计数器，启动后采样到 Trigger 信号则清零并重启计数器。如果计数器在溢出前没有采样到 Trigger 信号，则产生溢出中断并停止计数，清除使能。使能后有两个计数时钟周期的同步过程。

21.5.5 计数模式

LPTIM 有两种计数模式：

连续计数模式：计数器使能后保持运行，直到被关闭为止。计数器达到目标值后回到 0 重新开始计数，并产生溢出中断。

单次计数模式：计数器被触发后计数到目标值后回到 0，并自动停止，产生溢出中断。由于溢出信号和 Lpten 使能信号位于不同的时钟域，关闭使能信号采用异步复位同步释放的方式实现。

21.5.6 外部触发的超时唤醒

LPTIM 可以由外部输入的 trigger 信号触发使能，也可以由软件触发使能。在 Timeout 模式下，第一个外部触发输入的有效沿将启动计数器，而后续触发信号将清零计数器。如果在计数器达到比较值之前没有有效触发信号到来，则产生超时中断，唤醒 MCU。

外部输入 trigger 信号的有效沿可以由寄存器配置，外部 trigger 信号被认为是一个异步输入，因此有效沿的采样和判决有至少 2 个计数时钟周期的延迟。

21.5.7 16 位 PWM

使能 PWM 模式后 LPTIM 从 0x0000 开始计数，计数值等于比较值时输出置高，计数值等于目标值寄存器时输出变低；PWM 周期由目标值寄存器决定，占空比由比较值寄存器决定。

21.6 寄存器描述

LPTIM0 寄存器基地址：0x40B0_9000

LPTIM1 寄存器基地址：0x40B0_A000

寄存器列表如下：

表 21-2：LPTIM 寄存器列表

偏移地址	名称	描述
0x00	LPTIM_CFG	配置寄存器
0x04	LPTIM_CNT	计数值寄存器
0x08	LPTIM_CMP	比较值寄存器
0x0C	LPTIM_TARGET	目标值寄存器
0x10	LPTIM_IE	中断使能寄存器
0x14	LPTIM_IF	中断状态寄存器
0x18	LPTIM_CTRL	控制寄存器

以下各节详细介绍寄存器。

21.6.1 配置寄存器（LPTIM_CFG)

偏移地址：0x00

复位值：0x0000 0200

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	FLTEN	R/W	0x0	外部触发信号滤波使能： 使能后，将滤除维持时间短于 2 个计数时钟周期的外部触发信号 0：禁用滤波功能 1：启用滤波功能
14:13	RSV	-	-	保留
12:10	DIVSEL	R/W	0x0	计数时钟分频选择： 000：1 分频 001：2 分频 010：4 分频 011：8 分频 100：16 分频 101：32 分频 110：64 分频 111：128 分频

位	名称	属性	复位值	描述
9:8	CLKSEL	R/W	0x2	计数时钟源选择： 00：选择 LSCLK（即系统低速时钟 RCL 或 XTL）作为计数时钟 01：选择 RCLP（CLK_1Hz）作为计数时钟 10：选择 LPTIM 的 PCLK 作为计数时钟 11：选择 LPTIM（由 SYSCFG->SYSCTRL1[11] 选择的外部引脚或 CLK_1Hz）作为计数时钟
7	EDGESEL	R/W	0x0	LPTIM 输入边沿选择： 0：LPTIM 的上升沿计数 1：LPTIM 的下降沿计数
6:5	TRIGCFG	R/W	0x0	外部触发边沿选择： 00：外部输入信号上升沿 trigger 01：外部输入信号下降沿 trigger 10/11：外部输入信号上升下降沿 trigger
4	POLARITY	R/W	0x0	计数时钟分频选择： 0：正极性波形，即第一次计数值=比较值时产生输出波形上升沿 1：负极性波形，即第一次计数值=比较值时产生输出波形下降沿
3	PWM	R/W	0x0	脉宽调制模式： 0：周期方波输出模式 1：PWM 输出模式
2	MODE	R/W	0x0	计数模式： 0：连续计数模式：计数器被触发后保持运行，直到被关闭为止。计数器达到目标值后回到 0 重新开始计数，并产生溢出中断。 1：单次计数模式：计数器被触发后计数到目标值后回到 0，并自动停止，产生溢出中断。
1:0	TMODE	R/W	0x0	工作模式选择： 00：带波形输出的普通定时器模式 01：Trigger 脉冲触发计数模式 10：外部异步脉冲计数模式 11：Timeout 模式

21.6.2 计数值寄存器（LPTIM_CNT)

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	CNT	R	0x0	计数器数值

21.6.3 比较值寄存器 (LPTIM_CMP)

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	CMP	R/W	0x0	比较值

21.6.4 目标值寄存器 (LPTIM_TARGET)

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	TARGET	R/W	0x0	目标值

21.6.5 中断使能寄存器 (LPTIM_IE)

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	TRIGIE	R/W	0x0	外部触发到来中断使能位： 1：外部触发到来中断使能 0：外部触发到来中断禁止
1	OVIE	R/W	0x0	计数器溢出中断使能位： 1：计数器溢出中断使能 0：计数器溢出中断禁止
0	COMPIE	R/W	0x0	比较匹配中断使能位： 1：计数器值和比较值匹配中断使能 0：计数器值和比较值匹配中断禁止

21.6.6 中断标志寄存器 (LPTIM_IF)

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	TRIGIF	R/W1C	0x0	**外部触发到来中断标志位：** 1：外部触发到来中断产生 0：无中断产生
1	OVIF	R/W1C	0x0	**计数器溢出中断标志位：** 1：计数器溢出中断产生 0：无中断产生
0	COMPIF	R/W1C	0x0	**比较匹配中断标志位：** 1：计数器值和比较值匹配中断产生 0：无中断产生

21.6.7 控制寄存器 (LPTIM_CTRL)

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	LPTEN	R/W	0x0	**LPTIM 使能位：** 1：使能计数器计数 0：禁止计数器计数

21.7 使用流程

21.7.1 使能 LPTIM 时钟

1. 写 0xABCD 到 PMU_CPR 寄存器，使能 PMU 寄存器写操作。
2. 配置 PMU_FCCR[3:2]，使能 LPTIM0/1 控制器时钟。
3. 配置 PMU_FRCR[3:2]，释放 LPTIM0/1 控制器复位。
4. 写 0x459E 到 PMU_CPR 寄存器，结束 PMU 寄存器写操作。

21.7.2 LPTIM 计数

1. 使能 LPTIM 时钟。
2. 配置 LPTIM_CFG 寄存器，写 0 清空配置。
3. 配置 LPTIM_CFG[12:10]设置时钟分频系数，配置 LPTIM_CFG [9:8]设置 LPTIM 时钟源，配置 LPTIM_CFG [2]设置 LPTIM 计数模式，配置 LPTIM_CFG [1:0]设置 LPTIM 工作模式（带波形输出的普通定时器模式）。
4. 配置 LPTIM_TARGET[15:0]，设置计数目标值。

5. 配置 LPTIM_IE[1]，使能 LPTIM 溢出中断。
6. 配置 LPTIM_CTRL[0]，使能 LPTIM。

21.7.3 LPTIM PWM 输出

1. 使能 LPTIM 时钟。
2. 配置 LPTIM_OUT 所在引脚，使能其时钟并将其功能复用为 LPTIM_OUT。
3. 配置 LPTIM_CFG 寄存器，写 0 清空配置。
4. 配置 LPTIM_CFG[12:10]设置时钟分频系数，配置 LPTIM_CFG [9:8]设置 LPTIM 时钟源，配置 LPTIM_CFG [2]设置 LPTIM 计数模式，配置 LPTIM_CFG [1:0]设置 LPTIM 工作模式（带波形输出的普通定时器模式），将 PWM 位脉宽调制模式设置为 PWM 输出模式。
5. 配置 LPTIM_CMP[15:0]，设置计数比较值。
6. 配置 LPTIM_TARGET[15:0]，设置计数目标值。
7. 配置 LPTIM_IE[0]，使能 LPTIM 比较匹配中断。
8. 配置 LPTIM_CTRL[0]，使能 LPTIM。

21.7.4 LPTIM 外部触发计数中断

1. 使能 LPTIM 时钟。
2. 配置 LPTIM_TRIGGER 所在引脚，使能其时钟并将其功能复用为 LPTIM_TRIGGER。
3. 配置 LPTIM_CFG 寄存器，写 0 清空配置。
4. 配置 LPTIM_CFG[12:10]设置时钟分频系数，配置 LPTIM_CFG [9:8]设置 LPTIM 时钟源，配置 LPTIM_CFG [2]设置 LPTIM 计数模式，配置 LPTIM_CFG [1:0]设置 LPTIM 工作模式（Trigger 脉冲触发计数模式），配置 TRIGCFG 位设置外部触发信号触发边沿，将 FLTEN 位置 1，开启触发信号滤波使能。
5. 配置 LPTIM_TARGET[15:0]，设置计数目标值。
6. 配置 LPTIM_IE[2，使能 LPTIM 触发中断。
7. 配置 LPTIM_CTRL[0]，使能 LPTIM。

21.7.5 LPTIM Timeout 模式

1. 使能 LPTIM 时钟。
2. 配置 LPTIM_TRIGGER 所在引脚，使能其时钟并将其功能复用为 LPTIM_TRIGGER。
3. 配置 LPTIM_CFG 寄存器，写 0 清空配置。
4. 配置 LPTIM_CFG[12:10]设置时钟分频系数，配置 LPTIM_CFG [9:8]设置 LPTIM 时钟源，配置 LPTIM_CFG [2]设置 LPTIM 计数模式，配置 LPTIM_CFG [1:0]设置 LPTIM 工作模式（TIMEOUT 模

式），配置 TRIGCFG 位设置外部触发信号触发边沿，将 FLTEN 位置 1，开启触发信号滤波使能。

5. 配置 LPTIM_TARGET 寄存器，设置计数目标值。
6. 配置 LPTIM_IE 寄存器的 OVIE 位，使能 LPTIM 溢出中断。
7. 配置 LPTIM_CTRL 寄存器的 LPTEN 位，使能 LPTIM。

21.7.6 LPTIM 外部时钟源计数溢出中断

1. 使能 LPTIM 时钟。
2. 配置 LPTIM_IN 所在引脚，使能其时钟并将其功能复用为 LPTIM_IN。
3. 配置 LPTIM_CFG 寄存器，写 0 清空配置。
4. 配置 LPTIM_CFG 寄存器的 CLKSEL、DIVSEL、TMODE 位，对应设置 LPTIM 时钟源（CLKSEL 位设置为 11），时钟分频系数及 LPTIM 工作模式（外部异步脉冲计数模式），配置 EDGESEL 位，设置外部时钟源计数边沿。
5. 配置 LPTIM_TARGET[15:0]，设置计数目标值。
6. 配置 LPTIM_IE[1]，使能 LPTIM 溢出中断。
7. 配置 LPTIM_CTRL[0]，使能 LPTIM。

22 独立看门狗（IWDT）

22.1 概述

看门狗定时器在到达超时的值的时候可以产生不可屏蔽中断或者是复位。当系统由于软件错误或是由于外部设备故障而无法按照预期的方式响应的时候，使用看门狗定时器可以重新获得控制权。

22.2 主要特性

- 32 位递减并且可编程装载的定时器
- 独立的看门狗时钟使能
- 带中断屏蔽的中断生成逻辑
- 软件跑飞保护锁定寄存器
- 软件启动功能：IWDT 控制寄存器中复位使能/禁止的设置

22.3 寄存器描述

IWDT 寄存器基地址：0x40B0_6000

寄存器列表如下：

表 22-1：独立看门狗(IWDT)寄存器表

偏移地址	名称	描述
0x00	IWDT_LOAD	装载寄存器
0x04	IWDT_CNT	计数寄存器
0x08	IWDT_CTRL	控制寄存器
0x0C	IWDT_CLR	清除寄存器
0x10	IWDT_INTRAW	中断原始状态寄存器
0x14	IWDT_MINTS	中断状态寄存器
0x18	IWDT_STALL	时钟分频寄存器
0x1C	IWDT_LOCK	计数锁存寄存器

以下各节详细介绍寄存器。

22.3.1 装载寄存器 (IWDT_LOAD)

偏移地址：0x00

复位值：0x0000 3FFF

位	名称	属性	复位值	描述
31:0	LOAD	R/W	0x0000 3FFF	IWDOG初始装载值

22.3.2 计数寄存器（IWDT_CNT)

偏移地址：0x04

复位值：0x0000 3FFF

位	名称	属性	复位值	描述
31:0	CNT	R	0x0000 3FFF	IWDOG内部Cnt计数值。若时钟源频率是32768 Hz，则默认溢出时间约是0.5s。

22.3.3 控制寄存器（IWDT_CTRL)

偏移地址：0x08

复位值：0x8000 0004

位	名称	属性	复位值	描述
31	WRC	R	0x1	IWDT加载值设置或写IWDT_CTRL寄存器生效。向IWDT_LOAD或者IWDT_CTRL寄存器进行写操作时，设置位生效会有一定时间（3–4个不分频的IWDTCLK周期）的延时： 0：设置为仍未生效 1：设置位生效
30:3	RSV	-	-	保留
2	RST_MODE	R/W	0x1	IWDT溢出复位模式选择（未使能中断情况下）： 0：溢出复位首次需要计数两次 1：溢出复位需要计数一次
1	RSTEN	R/W	0x0	IWDT溢出复位使能： 0：不使能溢出复位功能 1：使能溢出复位功能
0	INTEN	R/W	0x0	IWDT中断使能： 0：不使能中断 1：使能中断 注：溢出中断优先级比溢出复位优先级高

注：

Rst_mode在设置为0时，未使能中断情况下，此模式首次计数溢出复位需要计数两次。

1. 若在第一次计数溢出前配置装载寄存器（IWDT_LOAD），则会重新计数两次，之后只要在第一次溢出前配置装载寄存器（IWDT_LOAD），则会重新计数两次；若有一次是在第一次计数溢出后配置装载寄存器（IWDT_LOAD），则之后的配置装载寄存器（IWDT_LOAD）只会复

位溢出计数一次。

2. 在第一次溢出后配置装载寄存器（IWDT_LOAD），则在之后每次溢出前配置装载寄存器（IWDT_LOAD），都只会重新计数一次溢出。

22.3.4 清除寄存器（IWDT_CLR)

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CLR_CARRY	W	0x0	向此寄存器写入任何值，将清除IWDT溢出状态，从而清除掉中断和复位，同时会将装载寄存器IWDT_LOAD值刷新到计数寄存器IWDT_CNT（相当于一次喂狗动作）。

22.3.5 中断原始状态寄存器（IWDT_INTRAW)

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	INTRAW	R	0x0	原始中断寄存器，未经中断使能屏蔽： 0：IWDT内部未发生溢出 1：IWDT内部发生溢出

22.3.6 中断状态寄存器（IWDT_MINTS)

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	INTMS	R	0x0	IWDT中断标志位： 0：IWDT未产生中断 1：IWDT产生中断

22.3.7 时钟分频寄存器（IWDT_STALL)

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	CLK_DIV	R/W	0x0	IWDT计数时钟分频值： 0x0：不分频 0x1：2分频 0x2：3分频 …. 0xFFFF：0xFFFF + 1分频
15:9	RSV	-	-	保留
8	STALL	R/W	0x0	IWDT在芯片处于HALT状态时不计数功能的使能位： 0：不使能HALT状态计数器停止工作功能 1：使能HALT状态计数器停止工作功能
7:0	RSV	-	-	保留

22.3.8 计数锁存寄存器（IWDT_LOCK)

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	LOCK	R/W	0x0	IWDT LOCK功能使能。默认不锁。 当使能LOCK功能时，除此寄存器外的所有WDT寄存器均不可写。向此寄存器写写除0x1ACCE551以外的任意值，使能IWDT LOCK功能，向此寄存器写0x1ACCE551清除LOCK功能。读此寄存器，返回1表示锁定状态，返回0表示解锁状态。

22.4 使用流程

22.4.1 使能 IWDT 时钟

1. 写 0xABCD 到 PMU_CPR 寄存器，使能 PMU 寄存器写操作。
2. 配置 PMU_FCCR[1]，使能 IWDT 控制器时钟。
3. 配置 PMU_FRCR[1]，释放 IWDT 控制器复位。
4. 写 0x459E 到 PMU_CPR 寄存器，结束 PMU 寄存器写操作。

22.4.2 IWDT 装载值设置

1. 写 0x1ACCE551 到 IWDT_LOCK 寄存器，解锁 IWDT 寄存器。
2. 配置 IWDT_LOAD[31:0]，设置 IWDT 计数装载值。

3. 等待 IWDT_CTRL[31]置位，即等待设置位生效。
4. 写除 0x1ACCE551 以外的任意值到 IWDT_LOCK 寄存器，锁定 IWDT 寄存器。

22.4.3 IWDT 计数溢出中断

1. 使能 IWDT 时钟。
2. 写 0x1ACCE551 到 IWDT_LOCK 寄存器，解锁 IWDT 寄存器。
3. 配置 IWDT_STALL[31:16]，设置 IWDT 计数时钟分频值。
4. 初始化 IWDT 计数溢出中断，将 IWDT_CTRL[1]置 1，使能 IWDT 中断。
5. 等待 IWDT_CTRL[31]置位，即等待设置位生效。
6. 配置 IWDT_LOAD[31:0]寄存器，设置 IWDT 计数装载值。
7. 等待 IWDT_CTRL[31]置位，即等待设置位生效。
8. 写除 0x1ACCE551 以外的任意值到 IWDT_LOCK 寄存器，锁定 IWDT 寄存器。

22.4.4 IWDT 计数溢出复位

1. 使能 IWDT 时钟。
2. 写 0x1ACCE551 到 IWDT_LOCK 寄存器，解锁 IWDT 寄存器。
3. 配置 IWDT_STALL[31:16]，设置 IWDT 计数时钟分频值。
4. 配置 IWDT_CTRL[2]，设置复位首次需要计数满一次或者复位首次需要计数满两次。
5. 等待 IWDT_CTRL[31]置位，即等待设置位生效。
6. 配置 IWDT_CTRL[1]，使能 IWDT 溢出复位。
7. 等待 IWDT_CTRL[31]置位，即等待设置位生效。
8. 写 IWDT_LOAD[31:0]，设置 IWDT 计数装载值。
9. 等待 IWDT_CTRL[31]置位，即等待设置位生效。
10. 写除 0x1ACCE551 以外的任意值到 IWDT_LOCK 寄存器，锁定 IWDT 寄存器。

23 窗口看门狗（WWDT）

23.1 概述

带窗口的看门狗是一个与 CPU 同步运行的看门狗，目的是实时监控 CPU 运行状态，在 CPU 运行异常的情况下复位 CPU，避免不可预计的后果。

为了保证同步性和实时性，WWDT 使用 PCLK 时钟工作，内部有一个预分频电路，以产生同步计数使能信号。

23.2 主要特性

- 计数时钟 PCLK0(APB0 PCLK)
- 计数模式为从 0 开始向上计数到溢出时间
- 溢出时间可选择 1/4/16/64/128/256/512/1024/2048/4096/8192/16384/32768/65536 个 4096 倍的 PCLK0 周期
- 窗口开放期为计数器大于或等于溢出时间的 50%
- 支持预警中断功能，在计数到溢出时间的 75%时候中断

23.3 功能描述

WWDT 在芯片复位后默认关闭，软件需对控制寄存器（WWDT_CTRL）写入 0x5A 来启动 WWDT。WWDT 启动后，如果软件在窗口开放期内对 WWDT 控制寄存器（WWDT_CTRL）写 0xAC，将清零计数器。WWDT 一旦使能后不能关闭，直到下一次复位，WWDT 复位发生后将会关闭 WWDT。

WWDT 使用 PCLK 工作，内置 4096 倍预分频电路，计数器溢出时间长度可配置为 1/4/16/64/128/256/512/1024/2048/4096/8192/16384/32768/65536 个 4096 倍的 PCLK 周期，溢出时间长度计算公式如下：

$$t_{WWDT} = f_{PCLK} \times 4096 \times N_{CFG}$$

下表为一些计算例子：

表 23-1：WWDT 计数器溢出时间计算

PCLK0 频率（f_{PCLK}）	溢出长度配置（N_{CFG}）	溢出时间（t_{WWDT}）（ms）
42MHz	1	0.097523
	4	0.390095
	16	1.560381
	64	6.241524

PCLK0 频率（f_{PCLK}）	溢出长度配置（N_{CFG}）	溢出时间（t_{WWDT}）（ms）
	128	12.483047
	256	24.966095
	512	49.93219
	1024	99.864381
	2048	199.728762
	4096	399.457524
	8192	798.915048
	16384	1597.830096
	32768	3195.660192
	65536	6391.320384

WWDT 只允许在窗口开放期内进行清除，否则将直接触发复位。使能窗口为计数器的后半周期，软件在清零看门狗之前应注意查询计数值。

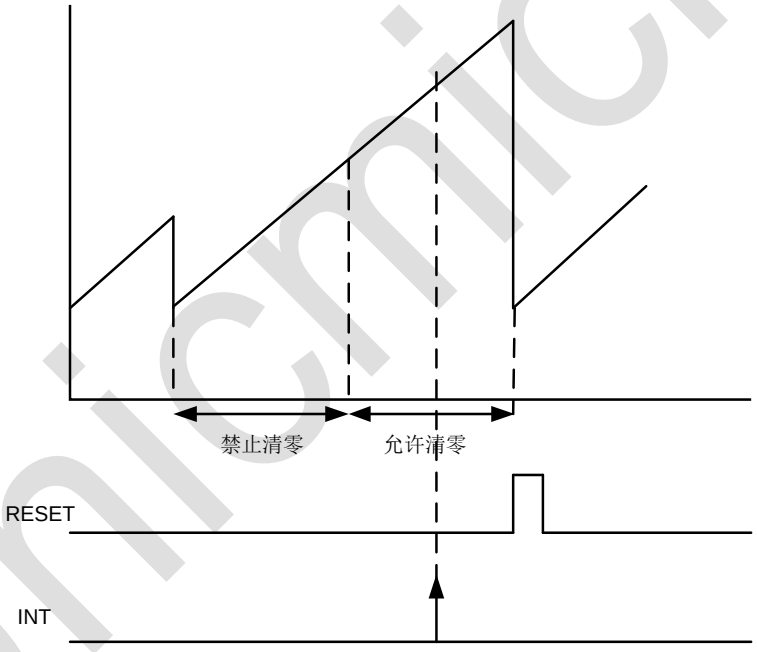


图 23-1：WWDT 计数器刷新时序图

当以下任一情况发生时，WWDT 将产生 CPU 复位信号：

- 计数器溢出
- 对 WWDT 控制寄存器写 0xAC 以外的值（可用于触发 CPU 软件复位）
- 在窗口关闭期内对 WWDT 控制寄存器写 0xAC

当计数器达到溢出时间的 75%时，会触发一个预警中断。

23.4 寄存器描述

WWDT 寄存器基地址：0x40B0_5000

寄存器列表如下：

表 23-2：WWDT 寄存器列表

偏移地址	名称	描述
0x00	WWDT_CTRL	控制寄存器
0x04	WWDT_CFG	配置寄存器
0x08	WWDT_CNT	计数寄存器
0x0C	WWDT_IE	中断使能
0x10	WWDT_IF	中断标志寄存器
0x14	WWDT_DIV_CNT	PCLK 预分频计数寄存器

23.4.1 控制寄存器（WWDT_CTRL）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CTRL	W	0x0	当 CPU 向此地址写入 0x5A 时启动 WWDT。 在启动 WWDT 后，当 CPU 向此地址写入 0xAC 时清零计数器。 在启动 WWDT 后，当 CPU 向此地址写入非 0xAC 的值时复位系统。

23.4.2 配置寄存器（WWDT_CFG）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:4	RSV	-	-	保留

位	名称	属性	复位值	描述
3:0	CFG	R/W	0x0	配置看门狗溢出时间： 0000：t_{PCLK} × 4096 × 1 0001：t_{PCLK} × 4096 × 4 0010：t_{PCLK} × 4096 × 16 0011：t_{PCLK} × 4096 × 64 0100：t_{PCLK} × 4096 × 128 0101：t_{PCLK} × 4096 × 256 0110：t_{PCLK} × 4096 × 512 0111：t_{PCLK} × 4096 × 1024 1000：t_{PCLK} × 4096 × 2048 1001：t_{PCLK} × 4096 × 4096 1010：t_{PCLK} × 4096 × 8192 1011：t_{PCLK} × 4096 × 16384 1100：t_{PCLK} × 4096 × 32768 1101：t_{PCLK} × 4096 × 65536 1110：保留 1111：保留

23.4.3 计数寄存器（WWDT_CNT）

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	CNT	R	0x0	WWDT 计数寄存器值，软件可通过查询此寄存器了解 WWDT 计时进度。

23.4.4 中断使能寄存器（WWDT_IE）

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	IE	R/W	0x0	WWDT 中断使能： 0：中断使能禁止 1：中断使能打开

23.4.5 中断标志寄存器（WWDT_IF）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留

位	名称	属性	复位值	描述
0	IF	R/W1C	0x0	WWDT 75%计时中断标志： 0：无中断产生 1：中断标志置位，写“1”清零

23.4.6 PCLK 预分频计数寄存器（WWDT_DIV_CNT）

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	DIV_CNT	R	0x0	本寄存器是 PCLK 预分频计数器。每经过 4096 个 PCLK 周期，WWDT 计数寄存器值加 1。

23.5 使用流程

23.5.1 中断方式

1. 写 0xA5A55A5A 到 RCM_RCMPR 寄存器，使能 PMU 寄存器写操作。
2. 配置 RCM_APB0CKENR[10]，使能 WWDT 控制器时钟。
3. 配置 RCM_APB0RSTR[10]，释放 WWDT 控制器复位。
4. 写 0xFFFFFFFF 到 RCM_RCMPR 寄存器，结束 PMU 寄存器写操作。
5. 配置 WWDT_CFG[3:0]，设置 WWDT 计数器的溢出时间。
6. 初始化 WWDT 中断，配置 WWDT_IE[0]，使能 WWDT 中断。
7. 写 0x5A 到 WWDT_CTRL 寄存器，启动 WWDT 计数器。
8. WWDT 的窗口开放期为计数器大于或等于溢出时间的 50%，在不同时间段进行如下操作的现象：
   A. 当计数值小于溢出时间的 50%时喂狗，系统会直接复位。
   B. 当计数值大于溢出时间的 50%时喂狗，WWDT 计数器清零并重新计数。
   C. 当计数值达到溢出时间的 75%时，触发 WWDT 预警中断。
   D. 当计数值超出溢出时间时，系统产生复位。
9. 等待计数寄存器（WWDT_CNT）计数到 75%中断触发，若配置寄存器（WWDT_CFG）配置的是 t_{PCLK} × 4096 × 1，则根据 PCLK 预分频计数寄存器（WWDT_DIV_CNT）的计数值计数到 0xC00 触发中断。
10. 在中断标志寄存器（WWDT_IF）写 1 清除中断。

23.5.2 WWDT 清零计数器

写 0xAC 到 WWDT_CTRL 寄存器，清零计数器。

24 实时时钟（RTC）

24.1 概述

备用电池模块 (Battery backup unit, BBU) 为一组功能模块，包括一个带日历的实时时钟(RTC)、1 个闹钟、1 个周期性唤醒源、一个篡改检测器和 80 个字节的备份寄存器。

24.2 特性列表

BBU 的特性列表如下：

- 备份系统支持始终运行
  - 实时时钟 (RTC) 模块
  - 篡改检测器
  - 20 个字（即 80 个字节）的备份寄存器
- 可生成中断的可编程闹钟

24.3 带有系统连接的模块框图

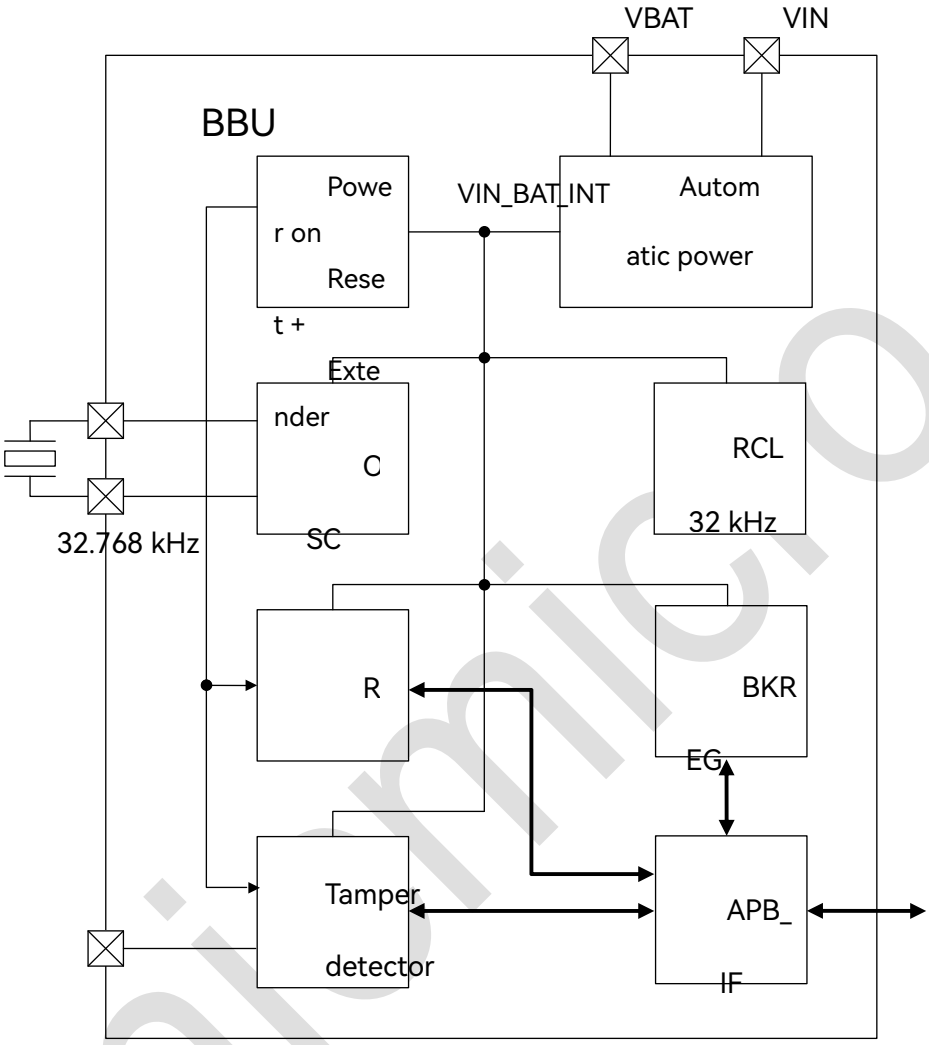


图 24-1：包含模拟模块的 BBU 模块

跨越不同电源域的信号边界需要隔离和电平转换器单元，如图 24-1 所示。

24.4 模块说明

24.4.1 RTC

24.4.1.1 时间和日历计数器

RTC 模块能够提供实时日期和时间信息，可供内核和外围设备访问。

日期和时间计数器：

- 厘秒（百分之一秒）计数器 (CENTISEC)
- 秒计数器 (SECOND)
- 分计数器 (MINUTE)

- 时计数器 (HOUR)
- 周计数器 (WEEK)
- 日计数器 (DAY)
- 月计数器 (MONTH)
- 年计数器 (YEAR and CENTURY)

YEAR 代表年份的后两位数。CENTURY 指示 2000 年代（21 世纪）或 2100 年代（22 世纪）。

每个寄存器均以 BCD 码（二进制编码的十进制，Binary Coded Decimal）进行编码。

24.4.1.2 厘秒计数器 (CENTISEC)

此计数器持续在每一秒钟内从 00 递增到 99。它每经过 326 – 329 个低速时钟周期就递增一次，具体几个周期取决于校准后的秒周期。1 秒内的抖动被控制在仅仅 1 个 HCLK 周期以内。

24.4.1.3 时间计数器 (SECOND, MINUTE and HOUR)

时间计数器的行为如下：

SECOND：当秒计数器从 59 数到 00（进位）时，分计数器会加 1。

MINUTE：当分计数器从 59 数到 00（进位）时，时计数器会加 1。

HOUR：时计数器由 H20_PA（1 位）和 HOUR19（5 位）组成。

- 在 12 小时制下，当[H20_PA，HOUR19]从[1,11]（下午 11 点）递增到[0,12]（上午 12 点）时，日计数器会加 1。
- 在 24 小时制下，当[H20_PA，HOUR19]从[1,3]（23 点）递增到[0,0]（0 点）时，日计数器会加 1。

通过设置 RTC_TIME 寄存器中的 HOUR12_24 位，可以从 12 小时制或 24 小时制中选择小时格式。

- 0：12 小时制
- 1：24 小时制

如下表所示，HOUR 由两个寄存器 HOUR19 和 H20_PA 表示，HOUR19 的值以 BCD 编码。时间格式取决于时钟系统设置。在 12 小时制时钟模式下，H20_PA 表示上午或下午。在 24 小时制时钟模式下，H20_PA 代表 20+点。

表 24-1：HOUR 寄存器编码

时间	24 小时制		12 小时制	
	H20_PA	HOUR19	H20_PA	HOUR19
0 a.m.	0	0	0	12
1 a.m.	0	1	0	1
2 a.m.	0	2	0	2

时间	24 小时制		12 小时制	
	H20_PA	HOUR19	H20_PA	HOUR19
3 a.m.	0	3	0	3
4 a.m.	0	4	0	4
5 a.m.	0	5	0	5
6 a.m.	0	6	0	6
7 a.m.	0	7	0	7
8 a.m.	0	8	0	8
9 a.m.	0	9	0	9
10 a.m.	0	10	0	10
11 a.m.	0	11	0	11
0 p.m.	0	12	1	12
1 p.m.	0	13	1	1
2 p.m.	0	14	1	2
3 p.m.	0	15	1	3
4 p.m.	0	16	1	4
5 p.m.	0	17	1	5
6 p.m.	0	18	1	6
7 p.m.	0	19	1	7
8 p.m.	1	0	1	8
9 p.m.	1	1	1	9
10 p.m.	1	2	1	10
11 p.m.	1	3	1	11

24.4.1.4 日期计数器

周计数器 (WEEK)

计数器值的范围是 0 到 6，输入 7 则当作 0（编码规则：0 或 7 = 周日，1 = 周一，……，5 = 周五，6 = 周六)。WEEK 计数器与实际星期几之间的关系是用户定义的。

日历计数器 (DAY, MONTH, YEAR, CENTURY)

自动日历功能提供如下显示的日历数字。

- DAY：计数器的范围是
  - 01 到 31：在一月、三月、五月、七月、八月、十月、十二月
  - 01 到 30：在四月、六月、九月、十一月
  - 01 到 29：在闰年的二月
  - 01 到 28：在平年的二月

  当日计数器跳转回 01 时，月计数器就会进位。
- MONTH：范围是 01 到 12，跳转回 01 时会进位到年计数器。
- YEAR：范围是 00 到 99，跳转回 00 时会使世纪计数器清零。

- CENTURY：
  - 当世纪计数器为 0 时，YEAR = 04, 08, …, 92, 96 是闰年。
  - 当世纪计数器为 1 时，YEAR = 00, 04, 08, …, 92, 96 是闰年。
  - 在 2000 年到 2399 年的范围内，可以正确识别闰年。

24.4.1.5 时间和日期设置

为了防止在写入时间和日期计数器时意外进位，全部计时器都将冻结直到写入操作结束。

时间/日期设置操作应遵循以下顺序：

1. 将 1 写入 RTC_ACCESS[2]，计数器将停止计时。
2. 尽快完成写入操作。**请注意，任何不存在的时间或日期设置都将被忽略。**
3. 将 1 写入 RTC_ACCESS[3]，从此点开始计时。

24.4.1.6 时间和日期读取

在开始对时间和日期寄存器进行读取操作之前，RTC 需要将时间/日期寄存器的实时值复制到影子读取寄存器中。这种隔离确保了读取时间/日期值的数据的完整性，而时间/日期值的数据不会因时间更新而发生变化。时间/日期读取操作应遵循以下顺序：

1. 将 1 写入 RTC_ACCESS[0]。
2. 尽快完成读取操作。
3. 将 1 写入 RTC_ACCESS[1]。

24.4.1.7 闹钟功能

BBU 中有两个闹钟，闹钟 1 和闹钟 2。

两个闹钟配置寄存器 RTC_ALM1TIME 和 RTC_ALM1DATE，以及一个闹钟使能寄存器 RTC_ALM1EN。

RTC_ALM1EN 寄存器控制闹钟的状态。当 RTC_ALM1EN 寄存器全为 0 时，闹钟 1 被禁用。对于每个位，1 表示仅在相应的寄存器匹配时闹钟开启，而 0 表示无论相应的寄存器值如何闹钟都开启。当时间增加到与 RTC_ALM1DATE 和 RTC_ALM1TIME 寄存器设置的启用时间相匹配时，就会生成闹钟标志和中断。

闹钟 1 设置的示例如下表所示：

表 24-2：闹钟 1 设置的示例

ALM1 date/time registers								ALM1_EN register								闹钟 1 的条件
ALM1_YEAR	ALM1_MONTH	ALM1_DAY	ALM1_WEEK	ALM1_HOUR	ALM1_MINUTE	ALM1_SECOND	ALM1_CENTISEC	ALM1_EN_YEAR	ALM1_EN_MON	ALM1_EN_DAY	ALM1_EN_WEEK	ALM1_EN_HOUR	ALM1_EN_MIN	ALM1_EN_SEC	ALM1_EN_CS	
2	3	4	5	6	7	8	9	0	0	0	0	0	0	0	0	禁用
2	3	4	5	6	7	8	9	0	0	0	0	0	1	1	0	每个小时的 07:08
2	3	4	5	6	7	8	9	1	1	1	0	1	1	1	0	02 年 3 月 4 日 06:07:08
2	3	4	5	6	7	8	9	0	0	1	0	1	1	1	0	每个月 4 日 06:07:08
2	3	4	5	6	7	8	9	0	0	0	1	1	1	1	0	每周五 06:07:08

闹钟 2 产生周期性的中断。根据寄存器设置，中断间隔从每分钟到每 1/128 秒不等。

可以通过寄存器 RTC_INTRAW 和 RTC_INTSTA 检查警报状态。

寄存器 RTC_INTEN 和 RTC_INTCLR 控制上述中断输出到 CPU，中断状态保持其值直到被 APB 访问清除。

24.4.1.8 振荡调准

如下图所示，此模块能够将低速时钟输入频率校准在标称 32.768 kHz 的±5 ppm 之内。调准功能将时钟计数器的最大值从 32768 更改为指定的值。调准步长为±1 个周期，最大调准范围为-128 个周期至+127 个周期。

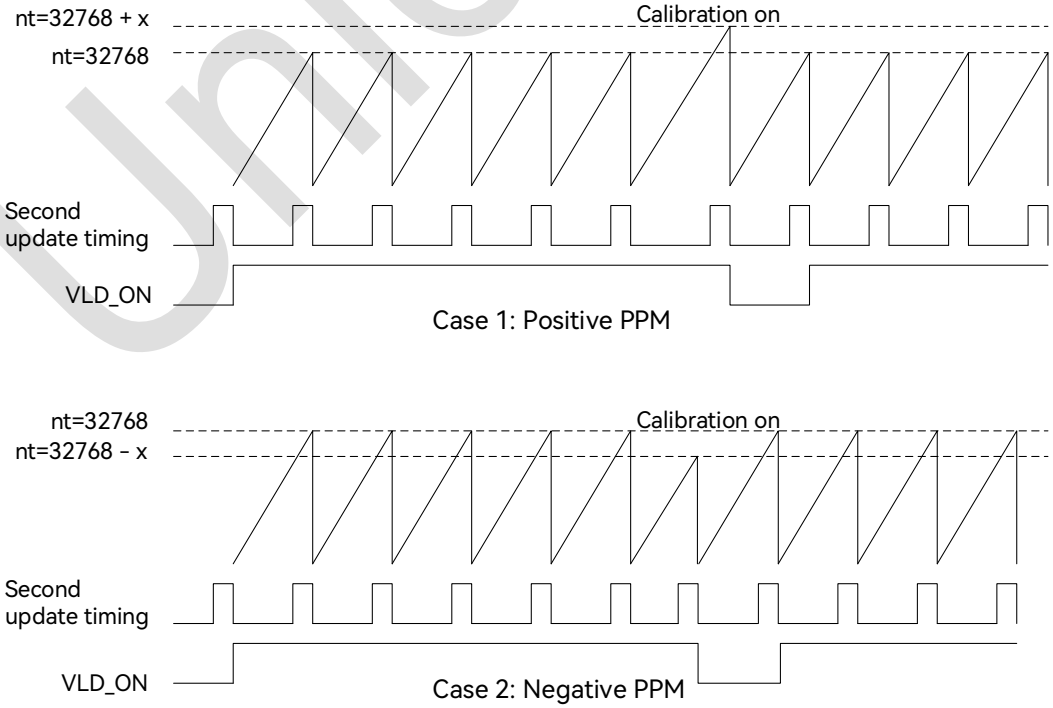


图 24-2：震荡校准

RTC_TRIM 寄存器中的 8 位微调寄存器（TRIM_VALUE[7:0]，或简称为 TRIM[7:0]）存储一个带符号的值，该值可调整低速时钟计数器的最大值。TRIM[7]是其符号，其中“0”表示非负值，而“1”表示负值。当 TRIM[7]为“1”时，TRIM[6:0]被视为二进制补码。

TRIM_MODE 寄存器定义执行此调准的频率。

- 0x0：每 60 秒一次（每当 SECOND = 00 时）
- 0x1：每 30 秒一次（每当 SECOND = 00 或 30 时）
- 0x2：每 15 秒一次（每当 SECOND = 00, 15, 30 或 45 时）
- 0x3：每 6 秒一次（每当 SECOND = 00, 06, 12, 18, 24, 30, 36, 42, 48 或 54 时）

这种补偿方案确保分钟刷新的时间点与振荡器频率精度无关。

振荡调准计算的示例在表 24-3 至表 24-6 中列出。

TRIM_MODE = 0x0

表 24-3：TRIM_MODE = 0x0 的振荡调准计算示例

二进制的 TRIM[7:0]	补偿值	60 秒内的总时钟周期	[ppm]
10000000	-128	32768*59 + 32640*1	-65.1
11000000	-64	32768*59 + 32704*1	-32.2
11111111	-1	32768*59 + 32767*1	-0.5
00000000	0	32768*60	0.0
00000001	1	32768*59 + 32769*1	0.5
00111111	63	32768*59 + 32831*1	32
01111111	127	32768*59 + 32895*1	64.6

TRIM_MODE = 0x1

表 24-4：TRIM_MODE = 0x1 的振荡调准计算示例

二进制的 TRIM[7:0]	补偿值	30 秒内的总时钟周期	[ppm]
10000000	-128	32768*29 + 32640*1	-130.2
11000000	-64	32768*29 + 32704*1	-65.1
11111111	-1	32768*29 + 32767*1	-1
00000000	0	32768*30	0.0
00000001	1	32768*29 + 32769*1	1
00111111	63	32768*29 + 32831*1	64.6
01111111	127	32768*29 + 32895*1	129.2

TRIM_MODE = 0x2

表 24-5：TRIM_MODE = 0x2 的振荡调准计算示例

二进制的 TRIM[7:0]	补偿值	15 秒内的总时钟周期	[ppm]
10000000	-128	32768*14 + 32640*1	-244.2
11000000	-64	32768*14+ 32704*1	-122
11111111	-1	32768*14 + 32767*1	-1.9
00000000	0	32768*15	0.0
00000001	1	32768*14 + 32769*1	1.9

二进制的 TRIM[7:0]	补偿值	15 秒内的总时钟周期	[ppm]
00111111	63	32768*14 + 32831*1	120.2
01111111	127	32768*14 + 32895*1	242.2

TRIM_MODE = 0x3

表 24-6：TRIM_MODE = 0x3 的振荡调准计算示例

二进制的 TRIM[7:0]	补偿值	6 秒内的总时钟周期	[ppm]
10000000	-128	32768*5 + 32640*1	-651
11000000	-64	32768*5 + 32704*1	-325.5
11111111	-1	32768*5 + 32767*1	-5.1
00000000	0	32768*6	0.0
00000001	1	32768*5 + 32769*1	5.1
00111111	63	32768*5 + 32831*1	320.4
01111111	127	32768*5 + 32895*1	646

对于给定的目标 ppm 值，需要正确地计算 TRIM 和 TRIM_MODE，使实际 ppm 尽可能地接近目标。还建议尽可能通过选择较高的 TRIM_MODE 设置，以较小的步长更频繁地执行调整。

输出信号 VLD_ON_O 置 1 用于调准时钟校验。VLD_ON_O 根据 VLD_MODE 设置输出 6, 15, 30 或 60 秒的高电平，这有助于使用外部参考时钟进行定时测量。

24.4.2 篡改检测器

篡改输入引脚可用于监视系统中的特定位置。篡改输入的状态变化可能表明发生了篡改事件。可以将要监视的事件配置为输入（上升或下降或双）边沿检测或（高或低）电平检测。在进行电平检测的情况下，去抖动电路可滤除短于 6 毫秒（时长可配置）的脉冲。

当检测到篡改时，备份系统的行为如下。

- 将时间戳（年月日时分秒）存储进三组寄存器之一。这些寄存器始终保留最近 3 次篡改事件的信息。在正常功耗模式下，可被 ARM 内核访问。
- 篡改事件检测的数量最多为 63。
- 如果使能，则向 CPU 发出中断请求。在备用电池模式下，逻辑电路的电源关闭，因此忽略任何中断请求。

24.4.3 备份寄存器

备份寄存器以 32 位为 1 个字的方式组织，总共有 20 个字的空间。

ARM CPU 可以通过 APB 接口对其访问。

24.5 寄存器描述

RTC 寄存器的基地址：0x40B0_0000

寄存器列表如下：

表 24-7：RTC 寄存器列表

偏移地址	名称	说明
0x00	RTC_TIME	RTC 时间寄存器
0x04	RTC_DATE	RTC 日期寄存器
0x08	RTC_ACCESS	RTC 计数器读写控制寄存器
0x10	RTC_TRIM	RTC 调准控制寄存器
0x14	RTC_TEST	RTC 计数器递增调试功能寄存器
0x20	RTC_ALM1TIME	闹钟 1 时间设置寄存器
0x24	RTC_ALM1DATE	闹钟 1 日期设置寄存器
0x28	RTC_ALM1EN	闹钟 1 使能设置寄存器
0x2C	RTC_ALM2SETTING	闹钟 2 设置寄存器
0x30	RTC_TAMPCTRL	篡改检测器控制和配置寄存器
0x34	RTC_TAMPCNT	篡改次数记录寄存器
0x38	RTC_TAMP3TIME	第 3 新的篡改事件时间戳寄存器
0x3C	RTC_TAMP3DATE	第 3 新的篡改事件日期戳寄存器
0x40	RTC_TAMP2TIME	次新的篡改事件时间戳寄存器
0x44	RTC_TAMP2DATE	次新的篡改事件日期戳寄存器
0x48	RTC_TAMP1TIME	最新的篡改事件时间戳寄存器
0x4C	RTC_TAMP1DATE	最新的篡改事件日期戳寄存器
0x50	RTC_INTEN	RTC 中断请求使能寄存器
0x54	RTC_INTRAW	RTC 中断原始状态寄存器
0x58	RTC_INTSTA	RTC 中断有效状态寄存器
0x5C	RTC_INTCLR	RTC 中断状态清除寄存器
0x60	RTC_BKREG0	备份寄存器，总共 20 个字或 80 个字节
0x64	RTC_BKREG1	
…	…	
0xAC	RTC_BKREG19	

注：由于 RTC 的所有寄存器都在系统低速（32-KHz）时钟域中，因此读或写 RTC 的任何寄存器需要两个低速时钟周期。

24.5.1 RTC 时间寄存器（RTC_TIME）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31	HOUR12_24	R/W	0x0	小时制格式的设置： 0：12 小时制 1：24 小时制
30	RSV	-	-	保留
29	H20_PA	R/W	0x0	用 12 小时制时，H20_PA 表示上午或下午： 0：上午； 1：下午。 用 24 小时制时，H20_PA 表示 20+点。 具体请参考第 24.4.1.3 节的“小时格式”。
28：24	HOUR19	R/W	0x0	HOUR19 的值用 BCD 码（二进制编码的十进制）来编码。小时格式取决于 HOUR12_24。 用 12 小时制时，当[H20_PA, HOUR19]从[1, 0x11]（下午 11 点）数到[0, 0x12]（上午 12 点）时，DAY 计数器将递增。 用 24 小时制时，当[H20_PA, HOUR19]从[1, 3]（23 点）计数到[0, 0]（0 点）时，DAY 计数器将递增。
23	RSV	-	-	保留
22:16	MINUTE	R/W	0x0	分钟计数器的值用 BCD 码来编码，值的范围是 0x00 到 0x59。当它从 0x59 数到 0x00 时，HOUR 计数器将递增。
15	RSV	-	-	保留
14:8	SECOND	R/W	0x0	秒钟计数器的值用 BCD 码来编码，值的范围是 0x00 到 0x59。当它从 0x59 数到 0x00 时，MINUTE 计数器将递增。
7:0	CENTISEC	R	0x0	厘秒计数器（百分之一秒计数器）的值用 BCD 码来编码，值的范围是 0x00 到 0x99。当它从 0x99 数到 0x00 时，SECOND 计数器将递增。

注：

1. 在读写此寄存器之前和之后都需要写入 RTC_ACCESS 寄存器。
2. 写入 RTC_TIME 的无效时间值将被忽略。写入 HOUR、MINUTE 和 SECOND 寄存器的值的有效性分开判断，其中一个寄存器的写入值无效不会影响其他寄存器的有效值的写入。

24.5.2 RTC 日期寄存器（RTC_DATE）

偏移地址：0x04

复位值：0x8001 0106

位	名称	属性	复位值	描述
31	CENTURY	R/W	0x1	世纪计数器。当它变成 0 后，将不再跟随 YEAR 计数器递增： 0：2100 年代至 2300 年代（22 世纪至 24 世纪） 1：2000 年代（21 世纪）
30	RSV	-	-	保留
29:22	YEAR	R/W	0x0	此寄存器代表十进制年份的后两位数。此值用 BCD 码来编码，范围是 0x00 到 0x99。当 YEAR 计数器从 0x99 数到 0x00 时，CENTURY 计数器将变成 0。当 CENTURY = 0 时，04, 08, …, 92, 96 年是闰年，00 年是平年。 当 CENTURY = 1 时，00, 04, 08, …, 92, 96 年是闰年。
21	RSV	-	-	保留
20:16	MONTH	R/W	0x1	月份计数器的值用 BCD 来编码，范围是 0x1 到 0x12。当它从 0x12 数到 0x1 时，YEAR 计数器将递增。
15:14	RSV	-	-	保留
13:8	DAY	R/W	0x1	日计数器的值用 BCD 来编码，范围是： 0x1 至 0x31，在 1, 3, 5, 7, 8, 10, 12 月 0x1 至 0x30，在 4, 6, 9, 11 月 0x1 至 0x29，在闰年 2 月 0x1 至 0x28，在平年 2 月 当 DAY 计数器从月底数到 0x1 时，MONTH 计数器将递增。
7:3	RSV	-	-	保留
2:0	WEEK	R/W	0x6	周计数器，值的范围是 0 到 6。其中，0 表示周日，1 表示周一，以此类推。 WEEK 计数器与日期之间的对应关系由用户定义。

注：

1. 在读写此寄存器之前和之后都需要写入 RTC_ACCESS 寄存器。
2. 写入 RTC_DATE 的无效日期值（日期和月份的组合）将被忽略。

24.5.3 RTC 计数器读写控制寄存器（RTC_ACCESS）

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:4	RSV	-	0x0	保留
3	WRSTP	W	0x0	向此位写入 1 表示 CPU 结束写入 RTC_TIME 和 RTC_DATE 寄存器。 此位写入 1 后，CENTISEC 计数器将继续计数。
2	WRSTA	W	0x0	向此位写入 1 表示 CPU 开始写入 RTC_TIME 和 RTC_DATE 寄存器。 在将此位写入 1 之后，直到对 WRSTP 写入 1 为止，CENTISEC 计数器将保持对其写入的值。
1	RDSTP	W	0x0	向此寄存器写入 1 表示 CPU 结束读取 RTC_TIME 和 RTC_DATE 寄存器。 当向此位写入 1 时，RTC_TIME 和 RTC_DATE 影子寄存器将锁存时间和日期计数器值。
0	RDSTA	W	0x0	向此寄存器写入 1 表示 CPU 开始读取 RTC_TIME 和 RTC_DATE 寄存器。 当向此位写入 1 时，RTC_TIME 和 RTC_DATE 影子寄存器将随着时间和日期计数器不停变化。

24.5.4 RTC 调准控制寄存器（RTC_TRIM）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:14	RSV	-	0x0	保留
13	VLD_EN	R/W	0x0	校验功能使能： 0：禁用校验功能 1：启用校验功能，VLD_ON_O 脉冲周期性置 1
12:11	VLD_MODE	R/W	0x0	VLD_ON_O 的校验时长设置： 0：VLD_ON_O 保持高电平 60 秒 1：VLD_ON_O 保持高电平 30 秒 2：VLD_ON_O 保持高电平 15 秒 3：VLD_ON_O 保持高电平 6 秒
10	TRIM_EN	R/W	0x0	调准功能使能： 0：禁用调准功能 1：启用调准功能
9:8	TRIM_MODE	R/W	0x0	调准频率设置： 0：每 60 秒一次 1：每 30 秒一次 2：每 15 秒一次 3：每 6 秒一次

位	名称	属性	复位值	说明
7:0	TRIM_VALUE	R/W	0x0	带符号的补偿常数，表示用于调准时钟误差的补偿量。

24.5.5 RTC 计数器递增功能测试寄存器（RTC_TEST）

RTC 计数器递增功能，用于测试或调试。

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	说明
31	TEST_EN	R/W	0x0	递增功能使能寄存器： 0：正常操作，禁用递增功能 1：启用递增功能
30:5	RSV	-	0x0	保留
4	MON_CNT_UP	W	0x0	MONTH 计数器递增生效： 0：无动作 1：MONTH 计数器和 SECOND 计数器递增
3	DAY_CNT_UP	W	0x0	DAY 计数器递增生效： 0：无动作 1：DAY 计数器和 SECOND 计数器递增
2	HR_CNT_UP	W	0x0	HOUR 计数器递增生效： 0：无动作 1：HOUR 计数器和 SECOND 计数器递增
1	MIN_CNT_UP	W	0x0	MINUTE 计数器递增生效： 0：无动作 1：MINUTE 计数器和 SECOND 计数器递增
0	SEC_CNT_UP	W	0x0	SECOND 计数器递增生效： 0：无动作 1：SECOND 计数器递增

注：

1. 在设置其他任何递增功能位之前，必须先将 RTC_TEST[31]位置 1。当 RTC_TEST[31]位从 0 切换为 1 时，任何其他递增功能位的同时写入操作都将无效。
2. 每次启用递增功能时，SECOND 计数器都会递增计数。如果在一次写入操作中设置了多个递增计数位，则 SECOND 计数器仍将递增 1。
3. 如果时间或日期计数器递增计数产生进位，将导致更高级别的计数器一连串递增计数。
4. 如果 DAY 计数器递增，则 WEEK 计数器也会递增；但是，如果 MONTH 计数器递增而 DAY 计数器不递增，则 WEEK 计数器将不改变。

24.5.6 闹钟 1 时间设置寄存器（RTC_ALM1TIME）

偏移地址：0x20

复位值：0x0000 0000

位	名称	属性	复位值	说明
31	ALM1_HOUR12_24	R/W	0x0	闹钟 1 的小时格式设置： 0：12 小时制 1：24 小时制
30	RSV	-	0x0	保留
29	ALM1_H20_PA	R/W	0x0	闹钟 1 的小时设置。 小时格式遵循 ALM1_HOUR12_24 的设置。
28:24	ALM1_HOUR19	R/W	0x0	
23	RSV	-	0x0	保留
22:16	ALM1_MINUTE	R/W	0x0	闹钟 1 的分钟设置。
15	RSV	-	0x0	保留
14:8	ALM1_SECOND	R/W	0x0	闹钟 1 的秒钟设置。
7:0	ALM1_CENTISEC	R/W	0x0	闹钟 1 的厘秒设置。

24.5.7 闹钟 1 日期设置寄存器（RTC_ALM1DATE）

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:30	RSV	-	0x0	保留
29:22	ALM1_YEAR	R/W	0x0	闹钟 1 年份设置。
21	RSV	-	0x0	保留
20:16	ALM1_MONTH	R/W	0x0	闹钟 1 月份设置。
15:14	RSV	-	0x0	保留
13:8	ALM1_DAY	R/W	0x0	闹钟 1 日期设置。
7:3	RSV	-	0x0	保留
2:0	ALM1_WEEK	R/W	0x0	闹钟 1 星期设置。

24.5.8 闹钟 1 设置使能寄存器（RTC_ALM1EN）

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	说明
31	ALM1_EN	R/W	0x0	闹钟 1 使能位： 0：关闭闹钟 1 1：启用闹钟 1
30:8	RSV	-	0x0	保留
7	ALM1_EN_YEAR	R/W	0x0	闹钟 1 的年份条件控制位： 0：条件不应用 1：条件应用

位	名称	属性	复位值	说明
6	ALM1_EN_MON	R/W	0x0	闹钟 1 的月份条件控制位： 0：条件不应用 1：条件应用
5	ALM1_EN_DAY	R/W	0x0	闹钟 1 的日期条件控制位： 0：条件不应用 1：条件应用
4	ALM1_EN_WEEK	R/W	0x0	闹钟 1 的星期条件控制位： 0：条件不应用 1：条件应用
3	ALM1_EN_HOUR	R/W	0x0	闹钟 1 的小时条件控制位： 0：条件不应用 1：条件应用
2	ALM1_EN_MIN	R/W	0x0	闹钟 1 的分钟条件控制位： 0：条件不应用 1：条件应用
1	ALM1_EN_SEC	R/W	0x0	闹钟 1 的秒钟条件控制位： 0：条件不应用 1：条件应用
0	ALM1_EN_CS	R/W	0x0	闹钟 1 的厘秒条件控制位： 0：条件不应用 1：条件应用

24.5.9 闹钟 2 设置寄存器（RTC_ALM2SETTING）

偏移地址：0x2C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31	ALM2_EN	R/W	0x0	闹钟 2 使能位： 0：关闭闹钟 2 1：启用闹钟 2
30:4	RSV	-	-	保留
3:0	ALM2_SETTING	R/W	0x0	闹钟 2 中断输出周期： 0：无输出 1：1 秒 2：1/2 秒 3：1/4 秒 4：1/8 秒 5：1/16 秒 6：1/32 秒 7：1/64 秒 8：1/128 秒 9：1 分钟 其他：1 秒

24.5.10 篡改检测器控制和配置寄存器（RTC_TAMPCTRL）

偏移地址：0x30

复位值： 0x0000 0030

位	名称	属性	复位值	说明
31:7	RSV	-	-	保留
6	TAMP_CNT_CLR	W	0x0	篡改计数器清除位。 向此位写 1 以清除篡改计数器 TAMP_CNT。
5:4	TAMP_DBNC	R/W	0x3	TAMPER_IN 的去抖动时间： 0：不去抖动 1：2 毫秒 2：4 毫秒 3：6 毫秒
3:1	TAMP_EDGE	R/W	0x0	要监视的 TAMPER_IN 引脚事件： 0：上升沿 1：下降沿 2：双边沿 3：时长超过 TAMP_DBNC 设定值的高电平 4：时长超过 TAMP_DBNC 设定值的低电平 其他：禁用篡改检测器
0	TAMP_EN	R/W	0x0	TAMPER_IN 引脚的篡改检测器使能位： 0：禁用篡改检测器 1：启用篡改检测器

24.5.11 篡改次数记录寄存器（RTC_TAMPCNT）

偏移地址：0x34

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:6	RSV	-	0x0	保留
5:0	TAMP_CNT	R	0x0	篡改事件计数器： 0x00：没有篡改事件 0x01~0x3E：篡改事件次数 0x3F：63 次或更多篡改事件

24.5.12 第 3 新的篡改事件时间戳寄存器（RTC_TAMP3TIME）

偏移地址：0x38

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:30	RSV	-	0x0	保留
29	TAMP3_H20_PA	R	0x0	第 3 新篡改事件的小时。 小时格式遵循 HOUR12_24 的设置。
28:24	TAMP3_HOUR19	R	0x0	
23	RSV	-	0x0	保留
22:16	TAMP3_MINUTE	R	0x0	第 3 新篡改事件的分钟。
15	RSV	-	0x0	保留
14:8	TAMP3_SECOND	R	0x0	第 3 新篡改事件的秒钟。
7:0	RSV	-	0x0	保留

24.5.13 第 3 新的篡改事件日期戳寄存器（RTC_TAMP3DATE）

偏移地址：0x3C

复位值： 0x0000 0000

位	名称	属性	复位值	说明
31:30	RSV	-	0x0	保留
29:22	TAMP3_YEAR	R	0x0	第 3 新篡改事件的年份。
21	RSV	-	0x0	保留
20:16	TAMP3_MONTH	R	0x0	第 3 新篡改事件的月份。
15:14	RSV	-	0x0	保留
13:8	TAMP3_DAY	R	0x0	第 3 新篡改事件的日期。
7:0	RSV	-	0x0	保留

24.5.14 次新的篡改事件时间戳寄存器（RTC_TAMP2TIME）

偏移地址：0x40

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:30	RSV	-	0x0	保留
29	TAMP2_H20_PA	R	0x0	次新篡改事件的小时。 小时格式遵循 HOUR12_24 的设置。
28:24	TAMP2_HOUR19	R	0x0	
23	RSV	-	0x0	保留
22:16	TAMP2_MINUTE	R	0x0	次新篡改事件的分钟。
15	RSV	-	0x0	保留
14:8	TAMP2_SECOND	R	0x0	次新篡改事件的秒钟。
7:0	RSV	-	0x0	保留

24.5.15 次新的篡改事件日期戳寄存器（RTC_TAMP2DATE）

偏移地址：0x44

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:30	--	I	0x0	保留
29:22	TAMP2_YEAR	R	0x0	次新篡改事件的年份。
21	--	I	0x0	保留
20:16	TAMP2_MONTH	R	0x0	次新篡改事件的月份。
15:14	--	I	0x0	保留
13:8	TAMP2_DAY	R	0x0	次新篡改事件的日期。
7:0	--	I	0x0	保留

24.5.16 最新的篡改事件时间戳寄存器（RTC_TAMP1TIME）

偏移地址：0x48

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:30	RSV	-	0x0	保留
29	TAMP1_H20_PA	R	0x0	最新篡改事件的小时。小时格式遵循 HOUR12_24 的设置。
28:24	TAMP1_HOUR19	R	0x0	
23	RSV	-	0x0	保留
22:16	TAMP1_MINUTE	R	0x0	最新篡改事件的分钟。
15	RSV	-	0x0	保留
14:8	TAMP1_SECOND	R	0x0	最新篡改事件的秒钟。
7:0	RSV	-	0x0	保留

24.5.17 最新的篡改事件日期戳寄存器（RTC_TAMP1DATE）

偏移地址：0x4C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:30	RSV	-	0x0	保留
29:22	TAMP1_YEAR	R	0x0	最新篡改事件的年份。
21	RSV	-	0x0	保留
20:16	TAMP1_MONTH	R	0x0	最新篡改事件的月份。
15:14	RSV	-	0x0	保留

位	名称	属性	复位值	说明
13:8	TAMP1_DAY	R	0x0	最新篡改事件的日期。
7:0	RSV	-	0x0	保留

24.5.18 RTC 中断请求使能寄存器（RTC_INTEN）

偏移地址：0x50

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:3	RSV	-	0x0	保留
2	TAMP_INT_EN	R/W	0x0	篡改中断请求使能位： 0：禁止篡改中断请求 1：允许篡改中断请求
1	ALM2_INT_EN	R/W	0x0	闹钟 2 中断请求使能位： 0：禁止闹钟 2 中断请求 1：允许闹钟 2 中断请求
0	ALM1_INT_EN	R/W	0x0	闹钟 1 中断请求使能位： 0：禁止闹钟 1 中断请求 1：允许闹钟 1 中断请求

24.5.19 RTC 原始中断状态寄存器（RTC_INTRAW）

偏移地址：0x54

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:3	RSV	-	0x0	保留
2	TAMP_INT_RAW	R	0x0	篡改原始中断状态： 0：篡改原始中断未触发 1：篡改原始中断已触发
1	ALM2_INT_RAW	R	0x0	闹钟 2 原始中断状态： 0：闹钟 2 原始中断未触发 1：闹钟 2 原始中断已触发
0	ALM1_INT_RAW	R	0x0	闹钟 1 原始中断状态： 0：闹钟 1 原始中断未触发 1：闹钟 1 原始中断已触发

24.5.20 RTC 使能后中断状态寄存器（RTC_INTSTA）

偏移地址：0x58

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:3	RSV	-	0x0	保留
2	TAMP_INT_STA	R	0x0	篡改使能后中断状态： 0：篡改使能后中断状态未触发 1：篡改使能后中断状态已触发
1	ALM2_INT_STA	R	0x0	闹钟 2 使能后中断状态： 0：闹钟 2 使能后中断状态未触发 1：闹钟 2 使能后中断状态已触发
0	ALM1_INT_STA	R	0x0	闹钟 1 使能后中断状态。 0：闹钟 1 使能后中断状态未触发 1：闹钟 1 使能后中断状态已触发

24.5.21 RTC 中断状态清除寄存器（RTC_INTCLR）

偏移地址：0x5C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:3	RSV	-	0x0	保留
2	TAMP_INT_CLR	W	0x0	篡改中断状态清除位： 0：无动作 1：清除篡改中断状态
1	ALM2_INT_CLR	W	0x0	闹钟 2 中断状态清除位： 0：无动作 1：清除闹钟 2 中断状态
0	ALM1_INT_CLR	W	0x0	闹钟 1 中断状态清除位： 0：无动作 1：清除闹钟 1 中断状态

24.5.22 备份寄存器（RTC_BKREG_n）

备份寄存器，总共 20 个字（即 80 个字节）。

偏移地址：0x60 + 0x4 × n，其中 n = 0, 1, …, 19

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	BKREG_n	R/W	0x0	备用寄存器 n (n = 0, 1, …, 19)

24.6 使用流程

24.6.1 使能 RTC 时钟

1. 写 0xABCD 到 PMU_CPR 寄存器，使能 PMU 寄存器写操作。

2. 配置 PMU_FCCR[6]，使能 RTC 控制器时钟。
3. 配置 PMU_FRCR[6]，释放 RTC 控制器复位。
4. 配置 PMU_SASR[0]，将外部 VBAT 供电置位（即选择有外部 VBAT 供电）。
5. 写 0x459E 到 PMU_CPR 寄存器，结束 PMU 寄存器写操作。

24.6.2 RTC 时间读取

1. 使能 RTC 时钟。
2. 开始初始化 RTC 日期时间，写 0x4 到 RTC_ACCESS 寄存器，开启 RTC 写使能。
3. 写日期、时间到 RTC_DATA、RTC_TIME 寄存器。
4. 结束初始化 RTC 日期时间，写 0x8 到 RTC_ACCESS 寄存器，结束 RTC 写入。
5. 写 0x1 到 RTC_ACCESS 寄存器，更新日期和时间。
6. 写 0x2 到 RTC_ACCESS 寄存器，保存当前时刻日期和时间。
7. 读 RTC_DATA、RTC_TIME 寄存器，获取日期和时间。

24.6.3 RTC 设置闹钟 1

1. 使能 RTC 时钟。
2. 开始初始化 RTC 日期时间，写 0x4 到 RTC_ACCESS 寄存器，开启 RTC 写使能。
3. 写日期、时间到 RTC_DATA、RTC_TIME 寄存器。
4. 结束初始化 RTC 日期时间，写 0x8 到 RTC_ACCESS 寄存器，结束 RTC 写入。
5. 设置闹钟 1 定时时间，写日期、时间到 RTC_ALM1DATE、RTC_ALM1TIME 寄存器。
6. 初始化闹钟 1 中断，配置 RTC_INTEN[0]，启用闹钟 1 中断。
7. 使能 RTC_ALM1EN 寄存器中的相应条件应用位，配置 RTC_ALM1EN[31]，使能闹钟 1。

24.6.4 RTC 设置闹钟 2

1. 使能 RTC 时钟。
2. 开始初始化 RTC 日期时间，写 0x4 到 RTC_ACCESS 寄存器，开启 RTC 写使能。
3. 写日期、时间到 RTC_DATA、RTC_TIME 寄存器。
4. 结束初始化 RTC 日期时间，写 0x8 到 RTC_ACCESS 寄存器，结束 RTC 写入。
5. 操作 RTC_ALM2SETTING[3:0]，设置闹钟 2 的中断输出周期。
6. 初始化闹钟 2 中断，配置 RTC_INTEN[1]，启用闹钟 2 中断。
7. 使能 RTC_ALM2SETTING[31]，使能闹钟 2。

24.6.5 RTC 备用寄存器

1. 写 0xABCD 到 PMU_CPR 寄存器，使能 PMU 寄存器写操作。
2. 配置 PMU_SASR[0]，将外部 VBAT 供电置位（即选择有外部 VBAT 供电）。
3. 写 0x459E 到 PMU_CPR 寄存器，结束 PMU 寄存器写操作。
4. 向 RTC_BKREGn 寄存器写入数据（一个备用寄存器储存 4 个字节，共 80 字节）。
5. VBAT 外接电源，VDDH 断电后重新上电再次读取备用寄存器，与断电之前保持一致。

24.6.6 RTC 篡改检测

1. 配置 GPIOC13 引脚为输入。
2. 写 0xABCD 到 PMU_CPR 寄存器，使能 PMU 寄存器写操作。
3. 配置 PMU_PDWKCR[3]，将 RTC_TAMPER 唤醒事件置为有效。
4. 写 0x459E 到 PMU_CPR 寄存器，结束 PMU 寄存器写操作。
5. 使能 RTC 时钟。
6. 开始初始化 RTC 日期时间，写 0x4 到 RTC_ACCESS 寄存器，开启 RTC 写使能。
7. 写日期、时间到 RTC_DATA、RTC_TIME 寄存器。
8. 结束初始化 RTC 日期时间，写 0x8 到 RTC_ACCESS 寄存器，结束 RTC 写入。
9. 配置 RTC_TAMPCTRL[3:1]，设置要监测 TAMPER_IN 引脚的电平事件。
10. 将 RTC_TAMPCTRL[6]置 1，清空篡改计数器的值。
11. 将 RTC_INTEN[2]置 1，使能篡改中断。
12. 将 RTC_TAMPCTRL[0]置 1，使能篡改检测。

24.6.7 RTC 清空篡改计数器值

将 RTC_TAMP_CTRL[6]置 1，清空篡改计数器的值。

24.6.8 RTC 调准

1. 使能 RTC 时钟。
2. 开始初始化 RTC 日期时间，写 0x4 到 RTC_ACCESS 寄存器，开启 RTC 写使能。
3. 写日期、时间到 RTC_DATA、RTC_TIME 寄存器。
4. 结束初始化 RTC 日期时间，写 0x8 到 RTC_ACCESS 寄存器，结束 RTC 写入。
5. 配置 RTC_TRIM[9:8]和 RTC_TRIM[7:0]，设置调准频率和补偿常数，将 RTC_TRIM[10]置 1 使能校准功能。
6. 配置 PC5 为复用功能 0，可以输出校验电平。

24.6.9 RTC 递增测试

1. 使能 RTC 时钟。
2. 清空 RTC_TEST 寄存器，配置 RTC_TEST 使相应的计数器递增。
3. 写 0x1 到 RTC_ACCESS 寄存器，更新日期和时间。
4. 写 0x2 到 RTC_ACCESS 寄存器，保存当前时刻日期和时间。
5. 读 RTC_DATA、RTC_TIME 寄存器，获取日期和时间。

   重复操作 2~5，即可观察到相应计数器递增。

25 CAN 总线控制器（CAN）

25.1 概述

CAN（Controller Area Network）控制器可以用于汽车电子和工业控制领域，支持 CAN2.0A/B 协议。

25.2 主要特性

- 符合 CAN2.0A, 2.0B 协议
- 支持 CAN 格式
- 支持最大 8 bytes 数据帧
- CAN2.0B 最大数据传输速率 1Mbps
- 64-byte RX FIFO
- 16-byte TX FIFO
- 支持传输停止
- 错误计数器可读

25.3 管脚说明

表 25-1：CAN 管脚说明

功能管脚	复用管脚	方向	功能描述
CAN0_TX	PA12，PB9，PD1	Output	发送数据
CAN0_RX	PA11，PB8，PD0	Input	接收数据
CAN1_TX	PB6，PB13	Output	发送数据
CAN1_RX	PB5，PB12	Input	接收数据

25.4 功能描述

25.4.1 标准帧内存缓冲区布局

要发送/接收的消息存储在发送/接收缓冲区中，发送缓冲区一次可以保存一条消息，接收 FIFO 缓冲区一次可以保存许多消息。存储器的地址不能直接使用，只能通过 TXBUFn 和 RXBUFn 寄存器间接获取。标准帧的发送/接收缓冲区布局如下：

表 25-2：标准帧发送/接收缓冲区布局图

offset	data bits	Standard Frame buffer content for TX/RX RAM							
0x000	7:0	FF	RTR	X/0	X/0	DLC3	DLC2	DLC1	DLC0
	15:8	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
	23:16	ID2	ID1	ID0	X/RTR	X/0	X/0	X/0	X/0
	31:24	data 1 (DLC<1 then free)							
0x001	7:0	data 2 (DLC<2 then free)							
	15:8	data 3 (DLC<3 then free)							
	23:16	data 4 (DLC<4 then free)							
	31:24	data 5 (DLC<5 then free)							
0x002	7:0	data 6 (DLC<6 then free)							
	15:8	data 7 (DLC<7 then free)							
	23:16	data 8 (DLC<8 then free)							
	31:24	free							

注：

FF（帧格式）：1 扩展帧，0 标准帧

RTR（远程发送请求位）：1 远程帧，0 数据帧

X：忽略

DLC：数据长度代码

ID：CAN 识别符

Data（1…8）：数据字节（7-MSB，0-LSB）

25.4.2 扩展帧内存缓冲区布局

表 25-3：扩展帧内存缓冲区布局

offset	data bits	Extended Frame buffer content for TX/RX RAM							
0x000	7:0	FF	RTR	X/0	X/0	DLC3	DLC2	DLC1	DLC0
	15:8	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
	23:16	ID20	ID19	ID18	ID17	ID16	ID15	ID14	ID13
	31:24	ID12	ID11	ID10	ID9	ID8	ID7	ID6	ID5
0x001	7:0	ID4	ID3	ID2	ID1	ID0	X/RTR	X/0	X/0
	15:8	data 1 (DLC<1 then free)							
	23:16	data 2 (DLC<2 then free)							
	31:24	data 3 (DLC<3 then free)							
0x002	7:0	data 4 (DLC<4 then free)							
	15:8	data 5 (DLC<5 then free)							
	23:16	data 6 (DLC<6 then free)							
	31:24	data 7 (DLC<7 then free)							
0x003	7:0	data 8 (DLC<8 then free)							
	15:8	free							
	23:16	free							
	31:24	free							

注：

FF（帧格式）：1 扩展帧，0 标准帧

RTR（远程发送请求位）：1 远程帧，0 数据帧

X：忽略

DLC：数据长度代码

ID：CAN 识别符

Data（1…8）：数据字节（7-MSB，0-LSB）

25.5 寄存器列表

CAN0 寄存器基地址：0x40E0_0000

CAN1 寄存器基地址：0x40E0_1000

寄存器列表如下：

表 25-4：CAN 寄存器列表

偏移地址	名称	描述	
0x00	CONFIG0	0x00	CAN_MR：模式寄存器
		0x01	CAN_CMR：命令寄存器
		0x02	CAN_SR：控制寄存器
		0x03	CAN_ISR：中断状态寄存器
0x04	CONFIG1	0x04	CAN_IMR：中断屏蔽寄存器
		0x05	CAN_RMC：接收数据计数寄存器
		0x06	CAN_BTR0：总线时序寄存器 0
		0x07	CAN_BTR1：总线时序寄存器 1
0x08	CAN_TXBUF	发送缓存寄存器	
0x0C	CAN_RXBUF	接收缓存寄存器	
0x10	CAN_ACR	接收过滤匹配寄存器	
0x14	CAN_AMR	接收过滤屏蔽寄存器	
0x18	ERRCR	0x18	CAN_ECC：错误码捕获寄存器
		0x19	CAN_RXERR：接收错误计数寄存器
		0x1A	CAN_TXERR：发送错误计数寄存器
		0x1B	CAN_ALC：仲裁丢失捕获寄存器

25.5.1 模式寄存器（CAN_MR（CONFIG0[7:0]））

偏移地址：0x00

复位值：0x04

位	名称	属性	复位值	描述
7	RXF_CLR	W	0	RX_FIFO指针清除位： 1：复位RX_FIFO的读写指针，复位后此位自动恢复为0 0：无变化
6:3	RSV	-	-	保留
2	RM	R/W	1	复位模式设置位： 1：CAN工作在复位模式 0：CAN工作在其他模式 在复位模式中不进行数据的发送和接收，此模式用于进行一些硬件的配置（某些寄存器只能在复位模式下进行写操作），在复位模式以后，可进入监听模式或者正常模式。

位	名称	属性	复位值	描述
1	LOM	R/W	0	监听模式设置位： 1：若RM=0，CAN进入监听模式* 0：若RM=0，CAN进入正常模式 此位只能在复位模式设置
0	AFM	R/W	0	硬件匹配数据选择位： 1：使用单过滤器 0：使用双过滤器 此位只能在复位模式设置

注*：在监听模式下，即使成功接收到消息，CAN 控制器也不会对 CAN 总线进行应答（不会发送 ACK 响应）。错误计数器将停止在当前值。监听模式主要用于比特率检测，不会干扰网络流量，监听模式还可用于 CAN 总线分析仪。

25.5.2 命令寄存器（CAN_CMR（CONFIG0[15:8]））

偏移地址：0x01

复位值：0x00

位	名称	属性	复位值	描述
7:3	RSV	-	-	保留
2	TR	W	0	发送请求设置位： 1：启动发送，进行帧传输 0：禁止发送
1	AT	W	0	中止传输允许位： 1：允许中止传输 0：禁止中止传输 同时设置TR和AT可启动单发传输，在总线错误或者仲裁丢失的情况下，不会执行帧的重新传输。中止只对即将要传输的帧作用，已经发出的帧无法中止。如果在上一个命令中将TR设为1来启动传输，则无法通过将TR位设为0来取消，此时可通过设置AT为1来取消传输。
0	RSV	-	-	保留

25.5.3 状态寄存器（CAN_SR（CONFIG0[23:16]））

偏移地址：0x02

复位值：0x20

位	名称	属性	复位值	描述
7	RBS	R	0	接收FIFO状态： 1：FIFO中至少有一条消息 0：FIFO中没有消息

位	名称	属性	复位值	描述
6	DSO	R	0	数据溢出状态： 1：RX FIFO溢出，RX溢出中断触发（如使能） 0：自上次清除数据溢出以来未发生溢出
5	TBS	R	1	发送BUFFER状态： 1：发送BUFFER可被CPU写入 0：发送BUFFER已锁定。正在发送消息或正在等待发送。如果CPU在锁定状态下（TBS = 0）尝试写入发送缓冲区，则不接受已写入的数据
4	RSV	-	-	保留
3	RS	R	0	接收状态位： 1：CAN正在接收 0：CAN未处于接收状态
2	TS	R	0	发送状态位： 1：CAN正在传输 0：CAN未处于传输状态
1	ES	R	0	错误状态位： 1：至少一个CAN错误计数器达到错误警告限制（96） 0：正常状态
0	BS	R	0	总线状态位 1：离线状态。CAN控制器处于复位模式，错误警告中断触发（如使能）。发送错误计数器设为127，接收错误计数器设为0。CAN将一直处于复位模式，直到CPU将RM位清掉。完成此操作后，CAN将等待128次总线空闲信号的出现（11个连续的隐性位），发送错误计数器向下计数。然后BS位清0，错误计数器复位，错误警告中断触发（如使能） 0：正常状态。可进行帧传输和接收

25.5.4 中断状态寄存器（CAN_ISR（CONFIG0[31:24]））

偏移地址：0x03

复位值：0x00

位	名称	属性	复位值	描述
7	RSV	-	-	保留
6	ALI	R/W	0	仲裁丢失中断状态位： 当CAN在消息传输过程中丢失仲裁并成为接收端时，此位置位，可以读取ALC寄存器以检查丢失了仲裁段中的哪一位，写1清除此中断
5	EWI	R/W	0	错误警告中断状态位： 当SR寄存器的ES或BS位改变时，错误警告中断置位。因此，它可用于检测CAN是否进入或退出总线关闭状态。写1清除中断

位	名称	属性	复位值	描述
4	EPI	R/W	0	错误被动中断状态位： 当CAN总线控制器达到或退出错误被动级别（即在状态更改为主动到被动或被动到主动）时，此位置位。写1清除中断
3	RI	R/W	0	接收中断状态位： 当接收FIFO中至少有一条CAN帧数据时，CAN将此位置1。读取消息后，CPU必须将RI位写1（消息读取确认），以减少RX消息计数器（RMC）计数，RMC不会自动递减
2	TI	R/W	0	发送中断状态位： 成功发送后，发送中断位被置位。在写入新的数据帧之前可通过清除TI位（写1清除）将写指针复位到TX RAM
1	BEI	R/W	0	总线错误中断状态位： 当CAN在发送或接收消息时遇到总线错误时，将BEI置位。写1清除中断
0	DOI	R/W	0	接收数据溢出中断状态位： 发生接收FIFO溢出时，DOI置位。写1清除中断

25.5.5 中断屏蔽寄存器（CAN_IMR（CONFIG1[7:0]））

偏移地址：0x04

复位值：0x00

位	名称	属性	复位值	描述
7	RSV	-	-	保留
6	ALIM	R/W	0	仲裁丢失中断使能位。使能CAN发送器在发送期间丢失仲裁并成为CAN接收器时触发中断： 1：使能ALI中断 0：禁止ALI中断
5	EWIM	R/W	0	错误警告中断使能位。使能当CAN_SR寄存器的BS或ES位状态改变时触发中断： 1：使能EWI中断 0：禁止EWI中断
4	EPIM	R/W	0	错误被动中断使能位。使能当CAN控制器进入或离开被动错误模式时触发中断： 1：使能EPI中断 0：禁止EPI中断
3	RIM	R/W	0	接收中断使能位： 1：使能RI中断 0：禁止RI中断
2	TIM	R/W	0	发送中断使能位： 1：使能TI中断 0：禁止TI中断

位	名称	属性	复位值	描述
1	BEIM	R/W	0	总线错误中断使能位。使能当CAN在发送或接收过程中发生总线错误时触发中断： 1：使能BEI中断 0：禁止BEI中断
0	DOIM	R/W	0	接收数据溢出中断使能位： 1：使能DOI中断 0：禁止DOI中断

25.5.6 接收数据计数寄存器（CAN_RMC（CONFIG1[15:8]））

偏移地址：0x05

复位值：0x00

位	名称	属性	复位值	描述
7:5	RSV	-	-	保留
4:0	RMC	R	0	接收FIFO中CAN帧个数。 接收FIFO最多可以存储32条标准ID消息（或者16条拓展ID消息）。以下等式允许计算要存储的最大消息数-RX FIFO: $\mathbf{n} = \frac{\mathbf{64}}{\mathbf{3} + \boldsymbol{data\_length\_code}}$ 注：此处data_length_code至少为1，若CAN数据段长度为0，data_length_code=1。

25.5.7 总线时序寄存器（CAN_BTR0（CONFIG1[23:16]））

此寄存器只能在复位模式写入，可在任何模式读取。

偏移地址：0x06

复位值：0x00

位	名称	属性	复位值	描述
7:6	SJW	R/W	0	同步跳跃宽度： $t_{SJW}=t_{SCLK}$ x (2 x SJW.1 + SJW.0 + 1) 为了补偿不同CAN总线控制器的时钟振荡器之间的相移，必须相应地缩短或延长位周期。 SJW定义了一个重新同步可以改变一个位周期的最大时钟周期数。再同步过程中，硬件会通过在PBS1段内增加1+SJW个t_{SCLK}，或者在PBS2段内减少1~（1+SJW）个t_{SCLK}来与接收信号达到同步
5:0	BRP	R/W	0	波特率预分频值: t_{SCLK}=2 x t_{CLK} x (32 x BRP.5 + 16 x BRP.4 + 8 x BRP.3 + 4 x BRP.2 +2 x BRP.1 + BRP.0 +1) 其中，$t_{CLK} = 1/f_{PCLK}$

25.5.8 总线时序寄存器（CAN_BTR1（CONFIG1[31:24]））

此寄存器只能在复位模式写入，可在任何模式读取。

偏移地址：0x07

复位值：0x00

位	名称	属性	复位值	描述
7	SAM	R/W	0	总线电平采样数选择位： 1：采样三次总线电平（适用于中/低速总线） 0：采样一次总线电平（适用于高速总线）
6:4	TSEG2	R/W	0	Time Segment 2的时钟周期数 t_{TSEG2}= t_{SCLK} x （4 x TSEG2.2 +2 x TSEG2.1 + TSEG2.0 +1）
3:0	TSEG1	R/W	0	Time Segment 1的时钟周期数 t_{TSEG1}= t_{SCLK} x （8 x TSEG1.3 + 4 x TSEG1.2 +2 x TSEG1.1 + TSEG1.0 +1）

CAN 的位周期结构如下图。其中同步段（SYNC SEG）为 1 x t_{SCLK}，相位缓冲段 1 和 2 长度由 TSEG1 和 TSEG2 决定。

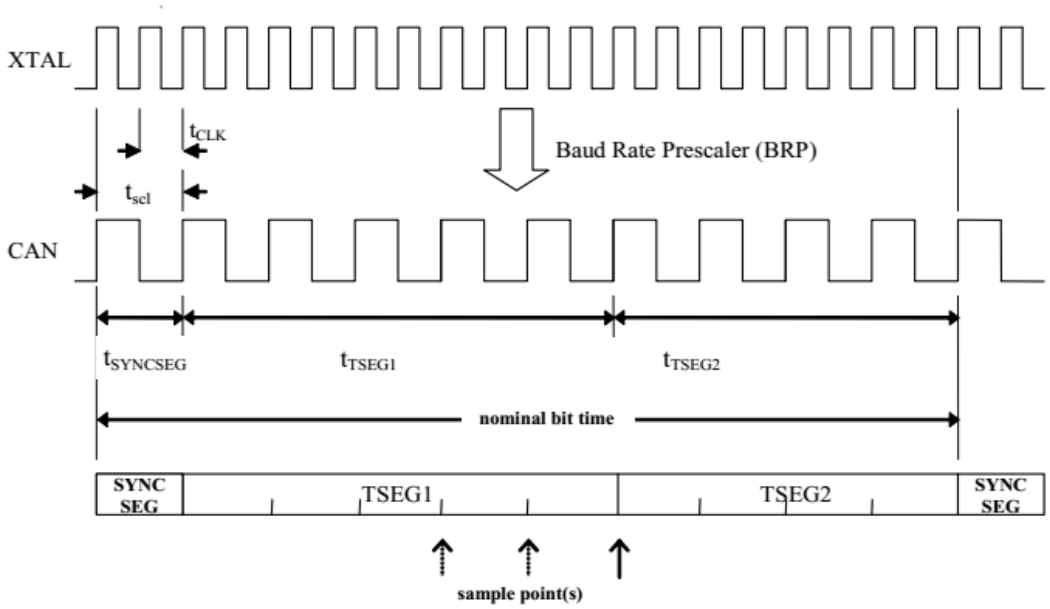


图 25-1：CAN 的位周期结构图

25.5.9 发送缓存寄存器（CAN_TXBUF）

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	TXBUF	W	0	发送缓存寄存器用于写入要通过CAN网络发送的CAN帧。 写入该寄存器执行内部写指针的自动递增，通过在ISR寄存器中写入TI位，可以将写指针复位到发送内存的地址0h处

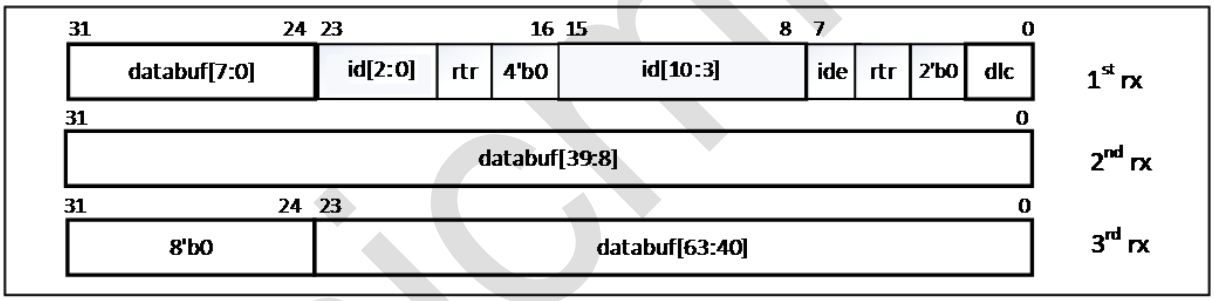
25.5.10 接收缓存寄存器（CAN_RXBUF）

偏移地址：0x0C

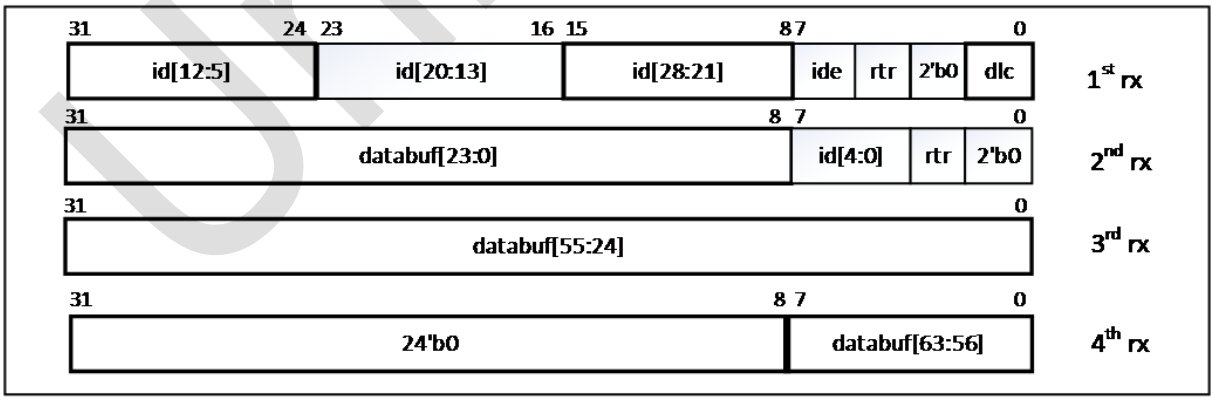
复位值：不确定

位	名称	属性	复位值	描述
31:0	RXBUF	R	不确定	接收缓存寄存器用于读取从CAN网络接收的CAN帧。 读取该寄存器将自动递增内部FIFO的读取地址指针（读取后递增）

在收到一帧 CAN 数据后，RXBUF 寄存器读到的数据格式如下（databuf 段长度由 DLC 段决定，数量为 0-8Bytes）：



CAN RX_FIFO for 11bits ID



CAN RX_FIFO for 29bits ID

图 25-2：RXBUF 寄存器数据格式

在接收完一帧 CAN 数据后，RMC 寄存器计数加 1，此时 CAN 控制器会往 RX FIFO 中逐个写入数据，当写入一个 32 位数据后，RBS 置位。在写入完一帧数据后，RI 标志位置位。

25.5.11 接收过滤匹配寄存器（CAN_ACR）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	ACR3-0	R/W	0	接收过滤匹配寄存器包含要接收的消息的仲裁位，而相应的接收过滤屏蔽寄存器定义了将比较哪些位位置和无关位

25.5.12 接收过滤屏蔽寄存器（CAN_AMR）

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	AMR3-0	R/W	0	接收过滤屏蔽寄存器定义了将比较哪些位位置和无关位。将相应的位设为1表示不对比ACR寄存器中相应的位

只有当接收到的消息的标识符位等于接收过滤器中的预定义位时，CAN 控制器中的接收过滤器才有可能将接收到的消息传递给 RX FIFO。接收过滤器由接收过滤匹配寄存器（ACR3:ACR0）和接收过滤屏蔽寄存器（AMR3:AMR0）组成。模式寄存器的 AFM 位可设置单/双过滤器。

不同过滤器设置以及不同仲裁长度（标准帧 11 位/拓展帧 29 位）对应的位格式如下图：

- 当设置为单过滤器时，过滤器为 4 字节长。

若接收的数据为标准帧模式，可接收到包括仲裁位，RTR 位和数据位的前 2 个字节（数据字节不是必须接收的部分）。所有单个位的比较都必须发出信号，表示成功接收到数据。

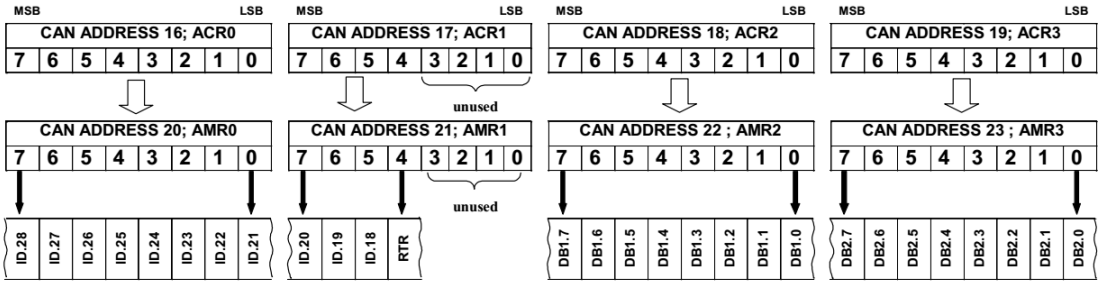


图 25-3：单过滤器标准帧对应位格式

若接收的数据为拓展帧格式，可接收到仲裁位和 RTR 位数据。对于格式中没定义的位，过滤器将不进行对比。

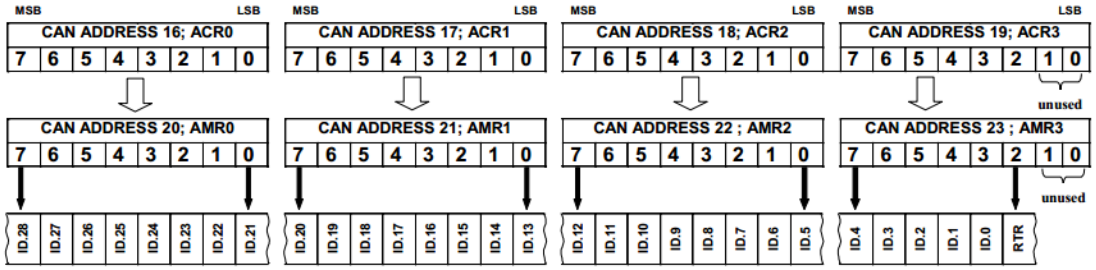


图 25-4：单过滤器扩展帧对应位格式

- 当设置为双过滤器时，双过滤器配置会定义两个长度更短的过滤器。接收到的数据将会跟两个过滤器对比，决定是否应该将数据存入 RX FIFO 中。如果至少一个接收过滤器对比成功，接收到的数据将被存储在 FIFO 中。

若接收的数据为标准帧模式，当接收到数据后，将会与第一个过滤器的 ID 包括 RTR 位，以及第一个收到的据字节进行对比，或者与第二个过滤器的 ID 位包括 RTR 位进行对比。

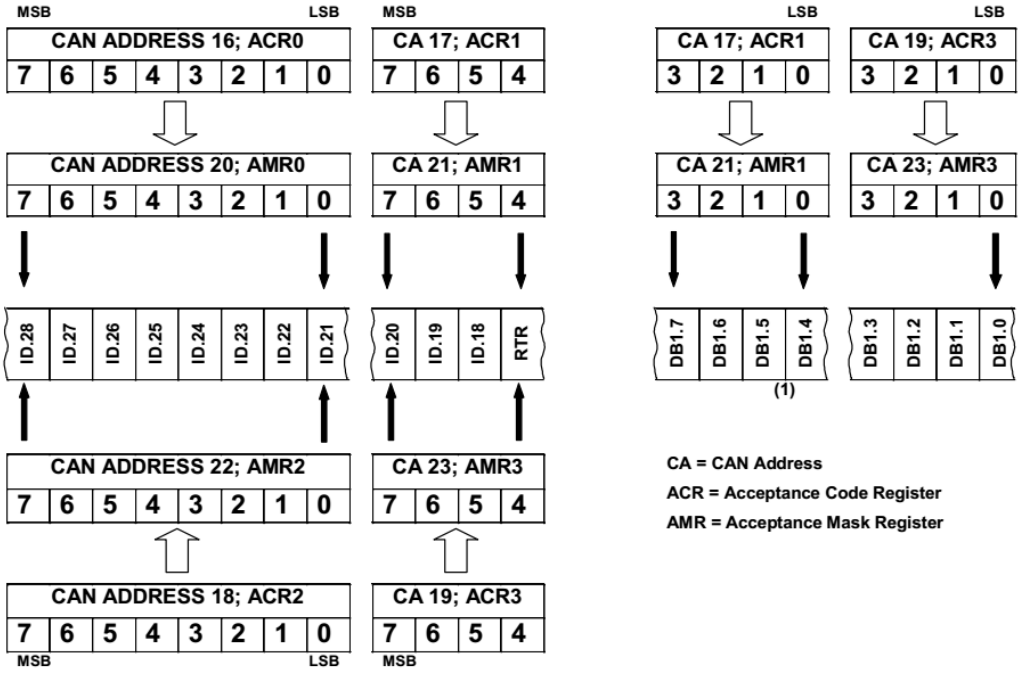


图 25-5：双过滤器标准帧对应位格式

若接收的数据为拓展帧格式，则两个过滤器仅比较扩展标识符范围的前两个字节。为了能成功接收，至少对其种一个滤波器的所有位进行比较。

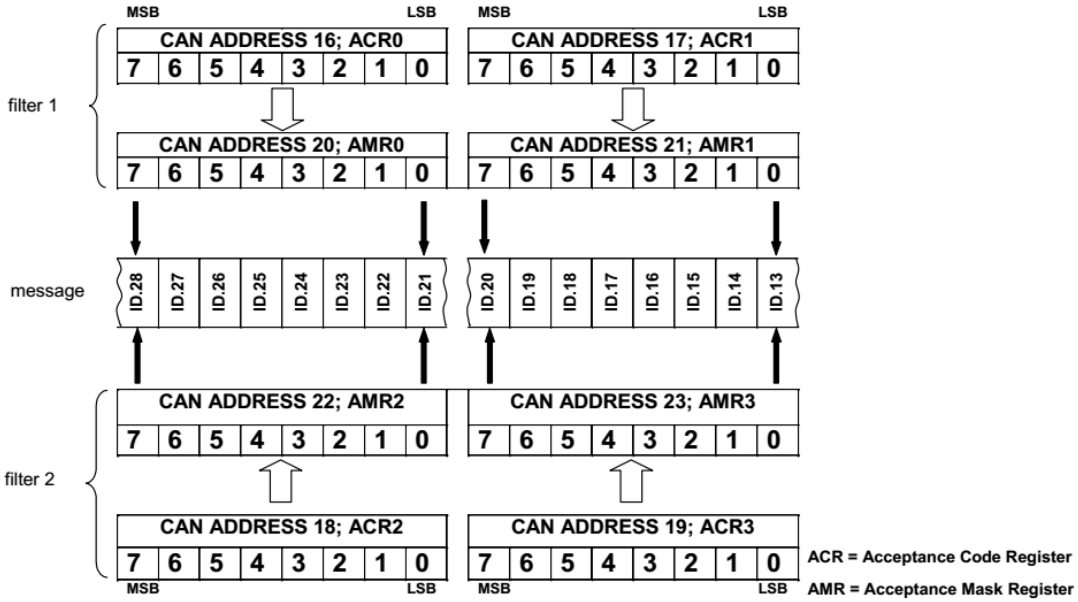


图 25-6：双过滤器扩展帧对应位格式

25.5.13 错误码捕获寄存器（CAN_ECC（ERRCR [7:0]））

ECC 只读寄存器保存有关 CAN 网络上发生的最后总线错误的错误代码。该寄存器是只读的。在确认先前的总线错误之前（通过确认总线错误中断），CAN 内核不会更新该寄存器。

偏移地址：0x18

复位值：0x00

位	名称	属性	复位值	描述
7	RXWRN	R	0	当RXERR计数器大于或等于96时置1
6	TXWRN	R	0	当TXERR计数器大于或等于96时置1
5	EDIR	R	0	表示错误发生时数据传输方向 0：发送 1：接收
4	ACKER	R	0	发生ACK错误时置位
3	FRMER	R	0	发生帧格式错误时置位
2	CRCER	R	0	发生CRC错误时置位
1	STFER	R	0	发生填充错误时置位
0	BER	R	0	发生位错误时置位

25.5.14 接收错误计数寄存器（CAN_RXERR（ERRCR [15:8]））

偏移地址：0x19

复位值：0x00

位	名称	属性	复位值	描述
7:0	RXERR	R	0	接收错误计数器的当前值。如果发生总线关闭事件，则RX错误计数器将初始化为0

25.5.15 发送错误计数寄存器（CAN_TXERR（ERRCR [23:16]））

偏移地址：0x1A

复位值：0x00

位	名称	属性	复位值	描述
7:0	TXERR	R	0	发送错误计数器的当前值的低8位。如果发生总线关闭事件，则将传输错误计数器初始化为127，以计算最小协议定义的时间（出现128次总线空闲信号）。在这段时间内读取TXERR可获得有关总线关闭恢复状态的信息

25.5.16 仲裁丢失捕获寄存器（CAN_ALC（ERRCR [31:24]））

CAN 控制器能够确定仲裁丢失的确切帧内位置。紧随其后，将产生“仲裁丢失中断”。 此外，在仲裁丢失捕获寄存器中捕获位数。一旦主机控制器读取了该寄存器的内容，就会为下一个仲裁丢失情况激活捕获功能。此功能允许 CAN 监视每个 CAN 总线访问。对于诊断或在系统配置期间，可以确定仲裁不成功的每种情况。

偏移地址：0x1B

复位值：0x00

位	名称	属性	复位值	描述
7:5	RSV	-	-	保留
4:0	ALC	R	0	仲裁丢失位置

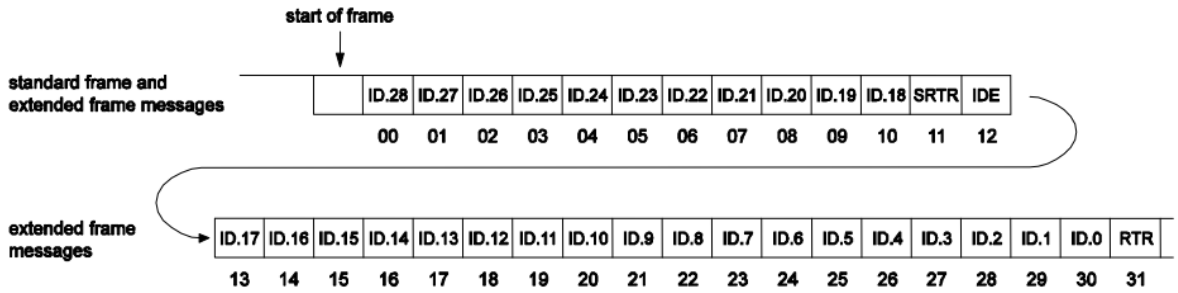


图 25-7：仲裁丢失位中断图

表 25-5：ALC 寄存器描述

Bits					Decimal Value	Description
ALC4	ALC3	ALC2	ALC1	ALC0		
0	0	0	0	0	00	Arbitration lost in **ID28 / 10**
0	0	0	0	1	01	Arbitration lost in **ID27 / 9**
0	0	0	1	0	02	Arbitration lost in **ID26 / 8**
0	0	0	1	1	03	Arbitration lost in **ID25 / 7**
0	0	1	0	0	04	Arbitration lost in **ID24 / 6**
0	0	1	0	1	05	Arbitration lost in **ID23 / 5**
0	0	1	1	0	06	Arbitration lost in **ID22 / 4**
0	0	1	1	1	07	Arbitration lost in **ID21 / 3**
0	1	0	0	0	08	Arbitration lost in **ID20 / 2**
0	1	0	0	1	09	Arbitration lost in **ID19 / 1**
0	1	0	1	0	10	Arbitration lost in **ID18 / 0**
0	1	0	1	1	11	Arbitration lost in **SRTR / RTR**
0	1	1	0	0	12	Arbitration lost in **IDE** bit
0	1	1	0	1	13	Arbitration lost in **ID17***
0	1	1	1	0	14	Arbitration lost in **ID16***
0	1	1	1	1	15	Arbitration lost in **ID15***
1	0	0	0	0	16	Arbitration lost in **ID14***
1	0	0	0	1	17	Arbitration lost in **ID13***
1	0	0	1	0	18	Arbitration lost in **ID12***
1	0	0	1	1	19	Arbitration lost in **ID11***
1	0	1	0	0	20	Arbitration lost in **ID10***
1	0	1	0	1	21	Arbitration lost in **ID9***
1	0	1	1	0	22	Arbitration lost in **ID8***
1	0	1	1	1	23	Arbitration lost in **ID7***
1	1	0	0	0	24	Arbitration lost in **ID6***
1	1	0	0	1	25	Arbitration lost in **ID5***
1	1	0	1	0	26	Arbitration lost in **ID4***
1	1	0	1	1	27	Arbitration lost in **ID3***
1	1	1	0	0	28	Arbitration lost in **ID2***
1	1	1	0	1	29	Arbitration lost in **ID1***
1	1	1	1	0	30	Arbitration lost in **ID0***
1	1	1	1	1	31	Arbitration lost in **RTR**

25.6 使用流程

25.6.1 发送 CAN 数据帧

1. 开启 CAN 时钟，释放复位，CAN 功能管脚复用。
2. 配置总线时序寄存器 CAN_BTR0/CAN_BTR1。
3. CAN_ISR 寄存器清除错误标志位/中断标志位。
4. 配置中断使能 CAN_IMR 寄存器，使能 TI 中断（可选）。
5. 配置模式 CAN_MR[1]，进入正常模式。
6. 配置发送缓存寄存器 CAN_TXBUF，根据定义的格式写入 CAN 数据帧内容，按发送的先后顺序写入，每次写入 32 位数据。
7. 配置指令寄存器 CAN_CMR[2]，启动发送。
8. 等待状态寄存器的 CAN_SR[5]置 1 后（若使能 TI 中断，此处可选择等待 TI 中断触发），数据发送完毕。

25.6.2 接收 CAN 数据帧

1. 开启 CAN 时钟，释放复位，CAN 功能管脚复用。
2. 配置总线时序寄存器 CAN_BTR0/CAN_BTR1。
3. CAN_ISR 寄存器清除错误标志位/中断标志位。
4. 配置中断使能 CAN_IMR 寄存器，使能 RI 中断（可选）。
5. 设置接收过滤器配置，若使用单过滤器，CAN_MR[0]置 1。CAN_ACR 寄存器配置用户需要过滤筛选的内容，CAN_AMR 寄存器选择需要与 CAN_ACR 寄存器进行对比的位。若不需要进行对，CAN_AMR 寄存器所有位置 1。
6. 配置模式寄存器 CAN_MR[1]，进入正常模式。
7. 等待状态寄存器 CAN_SR[7]置 1 后（若使能 RI 中断，此处可选择等待 RI 中断触发），读取接收缓存寄存器 CAN_RXBUF 数据，多次读取直到取出所有数据。

26 I2C 接口（I2C）

26.1 概述

I2C模块的作用是完成CPU对I2C总线上连接的从设备的读写。当CPU对从设备做写操作时，CPU通过总线配置I2C模块的配置寄存器，然后发送控制信息和操作数到I2C模块的数据通信寄存器；I2C模块解析命令后将数据通道寄存器的数据通过I2C总线发给从设备，发送完毕后将最终的状态通过中断反馈给CPU。CPU读取从设备数据的过程与写操作类似。

26.2 主要特性

- 双线I2C串行接口
- 支持标准模式100 Kb/s，快速模式400 Kb/s和快速模式+1 Mb/s
- 支持主机或从机模式
- 支持7位地址或10位地址
- 支持7位或10位组合格式传输
- 支持批量传输模式
- 8字节的发送和接收缓冲
- 支持中断和查询操作
- 支持所有速度下的位和字节等待
- 可编程SDA保持时间
- 支持总线清除
- 支持DMA操作
- 支持SMBus（System Management Bus）/PMBus（Power Management Bus）
- 支持SMBus从机检测并响应ARP命令
- 支持地址解析协议（Address resolution protocol（ARP）support）

26.3 管脚说明

表 26-1：I2C 管脚说明

功能管脚	复用管脚	方向	功能描述
I2C0_SDA	PB5,PB7,PB9	Input/Output	数据
I2C0_SCL	PB6,PB8	Input/Output	时钟
I2C1_SDA	PB11,PB12,PB15	Input/Output	数据
I2C1_SCL	PB10,PB14	Input/Output	时钟

功能管脚	复用管脚	方向	功能描述
I2C2_SDA	PA9,PC9,PD2	Input/Output	数据
I2C2_SCL	PA8,PC12	Input/Output	时钟

26.4 功能描述

26.4.1 SMBus/PMBus

SMBus用于在系统及其设备之间提供可预测的通信线路，它描述了设备超时定义及其条件。

26.4.1.1 总线协议

典型的SMBus具有一组命令，通过这些命令可以读取和写入数据。所有命令的长度都是一个字节，但它们的参数和返回值的长度可以不同。根据SMBus规格，最高有效位（MSB）先传输。对于给定的设备来说，有11种命令协议。这些命令有：Quick Command，Send Byte，Receive Byte，Write Byte，Write Word，Read Byte，Read Word，Process Call，Block Read，Block Write，and Block Write-Block Read Process Call。

对于消息事务的SMBus协议通常不同于I2C数据传输命令，但仍然可以通过对SMBus主机进行编程，进行I2C数据传输。下表描述了通过I2C的TX FIFO命令产生的SMBus总线协议。

在SMBus主机模式下，所有接收的数据在RX FIFO中都是可用的。在SMBus从机模式下，所有总线协议命令代码和数据字节将放在RX FIFO中，读取请求数据字节必须使用TX FIFO进行发送，类似于I2C模式。

表 26-2：SMBus 总线协议使用表

Protocol	Required TxFIFO Commands	DATA (I2C_DATA_CMD[7:0])	CMD (I2C_DATA_CMD[8])	STOP (I2C_DATA_CMD[9])	注释
Quick Command	1	不适用	设定R/W	设为 1	I2C_TAR的bit11和bit16设为1
Send Byte	1	Data Byte	设为 0	设为 1	
Receive Byte	1	不适用	设为 1	设为 1	
Write Byte	2	Command Code	设为 0	设为 0	
		Data Byte	设为 0	设为 1	
Write Word	3	Command Code	设为 0	设为 0	
		Data Byte Low	设为 0	设为 0	
		Data Byte High	设为 0	设为 1	
Read Byte	2	Command Code	设为 0	设为 0	

Protocol	Required TxFIFO Commands	DATA (I2C_DATA_CMD[7:0])	CMD (I2C_DATA_CMD[8])	STOP (I2C_DATA_CMD[9])	注释
		不适用	设为 1	设为 1	
Read Word	3	Command Code	设为 0	设为 0	
		不适用	设为 1	设为 0	
		不适用	设为 1	设为 1	
Process Call	5	Command Code	设为 0	设为 0	
		Data Byte Low	设为 0	设为 0	
		Data Byte High	设为 0	设为 0	
		不适用	设为 1	设为 0	
		不适用	设为 1	设为 1	
Block Write	N+1	Command Code	设为 0	设为 0	
		Data Byte	设为 0	设为 0	
		N+1) Data Byte N	设为 0	设为 1	
Block Read	N+1	Command Code	设为 0	设为 0	
		不适用	设为 0	设为 0	
		N+1) 不适用	设为 0	设为 1	
Block Write-Block Read Process Call	M+N+1	Command Code	设为 0	设为 0	
		Data Byte 1	设为 0	设为 0	
		M+1) Data Byte M	设为 0	设为 0	
		M+2) 不适用	设为 1	设为 0	
		M+3) 不适用	设为 1	设为 0	
		M+N+1) 不适用	设为 1	设为 1	
SMBUS Host Notify Protocol	3	Data Byte Low	设为 0	设为 0	I2C_TAR设为SMBus主机地址（0001 000）

I2C从机仅在I2C_CR的SMBUS_SLAVE_QUICK_CMD_EN使能的情况下，通过Quick命令进行接收。每当选择该位时，从机仅接收Quick命令，不接受其他总线协议。I2C从机在收到Quick命令时发出SMBUS_QUICK_DET中断。

SMBus通过在总线协议末尾附加PEC字节引入了数据包错误检查机制。这可以通过在传输时添加额外命令（PEC字节）以及接收时通过软件对其进行解码来实现。

26.4.1.2 SMBus 地址解析协议

SMBus从机地址冲突可以通过主机为每个从机设备动态分配新的唯一地址来解决。这特性允许设备进行“热插拔”。

SMBus为系统中的每个设备引入了128位唯一设备ID（UDID），以隔离每个设备以进行地址分配。UDID的高96bit为0，低32bit由I2C_SMBUS_ARP_UDID_LSB寄存器控制。

I2C使用IC_CR寄存器中的SMUBS_PERSISTANT_SLV_ADDR_EN位来指示I2C是否支持持久从机地址。

I2C主机可以发出通用和定向地址解析协议（ARP）命令，为SMBus系统中的从机分配动态地址。

表 26-3：通过 I2C 中的 TxFIFO 命令派生 SMBus ARP 命令表

ARP Command	Required TXFIFO Commands	Command/ Data (I2C_DATA_CMD[7:0])	CMD Bit (I2C_DATA_CMD[8])	STOP Bit (I2C_DATA_CMD[9])	注释
Prepare for ARP	2	Command = '0000 0001'	设为 0	设为 0	I2C_TAR[6:0]设为SMBus默认地址（1100 001）
		PEC Byte	设为 0	设为 1	
Reset Device (General)	2	Command = '0000 0010'	设为 0	设为 0	I2C_TAR[6:0]设为SMBus默认地址（1100 001）
		PEC Byte	设为 0	设为 1	
Get UDID (General)	20	Command = '0000 0011'	设为 0	设为 0	1. I2C_TAR[6:0]设为SMBus默认地址（1100 001） 2. 对128个UDID字节执行16次读取 3. 最后一个读命令是从机地址
		不适用	设为 1	设为 0	
		不适用	设为 1	设为 0	
		不适用	设为 1	设为 0	
		PEC Byte	设为 1	设为 1	
Assign Address	20	Command = '0000 0100'	设为 0	设为 0	1. I2C_TAR[6:0]设为SMBus默认地址（1100 001） 2. 对 128 个 UDID 字节执行 16 次写 3. 最后一个写命令是从机地址
		Byte Count = 17	设为 0	设为 0	-
		UDID Byte 15	设为 0	设为 0	
		UDID Byte 14	设为 0	设为 0	
		Assigned Address	设为 0	设为 0	

ARP Command	Required TXFIFO Commands	Command/ Data (I2C_DATA_CMD[7:0])	CMD Bit (I2C_DATA_CMD[8])	STOP Bit (I2C_DATA_CMD[9])	注释
		PEC Byte	设为 01	设为 1	
Get UDID (Directed)	19	Command = '0000 0011'	设为 0	设为 0	1. I2C_TAR[6:0]设为SMBus默认地址（1100 001） 2. 对128个UDID字节执行16次读取 3. 最后一个读命令是从机地址
		{Slave address[6:0],1}	设为 1	设为 0	
		不适用	设为 1	设为 0	
		不适用	设为 1	设为 0	
		PEC Byte	设为 1	设为 1	
Reset Device (Directed)	2	command = {slave address[6:0],0}	设为 0	设为 0	I2C_TAR[6:0]设为SMBus默认地址（1100 001）
		PEC byte	设为 0	设为 1	

26.4.1.3 主机模式下执行 ARP

当I2C用作SMBus主机为每个从设备分配唯一地址以解决从设备地址冲突时，请执行以下步骤：

1. 当复位或冷启动后，SMBus主机发出“Prepare for ARP”命令，表明主机执行ARP以便为所有设备分配动态地址。从机必须刷新任何挂起的主机通知命令。
2. 当接收到的“Prepare to ARP”命令的ACK时，表明系统中存在支持ARP的设备，然后发行“Get UDID”命令。NACK表示不存在支持ARP的设备，或者当前所有从机都有解析的地址。在这种情况下，主机必须完成step8。
3. I2C主机发出“Get UDID”以接收从机的UDID信息，用于分配动态地址。
4. 如果“Get UDID”命令的前三个字节都已ACK，并且接收字节计数位0x11，那么主机发行“Assign Address”命令。否则，主机必须完成步骤8中所述的步骤，以指示ARP已完成。
5. 主机发行“Assign Address”命令，给通过“Get UDID”命令接收的UDID从机动态分配地址。
6. 如果分配到的地址包已ACK，则主机将从地址池中删除已分配的地址并跳到步骤3以获取另一个从设备的UDID。如果地址包已NACK，主机不会从地址池中移除这个地址，然后跳至步骤3获取同一个从机或其他从机的UDID。
7. 如果分配到的地址包已ACK，则主机将分配的地址与设备的UDID特征一起存储在已用地址池中。
8. 主机跳至步骤3再次发行“Get UDID”命令接收其他从机的UDID。如果接收到“Get UDID”的NACK，主机跳至步骤9。

9. I2C可以切换到从机模式，以检测设备对主机通知协议的请求。
10. 如果I2C切换到从机模式且检测到主机通知协议，表明从机正在请求动态地址，主机必须按照步骤11所述进行ARP。
11. 如果在主机模式，跳至步骤3执行ARP，否则跳至步骤12。
12. I2C切换至主机模式，跳至步骤3执行ARP。

26.4.1.4 从机模式下执行 ARP

I2C作为SMBus从机执行以下任务：

- 解码ARP命令并根据内部状态标志进行响应
  I2C_SR寄存器的SMBUS_SLAVE_ADDR_VALID和SMBUS_SLAVE_ADDR_RESOLVED。
- 生成并验证ARP命令的PEC字节。
- 仅当PEC字节与根据其接收到的数据计算的CRC值相匹配时，才会为PEC字节生成ACK。如果不是，则NACK PEC字节。

当总线上的另一个SMBus主机产生ARP命令和请求已参加ARP时，I2C作为从机执行以下步骤：

1. 当复位或冷启动后，I2C从机检查是否支持永久从地址。
2. 如果 I2C 有一个永久从地址（PSA），该地址由设置的地址有效标志指示，PSA 在从机地址寄存器（I2C_SAR）寄存器中设置。如果该标志未设定，则执行步骤 4。
3. I2C 持久从机将持久地址存储在 I2C_SAR 中，并将地址有效标志设置为 1，地址解析标志为 0。
4. I2C 非持久从机（非 PSA）清除地址有效和地址解析标志。
5. I2C 检查接收的数据包中的从机地址字段中是否有 ARP 默认地址，以决定是 ARP 命令还是正常命令。如果匹配，那么从机进入步骤 6，否则跳至步骤 25。
6. 如果 I2C 检测到发往 SMBus 设备默认地址的数据包，它会检查命令字段以确定这是否是“准备 ARP”命令。如果是，则进入步骤 7，否则进入步骤 8。
7. 收到“Prepare to ARP”命令后，I2C 确认数据包并清除地址解析标志，以便参与 ARP 过程。I2C 继续执行步骤 5 并等待另一个 SMBus 数据包。
8. I2C检查命令字段，以验证是否发出了“Reset Device”命令。如果是，则I2C进入步骤9，否则进入步骤10。
9. 收到“Reset Device”命令后，I2C确认数据包并清除地址和地址有效（如果是非PSA，I2C_CR[19]=0）标志。I2C进入步骤5并等待另一个SMBus数据包。
10. 设备检查该命令，以验证是否发出了“Assign Address”命令。如果是，则进入步骤11，否则进入步骤13。
11. 收到“Assign Address”命令后，I2C将其UDID与接收到的字节进行比较。如果任何字节不匹配，

则I2C将不会确认该字节以及后续字节。如果UDID中的所有字节都匹配，则设备进入步骤12，否则进入步骤5并等待另一个SMBus数据包。

12. 在步骤 11 中匹配 UDID 后，DW_apb_i2c 将接收从机地址，并使用该从机地址设置 I2C_SAR 寄存器。I2C 设置其地址有效和地址解析标志，这意味着它已收到动态地址，并且不再响应“Get UDID”命令，除非收到“Prepare to ARP”或“Reset Device”命令。I2C 现在进入步骤 5 并等待另一个 SMBus 数据包。
13. I2C 检查命令字段，以验证是否发出了“Get UDID”命令。如果是，则进入步骤 14，否则进入步骤 19。
14. 收到“Get UDID”命令后，I2C 检查其地址解析标志，以确定是否必须参与 ARP 进程。如果被置起，则其地址已由 ARP 主机解析，因此设备继续执行步骤 5 并等待另一个 SMBus 数据包。如果 ARP 标志已被清除，然后进入步骤 15。
15. I2C 返回其 UDID 并监测 SMBus 数据线是否存在冲突。如果在任何时候检测到冲突，I2C 将置位 SLV_ARB_LOST 位并停止传输。此外，它继续执行步骤 5 并等待另一个 SMBus 数据包。如果未检测到冲突，则 I2C 进入步骤 16。
16. I2C 检查其地址有效（AV）标志，以确定返回到设备从属地址字段的值。如果 AV 标志被置起，则进入步骤 17，否则进入步骤 18。
17. AV 标志被置位时，当前 I2C_SAR 有效，因此设备会将其返回给设备从属地址字段，并监测 SMBus 数据线的冲突。I2C 续执行步骤 5 并等待另一个 SMBus 数据包。
18. AV 标志未被置位时，当前从机地址（I2C_SAR）无效。因此，I2C 返回 FFh 值，并监测 SMBus 数据线的冲突。如果 ARP 主机接收到 FFh 值，则设备需要地址分配。I2C 继续执行步骤 5 并等待另一个 SMBus 数据包。
19. I2C 可能正在接收直接命令。如果地址有效标志被置位且地址与 I2C_SAR 中的地址相同，则继续执行步骤 20，否则，继续执行步骤 5 以等待另一个 SMBus 数据包。
20. 如果地址有效标志被置位，检查该命令是否为“Reset Device”直接命令。如果是，则转至步骤 21，否则转至步骤 22。
21. 收到“Reset Device”命令后，I2C 确认数据包并清除地址解析和地址有效（如果非 PSA，I2C_CR[19]=0）标志。I2C 进入步骤 5 并等待另一个 SMBus 数据包。
22. I2C 检查收到的命令是否为“Directed Get UDID”命令。如果是，则转至步骤 23 并返回 UDID 信息。如果没有，则转至步骤 24。
23. 如果收到的命令是“Directed Get UDID”命令，则返回 UDID 信息和当前从机地址，转至步骤 5 并等待另一个 SMBus 数据包。
24. 如果接收到的命令是“Directed Get UDID”命令，则 I2C 未接收到有效的 ARP 命令，因此 I2C 会取消该命令并继续执行步骤 5 并等待另一个 SMBus 数据包。
25. 如果地址有效位被置位，则进入步骤 26，否则进入步骤 5 并等待另一个 SMBus 数据包。接收

到的地址不是 SMBus 设备默认地址，数据包可能是 I2C 核心功能的地址。设备检查其地址有效位以确定是否响应。

26. 当地址有效位被置位时，I2C 具有有效的从机地址。它将接收到的从机地址与其从机地址进行比较，如果匹配，I2C 将进入步骤 27，否则将进入步骤 5 并等待另一个 SMBus 数据包。
27. I2C 接收其核心功能的数据包地址，因此它确认该数据包并相应地对其进行处理。I2C 继续执行步骤 5 并等待另一个 SMBus 数据包。

26.4.2 总线清除

I2C 支持总线清除功能，在时钟或数据线被非正常拉低时，提供数据（SDA）和时钟（SCL）线的恢复。

26.4.2.1 SDA 拉低恢复

如果 SDA 线路被拉低，主机将执行以下操作，如图 26-1 和图 26-2 所示

1. 主机最多发送 9 个时钟脉冲，以恢复这 9 个时钟内的总线低电平。
2. 如果 SDA 在这 9 个时钟脉冲内恢复，主机会发送 STOP 释放总线。
3. 如果 SDA 在第九个时钟后仍未恢复，系统需要硬件复位。

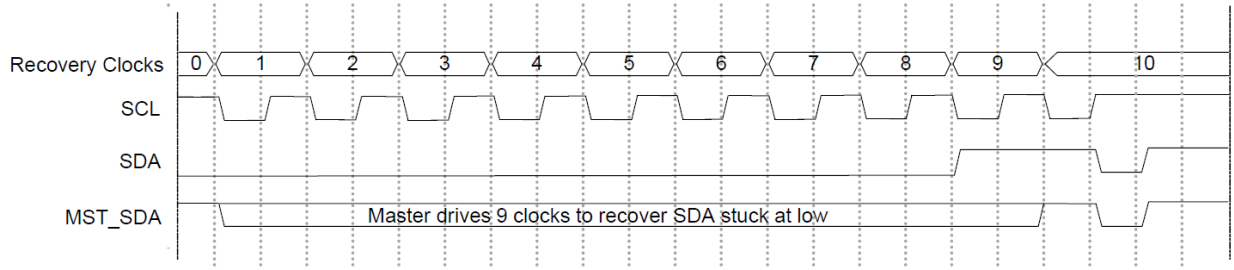


图 26-1：使用 9 个 SCL 时钟恢复 SDA 图

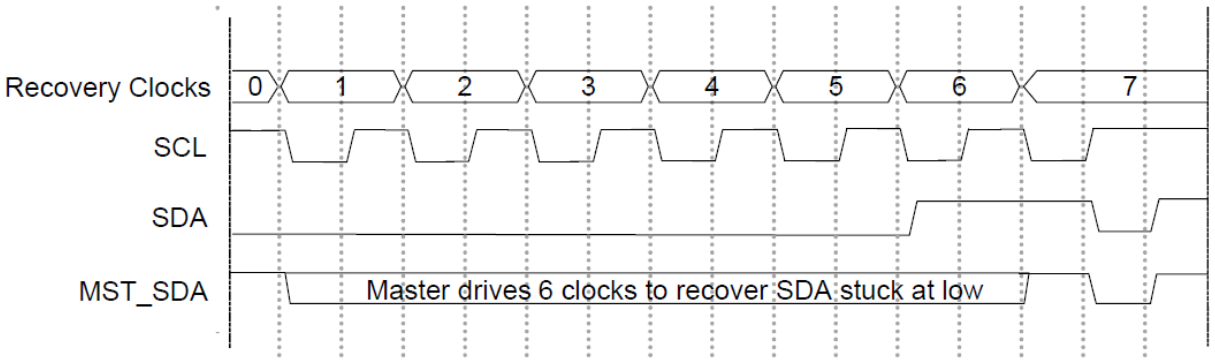


图 26-2：使用 6 个 SCL 时钟恢复 SDA 图

26.4.3 配置 I2C SCLHCNT 和 SCLLCNT

当I2C作为标准模式100Kb/s、快速模式400Kb/s或快速模式+1Mb/s的主机时，为保证I/O时序，*CNT寄存器必须在I2C传输前设定好，*CNT寄存器如下：

- I2C_SSSCLHCNT
- I2C_SSSCLLCNT
- I2C_FSSCLHCNT
- I2C_FSSCLLCNT

26.4.3.1 高、低电平最小设定值

- I2C_SSSCLLCNT和I2C_FSSCLLCNT寄存器必须大于I2C_FSSPKLEN+7
- I2C_SSSCLHCNT和I2C_FSSCLHCNT寄存器必须大于I2C_FSSPKLEN+5

26.4.3.2 SCLHCNT 和 SCLLCNT 计算方法

以下给出了如何计算快速模式下的I2C_FSSCLHCNT和I2C_FSSCLCNT，同理也可计算出标准模式和快速模式+下的值

- 快速模式400kb/s，SCL周期为2.5us
- 例如I2C的APB时钟是48MHz，则I2C_CLK周期是20.8ns
- 根据协议，SCL的高、低电平最小时间

  MIN_SCL_LOWtime_FS = 1300ns

  MIN_SCL_HIGHtime_FS = 600ns

  MIN_SCL_LOWtime_SS = 4700ns

  MIN_SCL_HIGHtime_SS = 4000ns

  MIN_SCL_LOWtime_FS+ = 500ns

  MIN_SCL_HIGHtime_FS+ = 260ns

$$\frac{\text{SCL\_PERIOD\_FS}}{\text{I2C\_FSSCLHCNT} + \text{I2C\_FSSCLLCNT}} = \text{I2C\_CLK\_PERIOD}$$

I2C_FSSCLLCNT x I2C_CLK_PERIOD = MIN_SCL_LOWtime_FS

将SCL和I2C时钟频率代入等式：

$$\frac{2500}{\text{I2C\_FSSCLHCNT} + \text{I2C\_FSSCLLCNT}} = 20.8$$

I2C_FSSCLLCNT×20.8=1300

求解I2C_FSSCLHCNT和I2C_FSSCLLCNT:

I2C_FSSCLHCNT+I2C_FSSCLLCNT=120.9

I2C_FSSCLLCNT=62.5

I2C_FSSCLLCNT向上舍入得I2C_FSSCLLCNT=63，则I2C_FSSCLHCNT=58

26.4.4 SDA 保持时间

I2C协议规范要求在标准模式和快速模式下，SDA信号的保持时间最小为300ns，快速模式+下，最小位0ns。

由于每个应用程序都会遇到不同的线路延迟，I2C包含一个可软件编写的I2C_SDA_HOLD寄存器，用于动态调整SDA保持时间。

Bit[15:0]用于主、从传输期间SDA的保持时间（SDA从高到低）。

Bit[23:16]用于扩展SDA转换（如果有），只要接收侧的SCL为高时（主机模式或从机模式）。

如果不同的速度模式需要不同的SDA保持时间，则在速度模式改变时，必须重新编写I2C_SDA_HOLD寄存器。只有在I2C禁用（I2C_EN[0]=0）时，才能对I2C_SDA_HOLD寄存器进行编写。

26.4.4.1 接收中的 SDA 时序

速度模式	RX_HOLD最大值
标准模式	I2C_SS_SCL_HCNT – I2C_FS_SPKLEN - 3
快速模式或快速模式+	I2C_SS_SCL_HCNT – I2C_FS_SPKLEN - 3

26.4.4.2 发送中的 SDA 时序

TX_HOLD寄存器可用于改变I2C生成的SDA（ic_sda_oe）的时序。TX_HOLD寄存器中的每个值表示一个I2C时钟周期。

当I2C在主机模式下，保持时间最小为一个I2C时钟周期。因此，当TX_HOLD被设为0时，在SCL变为0后I2C仍将驱动SDA一个I2C时钟周期。对于TX_HOLD中的其他值：

- 当SCL变为0后，I2C驱动SDA TX_HOLD * I2C时钟周期

当I2C在从机模式下，最小保持时间为（I2C_FS_SPKLEN + 7）* I2C时钟周期。因此，当TX_HOLD的值小于I2C_FS_SPKLEN+7时，在SCL变为0后，I2C仍将驱动SDA（I2C_FS_SPKLEN + 7）* I2C时钟周期，对于TX_HOLD中的其他值：

- 当SCL变为0后，I2C驱动SDA TX_HOLD * I2C时钟周期

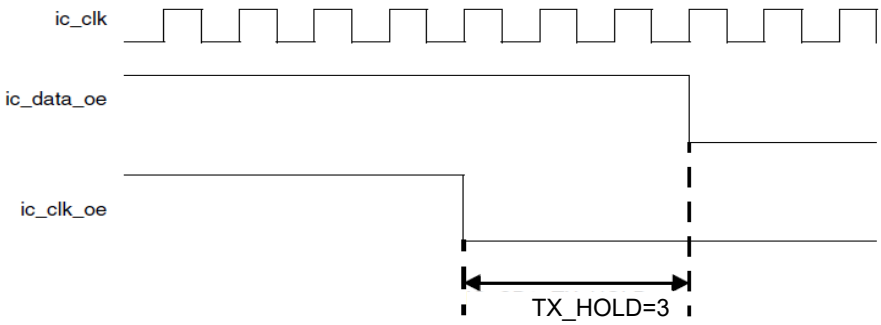


图 26-3：I2C 主机模式 TX_HOLD = 3 时序图

26.4.5 利用 DMA 通信

I2C接口支持用DMA来发送和接收数据。通过设置DMA寄存器中的对应位可以单独开启DMA发送或者DMA接收。发送时数据寄存器变空或接收时数据寄存器变满，则产生DMA请求。DMA请求必须在当前字节传输结束之前被响应。

- **DMA 发送**

通过设置DMA寄存器的TXEN位可以激活DMA发送模式。为I2C分配好DMA通道后，当发送数据时，DMA控制器会将数据从预置的存储区装载进DR寄存器。

- **DMA 接收**

通过设置DMA寄存器的RDMAE位可以激活DMA接收模式。为I2C分配好DMA通道后，当每次接收到数据字节时，DMA控制器会将数据从DR寄存器中传送到预置的存储区。

26.5 寄存器描述

I2C0寄存器基地址：0x4600_7000

I2C1寄存器基地址：0x4600_8000

I2C2寄存器基地址：0x4600_9000

寄存器列表如下：

表 26-4：I2C寄存器列表

偏移地址	名称	描述
0x00	I2C_CR	I2C控制寄存器
0x04	I2C_TAR	I2C访问从机地址寄存器
0x08	I2C_SAR	I2C从机地址寄存器
0x10	I2C_DATACMD	I2C数据命令寄存器
0x14	I2C_SSSCLHCNT	I2C标准模式SCL高电平配置寄存器
0x18	I2C_SSSCLLCNT	I2C标准模式SCL低电平配置寄存器
0x1C	I2C_FSSCLHCNT	I2C快速模式SCL高电平配置寄存器
0x20	I2C_FSSCLLCNT	I2C快速模式SCL低电平配置寄存器

偏移地址	名称	描述
0x2C	I2C_ISR	I2C中断状态寄存器
0x30	I2C_INTMASK	I2C中断屏蔽寄存器
0x34	I2C_RAWISR	I2C RAW中断状态寄存器
0x38	I2C_RXTL	I2C接收FIFO阈值寄存器
0x3C	I2C_TXTL	I2C发送FIFO阈值寄存器
0x40	I2C_CLR	I2C组合和独立中断清除寄存器
0x44	I2C_CLRRXUNDER	I2C清除_RX_UNDER中断寄存器
0x48	I2C_CLRRXOVER	I2C清除_RX_OVER中断寄存器
0x4C	I2C_CLRTXOVER	I2C清除_TX_OVER中断寄存器
0x50	I2C_CLRRDREQ	I2C清除_RD_REQ中断寄存器
0x54	I2C_CLRTXABRT	I2C清除_TX_ABRT中断寄存器
0x58	I2C_CLRRXDONE	I2C清除_RX_DONE中断寄存器
0x5C	I2C_CLRACTIVITY	I2C清除_ACTIVITY中断寄存器
0x60	I2C_CLRSTOPDET	I2C清除_STOP_DET中断寄存器
0x64	I2C_CLRSTARTDET	I2C清除_START_DET中断寄存器
0x68	I2C_CLRGENCALL	I2C清除_GEN_CALL中断寄存器
0x6C	I2C_EN	I2C使能寄存器
0x70	I2C_SR	I2C状态寄存器
0x74	I2C_TXFLR	I2C发送缓冲深度寄存器
0x78	I2C_RXFLR	I2C接收缓冲深度寄存器
0x7C	I2C_SDAHOLD	I2C SDA保持时间寄存器
0x84	I2C_SLVDATANACKONLY	I2C生成从机数据Nack寄存器
0x88	I2C_DMACR	I2C DMA控制寄存器
0x8C	I2C_DMATDLR	I2C DMA发送数据级别寄存器
0x90	I2C_DMARDLR	I2C DMA接收数据级别寄存器
0x94	I2C_SDASETUP	I2C SDA建立时间寄存器
0x98	I2C_ACKGENERALCALL	I2C广播呼叫ACK寄存器
0x9C	I2C_ENSR	I2C使能状态寄存器
0xA0	I2C_FSSPKLEN	I2C尖峰抑制极限寄存器
0xBC	I2C_SMBUSCLOCKLOWSEXT	I2C SMBUS从机时钟延长超时寄存器
0xC0	I2C_SMBUSCLOCKLOWMEXT	I2C SMBUS主机时钟延长超时寄存器
0xC4	I2C_SMBUSTHIGHMAXIDLECOUNT	I2C SMBUS最大总线空闲计数寄存器
0xC8	I2C_SMBUSISR	I2C SMBUS中断状态寄存器
0xCC	I2C_SMBUSINTMASK	I2C SMBUS中断屏蔽寄存器
0xD0	I2C_SMBUSRAWISR	I2C SMBUS RAW中断状态寄存器
0xD4	I2C_CLRSMBUSISR	I2C清除SMBUS中断状态寄存器
0xDC	I2C_SMBUSUDIDLSB	I2C SMBUS ARP UDID LSB寄存器

26.5.1 I2C 控制寄存器（I2C_CR）

偏移地址：0x00

复位值：0x0000 0065

位	名称	属性	复位值	描述
31:20	RSV	-	-	保留
19	SMBUS_PERSISTANT_SLV_ADDR_EN	R/W	0x0	控制启用I2C作为持久或非持久从机。
18	SMBUS_ARP_EN	R/W	0x0	仅限从机模式。 控制I2C是否在SMBus模式下启用地址解析逻辑。
17	SMBUS_SLAVE_QUICK_CMD_EN	R/W	0x0	仅限从机模式： 0：I2C接收除QUICK命令以外的所有协议 1：在SMBus模式下，I2C只接收QUICK命令
16:12	RSV	-	-	保留
11	BUS_CLEAR_FEATURE_CTRL	R/W	0x0	仅限主机模式： 0：使能总线清除功能 1：禁用总线清除功能
10:9	RSV	-	-	保留
8	TX_EMPTY_CTRL	R/W	0x0	该位控制TX_EMPTY中断产生，细节参考I2C_RAW_INTR_SR寄存器
7	STOP_DET_IFADDRESSED	R/W	0x0	在从机模式下，是否产生STOP_DET中断： 1：当地址匹配时才产生STOP_DET中断 0：无论地址是否匹配，都产生STOP_DET中断 该位仅适用于从机模式 注：广播地址寻址时，如果该位置位，从机不产生STOP_DET中断。STOP_DET中断仅当发送的地址与从机地址匹配时产生。
6	SLAVE_DISABLE	R/W	0x1	该位控制I2C接口从机禁止： 0：从机使能 1：从机禁止
5	RESTART_EN	R/W	0x1	当作为主机时该位控制是否发送RESTART条件： 0：禁止 1：使能 当RESTART禁止，I2C接口作为主机时不能执行以下功能： ● 发送起始字节 ● 组合格式模式下改变传输方向 ● 10位的地址格式的读操作 替换RESTART条件后，后面发送停止条件再发送起始条件时，拆分成多个I2C传输。 如果上述操作执行会置位IC_RAW_INTR_STAT寄存器的位6（TX_ABRT）
4	10ADDR_MASTER	R	0x0	I2C作为主机时的地址格式： 0：7位的地址格式 1：10位的地址格式

位	名称	属性	复位值	描述
3	10ADDR_SLAVE	R/W	0x0	当作为从机时，该位控制响应10位或者7位地址： 0：7位的寻址地址。I2C接口忽略处理10位的寻址。对于7位寻址，仅比较IC_SAR寄存器的低7位 1：10位的寻址地址。I2C仅响应10位的寻址，接收地址与IC_SAR的10位比较
2:1	SPEED	R/W	0x2	该两位控制I2C接口工作的速率模式 该设置仅当I2C接口工作在主机模式下有效： 1：标准模式(0-100Kb/s) 2：快速模式(400Kb/s)或快速模式+（0-1000Kb/s）
0	MASTER_MODE	R/W	0x1	该位控制主机模式： 0：主机禁止 1：主机使能

SLAVE_DISABLE(I2C_CR[6])和MASTER_MODE(I2C_CR[0])的配置如下表所列。

表 26-5：I2C SLAVE 和 MASTER 配置表

SLAVE_DISABLE (I2C_CR[6])	MASTER_MODE (I2C_CR[0])	State
0	0	从机
0	1	设定错误
1	0	设定错误
1	1	主机

26.5.2 I2C 访问从机地址寄存器（I2C_TAR）

偏移地址：0x04

复位值：0x0000 0055

位	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16	SMBUS_QUICK_CMD	R/W	0x0	当11（SPECIAL）=1时，该位表示I2C是否执行QUICK命令
15:12	RSV	-	-	保留
11	SPECIAL	R/W	0x0	该位指示软件执行的是否是特殊命令（广播呼叫或者起始字节命令）： 0：忽略第10位GC，正常使用ADDR位 1：执行特殊I2C命令如GC位描述
10	GC_OR_START	R/W	0x0	如果位11置位，该位显示I2C执行的是广播呼叫还是起始字节： 0：广播呼叫地址。发送广播呼叫地址时只能执行写操作。I2C接口一直工作在广播地址模式下直到SPECIAL(11)的值被清零 1：起始字节命令
9:0	ADDR	R/W	0x55	主操作的目标地址： 当发送一个广播地址，这些位就可以忽略。要产生开始字节的命令，CPU只需要对这些位写一次。

26.5.3 I2C 从机地址寄存器（I2C_SAR）

偏移地址：0x08

复位值：0x0000 0055

位	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9:0	ADDR	R/W	0x55	当I2C接口工作在从机模式下，保存了从机地址。对于7位的地址格式，ADDR[6:0]有效。

26.5.4 I2C 数据命令寄存器（I2C_DATACMD）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10	RESTART	W	0x0	发送或接收之前，是否产生一个RESTART信号： 1：如果RESTART_EN信号为”1’，数据接收或发送（根据CMD的值）前产生一个RESTART的信号，无论前一个命令是否改变数据的传输方向。如果RESTART_EN信号为‘0’，STOP信号后紧跟START信号 0：如果RESTART_EN信号为‘1’，仅在前一个命令改变传输方向时才产生RESTART信号。如果RESTART_EN信号为‘0’，STOP信号后紧跟START信号
9	STOP	W	0x0	STOP：发送或接收之后，是否产生一个STOP信号： 1：当前字节之后产生一个STOP信号，无论TXFIFO是否为空。如果TXFIFO不为空，主机立即发出一个新的传输及总线仲裁信号 0：当前字节之后不产生一个STOP信号，无论TXFIFO是否为空。主机继续当前传输（发送或接收数据根据CMD的值）。如果TXFIFO为空，主机将拉低SCL挂起总线直至TXFIFO收到新数据
8	CMD	W	0x0	控制在主模式下执行读或写操作： 1：读 0：写 当一个命令进入TXxFIFO，该位用于区分读写命令。在从接收模式下，该位写值操作被忽略。在从发送模式下，0表示DATA中的数据已发送。

位	名称	属性	复位值	描述
7:0	DATA	R/W	0x0	I2C总线待发送或者接收到的数据。 当对DATA进行写操作后再进行读，此时的读无效。但是，在读取该寄存器时，返回的是I2C接收到的数据值。

26.5.5 I2C 标准模式 SCL 高电平配置寄存器（I2C_SSSCLHCNT）

偏移地址：0x14

复位值：0x0000 0028

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	SS_SCL_HCNT	R/W	0x28	I2C接口标准模式下SCL时钟高电平周期 注意：该寄存器可配置值在6和65525之间，这是由于I2C接口使用了一个16位的计数器，该计数器值等于I2C_SSSCLHCNT+10时标志着I2C总线处于空闲状态。

26.5.6 I2C 标准模式 SCL 低电平配置寄存器（I2C_SSSCLLCNT）

偏移地址：0x18

复位值：0x0000 002F

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	SS_SCL_LCNT	R/W	0x2F	I2C接口标准模式下SCL时钟低电平周期 最小值为8

26.5.7 I2C 快速模式 SCL 高电平配置寄存器（I2C_FSSCLHCNT）

偏移地址：0x1C

复位值：0x0000_0006

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	FS_SCL_HCNT	R/W	0x6	I2C接口快速模式下SCL时钟高电平周期 最小值为6

26.5.8 I2C 快速模式 SCL 低电平配置寄存器（I2C_FSSCLLCNT）

偏移地址：0x20

复位值：0x0000 000D

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	FS_SCL_LCNT	R/W	0xD	I2C接口快速模式下SCL时钟低电平周期 最小值为8。

26.5.9 I2C 中断状态寄存器（I2C_ISR）

偏移地址：0x2C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11	GEN_CALL	R	0x0	General call请求被接收的中断状态： 0：无中断 1：有中断
10	START_DET	R	0x0	Start detect中断状态，指示I2C总线接口是否有START或RESTART条件： 0：无中断 1：有中断
9	STOP_DET	R	0x0	Stop detect中断状态，指示I2C总线接口是否有STOP条件： 0：无中断 1：有中断
8	ACTIVITY	R	0x0	activity中断状态，记录I2C活动状态，直至被清除： 0：无中断 1：有中断
7	RX_DONE	R	0x0	接收完成中断状态： 0：无中断 1：有中断
6	TX_ABRT	R	0x0	发送中止中断状态： 0：无中断 1：有中断
5	RD_REQ	R	0x0	当I2C作为从机，其他主机试图从I2C接口读取数据中断状态： 0：无中断 1：有中断
4	TX_EMPTY	R	0x0	TX FIFO到达或者低于阈值中断状态： 0：无中断 1：有中断

位	名称	属性	复位值	描述
3	TX_OVER	R	0x0	TX FIFO溢出中断状态： 0：无中断 1：有中断
2	RX_FULL	R	0x0	RX FIFO达到或超过阈值中断状态： 0：无中断 1：有中断
1	RX_OVER	R	0x0	RX FIFO溢出中断状态： 0：无中断 1：有中断
0	RX_UNDER	R	0x0	读数据溢出（即RX_FIFO为空时CPU对FIFO进行读取）中断状态： 0：无中断 1：有中断

26.5.10 I2C 中断屏蔽寄存器（I2C_INTMASK）

偏移地址：0x30

复位值：0x0000 08FF

位	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11	GEN_CALL	R/W	0x1	General call请求被接收的中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
10	START_DET	R/W	0x0	Start detect中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
9	STOP_DET	R/W	0x0	Stop detect中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
8	ACTIVITY	R/W	0x0	activity中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
7	RX_DONE	R/W	0x1	接收完成中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
6	TX_ABRT	R/W	0x1	发送中止中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
5	RD_REQ	R/W	0x1	作为从机时，另一个I2C主机读数据请求中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
4	TX_EMPTY	R/W	0x1	TX FIFO到达或者低于阈值中断屏蔽： 0：屏蔽中断 1：不屏蔽中断

位	名称	属性	复位值	描述
3	TX_OVER	R/W	0x1	TX FIFO溢出中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
2	RX_FULL	R/W	0x1	RX FIFO达到或超过阈值中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
1	RX_OVER	R/W	0x1	RX FIFO溢出中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
0	RX_UNDER	R/W	0x1	读数据溢出（即RX FIFO为空时CPU对FIFO进行读取）中断屏蔽： 0：屏蔽中断 1：不屏蔽中断

26.5.11 I2C RAW 中断寄存器（I2C_RAWISR）

偏移地址：0x34

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11	GEN_CALL	R	0x0	广播呼叫。 接收到广播呼叫地址时置位。 禁止I2C接口或者当CPU读GC寄存器时清零。I2C将接收的数据存储在接收缓冲中。
10	START_DET	R	0x0	起始条件检测： 无论I2C接口工作在主机或者从机，一旦检测到I2C接口上起始或者重复起始条件即置位该位
9	STOP_DET	R	0x0	停止条件检测： 该位状态依据CR寄存器的STOP INT的状态 当STOP INT=0，无论I2C接口工作在主机或者从机，一旦检测到I2C接口上停止条件时即置位该位。 从机模式下，无论寻址是否匹配都会产生一个STOP中断当STOPINT=1 在主机模式下（MASTER=1），该位显示I2C接口是否发生停止条件在从机模式下（MASTER=0），仅当从机地址匹配成功时产生一个STOP中断。
8	ACTIVITY	R	0x0	I2C接口激活，该位用于捕获I2C模块的活动状态置位后只能由以下四种方式清零： 禁止I2C接口； 读I2C_CLR_ACTIVITY寄存器； 读I2C_CLR寄存器； 系统复位； 一旦置位后，只能由上述方式清零，即使I2C处于空闲状态，该位也仍然保持为高直到被清零。

位	名称	属性	复位值	描述
7	RX_DONE	R	0x0	从机发送结束： 当I2C作为从发送时，如果发送一个字节的数据后主机没有响应，将会置位该位。 该情况发生在传输的最后一个字节，表示传输结束。
6	TX_ABRT	R	0x0	发送中止： 当I2C接口作为发送机时，不能发送完缓冲中的数据时置位。 注意：发送中止会将I2C接口中的接收和发送缓冲清空。发送缓冲会处于刷新状态直到读TX_ABRT寄存器。一旦该读操作执行后，发送就可以接收APB总线上的新的数据。
5	RD_REQ	R	0x0	读请求： 当I2C作为从机，其他主机试图从I2C接口读取数据时置位。 I2C接口会使总线保持等待状态（SCL=0）直到中断被处理。这就意味着I2C接口作为从机时被其他主机寻址成功且要求发送数据。处理器必须响应该中断然后写入数据到I2C_DATA_CMD寄存器中。当处理器读RD_REQ寄存器该位清零
4	TX_EMPTY	R	0x0	发送缓冲空： 该位状态取决于CR寄存器中的EMPINT状态： 当EMPINT = 0，发送缓冲为空时置位 当EMPINT = 1，发送缓冲为空且内部移位寄存器结束时置位 当发送缓冲非空时由硬件自动清零
3	TX_OVER	R	0x0	发送缓冲过载： 发送缓冲满时处理器写入新的数据导致溢出时置位
2	RX_FULL	R	0x0	接收缓冲非空： 当接收缓冲非空时置位。 当接收缓冲为空时由硬件清零
1	RX_OVER	R	0x0	接收缓冲过载： 接收缓冲满且有收到新的数据时置位。此时I2C 接口会响应，但新的数据会丢失
0	RX_UNDER	R	0x0	接收缓冲欠载： 当RX FIFO为空时处理器读DR寄存器时置位

26.5.12 I2C 接收 FIFO 阈值寄存器（I2C_RXTL）

偏移地址：0x38

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	RX_TL	R/W	0x0	接收FIFO阈值。 控制RX_FULL中断触发。

26.5.13 I2C 发送 FIFO 阈值寄存器（I2C_TXTL）

偏移地址：0x3C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	TX_TL	R/W	0x0	发送FIFO阈值。 控制TX_EMPTY中断触发

26.5.14 I2C 组合和独立中断清除寄存器（I2C_CLR）

偏移地址：0x40

复位值：0x0000_0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	CLR_INTR	R	0x0	读该寄存器将会清除所有组合中断、独立中断： 该位不清除硬件可自动清除的中断，仅清除软件可清除中断。

26.5.15 I2C 清除 RX_UNDER 中断寄存器（I2C_CLRRXUNDER）

偏移地址：0x44

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	CLR_RX_UNDER	R	0x0	读该寄存器清零RX_UNDER中断 （I2C_RAW_ISR[0]）

26.5.16 I2C 清除 RX_OVER 中断寄存器（I2C_CLRRXOVER）

偏移地址：0x48

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	CLR_RX_OVER	R	0x0	读该寄存器清零RX_OVER中断 （I2C_RAW_ISR[1]）

26.5.17 I2C 清除 TX_OVER 中断寄存器（I2C_CLRTXOVER）

偏移地址：0x4C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	CLR_TX_OVER	R	0x0	读该寄存器清零RX_OVER中断（I2C_RAW_ISR[3]）

26.5.18 I2C 清除 RD_REQ 中断寄存器（I2C_CLRRDREQ）

偏移地址：0x50

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	CLR_RD_REQ	R	0x0	读该寄存器清零RD_REQ中断（I2C_RAW_ISR[5]）

26.5.19 I2C 清除 TX_ABRT 中断寄存器（I2C_CLRTXABRT）

偏移地址：0x54

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	CLR_TX_ABRT	R	0x0	读该寄存器清零TX_ABRT中断（I2C_RAW_ISR6]） 同时也将TX FIFO从刷新/复位状态中释放，以便接收写入的数据。

26.5.20 I2C 清除 RX_DONE 中断寄存器（I2C_CLRRXDONE）

偏移地址：0x58

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	CLR_RX_DONE	R	0x0	读该寄存器清零RX_DONE中断（I2C_RAW_ISR[7]）

26.5.21 I2C 清除 ACTIVITY 中断寄存器（I2C_CLRACTIVITY）

偏移地址：0x5C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	CLR_ACTIVITY	R	0x0	如果I2C总线不活动则读该寄存器清零ACTIVITY中断（I2C_RAW_ISR[8]） 如果I2C仍然活动，那么ACTIV中断将继续置位。当I2C模块禁止或者在I2C总线不再活动时该位由硬件清零。可以通过读该寄存器得到I2C_RAW_ISR中的ACTIVITY（8）的状态。

26.5.22 I2C 清除 STOP_DET 中断寄存器（I2C_CLRSTOPDET）

偏移地址：0x60

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	CLR_STOP_DET	R	0x0	读该寄存器清零STOP中断（I2C_RAW_ISR[9]）

26.5.23 I2C 清除 START_DET 中断寄存器（I2C_CLRSTARTDET）

偏移地址：0x64

复位值：0x0000_0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	CLR_START_DET	R	0x0	读该寄存器清零START中断（I2C_RAW_ISR[10]）

26.5.24 I2C 清除 GEN_CALL 中断寄存器（I2C_CLRGENCALL）

偏移地址：0x68

复位值：0x0000_0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留

位	名称	属性	复位值	描述
0	CLR_GEN_CALL	R	0x0	读该寄存器清零GC中断（I2C_RAW_ISR[11]）

26.5.25 I2C 使能寄存器（I2C_EN）

偏移地址：0x6C

复位值：0x0000 0004

位	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16	SMBUS_CLK_RESET	R/W	0x0	SMBus主机模式发行SMBus主机时钟复位
15:4	RSV	-	-	保留
3	SDA_STUCK_RECOVERY_ENABLE	R/W	0x0	如果在TX_ABORT中断，SDA被低位，则该位用作发行SDA恢复机制。此位会自动清零
2	TX_CMD_BLOCK	R/W	0x1	在主机模式下： 0：一旦TX FIFO中的第一个数据可用，I2C自动开始传输数据 1：即使TX FIFO有数据要传输，也会阻止I2C的数据传输
1	ABORT	R/W	0x0	I2C传输中止： 0：中止没有发生或者已经结束 1：中止操作正在进行 I2C模块作为主机时置位时可以软件中止I2C的传输。一旦置位不能立即清除。置位后I2C模块控制逻辑会在完成当前传输之后产生一个STOP条件和清空发送缓冲，中止操作之后产生TX_ABRT中断。 该ABORT位会在中止操作结束后自动清零。
0	ENABLE	R/W	0x0	I2C模块使能： 0：禁止I2C模块（发送和接收缓冲保持在空状态） 1：使能I2C模块

26.5.26 I2C 状态寄存器（I2C_SR）

偏移地址：0x70

复位值：0x0000 0006

位	名称	属性	复位值	描述
31:19	RSV	-	-	保留
18	SMBUS_SLAVE_ADDR_RESOLVED	R	0x0	从机地址（I2C_SAR[6:0]）被ARP主机解析时，该位置1
17	SMBUS_SLAVE_ADDR_VALID	R	0x0	从机地址（I2C_SAR[6:0]）有效时，该位置1
16	SMBUS_QUICK_CMD	R	0x0	接收到QUICK命令的R/W位时，该位置1
15:12	RSV	-	-	保留
11	SDA_STUCK_NOT_RECOVERED	R	0x0	在恢复机制后，SDA仍处于拉低状态时，该位置1
10:7	RSV	-	-	保留
6	SLV_ACTIVITY	R	0x0	从机状态机活动状态位： 0：从机状态机处于IDLE状态，所以I2C从机部分不活动 1：从机状态机不处于IDLE状态，所以I2C从机部分活动
5	MST_ACTIVITY	R	0x0	主机状态机活动状态位： 0：主机状态机处于IDLE状态，所以I2C主机部分不活动 1：主机状态机不处于IDLE状态，所以I2C主机部分活动
4	RFF	R	0x0	接收缓冲满： 0：接收缓冲未满 1：接收缓冲满
3	RFNE	R	0x0	接收缓冲非空： 0：接收缓冲空 1：接收缓冲非空
2	TFE	R	0x1	发送缓冲空： 0：发送缓冲非空 1：发送缓冲空
1	TFNF	R	0x1	发送缓冲未满： 0：发送缓冲未满 1：发送缓冲满
0	ACTIVITY	R	0x0	I2C位活动状态： MST_ACTIVITY位与SLV_ACTIVITY位相或的结果。

26.5.27 I2C 发送缓冲深度寄存器（I2C_TXFLR）

偏移地址：0x74

复位值：0x0000 0000

位	名称	属性	复位值	描述
15:3	RSV	-	-	保留
2:0	CNT	R	0x0	发送缓冲中有效数据个数（0~7）

26.5.28 I2C 接收缓冲深度寄存器（I2C_RXFLR）

偏移地址：0x78

复位值：0x0000 0000

位	名称	属性	复位值	描述
15:3	RSV	-	-	保留
2:0	CNT	R	0x0	接收缓冲中有效数据个数（0~7）

26.5.29 I2C SDA 保持时间寄存器（I2C_SDAHOLD）

偏移地址：0x7C

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:16	RX_HOLD	R/W	0x0	当I2C作为接收时，SDA保持时间，单位为APB1时钟周期
15:0	TX_HOLD	R/W	0x1	当I2C作为发送时，SDA保持时间，单位为APB1时钟周期

26.5.30 I2C 生成从机数据 Nack 寄存器（I2C_SLVDATANACKONLY）

偏移地址：0x84

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	NACK	R/W	0x0	I2C仅作为从机接收： 0：正常产生NACK、ACK 1：在接收到数据后产生NACK

26.5.31 I2C DMA 控制寄存器（I2C_DMACR）

偏移地址：0x88

复位值：0x0000 0000

位	名称	属性	复位值	描述
15:2	RSV	-	-	保留
1	TDMAE	R/W	0x0	发送DMA使能： 0：发送DMA禁止 1：发送DMA使能

位	名称	属性	复位值	描述
0	RDMAE	R/W	0x0	接收DMA使能： 0：接收DMA禁止 1：接收DMA使能

26.5.32 I2C DMA 发送数据级别寄存器（I2C_DMATDLR）

偏移地址：0x8C

复位值：0x0000 0000

位	名称	属性	复位值	描述
15:2	RSV	-	-	保留
1:0	DMATDL	R/W	0x0	当TDMAE=1时，发送FIFO中的数据等于或小于DMATDL的值时，产生dma_tx_req

26.5.33 I2C DMA 接收数据级别寄存器（I2C_DMARDLR）

偏移地址：0x90

复位值：0x0000 0000

位	名称	属性	复位值	描述
15:2	RSV	-	-	保留
1:0	DMARDL	R/W	0x0	当RDMAE=1时，接收FIFO中的数据等于或大于DMARDL+1时，产生dma_tx_req

26.5.34 I2C SDA 建立时间寄存器（I2C_SDASETUP）

偏移地址：0x94

复位值：0x0000 0064

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	SDA_SETUP	R/W	0x64	SDA建立时间： 如果要求建议延迟时间为1000ns，APB1时钟频率为10MHz时，建议该寄存器设为11。 该寄存器最小值为2。

26.5.35 I2C 广播呼叫 ACK 寄存器（I2C_ACKGENERALCALL）

偏移地址：0x98

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	ACK_GEN_CALL	R/W	0x1	广播呼叫ACK： 1：接收到广播呼叫后响应ACK 0；接收到广播呼叫后不响应，也不产生中断

26.5.36 I2C 使能状态寄存器（I2C_ENSR）

偏移地址：0x9C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	SLV_RX_DATA_LOST	R	0x0	从机数据丢失： 1：I2C已终止传输。虽然数据字节响应了NACK，但已进入I2C数据传输阶段。 0：I2C已被禁用，从机接收未进行数据传输阶段。
1	SLV_DISABLED_WHILE_BUSY	R	0x0	Busy时禁用从机
0	IC_EN	R	0x0	I2C使能状态： 0：I2C禁用 1：I2C使能

26.5.37 I2C 尖峰抑制极限寄存器（I2C_FSSPKLEN）

偏移地址：0xA0

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	I2C_FS_SPKLEN	R/W	0x1	在进行任何I2C总线事务之前，必须先设置该寄存器，以确保稳定运行。 仅当I2C_CR[0]=0时写入有效。

26.5.38 I2C SMBus 从机时钟延长超时寄存器（I2C_SMBUSCLOCKLOWSEXT）

偏移地址：0xBC

复位值：0xFFFF FFFF

位	名称	属性	复位值	描述
31:0	SMBUS_CLK_LOW_SEXT_TIMEOUT	R/W	0xFFFFFFFF	用于检测主机模式下的从时钟延长超时，由从机延长从start到stop的时间

26.5.39 I2C SMBus 主机时钟延长超时寄存器（I2C_SMBUSCLOCKLOWMEXT）

偏移地址：0xC0

复位值：0xFFFF FFFF

位	名称	属性	复位值	描述
31:0	SMBUS_CLK_LOW_MEXT_TIMEOUT	R/W	0xFFFFFFFF	用于检测主机下从start到ack、ack到ack或ack到stop的延长SMBus时钟（SCL）超时。

26.5.40 I2C SMBus 最大总线空闲计数寄存器（I2C_SMBUSTHIGHMAXIDLECOUNT）

偏移地址：0xC4

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	SMBUS_THIGH_MAX_BUS_IDLE_CNT	R/W	0xFFFF	用于设置当主设备已动态添加到总线并且可能还未检测到SMBCLK或SMBDAT线路上的状态转换时所需的总线空闲时间段。

26.5.41 I2C SMBus 中断状态寄存器（I2C_SMBUSISR）

偏移地址：0xC8

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	R_SLV_RX_PEC_NACK	R	0x0	从机为来自从机ARP命令的PEC字节生成NACK的中断状态： 0：无中断 1：有中断
7	R_ARP_ASSGN_ADDR_CMD_DET	R	0x0	接收到Assign Address ARP中断状态： 0：无中断 1：有中断
6	R_ARP_GET_UDID_CMD_DET	R	0x0	接收到一个General或directed Get UDID ARP命令中断状态： 0：无中断 1：有中断
5	R_ARP_RST_CMD_DET	R	0x0	接收到一个General or Directed Reset ARP命令中断状态： 0：无中断 1：有中断
4	R_ARP_PREPARE_CMD_DET	R	0x0	接收到一个Prepare to ARP命令中断状态： 0：无中断 1：有中断
3	R_HOST_NOTIFY_MST_DET	R	0x0	接收到一个Host Notify命令中断状态： 0：无中断 1：有中断
2	R_QUICK_CMD_DET	R	0x0	接收到一个Quick命令中断状态： 0：无中断 1：有中断
1	R_MST_CLOCK_EXTND_TIMEOUT	R	0x0	从开始到停止的主机事务（START to ACK、ACK to ACK或ACK to STOP），在消息的每个字节中超过I2C_SMBUS_CLOCK_LOW_MEXT时间的中断状态： 0：无中断 1：有中断
0	R_SLV_CLOCK_EXTND_TIMEOUT	R	0x0	从机的事务（start到stop）超过IC_SMBUS_CLOCK_LOW_SEXT时间的中断状态： 0：无中断 1：有中断

26.5.42 I2C SMBus 中断屏蔽寄存器（I2C_SMBUSINTMASK）

偏移地址：0xCC

复位值：0x0000 01FF

位	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	M_SLV_RX_PEC_NACK	R/W	0x1	从机为来自从机ARP命令的PEC字节生成NACK的中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
7	M_ARP_ASSGN_ADDM_CMD_DET	R/W	0x1	接收到Assign Address ARP中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
6	M_ARP_GET_UDID_CMD_DET	R/W	0x1	接收到一个General或directed Get UDID ARP命令中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
5	M_ARP_RST_CMD_DET	R/W	0x1	接收到一个General or Directed Reset ARP命令中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
4	M_ARP_PREPARE_CMD_DET	R/W	0x1	接收到一个Prepare to ARP命令中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
3	M_HOST_NOTIFY_MST_DET	R/W	0x1	接收到一个Host Notify命令中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
2	M_QUICK_CMD_DET	R/W	0x1	接收到一个Quick命令中断屏蔽： 0：屏蔽中断 1：不屏蔽中断
1	M_MST_CLOCK_EXTND_TIMEOUT	R/W	0x1	从开始到停止的主机事务（START to ACK、ACK to ACK或ACK to STOP），在消息的每个字节中超过I2C_SMBUS_CLOCK_LOW_MEXT时间的中断屏蔽： 0：屏蔽中断 1：不屏蔽中断

位	名称	属性	复位值	描述
0	M_SLV_CLOCK_EXTND_TIMEOUT	R/W	0x1	从机的事务（start到stop）超过IC_SMBUS_CLOCK_LOW_SEXT时间的中断屏蔽： 0：屏蔽中断 1：不屏蔽中断

26.5.43 I2C SMBUS RAW 中断状态寄存器（I2C_SMBUSRAWISR）

偏移地址：0xD0

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	SLV_RX_PEC_NACK	R	0x0	从机为来自从机ARP命令的PEC字节生成NACK
7	ARP_ASSGN_ADDCMD_DET	R	0x0	接收到Assign Address ARP
6	ARP_GET_UDID_CMD_DET	R	0x0	接收到一个General或directed Get UDID ARP命令
5	ARP_RST_CMD_DET	R	0x0	接收到一个General or Directed Reset ARP命令中
4	ARP_PREPARE_CMD_DET	R	0x0	接收到一个Prepare to ARP命令
3	HOST_NOTIFY_MST_DET	R	0x0	接收到一个Host Notify命令
2	QUICK_CMD_DET	R	0x0	接收到一个Quick命令
1	MST_CLOCK_EXTND_TIMEOUT	R	0x0	从开始到停止的主机事务（START to ACK、ACK to ACK或ACK to STOP），在消息的每个字节中超过I2C_SMBUS_CLOCK_LOW_MEXT时间。 仅当 I2C_CR[0]=1 时有效。
0	SLV_CLOCK_EXTND_TIMEOUT	R	0x0	从机的事务（start到stop）超过IC_SMBUS_CLOCK_LOW_SEXT时间。 仅当 I2C_CR[0]=1 时有效。

26.5.44 I2C 清除 SMBus 中断状态寄存器（I2C_CLRSMBUSISR）

偏移地址：0xD4

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:9	RSV	-	-	保留

位	名称	属性	复位值	描述
8	CLR_SLV_RX_PEC_NACK	W	0x0	从机为来自从机ARP命令的PEC字节生成NACK的中断状态，写1清0。
7	CLR_ARP_ASSGN_ADDCLR_CMD_DET	W	0x0	接收到Assign Address ARP中断状态，写1清0。
6	CLR_ARP_GET_UDID_CMD_DET	W	0x0	接收到一个General或directed Get UDID ARP命令中断状态，写1清0。
5	CLR_ARP_RST_CMD_DET	W	0x0	接收到一个General or Directed Reset ARP命令中断状态，写1清0。
4	CLR_ARP_PREPARE_CMD_DET	W	0x0	接收到一个Prepare to ARP命令中断状态，写1清0。
3	CLR_HOST_NOTIFY_MST_DET	W	0x0	接收到一个Host Notify命令中断状态，写1清0。
2	CLR_QUICK_CMD_DET	W	0x0	接收到一个Quick命令中断状态，写1清0。
1	CLR_MST_CLOCK_EXTND_TIMEOUT	W	0x0	从开始到停止的主机事务（START to ACK、ACK to ACK或ACK to STOP），在消息的每个字节中超过I2C_SMBUS_CLOCK_LOW_MEXT时间的中断状态，写1清0。
0	CLR_SLV_CLOCK_EXTND_TIMEOUT	W	0x0	从机的事务（start到stop）超过IC_SMBUS_CLOCK_LOW_SEXT时间的中断状态，写1清0。

26.5.45 I2C SMBUS ARP UDID LSB 寄存器（I2C_SMBUSUDIDLSB）

偏移地址：0xDC

复位值：0xFFFF FFFF

位	名称	属性	复位值	描述
31:0	IC_SMBUS_ARP_UDID_LSB	R/W	0xFFFFFFFF	在地址解析协议下，存储LSB 32位从机唯一的设备标识符。

26.6 使用流程

I2C接口可以以下述4种方式中的一种运行：

- 从机发送器模式
- 从机接收器模式

- 主机发送器模式
- 主机接收器模式

注：I2C接口模块只能工作在主机模式或者从机模式，但不能同时工作在两种模式下。因此要确保寄存器I2C_CR[6]和I2C_CR[0]不能分别设置为0和1（或者分别为1和0）。

26.6.1 主从机初始化配置

主机初始化：

1. 配置 GPIO 复用，RCM 模块中使能 I2Cx 时钟。
2. 配置 I2C_EN[0]为 0，禁止 I2C 模块。
3. 配置 I2C_CR[6]为 1，禁止从机。
4. 配置 I2C_CR[4]，设置主机地址格式为 7bit/10bit。
5. 配置 I2C_CR[2:1]，设置 I2C 工作速率。
6. 为保证通信时序，需要在 I2C 传输前配置好 CNT 的值。标准模式下，配置 I2C_SSSCLHCNT[15:0]和 I2C_SSSCLLCNT[15:0]。快速模式下，配置 I2C_FSSCLHCNT[15:0]和 I2C_FSSCLLCNT[15:0]。
7. 配置 I2C_RXTL[7:0]和 I2C_TXTL[7:0]，设置接收和发送 FIFO 阈值，此处设置为 0。
8. 配置 I2C_CR[0]为 1，使能主机。
9. 配置 I2C_EN[0]为 1，使能 I2C 模块。

从机初始化：

1. 配置 GPIO 复用，RCM 模块中使能 I2Cx 时钟。
2. 配置 I2C_EN[0]为 0，禁止 I2C 模块。
3. 配置 I2C_CR[3]，设置从机地址格式为 7bit/10bit。
4. 配置 I2C_SAR[9:0]，根据设置的从机地址格式写入从机地址。
5. 配置 I2C_CR[6]为 0，使能从机。
6. 配置 I2C_CR[7]为 1，设置为当地址匹配时才产生 STOP_DET 中断。
7. 配置 I2C_CR[0]为 0，禁止主机。
8. 配置 I2C_RXTL[7:0]和 I2C_TXTL[7:0]，设置接收和发送 FIFO 阈值，此处设置为 0。
9. 配置 I2C_EN[0]为 1，使能 I2C 模块。

26.6.2 主机发送功能

1. 配置 I2C_TAR[9:0]，设置需要操作的从机目的地址。
2. 配置 I2C_DATACMD[8]为 0，设置写操作。同时往 I2C_DATACMD[7:0]写入需要发送的数据。
3. 每次操作完 I2C_DATACMD 寄存器后，读 I2C_RAWISR[4]，直到读取状态值为 1，表示发送缓存的数据发送完成（I2C_RAWISR[4]置位条件由 I2C_TXTL[7:0]设定）。

4. 若是发送单个数据，需要在步骤 2 上同时配置 I2C_DATACMD[9]为 1，在当前字节发送之后产生一个 STOP 信号。

5. 若是发送多个数据，则重复步骤 2、3。当发送最后一个数据时，配置 I2C_DATACMD[8]为 0，设置写操作。I2C_DATACMD[9]为 1，在当前字节发送之后产生一个 STOP 信号。同时往 I2C_DATACMD[7:0]写入需要发送的数据。

26.6.3 主机接收功能

1. 配置 I2C_TAR[9:0]，设置需要操作的从机目的地址。

2. 配置 I2C_DATACMD[8]为 1，设置读操作。

3. 读 I2C_RAWISR[2]，直到读取状态值为 1，表示接收缓存接收到 1 个数据（I2C_RAWISR[2]置位条件由 I2C_RXTL[7:0]设定）。此时读取 I2C_DATACMD[7:0]的数据。

4. 若是接收单个数据，需要在步骤 2 上同时配置 I2C_DATACMD[9]为 1，在读取当前字节数据之后产生一个 STOP 信号。

5. 若是发送多个数据，则重复步骤 2、3。当接收最后一个数据时，配置 I2C_DATACMD[8]为 1，设置读操作。I2C_DATACMD[9]为 1，在读取当前字节数据之后产生一个 STOP 信号。

26.6.4 从机发送功能

1. 读 I2C_RAWISR[5]，直到读取状态值为 1，表示有主机试图从该从机读取数据。

2. 配置 I2C_DATACMD[8]为 0，设置写操作。同时往 I2C_DATACMD[7:0]写入需要发送的数据。

3. 读 I2C_RAWISR[4]，直到读取状态值为 1，表示发送缓存的数据发送完成（I2C_RAWISR[4]置位条件由 I2C_TXTL[7:0]设定）。

4. 若为多数据发送，则重复步骤 2、3。

5. 发送完所有数据后，需要 I2C_CLRRDREQ[0]清除 RD_REQ 状态。

26.6.5 从机接收功能

1. 读 I2C_RAWISR[2]，直到读取状态值为 1，表示接收缓存接收到 1 个数据（I2C_RAWISR[2]置位条件由 I2C_RXTL[7:0]设定）。

2. 此时读取 I2C_DATACMD[7:0]的数据。

3. 若为多个数据读取，则重复步骤 1、2。

27 串口音频接口（I2S）

27.1 概述

集成电路内置音频总线 (Inter-IC Sound Bus, I²S) 是由飞利浦 (Philips) 开发的一种标准化的通信接口，用于许多基于（超）大规模集成电路的系统，尤其是在许多数字立体声音频系统中。

主要特性：

- 支持主机和从机模式
- 支持单工发送、单工接收、双工收发
- 支持飞利浦 (Philips)、左对齐、右对齐、PCM 长/短帧标准
- 声道长度可选 16 位或 32 位
- 音频数据长度可选 8 位、16 位、24 位或 32 位
- 对于非 PCM 标准，支持双声道数据和单声道数据
- 对于双声道音频数据，可选左声道先传输或右声道先传输
- 对于单声道音频数据，可选左声道或右声道传输
- 对于非 PCM 标准，WS 的极性可选
- 对于非 PCM 标准，SD 和 WS 的切换的时刻可选 SCK 的上升或下降沿
  对于 PCM 标准，SD 和 WS 的切换的时刻固定为 SCK 的上升沿
- 内置 8 字深的 TXFIFO 和 RXFIFO（1 字 = 32 位），各自可以储存 16 条 16 位的数据或 8 条 32 位的数据
- 主机模式下需要 PLL 生成的 MCLK，其频率是音频采样率 (Fs) 的 256 倍，音频采样率通常为 8/11.025/16/22.05/24/32/44.1/48/96/192 kHz
- 如果使能了中断，则下列情况会触发中断：
  - TXFIFO 有足够空间
  - RXFIFO 有足够的数据
  - TXFIFO 欠载
  - RXFIFO 溢出
- 支持 DMA 操作

27.2 管脚说明

表 27-1：I2S 管脚说明

功能管脚	复用管脚	方向	功能描述
I2S0_MCLK	PC6	Input/Output	音频主时钟，可以从内部audio PLL产生，或者从外部输入
I2S0_WS	PB9,PB12	Input/Output	字选择
I2S0_CK	PB10,PB13	Input/Output	位时钟
I2S0_SD	PC3,PB15	Input/Output	音频数据输入输出
I2S0_EXTSD	PC2,PB14	Input	全双工音频数据输入
I2S1_MCLK	PC7	Input/Output	音频主时钟，可以从内部audio PLL产生，或者从外部输入
I2S1_WS	PA4,PA15	Input/Output	字选择
I2S1_CK	PB3,PC10	Input/Output	位时钟
I2S1_SD	PB5,PC12	Input/Output	音频数据输入输出
I2S1_EXTSD	PB4,PC11	Input	全双工音频数据输入

27.3 功能描述

27.3.1 发送器 FIFO

发送器 FIFO (TXFIFO) 是一个 8 字深的 FIFO，用于保存 CPU 或 DMA 写入“写数据寄存器”（I2S_WR）的数据。

27.3.2 接收器 FIFO

接收器 FIFO (RXFIFO) 是一个 8 字深的 FIFO，用于保存从外部 I2S 模块接收的数据。CPU 或 DMA 从“读取数据寄存器”（I2S_RD）读取 RXFIFO 中的数据。

27.3.3 16 位声道的控制

这个模块用于控制16位长的声道。它通过端口将音频数据从TXFIFO发送到外部I2S设备，并通过端口从外部I2S设备接收音频数据并将其存储在RXFIFO中。

在I2S协议中，SCK（serial clock，串行时钟，也称为位时钟）用于指导移位寄存器在每个SCK周期中将数据移一位。WS信号（word select，字选择，也称为左右时钟或LRCLK）用于指示正在传输哪个声道数据。

在主机模式下，SCK由来自PLL模块的MCLK（master clock，主时钟）分频产生，这个控制模块对同步后的SCK的边沿进行计数，并生成WS信号。

在从机模式下，SCK和WS均来自外部I²S主设备，这个控制模块对同步后的SCK的边沿进行计数，并根据同步后的WS的边沿来调整SCK的计数。

当I2S_GCR[9]置1时，传输操作开始。当“发送移位寄存器”为空时，它将从TXFIFO中获取数据，然后将数据移至串行端口。

当I2S_GCR[8]置1时，接收操作开始。来自串行端口的输入数据被放入“接收移位寄存器”中。当数据的所有位都移入时，数据将被传输到RXFIFO。

27.3.4 32 位声道的控制

这个模块用于控制32位长的声道。它与16位通道控制逻辑的功能相同，并且它们的逻辑也类似。主要区别在于SCK频率和一些控制信号，这些信号控制来自TXFIFO或去往RXFIFO的双声道数据的传输时序。

27.3.5 端口同步

在主机模式下，此模块对来自 PLL 的 MCLK 分频生成 SCK。为了使采样率保持为 MCLK 频率的 1/256，对于 16 位长的通道，SCK 频率是 MCLK 的 1/8；而对于 32 位长的通道，SCK 的频率是 MCLK 的 1/4。对于 16 位长的通道，PCLK 的频率必须大于 MCLK 的 3/4 倍。对于 32 位长的通道，PCLK 的频率必须大于 MCLK 的 3/2 倍。在从机模式下，SCK 直接来自外部 I²S 设备。PCLK 的频率必须大于 SCK 的三倍。

下表总结了不同模式下对 PCLK 频率的要求。

表 27-2：I2S 不同模式下对 PCLK 频率的要求

主机模式		
16 位长的通道	$f_{SCK} = 1/8 \times f_{MCLK}$	$f_{PCLK} > 3/4 \times f_{MCLK}$
32 位长的通道	$f_{SCK} = 1/4 \times f_{MCLK}$	$f_{PCLK} > 3/2 \times f_{MCLK}$
从机模式		
$f_{PCLK} > 3 \times f_{SCK}$		

27.3.6 接口时序

I2S (Inter-IC sound)是飞利浦 (Philips)开发的总线标准。在飞利浦标准中，SD（serial audio data，串行音频数据）比 WS（word select，字选择或左右时钟）延迟一个 SCK（serial clock，串行时钟或位时钟）周期。如果 SD 比声道短，则在声道中 SD 之后填充零。默认情况下，SD 在 SCK 的下降沿切换，并在 SCK 的上升沿采样。

最高位对齐（左对齐）标准和最低位对齐（右对齐）标准后来被开发出来。在这些标准中，SD 和 WS 是同步传输的。如果 SD 比声道短，那么在最高位对齐标准中，会在声道中 SD 之后填

充零；在最低位对齐标准中，会在声道中 SD 之前填充零。默认情况下，SD 在 SCK 的下降沿切换，并在 SCK 的上升沿采样。

在 PCM 标准中，SD 比 WS 延迟一个 SCK 周期。PCM 标准中有两种帧同步格式，即短帧同步和长帧同步。在短帧同步格式中，当正在传输的 SD 的最低位（包括 SD 的有效位之后填充的零）时，WS 会在 1 个 SCK 周期内保持高电平。在长帧同步格式中，当 SD 的高 13 位正在传输时，WS 在 13 个 SCK 周期内保持高电平。请注意，WS 并不代表字选择（左右时钟），因为在 PCM 标准中不区分左声道或右声道。如果 SD 比声道短，则在声道中 SD 之后填充零。特别是，SD 在 SCK 的上升沿切换，并在 SCK 的下降沿采样，这是不可配置的。

图 27-1 展示了 16 位声道的 I2S 接口时序，图 27-2 展示了 32 位声道的 I2S 接口时序。请注意，“sck_cnt”是不可访问的内部计数器，不是 I2S 端口上的信号。

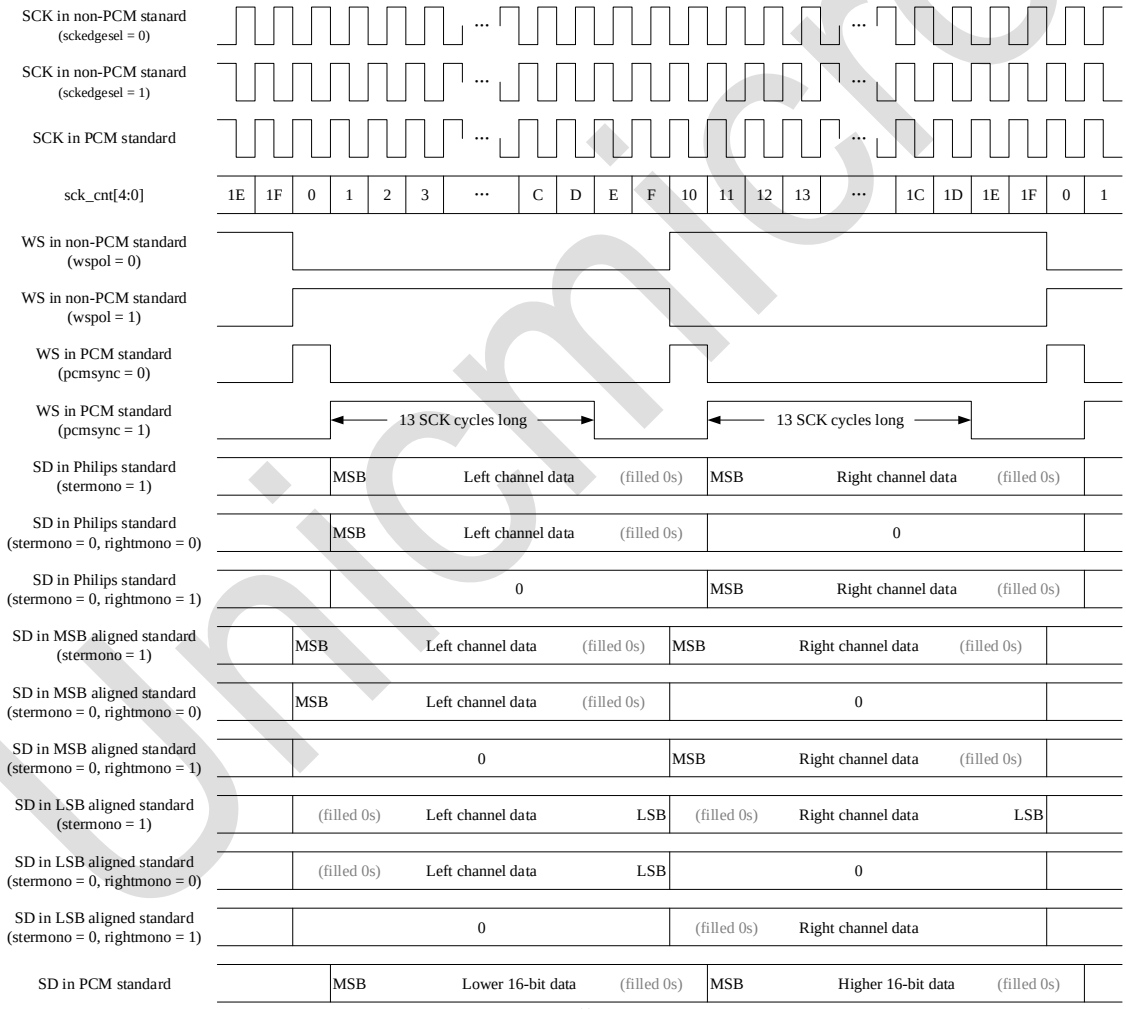


图 27-1：16 位声道的 I2S 信号波形

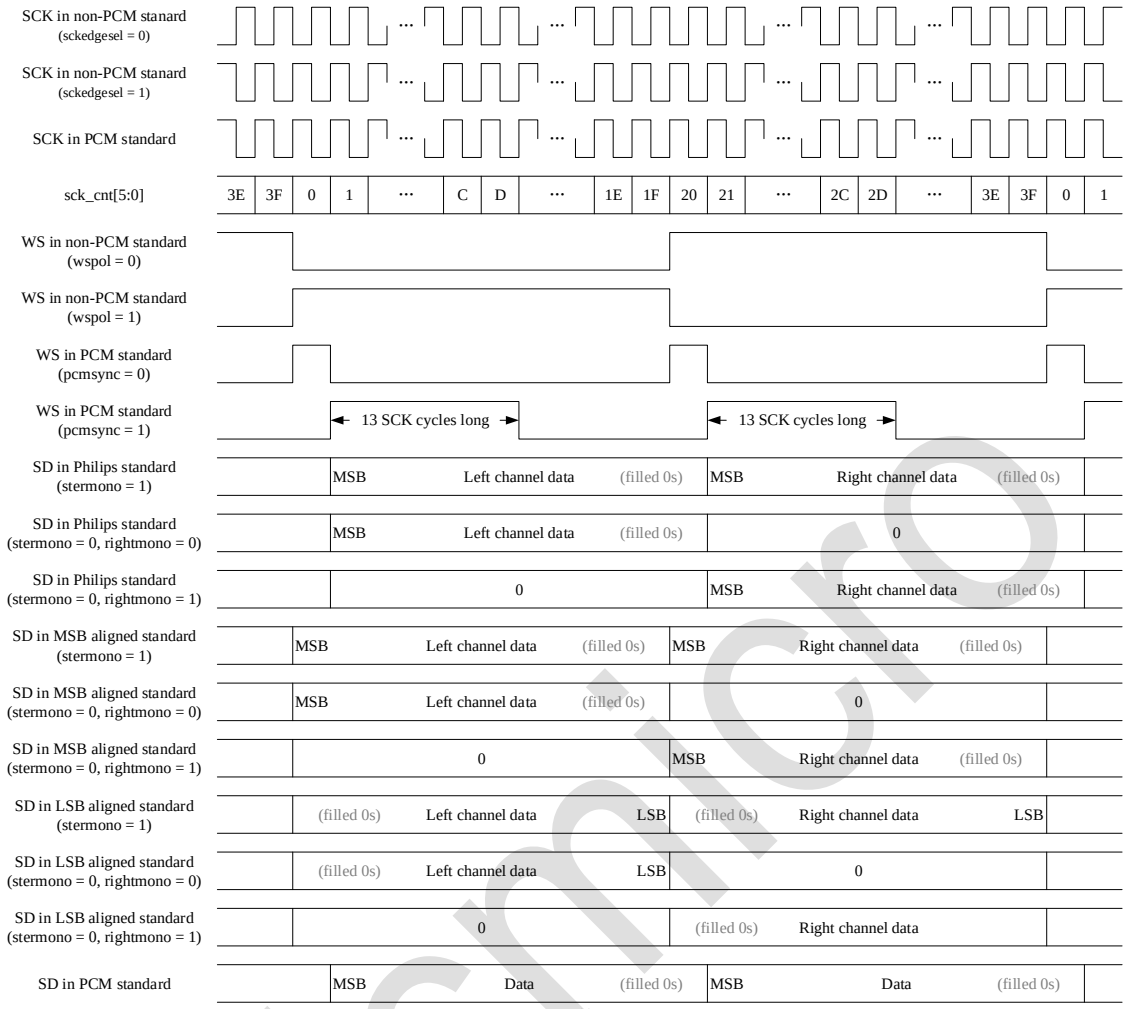


图 27-2：32 位声道的 I2S 信号波形

27.4 寄存器描述

I2S0 寄存器基地址：0x4700_D000

I2S1 寄存器基地址：0x4700_E000

寄存器列表如下：

表 27-3：I2S 寄存器列表

偏移地址	名称	描述
0x00	I2S_WR	I2S 写入数据寄存器
0x04	I2S_RD	I2S 读取数据寄存器
0x08	I2S_CSR	I2S 当前状态寄存器
0x0C	I2S_GCR	I2S 全局控制寄存器
0x10	I2S_DFR	I2S 数据格式寄存器
0x14	I2S_ISR	I2S 中断状态寄存器
0x18	I2S_IER	I2S 中断使能寄存器
0x1C	I2S_ICR	I2S 中断清除寄存器

27.4.1 I2S 写入数据寄存器（I2S_WR）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	I2S_WR	W/R	0x0	写入 TXFIFO 的音频数据： ● 对于 16 位长的声道中的数据：一个字的数据包含两条 16 位单声道数据（低 16 位先传输）或一对立体声数据（低 16 位先传输，默认左声道数据先传输，但是可以通过 I2S_DFR[8]配置哪个数据先传输。） ● 对于 32 位长的声道中的数据：一个字的数据包含一个 32 位单声道或双声道数据。

注：TXFIFO 中的数据必须保持最高有效位（MSB）对齐。

1. 对于 16 位长的声道中的数据：

   如果数据长 8 位，就必须存储在[31:24]位或[15:8]位中，而[23:16]位和[7:0]位则被忽略。

2. 对于 32 位长的声道中的数据：
   - 如果数据长 8 位，就必须存储在[31:24]位中，而[23:0]位则被忽略。
   - 如果数据长 16 位，就必须存储在[31:16]位中，而[15:0]位则被忽略。
   - 如果数据长 24 位，就必须存储在[31:8]位中，而[7:0]位则被忽略。

27.4.2 I2S 读取数据寄存器（I2S_RD）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	I2S_RD	R	0x0	从 RXFIFO 读取的音频数据： 对于 16 位长的声道中的数据：一个字的数据包含两条 16 位单声道数据（低 16 位先传输）或一对立体声数据（低 16 位先传输，默认左声道数据先传输，但是可以通过 I2S_DFR[8]“Right_first”配置哪个数据先传输。） 对于 32 位长的声道中的数据：一个字的数据包含一个 32 位单声道或双声道数据。

注：RXFIFO 中的数据总是保持最高有效位（MSB）对齐。

1. 对于 16 位长的声道中的数据：

   如果数据长 8 位，则会存储在[31:24]位或[15:8]位中，而[23:16]位和[7:0]位则会保持为 0。

2. 对于 32 位长的声道中的数据：
   - 如果数据长 8 位，则会存储在[31:24]位中，而[23:0]位则会保持为 0。
   - 如果数据长 16 位，则会存储在[31:16]位中，而[15:0]位则会保持为 0。
   - 如果数据长 24 位，则会存储在[31:8]位中，而[7:0]位则则会保持为 0。

27.4.3 I2S 当前状态寄存器（I2S_CSR）

偏移地址：0x08

复位值：0x0000 0105

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:12	RXFIFO_LEVEL	R	0x0	RXFIFO 数据量： 这些位指示 RXFIFO 中有多少个字的数据。范围是 0x0 – 0x8。
11	RSV	-	-	保留
10	RXFIFO_TRIG	R	0x0	RXFIFO 数据可用状态： 当 RXFIFO 接收到足够数据（达到由 I2S_GCR[1:0]规定的数据量）时，此位置 1。 0：RXFIFO 没有足够数据 1：RXFIFO 有足够数据
9	RXFIFO_FULL	R	0x0	RXFIFO 满状态： 0：RXFIFO 未满 1：RXFIFO 满
8	RXFIFO_EMPTY	R	0x1	RXFIFO 空状态： 0：RXFIFO 非空 1：RXFIFO 空
7:4	TXFIFO_LEVEL	R	0x0	TXFIFO 数据量： 这些位指示 TXFIFO 中有多少个字的数据。范围是 0x0 – 0x8。
3	RSV	-	-	保留
2	TXFIFO_TRIG	R	0x1	TXFIFO 空位充足状态： 当 TXFIFO 有充足空位（达到由 I2S_GCR[5:4]规定的空位数量）时，此位置 1。 0：TXFIFO 没有充足空位 1：TXFIFO 有充足空位
1	TXFIFO_FULL	R	0x0	TXFIFO 满状态： 0：TXFIFO 未满 1：TXFIFO 满
0	TXFIFO_EMPTY	R	0x1	TXFIFO 空状态： 0：TXFIFO 非空 1：TXFIFO 空

27.4.4 I2S 全局控制寄存器（I2S_GCR）

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:15	RSV	-	-	保留
15	SD_DIR	W/R	0x0	SD 管脚方向选择： 0：输入 1：输出 全双工下，SD 管脚必须设置为输出
14	I2S_EN	W/R	0x0	I2S 使能： 0：停用 I2S 1：启用 I2S
13	DMA_MODE	W/R	0x0	DMA 访问使能： 0：CPU 读写 TXFIFO 与 RXFIFO 1：DMA 读写 TXFIFO 与 RXFIFO
12	INTEN	W/R	0x0	I2S 中断使能： 0：禁止 I2S 中断 1：允许 I2S 中断
11	MST_MODE	W/R	0x0	主从机模式选择： 在主机模式下 I2S 输出 SCK 与 WS， 在从机模式下 I2S 输入 SCK 与 WS。 0：从机模式 1：主机模式
10	FILLDATASEL	W/R	0x0	当 TXFIFO 欠载时填充数据选择： 0：发送空包数据（全 0） 1：发送上一条数据
9	TXEN	W/R	0x0	发送控制逻辑和 TXFIFO 使能： 0：禁止发送 1：使能发送
8	RXEN	W/R	0x0	接收控制逻辑和 RXFIFO 使能： 0：禁止接收 1：使能接收
7:5	RSV	-	-	保留
4	TXFIFO_WTMK	W/R	0x0	TXFIFO 水印（触发数量）选择： 0：TXFIFO 未满 （TXFIFO 中有 1 个字或更多空位） 1：TXFIFO 中有 4 个字或更少数据 （TXFIFO 中有 4 个字或更多空位）
3:1	RSV	-	-	保留
0	RXFIFO_WTMK	W/R	0x0	RXFIFO 水印（触发数量）选择： 0：RXFIFO 非空 1：RXFIFO 中有 4 个字或更多数据

27.4.5 I2S 数据格式寄存器（I2S_DFR）

偏移地址：0x10

复位值：0x0000 0024

位	名称	属性	复位值	描述
31:11	RSV	-	-	保留

位	名称	属性	复位值	描述
10	PCM_SYNC	W/R	0x0	PCM 帧同步格式选择： 此位仅在 PCM 标准中有效。 0：短帧同步（WS 在 1 个 SCK 周期内保持高电平） 1：长帧同步（WS 在 13 个 SCK 周期内保持高电平）
9	RIGHT_MONO	W/R	0x0	左/右单声道选择： 此位仅对单声道数据有效。 0：选择左声道数据发送或接收 1：选择右声道数据发送或接收
8	RIGHT_FIRST	W/R	0x0	双声道音频优先声道选择： 此位用于选择优先发送或接收哪个声道的数据，仅对双声道数据有效。 0：左声道数据优先 1：右声道数据优先
7	SCK_EDGESEL	W/R	0x0	数据切换所在 SCK 边沿选择： 此位对 PCM 标准无效。注意：在 PCM 标准中，SD 和 WS 在 SCK 的上升沿切换。 1：SD 和 WS 在 SCK 的上升沿切换 0：SD 和 WS 在 SCK 的下降沿切换（推荐）
6	WSPOL	W/R	0x0	WS 极性选择： 此位对于 PCM 标准无效，因为在 PCM 标准中不关心左声道或右声道，WS 用于帧同步。 0：WS 低表示左声道（用于在 Philips 标准） 1：WS 高表示左声道（用于 MSB 对齐和 LSB 对齐标准）
5	STER_MONO	W/R	0x1	双声道或单声道数据选择： 此位对于 PCM 标准无效，因为在 PCM 标准中不关心左声道或右声道。实际上，PCM 标准的控制逻辑以双声道数据的方式设计。 0：单声道数据 1：立体声数据
4	CHANLEN32	W/R	0x0	声道长度选择： 0：16 位 1：32 位
3:2	DATA_LENSEL	R	0x1	声音数据长度选择： 16 位长的声道支持 8 位和 16 位数据，而 32 位长的声道支持全部四种数据。 0x0：8 位 0x1：16 位（默认） 0x2：24 位 0x3：32 位
1:0	I2S_STD	W/R	0x0	I2S 标准选择： 0x0：飞利浦标准 0x1：最高位对齐/左对齐标准 0x2：最低位对齐/右对齐标准 0x3：PCM 标准

27.4.6 I2S 中断状态寄存器（I2S_ISR）

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	FRAME_ERR_INTF	R	0x0	帧错误中断标志： 在从机模式下当 WS 在的意外时刻切换时，此位置 1。帧错误可能会导致 SD 的错误和遗漏，因为如果发生帧错误，SCK 计数器将自动纠正，但是 SCK 计数器的这种突然跳转可能会导致许多控制信号错误。 0：没有发生帧错误 1：发生帧错误
3	UNDERRUN_INTF	R	0x0	TXFIFO 欠载错误中断标志： 当发送使能后 TXFIFO 为空时，此位置 1。发送的数据可以是 0x0 或先前发送的数据。 0：未发生 TXFIFO 欠载错误 1：发生 TXFIFO 欠载错误
2	RXOERR_INTF	R	0x0	RXFIFO 溢出错误中断标志： 当试图将新到的数据字写入满的 RXFIFO 时，此位置 1。在这种情况下，新到的数据将丢失，并且接收过程将继续。 0：没有发生 RXFIFO 溢出错误 1：发生 RXFIFO 溢出错误
1	RX_INTF	R	0x0	RXFIFO 数据足够中断标志： 当 RXFIFO 接收到足够数据（达到由 I2S_GCR[1:0]规定的数据量）时，此位置 1。 0：RXFIFO 数据不足 1：RXFIFO 有足够的数据
0	TX_INTF	R	0x0	TXFIFO 空间充足中断标志： 当 TXFIFO 有充足空位（达到由 I2S_GCR[5:4]规定的空位数量）时，此位置 1。 0：TXFIFO 没有充足空位 1：TXFIFO 有充足空位

注：此中断状态寄存器是原始中断状态寄存器，不能被中断使能寄存器屏蔽，但可以被中断清除寄存器清除。

27.4.7 I2S 中断使能寄存器（I2S_IER）

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	FRAME_ERR_INTEN	W/R	0x0	帧错误中断使能： 0：中断未使能 1：中断使能
3	UNDERRUN_INTEN	W/R	0x0	TXFIFO 欠载错误中断使能： 0：中断未使能 1：中断使能
2	RXOERR_INTEN	W/R	0x0	RXFIFO 溢出错误中断使能： 0：中断未使能 1：中断使能
1	RX_INTEN	W/R	0x0	RXFIFO 数据足够中断使能： 0：中断未使能 1：中断使能
0	TX_INTEN	W/R	0x0	TXFIFO 空间充足中断使能： 0：中断未使能 1：中断使能

27.4.8 I2S 中断清除寄存器（I2S_ICR）

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	FRAME_ERR_INTCLR	W	0x0	帧错误中断清除： 0：不清除中断 1：清除中断
3	UNDERRUN_INTCLR	W	0x0	TXFIFO 欠载错误中断清除： 0：不清除中断 1：清除中断
2	RXOERR_INTCLR	W	0x0	RXFIFO 溢出错误中断清除： 0：不清除中断 1：清除中断
1	RX_INTCLR	W	0x0	RXFIFO 数据足够中断清除： 0：不清除中断 1：清除中断
0	TX_INTCLR	W	0x0	RXFIFO 空间充足中断清除： 0：不清除中断 1：清除中断

27.5 使用流程

1. 配置 I2S_DFR 寄存器来选择 I2S 标准并设置声道长度和数据格式等。
2. 配置 I2S_IER 寄存器来使能中断。
3. 配置 I2S_GCR 寄存器来使能 I2S 和中断，但暂不使能发送 I2S_GCR[9]或接收 I2S_GCR[8]，

同时要选择主从机模式、CPU 或 DMA 访问模式及 FIFO 水印等。

4. 如果选择 DMA 访问模式，请正确配置 DMA 控制器。
5. 设置 I2S_GCR[15]选择 SD 管脚作为输入或者输出（全双工传输时必须作为输出）
6. 设置 I2S_GCR[9]/I2S_GCR[8]来开始数据传输。
7. 如果选择 CPU 访问模式，每当 TXFIFO 有充足空间时给 I2S_WR 寄存器写待发送数据，且/或每当 RXFIFO 有足够数据时从 I2S_RD 寄存器读取收到的数据。TXFIFO 和 RXFIFO 的状态可以通过 I2S_CSR 寄存器和 I2S_ISR 寄存器检查，也可以利用中断请求获取。
8. 如果选择 DMA 访问模式，DMA 控制器会通过 DMA 握手信号给 I2S_WR 寄存器写入待发送数据，且/或从 I2S_RD 寄存器读取收到的数据。TXFIFO 和 RXFIFO 的状态可以随时通过 I2S_CSR 寄存器和 I2S_ISR 寄存器检查。
9. 不论选择 CPU 访问模式还是 DMA 访问模式，当传输结束后，通过清除 I2S_GCR[14]来停用 I2S。

28 通用异步收发器接口（UART）

28.1 概述

Universal Asynchronous Receiver/Transmitter 通用异步串口收发器（以下简称 UART）是使用非常广泛的串行通信接口，支持全双工通信。通用异步串口收发器是把存储器或处理器中并行传输的数据串行的发送到外设的 UART 接收端，或接收 UART 外设的串行数据并转换为并行数据提供给处理器。UART 支持与外部接口设备的串行通信。

28.2 主要特性

- 16 字节的硬件 FIFO
- 波特率支持整数和小数分频
- 支持 CTS，RTS 流控制
- 支持错误起始位侦测
- 支持帧中断检测
- 支持断路检测
- 支持数据位宽（5~9bit），停止位个数（1bit、1.5bit 和 2bit）设置
- 支持对数据进行固定校验、奇偶校验或无校验
- 支持 IrDA 1.0 协议，波特率范围在 9.6k~115.2k
- 支持 DMA 操作

28.3 管脚说明

表 28-1：UART 管脚说明

功能管脚	复用管脚	方向	功能描述
UART0_TX	PA9，PB6	Output	发送数据
UART0_RX	PA10，PB7	Input	接收数据
UART0_CTS	PA11	Input	硬件流控模式发送使能信号
UART0_RTS	PA12	Output	硬件流控模式发送请求信号
UART1_TX	PA2，PD5	Output	发送数据
UART1_RX	PA3，PD6	Input	接收数据
UART1_CTS	PA0，PD3	Input	硬件流控模式发送使能信号
UART1_RTS	PA1，PD4	Output	硬件流控模式发送请求信号
UART2_TX	PB10，PC10，PD8	Output	发送数据
UART2_RX	PB11，PC11，PD9	Input	接收数据

功能管脚	复用管脚	方向	功能描述
UART2_CTS	PB13，PD11	Input	硬件流控模式发送使能信号
UART2_RTS	PB14，PD12	Output	硬件流控模式发送请求信号
UART3_TX	PA0，PC10	Output	发送数据
UART3_RX	PA1，PC11	Input	接收数据
UART3_CTS	PA2	Input	硬件流控模式发送使能信号
UART3_RTS	PA3	Output	硬件流控模式发送请求信号
UART4_TX	PC12	Output	发送数据
UART4_RX	PD2	Input	接收数据
UART5_TX	PC6	Output	发送数据
UART5_RX	PC7	Input	接收数据
UART5_CTS	PB12	Input	硬件流控模式发送使能信号
UART5_RTS	PB13	Output	硬件流控模式发送请求信号

28.4 功能描述

根据用户需要，可配置任意波特率进行 UART 发送或者接收数据。

28.4.1 可配置任意波特率

UART 支持配置任意波特率进行数据的收发，主要由 DLL 寄存器、DLH 寄存器和 DLF 寄存器进行配置，计算过程如下：

Baud rate 为需要的波特率，fclk 为时钟频率，A 为$\frac{fclk}{16*Baud\ rate}$的整数部分，B 为$\frac{fclk}{16*Baud\ rate}$的小数部分，则 A 由 DLH、DLL 寄存器进行配置，B 由 DLF 寄存器进行配置。

28.4.2 UART 发送模式

UART 发送时可把并行数据转换成串行数据进行发送。当使用 UART 进行数据发送时，可配置数据大小、波特率，奇偶校验是否使能以及奇偶校验类型。

28.4.3 UART 接收模式

当使用 UART 进行数据接收时，可以配置任意的波特率进行数据接收，并且可以使用奇偶校验来检查数据在传送过程中是否出现了错误。

28.4.4 IrDA 模式

UART 兼容 IrDA1.0 物理层协议，IrDA 传输最大波特率可达 115.2K Baud。其数据格式固定为 1 个起始位+8 个数据位+1 个停止位，无奇偶校验位。

28.5 寄存器描述

UART0 寄存器基地址：0x4700_F000

UART1 寄存器基地址：0x40B0_3000

UART2 寄存器基地址：0x4600_3000

UART3 寄存器基地址：0x4600_4000

UART4 寄存器基地址：0x4600_5000

UART5 寄存器基地址：0x4600_6000

表 28-2：UART 寄存器列表

偏置地址	名称	描述
0x00	UART_RBR	接收缓冲寄存器
0x00	UART_THR	发送缓冲寄存器
0x00	UART_DLL	波特率分频低位寄存器
0x04	UART_DLH	波特率分频高位寄存器
0x04	UART_IER	中断使能寄存器
0x08	UART_IIR	中断状态寄存器
0x08	UART_FCR	FIFO控制寄存器
0x0C	UART_LCR	LINE控制寄存器
0x10	UART_MCR	流控制寄存器
0x14	UART_LSR	LINE状态寄存器
0x18	UART_MSR	流状态寄存器
0x20	UART_LPDLL	低功耗波特率分频低位寄存器
0x24	UART_LPDLH	低功耗波特率分频高位寄存器
0x7C	UART_USR	状态寄存器
0x80	UART_TFL	发送FIFO数据个数寄存器
0x84	UART_RFL	接收FIFO数据个数寄存器
0xC0	UART_DLF	小数分频寄存器
0xC4	UART_RAR	接收地址匹配寄存器
0xC8	UART_TAR	发送地址匹配寄存器
0xCC	UART_LCRE	LINE控制扩展寄存器

以下各节详细介绍寄存器。

28.5.1 接收缓冲寄存器（UART_RBR）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:9	RSV	-	-	保留

位	名称	属性	复位值	描述
8:0	RBR	R	0x0	接收数据寄存器，可保存UART模式或SIR模式下接收的数据。此字段为接收FIFO入口，仅当UART_LCR的DLAB位为0时，此字段才可以访问。

28.5.2 发送缓冲寄存器（UART_THR）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8:0	THR	W	0x0	发送数据寄存器，可保存UART模式或SIR模式下要发送的数据。此字段为发送FIFO入口，仅当UART_LCR的DLAB位为0时，此字段才可以访问。

28.5.3 波特率分频低位寄存器（UART_DLL）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	DLL	R/W	0x0	波特率配置寄存器低位。仅当UART_LCR的DLAB位为1时，此字段才可以访问。 波特率整数部分计算公式： baud rate = fclk / (16 * {DLH, DLL}) 注意：如果有小数分频，需要先配置好DLF，再配置DLL。

28.5.4 波特率分频高位寄存器（UART_DLH）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	DLH	R/W	0x0	波特率配置寄存器高位。仅当UART_LCR的DLAB位为1时，此字段才可以访问。 波特率整数部分计算公式： baud rate = fclk / (16 * {DLH, DLL}) 注意：如果有小数分频，需要先配置好DLF，再配置DLL。

28.5.5 中断使能寄存器（UART_IER）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	PTIME	R/W	0x0	THRE中断使能，仅当UART_LCR的DLAB位为0时，此字段才可以访问： 1：使能THRE中断 0：禁止THRE中断
6:3	RSV	-	-	保留
2	ELSI	R/W	0x0	LINE中断使能，仅当UART_LCR的DLAB位为0时，此字段才可以访问： 1：使能LINE中断 0：禁止LINE中断
1	ETBEI	R/W	0x0	发送FIFO空中断使能，仅当UART_LCR的DLAB位为0时，此字段才可以访问： 1：使能发送FIFO空中断 0：禁止发送FIFO空中断
0	ERBFI	R/W	0x0	接收数据中断使能，仅当UART_LCR的DLAB位为0时，此字段才可以访问： 1：使能接收FIFO非空中断 0：禁止接收FIFO非空中断

28.5.6 中断状态寄存器（UART_IIR）

偏移地址：0x08

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:6	FIFOSE	R	0x0	FIFO使能标志： 11：FIFO使能 00：FIFO禁止
5:4	RSV	-	-	保留
3:0	IID	R	0x01	状态ID： 0000：保留 0001：无中断 0010：发送FIFO空 0100：接收FIFO非空 0110：LINE中断状态 0111：Busy状态 1100：TimeOut状态，当使能FIFO和接收FIFO非空中断后，如果在接收FIFO中存在至少1个数据，在4个UART帧内，CPU如果未读FIFO，则此字段会进入TimeOut中断状态。 其它：保留

注：此寄存器中断状态会被读清除

28.5.7 FIFO 控制寄存器（UART_FCR）

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:6	RT	W	0x0	接收FIFO非空中断设置，当FIFO中数据大于或等于此设置对应的FIFO状态时，接收FIFO非空中断置位： 00：1帧数据 01：4帧数据 10：8帧数据 11：14帧数据
5:4	TET	W	0x0	发送FIFO空中断设置，当FIFO中数据少于或等于此设置对应的FIFO状态时，发送FIFO空中断置位： 00：FIFO空 01：2帧数据 10：4帧数据 11：8帧数据
3	RSV	-	-	保留
2	XFIFOR	W	0x0	发送FIFO复位位：此位固定为0： 1：复位发送FIFO 0：不复位发送FIFO
1	RFIFOR	W	0x0	接收FIFO复位位：此位固定为0： 1：复位接收FIFO 0：不复位接收FIFO
0	FIFOE	W	0x0	FIFO使能位： 1：使能FIFO 0：禁止FIFO 改变此位的值将会同时复位接收和发送FIFO。

28.5.8 LINE 控制寄存器（UART_LCR）

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	DLAB	R/W	0x0	UART_DLL和UART_DLH寄存器访问设置位，仅当UART处于空闲状态可写： 1：UART_DLL可以通过偏移地址0x0访问，UART_DLH可以通过偏移地址0x4访问； 0：UART_RBR/UART_THR可以通过偏移地址0x0访问，UART_IER可以通过偏移地址0x4访问。

位	名称	属性	复位值	描述
6	BC	R/W	0x0	Break控制位，用来产生断开条件传递到接收设备。UART模式下会将TX保持低电平，SIR模式下会将TX连续发送正脉冲。 1：开启break 0：关掉break 注意：当还有数据没有发送完成时，需等到发送完成才会触发。
5	SEPS	R/W	0x0	奇偶校验位强制设置位，仅当UART处于空闲状态时可写： 1：PEN为1，当EPS为1时，发送和接收检查奇偶校验位为0；PEN为1，当EPS为0时，发送和接收检查奇偶校验位为1；当PEN为0时，发送和接收均无奇偶校验位。 0：奇偶校验位强制设置功能禁止。
4	EPS	R/W	0x0	奇偶校验选择位，仅当UART处于空闲状态时可写： 1：偶校验 0：奇校验
3	PEN	R/W	0x0	奇偶校验位使能设置，仅当UART处于空闲状态时可写： 1：奇偶校验位使能 0：奇偶校验位禁止
2	STOP	R/W	0x0	STOP比特长度设置，仅当UART处于空闲状态时可写： 1：当DLS=00时1.5比特STOP位；否则2比特STOP位 0：1比特STOP位。
1:0	DLS	R/W	0	UART帧数据长度设置位，仅当UART处于空闲状态时可写： 00：5比特 01：6比特 10：7比特 11：8比特

28.5.9 流控制寄存器（UART_MCR）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	DMAE	R/W	0	DMA传输模式选择位： 0：DMA mode disabled 1：DMA mode enabled 注：要使用和外部的DMA控制器通讯，此位必须设置上。
6	SIRE	R/W	0	SIR（IrDA）模式选择位： 0：IrDA SIR mode disabled 1：IrDA SIR mode enabled 此模式下，一帧数据固定由1bit起始位、8bit数据位，1bit停止位构成，无法通过LCR寄存器修改（使用此模式时，不可以在此位使能之前配置LCR的低四位）。
5	AFCE	R/W	0	1：CTS/RTS自动流控制使能 0：CTS/RTS自动流控制禁止

位	名称	属性	复位值	描述
4	LB	R/W	0	在UART模式下，可以实现串口TX输出波形发送到同一个串口RX。 在SIR模式下，可以实现将串口TX的输出波形反相后回到同一个串口RX。 1：开启LOOP Mode 0：关闭LOOP Mode 注：TX发到RX的已在内部进行操作，实际上TX是没有波形发出。
3:2	RSV	-	-	保留
1	RTS	R/W	0	RTS接口软件控制位： 1：RTS请求输出有效 0：RTS请求输出无效 注：当开启LB模式（MCR[4]=1），RTS将会失效。
0	RSV	-	-	保留

28.5.10 LINE 状态寄存器（UART_LSR）

偏移地址：0x14

复位值：0x0000 0060

位	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	ADDR_RCVD	R	0x0	9比特数据模式下，接收到的数据是地址还是数据的标志： 1：接收的数据为地址信息； 0：接收的数据为数据信息； 读取此寄存器，清0。
7	RFE	R	0x0	接收FIFO错误标志： 1：接收FIFO中数据至少有一个有奇偶校验错误或者UART帧格式错误； 0：接收FIFO中数据没有错误； 当接收FIFO中出错的数据是下一个将要读取的数据，且接收FIFO中其它的数据没有错误时，读取此寄存器，清0。
6	TEMT	R	0x01	发送完成标志： 1：发送完成，发送FIFO为空，且移位寄存器为空 0：发送未完成
5	THRE	R	0x01	当Ptime和FIFO同时使能时，发送FIFO空标志： 1：发送FIFO满 0：发送FIFO非满 否则 发送FIFO空标志： 1：发送FIFO空 0：发送FIFO非空

位	名称	属性	复位值	描述
4	BI	R	0x0	Break中断标志位。 1：接收到break信号 0：未接收到break信号 读此寄存器或RBR寄存器将会清零
3	FE	R	0x0	帧格式出错标志： 1：帧格式错误 0：帧格式未出错 读此寄存器或RBR寄存器将会清零
2	PE	R	0x0	奇偶校验出错标志： 1：奇偶校验错误 0：奇偶校验未出错 读此寄存器或RBR寄存器将会清零
1	OE	R	0x0	接收FIFO溢出标志： 1：接收FIFO溢出 0：接收FIFO非溢出 读此寄存器清0。
0	DR	R	0x0	与USR寄存器的RFNE功能重复（当FIFO=ENABLE）接收FIFO非空标志： 1：接收FIFO非空 0：接收FIFO空

28.5.11 流状态寄存器（UART_MSR)

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	CTS	R	0x0	CTS标志位： 1：有CTS请求 0：无CTS请求
3:0	RSV	-	-	保留

28.5.12 低功耗波特率分频低位寄存器（UART_LPDLL）

偏移地址：0x20

复位值：0x0000_0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	LPDLL	R/W	0x0	低功耗下，波特率分频寄存器低位。仅当UART_LCR的DLAB位为1时，此字段才可以访问。此位用来配置SIR模式接收检测。 计算公式为：Low baud rate = fclk / (16 * {LPDLH,LPDLL})

28.5.13 低功耗波特率分频高位寄存器 (UART_LPDLH)

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	LPDLH	R/W	0x0	低功耗下，波特率分频寄存器高位。仅当UART_LCR的DLAB位为1时，此字段才可以访问。此位用来配置SIR模式接收检测。计算公式为：Low baud rate = fclk / (16 * { LPDLH,LPDLL })

28.5.14 状态寄存器（UART_USR）

偏移地址：0x7C

复位值：0x0000 0006

位	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4	RFF	R	0x0	接收FIFO满标志： 1：接收FIFO满 0：接收FIFO非满
3	RFNE	R	0x0	接收FIFO非空标志： 1：接收FIFO非空 0：接收FIFO空
2	TFE	R	0x01	发送FIFO空标志： 1：发送FIFO空 0：发送FIFO非空
1	TFNF	R	0x01	发送FIFO非满标志： 1：发送FIFO非满 0：发送FIFO满
0	BUSY	R	0x0	1：UART正在传输 0：UART处于空闲状态

28.5.15 发送 FIFO 数据个数寄存器（UART_TFL）

偏移地址：0x80

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4:0	TFL	R	0x0	发送FIFO中数据个数位。

28.5.16 接收 FIFO 数据个数寄存器（UART_RFL）

偏移地址：0x84

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4:0	RFL	R	0x0	接收FIFO中数据个数位。

28.5.17 小数分频寄存器 (UART_DLF)

偏移地址：0xC0

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:6	RSV	-	-	保留
5:0	DLF	R/W	0x0	小数分频寄存器。 小数部分波特率为DLF/64。 计算公式为：(PCLK%(BAUDRATE*64)) / BAUDRATE。

28.5.18 接收地址匹配寄存器 (UART_RAR)

偏移地址：0xC4

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	RAR	R/W	0x0	接收地址匹配寄存器。

28.5.19 发送地址匹配寄存器（UART_TAR）

偏移地址：0xC8

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	TAR	R/W	0x0	发送地址匹配寄存器。

28.5.20 LINE 控制扩展寄存器（UART_LCRE）

偏移地址：0xCC

复位值：0x00000000

位	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3	TRANSMIT_MODE	R/W	0x0	9 比特发送模式选择，仅在 UART 空闲状态可写： 1：发送 FIFO 为 9 比特，地址和数据的指示标志来自发送 FIFO 0：发送 FIFO 为 8 比特，发送的地址由 SEND_ADDR 位和 UART_TAR 来决定，发送的数据来自发送 FIFO
2	SEND_ADDR	R/W	0x0	发送地址匹配使能位： 1：当 9 比特模式下，UART 帧中的第 9 比特为 1（地址指示），UART 将发送 UART_TAR 寄存器中的数据； 0：当 9 比特模式下，UART 帧中的第 9 比特为 0（数据指示），UART 将发送 FIFO 中的数据。 此位将会发送完后自动清 0
1	ADDR_MATCH	R/W	0x0	接收数据地址匹配模式使能位，仅在 UART 处于空闲状态可写： 1：地址匹配模式使能 0：地址匹配模式禁止 此位仅在 DLS_E 为 1 时有效。
0	DLS_E	R/W	0x0	仅在 UART 处于空闲状态可写： 1：9 比特模式使能 0：帧格式由 DLS 位来决定

28.6 使用流程

28.6.1 UART 初始化

1. 开启对应引脚 GPIO 的时钟，把引脚配置成 UART_TX、UART_RX 复用功能。
2. 配置系统配置寄存器的串口模块时钟。
3. 配置 UART_DLF 寄存器设置小数分频。
4. 设置 UART_LCR[7]为 1，配置 UART_DLH、UART_DLL 寄存器设置串口波特率。
5. 配置 UART_LCR 寄存器设置串口奇偶校验、停止位长度、数据帧长度、Dlab 设为 0。
6. 配置 UART_FCR 寄存器使能 FIFO。
7. 配置 UART_IER 寄存器使能对应的串口中断。

28.6.2 UART 发送流程

1. 发送数据前软件可以配置波特率参数，奇偶校验类型、数据帧格式。
2. 设置 UART_LCR[7]为 0。

3. 写入第一个数据到 UART_THR 寄存器。
4. 查询发送完成标志 UART_LSR[6]，如果 UART_LSR[6]=1 表示当前数据发送完成。
5. 可以继续写入下一个字节到 UART_THR。

28.6.3 UART 接收流程

1. 发送数据前软件可以配置波特率参数，奇偶校验类型、数据帧格式。
2. 接收数据，查询 UART_USR 标志位或者等待中断，如果 UART_USR[3] = 1 时，接收 FIFO 非空，则读取 UART_RBR 中的数据；读取数据后相应的标志位自动清除。
3. 接收错误处理：等待中断或者查询 UART_LSR 寄存器标志位，判断错误类型，执行相应的错误处理，处理完之后软件清除标志位。
4. 继续接收数据。

29 低功耗串行接口（LPUART）

29.1 概述

LPUART 是一个低功耗 UART 接口，工作在 32KHz 时钟下，最高支持 9600 波特率的数据接收。

29.2 主要特性

- 异步数据收发
- 标准UART帧格式
  - 1bit起始位
  - 7或8bit数据
  - 奇校验、偶校验或无校验位
  - 1或2bit停止位
- 使用32768Hz XTL时钟或者32KHz RCL时钟工作，支持波特率300~9600Hz
- 可编程数据极性
- 中断标志
  - 接收Buffer满标志
  - 接收Buffer溢出标志
  - 接收帧格式错误标志
  - 接收校验位错误标志
  - START检测标志
  - 数据匹配标志
  - 发送完成标志
- 低功耗模式下唤醒芯片
  - RXD下降沿唤醒
  - 起始位检测唤醒
  - 1字节接收完成唤醒
  - 1字节数据匹配唤醒

29.3 系统框图

29.3.1 结构框图

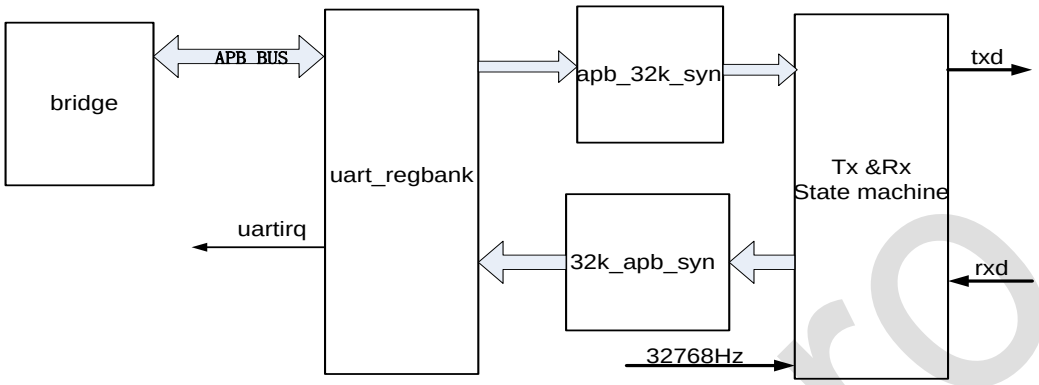


图 29-1：LPUART 结构框图

29.3.2 接口时序

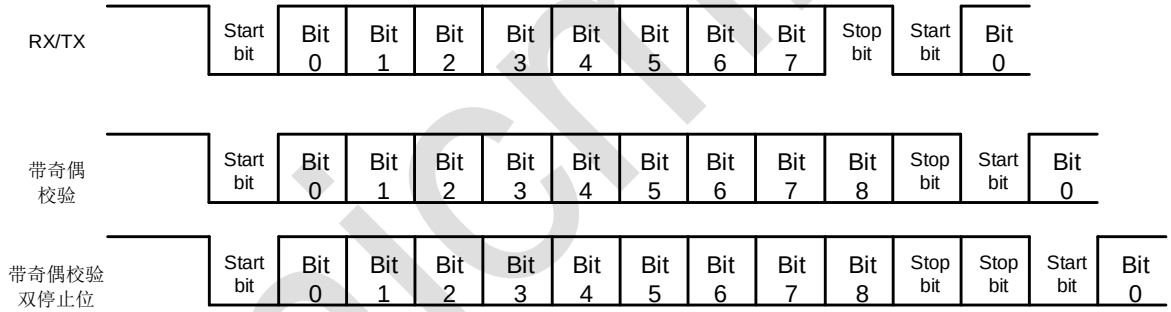


图 29-2：LPUART 接口时序图

29.3.3 接收时序

由于 LPUART 工作时钟不是波特率的整数倍，采用固定分频系数的话会引入累积误差，在接收时采用 3、4 分频交替进行接收，确保在每个 bit 的中间位置采样，每个 bit 采样一次。每个 bit 采用 3 分频还是 4 分频，则由 MCTL 寄存器控制。例如：

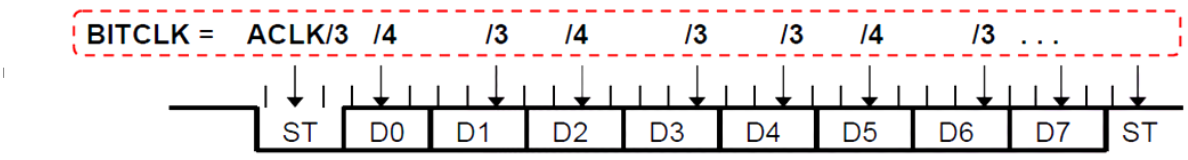


图 29-3：LPUART 接收时序图

29.3.4 发送时序

类似于 LPUART 接收，LPUART 工作时钟不是波特率的整数倍，采用固定分频系数同样会引入累积误差，在发送时也采用了 3、4 分频交替进行发送，每个 bit 采用 3 分频还是 4 分频，则由 MCTL 寄存器控制。例如：

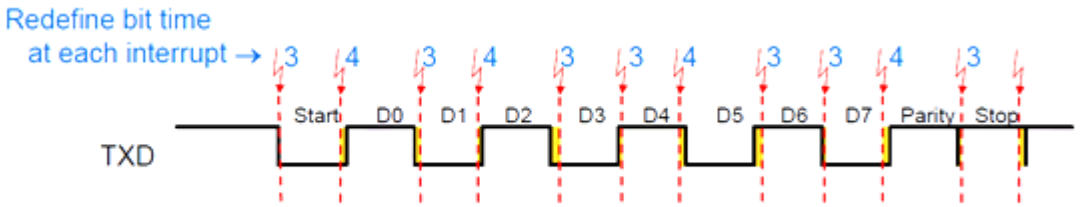


图 29-4：LPUART 发送时序图

29.4 管脚说明

表 29-1：LPUART 管脚说明

功能管脚	复用管脚	方向	功能描述
LPUART_TX	PA4,PB11	Output	发送数据
LPUART_RX	PC2,PB10	Input	接收数据

注：待机模式下，可以通过 PC2 管脚唤醒系统。PB10 待机模式下不能使用。

29.5 功能描述

29.5.1 发送模式

LPUART发送时可把并行数据转换成串行数据进行发送。发送的数据帧格式可配置为1bit起始位、7bit或8bit数据、奇校验、偶校验或无校验位、1bit或2bit停止位，传输波特率大小在300~9600。

29.5.2 接收模式

LPUART 可把串行数据转换成并行数据进行接收，支持在低功耗模式下，通过 LPUART 接收事件唤醒芯片，可在 Sleep/STOP 模式下的接收数据。传输波特率大小在 300~9600Hz。

29.6 寄存器描述

LPUART 寄存器基地址：0x40B0_7000

表 29-2：LPUART 寄存器列表

偏移地址	名称	描述
0x00	LPUART_RXD	接收数据寄存器
0x04	LPUART_TXD	发送数据寄存器
0x08	LPUART_STA	状态寄存器
0x0C	LPUART_CON	控制寄存器
0x10	LPUART_IF	中断标志寄存器
0x14	LPUART_BAUD	波特率寄存器
0x18	LPUART_EN	接收使能寄存器
0x1C	LPUART_COMPARE	数据匹配寄存器
0x20	LPUART_MODU	波特率调制控制寄存器

以下各节详细介绍寄存器。

29.6.1 接收数据寄存器（LPUART_RXD）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	LPURXD	R	0x0	接收数据寄存器

29.6.2 发送数据寄存器（LPUART_TXD）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	LPUTXD	W	0x0	发送数据寄存器

29.6.3 状态标志寄存器（LPUART_STA）

偏移地址：0x08

复位值：0x0000 00C0

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	TC	R	0x1	发送完成标志，当一帧数据发送完成且发送 buffer 为空时置位。 1:发送完成 0：发送未完成

位	名称	属性	复位值	描述
6	TXE	R	0x1	发送 buffer 空标志，硬件置位，软件向发送 buffer 写数据时自动清零 1：buffer 空 0：buffer 非空
5	START	R/W	0x0	起始位检测标志，写 1 清零
4	PERR	R/W	0x0	校验位错误，写 1 清零
3	FERR	R/W	0x0	帧格式错误，写 1 清零
2	RXOV	R/W	0x0	接收缓冲溢出，写 1 清零
1	RXF	R	0x0	接收缓冲满，读 LPUDATA 寄存器清零
0	MATCH	R/W	0x0	数据匹配标志，表示接收缓冲区内的数据与比较寄存器相同，写 1 清零

29.6.4 控制寄存器（LPUART_CON）

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12	TXPOL	R/W	0x0	数据发送极性： 0：非取反 1：取反
11	TCIE	R/W	0x0	发送完成中断使能： 0：禁止发送完成中断 1：允许发送完成中断
10	TXIE	R/W	0x0	发送 buffer 空中断使能： 0：禁止发送 buffer 空中断 1：允许发送 buffer 空中断
9	RSV	R	0x0	保留
8	PAREN	R/W	0x0	校验位使能： 0：数据帧无奇偶校验位 1：数据帧有奇偶校验位
7	PTYP	R/W	0x0	校验位类型： 0：偶校验 1：奇校验
6	SL	R/W	0x0	停止位长度： 0：1bit 1：2bits
5	DL	R/W	0x0	数据长度： 0：8bits 1：7bits
4	RXPOL	R/W	0x0	接收极性： 0：非取反 1：取反

位	名称	属性	复位值	描述
3	ERRIE	R/W	0x0	错误中断使能： 0：禁止接收错误中断 1：允许接收错误中断
2	RXIE	R/W	0x0	接收中断使能： 0：禁止接收中断 1：允许接收中断
1:0	RXEV	R/W	0x0	接收中断事件配置，用于控制何种事件下向 CPU 提供接收中断： 00：START 位检测唤醒 01：1byte 数据接收完成 10：接收数据匹配成功 11：下降沿检测唤醒

29.6.5 中断标志寄存器（LPUART_IF）

偏移地址：0x10

复位值：0x0000 0004

位	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3	TC_IF	R/W1C	0x0	发送完成中断标志： 1：发送完一帧数据后中断产生 0：无中断产生
2	TX_IF	R/W1C	0x1	发送 buffer 空中断标志： 1：发送 buffer 空后中断产生 0：无中断产生
1	RXNEG_IF	R/W1C	0x0	RXD 下降沿中断标志： 1：中断产生 0：无中断产生
0	RX_IF	R/W1C	0x0	接收完成中断标志： 1：接收完一帧数据后中断产生 0：无中断产生

29.6.6 波特率寄存器(LPUART_BAUD)

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2:0	BAUD	R/W	0x0	波特率控制（bps）： 000：9600 001：4800 010：2400 011：1200

位	名称	属性	复位值	描述
				100：600 101/110/111：300

29.6.7 发送接收使能寄存器（LPUART_EN）

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	TXEN	R/W	0x0	发送使能： 0：关闭 LPUART 发送 1：打开 LPUART 发送 CPU 写 1 使能后，要反复读取此寄存器，直到读到 1 为止才能进行后面的操作
0	RXEN	R/W	0x0	接收使能： 0：关闭 LPUART 接收 1：打开 LPUART 接收 CPU 写 1 使能后，要反复读取此寄存器，直到读到 1 为止才能进行后面的操作。

29.6.8 数据匹配寄存器（LPUART_COMPARE）

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	COMPARE	R/W	0x0	比较数据，如果 Rxev=10，当接收缓冲区内的数据与 Compare 相同时，触发接收完成中断

29.6.9 调制控制寄存器（LPUART_MODU）

偏移地址：0x20

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	MCTL	R/W	0x0	LPUART 每个 bit 的调制控制信号

29.7 使用流程

29.7.1 初始化流程

1. 配置对应 GPIO 引脚复用为 LPUART_TX、LPUART_RX。
2. 配置 PMU 中功能时钟控制寄存器 PMU_FCCR[0]使能 LPUART 时钟。
3. 配置波特率寄存器(LPUBAUD) 设置波特率。
4. 根据波特率配置调制控制寄存器(MCTL) 选择合适的调制参数。
5. 配置控制寄存器(LPUCON) 设置数据帧格式、奇偶校验、停止位长度。

29.7.2 接收流程

1. 配置发送接收使能寄存器 LPUART_EN[0]=1，打开接收使能。
2. 配置控制寄存器 LPUART_CON[1:0]，使能接收中断。
3. 等待接收中断触发，然后读取接收数据寄存器(LPURXD)的数据。
4. 向中断标志寄存器 LPUART_IF[0]写 1 清除接收完成中断标志。
5. 继续接收数据重复步骤 2、3、4。

29.7.3 发送流程

1. 配置发送接收使能寄存器 LPUART_EN[1]=1，打开发送使能。
2. 将要发送的数据写入发送数据寄存器(LPUTXD)。
3. 等待状态标志寄存器 LPUART_STA[6]置 1。
4. 继续发送数据重复步骤2、3。

29.7.4 调制寄存器建议配置

软件需要根据通信波特率的不同合理配置调制控制寄存器 MCTL，建议的配置参数表如下：

表 29-3：LPUART_MODU 的 MCTL 配置参数表

Baud	MCTL											
	Bit0（Start）	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7	Bit 8	Bit 9	Bit 10	Bit 11
9600	0	1	0	0	1	0	1	0	1	0	0	1
4800	1	1	0	1	1	1	1	1	0	1	1	1
2400	1	1	0	1	1	0	1	1	0	1	1	0
1200	0	1	0	0	1	0	0	1	0	0	1	0
600	0	1	1	0	1	0	1	1	0	1	1	0
300	0	1	0	0	0	0	1	0	0	0	0	1

以上参数表假设 LPUART 工作时钟为准确的 32768Hz，如果使用 RCLP 工作，则会引入额外的误差，可能需要微调波特率调制方案来获得更好的通信效果。

30 通用同步异步收发器（USART）

30.1 概述

通用同步异步收发器（USART）提供了一个完整的全双工通用同步异步串行链路。为支持最大的标准，数据帧格式可编程范围广（数据长度、奇偶校验、停止位数）。这个接收器实现奇偶校验错误、帧错误和溢出错误检测。接收器超时允许处理可变长度的帧，发送器时间保护便于和慢速远程设备进行通信。多点通信也可通过接收和传输中的地址位处理来支持。

30.2 主要特性

- 可编程波特率发生器
- 5至9位全双工同步或异步串行通信
  - 异步模式下 1、1.5 或 2 个停止位，或同步模式下 1 或 2 个停止位模式
  - 奇偶校验生成和错误检测
  - 帧错误检测、溢出错误检测
  - MSB 或 LSB 优先
  - 可选的断路符生成和检测
  - 8 倍或 16 倍过采样接收机频率
  - 可选的硬件握手 RTS - CTS
  - 接收器超时和发送器时间保护
  - 可选的带有地址生成和检测的多点模式
- IrDA调制解调
  - 通信速率高达 115.2 Kbps
- SPI模式
  - 主机或从机
  - 串行时钟的相位和极性可编程
  - SPI 串行时钟（SCK）频率高达内部时钟频率 MCK/6
- LIN模式
  - 符合 LIN1.3 和 LIN2.0 规范
  - 主机或从机
  - 处理多达 256 个数据字节的帧
  - 响应数据长度可以通过标识符进行配置或自动定义
  - 从机节点配置中的自同步

- 自动处理和验证“Synch Break”和“Synch Field”
- 即使“Synch Break”与数据字节部分叠加，也会被检测到
- 自动标识符奇偶校验计算、发送和验证
- 可以禁用奇偶校验发送和验证
- 自动校验和的计算、发送和验证
- 可以禁用校验和的发送和验证
- 支持“经典”和“增强”两种校验和类型
- 完整的 LIN 错误检查和报告

- 支持DMA操作

30.3 管脚说明

表 30-1：管脚说明

功能管脚	复用管脚	方向	功能描述
USART6_CTS	PA0,PC0	Input	硬件流控模式发送使能信号
USART6_RTS	PA1,PC1	Output	硬件流控模式发送请求信号
USART6_CK	PA4,PC4	Input/Output	时钟信号
USART6_TX	PA2,PC2	Input/Output	数据发送
USART6_RX	PA3,PC3	Input/Output	数据接收
USART7_CTS	PB6,PC5	Input	硬件流控模式发送使能信号
USART7_RTS	PB7,PC9	Output	硬件流控模式发送请求信号
USART7_CK	PB3,PC8	Input/Output	时钟信号
USART7_TX	PB4,PC6	Input/Output	数据发送
USART7_RX	PB5,PC7	Input/Output	数据接收

30.4 功能描述

30.4.1 波特率发生器

波特率发生器的时钟源可以通过设定模式寄存器（USART_MR）中的USCLKS字段来选择：

- 主机时钟
- 主机时钟的分频，除非取决于产品，但通常设置为8
- 外部时钟，在SCK引脚上有效

波特率发生器基于16分频，用波特率发生器寄存器（USART_BRGR）的CD字段进行编程。如果将0写入CD，则波特率发生器不生成任何时钟。如果将1写入CD，则会旁路掉分频器。

如果选择了外部SCK时钟，则所提供信号的低电平和高电平的持续时间必须大于主时钟（MCK）周期。在USART模式下，SCK上提供的信号必须至少比MCK低4.53倍，或6倍在SPI模式下。

30.4.1.1 异步模式

在异步模式下，时钟首先由USART_BRGR中的CD位进行分频。产生的时钟作为采样时钟提供给接收器，然后根据USART_MR寄存器的OVER位进行16或8分频。

波特率按以下公式计算：

$$\text{波特率} = \frac{\text{选择的时钟源}}{（8（2-Over）CD）}$$

30.4.1.2 异步模式的小数波特率

先前定义的波特率发生器受以下限制：输出频率仅以参考频率的整数倍变化。解决这个问题的一种方法是集成一个具有高分辨率的小数N时钟发生器，小数部分由USART_BRGR的FP位编程。如果FP不为0，小数部分有效。此功能仅在USART普通模式下有效。小数波特率按以下公式进行计算：

$$\text{波特率} = \frac{\text{选择的时钟源}}{（8（2-Over）（CD+\frac{FP}{8}））}$$

30.4.1.3 同步模式或 SPI 模式

在同步模式下，时钟由USART_BRGR寄存器的CD位设定：

$$\text{波特率} = \frac{\text{选择的时钟源}}{CD}$$

在同步模式中，如果选择外部时钟（USCLKS = 3）,时钟由USART SCK引脚信号提供；则不需分频，USART_BRGR中的值无效。外部时钟频率必须至少小于系统频率的1/3。同步模式的主机(USCLKS = 0或1, CLK0=1)，接收器SCK的最大频率限制为MCK/3。

无论是选择外部时钟或者是内部时钟分频器（MCK/DIV）。如果用户要保证SCK引脚上信号占空比为50:50，则必须设置CD位域的值为偶数。如果选择了内部时钟MCK，即使CD域的值为奇数，波特率发生器也会确保SCK引脚上50:50的占空比。

30.4.2 接收器和发送器控制

复位后，接收器被禁用。用户必须通过设置控制寄存器USART_CR[4]来启用接收器。然而，接收寄存器可以在接收器时钟使能之前进行编程。

复位后，发送器被禁用。用户必须通过设置控制寄存器USART_CR[6]来启用发送器。然而，发送寄存器可以在发送器时钟使能之前进行编程。

接收器和发射器可以一起使能，也可以单独使能。

在任何时候，软件可以设定USART_CR[2]和USART_CR[3]对USART的接收器或发射器执行复位。软件复位清除状态标志并重置内部状态机，但用户配置的寄存器维持不变。不管是在接收还是

发送，通信都会立即停止。

用户还可以通过设定USART_CR[5]和USART_CR[6]来单独禁用接收或发送。如果在字符接收期间接收器被禁用，则USART等待当前字符接收结束后停止接收。如果在发送期间被禁用，USART将等待当前字符和存储USART_THR寄存器中的字符传输结束。如果设定了时间保护，则可以正常处理。

30.4.3 同步和异步模式

30.4.3.1 发送器操作

发送器在同步和异步操作模式（同步=0或同步=1）下执行相同的操作。一个起始位、最多9个数据位、一个可选奇偶校验位和最多两个停止位，在串行时钟的下降沿，依次在TXD引脚上移出。数据位的数目由USART_MR寄存器的CHRL域及MODE9位决定。如果设置MODE9位，不管CHRL位域如何设置，数据位均为9位。奇偶校验位由USART_MR中的PAR域设置。可配置为奇检验、偶检验、空间检验、标志检验或无校验位。MSBF域配置先发送的位；若写入1，将先发送最高位；若写入0，将先发送最低位。停止位数目由NBSTOP域确定。异步模式下支持1.5个停止位。

- **字符发送**

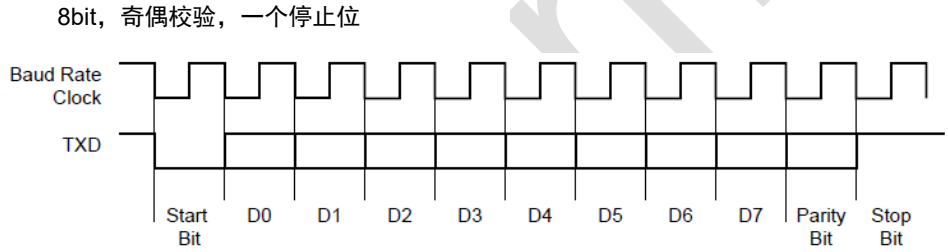


图 30-1：USART 字符发送

通过将字符写到发送保持寄存器（USART_THR）中来发送字符。发送器对应在通道状态寄存器 (USART_CSR)中有2个状态位：TXRDY (发送就绪) 表示USART_THR空，TXEMPTY表示所有写入USART_THR中的字符都已处理完。当当前字符已处理完，最后写入USART_THR的字符被送入发送器移位寄存器中时，则USART_THR变空，于是TXRDY 升高。

发送器被禁用后，TXRDY与TXEMPTY位均为低。当TXRDY为低时，往USART_THR 中写入字符无效，且写入的数据丢失。

- **发送器状态**

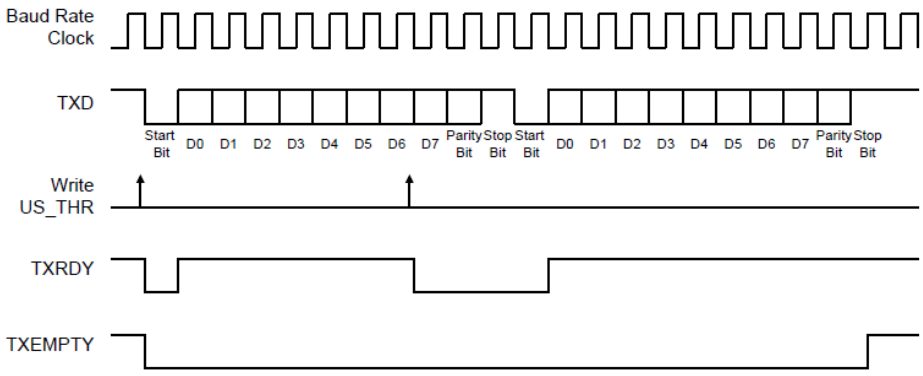


图 30-2：USART 发送器状态

30.4.3.2 曼彻斯特编码

当使用曼彻斯特编码时，通过USART发送的字符采用双相曼彻斯特编码II格式。为了使用这种模式，要将USART_MR寄存器的MAN位域置1。根据极性配置，一个逻辑电平（0或1）被编码成信号0到1或1到0的转换进行发送。因此，电平转换总是发生在每个位时间的中点。虽然它比原来的NRZ信号占用更多带宽（2倍），但是由于预期输入必须在半个位时钟时产生变化，它能实现更多的差错控制。一个曼彻斯特编码序列例子：如果采用默认极性的编码器，字节0xB1或10110001将被编码为1001101001010110。

- **NRZ码转为曼彻斯特码**

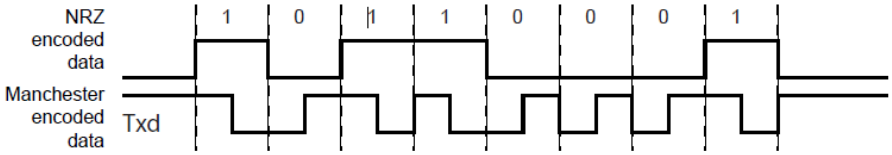


图 30-3：NRZ 码转为曼彻斯特码

曼彻斯特编码字符也可以通过增加一个可配置的前同步信号和一个帧起始定界符样式来封装。根据配置，前同步信号是一个训练序列，由预定义模式组成，其长度可编程为1到15个比特时间。若前同步信号长度被设为0，将不会产生前同步信号波形。前同步信号模式有以下几种序列进行选择：ALL_ONE、ALL_ZERO、ONE_ZERO或ZERO_ONE，将其写入USART_MAN寄存器的TX_PP位域，位TX_PL则用来设定前同步信号长度。下图说明并定义了有效模式。为了提高灵活性，可通过USART_MAN寄存器的TX_MPOL位域来配置编码方案。若设置TX_MPOL位为0（默认为0），则通过0到1的转换来对逻辑0进行编码，用1到0的转换来对逻辑1进行编码。若TX_MPOL位域设为1，则用0到1的转换来对逻辑1进行编码，用0到1的转换来对逻辑1进行编码。

- **前同步信号模式，采用默认极性**

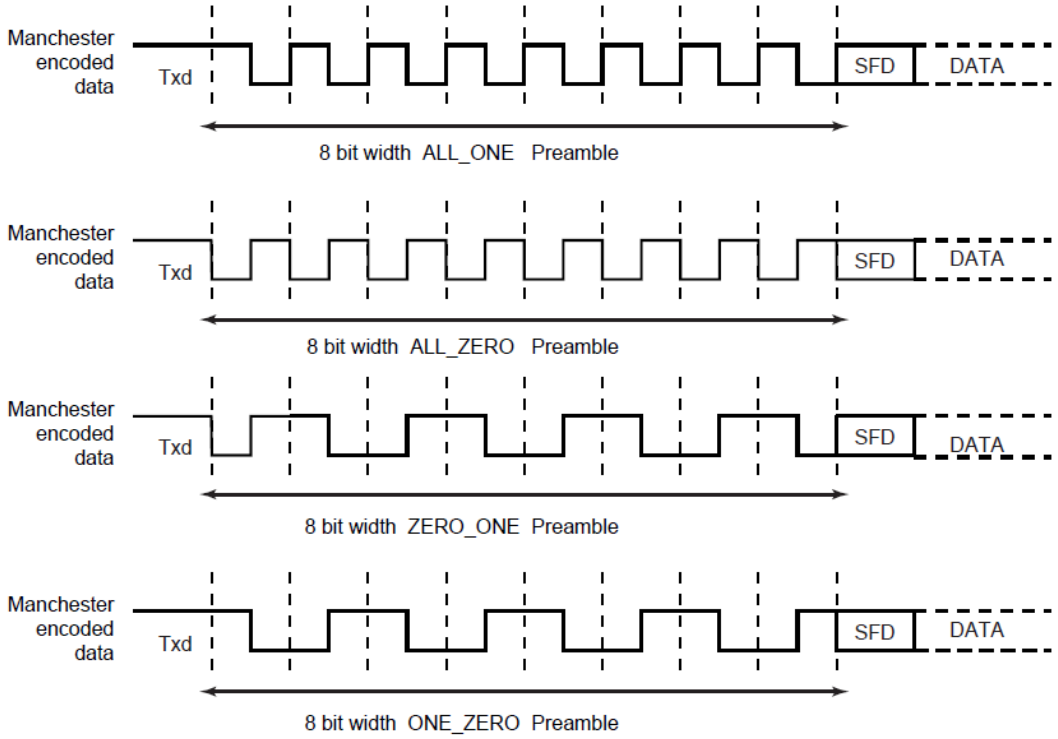


图 30-4：前同步信号模式

通过USART_MR寄存器的ONEBIT位可配置一个帧起始定界符，其由一个用户定义模式组成，用来表示有效数据的开始。图 30-4给出了这些模式。若帧起始定界符，即开始位，是一个比特位，（ONEBIT设为1），当检测到用曼彻斯特编码的逻辑0，则认为一个新的字符正在串行线上发送。若帧起始定界符是一种同步模式，或者说是一个同步（ONEBIT设为0）符，当有一个3个位时间的序列在线上串行发送时，则认为一个新字符的开始。当转换发生在第二个比特时间中间时，同步符波形本身就是一个无效的曼彻斯特波形。有两种不同的同步模式：命令同步符和数据同步符。命令同步符用一个高电平来表示1，持续1.5个位时间；然后转换到低电平表示第二个1，持续1.5个位时间。若将USART_MR寄存器的MODSYNC位域设置为1，则下一个字符则为命令同步符；如果设为0，则下一个字符则是数据。当使用DMA时，可通过修改内存中一个字符来更新MODSYNC位域。为了允许该模式，必须将USART_MR寄存器的VAR_SYNC位域值设置为1。这样USART_MR中的MODSYNC位被忽略，同步符由USART_THR的TXSYNH的位域进行配置。USART字符格式将被修改，并包含同步符信息。

- **帧起始定界符**

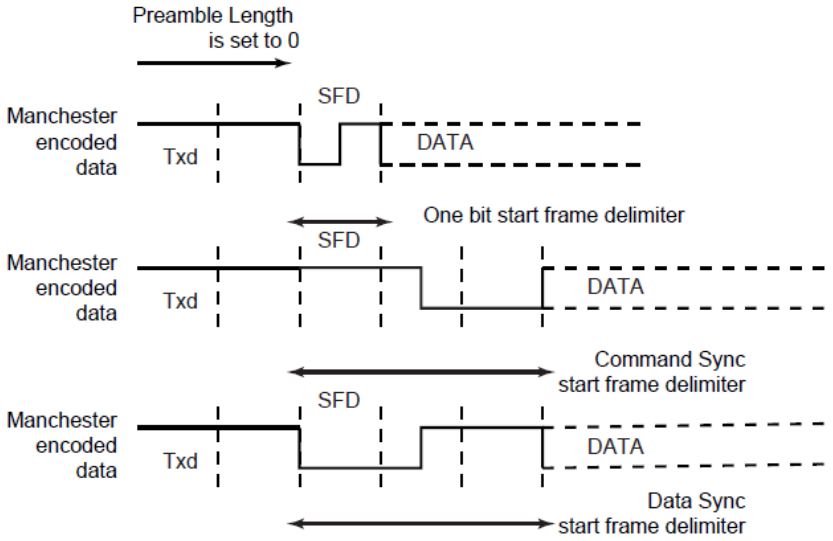


图 30-5：帧起始定界符

30.4.3.3 漂移补偿

漂移补偿仅在16倍过采样模式下有效。一个硬件修复系统允许更大的时钟漂移。可通过将USART_MAN位置1来允许此硬件系统。若RXD的边沿（上升沿或下降沿）处于期望的16倍时钟周期的边沿，这被认为是正常跳动，没有纠正操作。如果TXD事件发生在预期边沿之前的2到4个时钟周期内，当前周期被缩短一个时钟周期。如果TXD事件发生在预期边沿之后的2到3个时钟周期内，当前周期被延长一个时钟周期。这些间隔被当作漂移，纠正操作将自动进行。

- 位同步

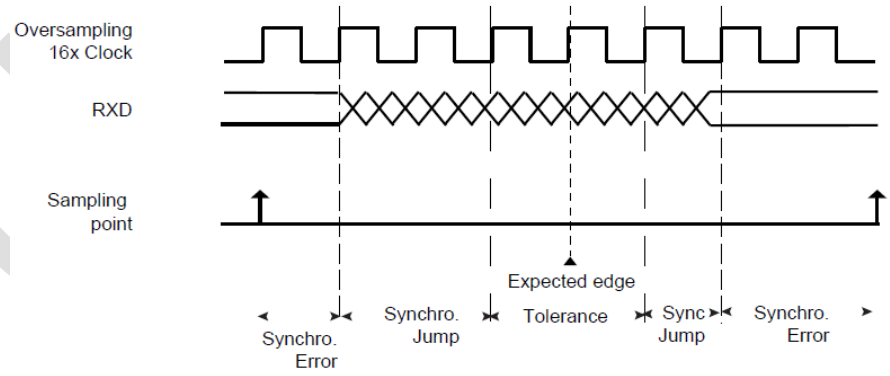


图 30-6：位同步图

30.4.3.4 异步接收器

若USART工作在异步模式下（SYNC = 0），接收器将对RXD输入线进行过采样。过采样频率为16或8倍波特率，由USART_MR中的OVER位设置。

接收器对RXD线采样。若在1.5个比特时间内采样值均为0，即表示检测到起始位，然后以比特率

对数据位、校验位、停止位等进行采样。

若过采样频率为16倍波特率（OVER为0），连续8次采样结果均为0，则表示检测到起始位。之后，每隔16个采样时钟周期对后续的数据位、校验位、停止位依次采样。若过采样频率为8倍波特率(OVER为1)，连续4次采样结果均为0，表示检测到起始位。之后，每隔8个采样时钟周期对数据位、校验位、停止位依次采样。

接收器设置数据的位数、最先发送位及校验模式的位域与发送器相同，即分别为CHRL、MODE9、MSBF及PAR。停止位数对接收器无效，因为无论NBSTOP域为何值，接收器都只确认1个停止位，因此发送器与接收器间可出现重同步。此外，接收器在检测到停止位后即开始寻找新的起始位，因此当发送器只有1个停止位时也能实现重同步。

下图给出当USART工作在异步模式下的起始位检测及字符接收。

- **异步起始位检测**

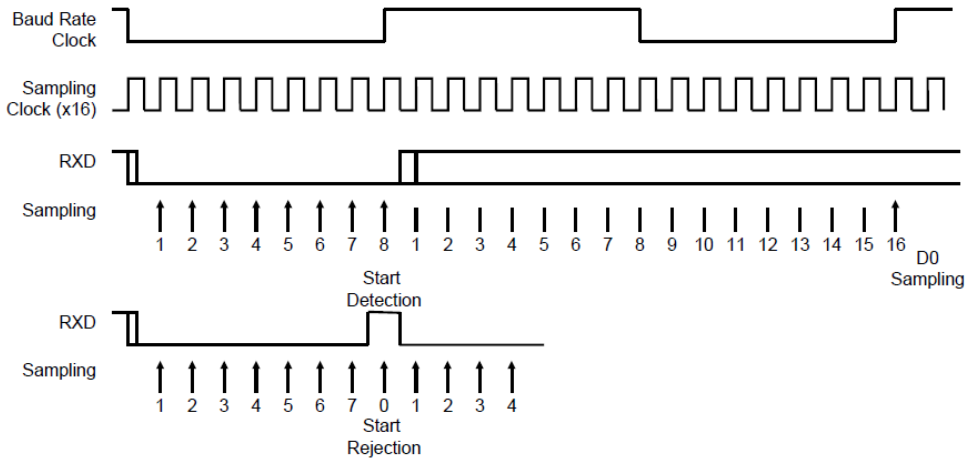


图 30-7：异步起始位检测

- **字符接收**

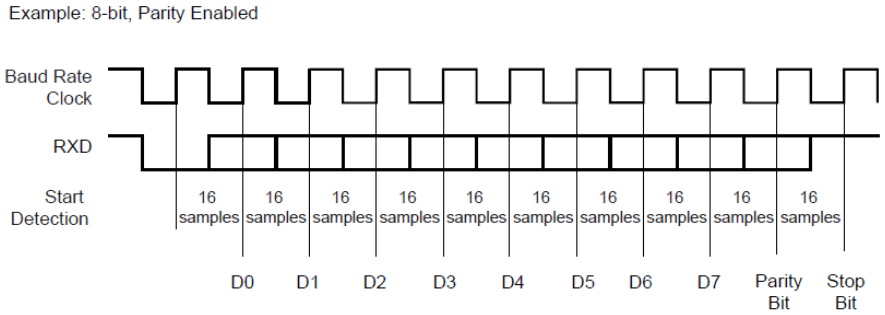


图 30-8：字符接收

30.4.3.5 曼彻斯特解码器

当USART_MR寄存器的MAN位域设置为1，曼彻斯特解码器被允许。解码器将进行前同步信号、帧起始定界符的检测。其中一条输入线专门用作曼彻斯特编码数据输入。

可以定义一个可选前同步信号序列，其长度也可用户定义，并且与发射端完全独立。通过设置

USART_MAN寄存器的RX_PL位来配置前同步信号序列长度。若长度设为0，不检测前同步信号且功能禁用。另外，输入流极性也可通过USART_MAN寄存器的RX_MPOL位域进行设置。根据应用的需求，可通过USART_MAN的RX_PP位设定前同步信号模式，使之相匹配。不像前同步信号，帧起始定界符在曼彻斯特编/解码器中共享。因此，如果ONEBIT位置1，只有0曼彻斯特编码能被检测到，并被当作有效的帧起始定界符。如果ONEBIT位置0，只有同步模式可以被检测到，并被当作有效的帧起始定界符。

解码器通过检测输入流的转换进行操作。如果RXD在1/4比特时间内被采样为低电平，则认为检测到一个开始位，如下图所示。采样脉冲拒绝设备请求。

- **异步起始位检测**

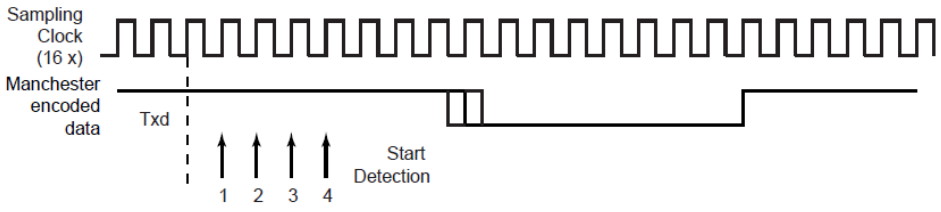


图 30-9：异步起始位检测

接收器被激活并且开始前同步信号和帧分隔符检测，并分别在1/4和3/4周期采样数据。若一个有效前同步信号或帧起始定界符被检测到，接收器将继续用同样的同步时钟进行解码。如果数据流不能和有效模式或有效帧起始定界符不匹配，则接收器将在下一个边沿重新同步。估计位值的最小时间阀值是3/4位时间。若检测到帧起始定界符之后跟着一个有效的前同步信号（如果使用），输入流将被解码成NRZ编码数据并传送给USART处理。图 30-10给出了曼彻斯特编码模式不匹配的情况。当输入数据流被传送给USART，接收器也能够检测到是否违反曼彻斯特编码。违反编码在位元中间缺少电平转换。这种情况，USART_CSR寄存器中的MANE标志被置1。通过对控制寄存器（USART_CR）的RSTSTA位置1可清除MANE标志。下图给出了一个在数据传送阶段曼彻斯特错误检测的例子。

- **前同步信号模式不匹配**

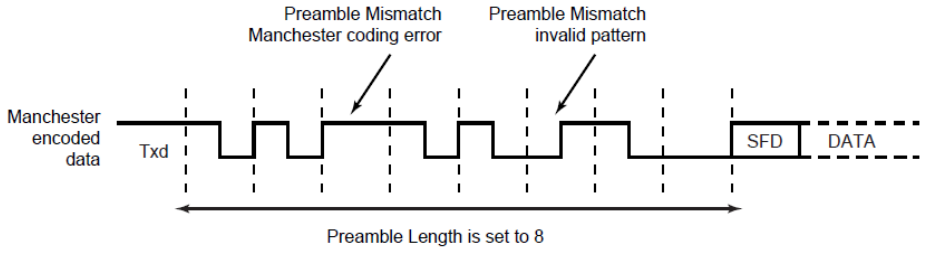


图 30-10：前同步信号模式不匹配

- **曼彻斯特错误标志**

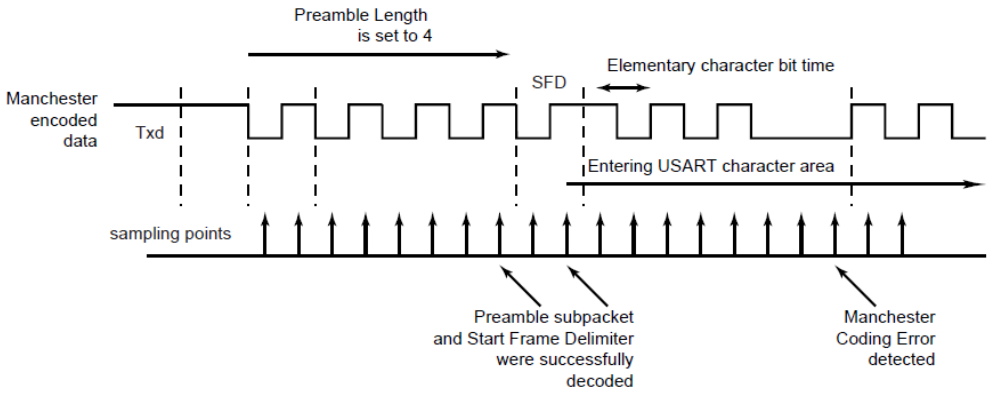


图 30-11：曼彻斯特错误标志

当帧起始定界符是同步模式（ONEBIT位置0），支持命令同步符和数据同步符。如果检测到一个有效同步，接收到的字符被写入USART_RHR寄存器的RXCHR的位域，同时RXSYNH被更新。当接收到的字符是命令时，RXCHR被置1，当接收到的字符是数据时RXCHR被置0。这种机制缓解和简化了直接内存访问DMA，因为在同样的寄存器中字符已经包含了它自己的同步位域。

由于解码器是被设置在单极模式中使用，帧的首位必须是一个0到1的电平转换。

30.4.3.6 无线接口：曼彻斯特编码在 USART 中的应用

本节描述低数据速率射频传输系统，以及它们与曼彻斯特编码USART的集成。这些系统是基于发射器和接收器的ICs，且ICs支持ASK和FSK调制方案。

系统的目的是使用两个不同的频率载波来实现无线全双工字符传输。所有配置如下图所示：

- **曼彻斯特编码字符射频传输**

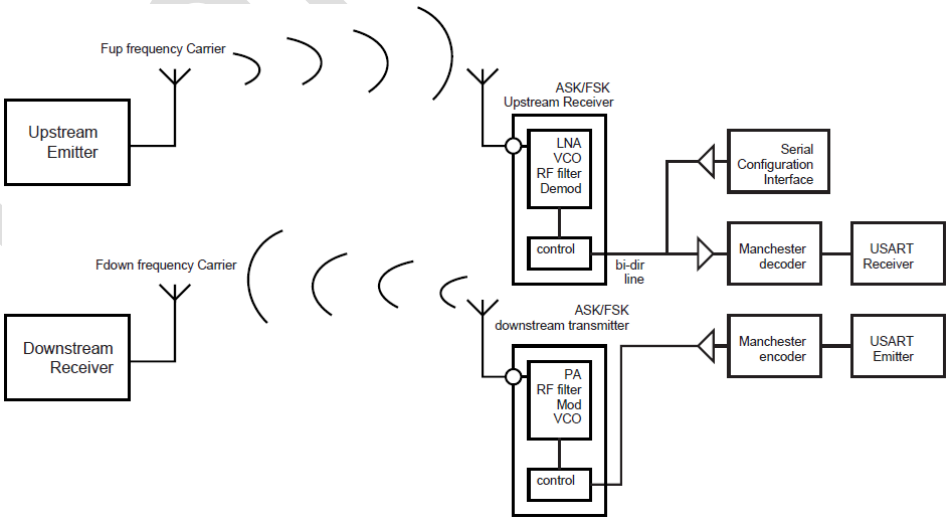


图 30-12：曼彻斯特编码字符射频传输图

USART模块被配置为曼彻斯特编码器/解码器。注意下行的通信，通道曼彻斯特编码字符被串行发送到射频发射器。这也许也包括用户定义的前同步信号和一个帧起始定界符。通常，前同步信号

被用于射频接收器，用来区分是由发射器产生的有效数据还是由于噪声信号。之后对曼彻斯特数据流进行调制。下图给出了ASK调制方案的一个例子。当ASK调制器收到一个逻辑高电平，功率放大器（简称PA）被允许，并用下行频率传输一个射频信号。当ASK调制器收到一个逻辑0，射频信号被关闭。若FSK调制器被激活，采用两种不同的频率来发送数据。当发送逻辑1，调制器以F0频率输出一个射频信号；若发送的数据是逻辑0则频率切换到F1频率，如下图所示。

对于接收端，采用另一个载波频率。射频接收器采用位检测操作来检测解调数据流。若检测到一个有效模式，接收器切换到接收模式，解调数据流被发往曼彻斯特解码器，由于射频IC里面的位检测，通过用户定义的位数可减少传输到控制器的数据。曼彻斯特前同步信号长度根据射频IC的配置来定义。

- **ASK调制器输出**

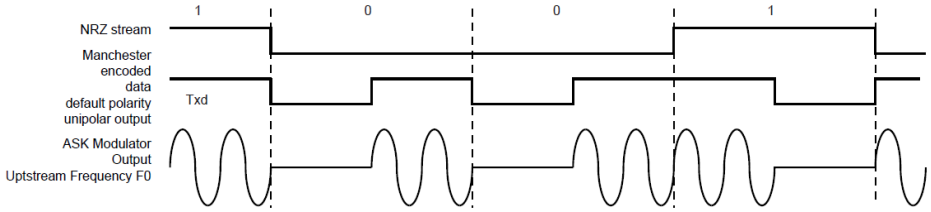


图 30-13：ASK 调制器输出

- **FSK调制器输出**

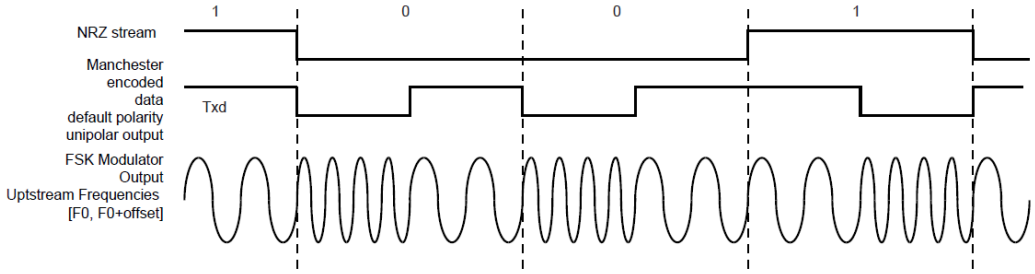


图 30-14：FSK 调制器输出

30.4.3.7 同步接收器

同步模式下（SYNC = 1），接收器在每个波特率时钟的上升沿对RXD信号采样。若检测到低电平，确定为起始位；依次采样所有数据位、校验位及停止位后；接收器将继续等待下一个起始位。同步模式提供高速传输能力。

域及位的配置与异步模式下相同。下图描述了同步模式下的字符接收时序。

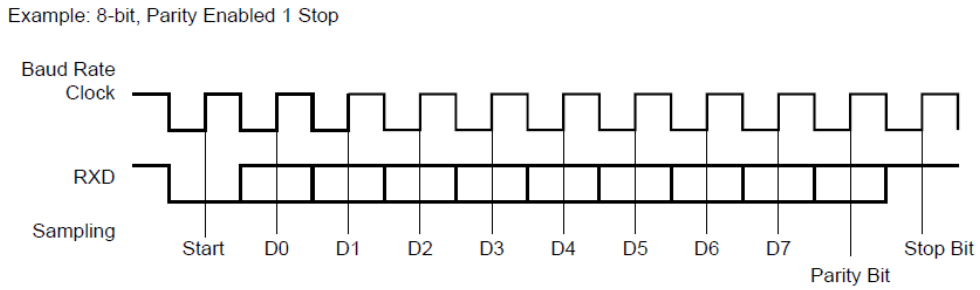


图 30-15：同步模式字符接收

30.4.3.8 接收器操作

当字符接收完成，它将被传输到接收保持寄存器（USART_RHR），且状态寄存器（USART_CSR）的RXRDY位变高。当RXRDY置位时接收完一个字符，则OVRE (溢出错误) 位置位。最后的字符传输到USART_RHR并覆盖上一个字符。通过控制寄存器（USART_CR）的RSTSTA（复位状态）位写1，可清除OVRE位。

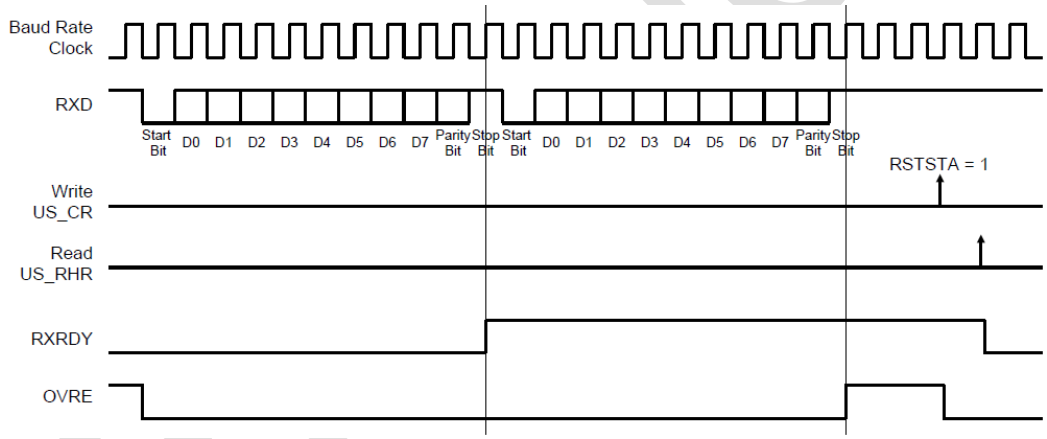


图 30-16：接收器状态

30.4.3.9 校验

通过设置模式寄存器（USART_MR）的PAR位域，USART可支持5种检验模式。PAR域还可以允许多点模式（Multidrop mode），详细介绍请参见30.4.3.10 多点模式章节。支持奇偶检验位的生成以及错误检测。

若选择偶检验，当发送器发送1的数目为偶数时，校验位发生器产生的校验位为0；当发送器发送1的数目为奇数时，校验位发生器产生的校验位为1。相应的，接收器校验位检测器会对收到的1计数，若计算所得校验位与采样所得校验位不同，则报告偶检验错误。若选择奇检验，当发送器发送1的数目为偶数时，校验位发生器产生校验位为1；当发送器发送1的数目为奇数时，校验位发生器产生校验位为0。相应的，接收器校验位检测器会对收到的1计数，若计算所得校验位与采样所得校验位不符，则报告奇检验错误。若使用标志检验，对于所有字符，检验发生器所产生的校验位均为1；

若接收器采样得到的校验位为0，则接收器校验位检测器报告检验错误。若使用空间检验，对于所有字符，检验发生器所产生校验位均为0；若接收器采样得到的校验位为1，接收器校验位检测器报告检验错误。若检验禁用，发送器不产生校验位，接收器也不报告检验错误。

下表给出根据USART不同配置，字符0x41 (ASCII字符 “A”)的所对应奇偶校验位的例子。由于有两位为1，当为奇校验时加“1”，为偶校验时加“0”。

表 30-2：USART 奇偶校验位示例

字符	十六进制	二进制	校验位	校验模式
A	0x41	0100 0001	1	奇
A	0x41	0100 0001	0	偶
A	0x41	0100 0001	1	标志
A	0x41	0100 0001	0	空间
A	0x41	0100 0001	无	无

当接收器检测到检验误差，它将置通道状态寄存器（USART_CSR）的PARE（检验错误）位。通过对控制寄存器（USART_CR）的RSTSTA位写1，可清除PARE位。如图给出校验位的置位与清零时序。

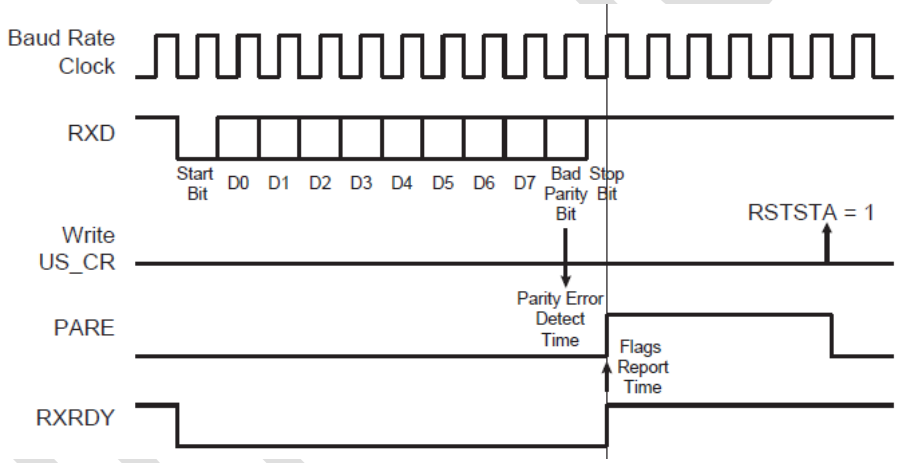


图 30-17：检验错误时序图

30.4.3.10 多点模式

若模式寄存器（USART_MR）的PAR域编程设置为0x6或0x7，USART将运行在多点模式下。该模式区分数据字符与地址字符。当校验位为0时发送数据；当校验位为1时发送地址。

USART配置为多点模式，当校验位为高时，接收器将对校验错误位PARE置位；当控制寄存器的SENDA位为1时，校验位为高发送器也可发送字符。

为处理检验错误，将控制寄存器RSTSTA位写1可对PARE位清零。当SENDA写入USART_CR时，发送器发出地址字节（校验位置位）。这种情况，下一个写入USART_THR的字节将作为地址来发送。如果没有写SENDA命令，任何写入USART_THR的字符将正常被发送（校验位为0）。

30.4.3.11 发送器时间保护

时间保护特性允许USART与慢速远程器件的连接。

时间保护功能允许发送器TXD线上两字符间插入空闲状态。该空闲状态实际上是一个长停止位。

空闲状态的持续时间由发送时间保护寄存器（USART_TTGR）的TG域编程设定。若该域编程值为零，则不产生时间保护。否则，在每次发送了停止位之外，TXD还保持TG中指定的位数周期的高电平。

如下图所示，TXRDY与TXEMPTY状态位的行为可由时间保护改变。TXRDY只有在下一字符起始位发送后才变高。即使字符已写入USART_THR，时间保护期间TXRDY仍保持为0。由于时间保护是当前传输的一部分，因此TXEMPTY为低要保持到时间保护阶段结束。

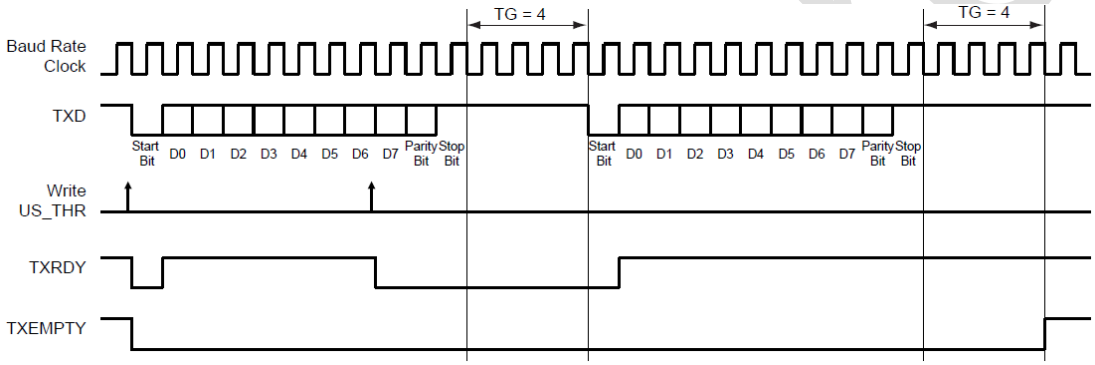


图 30-18：时间保护操作时序图

不同波特率下，发送器的时间保护周期的最大值：

表 30-3：保护周期的最大值

波特率（bit/s）	位时间（μs）	时间保护长度（ms）
1200	833	212.50
9600	104	26.56
14400	69.4	17.71
19200	52.1	13.28
28800	34.7	8.85
33400	29.9	7.63
56000	17.9	4.55
57600	17.4	4.43
115200	8.7	2.21

30.4.3.12 接收器超时

接收器超时支持对可变长度帧的处理。接收器可检测RXD线上的空闲状态，如果检测到超时，通道状态寄存器（USART_CSR）的TIMEOUT位将变高并产生中断，以告知驱动程序帧结束。

通过对接收器超时寄存器（USART_RTOR）的TO域编程，可设置超时延迟周期（接收器等待

新字符的时间）。若TO域设置为0，接收器超时被禁用，将不检测超时；USART_CSR寄存器中TIMEOUT位保持为0。否则，接收器将TO的值载入一个16位计数器。该计数器在每比特周期中自减，并在收到新字符后重载。若计数器达到0，状态寄存器中TIMEOUT位变高，用户可：

- 停止计数器时钟，直到接受到新字符。可通过对控制寄存器（USART_CR) 的 STTTO（启动超时）位写 1 来实现之。这样，在接收到字符之前 RXD 线上的空闲状态将不会产生超时，而且可避免在接收字符之前必须去处理中断，而且还允许帧接收之后时等待 RXD 上的下一个空闲状态。
- 在没有收到字符时将产生一个中断。可通过对 USART_CR 中 RETTO（重载与启动超时）位写 1 来实现之。若 RETTO 被执行，计数器开始从 TO 值向下计数。产生的周期性中断可以用来处理用户超时，例如当键盘上无键按下时。

若执行STTTO，计数器时钟在收到第一个字符前停止。帧启动之前RXD的空闲状态不提供超时。这样可防止周期性中断，并在检测到RXD为空闲状态时允许帧结束等待。

若执行RETTO，计数器开始从TO值向下计数。产生的周期性中断可用来处理用户超时，例如当键盘上无输入时。

表 30-4：某些标准波特率下的最大超时周期

波特率（bit/sec）	位时间（μs）	超时长度（ms）
600	1667	109225
1200	833	54613
2400	417	27306
4800	208	13653
9600	104	6827
14400	69	4551
19200	52	3413
28800	35	2276
33400	30	1962
56000	18	1170
57600	17	1 138
200000	5	328

30.4.3.13 帧错误

接收器可检测帧错误。当检测到收到字符的停止位为0时表示发生了帧错误。当接收器与发送器为完全不同步时可能出现帧错误。

帧错误由USART_CSR寄存器的FRAME位表示。检测到帧错误时，FRAME位在停止位时间的中间被设置。通过将USART_CR寄存器的RSTSTA位写为1，可将其清除。

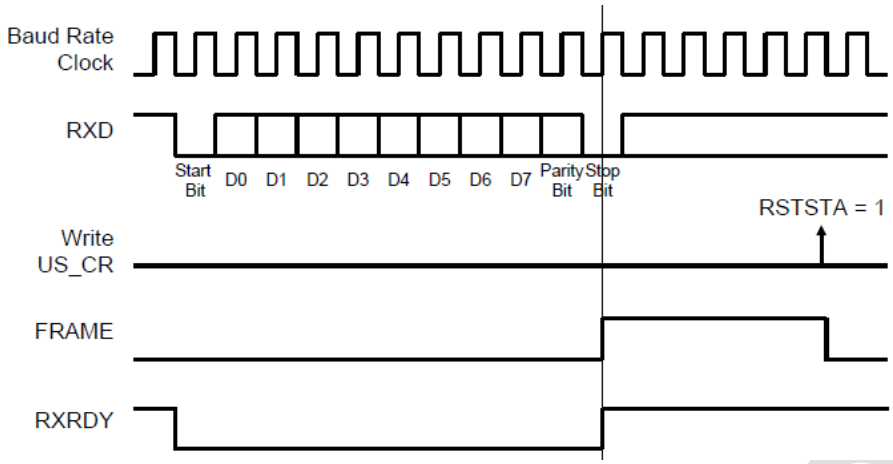


图 30-19：帧错误状态时序图

30.4.3.14 发送间断

用户可请求发送器在TXD线上产生间断条件。可使得在至少一个完整字符时间内TXD线为低。这与校验位及停止位均为0的0x00字符的发送情况相同。无论如何，发送器至少保证TXD线在一个完整的字符传输时间内为低，直到用户请求将间断条件删除。

将USART_CR寄存器STTBRK位写1可发送一个间断。这可在任何时间执行，即使发送器为空（移位寄存器及USART_THR中均无字符）或字符正在发送。若有字符正在移出时出现间断请求，在TXD线变低之前先完成字符传输。

一旦需要STTBRK命令，在间断完成前将忽略其他STTBRK命令。

通过往USART_CR寄存器的STPBRK位写1可删除间断条件。若在最小间断持续时间（一个字符，包括起始位、数据位、校验位及停止位）结束前请求STPBRK，发送器保证间断条件完成。

发送器将间断视为一个字符，即只有当USART_CSR寄存器中TXRDY为1时，STTBRK与STPBRK命令才被考虑；如处理一个字符一样，在间断条件启动时将清除TXRDY与TXEMPTY位。

将USART_CR寄存器中STTBRK与STPBRK位写为1，将导致无法预测的结果。所有前面没有STTBRK命令的STPBRK命令请求将被忽略。当间断挂起时，写入发送保持寄存器但未启动的字节将被忽略。

间断条件后，最多12比特时间内，发送器将TXD线变回1。因此，发送器保证远程接收器正确检测到间断结束以及下一个字符的启示位。若时间保障值大于12，TXD线将在时间保障期内保持高电平。

在TXD线保持这一周期为高之后，发送器恢复正常工作。

下图给出TXD线上开始间断（STTBRK）与停止间断（STPBRK）命令的作用。

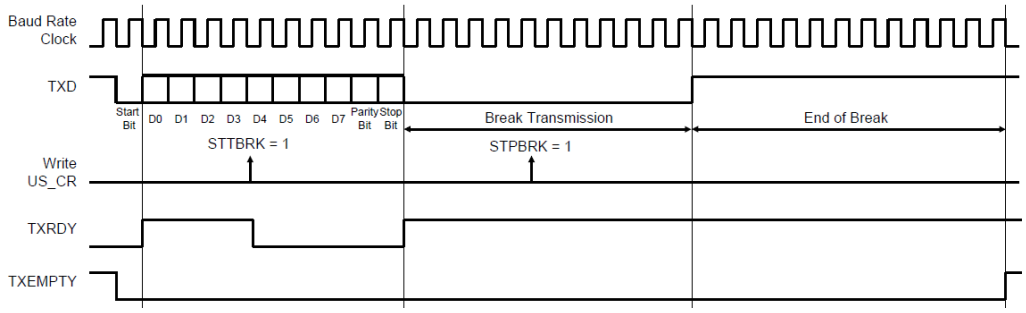


图 30-20：间断的传输

30.4.3.15 接收间断

当收到的所有数据、检验及停止位均为低时，接收器检测到间断条件。这与数据为0x00且FRAME为低（帧错误）的帧的检测结果相同。

当检测到低电平的停止位时，接收器将对USART_CSR寄存器的RXBRK位置位，该位可通过对USART_CR寄存器的RSTSTA位写1来清零。

在异步工作模式下检测到至少2/16个位周期为高电平，或在同步工作模式下有一个采样值为高，表明接收间断结束。间断结束也可通过对RXBRK置位来实现。

30.4.3.16 硬件握手

USART可以通过硬件握手来实现带外（out-of-band）数据流的控制。RTS与CTS引脚用于与远程器件连接，如下图所示：

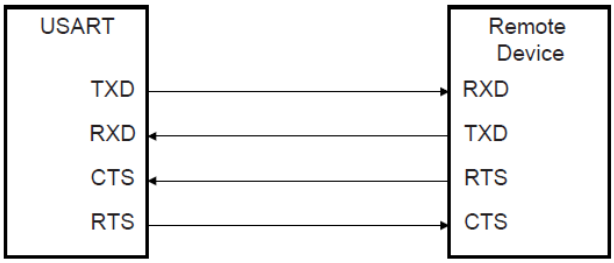


图 30-21：与远程器件连接的硬件握手

在USART_MR寄存器USART_MODE域写入0x2，则USART将执行硬件握手操作。

除了接收器对RTS引脚电平、发送器对CTS引脚电平的控制按后述方式改变外，允许进行硬件握手之后的USART操作与标准同步或异步模式下相同。使用该模式需要用PDC通道接收数据。发送器在任何情况下均可处理硬件握手。

下图给出当允许硬件握手时接收器的操作。若接收器被禁用且来自PDC通道的RXBUFF（接收缓冲满）状态为高，则RTS引脚拉高。通常当CTS引脚（由RTS驱动）为高时远程器件不会开始发送。一旦接收器被允许，RTS变低，告知远程器件可启动发送。给PDC定义一个新缓冲器，将RXBUFF状态位清零，则RTS引脚电平变低。

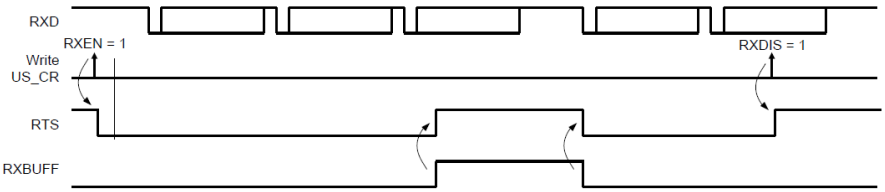


图 30-22：允许硬件握手时接收器工作行为

下图给出允许硬件握手后，发送器是如何操作的。CTS引脚禁用发送器；若有字符正在处理，则在当前字符处理完成后将发送器禁用；一旦CTS变低即开始下一个字符的发送。

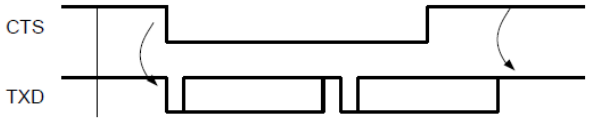


图 30-23：允许硬件握手时发送器工作行为

30.4.4 IrDA 模式

USART 的 IrDA 模式支持半双工点对点无线通信。它内置了与红外收发器无缝连接的调制器和解调器，如图所示。调制器与解调器与 IrDA 规范版本 1.1 兼容，支持的数据传输速度范围从 2.4Kb/s 到 115.2Kb/s。

通过对 USART_MR 寄存器的 USART_MODE 域写 0x8，可允许 USART IrDA 模式。通过 IrDA 滤波寄存器（USART_IF）可配置解调滤波器。USART 发送器与接收器工作在正常异步模式下，所有参数均可访问。注意，调制器与解调器均处于激活状态。

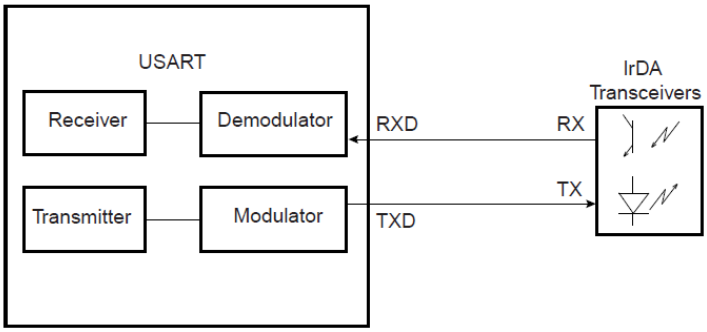


图 30-24：与 IrDA 收发器连接

接收器与发送器必须根据传输方向允许或禁止。要接收 IrDA 信号，必须进行以下配置：

- 禁止 TX，允许 RX
- 配置 TXD 为 PIO，并且设置其输出为 0（避免 LED 发光），禁止内部上拉（以减少功耗）。
- 接收数据。

30.4.4.1 IrDA 调制

当波特率小于或等于 115.2Kb/s，使用 RZI 调制方案。"0"由一个 3/16 比特周期的光脉冲表示。下表给出一些信号脉冲持续时间。

表 30-5：IrDA 脉冲持续时间

波特率	脉冲持续时间（3/16)
2.4 Kb/s	78.13 μs
9.6 Kb/s	19.53 μs
19.2 Kb/s	9.77 μs
38.4 Kb/s	4.88 μs
57.6 Kb/s	3.26 μs
115.2 Kb/s	1.63 μs

下图为字符发送的示例。

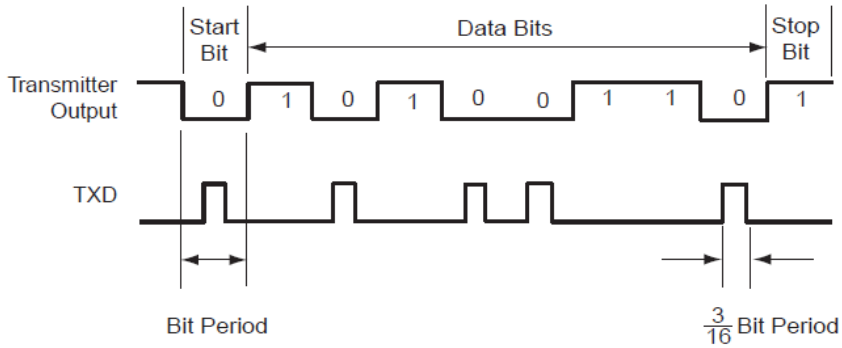


图 30-25：IrDA 调制

30.4.4.2 IrDA 波特率

下表给出一些 CD 值、波特率误差及脉冲持续时间的例子。注意，可接受的最高误差为 ±1.87%。

表 30-6：IrDA 波特率误差

外设时钟	波特率	CD	波特率误差	脉冲时间
3686400	115200	2	0.00%	1.63
20000000	115200	11	1.38%	1.63
32768000	115200	18	1.25%	1.63
40000000	115200	22	1.38%	1.63
3686400	57600	4	0.00%	3.26
20000000	57600	22	1.38%	3.26
32768000	57600	36	1.25%	3.26
40000000	57600	43	0.93%	3.26
3686400	38400	6	0.00%	4.88
20000000	38400	33	1.38%	4.88
32768000	38400	53	0.63%	4.88

外设时钟	波特率	CD	波特率误差	脉冲时间
40000000	38400	65	0.16%	4.88
3686400	19200	12	0.00%	9.77
20000000	19200	65	0.16%	9.77
32768000	19200	107	0.31%	9.77
40000000	19200	130	0.16%	9.77
3686400	9600	24	0.00%	19.53
20000000	9600	130	0.16%	19.53
32768000	9600	213	0.16%	19.53
40000000	9600	260	0.16%	19.53
3686400	2400	96	0.00%	78.13
20000000	2400	521	0.03%	78.13
32768000	2400	853	0.04%	78.13

30.4.4.3 IrDA 解调器

解调器基于 IrDA 接收滤波器，包含一个 8 位向下计数器，值从 USART_IF 中载入。当检测到 RXD 引脚上下降沿，滤波计数器开始以主机时钟（MCK）速度向下计数。若检测到 RXD 引脚上升沿，则计数器停止并重新载入 USART_IF 的值。若计数器到达 0 仍未检测到上升沿，则在一个位时间内将接收器的输入拉低。

下图给出 IrDA 解调器的操作。

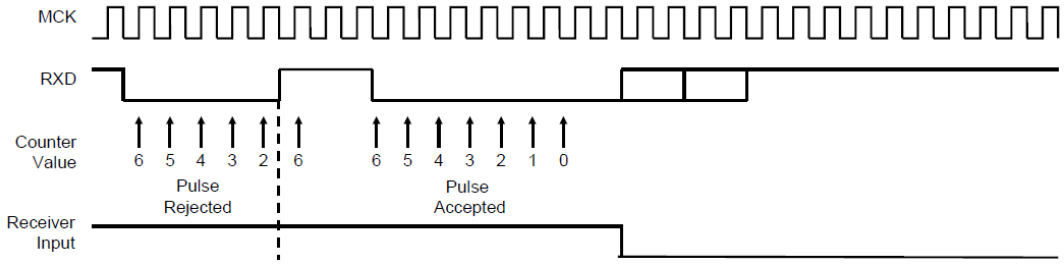


图 30-26：IrDA 解调器操作

注意 USART_FIDI 中的 FI_DI_RATIO 域值必须大于 0，以确保 IrDA 通信操作正确。

30.4.5 SPI 模式

串行外设接口（SPI）模式是同步串行数据链接，可以主机或从机模式与外部器件进行通信。

若外部处理器与系统连接，它还允许处理器间通信。

串行外设接口实质上是一个将数据串行传输到其它 SPI 的移位寄存器。数据传输时，一个 SPI 系统作为“主机”控制数据流，其它 SPI 作为“从机”，在主机的控制下移入或移出数据。不同的 CPU 可轮流作为主机且一个主机可同时将数据移入多个从机（多主机协议与单主机协议不同，单主机协议中只有一个 CPU 始终作为主机，其它 CPU 始终作为从机）。但任何时候只允许一个从机将其数

据写入主机。

SPI 系统由主机发出 NSS 信号选定一个从机。SPI 主机模式的 USART 只能连接一个从机，因为它只能产生一个 NSS 信号。

SPI 系统包括两条数据线及两条控制线：

- 主机输出从机输入（MOSI）：该数据线用于将主机输出数据移入到从机中。
- 主机输入从机输出（MISO）：该数据线用于将从机的数据输出到主机。
- 串行时钟（SCK）：该控制线由主机驱动，用来调节数据流；主机传输时数据波特率可变；每个 SCK 周期传输一位。
- 从机选择（NSS）：该控制线用于主机选择或取消选择从机。

30.4.5.1 工作模式

USART 可工作在 SPI 主机模式或 SPI 从机模式下。

通过对模式寄存器的 USART_MODE 位写 0XE，USART 可工作在 SPI 主机模式下。在这种情况下必须按如下说明进行连接 SPI 线:

- 输出引脚 TXD 驱动 MOSI 线
- MISO 线驱动输入引脚 RXD
- 输出引脚 SCK 驱动 SCK 线
- 输出引脚 RTS 驱动 NSS 线

通过对模式寄存器的 USART_MODE 位写 0XF，USART 可工作在 SPI 从机模式下。在这种情况下必须按如下说明进行 SPI 线连接:

- MOSI 线驱动输入引脚 RXD
- 输出引脚 TXD 驱动 MISO 线
- SCK 线驱动输入引脚 SCK
- NSS 线驱动输入引脚 CTS

为避免不可预测行为，SPI 模式一旦发生变化，必须对发送器和接收器进行软件复位（除了硬件复位后的的初始化配置）。

30.4.5.2 波特率

在 SPI 模式下，波特率发生器操作和 USART 同步模式相同。不过要必须遵守以下约束：

- SPI 主机模式：
  - 为了在 SCK 引脚上产生正确的串行时钟，不能选择外部时钟 SCK(USCLKS≠0x3)，且模式寄存器（USART_MR）的 CLKO 位必须置 1。
  - 为了接收器和发送器能够正常工作，CD 值必须大于等于 4。

- 若选择了内部时钟分频(MCK/DIV)，CD 值必须设为偶数，以使 SCK 引脚能够产生 50:50 占空比；如果选择了内部时钟（MCK），则 CD 值也可以设为奇数。

- SPI 从机模式
  - 必须选择外部时钟（SCK），模式寄存器（USART_MR）的 USCLKS 位域的值无效；同样 USART_BRGR 的值也无效。因为时钟是由 USART 的 SCK 引脚上的信号直接提供。
  - 为了接收器和发送器能够正常工作，外部时钟（SCK）频率不能超过系统时钟频率的 1/4。

30.4.5.3 数据传输

在每个可编程串行时钟的上升沿或下降沿（视 CPOL、CPHA 设置）最多有 9 位数据能连续地在 TXD 引脚上移出，且没有开始位，奇偶校验位和停止位。

可通过设置 CHRL 位和模式寄存器（USART_MR）的 MODE9 位来选择数据的位数。如果选择 9 位数据仅设置 MODE9 位即可，不用关心 CHRL 域。在 SPI 模式（主机或从机模式）下总是先发送最高数据位。

数据传输有四种极性与相位的组合。时钟极性由模式寄存器的 CPOL 位设置；时钟相位通过 CPHA 位设置。这两个参数确定在哪个时钟边沿驱动和采样数据。每个参数有两种状态，组合后就有四种可能。因此，一对主机/从机必须使用相同的参数对值来进行通信。若使用多从机，且每个从机固定为不同的配置，则主机与不同从机通信时必须重新配置。

表 30-7：SPI 总线协议模式

SPI 总线协议模式	CPOL	CPHA
0	0	1
1	0	0
2	1	1
3	1	0

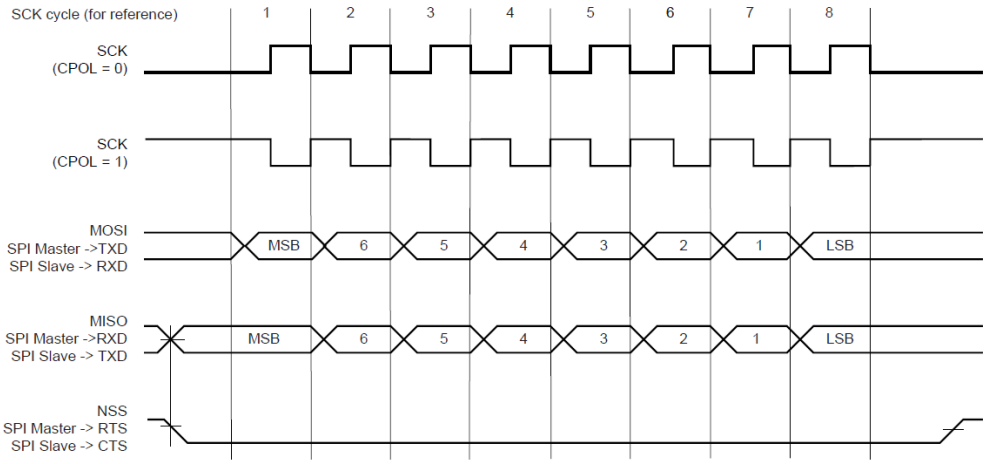


图 30-27：SPI 传输格式（CPHA=1，每次传输 8 位）

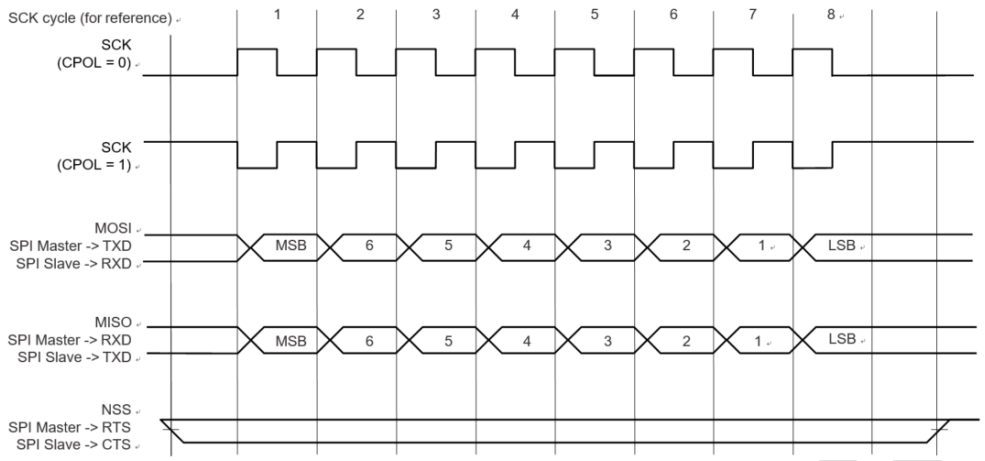


图 30-28：SPI 传输格式（CPHA=0，每次传输 8 位）

30.4.5.4 接收器和发送器控制

- **字符发送**

通过向发送保持寄存器（USART_THR）写入字符进行字符发送。若 USART 工作在 SPI 主机模式，可以增加发送字符的附加条件。当接收器没有准备好（没有读字符），设置 USART_MR 寄存器的 INACK 位值可以禁止任何字符的发送（尽管数据已写入 USART_THR）。若 INACK 设为 0，无论接收器是什么状态，字符都会被发送；若 INACK 设为 1，发送器在发送数据（RXRDY 标志清除）前要等待接收保持寄存器的数据被读取完，这样可以避免接收器产生任何溢出（字符丢失）。

发送器在通道状态寄存器（USART_CSR）有 2 个状态位：TXRDY（发送准备）用来表

USART_THR 为空；TXENPTY 用来表示所有写入 USART_THR 的字符已经被处理完成。当处理完当前字符，写入 USART_THR 的最后一个字符被发送到发送寄存器的移位寄存器，同时 USART_THR 清空，然后 TXRDY 置位。

当发送器被禁止时，TXRDY 和 TXENPTY 位都为 0。当 TXRDY 为 0 时，向 USART_THR 写入字符无效且写入的字符丢失。

若 USART 工作在 SPI 从机模式，并且当发送器保持寄存器（USART_THR）为空时如果一定要发送一个字符，则 UNRE（缓冲区数据为空出错）位置位。在此期间 TXD 发送线保持高电平。通过向控制寄存器（USART_CR）的 RSTSTA（复位状态）位写 1，可清除 UNRE 位。

在 SPI 主机模式下，发送最高位之前，在一个 1 个位时间里从机选择线（NSS）发出低电平信号；在发送最低位之后，NSS 保持一个 1 个位时间的高电平。因此，从机选择信号在字符发送之间总是被释放，总是插入最少 3 个位时间的延迟。然而，为了使从机设备支持 CSAAT 模式（传输后片选激活），可通过将控制寄存器（USART_CR）的 RTSEN 位置 1，将从机选择线（NSS）强制拉低。只有将控制寄存器（USART_CR）的 RTSDIS 位置 1 才能将从机选择线（NSS）拉高释放。（例如当所有数据已发往从机设备）。

在 SPI 从机模式下，发生器不会请求在从机选择线（NSS）的下降沿初始化字符发送，而仅在

低电平时进行。不过，在最高位对应的第一个串行时钟周期之前，从机选择线（NSS）上必须至少持续一个位时间的低电平。

- **字符接收**

当一个字符被接收完，它被转移到接收保持寄存器（USART_RHR），同时状态寄存器（USART_CSR）的 RXRDY 位被拉高。若字符在 RXRDY 置位时被接收，OVER（溢出错误）位置位。最后一个字符被转移到 USART_RHR，并覆盖当前字符。对控制寄存器（USART_CR）的 RSTSTA（复位状态）位写 1 可清空 OVRE 位。

为保证 SPI 从机模式下接收器的正常操作，主机设备在发送帧时必须确保发送每个字符之间有至少一个位时间的延迟。接收器不会请求在从机选择线（NSS）的下降沿时初始化字符接收，而仅在低电平时进行。不过，在最高位对应的第一个串行时钟周期之前，从机选择线（NSS）上必须至少持续一个位时间的低电平。

- **接收超时**

因为接收器波特率时钟仅在 SPI 模式中数据发送时可用，这种模式下接收器是不可能超时的，不管超时寄存器（USART_RTOR）的超时值为多少（TO 位域值）。

30.4.6 LIN 模式

LIN 模式在 LIN 总线上提供主节点和从节点连接。

LIN（本地互联网络）是一种串行通信协议，可有效支持分布式汽车应用中机电节点的控制。

LIN 总线的主要特性有：

- 单主/多从概念
- 基于通用 UART/SCI 接口硬件、等效软件或纯状态机，实现成本低
- 从节点不需晶振或陶瓷震荡器就能实现自同步
- 确定性信号传输
- 低成本单线实施
- 传输速率最高可达 20kbps

LIN 在不需要 CAN 带宽和多功能性的情况下，提供经济高效的总线通信。

LIN 模式允许微处理器以最小动作处理 LIN 帧。

30.4.6.1 操作模式

USART 可以充当 LIN 主节点或 LIN 从节点。

通过设定 USART 模式寄存器（USART_MR）的 USART_MODE 位来选择节点配置：

- LIN 主节点 (USART_MODE = 0xA)
- LIN 从节点 (USART_MODE = 0xB)

为了避免不可预测的行为，LIN 节点配置的任何更改必须在对发送器和接收器进行软件复位后（硬件复位后的初始化节点配置除外）。

30.4.6.2 波特率配置

参考“30.4.1.1 异步模式”章节。

波特率在波特率发生器寄存器（USART_BRGR）中配置。

- LIN 主节点：波特率在波特率发生器寄存器（USART_BRGR）中配置。
- LIN 从节点：初始波特率在波特率生成器寄存器（USART_BRGR）中配置。写入 USART_BRGR 时，此配置会自动复制到 LIN 波特率寄存器（USART_LINBRR）中。同步过程完成后，波特率在 USART_ LINBRR 中更新。

30.4.6.3 接收和发送控制

参考“30.4.2 接收器和发送器控制”章节。

30.4.6.4 字符传输

参考“30.4.3.1 发送器操作”章节。

30.4.6.5 字符接收

参考“30.4.3.8 接收器操作”章节。

30.4.6.6 报文头传输（主节点配置）

所有 LIN 帧都以主节点发送的报文头开始，报文头由同步间隔域、同步域和标识符域组成。

因此，在主节点配置中，帧处理从发送报文头开始。

一旦标识符写入 LIN 标识符寄存器（USART_LINIR），就会传输报文头。同时，TXRDY 标志变为低。

间隔域、同步域和标识符域会依次自动发送。

间隔域由 13 个显性位和 1 个隐性位组成，同步字段是字符 0x55，标识符对应于写入 LIN 标识符寄存器（USART_LINIR）的字符。标识符奇偶校验位会自动计算并发送。

当标识符域被传输到发送器的移位寄存器时，TXRDY 标志变高。

一旦发送同步间隔域，通道状态寄存器（USART_CSR）中的 LINBK 标志置位。同样，一旦发送标识符域，USART_CSR 中的 LINID 标志置位。通过向控制寄存器（USART_CR）中的 RSTSTA 位写 1 来复位这些标志。

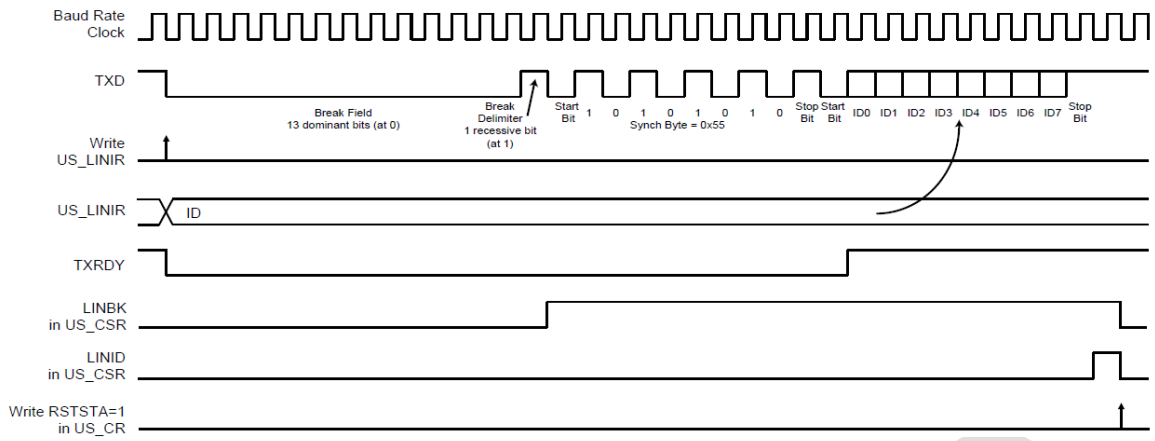


图 30-29：报文传输

30.4.6.7 报文头接收（从节点配置）

所有 LIN 帧都以主节点发送的报文头开始，报文头由同步间隔域、同步域和标识符域组成。

在从节点配置中，帧处理从接收报文头开始。

USART 在实际波特率下使用 11 位的间隔检测阈值。在任何时候，如果在总线上检测到 11 个连续的隐性位，USART 都会检测到一个间隔域。只要未检测到间隔域，USART 就会保持空闲状态，并且不会接收数据。

当检测到间隔域时，通道状态寄存器（USART_CSR）中的 LINBK 标志置位，USART 期望同步域字符为 0x55。此字段用于更新实际波特率以保持同步。如果接收到的同步字符不是 0x55，则会生成不一致的同步域错误。

在接收到同步域后，USART 希望接收标识符域。

当接收到标识符域，USART_CSR 中的 LINID 标志置位。此时，LIN 标识符寄存器（USART_LINIR）中的 IDCHR 字段更新为接收到的字符。标识符奇偶校验位可以自动计算和检查。

如果在报文头的 $t_{Header\_maximum}$ 指定的最大长度时间内未完全接收，则 USART_CSR 中的错误标志 LINHTE 标志置位。

通过向控制寄存器（USART_CR）中的 RSTSTA 位写入一个 1 对 LINID、LINBK 和 LINHTE 标志位复位。

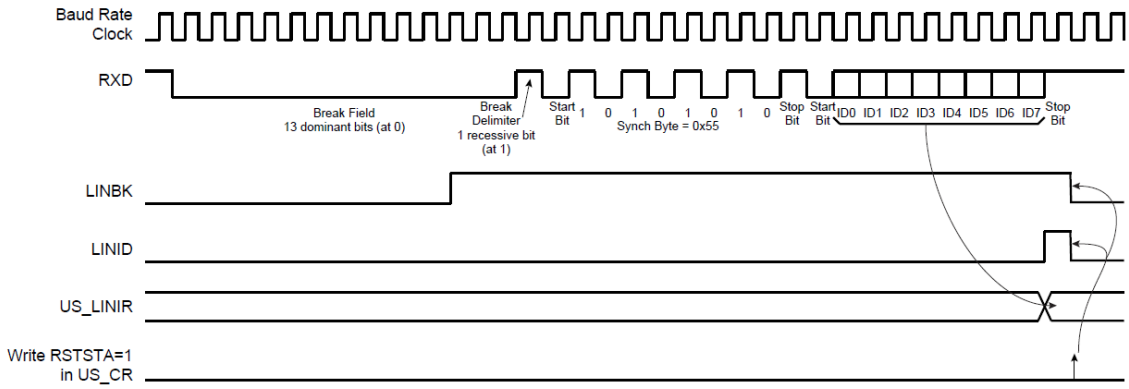


图 30-30：报文接收

30.4.6.8 从节点同步

同步仅在从节点配置中完成。该步骤基于同步域下降沿之间的时间测量。下降沿的距离为 2、4、6 和 8 位时间。

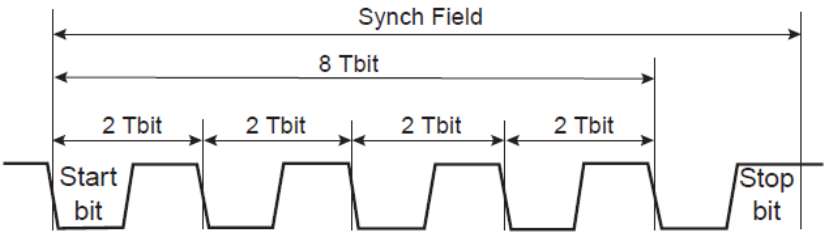


图 30-31：同步域

时间测量由基于采样时钟的 19 位计数器进行。

当检测到同步域的起始位时，计数器复位。然后，在同步域的下一个 8 Tbits 期间，计数器递增。在这 8 Tbits 结束时，计数器停止。此时，计数器的高 16 位（值除以 8）给到新的时钟分频器（LINCD），该值的低 3 位（余数）给到新的小数部分（LINFP）。

接收到同步域后，波特率生成器寄存器（USART_BRGR）中的时钟分频器（CD）和小数部分（FP）将更新。

如果采样的同步字符不等于 0x55，则通道状态寄存器（USART_CSR）中的错误标志 LINISFE 置位。将控制寄存器（USART_CR）中的 RSTSTA 位写 1 来进行复位。

一旦全部接收了同步域，如果 LIN 模式寄存器（USART_LINMR）中的 SYNCDIS 位未禁用同步，则 LIN 波特率寄存器（USART_LINBRR）中的时钟分频器（LINCD）和小数部分（LINFP）将更新。

同步域接收后：

- 如果与初始波特率相比，计算的波特率偏差大于最大公差 FTol_Unsynch（±15%），则时钟分频器（LINCD）和小数部分（LINFP）不会更新，通道状态寄存器（USART_CSR）中的错误标志 LINSTE 置位。
- 如果采样的同步字符不等于 0x55，则时钟分频器（LINCD）和小数部分（LINFP）不会更新，

并且通道状态寄存器（USART_CSR）中的错误标志 LINISFE 置位。

通过将控制寄存器（USART_CR）中的 RSTSTA 位写 1 来复位 LINSTE 和 LINISFE 标志。

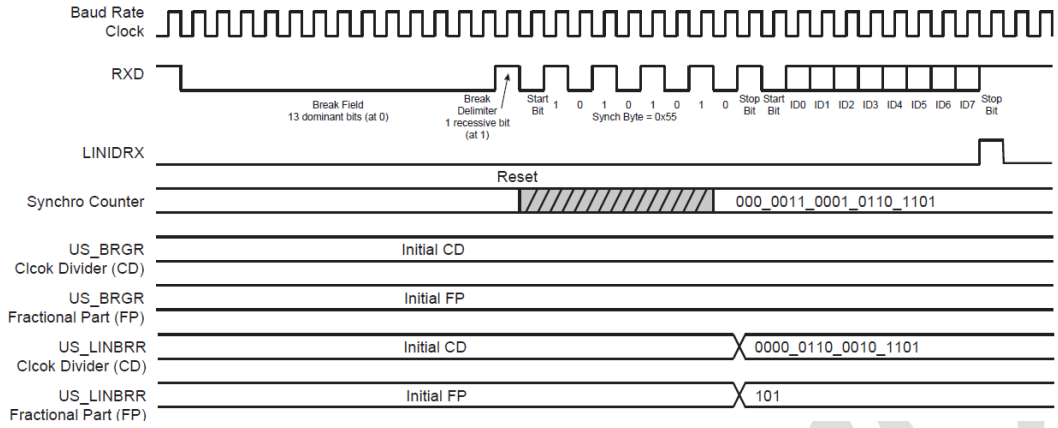


图 30-32：从节点同步时序图

同步的准确性取决于几个参数：

- 标称时钟频率（f_{Nom}）（理论从节点时钟频率）
- 波特率
- 过采样（Over = 0 ≥ 16X 或 Over = 0 ≥ 8X）

以下公式用于计算同步后从比特率相对于主比特率的偏差（f_{SLAVE} 是真正的从节点时钟频率）

$$Baudrate\_deviation = \left(100 \times \frac{[\alpha \times 8 \times (2 - Over) + \beta] \times Baudrate}{8 \times f_{SLAVE}}\right)\%$$

$$Baudrate\_deviation = \left(100 \times \frac{[\alpha \times 8 \times (2 - Over) + \beta] \times Baudrate}{8 \times \left(\frac{f_{TOL\_UNSYNCH}}{100}\right) \times f_{Nom}}\right)\%$$

$f_{TOL\_UNSYNCH}$ 是实际从节点时钟与标称时钟频率的偏差。LIN 标准规定其不得超过±15%。LIN 标准还规定对于两个节点之间的通信，它们的比特率差异不得超过±2%，这表示波特率偏差不得超过±1%。由此得出标称时钟频率的最小值：

$$f_{Nom}(min) = \left(100 \times \frac{[0{,}5 \times 8 \times (2 - Over) + 1] \times Baudrate}{8 \times \left(\frac{-15}{100} + 1\right) \times 1\%}\right)Hz$$

示例：

- Baud rate= 20 kbps, OVER=0 (Oversampling 16X) ≥ f_{Nom}(min) = 2.64 MHz
- Baud rate = 20 kbps, OVER=1 (Oversampling 8X) ≥ f_{Nom}(min) = 1.47 MHz
- Baud rate = 1 kbps, OVER=0 (Oversampling 16X) ≥ f_{Nom}(min) = 132 kHz
- Baud rate = 1 kbps, OVER=1 (Oversampling 8X) ≥ f_{Nom}(min) = 74 kHz

30.4.6.9 标识符奇偶校验

受保护的标识符由两个子字段组成：标识符和标识符奇偶校验。Bit5:0 分配给标识符，Bit7:6 分配给奇偶校验。

USART 接口可以生成、检查这些奇偶校验位，但也可以禁用此功能。用户可以通过 LIN 模式寄存器（USART_LINMR）的 PARDIS 位在两种模式之间进行选择：

- PARDIS = 0：
  - 在报文头传输期间，计算奇偶校验位，并将其与 LIN 标识符寄存器（USART_LINIR）的 IDCHR 字段的低 6 位一起发送，该寄存器的位 6 和 7 被丢弃。
  - 在报文头接收期间，检查标识符的奇偶校验位。如果奇偶校验位错误，则会发生标识符奇偶校验错误。只有 IDCHR 字段的低 6 位用接收到的标识符更新，Bit7:6 固定为 0。
- PARDIS = 1：
  - 在报文头传输期间，LIN 标识符寄存器（USART_LINIR）的 IDCHR 字段的所有位都在总线上发送。
  - 在报文头接收期间，IDCHR 字段的所有位都用接收到的标识符更新。

30.4.6.10 节点操作

在标识符的功能中，节点是否相关，与 LIN 响应有关。因此，在发送或接收标识符后，必须配置 USART。有三种配置：

- PUBLISH：节点发送响应
- SUBSCRIBE：节点接收响应
- IGNORE：节点与响应无关，它既不发送也不接收响应。

  由 USART_LINMR 寄存器中的节点操作（NACT）字段进行配置。比如：一个 LIN 群包含一个主机和两个从机：
- 数据从主机发送到从机 1 和从机 2：
  - NACT（主机）= PUBLISH
  - NACT（主机）= SUBSCRIBE
  - NACT（主机）= SUBSCRIBE
- 数据从主机发送数据到从机 1：
  - NACT（主机）= PUBLISH
  - NACT（主机）= SUBSCRIBE
  - NACT（主机）= IGNORE
- 数据从从机 1 发送到主机：
  - NACT（主机）= PUBLISH

- NACT（主机）= SUBSCRIBE
- NACT（主机）= IGNORE

- 数据从从机 1 发送到从机 2：
  - NACT（主机）= IGNORE
  - NACT（主机）= PUBLISH
  - NACT（主机）= SUBSCRIBE
- 数据从从机 2 发送到主机和从机 1：
  - NACT（主机）= SUBSCRIBE
  - NACT（主机）= SUBSCRIBE
  - NACT（主机）= PUBLISH

30.4.6.11 响应数据长度

LIN 响应数据长度是响应的数据字段数（字节），不包括校验和。

响应数据长度可以由用户配置，也可以由标识符的 Bit5:4 自动定义（与 LIN 规范 1.1 兼容）。用户可以通过 LIN 模式寄存器（USART_LINMR）的 DLM 位在这两种模式之间进行选择：

- DLM=0：响应数据长度由用户通过 USART_LINMR 的 DLC 字段进行配置。响应数据长度等于（DLC+1）个字节。DLC 设定范围从 0 到 255，因此响应可以包含 1 个数据字节到 256 个数据字节。
- DLM=1：响应数据长度由标识符（USART_LINIR 中的 IDCHR）根据下表定义。USART_LINMR 的 DLC 字段被丢弃。响应可以包含 2、4 或 8 个数据字节。

表 30-8：DLM=1 时的响应数据长度

IDCHR[5]	IDCHR[4]	响应数据长度（Bytes）
0	0	2
0	1	2
1	0	4
1	1	8

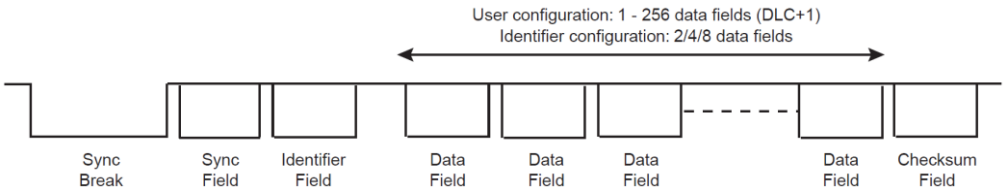


图 30-33：响应数据长度图

30.4.6.12 校验和

帧的最后一个字段是校验和。校验和包含带进位的反向 8 位和、所有数据字节或所有数据字节

以及受保护的标识符。仅计算数据字节的校验和称为经典校验和，用于与 LIN 1.3 从机通信。数据字节和受保护标识符字节的校验和计算称为增强校验和，用于与 LIN 2.0 从机通信。

USART 可以配置为：

- 自动发送、检查增强的校验和（CHKDIS = 0 & CHKTYP = 0）
- 自动发送、检查经典校验和（CHKDIS = 0 & CHKTYP = 1）
- 不发送、检查校验和（CHKDIS = 1）

此配置由LIN模式寄存器(USART_LINMR)的校验和类型(CHKTYP)和校验和禁用(CHKDIS)字段进行。

如果禁用了校验和功能，用户可以通过将校验和视为正常数据字节并在响应数据长度上添加 1 来手动发送它。

30.4.6.13 帧插槽模式

此模式仅对主节点有用。它遵守以下规则：每个帧插槽的长度应大于或等于 $t_{Frame\_maximum}$。

如果启用了帧时隙模式（FSDIS=0），并且帧传输已完成，则只有在 $t_{Frame\_Maximum}$ 后，才从帧开始重新置起 TXRDY 标志。因此，如果前一帧的帧时隙持续时间低于 $t_{Frame\_Maximum}$，则主节点无法发送新的标头。

如果禁用帧插槽模式（FSDIS=1），当帧传输完成，则 TXRDY 标志会立即置起。

$t_{Frame\_Maximum}$ 计算如下：

- 如果发送校验和（CHKDIS=0）：
  - $t_{Header\_Nominal} = 34 \times t_{bit}$
  - $t_{Response\_Nominal} = 10 \times (NData + 1) \times t_{bit}$
  - $t_{Frame\_Maximum} = 1.4 \times (t_{Header\_Nominal} + t_{Response\_Nominal} + 1)$ 注 1
  - $t_{Frame\_Maximum} = 1.4 \times (34 + 10 \times (DLC + 1 + 1) + 1) \times t_{bit}$
  - $t_{Frame\_Maximum} = (77 + 14 \times DLC) \times t_{bit}$
- 如果不发送校验和（CHKDIS=1）：
  - $t_{Header\_Nominal} = 34 \times t_{bit}$
  - $t_{Response\_Nominal} = 10 \times NData \times t_{bit}$
  - $t_{Frame\_Maximum} = 1.4 \times (t_{Header\_Nominal} + t_{Response\_Nominal} + 1)$ 注 1
  - $t_{Frame\_Maximum} = 1.4 \times (34 + 10 \times (DLC + 1) + 1) \times t_{bit}$
  - $t_{Frame\_Maximum} = (63 + 14 \times DLC) \times t_{bit}$

注：“+1”导致 $t_{Frame\_Maximum}$ 的结果为整数（LIN 规范 1.3）。

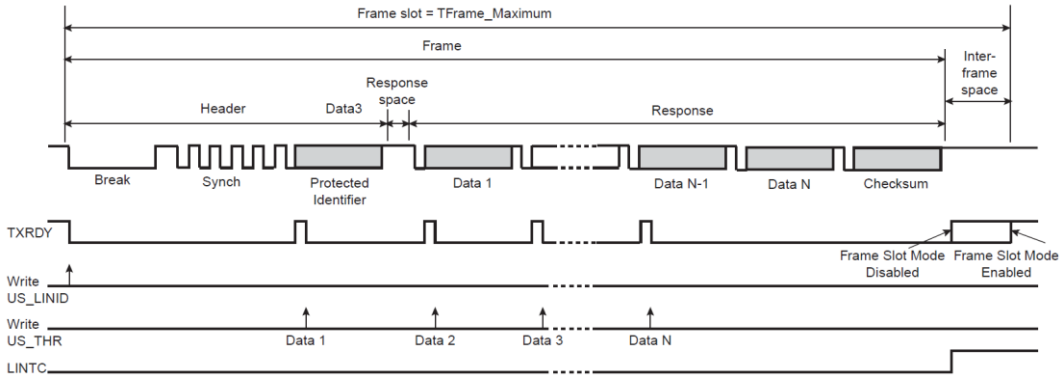


图 30-34：USART 帧插槽模式

30.4.7 LIN 错误

30.4.7.1 位错误

当 USART 正在传输时，如果 TX 线上传输的值与 RX 线上采样的值不同，则在从节点的主机配置中产生此错误。如果检测到位错误，则在下一个字节边界处中止传输。

此错误是通道状态寄存器（USART_CSR）中的 LINBE 标志。

30.4.7.2 同步域不一致错误

如果接收到的同步字段字符不是 0x55， 则在从节点配置中产生此错误。

此错误是通道状态寄存器（USART_CSR）中的 LINISFE 标志。

30.4.7.3 标识符奇偶校验错误

如果标识符的奇偶校验错误，则在从节点配置中会产生此错误。只有启用奇偶校验功能（PARDIS=0），才能生成此错误。

此错误是通道状态寄存器（USART_CSR）中的 LINIPE 标志。

30.4.7.4 校验和错误

如果接收到的校验和错误，则在从节点主的主机配置中会产生此错误。只有在启用校验和功能（CHKDIS=0）的情况下，此标志才会置位。

此错误是通道状态寄存器（USART_CSR）中的 LINCE 标志。

30.4.7.5 从机未响应错误

当 USART 期望来自其他节点（NACT=SUBSCRIBE）的响应，但在消息帧的最大长度

$t_{Frame\_maximum}$ 给定的时间内总线上没有出现有效的消息时，在从节点主机配置中会产生此错误。如果 USART 不期待任何消息（NACT=PUBLISH 或 NACT=IGNORE），则禁用此错误。

此错误是通道状态寄存器（USART_CSR）中的 LINSNRE 标志。

30.4.7.6 同步容差错误

如果在时钟同步程序之后，计算的波特率偏差与初始波特率相比大于最大公差 FTol_Unsynch（±15%），则从节点配置中会产生此错误。

此错误是通道状态寄存器（USART_CSR）中的 LINSTE 标志。

30.4.7.7 报文头超时错误

如果在报文头的最大长度 $t_{Header\_Maximum}$ 给定的时间内未完全接收到报文头，则在从节点配置中会产生此错误。

此错误是通道状态寄存器（USART_CSR）中的 LINHTE 标志。

30.4.8 LIN 帧处理

30.4.8.1 主节点配置

- 写 USART_CR 的 TXEN 和 RXEN，使能发送和接收
- 写 USART_MR 的 USART_MODE，以选择 LIN 模式和主节点
- 写 USART_BRGR 的 CD 和 FP 配置波特率
- 写 USART_LINMR 的 NACT、PARDIS、CHKDIS、CHKTYPE、DLCM、FSDIS 和 DLC 配置帧传输
- 检查 USART_CSR 中的 TXRDY 是否置位
- 写 USART_LINIR 的 IDCHR 发送报文头

  接下来的操作取决于 NACT 配置：
- Case1：NACT = PUBLISH，USART 发送响应
  1. 等待 USART_CSR 的 TXRDY 变高
  2. 写 USART_THR 的 TCHR 发送一个字节
  3. 如果尚未写入所有数据，重复执行上面两个步骤
  4. 等待 USART_CSR 的 LINTC 变高
  5. 检查 LIN 错误
- Case2：NACT = SUBSCRIBE，USART 接收响应
  1. 等待 USART_CSR 的 RXRDY 变高

2. 读 USART_RHR 的 RCHR
3. 如果尚未读取所有数据，重复执行上面两个步骤
4. 等待 USART_CSR 的 LINTC 变高
5. 检查 LIN 错误

- Case3：NACT = IGNORE，USART 不关注响应
  1. 等待 USART_CSR 的 LINTC 变高
  2. 检查 LIN 错误

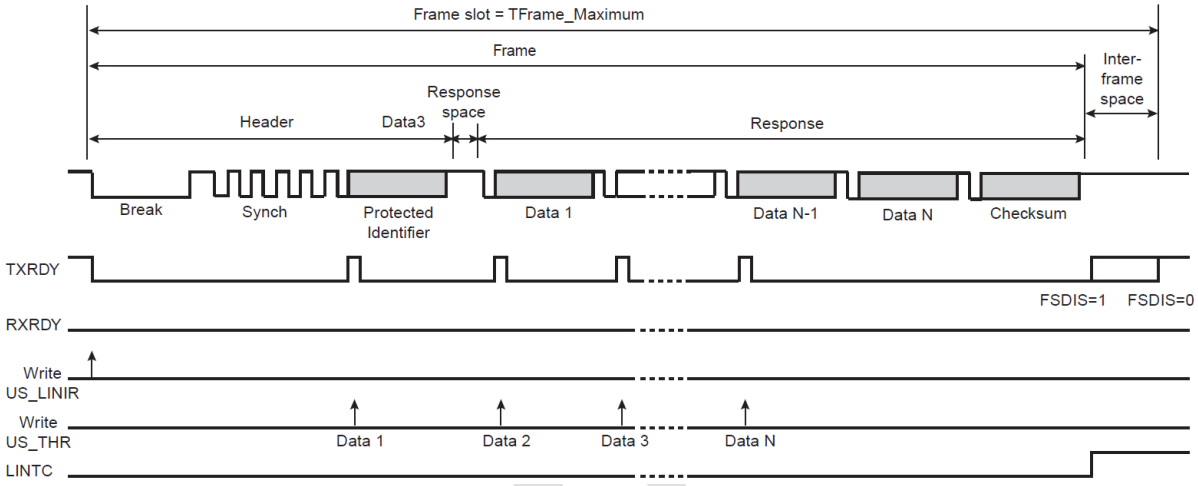


图 30-35：主节点配置，NACT = PUBLISH

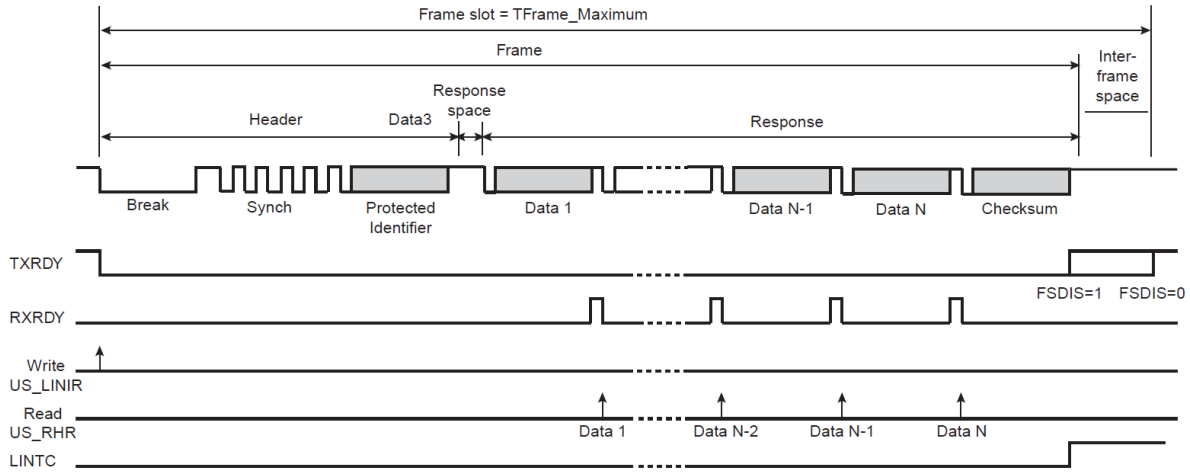


图 30-36：主节点配置，NACT = SUBSCRIBE

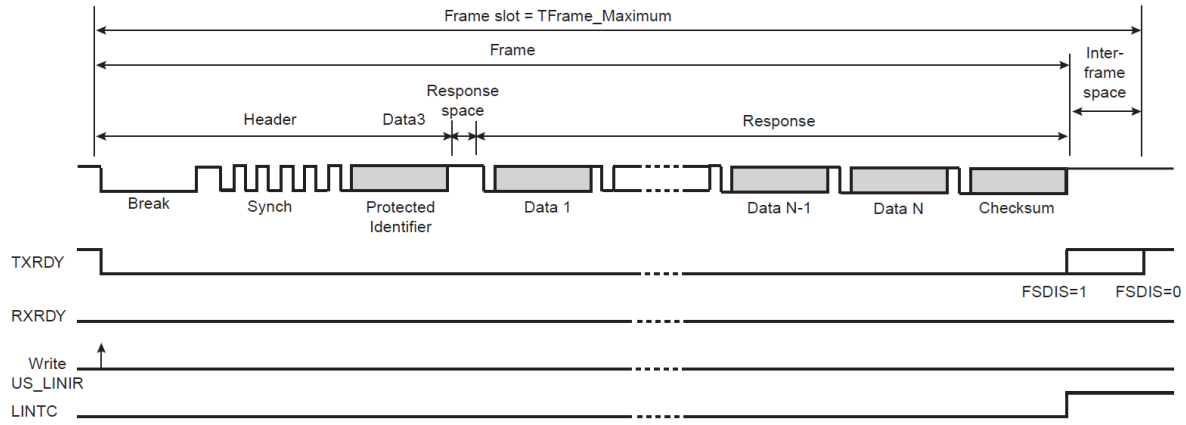


图 30-37：主节点配置，NACT = IGNORE

30.4.8.2 从节点配置

- 写 USART_CR 的 TXEN 和 RXEN，使能发送和接收
- 写 USART_MR 的 USART_MODE，以选择 LIN 模式和从节点
- 写 USART_BRGR 的 CD 和 FP 配置波特率。
- 等待 USART_CSR 的 LINID 变高
- 检查 LINISFE 和 LINPE 错误
- 读 USART_RHR 的 IDCHR
- 写 USART_LINMR 的 NACT、PARDIS、CHKDIS、CHKTYPE、DLCM 和 DLC 配置帧传输

重要：如果此帧的 NACT 配置为 PUBLISH，则即使此字段已正确配置，也必须设定 USART_LINMR 的 NACT=PUBLISH，以便设置 TXREADY 标志和相应的写传输请求。

接下来的操作取决于 NACT 配置：

- Case1：NACT = PUBLISH，USART 发送响应
  1. 等待 USART_CSR 的 TXRDY 变高
  2. 写 USART_THR 的 TCHR 发送一个字节
  3. 如果尚未写入所有数据，重复执行上面两个步骤
  4. 等待 USART_CSR 的 LINTC 变高
  5. 检查 LIN 错误
- Case2：NACT = SUBSCRIBE，USART 接收响应
  1. 等待 USART_CSR 的 RXRDY 变高
  2. 读 USART_RHR 的 RCHR
  3. 如果尚未读取所有数据，重复执行上面两个步骤
  4. 等待 USART_CSR 的 LINTC 变高
  5. 检查 LIN 错误

- Case3：NACT = IGNORE，USART 不关注响应
  1. 等待 USART_CSR 的 LINTC 变高
  2. 检查 LIN 错误

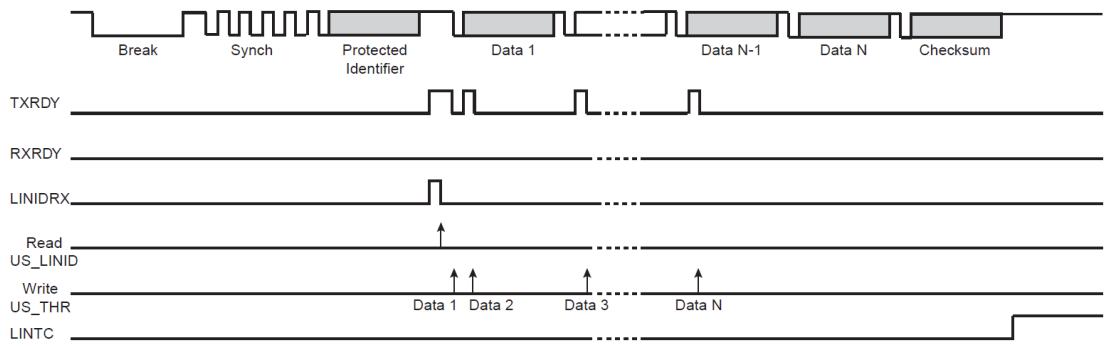


图 30-38：从节点配置，NACT = PUBLISH

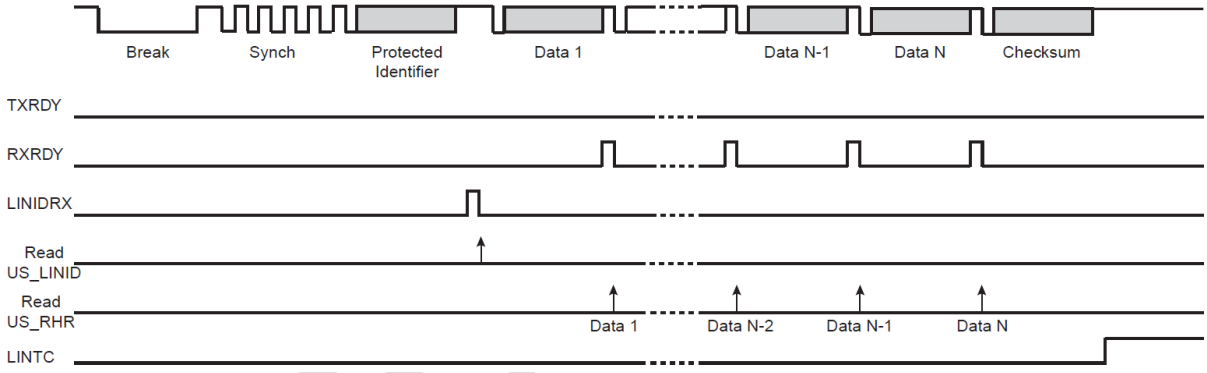


图 30-39：从节点配置，NACT = SUBSCRIBE

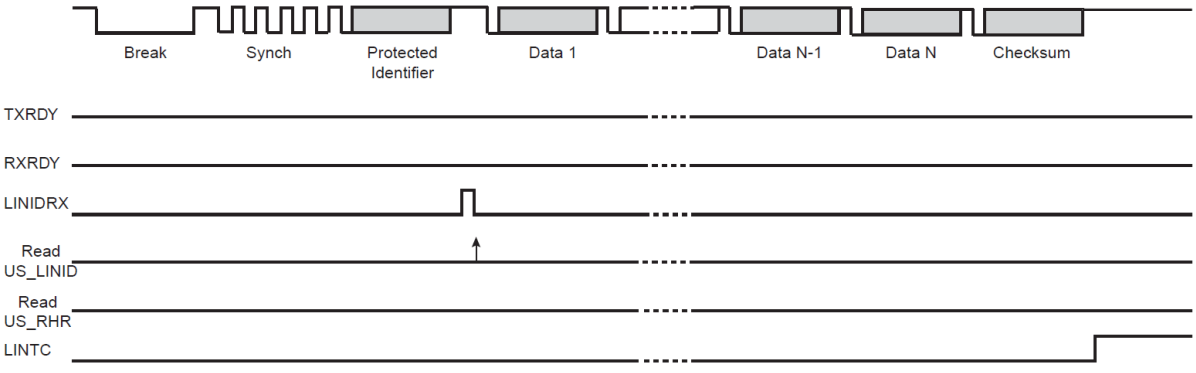


图 30-40：从节点配置，NACT = IGNORE

30.4.9 使用 DMA/PDC 处理 LIN 帧

USART 可与 DMA/PDC 结合使用，以便在无需任何处理器干预的情况下直接和片内、片外存

储器传输数据。

DMA/PDC 使用 TXRDY 和 RXRDY 触发标志写入或读取 USART。DMA/PDC 写入发送保持寄存器（USART_THR），读取接收保持寄存器（USART_RHR）。USART 中 DMA/PDC 写入或读取的数据大小为一个字节。

30.4.9.1 主节点配置

用户可以通过 LIN 模式寄存器（USART_LINMR）的 PDCM 位在两种 DMA/PDC 模式之间进行选择：

- PDCM=1：LIN 配置保存在写缓冲区中，由 DMA/PDC 写入传输保持寄存器 USART_THR（而不是 LIN 模式寄存器 USART_LINMR）。由于 DMA/PDC 传输大小限制为一个字节，因此传输被分为两次访问。在第一次访问期间，写入位 NACT、PARDIS、CHKDIS、CHKTYP、DLM 和 FSDIS。在第二次访问期间，写入 8 位 DLC 字段。
- PDCM=0：LIN 配置不保存在写入缓冲区中，必须由用户写 LIN 模式寄存器（USART_LINMR）。
  如果 USART 发送响应（NACT=PUBLISH），写缓冲区还包含标识符和数据。
  如果 USART 接收响应（NACT=SUBSCRIBE），读缓冲区包含数据。

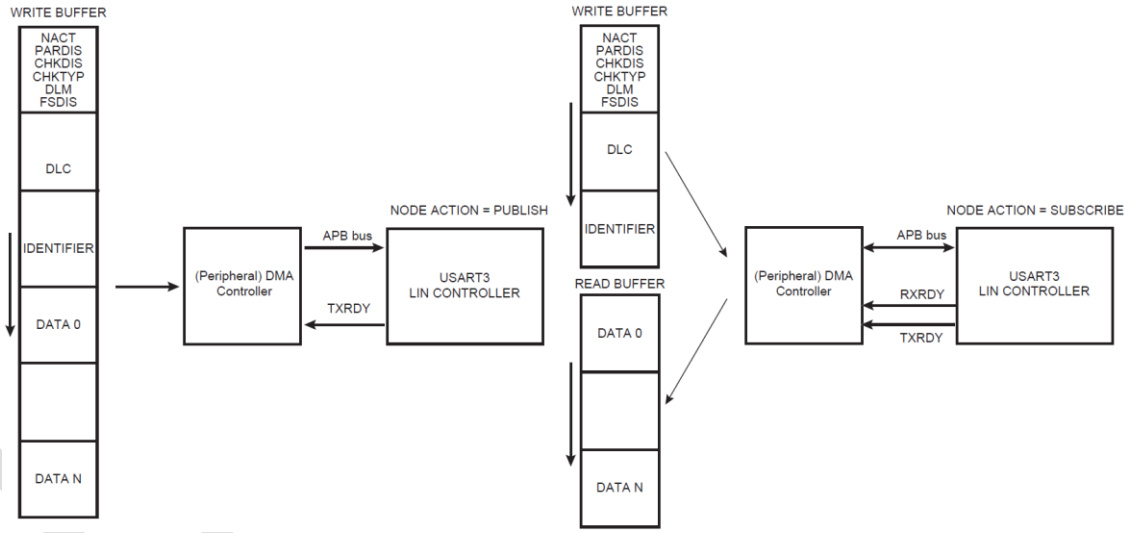


图 30-41：使用 DMA/PDC 的主节点（PDCM=1）

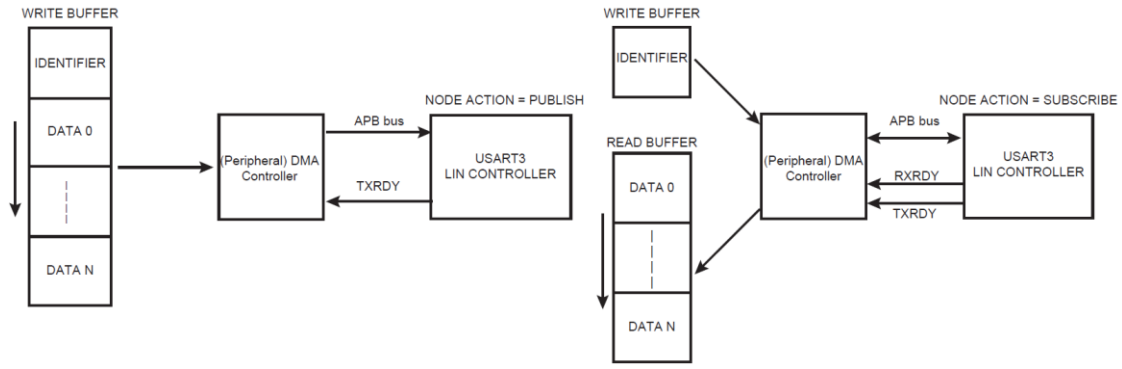


图 30-42：使用 DMA/PDC 的主节点（PDCM=0）

30.4.9.2 从节点配置

在此配置中，DMA/PDC 仅传输数据。标识符必须由用户读取 LIN 标识符寄存器（USART_LINIR）。LIN 模式必须由用户写入 LIN 模式寄存器（USART_LINMR）。

如果 USART 发送响应（NACT=PUBLISH），写缓冲区包含数据。

如果 USART 接收响应（NACT=SUBSCRIBE），读缓冲区包含数据。

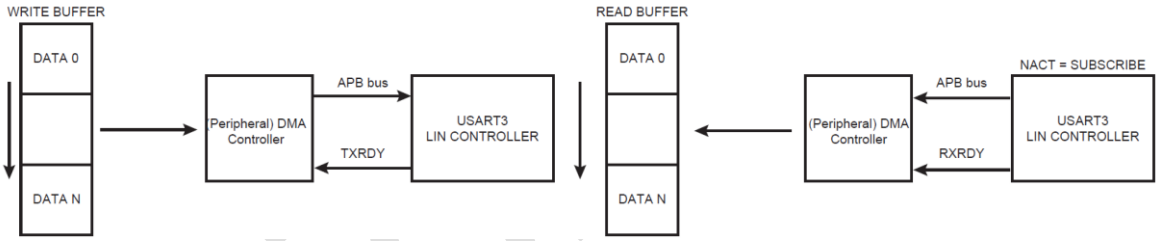


图 30-43：USART 从节点配置

30.4.9.3 唤醒请求

低功耗中的 LIN 群的任何节点都可以请求唤醒。

在 LIN 2.0 规范中，唤醒请求是通过将总线 Force 为 250μs 到 5 ms 的显性状态来发出的。为此，必须发送字符 0xF0，以便施加 5 个连续的显性位。无论波特率是多少，此字符期待指定的时序。

- Baud rate min = 1 kbps -> t_{bit} = 1 ms -> 5 t_{bit}s = 5 ms
- Baud rate max = 20 kbps -> t_{bit} = 50 μs -> 5 t_{bit}s = 250 μs

在 LIN 1.3 规范中，为了施加八个连续的显性位，应使用字符 0x80 生成唤醒请求。

用户可以通过 LIN 模式寄存器（USART_LINMR）中的 WKUPTYP 位选择发送 LIN 2.0 唤醒请求（WKUPTYP=0）或发送 LIN 1.3 唤醒请求（WKUPTYP=1）。

通过将控制寄存器（USART_CR）的 LINWKUP 位写 1 来传输唤醒请求。传输完成后，状态寄存器（USART_SR）的 LINTC 标志置位。通过将 USART_CR 的 RSTSTA 位写 1 来清除。

30.4.9.4 总线空闲超时

如果 LIN 总线在一定时间内处于非活动状态，则从节点应自动进入低功耗模式。在 LIN 2.0 规范中，超时固定为 4 秒。在 LIN 1.3 规范中，固定为 25000 t_{bit}s。

在从节点配置中，接收器超时检测 RXD 线路上的空闲状态。当检测到超时时，通道状态寄存器（USART_CSR）中的 TIMEOUT 位会变高，并会产生中断，从而指示驱动进入睡眠模式。

超时延迟周期（接收器等待接收新字符期间）在接收器超时寄存器（USART_RTOR）的 TO 字段中配置。如果 TO 写入 0，接收器超时被禁用且未检测到超时。USART_SR 中的 TIMEOUT 位保持为 0。否则，接收器加载一个带有 TO 设定值的 17 位计数器。该计数器在每个位周期递减，并在每次接收到新字符时重新加载。如果计数器达到 0，则 USART_CSR 中的 TIMEOUT 位变高。

如果执行 STTTO，计数器时钟将停止，直到收到第一个字符。

如果执行 RETTO，计数器立即开始从 TO 值递减。

30.4.10 测试模式

USART 可编程设置为三种不同的测试模式。内部回环可实现板上诊断。回环模式下，根据 USART 接口引脚不连接或连接分别配置为内部或外部回环。

30.4.10.1 普通模式

普通模式下将 RXD 引脚与接收器输入连接，而发送器输出与 TXD 引脚连接。

普通模式配置

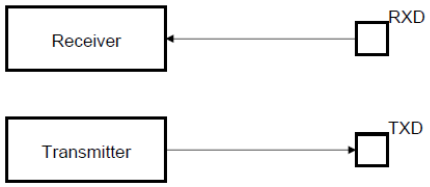


图 30-44：USART 普通模式

30.4.10.2 自动回应模式

自动回应模式允许一位一位重发。RXD 引脚收到一位后，将它发送到 TXD 引脚，如下图所示。对发送器编程不影响 TXD 引脚，RXD 引脚仍与接收器输入连接，因此接收器保持激活状态。

自动回应模式配置

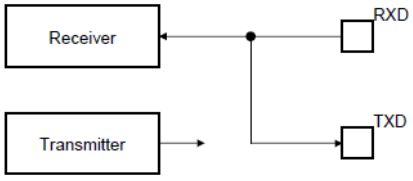


图 30-45：USART 自动回应模式

30.4.10.3 本地回环

本地回环模式下，发送器输出直接与接收器输入连接，如下图所示。TXD 与 RXD 引脚未使用。RXD 引脚对接收器无效，而 TXD 引脚与空闲状态一样，始终为高。

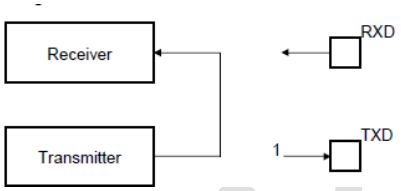


图 30-46：USART 本地环回模式

30.4.10.4 远程回环模式

远程回环模式下，直接将 RXD 引脚与 TXD 引脚连接，如下图所示。对发送器与接收器的禁止无效。该模式允许一位一位重传。

远程回环模式配置

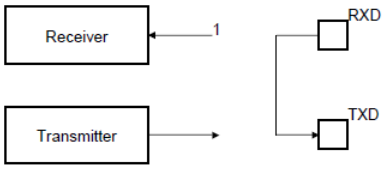


图 30-47：USART 远程环回模式

30.5 寄存器描述

USART6寄存器基地址：0x40B0_4000

USART7寄存器基地址：0x4700_C000

寄存器列表如下：

表 30-9：USART 寄存器列表

偏移地址	名称	描述
0x00	USART_CR	控制寄存器
0x04	USART_MR	模式寄存器
0x08	USART_IER	中断允许寄存器
0x0C	USART__IDR	中断禁止寄存器

偏移地址	名称	描述
0x10	USART_IMR	中断屏蔽寄存器
0x14	USART_CSR	通道状态寄存器
0x18	USART_RHR	接收器保持寄存器
0x1C	USART_THR	发送器保持寄存器
0x20	USART_BRGR	波特率发生器寄存器
0x24	USART_RTOR	接收器超时寄存器
0x28	USART_TTGR	发送器时间保障寄存器
0x2C~0x3C	-	保留
0x40	USART_FIDI	FI DI比率寄存器
0x44	-	保留
0x48	-	保留
0x4C	USART_IF	IrDA滤波寄存器
0x50	USART_MAN	曼彻斯特配置寄存器
0x54	USART_LINMR	LIN模式寄存器
0x58	USART_LINIR	LIN标识符寄存器
0x5C	USART_LINBRR	LIN波特率寄存器
0x60-0xE0	-	保留
0xE4	USART_WPMR	写保护模式寄存器
0xE8	USART_WPSR	写保护状态寄存器

30.5.1 USART 控制寄存器（USART_CR）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:22	RSV	-	-	保留
21	LINWKUP	W	0x0	发送LIN唤醒信号： 0：无效 1：向LIN总线发送唤醒信号
20	LINABT	W	0x0	中断LIN传输： 0：无效 1：中断当前LIN传输
19	RTSDIS	W	0x0	请求发送禁止： 0：无效 1：驱动RTS引脚为1
18	RTSEN	W	0x0	请求发送允许： 0：无效 1：驱动RTS引脚为0
17	DTRDIS	W	0x0	数据终端就绪禁止： 0：无效 1：驱动DTR引脚为0
16	DTREN	W	0x0	数据终端就绪启用： 0：无效 1：驱动DTR引脚为1

位	名称	属性	复位值	描述
15	RETTO	W	0x0	重载并启动超时： 0：无效 1：重启超时
14	RSTNACK	W	0x0	无应答复位： 0：无效 1：USART_CSR寄存器中的NACK复位
13	RSTIT	W	0x0	迭代复位： 0：无效 1：USART_CSR寄存器中ITERATION复位
12	SENDA	W	0x0	发送地址： 0：无效 1：只适用于多点模式，发送写入USART_THR的地址字符
11	STTTO	W	0x0	启动超时： 0：无效 1：在超时计数器计数前等待一个字符，复位USART_CSR中的状态位TIMEOUT
10	STPBRK	W	0x0	停止间断： 0：无效 1：在最少一字符时间且发送12位周期的高电平之后停止间断发送。若间断已发送则无效
9	STTBRK	W	0x0	启动间断： 0：无效 1：在USART_THR中有字符且发送移位寄存器中字符已发送后，开始发送间断。若间断已发送则无效
8	RSTSTA	W	0x0	状态位复位： 0：无效 1：USART_CSR寄存器中状态位PARE、FRAME、OVRE、MANERR和XBRK复位
7	TXDIS	W	0x0	发送器禁止： 0：无效 1：禁止发送器
6	TXEN	W	0x0	发送器允许： 0：无效 1：若TXDIS为0，允许发送器
5	RXDIS	W	0x0	接收器禁止： 0：无效 1：禁止接收器
4	RXEN	W	0x0	接收器允许： 0：无效 1：若RXDIS为0，允许接收器
3	RSTTX	W	0x0	发送器复位： 0：无效 1：发送器复位

位	名称	属性	复位值	描述
2	RSTRX	W	0x0	接收器复位： 0：无效 1：接收器复位
1:0	RSV	-	-	保留

30.5.2 USART 控制寄存器（USART_CR，SPI 模式）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:20	RSV	-	-	保留
19	RCS	W	0x0	取消SPI片选： 0：无效 1: 释放从机选择线NSS (RTS引脚)
18	FCS	W	0x0	强制SPI片选： 0：无效 1：即使USART没有发送数据，为了使SPI从机器件支持CSAAT（传输后片选激活）模式，强制从机选择线NSS（RTS引脚）为0
17:9	RSV	-	-	保留
8	RSTSTA	W	0x0	状态位复位： 0：无效 1：USART_CSR寄存器中状态位PARE、FRAME、OVRE、MANERR和XBRK复位
7	TXDIS	W	0x0	发送器禁止： 0：无效 1：禁止发送器
6	TXEN	W	0x0	发送器允许： 0：无效 1：若TXDIS为0，允许发送器
5	RXDIS	W	0x0	接收器禁止： 0：无效 1：禁止接收器
4	RXEN	W	0x0	接收器允许： 0：无效 1：若RXDIS为0，允许接收器
3	RSTTX	W	0x0	发送器复位： 0：无效 1：发送器复位
2	RSTRX	W	0x0	接收器复位： 0：无效 1：接收器复位
1:0	RSV	-	-	保留

30.5.3 USART 模式寄存器（USART_MR）

偏移地址：0x04

复位值：0xC000 0000

位	名称	属性	复位值	描述
31	ONEBIT	R/W	0x1	帧首定界符选择器： 0：帧首定界符是COMMAND或DATASYNC 1：帧首定界符是一个位
30	MODSYNC	R/W	0x1	曼彻斯特同步模式： 0：曼彻斯特起始位是0到1的电平转换 1：曼彻斯特起始位是1到0的电平转换
29	MAN	R/W	0x0	曼彻斯特编码器/解码器允许： 0：禁止曼彻斯特编码器/解码器 1：曼彻斯特编码器/解码器允许
28	FILTER	R/W	0x0	红外接收线滤波器： 0：USART不对接收线滤波 1:USART使用3点采样滤波器（1/16位时钟）（2/3多）对接收线滤波
27:24	RSV	-	-	保留
23	INVDATA	R/W	0x0	数据翻转： 0:TXD线上发送的数据域与往USART_THR寄存器里面写数据一样，或从USART_RHR寄存器读取的内容和RXD线接收的数据一样。普通模式操作 1：和写入USART_THR寄存器里面的值相比，TXD线上传输的数据被翻转（仅仅是电压极性）；或和RXD线接收到数据相比，从USART_RHR寄存器读取的值被翻转。翻转模式操作，在非接触式智能卡应用中很有用，可通过MSBF位进行配置
22	VAR_SYNC	R/W	0x0	命令同步因子/数据同步帧首定界符： 0：用户定义的命令配置或依据SYNC值确定的数据同步域 1:当有字符写入USART_THR寄存器时更新同步域
21	DSNACK	R/W	0x0	禁止连续NACK： 0：一旦收到的字符出现检验错误，NACK发送到ISO线上（除非INACK置位） 1：当连续检验错误数未达到MAX_ITERATION域给出的值，这些检验错误在ISO线上产生NACK。一旦达到该值，ISO线上不再发送另外的NACK，ITERATION标志被设置

位	名称	属性	复位值	描述
20	INACK	R/W	0x0	抑制无应答： 0：产生NACK 1：不产生NACK。注意：在SPI主机模式，若INACK为0，当有字符被写入USART_THR寄存器（假设TXRDY已置位），则立即发送字符开始位。当INACK为1时，还必须满足另一个条件，即当有字符被写入USART_THR寄存器且仅当RXRDY标志被清0（通过读接收器保持寄存器实现）时，才立即发送字符
19	OVER	R/W	0x0	过采样模式： 0：16倍过采样 1：8倍过采样
18	CLKO	R/W	0x0	时钟输出选择： 0：USART不驱动SCK引脚 1：若USCLKS未选择外部时钟SCK，USART驱动SCK引脚
17	MODE9	R/W	0x0	字符长度9位： 0：CHRL定义字符长度 1：字符长度为9位
16	MSBF	R/W	0x0	位顺序： 0：首先发送/接收最低有效位 1：首先发送/接收最高有效位
15:14	CHMODE	R/W	0x0	通道模式： 00：普通模式 01：自动回应，接收器输入与TXD引脚连接 10：本地回环，发送器输出与接收器输入连接 11：远程回环，RXD引脚与TXD引脚连接
13:12	NBSTOP	R/W	0x0	停止位数： 异步（SYNC=0） 00：1个 01：1.5个 10：2个 11：保留 同步（SYNC=1） 00：1个 01：保留 10：2个 11：保留
11:9	PAR	R/W	0x0	校验类型： 000：偶校验 001：奇校验 010：检验强制为0（空间） 011：检验强制为1（标志） 10x：无校验 11x：多点模式
8	SYNC	R/W	0x0	同步模式选择： 0：USART工作在异步模式下 1：USART工作在同步模式下

位	名称	属性	复位值	描述
7:6	CHRL	R/W	0x0	字符长度： 00：5位 01：6位 10：7位 11：8位
5:4	USCLKS	R/W	0x0	时钟选择： 00：MCK 01：MCK/DIV(DIV=8) 10：保留 11：SCK
3:0	MODE	R/W	0x0	工作模式： 0000：普通 0001：RS485 0010：硬件握手 0011：调制解调器 0100：保留 0110：保留 1000：IrDA 1001：保留 1010：LIN主机 1011：LIN从机 1110：SPI主机 1111：SPI从机 其他：保留

30.5.4 USART 模式寄存器（USART_MR，SPI 模式）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:21	RSV	-	-	保留
20	WRDBT	R/W	0x0	传输前等待读取数据： 0：字符写入USART_THR寄存器后，字符传输立即开始（假设设置了TXRDY） 1：只有在清除RXRDY标志（已读取USART_RHR）的情况下，字符传输才会在写入字符时开始。
19	RSV	-	-	保留
18	CLKO	R/W	0x0	时钟输出选择： 0：USART不驱动SCK引脚 1：若USCLKS未选择外部时钟SCK，USART驱动SCK引脚
17	RSV	-	-	保留
16	CPOL	R/W	0x0	SPI时钟极性： 0: SPCK无效状态值是逻辑低电平0 1: SPCK无效状态值是逻辑高电平1

位	名称	属性	复位值	描述
15:9	RSV	-	-	保留
8	CPHA	R/W	0x0	SPI时钟相位： 0：USART工作在异步模式下 1：USART工作在同步模式下
7:6	CHRL	R/W	0x0	字符长度： 11：8位 其他：保留
5:4	USCLKS	R/W	0x0	时钟选择： 00：MCK 01：MCK/DIV 10：保留 11：SCK DIV的设定： USART6：CFGR1[27:24] USART7：CFGR1[31:28]
3:0	MODE	R/W	0x0	工作模式： 1110：SPI主机 1111：SPI从机 其他：保留

30.5.5 USART 中断使能寄存器（USART_IER）

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:21	RSV	-	-	保留
20	MANE	W	0x0	曼彻斯特错误中断使能
19	CTSIC	W	0x0	发送输入变化清除中断使能
18	DCDIC	W	0x0	数据载体检测输入变化中断使能
17	DSRIC	W	0x0	数据就绪输入更改中断使能
16	RIIC	W	0x0	环形指示器输入更改中断使能
15:14	RSV	-	-	保留
13	NACK	W	0x0	无应答中断使能
12	RXBUFF	W	0x0	缓冲器满中断使能
11	TXBUFE	W	0x0	缓冲器空中断使能
10	ITER	W	0x0	达到最大重复次数的中断使能
9	TXEMPTY	W	0x0	TXEMPTY中断使能
8	TIMEOUT	W	0x0	超时中断使能
7	PARE	W	0x0	奇偶错误中断使能
6	FRAME	W	0x0	帧错误中断使能
5	OVRE	W	0x0	溢出错误中断使能
4	ENDTX	W	0x0	发送结束中断使能
3	ENDRX	W	0x0	接收结束中断使能
2	RXBRK	W	0x0	接收器间断中断使能
1	TXRDY	W	0x0	TXRDY中断使能
0	RXRDY	W	0x0	RXRDY中断使能

30.5.6 USART 中断使能寄存器（USART_IER，SPI 模式）

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12	RXBUFF	W	0x0	缓冲器满中断使能
11	TXBUFE	W	0x0	缓冲器空中断使能
10	UNRE	W	0x0	SPI欠载错误中断使能
9	TXEMPTY	W	0x0	TXEMPTY中断使能
8:6	RSV	-	-	保留
5	OVRE	W	0x0	溢出错误中断使能
4	ENDTX	W	0x0	发送结束中断使能
3	ENDRX	W	0x0	接收结束中断使能
2	RSV	-	-	保留
1	TXRDY	W	0x0	TXRDY中断使能
0	RXRDY	W	0x0	RXRDY中断使能

30.5.7 USART 中断使能寄存器（USART_IER，LIN 模式）

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31	LINHTE	W	0x0	LIN报文头超时错误中断使能
30	LINSTE	W	0x0	LIN同步容差错误中断使能
29	LINSNRE	W	0x0	LIN从机不响应错误中断使能
28	LINCE	W	0x0	LIN校验和错误中断使能
27	LINIPE	W	0x0	LIN标识符奇偶校验中断使能
26	LINISFE	W	0x0	LIN同步域不一致错误中断使能
25	LINBE	W	0x0	LIN总线错误中断使能
24:16	RSV	-	-	保留
15	LINTC	W	0x0	LIN传输完成中断使能
14	LINID	W	0x0	发送或接收LIN标识符中断使能
13	LINBK	W	0x0	发送或接收LIN间断中断使能
12	RXBUFF	W	0x0	缓冲器满中断使能
11	TXBUFE	W	0x0	缓冲器空中断使能
10	RSV	-	-	保留
9	TXEMPTY	W	0x0	TXEMPTY中断使能
8	TIMEOUT	W	0x0	超时中断使能
7	PARE	W	0x0	奇偶错误中断使能
6	FRAME	W	0x0	帧错误中断使能
5	OVRE	W	0x0	溢出错误中断使能
4	ENDTX	W	0x0	发送结束中断使能
3	ENDRX	W	0x0	接收结束中断使能
2	RSV	-	-	保留

位	名称	属性	复位值	描述
1	TXRDY	W	0x0	TXRDY中断使能
0	RXRDY	W	0x0	RXRDY中断使能

30.5.8 USART 中断禁止寄存器（USART_IDR）

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:21	RSV	-	-	保留
20	MANE	W	0x0	曼彻斯特错误中断禁止
19	CTSIC	W	0x0	发送输入变化清除中断禁止
18	DCDIC	W	0x0	数据载体检测输入变化中断禁止
17	DSRIC	W	0x0	数据就绪输入更改中断禁止
16	RIIC	W	0x0	环形指示器输入更改中断禁止
15:14	RSV	-	-	保留
13	NACK	W	0x0	无应答中断禁止
12	RXBUFF	W	0x0	缓冲器满中断禁止
11	TXBUFE	W	0x0	缓冲器空中断禁止
10	ITER	W	0x0	达到最大重复次数的中断禁止
9	TXEMPTY	W	0x0	TXEMPTY中断禁止
8	TIMEOUT	W	0x0	超时中断禁止
7	PARE	W	0x0	奇偶错误中断禁止
6	FRAME	W	0x0	帧错误中断禁止
5	OVRE	W	0x0	溢出错误中断禁止
4	ENDTX	W	0x0	发送结束中断禁止
3	ENDRX	W	0x0	接收结束中断禁止
2	RXBRK	W	0x0	接收器间断中断禁止
1	TXRDY	W	0x0	TXRDY中断禁止
0	RXRDY	W	0x0	RXRDY中断禁止

30.5.9 USART 中断禁止寄存器（USART_IDR，SPI 模式）

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12	RXBUFF	W	0x0	缓冲器满中断禁止
11	TXBUFE	W	0x0	缓冲器空中断禁止
10	UNRE	W	0x0	SPI欠载错误中断禁止
9	TXEMPTY	W	0x0	TXEMPTY中断禁止
8:6	RSV	-	-	保留
5	OVRE	W	0x0	溢出错误中断禁止

位	名称	属性	复位值	描述
4	ENDTX	W	0x0	发送结束中断禁止
3	ENDRX	W	0x0	接收结束中断禁止
2	RSV	-	-	保留
1	TXRDY	W	0x0	TXRDY中断禁止
0	RXRDY	W	0x0	RXRDY中断禁止

30.5.10 USART 中断禁止寄存器（USART_IDR，LIN 模式）

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31	LINHTE	W	0x0	LIN报文头超时错误中断禁止
30	LINSTE	W	0x0	LIN同步容差错误中断禁止
29	LINSNRE	W	0x0	LIN从机不响应错误中断禁止
28	LINCE	W	0x0	LIN校验和错误中断禁止
27	LINIPE	W	0x0	LIN标识符奇偶校验中断禁止
26	LINISFE	W	0x0	LIN同步域不一致错误中断禁止
25	LINBE	W	0x0	LIN总线错误中断禁止
24:16	RSV	-	-	保留
15	LINTC	W	0x0	LIN传输完成中断禁止
14	LINID	W	0x0	发送或接收LIN标识符中断禁止
13	LINBK	W	0x0	发送或接收LIN间断中断禁止
12	RXBUFF	W	0x0	缓冲器满中断禁止
11	TXBUFE	W	0x0	缓冲器空中断禁止
10	RSV	-	-	保留
9	TXEMPTY	W	0x0	TXEMPTY中断禁止
8	TIMEOUT	W	0x0	超时中断禁止
7	PARE	W	0x0	奇偶错误中断禁止
6	FRAME	W	0x0	帧错误中断禁止
5	OVRE	W	0x0	溢出错误中断禁止
4	ENDTX	W	0x0	发送结束中断禁止
3	ENDRX	W	0x0	接收结束中断禁止
2	RSV	-	-	保留
1	TXRDY	W	0x0	TXRDY中断禁止
0	RXRDY	W	0x0	RXRDY中断禁止

30.5.11 USART 中断屏蔽寄存器（USART_IMR）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:25	RSV	-	-	保留

位	名称	属性	复位值	描述
24:21	RSV	-	-	保留
20	MANE	R	0x0	曼彻斯特错误中断屏蔽
19	CTSIC	R	0x0	发送输入变化清除中断屏蔽
18	DCDIC	R	0x0	数据载体检测输入变化中断屏蔽
17	DSRIC	R	0x0	数据就绪输入更改中断屏蔽
16	RIIC	R	0x0	环形指示器输入更改中断屏蔽
15:14	RSV	-	-	保留
13	NACK	R	0x0	无应答中断屏蔽
12	RXBUFF	R	0x0	缓冲器满中断屏蔽
11	TXBUFE	R	0x0	缓冲器空中断屏蔽
10	ITER	R	0x0	达到最大重复次数的中断屏蔽
9	TXEMPTY	R	0x0	TXEMPTY中断屏蔽
8	TIMEOUT	R	0x0	超时中断屏蔽
7	PARE	R	0x0	奇偶错误中断屏蔽
6	FRAME	R	0x0	帧错误中断屏蔽
5	OVRE	R	0x0	溢出错误中断屏蔽
4	ENDTX	R	0x0	发送结束中断屏蔽
3	ENDRX	R	0x0	接收结束中断屏蔽
2	RXBRK	R	0x0	接收器间断中断屏蔽
1	TXRDY	R	0x0	TXRDY中断屏蔽
0	RXRDY	R	0x0	RXRDY中断屏蔽

30.5.12 USART 中断屏蔽寄存器（USART_IMR，SPI 模式）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12	RXBUFF	R	0x0	缓冲器满中断屏蔽
11	TXBUFE	R	0x0	缓冲器空中断屏蔽
10	UNRE	R	0x0	SPI欠载错误中断屏蔽
9	TXEMPTY	R	0x0	TXEMPTY中断屏蔽
8:6	RSV	-	-	保留
5	OVRE	R	0x0	溢出错误中断屏蔽
4	ENDTX	R	0x0	发送结束中断屏蔽
3	ENDRX	R	0x0	接收结束中断屏蔽
2	RSV	-	-	保留
1	TXRDY	R	0x0	TXRDY中断屏蔽
0	RXRDY	R	0x0	RXRDY中断屏蔽

30.5.13 USART 中断屏蔽寄存器（USART_IMR，LIN 模式）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31	LINHTE	R	0x0	LIN报文头超时错误中断屏蔽
30	LINSTE	R	0x0	LIN同步容差错误中断屏蔽
29	LINSNRE	R	0x0	LIN从机不响应错误中断屏蔽
28	LINCE	R	0x0	LIN校验和错误中断屏蔽
27	LINIPE	R	0x0	LIN标识符奇偶校验中断屏蔽
26	LINISFE	R	0x0	LIN同步域不一致错误中断屏蔽
25	LINBE	R	0x0	LIN总线错误中断屏蔽
24:16	RSV	-	-	保留
15	LINTC	R	0x0	LIN传输完成中断屏蔽
14	LINID	R	0x0	发送或接收LIN标识符中断屏蔽
13	LINBK	R	0x0	发送或接收LIN间断中断屏蔽
12	RXBUFF	R	0x0	缓冲器满中断屏蔽
11	TXBUFE	R	0x0	缓冲器空中断屏蔽
10	RSV	-	-	保留
9	TXEMPTY	R	0x0	TXEMPTY中断屏蔽
8	TIMEOUT	R	0x0	超时中断屏蔽
7	PARE	R	0x0	奇偶错误中断屏蔽
6	FRAME	R	0x0	帧错误中断屏蔽
5	OVRE	R	0x0	溢出错误中断屏蔽
4	ENDTX	R	0x0	发送结束中断屏蔽
3	ENDRX	R	0x0	接收结束中断屏蔽
2	RSV	-	-	保留
1	TXRDY	R	0x0	TXRDY中断屏蔽
0	RXRDY	R	0x0	RXRDY中断屏蔽

30.5.14 USART 通道状态寄存器（USART_CSR）

偏移地址：0x14

复位值：0x0078 0000

位	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	MANERR	R	0x0	曼彻斯特错误： 0：上次RSTSTA后未检测到曼彻斯特错误 1：上次RSTSTA后至少检测到一个曼彻斯特错误
23	CTS	R	0x0	CTS输入镜像： 0：CTS为0 1：CTS为1
22:20	RSV	-	-	保留
19	CTSIC	R	0x1	发送输入变化清除标志： 0：上次读USART_CSR后在CTS引脚上未检测到输入变化 1：上次读USART_CSR后在CTS引脚上至少检测到一次输入变化
18:14	RSV	-	-	保留

位	名称	属性	复位值	描述
13	NACK	R	0x0	无应答； 0：上次RSTNACK后未检测到无应答 1：上次RSTNACK后至少检测到一次无应答
12:11	RSV	-	-	保留
10	ITER	R	0x0	达到最大重复次数： 0：上次RSTSTA后没有达到最大重复次数 1：上次RSTSTA后达到最大重复次数
9	TXEMPTY	R	0x0	发送器空： 0：USART_THR与发送移位寄存器中至少有一个字符，或发送器被禁止 1：USART_THR与发送移位寄存器中均无字符
8	TIMEOUT	R	0x0	接收器超时： 0：上次启动超时命令STTTO后无超时或超时寄存器为0 1：上次启动超时命令STTTO后有超时
7	PARE	R	0x0	检验错误： 0：上次RSTSTA后未检测到检验错误 1：上次RSTSTA后至少检测到一次检验错误
6	FRAME	R	0x0	帧错误： 0：上次RSTSTA后未检测到停止位 1：上次RSTSTA后至少检测到一个停止位
5	OVRE	R	0x0	溢出错误： 0：上次RSTSTA后未出现溢出错误 1：上次RSTSTA后至少出现一次溢出错误
4:3	RSV	-	-	保留
2	RXBRK	R	0x0	间断接收/间断结束： 0：上次RSTSTA后未检测到间断接收或间断结束 1：上次RSTSTA后检测到间断接收或间断结束
1	TXRDY	R	0x0	发送器就绪： 0：USART_THR中字符等待发送到发送移位寄存器，或请求STTBRK命令，或发送器被禁止。一旦发送器允许，TXRDY变为1。 1：USART_THR中无字符
0	RXRDY	R	0x0	接收器就绪： 0：上次读USART_RHR后未接收到完整的字符或接收器禁止。若接收器被禁止时正在接收字符，当接收器允许时RXRDY变为1 1：上次读USART_RHR后至少收到一个完整的字符，且USART_RHR未被读取

30.5.15 USART 通道状态寄存器（USART_CSR，SPI 模式）

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10	UNRE	R	0x0	SPI欠载错误： 0：自上次RSTSTA以来，未发生SPI欠载错误 1：自上次RSTSTA以来，至少发生一次SPI欠载错误
9	TXEMPTY	R	0x0	发送器空： 0：USART_THR与发送移位寄存器中至少有一个字符，或发送器被禁止 1：USART_THR与发送移位寄存器中均无字符
8:6	RSV	-	-	保留
5	OVRE	R	0x0	溢出错误： 0：上次RSTSTA后未出现溢出错误 1：上次RSTSTA后至少出现一次溢出错误
4:2	RSV	-	-	保留
1	TXRDY	R	0x0	发送器就绪： 0：USART_THR中字符等待发送到发送移位寄存器，或请求STTBRK命令，或发送器被禁止。一旦发送器允许，TXRDY变为1 1：USART_THR中无字符
0	RXRDY	R	0x0	接收器就绪： 0：上次读USART_RHR后未接收到完整的字符或接收器禁止。若接收器被禁止时正在接收字符，当接收器允许时RXRDY变为1 1：上次读USART_RHR后至少收到一个完整的字符，且USART_RHR未被读取

30.5.16 USART 通道状态寄存器（USART_CSR，LIN 模式）

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31	LINHTE	R	0x0	LIN报文头超时错误： 0：自上次RSTSTA以来，未检测到LIN报文头超时错误 1：自上次RSTSTA以来，检测到LIN报文头超时错误

位	名称	属性	复位值	描述
30	LINSTE	R	0x0	LIN同步容差错误： 0：自上次RSTSTA以来，未检测到LIN同步容差错误 1：自上次RSTSTA以来，检测到LIN同步容差错误
29	LINSNRE	R	0x0	LIN从机不响应错误： 0：自上次RSTSTA以来，未检测到LIN从机无响应错误 1：自上次RSTSTA以来，检测到LIN从机无响应错误
28	LINCE	R	0x0	LIN校验和错误： 0：自上次RSTSTA以来，未检测到校验和错误 1：自上次RSTSTA以来，检测校验和错误
27	LINIPE	R	0x0	LIN标识符奇偶校验： 0：自上次RSTSTA以来，未检测到标识符奇偶校验错误 1：自上次RSTSTA以来，检测标识符奇偶校验错误
26	LINISFE	R	0x0	LIN同步域不一致错误： 0：自上次RSTSTA以来，未检测到同步域不一致错误 1：USART被配置位从机节点，自上次RSTSTA以来，检测到同步域不一致错误
25	LINBE	R	0x0	LIN位错误： 0：自上次RSTSTA以来，未检测到位错误 1：自上次RSTSTA以来，检测到位错误
24	RSV	-	-	保留
23	LINBLS	R	0x0	LIN总线状态： 0：LIN总线为0 1：LIN总线为1
22:16	RSV	-	-	保留
15	LINTC	R	0x0	LIN传输完成： 0：USART空闲或LIN传输正在进行 1：自上次RSTSTA以来，LIN传输已完成
14	LINID	R	0x0	发送或接收LIN标识符： LIN主机 0：自上次RSTSTA以来未发送LIN标识符 1：自上次RSTSTA以来，至少发送了一个LIN标识符 LIN从机 0：自上次RSTSTA以来未接收LIN标识符 1：自上次RSTSTA以来，至少接收了一个LIN标识符

位	名称	属性	复位值	描述
13	LINBK	R	0x0	发送或接收LIN间断： LIN主机 0：自上次RSTSTA以来未发送LIN间断 1：自上次RSTSTA以来，至少发送了一个LIN间断 LIN从机 0：自上次RSTSTA以来未接收LIN间断 1：自上次RSTSTA以来，至少接收了一个LIN间断
12:10	RSV	-	-	保留
9	TXEMPTY	R	0x0	发送器空： 0：USART_THR或移位寄存器中有字符，或者发送器被禁用 1：USART_THR或移位寄存器中没有字符
8	TIMEOUT	R	0x0	超时： 0：自上次启动超时命令（USART_CR中的STTTO）以来，没有超时，或超时寄存器为0 1：自上次启动超时命令（USART_CR中的STTTO）以来，出现超时
7	PARE	R	0x0	奇偶错误： 0：自上次RSTSTA以来未发生奇偶错误 1：自上次RSTSTA以来，至少发生了一个奇偶错误
6	FRAME	R	0x0	帧错误： 0：自上次RSTSTA以来未发生帧错误 1：自上次RSTSTA以来，至少发生了一个帧错误
5	OVRE	R	0x0	溢出错误： 0：自上次RSTSTA以来未发生溢出错误 1：自上次RSTSTA以来，至少发生了一个溢出错误
4:2	RSV	-	-	保留
1	TXRDY	R	0x0	发送器就绪： 0：USART_THR中字符等待发送到发送移位寄存器，或请求STTBRK命令，或发送器被禁止。一旦发送器允许，TXRDY变为1 1：USART_THR中无字符
0	RXRDY	R	0x0	接收器就绪； 0：上次读USART_RHR后未接收到完整的字符或接收器禁止。若接收器被禁止时正在接收字符，当接收器允许时RXRDY变为1 1：上次读USART_RHR后至少收到一个完整的字符，且USART_RHR未被读取

30.5.17 USART 接收保持寄存器（USART_RHR）

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	RXSYNH	R	0x0	接收到同步域: 0：上一个接收到的字符是数据 1：上一个接收到的字符是命令
14:9	RSV	-	-	保留
8:0	RXCHR	R	0x0	收到的字符 若RXRDY置位，为接收到的最后一个字符

30.5.18 USART 发送保持寄存器（USART_THR）

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	TXSYNH	W	0x0	发送的同步域; 0：下一个待发送的字符被当作数据进行编码，帧起始定界符是DATA SYNC 1：下一个待发送的字符被当作命令进行编码，帧起始定界符是COMMAND SYNC
14:9	RSV	-	-	保留
8:0	TXCHR	W	0x0	将要发送的字符 如果TXRDY未置位，则在当前字符之后传输下一个字符

30.5.19 USART 波特率发生器寄存器（USART_BRGR）

偏移地址：0x20

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:19	RSV	-	-	保留
18:16	FP	R/W	0x0	小数部分： 0：小数分频器被禁止 1-7：波特率分辨率，定义为FP x 1/8
15:0	CD	R/W	0x0	时钟分频 参考下表

表 30-10：USART 时钟分频参考表

CD	SYNC = 0		SYNC = 1或 USART_MODE = SPI (Master或 Slave)
	OVER = 0	OVER = 1	
0	禁用波特率时钟		
1到 65535	波特率 =选定时钟/16/CD	波特率 =选定时钟/8/CD	波特率=选定时钟/CD

30.5.20 USART 接收超时寄存器（USART_RTOR）

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16:0	TO	R/W	0x0	超时值： 0：禁用接收器超时 1 – 131071：接收器超时使能且超时延迟为TO个位周期

30.5.21 USART 发送时间保护寄存器（USART_TTGR）

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	TG	R/W	0x0	时间保护值： 0：禁止发送器时间保护 1-255：允许发送器时间保护，时间保护延迟为TG个位周期

30.5.22 USART FIDI 比率寄存器（USART_FIDI）

偏移地址：0x40

复位值：0x0000 0174

位	名称	属性	复位值	描述
31:11	RSV	-	-	保留
15:0	FI_DI_RATIO	R/W	0x174	FI与DI比值

30.5.23 USART IrDA 滤波寄存器（USART_IF）

偏移地址：0x4C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	IrDA_FILTER	R/W	0x0	IrDA滤波器 IrDA滤波器的值需满足以下标准： t_{MCK} * (IRDA_FILTER + 3) < 1.41 μs

30.5.24 USART 曼彻斯特配置寄存器（USART_MAN）

偏移地址：0x50

复位值：0xB001 1004

位	名称	属性	复位值	描述
31	RSV	-	-	保留
30	DRIFT	R/W	0x0	漂移补偿： 0：该USART不能恢复一个严重的时钟漂移 1：该USART能恢复时钟漂移。必须在16倍时钟模式下
29	ONE	R/W	0x1	必须设为1 写USART_MAN寄存器时，该位必须始终设置为1
28	RX_MPOL	R/W	0x1	接收器曼彻斯特极性： 0：逻辑0被编码成0到1的转换，逻辑1被编码成1到0的转换 1：逻辑0被编码成1到0的转换，逻辑1被编码成0到1的转换
27:26	RSV	-	-	保留
25:24	RX_PP	R/W	0x0	接收器前同步信号样式检测： 00：ALL_ONE 01：ALL_ZERO 10：ZERO_ONE 11：ONE_ZERO
23:20	RSV	-	-	保留
19:16	RX_PL	R/W	0x1	接收器前同步信号长度； 0：禁止接收器前同步信号样式检测。 1-15：检测的前同步信号长度为RX_PL个位周期
15:13	RSV	-	-	保留
12	TX_MPOL	R/W	0x1	发送器曼彻斯特极性： 0：逻辑0被编码成0到1的转换，逻辑1被编码成1到0的转换。 1：逻辑0被编码成1到0的转换，逻辑1被编码成0到1的转换。
11:10	RSV	-	-	保留

位	名称	属性	复位值	描述
9:8	TX_PP	R/W	0x0	发送器前同步信号模式： 00：ALL_ONE 01：ALL_ZERO 10：ZERO_ONE 11：ONE_ZERO
7:4	RSV	-	-	保留
3:0	TX_PL	R/W	0x4	发送器前同步信号长度： 0：禁止发送器前同步信号的产生。 1-15：前同步信号长度为TX_PL个位周期

30.5.25 USART LIN 模式寄存器（USART_LINMR）

偏移地址：0x54

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17	SYNCDIS	R/W	0x0	禁用同步： 0：同步过程在LIN从节点配置中执行 1：同步过程不在LIN从节点配置中执行
16	PDCM	R/W	0x0	DMA模式： 0：DMA不能写LIN模式寄存器USART_LINMR 1：LIN模式寄存器USART_LINMR（标志除外）由DMA写入
15:8	DLC	R/W	0x0	数据长度： 当DLM=0，定义响应数据长度，响应数据长度等于DLC+1字节
7	WKUPTYP	R/W	0x0	唤醒信号类型： 0：控制寄存器中设置LINWKUP位，发送LIN 2.0唤醒信号 1：控制寄存器中设置LINWKUP位，发送LIN 1.3唤醒信号
6	FSDIS	R/W	0x0	禁用帧插槽模式： 0：使能帧插槽模式 1：禁用帧插槽模式
5	DLM	R/W	0x0	数据长度模式： 0：响应数据长度由该寄存器的DLC定义 1：响应数据长度由标识符的bit5和bit6定义（USART_LINIR的IDCHR）
4	CHKTYP	R/W	0x0	校验和类型： 0：LIN 2.0“增强型”校验和 1：LIN 1.3“经典型”校验和

位	名称	属性	复位值	描述
3	CHKDIS	R/W	0x0	禁用校验和： 0：在主节点配置中，会自动计算并发送校验和。在从节点配置中，将自动检查校验和 1：不管节点如何配置，都不会计算/发送校验和，也不会对其进行检查
2	PARDIS	R/W	0x0	禁用奇偶校验： 0：在主节点配置中，会自动计算并发送标识符奇偶校验。在主节点和从节点配置中，将自动检查奇偶校验 1：不管节点如何配置，都不会计算/发送标识符奇偶校验，也不会对其进行检查
1:0	NACT	R/W	0x0	LIN节点操作： 00：PUBLISH发送应答 01：SUBSCRIBE接收应答 10：IGNORE既不发送也不接收应答 11：保留

30.5.26 USART LIN 标识符寄存器（USART_LINIR）

偏移地址：0x58

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	IDCHR	R/W	0x0	标识符字符： 当USART_MODE = 0xA（主节点），IDCHR是读写的，其值是要传输的标识符字符 当USART_MODE = 0xB（从节点），IDCHR是只读的，其值是要接收的标识符字符

30.5.27 USART LIN 波特率寄存器（USART_LINBRR）

偏移地址：0x5C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:19	RSV	-	-	保留
18:16	LINFP	R	0x0	同步后的小数部分
15:0	LINCD	R	0x0	同步后的时钟分频

30.5.28 USART 写保护模式寄存器（USART_WPMR）

偏移地址：0xE4

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	WPKEY	W	0x0	在该字段中写入0x555341外的任何其他值将中止WPEN位的写入操作
7:1	RSV	-	-	保留
0	WPEN	R/W	0x0	0：当WPKEY为0x555341，禁用写保护 1：当WPKEY为0x555341，使能写保护

注：保护如下寄存器：

- USART 模式寄存器（USART_MR）
- USART 波特率发生器寄存器（USART_BRGR）
- USART 接收超时寄存器（USART_RTOT）
- USART 发送时间保护寄存器（USART_TTGR）
- USART FIDI 比率寄存器（USART_FIDI）
- USART IrDA 滤波寄存器（USART_IF）
- USART 曼彻斯特配置寄存器（USART_MAN）

30.5.29 USART 写保护状态寄存器（USART_WPSR）

偏移地址：0xE8

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:8	WPVSRC	R	0x0	写入保护冲突状态： 0：自上次读取USART_WPSR以来，未发生写保护冲突 1：自上次读取USART_WPSR以来，发生了写保护冲突
7:1	RSV	-	-	保留
0	WPVS	R	0x0	写保护冲突源 改为有效时，表示尝试对写保护寄存器进行写操作

30.6 使用流程

30.6.1 UART 模式

30.6.1.1 初始化

1. 开启对应引脚 GPIO 的时钟，把引脚配置成 USART_TX、USART_RX 复用功能。
2. 配置系统配置寄存器的 USART 模块时钟。

3. 配置 USART_BRGR 寄存器设置串口通信的波特率。
4. 配置 USART_MR 寄存器设置串口通信数据帧长度、校验类型、停止位长度。
5. 配置 USART_CR 寄存器使能发送器和接收器。
6. 配置 USART_IER 寄存器使能需要用到的中断。

30.6.1.2 发送流程

1. 发送数据前软件可以配置波特率参数，奇偶校验类型、数据帧格式。
2. 写入第一个数据到 USART_THR 寄存器。
3. 查询发送完成标志 USART_CSR[9]，如果 USART_CSR[9]=1 表示当前数据发送完成。
4. 可以继续写入下一个字节到 USART_THR。

30.6.1.3 接收流程

1. 接收数据前软件可以配置波特率参数，校验类型、数据帧格式。
2. 接收数据，查询 USART_CSR[0]标志位或者等待中断，如果 USART_CSR[0]= 1 时，接收器非空，则读取 USART_RHR 中的数据；读取数据后相应的标志位自动清除。
3. 接收错误处理：等待中断或者查询 USART_CSR 寄存器标志位，判断错误类型，执行相应的错误处理，处理完之后软件清除标志位。
4. 继续接收数据。

30.6.2 SPI 模式

30.6.2.1 主机模式

1. 初始化
   A. 开启对应引脚 GPIO 的时钟，把引脚配置成 USART_RTS、USART_TX、USART_RX、USART_SCK 复用功能。
   B. 配置系统配置寄存器的 USART 模块时钟。
   C. 配置 USART_MR[3:0]为 SPI 主机模式。
   D. 配置 USART_MR[7:6]设置字符长度为 8 位。
   E. 配置 USART_MR[18]设置 SCK 引脚输出时钟。
   F. 配置 USART_BRGR 寄存器设置时钟分频（内部时钟 6 分频以上）。
   G. 配置 USART_MR[16]和 USART_MR[8]设置 SPI 时钟极性和时钟相位。
   H. 配置 USART_CR 寄存器使能 SPI 收发器。
2. 全双工数据传输

A. 设置 USART_CR[18]为 1，使能片选拉低 CS。
B. 等待发送器就绪后可以向 USART_THR 寄存器写入一个数据。
C. 查询发送完成标志 USART_CSR[9]，如果 USART_CSR[9] =1 表示当前数据发送完成。
D. 查询 USART_CSR 标志位或者等待中断，如果 USART_CSR[0] = 1 时，则读取 USART_RHR 中的数据；读取数据后相应的标志位自动清除。
E. 如需继续收发数据，重复步骤 B-D。
F. 收发数据完成，设置 USART_CR[19]为 1，不使能片选拉高 CS。

30.6.2.2 从机模式

1. 初始化
   A. 开启对应引脚 GPIO 的时钟，把引脚配置成 USART_CTS、USART_TX、USART_RX、USART_SCK 复用功能。
   B. 配置系统配置寄存器的 USART 模块时钟。
   C. 配置 USART_MR[3:0] 为 SPI 从机模式。
   D. 配置 USART_MR[7:6] 设置字符长度为 8 位。
   E. 配置 USART_MR[18] 为 3，选择时钟为 SCK。
   F. 配置 USART_MR[16] 和 USART_MR[8] 设置 SPI 时钟极性和时钟相位。
   G. 配置 USART_CR 寄存器使能 SPI 收发器。
2. 全双工数据传输
   A. 等待发送器就绪后可以向 USART_THR 寄存器写入一个数据。
   B. 查询发送完成标志 USART_CSR[9]，如果 USART_CSR[9] =1 表示当前数据发送完成。
   C. 查询 USART_CSR 标志位或者等待中断，如果 USART_CSR[0] = 1 时，则读取 USART_RHR 中的数据；读取数据后相应的标志位自动清除。
   D. 如需继续收发数据，重复步骤 B-D。

30.6.3 LIN 模式

30.6.3.1 主节点模式

1. 初始化
   A. 开启对应引脚 GPIO 的时钟，把引脚配置成 USART_TX、USART_RX 复用功能。
   B. 配置系统配置寄存器的 USART 模块时钟。
   C. 配置 USART_MR[3:0]为 LIN 主节点。
   D. 配置 USART_CR 寄存器使能 SPI 收发器。
   E. 配置 USART_LINMR 寄存器设置主节点的工作模式。

F. 配置 USART_BRGR 寄存器设置 LIN 通信的波特率。

2. 发送数据

A. 配置 USART_LINMR[1:0]设置主节点的工作模式为 PUBLISH 发送应答。

B. 配置 USART_LINMR[15:8]设置 LIN 帧的数据长度。

C. 等待 USART_CSR[1]，发送器就绪后向 USART_LINIR[7:0]写入 ID 标识符。

D. 等待 USART_CSR[1]置位，ID 标志符发送完成，接着向 USART_THR 写入一个待发送的数据，等待 USART_CSR[1] 置位数据发送完成再写入数据，直到全部数据发送完成。

E. 如需再次发送数据，重复步骤 C、D。

3. 接收数据

A. 配置 USART_LINMR[1:0]设置主节点的工作模式为 SUBSCRIBE 接收应答。

B. 配置 USART_LINMR[15:8] 设置 LIN 帧的数据长度。

C. 等待 USART_CSR[1]置位，发送器就绪后向 USART_LINIR[7:0]写入 ID 标识符。

D. 等待 USART_CSR[0]置位，读取 USART_RHR 中的数据。

E. 重复步骤 D 直到所有数据读取完毕。

F. 如需再次接收数据，重复步骤 C、D。

30.6.3.2 从节点模式

1. 初始化

A. 开启对应引脚 GPIO 的时钟，把引脚配置成 USART_TX、USART_RX 复用功能。

B. 配置系统配置寄存器的 USART 模块时钟。

C. 配置 USART_MR[3:0]为 LIN 从节点。

D. 配置 USART_CR 寄存器使能 SPI 收发器。

E. 配置 USART_LINMR 寄存器设置从节点的工作模式。

F. 配置 USART_BRGR 寄存器设置 LIN 通信的波特率。

2. 发送数据

A. 等待 USART_CSR[14] 置位，判断收到的标识符是否正确。

B. 配置 USART_LINMR[15:8] 设置 LIN 帧的数据长度。

C. 配置 USART_LINMR[1:0]设置从节点的工作模式为 PUBLISH 发送应答，等待 USART_CSR[1] 置位，向 USART_THR 写入一个待发送的数据，等待 USART_CSR[1] 置位数据发送完成再写入数据，直到全部数据发送完成。

3. 接收数据

A. 配置 USART_LINMR[1:0] 设置从节点的工作模式为 SUBSCRIBE 接收应答。

B. 等待 USART_CSR[14] 置位，判断收到的标识符是否正确。

C. 配置 USART_LINMR[15:8] 设置 LIN 帧的数据长度。

D. 等待 USART_CSR[0] 置位，读取 USART_RHR 中的数据。

E. 重复步骤 D 直到所有数据读取完毕。

F. 等待 USART_CSR[15] 置位 LIN 总线传输完成。

G. 设置 USART_CR[8] 置 1，复位状态位。

31 串行外围接口（SPI）

31.1 概述

串行外围接口（SPI）被广泛用于在 EEPROM、FLASH、微控制器、DAC、ADC 等不同设备之间提供经济的板级接口。

31.2 主要特性

- 可配置主机模式或从机模式
- 主机和从机模式下均支持串行的全双工、半双工、单工发送和单工接收操作
- 可配置 16 位 SPI 时钟频率控制寄存器：在主机模式下，SCK 频率最高为 PCLK 频率的 1/2；在从机模式下，SCK 频率最高为 PCLK 频率的 1/4
- 可配置 SCK 的极性和相位
- 可配置摩托罗拉（Motorola）或德州仪器（TI）时序模式
- 可配置按最高位或最低位优先的顺序发送和接收数据
- 可配置每个字符长度为 1 至 32 位，默认为 8 位
- 支持数据拼接功能，以充分利用 FIFO 空间
- 具有 8 级 32 位宽的发送 FIFO 和接收 FIFO
- 可配置软件或硬件控制片选
- 支持 DMA 操作

31.3 系统框图

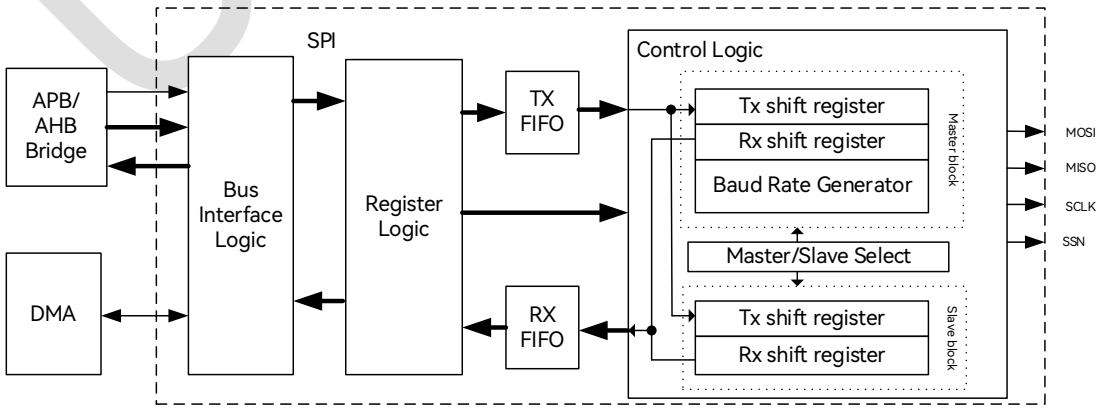


图 31-1：SPI 模块结构框图

如图 31-1 所示，SPI 模块由 5 个区块组成：APB 总线接口逻辑、配置寄存器逻辑、发送

FIFO 区块、接收 FIFO 区块和 SPI 控制逻辑。SPI 控制逻辑分为从机模块和主机模块，其中主机模块包括发送移位逻辑、接收移位逻辑和 SPI 时钟频率发生器逻辑。

31.4 管脚说明

表 31-1：SPI 管脚说明

功能管脚	复用管脚	方向	功能描述
SPI0_SCK	PA5,PB3	Input/Output	串行时钟信号
SPI0_MISO	PA6,PB4	Input/Output	数据信号
SPI0_MOSI	PA7,PB5	Input/Output	数据信号
SPI0_NSS	PA4,PA15	Input/Output	片选信号
SPI1_SCK	PB10,PB13	Input/Output	串行时钟信号
SPI1_MISO	PC2,PB14	Input/Output	数据信号
SPI1_MOSI	PC3,PB15	Input/Output	数据信号
SPI1_NSS	PB9,PB12	Input/Output	片选信号
SPI2_SCK	PB3,PC10	Input/Output	串行时钟信号
SPI2_MISO	PB4,PC11	Input/Output	数据信号
SPI2_MOSI	PB5,PC12	Input/Output	数据信号
SPI2_NSS	PA4,PA15	Input/Output	片选信号
SPI3_SCK	PE3	Input/Output	串行时钟信号
SPI3_MISO	PE5	Input/Output	数据信号
SPI3_MOSI	PE6	Input/Output	数据信号
SPI3_NSS	PE4	Input/Output	片选信号

31.5 功能描述

31.5.1 SPI 时钟频率控制寄存器

专用的 16 位“SPI 时钟频率控制寄存器（SPBRG）”仅适用于 SPI 主机模式，它控制发送和接收的频率。

SPI 时钟（SCLK）由 APBx 时钟（PCLK）分频产生，SPBRG 的值就是分频系数。

$$f_{SCLK} = f_{PCLK} / SPBRG$$

其中，SPBRG 寄存器中的值默认为 2，其范围应为 2 到 65535。在主机模式下，SCLK 频率最高为 PCLK 频率的 1/2；在从机模式下，SCLK 频率最高为 PCLK 频率的 1/4。

31.5.2 发送器 FIFO

发送器 FIFO（TX FIFO）的深度为 8 个字（一个字是 32 位）。CPU 或 DMA 将数据写入“发送数

据寄存器（TXREG）”时，数据即被写入 TX FIFO，发送操作开始。在发送过程中，TX FIFO 将数据移至“发送移位寄存器（TSR）”。

当“发送移位寄存器（TSR）”为空时，TX FIFO 将数据输出到 TSR。然后，TSR 将数据移至发送串行端口（TX）。

当 TX FIFO 未满并且需要数据发送时，将设置“发送突发请求（TXBREQ）”信号。信号发送到 DMA，以请求 DMA 将数据发送到 TX FIFO。

在发送过程中，当 TX FIFO 为空时，将产生“发送器变空中断标志（TXEPT_INTF）”。当发送器 FIFO 接收到足够的数据时（取决于“TXTLF”），将产生“发送器数据可用中断标志（TX_INTF）”。当 SPI 从机试图以空闲状态发送新字符时，将产生“从机发送器欠载中断标志（UNDERRUN_INTF）”。

31.5.3 接收器 FIFO

接收器 FIFO（RX FIFO）的深度为 8 个字（一个字是 32 位）。CPU 或 DMA 从 RX FIFO 接收数据。“接收移位寄存器（RSR）”用于从接收串行端口（RX）接收数据。传入数据放入 RSR。当所有位都移入时，数据传输到 RX FIFO。当数据从“移位寄存器”传输到“数据寄存器”时，“当前状态寄存器（CSTAT）”中的“接收寄存器就绪位（RXAVL）”置位。也可以在“中断状态寄存器（INTSTAT）”中设置“接收器 FIFO 数据可用中断标志位（RX_INTF）”。然后，CPU 从“接收数据寄存器（RXREG）”接收数据。如果收到“接收突发请求（RXBREQ）”，DMA 将从 RXREG 读取数据。

当主机或从机试图将数据写入满的 RX FIFO 时，最新的数据将无法写入，并会产生“接收器溢出错误中断标志（RXOERR_INTF）”。

31.5.4 SPI 控制逻辑

31.5.4.1 概述

SPI 允许 4 到 32 位数据同步发送和同时接收。可以在单主机环境中将 SPI 配置为从机或主机。使用时钟极性（CKPL）和时钟相位（CKPH）设置均可使用全部四种 SPI 时序模式。数据位顺序也可以设置为最低位（LSB）优先或最高位（MSB）优先。

发送器和接收器使用相同的时钟。数据仅在生成的串行时钟（SCLK）的上升沿或下降沿期间输出，并在 SCLK 的相反沿期间锁存。SPI 主设备和从设备的 SCLK 极性应相同，SCLK 相位也应相同。

SPI 用于交换数据，待发送的数据暂时保存在发送器 FIFO，接收到的数据保存在接收器 FIFO。如果禁用 SPI 模块，这些 FIFO 里面的数据都会丢失，因此必须在传输结束时先通过“接收数据寄存器（RXREG）”从接收器 FIFO 中读取数据，再禁用 SPI 模块。

标准的 SPI 协议包括两条数据线、一条时钟线和一条芯片选择线，介绍如下：

- 主机输出从机输入（MOSI 或 TX 或 SDA）：该数据线提供从主机转移到从机输入的输出数据。
- 主机输入从机输出（MISO 或 RX）：该数据线将从机的输出数据提供给主机的输入。在任何特定的传输过程中，最多只能有一个从设备传输数据。
- 串行时钟（SCLK 或 SCL）：该时钟线由主机驱动并调节数据流。主机可以以各种 SPI 时钟频率传输数据。SCLK 线对每个发送位循环一次。
- 芯片选择（CS 或 SSN）：来自主机的从选择输入信号，低电平有效。

31.5.4.2 主机模式

SPI 模块仅支持单主机环境。唯一的 SPI 主设备可以启动发送和接收。

发送器的核心是“发送移位寄存器”。软件通过 APB 总线或 DMA 总线将待发送的数据装入“发送缓冲寄存器（SPI_TXREG）”，TXREG 立即将数据发送到发送器 FIFO（TX FIFO）。当传输开始后，TSR 从 TX FIFO 获取数据，如果 TSR 为空，则 TX FIFO 中的数据字节将立即传输到 TSR。TSR 未映射到数据存储器中，因此用户无法使用。

一旦 TX FIFO 有足够空位，“发送器 FIFO 空位可用中断标志位（TX_INTF）”就会置 1。当 TX FIFO 和“发送移位寄存器”中的数据全部放送完毕时，“发送器变空中断标志位（txept_intf）”就会置 1。需要注意的是，这些中断标志位是否置 1 与“中断使能寄存器（SPI_TXIEN）”的状态无关；如果需要读取使能后的中断状态，可以读取“屏蔽后中断状态寄存器（SPI_MINTSTAT）”，给“中断清除寄存器（SPI_INTCLR）”写 1 会清除屏蔽前和屏蔽后的中断标志。

主机因为控制着 SPI 时钟（SCLK），可以随时启动数据传输。在主机模式下，一旦把数据写入 TXREG 寄存器，就开始发送并接收数据。如果只打算接收数据，可以禁用发送功能，接收移位寄存器（RSR）仍将按照 SCLK 频率移动 MISO 引脚上的信号。每接收完一个 SPI 字符，RSR 就会把 SPI 字符保存到接收器 FIFO 中，“中断状态寄存器（SPI_INTSTAT）”和“当前状态寄存器（SPI_CSTAT）”也将相应地发生改变。

SCLK 时钟仅在数据传输期间有效。

31.5.4.3 从机模式

SPI 从机接收来自 SPI 主机的信号。当外部时钟脉冲出现在 SCLK 引脚上时，数据就被发送和接收。该外部时钟必须满足电气规范中规定的高、低电平最短保持时间的要求。

为了准备数据传输，SPI 从机需要在传输开始前把待发送的数据写入 TXREG 寄存器。TXREG 寄存器将数据写入 TX FIFO，然后传输至“发送移位寄存器（TSR）”。一旦主机发出 SCLK，从机就将待发送的数据从 MISO 引脚移出，同时通过 MOSI 移入接收的数据。

31.5.5 数据拼接

通过设置寄存器 SPI_CCTL[7:6]，使用数据拼接功能，就可以将短的字符拼接到 TX FIFO 和 RX FIFO 中的一个字中。拼接格式如下：

1. 如果 SPI_CCTL[12:8]≤ 7，则每 4 个 SPI 字符将被拼接为 1 个字（32 位）。

   示例一：SPI_CCTL[12:8] = 5，每个 SPI 字符长 6 位，拼接格式如下：

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

图 31-1：数据拼接示例一（SPI_CCTL[12:8] = 5）

2. 如果 8 ≤ SPI_CCTL[12:8] ≤ 15，则每 2 个 SPI 字符将被拼接为 1 个字。

   示例二：SPI_CCTL[12:8] = 13，每个 SPI 字符长 14 位，拼接格式如下：

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

图 31-2：数据拼接示例二（SPI_CCTL[12:8] = 13）

3. 如果 16 ≤ SPI_CCTL[12:8] ≤ 31，则每 1 个 SPI 字符将被拼接为 1 个字。

   示例三：SPI_CCTL[12:8]= 28，每个 SPI 字符长 29 位，拼接格式如下：

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

图 31-3：数据拼接示例三（SPI_CCTL[12:8] = 28）

31.5.6 接口时序

31.5.6.1 摩托罗拉时序模式

摩托罗拉时序模式即为通常的时序模式。

表 31-2：摩托罗拉时序模式

模式	时钟极性（CKPL）	时钟相位（CKPH）
0	0：时钟在空闲时低电平	0：数据在时钟第一个边沿（上升沿）采样
1	0：时钟在空闲时低电平	1：数据在时钟第二个边沿（下降沿）采样
2	1：时钟在空闲时高电平	0：数据在时钟第一个边沿（下降沿）采样
3	1：时钟在空闲时高电平	1：数据在时钟第二个边沿（上升沿）采样

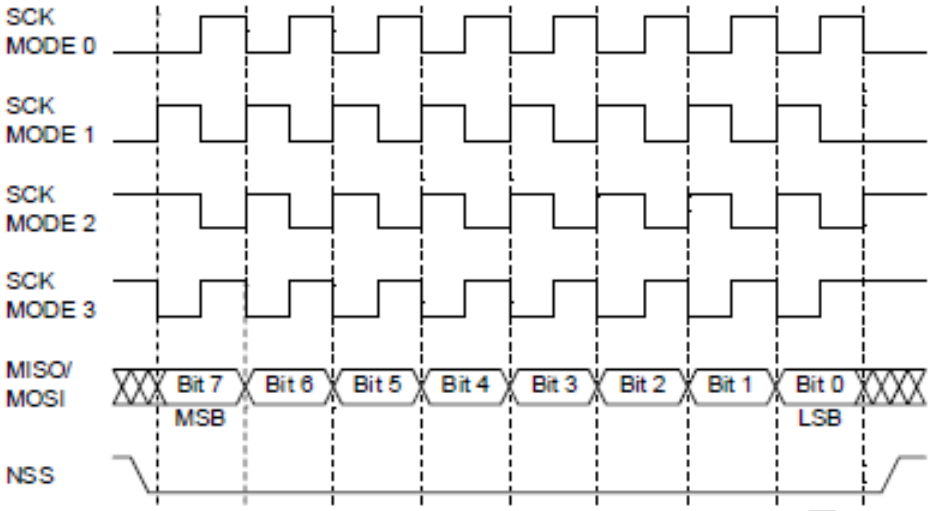


图 31-4：摩托罗拉模式下 SPI 不同模式下传输时序图

注：SCK 即为串行时钟（SCLK）。NSS 即为片选（SSN）。

31.5.6.2 德州仪器时序模式

在德州仪器时序模式（TI mode）下，CKPL 与摩托罗拉模式下的相同，但是 CKPH 与摩托罗拉模式下的相反。而且，如图 31-5 所示，每发送一个数据之前，需要拉高片选（SSN，图 31-5 中为 SPI_CS_N）信号一个串行时钟周期。本模块只支持单次传输格式，不支持连续传输格式。

表 31-3：TI 时序模式

模式	时钟极性（CKPL）	时钟相位（CKPH）
0	0：时钟在空闲时低电平	1：数据在时钟第二个边沿（下降沿）采样
1	0：时钟在空闲时低电平	0：数据在时钟第一个边沿（上升沿）采样
2	1：时钟在空闲时高电平	1：数据在时钟第二个边沿（上升沿）采样
3	1：时钟在空闲时高电平	0：数据在时钟第一个边沿（下降沿）采样

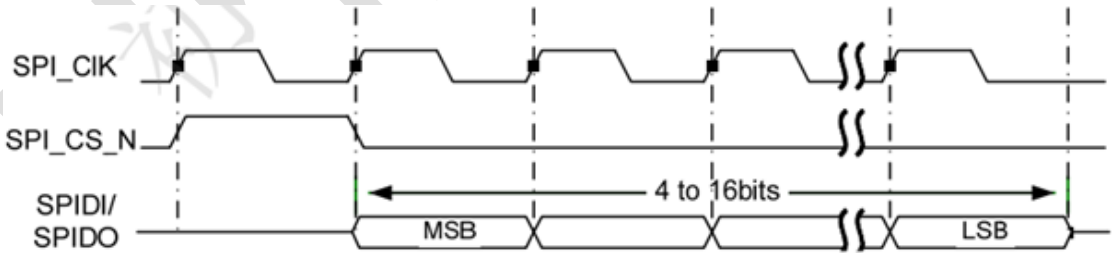


图 31-5：TI 模式下 SPI 在 MODE2 模式下传输时序图

注：SPI_CLK 即为串行时钟（SCLK）。SPI_CS_N 即为片选（SSN）。SPIDI/SPIDO 即为串行数据 MISO 和 MOSI。本 SPI 模块支持每个 SPI 字符长 1 至 32 位。

应用提示：主机和从机应保持时序模式一致，时钟极性一致，时钟相位一致。

31.6 寄存器描述

SPI0 寄存器基地址：0x4600_1000

SPI1 寄存器基地址：0x4700_0000

SPI2 寄存器基地址：0x4700_1000

SPI3 寄存器基地址：0x4700_2000

寄存器列表如下：

表 31-4：SPI 寄存器列表

偏移地址	名称	描述
0x00	SPI_TXREG	发送数据寄存器
0x04	SPI_RXREG	接收数据寄存器
0x08	SPI_CSTAT	当前状态寄存器
0x0C	SPI_INTSTAT	中断状态寄存器
0x10	SPI_MINTSTAT	屏蔽后中断状态寄存器
0x14	SPI_INTEN	中断使能寄存器
0x18	SPI_INTCLR	中断清除寄存器
0x1C	SPI_GCTL	全局控制寄存器
0x20	SPI_CCTL	通用控制寄存器
0x24	SPI_SPBRG	SPI 时钟频率控制寄存器
0x28	SPI_RXDNR	接收数据数量寄存器
0x2C	SPI_TXDNR	发送数据数量寄存器
0x30	SPI_SCSR	从机片选寄存器

31.6.1 发送数据寄存器（SPI_TXREG）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	TXREG	R/W	0x0	本寄存器存储着最近一次写入发送器 FIFO 的数据。数据的有效位数取决于 SPI_CCTL[12:8]的值和数据拼接。

31.6.2 接收数据寄存器（SPI_RXREG）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	RXREG	R	0x0	本寄存器存储着由接收器 FIFO 移出的最后一个数据。数据的有效位数取决于 SPI_CCTL[12:8]的值和数据拼接。本寄存器只读。

31.6.3 当前状态寄存器（SPI_CSTAT）

偏移地址：0x08

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3	RXAVL_4BYTE	R	0x0	RX FIFO 有 4 个字可用数据的标志。 当接收器 FIFO 接收到不少于 4 个字的可用数据时，本位置 1。 0：接收器 FIFO 中的数据不足 4 个字 1：接收器 FIFO 有 4 个字或更多的数据
2	TXFULL	R	0x0	TX FIFO 满状态标志： 0：发送器 FIFO 未满 1：发送器 FIFO 已满
1	RXAVL	R	0x0	收到可用数据标志。 当接收器 FIFO 收到完整一个字的数据时本位置 1： 0：接收器 FIFO 空 1：接收器 FIFO 收到完整一个字的数据
0	TXEPT	R	0x1	发送器空状态标志： 0：发送器非空 1：发送器 FIFO 和“发送移位寄存器”空

31.6.4 中断状态寄存器（SPI_INTSTAT）

偏移地址：0x0C

复位值：0x0000_0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	TXMATCH_INTF	R	0x0	发送完成中断标志： 当实际发送的数据量与 SPI_TXDNR 寄存器中要发送的数据量匹配时，发送过程即完成并产生中断。本中断在主机单发送模式、主机双工模式或从机单发送模式下才可以产生。 0：无发送完成中断产生 1：有发送完成中断产生
6	TXEPT_INTF	R	0x0	发送器变空中断标志： 当发送器 FIFO 和“发送移位寄存器”中的数据全部放送完毕时，或者禁用发送功能导致发送器 FIFO 和 TSR 中的数据清空时产生的中断。 0：无发送器变空中断产生 1：有发送器变空中断产生
5	RXFIFO_FULL_INTF	R	0x0	RX FIFO 满中断标志： 当接收器 FIFO 满时产生的中断。 0：无接收器 FIFO 满中断产生 1：有接收器 FIFO 满中断产生
4	RXMATCH_INTF	R	0x0	接收完成中断标志： 当实际接收的数据量与 SPI_RXDNR 寄存器中要接收的数据量匹配时，接收过程即完成并产生中断。本中断在主机单接收模式、从机单接收模式或从机双工模式下才可以产生。 0：无接收完成中断产生 1：有接收完成中断产生

位	名称	属性	复位值	描述
3	RXOERR_INTF	R	0x0	接收器溢出错误中断标志： 当主机或从机试图将数据写入满的 RX FIFO 时，最新的数据将无法写入，并将会发生溢出错误。 0：无接收器溢出错误中断产生 1：有接收器溢出错误中断产生
2	UNDERRUN_INTF	R	0x0	从机发送器欠载中断标志： 当 SPI 从机试图以空闲状态发送新字符时，发生欠载错误。 0：无从机发送器欠载错误中断产生 1：有从机发送器欠载错误中断产生
1	RX_INTF	R	0x0	接收器 FIFO 数据可用中断标志： 当接收被使能且接收器 FIFO 有足够（取决于“rxtlf”）数据时，本位置 1。 0：无接收器 FIFO 数据可用中断产生 1：有接收器 FIFO 数据可用中断产生
0	TX_INTF	R	0x0	发送器 FIFO 空位可用中断标志： 当发送被使能且发送器 FIFO 有足够（取决于“txtlf”）空位时，本位置 1。 0：无发送器 FIFO 空位可用中断产生 1：有发送器 FIFO 空位可用中断产生

注：中断使能寄存器（SPI_INTEN）的某一位为 0，可以阻止其对应条件的中断信号产生，但是不能防止在对应条件达到时本寄存器相应标志位置 1。给中断清除寄存器（SPI_INTCLR）某一位写 1，会把本寄存器对应的位清 0。

31.6.5 屏蔽后中断状态寄存器（SPI_MINTSTAT）

偏移地址：0x10

复位值：0x0000_0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	TXMATCH_MINTF	R	0x0	发送完成中断标志： 当实际发送的数据量与“SPI_TXDNR 寄存器”中要发送的数据量匹配时，发送过程即完成并产生中断。此中断可以产生于主机单发送模式、主机双工模式或从机单发送模式。 0：无发送完成中断产生 1：有发送完成中断产生
6	TXEPT_MINTF	R	0x0	发送器变空中断标志： 当发送器 FIFO 和“发送移位寄存器”空时产生的中断。 0：无发送器变空中断产生 1：有发送器变空中断产生

位	名称	属性	复位值	描述
5	RXFIFO_FULL_MINTF	R	0x0	RX FIFO 满中断标志： 当接收器 FIFO 满时产生的中断。 1：有接收器 FIFO 满中断产生 0：无接收器 FIFO 满中断产生
4	RXMATCH_MINTF	R	0x0	接收完成中断标志： 当实际接收的数据量与“RXDNR 寄存器”中要接收的数据量匹配时，接收过程即完成并产生中断。此中断可以产生于主机单接收模式、从机单接收模式或从机双工模式。 0：无接收完成中断产生 1：有接收完成中断产生
3	RXOERR_MINTF	R	0x0	接收器溢出错误中断标志： 当主机或从机试图将数据写入满的 RX FIFO 时，最新的数据将无法写入，并将会发生溢出错误。 0：无接收器溢出错误中断产生 1：有接收器溢出错误中断产生
2	UNDERRUN_MINTF	R	0x0	从机发送器欠载中断标志： 当 SPI 从机试图以空闲状态发送新字符时，发生欠载错误。 0：无从机发送器欠载错误中断产生 1：有从机发送器欠载错误中断产生
1	RX_MINTF	R	0x0	接收器数据可用中断标志： 当接收器 FIFO 接收到足够的数据时（取决于 SPI_GCTL[6:5]），本位置 1。 0：无接收器数据可用中断产生 1：有接收器数据可用中断产生
0	TX_MINTF	R	0x0	发送器 FIFO 空位可用中断标志： 当发送器 FIFO 有足够（取决于 SPI_GCTL[8:7]）空位时，本位置 1。 0：无发送器 FIFO 空位可用中断产生 1：有发送器 FIFO 空位可用中断产生

注：本寄存器是影子寄存器，映射中断状态寄存器（SPI_INTSTAT）逻辑“与”中断使能寄存器（SPI_INTEN）的结果。若某个中断没有使能，则本寄存器相应的中断标志不会置位。

31.6.6 中断使能寄存器（SPI_INTEN）

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	TXMATCHEN	R/W	0x0	发送完成中断使能： 0：禁止发送完成中断 1：允许发送完成中断

位	名称	属性	复位值	描述
6	TXEPT_IEN	R/W	0x0	发送器变空中断使能： 0：禁止发送器变空中断 1：允许发送器变空中断
5	RXFIFO_FULL_IEN	R/W	0x0	接收器 FIFO 满中断使能： 0：禁止接收器 FIFO 满中断 1：允许接收器 FIFO 满中断
4	RXMATCHEN	R/W	0x0	接收完成中断使能： 0：禁止接收完成中断 1：允许接收完成中断
3	RXOERREN	R/W	0x0	接收器溢出错误中断使能： 0：禁止接收器溢出错误中断 1：允许接收器溢出错误中断
2	UNDERRUNEN	R/W	0x0	从机发送器欠载中断使能： 0：禁止从机发送器欠载中断 1：允许从机发送器欠载中断
1	RXIEN	R/W	0x0	接收器 FIFO 数据可用中断使能： 0：禁止接收器 FIFO 数据可用中断 1：允许接收器 FIFO 数据可用中断
0	TXIEN	R/W	0x0	发送器 FIFO 空位可用中断使能： 0：禁止发送器 FIFO 空位可用中断 1：允许发送器 FIFO 空位可用中断

注：本寄存器的某一位为 0，可以阻止其对应条件的中断信号产生，但是不能防止在对应条件达到时中断状态寄存器（SPI_INTSTAT）相应标志位置 1。

31.6.7 中断清除寄存器（SPI_INTCLR）

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7	TXMATCHCLR	W	0x0	发送完成中断清除： 0：不清除发送完成中断 1：清除发送完成中断
6	TXEPT_ICLR	W	0x0	发送器变空中断清除： 0：不清除发送器变空中断 1：清除发送器变空中断
5	RXFIFO_FULL_ICLR	W	0x0	接收器 FIFO 满中断清除： 0：不清除接收器 FIFO 满中断 1：清除接收器 FIFO 满中断
4	RXMATCHCLR	W	0x0	接收完成中断清除： 0：不清除接收完成中断 1：清除接收完成中断
3	RXOERRCLR	W	0x0	接收器溢出错误中断清除： 0：不清除接收器溢出错误中断 1：清除接收器溢出错误中断

位	名称	属性	复位值	描述
2	UNDERRUNCLR	W	0x0	从机发送器欠载中断清除： 0：不清除从机发送器欠载中断 1：清除从机发送器欠载中断
1	RXICLR	W	0x0	接收器 FIFO 数据可用中断清除： 本位仅在 I2S_GCR[13]为 0 时有效。 0：不清除接收 FIFO 器数据可用中断 1：清除接收器 FIFO 数据可用中断
0	TXICLR	W	0x0	发送器 FIFO 空位可用中断清除： 本位仅在 I2S_GCR[13]为 0 时有效。 0：不清除发送器 FIFO 空位可用中断 1：清除发送器 FIFO 空位可用中断

注：给本寄存器某一位写 1，会把“中断状态寄存器（SPI_INTSTAT）”对应的位清 0。

31.6.8 全局控制寄存器（SPI_GCTL）

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:13	RSV	-	-	保留
12	HALF_DIR	R/W	0x0	**半双工模式数据输出使能** 0：数据输入 1：数据输出
11	BIDIRMODE	R/W	0x0	半双工模式使能。 在主机半双工模式下，MOSI 作为双向的数据线；在从机半双工模式下，MISO 作为双向的数据线。在半双工模式下，数据传输方向取决于“txen & !rxen”，即当 txen == 1 且 rxen == 0 时，使用的引脚输出数据，否则引脚输入数据。 对于单工模式，只需要将不使用的引脚配置成 GPIO 功能即可。 0：全双工模式或单工模式 1：半双工模式
10	CSN_SEL	R/W	0x0	片选信号控制方选择： 在摩托罗拉时序模式下，片选信号可以选择由软件或者硬件控制。在硬件控制下，片选信号在数据传输时仍然会与 SPI_SCSR 寄存器一致；但是每发送或接收完一个 SPI 字符后，片选信号会拉高一个 PCLK 周期，然后重新拉低以传输下一个 SPI 字符；当数据全部发送完成后，片选信号会自动拉高。 在德州仪器（TI）时序模式下，片选信号由软件和硬件共同控制，本位无效。片选信号在空闲状态或数据传输时会与 SPI_SCSR 寄存器一致；但是每发送或接收一个 SPI 字符之前，片选信号会拉高一个 SCLK 周期，然后重新拉低以传输这一个 SPI 字符。 0：芯片选择信号由 SPI_SCSR 寄存器配置 1：芯片选择信号由硬件控制

位	名称	属性	复位值	说明
9	DMAMODE	R/W	0x0	DMA 访问模式使能： 0：使用 CPU 访问模式（CPU 读写 TX FIFO 和 RX FIFO） 1：使用 DMA 访问模式（DMA 读写 TX FIFO 和 RX FIFO）
8:7	TXTLF	R/W	0x0	能触发发送器 FIFO 接收数据的空位数量选择： 00：TX FIFO 有至少 1 个空位（默认） 01：TX FIFO 有至少 4 个空位 10：TX FIFO 有至少 8 个空位 11：TX FIFO 有至少 16 个空位（不可用）
6:5	RXTLF	R/W	0x0	能触发接收器 FIFO 发送数据的数据数量选择： 00：RX FIFO 有至少 1 个数据（默认） 01：RX FIFO 有至少 4 个数据 10：RX FIFO 有至少 8 个数据 11：RX FIFO 有至少 16 个数据（不可用）
4	RXEN	R/W	0x0	接收使能： 0：禁止接收数据，且清空接收器 FIFO 1：允许接收数据
3	TXEN	R/W	0x0	发送使能： 0：禁止发送数据，且清空接收器 FIFO 1：允许发送数据
2	MM	R/W	0x0	主/从机模式选择： 0：从机模式（串行时钟由外部模块产生） 1：主机模式（串行时钟由本模块产生）
1	INT_EN	R/W	0x0	SPI 中断使能： 0：禁止 SPI 全部中断 1：允许 SPI 中断（还需要配置 SPI_INTEN 寄存器）
0	EN	R/W	0x0	SPI 使能： 0：禁用 SPI（大部分电路保持在复位状态） 1：启用 SPI

31.6.9 通用控制寄存器（SPI_CCTL）

偏移地址：0x20

复位值：0x0000 0700

位	名称	属性	复位值	说明
31:13	RSV	-	-	保留
12:8	SPILEN	R/W	0x7	SPI 字符长度控制： 定义每个 SPI 字符的长度为（spilen + 1）位。支持字符长度为 1 到 32 位，默认是 8 位。
7	RX_STITCH	R/W	0x0	接收数据拼接使能： 0：禁用接收数据拼接功能 1：启用接收数据拼接功能
6	TX_STITCH	R/W	0x0	发送数据拼接使能： 0：禁用发送数据拼接功能 1：启用发送数据拼接功能

位	名称	属性	复位值	说明
5	LSBFE	R/W	0x0	最低位优先传输选择： 0：首先发送或接收数据的最高位（MSB） 1：首先发送或接收数据的最低位（LSB）
4	RXEDGE	R/W	0x0	主机模式下数据接收相位调整： 0：在 SCLK 的接收边沿采样 1：在 SCLK 的下一个发送边沿采样（延迟）
3	TXEDGE	R/W	0x0	从机模式下数据发送相位调整： 注：当 2 f_{SCLK} ≤ f_{PCLK} < 3 f_{SCLK} 时，本位只能写 1； 当 f_{PCLK} ≥ 3 f_{SCLK} 时，本位建议写 0。 0：发送数据在 SCLK 发送边沿改变 1：发送数据在 PCLK 上升沿改变（提前）
2	TI_MOD	R/W	0x0	时序模式选择： 在德州仪器（TI）时序模式下，在传输每个字符之前需要额外 1 个空闲的 SCLK 周期来拉高 SSN，并且它的 CKPH 与摩托罗拉（Motorola）时序模式下相反。 0：摩托罗拉（Motorola）时序模式 1：德州仪器（TI）时序模式
1	CKPL	R/W	0x0	SCLK 极性： 时钟极性决定时钟在空闲状态下是高电平还是低电平。 0：空闲状态下，SCLK 为低电平。 1：空闲状态下，SCLK 为高电平。
0	CKPH	R/W	0x0	SCLK 相位： 0：Motorola 模式下，在第一个 SPI 时钟边沿采样数据，在第二个边沿改变数据；TI 模式下，在第一个 SPI 时钟沿改变数据，在第二个时钟沿采样数据。 1：Motorola 模式下，在第一个 SPI 时钟沿改变数据，在第二个时钟沿采样数据；TI 模式下，在第一个 SPI 时钟边沿采样数据，在第二个边沿改变数据。

31.6.10 SPI 时钟频率控制寄存器（SPI_SPBRG）

偏移地址：0x24

复位值：0x0000 0002

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	SPBRG	R/W	0x2	本寄存器用于控制 SPI 时钟频率。 SPI 时钟频率公式： f_{SCLK} = f_{PCLK} / SPBRG 其中，f_{PCLK} 是 APB 时钟频率。 注意：请勿给本寄存器写入 0 或 1，写入 0 或 1 无效。

31.6.11 接收数据数量寄存器（SPI_RXDNR）

偏移地址：0x28

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	RXDNR	R/W	0x1	本寄存器用于保存下一次接收过程中要接收的（未拼接）字符数。 SPI 主机模式下： 仅在单工接收模式下，本寄存器有效，而且会影响产生的 SPI 时钟的周期数，所以必须被适当配置。 SPI 从机模式下（仅在字符拼接功能下有效）： 本寄存器在单工接收模式和双工（接收加发送）模式下均有效。特别地，在双工模式下，它同时表示将要接收和将要发送的字符数。 本寄存器默认值为 1，此值在 CPU 向其写入新的值之前不会更改。注意：请勿将 0 写入本寄存器。

31.6.12 发送数据数量寄存器（SPI_TXDNR）

偏移地址：0x2C

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	TXDNR	R/W	0x1	本寄存器用于保存**下一次发送过程中要发送的（未拼接）字数**，仅在字符拼接功能下有效。 **SPI 主机模式下：** 本寄存器在单工发送模式和双工（接收加发送）模式下均有效特别地，在双工模式下，它同时表示将要发送和将要接收的字数。 **SPI 从机模式下：** 本寄存器仅在单工发送模式下有效。 本寄存器默认值为 1，此值在 CPU 向其写入新的值之前不会改。注意：请勿将 0 写入本寄存器。

31.6.13 从机片选寄存器（SPI_SCSR）

偏移地址：0x30

复位值：0x0000 00FF

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CS0	R/W	0x1	主机模式下的片选输出信号，低电平有效。本寄存器在 SPI 从机模式下不可用： 0：与该位对应的从机被选择 1：与该位对应的从机未被选择 注：只有 bit[0]有用，bit[7:1]不需要使用。

31.7 使用流程

31.7.1 主机模式

31.7.1.1 主机双工收发或单工发送

1. 配置寄存器
   A. 配置系统寄存器中外设时钟、复位和引脚复用相关的寄存器，开启时钟，释放复位。
   B. 配置 SPI_GCTL[0]，GCTL[2]，GCTL[3]置 1，其他位按需配置。
   C. 配置 SPI_CCTL 寄存器，设置时序模式和数据格式。
   D. 配置 SPI_SPBRG 寄存器，设置 SCLK 分频系数。
   E. 给 SPI_INTCLR 寄存器写 0xFF，清除 SPI_INTSTAT 寄存器的中断状态。
   F. 配置 SPI_INTEN 寄存器，可选择设置中断。
   G. 当字符拼接功能有效时，配置 SPI_TXDNR 寄存器，设置待发送的字符数量。
   H. 配置 SPI_SCSR 寄存器，拉低被选择的从机所对应的片选信号。
2. 传输数据
   A. 如果需要发送数据，则通过 CPU 或 DMA 给 SPI_TXREG 寄存器写入需要发送的数据。
   B. 当 SPI_CSTAT[1]为 1 时，表示收到数据。如果需要接收数据，则通过 CPU 或 DMA 从 SPI_RXREG 寄存器读取已接收的数据。
3. 处理中断

   如果有中断发生，则读取 SPI_MINTSTAT 寄存器，判断中断发生的原因。处理完成之后，给 SPI_INTCLR 寄存器的相应位写 1，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。
4. 结束传输
   A. 等待 SPI_INTSTAT[6]变成 1，或者等待 SPI_CSTAT[0]变成 1。因为数据发送的速率可能比写入发送器 FIFO 的速率慢，所以给 SPI_TXREG 寄存器写完待发送的数据并不代表全部数据已完成发送。如果发送器非空，则需要等待发送器内的数据全部发送出去，才真正表示发送完成。
   B. 配置 SPI_SCSR 寄存器，拉高片选信号。若选择硬件控制片选信号，也可以不配置此寄存器。
   C. 配置 SPI_GCTL[0]置 0，关闭 SPI 模块。
   D. 配置系统寄存器中外设时钟和复位相关的寄存器，关闭时钟，恢复复位。

31.7.1.2 主机半双工接收

1. 配置寄存器
   A. 配置系统寄存器中外设时钟、复位和引脚复用【主机使用 MOSI 进行通信】相关的寄存器，开启时钟，释放复位。
   B. 配置SPI_GCTL寄存器，注意SPI_GCTL[4]和SPI_GCTL[3]暂时不要置位，SPI_GCTL[11]、SPI_GCTL[0]、SPI_GCTL[2]置 1，其他位按需配置。
   C. 配置 SPI_CCTL 寄存器，设置时序模式和数据格式。
   D. 配置 SPI_SPBRG 寄存器，设置 SCLK 分频系数。
   E. 给 SPI_INTCLR 寄存器写 0xFF，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。
   F. 配置 SPI_INTEN 寄存器，可选择设置中断。
   G. 配置 SPI_SCSR 寄存器，拉低被选择的从机所对应的片选信号。
2. 传输数据
   A. 设置 SPI_GCTL[12]为 0，SPI_GCTL[3]为 0，SPI_GCTL[4]为 1，SPI_GCTL[0]为 1。
   B. 当 SPI_CSTAT[1]为 1 时，表示收到数据。如果需要接收数据，则通过 CPU 或 DMA 从 RXREG 寄存器读取已接收的数据。
   C. 设置 SPI_GCTL[4]为 0。
3. 处理中断
   如果有中断发生，则读取 SPI_MINTSTAT 寄存器，判断中断发生的原因。处理完成之后，给 SPI_INTCLR 寄存器的相应位写 1，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。
4. 结束传输
   A. 配置 SPI_SCSR 寄存器，拉高片选信号。若选择硬件控制片选信号，也可以不配置此寄存器。
   B. 配置 SPI_GCTL[0]置 0，关闭 SPI 模块。
   C. 配置系统寄存器中外设时钟和复位相关的寄存器，关闭时钟，恢复复位。

31.7.1.3 主机半双工发送

1. 配置寄存器
   A. 配置系统寄存器中外设时钟、复位和引脚复用【主机使用 MOSI 进行通信】相关的寄存器，开启时钟，释放复位。
   B. 配置SPI_GCTL寄存器，注意SPI_GCTL[4]和SPI_GCTL[3]暂时不要置位，SPI_GCTL[11]、SPI_GCTL[0]、SPI_GCTL[2]置 1，其他位按需配置。

C. 配置 SPI_CCTL 寄存器，设置时序模式和数据格式。

D. 配置 SPI_SPBRG 寄存器，设置 SCLK 分频系数。

E. 给 SPI_INTCLR 寄存器写 0xFF，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。

F. 配置 SPI_INTEN 寄存器，可选择设置中断。

G. 配置 SPI_SCSR 寄存器，拉低被选择的从机所对应的片选信号。

2. 传输数据

A. 设置 SPI_GCTL[0]为 0，SPI_GCTL[12]为 1，SPI_GCTL[3]为 1，SPI_GCTL[4]为 0，再置位 SPI_GCTL[0]为 1。

B. 当 SPI_CSTAT[2]为 0 时，表示发送器 FIFO 未满，可以往 SPI_TXREG 寄存器填充所要发送的数据。

3. 处理中断

如果有中断发生，则读取 SPI_MINTSTAT 寄存器，判断中断发生的原因。处理完成之后，给 SPI_INTCLR 寄存器的相应位写 1，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。

4. 结束传输

A. 当 SPI_CSTAT[0]为 1 时，表示数据从 SPI_TXREG 寄存器发送出去。

B. 配置 SPI_SCSR 寄存器，拉高片选信号。若选择硬件控制片选信号，也可以不配置此寄存器。

C. 配置 SPI_GCTL[0]寄置 0，关闭 SPI 模块。

D. 配置系统寄存器中外设时钟和复位相关的寄存器，关闭时钟，恢复复位。

31.7.1.4 主机单工接收

1. 配置寄存器

A. 配置系统寄存器中外设时钟、复位和引脚复用相关的寄存器，开启时钟，释放复位。

B. 配置 SPI_GCTL 寄存器，注意 SPI_GCTL[4]暂时不要置 1，SPI_GCTL[0]、SPI_GCTL[2]置 1，其他位按需配置。

C. 配置 SPI_CCTL 寄存器，设置时序模式和数据格式。

D. 配置 SPI_SPBRG 寄存器，设置 SCLK 分频系数。

E. 给 SPI_INTCLR 寄存器写 0xFF，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。

F. 配置 SPI_INTEN 寄存器，设置中断使能。

G. 配置 SPI_RXDNR 寄存器，设置待接收的字符数量。

H. 配置 SPI_SCSR 寄存器，拉低被选择的从机所对应的片选信号。

I. 配置 SPI_GCTL 寄存器，SPI_GCTL[4]置 1。

2. 传输数据

当 SPI_CSTAT[1]为 1 时，表示收到数据。如果需要接收数据，则通过 CPU 或 DMA 从 SPI_RXREG 寄存器读取已接收的数据。

3. 处理中断

如果有中断发生，则读取 SPI_MINTSTAT 寄存器，判断中断发生的原因。处理完成之后，给 SPI_INTCLR 寄存器的相应位写 1，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。

4. 结束传输

A. 由于主机控制时钟和片选信号，所以传输何时结束也由主机控制。当字符拼接功能有效时，也可等待 SPI_INTSTAT[4]变为 1，表示已经接收到 SPI_RXDNR 寄存器设定量的 SPI 字符，即接收完成。同时，配置 SPI_GCTL[4]置 0。

B. 配置 SPI_SCSR 寄存器，拉高片选信号。若选择硬件控制片选信号，也可以不配置此寄存器。

C. 配置 SPI_GCTL[0]置 0，关闭 SPI 模块。

D. 配置系统寄存器中外设时钟和复位相关的寄存器，关闭时钟，恢复复位。

31.7.2 从机模式

31.7.2.1 从机双工收发或单工接收

1. 配置寄存器

A. 配置系统寄存器中外设时钟、复位和引脚复用相关的寄存器，开启时钟，释放复位。

B. 配置 SPI_GCTL[0]和 SPI_GCTL[4]置 1，SPI_GCTL[2]置 0，其他位按需配置。

C. 配置 SPI_CCTL 寄存器，设置时序模式和数据格式。

D. 给 SPI_INTCLR 寄存器写 0xFF，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。

E. 配置 SPI_INTEN 寄存器，设置中断使能。

F. 当字符拼接功能有效时，配置 SPI_RXDNR 寄存器，设置待接收的字符数量。

2. 传输数据

A. 如果需要发送数据，则通过 CPU 或 DMA 给 SPI_TXREG 寄存器写入需要发送的数据。注意：从机必须在主机拉低片选信号之前，准备好待发送的数据。

B. 当 SPI_CSTAT[1]为 1 时，表示收到数据。如果需要接收数据，则通过 CPU 或 DMA 从 SPI_RXREG 寄存器读取已接收的数据。

3. 处理中断

如果有中断发生，则读取 SPI_MINTSTAT 寄存器，判断中断发生的原因。处理完成之后，给 SPI_INTCLR 寄存器的相应位写 1，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。

4. 结束传输
   A. 由于主机控制时钟和片选信号，所以传输何时结束也由主机控制。当字符拼接功能有效时，从机也可等待 SPI_INTSTAT[4]变为 1，表示已经接收到 SPI_RXDNR 寄存器设定量的 SPI 字符，即接收完成。同时，配置 SPI_GCTL[4]置 0。
   B. 传输结束后，配置 SPI_GCTL[0]置 0，关闭 SPI 模块。
   C. 配置系统寄存器中外设时钟和复位相关的寄存器，关闭时钟，恢复复位。

31.7.2.2 从机半双工接收

1. 配置寄存器
   A. 配置系统寄存器中外设时钟、复位和引脚复用【从机使用 MISO 进行通信】相关的寄存器，开启时钟，释放复位。
   B. 配置 SPI_GCTL 寄存器，注意 SPI_GCTL[4]和 SPI_GCTL[3]暂时不要置位，SPI_GCTL[11]、SPI_GCTL[0]置 1，SPI_GCTL[2]置 0，其他位按需配置。
   C. 配置 SPI_CCTL 寄存器，设置时序模式和数据格式。
   D. 给 SPI_INTCLR 寄存器写 0xFF，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。
   E. 配置 SPI_INTEN 寄存器，可选择设置中断。
2. 传输数据
   A. 设置 SPI_GCTL[12]为 0，SPI_GCTL[3]为 0，SPI_GCTL[4]为 1，再置位 SPI_GCTL[0]为 1。
   B. 当 SPI_CSTAT[1]为 1 时，表示收到数据。如果需要接收数据，则通过 CPU 或 DMA 从 SPI_RXREG 寄存器读取已接收的数据。
3. 处理中断

   如果有中断发生，则读取 SPI_MINTSTAT 寄存器，判断中断发生的原因。处理完成之后，给 SPI_INTCLR 寄存器的相应位写 1，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。
4. 结束传输
   A. 由于主机控制时钟和片选信号，所以传输何时结束也由主机控制。
   B. 传输结束后，配置 SPI_GCTL[0]置 0，关闭 SPI 模块。
   C. 配置系统寄存器中外设时钟和复位相关的寄存器，关闭时钟，恢复复位。

31.7.2.3 从机半双工发送

1. 配置寄存器
   A. 配置系统寄存器中外设时钟、复位和引脚复用【从机使用 MISO 进行通信】相关的寄存器，开启时钟，释放复位。
   B. 配置 SPI_GCTL 寄存器，注意 SPI_GCTL[4]和 SPI_GCTL[3]暂时不要置位，SPI_GCTL[11]、SPI_GCTL[0]置 1，SPI_GCTL[2]置 0，其他位按需配置。
   C. 配置 SPI_CCTL 寄存器，设置时序模式和数据格式。
   D. 给 SPI_INTCLR 寄存器写 0xFF，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。
   E. 配置 SPI_INTEN 寄存器，可选择设置中断。
2. 传输数据
   A. 如果需要发送数据，则通过 CPU 或 DMA 给 SPI_TXREG 寄存器写入需要发送的数据。注意：从机必须在主机拉低片选信号之前，准备好待发送的数据。
   B. 设置 SPI_GCTL[12]为 1，SPI_GCTL[3]为 1，SPI_GCTL[4]为 0，再置位 SPI_GCTL[0]为 1。
   C. 往发送数据寄存器 SPI_TXREG 写入数据。
   D. 当 SPI_CSTAT[0]为 1 时，表示数据已经发送完成。
   E. 设置 SPI_GCTL[3]为 0。
3. 处理中断
   如果有中断发生，则读取 SPI_MINTSTAT 寄存器，判断中断发生的原因。处理完成之后，给 SPI_INTCLR 寄存器的相应位写 1，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。
4. 结束传输
   A. 由于主机控制时钟和片选信号，所以传输何时结束也由主机控制。建议从机等待 SPI_INTSTAT[6]变成 1，或者等待 SPI_CSTAT[0]变成 1，表示发送器内的数据已经全部发送出去。
   B. 传输结束后，配置 SPI_GCTL[0]置 0，关闭 SPI 模块。
   C. 配置系统寄存器中外设时钟和复位相关的寄存器，关闭时钟，恢复复位。

31.7.2.4 从机单工发送

1. 配置寄存器
   A. 配置系统寄存器中外设时钟、复位和引脚复用相关的寄存器，开启时钟，释放复位。
   B. 配置 SPI_GCTL[0]和 SPI_GCTL[3]置 1，SPI_GCTL[2]置 0，其他位按需配置。

C. 配置 SPI_CCTL 寄存器，设置时序模式和数据格式。

D. 给 SPI_INTCLR 寄存器写 0xFF，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。

E. 配置 SPI_INTEN 寄存器，设置中断使能。

F. 当字符拼接功能有效时，配置 SPI_TXDNR 寄存器，设置待发送的字符数量。

2. 传输数据

A. 如果需要发送数据，则通过 CPU 或 DMA 给 SPI_TXREG 寄存器写入需要发送的数据。注意：从机必须在主机拉低片选信号之前，准备好待发送的数据。

B. 当 SPI_CSTAT[1]为 1 时，表示收到数据。如果需要接收数据，则通过 CPU 或 DMA 从 SPI_RXREG 寄存器读取已接收的数据。

3. 处理中断

如果有中断发生，则读取 SPI_MINTSTAT 寄存器，判断中断发生的原因。处理完成之后，给 SPI_INTCLR 寄存器的相应位写 1，清除 SPI_INTSTAT 寄存器和 SPI_MINTSTAT 寄存器的中断状态。

4. 结束传输

A. 由于主机控制时钟和片选信号，所以传输何时结束也由主机控制。建议从机等待 SPI_INTSTAT[6]变成 1，或者等待 SPI_CSTAT[0]变成 1，表示发送器内的数据已经全部发送出去。

B. 传输结束后，配置 SPI_GCTL[0]置 0，关闭 SPI 模块。

C. 配置系统寄存器中外设时钟和复位相关的寄存器，关闭时钟，恢复复位。

32 四线 SPI 控制器（QSPI）

32.1 概述

QSPI控制器可向串行Flash器件提供访问权限。支持高性能单线、双线和四线SPI标准接口。

32.2 主要特性

- 存储器映射的直接操作模式，用来进行Flash数据传输和执行Flash存储器中的编码
- 软件触发的间接操作模式，用来进行延迟时间短和非处理器密集型的Flash数据传输
- 软件APB可访问的Flash控制寄存器组可执行任何Flash命令，包括一次数据传输可高达8个字节
- 支持XIP (Execute in Place)，有时也指连续模式
- 支持单线，双线，四线I/O指令
- 器件大小可编程
- 写保护区域可编程，可阻止系统写入生效
- 传输事务之间的延时可编程
- 传统模式下允许软件直接访问底层发送和接收FIFOs，旁路较高层流程
- 支持独立的异步的SPI通信参考时钟
- 串行时钟可配置极性
- 可编程波特率发生器
- 包含可提升高速读数据捕获机制的特性
- 可选择使用调整时钟来进一步提升读数据捕获
- 可编程中断生成
- 支持1个外部器件

32.3 系统框图

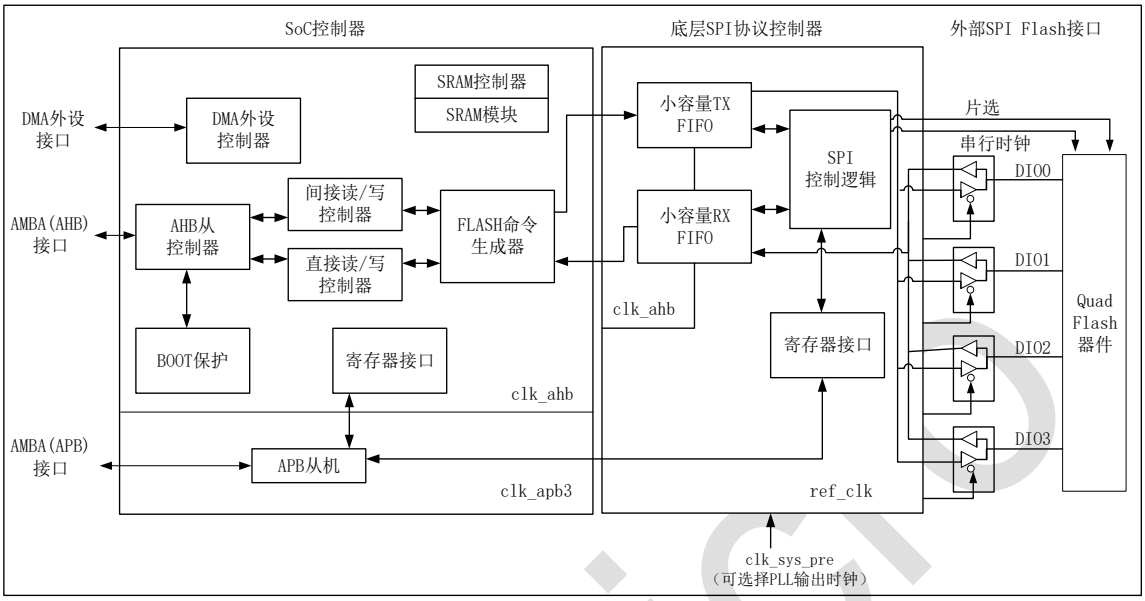


图 32-1：QSPI 系统框图

32.4 管脚说明

表 32-1：QSPI 管脚说明

功能管脚	复用管脚	方向	功能描述
QSPI_SCK	PE10，PB10	Output	串行时钟输出
QSPI_DIO0	PE12，PD4	Input/Output	串行数据输入输出信号
QSPI_DIO1	PE13，PD5	Input/Output	串行数据输入输出信号
QSPI_DIO2	PE14，PD6	Input/Output	串行数据输入输出信号
QSPI_DIO3	PE15，PD7	Input/Output	串行数据输入输出信号
QSPI_CSN	PA2，PE11，PD3	Output	从设备选择线

32.5 功能描述

32.5.1 AHB 控制接口

AHB从控制器可验证接收到的AHB访问；对无效请求作出响应；进行任意字节或者半字的重排序；屏蔽违反写保护规则的写操作（仅限直接访问）；向直接访问控制器或者间接访问控制器转发传输请求。

32.5.1.1 AHB 接口

AHB接口遵循ARM的AMBA 3 AHB-Lite协议规格。不支持传输锁定（HMASLOCK）和采用保护控制信号（HPROT）的传输。AHB的数据位宽为32位。因此，只允许字节，半字和字访问。关于写操作，只支持递增突发（incrementing bursts），接收到的wrapping写突发将会产生错误。支持INCR16，INCR8，INCR4，INCR和SINGLE的突发类型。关于读操作，所有突发类型，包括WRAPS，均支持。如果突发传输因互连突发终止而造成提前终止，或又如果在突发传输内从机发生错误访问，从机仍将正确运行。

32.5.1.2 AHB 地址重映射

QSPI控制器不会对接收到的错误地址进行具体的地址解码，但有AHB地址解码器。如果使能，QSPI Flash控制器能够检测到每一个独立器件的地址范围，基于AHB的地址，做出有效器件自动切换的选择。接收到的AHB地址默认直接映射到串行发送至Flash器件上的地址。如果Flash器件有24位地址，将会发送AHB地址的低24位。重映射功能将AHB总线上的地址映射为DDRESS+N，N为地址重映射寄存器（QSPI_RAR）中的值。重映射功能可通过QSPI配置寄存器（QSPI_ CR[0]）使能。当软件需要将BOOT代码移动至另一个Flash区域时，需要用到重映射功能。

32.5.1.3 写保护

为保护Flash器件，写保护功能可通过硬件实现，也可通过软件控制。任何检测到的AHB写操作，指向受保护的Flash区域，将通过HRESP错误输出产生一个错误信号。 Flash器件上的区域，若由多个Flash“块”定义而成，且起始“块”有特定编号，则该区域可受到写保护。提供3个可编程寄存器。QSPI写保护低位寄存器（QSPI_WPLR）定义了需要保护区域底部的Flash块。QSPI写保护高位寄存器（QSPI_WPHR）定义了需要保护区域顶部的Flash块。QSPI写保护配置寄存器（QSPI_WPCR）为一个2位控制寄存器。QSPI_WPCR[0]可将受保护的区域进行反转，使得该区域成为Flash存储器中唯一不受保护的区域。QSPI_WPCR[1]为写保护使能位。当QSPI_WPCR[1]为低电平，Flash器件不受保护。 AHB控制器需将接收到的AHB地址映射到相关Flash块。一个块的大小可通过器件容量配置寄存器（QSPI_DSCR）设置，范围为1~65K字节。

32.5.1.4 访问转发

为了合法访问，AHB控制器会将所有的AHB访问转发给2个访问控制器中的一个：直接访问控制器或间接访问控制器。假设直接访问控制器已通过QSPI配置寄存器（QSPI_CR）使能，则默认所有AHB访问将会被转发至直接访问控制器。在转发任何访问至间接访问控制器前，必须由软件先配置。该过程在间接访问控制器章节进行了全面的讲解。如果直接访问控制器禁止，任何不能转发

至间接访问控制器的AHB访问将被立即停止，并产生HRESP错误。如果直接访问控制器使能，同样的访问将由直接访问控制器转发和服务。

32.5.1.5 顺序访问检测和突发长度

为了使性能最大化，将不使用AHB接口信号htrans来识别顺序和非顺序访问。Htrans的问题在于它将一个新的AHB突发传输启动时的所有访问都识别为非顺序访问，1K边界以后的所有访问也识别为非顺序访问。取而代之的方法是，通过比较当前地址和上一个地址来检测非顺序访问。如果访问方向（写/读）已改变，则视为非顺序访问。如果读访问的大小（字节/半字/字）已改变，则同样视为非顺序访问。如果AHB地址解码器使能并检测到器件之间的切换，则需生成新的SPI传输，因为需要初始化当前选中的器件。该情况需视为非顺序访问，尽管地址为连续地址。否则，如果当前传输地址与上一个传输地址为连续地址，则视为顺序访问。另外，控制器不采用AHB的hburst信号来识别突发长度。突发传输将继续，直到收到非顺序访问。

32.5.2 直接访问控制器（DAC）

直接访问是指AHB访问直接触发对Flash存储器进行读或写操作。该操作为存储器映射，可访问并直接执行外部Flash存储器中的编码。任何接收到的不在可编程间接触发区域内的AHB访问都被假想为直接访问，并由直接访问控制器响应。注意，使用直接访问控制器的访问将不使用嵌入SRAM。当进行读或写突发操作时，AHB将会节流，等待状态的数量取决于控制器的延时。延时已设计的尽可能小。当XIP读指令使能时，延时将保持最小值。当响应AHB读操作时，DAC将会发送一个额外的下游访问，而不仅仅是对AHB单个burst的转发。该访问将不会显现于系统接口。该操作为一个预读操作，确保底层SPI核运行于最大带宽。这里定义的AHB突发不同于由AMBA定义的AHB突发。这里的AHB突发定义如下：

1. AHB突发的第一次访问由以下定义：
   - 非连续AHB访问（基于地址比较来判断是否为非连续，而非基于htrans）
   - 非连续或连续AHB访问（当下游模块处于空闲状态）
2. AHB突发的最后一次访问由连续AHB访问定义，优先于上面1中定义的新突发。
3. AHB突发的大小为AHB访问的数量，从第一次访问到最后一次访问。
4. 每个AHB突发的DAC请求的数量 = AHB突发大小 + 1

关于AHB写操作，直接访问控制器会触发一系列的写命令（类似于读操作的处理方式），尽管DAC写请求的数量等于接收到的AHB写请求的数量。写操作时，AHB控制器将确保Flash突发不会超出Flash页边界。当检测到页边界时，只有到边界为止的字节访问数会被转发。一个跨页边界的连续直接写请求必须检测为到下游模块的非连续请求，使得下游控制器强制Flash器件进入自定时页程序周期。 内核支持跨页分开写，仅限于字对齐的地址。 如果系统发送连续写操作时延迟太久，Flash

写周期可能提前启动，减少器件的有效寿命。注意，如若系统不能保证及时提供需要写入的数据，则采用间接写操作来避免这个问题。Flash擦操作由软件使用编程接口触发。一旦页编程周期启动，在允许后续AHB访问完成之前，QSPI Flash控制器将自动轮询写周期直到完成。该操作通过将后续AHB直接访问控制在等待状态来实现。

32.5.3 间接访问控制器（INDAC）

32.5.3.1 间接读控制器

间接操作的目的在于，无需AHB访问触发的情况下，从Flash存储器读取有效字节数。间接操作由软件通过间接读传输控制寄存器（QSPI_IRTR），间接读传输数据深度阈值寄存器（QSPI_IRTWR），间接读传输起始地址寄存器（QSPI_IRTSAR）和间接读传输字节数寄存器（QSPI_IRTNR）控制和触发。该模块与底层SPI协议状态机模块通信，展开有效优化的Flash读突发操作，将读数据放在本地SRAM模块中，为和外部AHB主机进行快速及低延时传输做准备。默认状态下，间接读控制器禁止。在使能前，软件必须配置读取数据的多少和起始地址。间接读传输起始地址寄存器（QSPI_IRTSAR）定义起始地址，间接读传输字节数寄存器（QSPI_IRTNR）定义字节数。一次可同时配置两个间接操作。第二个间接操作可在第一个操作过程中被触发。上一次间接操作完成与下一次间接操作开始之间允许一段短时间的周转。间接操作读取的字节总数不受SRAM大小限制。SRAM的大小仅会限制发送至DMA的请求数量（在DMA外设接口模块使能的情况下）。发生SRAM上溢时，控制器将压回Flash读直到SRAM有空余空间。通过完成当前读突发，在SPI接口处回压读操作，等待直到SRAM有剩余空间，在上一个突发操作终止的地址发送新的读突发操作。通过发送AHB读到QSPI控制器，总线主机能够获取控制器从外部Flash存储器读取的数据。接收的读访问AHB地址必须在AHB间接触发地址（通过间接AHB地址触发寄存器（QSPI_IATR）配置）到（AHB间接触发地址 + 2**（间接AHB触发地址范围）- 1）的范围内。该范围的默认值等于16，可有效地处理16次及以下的突发操作。实际的AHB地址必须在间接范围内同意SRAM为源。每个有效的AHB间接读将会引起内部SRAM出栈，从而将AHB地址从Flash地址中分离出来非直接映射。因此，AHB间接触发地址与Flash地址毫无任何关系。触发任何有效间接读之后，SRAM被视为源，而非Flash存储器阵列。间接读的Flash地址由间接读传输起始地址寄存器（QSPI_IRTSAR）获取。假设请求读取的数据已经在SRAM时，QSPI控制器一旦收到AHB访问地址，SRAM中数据将被取走，这就实现了读突发的最小延时响应。当数据从SRAM中读取，QSPI控制器将释放SRAM中的相关资源。如果接收到的AHB读访问的地址不在上述范围内，则该访问将不会由间接控制器完成，而是由直接访问控制器完成。 如果接收到的AHB读访问的地址在上述范围内，但读取的数据不存在于SRAM中，则AHB将处于等待状态直到从Flash读取到数据并压栈至SRAM中。 如果接收到的AHB读突发的访问元素横跨AHB间接触发范围，则间接触发范围内的访问将由间接访问控制器处理，其余部分由直接访问控制器处理。注意，这可能是软件配置错误。 总线主机仅允许发送32位AHB读，直到间接传输结束，

这可以减少SRAM控制逻辑的复杂程度。在读最后非32位对齐数据时，总线主机可发送16位（半字）或者字节访问来完成此次传输。同样总线主机可以始终发送32位读，末尾非32位对齐的数据高位用0填充。

理想状态下，在执行读操作时，SRAM将保持满的状态。SRAM的数据深度状态（Fill level）可直接由软件直接读取（SRAM数据深度状态寄存器（QSPI_SFLR））。如果DMA外设接口控制器使能，将自动请求外部DMA通过AHB以有效数据块（chunks of data）的形式从SRAM获取数据，每个数据块皆是整体间接传输的一部分。 通过设置QSPI_IRTR[1] ，可随时取消间接操作。 任何总线主机可发起间接访问。DMA总线外设接口可分担部分软件开销，有效管理数据传输。另外一个选择是：通过APB寄存器，软件直接访问SRAM数据深度状态，决定何时从SRAM获取数据。当DMA外设接口禁止，提供了可通过间接读传输数据深度阈值寄存器（QSPI_IRTWR）访问的数据深度阈值。当SRAM数据深度状态超过该数据深度阈值（watermark），会产生一个中断。如果数据深度阈值大于0，当数据的最后一个字节被QSPI控制器读取并放置于SRAM中，即便实际的SRAM数据深度状态还未超数据深度阈值，同样会产生数据深度阈值中断。该功能有助于避免软件追踪已被读取的数据量和复位间接读传输中最后几个字节的数据深度阈值。注意，该数据深度阈值寄存器是一个两用寄存器。当DMA外设接口使能，由硬件来控制DMA请求发送的速率。当DMA外设接口禁止，其行为如上文中描述。为了解间接操作的状态，另外还提供两个中断源。第一个：当完成一个间接操作时，会产生一个中断。第二个：如果发出间接读操作请求，但由于QSPI控制器中已缓存2个间接操作，而未被接受时，会产生中断。设置QSPI_IRTR[0]来启动间接读操作，QSPI_IRTR[2]可用来检查状态。

间接读传输流程

- 当DMA外设控制器使能，需遵循下列流程：
  1. 设置QSPI配置寄存器（QSPI_CR）。
  2. 通过间接读传输数据深度阈值寄存器（QSPI_IRTWR）设置SRAM数据深度阈值。
  3. 通过间接读传输起始地址寄存器（QSPI_IRTSAR）设置间接传输的Flash起始地址。
  4. 在间接读传输字节数寄存器（QSPI_IRTNR）中设置需传输字节的数量。
  5. 在间接AHB地址触发寄存器（QSPI_IATR）中设置间接传输AHB触发地址。
  6. 在间接触发地址范围寄存器（QSPI_ITARR）中设置间接传输AHB触发地址范围。
  7. 在DMA外设配置寄存器（QSPI_DMACR）中设置单次和突发传输的字节数。
  8. 设置QSPI_IRTR[0]= 1来触发间接读访问。
  9. 通过间接读传输控制寄存器（QSPI_IRTR[5]）来轮询间接读操作的完成状态。注意，该位为写清零（write-to-clear）。或者通过产生间接完成中断来判断间接读操作是否完成。
  10. 在相同的寄存器中读取已完成的间接读操作数量。

- 当DMA外设控制器禁止，需遵循下列流程：
  1. 设置QSPI配置寄存器（QSPI_CR）。
  2. 通过间接读传输起始地址寄存器（QSPI_IRTSAR）设置间接传输的Flash起始地址。
  3. 在间接读传输字节数寄存器（QSPI_IRTNR）中设置需传输字节的数量。
  4. 在间接AHB地址触发寄存器（QSPI_IATR）中设置间接传输AHB触发地址。
  5. 在间接触发地址范围寄存器（QSPI_ITARR）中设置间接传输AHB触发地址范围。
  6. 如果使用数据深度阈值中断功能，当数据深度状态超出间接读传输数据深度阈值寄存器（QSPI_IRTWR）设置的阈值，会产生中断。设置数据深度阈值有助于提示软件何时读取间接读传输的下一个部分。注意，如果数据深度阈值设置为一个非0值，即便数据深度阈值高于实际的数据深度状态，一旦获取间接传输的最后一个字节并存放于SRAM中，就会产生数据深度阈值中断。
  7. 设置QSPI_IRTR[0] = 1来触发间接读访问。
  8. 如果使用数据深度阈值中断功能，则等待数据深度阈值中断发生。否则，轮询SRAM的数据深度状态来决定什么时候SRAM中有足够的数据来触发获取AHB数据。
  9. 从SRAM中读取期望的数据量。如果还要获取更多数据才能完成间接读传输，则回到步骤8，否则，继续到10。
  10. 通过间接读传输控制寄存器（QSPI_IRTR[5] 来轮询间接读操作的完成状态。
  11. 当间接读操作完成，将产生间接完成中断。

32.5.3.2 间接写控制器

间接写操作的目的是以最有效的方式将大量的数据从处理器或DMA写到Flash存储器。间接传输在Flash器件内部执行尽可能少的写周期，因此可最大化器件的寿命。间接写操作在软件上可视为间接读的相反过程，通过间接写传输控制寄存器（QSPI_IWTR），间接写传输数据深度阈值寄存器（QSPI_IWTWR），间接写传输起始地址寄存器（QSPI_IWTSAR），间接写传输字节数寄存器（QSPI_IWTNR）来控制和触发。该模块通过外部AHB主机来等待写数据的传输，在与现有的传统SPI IP内核进行通信前，写数据会存放在自带的SRAM中。默认状态下，间接写控制器禁止。在使能前，软件必须配置数据的多少和起始地址。间接写传输起始地址寄存器（QSPI_IWTSAR）定义起始地址，间接写传输字节数寄存器（QSPI_IWTNR）定义字节数。一次性可同时编程两个间接操作。第二个间接操作可在第一个操作过程中被触发。上一次间接操作完成与下一次间接操作开始之间允许一段短时间的周转。间接写序列类似于间接读序列。间接操作中，写的字节总数不受SRAM的大小限制。SRAM的大小仅会限制能从外部AHB主机接收的数据数量。当总线主机为DMA，控制器通过DMA外设接口要求的数据量不会超出SRAM的现有Fill level。但是，这并不保证DMA或者其他总线主机不会试图发送更多的数据，超出SRAM所能接收的量。当发生SRAM上溢时，控制器会在等待状态下回压AHB。注意，通过SRAM数据深度状态寄存器（QSPI_SFLR）可读取SRAM的数据深度状

态，避免发生上溢。

总线主机会提供写数据，并通过发送AHB写操作发送至QSPI。接收的写访问AHB地址必须在AHB间接触发地址（通过间接AHB地址触发寄存器（QSPI_IATR）配置）到（AHB间接触发地址 + 2**（间接AHB触发地址范围）- 1）的范围内。该范围的默认值等于16，可有效地处理16次及以下的突发操作。另外，对于压栈连续地址序列也没有严格要求。实际的AHB地址必须在间接范围内同意SRAM为源。每个有效的AHB间接写将会引起内部SRAM出栈，从而将AHB地址从Flash地址中分离出来。因此，AHB间接触发地址与Flash地址毫无任何关系。触发任何有效间接写之后，SRAM被视为源，而非Flash存储器阵列。间接写的Flash地址由间接写传输起始地址寄存器（QSPI_IWTSAR）获取。假设在QSPI控制器接收AHB访问的时候，SRAM的状态为非满，则会以最小延时向SRAM中压栈数据。如果接收到的AHB写访问的地址不在上述范围内，则该访问将不会由间接控制器完成，而是由直接访问控制器完成。如果接收到的AHB写访问在上述范围内，但SRAM状态为满，则AHB将处于等待状态直到一部分数据或者全部数据从SRAM压栈到Flash中。如果接收到的AHB写突发的访问元素横跨AHB间接触发范围，则间接触发范围内的访问将由间接访问控制器处理，其余部分由直接访问控制器处理。注意，这可能是软件配置错误。总线主机仅允许发送32位AHB写，直到间接传输结束，用来减少SRAM控制逻辑的复杂程度。在写最后一个字时，总线主机可发送一个32位，16位（半字）或者一个字节来结束此次传输。如果最后一次传输少于4个字节，主机依旧可发送32位的数据传输，多余的字节将被丢弃。当SRAM中的字节数量等于或大于一页Flash的容量（默认256字节）或当SRAM中的字节为当前执行的间接传输的所有剩余字节，控制器会向命令发生器发起写突发操作。通过设置QSPI_IWTR.WRDIS，可随时取消间接操作。任何总线主机可发起间接访问。DMA总线外设接口可分担部分软件开销，有效管理数据传输。另外一个选择是：通过APB寄存器，软件直接访问SRAM数据深度状态，决定何时向SRAM写数据。当DMA外设接口禁止，提供了可通过间接写传输数据深度阈值寄存器（QSPI_IWTWR）访问的数据深度阈值。当SRAM数据深度状态低于该深度阈值，会产生一个中断。注意，该数据深度阈值寄存器为一个两用寄存器。当DMA外设接口使能，由硬件来控制DMA请求发送的速率。在该模式下，因为QSPI不会发起写操作除非SRAM中有至少一页Flash，或者间接传输的剩余字节在SRAM中，DMA模式下的数据深度阈值需设置为大于或等于1个Flash页。为了解间接操作的状态，另外还提供两个中断源。第一个：当完成一个间接操作时，会产生一个中断。第二个：如果发出间接写操作请求，但由于QSPI控制器中已缓存2个间接操作，而未被接受时，会产生中断。设置QSPI_IWTR[0]来启动间接写操作，QSPI_IWTR[5]可用来检查状态。

间接写传输流程

- 当DMA外设控制器使能，需遵循下列流程：
  1. 通过间接写传输起始地址寄存器（QSPI_IWTSAR）设置间接传输的Flash起始地址。
  2. 在间接写传输字节数寄存器（QSPI_IWTNR）中设置需传输字节的数量。

3. 在间接AHB地址触发寄存器（QSPI_IATR）中设置间接传输AHB触发地址。
4. 在间接触发地址范围寄存器（QSPI_ITARR）中设置间接传输AHB触发地址范围。
5. 在DMA外设配置寄存器（QSPI_DMACR）中设置DMA单发和突发传输的字节数。
6. 可选：设置间接写传输数据深度阈值寄存器（QSPI_IWTWR）来控制DMA请求的发送速率。
7. 设置QSPI_IWTR[0] = 1来触发间接写访问。
8. QSPI控制器将利用DMA请求接口来要求DMA传输数据。
9. 通过间接写传输控制寄存器（QSPI_IWTR[5]）来轮询间接写操作的完成状态。注意，该位为写清零（write-to-clear）。在相同的寄存器中读取已完成的间接写操作数量。
10. 当间接写操作完成，会产生间接完成中断。

- 当DMA外设控制器禁止，需遵循下列流程：
  1. 通过间接写传输起始地址寄存器（QSPI_IWTSAR）设置间接传输的Flash起始地址。
  2. 在间接写传输字节数寄存器（QSPI_IWTNR）中设置需传输字节的数量。
  3. 在间接AHB地址触发寄存器（QSPI_IATR）中设置间接传输AHB触发地址。
  4. 在间接触发地址范围寄存器（QSPI_ITARR）中设置间接传输AHB触发地址范围。
  5. 功能上，软件可简单的在一个块内将所有数据写到SRAM中。但是，如果写字节数超过SRAM的大小，SRAM极有可能会满，导致QSPI花大量时间回压系统AHB总线。该时间基于Flash的数据速率和器件的页写时间。为避免所有写数据在一个块内发送，软件可利用数据深度阈值中断在合适的时间一次发送一页数据。也可选择，软件直接轮询SRAM数据深度状态寄存器（QSPI_SFLR）来识别SRAM的空状态，再决定下一部分传输发送的最合适时间。
  6. 如果使用数据深度阈值中断，当数据深度状态低于深度阈值，将产生中断。数据深度阈值的设置范围为0 ~页容量。举例：如果页容量为256字节，数据深度阈值的合理设置为10到250之间。当数据深度状态下降到设定值以下，将触发中断。设置数据深度阈值有助于提示软件何时向SRAM写入间接写传输的下一页数据。
  7. 设置QSPI_IWTR[0]= 1来触发间接写访问。
  8. 如果当前间接传输剩余的字节数大于一个Flash页，则向SRAM中写入一个Flash页数据；否则，将所有剩余数据发送至SRAM。
  9. 如果间接传输中所有的数据都已发送给SRAM，跳至步骤11，等待完成状态。如果还有更多的数据需要传输，则：
     - 如果使用数据深度阈值中断，则等到中断发生。
     - 使用SRAM数据深度状态来判断发送数据的合适时间。
  10. 循环回步骤8。
  11. 可选：间接写操作的完成状态可通过间接写传输控制寄存器（QSPI_IWTR）查询。
  12. 当间接写操作完成，会产生间接完成中断。

32.5.3.3 间接访问队列

软件允许间接写控制器和间接读控制器有2个间接传输队列等候。支持前一个间接操作完成和后一个操作开始前有短暂周转时间。如果等候队列超出2个，则会产生中断。软件上，间接访问队列通过短时间内触发两次QSPI_IRTR[0]或QSPI_IWTR[0]来实现。在每一次触发QSPI_IRTR[0]或QSPI_IWTR[0]前，间接传输起始地址寄存器（QSPI_IRTSAR/QSPI_IWTSAR）和间接传输字节数寄存器（QSPI_IRTNR/QSPI_IWTNR）必须完成设置。因这些寄存器将进行周期性更新，在间接传输的周期过程中，硬件需要对这些寄存器保持采样的状态。内部寄存器块一次只发送一个间接起始触发给关键数据路径块。间接访问控制器中有2个独立的数据路径块，可接收和独立采样间接起始触发信息。第一个数据路径块存在于SRAM的AHB端。当进行间接读时，为读接口；当进行间接写时，为写接口。第二个数据路径块在SRAM的Flash端。当进行间接读时，为写接口；当进行间接写时，为读接口。这两个块可在不同的时间处理间接传输。举例：进行间接读操作时，当第一个传输中的最后一个字节已写入SRAM，SRAM中的Flash上的数据路径块即可开始处理第二个队列传输。开始第二个传输前，必须对间接传输起始地址寄存器（QSPI_IRTSAR/QSPI_IWTSAR）和间接传输字节数寄存器（QSPI_IRTNR/ QSPI_IWTNR）进行重采样。同样地，当第一个间接传输的所有的Flash数据都已从SRAM发送至AHB，AHB上的数据路径块也将重采样相同的寄存器。

32.5.3.4 间接传输：连续写读

间接写操作过程中，软件允许触发间接读操作。相似地，间接读操作过程中，也允许触发间接写操作。间接写操作优先级更高。

32.5.3.5 访问 SRAM

物理上，SRAM为一个单个端口模块。SRAM深度可配置。SRAM被划分为两块区域，低区域部分预留给间接读使用，高区域部分仅为间接写使用。两部分的容量可通过SRAM分块配置寄存器（QSPI_SPR）配置，用户可选择分配SRAM地址总线中多少的bit给间接读操作。默认地，该寄存器设置SRAM的一半为间接读控制器使用。为确保AHB读数据总线不是直接由SRAM读数据通过组合逻辑提供，间接读数据路径中包含了另一组寄存器。以下示例展示了在间接读和间接写之间，SRAM（和另一组寄存器）的分配方式。以下示例中，SRAM深度为8位，等同于256地址。

- 如果SRAM分块配置寄存器（QSPI_SPR）设置为0x00，256个地址分配给间接写，1个地址分配给间接读。
- 如果SRAM分块配置寄存器（QSPI_SPR）设置为0x01，255个地址分配给间接写，2个地址分配给间接读。
- 如果SRAM分块配置寄存器（QSPI_SPR）设置为0x02，254个地址分配给间接写，3个地址分配给间接读。

- 依次类推……
- 如果SRAM分块配置寄存器（QSPI_SPR）设置为0xFD，3个地址分配给间接写，254个地址分配给间接读。
- 如果SRAM分块配置寄存器（QSPI_SPR）设置为0xFE，2个地址分配给间接写，255个地址分配给间接读。
- 如果SRAM分块配置寄存器（QSPI_SPR）设置为0xFF，1个地址分配给间接写，256个地址分配给间接读。

注：避免将SRAM分块配置寄存器（QSPI_SPR）设置为0xFF或者0x00，因为仅SRAM数据深度状态底部的8位才能通过软件访问。如果间接读或间接写的数据深度达到256，当读取数据深度时，会显示为0。

有4个SRAM源，它们都在单个SRAM端口上进行仲裁和复用。高达3个源可在任意时间访问该端口。

- 间接写，写源，位于SRAM的AHB上。
- 间接写，读源，位于SRAM的Flash上。
- 间接读，写源，位于SRAM的Flash上。
- 间接读，读源，位于SRAM的AHB上。

优先级的仲裁方案如下：

表 32-2：SRAM 访问优先级

访问方式		SRAM访问优先级
间接写	写入SRAM（从系统AHB）	第3优先级（不包括AHB读请求）
	读取SRAM（从QSPI控制器）	第2优先级
间接读	写入SRAM（从QSPI控制器）	第1优先级
	读取SRAM（从系统AHB）	第3优先级（不包括AHB写请求）

在间接读操作过程中（SRAM中的Flash）除写端口外，驱动4个源的逻辑不能被认为是单个周期完成。为避免数据丢失，在间接读过程中，必须允许写SRAM可以立即完成。因此，该端口拥有最大优先级。

32.5.4 DMA 外设控制器

外设接口用来触发外部DMA进行短延时AHB数据突发访问。DMA外设接口仅用于间接操作模式，数据缓存于自带的SRAM可更快速的响应AHB请求，并使内核在一个较长的周期内进行底层Flash传输。支持2个DMA请求，一个用于间接读控制器，另一个用于间接写控制器。关于间接读控制器，当数据已从Flash获取并写入自有SRAM后，QSPI控制器仅发送DMA请求。关于间接写控制器，当触发传输后，QSPI控制器会立即发送DMA请求并一直持续发送直到整个间接写传输完成。间接传输数据深度阈值寄存器（QSPI_IRTWR/QSPI_IWTWR）可改变发送请求的速率。

32.5.4.1 操作顺序

当间接操作被触发，DMA外设控制器可见所有Flash存储器的传输数据（以字节为单位）。控制器会将这些数据分为DMA突发请求和单次请求：所有字节数除以突发请求中设置的字节数，余下的除以单次请求中的字节数。软件需确保在这些除法之后无余数。举例，如果从Flash要读取的数据总字节为512，SRAM固定容量为256字节，软件配置的突发传输字节数为256，则当前256个字节已缓冲，SPI控制器将会触发DMA突发请求。当又有256个字节在SRAM中缓冲完成后才会触发第二次突发请求。由于SRAM本身容量为256字节，意味着在发送下一个请求前，DMA会从SRAM中获取所有内容。SRAM数据深度状态对DMA外设控制器可设计为可见。关于间接读，会基于以下状态机发送请求给外部DMA。

1. 等待START触发。
2. 如果下一个需发送BURST请求，等待直到下面两个条件均为TRUE。
   A. SRAM数据深度状态大于或等于QSPI_DMACR[11:8]设置的字节数。
   B. SRAM数据深度状态大于或等于间接传输数据深度阈值寄存器（QSPI_IRTWR/QSPI_IWTWR）中的值。

   当TRUE时，执行以下：
   A. 发送BURST。
   B. 计算数据深度状态期望值 = SRAM fill level – num_bytes_requested_to_DMA。
   C. 如果下一个需发送的为BURST请求，跳至4。
   D. 如果下一个需发送的为SINGLE请求，跳至5。
   E. 其他，跳至1。
3. 如果下一个需发送的为SINGLE请求，等待直到下面条件均为TRUE。

   SRAM数据深度状态大于或等于QSPI_DMACR[3:0]设置的字节数。

   当TRUE时，执行以下：
   A. 发送SINGLE。
   B. 计算数据深度状态期望值 = SRAM fill level – num_bytes_requested_to_DMA。
   C. 如果下一个需发送的为SINGLE请求，跳至5。
   D. 其他，跳至1。
   E. 下一个需发送的为BURST请求。如果（SRAM fill level – num_bytes_requested_to_DMA）小于QSPI_DMACR[11:8]设置的字节数，跳至2。其他，执行以下：
   F. 发送BURST。
   G. 计算数据深度状态期望值 = SRAM fill level – num_bytes_requested_to_DMA。
   H. 如果下一个需发送的为BURST请求，跳至4。
   I. 如果下一个需发送的为SINGLE请求，跳至5。

J. 其他，跳至1。

4. 下一个需发送的为SINGLE请求。如果（SRAM fill level – num_bytes_requested_to_DMA）小于QSPI_DMACR[3:0]设置的字节数，跳至3。其他，执行以下：

A. 发送SINGLE。

B. 计算数据深度状态期望值 = SRAM fill level – num_bytes_requested_to_DMA。

C. 如果下一个需发送的为SINGLE请求，跳至5。

D. 其他，跳至1。

关于间接写，会基于以下状态机发送请求给外部DMA。

1. 等待START触发。
2. 如果下一个需发送的为BURST请求，等待直到下面两个条件均为TRUE。

A. SRAM的剩余空间大于或等于QSPI_DMACR[11:8]设置的字节数。

B. SRAM数据深度状态小于或等于间接传输数据深度阈值寄存器（QSPI_IRTWR/QSPI_IWTWR）中的值。

当TRUE时，执行以下：

A. 发送BURST。

B. 计算数据深度状态期望值 = SRAM fill level + num_bytes_requested_to_DMA。

C. 如果下一个需发送的为BURST请求，跳至4。

D. 如果下一个需发送的为SINGLE请求，跳至5。

E. 其他，跳至1。

3. 如果下一个需发送的为SINGLE请求，等待直到下面条件均为TRUE。

RAM的剩余空间大于或等于QSPI_DMACR[3:0]设置的字节数。

当TRUE时，执行以下：

A. 发送SINGLE。

B. 计算数据深度状态期望值 = SRAM fill level + num_bytes_requested_to_DMA。

C. 如果下一个需发送的为SINGLE请求，跳至5。

D. 其他，跳至1。

4. 下一个需发送的为BURST请求。如果（SRAM剩余空间 – num_bytes_requested_to_DMA）小于QSPI_DMACR[11:8]设置的字节数，跳至2。其他，执行以下：

A. 发送BURST。

B. 计算数据深度状态期望值 = SRAM fill level + num_bytes_requested_to_DMA。

C. 如果下一个需发送的为BURST请求，跳至4。

D. 如果下一个需发送的为SINGLE请求，跳至5。

E. 其他，跳至1。

5. 下一个需发送的为SINGLE请求。如果（SRAM剩余空间– num_bytes_requested_to_DMA）小于QSPI_DMACR[3:0]设置的字节数，跳至3。其他，执行以下：

A. 发送SINGLE。

B. 计算fill level期望值 = SRAM fill level + num_bytes_requested_to_DMA。

C. 如果下一个需发送的为SINGLE请求，跳至5。

D. 其他，跳至1。

关于间接读操作，间接读传输数据深度阈值寄存器（QSPI_IRTWR）可定义控制器发送第一个DMA请求的最小数据深度状态，设置的值越大，在DMA获取之前，缓存在SRAM中的数据就越多。关于间接写操作，间接写传输数据深度阈值寄存器（QSPI_IWTWR）可定义控制器发送第一个DMA突发/单次请求的最大数据深度状态。间接传输数据深度阈值寄存器（QSPI_IRTWR/ QSPI_IWTWR）可使系统在短周期内选择性的集中于AHB传输。默认地，间接传输数据深度阈值寄存器（QSPI_IRTWR/QSPI_IWTWR）被复位为0，意味着外设控制器可尽快发送DMA请求。在间接读的情况下，意味着SRAM中有足够的数据进行突发请求或单次请求（如果剩余字节小于突发请求的设置值）。注意，如果SRAM为空，DMA外设不可发送间接读DMA请求；如果SRAM为满，DMA外设不可发送间接写请求。DMA外设接口可通过软件禁止。当接口禁止，将不会发送任何通道请求。注意，如果除了DMA以外的AHB主机将进行间接数据传输，DMA外设接口必须被禁止。

32.5.5 软件触发指令生成器（STIG）

直接和间接访问控制器用来进行数据传输。为进行擦除和访问易失性和非易失性配置寄存器，传统SPI状态寄存器，其他的状态/保护寄存器，需要有一个独立的软件控制器。软件触发指令生成器STIG由Flash命令控制寄存器（QSPI_FCR）通过设置命令发送去Flash器件来控制。这是一个通用寄存器，可用来执行Flash器件支持的扩展SPI协议的任何指令。指令如果不符合Flash器件规格可导致未知的控制器行为。读时，当命令已响应（QSPI_FCR[1]从1翻转为0），最多8个数据可放置于Flash命令读数据寄存器（QSPI_FCRLR/QSPI_FCRHR）。写时，写数据应该放置在Flash命令写数据寄存器（QSPI_FCWLR/QSPI_FCWHR）。实现以上功能的代码保存在QSPI的APB寄存器块内。

32.5.5.1 响应 STIG 请求

STIG请求会致使QSPI Flash控制器询问Flash命令控制寄存器（QSPI_FCR），决定该以什么方式发送多少字节去Flash器件。QSPI_FCR[31:24]指示要发送的指令，始终首先压栈。地址发送紧接在指令后面，地址大小同样在此寄存器中配置。地址本身保存在Flash命令地址寄存器（QSPI_FCAR）。Dummy字节此后发送（字节数量仍在Flash命令控制寄存器（QSPI_FCR）中配置）。写/读数据的字节数同样在Flash命令控制寄存器（QSPI_FCR）中配置。写时，最多可发送8个字节（保存于Flash命令写数据寄存器（QSPI_FCWLR/QSPI_FCWHR））。读时，当读数据已从Flash器件获取，QSPI

Flash控制器会将读数据保存在Flash命令读数据寄存器（QSPI_FCRLR/QSPI_FCRHR）。当QSPI Flash控制器开始响应STIG请求，QSPI_FCR[1]置1，表示正在执行命令中。当QSPI Flash控制器处于自动轮询状态，响应STIG请求会有一点不同。在一个程序操作过后，大部分器件都不可访问直到写操作完成。它们中的一些器件还有可能会停止编程页。QSPI_PFSR[8]指示了有效自动轮询状态。当发生STIG请求后，QSPI Flash控制器会立即发送合适的OPCODE去存储器。在响应STIG（自动轮询中）过程中，命令执行的状态位保持，其他位（比如ADDRESS或DUMMY位）都被禁止。另外还有一个可配置选项：在每个重复的轮询操作之间加入延时（延时由QSPI_WCR[31:24]定义）。该功能的作用是用来释放SPI带宽。

32.5.6 直接/间接访问控制器和 STIG 间的仲裁

当多个寄存器同时有效，一个简单的固定优先级仲裁方案可用来对每个接口进行仲裁，访问外部Flash。固定优先级定义如下，1为最高优先级。

1. 间接访问写
2. 直接访问写
3. STIG
4. 直接访问读
5. 间接访问读

等待响应时，每个控制器都会被回压。

32.5.7 SPI 命令转换

由直接访问控制器，间接访问控制器或STIG发送的请求会转换成字节传输序列，发送去下游。下例显示的是一个字节无序列读：

- INSTRUCTION OPCODE ADDRESS Mode Byte Dummy Bytes 1 byte of don’t care

当进行序列访问时，每一次读，都会在上面的序列最后向Flash器件压栈一个额外的字节。目的是为了每个传输字节之间没有空隙。 实际的发送去Flash器件的序列由要求的传输决定，是否该传输是非序列的还是序列的，是否器件已在XIP模式下配置完成，以及主要器件指令类型寄存器（器件读指令寄存器（QSPI_DRIR）/器件写指令寄存器（QSPI_DWIR））的状态。写时，在发送写序列前，Flash器件内的写使能锁存（WEL）必须为高电平。在通过直接或间接访问控制器（DAC/INDAC）触发写命令前，QSPI Flash控制器将自动发送写使能锁存命令。为提高灵活性和性能，用户可设置QSPI_DWIR[8]来关闭该功能。WREN的OPCODE为0x06，在器件之间共用。 当不再接收来自直接或间接访问控制器的写请求，并且所有请求都已发出，则Fash器件将自动开始页编程写周期。此时接收到的请求将会保持在等待状态，直到写周期完成。QSPI Flash控制器会自动轮询Flash器件的传

统SPI状态寄存器，来判断写周期是否完成：发送RDSR OPCODE去Flash器件，一直等待直到器件本身显示写周期已完成（直到Write In Progress bit[0]已清零，写使能锁存位也已清零）。WREN和RDSR器件指令为仅有的由两个由控制器在底层发送的指令。任何其他的具体指令都应发送去器件，通过STIG发送Flash命令来单独处理。

32.5.8 Flash 指令类型选择

为了发送正确的读写OPCODE，软件应初始化QSPI器件读指令寄存器（QSPI_DRIR）和QSPI器件写指令寄存器（QSPI_DWIR）。这两个寄存器中，可设置指令OPCODE、指令类型、边沿模式（DRR或SDR），和选择单个、双个还是4个引脚来进行地址和数据传输。为确保控制器可在复位状态中运行，寄存器会复位到兼容SIO器件的OPCODE。器件读指令寄存器（QSPI_DRIR）中包含指令类型字段，器件写指令寄存器（QSPI_DWIR）无指令类型字段。如果软件将指令类型设置为非0值，器件读指令类型寄存器（QSPI_DRIR）和器件写指令类型寄存器（QSPI_DWIR）中的地址传输类型和数据传输类型位无关。软件支持不太常见的Flash指令，在2或4通道发送OPCODE，地址和数据。注意，对于那些支持2或4通道发送OPCODE的指令的器件，这些指令的名字与Flash数据手册上的命名不一致。其中一个支持这些指令的器件为Numonyx（Micron）N25Q128。另外的读指令被称为DCFR和QCFR。写指令为DCPP和QCPP。下表以N25Q128作为参考，展示了软件该如何配置QSPI。

- 读

OPCODE	OPCODE发送通道数	ADDRESS/DUMMY/MODE发送通道数	DATA Bytes发送字节数	QSPI指令类型（器件读指令寄存器）	QSPI地址Xfer类型（器件读指令寄存器）	QSPI数据Xfer类型（器件读指令寄存器）
READ	1	1	1	0	0	0
FAST_READ	1	1	1	0	0	0
DOFR（DualO/PFastRead）	1	1	2	0	0	1
DIOFR（DualO/PFastRead）	1	2	2	0	1	1
QOFR（QuadO/PFast Read）	1	1	4	0	0	2
QIOFR（QuadO/PFast Read）	1	4	4	0	2	2
DCFR（DualCommand FastRead）	2	2	2	1	Don’t Care	Don’t Care
QCFR（QuadCommandFastRead）	4	4	4	2	Don’t Care	Don’t Care

- 写

OPCODE	OPCODE 发送通道数	ADDRESS / DUMMY/ MODE发送通道数	DATA Bytes发送字节数	QSPI指令类型（器件写指令寄存器）	QSPI地址Xfer类型（器件写指令寄存器）	QSPI数据Xfer类型（器件写指令寄存器）
PP	1	1	1	0	0	0
DIFP (Dual Input Fast Program)	1	1	2	0	0	1
DIEFP (Dual Input Extended Fast Program)	1	2	2	0	1	1
QIFP (Quad Input Fast Program)	1	1	4	0	0	2
QIEFP (Quad Input Extended Fast Program)	1	4	4	0	2	2
DCPP (Dual Command Fast Program)	2	2	2	1	Don’t Care	Don’t Care
QCPP (Quad Command Fast Program)	4	4	4	2	Don’t Care	Don’t Care

有几组器件（例如：Numonyx（Micro）N25Q512A）能够在双数据速率（DDR，也可叫做双传输速率模式（DTR））模式下处理读操作。在专有命令类型工作过程中，这些器件均可在上升沿和下降沿发送数据。这使得控制器可在低于2倍spi_clk频率的情况下保持流量。QSPI_DRIR[10]可告知QSPI Flash控制器，写入读OPCODE位段的OPCODE可处理DDR命令类型。读数据捕获寄存器（QSPI_RDCR）能在DDR模式下移动传输数据。默认地，数据移位1个时钟周期可确保在DDR传输中保持时间大于0。对于高ref_clk频率来说，位移1个时钟可能不够。DDR命令的配置与读表中列出的SDR命令兼容。Micron器件中的MT25系列还加入了DTR协议，可处理DTR模式中的所有命令。根据专门的OPCODE，DTR读命令可检测DTR模式。因此，OPCODE必须作为STR发送。当DTR协议使能，器件无需OPCODE来检测边沿模式，因为可从配置寄存器中（QSPI_CR）的易失性和非易失性位来识别。

32.5.9 APB 接口与寄存器模块

APB接口利用Flash命令控制寄存器（QSPI_FCR）来进行由软件控制的Flash访问。APB接口提供单个寄存器块，包含了可配置的寄存器组。该寄存器组由APB时钟定时。

32.5.10 SRAM 模块

间接操作模式需要存储器模块。存储器的引脚连接在顶层，使得积分器可选择使用SRAM存储器，寄存器阵列或FLOP组。存储器模块仅有一个端口。存储器模块的深度可选且由具体应用决定。

32.6 寄存器描述

QSPI 寄存器基地址：0x40E0_2000

寄存器列表如下：

表 32-3：QSPI Flash 控制器寄存器列表

偏移地址	寄存器名称	寄存器描述
0x00	QSPI_CR	QSPI配置寄存器
0x04	QSPI_DRIR	QSPI器件读指令寄存器
0x08	QSPI_DWIR	QSPI器件写指令寄存器
0x0C	QSPI_DDLR	QSPI器件延时寄存器
0x10	QSPI_RDCR	QSPI读数据捕获寄存器
0x14	QSPI_DSCR	QSPI器件容量配置寄存器
0x18	QSPI_SPR	QSPISRAM分块配置寄存器
0x1C	QSPI_IATR	QSPI间接AHB地址触发寄存器
0x20	QSPI_DMACR	QSPIDMA外设配置寄存器
0x24	QSPI_RAR	QSPI地址重映射寄存器
0x28	QSPI_MBR	QSPI模式位寄存器
0x2C	QSPI_SFLR	QSPISRAM数据深度状态寄存器
0x30	QSPI_TXHR	QSPI发送阈值寄存器
0x34	QSPI_RXHR	QSPI接收阈值寄存器
0x38	QSPI_WCR	QSPI写完成控制寄存器
0x3C	QSPI_PER	QSPI轮询结束寄存器
0x40	QSPI_IFR	QSPI中断标志寄存器
0x44	QSPI_IMR	QSPI中断屏蔽寄存器
0x48~0x4C	RSV	保留
0x50	QSPI_WPLR	QSPI写保护低位寄存器
0x54	QSPI_WPHR	QSPI写保护高位寄存器
0x58	QSPI_WPCR	QSPI写保护配置寄存器
0x5C	RSV	保留
0x60	QSPI_IRTR	QSPI间接读传输控制寄存器
0x64	QSPI_IRTWR	QSPI间接读传输数据深度阈值寄存器
0x68	QSPI_IRTSAR	QSPI间接读传输起始地址寄存器
0x6C	QSPI_IRTNR	QSPI间接读传输字节数寄存器

偏移地址	寄存器名称	寄存器描述
0x70	QSPI_IWTR	QSPI间接写传输控制寄存器
0x74	QSPI_IWTWR	QSPI间接写传输数据深度阈值寄存器
0x78	QSPI_IWTSAR	QSPI间接写传输起始地址寄存器
0x7C	QSPI_IWTNR	QSPI间接写传输字节数寄存器
0x80	QSPI_ITARR	QSPI间接触发地址范围寄存器
0x84~0x8C	RSV	保留
0x90	QSPI_FCR	QSPIFlash命令控制寄存器
0x94	QSPI_FCAR	QSPIFlash命令地址寄存器
0x98~0x9C	RSV	保留
0xA0	QSPI_FCRLR	QSPI Flash命令读数据低位寄存器
0xA4	QSPI_FCRHR	QSPI Flash命令读数据高位寄存器
0xA8	QSPI_FCWLR	QSPI Flash命令写数据低位寄存器
0xAC	QSPI_FCWHR	QSPI Flash命令写数据高位寄存器
0xB0	QSPI_PFSR	QSPI轮询Flash状态寄存器

32.6.1 QSPI 配置寄存器（QSPI_CR）

偏移地址：0x00

复位值：0x8078 0081

位	名称	属性	复位值	描述
31	IDLES	R	0x1	**串行接口和QSPIPIPELINE处于空闲：** 该位是一个重新计时的信号，所以生产该状态位信号时会有一些延时。 在设置IDLE位有效和切换配置域之间，建议等待至少4个ref_clk周期。该延时可确保底层的同步，FIFO为空，控制器可引入任何操作模式。
30:25	RSV	-	-	**保留**
24	DTRM	R/W	0x0	**DTR协议使能：** 如果器件使用DTR协议，则置位该位。 注意，DTR协议在DTR模式下提供所有命令。有些DTR读命令可在STR协议中处理（在DTR模式下）。将该位设置为0来执行这些命令。STR协议中的DTR命令由QSPI_DRIR.DDRM来控制。
23	AD_EN	R/W	0x0	**AHB解码器使能（仅限直接访问模式）：** 当设置为1时，PSL位为无关位。 有效从机取决于实际的AHB地址。 器件的配置基于QSPI_DSCR.CS_SIZE来计算。
22:19	BAUD	R/W	0xF	**主模式波特率分频：** SPI波特率=（主参考时钟）/BD 其中BD可配置为以下： 0000=/2 0001=/4 0010=/6 0011=/8 0100=/10 0101=/12 0110=/14 0111=/16

位	名称	属性	复位值	描述
				1000=/18 … 1111=/32 使能QSPI控制器前，优先设置好该寄存器。 注：QSPI通信时钟来源于clk_sys_pre。
18	XIPIM	R/W	0x0	**即刻进入XIP模式：** 0：如果XIP使能，当下一个读指令完成后，控制器将退出XIP模式。 1：即刻进入XIP模式，当外部器件从XIP模式唤醒时，使用该寄存器。控制器假定下一个读指令将作为XIP指令传递给器件，因此不会要求传输读操作码。 注意：为退出XIP模式，该位需设置为0。只有在下一个读指令执行后，才能在连接的器件中生效。因此，确定退出XIP模式前，在复位该位后，软件必须确保至少有一个读指令。该位由硬件同步。控制器运行过程中，可对该位进行设置或清零。
17	XIPNX	R/W	0x0	**在下一个读指令进入XIP模式：** 0：如果XIP使能，当下一个读指令完成后，控制器将退出XIP模式。 1：如果XIP禁止，该位置1可通知控制器，器件已就绪，可在下一个读指令进入XIP模式。接着，控制器将发送合适的命令序列，包括让器件进入XIP模式的模式位。当控制器已确保连接的Flash器件已准备就绪可进入XIP模式后，使用该寄存器。 注意：为退出XIP模式，该位需设置为0。只有在下一个读指令执行后，才能在连接的器件中生效。因此，确定退出XIP模式前，在复位该位后，软件必须确保至少有一个读指令。该位由硬件同步。控制器运行过程中，可对该位进行设置或清零。
16	AR_EN	R/W	0x0	**AHB地址重映射使能（仅限直接访问模式）：** 当该位置1，接收到的AHB地址将更新，并将（address+N）发送至Flash器件，其中N的值保存在地址重映射寄存器（QSPI_RAR）中。 该位由硬件同步。控制器运行过程中，可对该位进行设置或清零。
15	DMA_EN	R/W	0x0	**DMA外设接口使能：** 0：禁止 1：使能DMA握手逻辑。使能后，QSPI会通过DMA外设接口触发DMA传输请求。 该位由硬件同步。控制器运行过程中，可对该位进行设置或清零。
14	SWPP	R/W	0x0	**写保护引脚设置：** 设置该位来驱动Flash器件的写保护引脚，与生成的存储器时钟重新同步。 注意，WP引脚仅在SINGLE或DUAL传输模式下有效。在QUAD传输模式下，WP引脚用来传输数据，因此该寄存器位的设置将被无视。 该位由硬件同步。控制器运行过程中，可对该位进行设置或清零。
13:9	RSV	-	-	保留

位	名称	属性	复位值	描述
8	LIM_EN	R/W	0x0	**传统IP模式使能：** 0：使用直接访问控制器/间接访问控制器或STIG接口用作数据传输 1：使能传统模式。在传统模式下，任何AHB接口写操作都是串行发送至Flash器件。任何有效的AHB读会从内部RX-FIFO出栈，在SPI线上获取有外部Flash器件转发的数据，通过HSIZE输入控制和传输4，2或1个字节。 该位由硬件同步。控制器运行过程中，可对该位进行设置或清零。
7	DAC_EN	R/W	0x1	**直接访问控制器使能：** 0：一旦当前数据字（FF_W）传输完成，禁止直接访问控制器。 1：使能直接访问控制器 当直接访问控制器和间接访问控制器均禁止，所有AHB请求完成时都会产生错误反应。 该位由硬件同步。控制器运行过程中，可对该位进行设置或清零。
6:3	RSV	-	-	**保留**
2	CPHA	R/W	0x0	**时钟相位：** 时钟相位映射到标准SPI时钟相位传输格式。 注意：当运行于DDR模式或者DDR协议时，保持该位为低电平。
1	CPOL	R/W	0x0	**SPI字外的时钟极性：** 映射到标准SPI时钟极性传输格式。 注意：当运行于DDR模式或者DDR协议时，保持该位为低电平。
0	EN	R/W	0x1	**QSPI使能：** 0：一旦当前数据字（FF_W）传输完成，禁止QSPI。 1：使能QSPI 当spi_enable=0，所有的输出使能为无效，且所有引脚设置为输入模式。 该位由硬件同步。

32.6.2 QSPI 器件读指令寄存器（QSPI_DRIR）

偏移地址：0x04

复位值：0x0000 0003

位	名称	属性	复位值	描述
31:29	RSV	-	-	保留
28:24	DCYC	R/W	0x0	读指令所需的Dummy时钟周期数
23:21	RSV	-	-	保留
20	MB_EN	R/W	0x0	模式位使能： 置1确保模式位寄存器（QSPI_MBR）中定义的模式位跟在地址字节后发出。
19:18	RSV	-	-	保留

位	名称	属性	复位值	描述
17:16	DMODE	R/W	0x0	标准SPI模式下的数据传输类型（如IMODE位所定义）： 0：SIO模式 – 仅在DQ0上发送数据去器件，DQ1上接收器件上的数据。 1：用作DUAL输入/输出指令。数据传输时，DQ0和DQ1均用做为输入和输出。 2：用作QUAD输入/输出指令。数据传输时，DQ0，DQ1，DQ2和DQ3均用做为输入和输出。
15:14	RSV	-	-	保留
13:12	AD_MODE	R/W	0x0	标准SPI模式下的地址传输类型（如IMODE位所定义）： 0：地址仅可在DQ0上发送去器件。 1：地址仅可在DQ0和DQ1上发送去器件。 2：地址可在DQ0，DQ1，DQ2和DQ3上发送去器件。
11	RSV	-	-	保留
10	DDRM	R/W	0x0	DDR位使能 当QSPI_WCR.OPCODE符合DDR命令，该位置1。
9:8	IMODE	R/W	0x0	指令类型： 0：标准SPI模式（始终仅在DQ0上发送指令去器件） 1：DIO-SPI模式（始终在DQ0和DQ1上发送指令，地址和数据） 2：QIO-SPI模式（始终在DQ0，DQ1，DQ2和DQ3上发送指令，地址和数据） 注意：该位不仅仅只与读传输相关。该位为全局设置，将影响DAC和INDAC读，写和任何STIG传输。
7:0	RINST	R/W	0x3	在非XIP模式下，读操作码

32.6.3 QSPI 器件写指令寄存器（QSPI_DWIR）

偏移地址：0x08

复位值：0x0000 0002

位	名称	属性	复位值	描述
31:29	RSV	-	-	保留
28:24	DCYC	R/W	0x0	写指令所需的Dummy时钟周期数
23:18	RSV	-	-	保留
17:16	DMODE	R/W	0x0	标准SPI模式下的数据传输类型： 0：SIO模式 – 仅在DQ0上发送数据去器件，DQ1上接收器件上的数据。 1：用作DUAL输入/输出指令。数据传输时，DQ0和DQ1均用做为输入和输出。 2：用作QUAD输入/输出指令。数据传输时，DQ0，DQ1，DQ2和DQ3均用做为输入和输出。
15:14	RSV	-	-	保留
13:12	AD_MODE	R/W	0x0	标准SPI模式下的地址传输类型： 0：地址仅可在DQ0上发送去器件。 1：地址仅可在DQ0和DQ1上发送去器件。 2：地址可在DQ0，DQ1，DQ2和DQ3上发送去器件。

位	名称	属性	复位值	描述
11:9	RSV	-	-	保留
8	WELD	R/W	0x0	WEL禁止 在写操作前，该位允许关闭自动发送WEL命令
7:0	WINST	R/W	0x2	写操作码

32.6.4 QSPI 器件延时寄存器（QSPI_DDLR）

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:24	CSDA	R/W	0x0	片选设为无效 在两个传输之间，主模式片选输出设为无效的时间（主参考时钟ref_clk上的延时） 片选设为无效的最小延时： 1sclk_out+1ref_clk 为确保片选不会在1个sclk_out周期内再次变为有效。 如果CSDA=X，则片选无效时间为： 1sclk_out+1ref_clk+Xref_clks
23:16	RSV	-	-	保留
15:8	CSEOT	R/W	0x0	片选传输结束 当前传输的最后一位与将片选（n_ss_out）设为无效之间的延时（主参考时钟ref_clk上的延时） 默认，当CSEOT=0，在当前传输结束时，片选将在sclk_out的最后一个下降沿上设为无效。 如果CSEOT=X，片选会在sclk_out的最后一个下降沿之后的X个ref_clks后变为无效。
7:0	CSSOT	R/W	0x0	片选传输起始 设置n_ss_out为低电平和传输第一个比特位之间的延时。 默认，当CSSOT=0，在sclk_out的第一个上升沿之前的半个SCLK周期，片选将被设为有效。 如果CSSOT=X，在sclk_out+Xref_clks的第一个上升沿之前的半个sclk_out周期，片选将被设为有效。

32.6.5 QSPI 读数据捕获寄存器（QSPI_RDCR）

偏移地址：0x10

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:20	RSV	-	-	保留
19:16	DLYT	R/W	0x0	传输数据延时（延时时间为设定的ref_clk周期数） （目的为在DDR模式中提高传输过程中的保持时间） 当DDR读命令执行时，该位才有效；否则，无视。 delay_value=1xref_clk+<field_value>*ref_clk

位	名称	属性	复位值	描述
15:6	RSV	-	-	保留
5	SMES	R/W	0x0	采样边沿选择（Flash存储器数据输出） 0：在ref_clk的下降沿对Flash存储器的数据输出进行采样 1：在ref_clk的上升沿沿对Flash存储器的数据输出进行采样
4:1	DLYR	R/W	0x0	读数据捕获延时（延时时间为设定的ref_clk周期数）
0	RSV	-	-	保留，软件写1。

32.6.6 QSPI 器件容量配置寄存器（QSPI_DSCR）

偏移地址：0x14

复位值：0x0010 1002

位	名称	属性	复位值	描述
31:23	RSV	-	-	保留
22:21	CS_SIZE	R/W	0x0	连接CS引脚的Flash器件大小： 当AHB解码器使能时，该位有效。 00：512MB 01：1GB 10：2GB 11：4GB 注：不支持AHB解码器，此位无用。
20:16	BK_SIZE	R/W	0x10	每块的字节数： 控制器需要该位来执行写保护逻辑。每块的字节数必须为2的N次方： 0：1byte 1：2bytes 2：4bytes 3：8bytes 16：65535bytes
15:4	PA_SIZE	R/W	0x100	每个器件页的字节数： 控制器需要该位来执行跨页的Flash写操作。
3:0	AD_SIZE	R/W	0x2	地址字节数： 0：1 byte …

32.6.7 QSPI SRAM 分块配置寄存器（QSPI_SPR）

偏移地址：0x18

复位值：0x0000 0080

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留

7:0	SRAMPS	R/W	0x80	定义了SRAM中间接读区域的大小。 默认地，SRAM中一半的大小预留给间接读操作，另一半分配给间接写操作。该寄存器大小可随着SRAM深度改变。 分配给间接读的地址数量 = SRAMPS+1 分配给间接写的地址数量 = (2**8)-SRAMPS 不要将SRAMPS设置为0x00或(2**8)-1

32.6.8 QSPI 间接 AHB 地址触发寄存器（QSPI_IATR）

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	IND_TAD	R/W	0x0	间接触发地址： 该地址为基地址。当接收到的AHB读访问地址与（从触发地址到触发地址+<配置范围>）内的地址匹配，则通过从间接控制器SRAM中获取数据来完成AHB请求。

32.6.9 QSPI DMA 外设配置寄存器（QSPI_DMACR）

偏移地址：0x20

复位值：0x0000 0000

位数	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:8	BNUMB	R/W	0x0	DMA外设请求中Burst类型的字节数： 设置为0时，代表一个字节。该位需在间接读/写开始前设置完成。实际的字节数为2**（寄存器设置值） 突发传输的条件为： 1）SRAM中空闲的存储空间（字为单位）大于等于2**[11:8]（以字节为单位）的设置值； 2）间接访问总的数据量（间接读传输控制寄存器（QSPI_IRTR）/间接写传输控制寄存器（QSPI_IWTR））大于等于2**[11:8]的设置值。 如果不满足上述条件，系统会自动采用单次传输。例如：（SRAM大小为1KB）SRAM分块配置寄存器（QSPI_SPR）配置为0x80，第一次启动数据间接写传输时，SRAM中空闲的存储空间为0x80字大小，如果此时设置[11:8]的值大于等于8时，控制器会自动采用single方式传输。
7:4	RSV	-	-	保留

位数	名称	属性	复位值	描述
3:0	SNUMB	R/W	0x0	DMA外设请求中Single类型的字节数： 设置为0时，代表一个字节。该位需在间接读/写开始前设置完成。实际的字节数为2**（寄存器设置值），2**（寄存器设置值）表示一次single传输数据量的大小，如一次single传输一个字，则需要将此控制位设置为2，如果设置成0则一次single传输传输的数据量为1个字节。 注意应用时大小需要设置成Word的整数倍。

32.6.10 QSPI 地址重映射寄存器（QSPI_RAR）

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	READDR	R/W	0x0	该寄存器将接收到的AHB地址重映射到一个不同的地址。

32.6.11 QSPI 模式位寄存器（QSPI_MBR）

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	MODEB	R/W	0x0	模式位 在地址字节后，向外部器件发送这8位模式位

32.6.12 QSPI SRAM 数据深度状态寄存器（QSPI_SFLR）

偏移地址：0x2C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	IND_WSFL	R	0x0	SRAM深度状态（间接写区域），单位为字（4个字节）
15:0	IND_RSFL	R	0x0	SRAM深度状态（间接读区域），单位为字（4个字节）

32.6.13 QSPI 发送阈值寄存器（QSPI_TXHR）

偏移地址：0x30

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4:0	TXTH	R/W	0x1	小容量TX FIFO不满中断阈值设置值 只有在传统模式下访问Flash器件，该位才有效。其他情况下可无视。

32.6.14 QSPI 接收阈值寄存器（QSPI_RXHR）

偏移地址：0x34

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:5	RSV	-	-	保留
4:0	RXTH	R/W	0x1	小容量RX FIFO不空中断阈值设置值 只有在传统模式下访问Flash器件，该位才有效。其他情况下可无视。

32.6.15 QSPI 写完成控制寄存器（QSPI_WCR）

偏移地址：0x38

复位值：0x0001 0005

位	名称	属性	复位值	描述
31:24	PREPD	R/W	0x0	轮询重复延时： 在自动轮询阶段，片选保持无效的额外延时时间
23:16	PCNT	R/W	0x1	轮询次数： 定义轮询次数。当控制器吞吐量高时，有必要添加额外的周期。
15	POLL_EXP_EN	R/W	0x0	使能轮询结束中断： 0：禁止 1：当QSPI_PER中设定的周期数结束后，产生中断
14	PDIS	R/W	0x0	轮询禁止： 关闭自动轮询功能
13	PPLT	R/W	0x0	轮询极性： 0：如果轮询位为0，则写传输完成。 1：如果轮询位为1，则写传输完成。
12:11	RSV	-	-	保留
10:8	PBIND	R/W	0x0	轮询位检索： 设置为010时，表示轮询QSPI_PFSR寄存器的bit2。 设置为111时，表示轮询QSPI_PFSR寄存器的bit7。
7:0	OPCODE	R/W	0x5	操作码： 当自动轮询器件编程的完成状态时，控制器必须发送操作码。该命令在所有的器件写操作后发送。 默认使用操作码0x05来轮询标准器件状态寄存器。

32.6.16 QSPI 轮询结束寄存器（QSPI_PER）

偏移地址：0x3C

复位值：0xFFFF FFFF

位	名称	属性	复位值	描述
31:0	PCYCN	R/W	0xFFFFFFFF	轮询周期数 轮询周期数后，产生轮询结束中断。

32.6.17 QSPI 中断标志寄存器（QSPI_IFR）

偏移地址：0x40

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:14	RSV	-	-	保留
13	POLLF	R/W1C	0x0	最大的轮询周期数
12	IND_RSFF	R/W1C	0x0	SRAM中的间接读区域满，无法立即完成间接操作。
11	SRFFF	R/W1C	0x0	小容量RXFIFO满（当前FIFO状态）： 在非SPI传统模式下，可无视。 0：FIFO非满 1：FIFO满
10	SRFNEF	R/W1C	0x0	小容量RXFIFO非空（当前FIFO状态）： 在非SPI传统模式下，可无视。 0：FIFO<RX阈值 1：FIFO≥阈值
9	STFFF	R/W1C	0x0	小容量TXFIFO满（当前FIFO状态）： 在非SPI传统模式下，可无视。 0：FIFO非满 1：FIFO满
8	STFNFF	R/W1C	0x0	小容量TXFIFO非满（当前FIFO状态）： 在非SPI传统模式下，可无视。 0：FIFO>阈值 1：FIFO≤阈值
7	ROVF	R/W1C	0x0	接收上溢（仅在传统模式下发生）： 0：未检测到上溢 1：发生上溢
6	IND_TWF	R/W1C	0x0	超出间接传输深度阈值： 0：未超出阈值 1：超出阈值
5	AHB_AEF	R/W1C	0x0	检测到非法AHB访问： AHB写WrappingBurst和使用SPLIT/RETRY访问将触发该中断。 如果DAC禁止状态下进行AHB访问，也将触发该中断。
4	WPAF	R/W1C	0x0	企图写保护区域被拒绝
3	IND_RRF	R/W1C	0x0	不接收间接操作请求 已存在两个间接操作。

位	名称	属性	复位值	描述
2	IND_CF	R/W1C	0x0	控制器已完成上一个间接操作
1	UDFF	R/W1C	0x0	下溢检测： 0：未检测到下溢 1：检测到下溢：当小容量TXFIFO为空时，企图发送数据。当AHB写数据太慢，跟不上写请求时，有可能发生下溢。
0	RSV	-	-	保留

32.6.18 QSPI 中断屏蔽寄存器（QSPI_IMR）

偏移地址：0x44

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:14	RSV	-	-	保留
13:1	INT_EN	R/W	0x0	0：中断标志寄存器（QSPI_IFR）中的相关中断位禁止 1：中断标志寄存器（QSPI_IFR）中的相关中断位使能
0	RSV	-	-	保留

32.6.19 QSPI 写保护低位寄存器（QSPI_WPLR）

偏移地址：0x50

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	LBLK_NUM	R/W	0x0	定义了写保护块范围的起始块，通过器件容量配置寄存器（QSPI_DSCR）可配置块的字节数。 仅当写保护功能（QSPI_WPCR.WPEN）为低电平，可更改该位。

32.6.20 QSPI 写保护高位寄存器（QSPI_WPHR）

偏移地址：0x54

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	HBLK_NUM	R/W	0x0	定义了写保护块范围的结束块，通过器件容量配置寄存器（QSPI_DSCR）可配置块的字节数。 仅当写保护功能（QSPI_WPCR.WPEN）为低电平，可更改该位。

32.6.21 QSPI 写保护配置寄存器（QSPI_WPCR）

偏移地址：0x58

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	WP_EN	R/W	0x0	写保护使能： 1：拒绝访问任何AHB写访问（其写访问地址在写保护低位寄存器（QSPI_WPLR）和写保护高位寄存器（QSPI_WPHR）范围内）。产生一个AHB错误响应，触发中断源。 0：写保护禁止。该位由硬件内部同步，当改变该位状态时，无特别软件要求。
0	WPINV	R/W	0x0	写保护取反控制： 1：写保护取反：系统允许写由写保护低位寄存器（QSPI_WPLR）和写保护高位寄存器（QSPI_WPHR）定义的保护区域的地址。 0：写保护不取反：系统不允许写由写保护低位寄存器（QSPI_WPLR）和写保护高位寄存器（QSPI_WPHR）定义的保护区域的地址。仅当写保护功能（QSPI_WPCR.WPEN）为低电平，可更改该位。

32.6.22 QSPI 间接读传输控制寄存器（QSPI_IRTR）

偏移地址：0x60

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:6	IND_RNUM	R	0x0	间接操作的完成数 该位与Rdcs位联合使用。 当完成一个间接操作时，由硬件递增。 当Rdcs位写入1时，则递减。
5	RDCS	R/W1	0x0	间接读完成状态 当完成一个间接操作，该位由硬件置1，写1清零。
4	RDQS	R	0x0	已有两个间接读操作排队等候 该位由硬件置1和清零。
3	SRAMFS	R/W1	0x0	SRAM满，无法立即完成间接操作 硬件置1，写1清零。
2	RDPS	R	0x0	间接读操作进行中 仅由硬件清零。
1	RDDIS	W1	0x0	取消间接读 写1取消进行中的间接读操作。同时取消当前所有间接操作。 该位由硬件内部同步。

位	名称	属性	复位值	描述
0	RDST	W1	0x0	开始间接读 写1触发间接读操作。触发间接读操作前，先设置好间接读传输起始地址寄存器（QSPI_IRTSAR）和间接读传输字节数寄存器（QSPI_IRTNR）。 该位由硬件内部同步。

32.6.23 QSPI 间接读传输数据深度阈值寄存器（QSPI_IRTWR）

偏移地址：0x64

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	VALUE	R/W	0x0	数据深度 在DMA外设访问前，该值代表的是SRAM的最小数据深度。当SRAM数据深度超越该值，则触发中断源。 写全0可禁止。 单位为字节。

32.6.24 QSPI 间接读传输起始地址寄存器（QSPI_IRTSAR）

偏移地址：0x68

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	ADDR	R/W	0x0	间接访问起始地址

32.6.25 QSPI 间接读传输字节数寄存器（QSPI_IRTNR）

偏移地址：0x6C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	NUM	R/W	0x0	间接字节数，该设置值可大于SRAM大小。

32.6.26 QSPI 间接写传输控制寄存器（QSPI_IWTR）

偏移地址：0x70

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留

位	名称	属性	复位值	描述
7:6	IND_WNUM	R	0x0	间接操作的完成数 该位与Wrcs位联合使用。 当完成一个间接操作时，由硬件递增。 当Wrcs位写入1时，则递减。
5	WRCS	R/W1	0x0	间接写完成状态 当完成一个间接操作时，该位由硬件置1，写1清零。
4	WRQS	R	0x0	已有两个间接写操作排队等候 该位由硬件置1和清零。
3	RSV	-	-	保留
2	WRPS	R	0x0	间接写操作进行中 仅由硬件清零。
1	WR_DIS	W1	0x0	取消间接写 写1取消进行中的间接写操作。同时取消当前所有间接操作。 该位由硬件内部同步。
0	WR_ST	W1	0x0	开始间接写 写1触发间接写操作。触发间接写操作前，先设置好间接写传输起始地址寄存器（QSPI_IWTSAR）和间接写传输字节数寄存器（QSPI_IWTNR）。 该位由硬件内部同步。

32.6.27 QSPI 间接写传输数据深度阈值寄存器（QSPI_IWTWR）

偏移地址：0x74

复位值：0xFFFF FFFF

位	名称	属性	复位值	描述
31:0	VALUE	R/W	0xFFFFFFFF	数据深度 在DMA外设访问前，该值代表的是SRAM的最大数据深度。当SRAM数据深度低于该值，则触发中断源。写全1可禁止。 单位为字节。

32.6.28 QSPI 间接写传输起始地址寄存器（QSPI_IWTSAR）

偏移地址：0x78

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	ADDR	R/W	0x0	间接访问起始地址

32.6.29 QSPI 间接写传输字节数寄存器（QSPI_IWTNR）

偏移地址：0x7C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	NUM	R/W	0x0	间接字节数，该设置值可大于SRAM大小。

32.6.30 QSPI 间接触发地址范围寄存器（QSPI_ITARR）

偏移地址：0x80

复位值：0x0000 0004

位	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3:0	RNGW	R/W	0x4	间接范围宽度 该位为间接AHB地址触发寄存器（QSPI_IATR）的偏移地址。当触发间接访问且AHB地址在此范围内，访问请求就会发送至间接写/读控制器。 该值为2的N次方。默认2^4=16，允许进行16字节的Burst。该位段反映了范围宽度，有效地址的个数（单个地址32位）从间接触发地址到间接触发地址+<间接范围宽度-1>（默认值：间接触发地址~间接触发地址+15）

32.6.31 QSPI Flash 命令控制寄存器（QSPI_FCR）

偏移地址：0x90

复位值：0x0000 0000

位数	名称	属性	复位值	描述
31:24	OPCODE	R/W	0x0	命令操作码 命令操作码需在触发命令前设置。举例：0x20映射到SubSector擦除，写CMDT位可发送命令。 注意，利用该方式发送命令，将用到QSPI_DRIR.IMODE表示的指令类型。如果该位设置为2'b00，则会连续传输命令操作码，命令地址，命令dummy字节，命令数据。如果该位设置为2'b01，则会通过DQ0和DQ1引脚并行传输命令操作码，命令地址，命令dummy字节和命令数据。如果该位设置为2'b10，则会通过DQ0，DQ1，DQ2，DQ3引脚并行传输命令操作码，命令地址，命令dummy字节和命令数据。

位数	名称	属性	复位值	描述
23	RD_EN	R/W	0x0	读数据使能 如果OPCODE位中的命令要求接收读数据字节，将该位设置为1。
22:20	RD_NUM	R/W	0x0	读数据字节数： 高达8个数据字节可读取 0：1字节 1：2字节 … 7：8字节
19	ADDR_EN	R/W	0x0	命令地址使能 如果OPCODE位中的命令需要地址，将该位设置为1。通过CMDT位触发命令前，先设置好该位。
18	MODB_EN	R/W	0x0	模式位使能 该位置1确保模式位寄存器（QSPI_MBR）中的模式位在地址字节后发送。
17:16	AD_NUM	R/W	0x0	地址字节数 设置所需的地址字节数（地址本身在Flash命令地址寄存器（QSPI_FCAR）中设置）通过CMDT位触发命令前，先设置好该位 00：1 01：2地址字节 10：3地址字节 11：4地址字节
15	WR_EN	R/W	0x0	写数据使能 如果OPCODE位中的命令要求发送写数据字节，将该位设置为1。
14:12	WD_NUM	R/W	0x0	写数据字节数： 高达8个数据字节可写 0：1字节 1：2字节 … 7：8字节
11:7	DUM_NUM	R/W	0x0	Dummy时钟周期数 设置OPCODE（bit31:24）中指定的命令所需的dummy周期数。 通过CMDT位触发命令前，先设置好该位。
6:2	RSV	-	-	保留
1	CMDS	R	0x0	命令执行进行中
0	CMDT	R/W1C	0x0	执行命令 该位内部同步。

32.6.32 QSPI Flash 命令地址寄存器（QSPI_FCAR）

偏移地址：0x94

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CMD_ADR	R/W	0x0	命令地址 在触发QSPI_FCR.CMDT前，先设置好该位。该地址为QSPI_FCR.OPCODE命令的地址。

32.6.33 QSPI Flash 命令读数据低位寄存器（QSPI_FCRLR）

偏移地址：0xA0

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CMD_DL	R	0x0	命令读数据（低字节） 通过触发Flash命令控制寄存器（QSPI_FCR）中的事件，由Flash器件返回的任何状态或配置读操作的数据。当QSPI_FCR.Cmds为低电平，该寄存器有效。

32.6.34 QSPI Flash 命令读数据高位寄存器（QSPI_FCRHR）

偏移地址：0xA4

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CMD_DH	R	0x0	命令读数据（高字节） 通过触发Flash命令控制寄存器（QSPI_FCR）中的事件，由Flash器件返回的任何状态或配置读操作的数据。当QSPI_FCR.Cmds为低电平，该寄存器有效。

32.6.35 QSPI Flash 命令写数据低位寄存器（QSPI_FCWLR）

偏移地址：0xA8

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CMD_DL	R/W	0x0	命令写数据（低字节） 在触发QSPI_FCR.CMDT前，先设置好该位。通过触发Flash命令控制寄存器（QSPI_FCR）中的事件，向Flash器件写入的任何状态或配置写操作的数据。

32.6.36 QSPI Flash 命令写数据高位寄存器（QSPI_FCWHR）

偏移地址：0xAC

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CMD_DH	R/W	0x0	命令写数据（高字节） 在触发QSPI_FCR.CMDT前，先设置好该位。通过触发Flash命令控制寄存器（QSPI_FCR）中的事件，由Flash器件写入的任何状态或配置写操作的数据

32.6.37 QSPI 轮询 Flash 状态寄存器（QSPI_PFSR）

偏移地址：0xB0

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	PSV	R	0x0	轮询状态有效 当FLSS位有效，该位置1。
7:0	FLSS	R	0x0	Flash状态 定义了器件的实际状态寄存器。

32.7 使用流程

在与Flash器件通信前，软件负责配置QSPI控制器。在QSPI控制器通过QSPI_CR[0]使能前，控制器静态配置位需设置完成。该目的是：在未实施亚稳定保护的路径上避免时钟边界问题。如果用户希望更改控制器配置，在重新配置前，建议将QSPI的使能位设置为禁止。

32.7.1 复位后配置 QSPI 控制器

QSPI控制器可在合适的状态唤醒后，使用直接访问控制器进行基本读写。所有目标器件均支持基本读（opcode 0x03）和基本写（opcode 0x02）操作。控制器唤醒时，波特率分频为32。如果目标器件不使用3个地址字节，器件容量配置寄存器（QSPI_DSCR）必须更改为一个合适的值。所有在该项目中测试的器件，其页容量为256字节，无需再更新寄存器。若非硬性要求，在使能QSPI控制前，软件应慎重使能写保护功能。需设置写保护低位寄存器（QSPI_WPLR），写保护高位寄存器（QSPI_WPHR）和写保护配置寄存器（QSPI_WPCR），器件容量配置寄存器（QSPI_DSCR）中的每个器件块的字节数也需设置。初始化后，当控制器和DAC使能后，软件可读取和写入Flash器件。简单对配置寄存器（QSPI_CR）写入就可使能控制器。注意，不要更改该寄存器中波特率分频和CPOL/CPHA的默认值。建议写入值为0x00780081。

32.7.2 QSPI 控制器配置优化

为更优化地访问Flash，软件需准确配置控制器。

1. 等待直到挂起的STIG或INDAC操作都已完成或轮询QSPI_CR[31]。
2. 禁止DAC（QSPI_CR[7]）。不必要完全禁止QSPI控制器（QSPI_CR[0]）。
3. 根据用户希望使用的指令类型来更新器件指令配置寄存器（QSPI_DRIR/QSPI_DWIR） 。
4. 如果器件指令配置寄存器中的模式位（QSPI_DRIR[20]）使能，更新模式位寄存器（QSPI_MBR） 。
5. 如果内容不正确，更新器件容量配置寄存器（QSPI_DSCR）。注意，初始化后，部分或整个寄存器都有可能已更新。进行读写时，地址字节数是一个关键配置。进行任何写操作时，需要设置每页的字节数。只有在使用写保护功能的情况下，才需要每个器件块的字节数。如果目标器件的默认值正确，或者如果某些值（不包括地址字节数）不正确，并不允许器件写操作。
6. 更新QSPI器件延时寄存器（QSPI_DDLR）。该寄存器允许用户在每次Flash访问后，对如何驱动片选进行调整。原因是每个器件可能有不同的时序要求。当串行时钟频率增加，时序要求将变得越来越重要。注意，该寄存器中设置的值基于ref_clk周期。举例：在CS无效后重新设置为有效前，ATMEL器件需要至少50ns的时间。默认情况下，控制器仅会提供最小1 SCLK周期。当器件运行于100MHz，SCLK周期仅为10ns，因此需要额外的40ns。当ref_clk运行于400MHz（2.5ns周期），该寄存器中的CSDA位至少要设置为16。在自动轮询阶段，可增加该延时。写完成控制寄存器（QSPI_WCR）中可定义轮询重复延时。
7. 如有需要，更新地址重映射寄存器（QSPI_RAR）。仅影响DAC路径。
8. 如有需要，如果在过往的初始化中还未设置，则设置和使能写保护低位寄存器（QSPI_WPLR），写保护高位寄存器（QSPI_WPHR）和写保护配置寄存器（QSPI_WPCR）。
9. 通过中断屏蔽寄存器（QSPI_IMR）使能需要的中断。
10. 在配置寄存器（QSPI_CR[22:19]）中设置波特率分频，定义目标器件所需的时钟频率。
11. 更新读数据捕获寄存器（QSPI_RDCR）。当捕获读数据时，该寄存器可延时。当由器件到控制器的读数据路径很长且器件时钟频率很高，该寄存器的设置也会有所帮助。
12. 通过配置寄存器（QSPI_CR[0]）使能QSPI控制器和DAC。

32.7.3 Flash 命令控制寄存器的使用（STIG 操作）

Flash命令控制寄存器（QSPI_FCR）提供了一种灵活可编程的方式访问Flash器件，被称为STIG操作。指令OPCODE，地址字节数，地址本身，dummy字节数，写数据字节数，写数据本身和读数据字节数，均可设置。一旦设置完成，软件可通过QSPI_FCR[0]触发命令，轮询QSPI_FCR[1]等待被接受。当该位变为无效，就可触发另一个STIG。软件可用此种访问Flash的典型方法来访问Flash器件寄存器，也可进行擦除操作。尽管最多8个数据字节（由Flash命令读和写数据寄存器

（QSPI_FCRLR/QSPI_FCRHR/QSPI_FCWLR/QSPI_FCWHR）定义）可一次读或写，该方法同样可用来访问Flash阵列本身。对比AHB的所有其他读访问，该接口发送的命令拥有更高优先级。

32.7.4 SPI 传统模式

SPI传统模式允许软件直接访问内部TX-FIFO和RX-FIFO，因此可绕过直接，间接和STIG控制器。传统模式允许用户发送任意Flash指令，但会花费太多的软件开销，确保了FIFO fill level的有效管理。传统SPI内核是双向传输，当片选使能，数据可在任一方向连续传输。即使驱动器只希望从Flash器件读取数据，dummy数据必须写出确保片选保持有效。对于写传输，反之亦然。这意味着，向一个器件（3个地址字节）读4个字节，软件需向TX FIFO中写入一共8个字节。第一个字节为指令OPCODE，接下来的3个字节为地址，最后4个字节为dummy数据。类似的，因TX FIFO中写入了8个字节，软件会期望有8个字节返回至RX FIFO中。前4个字节会被丢弃，后4个字节为所需的读数据。 由于TX FIFO和RX FIFO深度有限，软件有责任保持FIFO的深度，确保在指令执行过程中TX FIFO不会下溢，RX FIFO不会上溢。这可能会增加软件开销。当数据深度超过深度阈值时，将会产生中断。当传统模式使能，软件可访问TX FIFO，通过AHB接口向QSPI控制器的任意地址内写入任意值。当传统模式使能，软件可访问RX FIFO，软件可通过AHB接口向QSPI控制器读取任意地址。

32.7.5 进入和退出 XIP 模式

32.7.5.1 从 POR 进入 XIP 模式

如果XIP以非易失性配置的使能，则XIP模式可以非易失性方式进入。Flash器件中仅一小部分支持该功能。由于Flash器件唯一能识别的操作为XIP读操作，软件无法通过轮询Flash状态寄存器（QSPI_PFSR）读发现POR后XIP的状态。 如果已得知器件会从POR进入XIP，QSPI_MBR寄存器和QSPI_CR[18]应在初始BOOT中设置完成。如果不知道器件会从POR进入XIP，且连接的Flash器件支持从POR进入XIP，则软件通过STIG命令发送XIP退出命令来尝试退出XIP模式。软件需了解器件的模式位要求，因为每个器件的XIP进入和退出都不同。Micron N25Q和MT25支持XIP从POR进入/退出。退出XIP模式时，需设置OPCODE为8’h00，地址字节数为3，dummy周期数为16，读字节数为1。地址需配置为{8模式位，16’h0000}。Micron器件模式位需设置为8’b10000000。注意，通过Flash命令控制寄存器（QSPI_FCR）发送NVCR命令可使能从POR进入XIP。直到下一个POR序列时，才会生效。

32.7.5.2 其他进入 XIP 模式的情况

大部分Flash器件支持XIP模式。但Flash生产厂商并没有一致的标准。大部分是在地址字节后直接发送签名位。其他，像Micron器件，采用signature bits的同时，也要求写Flash器件配置寄存器来

使能XIP。对于必须服从该控制器的Flash器件，需遵循以下步骤进入XIP模式。Micron N25Q和MT25（不支持基本XIP）通过设置Flash器件内的VCR[3]来使能XIP模式。利用Flash命令控制寄存器（QSPI_FCR）写VCR来发送VCR写命令，步骤如下：

1. 禁止直接访问控制器和间接访问控制器，确保没有新的AHB读访问发送到Flash器件。
2. 配置Flash命令控制寄存器（QSPI_FCR），发送VCR写至Flash存储器。
3. 设置模式位寄存器（QSPI_MBR[7:0]）中的XIP模式位为8’b0000000。
4. 通过设置QSPI_CR[17]来使能自有控制器的XIP模式。
5. 如有需要，重新使能直接访问控制器，间接访问控制器。

其他Micron器件（支持基本XIP）

1. 禁止直接访问控制器和间接访问控制器，确保没有新的AHB读访问发送到Flash器件。
2. 设置模式位寄存器（QSPI_MBR[7:0]）中的XIP模式位为8’b1000000。
3. 通过设置QSPI_CR[17]来使能自有控制器的XIP模式。
4. 如有需要，重新使能直接访问控制器，间接访问控制器。

Winbond器件

1. 禁止直接访问控制器和间接访问控制器，确保没有新的AHB读访问发送到Flash器件。
2. 设置模式位寄存器（QSPI_MBR[7:0]）中的XIP模式位为8’b0010000。
3. 通过设置QSPI_CR[17]来使能自有控制器的XIP模式。
4. 如有需要，重新使能直接访问控制器，间接访问控制器。

Spansion器件

1. 禁止直接访问控制器和间接访问控制器，确保没有新的AHB读访问发送到Flash器件。
2. 设置模式位寄存器（QSPI_MBR[7:0]）中的XIP模式位为8’b1010000。
3. 通过设置QSPI_CR[17]来使能自有控制器的XIP模式。
4. 如有需要，重新使能直接访问控制器，间接访问控制器。

32.7.5.3 退出 XIP 模式

为退出XIP模式，软件需禁止直接访问控制器和间接访问控制器，确保没有新的AHB读访问发送到Flash器件。然后将模式位设置为其他任何值，而不是相应Flash器件规格上的特定模式位。接着软件需复位QSPI_CR[17]。 注意，在禁止内部XIP模式状态前，Flash器件必须见到读指令，意味着，XIP模式将一直保持有效直到下一个读指令被响应。需确保XIP模式在读序列结束前禁止。

32.7.6 间接数据传输模式

1. 初始化 QSPI 相关 GPIO 引脚。
2. 初始化 QSPI 时钟和释放复位，等待 QSPI_CR[31]置位。
3. 设置 QSPI_DDLR 寄存器，配置通信时序延时。
4. 设置 QSPI_DSCR 寄存器，设置设备地址，设备页大小和设备块大小。
5. 设置 QSPI_CFGR 寄存器，设置 QSPI 工作模式，主频率，使能 INDAC 模式。
6. 设置 QSPI_RDCR 寄存器，设置数据捕获延时。
7. 如果以上设置命令需要设备进入 QE 模式则往设备填写相关进入指令，等待空闲，没有则执行步骤 8。
8. 要进行间接写操作请配置步骤 9-步骤 15，要进行间接读操作请配置步骤 16-步骤 22。
9. 设置 QSPI_IWTSAR，配置间接写操作起始地址
10. 设置 QSPI_IWTNR，配置间接写操作字节数。
11. 设置 QSPI_IATR，配置间接写操作外部 flash 地址。
12. 设置 QSPI_ITARR，配置间接写操作传输地址范围。
13. 设置 QSPI_IWTR[5]为 1，间接写操作完成。
14. 直接往外部 flash 地址写 32bit 的值。
15. 等待 QSPI_IWTR[5]为 1，间接写操作完成，随后写 1 清零。
16. 设置 QSPI_IRTSAR，配置间接读操作起始地址。
17. 设置 QSPI_IRTNR，配置间接读操作字节数。
18. 设置 QSPI_IATR，配置间接写操作外部 flash 地址。
19. 设置 QSPI_ITARR，配置间接写操作传输地址范围。
20. 设置 QSPI_IRTR[0]为 1，间接写操作开始。
21. 直接往外部 flash 地址读 32bit 的值。
22. 等待 QSPI_IRTR[5]为 1，间接读操作完成。

32.7.7 AHB 地址重映射

当访问经过直接访问控制器时，QSPI的重映射功能定义了如何转换接收到的AHB地址。默认为1：1映射。当QSPI_CR[16]使能，接收到的AHB地址重映射至address + N，其中N的值保存在地址重映射寄存器（QSPI_RAR）中。建议在配置地址重映射寄存器（QSPI_RAR）前，先禁止QSPI。

32.7.8 中断响应

QSPI提供一个高有效的中断引脚。关于所有中断源的信息，请参考中断标志寄存器（QSPI_IFR）。

当软件读中断标志寄存器（QSPI_IFR）时，中断源清零。QSPI控制器也可配置成，写中断标志寄存器（QSPI_IFR）时，中断清零。默认的清零方式为：写清零（write-to-clear）。中断屏蔽寄存器（QSPI_IMR）可屏蔽所有中断。写0使能，写1清零。

32.7.9 AHB 保护寄存器

发生POR时，AHB保护机制处于禁止状态。软件可利用保护寄存器来阻止AHB写入特定的寄存器。在配置保护寄存器前，建议禁止QSPI。

32.7.10 DAC 模式配置流程示例

1. 初始化 QSPI 相关 GPIO 引脚。
2. 初始化 QSPI 时钟和释放复位，等待 QSPI_CR[31]置位。
3. 设置 QSPI_DRIR 寄存器，配置 DAC 模式下命令的 dummy 时钟，数据传输地址宽度，地址传输宽度，读命令。
4. 设置 QSPI_DWIR 寄存器，配置 DAC 模式下命令的数据传输地址宽度，地址传输宽度，写命令。
5. 设置 QSPI_DDLR 寄存器，配置通信时序延时。
6. 设置 QSPI_DSCR 寄存器，设置设备地址，设备页大小和设备块大小。
7. 设置 QSPI_CFGR 寄存器，设置 QSPI 工作模式，主频率，使能 DAC 模式。
8. 设置 QSPI_RDCR 寄存器，设置数据捕获延时。
9. 如果以上设置命令需要设备进入 QE 模式则往设备填写相关进入指令，等待空闲，没有则执行步骤 10。
10. 此时已完成对于 flash 的 DAC 模式的初始化，可对 flash 进行擦读写。

33 安全数字输入输出口（SDIO）

33.1 概述

SDIO 控制器作为数据传输接口，支持 SD 和 eMMC 标准协议，可作为 SD 卡读卡器和 EMMC 卡读卡器的 master 控制器，也支持安全数字 I/O (SDIO)。

33.2 主要特性

- 支持 SD2.0 和 eMMC4.4.1 协议
- 内置 FIFO，宽度为 32bit，深度为 16，支持当 over-run 和 under-run 时停止时钟
- 支持 CRC 产生和校验
- 支持可编程的波特率。支持一种时钟分配率以满足在不同波特率下通信的要求
- 支持时钟控制开关
- 支持卡检测
- 支持卡写保护
- 支持 1bit,4bit,8bit 模式的 SDIO 中断
- 支持 block size 从 1 到 65536
- 支持 DMA 传输

33.3 管脚说明

表 33-1：SDIO 管脚说明

功能管脚	复用管脚	方向	功能描述
SDIO_CK	PC12	Output	时钟
SDIO_CMD	PD2	Output	命令
SDIO_DATA0	PC8	Input/Output	数据线
SDIO_DATA1	PC9	Input/Output	数据线
SDIO_DATA2	PC10	Input/Output	数据线
SDIO_DATA3	PC11	Input/Output	数据线
SDIO_DATA4	PB8	Input/Output	数据线
SDIO_DATA5	PB9	Input/Output	数据线
SDIO_DATA6	PC6	Input/Output	数据线
SDIO_DATA7	PC7	Input/Output	数据线
SDIO_CD	PB4	Input	卡检查信号，低电平有效
SDIO_WP	PB5	Input	写保护

功能管脚	复用管脚	方向	功能描述
SDIO_RSTN	PB6	Output	eMMC复位输出信号

33.4 功能描述

33.4.1 内部 DMA（IDMA）

IDMA 具有一个控制和状态寄存器（CSR）和一个发送/接收引擎，该引擎将数据从主机内存传输到设备端口，反之亦然。控制器在 CPU 干预最小的情况下，利用描述符有效地将数据从源移动到目标。在对控制器进行编程后，可以在数据传输和接收传输完成以及其他正常或错误情况下中断主机 CPU。

IDMA 和主机驱动程序通过单个数据结构进行通信。CSR 地址 0x80 到 0x98 保留用于主机编程。

IDMA 将从卡接收的数据传输到主机的数据缓冲区，并将数据从主机的数据缓冲区传输到 FIFO。驻留在主机内存中的描述符充当指向这些缓冲区的指针。

数据缓冲区位于主机的物理内存中，由完整数据或部分数据组成。缓冲区仅包含数据，而缓冲区状态保留在描述符中。数据链指跨越多个数据缓冲区的数据。但是，单个描述符不能跨越多个数据。

列表的基址写入 DBADDR。描述符列表是前向链接的，最后一个描述符可以指向第一个条目以创建环形结构。描述符列表位于主机的物理内存地址空间中，每个描述符最多可以指向两个数据缓冲区。

描述符

IDMA 使用以下类型的描述符结构：

1. 双缓冲区结构——两个描述符之间的距离由 BMOD 寄存器的 DSL 字段来决定。

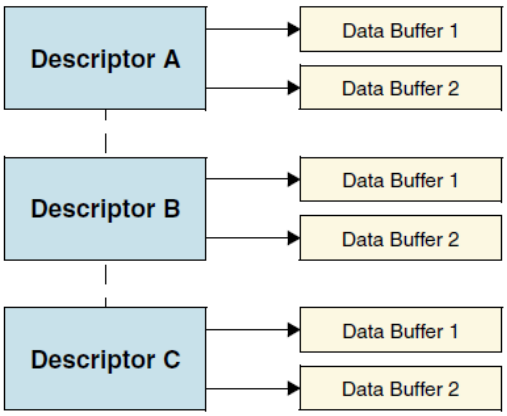


图 33-1：双缓冲区结构

2. 链结构——每个描述符指向一个唯一的缓冲区和下一个描述符。

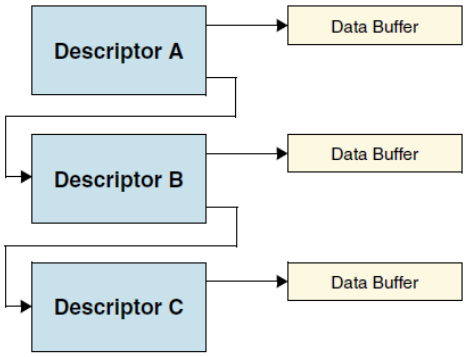


图 33-2：链结构图

描述符地址必须与 32 位 AHB 数据总线的宽度对齐。每个描述符包含 16 字节的控制和状态信息。

DES0 是用于表示[31:0]位的符号，DES1 表示[63:32]位，DES2 表示[95:64]位，DES3 表示[127:96]位。

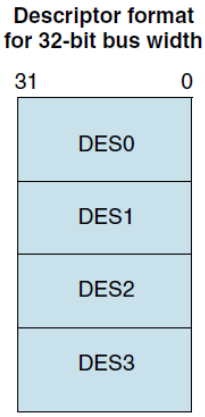


图 33-3：DES0~DES3 结构图

33.4.1.1 DES0

表 33-2：DES0 定义

位	名称	描述
31	OWN	0：描述符归主机所有 1：描述符归 IDMA 所有 IDMA 在完成数据传输时清除该位
30	Card Error Summary (CES)	表示进出卡的事务状态，同时也体现在RINTSTS中 以下位的逻辑或： ▪ EBE – End Bit Error ▪ RTO – Response Timeout/Boot Ack Timeout ▪ RCRC – Response CRC ▪ SBE – Start Bit Error ▪ DRTO – Data Read Timeout/BDS timeout ▪ DCRC – Data CRC for Receive ▪ RE – Response Error
29:6	RSV	保留

位	名称	描述
5	End of Ring (ER)	该位置 1 时，表示描述符列表已到达最后一个描述符。IDMA 返回列表的基址，创建描述符环。这仅对双缓冲区描述符结构有意义。
4	Second Address Chained (CH)	该位置 1 时，表示描述符中的第二个地址是下一个描述符地址，而不是第二个缓冲区地址。 设置此位时，BS2（DES1[25:13]）应全部为零。
3	First Descriptor (FS)	该位置 1 时，表示此描述符包含数据的第一个缓冲区。如果第一个缓冲区的大小为 0，则下一个描述符包含数据的开头。
2	Last Descriptor (LD)	该位与 DMA 传输的最后一个块相关联。该位置 1 时，表示此描述符指向的缓冲区是数据的最后一个缓冲区。当这个描述符完成后，剩余字节计数为 0。
1	Disable Interrupt on Completion (DIC)	该位置 1 时，将阻止设置 IDSTS 的 TI/RI 位，描述符指向的缓冲器的最后一个数据。
0	-	保留

33.4.1.2 DES1

表 33-3：DES1 定义

位	名称	说明
31:26	RSV	保留
25:13	Buffer2 Size (BS2)	表示第二个数据缓冲区字节大小 如果该字段为 0，DMA 将忽略该字段。如果是双缓冲区结构，则返回下一个缓冲区。 此字段对链结构无效（即 DES0[4]被置位时） 如果在任何描述符中将缓冲区 2 大小设置为 0，则剩余的描述符在描述符结束之前，缓冲区 2 大小的值不能为非零值。
12:0	Buffer1 Size (BS1)	数据缓冲区字节大小，此字段不能为零。 注：如果只有一个缓冲器需要编程，则只需使用缓冲器 1，而不需要使用缓冲器 2。

33.4.1.3 DES2

表 33-4：DES2 定义

位	名称	说明
31:0	Buffer Address Pointer1	当使用双缓冲区结构时，这些位表示第一个数据缓冲区的物理地址。对于链描述符，这些位表示数据缓冲区的物理地址。

33.4.1.4 DES3

表 33-5：DES3 定义

位	名称	说明
31:0	Buffer Address Pointer 2/ Next Descriptor Address (BAP2)	当使用双缓冲区结构时，这些位表示第二个缓冲区的物理地址。如果设置了第二个地址链（DES0[4]）位，则该地址包含指向下一个描述符所在物理内存的指针。

33.4.2 中断

表 33-6：SDIO 中断源描述

位	中断	描述
15	最终位错误/无CRC/CRC错误	在读操作中，数据的最终位不是1，在写操作中，没有数据CRC或者CRC错误。
14	自动结束命令	当数据传输结束时，由host自动发送stop命令
13	起始位错误	在读操作中，数据起始位不为0；在4bit模式中，所有的起始位都不为0。
12	改写锁寄存器错误	在命令start位起来后6个系统时钟内，试图改写锁住的命令相关寄存器。
11	FIFO under-run/over-run	FIFO满了，但是HOST要继续往FIFO里搬数据；或者FIFO空了，但是HOST要继续从FIFO中取数据。
10	数据缺失超时中断	如果FIFO满了，还要继续从卡读数据；或者FIFO空了，还要继续向卡写数据，那此时就会停掉输出时钟，以避免丢失数据。在超时寄存器个输出时钟时间内还没有解决问题，中断就会产生。
9	读数据超时	读数据超时时此中断产生，同时数据传输完成中断也会产生
8	应答超时	应答超时时此中断产生，同时命令结束中断也会产生
7	数据CRC错误	从卡收到的数据CRC与内部产生的CRC不match
6	应答CRC错误	从卡收到的应答CRC与内部产生的CRC不match
5	接受数据FIFO请求	从卡读操作中，FIFO count大于等于rx_level
4	发送数据FIFO请求	从卡读操作中，FIFO count小于等于tx_level
3	数据传输完成	数据传输完成后，即使有起始位错误或者CRC错误，中断也会产生。当读数据超时中断产生时，这一位也会置位。 （建议，当数据传输完成后，host应该去把在FIFO剩下的数据搬出去）
2	命令结束	命令发送出去后，收到了卡的应答，即使应答错误或者CRC错误，这一位中断也会置位。当应答超时时这一位也会置位。
1	应答错误	发生以下情况： 1.应答起始位不为0 2.命令index不match 3.应答终止位不为1
0	卡检测	当有卡插入或者拔出时。

33.4.3 数据传输

在数据写命令的应答接收到 2 个时钟之后，host 就会开始数据传输。即使在应答错误或者应答 CRC 错误，也是这样。但是如果应答超时，那么数据传输就会停止。

- Single block传输

如果传输模式为 block 传输，而且字节个数寄存器等于 block size 寄存器，那么就会产生 single block 传输。

- Multiple block传输

如果传输模式为 block 传输，而且字节个数寄存器不等于 block size 寄存器，那么就会产生 multiple block 传输。

如果字节个数为 0，而 block size 不为 0，那么就是一个无终点的传输。Host 将会继续传输数据直到发送 stop 命令。

- 自动停止

当命令寄存器中的 send_auto_stop 置位后，host 会发送一个额外的 block 传输命令把在 FIFO 中剩余的数据传输完毕，然后发送停止命令。

- 时钟控制

时钟的低功耗模式：时钟控制寄存器的低功耗模式位置 1 可以使得输出时钟处于低功耗模式，当卡处于 idle 状态时至少 8 个系统时钟后，会把输出时钟关掉。低功耗模式位会在发送一个新的命令时刷新。另外，当 FIFO over-run 或者 under-run 时，也会把输出时钟关掉。

总结：在以下情况下，输出时钟会被关掉：

clk_en = 1;

低功耗模式下卡处于 idle 状态至少 8 个系统周期；

FIFO 满了，不能再从卡接受数据，因此要关掉时钟以避免 FIFO over-run，导致数据丢失；FIFO 空了，不能再向卡发送数据，因此要关掉时钟以避免 FIFO under-run。

请注意，要改变输出时钟分频率，请先把时钟使能关闭后再改变。

- 错误检测应答：

应答超时：在超时寄存器中写入的那么多个时钟周期内，没有接收到应答的起始位。应答 CRC 错误：应答的 CRC7 与内部产生的 CRC7 不匹配。

应答错误：应答的起始位不是 0，或者命令 index 与发送的命令 index 不匹配，或者应答终止位不是 1。

数据发送:

无 CRC 状态：在一次数据写传输时，如果在数据部分的最后一位两个输出时钟周期后，没有收到 CRC 的起始位，产生数据 CRC 错误，并在中断相应位置 1。然后 host 将停止后面的数据传输。如果 CRC 起始位后的 CRC 状态不是 010，也会产生数据 CRC 错误，并在中断相应位置 1。

FIFO under-run 时，会产生数据传输错误。数据接收据超时：在一次数据读传输时，如果在超时寄存器中写入的那么多个时钟周期内，没有收到数据的起始位，会产生数据超时，并产生 DTO 中断。然后 host 将停止后面的数据传输。

数据起始位错误：在 4bit 或者 8bit 读传输中，如果数据线上没有起始位，那么将产生起始位错误。

数据 CRC 错误：在一次读数据传输中，如果接收到的 CRC16 与内部产生的 CRC16 不匹配，产生 CRC 数据错误并停止后面的数据传输。

数据停止位错误：在一次读传输中，如果停止位不是 1，产生停止位错误并停止后面的数据传输。

FIFO over-run 时，会产生数据传输错误。

33.5 寄存器描述

SDIO 寄存器基地址：0x4050_0000

寄存器列表如下：

表 33-7：SDIO 寄存器列表

偏移地址	名称	描述
0x00	SDIO_CTRL	控制寄存器
0x04	SDIO_POWEN	电源开关寄存器
0x08	SDIO_CLKDIV	时钟分频寄存器
0x10	SDIO_CLKENA	时钟使能寄存器
0x14	SDIO_TIMEOUT	超时寄存器
0x18	SDIO_WIDTH	位宽寄存器
0x1C	SDIO__BLKSIZ	BlockSize 块大小寄存器
0x20	SDIO_BYTCNT	字节个数寄存器
0x24	SDIO_INTMASK	中断屏蔽寄存器
0x28	SDIO_CMDARG	命令参数寄存器
0x2C	SDIO_CMD	命令寄存器
0x30	SDIO_RESP0	应答 0 寄存器
0x34	SDIO_RESP1	应答 1 寄存器
0x38	SDIO_RESP2	应答 2 寄存器
0x3C	SDIO_RESP3	应答 3 寄存器
0x40	SDIO_MNTSTS	可被屏蔽中断状态寄存器
0x44	SDIO_RINTSTS	原始中断状态寄存器
0x48	SDIO_STATUS	状态寄存器
0x4C	SDIO_FIFOTH	FIFO 比较寄存器
0x50	SDIO_CDETECT	卡检测寄存器
0x54	SDIO_WRTPRT	写保护寄存器
0x5C	SDIO_TCBCNT	TCBCNT 寄存器
0x60	SDIO_TBBCNT	TBBCNT 寄存器
0x64	SDIO_DEBOUNCE	DEBOUNCE 寄存器
0x78	SDIO_RST	硬件复位寄存器
0x80	SDIO_BMOD	总线模式寄存器
0x84	SDIO_PLDMND	轮询请求寄存器
0x88	SDIO_DBADDR	描述符列表基地址寄存器
0x8C	SDIO_IDSTS	内部 DMA 状态寄存器
0x90	SDIO_IDINTEN	内部 DMA 中断使能寄存器
0x94	SDIO_DSCADDR	当前主机描述符地址寄存器
0x98	SDIO_BUFADDR	当前缓冲区描述符地址寄存器
0x110	SDIO_ENABLESHIFT	相移使能寄存器
0x200	SDIO_DATA	数据寄存器

33.5.1 控制寄存器（SDIO_CTRL）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:26	RSV	-	-	保留
25	USE_INTERNAL_DMAC	R/W	0x0	仅适用于内部 DMA 配置；否则，保留： 0：主机通过从接口执行数据传输 1：使用内部 DMA 传输数据
24	OD_PULLUP_EN	R/W	0x0	开漏上拉使能： 0：没有开漏上拉 1：有开漏上拉 当设置该位时，命令通常是以开漏方式输出；也就是说，只驱动 0 或高阻，无法驱动 1
23:12	RSV	-	-	保留
11	CEATA_DEVICE_INTERRUPT_STATUS	R/W	0x0	0：CE-ATA 设备中关闭中断 1：CE-ATA 设备中使能中断
10	SEND_AUTO_STOP_CCSD	R/W	0x0	0：如果模块未复位该位，则清除该位 1：发送 CCSD 到 CE-ATA 设备
9	SEND_CCSD	R/W	0x0	0：如果模块未复位该位，则清除该位 1：在发送 CCSD 后，发送内部生成的 STOP 到 CE-ATA 设备
8	ABORT_READ_DATA	R/W	0x0	0：没有变化 1：如果在读数据过程中发送了 suspend 命令后，软件要轮询每张卡以确定发生 suspend 的时间。一旦 suspend 发送，软件将此位置 1 以 reset 数据状态机（此前数据状态机在等待下一个 data block）。数据状态机 reset 后，此位自动清 0
7	SEND_IRQ_RESP	R/W	0x0	0：没有变化 1：自动发送 IRQ 应答 为了等待 MMC 卡中断，Host 发送 CMD40。如果 host 希望退出等待中断状态，就可以把该位置 1，这样 CMD 状态机就会退出等待中断状态，回到 idle 状态。
6	READ_WAIT	R/W	0x0	0：清除读等待 1：发送 read_wait 到 sdio 卡
5	RSV	-	-	保留
4	INT_EN	R/W	0x0	中断使能： 0：关闭中断 1：使能中断
3	RSV	-	-	保留

位	名称	属性	复位值	描述
2	DMA_RST	R/W	0x0	DMA reset： 0：无影响 1：DMA reset 在 DMA reset 后，该位自动清 0
1	FIFO_RST	R/W	0x0	FIFO reset： 0：无影响 1：FIFO reset 在 FIFO reset 后，该位自动清 0
0	HOST_RST	R/W	0x0	Host reset： 0：无影响 1：host reset 复位 在 host reset 后，该位自动清 0

33.5.2 电源开关寄存器（SDIO_POWEN）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	POWER_ENABLE	R/W	0x0	0：关闭电源 1：打开电源

33.5.3 时钟分频寄存器（SDIO_CLKDIV）

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CLK_DIV0	R/W	0x0	7 到 0 位：clock0 的分频为 2*n。比如 clk_div0=0，分频为 2*0=0，没有分频，输出 card 时钟为原频率；clk_div0=1，分频为 2*1=2，分频为 2，输出 card 时钟为原时钟频率的一半，以此类推。

33.5.4 时钟使能寄存器（SDIO_CLKENA）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:17	RSV	-	-	保留

位	名称	属性	复位值	描述
16	CLK_LOW_POWER	R/W	0x0	低功耗控制： 0：非低功耗模式 1：低功耗模式。当卡处于 idle 状态时，停止时钟
15:1	RSV	-	-	保留
0	CLK_EN	R/W	0x0	卡对应的时钟输出使能： 0：关闭时钟 1：使能时钟

33.5.5 超时寄存器（SDIO_TIMEOUT）

偏移地址：0x14

复位值：0xFFFF FF40

位	名称	属性	复位值	描述
31:8	DATA_TIMEOUT	R/W	0xFFFFFF	读写 card 时，读入数据或者写出数据超时时间 超时计数器仅在卡时钟停止后启动
7:0	RESPONSE_TIMEOUT	R/W	0x40	应答的超时时间（cclk_out 时钟的个数）

33.5.6 位宽寄存器（SDIO_WIDTH）

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16	WIDTH1	R/W	0x0	0：非 8 位模式 1：8 位模式 注意：width1 寄存器比 width0 寄存器优先级高。当 width1 为 1 时，controller 为 8 位模式
15:1	RSV	-	-	保留
0	WIDTH0	R/W	0x0	0：1 位模式 1：4 位模式

33.5.7 块大小寄存器（SDIO_BLKSIZ）

偏移地址：0x1C

复位值：0x0000 0200

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:0	BLOCK_SIZE	R/W	0x200	一个 block 所包含的字节个数

33.5.8 字节个数寄存器（SDIO_BYTCNT）

偏移地址：0x20

复位值：0x0000 0200

位	名称	属性	复位值	描述
31:0	BYTE_CNT	R/W	0x200	待传输数据的 byte 个数。byte_cnt 应该是 block_size 的整数倍。 byte_cnt 为 0 表示未定义传输数据的数量，这时候应该由 host 发送 stop、abort 命令来停止数据传输。

33.5.9 中断屏蔽寄存器（SDIO_INTMASK）

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16	SDIO_INT_MASK	R/W	0x0	屏蔽 SDIO 卡中断，屏蔽中断。将相应位清 0 表示屏蔽中断，置 1 表示使能中断。
15:0	INT_MASK	R/W	0x0	屏蔽各个位的中断： 0：屏蔽中断 1：使能中断 bit15：最终位错误（读）/写没有 CRC bit 14：自动命令完成 bit 13：起始位错误/忙清除中断 bit 12：硬件锁定写入错误 bit 11：FIFO under-run/over-run bit 10：数据缺失超时中断 bit 9：读数据超时 bit 8：应答超时 bit 7：数据 CRC 错误 bit 6：应答 CRC 错误 bit 5：接收 FIFO 数据请求 bit 4：发送 FIFO 数据请求 bit 3：数据传输完成 bit 2：命令结束 bit 1：应答错误 bit 0：卡检测

33.5.10 命令参数寄存器（SDIO_CMDARG）

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	CMD_ARG	R/W	0x0	传递给卡的命令参数

33.5.11 命令寄存器（SDIO_CMD）

偏移地址：0x2C

复位值：0x2000 0000

位	名称	属性	复位值	描述
31	START_CMD	R/W	0x0	Start 命令。 当 start_cmd 置 1 时，host 不应该尝试写任何命令寄存器，如果写了的话，raw interrupt 寄存器中的 hardware lock error interrupt 被置 1。 当命令 start 以后，该位被清 0。
30	RSV	-	-	保留
29	USE_HOLD_REG	R/W	0x1	使用 Hold 寄存器： 0：CMD 和 DATA 发送至卡时旁路 Hold 寄存器 1：CMD 和 DATA 发送至卡时通过 Hold 寄存器
28	VOLT_SWITCH	R/W	0x0	0：没有电压切换 1：使能电压切换，只有在 CMD11 时设定
27	BOOT_MODE	R/W	0x0	0：强制启动 1：交替启动
26	DISABLE_BOOT	R/W	0x0	关闭 boot 不能同时设 enable_boot 和 disable_boot
25	EXPECT_BOOT_ACK	R/W	0x0	期待启动 ack 当软件将此位与 enable_boot 一起设置时，期望所选卡要回应 0-1-0 的启动 ack
24	ENABLE_BOOT	R/W	0x0	Boot 使能 仅当强制启动模式下可设，不能同时设 enable_boot 和 disable_boot
23	CCS_EXPECTED	R/W	0x0	0：CE-ATA 设备关闭中断，或者命令不希望来自设备的 CCS 1：CE-ATA 设备使能中断，RW_BLK 命令需要 CE-ATA 设备发出命令完成信号
22	READ_CEATA_DEVICE	R/W	0x0	0：主机不对 CE-ATA 设备执行读操作 1：主机对 CE-ATA 设备执行读操作
21	UPDATE_CLK	R/W	0x0	0：正常的命令序列 1：不发送命令，仅仅更新时钟相关寄存器的值 时钟相关寄存器包括以下寄存器： CLKDIV, CLRSRC, CLKENA
20:17	RSV	-	-	保留
16	CARD_NUM	R/W	0x0	选择 card number。

位	名称	属性	复位值	描述
15	SEND_INI_SEQ	R/W	0x0	0：在发送命令之前不发送初始化序列（80 个 card 时钟）。 1：在发送命令之前发送初始化序列（80 个 card 时钟）。 在 power on 之后，给 card 发送任何命令之前应该先发送 80 个时钟。
14	STOP_ABORT_CMD	R/W	0x0	在发送 stop 命令（cmd12）时，需要把该位置 1；如果在数据传输过程中要发送 reset 命令（CMD0, CMD15, CMD52_reset），必须把把该位置 1，这样就会在命令发送完毕之后停止数据传输。
13	WAIT_PRV_DATA_FINISH	R/W	0x0	0：立刻发送命令，即使此前的数据传输尚未完成。 1：等待此前的数据传输完成后再发送命令。 这里有两种情况： 1）如果此前数据传输已经完成或是字节个数寄存器为 0 的传输，那么就马上发送命令。 2）字节个数寄存器不为 0，那么等待传输完毕后再发送 command。
12	SEND_AUTO_STOP	R/W	0x0	0：数据传输完成后不发送 stop 命令 1：数据传输完成后发送 stop 命令
11	TRANSFER_MODE	R/W	0x0	0：Block 数据传输命令 1：stream 数据传输命令 当没有数据传输时，该位无作用。
10	READ_WRITE	R/W	0x0	0：从卡读数据 1：向卡写数据
9	DATA_TRANSFER_EXPE CTED	R/W	0x0	0：不期望数据传输 1：期望数据传输
8	CHECK_REP_CRC	R/W	0x0	0：不检查应答 CRC 1：检查应答 CRC
7	REP_LONG	R/W	0x0	0：期望卡应答为 48 位 1：期望卡应答为 136 位
6	REP_EXPECT	R/W	0x0	0：不期望得到卡的应答 1：期望得到卡的应答
5:0	CMD_INDEX	R/W	0x0	命令 index

33.5.12 应答 0 寄存器（SDIO_RESP0）

偏移地址：0x30

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	RESPONSE0	R	0x0	应答的[31:0]位

33.5.13 应答 1 寄存器（SDIO_RESP1）

偏移地址：0x34

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	RESPONSE1	R	0x0	应答的[63:32]位

33.5.14 应答 2 寄存器（SDIO_RESP2)

偏移地址：0x38

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	RESPONSE2	R	0x0	应答的[95:64]位

33.5.15 应答 3 寄存器（SDIO_RESP3)

偏移地址：0x3C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	RESPONSE3	R	0x0	应答的[127:96]位

33.5.16 可被屏蔽中断状态寄存器（SDIO_MINTSTS）

偏移地址：0x40

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16	SDIO_INTERRUPT	R	0x0	SDIO 卡中断，表示卡产生了中断。只有当对应位没有被中断屏蔽寄存器屏蔽时，中断位才有效

位	名称	属性	复位值	描述
15:0	INT_STATUS	R	0x0	仅当设置了中断屏蔽寄存器中的相应位时，才使能中断： bit15：最终位错误（读）/写没有 CRC bit 14：自动命令完成 bit 13：起始位错误/忙清除中断 bit 12：硬件锁定写入错误 bit 11：FIFO under-run/over-run bit 10：数据缺失超时中断 bit 9：读数据超时 bit 8：应答超时 bit 7：数据 CRC 错误 bit 6：应答 CRC 错误 bit 5：接收 FIFO 数据请求 bit 4：发送 FIFO 数据请求 bit 3：数据传输完成 bit 2：命令结束 bit 1：应答错误 bit 0：卡检测

33.5.17 原始中断状态寄存器（SDIO_RINTSTS）

偏移地址：0x44

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16	SDIO_INTERRUPT	R/W1	0x0	SDIO 卡中断，表示卡产生了中断。该中断位不受中断屏蔽寄存器影响，始终有效。写 1 清，写 0 无效。
15:0	INT_STATUS	R/W1	0x0	写各个位清除状态位。1 清除状态位，0 保持不变。无论中断屏蔽状态如何，位都会被记录。 bit 15：最终位错误（读）/写没有 CRC bit 14：自动命令完成 bit 13：起始位错误/忙清除中断 bit 12：硬件锁定写入错误 bit 11：FIFO under-run/over-run bit 10：数据缺失超时中断 bit 9：读数据超时 bit 8：应答超时 bit 7：数据 CRC 错误 bit 6：应答 CRC 错误 bit 5：接收 FIFO 数据请求 bit 4：发送 FIFO 数据请求 bit 3：数据传输完成 bit 2：命令结束 bit 1：应答错误 bit 0：卡检测

33.5.18 状态寄存器（SDIO_STATUS）

偏移地址：0x48

复位值：0x0000 0406

位	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:17	FIFO_COUNT	R	0x0	FIFO count
16:11	RESPONSE_INDEX	R	0x0	前一个 command 的 response index
10	DATA_STATE_MACHINE_BUSY	R	0x1	Data 状态机处于 busy 状态 0：不在 busy 状态 1：busy 状态
9	DATA_ BUSY	R	0x1 或者 0x0，取决于 dat[0]的值	Data[0]的取反，卡处于 busy 状态 0：不在 busy 状态 1：busy 状态
8	DATA_3_STATUS	R	0x1 或者 0x0，取决于 dat[3]的值	可以检查卡是否插入读卡器 0：无卡 1：有卡
7:4	RSV	-	-	保留
3	FIFO_FULL	R	0x0	FIFO 已填满
2	FIFO_EMPTY	R	0x1	FIFO 已空
1	FIFO_TX_WATERMARK	R	0x1	FIFO 达到传输 watermark 水平；不符合数据传输的要求。
0	FIFO_RX_MATERMARK	R	0x0	FIFO 达到接收 watermark 电平；不符合数据传输的要求。

33.5.19 FIFO 比较寄存器（SDIO_FIFOTH）

偏移地址：0x4C

复位值：0x000F 0000

位	名称	属性	复位值	描述
31	RSV	-	-	保留
30:28	MULTI_TRANSACTION_ SIZE	R/W	0x0	Multi-block 读写时的 burst size。应该与 DMA controller 的 source/dest msize 相等。 000：1 transfer 001：4 010：8 011：16 100：32 101：64 110：128 111：256

位	名称	属性	复位值	描述
27:16	RX_WMARK	R/W	SDIO:0x0F	从卡读数据时 FIFO 的 level 值。当 FIFO 的数据 count 大于等于这个值时，DMA/FIFO 请求信号起来。在整个传输的末端，不管 level 值的大小，DMA/FIFO 请求信号都会起来使得剩下的数据能被传出去。 在非 DMA 模式下，当 RXDR 中断使能，会产生 RXDR 中断以代替 DMA 请求。在整个传输的末端，当 level 值比剩下的数据个数大的时候，不会产生中断。当看到数据 Transfer Done 中断时，host 应该去读剩下的未完成的数据。 在 DMA 模式下，在整个传输的末端，即使剩下的数据个数比 level 值小，DMA 请求信号也会起来以进行一次单 block 传输，在数据 Transfer Done 中断起来之前传输完所有的数据。
15:12	RSV	-	-	保留
11:0	TX_WMARK	R/W	0x0	向卡写数据时 FIFO 的 level 值。当 FIFO 的数据 count 小于等于这个值时，DMA/FIFO 请求信号起来。在整个传输的末端，不管 level 值的大小，DMA/FIFO 请求信号都会起来。 在非 DMA 模式下，当 TXDR 中断使能，会产生 TXDR 中断以代替 DMA 请求。在整个传输的末端，在最后一次中断产生时，host 要把要求传输数量中尚未完成的数据填充到 FIFO 中去（不是在 FIFO 满之前或者传输完成之后，因为 FIFO 可能并不是空的）。 在 DMA 模式下，在整个传输的末端，如果最后一个传输少于一个 burst size，DMA controller 执行一个 single cycle 直到要求数量的数据传输完成。

注：每当向 FIFO 里搬入数据时，FIFO 的数据 count 加 1；每当从 FIFO 里搬出数据时，FIFO 的数据 count 减 1。

33.5.20 卡检测寄存器（SDIO_CDETECT）

偏移地址：0x50

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留位。
0	CARD_DETECT	R/W	卡检测的输入	0：成功检测到卡 1：未能检测到卡

33.5.21 写保护寄存器（SDIO_WRTPRT）

偏移地址：0x54

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	WRITE_PROTECT	R	写保护的输入	0：没有写保护 1：有写保护

33.5.22 TCBCNT 寄存器（SDIO_TCBCNT）

偏移地址：0x5C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	TRANS_CARD_BYTE_COUNT	R	0x0	由 host 发往卡的数据的字节数 在数据传输过程中，该寄存器读到的是 0

33.5.23 TBBCNT 寄存器（SDIO_TBBCNT）

偏移地址：0x60

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	TRANS_FIFO_BYTE_COUNT	R	0x0	主机/DMA 和 FIFO 之间传输的字节数

33.5.24 DEBOUNCE 寄存器（SDIO_DEBOUNCE）

偏移地址：0x64

复位值：0x00FF FFFF

位	名称	属性	复位值	描述
31:24	RSV	-	-	保留
23:0	DEBOUNCE_COUNT	R/W	0xFFFFFF	防抖滤波器逻辑使用的主机时钟数（clk） 典型的防抖时间为 5-25ms

33.5.25 硬件复位寄存器（SDIO_RST）

偏移地址：0x78

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:1	RSV	-	-	保留
0	CARD_RESET	R/W	1	硬件复位： 0：Reset 1：Active 模式

33.5.26 总线模式寄存器（SDIO_BMOD）

偏移地址：0x80

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10:8	PBL	R	0x0	可编程突发长度： 000：1 次传输 001：4 次传输 010：8 次传输 011：16 次传输 100：32 次传输 101：64 次传输 110：128 次传输 111：256 次传输
7	DE	R/W	0x0	IDMA enable： 0：关闭 IDMA 1：使能 IDMA
6:2	DSL	R/W	0x0	描述符跳过长度：指定要在两个未标记的描述符之间跳过的 HWord/Word/Dword 数（取决于 16/32/64 位总线） 这仅适用于双缓冲结构。
1	FB	R/W	0x0	固定突发传输，控制 AHB 是否执行固定突发传输： 0：AHB 将使用单脉冲和增量脉冲传输操作 1：AHB 将在正常突发传输开始期间仅使用 SINGLE、INCR4、INCR8 或 INCR16
0	SWR	R/W	0x0	软件复位 该位置 1 时，DMA 控制器会复位内部所有寄存器。该位会自动清零。

33.5.27 轮询请求寄存器（SDIO_PLDMND）

偏移地址：0x84

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	PD	W	0x0	轮询请求 如果未设置描述符的 OWN 位，则 FSM 转到暂停状态。主机对该寄存器写任意值，IDMA FSM 恢复正常的描述符提取操作。

33.5.28 描述符列表基地址寄存器（SDIO_DBADDR）

偏移地址：0x88

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	SDL	R/W	0x0	描述符列表的起始。包含第一个描述符的基址。

33.5.29 内部 DMA 状态寄存器（SDIO_IDSTS）

偏移地址：0x8C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12:10	FBE_CODE	R	0x0	总线错误代码，表示导致总线错误的错误类型： 3’b001：传输期间接收到主机中止 3’b010：接收期间接收到主机中止 其他：保留
9	AIS	R/W1	0x0	异常中断摘要。以下两项的逻辑“或”： ■ IDSTS[2]：致命总线中断 ■ IDSTS[4]：DU 位中断 只有未屏蔽的位影响此位，写 1 清零。
8	NIS	R/W1	0x0	正常中断摘要。以下两项的逻辑“或”： ■ IDSTS[0]：传输中断 ■ IDSTS[1]：接收中断 只有未屏蔽的位影响此位，写 1 清零。
7:6	RSV	-	-	保留
5	CES	R/W1	0x0	卡错误摘要 表示进出卡的事务状态，同时也体现在 RINTSTS 中以下位的逻辑或： ■ EBE：End Bit Error ■ RTO：Response Timeout/Boot Ack Timeout ■ RCRC：Response CRC ■ SBE：Start Bit Error ■ DRTO：Data Read Timeout/BDS timeout ■ DCRC：Data CRC for Receive ■ RE：Response Error 写 1 清零。

位	名称	属性	复位值	描述
4	DU	R/W1	0x0	描述符不可用中断 当描述符因 OWN 位=0（DES0[31]=0）而不可用时，该位被置 1。 写 1 清零。
3	RSV	-	-	保留
2	FBE	R/W1	0x0	致命总线错误中断 表示发生总线错误（IDSTS[12:10]）该位被置 1 时，DMA 将禁用其所有总线访问。 写 1 清零。
1	RI	R/W1	0x0	接收中断 表示描述符的数据接收已完成，写 1 清零。
0	TI	R/W1	0x0	发送中断 表示描述符的数据传输已完成，写 1 清零。

33.5.30 内部 DMA 中断使能寄存器（SDIO_IDINTEN）

偏移地址：0x90

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:10	RSV	-	-	保留
9	AI	R/W	0x0	异常中断摘要使能 此位使能以下两个位： IDINTEN[2]：致命总线错误中断 IDINTEN[4]：DU 位中断
8	NIS	R/W	0x0	正常中断摘要使能 此位使能以下两个位： IDINTEN[0]：传输中断 IDINTEN[1]：接收中断
7:6	RSV	-	-	保留
5	CES	R/W	0x0	卡错误摘要中断使能
4	DU	R/W	0x0	描述符不可用中断使能
3	RSV	-	-	保留
2	FBE	R/W	0x0	致命总线错误中断使能
1	RI	R/W	0x0	接收中断使能
0	TI	R/W	0x0	发送中断使能

33.5.31 当前主机描述符地址寄存器（SDIO_DSCADDR）

偏移地址：0x94

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	HDA	R	0x0	主机描述符地址指针，复位时清零。 操作期间 IDMA 更新指针。该寄存器指向 IDMA 读取到的当前描述符的起始地址。

33.5.32 当前缓冲区描述符地址寄存器（SDIO_BUFADDR）

偏移地址：0x98

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	HBA	R	0x0	主机缓冲区地址指针，复位时清零。 操作期间 IDMA 更新指针。该寄存器指向 IDMA 正在访问的当前数据缓冲区地址。

33.5.33 相移使能寄存器（SDIO_ENABLESHIFT）

偏移地址：0x110

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1:0	ENABLE_SHIFT	R/W	0x0	默认设置上的相移的使能控制： 00：默认相移 01：使能移动到下一个正边沿 10：使能移动到下一个负边沿 11：保留

33.5.34 数据寄存器（SDIO_DATA）

偏移地址：0x200

复位值：不确定

位	名称	属性	复位值	描述
31:0	DATA	R/W	-	写：把数据写入 FIFO 读：从 FIFO 中读出数据

33.6 使用流程

33.6.1 初始化

1. 使能 SDIO_CTRL[24]，仅针对 MMC-Ver3.3 模式。
2. 使能电源 SDIO_POWEN[0]。

3. SDIO_RINTSTS 写 0xFFFF_FFFF，清除原始中断状态。
4. 使能中断 SDIO_CTRL[4]，并设定中断屏蔽寄存器 SDIO_INTMASK。
5. 枚举卡片，根据 SD_MMC_CEATA 标准将时钟频率限制为 400kHz。
6. 设定时钟源。
7. 设定参数：response time、data time out、FIFO 阈值。
8. 根据 MMC 标准，仅在枚举阶段需要开漏上拉电阻。因此，对于 MMC-Ver3.3 模式，清除 SDIO_CTRL[24]，禁用开漏上拉。

33.6.2 卡片枚举

- MMC_Card
  1. 清除位宽寄存器 SDIO_WIDTH，将卡宽度设置为 1 位模式。
  2. 设定时钟分频寄存器，将频率设为 400 KHz。
  3. 发送 CMD0，80 个周期的初始化顺序。
  4. 发送 CMD1。
  5. 重复 CMD1，直到接收到的应答的卡忙位被置起（bit31）。
  6. 发送 CMD2。
  7. 发送 CMD3，检查卡检测寄存器 SDIO_CDETECT。

- SD_Card
  1. 从端口 0 开始。
  2. 检查卡是否已连接。
  3. 清零位宽寄存器 SDIO_WIDTH。
  4. 识别卡片类型，SD、MMC、SDIO 或组合卡：

     A. 发送CMD5，参数为0。

     B. 读取应答寄存器（SDIO_RESP0），该寄存器给出了卡支持的电压值。

     C. 再次发送CMD5，参数是电压值。CMD5用于设置电压并将卡状态m/c移出初始化状态。

     D. 检查应答的bit27。如果bit[27]=1，表示内存存在且是组合卡。

     E. 如果是SDIO卡，跳至步骤5；如果是组合卡或者没有收到应答，则执行下面的步骤：

     F. 发送CMD8，参数如下：

        Bit[31:12] = 20’h0

        Bit[11:8] = 4’b0001

        Bit[7:0] = 8’b10101010

     G. 如果收到应答，该卡支持大容量SD2.0，使用以下参数发送ACMD41：

        Bit[31] = 1’b0

Bit[30] = 1’b1

Bit[29:25] = 6’h0

Bit[24] = 1'b1

Bit[23:0] =支持的电压范围

H. 如果收到ACMD41的响应，则该卡为SD；否则，该卡为MMC或CEATA。

I. 如果在ACMD41的应答中bit[24]为1'b1，则主机可以选择将电压切换到1.8V，因为卡支持1.8V。电压切换可通过发送CMD11来完成。

J. 如果未收到初始CMD8的响应，则卡不支持高容量SD2.0；发带有以下参数的CMD0，后跟ACMD41：

Bit[31] = 1’b0；

Bit[30] = 1’b0；

Bit[29:24] = 6’h0；

Bit[23:0] = 支持的电压范围

K. 若收到ACMD41的应答，则卡为SD，否则卡为MMC或CEATA。

5. 根据卡片类型枚举卡片。
6. 使用 400 KHz 的时钟源，进行以下枚举命令：
   - SD：发送CMD0，CMD8，ACMD41，CMD2，CMD3。
   - SDIO：发送CMD5。如果功能计数有效，则发送CMD3。对于SDIO内存部分，执行与SD卡相同的命令。
   - MMC：发送CMD0，CMD1，CMD2，CMD3。
7. 识别 MMC/CE-ATA 设备。
8. 枚举后更改时钟频率。

33.6.3 IDMA

33.6.3.1 初始化

IDMA 初始化流程如下：

1. 写 IDMA 总线模式寄存器（SDIO_BMOD）以设置主机总线访问参数。
2. 写 IDMA 中断使能寄存器（SDIO_IDINTEN）以屏蔽不必要的中断。
3. 软件驱动程序创建发送或接收描述符列表，然后写入 IDMA 描述符列表基地址寄存器（SDIO_DBADDR），为 IDMA 提供列表的起始地址。
4. IDMA 尝试从描述符列表中获取描述符。

33.6.3.2 发送

IDMA 写数据流程如下：

1. 主机设置 DES0-DES3 和 OWN 位（DES0[31]）。
2. 主机将写数据命令写入 SDIO_CMD 寄存器。
3. 主机设置传输阈值（SDIO_FIFOTH[11:0]）。
4. 根据步骤 2，IDMA 确定需要进行写数据传输。
5. IDMA 获取描述符并检查 OWN 位。如果 OWN 位未设置，则表示主机拥有描述符。在这种情况下，IDMA 进入暂停状态，并将 SDIO_IDSTS[4]置 1。在这种情况下，主机需要通过将任何值写入轮询请求寄存器（SDIO_PLDMND）来释放 IDMA。
6. 等待 Command done（CD）位，并且总线接口单元上没有错误，这表明可以完成传输。
7. IDMA 等待的 DMA 接口请求（dw_DMA_req），该请求将根据设定的传输阈值生成。对于无法使用突发访问的最后数据字节，将执行单次传输。
8. IDMA 从数据缓冲区获取传输数据，并传输到 FIFO。
9. 当数据跨越多个描述符时，IDMA 将获取下一个描述符并继续对下一个描述符进行操作。描述符中的最后一个描述符位指示数据是否跨越多个描述符。
10. 数据传输完成后，如果使能了传输中断，那么将在传输中断中更新 IDSTS 寄存器中的状态信息。此外，IDMA 通过对 DES0 执行写入事务来清除 OWN 位。

33.6.3.3 接收

IDMA 读数据流程如下

1. 主机设置 DES0-DES3 和 OWN 位（DES0[31]）。
2. 主机将读数据命令写入 SDIO_CMD 寄存器。
3. 主机设置接收阈值（SDIO_FIFOTH[27:16]）。
4. 根据步骤 2，IDMA 确定需要进行读数据传输。
5. IDMA 获取描述符并检查 OWN 位。如果 OWN 位未设置，则表示主机拥有描述符。在这种情况下，IDMA 进入暂停状态，并将 SDIO_IDSTS[4]置 1。在这种情况下，主机需要通过将任何值写入轮询请求寄存器（SDIO_PLDMND）来释放 IDMA。
6. 等待 Command done（CD）位，并且总线接口单元上没有错误，这表明可以完成传输。
7. IDMA 等待的 DMA 接口请求（dw_DMA_req），该请求将根据设定的接收阈值生成。对于无法使用突发访问的最后数据字节，将执行单次传输。
8. IDMA 从 FIFO 获取数据并传输到主机。
9. 当数据跨越多个描述符时，IDMA 将获取下一个描述符并继续对下一个描述符进行操作。描述符中的最后一个描述符位指示数据是否跨越多个描述符。

10. 数据接收完成后，如果使能了接收中断，那么将在接收中断中更新IDSTS寄存器中的状态信息。此外，IDMA 通过对 DES0 执行写入事务来清除 OWN 位。

33.6.4 SDIO 使用要点

- **软件编程要求**

1. 在一个命令发送完成之前，不要改变跟命令相关的各个寄存器。这些寄存器包括：命令寄存器，命令参数寄存器，字节个数寄存器，block size寄存器，时钟分频率寄存器，时钟使能寄存器，超时寄存器，位宽寄存器。
2. 跟数据传输相关各个参数的设置请在数据传输之前设置好。当一次传输开始后，不要改变各个参数。
3. 当要发送一个命令时，将SDIO_CMD[31]置1。此时，如果SDIO_CMD[21]=0，所有与命令发送相关的参数（命令寄存器，命令参数寄存器，字节个数寄存器，block size寄存器，位宽寄存器，超时寄存器，时钟使能寄存器，时钟分频率寄存器）将被更新。这个更新需要一定的时间（6个系统时钟），在这段时间内不应该再改变命令相关的参数，否则会发生改写锁寄存器错误。一旦命令被卡接受，那就可以发送下一个命令，但是这会有以下几种情况：
   - 如果前一个命令不是数据传输命令，新的命令将会在前一个命令完成后发送。
   - 如果前一个命令是数据传输命令，并且SDIO_CMD[13]置位，新的命令会在数据完成之后再发送。
   - 如果SDIO_CMD[13]为 0，新的命令会在前一个命令完成之后发送。比如，你要停止数据传输，或者你要在数据传输过程中查询卡状态。
4. 当数据传输时，命令不会发送。
5. 一次只能读写一张卡，也只能向一张卡发送命令。比如，当你与一张卡传输数据时，你不能同时向另一种卡发送命令。
6. 对于同一张卡来说，一次只能发送一个命令。
7. 在一个字节个数寄存器为0的写操作中，如果由于FIFO空了而导致时钟停止，那么应该先把FIFO填上数据，使得时钟开始，然后才能发送stop命令。
8. 如果在数据传输过程中要发送reset命令（CMD0, CMD15, CMD52_reset），必须把命令寄存器中的 SDIO_CMD[14]置位。这样就会在命令发送完毕之后停止数据传输。
9. 如果在读卡时，由于FIFO满了导致时钟停止，软件应该至少读两个FIFO才能使得时钟恢复。
10. 在传输时，如果卡被暂停或者停止了，恢复的时候应该把fifo reset一下。

在发送一个新的命令之前，需要确保卡不处于 busy 状态。在改变输出时钟频率之前，软件必须保证当前没有数据或者命令传输。

为了避免在输出时钟产生毛刺，改变卡时钟之前应该使用下列步骤：

1. 关闭时钟使能寄存器并用update_clk更新。为了保证之前的命令已经完成，先置位以下位：SDIO_CMD[31]，SDIO_CMD[21]，SDIO_CMD[13]。
2. 写时钟分频率寄存器，置位SDIO_CMD[31]。
3. 打开时钟使能，然后置位SDIO_CMD[31]。

- **读操作流程**

1. 设置字节个数寄存器
2. 设置block size寄存器
3. 初始化命令
4. 设置命令参数寄存器
5. 发送CMD17/18

- **写操作流程**

1. 设置字节个数寄存器
2. 设置 block size 寄存器
3. 初始化命令
4. 设置命令参数寄存器
5. 发送CMD24/25

34 以太网接收发送器（EMAC）

34.1 概述

EMAC 可以接收和发送以太网数据，符合 IEEE802.3-2002 标准。

34.2 特性

- 支持 MII、RMII 和 RGMII 接口
- 支持外部 PHY 接口实现 10M/100M/1000M bit/s 的数据传输速率
- 支持全双工和半双工工作模式
- 使用 MDIO 接口配置和管理 PHY 设备
- 支持以太网时间戳（IEEE 1588-2002）。每个帧的发送或接收状态下给出 64 位时间戳
- 报头和帧起始数据（SFD）在发送路径中插入、在接收路径中删除
- 支持帧长有效性检测，丢弃超长帧和超短帧
- 支持对输入帧进行 CRC 校验，可丢弃校验错的帧
- 支持对输出帧添加 CRC 校验
- 支持短帧填充功能
- 支持对接收和发送帧进行统计计数
- 支持广播帧、组播帧和单播帧过滤
- 支持控制报文、IP 报文、广播或多播报文限速处理功能可配置
- 支持包过滤功能
- 支持入队中断和超时中断两种中断方式
- 支持收发包缓存
- 支持 EEE 能效以太网功能
- 相互独立的两个 2K 字节的 FIFO 分别用于发送与接收

34.3 管脚说明

表 34-1：EMAC 管脚说明

功能管脚	RGMII	RMII	MII	复用管脚	方向	功能描述
ETH_GTXCLK/ ETH_TXCLK	ETH_GTXCLK	-	ETH_TXCLK	PC3	Input/ Output	RGMII：通道发送时钟 125M 输出 MII：通道发送时钟输入

功能管脚	RGMII	RMII	MII	复用管脚	方向	功能描述
ETH_GTXCLK/ ETH_TXCLK	ETH_RXCLK	ETH_RXCLK/ ETH_REF_50M	ETH_RXCLK	PA1	Input	通道接收时钟 10M/100M/1000M，RGMII/RMII/MII, RMII(50M)
ETH_CLK125M (REF_50M_125M)	ETH_CLK125M	-	-	PD2	Input	通道参考 125M 时钟，RGMII
ETH_TXD0	ETH_TXD0	ETH_TXD0	ETH_TXD0	PB12	Output	通道发送数据 0
ETH_TXD1	ETH_TXD1	ETH_TXD1	ETH_TXD1	PB13	Output	通道发送数据 1
ETH_TXD2	ETH_TXD2	-	ETH_TXD2	PC2	Output	通道发送数据 2
ETH_TXD3	ETH_TXD3	-	ETH_TXD3	PB8,PE2	Output	通道发送数据 3
ETH_TXEN	ETH_TXEN	ETH_TXEN	ETH_TXEN	PB11	Output	通道发送数据使能
ETH_RXD0	ETH_RXD0	ETH_RXD0	ETH_RXD0	PC4	Input	通道接收数据 0
ETH_RXD1	ETH_RXD1	ETH_RXD1	ETH_RXD1	PC5	Input	通道接收数据 1
ETH_RXD2	ETH_RXD2	-	ETH_RXD2	PB0	Input	通道接收数据 2
ETH_RXD3	ETH_RXD3	-	ETH_RXD3	PB1	Input	通道接收数据 3
ETH_RXDV	ETH_RXDV	ETH_RXDV	ETH_RXDV	PA7	Input	通道接收数据有效
ETH_MDIO	ETH_MDIO	ETH_MDIO	ETH_MDIO	PA2	Input/ Output	通道串行数据
ETH_MDC	ETH_MDC	ETH_MDC	ETH_MDC	PC1	Output	通道串行时钟输出
ETH_CRS	-	-	-	PA0	Input	-
ETH_COL	-	-	-	PA3	Input	-
ETH_RX_ER	-	-	-	PB10	Input	-
ETH_TX_ER	-	-	-	PD6	Output	-

34.4 EMAC 寄存器列表

EMAC 寄存器基地址：0x4010_0000

EMAC 寄存器列表如下：

表 34-2：EMAC 寄存器列表

地址偏移	寄存器名	说明
0x00	EMAC_CONFIG	EMAC 设置寄存器
0x04	EMAC_FRAMEFILTER	帧过滤寄存器
0x08	EMAC_HASHTABLEHIGH	Hash 表高位寄存器
0x0C	EMAC_HASHTABLELOW	Hash 表低位寄存器
0x10	EMAC_GMIIADDRESS	GMII 地址寄存器
0x14	EMAC_GMIIDATA	GMII 数据寄存器
0x18	EMAC_FLOWCONTROL	流控寄存器
0x1C	EMAC_VLANTAG	VLAN 标签寄存器
0x24	EMAC_DEBUGER	调试寄存器
0x30	EMAC_LPICS	LPI 控制状态寄存器
0x34	EMAC_LPIT	LPI 计时寄存器
0x38	EMAC_INTSTATUS	中断状态寄存器
0x3C	EMAC_INTMASK	中断屏蔽寄存器
0x40	EMAC_ADDR0H	EMAC 地址 0 高位寄存器
0x44	EMAC_ADDR0L	EMAC 地址 0 低位寄存器
0x48	EMAC_ADDR1H	EMAC 地址 1 高位寄存器

地址偏移	寄存器名	说明
0x4C	EMAC_ADDR1L	EMAC 地址 1 低位寄存器
0x50	EMAC_ADDR2H	EMAC 地址 2 高位寄存器
0x54	EMAC_ADDR2L	EMAC 地址 2 低位寄存器
0x58	EMAC_ADDR3H	EMAC 地址 3 高位寄存器
0x5C	EMAC_ADDR3L	EMAC 地址 3 低位寄存器
0x60	EMAC_ADDR4H	EMAC 地址 4 高位寄存器
0x64	EMAC_ADDR4L	EMAC 地址 4 低位寄存器
0x68	EMAC_ADDR5H	EMAC 地址 5 高位寄存器
0x6C	EMAC_ADDR5L	EMAC 地址 5 低位寄存器
0x70	EMAC_ADDR6H	EMAC 地址 6 高位寄存器
0x74	EMAC_ADDR6L	EMAC 地址 6 低位寄存器
0x78	EMAC_ADDR7H	EMAC 地址 7 高位寄存器
0x7C	EMAC_ADDR7L	EMAC 地址 7 低位寄存器
0x80	EMAC_ADDR8H	EMAC 地址 8 高位寄存器
0x84	EMAC_ADDR8L	EMAC 地址 8 低位寄存器
0x88	EMAC_ADDR9H	EMAC 地址 9 高位寄存器
0x8C	EMAC_ADDR9L	EMAC 地址 9 低位寄存器
0x90	EMAC_ADDR10H	EMAC 地址 10 高位寄存器
0x94	EMAC_ADDR10L	EMAC 地址 10 低位寄存器
0x98	EMAC_ADDR11H	EMAC 地址 11 高位寄存器
0x9C	EMAC_ADDR11L	EMAC 地址 11 低位寄存器
0xA0	EMAC_ADDR12H	EMAC 地址 12 高位寄存器
0xA4	EMAC_ADDR12L	EMAC 地址 12 低位寄存器
0xA8	EMAC_ADDR13H	EMAC 地址 13 高位寄存器
0xAC	EMAC_ADDR13L	EMAC 地址 13 低位寄存器
0xB0	EMAC_ADDR14H	EMAC 地址 14 高位寄存器
0xB4	EMAC_ADDR14L	EMAC 地址 14 低位寄存器
0xB8	EMAC_ADDR15H	EMAC 地址 15 高位寄存器
0xBC	EMAC_ADDR15L	EMAC 地址 15 低位寄存器
0xD8	EMAC_RGMIICS	RGMII 控制状态寄存器
0xDC	EMAC_WDG	超时看门狗寄存器

34.4.1 EMAC 设置寄存器（EMAC_CONFIG）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:28	RSV	-	-	-
27	TKPE	R/W	0	当 bit20（JE）为 0 而此位为 1 时，EMAC 将所有超过 2000 字节的帧认定为巨大帧，少于 2000 字节的认定为普通帧。 当 bit20（JE）和此位均为 0 时，EMAC 将所有超过

位	名称	属性	复位值	说明
				1518 字节的帧认定为巨大帧。
26:25	RSV	-	-	-
24	TC	R/W	0	1：激活 RGMII 的 PHY 的双工模式、连接速度和上下传信息 0：不传输这些信息到 PHY
23	WDGPD	R/W	0	1：关闭接收器看门狗，最多能接收 16383 字节的帧 0：打开看门狗，只允许接收 2048 字节（JE 打开时 10240 字节）的数据
22	JPD	R/W	0	1：关闭 Jabber 定时器，最多能传输 16383 字节数据 0：开启 Jabber 定时器，只允许传输 2048 字节（JE 打开时 10240 字节）的数据
21	BURST	R/W	0	1：保留 0：不开启
20	JE	R/W	0	1：允许接收一个 9018 字节以内的帧而不触发巨大帧错误 0：不开启
19:17	IFG	R/W	0	设定帧之间的间隔： 000：96 位时间 001：88 位时间 010：80 位时间 …… 111：40 位时间 半双工模式下至少需要 64 位时间(设定为 100)。
16	CSPD	R/W	0	1：在半双工模式下无视 MII 的 CRS 信号，载波丢失时不报错 0：打开载波检测
15:14	SPEED	R/W	0	选择速度： 00/01：1000Mbps 10：10Mbps 11：100Mbps
13	RCPD	R/W	0	1：在半双工模式下，发送时不接收帧 0：在半双工模式下，发送时接收帧
12	LB	R/W	0	1：开启 MII 下的自循环模式，需要 RX 时钟输入 0：不开启
11	DP	R/W	0	1：全双工 0：半双工
10	CSE	R/W	0	1：开启 TCP/UDP/ICMP header 和 payload 校验 0：关闭
9	RTPD	R/W	0	1：关闭重试，当 MII 接口发生冲突时会放弃传输并报告错误 0：开启重试
8	RSV	-	-	保留
7	ACS	R/W	0	1：接收帧少于 1536 字节时会自动去除填充字节和 CRC 域 0：关闭
6:5	BL	R/W	0	设定在半双工模式下发生传输冲突后，重试前的等待时间 00：k = min（n, 10） 01：k = min（n, 8）

位	名称	属性	复位值	说明
				10：k = min（n, 4） 11：k = min（n, 1） n=重试次数 随机等待时间先从[0, 2^k]当中取值，1000Mbps 下再乘以 4096 位时间，10/100Mbps 下再乘以 512 位时间。
4	DC	R/W	0	1：在半双工模式下，帧发送延迟了 24288（10/100Mbps 模式下）或 155680（1000Mbps 模式下或开启了 JE 的情况下）个位时间后，中止发送并产生顺延过久错误； 0：一直延迟直到 CRS 信号无效后继续进行发送。
3	TE	R/W	0	1：开启发送 0：关闭发送，已经开始的发送帧仍会继续完成
2	RE	R/W	0	1：开启接收 0：关闭接收，已经开始接收的帧仍会继续完成
1:0	PL	R/W	0	前言长度： 00：7 字节 01：5 字节 10：3 字节

34.4.2 帧过滤寄存器（EMAC_FRAMEFILTER）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	说明
31	RA	R/W	0	1：接收所有帧，无视过滤器设定 0：按照过滤器设定，只接收满足条件的帧
30:22	RSV	-	-	-
21	DNTU	R/W	0	1：过滤非 TCP 或 UDP 的 IP 超界帧 0：不过滤
20	IPFE	R/W	0	1：过滤线路 3 和线路 4 过滤器中指定的帧 0：不过滤
19:17	RSV	-	-	-
16	VTFE	R/W	0	1：过滤 VLAN 标签不能成功比较的帧 0：不过滤
15:11	RSV	-	-	-
10	HPF	R/W	0	1：根据 HMC 和 HUC 的设定来决定过滤的条件 0：如果 HMC 或 HUC 为 1，只要符合 HASH 过滤器就能通过
9	SAF	R/W	0	1：过滤源地址不满足设定的帧 0：不过滤
8	SAIF	R/W	0	1：源地址过滤条件反转 0：源地址过滤条件正常
7:6	PCF	R/W	0	配置控制帧的传送： 00：过滤所有控制帧 01：过滤暂停帧 10：传送所有控制帧 11：过滤不满足地址过滤器的控制帧

位	名称	属性	复位值	说明
5	DBF	R/W	0	1：过滤广播帧 0：不过滤
4	PM	R/W	0	1：不过滤多播帧（地址第一位为 1 的） 0：多播帧受 HMC 影响
3	DAIF	R/W	0	1：目的地址过滤条件反转 0：目的地址过滤条件正常
2	HMC	R/W	0	1：收到多播帧时根据 HASH 列表对目的地址进行过滤 0：收到多播帧时对目的地址进行完美过滤（与目的地址寄存器比较）
1	HUC	R/W	0	1：收到单播帧时根据 HASH 列表对目的地址进行过滤 0：收到单播帧时对目的地址进行完美过滤（与目的地址寄存器比较）
0	PR	R/W	0	1：关闭地址过滤 0：开启地址过滤

34.4.3 HASH 表高位寄存器 (EMAC_HASHTABLEHIGH)

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	HTH	R/W	0	进行 HASH 过滤时，帧中的目的地址被送入 CRC 逻辑，随后 CRC 寄存器中的高 6 位会被用于指定 HASH 表中的 64 位中的 1 位，如果 HASH 表中该位为 1，则可以通过过滤。

34.4.4 HASH 表低位寄存器 (EMAC_HASHTABLELOW)

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	HTL	R/W	0	进行 HASH 过滤时，帧中的目的地址被送入 CRC 逻辑，随后 CRC 寄存器中的高 6 位会被用于指定 HASH 表中的 64 位中的 1 位，如果 HASH 表中该位为 1，则可以通过过滤。

34.4.5 GMII 地址寄存器 (EMAC_GMIIADDRESS)

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:16	RSV	-	-	-
15:11	PA	R/W	0	正在访问的 PHY 的地址
10:6	GR	R/W	0	用于选定 PHY 中的寄存器

位	名称	属性	复位值	说明
5:2	CR	R/W	0	选定 MDC 时钟频率： 0000：MDCCLK=HCLK/42，HCLK 在 60~100MHz 时使用； 0001：MDCCLK=HCLK/62，HCLK 在 100~150MHz 时使用； 0010：MDCCLK=HCLK/16，HCLK 在 20~35MHz 时使用； 0011：MDCCLK=HCLK/26，HCLK 在 35~60MHz 时使用； 0100：MDCCLK=HCLK/102，HCLK 在 150~250MHz 时使用； 0101：MDCCLK=HCLK/124，HCLK 在 250~300MHz 时使用。 以下设定会使 MDC 时钟频率过快，需要确认设备能否支持 1000：MDCCLK=HCLK/4； 1001：MDCCLK=HCLK/6； 1010：MDCCLK=HCLK/8； 1011：MDCCLK=HCLK/10； 1100：MDCCLK=HCLK/12； 1101：MDCCLK=HCLK/14； 1110：MDCCLK=HCLK/16； 1111：MDCCLK=HCLK/18。
1	WR	R/W	0	1：对 PHY 进行写入操作 0：对 PHY 进行读取操作
0	BUSY	R	0	指示正在对 PHY 寄存器进行操作，此时不要改变 GMII 数据寄存器的数值。

34.4.6 GMII 数据寄存器 (EMAC_GMIIDATA)

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:16	RSV	-	-	-
15:0	DATA	R/W	0	PHY 寄存器读写数据

34.4.7 流控寄存器（EMAC_FLOWCONTROL）

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:16	PT	R/W	0	设定控制帧 PAUSE 时间域的值
15:8	RSV	-	-	-

位	名称	属性	复位值	说明
7	DZPQ	R/W	0	1：在解除 FIFO 层流控信号时，不自动生成零值暂停帧； 0：会自动生成零值暂停帧。
6	RSV	-	-	-
5:4	PLT	R/W	0	设定 Pause 低阈值，需要小于 PT 的值： 00：4 01：28 10：144 11：256 单位是接口上传输 64 字节的时间
3	UP	R/W	0	1：EMAC 除了检测带唯一多播地址的暂停帧以外，还会带单播地址的暂停帧； 0：只检测带唯一多播地址的暂停帧。
2	RFE	R/W	0	1：EMAC 解析暂停帧，并发送一段时间； 0：不解析暂停帧。
1	TFE	R/W	0	**全双工模式下：** 1：激活流控，可发送暂停帧； 0：关闭流控，不发送暂停帧。 **半双工模式下：** 1：开启背压功能； 0：关闭背压功能。
0	FCBBPA	R/W	0	**全双工模式下：** 对此位写 1 可以发送一个暂停帧，自动清零，未清零的情况下不要操作此寄存器。 **半双工模式下：** 当此位和 TFE 均为 1 时，EMAC 会开启背压功能，在接收到帧的时候发送阻塞信号来产生线路冲突。

34.4.8 VLAN 标签寄存器（EMAC_VLANTAG）

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:20	RSV	-	-	-
19	VTHE	R/W	0	1：开启 VLAN 标签 HASH 表匹配功能 0：不开启
18	ESVL	R/W	0	1：将 S-VLAN（88A8）类型的帧认定为 VLAN 标签帧 0：不使用 S-VLAN
17	VTIM	R/W	0	1：VLAN 标签匹配条件反转 0：VLAN 标签匹配条件正常
16	ETV	R/W	0	1：使用 12 位标签进行比较 0：使用 16 位标签进行比较

位	名称	属性	复位值	说明
15:0	VLT	R/W	0	VLAN 标签标识符： [15:13]：用户优先级 [12]：CFI/DEI [11:0]：VID 如果 VLT 全部为 0，就不会比较 VLAN 标签并且认定所有 8100 和 88A8 类型的帧为 VLAN 帧。

34.4.9 调试寄存器（EMAC_DEBUGER）

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:26	RSV	-	-	-
25	TXSTSFSTS	R	0	指示 MTL 发送 FIFO 已满
24	TXDSTS	R	0	指示 MTL 发送 FIFO 非空
23	RSV	-	-	-
22	TWCSTS	R	0	指示 MTL 发送 FIFO 写入控制器正在工作
21:20	TRCSTS	R	0	指示 MTL 发送 FIFO 读取控制器状态： 00：空闲 01：正在读取 10：等待 EMAC 发送传输请求 11：正在写入或清空
19	TXPAUSED	R	0	指示 EMAC 发送传输正处于暂停状态
18:17	TFCSTS	R	0	指示 EMAC 发送传输控制器状态： 00：空闲 01：等待前一帧完成 10：产生或发送暂停帧 11：正在取得将要发送的帧
16	TPESTS	R	0	指示 EMAC MII 发送协议引擎正在工作
15:10	RSV	-	-	-
9:8	RXFSTS	R	0	指示 MTL 接收 FIFO 的状态： 00：空 01：低于非激活阈值 10：高于激活阈值 11：满
7	RSV	-	-	-
6:5	RRCSTS	R	0	指示 MTL 接收 FIFO 读取控制器状态： 00：空闲 01：读取帧数据 10：读取帧状态或时间戳 11：正在清空
4	RWCSTS	R	0	指示 MTL 接收 FIFO 控制器正在工作
3	RSV	-	-	保留
2	RFCFCSTRS	R	0	指示接收帧 FIFO 读控制器正在工作
1	RFCFCSTWS	R	0	指示接收帧 FIFO 写控制器正在工作
0	RPESTS	R	0	指示 EMAC MII 接收协议引擎正在工作

34.4.10 LPI 控制状态寄存器（EMAC_LPICS）

偏移地址：0x30

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:20	RSV	-	-	-
19	LPITXA	R/W	0	1：EMAC 在传输完所有帧后才受 LPIEN 影响进入 LPI 模式 0：EMAC 直接受 LPIEN 影响进入 LPI 模式
18	PLSEN	R/W	0	1：LPI 触发受 RGMII 控制状态寄存器和 PLS 影响 0：LPI 触发只受 PLS 影响
17	PLS	R/W	0	1：认定连接正常 0：认定连接断开
16	LPIEN	R/W	0	1：令 EMAC 进入 LPI 模式 0：EMAC 处于正常模式 如果 LPITXA 为 1，开始一个新的发送传输时此位会自动清零。
15:10	RSV	-	-	-
9	RLPIST	R	0	指示 EMAC 正在从 MII 接口接收 LPI 信号
8	TLPIST	R	0	指示 EMAC 正在向 MII 接口发送 LPI 信号
7:4	RSV	-	-	-
3	RLPIEX	R	0	指示从 MII 不再收到 LPI 信号并退出 LPI 模式，读此寄存器后清零
2	RLPIEN	R	0	指示从 MII 收到 LPI 信号并进入了 LPI 模式，读此寄存器后清零
1	TLPIEX	R	0	指示 EMAC 发送器已经因为 LPIEN 位的设定退出了 LPI 模式，读此寄存器后清零
0	TLPIEN	R	0	指示 EMAC 发送器已经因为 LPIEN 位的设定进入了 LPI 模式，读此寄存器后清零

34.4.11 LPI 计时寄存器（EMAC_LPIT）

偏移地址：0x34

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:26	RSV	-	-	-
25:16	LST	R/W	0x3E8	设定 EMAC 在满足条件后向 PHY 发送 LPI 信号前的等待时间
15:0	TWT	R/W	0	设定 EMAC 退出 LPI 模式后到进行下一次传输前的等待时间

34.4.12 中断状态寄存器（EMAC_INTSTATUS）

偏移地址：0x38

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:11	RSV	-	-	-
10	LPIIS	R	0	LPI 模式进入与退出中断状态，读 LPICS 寄存器清零
9	TSIS	R	0	时间戳中断状态，系统时间超过目标高低时间界限时触发，读此寄存器清零
8	RSV	-	-	-
7	MMCRIPIS	R	0	MMC 接收校验中断状态，反映 MMCRCI 寄存器中的中断状态，读 MMCRCI 寄存器清零
6	MMCTIS	R	0	MMC 发送中断状态，反映 MMCTI 寄存器中的中断状态，读 MMCTI 寄存器清零
5	MMCRIS	R	0	MMC 接收中断状态，反映 MMRTI 寄存器中的中断状态，读 MMRTI 寄存器清零
4	MMCIS	R	0	MMC 中断状态，反映 bit[7:5]中的中断状态
3:1	RSV	-	-	-
0	RGMIIIS	R	0	RGMII 中断状态，RGMII 连接状态改变时触发，读 RGMIICS 寄存器清零

34.4.13 中断屏蔽寄存器（EMAC_INTMASK）

偏移地址：0x3C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:11	RSV	-	-	-
10	LPIIM	R/W	0	1：LPI 模式进入与退出中断不会触发 0：中断会触发
9	TSIM	R/W	0	1：时间戳中断不会触发 0：中断会触发
8:1	RSV	-	-	-
0	RGMIIIM	R/W	0	1：RGMII 中断不会触发 0：中断会触发

34.4.14 EMAC 地址 0~15 高位寄存器（EMAC_ADDR0~15 H）

偏移地址：：0x40，0x48，… ，0xB8，x=0~15

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:16	RSV	-	0x8000	-
15:0	AxH	R/W	0	EMAC 地址的高 16 位

34.4.15 EMAC 地址 0~15 低位寄存器（EMAC_ADDR0~15L）

偏移地址：0x44，0x4C，… ，0xBC，x=0~15

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	AxL	R/W	0	EMAC 地址的低 32 位

34.4.16 RGMII 控制状态寄存器（EMAC_RGMIICS）

偏移地址：0xD8

复位值：0x0000 0000

位	名称	R/W	复位值	说明
31:4	RSV	-	-	-
3	LSTATUS	R	0	指示 MAC 与 PHY 的连接状态
2:1	LSPEED	R	0	指示连接速度： 00：2.5MHz 01：25MHz 10：125MHz
0	LM	R	0	指示连接模式： 1：全双工 0：半双工

34.4.17 超时看门狗寄存器（EMAC_WDG）

偏移地址：0xDC

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:17	RSV	-	-	-
16	WDGEN	R/W	0	如果此位为 1 而且 EMAC Config 中的 bit23 WDGPD 为 0，则使用此寄存器中的计时来控制对超时帧的判定。
15:14	RSV	-	-	-
13:0	TOT	R/W	0	设定超时时间，超时帧会被认定成错误帧（至少应当大于 1522）。

34.5 PTP 寄存器列表

EMAC 寄存器基地址：0x4010_0000

PTP 寄存器列表：

表 34-3：PTP 寄存器列表

地址偏移	寄存器名	说明
0x700	PTP_TSCTL	时间戳控制寄存器
0x704	PTP_SUBSECINC	时间戳亚秒递增寄存器
0x708	PTP_SEC	时间戳秒数寄存器
0x70C	PTP_NANOSEC	时间戳纳秒寄存器
0x710	PTP_SECUD	时间戳秒数更新寄存器
0x714	PTP_NANOSECUD	时间戳纳秒更新寄存器
0x718	PTP_TSADDEND	时间戳加数寄存器

地址偏移	寄存器名	说明
0x71C	PTP_TGTSEC	时间戳目标秒数寄存器
0x720	PTP_TGTNANOSEC	时间戳目标纳秒寄存器

34.5.1 时间戳控制寄存器（PTP_TSCTL）

偏移地址：0x700

复位值：0x0000 2000

位	名称	属性	复位值	说明
31:19	RSV	-	-	-
18	ADDRFLT	R/W	0	1：发送 PTP 帧时过滤地址 0：不过滤
17:16	SNAPTPYE	R/W	0	选择需要时间戳快照的 PTP 数据包 参考表 34-3
15	MSTSNAP	R/W	0	1：主节点相关的消息使用快照 0：不使用
14	EVENTSNAP	R/W	0	1：事件相关的消息使用快照 0：不使用
13	TSIPV4EN	R/W	1	1：接收器处理 IPV4-UDP 封装的 PTP 数据包 0：忽略 IPV4-UDP 封装的 PTP 数据包
12	TSIPV6EN	R/W	0	1：接收器处理 IPV6-UDP 封装的 PTP 数据包 0：忽略 IPV6-UDP 封装的 PTP 数据包
11	TSIPEN	R/W	0	1：接收器处理直接传输的 PTP 数据包 0：忽略直接传输的 PTP 数据包
10	TSV2EN	R/W	0	1：使用 IEEE 1588 V2 格式处理数据包 0：使用 IEEE 1588 V1 格式处理数据包
9	TSDBRC	R/W	0	1：时间戳低寄存器在达到 0x3B9AC9FF 后滚动并使时间戳高秒递增 0：时间戳低寄存器在达到 0x7FFFFFFF 后滚动并使时间戳高秒递增 必须根据 PTP 参考时钟频率和该位的值来设置亚秒递增量
8	TSENALL	R/W	0	1：激活所有接收帧的时间戳快照功能 0：不激活
7:6	RSV	-	-	-
5	ADDENDUD	R/W	0	对此位写 1 使 TSADDEND 寄存器的内容更新，更新完成后会自动清零，写 1 前需确保此位为 0
4	INTRTRIG	R/W	0	对此位写 1 使能中断，在系统时间大于目标秒数时触发，触发中断会使此位自动清零
3	TSUD	R/W	0	对此位写 1 将 SECUD 和 NANOSECUD 寄存器的内容加到 SEC 和 NANOSEC 寄存器，更新完成后会自动清零，写 1 前需确保此位为 0
2	TSINIT	R/W	0	对此位写 1 将 SECUD 和 NANOSECUD 寄存器的内容复制到 SEC 和 NANOSEC 寄存器，更新完成后会自动清零，写 1 前需确保此位为 0
1	TSFUD	R/W	0	1：时间戳精细更新 0：时间戳粗略更新

位	名称	属性	复位值	说明
0	TSEN	R/W	0	1：使能时间戳，使能后需要初始化 0：关闭时间戳，接收器不处理 IEEE 1588 帧

表 34-3：时间戳快照关系表

SNAPTPYE (17:16 位)	MSTSNAP (15 位)	EVENTSNAP (14 位)	PTP 信息
00	X	0	SYNC, Follow_Up, Delay_Req, Delay_Resp
00	0	1	SYNC
00	1	1	Delay_Req
01	X	0	SYNC, Follow_Up, Delay_Req, Delay_Resp, Pdelay_Req, Pdelay_Resp, Pdelay_Resp_Follow_Up
01	0	1	SYNC, Pdelay_Req, Pdelay_Resp
01	1	1	Delay_Req, Pdelay_Req, Pdelay_Resp
10	X	X	SYNC, Delay_Req
11	X	X	Pdelay_Req, Pdelay_Resp

34.5.2 时间戳亚秒递增寄存器（PTP_SUBSECINC）

偏移地址：0x704

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:8	RSV	-	-	-
7:0	SSINC	R/W	0	亚秒递增值 该位域的设定值会在每个时钟周期（clk_ptp_i）随着亚秒寄存器的值递增。例如，如果 PTP 时钟是 50MHz (周期 20ns), 当系统时间纳秒寄存器精度为 1ns（置位 PTP_TSCTL[9]）时，需要将这些位的值设为 20（0x14）。当 PTP_TSCTL[9]为 0 时，纳秒寄存器分辨精度为约 0.465ns。此时，需要将这些位的值设为 43（0x2B），即 20ns/0.465

34.5.3 时间戳秒数寄存器（PTP_SEC）

偏移地址：0x708

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	SEC	RO	0	显示系统时间的秒数

34.5.4 时间戳纳秒寄存器（PTP_NANOSEC）

偏移地址：0x70C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31	RSV	-	-	-
30:0	NANOSEC	RO	0	显示亚秒时间，精度 0.46ns。PTP_TSCTL[9]置 1 时，精度 1ns，数值不能超过 0x3B9AC9FF

34.5.5 时间戳秒数更新寄存器（PTP_SECUD）

偏移地址：0x710

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	SECUD	R/W	0	设定将要被复制或加到系统时间的秒数

34.5.6 时间戳纳秒更新寄存器（PTP_NANOSECUD）

偏移地址：0x714

复位值：0x0000 0000

位	名称	属性	复位值	说明
31	SIGN	R/W	0	1：更新时间时，时间值减去更新寄存器中的值 0：更新时间时，时间值增加更新寄存器中的值
30:0	NANOSECUD	R/W	0	设定将要被复制或加到系统时间的亚秒数

34.5.7 时间戳加数寄存器（PTP_TSADDEND）

偏移地址：0x718

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	ADDEND	R/W	0	用于同步现实时间的数值，只在时间戳精细更新时有用，其中的数值在每个周期（clk_ptp_ref_i）加到一个 32 位计数器，这个计数器溢出时系统时间会更新

34.5.8 时间目标秒数寄存器（PTP_TGTSEC）

偏移地址：0x71C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	TGTSEC	R/W	0	目标秒数，用于产生中断，中断状态显示在 EMAC_INTSTATUS 寄存器第 9 位

34.5.9 时间目标纳秒寄存器（PTP_TGTNANOSEC）

偏移地址：0x720

复位值：0x0000 0000

位	名称	属性	复位值	说明
31	RSV	-	-	-
31:0	TGTNSEC	R/W	0	目标亚秒数，用于产生中断，中断状态显示在 EMAC_INTSTATUS 寄存器第 9 位。 精度 0.46ns。当 PTP_TSCTL[9]置 1 时，精度 1ns，数值不能超过 0x3B9AC9FF

34.6 DMA 寄存器列表

EMAC 寄存器基地址：0x4010_0000

寄存器列表如下：

表 34-4：DMA 寄存器列表

地址偏移	寄存器名	说明
0x1000	DMA_BUSMODE	总线模式寄存器
0x1004	DMA_TPD	发送寄存器
0x1008	DMA_RPD	接收寄存器
0x100C	DMA_RLA	接收描述符地址寄存器
0x1010	DMA_TLA	发送描述符地址寄存器
0x1014	DMA_STATUS	DMA 状态寄存器
0x1018	DMA_OPMODE	DMA 工作模式寄存器
0x101C	DMA_INTEN	DMA 中断使能寄存器
0x1020	DMA_FMBOCNT	DMA 丢失帧和缓存溢出计数器寄存器
0x1024	DMA_RIWDT	接收中断看门狗定时器寄存器
0x1048	DMA_CTD	DMA 当前发送描述符寄存器
0x104C	DMA_CRD	DMA 当前接收描述符寄存器
0x1050	DMA_CTB	DMA 当前发送缓存地址寄存器
0x1054	DMA_CRB	DMA 当前接收缓存地址寄存器

34.6.1 总线模式寄存器（DMA_BUSMODE）

偏移地址：0x1000

复位值：0x0002 0100

位	名称	属性	复位值	说明
31:26	RSV	-	-	-
25	ALIGN	R/W	0	地址对齐 1：如果 FB 为 1，对齐所有传输；如果 FB 为 0，第一次 burst 传输（起始地址）不对齐，后面的 burst 传输对齐。
24	PBL8	R/W	0	1：将[22:17]和[13:8]中的数值乘 8
23	SEPPBL	R/W	0	1：[22:17]控制接收，[13:8]控制发送 0：[13:8]控制发送和接收

位	名称	属性	复位值	说明
22:17	RXPBL	R/W	1	设定一次 DMA 传输时的最大传输次数，数值只能从 1、2、4、8、16、32 当中选
16	FB	R/W	0	1：传输时使用 SINGLE，INCR4，INCR8，INCR16 0：传输时使用 SINGLE 和 INCR
15:14	PE	R/W	0	指定接收与发送的优先比例： 00：1:1 01：2:1 10：3:1 11：4:1
13:8	TXPBL	R/W	1	设定一次 DMA 传输时的最大传输次数，数值只能从 1、2、4、8、16、32 当中选
7	RSV	-	-	-
6:2	DSL	R/W	0	设定不以链式结构连接的描述符之间的距离，单位为 32 位字。
1	DA	R/W	0	DMA 仲裁模式： 1：DMA 总是优先进行接收传输 0：按照[15:14]中设定的比例循环进行接收和发送的传输
0	RESET	R/W	0	向该位写 1 会复位整个 EMAC，该位自动清零，在清零前不要再次写入。

34.6.2 发送寄存器（DMA_TPD）

偏移地址：0x1004

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	TPD	W	0	对此位写任意值，可命令 DMA 查询 CHTD 寄存器指向的当前描述符，如果不可用则使 DMA 进入暂停状态，并将状态寄存器第 2 位置 1；如果可用则继续传输，可用于唤醒处于暂停状态的传输，发送传输通常会因为数据下溢或 DMA 不能占有需要的描述符而暂停。

34.6.3 接收寄存器（DMA_RPD）

偏移地址：0x1008

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	RPD	W	0	对此位写任意值，可命令 DMA 查询 CHRD 寄存器指向的当前描述符，如果不可用则使 DMA 进入暂停状态，并将 DMA_STATUS[7]置 1；如果可用则继续传输，可用于唤醒处于暂停状态的传输，接收传输通常会因为 DMA 不能占有需要的描述符而暂停。

34.6.4 接收描述符地址寄存器（DMA_RL）

偏移地址：0x100C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	RLA	R/W	0	设定接收队列基地址，对齐到 32 位地址

34.6.5 发送描述符地址寄存器（DMA_TLA）

偏移地址：0x1010

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	TLA	R/W	0	设定发送队列基地址，对齐到 32 位地址

34.6.6 DMA 状态寄存器（DMA_STATUS）

偏移地址：0x1014

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:30	RSV	-	-	-
29	TSTIS	R	0	时间戳中断状态，读 EMAC_INTSTATUS 寄存器清除
28	RSV	-	-	-
27	MMCIS	R	0	MMC 中断状态，在 MMC 相关寄存器中清除
26	RSV	-	-	-
25:23	EB	R	0	指示错误类型，在第 13 位为 1 时才有效： 000：写入接收数据时出错 011：读取发送数据时出错 100：写入接收描述符时出错 101：写入发送描述符时出错 110：读取接收描述符时出错 111：读取发送描述符时出错
22:20	TPS	R	0	指示发送状态： 000：已停止；处于复位后或收到停止命令 001：正在读取发送描述符 010：正在等待发送状态 011：正在将发送数据转移到发送 FIFO 中 100：正在写时间戳 110：已暂停；发送描述符不可用或发送数据下溢 111：正在关闭发送描述符

位	名称	属性	复位值	说明
19:17	RPS	R	0	指示接收状态： 000：已停止；处于复位后或收到停止命令 001：正在读取接收描述符 011：正在等待数据包 100：已暂停；接收描述符不可用 101：正在关闭接收描述符 110：正在写时间戳 111：正在将接收数据转移到外部储存中
16	NIS	R	0	指示第 0、2、6、14 位的中断至少有一个挂起
15	AIS	R	0	指示第 1、3、4、5、7、8、9、10、13 位的中断至少有一个挂起
14	ER	R	0	接收早期中断，指示 DMA 已将第一个接收数据填入缓存；对此位或第 6 位写 1 清零
13	FBE	R	0	总线错误中断，参考[25:23]进行判断；会使 DMA 控制器停止传输；写 1 清零
12:11	RSV	-	-	-
10	ET	R	0	发送早期中断，指示发送数据已经完全位于发送 FIFO 中；写 1 清零
9	RWT	R	0	接收看门狗中断，指示当前接收帧过长；写 1 清零
8	RPSI	R	0	接收停止中断；写 1 清零
7	RBU	R	0	接收缓存不可用中断，指示 DMA 无法占有传输指示符；可以用 RPD 寄存器命令 DMA 尝试恢复传输；下一帧被接收后也自动恢复传输；写 1 清零
6	RI	R	0	接收完成中断，接收仍会运行；写 1 清零
5	TU	R	0	发送数据下溢中断，发送会暂停；写 1 清零
4	RO	R	0	接收数据溢出中断，如果不完整的帧已经发出，则 RDES0 描述符第 11 位会置 1；写 1 清零
3	TJTO	R	0	发送传输 Jabber 定时器超时中断，通常在帧大小过大时发生；会使发送停止并将 TDES0 描述符第 14 位置 1；写 1 清零
2	TBU	R	0	发送缓存不可用中断，指示 DMA 无法占有传输指示符，第[22:20]位会显示当前传输状态；可以设置 TDES0 第 31 位并用 TPD 寄存器命令 DMA 尝试恢复传输；写 1 清零
1	TSI	R	0	发送停止中断；写 1 清零
0	TC	R	0	发送帧完成中断；写 1 清零

34.6.7 DMA 工作模式寄存器（DMA_OPMODE）

偏移地址：0x1018

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:27	RSV	-	-	-
26	DT	R/W	0	1：EMAC 不会丢弃接收到的校验错误的帧 0：如果第 7 位为 0，错误帧会被丢弃

位	名称	属性	复位值	说明
25	RSF	R/W	0	1：DMA 在完全一个接收帧被完全写入 FIFO 后，才发送到外部储存。 0：当接收 FIFO 中的数据超过第[4:3]位设定的阈值后，DMA 将数据发送到外部储存。
24	DFF	R/W	0	1：不会因为接收描述符或缓存不可用就清空接收 FIFO。 0：因为接收描述符或缓存不可用时会清空 FIFO。
23:22	RSV	-	-	-
21	TSF	R/W	0	1：EMAC 会在一个发送帧被完整放入 FIFO 之后才将它发出去。 0：当发送 FIFO 中的数据超过第[16:14]位设定的阈值后，EMAC 会开始发送。
20	FTF	R/W	0	对此位写 1 来命令 EMAC 清空发送 FIFO，在此位自动清零之前，不要再次写入此寄存器。
19:17	RSV	-	-	-
16:14	TT	R/W	0	设定发送阈值： 000：64 001：128 010：192 011：256 100：40 101：32 110：24 111：16
13	ST	R/W	0	1：将此位置 1 后，DMA 开始寻找待发送的帧的描述符，会寻找上次未完成传输的描述符或从 TLA 寄存器的地址寻找；如果没有可用的描述符，会暂停并将 DMASTATUS 的第 2 位置 1；需要在停止的状态下将此位置 1 才有效。 0：发送完当前帧后，会进入停止状态，并使下一描述符会变成当前描述符。
12:8	RSV	-	-	-
7	FEF	R/W	0	1：接收到的帧，除了过短帧之外，都会进入 DMA 0：接收 FIFO 会丢弃错误帧，但是在第 25 位为 0 的情况下，如果帧已经通过 DMA 发送了一部分，就不会丢弃。
6	FUGF	R/W	0	1：DMA 会将长度短于 64 字节的无错误接收帧转发到外部储存。 0：DMA 不会转发过短帧，除非因为帧阈值的设定已经发出了一部份。
5	DGF	R/W	0	1：丢弃接收到的巨大帧 0：不丢弃接收到的巨大帧
4:3	RT	R/W	0	设定接收阈值： 00：64 01：32 10：96 11：128
2	OSF	R/W	0	1：DMA 会提前开始传送下一个发送帧 0：DMA 不会提前开始传送下一个发送帧

位	名称	属性	复位值	说明
1	SR	R/W	0	1：将此位置 1 后，DMA 寻找接收帧的描述符，会从 RLA 寄存器的地址寻找；如果没有可用的描述符，会暂停并将 DMASTATUS 的第 7 位置 1；需要在停止的状态下将此位置 1 才有效。 0：传送完当前正在接收的帧后，会进入停止状态，并使下一描述符会变成当前描述符。
0	RSV	-	-	-

34.6.8 DMA 中断使能寄存器（DMA_INTEN）

偏移地址：0x101C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:17	RSV	-	-	-
16	NIE	R/W	0	1：与第 0、2、6、14 位配合使用，允许它们开启 0：禁止这些中断开启
15	AIE	R/W	0	1：与第 1、3、4、5、7、8、9、10、13 位配合使用，允许它们开启 0：禁止这些中断开启
14	ERE	R/W	0	1：开启接收早期中断 0：关闭中断
13	FBEE	R/W	0	1：开启总线错误中断 0：关闭中断
12:11	RSV	-	-	-
10	ETE	R/W	0	1：开启发送早期中断 0：关闭中断
9	RWTE	R/W	0	1：开启接收看门狗中断 0：关闭中断
8	RPSIE	R/W	0	1：开启接收停止中断 0：关闭中断
7	RBUE	R/W	0	1：开启接收缓存不可用中断 0：关闭中断
6	RIE	R/W	0	1：开启接收完成中断 0：关闭中断
5	TUE	R/W	0	1：开启发送数据下溢中断 0：关闭中断
4	ROE	R/W	0	1：开启接收数据溢出中断 0：关闭中断
3	TJTOE	R/W	0	1：开启发送传输 Jabber 超时中断 0：关闭中断
2	TBUE	R/W	0	1：开启发送缓存不可用中断 0：关闭中断
1	TSIE	R/W	0	1：开启发送停止中断 0：关闭中断
0	TCE	R/W	0	1：开启发送帧完成中断 0：关闭中断

34.6.9 DMA 丢失帧和缓存溢出计数器寄存器（DMA_FMBOCNT）

偏移地址：0x1020

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:29	RSV	-	-	-
28	OVCNTOV	R	0	指示接收 FIFO 溢出计数器溢出，读此寄存器清零
27:17	OVCNT	R	0	记录接收 FIFO 溢出导致丢失帧的数量，在 mci_be_i[2]为 1 时读此寄存器清零
16	MISCNTOV	R	0	指示丢失帧计数器溢出，读此寄存器清零
15:0	MISCNT	R	0	记录 DMA 丢弃的帧的数量，在 mci_be_i[0]为 1 时读此寄存器清零

34.6.10 接收中断看门狗定时器寄存器（DMA_RIWDT）

偏移地址：0x1024

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:8	RSV	-	-	-
7:0	RIWDT	R/W	0	接收 DMA 完成一个传输后，如果 RDES1[31]为 0，则此计数器会随系统时钟的 256 分频时钟减少；减少到 0 时会使 RDES1[31]置位

34.6.11 DMA 当前发送描述符寄存器（DMA_CTD）

偏移地址：0x1048

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	CTD	R	0	指示当前描述符的地址 复位时清零，在 DMA 工作时自动更新

34.6.12 DMA 当前接收描述符寄存器（DMA_CRD）

偏移地址：0x104C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	CRD	R	0	指示当前描述符的地址 复位时清零，在 DMA 工作时自动更新

34.6.13 DMA 当前发送缓存地址寄存器（DMA_CTB）

偏移地址：0x1050

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	CTB	R	0	指示当前 DMA 操作的地址 复位时清零，在 DMA 工作时自动更新

34.6.14 DMA 当前接收缓存地址寄存器（DMA_CRB）

偏移地址：0x1054

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	CRB	R	0	指示当前 DMA 操作的地址 复位时清零，在 DMA 工作时自动更新

34.7 MMC 寄存器列表

EMAC 寄存器基地址：0x4010_0000

寄存器列表如下：

表 34-5：MMC 寄存器列表

地址偏移	寄存器名	说明
0x100	MMC_CONTROL	控制寄存器
0x104	MMC_RI	接收中断寄存器
0x108	MMC_TI	发送中断寄存器
0x10C	MMC_RIM	接收中断屏蔽寄存器
0x110	MMC_TIM	发送中断屏蔽寄存器
0x114	MMC_TXBYTE	已发送字节数，不包括前言和重试的字节
0x118	MMC_TXF	已发送帧数量，不包括重试帧
0x11C	MMC_TXBCGF	已发送正确播放帧数量
0x120	MMC_TXMCGF	已发送正确多播帧数量
0x124	MMC_TXF64	已发送的 64 字节以内的帧数量，不包括前言和重试帧
0x128	MMC_TXF65T127	已发送的 65~127 字节的帧数量，不包括前言和重试帧
0x12C	MMC_TXF128T255	已发送的 128~255 字节的帧数量，不包括前言和重试帧
0x130	MMC_TXF256T511	已发送的 256~511 字节的帧数量，不包括前言和重试帧
0x134	MMC_TXF512T1023	已发送的 512~1023 字节的帧数量，不包括前言和重试帧
0x138	MMC_TXF1024	已发送的多于 1024 字节的帧数量，不包括前言和重试帧
0x13C	MMC_TXUCF	已发送单播帧数量
0x140	MMC_TXMCF	已发送多播帧数量
0x144	MMC_TXBCF	已发送播放帧数量
0x148	MMC_TXUDF	因下溢错误被丢弃的帧数量
0x14C	MMC_TXSGCOL	半双工模式下发生单冲突后成功发送的帧数量
0x150	MMC_TXMTCOL	半双工模式下发生多冲突后成功发送的帧数量
0x154	MMC_TXDEF	半双工模式下发生延迟之后成功发送的帧数量
0x158	MMC_TXLATECOL	因后期冲突丢弃的帧数量

地址偏移	寄存器名	说明
0x15C	MMC_TXEXSCOL	因过多冲突丢弃的帧数量
0x160	MMC_TXCARERR	因载波错误丢弃的帧数量
0x164	MMC_TXGBYTE	已发送正确帧里除前言外的字节数
0x168	MMC_TXGF	已发送正确帧数量
0x16C	MMC_TXEXSDEF	因过长延迟被丢弃的帧数量
0x170	MMC_TXPAUSEF	已发送正确暂停帧数量
0x174	MMC_TXVLANGF	已发送正确 VLAN 帧数量
0x178	MMC_TXOVSIZEGF	已发送无错误过长帧数量
0x180	MMC_RXF	已接收帧数量
0x184	MMC_RXBYTE	已接收字节数，不包括前言
0x188	MMC_RXGBYTE	已接收正确帧字节数，不包括前言
0x18C	MMC_RXBCGF	已接收正确播放帧数量
0x190	MMC_RXMCGF	已接收正确多播帧数量
0x194	MMC_RXCRCERR	已接收 CRC 错误帧数量
0x198	MMC_RXALGERR	已接收对齐错误帧，仅在 10/100 模式下出现
0x19C	MMC_RXRUNTERR	已接收过短且 CRC 错误帧数量
0x1A0	MMC_RXJABERR	已接收巨大且 CRC 错误帧数量
0x1A4	MMC_RXUDSIZEGF	已接收过短正确帧数量
0x1A8	MMC_RXOVSIZEGF	已接收过大正确帧数量
0x1AC	MMC_RXF64	已接收的 64 字节以内的帧数量，不包括前言
0x1B0	MMC_RXF65T127	已接收的 65~127 字节的帧数量，不包括前言
0x1B4	MMC_RXF128T255	已接收的 128~255 字节的帧数量，不包括前言
0x1B8	MMC_RXF256T511	已接收的 256~511 字节的帧数量，不包括前言
0x1BC	MMC_RXF512T1023	已接收的 512~1023 字节的帧数量，不包括前言
0x1C0	MMC_RXF1024	已接收的多于 1024 字节的帧数量，不包括前言
0x1C4	MMC_RXUCGF	已接收单播正确帧数量
0x1C8	MMC_RXLERR	已接收长度错误帧数量
0x1CC	MMC_OUTRANGE	已接收长度非法错误帧数量
0x1D0	MMC_RXPAUSEF	已接收暂停帧
0x1D4	MMC_RXFIFOOVF	因接收 FIFO 溢出丢失的帧数量
0x1D8	MMC_RXVLANF	已接收 VLAN 帧数量
0x1DC	MMC_RXWDGERR	已接收看门狗超时错误帧数量
0x1E0	MMC_RXRCVERR	接收错误和扩展错误帧数量
0x1E4	MMC_RXCTRLGF	已接收正确控制帧数量
0x200	MMC_IPCINTRMASK	接收校验中断屏蔽寄存器
0x208	MMC_IPCINTR	接收校验中断寄存器
0x210	MMC_RXIPV4F	已接收正确 IPV4 数据帧数量
0x214	MMC_RXIPV4HDRERR	已接收 IPV4 报头错误数据帧数量 （校验、长度、版本错误）
0x218	MMC_RXIPV4NPF	已接收 IPV4 无报文帧数量
0x21C	MMC_RXIPV4FRAG	已接收 IPV4 分散数据帧数量
0x220	MMC_RXIPV4UDSBL	已接收 IPV4 无校验 UDP 帧数量
0x224	MMC_RXIPV6F	已接收正确 IPV6 数据帧数量
0x228	MMC_RXIPV6HDRERR	已接收 IPV6 报头错误数据帧数量（长度、版本错误）
0x22C	MMC_RXIPV6NPF	已接收 IPV6 无报文帧数量，包括所有 IPV6 有分散数据和安全拓展的报头的帧
0x230	MMC_RXUDPF	已接收 UDP 帧数量，当 RXIPV4UDSBL 增加时这个不会增加

地址偏移	寄存器名	说明
0x234	MMC_RXUDPERRF	已接收有校验错误的 UDP 帧数量
0x238	MMC_RXTCPF	已接收 TCP 帧数量
0x23C	MMC_RXTCPERRF	已接收有检验错误的 TCP 帧数量
0x240	MMC_RXICMPF	已接收 ICMP 帧数量
0x244	MMC_RXICMPERRF	已接收有检验错误的 ICMP 帧数量
0x250	MMC_RXIPV4BYTE	已接收 IPV4 正确数据帧字节数
0x254	MMC_RXIPV4HDRERB	已接收 IPV4 报头错误帧字节数
0x258	MMC_RXIPV4NPB	已接收 IPV4 无报文帧字节数
0x25C	MMC_RXIPV4FRAGB	已接收 IPV4 分散数据帧字节数
0x260	MMC_RXIPV4UDSBLB	已接收 IPV4 无校验 UDP 帧字节数
0x264	MMC_RXIPV6BYTE	已接收 IPV6 正确数据帧字节数
0x268	MMC_RXIPV6HDRERB	已接收 IPV6 报头错误帧字节数
0x26C	MMC_RXIPV6NPB	已接收 IPV6 无报文帧字节数
0x270	MMC_RXUDPBYTE	已接收正确 UDP 帧字节数
0x274	MMC_RXUDPERRB	已接收 UDP 校验错误帧字节数
0x278	MMC_RXTCPBYTE	已接收正确 TCP 帧字节数
0x27C	MMC_RXTCPERRB	已接收 TCP 校验错误帧字节数
0x280	MMC_RXICMPBYTE	已接收正确 ICMP 帧字节数
0x284	MMC_RXICMPERRB	已接收 ICMP 校验错误帧字节数

34.7.1 MMC 控制寄存器（MMC_CONTROL）

偏移地址：0x100

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:9	RSV	-	-	-
8	UCDBC	R/W	0	1：被过滤掉的广播帧会更新 MMC 计数器 0：被过滤的帧不会使 MMC 计数器更新
7:6	RSV	-	-	-
5	PRSTVL	R/W	0	1：预设值为计数器满 0：预设值为计数器半满
4	PRST	W	0	写 1 使计数器的数值变为预设值
3	CNTFREEZ	R/W	0	1：计数器数值不会因为发送和接收帧而更新 0：计数器数值会更新
2	RSTONRD	R/W	0	1：读取计数器会使数值清零 0：读取不会使数值清零
1	NRO	R/W	0	1：计数值达到满时不会回到 0 0：计数值达到满时会回到 0
0	RST	W	0	写 1 使计数器复位

34.7.2 MMC 接收中断寄存器（MMC_RI）

偏移地址：0x104

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:26	RSV	-	-	-
25	RXCTRFIS	R	0	接收控制帧计数器 RXCTRLGF 达到半满或满时触发，读清除
24	RXERRFIS	R	0	接收错误帧计数器 RXRCVERR 达到半满或满时触发，读清除
23	RXWDERRFIS	R	0	接收超时错误帧计数器 RXWDGERR 达到半满或满时触发，读清除
22	RXVLANIS	R	0	接收 VLAN 帧计数器 RXVLANF 达到半满或满时触发，读清除
21	RXFOVFIS	R	0	接收 FIFO 溢出计数器 RXFIFOOVF 达到半满或满时触发，读清除
20	RXPAUSFIS	R	0	接收暂停帧计数器 RXPAUSEF 达到半满或满时触发，读清除
19	RXORANGFIS	R	0	接收地址超界计数器 OUTRANGE 达到半满或满时触发，读清除
18	RXLERRFIS	R	0	接收帧长度错误计数器 RXLERR 达到半满或满时触发，读清除
17	RXUCGFIS	R	0	接收单播正确帧计数器 RXUCGF 达到半满或满时触发，读清除
16	RX1024TMAXFIS	R	0	接收帧长度超过 1024 计数器 RXF1024 达到半满或满时触发，读清除
15	RX512T1023FIS	R	0	接收帧长度 512 至 1023 计数器 RXF512T1023 达到半满或满时触发，读清除
14	RX256T511FIS	R	0	接收帧长度 256 至 511 计数器 RXF256T511 达到半满或满时触发，读清除
13	RX128T255FIS	R	0	接收帧长度 128 至 255 计数器 RXF128T255 达到半满或满时触发，读清除
12	RX65T127FIS	R	0	接收帧长度 65 至 127 计数器 RXF65T127 达到半满或满时触发，读清除
11	RX64FIS	R	0	接收帧长度低于 64 计数器 RXF64 达到半满或满时触发，读清除
10	RXOSIZEGFIS	R	0	接收过大正确帧计数器 RXOVSIZEGF 达到半满或满时触发，读清除
9	RXUSIZEGFIS	R	0	接收过小正确帧计数器 RXUDSIZEGF 达到半满或满时触发，读清除
8	RXJABERFIS	R	0	接收 Jabber 错误帧计数器 RXJABERR 达到半满或满时触发，读清除
7	RXRUNTFIS	R	0	接收过短错误帧计数器 RXRUNTERR 达到半满或满时触发，读清除
6	RXALIGNFIS	R	0	接收对齐错误帧计数器 RXALGERR 达到半满或满时触发，读清除
5	RXCRCERFIS	R	0	接收 CRC 错误帧计数器 RXCRCERR 达到半满或满时触发，读清除
4	RXMCGFIS	R	0	接收多播正确帧计数器 RXMCGF 达到半满或满时触发，读清除
3	RXBCGFIS	R	0	接收广播正确帧计数器 RXBCGF 达到半满或满时触发，读清除

位	名称	属性	复位值	说明
2	RXGBOIS	R	0	接收正确数据计数器 RXGBYTE 达到半满或满时触发，读清除
1	RXOIS	R	0	接收数据计数器 RXBYTE 达到半满或满时触发，读清除
0	RXFIS	R	0	接收帧计数器 RXF 达到半满或满时触发，读清除

34.7.3 MMC 发送中断寄存器（MMC_TI）

偏移地址：0x108

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:26	RSV	-	-	-
25	TXOSIZEGFIS	R	0	发送过大正确帧计数器 TXOVSIZEGF 达到半满或满时触发，读清除
24	TXVLANGFIS	R	0	发送 VLAN 正确帧计数器 TXVLANGF 达到半满或满时触发，读清除
23	TXPAUSFIS	R	0	发送暂停帧计数器 TXPAUSEF 达到半满或满时触发，读清除
22	TXEXDEFFIS	R	0	发送延迟过长帧计数器 TXEXSDEF 达到半满或满时触发，读清除
21	TXGFIS	R	0	发送正确帧计数器 TXGF 达到半满或满时触发，读清除
20	TXGOIS	R	0	发送正确数据计数器 TXGBYTE 达到半满或满时触发，读清除
19	TXCARERFIS	R	0	发送载波错误帧计数器 TXCARERR 达到半满或满时触发，读清除
18	TXEXCOLFIS	R	0	发送过度冲突帧计数器 TXEXSCOL 达到半满或满时触发，读清除
17	TXLACOLFIS	R	0	发送后期冲突帧计数器 TXLATECOL 达到半满或满时触发，读清除
16	TXDEFFIS	R	0	发送延迟帧计数器 TXDEF 达到半满或满时触发，读清除
15	TXMCOLGFIS	R	0	发送多冲突正确帧计数器 TXMTCOL 达到半满或满时触发，读清除
14	TXSCOLGFIS	R	0	发送单个冲突正确帧计数器 TXSGCOL 达到半满或满时触发，读清除
13	TXUFERFIS	R	0	发送下溢错误帧计数器 TXUDF 达到半满或满时触发，读清除
12	TXBCFIS	R	0	发送广播帧计数器 TXBCF 达到半满或满时触发，读清除
11	TXMCFIS	R	0	发送多播帧计数器 TXMCF 达到半满或满时触发，读清除
10	TXUCFIS	R	0	发送单播帧计数器 TXUCF 达到半满或满时触发，读清除
9	TX1024TMAXFIS	R	0	发送帧长度超过 1024 计数器 TXF1024 达到半满或满时触发，读清除

位	名称	属性	复位值	说明
8	TX512T1023FIS	R	0	发送帧长度 512 至 1023 计数器 TXF512T123 达到半满或满时触发，读清除
7	TX256T511FIS	R	0	发送帧长度 256 至 511 计数器 TXF256T511 达到半满或满时触发，读清除
6	TX128T255FIS	R	0	发送帧长度 128 至 255 计数器 TXF128T255 达到半满或满时触发，读清除
5	TX65T127FIS	R	0	发送帧长度 65 至 127 计数器 TXF65T127 达到半满或满时触发，读清除
4	TX64FIS	R	0	发送帧长度低于 64 计数器 TXF64 达到半满或满时触发，读清除
3	TXMCGFIS	R	0	发送多播正确帧计数器 TXMCGF 达到半满或满时触发，读清除
2	TXBCGFIS	R	0	发送广播正确帧计数器 TXBCGF 达到半满或满时触发，读清除
1	TXFIS	R	0	发送帧计数器 TXF 达到半满或满时触发，读清除
0	TXOIS	R	0	发送数据计数器 TXBYTE 达到半满或满时触发，读清除

34.7.4 MMC 接收中断屏蔽寄存器（MMC_RIM）

偏移地址：0x10C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:26	RSV	-	-	-
25	RXCTRFIM	R/W	0	接收控制帧计数器 RXCTRLGF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
24	RXERRFIM	R/W	0	接收错误帧计数器 RXRCVERR 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
23	RXWDERRFIM	R/W	0	接收超时错误帧计数器 RXWDGERR 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
22	RXVLANIM	R/W	0	接收 VLAN 帧计数器 RXVLANF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
21	RXFOVFIM	R/W	0	接收 FIFO 溢出计数器 RXFIFOOVF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
20	RXPAUSFIM	R/W	0	接收暂停帧计数器 RXPAUSEF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发

位	名称	属性	复位值	说明
19	RXORANGFIM	R/W	0	接收地址超界计数器 OUTRANGE 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
18	RXLERRFIM	R/W	0	接收帧长度错误计数器 RXLERR 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
17	RXUCGFIM	R/W	0	接收单播正确帧计数器 RXUCGF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
16	RX1024TMAXFIM	R/W	0	接收帧长度超过 1024 计数器 RXF1024 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
15	RX512T1023FIM	R/W	0	接收帧长度 512 至 1023 计数器 RXF512T1023 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
14	RX256T511FIM	R/W	0	接收帧长度 256 至 511 计数器 RXF256T511 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
13	RX128T255FIM	R/W	0	接收帧长度 128 至 255 计数器 RXF128T255 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
12	RX65T127FIM	R/W	0	接收帧长度 65 至 127 计数器 RXF65T127 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
11	RX64FIM	R/W	0	接收帧长度低于 64 计数器 RXF64 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
10	RXOSIZEGFIM	R/W	0	接收过大正确帧计数器 RXOVSIZEGF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
9	RXUSIZEGFIM	R/W	0	接收过小正确帧计数器 RXUDSIZEGF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
8	RXJABERFIM	R/W	0	接收 Jabber 错误帧计数器 RXJABERR 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发

位	名称	属性	复位值	说明
7	RXRUNTFIM	R/W	0	接收 Runt 错误帧计数器 RXRUNTERR 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
6	RXALIGNFIM	R/W	0	接收对齐错误帧计数器 RXALGERR 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
5	RXCRCERFIM	R/W	0	接收 CRC 错误帧计数器 RXCRCERR 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
4	RXMCGFIM	R/W	0	接收多播正确帧计数器 RXMCGF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
3	RXBCGFIM	R/W	0	接收广播正确帧计数器 RXBCGF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
2	RXGBOIM	R/W	0	接收正确数据计数器 RXGBYTE 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
1	RXOIM	R/W	0	接收数据计数器 RXBYTE 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
0	RXFIM	R/W	0	接收帧计数器 RXF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发

34.7.5 MMC 发送中断屏蔽寄存器（MMC_TIM）

偏移地址：0x110

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:26	RSV	-	-	-
25	TXOSIZEGFIM	R/W	0	发送过大正确帧计数器 TXOVSIZEGF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
24	TXVLANGFIM	R/W	0	发送 VLAN 正确帧计数器 TXVLANGF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发

位	名称	属性	复位值	说明
23	TXPAUSFIM	R/W	0	发送暂停帧计数器 TXPAUSEF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
22	TXEXDEFFIM	R/W	0	发送延迟过长帧计数器 TXEXSDEF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
21	TXGFIM	R/W	0	发送正确帧计数器 TXGF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
20	TXGOIM	R/W	0	发送正确数据计数器 TXGBYTE 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
19	TXCARERFIM	R/W	0	发送载波错误帧计数器 TXCARERR 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
18	TXEXCOLFIM	R/W	0	发送过度冲突帧计数器 TXEXSCOL 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
17	TXLACOLFIM	R/W	0	发送后期冲突帧计数器 TXLATECOL 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
16	TXDEFFIM	R/W	0	发送延迟帧计数器 TXDEF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
15	TXMCOLGFIM	R/W	0	发送多冲突正确帧计数器 TXMTCOL 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
14	TXSCOLGFIM	R/W	0	发送单个冲突正确帧计数器 TXSGCOL 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
13	TXUFERFIM	R/W	0	发送下溢错误帧计数器 TXUDF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
12	TXBCFIM	R/W	0	发送广播帧计数器 TXBCF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
11	TXMCFIM	R/W	0	发送多播帧计数器 TXMCF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发

位	名称	属性	复位值	说明
10	TXUCFIM	R/W	0	发送单播帧计数器 TXUCF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
9	TX1024TMAXFIM	R/W	0	发送帧长度超过 1024 计数器 TXF1024 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
8	TX512T1023FIM	R/W	0	发送帧长度 512 至 1023 计数器 TXF512T1023 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
7	TX256T511FIM	R/W	0	发送帧长度 256 至 511 计数器 TXF256T511 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
6	TX128T255FIM	R/W	0	发送帧长度 128 至 255 计数器 TXF128T255 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
5	TX65T127FIM	R/W	0	发送帧长度 65 至 127 计数器 TXF65T127 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
4	TX64FIM	R/W	0	发送帧长度低于 64 计数器 TXF64 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
3	TXMCGFIM	R/W	0	发送多播正确帧计数器 TXMCGF 中断屏蔽设置位 1：中断不会触发 0：中断可以触发
2	TXBCGFIM	R/W	0	发送广播正确帧计数器 TXBCGF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
1	TXFIM	R/W	0	发送帧计数器 TXF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
0	TXOIM	R/W	0	发送数据计数器 TXBYTE 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发

34.7.6 MMC 接收校验中断屏蔽寄存器（MMC_IPCINTRMASK）

偏移地址：0x200

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:30	RSV	-	-	-

位	名称	属性	复位值	说明
29	RXICMPEROIM	R/W	0	已接收 ICMP 校验错误帧字节数计数器 RXICMPERRB 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
28	RXICMPGOIM	R/W	0	已接收正确 ICMP 帧字节数计数器 RXICMPBYTE 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
27	RXTCPEROIM	R/W	0	已接收 TCP 校验错误帧字节数计数器 RXTCPERRB 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
26	RXTCPGOIM	R/W	0	已接收正确 TCP 帧字节数计数器 RXTCPBYTE 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
25	RXUDPEROIM	R/W	0	已接收 UDP 校验错误帧字节数计数器 RXUDPERRB 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
24	RXUDPGOIM	R/W	0	已接收正确 UDP 帧字节数计数器 RXUDPBYTE 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
23	RXIPV6NPOIM	R/W	0	已接收 IPV6 无报文帧字节数计数器 RXIPV6NPB 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
22	RXIPV6HEROIM	R/W	0	已接收 IPV6 报头错误帧字节数计数器 RXIPV6HDRERB 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
21	RXIPV6GOIM	R/W	0	已接收 IPV6 正确数据帧字节数计数器 RXIPV6BYTE 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
20	RXIPV4UDSBOIM	R/W	0	已接收 IPV4 无校验 UDP 帧字节数计数器 RXIPV4UDSBL 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
19	RXIPV4FRAOIM	R/W	0	已接收 IPV4 分散数据帧字节数计数器 RXIPV4FRAGB 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
18	RXIPV4NPOIM	R/W	0	已接收 IPV4 无报文帧字节数计数器 RXIPV4NPB 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发

位	名称	属性	复位值	说明
17	RXIPV4HEROIM	R/W	0	已接收 IPV4 报头错误帧字节数计数器 RXIPV4HDRERB 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
16	RXIPV4GOIM	R/W	0	已接收 IPV4 正确数据帧字节数计数器 RXIPV4BYTE 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
15:14	RSV	-	-	-
13	RXICMPERFIM	R/W	0	已接收有校验错误 ICMP 帧计数器 RXICMPERRF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
12	RXICMPGFIM	R/W	0	已接收 ICMP 帧计数器 RXICMPF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
11	RXTCPERFIM	R/W	0	已接收有校验错误 TCP 帧计数器 RXTCPERRF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
10	RXTCPGFIM	R/W	0	已接收 TCP 帧计数器 RXTCPF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
9	RXUDPERFIM	R/W	0	已接收有校验错误 UDP 帧计数器 RXUDPERRF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
8	RXUDPGFIM	R/W	0	已接收 UDP 帧计数器 RXUDPF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
7	RXIPV6NPFIM	R/W	0	已接收 IPV6 无报文帧计数器 RXIPV6NPF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
6	RXIPV6HERFIM	R/W	0	已接收 IPV6 报头错误数据帧计数器 RXIPV6HDRERR 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
5	RXIPV6GFIM	R/W	0	已接收 IPV6 正确数据帧计数器 RXIPV6F 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
4	RXIPV4UDSBFIM	R/W	0	已接收 IPV4 无校验 UDP 帧计数器 RXIPV4UDSBL 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发

位	名称	属性	复位值	说明
3	RXIPV4FRAFIM	R/W	0	已接收 IPV4 分散数据帧计数器 RXIPV4FRAG 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
2	RXIPV4NPFIM	R/W	0	已接收 IPV4 无报文帧计数器 RXIPV4NPF 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
1	RXIPV4HERFIM	R/W	0	已接收 IPV4 报头错误数据帧计数器 RXIPV4HDRERR 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发
0	RXIPV4GFIM	R/W	0	已接收 IPV4 正确数据帧计数器 RXIPV4F 中断屏蔽设置位： 1：中断不会触发 0：中断可以触发

34.7.7 MMC 接收校验中断寄存器（MMC_IPCINTR）

偏移地址：0x208

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:30	RSV	-	-	-
29	RXICMPEROIS	R	0	已接收 ICMP 校验错误帧字节数计数器 RXICMPERRB 达到半满或满时触发，读清除
28	RXICMPGOIS	R	0	已接收正确 ICMP 帧字节数计数器 RXICMPBYTE 达到半满或满时触发，读清除
27	RXTCPEROIS	R	0	已接收 TCP 校验错误帧字节数计数器 RXTCPERRB 达到半满或满时触发，读清除
26	RXTCPGOIS	R	0	已接收正确 TCP 帧字节数计数器 RXTCPBYTE 达到半满或满时触发，读清除
25	RXUDPEROIS	R	0	已接收 UDP 校验错误帧字节数计数器 RXUDPERRB 达到半满或满时触发，读清除
24	RXUDPGOIS	R	0	已接收正确 UDP 帧字节数计数器 RXUDPBYTE 达到半满或满时触发，读清除
23	RXIPV6NPOIS	R	0	已接收 IPV6 无报文帧字节数计数器 RXIPV6NPB 达到半满或满时触发，读清除
22	RXIPV6HEROIS	R	0	已接收 IPV6 报头错误帧字节数计数器 RXIPV6HDRERB 达到半满或满时触发，读清除
21	RXIPV6GOIS	R	0	已接收 IPV6 正确数据帧字节数计数器 RXIPV6BYTE 达到半满或满时触发，读清除
20	RXIPV4UDSBOIS	R	0	已接收 IPV4 无校验 UDP 帧字节数计数器 RXIPV4UDSBL 达到半满或满时触发，读清除
19	RXIPV4FRAOIS	R	0	已接收 IPV4 分散数据帧字节数计数器 RXIPV4FRAGB 达到半满或满时触发，读清除
18	RXIPV4NPOIS	R	0	已接收 IPV4 无报文帧字节数计数器 RXIPV4NPB 达到半满或满时触发，读清除

位	名称	属性	复位值	说明
17	RXIPV4HEROIS	R	0	已接收 IPV4 报头错误帧字节数计数器 RXIPV4HDRERB 达到半满或满时触发，读清除
16	RXIPV4GOIS	R	0	已接收 IPV4 正确数据帧字节数计数器 RXIPV4BYTE 达到半满或满时触发，读清除
15:14	RSV	-	-	-
13	RXICMPERFIS	R	0	已接收有校验错误 ICMP 帧计数器 RXICMPERRF 达到半满或满时触发，读清除
12	RXICMPGFIS	R	0	已接收 ICMP 帧计数器 RXICMPF 达到半满或满时触发，读清除
11	RXTCPERFIS	R	0	已接收有校验错误 TCP 帧计数器 RXTCPERRF 达到半满或满时触发，读清除
10	RXTCPGFIS	R	0	已接收 TCP 帧计数器 RXTCPF 达到半满或满时触发，读清除
9	RXUDPERFIS	R	0	已接收有校验错误 UDP 帧计数器 RXUDPERRF 达到半满或满时触发，读清除
8	RXUDPGFIS	R	0	已接收 UDP 帧计数器 RXUDPF 达到半满或满时触发，读清除
7	RXIPV6NPFIS	R	0	已接收 IPV6 无报文帧计数器 RXIPV6NPF 达到半满或满时触发，读清除
6	RXIPV6HERFIS	R	0	已接收 IPV6 报头错误数据帧计数器 RXIPV6HDRERR 达到半满或满时触发，读清除
5	RXIPV6GFIS	R	0	已接收 IPV6 正确数据帧计数器 RXIPV6F 达到半满或满时触发，读清除
4	RXIPV4UDSBFIS	R	0	已接收 IPV4 无校验 UDP 帧计数器 RXIPV4UDSBL 达到半满或满时触发，读清除
3	RXIPV4FRAFIS	R	0	已接收 IPV4 分散数据帧计数器 RXIPV4FRAG 达到半满或满时触发，读清除
2	RXIPV4NPFIS	R	0	已接收 IPV4 无报文帧计数器 RXIPV4NPF 达到半满或满时触发，读清除
1	RXIPV4HERFIS	R	0	已接收 IPV4 报头错误数据帧计数器 RXIPV4HDRERR 达到半满或满时触发，读清除
0	RXIPV4GFIS	R	0	已接收 IPV4 正确数据帧计数器 RXIPV4F 达到半满或满时触发，读清除

34.8 DMA 描述符

34.8.1 发送描述符 0（TDES0）

位	名称	说明
31	OWN	1：指示描述符被 DMA 占有 0：指示描述符被 CPU 占有 当帧传输结束或描述符指向的缓存为空时，DMA 会清空此位； 此位应当在该帧的描述符其余部分完全配置完成后再设置。
30:18	RSV	-
17	TTS	表示时间戳已被截取并置于 TDES2 和 TDES3 中，仅当 TDES1 第 30 位为 1 时生效

位	名称	说明
16	IPHE	指示 IP 报头错误，仅当校验开启时生效；如果使用了 IPV4 内容，则依然会插入校验码
15	ES	指示第 16、14、13、12、11、10、9、8、2、1 位中至少有 1 个错误出现
14	JTO	指示 Jabber 计时器超时
13	FF	指示 CPU 命令了 DMA 把 FIFO 中的帧数据清空
12	PLE	指示校验错误，通常由错误的数据长度或帧长度大于 FIFO 引起，此时不会插入校验码，仅当校验开启时生效
11	LC	指示载波丢失错误，仅在半双工模式下且发送帧没有冲突时生效
10	NC	指示无载波
9	LCOL	指示在冲突窗口期过后出现了冲突
8	ECOL	指示冲突过多，出现超过 16 次冲突引起，如果关闭了重试，冲突出现 1 次就会引起；会使帧传输被放弃
7	VF	指示发送的是 VLAN 帧
6:3	COLCNT	记录发生冲突测次数，TDES0 第 8 位为 0 时有效
2	ED	指示顺延过多，超过 24288 位（1000Mbps 或巨大帧模式下 155680 位）时间，会使传输被结束，EMAC Config 寄存器第 4 位为 1 时有效
1	UF	指示发送 FIFO 数据下溢，会使发送暂停
0	DB	指示发生了顺延，半双工模式下由载波缺失引起

34.8.2 发送描述符 1（TDES1）

位	名称	说明
31	IC	1：使 DMASTATUS 寄存器第 0 位的中断在此帧发送完成后置位，仅当第 30 位为 1 时有效
30	LS	1：指示缓存中储存了当前帧的最后一段；需要 TBS1 或 TBS2 为非零值
29	FS	1：指示缓存中储存了当前帧的第一段
28:27	CIC	设置插入的校验码 00：不插入校验码 01：插入 IPV4 报头校验码 10：插入 TCP/UDP/ICMP 校验码，不包含伪报头 11：插入 TCP/UDP/ICMP 校验码，包含伪报头 仅当第 29 位为 1 时有效
26	DC	1：不在帧末尾插入 CRC 仅当第 29 位为 1 时有效
25	TER	1：指示这是描述符列表中的最后一个，之后会返回到最初的描述符地址
24	TCH	1：表示描述符的第二个地址是下一个描述符而不是第二个缓存的地址，仅当第 25 位为 0 时有效
23	DP	1：对于不到 64 字节的帧，不会填充 0：自动填充短于 64 字节的帧 仅当第 29 位为 1 时有效
22	TTSE	1：打开 IEEE1588 硬件时间戳功能，仅当第 29 位为 1 时有效
22:11	TBS2	设置第二个数据缓存的大小，仅当第 24 位为 0 时有效
10:0	TBS1	设置第一个数据缓存的大小

34.8.3 发送描述符 2（TDES2）

位	名称	说明
31:0	B1A	设置第一个缓存的地址或时间戳低位

34.8.4 发送描述符 3（TDES3）

位	名称	说明
31:0	B2A	设置第二个缓存的地址（或下一个描述符的地址）或时间戳高位

34.8.5 接收描述符 0（RDES0）

位	名称	说明
31	OWN	1：指示描述符被 DMA 占有 0：指示描述符被 CPU 占有 当帧传输结束或描述符指向的缓存为满时，DMA 会清空此位
30	DAFE	指示有一个帧未通过目的地址过滤器
29:16	FL	指示帧的长度（字节），包含 CRC 在内，仅当第 8 位为 1 且第 14 位为 0 时有效； 第 8 位为 0 时，指示当前帧已传输的字节数
15	ES	指示 14、11、7、6、4、3、1、0 位中至少有 1 个错误出现
14	DE	指示当前因为描述符指示的缓存无法装下当前帧，而 DMA 又不占有下一个描述符，导致当前帧被切断，仅当第 8 位为 1 时有效
13	SAFE	指示有一个帧未通过源地址过滤器
12	LE	指示帧实际长度与长度/类型域中指示的不符
11	OVF	指示发生了 FIFO 溢出
10	VF	指示该帧为 VLAN 帧
9	FS	指示该描述符为该帧的第一个描述符，该描述符的缓存用于装载帧的最初部分
8	LS	指示该描述符为该帧的最后一个描述符，该描述符的缓存用于装载帧的最终部分
7	IPCSE	指示发生了报头校验错误
6	LC	指示在半双工模式下发生了晚冲突
5	FT	1：指示该帧为以太网帧 0：指示该帧为 IEEE802.3 帧
4	RWDTO	指示发生了接收看门狗超时
3	RE	指示在接收过程中 RXDV 信号有效时出现了 RXER 信号
2	DBE	指示接收到的数据长度不是字节的整数倍，仅在 MII 模式下有效
1	CRCE	指示发生了 CRC 校验错误，仅当第 8 位为 1 时有效
0	PLCSE	指示发生了数据校验错误

RDES0 中的第 0、5、7 位表示的接收帧的类型与状态如下表所示：

Bit5:FT	Bit7:IPCSE	Bit0:PLCSE	帧类型与状态
0	0	0	IEEE802.3 帧，长度小于 1536

Bit5:FT	Bit7:IPCSE	Bit0:PLCSE	帧类型与状态
1	0	0	EMAC Config 第 10 位为 1 时： IPv4/IPv6 帧，无校验错误 EMAC Config 第 10 位为 0 时： IEEE802.3 帧，长度大于或等于 1536
1	0	1	IPv4/IPv6 帧，数据校验错误
1	1	0	IPv4/IPv6 帧，报头校验错误
1	1	1	IPv4/IPv6 帧，数据和报头校验错误
0	0	1	IPv4/IPv6 帧，数据校验因格式不符被跳过
0	1	1	帧类型不是 IPv4/IPv6，未进行校验

34.8.6 接收描述符 1 (RDES1)

位	名称	说明
31	DIC	1：关闭接收中断，仅当 RDES0 第 8 位为 1 时有效
30:26	RSV	保留
25	RER	1：指示这是描述符列表中的最后一个，之后会返回到最初的描述符地址
24	RCH	1：表示描述符的第二个地址是下一个描述符而不是第二个缓存的地址，仅当第 25 位为 0 时有效
23:22	RSV	保留
22:11	RBS2	设置第二个数据缓存的大小，仅当第 24 位为 0 时有效，缓存大小应当为 4 的倍数
10:0	RBS1	设置第一个数据缓存的大小，缓存大小应当为 4 的倍数

34.8.7 接收描述符 2 (RDES2)

位	名称	说明
31:0	B1A	设置第一个缓存的地址 或时间戳低位

34.8.8 接收描述符 3 (RDES3)

位	名称	说明
31:0	B2A	设置第二个缓存的地址（或下一个描述符的地址） 或时间戳高位

34.9 使用流程

34.9.1 初始化 DMA

1. 操作 EMAC_CONFIG 寄存器，对模块使用软件复位。
2. 查询 DMA_BUSMODE 寄存器，直到复位操作完成。
3. 配置 DMA_BUSMODE 寄存器中需要使用的位。
4. 在储存器中创建传输描述符，确保接收描述符由 EMAC-DMA 占有，在重复使用同一个描述符

之前，最好有多个不同的描述符。

5. 在 DMA_TPD 和 DMA_RPD 寄存器中初始化描述符地址。
6. 配置 DMA_OPMODE 寄存器。
7. 配置 DMA_INTEN 寄存器开启中断。
8. 使用 DMA_OPMODE 寄存器开始传输。

34.9.2 初始化 EMAC

1. 配置 PHY。
2. 配置 EMAC_ADDR0H 寄存器及 EMAC_ADDR0L 寄存器，如果需要使用更多地址，配置其余地址寄存器。
3. 配置 EMAC_HASHTABLEHIGH 寄存器、EMAC_HASHTABLELOW 寄存器和 EMAC_FRAMEFILTER 寄存器设定过滤方式。
4. 配置 EMAC_FIOWCONTROL 寄存器。
5. 配置 EMAC_INTMASK 寄存器，选择需要的中断。
6. 配置 EMAC_CONFIG 寄存器并启动传输。

34.9.3 进行传输

1. 设定中断处理程序：检查中断状态以及描述符状态。
2. 设定合适的描述符，确保描述符为 DMA 占有。
3. 如果描述符并非 DMA 占有，会使 DMA 进入暂停模式，可以使用 TPD 和 RPD 寄存器使传输开始。

34.9.4 停止和开始传输

如果需要暂停传输一段时间，按以下步骤操作：

1. 清除 DMA_OPMODE[13]以停用 DMA 发送传输。
2. 等待前一帧发送传输完成，可以查询 Debug 寄存器以检查状态。
3. 清除 EMAC_CONFIG[3:2]。
4. 查询 EMAC_DEBUG 寄存器，在接收 FIFO 中的数据完全传输到系统储存之后，关闭接收 DMA。
5. 要重新开始传输时，先启动 DMA，再启动 EMAC 传输。

34.9.5 EEE 特性的使用指引

进入和退出 LPI 模式：

1. 检查 PHY 是否支持 EEE 特性。
2. 配置 PHY 使用 EEE 功能。
3. 配置 EMAC_LPIT 寄存器。
4. 读取 PHY 状态，更新 EMAC_LPICS[17]。
5. 配置 EMAC_LPICS[16]使 EMAC 进入 LPI 模式。
6. 可以清除 EMAC_LPICS[16]以退出 LPI 模式。

35 USB FS Device 控制器（USB）

35.1 概述

USB设备控制器是一个兼容 USB2.0全速协议设备接口，其与USB PHY配合使用，提供芯片与USB HOST通讯功能。

35.2 主要特性

- 兼容 USB1.1 和 USB2.0 全速协议
- 含有 4 个通用双向传输 End Point（EP1、EP2、EP3、EP4）
- End Point 支持最大包长度 64Byte，支持 Memory 和 Fifo 两种访问功能
- FIFO 模式支持 32bit 方式访问
- Memory 访问支持 8、16、32bit 三种访问方式
- 支持挂起、唤醒和远程唤醒功能
- 支持 Toggle 硬件比对与软件控制功能
- 支持每一个 End Point 数据传输产生中断功能
- 支持 Bus Reset、Suspend 和 Resume 中断功能
- 支持可选的 CRC 错误回复 NAK 功能
- 支持数据包超过最大包长度（64bytes）自动回复 NAK 功能
- 支持 IN 操作主机未回 ACK，接下来 IN 操作 USB 设备回复 NAK 功能
- 支持令牌包与数据包 EOP 丢失检测，支持可选的丢失 EOP 自动回复 NAK 功能

35.3 功能描述

35.3.1 中断状态和控制寄存器

USB设备对应每一个中断都有中断状态、中断使能、中断清除寄存器，分别是：

- USB_INTSTATRAW：原始中断标志寄存器，无论中断有无使能都会显示中断发生的状态。
- USB_INTEN：中断使能寄存器，使能某个中断的发生。
- USB_INTCLR：中断清除寄存器，用户用这个寄存器来清除中断发生的标志。只需要写入1即可清除中断，该寄存器会自动清零。

35.3.2 地址设置 Set Address

控制传输中Set Address命令可以全部由硬件完成。软件如果需要知道Set Address命令已经发生，则可以使用中断位SETADDR。

35.3.3 远程唤醒

远程唤醒功能，由USB_WORKINGMODE[20]来实现。向USB_WORKINGMODE[20]写1，将会在USB端口上发出一个K状态，其持续的时间由软件控制。软件向USB_WORKINGMODE[20]写0，USB端口停止发送K状态。

在发起远程唤醒前，需要确认当前USB PHY是否为使能状态，PHY产生的CLK48M时钟是否开启。如果没有CLK48M时钟，控制器将无法发起远程唤醒操作。

35.3.4 令牌包与数据包 CRC 出错选择性回复 NAK

令牌包和数据包发生CRC错误，控制器可以选择性的回复NAK，通过USB_WORKINGMODE[10]来控制。当USB_WORKINGMODE[10]为1时，USB设备在CRC出错时的握手时相回复NAK握手包，否则将不回复握手包。

35.3.5 数据包长度超过 64byte

当USB控制器的端点接收到的数据包长度超过64byte时，状态寄存器USB_INTSTATRAW[24]将会置1。如果USB_INTEN[24]为1，将会将会产生中断。

35.3.6 包丢失 Eop

当USB_WORKINGMODE[23]与USB_WORKINGMODE[24]为1时，令牌包或数据包在限定的包长度内未收到包长度中状态（Eop），状态寄存器USB_INTSTATRAW[30]将会置1。如果USB_INTEN[30]为1，将会将会产生中断。控制器对令牌包按长度32个Full Speed比特位来检测，当令牌包在超过32个Full Speed比特位时未收到EOP，则会触发此错误，设备会复位内部状态机，从而不影响下一个包的接收。数据包按有效数据64byte+8byte检测，当数据包内有效数据长度超过此值时，也会产生此错误。

令牌包和数据包在发生此类错误时，可以选择是否向USB主机回复NAK握手包。分别通过设置USB_WORKINGMODE[21]和USB_WORKINGMODE[22]来设置。当USB_WORKINGMODE[21]为1时，发生令牌包此类错误时，在令牌包的第32+11个Full Speed比特位后，USB控制器将回复NAK握手包。当USB_WORKINGMODE[22]位为1时，发生数据包此类错误时，在数据包的第

64byte+8byte+11bit个Full Speed比特位后，USB控制器将回复NAK握手包。

35.3.7 IN 操作 ACK 超时下次 IN 操作回复 NAK

当主机发起IN操作时，如果在握手时相，主机需回复的ACK握手包超时，则根据不同的端点，状态寄存器USB_INTSTATRAW中的EPx_IN_HANDSHAKE_ERR_RAW将会被置1。当EPx_IN_HANDSHAKE_ERR_RAW为1且主机再次发起针对此端点的IN操作时，USB控制器将会回复NAK握手包，直到软件清除EPx_IN_HANDSHAKE_ERR_RAW位。

35.3.8 FIFO 访问与 Memory 访问

此USB控制器支持FIFO访问与Memory访问两种方式。FIFO只支持32bit访问，而Memory访问支持8、16、32bit访问。

35.4 寄存器描述

USB 寄存器基地址：0x4090_0000

寄存器列表如下：

表 35-1：USB 寄存器列表

偏置	名称	描述
0X00	USB_WORKINGMODE	工作模式寄存器
0X04	USB_EP0CSR	EP0 传输控制寄存器
0X08	USB_EP1CSR	EP1 传输控制寄存器
0X0C	USB_EP2CSR	EP2 传输控制寄存器
0X10	USB_EP3CSR	EP3 传输控制寄存器
0X14	USB_EP4CSR	EP4 传输控制寄存器
0X18	USB_ADDR	USB 地址寄存器
0X1C	USB_SETUP03DATA	SETUP 数据包寄存器 0
0X20	USB_SETUP47DATA	SETUP 数据包寄存器 1
0X24	USB_EPADDR	End Point 地址配置寄存器
0X28	USB_CURRENTPID	当前 USB 总线包 PID 寄存器
0X2C	USB_CURRENTFRAMENUMBER	Frame Number 寄存器
0X30	USB_CRCERRORCNT	CRC 错误 Counter 寄存器
0X34	USB_STATUSDETECTCNT	Suspend/Resume/Reset 探测时间寄存器
0X40	USB_EP0SENDBN	EP0 发送数据数目寄存器
0X44	USB_EP1SENDBN	EP1 发送数据数目寄存器
0X48	USB_EP2SENDBN	EP2 发送数据数目寄存器
0X4C	USB_EP3SENDBN	EP2 发送数据数目寄存器
0X50	USB_EP4SENDBN	EP4 发送数据数目寄存器
0X100	USB_EP0FIFO	EP0 FIFO 访问入口
0X104	USB_EP1FIFO	EP1 FIFO 访问入口
0X108	USB_EP2FIFO	EP2 FIFO 访问入口
0X10C	USB_EP3FIFO	EP3 FIFO 访问入口

偏置	名称	描述
0X110	USB_EP4FIFO	EP4 FIFO 访问入口
0X200~23C	USB_EP0MEM	EP0 Memory 访问入口
0X240~27C	USB_EP1MEM	EP1 Memory 访问入口
0X280~2BC	USB_EP2MEM	EP2 Memory 访问入口
0X2C0~2FC	USB_EP3MEM	EP3 Memory 访问入口
0X300~33C	USB_EP4MEM	EP4 Memory 访问入口
0XFFE4	USB_INTSTATRAW	状态寄存器
0XFFE8	USB_INTEN	中断使能寄存器
0XFFF0	USB_INTCLR	中断清除寄存器

35.4.1 工作模式寄存器（USB_WORKINGMODE）

偏移地址：0x00

复位值：0x0000 0009

位	名称	属性	复位值	描述
31:26	RSV	-	-	保留
25	USB_EP0_ZOD_INTR_EN	R/W	0x0	USB OUT操作中的0长度数据包是否产生中断选择： 1：使能此中断 0：不使能此中断
24	USB_DATA_NOEOP_EN	R/W	0x0	USB控制器在收到的数据包长度超过64+8Byte时，会认为此包数据丢失了EOP，控制器内部会复位状态机，并产生错误状态： 1：使能此功能 0：不使能此功能
23	USB_TOKEN_NOEOP_EN	R/W	0x0	USB控制器在收到的令牌包长度超过规定长度时，会认为此令牌包丢失了EOP，控制器内部会复位状态机，并产生错误状态： 1：使能此功能 0：不使能此功能
22	USB_DATA_NOEOP_NAK_EN	R/W	0x0	USB控制器在收到的数据包长度超过64+8Byte时，会认为此包数据丢失了EOP。此位为1时，控制器会回复NAK握手包给主机。 1：回复NAK 0：不回复NAK 此功能只有在USB_DATA_NOEOP_En为1时有效。
21	USB_TOKEN_NOEOP_NAK_EN	R/W	0x0	USB控制器在收到的令牌包长度超过规定长度时，会认为此令牌包丢失了EOP。此位为1时，控制器会回复NAK握手包给主机： 1：回复NAK 0：不回复NAK 此功能只有在USB_TOKEN_NOEOP_En为1时有效。

位	名称	属性	复位值	描述
20	USB_REMOTE_WAKEUP	R/W	0x0	USB远程唤醒控制位： 1：控制器发送K状态 0：控制器不发送K状态 唤醒结束后，此位需要软件写0清除。
19:13	RSV	-	-	保留
12	USB_MORETHAN64_NAK_EN	R/W	0x0	收到的数据包长度超过64Byte，回复NAK握手包使能位： 1：使能此功能 0：不使能此功能
11	USB_IN_TIMEOUT_NAK_EN	R/W	0x0	控制器在IN操作数据包发送完毕后，未收到主机的ACK握手包，下次该端点的IN操作设备将回复NAK，直到软件清除错误状态位： 1：使能此功能 0：不使能此功能
10	USB_CRCERR_NAK_EN	R/W	0x0	令牌包和数据包CRC错误回复NAK功能使能信号： 1：使能此功能 0：不使能此功能
9:8	USB_LINE_STATE	R	0x0	USB DP/DM信号状态位： 8bit：DM 9bit：DP
7	RSV	-	-	保留
6	USB_DPPU_LO	R/W	0x0	USB控制器DP信号2.8k上拉电阻控制位： 1：开启上拉 0：关闭上拉
5	RSV	-	-	保留
4	USB_DPPU	R/W	0x0	USB控制器Dp信号2.1k上拉电阻控制位： 1：开启上拉； 0：关闭上拉；
3	USB_BUS_AUTO_RST_EN	R/W	0x1	USB总线复位可以复位控制器地址、收发状态机、收发FIFO使能位： 1：使能 0：不使能
2	USB_FORCE_RST	R/W	0x0	复位控制器地址、收发状态机、收发FIFO： 1：复位 0：不复位 每次复位操作后，需软件写0清除该位。
1	USB_SUSPEND	R/W	0x0	1：设置USB PHY为挂起状态 0：清除USB PHY的挂起状态
0	SPEED_MODE	R/W	0x1	USB工作模式选择位： 1：全速模式 0：低速模式 此芯片仅支持全速模式。

35.4.2 EP0 传输控制寄存器（USB_EP0CSR）

偏移地址：0x04

复位值：0x0000 0100

位	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	EP0_RECEIVED_ACK	R	0x0	EP0接收到Host的ACK握手包标志信号： 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset或者向EP0_DATA_START写1可以清除该位。
23:20	RSV	-	-	保留
19	EP0_OUT_TOGGLE_STATE	R	0x0	EP0 Out/Setup状态错误标志： 1：Toggle收到的与预期不符 0：Toggle正常
18	EP0_OUT_TOGGLE_CTRL_En	W	0x0	EP0_OUT_TOGGLE_WANT中的值生效使能： 1：EP0_OUT_TOGGLE_WANT值写入生效
17	EP0_OUT_TOGGLE_WANT	R/W	0x0	EP0 Out/Setup数据包Toggle的比对值： 1：Data1 0：Data0
16	EP0_OUT_TOGGLE_VALUE	R	0x0	EP0收到的数据包的Toggle值： 1：Data1 0：Data0
15	EP0_IN_TOGGLE_CTRL_EN	W	0x0	EP0_IN_TOGGLE_VALUE中值生效使能： 1：EP0_IN_TOGGLE_VALUE值写入生效。
14	EP0_IN_TOGGLE_VALUE	R/W	0x0	IN操作Toggle控制位： 1：Data1 0：Data0 写此位会更改比对寄存器的值，读此位返回的是当前IN Toggle的值。
13	EP0_SEND_STALL_DONE	R/W	0x0	EP0发送STALL完成标志位： 1：STALL发送完成 0：STALL未发送完成 写1清0。
12	EP0_SEND_STALL	W	0x0	STALL发送控制位： 1：发送STALL 0：不发送STALL 一次写操作只发送一次STALL。
11	EP0_RECEIVED_DONE	W	0x0	接收完成控制位。 1：接收完成； 0：接收未完成； 向此位写1，会将FIFO置为Ready状态，每次接受完数据并且读取完数据后都需要向此位写1，否则下次OUT/SETUP操作，设备将会回NAK。
10	EP0_DATA_START	W	0x0	发送START： 1：将FIFO中的数据发出 0：无操作

位	名称	属性	复位值	描述
9	EP0_FIFOCLR	W	0x0	EP0 FIFO指针复位控制位： 1：复位FIFO指针 0：不复位FIFO指针
8	EP0_EN	R/W	0x1	EP0端点使能位： 1：使能 0：不使能
7:0	EP0_RECEIVED_BYTE	R	0x0	EP0接收到的数据Byte数目。此位需要小于最大包长度。

35.4.3 EP1 传输控制寄存器（USB_EP1CSR）

偏移地址：0x08

复位值：0x0000 0100

位	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	EP1_RECEIVED_ACK	R	0x0	EP1接收到Host的ACK握手包标志信号： 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset或者向EP1_DATA_START写1可以清除该位。
23:20	RSV	-	-	保留
19	EP1_OUT_TOGGLE_STATE	R	0x0	EP1 Out/Setup状态错误标志： 1：Toggle收到的与预期不符 0：Toggle正常
18	EP1_OUT_TOGGLE_CTRL_En	W	0x0	EP1_OUT_TOGGLE_WANT中的值生效使能： 1：EP1_OUT_TOGGLE_WANT值写入生效
17	EP1_OUT_TOGGLE_WANT	R/W	0x0	EP1 Out/Setup数据包Toggle的比对值： 1：Data1 0：Data0
16	EP1_OUT_TOGGLE_VALUE	R	0x0	EP1收到的数据包的Toggle值： 1：Data1 0：Data0
15	EP1_IN_TOGGLE_CTRL_EN	W	0x0	EP1_IN_TOGGLE_VALUE中值生效使能： 1：EP1_IN_TOGGLE_VALUE值写入生效
14	EP1_IN_TOGGLE_VALUE	R/W	0x0	IN操作Toggle控制位： 1：Data1 0：Data0 写此位会更改比对寄存器的值，读此位返回的是当前IN Toggle的值。

位	名称	属性	复位值	描述
13	EP1_SEND_STALL_DONE	R	0x0	EP1发送STALL完成标志位： 1：STALL发送完成 0：STALL未发送完成 写1清0。
12	EP1_SEND_STALL	R/W	0x0	STALL发送控制位： 1：发送STALL 0：不发送STALL 一次写操作只发送多次STALL，直到清除该位。
11	EP1_RECEIVED_DONE	W	0x0	接收完成控制位： 1：接收完成 0：接收未完成 向此位写1，会将FIFO置为Ready状态，每次接受完数据并且读取完数据后都需要向此位写1，否则下次OUT/SETUP操作,设备将会回NAK。
10	EP1_DATA_START	W	0x0	发送START： 1：将FIFO中的数据发出 0：无操作
9	EP1_FIFOCLR	W	0x0	EP1 FIFO指针复位控制位： 1：复位FIFO指针 0：不复位FIFO指针
8	EP1_EN	R/W	0x1	EP1端点使能位： 1：使能 0：不使能
7:0	EP1_RECEIVED_BYTE	R	0x0	EP1接收到的数据Byte数目。此位需要小于最大包长度。

35.4.4 EP2 传输控制寄存器（USB_EP2CSR）

偏移地址：0x0C

复位值：0x0000 0100

位	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	EP2_RECEIVED_ACK	R	0x0	EP2接收到Host的ACK握手包标志信号： 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset或者向EP2_DATA_START写1可以清除该位。
23:20	-	-	-	-
19	EP2_OUT_TOGGLE_STATE	R	0x0	EP2 Out/Setup状态错误标志： 1：Toggle收到的与预期不符 0：Toggle正常

位	名称	属性	复位值	描述
18	EP2_OUT_TOGGLE_CTRL_En	W	0x0	EP2_OUT_TOGGLE_WANT中的值生效使能： 1：EP2_OUT_TOGGLE_WANT 值写入生效
17	EP2_OUT_TOGGLE_WANT	R/W	0x0	EP2 Out/Setup数据包Toggle的比对值： 1：Data1 0：Data0
16	EP2_OUT_TOGGLE_VALUE	R	0x0	EP2收到的数据包的Toggle值： 1：Data1 0：Data0
15	EP2_IN_TOGGLE_CTRL_EN	W	0x0	EP2_IN_TOGGLE_VALUE 中值生效使能： 1：EP2_IN_TOGGLE_VALUE值写入生效
14	EP2_IN_TOGGLE_VALUE	R/W	0x0	IN操作Toggle控制位： 1：Data1 0：Data0 写此位会更改比对寄存器的值，读此位返回的是当前IN Toggle的值。
13	EP2_SEND_STALL_DONE	R	0x0	EP2发送STALL完成标志位： 1：STALL发送完成 0：STALL未发送完成 写1清0。
12	EP2_SEND_STALL	R/W	0x0	STALL发送控制位： 1：发送STALL 0：不发送STALL 一次写操作只发送多次STALL，直到清除该位。
11	EP2_RECEIVED_DONE	W	0x0	接收完成控制位： 1：接收完成 0：接收未完成 向此位写1，会将FIFO置为Ready状态，每次接受完数据并且读取完数据后都需要向此位写1，否则下次OUT/SETUP操作,设备将会回NAK。
10	EP2_DATA_START	W	0x0	发送START： 1：将FIFO中的数据发出 0：无操作
9	EP2_FIFOCLR	W	0x0	EP2 FIFO指针复位控制位： 1：复位FIFO指针 0：不复位FIFO指针
8	EP2_EN	R/W	0x1	EP2端点使能位： 1：使能 0：不使能
7:0	EP2_RECEIVED_BYTE	R	0x0	EP2接收到的数据Byte数目。此位需要小于最大包长度。

35.4.5 EP3 传输控制寄存器（USB_EP3CSR）

偏移地址：0x10

复位值：0x0000 0100

位	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	EP3_RECEIVED_ACK	R	0x0	EP3接收到Host的ACK握手包标志信号： 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset或者向EP3_DATA_START写1可以清除该位。
23:20	-	-	-	-
19	EP3_OUT_TOGGLE_STATE	R	0x0	EP3 Out/Setup状态错误标志： 1：Toggle收到的与预期不符 0：Toggle正常
18	EP3_OUT_TOGGLE_CTRL_En	W	0x0	EP3_OUT_TOGGLE_WANT 中的值生效使能： 1：EP3_OUT_TOGGLE_WANT值写入生效
17	EP3_OUT_TOGGLE_WANT	R/W	0x0	EP3 Out/Setup数据包Toggle的比对值： 1：Data1 0：Data0
16	EP3_OUT_TOGGLE_VALUE	R	0x0	EP3收到的数据包的Toggle值： 1：Data1 0：Data0
15	EP3_IN_TOGGLE_CTRL_EN	W	0x0	EP3_IN_TOGGLE_VALUE中值生效使能： 1:EP3_IN_TOGGLE_VALUE值写入生效
14	EP3_IN_TOGGLE_VALUE	R/W	0x0	IN操作Toggle控制位： 1：Data1 0：Data0 写此位会更改比对寄存器的值，读此位返回的是当前IN Toggle的值。
13	EP3_SEND_STALL_DONE	R	0x0	EP3发送STALL完成标志位： 1：STALL发送完成 0：STALL未发送完成 写1清0。
12	EP3_SEND_STALL	R/W	0x0	STALL发送控制位： 1：发送STALL 0：不发送STALL 一次写操作只发送多次STALL，直到清除该位。

位	名称	属性	复位值	描述
11	EP3_RECEIVED_DONE	W	0x0	接收完成控制位： 1：接收完成 0：接收未完成 向此位写1，会将FIFO置为Ready状态，每次接受完数据并且读取完数据后都需要向此位写1，否则下次OUT/SETUP操作，设备将会回NAK。
10	EP3_DATA_START	W	0x0	发送START： 1：将FIFO中的数据发出 0：无操作
9	EP3_FIFOCLR	W	0x0	EP3 FIFO指针复位控制位： 1：复位FIFO指针 0：不复位FIFO指针
8	EP3_EN	R/W	0x1	EP3端点使能位： 1：使能 0：不使能
7:0	EP3_RECEIVED_BYTE	R	0x0	EP3接收到的数据Byte数目。此位需要小于最大包长度。

35.4.6 EP4 传输控制寄存器（USB_EP4CSR）

偏移地址：0x14

复位值：0x0000 0100

位	名称	属性	复位值	描述
31:25	RSV	-	-	保留
24	EP4_RECEIVED_ACK	R	0x0	EP4接收到Host的ACK握手包标志信号： 1：接收到握手包 0：未接收到握手包 此位为只读位，USB Reset或者向EP4_DATA_START写1可以清除该位。
23:20	-	-	-	-
19	EP4_OUT_TOGGLE_STATE	R	0x0	EP4 Out/Setup状态错误标志： 1：Toggle收到的与预期不符 0：Toggle正常
18	EP4_OUT_TOGGLE_CTRL_En	W	0x0	EP4_OUT_TOGGLE_WANT中的值生效使能： 1：EP4_OUT_TOGGLE_WANT值写入生效
17	EP4_OUT_TOGGLE_WANT	R/W	0x0	EP4 Out/Setup数据包Toggle的比对值： 1：Data1 0：Data0
16	EP4_OUT_TOGGLE_VALUE	R	0x0	EP4收到的数据包的Toggle值： 1：Data1 0：Data0

位	名称	属性	复位值	描述
15	EP4_IN_TOGGLE_CTRL_EN	W	0x0	EP4_IN_TOGGLE_VALUE 中值生效使能： 1：EP4_IN_TOGGLE_VALUE值写入生效
14	EP4_IN_TOGGLE_VALUE	R/W	0x0	IN操作Toggle控制位： 1：Data1 0：Data0 写此位会更改比对寄存器的值，读此位返回的是当前IN Toggle的值。
13	EP4_SEND_STALL_DONE	R	0x0	EP4发送STALL完成标志位： 1：STALL发送完成 0：STALL未发送完成 写1清0。
12	EP4_SEND_STALL	R/W	0x0	STALL发送控制位： 1：发送STALL 0：不发送STALL 一次写操作只发送多次STALL，直到清除该位。
11	EP4_RECEIVED_DONE	W	0x0	接收完成控制位： 1：接收完成 0：接收未完成 向此位写1，会将FIFO置为Ready状态，每次接受完数据并且读取完数据后都需要向此位写1，否则下次OUT/SETUP操作,设备将会回NAK。
10	EP4_DATA_START	W	0x0	发送START： 1：将FIFO中的数据发出 0：无操作
9	EP4_FIFOCLR	W	0x0	EP4 FIFO指针复位控制位： 1：复位FIFO指针 0：不复位FIFO指针
8	EP4_EN	R/W	0x1	EP4端点使能位： 1：使能 0：不使能
7:0	EP4_RECEIVED_BYTE	R	0x0	EP4接收到的数据Byte数目。此位需要小于最大包长度。

35.4.7 USB 地址寄存器（USB_ADDR）

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6:0	USB_ADDR	R/W	0x0	USB地址寄存器（只可读）

35.4.8 SETUP 数据包寄存器（USB_SETUP03DATA）

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	SETUP_0_3_DATA	R	0x0	SETUP Data包Byte0~Byte3寄存器。

35.4.9 SETUP 数据包寄存器（USB_SETUP47DATA）

偏移地址：0x20

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	SETUP_4_7_DATA	R	0x0	SETUP Data包Byte4~Byte7寄存器。

35.4.10 END POINT 地址配置寄存器（USB_EPADDR）

偏移地址：0x24

复位值：0x0000 4321

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:12	EP4_ADDR	R/W	0x4	Ep4地址配置。
11:8	EP3_ADDR	R/W	0x3	Ep3地址配置。
7:4	EP2_ADDR	R/W	0x2	Ep2地址配置。
3:0	EP1_ADDR	R/W	0x1	Ep1地址配置。

35.4.11 总线包 PID 寄存器（USB_CURRENTPID）

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:4	RSV	-	-	保留
3:0	CURRENT_PID	R	0x0	当前接收的USB包的PID值。

35.4.12 FRAME NUMBER 寄存器（USB_CURRENTFRAMENUMBER）

偏移地址：0x2C

复位值：0x0000 003F

位	名称	属性	复位值	描述
31:11	RSV	-	-	保留
10:0	CURRENT_FRAME_NUMBER	R	0x3F	当前帧序号。

35.4.13 CRC 错误 COUNTER 寄存器（USB_CRCERRORCNT）

偏移地址：0x30

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	CRC_ERROR_CNT	R	0x0	CRC错误包的个数，在USB Reset时复位。

35.4.14 探测时间寄存器（USB_STATUSDETECTCNT）

偏移地址：0x34

复位值：0x0000 00FF

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	USB_STATUS_DETECT_CNT	R/W	0xFF	Reset/Resume/Suspend检测阈值设定。每个单位为2.5μs（120个48M时钟周期）。 0x0：阈值为2.5μs 0x1：阈值为5.0μs …..

35.4.15 EP0 发送数据数目寄存器（USB_EP0SENDBN）

偏移地址：0x40

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	EP0_SEND_BYTE	R/W	0x0	EP0发送数据Byte数量寄存器。

35.4.16 EP1 发送数据数目寄存器（USB_EP1SENDBN）

偏移地址：0x44

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	EP1_SEND_BYTE	R/W	0x0	EP1发送数据Byte数量寄存器。

35.4.17 EP2 发送数据数目寄存器（USB_EP2SENDBN）

偏移地址：0x48

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	EP2_SEND_BYTE	R/W	0x0	EP2发送数据Byte数量寄存器。

35.4.18 EP3 发送数据数目寄存器（USB_EP3SENDBN）

偏移地址：0x4C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	EP3_SEND_BYTE	R/W	0x0	EP3发送数据Byte数量寄存器。

35.4.19 EP4 发送数据数目寄存器（USB_EP4SENDBN）

偏移地址：0x50

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	EP4_SEND_BYTE	R/W	0x0	EP4发送数据Byte数量寄存器。

35.4.20 EP0 FIFO 访问入口（USB_EP0FIFO）

偏移地址：0x100

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	EP0FIFO	R/W	0x0	EP0 FIFO入口地址，只支持32bit访问。

35.4.21 EP1 FIFO 访问入口（USB_EP1FIFO）

偏移地址：0x104

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	EP1FIFO	R/W	0x0	EP1 FIFO入口地址，只支持32bit访问。

35.4.22 EP2 FIFO 访问入口（USB_EP2FIFO）

偏移地址：0x108

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	EP2FIFO	R/W	0x0	EP2 FIFO入口地址，只支持32bit访问。

35.4.23 EP3 FIFO 访问入口（USB_EP3FIFO）

偏移地址：0x10C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	EP3FIFO	R/W	0x0	EP3 FIFO入口地址，只支持32bit访问。

35.4.24 EP4 FIFO 访问入口（USB_EP4FIFO）

偏移地址：0x110

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	EP4FIFO	R/W	0x0	EP4 FIFO入口地址，只支持32bit访问。

35.4.25 状态寄存器（USB_INTSTATRAW）

偏移地址：0xFFE4

复位值：0x0000 0000

位	名称	属性	复位值	描述
31	TOGGLE_STATE_ERR_RAW	R	0x0	IN/OUT/Setup操作发生Toggle错误时，此位置1。
30	NOEOP_ERR_RAW	R	0x0	设备接收到的令牌包如果长度超过协议规定值，或者数据包数据超过64+8byte，此状态位会置1。
29	EP4_IN_HANDSHAKE_ERR_RAW	R	0x0	EP4 IN操作，主机未成功返回ACK信号时，此位会置1。
28	EP3_IN_HANDSHAKE_ERR_RAW	R	0x0	EP3 IN操作，主机未成功返回ACK信号时，此位会置1。
27	EP2_IN_HANDSHAKE_ERR_RAW	R	0x0	EP2 IN操作，主机未成功返回ACK信号时，此位会置1。
26	EP1_IN_HANDSHAKE_ERR_RAW	R	0x0	EP1 IN操作，主机未成功返回ACK信号时，此位会置1。
25	EP0_IN_HANDSHAKE_ERR_RAW	R	0x0	EP0 IN操作，主机未成功返回ACK信号时，此位会置1。
24	DATA_BYTE_MORETHAN_64_RAW	R	0x0	收到的DATA数据包长度超过64byte，此位会置1。
23	CRC_ERR_RAW	R	0x0	收到令牌包或者数据包的CRC Error，此位会置1。
22	SETADDR_RAW	R	0x0	当Host设置USB设备地址完成，此位会置1。
21	TURNAROUND_ERROR_RAW	R	0x0	Host回复ACK包发生TimeOut中断。
20	EP4_ACK_RAW	R	0x0	EP4 ACK状态，发送或者接受ACK包，此位会置1。

位	名称	属性	复位值	描述
19	EP4_OUT_RAW	R	0x0	EP4 Out中断，当有效数据进入FIFO时，此位会置1。
18	EP4_IN_RAW	R	0x0	EP4接收到IN令牌包时，此位会置1。
17	EP3_ACK_RAW	R	0x0	Ep3 ACK状态，发送或者接受Ack包，此位会置1。
16	EP3_OUT_RAW	R	0x0	EP3 Out中断，当有效数据进入FIFO时，此位会置1。
15	EP3_IN_RAW	R	0x0	EP3接收到IN令牌包时，此位会置1。
14	EP2_ACK_RAW	R	0x0	EP2 ACK状态，发送或者接受Ack包，此位会置1。
13	EP2_OUT_RAW	R	0x0	EP2 Out中断，当有效数据进入FIFO时，此位会置1。
12	EP2_IN_RAW	R	0x0	Ep2接收到IN令牌包时，此位会置1。
11	EP1_ACK_RAW	R	0x0	EP1 ACK状态，发送或者接受Ack包，此位会置1。
10	EP1_OUT_RAW	R	0x0	EP1 Out中断，当有效数据进入FIFO时，此位会置1。
9	EP1_IN_RAW	R	0x0	Ep1接收到IN令牌包时，此位会置1。
8	EP0_ACK_RAW	R	0x0	EP0 ACK状态，发送或者接受Ack包，此位会置1。
7	EP0_OUT_RAW	R	0x0	EP0 Out中断，当有效数据进入FIFO时，此位会置1。
6	EP0_IN_RAW	R	0x0	EP0接收到IN令牌包时，此位会置1。
5	SUDAV_RAW	R	0x0	接收到Setup数据包，此位会置1。
4	SETUPTOK_RAW	R	0x0	接收到Setup令牌包，此位会置1。
3	SOF_RAW	R	0x0	接收到Sof包，此位会置1。
2	RESUME_RAW	R	0x0	Host Resume，此位会置1。
1	SUSPEND_RAW	R	0x0	Host Suspend，此位会置1。
0	BUS_RESET_RAW	R	0x0	Host Reset，此位会置1。

35.4.26 中断使能寄存器（USB_INTEN）

偏移地址：0xffe8

复位值：0x0000 0000

位	名称	属性	复位值	描述
31	TOGGLE_STATE_ERR_EN	R/W	0x0	TOGGLE_STATE_ERR中断使能。
30	NOEOP_ERR_EN	R/W	0x0	NOEOP_ERR中断使能。
29	EP4_IN_HANDSHAKE _ERR_EN	R/W	0x0	EP4_IN_HANDSHAKE_ERR中断使能。

位	名称	属性	复位值	描述
28	EP3_IN_HANDSHAKE _ERR_EN	R/W	0x0	EP3_IN_HANDSHAKE_ERR 中断使能。
27	EP2_IN_HANDSHAKE _ERR_EN	R/W	0x0	EP2_IN_HANDSHAKE_ERR 中断使能。
26	EP1_IN_HANDSHAKE _ERR_EN	R/W	0x0	EP1_IN_HANDSHAKE_ERR 中断使能。
25	EP0_IN_HANDSHAKE_ERR_EN	R/W	0x0	EP0_IN_HANDSHAKE_ERR 中断使能。
24	DATA_BYTE_MORETHAN_64_EN	R/W	0x0	收到的DATA数据包长度超过64byte中断使能。
23	CRC_ERR_EN	R/W	0x0	CRC错误中断使能。
22	SETADDR_EN	R/W	0x0	当Host设置USB设备地址完成中断使能。
21	TURNAROUND_ERROR_EN	R/W	0x0	Host回复Ack包发生TimeOut中断使能。
20	EP4_ACK_EN	R/W	0x0	Ep4 Ack状态，发送或者接受Ack包中断使能。
19	EP4_OUT_EN	R/W	0x0	Ep4 Out中断，当有效数据进入FIFO时产生中断使能。
18	EP4_IN_EN	R/W	0x0	Ep4接收到IN令牌包中断使能。
17	EP3_ACK_EN	R/W	0x0	Ep3 Ack状态，发送或者接受Ack包中断使能。
16	EP3_OUT_EN	R/W	0x0	Ep3 Out中断，当有效数据进入FIFO时产生中断使能。
15	EP3_IN_EN	R/W	0x0	Ep3接收到IN令牌包中断使能。
14	EP2_ACK_EN	R/W	0x0	Ep2 Ack状态，发送或者接受Ack包中断使能。
13	EP2_OUT_EN	R/W	0x0	Ep2 Out中断，当有效数据进入FIFO时产生中断使能。
12	EP2_IN_EN	R/W	0x0	Ep2接收到IN令牌包中断使能。
11	EP1_ACK_EN	R/W	0x0	Ep1 Ack状态，发送或者接受Ack包中断使能。
10	EP1_OUT_EN	R/W	0x0	Ep1 Out中断，当有效数据进入FIFO时产生中断使能。
9	EP1_IN_EN	R/W	0x0	Ep1接收到IN令牌包中断使能。
8	EP0_ACK_EN	R/W	0x0	Ep0 Ack状态，发送或者接受Ack包中断使能。
7	EP0_OUT_EN	R/W	0x0	Ep0 Out中断，当有效数据进入FIFO时产生中断使能。
6	EP0_IN_EN	R/W	0x0	Ep0接收到IN令牌包中断使能。
5	SUDAV_EN	R/W	0x0	接收到Setup数据包中断使能。
4	SETUPTOK_EN	R/W	0x0	接收到Setup令牌包中断使能。
3	SOF_EN	R/W	0x0	接收到Sof包中断使能。
2	RESUME_EN	R/W	0x0	Host Resume中断使能。
1	SUSPEND_EN	R/W	0x0	Host Suspend中断使能。
0	BUS_RESET_EN	R/W	0x0	Host Reset中断使能。

35.4.27 中断清除寄存器（USB_INTCLR）

偏移地址：0xFFF0

复位值：0x0000 0000

位	名称	属性	复位值	描述
31	TOGGLE_STATE_ERR_CLR	R/W	0x0	TOGGLE_STATE_ERR _RAW中断寄存器清除位： 1：TOGGLE_STATE_ERR_RAW清0 0：TOGGLE_STATE_ERR_RAW保持不变
30	NOEOP_ERR_CLR	R/W	0x0	NOEOP_ERR_RAW中断寄存器清除位： 1：NOEOP_ERR_RAW清0 0：NOEOP_ERR_RAW保持不变
29	EP4_IN_HANDSHAKE _ERR_CLR	R/W	0x0	EP4_IN_HANDSHAKE_ERR_RAW中断寄存器清除位： 1：EP4_IN_HANDSHAKE_ERR_RAW清0 0：EP4_IN_HANDSHAKE_ERR_RAW保持不变
28	EP3_IN_HANDSHAKE _ERR_CLR	R/W	0x0	EP3_IN_HANDSHAKE_ERR_RAW中断寄存器清除位： 1：EP3_IN_HANDSHAKE_ERR_RAW清0 0：EP3_IN_HANDSHAKE_ERR_RAW保持不变
27	EP2_IN_HANDSHAKE _ERR_CLR	R/W	0x0	EP2_IN_HANDSHAKE_ERR_RAW中断寄存器清除位： 1：EP2_IN_HANDSHAKE_ERR_RAW清0； 0：EP2_IN_HANDSHAKE_ERR_RAW保持不变。
26	EP1_IN_HANDSHAKE _ERR_CLR	R/W	0x0	EP1_IN_HANDSHAKE_ERR_RAW中断寄存器清除位： 1：EP1_IN_HANDSHAKE_ERR_RAW清0 0：EP1_IN_HANDSHAKE_ERR_RAW保持不变
25	EP0_IN_HANDSHAKE_ERR_ CLR	R/W	0x0	EP0_IN_HANDSHAKE_ERR_RAW中断寄存器清除位： 1：EP0_IN_HANDSHAKE_ERR_RAW清0 0：EP0_IN_HANDSHAKE_ERR_RAW保持不变
24	DATA_BYTE_MORETHAN_64 _CLR	R/W	0x0	DATA_BYTE_MORETHAN_64_RAW中断寄存器清除位： 1：DATA_BYTE_MORETHAN_64_RAW清0； 0：DATA_BYTE_MORETHAN_64_RAW保持不变。

位	名称	属性	复位值	描述
23	CRC_ERR_CLR	R/W	0x0	CRC_ERR_RAW中断寄存器清除位： 1：CRC_ERR_RAW清0 0：CRC_ERR_RAW保持不变
22	SETADDR_CLR	R/W	0x0	SETADDR_RAW中断寄存器清除位： 1：SETADDR_RAW清0 0：SETADDR_RAW保持不变
21	TURNAROUND_ERROR_CLR	R/W	0x0	TURNAROUND_ERROR_RAW中断寄存器清除位： 1：TURNAROUND_ERROR_RAW清0 0：TURNAROUND_ERROR_RAW保持不变
20	EP4_ACK_CLR	R/W	0x0	EP4_ACK_RAW中断寄存器清除位： 1：EP4_ACK_RAW清0； 0：EP4_ACK_RAW保持不变。
19	EP4_OUT_CLR	R/W	0x0	EP4_OUT_RAW中断寄存器清除位： 1：EP4_OUT_RAW清0 0：EP4_OUT_RAW保持不变
18	EP4_IN_CLR	R/W	0x0	EP4_IN_RAW中断寄存器清除位： 1：EP4_IN_RAW清0 0：EP4_IN_RAW保持不变
17	EP3_ACK_CLR	R/W	0x0	EP3_ACK_RAW中断寄存器清除位： 1：EP3_ACK_RAW清0 0：EP3_ACK_RAW保持不变
16	EP3_OUT_CLR	R/W	0x0	EP3_OUT_RAW中断寄存器清除位： 1：EP3_OUT_RAW清0 0：EP3_OUT_RAW保持不变
15	EP3_IN_CLR	R/W	0x0	EP3_IN_RAW中断寄存器清除位： 1：EP3_IN_RAW清0 0：EP3_IN_RAW保持不变
14	EP2_ACK_CLR	R/W	0x0	EP2_ACK_RAW中断寄存器清除位： 1：EP2_ACK_RAW清0 0：EP2_ACK_RAW保持不变
13	EP2_OUT_CLR	R/W	0x0	EP2_OUT_RAW中断寄存器清除位： 1：EP2_OUT_RAW清0 0：EP2_OUT_RAW保持不变
12	EP2_IN_CLR	R/W	0x0	EP2_IN_RAW中断寄存器清除位： 1：EP2_IN_RAW清0 0：EP2_IN_RAW保持不变
11	EP1_ACK_CLR	R/W	0x0	EP1_ACK_RAW中断寄存器清除位： 1：EP1_ACK_RAW清0 0：EP1_ACK_RAW保持不变
10	EP1_OUT_CLR	R/W	0x0	EP1_OUT_RAW中断寄存器清除位： 1：EP1_OUT_RAW清0 0：EP1_OUT_RAW保持不变
9	EP1_IN_CLR	R/W	0x0	EP1_IN_RAW中断寄存器清除位： 1：EP1_IN_RAW清0 0：EP1_IN_RAW保持不变

位	名称	属性	复位值	描述
8	EP0_ACK_CLR	R/W	0x0	EP0_ACK_RAW中断寄存器清除位： 1：EP0_ACK_RAW清0 0：EP0_ACK_RAW保持不变
7	EP0_OUT_CLR	R/W	0x0	EP0_OUT_RAW中断寄存器清除位： 1：EP0_OUT_RAW清0 0：EP0_OUT_RAW保持不变
6	EP0_IN_CLR	R/W	0x0	EP0_IN_RAW中断寄存器清除位： 1：EP0_IN_RAW清0 0：EP0_IN_RAW保持不变
5	SUDAV_CLR	R/W	0x0	接收到Setup数据包中断寄存器清除位： 1：SUDAV_RAW清0 0：SUDAV_RAW保持不变
4	SETUPTOK_CLR	R/W	0x0	接收到Setup令牌包中断寄存器清除位： 1：SETUPTOK_RAW清0 0：SETUPTOK_RAW保持不变
3	SOF_CLR	R/W	0x0	接收到Sof包中断寄存器清除位： 1：SOF_RAW清0 0：SOF_RAW保持不变
2	RESUME_CLR	R/W	0x0	Host Resume中断寄存器清除位： 1：RESUME_RAW清0 0：RESUME_RAW保持不变
1	SUSPEND_CLR	R/W	0x0	Host Suspend中断寄存器清除位： 1：SUSPEND_RAW清0 0：SUSPEND_RAW保持不变
0	BUS_RESET_CLR	R/W	0x0	Host Reset中断寄存器清除位： 1：BUS_RESET_RAW清0 0：BUS_RESET_RAW保持不变

35.5 使用流程

35.5.1 USB 连接

默认状态USB是不连接的，需要在完成初始化之后，将USB_WORKINGMODE[4]和USB_WORKINGMODE[6]置位，反之可以断开USB连接。

35.5.2 SETUP 数据和 EP0 控制传输数据

控制传输中 set address命令全部由硬件完成，软件如果需要知道set address命令已经发生，则可以使用最后一个中断位SETADDR。

USB 每次控制传输都要经过，SETUP phase，DATA phase(可选)，STATUS phase。

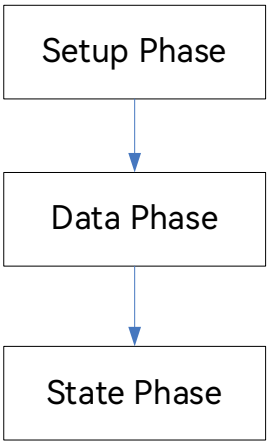


图 35-1：USB传输图

STATUS phase由长度为0的数据包来完成。

- Setup phase：当控制传输SETUP数据接收完毕，USB会产生SUDAV_INT中断，这时SETUP数据会在寄存器USB_SETUP03DATA和USB_SETUP47DATA。
- Data phase：当数据Data Phase方向为OUT，那么当数据传输完成EP0OUT_INT中断将会发生，随后数据的结果将保存在EP0的FIFO中，而数据发生的状态将保存在EP0CSR中。当数据Data Phase方向为IN，那么当Host发来Ep0的In Token，这时EP0IN_INT中断将被触发，同时会将EP0 FIFO中的数据按照EP0CSR中的指示发送会Host。如果数据没有准备好，那么USB将自动回复NAK packet。
- Status phase：Status Phase的Status可能有Host发给Device，也有可能由Device发回Host，取决于Data Phase的方向。这需要软件维护这一状态。如果需要由Device发给Host，用户需要采用Data Phase的方法，将0长度数据准备好。USB会在EN0IN_INT中断发生时将数据回给Host已完成整个控制传输的Status Phase。

35.5.3 Endpoint In 传输

当host发来Epx的IN token，这时EPxIN_INT中断将被触发，同时会将Epx FIFO中的数据按照EPxCSR中的指示发送会Host，包括发送的数据长度等。Endpoint In传输的流流程图如下图所示：

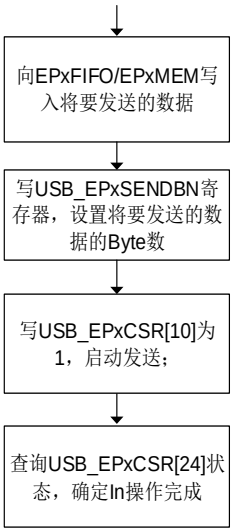


图 35-2：USB Endpoint In 传输流程图

35.5.4 Endpoint Out 传输

当EPxOUT_INT发生时，USB_EPxCSR[7:0]字段记录了数据接收的长度，用户可以根据这一长度，从EPxFIFO或者EPxMEM中读取数据。当EPxOUT的数据读取完成之后，用户需要写一下USB_EPxCSR[11]，已表示可以允许接受下一笔数据。Endpoint Out传输的流程图如下图所示：

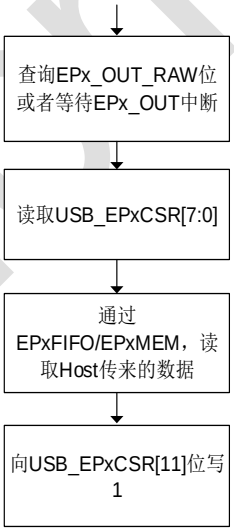


图 35-3：USB Endpoint Out 传输流程图

36 模数转换器（ADC）

36.1 概述

模数转换器（ADC）对通道电压采样，然后接收模数转换结果，并将经过简单处理的结果传递给 CPU 或系统中的其他模块。

本 ADC 控制器控制着一个 12 位逐次逼近模数转换器，它具有 16 个外部通道，而且相邻的两个外部通道可以组成一对差分输入信号。本 ADC 控制器支持最多 16 个位置的常规序列和最多 4 个位置的注入序列，支持双 ADC 协同扫描，具有模拟看门狗功能。本 ADC 控制器有一个 32 级深的接收器 FIFO，而且每个通道都有独立的数据寄存器。

36.2 主要特性

- $V_{REFN} \le V_{IN} \le V_{REFP}$，其中，$V_{REFN}$ 连接模拟电路的地 V_{SSA}，V_{REFP} 可选模拟电路的电源 V_{DDA} 或外部输入的参考电压 V_{REF}
- 2 个 ADC 可与内部运算放大器（OPA）配合使用，在采样之前先放大信号或者作为 BUFFER
- 可配置 ADCCLK 时钟
- 分辨率 12 位
- 上电稳定时间可配置 32~63 个 ADCCLK 周期
- 提供 16 路单端外部输入通道，其中相邻的两路通道可作为一对差分外部输入通道常规序列提供最多 8 个通道转换的位置，每个位置可以选择任意一个通道。
- 触发方式可由软件配置寄存器触发，或内部定时器事件（上升沿、下降沿或双边沿）触发，或 GPIOA 中断触发
- 转换模式有单次模式、连续模式、断续模式、双 ADC 协同模式
- 有常规序列和注入序列。常规序列最多包含 20 个通道转换的位置，注入序列最多包含 4 个通道转换的位置，常规序列和注入序列中的每个位置可以选择任意一个通道。注入序列的优先级高于常规序列。
- 可连续多次（2, 4, 8, 16, 32, 64 或 128 次）对某个或某些通道采样转换并计算平均数
- 支持模拟看门狗功能，监视转换结果
- 每个通道都有独立的数据寄存器，可统一设定读取后是否自动清除数据
- 两个 ADC 控制器各有一个 32 级深 16 位宽的接收器 FIFO，用于保存常规序列和注入序列的转换结果
- 支持 DMA 模式

36.3 系统框图

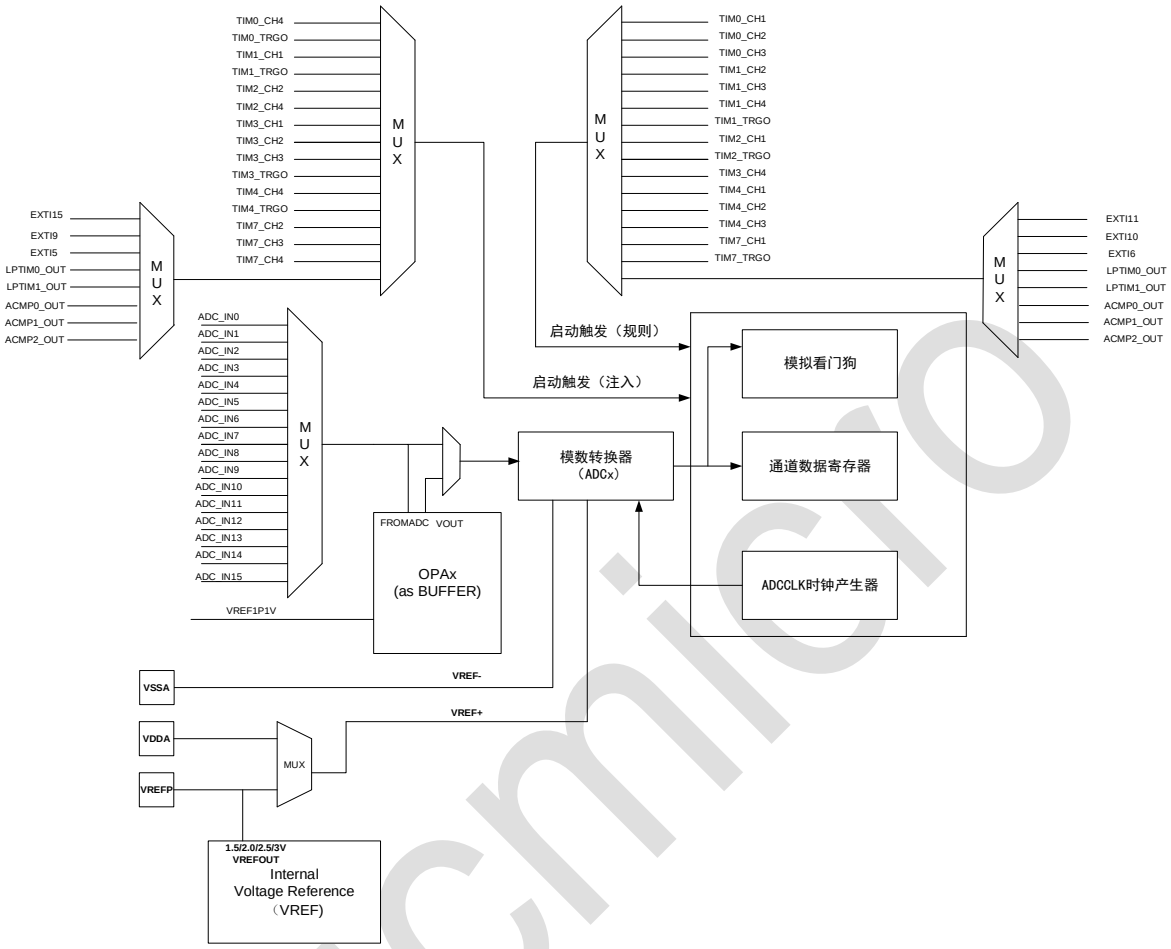


图 36-1：ADC 系统框图

注：

- OPA0 作为 ADC0 的 BUFFER 功能使用。
- OPA1 作为 ADC1 的 BUFFER 功能使用。

36.4 管脚说明

表 36-1：ADC 管脚说明

功能管脚	复用管脚	方向	功能描述
ADC_IN0	PA0	AI	ADC模拟输入通道
ADC_IN1	PA1	AI	ADC模拟输入通道
ADC_IN2	PA2	AI	ADC模拟输入通道
ADC_IN3	PA3	AI	ADC模拟输入通道
ADC_IN6	PA6	AI	ADC模拟输入通道
ADC_IN7	PA7	AI	ADC模拟输入通道
ADC_IN8	PB0	AI	ADC模拟输入通道
ADC_IN9	PB1	AI	ADC模拟输入通道
ADC_IN10	PC0	AI	ADC模拟输入通道

功能管脚	复用管脚	方向	功能描述
ADC_IN11	PC1	AI	ADC模拟输入通道
ADC_IN12	PC2	AI	ADC模拟输入通道
ADC_IN13	PC3	AI	ADC模拟输入通道
ADC_IN14	PC4	AI	ADC模拟输入通道
ADC_IN15	PC5	AI	ADC模拟输入通道
VDDA	-	AP	模拟电源3.3V
VSSA	-	AG	模拟地
VREFP	-	AP	模拟正向参考电压

36.5 功能描述

36.5.1 模拟看门狗

模拟看门狗可以比较通道数据与上下阈值，如果通道数据超出由上下阈值划定的范围，看门狗就会产生中断标志。需要比较的通道可以独立使能，上下两个阈值均可以自由配置，比较条件也可以选择。中断标志有三个：通道数据看门狗报警中断标志、常规序列看门狗报警中断标志和注入序列看门狗报警中断标志，其中，通道数据看门狗报警中断标志不区分数据所属的序列。这三个中断标志都需要软件写 1 清除，硬件不会自动清除。

提示：除了在开启转换之前设置好模拟看门狗以外，还可以在转换过程中修改模拟看门狗的设置。每当模拟看门狗完成对最新通道数据的比较，当前状态寄存器（STAT）的第 27 位看门狗比较通道数据完成标志（Wdg_cmplt）就会置 1。这时，就可以改变模拟看门狗的设置，设置会立即生效。Wdg_cmplt 标志会在软件写入看门狗比较条件寄存器时被硬件清除，或者软件直接对这个标志位写 1 也可以清 0。然而，由于通道切换频次较高，而软件读取寄存器、分析数据、再写入寄存器的过程耗时较长，因此这种做法存在比较条件更新不及时的风险。

36.5.2 ADC 时钟生成器

ADC 时钟生成器，按照 CLKCTRL 寄存器设定的分频系数 PCLK_DIV，对 APB 时钟（PCLK）分频，生成仅供模拟 ADC 使用的时钟 ADCCLK；而 ADC 控制器所使用的时钟是系统时钟（PCLK）。

在双 ADC 协同工作模式下，无论是常规序列协同还是注入序列协同，ADC1 都永远使用与 ADC0 相同的时钟，即 ADC1 的时钟由 ADC 控制器 0 的 CLKCTRL 寄存器所选择和生成，而 ADC 控制器 1 的 CLKCTRL 寄存器中的时钟选择和时钟分频设置则不起作用。当 ADC1 独立工作时，ADC1 的时钟仍然由 ADC 控制器 1 产生。

ADCCLK 有两个来源：由 ADC 控制器内部时钟生成器产生的 ADC 时钟，或者外部输入时钟。

36.5.2.1 内部时钟生成器

内部生成的 ADC 时钟频率公式为：

$$f_{ADCCLK} = f_{PCLK}/(Pclk\_div + 1)$$

其中，f_{ADCCLK} 是 ADC 内部时钟的频率，f_{PCLK} 是 APB 时钟频率，Pclk_div 是储存在 CLKCTRL 寄存器中的分频系数设置寄存器，其有效范围是 0 ≤ Pclk_div ≤ 65535。

就 ADC 控制器本身逻辑的时序而言，对于 ADC 独立工作模式或常规序列延迟连续扫描模式，支持最快 1 分频；对于双 ADC 协同模式，支持最快 2 分频。ADCC 单独扫描或延迟连续扫描时 ADCCLK 最快允许 1 分频，ADCC 并行扫描、延迟单次扫描或延迟断续扫描时 ADCCLK 最快允许 2 分频。

注意：模拟 ADC 最高支持 5.25MSPS 的采样率。

36.5.2.2 外部时钟同步电路

外部输入的 ADC 时钟先经过 PCLK 时钟域的同步（用 PCLK 采样外部时钟），再输入到模拟 ADC，所以外部输入的 ADC 时钟频率不能大于 PCLK 频率的一半，即：

$$f_{ADCCLK} \leq f_{PCLK} / 2$$

36.5.3 数据接收器

数据接收器用于接收 ADC 的转换结果，计算总和与平均值，并将平均值传递给寄存器和 FIFO。

36.5.4 接收器 FIFO

接收器 FIFO 有 32 级深（可以保存 32 条数据），每条数据由 4 位的通道编号和 12 位的转换结果组成，如下表所示，接收器 FIFO 的数据从寄存器逻辑模块读取，发送给 CPU 或 DMA。

表 36-2：接收器 FIFO 中数据格式

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
通道编号低 4 位				转换结果											

接收器 FIFO 接收到足够（由 DGCTRL 寄存器的 Watermark 选择数量）数据时，RXFIFO 数据可用中断标志（FIFO_AVL）将会置 1；如果使能了 DMA 传输，同时还将发出 DMA 请求。

当接收器 FIFO 已满时，新的数据将无法写入。

如果接收器 FIFO 溢出，RXFIFO 溢出中断标志（FIFO_OVF）将会置 1。

36.6 转换工作模式描述

36.6.1 转换序列概述

ADC 控制器有常规序列和注入序列两个序列。常规序列最多包含 20 个通道转换的位置，注入序列最多包含 4 个通道转换的位置，规则序列和注入序列中的每个位置可以选择任意一个通道。

当 ADC 处于稳定化状态或空闲状态下，都可以接收常规触发信号；当 ADC 处于稳定化状态、空闲状态或常规状态下，都可以接收注入触发信号。如果 ADC 在稳定化状态下接收到常规或注入触发信号，那么 ADC 将在稳定之后才开始转换。如果 ADC 在空闲状态下接收到常规或注入触发信号，那么 ADC 将立即开始转换。

注入序列的优先级总是高于常规序列，但新来的注入触发信号不能打断正在进行的注入序列转换。当常规序列正在转换时触发了注入序列转换，则需看被打断的常规序列通道 adc_soc 信号次数是否等于设置的通道转换次数（即被打断的通道是否在进行最后一次转换），若以达到通道的转换次数，那么此被打断通道可正常进行转换并写入数据寄存器或者 FIFO，若没有达到通道转换次数，则等待最近一次转换完成后丢弃数据开始，进入注入序列转换。如果当注入序列正在转换时，有常规触发信号或新的注入触发信号到来，那么新来的触发 信号将被忽略，原先正在转换的注入序列将继续进行。如果在空闲状态下，同一个 PCLK 时钟周期内触发了常规序列转换和注入序列转换，那么常规触发信号将被忽略，注入序列将开始转换。如果常规序列的转换被注入序列打断，那么当注入序列转换结束后，将需要 16 个 ADCCLK 周期（期间 ADC_SOC 信号没有发出）来按照 DGCTRL 寄存器中 RGL_RCV_SEL 位的选择来恢复常规序列的转换，要么从被打断的位置开始继续转换，要么从首个位置开始重新转换。

对于独立工作的 ADC，常规序列的转换模式有单次扫描模式、连续扫描模式和断续扫描模式共三种，而注入序列的转换模式只有单次触发扫描一种。

对于协同工作的 ADC，常规序列的协同方式有独立扫描模式、同步扫描模式和延迟扫描模式共三种，而注入序列的协同方式有独立扫描模式、并行扫描模式和轮流触发扫描模式共三种。协同工作方式的选项应该和独立工作模式的选项组合使用，例如“常规序列同步连续扫描+注入序列轮流触发扫描”模式。

特别地，如果常规序列的协同方式选择了同步扫描模式，那么为了保证每轮常规序列的转换都能同步启动，序列“短”的 ADC 会等待序列“长”的 ADC。因为两个 ADC 的常规序列位置数可能不同，每个通道的采样次数（多次采样并计算平均值）也可能不同，所以有可能出现两个 ADC 的常规序列转换时间不同的情况；同理，注入序列转换时间也可能不同。

值得注意的是，注入序列的并行转换模式不能保证同步启动，但通过互相等待一定同步结束。如果常规序列独立扫描，那么两个 ADC 的时钟频率、相位以及两个 ADC 常规序列开启扫描的时间点都可以不相同。当有一个注入序列转换启动到来时，两个 ADC 就会在结束当前进行的长达 16 个

ADCCLK 周期的一次常规转换之后，不经过等待，根据 ADC 控制器 0 产生的时钟，立即启动注入序列转换，所以注入序列启动的时间点很可能不相同。然而，当两个 ADC 结束注入序列转换时，需要等待彼此都结束注入序列转换，状态机才会恢复到进入注入状态之前的状态（可以是常规状态或空闲状态）。

综上所述，如果常规序列的协同方式选择了同步扫描模式，那么存在两种“等待”的情形：一种是已经完成常规序列转换的 ADC 等待仍在进行常规序列转换的 ADC；另一种是已经完成注入序列转换的 ADC 等待仍在进行注入序列转换的 ADC，然后同步恢复常规序列转换。注意，处于等待状态的 ADC 不属于空闲状态，即 IDLE 状态标志依然为 0。

36.6.2 常规序列的基础转换模式

常规序列的基础转换模式有单次扫描模式、连续扫描模式和断续扫描模式共三种，这些模式可以和双 ADC 协同方式组合使用。

36.6.2.1 常规序列单次扫描模式

在单次扫描模式下，当常规序列转换触发以后，常规序列只转换一轮，然后回到空闲状态。

36.6.2.2 常规序列连续扫描模式

在连续扫描模式下，当常规序列转换触发以后，常规序列会连续转换。如果需要停止常规序列的转换，那么可以通过软件给模拟电路控制寄存器(ANCTRL)寄存器的第 2 位 Adc_stop 写 1 来短暂地重置 ADC 控制器的采样控制器和数据接收器，使状态机回到稳定化状态。

36.6.2.3 常规序列断续扫描模式

在断续扫描模式下，每当常规序列转换触发以后，会转换常规序列的(Short_leng + 1)个位置所对应的通道，然后回到空闲状态；或者如果常规序列中剩余未转换的位置不足(Short_leng + 1)个，那么会把剩余未转换的位置都转换完，然后回到空闲状态。例如，常规序列有 8 个位置，常规短序列有 3 个位置，那么第一次触发将转换位置 0、1、2 所对应的通道，第二次触发将转换位置 3、4、5 所对应的通道，第三次触发将转换位置 6、7 所对应的通道。

36.6.3 注入序列的基础转换模式

注入序列的基础转换模式只有单次扫描模式一种。在单次扫描模式下，当注入序列转换触发以后，注入序列只转换一轮，然后回到空闲状态。

36.6.4 常规序列的双 ADC 协同模式

常规序列的双 ADC 协同模式有同步扫描模式（又称为同时模式或并行模式）和延迟扫描模式（又称为交替模式或跟随模式）两种。默认情况下两个 ADC 独立工作。

当常规序列选择双 ADC 协同模式的其中一种时，两个 ADC 控制器的常规触发信号都将选用 ADC 控制器 0 的，两个 ADC 的时钟也都将选用 ADC 控制器 0 所产生的。

36.6.4.1 双 ADC 常规序列同步扫描模式

在常规序列同步扫描模式下，如果两个 ADC 都处于空闲状态下，那么当 ADC 控制器 0 接收到常规触发信号后，两个 ADC 将开始同步转换。

如果两个 ADC 控制器的常规序列或（断续扫描模式下）常规短序列不同时结束，那么两个 ADC 控制器会在结束当前序列时互相等待，以确保下一次常规序列或常规短序列的转换仍然同步开始。

注意：如果在 ADC 控制器 0 的常规触发信号到来时，两个 ADC 不全都处于空闲状态下，那么两个 ADC 将先后开始第一轮转换，在经过一轮等待过程之后，才能实现同步开始转换。

36.6.4.2 双 ADC 常规序列延迟扫描模式

在常规序列延迟扫描模式下，当 ADC 控制器 0 接收到常规触发信号后，ADC0 将首先开始转换，经过 6~21 个 ADCCLK 周期的延迟后，ADC1 接着开始转换。如果延迟时间不超过 10 个 ADCCLK 周期，那么 ADC1 的转换看上去就像整体相对 ADC0 的转换延迟了。

如果延迟时间多于 10 个 ADCCLK 周期，那么除了 ADC0 的第一次转换以外，自身 ADC 的每次开始转换时刻都相对另一个 ADC 的开始转换时刻有延迟。

另外，常规序列延迟连续扫描模式是唯一没有常规序列结束等待过程的协同模式，常规序列延迟单次或断续扫描模式仍有结束等待的过程。

36.6.5 注入序列的双 ADC 协同模式

注入序列的双 ADC 协同模式有并行扫描模式和轮流触发扫描模式（又称为交替触发模式）两种。默认情况下两个 ADC 独立工作。

当注入序列选择双 ADC 协同模式的其中一种时，两个 ADC 控制器的注入触发信号都将选用 ADC 控制器 0 的，两个 ADC 的时钟也都将选用 ADC 控制器 0 所产生的。

36.6.5.1 双 ADC 注入序列并行扫描模式

在注入序列并行扫描模式下，当 ADC 控制器 0 接收到注入触发信号时，如果两个 ADC 正在进行常规转换，那么它们将各自在完成最近一次常规转换后开始注入并行转换；如果两个 ADC 处于空

闲状态，那么它们将直接进入注入状态，开始注入并行转换。

假设在常规序列独立连续转换、注入序列并行转换的部分过程，其中因为两个 ADC 的常规序列转换独立，所以它们的 ADCCLK 计数器的有可能步伐不同，开始注入序列并行转换的时刻也不同，不过它们在结束注入序列转换时仍然有一个等待的过程。常规序列延迟连续转换、注入序列并行转换的情况也类似。

然而，如果常规序列同步连续转换、注入序列并行转换，那么因为两个 ADC 的 ADCCLK 计数器的步伐一致，所以开始注入序列并行转换的时刻也相同。

36.6.5.2 双 ADC 注入序列轮流触发扫描模式

在注入序列轮流触发扫描模式下，当 ADC 控制器 0 接收到注入触发信号时，两个 ADC 中只有一个开始转换注入序列，而另一个则进入注入状态等待，然后两个 ADC 同时恢复之前的状态。第 1、3、5、…（奇数）次注入触发信号将使 ADC0 转换注入序列；第 2、4、6、…（偶数）次注入触发信号将使 ADC1 转换注入序列。

注意：在轮流触发模式下，只允许在前一个 ADC 完成注入转换后，才触发后一个 ADC 的注入转换。若前一个 ADC 注入转换未完成时有新的触发信号到来，新来的触发信号将被忽略。

36.7 寄存器描述

ADCC0 寄存器基地址：0x4700_6000

ADCC1 寄存器基地址：0x4700_7000

寄存器列表如下：

表 36-3：ADC 内存映射寄存器总表

偏移地址	名称	复位值	描述
0x0	ADC_ANCTRL	0x0	模拟电路控制寄存器
0x4	ADC_DGCTRL	0x0	数字电路控制寄存器
0x8	ADC_CLKCTRL	0xA400003	ADC 时钟控制寄存器
0xC	ADC_CHAVGCFG0	0x0	多次平均设置寄存器 0
0x10	ADC_CHAVGCFG1	0x0	多次平均设置寄存器 1
0x14	ADC_RGLCHCFG0	0x0	常规序列通道设置寄存器 0
0x18	ADC_RGLCHCFG1	0x0	常规序列通道设置寄存器 1
0x1C	ADC_RGLCHCFG2	0x0	常规序列通道设置寄存器 2
0x20	ADC_RGLCHCFG3	0x0	常规序列通道设置寄存器 3
0x24	ADC_INJCHCFG	0x0	注入序列通道设置寄存器
0x28	ADC_CHDAT0	0x0	通道数据寄存器 0
0x2C	ADC_CHDAT1	0x0	通道数据寄存器 1
0x30	ADC_CHDAT2	0x0	通道数据寄存器 2
0x34	ADC_CHDAT3	0x0	通道数据寄存器 3
0x38	ADC_CHDAT4	0x0	通道数据寄存器 4
0x3C	ADC_CHDAT5	0x0	通道数据寄存器 5
0x40	ADC_CHDAT6	0x0	通道数据寄存器 6

偏移地址	名称	复位值	描述
0x44	ADC_CHDAT7	0x0	通道数据寄存器 7
0x48	ADC_CHDAT8	0x0	通道数据寄存器 8
0x4C	ADC_CHDAT9	0x0	通道数据寄存器 9
0x50	ADC_CHDAT10	0x0	通道数据寄存器 10
0x54	ADC_CHDAT11	0x0	通道数据寄存器 11
0x58	ADC_CHDAT12	0x0	通道数据寄存器 12
0x5C	ADC_CHDAT13	0x0	通道数据寄存器 13
0x60	ADC_CHDAT14	0x0	通道数据寄存器 14
0x64	ADC_CHDAT15	0x0	通道数据寄存器 15
0x68	ADC_CHDAT16	0x0	通道数据寄存器 16
0x6C	ADC_DUALDAT	0x0	双数据影子寄存器
0x70	ADC_FIFO_OUT	0x0	接收器 FIFO 数据寄存器
0x74	ADC_WDGEN	0x0	看门狗使能寄存器
0x78	ADC_WDGCOND	0xFFF0000	看门狗比较条件寄存器
0x7C	ADC_STAT	ADC0：0x3000000 ADC1：0x2000000	当前状态寄存器
0x80	ADC_INTSTAT	0x0	中断状态/清除寄存器
0x84	ADC_INTEN	0x0	中断使能寄存器
0x88	ADC_MINTSTAT	0x0	使能后中断状态影子寄存器

以下各节详细介绍寄存器。

36.7.1 模拟电路控制寄存器（ADC_ANCTRL）

偏移地址：0x00

复位值：0x000 0000

位	名称	属性	复位值	描述
32:15	RSV	-	-	保留
14	OPAMP_EN	R/W	0x0	ADC 与 OPA 的连接使能： 说明：ADC 控制器 0 的此寄存器对应 OPA0，ADC 控制器 1 对应 OPA1。（OPA 正通道选择信号（SELP[2:0]）要选择从 ADC MUX 输出连接过来，详细使用方法请参考使用流程“ADC 经 OPA 缓冲采样”）。 0：断开连接 1：保持连接
13	DIFF_IN14_EN	R/W	0x0	通道 14 和 15 差分输入使能： 0：单端输入 1：差分输入
12	DIFF_IN12_EN	R/W	0x0	通道 12 和 13 差分输入使能： 0：单端输入 1：差分输入
11	DIFF_IN10_EN	R/W	0x0	通道 10 和 11 差分输入使能： 0：单端输入 1：差分输入

位	名称	属性	复位值	描述
10	DIFF_IN8_EN	R/W	0x0	通道 8 和 9 差分输入使能： 0：单端输入 1：差分输入
9	DIFF_IN6_EN	R/W	0x0	通道 6 和 7 差分输入使能： 0：单端输入 1：差分输入
8	DIFF_IIN4_EN	R/W	0x0	通道 4 和 5 差分输入使能： 0：单端输入 1：差分输入
7	DIFF_IN2_EN	R/W	0x0	通道 2 和 3 差分输入使能： 0：单端输入 1：差分输入
6	DIFF_IN0_EN	R/W	0x0	通道 0 和 1 差分输入使能： 0：单端输入 1：差分输入
5:4	VREFP_SEL	R/W	0x0	ADC 正参考电压选择 0 或 1：模拟电路的电源 V_{DDA} 2：外部输入的参考电压 V_{REFP} 或者内部的 VREF 模块提供 3：保留
3	POWER_ON	R/W	0x0	ADC 上电控制： 模拟 ADC 上电后需要至少 32 个 ADCCLK 周期的稳定化过程。 0：掉电 1：上电
2	ADC_STOP	W1C	0x0	转换停止控制： 在任何状态下写 1 将使 ADC 控制器的采样器和接收器恢复初始值，并进入空闲状态。采样器停止向模拟 ADC 发送 adc_soc 信号，接收器不再接收新来的数据，除非软件或硬件启动新的序列转换。此位不影响 ADCCLK 的产生，也不影响寄存器或 RXFIFO。 0：写入无效 1：停止转换
1	INJ_START	W1C	0x0	注入序列转换启动控制： 在稳定化状态、空闲状态或常规状态下，写 1 将在本次转换结束后启动注入序列转换；在注入状态下写 1 无效。 0：写入无效 1：软件启动注入序列转换
0	RGL_START	W1C	0x0	常规序列转换启动控制： 在稳定化状态或空闲状态下，写 1 将立即启动常规序列转换；在常规状态或注入状态下写 1 无效。 0：写入无效 1：软件启动常规序列转换

36.7.2 数字电路控制寄存器（ADC_DGCTRL）

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:29	RSV	-	-	保留
28	COMPLEMNT_EN	R/W	0x0	保存在寄存器和 RXFIFO 中的 ADC 转换数据取补码使能。看门狗比较原码与阈值，不受此位的影响： 0：保存数据原码 1：保存数据补码
27:26	DMA_MOD	R/W	0x0	DMA 接收模式选择： 0 或 3：禁用 DMA 接收 1：DMA 接收模式 1：当 FIFO 中保存的（常规序列和注入序列混合）数据达到由 Watermark 设定的数量时会发出 DMA 请求。 2：DMA 接收模式 2：当自身 ADC 控制器接收到数据时，会发出 DMA 请求，可读取通道数据寄存器 0~15（CHDAT0~15）或双数据影子寄存器（DUALDAT）的数据。
25:24	WATERMARK	R/W	0x0	RXFIFO 数据可用触发数量选择 当 RXFIFO 中数据达到一定数量时，将触发 FIFO_AVL 置 1，且触发 DMA 传输： 0：RXFIFO 中有至少 1 个数据 1：RXFIFO 中有至少 4 个数据 2：RXFIFO 中有至少 8 个数据 3：RXFIFO 中有至少 16 个数据
23	FIFO_EN	R/W	0x0	RXFIFO 使能： 0：禁用 RXFIFO，清除数据 1：启用 RXFIFO
22	DATA_R_CLR	R/W	0x0	通道数据读取自动清除使能： 0：读取通道数据寄存器后不会清除数据 1：读取通道数据寄存器后将自动清除数据
21:20	INJ_TRIG_EDG	R/W	0x0	注入序列转换触发信号边沿选择： 0：禁止信号触发 1：上升沿触发 2：下降沿触发 3：双边沿触发
19:16	INJ_TRIG_SEL	R/W	0x0	注入序列转换触发信号源选择： 0x0：定时器 0 的 CC4 事件 0x1：定时器 0 的 TRGO 事件 0x2：定时器 1 的 CC1 事件 0x3：定时器 1 的 TRGO 事件 0x4：定时器 2 的 CC2 事件 0x5：定时器 2 的 CC4 事件 0x6：定时器 3 的 CC1 事件 0x7：定时器 3 的 CC2 事件 0x8：定时器 3 的 CC3 事件 0x9：定时器 3 的 TRGO 事件

位	名称	属性	复位值	描述
				0xA：定时器 4 的 CC4 事件 0xB：定时器 4 的 TRGO 事件 0xC：定时器 7 的 CC2 事件 0xD：定时器 7 的 CC3 事件 0xE：定时器 7 的 CC4 事件 0xF：外部输入端口 15
15:14	RGL_TRIG_EDG	R/W	0x0	常规序列转换触发信号边沿选择： 0：禁止信号触发 1：上升沿触发 2：下降沿触发 3：双边沿触发
13:10	RGL_TRIG_SEL	R/W	0x0	常规序列转换触发信号源选择： 0x0：定时器 0 的 CC1 事件 0x1：定时器 0 的 CC2 事件 0x2：定时器 0 的 CC3 事件 0x3：定时器 1 的 CC2 事件 0x4：定时器 1 的 CC3 事件 0x5：定时器 1 的 CC4 事件 0x6：定时器 1 的 TRGO 事件 0x7：定时器 2 的 CC1 事件 0x8：定时器 2 的 TRGO 事件 0x9：定时器 3 的 CC4 事件 0xA：定时器 4 的 CC1 事件 0xB：定时器 4 的 CC2 事件 0xC：定时器 4 的 CC3 事件 0xD：定时器 7 的 CC1 事件 0xE：定时器 7 的 TRGO 事件 0xF：外部输入端口 11
9	RGL_RCV_SEL	R/W	0x0	常规序列被注入序列打断后恢复扫描时的位置选择： 0：从被打断的位置开始恢复扫描 1：从头开始重新扫描
8:7	DUAL_INJ_MOD	R/W	0x0	双 ADC 协同工作时注入序列转换协同方式选择： 0 或 3：注入序列独立扫描模式 1：注入序列并行扫描模式 2：注入序列轮流触发扫描模式 注意：如果配置成注入序列并行扫描模式或者轮流触发扫描模式，那么 ADC1 将永远使用与 ADC0 相同的时钟，即 ADC1 的时钟由 ADC 控制器 0 的 CLKCTRL 寄存器所选择和生成，而 ADC 控制器 1 的 CLKCTRL 寄存器中的时钟选择和时钟分频设置则不起作用。
6:5	DUAL_RGL_MOD	R/W	0x0	双 ADC 协同工作时常规序列转换协同方式选择： 0 或 3：常规序列独立扫描模式 1：常规序列同步扫描模式 2：常规序列延迟扫描模式 注意：如果配置成常规序列同步扫描模式或者延迟扫描模式，那么 ADC1 将永远使用与 ADC0 相同的时钟，即 ADC1 的时钟由 ADC 控制器 0 的 CLKCTRL 寄存器所选择和生成，而

位	名称	属性	复位值	描述
				ADC 控制器 1 的 CLKCTRL 寄存器中的时钟选择和时钟分频设置则不起作用。
4	INJ_MOD	R/W	0x0	注入序列转换模式选择： 0：不转换 1：单次触发扫描模式
3:2	RGL_MOD	R/W	0x0	常规序列转换模式选择： 0：不转换 1：单次扫描模式 2：连续扫描模式 3：断续扫描模式
1	RD_EDGE	R/W	0x0	触发 ADC 控制器读取转换结果的 ADC_EOC 边沿选择： 0：ADC_EOC 上升沿 1：ADC_EOC 下降沿
0	ADCC_EN	R/W	0x0	ADC 控制器使能： 此位控制时钟生成器、采样控制器和数据接收器。注意：RXFIFO 数据通过此寄存器第 23 位 FIFO_EN 清除。 0：关闭 ADC 控制器 1：启用 ADC 控制器

36.7.3 ADCx 时钟控制寄存器（ADC_CLKCTRL）

偏移地址：0x08

复位值：0x0A40 0003

位	名称	属性	复位值	描述
31:28	SYNC_DELAY	R/W	0x0	双 ADC 协同模式中的常规序列延迟模式下，自身 ADC 相对于上一个 ADC 启动转换的延迟时间设置，此设置在常规序列的其他工作模式下不起作用 例如，在 ADC 控制器 0 中的此位表示 ADC0 的 ADC_SOC 信号置位相对于 ADC1 的延时，在 ADC 控制器 1 中的此位表示 ADC1 的 ADC_SOC 信号置位相对于 ADC0 的延时。 为了错开 4 个 ADCCLK 周期的采样时间，延迟时间为（Sync_delay + 6）个 ADCCLK 周期，延迟时间范围：6~21 个 ADCCLK 周期。
27:24	CH_SWITCH	R/W	0xA	通道选择切换的时间点： 当 ADCCLK 计数器数到多少时切换通道。 注意：1)请勿设为 0 – 3，因为当 ADCCLK 计数器数到 0 – 3 时，ADC 正在采样，此时禁止切换通道。 2)差分输入时，此寄存器建议设置 0xE 或 0xF

位	名称	属性	复位值	描述
23:22	SOC_WIDTH	R/W	0x1	ADC_SOC 信号宽度： ADC_SOC 信号持续高电平的时间是从 ADCCLK 的下降沿到第(Soc_width + 2)个 ADCCLK 的上升沿。 0：1.5 个 ADCCLK 周期 1：2.5 个 ADCCLK 周期 2：3.5 个 ADCCLK 周期 3：4.5 个 ADCCLK 周期
21:17	STABL_TIME	R/W	0x0	ADC 稳定化时间配置： ADC 稳定化时间 = (Stabl_time + 32)个 ADCCLK 周期，其范围是 32~63。
16	ADCCLK_SEL	R/W	0x0	ADC 时钟源选择： 0：PCLK 的分频时钟 1：经过同步的外部时钟（仅支持 XTH 时钟）
15:0	PCLK_DIV	R/W	0x3	PCLK 的分频系数设置： 实际生效的分频系数是(Pclk_div + 1)，其范围是 1~65536。 注意：1. 若 f_{PCLK} = 168 MHz，则实际分频系数应该大于或等于 2，即应该设置 PCLK_DIV ≥ 1。 2. 对于不同的 PCLK 频率，ADC 独立扫描模式和双 ADC 无注入序列的常规序列延迟连续扫描模式最快支持 1 分频；其他双 ADC 协同模式最快支持 2 分频，即应该设置 Pclk_div ≥ 1。

说明：ADC0 总是使用由 ADC 控制器 0 生成的 ADCCLK 时钟。ADC1 在独立工作模式下使用由 ADC 控制器 1 生成的 ADCCLK 时钟，但在双 ADC 协同工作模式下使用由 ADC 控制器 0 生成的 ADCCLK 时钟。

注：在双 ADC 协同工作模式下，ADC 控制器 1 仍需正确配置此寄存器的[31:17]位。

36.7.4 多次平均设置寄存器 0（ADC_CHAVGCFG0）

偏移地址：0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:27	CH9_AVG_SEL	R/W	0x0	通道 9 转换次数设置 （参考通道 0 转换次数设置）
26:24	CH8_AVG_SEL	R/W	0x0	通道 8 转换次数设置 （参考通道 0 转换次数设置）
23:21	CH7_AVG_SEL	R/W	0x0	通道 7 转换次数设置 （参考通道 0 转换次数设置）
20:18	CH6_AVG_SEL	R/W	0x0	通道 6 转换次数设置 （参考通道 0 转换次数设置）
17:15	CH5_AVG_SEL	R/W	0x0	通道 5 转换次数设置 （参考通道 0 转换次数设置）
14:12	CH4_AVG_SEL	R/W	0x0	通道 4 转换次数设置 （参考通道 0 转换次数设置）

位	名称	属性	复位值	描述
11:9	CH3_AVG_SEL	R/W	0x0	通道 3 转换次数设置 （参考通道 0 转换次数设置）
8:6	CH2_AVG_SEL	R/W	0x0	通道 2 转换次数设置 （参考通道 0 转换次数设置）
5:3	CH1_AVG_SEL	R/W	0x0	通道 1 转换次数设置 （参考通道 0 转换次数设置）
2:0	CH0_AVG_SEL	R/W	0x0	通道 0 转换次数设置： 0：1 次转换 1：2 次转换，结果取平均值 2：4 次转换，结果取平均值 3：8 次转换，结果取平均值 4：16 次转换，结果取平均值 5：32 次转换，结果取平均值 6：64 次转换，结果取平均值 7：128 次转换，结果取平均值

36.7.5 多次平均设置寄存器 1（ADC_CHAVGCFG1）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:21	RSV	-	-	保留
20:18	CH16_AVG_SEL	R/W	0x0	通道 16 转换次数设置 （参考通道 0 转换次数设置）
17:15	CH15_AVG_SEL	R/W	0x0	通道 15 转换次数设置 （参考通道 0 转换次数设置）
14:12	CH14_AVG_SEL	R/W	0x0	通道 14 转换次数设置 （参考通道 0 转换次数设置）
11:9	CH13_AVG_SEL	R/W	0x0	通道 13 转换次数设置 （参考通道 0 转换次数设置）
8:6	CH12_AVG_SEL	R/W	0x0	通道 12 转换次数设置 （参考通道 0 转换次数设置）
5:3	CH11_AVG_SEL	R/W	0x0	通道 11 转换次数设置 （参考通道 0 转换次数设置）
2:0	CH10_AVG_SEL	R/W	0x0	通道 10 转换次数设置 （参考通道 0 转换次数设置）

36.7.6 常规序列通道设置寄存器 0（ADC_RGLCHCFG0）

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:25	RGL_CH6_SEL	R/W	0x0	常规序列第 6 位转换通道选择 （参考第 1 位转换通道选择）

位	名称	属性	复位值	描述
24:20	RGL_CH5_SEL	R/W	0x0	常规序列第 5 位转换通道选择 （参考第 1 位转换通道选择）
19:15	RGL_CH4_SEL	R/W	0x0	常规序列第 4 位转换通道选择 （参考第 1 位转换通道选择）
14:10	RGL_CH3_SEL	R/W	0x0	常规序列第 3 位转换通道选择 （参考第 1 位转换通道选择）
9:5	RGL_CH2_SEL	R/W	0x0	常规序列第 2 位转换通道选择 （参考第 1 位转换通道选择）
4:0	RGL_CH1_SEL	R/W	0x0	常规序列第 1 位转换通道选择： 0x0：外部通道 0 0x1：外部通道 1 0x2：外部通道 2 0x3：外部通道 3 0x4：外部通道 4 0x5：外部通道 5 0x6：外部通道 6 0x7：外部通道 7 0x8：外部通道 8 0x9：外部通道 9 0xA：外部通道 10 0xB：外部通道 11 0xC：外部通道 12 0xD：外部通道 13 0xE：外部通道 14 0xF：外部通道 15 0x10~0x1F：保留

36.7.7 常规序列通道设置寄存器 1（ADC_RGLCHCFG1）

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:25	RGL_CH12_SEL	R/W	0x0	常规序列第 12 位转换通道选择 （参考第 1 位转换通道选择）
24:20	RGL_CH11_SEL	R/W	0x0	常规序列第 11 位转换通道选择 （参考第 1 位转换通道选择）
19:15	RGL_CH10_SEL	R/W	0x0	常规序列第 10 位转换通道选择 （参考第 1 位转换通道选择）
14:10	RGL_CH9_SEL	R/W	0x0	常规序列第 9 位转换通道选择 （参考第 1 位转换通道选择）
9:5	RGL_CH8_SEL	R/W	0x0	常规序列第 8 位转换通道选择 （参考第 1 位转换通道选择）
4:0	RGL_CH7_SEL	R/W	0x0	常规序列第 7 位转换通道选择 （参考第 1 位转换通道选择）

36.7.8 常规序列通道设置寄存器 2（ADC_RGLCHCFG2）

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29:25	RGL_CH18_SEL	R/W	0x0	常规序列第 18 位转换通道选择 （参考第 1 位转换通道选择）
24:20	RGL_CH17_SEL	R/W	0x0	常规序列第 17 位转换通道选择 （参考第 1 位转换通道选择）
19:15	RGL_CH16_SEL	R/W	0x0	常规序列第 16 位转换通道选择 （参考第 1 位转换通道选择）
14:10	RGL_CH15_SEL	R/W	0x0	常规序列第 15 位转换通道选择 （参考第 1 位转换通道选择）
9:5	RGL_CH14_SEL	R/W	0x0	常规序列第 14 位转换通道选择 （参考第 1 位转换通道选择）
4:0	RGL_CH13_SEL	R/W	0x0	常规序列第 13 位转换通道选择 （参考第 1 位转换通道选择）

36.7.9 常规序列通道设置寄存器 3（ADC_RGLCHCFG3）

偏移地址：0x20

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:18	RSV	-	-	保留
17:15	SHORT_LENG	R/W	0x0	在断续扫描模式下，常规短序列中的通道转换位置数设置： 常规短序列通道位置数为(Short_leng + 1)。 常规短序列最多包含 8 个通道转换的位置，即 Short_leng 的范围是 0~7。
14:10	RGL_LENG	R/W	0x0	常规序列中的通道转换位置数设置： 常规序列通道位置数为(Rgl_leng + 1)。 常规序列最多包含 20 个通道转换的位置，即 Rgl_leng 的正确范围是 0~19。注意：软件应该避免配置 Rgl_leng > 19，否则每个超出的位置都是通道 0，且只转换一次。
9:5	RGL_CH20_SEL	R/W	0x0	常规序列第 20 位转换通道选择 （参考第 1 位转换通道选择）
4:0	RGL_CH19_SEL	R/W	0x0	常规序列第 19 位转换通道选择 （参考第 1 位转换通道选择）

36.7.10 注入序列通道设置寄存器（ADC_INJCHCFG）

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:22	RSV	-	-	保留
21:20	INJ_LENG	R/W	0x0	注入序列中的通道转换位置数设置 注入序列通道转换位置数为(Inj_leng + 1)。 注入序列最多包含 4 个通道转换的位置。
19:15	INJ_CH4_SEL	R/W	0x0	注入序列第 4 位转换通道选择 （参考第 1 位转换通道选择）
14:10	INJ_CH3_SEL	R/W	0x0	注入序列第 3 位转换通道选择 （参考第 1 位转换通道选择）
9:5	INJ_CH2_SEL	R/W	0x0	注入序列第 2 位转换通道选择 （参考第 1 位转换通道选择）
4:0	INJ_CH1_SEL	R/W	0x0	注入序列第 1 位转换通道选择： 0x0：外部通道 0 0x1：外部通道 1 0x2：外部通道 2 0x3：外部通道 3 0x4：外部通道 4 0x5：外部通道 5 0x6：外部通道 6 0x7：外部通道 7 0x8：外部通道 8 0x9：外部通道 9 0xA：外部通道 10 0xB：外部通道 11 0xC：外部通道 12 0xD：外部通道 13 0xE：外部通道 14 0xF：外部通道 15 0x10~0x1F：保留

36.7.11 通道数据寄存器 0~16（ADC_CHDAT0~16）

偏移地址：0x28~0x68

复位值：0000 0000

位	名称	属性	复位值	描述
31:17	RSV	-	-	保留
16	DATA_VALID	R	0x0	数据有效信号： 在获取有效数据后，此信号激活。当 Adc_en = 0 时，或在软件读取此寄存器后，由硬件清除。 0：数据无效 1：数据有效
15:12	RSV	-	-	保留
11:0	CH_DATA	R	0x0	通道数据： 若 Dat_r_clr = 1，则会在软件读取此寄存器后，由硬件清除。

注：寄存器 0~15 对应 16 个外部输入通道数据。对于双端口差分输入，端口 0 和 1 的差分数据会保存在寄存器 0，端口 2 和 3 的差分数据会保存在寄存器 2，以此类推。

36.7.12 双数据影子寄存器（ADC_DUALDAT）

偏移地址：0x6C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:28	ADC1_LAST_CH_NUM	R	0x0	ADC1 最近转换的通道编号
27:16	ADC1_DATA	R	0x0	ADC1 最近转换的结果
15:12	ADC0_LAST_CH_NUM	R	0x0	ADC0 最近转换的通道编号
11:0	ADC0_DATA	R	0x0	ADC0 最近转换的结果

注：

- 两个 ADC 最近一次转换的通道编号和结果会同时映射到此寄存器地址，这将有助于在双 ADC 协同工作模式下读取数据。
- 在 DMA 模式 2 中，只要自身 ADC 准备好数据（有效），就会发出 DMA 请求，但如果此时另一个 ADC 没有准备好数据，那么另一个 ADC 的通道编号表示正在准备的通道编号，而数据是正在准备的通道的上一条数据（无效）。因此，为了使两个 ADC 同时准备好数据，可以将两个 ADC 的每个通道的转换次数都需要设为相同值，而且通过等待 IDLE_STATE 状态标志置位之后再触发转换，使两个 ADC 同时开始转换。当然，因为两个 ADC 控制器分别发送 DMA 请求，每次只读取自身的有效数据也可行。

36.7.13 接收器 FIFO 数据寄存器（ADC_FIFO_OUT）

偏移地址：0x70

复位值：0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:12	LAST_CH_NUM	R	0x0	通道编号
11:0	FIFO_OUT	R	0x0	RXFIFO 数据

36.7.14 看门狗使能寄存器（ADC_WDGEN）

偏移地址：0x74

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15	WDG_EN_15	R/W	0x0	通道 15 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
14	WDG_EN_14	R/W	0x0	通道 14 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能

位	名称	属性	复位值	描述
13	WDG_EN_13	R/W	0x0	通道 13 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
12	WDG_EN_12	R/W	0x0	通道 12 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
11	WDG_EN_11	R/W	0x0	通道 11 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
10	WDG_EN_10	R/W	0x0	通道 10 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
9	WDG_EN_9	R/W	0x0	通道 9 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
8	WDG_EN_8	R/W	0x0	通道 8 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
7	WDG_EN_7	R/W	0x0	通道 7 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
6	WDG_EN_6	R/W	0x0	通道 6 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
5	WDG_EN_5	R/W	0x0	通道 5 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
4	WDG_EN_4	R/W	0x0	通道 4 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
3	WDG_EN_3	R/W	0x0	通道 3 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
2	WDG_EN_2	R/W	0x0	通道 2 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
1	WDG_EN_1	R/W	0x0	通道 1 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能
0	WDG_EN_0	R/W	0x0	通道 0 看门狗使能： 0：关闭看门狗功能 1：开启看门狗功能

36.7.15 看门狗比较条件寄存器（ADC_WDGCOND）

偏移地址：0x78

复位值：0x0FFF 0000

位	名称	属性	复位值	描述
31	CONDITION	R/W	0x0	看门狗事件触发条件选择： 0：转换数据原码超出阈值范围，即转换数据原码 > H_threshold 或转换数据原码 < L_threshold 时，看门狗报警 1：转换数据原码在阈值范围内，即 L_threshold ≤ 转换数据原码 ≤H_threshold 时，看门狗报警
30:28	RSV	R	0x0	保留
27:16	H_THRESHOLD	R/W	0xFFF	看门狗阈值上限
15:12	RSV	R	0x0	保留
11:0	L_THRESHOLD	R/W	0x0	看门狗阈值下限

注：

1. 如果使用了多次采样模式，那么看门狗仅比较转换数据的平均值与阈值。
2. 除了可以在 ADC 启动之前设置看门狗比较条件，还可以在 ADC 转换的过程中，每当 WDG_CMPLT 标志置 1 时，改变看门狗比较条件，这样每个通道数据的比较条件都可以不同。不过这种做法存在比较条件更新不及时的风险。
3. 若设定成看门狗的复位值，则看门狗不会报警。

36.7.16 当前状态寄存器（ADC_STAT）

偏移地址：0x7C

复位值：ADC0：0x3000000

ADC1：0x2000000

位	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27	WDG_CMPLT	R/W1C	0x0	看门狗比较通道数据完成标志，写 1 清 0 当看门狗完成一个通道数据的比较后，标志会置 1。当标志为 1 时，可以改变看门狗的比较条件，新的比较条件用于下一个通道数据的比较（不区分常规序列或注入序列）。写看门狗比较条件寄存器（WDGCOND）后，此标志也会清 0。在 ADC 运行过程中，当此标志为 0 时，请勿改变看门狗的比较条件，以免比较结果出错。 0：看门狗比较通道数据完成 1：看门狗比较通道数据未完成
26	FIFO_FULL	R	0x0	RXFIFO 满标志： 0：RXFIFO 中不足 32 条数据 1：RXFIFO 中有 32 条数据
25	FIFO_EMPTY	R	0x1	RXFIFO 空标志： 0：RXFIFO 中有数据 1：RXFIFO 中没有数据

位	名称	属性	复位值	描述
24	INJ_IN_TURN	R	ADC0：0x1 ADC1：0x0	轮流触发注入轮次： 此寄存器指示在注入序列轮流触发扫描模式下，正在执行注入转换的是哪个 ADC，或下一次触发注入转换时将轮到哪个 ADC 执行转换。 0：未轮到当前 ADC 1：轮到当前 ADC
23:21	INJ_NUM	R	0x0	注入序列中正在转换的位置编号： 1~4 表示正在转换注入序列中第几个位置的通道，0 表示当前没有转换注入序列。
20:16	RGL_NUM	R	0x0	常规序列中正在转换的位置编号： 1~20 表示正在转换常规序列中第几个位置的通道，0 表示当前没有转换常规序列（包括正在转换注入序列）。
15:11	CH_NUM	R	0x0	正在转换的通道编号： 有效范围：0~15，空闲时显示常规序列第 1 个位置的通道编号（由 Rgl_ch1_sel 设置）
10:4	CONV_CNT	R	0x0	当前转换位置已完成的转换次数： 由于每个通道都可以独立设置用于计算平均值的转换次数，此寄存器的值的有效范围是不大于(Chn_avg_sel 的选定值 - 1)
3:1	RSV	-	-	保留
0	IDLE_STATE	R	0x0	空闲状态标志： 当此标志为 1 时，ADC 能响应常规序列的触发信号。 ADC 复位后，首先进入稳定化状态，过了 32 个 ADC 时钟周期后才进入空闲状态。 在双 ADC 协同模式下，如果两个 ADC 的转换时间长度不同，那么当已完成转换的 ADC 在等待另一个正在转换的 ADC 时，此标志仍保持为 0。

36.7.17 中断状态/清除寄存器（ADC_INTSTAT）

偏移地址：0x80

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6	WDG_INJ	R/W1C	0x0	注入序列看门狗报警中断标志，写 1 清 0 在注入序列的转换过程中，如果最近一次转换的通道开启了看门狗功能，而且其转换结果满足看门狗报警条件，将触发中断。
5	WDG_RGL	R/W1C	0x0	常规序列看门狗报警中断标志，写 1 清 0 在规则序列的转换过程中，如果最近一次转换的通道开启了看门狗功能，而且其转换结果满足看门狗报警条件，将触发中断。
4	WDG_CH	R/W1C	0x0	通道数据看门狗报警中断标志，写 1 清 0 如果最近一次转换的通道开启了看门狗功能，而且其转换结果满足看门狗报警条件，将触发中断。
3	FIFO_OVF	R/W1C	0x0	RXFIFO 溢出中断标志，写 1 清 0

位	名称	属性	复位值	描述
2	FIFO_AVL	R/W1C	0x0	RXFIFO 数据可用中断标志，写 1 清 0 当 RXFIFO 中的数据量达到 DGCTRL 寄存器中 WATERMARK 的设定值后，此标志会置 1。
1	EOIC	R/W1C	0x0	注入序列转换结束中断标志，写 1 清 0
0	EORC	R/W1C	0x0	常规序列转换结束中断标志，写 1 清 0

36.7.18 中断使能寄存器（ADC_INTEN）

偏移地址：0x84

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:7	RSV	R	0x0	保留
6	WDG_INJ_EN	R/W	0x0	注入序列看门狗报警中断使能
5	WDG_RGL_EN	R/W	0x0	常规序列看门狗报警中断使能
4	WDG_CH_EN	R/W	0x0	通道数据看门狗报警中断使能
3	FIFO_OVF_EN	R/W	0x0	RXFIFO 溢出中断使能
2	FIFO_AVL_EN	R/W	0x0	RXFIFO 数据可用中断使能
1	EOIC_EN	R/W	0x0	注入序列转换结束中断使能
0	EORC_EN	R/W	0x0	常规序列转换结束中断使能

36.7.19 使能的中断状态影子寄存器（ADC_MINTSTAT）

偏移地址：0x88

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:7	RSV	-	-	保留
6	WDG_INJ_M	R	0x0	使能的注入序列看门狗报警中断标志
5	WDG_RGL_M	R	0x0	使能的常规序列看门狗报警中断标志
4	WDG_CH_M	R	0x0	使能的通道数据看门狗报警中断标志
3	FIFO_OVF_M	R	0x0	使能的 RXFIFO 溢出中断标志
2	FIFO_AVL_M	R	0x0	使能的 RXFIFO 数据可用中断标志
1	EOIC_M	R	0x0	使能的注入序列转换结束中断标志
0	EORC_M	R	0x0	使能的常规序列转换结束中断标志

注：此寄存器地址映射的是中断状态/清除寄存器（INTSTAT）逻辑“与”中断使能寄存器（INTEN）的结果。若要清除中断标志，请给中断状态/清除寄存器（INTSTAT）中的（原始）中断标志写“1”。

36.8 使用流程

36.8.1 常规序列单次扫描模式多通道 A/D 转换

单次扫描模式下，ADC 启动转换后只执行一次转换。

1. 根据 ADC 输入通道对应的 GPIO 管脚，将待转换通道配置为模拟接口（GPIOx_MODE）。

2. 配置 APB2 外围时钟使能寄存器（APB2CKENR）使能 ADC 时钟。
3. 配置 ADC_ANCTRL[3]为 1，设置 ADC 上电。
4. 配置 ADC_ANCTRL[5:4]，设置 ADC 正参考电压为模拟电路的电源 V_{DDA}。
5. 配置 ADC_DGCTRL[3:2]为 1，设置常规序列转换模式为单次扫描模式。
6. 配置 ADC_DGCTRL[22]为 1，设置通道数据读取后自动清除。
7. 配置 ADC_CLKCTRL[27:24]，设置通道选择切换的时间点。
8. 配置 ADC_CLKCTRL[15:0]，设置 ADC 的采样率。
9. 配置多次平均设置寄存器，设置各通道的转换次数，结果取平均值。
10. 配置常规序列通道设置寄存器，设置各个位置转换的通道，设置常规序列中通道转换的位置数。
11. 配置 ADC_DGCTRL[0]为 1，使能 ADC 控制器。
12. 配置 ADC_ANCTRL[0]为 1， 软件启动常规序列转换。
13. 等待 ADC 常规序列转换完成，读取对应通道的数据寄存器的数据。
14. 如需进行多次转换，则重复执行步骤 12 和 13。

36.8.2 常规序列连续扫描模式多通道 A/D 转换

1. 根据 ADC 输入通道对应的 GPIO 管脚，将待转换通道配置为模拟接口（GPIOx_MODE）。
2. 配置 APB2 外围时钟使能寄存器（APB2CKENR）使能 ADC 时钟。
3. 配置 ADC_ANCTRL[3]为 1，设置 ADC 上电。
4. 配置 ADC_ANCTRL[5:4]，设置 ADC 正参考电压为模拟电路的电源 V_{DDA}。
5. 配置 ADC_DGCTRL[3:2]为 2，设置常规序列转换模式为连续扫描模式。
6. 配置 ADC_DGCTRL[22]为 1，设置通道数据读取后自动清除。
7. 配置 ADC_CLKCTRL[27:24]，设置通道选择切换的时间点。
8. 配置 ADC_CLKCTRL[15:0]，设置 ADC 的采样率。
9. 配置多次平均设置寄存器，设置各通道的转换次数，结果取平均值。
10. 配置常规序列通道设置寄存器，设置各个位置转换的通道，设置常规序列中通道转换的位置数。
11. 配置 ADC_DGCTRL[0]为 1，使能 ADC 控制器。
12. 配置 ADC_ANCTRL[0]为 1，软件启动常规序列转换。
13. 每次等待 ADC 常规序列转换完成，就可以读取对应通道的数据寄存器的数据。

36.8.3 常规短序列断续扫描模式多通道 A/D 转换

1. 根据 ADC 输入通道对应的 GPIO 管脚，将待转换通道配置为模拟接口（GPIOx_MODE）。
2. 配置 APB2 外围时钟使能寄存器（APB2CKENR）使能 ADC 时钟。
3. 配置 ADC_ANCTRL[3]为 1，设置 ADC 上电。

4. 配置 ADC_ANCTRL[5:4]，设置 ADC 正参考电压为模拟电路的电源 V_{DDA}。
5. 配置 ADC_DGCTRL[3:2]为 3，设置常规序列转换模式为断续扫描模式。
6. 配置 ADC_DGCTRL[22]为 1，设置通道数据读取后自动清除。
7. 配置 ADC_CLKCTRL[27:24]，设置通道选择切换的时间点。
8. 配置 ADC_CLKCTRL[15:0]，设置 ADC 的采样率。
9. 配置多次平均设置寄存器，设置各通道的转换次数，结果取平均值。
10. 配置常规序列通道设置寄存器，设置各个位置转换的通道，设置常规序列中通道转换的位置数。
11. 配置 ADC_RGLCHCFG3[17:15]，设置常规短序列的通道转换位置数。
12. 配置 ADC_DGCTRL[0]为 1，使能 ADC 控制器。
13. 配置 ADC_ANCTRL[0]为 1，软件启动常规序列转换，每次转换一个常规短序列。
14. 等待 ADC_ADC 常规序列转换完成，就可以读取对应通道的数据寄存器的数据。
15. 如需进行多次转换，则重复执行步骤 13 和 14，当所有通道位置转换完成，重新从通道位置 0 开始转换。

36.8.4 注入序列单次扫描模式多通道 A/D 转换

1. 根据 ADC 输入通道对应的 GPIO 管脚，将待转换通道配置为模拟接口（GPIOx_MODE）。
2. 配置 APB2 外围时钟使能寄存器（APB2CKENR）使能 ADC 时钟。
3. 配置 ADC_ANCTRL[3]为 1，设置 ADC 上电。
4. 配置 ADC_ANCTRL[5:4]，设置 ADC 正参考电压为模拟电路的电源 V_{DDA}。
5. 配置 ADC_DGCTRL.[4]为 1，设置注入序列转换模式为单次触发扫描模式。
6. 配置 ADC_DGCTRL[22]为 1，设置通道数据读取后自动清除。
7. 配置 ADC_CLKCTRL[27:24]，设置通道选择切换的时间点。
8. 配置 ADC_CLKCTRL[15:0]，设置 ADC 的采样率。
9. 配置多次平均设置寄存器，设置各通道的转换次数，结果取平均值。
10. 配置注入序列通道设置寄存器，设置各个位置转换的通道，设置注入序列中通道转换的位置数。
11. 配置 ADC_DGCTRL[0]为 1，使能 ADC 控制器。
12. 配置 ADC_ANCTRL[1]为 1， 软件启动注入序列转换。
13. 等待 ADC 注入序列转换完成，就可以读取对应通道的数据寄存器的数据。
14. 如需进行多次转换，则重复执行步骤 12 和 13。

36.8.5 模拟看门狗

1. 根据 ADC 输入通道对应的 GPIO 管脚，将待转换通道配置为模拟接口（GPIOx_MODE）。
2. 配置 APB2 外围时钟使能寄存器（APB2CKENR）使能 ADC 时钟。

3. 配置 ADC_ANCTRL[3]为 1，设置 ADC 上电。
4. 配置 ADC_ANCTRL[5:4]，设置 ADC 正参考电压为模拟电路的电源 V_{DDA}。
5. 配置 ADC_DGCTRL[3:2]为 1，设置常规序列转换模式为单次扫描模式。
6. 配置 ADC_DGCTRL[22]为 1，设置通道数据读取后自动清除。
7. 配置 ADC_CLKCTRL[27:24]，设置通道选择切换的时间点。
8. 配置 ADC_CLKCTRL[15:0]，设置 ADC 的采样率。
9. 配置多次平均设置寄存器，设置各通道的转换次数，结果取平均值。
10. 配置常规序列通道设置寄存器，设置各个位置转换的通道，设置常规序列中通道转换的位置数。
11. 配置看门狗比较条件寄存器，设置看门狗阈值上限和看门狗阈值下限。
12. 配置 ADC_DGCTRL[0] 为 1，使 ADC 控制器。
13. 配置 ADC_INTEN[5]为 1，使能常规序列看门狗报警中断。
14. 配置 ADC_ANCTRL[0]为 1，软件启动常规序列转换。
15. 等待 ADC 常规序列转换完成，当转换的结果超出看门狗上下限时会产生看门狗中断。
16. 软件写 1 清除 ADC_INTSTAT[5] 常规序列看门狗报警中断标志，继续下一次转换。

36.8.6 差分通道输入

1. 根据 ADC 输入通道对应的 GPIO 管脚，将待转换通道配置为模拟接口（GPIOx_MODE）。
2. 配置 APB2 外围时钟使能寄存器（APB2CKENR）使能 ADC 时钟。
3. 配置 ADC_ANCTRL[3]为 1，设置 ADC 上电。
4. 配置 ADC_ANCTRL[5:4]，设置 ADC 正参考电压为模拟电路的电源 V_{DDA}。
5. 配置 ADC_DGCTRL[3:2]为 1，设置常规序列转换模式为单次扫描模式。
6. 配置 ADC_DGCTRL[22]为 1，设置通道数据读取后自动清除。
7. 配置 ADC_CLKCTRL[27:24]，设置通道选择切换的时间点。
8. 配置 ADC_CLKCTRL[15:0]，设置 ADC 的采样率。
9. 配置 ADC_ANCTRL 寄存器，使能通道差分输入。
10. 配置多次平均设置寄存器，设置各通道的转换次数，结果取平均值。
11. 配置常规序列通道设置寄存器，设置各个位置转换的通道，设置常规序列中通道转换的位置数。
12. 配置 ADC_DGCTRL[0]为 1，使能 ADC 控制器。
13. 配置 ADC_ANCTRL[0]为 1， 软件启动常规序列转换。
14. 等待 ADC 常规序列转换完成，读取对应通道的差分数据，例如通道 0 和 1 的差分数据会保存在通道数据寄存器 0，以此类推。
15. 如需进行多次转换，则重复执行步骤 12 和 13。

36.8.7 常规序列双 ADC 协同模式

1. 根据 ADC0/1 输入通道对应的 GPIO 管脚，将待转换通道配置为模拟接口（GPIOx_MODE）。
2. 配置 APB2 外围时钟使能寄存器（APB2CKENR）使能 ADC0/1 时钟。
3. 配置 ADC_ANCTRL[3]为 1，设置 ADC0/1 上电。
4. 配置 ADC_ANCTRL[5:4]，设置 ADC0/1 正参考电压为模拟电路的电源 V_{DDA}。
5. 配置 ADC_DGCTRL[3:2]，设置常规序列转换模式。
6. 配置 ADC_DGCTRL[22] 为 1，设置通道数据读取后自动清除。
7. 配置 ADC_CLKCTRL[27:24]，设置通道选择切换的时间点。
8. 配置 ADC_CLKCTRL[15:0]，设置 ADC0 的采样率。
9. 配置多次平均设置寄存器，设置各通道的转换次数，结果取平均值。
10. 配置常规序列通道设置寄存器，设置各个位置转换的通道，设置常规序列中通道转换的位置数。
11. 配置 ADC_DGCTRL[0]为 1，使 ADC0/1 控制器。
12. 配置 ADC_DGCTRL[6:5]，设置 ADC0/1 协同工作时常规序列转换协同方式。
13. 配置 ADC_ANCTRL[0]为 1，软件启动 ADC0 常规序列转换。
14. 等待 ADC0/1 常规序列转换完成，读取 ADC0/1 对应通道的数据寄存器的数据。
15. 如需进行多次转换，则重复执行步骤 13 和 14。

36.8.8 注入序列双 ADC 协同模式

1. 根据 ADC0/1 输入通道对应的 GPIO 管脚，将待转换通道配置为模拟接口（GPIOx_MODE）。
2. 配置 APB2 外围时钟使能寄存器（APB2CKENR）使能 ADC0/1 时钟。
3. 配置 ADC_ANCTRL[3]为 1，设置 ADC0/1 上电。
4. 配置 ADC_ANCTRL[5:4]，设置 ADC0/1 正参考电压为模拟电路的电源 V_{DDA}。
5. 配置 ADC_DGCTRL[4]为 1，设置注入序列转换模式为单次触发扫描模式。
6. 配置 ADC_DGCTRL[22]为 1，设置通道数据读取后自动清除。
7. 配置 ADC_CLKCTRL[27:24]，设置通道选择切换的时间点。
8. 配置 ADC_CLKCTRL[15:0]，设置 ADC0 的采样率。
9. 配置多次平均设置寄存器，设置各通道的转换次数，结果取平均值。
10. 配置注入序列通道设置寄存器，设置各个位置转换的通道，设置注入序列中通道转换的位置数。
11. 配置 ADC_DGCTRL[0] 为 1，使能 ADC0/1 控制器。
12. 配置 ADC_DGCTRL[8:7]，设置 ADC0/1 协同工作时注入序列转换协同方式。
13. 配置 ADC_ANCTRL[1] 为 1，软件启动 ADC0 注入序列转换。
14. 等待 ADC 注入序列转换完成，就可以读取对应通道的数据寄存器的数据。
15. 如需进行多次转换，则重复执行步骤 13 和 14。

36.8.9 ADC 经 OPA 缓冲采样

1. 设置 OPA 为 unintbuffer 模式，OPA 正通道选择信号 SELP 选从 ADC MUX 输出连接过来。
2. 根据 ADC 输入通道对应的 GPIO 管脚，将待转换通道配置为模拟接口（GPIOx_MODE）。
3. 配置 APB2 外围时钟使能寄存器（APB2CKENR）使能 ADC 时钟。
4. 配置 ADC_ANCTRL[3]为 1，设置 ADC 上电。
5. 配置 ADC_ANCTRL[5:4]，设置 ADC 正参考电压为模拟电路的电源 V_{DDA}。
6. 配置 ADC_DGCTRL[3:2]为 1，设置常规序列转换模式为单次扫描模式。
7. 配置 ADC_DGCTRL[22]为 1，设置通道数据读取后自动清除。
8. 配置 ADC_CLKCTRL[27:24]，设置通道选择切换的时间点。
9. 配置 ADC_CLKCTRL[15:0]，设置 ADC 的采样率。
10. 配置多次平均设置寄存器，设置各通道的转换次数，结果取平均值。
11. 配置常规序列通道设置寄存器，设置各个位置转换的通道，设置常规序列中通道转换的位置数。
12. 配置 ADC_ANCTRL[14]为 1，使能 ADC 与 OPA 的连接。
13. 配置 ADC_DGCTRL[0]为 1，使能 ADC 控制器。
14. 配置 ADC_ANCTRL[0]为 1，软件启动常规序列转换。
15. 等待 ADC 常规序列转换完成，读取对应通道的数据寄存器的数据。
16. 如需进行多次转换，则重复执行步骤 14 和 15。

37 数模转换器（DAC）

37.1 概述

DAC 可以控制数模转换器将 12 位数字量转换为模拟电压输出。

37.2 主要特性

- 支持 8/12 位数字输入
- 支持两个转换器同时更新转换数据
- 支持产生三角波和噪声波
- 支持 DMA 操作
- 支持外部触发更新转换数据

37.3 系统框图

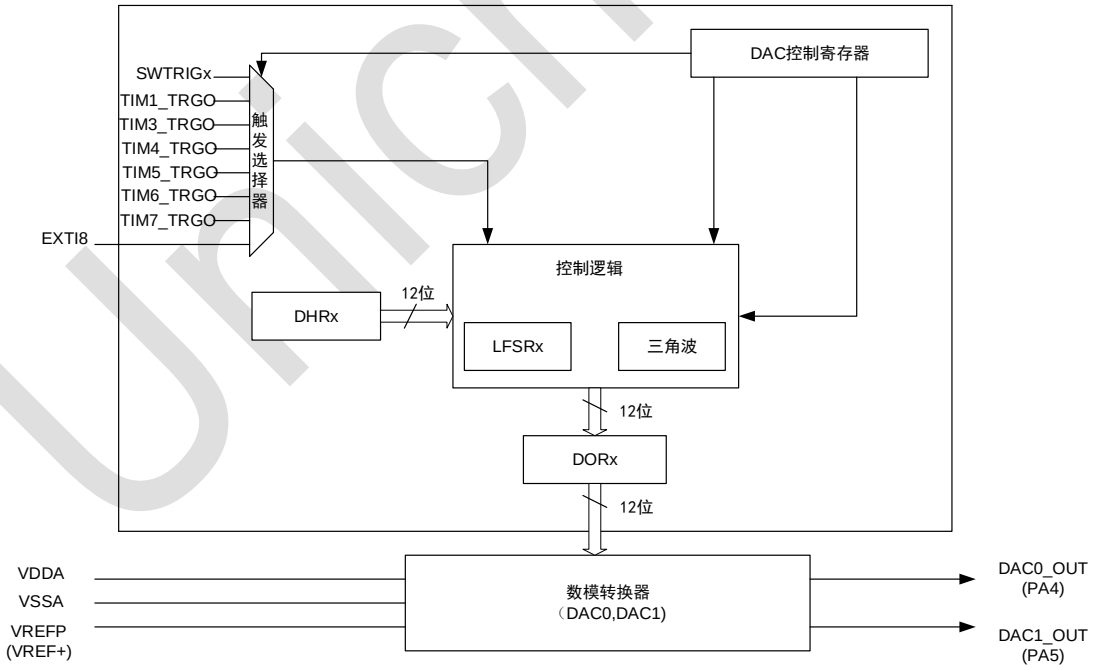


图 37-1：DAC 系统框图

37.4 功能描述

37.4.1 DAC 转换控制

两个 DAC 通道可以使用 DAC_CTRL 寄存器中的 DACENx 位独立开启或关闭。使能后，DAC 会将 DORx 中的数据转换为模拟电压输出。输出电压为 V_{REF}*(DOR/4095)。

通过 DAC_CTRL 寄存器中的 BUFx 位可以开启或关闭 DAC 输出缓冲器，它可以降低输出阻抗并直接驱动一定的外部负载。

DAC 支持 8 位或 12 位数据，当使用 8 位数据时，DORx 的低 4 位会自动加载为 0。

37.4.2 触发模式

DAC 支持 8 种不同的触发源。当开启触发模式时，DHR 中的数据会在触发后加载到 DORx。

37.4.3 噪声波

DAC 可以控制产生噪声波，开启噪声波模式后，LFSR 会按照一定的方法产生噪音波幅值。每当检测到触发事件，就会将 LFSR 值与 DHR 中的值相加后加载到 DORx。可以屏蔽 LFSR 中的高位来控制噪音幅度。

37.4.4 三角波

DAC 可以控制产生三角波，开启三角波模式后，三角波计数器会在 DAC_CTRL 寄存器中的 MAMPx 规定的幅度内生成三角波幅值。每当检测到触发事件，就会将三角波计数器值与 DHR 中的值相加后加载到 DORx。

37.4.5 DMA

当开启 DMA 模式时，DAC 会在接收到硬件触发后产生一个 DMA 请求，写入到 DHR 寄存器中的值会在一个周期后加载到 DORx，如果接收到硬件触发时上一个 DMA 请求仍未得到回应，就会产生下溢中断并关闭 DMA 模式。

37.4.6 双 DAC 配合

通过使用 DAC_DHR12RD、DAC_DHR12LD 或 DAC_DHR8RD 寄存器，两个 DAC 可以同时加载数据。当使用触发模式时，TENx 和 TSELx 仅对对应的 DAC 通道独立有效，可以使用相同的 TSEL 来使他们同步更新。当使用 DMA 模式时，对应通道的 DHR 值会被立即加载，如果要通过

DMA 来同步更新，需要关闭另一个通道的触发模式。

37.5 寄存器描述

DAC 寄存器基地址：0x4600_C000

寄存器列表如下：

表 37-1：DAC 寄存器列表

偏移地址	名称	描述
0x00	DAC_CTRL	控制寄存器
0x04	DAC_SWTRG	软件触发寄存器
0x08	DAC_DHR12R0	DAC0 右对齐 12 位数据寄存器
0x0C	DAC_DHR12L0	DAC0 左对齐 12 位数据寄存器
0x10	DAC_DHR8R0	DAC0 右对齐 8 位数据寄存器
0x14	DAC_DHR12R1	DAC1 右对齐 12 位数据寄存器
0x18	DAC_DHR12L1	DAC1 左对齐 12 位数据寄存器
0x1C	DAC_DHR8R1	DAC1 右对齐 8 位数据寄存器
0x20	DAC_DHR12RD	双 DAC 右对齐 12 位数据寄存器
0x24	DAC_DHR12LD	双 DAC 左对齐 12 位数据寄存器
0x28	DAC_DHR8RD	双 DAC 右对齐 8 位数据寄存器
0x2C	DAC_DOR0	DAC0 输出数据寄存器
0x30	DAC_DOR1	DAC1 输出数据寄存器
0x34	DAC_IS	中断状态寄存器
0x38	DAC_CLK	DAC 时钟设定寄存器

37.5.1 控制寄存器 (DAC_CTRL)

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:30	DAC1_VREF_MODE	R/W	0x0	DAC1 参考电压选择： 00/01：VREFP_AVDD（VDDA） 10：外部输入的参考电压 VREFP 或者内部的 VREF 模块提供 11：保留
29	DMAIE1	R/W	0x0	选择使能 DAC1 DMA 下溢中断： 0：不使能中断 1：使能中断
28	DMAEN1	R/W	0x0	选择使能 DAC1 的 DMA 模式： 0：不使能 DMA 1：使能 DMA

位	名称	属性	复位值	描述
27:24	MAMP1	R/W	0x0	选择 DAC1 噪声模式下的掩码/三角波模式下的振幅： 0000：不屏蔽 LSFR 的位[0]/三角波振幅为 1 0001：不屏蔽 LSFR 的位[1:0]/三角波振幅为 3 0010：不屏蔽 LSFR 的位[2:0]/三角波振幅为 7 0011：不屏蔽 LSFR 的位[3:0]/三角波振幅为 15 0100：不屏蔽 LSFR 的位[4:0]/三角波振幅为 31 0101：不屏蔽 LSFR 的位[5:0]/三角波振幅为 63 0110：不屏蔽 LSFR 的位[6:0]/三角波振幅为 127 0111：不屏蔽 LSFR 的位[7:0]/三角波振幅为 255 1000：不屏蔽 LSFR 的位[8:0]/三角波振幅为 511 1001：不屏蔽 LSFR 的位[9:0]/三角波振幅为 1023 1001：不屏蔽 LSFR 的位[10:0]/三角波振幅为 2047 ≥1011：不屏蔽 LSFR 的位[11:0]/三角波振幅为 4095
23:22	WAVE1	R/W	0x0	选择 DAC1 产生噪声波/三角波： 00：不产生噪声波/三角波 01：产生噪声波 10/11：产生三角波
21:19	TSEL1	R/W	0x0	选择 DAC1 触发源： 000：TIM5 TRGO 001：TIM7 TRGO 010：TIM6 TRGO 011：TIM4 TRGO 100：TIM1 TRGO 101：TIM3 TRGO 110：外部中断引脚 111：软件触发
18	TEN1	R/W	0x0	选择使能 DAC1 触发模式： 0：不使能触发，写入 DACDHR 的数据在一个周期后开始转换输出 1：使能触发，写入 DACDHR 的数据在触发后的 3 个（硬件触发）/1 个（软件触发）周期后开始转换输出
17	BUF1	R/W	0x0	选择使能 DAC1 输出缓冲器： 0：不使能 1：使能
16	DACEN1	R/W	0	选择使能 DAC1： 0：不使能 1：使能，切换到使能后需要时间上电

位	名称	属性	复位值	描述
15:14	DAC0_VREF_MODE	R/W	0	DAC0 参考电压选择： 00/01：VREFP_AVDD（VDDA） 10：外部输入的参考电压 VREFP 或者内部的 VREF 模块提供 11：保留
13	DMAIE0	R/W	0	选择使能 DAC0 DMA 下溢中断： 0：不使能中断 1：使能中断
12	DMAEN0	R/W	0	选择使能 DAC0 的 DMA 模式： 0：不使能 DMA 1：使能 DMA
11:8	MAMP0	R/W	0	选择 DAC0 噪声模式下的掩码/三角波模式下的振幅： 0000：不屏蔽 LSFR 的位[0]/三角波振幅为 1 0001：不屏蔽 LSFR 的位[1:0]/三角波振幅为 3 0010：不屏蔽 LSFR 的位[2:0]/三角波振幅为 7 0011：不屏蔽 LSFR 的位[3:0]/三角波振幅为 15 0100：不屏蔽 LSFR 的位[4:0]/三角波振幅为 31 0101：不屏蔽 LSFR 的位[5:0]/三角波振幅为 63 0110：不屏蔽 LSFR 的位[6:0]/三角波振幅为 127 0111：不屏蔽 LSFR 的位[7:0]/三角波振幅为 255 1000：不屏蔽 LSFR 的位[8:0]/三角波振幅为 511 1001：不屏蔽 LSFR 的位[9:0]/三角波振幅为 1023 1001：不屏蔽 LSFR 的位[10:0]/三角波振幅为 2047 ≥1011：不屏蔽 LSFR 的位[11:0]/三角波振幅为 4095
7:6	WAVE0	R/W	0	选择 DAC0 产生噪声波/三角波： 00：不产生噪声波/三角波 01：产生噪声波 10/11：产生三角波
5:3	TSEL0	R/W	0	选择 DAC0 触发源： 000：TIM5 TRGO 001：TIM7 TRGO 010：TIM6 TRGO 011：TIM4 TRGO 100：TIM1 TRGO 101：TIM3 TRGO 110：外部中断引脚 111：软件触发

位	名称	属性	复位值	描述
2	TEN0	R/W	0	选择使能 DAC0 触发模式： 0：不使能触发，写入 DACDHR 的数据在一个周期后开始转换输出 1：使能触发，写入 DACDHR 的数据在触发后的 3 个（硬件触发）/1 个（软件触发）周期后开始转换输出
1	BUF0	R/W	0	选择使能 DAC0 输出缓冲器： 0：不使能 1：使能
0	DACEN0	R/W	0	选择使能 DAC0： 0：不使能 1：使能，切换到使能后需要时间上电

37.5.2 软件触发寄存器 (DAC_SWTRG)

偏移地址：0x04

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:2	RSV	-	-	保留
1	SWTRG1	W	0x0	对此位写入 1 即可产生对 DAC1 的软件触发，此位自动清零
0	SWTRG0	W	0x0	对此位写入 1 即可产生对 DAC0 的软件触发，此位自动清零

37.5.3 DAC0 右对齐 12 位数据寄存器 (DAC_DHR12R0)

偏移地址：0x08

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	DHR0	R/W	0x0	为 DAC0 指定 12 位数据

37.5.4 DAC0 左对齐 12 位数据寄存器 (DAC_DHR12L0)

偏移地址: 0x0C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:4	DHR0	R/W	0x0	为 DAC0 指定 12 位数据
3:0	RSV	-	-	保留

37.5.5 DAC0 右对齐 8 位数据寄存器 (DAC_DHR8R0)

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	DHR0	R/W	0x0	为 DAC0 指定 8 位数据

37.5.6 DAC1 右对齐 12 位数据寄存器 (DAC_DHR12R1)

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	DHR1	R/W	0x0	为 DAC1 指定 12 位数据

37.5.7 DAC1 左对齐 12 位数据寄存器 (DAC_DHR12L1)

偏移地址：0x18

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:4	DHR1	R/W	0x0	为 DAC1 指定 12 位数据
3:0	RSV	-	-	保留

37.5.8 DAC1 右对齐 8 位数据寄存器 (DAC_DHR8R1)

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:8	RSV	-	-	保留
7:0	DHR1	R/W	0x0	为 DAC1 指定 8 位数据

37.5.9 双 DAC 右对齐 12 位数据寄存器 (DAC_DHR12RD)

偏移地址：0x20

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:28	RSV	-	-	保留
27:16	DHR1	R/W	0x0	为 DAC1 指定 12 位数据
15:12	RSV	-	-	保留
11:0	DHR0	R/W	0x0	为 DAC0 指定 12 位数据

37.5.10 双 DAC 左对齐 12 位数据寄存器 (DAC_DHR12LD)

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:20	DHR1	R/W	0x0	为 DAC1 指定 12 位数据
19:16	RSV	-	-	保留
15:4	DHR0	R/W	0x0	为 DAC0 指定 12 位数据
3:0	RSV	-	-	保留

37.5.11 双 DAC 右对齐 8 位数据寄存器 (DAC_DHR8RD)

偏移地址：0x28

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:16	RSV	-	-	保留
15:8	DHR1	R/W	0x0	为 DAC1 指定 8 位数据
7:0	DHR0	R/W	0x0	为 DAC0 指定 8 位数据

37.5.12 DAC0 输出数据寄存器 (DAC_DOR0)

偏移地址：0x2C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	DOR0	R	0x0	可以读出 DAC0 正在输出的 12 位数据

37.5.13 DAC1 输出数据寄存器 (DAC_DOR1)

偏移地址：0x30

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:12	RSV	-	-	保留
11:0	DOR1	R	0x0	可以读出 DAC1 正在输出的 12 位数据

37.5.14 中断状态寄存器（DAC_IS)

偏移地址：0x34

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:30	RSV	-	-	保留
29	DMAIT1	R/W1C	0x0	指示 DAC1 DMA 下溢中断状态，写 1 清零： 0：未发生中断 1：发生中断
28:14	RSV	-	-	保留
13	DMAIT0	R/W1C	0x0	指示 DAC0 DMA 下溢中断状态，写 1 清零： 0：未发生中断 1：发生中断
12:0	RSV	-	-	保留

37.5.15 DAC 时钟设定寄存器（DAC_CLK)

偏移地址：0x38

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:9	RSV	-	-	保留
8	EXT	R/W	0x0	低电平额外再加一个 APBCLK 周期
7:0	DIV	R/W	0x0	分频系数 例如：填十进制 82 可从 168 分频至 1.00 注：默认不设

37.6 使用流程

下面以 DAC 通道 0 配置为例，通道 1 的配置相同。

37.6.1 直接输出新的数值

1. 配置 PA4 为模拟接口，RCM 模块中使能 DAC 时钟。
2. 配置 DAC_CTRL[2]为 0，不使能触发。
3. 配置 DAC_CTRL[0]为 1 开启 DAC0，DAC 开启后需要时间上电（软件上稍加几毫秒延时）。
4. 写输出值到相应的 DAC_DHR 寄存器，相应通道直接输出对应值的电压。

37.6.2 使用触发控制输出值更新

1. 配置 PA4 为模拟接口，RCM 模块中使能 DAC 时钟。

2. 配置寄存器 DAC_CTRL[2]为 1，使能触发。
3. 配置 DAC_CTRL[5:3]，选择触发源。此处以软件触发为例，配置为 0x7。
4. 配置 DAC_CTRL[0]为 1 开启 DAC0，DAC 开启后需要时间上电（软件上稍加几毫秒延时）。
5. 写输出值到相应的 DAC_DHR 寄存器，每当触发到来，相应通道输出对应值的电压。此处例子为写 DAC_SWTRG[0]为 1，产生软件触发。

37.6.3 使用 DMA 控制输出值更新

1. 配置 PA4 为模拟接口，RCM 模块中使能 DAC 时钟。
2. 此处以 TIM1 作为触发源，配置 TIM1 在 Update 事件产生 TRGO 信号。
3. 配置 DAC_CTRL[12]为 1，使能 DAC 的 DMA 模式。
4. 配置寄存器 DAC_CTRL[2]为 1，使能触发。
5. 配置 DAC_CTRL[5:3]，选择触发源。此处以 TIM1_TRGO 触发为例，配置为 0x4。
6. 配置 DAC_CTRL[0]为 1 开启 DAC0，DAC 开启后需要时间上电（软件上稍加几毫秒延时）。
7. 配置 DAM0，DMA 控制器配置请参见“11 DMA 控制器（DMA）”章节。
8. 每当发生触发事件时，DAC 会产生 DMA 请求，DMA 将输出值写入相应的 DAC_DHR 寄存器中，相应通道输出对应值的电压；此处例子为 TIM1 在每次 Update 事件产生 TRGO 触发 DMA 传输。

37.6.4 产生噪声波

1. 配置 PA4 为模拟接口，RCM 模块中使能 DAC 时钟。
2. 配置 DAC_CTRL[7:6]，配置为 0x1，产生噪声波。
3. 配置 DAC_CTRL[11:8]，选择噪声模式下的输出幅度。
4. 配置 DAC_CTRL[2]为 1，使能触发。
5. 配置 DAC_CTRL[5:3]，选择触发源。
6. 配置相应的 DAC_DHR 寄存器设定输出的初始值。
7. 配置 DAC_CTRL[0]为 1 开启 DAC0，DAC 开启后需要时间上电（软件上稍加几毫秒延时）。
8. 每当触发到来，相应通道会按照一定变化输出噪声波。

37.6.5 产生三角波

1. 配置 PA4 为模拟接口，RCM 模块中使能 DAC 时钟。
2. 配置 DAC_CTRL[7:6]，配置为 0x2，产生三角波。
3. 配置 DAC_CTRL[11:8]，选择三角波的输出幅度。
4. 配置 DAC_CTRL[2]为 1，使能触发。

5. 配置 DAC_CTRL[5:3]，选择触发源。
6. 配置相应的 DAC_DHR 寄存器设定输出的初始值。
7. 配置 DAC_CTRL[0]为 1 开启 DAC0，DAC 开启后需要时间上电（软件上稍加几毫秒延时）。

每当触发到来，相应通道会按照一定变化输出三角波。

38 模拟比较器（ACMP）

38.1 概述

内置 3 个独立的模拟比较器（ACMP0、ACMP1、ACMP2）。

ACMP 模块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/低电平到芯片内部，可产生中断。当“INP”输入端电压高于”INM”输入端电压时，比较器输出为高电平，当“INP”输入端电压低于“INM”输入端电压时，比较器输出为低电平。

38.2 主要特性

- 模拟比较器输出，可以产生中断，中断产生方式可编程
- 最多三个独立的比较器
- 可配置迟滞电压
- 输入通道可选 I/O 端口、DAC 输出通道、内部参考电压 VBG、VDDH、VREF
- 可通过产生中断的方式将系统从 Sleep 模式唤醒

38.3 系统框图

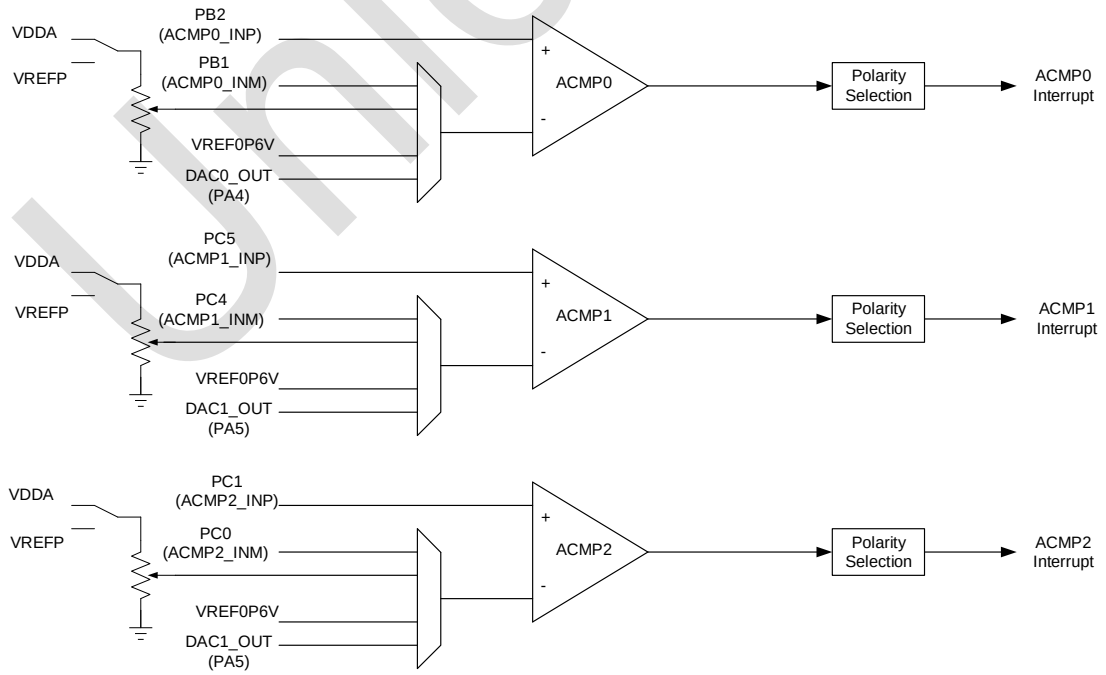


图 38-1：ACMP 系统框图

38.4 功能描述

38.4.1 电压比较器

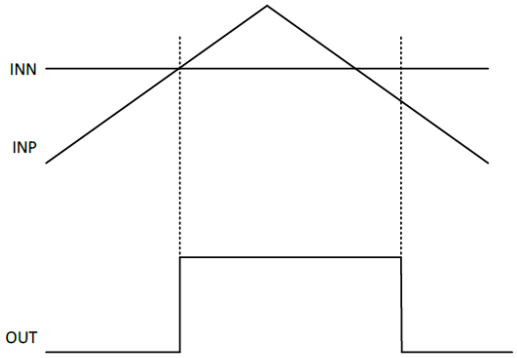


图 38-2：比较器基本功能图

在配置独立比较器功能时，当正端输入信号 INP 大于负输入信号 INN 时，比较器输出信号 output 为高，否则为低。

38.4.2 比较器迟滞功能

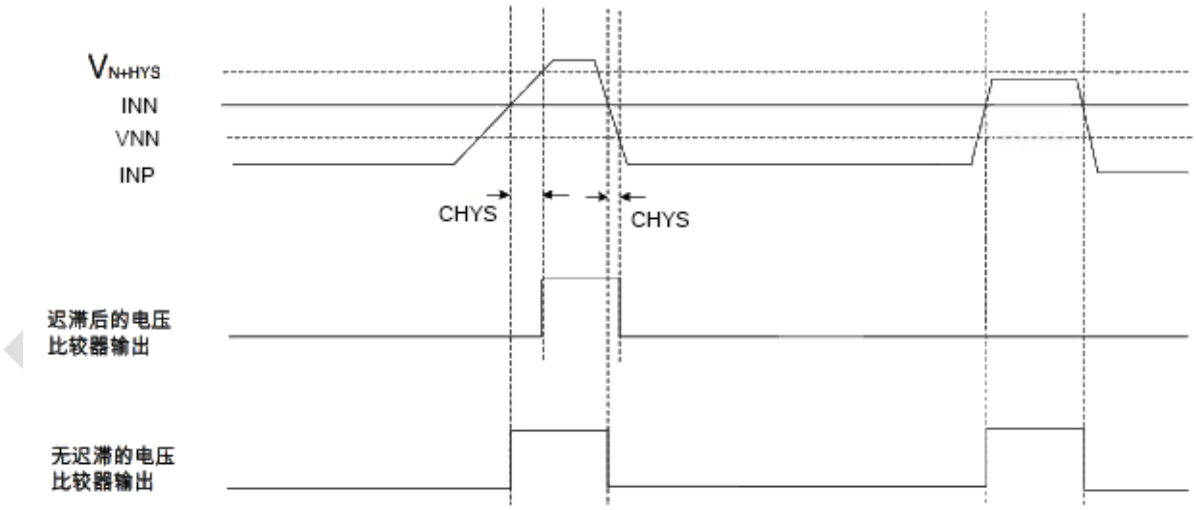


图 38-3：带迟滞功能的电压比较功能图

比较器的迟滞功能通过设置 ACMP_REG.CHYS 来设置，当迟滞电压被设置后，正端输入 INP 电压高于负端输入 INN 电压+迟滞电压 CHYS 时，比较器输出高，当正端输入电压低于负端输入电压-迟滞电压 CHYS 时，比较器输出为低。

38.5 寄存器描述

寄存器基地址：0x4008_0000

寄存器列表如下：

表 38-1 ACMP 寄存器列表

偏移地址	寄存器名	说明
0x00	ACMP_UNLOCK	写入解锁寄存器
0x1C	ACMP_CFG0	模拟比较器 0 寄存器
0x20	ACMP_CFG1	模拟比较器 1 寄存器
0x24	ACMP_CFG2	模拟比较器 2 寄存器

38.5.1 写入解锁寄存器（ACMP_UNLOCK）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	UNLOCK	R/W	0	写入解锁寄存器 写入 0xA5A55A5A 使此位变为 1，以解锁其它寄存器的写入。 写入其它值重新锁定，防止误操作。

38.5.2 模拟比较器 0 寄存器（ACMP_CFG0）

偏移地址：0x1C

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:19	RSV	-	-	保留
18	INTS	R/W1C	0	中断状态，写 1 清除
17:16	INT_CFG	R/W	0	控制中断使能： 00：关闭中断 01：OUTPUT 任意边沿产生中断 10：OUTPUT 下降沿产生中断 11：OUTPUT 上升沿产生中断
15:14	RSV	-	-	保留
13	OUTPUT	R	0	比较器 0 输出信号
12	CRVINCTRL	R/W	0	负端输入电阻分压输入选择： 0：V_{DDA} 1：V_{REF}
11:8	CRVCTRL	R/W	0	负端输入电阻分压比例 分压输出为（1/6+CRVCTRL/24）*（V_{DDA} 或 V_{REF}）

位	名称	属性	复位值	说明
7:6	CNEGSEL	R/W	0	负端输入通道选择： 0：CIN 1：VDDA/VREF 2：VBG（0.6V） 3：DACOUT（DAC0 输出）
5:4	CPOSSEL	R/W	0	正端输入通道选择： 0：CIP 0 1：CIP 1 (不支持) 2：CIP 2 (不支持） 3：CIP 3 (不支持) 注：这个寄存器使用时，固定为 0，不要修改
3:2	CHYS	R/W	0	迟滞电压选择： 0：0 mV 1：10 mV 2：20 mV 3：30 mV
1	CLPM	R/W	0	低功耗模式开关： 0：关闭低功耗模式 1：开启低功耗模式
0	EN	R/W	0	ACMP0 使能信号 0：关闭 1：开启

38.5.3 模拟比较器 1 寄存器（ACMP_CFG1）

偏移地址：0x20

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:19	RSV	-	-	保留
18	INTS	R/W1C	0	中断状态，写 1 清除
17:16	INT_CFG	R/W	0	控制中断使能： 00：关闭中断 01：OUTPUT 任意边沿产生中断 10：OUTPUT 下降沿产生中断 11：OUTPUT 上升沿产生中断
15:14	RSV	-	-	保留
13	OUTPUT	R	0	比较器 1 输出信号
12	CRVINCTRL	R/W	0	负端输入电阻分压输入选择： 0：VDDA 1：VREF
11:8	CRVCTRL	R/W	0	负端输入电阻分压比例 分压输出为(1/6+CRVCTRL/24)*(V_{DDA} 或 V_{REF})

位	名称	属性	复位值	说明
7:6	CNEGSEL	R/W	0	负端输入通道选择： 0：CIN 1：VDDA/VREF 2：VBG（0.6V） 3：DACOUT（DAC1 输出）
5:4	CPOSSEL	R/W	0	正端输入通道选择： 0：CIP 0 1：CIP 1 (不支持) 2：CIP 2 (不支持） 3：CIP 3 (不支持) 注：这个寄存器使用时，固定为 0，不要修改
3:2	CHYS	R/W	0	迟滞电压选择： 0：0 mV 1：10 mV 2：20 mV 3：30 mV
1	CLPM	R/W	0	低功耗模式开关： 0：关闭低功耗模式 1：开启低功耗模式
0	EN	R/W	0	ACMP1 使能信号： 0：关闭 1：开启

38.5.4 模拟比较器 2 寄存器（ACMP_CFG2）

偏移地址：0x24

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:19	RSV	-	-	保留
18	INTS	R/W1C	0	中断状态，写 1 清除
17:16	INT_CFG	R/W	0	控制中断使能： 00：关闭中断 01：OUTPUT 任意边沿产生中断 10：OUTPUT 下降沿产生中断 11：OUTPUT 上升沿产生中断
15:14	RSV	-	-	保留
13	OUTPUT	R	0	比较器 2 输出信号
12	CRVINCTRL	R/W	0	负端输入电阻分压输入选择： 0：V_{DDA} 1：V_{REF}
11:8	CRVCTRL	R/W	0	负端输入电阻分压比例 分压输出为(1/6+CRVCTRL/24)*(V_{DDA} 或 V_{REF})
7:6	CNEGSEL	R/W	0	负端输入通道选择： 0：CIN 1：V_{DDA}/V_{REF} 2：VBG（0.6V） 3：DACOUT（DAC1 输出）

位	名称	属性	复位值	说明
5:4	CPOSSEL	R/W	0	正端输入通道选择： 0：CIP 0 1：CIP 1 (不支持) 2：CIP 2 (不支持） 3：CIP 3 (不支持) 注：这个寄存器使用时，固定为 0，不要修改
3:2	CHYS	R/W	0	迟滞电压选择： 0：0 mV 1：10 mV 2：20 mV 3：30 mV
1	CLPM	R/W	0	低功耗模式开关： 0：关闭低功耗模式 1：开启低功耗模式
0	EN	R/W	0	ACMP2 使能信号： 0：关闭 1：开启

38.6 使用流程

1. 配置 ACMP 所使用的模拟 GPIO。
2. 配置解锁寄存器 ACMP_UNLOCK ，一次性写入 0xA5A55A5A 以解锁。
3. 配置 ACMP 输入信号选择，设置迟滞电压是否开启。
4. 选择是否配置 ACMP 中断信号，并在中断服务函数中清除中断标志。
5. 使能 ACMP 模块正常工作。
6. 当 INP>INN 时查看比较器输出信号是否为 1。

39 运算放大器（OPA）

39.1 概述

内置 2 个独立的运算放大器（OPA0、OPA1），它可以设置为纯运算放大器、比较器、单位缓冲器或 PGA。运算放大器是运用得非常广泛的一种线性集成电路，在音响方面应用得最多。

39.2 主要特性

- 可配置多种工作模式：运算放大器/比较器/单位缓冲器/PGA
- 工作电流：5mA
- 工作温度：-40℃-105℃

39.3 系统框图

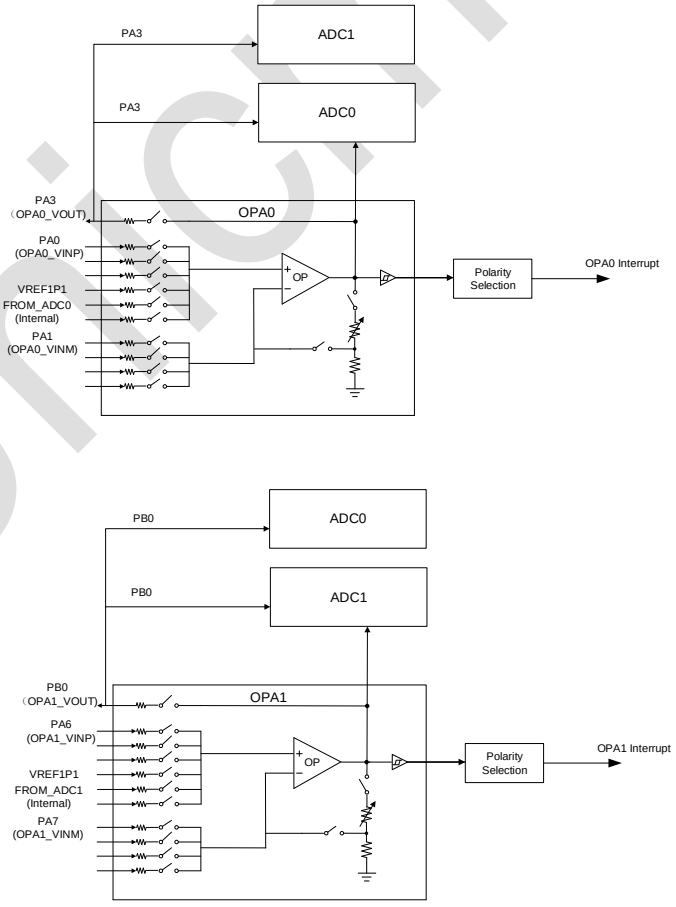


图 39-1：OPA 系统框图

39.4 寄存器描述

OPA 寄存器基地址：0x4008_0000

寄存器列表如下：

表 39-1：OPA 寄存器列表

偏移地址	寄存器名	说明
0x00	OPA_UNLOCK	写入解锁寄存器
0x10	OPA_CFG0	OPA0 设置寄存器
0x14	OPA_CFG1	OPA1 设置寄存器

39.4.1 写入解锁寄存器（OPA_UNLOCK）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:0	UNLOCK	R/W	0	写入解锁寄存器 写入 0xA5A55A5A 使此位变为 1，以解锁其它寄存器的写入。 写入其它值重新锁定，防止误操作。

39.4.2 OPA0 设置寄存器（OPA_CFG0）

偏移地址：0x10

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:19	RSV	-	-	保留
18	INT	R/W1C	0	中断状态，写 1 清除
17:16	INT_CFG	R/W	0	控制中断使能： 00：关闭中断 01：COMPOUT 任意边沿产生中断 10：COMPOUT 下降沿产生中断 11：COMPOUT 上升沿产生中断
15	RSV	-	-	保留
14	COMPOUT	R	0	OPA 作为比较器时的输出状态
13	COMPEN	R/W	0	比较器功能使能信号： 0：不使能 1：使能

位	名称	属性	复位值	说明
12:10	GAINSEL	R/W	0	单端 PGA 增益选择信号： 000：1X 001：2X 010：4X 011：8X 100：16X 101：32X 110:：64X 111：保留 注：如果选择 16X/32X/64X 的增益，需外加反馈电阻
9:7	SELP	R/W	0	OPA 正通道选择信号（SELP[2:0]）： 000: 从管脚 PA0 输入 001：保留 010：保留 011：从 CORE 电压 1.1V 输入 100：从 ADC MUX 输出连接过来 其他：保留
6:4	SELN	R/W	0	OPA 负通道选择信号（SELN[2:0]）： 000：从管脚 PA1 输入 001：模拟地 010：模拟地 011：模拟地 其他：不选择上面的（什么也不选）
3	OTPEN	R/W	0	OPA 输出到 IO 管脚的使能信号： 0：OPA 输出不到 IO 管脚 1：OPA 输出连接到 IO 管脚 PA3
2	CAPEN	R/W	0	PGA 内部反馈电容使能信号： 0：不使能内部反馈电容 1：使能内部反馈电容
1	FBRESEN	R/W	0	PGA 内部反馈电阻使能信号： 0：不使能内部反馈电阻 1：使能内部反馈电阻
0	EN	R/W	0	OPA 模块使能信号： 0：不使能 1：使能

39.4.3 OPA1 设置寄存器（OPA_CFG1）

偏移地址：0x14

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:19	RSV	-	-	保留
18	INT	R/W1C	0	中断状态，写 1 清除

位	名称	属性	复位值	说明
17:16	INT_CFG	R/W	0	控制中断使能： 00：关闭中断 01：COMPOUT 任意边沿产生中断 10：COMPOUT 下降沿产生中断 11：COMPOUT 上升沿产生中断
15	RSV	-	-	保留
14	COMPOUT	R	0	OPA 作为比较器时的输出状态
13	COMPEN	R/W	0	比较器功能使能信号： 0：不使能 1：使能
12:10	GAINSEL	R/W	0	PGA 增益选择信号： 000：1X 001：2X 010：4X 011：8X 100：16X 101：32X 110：64X 111：保留 注：如果选择 16X/32X/64X 的增益，需外加反馈电阻
9:7	SELP	R/W	0	OPA 正通道选择信号（SELP[2:0]）： 000；从管脚 PA6 输入 001：从 CORE 电压 1.1V 输入 010：保留 011：保留 100：从 ADC MUX 输出连接过来 其他：保留
6:4	SELN	R/W	0	OPA 负通道选择信号（SELN[2:0]）： 000:从管脚 PA7 输入 001：模拟地 010：模拟地 011：模拟地 其他：不选择上面的（什么也不选）
3	OTPEN	R/W	0	OPA 输出到 IO 管脚的使能信号： 0：OPA 输出不到 IO 管脚 1：OPA 输出连接到 IO 管脚（PB0）
2	CAPEN	R/W	0	PGA 内部反馈电容使能信号： 0：不使能内部反馈电容 1：使能内部反馈电容
1	FBRESEN	R/W	0	PGA 内部反馈电阻使能信号： 0：不使能内部反馈电阻 1：使能内部反馈电阻
0	EN	R/W	0	OPA 模块使能信号： 0：不使能 1：使能

39.5 使用流程

39.5.1 UNITBUFF 模式

1. 复用 OPA 相关 GPIO 为模拟功能。
2. 配置解锁寄存器 0PA_UNLOCK，一次性写入 0xA5A55A5A 以解锁。
3. 设置 OPA_CFGx[1]为 1，OPA_CFGx[2]为 1，OPA_CFGx[3]为 1，OPA_CFGx[12:10]为 0，OPA_CFGx[13]为 0。
4. 设置 OPA_CFGx 寄存器，设置 SELN 为 0，选择 SELP 输入信号。
5. 配置 OPA_CFGx[0]为 1，使能 OPA 正常工作。

39.5.2 OPA 模式下的外部反馈电路搭建

1. 复用 OPA 相关 GPIO 为模拟功能。
2. 配置解锁寄存器 0PA_UNLOCK，一次性写入 0xA5A55A5A 以解锁。
3. 设置 OPA_CFGx[1]为 0，OPA_CFGx[2]为 0，OPA_CFGx[3]为 1，OPA_CFGx[13]为 0。
4. 设置 OPA_CFGx 寄存器，设置 SELN 为 0，选择 SELP 输入信号。
5. 配置第 0 位为 1，使能 OPA 正常工作。
6. 按照图示将外线接线。

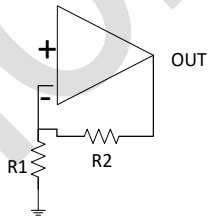


图 39-1 外部放大电路连接图

备注：放大倍数=(R1+R2)/R1

7. 查看输出端电压是否根据所需要放大倍数放大。

39.5.3 PGA 模式

1. 复用 OPA 相关 GPIO 为模拟功能。
2. 配置解锁寄存器 0PA_UNLOCK ，一次性写入 0xA5A55A5A 以解锁。
3. 设置 OPA_CFGx[1]为 1，OPA_CFGx[2]为 1，OPA_CFGx[3]为 1，OPA_CFGx[13]为 0。
4. 选择增益倍数，设置 OPA_CFGx[12:10]。
5. 设置 OPA_CFGx 寄存器，设置 SELN 为 7，选择 SELP 输入信号。

6. 配置 OPA_CFGx[0]为 1，使能 OPA 正常工作。

39.5.4 CMP 模式

1. 复用 OPA 相关 GPIO 为模拟功能。
2. 配置解锁寄存器 0PA_UNLOCK，一次性写入 0xA5A55A5A 以解锁。
3. 设置 OPA_CFGx[1]为 0，OPA_CFGx[2]为 0，OPA_CFGx[3]为 0，OPA_CFGx[12:10]为 7，OPA_CFGx[13]为 1。
4. 设置 OPA_CFGx 寄存器，设置 SELN 和 SELP 输入信号。
5. 配置中断触发方式，选择是否开启中断【开启中断之后需要清除中断信号】。
6. 配置 OPA_CFGx[0]为 1，使能 OPA 正常工作。
7. 查看 OPA_CFGx[14]是否在符合条件下置位。

40 温度传感器（TS）

40.1 概述

这是一个低功耗、高精度的温度传感器，其内置了 12 位分辨率模数转换器，可用于测量器件的环境温度（T_A）,不使用时可将传感器置于掉电模式。

40.2 主要特性

- 支持的温度范围：-40℃~105℃
- 模数转换器分辨率：12 位
- 可编程斩波时钟
- 精度：±3℃

40.3 功能描述

40.3.1 温度测量

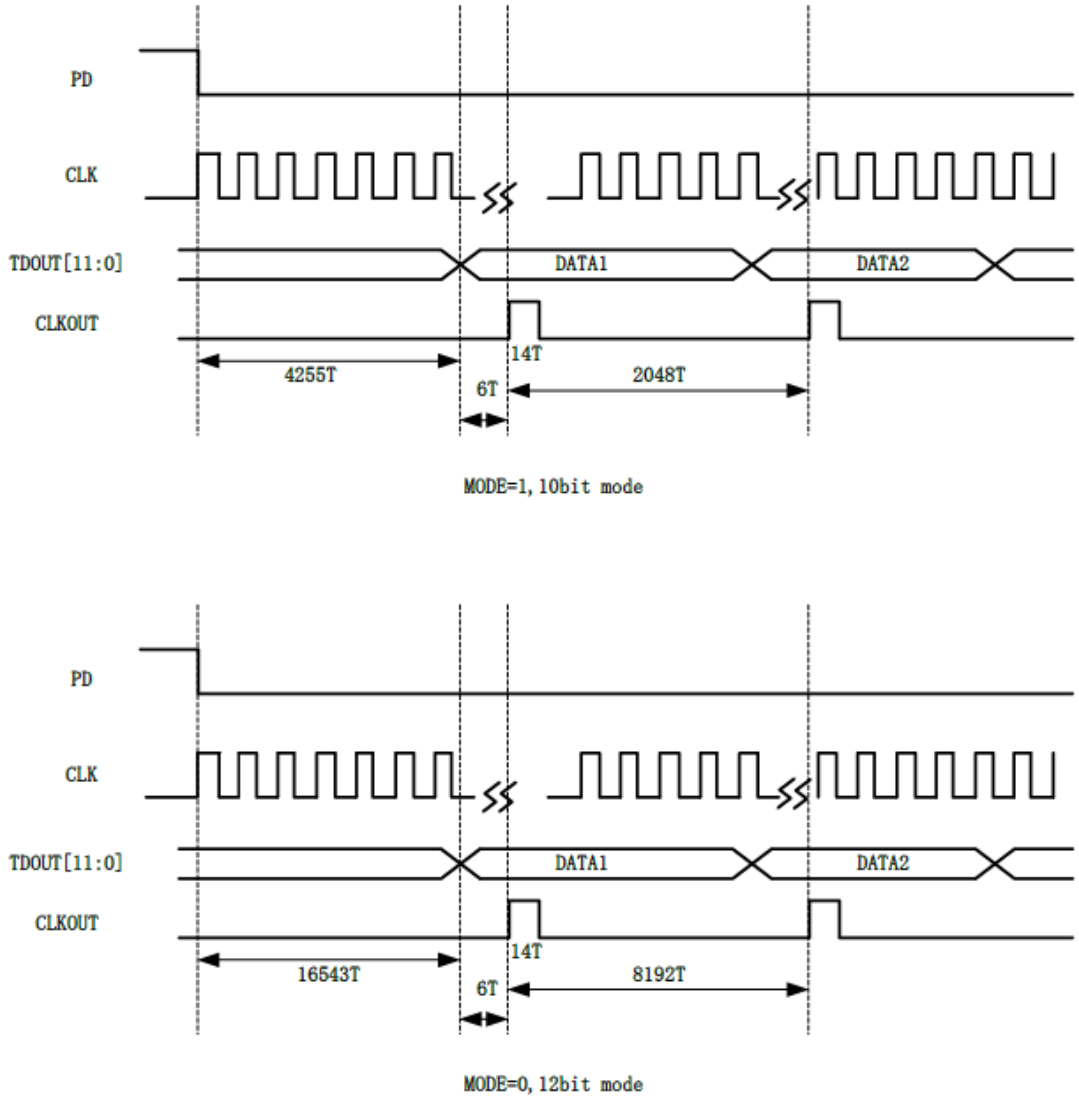


图 40-1：温度传感器转换时序图

如图 40-1，当 PD 信号变为低电平时，TS 开始工作。CLK 是 TS 的时钟。当 MODE=“1”时，在高速模式，输出数据分辨率为 10 位，输出数据速率为 f_{CLK}/2048；当 MODE =“0”时，低速模式，输出数据分辨率为 12 位，输出数据速率 f_{CLK}/8192。

40.4 寄存器描述

TS 寄存器基地址：0x4008_0000

寄存器列表如下：

表 40-1：TS 寄存器列表

偏移地址	寄存器名	说明
0x00	TS_UNLOCK	写入解锁寄存器
0x18	TS_VREFCFG	TSVREF 设置寄存器
0x28	TS_CFG	温度传感器设置寄存器
0x2C	TS_DATA	温度传感器数据寄存器

40.4.1 写入解锁寄存器（TS_UNLOCK）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:0	UNLOCK	R/W	0	写入解锁寄存器 写入 0xA5A55A5A 使此位变为 1，以解锁其它寄存器的写入。 写入其它值重新锁定，防止误操作。

40.4.2 TSVREF 设置寄存器（TS_VREFCFG）

偏移地址：0x18

复位值：0x0000 0EF1

位	名称	属性	复位值	描述
31:13	RSV	-	-	保留
13:4	CHOP_CLK_DIV	R/W	10’hEF	斩波时钟分频值设置： 斩波时钟频率=系统时钟/ CHOP_CLK_DIV
3	CHOP_CLK_EN	R/W	0	斩波时钟使能信号： 0：禁止斩波时钟 1：使能斩波时钟
2:1	VREF_SEL	R/W	0	VREF 电压选择： 00：1.5V 01：2.0V 10：2.5V 11：3.0V
0	PD	R/W	1	VREF 使能信号： 0：正常工作模式 1：掉电模式

注：此寄存器和 VREF 共用一个寄存器，如果同时使用 VREF 模块和 TS 模块时应注意斩波时钟配置，典型设置位 50k。

40.4.3 温度传感器设置寄存器（TS_CFG）

偏移地址：0x28

复位值：0x0000 0001

位	名称	属性	复位值	描述
31:3	RSV	-	-	保留
2	IRQ_EN	R/W	0	当有新数据可用时引发中断，读取 TSDATA 以清除中断。
1	MODE	R/W	0	模式选择： 0：低速模式，输出数据分辨率为 12 位，输出数据速率 f_{CLK} / 8192 1：高速模式，输出数据分辨率为 10 位，输出数据速率 f_{CLK} / 2048
0	PD	R/W	1	TS 使能信号： 0：正常工作模式 1：掉电模式

40.4.4 温度传感器数据寄存器（TS_DATA）

偏移地址：0x2C

复位值：0x0000 0000

位	名称	属性	复位值	描述
31:13	RSV	-	-	保留
12	UPDATED	R	0	表示自更新以来首次读取当前数据
11:0	DATA	R	0	传感器数据

40.5 使用流程

1. 配置解锁寄存器 TS_UNLOCK ，一次性写入 0xA5A55A5A 以解锁。
2. 配置 TS_CFG[2]，设置中断状态，选择中断开关。
3. 配置 TS_CFG[1]，选择转换模式。
4. 配置 TS_VREFCFG[3]为 1，使能斩波时钟。
5. 配置 TS_VREFCFG[13:4]，设置斩波时钟分频值，典型值为 50k。
6. 等待 TS_DATA[12]置 1，表示有新数据。
7. 读取 TS_DATA[11:0]以获取转换数据。

41 内部基准电压参考源（VREF）

41.1 概述

内置独立的电压参考源（VREF），可输出 1.5V/2.0V/2.5V/3.0V 电压。

41.2 寄存器描述

VREF 寄存器基地址：0x4008_0000

寄存器列表如下：

表 41-1：VREF 寄存器列表

偏移地址	寄存器名	说明
0x00	VREF_UNLOCK	写入解锁寄存器
0x18	TS_VREFCFG	TSVREF 设置寄存器

41.2.1 写入解锁寄存器（VREF_UNLOCK）

偏移地址：0x00

复位值：0x0000 000

位	名称	属性	复位值	说明
31:0	UNLOCK	R/W	0	**写入解锁寄存器** 写入 0xA5A55A5A 使此位变为 1，以解锁其它寄存器的写入。 写入其它值重新锁定，防止误操作。

41.2.2 TSVREF 设置寄存器（TS_VREFCFG）

偏移地址：0x18

复位值：0x0000 0EF1

位	名称	访问	复位值	说明
31:14	RSV	R	0	保留
13:4	CHOP_CLK_DIV	R/W	10’hEF	斩波时钟分频值设置， 斩波时钟频率=系统时钟/ CHOP_CLK_DIV
3	CHOP_CLK_EN	R/W	0	斩波时钟使能信号： 0：禁止斩波时钟 1：使能斩波时钟
2:1	VREF_SEL	R/W	0	VREF 电压选择： 00：1.5V 01：2.0V 10：2.5V 11：3.0V

位	名称	访问	复位值	说明
0	PD	R/W	1	VREF 使能信号： 0：正常工作模式 1：掉电模式

41.3 使用流程

1. 配置解锁寄存器 VREF_UNLOCK ，一次性写入 0xA5A55A5A 以解锁。
2. 设置 TS_VREFCFG[3]，使能斩波时钟。
3. 配置 TS_VREFCFG[13:4]，配置斩波时钟分频值。
4. 配置 TS_VREFCFG[2:1]，设置 VREF 输出电压。
5. 配置 TS_VREFCFG[0]，使能 VREF 正常工作。
6. 查看引脚输出电压。

42 系统嘀嗒定时器（SysTick）

42.1 概述

OS 要想支持多任务，就需要周期执行上下文切换，这样就需要有定时器之类的硬件资源打断程序执行。当定时器中断产生时，处理器就会在异常处理中进行 OS 任务调度，同时还会进行 OS 维护的工作。Cortex-M4 处理器中有一个称为 SysTick 的简单定时器，用于产生周期性的中断请求。SysTick 为 24 位的定时器，并且向下计数。定时器的计数减到 0 后，就会重新装载一个可编程的数值，并且同时产生 SysTick 异常 (异常编号为 15)，该异常事件会引起 SysTick 异常处理的执行，这个过程是 OS 的一部分。

对于不需要 OS 的系统，SysTick 定时器也可以用作其他用途，比如定时、计时或者为需要周期执行的任务提供中断源。SysTick 异常的产生是可控的，如果异常被禁止，仍然可以用轮询的方法使用 SysTick 定时器，比如检查当前的计数值或者轮询溢出标志。

42.2 寄存器描述

SysTick 寄存器基地址：0xE000_E010

寄存器列表如下：

表 42-1：SysTick 寄存器列表

偏移地址	寄存器名	说明
0x00	SYSTICK_CTRL	SysTick 控制和状态寄存器
0x04	SYSTICK_LOAD	SysTick 重载值寄存器
0x08	SYSTICK_VAL	SysTick 当前值寄存器

42.2.1 SysTick 控制和状态寄存器（SYSTICK_CTRL）

偏移地址：0x00

复位值：0x0000 0000

位	名称	属性	复位值	说明
31:17	RSV	R	0	保留
16	COUNTFLAG	R	0	Systick 定时器溢出标志： 1：Systick 定时器发生下溢出 0：Systick 定时器未发生溢出 读该寄存器，可清除 COUNTFLAG 标志
15:3	RSV	R	0	保留
2	CLKSOURCE	R/W	0	SysTick 时钟源选择： 1：HCLK 0：无效

位	名称	属性	复位值	说明
1	TICKINT	R/W	0	SysTick 中断使能： 1：使能中断 0：禁止中断
0	ENABLE	R/W	0	SysTick 定时器使能： 1：使能 SysTick 0：禁止 SysTick

42.2.2 SysTick 重载值寄存器（SYSTICK_LOAD）

偏移地址：0x04

复位值：0x0000 0000

位	名称	访问	复位值	说明
31:24	RSV	R	0	保留
23:0	RELOAD	R/W	0	SysTick 定时器重载值

42.2.3 SysTick 当前值寄存器（SYSTICK_VAL）

偏移地址：0x08

复位值：0x0000 0000

位	名称	访问	复位值	说明
31:24	RSV	R	0	保留
23:0	CURRENT	R/W	0	读该寄存器，获取 SysTick 定时器的当前计数值；写任意值到该寄存器，清零该寄存器及 COUNTFLAG。

42.3 使用流程

由于 SysTick 定时器的重载值和当前值在复位时都是未定义的，为了防止产生异常结果，对 SysTick 的配置需要遵循一定的流程：

1. 配置 SYSTICK_LOAD[23:0]，设置 SysTick 的溢出周期。
2. 配置 SYSTICK_CTRL[2]为 1，选择 SysTick 的时钟源为系统时钟。
3. SYSTICK_VAL[23:0]写任意值，清除 count 计数值和 count 标志。
4. 如需使用 SysTick 中断，则配置 SYSTICK_CTRL[1]为 1，使能 SysTick 中断。
5. 配置 SYSTICK_CTRL[0]为 1，使能 SysTick。
6. 查询等待定时器溢出标志到来之后关闭和清空计数器，或者在中断服务程序中读取 SYSTICK_CTRL[16]以清除溢出标志。

43 调试支持（DBG）

集成硬件调试模块的 Cortex®-M4 内核。支持指令断点（指令取值时停止）和数据断点（数据访问时停止）。当内核停止时，用户可以查看内核的内部状态和系统的外部状态。用户查询操作完成后，可以恢复内核和外设，继续执行相应的程序。 芯片内核的硬件调试模块在连接到调试器时即可使用（在未被禁用的情况下）。

支持以下调试接口：

- 串行接口（两线 SWD）
- JTAG 调试接口（4 线或者 5 线 JTAG）

44 版本维护

日期	版本	描述
2022.10.25	V1.0	初始版本
2022.11.16	V1.1	1. 更新部分寄存器名称（名称中只保留第一个下横线“_”，其余删除）。 2. 更新部分寄存器位段名称（名称中小写改为大写）。 3. 更新部分“使用流程”章节描述。
2022.12.13	V1.2	1. 新增 PTP 寄存器列表。 2. 修订部分描述。
2023.01.12	V1.3	1. 更新 I2C 章节描述。 2. 新增千兆以太网描述。
2023.01.29	V1.4	1. 更新表 8-1 中外设中断名称（小写部分改成大写）。 2. 更新中断使能寄存器（EFC_INTEN）章节中 PVFE 位的描述。 3. 更新 SYSCFG_EXTIER[15:0]的位名称。 4. 修订 SDMMC、SDIO0 为 SDIO、USB0 为 USB、LPUART0 为 LPUART。 5. 更新 RGMII 控制状态寄存器（EMAC_RGMIICS）的[3:1]位描述。 6. 更新帧过滤寄存器（EMAC_FRAMEFILTER）第 0 位的描述。 7. 新增 EMAC 章节中 EMAC_GMIIADDRESS 及 EMAC_GMIIDATA 寄存器。 8. 更新存储器地址映射图。
2023.02.06	V1.5	更新 EMAC_GMIIADDRESS 寄存器中的[5:2]位的描述。
2023.04.10	V1.6	1. 更新最高主频为 240MHz。 2. 新增“18.5 定时器内部互联”章节。 3. 更新 PLL 寄存器相关描述。
2023.08.11	V1.7	1. 修订 CAN 章节描述。 2. 修订部分寄存器描述。 3. 更新 ADC 章节系统框图。 4. 新增 SysTick 章节。
2023.11.22	V1.8	1. 更新 I2C 管脚说明。 2. 修订 GPIO_CLR 寄存器说明。 3. 更新 I2C，UART，QSPI 及 EMAC 章节中的管脚说明表格。 4. 更新 FLASH 最大容量为 512KB。
2024.04.09	V1.9	1. 更新“低功耗模式表”描述。 2. 更新部分寄存器描述。 3. 删除“ADC 管脚说明”表中 ADC_IN4 及 ADC_IN5 描述。
2025.08.14	V1.9.1	1. “5.3.4 PDR/BOR/LVD 配置寄存器（PMU_VDCR）”中 bit 1 增加注释。 2. 更新“6.3.2 PLL0 配置寄存器 0（RCM_PLL0CFGR0）”表格下方的注。

