

基于 ARM®32 位的 Cortex®-M4F 微控制器，带 256K 字节至 512K 字节内部闪存、**sLib**、**QSPI**、**SDRAM**、**17** 个定时器、**2** 个 **ADC**、**23** 个通信接口（**3** 个 **CAN**、**OTGFS**、**EMAC**）

- 内核：带有FPU的ARM®32位的Cortex®-M4F CPU
 - 最高 192MHz 工作频率，带存储器保护单元 (MPU)，内建单周期乘法和硬件除法
 - 内建浮点运算 (FPU)
 - 具有 DSP 指令集
- 存储器
 - 256 K 到 512 K 字节的内部闪存存储器
 - 26 K 字节的系统存储器作启动加载程序 (Bootloader) 用，可一次性配置成一般用户程序和数据区
 - 4 K 字节的 OTP 存储器
 - sLib：将指定之主存储区设为执行代码安全库区，此区代码仅能调用无法读取
 - 108 K 到 144 K 字节的 SRAM (可规划为 128 K 字节到 96 K 字节带奇偶校验)
 - 具有 16 位数据总线的外部存储器控制器 (XMC)：支持 NOR、PSRAM、SRAM 和 SDRAM 存储器
 - 1 个 QSPI 接口，用于连接外部 SPI 闪存存储器或 SPI RAM 扩展，支持地址映射模式
- XMC 作为 LCD 并口，兼容 8080/6800 模式
- 电源控制 (PWC)
 - 2.4V 至 3.6V 供电
 - 上电复位 (POR) / 低电压复位 (LVR)、电源电压监测器 (PVM)
 - 低功耗模式：睡眠、深度睡眠和待机，6 个 WKUP 引脚可唤醒待机模式
 - V_{BAT} 为 LEXT、ERTC 和 20 个 32 位电池供电寄存器 (ERTC_BPR) 供电
- 时钟和复位管理 (CRM)
 - 4 至 25 MHz 晶体 (HEXT)
 - 内嵌经出厂调校的 48 MHz 时钟 (e)，25 °C 达 1% 精度，-40 °C 至 +105 °C 达 2.5% 精度，带自动时钟校准 (ACC) 功能
 - PLL 可灵活配置倍频和分频系数
 - 32kHz 晶体 (LEXT)
 - 内嵌低速时钟 (LICK)
- 模拟模块
 - 2 个 12 位 5.33MSPS A/D 转换器，多达 16 个外部输入通道；分辨率 12/10/8/6 位可调；硬件过采样最高达 16 位分辨率
 - 温度传感器 (V_{TS})、内部参考电压 (V_{INTR})、VBAT 电池电压监控 (VBAT/4)
 - 2 个 12 位 D/A 转换器
- DMA
 - 2 个 7 通道 DMA 控制器共 14 通道，支持完全弹性映射
- 多达 117 个快速 I/O
 - 所有 GPIO 口可以映像到 16 个外部中断
 - 几乎所有 GPIO 口可容忍 5V 输入信号
- 多达 17 个定时器 (TMR)
 - 2 个 16 位 8 通道高级定时器，每个通道均可 PWM 输出，带死区控制和紧急停止功能
 - 多达 10 个 16 位通用定时器，其中 2 个可扩展至 32 位，其中 6 个定时器支持互补输出并带有死区控制和紧急停止功能。每个定时器最多达 4 个用于输入捕获/输出比较/PWM 或脉冲计数的通道和增量编码器输入
 - 2 个 16 位基本定时器
 - 2 个看门狗定时器 (独立的和窗口型的)
 - 系统滴答定时器：24 位递减计数器
- ERTC：增强型 RTC，具有自动唤醒、闹钟、亚秒级精度及硬件日历，带校准功能
- 多达 32 个通信接口
 - 多达 8 个 USART/UART 接口 (支持 ISO7816, LIN, IrDA 接口、调制解调控制和 RS485 驱动使能，支持 TX/RX 可配置引脚互换)
 - 多达 4 个 SPI 接口 (最高可达 40M 位/秒)，4 个均可复用为 I²S 接口，其中 I²S2/I²S3 支持全双工
 - 1 个独立全双工 I²S 接口 (I²SF)
 - 3 个 CAN 接口，各内置 1408 字节的专用缓存 (AT32F455: 2.0B 主动, AT32F456/AT32F457: FD 主动)
 - SDIO 接口
 - 全速/主机/OTG 设备接口，设备模式支持无晶振 (crystal-less)
 - 10/100M 以太网 MAC (EMAC)：有专用 DMA 和 4 K 字节 SRAM，支持 IEEE 1588, MII/RMII 接口 (只有 AT32F457 支持)
 - 红外发射器 (IRTMR)
- CRC 计算单元
- 96 位的芯片唯一代码 (UID)
- AES 硬件加速器，支持 256/192/128 位密钥大小
- 真随机数发生器 (TRNG)
- 调试模式
 - 串行单线调试 (SWD) 接口和串行线输出 (SWO) 接口
- 温度范围：-40 至 +105 °C

■ 封装

- LQFP144 20 x 20 mm
- LQFP100 14 x 14 mm
- LQFP64 10 x 10 mm
- LQFP48 7 x 7 mm
- QFN48 6 x 6 mm

■ 选型列表

闪存存储器	型号
256 K字节	AT32F455ZCT7 AT32F455VCT7 AT32F455RCT7 AT32F455CCT7 AT32F455CCU7 AT32F456ZCT7 AT32F456VCT7 AT32F456RCT7 AT32F456CCT7 AT32F456CCU7 AT32F457ZCT7 AT32F457VCT7 AT32F457RCT7
512 K字节	AT32F455ZET7 AT32F455VET7 AT32F455RET7 AT32F455CET7 AT32F455CEU7 AT32F456ZET7 AT32F456VET7 AT32F456RET7 AT32F456CET7 AT32F456CEU7 AT32F457ZET7 AT32F457VET7 AT32F457RET7

目 录

1 系统架构	31
1.1 系统概述	32
1.1.1 ARM Cortex®-M4F处理器	32
1.1.2 位带	32
1.1.3 中断和异常向量	35
1.1.4 系统嘀嗒定时器（SysTick）	37
1.1.5 复位流程	37
1.2 寄存器描述缩写说明	38
1.3 器件特征信息	39
1.3.1 闪存容量寄存器	39
1.3.2 器件电子签名	39
2 存储器资源	40
2.1 内部存储器地址映射	40
2.2 Flash存储器	41
2.3 SRAM存储器	42
2.4 外设地址映射	42
3 电源控制（PWC）	45
3.1 简介	45
3.2 主要特点	45
3.3 上电下电复位	46
3.4 电压监测器（PVM）	46
3.5 电源域划分	47
3.6 省电模式	47
3.7 PWC寄存器	49
3.7.1 电源控制寄存器（PWC_CTRL）	49
3.7.2 电源控制及状态寄存器（PWC_CTRLSTS）	51
3.7.3 电源标志清除寄存器（PWC_CLR）	53
3.7.4 LDO调校寄存器（PWC_LDOOV）	54
3.7.5 电源校准寄存器（PWC_VBGTRIM）	54
4 时钟和复位管理（CRM）	55
4.1 时钟	55
4.1.1 时钟源	56
4.1.2 系统时钟	57
4.1.3 外设时钟	57
4.1.4 时钟失效检测	57
4.1.5 自动滑顺频率切换	57
4.1.6 内部时钟输出	57
4.1.7 中断	58
4.2 复位	58
4.2.1 系统复位	58
4.2.2 电池供电域复位	58

4.3 CRM寄存器描述	59
4.3.1 时钟控制寄存器（CRM_CTRL）	60
4.3.1 PLL时钟配置寄存器（CRM_PLLCFG）	61
4.3.2 时钟配置寄存器（CRM_CFG）	62
4.3.3 时钟中断寄存器（CRM_CLKINT）	63
4.3.4 AHB外设复位寄存器1（CRM_AHBRST1）	65
4.3.5 AHB外设复位寄存器2（CRM_AHBRST2）	65
4.3.6 AHB外设复位寄存器3（CRM_AHBRST3）	66
4.3.7 APB1外设复位寄存器（CRM_APB1RST）	66
4.3.8 APB2外设复位寄存器（CRM_APB2RST）	67
4.3.9 AHB外设时钟使能寄存器1（CRM_AHBEN1）	68
4.3.10 AHB外设时钟使能寄存器2（CRM_AHBEN2）	69
4.3.11 AHB外设时钟使能寄存器3（CRM_AHBEN3）	69
4.3.12 APB1外设时钟使能寄存器（CRM_APB1EN）	70
4.3.13 APB2外设时钟使能寄存器（CRM_APB2EN）	71
4.3.14 AHB外设时钟低功耗使能寄存器1（CRM_AHBLPEN1）	72
4.3.15 AHB外设时钟低功耗使能寄存器2（CRM_AHBLPEN2）	73
4.3.16 AHB外设时钟低功耗使能寄存器3（CRM_AHBLPEN3）	74
4.3.17 APB1外设时钟低功耗使能寄存器（CRM_APB1LPEN）	74
4.3.18 APB2外设时钟低功耗使能寄存器（CRM_APB2LPEN）	76
4.3.19 外设独立时钟选择寄存器（CRM_PICLKS）	78
4.3.20 备份域控制寄存器（CRM_BPDC）	78
4.3.21 控制/状态寄存器（CRM_CTRLSTS）	79
4.3.22 额外寄存器（CRM_MISC1）	79
4.3.23 额外寄存器2（CRM_MISC2）	80
5 闪存控制器（FLASH）	82
5.1 FLASH介绍	82
5.2 主存储器操作	85
5.2.1 解锁/锁定	85
5.2.2 擦除	85
5.2.3 编程	87
5.2.4 读取	88
5.3 主存扩展区操作	89
5.4 OTP操作	89
5.5 用户系统数据区操作	89
5.5.1 解锁/锁定	89
5.5.2 擦除	89
5.5.3 编程	90
5.5.4 读取	91
5.6 闪存保护	92
5.6.1 访问保护	92
5.6.2 擦写保护	92
5.7 读取性能	93

5.8 特殊功能	93
5.8.1 安全库区设定	93
5.8.2 启动程序代码区域作为主存扩展使用	94
5.8.3 CRC校验	94
5.9 FLASH寄存器	94
5.9.1 闪存性能选择寄存器（FLASH_PSR）	95
5.9.2 闪存解锁寄存器（FLASH_UNLOCK）	95
5.9.3 闪存用户系统数据解锁寄存器（FLASH_USD_UNLOCK）	96
5.9.4 闪存状态寄存器（FLASH_STS）	96
5.9.5 闪存控制寄存器（FLASH_CTRL）	96
5.9.6 闪存地址寄存器（FLASH_ADDR）	97
5.9.7 用户系统数据寄存器（FLASH_USD）	97
5.9.8 擦除编程保护状态寄存器（FLASH_EPPS）	97
5.9.9 闪存安全库区状态寄存器0（SLIB_STS0）	97
5.9.10 闪存安全库区状态寄存器1（SLIB_STS1）	98
5.9.11 闪存安全库区密码清除寄存器（SLIB_PWD_CLR）	98
5.9.12 闪存安全库区额外状态寄存器（SLIB_MISC_STS）	99
5.9.13 闪存CRC校验地址寄存器（FLASH_CRC_ADDR）	99
5.9.14 闪存CRC校验控制寄存器（FLASH_CRC_CTRL）	99
5.9.15 闪存CRC校验结果寄存器（FLASH_CRC_CHK）	99
5.9.16 闪存安全库区密码设定寄存器（SLIB_SET_PWD）	99
5.9.17 闪存安全库区地址设定寄存器（SLIB_SET_RANGE）	100
5.9.18 主存扩展存储区域安全库区设定寄存器（EM_SLIB_SET）	101
5.9.19 启动程序代码区模式设定寄存器（BTM_MODE_SET）	101
5.9.20 闪存安全库区解锁寄存器（SLIB_UNLOCK）	101
6 通用和复用功能I/O（GPIO和IOMUX）	102
6.1 简介	102
功能描述	102
6.2 102	
6.2.1 GPIO结构	102
6.2.2 GPIO复位状态	102
6.2.3 通用输入配置	103
6.2.4 模拟配置	103
6.2.5 通用输出配置	103
6.2.6 GPIO端口保护	104
6.2.7 IOMUX功能结构	104
6.2.8 复用功能配置	105
6.2.9 IOMUX功能输入/输出	106
6.2.10 外设复用功能引脚配置	121
6.2.11 IOMUX映射优先级	121
6.2.12 外部中断/唤醒线	121
6.3 GPIO寄存器	121
6.3.1 GPIO配置寄存器（GPIOx_CFGR）（x=A..H）	122

6.3.2	GPIO输出模式寄存器 (GPIOx_OMODE) (x=A..H)	122
6.3.3	GPIO电流推动/吸入能力切换控制寄存器 (GPIOx_ODRVR) (x=A..H)	123
6.3.4	GPIO上/下拉寄存器 (GPIOx_PULL) (x=A..H)	123
6.3.5	GPIO输入数据寄存器 (GPIOx_IDT) (x=A..H)	123
6.3.6	GPIO输出数据寄存器 (GPIOx_ODT) (x=A..H)	123
6.3.7	GPIO设置/清除寄存器 (GPIOx_SCR) (x=A..H)	124
6.3.8	GPIO写保护寄存器 (GPIOx_WPR) (x=A..H)	124
6.3.9	GPIO复用低位寄存器 (GPIOx_MUXL) (x=A..H)	124
6.3.10	GPIO复用高位寄存器 (GPIOx_MUXH) (x=A..H)	125
6.3.11	GPIO位清除寄存器 (GPIOx_CLR) (x=A..H)	125
6.3.12	GPIO位翻转寄存器 (GPIOx_TOGR) (x=A..H)	125
6.3.13	极大电流推动/吸入能力切换控制寄存器 (GPIOx_HDRV) (x=A..H)	125
7	系统配置控制器 (SCFG)	126
7.1	简介	126
7.2	SCFG寄存器	126
7.2.1	SCFG配置寄存器1 (SCFG_CFG1)	126
7.2.2	SCFG配置寄存器2 (SCFG_CFG2)	127
7.2.3	SCFG外部中断配置寄存器1 (SCFG_EXINTC1)	127
7.2.4	SCFG外部中断配置寄存器2 (SCFG_EXINTC2)	128
7.2.5	SCFG外部中断配置寄存器3 (SCFG_EXINTC3)	129
7.2.6	SCFG外部中断配置寄存器4 (SCFG_EXINTC4)	130
7.2.7	SCFG超高电流推动/吸入能力 (SCFG_UHDRV)	131
8	外部中断/事件控制器 (EXINT)	132
8.1	EXINT介绍	132
8.2	功能描述和配置流程	132
8.3	EXINT寄存器描述	133
8.3.1	中断使能寄存器 (EXINT_INTEN)	133
8.3.2	事件使能寄存器 (EXINT_EVTEN)	133
8.3.3	极性配置寄存器1 (EXINT_POLCFG1)	133
8.3.4	极性配置寄存器2 (EXINT_POLCFG2)	134
8.3.5	软件触发寄存器 (EXINT_SWTRG)	134
8.3.6	中断状态寄存器 (EXINT_INTSTS)	134
9	DMA控制器 (DMA)	135
9.1	简介	135
9.2	特性	135
9.3	功能描述	136
9.3.1	通道配置	136
9.3.2	握手机制	136
9.3.3	仲裁	136
9.3.4	可编程数据传输宽度	137
9.3.5	错误事件	138
9.3.6	中断	138
9.4	多路复用器 (DMAMUX)	138

9.4.1 DMAMUX功能描述	138
9.4.2 DMAMUX 溢出中断	140
9.5 DMA寄存器	141
9.5.1 DMA状态寄存器 (DMA_STS)	143
9.5.2 DMA标志清除寄存器 (DMA_CLR)	145
9.5.3 DMA通道x配置寄存器 (DMA_CxCTRL) (x = 1...7)	147
9.5.4 DMA通道x数据传输量寄存器 (DMA_CxDTCNT) (x = 1...7)	148
9.5.5 DMA通道x外设地址寄存器 (DMA_CxPADDR) (x = 1...7)	148
9.5.6 DMA通道x存储器地址寄存器 (DMA_CxMADDR) (x = 1...7)	148
9.5.7 DMAMUX选择寄存器 (DMA_MUXSEL)	148
9.5.8 DMAMUX通道x控制寄存器 (DMA_MUXCxCTRL) (x = 1...7)	149
9.5.9 DMAMUX生成器x控制寄存器 (DMA_MUXGxCTRL) (x = 1...4)	150
9.5.10 DMAMUX通道同步状态寄存器 (DMA_MUXSYNCSTS)	150
9.5.11 DMAMUX通道中断清除标志寄存器 (DMA_MUXSYNCCLR)	150
9.5.12 DMAMUX发生器中断状态寄存器 (DMA_MUXGSTS)	151
9.5.13 DMAMUX发生器中断清除标志寄存器 (DMA_MUXGCLR)	151
10 CRC计算单元 (CRC)	152
10.1 CRC介绍	152
10.2 CRC功能说明	153
10.3 CRC寄存器	153
10.3.1 数据寄存器 (CRC_DT)	154
10.3.2 通用数据寄存器 (CRC_CDT)	154
10.3.3 控制寄存器 (CRC_CTRL)	154
10.3.4 初始化寄存器 (CRC_IDT)	154
10.3.5 生成多项式系数寄存器 (CRC_POLY)	154
11 I ² C接口	155
11.1 I ² C简介	155
11.2 I ² C的主要特点	155
11.3 I ² C总线特性	155
11.4 I ² C接口	156
11.4.1 I ² C时序控制	158
11.4.2 数据传输管理	159
11.4.3 I ² C主机通信流程	160
11.4.4 I ² C从机通信流程	164
11.4.5 SMBus功能	168
11.4.6 SMBus主机通信流程	170
11.4.7 SMBus从机通信流程	173
11.4.8 DMA传输	176
11.4.9 错误管理	177
11.5 I ² C中断	179
11.6 I ² C调试模式	179
11.7 I ² C寄存器	179
11.7.1 控制寄存器1 (I2C_CTRL1)	180

11.7.2 控制寄存器2 (I2C_CTRL2)	181
11.7.3 地址寄存器1 (I2C_OADDR1)	182
11.7.4 地址寄存器2 (I2C_OADDR2)	182
11.7.5 时序寄存器 (I2C_CLKCTRL)	182
11.7.6 超时寄存器 (I2C_TIMEOUT)	183
11.7.7 状态寄存器 (I2C_STS)	183
11.7.8 状态清除寄存器 (I2C_CLR)	185
11.7.9 PEC寄存器 (I2C_PEC)	185
11.7.10 接收寄存器 (I2C_RXDT)	185
11.7.11 发送寄存器 (I2C_TXDT)	185
12 通用同步异步收发器 (USART)	186
12.1 USART介绍	186
12.2 全双工半双工选择器简述和配置流程	188
12.3 模式选择器简述和配置流程	188
12.3.1 模式选择器简述	188
12.3.2 模式选择器配置方法	188
12.4 USART帧格式简述和配置流程	192
12.5 DMA传输简述和配置流程	193
12.5.1 DMA发送配置流程	193
12.5.2 DMA接收配置流程	194
12.6 波特率发生器简述及配置流程	194
12.6.1 波特率发生器简述	194
12.6.2 波特率发生器配置方法	194
12.7 发送器简述和配置流程	194
12.7.1 发送器简述	194
12.7.2 发送器配置流程	195
12.8 接收器简述和配置流程	195
12.8.1 接收器简述	195
12.8.2 接收器配置流程	196
12.8.3 起始侦测和噪声检测	196
12.9 Tx/Rx可配置引脚互换	197
12.10	中断 198
12.11	I/O引脚控制 199
12.12	USART寄存器描述 199
12.12.1 状态寄存器 (USART_STS)	199
12.12.2 数据寄存器 (USART_DT)	201
12.12.3 波特比率寄存器 (USART_BAUDR)	201
12.12.4 控制寄存器1 (USART_CTRL1)	201
12.12.5 控制寄存器2 (USART_CTRL2)	202
12.12.6 控制寄存器3 (USART_CTRL3)	204
12.12.7 保护时间和预分频寄存器 (GDIV)	205
12.12.8 接收器超时检测值寄存器 (RTOV)	205
12.12.9 中断标志位清除寄存器 (IFC)	205

13 串行外设接口 (SPI)	206
13.1 串行外设接口 (SPI) 简介	206
13.2 SPI功能描述	206
13.2.1 SPI简述	206
13.2.2 全双工半双工选择器简述和配置流程	207
13.2.3 CS控制器简述和配置流程	209
13.2.4 SPI_SCK控制器简述和配置流程	210
13.2.5 CRC简述和配置流程	210
13.2.6 DMA传输简述和配置流程	210
13.2.7 TI模式简述和配置流程	211
13.2.8 发送器简述和配置流程	211
13.2.9 接收器简述和配置流程	212
13.2.10 Motorola模式通信时序	212
13.2.11 TI模式通信时序	215
13.2.12 中断	216
13.2.13 IO管脚控制	216
13.3 I ² S功能描述	216
13.3.1 I ² S简述	216
13.3.2 I ² S 全双工	217
13.3.3 操作模式选择器简述和配置流程	218
13.3.4 音频协议选择器简述和配置流程	219
13.3.5 I2S_CLK控制器简述和配置流程	220
13.3.6 DMA传输简述和配置流程	221
13.3.7 发送器接收器简述和配置流程	222
13.3.8 I2S通信时序	223
13.3.9 中断	223
13.3.10 IO管脚控制	223
13.4 SPI寄存器	224
13.4.1 SPI控制寄存器1 (SPI_CTRL1) (I2S模式下不使用)	225
13.4.2 SPI控制寄存器2 (SPI_CTRL2)	226
13.4.3 SPI状态寄存器 (SPI_STS)	227
13.4.4 SPI数据寄存器 (SPI_DT)	228
13.4.5 SPICRC多项式寄存器 (SPI_CPOLY) (I2S模式下不使用)	228
13.4.6 SPIRxCRC寄存器 (SPI_RCRC) (I2S模式下不使用)	228
13.4.7 SPITxCRC寄存器 (SPI_TCRC)	228
13.4.8 SPI_I2S配置寄存器 (SPI_I2SCTRL)	228
13.4.9 SPI_I2S预分频寄存器 (SPI_I2SCLK)	229
14 全双工音频接口 (I2SF)	230
14.1 全双工音频接口 (I2SF) 简介	230
14.2 I2SF功能描述	230
14.2.1 I2SF全双工	230
14.2.2 I2SF 主时钟源选择	231
14.2.3 PCM模式通信时序	232

14.2.4 中断.....	233
14.2.5 I0管脚控制	233
14.2.6 I2SF注意事项	233
14.3 I2SF寄存器	233
14.3.1 I2SF控制寄存器2 (I2SF_CTRL2)	234
14.3.2 I2SF状态寄存器 (I2SF_STS)	234
14.3.3 I2SF数据寄存器 (I2SF_DT)	234
14.3.4 I2SF配置寄存器 (I2SF_I2SCTRL)	235
14.3.5 I2SF预分频寄存器 (I2SF_I2SCLKP)	235
14.3.6 I2SF额外寄存器1 (I2SF_MISC1)	236
15 定时器 (TIMER)	237
15.1 基本定时器 (TMR6和TMR7)	238
15.1.1 TMR6和TMR7简介	238
15.1.2 TMR6和TMR7主要功能	238
15.1.3 TMR6和TMR7功能描述	238
15.1.4 TMR6和TMR7寄存器	240
15.2 通用定时器 (TMR2到TMR5)	242
15.2.1 TMR2到TMR5简介	242
15.2.2 TMR2到TMR5主要功能	242
15.2.3 TMR2到TMR5功能描述	242
15.2.4 TMR2到TMR5寄存器描述	258
15.3 通用定时器 (TMR9到TMR14)	270
15.3.1 TMR9到TMR14简介	270
15.3.2 TMR9到TMR14主要功能	270
15.3.3 TMR9到TMR14功能描述	271
15.3.4 TMR9和TMR12寄存器描述	284
15.3.5 TMR10、TMR11、TMR13和TMR14寄存器描述	296
15.4 高级控制定时器 (TMR1和TMR8)	306
15.4.1 TMR1和TMR8简介	306
15.4.2 TMR1和TMR8主要功能	306
15.4.3 TMR1和TMR8功能描述	307
15.4.4 TMR1和TMR8寄存器描述	324
16 窗口看门狗 (WWDT)	340
16.1 WWDT简介	340
16.2 WWDT主要特性	340
16.3 WWDT功能描述	340
16.4 WWDT寄存器	341
16.4.1 控制寄存器 (WWDT_CTRL)	341
16.4.2 配置寄存器 (WWDT_CFG)	342
16.4.3 状态寄存器 (WWDT_STS)	342
17 看门狗 (WDT)	343
17.1 WDT简介	343
17.2 WDT主要特性	343

17.3 WDT功能描述	343
17.4 调试模式	344
17.5 WDT寄存器	345
17.5.1 命令寄存器 (WDT_CMD)	345
17.5.2 预分频寄存器 (WDT_DIV)	345
17.5.3 重装载寄存器 (WDT_RLD)	345
17.5.4 状态寄存器 (WDT_STS)	346
17.5.5 窗口寄存器 (WDT_WIN)	346
18 实时时钟 (ERTC)	347
18.1 ERTC简介	347
18.2 ERTC主要特性	347
18.3 ERTC功能说明	347
18.3.1 ERTC时钟	347
18.3.2 ERTC初始化	348
18.3.3 周期性自动唤醒	350
18.3.4 ERTC校准	350
18.3.5 参考时钟检测	350
18.3.6 时间戳	351
18.3.7 入侵检测	351
18.3.8 复用功能输出	352
18.3.9 ERTC唤醒	352
18.4 ERTC寄存器	353
18.4.1 ERTC时间寄存器(ERTC_TIME)	353
18.4.2 ERTC日期寄存器(ERTC_DATE)	354
18.4.3 ERTC控制寄存器(ERTC_CTRL)	354
18.4.4 ERTC初始化和状态寄存器(ERTC_STS)	356
18.4.5 ERTC预分频器寄存器(ERTC_DIV)	357
18.4.6 ERTC唤醒定时器寄存器(ERTC_WAT)	357
18.4.7 ERTC粗校准寄存器(ERTC_CCAL)	357
18.4.8 ERTC闹钟A寄存器(ERTC_ALA)	358
18.4.9 ERTC闹钟B寄存器(ERTC_ALB)	358
18.4.10 ERTC写保护寄存器(ERTC_WP)	359
18.4.11 ERTC亚秒寄存器(ERTC_SBS)	359
18.4.12 ERTC时间微调寄存器(ERTC_TADJ)	359
18.4.13 ERTC时间戳时间寄存器(ERTC_TSTM)	359
18.4.14 ERTC时间戳日期寄存器(ERTC_TS DT)	360
18.4.15 ERTC时间戳亚秒寄存器(ERTC_TSSBS)	360
18.4.16 ERTC精密校准寄存器(ERTC_SCAL)	360
18.4.17 ERTC入侵配置寄存器(ERTC_TAMP)	360
18.4.18 ERTC闹钟A亚秒寄存器(ERTC_ALASBS)	362
18.4.19 ERTC闹钟B亚秒寄存器(ERTC_ALBSBS)	362
18.4.20 ERTC电池供电数据寄存器(ERTC_BPRx)	362
19 模拟/数字转换 (ADC)	363

19.1 ADC简介	363
19.2 ADC主要特征	363
19.3 ADC架构	364
19.4 ADC功能介绍	365
19.4.1 通道管理	365
19.4.2 ADC操作流程	365
19.4.3 转换顺序管理	368
19.4.4 转换中止	369
19.4.5 触发转换失败	370
19.4.6 过采样器	370
19.4.7 数据管理	372
19.4.8 电压监测	373
19.5 主从模式	373
19.5.1 数据管理	374
19.5.2 同时模式	374
19.5.3 抢占交错触发模式	375
19.5.4 普通位移模式	376
19.6 ADC寄存器	377
19.6.1 ADC状态寄存器 (ADC_STS)	377
19.6.2 ADC控制寄存器1 (ADC_CTRL1)	378
19.6.3 ADC控制寄存器2 (ADC_CTRL2)	380
19.6.4 ADC采样时间寄存器1 (ADC_SPT1)	381
19.6.5 ADC采样时间寄存器2 (ADC_SPT2)	383
19.6.6 ADC抢占通道数据偏移寄存器x (ADC_PCDTOx) (x=1..4)	384
19.6.7 ADC电压监测高边界寄存器 (ADC_VMHB)	385
19.6.8 ADC电压监测低边界寄存器 (ADC_VMLB)	385
19.6.9 ADC普通序列寄存器1 (ADC_OSQ1)	385
19.6.10 ADC普通序列寄存器2 (ADC_OSQ2)	385
19.6.11 ADC普通序列寄存器3 (ADC_OSQ3)	385
19.6.12 ADC抢占序列寄存器 (ADC_PSQ)	386
19.6.13 ADC抢占数据寄存器x (ADC_PDTx) (x= 1..4)	386
19.6.14 ADC普通数据寄存器 (ADC_ODT)	387
19.6.15 ADC过采样寄存器 (ADC_OVSP)	387
19.6.16 ADC通用状态寄存器 (ADC_CSTS)	388
19.6.17 ADC通用控制寄存器 (ADC_CCTRL)	389
19.6.18 ADC通用普通数据寄存器 (ADC_CODT)	390
20 数字/模拟转换 (DAC)	391
20.1 简介	391
20.2 主要特性	391
20.3 设计提示	391
20.4 功能描述	392
20.4.1 触发事件	392
20.4.2 噪声/三角波生成	392

20.4.3 数据配置	393
20.5 DAC寄存器	394
20.5.1 DAC控制寄存器 (DAC_CTRL)	394
20.5.2 DAC软件触发寄存器 (DAC_SWTRG)	396
20.5.3 DAC1的12位右对齐数据保持寄存器 (DAC_D1DTH12R)	396
20.5.4 DAC1的12位左对齐数据保持寄存器 (DAC_D1DTH12L)	396
20.5.5 DAC1的8位右对齐数据保持寄存器 (DAC_D1DTH8R)	397
20.5.6 DAC2的12位右对齐数据保持寄存器 (DAC_D2DTH12R)	397
20.5.7 DAC2的12位左对齐数据保持寄存器 (DAC_D2DTH12L)	397
20.5.8 DAC2的8位右对齐数据保持寄存器 (DAC_D2DTH8R)	397
20.5.9 双DAC的12位右对齐数据保持寄存器 (DAC_DDTH12R)	397
20.5.10 双DAC的12位左对齐数据保持寄存器 (DAC_DDTH12L)	397
20.5.11 双DAC的8位右对齐数据保持寄存器 (DAC_DDTH8R)	397
20.5.12 DAC1数据输出寄存器 (DAC_D1ODT)	398
20.5.13 DAC2数据输出寄存器 (DAC_D2ODT)	398
20.5.14 DAC状态寄存器 (DAC_STS)	398
21 控制器区域网络 (CAN)	399
21.1 概述	399
21.1.1 CAN控制器内核	399
21.1.2 CAN控制器协议	399
21.1.3 Classic CAN2.0B以及CANFD	399
21.1.4 向上兼容性和协议例外事件	402
21.1.5 时间触发	402
21.1.6 CiA 603时间戳	402
21.2 特性	403
21.2.1 特征列表	403
21.2.2 中断	403
21.2.3 软件操作接口	404
21.3 操作指南	407
21.3.1 接收过滤器	407
21.3.2 帧接收	408
21.3.3 帧接收处理	408
21.3.4 帧发送处理	409
21.3.5 中止帧传输	409
21.3.6 STB满	410
21.3.7 错误处理	410
21.3.8 总线离线	411
21.3.9 扩展状态和错误报告	411
21.3.10 比特率切换和收发模式转换	414
21.3.11 扩展特性	414
21.4 时间触发CAN (TTCAN)	419
21.4.1 介绍	419
21.4.2 TTCAN模式下TBUF配置	420

21.4.3 TTCAN操作	420
21.4.4 TTCAN时序	420
21.4.5 TTCAN触发类型	421
21.4.6 TTCAN窗口触发	422
21.5 CiA 603时间戳	422
21.6 CAN位时间	423
21.6.1 位速率	423
21.6.2 位时间定义	423
21.6.3 CANFD位速率转换和采样点	424
21.6.4 TDC和RDC	424
21.7 CAN寄存器	425
21.7.1 CAN CiA 603时间戳及节点控制寄存器 (TNCFG)	426
21.7.2 Classic CAN2.0B位时序寄存器 (ACTIME)	426
21.7.3 CANFD位时序寄存器 (FDTIME)	427
21.7.4 CAN限制和位时间配置寄存器 (LBTCFG)	427
21.7.5 CAN状态寄存器 (STS)	428
21.7.6 CAN发送状态寄存器 (TSTAT)	429
21.7.7 CAN发送时间戳32位 (TTS)	429
21.7.8 CAN控制和状态寄存器 (CTRLSTAT)	430
21.7.9 CAN错误配置及状态寄存器 (ERR)	433
21.7.10 CAN TTCAN: 参考信息 (REFMSG)	434
21.7.11 CAN TTCAN: 触发配置寄存器 (TTCFG)	434
21.7.12 CAN TTCAN: 触发时间 (TTTRIG)	435
21.7.13 CAN接收过滤器控制器 (ACFCTRL)	435
21.7.14 CAN过滤器编码ID区 (FCID)	436
21.7.15 CAN过滤器编码格式区 (FCFMT)	436
21.7.16 CAN过滤器编码类型区 (FCTYP)	436
21.7.17 CAN过滤器掩码ID区 (FMID)	436
21.7.18 CAN过滤器掩码格式区 (FMFMT)	436
21.7.19 CAN过滤器掩码类型区 (FMTYP)	436
21.7.20 CAN保留寄存器1 (Res 1)	437
21.7.21 CAN接收缓冲区ID区 (RBID)	437
21.7.22 CAN接收缓冲区格式区 (RBFMT)	437
21.7.23 CAN接收缓冲区类型区 (RBTP)	437
21.7.24 CAN接收缓冲区数据区 (RBDAFn)	437
21.7.25 CAN接收缓冲区 CiA 603接收时间戳1 (RBCIA1)	437
21.7.26 CAN接收缓冲区 CiA 603接收时间戳2 (RBCIA2)	437
21.7.27 CAN接收缓冲区 TTCAN周期时间 (RBTTCAN)	437
21.7.28 CAN发送缓冲区ID区 (TBID)	437
21.7.29 CAN发送缓冲区格式区 (TBFMT)	437
21.7.30 CAN发送缓冲区类型区 (TBTP)	438
21.7.31 CAN保留寄存器2 (Res 2)	438
21.7.32 CAN发送缓冲区数据区 (TBDAFn)	438

21.7.33 CAN LLC帧计算输入 (LLCFORMAT)	438
21.7.34 CAN LLC帧计算输出 (LLCSIZE)	438
21.7.35 CAN中断使能寄存器 (INTEN)	439
22 USB OTG全速 (OTGFS)	440
22.1 OTGFS系统结构框图	440
22.2 OTGFS 功能概述	440
22.3 OTGFS时钟与管脚配置	441
22.3.1 OTGFS时钟配置	441
22.3.2 OTGFS管脚配置	441
22.4 OTGFS中断	442
22.5 OTGFS 功能操作	443
22.5.1 OTGFS初始化	443
22.5.2 OTGFS FIFO配置	443
22.5.3 OTGFS主机模式	445
22.5.4 OTGFS设备模式	459
22.6 OTGFS控制和状态寄存器	472
22.6.1 CSR寄存器映像	473
22.6.2 OTGFS寄存器地址映象	474
22.6.3 OTGFS全局寄存器	478
22.6.4 主机模式下的寄存器	489
22.6.5 设备模式下的寄存器	495
22.6.6 供电和时钟控制寄存器	510
23 HICK自动时钟校准 (ACC)	511
23.1 简介	511
23.2 主要特性	511
23.3 中断请求	511
23.4 功能概述	512
23.5 原理分析	513
23.6 寄存器描述	514
23.6.1 ACC寄存器地址映象	514
23.6.2 状态寄存器 (ACC_STS)	514
23.6.3 控制寄存器1 (ACC_CTRL1)	514
23.6.4 控制寄存器2 (ACC_CTRL2)	515
23.6.5 比较值1 (ACC_C1)	515
23.6.6 比较值2 (ACC_C2)	516
23.6.7 比较值3 (ACC_C3)	516
24 红外线接口 (IRTMR)	517
25 外部存储控制器 (XMC)	518
25.1 XMC简介	518
25.2 XMC主要特征	518
25.3 XMC构造	519
25.3.1 框图	519
25.3.2 地址映射	520

25.4 NOR/PSRAM界面	521
25.4.1 操作方式	522
25.4.2 访问模式	523
25.5 SDRAM界面	539
25.5.1 SDRAM访问管理	539
25.5.2 自刷新模式和掉电模式	541
25.6 XMC寄存器	542
25.6.1 NOR闪存和PSRAM控制器寄存器	543
25.6.2 SDRAM控制器寄存器	547
26 SDIO接口	551
26.1 简介	551
26.2 主要特点	551
26.3 功能描述	553
26.3.1 卡功能描述	553
26.3.2 命令与响应	557
26.3.3 SDIO功能描述	562
26.3.4 SDIO I/O卡特定的操作	567
26.4 SDIO寄存器	569
26.4.1 SDIO电源控制寄存器 (SDIO_PWRCTRL)	570
26.4.2 SDIO时钟控制寄存器 (SDIO_CLKCTRL)	570
26.4.3 SDIO参数寄存器 (SDIO_ARG)	571
26.4.4 SDIO命令寄存器 (SDIO_CMD)	571
26.4.5 SDIO命令响应寄存器 (SDIO_RSPCMD)	572
26.4.6 SDIO响应1..4寄存器 (SDIO_RSPx)	572
26.4.7 SDIO数据定时器寄存器 (SDIO_DTTMR)	572
26.4.8 SDIO数据长度寄存器 (SDIO_DTLEN)	572
26.4.9 SDIO数据控制寄存器 (SDIO_DTCTRL)	573
26.4.10 SDIO数据计数器寄存器 (SDIO_DTCNTR)	574
26.4.11 SDIO状态寄存器 (SDIO_STS)	574
26.4.12 SDIO清除中断寄存器 (SDIO_INTCLR)	575
26.4.13 SDIO中断屏蔽寄存器 (SDIO_INTEN)	576
26.4.14 SDIOBUF计数器寄存器 (SDIO_BUFCNTR)	578
26.4.15 SDIO数据BUF寄存器 (SDIO_BUF)	578
27 以太网控制器 (EMAC)	579
27.1 EMAC简介	579
27.1.1 EMAC总体结构图	579
27.1.2 EMAC主要特征	579
27.2 EMAC模块功能详述	580
27.2.1 EMAC通信接口介绍	580
27.2.2 EMAC帧通信	584
27.2.3 专用DMA对以太网帧的传输调度	590
27.2.4 EMAC掉电模式进入及唤醒	601
27.2.5 IEEE1588定义的精确时间协议	603

27.2.6 EMAC中断	606
27.3 EMAC寄存器	606
27.3.1 以太网MAC配置寄存器(EMAC_MACCTRL)	608
27.3.2 以太网MAC帧过滤器寄存器(EMAC_MACFRMF)	610
27.3.3 以太网MAC Hash列表高寄存器(EMAC_MACHTH)	611
27.3.4 以太网MAC Hash列表低寄存器(EMAC_MACHTL)	612
27.3.5 以太网MAC MII地址寄存器(EMAC_MACMIIADDR)	612
27.3.6 以太网MAC MII数据寄存器(EMAC_MACMIIDT)	613
27.3.7 以太网MAC流控寄存器(EMAC_MACFCTRL)	613
27.3.8 以太网MAC VLAN标签寄存器(EMAC_MACVLT)	614
27.3.9 以太网MAC远程唤醒帧过滤器寄存器(EMAC_MACRWFF)	615
27.3.10 以太网MAC PMT控制和状态寄存(EMAC_MACPMTCTRLSTS)	615
27.3.11 以太网MAC中断状态寄存器(EMAC_MACISTS)	616
27.3.12 以太网MAC中断屏蔽寄存器(EMAC_MAISR)	616
27.3.13 以太网MAC地址0高寄存器(EMAC_MACA0H)	616
27.3.14 以太网MAC地址0低寄存器(EMAC_MACA0L)	617
27.3.15 以太网MAC地址1高寄存器(EMAC_MACA1H)	617
27.3.16 以太网MAC地址1低寄存器(EMAC_MACA1L)	617
27.3.17 以太网MAC地址2高寄存器(EMAC_MACA2H)	618
27.3.18 以太网MAC地址2低寄存器(EMAC_MACA2L)	618
27.3.19 以太网MAC地址3高寄存器(EMAC_MACA3H)	618
27.3.20 以太网MAC地址3低寄存器(EMAC_MACA3L)	619
27.3.21 以太网DMA总线模式寄存器(EMAC_DMABM)	619
27.3.22 以太网DMA发送轮询请求寄存器(EMAC_DMATPD)	620
27.3.23 以太网DMA接收轮询请求寄存器(EMAC_DMARPD)	621
27.3.24 以太网DMA接收描述符列表地址寄存器(EMAC_DMARDLADDR)	621
27.3.25 以太网DMA发送描述符列表地址寄存器(EMAC_DMATDLADDR)	621
27.3.26 以太网DMA状态寄存器(EMAC_DMASTS)	622
27.3.27 以太网DMA工作模式寄存器(EMAC_DMAOPM)	624
27.3.28 以太网DMA中断使能寄存器(EMAC_DMAIE)	626
27.3.29 以太网DMA丢失帧和缓存溢出计数器寄存器(EMAC_DMAMFBOCNT)	627
27.3.30 以太网DMA当前发送描述符寄存器(EMAC_DMACTD)	627
27.3.31 以太网DMA当前接收描述符寄存器(EMAC_DMACRD)	628
27.3.32 以太网DMA当前发送缓存地址寄存器(EMAC_DMACTBADDR)	628
27.3.33 以太网DMA当前接收缓存地址寄存器(EMAC_DMACRBADDR)	628
27.3.34 以太网MMC控制寄存器(EMAC_MMCCTRL)	628
27.3.35 以太网MMC接收中断寄存器(EMAC_MMCRRI)	628
27.3.36 以太网MMC发送中断寄存器(EMAC_MMCTI)	629
27.3.37 以太网MMC接收中断屏蔽寄存器(EMAC_MMCRIM)	629
27.3.38 以太网MMC发送中断屏蔽寄存器(EMAC_MMCTIM)	630
27.3.39 以太网MMC 1次冲突后发送“好”帧的计数器寄存器(EMAC_MMCTFSCC)	630
27.3.40 以太网MMC 1次以上冲突后发送“好”帧的计数器寄存器(EMAC_MMCTFSCC)	630
630	

27.3.41 以太网MMC发送“好”帧计数器寄存器(EMAC_MMCTFCNT).....	630
27.3.42 以太网MMC CRC错误接收帧计数器寄存器(EMAC_MMCRFCECNT)	630
27.3.43 以太网MMC对齐错误接收帧计数器寄存器(EMAC_MMCRFAECNT).....	631
27.3.44 以太网MMC 接收帧“好”单播帧计数器寄存器(EMAC_MMCRGUFCNT)	631
27.3.45 以太网PTP时间戳控制寄存器(EMAC_PTPTSCTRL)	631
27.3.46 以太网PTP亚秒递增寄存器(EMAC_PTPSSINC)	633
27.3.47 以太网PTP时间戳高寄存器(EMAC_PTPTSH)	633
27.3.48 以太网PTP时间戳低寄存器(EMAC_PTPTSL)	633
27.3.49 以太网PTP时间戳高更新寄存器(EMAC_PTPTSHUD).....	633
27.3.50 以太网PTP时间戳低更新寄存器(EMAC_PTPTSLUD).....	634
27.3.51 以太网PTP时间戳加数寄存器(EMAC_PTPTSAD)	634
27.3.52 以太网PTP目标时间高寄存器(EMAC_PTPTTH).....	634
27.3.53 以太网PTP目标时间低寄存器(EMAC_PTPTTL)	634
27.3.54 以太网 PTP 时间戳状态寄存器 (EMAC_PTPTSSR).....	634
27.3.55 以太网 PTP PPS 控制寄存器 (EMAC_PTPPPSCR).....	635
28 AES硬件加密单元 (AES)	636
28.1 简介	636
28.2 主要特点	636
28.3 密码算法操作	637
28.3.1 加密	637
28.3.2 解密	637
28.4 工作串接模式	638
28.4.1 电子密码本模式 (ECB)	638
28.4.2 密码块链接模式 (CBC)	639
28.4.3 计数器模式 (CTR)	640
28.4.4 伽罗瓦/计数器模式 (GCM)	641
28.4.5 伽罗瓦消息验证代码(GMAC).....	642
28.4.6 计数器模式搭配密码块链接消息验证代码(CCM)	643
28.5 数据格式与数据充填	644
28.5.1 数据格式	644
28.5.2 数据充填流程	645
28.5.3 停滞与恢复流程	646
28.6 中断与中断控制	647
28.7 寄存器描述	648
28.7.1 AES控制寄存器 (AES_CTRL)	649
28.7.2 AES状态寄存器 (AES_STS)	650
28.7.3 AES数据输入寄存器 (AES_IDT)	651
28.7.4 AES数据输出寄存器 (AES_ODT)	651
28.7.5 AES密钥寄存器0 (AES_KEY0)	651
28.7.6 AES密钥寄存器1 (AES_KEY1)	651
28.7.7 AES密钥寄存器2 (AES_KEY2)	651
28.7.8 AES密钥寄存器3 (AES_KEY3)	651
28.7.9 AES初始向量寄存器0 (AES_IV0)	651

28.7.10 AES初始向量寄存器1 (AES_IV1)	651
28.7.11 AES初始向量寄存器2 (AES_IV2)	651
28.7.12 AES初始向量寄存器3 (AES_IV3)	652
28.7.13 AES密钥寄存器4 (AES_KEY4)	652
28.7.14 AES密钥寄存器5 (AES_KEY5)	652
28.7.15 AES密钥寄存器6 (AES_KEY6)	652
28.7.16 AES密钥寄存器7 (AES_KEY7)	652
28.7.17 AES停滞讯息寄存器0 (AES_SI0)	652
28.7.18 AES停滞讯息寄存器1 (AES_SI1)	652
28.7.19 AES停滞讯息寄存器2 (AES_SI2)	653
28.7.20 AES停滞讯息寄存器3 (AES_SI3)	653
28.7.21 AES停滞讯息寄存器4 (AES_SI4)	653
28.7.22 AES停滞讯息寄存器5 (AES_SI5)	653
28.7.23 AES停滞讯息寄存器6 (AES_SI6)	653
28.7.24 AES停滞讯息寄存器7 (AES_SI7)	653
28.7.25 AES标志清除寄存器 (AES_FCLR)	654
29 真随机数发生器 (TRNG)	655
29.1 简介	655
29.2 主要特点	655
29.3 熵源	656
29.3.1 噪声源	656
29.3.2 调节组件	656
29.3.3 健康检测机制	656
29.4 状态与中断控制	658
29.5 硬件运行、配置流程与错误重置	658
29.5.1 真随机数发生器运行流程	658
29.5.2 配置流程	659
29.5.3 错误状态重置流程	660
29.6 寄存器描述	660
29.6.1 TRNG控制寄存器 (TRNG_CTRL)	660
29.6.2 TRNG状态寄存器 (TRNG_STS)	661
29.6.3 TRNG数据寄存器 (TRNG_DT)	661
29.6.4 TRNG健康测试控制寄存器 (TRNG_HTCTRL)	661
30 四线SPI (QSPI)	662
30.1 QSPI简介	662
30.2 QSPI主要特性	662
30.3 功能描述	662
30.3.1 命令从端口	662
30.3.2 CPU PIO模式	662
30.3.3 DMA握手模式	663
30.3.4 XIP (直接地址映射读取/写入) 端口	663
30.3.5 XIP端口预取功能	663
30.3.6 SPI器件操作	663

30.4 QSPI寄存器.....	670
30.4.1 命令字0 (CMD_W0)	670
30.4.2 命令字1 (CMD_W1)	670
30.4.3 命令字2 (CMD_W2)	671
30.4.4 命令字3 (CMD_W3)	671
30.4.5 控制寄存器 (CTRL)	672
30.4.6 FIFO状态寄存器 (FIFOSTS)	673
30.4.7 控制寄存器2 (CTRL2)	674
30.4.8 命令状态寄存器 (CMDSTS)	674
30.4.9 读取状态寄存器 (RSTS)	674
30.4.10 闪存大小寄存器 (FSIZE)	674
30.4.11 XIP命令字0 (XIP_CMD_W0)	675
30.4.12 XIP命令字1 (XIP_CMD_W1)	675
30.4.13 XIP命令字2 (XIP_CMD_W2)	676
30.4.14 XIP命令字3 (XIP_CMD_W3)	677
30.4.15 控制寄存器 (CTRL3)	677
30.4.16 修订寄存器 (REV)	677
30.4.17 数据端口寄存器 (DT)	677
31 调试 (DEBUG)	678
31.1 简介	678
31.2 调试与跟踪功能	678
31.3 I/O控制	678
31.4 DEBUG寄存器	678
31.4.1 DEBUG设备ID (DEBUG_IDCODE)	679
31.4.2 DEBUG控制寄存器 (DEBUG_CTRL)	680
31.4.3 DEBUG APB1暂停控制寄存器 (DEBUG_APB1_PAUSE)	680
31.4.4 DEBUG APB2暂停控制寄存器 (DEBUG_APB2_PAUSE)	682
31.4.5 DEBUG APB3暂停控制寄存器 (DEBUG_APB3_PAUSE)	682
31.4.6 DEBUG SERIES ID寄存器 (DEBUG_SER_ID)	682
32 版本历史	683

图目录

图 1-1 AT32F455/456/457 系列微控制器系统架构	31
图 1-2 Cortex®-M4F 内部框图	32
图 1-3 位带区与位带别名区的膨胀关系图 A	32
图 1-3 位带区与位带别名区的膨胀关系图 B	33
图 1-4 复位流程	37
图 2-1 AT32F455/456/457 地址配置	40
图 3-1 各电源域框图	45
图 3-2 上电/低电压复位波形图	46
图 3-3 PVM 的阈值与输出	46
图 4-1 AT32F455/456 时钟结构图	55
图 4-2 AT32F457 时钟结构图	56
图 4-3 系统复位电路	58
图 5-1 主存储器扇区擦除流程图	86
图 5-2 主存储器整片擦除流程图	87
图 5-3 主存储器编程流程图	88
图 5-4 系统数据区擦除图	90
图 5-5 系统数据区编程图	91
图 6-1 GPIO 基本结构	102
图 6-2 IOMUX 复用结构	104
图 8-1 外部中断/事件控制器框图	132
图 9-1 DMA 框图	135
图 9-2 请求/应答对后重新仲裁	136
图 9-3 PWIDHT: byte, MWIDHT: half-word	137
图 9-4 PWIDHT: half-word, MWIDHT: word	137
图 9-5 PWIDHT: word, MWIDHT: byte	137
图 9-6 DMAMUX 框图	138
图 9-7 DMAMUX 请求同步模式	140
图 9-8 DMAMUX 事件生成	141
图 10-1 CRC 计算单元框图	152
图 10-2 字节翻转示意图	153
图 11-1 I²C 总线协议	155
图 11-2 I²C 框图	156
图 11-3 建立和保持时间	158
图 11-4 I²C 主机发送流程图	162
图 11-5 I²C 主机发送时序图	162
图 11-6 I²C 主机接收流程图	163
图 11-7 I²C 主机接收时序图	163
图 11-8 10 位地址读访问 READH10=1	164
图 11-9 10 位地址读访问 READH10=0	164
图 11-10 I²C 从机发送流程图	166
图 11-11 I²C 从机发送时序图	166
图 11-12 I²C 从机接收流程图	167
图 11-13 I²C 从机接收时序图	167
图 11-14 SMBus 主机发送流程图	171
图 11-15 SMBus 主机发送时序图	172
图 11-16 SMBus 主机接收流程图	172
图 11-17 SMBus 主机接收时序图	173
图 11-18 SMBus 从机发送流程图	175
图 11-19 SMBus 从机发送时序图	175
图 11-20 SMBus 从机接收流程图	176
图 11-21 SMBus 从机接收时序图	176
图 12-1 USART 框图	186

图 12-2 LIN 模式下的 BFF 检测与 FERR 检测.....	188
图 12-3 Smartcard frame format.....	189
图 12-4 IrDA DATA(3/16)-普通模式.....	189
图 12-5 Hardware flow control.....	190
图 12-6 Mute mode using Idle line or Address mark detection.....	191
图 12-7 8-bit format USART 同步模式	191
图 12-8 字长设置	192
图 12-9 配置停止位	193
图 12-10 发送时 TDC/TDBE 的变化情况	195
图 12-11 检测噪声的数据采样	197
图 12-12 Tx/Rx 可配置引脚互换.....	198
图 12-13 USART 中断映像图.....	199
图 13-1 SPI 框图	206
图 13-2 数据时钟时序图	207
图 13-3 SPI 双线单向全双工连接示意图	208
图 13-4 SPI 作主机单线单向只收连接示意图	208
图 13-5 SPI 作从机单线单向只收连接示意图	208
图 13-6 SPI 作单线双向半双工连接示意图	209
图 13-7 主机全双工通信	213
图 13-8 从机全双工通信	213
图 13-9 主机半双工发送通信	213
图 13-10 从机半双工接收通信	214
图 13-11 从机半双工发送通信	214
图 13-12 主机半双工接收通信	214
图 13-13 TI 模式连续通信时序	215
图 13-14 TI 模式带 dummy CLK 的连续通信时序	215
图 13-15 TI 模式非连续通信时序	215
图 13-16 TI 模式 SCK 及从机 MISO 释放时刻	215
图 13-17 SPI 中断	216
图 13-18 I ² S 框图	217
图 13-19 I ² S 全双工结构图	218
图 13-20 I ² S 从设备发送连接示意图	218
图 13-21 I ² S 从设备接收连接示意图	219
图 13-22 I ² S 主设备发送连接示意图	219
图 13-23 I ² S 主设备接收连接示意图	219
图 13-24 SPI 作主机 CK & MCK 来源示意图	221
图 13-25 各音频标准时序	223
图 13-26 I ² S 中断	223
图 14-1 I2SF 全双工主发/从发通讯	230
图 14-2 I2SF 全双工主发/从收通讯	230
图 14-3 I2SF 全双工主收/从发通讯	231
图 14-4 I2SF 全双工主收/从收通讯	231
图 14-5 I2SF 主时钟来源选择	232
图 14-6 下降沿采样 PCM 通讯时序图	232
图 14-7 上升沿采样 PCM 通讯时序图	232
图 14-8 I2SF 中断	233
图 15-1 基本定时器框图	238
图 15-2 内部时钟计数示例	238
图 15-3 计数器基本结构	239
图 15-4 PRBEN=0 时的溢出事件	239
图 15-5 PRBEN=1 时的溢出事件	239
图 15-6 内部时钟分频因子为 3 计数示例	239
图 15-7 通用定时器框图	242

图 15-8 计数时钟	242
图 15-9 内部时钟计数示例	243
图 15-10 外部时钟模式 A 框图	244
图 15-11 外部时钟模式 A 计数示例	244
图 15-12 外部时钟模式 B 框图	244
图 15-13 外部时钟模式 B 计数示例	244
图 15-14 预分频器的参数从 0 变到 3 计数示例	245
图 15-15 计数器基本结构	246
图 15-16 PRBEN=0 时的溢出事件	246
图 15-17 PRBEN=1 时的溢出事件	246
图 15-18 内部时钟分频因子为 3 计数示例	246
图 15-19 内部时钟分频因子为 0 计数示例	247
图 15-20 编码模式结构	247
图 15-21 编码模式计数实例（编码器模式 C）	248
图 15-22 输入/输出通道 1 的主电路	249
图 15-23 通道 1 输入部分	249
图 15-24 PWM 输入模式配置实例	250
图 15-25 PWM 输入模式	250
图 15-26 捕获/比较通道的输出部分（通道 1 至 4）	251
图 15-27 向上计数下 PWM 模式 A	251
图 15-28 中央双向对齐计数下 PWM 模式 B	252
图 15-29 计数值与 C1DT 值匹配时翻转 C1ORAW	252
图 15-30 单周期模式	253
图 15-31 EXT 清除 CxORAW(PWM 模式 B)	253
图 15-32 复位模式例子	254
图 15-33 挂起模式下例子	254
图 15-34 触发器模式例子	255
图 15-35 主/次定时器连接框图	255
图 15-36 主定时器启动次定时器例子	256
图 15-37 外部触发同时启动主、次定时器	256
图 15-38 通用定时器 TMR9/12 框图	270
图 15-39 通用定时器 TMR10/11/13/14 框图	271
图 15-40 计数时钟	271
图 15-41 内部时钟计数示例	271
图 15-42 外部时钟模式 A 框图	272
图 15-43 外部时钟模式 A 计数示例	272
图 15-44 预分频器的参数从 0 变到 3 计数示例	273
图 15-45 计数器基本结构	273
图 15-46 PRBEN=0 时的溢出事件	274
图 15-47 PRBEN=1 时的溢出事件	274
图 15-48 内部时钟分频因子为 3 计数示例	274
图 15-49 内部时钟分频因子为 0 计数示例	275
图 15-50 向上计数模式和中央双向对齐计数模式时 OVFIF	275
图 15-51 输入/输出通道 1 的主电路	276
图 15-52 通道 1 输入部分	276
图 15-53 PWM 输入模式配置实例	277
图 15-54 PWM 输入模式	277
图 15-55 捕获/比较通道的输出部分	278
图 15-56 向上计数下 PWM 模式 A	278
图 15-57 计数值与 C1DT 值匹配时翻转 C1ORAW	279
图 15-58 单周期模式	279
图 15-59 带死区插入的互补输出	280
图 15-60 TMR 输出控制	281

图 15-61 TMR 刹车功能的例子	282
图 15-62 复位模式例子	282
图 15-63 挂起模式下例子	282
图 15-64 触发器模式例子	283
图 15-65 高级控制定时器框图	306
图 15-66 计数时钟	307
图 15-67 内部时钟计数示例	307
图 15-68 外部时钟模式 A 框图	308
图 15-69 外部时钟模式 A 计数示例	308
图 15-70 外部时钟模式 B 框图	308
图 15-71 外部时钟模式 B 计数示例	309
图 15-72 预分频器的参数从 0 变到 3 时计数示例	309
图 15-73 计数器基本结构	310
图 15-74 PRBEN=0 时的溢出事件	310
图 15-75 PRBEN=1 时的溢出事件	310
图 15-76 计数器时序图, 内部时钟分频因子为 3	311
图 15-77 内部时钟分频因子为 0 计数示例	311
图 15-78 向上计数模式和中央双向对齐计数模式时 OVFIF	312
图 15-79 编码模式结构	312
图 15-80 编码模式计数实例 (编码器模式 C)	313
图 15-81 输入/输出通道 1 的主电路	314
图 15-82 通道 1 输入部分	314
图 15-83 PWM 输入模式配置实例	315
图 15-84 PWM 输入模式	316
图 15-85 通道 1 至 4 输出部分	316
图 15-86 向上计数下 PWM 模式 A	317
图 15-87 中央双向对齐计数下 PWM 模式 B	317
图 15-88 计数值与 C1DT 值匹配时翻转 C10RAW	318
图 15-89 单周期模式	318
图 15-90 EXT 清除 CxRAW(PWM 模式 B)	319
图 15-91 带死区插入的互补输出	319
图 15-92 TMR 输出控制	320
图 15-93 TMR 刹车功能的例子	321
图 15-94 复位模式例子	321
图 15-95 挂起模式下例子	322
图 15-96 触发器模式例子	322
图 16-1 窗口看门狗框图	340
图 16-2 窗口看门狗时序图	341
图 17-1 看门狗框图	344
图 18-1 ERTC 框图	347
图 19-1 ADC1 框图	364
图 19-2 ADC 基础操作流程	366
图 19-3 ADC 上电与校准	366
图 19-4 序列模式	368
图 19-5 抢占自动转换模式	368
图 19-6 反复模式	369
图 19-7 分割模式	369
图 19-8 ADABRT 时序	370
图 19-9 普通过采样重转模式选择	371
图 19-10 普通过采样触发模式	371
图 19-11 抢占过采样	372
图 19-12 数据内容处理	372
图 19-13 主从模式的 ADC 框图	374

图 19-14 普通同时模式.....	375
图 19-15 抢占同时模式.....	375
图 19-16 抢占交错触发模式	375
图 19-17 普通位移模式.....	376
图 20-1 DAC1/DAC2 模块框图.....	391
图 20-2 DAC LFSR 寄存器算法	393
图 20-3 DAC 三角波生成.....	393
图 21-1 CAN 控制器与 CAN 总线连接示意图	399
图 21-2 CANFD 帧类型	400
图 21-3 CAN2.0 帧类型	401
图 21-4 CAN1 发送中断的产生	403
图 21-5 CAN1 接收中断的产生	404
图 21-6 CAN1 状态中断的产生	404
图 21-7 CAN1 错误中断的产生	404
图 21-8 接收过滤器示例	407
图 21-9 接收缓冲器 RB 结构图	408
图 21-10 PTB 和 STB 在 FIFO 模式下的结构图.....	409
图 21-11 环回模式：内部环回（LBMI）和外部环回（LBME）	416
图 21-12 系统矩阵示例	419
图 21-13 时间戳和 CDC 机制	422
图 21-14 CAN 位时序规范	423
图 21-15 CANFD 帧 BRS 位比特率切换	424
图 21-16 发送器延迟	424
图 22-1 OTGFS 的系统结构框图	440
图 22-2 OTGFS 中断结构示意图	442
图 22-3 写入发送 FIFO 流程图	447
图 22-4 读取接收 FIFO 流程图	448
图 22-5 HFIRRLDCTRL 为 0x0 时的 HFIR 行为	449
图 22-6 HFIRRLDCTRL 为 0x1 时的 HFIR 行为	450
图 22-7 普通 Bulk/Control OUT/SETUP 和 Bulk/Control IN 传输例程示例图	452
图 22-8 普通中断 OUT/IN 传输示例图	455
图 22-9 普通同步 OUT/IN 传输示例图	458
图 22-10 读取接收 FIFO	463
图 22-11 SETUP 数据包流程图	464
图 22-12 BULK OUT 传输示例图	467
图 22-13 CSR 存储器映像	473
图 23-1 ACC 中断映像图	511
图 23-2 ACC 框图	512
图 23-3 cross-return 策略	513
图 24-1 IRTMR 结构框图	517
图 25-1 XMC 框图	519
图 25-2 XMC 存储块区	520
图 25-3 NOR/PSRAM 界面模式 1 读	524
图 25-4 NOR/PSRAM 界面模式 1 写	524
图 25-5 NOR/PSRAM 界面模式 2 读	526
图 25-6 NOR/PSRAM 界面模式 2 写	526
图 25-7 NOR/PSRAM 界面模式 A 读	528
图 25-8 NOR/PSRAM 界面模式 A 写	528
图 25-9 NOR/PSRAM 界面模式 B 读	530
图 25-10 NOR/PSRAM 界面模式 B 写	530
图 25-11 NOR/PSRAM 界面模式 C 读	532
图 25-12 NOR/PSRAM 界面模式 C 写	532
图 25-13 NOR/PSRAM 界面模式 D 读	534

图 25-14 NOR/PSRAM 界面模式 D 写.....	534
图 25-15 NOR/PSRAM 界面复用模式读.....	536
图 25-16 NOR/PSRAM 界面复用模式写.....	536
图 25-17 NOR/PSRAM 界面同步模式复用读.....	538
图 25-18 NOR/PSRAM 界面同步模式复用写.....	538
图 25-19 SDRAM 写访问波形 (Trcd=2, 某个写过程中 SDRAM 端连续 9 次写)	540
图 25-20 未使用读缓存 FIFO SDRAM 读访问(Trcd=2,CL=3,SDRAM 端连续 4 次读).....	540
图 26-1 SDIO“无响应”和“有响应”操作.....	551
图 26-2 SDIO (多) 数据块读操作.....	552
图 26-3 SDIO (多) 数据块写操作.....	552
图 26-4 SDIO MMC 卡数据流读操作	552
图 26-5 SDIO MMC 卡数据流写操作	553
图 26-6 SDIO 框图	563
图 26-7 命令通道状态机 (CCSM)	565
图 26-8 SDIO 命令传输	565
图 26-9 数据通道状态机 (DCSM)	566
图 27-1 EMAC 框图.....	579
图 27-2 SMI 接口信号.....	580
图 27-3 MII 信号线.....	581
图 27-4 精简的独立于媒体的接口信号.....	583
图 27-5 MII 时钟源 (CLKOUT 脚提供时钟)	583
图 27-6 MII 时钟源 (外部晶振提供时钟)	583
图 27-7 MAC 帧格式.....	584
图 27-8 带标签的 MAC 帧格式	585
图 27-9 环结构描述符和链结构描述符	591
图 27-10 传输描述符	594
图 27-11 RXDMA 描述符	598
图 27-12 唤醒帧过滤寄存器	602
图 27-13 用精调的方式更新系统时间.....	604
图 27-14 PTP 触发输出连接到 TMR2 的 ITR1.....	605
图 27-15 PPS 输出	606
图 27-16 以太网中断	606
图 27-17 以太网 MAC 唤醒帧过滤器寄存器(EMAC_MACRWF)	615
图 30-1 功能框图	662
图 30-2 DMA 握手模式	663
图 30-3 写使能	664
图 30-4 页面编程	664
图 30-5 状态读取	664
图 30-6 数据读取	665
图 30-7 双线读取命令	665
图 30-8 快速读取双线 I/O 命令	666
图 30-9 四线读取命令	667
图 30-10 快速读取四线 I/O 命令	667
图 30-11 双线 DPI 命令	668
图 30-12 四线 QPI 命令	668

表目录

表 1-1 SRAM 区中的位带地址映射	33
表 1-2 外设区中的位带地址映射	34
表 1-3 AT32F455/456/457 产品的向量表	35
表 1-4 寄存器描述缩写说明	38
表 1-5 器件特征信息相关寄存器地址和复位值	39
表 2-1 512KB 闪存模块的组织	41
表 2-2 256KB 闪存模块的组织	41
表 2-3 各外设起始地址	42
表 3-1 PWC 寄存器的映像和复位值	49
表 4-1 CRM 寄存器的映像和复位值	59
表 5-1 闪存存储结构 (512K)	82
表 5-2 闪存存储结构 (256K)	82
表 5-3 用户系统数据说明	82
表 5-4 OTP 的 DATA 与 LOCK 关系	89
表 5-5 闪存访问权限	92
表 5-6 闪存接口寄存器映射和复位值	94
表 6-1 通过 GPIOA_MUX*寄存器配置端口 A 的复用功能	106
表 6-2 通过 GPIOB_MUX*寄存器配置端口 B 的复用功能	108
表 6-3 通过 GPIOC_MUX*寄存器配置端口 C 的复用功能	110
表 6-4 通过 GPIOD_MUX*寄存器配置端口 D 的复用功能	112
表 6-5 通过 GPIOE_MUX*寄存器配置端口 E 的复用功能	114
表 6-6 通过 GPIOF_MUX*寄存器配置端口 F 的复用功能	116
表 6-7 通过 GPIOG_MUX*寄存器配置端口 G 的复用功能	118
表 6-8 通过 GPIOH_MUX*寄存器配置端口 H 的复用功能	120
表 6-9 硬件抢占功能	121
表 6-10 GPIO 寄存器地址映像和复位值	122
表 7-1 SCFG 寄存器地址映像和复位值	126
表 8-1 外部中断/事件控制器寄存器映像和复位值	133
表 9-1 DMA 错误事件	138
表 9-2 DMA 中断	138
表 9-3 DMA1 / DMA2 请求弹性映射	139
表 9-4 DMAMUX EXINT LINE 用于触发输入和同步输入	140
表 9-5 DMA 寄存器的映像和复位值	141
表 10-1 CRC 计算单元寄存器映像	153
表 11-1 I ² C 时间规范	159
表 11-2 I ² C 配置表	160
表 11-3 SMBus 超时规范	169
表 11-4 SMBus 超时检测配置	169
表 11-5 SMBus 模式配置表	169
表 11-6 I ² C 错事件	177
表 11-7 I ² C 中断请求	179
表 11-8 寄存器映像和复位值	179
表 12-1 设置波特率时的误差计算	194
表 12-2 检测起始位和噪声的数据采样	197
表 12-3 检测有效数据和噪声的数据采样	197
表 12-4 最大允许偏差	197
表 12-5 USART 中断请求	198
表 12-6 USART 寄存器映像和复位值	199
表 13-1 使用系统时钟得到精确的音频频率	221
表 13-2 SPI 寄存器列表及其复位值	224
表 14-1 I2SF5 寄存器列表及其复位值	233
表 15-1 TMR 功能对比	237

表 15-2 TMR6 和 TMR7 寄存器和复位值	240
表 15-3 TMRx 内部触发连接	245
表 15-4 计数方向与编码器信号的关系	248
表 15-5 TMR2 到 TMR5 寄存器和复位值	258
表 15-6 标准 CxOUT 通道的输出控制位	267
表 15-7 TMRx 内部触发连接	273
表 15-8 TMR9 和 TMR12 寄存器复位值	284
表 15-9 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位	292
表 15-10 TMR10、TMR11、TMR13 和 TMR14 寄存器和复位值	296
表 15-11 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位	302
表 15-12 TMRx 内部触发连接	309
表 15-13 计数方向与编码器信号的关系	313
表 15-14 TMR1 和 TMR8 寄存器和复位值	324
表 15-15 带刹车功能的互补输出通道 CxOUT 和 CxCOUT 的控制位	335
表 16-1 PCLK1 频率为 72MHz 时，最大和最小看门狗超时时间	340
表 16-2 WWDT 寄存器映像和复位值	341
表 17-1 看门狗超时时间（当 LICK=40kHz 时）	344
表 17-2 WDT 寄存器映像和复位值	345
表 18-1 ERTC 寄存器配置表	348
表 18-2 ERTC 唤醒低功耗模式	352
表 18-3 中断控制位	352
表 18-4 ERTC-寄存器映像和复位值	353
表 19-1 普通通道触发来源	367
表 19-2 抢占通道触发来源	367
表 19-3 最大累加数据与过采样倍数及位移系数关系	370
表 19-4 主从模式 DMA 模式	374
表 19-5 ADC 寄存器映像和复位值	377
表 20-1 触发源选择	392
表 20-2 DAC 寄存器映像和复位值	394
表 21-1 LLC 帧（逻辑链路控制帧）定义（包含时间戳）	404
表 21-2 LLC 帧缩写定义	405
表 21-3 DLC 的定义	406
表 21-4 CAN 节点错误状态	411
表 21-5 RBALL 和 KOER	412
表 21-6 TSTAT 状态编码	413
表 21-7 TSTAT 状态事件	413
表 21-8 软件复位	417
表 21-9 CAN 寄存器列表及其复位值	425
表 22-1 OTGFS 输入/输出引脚	441
表 22-2 OTGFS 发送 FIFO SRAM 分配表	444
表 22-3 OTGFS 内部存储空间分配表	444
表 22-4 OTGFS 模块的寄存器图及其复位值	474
表 22-5 软件断开的最短持续时间	497
表 23-1 ACC 中断请求	511
表 23-2 ACC 寄存器映像和复位值	514
表 25-1 NOR/PSRAM 界面引脚	519
表 25-2 SDRAM 界面引脚	520
表 25-3 存储区块选择	520
表 25-4 8 位宽 SDRAM 地址映射	521
表 25-5 16 位宽 SDRAM 地址映射	521
表 25-6 NOR 闪存与 PSRAM 典型引脚信号	522
表 25-7 HADDR 与外部存储器地址转换	522
表 25-8 访问数据宽度与外部存储器数据宽度对照表	522

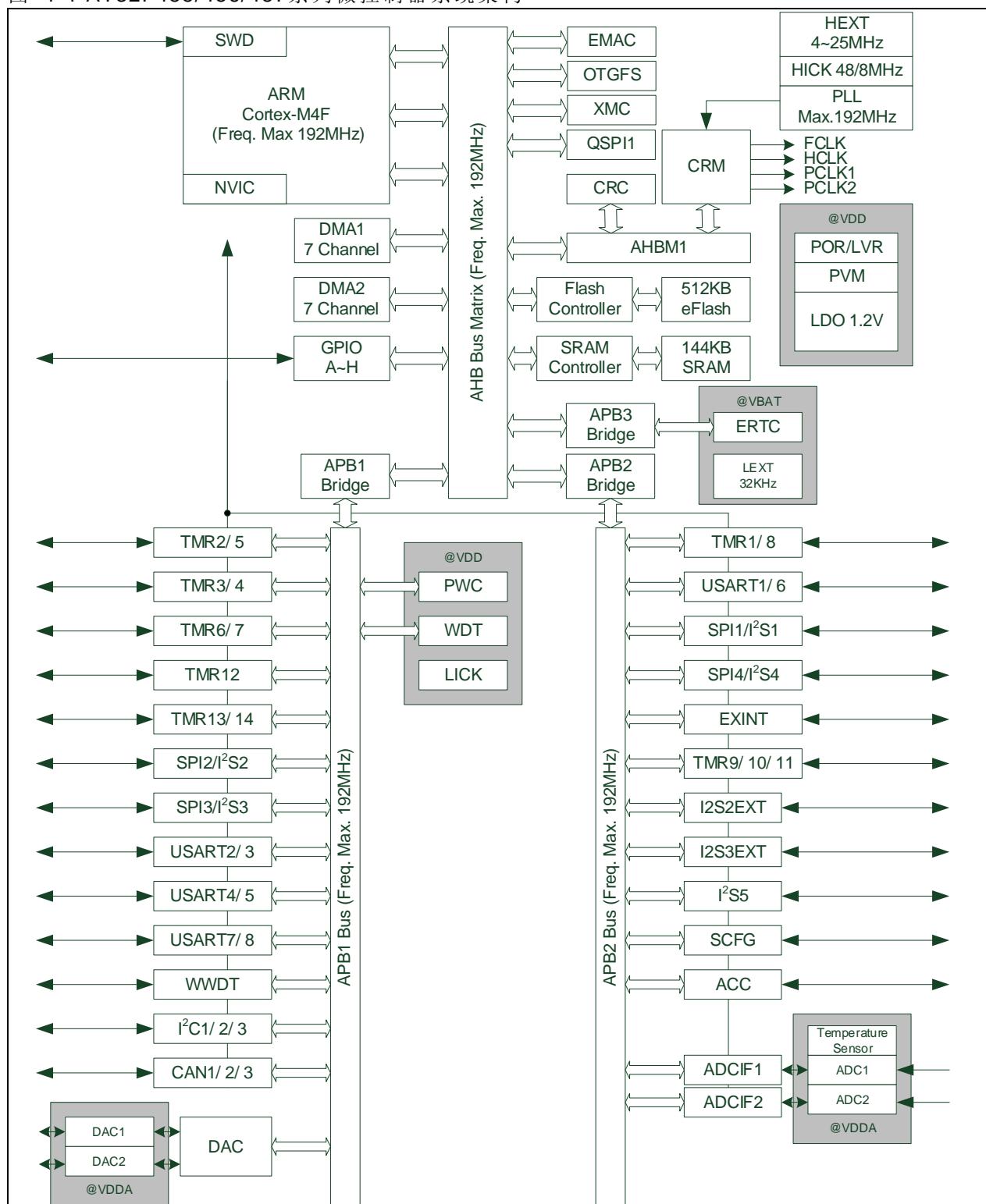
表 25-9 NOR/PSRAM 参数寄存器	523
表 25-10 模式 1 的 SRAM/NOR 闪存片选控制寄存器配置	523
表 25-11 模式 1 的 SRAM/NOR 闪存片选时序寄存器配置	523
表 25-12 模式 2 的 SRAM/NOR 闪存片选控制寄存器配置	525
表 25-13 模式 2 的 SRAM/NOR 闪存片选时序寄存器配置	525
表 25-14 模式 A 的 SRAM/NOR 闪存片选控制寄存器配置	527
表 25-15 模式 A 的 SRAM/NOR 闪存片选时序寄存器配置	527
表 25-16 模式 A 的 SRAM/NOR 闪存写时序寄存器配置	527
表 25-17 模式 B 的 SRAM/NOR 闪存片选控制寄存器配置	529
表 25-18 模式 B 的 SRAM/NOR 闪存片选时序寄存器配置	529
表 25-19 模式 B 的 SRAM/NOR 闪存写时序寄存器配置	529
表 25-20 模式 C 的 SRAM/NOR 闪存片选控制寄存器配置	531
表 25-21 模式 C 的 SRAM/NOR 闪存片选时序寄存器配置	531
表 25-22 模式 C 的 SRAM/NOR 闪存写时序寄存器配置	531
表 25-23 模式 D 的 SRAM/NOR 闪存片选控制寄存器配置	533
表 25-24 模式 D 的 SRAM/NOR 闪存片选时序寄存器配置	533
表 25-25 模式 D 的 SRAM/NOR 闪存写时序寄存器配置	533
表 25-26 复用模式的 SRAM/NOR 闪存片选控制寄存器配置	535
表 25-27 复用模式的 SRAM/NOR 闪存片选时序寄存器配置	535
表 25-28 同步模式的 SRAM/NOR 闪存片选控制寄存器配置	537
表 25-29 同步模式的 SRAM/NOR 闪存片选时序寄存器配置	537
表 25-30 XMC 寄存器地址映像	542
表 26-1 锁定/解锁命令的结构	555
表 26-2 基于命令	557
表 26-3 数据块读取命令	558
表 26-4 数据流读取和写入命令	558
表 26-5 数据块写入命令	559
表 26-6 基于块传输的写保护命令	559
表 26-7 擦除命令	559
表 26-8 I/O 模式命令	560
表 26-9 卡锁定命令	560
表 26-10 应用相关命令	560
表 26-11 R1 响应	560
表 26-12 R2 响应	561
表 26-13 R3 响应	561
表 26-14 R4 响应	561
表 26-15 R4b 响应	561
表 26-16 R5 响应	562
表 26-17 R6 响应	562
表 26-18 SDIO 管脚定义	563
表 26-19 命令格式	564
表 26-20 短响应格式	564
表 26-21 长响应格式	564
表 26-22 命令通道状态标志	564
表 26-23 数据令牌格式	566
表 26-24 SDIO 寄存器映像	569
表 26-25 响应类型和 SDIO_RSPx 寄存器	572
表 27-1 时钟范围	581
表 27-2 发送接口信号编码	582
表 27-3 接收接口信号编码	582
表 27-4 以太网模块管脚配置	584
表 27-5 目的地地址过滤器结果列表	586
表 27-6 源地址过滤器结果列表	586

表 27-7 接收描述符 0.....	599
表 27-8 以太网寄存器映像及其复位值	606
表 28-1 AES 寄存器映像和复位值.....	648
表 29-1 TRNG 寄存器映像和复位值	660
表 30-1 QSPI 寄存器映像和复位值	670
表 31-1 DEBUG 寄存器地址和复位值.....	678

1 系统架构

AT32F455/456/457 系列微控制器内部集成了：32 位 ARM Cortex®-M4F 处理器，多个 16 位和 32 位的定时器，红外线接口 IRTMR，DMA 控制器，实时时钟 ERTC，SPI 通信接口，I²C 通信接口，USART/UART 通信接口，CAN 总线控制器，外部存储控制器 XMC，USB2.0 OTG 全速接口，HICK 自动时钟校准 ACC，12 位 ADC 和 PVM 模块等外设。大量的外设和存储器。Cortex®-M4F 处理器支持增强的高效 DSP 指令集，包含扩展的单周期 16/32 位乘法累加器（MAC）、双 16 位 MAC 指令、优化的 8/16 位 SIMD 运算及饱和运算指令，并且具有单精度（IEEE-754）浮点运算单元（FPU）。系统详细架构见下图。

图 1-1 AT32F455/456/457 系列微控制器系统架构



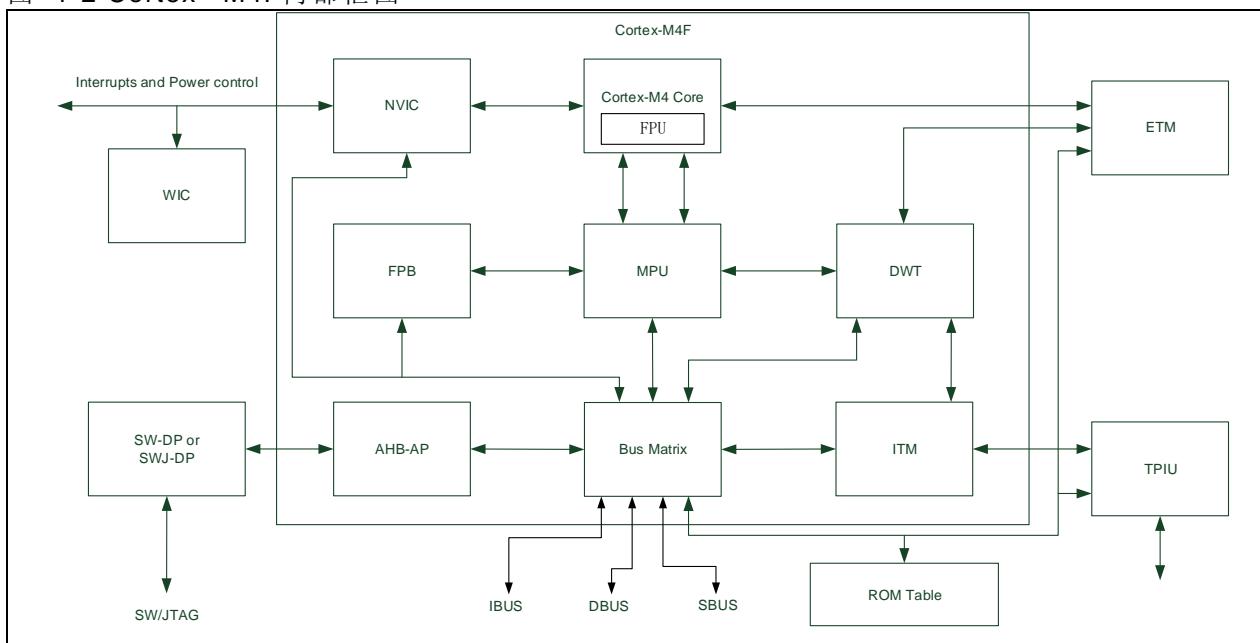
1.1 系统概述

1.1.1 ARM Cortex®-M4F处理器

Cortex®-M4F 处理器是一款低功耗处理器，具有低门数，低中断延迟和低成本调试的特点。支持包括 DSP 指令集与浮点运算功能，特别适合用于深度嵌入式应用程序需要快速中断响应功能。Cortex®-M4F 处理器是基于 ARMv7-M 架构，既支持 Thumb 指令集也支持 DSP 指令集。

下图为 Cortex®-M4F 处理器的内部框图，请参阅《ARM®Cortex-M4 技术参考手册》了解关于 Cortex®-M4F 更详尽信息。

图 1-2 Cortex®-M4F 内部框图



1.1.2 位带

利用位带操作，可以使用普通的加载/存储操作来对单一比特进行读写访问。在 Cortex®-M4F 中提供了两个位带区：SRAM 最低 1M 字节空间和外设区间的最低 1M 字节空间。这两个区中的地址除了可以像普通存储器一样访问外，还可以通过它们各自的位带别名区来快捷访问这两个区中任意地址的任意比特位，位带别名区将位带区每个比特膨胀成一个 32 位的字。当你访问位带别名区的一个地址时，等同于直接访问位带区的一个比特位。

图 1-3 位带区与位带别名区的膨胀关系图 A

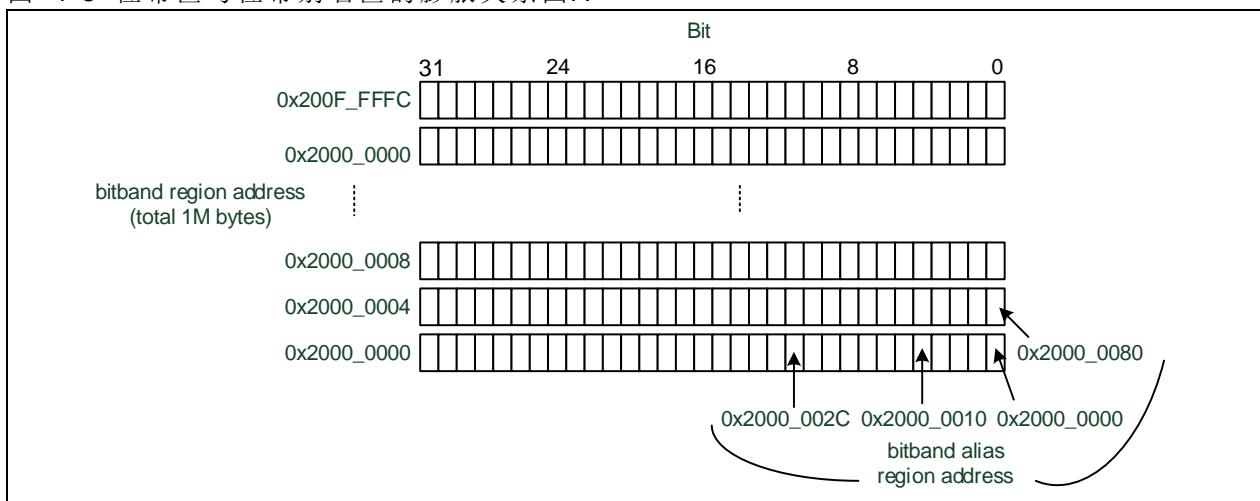
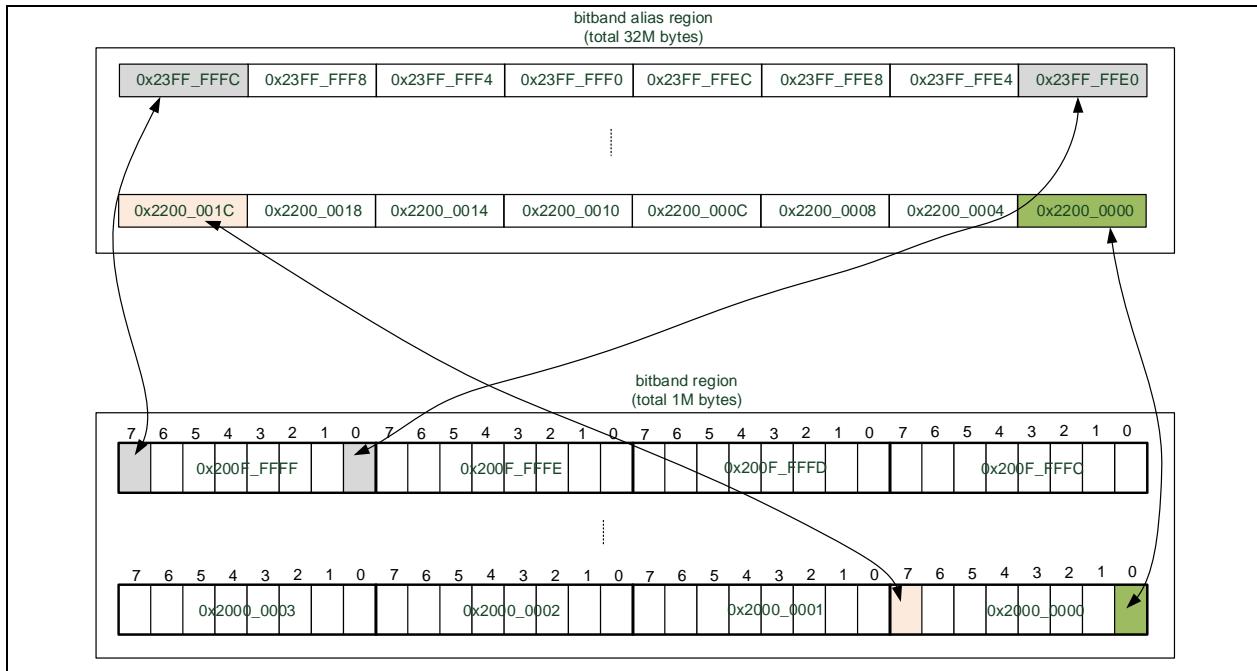


图 1-4 位带区与位带别名区的膨胀关系图B



位带区：支持位带操作的地址区

位带别名区：对别名区地址的访问最终作用到位带区的访问上

在位带区中，每个比特都映射到别名地址区的一个字（这是只有 LSB 有效的字）。当一个位带别名区地址被访问时，会先把该地址转换成位带区地址。对于读操作，读取位带区地址中的一个字，再把需要的位右移到 LSB，并把 LSB 返回。对于写操作，把需要写的位左移到对应的位序号处，然后执行一个比特级的“读-改-写”过程。

支持位带操作的两个内存区的地址范围为：

SRAM 区中的最低 1M 字节：0x2000_0000~0x200F_FFFF

外设区间的最低 1M 字节：0x4000_0000~0x400F_FFFF

对于 SRAM 位带区的某个比特，如果所在字节地址为 A，位序号为 n(0<=n<=7)，则该比特在别名区的地址为：

$$\text{AliasAddr} = 0x2200_0000 + (A - 0x2000_0000) * 32 + n * 4$$

对于外设区间位带区的某个比特，如果所在字节地址为 A，位序号为 n(0<=n<=7)，则该比特在别名区的地址为：

$$\text{AliasAddr} = 0x4200_0000 + (A - 0x4000_0000) * 32 + n * 4$$

对于 SRAM 区中，位带区与位带别名区的映射如下表所示：

表 1-1 SRAM 区中的位带地址映射

位带区	等效别名区地址
0x2000_0000.0	0x2200_0000.0
0x2000_0000.1	0x2200_0004.0
0x2000_0000.2	0x2200_0008.0
...	...
0x2000_0000.31	0x2200_007C.0
0x2000_0004.0	0x2200_0080.0
0x2000_0004.1	0x2200_0084.0
0x2000_0004.2	0x2200_0088.0
...	...
0x200F_FFFC.31	0x23FF_FFFC.0

对于外设区中，位带区与位带别名区的映射如下表所示：

表 1-2 外设区中的位带地址映射

位带区	等效别名区地址
0x4000_0000.0	0x4200_0000.0
0x4000_0000.1	0x4200_0004.0
0x4000_0000.2	0x4200_0008.0
...	...
0x4000_0000.31	0x4200_007C.0
0x4000_0004.0	0x4200_0080.0
0x4000_0004.1	0x4200_0084.0
0x4000_0004.2	0x4200_0088.0
...	...
0x400F_FFFC.31	0x43FF_FFFC.0

位带操作的优越性最容易想到的是通过 GPIO 的管脚来单独控制每盏 LED 的点亮与熄灭。另一方面，也对操作串行接口提供很大的方便。总之，位带操作对于硬件 I/O 密集型的底层程序最有用处。

位带操作还能简化跳转的判断。当跳转依据是某个位时，以前必须这样做：

- 读取整个寄存器
- 屏蔽不需要的位
- 比较并跳转

现在只需要：

- 从位带别名区读取该位的状态
- 比较并跳转

使代码更简洁，这只是位带操作优越性的初步体现，位带操作还有一个重要的好处是在多任务以及多任务环境中，将以前的读-改-写需要的三条指令，做成了一个硬件级别支持的原子操作，消除了以前读-改-写可能被中断，导致出现紊乱的情况。

1.1.3 中断和异常向量

下面列出了 AT32F455/456/457 产品的向量表。

表 1-3 AT32F455/456/457 产品的向量表

位置	优先级 类型	名称	说明	地址
-	-	-	保留	0x0000_0000
-3	固定	Reset	复位	0x0000_0004
-2	固定	NMI	CRM 时钟失效检测 (CFD) 和 SRAM 校验联接到 NMI 向量	0x0000_0008
-1	固定	硬件失效 (HardFault)	所有类型的失效	0x0000_000C
0	可设置	存储管理 (MemoryManage)	存储器管理	0x0000_0010
1	可设置	总线错误 (BusFault)	预取指失败, 存储器访问失败	0x0000_0014
2	可设置	错误应用 (UsageFault)	未定义的指令或非法状态	0x0000_0018
-	-	-	保留	0x0000_001C ~0x0000_002B
3	可设置	SVCall	通过 SWI 指令的系统服务调用	0x0000_002C
4	可设置	调试监控 (Debug Monitor)	调试监控器	0x0000_0030
-	-	-	保留	0x0000_0034
5	可设置	PendSV	可挂起的系统服务	0x0000_0038
6	可设置	SysTick	系统滴答定时器	0x0000_003C
0	7	WWDT	窗口定时器中断	0x0000_0040
1	8	PVM	连到 EXINT 的电源电压检测 (PVM) 中断	0x0000_0044
2	9	TAMPER	侵入检测中断	0x0000_0048
3	10	ERTC_WKUP	实时时钟 (ERTC) 唤醒中断	0x0000_004C
4	11	FLASH	闪存全局中断	0x0000_0050
5	12	CRM	时钟和复位管理 (CRM) 中断	0x0000_0054
6	13	EXINT0	EXINT 线 0 中断	0x0000_0058
7	14	EXINT1	EXINT 线 1 中断	0x0000_005C
8	15	EXINT2	EXINT 线 2 中断	0x0000_0060
9	16	EXINT3	EXINT 线 3 中断	0x0000_0064
10	17	EXINT4	EXINT 线 4 中断	0x0000_0068
11	18	DMA1 通道 1	DMA1 通道 1 全局中断	0x0000_006C
12	19	DMA1 通道 2	DMA1 通道 2 全局中断	0x0000_0070
13	20	DMA1 通道 3	DMA1 通道 3 全局中断	0x0000_0074
14	21	DMA1 通道 4	DMA1 通道 4 全局中断	0x0000_0078
15	22	DMA1 通道 5	DMA1 通道 5 全局中断	0x0000_007C
16	23	DMA1 通道 6	DMA1 通道 6 全局中断	0x0000_0080
17	24	DMA1 通道 7	DMA1 通道 7 全局中断	0x0000_0084
18	25	ADC1_2	ADC 的全局中断	0x0000_0088
19	26	-	-	0x0000_008C
20	27	-	-	0x0000_0090
21	28	-	-	0x0000_0094
22	29	-	-	0x0000_0098
23	30	EXINT9_5	EXINT 线[9: 5]中断	0x0000_009C
24	31	TMR1_BRK_TMR9	TMR1 停止中断和 TMR9 全局中断	0x0000_00A0
25	32	TMR1_OVF_TMR10	TMR1 溢出中断和 TMR10 全局中断	0x0000_00A4
26	33	TMR1_TRG_HALL_TMR11	TMR1 触发和 HALL 中断和 TMR11 全局中断	0x0000_00A8
27	34	TMR1_CH	TMR1 通道中断	0x0000_00AC
28	35	TMR2	TMR2 全局中断	0x0000_00B0
29	36	TMR3	TMR3 全局中断	0x0000_00B4
30	37	TMR4	TMR4 全局中断	0x0000_00B8
31	38	I2C1_EVT	I ² C1 事件中断	0x0000_00BC
32	39	I2C1_ERR	I ² C1 错误中断	0x0000_00C0

33	40	可设置	I2C2_EVT	I ² C2 事件中断	0x0000_00C4
34	41	可设置	I2C2_ERR	I ² C2 错误中断	0x0000_00C8
35	42	可设置	SPI1	SPI1 全局中断	0x0000_00CC
36	43	可设置	SPI2_I2S2EXT	SPI2 和 IS2EXT 全局中断	0x0000_00D0
37	44	可设置	USART1	USART1 全局中断	0x0000_00D4
38	45	可设置	USART2	USART2 全局中断	0x0000_00D8
39	46	可设置	USART3	USART3 全局中断	0x0000_00DC
40	47	可设置	EXINT15_10	EXINT 线[15: 10]中断	0x0000_00E0
41	48	可设置	ERTCAlarm	连到 EXINT 的 ERTC 闹钟中断	0x0000_00E4
42	49	可设置	OTGFS1_WKUP	连到 EXINT 的 OTGFS1 待机唤醒中断	0x0000_00E8
43	50	可设置	TMR8_BRK_TMR12	TMR8 停止中断和 TMR12 全局中断	0x0000_00EC
44	51	可设置	TMR8_OVF_TMR13	TMR8 溢出中断和 TMR13 全局中断	0x0000_00F0
45	52	可设置	TMR8_TRG_HALL_TMR14	TMR8 触发和 HALL 中断和 TMR14 全局中断	0x0000_00F4
46	53	可设置	TMR8_CH	TMR8 通道中断	0x0000_00F8
47	54	-	-	-	0x0000_00FC
48	55	可设置	XMC	XMC 全局中断	0x0000_0100
49	56	可设置	SDIO1	SDIO1 全局中断	0x0000_0104
50	57	可设置	TMR5	TMR5 全局中断	0x0000_0108
51	58	可设置	SPI3_I2S3EXT	SPI3 和 IS3EXT 全局中断	0x0000_010C
52	59	可设置	USART4	USART4 全局中断	0x0000_0110
53	60	可设置	USART5	USART5 全局中断	0x0000_0114
54	61	可设置	TMR6_DAC	TMR6 全局中断 DAC1 和 DAC2 下溢错误中断	0x0000_0118
55	62	可设置	TMR7	TMR7 全局中断	0x0000_011C
56	63	可设置	DMA2 通道 1	DMA2 通道 1 全局中断	0x0000_0120
57	64	可设置	DMA2 通道 2	DMA2 通道 2 全局中断	0x0000_0124
58	65	可设置	DMA2 通道 3	DMA2 通道 3 全局中断	0x0000_0128
59	66	可设置	DMA2 通道 4	DMA2 通道 4 全局中断	0x0000_012C
60	67	可设置	DMA2 通道 5	DMA2 通道 5 全局中断	0x0000_0130
61	68	可设置	EMAC ¹	以太网全局中断	0x0000_0134
62	69	可设置	EMAC_WKUP ¹	连接EXINT的以太网唤醒中断	0x0000_0138
63	70	-	-	-	0x0000_013C
64	71	-	-	-	0x0000_0140
65	72	-	-	-	0x0000_0144
66	73	-	-	-	0x0000_0148
67	74	可设置	OTGFS1	OTGFS1 全局中断	0x0000_014C
68	75	可设置	DMA2 通道 6	DMA2 通道 6 全局中断	0x0000_0150
69	76	可设置	DMA2 通道 7	DMA2 通道 7 全局中断	0x0000_0154
70	77	-	-	-	0x0000_0158
71	78	可设置	USART6	USART6 全局中断	0x0000_015C
72	79	可设置	I2C3_EVT	I ² C2 事件中断	0x0000_0160
73	80	可设置	I2C3_ERR	I ² C2 错误中断	0x0000_0164
74	81	-	-	-	0x0000_0168
75	82	-	-	-	0x0000_016C
76	83	-	-	-	0x0000_0170
77	84	-	-	-	0x0000_0174
e	85	-	-	-	0x0000_0178
79	86	-	-	-	0x0-000_017C
80	87	-	-	-	0x0000_0180
81	88	可设置	FPU	FPU 例外中断	0x0000_0184
82	89	可设置	USART7	USART7 全局中断	0x0000_0188
83	90	可设置	USART8	USART8 全局中断	0x0000_018C
84	91	可设置	SPI4	SPI4 全局中断	0x0000_0190
85	92	可设置	I2SF5	I2SF5 全局中断	0x0000_0194
86	93	-	-	-	0x0000_0198
87	94	-	-	-	0x0000_019C
88	95	-	-	-	0x0000_01A0
89	96	-	-	-	0x0000_01A4
90	97	-	-	-	0x0000_01A8

91	98	-	-	-	0x0000_01AC
92	99	可设置	QSPI1	QSPI1全局中断	0x0000_01B0
93	100	-	-	-	0x0000_01B4
94	101	可设置	DMAMUX	DMAMUX溢出中断	0x0000_01B8
95	102	-	-	-	0x0000_01BC
96	103	-	-	-	0x0000_01C0
97	104	-	-	-	0x0000_01C4
98	105	-	-	-	0x0000_01C8
99	106	-	-	-	0x0000_01CC
100	107	-	-	-	0x0000_01D0
101	108	-	-	-	0x0000_01D4
102	109	-	-	-	0x0000_01D8
103	110	可设置	ACC	ACC全局中断	0x0000_01DC

注意：AT32F457 支持 EMAC 和 EMAC_WKUP 中断，AT32F455/AT32F456 不支持这两个中断。

1.1.4 系统嘀嗒定时器（SysTick）

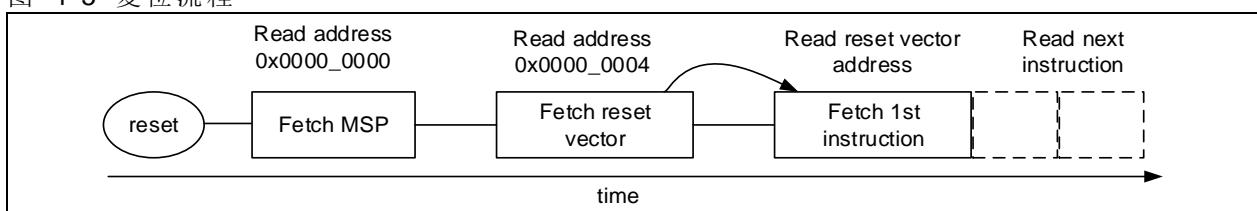
系统嘀嗒定时器是一个 24 位递减计数器，递减至零可自动重载计数初值。可产生周期性异常，用作嵌入式操作系统的多任务调度计数器，或对于无嵌入式操作系统，可用于调用需周期性执行的任务。系统嘀嗒定时器校准值固定值 9000，当系统嘀嗒时钟设定为 9MHz，产生 1ms 时间基准。

1.1.5 复位流程

系统复位后以及处理器开始执行程序前，处理器会从 CODE 存储器中读出前两个字。

- 从地址 0x0000_0000 处取出主栈指针（MSP）的初始值。
- 从地址 0x0000_0004 处取出程序计数器（PC）的初始值，这个值是复位向量，LSB 必须是 1。然后从这个值所对应的地址处取指。

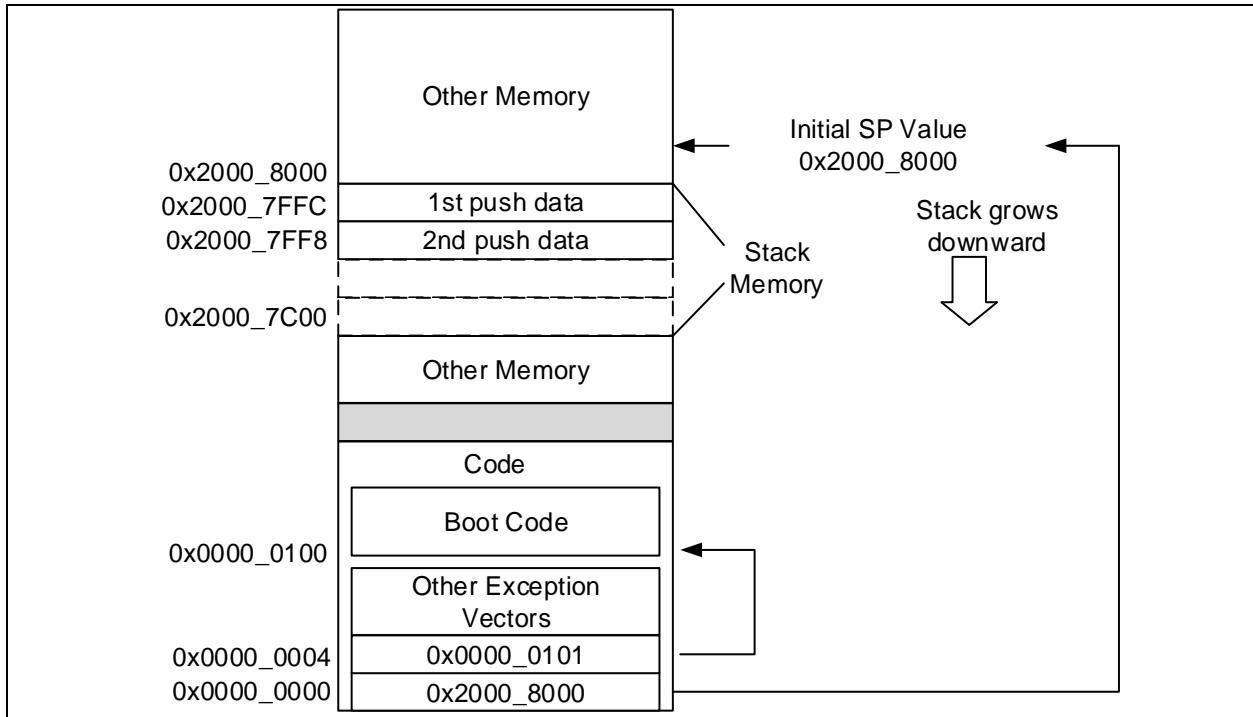
图 1-5 复位流程



Cortex®-M4F 使用的是向下生长的满栈，所以 MSP 的初始值必须是堆栈内存的末地址加 1。举例来说，堆栈区域设定在 0x2000_7C00~0x2000_7FFF 之间，那么 MSP 的初始值必须是 0x2000_8000。

向量表跟随在 MSP 的初始值之后。Cortex®-M4F 是在 Thumb 态下执行，所以向量表中的每个数值都必须将 LSB 置 1，所以，下图中使用 0x0000_0101 来表示地址 0x0000_0100。当 0x0000_0100 处的指令得到执行后，就正式开始程序的执行。在此之前初始化 MSP 是必须的，因为可能第一条指令还没执行就会被 NMI 或是其他 fault 打断。MSP 初始化好后就可以为它们的服务程序准备好堆栈空间。

图 1-6 MSP 及 PC 初始化的一个范例



在 AT32F455/456/457 中，可以将主闪存存储器、启动程序存储器或片上 SRAM 这三块存储器重映射到 0x0000_0000~0x07FF_FFFF 的 CODE 区，由 BOOT1 和 BOOT0 管脚来设定 CODE 从哪块存储器启动：

当{BOOT1, BOOT0}=00/10 时，CODE 从主闪存存储器启动。

当{BOOT1, BOOT0}=01 时，CODE 从启动程序存储器启动。

当{BOOT1, BOOT0}=11 时，CODE 从片上 SRAM 启动。

系统复位后或从待机模式退出时，BOOT1 和 BOOT0 管脚值都会被重新锁存。

启动程序存储器中包含内嵌的 Bootloader 程序，可提供 flash 编程功能，通过 USART、I²C、SPI、CAN 或 USB 接口对 flash 进行重新编程；也可以提供通信协议栈等额外的固件，可被软件开发人员通过 API 调用。

1.2 寄存器描述缩写说明

表 1-4 寄存器描述缩写说明

寄存器类型	说明
rw	可以读或写这些位
ro	只能读这些位
wo	只能写这些位；如果读这些位，则返回它们的复位值
rrc	可以读，读取这些位时，自动清除这些位
rw0c	可以读并写'0'清除这些位，写'1'将不对该位产生影响
rw1c	可以读并写'1'清除这些位，写'0'将不对该位产生影响
rw1s	可以读并写'1'设置这些位，写'0'将不对该位产生影响
tog	可以读，写'1'将翻转此位值，写'0'将不对该位产生影响
rwt	可以读，写任何值时，将触发事件
resd	保留

1.3 器件特征信息

表 1-5 器件特征信息相关寄存器地址和复位值

寄存器简称	基地址	复位值
F_SIZE	0x1FFF F7E0	0XXXXX
UID[31:0]	0x1FFF F7E8	0XXXXX XXXX
UID[63:32]	0x1FFF F7EC	0XXXXX XXXX
UID[95:64]	0x1FFF F7F0	0XXXXX XXXX

1.3.1 闪存容量寄存器

闪存容量寄存器提供该芯片闪存容量信息，用户可透过该寄存器取得闪存容量。

域	简称	复位值	类型	功能
位 15: 0	F_SIZE	0XXXXX	ro	闪存容量，以 KByte 为单位 例如：0x0040 = 64KByte

1.3.2 器件电子签名

器件电子签名包含产品容量信息和器件唯一 ID（96 位 UID），它位于闪存的信息区块中。96 位器件唯一 ID 对任何器件来说都是独一无二的，且用户不可更改。ID 可以用来作为下列用途：

- 序列号；例如 USB 字串序列
- 或者做为密钥的一部分

域	简称	复位值	类型	功能
位 31: 0	UID[31: 0]	0XXXXX XXXX	ro	UID 的 bit31 到 bit0 信息

域	简称	复位值	类型	功能
位 31: 0	UID[63: 32]	0XXXXX XXXX	ro	UID 的 bit63 到 bit32 信息

域	简称	复位值	类型	功能
位 31: 0	UID[95: 64]	0XXXXX XXXX	ro	UID 的 bit95 到 bit64 信息

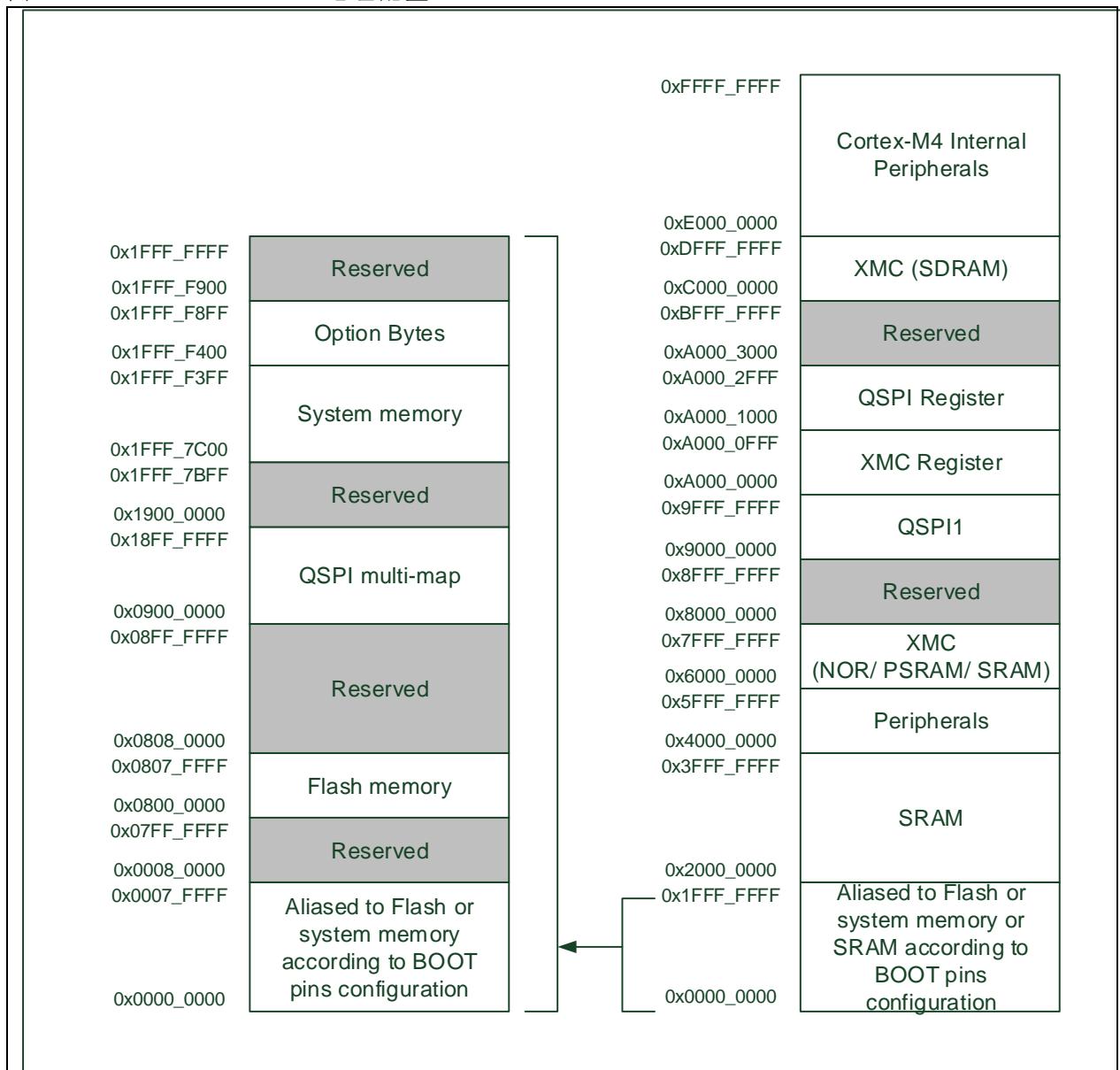
注：UID[95:88]为 Series ID，为 0x15 (AT32F455)、0x16 (AT32F456) 和 0x17 (AT32F457)。

2 存储器资源

2.1 内部存储器地址映射

芯片内部存储器包括程序存储器 flash，数据存储器 SRAM，外设寄存器和内核寄存器等。各区域地址映射如下图：

图 2-1 AT32F455/456/457 地址配置



2.2 Flash存储器

AT32F455/456/457 系列提供最大 512KB 的片上闪存，支持单周期最大 32 位读取操作。闪存存储器由闪存控制器操作，有关闪存控制器的操作与寄存器配置信息请参考第 5 章节。

闪存存储结构 (512K)

主存储器只有闪存容量为 512K 字节的片 1 闪存，包含 256 个扇区，每扇区大小为 2K 字节。

表 2-1 512KB 闪存模块的组织

结构		名称	地址范围	
主存储器	片 1 (Bank1) 512KB	扇区 0	0x0800 0000 – 0x0800 07FF	
		扇区 1	0x0800 0800 – 0x0800 0FFF	
		扇区 2	0x0800 1000 – 0x0800 17FF	
		
		扇区 255	0x0807 F800 – 0x0807 FFFF	
信息块		启动程序代码区 26KB	0x1FFF 7C00 – 0x1FFF E3FF	
		OTP 4KB	0x1FFF E400 – 0x1FFF F3FF	
		LOCK 128B	0x1FFF F500 – 0x1FFF F57F	
		用户系统数据区 512B	0x1FFF F800 – 0x1FFF F9FF	

闪存存储结构 (256K)

主存储器只有闪存容量为 256K 字节的片 1 闪存，包含 128 个扇区，每扇区大小为 2K 字节。

表 2-2 256KB 闪存模块的组织

结构		名称	地址范围	
主存储器	片 1 (Bank1) 256KB	扇区 0	0x0800 0000 – 0x0800 07FF	
		扇区 1	0x0800 0800 – 0x0800 0FFF	
		扇区 2	0x0800 1000 – 0x0800 17FF	
		
		扇区 127	0x0803 F800 – 0x0803 FFFF	
信息块		启动程序代码区 26KB	0x1FFF 7C00 – 0x1FFF E3FF	
		OTP 4KB	0x1FFF E400 – 0x1FFF F3FF	
		LOCK 128B	0x1FFF F500 – 0x1FFF F57F	
		用户系统数据区 512B	0x1FFF F800 – 0x1FFF F9FF	

2.3 SRAM存储器

AT32F455/456/457 系列内置最高可达 144K 字节的片上 SRAM，起始地址为 0x2000_0000。它可以以字节、半字（16 位）或全字（32 位）访问。

用户可配置闪存用户数据系统区的 nRAM_PRT_CHK 位，选择使能或关闭 SRAM 的奇校验。一旦使能 SRAM 奇校验，SRAM 可使用的容量最高将只有 128K 字节，另外 16K 字节将作为存放奇校验结果使用。并且当写入数据到 SRAM 中时，硬件自动以 1 字节为单位计算奇校验，并把产生的 1 位奇校验位存放进 SRAM 中。当读取时，硬件自动检查奇校验结果的正确性，一旦错误，将置起 NMI，并且把错误状态反映到 SCFG_CFG2 位 8。使能 SRAM_OPERR_LK 位可以把 SRAM 奇校验错误状态连接到 TMR1/TMR9/10/11/13/14 的刹车输入上。

注意：SRAM 奇校验仅针对 SRAM 前 64KB。当使能 SRAM 奇校验检测时，需要初始化整个 SRAM，防止误产生奇校验错误。

2.4 外设地址映射

表 2-3 各外设起始地址

总线	起始地址	外设
AHB	0xC000 0000 - 0xDFFF FFFF	XMC_MEM (SDRAM)
	0xE000 0000 - 0xFFFF FFFF	保留
	0xA000 1400 - 0xBFFF FFFF	保留
	0xA000 1000 - 0xA000 13FF	QSPI1 Register
	0xA000 0000 - 0xA000 0FFF	XMC Register
	0x9000 0000 - 0x9FFF FFFF	QSPI1_MEMORY
	0x6000 0000 - 0x8FFF FFFF	XMC_MEM
	0x5004 0000 - 0x5FFF FFFF	保留
	0x5000 0000 - 0x5003 FFFF	OTG_FS1
	0x4008 0000 - 0x4FFF FFFF	保留
	0x4002 A000 - 0x4007 FFFF	保留
	0x4002 8000 - 0x4002 9FFF	EMAC
	0x4002 6800 - 0x4002 7FFF	保留
	0x4002 6400 - 0x4002 67FF	DMA2
	0x4002 6000 - 0x4002 63FF	DMA1
	0x4002 5000 - 0x4002 5FFF	保留
	0x4002 4C00 - 0x4002 4FFF	SDIO1
	0x4002 4000 - 0x4002 4BFF	保留
	0x4002 3C00 - 0x4002 3FFF	闪存存储器接口 (FLASH)
	0x4002 3800 - 0x4002 3BFF	时钟和复位管理 (CRM)
	0x4002 3400 - 0x4002 37FF	保留
	0x4002 3000 - 0x4002 33FF	CRC
	0x4002 2000 - 0x4002 2FFF	保留
	0x4002 1C00 - 0x4002 1FFF	GPIO 端口 H
	0x4002 1800 - 0x4002 1BFF	GPIO 端口 G
	0x4002 1400 - 0x4002 17FF	GPIO 端口 F
	0x4002 1000 - 0x4002 13FF	GPIO 端口 E
	0x4002 0C00 - 0x4002 0FFF	GPIO 端口 D

	0x4002 0800 - 0x4002 0BFF	GPIO 端口 C
	0x4002 0400 - 0x4002 07FF	GPIO 端口 B
	0x4002 0000 - 0x4002 03FF	GPIO 端口 A
	0x4001 8000 - 0x4001 FFFF	保留
	0x4001 7C00 - 0x4001 7FFF	I ² S3EXT
	0x4001 7800 - 0x4001 7BFF	I ² S2EXT
	0x4001 7400 - 0x4001 77FF	ACC
	0x4001 7000 - 0x4001 73FF	保留
	0x4001 6C00 - 0x4001 6FFF	保留
	0x4001 6800 - 0x4001 6BFF	保留
	0x4001 6400 - 0x4001 67FF	保留
	0x4001 6000 - 0x4001 63FF	保留
	0x4001 5C00 - 0x4001 5FFF	保留
	0x4001 5800 - 0x4001 5BFF	保留
	0x4001 5400 - 0x4001 57FF	保留
	0x4001 5000 - 0x4001 53FF	I ² SF5
	0x4001 4C00 - 0x4001 4FFF	保留
	0x4001 4800 - 0x4001 4BFF	TMR11 定时器
	0x4001 4400 - 0x4001 47FF	TMR10 定时器
APB2	0x4001 4000 - 0x4001 43FF	TMR9 定时器
	0x4001 3C00 - 0x4001 3FFF	EXINT
	0x4001 3800 - 0x4001 3BFF	SCFG
	0x4001 3400 - 0x4001 37FF	SPI4/I ² S4
	0x4001 3000 - 0x4001 33FF	SPI1/I ² S1
	0x4001 2C00 - 0x4001 2FFF	保留
	0x4001 2800 - 0x4001 2BFF	保留
	0x4001 2400 - 0x4001 27FF	保留
	0x4001 2000 - 0x4001 23FF	ADC1_2
	0x4001 1C00 - 0x4001 1FFF	保留
	0x4001 1800 - 0x4001 1BFF	保留
	0x4001 1400 - 0x4001 17FF	USART6
	0x4001 1000 - 0x4001 13FF	USART1
	0x4001 0C00 - 0x4001 0FFF	保留
	0x4001 0800 - 0x4001 0BFF	保留
	0x4001 0400 - 0x4001 07FF	TMR8 定时器
	0x4001 0000 - 0x4001 03FF	TMR1 定时器
	0x4000 8000 - 0x4000 FFFF	保留
	0x4000 7C00 - 0x4000 7FFF	USART8
APB1	0x4000 7800 - 0x4000 7BFF	USART7
	0x4000 7400 - 0x4000 77FF	DAC
	0x4000 7000 - 0x4000 73FF	电源控制 (PWC)

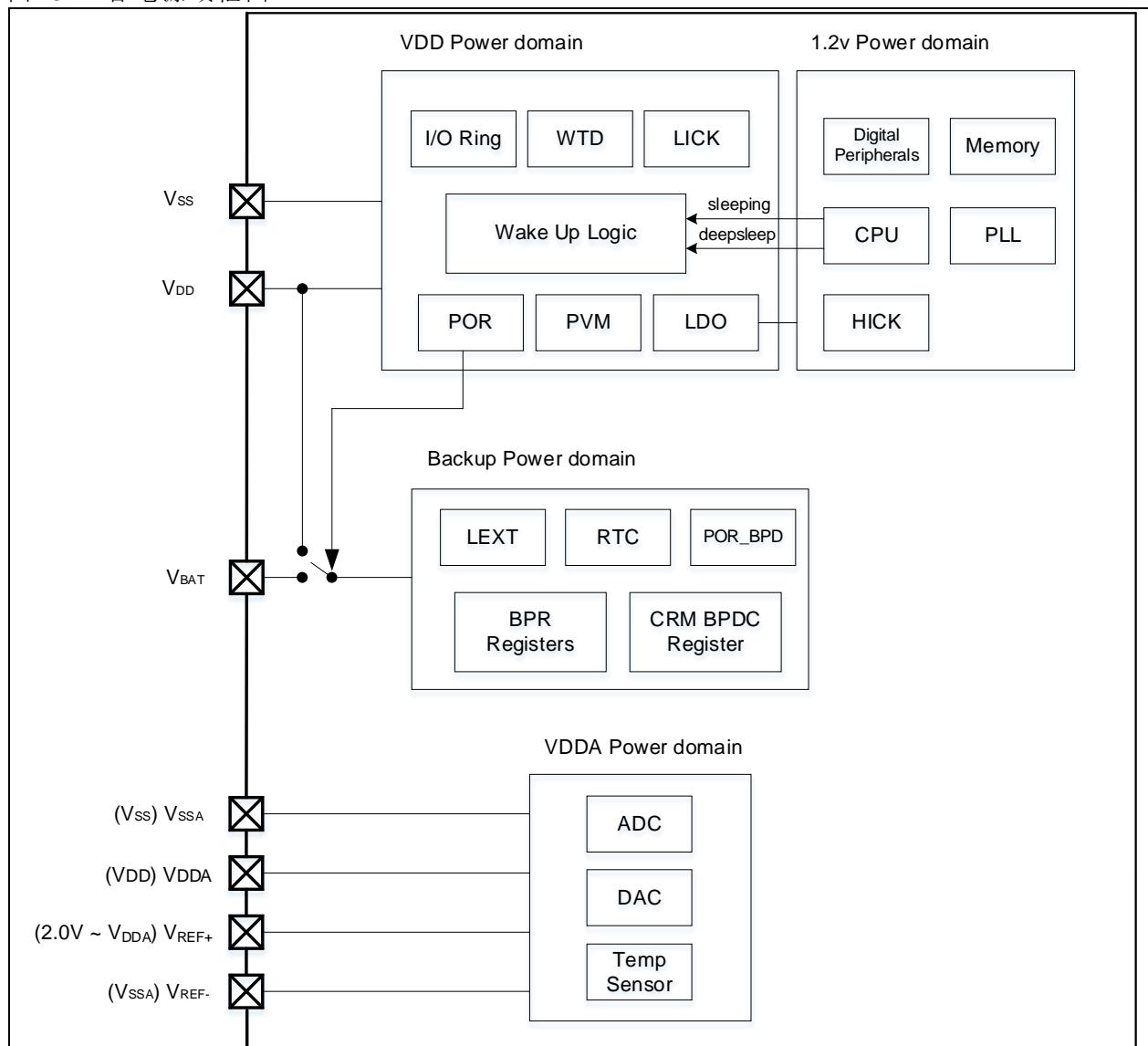
0x4000 6C00 - 0x4000 6FFF	CAN3
0x4000 6800 - 0x4000 6BFF	CAN2
0x4000 6400 - 0x4000 67FF	CAN1
0x4000 6000 - 0x4000 63FF	保留
0x4000 5C00 - 0x4000 5FFF	I ² C3
0x4000 5800 - 0x4000 5BFF	I ² C2
0x4000 5400 - 0x4000 57FF	I ² C1
0x4000 5000 - 0x4000 53FF	USART5
0x4000 4C00 - 0x4000 4FFF	USART4
0x4000 4800 - 0x4000 4BFF	USART3
0x4000 4400 - 0x4000 47FF	USART2
0x4000 4000 - 0x4000 43FF	保留
0x4000 3C00 - 0x4000 3FFF	SPI3/I ² S3
0x4000 3800 - 0x4000 3BFF	SPI2/I ² S2
0x4000 3400 - 0x4000 37FF	保留
0x4000 3000 - 0x4000 33FF	看门狗 (WDT)
0x4000 2C00 - 0x4000 2FFF	窗口看门狗 (WWDT)
0x4000 2800 - 0x4000 2BFF	ERTC
0x4000 2400 - 0x4000 27FF	保留
0x4000 2000 - 0x4000 23FF	TMR14 定时器
0x4000 1C00 - 0x4000 1FFF	TMR13 定时器
0x4000 1800 - 0x4000 1BFF	TMR12 定时器
0x4000 1400 - 0x4000 17FF	TMR7 定时器
0x4000 1000 - 0x4000 13FF	TMR6 定时器
0x4000 0C00 - 0x4000 0FFF	TMR5 定时器
0x4000 0800 - 0x4000 0BFF	TMR4 定时器
0x4000 0400 - 0x4000 07FF	TMR3 定时器
0x4000 0000 - 0x4000 03FF	TMR2 定时器

3 电源控制 (PWC)

3.1 简介

AT32F455/456/457 系列设备工作电压范围为 2.4V 至 3.6V，正常工作温度范围为 -40~+105 °C。AT32F455/456/457 系列设备为了降低功耗，使用户可以在 CPU 运行时间要求、速度和功耗进行折中取舍，提供了三种省电模式——睡眠模式，深度睡眠模式和待机模式。AT32F455/456/457 系列设备有三个电源域——VDD/VDDA 域，1.2V 域和电池供电域。其中 VDD/VDDA 域由电源直接供电，1.2V 域由 VDD/VDDA 域中嵌入的 LDO 供电，电池供电域由 V_{BAT} 引脚供电。

图 3-1 各电源域框图



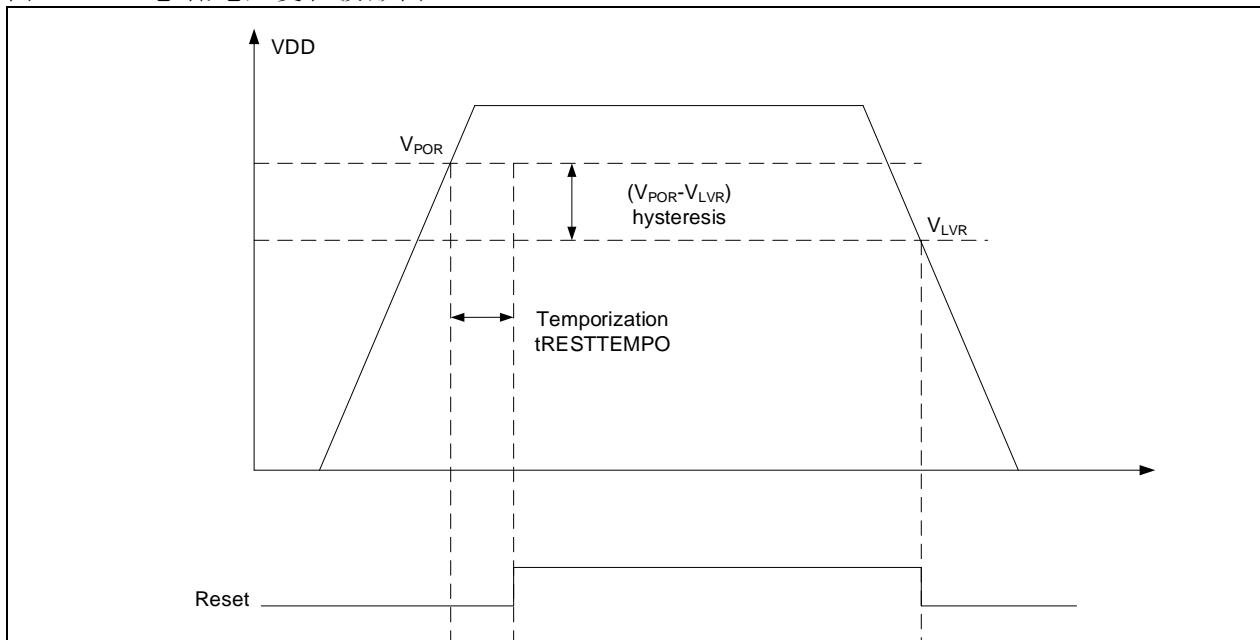
3.2 主要特点

- 具备三个电源域：VDD/VDDA 域、1.2V 内核域和电池供电域。
- 支持三种省电模式：睡眠模式、深度睡眠模式和待机模式。
- 内建电压调节器，提供 1.2V 给内核域。
- 提供电压监测器，能在电压低于阈值或高于阈值时产生中断或事件。
- 当 VDD 供电关闭时，电池供电域由电池 V_{BAT} 供电。
- VDD/VDDA 采用独立的数字和模拟地，用于隔离电源噪声。

3.3 上电下电复位

VDD/VDDA 域内置一个 POR 模拟模块用于产生电源复位，当 VDD 由 0V 上升至工作电压过程中，电源复位信号在 V_{POR} 时刻被上电释放。当 VDD 由工作电压下降至 0V 过程中，电源复位信号在 V_{LVR} 时刻被低电压复位。上电复位过程，复位信号的释放相较于 VDD 升压过程存在一定的时间延迟，同时上电低电压复位具有一定迟滞。

图 3-2 上电/低电压复位波形图

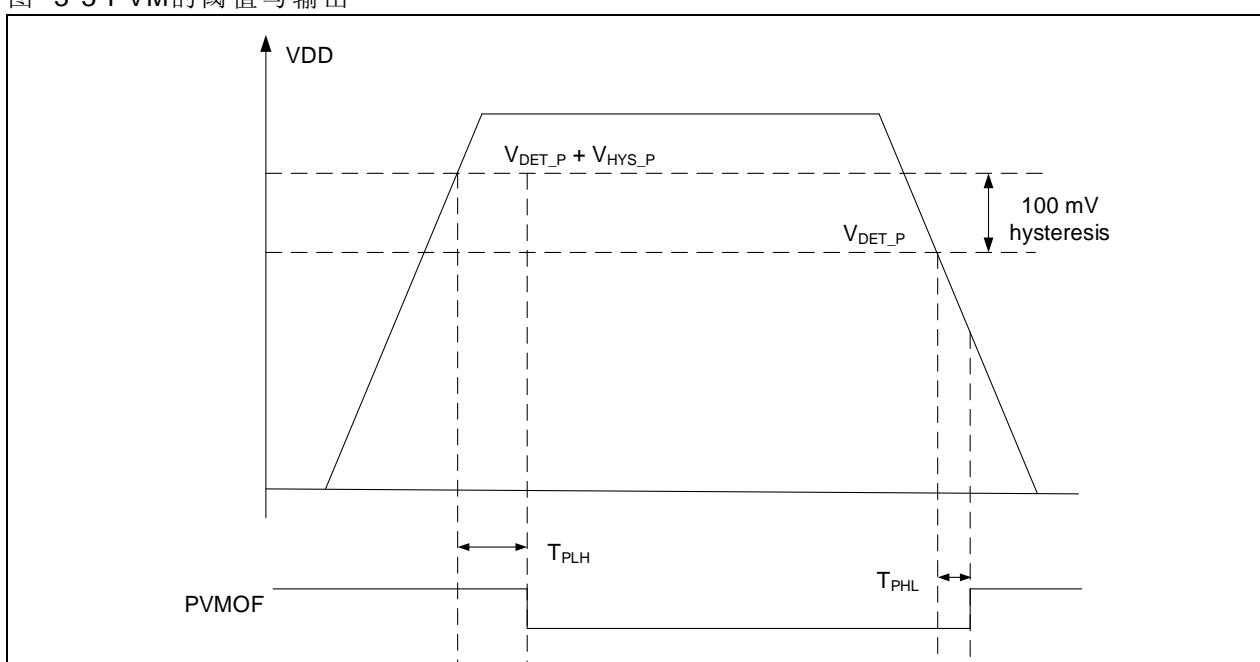


3.4 电压监测器 (PVM)

电压监测器 PVM 主要用来监控供电电源的跳变，可通过电源控制寄存器 (PWC_CTRL) 中的 PVMEN 位开启电压监测功能，并通过 PVMSEL[2: 0]来选择监控阈值。

电压监测器开启后，电源控制及状态寄存器 (PWC_CTRLSTS) 中的 PVMOF 位会指示 VDD 与设定阈值比较的结果，迟滞电压 VHYS_P 为 100mV。当 VDD 越过 PVM 阈值边界时，产生的 PVMOF 位电平变化可以通过外部中断第 16 号线产生 PVM 中断。

图 3-3 PVM 的阈值与输出



3.5 电源域划分

1.2V 域

1.2V 内核域包括 CPU 内核、存储器 SRAM、内嵌数字外设以及时钟锁相环 PLL，由 LDO（电压调节器）供电。

VDD/VDDA 域

VDD/VDDA 域包括 VDD 域和 VDDA 域两部分。VDD 域包括 I/O 电路、省电模式唤醒电路、看门狗 WDT、上电/低电压复位(POR/LVR)、电压调节器 LDO 以及除 PC13、PC14 和 PC15 之外的所有 PAD 电路等。VDDA 域包括 ADC/DAC (AD/DA 转换器)、温度传感器 Temp Sensor 等。

一般来说，为保证低电压时 ADC/DAC 的高精度，数字电路由 VDD 供电，模拟电路由 VDDA 独立供电，在 64 PIN 封装及以下型号中，外部参考电压 VREF+连接至 VDDA 管脚，VREF-连接至 VSSA 管脚。

在芯片正常工作(Run Mode)时，电压调节器(LDO)为 1.2V 域供电，默认输出 1.2V 电压。可配置 LDO 调校寄存器(PWC_LDOOV)来选择 LDO 的输出电压，不同的输出电压，系统可以工作的最高工作频率不同，详细参见 AT32F455/456/457 数据手册。限定在系统时钟使用 HEXT 或 HICK 时方可改变 LDO 的输出电压。

LDO 输出电压调节

- 1) 系统时钟切换至 HICK 或 HEXT
- 2) 修改 LDO 电压 (LDOOVSEL[1: 0])
- 3) 设置闪存性能选择寄存器 (FLASH_PSR)
- 4) 设置 PLL 相关寄存器至目标频率，开启 PLL，等待 PLL_STBL
- 5) 设置 AHB 及 APB 预除频系数
- 6) 若 PLL 频率大于 108MHz，打开顺滑切换
- 7) 切换系统时钟至 PLL

电池供电域

电池供电域包括 ERTC 电路、LEXT 振荡器、PC13、PC14 以及 PC15。电池供电域由 VDD 或 VBAT 引脚供电，当 VDD 主电源被切断时，电池供电域自动切换至 VBAT 引脚供电，以保障 ERTC 正常工作。

- 1) 当电池供电域由 VDD 供电时，PC13 可以作为通用 I/O 口、TAMPER 引脚、ERTC 校准时钟、ERTC 闹钟或秒输出，PC14 和 PC15 可以用于 GPIO 或 LEXT 引脚。(PC13 至 PC15 作为 I/O 口的速度必须限制在 2MHz 以下，最大负载为 30pF，而且这些 I/O 口绝对不能当作电流源)。
- 2) 当电池供电域由 VBAT 供电时，PC13 可以作为 TAMPER 引脚、ERTC 闹钟或秒输出，PC14 和 PC15 只能用于 LEXT 引脚。

电池供电域的电源开关不会因为 VDD 在上升阶段或是因为 VDD 低电压复位而断开与 VBAT 的连接。当主电源上 VDD 上电较快，电源开关还未切换至主电源 VDD 时，为防止电流从 VDD 注入到 VBAT，推荐在 VDD 与 VBAT 之间接一个低压降二极管。若应用中没有外部电池，VBAT 最好连接一个 100nF 的陶瓷滤波电容并在外部连接到 VDD。

3.6 省电模式

当 CPU 无需继续运行时，AT32F455/456/457 支持三种低功耗模式（睡眠模式、深度睡眠模式、待机模式）可以实现更低的功耗。用户可以在启动时间，唤醒源，电源消耗等方面进行折中。此外在运行模式下，还可以通过降低系统时钟或关闭 APB 和 AHB 总线上未被使用的外设时钟来降低功耗。

睡眠模式 (Sleep Mode)

执行 WFI 或 WFE 指令可以进入睡眠状态。结合 Cortex®-M4F 系统控制寄存器中的 SLEEPONEXIT 位的设定，提供两种进入睡眠模式的机制：

SLEEP-NOW 模式

当 SLEEPDEEP=0, SLEEPONEXIT=0 时，执行 WFI 或 WFE 指令，此时可立即进入睡眠模式。

SLEEP-ON-EXIT 模式

当 SLEEPDEEP=0, SLEEPONEXIT=1 时，执行 WFI 指令，此时当系统从最低优先级的中断处理程序中退出时，可立即进入睡眠模式。

在睡眠模式下，CPU 时钟关闭，其他时钟均正常工作，电压调节器正常工作，所有的 I/O 管脚都保持它

们在运行模式时的状态，调节器 LDO 以正常功耗模式提供 1.2V 电源（CPU 内核、内存和内嵌外设）。

- 1) 执行 WFI 指令进入睡眠模式时，只要产生外设中断，都能使系统退出睡眠模式。
- 2) 执行 WFE 指令进入睡眠模式时，存在两种方式的唤醒事件，使系统退出睡眠模式：
 - 使能任一外设中断（未在 NVIC 中使能）且使能 SEVONPEND 位可以产生唤醒事件。系统唤醒后，需清除外设中断挂起位和 NVIC 通道挂起位。
 - 配置内部 EXINT 线为事件模式来产生唤醒事件。

从执行 WFE 指令进入睡眠模式唤醒所需的时间最短，因为没有时间损失在中断的进入或退出上。

深度睡眠模式（Deepsleep Mode）

通过设置 Cortex®-M4F 系统控制寄存器中的 SLEEPDEEP 位，清除电源控制寄存器（PWC_CTRL）中的 LPSEL 位，再执行 WFI 或 WFE 指令即可进入深度睡眠模式。

还可以通过设置电源控制寄存器（PWC_CTRL）中 VRSEL 位选择深度睡眠模式下电压调节器的工作状态。当 VRSEL=0，电压调节器正常工作，当 VRSEL=1，电压调节器处于低功耗模式。

另外，深度睡眠模式下，在设置 VRSEL=1，控制电压调节器处于低功耗模式的基础上，可以通过设置内部电压调节器额外低功耗模式使能位（VREXLPE），进一步降低深度睡眠模式下整个系统的功耗。

在深度睡眠模式下，所有 1.2V 时钟关闭，HICK 和 HEXT 振荡器都被关闭，电压调节器以正常工作或低功耗工作状态给 1.2V 域供电，所有 I/O 管脚都保持它们在运行模式时的状态，SRAM 和寄存器内容保持。

- 1) 执行 WFI 指令进入深度睡眠模式，任一外部中断线在中断模式下产生的中断，即可使系统退出深度睡眠模式。
- 2) 如果执行 WFE 指令进入深度睡眠模式，任一外部中断线在事件模式下产生的事件，即可使系统退出深度睡眠模式。

系统从深度睡眠模式退出时，HICK RC 振荡器开启并在稳定后被选为系统时钟。当电压调节器处于低功耗模式时，退出深度睡眠模式时，需要额外等待电压调节器稳定，从而会增加一段额外的唤醒时间。

低功耗 deepsleep LDO 电压调节流程（注：sleep 和 standby 没有此限制）

- 1) 系统时钟切换至 HICK
- 2) 修改 LDO 电压（LDOOVSEL[1: 0]）为 1.0V
- 3) 配置 LDO 是否工作在低功耗模式（VRSEL）
- 4) 系统进入 deepsleep 状态
- 5) 系统退出 deepsleep 状态（满足唤醒条件后）
- 6) 修改 LDO 电压（LDOOVSEL[1: 0]）
- 7) 设置闪存性能选择寄存器（FLASH_PSR）
- 8) 若 PLL 时钟源为 HEXT，需要开启 HEXT 并等待 HEXTSTBL
- 9) 设置 PLL 相关寄存器至目标频率
- 10) 开启 PLL，等待 PLL_STBL
- 11) 设置 AHB 及 APB 预除频系数
- 12) 若 PLL 频率大于 108MHz，打开顺滑切换
- 13) 切换系统时钟至 PLL

注：若期望低功耗唤醒后保持进低功耗前的时钟状态，前述 3/9/11 步骤可省略。

待机模式（Standby Mode）

待机模式可最大限度的降低系统功耗，在该模式下，电压调节器关闭，只有电池供电的寄存器和待机电路维持供电，其他的 1.2V 供电区域，PLL、HICK 和 HEXT 振荡器都被断电。寄存器和 SRAM 中的内容也会丢失。

通过设置 Cortex®-M4F 系统控制寄存器中的 SLEEPDEEP 位，设置电源控制寄存器（PWC_CTRL）中 LPSEL 位，并清除电源控制及状态寄存器（PWC_CTRLSTS）中的所有唤醒事件标志位的情况下，最后执行 WFI 或 WFE 指令即可进入待机模式。

在待机模式下，除了复位管脚、被设置为防侵入或校准输出时的 TAMPER 管脚和被使能的唤醒管脚之外，所有的 I/O 管脚处于高阻态。

当产生 WKUP 管脚的有效沿、ERTC 闹钟事件的上升沿、ERTC 入侵事件、ERTC 时间戳、ERTC 周期性自动唤醒、NRST 管脚上外部复位、WDT 复位时，微控制器将退出待机模式。

调试配置

默认情况下，在进行调试时，微处理器一旦进入深度睡眠或待机模式，会因为 Cortex®-M4F 的内核失去

了时钟而失去调试连接。只需通过设置 DEBUG 控制寄存器 (DEBUG_CTRL) 中的某些配置位，就可以在低功耗模式下继续调试软件。

3.7 PWC 寄存器

可以用半字 (16 位) 或字 (32 位) 的方式操作这些外设寄存器。

表 3-1 PWC 寄存器的映像和复位值

寄存器简称	基址偏移量	复位值
PWC_CTRL	0x00	0x0000 0000
PWC_CTRLSTS	0x04	0x0000 0000
PWC_CLR	0x08	0x0000 0000
PWC_LDOOV	0x10	0x0000 0012

3.7.1 电源控制寄存器 (PWC_CTRL)

域	简称	复位值	类型	功能
位 31	保留	0x0	resd	保持默认值。
位 30	SWP7POL	0x0	rw	待机唤醒引脚 7 极性选择 (Standby wake-up pin7 polarity) 0: 上升沿触发 1: 下降沿触发
位 29	SWP6POL	0x0	rw	待机唤醒引脚 6 极性选择 (Standby wake-up pin6 polarity) 0: 上升沿触发 1: 下降沿触发
位 28	SWP5POL	0x0	rw	待机唤醒引脚 5 极性选择 (Standby wake-up pin5 polarity) 0: 上升沿触发 1: 下降沿触发
位 27	SWP4POL	0x0	rw	待机唤醒引脚 4 极性选择 (Standby wake-up pin4 polarity) 0: 上升沿触发 1: 下降沿触发
位 23	保留	0x0	resd	保持默认值。
位 25	SWP2POL	0x0	rw	待机唤醒引脚 2 极性选择 (Standby wake-up pin2 polarity) 0: 上升沿触发 1: 下降沿触发
位 24	SWP1POL	0x0	rw	待机唤醒引脚 1 极性选择 (Standby wake-up pin1 polarity) 0: 上升沿触发 1: 下降沿触发
位 23	保留	0x0	resd	保持默认值。
位 22	SWPEN7	0x0	rw	待机唤醒引脚 7 使能 (Standby wake-up pin6 enable) 0: 关闭 (该引脚可用作通用 I/O)； 1: 开启 (该引脚被强置为输入下拉模式，且无法再用作通用 I/O)。 注: 在系统复位时硬件将清除这一位。
位 21	SWPEN6	0x0	rw	待机唤醒引脚 6 使能 (Standby wake-up pin6 enable) 0: 关闭 (该引脚可用作通用 I/O)； 1: 开启 (该引脚被强置为输入下拉模式，且无法再用作通用 I/O)。 注: 在系统复位时硬件将清除这一位。
位 20	SWPEN5	0x0	rw	待机唤醒引脚 5 使能 (Standby wake-up pin5 enable) 0: 关闭 (该引脚可用作通用 I/O)；

				1: 开启（该引脚被强置为输入下拉模式，且无法再用作通用 I/O）。 注：在系统复位时硬件将清除这一位。
位 19	SWPEN4	0x0	rw	待机唤醒引脚 4 使能（Standby wake-up pin4 enable） 0: 关闭（该引脚可用作通用 I/O）； 1: 开启（该引脚被强置为输入下拉模式，且无法再用作通用 I/O）。 注：在系统复位时硬件将清除这一位。
位 18	保留	0x0	resd	保持默认值。 待机唤醒引脚 2 使能（Standby wake-up pin2 enable） 0: 关闭（该引脚可用作通用 I/O）； 1: 开启（该引脚被强置为输入下拉模式，且无法再用作通用 I/O）。 注：在系统复位时硬件将清除这一位。
位 17	SWPEN2	0x0	rw	待机唤醒引脚 1 使能（Standby wake-up pin1 enable） 0: 关闭（该引脚可用作通用 I/O）； 1: 开启（该引脚被强置为输入下拉模式，且无法再用作通用 I/O）。 注：在系统复位时硬件将清除这一位。
位 16	SWPEN1	0x0	rw	0: 关闭（该引脚可用作通用 I/O）； 1: 开启（该引脚被强置为输入下拉模式，且无法再用作通用 I/O）。 注：在系统复位时硬件将清除这一位。
位 15: 9	保留	0x0	resd	保持默认值。 电池供电区域的写入使能（Battery powered domain write enable） 0: 关闭； 1: 开启。 注： 复位后，电池供电区域禁止写入。要对电池供电区域进行写操作的话，需先设置这位为允许写入状态。
位 8	BPWEN	0x0	rw	电压监测临界值选择（Power voltage monitoring boundary select） 000: 未用，禁止配置。 001: 2.3V 010: 2.4V 011: 2.5V 100: 2.6V 101: 2.7V 110: 2.8V 111: 2.9V
位 7: 5	PVMSEL	0x0	rw	电压监测使能（Power voltage monitoring enable） 0: 关闭； 1: 开启。
位 4	PVMEN	0x0	rw	保持默认值。
位 3: 2	保留	0x0	resd	SLEEPDEEP 状态下的低功耗模式选择位（Low power mode select when Cortex®-M4F sleepdeep） 0: 进入 DEEPSLEEP 模式； 1: 进入待机模式。
位 1	LPSEL	0x0	rw	DEEPSLEEP 模式下电压调节器状态选择（Voltage regulator state select when deepsleep mode） 0: 正常开启； 1: 低功耗模式。
位 0	VRSEL	0x0	rw	

3.7.2 电源控制及状态寄存器 (PWC_CTRLSTS)

域	简称	复位值	类型	功能
位 31: 16	保留	0x000000	resd	保持默认值。 待机唤醒事件标志 (Standby wake-up event flag) 0: 无唤醒事件产生; 1: 有唤醒事件产生。 注:
位 15	SWEF	0x0	ro	该位被硬件置起 (产生唤醒事件时), 由 POR/LVR 将其清零或唤醒事件被清除。 唤醒事件将由以下几种情况产生: 当产生使能中断的 RTC 闹钟/入侵/时间戳/周期性自动唤醒事件时, 将对应置位此标志。
位 14: 10	保留	0x00	resd	保持默认值。
位 9	PVMOF	0x0	ro	电源电压检测输出标志 (Power voltage monitoring output flag) 0: 电源电压高于临界值; 1: 电源电压低于临界值。 注: 待机模式下电压监测停止工作。
位 8	SEF	0x0	ro	进入待机模式标志 (Standby mode entry flag) 0: 未进过待机模式; 1: 有进过待机模式。 注: 该位被硬件置起 (进入待机模式时), 由 POR/LVR 或写 CLSEF 位将其清零。
位 7	保留	0x0	resd	保持默认值。 待机唤醒事件 7 标志 (Standby wake-up7 event flag) 0: 无唤醒事件产生; 1: 有唤醒事件产生。 注:
位 6	SW7EF	0x0	ro	该位被硬件置起 (产生唤醒事件时), 由 POR/LVR 或写 CLSW7EF 位将其清零。 唤醒事件将由以下几种情况产生: 在待机唤醒引脚 7 出现上升/下降沿时, 将产生唤醒事件; 待机唤醒引脚保持高电平期间使能该待机唤醒引脚, 将产生唤醒事件。
位 5	SW6EF	0x0	ro	待机唤醒事件 6 标志 (Standby wake-up6 event flag) 0: 无唤醒事件产生; 1: 有唤醒事件产生。 注:
位 4	SW5EF	0x0	ro	该位被硬件置起 (产生唤醒事件时), 由 POR/LVR 或写 CLSW5EF 位将其清零。 唤醒事件将由以下几种情况产生: 在待机唤醒引脚 6 出现上升/下降沿时, 将产生唤醒事件; 待机唤醒引脚保持高电平期间使能该待机唤醒引脚, 将产生唤醒事件。
位 3	SW4EF	0x0	ro	待机唤醒事件 5 标志 (Standby wake-up5 event flag) 0: 无唤醒事件产生; 1: 有唤醒事件产生。 注:
位 2	SW3EF	0x0	ro	该位被硬件置起 (产生唤醒事件时), 由 POR/LVR 或写 CLSW3EF 位将其清零。 唤醒事件将由以下几种情况产生: 在待机唤醒引脚 5 出现上升/下降沿时, 将产生唤醒事件; 待机唤醒引脚保持高电平期间使能该待机唤醒引脚, 将产生唤醒事件。
位 1	SW2EF	0x0	ro	待机唤醒事件 4 标志 (Standby wake-up4 event flag) 0: 无唤醒事件产生; 1: 有唤醒事件产生。 注:
位 0	SW1EF	0x0	ro	该位被硬件置起 (产生唤醒事件时), 由 POR/LVR 或写 CLSW1EF 位将其清零。 唤醒事件将由以下几种情况产生: 在待机唤醒引脚 4 出现上升/下降沿时, 将产生唤醒事件; 待机唤醒引脚保持高电平期间使能该待机唤醒引脚, 将产生唤醒事件。

位 3	SW4EF	0x0	ro	待机唤醒事件 4 标志 (Standby wake-up4 event flag) 0: 无唤醒事件产生; 1: 有唤醒事件产生。 注: 该位被硬件置起 (产生唤醒事件时), 由 POR/LVR 或写 CLSW4EF 位将其清零。 唤醒事件将由以下几种情况产生: 在待机唤醒引脚 4 出现上升/下降沿时, 将产生唤醒事件; 待机唤醒引脚保持高电平期间使能该待机唤醒引脚, 将产生唤醒事件。
位 2	保留	0x0	resd	保持默认值。
位 1	SW2EF	0x0	ro	待机唤醒事件 2 标志 (Standby wake-up2 event flag) 0: 无唤醒事件产生; 1: 有唤醒事件产生。 注: 该位被硬件置起 (产生唤醒事件时), 由 POR/LVR 或写 CLSW2EF 位将其清零。 唤醒事件将由以下几种情况产生: 在待机唤醒引脚 2 出现上升/下降沿时, 将产生唤醒事件; 待机唤醒引脚保持高电平期间使能该待机唤醒引脚, 将产生唤醒事件。
位 0	SW1EF	0x0	ro	待机唤醒事件 1 标志 (Standby wake-up1 event flag) 0: 无唤醒事件产生; 1: 有唤醒事件产生。 注: 该位被硬件置起 (产生唤醒事件时), 由 POR/LVR 或写 CLSW1EF 位将其清零。 唤醒事件将由以下几种情况产生: 在待机唤醒引脚 1 出现上升/下降沿时, 将产生唤醒事件; 待机唤醒引脚保持高电平期间使能该待机唤醒引脚, 将产生唤醒事件。

3.7.3 电源标志清除寄存器 (PWC_CLR)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。 清除 SWEF 标志 (Clear SWEF flag) 0: 无效; 1: 清除 SWEF 标志。 注: 该位在清除 SWEF 后由硬件将其清零, 且任何时刻读取该位返回值均是零。
位 15	CLSWEF	0x0	wo	保持默认值。
位 14: 9	保留	0x00	resd	保持默认值。 清除 SEF 标志 (Clear SEF flag) 0: 无效; 1: 清除 SEF 标志。 注: 该位在清除 SEF 后由硬件将其清零, 且任何时刻读取该位返回值均是零。
位 8	CLSEF	0x0	wo	保持默认值。 清除 SW7EF 标志 (Clear SW7EF flag) 0: 无效; 1: 清除 SW7EF 标志。 注: 该位在清除 SW7EF 后由硬件将其清零, 且任何时刻读取该位返回值均是零。
位 7	保留	0x0	resd	保持默认值。 清除 SW6EF 标志 (Clear SW6EF flag) 0: 无效; 1: 清除 SW6EF 标志。 注: 实际 SW6EF 标志的清除大约需要 2 个系统时钟周期; 该位在清除 SW6EF 后由硬件将其清零, 且任何时刻读取该位返回值均是零。
位 6	CLSW6EF	0x0	wo	保持默认值。 清除 SW5EF 标志 (Clear SW5EF flag) 0: 无效; 1: 清除 SW5EF 标志。 注: 实际 SW5EF 标志的清除大约需要 2 个系统时钟周期; 该位在清除 SW5EF 后由硬件将其清零, 且任何时刻读取该位返回值均是零。
位 5	CLSW5EF	0x0	wo	保持默认值。 清除 SW4EF 标志 (Clear SW4EF flag) 0: 无效; 1: 清除 SW4EF 标志。 注: 实际 SW4EF 标志的清除大约需要 2 个系统时钟周期; 该位在清除 SW4EF 后由硬件将其清零, 且任何时刻读取该位返回值均是零。
位 4	CLSW4EF	0x0	wo	保持默认值。 清除 SW3EF 标志 (Clear SW3EF flag) 0: 无效; 1: 清除 SW3EF 标志。 注: 实际 SW3EF 标志的清除大约需要 2 个系统时钟周期; 该位在清除 SW3EF 后由硬件将其清零, 且任何时刻读取该位返回值均是零。
位 3	CLSW3EF	0x0	wo	保持默认值。 清除 SW2EF 标志 (Clear SW2EF flag) 0: 无效; 1: 清除 SW2EF 标志。 注: 实际 SW2EF 标志的清除大约需要 2 个系统时钟周期; 该位在清除 SW2EF 后由硬件将其清零, 且任何时刻读取该位返回值均是零。
位 2	保留	0x0	resd	保持默认值。 清除 SW1EF 标志 (Clear SW1EF flag) 0: 无效; 1: 清除 SW1EF 标志。 注: 实际 SW1EF 标志的清除大约需要 2 个系统时钟周期; 该位在清除 SW1EF 后由硬件将其清零, 且任何时刻读取该位返回值均是零。
位 1	CLSW1EF	0x0	wo	保持默认值。 清除 SW0EF 标志 (Clear SW0EF flag) 0: 无效; 1: 清除 SW0EF 标志。 注: 实际 SW0EF 标志的清除大约需要 2 个系统时钟周期; 该位在清除 SW0EF 后由硬件将其清零, 且任何时刻读取该位返回值均是零。
位 0	CLSW0EF	0x0	wo	保持默认值。

3.7.4 LDO调校寄存器 (PWC_LDOOV)

域	简称	复位值	类型	功能
位 31: 5	保留	0x0000000	resd	保持默认值。
位 4	VREXLPMEN	0x1	rw	内部电压调压器额外低功耗模式使能位 (Voltage regulator extra low power mode enable) 与电源控制寄存器 (PWC_CTRL) 的 LPSEL 和 VRSEL 位协同工作，在 VRSEL = 1 时，且芯片进入深度睡眠模式时才有效。 0: 内部电压调压器额外低功耗模式关闭。 1: 内部电压调压器额外低功耗模式开启。 注： 如需启动额外低功耗模式，请先配置 VREXLPMEN，再配置 LPSEL 和 VRSEL 位。 开启此位并进入 DEEPSLEEP 模式后，LDO 驱动能力受限，此时禁止开启 DEEPSLEEP_DEBUG 进行调试。
位 3: 2	保留	0x0	resd	保持默认值。
位 1: 0	LDOOVSEL	0x02	rw	内部电压调节器输出电压选择 (LDO output voltage select) 01: 1.1V 10: 1.2V 11: 1.3V

3.7.5 电源校准寄存器 (PWC_VBGTRIM)

偏移地址: 0x20

域	简称	复位值	类型	功能
位 31: 9	保留	0x00000000	resd	保持默认值。
位 12: 8	VGT_TRIM_OTP1	0xXX	ro	VBG 调校值 默认值参考 OTP1 设定
位 7: 5	保留	0x0	resd	保持默认值
位 4: 0	VBG_TRIM	0x00	rw	VBG 调校软件设置位 当软件设置此调校位时， VBG 调校值由软件设定

4 时钟和复位管理 (CRM)

4.1 时钟

AT32F455/456/457 的时钟源包含：HEXT 振荡器时钟，HICK 振荡器时钟，PLL 时钟，LEXT 振荡器时钟和 LICK 振荡器时钟。时钟结构如下：

图 4-1 AT32F455/456 时钟结构图

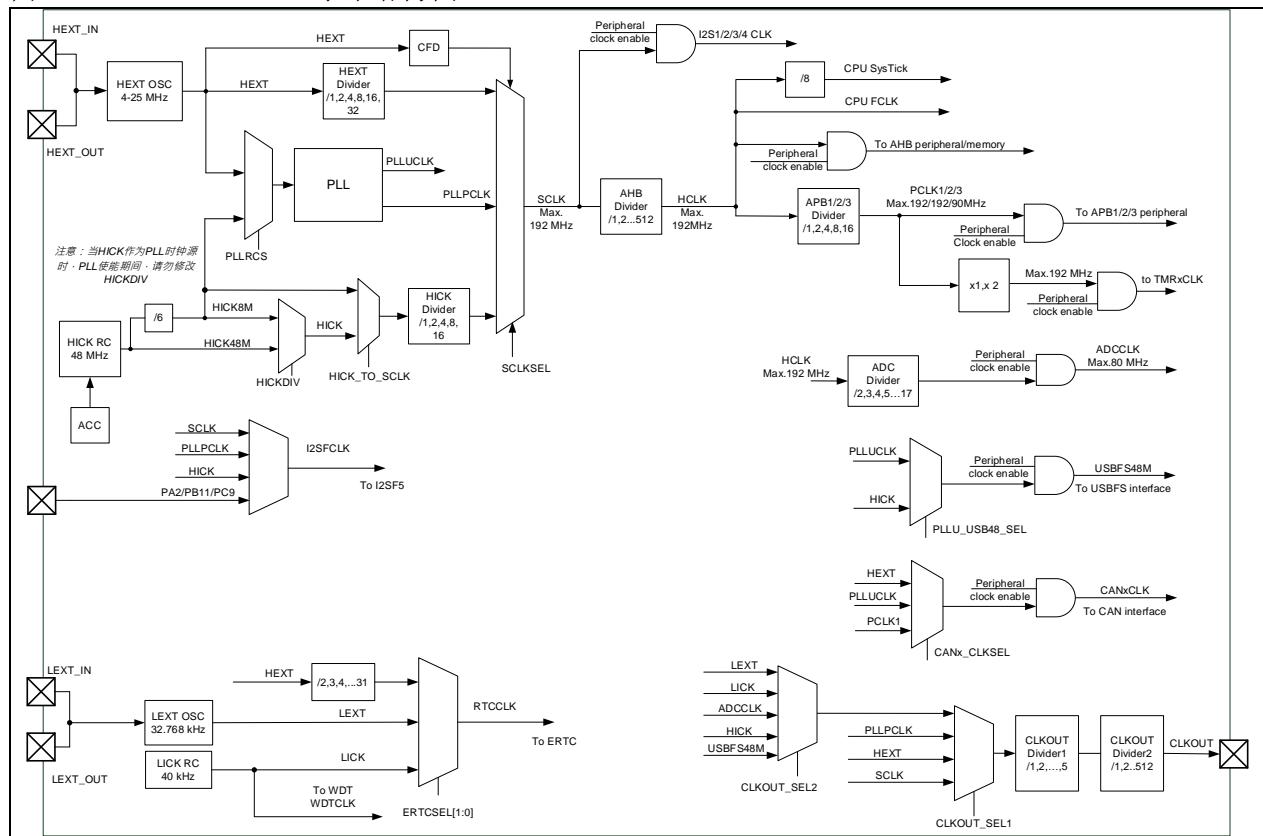
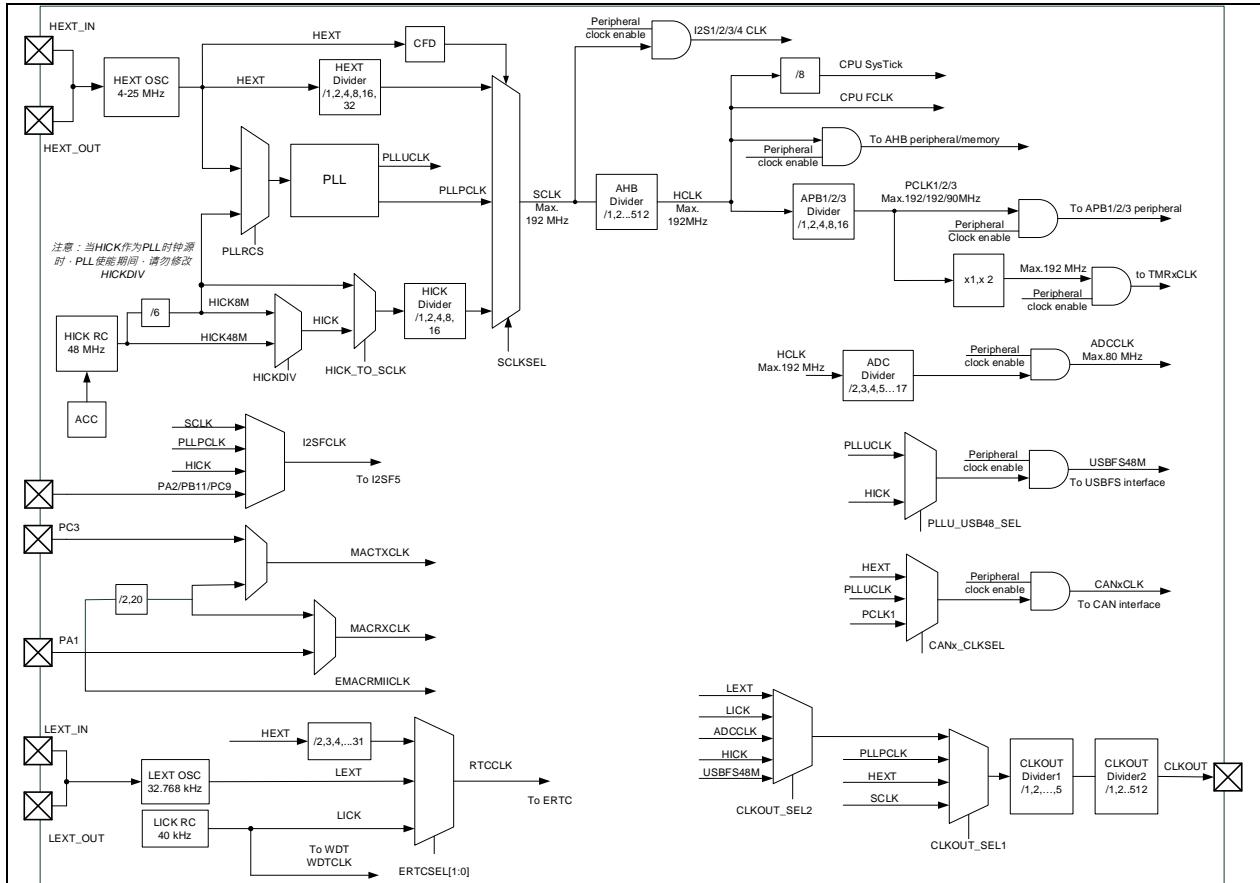


图 4-2 AT32F457时钟结构图



AHB、APB1、APB2 和 APB3 的频率都支持多种分频。AHB、APB1 和 APB2 域的最大频率是 192MHz，APB3 域的最大允许频率是 90MHz。

4.1.1 时钟源

- HEXT 振荡器时钟

包括 HEXT 晶体/陶瓷谐振器和 HEXT 旁路时钟两个时钟源。

HEXT 晶体/陶瓷谐振器外接一颗频率范围为 4~25MHz HEXT 的晶体，可为系统提供高精度的时钟。HEXT 时钟直到时钟稳定后才会被释放出来。

HEXT 旁路时钟可以提供频率高达 25MHz 的外部时钟。外部时钟信号必须连到 HEXT_IN 引脚，HEXT_OUT 引脚可以释放给 GPIO 控制。

- HICK 振荡器时钟

HICK 振荡器时钟由芯片内的高速 RC 振荡器提供。HICK 时钟的内部频率为 48MHz，频率精度较差，但启动时间比 HEXT 晶体振荡器短，每颗芯片的 HICK 时钟频率在出厂前已经被校准到 1% (25° C)，工厂校准值被装载到时钟控制寄存器的 HICKCAL[7: 0]位。考虑不同的电压或环境温度对 HICK 的 RC 振荡器的影响，用户可以通过时钟控制寄存器里的 HICKTRIM[5: 0]位来调整 HICK 频率。

HICK 时钟直到稳定后才会被释放出来。

- PLL 时钟

PLL 可以选择 HICK 时钟或 HEXT 时钟作为输入时钟，PLL 的输入时钟在 PLL 内部经过预分频器分频后送给 VCO 倍频，VCO 输出频率经过后分频器分频后输出。其中预分频后时钟需保证在 2M~16MHz 之间，VCO 的工作频率需保证在 500MHz~1200MHz 之间。使用 PLL 前，一定要先配置 PLL 参数，否则，PLL 使能后，这些参数将无法改动。PLL 时钟直到稳定后才会被释放出来。

PLL 公式如下：

$$\text{PLL 输出时钟} = \text{PLL 输入时钟} \times \text{PLL 倍频系数} / (\text{PLL 预分频系数} \times \text{PLL 后分频系数})$$

$$500\text{MHz} \leq \text{PLL 输入时钟} \times \text{PLL 倍频系数} / \text{PLL 预分频系数} \leq 1200\text{MHz}$$

$$2\text{MHz} \leq \text{PLL 输入时钟} / \text{PLL 预分频系数} \leq 16\text{MHz}$$

例如：当 PLL 输入时钟为 25 MHz 时，可以配置 PLL 输出频率= $25 \times 192 / (5 \times 5) = 192\text{ MHz}$

● LEXT 振荡器时钟

LEXT 振荡器时钟包括 LEXT 晶体/陶瓷谐振器和 LEXT 旁路时钟两个时钟源。

LEXT 晶体/陶瓷谐振器

LEXT 晶体/陶瓷谐振器提供一个低功耗且精确的 32.768KHz 低速时钟源。LEXT 时钟直到稳定后，才会被释放出来。

● LEXT 旁路时钟

在 LEXT 旁路模式下，可以提供最高频率达 32.768kHz 的外部时钟源。外部时钟信号必须连到 LEXT_IN 引脚，LEXT_OUT 引脚可以释放给 GPIO 控制。

● LICK 振荡器时钟

LICK 振荡器时钟由芯片内的低速 RC 振荡器提供，作为一个频率在 30kHz 和 60kHz 之间的低功耗时钟源，它可以为看门狗和自动唤醒单元提供时钟，并能在深度睡眠和待机模式下保持运行。

LICK 时钟直到稳定后，才会被释放出来。

4.1.2 系统时钟

系统复位以后，系统时钟使用 HICK 时钟作为默认时钟。系统时钟可在 HICK 振荡器时钟、HEXT 振荡器时钟和 PLL 时钟之间进行灵活切换，只有当目标时钟源稳定后，系统时钟切换才会发生。当 HICK 振荡器时钟直接作为系统时钟或间接通过 PLL 作为系统时钟时，它将无法被停止。

4.1.3 外设时钟

大多数外设使用系统时钟 HCLK、PCLK1 或 PCLK2 时钟。个别外设还有专用时钟。

系统滴答定时器（SysTick）使用 CPU FCLK（HCLK）或 CPU Systick（HCLK 的 8 分频）作为时钟。

ADC 使用 HCLK 时钟的 2、3、4、5…17 分频作为时钟。

定时器使用 APB1/2 作为时钟，特别地，当 APB 预分频系数是 1 时，定时器的时钟频率等于 APB1/2 的时钟频率；当 APB 预分频系数不为 1 时，定时器的时钟频率等于 APB1/2 时钟频率的 2 倍。

USB 时钟可在 HICK 和 PLLU 时钟之间切换。当选 HICK 时钟源时，需配置 USB 时钟为 48MHz 时钟；当选 PLLU 时钟时，PLLU 需设置为 48MHz。

ERTC 的时钟源有：HEXT 振荡器分频时钟，LEXT 振荡器时钟和 LICK 振荡器时钟。ERTC 的时钟源一旦选择后就不可再更改，只有将电池供电域复位后才能重新配置 ERTC 时钟源。当 VDD 掉电时，ERTC 使用 LEXT 作为时钟的话，ERTC 可以继续工作，但 ERTC 使用 HEXT 或 LICK 作为时钟源时，由于 HEXT 和 LICK 均掉电，会导致 ERTC 状态不定。

看门狗使用 LICK 振荡器时钟作为时钟源。硬件选项或软件开启看门狗后，将强制打开 LICK 振荡器，LICK 振荡器稳定后，才给看门狗提供时钟。

4.1.4 时钟失效检测

当 HEXT 时钟直接或间接作为系统时钟时，为防止 HEXT 时钟出现故障，特设计了时钟失效检测模块（CFD）。当 HEXT 时钟出现故障，CFD 侦测到失效后，将时钟失效事件送到 TMR1/9/10/11/12/13/14 的刹车输入端，并产生 CFD 中断，此 CFD 中断直接连到 CPU 的 NMI 中断，供软件完成营救操作。NMI 中断将一直重复执行，直到 CFD 中断挂起位被清除为止，所以在 NMI 的处理程序中必须清除 CFD 中断。当 HEXT 时钟出现故障时，将导致系统时钟切换到 HICK 时钟，同时关闭 CFD，关闭 HEXT 时钟，如果 HEXT 时钟通过 PLL 做为系统时钟时，也会关闭 PLL 模块。

4.1.5 自动滑顺频率切换

当系统时钟源从其他时钟切换到 PLL 或是 AHB 分频因子由大切换到小时，为了使系统稳定顺滑切换，特设计了自动顺滑频率切换功能，当系统频率操作目标大于 108MHz 时，建议开启自动顺滑频率切换功能。

当自动顺滑频率切换功能开启时，硬件会暂停 AHB 总线，直到整个自动顺滑频率切换才恢复。此期间 DMA 仍正常工作，中断事件会被记忆并待 AHB 总线恢复后 NVIC 即可处理。

4.1.6 内部时钟输出

微控制器允许输出时钟信号到外部 CLKOUT 引脚。ADCCLK、USB48M、SCLK、LICK、LEXT、HICK、HEXT、PLLCLK 等时钟信号可作为 CLKOUT 时钟。作为 CLKOUT 钟输出脚时，相应的 GPIO 端口寄存器必须被配置为相应功能。

4.1.7 中断

微控制器为每个时钟源设计了一个稳定标志，当用户开启一个时钟源后，可查询对应的时钟源的稳定标志来判断时钟是否稳定。当用户开启对应时钟源的中断使能的话，将产生中断请求。

当 HEXT 时钟出现故障，CFD 侦测到失效后，将产生 CFD 中断，此中断直接连到 CPU 的 NMI 中断。

4.2 复位

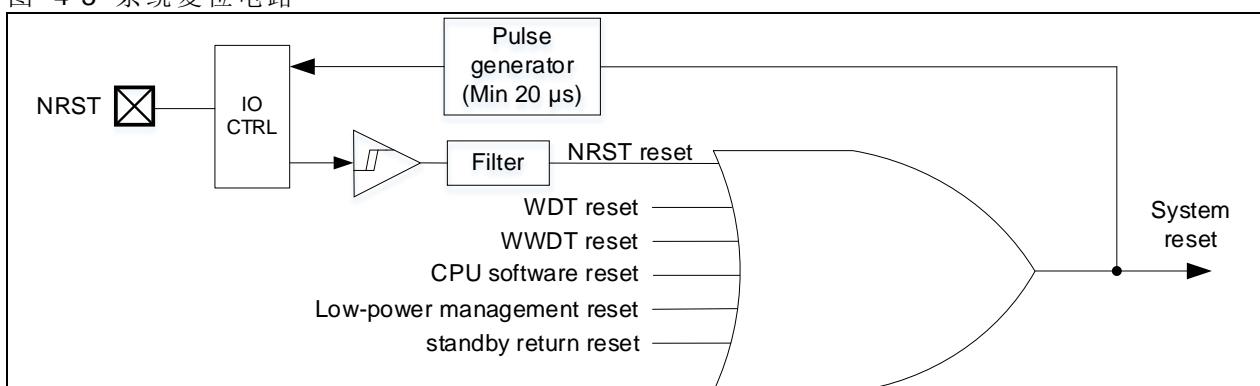
4.2.1 系统复位

AT32F455/456/457 系统复位包括以下复位源：

- NRST 复位：外部 NRST 管脚复位
- WDT 复位：看门狗溢出复位
- WWDT 复位：窗口看门狗溢出复位
- CPU 软件复位：Cortex®-M4F 软件复位
- 低功耗管理复位：将用户系统数据中的 nSTDBY_RST 位清 0 并进入待机模式，将产生低功耗管理复位；将用户系统数据中的 nDEPSLP_RST 位清 0 并进入深度睡眠模式，也将产生低功耗管理复位。
- POR 复位：上电复位
- LVR 复位：低电压复位
- 从待机模式中返回等事件产生复位。

NRST 复位，WDT 复位，WWDT 复位，软件复位和低功耗管理复位将复位所有寄存器至它们的复位状态，时钟控制器的控制/状态寄存器（CRM_CTRLSTS）和电池供电域中的寄存器除外；上电复位、低电压复位或者从待机模式中返回等事件产生复位会复位所有寄存器至复位状态，电池供电寄存器除外。

图 4-3 系统复位电路



4.2.2 电池供电域复位

电池供电域复位包括以下复位源：

- 电池供电域软件复位：设置电池供电域控制寄存器（CRM_BPDC）中的 BPDRST 位来产生复位
- 在 VDD 和 VBAT 两者掉电的前提下，VDD 或 VBAT 再上电将产生复位。

电池供电域软件复位只影响电池供电域。

4.3 CRM寄存器描述

下表列出了 CRM 寄存器的映像和复位值。

可以用字节（8 位）、半字（16 位）或字（32 位）的方式操作这些外设寄存器。

表 4-1 CRM寄存器的映像和复位值

寄存器简称	基址偏移量	复位值
CRM_CTRL	0x000	0x0000 XX83
CRM_PLLCFG	0x004	0x0000 07C1
CRM_CFG	0x008	0x4000 0000
CRM_CLKINT	0x00C	0x0000 0000
CRM_AHBRST1	0x010	0x0000 0000
CRM_AHBRST2	0x014	0x0000 0000
CRM_AHBRST3	0x018	0x0000 0000
CRM_APB1RST	0x020	0x0000 0000
CRM_APB2RST	0x024	0x0000 0000
CRM_AHBEN1	0x030	0x0000 0000
CRM_AHBEN2	0x034	0x0000 0000
CRM_AHBEN3	0x038	0x0000 0000
CRM_APB1EN	0x040	0x0000 0000
CRM_APB2EN	0x044	0x0000 0000
CRM_AHBLPEN1	0x050	0x1F41 90FF
CRM_AHBLPEN2	0x054	0x0000 80D0
CRM_AHBLPEN3	0x058	0x0000 0003
CRM_APB1LPEN	0x060	0xFEFE C9FF
CRM_APB2LPEN	0x064	0xE017 7333
CRM_PICLKS	0x068	0x0000 0000
CRM_BPDC	0x070	0x0000 0018
CRM_CTRLSTS	0x074	0x0C00 0000
CRM_MISC1	0x0A0	0x000F 0000
CRM_MISC2	0x0A4	0x4380 500D

4.3.1 时钟控制寄存器 (CRM_CTRL)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 27	保留	0x00	resd	请保持默认值。
位 26	PLLUSTBL	0x0	ro	PLLU 时钟稳定 (PLL clock stable) 该位待 PLLU 稳定后由硬件置起。 0: 未稳定； 1: 已稳定。
位 25	PLLSTBL	0x0	ro	PLL 时钟稳定 (PLL clock stable) 该位待 PLL 稳定后由硬件置起。 0: 未稳定； 1: 已稳定。
位 24	PLLEN	0x0	rw	PLL 使能 (PLL enable) 该位可由软件置起或清除，也可在进入待机或深度睡眠模式时，由硬件清除。当系统时钟为 PLL 时钟时，该位无法清除。 0: 关闭； 1: 开启。
位 23: 20	保留	0x0	resd	保持默认值。
位 19	CFDEN	0x0	rw	时钟失效检测使能 (Clock Failure Detection enable) 0: 关闭； 1: 开启。
位 18	HEXTBYPSS	0x0	rw	HEXT 旁路使能 (High speed external crystal bypass) 只有在 HEXT 关闭时，软件才能操作该位。 0: 关闭； 1: 开启。
位 17	HEXTSTBL	0x0	ro	HEXT 时钟稳定 (High speed external crystal stable) 该位待 HEXT 稳定后由硬件置起。 0: 未稳定； 1: 已稳定。
位 16	HEXTEN	0x0	rw	HEXT 使能 (High speed external crystal enable) 该位可由软件置起或清除，也可在进入待机或深度睡眠模式时，由硬件清除。当系统时钟有用到 HEXT 时，该位无法清除。 0: 关闭； 1: 开启。
位 15: 8	HICKCAL	0xFF	rw	HICK 时钟校准值 (High speed internal clock calibration) 默认值为出厂校准初始值。 HICK 输出频率为 48 MHZ 时，每 HICKCAL 数值的变化对应频率调整 240 kHz (设计值)；HICK 输出频率是 8 MHZ 时，每 HICKCAL 数值的变化对应频率调整 40 kHz (设计值)。 注意：此位只有在 HICKCAL_KEY[7: 0]为 0x5A 的时候可被写入。
位 7: 2	HICKTRIM	0x20	rw	HICK 时钟调整值 (High speed internal clock trimming) 该数值和 HICKCAL[7: 0]数值共同决定 HICK 振荡器的频率，默认数值为 32，可以把 HICK 调整到精度±1%。
位 1	HICKSTBL	0x1	ro	HICK 时钟稳定 (High speed internal clock stable) 该位待 HICK 稳定后由硬件置起。 0: 未稳定； 1: 已稳定。

位 0	HICKEN	0x1	rw	HICK 使能 (High speed internal clock enable) 该位可由软件置起或清除，在退出待机或深度睡眠模式，或 HEXT 发生故障时，该位也可被硬件置起。当系统时钟有用到 HICK 时，该位无法清除。 0: 关闭； 1: 开启。
-----	--------	-----	----	---

4.3.1 PLL时钟配置寄存器 (CRM_PLLCFG)

访问：无等待周期，字，半字和字节访问。

域	简称	复位值	类型	功能
位 31	保留	0x0	resd	请保持为复位值。
位 30	PLLRCS	0x0	rw	PLL 输入时钟选择 (PLL reference clock select) 由软件置'1'或清'0'来选择 PLL 输入时钟源。只能在关闭 PLLEN 时才能写入此位。 0: HICK 时钟作为 PLL 输入时钟； 1: HEXT 时钟作为 PLL 输入时钟源。
位 29	PLL_U_EN	0x0	rw	PLL_U 时钟输出使能 (PLL_U enable) 该位可由软件置起或清除，也可在进入待机或深度睡眠模式时，由硬件清除。 0: 关闭； 1: 开启。
位 28: 23 保留		0x0	resd	请保持为复位值。
位 22: 20	PLL_FU	0x0	rw	PLL_U 后分频配置值 (PLL_U post-division) PLL_FU 范围 (0~7) 当 PLLEN 使能时，该位无法配置。 000: PLL_U 后分频系数为 11, 11 分频输出； 001: PLL_U 后分频系数为 13, 13 分频输出； 010: PLL_U 后分频系数为 12, 12 分频输出； 011: PLL_U 后分频系数为 14, 14 分频输出； 100: PLL_U 后分频系数为 16, 16 分频输出； 101: PLL_U 后分频系数为 18, 18 分频输出； 110: PLL_U 后分频系数为 20, 20 分频输出； 111: PLL_U 后分频系数为 11, 11 分频输出； 请注意 PLL_FR 值和后分频系数对应关系
位 19: 16	PLL_FP	0x0	rw	PLL_P 后分频配置值 (PLL_P post-division) PLL_FP 范围 (0~15) 当 PLLEN 使能时，该位无法配置。 0000: PLL_P 后分频系数为 1, 1 分频输出； 0001: PLL_P 后分频系数为 2, 2 分频输出； 0010: PLL_P 后分频系数为 4, 4 分频输出； 0011: PLL_P 后分频系数为 6, 6 分频输出； 0100: PLL_P 后分频系数为 8, 8 分频输出； 0101: PLL_P 后分频系数为 10, 10 分频输出； 0110: PLL_P 后分频系数为 12, 12 分频输出； 0111: PLL_P 后分频系数为 14, 14 分频输出； 1000: PLL_P 后分频系数为 16, 16 分频输出； 1001: PLL_P 后分频系数为 18, 18 分频输出； 1010: PLL_P 后分频系数为 20, 20 分频输出； 1011: PLL_P 后分频系数为 22, 22 分频输出； 1100: PLL_P 后分频系数为 24, 24 分频输出； 1101: PLL_P 后分频系数为 26, 16 分频输出； 1110: PLL_P 后分频系数为 28, 28 分频输出； 1111: PLL_P 后分频系数为 30, 30 分频输出； 请注意 PLL_FR 值和后分频系数对应关系

位 15	保留	0x0	resd	请保持为复位值。
位 14: 6 PLL_NS	0x01F	rw		PLL 倍频系数(PLL Multiplication Factor) PLL_NS 适用范围 (31~500) 当 PLLEN 使能时, 该位无法配置。 00000000 ~ 000011110: 禁止使用 00011111: 31 000100000: 32 000100001: 33 111110011: 499 111110100: 500 111110101~111111111: 禁止使用
位 5: 4 保留	0x0	resd		请保持为复位值。
位 3: 0 PLL_MS	0x1	rw		PLL 预分频系数(PLL pre-division) PLL_MS 适用范围 (1~15) 当 PLLEN 使能时, 该位无法配置。 0000: 禁止使用 0001: 1 0010: 2 0011: 3 1110: 14 1111: 15

注意: PLL 时钟计算公式:

PLL_P 输出时钟 = PLL 输入时钟 \times PLL 倍频系数 / (PLL 预分频系数 \times PLL_FP 后分频系数)

PLL_U 输出时钟 = PLL 输入时钟 \times PLL 倍频系数 / (PLL 预分频系数 \times PLL_FU 后分频系数)

$500MHz \leq PLL$ 输入时钟 \times PLL 倍频系数 / PLL 预分频系数 $\leq 1000MHz$

$2MHz \leq PLL$ 输入时钟 / PLL 预分频系数 $\leq 16MHz$

4.3.2 时钟配置寄存器 (CRM_CFG)

访问: 0 到 2 个等待周期, 字, 半字和字节访问, 只有当访问发生在时钟切换时, 才会插入 1 或 2 个等待周期。

域	简称	复位值	类型	功能
位 31: 30 CLKOUT_SEL1	0x1	rw		内部输出时钟 2 选择 1 (clock output selection 1) 由软件置'1'或清零。 00: 系统时钟 (SCLK) 输出; 01: 二级时钟输出, 由 CRM_MISC1 寄存器的 CLKOUT_SEL2 位选择二级时钟输出 10: 外部振荡器时钟 (HEXT) 输出; 11: PLL 时钟输出; 注意: - 该时钟输出在启动和切换 CLKOUT 时钟源时可能会被截断。 - 在系统时钟作为输出至 CLKOUT 引脚时, 请保证输出时钟频率不超过 50MHz (I/O 口最高频率)。
位 29: 27 CLKOUTDIV1	0x0	rw		CLKOUT 分频因子 1 (Clock output division1) 0xx: CLKOUT 100: CLKOUT/2 101: CLKOUT/3 110: CLKOUT/4 111: CLKOUT/5
位 26:24 保留	0x0	resd		请保持为复位值。
位 23:22 I2SF5CLKSEL	0x0	resd		I2SF5 时钟来源选择寄存器 00: System Clock 01: PLL 10: HICK

位 21	保留	0x0	resd	11: 外部输入时钟 请保持为复位值。
位 20: 16	ERTCDIV	0x00	rw	ERTC 时钟源 HEXT 分频因子 (HEXT division for ERTC clock) 由软件置'1'或清'0'来控制 ERTC 时钟源 HEXT 的预分频系数。 此控制位必须在 ERTC 时钟源之前设定。 00000: 禁止使用 00001: 禁止使用 00010: HEXT/2 00011: HEXT/3 00100: HEXT/4 ... 11110: HEXT/30 11111: HEXT/31
位 15: 13	APB2DIV	0x0	rw	APB2 分频因子 (APB2 division) HCLK 分频后作为 APB2 时钟。 0xx: 不分频 100: 2 分频 101: 4 分频 110: 8 分频 111: 16 分频
位 12: 10	APB1DIV	0x0	rw	APB1 分频因子 (APB1 division) HCLK 分频后作为 APB1 时钟。 0xx: 不分频 100: 2 分频 101: 4 分频 110: 8 分频 111: 16 分频
位 9: 8	保留	0x0	resd	请保持为复位值。
位 7: 4	AHBDIV	0x0	rw	AHB 分频因子 (AHB division) 0xxx: SCLK 不分频 1000: SCLK 2 分频 1100: SCLK 64 分频 1001: SCLK 4 分频 1101: SCLK 128 分频 1010: SCLK 8 分频 1110: SCLK 256 分频 1011: SCLK 16 分频 1111: SCLK 512 分频
位 3: 2	SCLKSTS	0x0	ro	系统时钟选择状态位 (System clock select status) 00: HICK; 01: HEXT; 10: PLL/2; 11: 保留, 保持默认值。
位 1: 0	SCLKSEL	0x0	rw	系统时钟选择 (System clock select) 00: HICK; 01: HEXT; 10: PLL/2; 11: 保留, 保持默认值。

4.3.3 时钟中断寄存器 (CRM_CLKINT)

访问: 无等待周期, 字, 半字和字节访问

域	简称	复位值	类型	功能
位 31: 24	保留	0x00	resd	请保持为复位值。
位 23	CFDFC	0x0	wo	清除时钟失效标志 (Clock failure detection interrupt clear) 由软件置'1'来清除 CFDF。 0: 无作用; 1: 清除 CFDF 位。
位 22: 21	保留	0x0	resd	请保持为复位值。
位 20	PLLSTBLFC	0x0	wo	清除 PLL 稳定标志 (PLL stable flag clear) 由软件写'1'清除 PLLSTBLF。 0: 不清除;

				1: 清除。 清除 HEXT 稳定标志 (HEXT stable flag clear) 由软件写'1'清除 HEXTSTBLF。 0: 不清除; 1: 清除。
位 19	HEXTSTBLFC	0x0	wo	清除 HICK 稳定标志 (HICK stable flag clear) 由软件写'1'清除 HICKSTBLF。 0: 不清除; 1: 清除。
位 18	HICKSTBLFC	0x0	wo	清除 LEXT 稳定标志 (LEXT stable flag clear) 由软件写'1'清除 LEXTSTBLF。 0: 不清除; 1: 清除。
位 17	LEXTSTBLFC	0x0	wo	清除 LICK 稳定标志 (LICK stable flag clear) 由软件写'1'清除 LICKSTBLF。 0: 不清除; 1: 清除。
位 16	LICKSTBLFC	0x0	wo	清除 PLL 稳定中断使能 (PLL stable interrupt enable) 由软件写'1'清除 PLLSTBLIEN。 0: 不清除; 1: 清除。
位 15: 13	保留	0x0	resd	请保持为复位值。
位 12	PLLSTBLIEN	0x0	rw	HEXT 稳定中断使能 (HEXT stable interrupt enable) 0: 关闭; 1: 开启。
位 11	HEXTSTBLIEN	0x0	rw	HICK 稳定中断使能 (HICK stable interrupt enable) 0: 关闭; 1: 开启。
位 10	HICKSTBLIEN	0x0	rw	LEXT 稳定中断使能 (LEXT stable interrupt enable) 0: 关闭; 1: 开启。
位 9	LEXTSTBLIEN	0x0	rw	LICK 稳定中断使能 (LICK stable interrupt enable) 0: 关闭; 1: 开启。
位 7	CFDF	0x0	ro	时钟失效标志 (Clock Failure Detection flag) 在 HEXT 时钟出现故障时, 由硬件置起。 0: 未出现; 1: 出现。
位 6: 5	保留	0x0	resd	请保持为复位值。
位 4	PLLSTBLF	0x0	ro	PLL 稳定标志 (PLL stable flag) 由硬件置起。 0: 未稳定; 1: 已稳定。
位 3	HEXTSTBLF	0x0	ro	HEXT 稳定标志 (HEXT stable flag) 由硬件置起。 0: 未稳定; 1: 已稳定。
位 2	HICKSTBLF	0x0	ro	HICK 稳定标志 (HICK stable flag) 由硬件置起。 0: 未稳定; 1: 已稳定。
位 1	LEXTSTBLF	0x0	ro	LEXT 稳定标志 (LEXT stable flag) 由硬件置起。 0: 未稳定; 1: 已稳定。
位 0	LICKSTBLF	0x0	ro	LICK 稳定中断标志 (LICK stable flag) 由硬件置起。 0: 未稳定; 1: 已稳定。

4.3.4 AHB外设复位寄存器1 (CRM_AHBRST1)

访问：无等待周期，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 26	保留	0x0	resd	请保持为复位值。
位 25	EMACRST	0x0	rw	EMCA 复位 (EMAC reset) 0: 无复位； 1: 复位。
位 24	DMA2RST	0x0	rw	DMA2 复位 (DMA2 reset) 0: 无复位； 1: 复位。
位 23	保留	0x0	resd	请保持为复位值。
位 22	DMA1RST	0x0	rw	DMA1 复位 (DMA1 reset) 0: 无复位； 1: 复位。
位 21: 13	保留	0x000	resd	请保持为复位值。
位 12	CRCRST	0x0	rw	CRC 复位 (CRC reset) 0: 无复位； 1: 复位。
位 11: 8	保留	0x00	resd	请保持为复位值。
位 7	GPIOHRST	0x0	rw	IO 端口 H 复位 (IO port H reset) 0: 无复位； 1: 复位。
位 6	GPIOGRST	0x0	rw	IO 端口 G 复位 (IO port G reset) 0: 无复位； 1: 复位。
位 5	GPIOFRST	0x0	rw	IO 端口 F 复位 (IO port F reset) 0: 无复位； 1: 复位。
位 4	GPIOERT	0x0	rw	IO 端口 E 复位 (IO port E reset) 0: 无复位； 1: 复位。
位 3	GPIODRST	0x0	rw	IO 端口 D 复位 (IO port D reset) 0: 无复位； 1: 复位。
位 2	GPIOCRST	0x0	rw	IO 端口 C 复位 (IO port C reset) 0: 无复位； 1: 复位。
位 1	GPIOBRST	0x0	rw	IO 端口 B 复位 (IO port B reset) 0: 无复位； 1: 复位。
位 0	GPIOARST	0x0	rw	IO 端口 A 复位 (IO port A reset) 0: 无复位； 1: 复位。

4.3.5 AHB外设复位寄存器2 (CRM_AHBRST2)

访问：无等待周期，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 16	保留	0x000000	resd	请保持为复位值。
位 15	SDIO1RST	0x0	rw	SDIO 复位 (SDIO reset) 0: 无复位； 1: 复位。
位 14: 8	保留	0x00	resd	请保持为复位值。
位 7	OTGFS1RST	0x0	rw	OTGFS1 复位 (OTGFS1 reset) 0: 无复位； 1: 复位。
位 6	TRNGRST	0x0	rw	TRNG 复位 (TRNG reset) 0: 无复位； 1: 复位。
位 5	保留	0x00	resd	请保持为复位值。

位 4	AESRST	0x0	rw	AES 复位 (AES reset) 0: 无复位; 1: 复位。
位 3: 0	保留	0x00	resd	请保持为复位值。

4.3.6 AHB外设复位寄存器3 (CRM_AHBRST3)

访问: 无等待周期, 字, 半字和字节访问

域	简称	复位值	类型	功能
位 31: 2	保留	0x00000000	resd	请保持为复位值。
位 1	QSPI1RST	0x0	rw	QSPI 复位 (QSPI reset) 0: 无复位; 1: 复位。
位 0	XMCRST	0x0	rw	XMC 复位 (XMC reset) 0: 无复位; 1: 复位。

4.3.7 APB1外设复位寄存器 (CRM_APB1RST)

访问: 无等待周期, 字, 半字和字节访问

域	简称	复位值	类型	功能
位 31	USART8RST	0x0	rw	USART8 复位 (USART8 reset) 0: 无复位; 1: 复位。
位 30	USART7RST	0x0	rw	USART7 复位 (USART7 reset) 0: 无复位; 1: 复位。
位 29	DACRST	0x0	rw	DAC 复位 (DAC reset) 0: 无复位; 1: 复位。
位 28	PWCRST	0x0	rw	电源接口复位 (Power interface reset) 0: 无复位; 1: 复位。
位 27	CAN3RST	0x0	rw	CAN3 复位 (CAN3 reset) 0: 无复位; 1: 复位。
位 26	CAN2RST	0x0	rw	CAN2 复位 (CAN2 reset) 0: 无复位; 1: 复位。
位 25	CAN1RST	0x0	rw	CAN1 复位 (CAN1 reset) 0: 无复位; 1: 复位。
位 24	保留	0x0	resd	请保持为复位值。
位 23	I2C3RST	0x0	rw	I ² C3 复位 (I ² C3 reset) 0: 无复位; 1: 复位。
位 22	I2C2RST	0x0	rw	I ² C2 复位 (I ² C2 reset) 0: 无复位; 1: 复位。
位 21	I2C1RST	0x0	rw	I ² C1 复位 (I ² C1 reset) 0: 无复位; 1: 复位。
位 20	USART5RST	0x0	rw	USART5 复位 (USART5 reset) 0: 无复位; 1: 复位。
位 19	USART4RST	0x0	rw	USART4 复位 (USART4 reset) 0: 无复位; 1: 复位。
位 18	USART3RST	0x0	rw	USART3 复位 (USART3 reset) 由软件置'1'或清'0' 0: 无作用;

				1: 复位 USART3。 USART2 复位 (USART2 reset) 0: 无复位; 1: 复位。
位 17	USART2RST	0x0	rw	0: 无复位; 1: 复位。
位 16	保留	0x0	resd	请保持为复位值。 SPI3 复位 (SPI3 reset) 0: 无复位; 1: 复位。
位 15	SPI3RST	0x0	rw	0: 无复位; 1: 复位。
位 14	SPI2RST	0x0	rw	0: 无复位; 1: 复位。
位 13: 12	保留	0x0	resd	请保持为复位值。 窗口看门狗复位 (Window watchdog reset) 0: 无复位; 1: 复位。
位 11	WWDTRST	0x0	rw	0: 无复位; 1: 复位。
位 10: 9	保留	0x0	resd	请保持为复位值。 定时器 14 复位 (Timer14 reset) 0: 无复位; 1: 复位。
位 8	TMR14RST	0x0	rw	0: 无复位; 1: 复位。
位 7	TMR13RST	0x0	rw	0: 无复位; 1: 复位。
位 6	TMR12RST	0x0	rw	0: 无复位; 1: 复位。
位 5	TMR7RST	0x0	rw	0: 无复位; 1: 复位。
位 4	TMR6RST	0x0	rw	0: 无复位; 1: 复位。
位 3	TMR5RST	0x0	rw	0: 无复位; 1: 复位。
位 2	TMR4RST	0x0	rw	0: 无复位; 1: 复位。
位 1	TMR3RST	0x0	rw	0: 无复位; 1: 复位。
位 0	TMR2RST	0x0	rw	0: 无复位; 1: 复位。

4.3.8 APB2外设复位寄存器 (CRM_APB2RST)

访问：无等待周期，字，半字和字节访问

域	简称	复位值	类型	功能
位 31	I2S3EXTRST	0x0	rw	I2S3EXT APB 复位 (I2S3EXT APB reset) 0: 无复位; 1: 复位。
位 30	I2S2EXTRST	0x0	rw	I2S2EXT APB 复位 (I2S2EXT APB reset) 0: 无复位; 1: 复位。
位 29	ACCRST	0x0	rw	ACC 复位 (ACC reset) 0: 无复位; 1: 复位。
位 28: 21	保留	0x00	resd	请保持为复位值。
位 20	I2SF5RST	0x0	rw	I2SF5 复位 (I2SF5 reset) 0: 无复位; 1: 复位。

位 19	保留	0x0	resd	请保持为复位值。
位 18	TMR11RST	0x0	rw	定时器 11 复位 (Timer11 reset) 0: 无复位; 1: 复位。
位 17	TMR10RST	0x0	rw	定时器 10 复位 (Timer10 reset) 0: 无复位; 1: 复位。
位 16	TMR9RST	0x0	rw	定时器 9 复位 (Timer9 reset) 0: 无复位; 1: 复位。
位 15	保留	0x0	resd	请保持为复位值。
位 14	SCFGRST	0x0	rw	SCFG 复位 (SCFG reset) 0: 无复位; 1: 复位。
位 13	SPI4RST	0x0	rw	SPI4 复位 (SPI4 reset) 0: 无复位; 1: 复位。
位 12	SPI1RST	0x0	rw	SPI1 复位 (SPI1 reset) 0: 无复位; 1: 复位。
位 11: 9	保留	0x0	resd	请保持为复位值。
位 8	ADCRST	0x0	rw	ADC1 接口复位 (ADC interface reset) 0: 无复位; 1: 复位。
位 7: 6	保留	0x0	resd	请保持为复位值。
位 5	USART6RST	0x0	rw	USART6 复位 (USART6 reset) 0: 无复位; 1: 复位。
位 4	USART1RST	0x0	rw	USART1 复位 (USART1 reset) 0: 无复位; 1: 复位。
位 3: 2	保留	0x0	resd	请保持为复位值。
位 1	TMR8RST	0x0	rw	TMR8 定时器复位 (TMR8 timer reset) 0: 无复位; 1: 复位。
位 0	TMR1RST	0x0	rw	TMR1 定时器复位 (TMR1 timer reset) 0: 无复位; 1: 复位。

4.3.9 AHB外设时钟使能寄存器1 (CRM_AHBEN1)

访问：无等待周期，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 29	保留	0x0	resd	请保持为复位值。
位 28	EMACPTPEN	0x0	rw	以太网 MAC PTP 时钟使能 (EMAC PTP clock enable) 0: 关闭; 1: 启开。
位 27	EMACRXEN	0x0	rw	以太网 MAC 接收时钟使能 (EMAC RX clock enable) 0: 关闭; 1: 启开。 注：在 RMII 模式下，如果使能了这个时钟，MAC 的 RMII 时钟也被使能。
位 26	EMACTXEN	0x0	rw	以太网 MAC 发送时钟使能 (EMAC TX clock enable) 0: 关闭; 1: 启开。 注：在 RMII 模式下，如果使能了这个时钟，MAC 的 RMII 时钟也被使能。
位 25	EMACEN	0x0	rw	以太网 MAC 时钟使能 (EMAC TX clock enable) 0: 关闭; 1: 启开。
位 24	DMA2EN	0x0	rw	DMA2 时钟使能 (DMA2 clock enable)

				0: 关闭; 1: 开启。
位 23	保留	0x0	resd	请保持为复位值。
位 22	DMA1EN	0x0	rw	DMA1 时钟使能 (DMA1 clock enable) 0: 关闭; 1: 开启。
位 21: 13	保留	0x000	resd	请保持为复位值。
位 12	CRCEN	0x0	rw	CRC 时钟使能 (CRC clock enable) 0: 关闭; 1: 开启。
位 11: 8	保留	0x00	resd	请保持为复位值。
位 7	GPIOHEN	0x0	rw	IO 端口 H 时钟使能 (IO port H clock enable) 0: 关闭; 1: 开启。
位 6	GPIOGEN	0x0	rw	IO 端口 G 时钟使能 (IO port G clock enable) 0: 关闭; 1: 开启。
位 5	GPIOFEN	0x0	rw	IO 端口 F 时钟使能 (IO port F clock enable) 0: 关闭; 1: 开启。
位 4	GPIOEEN	0x0	rw	IO 端口 E 时钟使能 (IO port E clock enable) 0: 关闭; 1: 开启。
位 3	GPIODEN	0x0	rw	IO 端口 D 时钟使能 (IO port D clock enable) 0: 关闭; 1: 开启。
位 2	GPIOCEN	0x0	rw	IO 端口 C 时钟使能 (IO port C clock enable) 0: 关闭; 1: 开启。
位 1	GPIOBEN	0x0	rw	IO 端口 B 时钟使能 (IO port B clock enable) 0: 关闭; 1: 开启。
位 0	GPIOAEN	0x0	rw	IO 端口 A 时钟使能 (IO port A clock enable) 0: 关闭; 1: 开启。

4.3.10 AHB外设时钟使能寄存器2 (CRM_AHBEN2)

访问：无等待周期，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 16	保留	0x000000	resd	请保持为复位值。
位 15	SDIO1EN	0x0	rw	SDIO 时钟使能 (SDIO clock enable) 0: 关闭; 1: 开启。
位 14: 8	保留	0x000000	resd	请保持为复位值。
位 7	OTGFS1EN	0x0	rw	OTGFS1 时钟使能 (OTGFS1 clock enable) 0: 关闭; 1: 开启。
位 6	TRNGEN	0x0	rw	TRNG 时钟使能 (TRNG clock enable) 0: 关闭; 1: 开启。
位 5	保留	0x0	resd	请保持为复位值。
位 4	AESEN	0x0	rw	AES 时钟使能 (AES clock enable) 0: 关闭; 1: 开启。
位 3: 0	保留	0x00	resd	请保持为复位值。

4.3.11 AHB外设时钟使能寄存器3 (CRM_AHBEN3)

访问：无等待周期，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 2	保留	0x00000000	resd	请保持为复位值。
位 1	QSPI1EN	0x0	rw	QSPI 时钟使能 (QSPI clock enable) 0: 关闭; 1: 开启。
位 0	XMCEN	0x0	rw	XMC 时钟使能 (XMC clock enable) 0: 关闭; 1: 开启。

4.3.12 APB1外设时钟使能寄存器 (CRM_APB1EN)

访问: 无等待周期, 字, 半字和字节访问

域	简称	复位值	类型	功能
位 31	USART8EN	0x0	rw	USART8 时钟使能 (USART8 clock enable) 0: 关闭; 1: 开启。
位 30	USART7EN	0x0	rw	USART7 时钟使能 (USART7 clock enable) 0: 关闭; 1: 开启。
位 29	DACEN	0x0	rw	DAC 时钟使能 (DAC clock enable) 0: 关闭; 1: 开启。
位 28	PWCEN	0x0	rw	电源接口时钟使能 (Power interface clock enable) 0: 关闭; 1: 开启。
位 27	CAN3EN	0x0	rw	CAN3 时钟使能 (CAN3 clock enable) 0: 关闭; 1: 开启。
位 26	CAN2EN	0x0	rw	CAN2 时钟使能 (CAN2 clock enable) 0: 关闭; 1: 开启。
位 25	CAN1EN	0x0	rw	CAN1 时钟使能 (CAN1 clock enable) 0: 关闭; 1: 开启。
位 24	保留	0x0	resd	请保持为复位值。
位 23	I2C3EN	0x0	rw	I ² C3 时钟使能 (I ² C3 clock enable) 0: 关闭; 1: 开启。
位 22	I2C2EN	0x0	rw	I ² C2 时钟使能 (I ² C2 clock enable) 0: 关闭; 1: 开启。
位 21	I2C1EN	0x0	rw	I ² C1 时钟使能 (I ² C1 clock enable) 0: 关闭; 1: 开启。
位 20	USART5EN	0x0	rw	USART5 时钟使能 (USART5 clock enable) 0: 关闭; 1: 开启。
位 19	USART4EN	0x0	rw	USART4 时钟使能 (USART4 clock enable) 0: 关闭; 1: 开启。
位 18	USART3EN	0x0	rw	USART3 时钟使能 (USART3 clock enable) 0: 关闭; 1: 开启。
位 17	USART2EN	0x0	rw	USART2 时钟使能 (USART2 clock enable) 0: 关闭; 1: 开启。
位 16	保留	0x0	resd	请保持为复位值。
位 15	SPI3EN	0x0	rw	SPI3 时钟使能 (SPI3 clock enable) 0: 关闭; 1: 开启。

位 14	SPI2EN	0x0	rw	SPI2 时钟使能 (SPI2 clock enable) 0: 关闭; 1: 开启。
位 13: 12	保留	0x0	resd	请保持为复位值。
位 11	WWDTEN	0x0	rw	窗口看门狗时钟使能 (Window watchdog clock enable) 0: 关闭; 1: 开启。
位 10: 9	保留	0x0	resd	请保持为复位值。
位 8	TMR14EN	0x0	rw	定时器 14 时钟使能 (Timer14 clock enable) 0: 关闭; 1: 开启。
位 7	TMR13EN	0x0	rw	定时器 13 时钟使能 (Timer13 clock enable) 0: 关闭; 1: 开启。
位 6	TMR12EN	0x0	rw	定时器 12 时钟使能 (Timer12 clock enable) 0: 关闭; 1: 开启。
位 5	TMR7EN	0x0	rw	定时器 7 时钟使能 (Timer 7 clock enable) 0: 关闭; 1: 开启。
位 4	TMR6EN	0x0	rw	定时器 6 时钟使能 (Timer 6 clock enable) 0: 关闭; 1: 开启。
位 3	TMR5EN	0x0	rw	定时器 5 时钟使能 (Timer 5 clock enable) 0: 关闭; 1: 开启。
位 2	TMR4EN	0x0	rw	定时器 4 时钟使能 (Timer 4 clock enable) 0: 关闭; 1: 开启。
位 1	TMR3EN	0x0	rw	定时器 3 时钟使能 (Timer 3 clock enable) 0: 关闭; 1: 开启。
位 0	TMR2EN	0x0	rw	定时器 2 时钟使能 (Timer 2 clock enable) 0: 关闭; 1: 开启。

4.3.13 APB2外设时钟使能寄存器 (CRM_APB2EN)

访问：无等待周期，字，半字和字节访问

域	简称	复位值	类型	功能
位 31	I2S3EXTEN	0x0	rw	I2S3EXT APB 时钟使能 (I2S3EXT APB clock enable) 0: 关闭; 1: 开启。
位 30	I2S2EXTEN	0x0	rw	I2S2EXT APB 时钟使能 (I2S2EXT APB clock enable) 0: 关闭; 1: 开启。
位 29	ACCEN	0x0	rw	ACC 时钟使能 (ACC clock enable) 0: 关闭; 1: 开启。
位 28: 21	保留	0x00	resd	请保持为复位值。
位 20	I2SF5EN	0x0	rw	I2SF5 时钟使能 (I2SF5 clock enable) 0: 关闭; 1: 开启。
位 19	保留	0x0	resd	请保持为复位值。
位 18	TMR11EN	0x0	rw	定时器 11 时钟使能 (Timer11 clock enable) 0: 关闭; 1: 开启。
位 17	TMR10EN	0x0	rw	定时器 10 时钟使能 (Timer10 clock enable) 0: 关闭; 1: 开启。

位 16	TMR9EN	0x0	rw	定时器 9 时钟使能 (Timer9 clock enable) 0: 关闭; 1: 开启。
位 15	保留	0x0	resd	请保持为复位值。
位 14	SCFGGEN	0x0	rw	SCFG 时钟使能 (SCFG clock enable) 0: 关闭; 1: 开启。
位 13	SPI4EN	0x0	rw	SPI4 时钟使能 (SPI4 clock enable) 0: 关闭; 1: 开启。
位 12	SPI1EN	0x0	rw	SPI1 时钟使能 (SPI1 clock enable) 0: 关闭; 1: 开启。
位 11: 10	保留	0x0	resd	请保持为复位值。
位 9	ADC2EN	0x0	rw	ADC2 接口时钟使能 (ADC2 interface clock enable) 0: 关闭; 1: 开启。
位 8	ADC1EN	0x0	rw	ADC1 接口时钟使能 (ADC1 interface clock enable) 0: 关闭; 1: 开启。
位 7: 6	保留	0x0	resd	请保持为复位值。
位 5	USART6EN	0x0	rw	USART6 时钟使能 (USART6 clock enable) 0: 关闭; 1: 开启。
位 4	USART1EN	0x0	rw	USART1 时钟使能 (USART1 clock enable) 0: 关闭; 1: 开启。
位 3: 2	保留	0x0	resd	请保持为复位值。
位 1	TMR8EN	0x0	rw	TMR8 定时器时钟使能 (TMR1 timer clock enable) 0: 关闭; 1: 开启。
位 0	TMR1EN	0x0	rw	TMR1 定时器时钟使能 (TMR1 timer clock enable) 0: 关闭; 1: 开启。

4.3.14 AHB外设时钟低功耗使能寄存器1 (CRM_AHBLPEN1)

访问：无等待周期，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 29	保留	0x0	resd	请保持为复位值。
位 28	EMACPTPLPEN	0x1	rw	睡眠模式下以太网 MAC PTP 时钟使能 (EMAC PTP clock enable during sleep mode) 0: 关闭; 1: 开启。
位 27	EMACRXLPEN	0x1	rw	睡眠模式下以太网 MAC 接收时钟使能 (EMAC RX clock enable during sleep mode) 0: 关闭; 1: 开启。 注：在 RMII 模式下，如果使能了这个时钟，MAC 的 RMII 时钟也被使能。
位 26	EMACTXLPEN	0x1	rw	睡眠模式下以太网 MAC 发送时钟使能 (EMAC TX clock enable during sleep mode) 0: 关闭; 1: 开启。 注：在 RMII 模式下，如果使能了这个时钟，MAC 的 RMII 时钟也被使能。
位 25	EMACLPPEN	0x1	rw	睡眠模式下以太网 MAC 时钟使能 (EMAC TX clock enable during sleep mode) 0: 关闭; 1: 开启。

位 24	DMA2LPEN	0x1	rw	睡眠模式下 DMA2 时钟使能 (DMA2 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 23	保留	0x0	resd	请保持为复位值。
位 22	DMA1LPEN	0x1	rw	睡眠模式下 DMA1 时钟使能 (DMA1 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 21: 17	保留	0x00	resd	请保持为复位值。
位 16	SRAMLPEN	0x1	rw	睡眠模式下 SRAM 时钟使能 (SRAM clock enable during sleep mode) 0: 关闭; 1: 开启。
位 15	FLASHLPEN	0x1	rw	睡眠模式下 FLASH 时钟使能 (FLASH clock enable during sleep mode) 0: 关闭; 1: 开启。
位 14: 13	保留	0x0	resd	请保持为复位值。
位 12	CRCLPEN	0x1	rw	睡眠模式下 CRC 时钟使能 (CRC clock enable during sleep mode) 0: 关闭; 1: 开启。
位 11: 8	保留	0x0	resd	请保持为复位值。
位 7	GPIOHLPEN	0x1	rw	睡眠模式下 IO 端口 H 时钟使能 (IO port H clock enable during sleep mode) 0: 关闭; 1: 开启。
位 6	GPIOGLPEN	0x1	rw	睡眠模式下 IO 端口 G 时钟使能 (IO port G clock enable during sleep mode) 0: 关闭; 1: 开启。
位 5	GPIOFLPEN	0x1	rw	睡眠模式下 IO 端口 F 时钟使能 (IO port F clock enable during sleep mode) 0: 关闭; 1: 开启。
位 4	GPIOELPEN	0x1	rw	睡眠模式下 IO 端口 E 时钟使能 (IO port E clock enable during sleep mode) 0: 关闭; 1: 开启。
位 3	GPIOELPEN	0x1	rw	睡眠模式下 IO 端口 D 时钟使能 (IO port D clock enable during sleep mode) 0: 关闭; 1: 开启。
位 2	GPIOCLPEN	0x1	rw	睡眠模式下 IO 端口 C 时钟使能 (IO port C clock enable during sleep mode) 0: 关闭; 1: 开启。
位 1	GPIOBLPEN	0x1	rw	睡眠模式下 IO 端口 B 时钟使能 (IO port B clock enable during sleep mode) 0: 关闭; 1: 开启。
位 0	GPIOALPEN	0x1	rw	睡眠模式下 IO 端口 A 时钟使能 (IO port A clock enable during sleep mode) 0: 关闭; 1: 开启。

4.3.15 AHB外设时钟低功耗使能寄存器2 (CRM_AHBLPEN2)

访问: 无等待周期, 字, 半字和字节访问

域	简称	复位值	类型	功能
---	----	-----	----	----

位 31: 16	保留	0x000000	resd	请保持为复位值。
位 15	SDIO1LPEN	0x1	rw	睡眠模式下 SDIO 时钟使能 (SDIO clock enable during sleep mode) 0: 关闭; 1: 开启。
位 14: 8	保留	0x00	resd	请保持为复位值。
位 7	OTGFS1LPEN	0x1	rw	睡眠模式下 OTGFS1 时钟使能 (OTGFS1 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 6	TRNGLPEN	0x1	rw	睡眠模式下 TRNG 时钟使能 (TRNG clock enable during sleep mode) 0: 关闭; 1: 开启。
位 5	保留	0x0	resd	请保持为复位值。
位 4	AESLPEN	0x1	rw	睡眠模式下 AES 时钟使能 (AES clock enable during sleep mode) 0: 关闭; 1: 开启。
位 3: 0	保留	0x00	resd	请保持为复位值。

4.3.16 AHB外设时钟低功耗使能寄存器3 (CRM_AHBLPEN3)

访问：无等待周期，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 2	保留	0x00000000	resd	请保持为复位值。
位 1	QSPI1LPEN	0x1	rw	睡眠模式下 QSPI 时钟使能 (QSPI clock enable during sleep mode) 0: 关闭; 1: 开启。
位 0	XMCLPEN	0x1	rw	睡眠模式下 XMC 时钟使能 (XMC clock enable during sleep mode) 0: 关闭; 1: 开启。

4.3.17 APB1外设时钟低功耗使能寄存器 (CRM_APB1LPEN)

访问：无等待周期，字，半字和字节访问

域	简称	复位值	类型	功能
位 31	USART8LPEN	0x1	rw	睡眠模式下 USART8 时钟使能 (USART8 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 30	USART7LPEN	0x1	rw	睡眠模式下 USART7 时钟使能 (USART7 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 29	DACLPE	0x1	rw	睡眠模式下 DAC 时钟使能 (DAC clock enable during sleep mode) 0: 关闭; 1: 开启。
位 28	PWCLPEN	0x1	rw	睡眠模式下 PWC 时钟使能 (Power interface clock enable during sleep mode) 0: 关闭; 1: 开启。
位 27	CAN3LPEN	0x1	rw	睡眠模式下 CAN3 时钟使能 (CAN3 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 26	CAN2LPEN	0x1	rw	睡眠模式下 CAN2 时钟使能 (CAN2 clock enable during sleep mode)

				0: 关闭; 1: 开启。
位 25	CAN1LPEN	0x1	rw	睡眠模式下 CAN1 时钟使能 (CAN1 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 24	保留	0x0	resd	请保持为复位值。
位 23	I2C3LPEN	0x1	rw	睡眠模式下 I ² C3 时钟使能 (I ² C3 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 22	I2C2LPEN	0x1	rw	睡眠模式下 I ² C2 时钟使能 (I ² C2 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 21	I2C1LPEN	0x1	rw	睡眠模式下 I ² C1 时钟使能 (I ² C1 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 20	USART5LPEN	0x1	rw	睡眠模式下 USART5 时钟使能 (USART5 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 19	USART4LPEN	0x1	rw	睡眠模式下 USART4 时钟使能 (USART4 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 18	USART3LPEN	0x1	rw	睡眠模式下 USART3 时钟使能 (USART3 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 17	USART2LPEN	0x1	rw	睡眠模式下 USART2 时钟使能 (USART2 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 16	保留	0x0	resd	请保持为复位值。
位 15	SPI3LPEN	0x1	rw	睡眠模式下 SPI3 时钟使能 (SPI3 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 14	SPI2LPEN	0x1	rw	睡眠模式下 SPI2 时钟使能 (SPI 2 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 13: 12	保留	0x0	resd	请保持为复位值。
位 11	WWDTLPEN	0x1	rw	睡眠模式下 WWDT 时钟使能 (Window watchdog clock enable during sleep mode) 0: 关闭; 1: 开启。
位 10: 9	保留	0x0	resd	请保持为复位值。
位 8	TMR14LPEN	0x1	rw	睡眠模式下 TMR14 时钟使能 (Timer14 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 7	TMR13LPEN	0x1	rw	睡眠模式下 TMR13 时钟使能 (Timer13 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 6	TMR12LPEN	0x1	rw	睡眠模式下 TMR12 时钟使能 (Timer12 clock enable during sleep mode) 0: 关闭;

				1: 开启。 睡眠模式下 TMR7 时钟使能 (Timer 7 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 5	TMR7LPEN	0x1	rw	睡眠模式下 TMR6 时钟使能 (Timer 6 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 4	TMR6LPEN	0x1	rw	睡眠模式下 TMR5 时钟使能 (Timer 5 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 3	TMR5LPEN	0x1	rw	睡眠模式下 TMR4 时钟使能 (Timer 4 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 2	TMR4LPEN	0x1	rw	睡眠模式下 TMR3 时钟使能 (Timer 3 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 1	TMR3LPEN	0x1	rw	睡眠模式下 TMR2 时钟使能 (Timer 2 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 0	TMR2LPEN	0x1	rw	睡眠模式下 I2S3EXT APB 时钟使能 (I2S3EXT clock enable during sleep mode) 0: 关闭; 1: 开启。

4.3.18 APB2外设时钟低功耗使能寄存器 (CRM_APB2LPEN)

访问：无等待周期，字，半字和字节访问

域	简称	复位值	类型	功能
位 31	I2S3EXTLPEN1	0x1	rw	睡眠模式下 I2S3EXT APB 时钟使能 (I2S3EXT clock enable during sleep mode) 0: 关闭; 1: 开启。
位 30	I2S2EXTLPEN1	0x1	rw	睡眠模式下 I2S2EXT APB 时钟使能 (I2S2EXT clock enable during sleep mode) 0: 关闭; 1: 开启。
位 29	ACCLPEN	0x1	rw	睡眠模式下 ACC 时钟使能 (ACC clock enable during sleep mode) 0: 关闭; 1: 开启。
位 28: 21	保留	0x00	resd	请保持为复位值。
位 20	I2SF5LPEN	0x1	rw	睡眠模式下 I2SF5 时钟使能 (I2SF5 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 19	保留	0x0	resd	请保持为复位值。
位 18	TMR11LPEN	0x1	rw	睡眠模式下定时器 11 时钟使能 (Timer11 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 17	TMR10LPEN	0x1	rw	睡眠模式下定时器 10 时钟使能 (Timer10 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 16	TMR9LPEN	0x1	rw	睡眠模式下定时器 9 时钟使能 (Timer9 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 15	保留	0x0	resd	请保持为复位值。

位 14	SCFGLPEN	0x1	rw	睡眠模式下 SCFG 时钟使能 (SCFG clock enable during sleep mode) 0: 关闭; 1: 开启。
位 13	SPI4LPEN	0x1	rw	睡眠模式下 SPI4 时钟使能 (SPI4 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 12	SPI1LPEN	0x1	rw	睡眠模式下 SPI1 时钟使能 (SPI1 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 11: 10	保留	0x0	resd	请保持为复位值。
位 9	ADC2LPEN	0x1	rw	睡眠模式下 ADC2 接口时钟使能 (ADC2 interface clock enable during sleep mode) 0: 关闭; 1: 开启。
位 8	ADC1LPEN	0x1	rw	睡眠模式下 ADC1 接口时钟使能 (ADC1 interface clock enable during sleep mode) 0: 关闭; 1: 开启。
位 7: 6	保留	0x0	resd	请保持为复位值。
位 5	USART6LPEN	0x1	rw	睡眠模式下 USART6 时钟使能 (USART6 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 4	USART1LPEN	0x1	rw	睡眠模式下 USART1 时钟使能 (USART1 clock enable during sleep mode) 0: 关闭; 1: 开启。
位 3: 2	保留	0x0	resd	请保持为复位值。
位 1	TMR8LPEN	0x1	rw	睡眠模式下 TMR8 定时器时钟使能 (TMR8 timer clock enable during sleep mode) 0: 关闭; 1: 开启。
位 0	TMR1LPEN	0x1	rw	睡眠模式下 TMR1 定时器时钟使能 (TMR1 timer clock enable during sleep mode) 0: 关闭; 1: 开启。

4.3.19 外设独立时钟选择寄存器 (CRM_PICLKS)

访问：无等待周期，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 30	保留	0x0	resd	请保持为复位值。
			rw	CAN3 主机时钟选择位 (CAN3 host clock select) 00: HEXT 01: PLLU 10: PCLK1 11: Reserved
位 29: 28	CAN3_CLKSEL	0x0		
位 27: 26	CAN2_CLKSEL	0x0	rw	CAN2 主机时钟选择位 (CAN2 host clock select) 00: HEXT 01: PLLU 10: PCLK1 11: Reserved
位 25: 24	CAN1_CLKSEL	0x0	rw	CAN1 主机时钟选择位 (CAN1 host clock select) 00: HEXT 01: PLLU 10: PCLK1 11: Reserved
位 23: 0	保留	0x00 0000	resd	请保持为复位值。

4.3.20 备份域控制寄存器 (CRM_BPDC)

只能由备份域复位有效复位

访问：0 到 3 等待周期，字、半字和字节访问；当连续对该寄存器进行访问时，将插入等待状态。

注意：电池供电域控制寄存器中 (CRM_BPDC) LEXTEN、LEXTBYP、ERTCSEL 和 ERTCEN 位处于电池供电域。因此，这些位在复位后处于写保护状态，只有在电源控制寄存器 (PWC_CTRL) 中的 BPWEN 位置位后才能对这些位进行改动。这些位只能由电池供电域软件复位清除。任何内部或外部复位都不会影响这些位。

域	简称	复位值	类型	功能
位 31: 17	保留	0x0000	resd	请保持为复位值。
位 16	BPDRST	0x0	rw	电池供电域软件复位 (Battery powered domain software reset) 0: 无复位； 1: 复位。
位 15	ERTCEN	0x0	rw	ERTC 时钟使能 (ERTC clock enable) 由软件置'1'或清'0' 0: 关闭 ERTC 时钟； 1: 开启 ERTC 时钟。
位 14: 10	保留	0x00	resd	请保持为复位值。
位 9: 8	ERTCSEL	0x0	rw	ERTC 时钟源选择 (ERTC clock source selection) 确定了 ERTC 时钟选择后，如果想要再次更改，必须设置 BPDRST 位复位后，才能重新改写 ERTC 时钟选择。 00: 无； 01: LEXT； 10: LICK； 11: HEXT 分频时钟 (分频由 CRM_CFG 的 ERTC_DIV 设定)。
位 7: 5	保留	0x00	resd	请保持为复位值。
位 4: 3	LEXTDRV	0x3	rw	LEXT 驱动 (Low speed external crystal driving strength) 00: LOW； 01: MEDIUM LOW； 10: MEDIUM HIGH； 11: HIGH。。
位 2	LEXTBYP	0x0	rw	LEXT 旁路使能 (Low speed external crystal bypass) 0: 关闭； 1: 开启。
位 1	LEXTSTBL	0x0	ro	LEXT 稳定 (External low-speed oscillator stable) 该位待 LEXT 稳定后由硬件置起。

				0: 未稳定; 1: 已稳定。
位 0	LEXTEN	0x0	rw	LEXT 使能 (External low-speed oscillator enable) 0: 关闭; 1: 开启。

4.3.21 控制/状态寄存器 (CRM_CTRLSTS)

除复位标志外由系统复位清除, 复位标志能由电源复位或写 RSTFC 位进行清除。访问: 0 到 3 等待周期, 字、半字和字节访问; 当连续对该寄存器进行访问时, 将插入等待状态。

域	简称	复位值	类型	功能
位 31	LPRSTF	0x0	ro	低功耗复位标志 (Low-power reset flag) 该位由硬件置起, 软件写 RSTFC 位清除。 0: 无; 1: 有。
位 30	WWDTRSTF	0x0	ro	窗口看门狗复位标志 (WWDT reset flag) 该位由硬件置起, 软件写 RSTFC 位清除。 0: 无; 1: 有。
位 29	WDTRSTF	0x0	ro	看门狗复位标志 (WDT reset flag) 该位由硬件置起, 软件写 RSTFC 位清除。 0: 无; 1: 有。
位 28	SWRSTF	0x0	ro	软件复位标志 (Software reset flag) 该位由硬件置起, 软件写 RSTFC 位清除。 0: 无; 1: 有。
位 27	PORRSTF	0x1	ro	上电/低电压复位标志 (POR/LVR reset flag) 该位由硬件置起, 软件写 RSTFC 位清除。 0: 无; 1: 有。
位 26	NRSTF	0x1	ro	NRST 引脚复位标志 (NRST reset flag) 该位由硬件置起, 软件写 RSTFC 位清除。 0: 无; 1: 有。
位 25	保留	0x0	resd	请保持为复位值。
位 24	RSTFC	0x0	rw	清除复位标志 (Reset flag clear) 由软件置'1'来清除复位标志。 0: 无作用; 1: 清除复位标志。
位 23: 2	保留	0x000000	resd	请保持为复位值。
位 1	LICKSTBL	0x0	ro	LICK 稳定 (LICK stable) 0: 未稳定; 1: 已稳定。
位 0	LICKEN	0x0	rw	LICK 使能 (LICK enable) 0: 关闭; 1: 开启。

4.3.22 额外寄存器 (CRM_MISC1)

访问: 无等待周期, 字, 半字和字节访问。

域	简称	复位值	类型	功能
位 31: 28	CLKOUTDIV2	0x0	rw	CLKOUT 分频因子 2 (Clock output division2) 0xxx: 不分频 1000: 2 分频; 1001: 4 分频; 1010: 8 分频; 1011: 16 分频; 1100: 64 分频; 1101: 128 分频; 1110: 256 分频;

位 27: 20 保留	0x00	resd	1111: 512 分频。 请保持为复位值。
-------------	------	------	---------------------------

位 19: 16 CLKOUT_SEL2	0xF	rw	CLKOUT 时钟选择位 2 (Clock output sel2) 0000: USBFS 48M 时钟输出 0001: ADC 时钟输出 0010: 内部 RC 振荡器时钟 (HICK) 除频输出 0011: LICK 时钟输出 0100: LEXT 时钟输出 0101: 保留 0110: 保留 0111: 保留 1xxx: 保留
----------------------	-----	----	---

位 15 保留	0x0	resd	请保持为复位值。
位 14 HICK_TO_SCLK	0x0	rw	HICK 作为系统时钟的频率选择位 (HICK as system clock frequency select) 当 SCLKSEL 选择 HICK 为时钟源时, SCLK 的频率为 0: 固定是 8Mhz, 即选择 HICK 时钟的 6 分频; 1: 48MHz。
位 13 保留	0x0	resd	请保持为复位值。
位 12 保留	0x1	resd	请保持为复位值。
位 11: 8 保留	0x0	resd	请保持为复位值。
位 7: 0 HICKCAL_KEY	0x00	rw	HICKCAL 写入键值 (HICK calibration key) 此字段为 0x5A 时, HICKCAL [7: 0]才可被写入。

4.3.23 额外寄存器2 (CRM_MISC2)

访问: 无等待周期, 字, 半字和字节访问。

域	简称	复位值	类型	功能
位 31: 30 保留		0x1	resd	固定为 0x1, 请勿修改。
位 29: 26 保留		0x0	resd	请保持为复位值。
位 25: 24 VBATHDIV		0x3	rw	VBATAHB 分频因子 (VBATAHB division) 0x: SCLK 不分频 10: SCLK 2 分频 11: SCLK 4 分频
位 23: 22 HEXTDRV		0x2	rw	HEXT 驱动 (High speed external crystal driving strength) 00: LOW; 01: MEDIUM LOW; 10: MEDIUM HIGH; 11: HIGH。
位 21: 19 HEXT_TO_SCLK_DIV	0x00	rw		HEXT 作为系统时钟的分频因子 (HEXT as system clock frequency division) 000: HEXT 001: HEXT/2 010: HEXT/4 011: HEXT/8 100: HEXT/16 101: HEXT/32 其他: 保留
位 18: 16 HICK_TO_SCLK_DIV	0x00	rw		HICK 作为系统时钟的分频因子 (HICK as system clock frequency division) 000: HICK 001: HICK/2 010: HICK/4 011: HICK/8

				100: HICK/16 其他: 保留
位 15	保留	0x0	resd	请保持为复位值。
位 14: 12 APB3DIV		0x5	rw	APB3 分频因子 (APB3 division) HCLK 分频后作为 APB3 时钟。 0xx: 不分频 100: 2 分频 101: 4 分频 110: 8 分频 111: 16 分频
位 11	保留	0x0	resd	请保持为复位值。
位 10	PLL_U_USB48_SEL	0x0	rw	USBFS 48M 时钟选择源 0: PLLU 1: HICK
位 9	EMAC_PPS_SEL	0x0	rw	以太网 PPS 输出选择 (EMAC PPS select) 请参照 27.3.55 POFC 位
位 8: 6	保留	0x0	resd	请保持为复位值。
位 5: 4 AUTO_STEP_EN		0x0	rw	自动滑顺频率切换 (auto step system clock switch enable) 为使切换系统时钟源到 PLL 或是切换 AHB 分频因子由大到小时平顺(系统频率由小变大)，建议系统频率操作目标大于 108MHz 时启动自动滑顺频率切换。 当自动滑顺频率切换功能作用时，硬件会暂停 AHB 总线，直到整个自动滑顺频率切换完成才恢复。此期间 DMA 仍正常工作，中断事件会被记忆并待 AHB 总线恢复后 NVIC 即可处理。 00: 关闭; 01: 保留; 10: 保留; 11: 开启，当 AHBDIV 或 SCLKSEL 这两个控制位被改动时，会自动触发自动滑顺频率切换功能。
位 3: 0	保留	0xD	resd	固定为 0xD，请勿修改。

5 闪存控制器 (FLASH)

5.1 FLASH介绍

闪存由主存储器、信息块、闪存寄存器这三个部分组成。

- 主存储器容量可达 512K 字节
- 信息块由 26K 字节的系统启动程序代码区和用户系统数据区组成。系统启动程序使用芯片通讯外设实现 ISP 编程

主存储器只有闪存容量为 512K 字节的片 1 闪存，包含 256 扇区，每扇区大小为 2K 字节。

表 5-1 闪存存储结构 (512K)

结构		名称	地址范围
主存储器	片 1 (Bank1) 512KB	扇区 0	0x0800 0000 – 0x0800 07FF
		扇区 1	0x0800 0800 – 0x0800 0FFF
		扇区 2	0x0800 1000 – 0x0800 17FF
	
		扇区 255	0x0807 F800 – 0x0807 FFFF
		启动程序代码区 26KB	0x1FFF 7C00 – 0x1FFF E3FF
信息块	信息块	OTP DATA 4KB	0x1FFF E400 – 0x1FFF F3FF
		OTP LOCK 128B	0x1FFF F500 – 0x1FFF F57F
		用户系统数据区 512B	0x1FFF F800 – 0x1FFF F9FF

主存储器只有闪存容量为 256K 字节的片 1 闪存，包含 128 扇区，每扇区大小为 2K 字节。

表 5-2 闪存存储结构 (256K)

结构		名称	地址范围
主存储器	片 1 (Bank1) 256KB	扇区 0	0x0800 0000 – 0x0800 07FF
		扇区 1	0x0800 0800 – 0x0800 0FFF
		扇区 2	0x0800 1000 – 0x0800 17FF
	
		扇区 127	0x0803 F800 – 0x0803 FFFF
		启动程序代码区 26KB	0x1FFF 7C00 – 0x1FFF E3FF
信息块	信息块	OTP DATA 4KB	0x1FFF E400 – 0x1FFF F3FF
		OTP LOCK 128B	0x1FFF F500 – 0x1FFF F57F
		用户系统数据区 512B	0x1FFF F800 – 0x1FFF F9FF

用户系统数据区

每次系统复位后将从闪存信息块中读出系统数据信息并保存在 FLASH_USD 以及 FLASH_EPPS 寄存器中。

每个系统数据实际占用 2 个字节，低字节对应系统数据的内容，高字节对应系统数据的反码，用于验证选择位的正确性。当读出的高字节不等于低字节的反码时（高字节及低字节均为 0xFF 时除外），系统数据装载器会产生一个系统数据错误的标志 (USDERR)，并把对应的系统数据及其反码都强置为 0xFF。
注意：用户系统数据内容的更新需要一次系统复位才能真正实现。

表 5-3 用户系统数据说明

地址	位	内容
	[7:0]	FAP[7:0]: 闪存访问保护 (访问保护启动/解除结果存放在寄存器 FLASH_USD[1] 以及 [26]) 0xA5: 闪存访问保护解除 0xCC: 高级闪存访问保护启动 其他值: 低级闪存访问保护启动
e0x1FFF_F800_0	[15:8]	nFAP[7:0]: FAP[7:0] 的反码 SSB[7:0]: 系统配置字节 (存放在寄存器 FLASH_USD[9:2])
	[23:16]	位 7 (nRAM_PRT_CHK) 0: 开启 RAM 的奇偶校验 1: 关闭 RAM 的奇偶校验 位 6 (nSTDBY_WDT) 0: 进入待机模式时停止计数 1: 进入待机模式时不停止计数 位 5 (nDEPSLP_WDT) 0: 进入深度睡眠模式时停止计数

			1: 进入深度睡眠模式时不停止计数 保留不用
		位 4: 位 3	0: 进入待机模式时产生复位 1: 进入待机模式时不产生复位
		位 2 (nSTDBY_RST)	0: 进入深度睡眠模式时产生复位 1: 进入深度睡眠模式时不产生复位
		位 1 (nDEPSLP_RST)	0: 进入深度睡眠模式时产生复位 1: 进入深度睡眠模式时不产生复位
		位 0 (nWDT_ATO_EN)	0: 看门狗自启动开启 1: 看门狗自启动关闭
	[31:24]	nSSB[7:0]: SSB[7:0]的反码	
0x1FFF_F804	[7:0]	Data0[7:0]: 用户数据 0 (存放在寄存器 FLASH_USD[17:10])	
	[15:8]	nData0[7:0]: Data0[7:0]的反码	
	[23:16]	Data1[7:0]: 用户数据 1 (存放在寄存器 FLASH_USD[25:18])	
	[31:24]	nData1[7:0]: Data1[7:0]的反码	
0x1FFF_F808	[7:0]	EPP0[7:0]: 闪存擦写保护字节 0 (存放在寄存器 FLASH_EPPS[7:0]) 用于保护主闪存存储器的扇区 0 ~ 扇区 15, 每个比特位保护 4K 字节 0: 擦写保护启动 1: 擦写保护解除	
	[15:8]	nEPP0[7:0]: EPP0[7:0]的反码	
	[23:16]	EPP1[7:0]: 闪存擦写保护字节 1 (存放在寄存器 FLASH_EPPS[15:8]) 用于保护主闪存存储器的扇区 16 ~ 扇区 31, 每个比特位保护 4K 字节 0: 擦写保护启动 1: 擦写保护解除	
	[31:24]	nEPP1[7:0]: EPP1[7:0]的反码	
0x1FFF_F80C	[7:0]	EPP2[7:0]: 闪存擦写保护字节 2 (存放在寄存器 FLASH_EPPS[23:16]) 用于保护主闪存存储器的扇区 32 ~ 扇区 47, 每个比特位保护 4K 字节 0: 擦写保护启动 1: 擦写保护解除	
	[15:8]	nEPP2[7:0]: EPP2[7:0]的反码	
	[23:16]	EPP3[7:0]: 闪存擦写保护字节 3 (存放在寄存器 FLASH_EPPS[31:24]) 位 6 到位 0 用于保护主闪存存储器的扇区 48 ~ 扇区 61, 每个比特位保护 4K 字节 位 7 用于保护 512KB 主闪存存储器的扇区 62 ~ 扇区 255, 256KB 主闪存存储器的扇区 62 ~ 扇区 127, 位 7 同时用于保护 512KB 与 256KB 主闪存存储器的主存扩展区 0: 擦写保护启动 1: 擦写保护解除	
	[31:24]	nEPP3[7:0]: EPP3[7:0]的反码	
0x1FFF_F810 ~ 0x1FFF_F830	[31:0]	保留不用	
0x1FFF_F834	[7:0]	QSPIKEY0[7: 0]: 四线 SPI (QSPI) 密文存取区加密键值字节 0 不加密的设定条件包括: QSPIKEYx 以及 nQSPIKEYx 均为 0xFF (即默认擦除状态) QSPIKEYx 写入 0x00 QSPIKEYx 写入 0xFF 即{nQSPIKEYx, QSPIKEYx}均设为 0xFFFF, 0xFF00, 0x00FF 其中 QSPIKEY0-QSPIKEY3 作为 QSPI1 加密密钥	
	[15:8]	nQSPIKEY0[7: 0]: QSPIKEY0[7: 0]的反码	
	[23:16]	QSPIKEY1[7: 0]: 四线 SPI (QSPI) 密文存取区加密键值字节 1	
	[31:24]	nQSPIKEY1[7: 0]: QSPIKEY1[7: 0]的反码	
0x1FFF_F838	[7:0]	QSPIKEY2[7: 0]: 四线 SPI (QSPI) 密文存取区加密键值字节 2	
	[15:8]	nQSPIKEY2[7: 0]: QSPIKEY2[7: 0]的反码	
	[23:16]	QSPIKEY3[7: 0]: 四线 SPI (QSPI) 密文存取区加密键值字节 3	
	[31:24]	nQSPIKEY3[7: 0]: QSPIKEY3[7: 0]的反码	
0x1FFF_F83C ~ 0x1FFF_F848	[31:0]	保留不用	
0x1FFF_F84C		Bootloader 外设开关使能需要检测 BOOT_PERIP_KEY1 和 BOOT_PERIP_KEY2 都有效, 才会使用 BOOT_PERIP1_EN 和 BOOT_PERIP2_EN 的配置, 其它情况默认开启支持的所有外设。	

0x1FFF_F850	[7:0]	BOOT_PERIP_KEY1[7:0]: Bootloader 外设使能 KEY1 0x4B: 有效 其他值: 无效 (默认开启所有 Bootloader 外设)
	[15:8]	nBOOT_PERIP_KEY1[7:0]: BOOT_PERIP_KEY1 [7:0]的反码
	[23:16]	BOOT_PERIP_KEY2[7:0]: Bootloader 外设使能 KEY2 0x5C: 有效 其他值: 无效 (默认开启所有 Bootloader 外设)
	[31:24]	nBOOT_PERIP_KEY2 [7:0]: BOOT_PERIP_KEY2 [7:0]的反码 BOOT_PERIP_EN1[7:0]: Bootloader 外设使能 1
	[7:0]	位 7 (BOOT_I2C3_EN) 选择是否使能 Bootloader 的 I2C3 0: 关闭 I2C3 1: 使能 I2C3
		位 6 (BOOT_I2C2_EN) 选择是否使能 Bootloader 的 I2C2 0: 关闭 I2C2 1: 使能 I2C2
		位 5 (BOOT_I2C1_EN) 选择是否使能 Bootloader 的 I2C1 0: 关闭 I2C1 1: 使能 I2C1
		位 4 保留
		位 3 (BOOT_USB1_EN) 选择是否使能 Bootloader 的 USB DFU 0: 关闭 USB DFU 1: 使能 USB DFU
		位 2 (BOOT_USART3_EN) 选择是否使能 Bootloader 的 USART2 0: 关闭 USART2 1: 使能 USART2
		位 1 (BOOT_USART2_EN) 选择是否使能 Bootloader 的 USART2 0: 关闭 USART2 1: 使能 USART2
		位 0 (BOOT_USART1_EN) 选择是否使能 Bootloader 的 USART1 0: 关闭 USART1 1: 使能 USART1
	[15:8]	nBOOT_PERIP1_EN[7:0]: BOOT_PERIP_EN1 [7:0]的反码 BOOT_PERIP_EN2[7:0]: Bootloader 外设使能 2
	[23:16]	位 7: 位 4 保留
		位 3 (BOOT_SPI2_EN) 选择是否使能 Bootloader 的 SPI2 0: 关闭 SPI2 1: 使能 SPI2
		位 2 (BOOT_SPI1_EN) 选择是否使能 Bootloader 的 SPI1 0: 关闭 SPI1 1: 使能 SPI1
		位 1 (BOOT_CAN2_EN) 选择是否使能 Bootloader 的 CAN2 0: 关闭 CAN2 1: 使能 CAN2
		位 0 (BOOT_CAN1_EN) 选择是否使能 Bootloader 的 CAN1 0: 关闭 CAN1 1: 使能 CAN1
0x1FFF_F854	[31:24]	nBOOT_PERIP2_EN[7:0]: nBOOT_PERIP2_EN [7:0]的反码
	[7:0]	Data2[7: 0]: 用户数据 2
	[15:8]	nData2[7: 0]: Data2[7: 0]的反码
	[23:16]	Data3[7: 0]: 用户数据 3
	[31:24]	nData3[7: 0]: Data3[7: 0]的反码
0x1FFF_F9FC
	[7:0]	Data214[7: 0]: 用户数据 214
	[15:8]	nData214[7: 0]: Data214[7: 0]的反码
	[23:16]	Data215[7: 0]: 用户数据 215
	[31:24]	nData215[7: 0]: Data215[7: 0]的反码

5.2 主存储器操作

5.2.1 解锁/锁定

复位后，主存储器默认是被锁定的，此时不允许配置 FLASH_CTRL 寄存器，需要对闪存解锁后才能成功实现对闪存的写入与擦除操作。

解锁流程：

对 FLASH_UNLOCK 寄存器顺序写入键值 KEY1 (0x45670123) 和键值 KEY2 (0xCDEF89AB)，能够解锁对应区域闪存。

注意：解锁必须顺序写入正确的键值，否则会产生总线错误并且闪存会被锁死，直到下一次复位才能恢复。

锁定流程：

软件置起闪存控制寄存器（FLASH_CTRL）中的 OPLK 位，锁定对应区域闪存。

5.2.2 擦除

编程之前必须先进行擦除操作，主存储器有扇区擦除和整片擦除两种擦除方式。

扇区擦除

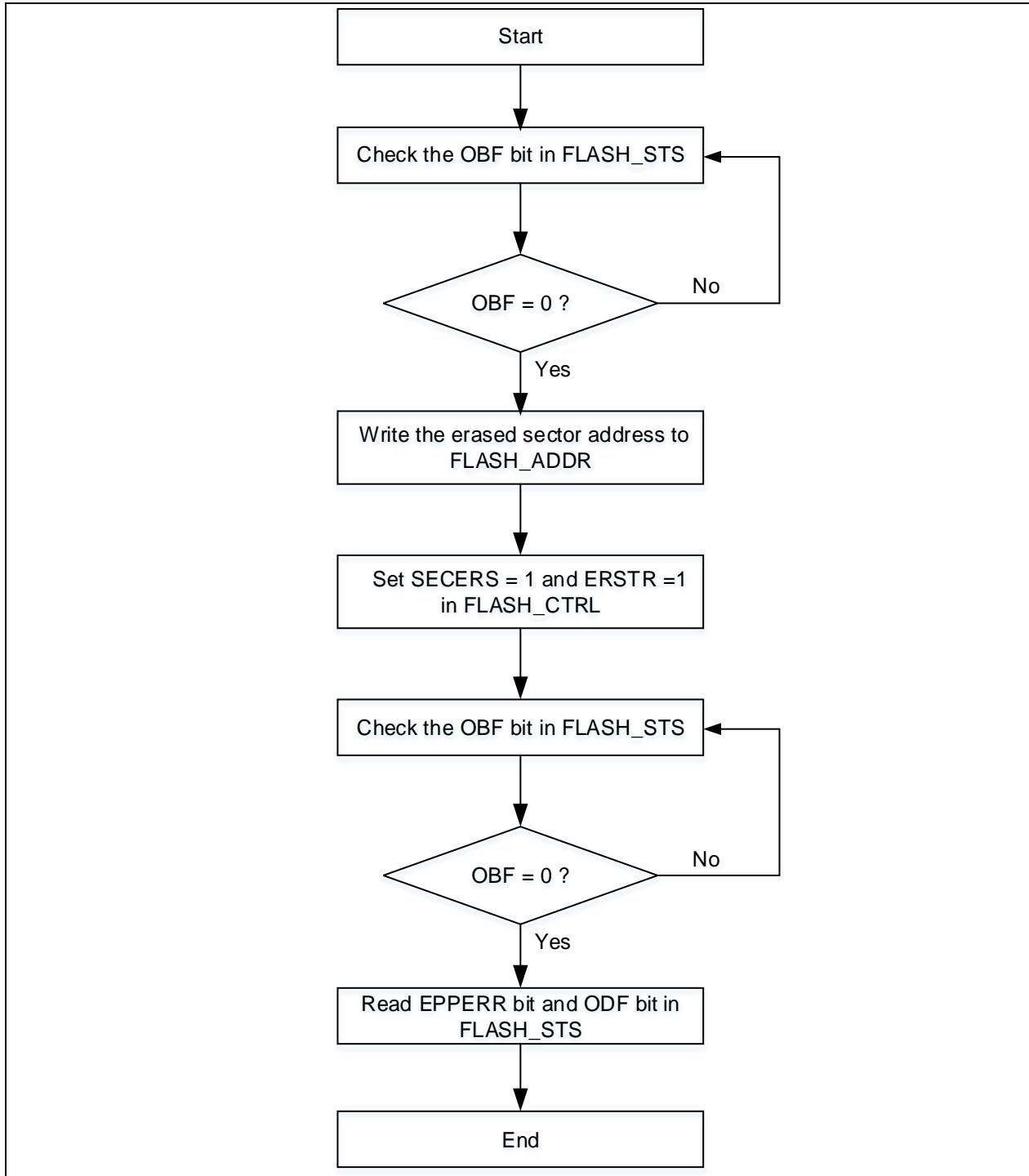
主闪存存储器的每一扇区以及主存扩展区都可以使用扇区擦除功能独立擦除。

擦除流程如下：

- 检查闪存状态寄存器（FLASH_STS）的OBF位，确认没有正在进行的闪存操作；
- 对FLASH_ADDR寄存器写入要擦除的扇区地址；
- 对闪存控制寄存器（FLASH_CTRL）的SECERS位以及ERSTR位均置1，启动扇区擦除；
- 等待闪存状态寄存器（FLASH_STS）的OBF位变为‘0’，并查询闪存状态寄存器（FLASH_STS）的EPPER位和ODF位，确认擦除结果。

注意：当启动程序代码区域是设定为主存扩展区时，执行扇区擦除实际上是对整个主存扩展区的擦除。

图 5-1 主存储器扇区擦除流程图



整片擦除

主闪存存储器可以使用整片擦除功能直接擦除。

擦除流程如下：

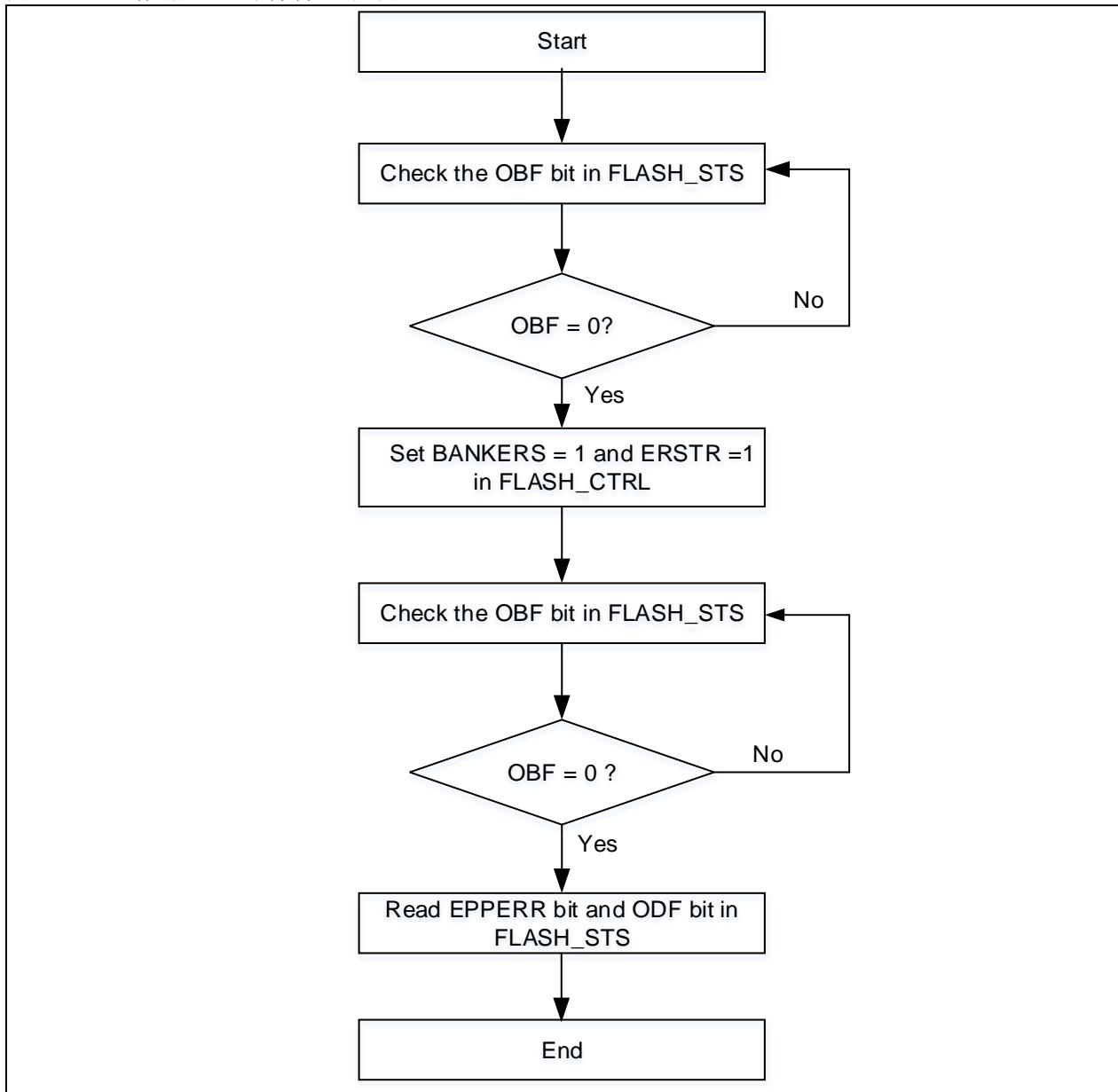
- 检查闪存状态寄存器（FLASH_STS）的OBF位，确认没有正在进行的闪存操作；
- 对闪存控制寄存器（FLASH_CTRL）的BANKERS位以及ERSTR位均置1，启动整片擦除；
- 等待闪存状态寄存器（FLASH_STS）的OBF位变为‘0’，并查询闪存状态寄存器（FLASH_STS）的EPPERR位和ODF位，确认擦除结果。

注意：

- 1) 当启动程序代码区域是设定为主存扩展区时，执行整片擦除操作会自动擦除主闪存以及主存扩展区。

- 2) 擦除期间进行读闪存的操作，将导致 CPU 会被暂停直到擦除完成才处理读闪存操作。
 3) 擦除操作前必须保证内部的 HICK 有打开

图 5-2 主存储器整片擦除流程图



5.2.3 编程

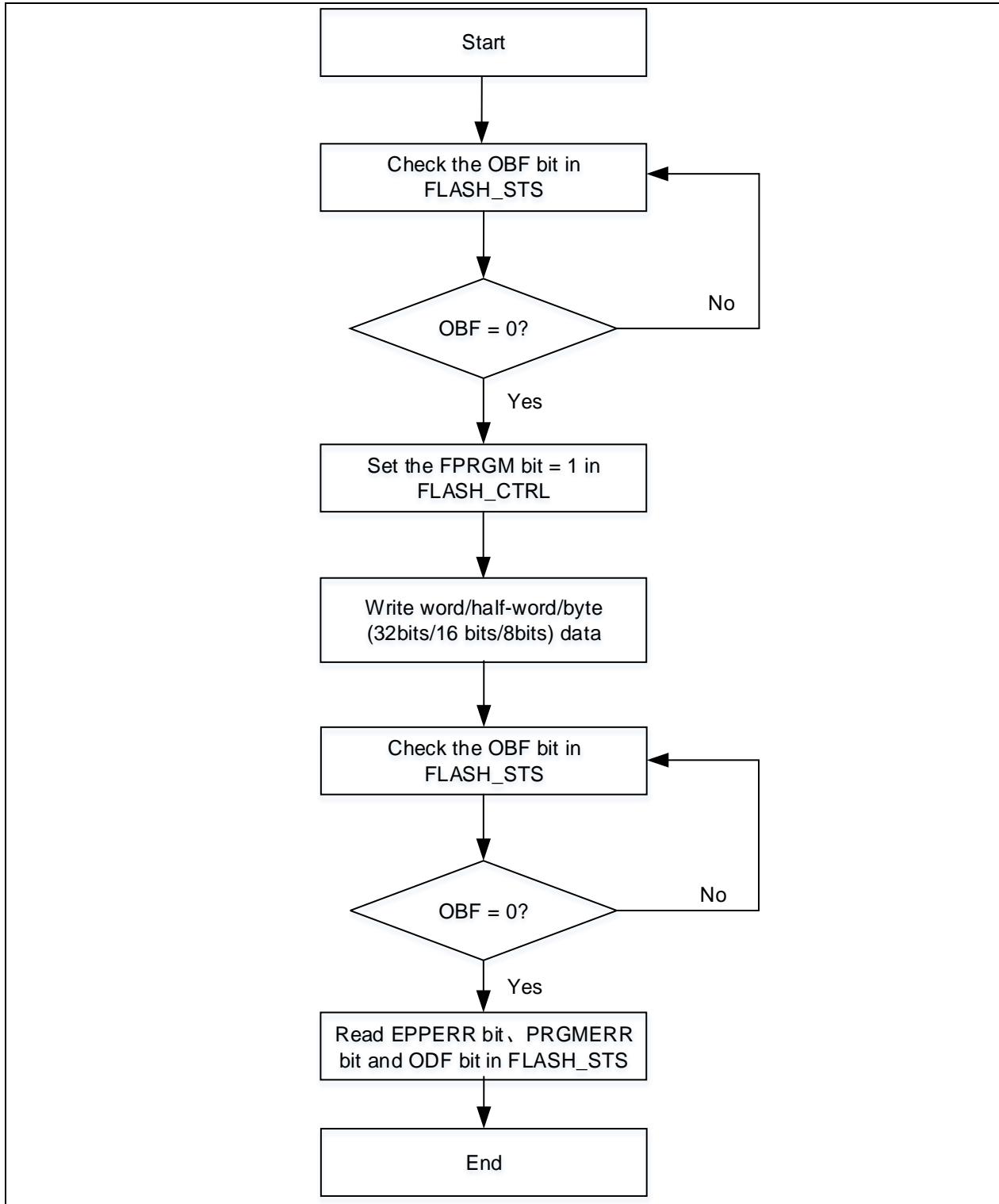
当想要改写主存储器的内容时，可以通过主存储器编程流程完成一次写入 32 位、16 位或 8 位的数据。主存储器编程流程：

- 检查闪存状态寄存器（FLASH_STS）的 OBF 位，确认没有正在进行的闪存操作；
- 对闪存控制寄存器（FLASH_CTRL）的 FPRGM 位置 1，此时可以接受对主闪存的编程指令；
- 对指定的地址写入要编程的数据（任意字/半字/字节）；
- 等待闪存状态寄存器（FLASH_STS）的 OBF 位变为‘0’，并查询闪存状态寄存器（FLASH_STS）的 EPPERR 位、PRGMERR 位和 ODF 位，确认编程结果。

注意：

- 1) 当要写入的地址未被提前擦除时，除非要写入的数据值是全 0，否则编程不被执行，并置位闪存状态寄存器（FLASH_STS）的 PRGMERR 位来告知编程发生错误。
- 2) 编程期间，CPU 会被暂停直到编程完成才处理后续操作。
- 3) 编程操作前必须保证内部的 HICK 有打开

图 5-3 主存储器编程流程图



5.2.4 读取

通过 CPU 的 AHB 总线可以直接寻址访问主闪存存储区。

5.3 主存扩展区操作

启动程序代码区也可以设定为主存扩展区存放用户应用代码。当作为主存扩展区时，其操作方法，包括读取、解锁、擦除、编程都跟主存储器相同。

5.4 OTP 操作

OTP 的 DATA 与 LOCK 区不可擦除，其操作方法，包括读取、解锁、编程都跟主存储器相同。

每 32 字节 DATA 对应 1 字节 LOCK，当 LOCK 字节被写为 0x00 时，该 32 字节 DATA 无法再写入。

DATA 与 LOCK 同样有访问保护，请参照表闪存访问权限。

表 5-4 OTP 的 DATA 与 LOCK 关系

OTP LOCK 地址范围	OTP DATA 地址范围
LOCK0	0x1FFF F500 – 0x1FFF F501
LOCK1	0x1FFF F501 – 0x1FFF F502
LOCK2	0x1FFF F502 – 0x1FFF F503
...	...
LOCK127	0x1FFF F57F – 0x1FFF F580

5.5 用户系统数据区操作

5.5.1 解锁/锁定

复位后，用户系统数据区默认是锁定的，需要在闪存解锁后再对用户系统数据区解锁才能成功实现写入与擦除操作。

解锁流程：

对 FLASH_UNLOCK 寄存器顺序写入键值 KEY1 (0x45670123) 和键值 KEY2 (0xCDEF89AB)；

对 FLASH_USD_UNLOCK 寄存器顺序写入键值 KEY1 (0x45670123) 和键值 KEY2 (0xCDEF89AB)，闪存控制寄存器 (FLASH_CTRL) 中的 USDULKS 位将被硬件自动置起，表示允许对用户系统数据区的写、擦除操作。

注意：解锁必须顺序写入正确的键值，否则会产生总线错误并且闪存会被锁死，直到下次复位才能恢复。

锁定流程：

软件清除闪存控制寄存器 (FLASH_CTRL) 中的 USDULKS 位，锁定用户系统数据区。

5.5.2 擦除

在编程之前必须先进行擦除操作，用户系统数据区域可单独实现擦除功能。

擦除流程如下：

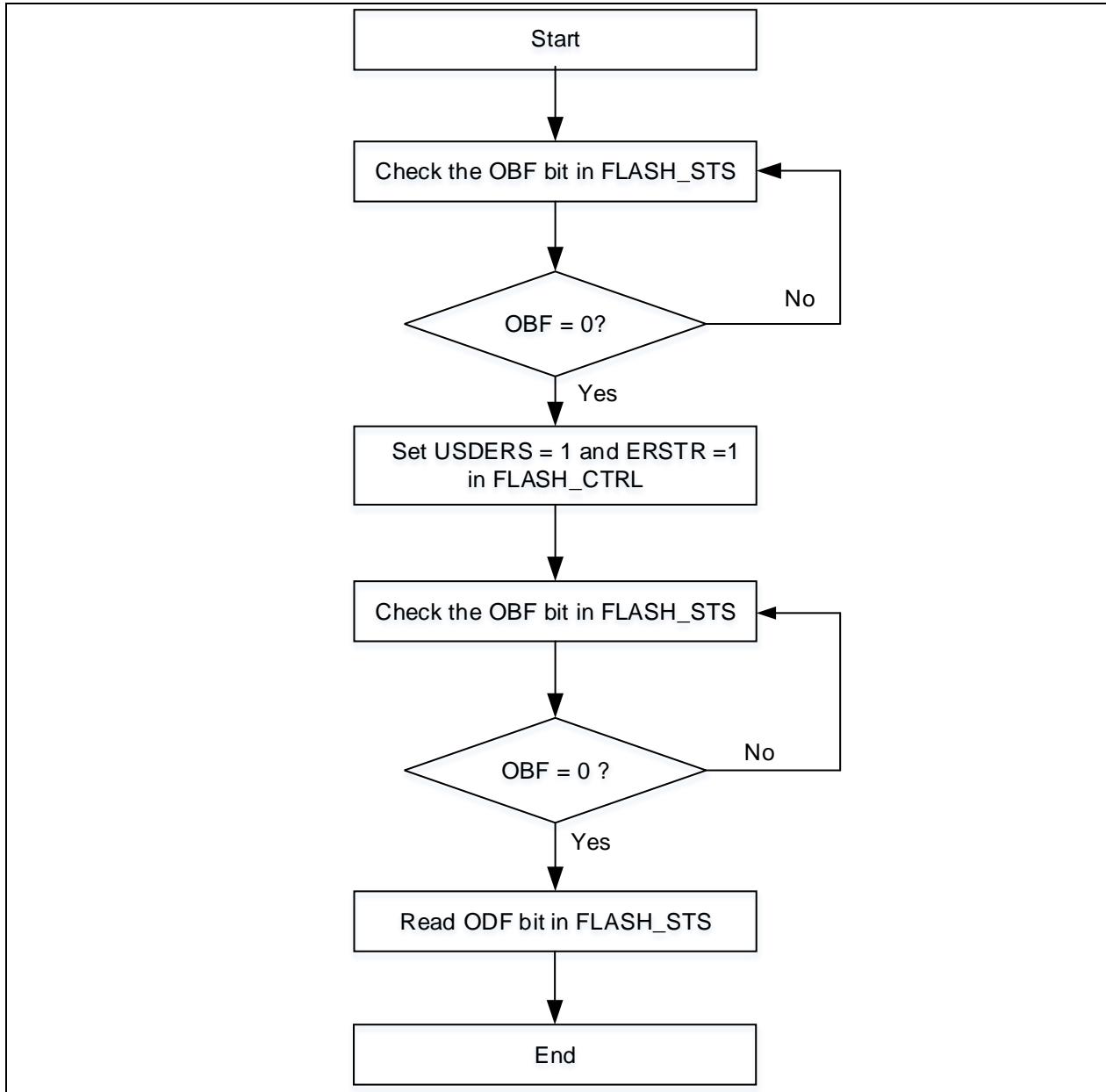
- 检查闪存状态寄存器 (FLASH_STS) 的 OBF 位，确认没有正在进行的闪存操作；
- 对闪存控制寄存器 (FLASH_CTRL) 的 USDRS 位以及 ERSTR 位均置 1，启动整块系统数据区擦除；
- 等待闪存状态寄存器 (FLASH_STS) 的 OBF 位变为‘0’，并查询闪存状态寄存器 (FLASH_STS) 的 ODF 位，确认擦除结果。

注意：

擦除期间进行读闪存的操作，将导致 CPU 会被暂停直到擦除完成才处理读闪存操作。

擦除操作前必须保证内部的 HICK 有打开。

图 5-4 系统数据区擦除图



5.5.3 编程

当想要改写用户系统数据区域的内容时，可以通过用户系统数据区编程流程完成一次写入 32 位或 16 位数据。

系统数据区的编程流程：

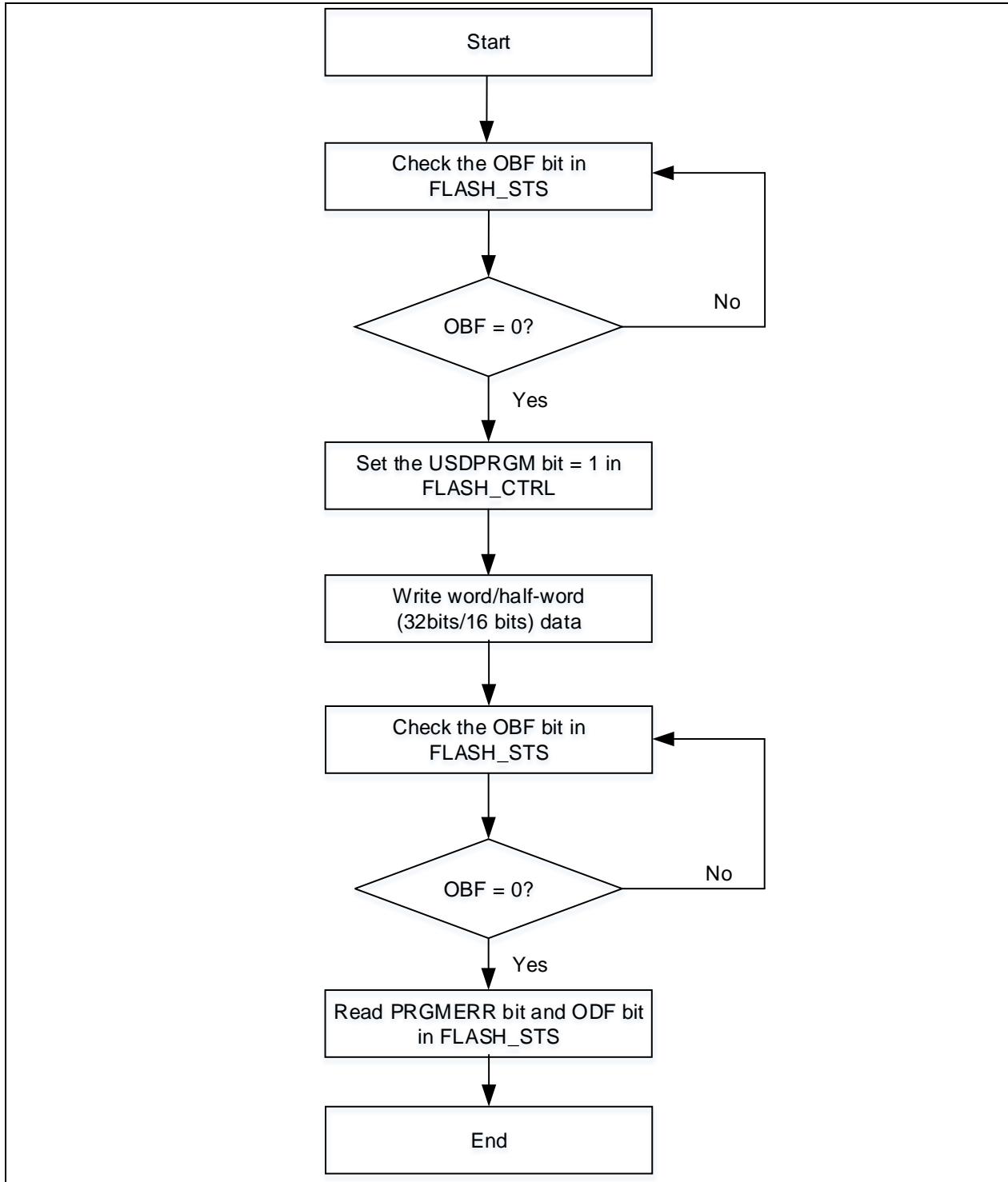
- 检查闪存状态寄存器（FLASH_STS）的OBF位，确认没有正在进行的闪存操作；
- 对闪存控制寄存器（FLASH_CTRL）的USDPRGM位置1，此时可以接受对用户系统数据区的编程指令；
- 对指定的地址写入要编程的数据（任意字/半字）；
- 等待闪存状态寄存器（FLASH_STS）的OBF位变为‘0’，并查询闪存状态寄存器（FLASH_STS）的PRGMERR位和ODF位，确认编程结果。

注意：

编程期间，CPU 会被暂停直到编程完成才处理后续操作。

编程操作前必须保证内部的 HICK 有打开。

图 5-5 系统数据区编程图



5.5.4 读取

通过 CPU 的 AHB 总线可以直接寻址访问用户系统数据区。

5.6 闪存保护

闪存存储器有访问保护以及擦写保护两种保护方式。

5.6.1 访问保护

闪存访问保护分为两类：闪存低级访问保护，闪存高级访问保护。

访问保护启动后，只允许闪存程序对闪存存储器数据进行读出访问，禁止在调试模式下或是从非主闪存存储器启动对闪存存储器数据的读出访问。

闪存低级访问保护

当 nFAP 字节和 FAP 字节存放的内容不等于 0x5A 和 0xA5 以及不等于 0x33 和 0xCC 时，闪存在系统复位后，将启动闪存低级访问保护。

此保护下，用户可以重新擦除系统数据区，并对 FAP 字节写入 0xA5 解除闪存低级访问保护（从低级保护状态变为未保护状态，将自动产生对主闪存以及主存扩展区的整片擦除操作），最后进行系统复位，系统数据装载器重新加载系统数据信息，更新闪存访问保护解除信息（FAP 字节）。

闪存高级访问保护

当 nFAP 字节存放的内容等于 0x33，并且 FAP 字节存放的内容等于 0xCC 时，闪存在系统复位后，将启动闪存高级访问保护。

一旦此保护启动后，将不能被解除，并且禁止用户以任何方式重新擦除以及写入系统数据区。

注意：

1) 主存扩展区也支持访问保护功能

2) 如果访问保护被置位的时候仍然处于调试模式，必须用 POR（上电复位）代替系统复位
清除调试模式，才能恢复闪存程序访问闪存存储器数据的权限。

下表是启动闪存访问保护后，闪存不同区域访问权限说明：

表 5-5 闪存访问权限

区域		保护等级		访问权限					
		调试模式或是从 SRAM 启动以及从启动程序代码区启动				从主闪存启动			
		读	写	擦除	读	写	擦除		
主闪存区	低级访问保护	禁止		禁止 (1) (2)	允许				
	高级访问保护	无 (3)			允许				
用户系统数据区	低级访问保护	禁止	允许		允许				
	高级访问保护	无 (3)			允许	禁止			
OTP 与 LOCK 区	低级访问保护	禁止			允许	禁止			
	高级访问保护	无 (3)			允许	禁止			

(1) 主闪存区会在解除闪存访问保护时被硬件自动擦除

(2) 只禁止扇区擦除，允许整片擦除

(3) 高级访问保护开启时，系统如果配置启动程序代码区启动将自动从主闪存启动

5.6.2 擦写保护

擦写保护的基本单位为 4KB。通过擦写保护可以防止程序在跑飞时闪存存储器的内容被意外更改。

在下面列出的情况下，擦写将不被允许，并会置位 EPPERR 位：

- 对被设置为擦写保护的扇区（主闪存以及闪存扩展区）做扇区擦除操作以及编程操作将不被允许
- 对存在任一扇区被设置为擦写保护的主闪存以及闪存扩展区做整片擦除将不被允许
- 闪存访问保护启动后，主闪存扇区 0~1 将被自动擦写保护，不允许做扇区擦除操作以及编程操作
- 闪存访问保护启动后，主存储器和主存扩展区在调试模式或是从非主闪存存储器启动下被自动擦写保护，不允许做扇区擦除操作以及编程操作

5.7 读取性能

提升系统时钟频率前须先按照闪存性能选择寄存器 (FLASH_PSR) 的 WTCYC 位说明配置读取闪存须插入的延迟时间。

使能闪存性能选择寄存器 (FLASH_PSR) 的 PFT_EN 位、PFT_EN2 位与 PFT_LAT_DIS 位可降低去闪存读取次数。

5.8 特殊功能

5.8.1 安全库区设定

设定以密码保护主存中指定范围的程序区，即安全库区，仅能被执行，无法写入与删除，除非输入指定密码。安全库区划分为 指令区 与 唯读区。指令区无法读取，唯独区可被读取。

设定安全库区的益处：

以密码保护安全库区，方案商可刻录核心算法到此区域；

安全库区仅能执行，无法被读取，除非输入方案商指定密码，也无法删除（包含 ISP/IAP/SWD）；

其余空白程序区可以提供给方案商客户进行二次开发；

方案商可以藉由安全库功能销售核心算法，不需要每个客户都开发完整方案；

设定安全库区，可防止蓄意破坏或更改终端产品应用程序代码。

注意：

安全库区代码必须以扇区为单位进行烧录，并且起始地址与主存地址对齐；

仅允许 CPU 指令读取指令库区；

写入或擦除安全库区代码（指令区和唯读区），将在闪存状态寄存器 (FLASH_STS) 的 EPERR 位置'1'提出警告；

执行主存的整片擦除时，将不会擦除安全库区。

默认状态下，安全库区设定寄存器始终是不可读且被锁定的。要想对安全库区设定寄存器进行写操作，首先要对安全库区解锁，对 SLIB_UNLOCK 寄存器写入 0xA35F6D24 值，通过查看闪存安全库区额外状态寄存器 (SLIB_MISC_STS) 的位 SLIB_ULKF 确认解锁成功，随后对安全库区设定寄存器写入设定值。启动安全库区的流程如下：

- 检查闪存状态寄存器 (FLASH_STS) 的 OBF 位，以确认没有其他正在进行的编程操作；
- 对 SLIB_UNLOCK 寄存器写入 0xA35F6D24，以进行安全库区解锁；
- 检查闪存安全库区额外状态寄存器 (SLIB_MISC_STS) 的 SLIB_ULKF 位，以确认解锁成功；
- 如果安全库区设在主闪存内，需要在 SLIB_SET_RANGE 寄存器设定要保护的扇区，包含指令区与唯读区的地址；如果安全库区设在主存扩展区域，需要设定 EM_SLIB_SET 寄存器。
- 等待 OBF 位变为 '0'；
- 在 SLIB_SET_PWD 寄存器设定安全库区密码；
- 等待 OBF 位变为 '0'；
- 烧录将存入安全库区的代码；
- 进行系统复位，重装载安全库区设定字；
- 读出 SLIB_STS0/STS1 寄存器用于判断安全库区设定结果。

注意：

不支持同时设定主闪存和主存扩展区域为安全库区；

启动安全库区的流程需要在闪存访问保护未启动时执行

解除安全库区的流程是：

- 在 SLIB_PWD_CLR 寄存器写入先前设置的安全区域密码；
- 等待 OBF 位变为 '0'；
- 进行系统复位，重装载安全库区设定字；
- 读出 SLIB_STS0 寄存器用于判断安全库区解除结果。

注意：解除安全库区将会自动执行主存及主存扩展区的整片擦除，以及安全库设定块擦除。

5.8.2 启动程序代码区域作为主存扩展使用

用户只有一次机会将启动程序代码区域作为主存扩展使用。一旦设定成功，主存扩展区将具有主闪存特性。设定启动程序代码区域作为主存扩展使用的流程是：

- 用户读取SLIB_STS0的位0，获知启动程序代码区域当前的模式
- 对SLIB_UNLOCK寄存器写入0xA35F6D24，以进行启动程序代码区域模式设定解锁
- 写非0xFF到BTM_MODE_SET寄存器的位7-0
- 等待OBF位变为'0'；
- 进行系统复位，重装载设定字；
- 读出SLIB_STS0寄存器用于判断设定结果。

注意：启动设定主存扩展区的流程需要在闪存访问保护未启动时执行
当开启该功能后，原启动程序存储器启动模式将强制变为主闪存存储器启动模式。

5.8.3 CRC校验

以扇区为单位对安全库区代码或用户代码进行可选的CRC校验。

- CRC生成多项式：0x4C11DB7，
即 $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
- CRC初始值：0x00000000

校验流程如下：

- 检查闪存状态寄存器(FLASH_STS)的OBF位，以确认没有其他正在进行的编程操作；
- 在FLASH_CRC_ADDR寄存器设定要校验的代码起始地址；
- 在FLASH_CRC_CTRL寄存器位15-0，设定要校验的代码数量（单位是扇区）；
- 在FLASH_CRC_CTRL寄存器设置位16，启动CRC校验；
- 等待OBF位变为'0'；
- 读出FLASH_CRC_CHK寄存器用于判断CRC校验结果。

注意：

FLASH_CRC_ADDR寄存器设定值必须与扇区起始地址对齐；
不允许跨主存及主存扩展区的CRC校验。

5.9 FLASH寄存器

必须用字（32位）的方式操作这些外设寄存器。

表 5-6 闪存接口寄存器映射和复位值

寄存器简称	基址偏移量	复位值
FLASH_PSR	0x00	0x0000 01F0
FLASH_UNLOCK	0x04	0xFFFFFFFF
FLASH_USD_UNLOCK	0x08	0xFFFFFFFF
FLASH_STS	0x0C	0x0000 0000
FLASH_CTRL	0x10	0x0002 0080
FLASH_ADDR	0x14	0x0000 0000
FLASH_USD	0x1C	0x03FF FFFF
FLASH_EPPS	0x20	0xFFFF FFFF
SLIB_STS0	0x74	0x00FF 0000
SLIB_STS1	0x78	0xFFFF FFFF
SLIB_PWD_CLR	0x7C	0xFFFF FFFF
SLIB_MISC_STS	0x80	0x0000 0000

FLASH_CRC_ADDR	0x84	0x0000 0000
FLASH_CRC_CTRL	0x88	0x0000 0000
FLASH_CRC_CHKR	0x8C	0x0000 0000
SLIB_SET_PWD	0x160	0x0000 0000
SLIB_SET_RANGE	0x164	0x0000 0000
EM_SLIB_SET	0x168	0x0000 0000
BTM_MODE_SET	0x16C	0x0000 0000
SLIB_UNLOCK	0x170	0x0000 0000

5.9.1 闪存性能选择寄存器 (FLASH_PSR)

域	简称	复位值	类型	功能
位 31:9	保留	0x0000000	resd	保持为默认值。
位 8	PFT_LAT_DIS	1	rw	预取时延无效 (Prefetch latency disable) 0: 闪存预取缓冲区时延开启, 访问缓冲区需要等待 1 个系统时钟周期; 1: 闪存预取缓冲区时延关闭, 访问缓冲区零等待。 推荐此位配置为 1, 后续请勿修改。
位 7	PFT_ENF2	1	ro	预取使能标志 2 (Prefetch enabled flag2) 该位置起时, 表示启动闪存预取缓冲区第二数据块
位 6	PFT_EN2	1	rw	预取使能 2 (Prefetch enable2) 0: 闪存预取缓冲区第二数据块关闭; 1: 闪存预取缓冲区第二数据块开启。 推荐此位配置为 1, 后续请勿修改。
位 5	PFT_ENF	1	ro	预取使能标志 (Prefetch enabled flag) 该位置起时, 表示启动闪存预取缓冲区
位 4	PFT_EN	1	rw	预取使能 (Prefetch enable) 0: 闪存预取缓冲区关闭; 1: 闪存预取缓冲区开启。
位 3	保留	0	resd	保持为默认值。 等待周期 (Wait cycle) 需要根据系统时钟大小来设定闪存访问的等待周期, 以系统时钟为单位。 000: 零个等待周期, 0MHz < 系统时钟 ≤ 32MHz 使用; 001: 一个等待周期, 32MHz < 系统时钟 ≤ 64MHz 使用; 010: 两个等待周期, 64MHz < 系统时钟 ≤ 96MHz 使用; 011: 三个等待周期, 96MHz < 系统时钟 ≤ 128MHz 使用; 100: 四个等待周期, 128MHz < 系统时钟 ≤ 160MHz 使用; 101: 五个等待周期, 160MHz < 系统时钟 ≤ 192MHz 使用; 110: 六个等待周期, 192MHz < 系统时钟 ≤ 224MHz 使用; 111: 七个等待周期, 224MHz < 系统时钟 ≤ 256MHz 使用。
位 2:0	WTCYC	0x0	rw	

5.9.2 闪存解锁寄存器 (FLASH_UNLOCK)

域	简称	复位值	类型	功能
位 31:0	UKVAL	0xFFFF XXXX wo		解锁键值 (Unlock key value) 该寄存器用于解锁主闪存及闪存扩展区。

注意: 所有这些位是只写的, 读出时返回 0。

5.9.3 闪存用户系统数据解锁寄存器（FLASH_USD_UNLOCK）

域	简称	复位值	类型	功能
位 31:0	USD_UKVAL	0xXXXX XXXX wo		用户系统数据解锁键值（User system data unlock key value）

注意：所有这些位是只写的，读出时返回 0。

5.9.4 闪存状态寄存器（FLASH_STS）

域	简称	复位值	类型	功能
位 31:6	保留	0x0000000	resd	保持为默认值
位 5	ODF	0	rwc1	操作完成标志（Operation done flag） 当闪存操作（编程/擦除）成功完成时，硬件会置起该位，软件写'1'可以清除。
位 4	EPPERR	0	rwc1	擦写保护错误（Erase/Program protection error） 当擦除或编程的闪存地址在擦写保护设定范围内时，硬件会置起该位，软件写'1'可以清除。
位 3	保留	0	resd	保持为默认值
位 2	PRGMERR	0	rwc1	编程错误（Program error） 当编程的闪存地址的值为非擦除状态时，硬件会置起该位，软件写'1'可以清除。
位 1	保留	0	resd	保持为默认值
位 0	OBF	0	ro	操作忙标志（Operation busy flag） 该位置起表示闪存操作正在进行，该位清除表示操作结束。

5.9.5 闪存控制寄存器（FLASH_CTRL）

域	简称	复位值	类型	功能
位 31:13	保留	0x0000	resd	保持为默认值
位 12	ODFIE	0	rw	操作完成中断使能（Operation done flag interrupt enable） 0: 关闭； 1: 开启。
位 11,8,3	保留	0	resd	保持为默认值
位 10	ERRIE	0	rw	错误中断使能（Error interrupt enable） 开启后 EPPERR 或 PRGMERR 都会产生中断。 0: 关闭； 1: 开启。
位 9	USDULKS	0	rw	用户系统数据解锁成功（User system data unlock success） 一旦用户系统数据区解锁成功，该位将被硬件自动置起，表示允许对用户系统数据的编程/擦除操作。软件写'0'可以清除此位，重新锁定用户系统数据区。
位 7	OPLK	1	rw	操作锁定（Operation lock） 该位默认处于置起状态，表示闪存锁定，锁定时不允许操作，解锁成功后，硬件会自动清除此位，表示允许闪存编程/擦除操作。软件写'1'可以重新锁定闪存操作。
位 6	ERSTR	0	rw	擦除开始（Erasing start） 软件置起该位，开始执行擦除操作。擦除完成后硬件自动清除该位。
位 5	USDERS	0	rw	用户系统数据擦除（User system data erase） 用户系统数据区擦除。
位 4	USDPRGM	0	rw	用户系统数据编程（User system data program） 用户系统数据编程。
位 2	BANKERS	0	rw	片擦除（Bank erase） 擦除片操作。
位 1	SECERS	0	rw	扇区擦除（Sector erase） 擦除扇区操作。
位 0	FPRGM	0	rw	闪存编程（Flash program） 编程操作。

5.9.6 闪存地址寄存器 (FLASH_ADDR)

域	简称	复位值	类型	功能
位 31:0	FA	0x0000 0000	wo	闪存地址 (Flash address) 扇区擦除时选择对应的闪存扇区地址。

5.9.7 用户系统数据寄存器 (FLASH_USD)

域	简称	复位值	类型	功能
位 31:27	保留	0x00	resd	保持为默认值
位 26	FAP_HL	0	ro	闪存访问保护高级 (Flash access protection high level) 闪存访问保护状态使用 {位 26, 位 1}联合判断。 00: 未启动访问保护, 且 FAP 值=0xA5 01: 启动低级访问保护, 且 FAP 值非 0xCC 以及 0xA5 10: 保留 11: 启动高级访问保护, 且 FAP 值=0xCC
位 25:18	USER_D1	0xFF	ro	用户数据 1
位 17:10	USER_D0	0xFF	ro	用户数据 0
位 9:2	SSB	0xFF	ro	系统配置字节 (System setting byte) 这里包含加载的用户系统数据区中的系统配置字节 位 7: nRAM_PRT_CHK 位 6: nSTDBY_WDT 位 5: nDEPSLP_WDT 位 4: 未用 位 3: 未用 位 2: nSTDBY_RST 位 1: nDEPSLP_RST 位 0: nWDT_ATO_EN
位 1	FAP	0	ro	闪存访问保护 (Flash access protection)
位 0	USDERR	0	ro	用户系统数据错误 (User system data error) 该位置起表示用户系统数据中某字节和它的反码不匹配。 此时该字节和它的反码读出值将被硬件自动强制置为 0xFF

5.9.8 擦除编程保护状态寄存器 (FLASH_EPPS)

域	简称	复位值	类型	功能
位 31:0	EPPS	0xFFFF FFFF	ro	擦除/编程保护状态 (Erase/Program protection status) 该寄存器反映的是加载的用户系统数据中的擦写保护字节状态。

5.9.9 闪存安全库区状态寄存器0 (SLIB_STS0)

专用于闪存安全库区。

域	简称	复位值	类型	功能
位 31:24	保留	0x00	resd	保持为默认值
位 23:16	EM_SLIB_INST_SS	0xFF	ro	主存扩展存储区安全库区指令起始扇区 (Extension memory sLib instruction start sector) 00000000: 扇区 0 00000001: 扇区 1 00000010: 扇区 2 ... 00001100: 扇区 12 11111111: 无指令安全区 其余设定值: 无效
位 15:4	保留	0x000	resd	保持为默认值
位 3	SLIB_ENF	0	ro	sLib 使能标志 (sLib enabled flag) 该位置起时, 表示闪存主存区域部分或是全部 (依照 SLIB_STS1 设定) 作为安全库代码。
位 2	EM_SLIB_ENF	0	ro	主存扩展存储区 sLib 使能标志 (Extension memory sLib enabled flag)

				该位置起时，表示启动程序代码区域是作为主闪存扩展区域（BTM_AP_ENF 置起），并且存放的应用代码为安全库代码
位 1	保留	0	resd	保持为默认值
位 0	BTM_AP_ENF	0	ro	启动程序代码区域存放应用代码使能标志（Boot memory store application code enabled flag） 该位置起时，表示启动程序代码区域可以作为主存扩展区域存放用户应用代码；否则仅用于存放系统启动代码

5.9.10 闪存安全库区状态寄存器1（SLIB_STS1）

专用于闪存安全库区。

域	简称	复位值	类型	功能
位 31:22	SLIB_ES	0x3FF	ro	主存安全库区结束扇区（sLib end sector） 0000000000: 扇区 0 0000000001: 扇区 1 0000000010: 扇区 2 ... 0001111111: 扇区 127 (256KB 主闪存存储器的最后扇区) ...0011111111: 扇区 255 (512KB 主闪存存储器的最后扇区)
位 21:11	SLIB_INST_SS	0x7FF	ro	主存安全库区指令区起始扇区（sLib instruction start sector） 0000000000: 扇区 0 0000000001: 扇区 1 0000000010: 扇区 2 ... 0001111111: 扇区 127 (256KB 主闪存存储器的最后扇区) ... 0011111111: 扇区 255 (512KB 主闪存存储器的最后扇区) 1111111111: 无安全库区指令区
位 10:0	SLIB_SS	0x7FF	ro	主存安全库区起始扇区（sLib start sector） 0000000000: 扇区 0 0000000001: 扇区 1 0000000010: 扇区 2 ... 0001111111: 扇区 127 (256KB 主闪存存储器的最后扇区) ... 0011111111: 扇区 255 (512KB 主闪存存储器的最后扇区)

5.9.11 闪存安全库区密码清除寄存器（SLIB_PWD_CLR）

专用于闪存安全库区。

域	简称	复位值	类型	功能
位 31:0	SLIB_PCLR_VAL	0x0000 0000	wo	安全库区密码清除（sLib password clear value） 用于写入正确的安全库区密码，将实现解除安全库区功能。 此寄存器写入状态将在闪存安全库区额外状态寄存器（SLIB_MISC_STS）位 0 与位 1 中体现。

5.9.12 闪存安全库区额外状态寄存器（SLIB_MISC_STS）

专用于闪存安全库区。

域	简称	复位值	类型	功能
位 31:3	保留	0x00000000	resd	保持为默认值
位 2	SLIB_ULKF	0	ro	SLib 解锁标志（sLib unlock flag） 当该位置起时表示 SLib 相关设定寄存器允许配置。
位 1	SLIB_PWD_OK	0	ro	密码正确（sLib password ok） 当密码正确，该位被硬件置起。
				密码错误（sLib password error） 当密码错误，并且设定的密码清除寄存器的值不等于 0xFFFF_FFFF，该位被硬件置起。 注意：当该位置起后，硬件将不再接受重新设定密码清除寄存器，直到再次复位。
位 0	SLIB_PWD_ERR	0	ro	

5.9.13 闪存CRC校验地址寄存器（FLASH_CRC_ADDR）

专用于主闪存以及主存扩展区域。

域	简称	复位值	类型	功能
位 31:0	CRC_ADDR	0x0000 0000	wo	CRC 地址（CRC address） 选择要校验的闪存扇区起始地址。 注意：必须与扇区起始地址对齐

注意：所有这些位是只写的，读出无反应。

5.9.14 闪存CRC校验控制寄存器（FLASH_CRC_CTRL）

专用于主闪存以及主存扩展区域。

域	简称	复位值	类型	功能
位 31:17	保留	0x0000	resd	保持为默认值
位 16	CRC_STRT	0	wo	启动 CRC 校验（CRC start） 设置该位去启动用户代码或是安全库代码的 CRC 校验功能。 硬件启动 CRC 后，会自动清除该位。 注意： 校验数据从 CRC_ADDR ~ CRC_ADDR+CRC_SN*1 扇区
位 15:0	CRC_SN	0x0000	wo	CRC 校验扇区数量（CRC sector number） 设定本次 CRC 校验的数据量，单位是扇区

5.9.15 闪存CRC校验结果寄存器（FLASH_CRC_CHK）

专用于主闪存以及主存扩展区域。

域	简称	复位值	类型	功能
位 31:0	CRC_CHK	0x0000 0000	ro	CRC 校验结果（CRC check result）

注意：所有这些位是只读的，写入无反应。

5.9.16 闪存安全库区密码设定寄存器（SLIB_SET_PWD）

专用于闪存安全库区密码设定。

域	简称	复位值	类型	功能
位 31:0	SLIB_PSET_VAL	0x0000 0000	wo	安全库区密码（sLib password setting value） 注意：在解除安全库区锁定后，此寄存器才允许被写入，用于设定安全库区启动密码。但写入 0xFFFF_FFFF 以及 0x0000_0000 值无效。

注意：所有这些位是只写入，读出为 0。

5.9.17 闪存安全库区地址设定寄存器 (SLIB_SET_RANGE)

专用于主存安全库区地址设定。

域	简称	复位值	类型	功能
位 31:22	SLIB_ES_SET	0x000	wo	主存安全库区结束扇区设定 (sLib end sector setting) 用于设定启动安全库区时的安全库区结束扇区位置 0000000000: 扇区 0 0000000001: 扇区 1 0000000010: 扇区 2 ... 0001111111: 扇区 127 (256KB 主闪存存储器的最后扇区) ... 0011111111: 扇区 255 (512KB 主闪存存储器的最后扇区)
位 21:11	SLIB_ISS_SET	0x000	wo	主存安全库区指令区起始扇区设定 (sLib instruction start sector setting) 用于设定启动安全库区时的指令区起始扇区位置 0000000000: 扇区 0 0000000001: 扇区 1 0000000010: 扇区 2 ... 0001111111: 扇区 127 (256KB 主闪存存储器的最后扇区) ... 0011111111: 扇区 255 (512KB 主闪存存储器的最后扇区) 1111111111: 无安全库区指令区
位 10:0	SLIB_SS_SET	0x000	wo	主存安全库区起始扇区设定 (sLib start sector setting) 用于设定启动安全库区时的安全库区起始扇区位置 0000000000: 扇区 0 0000000001: 扇区 1 0000000010: 扇区 2 ... 0001111111: 扇区 127 (256KB 主闪存存储器的最后扇区) ... 0011111111: 扇区 255 (512KB 主闪存存储器的最后扇区)

注意：

所有这些位是只写入，读出为 0。

在解除安全库区锁定后，此寄存器才允许被写入。

超过主闪存存储地址范围均是无效设定。

5.9.18 主存扩展存储区域安全库区设定寄存器 (EM_SLIB_SET)

专用于主存扩展区域。

域	简称	复位值	类型	功能
位 31:24	保留	0x00	resd	保持为默认值
位 23:16	EM_SLIB_ISS_SET	0x000	wo	主存扩展区安全库区指令区起始扇区设定 (Extension memory sLib instruction start sector setting) 用于设定启动安全库区时的指令区起始扇区位置 00000000: 扇区 0 00000001: 扇区 1 00000010: 扇区 2 ... 00001100: 扇区 12 11111111: 无指令安全区 其余设定值: 无效 注意: 当设为 0xFF, 表示主存扩展区从扇区 0 至扇区 3 都为安全库区, 且整个安全库区作为唯读取;
位 15:0	EM_SLIB_SET	0x000	wo	主存扩展存储区 sLib 设定 (Extension memory sLib setting) 写入 0x5AA5 将启动主存扩展区作为存放安全库区代码功能

注意：所有这些位是只写的，读出无反应。

5.9.19 启动程序代码区模式设定寄存器 (BTM_MODE_SET)

专用于启动程序代码区域。

域	简称	复位值	类型	功能
位 31:8	保留	0x0000000	resd	保持为默认值
位 7:0	BTM_MODE_SET	0x00	wo	启动程序代码区模式设定 (Boot memory mode setting) 0xFF: 启动程序代码区域作为系统区域, 存放系统启动代码功能 其他值: 启动程序代码区域作为主存扩展区域, 存放应用代码功能 注意: 此寄存器的设定需要在闪存访问保护未启动下进行

注意：所有这些位是只写的，读出无反应。

5.9.20 闪存安全库区解锁寄存器 (SLIB_UNLOCK)

专用于安全库区寄存器的解锁设定。

域	简称	复位值	类型	功能
位 31:0	SLIB_UKVAL	0x0000 0000	wo	安全库区解锁键值 (sLib unlock key value) 固定键值 0xA35F_6D24, 用于安全库区设定寄存器的解锁。

注意：所有这些位是只写入，读出为 0。

6 通用和复用功能 I/O (GPIO 和 IOMUX)

6.1 简介

AT32F455 支持多达 117 个双向 I/O 引脚，这些引脚分为 8 组，分别为 PA0-PA15、PB0-PB15、PC0-PC15、PD0-PD15、PE0-PE15、PF0-PF15、PG0-PG15、PH0-PH4，每个引脚都可以实现与外部的通讯、控制以及数据采集的功能。

每个引脚都可以软件配置成通用输入、通用输出、复用功能和模拟引脚。当配置成通用输出或复用功能时，可选择推挽或是开漏输出。

每个引脚都有独立的弱上拉/下拉功能。

每个引脚都可以软件配置输出驱动能力。

每个引脚都可以配置为外部中断输入。

每个引脚都支持配置锁定功能。

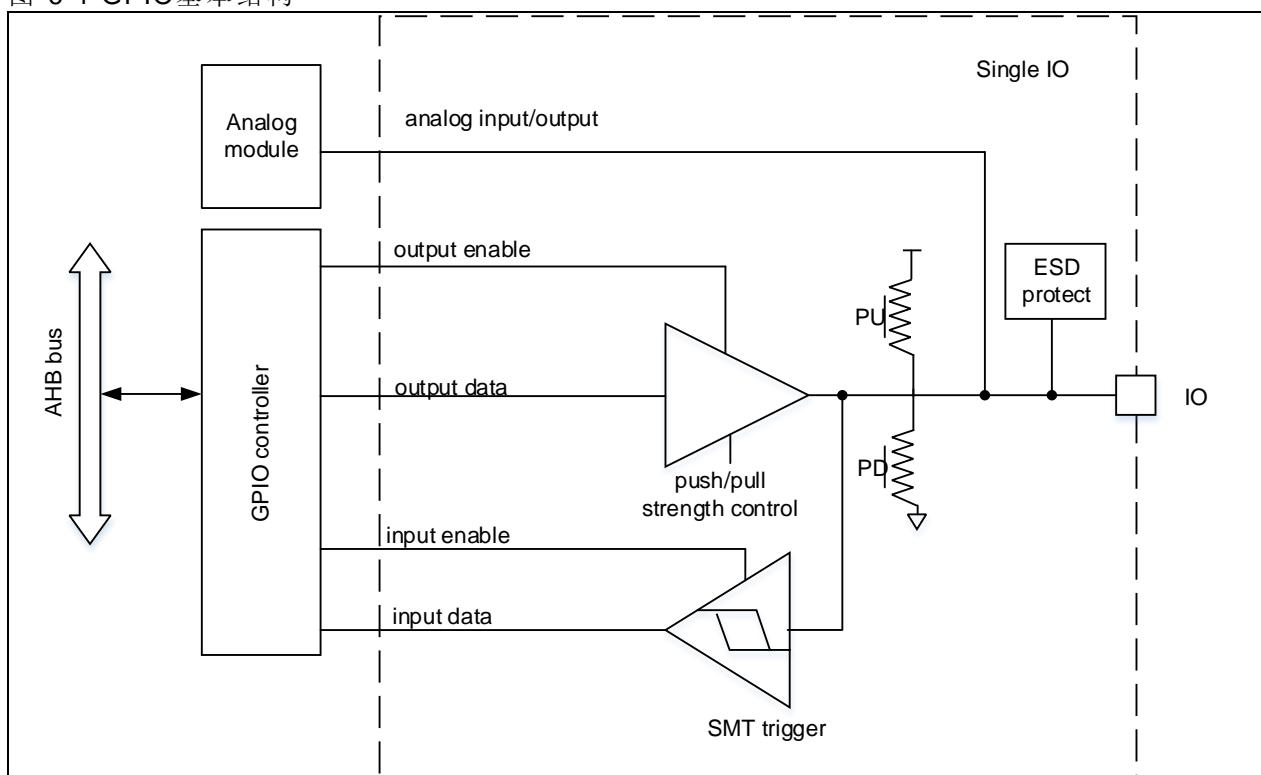
6.2 功能描述

6.2.1 GPIO结构

每个 I/O 端口对应的寄存器允许半字或字节访问，每个 I/O 端口位可以自由编程。

下图给出了一个 I/O 端口位的基本结构。

图 6-1 GPIO 基本结构



6.2.2 GPIO复位状态

系统上电或复位后，所有引脚除了 CPU debug 相关引脚和 BOOT1 引脚以外，都被配置为模拟模式。CPU debug 相关引脚则配置为：PA13/SWDIO 为复用功能上拉模式，PA14/SWCLK 为复用功能下拉模式，PB3/SWO 为复用功能无上拉下拉模式。PB2/BOOT1 为通用浮空输入模式。

6.2.3 通用输入配置

配置模式	IOMC	PULL
通用浮空输入		00 或 11
通用下拉输入	00	10
通用上拉输入		01

当引脚配置为输入时：

- 引脚状态可通过对输入数据寄存器的读访问得到
- 施密特触发器有效

注意：如果是浮空输入模式，为避免复杂环境下，没有使用的引脚有干扰，导致漏电，建议，如引脚不使用，则配置为模拟模式。

6.2.4 模拟配置

配置模式	IOMC	PULL
模拟	11	不使用

当引脚被配置为模拟时：

- 施密特触发器无效
- 输出数据寄存器无效
- 设置上拉/下拉无效
- 引脚状态不可通过输入数据寄存器访问得到

6.2.5 通用输出配置

配置模式	IOMC	OM	HDRV	ODRV[1: 0]	PULL
通用推挽无上拉/下拉	01	0			00 或 11
通用推挽上拉	01	0			01
通用推挽下拉	01	0			10
通用开漏无上拉/下拉	01	1			00 或 11
通用开漏上拉	01	1			01
通用开漏下拉	01	1			10

当引脚被配置为通用输出时：

- 施密特触发器有效
 - 在开漏输出模式时，当输出数据寄存器配置为数字0时，引脚输出数字0；当输出数据寄存器配置为数字1时，无内部上拉和下拉时，引脚为高阻状态，有内部上拉时，引脚上拉到数字1，有内部下拉时，引脚下拉到数字0
 - 在推挽输出模式时，可通过输出数据寄存器输出数字0/1
 - 引脚状态可通过对输入数据寄存器的读访问得到
 - 通过GPIO 翻转/设置/清除寄存器控制对应的GPIO数据输出寄存器的翻转/设置/清除
- 注意：GPIO 设置/清除寄存器 对应同一个引脚的 IOCB/IOSB 同时写1，IOSB 优先级高于 IOCB

6.2.6 GPIO端口保护

为了防止误操作导致 GPIO 功能混乱，提供每个对应引脚的锁定机制。一旦设定写保护，在下次复位或者上电之前都不能进行对应引脚的 GPIO 配置。

6.2.7 IOMUX功能结构

大多数引脚支持多个外设的输入/输出功能映射，通过 IOMUX 功能输入/输出章节查找表来选择每个引脚对应的外设输入输出功能。通过引脚所对应的 GPIOx_MUXL（从引脚 0 到引脚 7）或 GPIOx_MUXH(从引脚 8 到引脚 15) 进行对应的设置,单一引脚有多达 16 种不同的 IOMUX 映射方案，方便灵活选用

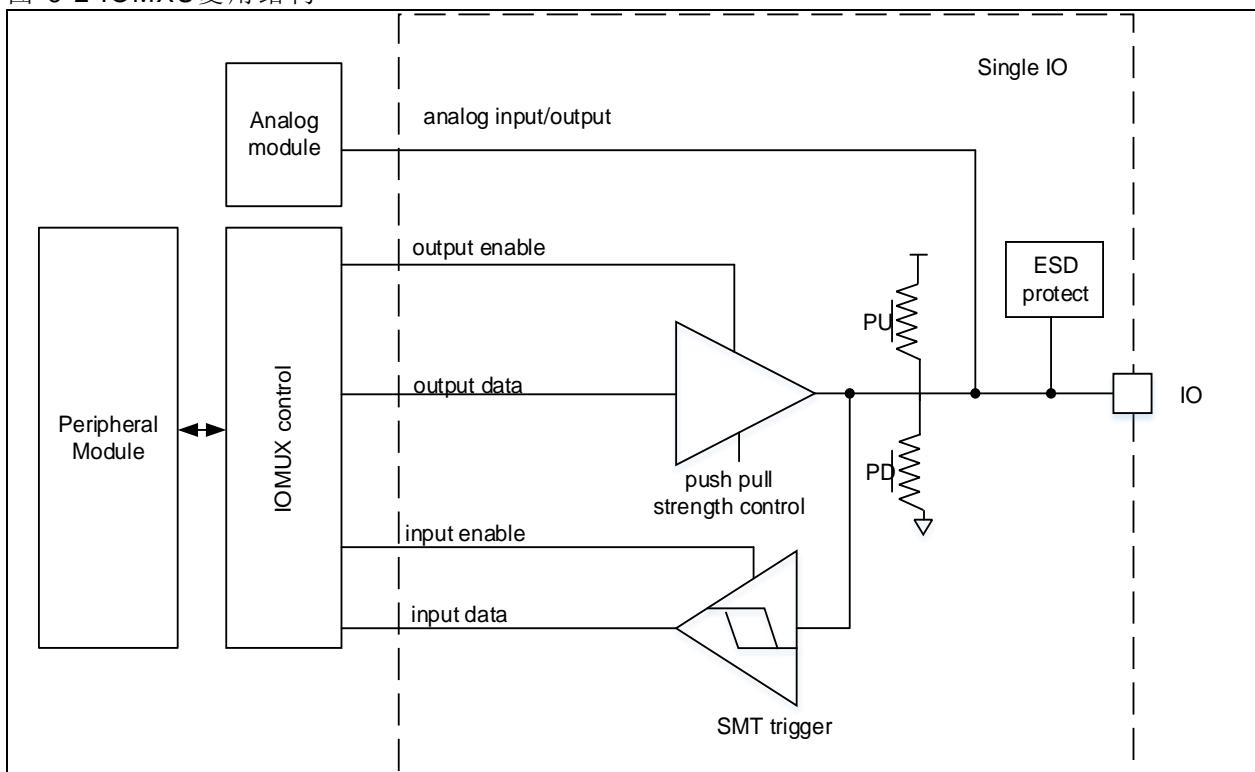
每个引脚通过设定 GPIOx_MUXL 或 GPIOx_MUXH，只会和单一外设的单一脚进行对应，不存在单一引脚多个外设抢占的冲突。

引脚作为复用功能时，引脚和 GPIO 控制器断开，由 IOMUX 控制器进行控制。

引脚作为复用输入时，配置 GPIOx_CFGR 将该引脚设定为复用模式，根据外设特性可配置 GPIOx_PULL 将引脚配置为浮空、上拉或下拉。

引脚作为复用输出或双向复用功能时，配置 GPIOx_CFGR 将该引脚设定为复用模式，根据外设特性可配置 GPIOx_OMODE 将引脚配置为推挽或开漏模式，可配置 GPIOx_ODRVR 和 GPIOx_HDRV 配置引脚的输出驱动能力以及 GPIOx_PULL 配置引脚的上下拉。

图 6-2 IOMUXU 复用结构



6.2.8 复用功能配置

配置模式	IOMC	OM	HDRV	ODRV[1: 0]	PULL
复用推挽无上拉/下拉	10	0			00 或 11
复用推挽上拉	10	0			01
复用推挽下拉	10	0			10
复用开漏无上拉/下拉	10	1	000: 输出模式, 适中电流推动/吸入能力 001: 输出模式, 较大电流推动/吸入能力 010: 输出模式, 适中电流推动/吸入能力 011: 输出模式, 适中电流推动/吸入能力 1xx: 输出模式, 极大电流推动/吸入能力 仅对复用输出或双向复用功能起作用	00 或 11	
复用开漏上拉	10	1		01	
复用开漏下拉	10	1		10	
复用输入无上拉/下拉	10	x		00 或 11	
复用输入上拉	10	x		01	
复用输入下拉	10	x		10	

当引脚被配置为复用功能时：

- 施密特触发器有效
- 在开漏输出模式时，当外设输出为数字0时，引脚输出数字0；当外设输出为数字1时，无内部上拉和下拉时，引脚为高阻状态，有内部上拉时，引脚上拉到数字1，有内部下拉时，引脚下拉到数字0
- 在推挽输出模式时，根据外设的输出引脚输出数字0/1
- 引脚状态可通过对输入数据寄存器的读访问得到

6.2.9 IOMUX功能输入/输出

选择每个端口线的有效复用功能是通过 GPIOx_MUXL (从引脚 0 到引脚 7) 或 GPIOx_MUXH(从引脚 8 到引脚 15) 进行设置。

表 6-1 通过 GPIOA_MUX* 寄存器配置端口 A 的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PA0		TMR2_CH1 TMR2_EXT	TMR5_CH1	TMR8_EXT	I2C2_SCL		USART2_RX	USART2_CTS
PA1		TMR2_CH2	TMR5_CH2	TMR9_CH1C	I2C2_SDA	SPI4_MOSI I2S4_SD	SPI3_CS I2S3_WS	USART2_RTS_DE
PA2		TMR2_CH3	TMR5_CH3	TMR9_CH1		I2SF5_CKIN		USART2_TX
PA3		TMR2_CH4	TMR5_CH4	TMR9_CH2		I2S2_MCK		USART2_RX
PA4					I2C1_SCL	SPI1_CS I2S1_WS	SPI3_CS I2S3_WS	USART2_CK
PA5		TMR2_CH1 TMR2_EXT		TMR8_CH1C		SPI1_SCK I2S1_CK	USART3_CK	USART3_RX
PA6		TMR1_BRK	TMR3_CH1	TMR8_BRK		SPI1_MISO	I2S2_MCK	USART3_CTS
PA7		TMR1_CH1C	TMR3_CH2	TMR8_CH1C	I2C3_SCL	SPI1_MOSI I2S1_SD		USART3_TX
PA8	CLKOUT	TMR1_CH1		TMR9_BRK	I2C3_SCL			USART1_CK
PA9	CLKOUT	TMR1_CH2			I2C3_SMBA	SPI2_SCK I2S2_CK		USART1_TX
PA10	ERTC_REFIN	TMR1_CH3				SPI2_MOSI I2S2_SD	I2S4_MCK	USART1_RX
PA11		TMR1_CH4			I2C2_SCL	SPI2_CS I2S2_WS	SPI4_MISO	USART1_CTS
PA12		TMR1_EXT			I2C2_SDA	SPI2_MISO	I2SF5_SDEXT	USART1_RTS_DE
PA13	SWDIO	IR_OUT			I2C1_SDA		SPI3_MISO	
PA14	SWCLK				I2C1_SMBA		SPI3_MOSI I2S3_SD	
PA15		TMR2_CH1 TMR2_EXT				SPI1_CS I2S1_WS	SPI3_CS I2S3_WS	USART1_TX

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PA0	USART4_TX	TMR9_CH2C		EMAC_MII_CRS				EVENTOUT
PA1	USART4_RX	QSPI1_IO3		EMAC_MII_RX_CLK EMAC_RMII_REF_CLK		I2SF5_SD	I2C1_SMBA	EVENTOUT
PA2		CAN2_RX	QSPI1_CS	EMAC_MDIO			XMC_D4	EVENTOUT
PA3		CAN2_TX		EMAC_MII_COL			XMC_D5	EVENTOUT
PA4	USART6_TX	TMR14_CH1				OTGFS1_OE	XMC_D6	EVENTOUT
PA5	USART6_RX	TMR13_CH1C					XMC_D7	EVENTOUT
PA6	USART3_RX	TMR13_CH1	QSPI1_MOSI_IO0		SDIO1_CMD	QSPI1_IO2		EVENTOUT
PA7		TMR14_CH1	QSPI1_MISO_IO1	EMAC_MII_RX_DV EMAC_RMII_CRS_DV	XMC_SDNWE			EVENTOUT
PA8	USART2_TX	USART7_RX	OTGFS1_SOF	CAN3_RX	SDIO1_D1		XMC_A4	EVENTOUT
PA9	I2C1_SCL	TMR14_BRK	OTGFS1_VBUS		SDIO1_D2			EVENTOUT
PA10	I2C1_SDA		OTGFS1_ID			I2SF5_MCK	I2SF5_SD	EVENTOUT
PA11	USART6_TX	CAN1_RX				USART4_RX	I2C1_SMBA	EVENTOUT
PA12	USART6_RX	CAN1_TX				USART4_TX		EVENTOUT
PA13			OTGFS1_OE					EVENTOUT
PA14	USART2_TX							EVENTOUT
PA15	USART2_RX	USART7_TX	QSPI1_IO2	CAN3_TX	XMC_NE2	USART4_RTS_DE		EVENTOUT

表 6-2 通过GPIOB_MUX*寄存器配置端口B的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PB0		TMR1_CH2C	TMR3_CH3	TMR8_CH2C		SPI1_MISO	USART2_RX	SPI3_MOSI I2S3_SD
PB1		TMR1_CH3C	TMR3_CH4	TMR8_CH3C		SPI1_MOSI I2S1_SD	SPI2_SCK I2S2_CK	USART2_CK
PB2		TMR2_CH4	TMR3_EXT		I2C3_SMBA			SPI3_MOSI I2S3_SD
PB3	SWO	TMR2_CH2			I2C2_SDA	SPI1_SCK I2S1_CK	SPI3_SCK I2S3_CK	USART1_RX
PB4		TMR1_CH4C	TMR3_CH1	TMR11_BRK	I2C3_SDA	SPI1_MISO	SPI3_MISO	I2S3_SDEXT
PB5			TMR3_CH2	TMR10_BRK	I2C1_SMBA	SPI1_MOSI I2S1_SD	SPI3_MOSI I2S3_SD	USART1_CK
PB6			TMR4_CH1	TMR10_CH1C	I2C1_SCL	I2S1_MCK	SPI4_CS I2S4_WS	USART1_TX
PB7			TMR4_CH2	TMR8_BRK	I2C1_SDA		SPI4_SCK I2S4_CK	USART1_RX
PB8		TMR2_CH1 TMR2_EXT	TMR4_CH3	TMR10_CH1	I2C1_SCL		SPI4_MISO	USART1_TX
PB9	IR_OUT	TMR2_CH2	TMR4_CH4	TMR11_CH1	I2C1_SDA	SPI2_CS I2S2_WS	SPI4_MOSI I2S4_SD	I2C2_SDA
PB10		TMR2_CH3			I2C2_SCL	SPI2_SCK I2S2_CK	I2S3_MCK	USART3_TX
PB11		TMR2_CH4	TMR5_CH4		I2C2_SDA	I2SF5_CKIN		USART3_RX
PB12		TMR1_BRK	TMR5_CH1	TMR12_BRK	I2C2_SMBA	SPI2_CS I2S2_WS	SPI4_CS I2S4_WS	SPI3_SCK I2S3_CK
PB13	CLKOUT	TMR1_CH1C		TMR12_CH1C	I2C3_SMBA	SPI2_SCK I2S2_CK	SPI4_SCK I2S4_CK	I2C3_SCL
PB14		TMR1_CH2C		TMR8_CH2C	I2C3_SDA	SPI2_MISO	I2S2_SDEXT	USART3_RTS_DE
PB15	ERTC_REFIN	TMR1_CH3C		TMR8_CH3C	I2C3_SCL	SPI2_MOSI I2S2_SD		

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PB0	USART3_CK		QSPI1_MOSI_IO0	EMAC_MII_RXD2	SDIO1_D1	I2SF5_CK		EVENTOUT
PB1	USART3_RTS_DE	QSPI1_SCK		EMAC_MII_RXD3	SDIO1_D2	I2SF5_WS	TMR14_CH1	EVENTOUT
PB2		QSPI1_SCK	CAN3_STB		SDIO1_CK		TMR14_CH1C	EVENTOUT
PB3	USART1_RTS_DE	USART7_RX	QSPI1_IO3	CAN3_RX		USART5_TX		EVENTOUT
PB4	USART1_CTS	USART7_TX	QSPI1_SCK	CAN3_TX	SDIO1_D0	USART5_RX		EVENTOUT
PB5	USART5_RX	CAN2_RX	QSPI1_MOSI_IO0	EMAC_PPS_OUT	XMC_SDCKE1	USART5_CK_DE	SDIO1_D3	EVENTOUT
PB6	USART5_TX	CAN2_TX	QSPI1_CS	USART4_CK	XMC_SDNE1	I2SF5_WS	SDIO1_D0	EVENTOUT
PB7	USART4_CTS	TMR11_CH1C	QSPI1_MISO_IO1	CAN1_STB	XMC_NADV	I2SF5_CK	SDIO1_D0	EVENTOUT
PB8	USART5_RX	CAN1_RX		EMAC_MII_TXD3	SDIO1_D4	I2SF5_SDEXT	I2SF5_SD	EVENTOUT
PB9	USART5_TX	CAN1_TX	QSPI1_CS		SDIO1_D5	I2SF5_SD	I2S1_MCK	EVENTOUT
PB10		QSPI1_CS	QSPI1_MISO_IO1	EMAC_MII_RX_ER	SDIO1_D7		XMC_NOE	EVENTOUT
PB11		TMR13_BRK	QSPI1_MOSI_IO0	EMAC_MII_TX_EN EMAC_RMII_TX_EN		CAN2_STB	XMC_LB	EVENTOUT
PB12	USART3_CK	CAN2_RX		EMAC_MII_TXD0 EMAC_RMII_TXD0	USART5_RX	I2SF5_WS	XMC_D13	EVENTOUT
PB13	USART3_CTS	CAN2_TX		EMAC_MII_TXD1 EMAC_RMII_TXD1	USART5_TX	I2SF5_CK	XMC_UB	EVENTOUT
PB14		TMR12_CH1				SDIO1_D6	XMC_D0	EVENTOUT
PB15		TMR12_CH2				SDIO1_CK	TMR12_CH1C	EVENTOUT

表 6-3 通过GPIOC_MUX*寄存器配置端口C的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PC0					I2C3_SCL		I2C1_SCL	
PC1					I2C3_SDA	SPI3_MOSI I2S3_SD	I2C1_SDA	SPI2_MOSI I2S2_SD
PC2						SPI2_MISO	I2S2_SDEXT	
PC3						SPI2_MOSI I2S2_SD		
PC4				TMR9_CH1		I2S1_MCK		USART3_TX
PC5		TMR1_CH4C		TMR9_CH2	I2C1_SMBA			USART3_RX
PC6		TMR1_CH1	TMR3_CH1	TMR8_CH1	I2C1_SCL	I2S2_MCK		
PC7		TMR1_CH2	TMR3_CH2	TMR8_CH2	I2C1_SDA	SPI2_SCK I2S2_CK	I2S3_MCK	
PC8		TMR1_CH3	TMR3_CH3	TMR8_CH3		I2S4_MCK	I2SF5_MCK	USART8_TX
PC9	CLKOUT	TMR1_CH4	TMR3_CH4	TMR8_CH4	I2C3_SDA	I2SF5_CKIN		USART8_RX
PC10			TMR5_CH2				SPI3_SCK I2S3_CK	USART3_TX
PC11			TMR5_CH3			I2S3_SDEXT	SPI3_MISO	USART3_RX
PC12				TMR11_CH1	I2C2_SDA		SPI3_MOSI I2S3_SD	USART3_CK
PC13				TMR8_CH4C				
PC14								
PC15								

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PC0	USART6_TX	USART7_TX			XMC_SDNWE			EVENTOUT
PC1	USART6_RX	USART7_RX		EMAC_MDC				EVENTOUT
PC2	USART8_TX			EMAC_MII_RXD2	XMC_SDNE0		XMC_NWE	EVENTOUT
PC3	USART8_RX			EMAC_MII_TX_CLK	XMC_SDCKE0		XMC_A0	EVENTOUT
PC4		TMR13_CH1	QSPI1_IO2	EMAC_MII_RXD0 EMAC_RMII_RXD0	XMC_SDNE0		XMC_NE4	EVENTOUT
PC5		TMR13_CH1C	QSPI1_IO3	EMAC_MII_RXD1 EMAC_RMII_RXD1	XMC_SDCKE0		XMC_NOE	EVENTOUT
PC6	USART6_TX	USART7_TX	XMC_A0		SDIO1_D6		XMC_D1	EVENTOUT
PC7	USART6_RX	USART7_RX	XMC_A1		SDIO1_D7		XMC_NADV	EVENTOUT
PC8	USART6_CK	QSPI1_IO2	XMC_A2		SDIO1_D0			EVENTOUT
PC9	I2C1_SDA	QSPI1_MOSI_IO0	XMC_A3	OTGFS1_OE	SDIO1_D1			EVENTOUT
PC10	USART4_TX	QSPI1_MISO_IO1			SDIO1_D2			EVENTOUT
PC11	USART4_RX	QSPI1_CS			SDIO1_D3		XMC_D2	EVENTOUT
PC12	USART4_CK	USART5_TX			SDIO1_CK		XMC_D3	EVENTOUT
PC13								EVENTOUT
PC14								EVENTOUT
PC15								EVENTOUT

表 6-4 通过GPIOD_MUX*寄存器配置端口D的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PD0				TMR8_CH4C		SPI4_MISO	SPI3_MOSI I2S3_SD	SPI2_CS I2S2_WS
PD1				TMR8_CH4			SPI2_SCK I2S2_CK	SPI2_CS I2S2_WS
PD2			TMR3_EXT					USART3_RTS_DE
PD3						SPI2_SCK I2S2_CK	SPI2_MISO	USART2_CTS
PD4							SPI2_MOSI I2S2_SD	USART2_RTS_DE
PD5								USART2_TX
PD6						SPI3_MOSI I2S3_SD		USART2_RX
PD7								USART2_CK
PD8								USART3_TX
PD9								USART3_RX
PD10								USART3_CK
PD11					I2C2_SMBA			USART3_CTS
PD12			TMR4_CH1		I2C2_SCL			USART3_RTS_DE
PD13			TMR4_CH2		I2C2_SDA			
PD14			TMR4_CH3		I2C3_SCL			
PD15			TMR4_CH4		I2C3_SDA			

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PD0	USART4_RX	CAN1_RX	XMC_A5		XMC_D2			EVENTOUT
PD1	USART4_TX	CAN1_TX	XMC_A6		XMC_D3			EVENTOUT
PD2	USART5_RX		XMC_A7		SDIO1_CMD		XMC_NWE	EVENTOUT
PD3		QSPI1_SCK	XMC_A8		XMC_CLK			EVENTOUT
PD4			XMC_A9		XMC_NOE			EVENTOUT
PD5			XMC_A10		XMC_NWE			EVENTOUT
e			XMC_A11		XMC_NWAIT			EVENTOUT
PD7			XMC_A12		XMC_NE1			EVENTOUT
PD8		TMR12_CH2C		EMAC_MII_RX_DV EMAC_RMII_CRS_DV	XMC_D13			EVENTOUT
PD9				EMAC_MII_RXD0 EMAC_RMII_RXD0	XMC_D14			EVENTOUT
PD10	USART4_TX			EMAC_MII_RXD1 EMAC_RMII_RXD1	XMC_D15			EVENTOUT
PD11		QSPI1_MOSI_IO0	XMC_A14 XMC_SDBA0	EMAC_MII_RXD2	XMC_A16	CAN3_STB		EVENTOUT
PD12	USART8_CK_DE	QSPI1_MISO_IO1	XMC_A15 XMC_SDBA1	EMAC_MII_RXD3	XMC_A17	CAN3_RX		EVENTOUT
PD13	USART8_TX	QSPI1_IO3	XMC_SDCLK		XMC_A18	CAN3_TX		EVENTOUT
PD14	USART8_RX				XMC_D0			EVENTOUT
PD15		USART7_CK_DE			XMC_D1			EVENTOUT

表 6-5 通过GPIOE_MUX*寄存器配置端口E的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PE0			TMR4_EXT					
PE1		TMR1_CH2C						
PE2			TMR3_EXT	TMR9_BRK		SPI4_SCK I2S4_CK		
PE3			TMR3_CH1	TMR9_CH2C				
PE4	CLKOUT		TMR3_CH2	TMR9_CH1C		SPI4_CS I2S4_WS		
PE5			TMR3_CH3	TMR9_CH1		SPI4_MISO		
PE6			TMR3_CH4	TMR9_CH2		SPI4_MOSI I2S4_SD		
PE7		TMR1_EXT						
PE8		TMR1_CH1C						
PE9		TMR1_CH1						
PE10		TMR1_CH2C						
PE11		TMR1_CH2				SPI4_CS I2S4_WS		
PE12		TMR1_CH3C			SPI1_CS I2S1_WS	SPI4_SCK I2S4_CK		
PE13		TMR1_CH3			SPI1_SCK I2S1_CK	SPI4_MISO		
PE14		TMR1_CH4			SPI1_MISO	SPI4_MOSI I2S4_SD		
PE15		TMR1_BRK	TMR1_CH4C		SPI1_MOSI I2S1_SD			

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PE0	USART8_RX	TMR13_CH1			XMC_LB XMC_SDDQML			EVENTOUT
PE1	USART8_TX	TMR14_CH1			XMC_UB XMC_SDDQMH			EVENTOUT
PE2		QSPI1_IO2	XMC_SDNCAS	EMAC_MII_TXD3	XMC_A23	TMR14_CH1C		EVENTOUT
PE3		TMR14_BRK			XMC_A19			EVENTOUT
PE4					XMC_A20			EVENTOUT
PE5					XMC_A21			EVENTOUT
PE6			XMC_SDNRAS		XMC_A22			EVENTOUT
PE7	USART5_CK_DE	USART7_RX	XMC_A13		XMC_D4			EVENTOUT
PE8	USART4_TX	USART7_TX	XMC_A16		XMC_D5			EVENTOUT
PE9	USART4_RX		XMC_A17		XMC_D6			EVENTOUT
PE10	USART5_TX		XMC_A18		XMC_D7			EVENTOUT
PE11	USART5_RX				XMC_D8			EVENTOUT
PE12					XMC_D9			EVENTOUT
PE13					XMC_D10			EVENTOUT
PE14					XMC_D11			EVENTOUT
PE15					XMC_D12			EVENTOUT

表 6-6 通过GPIOF_MUX*寄存器配置端口F的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PF0					I2C2_SDA			
PF1					I2C2_SCL			
PF2					I2C2_SMBA			
PF3								
PF4								
PF5								
PF6				TMR10_CH1				
PF7				TMR11_CH1				
PF8								
PF9								
PF10		TMR1_EXT	TMR5_CH4					
PF11				TMR8_EXT				
PF12				TMR8_BRK				
PF13					I2C3_SMBA			
PF14					I2C3_SCL			
PF15					I2C3_SDA			

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PF0					XMC_A0			EVENTOUT
PF1					XMC_A1			EVENTOUT
PF2					XMC_A2			EVENTOUT
PF3					XMC_A3			EVENTOUT
PF4					XMC_A4			EVENTOUT
PF5			CAN3_STB		XMC_A5			EVENTOUT
PF6	USART7_RX	QSPI1_IO3		CAN3_RX				EVENTOUT
PF7	USART7_TX	QSPI1_IO2		CAN3_TX				EVENTOUT
PF8	USART8_RX	TMR13_CH1	QSPI1_MOSI_IO0					EVENTOUT
PF9	USART8_TX	TMR14_CH1	QSPI1_MISO_IO1					EVENTOUT
PF10		QSPI1_SCK						EVENTOUT
PF11					XMC_SDNRAS			EVENTOUT
PF12					XMC_A6			EVENTOUT
PF13					XMC_A7			EVENTOUT
PF14					XMC_A8			EVENTOUT
PF15			CAN1_STB		XMC_A9			EVENTOUT

表 6-7 通过GPIOG_MUX*寄存器配置端口G的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PG0						SPI1_MISO		
PG1						SPI1_MOSI I2S1_SD		
PG2								
PG3								
PG4								
PG5								
PG6								
PG7								
PG8								
PG9								
PG10								
PG11							SPI4_SCK I2S4_CK	
PG12							SPI4_MISO	
PG13							SPI4_MOSI I2S4_SD	
PG14							SPI4_CS I2S4_WS	
PG15								

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PG0		CAN1_RX			XMC_A10			EVENTOUT
PG1		CAN1_TX			XMC_A11			EVENTOUT
PG2					XMC_A12			EVENTOUT
PG3					XMC_A13			EVENTOUT
PG4					XMC_A14 XMC_SDBA0			EVENTOUT
PG5					XMC_A15 XMC_SDBA1			EVENTOUT
PG6			QSPI1_CS					EVENTOUT
PG7	USART6_CK							EVENTOUT
PG8	USART6_RTS_DE			EMAC_PPS_OUT	XMC_SDCLK			EVENTOUT
PG9	USART6_RX	QSPI1_IO2		CAN3_TX	XMC_NE2			EVENTOUT
PG10				CAN3_RX	XMC_NE3			EVENTOUT
PG11		CAN2_RX		EMAC_MII_TX_EN EMAC_RMII_TX_EN				EVENTOUT
PG12	USART6_RTS_DE	CAN2_TX			XMC_NE4			EVENTOUT
PG13	USART6_CTS		CAN2_STB	EMAC_MII_TXD0 EMAC_RMII_TXD0	XMC_A24			EVENTOUT
PG14	USART6_TX	QSPI1_IO3		EMAC_MII_TXD1 EMAC_RMII_TXD1	XMC_A25			EVENTOUT
PG15	USART6_CTS				XMC_SDNCAS			EVENTOUT

表 6-8 通过GPIOH_MUX*寄存器配置端口H的复用功能

引脚名	MUX0	MUX1	MUX2	MUX3	MUX4	MUX5	MUX6	MUX7
PH0		TMR1_CH1			I2C1_SDA			
PH1		TMR1_CH2C			I2C1_SCL	SPI2_CS I2S2_WS		
PH2		TMR2_CH1	TMR5_CH1		I2C2_SCL			
PH3		TMR2_CH2	TMR5_CH2		I2C2_SDA			
PH4						SPI2_SCK I2S2_CK		

引脚名	MUX8	MUX9	MUX10	MUX11	MUX12	MUX13	MUX14	MUX15
PH0								EVENTOUT
PH1								EVENTOUT
PH2	USART4_RX	USART7_RX	QSPI1_MOSI_IO0					EVENTOUT
PH3	USART4_TX	USART7_TX	QSPI1_MISO_IO1					EVENTOUT
PH4		USART7_CK_DE						EVENTOUT

注意：EVENTOUT 是 Cortex-M 的 EVENTOUT 信号

6.2.10 外设复用功能引脚配置

当外设需要使用 IOMUX 复用功能时：

- 如果外设引脚需要作为复用输出则对应的引脚配置成复用推挽/开漏
- 如果外设引脚需要作为复用输入则对应的引脚配置成复用输入（浮空/上拉/下拉）
- ADC 外设需要将模拟通道对应的引脚配置为模拟模式
- I²C 外设需要对应引脚作为双向复用功能时，需把对应的引脚配置复用开漏模式
- USB 的 OTGFS1_ID 引脚只需配置 IOMUX 对应的复用，以及 CRM 对应的时钟使能就可以，不需要配置 GPIO 状态。

6.2.11 IOMUX 映射优先级

除了某些引脚可能会被硬件直接抢占，其他外设都可通过配置 GPIOx_MUXL/GPIOx_MUXH 寄存器得到唯一外设复用。

某些引脚不管 GPIO 配置为任何模式，都会被特定的硬件功能直接占用。

表 6-9 硬件抢占功能

引脚名字	抢占使能位	说明
PA0	PWC_CTRL[16] = 1	抢占使能位有效之后，PA0 引脚直接作为 PWC 的 WKUP1 功能使用
PC13	PWC_CTRL[17] = 1	抢占使能位有效之后，PC13 引脚直接作为 PWC 的 WKUP2 功能使用
PA2	PWC_CTRL[19] = 1	抢占使能位有效之后，PA2 引脚直接作为 PWC 的 WKUP4 功能使用
PC5	PWC_CTRL[20] = 1	抢占使能位有效之后，PC5 引脚直接作为 PWC 的 WKUP5 功能使用
PB5	PWC_CTRL[21] = 1	抢占使能位有效之后，PB5 引脚直接作为 PWC 的 WKUP6 功能使用
PB15	PWC_CTRL[22] = 1	抢占使能位有效之后，PB15 引脚直接作为 PWC 的 WKUP7 功能使用
	(ERTC_CTRL[23]=1) (ERTC_CTRL[22: 21]!=00)	
PC13	(ERTC_CTRL[11]=1& ERTC_TAMP[17]=0) (ERTC_TAMP[0]=1& ERTC_TAMP[16]=0)	抢占使能位有效之后，PC13 作为 RTC 通道使用
	(ERTC_CTRL[11]=1& ERTC_TAMP[17]=1) (ERTC_TAMP[0]=1& ERTC_TAMP[16]=1) (ERTC_TAMP[3]=1)	
PA0	(ERTC_TAMP[0]=1& ERTC_TAMP[16]=1) (CRM_BPDC[0]=1)	抢占使能位有效之后，PA0 引脚直接作为 TAMPER2_BPR 功能使用
PC14	CRM_BPDC[0]=1	抢占使能位有效之后，PC14 作为 LEXT 通道使用
PC15	CRM_BPDC[0]=1 & CRM_BPDC[2]=0	抢占使能位有效之后，PC15 作为 LEXT 通道使用
PA4	DAC_CTRL[2] = 1	抢占使能位有效之后，PA4 作为 DAC1 模拟通道使用
PA5	DAC_CTRL[18] = 1	抢占使能位有效之后，PA5 作为 DAC2 模拟通道使用
PH0	CRM_CTRL[16]=1	抢占使能位有效之后，PH0 作为 HEXT 通道使用
PH1	CRM_CTRL[16]=1& CRM_CTRL[18]=0	抢占使能位有效之后，PH1 作为 HEXT 通道使用
PA10	CRM_AHBEN2[7]=1 & GPIOA_CFG[21:20]=10 & GPIOA_MUXH[11:8]=1010	抢占使能位有效之后，PA10 自动配置为复用输入上拉模式， 给 OTGFS1_ID 使用
PA11	CRM_AHBEN2[7] & OTGFS_GCCFG[16]	抢占使能位有效之后，PA11 作为 OTGFS_D-通道使用
PA12	CRM_AHBEN2[7] & OTGFS_GCCFG[16]	抢占使能位有效之后，PA12 作为 OTGFS_D+通道使用

注意： PA0 或者 PC13 不能同时使能 TAMPER_BPR 功能和 PWC 的 WKUP 功能。

6.2.12 外部中断/唤醒线

每个引脚都支持作为外部中断的输入，对应的引脚须配置为输入模式。

6.3 GPIO 寄存器

下面列出了 GPIO 寄存器映像和复位数值。

可以用字节（8 位）、半字（16 位）或字（32 位）的方式操作这些外设寄存器。

表 6-10 GPIO寄存器地址映像和复位值

寄存器简称	基址偏移量	复位值
GPIOA_CFGR	0x00	0xEBFF FFFF
GPIOx_CFGR(x =B,C,D,E,F,G,H)	0x00	0xFFFF FF8F(B) 0xFFFF FFFF
GPIOx_OMODER	0x04	0x0000 0000
GPIOx_ODRVR	0x08	0xC00 0000(A) 0x0000 00C0(B) 0x0000 0000
GPIOA_PULL	0x0C	0x2400 0000(A)
GPIOx_PULL(x = B,C,D,E,F,G,H)	0x0C	0x0000 0000
GPIOx_IDT	0x10	0x0000 XXXX
GPIOx_ODT	0x14	0x0000 0000
GPIOx_SCR	0x18	0x0000 0000
GPIOx_WPR	0x1C	0x0000 0000
GPIOx_MUXL	0x20	0x0000 0000
GPIOx_MUXH	0x24	0x0000 0000
GPIOx_CLR	0x28	0x0000 0000
GPIOx_TOGR	0x2C	0x0000 0000
GPIOx_HDRV	0x3C	0x0000 0000

6.3.1 GPIO配置寄存器 (GPIOx_CFGR) (x=A..H)

偏移地址: 0x00

复位值: 0xEBFF_FFFF 端口 A

0xFFFF_FF8F 端口 B

0xFFFF_FFFF 其它端口

域	简称	复位值	类型	功能
位 2y+1: 2y	IOMCy	0xEBFF FFFF rw		GPIOx 模式配置 (y=0~15) (GPIOx mode configure) 用于配置 GPIOx 的工作模式: 00: 输入 01: 通用输出 10: 复用功能 11: 模拟 (复位后的模式)

6.3.2 GPIO输出模式寄存器 (GPIOx_OMODE) (x=A..H)

域	简称	复位值	类型	功能
位 31: 16 保留		0x0000	resd	始终读为 0。
位 15: 0 OM		0x0000	rw	GPIOx 的输出模式配置 (y=0..15) (GPIOx output mode configure) 当 GPIOx 用作输出时, 可选择以下两种输出模式: 0: 推挽 (复位状态) 1: 开漏

6.3.3 GPIO电流推动/吸入能力切换控制寄存器（GPIOx_ODRVR） (x=A..H)

偏移地址: 0x08

复位值: 0x0C00_0000 端口 A

0x0000_00C0 端口 B

0x0000_0000 其它端口

域	简称	复位值	类型	功能
位 2y+1: 2y	ODRVy	0x0000 0000	rw	GPIOx 的驱动能力配置 (y=0...15) (GPIOx drive capability) 用于配置相应的 I/O 端口电流能力: x0: 适中电流推动/吸入能力 01: 较大电流推动/吸入能力 11: 适中电流推动/吸入能力

6.3.4 GPIO上/下拉寄存器（GPIOx_PULL）(x=A..H)

偏移地址: 0x0C

复位值: 0x2400_0000 端口 A

0x0000_0000 其它端口

域	简称	复位值	类型	功能
位 2y+1: 2y	PULLy	0x2400 0000	rw	GPIOx 的上下拉配置 (y=0...15) (GPIOx pull configurate) 用于配置相应的 I/O 上拉或下拉。 00,11: 无作用 01: 上拉 10: 下拉

6.3.5 GPIO输入数据寄存器（GPIOx_IDT）(x=A..H)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	始终读为 0。
位 15: 0	IDT	0xFFFF	ro	GPIOx 输入的数据 (GPIOx input data) GPIOx 对应 IO 口的输入电平状态，每一位对应 GPIOx 的一个 IO。

6.3.6 GPIO输出数据寄存器（GPIOx_ODT）(x=A..H)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	始终读为 0。
位 15: 0	ODT	0x0000	rw	GPIOx 输出的数据 (IO output data)。 每一位对应 GPIOx 的一个 IO。 GPIOx 对应 IO 口的输出电平状态。 0: 低电平； 1: 高电平。

6.3.7 GPIO设置/清除寄存器 (GPIOx_SCR) (x=A..H)

域	简称	复位值	类型	功能
位 31: 16	IOCB	0x0000	wo	清除 GPIOx 位 (GPIOx clear bit) 写'1'的位其对应 ODT 寄存器位会清除, 写'0'的位其对应 ODT 寄存器位维持不变, 相当于 ODT 寄存器的位操作。 0: 对应位不变; 1: 对应位清除。
位 15: 0	IOSB	0x0000	wo	设置 GPIOx 位 (GPIOx set bit) 写'1'的位其对应 ODT 寄存器位会置起, 写'0'的位其对应 ODT 寄存器位维持不变, 相当于 ODT 寄存器的位操作。 如果 IOCB 和 IOSB 同一个位都写'1', 那么优先级更高的 IOSB 会生效。 0: 对应位不变; 1: 对应位置起。

6.3.8 GPIO写保护寄存器 (GPIOx_WPR) (x=A..H)

域	简称	复位值	类型	功能
位 31: 17	保留	0x0000	resd	保持为默认值。
位 16	WPSEQ	0x0	rw	写保护使能序列 (Write protect sequence) 想保护某些 IO 位不被写入, 需配合同时操作写保护使能序列位和 WPEN 位。 写保护使能位操作按照以下方式操作 4 次, 写'1' ->写'0' ->写'1' ->读, 操作期间 WPSEL 位值不可修改。
位 15: 0	WPEN	0x0000	rw	写保护使能 (Write protect enable) 每一位对应 GPIOx 的一个 IO。 0: 无写保护; 1: 写保护。

6.3.9 GPIO复用低位寄存器 (GPIOx_MUXL) (x=A..H)

偏移地址: 0x20

复位值: 0x00000000

域	简称	复位值	类型	功能
位 4y+3: 4y	MUXLy	0x0	rw	GPIOx 引脚 y 的复用功能选择 (y=0...7) (GPIOx pin y muxing) 用于配置对应 IO 口的复用功能。 0000: MUX0 0001: MUX1 0010: MUX2 0011: MUX3 0100: MUX4 0101: MUX5 0110: MUX6 0111: MUX7 1000: MUX8 1001: MUX9 1010: MUX10 1011: MUX11 1100: MUX12 1101: MUX13 1110: MUX14 1111: MUX15

6.3.10 GPIO复用高位寄存器（GPIOx_MUXH）（x=A..H）

域	简称	复位值	类型	功能
位 4(y-8)+3 : 4(y-8)	MUXHy	0x0	rw	端口 x 引脚 y 的复用功能选择 (y=8...15) (GPIOx pin y muxing) 用于配置对应 IO 口的复用功能。 0000: MUX0 0001: MUX1 0010: MUX2 0011: MUX3 0100: MUX4 0101: MUX5 0110: MUX6 0111: MUX7 1000: MUX8 1001: MUX9 1010: MUX10 1011: MUX11 1100: MUX12 1101: MUX13 1110: MUX14 1111: MUX15

6.3.11 GPIO位清除寄存器（GPIOx_CLR）（x=A..H）

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	rstd	保持为默认值。
位 15: 0	IOCB	0x0000	wo	清除 GPIOx 的位 (GPIOx clear bit) 写'1'的位其对应 ODT 寄存器位会清除，写'0'的位其对应 ODT 寄存器位维持不变，相当于 ODT 寄存器的位操作。 0: 对应位不变； 1: 对应位清除。

6.3.12 GPIO位翻转寄存器（GPIOx_TOGR）（x=A..H）

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	rstd	保持为默认值。
位 15: 0	IOTB	0x0000	wo	翻转端口 x 的位 y (y=0...15) 这些位只能写入。读这些位时返回 0x0000 数值。 0: 相应 ODTy 位没有改变 1: 翻转相应的 ODTy

6.3.13 极大电流推动/吸入能力切换控制寄存器（GPIOx_HDRV）（x=A..H）

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	rstd	保持为默认值。
位 15: 0	HDRV	0x0000	rw	极大电流推动/吸入能力切换控制寄存器 0: 无效 1: GPIO 切换为极大电流推动/吸入能力

7 系统配置控制器 (SCFG)

7.1 简介

该器件具有一组配置寄存器。系统配置控制器的主要用途如下：

- 管理连接到 GPIO 口的外部中断
- 管理存储器映射方式
- 管理部分 IRTMR GPIO 配置

7.2 SCFG 寄存器

下面列出了 SCFG 寄存器映像和复位数值。

必须以字（32 位）的方式操作这些外设寄存器。

表 7-1 SCFG 寄存器地址映像和复位值

寄存器简称	基址偏移量	复位值
SCFG_CFG1	0x00	0x0000 000X
SCFG_CFG2	0x04	0x0000 0000
SCFG_EXINTC1	0x08	0x0000 0000
SCFG_EXINTC2	0x0C	0x0000 0000
SCFG_EXINTC3	0x10	0x0000 0000
SCFG_EXINTC4	0x14	0x0000 0000
SCFG_UHDRV	0x2C	0x0000 0000

7.2.1 SCFG 配置寄存器1 (SCFG_CFG1)

域	简称	复位值	类型	功能
位 31: 13	保留	0x00000 0	resd	请保持为复位值。
位 12	SWAP_QSPI	0x0	rw	QSPI 存储器地址映射控制位 (QSPI address mapping) 0: QSPI 存储器地址无重映射 1: QSPI 存储器地址重映射到 0x0900 0000 - 0x18FF FFFF
位 11	保留	0x0	resd	请保持为复位值。
位 10	SWAP_XMC	0x0	rw	XMC 存储器地址映射交换控制位 (XMC address mapping swap) 0: XMC 地址映射无交换 1: SDRAM Bank0 存储器地址交换到 0x6000 0000 - 0x6FFF FFFF 地址, 同时 NOR/ PSRAM /SRAM 存储器地址交换到 0xC000 0000 - 0xCFFF FFFF 地址
位 9: 8	保留	0x0	resd	请保持为复位值。
位 7: 6	IR_SRC_SEL	0x0	rw	红外调制包络信号源选择 (Infrared modulation envelope signal source selection) 用于选择红外调制包络信号源: 00: TMR10 01: 保留 10: 保留 11: 保留
位 5	IR_POL	0x0	rw	红外输出极性选择 (Infrared output polarity selection) 0: 红外线发射输出 (IR_OUT) 不反向 1: 红红外线发射输出 (IR_OUT) 反向
位 4: 2	保留	0x0	resd	请保持为复位值。
位 1: 0	MEM_MAP_SEL	0xX	ro	启动模式状态位 此位仅供读取, 显示复位后的启动区域。 X0: 从主存存储器启动 01: 从系统存储器启动 11: 从内置 SRAM 启动

7.2.2 SCFG配置寄存器2 (SCFG_CFG2)

域	简称	复位值	类型	功能
位 31: 27	保留	0x0000 000	resd	请保持为复位值。
位 26	CAN3_TST_SEL	0x0	rw	CAN3 时间戳计数源选择位 (CAN3 timestamp counter source sel) 0: TMR3 1: TMR5
位 25	CAN2_TST_SEL	0x0	rw	CAN2 时间戳计数源选择位 (CAN2 timestamp counter source sel) 0: TMR3 1: TMR5
位 24	CAN1_TST_SEL	0x0	rw	CAN1 时间戳计数源选择位 (CAN1 timestamp counter source sel) 0: TMR3 1: TMR5
位 23	MII_RMII_SEL	0x0	rw	MII 或 RMII 选择 (MII or RMII selection) 选择以太网使用 MII 接口还是 RMII 接口。 0: MII 1: RMII 注: 该位只在 AT32F457 上有效。
位 22: 9	保留	0x0000 000	resd	请保持为复位值。
位 8	SRAM_OPERR_STS	0	rc_w1	SRAM 奇校验错误状态(SRAM odd parity error status) 此位可以写 1 去清除。 0: 没有 SRAM 奇校验错误 1: SRAM 奇校验错误
位 7: 3	保留	0x0000 000	resd	请保持为复位值。
位 2	PVM_LK	0x0	rw	PVM 锁使能位(PVM lock enable) 0: 断开 PVM 中断与 TIM1/TIM9/TIM10/11/12/13/14 刹车输入的连接。PVMSEL 和 PVMEN 位可以被软件修改。 1: 使能 PVM 中断与 TIM1/TIM9/TIM10/11/12/13/14 刹车输入的连接。PVMSEL 和 PVMEN 位为只读位, 不能被软件修改
位 1	SRAM_OPERR_LK	0	rw	SRAM 奇校验错误锁使能位(SRAM odd parity error lock enable) 0: 断开 SRAM 奇校验错误与 TIM1/TIM9/10/11/13/14 刹车输入的连接。 1: 使能 SRAM 奇校验错误与 TIM1/TIM9/10/11/13/14 刹车输入的连接。
位 0	LOCKUP_LK	0x0	rw	CM4F LOCKUP 位使能

7.2.3 SCFG外部中断配置寄存器1 (SCFG_EXINTC1)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 12	EXINT3	0x0	rw	配置 EXINT3 的输入源 (configure EXINT3 source) 选择 EXINT3 外部中断的输入源。 0000: GPIOA 引脚 3 0001: GPIOB 引脚 3 0010: GPIOC 引脚 3 0011: GPIOD 引脚 3 0100: GPIOE 引脚 3 0101: GPIOF 引脚 3 0110: GPIOG 引脚 3 0111: GPIOH 引脚 3 其他: 保留
位 11: 8	EXINT2	0x0	rw	配置 EXINT2 的输入源 (configure EXINT2 source) 选择 EXINT2 外部中断的输入源。 0000: GPIOA 引脚 2 0001: GPIOB 引脚 2

				0010: GPIOC 引脚 2 0011: GPIOD 引脚 2 0100: GPIOE 引脚 2 0101: GPIOF 引脚 2 0110: GPIOG 引脚 2 0111: GPIOH 引脚 2 其他: 保留
位 7: 4	EXINT1	0x0	rw	配置 EXINT1 的输入源 (configure EXINT1 source) 选择 EXINT1 外部中断的输入源。 0000: GPIOA 引脚 1 0001: GPIOB 引脚 1 0010: GPIOC 引脚 1 0011: GPIOD 引脚 1 0100: GPIOE 引脚 1 0101: GPIOF 引脚 1 0110: GPIOG 引脚 1 0111: GPIOH 引脚 1 其他: 保留
位 3: 0	EXINT0	0x0	rw	配置 EXINT0 的输入源 (configure EXINT0 source) 选择 EXINT0 外部中断的输入源。 0000: GPIOA 引脚 0 0001: GPIOB 引脚 0 0010: GPIOC 引脚 0 0011: GPIOD 引脚 0 0100: GPIOE 引脚 0 0101: GPIOF 引脚 0 0110: GPIOG 引脚 0 0111: GPIOH 引脚 0 其他: 保留

7.2.4 SCFG外部中断配置寄存器2 (SCFG_EXINTC2)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 12	EXINT7	0x0	rw	配置 EXINT7 的输入源 (configure EXINT7 source) 选择 EXINT7 外部中断的输入源。 0000: GPIOA 引脚 7 0001: GPIOB 引脚 7 0010: GPIOC 引脚 7 0011: GPIOD 引脚 7 0100: GPIOE 引脚 7 0101: GPIOF 引脚 7 0110: GPIOG 引脚 7 其他: 保留
位 11: 8	EXINT6	0x0	rw	配置 EXINT6 的输入源 (configure EXINT6 source) 选择 EXINT6 外部中断的输入源。 0000: GPIOA 引脚 6 0001: GPIOB 引脚 6 0010: GPIOC 引脚 6 0011: GPIOD 引脚 6 0100: GPIOE 引脚 6 0101: GPIOF 引脚 6 0110: GPIOG 引脚 6 其他: 保留
位 7: 4	EXINT5	0x0	rw	配置 EXINT5 的输入源 (configure EXINT5 source) 选择 EXINT5 外部中断的输入源。 0000: GPIOA 引脚 5 0001: GPIOB 引脚 5 0010: GPIOC 引脚 5 0011: GPIOD 引脚 5 0100: GPIOE 引脚 5 0101: GPIOF 引脚 5 其他: 保留

				0110: GPIOG 引脚 5 其他: 保留
				配置 EXINT4 的输入源 (configure EXINT4 source) 选择 EXINT4 外部中断的输入源。
				0000: GPIOA 引脚 4 0001: GPIOB 引脚 4 0010: GPIOC 引脚 4 0011: GPIOD 引脚 4 0100: GPIOE 引脚 4 0101: GPIOF 引脚 4 0110: GPIOG 引脚 4 0111: GPIOH 引脚 4 其他: 保留
位 3: 0	EXINT4	0x0	rw	

7.2.5 SCFG外部中断配置寄存器3 (SCFG_EXINTC3)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 12	EXINT11	0x0	rw	配置 EXINT11 的输入源 (configure EXINT11 source) 选择 EXINT11 外部中断的输入源。 0000: GPIOA 引脚 11 0001: GPIOB 引脚 11 0010: GPIOC 引脚 11 0011: GPIOD 引脚 11 0100: GPIOE 引脚 11 0101: GPIOF 引脚 11 0110: GPIOG 引脚 11 其他: 保留
位 11: 8	EXINT10	0x0	rw	配置 EXINT10 的输入源 (configure EXINT10 source) 选择 EXINT10 外部中断的输入源。 0000: GPIOA 引脚 10 0001: GPIOB 引脚 10 0010: GPIOC 引脚 10 0011: GPIOD 引脚 10 0100: GPIOE 引脚 10 0101: GPIOF 引脚 10 0110: GPIOG 引脚 10 其他: 保留
位 7: 4	EXINT9	0x0	rw	配置 EXINT9 的输入源 (configure EXINT9 source) 选择 EXINT9 外部中断的输入源。 0000: GPIOA 引脚 9 0001: GPIOB 引脚 9 0010: GPIOC 引脚 9 0011: GPIOD 引脚 9 0100: GPIOE 引脚 9 0101: GPIOF 引脚 9 0110: GPIOG 引脚 9 其他: 保留
位 3: 0	EXINT8	0x0	rw	配置 EXINT8 的输入源 (configure EXINT8 source) 选择 EXINT8 外部中断的输入源。 0000: GPIOA 引脚 8 0001: GPIOB 引脚 8 0010: GPIOC 引脚 8 0011: GPIOD 引脚 8 0100: GPIOE 引脚 8 0101: GPIOF 引脚 8 0110: GPIOG 引脚 8 其他: 保留

7.2.6 SCFG外部中断配置寄存器4 (SCFG_EXINTC4)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 12	EXINT15	0x0	rw	配置 EXINT15 的输入源 (configure EXINT15 source) 选择 EXINT15 外部中断的输入源。 0000: GPIOA 引脚 15 0001: GPIOB 引脚 15 0010: GPIOC 引脚 15 0011: GPIOD 引脚 15 0100: GPIOE 引脚 15 0101: GPIOF 引脚 15 0110: GPIOG 引脚 15 其他: 保留
位 11: 8	EXINT14	0x0	rw	配置 EXINT14 的输入源 (configure EXINT14 source) 选择 EXINT14 外部中断的输入源。 0000: GPIOA 引脚 14 0001: GPIOB 引脚 14 0010: GPIOC 引脚 14 0011: GPIOD 引脚 14 0100: GPIOE 引脚 14 0101: GPIOF 引脚 14 0110: GPIOG 引脚 14 其他: 保留
位 7: 4	EXINT13	0x0	rw	配置 EXINT13 的输入源 (configure EXINT13 source) 选择 EXINT13 外部中断的输入源。 0000: GPIOA 引脚 13 0001: GPIOB 引脚 13 0010: GPIOC 引脚 13 0011: GPIOD 引脚 13 0100: GPIOE 引脚 13 0101: GPIOF 引脚 13 0110: GPIOG 引脚 13 其他: 保留
位 3: 0	EXINT12	0x0	rw	配置 EXINT12 的输入源 (configure EXINT12 source) 选择 EXINT12 外部中断的输入源。 0000: GPIOA 引脚 12 0001: GPIOB 引脚 12 0010: GPIOC 引脚 12 0011: GPIOD 引脚 12 0100: GPIOE 引脚 12 0101: GPIOF 引脚 12 0110: GPIOG 引脚 12 其他: 保留

7.2.7 SCFG超高电流推动/吸入能力 (SCFG_UHDRV)

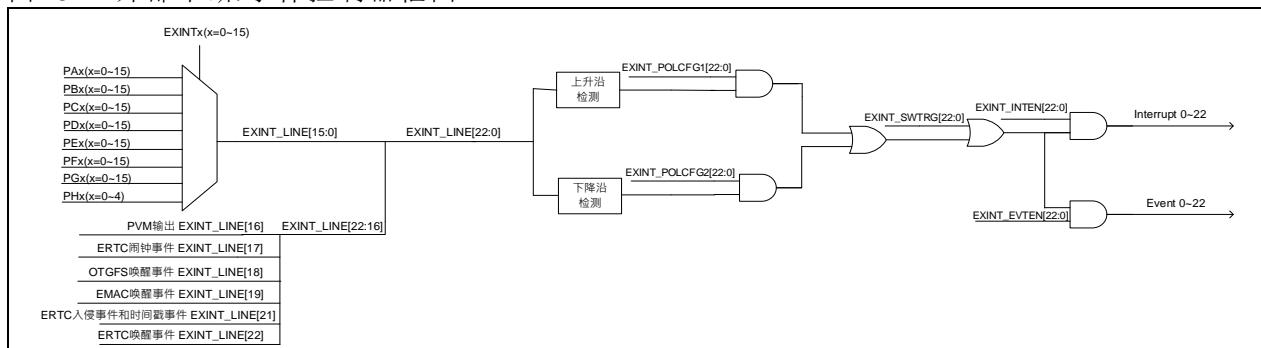
域	简称	复位值	类型	功能
位 31: 6	保留	0x0000 0000	resd	保持默认值。
位 5	PD15_UH	0x0	rw	<p>PD15 超高电流推动/吸入能力 (PD15 Ultra high sourcing/sinking strength) 这些位可由软件读写, 用于控制 PB15 PAD 电流推动/吸入能力。 0: 无效 1: 对应的 GPIO 切换为超高电流推动/吸入能力 当该位使能时, 所对应的 GPIO 电流推动/吸入能力 (GPIOx_ODRVR&GPIOx_HDRV) 控制 bit 无效</p>
位 4	PD14_UH	0x0	rw	<p>PD14 超高电流推动/吸入能力 (PD14 Ultra high sourcing/sinking strength) 这些位可由软件读写, 用于控制 PD14 PAD 电流推动/吸入能力。 0: 无效 1: 对应的 GPIO 切换为超高电流推动/吸入能力 当该位使能时, 所对应的 GPIO 电流推动/吸入能力 (GPIOx_ODRVR&GPIOx_HDRV) 控制 bit 无效</p>
位 3	PD13_UH	0x0	rw	<p>PD13 超高电流推动/吸入能力 (PD13 Ultra high sourcing/sinking strength) 这些位可由软件读写, 用于控制 PB10 PAD 电流推动/吸入能力。 0: 无效 1: 对应的 GPIO 切换为超高电流推动/吸入能力 当该位使能时, 所对应的 GPIO 电流推动/吸入能力 (GPIOx_ODRVR&GPIOx_HDRV) 控制 bit 无效</p>
位 2	PD12_UH	0x0	rw	<p>PD12 超高电流推动/吸入能力 (PD12 Ultra high sourcing/sinking strength) 这些位可由软件读写, 用于控制 PD12 PAD 电流推动/吸入能力。 0: 无效 1: 对应的 GPIO 切换为超高电流推动/吸入能力 当该位使能时, 所对应的 GPIO 电流推动/吸入能力 (GPIOx_ODRVR&GPIOx_HDRV) 控制 bit 无效</p>
位 1	PB9_UH	0x0	rw	<p>PB9 超高电流推动/吸入能力 (PB9 Ultra high sourcing/sinking strength) 这些位可由软件读写, 用于控制 PB9 PAD 电流推动/吸入能力。 0: 无效 1: 对应的 GPIO 切换为超高电流推动/吸入能力 当该位使能时, 所对应的 GPIO 电流推动/吸入能力 (GPIOx_ODRVR&GPIOx_HDRV) 控制 bit 无效</p>
位 0	PB8_UH	0x0	rw	<p>PB8 超高电流推动/吸入能力 (PB8 Ultra high sourcing/sinking strength) 这些位可由软件读写, 用于控制 PB8 PAD 电流推动/吸入能力。 0: 无效 1: 对应的 GPIO 切换为超高电流推动/吸入能力 当该位使能时, 所对应的 GPIO 电流推动/吸入能力 (GPIOx_ODRVR&GPIOx_HDRV) 控制 bit 无效</p>

8 外部中断/事件控制器 (EXINT)

8.1 EXINT介绍

EXINT 共计有 22 条中断线 EXINT_LINE[22:0](其中位 20 为保留位), 每条中断线均支持通过边沿检测触发和软件触发来产生中断或事件。EXINT 可以根据软件配置, 独立的使能或禁止中断或事件, 并采取不同的边沿检测方式(检测上升沿或检测下降沿或同时检测上升沿和下降沿)以及触发方式(边沿检测触发或软件触发或边沿检测和软件同时触发)响应触发源独立的产生中断或事件。

图 8-1 外部中断/事件控制器框图



EXINT 控制器的主要特性:

- 中断线 0~15 所映射的 IO 可以独立的配置
- 每个中断线都有独立的触发方式选择
- 每个中断都有独立的使能位
- 每个事件都有独立的使能位
- 共 22 个可独立产生和清除的软件触发
- 每个中断都有独立的状态位
- 每个中断都可以被独立的清除

8.2 功能描述和配置流程

EXINT 共计有 22 条中断线 EXINT_LINE[22:0](其中位 20 为保留位), 可以通过边沿检测的方式分别检测来自 GPIO 的外部中断源以及包括 PVM 输出, ERTC 闹钟事件, ERTC 入侵事件和时间戳事件, ERTC 唤醒事件, OTGFS 唤醒事件以及 EMAC 唤醒事件共六种芯片内部的中断源, 其中来自 GPIO 的中断源可以通过软件编程配置 SCFG 中的外部中断配置寄存器 x (SCFG_EXINTCx) 灵活的选择, 需要注意的是这些输入源是互斥的, 例如 EXINT_LINE0 只能选择 PA0/PB0/PC0/PD0…中的某一个, 而不能同时选择 PA0 和 PB0 作为输入源。

EXINT 支持多种边沿检测方式, 每条中断线可以通过软件编程配置极性配置寄存器 1 (EXINT_POLCFG1) 和极性配置寄存器 2 (EXINT_POLCFG2) 独立的选择上升沿检测或下降沿检测或同时进行上升沿和下降沿检测, 中断线上检测到的有效边沿触发可以用于产生事件或中断。

EXINT 支持独立的软件触发产生中断或事件, 即除了来自中断线上的有效边沿外, 用户可以通过软件编程配置软件触发寄存器 (EXINT_SWTRG) 对应位来产生对应的中断或事件。

EXINT 具备独立的中断和事件使能位, 用户可以通过软件编程配置中断使能寄存器 (EXINT_INTEN) 和事件使能寄存器 (EXINT_EVTEN) 来使能或关闭对应的中断或事件, 这意味着无论是通过边沿检测还是软件触发产生中断或事件, 都需要提前使能对应的中断或事件。

EXINT 具备独立的中断状态位, 用户可以通过中断状态寄存器 (EXINT_INTSTS) 读取对应的中断状态并通过对该寄存器相应位写 1 来清除已置位的状态标志。

中断初始化流程

1. 选择中断源
即配置复用外部中断配置寄存器 x (SCFG_EXINTCx) (如果需要使用 GPIO 作为中断源需要该步骤)。
2. 选择触发方式
即配置极性配置寄存器 1 (EXINT_POLCFG1) 和极性配置寄存器 2 (EXINT_POLCFG2)。

3. 使能中断或事件

即配置中断使能寄存器（EXINT_INTEN）和事件使能寄存器（EXINT_EVTEN）。

4. 产生软件触发

即配置软件触发寄存器（EXINT_SWTRG）产生软件触发（此步骤仅适用于需软件触发产生中断的应用）。

注意：若需要更改中断源配置，应先关闭中断使能寄存器和事件使能寄存器后，再重新开始中断初始化流程的配置。

中断清除流程

- 清除标志，即对中断状态寄存器（EXINT_INTSTS）对应位写 1 来清除已产生的中断，同时该操作会同步清除软件触发寄存器（EXINT_SWTRG）中的对应位。

8.3 EXINT 寄存器描述

下表列出了 EXINT 寄存器的映像和复位值。

必须以字（32 位）的方式操作这些外设寄存器。

表 8-1 外部中断/事件控制器寄存器映像和复位值

寄存器简称	基址偏移量	复位值
EXINT_INTEN	0x00	0x0000 0000
EXINT_EVTEN	0x04	0x0000 0000
EXINT_POLCFG1	0x08	0x0000 0000
EXINT_POLCFG2	0x0C	0x0000 0000
EXINT_SWTRG	0x10	0x0000 0000
EXINT_INTSTS	0x14	0x0000 0000

8.3.1 中断使能寄存器（EXINT_INTEN）

域	简称	复位值	类型	功能
位 31: 23	保留	0x000	resd	硬件强制为 0。
位 22: 0	INTENx	0x000000	rw	线 x 上的中断使能/禁止位（Interrupt enable or disable on line x） 0: 禁止中断请求； 1: 使能中断请求。 注：位 20 为保留位，未使用。

8.3.2 事件使能寄存器（EXINT_EVTEN）

域	简称	复位值	类型	功能
位 31: 23	保留	0x000	resd	硬件强制为 0。
位 22: 0	EVTENx	0x000000	rw	线 x 上的事件使能/禁止位（Event enable or disable on line x） 0: 禁止事件请求； 1: 使能事件请求。 注：位 20 为保留位，未使用。

8.3.3 极性配置寄存器1（EXINT_POLCFG1）

域	简称	复位值	类型	功能
位 31: 23	保留	0x000	resd	硬件强制为 0。
位 22: 0	RPx	0x000000	rw	线 x 上的上升沿触发事件配置位（Rising polarity configuration bit of line x） 这些位用于选择线 x 由上升沿触发中断和事件。 0: 禁止上升沿触发； 1: 使能上升沿触发。 注：位 20 为保留位，未使用。

8.3.4 极性配置寄存器2 (EXINT_POLCFG2)

域	简称	复位值	类型	功能
位 31: 23	保留	0x000	resd	硬件强制为 0。
位 22: 0	FRx	0x000000	rw	线 x 上的下降沿触发事件配置位 (Falling polarity event configuration bit of line x) 这些位用于选择线 x 由下降沿触发中断和事件。 0: 禁止下降沿触发; 1: 允许下降沿触发。 注: 位 20 为保留位, 未使用。

8.3.5 软件触发寄存器 (EXINT_SWTRG)

域	简称	复位值	类型	功能
位 31: 23	保留	0x000	resd	硬件强制为 0。
位 22: 0	SWTx	0x000000	rw	软件触发线 x (Software trigger on line x) 当 EXINT_INTEN 寄存器中的对应位为 1, 则软件写此位 硬件将自动置起 EXINT_INTSTS 寄存器中的对应位并产 生中断。 当 EXINT_EVTEN 寄存器中的对应位为 1, 则软件写此位 硬件将自动产生对应中断线上的事件。 0: 默认值; 1: 产生软件触发。 注: 通过清除 EXINT_INTSTS 的对应位 (写入 1), 可 以清除该位为 0。 注: 位 20 为保留位, 未使用。

8.3.6 中断状态寄存器 (EXINT_INTSTS)

域	简称	复位值	类型	功能
位 31: 23	保留	0x000	resd	硬件强制为 0。
位 22: 0	LINE _x	0x000000	rw1c	线 x 状态位 (Line x state bit) 0: 没有发生中断; 1: 发生了中断。 注: 在该位中写入'1'可以清除它。 注: 位 20 为保留位, 未使用。

9 DMA 控制器 (DMA)

9.1 简介

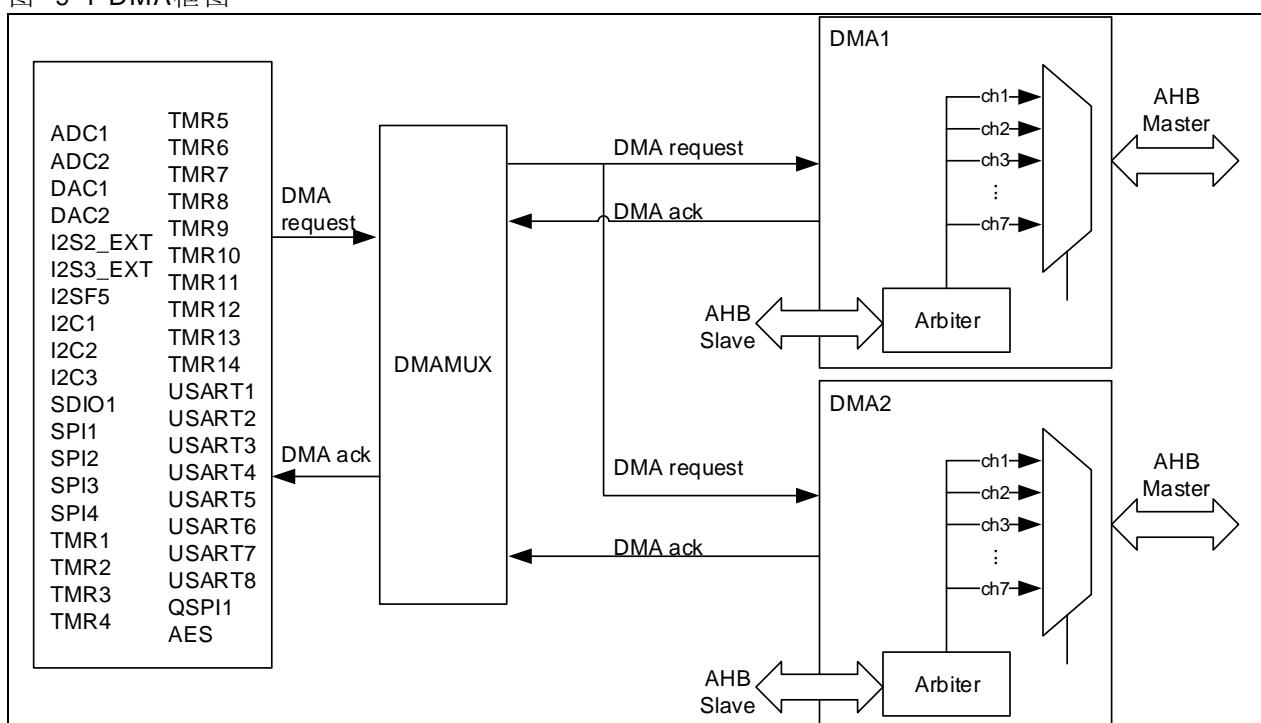
直接存储器存取 (DMA) 用来提供在外设和存储器之间或者存储器和存储器之间的高速数据传输。无须 CPU 干预，数据可以通过 DMA 快速地移动，这就节省了 CPU 的资源来做其他操作。

一个处理器包含 2 个 DMA 控制器。每个控制器各有 7 个 DMA 通道，每个通道管理来自于外设对存储器访问的请求，并由仲裁器来协调各个 DMA 请求的优先权。

9.2 特性

- 符合 AMBA 规范 (Rev. 2.0)
- 仅支持 AHB OKAY 和 ERROR 响应
- 不支持 AHB 主接口的 HBUSREQ 和 HGRANT
- 支持 7 个通道
- 支持外设到存储器，存储器到外设和存储器到存储器的传输
- 支持硬件握手
- 支持 8 位，16 位和 32 位数据宽度传输
- 传输数据长度最大为 0xFFFFFFFF，可由编程配置
- 支持多路复用器

图 9-1 DMA 框图



注意：根据不同型号，图中 DMA 外设可能会有所减少。

9.3 功能描述

9.3.1 通道配置

1. 设置外设地址 (DMA_CPBAX寄存器)

数据传输的初始外设地址，在传输过程中不会被改变。

2. 设置存储器地址 (DMA_CMBAX寄存器)

数据传输的初始存储器地址，在传输过程中不会被改变。

3. 配置数据传输量 (DMA_DTCNTx寄存器)

可编程的传输数据长度最大为0xFFFFFFFF。在传输过程中，该传输数据量的值会逐渐递减。

4. 配置通道设定 (DMA_CHCTRLx寄存器)

包含通道优先级，数据传输的方向、宽度，地址增量模式、循环模式和中断方式。

通道优先级 (CHPL)

分为4个等级，最高优先级、高优先级、中等优先级和低优先级。

若有2个通道优先级设定相同，则较低编号的通道有较高的优先权。举例，通道1优先于通道2。

数据传输方向 (DTD)

分为存储器到外设 (M2P)，外设到存储器 (P2M)。

地址增量模式 (PINCM/MINCM)

当设置为增量模式时，下一笔传输的地址将是前一笔传输地址加上传输宽度 (PWIDH/MWIDH)。

循环模式 (LM)

当通道配置设定为循环模式时，在最后一次传输后 DMA_DTCNTx 寄存器的内容会恢复成初始值。

存储器到存储器模式 (M2M)

存储器到存储器模式是 DMA 在没有外设请求的情况下进行数据传输。

循环模式与存储器到存储器模式不能同时使用。

5. 使能该通道的DMA传输 (DMA_CHCTRLx寄存器的CHEN位)

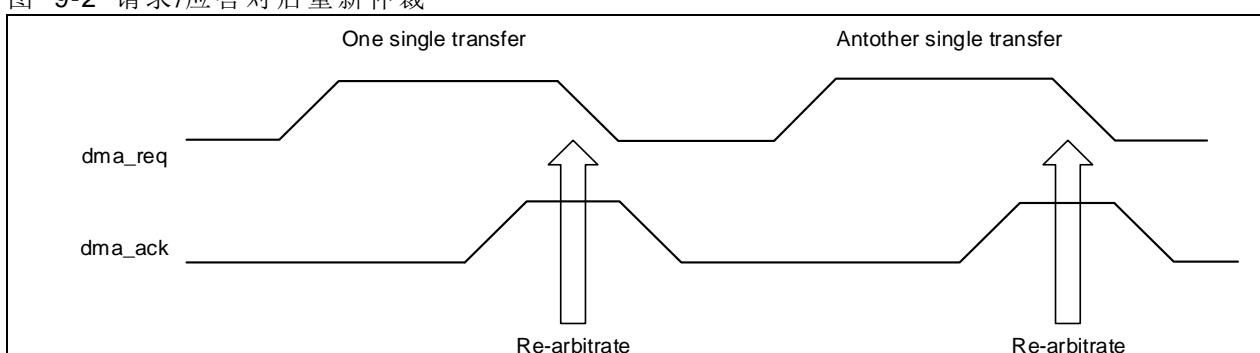
9.3.2 握手机制

在P2M和M2P传输模式，外设需要向DMA控制器发送请求信号。该通道将发出外设传输（单次），直到请求信号被应答为止。外设传输完成后，DMA控制器将应答信号发送到外设。外设从DMA控制器获得应答信号后立即释放其请求。一旦外设取消了请求，DMA控制器将释放应答信号。

9.3.3 仲裁

当同时启用多个通道时，仲裁器将在主控制器完全传输数据后重新进行仲裁。优先级最高的通道将在先前的通道完成占用主控制器之后，具有主控制器优先级。每当通道以外设主控制器的优先级完成一个单次传输后，外设主控制器就会重新仲裁以服务其他通道。

图 9-2 请求/应答对后重新仲裁



9.3.4 可编程数据传输宽度

通过 DMA_CHCTRLx 中的 PWIDTH 和 MWIDTH 位可以对源数据和目标数据的数据宽度进行编程，当 PWIDTH 不等于 MWIDTH 时，会依据 PWIDTH/ MWIDTH 设定将资料对齐。

图 9-3 PWIDTH: byte, MWIDTH: half-word

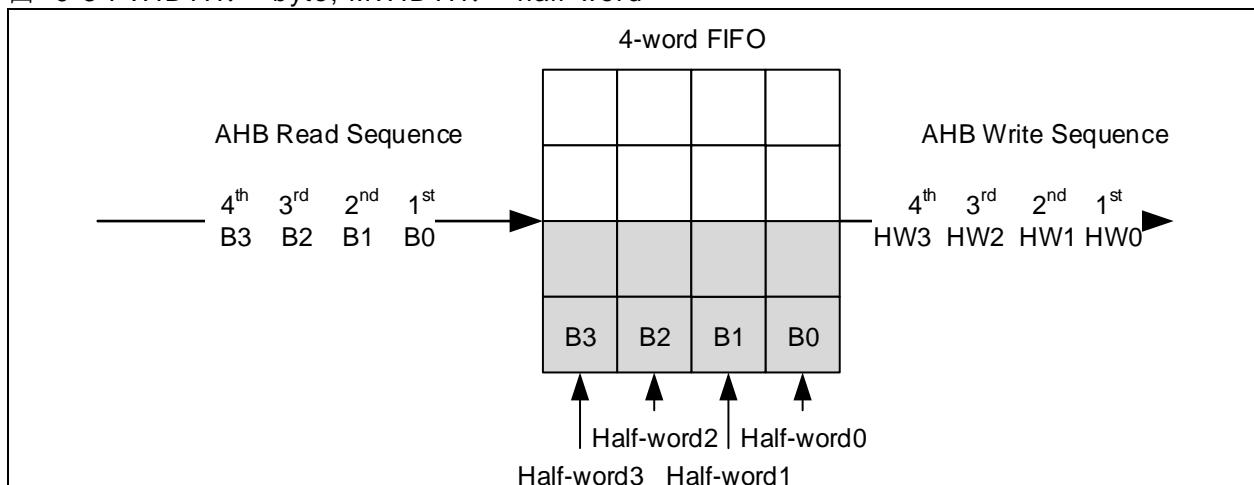


图 9-4 PWIDTH: half-word, MWIDTH: word

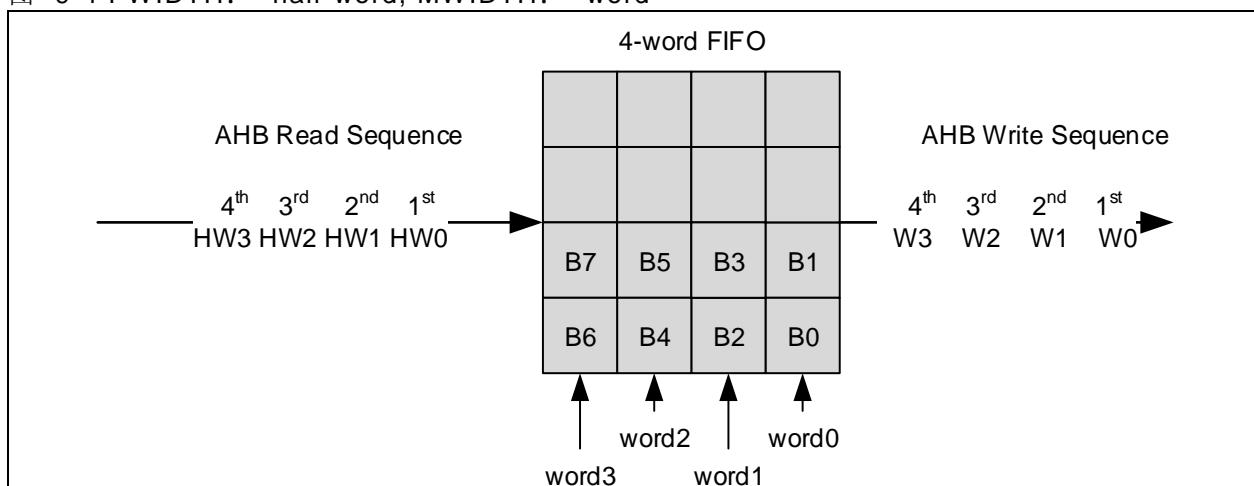
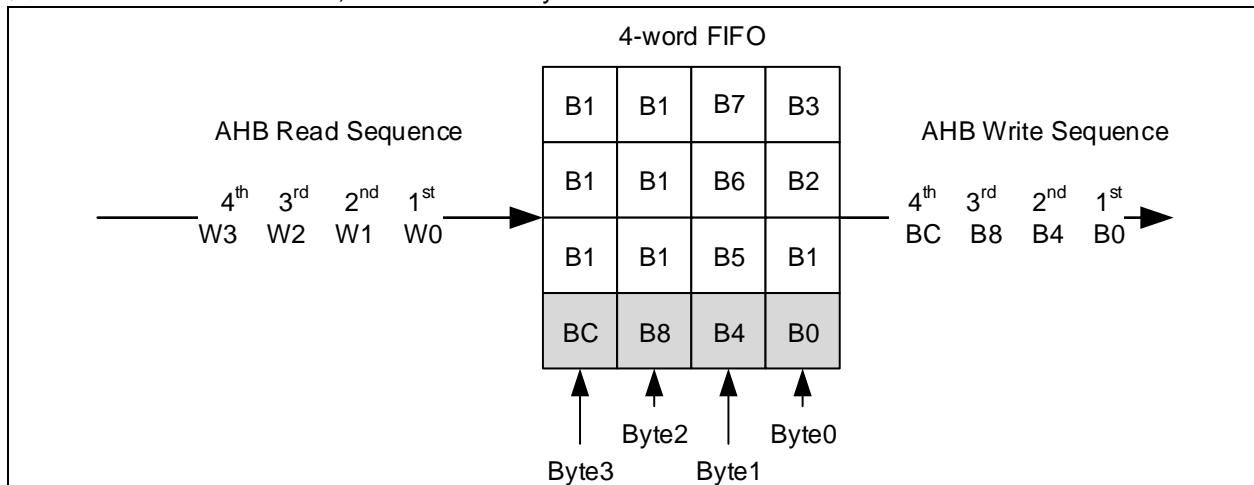


图 9-5 PWIDTH: word, MWIDTH: byte



9.3.5 错误事件

表 9-1 DMA 错误事件

错误事件

传输错误 DMA 读/写访问期间发生 AHB 响应错误

9.3.6 中断

DMA 可在传输过半、传输完成和传输错误时产生中断。每个通道的中断都有专用标志，清除和使能位如下表所示。

表 9-2 DMA 中断

中断事件	事件标志位	清除控制位	使能控制位
半传输	HDTF	HDTFC	HDTIEN
传输完成	FDTF	FDTFC	FDTIEN
传输错误	DTERRF	DTERRFC	DTERRIEN

9.4 多路复用器 (DMAMUX)

DMAMUX 在外设和 DMA 控制器之间路由 DMA 请求/确认。

DMA 控制器通过 DMA_MUXSEL 中的 TBL_SEL 位选择 DMA 映射表。每个 DMA 控制器流可从弹性映射表中选择一个唯一的 DMA 请求。在弹性映射中，每个通道可以通过 DMA_MUXCxCTRL 中的 REQSEL [6: 0] 字段旁路或同步来自外设或生成器的 127 个可能的通道请求。

9.4.1 DMAMUX 功能描述

DMAMUX 具有两个功能：请求生成器和请求多路复用器。

每个 DMAMUX 生成器通道 x 在 DMA_MUXGxCTRL 中都有一个使能 GEN 位。通过 SIGSEL 字段选择 DMAMUX 生成器的触发输入，通常 DMA 请求的数量为 GREQCNT + 1。通过 DMA_MUXGxCTRL 中的 GPOL 字段选择触发事件，该事件可以是上升沿，下降沿或任一沿。

每个 DMAMUX 多路复用器流 x 来自弹性映射 all_req [127: 1] 请求组输入。

在弹性映射中，可以通过 DMA_MUXSxCTRL 中的 SYNCEN 位来同步选定的 DMA 请求输入。当通道处于同步模式时，通过 DMA_MUXSxCTRL 中的 SYNCSEL 字段选择同步输入，然后一旦通过 DMA_MUXSxCTRL 中的 SYNCPOL [1: 0] 字段检测到同步输入的有效沿，则所选的 DMA 请求输入将传播到多路复用器请求输出 chx_mux_req [7: 0]。另外，当通过 DMA_MUXCxCTRL 中的 EVTGEN 位启用流的事件生成时，可编程请求计数器 REQCNT 用于请求输出生成和事件生成。

图 9-6 DMAMUX 框图

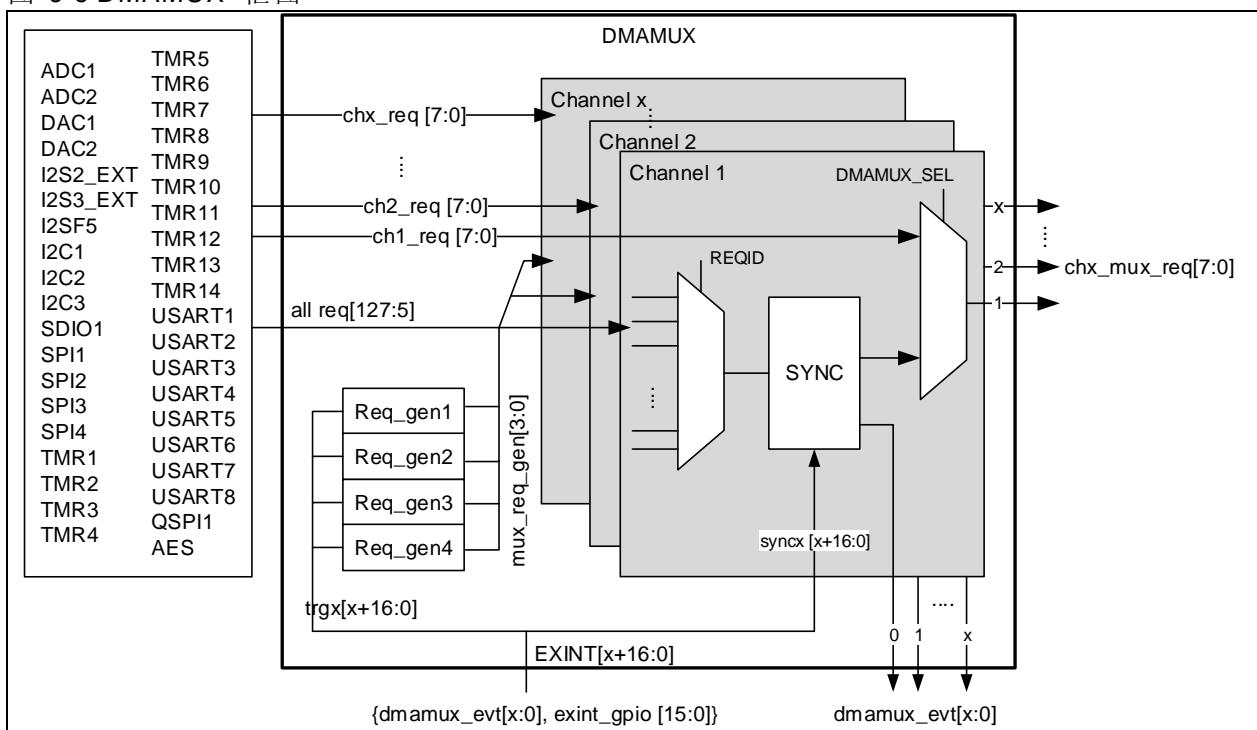


表 9-3 DMA1 / DMA2请求弹性映射

CHx_SRC	请求来源	CHx_SRC	请求来源	CHx_SRC	请求来源	CHx_SRC	请求来源
1	DMA_MUXREQG1	33	USART5_TX	65	TMR3_OVERFLOW	97	TMR12_TRIG
2	DMA_MUXREQG2	34	reserved	66	TMR3_TRIG	98	TMR12_HALL
3	DMA_MUXREQG3	35	reserved	67	TMR4_CH1	99	reserved
4	DMA_MUXREQG4	36	ADC2	68	TMR4_CH2	100	reserved
5	ADC1	37	reserved	69	TMR4_CH3	101	reserved
6	DAC1	38	reserved	70	TMR4_CH4	102	reserved
7	reserved	39	SDIO1	71	TMR4_OVERFLOW	103	reserved
8	TMR6_OVERFLOW	40	QSPI1	72	TMR5_CH1	104	reserved
9	TMR7_OVERFLOW	41	DAC2	73	TMR5_CH2	105	reserved
10	SPI1_RX	42	TMR1_CH1	74	TMR5_CH3	106	SPI4_RX
11	SPI1_TX	43	TMR1_CH2	75	TMR5_CH4	107	SPI4_TX
12	SPI2_RX	44	TMR1_CH3	76	TMR5_OVERFLOW	108	I2SF5_RX
13	SPI2_TX	45	TMR1_CH4	77	TMR5_TRIG	109	I2SF5_TX
14	SPI3_RX	46	TMR1_OVERFLOW	78	TMR9_CH1	110	I2S2EXT_RX
15	SPI3_TX	47	TMR1_TRIG	79	TMR9_OVERFLOW	111	I2S2EXT_TX
16	I2C1_RX	48	TMR1_HALL	80	TMR9_TRIG	112	I2S3EXT_RX
17	I2C1_TX	49	TMR8_CH1	81	TMR9_HALL	113	I2S3EXT_TX
18	I2C2_RX	50	TMR8_CH2	82	TMR10_CH1	114	USART6_RX
19	I2C2_TX	51	TMR8_CH3	83	TMR10_OVERFLOW	115	USART6_TX
20	I2C3_RX	52	TMR8_CH4	84	TMR11_CH1	116	USART7_RX
21	I2C3_TX	53	TMR8_OVERFLOW	85	TMR11_OVERFLOW	117	USART7_TX
22	reserved	54	TMR8_TRIG	86	reserved	118	USART8_RX
23	reserved	55	TMR8_HALL	87	reserved	119	USART8_TX
24	USART1_RX	56	TMR2_CH1	88	reserved	120	TMR13_CH1
25	USART1_TX	57	TMR2_CH2	89	reserved	121	TMR13_OVERFLOW
26	USART2_RX	58	TMR2_CH3	90	reserved	122	TMR14_CH1
27	USART2_TX	59	TMR2_CH4	91	AES_IN	123	TMR14_OVERFLOW
28	USART3_RX	60	TMR2_OVERFLOW	92	AES_OUT	124	TMR9_CH2
29	USART3_TX	61	TMR3_CH1	93	reserved	125	TMR12_CH2
30	USART4_RX	62	TMR3_CH2	94	reserved	126	TMR2_TRIG
31	USART4_TX	63	TMR3_CH3	95	TMR12_CH1	127	TMR4_TRIG
32	USART5_RX	64	TMR3_CH4	96	TMR12_OVERFLOW		

表 9-4 DMAMUX EXINT LINE 用于触发输入和同步输入

EXINT LINE	来源	EXINT LINE	来源	EXINT LINE	来源	EXINT LINE	来源
0	exint_gpio[0]	8	exint_gpio[8]	16	DMA_MUXevt1	24	reserved
1	exint_gpio[1]	9	exint_gpio[9]	17	DMA_MUXevt2	25	reserved
2	exint_gpio[2]	10	exint_gpio[10]	18	DMA_MUXevt3	26	reserved
3	exint_gpio[3]	11	exint_gpio[11]	19	DMA_MUXevt4	27	reserved
4	exint_gpio[4]	12	exint_gpio[12]	20	DMA_MUXevt5	28	reserved
5	exint_gpio[5]	13	exint_gpio[13]	21	DMA_MUXevt6	39	reserved
6	exint_gpio[6]	14	exint_gpio[14]	22	DMA_MUXevt7	30	reserved
7	exint_gpio[7]	15	exint_gpio[15]	23	reserved	31	reserved

9.4.2 DMAMUX 溢出中断

在 DMAMUX 请求生成中，如果在请求生成器计数器 GREQCNT 下溢之前发生了新的触发输入，则在 DMA_MUXGSTS 中声明请求触发溢出标志位 TRGOVFX。通过将 DMA_MUXGCLR 中的相关位 TRGOVFCx 置 1，可以清除溢出标志 TRGOVFX。

如果在 DMA_MUXGxCTRL 中设置了中断允许位 TRGOVIEN，则 DMAMUX 请求触发溢出标志会产生一个中断。

在 DMAMUX 同步模式下，如果在请求计数器 REQCNT 下溢之前发生了新的同步输入，则在 DMA_MUXSYNCSTS 中声明同步溢出标志位 SYNCOVFX。通过将 DMA_MUXSYNCCLR 中的相关位 SYNCOVFCx 置 1，可以清除溢出标志 SYNCOVFX。

如果在 DMA_MUXSxCTRL 中设置了中断允许位 SYNCOVIEN，则 DMAMUX 同步溢出标志会产生一个中断。

图 9-7 DMAMUX 请求同步模式

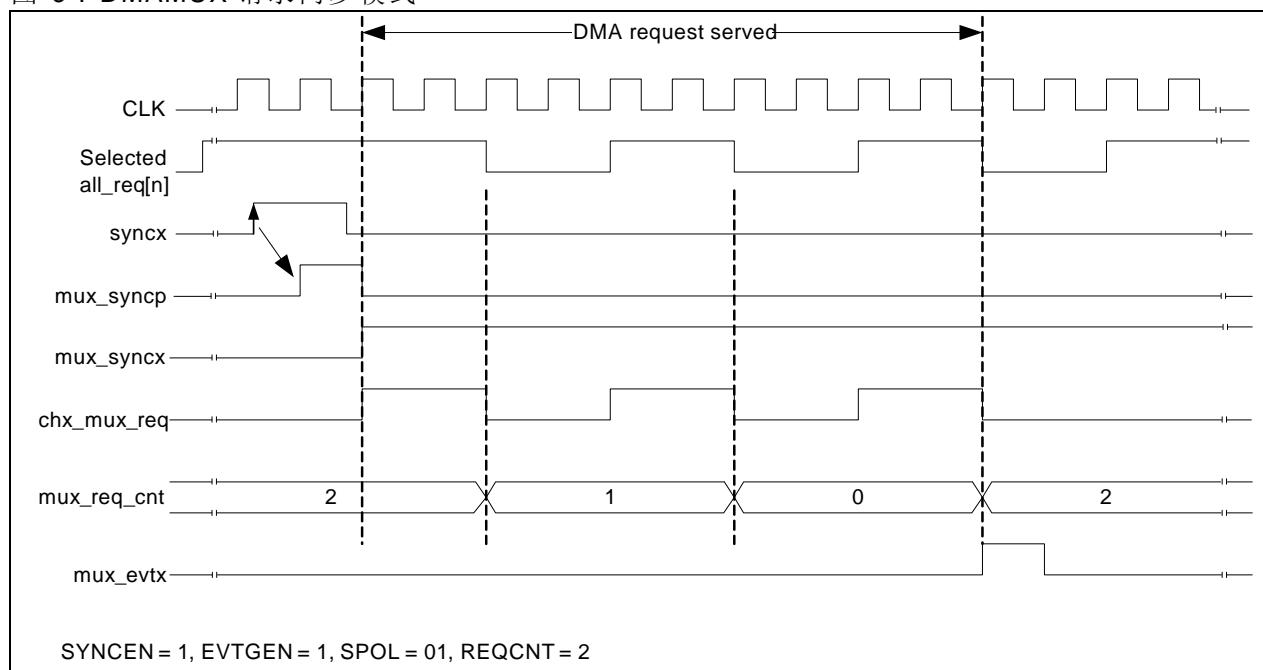
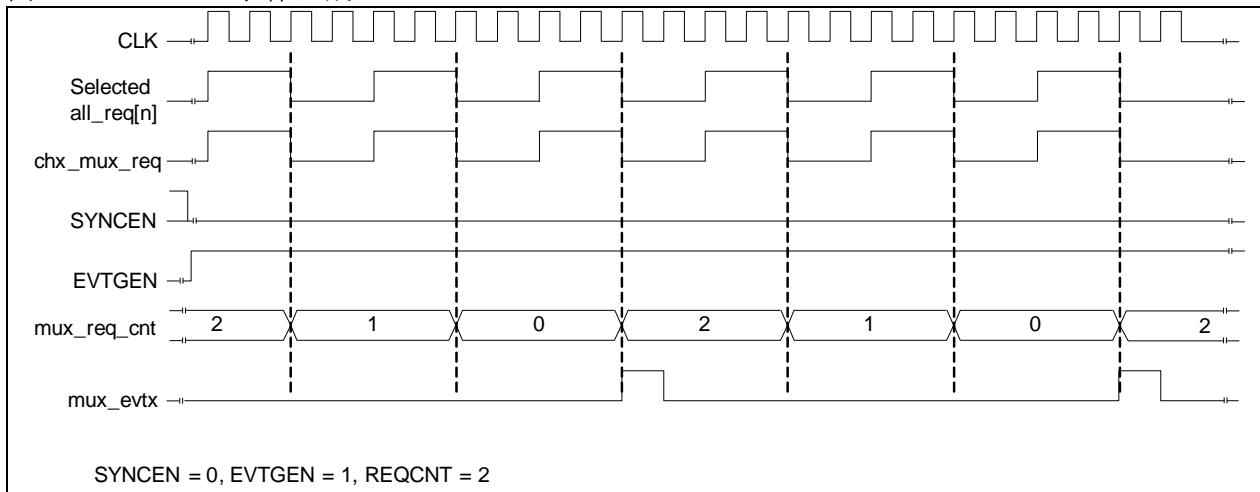


图 9-8 DMAMUX 事件生成



9.5 DMA 寄存器

下表列出了 DMA 寄存器的映像和复位值。

可以用字节（8 位）、半字（16 位）或字（32 位）的方式操作这些外设寄存器。

表 9-5 DMA 寄存器的映像和复位值

寄存器简称	基址偏移量	复位值
DMA_STS	0x00	0x0000 0000
DMA_CLR	0x04	0x0000 0000
DMA_C1CTRL	0x08	0x0000 0000
DMA_C1DTCNT	0x0c	0x0000 0000
DMA_C1PADDR	0x10	0x0000 0000
DMA_C1MADDR	0x14	0x0000 0000
DMA_C2CTRL	0x1c	0x0000 0000
DMA_C2DTCNT	0x20	0x0000 0000
DMA_C2PADDR	0x24	0x0000 0000
DMA_C2MADDR	0x28	0x0000 0000
DMA_C3CTRL	0x30	0x0000 0000
DMA_C3DTCNT	0x34	0x0000 0000
DMA_C3PADDR	0x38	0x0000 0000
DMA_C3MADDR	0x3c	0x0000 0000
DMA_C4CTRL	0x44	0x0000 0000
DMA_C4DTCNT	0x48	0x0000 0000
DMA_C4PADDR	0x4c	0x0000 0000
DMA_C4MADDR	0x50	0x0000 0000
DMA_C5CTRL	0x58	0x0000 0000
DMA_C5DTCNT	0x5c	0x0000 0000
DMA_C5PADDR	0x60	0x0000 0000
DMA_C5MADDR	0x64	0x0000 0000
DMA_C6CTRL	0x6c	0x0000 0000
DMA_C6DTCNT	0x70	0x0000 0000

DMA_C6PADDR	0x74	0x0000 0000
DMA_C6MADDR	0x78	0x0000 0000
DMA_C7CTRL	0x80	0x0000 0000
DMA_C7DTCNT	0x84	0x0000 0000
DMA_C7PADDR	0x88	0x0000 0000
DMA_C7MADDR	0x8c	0x0000 0000
DMA_MUXSEL	0x100	0x0000 0000
DMA_MUXC1CTRL	0x104	0x0000 0000
DMA_MUXC2CTRL	0x108	0x0000 0000
DMA_MUXC3CTRL	0x10c	0x0000 0000
DMA_MUXC4CTRL	0x110	0x0000 0000
DMA_MUXC5CTRL	0x114	0x0000 0000
DMA_MUXC6CTRL	0x118	0x0000 0000
DMA_MUXC7CTRL	0x11c	0x0000 0000
DMA_MUXG1CTRL	0x120	0x0000 0000
DMA_MUXG2CTRL	0x124	0x0000 0000
DMA_MUXG3CTRL	0x128	0x0000 0000
DMA_MUXG4CTRL	0x12c	0x0000 0000
DMA_MUXSYNCSTS	0x130	0x0000 0000
DMA_MUXSYNCLR	0x134	0x0000 0000
DMA_MUXGSTS	0x138	0x0000 0000
DMA_MUXGCLR	0x13c	0x0000 0000

9.5.1 DMA状态寄存器 (DMA_STS)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 28	保留	0x0	resd	保持默认值。
位 27	DTERRF7	0x0	ro	通道 7 数据传输错误事件标志 (Data transfer error event flag) 0: 未发生错误传输事件 1: 发生错误传输事件
位 26	HDTF7	0x0	ro	通道 7 半数据传输事件标志 (Half data transfer event flag) 0: 未发生半传输事件 1: 发生半传输事件
位 25	FDTF7	0x0	ro	通道 7 数据传输完成事件标志 (Full data transfer event flag) 0: 未发生传输完成事件 1: 发生传输完成事件
位 24	GF7	0x0	ro	通道 7 全局事件标志 (Global event flag) 0: 未发生传输错误、半传输完成或传输完成事件 1: 发生传输错误、半传输完成或传输完成事件
位 23	DTERRF6	0x0	ro	通道 6 数据传输错误事件标志 (Data transfer error event flag) 0: 未发生错误传输事件 1: 发生错误传输事件
位 22	HDTF6	0x0	ro	通道 6 半数据传输事件标志 (Half data transfer event flag) 0: 未发生半传输事件 1: 发生半传输事件
位 21	FDTF6	0x0	ro	通道 6 数据传输完成事件标志 (Full data transfer event flag) 0: 未发生传输完成事件 1: 发生传输完成事件
位 20	GF6	0x0	ro	通道 6 全局事件标志 (Global event flag) 0: 未发生传输错误、半传输完成或传输完成事件 1: 发生传输错误、半传输完成或传输完成事件
位 19	DTERRF5	0x0	ro	通道 5 数据传输错误事件标志 (Data transfer error event flag) 0: 未发生错误传输事件 1: 发生错误传输事件
位 18	HDTF5	0x0	ro	通道 5 半数据传输事件标志 (Half data transfer event flag) 0: 未发生半传输事件 1: 发生半传输事件
位 17	FDTF5	0x0	ro	通道 5 数据传输完成事件标志 (Full data transfer event flag) 0: 未发生传输完成事件 1: 发生传输完成事件
位 16	GF5	0x0	ro	通道 5 全局事件标志 (Global event flag) 0: 未发生传输错误、半传输完成或传输完成事件 1: 发生传输错误、半传输完成或传输完成事件
位 15	DTERRF4	0x0	ro	通道 4 数据传输错误事件标志 (Data transfer error event flag) 0: 未发生错误传输事件 1: 发生错误传输事件

位 14	HDTF4	0x0	ro	通道 4 半数据传输事件标志 (Half data transfer event flag) 0: 未发生半传输事件 1: 发生半传输事件
位 13	FDTF4	0x0	ro	通道 4 数据传输完成事件标志 (Full data transfer event flag) 0: 未发生传输完成事件 1: 发生传输完成事件
位 12	GF4	0x0	ro	通道 4 全局事件标志 (Global event flag) 0: 未发生传输错误、半传输完成或传输完成事件 1: 发生传输错误、半传输完成或传输完成事件
位 11	DTERRF3	0x0	ro	通道 3 数据传输错误事件标志 (Data transfer error event flag) 0: 未发生错误传输事件 1: 发生错误传输事件
位 10	HDTF3	0x0	ro	通道 3 半数据传输事件标志 (Half data transfer event flag) 0: 未发生半传输事件 1: 发生半传输事件
位 9	FDTF3	0x0	ro	通道 3 数据传输完成事件标志 (Full data transfer event flag) 0: 未发生传输完成事件 1: 发生传输完成事件
位 8	GF3	0x0	ro	通道 3 全局事件标志 (Global event flag) 0: 未发生传输错误、半传输完成或传输完成事件 1: 发生传输错误、半传输完成或传输完成事件
位 7	DTERRF2	0x0	ro	通道 2 数据传输错误事件标志 (Data transfer error event flag) 0: 未发生错误传输事件 1: 发生错误传输事件
位 6	HDTF2	0x0	ro	通道 2 半数据传输事件标志 (Half data transfer event flag) 0: 未发生半传输事件 1: 发生半传输事件
位 5	FDTF2	0x0	ro	通道 2 数据传输完成事件标志 (Full data transfer event flag) 0: 未发生传输完成事件 1: 发生传输完成事件
位 4	GF2	0x0	ro	通道 2 全局事件标志 (Global event flag) 0: 未发生传输错误、半传输完成或传输完成事件 1: 发生传输错误、半传输完成或传输完成事件
位 3	DTERRF1	0x0	ro	通道 1 数据传输错误事件标志 (Data transfer error event flag) 0: 未发生错误传输事件 1: 发生错误传输事件
位 2	HDTF1	0x0	ro	通道 1 半数据传输事件标志 (Half data transfer event flag) 0: 未发生半传输事件 1: 发生半传输事件
位 1	FDTF1	0x0	ro	通道 1 数据传输完成事件标志 (Full data transfer event flag) 0: 未发生传输完成事件 1: 发生传输完成事件
位 0	GF1	0x0	ro	通道 1 全局事件标志 (Global event flag) 0: 未发生传输错误、半传输完成或传输完成事件 1: 发生传输错误、半传输完成或传输完成事件

9.5.2 DMA标志清除寄存器 (DMA_CLR)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 28	保留	0x0	resd	保持默认值。
位 27	DTERRFC7	0x0	rw1c	清除通道 7 的数据传输错误标志 (Data transfer error flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF7 标志
位 26	HDTFC7	0x0	rw1c	清除通道 7 的半数据传输标志 (Half data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 HDTF7 标志
位 25	FDTFC7	0x0	rw1c	清除通道 7 的数据传输完成标志 (Full data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 FDTF7 标志
位 24	GFC7	0x0	rw1c	清除通道 7 的全局中断标志 (Global flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF7、HDTF7、FDTF7 和 GF7 标志
位 23	DTERRFC6	0x0	rw1c	清除通道 6 的数据传输错误标志 (Data transfer error flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF6 标志
位 22	HDTFC6	0x0	rw1c	清除通道 6 的半数据传输标志 (Half data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 HDTF6 标志
位 21	FDTFC6	0x0	rw1c	清除通道 6 的数据传输完成标志 (Full data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 FDTF6 标志
位 20	GFC6	0x0	rw1c	清除通道 6 的全局中断标志 (Global flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF6、HDTF6、FDTF6 和 GF6 标志
位 19	DTERRFC5	0x0	rw1c	清除通道 5 的数据传输错误标志 (Data transfer error flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF5 标志
位 18	HDTFC5	0x0	rw1c	清除通道 5 的半数据传输标志 (Half data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 HDTF5 标志
位 17	FDTFC5	0x0	rw1c	清除通道 5 的数据传输完成标志 (Full data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 FDTF5 标志
位 16	GFC5	0x0	rw1c	清除通道 5 的全局中断标志 (Global flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF5、HDTF5、FDTF5 和 GF5 标志
位 15	DTERRFC4	0x0	rw1c	清除通道 4 的数据传输错误标志 (Data transfer error flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF4 标志

位 14	HDTFC4	0x0	rw1c	清除通道 4 的半数据传输标志 (Half data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 HDTF4 标志
位 13	FDTFC4	0x0	rw1c	清除通道 4 的数据传输完成标志 (Full data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 FDTF4 标志
位 12	GFC4	0x0	rw1c	清除通道 4 的全局中断标志 (Global flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF4、HDTF4 FDTF4 和 GF4 标志
位 11	DTERRFC3	0x0	rw1c	清除通道 3 的数据传输错误标志 (Data transfer error flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF3 标志
位 10	HDTFC3	0x0	rw1c	清除通道 3 的半数据传输标志 (Half data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 HDTF3 标志
位 9	FDTFC3	0x0	rw1c	清除通道 3 的数据传输完成标志 (Full data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 FDTF3 标志
位 8	GFC3	0x0	rw1c	清除通道 3 的全局中断标志 (Global flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF3、HDTF3 FDTF3 和 GF3 标志
位 7	DTERRFC2	0x0	rw1c	清除通道 2 的数据传输错误标志 (Data transfer error flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF2 标志
位 6	HDTFC2	0x0	rw1c	清除通道 2 的半数据传输标志 (Half data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 HDTF2 标志
位 5	FDTFC2	0x0	rw1c	清除通道 2 的数据传输完成标志 (Full data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 FDTF2 标志
位 4	GFC2	0x0	rw1c	清除通道 2 的全局中断标志 (Global flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF2、HDTF2 FDTF2 和 GF2 标志
位 3	DTERRFC1	0x0	rw1c	清除通道 1 的数据传输错误标志 (Data transfer error flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF1 标志
位 2	HDTFC1	0x0	rw1c	清除通道 1 的半数据传输标志 (Half data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 HDTF1 标志
位 1	FDTFC1	0x0	rw1c	清除通道 1 的数据传输完成标志 (Full data transfer flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 FDTF1 标志

位 0	GFC1	0x0	rw1c	清除通道 1 的全局中断标志 (Global flag clear) 0: 无效 1: 清除 DMA_STS 寄存器中 DTERRF1、HDTF1、FDTF1 和 GF1 标志
-----	------	-----	------	--

9.5.3 DMA通道x配置寄存器 (DMA_CxCTRL) (x = 1…7)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 15	保留	0x00000	resd	保持默认值。
位 14	M2M	0x0	rw	存储器到存储器模式 (Memory to memory mode) 0: 关闭 1: 开启
位 13: 12	CHPL	0x0	rw	通道优先级 (Channel preemptive level) 00: 低优先级 01: 中优先级 10: 高优先级 11: 最高优先级
位 11: 10	MWIDTH	0x0	rw	存储器数据宽度 (Memory data bit width) 00: 8 bit 位宽 01: 16 bit 位宽 10: 32 bit 位宽 11: 保留
位 9: 8	PWIDTH	0x0	rw	外设数据宽度 (Peripheral data bit width) 00: 8 bit 位宽 01: 16 bit 位宽 10: 32 bit 位宽 11: 保留
位 7	MINCM	0x0	rw	存储器地址递增模式 (Memory address increment mode) 0: 关闭 1: 开启
位 6	PINCM	0x0	rw	外设地址递增模式 (Peripheral address increment mode) 0: 关闭 1: 开启
位 5	LM	0x0	rw	循环模式 (Loop mode) 0: 关闭 1: 开启
位 4	DTD	0x0	rw	数据传输方向 (Data transfer direction) 0: 外设为源 1: 存储器为源
位 3	DTERRIEN	0x0	rw	允许数据传输错误中断 (Data transfer error interrupt enable) 0: 禁止数据传输错误中断 1: 允许数据传输错误中断
位 2	HDTIEN	0x0	rw	允许半数据传输中断 (Half data transfer interrupt enable) 0: 禁止半数据传输中断 1: 允许半数据传输中断
位 1	FDTIEN	0x0	rw	允许数据传输完成中断 (Full data transfer interrupt enable) 0: 禁止数据传输完成中断 1: 允许数据传输完成中断
位 0	CHEN	0x0	rw	通道使能 (Channel enable) 0: 关闭 1: 开启

9.5.4 DMA通道x数据传输量寄存器 (DMA_CxDTCNT) (x = 1…7)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 0	CNT	0x0000 0000	rw	DMA 通道数据传输个数 (Number of data to transfer) DMA 通道传输数据个数范围为 0x0~0xFFFFFFFF，在更改 DMA 通道传输数据个数时需要确保对应通道的 CHEN 位为 0，否则无法写入；DMA 控制器每传输完一笔数据，此值会硬件减 1。 注：此寄存器为传输数据个数，不是传输数据量大小；传输数据量大小需要根据数据宽度换算得到。

9.5.5 DMA通道x外设地址寄存器 (DMA_CxPADDR) (x = 1…7)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 0	PADDR	0x0000 0000	rw	外设端基地址 (Peripheral base address) 外设数据寄存器的基地址，作为数据传输的源或目标。

9.5.6 DMA通道x存储器地址寄存器 (DMA_CxMADDR) (x = 1…7)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 0	MADDR	0x0000 0000	rw	储存器端基地址 (Memory base address) 储存器地址作为数据传输的源或目标。

9.5.7 DMAMUX选择寄存器 (DMA_MUXSEL)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
31: 1	保留	0x0000 0000	resd	保持默认值。
位 0	TBL_SEL	0x0	rw	多路复用器表选择 (multiplexer table select) 0x1: 弹性映射表

9.5.8 DMAMUX通道x控制寄存器 (DMA_MUXCxCTRL) (x = 1...7)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31:25	保留	0x00	resd	保持默认值。
位 28: 24	SYNCSEL	0x00	rw	同步标识 (Synchronization select)
位 23: 19	REQCNT	0x00	rw	DMA 请求数 (Request count) 定义同步事件之后向 DMA 控制器的 DMA 请求数，和/或生成输出事件之前的 DMA 请求数。 仅当 SYNCEN 和 EVTGEN 位均为 LOW 时，才写入字段保留。
位 18: 17	SYNCPOL	0x0	rw	同步极性 (Synchronization polarity) 定义所选同步输入的边缘极性。 0x0: 无事件 0x1: 上升沿 0x2: 下降沿 0x3: 上升沿和下降沿
位 16	SYNCEN	0x0	rw	同步启用 (Synchronization enable) 0: 禁用同步 1: 同步使能
位 15: 10	保留	0x00	resd	保持默认值。
位 9	EVTGEN	0x0		启用事件 (Event generate enable) 0: 禁用事件生成 1: 启用事件生成
位 8	SYNCOVEN	0x0		同步溢出中断启用 (Synchronization overrun interrupt enable) 0: 禁止中断 1: 中断使能
位 7	保留	0x0	resd	保持默认值。
位 6: 0	REQSEL	0x00		DMA 请求标识 (DMA Request select) 选择输入的 DMA 请求。请参考 DMAMUX 表以了解资源的多路复用器输入的分配。

9.5.9 DMAMUX生成器x控制寄存器 (DMA_MUXGxCTRL) (x = 1…4)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 24	保留	0x00	resd	保持默认值。
位 23: 19	GREQCNT	0x00	rw	DMA 请求生成数 (Request generation count) 定义触发事件后将生成的 DMA 请求 (GNBREQ + 1) 的数量。 仅当禁用 GEN 位时，才写入字段保留。
位 18: 17	GPOL	0x0	rw	DMA 请求生成器触发极性 (Generation polarity) 定义所选触发输入的边缘极性。 0x0: 无事件 0x1: 上升沿 0x2: 下降沿 0x3: 上升沿和下降沿
位 16	GEN	0x0	rw	DMA 请求生成器 (DMA request generation enable) 0: 禁用 DMA 请求生成器 1: 启用 DMA 请求生成器
位 15: 9	保留	0x00	resd	保持默认值。
位 8	TRGOVIEN	0x0	rw	触发溢出中断使能 (Trigger overrun interrupt enable) 0: 禁止中断 1: 中断使能
位 7: 5	保留	0x0	resd	保持默认值。
位 4: 0	SIGSEL	0x00	rw	信号识别 (Signal select) 选择用于 DMA 请求生成器的 DMA 触发输入。

9.5.10 DMAMUX通道同步状态寄存器 (DMA_MUXSYNCSTS)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 7	保留	0x0000 00	resd	保持默认值。
位 6: 0	SYNCOVF	0x00	ro	同步溢出中断标志 (Synchronization overrun interrupt flag) 当 DMA 请求计数器低于 REQCNT 时发生，如果发生新的同步事件，将设置该标志。

9.5.11 DMAMUX通道中断清除标志寄存器 (DMA_MUXSYNCCLR)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 7	保留	0x0000 00	resd	保持默认值。
位 6: 0	SYNCOVFC	0x00	rw1c	清除同步溢出中断标志 (Synchronization overrun interrupt flag clear) 在每个位写入 1 以清除 MUXSYNCSTS 寄存器中的相应溢出标志 SYNCOVF。

9.5.12 DMAMUX发生器中断状态寄存器（DMA_MUXGSTS）

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 4	保留	0x0000 000	resd	保持默认值。
位 3: 0	TRGOVF	0x00	ro	触发溢出中断标志（Trigger overrun interrupt flag） 当 DMA 请求计数器低于 GREQCNT 时，如果发生新的触发事件，将设置该标志。

9.5.13 DMAMUX发生器中断清除标志寄存器（DMA_MUXGCLR）

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 4	保留	0x0000 000	resd	保持默认值。
位 3: 0	TRGOVFC	0x00	rw1c	清除触发器溢出中断标志（Trigger overrun interrupt flag clear） 在每个位写入 1，以清除 DMA_MUXGSTS 寄存器中的相应溢出标志 TRGOVF。

10 CRC 计算单元 (CRC)

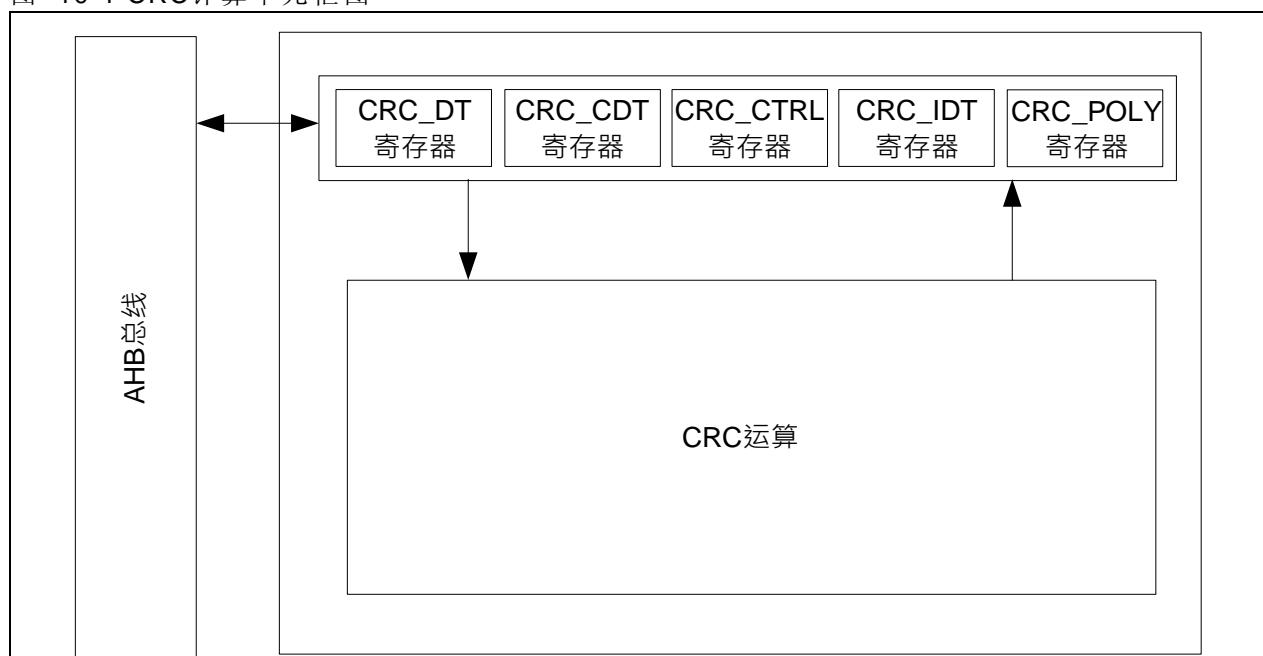
10.1 CRC介绍

CRC计算单元是一个独立的具备CRC计算功能的外设，CRC计算单元采用CRC32/MPEG-2。

用户可以通过软件编程配置CRC_CTRL选择是否进行输出数据翻转(全字翻转, REVID=1)或输入数据翻转(字节翻转, REVID=01; 半字翻转, REVID=10; 全字翻转, REVID=11), CRC计算单元还提供初始化功能, 每次RESET操作后, CRC计算单元会将CRC_IDT中的值搬入CRC_DT。CRC_POLY寄存器可让用户软件编程不同的生成多项式系数, 并通过CRC_CTRL的POLY_SIZE将生成多项式的大小配置为7/8/16/32位。

用户通过写和读CRC_DT寄存器的方式, 写入想要进行计算的值, 读出计算的结果, 注意每次的计算结果为此前计算结果和当前待计算值的组合。

图 10-1 CRC计算单元框图



CRC 主要特性:

- 预设采用 CRC-32 标准
- 可编程生成多项式
- 一次 CRC 计算需要 4 个 HCLK
- 输入输出数据格式可翻转
- 待计算值的写入和计算结果的读出都通过写和读 CRC_DT 实现
- 配置 CRC_IDT 写入初始化值, 在每次 CRC 复位后该值会加载到 CRC_DT

10.2 CRC功能说明

CRC 的计算原理是将输入数据做为被除数，与作为除数的生成多项式进行模二除法，得到的余数即为 CRC 值。

CRC 运算流程

- 输入翻转，即数据输入后，先依据 CRC_CTRL 的 REVID 值进行输入数据翻转。
- 初始化，首次计算会与 CRC_IDT 设定的初始值做 XOR。若非首次计算，则初始值为上次的计算结果。
- CRC 计算，与生成多项式(0x4C11DB7)进行模二除法，所得余数为 CRC 值。
- 输出翻转，依据 CRC_CTRL 的 REVOD 决定是否将 CRC 值执行全字翻转后再输出。
- 对结果进行 XOR 运算，结果异或值固定为 0x0000 0000。

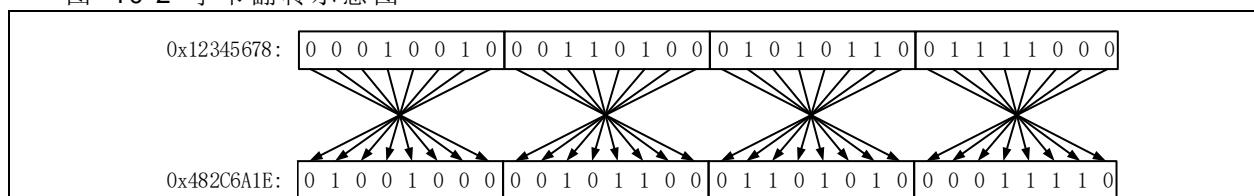
CRC-32/MPEG-2 参数说明

- 生成多项式: 0x4C11DB7，
即 $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$
- 初始值: 0xFFFF FFFF，目的为避免待测数据为 1 字节 0x00 和多字节 0x00 的结果相同。
- 结果异或值: 0x0000 0000，此值表示不对 CRC 结果再进行一次 XOR 运算。

翻转功能说明

- 选择以字节翻转，则 8bit 为一组，组内排列顺序颠倒。如下图所示，若原数据为 0x12345678，翻转后为 0x482C6A1E。
- 选择以半字翻转，则 16bit 为一组，组内排列顺序颠倒。
- 选择以字翻转，则 32bit 为一组，组内排列顺序颠倒。

图 10-2 字节翻转示意图



10.3 CRC寄存器

表 10-1 CRC计算单元寄存器映像

寄存器简称	基址偏移量	复位值
CRC_DT	0x00	0xFFFF FFFF
CRC_CDT	0x04	0x0000 0000
CRC_CTRL	0x08	0x0000 0000
CRC_IDT	0x10	0xFFFF FFFF
CRC_POLY	0x14	0x04C1 1DB7

10.3.1 数据寄存器 (CRC_DT)

域	简称	复位值	类型	功能
位 31: 0	DT	0xFFFF FFFF	rw	数据寄存器位 (Data value) 写入 CRC 计算器的新数据时, 作为输入寄存器读取时返回 CRC 计算的结果。

10.3.2 通用数据寄存器 (CRC_CDT)

域	简称	复位值	类型	功能
位 31: 8	保留	0x000000	resd	保持默认值。
位 7: 0	CDT	0x00	rw	通用 8 位数据寄存器位 (Common 8-bit data value) 可用于临时存放 1 字节的数据。寄存器 CRC_CTRL 的 RST 位产生的 CRC 复位对本寄存器没有影响。

10.3.3 控制寄存器 (CRC_CTRL)

域	简称	复位值	类型	功能
位 31: 8	保留	0x000000	resd	保持默认值。
位 7	REVOD	0x0	resd	输出数据翻转 (Reverse output data) 由软件置起或清零。该位控制是否翻转输出数据。 0: 不翻转; 1: 全字翻转。
位 6: 5	REVID	0x0	rw	输入数据翻转 (Reverse input data) 由软件置起或清零。该位控制如何翻转输入数据。 00: 不翻转; 01: 字节翻转; 10: 半字翻转; 11: 全字翻转。
位 4: 3	POLY_SIZE	0x0	rw	生成多项式位宽(Polynomial size) 该位控制生成多项式的位宽大小, 与 CRC_POLY 寄存器相配合。 00: 位宽为 32 位 01: 位宽为 16 位 10: 位宽为 8 位 11: 位宽为 7 位
位 2: 1	保留	0x0	resd	保持默认值。
位 0	RST	0x0	wo	RESET 位 (Reset CRC calculation unit) 由软件置起, 由硬件自动清零。复位 CRC 计算单元, 设置数据寄存器为 0xFFFF FFFF。 0: 无作用; 1: 复位。

10.3.4 初始化寄存器 (CRC_IDT)

域	简称	复位值	类型	功能
位 31: 0	IDT	0xFFFF FFFF	rw	初始化数据寄存器 (Initial data value) 当 CRC_CTRL 寄存器的 RST 位产生的 CRC 复位时, 初始化寄存器中的数值将作为 CRC_DT 寄存器的初始值写入。

10.3.5 生成多项式系数寄存器 (CRC_POLY)

域	简称	复位值	类型	功能
位 31: 0	POLY	0x04C1 1DB7	rw	生成多项式系数寄存器 (polynomial coefficient) 生成多项式为 CRC 计算中的除数, 预设使用 CRC32 参数模型, 所以系数为 0x4C11DB7。用户亦可自行编程该生成多项式。

11 I²C 接口

11.1 I²C 简介

I²C总线接口处理微控制器和串行I²C总线之间的通信，支持主机和从机模式，最大通信速度为1Mbit/s(增强快速模式)。

11.2 I²C的主要特点

- I²C 总线
 - 主机和从机模式
 - 多主机功能
 - 标准模式（standard mode, 最高 100kHz）、快速模式（fast mode, 最高 400kHz）和增强快速模式（fast mode plus, 最高 1 MHz）
 - 7-bit 和 10-bit 地址模式
 - 两组 7 位从地址（2 个地址，其中一个可屏蔽）
 - 广播呼叫
 - 可编程数据建立和保持时间
 - 时钟延展功能
- 支持 DMA 功能
- 可编程数字噪声滤波器
- 支持 SMBus 2.0 版协议
 - PEC 产生和检查
 - 命令和数据的应答控制
 - ARP(地址解析协议)
 - 主机功能
 - 设备功能
 - SMBus 提醒功能
 - 超时检测
 - 空闲检测
- PMBus

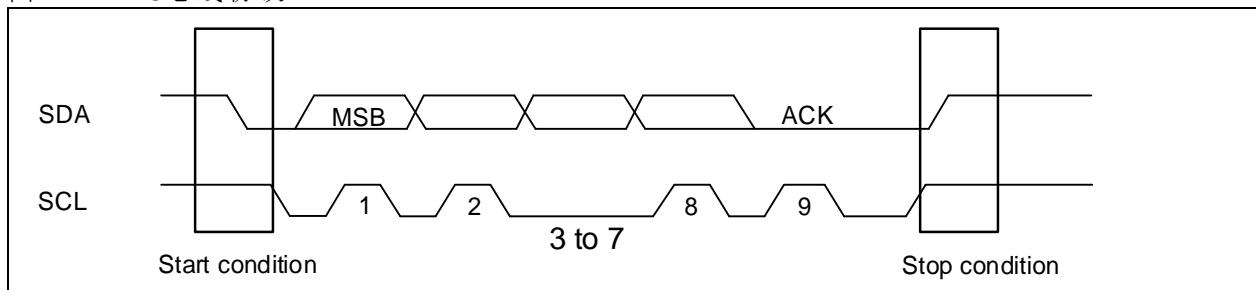
11.3 I²C总线特性

I²C 总线是由数据线 SDA 和时钟线 SCL 构成，在标准模式下通信速度可达到 100kHz，快速模式下则可以达到 400kHz，增强快速模式可达到 1MHz。一帧数据传输从起始条件开始，在停止条件后停止，在收到起始条件后总线被认为是繁忙的，当收到停止条件后，总线被认为再次空闲。

起始条件：SCL 为高电平时，SDA 由高电平变为低电平；

停止条件：SCL 为高电平时，SDA 由低电平变为高电平。

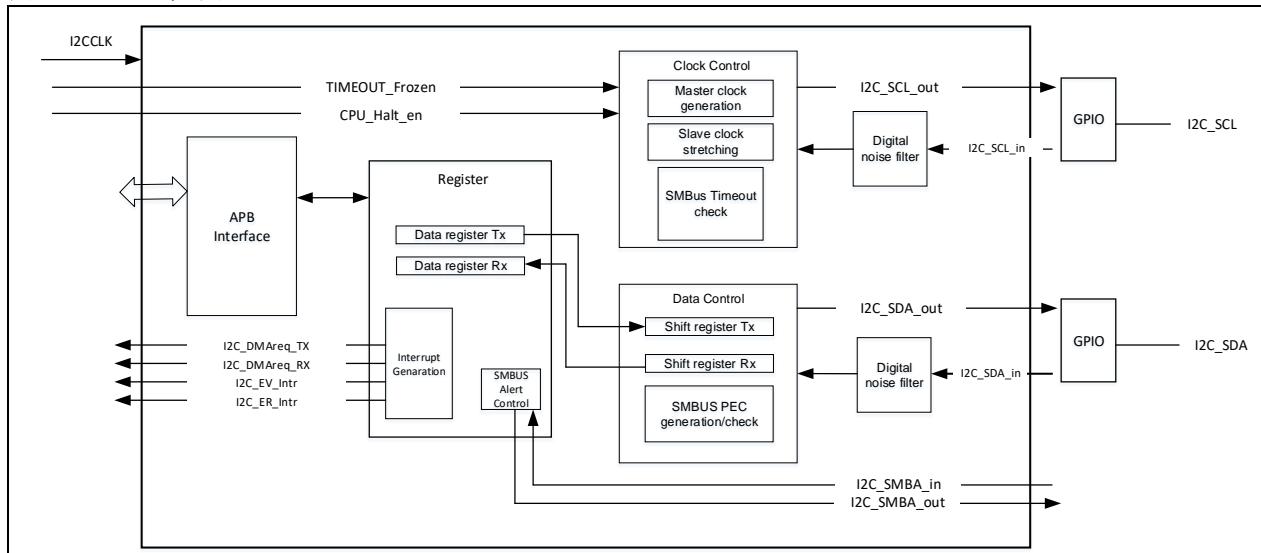
图 11-1 I²C总线协议



11.4 I²C 接口

I²C 接口的功能框图示于下图：

图 11-2 I²C 框图



1. 接口工作模式

I²C 总线接口可以工作在主机模式与从机模式，并且可以相互切换。默认情况下处于从机模式，当产生了一个起始信号后，I²C 总线接口切换成主模式，当数据传输完成之后，也就是停止条件产生了之后，I²C 总线接口自动返回为从机模式。

2. 通信流程

- 主机模式通信流程：

1. 产生起始条件
2. 发送地址
3. 发送或接收数据
4. 产生停止条件
5. 通信结束

- 从机模式通信流程：

1. 等待地址匹配
2. 发送或接收数据
3. 等待停止条件产生
4. 通信结束

3. 数字滤波功能

SCL 和 SDA 总线上均有数字滤波器，数字滤波器可以有效降低总线上的干扰，通过配置 I²C_CTRL1 的 DFLT[3: 0]位（范围 0~15），可以启用数字滤波器，滤波时间为 $DFLT \times T_{I^2C_CLK}$ ，当 I²C 启用时不允许更改数字滤波器的配置。

4. 地址控制

主机和从机都支持 7 位和 10 位地址模式

从机地址模式：

- 7 位地址模式 (ADDR1MODE=0)
 - 单地址模式 ADDR1EN=1, ADDR2EN=0：此时只匹配OADDR1；
 - 双地址模式 ADDR1EN=1, ADDR2EN=1：此时匹配OADDR1和OADDR2。
- 10 位地址模式 (ADDR1MODE=1)
 - 只支持单地址模式 ADDR1EN=1, ADDR2EN=0，匹配OADDR1

从机地址屏蔽功能

从机地址 2 (OADDR2) 支持地址屏蔽功能，通过设置 ADDR2MASK[2: 0]启用地址屏蔽功能

- 0: 匹配地址位[7: 1];
- 1: 只匹配地址位[7: 2];
- 2: 只匹配地址位[7: 3];
- 3: 只匹配地址位[7: 4];
- 4: 只匹配地址位[7: 5];
- 5: 只匹配地址位[7: 6];
- 6: 只匹配地址位[7];
- 7: 所有非I²C保留地址都会响应。

从机特殊地址支持:

- 广播地址 (0b0000000x) : 当 GCAEN=1 时该地址启用;
- SMBus 设备默认地址 (0b1100001x) : 当在 SMBus 设备模式下 (DEVADDREN = 1) 该地址启用，该地址用于 SMBus 地址解析协议;
- SMBus 主机默认地址 (0b0001000x) : 当在 SMBus 主机模式下 (HADDREN = 1) 该地址启用，该地址用于 SMBus 主机通知协议;
- SMBus 提醒地址 (0b0001100x) : 当在 SMBus 在设备模式下，并且拉低 SMBbus ALERT 引脚 (SMBALERT = 1) 下该地址启用，该地址用于 SMBus 提醒响应协议。
关于 SMBus 协议更详细的信息请参考 SMBus2.0 协议。

从机地址匹配流程:

- 收到起始条件
- 匹配地址
- 若地址成功匹配，从机回一个 ACK
- 此时 ADDR1 置 1, SDIR 指示传输方向
 - 如果 SDIR=0 从机进入接收模式，开始接收数据
 - 如果 SDIR=1 从机进入发送模式，开始发送数据

5. 时钟延展功能

在默认情况下，从机的时钟延展功能是打开的，即 I2C_CTRL1 的 STRETCH 位为 0，从机可以根据需要将 SCL 信号拉低，延展 SCL 信号的低电平时间以便执行软件的操作，如果主机不支持时钟延展功能，则 I2C_CTRL1 的 STRETCH 位需配置为 1。需要注意的是 I²C 从机的时钟延展模式必须在使能 I²C 外设之前配置。

从机时钟延展

I²C 从机在以下情况下延展 SCL 时钟:

- 地址接收阶段：从机接收到的地址与启用的本机地址匹配 (I2C_STS 的 ADDR1=1) 时会将 SCL 线拉低，直到软件将 I2C_CLR 的 ADDRC 位置 1 清掉 ADDR1 时释放时钟延展；
- 数据接收阶段：I2C_RXDT 寄存器的数据尚未被读取，移位寄存器也接收到一个新的数据，这时会将 SCL 线拉低直到 I2C_RXDT 寄存器的数据被读取；
- 数据发送阶段：ADDR1 被清除后未写入数据时，I2C_STS 的 TDBE= 1，这时会将 SCL 线拉低直到数据写入 I2C_TXDT；
- 数据发送阶段：上一笔的数据传输已完成，尚未有新的数据写到 I2C_TXDT，这时会将 SCL 线拉低直到数据写入 I2C_TXDT；
- 当启用从机字节控制模式(I2C_CTRL1 的 SCTRL 为 1)且 I2C_CTRL2 的 RLDE1 位为 1，如果 TCRLD = 1，代表此时最后一个数据字节已经传输完成，直到向 I2C_CTRL2 的 CNT 位写入一个非零值，硬件自动将 TCRLD 清 0，从机释放 SCL 线。

从机不带时钟延展

当 I2C_CTRL1 的 STRETCH 位为 1 时，I²C 从机不会延展 SCL 信号，需要特别注意下列情况：

- 地址接收阶段：从机接收到的地址与启用的本机地址匹配 (I2C_STS 的 ADDR1=1) 时 SCL 时钟不会延展；
- 数据接收阶段：下一个字节的 ACK 发生前，尚未将数据从 I2C_RXDT 寄存器读走，会发生上溢，I2C_STS 的 OUF 位会被置 1；
- 数据发送阶段：上一笔的数据传输已完成，尚未有新的数据写到 I2C_TXDT，会发生下溢，I2C_STS 的 OUF 位会被置 1。

11.4.1 I²C时序控制

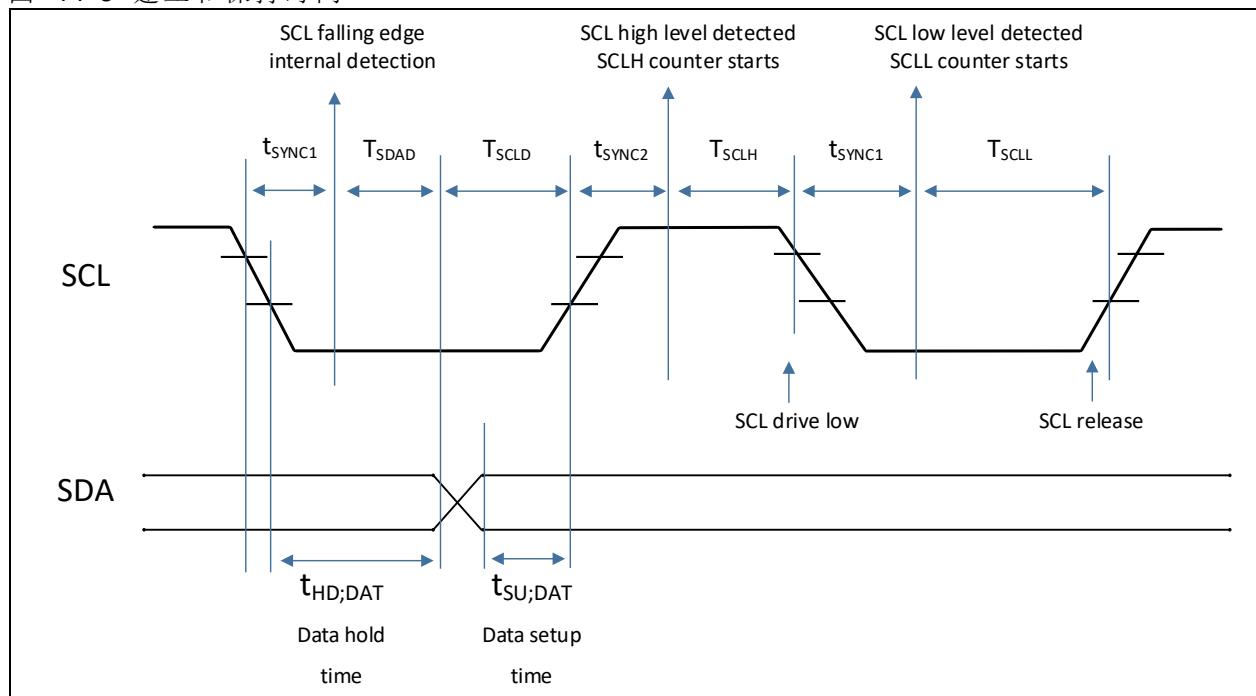
I²C 内核由 I²C_CLK 提供时钟，I²C_CLK 由 PCLK1 提供，PCLK1 周期需满足小于 4/3 SCL 周期。

通过 I²C_CLKCTRL 寄存器的各个位，配置各个时序。

- DIV[7: 0]: I²C时钟分频；
- SDAD[3: 0]: 数据保持时间 ($t_{HD;DAT}$)；
- SCLD[3: 0]: 数据建立时间 ($t_{SU;DAT}$)；
- SCLH[7: 0]: SCL高电平时间；
- SCLL[7: 0]: SCL低电平时间。

注：当 I²C 启用时不允许更改时序配置。

图 11-3 建立和保持时间



通过配置 I²C_CLKCTRL 的 DIV[7: 0]、SDAD[3: 0]、SCLD[3: 0]，可以灵活的配置数据保持时间 ($t_{HD;DAT}$) 和数据建立时间 ($t_{SU;DAT}$)

- 数据保持时间 ($t_{HD;DAT}$)：SCL 下降沿到 SDA 输出的延时

$$t_{HD;DAT} = T_{SDAD} + t_{SYNC}$$

$$T_{SDAD} = SDAD \times (DIV + 1) \times T_{I2C_CLK}$$

$$t_{SYNC} = (DFLT + 3) \times T_{I2C_CLK} - t_f$$

t_{SYNC} 由以下几部分组成

- SCL下降沿时间 t_f
- 数字滤波器的输入延迟 ($DFLT \times T_{I2C_CLK}$)
- SCL和I²C_CLK的同步延时 (2~3个I²C_CLK)

- 数据建立时间 ($t_{SU;DAT}$)：SDA 输出后到 SCL 上升沿的延时

$$t_{SU;DAT} = SCLD \times (DIV+1) \times T_{I2C_CLK} - t_r$$

在主机模式下，通过配置 I²C_CLKCTRL 的 DIV[7: 0]、SCLH[7: 0]、SCLL[7: 0]，可以灵活的配置 SCL 高电平宽度和低电平宽度

低电平控制：当检测到 SCL 总线为低电平时，内部 SCLL 计数器开始计数，当计数值达到 SCLL 值时，释放 SCL 线，SCL 线变为高电平。

高电平控制：当检测到 SCL 总线为高电平时，内部 SCLH 计数器开始计数，当计数值达到 SCLH 值时，拉低 SCL 线，SCL 线变为低电平，当在高电平期间，如果被外部总线拉低，那么内部 SCLH 计数器停止计数，并开始低电平计数，这为时钟同步提供了条件。

- 高电平宽度：

$$t_{HIGH} = T_{SCLH} = (SCLH + 1) \times (DIV + 1) \times T_{I2C_CLK}$$

- 低电平宽度：
 $t_{LOW} = T_{SCL} \times (SCLL + 1) \times (DIV + 1) \times T_{I2C_CLK}$

表 11-1 I²C时间规范

参数		标准模式 Standard mode		快速模式 Fast mode		增强快速模式 Fast mode plus		SMBus 模式	
		最小值	最大值	最小值	最大值	最小值	最大值	最小值	最大值
f _{SCL} (kHz)	SCL 时钟频率	100		400		1000		100	
t _{LOW} (us)	SCL 时钟低电平	4.7		1.3		0.5		4.7	
t _{HIGH} (us)	SCL 时钟高电平	4.0		0.6		0.26		4.0	50
t _{HOLD:DAT} (us)	数据保持时间	0		0	0.9	0	0.45	300	
t _{SETUP:DAT} (ns)	数据建立时间	250		100		50		250	
t _r (ns)	SCL、SDA 上升沿	1000		300		120		1000	
t _f (ns)	SCL、SDA 下降沿	300		300		120		300	

11.4.2 数据传输管理

I²C 内部有一个管理数据传输的计数器，用来管理通信流程，主要应用于以下地方：

- NACK发送：主机接收模式；
- STOP发送：主机接收/发送模式；
- RESTART产生：主机接收/发送模式；
- ACK控制：从机模式（SMBus下）；
- PEC发送/接收：主机/从机模式。

通常字节传输管理计数器（I2C_CTRL2 的 CNT[7:0]位配置）应用于主模式，在从机模式下是关闭的，只有在 SMBus 模式下，从机为了对每一个字节进行 ACK 的控制和 PEC 的接收，才会使用此计数器，在 SMBus 模式下从机可以通过 I2C_CTRL1 的 SCTRL 位来启用字节计数器功能。

主机字节控制

I2C_CTRL2 的 CNT[7:0]用于配置发送字节个数，配置范围为 1~255，当一次传输的数据超过 255 个以上需要将 I2C_CTRL2 的 RLDEN 位置 1，使能重载模式，所以发送数据个数分为≤255 字节和>255 字节两种情况：

- ≤255字节，例如要传输数据个数为100个字节
 - 步骤 1：配置 RLDEN=0，关闭重载模式；
 - 步骤 2：配置 CNT[7:0]=100；
- >255字节，例如要传输数据个数为600个字节
 - 步骤 1：配置 RLDEN=1，使能重载模式；
 - 步骤 2：配置 CNT[7:0]=255，此时还剩下 600-255=345 字节；
 - 步骤 3：当这 255 字节传输完成后，I2C_STS 的 TCRLD 位置 1，然后配置 CNT[7:0]=255，继续传输，此时还剩下 345-255=90；
 - 步骤 4：当这 255 字节传输完成后，I2C_STS 的 TCRLD 位置 1，然后配置 RLDEN=0 禁止重载模式，再配置 CNT[7:0]=90，继续传输。

当是最后一笔数据传输时（重载模式禁止，RLDEN=0），有两种结束数据传输模式

- 自动结束模式（I2C_CTRL2的ASTOPEN=1）
 - 当发送了 CNT[7:0]个字节后，主机自动发送 STOP 条件。
- 软件结束模式（I2C_CTRL2的ASTOPEN=0）
 - 当发送了 CNT[7:0]个字节后，I2C_STS 的 TDC 位置 1，此时 SCL 被拉低，如果使能了 TDCIEN，那么可以产生一个中断，此时软件可以设置 I2C_CTRL2 的 GENSTOP=1 产生一个 STOP 条件，也可以设置 I2C_CTRL2 的 GENSTART=1 产生一个 RESTART 条件，然后软件清除 TDC 标志。

从机字节控制

通过 I2C_CTRL1 的 SCTRL 位来启用从机字节控制功能，该功能可以让从机对接收到的每一个字节进行单独的 ACK/NACK 控制。

- 使用步骤：

- 设置 SCTRL=1，启用从机字节控制功能；
- 在从机地址匹配后 (ADDRF=1)，使能重载模式 (RLDEN=1)，并设置 CNT[7:0]=1；
- 当接收到一个字节后，I2C_STS 的 TCRLD 置 1，从机在 SCL 的第 8 个和第 9 个时钟沿中间拉低 SCL 总线，此时用户读取 RXDT 寄存器，然后根据需要配置 I2C_CTRL2 的 NACKEN 位，来产生一个 ACK 或 NACK；
- 如果产生一个 NACK，那么通信结束；
- 如果产生一个 ACK，通信继续，此时写入 CNT[7:0]=1，硬件自动清除 TCRLD 标志，从机释放 SCL 总线，继续接收下一个字节。

当然 CNT[7:0] 值不仅仅局限于 1，例如想接收 8 个数据，但只想控制第 8 个数据的 ACK/NACK，此时就可以设置 CNT[7:0]=8，那么从机就会连续接收 7 个数据，并且回复 ACK，在第 8 个数据接收完后，从机拉低总线，然后同上述步骤，决定 ACK/NACK 的回复。

需要注意的是：要使用从机字节控制模式必须要使能时钟延展 (I2C_CTRL1 的 STRETCH 位=0)。

表 11-2 I²C 配置表

功能	RLDEN	ASTOPEN	SCTRL
主机发送/接收 RESTART	0	0	×
主机发送/接收 STOP	0	1	×
从机接收（每个字节 ACK/NACK 控制）	1	×	1
从机发送/接收（所有字节响应 ACK）	×	×	0

11.4.3 I²C 主机通信流程

1. I²C 时钟初始化（配置 I2C_CLKCTRL 寄存器）

- I²C 时钟分频：DIV[7: 0]
- 数据保持时间 (t_{HLD;DAT})：SDAD[3: 0]
- 数据建立时间 (t_{SU;DAT})：设置 SCLD[3: 0]
- SCL 高电平时间：设置 SCLH[7: 0]
- SCL 低电平时间：设置 SCLL[7: 0]

该寄存器的配置可以使用 Artery_I2C_Timing_Configuration 时钟配置工具计算。

2. 设置传输字节数

- ≤255 字节
 - 配置 I2C_CTRL2 的 RLDEN=0，关闭重载模式
 - 配置 I2C_CTRL2 的 CNT[7:0]=N
- >255 字节
 - 配置 I2C_CTRL2 的 RLDEN=1，使能重载模式
 - 配置 I2C_CTRL2 的 CNT[7:0]=255
 - 剩余传输字节数 N=N-255

3. 设置传输结束模式

- ASTOPEN=0：软件结束模式，当数据传输完成后，I2C_STS 的 TDC 标志置 1，软件设置 GENSTOP=1 或者 GENSTART=1，发送 STOP 条件或者 START 条件。
- ASTOPEN=1：自动结束模式，当数据传输完成后，自动发送 STOP 条件。

4. 设置从机地址

- 设置寻址的从机地址值 (I2C_CTRL2 的 SADDR)
- 设置从机地址模式 (I2C_CTRL2 的 ADDR10)：
 - ADDR10=0：7 位地址模式
 - ADDR10=1：10 位地址模式

5. 设置传输方向 (I2C_CTRL2 的 DIR)

- DIR=0: 主机接收数据
- DIR=1: 主机发送数据

6. 开始传输

设置 I2C_CTRL2 的 GENSTART=1, 主机开始在总线上发送 START 条件和从机地址, 当从机响应了地址之后, 主机的 I2C_STS 的 ADDRF=1, 设置 I2C_CLR 的 ADDRC=1 清除 ADDRF 标志后, 开始数据传输。

7. 主机发送数据

1. I2C_TXDT 数据寄存器为空, 移位寄存器为空, I2C_STS 的 TDIS=1;
2. 向 TXDT 数据寄存器写入数据 1, 然后数据将被立即放进移位寄存器;
3. 此时 TXDT 数据寄存器为空, TDIS 又置 1;
4. 向 TXDT 数据寄存器写入数据 2, TDIS 被清 0;
5. 重复 2、3 步骤直到发送 CNT[7:0] 个数据;
6. 如果此时 I2C_STS 的 TCRLD=1 (重载模式), 分为以下两种情况:
剩余字节数 N>255: 向 CNT 写入 255, N=N-255, TCRLD 被自动清 0, 传输继续;
剩余字节数 N≤255: 关闭重载模式 (RLDEN=0), 向 CNT 写入 N, TCRLD 被自动清 0, 传输继续。

8. 主机接收数据

1. 当正确匹配了从机地址之后, 主机的 I2C_STS 的 ADDRF=1, 设置 I2C_CLR 的 ADDRC=1 清除 ADDRF 标志后, 开始数据传输;
2. 当收到数据后, RDBF=1, 读取 RXDT 数据寄存器, RDBF 被自动清零;
3. 重复步骤 2 直到接收 CNT[7:0] 个数据;
4. 如果此时 I2C_STS 的 TCRLD=1 (重载模式), 分为以下两种情况:
剩余字节数 N>255: 向 CNT 写入 255, N=N-255, TCRLD 被自动清 0, 传输继续;
剩余字节数 N≤255: 关闭重载模式 (RLDEN=0), 向 CNT 写入 N, TCRLD 被自动清 0, 传输继续。
5. 当在接收到最后一个字节时, 主机会自动发送一个 NACK。

9. 结束时序

- 停止条件产生:
软件结束模式 (ASTOPEN=0): 此时 I2C_STS 的 TDC 置 1, 设置 GENSTOP=1 产生 STOP 条件。
自动结束模式 (ASTOPEN=1): 自动产生 STOP 条件。
- 等待产生 STOP 条件, 当 STOP 条件产生时, I2C_STS 的 STOPF 置 1, 将 I2C_CLR 的 STOPC 写 1, 清除 STOPF 标志, 传输结束。

主机在传输过程中, 如果收到了 NACK, 那么此时 I2C_STS 的 ACKFAIL 置 1, 并自动发送 STOP 条件结束通信, 无论当前是软件结束模式 (ASTOPEN=0) 还是自动结束模式 (ASTOPEN=1)。

主机发送流程

图 11-4 I²C 主机发送流程图

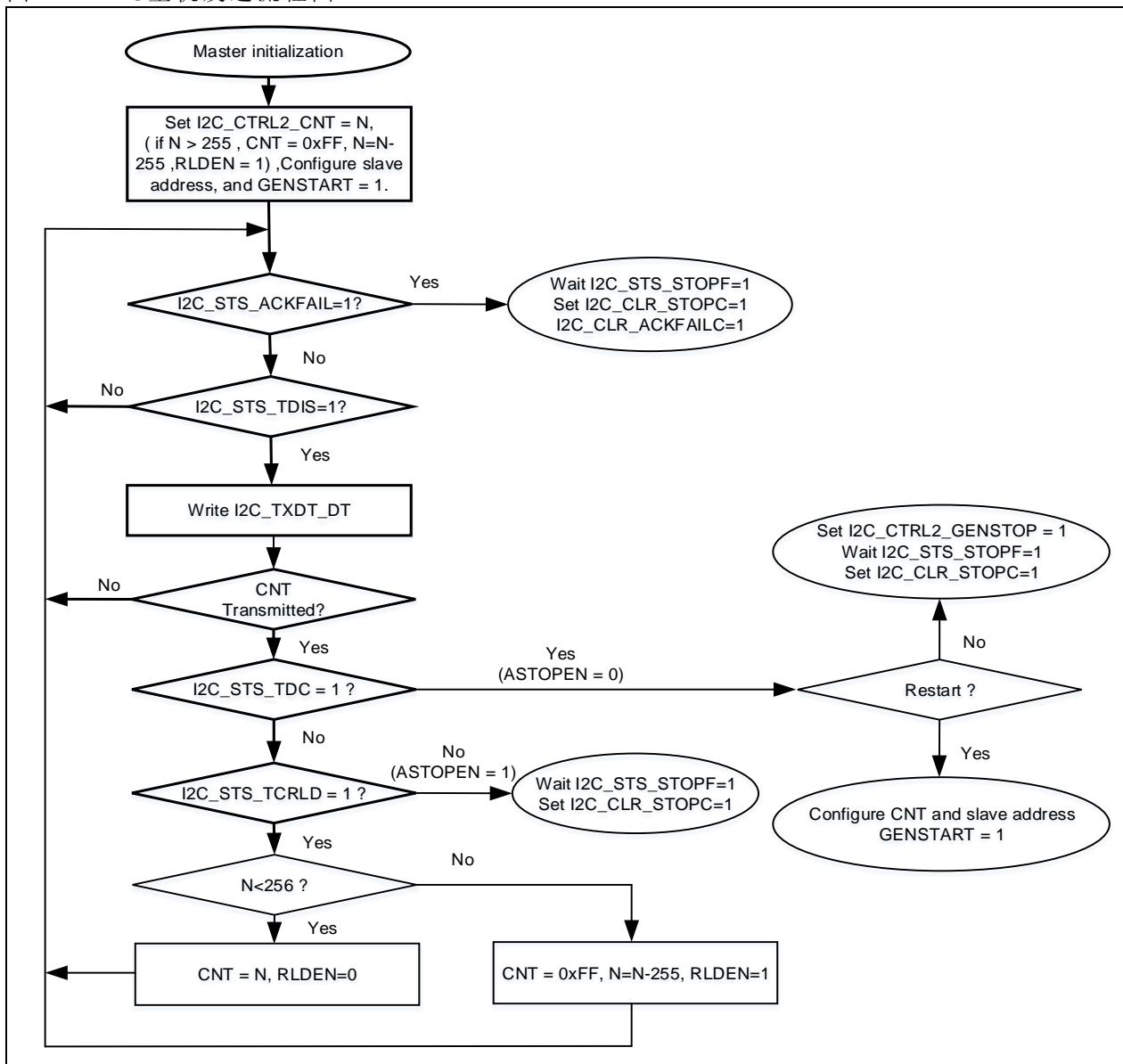
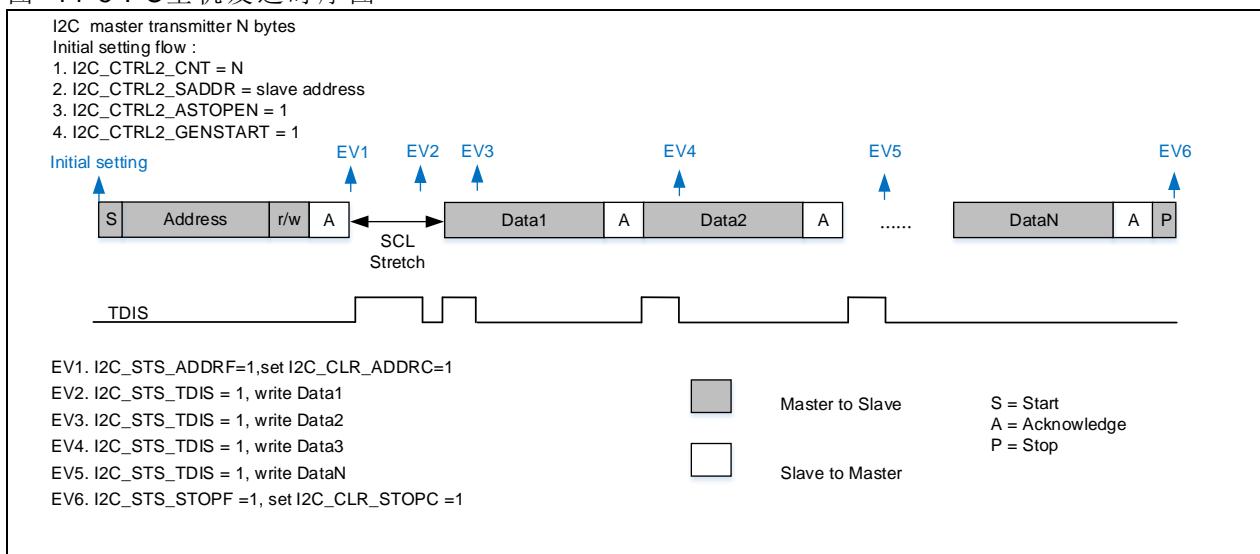


图 11-5 I²C 主机发送时序图



主机接收流程

图 11-6 I²C 主机接收流程图

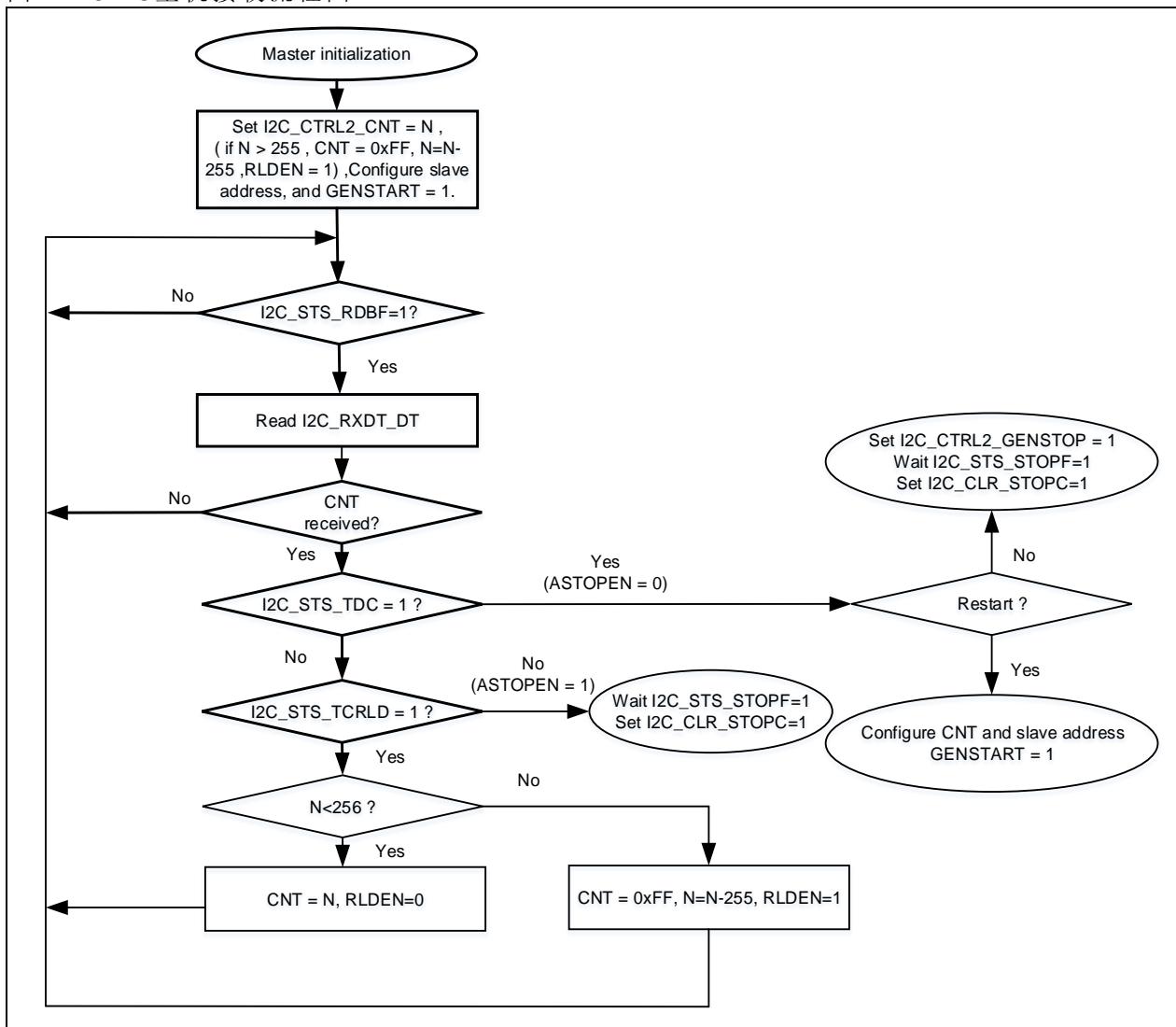
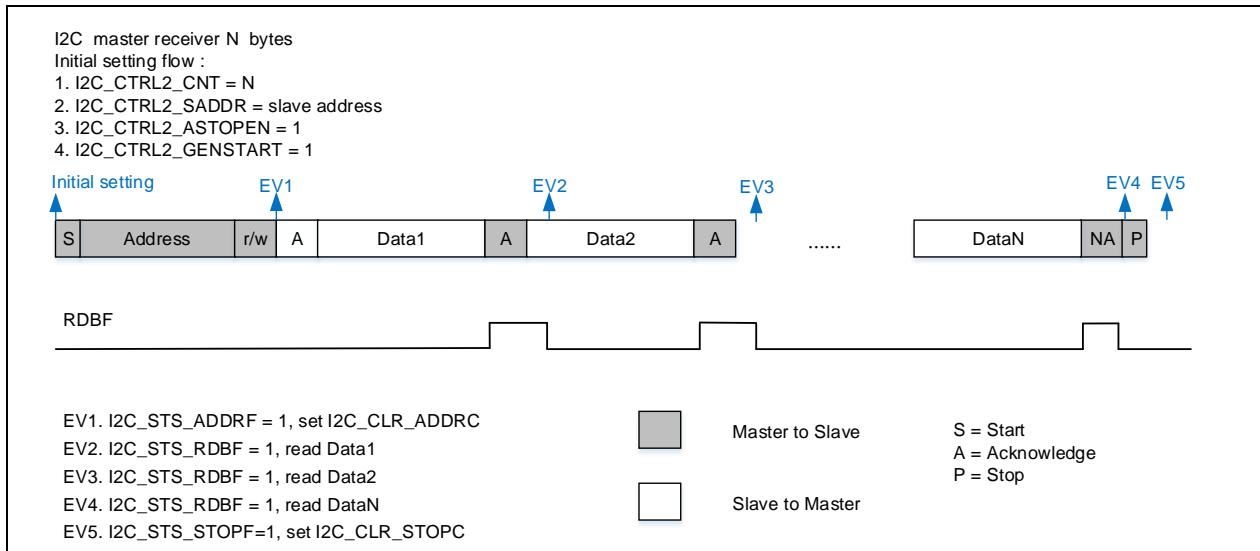


图 11-7 I²C 主机接收时序图



主机特殊传输时序支持

在 10 位地址传输模式下, I2C_CTRL2 的 READH10 用于产生特殊时序, 当 READH10=1 时, 支持如下传输序: 主机先发送数据给从机, 然后再从从机读取数据, 传输时序图如下图所示:

使用方法:

主机在软件结束模式 (ASTOPEN = 0) 下, 发送数据到从机, 当数据发送完成后设置 READH10=0, 然后再从从机接收数据。

图 11-8 10位地址读访问 READH10=1

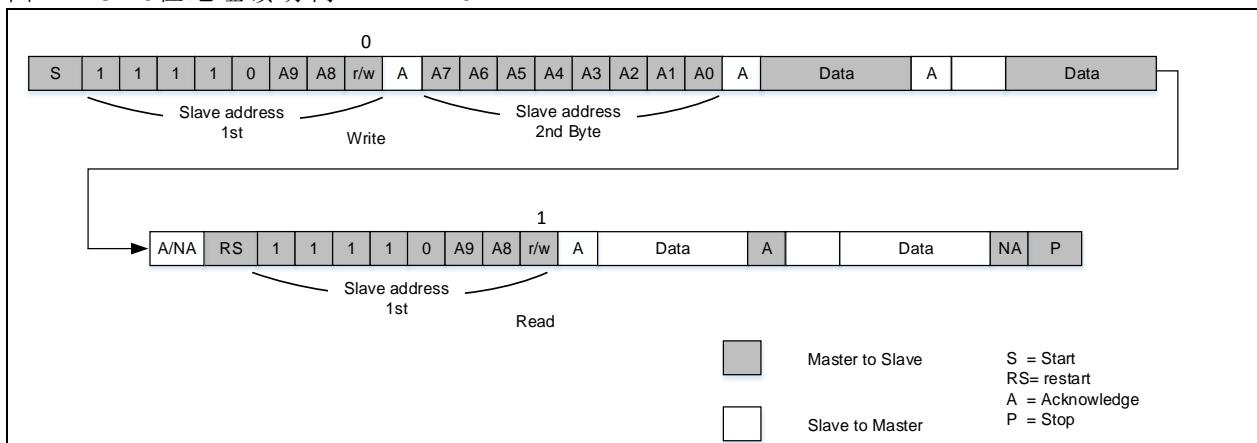
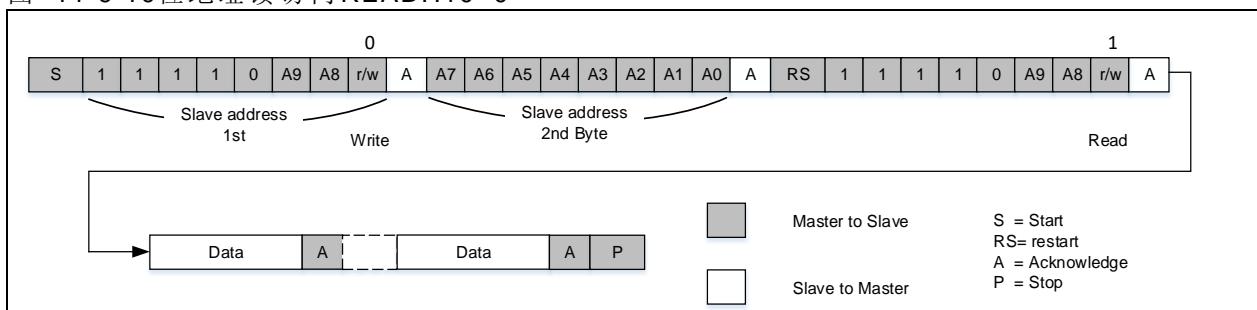


图 11-9 10位地址读访问 READH10=0



11.4.4 I²C从机通信流程

1. I²C时钟初始化（配置I2C_CLKCTRL寄存器）

- I²C 时钟分频: DIV[7: 0]
- 数据保持时间 (t_{HD;DAT}) : SDAD[3: 0]
- 数据建立时间 (t_{SU;DAT}) : 设置 SCLD[3: 0]

该寄存器的配置可以使用 Artery_I2C_Timing_Configuration 时钟配置工具计算。

2. 设置本机地址1

- 设置地址模式:
 - 7 位地址: 设置 I2C_OADDR1 的 ADDR1MODE = 0
 - 10 位地址: 设置 I2C_OADDR1 的 ADDR1MODE = 1
- 设置地址 1: 设置 I2C_OADDR1 的 ADDR1
- 使能地址 1: 设置 I2C_OADDR1 的 ADDR1EN=1

3. 设置本机地址2

- 设置地址 2: 设置 I2C_OADDR2 的 ADDR2
- 设置地址 2 屏蔽位: 设置 I2C_OADDR2 的 ADDR2MASK
- 使能地址 2: 设置 I2C_OADDR2 的 ADDR2EN=1

需要注意的是, 地址 2 只支持 7 位地址模式, 并且可以通过 ADDR2MASK 位来灵活的屏蔽一些地址位, 使从机响应一些特定的地址, 关于 ADDR2MASK 位的详细用法, 请参照章节 14.2。

在只使用一个地址的情况下, 此地址不用配置, 只需要配置地址 1。

4. 等待地址匹配

当接收到本机地址后，I2C_STS的ADDRF标志置1，此时可以读取I2C_STS的SDIR位，得到数据传输方向，当SDIR=0时数据传输方向为从机接收数据，当SDIR=1时，数据方向为从机发送数据，通过读取I2C_STS的ADDR[6:0]标志，可以知道接收到的地址是多少，这在双地址模式以及使用了地址2的屏蔽功能模式下，比较有用。

当通过设置I2C_CLR的ADDRC=1清除ADDRF标志后，开始数据传输。

5. 传输数据（从机发送，开启时钟延展，STRETCH=0）

当在地址匹配后：

1. I2C_TXDT 数据寄存器为空，移位寄存器为空，I2C_STS 的 TDIS=1；
2. 向 TXDT 数据寄存器写入数据 1，然后数据将被立即放进移位寄存器；
3. 此时 TXDT 数据寄存器为空，TDIS 又置 1；
4. 向 TXDT 数据寄存器写入数据 2，TDIS 被清 0；
5. 重复 3、4 步骤直到数据发送完成；
6. 等待收到 NACK 条件，当收到 NACK 条件时，I2C_STS 的 ACKFAILF 置 1，将 I2C_CLR 的 ACKFAILC 写 1，清除 ACKFAILF 标志；
7. 等待收到 STOP 条件，当收到 STOP 条件时，I2C_STS 的 STOPF 置 1，将 I2C_CLR 的 STOPC 写 1，清除 STOPF 标志，传输结束。

需要注意的是，在时钟延展关闭（STRETCH=1）的情况下，如果在将要传输数据的第一个 Bit 位开始发送之前，也就是 SDA 边沿产生之前，如果数据还未写入 TXDT 数据寄存器，那么会发生欠载错误，此时 I2C_STS 的 OUF 将会置 1，并将 0xFF 发送到总线。

为了能及时的写入数据，可以在通信开始前，先将数据写入到 DT 寄存器，写入的方式有如下两种：

- 直接写入：软件先将 TDBE 置 1，目的是为了清空 TXDT 寄存器的数据，然后将第一个数据写入 TXDT 寄存器，此时 TDBE 清零；
- 通过中断或者 DMA 写入：软件先将 TDBE 置 1，目的是为了清空 TXDT 寄存器的数据，再将 TDIS 置 1，目的是为了产生一个 TDIS 事件，TDIS 事件可以产生一个中断或者 DMA 请求，此时就可以通过 DMA 或者在中断函数内将数据写入 TXDT 寄存器。

6. 传输数据（从机接收，开启时钟延展，STRETCH=0）

当在地址匹配后：

1. I2C_RXDT 数据寄存器为空，移位寄存器为空，I2C_STS 的 RDBF=0；
2. 当收到数据后，RDBF=1，读取 RXDT 数据寄存器，RDBF 被自动清零；
3. 重复步骤 2 直到所有数据接收完成；
4. 等待收到 STOP 条件，当收到 STOP 条件时，I2C_STS 的 STOPF 置 1，将 I2C_CLR 的 STOPC 写 1，清除 STOPF 标志，传输结束。

在从机接收模式下，可以选择使用从机字节控制模式（SCTRL=1）接收数据，在从机字节控制模式下，可以对每一个接收到的字节进行 ACK/NACK 的控制，这种模式通常使用在 SMBus 协议中，关于从机字节控制模式的详细用法，请参照 11.4.2 数据传输管理章节。

需要注意的是，在时钟延展关闭（STRETCH=1）的情况下，在收到数据后从机应该及时的将数据读走，如果在已经收到 1 个字节后，在下一个字节接收完成之前，如果还未将数据读走，从机将产生过载错误，此时 I2C_STS 的 OUF 将会置 1，并自动回复 NACK。

上述提到的标志，均可通过相应的中断使能位，产生中断，具体的对应关系，请见中断章节。

从机发送流程

图 11-10 I²C从机发送流程图

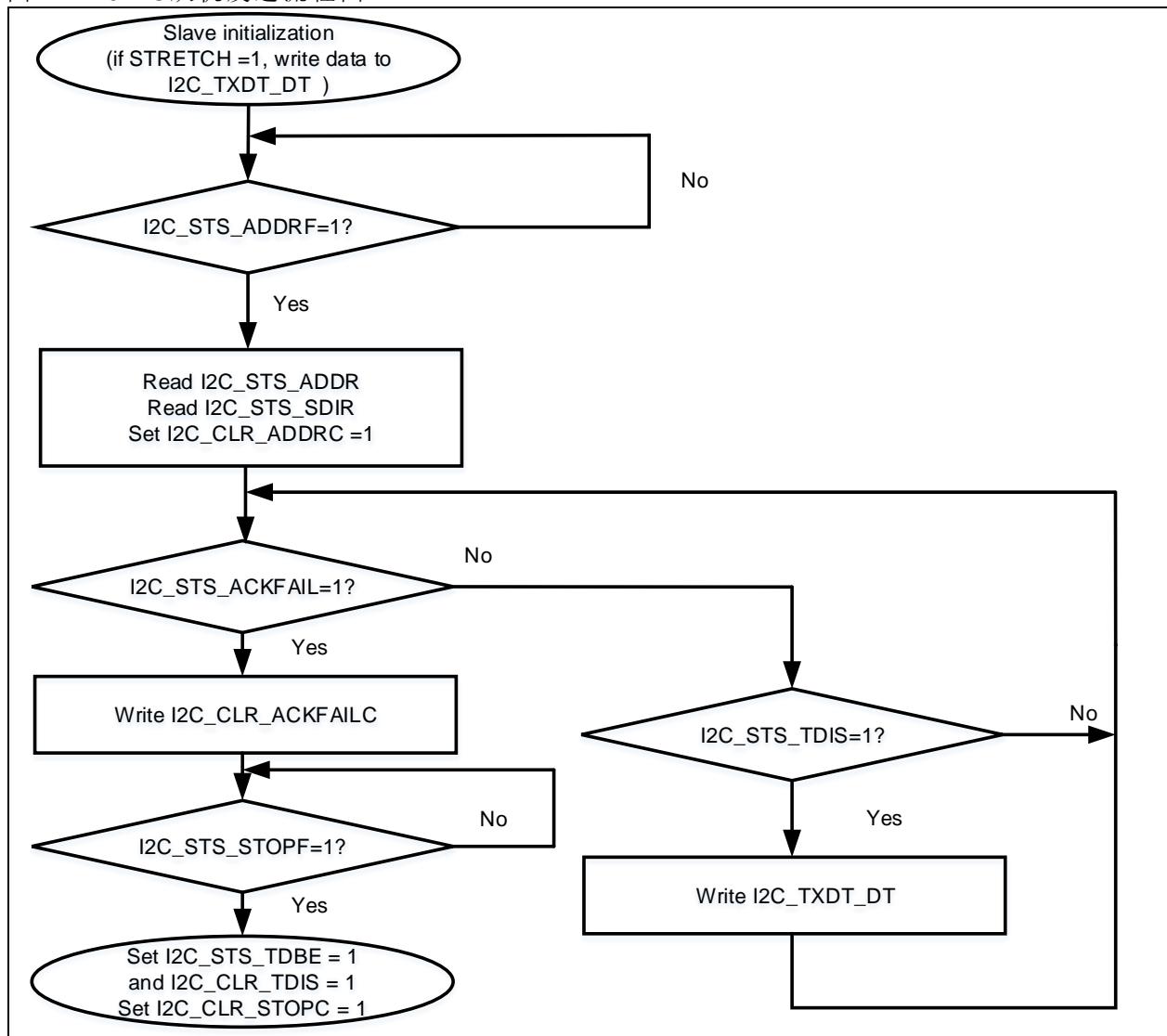
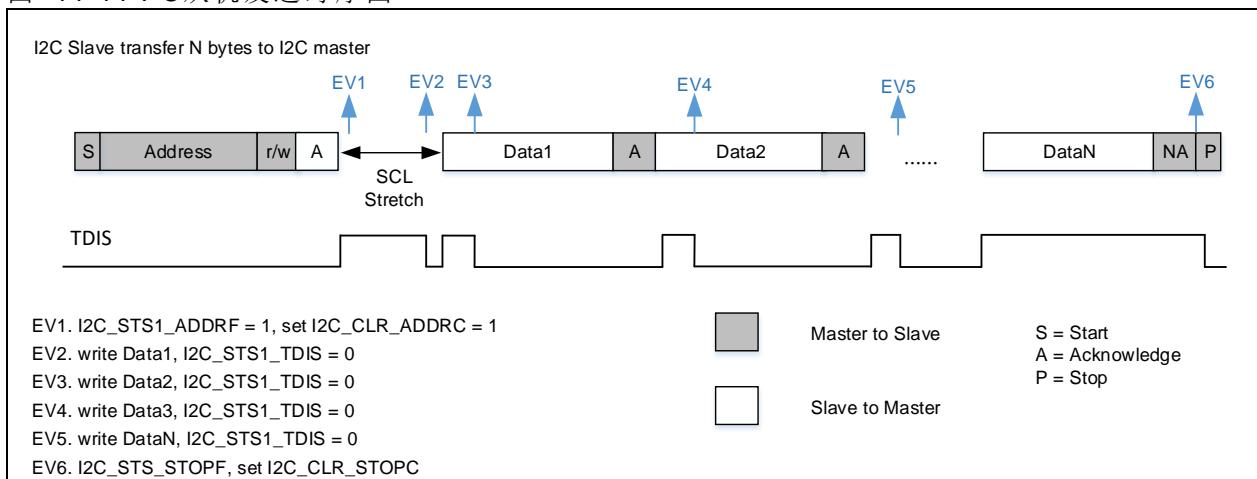
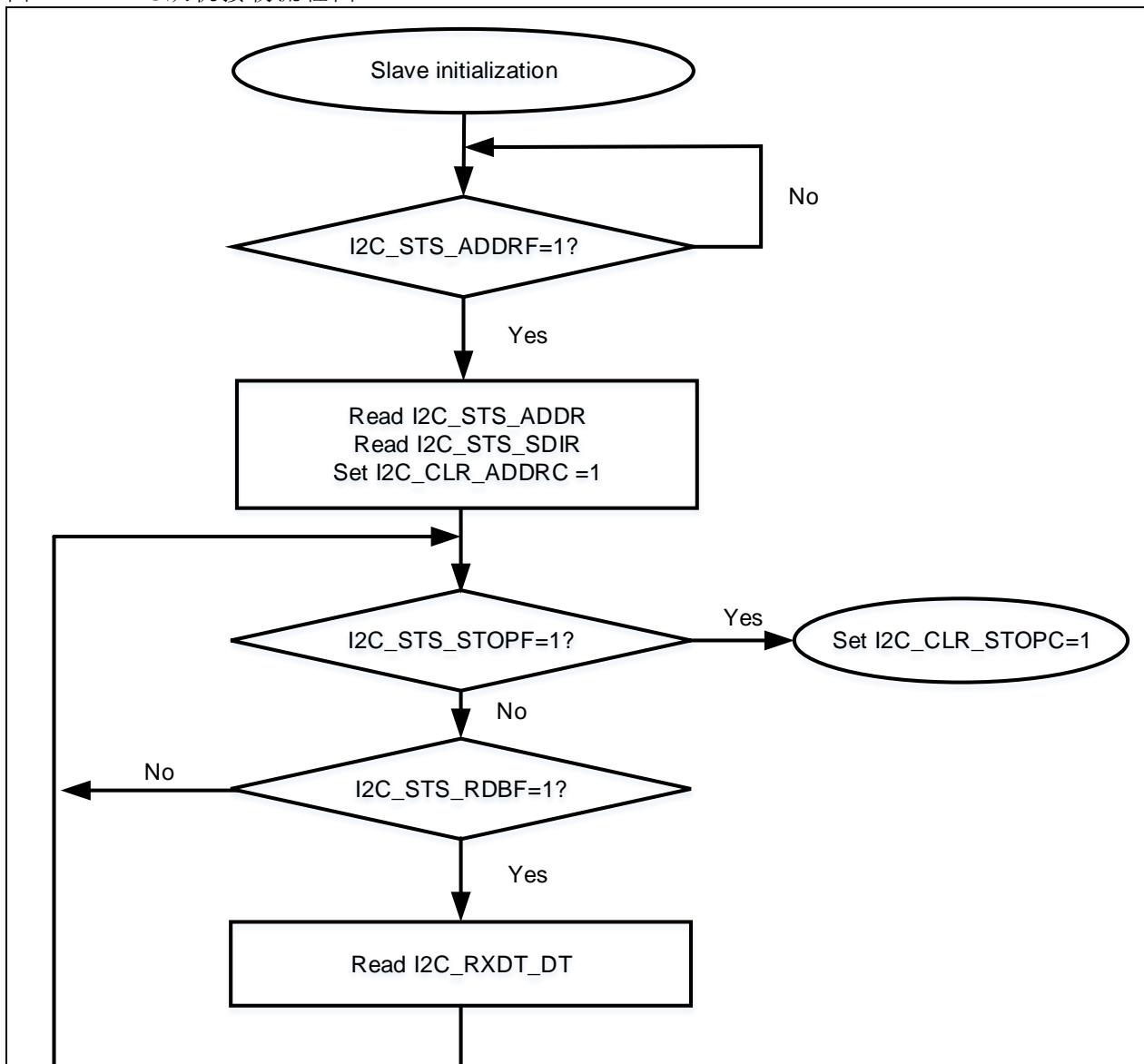
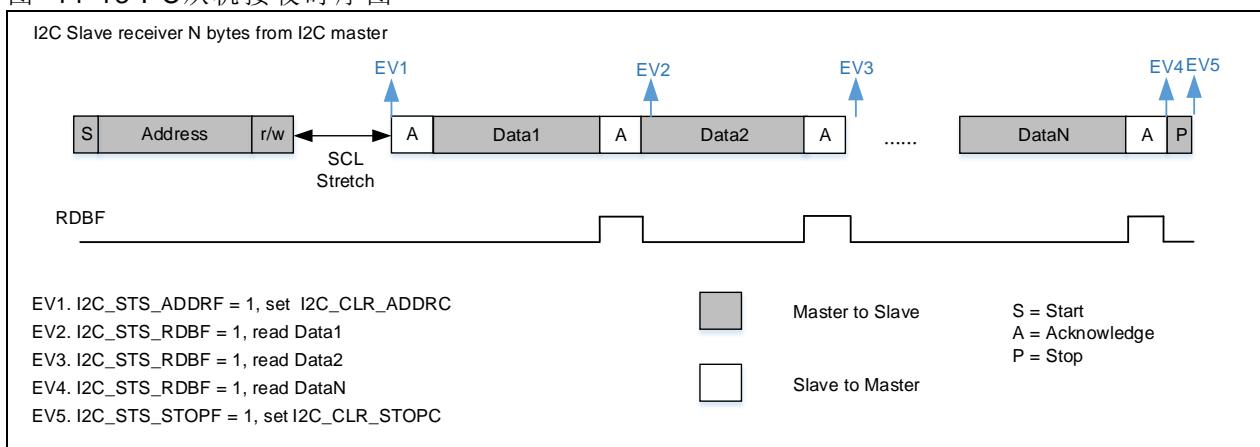


图 11-11 I²C从机发送时序图



从机接收流程

图 11-12 I²C从机接收流程图图 11-13 I²C从机接收时序图

11.4.5 SMBus功能

SMBus即系统管理总线是一双线制总线，基于I²C的操作原理，系统中各设备之间通过SMBus总线传送和接收讯息，通过SMBus总线，设备可以提供制造商信息，告诉系统型号，报告不同类型错误，接受控制参数等。关于SMBus更加详细的信息请参考SMBus2.0协议。

SMBus 和 I²C 的差异

1. SMBus需维持最低10kHz以上的运作频率主要为了管理监控，只要在保持一定传速运作的情况下加入参数，就可轻松获知总线目前是否处于闲置（Idle）中，省去逐一侦测传输过程中的停止条件，或持续保有停断侦测并辅以额外参数侦测，I²C则无此功能
2. SMBus传输速度从最小10kHz到最大100kHz，I²C则是无最小传输速度，根据不同模式有不同的最大传输速度，分为标准模式（100kHz）和快速模式（400kHz）
3. SMBus有超时机制，I²C则无超时机制

SMBus 地址解析协议（ARP）

通过ARP协议可以给总线上的设备动态的分配一个唯一的新地址，解决地址冲突问题。关于ARP协议更详细的信息请参考SMBus2.0协议。

通过使能I2C_CTRL1寄存器的DEVADDREN位，可以使能I²C接口对设备默认地址（0b1100001x）的识别，但是像唯一设备标识（UDID）以及具体的协议实现过程，需要由软件来处理。

SMBus 主机通知协议

通过SMBus主机通知协议，可让从设备发送数据到主设备，例如从机可以通过此协议通知主机进行ARP。关于SMBus主机通知协议更详细的信息请参考SMBus2.0协议。

当在主机模式（HADDREN=1）下，I²C接口使能对主机默认地址（0b0001000x）的识别。

SMBus 提醒协议（SMBus Alert）

SMBALERT是一个可选信号，连接主机和从机的ALERT引脚，用于从机通知主机访问从机，SMBALERT是一个线与信号。关于SMBus提醒协议更详细的信息请参考SMBus2.0协议。

操作流程如下：

SMBus 主机

1. 启用SMBus提醒模式（SMBALERT=1）；
2. 根据实际需求启用ALERT中断；
3. 当ALERT引脚上产生了提醒事件时（ALERT引脚电平由高变低）；
4. 如果使能了中断，主机将产生ALERT中断；
5. 主机处理该中断并向从机发送提醒响应地址ARA(Alert Response Address)(0001100x)，访问所有设备，获取从机地址，只有那些将SMBALERT拉低的设备才会应答；
6. 主机通过获取到的从机地址进行下一步操作。

SMBus 从机

1. 产生提醒事件，ALERT引脚由高变低（SMBALERT=1），此时从机响应ARA（Alert Response Address）（0001100x）；
2. 等待主机通过发送ARA地址获取从机地址；
3. 上报自己的地址，如果发生了仲裁丢失，继续等待；
4. 地址上报成功，释放ALERT引脚（SMBALERT=0）。

包错误检查

包错误校验（PEC）用于保证数据传输的正确性和完整性，使用CRC-8进行校验，多项式为：

$$C(x) = x^8 + x^2 + x + 1$$

PEC计算：当I2C_CTRL1的PECEN=1时启动PEC计算，检验数据包括地址以及数据。

PEC传输：当I2C_CTRL2的PECTEN=1时，启动PEC传输使能，当数据传输个数达到N-1(CNT=N)了之后：

- 主机：会自动发送PEC；
- 从机：会自动将第N个数据作为PEC检验，如果PEC校验不正确将回复NACK，并且I2C_STS的PECERR标志将会置1，当在从机发送模式下，无论校验是否正确，都将回复NACK。

SMBus 超时

在SMBus协议里面，主要有三种超时检测：

- 低电平超时T_{TIMEOUT}：时钟线SCL单次低电平时间（主机从机都会检测，主动被动拉低都会计算）；
- 从机低电平累积超时T_{LOW:SEXT}：从机在START条件到STOP条件之间的主动拉低SCL时

间总和；

- 主机低电平累积超时 $T_{LOW:MEXT}$: 主机在上一个字节的 ACK 到下一次数据的第 8 个 BIT 位之间（单个字节），主动拉低拉低 SCL 时间总和。

需要注意的是 $T_{LOW:SEXT}$ 和 $T_{LOW:MEXT}$ 只计算自己主动拉低的时间，外部拉低的时间不计算在内，而 $T_{TIMEOUT}$ 主动、被动拉低都会计算。

表 11-3 SMBus 超时规范

超时类型	最小值	最大值	单位
低电平超时 $T_{TIMEOUT}$	25	35	ms
从机低电平累积超时 $T_{LOW:SEXT}$	-	25	ms
主机低电平累积超时 $T_{LOW:MEXT}$	-	10	ms

I²C 外设内置两个计数器用于检测超时，在 I²C_TIMEOUT 寄存器中配置，当发生超时事件时 I²C_STS 的 TMOUT 置 1，将 I²C_CLR 的 TMOUTC 置 1 可清 0。

- EXTTIME: 用于主机、从机低电平累积超累计超时检测

超时时间 = (EXTTIME + 1) × 2048 × T_{I²C_CLK}

- TOTIME: 时钟电平超时检测，可以通过 TOMODE 位来选择检测电平

TOMODE=0: 低电平超时检测，超时时间 = (TOTIME + 1) × 2048 × T_{I²C_CLK}

TOMODE=1: 高电平超时检测，超时时间 = (TOTIME + 1) × 4 × T_{I²C_CLK}

表 11-4 SMBus 超时检测配置

超时类型	其他配置	使能	超时时间计算
低电平超时 $T_{TIMEOUT}$	TOMODE=0	TOEN=1	(TOTIME + 1) × 2048 × T _{I²C_CLK}
从机低电平累积超时 $T_{LOW:SEXT}$	-	EXTEN=1	(EXTTIME + 1) × 2048 × T _{I²C_CLK}
主机低电平累积超时 $T_{LOW:MEXT}$	-	EXTEN=1	(EXTTIME + 1) × 2048 × T _{I²C_CLK}

从机字节控制

在从机接收模式下，如果需要对接收到的每一个字节进行 ACK/NACK 的控制，可以选择从机接收数据控制模式 (SCTRL=1) 接收数据，关于从机字节控制模式的用法，请参照 11.4.2 数据传输管理章节。

表 11-5 SMBus 模式配置表

传输模式	PECEN	PECTEN	RLDEN	ASTOPEN	SCTRL
主机发送/接收 +STOP	1	1	0	1	-
主机发送/接收 +RESTART	1	1	0	0	-
从机接收	1	1	1	-	1
从机发送	1	1	0	-	-

SMBus 使用流程

- 设置 SMBus 默认地址响应：

如果 HADDREN=1：响应主机默认地址 (0b0001000x)；

如果 DEVADDREN=1：响应设备默认地址 (0b1100001x)。

- PEC 配置：

- 如果是从机，可以根据需要使能从机字节控制模式 (I²C_CTRL1 的 SCTRL)；

- 其他配置和 I²C 使用配置一样。

需要注意的是各种 SMBus 协议需要由软件来实现，I²C 接口只提供了这些协议的地址识别。

11.4.6 SMBus 主机通信流程

SMBus 主机通信流程和 I²C 主机通信流程类似。

1. I²C 时钟初始化（配置 I2C_CLKCTRL 寄存器）

- I²C 时钟分频：DIV[7:0]
- 数据保持时间 (t_{HD;DAT})：SDAD[3:0]
- 数据建立时间 (t_{SU;DAT})：设置 SCLD[3:0]
- SCL 高电平时间：设置 SCLH[7:0]
- SCL 低电平时间：设置 SCLL[7:0]

该寄存器的配置可以使用 Artery_I2C_Timing_Configuration 时钟配置工具计算。

2. SMBus 相关初始化

- 选择为 SMBus 主机：设置 HADDREN=1，响应主机默认地址 (0b0001000x)
- 使能 PEC 计算：设置 I2C_CTRL1 的 PECEN=1
- 使能 PEC 传输：设置 I2C_CTRL2 的 PECTEN=1

3. 设置传输字节数

- SMBus 模式下单次传输字节数<255，配置 I2C_CTRL2 的 RLDEN=0 以及 CNT[7:0]=N

4. 设置传输结束模式

- ASTOPEN=0：软件结束模式，当数据传输完成后，I2C_STS 的 TDC 标志置 1，软件设置 GENSTOP=1 或者 GENSTART=1，发送 STOP 条件或者 START 条件。
- ASTOPEN=1：自动结束模式，当数据传输完成后，自动发送 STOP 条件。

5. 设置从机地址

- 设置寻址的从机地址值 (I2C_CTRL2 的 SADDR)
- 设置从机地址模式为 7 位地址模式 (I2C_CTRL2 的 ADDR10=0)

6. 设置传输方向 (I2C_CTRL2 的 DIR)

- DIR=0：主机接收数据
- DIR=1：主机发送数据

7. 开始传输

设置 I2C_CTRL2 的 GENSTART=1，主机开始在总线上发送 START 条件和从机地址，当从机响应了地址之后，主机的 I2C_STS 的 ADDR=1，设置 I2C_CLR 的 ADDRC=1 清除 ADDR 标志后，开始数据传输。

8. 主机发送数据

1. I2C_TXDT 数据寄存器为空，移位寄存器为空，I2C_STS 的 TDIS=1；
2. 向 TXDT 数据寄存器写入数据 1，然后数据将被立即放进移位寄存器；
3. 此时 TXDT 数据寄存器为空，TDIS 又置 1；
4. 向 TXDT 数据寄存器写入数据 2，TDIS 被清 0；
5. 重复 2、3 步骤直到发送 N-1 个数据；
6. 此时主机将自动发送第 N 个数据，也就是 PEC。

9. 主机接收数据

1. 当收到数据后，RDBF=1，读取 RXDT 数据寄存器，RDBF 被自动清零；
2. 重复步骤 1 直到接收 N 个数据，第 N 个数据为 PEC，在接收第 N 个字节，也就是 PEC 时，无论 PEC 是否正确，主机会自动发送一个 NACK。

10. 结束时序

- 停止条件产生：
 - 软件结束模式 (ASTOPEN=0)：当 I2C_STS 的 TDC 置 1，设置 GENSTOP=1 产生 STOP 条件；
 - 自动结束模式 (ASTOPEN=1)：当 I2C_STS 的 TDC 置 1，硬件自动产生 STOP 条件。
- 等待产生 STOP 条件，当 STOP 条件产生时，I2C_STS 的 STOPF 置 1，将 I2C_CLR 的 STOPC 写 1，清除 STOPF 标志，传输结束。

SMBus 主机发送流程

图 11-14 SMBus 主机发送流程图

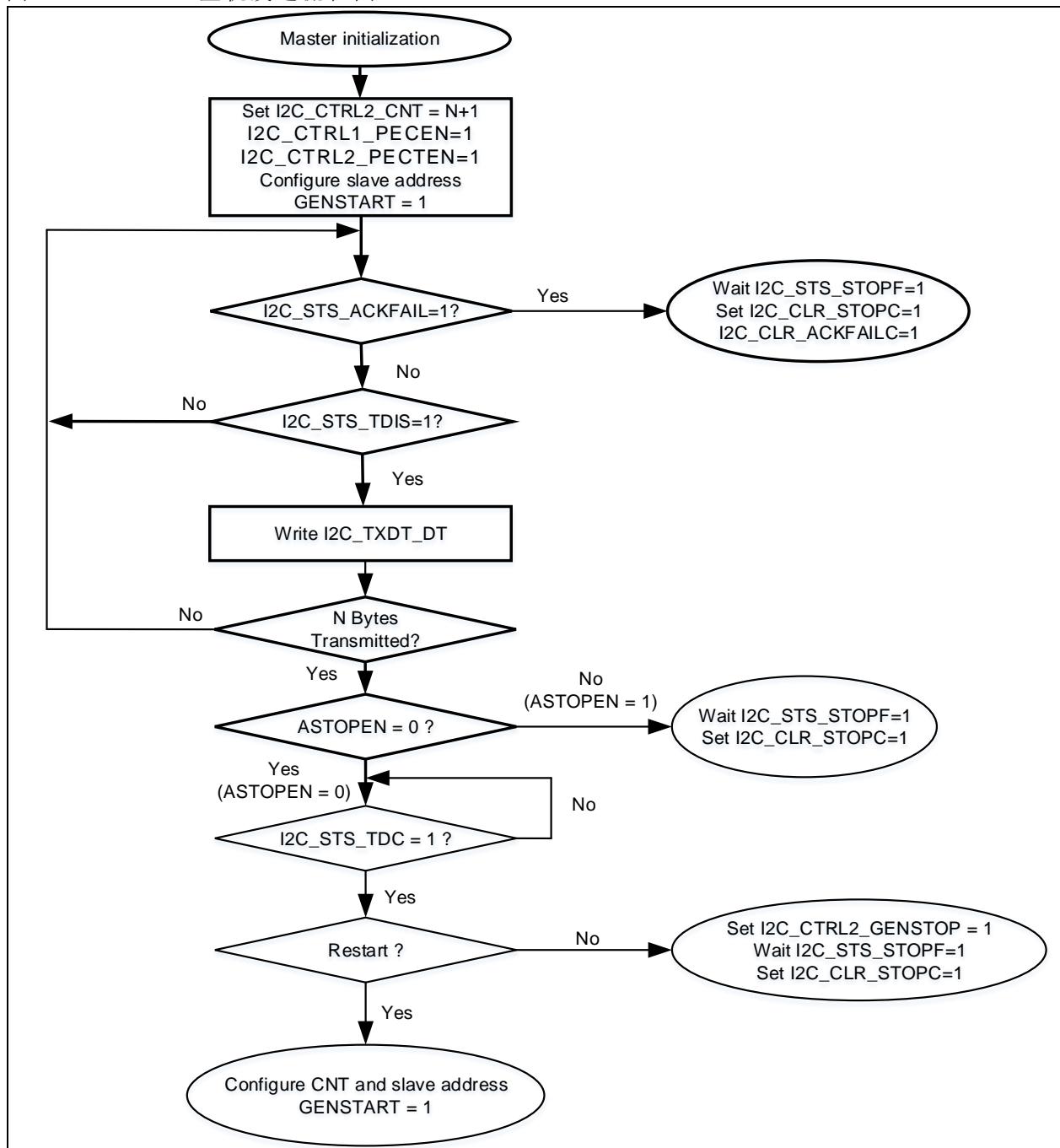


图 11-15 SMBus 主机发送时序图

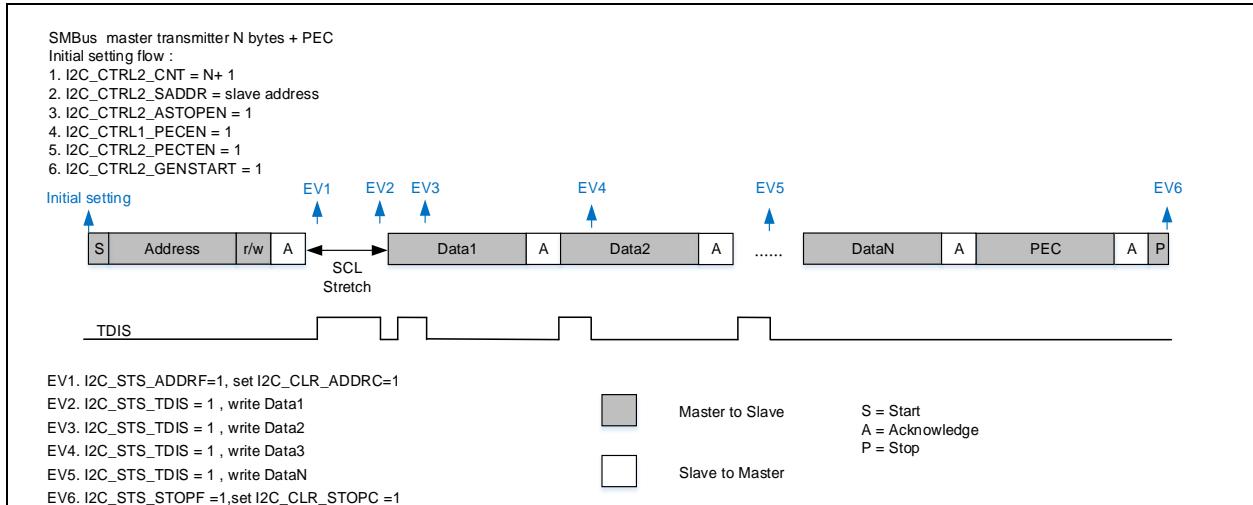
**SMBus 主机接收流程**

图 11-16 SMBus 主机接收流程图

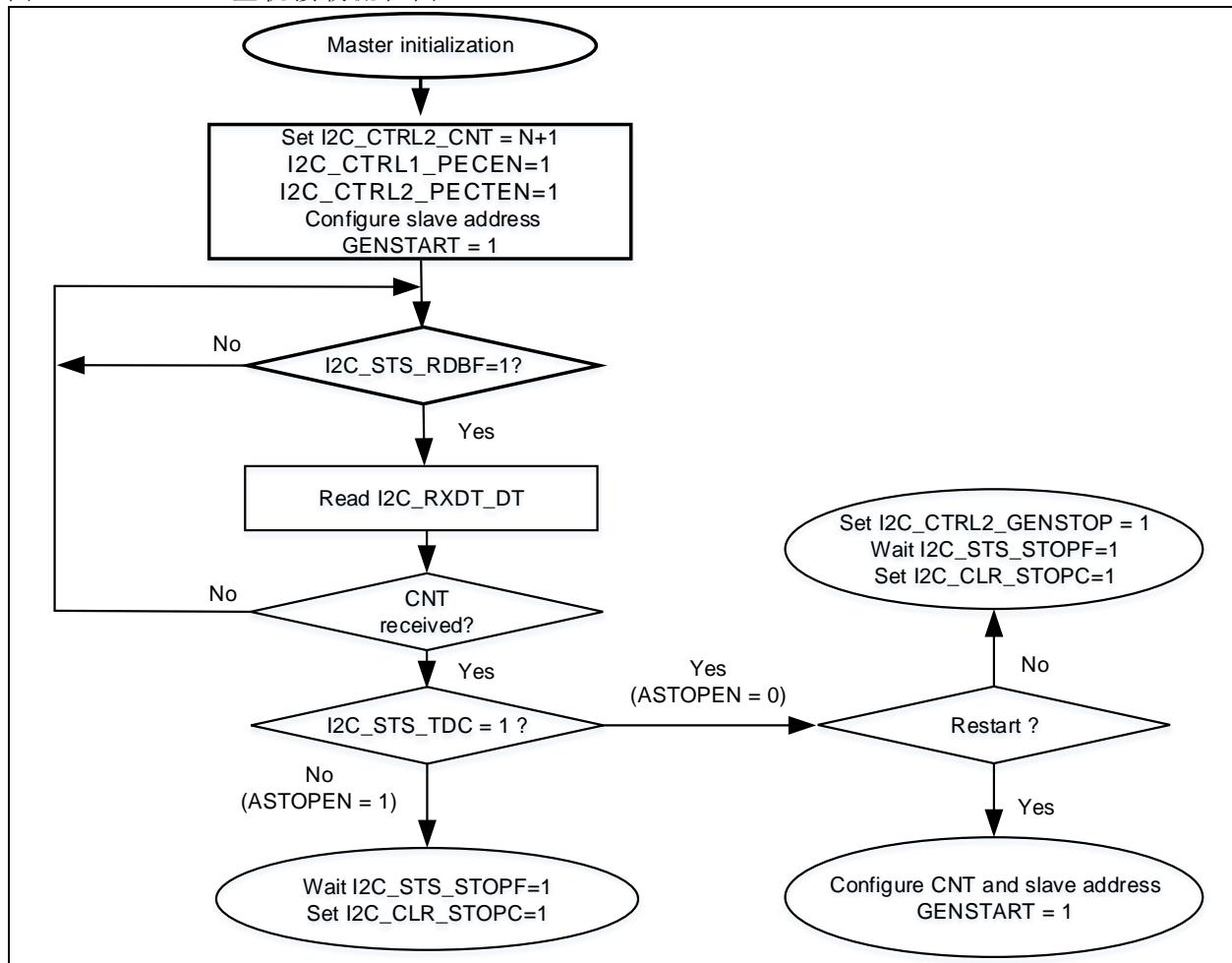
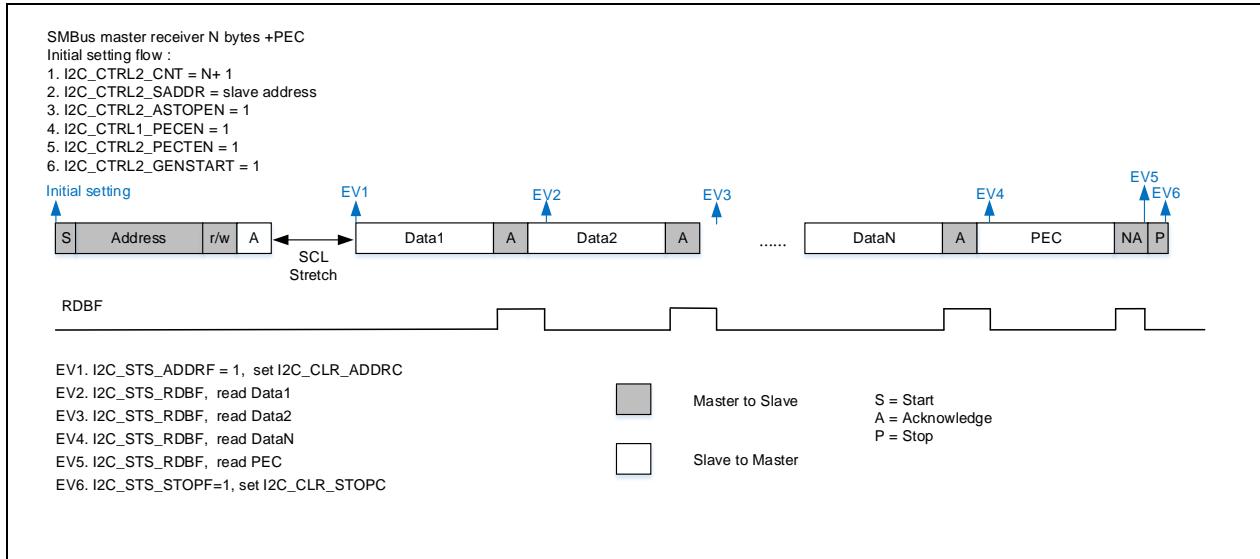


图 11-17 SMBus 主机接收时序图



11.4.7 SMBus 从机通信流程

SMBus 从机通信流程和 I²C 从机通信流程类似。

1. I²C 时钟初始化（配置 I2C_CLKCTRL 寄存器）

- I²C 时钟分频：DIV[7: 0]
- 数据保持时间（t_{HD;DAT}）：SDAD[3: 0]
- 数据建立时间（t_{SU;DAT}）：设置 SCLD[3: 0]

该寄存器的配置可以使用 Artery_I2C_Timing_Configuration 时钟配置工具计算。

2. 设置本机地址

- 设置地址模式为 7 位：设置 I2C_OADDR1 的 ADDR1MODE = 0
- 设置地址 1：设置 I2C_OADDR1 的 ADDR1
- 使能地址 1：设置 I2C_OADDR1 的 ADDR1EN=1

3. SMBus 相关初始化

- 选择为 SMBus 设备：设置 DEVADDREN=1，响应设备默认地址（0b1100001x）
- 使能 PEC 计算：设置 I2C_CTRL1 的 PECEN=1
- 设置从机字节控制模式：
从机发送：关闭字节控制模式，设置 I2C_CTRL1 的 SCTRL=0；
从机接收：使能字节控制模式，设置 I2C_CTRL1 的 SCTRL=1。

4. 等待地址匹配

当接收到本机地址后，I2C_STS 的 ADDRF 标志置 1，此时可以读取 I2C_STS 的 SDIR 位，得到数据传输方向，当 SDIR=0 时数据传输方向为从机接收数据，当 SDIR=1 时，数据方向为从机发送数据，通过读取 I2C_STS 的 ADDR[6:0] 标志，可以知道接收到的地址是多少。

使能 PEC 传输：设置 I2C_CTRL2 的 PECTEN=1。

设置传输个数：

- 从机发送：设置 I2C_CTRL2 的 CNT=N；
- 从机接收：设置 I2C_CTRL2 的 CNT=1；

重载模式设置：

- 从机发送：设置 I2C_CTRL2 的 RLDEN=0；
- 从机接收：设置 I2C_CTRL2 的 RLDEN=1；

设置 I2C_CTRL 的 ADDRC=1，清除 ADDRF 标志，开始数据传输。

5. 传输数据（从机发送，开启时钟延展，STRETCH=0）

当在地址匹配后：

1. I2C_TXDT 数据寄存器为空，移位寄存器为空，I2C_STS 的 TDIS=1；
2. 向 TXDT 数据寄存器写入数据 1，然后数据将被立即放进移位寄存器；
3. 此时 TXDT 数据寄存器为空，TDIS 又置 1；
4. 向 TXDT 数据寄存器写入数据 2，TDIS 被清 0；
5. 重复 3、4 步骤直到数据发送 N-1 个数据；
6. 此时从机将自动发送第 N 个数据，也就是 PEC；
7. 等待收到 NACK 条件，当收到 NACK 条件时，I2C_STS 的 ACKFAILF 置 1，将 I2C_CLR 的 ACKFAILC 写 1，清除 ACKFAILF 标志；
8. 等待收到 STOP 条件，当收到 STOP 条件时，I2C_STS 的 STOPF 置 1，将 I2C_CLR 的 STOPC 写 1，清除 STOPF 标志，传输结束。

6. 传输数据（从机接收，开启时钟延展，STRETCH=0）

当在地址匹配后：

1. I2C_RXDT 数据寄存器为空，移位寄存器为空，I2C_STS 的 RDBF=0；
2. 当收到一个字节后，RDBF=1，TCRLD 置 1，从机拉住 SCL 总线；
3. 读取 RXDT 数据寄存器，RDBF 被自动清零；
4. 根据需要配置 I2C_CTRL2 的 NACKEN 位，来产生一个 ACK 或 NACK；
如果产生一个 NACK，通信结束；
如果产生一个 ACK，通信继续，此时写入 CNT=1，硬件自动清除 TCRLD 标志，从机释放 SCL 总线，继续接收下一个字节；
5. 重复步骤 2、3、4 直到接收 N-1 个数据；
6. 设置 I2C_CTRL2 的 RLDEN=0，关闭重载模式，设置 CNT=1，重复 2、3 步骤接收 PEC
如果 PEC 校验错误，PECERR 标志将会置 1；
7. 等待收到 STOP 条件，当收到 STOP 条件时，I2C_STS 的 STOPF 置 1，将 I2C_CLR 的 STOPC 写 1，清除 STOPF 标志，传输结束。

SMBus 从机发送

图 11-18 SMBus从机发送流程图

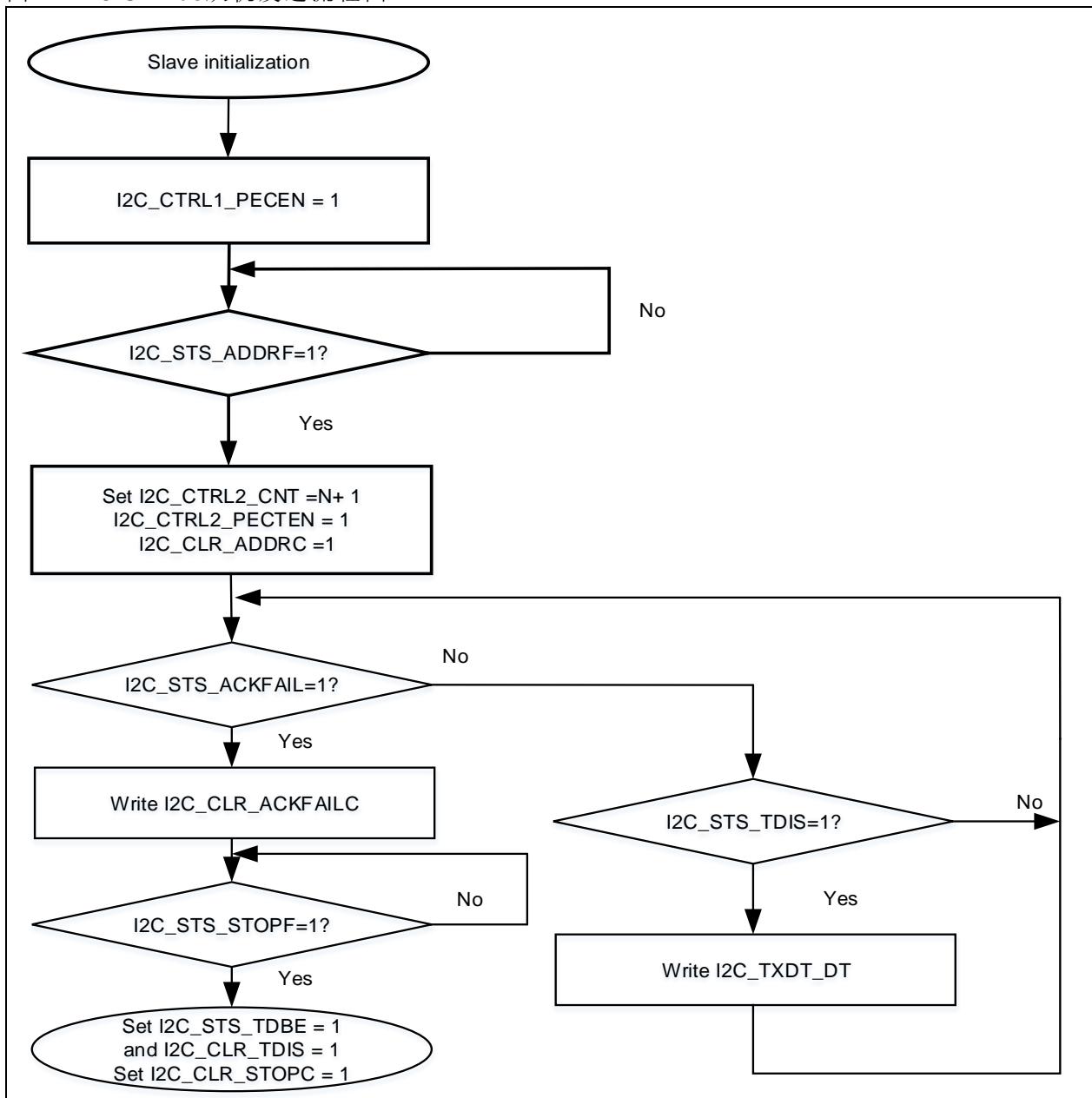


图 11-19 SMBus从机发送时序图

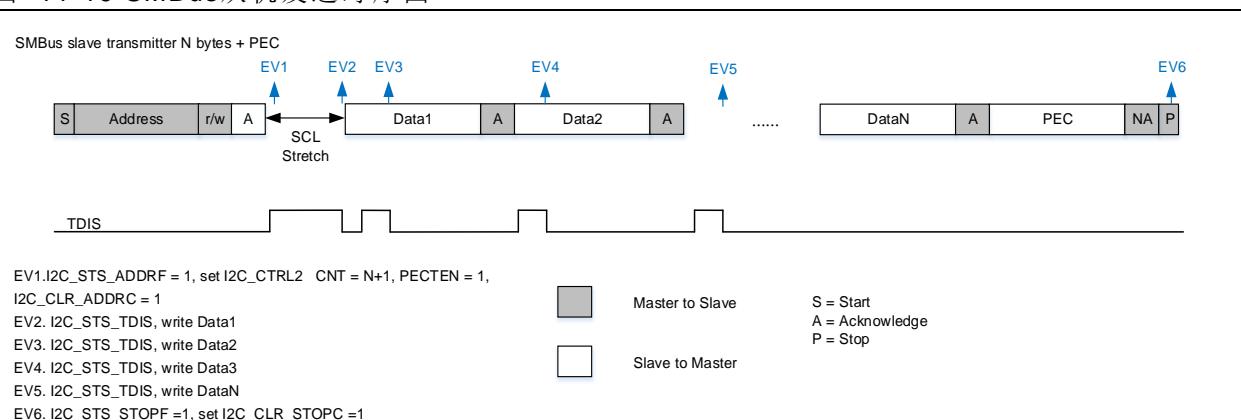
**SMBus 从机接收**

图 11-20 SMBus从机接收流程图

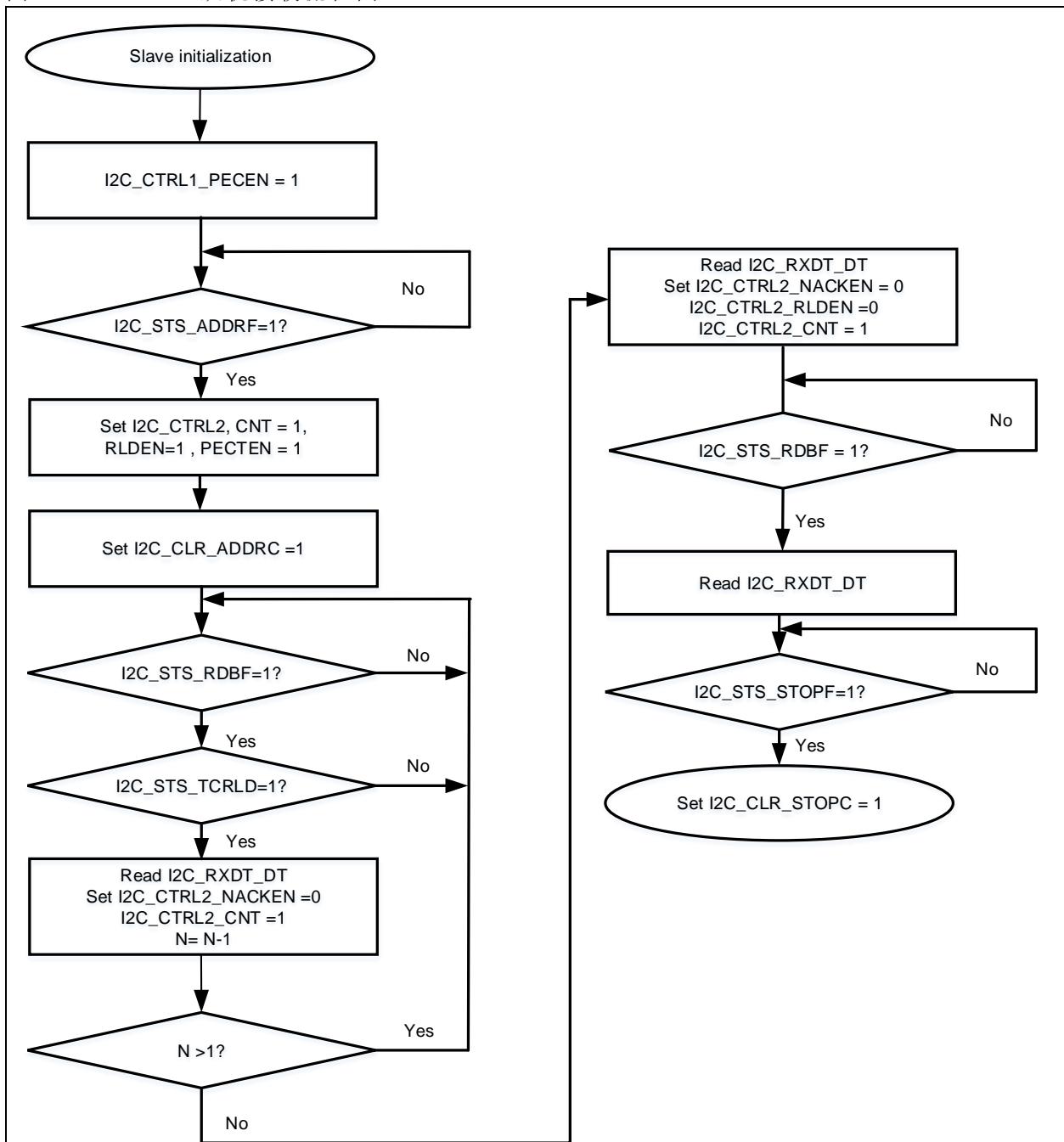
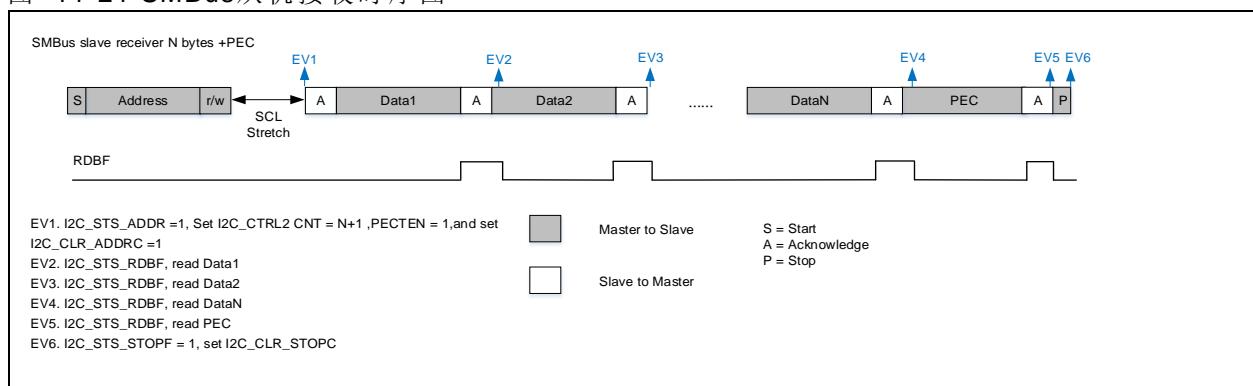


图 11-21 SMBus从机接收时序图



11.4.8 DMA 传输

I²C 可以使用 DMA 进行数据传输，降低 CPU 负担，在使用 DMA 传输时，TDIEN 和 RDIEN 应该保持关

闭。

DMA 发送 (DMATEN=1)

1. 设置外设地址 (DMA通道x外设地址寄存器 (DMA_CxPADDR) =数据寄存器(I2C_TXDT)地址)
2. 设置数据存储地址 (DMA通道x存储器地址寄存器 (DMA_CxMADDR) =数据存储地址)
3. 设置传输方向为内存到外设 (DMA_CHCTRL的DTD=1)
4. 设置传输字节数 (DMA通道x数据传输量寄存器 (DMA_CxDTCNT))
5. 设置DMA通道的其他配置, 例如: 优先级、存储器数据宽度、外设数据宽度、中断等 (DMA_CHCTRL)
6. 使能DMA通道 (DMA通道x配置寄存器 (DMA_CxCTRL) 的CHEN=1)。
7. 使能I²C DMA请求 (控制寄存器1 (I2C_CTRL1) 的DMATEN=1), 当状态寄存器1 (I2C_STS) 的TDIS位被置1时, DMA将数据从内存地址传输到数据寄存器(I2C_TXDT)
8. 等待传输字节数DMA通道x数据传输量寄存器 (DMA_CxDTCNT) =0时, 数据传输完成, (可以通过DMA传输完成中断来等待)。
9. 主机发送模式: 停止时序见I²C主机通信流程章节。
从机发送模式: 停止时序见I²C从机机通信流程章节。

DMA 接收 (DMAREN=1)

1. 设置外设地址 (DMA通道x外设地址寄存器 (DMA_CxPADDR) =数据寄存器(I2C_RXDT)地址)
2. 设置数据存储地址 (DMA通道x存储器地址寄存器 (DMA_CxMADDR) =数据存储地址)
3. 设置传输方向为外设到内存 (DMA_CHCTRL的DTD=0)
4. 设置传输字节数 (DMA通道x数据传输量寄存器 (DMA_CxDTCNT))
5. 设置DMA通道的其他配置, 例如: 优先级、存储器数据宽度、外设数据宽度、中断等 (DMA_CHCTRL)
6. 使能DMA通道 (DMA通道x配置寄存器 (DMA_CxCTRL) 的CHEN=1)。
7. 使能I²C DMA请求 (控制寄存器1 (I2C_CTRL1) 的DMAREN=1), 当状态寄存器1 (I2C_STS) 的RDBF位被置1时, DMA将数据从I2C_DT寄存器传输到数据存储地址。
8. 等待传输字节数DMA_TCNTx=0时, 数据传输完成, (可以通过DMA传输完成中断来等待)。
9. 主机接收模式: 停止时序见I²C主机通信流程章节。
从机接收模式: 停止时序见I²C从机机通信流程章节。

11.4.9 错误管理

I²C 内部有多种错误管理, 可以极大的提高通信的可靠性, 支持错误管理的事件如下:

表 11-6 I²C错事件

错误事件	事件标志	中断使能位	事件清除位
SMBus 提醒	ALERTF	ERRIEN	ALERTC
超时错误	TMOUT	ERRIEN	TMOUTC
PEC 错误	PECERR	ERRIEN	PECERRC
过载/欠载	OUF	ERRIEN	OUFC
仲裁丢失	ARLOST	ERRIEN	ARLOSTC
总线错误	BUSERR	ERRIEN	BUSERRC

过载或者欠载 (OUE)

只有在从机模式下，且关闭时钟延展 (I2C_CTRL1 的 STRETCH=1) 时，才有可能出现欠载或者过载错误。

从机发送模式：如果在将要传输数据的第一个 bit 位开始发送之前，也就是 SDA 边沿产生之前，如果数据还未写入 TXDT 数据寄存器，那么会发生欠载错误，此时 I2C_STS 的 OUE 将会置 1，并将 0xFF 发送到总线。

从机接收模式：在收到数据后从机应该及时的将数据读走，如果在已经收到 1 个字节后，在下一个字节接收完成之前，如果还未将数据读走，从机将产生过载错误，此时 I2C_STS 的 OUE 将会置 1，并自动回复 NACK。

仲裁丢失 (ARLOST)

当设备控制 SDA 线输出高电平，但是总线上实际输出低电平时，发生仲裁丢失事件。

- 主机发送数据：仲裁可以发生在地址传输、数据传输阶段；
- 主机接收数据：仲裁可以发生在地址传输、响应 ACK 阶段；
- 从机发送数据：仲裁可以发生在数据传输阶段；
- 从机接收数据：仲裁可以发生在响应 ACK 阶段。

当在发生仲裁丢失后，硬件自动将 I2C_STS 的 ARLOST 置 1，无论是主机还是从机，都将会立即释放 SCL、SDA 总线，并自动回到从机状态。

总线错误(BUSERR)

在数据传输阶段，在 SCL 高周期区间 SDA 线必须保持稳定，当 SCL 信号为低时，SDA 才能改变，否则将会出现总线错误，当 SCL 为高电平时：

- SDA 从 1 变成 0：错误的开始条件；
- SDA 从 0 变成 1：错误的停止条件；

以上两种情况都会触发总线错误，硬件自动将 I2C_STS 的 BUSERR 置 1。

PEC 错误 (PECERR)

只有在 SMBus 模式下才存在 PEC，当在主机接收模式和从机接收模式下，如果接收到的 PEC 和内部计算的 PEC 不相等时，会出现 PEC 错误，此时硬件自动将 I2C_STS 的 PECERR 置 1。

在从机接收模式下，如果 PEC 不正确，从机将回复 NACK。

当在主机接收模式下，无论 PEC 是否正确，主机都将回复 NACK。

SMBus 提醒 (ALERTF)

在 SMBus 主机模式下 (HADDREN=1) 且启用了 SMBus 提醒模式 (SMBALERT=1) 时，SMBus 提醒功能可以使用，当 ALERT 引脚上产生了提醒事件时 (ALERT 引脚电平由高变低)，硬件自动将 I2C_STS 的 ALERTF 置 1。

超时错误 (TMOUT)

超时错误是 SMBus 协议所定义的，用来提高系统稳定性的机制，用来避免主机或者从机出现故障时一直拉低总线，导致总线无法使用的情况。当发生了超时事件时 (SMBus 章节所定义的)，硬件自动将 I2C_STS 的 TMOUT 置 1。如果是在从机模式下发生超时事件时，从机将立即释放 SCL 和 SDA 总线；如果是主机发生超时事件时，主机将自动发送一个 STOP 条件结束通信。

11.5 I²C中断

下表列出了所有的 I²C 中断请求。

表 11-7 I²C 中断请求

中断事件	事件标志	中断使能位
地址匹配	ADDRF	ADDRIEN
应答失败	ACKFAIL	ACKFAILIEN
停止条件产生完成	STOPF	STOPIEN
发送中断状态	TDIS	TDIEN
接收数据缓冲器满	RDBF	RDIEN
传输完成等待加载数据	TCRLD	TDCIEN
数据传输完成	TDC	
SMBus 提醒	ALERTF	ERRIEN
超时错误	TMOUT	
PEC 错误	PECERR	ERRIEN
过载/欠载	OUF	
仲裁丢失	ARLOST	ERRIEN
总线错误	BUSERR	

11.6 I²C 调试模式

当微控制器进入调试模式 (Cortex®-M4F 核心处于停止状态) 时，根据 DEBUG 模块中的 I2Cx_SMBUS_TIMEOUT 配置位，SMBUS 超时控制可以继续正常工作或者可以停止。

11.7 I²C 寄存器

下表给出了 I²C 寄存器映像和复位值。

必须以字（32 位）的方式操作这些外设寄存器。

表 11-8 寄存器映像和复位值

寄存器简称	基址偏移量	复位值
I2C_CTRL1	0x00	0x00000000
I2C_CTRL2	0x04	0x00000000
I2C_OADDR1	0x08	0x00000000
I2C_OADDR2	0x0C	0x00000000
I2C_CLKCTRL	0x10	0x00000000
I2C_TIMEOUT	0x14	0x00000000
I2C_STS	0x18	0x00000000
I2C_CLR	0x1C	0x00000000
I2C_PEC	0x20	0x00000000
I2C_RXDT	0x24	0x00000000
I2C_TXDT	0x28	0x00000000

11.7.1 控制寄存器1 (I2C_CTRL1)

域	简称	复位值	类型	功能
位 31: 24	保留	0x00	res	保持默认值。
位 23	PECEN	0x0	rw	PEC 计算使能 (PEC calculation enable) 0: 关闭; 1: 开启。
位 22	SMBALERT	0x0	rw	SMBus 提醒功能使能/引脚设置 (SMBus alert enable / pin set) SMBus 主机, 提醒功能使能: 0: 关闭; 1: 开启。 SMBus 从机, 提醒地址使能: 0: 置高; 1: 置低, 响应 0001100x。
位 21	DEVADDREN	0x0	rw	SMBus 设备默认地址使能 (SMBus device default address enable) 0: 关闭; 1: 开启, 响应设备默认地址 1100001x。
位 20	HADDREN	0x0	rw	SMBus 主机地址使能 (SMBus host default address enable) 0: 关闭; 1: 开启, 响应主机地址 0001000x。
位 19	GCAEN	0x0	rw	广播地址使能 (General call address enable) 0: 关闭; 1: 开启, 响应地址 0000000x。
位 18	保留	0x0	res	保持默认值。
位 17	STRETCH	0x0	rw	时钟延展模式 (Clock stretching mode) 0: 开启; 1: 关闭。 注: 只在从机模式下有效。 注: 只有当 I ² C 被关闭时(I2CEN=0)才能设置这些位。
位 16	SCTRL	0x0	rw	从机接收数据控制 (Slave receiving data control) 0: 关闭; 1: 开启。
位 15	DMAREN	0x0	rw	DMA 数据接收使能 (DMA receive data request enable) 0: 关闭; 1: 开启。
位 14	DMATEN	0x0	rw	DMA 数据发送使能 (DMA Transmit data request enable) 0: 关闭; 1: 开启。
位 13: 12	保留	0x0	res	保持默认值。
位 11: 8	DFLT	0x0	rw	数字滤波值 (Digital filter value) 总线上小于此宽度的毛刺将被滤除, 滤波时间 = DFLT × T _{I2C_CLK} 。 注: 只有当 I ² C 被关闭时(I2CEN=0)才能设置这些位。
位 7	ERRIEN	0x0	rw	错误中断使能 (Error interrupt enable) 0: 关闭; 1: 开启。
位 6	TDCIEN	0x0	rw	数据传输完成中断使能 (Transfer data complete interrupt enable) 0: 关闭; 1: 开启。
位 5	STOPIEN	0x0	rw	停止条件产生完成中断使能 (Stop generation complete interrupt enable) 0: 关闭; 1: 开启。
位 4	ACKFAILIEN	0x0	rw	应答失败中断使能 (Acknowledge fail interrupt enable) 0: 关闭; 1: 开启。
位 3	ADDRIEN	0x0	rw	地址匹配中断使能 (Address match interrupt enable)

				0: 关闭; 1: 开启。
位 2	RDIEN	0x0	rw	数据接收中断使能 (Receive data interrupt enable) 0: 关闭; 1: 开启。
位 1	TDIEN	0x0	rw	数据发送中断使能 (Transmit data interrupt enable) 0: 关闭; 1: 开启。
位 0	I2CEN	0x0	rw	I ² C 外设使能 (I ² C peripheral enable) 0: 关闭; 1: 开启。

11.7.2 控制寄存器2 (I2C_CTRL2)

域	简称	复位值	类型	功能
位 31: 27	保留	0x00	res	保持默认值。
位 26	PECTEN	0x0	rw	请求 PEC 传输使能 (Request PEC transmission enable) 0: 停止传输; 1: 启动传输。 注: 只有当 I ² C 被开启时(I2CEN=1)才能设置这些位。
位 25	ASTOPEN	0x0	rw	自动发送停止条件使能 (Automatically send stop condition enable) 0: 关闭 (软件结束模式); 1: 开启 (自动结束模式)。
位 24	RLDEN	0x0	rw	发送数据重载模式使能 (Send data reload mode enable) 0: 关闭; 1: 开启。
位 23: 16 CNT[7: 0]		0x00	rw	发送数据个数 (Transmit data counter) 注: 在从机模式下如果 SCTRL=0 则这些位无效, 注: 只有当 I ² C 被关闭时(I2CEN=0)才能设置这些位。
位 15	NACKEN	0x0	rw	不应答使能 (Not acknowledge enable) 0: 应答; 1: 不应答。 注: 只有当 I ² C 被开启时(I2CEN=1)才能设置这些位, 注: 当 I ² C 被关闭时(I2CEN=0)这些位被复位。
位 14	GENSTOP	0x0	rw	产生停止条件 (Generate stop condition) 0: 未产生; 1: 产生。 注: 只有当 I ² C 被开启时(I2CEN=1)才能设置这些位, 注: 当侦测到停止条件被产生硬件自动复位这些位。
位 13	GENSTART	0x0	rw	产生起始条件 (Generate start condition) 0: 未产生; 1: 产生。 注: 只有当 I ² C 被开启时(I2CEN=1)才能设置这些位, 注: 当侦测到起始条件被产生硬件自动复位这些位。
位 12	READH10	0x0	rw	10 位地址头读取时序使能 (10-bit address header read enable) 0: 关闭; 1: 开启。
位 11	ADDR10	0x0	rw	主机发送 10 位地址模式使能 (Host send 10-bit address mode enable) 0: 7 位地址; 1: 10 位地址。
位 10	DIR	0x0	rw	主机数据传输方向 (Master data transmission direction) 0: 发送; 1: 接收。
位 9: 0	SADDR[9: 0]	0x000	rw	主机发送的从机地址 (The slave address sent by the master) 当在 7 位地址模式下时 BIT0 以及 BIT[9: 8]不关心。

11.7.3 地址寄存器1 (I2C_OADDR1)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	res	保持默认值。
位 15	ADDR1EN	0x0	rw	本机地址 1 使能 (Own Address 1 enable) 0: 关闭; 1: 开启。
位 14: 11	保留	0x0	res	保持默认值。
位 10	ADDR1MODE	0x0	rw	本机地址 1 模式 (Own Address mode) 0: 7 位地址; 1: 10 位地址。 注: 只有当 ADDR1EN 被开启时(ADDR1EN=1)才能设置这些位。
位 9: 0	ADDR1[9: 0]	0x000	rw	本机地址 1 (Own address 1) 当在 7 位地址模式下时 BIT0 以及 BIT[9: 8]不关心。 注: 只有当 ADDR1EN 被开启时(ADDR1EN=1)才能设置这些位。

11.7.4 地址寄存器2 (I2C_OADDR2)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	res	保持默认值。
位 15	ADDR2EN	0x0	rw	本机地址 2 使能 (Own address 2 enable) 0: 关闭; 1: 开启。
位 14: 11	保留	0x0	res	保持默认值。
位 10: 8	ADDR2MASK[2: 0]	0x0	rw	本机地址 2 位屏蔽 (Own address 2-bit mask) 000: 匹配地址位[7: 1]; 001: 只匹配地址位[7: 2]; 010: 只匹配地址位[7: 3]; 011: 只匹配地址位[7: 4]; 100: 只匹配地址位[7: 5]; 101: 只匹配地址位[7: 6]; 110: 只匹配地址位[7]; 111: 所有非 I ² C 保留地址都会响应。 注: 只有当 ADDR2EN 被开启时(ADDR2EN=1)才能设置这些位。
位 7: 1	ADDR2[7: 1]	0x00	rw	本机地址 2 (Own address 2) 7 位地址。 注: 只有当 ADDR2EN 被开启时(ADDR2EN=1)才能设置这些位。
位 0	保留	0x0	res	保持默认值。

11.7.5 时序寄存器 (I2C_CLKCTRL)

域	简称	复位值	类型	功能
位 31: 28	DIVL[3: 0]	0x0	rw	时钟分频值低 4 位 (Low 4 bits of clock divider value)
位 27: 24	DIVH[7: 4]	0x0	rw	时钟分频值高 4 位 (High 4 bits of clock divider value) DIV = (DIVH << 4) + DIVL。
位 23: 20	SCLD[3: 0]	0x0	rw	SCL 输出延时 (SCL output delay) $T_{SCLD} = (SCLD + 1) \times (DIV + 1) \times T_{I2C_CLK}$ 。
位 19: 16	SDAD[3: 0]	0x0	rw	SDA 输出延时 (SDA output delay) $T_{SDAD} = (SDAD + 1) \times (DIV + 1) \times T_{I2C_CLK}$ 。
位 15: 8	SCLH[7: 0]	0x00	rw	SCL 高电平 (SCL high level) $T_{SCLH} = (SCLH + 1) \times (DIV + 1) \times T_{I2C_CLK}$ 。
位 7: 0	SCLL[7: 0]	0x00	rw	SCL 低电平 (SCL low level) $T_{SCLL} = (SCLL + 1) \times (DIV + 1) \times T_{I2C_CLK}$ 。

注意: 只有当I²C被关闭时(I2CEN=0)才能设置时钟控制寄存器(I2C_CLKCTRL)。

11.7.6 超时寄存器 (I2C_TIMEOUT)

域	简称	复位值	类型	功能
位 31	EXTEN	0x0	rw	累计时钟延展超时使能 (Cumulative clock low extend timeout enable) 0: 关闭; 1: 开启。 SMBus 协议里面的 $T_{LOW:SEXT} / T_{LOW:MEXT}$ 。
位 30: 28	保留	0x0	res	保持默认值。
位 27: 16	EXTTIME[11:0]	0x000	rw	累计时钟延展超时时间 (Cumulative clock low extend timeout value) 超时时间 = $(EXTTIME + 1) \times 2048 \times T_{I2C_CLK}$ 。 注：只有当 EXTEN 位被开启时(EXTEN=1)才能设置这些位。
位 15	TOEN	0x0	rw	时钟电平超时检测使能 (Detect clock low/high timeout enable) 0: 关闭; 1: 开启。 SMBus 协议里面的 $T_{TIMEOUT}$ 。
位 14: 13	保留	0x0	res	保持默认值。
位 12	TOMODE	0x0	rw	时钟电平超时检测模式 (Clock timeout detection mode) 0: 检测低电平; 1: 检测高电平。 注：只有当 TOEN 位被开启时(TOEN =1)才能设置这些位。
位 11: 0	TOTIME[11:0]	0x000	rw	时钟电平超时检测时间 (Clock timeout detection time) 当检测低电平 (TOMODE = 0) 时： 超时时间 = $(TOTIME + 1) \times 2048 \times T_{I2C_CLK}$ 。 当检测高电平 (TOMODE = 1) 时： 超时时间 = $(TOTIME + 1) \times 4 \times T_{I2C_CLK}$ 。 注：只有当 TOEN 位被开启时(TOEN =1)才能设置这些位。

11.7.7 状态寄存器 (I2C_STS)

域	简称	复位值	类型	功能
位 31: 24	保留	0x00	res	保持默认值。
位 23: 17	ADDR[6: 0]	0x00	r	从机地址匹配值 (Slave address matching value) 7 位地址下：从机接收到的地址。 10 位地址下：从机接收到的 10 位地址头。
位 16	SDIR	0x0	r	从机数据传输方向 (Slave data transmit direction) 0: 接收数据; 1: 发送数据。
位 15	BUSYF	0x0	r	总线忙标志 (Bus busy flag transmission mode) 0: 空闲; 1: 忙。 当检测到 START 条件置起，检测到停止条件自动清零。
位 14	保留	0x00	res	保持默认值。
位 13	ALERTF	0x0	r	SMBus 提醒标志 (SMBus alert flag) SMBus 主机指示提醒信号接收 (ALERT 引脚由高变低) 0: 未收到; 1: 收到。
位 12	TMOUT	0x0	r	SMBus 超时标志 (SMBus timeout flag) 0: 未超时; 1: 超时。
位 11	PECERR	0x0	r	PEC 接收错误标志 (PEC receive error flag) 0: 正确; 1: 错误。
位 10	OUF	0x0	r	过载或者欠载标志 (Overflow or underflow flag)

				当传输方向为发送数据时： 0: 正常; 1: 欠载。 当传输方向为接收数据时： 0: 正常; 1: 过载。
位 9	ARLOST	0x0	r	仲裁丢失标志 (Arbitration lost flag) 0: 正常; 1: 仲裁丢失。
位 8	BUSERR	0x0	r	总线错误标志 (Bus error flag) 0: 正常; 1: 错误。
位 7	TCRLD	0x0	r	传输完成, 等待加载数据 (Transmission is complete, waiting to load data) 0: 未完成; 1: 已完成。 当数据被发送完成 (CNT = 1), 并且使能了重载模式 (RLDEN=1) 时被置起, 当写入 CNT 值时自动清零。在主机模式或者当从机在 SCTRL=1 时使用。
位 6	TDC	0x0	r	数据传输完成标志 (Transmit data complete flag) 0: 未完成 (移位寄存器还有数据); 1: 已完成 (移位寄存器空, 数据已经完全发送到总线上)。 当在软件结束模式, 并且数据传输完成后置起 (ASTOPEN = 0, RLDEN = 0, CNT = 0)。 收到 START 或者 STOP 条件后自动清零。
位 5	STOPF	0x0	r	停止条件产生完成标志 (Stop condition generation complete flag) 0: 未产生; 1: 已产生。
位 4	ACKFAILF	0x0	r	应答失败标志 (Acknowledge failure flag) 0: 正常; 1: 失败。
位 3	ADDRF	0x0	r	地址匹配标志 (0~7 bit address match flag) 0: 未匹配; 1: 已匹配。
位 2	RDBF	0x0	r	接收数据缓冲器满 (Receive data buffer full flag) 0: 数据寄存器 (DT) 未接收到数据; 1: 数据寄存器 (DT) 接收到数据。
位 1	TDIS	0x0	rw1s	发送中断状态 (Transmit data interrupt status) 0: 数据已写入 I2C_TXDT; 1: 数据已从 I2C_TXDT 发送到移位寄存器, I2C_TXDT 为空, 这时必须把要发的数据写到 I2C_TXDT 寄存器。在禁止时钟延展模式下, 可以向此位写 1, 以生成一个 TDIS 事件以便提前写数据到 I2C_TXDT 寄存器。
位 0	TDBE	0x1	rw1s	发送数据寄存器空标志 (Transmit data buffer empty flag) 0: I2C_TXDT 有数据, 不为空; 1: I2C_TXDT 无数据, 为空。 该位只用以表示当前 I2C_TXDT 的状态, 并可以通过软件写 1, 可清除 I2C_TXDT 寄存器里的数据。

11.7.8 状态清除寄存器 (I2C_CLR)

域	简称	复位值	类型	功能
位 31: 14	保留	0x00000	res	保持默认值。
位 13	ALERTC	0x0	w	清除 SMBus 提醒标志 (Clear SMBus alert flag) 写 1 清除。
位 12	TMOUTC	0x0	w	清除 SMBus 超时标志 (Clear SMBus timeout flag) 写 1 清除。
位 11	PECERRC	0x0	w	清除 PEC 接收错误标志 (Clear PEC receive error flag) 写 1 清除。
位 10	OUFC	0x0	w	清除溢出标志 (Clear overload / underload flag) 写 1 清除。
位 9	ARLOSTC	0x0	w	清除仲裁丢失标志 (Clear arbitration lost flag) 写 1 清除。
位 8	BUSERRC	0x0	w	清除总线错误标志 (Clear bus error flag) 写 1 清除。
位 7: 6	保留	0x0	res	保持默认值。
位 5	STOPC	0x0	w	清除停止条件产生完成标志 (Clear stop condition generation complete flag) 写 1 清除。
位 4	ACKFAILC	0x0	w	清除应答失败标志 (Clear acknowledge failure flag) 写 1 清除。
位 3	ADDRC	0x0	w	清除地址匹配标志 (Clear 0~7 bit address match flag) 写 1 清除。
位 2: 0	保留	0x0	res	保持默认值。

11.7.9 PEC寄存器 (I2C_PEC)

域	简称	复位值	类型	功能
位 31: 8	保留	0x0000000	res	保持默认值。
位 7: 0	PECVAL[7: 0]	0x00	r	PEC 值 (PEC value)

注意：当 I2C 被关闭时(I2CEN=0)会复位 PEC 寄存器(I2C_PEC)。

11.7.10 接收寄存器 (I2C_RXDT)

域	简称	复位值	类型	功能
位 31: 8	保留	0x0000000	res	保持默认值。
位 7: 0	DT[7: 0]	0x00	r	接收数据寄存器 (Receive data register)

注意：当 I2C 被关闭时(I2CEN=0)会复位接收寄存器(I2C_RXDT)。

11.7.11 发送寄存器 (I2C_TXDT)

域	简称	复位值	类型	功能
位 31: 8	保留	0x0000000	res	保持默认值。
位 7: 0	DT[7: 0]	0x00	rw	发送数据寄存器 (Transmit data register)

12 通用同步异步收发器 (USART)

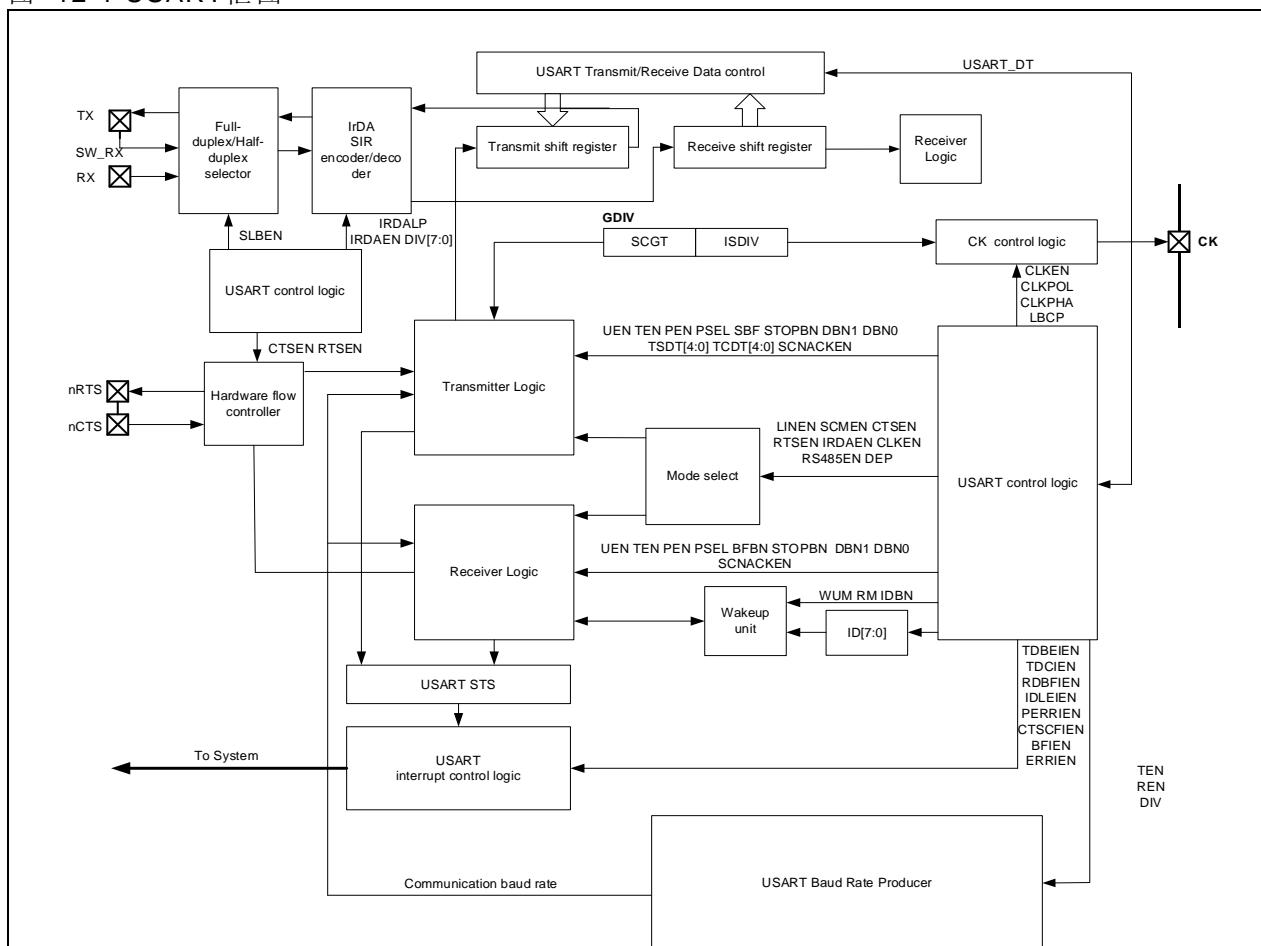
12.1 USART介绍

通用同步异步收发器 (USART) 是一个能通过多种不同的配置与使用不同的数据格式的外设进行通信的通用接口，同时支持异步全双工，异步半双工以及同步传输。USART 提供了可编程的波特率发生器，根据系统频率以及分频系数的不同，用户可以通过配置系统时钟以及分频系数以此产生所需要的特定通信频率。

USART 除了支持标准的 NRZ 异步以及同步收发通信协议外，还支持一些常用的其他类型的串行通信协议，如 LIN(局域互联网)，IrDA (红外数据组织) SIRENDEC 规范，ISO7816-3 标准的异步智能卡协议，以及 CTS/RTS (Clear To Send/Request To Send) 硬件流操作，RS485 和 Modbus。

USART 还支持多处理器通信，以及可配置通过空闲帧或地址匹配唤醒的静默模式，以此搭建 USART 网络，并且同时支持使用 DMA 进行数据的收发，以此实现高速通信。

图 12-1 USART 框图



USART 主要特性如下所列：

- 可编程配置的全双工或半双工通信
 - 全双工异步通信
 - 单线半双工通信
- 可编程配置的通信模式
 - NRZ 标准格式 (Mark/Space)
 - LIN (局域互联网)
 - IrDA SIR (串行红外)
 - ISO7816-3 标准里定义的异步智能卡协议： 智能卡模式支持 0.5 或 1.5 个停止位
 - RS-232 CTS/RTS (Clear To Send/Request To Send) 硬件流操作
 - 通过静默模式实现多处理器通信 (具有地址匹配和总线空闲两种可编程配置的唤醒方式)
 - 同步模式
- 可编程配置的波特率发生器
 - 发送和接收共用的可编程波特率
- 可编程配置的帧格式
 - 可编程的数据位位数 (7 位或 8 位或 9 位)
 - 可编程的停止位位数-支持 1 或 2 个停止位
 - 可编程的校验控制：发送方具备发送校验位的能力，接收方具备对接收到的数据进行校验的能力
 - 可编程配置的数据发送顺序 (MSB/LSB)
 - 可编程配置的 Tx/Rx 引脚极性
 - 可编程配置的 DT 数据极性
- 可编程配置的 DMA 多缓冲器通信
- 可编程配置的独立的发送器和接收器使能位
- 可编程配置的输出 CLK 的相位和极性以及频率
- 检测标志
 - 接收缓冲器满
 - 发送缓冲器空
 - 传输结束标志
- 四个错误检测标志
 - 溢出错误
 - 噪音错误
 - 帧错误
 - 校验错误
- 可编程配置的 12 个带标志的中断源
 - CTS 电平改变
 - LIN 断开符检测
 - 发送数据寄存器空
 - 发送完成
 - 接收数据寄存器满
 - 检测到总线为空闲
 - 溢出错误
 - 帧错误
 - 噪音错误
 - 校验错误
 - 接收器超时检测
 - 字节匹配检测

12.2 全双工半双工选择器简述和配置流程

USART 全双工半双工选择器通过软件编程配置相应寄存器的方式，使得 USART 可以采用全双工或半双工的方式和外设进行数据交换。

USART 默认选择使用双线单向全双工时，TX 引脚用于数据输出，RX 引脚用于数据输入，USART 接收器和发送器相互独立，这使得 USART 可以同时进行数据发送和数据接收，以此实现全双工通信。

USART 在 SLBEN 位置 1 时选择使用单线双向半双工的方式进行数据通信，在此条件下，LINEN 位，CLKEN 位，SCMEN 位以及 IRDAEN 位需置 0，此时在 USART 内部，RX 引脚无效，TX 引脚和 SW_RX 引脚互连，对 USART 来说，TX 引脚用于数据输出，SW_RX 用于数据输入，对外设来说，数据都从 TX 引脚映射的 IO 双向传输。

12.3 模式选择器简述和配置流程

12.3.1 模式选择器简述

USART 模式选择器通过软件编程配置相应寄存器的方式，使得 USART 可以根据软件的不同配置工作在不同的工作模式下，以此能与使用不同通信协议的外设之间实现数据交换。

USART 默认支持 NRZ 标准格式（Mark/Space），根据 USART 模式选择器配置的不同，USART 还可以支持 LIN（局域互联网），IrDA SIR（串行红外），ISO7816-3 标准里定义的异步智能卡协议，RS-232 CTS/RTS（Clear To Send/Request To Send）硬件流操作以及静默模式和同步模式。

12.3.2 模式选择器配置方法

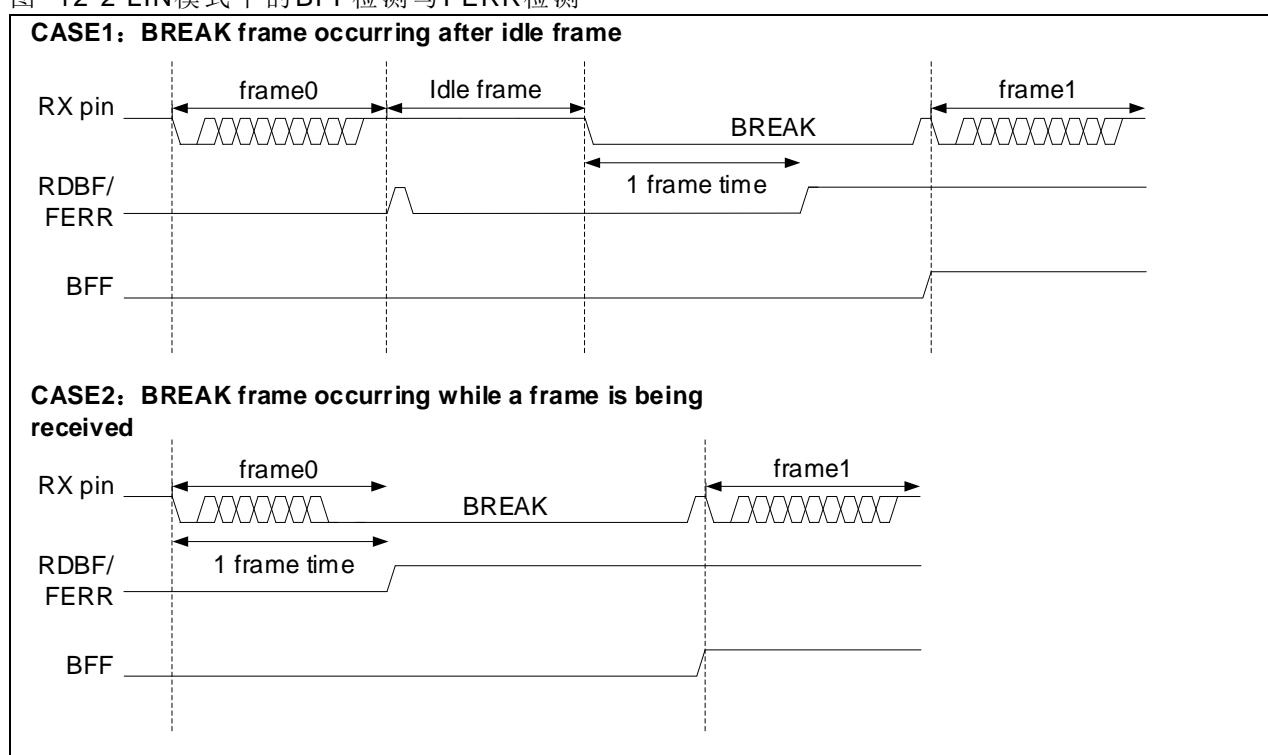
用户可以通过不同的配置以此选择不同的工作模式，配置方法分列如下所列，请将如下配置方法配合本章后述的接收器和发送器配置方法结合使用以完成 USART 初始化配置。

1. LIN模式

基础设置：LINEN位置1，CLKEN位置0，STOPBN[1: 0]位置0，SCMEN位置0，SLBEN位置0，IRDAEN位置0，DBN[1: 0]=00。

LIN主机有发送间隔帧的能力，可以使用SBF位置1发送13位低电平的LIN同步间隔帧。同时LIN从机也有检测间隔帧的能力，可以选择BFBN位置1或0来选择是11位还是10位间隔帧检测。

图 12-2 LIN模式下的BFF检测与FERR检测



2. 智能卡模式

基础设置：SCMEN位置1，LINEN位置0，SLBEN位置0，IRDAEN位置0，CLKEN位置1，DBN[1: 0]=01，PEN位置1，STOPBN[1: 0]=11。

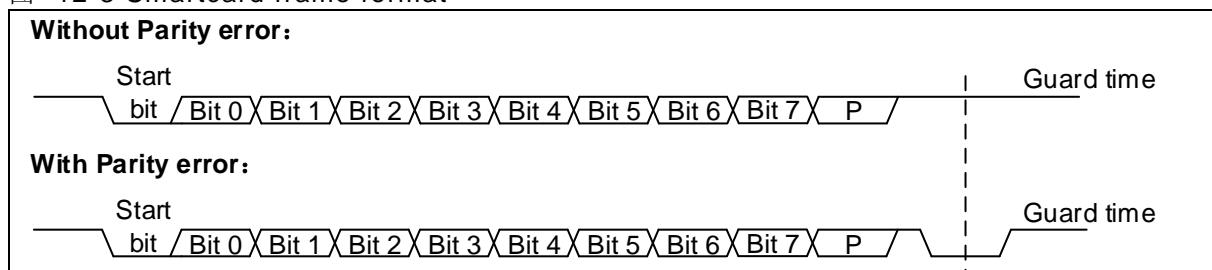
可以选择配置CLKPOL位和CLKPH位以及LBCP位以满足不同的时钟极性以及时钟相位和时钟脉冲个数，具体可见同步模式部分。

通过配置SCGT[7: 0]位选择保护时间，使TDC标志的置起可以得到延时，直到保护时间计数器向上计数到SCGT[7: 0]的值，TDC才得以置起。

而智能卡属于单线双向半双工通信，可以通过配置SCNACKEN位选择是否在校验出错时发送NACK，以告知数据没有被正确接收。

注意：发送和接收时，都只可配置为1.5个停止位，且必须确保智能卡保护时间皆满足1.5位长才能正常接收连续数据。

图 12-3 Smartcard frame format

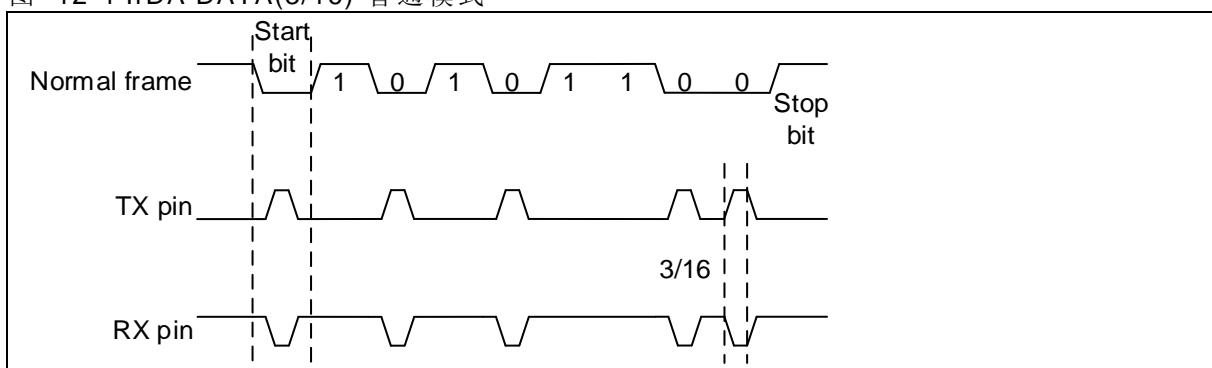


3. 红外模式

基础设置：IRDAEN位置1，CLKEN位置0，STOPBN[1: 0]位置0，SCMEN位置0，SLBEN位置0。

可以选择IRDALP位置1以开启红外低功耗模式，在普通模式下持续时间为3/16位，在红外低功耗模式下位持续时间可调，并配合ISDIV[7: 0]配置想要产生的低功耗频率。

图 12-4 IrDA DATA(3/16)-普通模式



4. Modbus

USART仅提供实现Modbus/RTU和Modbus/ASCII所需的基础的硬件支持，这意味着协议的控制部分必须由软件完成，USART只是提供块结束侦测。

在Modbus/RTU中，块结束侦测通过可编程的超时功能识别接收线为空闲状态的时间大于2个字节时间实现，用户可以通过配置RTOV寄存器设定需要的超时值（时间单位是1bit位宽），通过RTODEN位置1开启超时检测，当USART接收器检测到接收线为空闲的时间等于设定的超时值时，USART会置起RTODF，如果RTODIE位置1，产生中断，可以通过RTODCF位写1清除RTODF位；

在Modbus/ASCII中，块结束的侦测通过字节匹配功能识别特殊的字节序列（CR/LF），通过软件编程将LF ASCII码写入ID[7: 0]，默认开启字符匹配功能，当USART接收器接收到的数据和ID[7: 0]匹配时，USART会置起CMDF，如果CMDIE位置1，产生中断，可以通过CMDCF位写1清除CMDF位。

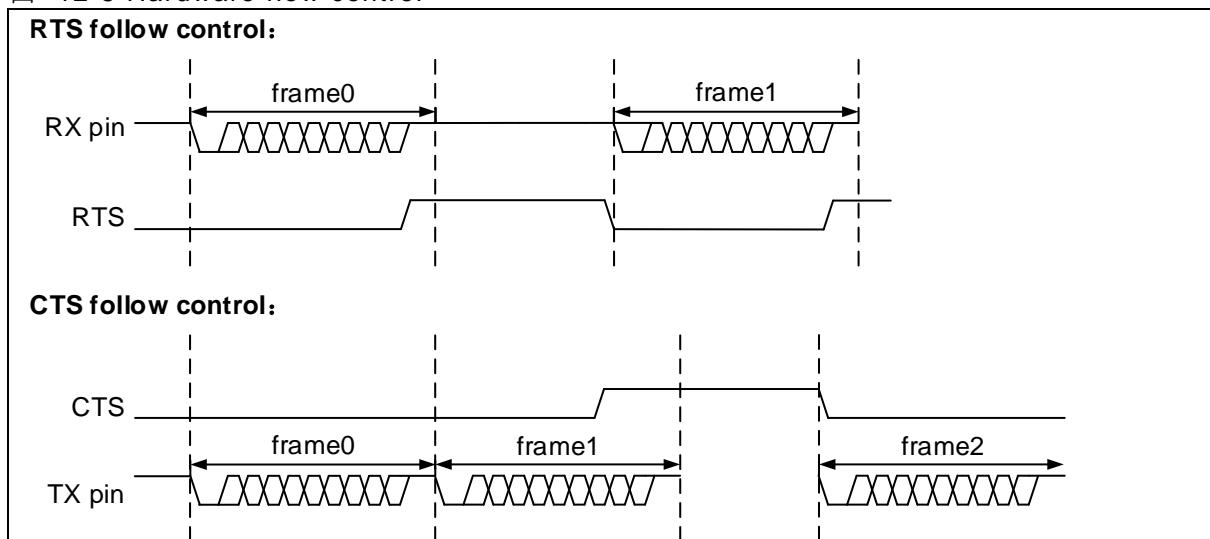
5. 硬件流控制模式

RTSEN位置1和CTSEN位置1可以分别开启RTS和CTS流控制。

RTS流控制：USART接收器准备好接收新的数据，RTS就变成有效（下拉为低电平）。当接收寄存器内有数据到达时（在每个stop位开始时），RTS被置位，由此表明希望在当前帧结束时停止数据传输。

CTS流控制：USART发送器在发送下一帧前检查CTS输入。如果CTS有效（也即CTS为低电平），则下一个数据被发送；若CTS在传输期间被变成无效（也即CTS为高电平），当前的传输完成后停止发送。

图 12-5 Hardware flow control



6. RS485模式

RS485EN位置1开启RS485模式，驱动使能信号从RTS引脚输出，用户可以通过配置DEP位选择DE信号的极性，用户可以通过分别配置TSDT[4: 0]和TCDT[4: 0]设置发送器开始发送起始位前延迟的时间，和发送器发送完最后一笔数据的停止位后置起TC标志前延迟的时间。

7. 静默模式

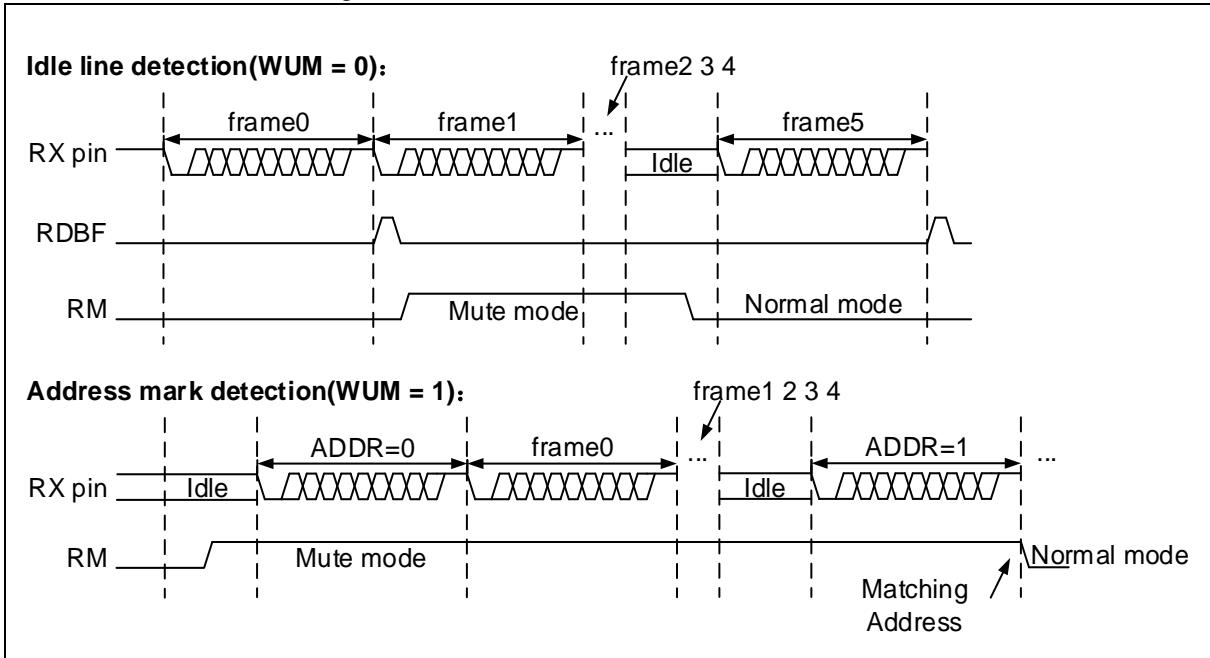
RM位置1进入静默模式，根据WUM位置1和置0，可以分别通过ID匹配和空闲总线从静默模式中唤醒，其中ID号ID[7: 0]可编程配置，并且可以通过配置IDBN选择使用ID[7: :0]或ID[3: 0]，当选择ID匹配时，数据位的MSB为1表示当前数据是ID。

关闭奇偶校验功能时，当DBN[1: 0]=10时，MSB是USART_DT[6]，当DBN[1: 0]=00时，MSB是USART_DT[7]，当DBN[1: 0]=01时，MSB是USART_DT[8]。

开启奇偶校验功能时，当DBN[1: 0]=10时，MSB是USART_DT[5]，当DBN[1: 0]=00时，MSB是USART_DT[6]，当DBN[1: 0]=01时，MSB是USART_DT[7]。

当选择ID[3: 0]时，数据位的4个LSB表示ID值；当选择ID[7: 0]时，数据位除上述奇偶校验位以及MSB位以外，所有的LSB位表示ID值。

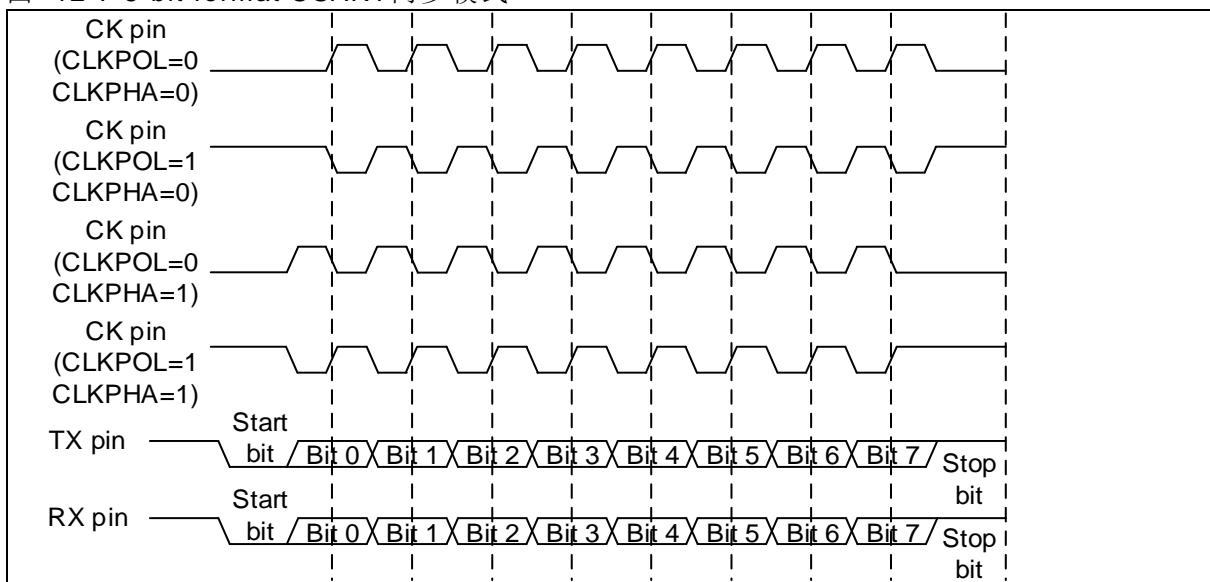
图 12-6 Mute mode using Idle line or Address mark detection



8. 同步模式

CLKEN位置1开启同步模式并使能时钟引脚输出，通过配置CLKPOL位置1或0可以选择空闲状态下CK引脚上的电平为高或低，通过配置CLKPHA位置1或0可以选择在时钟的第二个或第一个边沿开始采样数据，通过配置LBCP位置1或0可以选择最后一位数据是否输出时钟，通过配置ISDIV[4: 0]可以选择想要输出的时钟频率。

图 12-7 8-bit format USART 同步模式



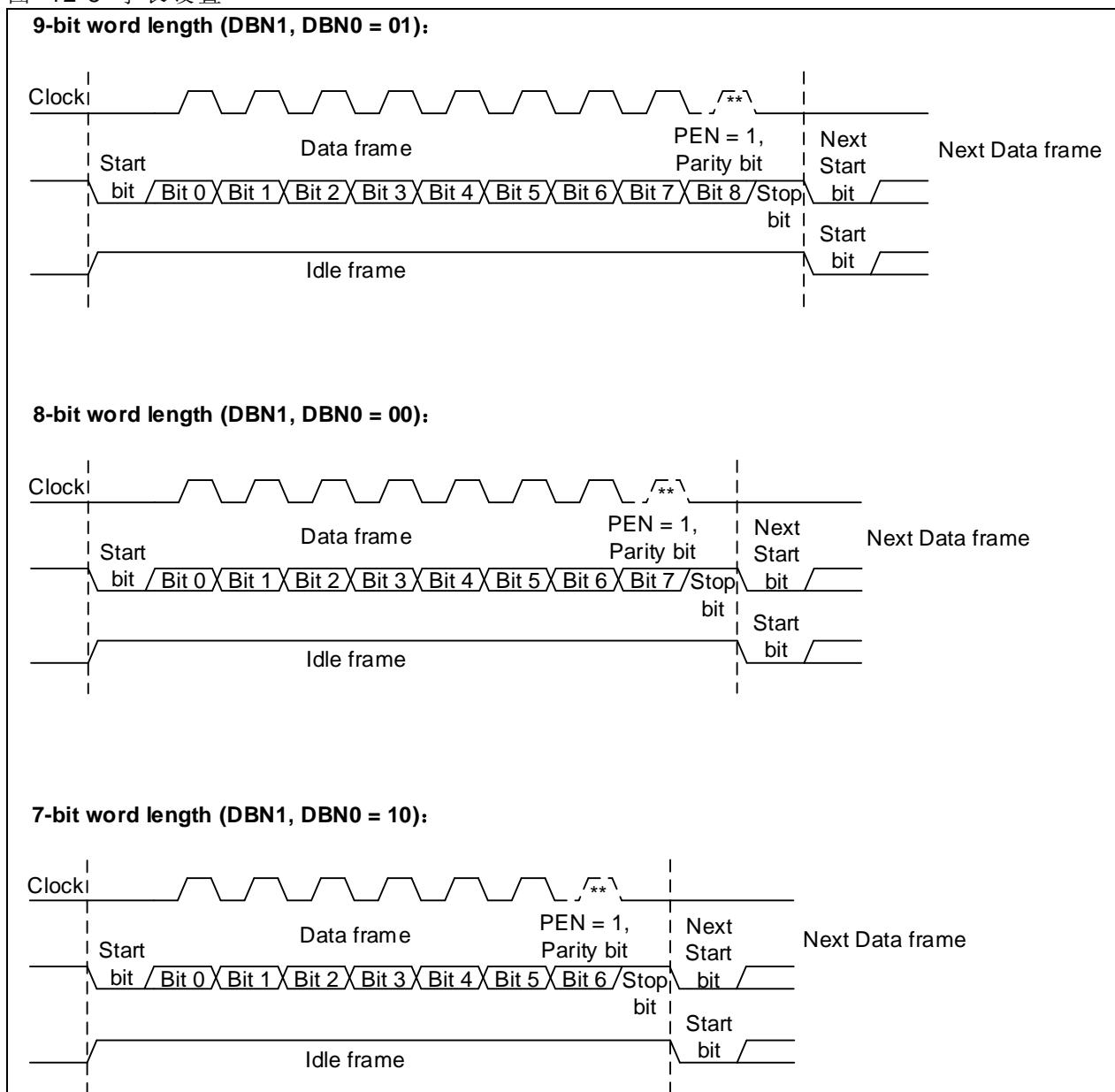
12.4 USART帧格式简述和配置流程

USART一笔数据帧由起始位，数据位，停止位依次组成，最后一位数据位可以作为校验位。

USART一笔空闲帧的长度等于当前配置下数据帧的长度，但所有位都为1。

USART一笔断开帧的长度等于当前配置下数据帧的长度加上停止位，停止位之前的所有位都等于0。需要特别注意的是，在非 LIN 模式下，发送和检测断开帧的长度都需遵守此规则，例 DBN[1: 0]=00，那么发送和检测的断开帧长度就是 10 位低电平加停止位。LIN 模式请参考模式选择器简述和配置流程部分。通过 DBN1 位和 DBN0 配置 7 位（DBN[1: 0]=10）8 位（DBN[1: 0]=00）或 9 位（DBN[1: 0]=01）数据位。

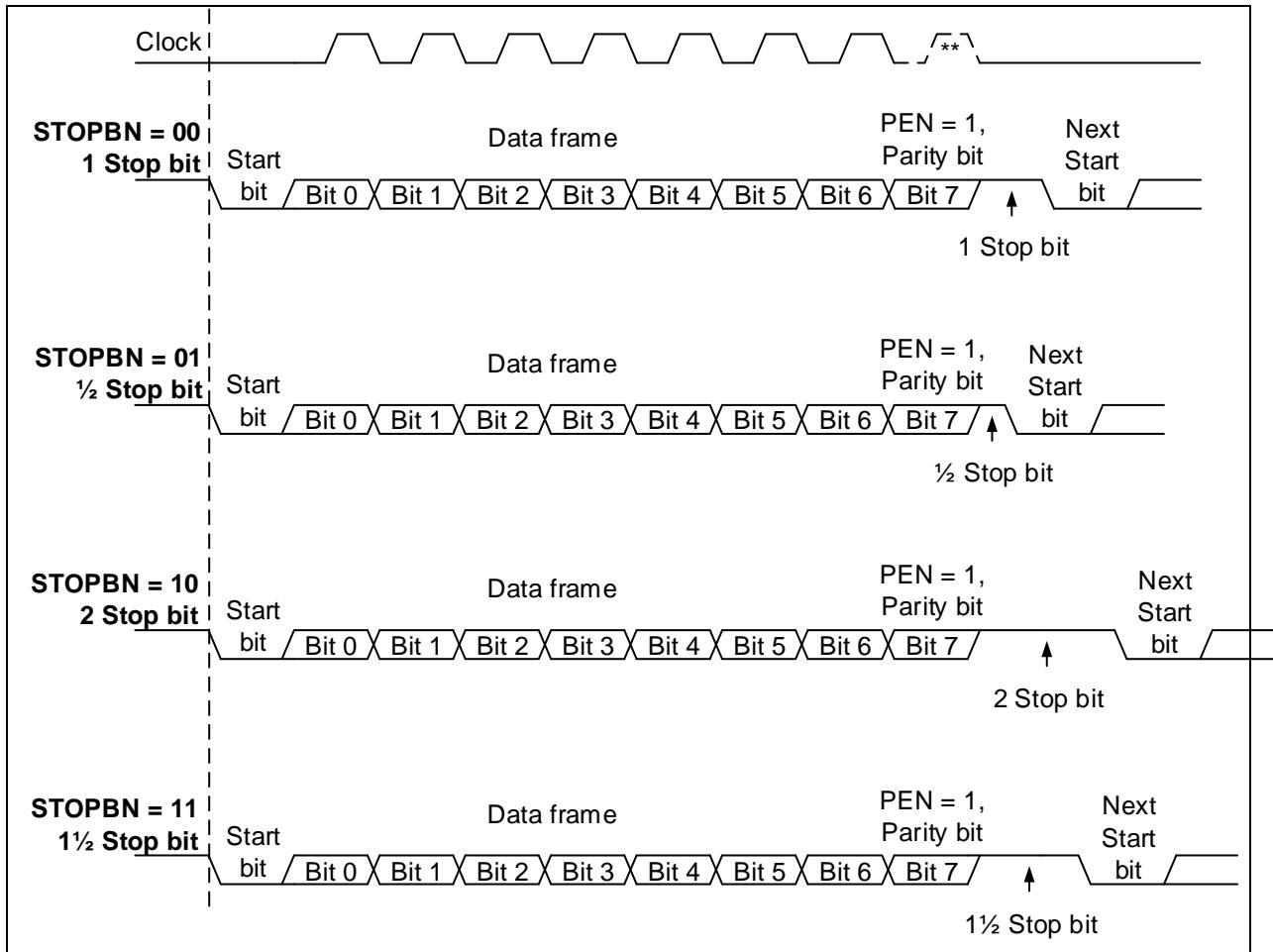
图 12-8 字长设置



通过 STOPBN 位配置 1 位 (STOPBN=00)，0.5 位 (STOPBN=01)，2 位 (STOPBN=10)，1.5 位 (STOPBN=11) 停止位。

通过 PEN 位置“1”配置校验控制使能，通过 PSEL 位配置奇校验 (PSEL=1) 或偶校验 (PSEL=0)，校验控制使能后数据位的 MSB 将由奇偶校验位替代，即有效数据位减少一位。

图 12-9 配置停止位



通过 MTF 位配置数据是先传输 MSB (MTF=1) 还是 LSB (MTF=0)。

通过 DTREV 位配置 USART_DT 是以 1=L,0=H (DTREV=1) 还是 0=L,1=H (DTREV=0) 的方式发送和接收。

通过 TXREV 位配置 USART_TX 引脚上的信号是以 VDD=0/mark,Gnd=1/idle (TXREV=1) 还是 VDD=1/idle,Gnd=0/mark (TXREV=0) 的方式传输。

通过 RXREV 位配置 USART_RX 引脚上的信号是以 VDD=0/mark,Gnd=1/idle (RXREV=1) 还是 VDD=1/idle,Gnd=0/mark (RXREV=0) 的方式传输。

12.5 DMA 传输简述和配置流程

USART 可以使用 DMA 操作发送数据缓冲器和接收数据缓冲器以实现高速连续传输，USART 的 DMA 传输需要配合 DMA 使用，下方会简述配置流程，但具体和 DMA 配置相关部分请参见 DMA 章节的描述。

12.5.1 DMA 发送配置流程

1. 选择 DMA 传输通道：在 DMA 章节 DMA 通道映射表中选择用于当前所用 USART 的 DMA 通道。
2. 配置 DMA 传输目标地址：在 DMA 控制寄存器中 DMA 传输目的地址位写入当前所使用的 USART 的 USART_DT 寄存器地址，DMA 将会在接收到发送请求后将待发送的数据写入该地址。
3. 配置 DMA 传输源地址：在 DMA 控制寄存器中 DMA 传输源地址位写入待发送数据存放的地址，DMA 将会在接收到发送请求后将该地址内的数据写入到目标地址中，即写入到当前所使用的 USART 的 USART_DT 寄存器中。
4. 配置 DMA 传输字节个数：在 DMA 控制寄存器相关位置配置期望传输的字节个数。
5. 配置 DMA 传输通道优先级：在 DMA 控制寄存器相关位置配置当前所使用通道的 USART 的 DMA 传输通道优先级。
6. 配置 DMA 中断产生时机：在 DMA 控制寄存器相关位置配置是在传输完成或传输完成一半时产生 DMA 中断。

7. 使能DMA传输通道：在DMA控制寄存器相关位置使能当前所选用的DMA通道

12.5.2 DMA接收配置流程

1. 选择DMA传输通道：在DMA章节DMA通道映射表中选择用于当前所用USART的DMA通道。
2. 配置DMA传输目标地址：在DMA控制寄存器中DMA传输目的地址位写入期望存放接收数据的地址，DMA将会在接收到接收请求后，将当前所使用的USART的USART_DT寄存器中的数据存放在目的地址中。
3. 配置DMA传输源地址：在DMA控制寄存器中DMA传输源地址位写入当前所使用的USART的USART_DT寄存器的地址，DMA将会在接收到接收请求后将该地址内的数据写入到目标地址中，即写入到期望存放接收数据的地址。
4. 配置DMA传输字节个数：在DMA控制寄存器相关位置配置期望传输的字节个数
5. 配置DMA传输通道优先级：在DMA控制寄存器相关位置配置当前所使用通道的USART的DMA传输通道优先级。
6. 配置DMA中断产生时机：在DMA控制寄存器相关位置配置是在传输完成或传输完成一半时产生DMA中断。
7. 使能DMA传输通道：在DMA控制寄存器相关位置使能当前所选用的DMA通道

12.6 波特率发生器简述及配置流程

12.6.1 波特率发生器简述

USART 波特率发生器通过使用内部计数器，以 PCLK 为基准，DIV (USART_BAUDR[15: 0]) 即为该计数器的溢出值，该计数器计满一次代表一位数据，所以每位数据位宽为 DIV 个 PCLK 周期。

由于 USART 的接收器和发送器共用同一个波特率发生器，并且接收器将每位数据拆分为 16 份等长的部分以此来实现过采样，所以数据位宽不得小于 16 个 PCLK 周期，即 DIV 中的值必须大于或等于 16。

12.6.2 波特率发生器配置方法

用户可通过配置不同的系统时钟以及在 USART_BAUDR 中写入不同的值以此产生特定的波特率，具体的运算关系见如下公式

$$\text{TX/RX 波特率} = \frac{f_{CK}}{\text{DIV}}$$

这里的 f_{CK} 是指 USART 的系统时钟(PCLK1 用于 USART2、3、4、5 和 USART7、8, PCLK2 用于 USART1、6)

注：1. USART_BAUDR 中的值需要在 UEN 之前写入，且 UEN=1 时，不可更改这些位。

2. 关闭 USART 接收器或发送器会使内部计数器复位，波特率发生中断。

表 12-1 设置波特率时的误差计算

波特率			fPCLK=36MHz		fPCLK=72MHz		
序号	Kbps	实际	置于波特率寄存器中的值	误差%	实际	置于波特率寄存器中的值	误差%
1	2.4	2.4	15000	0%	2.4	30000	0%
2	9.6	9.6	3750	0%	9.6	7500	0%
3	19.2	19.2	1875	0%	19.2	3750	0%
4	57.6	57.6	625	0%	57.6	1250	0%
5	115.2	115.384	312	0.15%	115.2	625	0%
6	230.4	230.769	156	0.16%	230.769	312	0.16%
7	460.8	461.538	78	0.16%	461.538	156	0.16%
8	921.6	923.076	39	0.16%	923.076	78	0.16%
9	2250	2250	16	0%	2250	32	0%
10	4500	不可能	不可能	不可能	4500	16	0%

以波特率 115.2Kbps 为例，假设 fPCLK 为 36MHz，此时波特率寄存器应设置为 312(0x138)，经由公式计算： $36000000 / 312 = 115384 = 115.384\text{Kbps}$

而它们的误差计算为(实际值 - 理论值) / 理论值 * 100%: $(115.384 - 115.2) / 115.2 * 100\% = 0.15\%$

12.7 发送器简述和配置流程

12.7.1 发送器简述

USART 发送器具有独立的使能位 TEN，发送器与接收器共用同一个波特率且该波特率可编程配置，

USART 具有一个发送数据缓冲器 (TDR) 和一个发送移位寄存器，当发送数据缓冲器 (TDR) 为空时，TDBE 置起，如果设置了 TDBEIEN 将会产生中断。

软件写入的值会先存储在发送数据缓冲器 (TDR) 中，当发送移位寄存器为空时，USART 会将发送数据缓冲器中的值移入到发送移位寄存器，USART 发送器将以 LSB 的方式将发送移位寄存器中的数据从 TX 脚输出，具体的输出格式取决于软件配置的帧格式。

如若选择了同步传输或者配置了时钟输出，USART 发送器将时钟脉冲从 CK 脚输出，如若选择了硬件流控制，USART 发送器将控制信号将从 CTS 引脚输入。

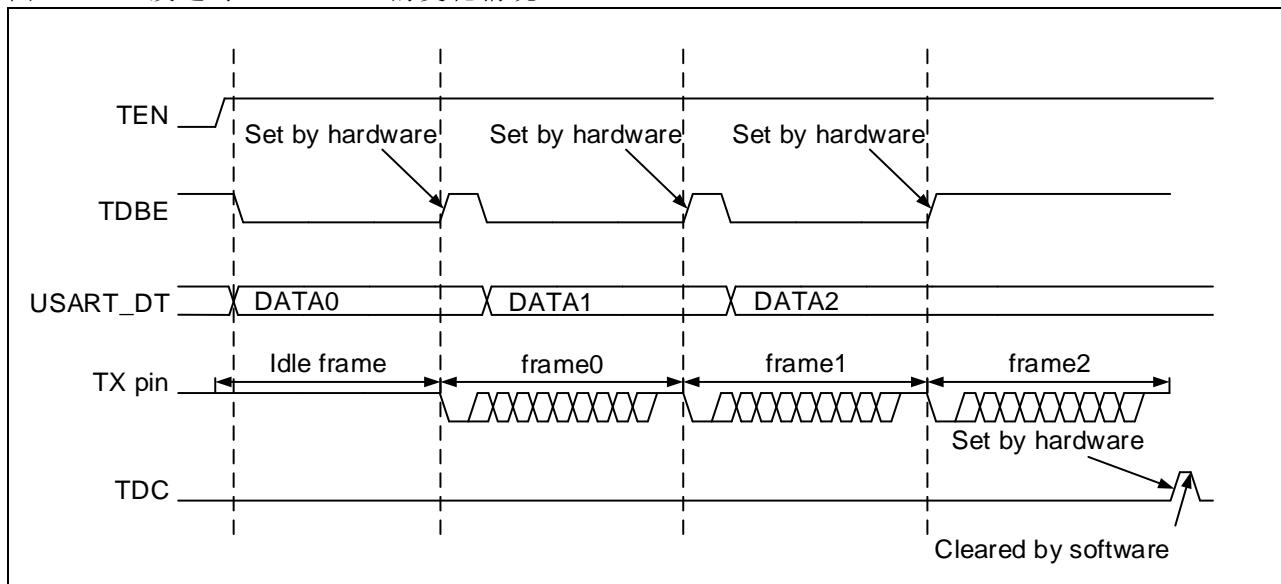
注意： 1. 在数据传输期间不能复位 TEN 位，否则将破坏 TX 脚上的数据。

2. TEN 位被激活后，USART 将自动发送一个空闲帧。

12.7.2 发送器配置流程

1. USART使能：UEN位置1。
2. 全双工半双工配置：具体参见全双工半双工选择器配置部分（12.2）。
3. 模式配置：具体参见模式选择器配置部分（12.3）。
4. 帧格式配置：具体参见帧格式配置部分（12.4）。
5. 中断配置：具体参见中断发生器配置部分（12.10）。
6. DMA发送配置：如果选择使用DMA发送，DMATEN位USART_CTRL3[7]置1，并按照DMA传输中的描述配置DMA寄存器。
7. 波特率配置：具体参见波特率发生器配置部分（12.6）。
8. 发送器使能：TEN位置1，置1后USART发送器会自动发送一个空闲帧。
9. 数据写入：等待TDBE位置起后，将要发送的数据写入USART_DT寄存器（此操作会清除TDBE位），在非DMA模式下，重复此操作。
10. 在写入最后一个期望传输的数据后，等待TDC位置起，这表示最后一个数据帧的传输结束，在该标志置起前，禁止关闭USART，否则传输可能出错。
11. 在TDC=1后，可以采用先读一次USART_STS寄存器，再写一次USART_DT寄存器的方式来清除TDC；也可以采用软件对它写‘0’来清除，但此方法只推荐在DMA模式下使用。

图 12-10 发送时 TDC/TDBE 的变化情况



12.8 接收器简述和配置流程

12.8.1 接收器简述

USART 接收器具有独立的接收器使能位 REN(USART_CTRL1[2])，接收器和发送器共用同一个波特率且该波特率可编程配置，USART 具有一个接收数据缓冲器 (RDR) 和一个接收移位寄存器。

数据从 USART 的 RX 脚输入，当接收器判断到一个有效的起始位后，接收器会以 LSB 的方式将接收到的数据依次移入接收移位寄存器，并根据软件配置的帧格式，在接收到一个完整的数据帧后将接收移位寄存器中的值移入接收数据缓冲器并置起 RDBF，如果设置了 RDBFIEN 将会产生中断。

如若选择了硬件流控制，USART 接收器将控制信号将从 RTS 引脚输出。

在数据接收过程中, USART 接收器会根据软件的配置检测帧错误, 溢出错误, 奇偶校验错误以及噪声错误, 并根据相应的中断使能位是否置位来判断是否产生相应的中断。

12.8.2 接收器配置流程

配置步骤:

1. USART 使能: UEN 位置 1。
2. 全双工半双工配置: 具体参见全双工半双工选择器配置部分 (12.2)。
3. 模式配置: 具体参见模式选择器配置部分 (12.3)。
4. 帧格式配置: 具体参见帧格式配置部分 (12.4)。
5. 中断配置: 具体参见中断发生器配置部分 (12.10)。
6. DMA 接收配置: 如果选择使用 DMA 接收, DMAREN 位置 1, 并按照 DMA 传输中的描述配置 DMA 寄存器。
7. 波特率配置: 具体参见波特率发生器配置部分 (12.6)。
8. 接收器使能: REN 位置 1。

当一字符被接收到时:

- RDBF 位被置位。它表明移位寄存器的内容被转移到 RDR (Receiver Data Register)。换句话说, 数据已经被接收并且可以被读出 (包括与之有关的错误标志)。
- 如果 RDBFIEN 位被设置, 则产生中断。
- 在接收期间如果检测到帧错误, 噪音或溢出错误, 错误标志将被置起。
- 在 DMA 传输时, RDBF 在每个字节接收后被置起, 并由 DMA 对数据寄存器的读操作而清零。
- 在非 DMA 传输时, 由软件读 USART_DT 寄存器完成对 RDBF 位清除。RDBF 标志也可以通过对它写 0 来清除。RDBF 位必须在下一帧数据接收结束前被清零, 以避免溢出错误。

当一个断开帧被接收到时:

- 非 LIN 模式: USART 接收器按照帧错误处理, 并置起 FERR 位, 若相应中断使能, 中断产生, 具体可见下方错误帧的描述。
- LIN 模式: USART 接收器按断开帧处理, 并置起 BFF 位, 若 BFIEN 置位, 则中断产生。

当一个空闲帧被接收到时:

- USART 接收器按数据帧处理, 并置起 IDLEF 位, 若 IDLEIEN 置位, 则中断产生。

当一个帧错误产生时:

- FERR 位置位。
- USART 接收器将错误的数据从接收移位寄存器转移到接收数据缓冲器。
- 在非 DMA 传输时, 这个位和 RDBF 位同时置起, 后者将产生中断。在 DMA 传输时, 如果 ERRIEN 置位的话, 将产生中断。

当一个溢出错误产生时:

- ROERR 位被置位。
- 接收数据缓冲器中的数据不会被覆盖, 读 USART_DT 寄存器仍能得到先前的数据。
- 接收移位寄存器中的内容会被覆盖, 随后接收到的数据都将丢失。
- 如果 RDBFIEN 位置位或 ERRIEN 和 DMAREN 位都被置位, 中断产生。
- 先读 USART_STS, 再读 USART_DT 寄存器, 可清除 ROERR。

注意: 当 ROERR 置位时, 表明至少有 1 个数据已经丢失。有两种可能性:

- 如果 RDBF=1, 上一个有效数据还存储在接收数据缓冲器中, 可以被读出。
- 如果 RDBF=0, 这意味着上一个有效数据已经从接收数据缓冲器中读走。

注意: 在接收数据时, REN 位不应该被复位。如果 REN 位在接收时被清零, 当前字节的接收被丢失。

12.8.3 起始侦测和噪声检测

USART 接收器在 REN 位置位后便开始侦测起始位, USART 接收器通过过采样技术, 在第 3、5、7、8、9、10 位共 6 个点进行数据采样, 以此侦测有效起始位以及识别噪声, 具体的噪声和有效起始位的判别方式可以参见下方检测起始位和噪声的数据采样。

表 12-2 检测起始位和噪声的数据采样

采样值 (3 • 5 • 7)	采样值 (8 • 9 • 10)	NERR 位	起始位有效性
000	000	0	有效
001/010/100	001/010/100	1	有效
001/010/100	000	1	有效
000	001/010/100	1	有效
111/110/101/011	任意值	0	无效
任意值	111/110/101/011	0	无效

注意：如果在第 3、5、7、8、9、10 位的采样值满足不了上表任意一种组合，则 USART 接收器认为没有接受到正确的起始位，将退出起始位侦测并回到空闲状态等待下降沿。

USART 接收器具备噪声检测功能，在非同步模式时，使用过采样技术，在第 7、8、9 采样点，根据不同的采样值，区别有效输入数据和噪音，并恢复数据和置起噪声错误标志位 NERR。具体的采样方法以及噪声和有效数据的判别方式可以参见下方检测有效数据和噪声的数据采样。

表 12-3 检测有效数据和噪声的数据采样

采样值	NERR 位	接收的位	数据有效性
000	0	0	有效
001	1	0	无效
010	1	0	无效
011	1	1	无效
100	1	0	无效
101	1	1	无效
110	1	1	无效
111	0	1	有效

USART 接收器在最大允许偏差下，皆可以正常接收数据，其值取决于 USART_CTRL1 的 DBN[1: 0]以及 USART_BAUDR 的 DIV[3: 0]。

注意：以下表格的最大允许偏差是以波特率 115.2Kbps 为基准进行计算，实际接收器最大允许偏差会随着波特率设定大小有所改变，波特率越大时其最大允许偏差会越小，反过来波特率越小其最大允许偏差会越大。

表 12-4 最大允许偏差

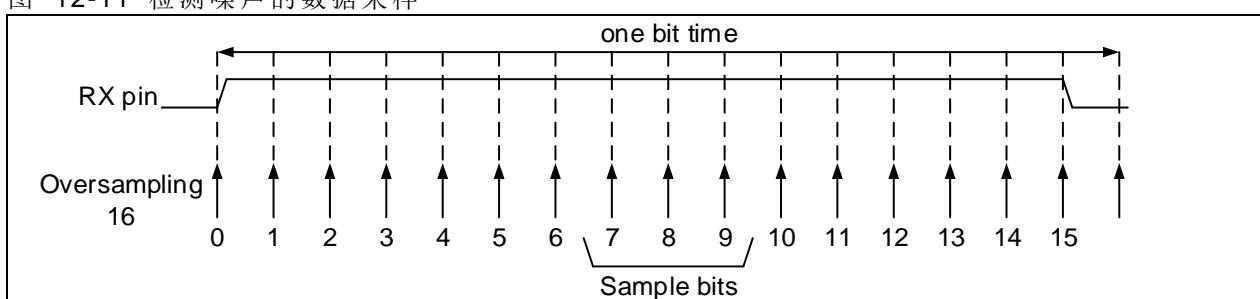
DBN[1:0]	DIV[3:0] = 0	DIV[3:0] != 0
00	3.75%	3.33%
01	3.41%	3.03%
10	4.16%	3.7%

当 USART 接收器在数据帧中检测到噪音时：

- 在 RDBF 位置起的同时置起 NERR 位。
- USART 接收器将错误数据从接收移位寄存器转移到接收数据缓冲器。
- 在非 DMA 传输时，没有噪声中断产生。然而，因为 NERR 位和 RDBF 位是同时置位，RDBF 将产生中断。在 DMA 传输时，如果 ERRIEN 位置位，中断产生。

先读 USART_STS，再读 USART_DT 寄存器，将清除 NERR 位。

图 12-11 检测噪声的数据采样



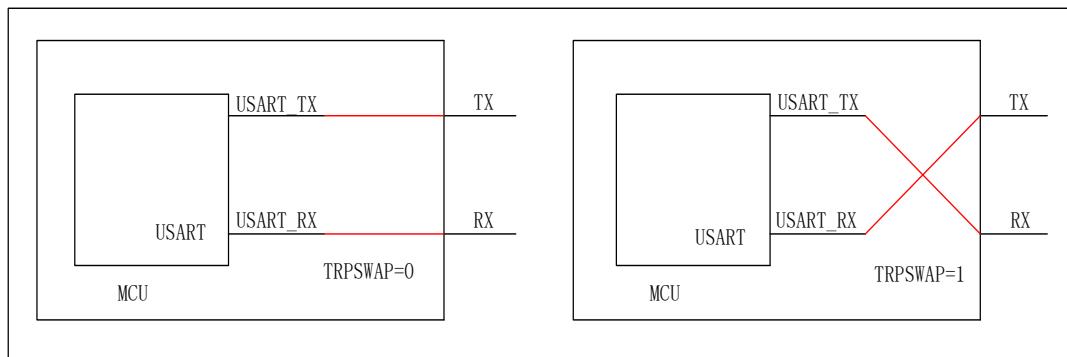
12.9 Tx/Rx 可配置引脚互换

如果 TRPSWAP (USART_CTRL2[15]) 位被使能，MCU 的 Tx/Rx 引脚顺序将被交换。以下举例两种常见应用场景：

- 若用户在外接 RS-232 芯片时不慎将 Tx/Rx 接反，可通过修改 TRPSWAP 位更换引脚顺序，无需修改硬件连接。

- 若用户在全双工模式下只将主机的 Tx 和从机的 Rx 连接，在主机和从机互换后，也可通过 TRPSWAP 位更换引脚顺序，无需修改硬件连接。

图 12-12 Tx/Rx 可配置引脚互换



注意：SWAP (USART_CTRL2[15]) 位必须在 USART 未被使能 (UEN=0) 时才能被改写

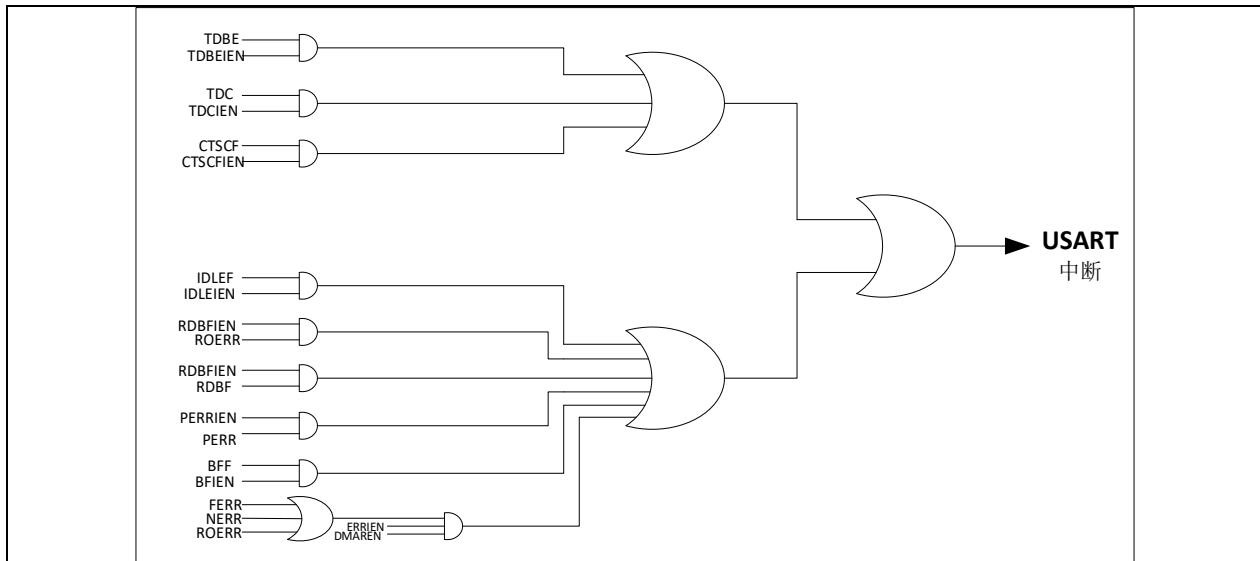
12.10 中断

USART 中断发生器是 USART 中断的控制中枢，USART 中断产生器会实时监测 USART 内部的中断源，并根据软件配置的相应中断源的中断使能位，以此决定是否产生中断，下表所示为 USART 的中断源以及相应的中断使能位，对相应的中断使能位置 1 时，即可在相应事件出现后产生中断。

表 12-5 USART 中断请求

中断事件	事件标志	使能位
发送数据寄存器空	TDBE	TDBEIEN
CTS 标志	CTSCF	CTSCFIEN
发送完成	TDC	TDCIEN
接收数据就绪可读	RDBF	RDBFIEN
检测到数据溢出	ROERR	
检测到空闲线路	IDLEF	IDLEIEN
奇偶检验错	PERR	PERRIEN
断开标志	BFF	BFIEN
噪声标志，多缓冲通信中的溢出错误和帧错误	NERR 或 ROERR 或 FERR	ERRIEN (1)
接收器超时检测	RTODF	RETODIE
字节匹配检测	CMDF	CMDIE

图 12-13 USART中断映像图



12.11 I/O 引脚控制

USART 通过五个接口外部设备进行通信，引脚定义如下：

RX: 串行数据输入端。

TX: 串行数据输出端。在单线半双工模式和智能卡模式里，TX 脚作为 I/O 使用，即用于发送数据也用于接收数据。

CK: 发送器时钟输出。输出的 CLK 相位和极性以及频率均可编程配置。

CTS: 发送器输入端，硬件流控制模式发送使能信号。

RTS: 接收器输出端，硬件流控制模式发送请求信号。

12.12 USART 寄存器描述

表 12-6 USART 寄存器映像和复位值

寄存器简称	基址偏移量	复位值
USART_STS	0x00	0x00C0
USART_DT	0x04	0x0000
USART_BAUDR	0x08	0x0000
USART_CTRL1	0x0C	0x0000
USART_CTRL2	0x10	0x0000
USART_CTRL3	0x14	0x0000
USART_GDIV	0x18	0x0000
USART_RTOV	0x1C	0x0000
USART_IFC	0x20	0x0000

12.12.1 状态寄存器 (USART_STS)

域	简称	复位值	类型	功能
位 31: 18				
位 16: 12	保留位	0x000000	resd	硬件强制为 0。
位 10				
位 17	CMDF	0	r	字节匹配检测标志 当接收到由 ID[7:0] 定义的字节时，该位被硬件置起，由软件将其清零。 0: 无； 1: 有。
位 11	RTODF	0	r	接收器超时检测标志

				当超时值达到 RTOV 寄存器编程的值，若无任何通信，该位被硬件置起，由软件将其清零。 0: 无； 1: 有。
位 9	CTSCF	0x0	rw0c	CTS 变化标志 (CTS change flag) 当 CTS 线发送变化时，该位被硬件置起，由软件将其清零。 0: 无； 1: 有。
位 8	BFF	0x0	rw0c	间隔帧标志 (break frame flag) 当检测到间隔帧时，该位被硬件置起，由软件将其清零。 0: 无； 1: 有。
位 7	TDBE	0x1	ro	发送缓冲器空 (Transmit data buffer empty) 当发送缓冲器为空，可以再次写入数据时，该位被硬件置起。对 USART_DT 的写操作，将清零该位。 0: 非空； 1: 空。
位 6	TDC	0x1	rw0c	发送数据完成 (Transmit data CMplete) 当发送数据完成，该位被硬件置起，由软件将其清零（方式 1：先读 USART_STS，再写 USART_DT；方式 2：操作该位写'0'）。 0: 未完成； 1: 完成。
位 5	RDBF	0x0	rw0c	接收数据缓冲器满 (Receive data buffer full) 当接收到数据时，该位被硬件置起，由软件将其清零（方式 1：读 USART_DT；方式 2：操作该位写'0'）。 0: 未收到； 1: 收到。
位 4	IDLEF	0x0	ro	总线空闲 (Idle flag) 当检测到总线空闲时，该位被硬件置起，由软件将其清零（先读 USART_STS，再读 USART_DT）。 0: 无； 1: 有。
位 3	ROERR	0x0	ro	接收器溢出错误 (Receiver overflow error) 当 RDBF 仍然置起没有清除的时候，如果此时又收到数据，该位被硬件置起，由软件将其清零（先读 USART_STS，再读 USART_DT）。 0: 无； 1: 有。 注意：该位被置位时，DT 寄存器中的数据不会丢失，但是后续的数据会被覆盖。
位 2	NERR	0x0	ro	杂讯错误 (Noise error) 接收到的数据有杂讯时，该位被硬件置起，由软件将其清零（先读 USART_STS，再读 USART_DT）。 0: 无； 1: 有。
位 1	FERR	0x0	ro	帧错误 (Framing error) 当检测到停止位异常（检测到低电平）、过多的杂讯噪声或者检测到间隔帧，该位被硬件置起，由软件将其清零（先读 USART_STS，再读 USART_DT）。 0: 无； 1: 有。
位 0	PERR	0x0	ro	校验错误 (Parity error) 接收如果出现奇偶校验错误，该位被硬件置起，由软件将其清零（先读 USART_STS，再读 USART_DT）。 0: 无； 1: 有。

12.12.2 数据寄存器 (USART_DT)

域	简称	复位值	类型	功能
位 31: 9	保留位	0x000000	resd	硬件强制为 0。 数据值 (Data value)
位 8: 0	DT	0x000	rw	该寄存器包含读和写的功能。当奇偶校验位使能，发送操作时，写到 MSB 的值会被校验位取代。接收操作时，读到的 MSB 位是接收到的校验位。

12.12.3 波特比率寄存器 (USART_BAUDR)

注意：如果 TEN 或 REN 均被禁止，波特计数器停止计数。

域	简称	复位值	类型	功能
位 31: 16	保留位	0x0000	resd	硬件强制为 0。
位 15: 0	DIV	0x0000	rw	分频系数 (Division) 这 16 位定义了 USART 分频系数。

12.12.4 控制寄存器1 (USART_CTRL1)

域	简称	复位值	类型	功能
位 31: 29	保留位	0x000000	resd	硬件强制为 0。
位 28	DBN1	0x0	rw	数据位个数高位 (Data bit num) 该位和 DBN0 位一起定义了数据位的个数。 10: 7 位; 00: 8 位; 01: 9 位; 11: 禁止写入，否则数据异常。
位 27	RTODEN	0	rw	接收器超时检测使能位 (receiver time out detection enable) 0: 关闭; 1: 开启。
位 26	RETODIE	0	rw	接收器超时检测中断使能位 (receiver time out detection interrupt enable) 0: 关闭; 1: 开启。
位 25: 21	TSDT	0x00	rw	发送器开始延迟时间 (transmit start delay time) RS485 模式下一系列连续发送的第一笔数据会在数据写入后延迟一段时间后发送，以确保外部收发器已将传输方向切换为发送，该时间由 TSDT 的值决定，时间单位为 1/16 个波特率周期。
位 20: 16	TCDT	0x00	rw	发送器完成延迟时间 (transmit complete delay time) RS485 模式下一系列连续发送的最后一笔数据会在数据最后一个停止位发送完成后延迟一段时间结束，以确保外部收发器转接器已将传输方向切换为接收，该时间由 TCDT 的值决定，时间单位为 1/16 个波特率周期。
位 15	保留位	0	resd	保持默认值。
位 14	CMDIE	0	rw	字节匹配检测中断使能位 (character match detection interrupt enable) 0: 关闭; 1: 开启。
位 13	UEN	0x0	rw	USART 使能 (USART enable) 0: 关闭; 1: 开启。
位 12	DBN0	0x0	rw	数据位个数低位 (Data bit num) 该位和 DBN1 位一起定义了数据位的个数。 10: 7 位; 00: 8 位; 01: 9 位。 11: 禁止写入，否则数据异常
位 11	WUM	0x0	rw	唤醒方式 (Wake up mode) 该位定义静默状态下被唤醒的方式。

				0: 空闲帧唤醒; 1: ID 匹配唤醒。
位 10	PEN	0x0	rw	奇偶校验使能 (Parity enable) 该位定义使能硬件奇偶校验 (对于发送来说就是校验位的产生; 对于接收来说就是校验位的检测)。当使能了该位, 硬件将发送数据的最高位替换成校验位; 对接收到的数据检查其校验位是否正确。 0: 关闭; 1: 开启。
位 9	PSEL	0x0	rw	奇偶校验选择 (Parity selection) 该位定义是采用奇校验还是偶校验。 0: 偶校验; 1: 奇校验。
位 8	PERRIEN	0x0	rw	PERR 中断使能 (PERR interrupt enable) 0: 关闭; 1: 开启。
位 7	TDBEIEN	0x0	rw	发送数据缓冲器空中断使能 (TDBE interrupt enable) 0: 关闭; 1: 开启。
位 6	TDCIEN	0x0	rw	发送数据完成中断使能 (TDC interrupt enable) 0: 关闭; 1: 开启。
位 5	RDBFIEN	0x0	rw	接收数据缓冲器满中断使能 (RDBF interrupt enable) 0: 关闭; 1: 开启。
位 4	IDLEIEN	0x0	rw	总线空闲中断使能 (IDLE interrupt enable) 0: 关闭; 1: 开启。
位 3	TEN	0x0	rw	发送使能 (Transmitter enable) 该位定义发送端的使能。 0: 关闭; 1: 开启。
位 2	REN	0x0	rw	接收使能 (Receiver enable) 该位定义接收端的使能。 0: 关闭; 1: 开启。
位 1	RM	0x0	rw	接收静默 (Receiver mute) 该位定义接收端静默的开启, 可由软件置起或清零。当配置为空闲帧唤醒时, 唤醒后硬件也会将其清零, 当配置为匹配地址唤醒时, 收到匹配地址唤醒后硬件会将其清零, 收到不匹配地址后硬件会再次将其置起进入静默状态。 0: 普通; 1: 静默。
位 0	SBF	0x0	rw	发送间隔帧 (Send break frame) 使用该位来发送间隔帧。该位可以由软件置起或清零。常规用法是软件置起该位, 间隔帧发送完成后, 由硬件将该位清零。 0: 无; 1: 发送。

12.12.5 控制寄存器2 (USART_CTRL2)

域	简称	复位值	类型	功能
位 31: 28	IDH	0x0	rw	USART 的 ID 号高 4 位 (USART identification) 可配置的 USART 的 ID 号。
位 27: 20	保留位	0x000	resd	保持默认值。
位 19	MTF	0	rw	MSB 先传输 (MSB transmit first) 该位用于选择数据先传输 MSB 还是 LSB。 0: LSB; 1: MSB。 注意: 使能 MTF 时, 不支持奇偶校验。

位 18	DTREV	0	rw	DT 寄存器极性反向 (DT register polarity reverse) 0: 0: 1=H, 0=L 1: 1=L, 0=H
位 17	TXREV	0	rw	TX 引脚极性反向 (TX polarity reverse) 0: VDD=1/idle,Gnd=0/mark 1: VDD=0/mark,Gnd=1/idle
位 16	RXREV	0	rw	RX 引脚极性反向 (RX polarity reverse) 0: VDD=1/idle,Gnd=0/mark 1: VDD=0/mark,Gnd=1/idle
位 15	TRPSWAP	0x0	rw	收发管脚交换 (Transmit receive pin swap) 0: 关闭; 1: 开启。
位 14	LINEN	0x0	rw	LIN 模式使能 (LIN mode enable) 0: 关闭; 1: 开启。
位 13: 12	STOPBN	0x0	rw	停止位个数 (STOP bit num) 这 2 位用来设置停止位的个数 00: 1 位; 01: 0.5 位; 10: 2 位; 11: 1.5 位;
位 11	CLKEN	0x0	rw	时钟使能 (Clock enable) 该位用来使能同步模式或智能卡模式的时钟引脚。 0: 关闭; 1: 开启。
位 10	CLKPOL	0x0	rw	时钟极性 (Clock polarity) 在同步模式或智能卡模式下, 可以用该位选择时钟引脚上总线空闲时时钟输出的极性。 0: 低电平; 1: 高电平。
位 9	CLKPHA	0x0	rw	时钟相位 (Clock phase) 在同步模式或智能卡模式下, 可以用该位选择时钟引脚上时钟输出的相位。 0: 第一个边沿进行数据捕获; 1: 第二个边沿进行数据捕获。
位 8	LBCP	0x0	rw	最后一位时钟脉冲 (Last bit clock puLEXT) 在同步模式下, 使用该位来控制是否在时钟引脚上输出数据的最后一位对应的时钟脉冲 0: 不输出; 1: 输出。
位 7	保留位	0x0	resd	保持默认值。
位 6	BFIEN	0x0	rw	间隔帧中断使能 (break frame interrupt enable) 0: 关闭; 1: 开启。
位 5	BFBN	0x0	rw	间隔帧位数 (break frame bit num) 该位用来选择是 11 位还是 10 位的间隔帧。 0: 10 位; 1: 11 位。
位 4	IDBN	0	rw	ID 号位数(Identification bit num) 此位用于选择 ID 号位数。 0: 4 位; 1: 数据位-1 位。 注意: 当该位置'1'时, 在 7、8 或 9 位数据模式下, ID 号位数分别为低 6、7 或者 8 位, 如有配置奇偶校验则为 5、6 或者 7 位。
位 3: 0	IDL	0x0	rw	USART 的 ID 号 (USART identification) 可配置的 USART 的 ID 号。

注意: 在使能发送后不能改写这三个位 (CLKPOL、CLKPHA、LBCP)。

12.12.6 控制寄存器3 (USART_CTRL3)

域	简称	复位值	类型	功能
位 31: 16 位 13: 11	保留位	0x000000	resd	硬件强制为 0。
位 15	DEP	0	rw	DE 信号极性选择 (DE polarity selection) 0: 高电平有效。 1: 低电平有效。
位 14	RS485EN	0	rw	RS485 使能 (RS485 enable) 此位用于使能 RS485 模式, RS485 模式下 USART 通过控制信号 DE 控制外部收发器传输方向。 0: 禁止 RS485 模式, 控制信号 DE 禁止输出, RTS 引脚作 RS232 模式使用。 1: 使能 RS485 模式, 控制信号 DE 在 RTS 引脚上输出。
位 10	CTSCFIEN	0x0	rw	CTSCF 中断使能 (CTSCF interrupt enable) 0: 关闭; 1: 开启。
位 9	CTSEN	0x0	rw	CTS 使能 (CTS enable) 0: 关闭; 1: 开启。
位 8	RTSEN	0x0	rw	RTS 使能 (RTS enable) 0: 关闭; 1: 开启。
位 7	DMATEN	0x0	rw	DMA 发送使能 (DMA transmit enable) 0: 关闭; 1: 开启。
位 6	DMAREN	0x0	rw	DMA 接收使能 (DMA receiver enable) 0: 关闭; 1: 开启。
位 5	SCMEN	0x0	rw	智能卡模式使能 (Smart card mode enable) 0: 关闭; 1: 开启。
位 4	SCNACKEN	0x0	rw	智能卡 NACK 使能 (Smart card NACK enable) 该位用于配置校验错误出现时, 发送 NACK。 0: 不发送; 1: 发送。
位 3	SLBEN	0x0	rw	单线双向半双工模式使能 (Single line bidirectional half-duplex enable) 0: 关闭; 1: 开启。
位 2	IRDALP	0x0	rw	红外低功耗模式配置 (IrDA low-power mode) 该位用来配置红外低功耗模式。 0: 关闭; 1: 开启。
位 1	IRDAEN	0x0	rw	红外功能使能 (IrDA enable) 0: 关闭; 1: 开启。
位 0	ERRIEN	0x0	rw	错误中断使能 (Error interrupt enable) 当有帧错误、接收溢出错误或者杂讯错误时产生中断。 0: 关闭; 1: 开启。

12.12.7 保护时间和预分频寄存器 (GDIV)

域	简称	复位值	类型	功能
位 31: 16	保留位	0x0000	resd	硬件强制为 0。
位 15: 8	SCGT	0x00	rw	智能卡保护时间值 (Smart card guard time) 在智能卡模式下, 当保护时间过去后, 才会设置发送完成标志, 这几位配置保护时间值。
位 7: 0	ISDIV	0x00	rw	红外或者智能卡分频系数 (IrDA/smaERTCard division) 红外 (IrDA) 模式: 8 位[7: 0]有效, 普通模式无效且只能设置为 00000001, 低功耗模式分频系数对外设时钟进行分频, 作为脉冲宽度的基数周期; 00000000: 保留 - 不要写入该值; 00000001: 1 分频; 00000010: 2 分频; 智能卡模式: 低 5 位[4: 0]有效, 分频系数对外设时钟进行分频, 给智能卡提供时钟。可以设置为如下值: 00000: 保留 - 不要写入该值; 00001: 2 分频; 00010: 4 分频; 00011: 6 分频;

12.12.8 接收器超时检测值寄存器 (RTOV)

域	简称	复位值	类型	功能
位 31: 24	保留位	0x00	resd	硬件强制为 0。
位 23: 0	RTOV	0x00	rw	接收器超时检测值 (receiver time out value) 单位为 1bit 位宽

12.12.9 中断标志位清除寄存器 (IFC)

域	简称	复位值	类型	功能
位 31: 18				
位 16: 12	保留位	0x00	resd	硬件强制为 0。
位 10: 0				
位 17	CMDFC	0	w1	字节匹配检测标志清除位 (character match detection flag clear)
位 11	RTODFC	0	w1	接收器超时检测标志清除位 (receiver time out detection flag clear)

13 串行外设接口 (SPI)

13.1 串行外设接口 (SPI) 简介

SPI 接口提供软件编程配置选项，根据软件编程配置方式不同，可以分别作为 SPI 和 I²S 使用。本章将分 SPI 和 I²S 分别介绍 SPI 作 SPI 或 I²S 的功能特性以及配置流程。

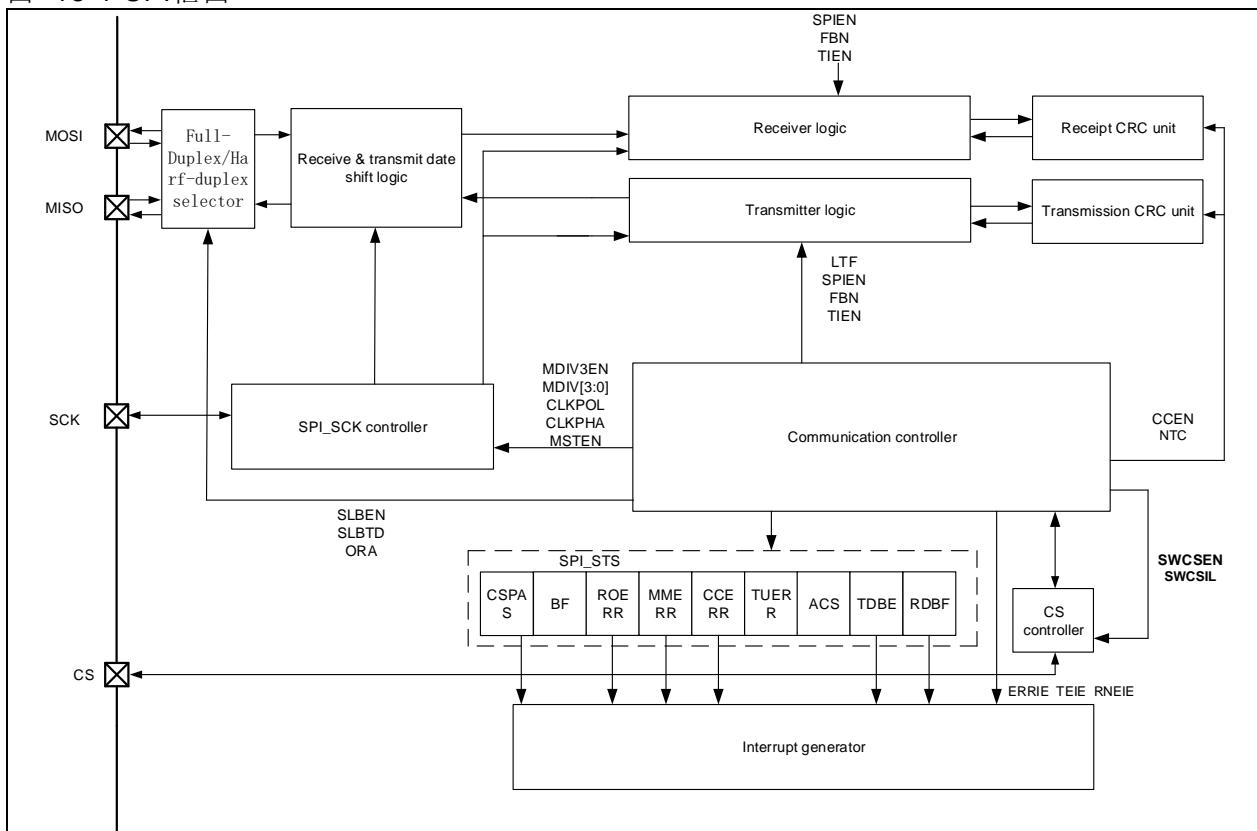
13.2 SPI 功能描述

13.2.1 SPI 简述

串行外设接口 (SPI) 根据软件编程配置的方式不同，可以分别作为主机和从机使用，又可以分别工作在全双工、全双工只收、半双工只发/只收四种不同的模式下，并且还提供 DMA 传输，SPI 内部硬件自动 CRC 计算和校验等功能，同时可以通过软件编程配置使 SPI 接口兼容 TI 协议。

SPI 的架构框图见下图：

图 13-1 SPI 框图



SPI 接口作为 SPI 使用时主要特征如下：

- 可编程配置的全双工或半双工通信
 - 全双工同步通信（可以选择全双工只收以此释放用于发送的 IO）
 - 半双工同步通信（可以根据软件编程配置选择传输方向：发送或接收）
- 可编程配置主/从模式
- 可编程配置的 CS 信号处理方式
 - 硬件处理 CS
 - 软件处理 CS
- 可编程配置的 8 位或 16 位帧位数
- 可编程配置的通信频率以及分频系数（最大分频系数为 $f_{PCLK}/2$ ）
- 可编程配置的时钟极性和相位
- 可编程配置的数据传输顺序(先发 MSB/LSB)
- 可编程配置的错误中断标志（CS 脉冲异常，接收器溢出错误，主模式错误，CRC 校验错误）

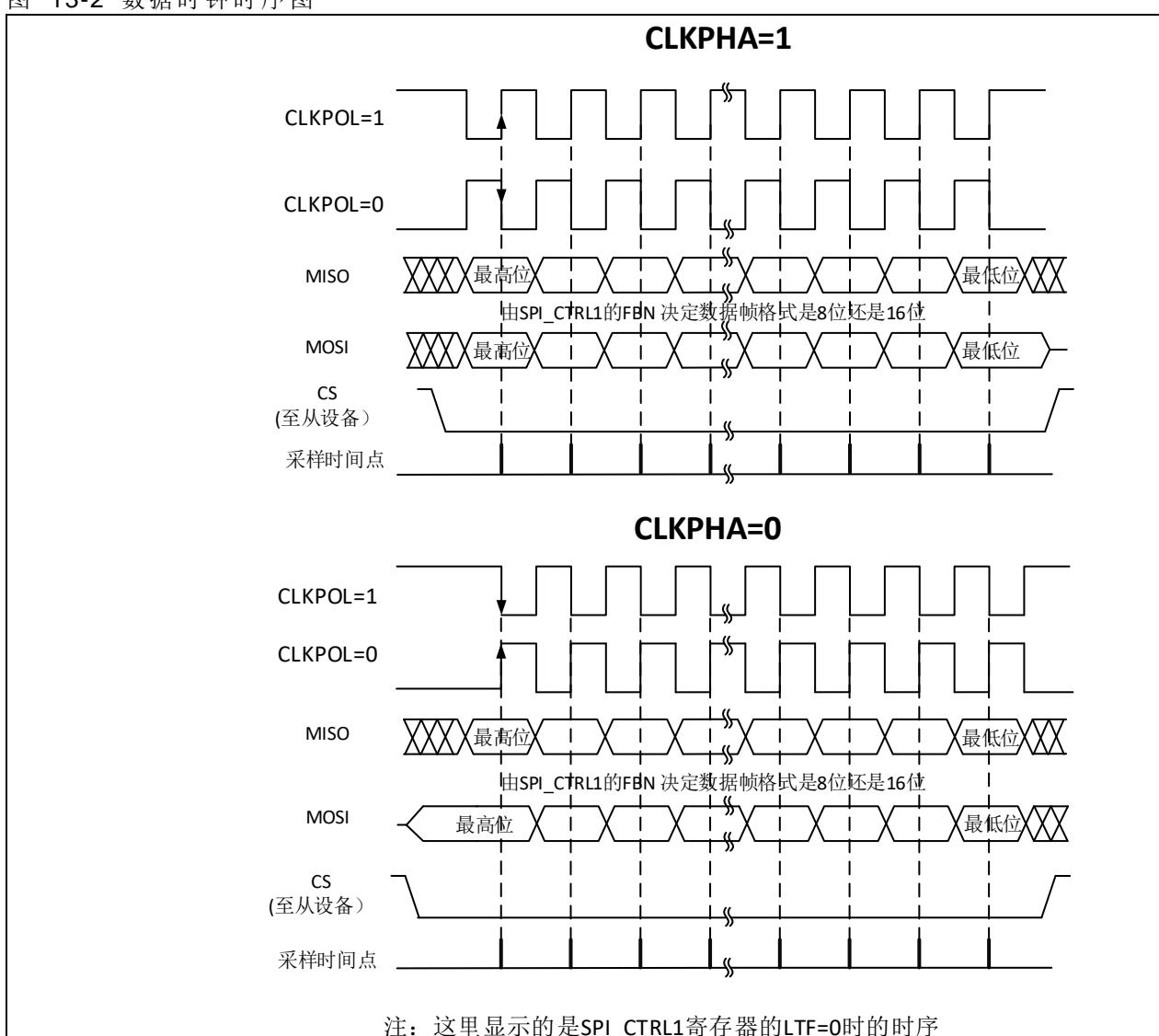
- 可编程配置的发送数据缓冲器空中断以及接收数据缓冲器满中断
- 支持 DMA 发送和接收
- 支持硬件 CRC 发送和校验
- 具备通信忙标志
- 兼容 TI 协议

时钟信号的相位和极性

SPI_CTRL1 寄存器的 CLKPOL 和 CLKPHA 位，能够组合四种可能的时序关系。CLKPOL（时钟极性）位控制在没有数据传输时时钟的空闲状态电平，此位对主模式和从模式下的设备都有效。如果 CLKPOL 被清‘0’，SCK 引脚在空闲状态保持低电平；如果 CLKPOL 被置‘1’，SCK 引脚在空闲状态保持高电平。

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

图 13-2 数据时钟时序图



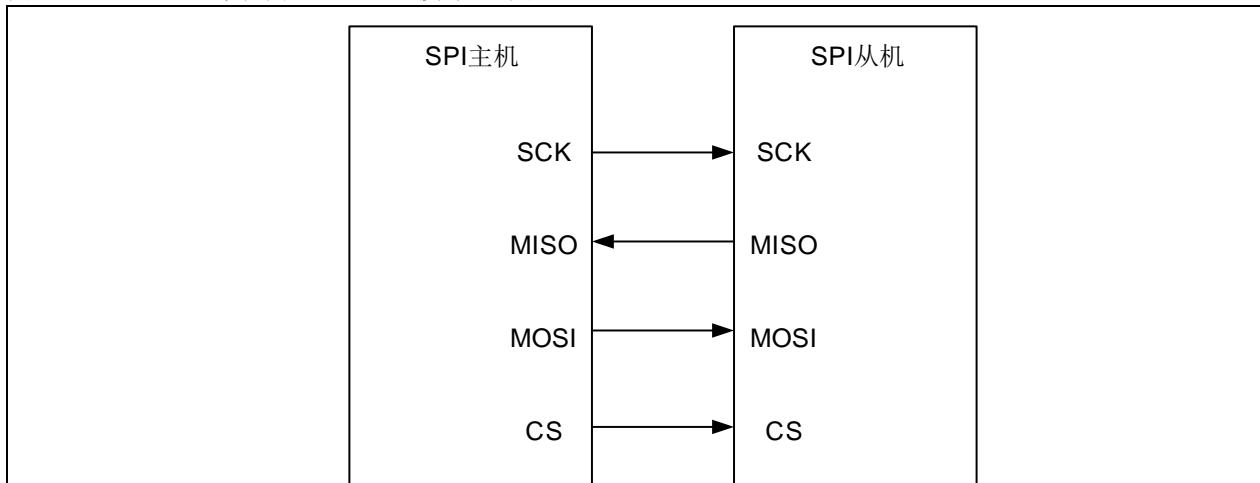
13.2.2 全双工半双工选择器简述和配置流程

SPI 全双工半双工选择器通过软件编程配置的方式，可以使 SPI 接口作为 SPI 使用时，可以工作在双线单向全双工，单线单向只收，单线双向半双工发送和单线双向半双工接收四种同步模式。

双线单向全双工模式配置方式以及 SPI IO 连接方式如下：

SLBEN 位置 0, ORA 置 0 时, SPI 工作在双线单向全双工, 此时 SPI 可以同时进行数据的收发, IO 连接方式如下图。

图 13-3 SPI 双线单向全双工连接示意图



SPI 作主机或从机在此模式下, 关闭 SPI 或进入省电模式 (或关闭 SPI 系统时钟) 之前需要等待 RDBF 置位, TDBE 置位, 并等待 BF=0。

单线双向只收模式配置方式以及 SPI IO 连接方式如下：

SLBEN 位置 0, ORA 置 1 时, SPI 工作在单线双向只收模式, 此时 SPI 只能作为数据接收方, 无法发送数据。作为主机时使用 MISO 接收数据, MOSI 管脚所映射的 IO 释放。作为从机时使用 MOSI 接收数据, MISO 管脚所映射的 IO 释放。

图 13-4 SPI 作主机单线双向只收连接示意图

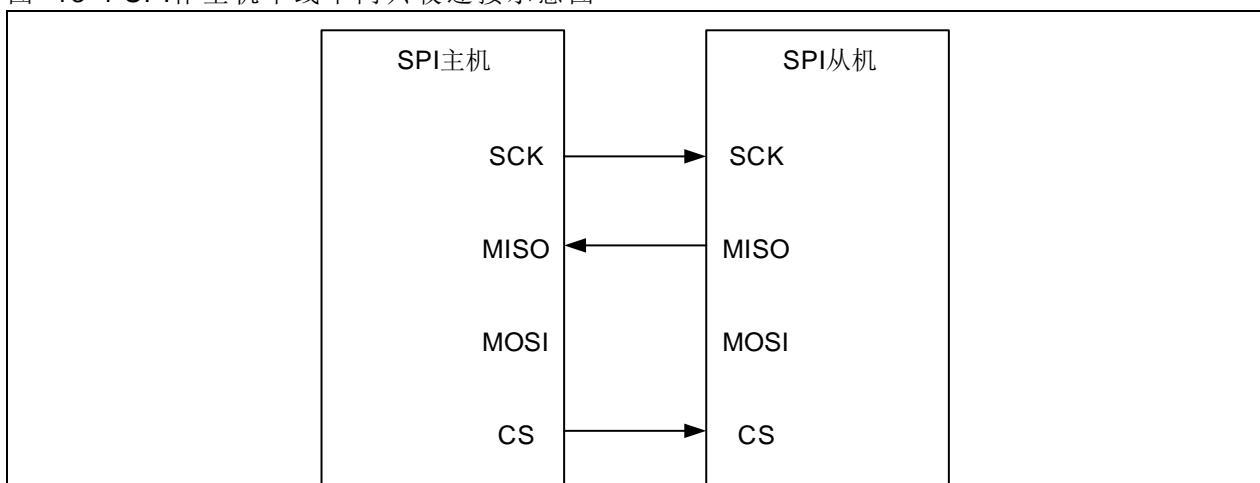
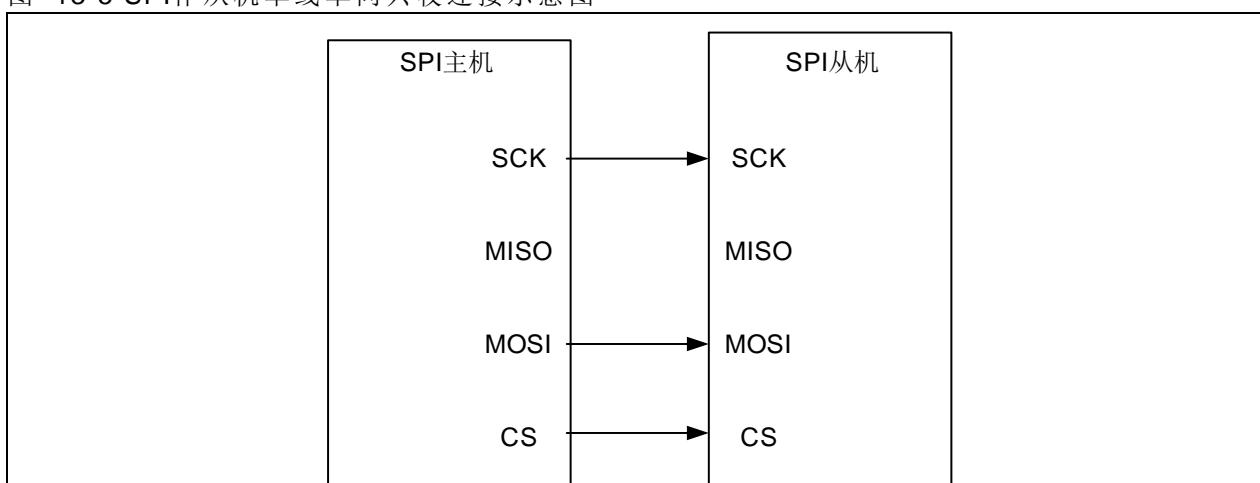


图 13-5 SPI 作从机单线双向只收连接示意图



SPI 作主机时，在此模式下，需要等待倒数第二个 RDBF 置起，关闭 SPI 之前等待一个 SPI_SCK 周期，在进入省电模式(或关闭 SPI 系统时钟)之前等待最后一个 RDBF=1。

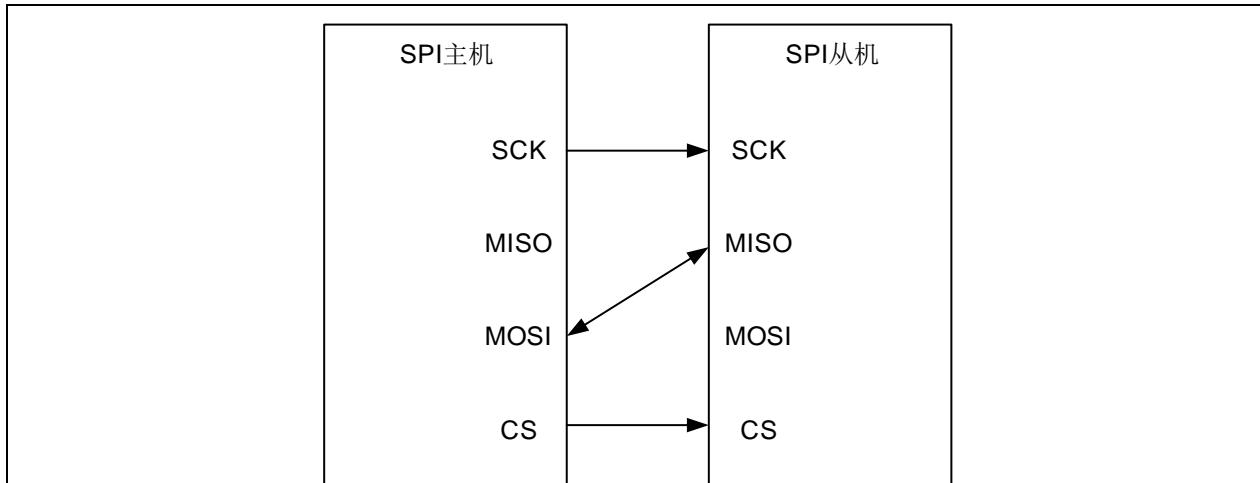
SPI 作从机时，在此模式下，关闭 SPI 无需判断任何标志，但是在进入省电模式(或关闭 SPI 系统时钟)之前需要等待 BF=0。

单线双向半双工模式配置方式以及 SPI IO 连接方式如下：

SLBEN 位置 1 时，SPI 工作在单线双向半双工模式，此时 SPI 可以分时进行数据收发。作为主机时使用 MOSI 收发数据，MISO 管脚所映射的 IO 释放。作为从机时使用 MISO 收发数据，MOSI 管脚所映射的 IO 释放。

软件通过编程控制 SLBD 位控制传输方向，SLBD 位置 1 时，SPI 只能发送数据，SLBD 位置 0 时，SPI 只能接收数据。

图 13-6 SPI 作单线双向半双工连接示意图



SPI 作主机或从机时，在单线双向半双工，传输方向选择为发送时，需要等待 TDBE 置位，BF=0 后才能关闭 SPI，在关闭 SPI 后才可以进入省电模式(或关闭 SPI 系统时钟)。

SPI 作主机时，在单线双向半双工，传输方向选择为接收时，需要等待倒数第二个 RDBF 置起，关闭 SPI 之前等待一个 SPI_SCK 周期，在进入省电模式(或关闭 SPI 系统时钟)之前等待最后一个 RDBF=1。

SPI 作从机时，在单线双向半双工，传输方向选择为接收时，关闭 SPI 无需判断任何标志，但是在进入省电模式(或关闭 SPI 系统时钟)之前需要等待 BF=0。

13.2.3 CS 控制器简述和配置流程

SPI 的 CS 控制器提供通过软件可编程配置的方式选择硬件控制 CS 信号或软件控制 CS 信号，以此实现 CS 信号的控制，用于多处理器模式下主机从机选择，以及通过 CS 信号后于 SCK 信号使能，有效地屏蔽总线上的干扰，下面将分软件 CS 以及硬件 CS 来介绍 CS 控制器的配置流程，并会简述在主机和从机模式下软件和硬件 CS 的输入输出方式。

硬件 CS 配置流程：

当 SPI 作主机，硬件 CS 输出时，HWCSOE 位置 1，SWCSEN 置 0，开启硬件 CS 控制，SPI 在使能之后会在 CS 管脚上输出低电平，在 SPI 关闭并且发送完成后，释放 CS 信号。

当 SPI 作主机，硬件 CS 输入时，HWCSOE 位置 0，SWCSEN 置 0，开启硬件 CS 控制，此时一旦主机 SPI 检测到 CS 管脚为低电平时，SPI 硬件自动关闭 SPI 并进入从机模式，模式错误标志 MMERR 同时置位，若使能了错误中断 (ERRIE=1)，将会产生中断，在 MMERR 置位期间，硬件不允许软件置位 SPIEN 和 MSTEN 位，通过读或写 SPI_STS 再写 SPI_CTRL1 可以清除 MMERR。

当 SPI 作从机，硬件 CS 输入时，HWCSOE 位置 0，SWCSEN 置 0，开启硬件 CS 控制，从机根据 CS 管脚上的电平判断是否发送或接收数据，只有 CS 管脚上为低电平时，从机才会被选中并进行数据的收发。

软件 CS 配置流程：

当 SPI 作主机，软件 CS 输入时，SWCSEN 位置 1，开启软件 CS 控制，当 SWCSIL 位置 0 时，SPI 硬件自动关闭 SPI 并进入从机模式，模式错误标志 MMERR 同时置位，若使能了错误中断 (ERRIE=1)，将会产生中断，在 MMERR 置位期间，硬件不允许软件置位 SPIEN 和 MSTEN 位，通过读或写 SPI_STS 再写 SPI_CTRL1 可以清除 MMERR。

当 SPI 作从机，软件 CS 输入时，SWCSEN 位置 1，开启软件 CS 控制，SPI 根据 SWCSIL 位判断 CS 信号电平，不使用 CS 管脚，当 SWCSIL=0 时，从机才会被选中并进行数据的收发。

13.2.4 SPI_SCK控制器简述和配置流程

SPI 协议采用同步传输，所以 SPI 接口在作为 SPI 使用时，作主机时，需要产生通信时钟用于 SPI 接口的数据收发，并且需要将该通信时钟通过 IO 输出给从机，用于从机的数据收发；作从机时，需要外设提供通信时钟从 IO 输入到 SPI 接口内部作为通信时钟使用，所以实际上，SPI_SCK 控制器便是扮演着产生 SPI_SCK 以及分配 SPI_SCK 的角色，详细的配置方法如下所述。

SPI_SCK 控制器配置流程：

- 时钟极性相位选择：配置 CLKPOL,CLKPHA 选择需要的极性和相位。
- 时钟分频系数选择：配置 CRM 选择需要的 PCLK 频率，配置 MDIV[3: 0] 或 MDIV3EN 选择需要的分频系数。
- 主机或从机选择：配置 MSTEN 选择 SPI 作主机或从机使用，注意主机只收模式在 SPI 使能后就会开始输出时钟，直到 SPI 被关闭且接收完成。

13.2.5 CRC简述和配置流程

SPI 接口内部具有独立的发送和接收 CRC 计算单元，通过软件编程配置，SPI 接口在作为 SPI 使用时，可以同时在用户使用 DMA 读写数据或 CPU 读写数据的情况下，自动进行 CRC 计算以及 CRC 校验，如果在传输过程中，硬件检测到接收到的数据与 SPI_RCRC 中的数据不符，且该笔数据又是 CRC 数据时，CCERR 位会置起，若使能了错误中断（ERRIE=1），将会产生中断。

下面分 DMA 和 CPU 操作数据寄存器分别描述 SPI 的 CRC 功能以及 CRC 配置流程。

CRC 配置流程

- CRC 计算多项式配置：配置 SPI_CPOLY 选择 CRC 计算多项式。
- 使能 CRC：置起 CCEN 位使能 CRC 计算，该操作将会复位 SPI_RCRC 以及 SPI_TCRC。
- 根据 DMA 或 CPU 操作数据寄存器选择是否以及何时置位 NTC 位，具体请参见下方描述。

DMA 发送模式：

在采用 DMA 写入待发送的数据时，当使能 CCEN 后，硬件会根据 SPI_CPOLY 中的值以及每笔发送的数据自动计算 CRC 值，并在最后一笔数据发送完成后自动发送 CRC 值，该值即 SPI_TCRC 中的值。

DMA 接收模式：

在采用 DMA 读取待接收的数据时，当使能 CCEN 后，硬件会根据 SPI_CPOLY 中的值以及每笔接收的数据自动计算 CRC 值，并在最后一笔数据接收完成后等待 CRC 数据接收完成，并将收到的 CRC 值和 SPI_RCRC 中的值作比较，若校验出错，会置起 CCERR 标志，若使能了 ERRIE 位，则产生错误中断。

CPU 发送模式：

相较于 DMA 发送模式，该模式需要软件在写入最后一笔待发送的数据后，在最后一笔数据发送完成之前置起 NTC 位。

CPU 接收模式

在双线单向全双工模式下，按照 CPU 发送模式操作 NTC 位，CPU 接收模式的 CRC 计算和校验会自动完成，在单线单向只接收以及单线双向只接收模式下，相较于 DMA 接收需要软件在接收到倒数第二笔数据之后，接收到最后一笔数据之前置起 NTC 位。

13.2.6 DMA传输简述和配置流程

SPI 接口支持使用 DMA 进行发送数据的写入，接收数据的读取，具体配置流程分别见下述的 DMA 发送配置流程以及 DMA 接收配置流程。

需要特别注意的是，在开启 CRC 计算和校验时，DMA 发送数据的个数配置为待发送的数据个数，DMA 读取数据的个数配置为待接收的数据个数，此时硬件在所有数据传输完毕后自动进行 CRC 传输，且接收方还会自动进行 CRC 校验，需要注意，接收到的 CRC 数据，硬件会搬到 SPI_DT 寄存器中，并置位 RDBF，以及在开启了 DMA 传输时发出 DMA 读请求，所以这里推荐当 CRC 接收完毕后软件要去读 DT 寄存器来取走 CRC 值，防止后续传输出错。

DMA 发送配置流程：

- 选择 DMA 传输通道：在 DMA 章节 DMA 通道映射表中选择用于当前所用 SPI 的 DMA 通道。
- 配置 DMA 传输目标地址：在 DMA 控制寄存器中 DMA 传输目的地址位写入当前所使用的 SPI 的 SPI_DT 寄存器地址，DMA 将会在接收到发送请求后将待发送的数据写入该地址。

- 配置 DMA 传输源地址：在 DMA 控制寄存器中 DMA 传输源地址位写入待发送数据存放的地址，DMA 将会在接收到发送请求后将该地址内的数据写入到目标地址中，即写入到当前所使用的 SPI 的 SPI_DT 寄存器中。
- 配置 DMA 传输数据个数：在 DMA 控制寄存器相关位置配置期望传输的数据个数。
- 配置 DMA 传输通道优先级：在 DMA 控制寄存器相关位置配置当前所使用通道的 SPI 的 DMA 传输通道优先级。
- 配置 DMA 中断产生时机：在 DMA 控制寄存器相关位置配置是在传输完成或传输完成一半时产生 DMA 中断。
- 使能 DMA 传输通道：在 DMA 控制寄存器相关位置使能当前所选用的 DMA 通道。

DMA 接收配置流程：

- 选择 DMA 传输通道：在 DMA 章节 DMA 通道映射表中选择用于当前所用 SPI 的 DMA 通道。
- 配置 DMA 传输目标地址：在 DMA 控制寄存器中 DMA 传输目的地址位写入期望存放接收数据的地址，DMA 将会在接收到接收请求后，将当前所使用的 SPI 的 SPI_DT 寄存器中的数据存放在目的地址中。
- 配置 DMA 传输源地址：在 DMA 控制寄存器中 DMA 传输源地址位写入当前所使用的 SPI 的 SPI_DT 寄存器的地址，DMA 将会在接收到接收请求后将该地址内的数据写入到目标地址中，即写入到期望存放接收数据的地址。
- 配置 DMA 传输数据个数：在 DMA 控制寄存器相关位置配置期望传输的数据个数。
- 配置 DMA 传输通道优先级：在 DMA 控制寄存器相关位置配置当前所使用通道的 SPI 的 DMA 传输通道优先级。
- 配置 DMA 中断产生时机：在 DMA 控制寄存器相关位置配置是在传输完成或传输完成一半时产生 DMA 中断。
- 使能 DMA 传输通道：在 DMA 控制寄存器相关位置使能当前所选用的 DMA 通道。

13.2.7 TI 模式简述和配置流程

SPI 接口支持 TI 协议，用户可以通过将 TIEN 位置 1 使能 TI 模式。

使能 TI 模式后，SPI 接口将按照 TI 协议要求产生通信时钟 SPI_CLK，这意味着 SPI_CLK 的极性和相位会强制符合 TI 协议要求，用户无需也无法通过配置 CLKPOL 和 CLKPHA 来改变 SPI_CLK 的极性和相位。

使能 TI 模式后，SPI 接口将按照 TI 协议要求产生 CS 信号，这意味着 CS 的输出和输入将强制符合 TI 协议的要求，用户无需也无法通过配置 SWCSEN 位，SWCSIL 位，HWCSOE 位来管理 CS 信号。

使能 TI 模式后，SPI 从机只会在数据发送期间控制 MISO 引脚，这意味着 SPI 从机的 MISO 引脚将会在空闲状态保持高阻态。

使能 TI 模式后，SPI 接口作从机使用时，SPI 接口会在数据传输期间侦测错误的 CS 脉冲，并在侦测到错误的 CS 脉冲后置起 CSPAS 位(该位可以通过软件读 SPI_STS 清除)，此时 SPI 会忽略掉该错误的脉冲，但由于此时 CS 信号已经出现异常，软件应当关闭 SPI 从机，重新配置 SPI 主机，再打开 SPI 从机以重新开始通信。

13.2.8 发送器简述和配置流程

SPI 发送器时钟由 SPI_SCK 控制器提供，根据软件编程配置，发送器可以输出不同的数据帧格式，SPI 具有一个数据缓冲寄存器 SPI_DT，软件需要将待发送的数据先写入 SPI_DT，发送器在有时钟时，会把 SPI_DT 中的数据保存到发送器中的数据缓冲器(有别于 SPI_DT，SPI 发送器中的数据缓冲器由 SPI_SCK 驱动，且硬件自动控制，软件不可操作)，并按照配置好的帧格式将数据依次发出。

用户可以选择 DMA 或 CPU 来控制数据的写入，若选择 DMA 传输，详细配置请参见 DMA 传输章节，若选择 CPU 传输，则用户需要判断 TDBE 位，该位复位值为 1，代表 SPI_DT 为空，若 TDBEIE 置位，则产生中断，数据写入后，TDBE 拉低，直到数据被同步到发送器中的数据缓冲器后，TDBE 再次被拉起，即用户只可以在 TDBE 置位时写入待发送的数据。

发送器配置完成并使能 SPI 后，SPI 将进入数据发送状态，所以在此之前，应需要参考全双工半双工章节配置通信选用的是全双工或半双工等，并参考 CS 控制器章节配置选用的 CS 控制模式，还需要参考 SPI_SCK 控制章节配置通信时钟，若使用了 CRC 以及 DMA，还需参考 CRC 以及 DMA 传输章节配置 CRC 以及 DMA，如下为推荐的发送器配置流程。

发送器配置流程:

- 配置全双工半双工选择器。
- 配置CS控制器
- 配置SPI_SCK控制器
- 配置CRC（若需要使用CRC自动计算和校验功能）
- 配置DMA传输（若需要使用DMA传输功能）
- 若没有选择DMA传输功能，软件需要判断TDBE位，软件需要根据需求判断是否要打开发送数据中断，即置位TDBIE。
- 配置帧格式：配置LTF位选择MSB/LSB格式，配置FBN选择8/16位数据。
- 置位SPIEN位使能SPI

13.2.9 接收器简述和配置流程

SPI接收器时钟由 SPI_SCK 控制器提供，根据软件编程配置，接收器可以接收不同的数据帧格式，SPI接收器具有一个接收数据缓冲寄存器，该寄存器由 SPI_SCK 驱动，在每笔传输的最后一个 CLK，数据从移位寄存器压入该接收数据缓冲寄存器，随后发送器会给出数据接收完成的标志给到 SPI 的控制逻辑，SPI 的控制逻辑在检测到该标志后会自动把接收器中的数据缓冲器中的值压入 SPI_DT，RDBF 随之置起，这意味着有数据被收到，且该数据已被压入 SPI_DT，此时读 SPI_DT 可以读出该笔数据，同时 RDBF 随之清除。

用户可以选择 DMA 或 CPU 来控制数据的读出，若选择 DMA 传输，详细配置请参见 DMA 传输章节，若选择 CPU 传输，则用户需要判断 RDBF 位，该位复位值为 0，代表 SPI_DT 为空，当有数据被接收到，且数据被移入 SPI_DT 时，RDBF 置位，代表 SPI_DT 内有数据等待读取，此时若 RDBFIE 置位则产生中断。

若在下一笔接收器接收到的数据准备压入 SPI_DT 时，之前接收到的数据仍未被读走，即 RDBF 仍为 1，则代表数据溢出，在此之前接收到的数据不会丢失，但之后的数据都将丢失，此时 ROERR 置起，若 ERRIE 置位，则产生错误中断，依次读 SPI_DT 寄存器和 SPI_STS 寄存器可将 ROERR 清除，如下为推荐的接收器配置流程。

接收器配置流程:

- 配置全双工半双工选择器。
- 配置CS控制器。
- 配置SPI_SCK控制器。
- 配置CRC（若需要使用CRC自动计算和校验功能）。
- 配置DMA传输（若需要使用DMA传输功能）。
- 若没有选择DMA传输功能，软件需要判断RDBF位，软件需要根据需求判断是否要打开接收数据中断，即置位RDBFIE。
- 配置帧格式：配置LTF位选择MSB/LSB格式，配置FBN选择8/16位数据。
- 置位SPIEN位使能SPI。

13.2.10 Motorola模式通信时序

本节介绍 SPI 通信时序，包括全双工和半双工的主/从通信时序。

全双工通信-主机通信时序

其中主机端配置如下：

MSTEN=1：设备为主机；

SLBEN=0：全双工模式；

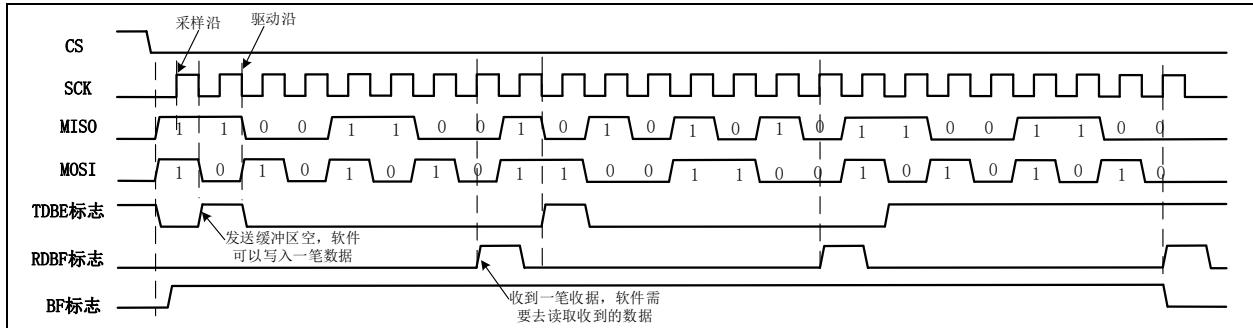
CLKPOL=0, CLKPHA=0：SCK 空闲输出低电平，第一个边沿作为采样边沿；

FBN=0：帧数据的长度为 8 位；

主机发送数据（MOSI）: 0xaa, 0xcc, 0xaa

从机发送数据（MISO）: 0xcc, 0xaa, 0xcc

图 13-7 主机全双工通信



全双工通信-从机通信时序

其中从机端配置如下：

MSTEN=0：设备为从机；

SLBEN=0：全双工模式；

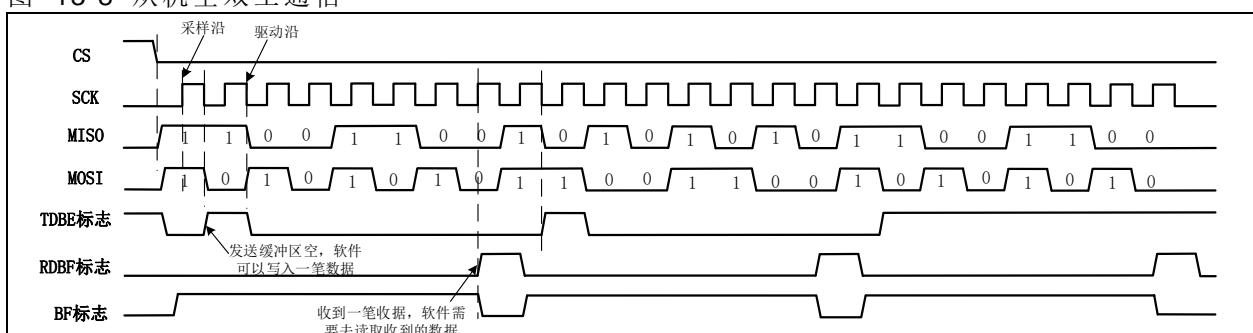
CLKPOL=0, CLKPHA=0：SCK 空闲输出低电平，第一个边沿作为采样边沿；

FBN=0：帧数据的长度为 8 位；

主机发送数据（MOSI）: 0xaa, 0xcc, 0xaa

从机发送数据（MISO）: 0xcc, 0xaa, 0xcc

图 13-8 从机全双工通信



半双工通信-主机发送时序

MSTEN=1：设备为主机；

SLBEN=1：单线双向模式；

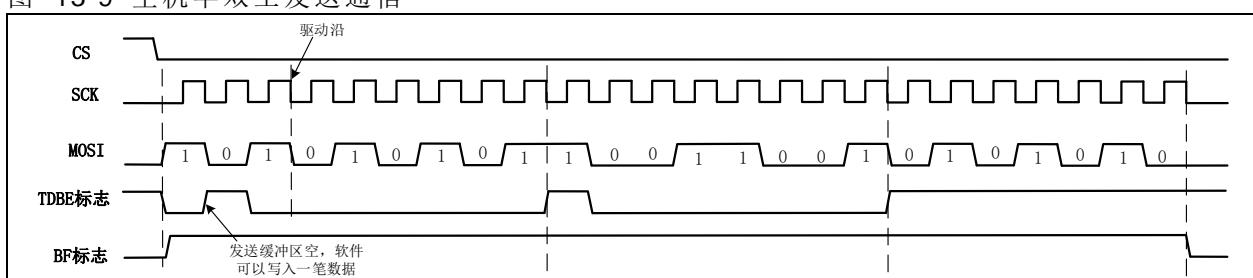
SLBD=1：发送模式；

CLKPOL=0, CLKPHA=0：SCK 空闲输出低电平，第一个边沿为采样边沿；

FBN=0：帧数据的长度为 8 位；

主机发送数据: 0xaa, 0xcc, 0xaa

图 13-9 主机半双工发送通信



半双工通信-从机接收时序

MSTEN=0: 设备为从机;

SLBEN=1: 单线双向模式;

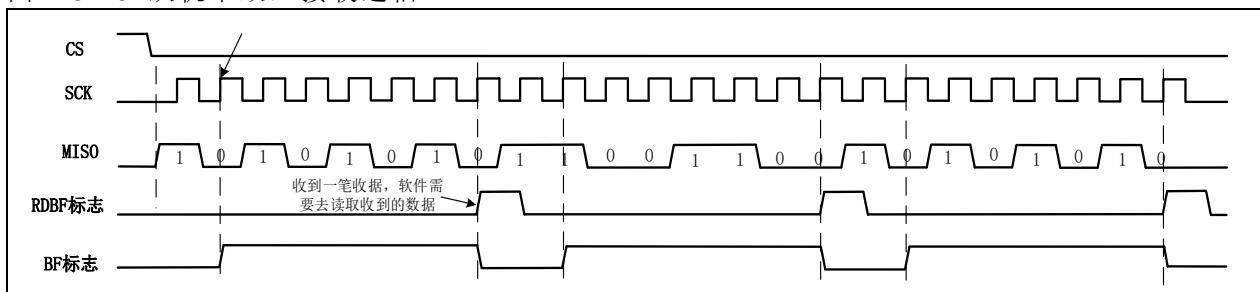
SLBTD=0: 接收模式;

CLKPOL=0, CLKPHA=0: SCK 空闲输出低电平, 第一个边沿为采样边沿;

FBN=0: 帧数据的长度为 8 位;

从机接收数据: 0xaa, 0xcc, 0xaa

图 13-10 从机半双工接收通信



半双工通信-从机发送时序

MSTEN=0: 设备为从机;

SLBEN=1: 单线双向模式;

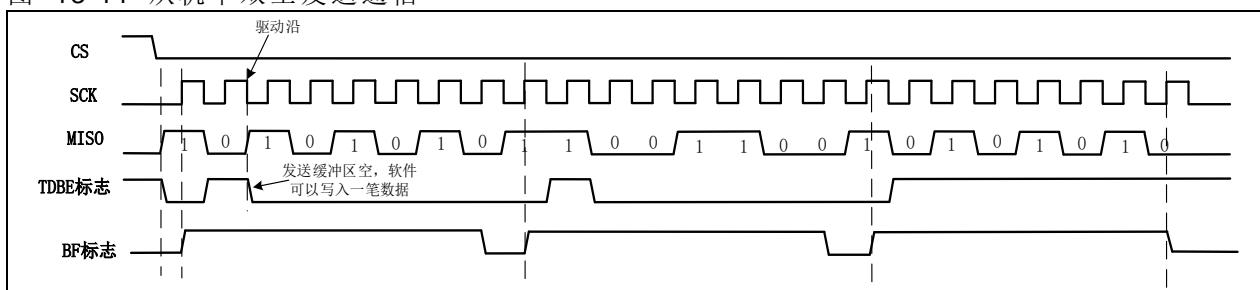
SLBTD=1: 发送模式;

CLKPOL=0, CLKPHA=0: SCK 空闲输出低电平, 第一个边沿为采样边沿;

FBN=0: 帧数据的长度为 8 位;

从机发送数据: 0xaa, 0xcc, 0xaa

图 13-11 从机半双工发送通信



半双工通信-主机接收时序

MSTEN=1: 设备为主机;

SLBEN=1: 单线双向模式;

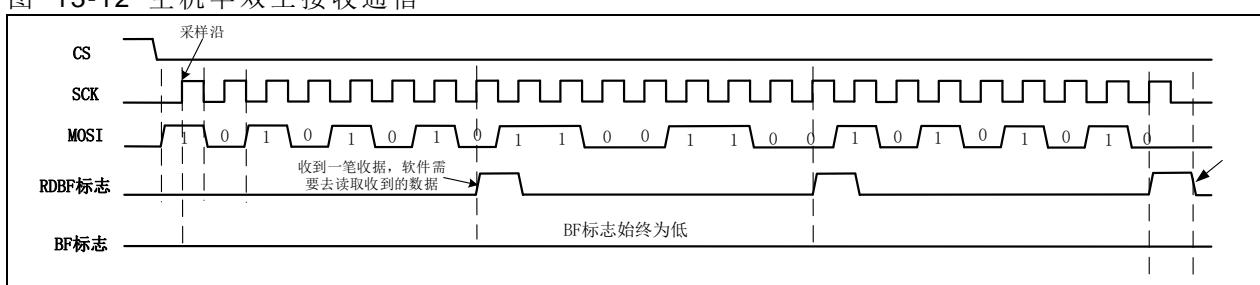
SLBTD=0: 接收模式;

CLKPOL=0, CLKPHA=0: SCK 空闲输出低电平, 第一个边沿为采样边沿;

FBN=0: 帧数据的长度为 8 位;

主机接收数据: 0xaa, 0xcc, 0xaa

图 13-12 主机半双工接收通信

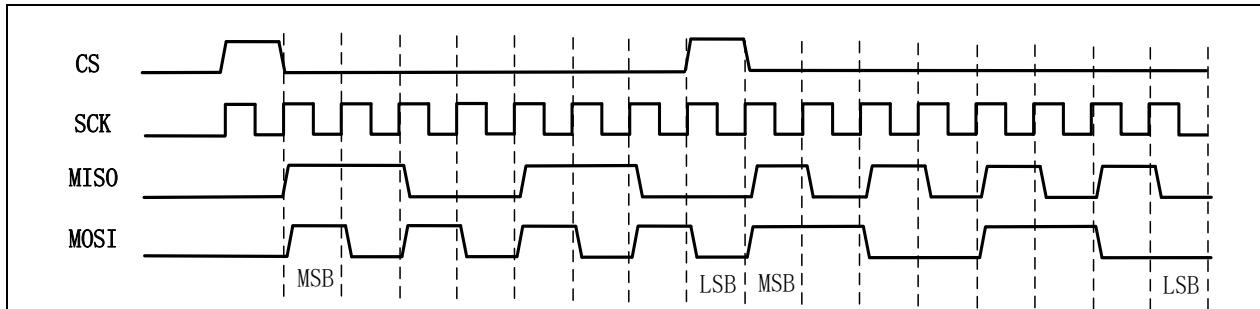


13.2.11 TI模式通信时序

SPI 接口支持 TI 模式，用户可以通过 TIEN 位置 1 使能该模式。

TI 模式下，连续与不连续通信的时序图稍有区别。当待发送数据在当前发送数据帧最后一位对应时钟 SCK 的上升沿之前写入，则通信连续，每笔数据的中间不存在 dummy CLK，主机在发送当前数据帧的最后一位数据的同时发出 CS 有效脉冲。

图 13-13 TI模式连续通信时序



TI 模式下，当待发送数据在当前发送数据帧最后一位对应时钟 SCK 的上升沿与下降沿之间写入，则每笔数据的中间存在一个 dummy CLK。

图 13-14 TI模式带dummy CLK的连续通信时序

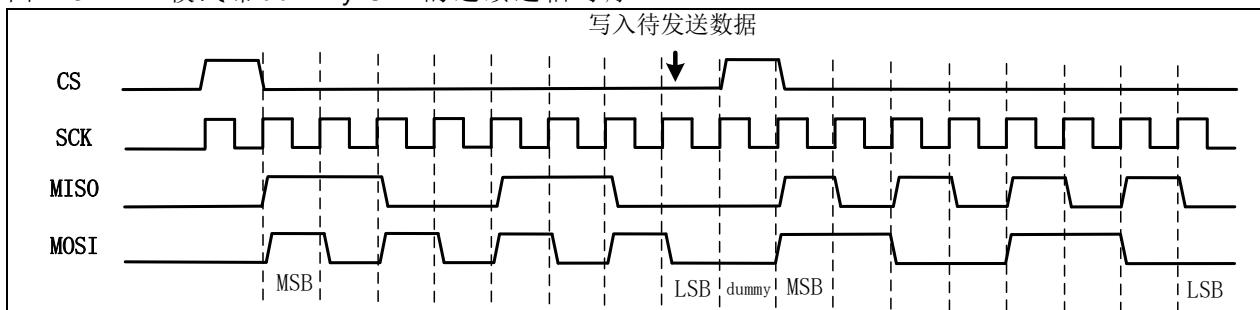
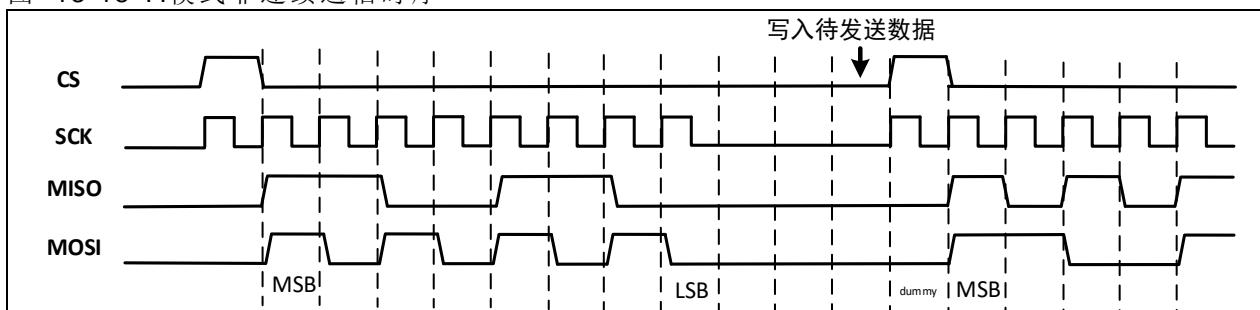
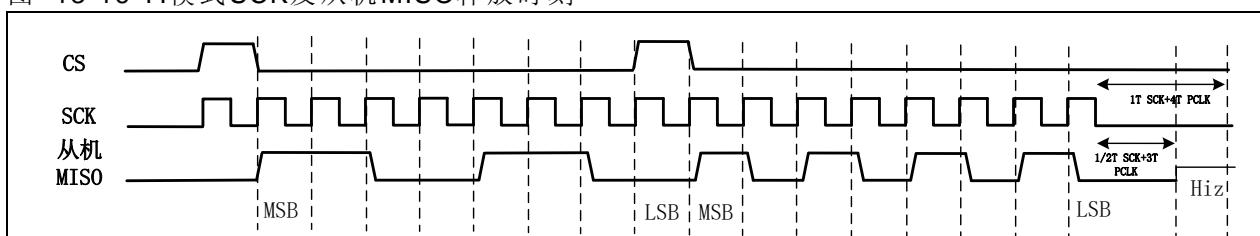


图 13-15 TI模式非连续通信时序



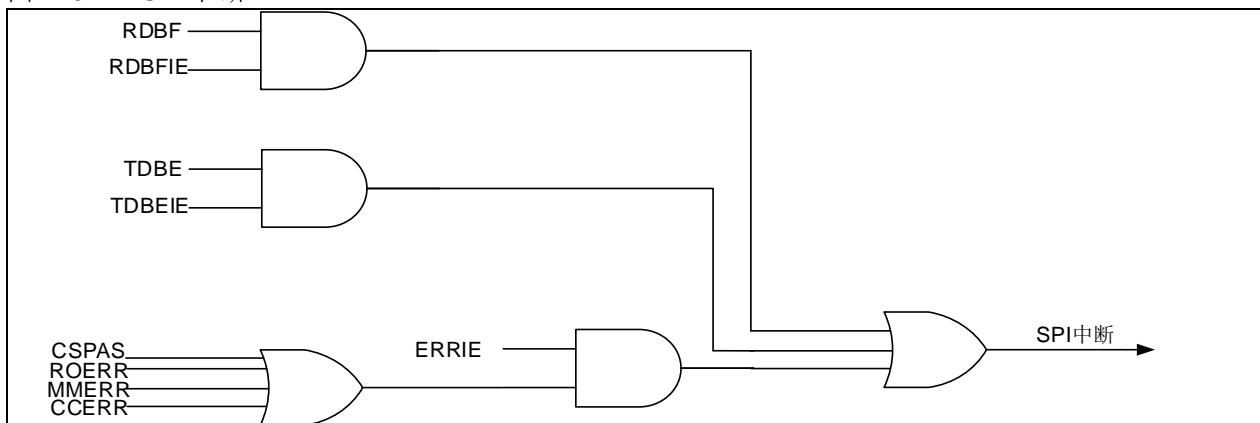
TI 模式下，当待发送数据在当前发送数据帧最后一位对应时钟 SCK 的下降沿之后写入，则主机固定在 1T SCK + 4T PCLK 后才能重新发出有效 SCK 时钟，从机在接收当前数据帧最后一位时，仍然没有检测到有效的 CS 脉冲，则在 1/2T SCK + 3T PCLK 后关闭 MISO 的输出功能，控制 MISO 浮空。

图 13-16 TI模式SCK及从机MISO释放时刻



13.2.12 中断

图 13-17 SPI中断



13.2.13 IO管脚控制

SPI 接口作为 SPI 使用时最多可有 4 根管脚与外设相连，各管脚的使用方法可以参见全双工半双工选择器简述和配置流程以及 CS 控制器简述和配置流程章节，各管脚的定义如下。

MISO: 主机输入/从机输出管脚。在 SPI 接口作 SPI 主机使用时，从机送出的数据从该管脚输入。在 SPI 接口作 SPI 从机使用时，从机待发送的数据从该管脚输出。

MOSI: 主设备输出/从设备输入管脚。在 SPI 接口作 SPI 主机使用时，主机待发送的数据从该管脚输出。在 SPI 接口作 SPI 从机使用时，主机送出的数据从该管脚输入。

SCK: SPI 的通信时钟管脚。在 SPI 接口作 SPI 主机使用时，通信时钟从此管脚输出送给外设。在 SPI 接口作 SPI 从机使用时，主机提供的通信时钟从该管脚输入以作为 SPI 接口的通信时钟。

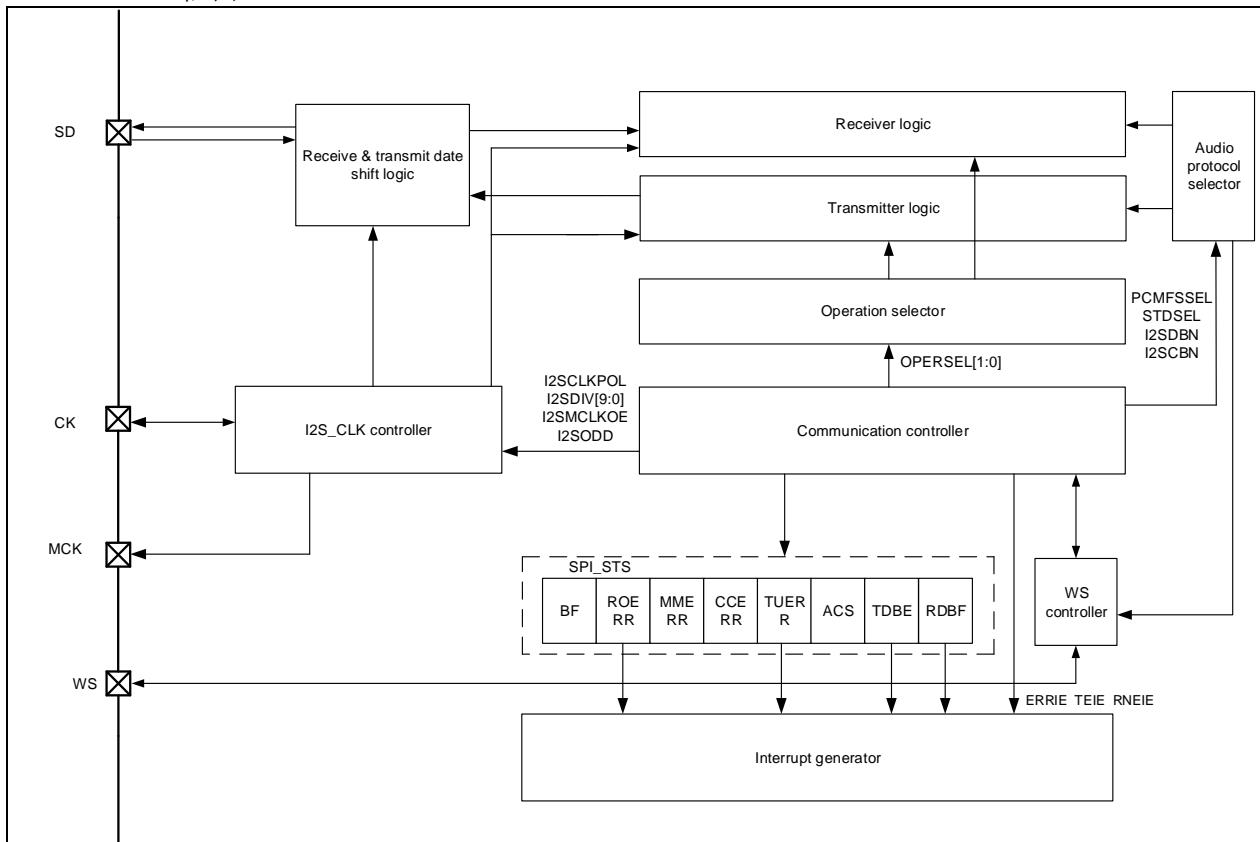
CS: 片选信号。这是一个可选的管脚，用来选中主/从设备，具体使用方式可以参见 CS 控制器章节。

13.3 I²S功能描述

13.3.1 I²S简述

I²S 根据软件配置的不同，可以分别工作在主机接收，主机发送，从机接收，从机发送四种操作模式，并且可以分别支持包括飞利浦标准，高字节对齐标准，低字节对齐标准，PCM 标准在内的共四种音频标准，并同时支持 DMA 传输。

I²S 的框图如下图所示：

图 13-18 I²S 框图

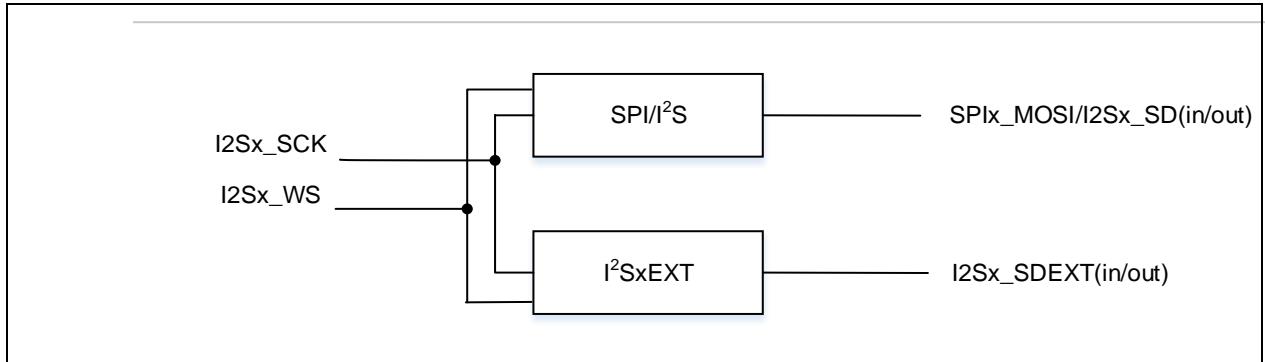
SPI 接口作为 I²S 使用时主要特征如下：

- 可编程配置的操作模式
 - 从设备发送
 - 从设备接收
 - 主设备发送
 - 主设备接收
- 可编程配置的时钟极性
- 可编程配置的时钟频率（8KHz到192KHz）
- 可编程配置的数据位数（16位，24位，32位）
- 可编程配置的声道位数（16位，32位）
- 可编程配置的音频协议
 - I²S飞利浦标准
 - 高字节对齐标准（左对齐）
 - 低字节对齐标准（右对齐）
 - PCM标准（带长或短帧同步的通道帧）
- 支持DMA传输
- 支持提供频率固定比例为256倍Fs（音频采样频率）的外设主时钟

13.3.2 I²S 全双工

为了支持I²S 全双工模式，额外例化了两个I²S 模块（I²S2EXT, I²S3EXT）。I²S2与I²S2EXT组合在一起支持全双工模式，I²S3与I²S3EXT组合在一起支持全双工模式。

注意： I²S2EXT 和 I²S3EXT 只用于 I²S 全双工模式。

图 13-19 I²S全双工结构图

x可以是2或者3

I²Sx 可以作为主机

- 半双工模式下，只有 I²Sx 可以输出 SCK 和 WS。

- 全双工模式下，只有 I²Sx 可以输出 SCK 和 WS。

新增的 I²SxEXT 只能在 I²S 全双工模式下，I²SxEXT 只可以配置成从机模式。

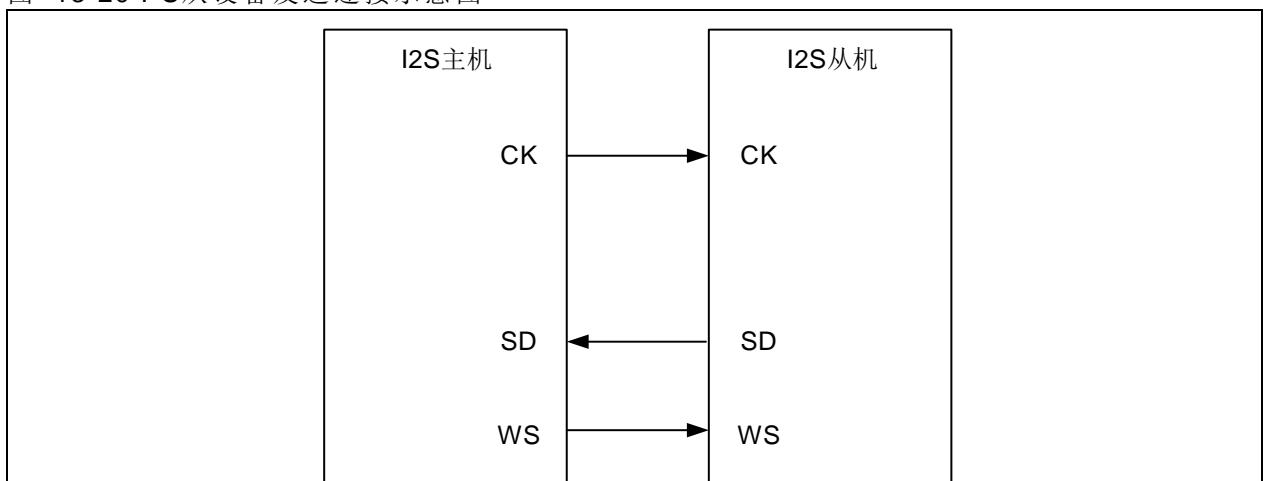
I²Sx 和 I²SxEXT 都可以做发送或者接收方。

13.3.3 操作模式选择器简述和配置流程

SPI 接口作 I²S 选择器使用时提供了多种操作模式，用户可以通过软件编程控制操作模式选择器，选择需要的操作模式，本节会分从设备发送，从设备接收，主设备发送，主设备接收四种操作模式简单介绍配置流程以及连接方式。

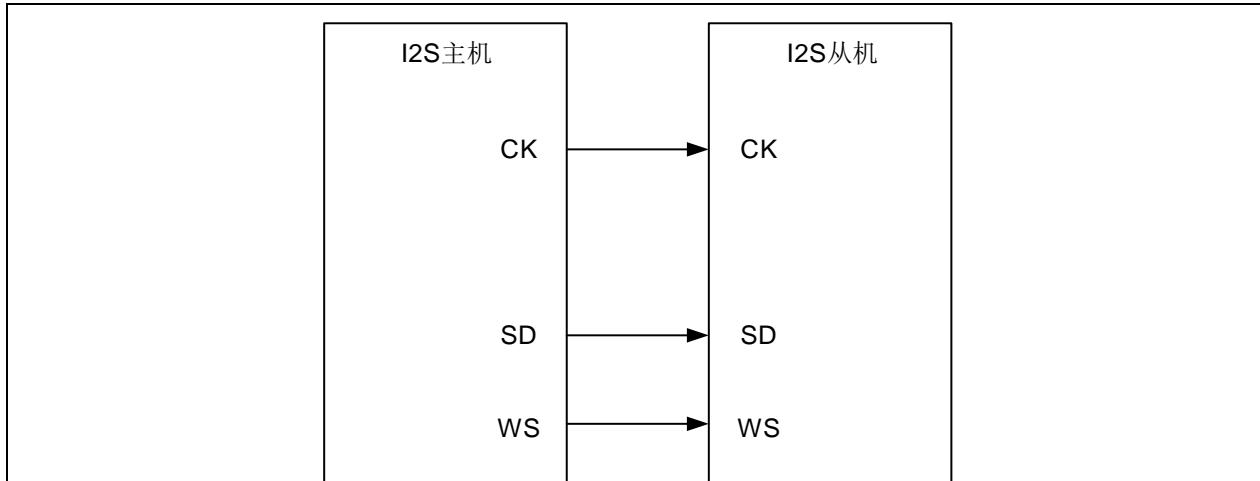
从设备发送：

置位 I²SMSEL 位，配置 OPERSEL[1: 0]位为 00，I²S 将工作在从设备发送模式下

图 13-20 I²S从设备发送连接示意图

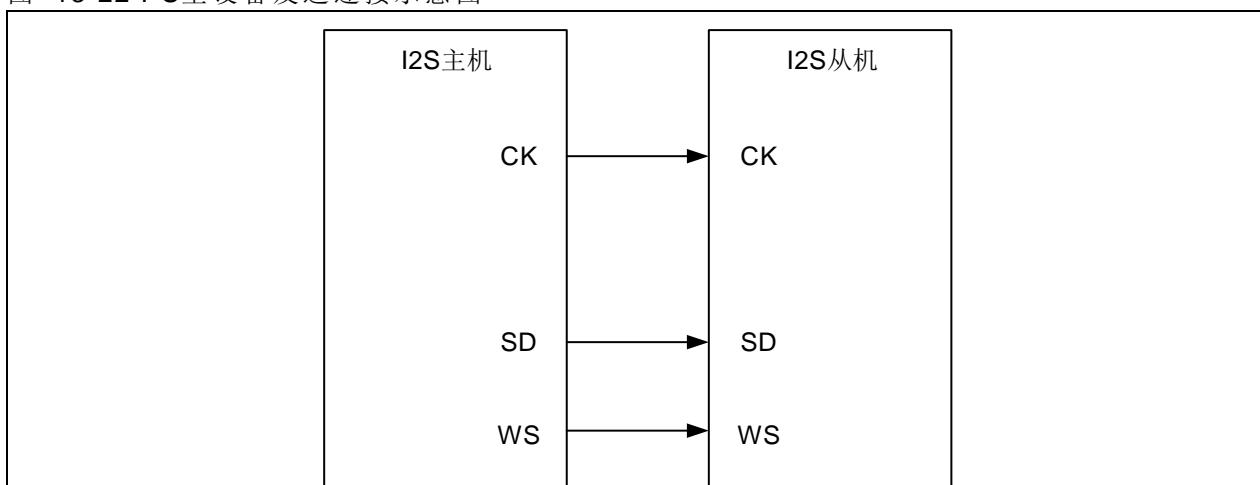
从设备接收：

置位 I²SMSEL 位，配置 OPERSEL[1: 0]位为 01，I²S 将工作在从设备接收模式下

图 13-21 I²S从设备接收连接示意图

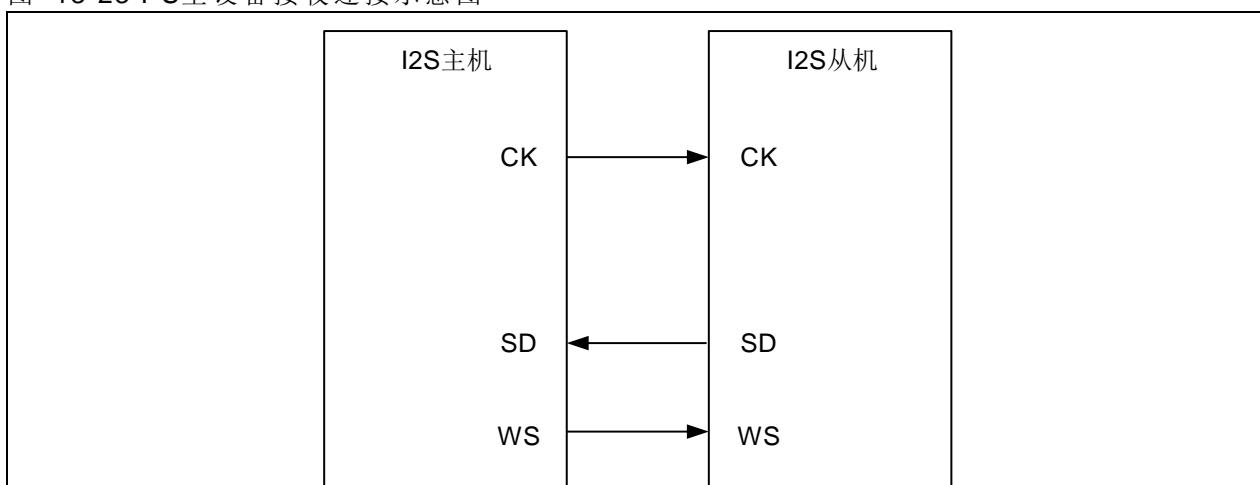
主设备发送:

置位 I2SMSEL 位, 配置 OPERSEL[1: 0]位为 10, I²S 将工作在主设备发送模式下

图 13-22 I²S主设备发送连接示意图

主设备接收:

置位 I2SMSEL 位, 配置 OPERSEL[1: 0]位为 11, I²S 将工作在主设备接收模式下

图 13-23 I²S主设备接收连接示意图

13.3.4 音频协议选择器简述和配置流程

SPI接口作为I²S使用时支持多种音频协议，用户可以通过软件编程控制音频协议选择器选择需要的音频协议，数据位个数以及声道位个数同样由音频协议选择器控制，用户同样可以通过软件编程配置的方式选择需要的数据位个数以及声道位个数，同时，音频协议选择器会自动控制WS控制器，输出或检测符合协

议要求的WS信号，具体的配置流程如下。

- 音频协议选择：配置STDSEL位选择需要的音频协议
STDSLE=00：飞利浦标准
STDSLE=01：高字节对齐标准（左对齐）
STDSLE=10：低字节对齐标准（右对齐）
STDSLE=11：PCM标准
- PCM帧同步格式选择：配置PCM长帧同步（PCMFSSEL=1）或短帧同步（PCMFSSEL=0）（该步骤在选择PCM协议时需要）
- 数据位个数选择：配置I2SDBN位选择需要的数据位个数
I2SDBN=00：16位
I2SDBN =01：24位
I2SDBN =10：32位
- 声道位个数选择：配置I2SCBN位选择需要的声道位个数
I2SCBN =0：16位
I2SCBN =1：32位

需要注意的是，不同的音频协议以及不同的数据位数和声道位数组合所对应的数据写入方式存在较大不同，下面将依次罗列所有的允许的配置组合以及其数据的读写方式。

- 飞利浦标准或PCM标准或高字节或低字节标准，16位数据，16位声道
数据位数和声道位数一致，每个声道只需读写一次SPI_DT寄存器，DMA传输个数为1。
- 飞利浦标准或PCM标准或高字节标准，16位数据，32位声道
数据位数和声道位数不一致，每个声道只需读写一次SPI_DT寄存器，DMA传输个数为1。只有前16位是有效数据，后16位数据硬件默认输出和接收0。
- 飞利浦标准或PCM标准或高字节标准，24位数据，32位声道
数据位数和声道位数不一致，每个声道需读写二次SPI_DT寄存器，DMA传输个数为2。前16位发送和接收第一笔16位数据，后16位发送和接收高8位数据，低8位数据硬件默认输出和接收0。
- 飞利浦标准或PCM标准或高字节或低字节标准，32位数据，32位声道
数据位数和声道位数一致，每个声道需读写二次SPI_DT寄存器，DMA传输个数为2。数据分两次，依次发送和接收16位数据。
- 低字节标准，16位数据，32位声道
数据位数和声道位数不一致，每个声道只需读写一次SPI_DT寄存器，DMA传输个数为1。只有后16位是有效数据，前16位数据硬件默认输出和接收0。
- 低字节标准，24位数据，32位声道
数据位数和声道位数不一致，每个声道需读写二次SPI_DT寄存器，DMA传输个数为2。前16位数据只有低八位有效，高八位数据硬件默认输出和接收0，后16位发送和接收第二笔16位数据

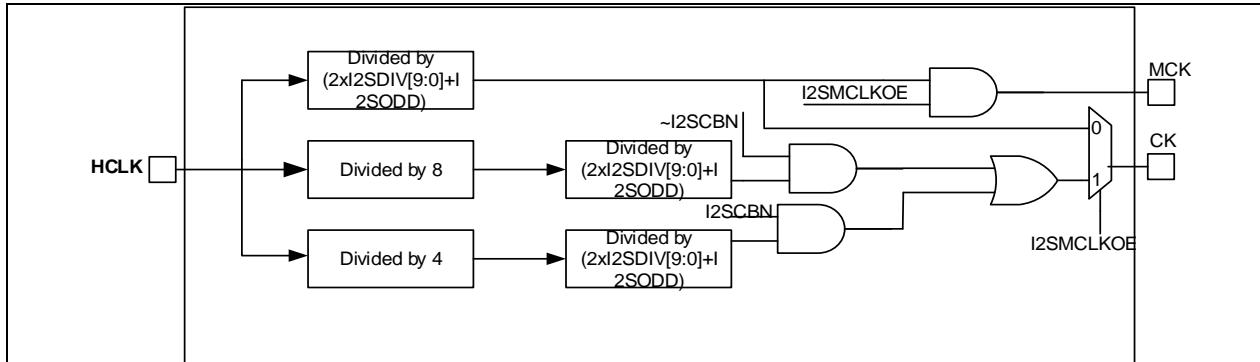
13.3.5 I2S_CLK控制器简述和配置流程

SPI接口作I²S使用时，所有该接口支持的音频协议均为同步协议，作主机时，需要产生通信时钟用于SPI接口的数据收发，并且需要将该通信时钟通过IO输出给从机，用于从机的数据收发；作从机时，需要主机提供通信时钟从IO输入到SPI接口内部作为通信时钟使用，所以实际上，I2S_CLK控制器便是扮演着产生I2S_CLK以及分配I2S_CLK的角色。

SPI接口作I²S主机时支持提供通信时钟CK以及外设主时钟MCK，CK和MCK的来源如图13-24所示，CK和MCK都是由HCLK分频得到，其中MCK的分频系数由I2SDIV以及I2SODD决定，具体计算公式见图13-24。

CK的分频系数与是否给外设提供主时钟有关，为了满足主时钟始终是音频采样频率的256倍，取决于是否提供主时钟以及声道位个数，当需要给外设提供主时钟时，CK需要先做8(I2SCBN=0时)或4(I2SCBN=1时)的预分频，随后再做和MCK相同分频系数的分频得到最终的通信时钟CK；如果不需要给外设提供主时钟，则CK的分频系数只由I2SDIV以及I2SODD决定，具体计算公式见图13-24。

图 13-24 SPI作主机CK & MCK来源示意图



除了根据上面的描述自行配制想要的时钟外，我们也提供一些特定的时钟频率其对应的 I2SDIV, I2SODD 的值，以及相应的误差，用户可以直接按此表配置 I2SDIV 和 I2SODD。

表 13-1 使用系统时钟得到精确的音频频率

SCLK (MHz)	MCK	Target Fs (Hz)	16bit				32bit			
			I2S DIV	I2S_ODD	RealFs	Error	I2S DIV	I2S_ODD	RealFs	Error
192	No	192000	15	1	193548.4	0.80%	8	0	187500	2.34%
192	No	96000	31	1	95238.1	0.79%	15	1	96774.19	0.80%
192	No	48000	62	1	48000	0%	31	1	47619.05	0.79%
192	No	44100	68	0	44117.65	0.04%	34	0	44117.65	0.04%
192	No	32000	94	0	31914.89	0.26%	47	0	31914.89	0.26%
192	No	22050	136	0	22058.82	0.04%	68	0	22058.82	0.04%
192	No	16000	187	1	16000	0%	94	0	15957.45	0.26%
192	No	11025	272	0	11029.41	0.04%	136	0	11029.42	0.04%
192	No	8000	375	0	8000	0%	187	1	8000	0%
192	Yes	96000	4	0	93750	2.34%	4	0	93750	2.34%
192	Yes	48000	8	0	46875	2.34%	8	0	46875	2.34%
192	Yes	44100	8	1	44117.65	0.04%	8	1	44117.65	0.04%
192	Yes	32000	11	1	32608.7	1.90%	11	1	32608.7	1.90%
192	Yes	22050	17	0	22058.82	0.04%	17	0	22058.82	0.04%
192	Yes	16000	23	1	15957.45	0.26%	23	1	15957.45	0.26%
192	Yes	11025	34	0	11029.41	0.04%	32	0	11029.41	0.04%
192	Yes	8000	47	0	7978.72	0.26%	47	0	7978.72	0.26%

13.3.6 DMA传输简述和配置流程

SPI 接口支持使用 DMA 进行发送数据的写入，接收数据的读取，由于无论 SPI 接口作 I²S 使用还是作 SPI 使用，对 DMA 来说，读写请求的来源都是同一个外设，所以实际上 SPI 接口作 I²S 使用时 DMA 传输的配置方法和作 SPI 使用并无不同，具体配置流程分别见下述的 DMA 发送配置流程以及 DMA 接收配置流程。

DMA 发送配置流程：

- 选择 DMA 传输通道：在 DMA 章节 DMA 通道映射表中选择用于当前所用 SPI 的 DMA 通道。
- 配置 DMA 传输目标地址：在 DMA 控制寄存器中 DMA 传输目的地址位写入当前所使用的 SPI 的 SPI_DT 寄存器地址，DMA 将会在接收到发送请求后将待发送的数据写入该地址。
- 配置 DMA 传输源地址：在 DMA 控制寄存器中 DMA 传输源地址位写入待发送数据存放的地址，DMA 将会在接收到发送请求后将该地址内的数据写入到目标地址中，即写入到当前所使用的 SPI 的 SPI_DT 寄存器中。
- 配置 DMA 传输数据个数：在 DMA 控制寄存器相关位置配置期望传输的数据个数
- 配置 DMA 传输通道优先级：在 DMA 控制寄存器相关位置配置当前所使用通道的 SPI 的 DMA 传输通道优先级。

- 配置DMA中断产生时机：在DMA控制寄存器相关位置配置是在传输完成或传输完成一半时产生DMA中断。
- 使能DMA传输通道：在DMA控制寄存器相关位置使能当前所选用的DMA通道

DMA接收配置流程：

- 选择DMA传输通道：在DMA章节DMA通道映射表中选择用于当前所用SPI的DMA通道。
- 配置DMA传输目标地址：在DMA控制寄存器中DMA传输目的地址位写入期望存放接收数据的地址，DMA将会在接收到接收请求后，将当前所使用的SPI的SPI_DT寄存器中的数据存放在目的地址中。
- 配置DMA传输源地址：在DMA控制寄存器中DMA传输源地址位写入当前所使用的SPI的SPI_DT寄存器的地址，DMA将会在接收到接收请求后将该地址内的数据写入到目标地址中，即写入到期望存放接收数据的地址。
- 配置DMA传输数据个数：在DMA控制寄存器相关位置配置期望传输的数据个数。
- 配置DMA传输通道优先级：在DMA控制寄存器相关位置配置当前所使用通道的SPI的DMA传输通道优先级。
- 配置DMA中断产生时机：在DMA控制寄存器相关位置配置是在传输完成或传输完成一半时产生DMA中断。
- 使能DMA传输通道：在DMA控制寄存器相关位置使能当前所选用的DMA通道

13.3.7 发送器接收器简述和配置流程

由于无论SPI接口作I²S使用还是作SPI使用，对于CPU来说都是同一个外设，共用同一个基址，并且SPI接口内部，作I²S使用和作SPI使用时，都共用同一个数据寄存器SPI_DT，并且实际上发送器和接收器也是共用的，所以SPI接口的发送器和接收器只是根据通信控制器的配置发送和接收期望的数据帧格式，所以如TDBE和RDBF以及ROERR等状态标志，以及TDBEIE和RDBFIE以及ERRIE等中断使能位都是共用的。

但需要注意的是：

- I²S不支持CRC校验，所以和CRC有关的操作，以及CCERR标志和与之相对应的中断都不能使用。
- I²S协议需要解析当前的声道状态，用户可以根据ACS位判断当前传输是左声道(ACS=0)还是右声道(ACS=1)。
- I²S使用TUERR位表示当前是否发生欠载，TUERR=1，表示当前发送器出现了欠载错误，如果ERRIE置位，则产生中断。
- I²S在不同的音频协议和数据位数以及声道位数的组合下，操作SPI_DT寄存器的方式是不同的，具体可以参考音频协议选择器简述和配置流程部分描述。
- I²S的关闭方式同样需要特别注意，依据不同的配置方式罗列如下
 - I2SDBN=00, I2SCBN=1, STDSLE=10：等待倒数第二个RDBF=1，等待17个CK周期，关闭I²S。
 - I2SDBN=00, I2SCBN=1, STDSLE=00或STDSLE=01或STDSLE=11：等待最后一个RDBF=1，等待一个CK时钟周期，关闭I²S。
 - 其它I2SDBN, I2SCBN, STDSLE组合：等待倒数第二个RDBF=1，等待一个CK时钟周期，关闭I²S。

下面给出发送器和接收器的配置流程

I²S发送器配置流程：

- 配置操作模式选择器
- 配置音频协议选择器
- 配置I2S_CLK控制器
- 配置DMA(若需要开启DMA传输)
- 置位I2SEN位开启I²S

I²S接收器配置流程：

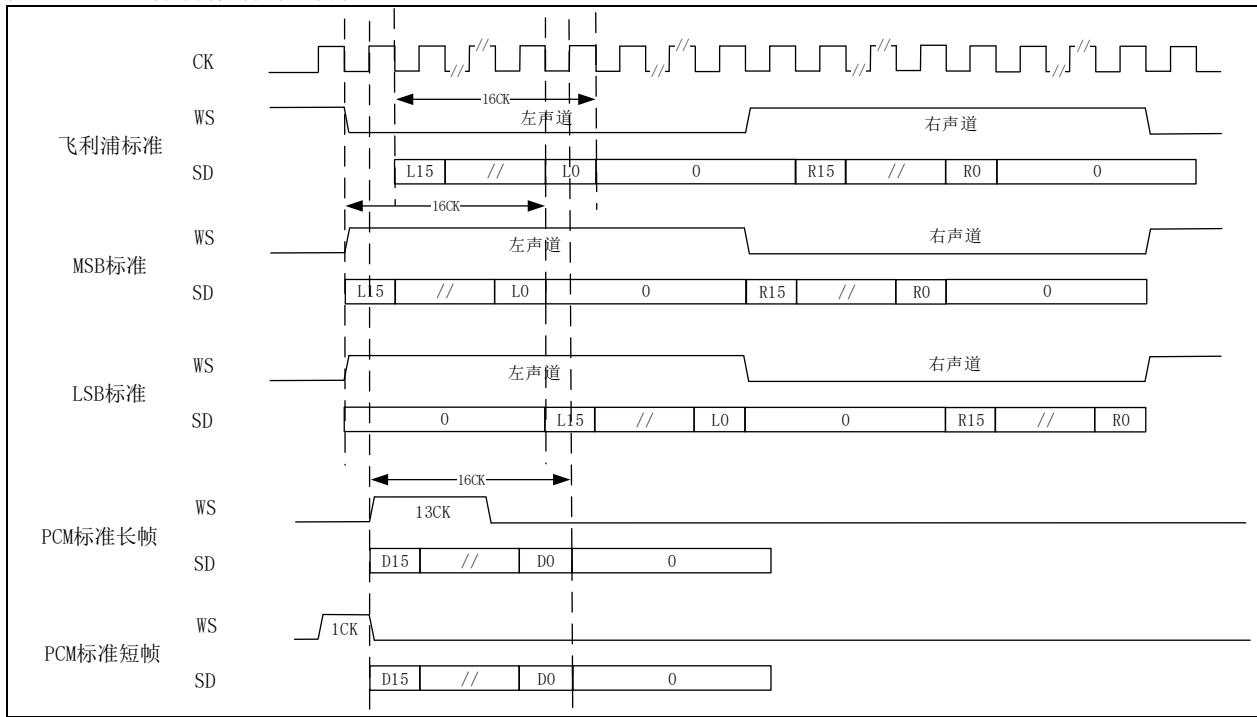
- 配置操作模式选择器
- 配置音频协议选择器
- 配置I2S_CLK控制器

- 配置DMA(若需要开启DMA传输)
- 置位I2SEN位开启I²S

13.3.8 I²S通信时序

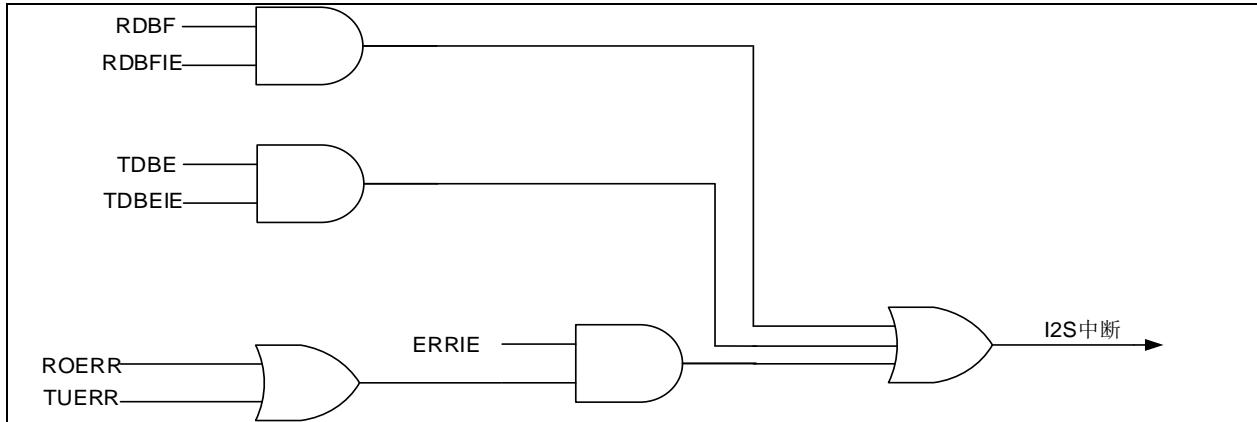
I²S 支持以下 4 种音频协议：飞利浦标准，高字节对齐标准（左对齐），低字节对齐标准（右对齐），PCM 标准，各标准音频时序如下。

图 13-25 各音频标准时序



13.3.9 中断

图 13-26 I²S中断



13.3.10 管脚控制

SPI 接口作 I²S 使用时，I²S 传输需要三个管脚，分别是数据管脚 SD，同步管脚 WS，通信时钟管脚 CK，如果需要给外设提供主时钟还需要主时钟输出管脚 MCK，由于一个 SPI 接口不可能同时作 I²S 和 SPI 使用，所以 I²S 和 SPI 部分管脚映射是共用的各管脚的映射和定义如下。

- SD：数据管脚（和 MOSI 管脚共用同样的 GPIO 映射关系），数据的双向收发管脚。
- WS：同步管脚（和 CS 管脚共用同样的 GPIO 映射关系），通信同步信号的双向控制管脚，主模式输出，从模式输入。
- CK：通信时钟管脚（和 SCK 管脚共用同样的 GPIO 映射关系），通信时钟双向输入输出管脚，主模式输出，从模式输入。

- MCK：主时钟管脚（独立映射），主时钟输出管脚，用于给外设提供主时钟，输出的时钟频率固定为音频采样频率的256倍。

13.4 SPI寄存器

必须用字（32位）的方式操作这些外设寄存器。

表 13-2 SPI寄存器列表及其复位值

寄存器简称	基址偏移量	复位值
SPI_CTRL1	0x00	0x0000
SPI_CTRL2	0x04	0x0000
SPI_STS	0x08	0x0002
SPI_DT	0x0C	0x0000
SPI_CPOLY	0x10	0x0007
SPI_RCRC	0x14	0x0000
SPI_TCRC	0x18	0x0000
SPI_I2SCTRL	0x1C	0x0000
SPI_I2SCLK	0x20	0x0002

13.4.1 SPI控制寄存器1 (SPI_CTRL1) (I2S模式下不使用)

域	简称	复位值	类型	功能
位 15	SLBEN	0x0	rw	单线双向半双工模式使能 (Single line bidirectional half-duplex enable) 0: 关闭; 1: 开启。
位 14	SLBTD	0x0	rw	单线双向半双工模式传输方向 (Single line bidirectional half-duplex transmission direction) 和 SLBEN 位一起决定在“单线双向半双工”模式下数据的传输方向 0: 接收模式; 1: 发送模式。
位 13	CCEN	0x0	rw	CRC 校验使能 (CRC calculation enable) 0: 关闭; 1: 开启。
位 12	NTC	0x0	rw	下一笔传输数据为 CRC (Next transmission CRC) 该位置起表示下一笔传输的数据为 CRC 数据。 0: 普通数据; 1: CRC 数据。
位 11	FBN	0x0	rw	帧位个数 (frame bit num) 该位配置发送/接收时数据帧位个数。 0: 8 位; 1: 16 位。
位 10	ORA	0x0	rw	仅接收有效 (Only receive active) 在“双线单向”模式时, 该位置起表示只有接收有效, 发送被禁止。 0: 发送和接收; 1: 仅接收。
位 9	SWCSEN	0x0	rw	软件 CS 模式使能 (Software CS enable) 当该位被置起时, CS 管脚上的电平由 SWCSIL 位的值决定, 此时在 CS 管脚上的 I/O 电平状态无效。 0: 关闭; 1: 开启。
位 8	SWCSIL	0x0	rw	软件 CS 内部电平 (Software CS internal level) 该位只在 SWCSEN 位置起时有意义, 它决定了 CS 上的内部电平状态。 做主设备时, 该位必须设置置起。 0: 低电平; 1: 高电平。
位 7	LTF	0x0	rw	LSB 先传输 (LSB transmit first) 该位用于选择数据先传输 MSB 还是 LSB。 0: MSB; 1: LSB。
位 6	SPIEN	0x0	rw	SPI 使能 (SPI enable) 0: 关闭; 1: 开启。
位 5: 3	MDIV	0x0	rw	主模式时钟频率分频系数 (Master clock frequency division) 作主模式时, 分频系数对外设时钟进行分频, 作为 SPI 时钟, MDIV[3]位在 SPI_CTRL2 寄存器, MDIV[3: 0]: 0000: 2 分频 0001: 4 分频 0010: 8 分频 0011: 16 分频 0100: 32 分频 0101: 64 分频 0110: 128 分频 0111: 256 分频 1000: 512 分频 1001: 1024 分频

位 2	MSTEN	0x0	rw	主模式使能 (Master enable) 0: 关闭 (从设备)； 1: 开启 (主设备)。
位 1	CLKPOL	0x0	rw	时钟极性 (Clock polarity) 空闲时时钟输出的极性。 0: 低电平； 1: 高电平。
位 0	CLKPHA	0x0	rw	时钟相位 (Clock phase) 0: 第一个边沿进行数据捕获； 1: 第二个边沿进行数据捕获。

注：在 I²S 模式下，SPI_CTRL1 寄存器需置 0。

13.4.2 SPI控制寄存器2 (SPI_CTRL2)

域	简称	复位值	类型	功能
位 15: 10	保留	0x00	resd	硬件强制为 0
位 9	MDIV3EN	0x0	rw	主模式时钟频率三分频使能 (Master clock frequency3 division enable) 0: 关闭； 1: 开启。 注：该位开启时，MDIV[3: 0]无效，SPI 时钟被强制为 PCLK 三分频。使用 SPI 三分频功能，且 PCLK/SYSCLK ≠ 1 时，SPI 时钟不满足 50% 占空比 具体时序参考数据手册
位 8	MDIV[3]	0x0	rw	主模式时钟频率分频系数 (Master clock frequency division) 详见 MDIV[2: 0]在 SPI_CTRL1 寄存器。
位 7	TDBEIE	0x0	rw	发送数据缓冲器空中断使能 (Transmit data buffer empty interrupt enable) 0: 关闭； 1: 开启。
位 6	RDBFIE	0x0	rw	接收数据缓冲器满中断使能 (Receive data buffer full interrupt enable) 0: 关闭； 1: 开启。
位 5	ERRIE	0x0	rw	错误中断使能 (Error interrupt enable) 当错误 (CCERR、MMERR、ROERR、TUERR) 产生时，该位控制是否产生中断 0: 关闭； 1: 开启。
位 4	TIEN	0x0	rw	TI 模式使能 (TI mode enable) 0: 关闭 (Motorola 模式)； 1: 开启 (TI 模式)。 注：该位在 I ² S 模式下没有用，在 I ² S 模式下需保持为 0。
位 3	保留	0x0	resd	保持默认值。
位 2	HWCSOE	0x0	rw	硬件 CS 输出使能 (Hardware CS output enable) 该位做主设备时才有意义，设置为'1'时，CS 脚 I/O 口输出低电平，设置为'0'时，必须保证 CS 脚 I/O 口输入为高电平。 0: 关闭； 1: 开启。
位 1	DMATEN	0x0	rw	DMA 发送使能 (DMA transmit enable) 0: 关闭； 1: 开启。
位 0	DMAREN	0x0	rw	DMA 接收使能 (DMA receive enable) 0: 关闭； 1: 开启。

13.4.3 SPI状态寄存器 (SPI_STS)

域	简称	复位值	类型	功能
位 15: 9	保留	0x00	resd	硬件强制为 0
位 8	CSPAS	0x0	ro	CS 脉冲异常置位标志 (CS pulse abnormal setting flag) 0: 无异常; 1: 有异常置位; 注: 该位用于 TI slave mode, 由软件读 STS 寄存器清零。
位 7	BF	0x0	ro	通信忙标志 (Busy flag) 0: 通信空闲; 1: 通信忙。
位 6	ROERR	0x0	ro	接收器溢出错误 (Receiver overflow error) 0: 无; 1: 有。
位 5	MMERR	0x0	ro	主模式错误 (Master mode error) 该位由硬件置位, 软件清除 (先读或写 SPI_STS 寄存器, 再写 SPI_CTRL1 寄存器)。 0: 无; 1: 有。
位 4	CCERR	0x0	rw0c	CRC 校验错误 (CRC calculation error) 该位由硬件置起, 由软件清除。 0: 正确; 1: 错误。
位 3	TUERR	0x0	ro	发送器欠载错误 (Transmitter underload error) 该位由硬件置起, 软件清除 (读 SPI_STS 寄存器)。 0: 无; 1: 有。 注: 该位只在 I ² S 模式使用。
位 2	ACS	0x0	ro	音频通道状态 (Audio channel state) 该位表示当前传输的音频左右声道状态。 0: 左声道; 1: 右声道。 注: 该位只在 I ² S 模式使用。
位 1	TDBE	0x1	ro	发送数据缓冲器空 (Transmit data buffer empty) 0: 非空; 1: 空。
位 0	RDBF	0x0	ro	接收数据缓冲器满 (Receive data buffer full) 0: 未满; 1: 满。

13.4.4 SPI数据寄存器 (SPI_DT)

域	简称	复位值	类型	功能
位 15: 0	DT	0x0000	rw	数据值 (Data value) 该寄存器包含读和写的功能, 当数据位配置为 8 位时, 该寄存器只有低 8 位[7: 0]有效。

13.4.5 SPICRC多项式寄存器 (SPI_CPOLY) (I2S模式下不使用)

域	简称	复位值	类型	功能
位 15: 0	CPOLY	0x0007	rw	CRC 多项式寄存器 (CRC polynomial) 该寄存器为 CRC 计算时用到的多项式, 可以根据应用设置。 注: 该寄存器只在 SPI 模式下使用。

13.4.6 SPIRxCRC寄存器 (SPI_RCRC) (I2S模式下不使用)

域	简称	复位值	类型	功能
位 15: 0	RCRC	0x0000	ro	接收 CRC 寄存器 (receive CRC) CRC 使能后, 该寄存器值为根据接收到的数据计算得到的 CRC 值, 要复位该寄存器, 需操作 SPI_CTRL1 的 CCEN 位先清除再置起。 当数据位配置为 8 位时, 该寄存器只有低 8 位[7: 0]有效, 按照 CRC8 计算; 当数据位配置为 16 位时, 按照 CRC16 计算。 注: 该寄存器只在 SPI 模式下使用。

13.4.7 SPITxCRC寄存器 (SPI_TCRC)

域	简称	复位值	类型	功能
位 15: 0	TCRC	0x0000	ro	发送 CRC 寄存器 (transmit CRC) CRC 使能后, 该寄存器值为根据发送的数据计算得到的 CRC 值。要复位该寄存器, 需操作 SPI_CTRL1 的 CCEN 位先清除再置起。 当数据位配置为 8 位时, 该寄存器只有低 8 位[7: 0]有效, 按照 CRC8 计算; 当数据位配置为 16 位时, 按照 CRC16 计算。 注: 该寄存器只在 SPI 模式下使用。

13.4.8 SPI_I2S配置寄存器 (SPI_I2SCTRL)

域	简称	复位值	类型	功能
位 15: 12	保留位	0x0	resd	硬件强制为 0
位 11	I2SMSEL	0x0	rw	I ² S 模式选择 (I ² S mode select) 0: SPI 模式; 1: I ² S 模式。
位 10	I2SEN	0x0	rw	I ² S 使能 (I ² S enable) 0: 关闭; 1: 开启。
位 9: 8	OPERSEL	0x0	rw	I ² S 操作选择 (I ² S operation select) 00: 从设备发送; 01: 从设备接收; 10: 主设备发送; 11: 主设备接收。
位 7	PCMFSSEL	0x0	rw	PCM 帧同步 (PCM frame synchronization select) 该位只在使用 PCM 标准时才有意义。 0: 短帧同步; 1: 长帧同步。
位 6	保留位	0x0	resd	保持默认值。
位 5: 4	STDSEL	0x0	rw	I ² S 标准选择 (I ² S standard select) 00: 飞利浦标准;

				01: 高字节对齐标准（左对齐）； 10: 低字节对齐标准（右对齐）； 11: PCM 标准。
位 3	I2SCLKPOL	0x0	rw	I ² S 时钟极性 (I ² S clock polarity) 时钟管脚上总线空闲时时钟输出的极性。 0: 低电平； 1: 高电平。
位 2: 1	I2SDBN	0x0	rw	I ² S 数据位个数 (I ² S data bit num) 00: 16 位； 01: 24 位； 10: 32 位； 11: 不允许。
位 0	I2SCBN	0x0	rw	I ² S 声道位个数 (I ² S channel bit num) 该位只有在 I ² S 数据位个数为 16 位时配置才有意义，否则都由硬件固定为 32 位。 0: 16 位宽； 1: 32 位宽。

13.4.9 SPI_I2S预分频寄存器 (SPI_I2SCLK)

域	简称	复位值	类型	功能
位 15: 12	保留位	0x0	resd	硬件强制为 0
位 9	I2SMCLKOE	0x0	rw	I ² S 主设备时钟输出使能 (I ² S Master clock output enable) 0: 关闭； 1: 开启。
位 8	I2SODD	0x0	rw	I ² S 分频系数配置奇数 (Odd result for I ² S division) 0: 实际分频系数=I2SDIV*2； 1: 实际分频系数=(I2SDIV*2)+1。
位 11: 10 位 7: 0	I2SDIV	0x02	rw	I ² S 分频系数 (I ² S division) I2SDIV[9: 0]禁止设置为 0 或者 1。

14 全双工音频接口 (I2SF)

14.1 全双工音频接口 (I2SF) 简介

I2SF 音频接口在串行外设接口 (SPI) 的 I²S 功能基础上新增了全双工的功能，同时 I2SF 的输入主时钟可以来源于系统时钟，PLL 输出时钟，HICK 输出时钟以及外部输入时钟。通过配置 I2SF 的输入主时钟，可以得到更加精确的音频频率。

14.2 I2SF功能描述

I2SF 的详细功能描述参考第 13 章的 I²S，且在此基础上新增了如下内容。

14.2.1 I2SF全双工

I2SF 可以通过配置 I2SF_I2SCTRL 寄存器 I2SF_DUPEN 位 ‘1’ 使能全双工功能，寄存器 OPERSEL[1:0] 配置 I2SF 为主机/从机模式，寄存器 OPERSEL[1:0] 配置 I2SF 为主发/从发模式时，SD 为输出，SDEXT 为输入；寄存器 OPERSEL[1:0] 配置 I2SF 为主收/从收模式时，SD 为输入，SDEXT 为输出。

I2SF 全双工模式下通讯连接如下图所示：

图 14-1 I2SF全双工主发/从发通讯

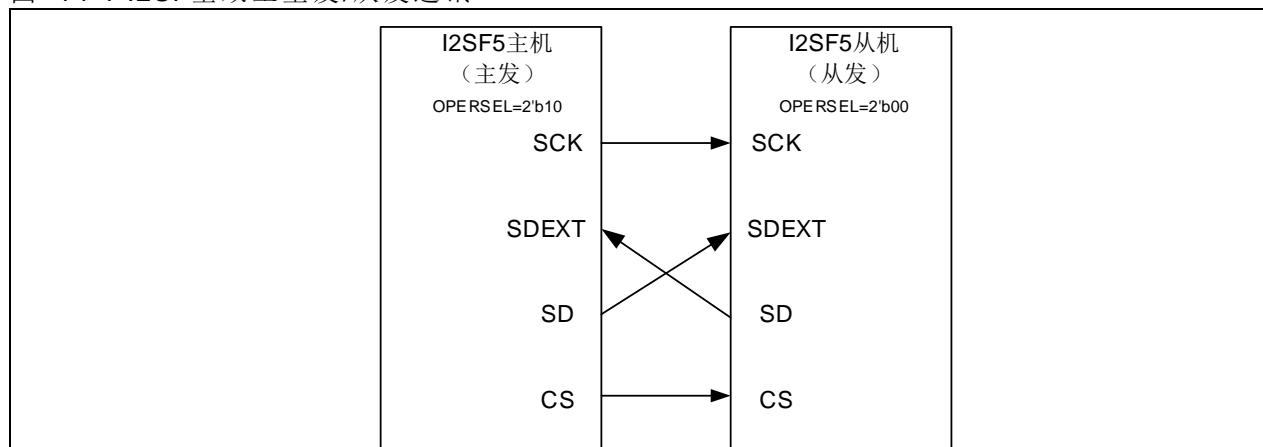


图 14-2 I2SF全双工主发/从收通讯

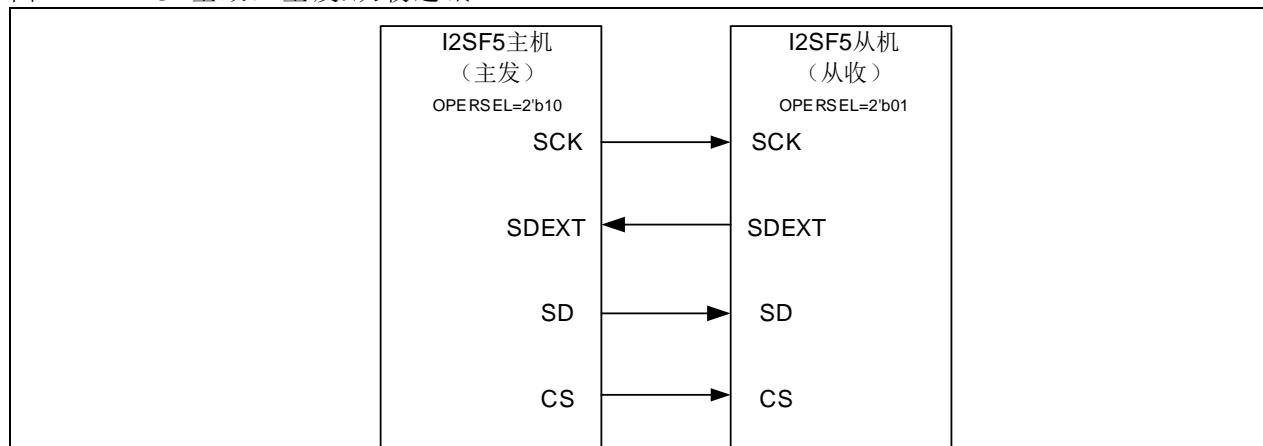


图 14-3 I2SF全双工主收/从发通讯

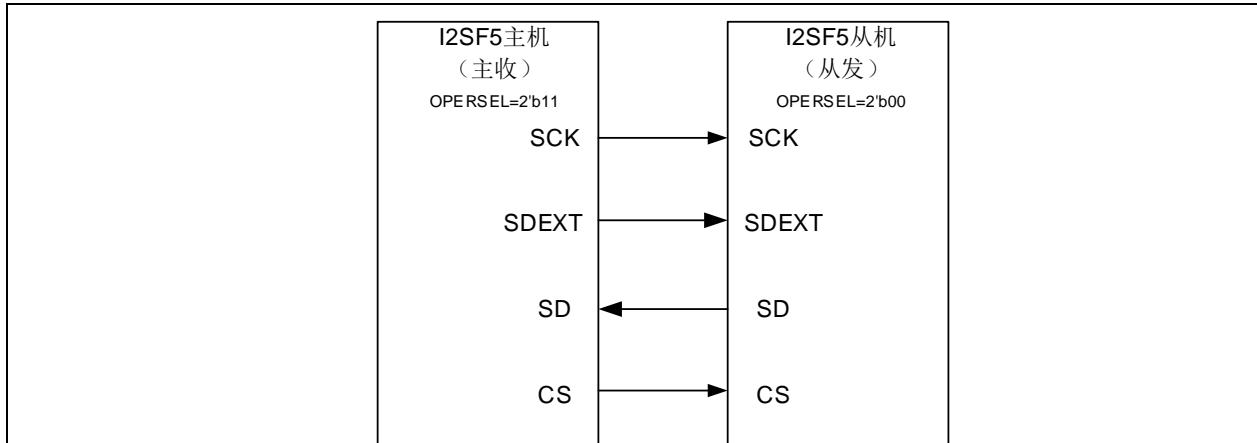
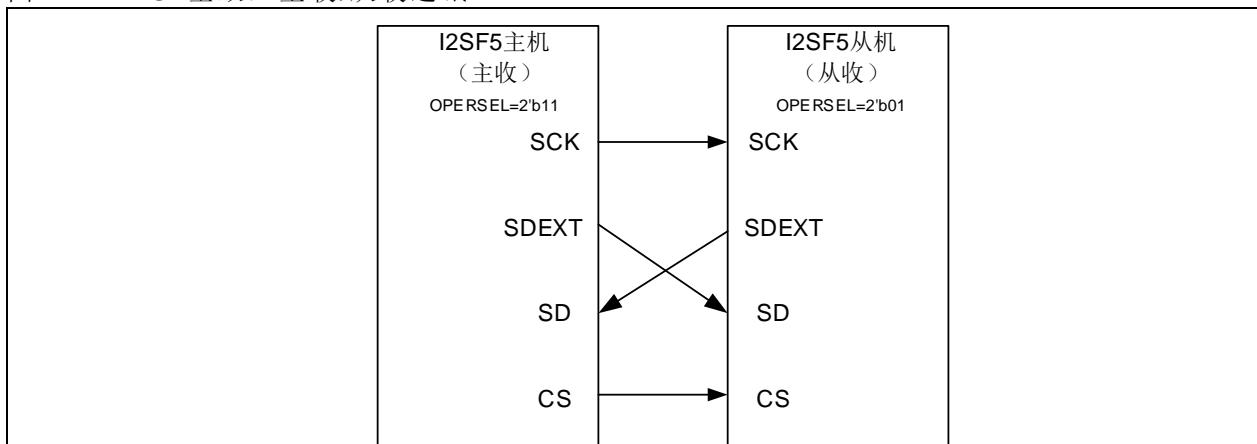


图 14-4 I2SF全双工主收/从收通讯



主模式下全双工传输 (`I2SFDUPEN=1` 并且 `OPERSEL[9]=1`)

- 当写入数据到I2SF_DT寄存器（发送缓冲器）后，传输开始；
- 在传送第一位数据的同时，数据被并行地从发送缓冲器传送到移位寄存器中，然后按顺序被串行移位到SD/SDEXT引脚上；
- 与此同时，在SDEXT/SD引脚上接收到的数据，按顺序被串行移位到移位寄存器中，然后并被串行地传送到I2SF_DT寄存器中。

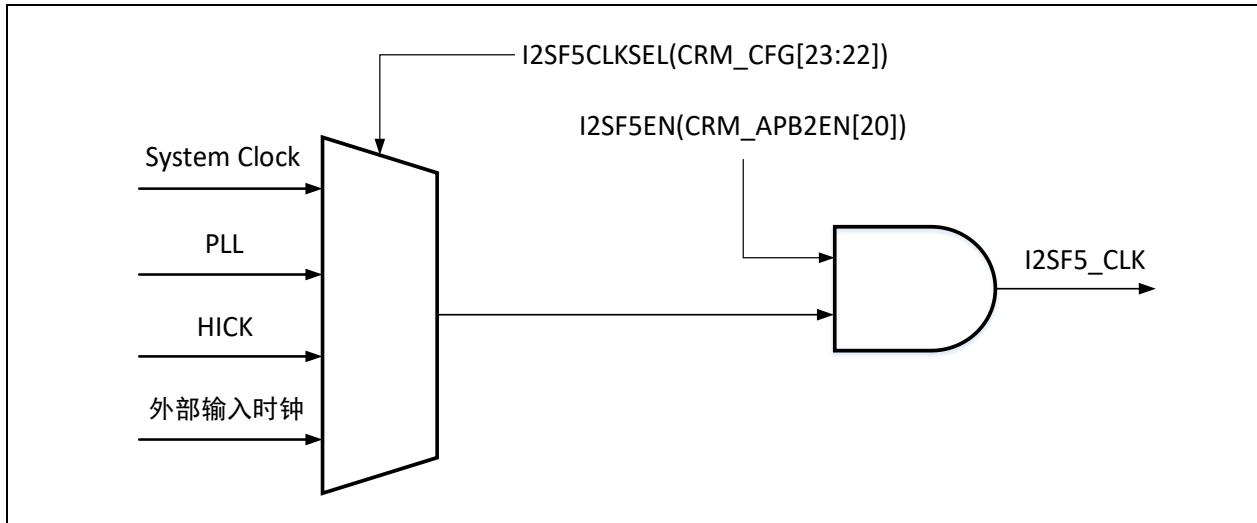
从模式下全双工传输 (`I2SFDUPEN=1` 并且 `OPERSEL[9]=0`)

- 当从机接收到时钟信号时，数据传输开始，随后SDEXT/SD的数据依次移位进入移位寄存器，每完成16位数据的传输，就将移位寄存器的数据传送到I2SF_DT寄存器中。
- 与此同时，I2SF_DT寄存器（发送缓冲器）中的数据并行传送到移位寄存器中，随后被串行地发送到SD/SDEXT引脚上。软件必须保证在I2SF主机开始数据传输之前写入I2SF_DT寄存器待发送数据。

14.2.2 I2SF 主时钟源选择

I2SF控制器有四个时钟来源，可以通过I2SF5CLKSEL[1: 0]寄存器进行配置。其中外部输入时钟由IO管脚输入，HICK为HICK振荡器时钟输出48MHz，PLL来源于时钟PLLP。

图 14-5 I2SF主时钟来源选择



14.2.3 PCM模式通信时序

I2SF新增寄存器I2SFPCMCKSEL用于控制PCM长帧或短帧格式选择通讯时钟的上升沿/下降沿进行数据采样，各种配置下的通讯时序图如下所示。

图 14-6 下降沿采样PCM通讯时序图

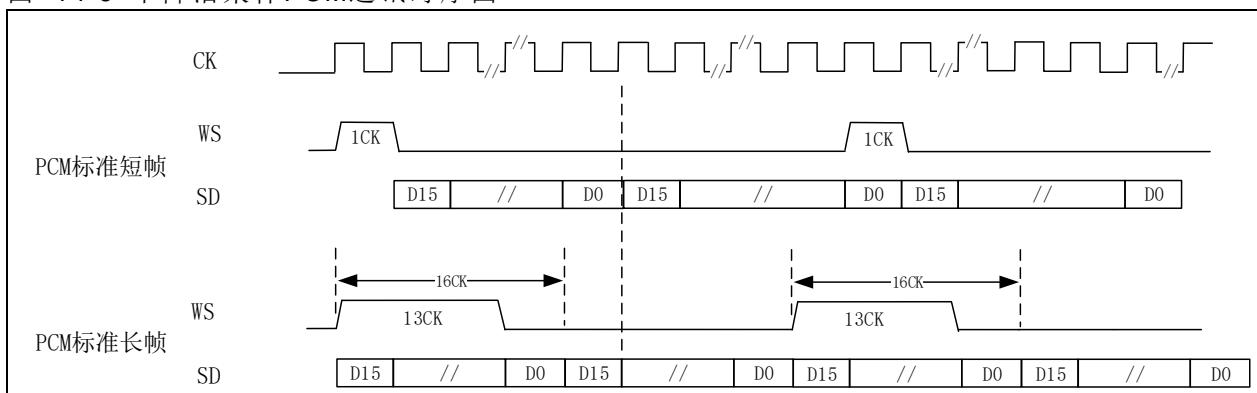
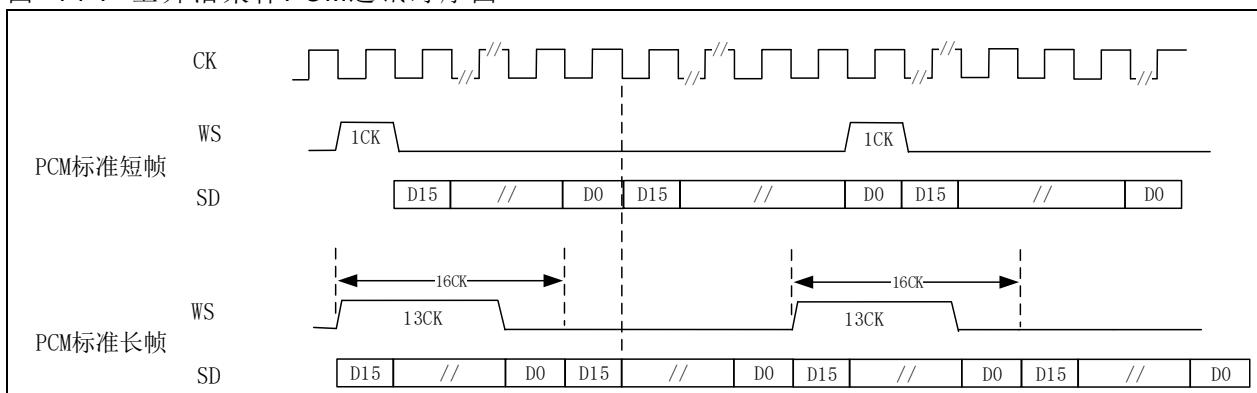
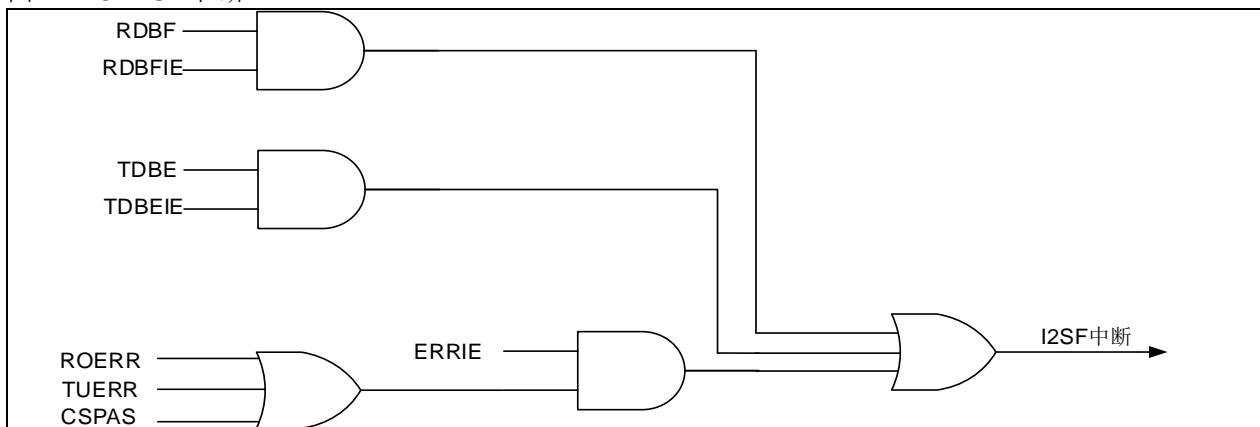


图 14-7 上升沿采样PCM通讯时序图



14.2.4 中断

图 14-8 I2SF中断



14.2.5 IO管脚控制

I2SF 在全双工功能关闭的情况下，需要三个管脚实现 I²S 通讯，分别是数据管脚 SD，同步管脚 WS，通讯时钟管脚 CK，如果需要给外设提供主时钟还需要主时钟输出管脚 MCK。在全双工功能打开的情况下，另外需要数据管脚 SDEXT 用于双向数据传输。I2SF 在主机模式下，可以通过配置寄存器 I2SF5CLKSEL[1:0]选择 CLKIN_IN 作为 I²S 主时钟外部输入接口。

- **SD:** 数据管脚（和 MOSI 管脚共用同样的 GPIO 映射关系），在 I2SF 全双工功能关闭情况下为数据的双向收发管脚，在全双工功能打开情况下，根据寄存器配置决定数据传输方向。
- **SDEXT:** 在 I2SF 全双工功能关闭情况下不可用，在全双工功能打开情况下，根据寄存器配置决定数据传输方向。
当 OPERSEL[9:8] 配置为主发/从发情况下，SD 为输出，SDEXT 为输入；当 OPERSEL[9:8] 配置为主收/从收情况下，SD 为输入，SDEXT 为输出。
- **WS:** 同步管脚（和 CS 管脚共用同样的 GPIO 映射关系），通信同步信号的双向控制管脚，主模式输出，从模式输入。
- **CK:** 通信时钟管脚（和 SCK 管脚共用同样的 GPIO 映射关系），通信时钟双向输入输出管脚，主模式输出，从模式输入。
- **CLKIN_IN:** 主机模式下，I²S 主时钟外部输入接口，从机模式不可用。
- **MCK:** 主时钟管脚（独立映射），主时钟输出管脚，用于给外设提供主时钟，输出的时钟频率固定为音频采样频率的 256 倍。

14.2.6 I2SF注意事项

- APB2 的配置时钟应该大于 I2SF 的通信时钟。
- 在从机模式下，CS 线在不期望的地方检测到边沿时，CS 脉冲异常置位标志由硬件置位，并且通过软件读该寄存器时进行清零。

14.3 I2SF寄存器

必须用字（32 位）的方式操作这些外设寄存器。

表 14-1 I2SF5 寄存器列表及其复位值

寄存器简称	基址偏移量	复位值
I2SF_CTRL2	0x04	0x0000
I2SF_STS	0x08	0x0002
I2SF_DT	0x0C	0x0000
I2SF_I2SCTRL	0x1C	0x0000
I2SF_I2SCLKP	0x20	0x0002
I2SF_MISC1	0x30	0x0000

14.3.1 I2SF控制寄存器2 (I2SF_CTRL2)

域	简称	复位值	类型	功能
位 15: 8	保留	0x00	resd	硬件强制为 0
位 7	TDBEIE	0x0	rw	发送数据缓冲器空中断使能 (Transmit data buffer empty interrupt enable) 0: 关闭; 1: 开启。
位 6	RDBFIE	0x0	rw	接收数据缓冲器满中断使能 (Receive data buffer full interrupt enable) 0: 关闭; 1: 开启。
位 5	ERRIE	0x0	rw	错误中断使能 (Error interrupt enable) 当错误 (CCERR、MMERR、ROERR、TUERR) 产生时, 该位控制是否产生中断 0: 关闭; 1: 开启。
位 4: 2	保留	0x0	resd	保持默认值。
位 1	DMATEN	0x0	rw	DMA 发送使能 (DMA transmit enable) 0: 关闭; 1: 开启。
位 0	DMAREN	0x0	rw	DMA 接收使能 (DMA receive enable) 0: 关闭; 1: 开启。

14.3.2 I2SF状态寄存器 (I2SF_STS)

域	简称	复位值	类型	功能
位 15: 9	保留	0x00	resd	硬件强制为 0
位 8	CSPAS	0x0	ro	CS 脉冲异常置位标志 (CS pulse abnormal setting flag) 0: 无异常; 1: 有异常置位;
位 7	BF	0x0	ro	通信忙标志 (Busy flag) 0: 通信空闲; 1: 通信忙。
位 6	ROERR	0x0	ro	接收器溢出错误 (Receiver overflow error) 0: 无; 1: 有。
位 5: 4	保留	0x0	resd	硬件强制为 0
位 3	TUERR	0x0	ro	发送器欠载错误 (Transmitter underload error) 该位由硬件置起, 软件清除 (读 SPI_STS 寄存器)。 0: 无; 1: 有。
位 2	ACS	0x0	ro	音频通道状态 (Audio channel state) 该位表示当前传输的音频左右声道状态。 0: 左声道; 1: 右声道。
位 1	TDBE	0x1	ro	发送数据缓冲器空 (Transmit data buffer empty) 0: 非空; 1: 空。
位 0	RDBF	0x0	ro	接收数据缓冲器满 (Receive data buffer full) 0: 未满; 1: 满。

14.3.3 I2SF数据寄存器 (I2SF_DT)

域	简称	复位值	类型	功能
位 15: 0	DT	0x0000	rw	数据值 (Data value) 该寄存器包含读和写的功能, 当数据位配置为 8 位时, 该寄存器只有低 8 位[7: 0]有效。

14.3.4 I2SF配置寄存器 (I2SF_I2SCTRL)

域	简称	复位值	类型	功能
位 15: 12	保留位	0x0	resd	硬件强制为 0
位 13	I2SFDUPEN	0x0	rw	I ² S 全双工使能 (I ² S full duplex enable) 0: 全双工关闭 1: 全双工使能
位 12: 11	保留位	0x0	resd	硬件强制为 0
位 10	I2SEN	0x0	rw	I ² S 使能 (I ² S enable) 0: 关闭; 1: 开启。
位 9: 8	OPERSEL	0x0	rw	I ² S 操作选择 (I ² S operation select) 00: 从设备发送; 01: 从设备接收; 10: 主设备发送; 11: 主设备接收。
位 7	PCMFSSEL	0x0	rw	PCM 帧同步 (PCM frame synchronization select) 该位只在使用 PCM 标准时才有意义。 0: 短帧同步; 1: 长帧同步。
位 6	保留位	0x0	resd	保持默认值。
位 5: 4	STDSEL	0x0	rw	I ² S 标准选择 (I ² S standard select) 00: 飞利浦标准; 01: 高字节对齐标准 (左对齐); 10: 低字节对齐标准 (右对齐); 11: PCM 标准。
位 3	I2SCLKPOL	0x0	rw	I ² S 时钟极性 (I ² S clock polarity) 时钟管脚上总线空闲时时钟输出的极性。 0: 低电平; 1: 高电平。
位 2: 1	I2SDBN	0x0	rw	I ² S 数据位个数 (I ² S data bit num) 00: 16 位; 01: 24 位; 10: 32 位; 11: 不允许。
位 0	I2SCBN	0x0	rw	I ² S 声道位个数 (I ² S channel bit num) 该位只有在 I ² S 数据位个数为 16 位时配置才有意义, 否则都由硬件固定为 32 位。 0: 16 位宽; 1: 32 位宽。

14.3.5 I2SF预分频寄存器 (I2SF_I2SCLKP)

域	简称	复位值	类型	功能
位 15: 12	保留位	0x0	resd	硬件强制为 0
位 9	I2SMCLKOE	0x0	rw	I ² S 主设备时钟输出使能 (I ² S Master clock output enable) 0: 关闭; 1: 开启。
位 8	I2SODD	0x0	rw	I ² S 分频系数配置奇数 (Odd result for I ² S division) 0: 实际分频系数=I2SDIV*2; 1: 实际分频系数=(I2SDIV*2)+1。
位 11: 10 位 7: 0	I2SDIV	0x02	rw	I ² S 分频系数 (I ² S division) I2SDIV[9: 0]禁止设置为 0 或者 1。

14.3.6 I2SF额外寄存器1 (I2SF_MISC1)

域	简称	复位值	类型	功能
位 15: 1	保留位	0x0	resd	硬件强制为 0
位 0	I2SFPCMCKSEL	0x02	rw	I ² S PCM 模式时钟边沿选择 (I ² S PCM clock edge select) 0: PCM 模式选择下降沿作为采样沿 1: PCM 模式选择上升沿作为采样沿

15 定时器 (TIMER)

AT32F455 定时器种类有基本定时器、通用定时器、高级控制定时器，详细功能模式可参考 [15.1~15.4](#) 节说明，下表为各种类型定时器的功能总表。

表 15-1 TMR功能对比

Timer 类型	Timer	计数位数	计数方式	重复计数器	预分频系数	DMA 请求产生	捕获/比较通道	PWM 输入模式	EXT 输入	刹车输入
高级控制定时器	TMR1 TMR8	16	向上 向下 向上/向下	16 位	1~65536	支持	4	支持	支持	支持
	TMR2 TMR5	16/32	向上 向下 向上/向下	不支持	1~65536	支持	4	支持	仅 TMR2 支持	不支持
	TMR3 TMR4	16	向上 向下 向上/向下	不支持	1~65536	支持	4	支持	支持	不支持
通用定时器	TMR9 TMR12	16	向上 向下 向上/向下	8 位	1~65536	支持	2	支持	不支持	支持
	TMR10 TMR11 TMR13 TMR14	16	向上 向下 向上/向下	8 位	1~65536	支持	1	不支持	不支持	支持
基本定时器	TMR6 TMR7	16	向上	不支持	1~65536	支持	不支持	不支持	不支持	不支持
Timer 类型	Timer	计数位数	计数方式	PWM 输出	单周期输出	互补输出	死区	编码器接口连接	霍尔传感器接口连接	连动外设
高级控制定时器	TMR1 TMR8	16	向上 向下 向上/向下	支持	支持	支持	支持	支持	支持	定时器同步 /ADC/DAC
	TMR2 TMR5	16/32	向上 向下 向上/向下	支持	支持	不支持	不支持	支持	支持	定时器同步 /ADC/DAC
	TMR3 TMR4	16	向上 向下 向上/向下	支持	支持	不支持	不支持	支持	支持	定时器同步 /ADC/DAC
通用定时器	TMR9 TMR12	16	向上 向下 向上/向下	支持	支持	支持	支持	不支持	不支持	定时器同步
	TMR10 TMR11 TMR13 TMR14	16	向上 向下 向上/向下	支持	支持	支持	支持	不支持	不支持	定时器
基本定时器	TMR6 TMR7	16	向上	不支持	不支持	不支持	不支持	不支持	不支持	ADC/DAC

15.1 基本定时器（TMR6和TMR7）

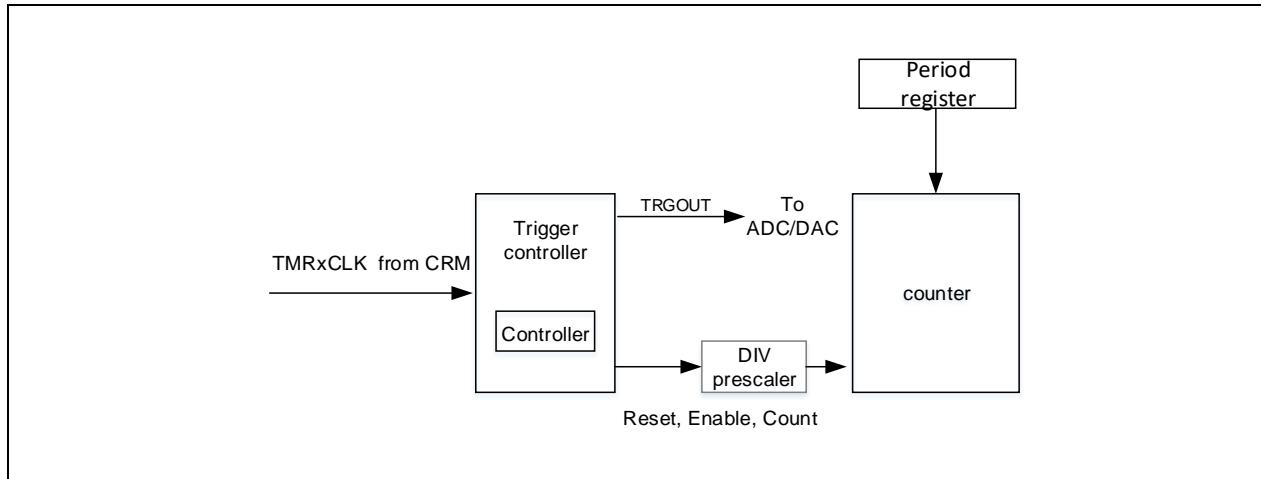
15.1.1 TMR6和TMR7简介

基本定时器（TMR6 和 TMR7）包含一个 16 位向上计数器以及对应的控制逻辑，没有外部 I/O 接入。可用于简单的定时功能、为 DAC 提供时钟以及作为 ADC 的外部触发源。

15.1.2 TMR6和TMR7主要功能

- 16 位向上计数器，可自动装载
- 16 位预分频器，用于对 TMR_CLK 时钟分频，分频系数为 1~65536 之间的任意数值
- 触发 DAC 的同步电路
- 作为 ADC 的外部触发源

图 15-1 基本定时器框图

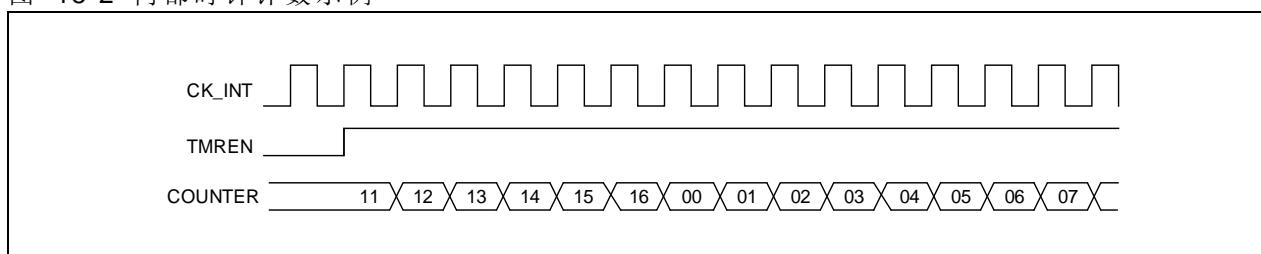


15.1.3 TMR6和TMR7功能描述

15.1.3.1 计数时钟

TMR6 和 TMR7 由内部时钟源（CK_INT）经由预分频器提供计数器计数。当 TMR 对应的 APB 时钟预分频系数是 1 时，CK_INT 频率等于 APB 时钟频率，否则 CK_INT 频率等于 APB 时钟频率的 2 倍。

图 15-2 内部时钟计数示例



15.1.3.2 计数模式

基本定时器仅提供向上计数模式。其内部拥有一个 16 位计数器。

TMRx_PR 寄存器用于设置计数器计数周期。默认 TMRx_PR 寄存器值会立即传入它的影子寄存器；当开启周期缓冲功能后（TMRx_CTRL1 寄存器 PRBEN 置 1），TMRx_PR 寄存器值在溢出事件发生时传入它的影子寄存器。

TMRx_DIV 寄存器用于设置计数器计数频率，每（DIV[15:0]+1）个计数时钟周期，计数器计数一次。TMRx_DIV 寄存器值在溢出事件时更新至它的影子寄存器。

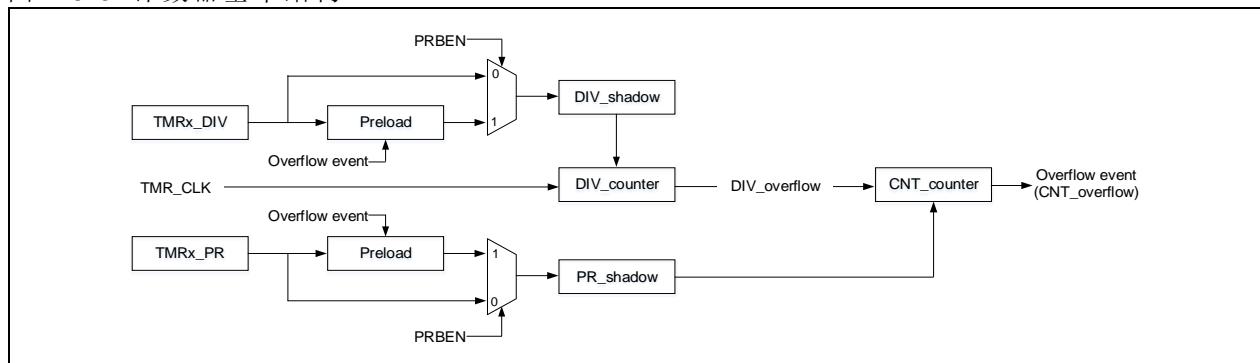
读取 TMRx_CNT 寄存器会返回当前计数器计数值，写入 TMRx_CNT 寄存器会更新计数器当前计数值为写入值。

默认允许产生溢出事件，设置 TMRx_CTRL1 寄存器 OVFEN=1 将禁止溢出事件产生。TMRx_CTRL1 寄存器 OVFS 用于选择溢出事件来源，默认计数器上溢、置位 OVFSWTR 产生溢出事件。置位 OVFS 后，只有计数器上溢产生溢出事件。

TMREN 位置 1 将使能定时器计数，由于同步逻辑，实际驱动计数器的使能信号 TMR_EN 相对于 TMREN

延迟一个时钟周期。

图 15-3 计数器基本结构



向上计数模式

在向上计数模式中，计数值达到 $TMRx_PR$ 值时，重新从 0 向上计数，计数器上溢并产生溢出事件，同时 OVFIF 位置 1。若禁止产生溢出事件，计数器溢出后不再重载预分频值和周期值，否则预分频值和周期值在溢出事件后更新。

图 15-4 PRBEN=0 时的溢出事件

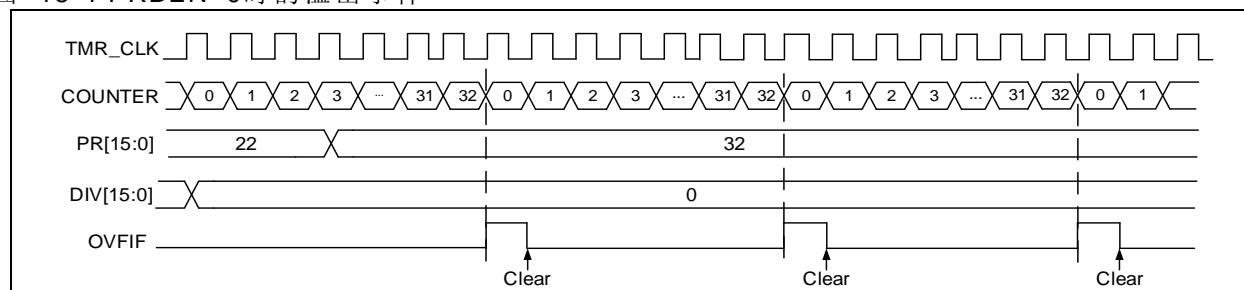


图 15-5 PRBEN=1 时的溢出事件

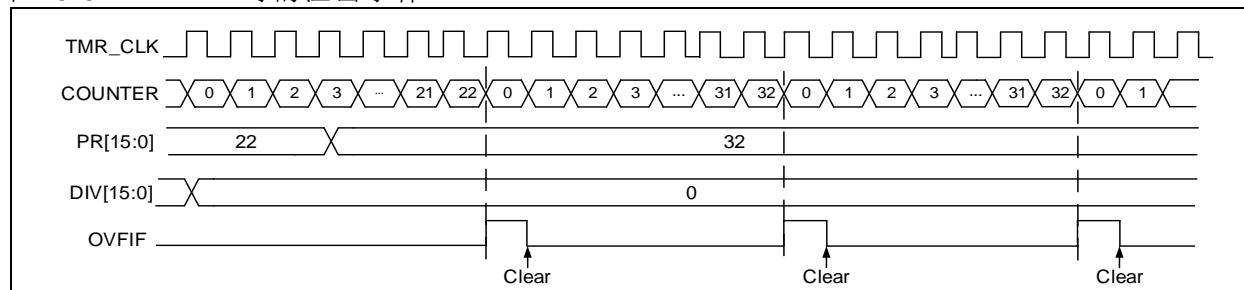
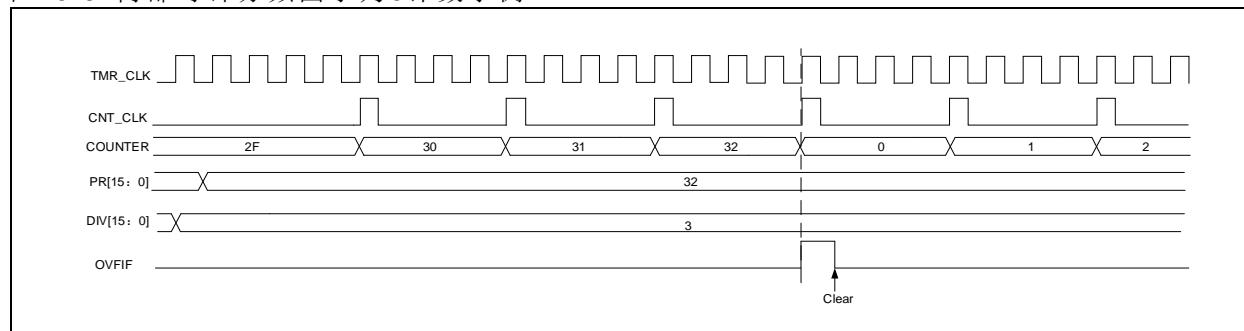


图 15-6 内部时钟分频因子为 3 计数示例



15.1.3.3 调试模式

当微控制器进入调试模式（Cortex®-M4F 核心停止）时，将 DEBUG 模块中的 $TMRx_PAUSE$ 置 1，可以使 $TMRx$ 计数器暂停计数。

15.1.4 TMR6和TMR7寄存器

必须以字（32位）的方式操作这些外设寄存器。

表 15-2 TMR6和TMR7寄存器和复位值

寄存器简称	基址偏移量	复位值
TMRx_CTRL1	0x00	0x0000 0000
TMRx_CTRL2	0x04	0x0000 0000
TMRx_IDEN	0x0C	0x0000 0000
TMRxISTS	0x10	0x0000 0000
TMRx_SWEVT	0x14	0x0000 0000
TMRxCVAL	0x24	0x0000 0000
TMRxDIV	0x28	0x0000 0000
TMRx_PR	0x2C	0x0000 FFFF

15.1.4.1 TMR6和TMR7控制寄存器1 (TMRx_CTRL1)

域	简称	复位值	类型	功能
位 31: 8	保留	0x00 0000	resd	保持默认值。
位 7	PRBEN	0x0	rw	周期缓冲使能 (Period buffer enable) 0: 缓冲关闭; 1: 缓冲开启。
位 6: 4	保留	0x0	resd	保持默认值。
位 3	OCMEN	0x0	rw	单周期使能 (One cycle mode enable) 该功能用于选择溢出事件后，计数器是否停止。 0: 关闭; 1: 开启。
位 2	OVFS	0x0	rw	溢出事件源选择 (Overflow event source) 配置溢出事件或 DMA 请求来源。 0: 来源于计数器溢出或设置 OVFSWTR 位产生的溢出事件; 1: 只能来源于计数器溢出。
位 1	OVFEN	0x0	rw	溢出事件使能 (Overflow event enable) 该位用于允许或禁止溢出事件 (OEV) 产生。 0: 允许溢出事件产生，溢出事件可以由下列事件产生: - 计数器溢出 - 将 OVFSWTR 位置 1 1: 禁止溢出事件产生。 如果将 OVFSWTR 位置 1 产生了一个软件复位，则计数器和预分频器将被重新初始化。 注：该位由软件置 1 和清 0。
位 0	TMREN	0x0	rw	使能定时器 (TMR enable) 0: 关闭; 1: 开启。

15.1.4.2 TMR6和TMR7控制寄存器2 (TMRx_CTRL2)

域	简称	复位值	类型	功能
位 31: 7	保留	0x000 0000	resd	保持默认值。
位 6: 4	PTOS	0x0	rw	主定时器输出信号选择 (Primary TMR output selection) TMRx 输出到次定时器的信号选择: 000: 软件溢出; 001: 使能; 010: 溢出;
位 3: 0	保留	0x0	resd	保持默认值。

15.1.4.3 TMR6和TMR7 DMA/中断使能寄存器 (TMRx_IDEN)

域	简称	复位值	类型	功能
位 31: 9	保留	0x00 0000	resd	保持默认值。
位 8	OVFDEN	0x0	rw	溢出事件的 DMA 请求使能 (overflow event DMA request enable) 0: 关闭; 1: 开启。
位 7: 1	保留	0x0	resd	保持默认值。
位 0	OVFIEN	0x0	rw	溢出中断使能 (overflow interrupt enable) 0: 关闭; 1: 开启。

15.1.4.4 TMR6和TMR7中断状态寄存器 (TMRxISTS)

域	简称	复位值	类型	功能
位 31: 1	保留	0x0000 0000	resd	保持默认值。
位 0	OVFIF	0x0	rw0c	溢出中断标记 (Overflow interrupt flag) 当溢出事件发生时由硬件置'1'，由软件清'0'。 0: 无溢出事件发生; 1: 发生溢出事件，若 TMRx_CTRL1 的 OVFEN=0、 OVFS=0 时： - 当 TMRx_SWEVT 寄存器的 OVFSWTR=1 时产生溢 出事件;

15.1.4.5 TMR6和TMR7软件事件寄存器 (TMRx_SWEVT)

域	简称	复位值	类型	功能
位 31: 1	保留	0x0000 0000	resd	保持默认值。
位 0	OVFSWTR	0x0	rw0c	软件触发溢出事件 (Overflow event triggered by software) 通过软件触发一个溢出事件。 0: 无作用; 1: 制造一个溢出事件。

15.1.4.6 TMR6和TMR7计数值寄存器 (TMRx_CVAL)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	CVAL	0x0000	rw	计数值 (Counter value)

15.1.4.7 TMR6和TMR7分频系数寄存器 (TMRx_DIV)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	DIV	0x0000	rw	分频系数 (Divider value) 计数器时钟频率 $f_{CK_CNT} = f_{TMR_CLK} / (DIV[15: 0]+1)$ 。 DIV 为溢出事件发生时写入的分频系数。

15.1.4.8 TMR6和TMR7周期寄存器 (TMRx_PR)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	PR	0xFFFF	rw	周期值 (Period value) 定时器计数的周期值。当周期值为 0 时，定时器不工作。

15.2 通用定时器（TMR2到TMR5）

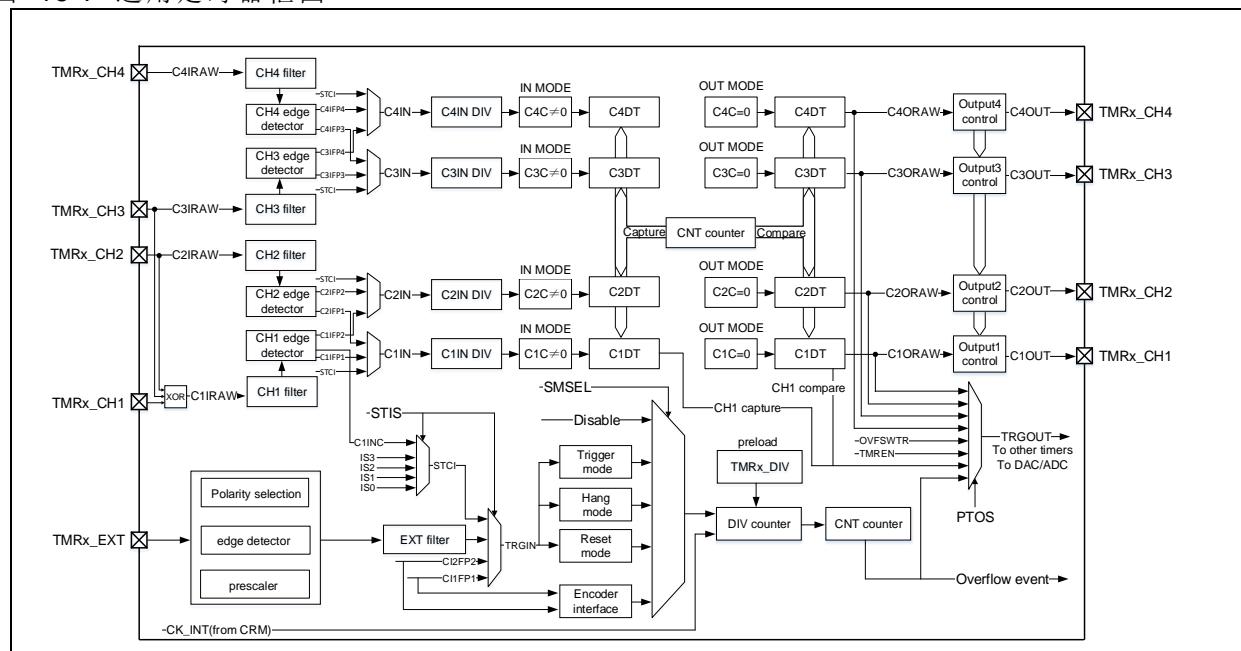
15.2.1 TMR2到TMR5简介

通用定时器 TMR2 到 TMR5 包含一个支持向上、向下、中央双向对齐计数的 16 位计数器、4 个捕获/比较寄存器、4 组独立的通道。可实现输入捕获、可编程 PWM 输出。

15.2.2 TMR2到TMR5主要功能

- 可选内部、外部、内部触发输入用作计数时钟
- 16 位支持向上、向下、双向、编码器模式的计数器（TMR2/5 可扩展至 32 位）
- 4 组独立通道，支持输入捕获、输出比较、PWM 生成、单周期模式
- 定时器之间可互联同步
- 支持溢出事件、触发事件、通道事件触发中断/DMA
- 支持 TMR Burst DMA 传输

图 15-7 通用定时器框图

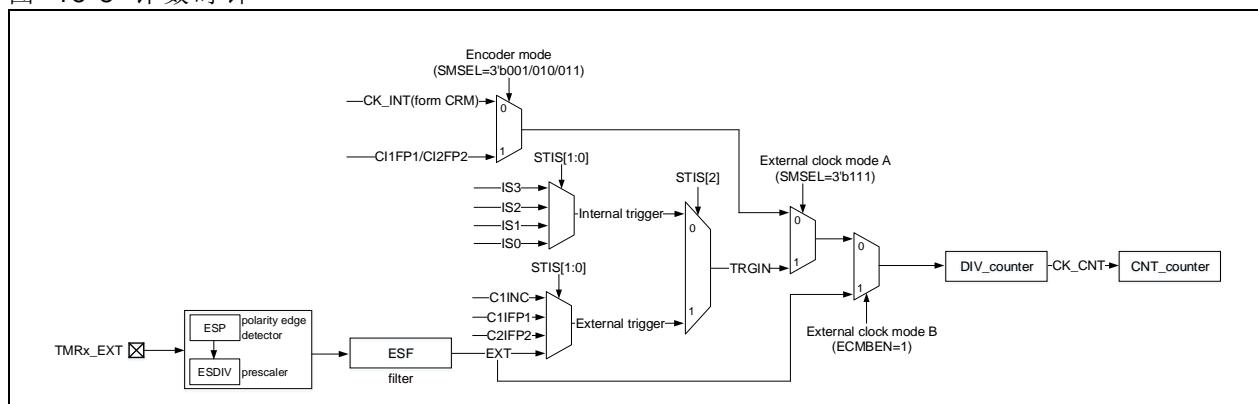


15.2.3 TMR2到TMR5功能描述

15.2.3.1 计数时钟

TMR2 至 5 计数时钟可从内部时钟（CK_INT）、外部时钟（外部时钟模式 A、B）、内部触发输入（ISx）这些时钟源提供。

图 15-8 计数时钟

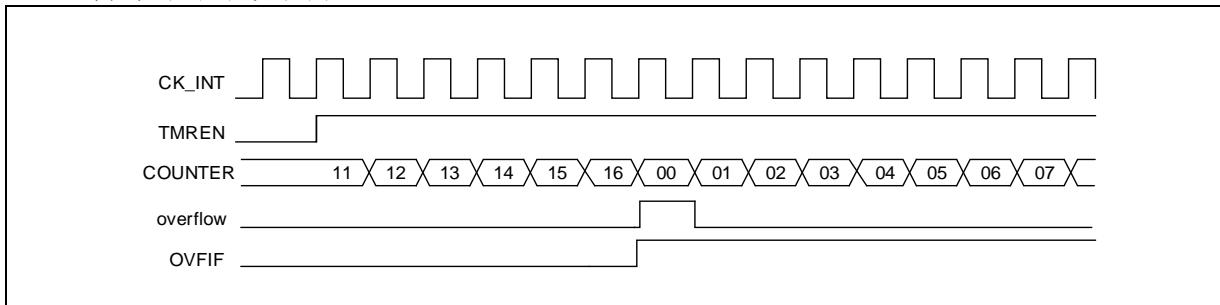


内部时钟 (CK_INT)

默认下使用 CK_INT 经由预分频器驱动计数器计数, 当 TMR 对应的 APB 时钟预分频系数是 1 时, CK_INT 频率等于 APB 时钟频率, 否则 CK_INT 频率等于 APB 时钟频率的 2 倍。相关配置流程如下:

- 配置 TMRx_CTRL1 寄存器 TWCMSEL[1:0], 选择计数模式, 若选择单向对齐计数模式, 还需配置 TMRx_CTRL1 寄存器 OWCDIR 选择计数方向。
- 配置 TMRx_DIV 寄存器, 设置计数器计数频率。
- 配置 TMRx_PR 寄存器, 设置计数器计数周期。
- 配置 TMRx_CTRL1 寄存器 TMREN, 使能计数器。

图 15-9 内部时钟计数示例



外部时钟 (TRGIN/EXT)

计数时钟可由两种外部时钟源提供, 分别为 TRGIN 和 EXT 信号。

当配置 TMRx_STCTRL 寄存器 SMSEL=3'b111 时, 外部时钟模式 A 被选中, 配置 TMRx_STCTRL 寄存器 STIS[2:0] 来选择外部时钟源 TRGIN 信号驱动计数器计数。外部时钟源 TRGIN 可选则 C1INC (STIS=3'b100, 通道 1 上升沿和下降沿信号)、C1IFP1 (STIS=3'b101, 通道 1 滤波且极性选择后信号)、C2IFP2 (STIS=3'b110, 通道 2 滤波且极性选择后信号) 和 EXT (STIS=3'b111, 外部输入经极性选择、分频和滤波后信号)。

当 TMRx_STCTRL 寄存器 ECMBEN=1 时, 外部时钟模式 B 被选中, 计数器由外部输入经极性选择、分频和滤波后 EXT 信号驱动计数。外部时钟模式 B 等效于外部时钟模式 A 选择 EXT 信号作为外部时钟源 TRGIN。

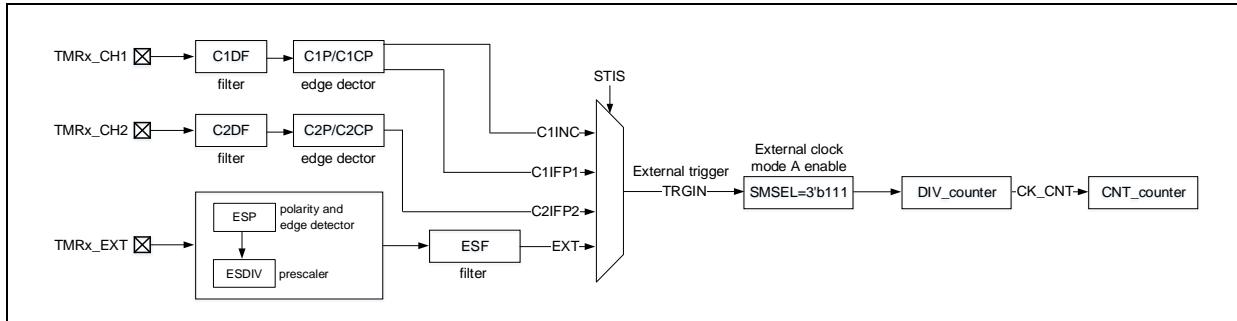
若要使用外部时钟模式 A, 可按如下步骤配置:

- 配置外部时钟源 TRGIN 参数。
 - 若选择 TRGIN 来源为 TMRx_CH1, 需配置通道 1 输入滤波 (TMRx_CM1 寄存器 C1DF[3:0]) 和通道 1 输入极性 (TMRx_CCTRL 寄存器 C1P/C1CP)。
 - 若选择 TRGIN 来源为 TMRx_CH2, 需配置通道 2 输入滤波 (TMRx_CM1 寄存器 C2DF[3:0]) 和通道 1 输入极性 (TMRx_CCTRL 寄存器 C2P/C2CP)。
 - 若选择 TRGIN 来源为 TMRx_EXT, 需配置外部信号极性 (TMRx_STCTRL 寄存器 ESP)、外部信号分频 (TMRx_STCTRL 寄存器 ESDIV[1:0]) 和外部信号滤波 (TMRx_STCTRL 寄存器 ESF[3:0])。
- 配置 TMRx_STCTRL 寄存器 STIS[2:0], 设置 TRGIN 信号来源。
- 配置 TMRx_STCTRL 寄存器 SMSEL=3'b111, 使能外部时钟模式 A。
- 配置 TMRx_DIV 寄存器 DIV[15:0], 设置计数器计数频率。
- 配置 TMRx_PR 寄存器 PR[15:0], 设置计数器计数周期。
- 配置 TMRx_CTRL1 寄存器 TMREN, 使能计数器。

若要使用外部时钟模式 B, 可按如下步骤配置:

- 配置 TMRx_STCTRL 寄存器 ESP, 设置外部信号极性。
- 配置 TMRx_STCTRL 寄存器 ESDIV[1:0], 设置外部信号分频。
- 配置 TMRx_STCTRL 寄存器 ESF[3:0], 设置外部信号滤波。
- 配置 TMRx_STCTRL 寄存器 ECMBEN, 使能外部时钟模式 B。
- 配置 TMRx_DIV 寄存器 DIV[15:0], 设置计数器计数频率。
- 配置 TMRx_PR 寄存器 PR[15:0], 设置计数器计数周期。
- 配置 TMRx_CTRL1 寄存器 TMREN, 使能计数器。

图 15-10 外部时钟模式A框图



注：由于同步逻辑，输入端信号与计数器实际时钟之间存在一定延时。

图 15-11 外部时钟模式A计数示例

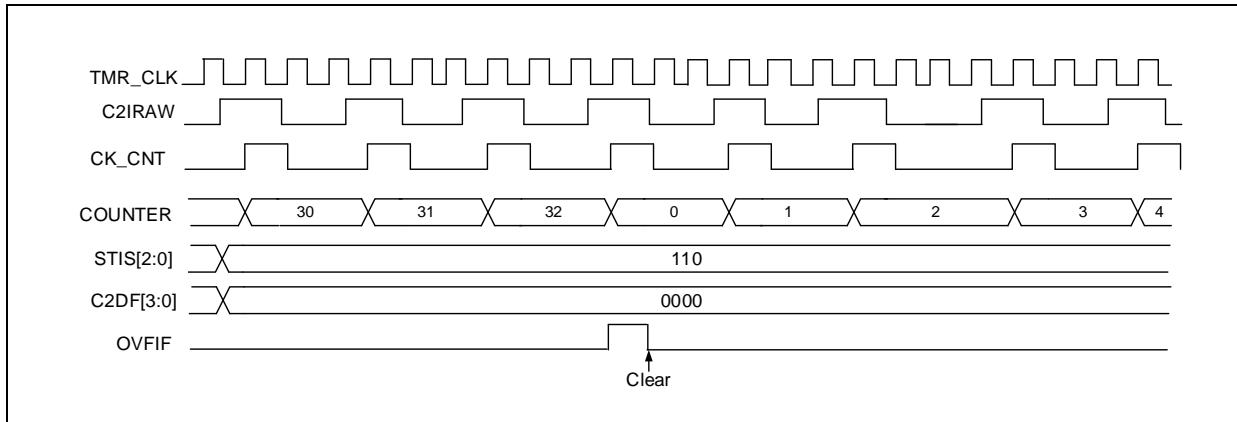
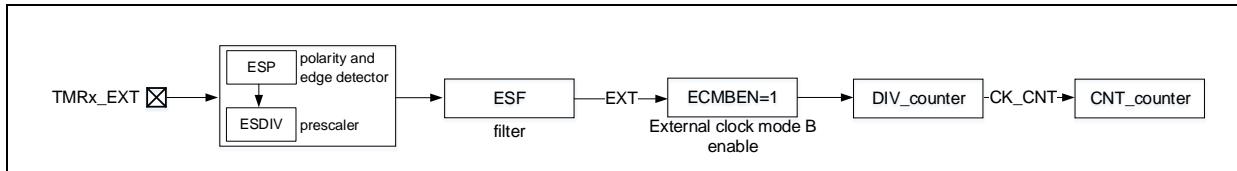
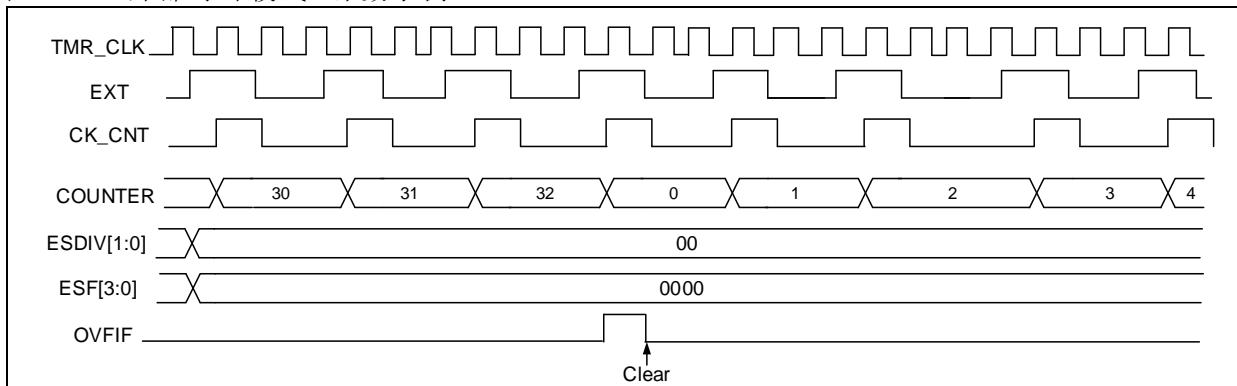


图 15-12 外部时钟模式B框图



注：由于同步逻辑。输入端 EXT 信号与计数器实际时钟之间存在一定延时。

图 15-13 外部时钟模式B计数示例



内部触发输入 (ISx)

定时器之间支持互连同步，因此一个定时器的 TMR_CLK 可由另一个定时器输出信号 TRGOUT 提供。配置 STIS[2: 0]选择内部触发信号驱动计数器计数。

TMR2 至 5 定时器内含一个 16 位预分频器，用于产生驱动计数器计数的时钟 CK_CNT，通过配置 TMRx_DIV 寄存器值，可灵活调整 CK_CNT 与 TMR_CLK 之间的分频关系。预分频值可在任何时刻修改，但只在下一个溢出事件发生时，新值才会生效。

内部触发输入配置流程如下：

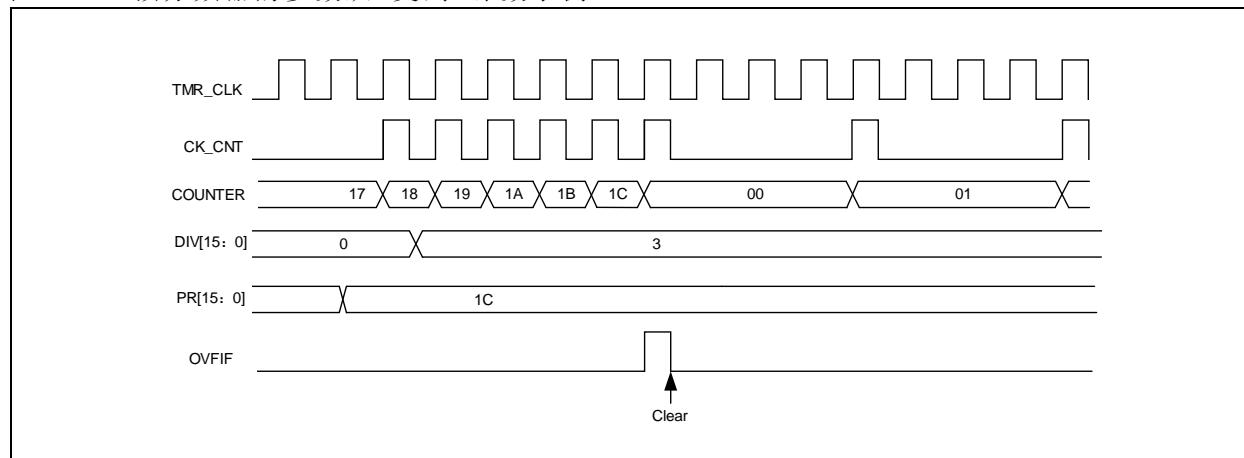
- 配置 TMRx_PR 寄存器，设置计数器计数周期。
- 配置 TMRx_DIV 寄存器，设置计数器计数频率。
- 配置 TMRx_CTRL1 寄存器 TWCMSEL[1:0]位，设置计数器计数模式。
- 配置 TMRx_STCTRL 寄存器 STIS[2:0]位范围为 3'b000~3'b011，选择内部触发。
- 配置 TMRx_STCTRL 寄存器 SMSEL[2:0]=3'b111，选择外部时钟模式 A。
- 配置 TMRx_CTRL1 寄存器 TMREN 位，使能 TMRx 计数。

表 15-3 TMRx 内部触发连接

次定时器	IS0 (STIS = 000)	IS1 (STIS = 001)	IS2 (STIS = 010)	IS3 (STIS = 011)
TMR2	TMR1	TMR8/USB_SOF/EMA_C_PTP	TMR3	TMR4
TMR3	TMR1	TMR2	TMR5	TMR4
TMR4	TMR1	TMR2	TMR3	TMR8
TMR5	TMR2	TMR3	TMR4	TMR8

注意：如果某个产品中没有相应的定时器，则对应的触发信号 ISx 也不存在。

图 15-14 预分频器的参数从0变到3计数示例



15.2.3.2 计数模式

TMR2 至 5 定时器提供了多种计数模式，用来满足不同的应用场景。其内部拥有一个支持 16 位向上、向下、中央双向对齐计数的计数器，TMR2/5 可通过将 TMRx_CTRL1 寄存器 PMEN 位置 1 扩展至 32 位。

TMRx_PR 寄存器用于设置计数器计数周期。默认 TMRx_PR 寄存器值会立即传入它的影子寄存器；当开启周期缓冲功能后 (TMRx_CTRL1 寄存器 PRBEN 置 1)，TMRx_PR 寄存器值在溢出事件发生时传入它的影子寄存器。

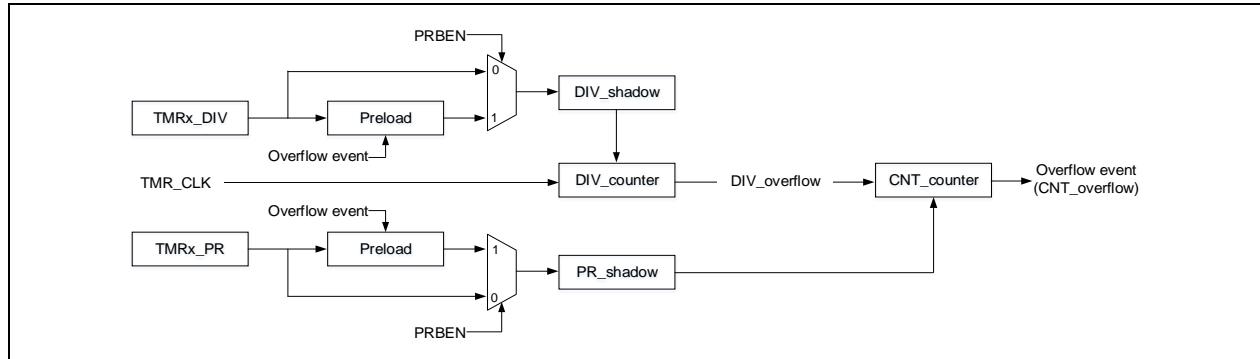
TMRx_DIV 寄存器用于设置计数器计数频率，每 (DIV[15:0]+1) 个计数时钟周期，计数器计数一次。TMRx_DIV 寄存器值在溢出事件时更新至它的影子寄存器。

读取 TMRx_CNT 寄存器会返回当前计数器计数值，写入 TMRx_CNT 寄存器会更新计数器当前计数值为写入值。

默认允许产生溢出事件，设置 TMRx_CTRL1 寄存器 OVFEN=1 将禁止溢出事件产生。TMRx_CTRL1 寄存器 OVFS 用于选择溢出事件来源，默认计数器上溢或下溢、置位 OVFSWTR、复位模式次定时器控制器产生的复位信号产生溢出事件。置位 OVFS 后，只有计数器上溢或下溢产生溢出事件。

TMREN 位置 1 将使能定时器计数，由于同步逻辑，实际驱动计数器的使能信号 TMR_EN 相对于 TMREN 延迟一个时钟周期。

图 15-15 计数器基本结构



向上计数模式

配置 TMRx_CTRL1 寄存器 TWCSEL[1:0]=2'b00, OWCDIR=1'b0 开启向上计数模式，计数值达到 TMRx_PR 值时，重新从 0 向上计数，计数器上溢并产生溢出事件，同时 OVFIF 位置 1。若禁止产生溢出事件，计数器溢出后不再重载预分频值和周期值，否则预分频值和周期值在溢出事件后更新。

图 15-16 PRBEN=0 时的溢出事件

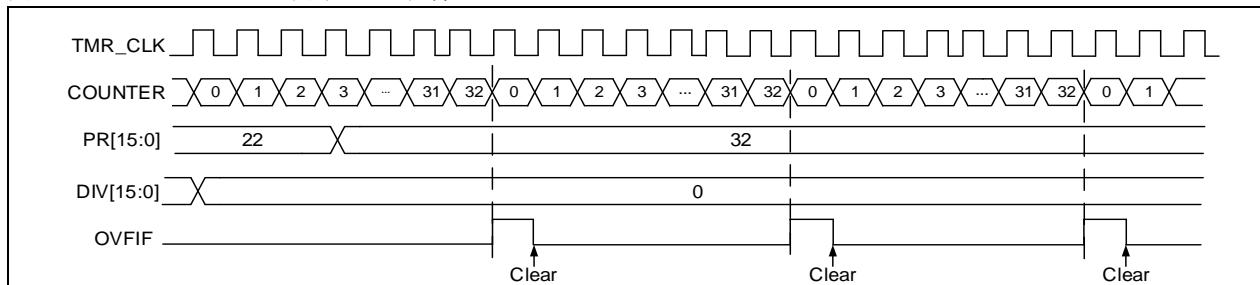
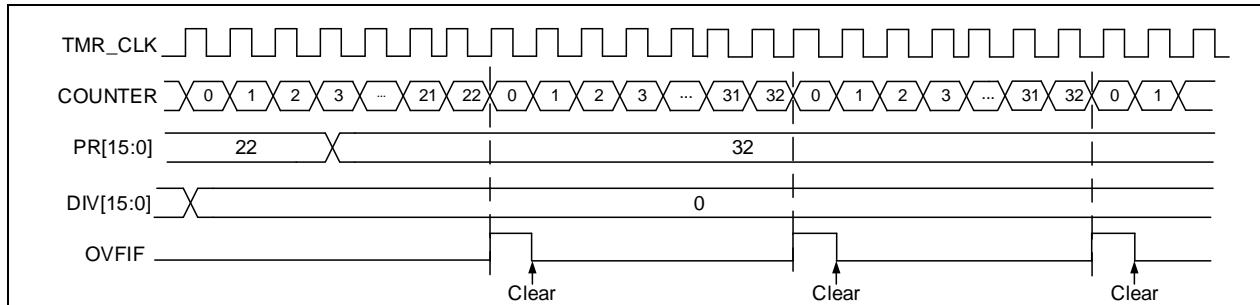


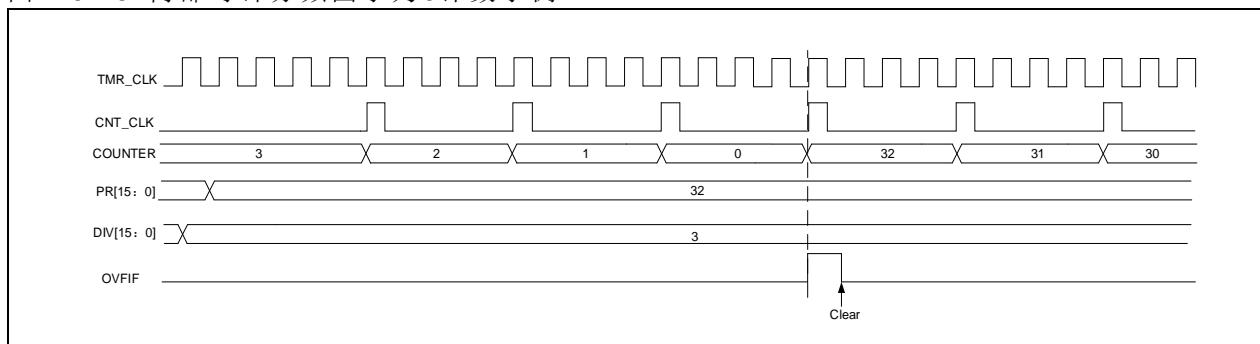
图 15-17 PRBEN=1 时的溢出事件



向下计数模式

配置 TMRx_CTRL1 寄存器 TWCSEL[1:0]=2'b00, OWCDIR=1'b1 开启向下计数模式，计数值达到 0 值并重新从 TMRx_PR 向上下数时，计数器下溢并产生溢出事件。

图 15-18 内部时钟分频因子为 3 计数示例



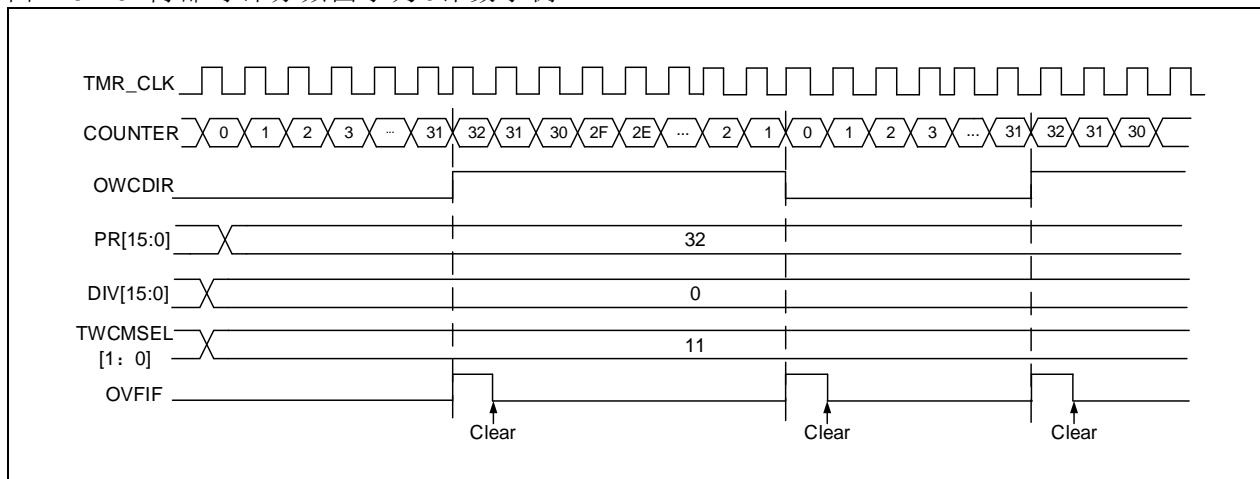
中央双向对齐计数模式

配置 TMRx_CTRL1 寄存器 TWCSEL[1:0]≠2' b00 开启中央双向对齐计数模式，中央双向对齐计数模式下计数器交替向上、向下计数。计数值从 TMRx_PR 值向下计数到 1 值，产生下溢事件，然后从 0 开始向上计数；向上计数到 TMRx_PR 值-1，产生上溢事件，之后从 TMRx_PR 值向下计数。计数器计数方向由计数器方向控制位（OWCDIR）实时查看。

TMRx_CTRL1 寄存器 TWCSEL[1:0]位还用于选择中央双向对齐计数模式下 CxIF 标志置起方式，中央双向对齐计数模式 1（TWCSEL[1:0]=2'b01）仅允许 CxIF 标志位在计数器向下计数时置起；双向对齐计数模式 2（TWCSEL[1:0]=2'b10）仅允许 CxIF 标志位在计数器向上计数时置起；双向对齐计数模式 3（TWCSEL[1:0]=2'b11）允许 CxIF 标志位在计数器向上和向下计数时置起。

注意： 中央双向对齐计数模式下，OWCDIR 位为只读位。

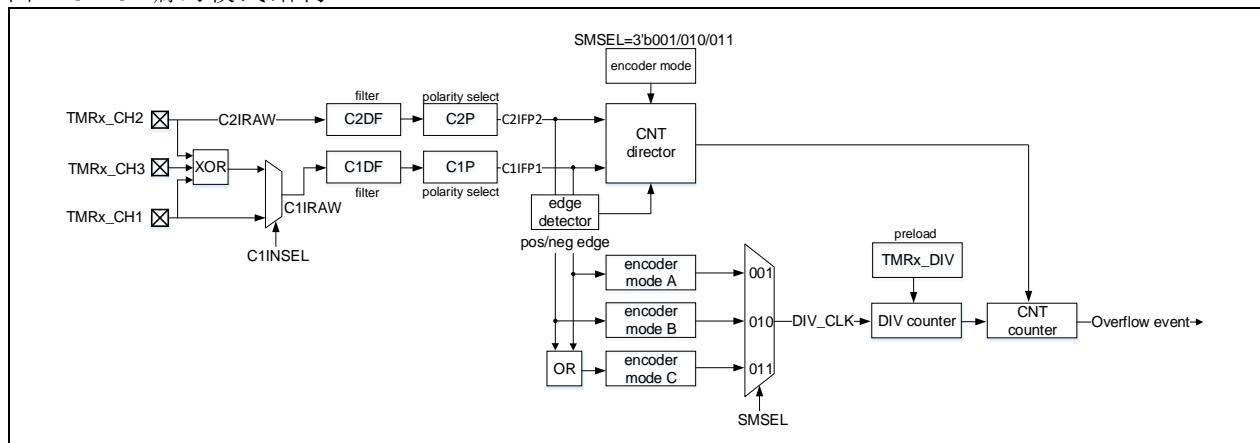
图 15-19 内部时钟分频因子为0计数示例



编码器模式

编码器模式下需提供两组输入信号 TMRx_CH1 和 TMRx_CH2，根据一组输入信号电平值，计数器在另一组输入信号边沿向上或向下计数。计数方向由 OWCDIR 值指示。

图 15-20 编码模式结构



编码器模式 A: SMSEL=3'b001，计数器在 C1IFP1 边沿计数（上升沿和下降沿），计数方向由 C1IFP1 边沿方向和 C2IFP2 电平高低共同决定。

编码器模式 B: SMSEL=3'b010，计数器在 C2IFP2 边沿计数（上升沿和下降沿），计数方向由 C2IFP2 边沿方向和 C1IFP1 电平高低共同决定。

编码器模式 C: SMSEL=3'b011，计数器在 C1IFP1 和 C2IFP2 边沿计数（上升沿和下降沿），计数方向由 C1IFP1 边沿方向和 C2IFP2 电平高低、C2IFP2 边沿方向和 C1IFP1 电平高低共同决定共同决定。

若要使用编码器模式可按下面步骤配置：

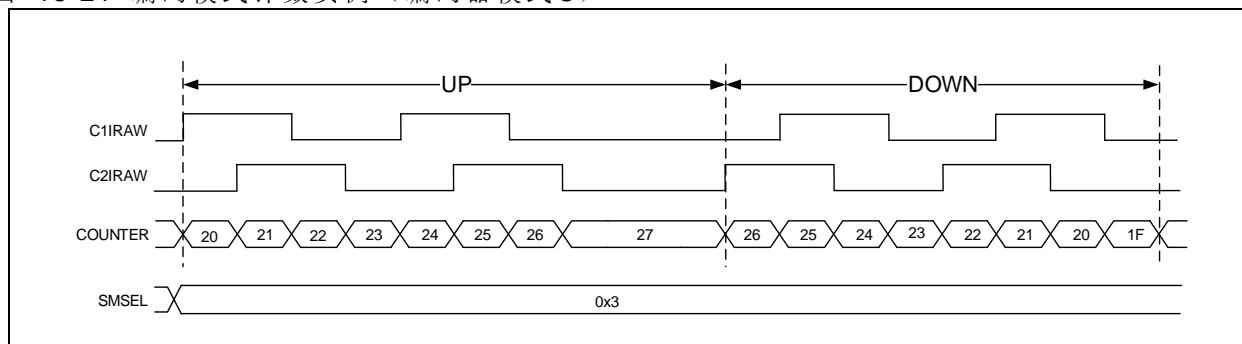
- 配置 `TMRx_CM1` 寄存器 `C1DF[3:0]`, 设置通道 1 输入信号滤波；配置 `TMRx_CCTRL` 寄存器 `C1P`, 设置通道 1 输入信号有效电平。
- 配置 `TMRx_CM1` 寄存器 `C2DF[3:0]`, 设置通道 2 输入信号滤波；配置 `TMRx_CCTRL` 寄存器 `C2P`, 设置通道 2 输入信号有效电平。
- 配置 `TMRx_CM1` 寄存器 `C1C[1:0]`, 设置通道 1 为输入模式；配置 `TMRx_CM1` 寄存器 `C2C[1:0]`, 设置通道 2 为输入模式；
- 配置 `TMRx_STCTRL` 寄存器 `SMSEL[2:0]`, 选择编码器模式 A (`SMSEL=3'b001`)、编码器模式 B (`SMSEL=3'b010`) 或编码器模式 C (`SMSEL=3'b011`)。
- 配置 `TMRx_PR` 寄存器 `PR[15:0]`, 设置计数器计数周期。
- 配置 `TMRx_DIV` 寄存器 `DIV[15:0]`, 设置计数器计数频率。
- 配置 `TMRx_CH1` 和 `TMRx_CH2` 对应 IO 为复用模式。
- 配置 `TMRx_CTRL1` 寄存器 `TMREN`, 使能计数器。

编码模式下计数器计数方向如下表所示：

表 15-4 计数方向与编码器信号的关系

计数边沿	计数边沿相对信号的电平 (<code>C1IFP1</code> 边沿对应 <code>C2IFP2</code> 电平, <code>C2IFP2</code> 边沿对应 <code>C1IFP1</code> 电平)	<code>C1IFP1</code> 边沿方向		<code>C2IFP2</code> 边沿方向	
		上升	下降	上升	下降
<code>C1IFP1</code>	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
<code>C2IFP2</code>	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
<code>C1IFP1</code> 和 <code>C2IFP2</code>	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

图 15-21 编码模式计数实例（编码器模式 C）



15.2.3.3 TMR输入部分

TMR2 至 5 拥有 4 个独立通道，每个通道可配置为输入或输出，当配置位输入时，每个通道输入信号依次经过以下处理：

- `TMRx_CHx` 经过预处理输出 `CxIRAW`。配置 `C1INSEL` 位，选择 `C1IRAW` 来源是 `TMRx_CH1` 或是 `TMRx_CH1`、`TMRx_CH2`、`TMRx_CH3` 异或。`C2IRAW`、`C3IRAW`、`C4IRAW` 来源是 `TMRx_CH2`、`TMRx_CH3`、`TMRx_CH4`。
- `CxIRAW` 输入数字滤波器，输出滤波后信号 `CxIF`。数字滤波器通过 `TMRx_CM1/CM2` 寄存器 `CxDf` 位配置采样频率和次数。
- `CxIF` 输入边沿检测器，输出边沿选择后信号 `CxIFPx`。边沿选择由 `TMRx_CCTRL` 寄存器 `CxP` 和 `CxCP` 位共同控制，可选择输入上升沿、下降沿或双边沿有效。
- `CxIFPx` 输入捕获信号选择器，输出选择后信号 `CxIN`。捕获信号选择器由 `TMRx_CM1/CM2` 寄存器 `CxC` 控制，可选择 `CxIN` 来源为 `CxIFPx`、`CyIFPx`、`STCI`。其中

CyIFPx ($x \neq y$) 是来自通道 y 的 CyIFPy 信号; STCI 来自次定时器控制器, 由 STIS 位选择来源。

- CxIN 经由输入通道分频器, 输出分频后信号 CxIPS。分频系数由 TMRx_CM1/CM2 寄存器 CxIDIV 位配置为不分频、2 分频、4 分频或 8 分频。

图 15-22 输入/输出通道 1 的主电路

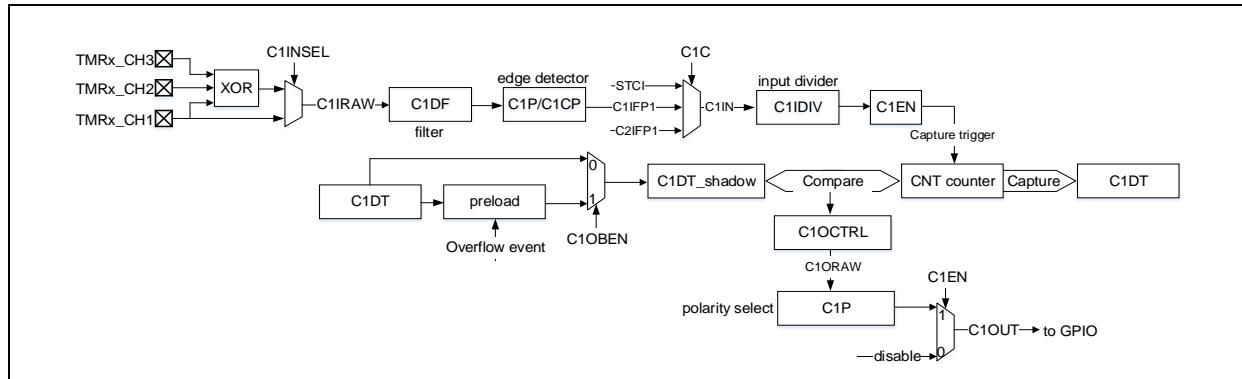
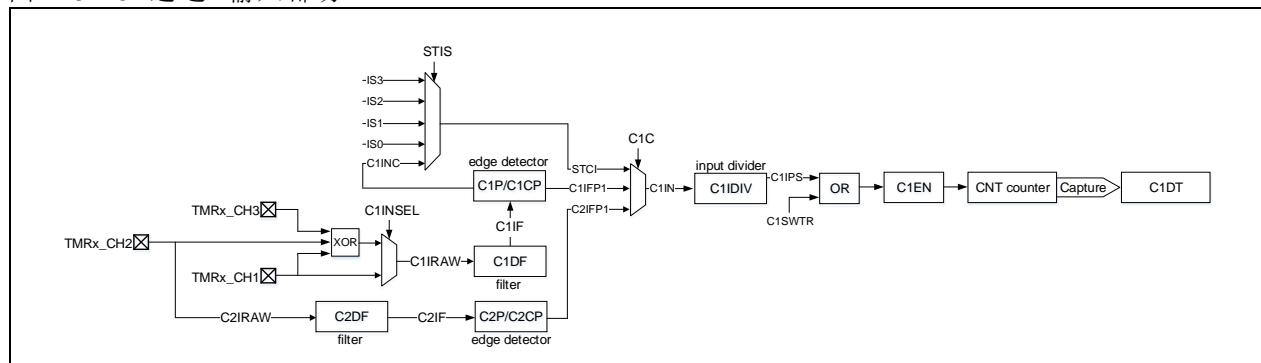


图 15-23 通道 1 输入部分



输入模式

此模式下, 当选中的触发信号被检测到, 通道寄存器 (TMRx_CxDT) 记录当前计数器计数值, 并将捕获比较中断标志位 (TMRx_ISTS 寄存器 CxIF) 置 1, 若已使能通道中断 (TMRx_IDEN 寄存器 CxIEN)、通道 DMA 请求 (TMRx_IDEN 寄存器 CxDEN) 则产生相应的中断和 DMA 请求。若在 CxIF 置 1 后检测到触发信号, 将产生捕获溢出事件, TMRx_CxDT 会使用当前计数器计数值覆盖之前记录的计数器计数值, 同时通道再捕获标志位 (TMRx_ISTS 寄存器 CxRF) 置 1。

若要捕获 C1IN 输入的上升沿, 可按如下进行配置:

- 将通道模式寄存器 1 (TMRx_CM1) 中的 C1C 位配置为 2'b01, 选择 C1IN 作为通道 1 输入。
- 配置 C1IN 信号滤波器带宽 (CxDF[3: 0])。
- 配置 C1IN 通道的有效沿, 在 TMRx_CCTRL 寄存器中写入 C1P=1'b0 (上升沿)。
- 配置 C1IN 信号捕获分频 (C1DIV[1: 0])。
- 使能通道 1 输入捕获 (C1EN=1)。
- 根据需要设置 TMRx_IDEN 寄存器中的 C1IEN 为、TMRx_IDEN 寄存器中的 C1DEN 位, 选择中断请求或 DMA 请求。

多输入异或

通道 1 的输入端可选择 TMRx_CH1、TMRx_CH2 和 TMRx_CH3 经异或逻辑后输入。将 TMRx_CTRL2 寄存器中的 C1INSEL 位置 1 可开启此功能。

多输入异或功能可用于连接霍尔传感器, 例如, 将异或输入的三个输入端分别连接到三个霍尔传感器, 通过分析三路霍尔传感器信号可计算出转子的位置和速度。

输入选择

TMR2 和 TMR5 可通过 TMRx_RMP 寄存器配置内部连接或通道输入的映射关系。TMR2 的内部连接信号 IS1 可映射为 TMR8_TRGOUT、EMAC_PTP、USB_SOF 其中之一；TMR5 通道 4 的输入可映射为 GPIO、LICK、LEXT、ERTC 唤醒中断其中之一。

PWM 输入

PWM 输入模式适用于通道 1 和 2，要使用此模式，需要将 C1IN 和 C2IN 映射到同一 TMRx_CHx，并且通道 1 或 2 的 CxIFPx 配置成触发次定时器控制器复位。

PWM 输入模式可用于测量输入信号的周期和占空比，如需测量通道 1 输入信号的周期和占空比，操作步骤如下：

- 配置 TMRx_CM1 寄存器 C1C=2'b01，选择 C1IN 为 C1IFP1。
- 配置 TMRx_CCTRL 寄存器 C1P=1'b0，选择 C1IFP1 上升沿有效。
- 配置 TMRx_CM1 寄存器 C2C=2'b10，选择 C2IN 为 C1IFP2。
- 配置 TMRx_CCTRL 寄存器 C2P=1'b1，选择 C1IFP2 下降沿有效。
- 配置 TMRx_STCTRL 寄存器 STIS=3'b101，选择次定时器触发信号为 C1IFP1。
- 配置 TMRx_STCTRL 寄存器 SMSEL=3'b110，选择次定时器模式为复位模式。
- 配置 TMRx_CCTRL 寄存器 C1EN=1'b1，C2EN=1'b1。使能通道 1 和通道 2 输入捕获。

上述配置下，通道 1 输入信号的上升沿会触发捕获并将捕获值存储到 C1DT 寄存器，同时通道 1 输入信号上升沿复位计数器。通道 1 输入信号下降沿触发捕获并将捕获值存储到 C2DT 寄存器。通道 1 输入信号的周期可通过 C1DT 计算，占空比可通过 C2DT 计算。

图 15-24 PWM 输入模式配置实例

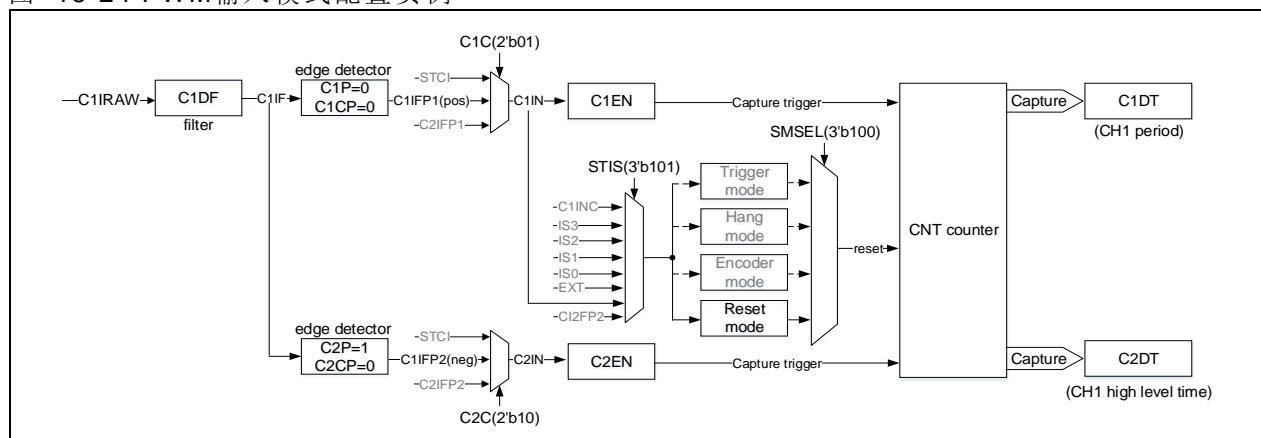
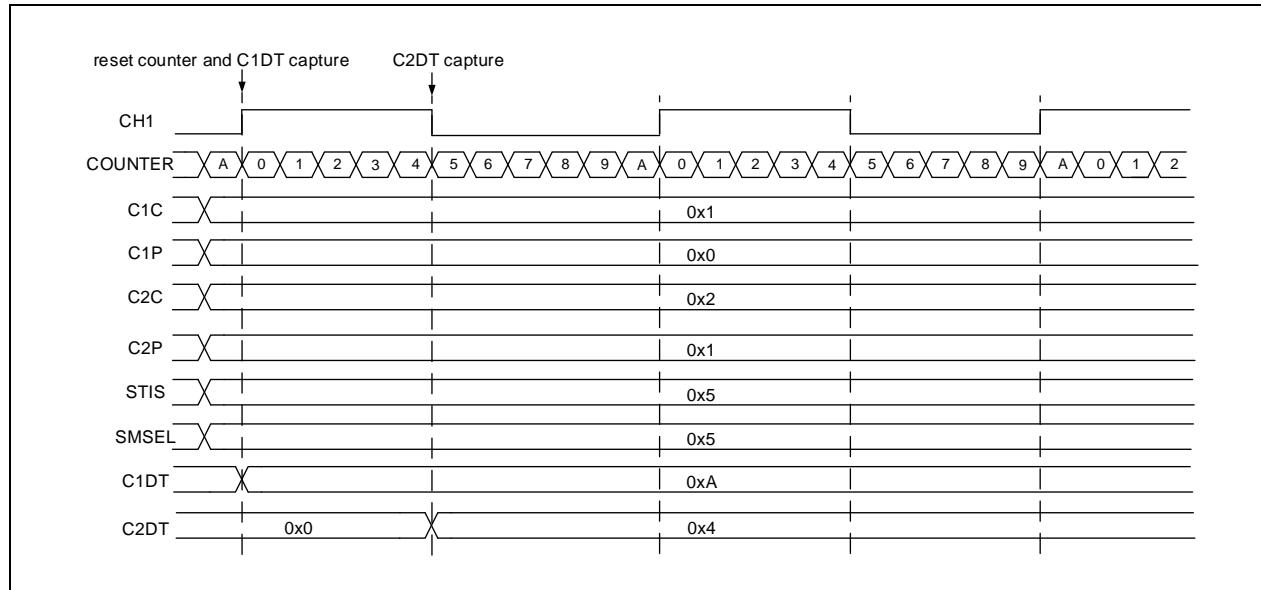


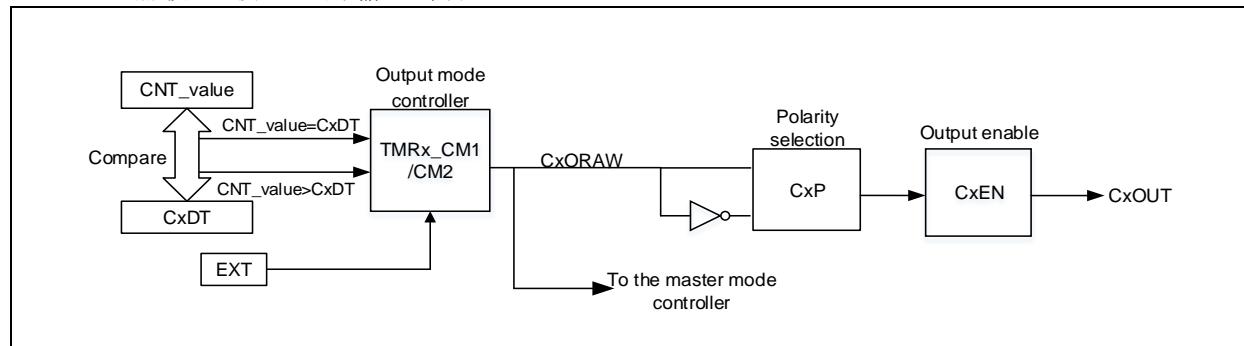
图 15-25 PWM 输入模式



15.2.3.4 TMR输出部分

TMR 的输出部分由比较器和输出控制构成，用于编程输出信号的周期、占空比、极性。

图 15-26 捕获/比较通道的输出部分（通道 1 至 4）



输出模式

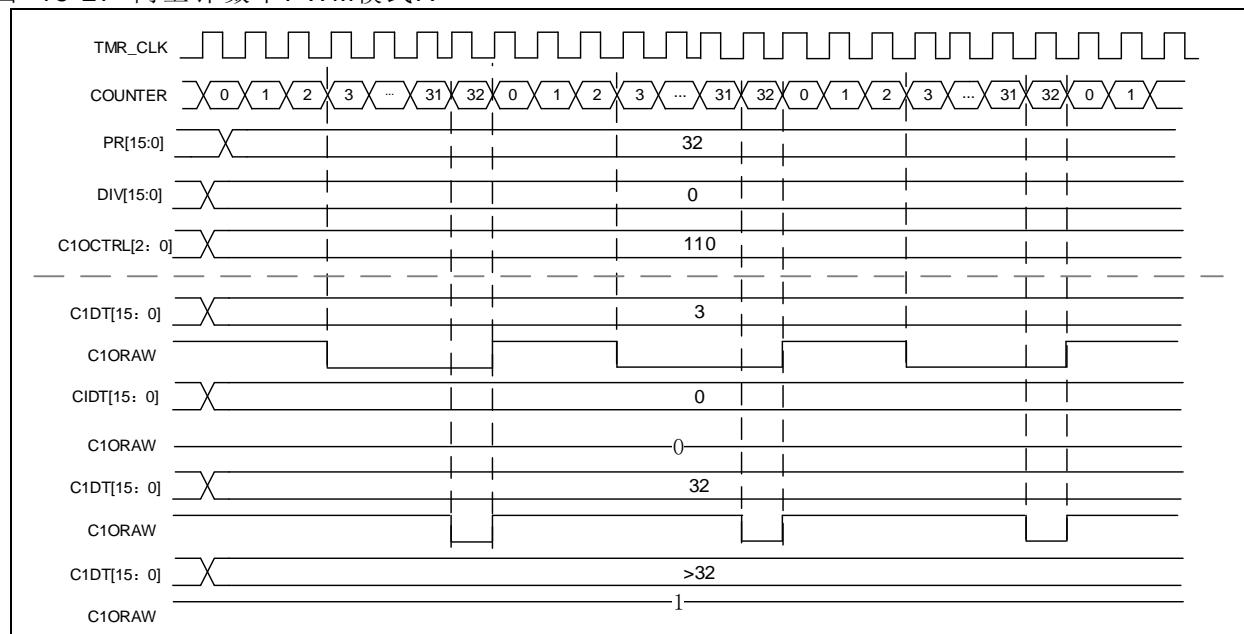
配置 $CxOCTRL[1:0] \neq 2'$ b00 将通道配置为输出可实现多种输出模式，此时，计数器计数值将与 $CxDT$ 寄存器值比较，并根据 $CxOCTRL[2:0]$ 位配置的输出模式，产生中间信号 $CxORAW$ ，再经过输出控制逻辑处理后输送至 IO。输出信号的周期由 $TMRx_PR$ 寄存器值配置，占空比则由 $CxDT$ 寄存器值配置。

输出比较模式有以下子类：

PWM 模式 A: $CxOCTRL=3'b110$ 时，开启 PWM 模式 A。向上计数时， $TMRx_C1DT > TMRx_CVAL$ 时 $C1ORAW$ 输出高电平，否则为低电平；向下计数时， $TMRx_C1DT < TMRx_CVAL$ 时 $C1ORAW$ 输出低电平，否则为高电平。图 15-27 展示了计数器向上计数与 PWM 模式 A 配合的例子， $PR=0x32$ ， $CxDT$ 配置为不同的值时输出信号的翻转情况。若要使用 PWM 模式 A，可按如下方式配置。

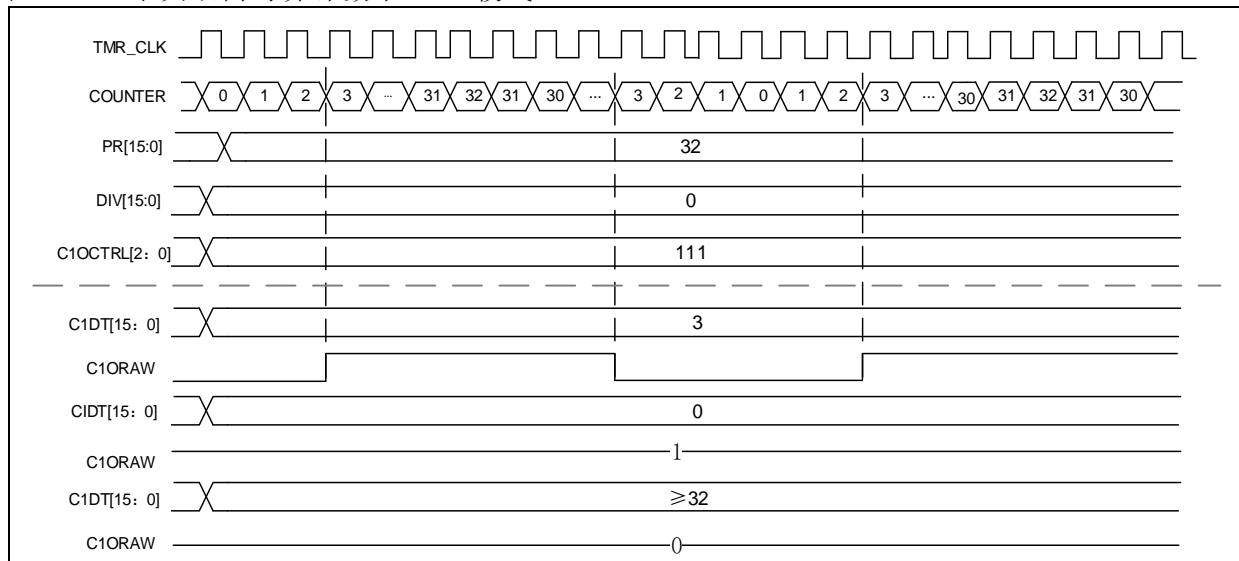
- 配置 $TMRx_PR$ 寄存器，设置 PWM 周期。
- 配置 $TMRx_CxDT$ 寄存器，设置 PWM 占空比。
- 配置 $TMRx_CM1/CM2$ 寄存器 $CxOCTRL$ 位为 3'b110，设置输出模式为 PWM 模式 A。
- 配置 $TMRx_DIV$ 寄存器，设置计数器计数频率。
- 配置 $TMRx_CTRL1$ 寄存器 $TWCMSEL[1:0]$ 位，设置计数器计数模式。
- 配置 $TMRx_CCTRL$ 寄存器 CxP 位，设置输出极性。
- 配置 $TMRx_CCTRL$ 寄存器 $CxEN$ 位，使能通道输出。
- 配置 TMR 输出通道对应 GPIO 为对应的复用模式。
- 配置 $TMRx_CTRL1$ 寄存器 $TMREN$ 位，使能 $TMRx$ 计数。

图 15-27 向上计数下 PWM 模式 A



PWM 模式 B: CxOCTRL=3'b111 时，开启 PWM 模式 B。向上计数时，TMRx_C1DT>TMRx_CVAL 时 C1ORAW 输出低电平，否则为高电平；向下计数时，TMRx_C1DT<TMRx_CVAL 时 C1ORAW 输出高电平，否则为低电平。图 15-28 展示了计数器中央双向对齐计数与 PWM 模式 B 配合的例子，PR=0x32，CxDT 配置为不同的值时输出信号的翻转情况。

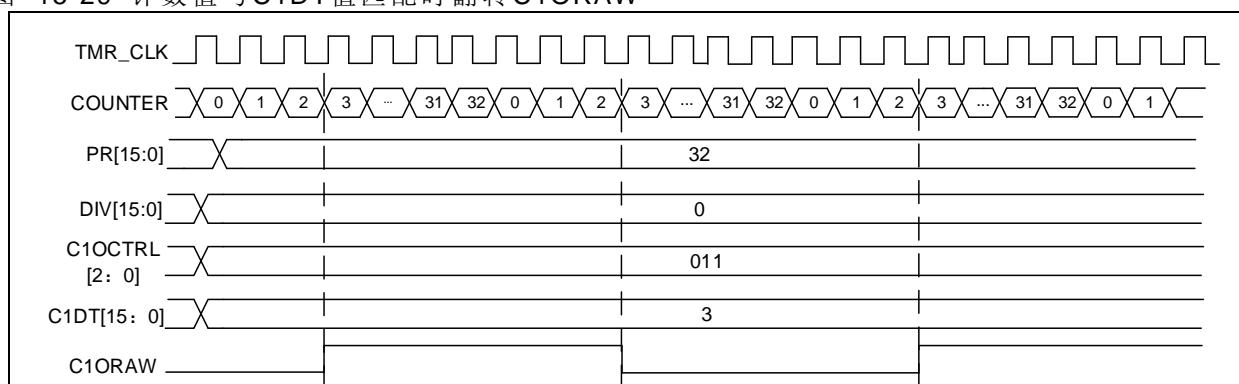
图 15-28 中央双向对齐计数下 PWM 模式 B



强制输出模式: CxOCTRL=3'b100/101 时，开启强制输出模式。此时，CxORAW 信号的电平被强制输出为配置的电平，而与计数值无关。虽然输出信号不依赖于比较结果，但通道标志位和 DMA 请求仍依赖于比较结果。

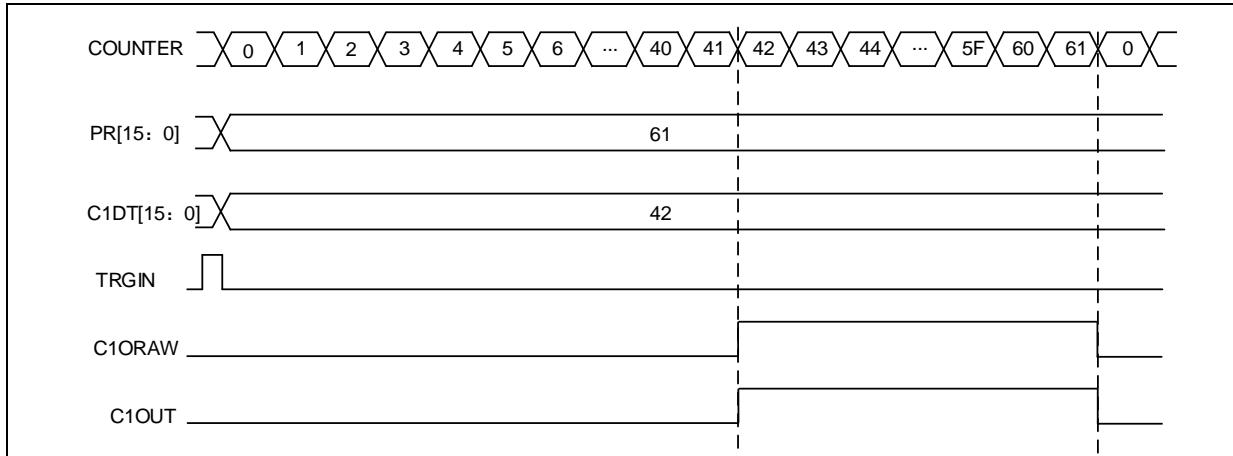
输出比较模式: CxOCTRL=3'b001/010/011 时，开启输出比较模式。此时，当计数值与 CxDT 值匹配时，CxORAW 强制输出高电平（CxOCTRL=3'b001）、低电平（CxOCTRL=3'b010）或进行电平翻转（CxOCTRL=3'b011）。图 15-29 展示了输出比较模式（翻转）的例子，C1DT=0x3，当计数值等于 0x3 时，输出电平 C1OUT 被翻转。

图 15-29 计数值与 C1DT 值匹配时翻转 C1ORAW



单周期模式: PWM 模式的特例，将 TMRx_CTRL1 寄存器 OCMEN 位置 1 可开启单周期模式，此模式下，仅在当前计数周期中进行比较匹配，完成当前计数后，TMREN 位清 0，因此仅输出一个脉冲。当配置为向上计数模式时，需要严格配置 CVAL<CxDT≤PR；向下计数时，需严格配置 CVAL>CxDT。图 15-30 展示了计数器向上计数与单周期模式下 PWM 模式 B 配合的例子，计数器仅计数了一个周期，输出信号在这个周期中只输出了一个脉冲。

图 15-30 单周期模式



快速输出模式: 将 TMRx_CM1/CM2 寄存器 CxOEN 位置 1 可开启此功能, 开启后 CxORAW 电平值不再在计数值与 CxDT 匹配时变化, 而是在当前计数周期开始时, 也就是说, 比较结果被提前了, 计数器值与 CxDT 寄存器的比较结果将会提前决定 CxORAW 的电平。

主定时器事件输出

当 TMR 作为主定时器时, 可选择如下信号源作为 TRGOUT 信号输出到次定时器, 选择信号为 TMRxCTRL2 寄存器 PTOS 位。

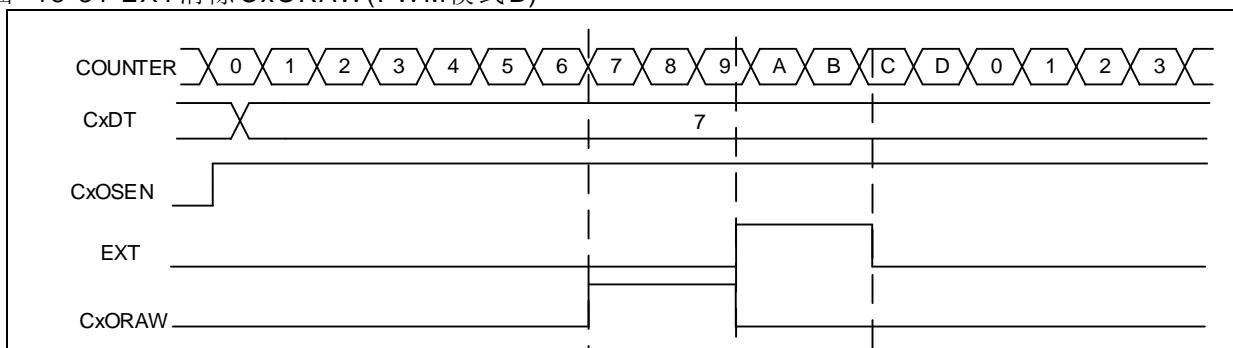
- PTOS=3'b000, TRGOUT 输出软件溢出事件 (TMRx_SWEVT 寄存器 OVFSWTR 位) 或复位事件。
- PTOS=3'b001, TRGOUT 输出计数器使能信号。
- PTOS=3'b010, TRGOUT 输出计数器溢出事件。
- PTOS=3'b011, TRGOUT 输出捕获、比较事件。
- PTOS=3'b100, TRGOUT 输出 C1ORAW 信号。
- PTOS=3'b101, TRGOUT 输出 C2ORAW 信号。
- PTOS=3'b110, TRGOUT 输出 C3ORAW 信号。
- PTOS=3'b111, TRGOUT 输出 C4ORAW 信号。

CxORAW 信号清除

将 TMRx_CM1/CM2 寄存器 CxOSEN 位置 1 后, 指定通道的 CxORAW 信号由 EXT 高电平清 0, 在下一次溢出事件发生前 CxORAW 信号无法被改变。

强制输出模式时, CxORAW 信号清除功能不可用, 只有在输出比较模式或 PWM 模式, 此功能有效。下图显示了使用 EXT 信号清除 CxORAW 的例子, 当 EXT 为高电平时, 原本为高电平的 CxORAW 信号被拉低, 当 EXT 为低电平时, CxORAW 根据计数值和 CxDT 比较结果输出电平。

图 15-31 EXT 清除 CxORAW(PWM 模式 B)



15.2.3.5 定时器同步

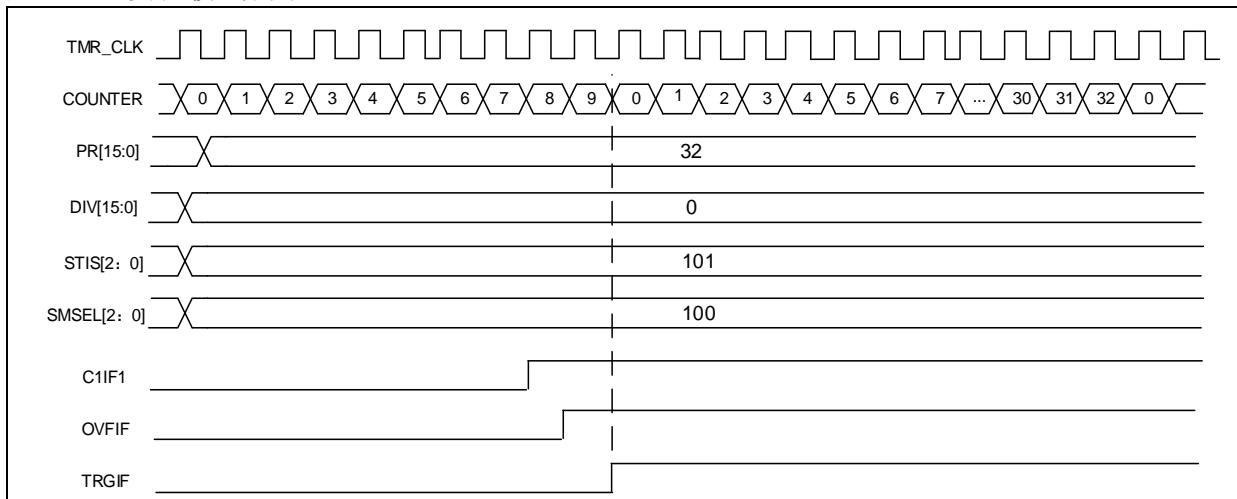
主次定时器之间可由内部连接信号进行同步。主定时器可由 **TMRx_CTRL2** 寄存器 PTOS[2: 0]位选择主定时器输出，即同步信息；次定时器由 **TMRx_STCTRL** 寄存器 SMSEL[2: 0]位选择从模式，即次定时器的工作模式。

定时器从模式有以下几种：

从模式：复位模式

选中的触发信号将复位计数器和预分频器，若 **TMRx_CTRL1** 寄存器 OVFS 位为 0，将产生一个溢出事件。

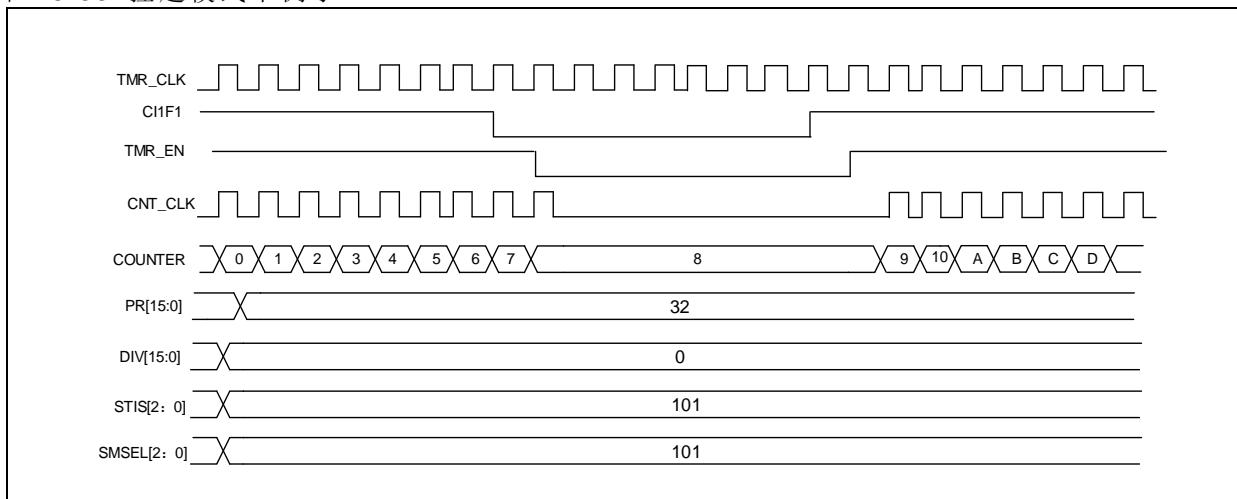
图 15-32 复位模式例子



从模式：挂起模式

挂起模式下，计数的计数和停止受选中触发输入信号控制，当触发输入为高电平时计数器开始计数；当为低电平时，计数器暂停计数。

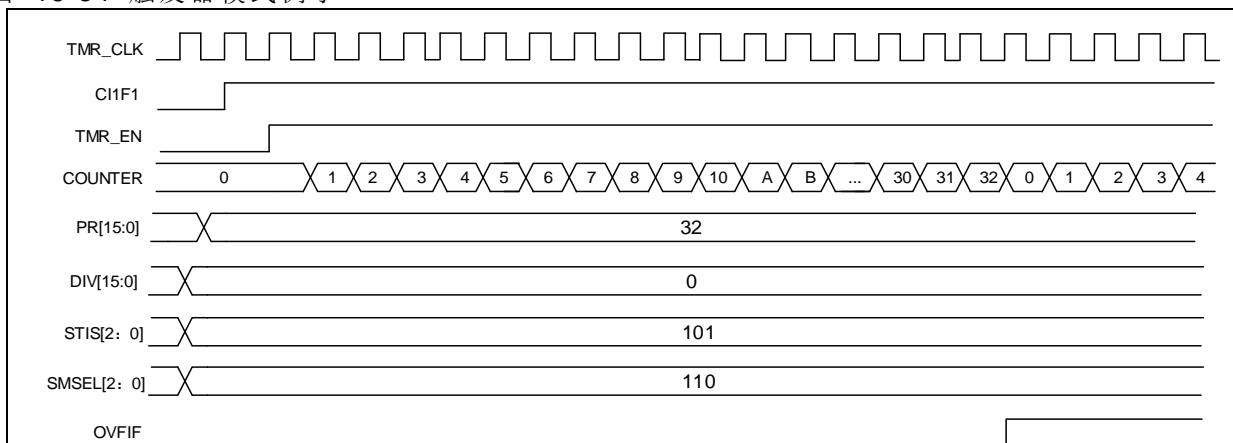
图 15-33 挂起模式下例子



从模式：触发模式

计数器将在选中的触发输入上升沿启动计数（将 TMR_EN 置 1）。

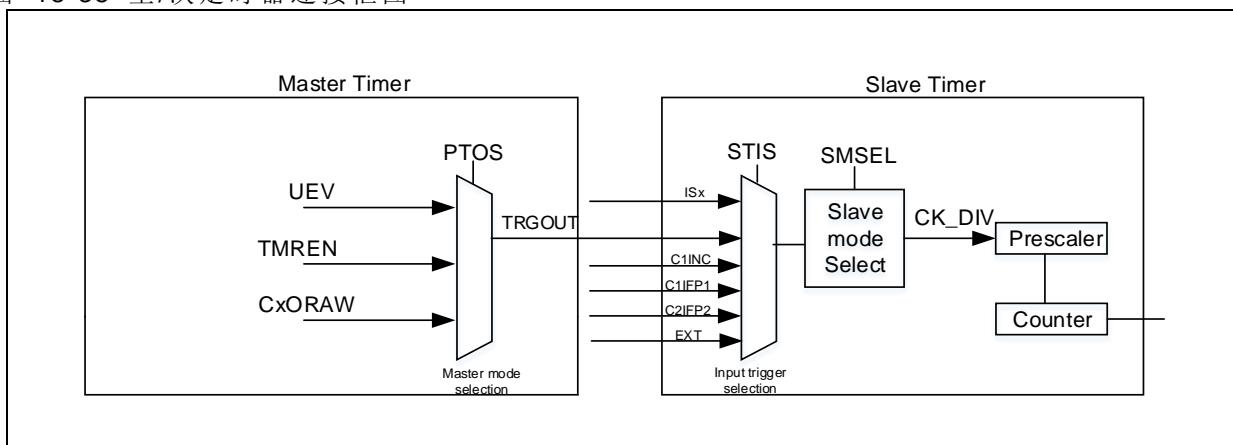
图 15-34 触发器模式例子



主/次定时器互联实例

主/次定时器可分别配置不同的主模式和从模式，两者搭配可实现多种功能，以下提供了一些定时器互联的例子。

图 15-35 主/次定时器连接框图



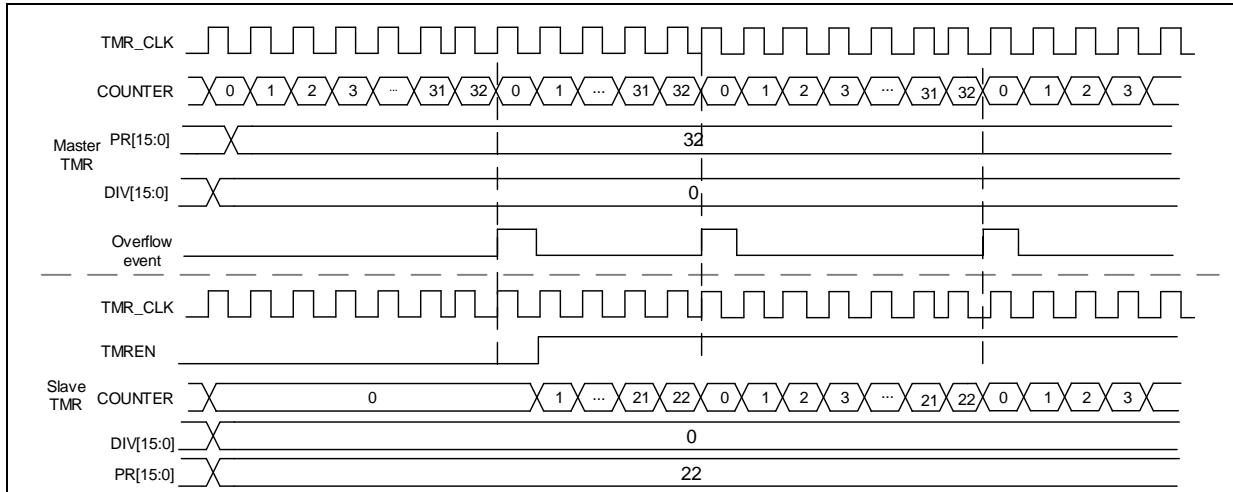
主定时器为次定时器提供时钟：

- 配置主定时器输出信号 TRGOUT 为溢出事件，配置 TMRx_CTRL2 寄存器 PTOS[2:0]=3'b010，主定时器每次计数器溢出输出一个脉冲信号，用作次定时器计数时钟。
- 配置主定时器计数周期（TMRx_PR 寄存器）。
- 配置次定时器触发输入信号 TRGIN 为主定时器输出（TMRx_STCTRL 寄存器的 STIS[2:0]）。
- 配置次定时器使用外部时钟模式 A（TMRx_STCTRL 寄存器的 SMSEL[2:0]=3'b111）。
- 将主定时器和次定时器的 TMREN 位置 1 启动定时器。

主定时器启动次定时器：

- 配置主定时器输出信号 TRGOUT 为溢出事件，配置 TMRx_CTRL2 寄存器 PTOS[2:0]=3'b010，主定时器每次计数器溢出输出一个脉冲信号，用作次定时器计数时钟。
- 配置主定时器计数周期（TMRx_PR 寄存器）。
- 配置次定时器触发输入 TRGIN 为主定时器输出。
- 配置次定时器为触发模式（TMRx_STCTRL 寄存器的 SMSEL=3'b110）
- 置主定时器 TMREN=1 以启动主定时器。

图 15-36 主定时器启动次定时器例子

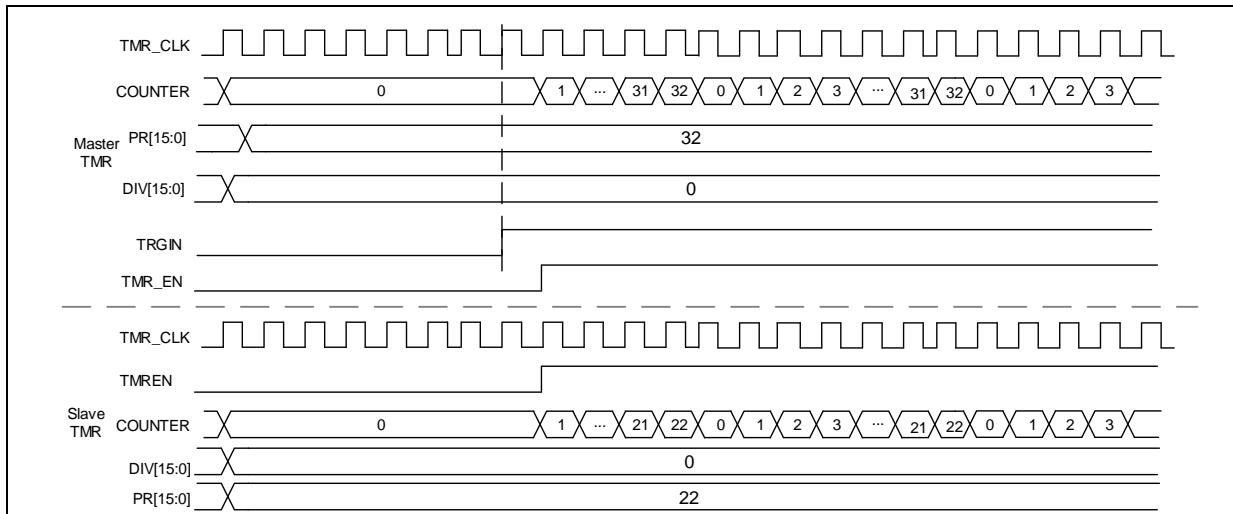


外部触发信号同步启动主、次定时器：

这个例子中，主定时器同时作为主定时器和次定时器，将主定时器的次定时器同步功能开启，此模式用于将主定时器和次定时器保持同步。

- 配置主定时器 TMRx_STCTRL 寄存器 STS 位为 1。
- 配置主定时器输出信号 TRGOUT 为溢出事件，配置 TMRx_CTRL2 寄存器 PTOS[2:0]=3'b010，主定时器每次计数器溢出输出一个脉冲信号，用作次定时器计数时钟。
- 配置主定时器的次定时模式为触发模式，触发源选择 C1IN。
- 配置次定时器触发输入 TRGIN 为主定时器输出。
- 配置次定时器为触发模式（TMRx_STCTRL 寄存器的 SMSEL=3'b110）。

图 15-37 外部触发同时启动主、次定时器



15.2.3.6 TMR DMA

TMR 中有溢出事件 DMA 请求、触发事件 DMA 请求、霍尔事件 DMA 请求以及通道事件 DMA 请求，通过使能 `TMRx_IDEN` 寄存器中相应位，开启 DMA 请求功能，产生相应事件时触发输出 DMA 请求给到 DMA 外设。

TMR DMA Burst 功能

TMR 还支持 TMR DMA Burst 功能，通过 `TMRx_IDEN` 寄存器使能某个事件的 DMA 请求，触发 DMA 改写多个 TMR 连续寄存器。

以溢出事件触发 TMR DMA Burst 功能为例，配置如下：

- 配置 `TMRx_IDEN` 寄存器中 `OVFDEN` 位，使能溢出事件触发 DMA 请求；
- 配置 `TMRx_DMACTRL` 寄存器中 `DTB` 位设置 Burst 传输次数；
- 配置 `TMRx_DMACTRL` 寄存器中 `ADDR` 位设置 Burst 传输的起始地址；
- 使能计数器。

以上配置的 TMR DMA Burst 传输流程如下：

当发生溢出事件时，TMR 将会产生溢出事件的 DMA 请求给到 DMA，DMA 根据请求将数据写入 `TMRx_DMADT` 寄存器，在 TMR 内部则会将 `DMADT` 位的数据写入 Burst 传输起始地址寄存器，并发送 ACK 信号给到 TMR，TMR 接收到 ACK 信号后，将清除当前 DMA 请求；TMR 检测到 Burst 传输并未全部完成，会重新置起溢出事件 DMA 请求给到 DMA，DMA 根据请求将数据再次写入 `TMRx_DMADT` 寄存器，在 TMR 内部则会将 `DMADT` 寄存器的数据写入 Burst 传输起始地址+`0x4` 地址寄存器，并发送 ACK 信号给到 TMR；以此往复，直到 Burst 传输最后一次操作时，检测到此时 Burst 传输已全部完成，溢出事件 DMA 请求信号将不会再被重新置起，直到下一次新的溢出事件到来。

注：使用 TMR DMA Burst 功能时，起始地址到结束地址这个区间，不应有空寄存器以及不应包含 `TMRx_DMACTRL` 和 `TMRx_DMADT` 寄存器。

15.2.3.7 调试模式

当微控制器进入调试模式（Cortex®-M4F 核心停止）时，将 DEBUG 模块中的 `TMRx_PAUSE` 置 1，可以使 TMRx 计数器暂停计数。

15.2.4 TMR2到TMR5寄存器描述

必须以字（32位）的方式操作这些外设寄存器。

表 15-5 TMR2到TMR5寄存器和复位值

寄存器简称	基址偏移量	复位值
TMRx_CTRL1	0x00	0x0000 0000
TMRx_CTRL2	0x04	0x0000 0000
TMRx_STCTRL	0x08	0x0000 0000
TMRx_IDEN	0x0C	0x0000 0000
TMRxISTS	0x10	0x0000 0000
TMRx_SWEVT	0x14	0x0000 0000
TMRx_CM1	0x18	0x0000 0000
TMRx_CM2	0x1C	0x0000 0000
TMRx_CCTRL	0x20	0x0000 0000
TMRx_CVAL	0x24	0x0000 0000
TMRx_DIV	0x28	0x0000 0000
TMRx_PR	0x2C	0x0000 FFFF
TMRx_C1DT	0x34	0x0000 0000
TMRx_C2DT	0x38	0x0000 0000
TMRx_C3DT	0x3C	0x0000 0000
TMRx_C4DT	0x40	0x0000 0000
TMRx_DMACTRL	0x48	0x0000 0000
TMRx_DMADT	0x4C	0x0000 0000
TMR2_RMP	0x50	0x0000 0000
TMR5_RMP	0x50	0x0000 0000

15.2.4.1 TMR2到TMR5控制寄存器1 (TMRx_CTRL1)

域	简称	复位值	类型	功能
位 31: 11	保留	0x00 0000	resd	保持默认值。
位 10	PMEN	0x0	rw	<p>增强模式使能 (Plus Mode Enable) 开启 TMRx 增强模式, 该模式下 TMRx_CVAL, TMRx_PR, TMRx_CxDT 由 16 位扩展为 32 位。 0: 关闭; 1: 开启。 注: TMR2 和 TMR5 才具有此功能, 其它 TMR 设置此位无效。在增强模式关闭状态下, TMRx_CVAL, TMRx_PR, TMRx_CxDT 寄存器只能写入 16 位值。</p>
位 9: 8	CLKDIV	0x0	rw	<p>时钟除频 (Clock divider) 此位用于设置数字滤波器采样频率 f_{DTS} 和定时器时钟频率 f_{CK_INT} 之间的分频比。 00: 无除频, $f_{DTS}=f_{CK_INT}$; 01: 2 除频, $f_{DTS}=f_{CK_INT}/2$; 10: 4 除频, $f_{DTS}=f_{CK_INT}/4$; 11: 保留。</p>
位 7	PRBEN	0x0	rw	<p>周期缓冲使能 (Period buffer enable) 0: 缓冲关闭; 1: 缓冲开启。</p>
位 6: 5	TWCMSEL	0x0	rw	<p>中央双向对齐计数模式选择 (Two-way count mode selection) 00: 单向对齐计数模式, 方向由 OWCDIR 配置; 01: 中央双向对齐计数模式 1, 上下交替计数, CxIF 位只在计数器向下计数时被置起; 10: 中央双向对齐计数模式 2, 上下交替计数, CxIF 位只在计数器向上计数时被置起; 11: 中央双向对齐计数模式 3, 上下交替计数, CxIF 位在计数器向上和向下计数时皆被置起。</p>
位 4	OWCDIR	0x0	rw	<p>单向对齐计数方向 (One-way count direction) 0: 向上; 1: 向下。</p>
位 3	OCMEN	0x0	rw	<p>单周期使能 (One cycle mode enable) 该功能用于选择溢出事件后, 计数器是否停止。 0: 关闭; 1: 开启。</p>
位 2	OVFS	0x0	rw	<p>溢出事件源选择 (Overflow event source) 配置溢出事件或 DMA 请求来源。 0: 来源于计数器溢出、设置 OVFSWTR 位或次定时器控制器产生的溢出事件; 1: 只能来源于计数器溢出。</p>
位 1	OVFEN	0x0	rw	<p>溢出事件使能 (Overflow event enable) 0: 开启; 1: 关闭。</p>
位 0	TMREN	0x0	rw	<p>使能定时器 (TMR enable) 0: 关闭; 1: 开启。</p>

15.2.4.2 TMR2到TMR5控制寄存器2 (TMRx_CTRL2)

域	简称	复位值	类型	功能
位 31: 8	保留	0x00 0000	resd	保持默认值。
位 7	C1INSEL	0x0	rw	C1IN 选择 (C1IN selection) 0: CH1 引脚连到 C1IRAW 输入; 1: CH1、CH2 和 CH3 引脚异或结果连到 C1IRAW 输入。
位 6: 4	PTOS	0x0	rw	主定时器输出信号选择 (Primary TMR output selection) TMRx 输出到次定时器的信号选择: 000: 软件溢出或复位; 001: 使能; 010: 溢出; 011: 比较脉冲; 100: C1ORAW 信号; 101: C2ORAW 信号; 110: C3ORAW 信号; 111: C4ORAW 信号。
位 3	DRS	0x0	rw	DMA 请求源 (DMA request source) DMA 请求来源。 0: 捕获/比较事件; 1: 溢出事件。
位 2: 0	保留	0x0	resd	保持默认值。

15.2.4.3 TMR2到TMR5次定时器控制寄存器 (TMRx_STCTRL)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15	ESP	0x0	rw	外部信号极性 (External signal polarity) 用于选择外部方式。 0: 高电平或上升沿; 1: 低电平或下降沿。
位 14	ECMBEN	0x0	rw	外部时钟模式 B 使能 (External clock mode B enable) 用于启用外部时钟模式 B 0: 关闭; 1: 启开。
位 13: 12	ESDIV	0x0	rw	外部信号除频 (External signal divide) 用于选择降低外部触发频率的除频。 00: 关闭分频; 01: 2 分频; 10: 4 分频; 11: 8 分频。
位 11: 8	ESF	0x0	rw	外部信号滤波 (External signal filter) 用于过滤外部信号, 当外部信号产生了 N 次之后才能被采样。 0000: 无滤波器, 以 f_{DTS} 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。

位 7	STS	0x0	rw	次定时器同步 (Subordinate TMR synchronization) 该位开启后，主次定时器可实现高度同步。 0: 关闭; 1: 开启。
位 6: 4	STIS	0x0	rw	次定时器输入选择 (Subordinate TMR input selection) 用于次定时器的输入选择。 000: 内部选择 0 (IS0); 001: 内部选择 1 (IS1); 010: 内部选择 2 (IS2); 011: 内部选择 3 (IS3); 100: C1IRAW 的输入检测器 (C1INC); 101: 滤波输入 1 (C1IFP1); 110: 滤波输入 2 (C2IFP2); 111: 外部输入 (EXT)。 关于每个定时器中 ISx 的细节，参见表 15-3。
位 3	保留	0x0	resd	保持默认值。
位 2: 0	SMSEL	0x0	rw	次定时器模式选择 (Subordinate TMR mode selection) 000: 关闭从模式; 001: 编码模式 A; 010: 编码模式 B; 011: 编码模式 C; 100: 复位模式 - TRGIN 输入上升沿时，重新初始化计数器; 101: 挂起模式 - TRGIN 输入高电平时，计数器计数; 110: 触发模式 - TRGIN 输入上升沿时，产生触发事件; 111: 外部时钟模式 A - TRGIN 输入上升沿提供时钟; 注：编码模式 A/B/C 配置方法请查看计数模式模式章节。

15.2.4.4 TMR2到TMR5 DMA/中断使能寄存器 (TMRx_IDEN)

域	简称	复位值	类型	功能
位 31: 15	保留	0x0 0000	resd	保持默认值。
位 14	TDEN	0x0	rw	触发 DMA 请求使能 (Trigger DMA request enable) 0: 关闭; 1: 开启。
位 13	保留	0x0	resd	保持默认值。
位 12	C4DEN	0x0	rw	通道 4 的 DMA 请求使能 (Channel 4 DMA request enable) 0: 关闭; 1: 开启。
位 11	C3DEN	0x0	rw	通道 3 的 DMA 请求使能 (Channel 3 DMA request enable) 0: 关闭; 1: 开启。
位 10	C2DEN	0x0	rw	通道 2 的 DMA 请求使能 (Channel 2 DMA request enable) 0: 关闭; 1: 开启。
位 9	C1DEN	0x0	rw	通道 1 的 DMA 请求使能 (Channel 1 DMA request enable) 0: 关闭; 1: 开启。
位 8	OVFDEN	0x0	rw	溢出事件的 DMA 请求使能 (overflow event DMA request enable) 0: 关闭; 1: 开启。
位 7	保留	0x0	resd	保持默认值。
位 6	TIEN	0x0	rw	触发中断使能 (Trigger interrupt enable) 0: 关闭; 1: 开启。
位 5	保留	0x0	resd	保持默认值。

位 4	C4IEN	0x0	rw	通道 4 中断使能 (Channel 4 interrupt enable) 0: 关闭; 1: 开启。
位 3	C3IEN	0x0	rw	通道 3 中断使能 (Channel 3 interrupt enable) 0: 关闭; 1: 开启。
位 2	C2IEN	0x0	rw	通道 2 中断使能 (Channel 2 interrupt enable) 0: 关闭; 1: 开启。
位 1	C1IEN	0x0	rw	通道 1 中断使能 (Channel 1 interrupt enable) 0: 关闭; 1: 开启。
位 0	OVFIEN	0x0	rw	溢出中断使能 (overflow interrupt enable) 0: 关闭; 1: 开启。

15.2.4.5 TMR2到TMR5中断状态寄存器 (TMRx_ISTS)

域	简称	复位值	类型	功能
位 31: 13	保留	0x0 0000	resd	保持默认值。
位 12	C4RF	0x0	rw0c	通道 4 再捕获标记 (Channel 4 recapture flag) 见 C1RF 的描述。
位 11	C3RF	0x0	rw0c	通道 3 再捕获标记 (Channel 3 recapture flag) 见 C1RF 的描述。
位 10	C2RF	0x0	rw0c	通道 2 再捕获标记 (Channel 2 recapture flag) 见 C1RF 的描述。
位 9	C1RF	0x0	rw0c	通道 1 再捕获标记 (Channel 1 recapture flag) C1IF 的状态已经为'1'时是否再次发生了捕获，由硬件置'1'，写'0'清除。 0: 无捕获发生; 1: 捕获发生。
位 8: 7	保留	0x0	resd	保持默认值。
位 6	TRGIF	0x0	rw0c	触发中断标记 (Trigger interrupt flag) 当发生触发事件时由硬件置'1'，写'0'清除。 0: 无触发事件发生; 1: 发生触发事件。 触发事件：在 TRGIN 接收到有效边沿，或挂起模式下接收到任意边沿。
位 5	保留	0x0	resd	保持默认值。
位 4	C4IF	0x0	rw0c	通道 4 中断标记 (Channel 4 interrupt flag) 参考 C1IF 描述。
位 3	C3IF	0x0	rw0c	通道 3 中断标记 (Channel 3 interrupt flag) 参考 C1IF 描述。
位 2	C2IF	0x0	rw0c	通道 2 中断标记 (Channel 2 interrupt flag) 参考 C1IF 描述。
位 1	C1IF	0x0	rw0c	通道 1 中断标记 (Channel 1 interrupt flag) 若通道 1 为输入模式时： 捕获事件发生时由硬件置'1'，由软件清'0'或读 TMRx_C1DT 清'0'。 0: 无捕获事件发生; 1: 发生捕获事件。 若通道 1 为输出模式时： 比较事件发生时由硬件置'1'，由软件清'0'。 0: 无比较事件发生; 1: 发生比较事件。
位 0	OVFIF	0x0	rw0c	溢出中断标记 (Overflow interrupt flag) 当溢出事件发生时由硬件置'1'，由软件清'0'。 0: 无溢出事件发生; 1: 发生溢出事件，若 TMRx_CTRL1 的 OVFEN=0、 OVFS=0 时： - 当 TMRx_SWEVT 寄存器的 OVFSWTR=1 时产生溢出事件；

- 当计数值 CVAL 被触发事件重初始化时产生溢出事件。

15.2.4.6 TMR2到TMR5软件事件寄存器 (TMRx_SW EVT)

域	简称	复位值	类型	功能
位 31: 7	保留	0x000 0000	resd	保持默认值。
位 6	TRGSWTR	0x0	rw	软件触发触发事件 (Trigger event triggered by software) 通过软件触发一个触发事件。 0: 无作用; 1: 制造一个触发事件。
位 5	保留	0x0	resd	保持默认值。
位 4	C4SWTR	0x0	wo	软件触发通道 4 事件 (Channel 4 event triggered by software) 见 C1M 的描述。
位 3	C3SWTR	0x0	wo	软件触发通道 3 事件 (Channel 3 event triggered by software) 见 C1M 的描述。
位 2	C2SWTR	0x0	wo	软件触发通道 2 事件 (Channel 2 event triggered by software) 见 C1M 的描述。
位 1	C1SWTR	0x0	wo	软件触发通道 1 事件 (Channel 1 event triggered by software) 通过软件触发一个通道 1 事件。 0: 无作用; 1: 制造一个通道 1 事件。
位 0	OVFSWTR	0x0	wo	软件触发溢出事件 (Overflow event triggered by software) 通过软件触发一个溢出事件。 0: 无作用; 1: 制造一个溢出事件。

15.2.4.7 TMR2到TMR5通道模式寄存器1 (TMRx_CM1)

输出比较模式:

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15	C2OSEN	0x0	rw	通道 2 输出开关使能 (Channel 2 output switch enable)
位 14: 12	C2OCTRL	0x0	rw	通道 2 输出控制 (Channel 2 output control)
位 11	C2OBEN	0x0	rw	通道 2 输出缓存使能 (Channel 2 output buffer enable)
位 10	C2OIEN	0x0	rw	通道 2 输出立即使能 (Channel 2 output immediately enable)
位 9: 8	C2C	0x0	rw	通道 2 配置 (Channel 2 configure) 当 C2EN=0'时, 这些位用于选择通道 2 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C2IN 映射在 C2IFP2 上; 10: 输入, C2IN 映射在 C1IFP2 上; 11: 输入, C2IN 映射在 STCI 上, 只有在 STIS 选择内部 触发输入时才工作。
位 7	C1OSEN	0x0	rw	通道 1 输出开关使能 (Channel 1 output switch enable) 0: EXT 输入不影响 C1ORAW; 1: 当 EXT 输入高电平时, 将 C1ORAW 清 0。
位 6: 4	C1OCTRL	0x0	rw	通道 1 输出控制 (Channel 1 output control) 这些位用于设置原始信号 C1ORAW 的工作状态。 000: 断开。断开 C1ORAW 到 C1OUT 的输出; 001: 当 TMRx_CVAL=TMRx_C1DT 时, 设置 C1ORAW 为高。 010: 当 TMRx_CVAL=TMRx_C1DT 时, 设置 C1ORAW 为低。

011: 当 TMRx_CVAL=TMRx_C1DT 时, 切换 C1ORAW 的电平。

100: 固定 C1ORAW 为低。

101: 固定 C1ORAW 为高。

110: PWM 模式 A

—OWCDIR=0, 若 TMRx_C1DT>TMRx_CVAL 时设置 C1ORAW 为高, 否则为低;

—OWCDIR=1, 若 TMRx_C1DT<TMRx_CVAL 时设置 C1ORAW 为低, 否则为高。

111: PWM 模式 B

—OWCDIR=0, 若 TMRx_C1DT>TMRx_CVAL 时设置 C1ORAW 为低, 否则为高;

—OWCDIR=1, 若 TMRx_C1DT<TMRx_CVAL 时设置 C1ORAW 为高, 否则为低。

注: 除'000'外, 其余配置下 C1OUT 将连接到 C1ORAW, C1OUT 的输出电平除了会根据 C1ORAW 变化外, 还与 CCTRL 所配置的输出极性有关。

通道 1 输出缓存使能 (Channel 1 output buffer enable)

0: 关闭 TMRx_C1DT 的缓存功能, 写入 TMRx_C1DT 的内容会立即生效。

1: 启用 TMRx_C1DT 的缓存功能, 写入 TMRx_C1DT 的内容将保存到缓存寄存器中, 当发生溢出事件时再更新到 TMRx_C1DT 中。

通道 1 输出立即使能 (Channel 1 output immediately enable)

在 PWM 模式 A 或模式 B 下, 该位能够缩短触发事件到通道 1 的输出响应间的时间。

0: 需要比较 CVAL 与 C1DT 的值之后再产生输出。

1: 无需比较 CVAL 与 C1DT 的值, 当发生触发事件时立即产生输出。

通道 1 配置 (Channel 1 configure)

当 C1EN='0'时, 这些位用于选择通道 1 为输出或输入, 以及输入时的映射选择:

00: 输出;

01: 输入, C1IN 映射在 C1IFP1 上;

10: 输入, C1IN 映射在 C2IFP1 上;

11: 输入, C1IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。

位 3	C1OBEN	0x0	rw	通道 1 输出缓存使能 (Channel 1 output buffer enable) 0: 关闭 TMRx_C1DT 的缓存功能, 写入 TMRx_C1DT 的内容会立即生效。 1: 启用 TMRx_C1DT 的缓存功能, 写入 TMRx_C1DT 的内容将保存到缓存寄存器中, 当发生溢出事件时再更新到 TMRx_C1DT 中。
位 2	C1OIEN	0x0	rw	通道 1 输出立即使能 (Channel 1 output immediately enable) 在 PWM 模式 A 或模式 B 下, 该位能够缩短触发事件到通道 1 的输出响应间的时间。 0: 需要比较 CVAL 与 C1DT 的值之后再产生输出。 1: 无需比较 CVAL 与 C1DT 的值, 当发生触发事件时立即产生输出。
位 1: 0	C1C	0x0	rw	通道 1 配置 (Channel 1 configure) 当 C1EN='0'时, 这些位用于选择通道 1 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C1IN 映射在 C1IFP1 上; 10: 输入, C1IN 映射在 C2IFP1 上; 11: 输入, C1IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。

输入模式:

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 12	C2DF	0x0	rw	通道 2 滤波器 (Channel 2 digital filter)
位 11: 10	C2IDIV	0x0	rw	通道 2 分频系数 (Channel 2 input divider)
位 9: 8	C2C	0x0	rw	通道 2 配置 (Channel 2 configure) 当 C2EN='0'时, 这些位用于选择通道 2 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C2IN 映射在 C2IFP2 上; 10: 输入, C2IN 映射在 C1IFP2 上; 11: 输入, C2IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。
位 7: 4	C1DF	0x0	rw	通道 1 滤波器 (Channel 1 digital filter) 这些位用于配置通道 1 的滤波器。滤波的个数为 N, 则表示发生了 N 次采样事件后输入边沿才能通过滤波器: 0000: 无滤波器, 以 f_{DTS} 采样 0001: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=2 0010: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=4 0011: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=8 0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=6 0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=8 0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=6

				0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=8 1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=6 1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=8 1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=5 1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=6 1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=8 1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=5 1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=6 1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=8
位 3: 2	C1IDIV	0x0	rw	通道 1 分频系数 (Channel 1 input divider) 这些位定义了通道 1 的分频系数。 00: 不分频, 每一个有效的边沿都会产生一次输入; 01: 每 2 个有效的边沿产生一次输入; 10: 每 4 个有效的边沿产生一次输入; 11: 每 8 个有效的边沿产生一次输入。 注: C1EN='0'时, 分频系数复位。
位 1: 0	C1C	0x0	rw	通道 1 配置 (Channel 1 configure) 当 C1EN='0'时, 这些位用于选择通道 1 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C1IN 映射在 C1IFP1 上; 10: 输入, C1IN 映射在 C2IFP1 上; 11: 输入, C1IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。

15.2.4.8 TMR2到TMR5通道模式寄存器2 (TMRx_CM2)

输出比较模式:

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15	C4OSEN	0x0	rw	通道 4 输出开关使能 (Channel 4 output switch enable)
位 14: 12	C4OCTRL	0x0	rw	通道 4 输出控制 (Channel 4 output control)
位 11	C4OBEN	0x0	rw	通道 4 输出缓存使能 (Channel 4 output buffer enable)
位 10	C4OIEN	0x0	rw	通道 4 输出立即使能 (Channel 4 output immediately enable)
位 9: 8	C4C	0x0	rw	通道 4 配置 (Channel 4 configure) 当 C4EN='0'时, 这些位用于选择通道 4 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C4IN 映射在 C4IFP4 上; 10: 输入, C4IN 映射在 C3IFP4 上; 11: 输入, C4IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。
位 7	C3OSEN	0x0	rw	通道 3 输出开关使能 (Channel 3 output switch enable)
位 6: 4	C3OCTRL	0x0	rw	通道 3 输出控制 (Channel 3 output control)
位 3	C3OBEN	0x0	rw	通道 3 输出缓存使能 (Channel 3 output buffer enable)
位 2	C3OIEN	0x0	rw	通道 3 输出立即使能 (Channel 3 output immediately enable)
位 1: 0	C3C	0x0	rw	通道 3 配置 (Channel 3 configure) 当 C3EN='0'时, 这些位用于选择通道 3 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C3IN 映射在 C3IFP3 上; 10: 输入, C3IN 映射在 C4IFP3 上; 11: 输入, C3IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。

输入模式:

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 12	C4DF	0x0	rw	通道 4 滤波器 (Channel 4 digital filter)
位 11: 10	C4IDIV	0x0	rw	通道 4 分频系数 (Channel 4 input divider)

位 9: 8	C4C	0x0	rw	通道 4 配置 (Channel 4 configure) 当 C4EN=0'时, 这些位用于选择通道 4 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C4IN 映射在 C4IFP4 上; 10: 输入, C4IN 映射在 C3IFP4 上; 11: 输入, C4IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。
位 7: 4	C3DF	0x0	rw	通道 3 滤波器 (Channel 3 digital filter)
位 3: 2	C3IDIV	0x0	rw	通道 3 分频系数 (Channel 3 input divider)
位 1: 0	C3C	0x0	rw	通道 3 配置 (Channel 3 configure) 当 C3EN=0'时, 这些位用于选择通道 3 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C3IN 映射在 C3IFP3 上; 10: 输入, C3IN 映射在 C4IFP3 上; 11: 输入, C3IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。

15.2.4.9 TMR2到TMR5通道控制寄存器 (TMRx_CCTRL)

域	简称	复位值	类型	功能
位 31: 14	保留	0x0 0000	resd	保持默认值。
位 13	C4P	0x0	rw	通道 4 极性 (Channel 4 polarity) 见 C1P 的描述。
位 12	C4EN	0x0	rw	通道 4 使能 (Channel 4 enable) 见 C1EN 的描述。
位 11	C3CP	0x0	rw	通道 3 互补极性 (Channel 3 complementary polarity) 定义输入信号的有效沿, 详见 C1P 位描述。
位 10	保留	0x0	resd	保持默认值。
位 9	C3P	0x0	rw	通道 3 极性 (Channel 3 polarity) 见 C1P 的描述。
位 8	C3EN	0x0	rw	通道 3 使能 (Channel 3 enable) 见 C1EN 的描述。
位 7	C2CP	0x0	rw	通道 2 互补极性 (Channel 2 complementary polarity) 定义输入信号的有效沿, 详见 C1P 位描述。
位 6	保留	0x0	resd	保持默认值。
位 5	C2P	0x0	rw	通道 2 极性 (Channel 2 polarity) 见 C1P 的描述。
位 4	C2EN	0x0	rw	通道 2 使能 (Channel 2 enable) 见 C1EN 的描述。
位 3	C1CP	0x0	rw	通道 1 互补极性 (Channel 1 complementary polarity) 定义输入信号的有效沿, 详见 C1P 位描述。
位 2	保留	0x0	resd	保持默认值。 通道 1 极性 (Channel 1 polarity) 通道 1 配置为输出: 0: C1OUT 的有效电平为高 1: C1OUT 的有效电平为低 通道 1 配置为输入: C1CP/C1P 位共同定义输入信号有效沿。 00: C1IN 的有效边沿为上升沿; 作为外部触发使用时, C1IN 不反相。 01: C1IN 的有效边沿为下降沿; 作为外部触发使用时, C1IN 反相。 10: 保留 11: C1IN 的有效边沿为上升沿和下降沿; 作为外部触发使用时, C1IN 不反相。
位 1	C1P	0x0	rw	通道 1 使能 (Channel 1 enable) 0: 禁止输入或输出; 1: 使能输入或输出。
位 0	C1EN	0x0	rw	通道 1 使能 (Channel 1 enable) 0: 禁止输入或输出; 1: 使能输入或输出。

表 15-6 标准 CxOUT 通道的输出控制位

CxEN 位	CxOUT 输出状态
0	禁止输出 (CxOUT=0, CxEN=0)
1	CxOUT = CxORAW + 极性, CxEN=1

注意：连接到标准 CxOUT 通道的外部 I/O 引脚状态，取决于 CxOUT 通道状态和 GPIO 以及 IOMUX 寄存器。

15.2.4.10 TMR2到TMR5计数值寄存器 (TMRx_CVAL)

域	简称	复位值	类型	功能
位 31: 16	CVAL	0x0000	rw	计数值 (Counter value) 当 TMR2 或 TMR5 开启增强模式时 (TMR_CTRL1 中的 PMEN 位)，CVAL 被扩展为 32 位。
位 15: 0	CVAL	0x0000	rw	计数值 (Counter value)

15.2.4.11 TMR2到TMR5分频系数寄存器 (TMRx_DIV)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	DIV	0x0000	rw	分频系数 (Divider value) 计数器时钟频率 $f_{CK_CNT} = f_{TMR_CLK} / (DIV[15: 0]+1)$ 。 DIV 为溢出事件发生时写入的分频系数。

15.2.4.12 TMR2到TMR5周期寄存器 (TMRx_PR)

域	简称	复位值	类型	功能
位 31: 16	PR	0x0000	rw	周期值 (Period value) 当 TMR2 或 TMR5 开启增强模式时 (TMR_CTRL1 中的 PMEN 位)，PR 被扩展为 32 位。
位 15: 0	PR	0xFFFF	rw	周期值 (Period value) 定时器计数的周期值。当周期值为 0 时，定时器不工作。

15.2.4.13 TMR2到TMR5通道1数据寄存器 (TMRx_C1DT)

域	简称	复位值	类型	功能
位 31: 16	C1DT	0x0000	rw	通道 1 数据寄存器值 (Channel 1 data register) 当 TMR2 或 TMR5 开启增强模式时 (TMR_CTRL1 中的 PMEN 位)，C1DT 被扩展为 32 位。
位 15: 0	C1DT	0x0000	rw	通道 1 数据寄存器值 (Channel 1 data register) 若通道 1 配置为输入： C1DT 是前一次通道 1 输入事件 (C1IN) 所保存的 CVAL。 若通道 1 配置为输出： C1DT 是将要和 CVAL 进行比较的值，写入的值是否会立即生效取决于输出缓存使能位 (C1OBEN)，并根据设置在 C1OUT 上产生相应的输出。

15.2.4.14 TMR2到TMR5通道2数据寄存器 (TMRx_C2DT)

域	简称	复位值	类型	功能
位 31: 16	C2DT	0x0000	rw	通道 2 数据寄存器值 (Channel 2 data register) 当 TMR2 或 TMR5 开启增强模式时 (TMR_CTRL1 中的 PMEN 位)，C2DT 被扩展为 32 位。
位 15: 0	C2DT	0x0000	rw	通道 2 数据寄存器值 (Channel 2 data register) 若通道 2 配置为输入： C2DT 是前一次通道 2 输入事件 (C2IN) 所保存的 CVAL。 若通道 2 配置为输出：

C2DT 是将要和 CVAL 进行比较的值，写入的值是否会立即生效取决于输出缓存使能位（C2OBEN），并根据设置在 C2OUT 上产生相应的输出。

15.2.4.15 TMR2到TMR5通道3数据寄存器（TMRx_C3DT）

域	简称	复位值	类型	功能
位 31: 16	C3DT	0x0000	rw	通道 3 数据寄存器值（Channel 3 data register） 当 TMR2 或 TMR5 开启增强模式时（TMR_CTRL1 中的 PMEN 位），C3DT 被扩展为 32 位。
位 15: 0	C3DT	0x0000	rw	通道 3 数据寄存器值（Channel 3 data register） 若通道 3 配置为输入： C3DT 是前一次通道 3 输入事件（C3IN）所保存的 CVAL。 若通道 3 配置为输出： C3DT 是将要和 CVAL 进行比较的值，写入的值是否会立即生效取决于输出缓存使能位（C3OBEN），并根据设置在 C3OUT 上产生相应的输出。

15.2.4.16 TMR2到TMR5通道4数据寄存器（TMRx_C4DT）

域	简称	复位值	类型	功能
位 31: 16	C4DT	0x0000	rw	通道 4 数据寄存器值（Channel 4 data register） 当 TMR2 或 TMR5 开启增强模式时（TMR_CTRL1 中的 PMEN 位），C4DT 被扩展为 32 位。
位 15: 0	C4DT	0x0000	rw	通道 4 数据寄存器值（Channel 4 data register） 若通道 4 配置为输入： C4DT 是前一次通道 4 输入事件（C4IN）所保存的 CVAL。 若通道 4 配置为输出： C4DT 是将要和 CVAL 进行比较的值，写入的值是否会立即生效取决于输出缓存使能位（C4OBEN），并根据设置在 C4OUT 上产生相应的输出。

15.2.4.17 TMR2到TMR5 DMA控制寄存器（TMRx_DMACTRL）

域	简称	复位值	类型	功能
位 31: 13	保留	0x0 0000	resd	保持默认值。
位 12: 8	DTB	0x00	rw	DMA 传输字节（DMA transfer bytes） 这些位定义了传输的字节个数： 00000: 1 个字节 00001: 2 个字节 00010: 3 个字节 00011: 4 个字节 10000: 17 个字节 10001: 18 个字节
位 7: 5	保留	0x0	resd	保持默认值。
位 4: 0	ADDR	0x00	rw	DMA 传输地址偏移（DMA transfer address offset） ADDR 定义了从 TMRx_CTRL1 所在地址开始的偏移量： 00000: TMRx_CTRL1, 00001: TMRx_CTRL2, 00010: TMRx_STCTRL,

15.2.4.18 TMR2到TMR5 DMA数据寄存器（TMRx_DMADT）

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	DMADT	0x0000	rw	DMA 传输的数据寄存器（DMA data register） 通过对 DMADT 寄存器的读写能够实现对任意 TMR 寄存器的操作，其操作的寄存器地址范围是：TMRx 外设地址 + ADDR*4 至 TMRx 外设地址 + ADDR*4 + DTB*4。

15.2.4.19 TMR2通道输入重映射寄存器（TMR2_RMP）

域	简称	复位值	类型	功能
---	----	-----	----	----

位 31: 12	保留	0x0 0000	resd	保持默认值。
位 11: 10	TMR2_IS1_IRMP	0x0	rw	TMR2 内部触发 1 重映射 (TMR2 IS1 input remap) 00: TMR8_TRGOUT 输出 01: EMAC_PTP 1x: USB_SOF
位 9: 0		0x000	resd	保持默认值

15.2.4.20 TMR5通道输入重映射寄存器 (TMR5_RMP)

域	简称	复位值	类型	功能
位 31: 8	保留	0x00 0000	resd	保持默认值。
位 7: 6	TMR5_CH4_IRMP	0x0	rw	TMR5 通道 4 输入重映射 (TMR5 channel 4 input remap) 00: TMR5 通道 4 输入与 GPIO 相连接 01: 内部时钟 LICK 10: 内部时钟 LEXT 11: ERTC 唤醒中断
位 5: 0		0x00	resd	保持默认值。

15.3 通用定时器（TMR9到TMR14）

15.3.1 TMR9到TMR14简介

通用定时器 TMR9 到 TMR14 是支持向上、向下、中央双向对齐计数的 16 位计数器，可实现输入捕获、可编程 PWM 输出、可通过同步功能进行互联。

TMR9/TMR12 包含 2 个捕获/比较寄存器、2 组独立的通道、可实现嵌入死区。

TMR10/TMR11/TMR13/TMR14 包含 1 个捕获/比较寄存器、1 组独立的通道、可实现嵌入死区。

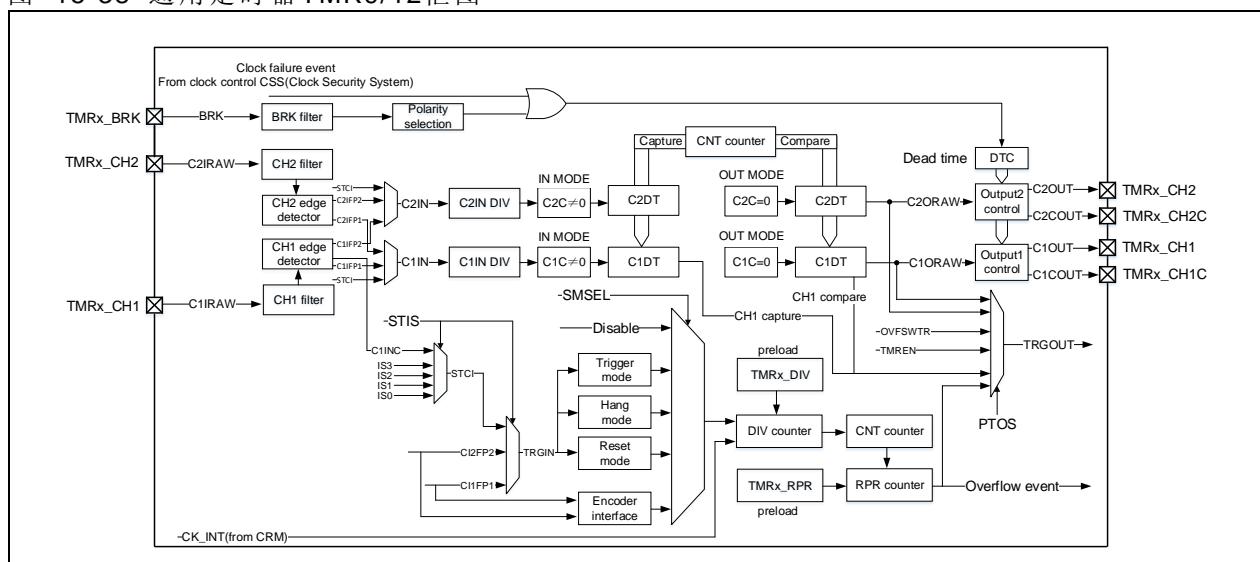
15.3.2 TMR9到TMR14主要功能

15.3.2.1 TMR9和TMR12主要功能

TMR9 和 TMR12 功能包括：

- 可选内部时钟、外部输入、内部触发输入用作计数时钟
- 16 位向上/向下/双向计数器、8 位重复计数计数器
- 2 组独立通道，支持输入捕获、输出比较、PWM 生成、单周期模式、死区插入
- 2 组支持互补输出的独立通道
- 支持 TMR 刹车功能
- 定时器之间可互联同步
- 支持溢出事件、触发事件、刹车输入、通道事件触发中断/DMA
- 支持 TMR DMA Burst 传输

图 15-38 通用定时器 TMR9/12 框图

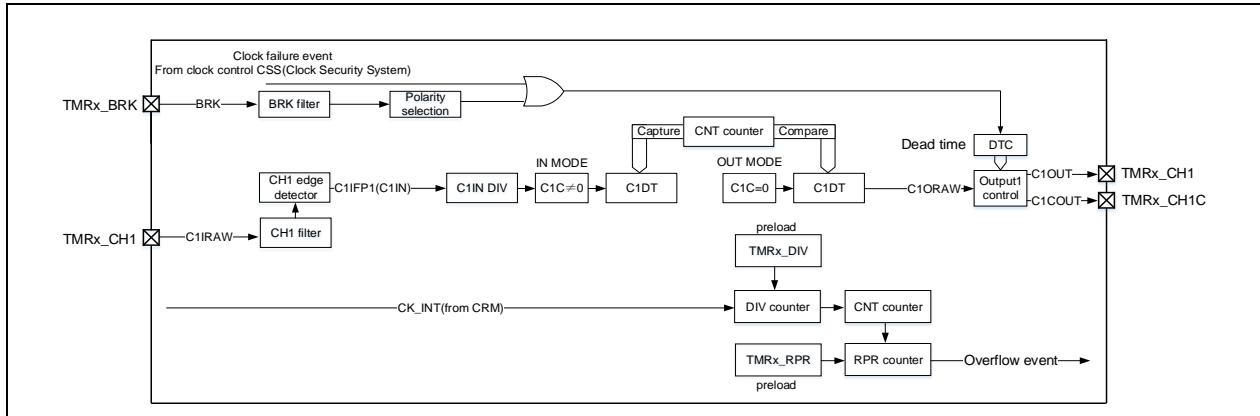


15.3.2.2 TMR10、TMR11、TMR13和TMR14主要功能

通用 TMRx (TMR10、TMR11、TMR13、TMR14) 定时器功能包括：

- 由内部用作计数时钟
- 16 位支持向上/向下/双向计数器、8 位重复计数计数器
- 1 组独立通道，支持输入捕获、输出比较、PWM 生成、单周期模式、死区插入
- 1 组支持互补输出的独立通道
- 支持 TMR 刹车功能
- 支持溢出事件、刹车输入、通道事件触发中断/DMA
- 支持 TMR Burst DMA 传输

图 15-39 通用定时器 TMR10/11/13/14 框图

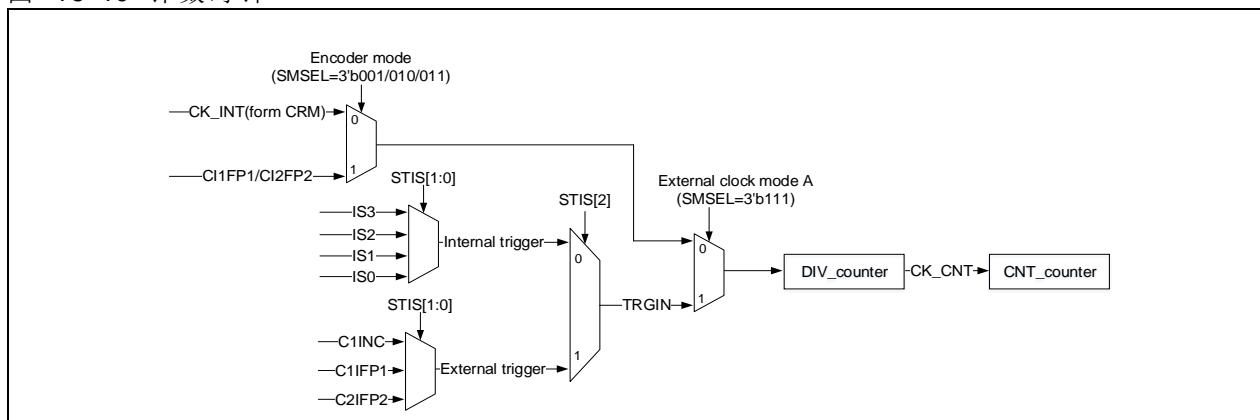


15.3.3 TMR9到TMR14功能描述

15.3.3.1 计数时钟

通用定时器计数时钟可从内部时钟（CK_INT）、外部时钟（外部时钟模式 A）、内部触发输入这些时钟源提供。

图 15-40 计数时钟

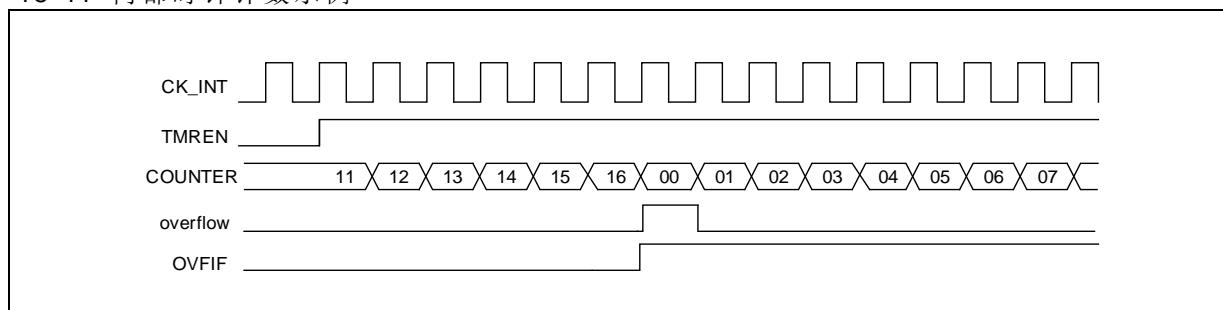


内部时钟（CK_INT）

默认下使用 CK_INT 经由预分频器驱动计数器计数，当 TMR 对应的 APB 时钟预分频系数是 1 时，CK_INT 频率等于 APB 时钟频率，否则 CK_INT 频率等于 APB 时钟频率的 2 倍。相关配置流程如下：

- 配置 TMRx_CTRL1 寄存器 TWCMSEL[1:0]，选择计数模式，若选择单向对齐计数模式，还需配置 TMRx_CTRL1 寄存器 OWCDIR 选择计数方向。
- 配置 TMRx_DIV 寄存器，设置计数器计数频率。
- 配置 TMRx_PR 寄存器，设置计数器计数周期。
- 配置 TMRx_CTRL1 寄存器 TMREN，使能计数器。

图 15-41 内部时钟计数示例



外部时钟（仅 TMR9/12 支持）

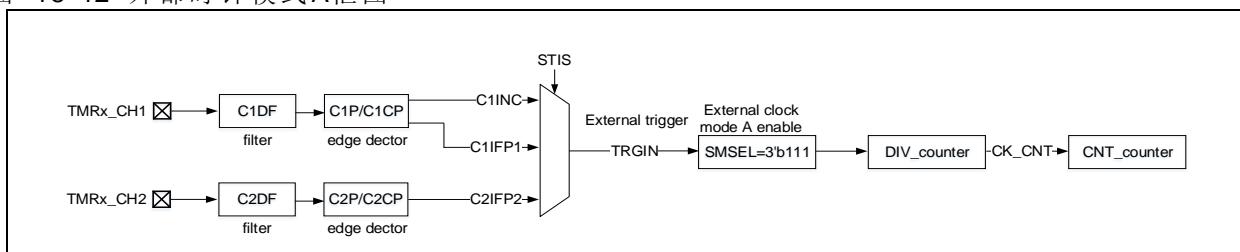
计数时钟由 TRGIN 外部时钟源提供。

当 TMRx_STCTRL 寄存器 SMSEL=3'b111 时，外部时钟模式 A 被选中，配置 TMRx_STCTRL 寄存器 STIS[2:0] 来选择外部时钟源 TRGIN 信号驱动计数器计数。外部时钟源 TRGIN 可选则 C1INC (STIS=3'b100, 通道 1 上升沿和下降沿信号)、C1IFP1 (STIS=3'b101, 通道 1 滤波且极性选择后信号) 和 C2IFP2 (STIS=3'b110, 通道 2 滤波且极性选择后信号)。

若要使用外部时钟模式 A，可按如下步骤配置：

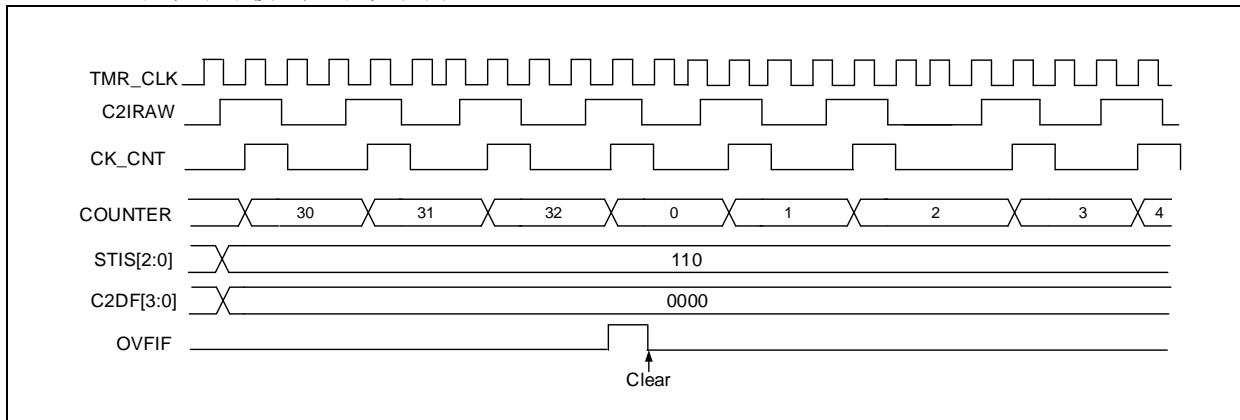
- 配置外部时钟源 TRGIN 参数。
 - 若选择 TRGIN 来源为 TMRx_CH1，需配置通道 1 输入滤波 (TMRx_CM1 寄存器 C1DF[3:0]) 和通道 1 输入极性 (TMRx_CCTRL 寄存器 C1P/C1CP)。
 - 若选择 TRGIN 来源为 TMRx_CH2，需配置通道 2 输入滤波 (TMRx_CM1 寄存器 C2DF[3:0]) 和通道 1 输入极性 (TMRx_CCTRL 寄存器 C2P/C2CP)。
- 配置 TMRx_STCTRL 寄存器 STIS[2:0]，设置 TRGIN 信号来源。
- 配置 TMRx_STCTRL 寄存器 SMSEL=3'b111，使能外部时钟模式 A。
- 配置 TMRx_DIV 寄存器 DIV[15:0]，设置计数器计数频率。
- 配置 TMRx_PR 寄存器 PR[15:0]，设置计数器计数周期。
- 配置 TMRx_CTRL1 寄存器 TMREN，使能计数器。

图 15-42 外部时钟模式 A 框图



注：由于同步逻辑，输入端信号与计数器实际时钟之间存在一定延时。

图 15-43 外部时钟模式 A 计数示例



内部触发输入 (ISx, 仅 TMR9/12 支持)

定时器之间支持互联同步，因此一个定时器的 TMR_CLK 可由另一个定时器输出信号 TRGOUT 提供。配置 STIS[2:0] 选择内部触发信号驱动计数器计数。

TMR9 到 TMR14 内含一个 16 位预分频器，用于产生驱动计数器计数的时钟 CK_CNT，通过配置 TMRx_DIV 寄存器值，可灵活调整 CK_CNT 与 TMR_CLK 之间的分频关系。预分频值可在任何时刻修改，但只在下一个溢出事件发生时，新值才会生效。

内部触发输入配置流程如下：

- 配置 TMRx_PR 寄存器，设置计数器计数周期。
- 配置 TMRx_DIV 寄存器，设置计数器计数频率。
- 配置 TMRx_CTRL1 寄存器 TWCMSEL[1:0] 位，设置计数器计数模式。
- 配置 TMRx_STCTRL 寄存器 STIS[2:0] 位范围为 3'b000~3'b011，选择内部触发。
- 配置 TMRx_STCTRL 寄存器 SMSEL[2:0]=3'b111，选择外部时钟模式 A。
- 配置 TMRx_CTRL1 寄存器 TMREN 位，使能 TMRx 计数。

图 15-44 预分频器的参数从0变到3计数示例

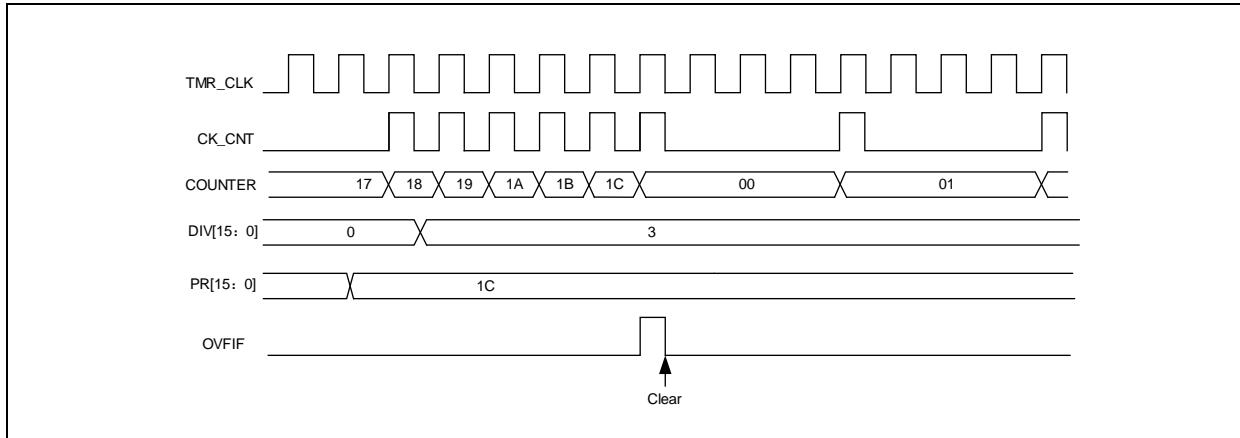


表 15-7 TMRx 内部触发连接

次定时器	IS0 (STIS = 000)	IS1 (STIS = 001)	IS2 (STIS = 010)	IS3 (STIS = 011)
TMR9	TMR2	TMR3	TMR10_OC	TMR11_OC
TMR12	TMR4	TMR5	TMR13_OC	TMR14_OC

注意：如果某个产品中没有相应的定时器，则对应的触发信号 ISx 也不存在。

15.3.3.2 计数模式

TMR9 到 TMR14 支持多种计数模式，用来满足不同的应用场景。其内部拥有一个支持 16 位向上计、向下、中央双向对齐计数模式计数器。

TMRx_PR 寄存器用于设置计数器计数周期。默认 TMRx_PR 寄存器值会立即传入它的影子寄存器；当开启周期缓冲功能后 (TMRx_CTRL1 寄存器 PRBEN 置 1)，TMRx_PR 寄存器值在溢出事件发生时传入它的影子寄存器。

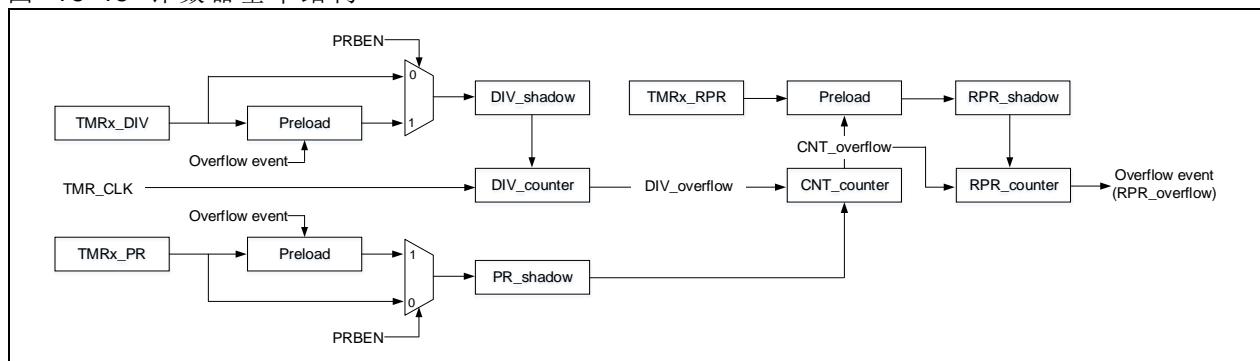
TMRx_DIV 寄存器用于设置计数器计数频率，每 (DIV[15:0]+1) 个计数时钟周期，计数器计数一次。TMRx_DIV 寄存器值在溢出事件时更新至它的影子寄存器。

读取 TMRx_CNT 寄存器会返回当前计数器计数值，写入 TMRx_CNT 寄存器会更新计数器当前计数值为写入值。

默认允许产生溢出事件，设置 TMRx_CTRL1 寄存器 OVFEN=1 将禁止溢出事件产生。TMRx_CTRL1 寄存器 OVFS 用于选择溢出事件来源，默认计数器上溢或下溢、置位 OVFSWTR、复位模式次定时器控制器产生的复位信号产生溢出事件。置位 OVFS 后，只有计数器上溢或下溢产生溢出事件。

TMREN 位置 1 将使能定时器计数，由于同步逻辑，实际驱动计数器的使能信号 TMR_EN 相对于 TMREN 延迟一个时钟周期。

图 15-45 计数器基本结构



向上计数模式

配置 TMRx_CTRL1 寄存器 TWCSEL[1:0]=2'b00, OWCDIR=1'b0 开启向上计数模式，计数值达到 TMRx_PR 值时，重新从 0 向上计数，计数器上溢并产生溢出事件，同时 OVFIF 位置 1。若禁止产生溢出事件，计数器溢出后不再重载预分频值和周期值，否则预分频值和周期值在溢出事件后更新。

图 15-46 PRBEN=0 时的溢出事件

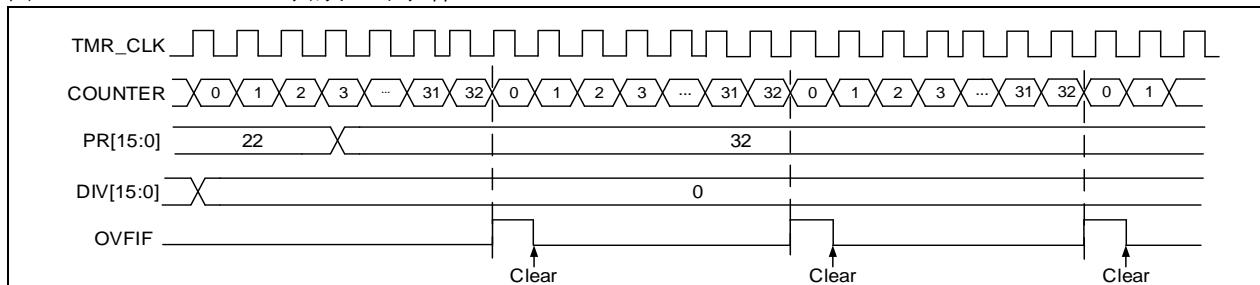
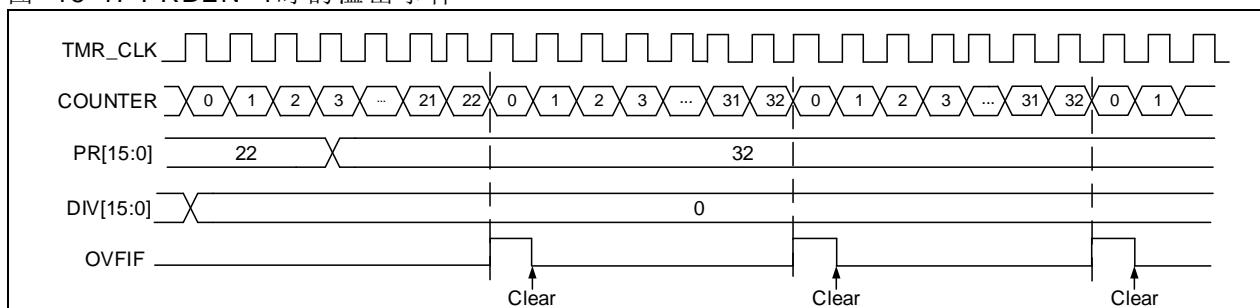


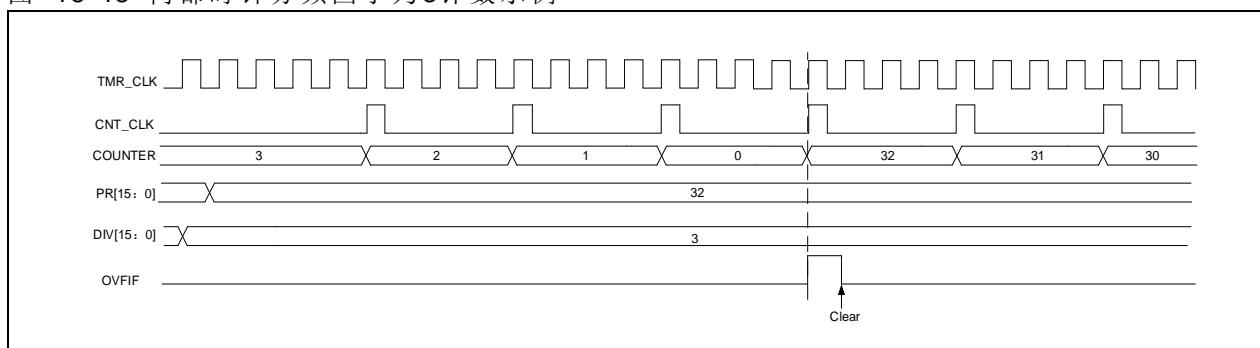
图 15-47 PRBEN=1 时的溢出事件



向下计数模式

配置 TMRx_CTRL1 寄存器 TWCSEL[1:0]=2'b00, OWCDIR=1'b1 开启向下计数模式，计数值达到 0 值并重新从 TMRx_PR 向上下数时，计数器下溢并产生溢出事件。

图 15-48 内部时钟分频因子为 3 计数示例



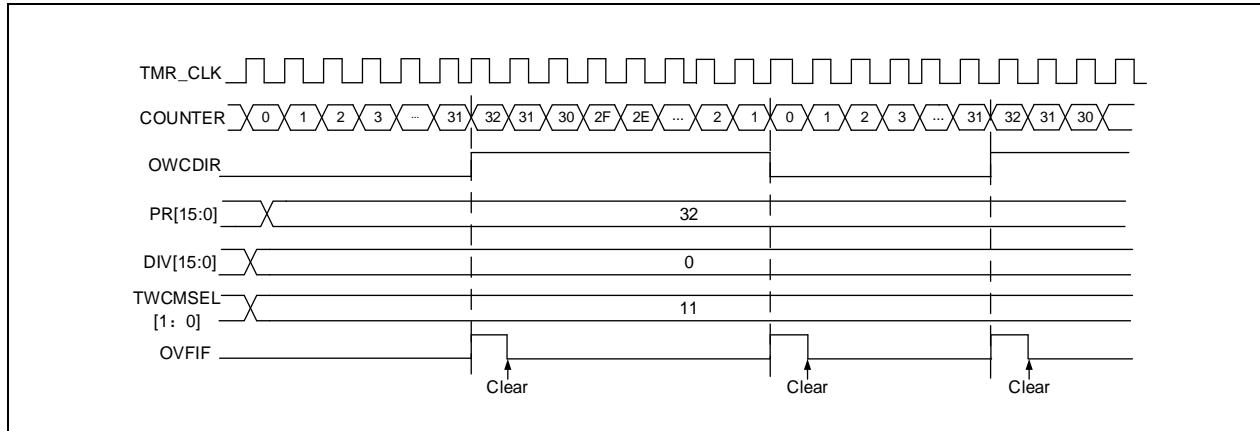
中央双向对齐计数模式

配置 TMRx_CTRL1 寄存器 TWCSEL[1:0]≠2'b00 开启中央双向对齐计数模式，中央双向对齐计数模式下计数器交替向上、向下计数。计数值从 TMRx_PR 值向下计数到 1 值，产生下溢事件，然后从 0 开始向上计数；向上计数到 TMRx_PR 值-1，产生上溢事件，之后从 TMRx_PR 值向下计数。计数器计数方向由计数器方向控制位 (OWCDIR) 实时查看。

TMRx_CTRL1 寄存器 TWCSEL[1:0]位还用于选择中央双向对齐计数模式下 CxIF 标志置起方式，中央双向对齐计数模式 1 (TWCSEL[1:0]=2'b01) 仅允许 CxIF 标志位在计数器向下计数时置起；双向对齐计数模式 2 (TWCSEL[1:0]=2'b10) 仅允许 CxIF 标志位在计数器向上计数时置起；双向对齐计数模式 3 (TWCSEL[1:0]=2'b11) 允许 CxIF 标志位在计数器向上和向下计数时置起。

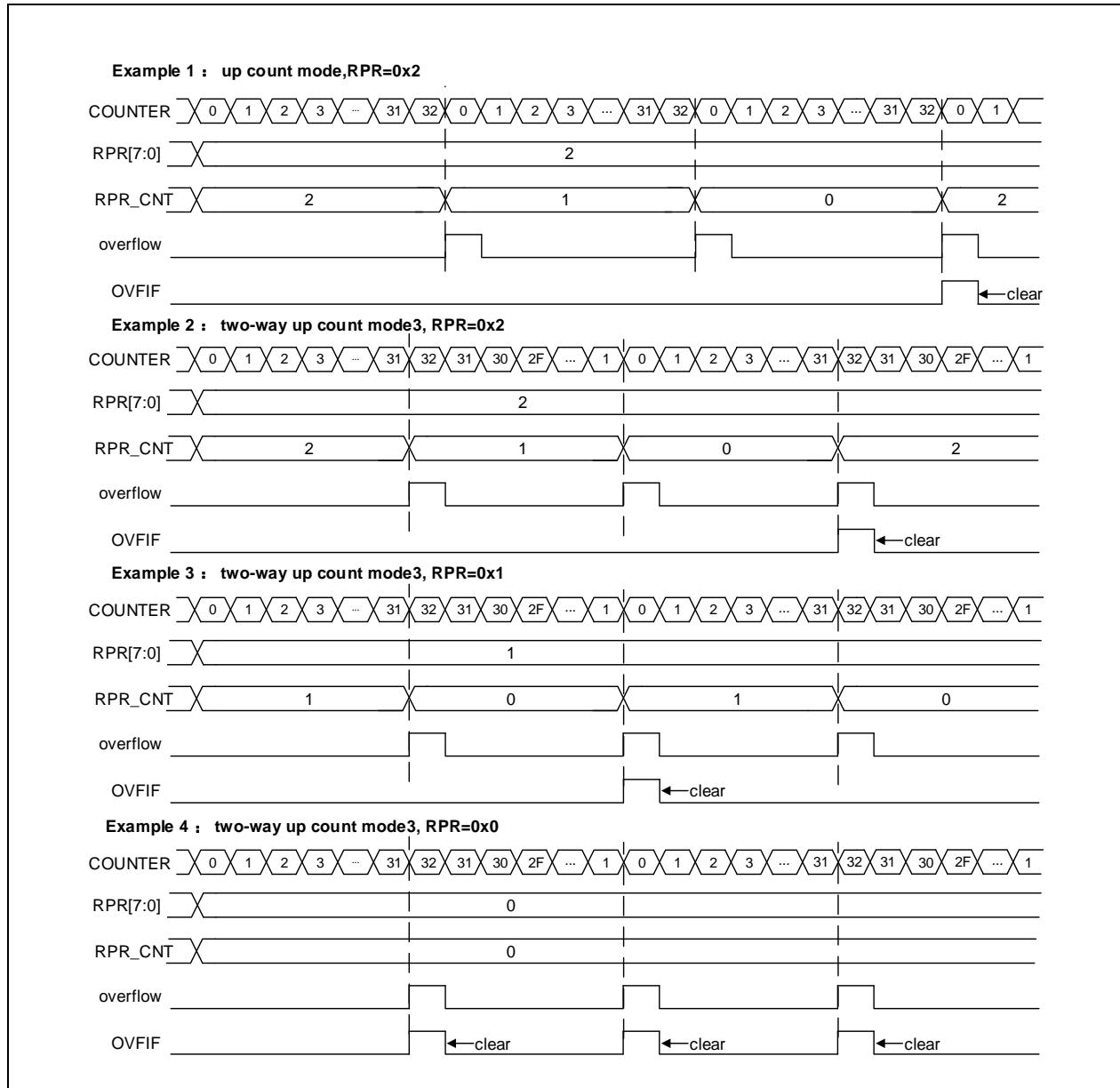
注意： 中央双向对齐计数模式下，OWCDIR 位为只读位。

图 15-49 内部时钟分频因子为0计数示例

**重复计数模式：**

TMRx_RPR 寄存器用于配置重复计数器计数周期，TMRx_RPR 寄存器为非 0 值时，重复计数模式启动。重复计数模式下，每 ($RPR[7:0]+1$) 次计数器溢出将产生一次溢出事件。每次计数器溢出，重复计数器递减，仅当重复计数器计数值等于 0 值时，计数器溢出会产生溢出事件。通过配置不同重复计数器值，可调整溢出事件产生的频率。

图 15-50 向上计数模式和中央双向对齐计数模式时 OVFIF



15.3.3.3 TMR输入部分

TMR9 和 12 拥有两个独立通道，TMR10、11、13、14 拥有一个独立通道。每个通道可配置为输入或输出，当配置为输入时，每个通道输入信号依次经过以下处理：

- TMRx_CHx 经过预处理输出 CxIRAW。
- CxIRAW 输入数字滤波器，输出滤波后信号 CxIF。数字滤波器通过 TMRx_CM1 寄存器 CxDf 位配置采样频率和次数。
- CxIF 输入边沿检测器，输出边沿选择后信号 CxIFP_x。边沿选择由 TMRx_CCTRL 寄存器 CxP 和 CxCp 位共同控制，可选择输入上升沿、下降沿或双边沿有效。
- CxIFP_x 输入捕获信号选择器，输出选择后信号 CxIN。捕获信号选择器由 TMRx_CM1 寄存器 CxC 控制，可选择 CxIN 来源为 CxIFP_x、CyIFP_y、STCI。其中 CyIFP_y ($x \neq y$) 是来自通道 y 的 CyIFPy 信号；STCI 来自次定时器控制器，由 STIS 位选择来源。单通道 TMR 仅支持选择 CxIN 来源为 CxIFP_x。
- CxIN 经由输入通道分频器，输出分频后信号 CxIPS。分频系数由 TMRx_CM1 寄存器 CxIDIV 位配置为不分频、2 分频、4 分频或 8 分频。

图 15-51 输入/输出通道 1 的主电路

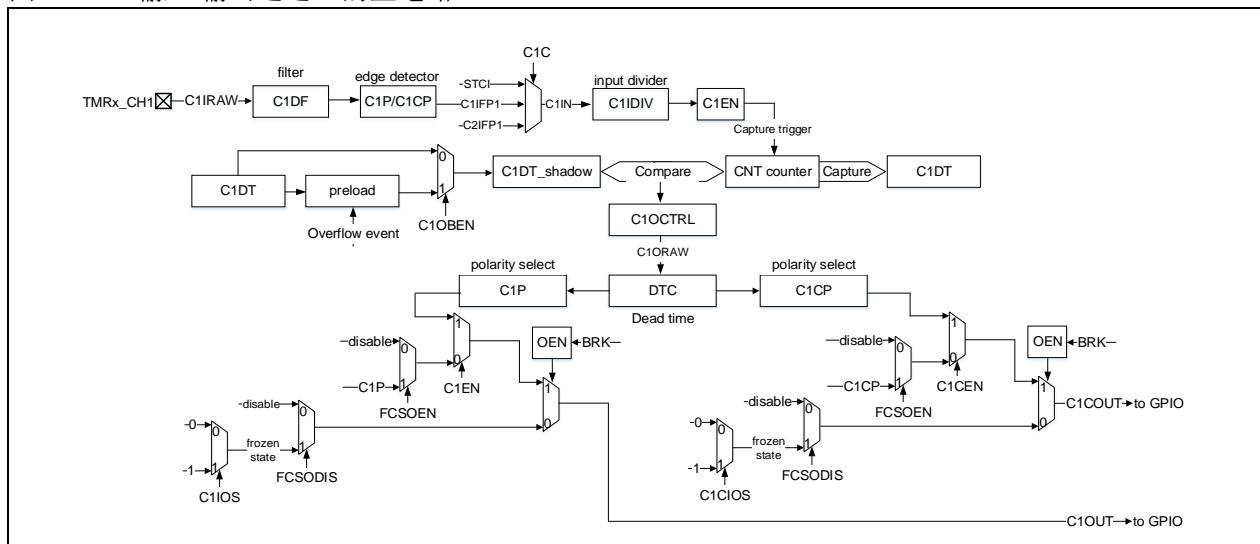
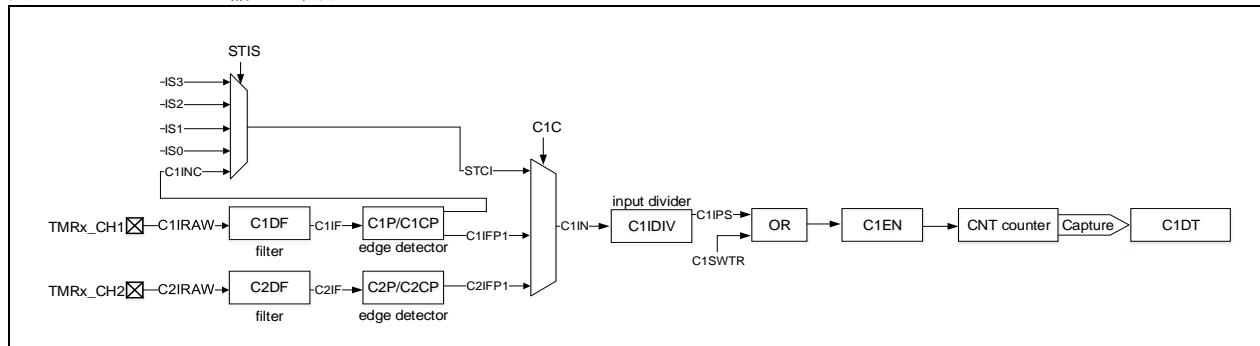


图 15-52 通道 1 输入部分



输入模式

此模式下，当选中的触发信号被检测到，通道寄存器（TMRx_CxDT）记录当前计数器计数值，并将捕获比较中断标志位（TMRx_ISTS 寄存器 CxIF）置 1，若已使能通道中断（TMRx_IDEN 寄存器 CxIEN）、通道 DMA 请求（TMRx_IDEN 寄存器 CxDEN）则产生相应的中断和 DMA 请求。若在 CxIF 置 1 后检测到触发信号，将产生捕获溢出事件，TMRx_CxDT 会使用当前计数器计数值覆盖之前记录的计数器计数值，同时通道再捕获标志位（TMRx_ISTS 寄存器 CxRF）置 1。

若要捕获 C1IN 输入的上升沿，可按如下进行配置：

- 将通道模式寄存器 1（TMRx_CM1）中的 C1C 位配置为 2'b01，选择 C1IN 作为通道 1 输入。
- 配置 C1IN 信号滤波器带宽（CxDF[3: 0]）。
- 配置 C1IN 通道的有效沿，在 TMRx_CCTRL 寄存器中写入 C1P=2'b0（上升沿）。

- 配置 C1IN 信号捕获分频 (C1DIV[1: 0])。
- 使能通道 1 输入捕获 (C1EN=1)。
- 根据需要设置 TMRx_IDEN 寄存器中的 C1IEN 位、TMRx_IDEN 寄存器中的 C1DEN 位，选择中断请求或 DMA 请求。

PWM 输入 (TMR9/12 支持)

PWM 输入模式适用于通道 1 和 2，要使用此模式，需要将 C1IN 和 C2IN 映射到同一 TMRx_CHx，并且通道 1 或 2 的 CxIFPx 配置成触发次定时器控制器复位。

PWM 输入模式可用于测量输入信号的周期和占空比，如需测量通道 1 输入信号的周期和占空比，操作步骤如下：

- 配置 TMRx_CM1 寄存器 C1C=2'b01，选择 C1IN 为 C1IFP1。
- 配置 TMRx_CCTRL 寄存器 C1P=1'b0，选择 C1IFP1 上升沿有效。
- 配置 TMRx_CM1 寄存器 C2C=2'b10，选择 C2IN 为 C2IFP2。
- 配置 TMRx_CCTRL 寄存器 C2P=1'b1，选择 C2IFP2 下降沿有效。
- 配置 TMRx_STCTRL 寄存器 STIS=3'b101，选择次定时器触发信号为 C1IFP1。
- 配置 TMRx_STCTRL 寄存器 SMSEL=3'b110，选择次定时器模式为复位模式。
- 配置 TMRx_CCTRL 寄存器 C1EN=1'b1，C2EN=1'b1。使能通道 1 和通道 2 输入捕获。

上述配置下，通道 1 输入信号的上升沿会触发捕获并将捕获值存储到 C1DT 寄存器，同时通道 1 输入信号上升沿复位计数器。通道 1 输入信号下降沿触发捕获并将捕获值存储到 C2DT 寄存器。通道 1 输入信号的周期可通过 C1DT 计算，占空比可通过 C2DT 计算。

图 15-53 PWM 输入模式配置实例

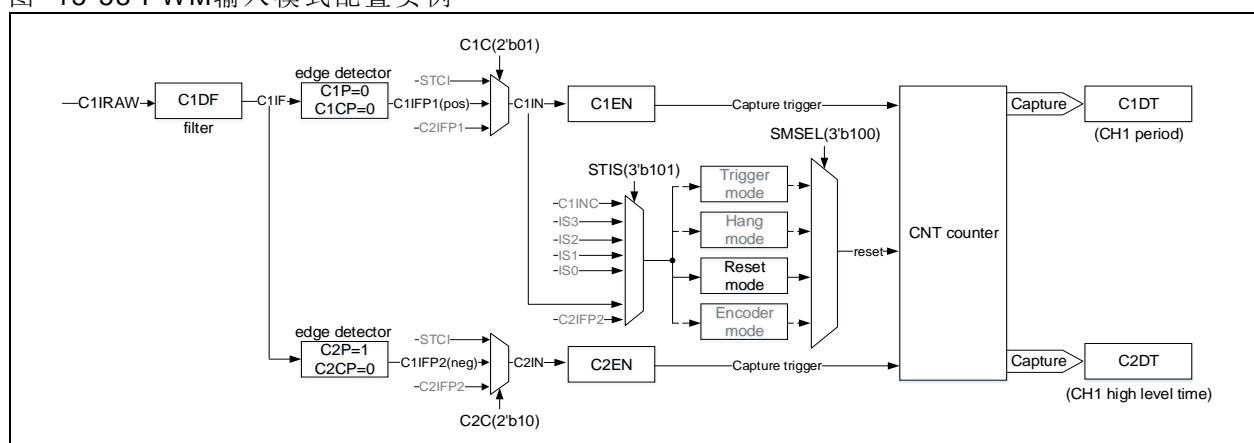
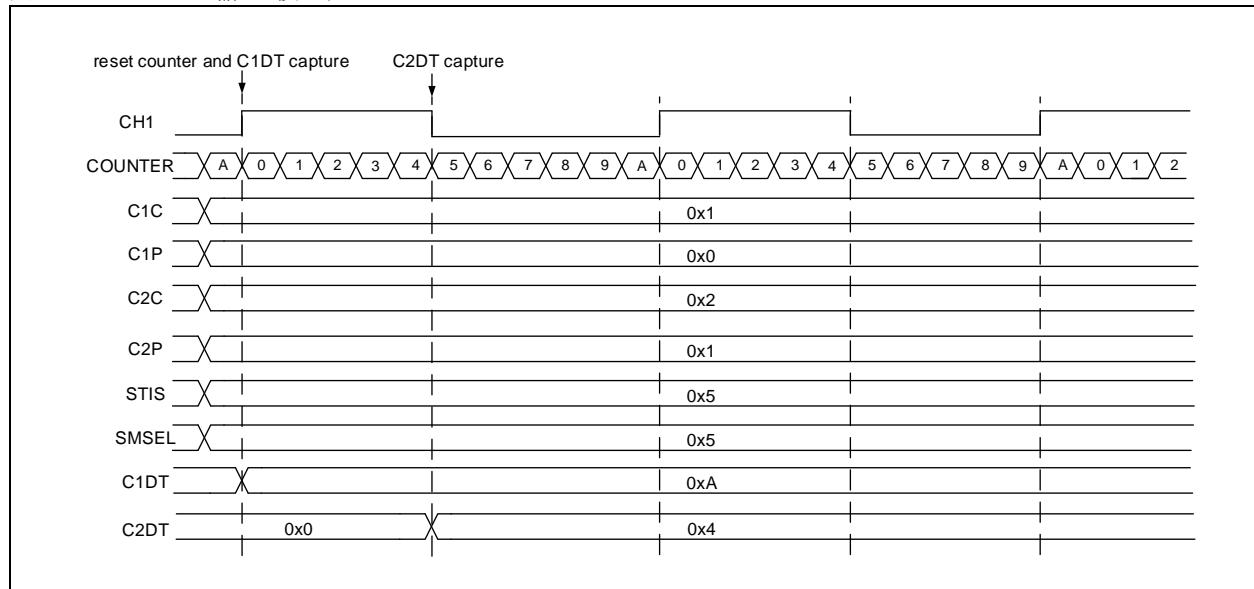


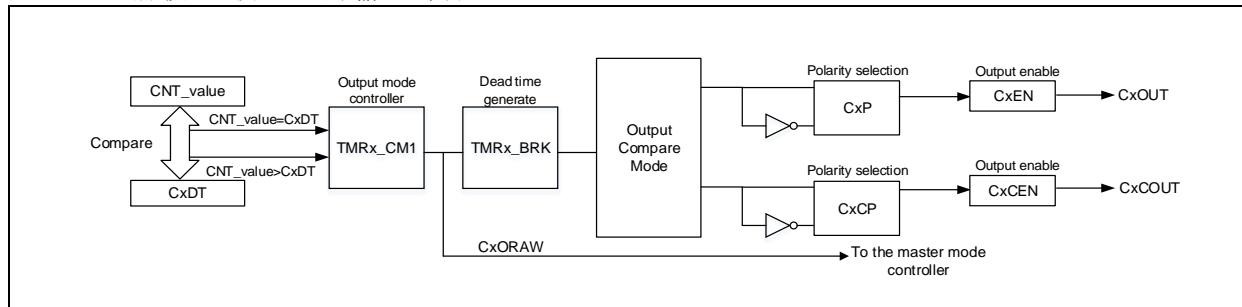
图 15-54 PWM 输入模式



15.3.3.4 TMR 输出部分

TMR 的输出部分由比较器和输出控制构成，用于编程输出信号的周期、占空比、极性。TMR9 和 TMR12 含有两个通道，TMR10/11/13/14 含有一个通道，如下图所示：

图 15-55 捕获/比较通道的输出部分



输出模式

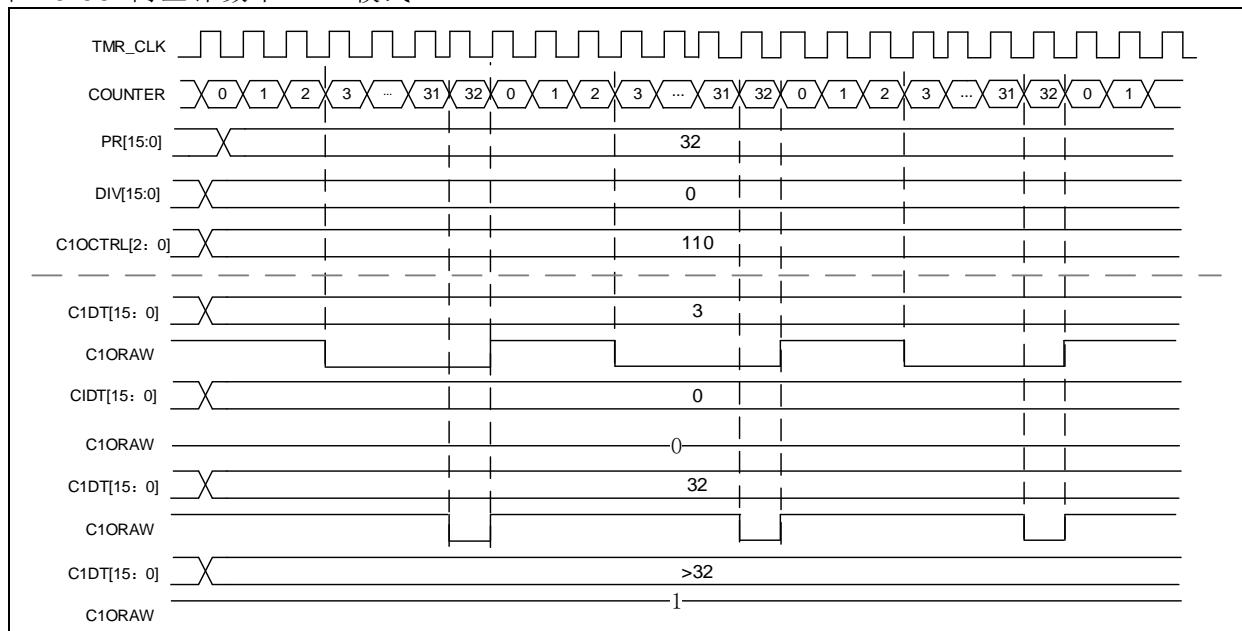
配置 $CxOCTRL[1:0] \neq 2'b00$ 将通道配置为输出可实现多种输出模式，此时，计数器计数值将与 $CxDT$ 寄存器值比较，并根据 $CxOCTRL[2:0]$ 位配置的输出模式，产生中间信号 $CxORAW$ ，再经过输出控制逻辑处理后输送到 IO。输出信号的周期由 $TMRx_PR$ 寄存器值配置，占空比则由 $CxDT$ 寄存器值配置。

输出比较模式有以下子类：

PWM 模式 A: $CxOCTRL=3'b110$ 时，开启 PWM 模式 A。向上计数时， $TMRx_C1DT > TMRx_CVAL$ 时 $C1ORAW$ 输出高电平，否则为低电平；向下计数时， $TMRx_C1DT < TMRx_CVAL$ 时 $C1ORAW$ 输出低电平，否则为高电平。图 15-56 展示了计数器向上计数与 PWM 模式 A 配合的例子， $PR=0x32$ ， $CxDT$ 配置为不同的值时输出时输出信号的翻转情况。若要使用 PWM 模式 A，可按如下方式配置。

- 配置 $TMRx_PR$ 寄存器，设置 PWM 周期。
- 配置 $TMRx_CxDT$ 寄存器，设置 PWM 占空比。
- 配置 $TMRx_CM1/CM2$ 寄存器 $CxOCTRL$ 位为 $3'b110$ ，设置输出模式为 PWM 模式 A。
- 配置 $TMRx_DIV$ 寄存器，设置计数器计数频率。
- 配置 $TMRx_CTRL1$ 寄存器 $TWCMSEL[1:0]$ 位，设置计数器计数模式。
- 配置 $TMRx_CCTRL$ 寄存器 CxP 位、 $CxCP$ 位，设置输出极性。
- 配置 $TMRx_CCTRL$ 寄存器 $CxEN$ 位、 $CxCEN$ 位，使能通道输出。
- 配置 $TMRx_BRK$ 寄存器 OEN 位，使能 $TMRx$ 输出。
- 配置 TMR 输出通道对应 GPIO 为对应的复用模式。
- 配置 $TMRx_CTRL1$ 寄存器 $TMREN$ 位，使能 $TMRx$ 计数。

图 15-56 向上计数下 PWM 模式 A



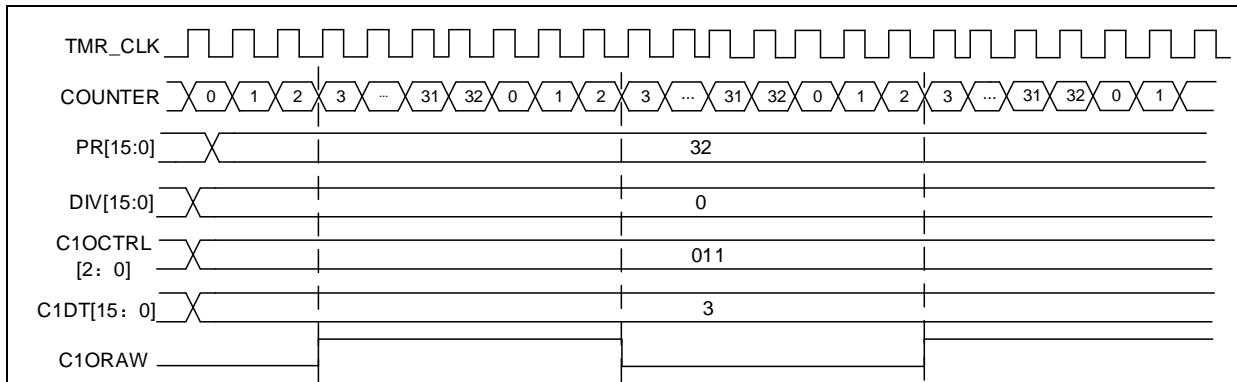
PWM 模式 B: $CxOCTRL=3'b111$ 时，开启 PWM 模式 B。向上计数时， $TMRx_C1DT > TMRx_CVAL$ 时 $C1ORAW$ 输出低电平，否则为高电平；向下计数时， $TMRx_C1DT < TMRx_CVAL$ 时 $C1ORAW$ 输出高电

平，否则为低电平。

强制输出模式：CxOCTRL=3'b100/101 时，开启强制输出模式。此时，CxORAW 信号的电平被强制输出为配置的电平，而与计数值无关。虽然输出信号不依赖于比较结果，但通道标志位和 DMA 请求仍依赖于比较结果。

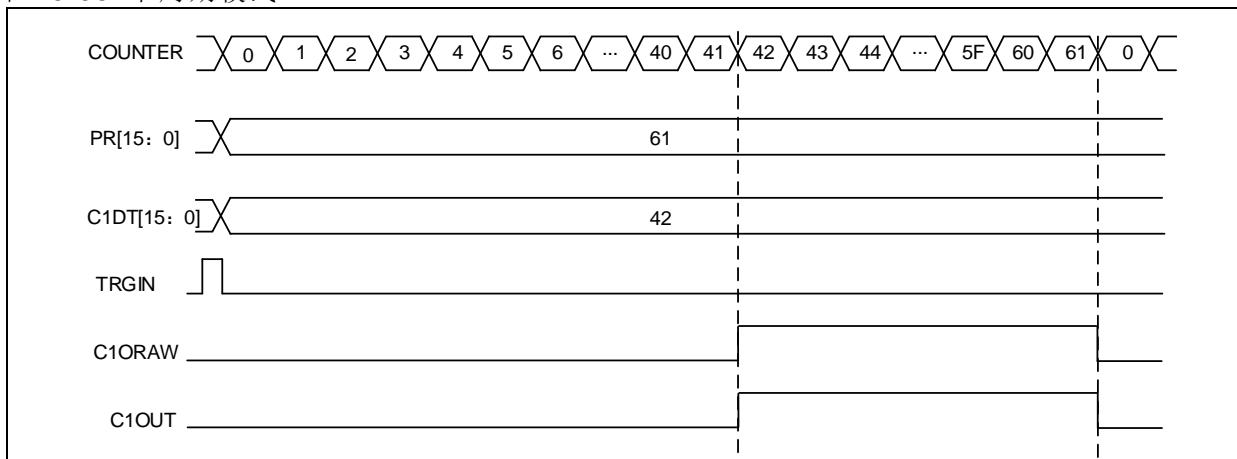
输出比较模式：CxOCTRL=3'b001/010/011 时，开启输出比较模式。此时，当计数值与 CxDT 值匹配时，CxORAW 强制输出高电平（CxOCTRL=3'b001）、低电平（CxOCTRL=3'b010）或进行电平翻转（CxOCTRL=3'b011）。图 15-57 展示了输出比较模式（翻转）的例子，C1DT=0x3，当计数值等于 0x3 时，输出电平 C1OUT 被翻转。

图 15-57 计数值与C1DT值匹配时翻转C1ORAW



单周期模式：PWM 模式的特例，将 TMRx_CTRL1 寄存器 OCMEN 位置 1 可开启单周期模式，此模式下，仅在当前计数周期中进行比较匹配，完成当前计数后，TMREN 位清 0，因此仅输出一个脉冲。当配置为向上计数模式时，需要严格配置 CVAL<CxDT≤PR；向下计数时，需严格配置 CVAL>CxDT。图 15-58 展示了计数器向上计数与单周期模式下 PWM 模式 B 配合的例子，计数器仅计数了一个周期，输出信号在这个周期中只输出了一个脉冲。

图 15-58 单周期模式



快速输出模式：将 TMRx_CM1 寄存器 CxOIEN 位置 1 可开启此功能，开启后 CxORAW 电平值不再在计数值与 CxDT 匹配时变化，而是在当前计数周期开始时，也就是说，比较结果被提前了，计数器值与 CxDT 寄存器的比较结果将会提前决定 CxORAW 的电平。

主定时器事件输出 (TMR9/12 支持)

当 TMR 作为主定时器时, 可选择如下信号源作为 TRGOUT 信号输出到次定时器, 选择信号为 TMRxCTRL2 寄存器 PTOS 位。

- PTOS=3'b000, TRGOUT 输出软件溢出事件 (TMRx_SWEVT 寄存器 OVFSWTR 位) 或复位事件。
- PTOS=3'b001, TRGOUT 输出计数器使能信号。
- PTOS=3'b010, TRGOUT 输出计数器溢出事件。
- PTOS=3'b011, TRGOUT 输出捕获、比较事件。
- PTOS=3'b100, TRGOUT 输出 C1ORAW 信号。
- PTOS=3'b101, TRGOUT 输出 C2ORAW 信号。

死区插入

TMR9 和 TMR12 通道 1 包含一组反向通道输出, 通过 CxCEN 使能, 通过 CxCP 配置极性。

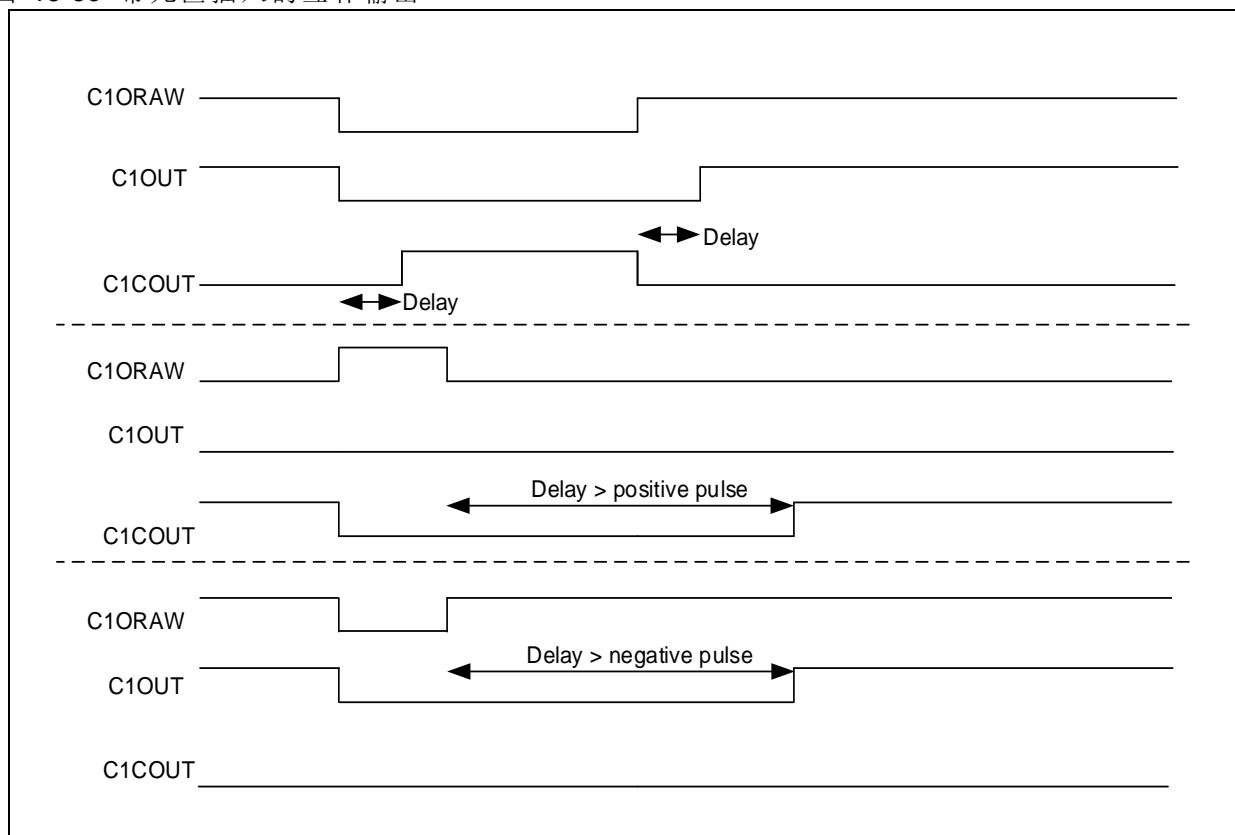
当转换为 IDLEF 状态, 即 OEN 下降到 0, 死区被激活。

将 CxEN 和 CxCEN 位置 1 后, 通过配置 DTC[7:0]死区发生器, 可插入不同时长的死区。插入死区后, CxOUT 的上升沿延迟于参考信号的上升沿; CxCOUT 的上升沿延迟于参考信号的下降沿。

如果延迟大于当前有效的输出宽度, C1OUT 和 C1COUT 不会产生相应的脉冲, 死区时间应小于有效的输出宽度。

下列图显示了 CxP=0、CxCP=0、OEN=1、CxEN=1 并且 CxCEN=1 时死区插入的例子

图 15-59 带死区插入的互补输出



15.3.3.5 TMR刹车功能

开启刹车功能后 (TMRx_BRK 寄存器 BRKEN 位置 1)，CxOUT 和 CxCOUT 由 OEN、FCSODIS、FCSOEN、CxIOS 和 CxCIOS 共同控制。但 CxOUT 和 CxCOUT 输出总是不能同时处于有效电平上的。详见表 15-11 带刹车功能的互补输出通道 CxOUT 和 CxCOUT 的控制位。

刹车信号来源可以是刹车输入引脚、时钟失效事件，刹车输入信号的极性由 BRKV 位控制。

当发生刹车事件时，有下述动作：

- OEN 位异步清零，通道输出状态由 FCSODIS 位选择。关闭 MCU 的振荡器不影响该功能。
- OEN 被清零后，通道输出电平由 CxIOS 位设定。如果 FCSODIS=0，则定时器输出使能被禁止，否则输出使能始终为高。
- 当使用互补输出时：
 - 输出最开始处于复位状态，也就是无效的状态（取决于极性）。这是异步操作，定时器有无时钟并不影响此功能。
 - 定时器的时钟如果有效，会开启死区生成功能，CxIOS 和 CxCIOS 位用来配置死区之后的电平。即使在这种情况下，CxOUT 和 CxCOUT 也不能被同时驱动到有效的电平。
 - 注意，由于 OEN 位同步逻辑，死区时间较通常情况会延长一段时间（大约 2 个 tmr_clk 的时钟周期）。
- 如果 FCSODIS=0，定时器释放使能输出，否则保持使能输出；或一旦 CxEN 与 CxCEN 之一变高时，使能输出变为高。
- 如果开启了刹车中断或 DMA 功能，刹车状态标志将置 1，并产生刹车中断或 DMA 请求。
- 如果将 AOEN 位置 1，在下一个溢出事件时 OEN 位被自动置 1。

注意：刹车输入电平有效时，OEN 不能被设置，状态标志 BRKIF 也不能被清除。

图 15-60 TMR 输出控制

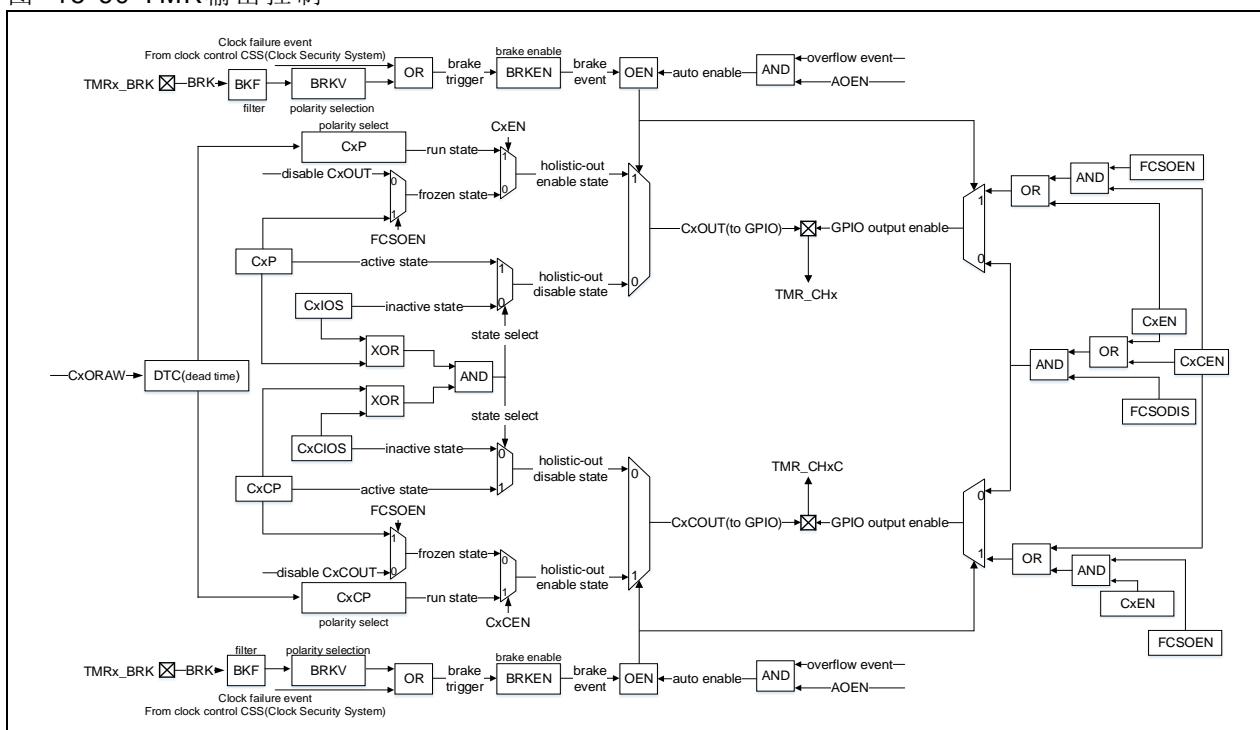
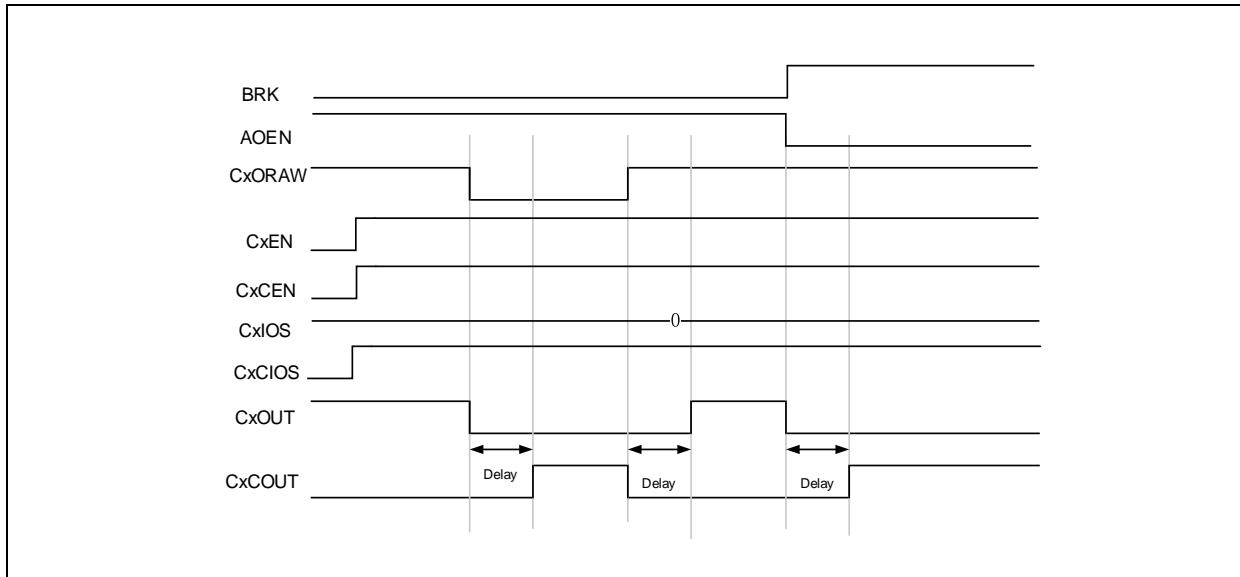


图 15-61 TMR刹车功能的例子



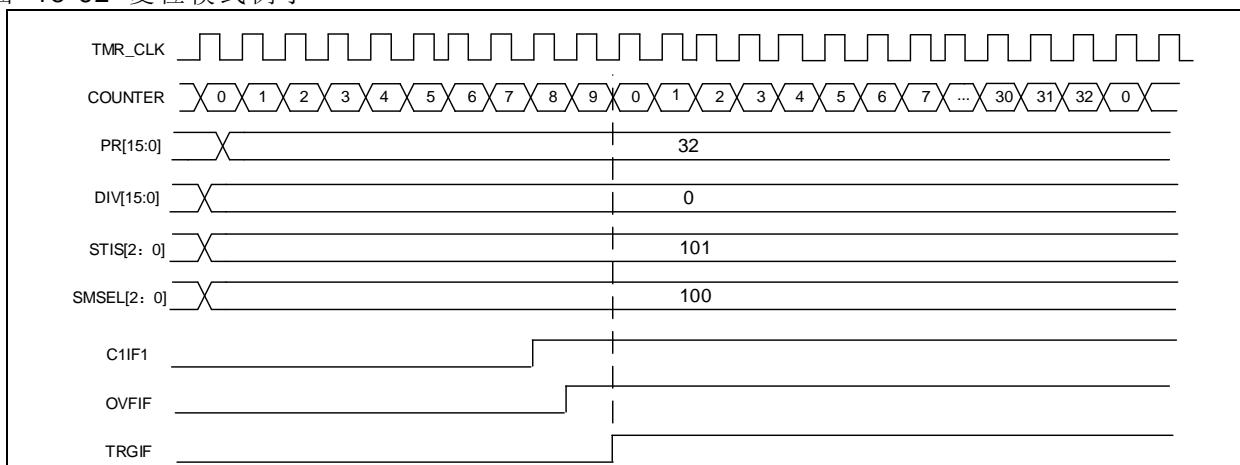
15.3.3.6 TMR同步

TMR9 和 12 可作为次定时器与主定时器由内部信号进行同步，次定时器由 `TMRx_STCTRL` 寄存器 `SMSEL[2: 0]` 位选择从模式，即次定时器的工作模式。

从模式：复位模式

选中的触发信号将复位计数器和预分频器，若 `OVFS` 位为 0，将产生一个溢出事件。

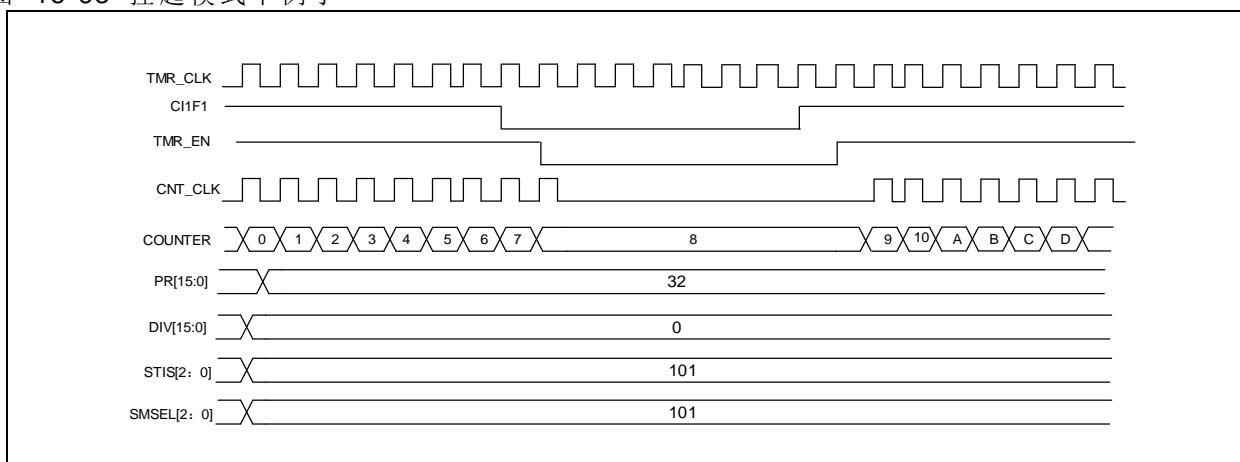
图 15-62 复位模式例子



从模式：挂起模式

挂起模式下，计数的计数和停止受选中触发输入信号控制，当触发输入为高电平时计数器开始计数；当为低电平时，计数器暂停计数。

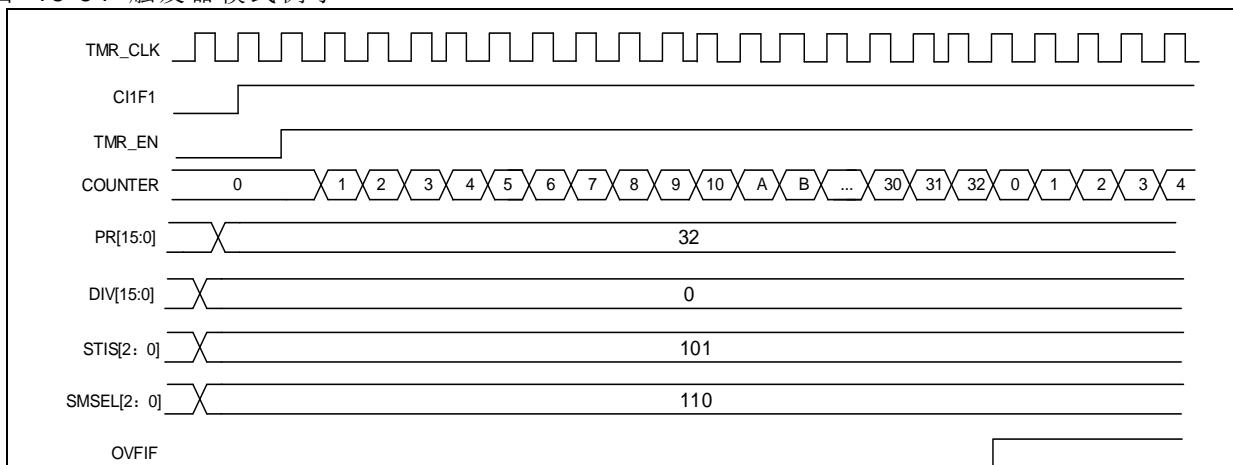
图 15-63 挂起模式下例子



从模式：触发模式

计数器将在选中的触发输入上升沿启动计数（将 TMR_EN 置 1）。

图 15-64 触发器模式例子



15.3.3.7 TMR DMA

TMR 中有溢出事件 DMA 请求、触发事件 DMA 请求、霍尔事件 DMA 请求以及通道事件 DMA 请求，通过使能 TMRx_IDEN 寄存器中相应位，开启 DMA 请求功能，产生相应事件时触发输出 DMA 请求给到 DMA 外设。

TMR DMA Burst 功能

TMR 还支持 TMR DMA Burst 功能，通过 TMRx_IDEN 寄存器使能某个事件的 DMA 请求，触发 DMA 改写多个 TMR 连续寄存器。

以溢出事件触发 TMR DMA Burst 功能为例，配置如下：

- 配置 TMRx_IDEN 寄存器中 OVFDEN 位，使能溢出事件触发 DMA 请求；
- 配置 TMRx_DMACTRL 寄存器中 DTB 位设置 Burst 传输次数；
- 配置 TMRx_DMACTRL 寄存器中 ADDR 位设置 Burst 传输的起始地址；
- 使能计数器。

以上配置的 TMR DMA Burst 传输流程如下：

当发生溢出事件时，TMR 将会产生溢出事件的 DMA 请求给到 DMA，DMA 根据请求将数据写入 TMRx_DMADT 寄存器，在 TMR 内部则会将 DMADT 位的数据写入 Burst 传输起始地址寄存器，并发送 ACK 信号给到 TMR，TMR 接收到 ACK 信号后，将清除当前 DMA 请求；TMR 检测到 Burst 传输并未全部完成，会重新置起溢出事件 DMA 请求给到 DMA，DMA 根据请求将数据再次写入 TMRx_DMADT 寄存器，在 TMR 内部则会将 DMADT 寄存器的数据写入 Burst 传输起始地址+0x4 地址寄存器，并发送 ACK 信号给到 TMR；以此往复，直到 Burst 传输最后一次操作时，检测到此时 Burst 传输已全部完成，溢出事件 DMA 请求信号将不会再被重新置起，直到下一次新的溢出事件到来。

注：使用 TMR DMA Burst 功能时，起始地址到结束地址这个区间，不应有空寄存器以及不应包含 TMRx_DMACTRL 和 TMRx_DMADT 寄存器。

15.3.3.8 调试模式

当微控制器进入调试模式（Cortex®-M4F 核心停止）时，将 DEBUG 模块中的 TMRx_PAUSE 置 1，可以使 TMRx 计数器暂停计数。

15.3.4 TMR9和TMR12寄存器描述

必须以字（32位）的方式操作这些外设寄存器。

表 15-8 TMR9和TMR12寄存器复位值

寄存器简称	基址偏移量	复位值
TMRx_CTRL1	0x00	0x0000 0000
TMRx_CTRL2	0x04	0x0000 0000
TMRx_STCTRL	0x08	0x0000 0000
TMRx_IDEN	0x0C	0x0000 0000
TMRxISTS	0x10	0x0000 0000
TMRxSWEVT	0x14	0x0000 0000
TMRx_CM1	0x18	0x0000 0000
TMRxCCTRL	0x20	0x0000 0000
TMRxCVAL	0x24	0x0000 0000
TMRxDIV	0x28	0x0000 0000
TMRx_PR	0x2C	0x0000 FFFF
TMRx_RPR	0x30	0x0000 0000
TMRxC1DT	0x34	0x0000 0000
TMRxC2DT	0x38	0x0000 0000
TMRx_BRK	0x44	0x0000 0000
TMRx_DMACTRL	0x48	0x0000 0000
TMRx_DMADT	0x4C	0x0000 0000

15.3.4.1 TMR9和TMR12控制寄存器1 (TMRx_CTRL1)

域	简称	复位值	类型	功能
位 31: 10	保留	0x00 0000	resd	保持默认值。
位 9: 8	CLKDIV	0x0	rw	时钟除频 (Clock divider) 此位用于设置数字滤波器采样频率 f_{DTS} 和定时器时钟频率 f_{CK_INT} 之间的分频比，也用于调整死区时间的时基 T_{DTS} 和定时器时钟周期 T_{CK_INT} 的分频比。 00: 无除频, $f_{DTS}=f_{CK_INT}$; 01: 2 除频, $f_{DTS}=f_{CK_INT}/2$; 10: 4 除频, $f_{DTS}=f_{CK_INT}/4$; 11: 保留。
位 7	PRBEN	0x0	rw	周期缓冲使能 (Period buffer enable) 0: 缓冲关闭; 1: 缓冲开启。
位 6: 5	TWCMSEL	0x0	rw	中央双向对齐计数模式选择 (Two-way count mode selection) 00: 单向对齐计数模式，方向由 OWCDIR 配置; 01: 中央双向对齐计数模式 1，上下交替计数，CxIF 位只在计数器向下计数时被置起; 10: 中央双向对齐计数模式 2，上下交替计数，CxIF 位只在计数器向上计数时被置起; 11: 中央双向对齐计数模式 3，上下交替计数，CxIF 位在计数器向上和向下计数时皆被置起。
位 4	OWCDIR	0x0	rw	单向对齐计数方向 (One-way count direction) 0: 向上; 1: 向下。
位 3	OCMEN	0x0	rw	单周期使能 (One cycle mode enable) 该功能用于选择溢出事件后，计数器是否停止。 0: 关闭; 1: 开启。
位 2	OVFS	0x0	rw	溢出事件源选择 (Overflow event source) 配置溢出事件或 DMA 请求来源。 0: 来源于计数器溢出、设置 OVFSWTR 位或次定时器控制器产生的溢出事件; 1: 只能来源于计数器溢出。
位 1	OVFEN	0x0	rw	溢出事件使能 (Overflow event enable) 0: 开启; 1: 关闭。
位 0	TMREN	0x0	rw	使能定时器 (TMR enable) 0: 关闭; 1: 开启。

15.3.4.2 TMR9和TMR12控制寄存器2 (TMRx_CTRL2)

域	简称	复位值	类型	功能
位 31: 12	保留	0x00 0000	resd	保持默认值。
位 11	C2CIOS	0x0	rw	通道 2 互补空闲输出状态 (Channel 2 complementary idle output state)
位 10	C2IOS	0x0	rw	通道 2 空闲输出状态 (Channel 2 idle output state)
位 9	C1CIOS	0x0	rw	通道 1 互补空闲输出状态 (Channel 1 complementary idle output state) 输出关闭 ($OEN = 0$)，死区发生后: 0: C1COUT=0; 1: C1COUT=1。
位 8	C1IOS	0x0	rw	通道 1 空闲输出状态 (Channel 1 idle output state) 输出关闭 ($OEN = 0$)，死区发生后: 0: C1OUT=0。 1: C1OUT=1。
位 7	保留	0x0	resd	保持默认值。
位 6: 4	PTOS	0x0	rw	主定时器输出信号选择 (Primary TMR output selection)

				TMRx 输出到次定时器的信号选择: 000: 软件溢出或复位; 001: 使能; 010: 溢出; 011: 比较脉冲; 100: C1ORAW 信号; 101: C2ORAW 信号; 110: C3ORAW 信号; 111: C4ORAW 信号。
位 3	DRS	0x0	rw	DMA 请求源 (DMA request source) DMA 请求来源。 0: 通道事件; 1: 溢出事件。
位 2	CCFS	0x0	rw	通道控制位刷新选择 (Channel control bit refresh select) 对具有互补输出的通道, 如果通道控制位有缓存时: 0: 通过设置 HALLSWTR 位刷新控制位; 1: 通过设置 HALLSWTR 位或 TRGIN 的上升沿刷新控制位。
位 1	保留	0x0	resd	保持默认值。
位 0	CBCTRL	0x0	rw	通道缓存控制 (Channel buffer control) 对具有互补输出的通道: 0: CxEN, CxCEN 和 CxOCTRL 位无缓存; 1: CxEN, CxCEN 和 CxOCTRL 位有缓存。 注: 当 CBCTRL="1"后, 仅当发生 HALL 事件时才会更新 CxEN, CxCEN 和 CxOCTRL 位。

15.3.4.3 TMR9和TMR12次定时器控制寄存器 (TMRx_STCTRL)

域	简称	复位值	类型	功能
位 31: 8	保留	0x00 0000	resd	保持默认值。
位 7	STS	0x0	rw	次定时器同步 (Subordinate TMR synchronization) 该位开启后, 主次定时器可实现高度同步。 0: 关闭; 1: 开启。
位 6: 4	STIS	0x0	rw	次定时器输入选择 (Subordinate TMR input selection) 用于次定时器的输入选择。 000: 内部选择 0 (IS0); 001: 内部选择 1 (IS1); 010: 内部选择 2 (IS2); 011: 内部选择 3 (IS3); 100: C1IRAW 的输入检测器 (C1INC); 101: 滤波输入 1 (C1IFP1); 110: 滤波输入 2 (C2IFP2); 111: 保留。 关于每个定时器中 ISx 的细节, 参见表 15-7。
位 3	保留	0x0	resd	保留, 保持默认值。
位 2: 0	SMSEL	0x0	rw	次定时器模式选择 (Subordinate TMR mode selection) 000: 关闭从模式; 001: 保留; 010: 保留; 011: 保留; 100: 复位模式 - TRGIN 输入上升沿时, 重新初始化计数器; 101: 挂起模式 - TRGIN 输入高电平时, 计数器计数; 110: 触发模式 - TRGIN 输入上升沿时, 产生触发事件; 111: 外部时钟模式 A - TRGIN 输入上升沿提供时钟;

15.3.4.4 TMR9和TMR12 DMA/中断使能寄存器 (TMRx_IDEN)

域	简称	复位值	类型	功能
位 31: 15	保留	0x0 0000	resd	保持默认值。
位 14	TDEN	0x0	rw	触发 DMA 请求使能 (Trigger DMA request enable) 0: 关闭; 1: 开启。
位 13	HALLDE	0x0	rw	HALL DMA 请求使能 (HALL DMA request enable) 0: 关闭; 1: 开启。
位 12: 11	保留	0x00	resd	保持默认值。
位 10	C2DEN	0x0	rw	通道 2 的 DMA 请求使能 (Channel 2 DMA request enable) 0: 关闭; 1: 开启。
位 9	C1DEN	0x0	rw	通道 1 的 DMA 请求使能 (Channel 1 DMA request enable) 0: 关闭; 1: 开启。
位 8	OVDEN	0x0	rw	溢出事件的 DMA 请求使能 (overflow event DMA request enable) 0: 关闭; 1: 开启。
位 7	BRKIE	0x0	rw	刹车中断使能 (Brake interrupt enable) 0: 关闭; 1: 开启。
位 6	TIEN	0x0	rw	触发中断使能 (Trigger interrupt enable) 0: 关闭; 1: 开启。
位 5	HALLIEN	0x0	rw	HALL 中断使能 (HALL interrupt enable) 0: 关闭; 1: 开启。
位 4: 3	保留	0x00	resd	保持默认值。
位 2	C2IEN	0x0	rw	通道 2 中断使能 (Channel 2 interrupt enable) 0: 关闭; 1: 开启。
位 1	C1IEN	0x0	rw	通道 1 中断使能 (Channel 1 interrupt enable) 0: 关闭; 1: 开启。
位 0	OVIEN	0x0	rw	溢出中断使能 (Overflow interrupt enable) 0: 关闭; 1: 开启。

15.3.4.5 TMR9和TMR12中断状态寄存器 (TMRxISTS)

域	简称	复位值	类型	功能
位 31: 11	保留	0x00 0000	resd	保持默认值。
位 10	C2RF	0x0	rw0c	通道 2 再捕获标记 (Channel 2 recapture flag) 见 C1RF 的描述。
位 9	C1RF	0x0	rw0c	通道 1 再捕获标记 (Channel 1 recapture flag) C1IF 的状态已经为'1'时是否再次发生了捕获, 由硬件置'1', 写'0'清除。 0: 无捕获发生; 1: 捕获发生。
位 8	保留	0x0	resd	保持默认值。
位 7	BRKIF	0x0	rw0c	刹车中断标记 (Brake interrupt flag) 用于标记刹车输入的电平是否有效, 由硬件置'1', 写'0'清除。 0: 无效; 1: 有效。
位 6	TRGIF	0x0	rw0c	触发中断标记 (Trigger interrupt flag) 当发生触发事件时由硬件置'1', 写'0'清除。

				0: 无触发事件发生; 1: 发生触发事件。 触发事件: 在 TRGIN 接收到有效边沿, 或挂起模式下接收到任意边沿。
位 5	HALLIF	0x0	rw0c	HALL 中断标记 (HALL interrupt flag) 当发生触发事件时由硬件置'1', 写'0'清除。 0: 无 HALL 事件发生; 1: 发生 HALL 事件。 HALL 事件: CxEN、CxCEN、CxOCTRL 已被更新。
位 4: 3	保留	0x0	resd	保持默认值。
位 2	C2IF	0x0	rw0c	通道 2 中断标记 (Channel 2 interrupt flag) 见 C1IF 的描述。
位 1	C1IF	0x0	rw0c	通道 1 中断标记 (Channel 1 interrupt flag) 若通道 1 为输入模式时: 捕获事件发生时由硬件置'1', 由软件清'0'或读 TMRx_C1DT 清'0'. 0: 无捕获事件发生; 1: 发生捕获事件。 若通道 1 为输出模式时: 比较事件发生时由硬件置'1', 由软件清'0'. 0: 无比较事件发生; 1: 发生比较事件。
位 0	OVFIF	0x0	rw0c	溢出中断标记 (Overflow interrupt flag) 当溢出事件发生时由硬件置'1', 由软件清'0'. 0: 无溢出事件发生; 1: 发生溢出事件, 若 TMRx_CTRL1 的 OVFEN=0、 OVFS=0 时: - 当 TMRx_SWEVT 寄存器的 OVFSWTR=1 时产生溢出事件; - 当计数值 CVAL 被触发事件重初始化时产生溢出事件。

15.3.4.6 TMR9和TMR12软件事件寄存器 (TMRx_SWEVT)

域	简称	复位值	类型	功能
位 31: 8	保留	0x00 0000	resd	保持默认值。
位 7	BRKSWTR	0x0	wo	软件触发刹车事件 (Brake event triggered by software) 通过软件触发一个刹车事件。 0: 无作用; 1: 制造一个刹车事件。
位 6	TRGSWTR	0x0	wo	软件触发触发事件 (Trigger event triggered by software) 通过软件触发一个触发事件。 0: 无作用; 1: 制造一个触发事件。
位 5	HALLSWTR	0x0	wo	软件触发 HALL 事件 (HALL event triggered by software) 通过软件产生一个 HALL 事件。 0: 无作用; 1: 产生一个 HALL 事件。 注: 该位只对拥有互补输出的通道有效。
位 4: 3	保留	0x0	resd	保持默认值。
位 2	C2SWTR	0x0	wo	软件触发通道 2 事件 (Channel 2 event triggered by software) 见 C1M 的描述。
位 1	C1SWTR	0x0	wo	软件触发通道 1 事件 (Channel 1 event triggered by software) 通过软件触发一个通道 1 事件。 0: 无作用; 1: 制造一个通道 1 事件。
位 0	OVFSWTR	0x0	wo	软件触发溢出事件 (Overflow event triggered by software)

通过软件触发一个溢出事件。

0: 无作用;

1: 制造一个溢出事件。

15.3.4.7 TMR9和TMR12通道模式寄存器1 (TMRx_CM1)

通道可用于输入（捕获模式）或输出（比较模式），通道的方向由相应的 CxC 定义。该寄存器其它位的作用在输入和输出模式下不同。CxOx 描述了通道在输出模式下的功能，CxIx 描述了通道在输出模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

输出比较模式：

域	简称	复位值	类型	功能
位 31: 15	保留	0x0 0000	resd	保持默认值。
位 14: 12	C2OCTRL	0x0	rw	通道 2 输出控制 (Channel 2 output control)
位 11	C2OBEN	0x0	rw	通道 2 输出缓存使能 (Channel 2 output buffer enable)
位 10	C2OIEN	0x0	rw	通道 2 输出立即使能 (Channel 2 output immediately enable)
位 9: 8	C2C	0x0	rw	通道 2 配置 (Channel 2 configure) 当 C2EN=0'时，这些位用于选择通道 2 为输出或输入，以及输入时的映射选择： 00: 输出； 01: 输入，C2IN 映射在 C2IFP2 上； 10: 输入，C2IN 映射在 C1IFP2 上； 11: 输入，C2IN 映射在 STCI 上，只有在 STIS 选择内部触发输入时才工作。
位 7	保留	0x0	resd	保持默认值
位 6: 4	C1OCTRL	0x0	rw	通道 1 输出控制 (Channel 1 output control) 这些位用于设置原始信号 C1ORAW 的工作状态。 000: 断开。断开 C1ORAW 到 C1OUT 的输出； 001: 当 TMRx_CVAL=TMRx_C1DT 时，设置 C1ORAW 为高。 010: 当 TMRx_CVAL=TMRx_C1DT 时，设置 C1ORAW 为低。 011: 当 TMRx_CVAL=TMRx_C1DT 时，切换 C1ORAW 的电平。 100: 固定 C1ORAW 为低。 101: 固定 C1ORAW 为高。 110: PWM 模式 A —OWCDIR=0, 若 TMRx_C1DT>TMRx_CVAL 时设置 C1ORAW 为高，否则为低； —OWCDIR=1, 若 TMRx_C1DT<TMRx_CVAL 时设置 C1ORAW 为低，否则为高。 111: PWM 模式 B —OWCDIR=0, 若 TMRx_C1DT>TMRx_CVAL 时设置 C1ORAW 为低，否则为高； —OWCDIR=1, 若 TMRx_C1DT<TMRx_CVAL 时设置 C1ORAW 为高，否则为低。 注：除'000'外，其余配置下 C1OUT 将连接到 C1ORAW，C1OUT 的输出电平除了会根据 C1ORAW 变化外，还与 CCTRL 所配置的输出极性有关。
位 3	C1OBEN	0x0	rw	通道 1 输出缓存使能 (Channel 1 output buffer enable) 0: 关闭 TMRx_C1DT 的缓存功能，写入 TMRx_C1DT 的内容会立即生效。 1: 启用 TMRx_C1DT 的缓存功能，写入 TMRx_C1DT 的内容将保存到缓存寄存器中，当发生溢出事件时再更新到 TMRx_C1DT 中。
位 2	C1OIEN	0x0	rw	通道 1 输出立即使能 (Channel 1 output immediately enable) 在 PWM 模式 A 或模式 B 下，该位能够缩短触发事件到通道 1 的输出响应间的时间。 0: 需要比较 CVAL 与 C1DT 的值之后再产生输出。

位 1: 0	C1C	0x0	rw	1: 无需比较 CVAL 与 C1DT 的值, 当发生触发事件时立即产生输出。 通道 1 配置 (Channel 1 configure) 当 C1EN=’0’时, 这些位用于选择通道 1 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C1IN 映射在 C1IFP1 上; 10: 输入, C1IN 映射在 C2IFP1 上; 11: 输入, C1IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。
输入模式:				
域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 12	C2DF	0x0	rw	通道 2 滤波器 (Channel 2 digital filter)
位 11: 10	C2IDIV	0x0	rw	通道 2 分频系数 (Channel 2 input divider)
位 9: 8	C2C	0x0	rw	通道 2 配置 (Channel 2 configure) 当 C2EN=’0’时, 这些位用于选择通道 2 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C2IN 映射在 C2IFP2 上; 10: 输入, C2IN 映射在 C1IFP2 上; 11: 输入, C2IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。
位 7: 4	C1DF	0x0	rw	通道 1 滤波器 (Channel 1 digital filter) 这些位用于配置通道 1 的滤波器。滤波的个数为 N, 则表示发生了 N 次采样事件后输入边沿才能通过滤波器: 0000: 无滤波器, 以 f_{DTS} 采样 0001: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=2 0010: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=4 0011: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=8 0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=6 0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=8 0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=6 0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=8 1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=6 1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=8 1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=5 1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=6 1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=8 1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=5 1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=6 1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=8
位 3: 2	C1IDIV	0x0	rw	通道 1 分频系数 (Channel 1 input divider) 这些位定义了通道 1 的分频系数。 00: 不分频, 每一个有效的边沿都会产生一次输入; 01: 每 2 个有效的边沿产生一次输入; 10: 每 4 个有效的边沿产生一次输入; 11: 每 8 个有效的边沿产生一次输入。 注: C1EN=’0’时, 分频系数复位。
位 1: 0	C1C	0x0	rw	通道 1 配置 (Channel 1 configure) 当 C1EN=’0’时, 这些位用于选择通道 1 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C1IN 映射在 C1IFP1 上; 10: 输入, C1IN 映射在 C2IFP1 上; 11: 输入, C1IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。

15.3.4.8 TMR9和TMR12通道控制寄存器 (TMRx_CCTRL)

域	简称	复位值	类型	功能
位 31: 8	保留	0x00 0000	resd	保持默认值。
位 7	C2CP	0x0	rw	通道 2 互补极性 (Channel 2 complementary polarity) 见 C1CP 的描述。
位 6	C2CEN	0x0	rw	通道 2 互补使能 (Channel 2 complementary enable) 见 C1CEN 的描述。
位 5	C2P	0x0	rw	通道 2 极性 (Channel 2 polarity) 见 C1P 的描述。
位 4	C2EN	0x0	rw	通道 2 使能 (Channel 2 enable) 见 C1EN 的描述。
位 3	C1CP	0x0	rw	通道 1 互补极性 (Channel 1 complementary polarity) 0: C1COUT 的有效电平为高 1: C1COUT 的有效电平为低
位 2	C1CEN	0x0	rw	通道 1 互补使能 (Channel 1 complementary enable) 0: 禁止输出; 1: 使能输出。
位 1	C1P	0x0	rw	通道 1 极性 (Channel 1 polarity) 通道 1 配置为输出: 0: C1OUT 的有效电平为高 1: C1OUT 的有效电平为低 通道 1 配置为输入: C1CP/C1P 位共同定义输入信号有效沿。 00: C1IN 的有效边沿为上升沿; 作为外部触发使用时, C1IN 不反相。 01: C1IN 的有效边沿为下降沿; 作为外部触发使用时, C1IN 反相。 10: 保留 11: C1IN 的有效边沿为上升沿和下降沿; 作为外部触发 使用时, C1IN 不反相。
位 0	C1EN	0x0	rw	通道 1 使能 (Channel 1 enable) 0: 禁止输入或输出; 1: 使能输入或输出。

表 15-9 带刹车功能的互补输出通道OCx和OCxN的控制位

控制位					输出状态 (1)	
OEN 位	FCSODIS 位	FCSOEN 位	CxEN 位	CxCEN 位	CxOUT 输出状态	CxCOUT 输出状态
1	X	0	0	0	输出禁止 (与定时器断开) CxOUT=0, CxEN=0	输出禁止 (与定时器断开) CxCOUT=0, CxCEN=0
		0	0	1	输出禁止 (与定时器断开) CxOUT=0, CxEN=0	CxORAW + 极性, CxCOUT=CxORAW xor CxCP, CxCEN=1
		0	1	0	CxORAW+极性, CxOUT= CxORAW xor CxP, CxEN=1	输出禁止 (与定时器断开) CxCOUT=0, CxCEN=0
		0	1	1	CxORAW+极性+死区, CxEN=1	CxORAW 反相+极性+死区, CxCEN=1
		1	0	0	输出禁止 (与定时器断开) CxOUT=CxP, CxEN=0	输出禁止 (与定时器断开) CxCOUT=CxCP, CxCEN=0
		1	0	1	关闭状态 (输出使能且为无效电平) CxOUT=CxP, CxEN=1	CxORAW + 极性, CxCOUT=CxORAW xor CxCP, CxCEN=1
		1	1	0	CxORAW + 极性, CxOUT= CxORAW xor CxP, CxEN=1	关闭状态 (输出使能且为无效电平) CxCOUT=CxCP, CxCEN=1
		1	1	1	CxORAW+极性+死区, CxEN=1	CxORAW 反相+极性+死区, CxCEN=1
0	0	X	0	0	输出禁止 (与定时器断开) 异步地: CxOUT=CxP , CxEN=0 , CxCOUT=CxCP , CxEN=0; 若时钟存在: 经过一个死区时间后 CxOUT=CxIOS, CxCOUT=CxCIOS, 假设 CxIOS 与 CxCIOS 并不都对应 CxOUT 和 CxCOUT 的有效电平。	
	0		0	1		
	0		1	0		
	0		1	1		
	1		0	0	关闭状态 (输出使能且为无效电平) 异步地: CxOUT =CxP, CxEN=1, CxCOUT=CxCP , CxEN=1; 若时钟存在: 经过一个死区 时间后 CxOUT =C1IOS, CxCOUT=CxCIOS, 假设 CxIOS 与 CxCIOS 并不都对应 CxOUT 和 CxCOUT 的有效电平。	

注意: 如果一个通道的 2 个输出都没有使用 ($CxEN = CxCEN = 0$) , 那么 $CxIOS$, $CxCIOS$, CxP 和 $CxCP$ 都必须清零。

注意: 连接到标准 CxOUT 通道的外部 I/O 引脚状态, 取决于 CxOUT 通道状态和 GPIO 以及 IOMUX 寄存器。

15.3.4.9 TMR9和TMR12计数值寄存器 (TMRx_CVAL)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	CVAL	0x0000	rw	计数值 (Counter value)

15.3.4.10 TMR9和TMR12预分频器寄存器 (TMRx_DIV)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	DIV	0x0000	rw	分频系数 (Divider value) 计数器时钟频率 $f_{CK_CNT} = f_{TMR_CLK} / (DIV[15: 0]+1)$ 。 DIV 为溢出事件发生时写入的分频系数。

15.3.4.11 TMR9和TMR12周期寄存器 (TMRx_PR)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	PR	0xFFFF	rw	周期值 (Period value) 定时器计数的周期值。当周期值为 0 时，定时器不工作。

15.3.4.12 TMR9和TMR12重复周期寄存器 (TMRx_RPR)

域	简称	复位值	类型	功能
位 31: 8	保留	0x00 0000	resd	保持默认值。
位 7: 0	RPR	0x00	rw	重复周期的次数 (Repetition of period value) 这些位用于减慢溢出事件发生的速率，当重复周期的次数减为 0 时才会发生溢出事件。

15.3.4.13 TMR9和TMR12通道1数据寄存器 (TMRx_C1DT)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值
位 15: 0	C1DT	0x0000	rw	通道 1 数据寄存器值 (Channel 1 data register) 若通道配置为输入： C1DT 是前一次通道 1 输入事件 (C1IN) 所保存的 CVAL。 若通道 1 配置为输出： C1DT 是将要和 CVAL 进行比较的值，写入的值是否会立即生效取决于输出缓存使能位 (C1OBEN)，并根据设置在 C1OUT 上产生相应的输出。

15.3.4.14 TMR9和TMR12通道2数据寄存器 (TMRx_C2DT)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值
位 15: 0	C2DT	0x0000	rw	通道 2 数据寄存器值 (Channel 2 data register) 若通道 2 配置为输入： C2DT 是前一次通道 2 输入事件 (C2IN) 所保存的 CVAL。 若通道 2 配置为输出： C2DT 是将要和 CVAL 进行比较的值，写入的值是否会立即生效取决于输出缓存使能位 (C2OBEN)，并根据设置在 C2OUT 上产生相应的输出。

15.3.4.15 TMR9和TMR12刹车寄存器 (TMRx_BRK)

域	简称	复位值	类型	功能
位 31: 20	保留	0x000	resd	保持默认值。
位 19:16	BRKF	0x0	rw	<p>刹车输入滤波 (brake input filter) 这些位用于配置刹车输入的滤波器。滤波的个数为 N，则表示发生了 N 次采样事件后输入边沿才能通过滤波器：</p> <ul style="list-style-type: none"> 0000: 无滤波器，以 f_{DTS} 采样 0001: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=2 0010: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=4 0011: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=8 0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=6 0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=8 0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=6 0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=8 1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=6 1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=8 1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=5 1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=6 1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=8 1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=5 1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=6 1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=8
位 15	OEN	0x0	rw	<p>输出使能 (Output enable) 对配置为输出的通道，该位用于使能 CxOUT 和 CxCOUT 的输出。 0: 关闭； 1: 开启。</p>
位 14	AOEN	0x0	rw	<p>输出自动使能 (Automatic output enable) 用于溢出事件时将 OEN 自动置'1' 0: 关闭； 1: 开启</p>
位 13	BRKV	0x0	rw	<p>刹车输入信号的有效性 (Brake input validity) 用于选择刹车输入信号的输入有效电平： 0: 低电平； 1: 高电平。</p>
位 12	BRKEN	0x0	rw	<p>刹车功能使能 (Brake enable) 用于开启刹车功能。 0: 关闭； 1: 开启。</p>
位 11	FCSOEN	0x0	rw	<p>总输出开时的冻结状态 (Frozen channel status when holistic output enable) 该位用于配置具有互补输出的通道，在定时器不工作且 OEN=1 时的通道状态。 0: 关闭 CxOUT/CxCOUT 输出； 1: 开启 CxOUT/CxCOUT 输出，输出为无效电平。</p>
位 10	FCSODIS	0x0	rw	<p>总输出关时的冻结状态 (Frozen channel status when holistic output disable) 该位用于配置具有互补输出的通道，在定时器不工作且 OEN=0 时的通道状态。 0 : 关闭 CxOUT/CxCOUT 输出； 1 : 开启 CxOUT/CxCOUT 输出，输出为空闲电平。</p>
位 9: 8	WPC	0x0	rw	<p>写保护配置 (Write protected configuration) 该位用于配置写保护。 00: 写保护关闭； 01: 3 级写保护，以下位受写保护： TMRx_BRK: BRKF、DTC、BRKEN、BRKV 和 AOEN TMRx_CTRL2: CxIOS 和 CxCIOs 10: 2 级写保护，除 3 级写保护的内容外，以下位也受写保护： TMRx_CCTRL: CxP 和 CxCP TMRx_BRK: FCSODIS 和 FCSOEN</p>

11: 1 级写保护, 除 2 级写保护的内容外, 以下位也受写保护:

TMRx_CM1: CxOCTRL 和 CxOBEN

注: WPC>0 时将无法再次被修改, 直到系统复位。

死区配置 (Dead-time configuration)

这些位用于配置死区时间。取 DTC[7: 0]的高 3 位为功能选择位:

位 7: 0	DTC	0x00	rw	0xx: DT = DTC [7: 0] * TDTS; 10x: DT = (64+ DTC [5: 0]) * TDTS * 2; 110: DT = (32+ DTC [4: 0]) * TDTS * 8; 111: DT = (32+ DTC [4: 0]) * TDTS * 16;
--------	-----	------	----	---

注意: 根据锁定设置, BRKF、AOEN、BRKV、BRKEN、FCSODIS、FCSCOEN 和 DTC[7: 0]位均可被写保护, 有必要在第一次写入 TMRx_BRK 寄存器时对它们进行配置。

15.3.4.16 TMR9和TMR12 DMA控制寄存器 (TMRx_DMACTRL)

域	简称	复位值	类型	功能
位 31: 13	保留	0x0 0000	resd	保持默认值。
位 12: 8	DTB	0x00	rw	DMA 传输字节 (DMA transfer bytes) 这些位定义了传输的字节个数: 00000: 1 个字节 00001: 2 个字节 00010: 3 个字节 00011: 4 个字节 10000: 17 个字节 10001: 18 个字节
位 7: 5	保留	0x0	resd	保持默认值。
位 4: 0	ADDR	0x00	rw	DMA 传输地址偏移 (DMA transfer address offset) ADDR 定义了从 TMRx_CTRL1 所在地址开始的偏移量: 00000: TMRx_CTRL1, 00001: TMRx_CTRL2, 00010: TMRx_STCTRL,

15.3.4.17 TMR9和TMR12 DMA数据寄存器 (TMRx_DMADT)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	DMADT	0x0000	rw	DMA 传输的数据寄存器 (DMA data register) 通过对 DMADT 寄存器的读写能够实现对任意 TMR 寄存器的操作, 其操作的寄存器地址范围是: TMRx 外设地址 + ADDR*4 至 TMRx 外设地址 + ADDR*4 + DTB*4。

15.3.5 TMR10、TMR11、TMR13和TMR14寄存器描述

必须以字（32位）的方式操作这些外设寄存器。

表 15-10 TMR10、TMR11、TMR13和TMR14寄存器和复位值

寄存器简称	基址偏移量	复位值
TMRx_CTRL1	0x00	0x0000 0000
TMRx_CTRL2	0x04	0x0000 0000
TMRx_IDEN	0x0C	0x0000 0000
TMRxISTS	0x10	0x0000 0000
TMRx_SW EVT	0x14	0x0000 0000
TMRx_CM1	0x18	0x0000 0000
TMRx_CCTRL	0x20	0x0000 0000
TMRx_CVAL	0x24	0x0000 0000
TMRx_DIV	0x28	0x0000 0000
TMRx_PR	0x2C	0x0000 FFFF
TMRx_RPR	0x30	0x0000 0000
TMRx_C1DT	0x34	0x0000 0000
TMRx_BRK	0x44	0x0000 0000
TMRx_DMACTRL	0x48	0x0000 0000
TMRx_DMADT	0x4C	0x0000 0000
TMR14_RMP	0x50	0x0000 0000

15.3.5.1 TMR10、TMR11、TMR13和TMR14控制寄存器1 (TMRx_CTRL1)

域	简称	复位值	类型	功能
位 31: 10	保留	0x00 0000	resd	保持默认值。
位 9: 8	CLKDIV	0x0	rw	时钟除频 (Clock divider) 此位用于设置数字滤波器采样频率 f_{DTS} 和定时器时钟频率 f_{CK_INT} 之间的分频比，也用于调整死区时间的时基 T_{DTS} 和定时器时钟周期 T_{CK_INT} 的分频比。 00: 无除频, $f_{DTS}=f_{CK_INT}$; 01: 2 除频, $f_{DTS}=f_{CK_INT}/2$; 10: 4 除频, $f_{DTS}=f_{CK_INT}/4$; 11: 保留。
位 7	PRBEN	0x0	rw	周期缓冲使能 (Period buffer enable) 0: 缓冲关闭; 1: 缓冲开启。
位 6: 5	TWCMSEL	0x0	rw	中央双向对齐计数模式选择 (Two-way count mode selection) 00: 单向对齐计数模式，方向由 OWCDIR 配置; 01: 中央双向对齐计数模式 1，上下交替计数，CxIF 位只在计数器向下计数时被置起; 10: 中央双向对齐计数模式 2，上下交替计数，CxIF 位只在计数器向上计数时被置起; 11: 中央双向对齐计数模式 3，上下交替计数，CxIF 位在计数器向上和向下计数时皆被置起。
位 4	OWCDIR	0x0	rw	单向计数方向 (One-way count direction) 0: 向上; 1: 向下。
位 3	OCMEN	0x0	rw	单周期使能 (One cycle mode enable) 该功能用于选择溢出事件后，计数器是否停止。 0: 关闭; 1: 开启。
位 2	OVFS	0x0	rw	溢出事件源选择 (Overflow event source) 配置溢出事件或 DMA 请求来源。 0: 来源于计数器溢出、设置 OVFSWTR 位或次定时器控制器产生的溢出事件; 1: 只能来源于计数器溢出。
位 1	OVFEN	0x0	rw	溢出事件使能 (Overflow event enable) 0: 开启; 1: 关闭。
位 0	TMREN	0x0	rw	使能定时器 (TMR enable) 0: 关闭; 1: 开启。

15.3.5.2 TMR10、TMR11、TMR13和TMR14控制寄存器2 (TMRx_CTRL2)

域	简称	复位值	类型	功能
位 31: 10	保留	0x00 0000	resd	保持默认值。
位 9	C1CIOS	0x0	rw	通道 1 互补空闲输出状态 (Channel 1 complementary idle output state) 输出关闭 ($OEN = 0$)，死区发生后： 0: C1COUT=0; 1: C1COUT=1。
位 8	C1IOS	0x0	rw	通道 1 空闲输出状态 (Channel 1 idle output state) 输出关闭 ($HOEN = 0$)，死区发生后： 0: C1OUT=0。 1: C1OUT=1。
位 7: 4	保留	0x0	resd	保持默认值。
位 3	DRS	0x0	rw	DMA 请求源 (DMA request source)

				DMA 请求来源。 0: 通道事件; 1: 溢出事件。
位 2	CCFS	0x0	rw	通道控制位刷新选择 (Channel control bit flash select) 对具有互补输出的通道, 如果通道控制位有缓存时: 0: 通过设置 HALLSWTR 位刷新控制位; 1: 通过设置 HALLSWTR 位或 TRGIN 的上升沿刷新控制位。
位 1	保留	0x0	resd	保持默认值。
位 0	CBCTRL	0x0	rw	通道缓存控制 (Channel buffer control) 对具有互补输出的通道: 0: CxEN, CxCEN 和 CxOCTRL 位无缓存; 1: CxEN, CxCEN 和 CxOCTRL 位有缓存。 注: 当 CBCTRL="1"后, 仅当发生 HALL 事件时才会更新 CxEN, CxCEN 和 CxOCTRL 位。

15.3.5.3 TMR10、TMR11、TMR13和TMR14 DMA/中断使能寄存器 (TMRx_IDEN)

域	简称	复位值	类型	功能
位 31: 10	保留	0x00 0000	resd	保持默认值。
位 9	C1DEN	0x0	rw	通道 1 的 DMA 请求使能 (Channel 1 DMA request enable) 0: 关闭; 1: 开启。
位 8	OVFDEN	0x0	rw	溢出事件的 DMA 请求使能 (overflow event DMA request enable) 0: 关闭; 1: 开启。
位 7	BRKIE	0x0	rw	刹车中断使能 (Brake interrupt enable) 0: 关闭; 1: 开启。
位 6	保留	0x0	resd	保持默认值。
位 5	HALLIEN	0x0	rw	HALL 中断使能 (HALL interrupt enable) 0: 关闭; 1: 开启。
位 4: 2	保留	0x0	resd	保持默认值。
位 1	C1IEN	0x0	rw	通道 1 中断使能 (Channel 1 interrupt enable) 0: 关闭; 1: 开启。
位 0	OVFIEN	0x0	rw	溢出中断使能 (Overflow interrupt enable) 0: 关闭; 1: 开启。

15.3.5.4 TMR10、TMR11、TMR13和TMR14 中断状态寄存器 (TMRx_ISTS)

域	简称	复位值	类型	功能
位 31: 10	保留	0x00 0000	resd	保持默认值。
位 9	C1RF	0x0	rw0c	通道 1 再捕获标记 (Channel 1 recapture flag) C1IF 的状态已经为'1'时是否再次发生了捕获, 由硬件置'1', 写'0'清除。 0: 无捕获发生; 1: 捕获发生。
位 8	保留	0x0	resd	保持默认值。
位 7	BRKIF	0x0	rw0c	刹车中断标记 (Brake interrupt flag) 用于标记刹车输入的电平是否有效, 由硬件置'1', 写'0'清除。 0: 无效; 1: 有效。
位 6	保留	0x0	resd	保持默认值。

				HALL 中断标记 (HALL interrupt flag) 当发生触发事件时由硬件置'1'，写'0'清除。 0: 无 HALL 事件发生； 1: 发生 HALL 事件。 HALL 事件: CxEN、CxLEN、CxOCTRL 已被更新。
位 5	HALLIF	0x0	rw0c	通道 1 中断标记 (Channel 1 interrupt flag) 若通道 1 为输入模式时： 捕获事件发生时由硬件置'1'，由软件清'0'或读 TMR1_C1DT 清'0'。 0: 无捕获事件发生； 1: 发生捕获事件。 若通道 1 为输出模式时： 比较事件发生时由硬件置'1'，由软件清'0'。 0: 无比较事件发生； 1: 发生比较事件。
位 4: 2	保留	0x0	resd	溢出中断标记 (Overflow interrupt flag) 当溢出事件发生时由硬件置'1'，由软件清'0'。 0: 无溢出事件发生； 1: 发生溢出事件，若 TMR1_CTRL1 的 OVFS=0、 OVFS=0 时： - 当 TMR1_SWEVT 寄存器的 OVFSWTR=1 时产生溢出事件； - 当计数值 CVAL 被触发事件重初始化时产生溢出事件。
位 1	C1IF	0x0	rw0c	
位 0	OVFIF	0x0	rw0c	

15.3.5.5 TMR10、TMR11、TMR13和TMR14软件事件寄存器 (TMRx_SWEVT)

域	简称	复位值	类型	功能
位 31: 8	保留	0x00 0000	resd	保持默认值。
位 7	BRKSWTR	0x0	wo	软件触发刹车事件 (Brake event triggered by software) 通过软件触发一个刹车事件。 0: 无作用； 1: 制造一个刹车事件。
位 6	保留	0x0	resd	保持默认值。
位 5	HALLSWTR	0x0	wo	软件触发 HALL 事件 (HALL event triggered by software) 通过软件产生一个 HALL 事件。 0: 无作用； 1: 产生一个 HALL 事件。 注：该位只对拥有互补输出的通道有效。
位 4: 2	保留	0x0	resd	保持默认值。
位 1	C1SWTR	0x0	wo	C1SWTR：软件触发通道 1 事件 (Channel 1 event triggered by software) 通过软件触发一个通道 1 事件。 0: 无作用； 1: 制造一个通道 1 事件。
位 0	OVFSWTR	0x0	wo	软件触发溢出事件 (Overflow event triggered by software) 通过软件触发一个溢出事件。 0: 无作用； 1: 制造一个溢出事件。

15.3.5.6 TMR10、TMR11、TMR13和TMR14通道模式寄存器1 (TMRx_CM1)

通道可用于输入（捕获模式）或输出（比较模式），通道的方向由相应的 CxC 定义。该寄存器其它位的作用在输入和输出模式下不同。CxOx 描述了通道在输出模式下的功能，CxIx 描述了通道在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

输出比较模式：

域	简称	复位值	类型	功能
位 31: 7	保留	0x000 0000	resd	保持默认值。
位 6: 4	C1OCTRL	0x0	rw	<p>通道 1 输出控制 (Channel 1 output control) 这些位用于设置原始信号 C1ORAW 的工作状态。 000: 断开 C1ORAW 到 C1OUT 的输出； 001: 当 TMRx_CVAL=TMRx_C1DT 时，设置 C1ORAW 为高。 010: 当 TMRx_CVAL=TMRx_C1DT 时，设置 C1ORAW 为低。 011: 当 TMRx_CVAL=TMRx_C1DT 时，切换 C1ORAW 的电平。 100: 固定 C1ORAW 为低。 101: 固定 C1ORAW 为高。 110: PWM 模式 A —OWCDIR=0, 若 TMRx_C1DT>TMRx_CVAL 时设置 C1ORAW 为高，否则为低； —OWCDIR=1, 若 TMRx_C1DT<TMRx_CVAL 时设置 C1ORAW 为低，否则为高。 111: PWM 模式 B —OWCDIR=0, 若 TMRx_C1DT>TMRx_CVAL 时设置 C1ORAW 为低，否则为高； —OWCDIR=1, 若 TMRx_C1DT<TMRx_CVAL 时设置 C1ORAW 为高，否则为低。 注：除'000'外，其余配置下 C1OUT 将连接到 C1ORAW，C1OUT 的输出电平除了会根据 C1ORAW 变化外，还与 CCTRL 所配置的输出极性有关。</p>
位 3	C1OBEN	0x0	rw	<p>通道 1 输出缓存使能 (Channel 1 output buffer enable) 0: 关闭 TMRx_C1DT 的缓存功能，写入 TMRx_C1DT 的内容会立即生效。 1: 启用 TMRx_C1DT 的缓存功能，写入 TMRx_C1DT 的内容将保存到缓存寄存器中，当发生溢出事件时再更新到 TMRx_C1DT 中。</p>
位 2	C1OIEN	0x0	rw	<p>通道 1 输出立即使能 (Channel 1 output immediately enable) 在 PWM 模式 A 或模式 B 下，该位能够缩短触发事件到通道 1 的输出响应间的时间。 0: 需要比较 CVAL 与 C1DT 的值之后再产生输出。 1: 无需比较 CVAL 与 C1DT 的值，当发生触发事件时立即产生输出。</p>
位 1: 0	C1C	0x0	rw	<p>通道 1 配置 (Channel 1 configure) 当 C1EN='0'时，这些位用于选择通道 1 为输出或输入，以及输入时的映射选择： 00: 输出； 01: 输入，C1IN 映射在 C1IFP1 上； 10: 保留； 11: 保留。</p>

输入模式:

域	简称	复位值	类型	功能
位 31: 8	保留	0x00 0000	resd	保持默认值。
位 7: 4	C1DF	0x0	rw	通道 1 滤波器 (Channel 1 digital filter) 这些位用于配置通道 1 的滤波器。滤波的个数为 N，则表示发生了 N 次采样事件后输入边沿才能通过滤波器： 0000: 无滤波器，以 f_{DTS} 采样 0001: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=2 0010: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=4 0011: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=8 0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=6 0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=8 0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=6 0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=8 1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=6 1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=8 1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=5 1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=6 1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=8 1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=5 1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=6 1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=8
位 3: 2	C1IDIV	0x0	rw	通道 1 分频系数 (Channel 1 input divider) 这些位定义了通道 1 的分频系数。 00: 不分频，每一个有效的边沿都会产生一次输入； 01: 每 2 个有效的边沿产生一次输入； 10: 每 4 个有效的边沿产生一次输入； 11: 每 8 个有效的边沿产生一次输入。 注：C1EN='0'时，分频系数复位。
位 1: 0	C1C	0x0	rw	通道 1 配置 (Channel 1 configure) 当 C1EN='0'时，这些位用于选择通道 1 为输出或输入，以及输入时的映射选择： 00: 输出； 01: 输入，C1IN 映射在 C1IFP1 上； 10: 保留； 11: 保留。

15.3.5.7 TMR10、TMR11、TMR13和TMR14通道控制寄存器 (TMRx_CCTRL)

域	简称	复位值	类型	功能
位 31: 4	保留	0x0000 0000	resd	保持默认值。
位 3	C1CP	0x0	rw	通道 1 互补极性 (Channel 1 complementary polarity) 0: C1COUT 的有效电平为高 1: C1COUT 的有效电平为低
位 2	C1CEN	0x0	rw	通道 1 互补使能 (Channel 1 complementary enable) 0: 禁止输出； 1: 使能输出。
位 1	C1P	0x0	rw	通道 1 极性 (Channel 1 polarity) 通道 1 配置为输出： 0: C1OUT 的有效电平为高 1: C1OUT 的有效电平为低 通道 1 配置为输入： C1CP/C1P 位共同定义输入信号有效沿。 00: C1IN 的有效边沿为上升沿；作为外部触发使用时，C1IN 不反相。 01: C1IN 的有效边沿为下降沿；作为外部触发使用时，C1IN 反相。 10: 保留 11: C1IN 的有效边沿为上升沿和下降沿；作为外部触发使用时，C1IN 不反相。

位 0	C1EN	0x0	rw	通道 1 使能 (Channel 1 enable) 0: 禁止输入或输出; 1: 使能输入或输出。
-----	------	-----	----	--

表 15-11 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

控制位			输出状态 (1)			
OEN 位	FCSODIS 位	FCSOEN 位	CxEN 位	CxCEN 位	CxOUT 输出状态	CxCOUT 输出状态
1	X	0	0	0	输出禁止 (与定时器断开) CxOUT=0, CxEN=0	输出禁止 (与定时器断开) CxOUT=0, CxCEN=0
		0	0	1	输出禁止 (与定时器断开) CxOUT=0, CxEN=0	CxORAW + 极性, CxOUT=CxORAW xor CxCP, CxCEN=1
		0	1	0	CxORAW+极性, CxOUT=CxORAW xor CxP, CxCEN=1	输出禁止 (与定时器断开) CxOUT=0, CxCEN=0
		0	1	1	CxORAW+极性+死区, CxEN=1	CxORAW 反相+极性+死区, CxCEN=1
		1	0	0	输出禁止 (与定时器断开) CxOUT=CxP, CxEN=0	输出禁止 (与定时器断开) CxOUT=CxP, CxCEN=0
		1	0	1	关闭状态 (输出使能且为无效电平) CxOUT=CxP, CxCEN=1	CxORAW + 极性, CxOUT=CxORAW xor CxCP, CxCEN=1
		1	1	0	CxORAW + 极性, CxOUT=CxORAW xor CxP, CxCEN=1	关闭状态 (输出使能且为无效电平) CxOUT=CxP, CxCEN=1
		1	1	1	CxORAW+极性+死区, CxEN=1	CxORAW 反相+极性+死区, CxCEN=1
0	X	0	0	0	输出禁止 (与定时器断开) 异步地: CxOUT=CxP, CxEN=0, CxCOUT=CxCP, CxCEN=0; 若时钟存在: 经过一个死区时间后 CxOUT=CxIOS, CxCOUT=CxCIOS, 假设 CxIOS 与 CxCIOS 并不都对应 CxOUT 和 CxCOUT 的有效电平。	
		0	0	1		
		0	1	0		
		0	1	1		
		1	0	0		
		1	0	1	关闭状态 (输出使能且为无效电平) 异步地: CxOUT =CxP, CxEN=1, CxCOUT=CxCP, CxCEN=1; 若时钟存在: 经过一个死区 时间后 CxOUT =C1IOS, CxCOUT=CxCIOS, 假设 CxIOS 与 CxCIOS 并不都对应 CxOUT 和 CxCOUT 的有效电平。	
		1	1	0		
		1	1	1		
		1	1	1		

注意: 如果一个通道的 2 个输出都没有使用 ($CxEN = CxCEN = 0$) , 那么 $CxIOS$, $CxCIOS$, CxP 和 $CxCP$ 都必须清零。

注意: 引脚连接到互补的 $CxOUT$ 和 $CxCOUT$ 通道的外部 I/O 引脚的状态, 取决于 $CxOUT$ 和 $CxCOUT$ 通道状态和 $GPIO$ 以及 $IOMUX$ 寄存器。

15.3.5.8 TMR10、TMR11、TMR13和TMR14计数值寄存器 (TMRx_CVAL)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	CVAL	0x0000	rw	计数值 (Counter value)

15.3.5.9 TMR10、TMR11、TMR13和TMR14预分频器寄存器 (TMRx_DIV)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	DIV	0x0000	rw	分频系数 (Divider value) 计数器时钟频率 $f_{CK_CNT} = f_{TMR_CLK} / (DIV[15: 0]+1)$ 溢出事件发生时该寄存器值被传送到实际的预分频寄存器中。

15.3.5.10 TMR10、TMR11、TMR13和TMR14周期寄存器 (TMRx_PR)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	PR	0xFFFF	rw	周期值 (Period value) 定时器计数的周期值。当周期值为 0 时，定时器不工作。

15.3.5.11 TMR10、TMR11、TMR13和TMR14重复周期寄存器 (TMRx_RPR) (x=10/11/13/14)

域	简称	复位值	类型	功能
位 31: 8	保留	0x00 0000	resd	保持默认值。
位 7: 0	RPR	0x00	rw	重复周期的次数 (Repetition of period value) 这些位用于减慢溢出事件发生的速率，当重复周期的次数减为 0 时才会发生溢出事件。

15.3.5.12 TMR10、TMR11、TMR13和TMR14通道1数据寄存器 (TMRx_C1DT)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	C1DT	0x0000	rw	通道 1 数据寄存器值 (Channel 1 data register) 若通道 1 配置为输入： C1DT 是前一次通道 1 输入事件 (C1IN) 所保存的 CVAL。 若通道 1 配置为输出： C1DT 是将要和 CVAL 进行比较的值，写入的值是否会立即生效取决于输出缓存使能位 (C1OBEN)，并根据设置在 C1OUT 上产生相应的输出。

15.3.5.13 TMR10、TMR11、TMR13和TMR14刹车寄存器 (TMRx_BRK) (x=10/11/13/14)

域	简称	复位值	类型	功能
位 31: 20	保留	0x000	resd	保持默认值。
位 19: 16	BRKF	0x0	rw	刹车输入滤波 (brake input filter) 这些位用于配置刹车输入的滤波器。滤波的个数为 N，则表示发生了 N 次采样事件后输入边沿才能通过滤波器： 0000: 无滤波器，以 f_{DTS} 采样 0001: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=2 0010: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=4 0011: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=8 0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=6 0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=8 0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=6 0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=8 1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=6 1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=8 1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=5 1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=6 1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=8 1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=5 1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=6 1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=8
位 15	OEN	0x0	rw	输出使能 (Output enable) 对配置为输出的通道，该位用于使能 CxOUT 和 CxCOUT 的输出。 0: 关闭； 1: 开启。
位 14	AOEN	0x0	rw	输出自动使能 (Automatic output enable) 用于溢出事件时将 OEN 自动置'1' 0: 关闭； 1: 开启
位 13	BRKV	0x0	rw	刹车输入信号的有效性 (Brake input validity) 用于选择刹车输入信号的输入有效电平： 0: 低电平； 1: 高电平。
位 12	BRKEN	0x0	rw	刹车功能使能 (Brake enable) 用于开启刹车功能。 0: 关闭； 1: 开启。
位 11	FCSOEN	0x0	rw	总输出开时的冻结状态 (Frozen channel status when holistic output enable) 该位用于配置具有互补输出的通道，在定时器不工作且 OEN=1 时的通道状态。 0: 关闭 CxOUT/CxCOUT 输出； 1: 开启 CxOUT/CxCOUT 输出，输出为无效电平。
位 10	FCSODIS	0x0	rw	总输出关时的冻结状态 (Frozen channel status when holistic output disable) 该位用于配置具有互补输出的通道，在定时器不工作且 OEN=0 时的通道状态。 0 : 关闭 CxOUT/CxCOUT 输出； 1 : 开启 CxOUT/CxCOUT 输出，输出为空闲电平。
位 9: 8	WPC	0x0	rw	写保护配置 (Write protected configuration) 该位用于配置写保护。 00: 写保护关闭； 01: 3 级写保护，以下位受写保护： TMRx_BRK: BRKF、AOEN、BRKV、BRKEN 和 DTC TMRx_CTRL2: C1IOS 和 C1CIOS 10: 2 级写保护，除 3 级写保护的内容外，以下位也受写保护：

位 7: 0	DTC	0x00	rw	TMRx_CCTRL: C1P 和 C1CP TMRx_BRK: FCSODIS 和 FCSOEN 11: 1 级写保护, 除 2 级写保护的内容外, 以下位也受写保护: TMRx_CM1: C1OCTRL 和 C1OBEN 注: WPC>0 时将无法再次被修改, 直到系统复位。 死区配置 (Dead-time configuration) 这些位用于配置死区时间。取 DTC[7: 0]的高 3 位为功能选择位: 0xx: DT = DTC [7: 0] * TDTS; 10x: DT = (64+ DTC [5: 0]) * TDTS * 2; 110: DT = (32+ DTC [4: 0]) * TDTS * 8; 111: DT = (32+ DTC [4: 0]) * TDTS * 16;
--------	-----	------	----	---

注意: 根据锁定设置, BRKF、AOEN、BRKV、BRKEN、FCSODIS、FCSOEN 和 DTC[7: 0]位均可被写保护, 有必要在第一次写入 TMRx_BRK 寄存器时对它们进行配置。

15.3.5.14 TMR10、TMR11、TMR13和TMR14 DMA控制寄存器 (TMRx_DMACTRL) (x=10/11/13/14)

域	简称	复位值	类型	功能
位 31: 13	保留	0x0 0000	resd	保持默认值。
位 12: 8	DTB	0x00	rw	DMA 传输字节 (DMA transfer bytes) 这些位定义了传输的字节个数: 00000: 1 个字节 00001: 2 个字节 00010: 3 个字节 00011: 4 个字节 10000: 17 个字节 10001: 18 个字节
位 7: 5	保留	0x0	resd	保持默认值。
位 4: 0	ADDR	0x00	rw	DMA 传输地址偏移 (DMA transfer address offset) ADDR 定义了从 TMR1_CTRL1 所在地址开始的偏移量: 00000: TMRx_CTRL1, 00001: TMRx_CTRL2, 00010: TMRx_STCTRL,

15.3.5.15 TMR10、TMR11、TMR13和TMR14 DMA数据寄存器 (TMRx_DMADT) (x=10/11/13/14)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	DMADT	0x0000	rw	DMA 传输的数据寄存器 (DMA data register) 通过对 DMADT 寄存器的读写能够实现对任意 TMR 寄存器的操作, 其操作的寄存器地址范围是: TMR1 外设地址 + ADDR*4 至 TMRx 外设地址 + ADDR*4 + DTB*4。

15.3.5.16 TMR14通道输入重映射寄存器 (TMR14_RMP)

域	简称	复位值	类型	功能
位 31: 2	保留	0x0000 0000	resd	保持默认值。
位 1: 0	TMR14_CH1_IRMP	0x0	rw	TMR14 通道 1 输入重映射 (TMR14 channel 1 input remap) 00: TMR14 通道 1 输入与 GPIO 相连接 01: ERTC_CLK 10: HEXT_DIV 11: CLK_OUT

15.4 高级控制定时器（TMR1和TMR8）

15.4.1 TMR1和TMR8简介

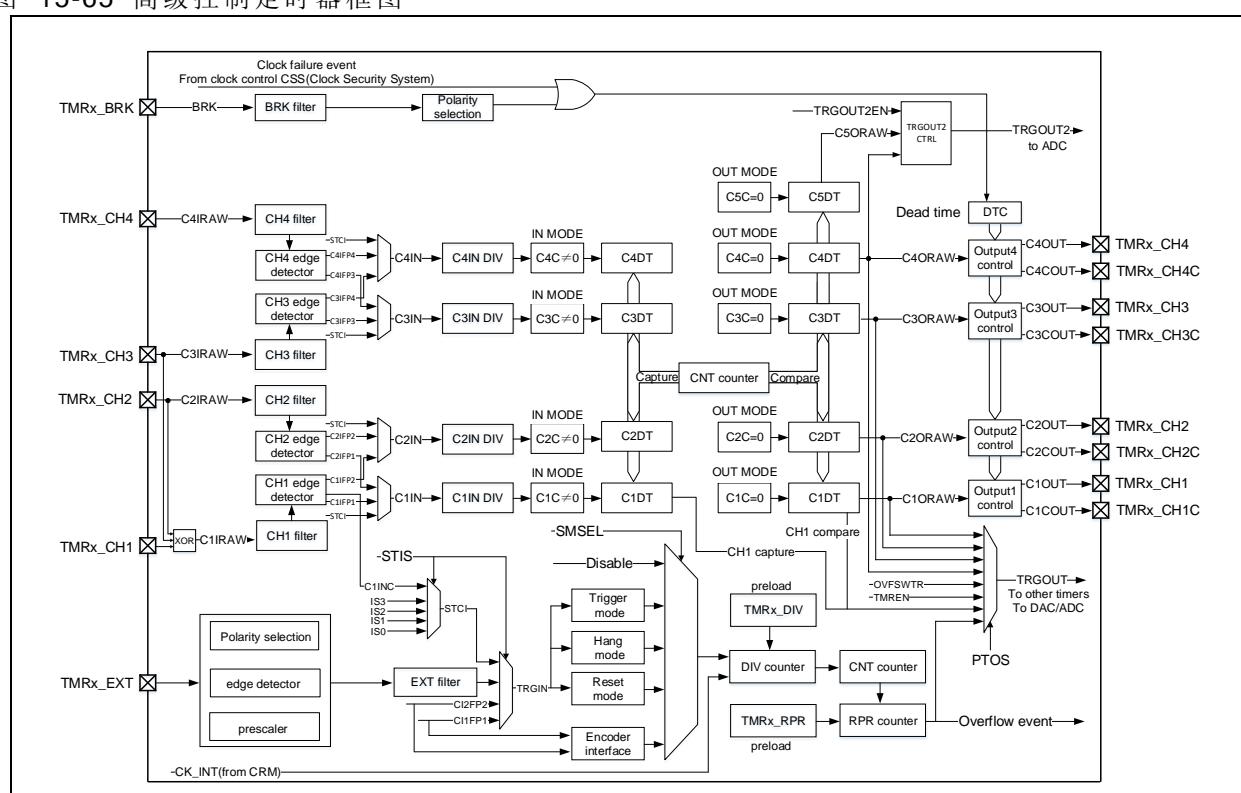
高级定时器 TMR1、TMR8 包含一个支持向上、向下计数的 16 位计数器、4 个捕获/比较寄存器、5 组独立的通道。可实现嵌入死区、输入捕获、可编程 PWM 输出。

15.4.2 TMR1和TMR8主要功能

TMR1 和 TMR8 定时器的功能包括：

- 可选内部、外部、内部触发输入用作计数时钟
- 16 位支持向上、向下、双向、重复计数、编码器模式的计数器
- 5 组独立通道，通道 5 仅支持输出比较，并通过 TRGOUT2 输出；通道 1 至 4 支持输入捕获、输出比较、PWM 生成、单周期模式、死区插入。
- 4 组支持互补输出的独立通道
- 支持 TMR 刹车功能
- 定时器之间可互联同步
- 支持溢出事件、触发事件、刹车输入、通道事件触发中断/DMA
- 支持 TMR Burst DMA 传输

图 15-65 高级控制定时器框图

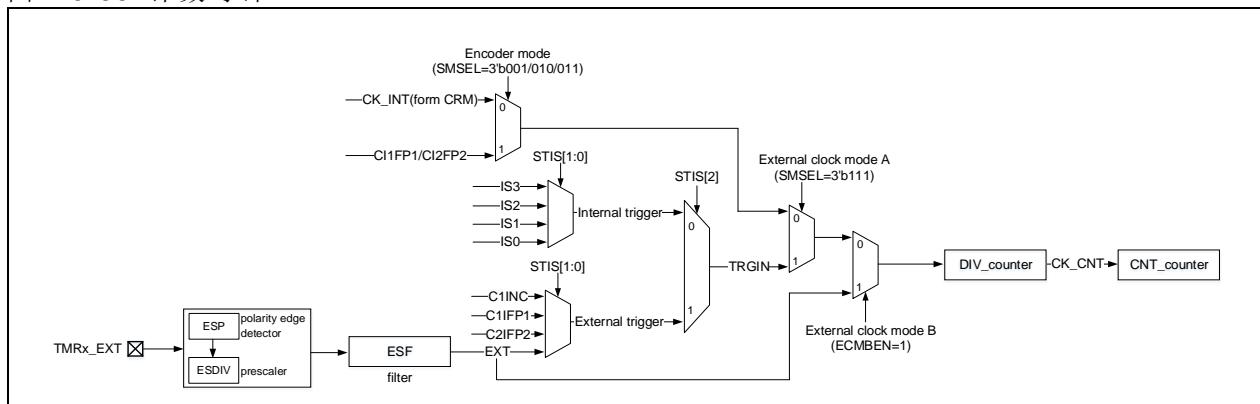


15.4.3 TMR1和TMR8功能描述

15.4.3.1 计数时钟

TMR1、TMR8 计数时钟可从内部时钟 (CK_INT)、外部时钟 (外部时钟模式 A、B)、内部触发输入 (ISx) 这些时钟源提供。

图 15-66 计数时钟

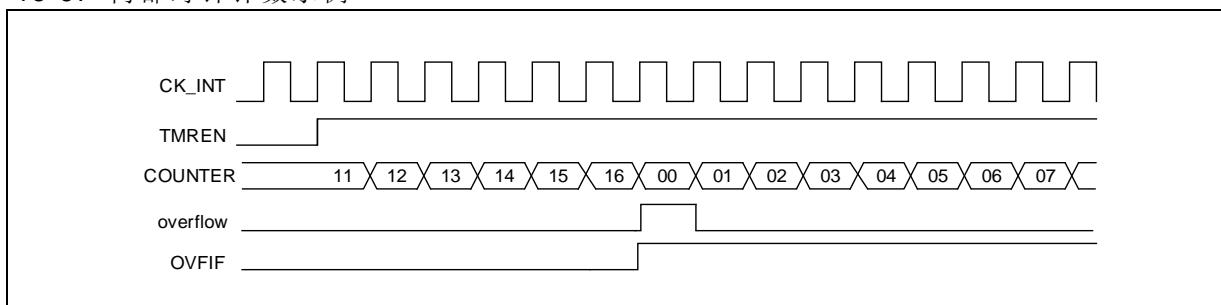


内部时钟 (CK_INT)

默认下使用 CK_INT 经由预分频器驱动计数器计数, 当 TMR 对应的 APB 时钟预分频系数是 1 时, CK_INT 频率等于 APB 时钟频率, 否则 CK_INT 频率等于 APB 时钟频率的 2 倍。相关配置流程如下:

- 配置 TMRx_CTRL1 寄存器 TWCMSEL[1:0], 选择计数模式, 若选择单向对齐计数模式, 还需配置 TMRx_CTRL1 寄存器 OWCDIR 选择计数方向。
- 配置 TMRx_DIV 寄存器, 设置计数器计数频率。
- 配置 TMRx_PR 寄存器, 设置计数器计数周期。
- 配置 TMRx_CTRL1 寄存器 TMREN, 使能计数器。

图 15-67 内部时钟计数示例



外部时钟 (TRGIN/EXT)

计数时钟可由两种外部时钟源提供, 分别为 TRGIN 和 EXT 信号。

当配置 TMRx_STCTRL 寄存器 SMSEL=3'b111 时, 外部时钟模式 A 被选中, 配置 TMRx_STCTRL 寄存器 STIS[2: 0] 来选择外部时钟源 TRGIN 信号驱动计数器计数。外部时钟源 TRGIN 可选则 C1INC (STIS=3'b100, 通道 1 上升沿和下降沿信号)、C1IFP1 (STIS=3'b101, 通道 1 滤波且极性选择后信号)、C2IFP2 (STIS=3'b110, 通道 2 滤波且极性选择后信号) 和 EXT (STIS=3'b111, 外部输入经极性选择、分频和滤波后信号)。

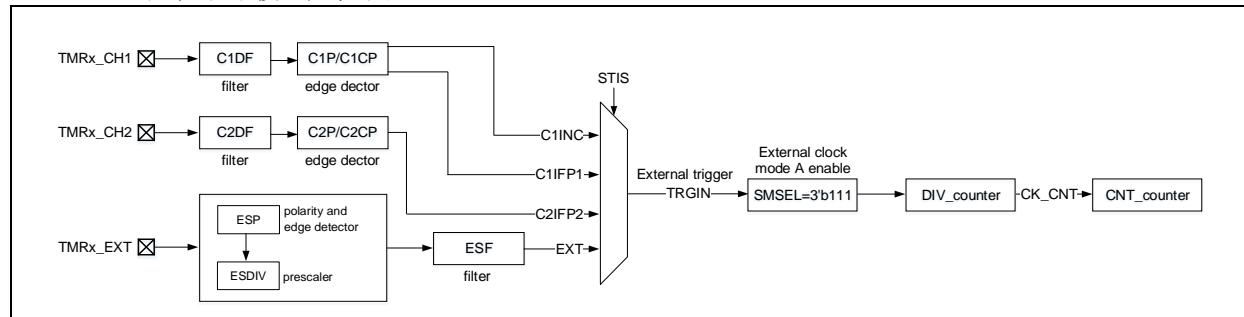
当配置 TMRx_STCTRL 寄存器 ECMBEN=1 时, 外部时钟模式 B 被选中, 计数器由外部输入经极性选择、分频和滤波后 EXT 信号驱动计数。外部时钟模式 B 等效于外部时钟模式 A 选择 EXT 信号作为外部时钟源 TRGIN。

若要使用外部时钟模式 A, 可按如下步骤配置:

- 配置外部时钟源 TRGIN 参数。
 - 若选择 TRGIN 来源为 TMRx_CH1, 需配置通道 1 输入滤波 (TMRx_CM1 寄存器 C1DF[3:0]) 和通道 1 输入极性 (TMRx_CCTRL 寄存器 C1P/C1CP)。
 - 若选择 TRGIN 来源为 TMRx_CH2, 需配置通道 2 输入滤波 (TMRx_CM1 寄存器 C2DF[3:0]) 和通道 1 输入极性 (TMRx_CCTRL 寄存器 C2P/C2CP)。

- 若选择 TRGIN 来源为 TMRx_EXT，需配置外部信号极性（TMRx_STCTRL 寄存器 ESP）、外部信号分频（TMRx_STCTRL 寄存器 ESDIV[1:0]）和外部信号滤波（TMRx_STCTRL 寄存器 ESF[3:0]）。
 - 配置 TMRx_STCTRL 寄存器 STIS[2:0]，设置 TRGIN 信号来源。
 - 配置 TMRx_STCTRL 寄存器 SMSEL=3'b111，使能外部时钟模式 A。
 - 配置 TMRx_DIV 寄存器 DIV[15:0]，设置计数器计数频率。
 - 配置 TMRx_PR 寄存器 PR[15:0]，设置计数器计数周期。
 - 配置 TMRx_CTRL1 寄存器 TMREN，使能计数器。
- 若要使用外部时钟模式 B，可按如下步骤配置：
- 配置 TMRx_STCTRL 寄存器 ESP，设置外部信号极性。
 - 配置 TMRx_STCTRL 寄存器 ESDIV[1:0]，设置外部信号分频。
 - 配置 TMRx_STCTRL 寄存器 ESF[3:0]，设置外部信号滤波。
 - 配置 TMRx_STCTRL 寄存器 ECMBEN，使能外部时钟模式 B。
 - 配置 TMRx_DIV 寄存器 DIV[15:0]，设置计数器计数频率。
 - 配置 TMRx_PR 寄存器 PR[15:0]，设置计数器计数周期。
 - 配置 TMRx_CTRL1 寄存器 TMREN，使能计数器。

图 15-68 外部时钟模式A框图



注：由于同步逻辑，输入端信号与计数器实际时钟之间存在一定延时。

图 15-69 外部时钟模式A计数示例

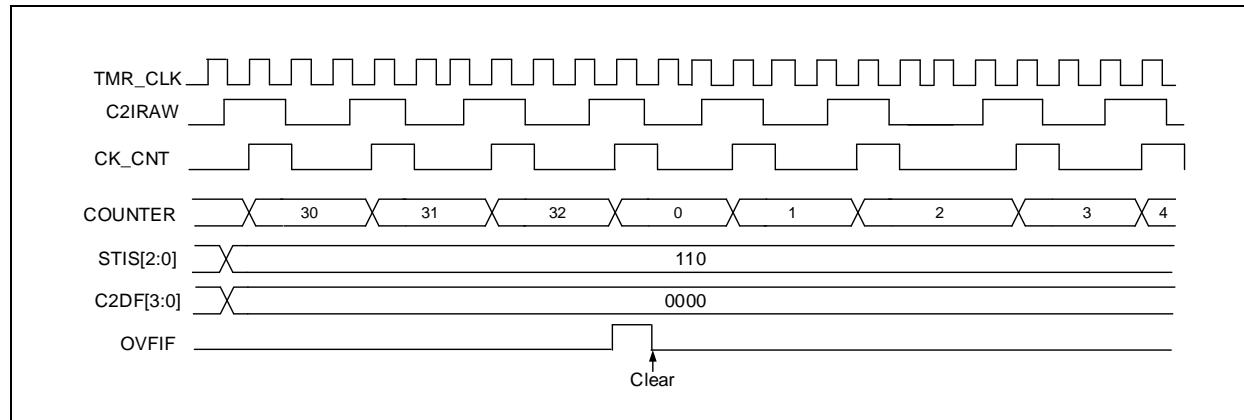
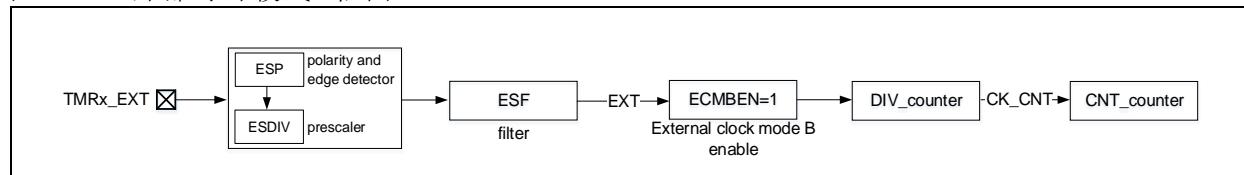
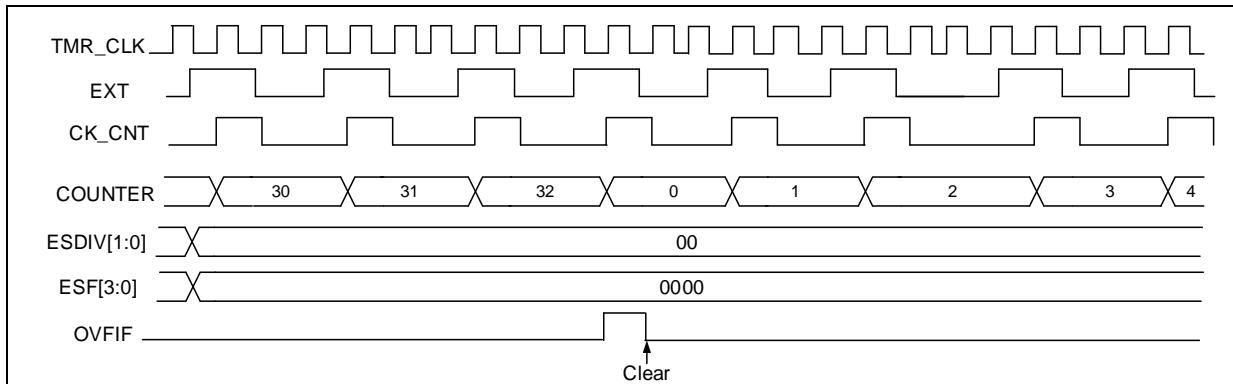


图 15-70 外部时钟模式B框图



注：由于同步逻辑。输入端 EXT 信号与计数器实际时钟之间存在一定延时。

图 15-71 外部时钟模式B计数示例



内部触发输入 (ISx)

定时器之间支持互联同步，因此一个定时器的 **TMR_CLK** 可由另一个定时器输出信号 **TRGOOUT** 提供。配置 **TMRx_STCTRL** 寄存器 **STIS[2: 0]** 选择内部触发信号驱动计数器计数。

高级定时器内含一个 16 位预分频器，用于产生驱动计数器计数的时钟 **CK_CNT**，通过配置 **TMRx_DIV** 寄存器值，可灵活调整 **CK_CNT** 与 **TMR_CLK** 之间的分频关系。预分频值可在任何时刻修改，但只在下一个溢出事件发生时，新值才会生效。

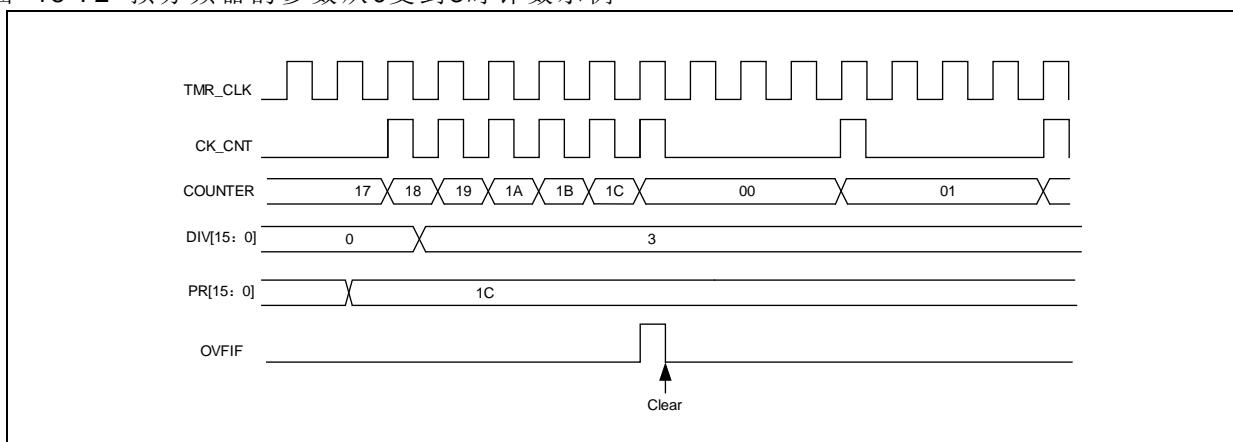
内部触发输入配置流程如下：

- 配置 **TMRx_PR** 寄存器，设置计数器计数周期。
- 配置 **TMRx_DIV** 寄存器，设置计数器计数频率。
- 配置 **TMRx_CTRL1** 寄存器 **TWCMSEL[1:0]** 位，设置计数器计数模式。
- 配置 **TMRx_STCTRL** 寄存器 **STIS[2:0]** 位范围为 3'b000~3'b011，选择内部触发。
- 配置 **TMRx_STCTRL** 寄存器 **SMSEL[2:0]=3'b111**，选择外部时钟模式 A。
- 配置 **TMRx_CTRL1** 寄存器 **TMREN** 位，使能 **TMRx** 计数。

表 15-12 TMRx 内部触发连接

次定时器	IS0 (STIS=000)	IS1 (STIS=001)	IS2 (STIS=010)	IS3 (STIS=011)
TMR1	TMR5	TMR2	TMR3	TMR4
TMR8	TMR1	TMR2	TMR4	TMR5

图 15-72 预分频器的参数从0变到3时计数示例



15.4.3.2 计数模式

高级定时器支持多种计数模式，用来满足不同的应用场景。其内部拥有一个支持 16 位向上计、向下、中央双向对齐计数模式计数器。

TMRx_PR 寄存器用于设置计数器计数周期。默认 **TMRx_PR** 寄存器值会立即传入它的影子寄存器；当开启周期缓冲功能后（**TMRx_CTRL1** 寄存器 **PRBEN** 置 1），**TMRx_PR** 寄存器值在溢出事件发生时传入它的影子寄存器。

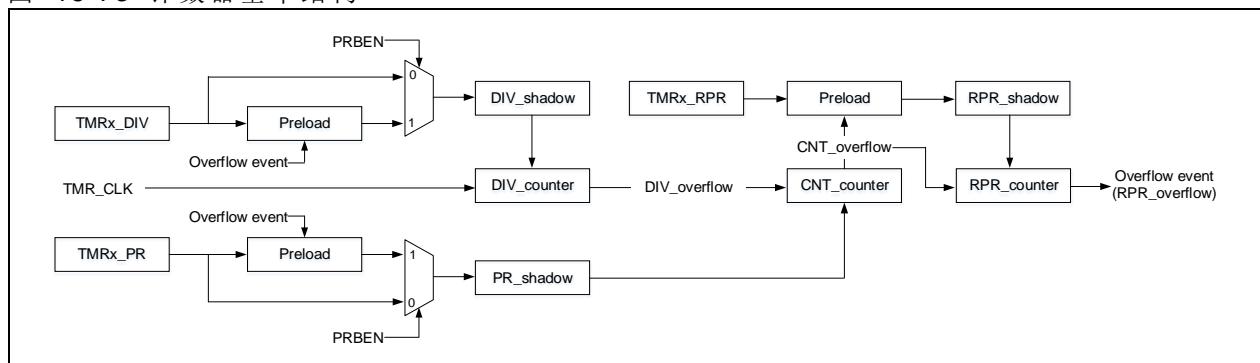
TMRx_DIV 寄存器用于设置计数器计数频率，每（**DIV[15:0]+1**）个计数时钟周期，计数器计数一次。**TMRx_DIV** 寄存器值在溢出事件时更新至它的影子寄存器。

读取 **TMRx_CNT** 寄存器会返回当前计数器计数值，写入 **TMRx_CNT** 寄存器会更新计数器当前计数值为写入值。

默认允许产生溢出事件，设置 **TMRx_CTRL1** 寄存器 **OVFEN=1** 将禁止溢出事件产生。**TMRx_CTRL1** 寄存器 **OVFS** 用于选择溢出事件来源，默认计数器上溢或下溢、置位 **OVFSWTR**、复位模式次定时器控制器产生的复位信号产生溢出事件。置位 **OVFS** 后，只有计数器上溢或下溢产生溢出事件。

TMREN 位置 1 将使能定时器计数，由于同步逻辑，实际驱动计数器的使能信号 **TMR_EN** 相对于 **TMREN** 延迟一个时钟周期。

图 15-73 计数器基本结构



向上计数模式

配置 **TMRx_CTRL1** 寄存器 **TWCMSEL[1:0]=2'b00**, **OWCDIR=1'b0** 开启向上计数模式，计数值达到 **TMRx_PR** 值时，重新从 0 向上计数，计数器上溢并产生溢出事件，同时 **OVFIF** 位置 1。若禁止产生溢出事件，计数器溢出后不再重载预分频值和周期值，否则预分频值和周期值在溢出事件后更新。

图 15-74 PRBEN=0 时的溢出事件

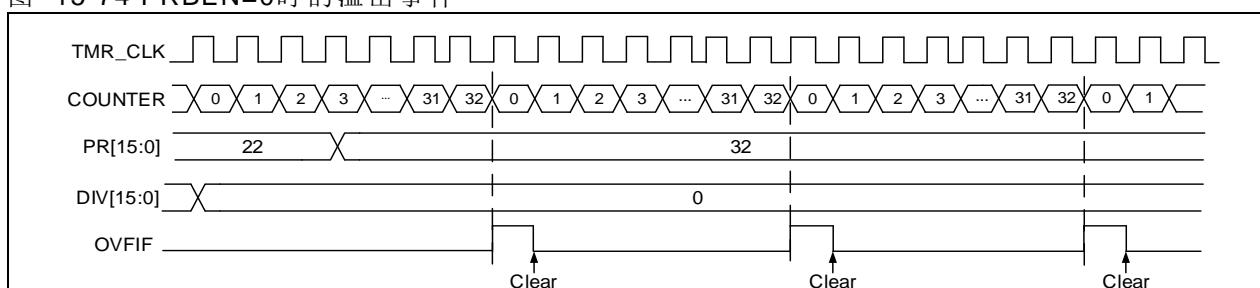
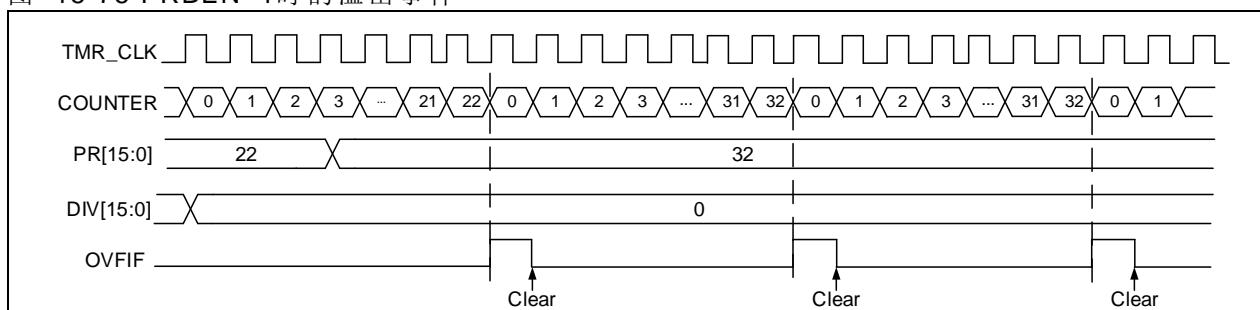


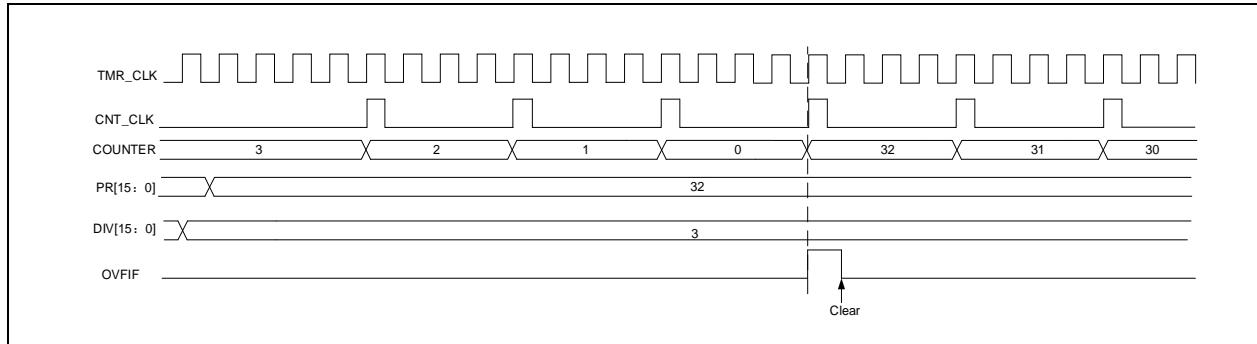
图 15-75 PRBEN=1 时的溢出事件



向下计数模式

配置 TMRx_CTRL1 寄存器 TWCSEL[1:0]=2'b00, OWCDIR=1'b1 开启向下计数模式，计数值达到 0 值并重新从 TMRx_PR 向上下数时，计数器下溢并产生溢出事件。

图 15-76 计数器时序图，内部时钟分频因子为 3



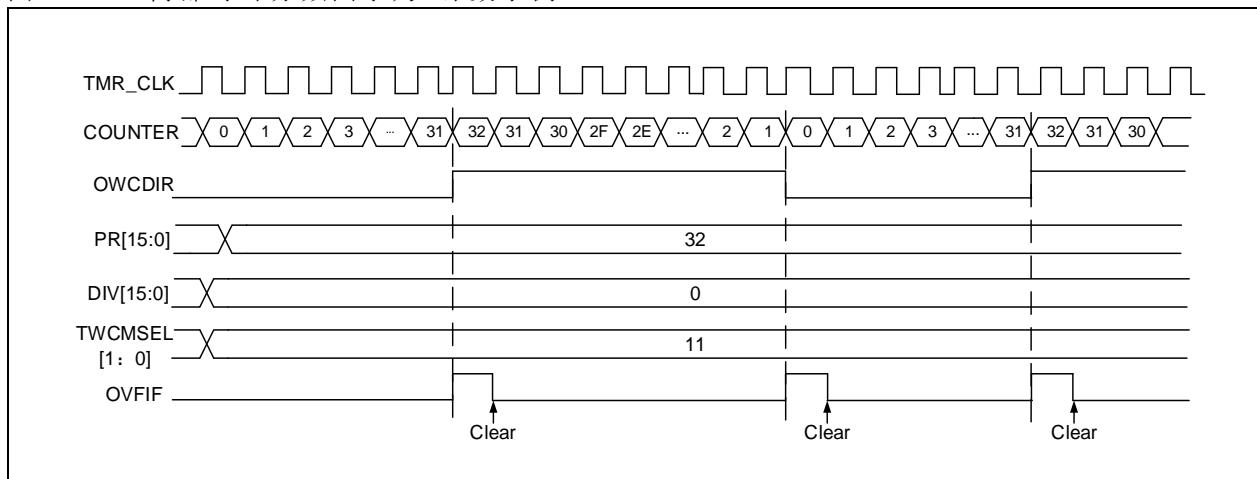
中央双向对齐计数模式

配置 TMRx_CTRL1 寄存器 TWCSEL[1:0]≠2'b00 开启中央双向对齐计数模式，中央双向对齐计数模式下计数器交替向上、向下计数。计数值从 TMRx_PR 值向下计数到 1 值，产生下溢事件，然后从 0 开始向上计数；向上计数到 TMRx_PR 值-1，产生上溢事件，之后从 TMRx_PR 值向下计数。计数器计数方向由计数器方向控制位（OWCDIR）实时查看。

TMRx_CTRL1 寄存器 TWCSEL[1:0]位还用于选择中央双向对齐计数模式下 CxIF 标志置起方式，中央双向对齐计数模式 1（TWCSEL[1:0]=2'b01）仅允许 CxIF 标志位在计数器向下计数时置起；双向对齐计数模式 2（TWCSEL[1:0]=2'b10）仅允许 CxIF 标志位在计数器向上计数时置起；双向对齐计数模式 3（TWCSEL[1:0]=2'b11）允许 CxIF 标志位在计数器向上和向下计数时置起。

注意： 中央双向对齐计数模式下，OWCDIR 位为只读位。

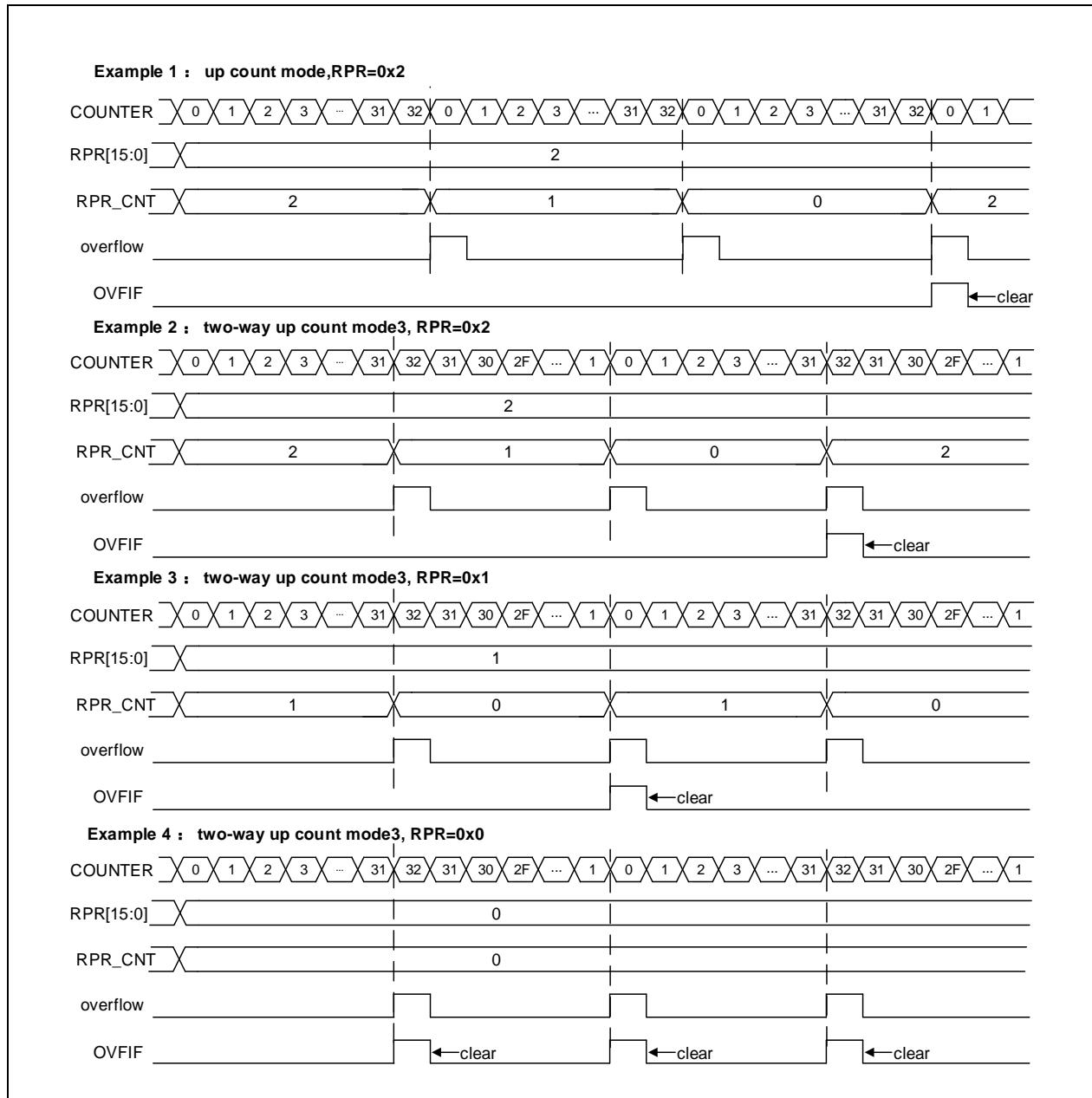
图 15-77 内部时钟分频因子为 0 计数示例



重复计数模式：

TMRx_RPR 寄存器用于配置重复计数器计数周期，TMRx_RPR 寄存器为非 0 值时，重复计数模式启动。重复计数模式下，每 (RPR[15:0]+1) 次计数器溢出将产生一次溢出事件。每次计数器溢出，重复计数器递减，仅当重复计数器计数值等于 0 值时，计数器溢出会产生溢出事件。通过配置不同重复计数器值，可调整溢出事件产生的频率。

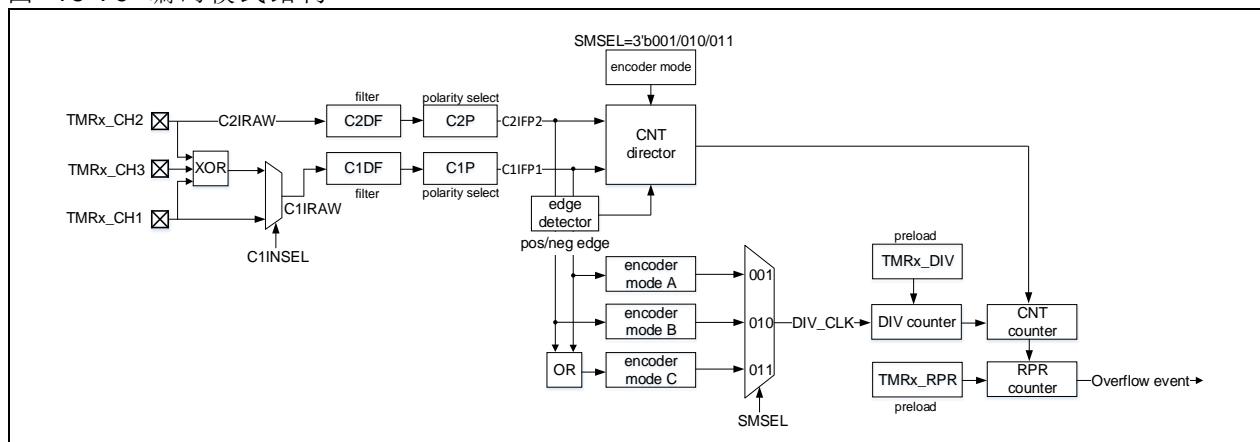
图 15-78 向上计数模式和中央双向对齐计数模式时 OVFIF



编码器模式

编码器模式下需提供两组输入信号 TMRx_CH1 和 TMRx_CH2，根据一组输入信号电平值，计数器在另一组输入信号边沿向上或向下计数。计数方向由 OWCDIR 值指示。

图 15-79 编码模式结构



编码器模式 A: SMSEL=3'b001, 计数器在 C1IFP1 边沿计数(上升沿和下降沿), 计数方向由 C1IFP1 边沿方向和 C2IFP2 电平高低共同决定。

编码器模式 B: SMSEL=3'b010, 计数器在 C2IFP2 边沿计数(上升沿和下降沿), 计数方向由 C2IFP2 边沿方向和 C1IFP1 电平高低共同决定。

编码器模式 C: SMSEL=3'b011, 计数器在 C1IFP1 和 C2IFP2 边沿计数(上升沿和下降沿), 计数方向由 C1IFP1 边沿方向和 C2IFP2 电平高低、C2IFP2 边沿方向和 C1IFP1 电平高低共同决定共同决定。

若要使用编码器模式可按下面步骤配置:

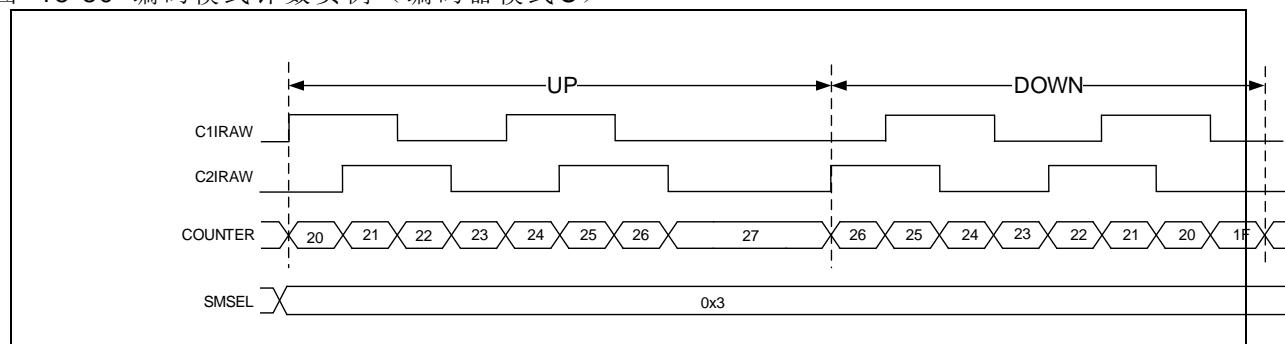
- 配置 TMRx_CM1 寄存器 C1DF[3:0], 设置通道 1 输入信号滤波; 配置 TMRx_CCTRL 寄存器 C1P, 设置通道 1 输入信号有效电平。
- 配置 TMRx_CM1 寄存器 C2DF[3:0], 设置通道 2 输入信号滤波; 配置 TMRx_CCTRL 寄存器 C2P, 设置通道 2 输入信号有效电平。
- 配置 TMRx_CM1 寄存器 C1C[1:0], 设置通道 1 为输入模式; 配置 TMRx_CM1 寄存器 C2C[1:0], 设置通道 2 为输入模式;
- 配置 TMRx_STCTRL 寄存器 SMSEL[2:0], 选择编码器模式 A (SMSEL=3'b001)、编码器模式 B (SMSEL=3'b010) 或编码器模式 C (SMSEL=3'b011)。
- 配置 TMRx_PR 寄存器 PR[15:0], 设置计数器计数周期。
- 配置 TMRx_DIV 寄存器 DIV[15:0], 设置计数器计数频率。
- 配置 TMRx_CH1 和 TMRx_CH2 对应 IO 为复用模式。
- 配置 TMRx_CTRL1 寄存器 TMREN, 使能计数器。

编码模式下计数器计数方向如下表所示:

表 15-13 计数方向与编码器信号的关系

计数边沿	计数边沿相对信号的电平 (C1IFP1 边沿对应 C2IFP2 电平, C2IFP2 边沿对应 C1IFP1 电平)	C1IFP1 边沿方向		C2IFP2 边沿方向	
		上升	下降	上升	下降
C1IFP1	高	向下计数	向上计数	不计数	不计数
	低	向上计数	向下计数	不计数	不计数
C2IFP2	高	不计数	不计数	向上计数	向下计数
	低	不计数	不计数	向下计数	向上计数
C1IFP1 和 C2IFP2	高	向下计数	向上计数	向上计数	向下计数
	低	向上计数	向下计数	向下计数	向上计数

图 15-80 编码模式计数实例 (编码器模式 C)



15.4.3.3 TMR输入部分

TMR1、TMR8 拥有 5 个独立通道，除通道 5 外，每个通道可配置为输入或输出，当配置位输入时，每个通道输入信号依次经过以下处理：

- TMRx_CHx 经过预处理输出 CxIRAW。配置 C1INSEL 位，选择 C1IRAW 来源是 TMRx_CH1 或是 TMRx_CH1、TMRx_CH2、TMRx_CH3 异或。C2IRAW、C3IRAW、C4IRAW 来源是 TMRx_CH2、TMRx_CH3、TMRx_CH4。
- CxIRAW 输入数字滤波器，输出滤波后信号 CxIF。数字滤波器通过 TMRx_CM1/CM2 寄存器 CxDL 位配置采样频率和次数。
- CxIF 输入边沿检测器，输出边沿选择后信号 CxIFP_x。边沿选择由 TMRx_CCTRL 寄存器 CxP 和 CxCP 位共同控制，可选择输入上升沿、下降沿或双边沿有效。
- CxIFP_x 输入捕获信号选择器，输出选择后信号 CxIN。捕获信号选择器由 TMRx_CM1/CM2 寄存器 CxC 控制，可选择 CxIN 来源为 CxIFP_x、CyIFP_y、STCI。其中 CyIFP_y ($x \neq y$) 是来自通道 y 的 CyIFP y 信号；STCI 来自次定时器控制器，由 STIS 位选择来源。
- CxIN 经由输入通道分频器，输出分频后信号 CxIPS。分频系数由 CxIDIV 位配置为不分频、2 分频、4 分频或 8 分频。

图 15-81 输入/输出通道 1 的主电路

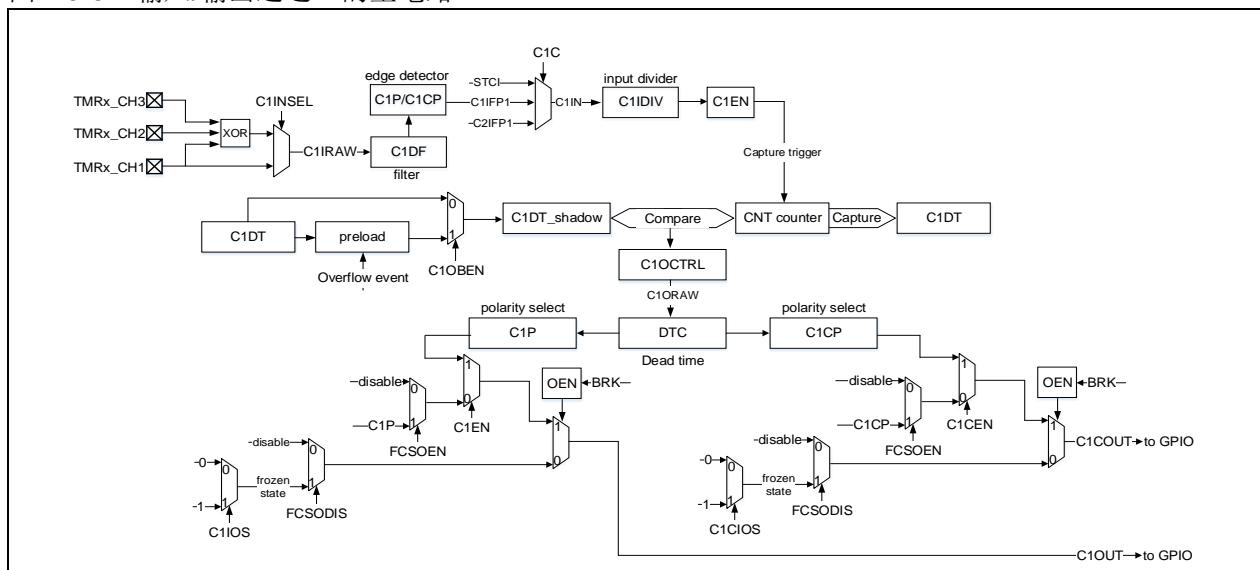
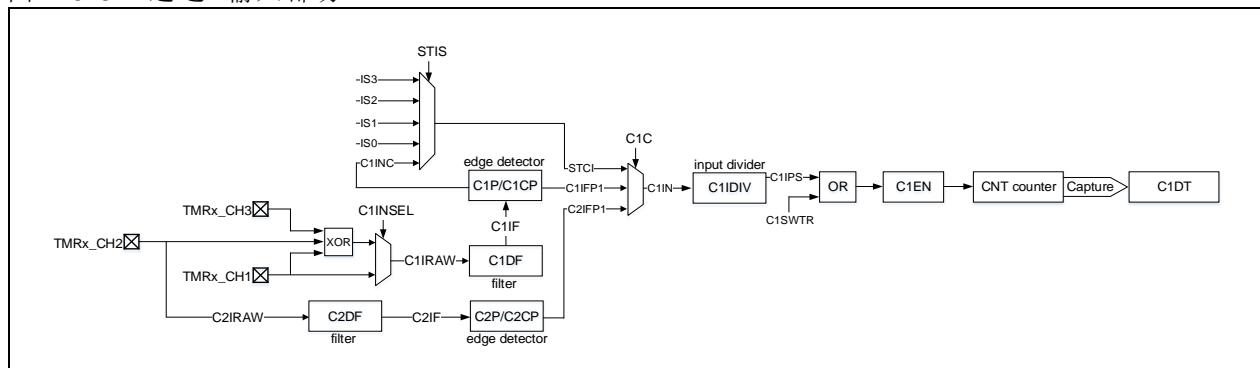


图 15-82 通道 1 输入部分



输入模式

此模式下，当选中的触发信号被检测到，通道寄存器 (TMRx_CxDT) 记录当前计数器计数值，并将捕获比较中断标志位 (TMRx_ISTS 寄存器 CxIF) 置 1，若已使能通道中断 (TMRx_IDEN 寄存器 CxIEN)、通道 DMA 请求 (TMRx_IDEN 寄存器 CxDEN) 则产生相应的中断和 DMA 请求。若在 CxIF 置 1 后检测到触发信号，将产生捕获溢出事件，TMRx_CxDT 会使用当前计数器计数值覆盖之前记录的计数器计数值，同时通道再捕获标志位 (TMRx_ISTS 寄存器 CxRF) 置 1。

若要捕获 C1IN 输入的上升沿，可按如下进行配置：

- 将通道模式寄存器 1 (TMRx_CM1) 中的 C1C 位配置为 2'b01，选择 C1IN 作为通道 1 输入。
- 配置 C1IN 信号滤波器带宽 (CxDF[3: 0])。
- 配置 C1IN 通道的有效沿，在 TMRx_CCTRL 寄存器中写入 C1P=2'b0 (上升沿)。
- 配置 C1IN 信号捕获分频 (C1DIV[1: 0])。
- 使能通道 1 输入捕获 (C1EN=1)。
- 根据需要设置 TMRx_IDEN 寄存器中的 C1IEN 为、TMRx_IDEN 寄存器中的 C1DEN 位，选择中断请求或 DMA 请求。

多输入异或

通道 1 的输入端可选择 TMRx_CH1、TMRx_CH2 和 TMRx_CH3 经异或逻辑后输入。将 TMRx_CTRL2 寄存器中的 C1INSEL 位置 1 可开启此功能。

多输入异或功能可用于连接霍尔传感器，例如，将异或输入的三个输入端分别连接到三个霍尔传感器，通过分析三路霍尔传感器信号可计算出转子的位置和速度。

PWM 输入

PWM 输入模式适用于通道 1 和 2，要使用此模式，需要将 C1IN 和 C2IN 映射到同一 TMRx_CHx，并且通道 1 或 2 的 CxIFPx 配置成触发次定时器控制器复位。

PWM 输入模式可用于测量输入信号的周期和占空比，如需测量通道 1 输入信号的周期和占空比，操作步骤如下：

- 配置 TMRx_CM1 寄存器 C1C=2'b01，选择 C1IN 为 C1IFP1。
- 配置 TMRx_CCTRL 寄存器 C1P=1'b0，选择 C1IFP1 上升沿有效。
- 配置 TMRx_CM1 寄存器 C2C=2'b10，选择 C2IN 为 C1IFP2。
- 配置 TMRx_CCTRL 寄存器 C2P=1'b1，选择 C1IFP2 下降沿有效。
- 配置 TMRx_STCTRL 寄存器 STIS=3'b101，选择次定时器触发信号为 C1IFP1。
- 配置 TMRx_STCTRL 寄存器 SMSEL=3'b110，选择次定时器模式为复位模式。
- 配置 TMRx_CCTRL 寄存器 C1EN=1'b1，C2EN=1'b1。使能通道 1 和通道 2 输入捕获。

上述配置下，通道 1 输入信号的上升沿会触发捕获并将捕获值存储到 C1DT 寄存器，同时通道 1 输入信号上升沿复位计数器。通道 1 输入信号下降沿触发捕获并将捕获值存储到 C2DT 寄存器。通道 1 输入信号的周期可通过 C1DT 计算，占空比可通过 C2DT 计算。

图 15-83 PWM 输入模式配置实例

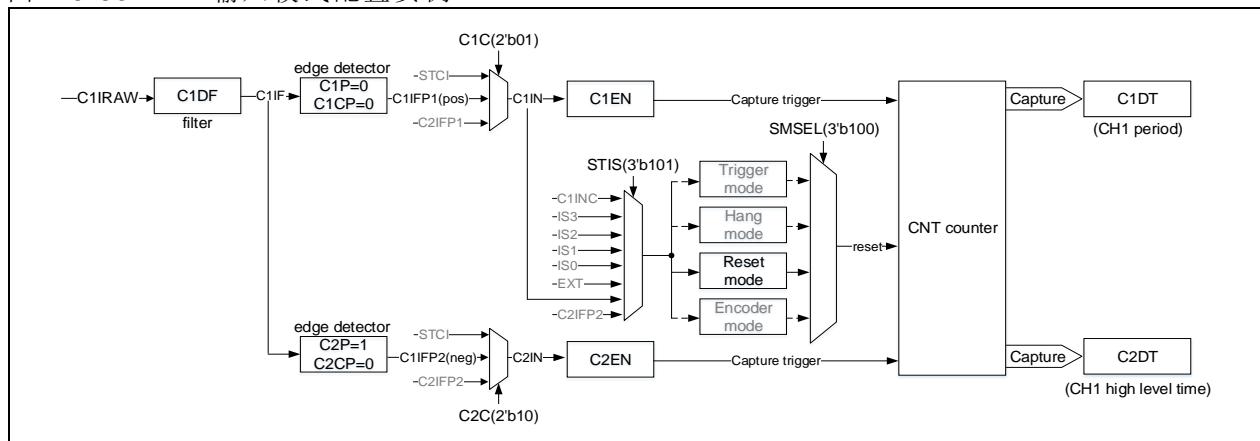
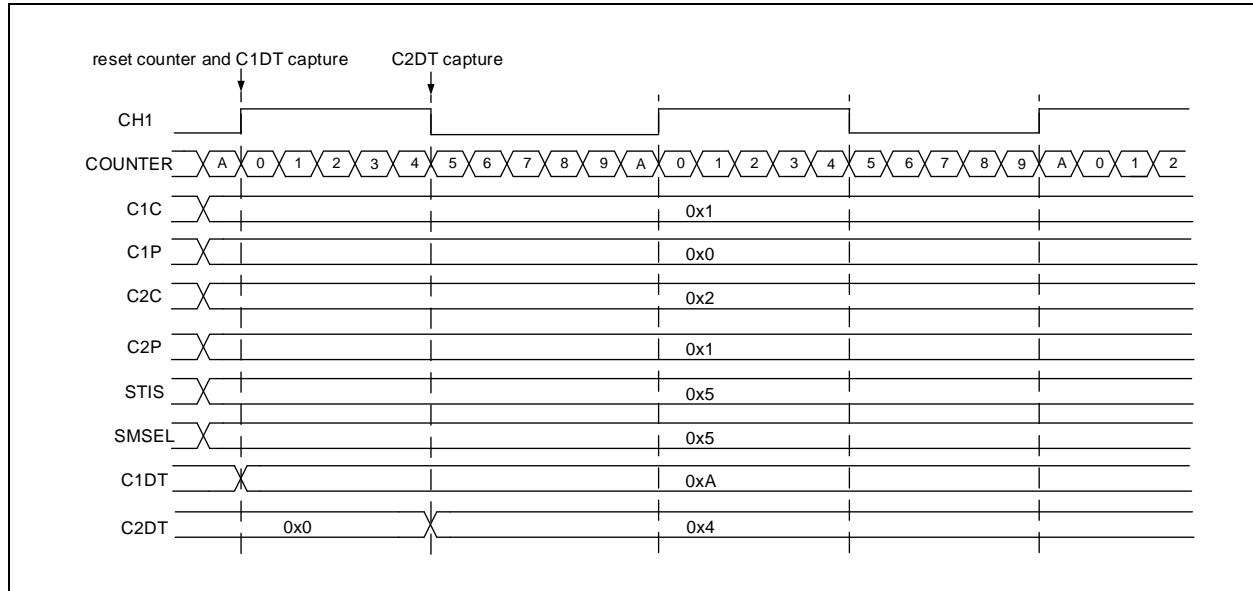


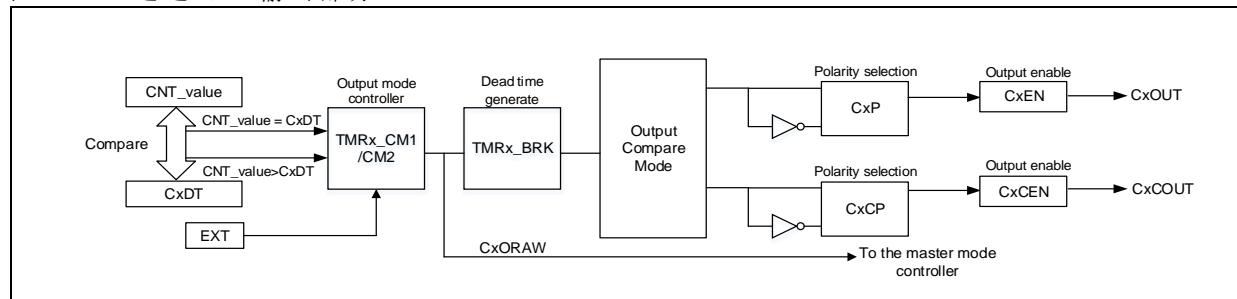
图 15-84 PWM 输入模式



15.4.3.4 TMR输出部分

TMR 的输出部分由比较器和输出控制构成，用于编程输出信号的周期、占空比、极性。高级定时器的输出部分在不同通道上有所不同。通道 5 的输出仅连接至 TRGOUT2，当配置 TMRx_CTRL2 寄存器 TRGOUT2EN=1 时，TRGOUT2 输出 C4ORAW 上升沿逻辑或 C5ORAW 下降沿，通道 1 至 4 如下图所示：

图 15-85 通道1至4输出部分



输出模式

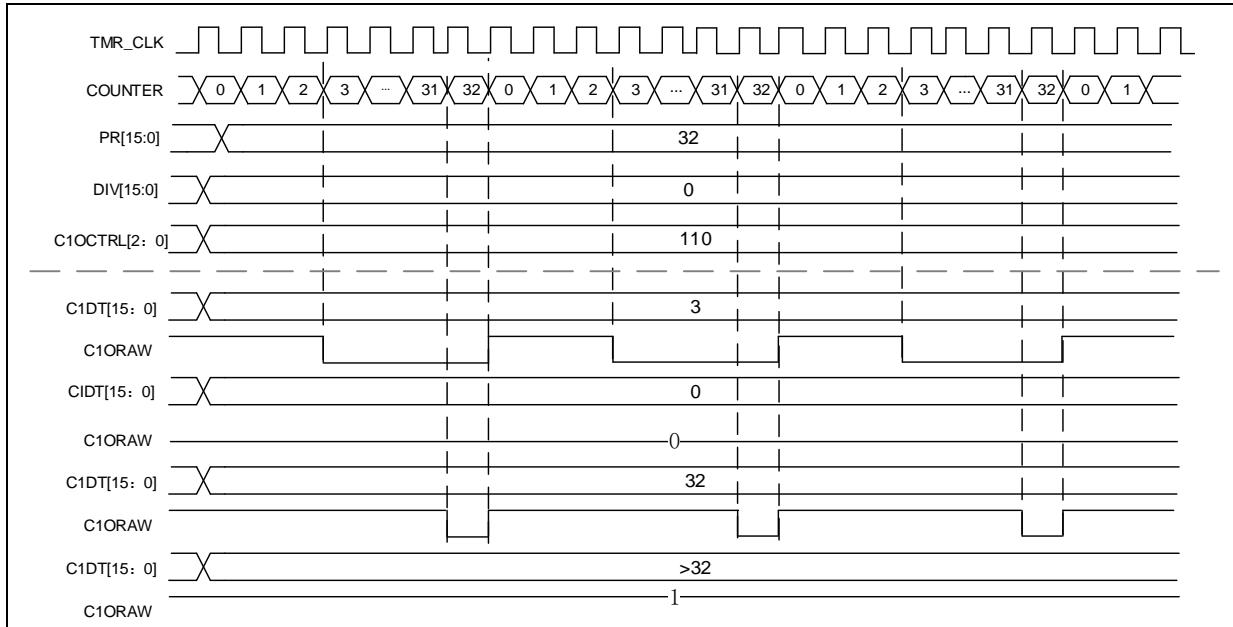
配置 $CxOCTRL[1:0] \neq 2'$ b00 将通道配置为输出可实现多种输出模式，此时，计数器计数值将与 $CxDT$ 寄存器值比较，并根据 $CxOCTRL[2:0]$ 位配置的输出模式，产生中间信号 $CxORAW$ ，再经过输出控制逻辑处理后输送到 IO。输出信号的周期由 $TMRx_PR$ 寄存器值配置，占空比则由 $CxDT$ 寄存器值配置。

输出比较模式有以下子类：

PWM 模式 A: $CxOCTRL=3'b110$ 时，开启 PWM 模式 A。向上计数时， $TMRx_C1DT > TMRx_CVAL$ 时 $C1ORAW$ 输出高电平，否则为低电平；向下计数时， $TMRx_C1DT < TMRx_CVAL$ 时 $C1ORAW$ 输出低电平，否则为高电平。图 15-86 展示了计数器向上计数与 PWM 模式 A 配合的例子， $PR=0x32$ ， $CxDT$ 配置为不同的值时输出信号的翻转情况。若要使用 PWM 模式 A，可按如下方式配置。

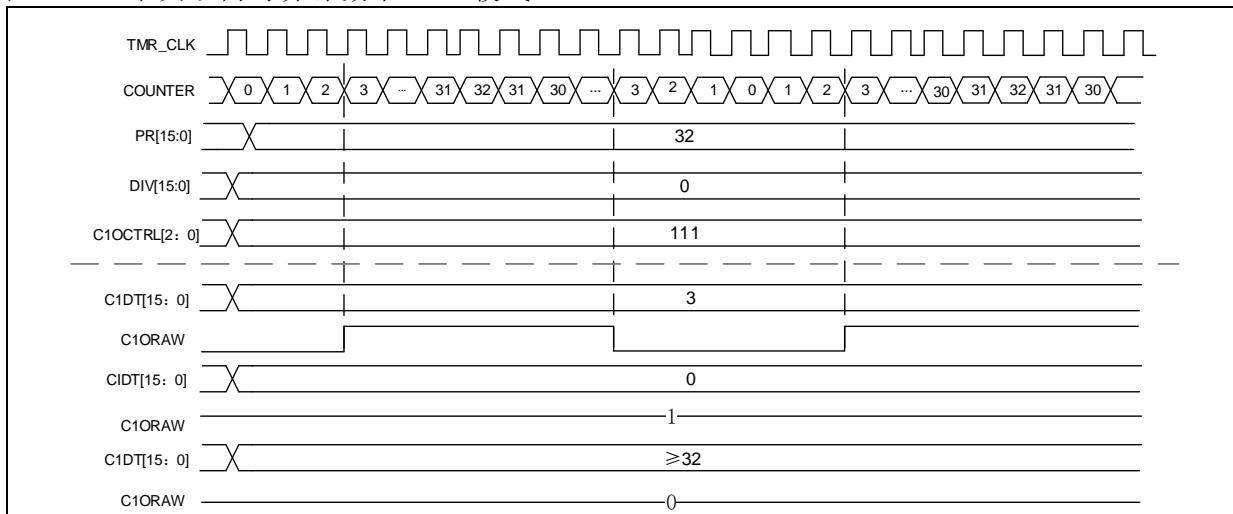
- 配置 $TMRx_PR$ 寄存器，设置 PWM 周期。
- 配置 $TMRx_CxDT$ 寄存器，设置 PWM 占空比。
- 配置 $TMRx_CM1/CM2$ 寄存器 $CxOCTRL$ 位为 $3'b110$ ，设置输出模式为 PWM 模式 A。
- 配置 $TMRx_DIV$ 寄存器，设置计数器计数频率。
- 配置 $TMRx_CTRL1$ 寄存器 $TWCMSL[1:0]$ 位，设置计数器计数模式。
- 配置 $TMRx_CCTRL$ 寄存器 CxP 位、 $CxCP$ 位，设置输出极性。
- 置 $TMRx_CCTRL$ 寄存器 $CxEN$ 位、 $CxCEN$ 位，使能通道输出。
- 配置 $TMRx_BRK$ 寄存器 OEN 位，使能 TMRx 输出。
- 配置 TMR 输出通道对应 GPIO 为对应的复用模式。
- 配置 $TMRx_CTRL1$ 寄存器 $TMREN$ 位，使能 TMRx 计数。

图 15-86 向上计数下 PWM 模式 A



PWM 模式 B: CxOCTRL=3'b111 时，开启 PWM 模式 B。向上计数时， $\text{TMR}_x\text{_C1DT} > \text{TMR}_x\text{_CVAL}$ 时 C1ORAW 输出低电平，否则为高电平；向下计数时， $\text{TMR}_x\text{_C1DT} < \text{TMR}_x\text{_CVAL}$ 时 C1ORAW 输出高电平，否则为低电平。图 15-87 展示了计数器中央双向对齐计数与 PWM 模式 B 配合的例子，PR=0X32，CxDT 配置为不同的值时输出时输出信号的翻转情况。

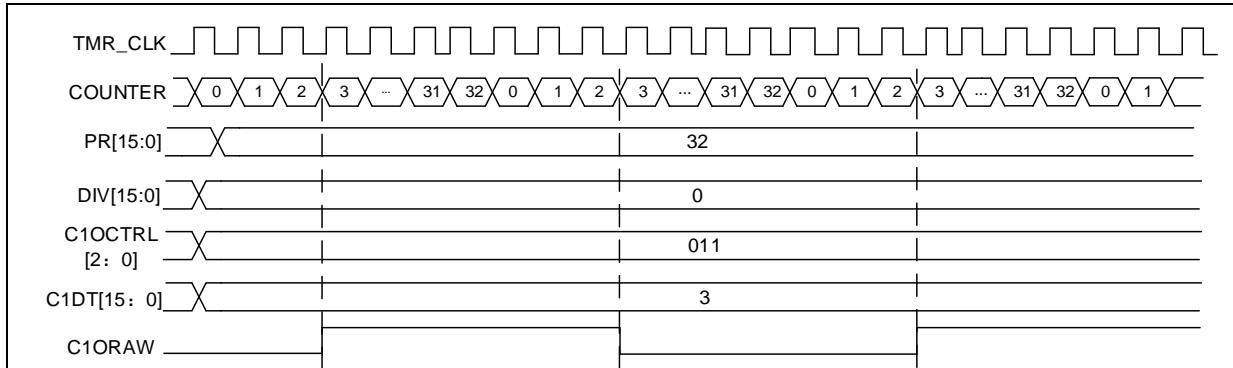
图 15-87 中央双向对齐计数下 PWM 模式 B



强制输出模式: CxOCTRL=3'b100/101 时，开启强制输出模式。此时，CxORAW 信号的电平被强制输出为配置的电平，而与计数值无关。虽然输出信号不依赖于比较结果，但通道标志位和 DMA 请求仍依赖于比较结果。

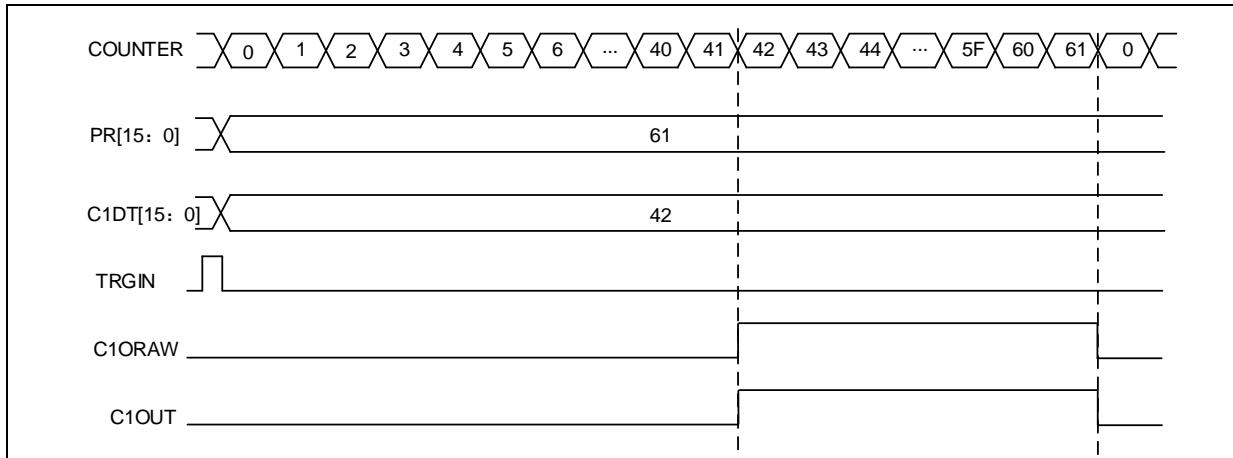
输出比较模式: CxOCTRL=3'b001/010/011 时，开启输出比较模式。此时，当计数值与 CxDT 值匹配时，CxORAW 强制输出高电平（CxOCTRL=3'b001）、低电平（CxOCTRL=3'b010）或进行电平翻转（CxOCTRL=3'b011）。图 15-88 展示了输出比较模式（翻转）的例子，C1DT=0x3，当计数值等于 0x3 时，输出电平 C1OUT 被翻转。

图 15-88 计数值与 C1DT 值匹配时翻转 C1ORAW



单周期模式: PWM 模式的特例，将 TMRx_CTRL1 寄存器 OCMEN 位置 1 可开启单周期模式，此模式下，仅在当前计数周期中进行比较匹配，完成当前计数后，TMREN 位清 0，因此仅输出一个脉冲。当配置为向上计数模式时，需要严格配置 $CVAL < CxDT \leq PR$ ；向下计数时，需严格配置 $CVAL > CxDT$ 。图 15-89 展示了计数器向上计数与单周期模式下 PWM 模式 B 配合的例子，计数器仅计数了一个周期，输出信号在这个周期中只输出了一个脉冲。

图 15-89 单周期模式



快速输出模式: 将 TMRx_CM1/CM2 寄存器 CxOEN 位置 1 可开启此功能，开启后 CxORAW 电平值不再在计数值与 CxDT 匹配时变化，而是在当前计数周期开始时，也就是说，比较结果被提前了，计数器值与 CxDT 寄存器的比较结果将会提前决定 CxORAW 的电平。

主定时器事件输出

当 TMR 作为主定时器时，可选择如下信号源作为 TRGOUT 信号输出到次定时器，选择信号为 TMRx_CTRL2 寄存器 PTOS 位。

- PTOS=3'b000, TRGOUT 输出软件溢出事件（TMRx_SWEVT 寄存器 OVFSWTR 位）或复位事件。
- PTOS=3'b001, TRGOUT 输出计数器使能信号。
- PTOS=3'b010, TRGOUT 输出计数器溢出事件。
- PTOS=3'b011, TRGOUT 输出捕获、比较事件。
- PTOS=3'b100, TRGOUT 输出 C1ORAW 信号。
- PTOS=3'b101, TRGOUT 输出 C2ORAW 信号。
- PTOS=3'b110, TRGOUT 输出 C3ORAW 信号。
- PTOS=3'b111, TRGOUT 输出 C4ORAW 信号。

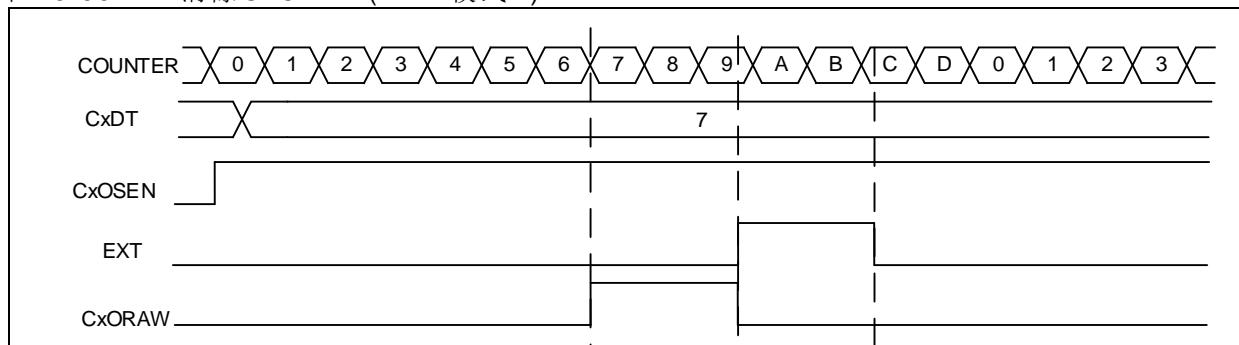
CxORAW 信号清除

将 TMRx_CM1/CM2 寄存器 CxOSEN 位置 1 后，指定通道的 CxORAW 信号由 EXT 高电平清 0，在下一次溢出事件发生前 CxORAW 信号无法被改变。

强制输出模式时，CxORAW 信号清除功能不可用，只有在输出比较模式或 PWM 模式，此功能有效。下图显示了使用 EXT 信号清除 CxORAW 的例子，当 EXT 为高电平期间，原本为高电平的 CxORAW 信号

被拉低，当 EXT 为低电平时，CxORAW 根据计数值和 CxDT 比较结果输出电平。

图 15-90 EXT 清除 CxORAW(PWM 模式 B)



死区插入

高级定时器通道 1 至 4 包含一组反向通道输出，通过 CxCEN 使能，通过 CxCP 配置极性。

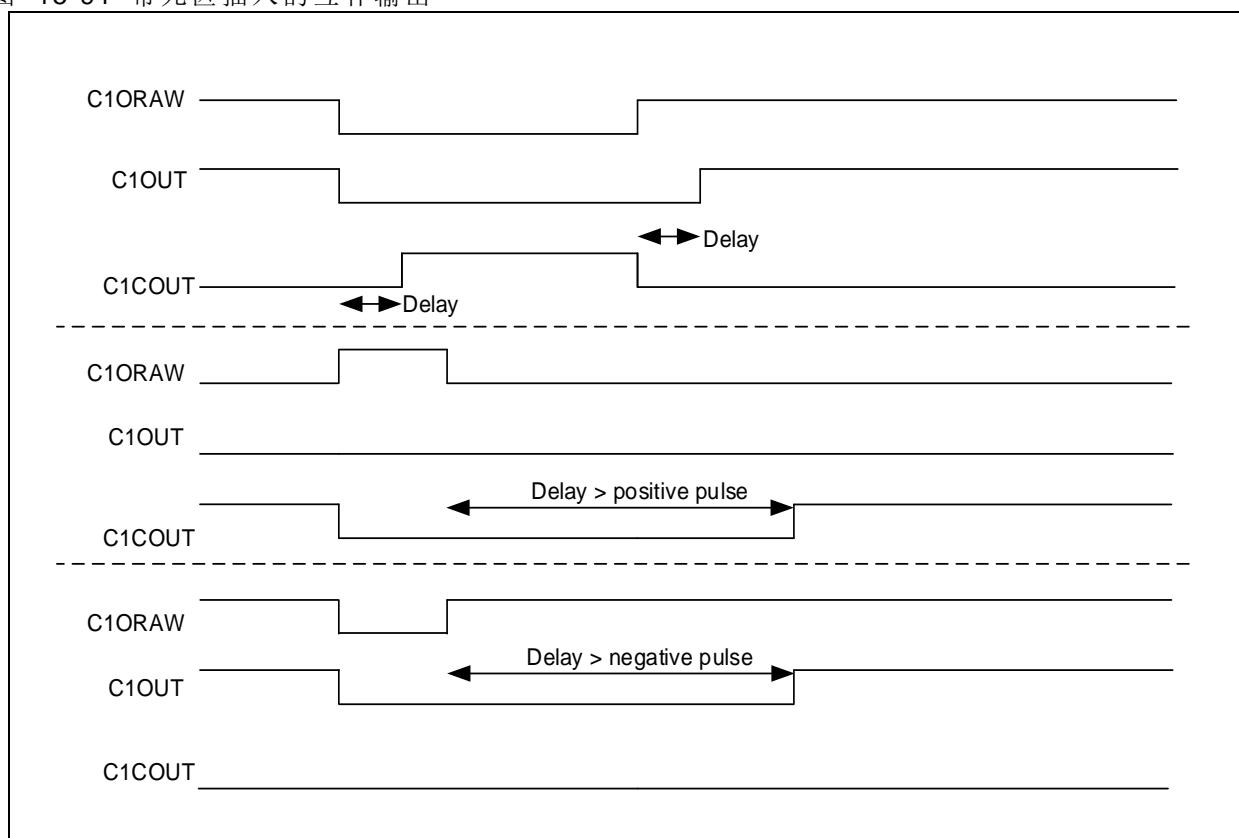
当转换为 IDLEF 状态，即 OEN 下降到 0，死区被激活。

将 CxEN 和 CxCEN 位置 1 后，通过配置 DTC[7: 0]死区发生器，可插入不同时长的死区。插入死区后，CxOUT 的上升沿延迟于参考信号的上升沿；CxCOUT 的上升沿延迟于参考信号的下降沿。

如果延迟大于当前有效的输出宽度，C1OUT 和 C1COUT 不会产生相应的脉冲，死区时间应小于有效的输出宽度。

下列图显示了 CxP=0、CxCP=0、OEN=1、CxEN=1 并且 CxCEN=1 时死区插入的例子

图 15-91 带死区插入的互补输出



15.4.3.5 TMR刹车功能

开启刹车功能后 (TMRx_BRK 寄存器 BRKEN 位置 1)，CxOUT 和 CxCOUT 由 OEN、FCSODIS、FCSOEN、CxIOS 和 CxCIOS 共同控制。但 CxOUT 和 CxCOUT 输出总是不能同时处于有效电平上的。详见表 15-15 带刹车功能的互补输出通道 CxOUT 和 CxCOUT 的控制位。

刹车信号来源可以是刹车输入引脚、时钟失效事件，刹车输入信号的极性由 BRKV 位控制。

当发生刹车事件时，有下述动作：

- OEN 位异步清零，通道输出状态由 FCSODIS 位选择。关闭 MCU 的振荡器不影响该功能。
- OEN 被清零后，通道输出电平由 CxIOS 位设定。如果 FCSODIS=0，则定时器输出使能被禁止，否则输出使能始终为高。
- 当使用互补输出时：
 - 输出最开始处于复位状态，也就是无效的状态（取决于极性）。这是异步操作，定时器有无时钟并不影响此功能。
 - 定时器的时钟如果有效，会开启死区生成功能，CxIOS 和 CxCIOS 位用来配置死区之后的电平。即使在这种情况下，CxOUT 和 CxCOUT 也不能被同时驱动到有效的电平。
 - 注意，由于 OEN 位同步逻辑，死区时间较通常情况会延长一段时间（大约 2 个 tmr_clk 的时钟周期）。
- 如果 FCSODIS=0，定时器释放使能输出，否则保持使能输出；或一旦 CxEN 与 CxCEN 之一变高时，使能输出变为高。
- 如果开启了刹车中断或 DMA 功能，刹车状态标志将置 1，并产生刹车中断或 DMA 请求。
- 如果将 AOEN 位置 1，在下一个溢出事件时 OEN 位被自动置 1。

注意：刹车输入电平有效时，OEN 不能被设置，状态标志 BRKIF 也不能被清除。

图 15-92 TMR 输出控制

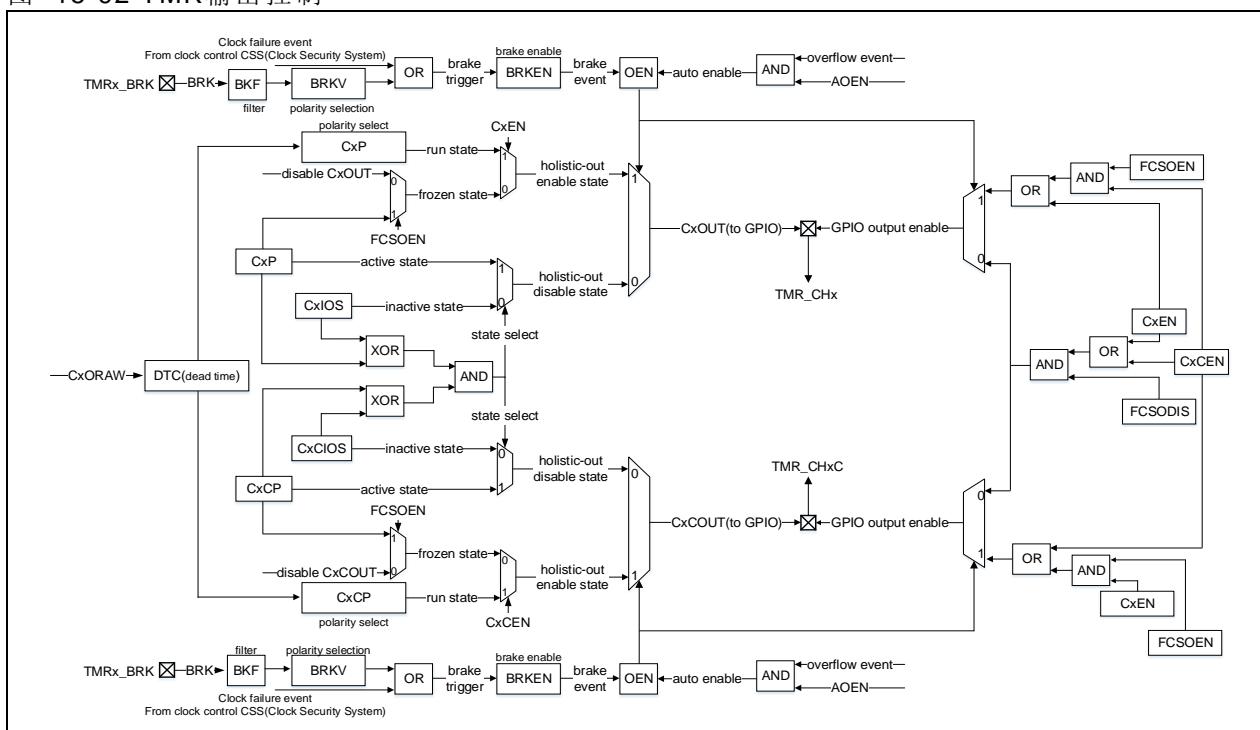
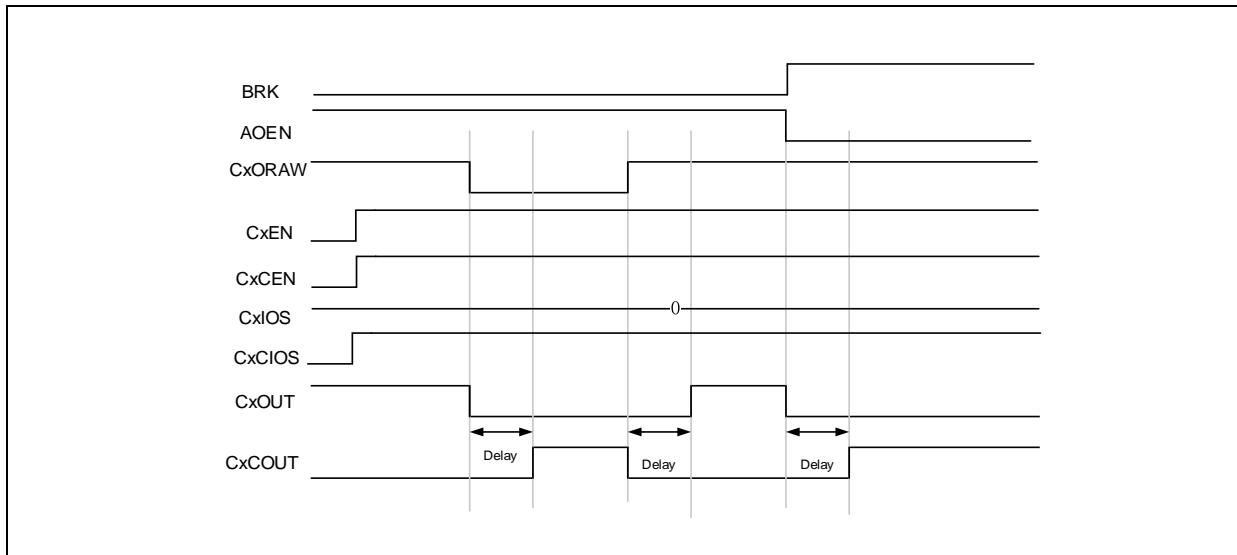


图 15-93 TMR刹车功能的例子



15.4.3.6 TMR同步

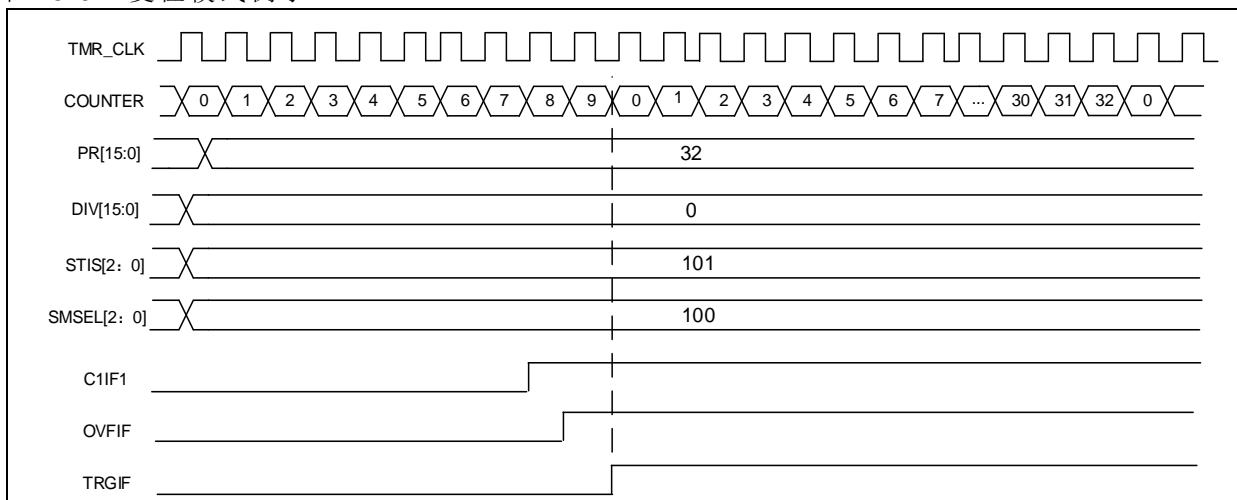
主次定时器之间可由内部连接信号进行同步。主定时器可由 **TMRx_CTRL2** 寄存器 PTOS[2: 0]位选择主定时器输出，即同步信息；次定时器由 **TMRx_STCTRL** 寄存器 SMSEL[2: 0]位选择从模式，即次定时器的工作模式。

定时器从模式有以下几种：

从模式：复位模式

选中的触发信号将复位计数器和预分频器，若 **TMRx_CTRL1** 寄存器 OVFS 位为 0，将产生一个溢出事件。

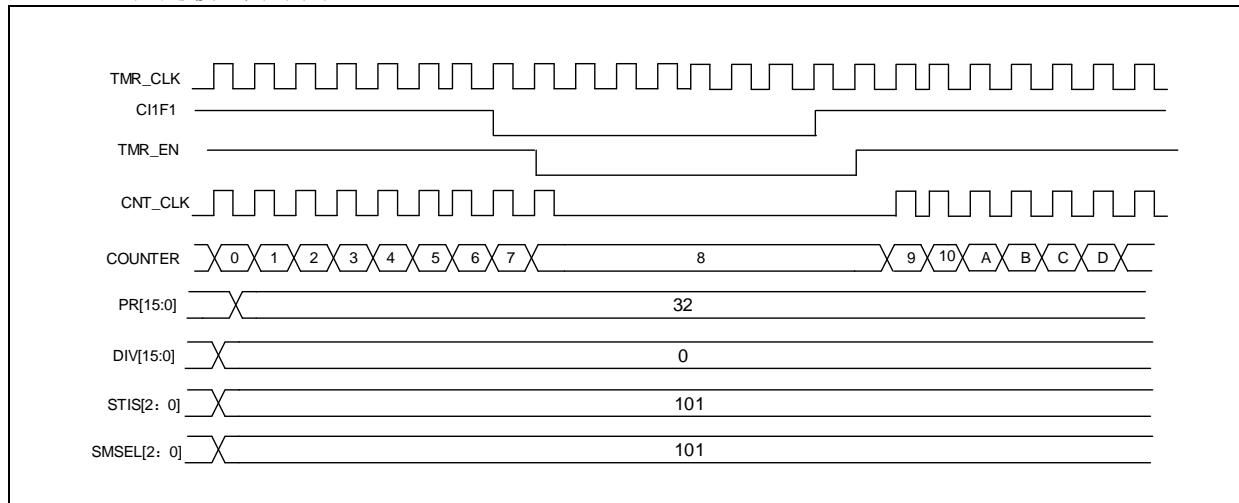
图 15-94 复位模式例子



从模式：挂起模式

挂起模式下，计数的计数和停止受选中触发输入信号控制，当触发输入为高电平时计数器开始计数；当为低电平时，计数器暂停计数。

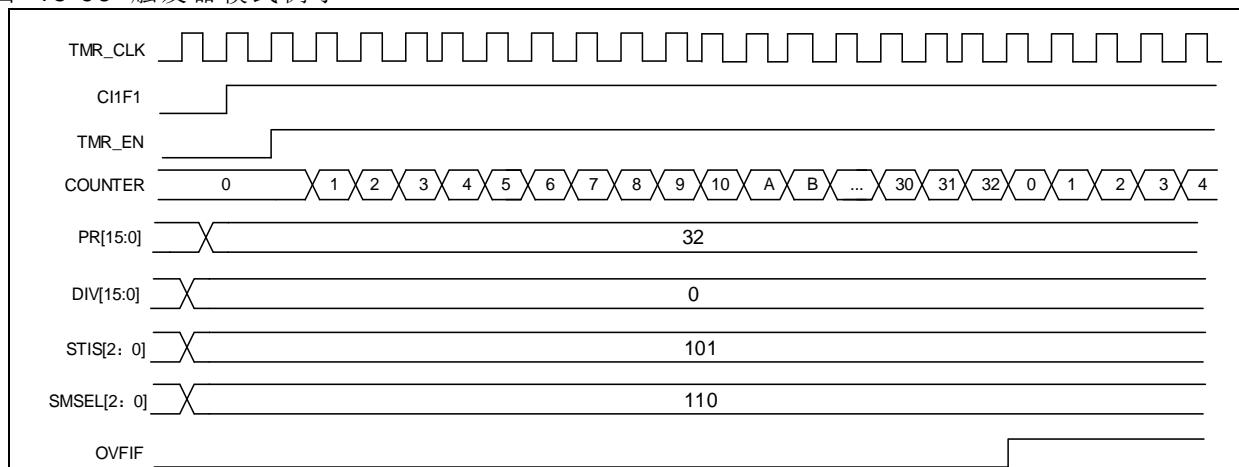
图 15-95 挂起模式下例子



从模式：触发模式

计数器将在选中的触发输入上升沿启动计数（将 TMR_EN 置 1）。

图 15-96 触发器模式例子



15.4.3.7 TMR DMA

TMR 中有溢出事件 DMA 请求、触发事件 DMA 请求、霍尔事件 DMA 请求以及通道事件 DMA 请求，通过使能 `TMRx_IDEN` 寄存器中相应位，开启 DMA 请求功能，产生相应事件时触发输出 DMA 请求给到 DMA 外设。

TMR DMA Burst 功能

TMR 还支持 TMR DMA Burst 功能，通过 `TMRx_IDEN` 寄存器使能某个事件的 DMA 请求，触发 DMA 改写多个 TMR 连续寄存器。

以溢出事件触发 TMR DMA Burst 功能为例，配置如下：

- 配置 `TMRx_IDEN` 寄存器中 `OVFDEN` 位，使能溢出事件触发 DMA 请求；
- 配置 `TMRx_DMACTRL` 寄存器中 `DTB` 位设置 Burst 传输次数；
- 配置 `TMRx_DMACTRL` 寄存器中 `ADDR` 位设置 Burst 传输的起始地址；
- 使能计数器。

以上配置的 TMR DMA Burst 传输流程如下：

当发生溢出事件时，TMR 将会产生溢出事件的 DMA 请求给到 DMA，DMA 根据请求将数据写入 `TMRx_DMADT` 寄存器，在 TMR 内部则会将 `DMADT` 位的数据写入 Burst 传输起始地址寄存器，并发送 ACK 信号给到 TMR，TMR 接收到 ACK 信号后，将清除当前 DMA 请求；TMR 检测到 Burst 传输并未全部完成，会重新置起溢出事件 DMA 请求给到 DMA，DMA 根据请求将数据再次写入 `TMRx_DMADT` 寄存器，在 TMR 内部则会将 `DMADT` 寄存器的数据写入 Burst 传输起始地址+`0x4` 地址寄存器，并发送 ACK 信号给到 TMR；以此往复，直到 Burst 传输最后一次操作时，检测到此时 Burst 传输已全部完成，溢出事件 DMA 请求信号将不会再被重新置起，直到下一次新的溢出事件到来。

注：使用 TMR DMA Burst 功能时，起始地址到结束地址这个区间，不应有空寄存器以及不应包含 `TMRx_DMACTRL` 和 `TMRx_DMADT` 寄存器。

15.4.3.8 调试模式

当微控制器进入调试模式（Cortex®-M4F 核心停止）时，将 DEBUG 模块中的 `TMRx_PAUSE` 置 1，可以使 TMRx 计数器暂停计数。

15.4.4 TMR1和TMR8寄存器描述

必须以字（32位）的方式操作这些外设寄存器。

表 15-14 TMR1和TMR8寄存器和复位值

寄存器简称	基址偏移量	复位值
TMRx_CTRL1	0x00	0x0000 0000
TMRx_CTRL2	0x04	0x0000 0000
TMRx_STCTRL	0x08	0x0000 0000
TMRx_IDEN	0x0C	0x0000 0000
TMRxISTS	0x10	0x0000 0000
TMRx_SWEVT	0x14	0x0000 0000
TMRx_CM1	0x18	0x0000 0000
TMRx_CM2	0x1C	0x0000 0000
TMRx_CCTRL	0x20	0x0000 0000
TMRx_CVAL	0x24	0x0000 0000
TMRx_DIV	0x28	0x0000 0000
TMRx_PR	0x2C	0x0000 FFFF
TMRx_RPR	0x30	0x0000 0000
TMRx_C1DT	0x34	0x0000 0000
TMRx_C2DT	0x38	0x0000 0000
TMRx_C3DT	0x3C	0x0000 0000
TMRx_C4DT	0x40	0x0000 0000
TMRx_BRK	0x44	0x0000 0000
TMRx_DMACTRL	0x48	0x0000 0000
TMRx_DMADT	0x4C	0x0000 0000
TMRx_CM3	0x70	0x0000 0000
TMRx_C5DT	0x74	0x0000 0000

15.4.4.1 TMR1、TMR8控制寄存器1 (TMRx_CTRL1)

域	简称	复位值	类型	功能
位 31: 10	保留	0x00 0000	resd	保持默认值。
位 9: 8	CLKDIV	0x0	rw	时钟除频 (Clock divider) 此位用于设置数字滤波器采样频率 f_{DTS} 和定时器时钟频率 f_{CK_INT} 之间的分频比, 也用于调整死区时间的时基 T_{DTS} 和定时器时钟周期 T_{CK_INT} 的分频比。 00: 无除频, $f_{DTS}=f_{CK_INT}$; 01: 2 除频, $f_{DTS}=f_{CK_INT}/2$; 10: 4 除频, $f_{DTS}=f_{CK_INT}/4$; 11: 保留。
位 7	PRBEN	0x0	rw	周期缓冲使能 (Period buffer enable) 0: 缓冲关闭; 1: 缓冲开启。
位 6: 5	TWCMSEL	0x0	rw	中央双向对齐计数模式选择 (Two-way count mode selection) 00: 单向对齐计数模式, 方向由 OWCDIR 配置; 01: 中央双向对齐计数模式 1, 上下交替计数, CxIF 位只在计数器向下计数时被置起; 10: 中央双向对齐计数模式 2, 上下交替计数, CxIF 位只在计数器向上计数时被置起; 11: 中央双向对齐计数模式 3, 上下交替计数, CxIF 位在计数器向上和向下计数时皆被置起。
位 4	OWCDIR	0x0	rw	单向对齐计数方向 (One-way count direction) 0: 向上; 1: 向下。
位 3	OCMEN	0x0	rw	单周期使能 (One cycle mode enable) 该功能用于选择溢出事件后, 计数器是否停止。 0: 关闭; 1: 开启。
位 2	OVFS	0x0	rw	溢出事件源选择 (Overflow event source) 配置溢出事件或 DMA 请求来源。 0: 来源于计数器溢出、设置 OVFSWTR 位或次定时器控制器产生的溢出事件; 1: 只能来源于计数器溢出。
位 1	OVFEN	0x0	rw	溢出事件使能 (Overflow event enable) 0: 开启; 1: 关闭。
位 0	TMREN	0x0	rw	使能定时器 (TMR enable) 0: 关闭; 1: 开启。

15.4.4.2 TMR1、TMR8控制寄存器2 (TMRx_CTRL2)

域	简称	复位值	类型	功能
位 31	TRGOUT2EN	0x0	rw	TRGOUT2 输出使能 (TRGOUT2 enable) 0: TRGOUT2 始终保持低电平; 1: TRGOUT2 输出 C4ORAW 上升沿或 C5ORAW 下降沿。
位 30: 16	保留	0x0000	resd	保持默认值。
位 15	C4CIOS	0x0	rw	通道 4 互补空闲输出状态 (Channel 4 complementary idle output state)
位 14	C4IOS	0x0	rw	通道 4 空闲输出状态 (Channel 4 idle output state)
位 13	C3CIOS	0x0	rw	通道 3 互补空闲输出状态 (Channel 3 complementary idle output state)
位 12	C3IOS	0x0	rw	通道 3 空闲输出状态 (Channel 3 idle output state)
位 11	C2CIOS	0x0	rw	通道 2 互补空闲输出状态 (Channel 2 complementary idle output state)
位 10	C2IOS	0x0	rw	通道 2 空闲输出状态 (Channel 2 idle output state)
位 9	C1CIOS	0x0	rw	通道 1 互补空闲输出状态 (Channel 1 complementary idle output state) 输出关闭 (OEN = 0), 死区发生后: 0: C1COUT=0; 1: C1COUT=1.
位 8	C1IOS	0x0	rw	通道 1 空闲输出状态 (Channel 1 idle output state) 输出关闭 (OEN = 0), 死区发生后: 0: C1OUT=0. 1: C1OUT=1.
位 7	C1INSEL	0x0	rw	C1IN 选择 (C1IN selection) 0: CH1 引脚连到 C1IRAW 输入; 1: CH1、CH2 和 CH3 引脚异或结果连到 C1IRAW 输入。
位 6: 4	PTOS	0x0	rw	主定时器输出信号选择 (Primary TMR output selection) TMRx 输出到次定时器的信号选择: 000: 软件溢出或复位; 001: 使能; 010: 溢出; 011: 比较脉冲; 100: C1ORAW 信号; 101: C2ORAW 信号; 110: C3ORAW 信号; 111: C4ORAW 信号。
位 3	DRS	0x0	rw	DMA 请求源 (DMA request source) DMA 请求来源。 0: 通道事件; 1: 溢出事件。
位 2	CCFS	0x0	rw	通道控制位刷新选择 (Channel control bit refresh select) 对具有互补输出的通道, 如果通道控制位有缓存时: 0: 通过设置 HALLSWTR 位刷新控制位; 1: 通过设置 HALLSWTR 位或 TRGIN 的上升沿刷新控制位。
位 1	保留	0x0	resd	保持默认值。
位 0	CBCTRL	0x0	rw	通道缓存控制 (Channel buffer control) 对具有互补输出的通道: 0: CxEN, CxCEN 和 CxOCTRL 位无缓存; 1: CxEN, CxCEN 和 CxOCTRL 位有缓存。 注: 当 CBCTRL="1" 后, 仅当发生 HALL 事件时才会更新 CxEN, CxCEN 和 CxOCTRL 位。

15.4.4.3 TMR1、TMR8次定时器控制寄存器 (TMRx_STCTRL)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15	ESP	0x0	rw	外部信号极性 (External signal polarity) 用于选择外部方式。 0: 高电平或上升沿; 1: 低电平或下降沿。
位 14	ECMBEN	0x0	rw	外部时钟模式 B 使能 (External clock mode B enable) 用于启用外部时钟模式 B 0: 关闭; 1: 开启。
位 13: 12	ESDIV	0x0	rw	外部信号除频 (External signal divide) 用于选择降低外部触发频率的除频。 00: 关闭分频; 01: 2 分频; 10: 4 分频; 11: 8 分频。
位 11: 8	ESF	0x0	rw	外部信号滤波 (External signal filter) 用于过滤外部信号, 当外部信号产生了 N 次之后才能被采样。 0000: 无滤波器, 以 f_{DTS} 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。
位 7	STS	0x0	rw	次定时器同步 (Subordinate TMR synchronization) 该位开启后, 主次定时器可实现高度同步。 0: 关闭; 1: 开启。
位 6: 4	STIS	0x0	rw	次定时器输入选择 (Subordinate TMR input selection) 用于次定时器的输入选择。 000: 内部选择 0 (IS0) ; 001: 内部选择 1 (IS1) ; 010: 内部选择 2 (IS2) ; 011: 内部选择 3 (IS3) ; 100: C1IRAW 的输入检测器 (C1INC) ; 101: 滤波输入 1 (C1IFP1) ; 110: 滤波输入 2 (C2IFP2) ; 111: 外部输入 (EXT) 。 关于每个定时器中 ISx 的细节, 参见表 15-12。
位 3	保留	0x0	resd	保留, 保持默认值。
位 2: 0	SMSEL	0x0	rw	次定时器模式选择 (Subordinate TMR mode selection) 000: 关闭从模式; 001: 编码模式 A; 010: 编码模式 B; 011: 编码模式 C; 100: 复位模式 - TRGIN 输入上升沿时, 重新初始化计数器;

101: 挂起模式 - TRGIN 输入高电平时, 计数器计数;
 110: 触发模式 - TRGIN 输入上升沿时, 产生触发事件;
 111: 外部时钟模式 A - TRGIN 输入上升沿提供时钟;
 注: 编码器模式 A/B/C 配置方法请查看计数模式章节。

15.4.4.4 TMR1、TMR8 DMA/中断使能寄存器 (TMRx_IDEN)

域	简称	复位值	类型	功能
位 31: 15	保留	0x0 0000	resd	保持默认值。
位 14	TDEN	0x0	rw	触发 DMA 请求使能 (Trigger DMA request enable) 0: 关闭; 1: 开启。
位 13	HALLDE	0x0	rw	HALL DMA 请求使能 (HALL DMA request enable) 0: 关闭; 1: 开启。
位 12	C4DEN	0x0	rw	通道 4 的 DMA 请求使能 (Channel 4 DMA request enable) 0: 关闭; 1: 开启。
位 11	C3DEN	0x0	rw	通道 3 的 DMA 请求使能 (Channel 3 DMA request enable) 0: 关闭; 1: 开启。
位 10	C2DEN	0x0	rw	通道 2 的 DMA 请求使能 (Channel 2 DMA request enable) 0: 关闭; 1: 开启。
位 9	C1DEN	0x0	rw	通道 1 的 DMA 请求使能 (Channel 1 DMA request enable) 0: 关闭; 1: 开启。
位 8	OVFDEN	0x0	rw	溢出事件的 DMA 请求使能 (overflow event DMA request enable) 0: 关闭; 1: 开启。
位 7	BRKIE	0x0	rw	刹车中断使能 (Brake interrupt enable) 0: 关闭; 1: 开启。
位 6	TIEN	0x0	rw	触发中断使能 (Trigger interrupt enable) 0: 关闭; 1: 开启。
位 5	HALLIEN	0x0	rw	HALL 中断使能 (HALL interrupt enable) 0: 关闭; 1: 开启。
位 4	C4IEN	0x0	rw	通道 4 中断使能 (Channel 4 interrupt enable) 0: 关闭; 1: 开启。
位 3	C3IEN	0x0	rw	通道 3 中断使能 (Channel 3 interrupt enable) 0: 关闭; 1: 开启。
位 2	C2IEN	0x0	rw	通道 2 中断使能 (Channel 2 interrupt enable) 0: 关闭; 1: 开启。
位 1	C1IEN	0x0	rw	通道 1 中断使能 (Channel 1 interrupt enable) 0: 关闭; 1: 开启。
位 0	OVFIEN	0x0	rw	溢出中断使能 (Overflow interrupt enable) 0: 关闭; 1: 开启。

15.4.4.5 TMR1、TMR8中断状态寄存器 (TMRxISTS)

域	简称	复位值	类型	功能
位 31: 17	保留	0x0000	resd	保持默认值。
位 16	C5IF	0x0	rw0c	通道 5 中断标记 (Channel 5 interrupt flag) 比较事件发生时由硬件置'1'，由软件清'0'。 0: 无比较事件发生； 1: 发生比较事件。
位 15: 13	保留	0x0	resd	保持默认值。
位 12	C4RF	0x0	rw0c	通道 4 再捕获标记 (Channel 4 recapture flag) 见 C1RF 的描述。
位 11	C3RF	0x0	rw0c	通道 3 再捕获标记 (Channel 3 recapture flag) 见 C1RF 的描述。
位 10	C2RF	0x0	rw0c	通道 2 再捕获标记 (Channel 2 recapture flag) 见 C1RF 的描述。
位 9	C1RF	0x0	rw0c	通道 1 再捕获标记 (Channel 1 recapture flag) C1IF 的状态已经为'1'时是否再次发生了捕获，由硬件置'1'，写'0'清除。 0: 无捕获发生； 1: 捕获发生。
位 8	保留	0x0	resd	保持默认值。
位 7	BRKIF	0x0	rw0c	刹车中断标记 (Brake interrupt flag) 用于标记刹车输入的电平是否有效，由硬件置'1'，写'0'清除。 0: 无效； 1: 有效。
位 6	TRGIF	0x0	rw0c	触发中断标记 (Trigger interrupt flag) 当发生触发事件时由硬件置'1'，写'0'清除。 0: 无触发事件发生； 1: 发生触发事件。 触发事件：在 TRGIN 接收到有效边沿，或挂起模式下接收到任意边沿。
位 5	HALLIF	0x0	rw0c	HALL 中断标记 (HALL interrupt flag) 当发生触发事件时由硬件置'1'，写'0'清除。 0: 无 HALL 事件发生； 1: 发生 HALL 事件。 HALL 事件：CxEN、CxCEN、CxOCTRL 已被更新。
位 4	C4IF	0x0	rw0c	通道 4 中断标记 (Channel 4 interrupt flag) 见 C1IF 的描述。
位 3	C3IF	0x0	rw0c	通道 3 中断标记 (Channel 3 interrupt flag) 见 C1IF 的描述。
位 2	C2IF	0x0	rw0c	通道 2 中断标记 (Channel 2 interrupt flag) 见 C1IF 的描述。
位 1	C1IF	0x0	rw0c	通道 1 中断标记 (Channel 1 interrupt flag) 若通道 1 为输入模式时： 捕获事件发生时由硬件置'1'，由软件清'0'或读 TMRx_C1DT 清'0'。 0: 无捕获事件发生； 1: 发生捕获事件。 若通道 1 为输出模式时： 比较事件发生时由硬件置'1'，由软件清'0'。 0: 无比较事件发生； 1: 发生比较事件。
位 0	OVFIF	0x0	rw0c	溢出中断标记 (Overflow interrupt flag) 当溢出事件发生时由硬件置'1'，由软件清'0'。 0: 无溢出事件发生； 1: 发生溢出事件，若 TMRx_CTRL1 的 OVREN=0、OVFS=0 时： - 当 TMRx_SWEVT 寄存器的 OVFSWTR=1 时产生溢出事件； - 当计数值 CVAL 被触发事件重初始化时产生溢出事件。

15.4.4.6 TMR1、TMR8软件事件寄存器 (TMRx_SWEVT)

域	简称	复位值	类型	功能
位 31: 8	保留	0x00 0000	resd	保持默认值。
位 7	BRKSWTR	0x0	wo	软件触发刹车事件 (Brake event triggered by software) 通过软件触发一个刹车事件。 0: 无作用; 1: 制造一个刹车事件。
位 6	TRGSWTR	0x0	wo	软件触发触发事件 (Trigger event triggered by software) 通过软件触发一个触发事件。 0: 无作用; 1: 制造一个触发事件。
位 5	HALLSWTR	0x0	wo	软件触发 HALL 事件 (HALL event triggered by software) 通过软件产生一个 HALL 事件。 0: 无作用; 1: 产生一个 HALL 事件。 注: 该位只对拥有互补输出的通道有效。
位 4	C4SWTR	0x0	wo	软件触发通道 4 事件 (Channel 4 event triggered by software) 见 C1M 的描述。
位 3	C3SWTR	0x0	wo	软件触发通道 3 事件 (Channel 3 event triggered by software) 见 C1M 的描述。
位 2	C2SWTR	0x0	wo	软件触发通道 2 事件 (Channel 2 event triggered by software) 见 C1M 的描述。
位 1	C1SWTR	0x0	wo	C1SWTR: 软件触发通道 1 事件 (Channel 1 event triggered by software) 通过软件触发一个通道 1 事件。 0: 无作用; 1: 制造一个通道 1 事件。
位 0	OVFSWTR	0x0	wo	软件触发溢出事件 (Overflow event triggered by software) 通过软件触发一个溢出事件。 0: 无作用; 1: 制造一个溢出事件。

15.4.4.7 TMR1、TMR8通道模式寄存器1 (TMRx_CM1)

通道可用于输入（捕获模式）或输出（比较模式），通道的方向由相应的 CxC 位定义。该寄存器其它位的作用在输入和输出模式下不同。CxOx 描述了通道在输出模式下的功能，CxIx 描述了通道在输入模式下的功能。因此必须注意，同一个位在输出模式和输入模式下的功能是不同的。

输出比较模式

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15	C2OSEN	0x0	rw	通道 2 输出开关使能 (Channel 2 output switch enable)
位 14: 12	C2OCTRL	0x0	rw	通道 2 输出控制 (Channel 2 output control)
位 11	C2OBEN	0x0	rw	通道 2 输出缓存使能 (Channel 2 output buffer enable)
位 10	C2OIEN	0x0	rw	通道 2 输出立即使能 (Channel 2 output immediately enable)
通道 2 配置 (Channel 2 configure) 当 C2EN=’0’时，这些位用于选择通道 2 为输出或输入，以及输入时的映射选择：				
位 9: 8	C2C	0x0	rw	00: 输出； 01: 输入，C2IN 映射在 C2IFP2 上； 10: 输入，C2IN 映射在 C1IFP2 上； 11: 输入，C2IN 映射在 STI 上，只有在 STIS 选择内部触发输入时才工作。
位 7	C1OSEN	0x0	rw	通道 1 输出开关使能 (Channel 1 output switch enable) 0: EXT 输入不影响 C1ORAW； 1: 当 EXT 输入高电平时，将 C1ORAW 清 0。
通道 1 输出控制 (Channel 1 output control) 这些位用于设置原始信号 C1ORAW 的工作状态。 000: 断开。断开 C1ORAW 到 C1OUT 的输出； 001: 当 TMRx_CVAL=TMRx_C1DT 时，设置 C1ORAW 为高。 010: 当 TMRx_CVAL=TMRx_C1DT 时，设置 C1ORAW 为低。 011: 当 TMRx_CVAL=TMRx_C1DT 时，切换 C1ORAW 的电平。 100: 固定 C1ORAW 为低。 101: 固定 C1ORAW 为高。 110: PWM 模式 A –OWCDIR=0, 若 TMRx_C1DT>TMRx_CVAL 时设置 C1ORAW 为高，否则为低； –OWCDIR=1, 若 TMRx_C1DT<TMRx_CVAL 时设置 C1ORAW 为低，否则为高。 111: PWM 模式 B –OWCDIR=0, 若 TMRx_C1DT>TMRx_CVAL 时设置 C1ORAW 为低，否则为高； –OWCDIR=1, 若 TMRx_C1DT<TMRx_CVAL 时设置 C1ORAW 为高，否则为低。 注：除’000’外，其余配置下 C1OUT 将连接到 C1ORAW，C1OUT 的输出电平除了会根据 C1ORAW 变化外，还与 CCTRL 所配置的输出极性有关。				
位 6: 4	C1OCTRL	0x0	rw	通道 1 输出缓存使能 (Channel 1 output buffer enable) 0: 关闭 TMRx_C1DT 的缓存功能，写入 TMRx_C1DT 的内容会立即生效。 1: 启用 TMRx_C1DT 的缓存功能，写入 TMRx_C1DT 的内容将保存到缓存寄存器中，当发生溢出事件时再更新到 TMRx_C1DT 中。
位 3	C1OBEN	0x0	rw	通道 1 输出立即使能 (Channel 1 output immediately enable) 在 PWM 模式 A 或模式 B 下，该位能够缩短触发事件到通道 1 的输出响应间的时间。 0: 需要比较 CVAL 与 C1DT 的值之后再产生输出。
位 2	C1OIEN	0x0	rw	

				1: 无需比较 CVAL 与 C1DT 的值, 当发生触发事件时立即产生输出。
位 1: 0	C1C	0x0	rw	通道 1 配置 (Channel 1 configure) 当 C1EN='0'时, 这些位用于选择通道 1 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C1IN 映射在 C1IFP1 上; 10: 输入, C1IN 映射在 C2IFP1 上; 11: 输入, C1IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。
输入模式				
域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 12	C2DF	0x0	rw	通道 2 滤波器 (Channel 2 digital filter)
位 11: 10	C2IDIV	0x0	rw	通道 2 分频系数 (Channel 2 input divider)
位 9: 8	C2C	0x0	rw	通道 2 配置 (Channel 2 configure) 当 C2EN='0'时, 这些位用于选择通道 2 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C2IN 映射在 C2IFP2 上; 10: 输入, C2IN 映射在 C1IFP2 上; 11: 输入, C2IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。
位 7: 4	C1DF	0x0	rw	通道 1 滤波器 (Channel 1 digital filter) 这些位用于配置通道 1 的滤波器。滤波的个数为 N, 则表示发生了 N 次采样事件后输入边沿才能通过滤波器: 0000: 无滤波器, 以 f_{DTS} 采样 0001: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=2 0010: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=4 0011: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=8 0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=6 0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=8 0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=6 0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=8 1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=6 1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=8 1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=5 1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=6 1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=8 1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=5 1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=6 1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=8
位 3: 2	C1IDIV	0x0	rw	通道 1 分频系数 (Channel 1 input divider) 这些位定义了通道 1 的分频系数。 00: 不分频, 每一个有效的边沿都会产生一次输入; 01: 每 2 个有效的边沿产生一次输入; 10: 每 4 个有效的边沿产生一次输入; 11: 每 8 个有效的边沿产生一次输入。 注: C1EN='0'时, 分频系数复位。
位 1: 0	C1C	0x0	rw	通道 1 配置 (Channel 1 configure) 当 C1EN='0'时, 这些位用于选择通道 1 为输出或输入, 以及输入时的映射选择: 00: 输出; 01: 输入, C1IN 映射在 C1IFP1 上; 10: 输入, C1IN 映射在 C2IFP1 上; 11: 输入, C1IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。

15.4.4.8 TMR1、TMR8通道模式寄存器2 (TMRx_CM2)

参看以上 CM1 寄存器描述

输出比较模式

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15	C4OSEN	0x0	rw	通道 4 输出开关使能 (Channel 4 output switch enable)
位 14: 12	C4OCTRL	0x0	rw	通道 4 输出控制 (Channel 4 output control)
位 11	C4OBEN	0x0	rw	通道 4 输出缓存使能 (Channel 4 output buffer enable)
位 10	C4OIEN	0x0	rw	通道 4 输出立即使能 (Channel 4 output immediately enable)
通道 4 配置 (Channel 4 configure) 当 C4EN='0'时, 这些位用于选择通道 4 为输出或输入, 以及输入时的映射选择:				
位 9: 8	C4C	0x0	rw	00: 输出; 01: 输入, C4IN 映射在 C4IFP4 上; 10: 输入, C4IN 映射在 C3IFP4 上; 11: 输入, C4IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。
位 7	C3OSEN	0x0	rw	通道 3 输出开关使能 (Channel 3 output switch enable)
位 6: 4	C3OCTRL	0x0	rw	通道 3 输出控制 (Channel 3 output control)
位 3	C3OBEN	0x0	rw	通道 3 输出缓存使能 (Channel 3 output buffer enable)
位 2	C3OIEN	0x0	rw	通道 3 输出立即使能 (Channel 3 output immediately enable)
通道 3 配置 (Channel 3 configure) 当 C3EN='0'时, 这些位用于选择通道 3 为输出或输入, 以及输入时的映射选择:				
位 1: 0	C3C	0x0	rw	00: 输出; 01: 输入, C3IN 映射在 C3IFP3 上; 10: 输入, C3IN 映射在 C4IFP3 上; 11: 输入, C3IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。

输入模式

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 12	C4DF	0x0	rw	通道 4 滤波器 (Channel 4 digital filter)
位 11: 10	C4IDIV	0x0	rw	通道 4 分频系数 (Channel 4 input divider)
通道 4 配置 (Channel 4 configure) 当 C4EN='0'时, 这些位用于选择通道 4 为输出或输入, 以及输入时的映射选择:				
位 9: 8	C4C	0x0	rw	00: 输出; 01: 输入, C4IN 映射在 C4IFP4 上; 10: 输入, C4IN 映射在 C3IFP4 上; 11: 输入, C4IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。
位 7: 4	C3DF	0x0	rw	通道 3 滤波器 (Channel 3 digital filter)
位 3: 2	C3IDIV	0x0	rw	通道 3 分频系数 (Channel 3 input divider)
通道 3 配置 (Channel 3 configure) 当 C3EN='0'时, 这些位用于选择通道 3 为输出或输入, 以及输入时的映射选择:				
位 1: 0	C3C	0x0	rw	00: 输出; 01: 输入, C3IN 映射在 C3IFP3 上; 10: 输入, C3IN 映射在 C4IFP3 上; 11: 输入, C3IN 映射在 STCI 上, 只有在 STIS 选择内部触发输入时才工作。

15.4.4.9 TMR1、TMR8通道控制寄存器 (TMRx_CCTRL)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15	C4CP	0x0	rw	通道 4 互补极性 (Channel 4 complementary polarity) 见 C1CP 的描述。
位 14	C4CEN	0x0	rw	通道 4 互补使能 (Channel 4 complementary enable) 见 C1CEN 的描述。
位 13	C4P	0x0	rw	通道 4 极性 (Channel 4 polarity) 见 C1P 的描述。
位 12	C4EN	0x0	rw	通道 4 使能 (Channel 4 enable) 见 C1EN 的描述。
位 11	C3CP	0x0	rw	通道 3 互补极性 (Channel 3 complementary polarity) 见 C1CP 的描述。
位 10	C3CEN	0x0	rw	通道 3 互补使能 (Channel 3 complementary enable) 见 C1CEN 的描述。
位 9	C3P	0x0	rw	通道 3 极性 (Channel 3 polarity) 见 C1P 的描述。
位 8	C3EN	0x0	rw	通道 3 使能 (Channel 3 enable) 见 C1EN 的描述。
位 7	C2CP	0x0	rw	通道 2 互补极性 (Channel 2 complementary polarity) 见 C1CP 的描述。
位 6	C2CEN	0x0	rw	通道 2 互补使能 (Channel 2 complementary enable) 见 C1CEN 的描述。
位 5	C2P	0x0	rw	通道 2 极性 (Channel 2 polarity) 见 C1P 的描述。
位 4	C2EN	0x0	rw	通道 2 使能 (Channel 2 enable) 见 C1EN 的描述。
位 3	C1CP	0x0	rw	通道 1 互补极性 (Channel 1 complementary polarity) 0: C1COUT 的有效电平为高 1: C1COUT 的有效电平为低
位 2	C1CEN	0x0	rw	通道 1 互补使能 (Channel 1 complementary enable) 0: 禁止输出; 1: 使能输出。
位 1	C1P	0x0	rw	通道 1 极性 (Channel 1 polarity) 通道 1 配置为输出: 0: C1OUT 的有效电平为高 1: C1OUT 的有效电平为低 通道 1 配置为输入: C1CP/C1P 位共同定义输入信号有效沿。 00: C1IN 的有效边沿为上升沿; 作为外部触发使用时, C1IN 不反相。 01: C1IN 的有效边沿为下降沿; 作为外部触发使用时, C1IN 反相。 10: 保留 11: C1IN 的有效边沿为上升沿和下降沿; 作为外部触发使用时, C1IN 不反相。
位 0	C1EN	0x0	rw	通道 1 使能 (Channel 1 enable) 0: 禁止输入或输出; 1: 使能输入或输出。

表 15-15 带刹车功能的互补输出通道CxOUT和CxCOUT的控制位

控制位					输出状态 (1)	
OEN 位	FCSODIS 位	FCSOEN 位	CxEN 位	CxCEN 位	CxOUT 输出状态	CxCOUT 输出状态
1	X	0	0	0	输出禁止 (与定时器断开) CxOUT=0, CxEN=0	输出禁止 (与定时器断开) CxOUT=0, CxCEN=0
		0	0	1	输出禁止 (与定时器断开) CxOUT=0, CxEN=0	CxORAW + 极性, CxOUT=CxORAW xor CxCP, CxCEN=1
		0	1	0	CxORAW+极性, CxOUT=CxORAW xor CxP, CxEN=1	输出禁止 (与定时器断开) CxOUT=0, CxCEN=0
		0	1	1	CxORAW+极性+死区, CxEN=1	CxORAW 反相+极性+死区, CxCEN=1
		1	0	0	输出禁止 (与定时器断开) CxOUT=CxP, CxEN=0	输出禁止 (与定时器断开) CxOUT=CxCP, CxCEN=0
		1	0	1	关闭状态 (输出使能且为无效电平) CxOUT=CxP, CxEN=1	CxORAW + 极性, CxOUT=CxORAW xor CxCP, CxCEN=1
		1	1	0	CxORAW + 极性, CxOUT=CxORAW xor CxP, CxEN=1	关闭状态 (输出使能且为无效电平) CxOUT=CxCP, CxCEN=1
		1	1	1	CxORAW+极性+死区, CxEN=1	CxORAW 反相+极性+死区, CxCEN=1
0	X	0	0	0	输出禁止 (对应 IO 与定时器断开, IO 浮空) 异步地: CxOUT=CxP, CxEN=0, CxCOUT=CxCP, CxCEN=0; 若时钟存在: 经过一个死区时间后 CxOUT=CxIOS, CxCOUT=CxCIOS, 假设 CxIOS 与 CxCIOS 并不都对应 CxOUT 和 CxCOUT 的有效电平。	
		0	0	1		
		0	1	0		
		0	1	1		
		1	0	0		
		1	0	1	CxEN=CxCEN=0 时: 输出禁止 (对应 IO 与定时器断开, IO 浮空); 其它情况下: 关闭状态 (对应通道输出无效电平) 异步地: CxOUT =CxP, CxEN=1, CxCOUT=CxCP, CxCEN=1;	
		1	1	0		
		1	1	1		

注意: 如果一个通道的 2 个输出都没有使用 ($CxEN = CxCEN = 0$) , 那么 $CxIOS$, $CxCIOS$, CxP 和 $CxCP$ 都必须清零。

注意: 引脚连接到互补的 CxOUT 和 CxCOUT 通道的外部 I/O 引脚的状态, 取决于 CxOUT 和 CxCOUT 通道状态和 GPIO 以及 IOMUX 寄存器。

15.4.4.10 TMR1、TMR8计数值寄存器 (TMRx_CVAL)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	CVAL	0x0000	rw	计数值 (Counter value)

15.4.4.11 TMR1、TMR8预分频器寄存器 (TMRx_DIV)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	DIV	0x0000	rw	分频系数 (Divider value) 计数器时钟频率 $f_{CK_CNT} = f_{TMR_CLK} / (DIV[15: 0]+1)$ 溢出事件发生时该寄存器值被传送到实际的预分频寄存器中。

15.4.4.12 TMR1、TMR8周期寄存器 (TMRx_PR)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	PR	0xFFFF	rw	周期值 (Period value) 定时器计数的周期值。当周期值为 0 时，定时器不工作。

15.4.4.13 TMR1、TMR8重复周期寄存器 (TMRx_RPR)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	RPR	0x0000	rw	重复周期的次数 (Repetition of period value) 这些位用于减慢溢出事件发生的速率，当重复周期的次数减为 0 时才会发生溢出事件。

15.4.4.14 TMR1、TMR8通道1数据寄存器 (TMRx_C1DT)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	C1DT	0x0000	rw	通道 1 数据寄存器值 (Channel 1 data register) 若通道 1 配置为输入： C1DT 是前一次通道 1 输入事件 (C1IN) 所保存的 CVAL。 若通道 1 配置为输出： C1DT 是将要和 CVAL 进行比较的值，写入的值是否会立即生效取决于输出缓存使能位 (C1OBEN)，并根据设置在 C1OUT 上产生相应的输出。

15.4.4.15 TMR1、TMR8通道2数据寄存器 (TMRx_C2DT)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	C2DT	0x0000	rw	通道 2 数据寄存器值 (Channel 2 data register) 若通道 2 配置为输入： C2DT 是前一次通道 2 输入事件 (C2IN) 所保存的 CVAL。 若通道 2 配置为输出： C2DT 是将要和 CVAL 进行比较的值，写入的值是否会立即生效取决于输出缓存使能位 (C2OBEN)，并根据设置在 C2OUT 上产生相应的输出。

15.4.4.16 TMR1、TMR8通道3数据寄存器 (TMRx_C3DT)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	C3DT	0x0000	rw	通道 3 数据寄存器值 (Channel 3 data register) 若通道 3 配置为输入: C3DT 是前一次通道 3 输入事件 (C3IN) 所保存的 CVAL。 若通道 3 配置为输出: C3DT 是将要和 CVAL 进行比较的值, 写入的值是否会立即生效取决于输出缓存使能位 (C3OBEN), 并根据设置在 C3OUT 上产生相应的输出。

15.4.4.17 TMR1、TMR8通道4数据寄存器 (TMRx_C4DT)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	C4DT	0x0000	rw	通道 4 数据寄存器值 (Channel 4 data register) 若通道 4 配置为输入: C4DT 是前一次通道 4 输入事件 (C4IN) 所保存的 CVAL。 若通道 4 配置为输出: C4DT 是将要和 CVAL 进行比较的值, 写入的值是否会立即生效取决于输出缓存使能位 (C4OBEN), 并根据设置在 C4OUT 上产生相应的输出。

15.4.4.18 TMR1、TMR8刹车寄存器 (TMRx_BRK)

域	简称	复位值	类型	功能
位 31: 20	保留	0x000	resd	保持默认值
位 19: 16	BRKF	0x0	rw	刹车输入滤波 (Brake input filter) 这些位用于配置刹车输入的滤波器。滤波的个数为 N, 则表示发生了 N 次采样事件后输入边沿才能通过滤波器: 0000: 无滤波器, 以 f_{DTS} 采样 0001: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=2 0010: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=4 0011: 采样频率 $f_{SAMPLING} = f_{CK_INT}$, N=8 0100: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=6 0101: 采样频率 $f_{SAMPLING} = f_{DTS}/2$, N=8 0110: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=6 0111: 采样频率 $f_{SAMPLING} = f_{DTS}/4$, N=8 1000: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=6 1001: 采样频率 $f_{SAMPLING} = f_{DTS}/8$, N=8 1010: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=5 1011: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=6 1100: 采样频率 $f_{SAMPLING} = f_{DTS}/16$, N=8 1101: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=5 1110: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=6 1111: 采样频率 $f_{SAMPLING} = f_{DTS}/32$, N=8
位 15	OEN	0x0	rw	输出使能 (Output enable) 对配置为输出的通道, 该位用于使能 CxOUT 和 CxCOUT 的输出。 0: 关闭; 1: 开启。
位 14	AOEN	0x0	rw	输出自动使能 (Automatic output enable) 用于溢出事件时将 OEN 自动置'1' 0: 关闭; 1: 开启
位 13	BRKV	0x0	rw	刹车输入信号的有效性 (Brake input validity) 用于选择刹车输入信号的输入有效电平: 0: 低电平; 1: 高电平。

位 12	BRKEN	0x0	rw	刹车功能使能 (Brake enable) 用于开启刹车功能。 0: 关闭; 1: 开启。
位 11	FCSOEN	0x0	rw	总输出开时的冻结状态 (Frozen channel status when holistic output enable) 该位用于配置具有互补输出的通道，在定时器不工作且 OEN=1 时的通道状态。 0: 关闭 CxOUT/CxCOUT 输出; 1: 开启 CxOUT/CxCOUT 输出，输出为无效电平。
位 10	FCSODIS	0x0	rw	总输出关时的冻结状态 (Frozen channel status when holistic output disable) 该位用于配置具有互补输出的通道，在定时器不工作且 OEN=0 时的通道状态。 0 : 关闭 CxOUT/CxCOUT 输出; 1 : 开启 CxOUT/CxCOUT 输出，输出为空闲电平。
位 9: 8	WPC	0x0	rw	写保护配置 (Write protected configuration) 该位用于配置写保护。 00: 写保护关闭; 01: 3 级写保护，以下位受写保护: TMRx_BRK: BRKF、DTC、BRKEN、BRKV 和 AOEN TMRx_CTRL2: CxIOS 和 CxCIOs 10: 2 级写保护，除 3 级写保护的内容外，以下位也受写保护: TMRx_CCTRL: CxP 和 CxCP TMRx_BRK: FCSODIS 和 FCSOEN 11: 1 级写保护，除 2 级写保护的内容外，以下位也受写保护: TMRx_CMx: CxOCTRL 和 CxOBEN 注: WPC>0 时将无法再次被修改，直到系统复位。
位 7: 0	DTC	0x00	rw	死区配置 (Dead-time configuration) 这些位用于配置死区时间。取 DTC[7: 0]的高 3 位为功能选择位: 0xx: DT = DTC [7: 0] * TDTS; 10x: DT = (64+ DTC [5: 0]) * TDTS * 2; 110: DT = (32+ DTC [4: 0]) * TDTS * 8; 111: DT = (32+ DTC [4: 0]) * TDTS * 16;

注意：根据锁定设置，BRKF、AOEN、BRKV、BRKEN、FCSODIS、FCSOEN 和 DTC[7: 0]位均可被写保护，有必要在第一次写入 TMRx_BRK 寄存器时对它们进行配置。

15.4.4.19 TMR1、TMR8 DMA控制寄存器 (TMRx_DMACTRL)

域	简称	复位值	类型	功能
位 31: 13	保留	0x0 0000	resd	保持默认值。
位 12: 8	DTB	0x00	rw	DMA 传输字节 (DMA transfer bytes) 这些位定义了传输的字节个数： 00000: 1 个字节 00001: 2 个字节 00010: 3 个字节 00011: 4 个字节 10000: 17 个字节 10001: 18 个字节
位 7: 5	保留	0x0	resd	保持默认值。
位 4: 0	ADDR	0x00	rw	DMA 传输地址偏移 (DMA transfer address offset) ADDR 定义了从 TMRx_CTRL1 所在地址开始的偏移量： 00000: TMRx_CTRL1, 00001: TMRx_CTRL2, 00010: TMRx_STCTRL,

15.4.4.20 TMR1、TMR8 DMA数据寄存器（TMRx_DMADT）

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	DMADT	0x0000	rw	DMA 传输的数据寄存器（DMA data register） 通过对 DMADT 寄存器的读写能够实现对任意 TMR 寄存器的操作，其操作的寄存器地址范围是：TMRx 外设地址 + ADDR*4 至 TMRx 外设地址 + ADDR*4 + DTB*4。

15.4.4.21 TMR1、TMR8通道模式寄存器3（TMRx_CM3）

域	简称	复位值	类型	功能
位 31: 8	保留	0x00 0000	resd	保持默认值。
位 7	C5OSEN	0x0	rw	通道 5 输出开关使能（Channel 5 output switch enable）
位 6: 4	C5OCTRL	0x0	rw	通道 5 输出控制（Channel 5 output control）
位 3	C5OBEN	0x0	rw	通道 5 输出缓存使能（Channel 5 output buffer enable）
位 2	C5OIEN	0x0	rw	通道 5 输出立即使能（Channel 5 output immediately enable）
位 1: 0	保留	0x0	resd	保持默认值。

15.4.4.22 TMR1、TMR8通道5数据寄存器（TMRx_C5DT）

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	C5DT	0x0000	rw	通道 5 数据寄存器值（Channel 5 data register） C5DT 是将要和 CVAL 进行比较的值，写入的值是否会立即生效取决于输出缓存使能位（C5OBEN），并根据设置在 C5OUT 上产生相应的输出。

16 窗口看门狗 (WWDT)

16.1 WWDT简介

当程序正常运行时，需在一个有限的时间窗口内重载窗口看门狗递减计数器，用来避免看门狗电路产生系统复位，以此来监测系统是否正常运行。

窗口看门狗时钟由 APB1_CLK 分频而来，由于 APB1_CLK 的精确性，窗口看门狗可对有限的时间窗口精确控制。

16.2 WWDT主要特性

- 7位递减计数器
- 启动看门狗后，当递减计数器的值小于 0x40 或是在窗口外被重新装载产生系统复位。
- 可以通过重载计数器中断重装载计数器。

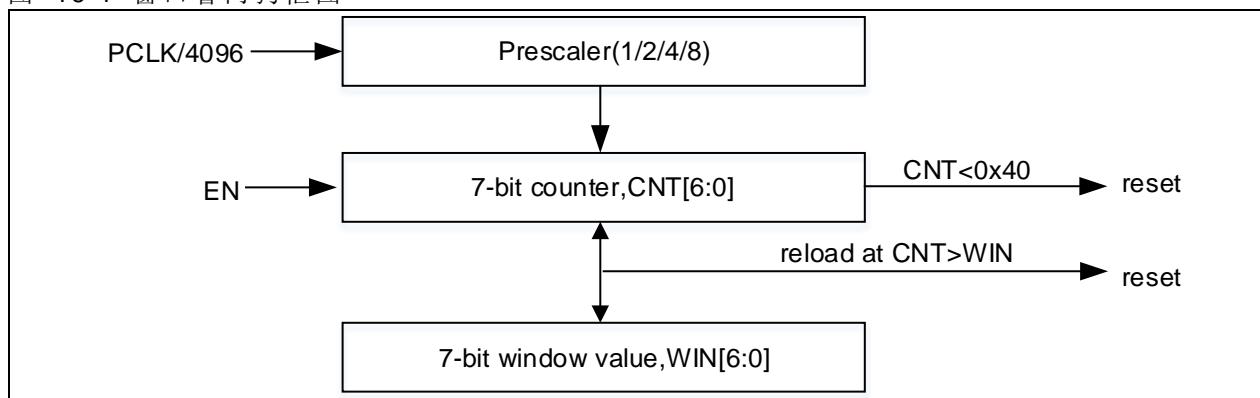
16.3 WWDT功能描述

启动窗口看门狗后，窗口看门狗可在以下两种情况下产生系统复位：

第一种，7位递减计数器值由 0x40 变为 0x3F。

第二种，7位递减计数器值大于 7位窗口值时，重载计数器值。

图 16-1 窗口看门狗框图



为避免重载计数器值时产生复位，应在计数器值小于窗口值大于 0x40 时重载计数器值。

WWDT 计数器时钟由 APB1_CLK 分频得到，分频系数可通过配置 WWDT_CFG 寄存器 DIV[1: 0]改变。计数器值决定了 WWDT 复位前的最大计数周期数，结合 WIN[6: 0] 可灵活的调整重载窗口。

WWDT 提供了重载计数器中断功能，开启后，WWDT 将在计数值达到 0x40h 时将 RLDF 标志位置 1，同时产生重载计数器中断，可在中断服务程序中重载计数器值，以避免发生系统复位。需要注意的是，若在 CNT[6] 为 0 时，将 WWDTEN 置 1 会产生一个系统复位，因此当写入 WWDT_CTRL 寄存器时，应始终保持 CNT[6] 为 1，避免使能窗口看门狗后立即产生一个系统复位。

窗口看门狗超时时间 T_{WWDT} 可由以下公式计算，其中 T_{PCLK1} 为 APB1 时钟周期，单位为 ms：

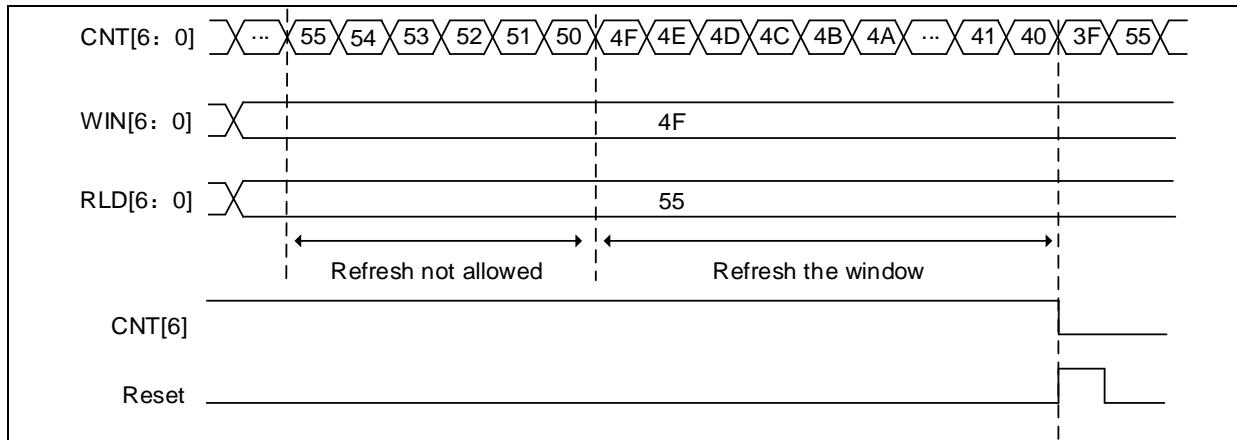
$$T_{WWDT} = T_{PCLK1} \times 4096 \times 2^{DIV[1: 0]} \times (CNT[5: 0] + 1); \quad (\text{ms})$$

下表给出了当 PCLK1 频率为 72MHz 时，最大和最小看门狗超时时间。

表 16-1 PCLK1频率为 72MHz 时，最大和最小看门狗超时时间

时钟预分频值	最小超时时间	最大超时时间
0	56.5 μs	3.64ms
1	113.5 μs	7.28ms
2	227.5 μs	14.56ms
3	455 μs	29.12ms

图 16-2 窗口看门狗时序图



调试模式

微控制器处于调试模式时，意味着 Cortex®-M4F 核心停止。将 DEBUG 模块中 WWDT_PAUSE 位置 1 可将 WWDT 计数器计数暂停。详见 [31_2](#) 节。

16.4 WWDT 寄存器

必须以字（32 位）的方式操作这些外设寄存器。

表 16-2 WWDT 寄存器映像和复位值

寄存器简称	基址偏移量	复位值
WWDT_CTRL	0x00	0x0000 007F
WWDT_CFG	0x04	0x0000 007F
WWDT_STS	0x08	0x0000 0000

16.4.1 控制寄存器 (WWDT_CTRL)

域	简称	复位值	类型	功能
位 31: 8	保留	0x000000	resd	保持默认值。
位 7	WWDTEN	0x0	rw1s	窗口看门狗使能 (Window watchdog enable) 0: 关闭; 1: 启用。 该位由软件置起，只能在复位后自动清零。
位 6: 0	CNT	0x7F	rw	递减计数器 (Decrement counter) 当计数器递减到 0x3F 时产生复位。

16.4.2 配置寄存器 (WWDT_CFG)

域	简称	复位值	类型	功能
位 31: 10	保留	0x000000	resd	保持默认值。
位 9	RLDIEN	0x0	rw1s	重载计数器中断 (Reload counter interrupt) 0: 关闭; 1: 启开。
位 8: 7	DIV	0x0	rw	时钟预分频值 (Clock division value) 00: PCLK1 除以 4096; 01: PCLK1 除以 8192; 10: PCLK1 除以 16384; 11: PCLK1 除以 32768。
位 6: 0	WIN	0x7F	rw	窗口值 (Window value) 当计数器值大于窗口值时, 此时重载计数器会产生复位, 重载计数器区间为 0x40~WIN[6: 0]

16.4.3 状态寄存器 (WWDT_STS)

域	简称	复位值	类型	功能
位 31: 1	保留	0x0000 0000	resd	保持默认值。
位 0	RLDF	0x0	rw0c	重载计数器中断标志 (Reload counter interrupt flag) 当递减计数器为 0x40 时, 该标志会置位。 该位被硬件置起, 由软件将其清零。

17 看门狗 (WDT)

17.1 WDT简介

看门狗由专用低速时钟 (LICK) 驱动，由于 LICK 时钟精度较低，因此看门狗适用于低时间精度、能够独立于主程序之外的应用。

17.2 WDT主要特性

- 12位递减计数器
- 计数器由LICK时钟驱动（可在深睡眠和待机模式下工作）
- 可选择在DEEPSLEEP、STANDBY模式下是否停止计数
- 支持两种复位方式：
 - 当递减计数器递减至0。
 - 当递减计数器在窗口外被重新装载。

17.3 WDT功能描述

WDT 启动方式:

WDT 的启动方式有两种，分别为软件启动和硬件启动。软件启动通过向 WDT_CMD 寄存器写入 0xCCCC 实现；硬件启动则需通过配置用户系统数据区来实现，使能硬件看门狗后，看门狗将在上电复位后自动开始运行。

WDT 复位条件:

当 WDT 计数器值递减至 0 时将产生 WDT 系统复位，因此需定时向 WDT_CMD 寄存器写入 0xAAAA 重载计数器值。此外，若将 WIN[11: 0]设置为非默认值 (0xFFFF) 将开启窗口看门狗功能，在计数值大于窗口值时重载计数器值将会产生系统复位。

WDT 写保护:

WDT_DIV、WDT_RLD、WDT_WIN 寄存器受写保护，向 WDG_CMD 寄存器写入 0x5555 可解锁寄存器写保护，之后可对其进行配置。这三个寄存器的更新状态分别由 WDT_STS 寄存器中 DIVF、RLDF、WINF 指示。向 WDG_CMD 寄存器写入其它值将重新启动 WDT_DIV、WDT_RLD、WDT_WIN 寄存器写保护。向 WDG_CMD 寄存器写入 0xAAAA 也会启动寄存器写保护。

WDT 时钟:

WDT 计数器由 LICK 时钟驱动，LICK 是内部 RC 时钟，范围为 30kHz~60kHz 之间，所以超时时间也是在一定区间内，使用时应注意在超时时间配置上应该留有余量，如果需要获得较为精确的看门狗超时时间，可对 LICK 进行校准，有关 LICK 校准的问题，详见 4.1.1 节。

WDT 低功耗计数模式:

WDT 能够在 SLEEP、DEEPSLEEP、STANDBY 模式下运行，用户可选择进入 DEEPSLEEP、STANDBY 模式后计数器是否停止计数，可由用户系统数据区中的 nDEPSLP_WDT、nSTDBY_WDT 位配置。

如果设置了停止计数，当进入了 DEEPSLEEP、STANDBY 模式后，看门狗计数器停止递减，意味着看门狗在这两种低功耗模式下不会发生复位，当从这两种模式唤醒后，计数器从进入时的值继续递减。

图 17-1 看门狗框图

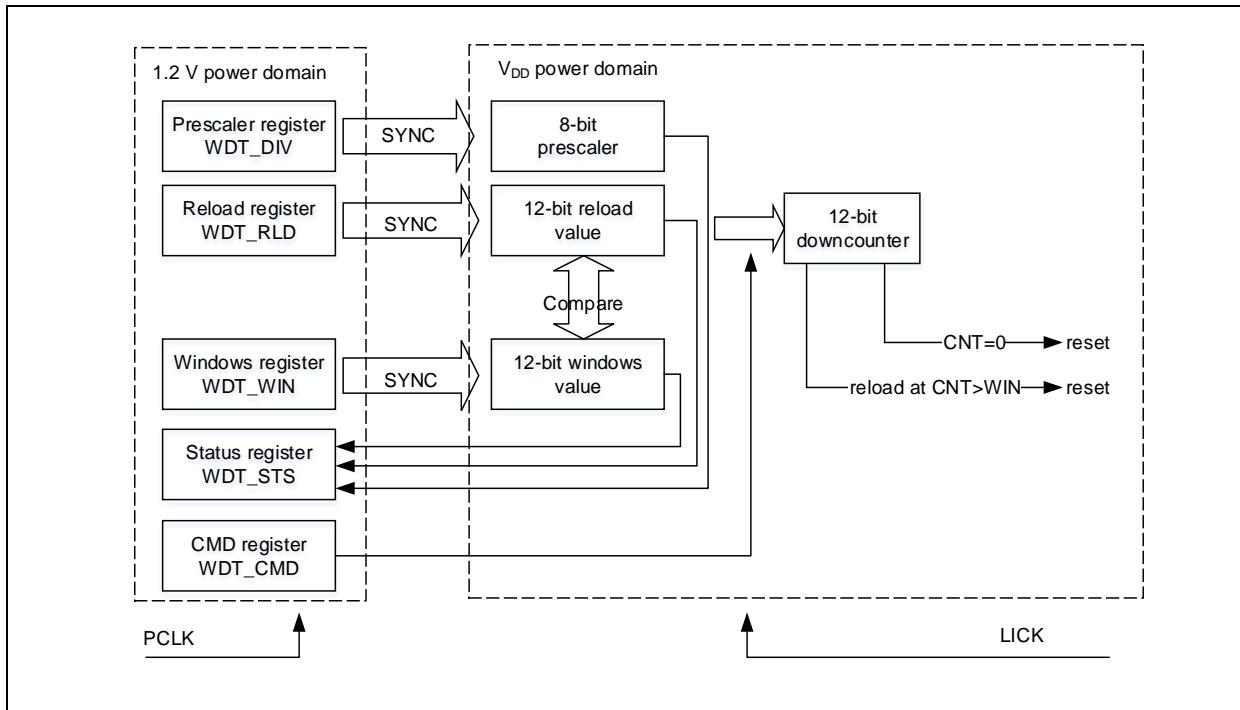


表 17-1 看门狗超时时间（当LICK=40kHz时）

预分频系数	DIV[2: 0]位	最短时间 (ms)	
		RLD[11: 0] = 0x000	RLD[11: 0] = 0xFFFF
/4	0	0.1	409.6
/8	1	0.2	819.2
/16	2	0.4	1638.4
/32	3	0.8	3276.8
/64	4	1.6	6553.6
/128	5	3.2	13107.2
/256	(6 或 7)	6.4	26214.4

17.4 调试模式

当微控制器处于调试模式时，意味着 Cortex®-M4F 核心停止。此时将 DEBUG 模块中 WDT_PAUSE 位置 1 会暂停 WDT 计数器计数。详见 31.2 节。

17.5 WDT寄存器

必须以字（32位）的方式操作这些外设寄存器。

表 17-2 WDT寄存器映像和复位值

寄存器简称	基址偏移量	复位值
WDT_CMD	0x00	0x0000 0000
WDT_DIV	0x04	0x0000 0000
WDT_RLD	0x08	0x0000 0FFF
WDT_STS	0x0C	0x0000 0000
WDT_WIN	0x10	0x0000 0FFF

17.5.1 命令寄存器（WDT_CMD）

（在待机模式复位）

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。 命令寄存器（Command register） 0xAAAA: 重载计数器； 0x5555: 解锁 WDT_DIV、WDT_RLD、WDT_WIN 写保护； 0xCCCC: 启动看门狗，如果使能了硬件看门狗，则不需要执行此操作。
位 15: 0	CMD	0x0000	wo	

17.5.2 预分频寄存器（WDT_DIV）

（待机模式时不复位）

域	简称	复位值	类型	功能
位 31: 3	保留	0x0000 0000	resd	保持默认值。 递减计数器时钟预分频值（Clock division value） 000: LICK 除以 4; 001: LICK 除以 8; 010: LICK 除以 16; 011: LICK 除以 32; 100: LICK 除以 64; 101: LICK 除以 128; 110: LICK 除以 256; 111: LICK 除以 256。 只有解锁写保护后才能写此寄存器，只有当 DIVF 为 0 时，才能读取此寄存器。
位 2: 0	DIV	0x0	rw	

17.5.3 重装载寄存器（WDT_RLD）

（待机模式时不复位）

域	简称	复位值	类型	功能
位 31: 12	保留	0x00000	resd	保持默认值。 重载值（Reload value）
位 11: 0	RLD	0xFFFF	rw	只有解锁写保护后才能写此寄存器，只有当 RLDF 为 0 时，才能读取此寄存器。

17.5.4 状态寄存器 (WDT_STS)

(在待机模式复位)

域	简称	复位值	类型	功能
位 31: 3	保留	0x0000 0000	resd	保持默认值。
位 2	WINF	0x0	ro	窗口值更新完成标志 (Window value update complete flag) 0: 更新完成; 1: 正在更新。 只有当 RLDF 为 0 时才能写 WDT_WIN 寄存器。
位 1	RLDF	0x0	ro	重载值更新完成标志 (Reload value update complete flag) 0: 更新完成; 1: 正在更新。 只有当 RLDF 为 0 时才能写 WDT_RLD 寄存器。
位 0	DIVF	0x0	ro	分频值更新完成标志 (Division value update complete flag) 0: 更新完成; 1: 正在更新。 只有当 DIVF 为 0 时才能写 WDT_DIV 寄存器。

17.5.5 窗口寄存器 (WDT_WIN)

(待机模式不复位)

域	简称	复位值	类型	功能
位 31: 12	保留	0x000000	resd	保持默认值。
位 11: 0	WIN	0xFFFF	rw	窗口值 (Window value) 当计数器值大于窗口值时, 此时重载计数器会产生复位, 重载计数器区间为 0~窗口值。

18 实时时钟 (ERTC)

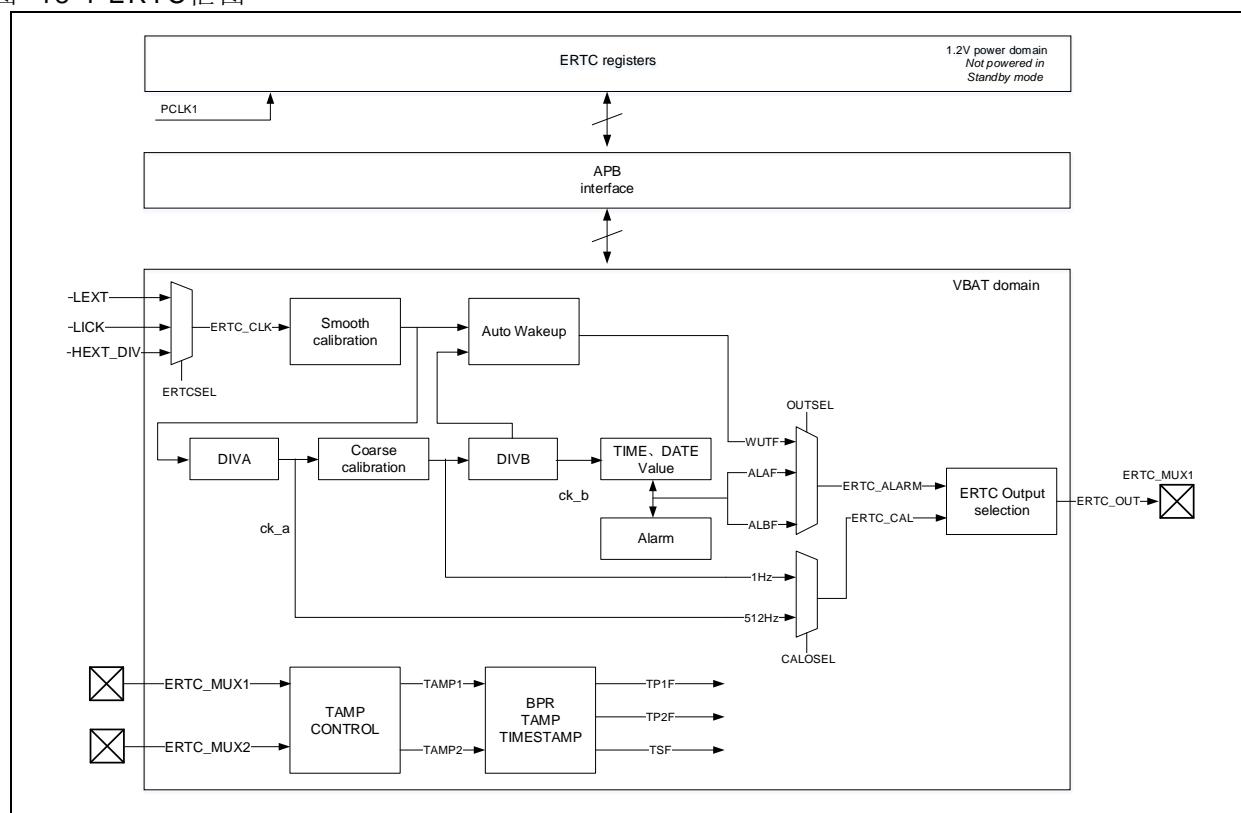
18.1 ERTC简介

实时时钟用于配置日历时钟，修改 ERTC 中日历寄存器值可以修改系统的当前时间和日期。ERTC 计数逻辑位于电池供电域，只要电池供电域有电，ERTC 便会一直运行，不受系统复位以及 VDD 掉电影响。

18.2 ERTC主要特性

- 功能强大的实时日历，自动处理月份天数28（平年2月）、29（闰年2月）、30（小月）、31（大月），其中当年份寄存器是4的倍数时为闰年，支持两组闹钟
- 周期性唤醒
- 参考时钟检测
- 两组可配置入侵检测，支持时间戳功能
- 支持精密校准、粗略校准两种校准
- 20个电池供电寄存器
- 5组中断：闹钟A、闹钟B、周期性唤醒、入侵检测、时间戳
- 复用功能输出，校准时钟输出、闹钟事件或唤醒事件
- 复用功能输入，参考时钟输入、两路入侵检测、时间戳

图 18-1 ERTC框图



18.3 ERTC功能说明

18.3.1 ERTC时钟

ERTC_CLK 可从 LEXT、LICK、分频后的 HEXT 中选择（由 CRM_BPDC 寄存器 ERTCSSEL[1: 0]配置），HEXT 的分频值由 CRM_CFG 寄存器 ERTCDIV[4: 0]配置。

ERTC 内置分频器 A 和分频器 B，分别由 DIVA[6: 0]、DIVB[14: 0]配置，推荐 DIVA 配置为较高的值，以最大程度降低功耗。ERTC_CLK 依次经由分频器 A、分频器 B 处理，得到 ck_a、ck_b 时钟，ck_a 用于更新亚秒，ck_b 用于更新日历和周期性唤醒。ck_a、ck_b 时钟频率可由下式计算：

$$f_{ck_a} = \frac{f_{ERTC_CLK}}{DIVA + 1}$$

$$f_{ck_b} = \frac{f_{ERTC_CLK}}{(DIVB + 1) \times (DIVA + 1)}$$

当配置 DIVA=127, DIVB=255, 且 ERTC_CLK 选用 32.768kHz 的 LEXT 时, 可得到 1Hz 的 ck_b, 用于更新日历。

注意: 若 ERTC_CLK 选用分频后的 HEXT, 应先配置 HEXT 分频值, 再将时钟源切换为 HEXT。

18.3.2 ERTC初始化

寄存器解锁:

上电复位后所有 ERTC 寄存器处于写保护状态, 需要先解除写保护, 才能写配置 ERTC 寄存器(除 ERTC_STS[14: 8]、ERTC_TAMP 和 ERTC_BPRx 寄存器外, 其它寄存器位均受写保护, 且写保护不受系统复位影响)。

解锁步骤:

- 1、使能电源接口时钟: CRM_APB1EN 的 PWCEN=1
- 2、解锁电池供电域写保护: PWC_CTRL 的 BPWEN=1
- 3、依次向 ERTC_WP 寄存器写入 0xCA, 0x53, 若向 ERTC_WP 寄存器写入错误的值, 将重新激活写保护。

下表列出了需要解除写保护和进入初始化模式才可以配置的 ERTC 寄存器:

表 18-1 ERTC 寄存器配置表

寄存器简称	是否受 ERTC_WP 写保护	是否需要进入初始化模式	其它
ERTC_TIME	是	是	-
ERTC_DATE	是	是	-
ERTC_CTRL	是	位 7、6、4 需要	-
ERTC_STS	除位[14: 8]外	-	-
ERTC_DIV	是	是	-
ERTC_WAT	是	否	WATWF 为 1 时可配置
ERTC_CCAL	是	是	-
ERTC_ALA	是	否	ALAWF 为 1 时可配置
ERTC_ALB	是	否	ALBWF 为 1 时可配置
ERTC_WP	-	-	-
ERTC_SBS	-	-	-
ERTC_TADJ	是	否	TADJF 为 0 时可配置
ERTC_TSTM	-	-	-
ERTC_TSDT	-	-	-
ERTC_TSSBS	-	-	-
ERTC_SCAL	是	否	CALUPDF 为 0 时可配置
ERTC_TAMP	否	否	-
ERTC_ALASBS	是	否	ALAWF 为 1 时可配置
ERTC_ALBSBS	是	否	ALBWF 为 1 时可配置
ERTC_BPRx	否	否	-

时钟、日历初始化:

寄存器解锁后, 时钟和日历的初始化配置可按以下步骤进行:

1. 将 IMEN 位置 1 进入初始化模式。

2. 等待初始化标志位 INITF 置 1。
3. 依次配置 DIVB、DIVA。
4. 配置时钟和日历值。
5. 将 IMEN 位清 0 退出初始化模式，等待 UPDF 置 1，表明日历值同步完成，日历开始计数。

为了方便时间微调，ERTC 还提供了夏令时和时间调整功能。

夏令时功能：用于增加（ADD1H=1）或减小（DEC1H=1）1 小时，而无需重新进行初始化配置。

时间调整功能：用于精确的调整当前时钟。若只配置 DECSBS[14: 0]值，该值将会加到分频器 B 计数器值中，时钟因此产生延迟；若只将 ADD1S 位置 1，当前时钟将增加 1 秒；若同时配置 DECSBS[14: 0]，ADD1S 位，时钟将增加零点几秒。

延迟时间（ADD1S=0）：延迟时间=DECSBS/(DIVB+1)；

提前时间（ADD1S=1）：提前时间=1-(DECSBS/(DIVB+1))。

注：设置时间调整寄存器前，必须先确认 SBS[15]=0，以免亚秒发生上溢；参考时钟检测与粗略数字校准功能不能同时使用：当 RC DEN=1 时，粗略数字校准功能不可用。

日历读取：

ERTC 提供两种日历访问方式，分别为同步读取（DREN=0）和异步读取（DREN=1）。

同步读取时，ERTC 通过 PCLK1 访问同步的影子寄存器来获取时钟和日历值。影子寄存器值由位于电池供电域的 ERTC 日历值同步而来，同步周期为两个 ERTC_CLK，同步完成后 UPDF 将置 1。影子寄存器由系统复位来复位。为保证读取的 ERTC_SBS、ERTC_TIME、ERTC_DATE 寄存器值来自同一时刻，读取低阶寄存器时会将高阶寄存器值锁定，直到读取 ERTC_DATE 寄存器。例如读取 ERTC_SBS，会将 ERTC_TIME、ERTC_DATE 寄存器值锁定。

异步读取时，ERTC 通过 PCLK1 直接读取位于电池供电域的 ERTC 时钟和日历值，这样避免了由于同步时间带来的误差。异步读取时，UPDF 标志将由硬件清 0。为保证异步读取时钟和日历值的准确性，软件必须两次读取时钟和日历值，并比较两次结果是否一致，如果不一致应该再读，直到两次结果一致，另外，也可以只比较两次结果的最低位来判定。

注：在 STANDBY 和 DEEPSLEEP 模式下，当前日历值不会复制到影子寄存器中，当从这两种模式唤醒时，必须先设置 UPDF=0，然后等待 UPDF=1，以保证读取的日历值是最新的；在同步读取时，需保证 PCLK1 频率至少为 ERTC_CLK 频率 7 倍；异步读取时，需要额外一个 APB 周期才能完成读取日历寄存器的指令。

闹钟初始化：

ERTC 包含闹钟 A、闹钟 B 两个闹钟，并分别提供了闹钟 A 中断、闹钟 B 中断。

闹钟值由 ERTC_ALASBS/ERTC_ALA (ERTC_ALBSBS/ERTC_ALB) 设定，开启闹钟后，当设定的闹钟值匹配日历值时将触发闹钟事件。通过 MASKx 位，可选择性的屏蔽日历字段，被屏蔽的字段不参与闹钟匹配。

闹钟 A、B 的配置可按以下步骤进行：

1. 关闭闹钟 A 或闹钟 B（设置 ALAEN=0 或 ALBEN=0）。
2. 等待闹钟 A 或闹钟 B 寄存器允许写（等待 ALAWF 或 ALBWF 位置 1）。
3. 配置闹钟 A 或闹钟 B 寄存器（ERTC_ALA/ERTC_ALASBS、ERTC_ALB/ERTC_ALBSBS）。
4. 使能闹钟 A 或闹钟 B（设置 ALAEN=1 或 ALBEN=1）。

注：当 ERTC_ALA 或 ERTC_ALB 中 MASK1 为 0 时，DIVB 至少为 3 才能使闹钟正常工作。

18.3.3 周期性自动唤醒

周期性唤醒功能用于 ERTC 周期性的自动唤醒低功耗模式，唤醒周期由 VAL[15: 0]设定 (WATCLK[2]=1 时扩展为 17 位，唤醒计数值为 $VAL + 2^{16}$)。当唤醒计数器值由 VAL 值递减至 0 时，WATF 标志置 1，产生唤醒事件，同时唤醒计数器值重载 VAL 值。若使能周期性唤醒中断，将产生周期性唤醒中断。

驱动唤醒定时器的时钟通过 WATCLK[2: 0]设定，可选 16/8/4/2 分频后的 ERTC_CLK 或 ck_b(通常 1Hz)，结合唤醒计数值可灵活调整唤醒周期。

周期性唤醒功能可按以下步骤进行配置：

1. 关闭周期性唤醒 (设置 WATEN=0)。
2. 等待唤醒自动重载定时器和 WATCLK[2: 0]位允许写 (WATWF 位置 1)。
3. 配置唤醒定时器计数值和唤醒时钟 (VAL[15: 0]、WATCLK[2: 0]位)。
4. 使能定时器 (设置 WATEN=1)。

注：系统复位以及低功耗模式（睡眠、深睡眠和待机）对唤醒定时器没有任何影响。

注：DEBUG 模式下，若唤醒时钟选择 ERTC_CLK，用于周期性唤醒的计数器正常运行。

18.3.4 ERTC 校准

ERTC 提供了两种校准方法：粗略校准和精密校准。但两种校准方法不能同时使用。

粗略数字校准：

粗略数字校准通过增加或减少 ck_a 周期值来实现提前或推迟更新日历的功能。

正校准时 (CALDIR=0)，在 64 分钟的前 $2 \times CALVAL$ 分钟时间内，每分钟 (约 15360 个 ck_a 周期) 插入 2 个 ck_a 周期，相当于提前更新日历。

负校准时 (CALDIR=1)，在 64 分钟前的 $2 \times CALVAL$ 分钟时间内，每分钟 (约 15360 个 ck_a 周期) 忽略 1 个 ck_a 周期，相当于推迟更新日历。

注：粗略数字校准至少要将 DIVA 值设置为 6。

精密数字校准：

区别于粗略数字校准，精密校准的校准效果更好且校准更加均匀。开启精密校准校准功能后，将均匀增加或减少 ERTC_CLK 来达到校准的目的。

当 ERTC_CLK 为 32.768kHz 时，精密校准周期约为 2^{20} 个 ERTC_CLK(32 秒)。DEC[8: 0]值指定了 2^{20} 个 ERTC_CLK 中忽略的脉冲数，最多可忽略 511 个脉冲；将 ADD 置 1，可在 2^{20} 个 ERTC_CLK 中插入 512 个脉冲。两者搭配使用，可在 2^{20} 个 ERTC_CLK 周期进行 -511~+512 的调整。

有效校准频率 F_{SCAL} ：

$$F_{SCAL} = F_{ERTC_CLK} \times [1 + \frac{ADD \times 512 - DEC}{2^{20} + DEC - ADD \times 512}]$$

当分频器 A 值小于 3 时，会按照 ADD 等于 0 校准。此时应降低分频器 B 值来实现每秒增加 8 个 ERTC_CLK，也就是 32 秒增加 256 个 ERTC_CLK 搭配 DEC[8: 0]位，可在 2^{20} 个 ERTC_CLK 周期进行 -255~+256 的调整。

此时有效校准频率 F_{SCAL} ：

$$F_{SCAL} = F_{ERTC_CLK} \times [1 + \frac{256 - DEC}{2^{20} + DEC - 256}]$$

精密数字校准的校准周期还可选择 8 秒或 16 秒(由 CAL8 和 CAL16 配置)，8 秒校准周期的优先级更高，同时使能 8 秒和 16 秒校准周期，将优先选择 8 秒校准周期。

ERTC 提供了 CALUPDF 标志用来指示校准值的状态，当配置 ERTC_SCAL 寄存器时，CALUPDF 标志位将置 1，指示校准值正在更新；当校准值被成功应用后，标志位自动清 0，指示校准值更新完成。

18.3.5 参考时钟检测

为保证日历长时间运行的精确性，ERTC 提供了时钟同步功能（低功耗模式不可用），用精度更高的参考时钟（一般用 50Hz 或者 60Hz 的市电）校准更新日历的 1Hz 时钟。

参考时钟检测功能开启后，在每次更新日历值的前 7 个 ck_a 周期检测参考时钟边沿，若检测到边沿，将使用此边沿更新日历值，后续采用 3 个 ck_a 周期检测参考时钟边沿。每一次检测到参考时钟边沿时，都会将分频器 A 的值进行重载，这使得内部 1Hz 的日历时钟与参考时钟边沿刚好对齐，当内部 1Hz 时钟出现微小偏移时，利用更精确的参考时钟，将 1Hz 时钟微调至与参考时钟边沿对齐。当没有检测到参考时钟边沿时，ERTC 会利用原来的时钟源更新日历。

需要注意的是，使能参考时钟功能后，需要将 DIVA、DIVB 设置为复位值 (0x7F、0xFF)，并且时钟同步功能不能与粗校准功能同时开启。

18.3.6 时间戳

时间戳功能用于在发生时间戳事件时（入侵引脚检测到有效边沿），将当前的日历值保存到时间戳寄存器中。

当发生时间戳时，TSF 位置 1，此时若再次发生时间戳事件，TSOF 标志位将置 1，但时间戳寄存器并不会更新，可以通过 TSIEN 位设置是否使能时间戳中断。

时间戳用法有两种

1. 单独的时间戳功能，此时入侵检测引脚用来检测时间戳，使用步骤：

选择时间戳引脚（设置 TSPIN）

选择上升沿还是下降沿触发（设置 TSEDG）

使能时间戳（设置 TSEN=1）

2. 发生入侵事件时保存时间戳，使用步骤：

配置入侵检测相关寄存器

使能发生入侵事件时保存时间戳（设置 TPTSEN=1）

注：发生时间戳事件后，TSF 在两个 ck_a 周期后置 1，建议在 TSF 已置 1 的情况下轮询
TSOF 位

18.3.7 入侵检测

ERTC 提供了两组入侵检测 TAMP1 和 TAMP2，且两组入侵检测可分别配置为滤波后的电平检测或边沿检测。TAMP1 通过 TSPIN 选择入侵引脚 ERTC_MUX1 或 ERTC_MUX2；TAMP2 只能选择 ERTC_MUX2。当检测到有效的入侵事件后，TP1F 或 TP2F 将置 1，若已使能了入侵检测中断，将产生对应的中断；若 TPTSEN 位已置 1，将同时产生时间戳事件。为保证位于电池供电域中的电池供电寄存器数据安全，入侵事件发生时将复位电池供电寄存器。

边沿入侵检测配置步骤：

1. 选择入侵检测方式为边沿检测（TPFLT=00），并选择有效沿（TP1EDG 或 TP2EDG）。
2. 根据需要配置是否在入侵事件时激活时间戳（TPTSEN 置 1）。
3. 根据需要开启入侵中断使能（TPIEN 置 1）。
4. 若选择 TAMP1 进行入侵检测，选择 TAMP1 映射关系为 ERTC_MUX1 或是 ERTC_MUX2（TP1PIN）并将 TAMP1 使能（TP1EN 置 1）；若选择 TAMP2 进行入侵检测，则只需使能 TAMP2（TP2EN 置 1）

电平入侵检测配置步骤：

1. 选择入侵检测方式为电平检测，并选择有效电平采样次数（TPFLT≠00）。
2. 选择入侵有效电平（TP1EDG 或 TP2EDG）。
3. 选择入侵检测采样频率（TPFREQ）。
4. 根据需要开启入侵检测上拉（TPPU 置 1），若开启，还需配置上拉电阻预充电时间（TPPR）。
5. 根据需要配置是否在入侵事件时激活时间戳（TPTSEN 置 1）。
6. 根据需要开启入侵中断使能（TPIEN 置 1）。
7. 若选择 TAMP1 进行入侵检测，选择 TAMP1 映射关系为 ERTC_MUX1 或是 ERTC_MUX2（TP1PIN）并将 TAMP1 使能（TP1EN 置 1）；若选择 TAMP2 进行入侵检测，则只需使能 TAMP2（TP2EN 置 1）

当配置为边沿检测时，有以下两点要注意：

1. 在使能入侵检测前，若入侵检测已被配置为上升沿有效，且入侵检测引脚已为高电平，在使能入侵检测后会立刻产生一个入侵检测事件。
2. 在使能入侵检测前，若入侵检测已被配置为下降沿有效，且入侵检测引脚已为低电平，在使能入侵检测后会立刻产生一个入侵检测事件。

注：当 VDD 电源关闭时，入侵检测仍有效。

注：standby 模式下，TAMP2（PA0 引脚）无法使用预充电功能。

18.3.8 复用功能输出

ERTC 提供了一组复用功能输出，可以输出以下事件：

- 校准后的时钟 (OUTSEL=0, CALOEN=1)

输出 512Hz (CALOSEL=0)

输出 1Hz (CALOSEL=1)

- 闹钟A (OUTSEL=1)

- 闹钟B (OUTSEL=2)

- 唤醒事件 (OUTSEL=3)

当输出闹钟事件或者唤醒事件时 (OUTSEL≠0)，可以通过 OUTTYPE 选择输出类型为开漏或是推挽，可以通过 OUTP 配置输出极性。

18.3.9 ERTC唤醒

ERTC 可由闹钟、周期性唤醒、时间戳、入侵事件进行唤醒，使能 ERTC 中断可按以下操作配置：

- 将 ERTC 对应中断的 EXINT 线配置为中断模式并使能，有效沿选择上升沿。
- 使能 ERTC 中断对应的 NVIC 通道。
- 使能对应的 ERTC 中断。

下表说明了 ERTC 时钟源、事件以及中断对唤醒低功耗模式的影响：

表 18-2 ERTC 唤醒低功耗模式

时钟源	事件	唤醒 SLEEP	唤醒 DEEPSLEEP	唤醒 STANDBY
HEXT	闹钟 A	√	×	×
	闹钟 B	√	×	×
	周期性唤醒	√	×	×
	时间戳	√	×	×
	入侵事件	√	×	×
LICK	闹钟 A	√	√	√
	闹钟 B	√	√	√
	周期性唤醒	√	√	√
	时间戳	√	√	√
	入侵事件	√	√	√
LEXT	闹钟 A	√	√	√
	闹钟 B	√	√	√
	周期性唤醒	√	√	√
	时间戳	√	√	√
	入侵事件	√	√	√

表 18-3 中断控制位

中断事件	事件标志	中断使能位	EXINT 线
闹钟 A	ALAF	ALAIEN	17
闹钟 B	ALBF	ALBIEN	17
周期性唤醒	WATF	WATIEN	22
时间戳	TSF	TSIEN	21
入侵事件	TP1F/TP2F	TP1EN	21

18.4 ERTC寄存器

必须以字（32位）的方式操作这些外设寄存器。

ERTC 寄存器是 32 位可寻址寄存器，具体描述如下：

表 18-4 ERTC-寄存器映像和复位值

寄存器简称	基址偏移量	复位值
ERTC_TIME	0x00	0x0000 0000
ERTC_DATE	0x04	0x0000 2101
ERTC_CTRL	0x08	0x0000 0000
ERTC_STS	0x0C	0x0000 0007
ERTC_DIV	0x10	0x007F 00FF
ERTC_WAT	0x14	0x0000 FFFF
ERTC_CCAL	0x18	0x0000 0000
ERTC_ALA	0x1C	0x0000 0000
ERTC_ALB	0x20	0x0000 0000
ERTC_WP	0x24	0x0000 0000
ERTC_SBS	0x28	0x0000 0000
ERTC_TADJ	0x2C	0x0000 0000
ERTC_TSTM	0x30	0x0000 0000
ERTC_TSDT	0x34	0x0000 000D
ERTC_TSSBS	0x38	0x0000 0000
ERTC_SCAL	0x3C	0x0000 0000
ERTC_TAMP	0x40	0x0000 0000
ERTC_ALASBS	0x44	0x0000 0000
ERTC_ALBSBS	0x48	0x0000 0000
ERTC_BPRx	0x50-0x9C	0x0000 0000

18.4.1 ERTC时间寄存器(ERTC_TIME)

域	简称	复位值	类型	功能
位 31: 23	保留	0x000	resd	保持默认值。
位 22	AMPM	0x0	rw	上午/下午 (AM/PM) 0: 上午; 1: 下午。 注：该位只用于 12 小时制，24 小时制保持为 0。
位 21: 20	HT	0x0	rw	小时十位 (Hour tens)
位 19: 16	HU	0x0	rw	小时个位 (Hour units)
位 15	保留	0x0	resd	保持默认值。
位 14: 12	MT	0x0	rw	分钟十位 (Minute tens)
位 11: 8	MU	0x0	rw	分钟个位 (Minute units)
位 7	保留	0x0	resd	保持默认值
位 6: 4	ST	0x0	rw	秒钟十位 (Second tens)
位 3: 0	SU	0x0	rw	秒钟个位 (Second units)

18.4.2 ERTC日期寄存器(ERTC_DATE)

域	简称	复位值	类型	功能
位 31: 24	保留	0x00	resd	保持默认值。
位 23: 20	YT	0x0	rw	年份十位 (Year tens)
位 19: 16	YU	0x0	rw	年份个位 (Year units)
				星期 (Week) 0: 禁用; 1: 星期一; 2: 星期二; 3: 星期三; 4: 星期四; 5: 星期五; 6: 星期六; 7: 星期日。
位 15: 13	WK	0x1	rw	
位 12	MT	0x0	rw	月份十位 (Month tens)
位 11: 8	MU	0x1	rw	月份个位 (Month units)
位 7: 6	保留	0x0	resd	保持默认值。
位 5: 4	DT	0x0	rw	日期十位 (Date tens)
位 3: 0	DU	0x1	rw	日期个位 (Date units)

18.4.3 ERTC控制寄存器(ERTC_CTRL)

域	简称	复位值	类型	功能
位 31: 24	保留	0x00	resd	保持默认值。
位 23	CALOEN	0x0	rw	校准输出使能 (Calibration output enable) 0: 关闭; 1: 开启。
位 22: 21	OUTSEL	0x0	rw	输出源选择 (Output source selection) 00: 关闭; 01: 阵列 A; 10: 阵列 B; 11: 唤醒事件。
位 20	OUTP	0x0	rw	输出极性 (Output polarity) 0: 高; 1: 低。
位 19	CALOSEL	0x0	rw	校准输出选择 (Calibration output selection) 0: 512Hz; 1: 1Hz。
位 18	BPR	0x0	rw	电池供电域数据寄存器 (Battery power domain data register) 该位在电池供电域, 不受系统复位影响, 可用来存储夏令时操作或者一些其他需要一直保存的数据。
位 17	DEC1H	0x0	wo	减少 1 小时 (Decrease 1 hour) 0: 无作用; 1: 减少一小时。 注: 当小时不为 0 时才有效, 该位置 1 后 (不要在小时递增时置 1), 下一秒生效。
位 16	ADD1H	0x0	wo	增加 1 小时 (Add 1 hour) 0: 无作用; 1: 增加一小时。 注: 该位置 1 后 (不要在小时递增时置 1), 下一秒生效。
位 15	TSIEN	0x0	rw	时间戳中断使能 (Timestamp interrupt enable) 0: 关闭; 1: 开启。
位 14	WATIEN	0x0	rw	唤醒定时器中断使能 (Wakeup timer interrupt enable) 0: 关闭; 1: 开启。
位 13	ALBIEN	0x0	rw	闹钟 B 中断使能 (Alarm B interrupt enable) 0: 关闭; 1: 开启。

位 12	ALAIEN	0x0	rw	闹钟 A 中断使能 (Alarm A interrupt enable) 0: 关闭; 1: 开启。
位 11	TSEN	0x0	rw	时间戳使能 (Timestamp enable) 0: 关闭; 1: 开启。
位 10	WATEN	0x0	rw	唤醒定时器使能 (Wakeup timer enable) 0: 关闭; 1: 开启。
位 9	ALBEN	0x0	rw	闹钟 B 使能 (Alarm B enable) 0: 关闭; 1: 开启。
位 8	ALAEN	0x0	rw	闹钟 A 使能 (Alarm A enable) 0: 关闭; 1: 开启。
位 7	CCALEN	0x0	rw	粗略校准使能 (Coarse calibration enable) 0: 关闭; 1: 开启。
位 6	HM	0x0	rw	小时模式 (Hour mode) 0: 24 小时制; 1: 12 小时制。
位 5	DREN	0x0	rw	日期/时间寄存器直接读取使能 (Date/time register direct read enable) 0: 关闭, ERTC_TIME、ERTC_DATE、ERTC_SBS 值从同步寄存器获取, 每两个 ERTC_CLK 更新一次; 1: 开启, ERTC_TIME、ERTC_DATE、ERTC_SBS 值从电池供电域获取。
位 4	RCDEN	0x0	rw	参考时钟检测使能 (Reference clock detection enable) 0: 关闭; 1: 开启。
位 3	TSEDG	0x0	rw	时间戳触发边沿 (Timestamp trigger edge) 0: 上升沿; 1: 下降沿。
位 2: 0	WATCLK	0x0	rw	唤醒定时器时钟选择 (Wakeup timer clock selection) 000: ERTC_CLK/16; 001: ERTC_CLK/8; 010: ERTC_CLK/4; 011: ERTC_CLK/2; 10x: ck_b; 11x: ck_b, 唤醒计数值增加 2^{16} , 唤醒时间 = ERTC_WAT + 2^{16} 。 注: 在 WATEN=0 且 WATWF=1 时可对这些位进行写操作。

18.4.4 ERTC 初始化和状态寄存器(ERTC_STS)

域	简称	复位值	类型	功能
位 31: 17	保留	0x0000	resd	保持默认值。
位 16	CALUPDF	0x0	ro	校准值更新完成标志 (Calibration value update completed flag) 0: 更新完成; 1: 正在更新。 当对精密校准寄存器 ERTC_SCAL 写时，该位自动置 1，当 ERTC 使用新的校准值时，该位自动清零，当该位为 1 时，不能写 ERTC_SCAL 寄存器。
位 15	保留	0x0	resd	保持默认值。
位 14	TP2F	0x0	rw0c	入侵检测 2 标志 (Tamper detection 2 flag) 0: 无入侵事件发生; 1: 有入侵事件发生。
位 13	TP1F	0x0	rw0c	入侵检测 1 标志 (Tamper detection 1 flag) 0: 无入侵事件发生; 1: 有入侵事件发生。
位 12	TSOF	0x0	rw0c	时间戳溢出标志 (Timestamp overflow flag) 0: 正常; 1: 溢出。 当产生了时间戳事件 (TSF 置 1) 时，又产生了时间戳事件，该标志置 1。
位 11	TSF	0x0	rw0c	时间戳标志 (Timestamp flag) 0: 无时间戳事件发生; 1: 有时间戳事件发生。 当读取了时间戳，并清除了 TSF 时，建议再检查 TSOF 标志，因为可能会存在当正在清除 TSF 时又产生了时间戳事件。 注：该位清 0 后 2 个 APB_CLK 后生效。
位 10	WATF	0x0	rw0c	唤醒定时器标志 (Wakeup timer flag) 0: 无唤醒事件发生; 1: 有唤醒事件发生。 注：该位清 0 后 2 个 APB_CLK 后生效。
位 9	ALBF	0x0	rw0c	闹钟 B 标志 (Alarm clock B flag) 0: 无闹钟事件发生; 1: 有闹钟事件发生。 注：该位清 0 后 2 个 APB_CLK 后生效。
位 8	ALAF	0x0	rw0c	闹钟 A 标志 (Alarm clock A flag) 0: 无闹钟事件发生; 1: 有闹钟事件发生。 注：该位清 0 后 2 个 APB_CLK 后生效。
位 7	IMEN	0x0	rw	初始化模式使能 (Initialization mode enable) 0: 关闭; 1: 开启。 当进入了初始化模式后，日历处于停止状态。
位 6	IMF	0x0	ro	进入初始化模式标志 (Enter initialization mode flag) 0: 未进入; 1: 进入。 当使能了初始化模式 (INITEN=1)，并进入了初始化模式 (INITEF=1) 时，才能更改 ERTC_TIME、ERTC_DATE、ERTC_DIV 寄存器。
位 5	UPDF	0x0	rw0c	日历更新标志 (Calendar update flag) 0: 正在更新; 1: 更新完成。 每当从电池供电域将日历更新到 ERTC_TIME、ERTC_DATE、ERTC_SBS 同步寄存器，该标志置 1，每两个 ERTC_CLK 更新一次。
位 4	INITF	0x0	ro	日历初始化标志 (Calendar initialization flag) 0: 未初始化; 1: 已初始化。

				当检测到 ERTC_DATE 里面的年份不为 0 时该位置 1，当年份为 0 时，该位清 0。
位 3	TADJF	0x0	ro	时间调整标志 (Time adjustment flag) 0: 无操作； 1: 正在执行时间调整。 当对时间调整存器 ERTC_TADJ 写时，该位自动置 1，当时间调整结束后，该位自动清零。
位 2	WATWF	0x1	ro	唤醒定时器寄存器允许写标志 (Wakeup timer register allows write flag) 0: 不允许写； 1: 允许写。
位 1	ALBWF	0x1	ro	闹钟 B 允许写标志 (Alarm B register allows write flag) 0: 不允许写； 1: 允许写。
位 0	ALAWF	0x1	ro	闹钟 A 允许写标志 (Alarm A register allows write flag) 0: 不允许写； 1: 允许写。

18.4.5 ERTC预分频器寄存器(ERTC_DIV)

域	简称	复位值	类型	功能
位 31: 23	保留	0x000	resd	保持默认值。
位 22: 16	DIVA	0x7F	rw	分频器 A (Diveder A)
位 15	保留	0x0	resd	保持默认值。
位 14: 0	DIVB	0x00FF	rw	分频器 B (Diveder B) 日历时钟=ERTC_CLK/((DIVA+1)x(DIVB+1))。

18.4.6 ERTC唤醒定时器寄存器(ERTC_WAT)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	VAL	0xFFFF	rw	唤醒定时器重载值 (Wakeup timer reload value)

18.4.7 ERTC粗校准寄存器(ERTC_CCAL)

域	简称	复位值	类型	功能
位 31: 8	保留	0x000000	resd	保持默认值。
位 7	CALDIR	0x0	rw	校准方向 (Calibration direction) 0: 正校准； 1: 负校准。
位 6: 5	保留	0x0	resd	保持默认值。
位 4: 0	CALVAL	0x00	rw	校准值 (Calibration value) 正校准： 00000: +0 ppm 00001: +4 ppm 00010: +8 ppm 11111: +126 ppm 负校准： 00000: -0 ppm 00001: -2 ppm 00010: -4 ppm ... 11111: -63 ppm

18.4.8 ERTC闹钟A寄存器(ERTC_ALA)

域	简称	复位值	类型	功能
位 31	MASK4	0x0	rw	日期/星期屏蔽 (Date/week mask) 0: 无屏蔽; 1: 闹钟和日期/星期无关。
位 30	WKSEL	0x0	rw	日期/星期选择 (Date/week mode select) 0: 日期; 1: 星期 (DT[1: 0]不使用)。
位 29: 28	DT	0x0	rw	日期十位 (Date tens)
位 27: 24	DU	0x0	rw	日期/星期个位 (Date/week units)
位 23	MASK3	0x0	rw	小时屏蔽 (Hour mask) 0: 无屏蔽; 1: 闹钟和小时无关。
位 22	AMPM	0x0	rw	上午/下午 (AM/PM) 0: 上午; 1: 下午。 注: 该位只用于 12 小时制, 24 小时制保持为 0。
位 21: 20	HT	0x0	rw	小时十位 (Hour tens)
位 19: 16	HU	0x0	rw	小时个位 (Hour units)
位 15	MASK2	0x0	rw	分钟屏蔽 (Minute mask) 0: 无屏蔽; 1: 闹钟和分钟无关。
位 14: 12	MT	0x0	rw	分钟十位 (Minute tens)
位 11: 8	MU	0x0	rw	分钟个位 (Minute units)
位 7	MASK1	0x0	rw	秒钟屏蔽 (Second mask) 0: 无屏蔽; 1: 闹钟和秒钟无关。
位 6: 4	ST	0x0	rw	秒钟十位 (Second tens)
位 3: 0	SU	0x0	rw	秒钟个位 (Second units)

18.4.9 ERTC闹钟B寄存器(ERTC_ALB)

域	简称	复位值	类型	功能
位 31	MASK4	0x0	rw	日期/星期屏蔽 (Date/week mask) 0: 无屏蔽; 1: 闹钟和日期/星期无关。
位 30	WKSEL	0x0	rw	日期/星期选择 (Date/week mode select) 0: 日期; 1: 星期 (DT[1: 0]不使用)。
位 29: 28	DT	0x0	rw	日期十位 (Date tens)
位 27: 24	DU	0x0	rw	日期/星期个位 (Date/week units)
位 23	MASK3	0x0	rw	小时屏蔽 (Hour mask) 0: 无屏蔽; 1: 闹钟和小时无关。
位 22	AMPM	0x0	rw	上午/下午 (AM/PM) 0: 上午; 1: 下午。 注: 该位只用于 12 小时制, 24 小时制保持为 0。
位 21: 20	HT	0x0	rw	小时十位 (Hour tens)
位 19: 16	HU	0x0	rw	小时个位 (Hour units)
位 15	MASK2	0x0	rw	分钟屏蔽 (Minute mask) 0: 无屏蔽; 1: 闹钟和分钟无关。
位 14: 12	MT	0x0	rw	分钟十位 (Minute tens)
位 11: 8	MU	0x0	rw	分钟个位 (Minute units)
位 7	MASK1	0x0	rw	秒钟屏蔽 (Second mask) 0: 无屏蔽; 1: 闹钟和秒钟无关。
位 6: 4	ST	0x0	rw	秒钟十位 (Second tens)
位 3: 0	SU	0x0	rw	秒钟个位 (Second units)

18.4.10 ERTC写保护寄存器(ERTC_WP)

域	简称	复位值	类型	功能
位 31: 8	保留	0x000000	resd	保持默认值。
位 7: 0	CMD	0x00	wo	命令寄存器 (Command register) 依次写入 0xCA、0x53 解锁所有 ERTC 寄存器写保护，当写一个其他值时，将重新开启写保护。

18.4.11 ERTC亚秒寄存器(ERTC_SBS)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	SBS	0x0000	ro	亚秒值 (Sub-second value) 亚秒为分频器 DIVB 的计数值，时钟频率为 ERTC_CLK/(DIVA+1)。

18.4.12 ERTC时间微调寄存器(ERTC_TADJ)

域	简称	复位值	类型	功能
位 31	ADD1S	0x0	wo	增加一秒 (Add 1 second) 0: 无效果； 1: 增加 1 秒。 当 TADJF=0 时，才能写此寄存器，通常 ADD1S 与 DECSBS 配合使用，达到微调时间的效果。
位 30: 15	保留	0x0000	resd	保持默认值。
位 14: 0	DECSBS	0x0000	wo	DECSBS[14: 0]: 减少亚秒值 (Decrease sub-second value) 延迟时间 (ADD1S=0)： 延迟=DECSBS/(DIVB+1); 提前时间 (ADD1S=1)： 延迟=1-(DECSBS/(DIVB+1))。

18.4.13 ERTC时间戳时间寄存器(ERTC_TSTM)

域	简称	复位值	类型	功能
位 31: 23	保留	0x000	resd	保持默认值。
位 22	AMPM	0x0	ro	上午/下午 (AM/PM) 0: 上午； 1: 下午。 注：该位只用于 12 小时制，24 小时制保持为 0。
位 21: 20	HT	0x0	ro	小时十位 (Hour tens)
位 19: 16	HU	0x0	ro	小时个位 (Hour units)
位 15	保留	0x0	resd	保持默认值
位 14: 12	MT	0x0	ro	分钟十位 (Minute tens)
位 11: 8	MU	0x0	ro	分钟个位 (Minute units)
位 7	保留	0x0	resd	保持默认值。
位 6: 4	ST	0x0	ro	秒钟十位 (Second tens)
位 3: 0	SU	0x0	ro	秒钟个位 (Second units)

注意：仅当 ERTC_STS 中的 TSF 置 1 时，该寄存器的内容才有效。当 TSF 位复位时，清零该寄存器。

18.4.14 ERTC时间戳日期寄存器(ERTC_TS DT)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 13	WK	0x0	ro	星期 (Week)
位 12	MT	0x0	ro	月十位 (Month tens)
位 11: 8	MU	0x0	ro	月个位 (Month units)
位 7: 6	保留	0x0	resd	保持默认值
位 5: 4	DT	0x0	ro	日期十位 (Date tens)
位 3: 0	DU	0x0	ro	日期个位 (Date units)

注意：仅当 ERTC_STS 中的 TSF 置 1 时，该寄存器的内容才有效。当 TSF 位复位时，清零该寄存器。

18.4.15 ERTC时间戳亚秒寄存器(ERTC_TSSBS)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	SBS	0x0000	ro	亚秒值 (Sub-second value)

注意：仅当 ERTC_STS/TSF 置 1 时，该寄存器的内容才有效。当 ERTC_STS/TSF 位复位时，清零该寄存器。

18.4.16 ERTC精密校准寄存器(ERTC_SCAL)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15	ADD	0x0	rw	增加 ERTC 时钟 (Add ERTC clock) 0: 无操作; 1: 每 2^{11} 个 ERTC_CLK, 插入一个 ERTC_CLK。
位 14	CAL8	0x0	rw	8 秒校准周期 (8-second calibration period) 0: 无效果; 1: 8 秒校准周期。
位 13	CAL16	0x0	rw	16 秒校准周期 (16 second calibration period) 0: 无效果; 1: 16 秒校准周期。
位 12: 9	保留	0x0	resd	保持默认值
位 8: 0	DEC	0x000	rw	减少 ERTC 时钟 (Decrease ERTC clock) 在 2^{20} 个 ERTC_CLK 周期内, 屏蔽 DEC 个 ERTC_CLK。 通常和 ADD 配合使用, 当 ADD 为 1 时, 在 2^{20} 个 ERTC_CLK 周期内, 实际的 ERTC_CLK 个数为 $2^{20}+512$ -DEC。

18.4.17 ERTC入侵配置寄存器(ERTC_TAMP)

域	简称	复位值	类型	功能
位 31: 19	保留	0x0000	resd	保持默认值。
位 18	OUTTYPE	0x0	rw	输出类型 (Output type) 0: 开漏; 1: 推挽。
位 17	TSPIN	0x0	rw	时间戳检测引脚选择 (Time stamp detection pin selection) 0: ERTC_MUX1; 1: ERTC_MUX2。
位 16	TP1PIN	0x0	rw	入侵检测 1 引脚选择 (Tamper detection pin selection) 0: ERTC_MUX1; 1: ERTC_MUX2。
位 15	TPPU	0x0	rw	入侵检测上拉 (Tamper detection pull-up) 0: 开启; 1: 关闭。
位 14: 13	TPPR	0x0	rw	入侵检测预充电时间 (Tamper detection pre-charge time)

				0: 1个 ERTC_CLK; 1: 2个 ERTC_CLK; 2: 4个 ERTC_CLK; 3: 8个 ERTC_CLK。
位 12: 11	TPFLT	0x0	rw	入侵检测滤波时间 (Tamper detection filter time) 0: 无滤波; 1: 连续 2 次采样有效, 判定入侵事件发生; 2: 连续 4 次采样有效, 判定入侵事件发生; 3: 连续 8 次采样有效, 判定入侵事件发生。
位 10: 8	TPFREQ	0x0	rw	入侵检测检测频率 (Tamper detection frequency) 0: ERTC_CLK/32768; 1: ERTC_CLK/16384; 2: ERTC_CLK/8192; 3: ERTC_CLK/4096; 4: ERTC_CLK/2048; 5: ERTC_CLK/1024; 6: ERTC_CLK/512; 7: ERTC_CLK/256。
位 7	TPTSEN	0x0	rw	入侵检测时间戳使能 (Tamper detection timestamp enable) 0: 关闭; 1: 开启, 当产生入侵事件时, 保持时间戳。
位 6: 5	保留	0x0	resd	保持默认值。
位 4	TP2EDG	0x0	rw	入侵检测 2 有效边沿 (Tamper detection 2 valid edge) 当无滤波时 (TPFLT=0) : 0: 上升沿; 1: 下降沿。 当有滤波时 (TPFLT>0) : 0: 低电平; 1: 高电平。
位 3	TP2EN	0x0	rw	入侵检测 2 使能 (Tamper detection 2 enable) 0: 关闭; 1: 开启。
位 2	TPIEN	0x0	rw	入侵检测中断使能 (Tamper detection interrupt enable) 0: 关闭; 1: 开启。
位 1	TP1EDG	0x0	rw	入侵检测 1 有效边沿 (Tamper detection 1 valid edge) 当无滤波时 (TPFLT=0) : 0: 上升沿; 1: 下降沿。 当有滤波时 (TPFLT>0) : 0: 低电平; 1: 高电平。
位 0	TP1EN	0x0	rw	入侵检测 1 使能 (Tamper detection 1 enable) 0: 关闭; 1: 开启。

18.4.18 ERTC闹钟A亚秒寄存器(ERTC_ALASBS)

域	简称	复位值	类型	功能
位 31: 28 保留		0x0	resd	保持默认值。
位 27: 24 SBSMSK		0x0	rw	亚秒屏蔽 (Sub-second mask) 0: 不匹配亚秒, 闹钟与亚秒无关; 1: 只匹配 SBS[0]; 2: 只匹配 SBS[1: 0]; 3: 只匹配 SBS[2: 0]; ... 14: 只匹配 SBS[13: 0]; 15: 匹配 SBS[14: 0]。
位 23: 15 保留		0x000	rw	保持默认值。
位 14: 0 SBS		0x0000	rw	亚秒值 (Sub-second value)

18.4.19 ERTC闹钟B亚秒寄存器(ERTC_ALBSBS)

域	简称	复位值	类型	功能
位 31: 28 保留		0x0	resd	保持默认值。
位 27: 24 SBSMSK		0x0	rw	亚秒屏蔽 (Sub-second mask) 0: 不匹配亚秒, 闹钟与亚秒无关; 1: 只匹配 SBS[0]; 2: 只匹配 SBS[1: 0]; 3: 只匹配 SBS[2: 0]; ... 14: 只匹配 SBS[13: 0]; 15: 匹配 SBS[14: 0]。
位 23: 15 保留		0x000	rw	保持默认值。
位 14: 0 SBS		0x0000	rw	亚秒值 (Sub-second value)

18.4.20 ERTC电池供电数据寄存器(ERTC_BPRx)

域	简称	复位值	类型	功能
位 31: 0 DT		0x0000 0000	rw	电池供电域数据 (Battery powered domain data) BPR_DTx 寄存器, 可以在只由电池供电下保存数据, 不会被系统复位所复位, 只能通过电池供电域复位或入侵事件进行复位。

19 模拟/数字转换 (ADC)

19.1 ADC简介

ADC 是一个将模拟输入信号转换为 12 位、10 位、8 位或 6 位的数字信号的外设。采样率最高可达 5.33MSPS。每组 ADC 多达 19 个通道源（包括内部及外部通道）可进行采样及转换，两组 ADC 总计 16 个外部输入通道源。

19.2 ADC主要特征

模拟方面有以下特征：

- 支持分辨率 12 位、10 位、8 位或 6 位的转换
- 自校准时间：205 个 ADC 时钟周期
- ADC 转换时间
 - 快速通道: ADC 时钟 80MHz, 分辨率 12 位时转换时间为 0.1875 μ s(5.33MSps)
 - 慢速通道: ADC 时钟 80MHz, 分辨率 12 位时转换时间为 0.2375 μ s(4.21MSps)
 - 快速通道: ADC 时钟 80MHz, 分辨率 6 位时转换时间为 0.1126 μ s(8.88MSps)
 - 慢速通道: ADC 时钟 80MHz, 分辨率 6 位时转换时间为 0.1626 μ s(6.15MSps)
- ADC 供电要求：请参考 Datasheet
- ADC 输入范围： $V_{REF^-} \leq V_{IN} \leq V_{REF^+}$

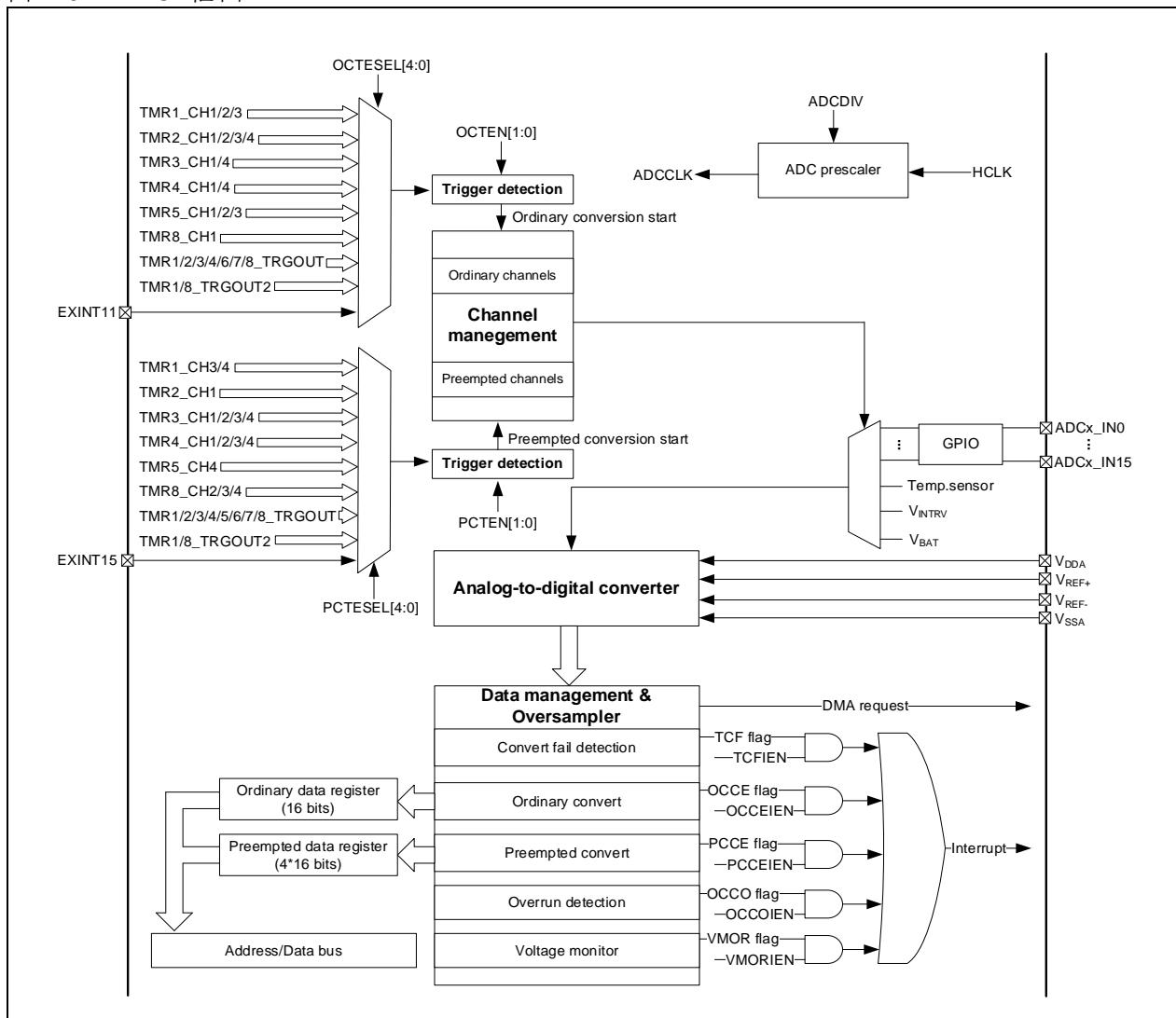
数字控制方面有以下特征：

- 通道管理区分优先权不同的普通通道与抢占通道
- 普通通道与抢占通道具备各自独立的触发侦测电路
- 各通道均可独立配置采样时间
- 转换顺序管理支持多种不同的多通道转换
- 过采样器：硬件过采样最高可实现 16 位分辨率
- 可选择的数据对齐方式
- 可配置的电压监测边界
- 支持 DMA 传输的普通通道数据
- 可设定以下事件发生时响应中断
 - 普通通道转换数据溢出
 - 抢占通道组转换结束
 - 普通通道转换结束
 - 电压监测超出范围
 - 触发转换失败
- 联动多 ADC 的主从模式
- 可配置多 ADC 之间时序位移长度的主从位移模式
 - 高效能需求可使用主从模式搭配 DMA

19.3 ADC架构

ADC1 的架构如下图所示。ADC2 与 ADC1 不同之处在于：ADC2 没有连接内部温度传感器（Temp. sensor）、内部参考电压（V_{INTRV}）与电池电压（V_{BAT}）。

图 19-1 ADC1框图



输入引脚介绍：

- V_{DDA}：模拟电源，ADC 模拟电源
- V_{SSA}：模拟电源地，ADC 模拟电源地
- V_{REF+}：模拟参考正极，ADC 使用的高端/正极模拟参考电压
- V_{REF-}：模拟参考负极，ADC 使用的低端/负极参考电压
- ADCx_IN：模拟输入信号通道

输入管脚的连接与电压范围限制请参考 Datasheet

19.4 ADC功能介绍

19.4.1 通道管理

模拟信号通道输入

每个ADC拥有多达19个模拟信号通道输入，以ADC_INx表示，x=0至18。

- ADC1_IN0至ADC1_IN15为外部模拟输入，ADC1_IN16为内部温度传感器，ADC1_IN17为内部参考电压，ADC1_IN18为电池电压。
- ADC2_IN0至ADC2_IN15为外部模拟输入，ADC2_IN16至ADC2_IN18为V_{SSA}。

通道转换

转换区分为普通通道转换与抢占通道转换，抢占通道的转换优先权高于普通通道。

抢占通道触发若发生于普通通道转换途中，优先进行抢占通道的转换，普通通道于抢占通道转换结束后重新开始转换被打断的通道。普通通道触发若发生于抢占通道转换途中，普通通道的转换会等待抢占通道转换完成后才开始。

将通道(ADC_INx)编排进普通通道序列(ADC_OSQx)以及抢占通道序列(ADC_PSQ)，相同通道可重复编排，序列总数由OCLEN与PCLEN定义，接着即可启动普通通道转换或抢占通道转换。

19.4.1.1 内部温度传感器

温度传感器接到ADC1_IN16，必须先使能ADC_CCTRL的ITSRVEN位并且等待上电时间后才可对温度传感通道进行转换。

温度传感器输出电压与温度呈线性变化，电压和温度曲线的平均斜率值参考数据手册。由于生产制造的偏差，温度曲线在不同芯片间存在差异，因此温度传感器更适合监测温度变化而非绝对温度。使用ADC转换ADC_IN16，计算当前温度传感器输出电压，结合温度计算公式，可计算出当前温度。

温度计算公式如下：

$$\text{温度}(\text{°C}) = \{(V_{25} - V_{\text{TS}})/\text{Avg_Slope}\} + 25$$

V₂₅: 温度传感器25°C时输出的电压值，参数值详见数据手册。

V_{TS}: 温度传感器当前输出的电压值。

Avg_Slope: 温度传感器温度曲线平均斜率，参数值详见数据手册。

19.4.1.2 内部参考电压

典型值1.2V的内部参考电压接到ADC1_IN17，必须先使能ADC_CCTRL的ITSRVEN位后才可对内部参考电压通道进行转换。此通道的转换数据可用于推算外部参考电压。

19.4.1.3 电池电压

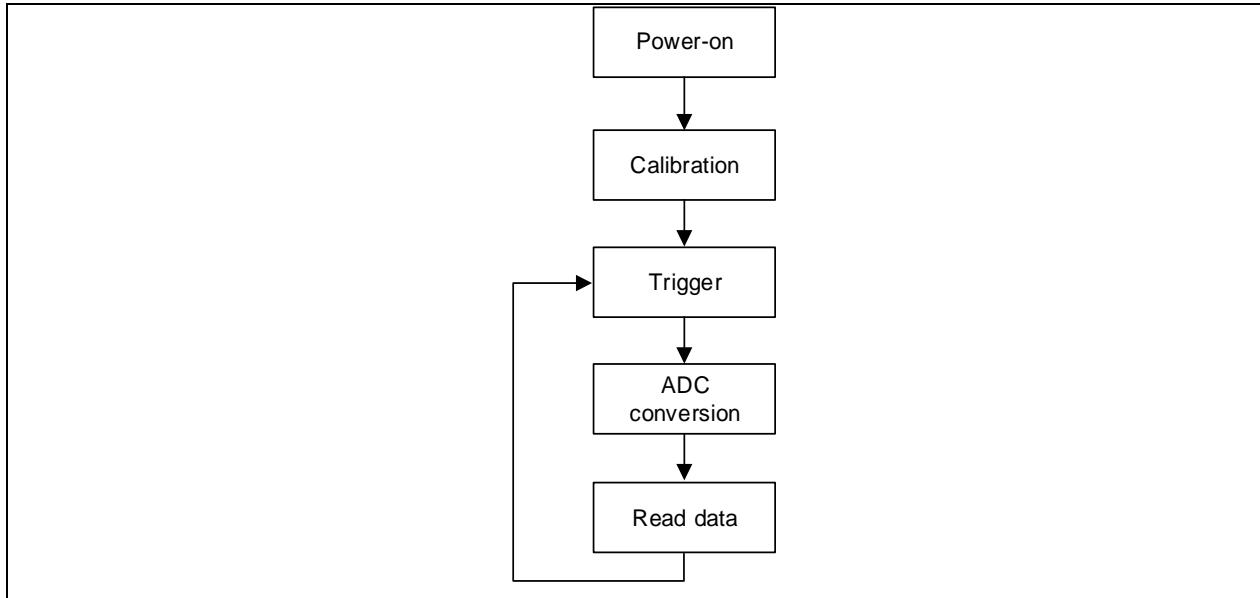
由于VBAT电压可能高于VDDA，因此VBAT经过除4电路才接到ADC1_IN18。必须先使能ADC_CCTRL的VBATEN位使除4电路上电后，才可对电池电压通道进行转换。

注意：使用ADC采集VBAT电压，当使能VBATEN后，VBAT会一直耗电，如果应用对功耗敏感，建议只在VBAT采样时才打开，采样结束后关闭。

19.4.2 ADC操作流程

ADC的基础操作流程如下图所示，建议第一次上电后进行校准，以提升采样与转换准确度。待校准完成后可靠触发引起ADC采样转换，转换结束后即可读取数据。

图 19-2 ADC基础操作流程



19.4.2.1 上电与校准

上电

用户须先使能 CRM_APB2EN 的 ADCxEN，以使能 ADC 的时钟：PCLK2 与 ADCCLK。

时钟使能后必须配置 ADC 预分频器 (ADC_CCTRL 的 ADCDIV)，将 ADCCLK 调整至需求的频率。ADCCLK 由 HCLK 除频而来。

注意：ADCCLK 不可大于 80MHz，ADCCLK 频率不可高于 PCLK2。

ADCCLK 频率调整完后，即可使能 ADC_CTRL2 的 ADCEN 位使 ADC 上电，等待 RDY 标志置位后才可对 ADC 进行后续操作。清除 ADCEN 会使 ADC 的转换中止并复位，同时 ADC 被断电以达到省电的效果。

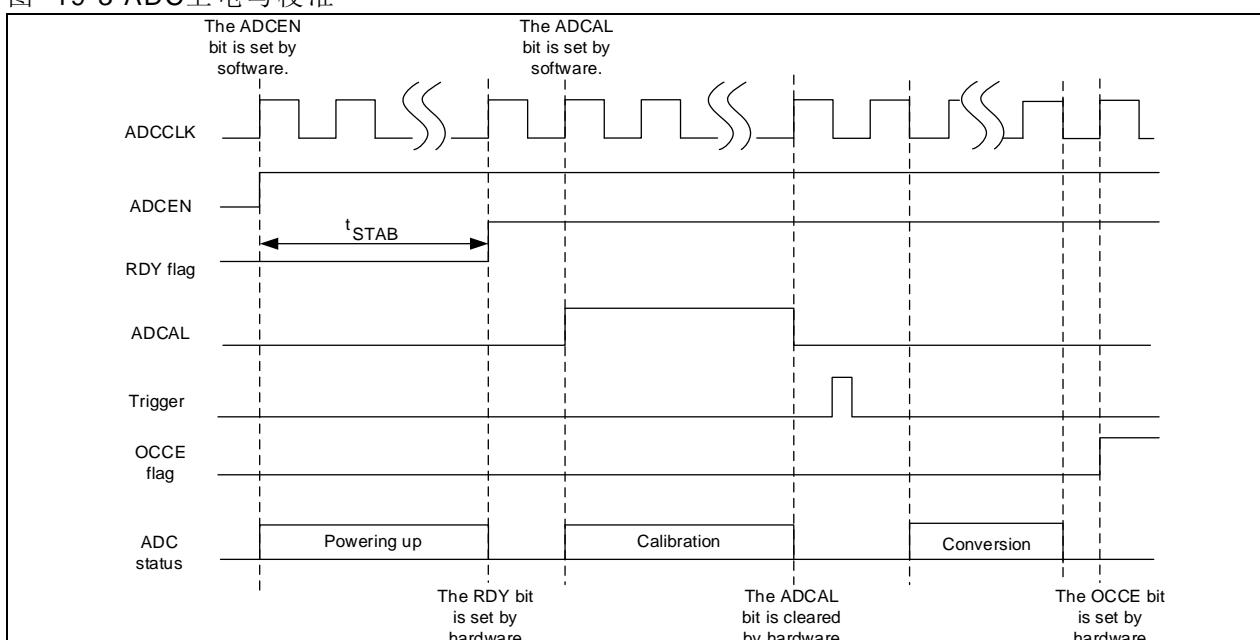
校准

上电完成后可设置 ADC_CTRL2 的 ADCAL 使 ADC 进行校准，校准完成后硬件清除 ADCAL 位，软件即可触发以进行转换。

每次校准后，校准值会被存放至 ADC_ODT 中，这个校准值自动反馈回 ADC 内部，以消除电容误差。该校准值的存放不会置位 OCCE 标志，不会产生中断或 DMA 请求。

注意：只能在分辨率 12 位时进行校准，校准完成后才可切换分辨率，切换分辨率后等待 RDY 标志置位即可触发。

图 19-3 ADC上电与校准



19.4.2.2 触发

ADC 触发分为普通通道触发与抢占通道触发，普通通道触发引发普通通道转换，抢占通道触发引发抢占通道转换。针对外部触发来源的有效极性是由 ADC_CTRL2 的 OCETE 与 PCETE 选择，ADC 会检测触发来源并响应转换。

触发来源可分为软件写寄存器触发（ADC_CTRL2 的 OCSWTRG 与 PCSWTRG）以及外部触发，外部触发包含定时器触发与引脚触发，由 ADC_CTRL2 的 OCTESEL 与 PCTESEL 选择触发来源，如下表所示。

表 19-1 普通通道触发来源

OCTESEL	触发来源	OCTESEL	触发来源
00000	TMR1_CH1 event	10000	保留
00001	TMR1_CH2 event	10001	保留
00010	TMR1_CH3 event	10010	保留
00011	TMR2_CH2 event	10011	保留
00100	TMR2_CH3 event	10100	保留
00101	TMR2_CH4 event	10101	TMR8_TRGOUT2 event
00110	TMR2_TRGOUT event	10110	TMR1_TRGOUT2 event
00111	TMR3_CH1 event	10111	TMR4_TRGOUT event
01000	TMR3_TRGOUT event	11000	TMR6_TRGOUT event
01001	TMR4_CH4 event	11001	TMR3_CH4 event
01010	TMR5_CH1 event	11010	TMR4_CH1 event
01011	TMR5_CH2 event	11011	TMR1_TRGOUT event
01100	TMR5_CH3 event	11100	TMR2_CH1 event
01101	TMR8_CH1 event	11101	保留
01110	TMR8_TRGOUT event	11110	TMR7_TRGOUT event
01111	EXINT line11 External pin	11111	保留

表 19-2 抢占通道触发来源

PCTESEL	触发来源	PCTESEL	触发来源
00000	TMR1_CH4 event	10000	保留
00001	TMR1_TRGOUT event	10001	保留
00010	TMR2_CH1 event	10010	保留
00011	TMR2_TRGOUT event	10011	TMR1_TRGOUT2 event
00100	TMR3_CH2 event	10100	TMR8_TRGOUT event
00101	TMR3_CH4 event	10101	TMR8_TRGOUT2 event
00110	TMR4_CH1 event	10110	TMR3_CH3 event
00111	TMR4_CH2 event	10111	TMR3_TRGOUT event
01000	TMR4_CH3 event	11000	TMR3_CH1 event
01001	TMR4_TRGOUT event	11001	TMR6_TRGOUT event
01010	TMR5_CH4 event	11010	TMR4_CH4 event
01011	TMR5_TRGOUT event	11011	TMR1_CH3 event
01100	TMR8_CH2 event	11100	保留
01101	TMR8_CH3 event	11101	保留
01110	TMR8_CH4 event	11110	TMR7_TRGOUT event
01111	EXINT line15 External pin	11111	保留

19.4.2.3 采样与转换时序

用户可于 ADC_SPT1 与 ADC_SPT2 的 CSPTx 配置各个通道(ADC_INx)的采样周期。通道 0/1/8/9/12/13 为快速通道，支持最小 2.5 周期采样时间；其他通道为慢速通道，支持最小 6.5 周期采样时间。藉由配置 ADC_CTRL1 的 CRSEL 可改变转换数据的分辨率，较低的分辨率所需的转换时间较短。一次转换所需的时间可利用以下公式推得：

$$\text{一次转换所需的时间(ADCCLK 的周期)} = \text{采样时间} + \text{分辨率位数} + 0.5$$

示例：

CSPTx 选择 2.5 周期，CRSEL 选择 12 位，一次转换需要 $2.5+12.5=15$ 个 ADCCLK 周期。

CSPTx 选择 6.5 周期，CRSEL 选择 10 位，一次转换需要 $6.5+10.5=17$ 个 ADCCLK 周期。

19.4.3 转换顺序管理

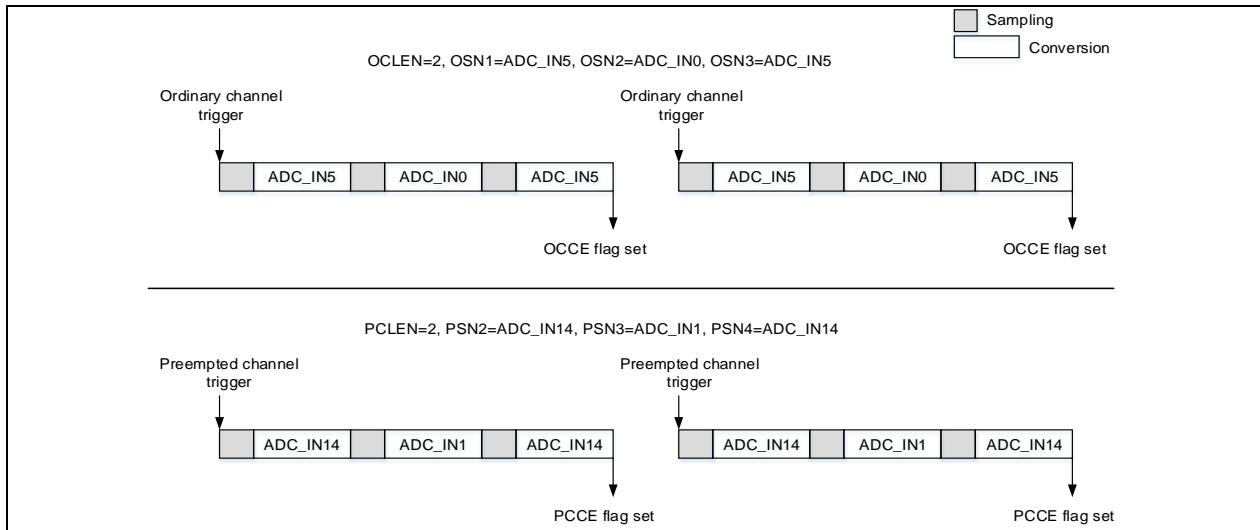
默认模式下，每次触发只会转换单个通道，即 OSN1（普通触发）或 PSN4（抢占触发）记录的通道。

下面介绍不同的转换顺序模式，即可使多个通道以特定顺序做转换。

19.4.3.1 序列模式

使能 ADC_CTRL1 的 SQEN，即开启序列模式，用户于 ADC_OSQx 配置普通通道顺序与总数，于 ADC_PSQ 配置抢占通道顺序与总数，开启序列模式后，一次触发将序列中的通道依序转换一次。普通通道从 OSN1 开始转换起，抢占通道是从 PSNx 开始转换起， $x=4-PCLEN$ ，下图示范了序列模式的行为。

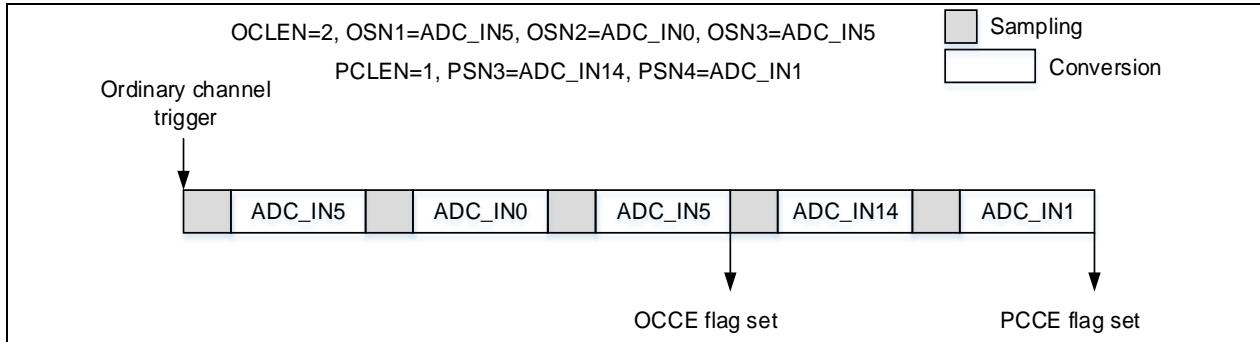
图 19-4 序列模式



19.4.3.2 抢占自动转换模式

使能 ADC_CTRL1 的 PCAUTOEN，即开启抢占自动转换模式，当普通通道转换完成后，抢占通道将自动接续着转换。可与序列模式共用，当普通通道序列完成后，即会自动开始抢占序列的转换。下图示范了与序列模式共用的抢占自动转换模式行为。

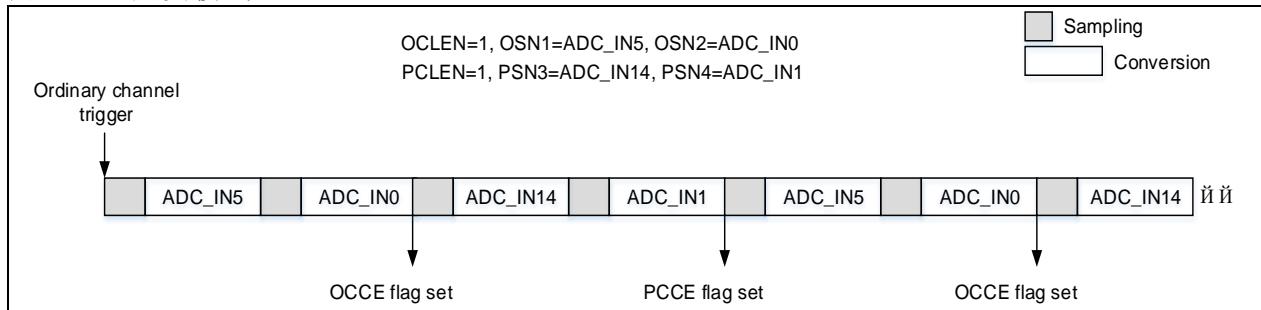
图 19-5 抢占自动转换模式



19.4.3.3 反复模式

使能 ADC_CTRL2 的 RPEN，即开启反复模式。当普通通道检测到触发后就会反复不断地转换。可与序列模式下的普通通道转换共用，将反复地转换普通通道序列。也可与抢占自动转换模式共用，将依次反复地转换普通通道序列与抢占通道序列。下图示范了与序列模式及抢占自动转换模式共用的反复模式行为。

图 19-6 反复模式

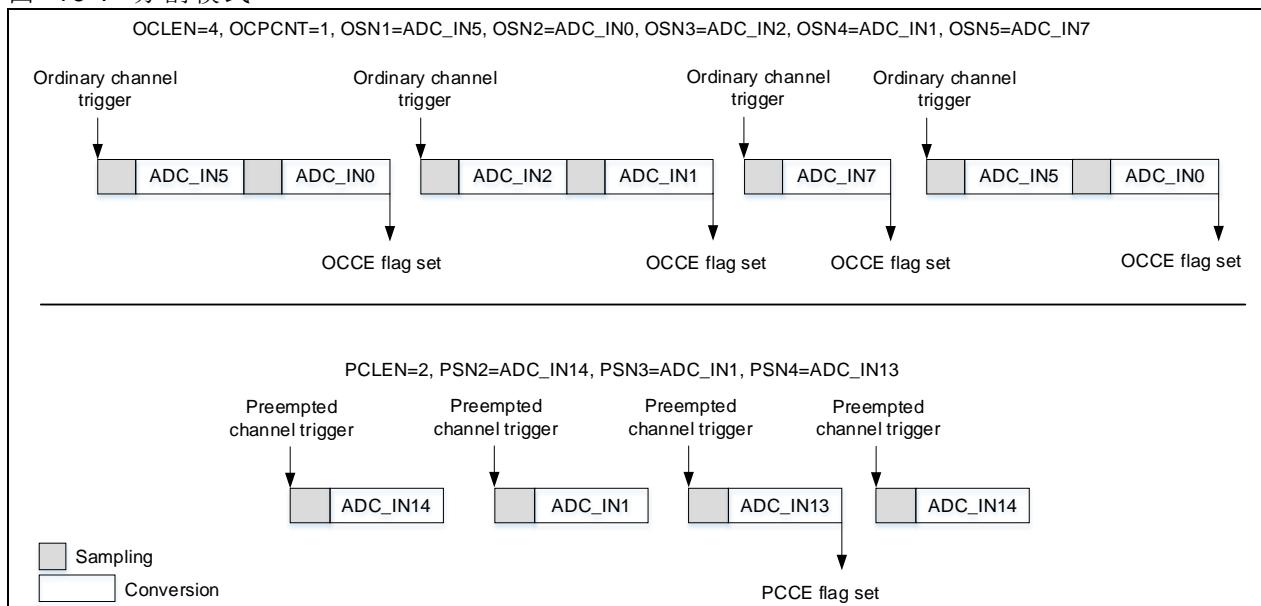


19.4.3.4 分割模式

使能 ADC_CTRL1 的 OCPEN，即开启普通通道的分割模式，此模式将 ADC_OSQ1 的 OCLEN 的序列长度分割成长度较小的子组别，子组别的通道数于 ADC_CTRL1 的 OCPCNT 配置，一次触发将转换子组别中的所有通道。每次触发会依序选择不同的子组别。

使能 ADC_CTRL1 的 PCPEN，即开启抢占通道的分割模式，此模式将 ADC_PSQ 的 PCLEN 的序列长度分割成只有一个通道的子组别，一次触发将转换子组别中的通道。每次触发会依序选择不同的子组别。分割模式与反复模式不可共用。下图分别示范了普通分割与抢占分割模式的行为。

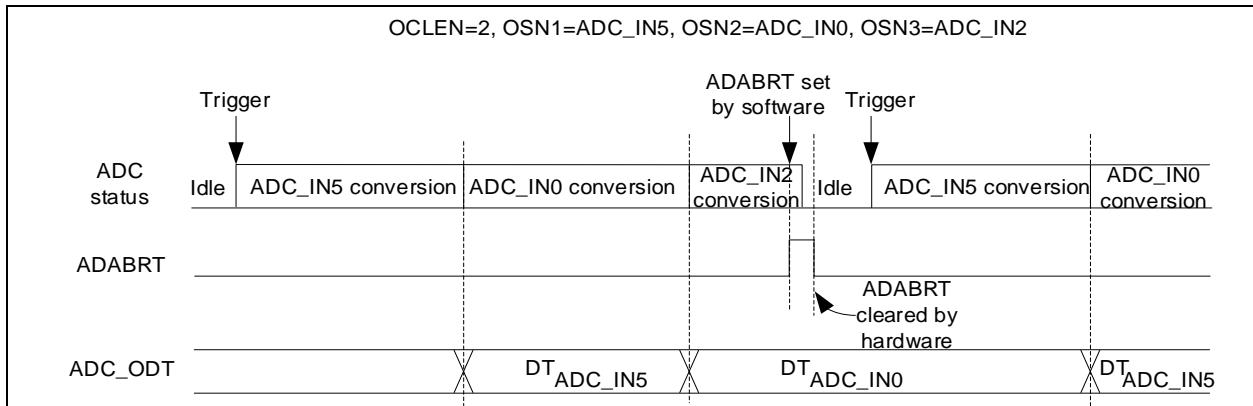
图 19-7 分割模式



19.4.4 转换中止

用户可利用 ADC_CTRL2 的 ADABRT 使 ADC 停止转换，转换顺序回归第一个通道，用户即可重新编排通道顺序，再次触发 ADC 将从头开始新序列的转换。若 ADABRT 已置位，需等待 ADABRT 被硬件清零后进行其他 ADC 操作。图 19-8 置位 ADABRT 时的时序图。

图 19-8 ADABRT 时序



19.4.5 触发转换失败

ADC 转换发生异常时，TCF 标志置起，用于指示此次 ADC 转换失败，普通或抢占数据寄存器内的数据无效。默认情况下，ADC 不会因转换失败停止转换。

若要在转换失败时停止 ADC 转换，可在触发转换失败中断(通过 TCFIEN 位使能)中重新配置转换序列或通道。另外，同时开启触发转换失败中断和触发转换失败后自动终止转换(通过 TCFACA 位使能)功能后，发生转换失败时，ADC 会自动停止当前进行转换的序列或通道，并将 ADC 复位为空闲状态，之后的有效触发会重新触发 ADC 重新从第一个通道开始转换。

19.4.6 过采样器

一次过采样转换数据是透过转换多次相同通道，累加转换数据后作平均实现的。

- 由 ADC_OVSP 的 OSRSEL 选择过采样率，此位用来定义过采样倍数，通过多次转换同个通道实现。
- 由 ADC_OVSP 的 OSSSEL 选择过采样移位，此位用来定义平均系数，通过右移位实现。

若平均后数据大于 16 位，只取靠右 16 位数据，放入 16 位数据寄存器。如表 19-3 所示。

示例：

若 OSRSEL 选择 4 倍，一次过采样转换同个通道转换 4 次，并将这 4 次转换数据累加。若 OSSSEL 选择 6 位，累加数据除以 2^6 ，以四舍五入进位。

表 19-3 最大累加数据与过采样倍数及位移系数关系

过采样率	2x	4x	8x	16x	32x	64x	128x	256x
最大累加数据	0x1FFE	0x3FFC	0x7FF8	0xFFFF0	0x1FFE0	0x3FFC0	0x7FF80	0xFFFF00
不移位	0x1FFE	0x3FFC	0x7FF8	0xFFFF0	0xFFE0	0xFFC0	0xFF80	0xFF00
移 1 位	0x0FFF	0x1FFE	0x3FFC	0x7FF8	0xFFFF0	0xFFE0	0xFFC0	0xFF80
移 2 位	0x0800	0x0FFF	0x1FFE	0x3FFC	0x7FF8	0xFFE0	0xFFC0	0xFF80
移 3 位	0x0400	0x0800	0x0FFF	0x1FFE	0x3FFC	0x7FF8	0xFFE0	0xFFC0
移 4 位	0x0200	0x0400	0x0800	0x0FFF	0x1FFE	0x3FFC	0x7FF8	0xFFE0
移 5 位	0x0100	0x0200	0x0400	0x0800	0x0FFF	0x1FFE	0x3FFC	0x7FF8
移 6 位	0x0080	0x0100	0x0200	0x0400	0x0800	0x0FFF	0x1FFE	0x3FFC
移 7 位	0x0040	0x0080	0x0100	0x0200	0x0400	0x0800	0x0FFF	0x1FFE
移 8 位	0x020	0x0040	0x0080	0x0100	0x0200	0x0400	0x0800	0x0FFF

使用过采样时，忽视 DTALIGN 与 PCDTOx，数据一律靠右摆放。切换 CRSEL 位改变过采样转换中每一次通道转换结果的分辨率，但是不改变过采样数据运算方式。

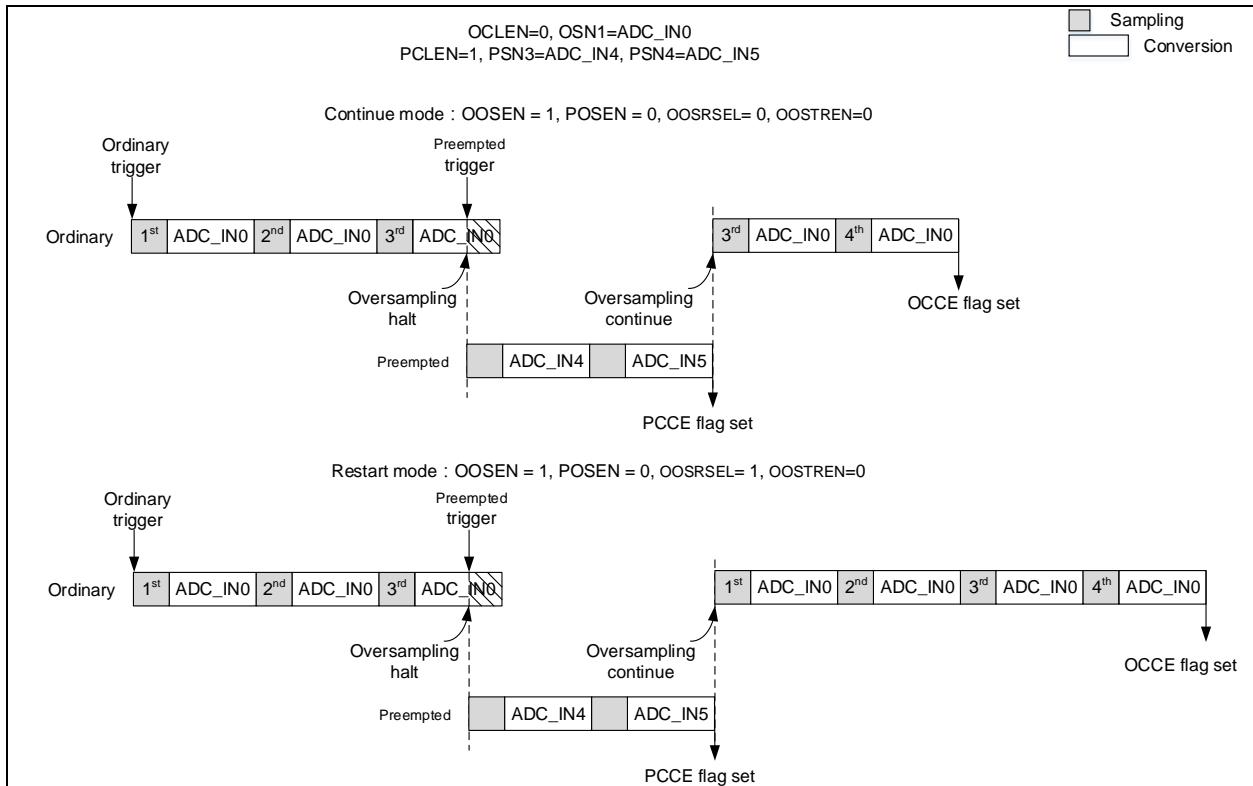
19.4.6.1 普通通道过采样

由 ADC_OVSP 的 OOSRSEL 控制普通过采样被打断后恢复的动作。

- OOSRSEL=0：接续模式，普通过采样中途被抢占通道转换插入后，保留已累加的数据，再次开始转换时将从打断处转换。
- OOSRSEL=1：重转模式。普通过采样中途被抢占通道转换插入后，累加的数据被清空。再次开始转换时是重新该通道的过采样转换。

下图以 4x 过采样率与序列模式下，示范了普通过采样接续模式与重转模式的差别。

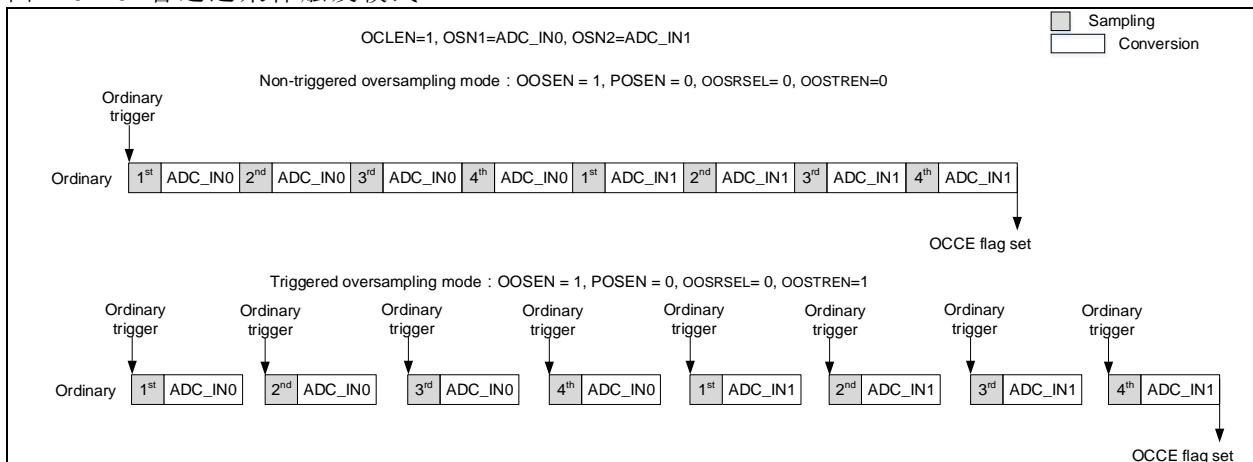
图 19-9 普通过采样重转模式选择



使能 ADC_OVSP 的 OOSTREN 即可使用触发模式。用户须触发普通过采样转换中每一次的普通通道转换。此模式下，中途被抢占通道触发打断后，须重新触发普通通道才会恢复转换普通通道过采样。

当触发模式与转换顺序管理模式搭配使用时，触发方式遵循触发模式，转换完成标志则遵循转换顺序管理模式。下图以 4x 过采样率与序列模式下，示范了普通过采样触发模式与恢复模式共用。
注意：触发模式与反复模式不可共用。

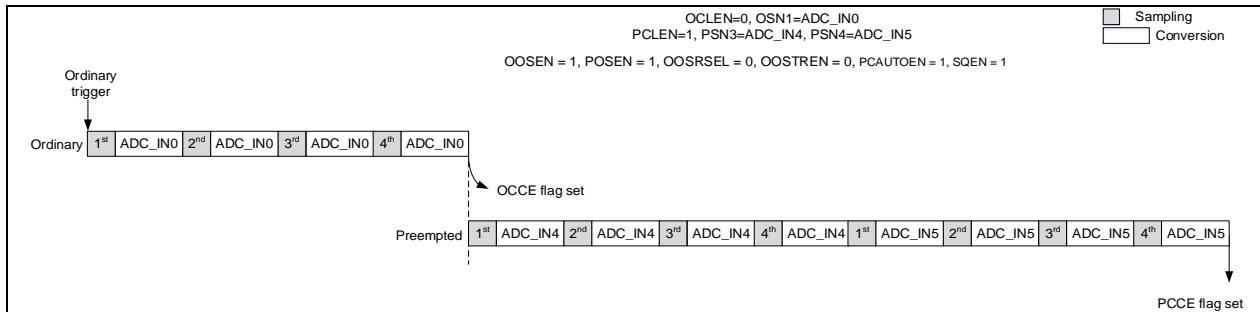
图 19-10 普通过采样触发模式



19.4.6.2 抢占通道过采样

抢占过采样可与普通过采样同时使用，也可分别使用。抢占过采样不影响到普通过采样的各种模式。下图以 4x 过采样率与抢占自动转换模式下，示范了抢占过采样与普通过采样触发模式共用。

图 19-11 抢占过采样



19.4.7 数据管理

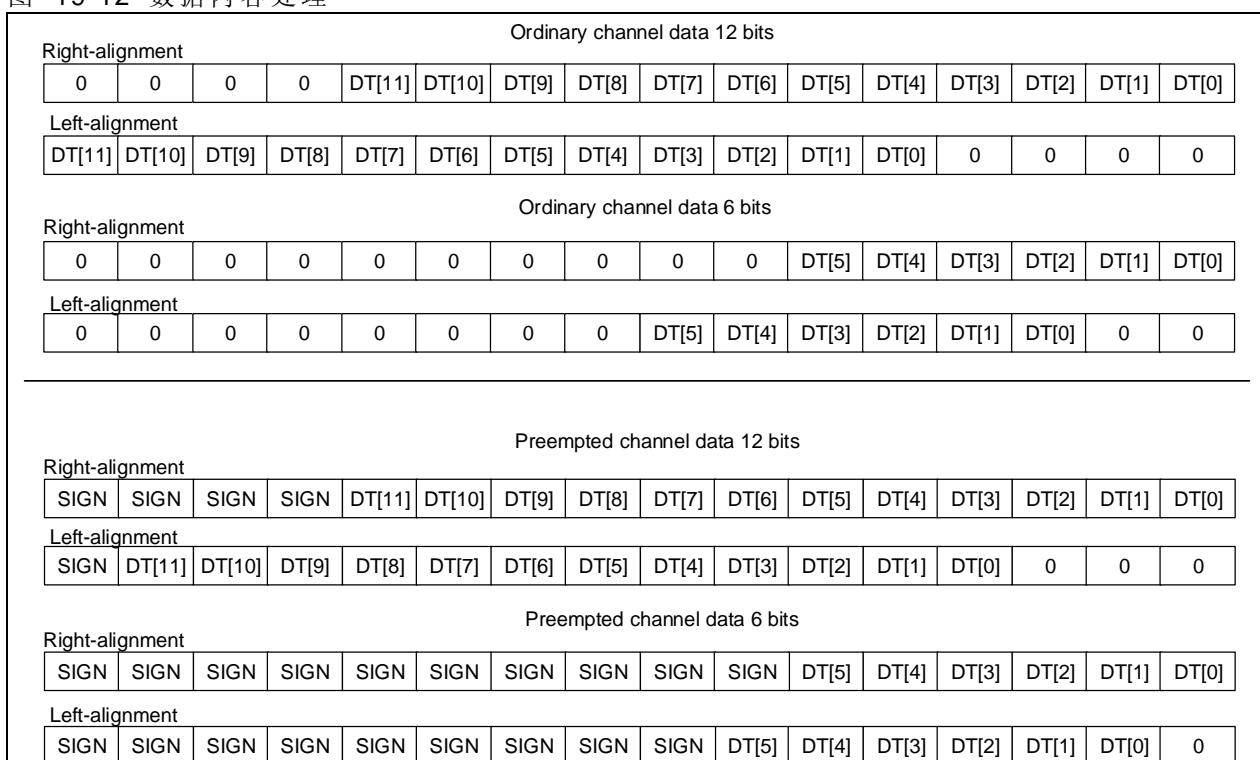
普通通道转换完成后数据存储于普通数据寄存器 (ADC_ODT)，抢占通道转换完成后数据存储于抢占数据寄存器 x (ADC_PDTx)。

19.4.7.1 数据内容处理

由 ADC_CTRL2 的 DTALIGN 选择转换数据靠右或是靠左对齐放置于数据寄存器，除此之外，抢占通道的数据还会减去抢占数据偏移寄存器 x (ADC_PCDTOx) 的偏移量，因此抢占通道数据有可能为负值，以 SIGN 作为符号。

分辨率 CRSEL 为 6 位时，数据存储方式以字节为基准摆放，其余皆以半字为基准摆放。如下图所示。

图 19-12 数据内容处理



19.4.7.2 数据获取

普通通道转换数据可藉由 CPU 或 DMA 读取普通数据寄存器 (ADC_ODT) 获得。抢占通道数据只可藉由 CPU 读取抢占数据寄存器 x (ADC_PDTx) 获得。

由 ADC_CTRL2 的 EOCSFEN 选择普通通道转换结束标志置位于序列结束或是每次普通数据寄存器更新时。

使能 ADC_CTRL2 的 OCDMAEN 后，ADC 会在每次普通数据寄存器更新时请求 DMA。

当 EOCSFEN 或是 OCDMAEN 置位时，ADC 会自动开启溢出检测。若发生溢出事件，OCCO 溢出事件标志置起，ADC 转换停下，数据寄存器存储着最后一个有效数据。若使用 DMA，DMA 请求持续置起以请求 DMA 读取最后的有效数据。软件需将 OCCO 清除，并且重新触发 ADC，ADC 将从有效数据的下一个通道开始转换，如此即使中途发生溢出事件，所有被读取的数据皆是有效且按照顺序的。

配置 ADC_CTRL2 的 OCDRCEN 来选择是否要在 DMA 传输数量寄存器归零后持续请求 DMA。

19.4.8 电压监测

使能 ADC_CTRL1 的 OCVMEN (普通通道) 或 PCVMEN (抢占通道) 即可通过对转换结果的判定来实现电压监测。

当转换结果大于高边界 ADC_VMH[11:0]寄存器或是小于低边界 ADC_VMLB[11:0]寄存器时，电压监测超出标志 VMOR 会置起。

若选择 10 位分辨率，ADC_VMH[11:10]和 ADC_VMLB[11:10]要设置成 2'b00。

若选择 8 位分辨率，ADC_VMH[11:8]和 ADC_VMLB[11:8]要设置成 4'b0000。

若选择 6 位分辨率，ADC_VMH[11:6]和 ADC_VMLB[11:6]要设置成 6'b000000。

透过 VMSGGEN 选择对单一特定通道或是所有通道监测。对单一通道监测的话，由 VMCSEL 配置通道。

电压监测一律以转换的原始数据与 12 位边界寄存器做比较，无视 CRSEL、PCDTox 与 DTALIGN 位的设定。

若是使用过采样器，则是以 ADC_VMH[15:0]与 ADC_VMLB[15:0]完整的 16 位寄存器与过采样数据作比较。

19.4.8.1 状态标志与中断

每个 ADC 拥有自己的状态寄存器 (ADCx_STS): 触发转换失败标志 (TCF)、准备就绪标志 (RDY)、溢出事件标志 (OCCO)、普通通道转换开始标志 (OCCS)、抢占通道转换开始标志 (PCCS)、抢占通道组转换结束标志 (PCCE)、普通通道转换结束标志 (OCCE) 及电压监测超出标志 (VMOR)。

两个 ADCx_STS 会映射到通用状态寄存器 (ADC_CSTS)，因此只需要读取通用状态寄存器即可一次获得所有 ADC 的状态。

其中溢出事件标志、抢占通道组转换结束标志、普通通道转换结束标志及电压监测超出标志拥有对应中断使能位，只要将中断使能，标志置起时便会对 CPU 发出中断。ADC1、ADC2 共用一个中断向量。

19.5 主从模式

开启主从模式即可通过触发主机来联动从机进行通道转换，并且将通用普通数据寄存器作为获取主从 ADC 普通通道数据的单一接口。

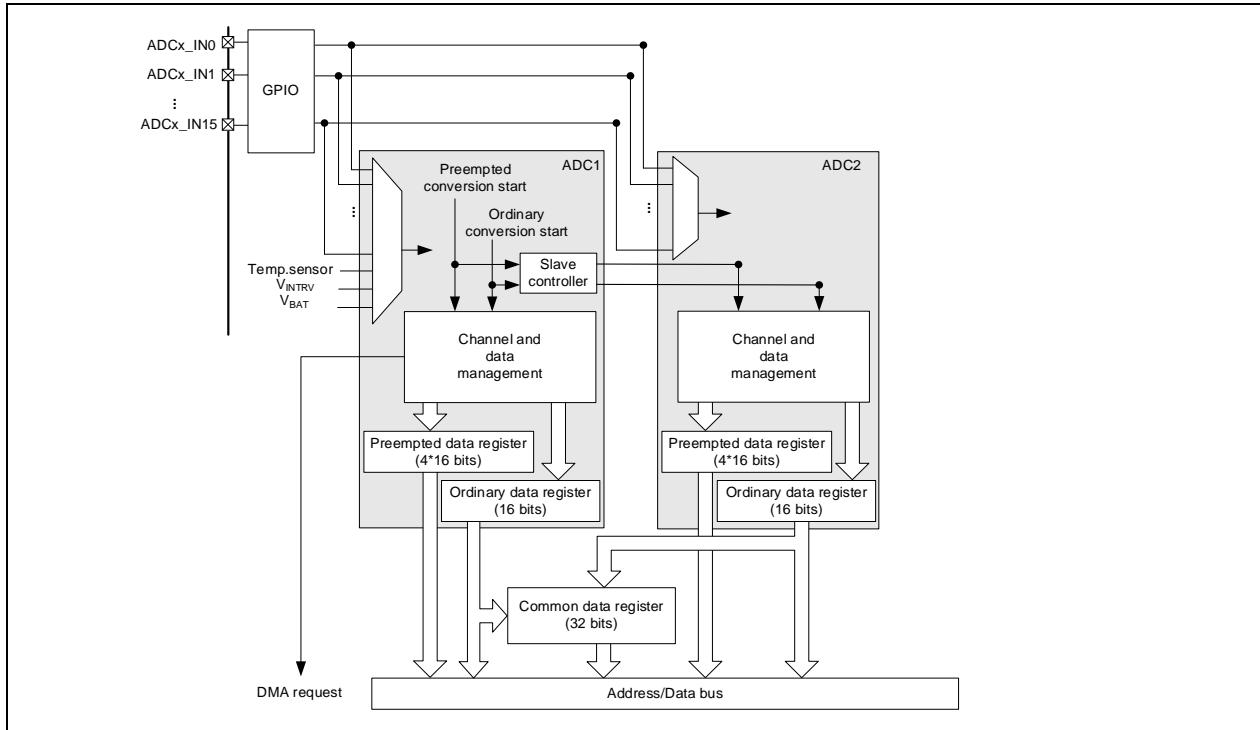
单从机主从模式以 ADC1 作为主机，ADC2 作为从机。主从模式下，需要同时使能主及从 ADC 的触发模式。

注意：转换中止 (ADABRT) 不可使用于主从模式，若在主从模式下想停止 ADC 的转换需清除每个 ADC 的 ADCEN。

注意：为避免主从间失去同步，主从机必须配置相同分辨率。

注意：若有同时使用多个 ADC 低解析度转换的需求，建议使用主从模式搭配 DMA1 或 DMA2。

图 19-13 主从模式的ADC框图



19.5.1 数据管理

主从模式时，普通通道数据会共同存储于通用普通数据寄存器 `ADC_CODT` 中，存储方式透过 `ADC_CCTRL` 的 `MSDMASEL` 位配置，提供 3 种模式，如表 19-4 所示。只要 `MSDMASEL` 不为 0，就会在每次数据备齐时使用 `ADC1` 的 DMA 通道请求 DMA，主机与从机的溢出检测也会启动，发生溢出事件的 ADC 停下转换，同时 DMA 请求也停下，主机与从机可能因此失去同步，建议发生溢出事件后重新初始化 ADC 再触发。

表 19-4 主从模式 DMA 模式

<code>MSDMASEL</code>	主从模式	DMA 请求	<code>ADC_CODT[31:0]</code>
001	单从机	第一次	16 位 0 , <code>ADC1_ODT[15:0]</code>
		第二次	16 位 0 , <code>ADC2_ODT [15:0]</code>
		第三次	16 位 0 , <code>ADC1_ODT [15:0]</code>
010	单从机	第一次	<code>ADC2_ODT[15:0]</code> , <code>ADC1_ODT[15:0]</code>
		第二次	<code>ADC2_ODT[15:0]</code> , <code>ADC1_ODT[15:0]</code>
011	单从机	第一次	16 位 0 , <code>ADC2_ODT[7:0]</code> , <code>ADC1_ODT[7:0]</code>
		第二次	16 位 0 , <code>ADC2_ODT[7:0]</code> , <code>ADC1_ODT[7:0]</code>

配置 `ADC_CCTRL` 的 `MSDRCEN` 来决定 DMA 请求会一直持续不断直到没有数据可传输或是当 DMA 传输数量寄存器归零就停止。

19.5.2 同时模式

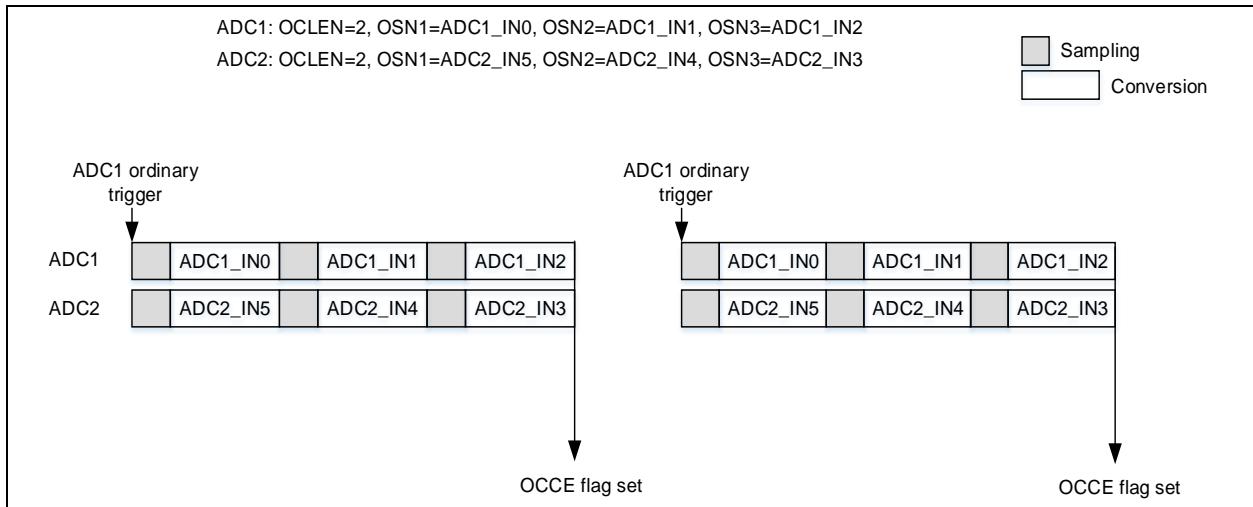
普通同时模式

配置 `ADC_CCTRL` 的 `MSSEL` 至普通同时模式后，可触发主机普通通道，使主机与从机同时转换普通通道。在此模式下，必须使用相同的采样时间以及相同的序列长度，以避免主从之间失去同步，遗失数据。下图示范了序列模式下的普通同时模式。

可搭配传输模式（`MSDMASEL`）中的模式 1、2 与 3。

注意：同样的通道不可同时被多个 ADC 采样，因此禁止将相同通道安排在不同 ADC 的同样序列位置。

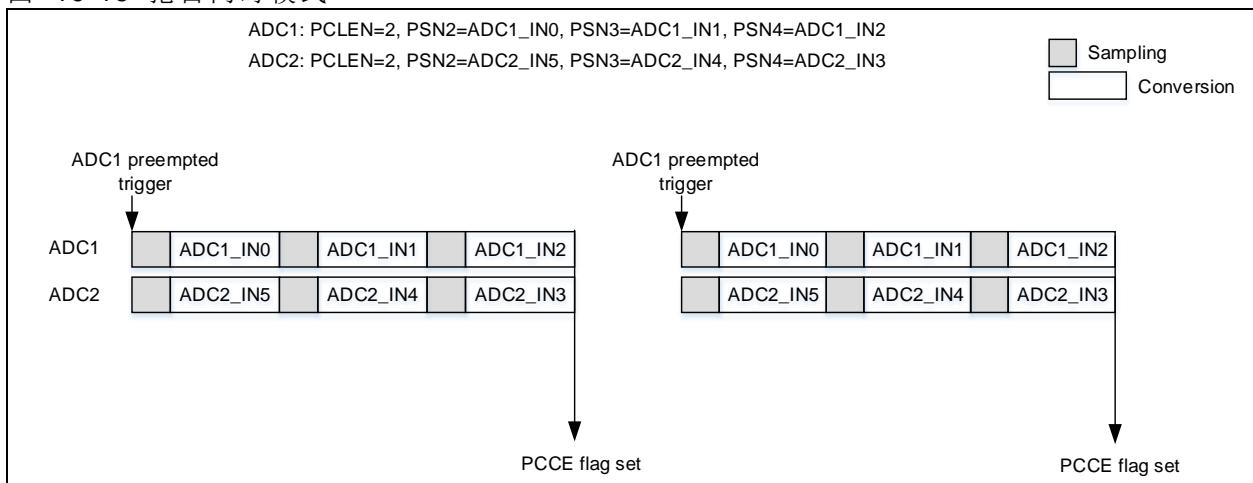
图 19-14 普通同时模式

**抢占同时模式**

配置 ADC_CCTRL 的 MSSEL 至抢占同时模式后，可触发主机抢占通道，使主机与从机同时转换抢占通道。下图示范了序列模式下的抢占同时模式。

注意：同样的通道不可同时被多个 ADC 采样，因此禁止将相同通道安排在不同 ADC 的同样序列位置。

图 19-15 抢占同时模式

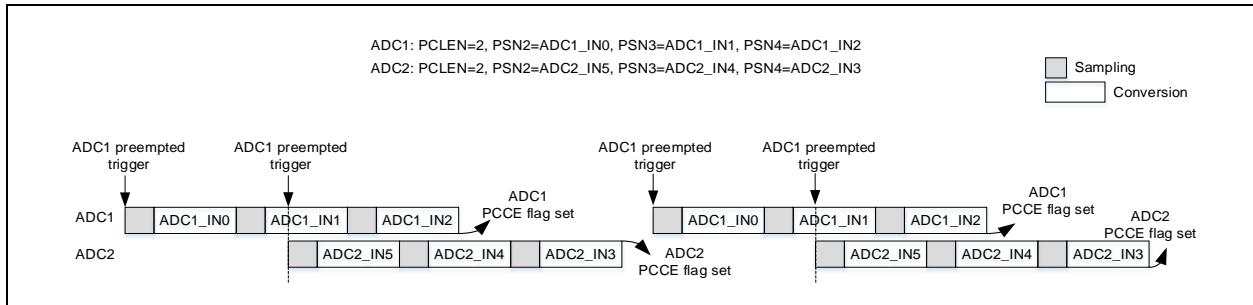
**混合的普通同时+抢占同时模式**

配置 ADC_CCTRL 的 MSSEL 至混合的普通同时+抢占同时模式后，可触发主机普通通道使主机与从机同时转换普通通道，也可触发主机抢占通道使主机与从机同时转换抢占通道。

19.5.3 抢占交错触发模式**抢占交错触发模式**

配置 ADC_CCTRL 的 MSSEL 至抢占交错触发模式后，可多次触发主机的抢占通道，促使主从 ADC 轮流转换抢占通道。下图示范了序列模式下的抢占交错触发模式。

图 19-16 抢占交错触发模式



混合的普通同时+抢占交错触发模式

配置 ADC_CCTRL 的 MSSEL 至混合的普通同时+抢占交错触发模式后，可触发主机普通通道使主机与从机同时转换普通通道，也可多次触发主机的抢占通道促使主从 ADC 轮流转换抢占通道。

当普通通道转换被抢占通道触发打断，所有的 ADC 停下普通通道转换，其中一个 ADC 进入抢占通道转换，此时主机将无视抢占通道触发，直到普通通道恢复转换后才会再接受抢占通道触发。

19.5.4 普通位移模式

配置 ADC_CCTRL 的 MSSEL 至普通位移模式后，可触发主机普通通道，使 ADC 之间自动在普通通道的转换上时序位移。

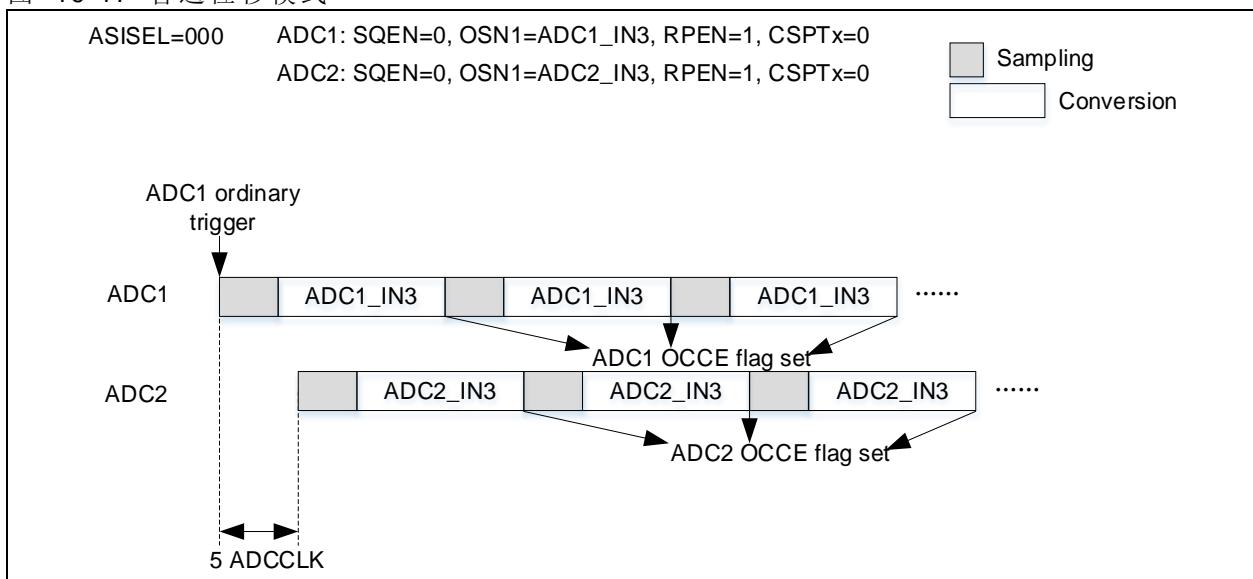
配置 ADC_CCTRL 寄存器中的 ASISEL 位选择位移的长度，如下图所示。

这个模式硬件会保证不同 ADC 间的采样间隔至少 2.5 个 ADCCLK，因此当 ASISEL 的配置无法满足这个条件时，ASISEL 将无效。利用这个特性，将相同通道安排在不同 ADC 的同样序列位置时，也不用担心采样时间重叠。

可搭配传输模式（MSDMASEL）中的模式 1、2 与 3。

注意：此模式下禁止抢占通道触发、禁止从机的普通通道触发。

图 19-17 普通位移模式



19.6 ADC寄存器

下表列出了 ADC 寄存器的映像和复位值。

必须以字(32 位)的方式操作这些外设寄存器。

表 19-5 ADC寄存器映像和复位值

寄存器简称	基址偏移量	复位值
ADC1_STS	0x000	0x0000 0000
ADC1_CTRL1	0x004	0x0000 0000
ADC1_CTRL2	0x008	0x0000 0000
ADC1_SPT1	0x00C	0x0000 0000
ADC1_SPT2	0x010	0x0000 0000
ADC1_PCDTO1	0x014	0x0000 0000
ADC1_PCDTO2	0x018	0x0000 0000
ADC1_PCDTO3	0x01C	0x0000 0000
ADC1_PCDTO4	0x020	0x0000 0000
ADC1_VMHB	0x024	0x0000 FFFF
ADC1_VMLB	0x028	0x0000 0000
ADC1_OSQ1	0x02C	0x0000 0000
ADC1_OSQ2	0x030	0x0000 0000
ADC1_OSQ3	0x034	0x0000 0000
ADC1_PSQ	0x038	0x0000 0000
ADC1_PDT1	0x03C	0x0000 0000
ADC1_PDT2	0x040	0x0000 0000
ADC1_PDT3	0x044	0x0000 0000
ADC1_PDT4	0x048	0x0000 0000
ADC1_ODT	0x04C	0x0000 0000
ADC1_OVSP	0x080	0x0000 0000
ADC2_STS	0x100	0x0000 0000
ADC2_CTRL1	0x104	0x0000 0000
ADC2_CTRL2	0x108	0x0000 0000
ADC2_SPT1	0x10C	0x0000 0000
ADC2_SPT2	0x110	0x0000 0000
ADC2_PCDTO1	0x114	0x0000 0000
ADC2_PCDTO2	0x118	0x0000 0000
ADC2_PCDTO3	0x11C	0x0000 0000
ADC2_PCDTO4	0x120	0x0000 0000
ADC2_VMHB	0x124	0x0000 FFFF
ADC2_VMLB	0x128	0x0000 0000
ADC2_OSQ1	0x12C	0x0000 0000
ADC2_OSQ2	0x130	0x0000 0000
ADC2_OSQ3	0x134	0x0000 0000
ADC2_PSQ	0x138	0x0000 0000
ADC2_PDT1	0x13C	0x0000 0000
ADC2_PDT2	0x140	0x0000 0000
ADC2_PDT3	0x144	0x0000 0000
ADC2_PDT4	0x148	0x0000 0000
ADC2_ODT	0x14C	0x0000 0000
ADC2_OVSP	0x180	0x0000 0000
ADC_CSTS	0x300	0x0000 0000
ADC_CCTRL	0x304	0x0000 0000
ADC_CODT	0x308	0x0000 0000

19.6.1 ADC状态寄存器 (ADC_STS)

访问：字访问

域	简称	复位值	类型	功能
位 31: 8	保留	0x000000	resd	请保持默认值。
位 7	TCF	0x0	rw0c	触发转换失败标志(Trigger convert fail flag) 当次触发转换失败，该位被硬件置起，由软件将其清零 (对自身写零) 0: 转换成功; 1: 转换失败。
位 6	RDY	0x0	ro	ADC 准备就绪标志 (ADC ready to conversion flag) 只读位，该位在 ADC 上电完毕后由硬件置位。

				0: 未就绪; 1: 已就绪。
位 5	OCCO	0x0	rw0c	普通通道转换溢出标志 (Ordinary channel conversion overflow flag) 该位被硬件置起, 由软件将其清零 (对自身写零) 0: 未发生溢出 1: 发生溢出 注: 溢出检测仅在使能 DMA 传输或者 EOCSFEN =1 时有效
位 4	OCCS	0x0	rw0c	普通通道转换开始标志 (Ordinary channel conversion start flag) 该位被硬件置起, 由软件将其清零 (对自身写零)。 0: 未开始; 1: 已开始。
位 3	PCCS	0x0	rw0c	抢占通道转换开始标志 (Preempted channel conversion start flag) 该位被硬件置起, 由软件将其清零 (对自身写零)。 0: 未开始; 1: 已开始。
位 2	PCCE	0x0	rw0c	抢占通道组转换结束标志 (Preempted channels conversion end flag) 该位被硬件置起, 由软件将其清零 (对自身写零)。 0: 未结束; 1: 已结束。
位 1	OCCE	0x0	rw0c	普通通道转换结束标志 (Ordinary channels conversion end flag) 该位被硬件置起, 由软件将其清零 (对自身写零), 或由读取 ADC_ODT 寄存器清零。 0: 未结束; 1: 已结束。
位 0	VMOR	0x0	rw0c	电压监测超出范围标志 (Voltage monitoring out of range flag) 该位被硬件置起, 由软件将其清零 (对自身写零)。 0: 无超出; 1: 有超出。

19.6.2 ADC控制寄存器1 (ADC_CTRL1)

访问: 字访问

域	简称	复位值	类型	功能
位 31: 29	保留	0x0	resd	请保持默认值。
位 28	TCFACA	0x0	rw	触发转换失败后自动终止转换(Trigger conversion fail auto conversion abort) 0: 关闭; 1: 开启。 注: 只可在非组合模式下开启此功能, 并且也只能在 TCFIEN 开启状态下使用此功能
位 27	TCFIEN	0x0	rw	触发转换失败中断使能(Trigger conversion fail interrupt enable) 0: 关闭; 1: 开启。
位 26	OCCOIEN	0x0	rw	普通通道转换溢出中断使能 (Ordinary channel conversion overflow interrupt enable) 0: 关闭; 1: 开启。
位 25:24	CRSEL	0x0	rw	转换分辨率选择 (Conversion resolution select) 00: 12 位; 01: 10 位; 10: 8 位; 11: 6 位。
位 23	OCVMEN	0x0	rw	普通通道的电压监测使能 (Voltage monitoring enable on ordinary channels)

				0: 关闭; 1: 开启。
位 22	PCVMEN	0x0	rw	抢占通道的电压监测使能 (Voltage monitoring enable on preempted channels) 0: 关闭; 1: 开启。
位 21: 16	保留	0x0	resd	请保持默认值。
位 15: 13	OCPCNT	0x0	rw	分割模式下每次触发转换的普通通道个数 (Partitioned mode conversion count of ordinary channels) 000: 1 个通道; 001: 2 个通道; 111: 8 个通道。 注: 抢占组在分割模式下每次触发固定只转换一个通道。
位 12	PCPEN	0x0	rw	抢占通道上的分割模式使能 (Partitioned mode enable on preempted channels) 0: 关闭; 1: 开启。
位 11	OCPEN	0x0	rw	普通通道上的分割模式使能 (Partitioned mode enable on ordinary channels) 该位由软件设置和清除, 用于开启或关闭普通通道组上的分割模式 0: 关闭; 1: 开启。
位 10	PCAUTOEN	0x0	rw	普通组转换结束后的抢占组自动转换使能 (Preempted group automatic conversion enable after ordinary group) 0: 关闭; 1: 开启。
位 9	VMSGEN	0x0	rw	单个通道的电压监测使能 (Voltage monitoring enable on a single channel) 0: 关闭 (电压监测所有通道); 1: 开启 (电压监测单一通道)。
位 8	SQEN	0x0	rw	序列模式使能 (Sequence mode enable) 0: 关闭 (转换选择的单一通道); 1: 开启 (转换设定的多个通道)。 注: 如果开启了 OCCEIEN 或 PCCEIEN 位, 则只在最后一个通道转换完毕后才会产生 OCCE 或 PCCE 中断。
位 7	PCCEIEN	0x0	rw	抢占通道组转换结束中断使能 (conversion end interrupt enable for Preempted channels) 0: 关闭; 1: 开启。
位 6	VMORIEN	0x0	rw	电压监测超出范围中断使能 (Voltage monitoring out of range interrupt enable) 0: 关闭; 1: 开启。
位 5	OCCEIEN	0x0	rw	普通通道转换结束中断使能 (Ordinary channel conversion end interrupt enable) 0: 关闭; 1: 开启。
位 4: 0	VMCSEL	0x00	rw	电压监测通道选择 (Voltage monitoring channel select) 仅在 VMSGEN 开启时有效。 00000: ADC_IN0 通道; 00001: ADC_IN1 通道; 01111: ADC_IN15 通道; 10000: ADC_IN16 通道; 10001: ADC_IN17 通道; 10010: ADC_IN18 通道 10011~11111: 未用, 禁止配置。

19.6.3 ADC控制寄存器2 (ADC_CTRL2)

访问：字访问

域	简称	复位值	类型	功能
位 30: 26	保留	0x00	resd	请保持默认值。
位 30	OCSWTRG	0x0	rw	软件触发普通通道转换 (Conversion trigger by software of ordinary channels) 0: 不触发; 1: 触发转换 (可由软件清除, 或在转换开始后由硬件自动清除)。
位 29:28	OCETE	0x0	rw	普通通道组的外部触发边沿选择 (Ordinary channel's external trigger edge select) 00: 禁止边沿触发; 01: 上升沿触发; 01: 下降沿触发; 11: 任意边沿触发。
位 31 位 27:24	OCTESEL	0x00	rw	普通通道组转换的触发事件选择 (Ordinary channel's conversion trigger event select) <small>注: 各 bit 的定义参考 19.4.2.2 章节</small>
位 22	PCSWTRG	0x0	rw	软件触发抢占通道转换 (Conversion trigger by software of preempted channels) 0: 不触发; 1: 触发转换 (可由软件清除, 或在转换开始后由硬件自动清除)。
位 21:20	PCETE	0x0	rw	抢占通道组的外部触发边沿选择 (Preempted channel's external trigger edge select) 00: 禁止边沿触发; 01: 上升沿触发; 01: 下降沿触发; 11: 任意边沿触发。
位 23 位 19:16	PCTESEL	0x00	rw	抢占通道组转换的触发事件选择 (Preempted channel's conversion trigger event select) <small>注: 各 bit 的定义参考 19.4.2.2 章节</small>
位 15:12	保留	0x0	resd	请保持默认值。
位 11	DTALIGN	0x0	rw	数据对齐方式 (Data alignment) 0: 右对齐; 1: 左对齐。
位 10	EOCSFEN	0x0	rw	每个普通通道转换置位 OCCE 标志使能 (Each ordinary channel conversion set OCCE flag enable) 0: 关闭; 1: 开启。 <small>注: 开启此位时, 溢出检测被自动使能。</small>
位 9	OCDRCEN	0x0	rw	普通通道数据的 DMA 请求接续使能 (Ordinary channel's DMA request continuation enable for independent mode) 0: 关闭 (传输完 DMA 设定个数后, 普通通道转换完毕不会再产生 DMA 请求); 1: 开启 (不关心 DMA 设定的个数, 每个普通通道转换完毕均产生 DMA 请求)。 <small>注: 此位仅作用于设定为非主从模式, 且 OCDMAEN = 1 时。</small>
位 8	OCDMAEN	0x0	rw	普通通道转换数据的 DMA 传输使能 (DMA transfer enable of ordinary channels) 0: 关闭; 1: 开启。
位 7: 5	保留	0x0	resd	请保持默认值。
位 4	ADABRT	0x0	rw	ADC 转换中止 (ADC conversion abort) 0: 无中止转换命令执行; 1: 中止当前正在执行的转换。

				注：当转换已中止时硬件将该位清零。该位被清零后，可重新进行触发转换。若 ADABRT 已置位，需等待 ADABRT 被硬件清零后进行其他 ADC 操作。
位 3	ADCALINIT	0x0	rw	A/D 初始化校准 (initialize A/D calibration) 该位由软件设置并由硬件清除。在校准寄存器被初始化后该位将被清除。 0: 校准寄存器无初始化执行或初始化结束； 1: 校准寄存器初始化或初始化进行中。
位 2	ADCAL	0x0	rw	A/D 校准 (A/D Calibration) 0: 无校准执行或校准结束； 1: 开始校准或校准进行中。
位 1	RPEN	0x0	rw	反复模式使能 (Repeat mode enable) 0: 关闭 SQEN=0 时，每次触发转换单个通道，SQEN=1 时，每次触发转换一组通道； 1: 开启 SQEN =0 时，一次触发后将反复转换单个通道，SQEN =1 时，一次触发后将反复转换一组通道。直到 ADCEN 被清零。
位 0	ADCEN	0x0	rw	A/D 转换器使能 (A/D converter enable) 0: 关闭 (ADC 进入断电模式)； 1: 开启。 注：当该位为关闭状态时，写入开启命令将把 ADC 从断电模式下唤醒。应用程序需注意，在转换器上电至转换开始有一个延迟 t_{STAB} 。

19.6.4 ADC采样时间寄存器1 (ADC_SPT1)

访问：字访问

域	简称	复位值	类型	功能
位 31: 27	保留	0x00	resd	请保持默认值。
位 26: 24	CSPT18	0x0	rw	选择 ADC_IN18 通道的采样时间 (Selection sample time of channel ADC_IN18) 000: 保留； 001: 6.5 周期； 010: 12.5 周期； 011: 24.5 周期； 100: 47.5 周期； 101: 92.5 周期； 110: 247.5 周期； 111: 640.5 周期。
位 23: 21	CSPT17	0x0	rw	选择 ADC_IN17 通道的采样时间 (Selection sample time of channel ADC_IN17) 000: 保留； 001: 6.5 周期； 010: 12.5 周期； 011: 24.5 周期； 100: 47.5 周期； 101: 92.5 周期； 110: 247.5 周期； 111: 640.5 周期。
位 20: 18	CSPT16	0x0	rw	选择 ADC_IN16 通道的采样时间 (Selection sample time of channel ADC_IN16) 000: 保留； 001: 6.5 周期； 010: 12.5 周期； 011: 24.5 周期； 100: 47.5 周期； 101: 92.5 周期； 110: 247.5 周期； 111: 640.5 周期。

位 17: 15 CSPT15	0x0	rw	选择 ADC_IN15 通道的采样时间 (Selection sample time of channel ADC_IN15) 000: 保留; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 14: 12 CSPT14	0x0	rw	选择 ADC_IN14 通道的采样时间 (Selection sample time of channel ADC_IN14) 000: 保留; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 11: 9 CSPT13	0x0	rw	选择 ADC_IN13 通道的采样时间 (Selection sample time of channel ADC_IN13) 000: 2.5 周期; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 8: 6 CSPT12	0x0	rw	选择 ADC_IN12 通道的采样时间 (Selection sample time of channel ADC_IN12) 000: 2.5 周期; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 5: 3 CSPT11	0x0	rw	选择 ADC_IN11 通道的采样时间 (Selection sample time of channel ADC_IN11) 000: 保留; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 2: 0 CSPT10	0x0	rw	选择 ADC_IN10 通道的采样时间 (Selection sample time of channel ADC_IN10) 000: 保留; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。

19.6.5 ADC采样时间寄存器2 (ADC_SPT2)

访问：字访问

域	简称	复位值	类型	功能
位 31: 30	保留	0x0	resd	请保持默认值。
位 29: 27	CSPT9	0x0	rw	选择 ADC_IN9 通道的采样时间 (Selection sample time of channel ADC_IN9) 000: 2.5 周期; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 26: 24	CSPT8	0x0	rw	选择 ADC_IN8 通道的采样时间 (Selection sample time of channel ADC_IN8) 000: 2.5 周期; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 23: 21	CSPT7	0x0	rw	选择 ADC_IN7 通道的采样时间 (Selection sample time of channel ADC_IN7) 000: 保留; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 20: 18	CSPT6	0x0	rw	选择 ADC_IN6 通道的采样时间 (Selection sample time of channel ADC_IN6) 000: 保留; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 17: 15	CSPT5	0x0	rw	选择 ADC_IN5 通道的采样时间 (Selection sample time of channel ADC_IN5) 000: 保留; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 14: 12	CSPT4	0x0	rw	选择 ADC_IN4 通道的采样时间 (Selection sample time of channel ADC_IN4) 000: 保留; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期;

				100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 11: 9	CSPT3	0x0	rw	选择 ADC_IN3 通道的采样时间 (Selection sample time of channel ADC_IN3) 000: 保留; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 8: 6	CSPT2	0x0	rw	选择 ADC_IN2 通道的采样时间 (Selection sample time of channel ADC_IN2) 000: 保留; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 5: 3	CSPT1	0x0	rw	选择 ADC_IN1 通道的采样时间 (Selection sample time of channel ADC_IN1) 000: 2.5 周期; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。
位 2: 0	CSPT0	0x0	rw	选择 ADC_IN0 通道的采样时间 (Selection sample time of channel ADC_IN0) 000: 2.5 周期; 001: 6.5 周期; 010: 12.5 周期; 011: 24.5 周期; 100: 47.5 周期; 101: 92.5 周期; 110: 247.5 周期; 111: 640.5 周期。

19.6.6 ADC抢占通道数据偏移寄存器x (ADC_PCDTOx) (x=1..4)

访问: 字访问

域	简称	复位值	类型	功能
位 31: 12	保留	0x00000	resd	请保持默认值。
位 11: 0	PCDTOx	0x000	rw	抢占通道 x 的数据偏移量设定 (Data offset for Preempted channel x) ADC_PDTx 内存放的转换数据 = 原始转换数据 - ADC_PCDTOx

19.6.7 ADC电压监测高边界寄存器 (ADC_VMHB)

访问：字访问

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	请保持默认值。
位 15: 0	VMHB	0xFFFF	rw	电压监测高边界设定 (Voltage monitoring high boundary)

19.6.8 ADC电压监测低边界寄存器 (ADC_VMLB)

访问：字访问

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	请保持默认值。
位 15: 0	VMLB	0x0000	rw	电压监测低边界设定 (Voltage monitoring low boundary)

19.6.9 ADC普通序列寄存器1 (ADC_OSQ1)

访问：字访问

域	简称	复位值	类型	功能
位 31: 24	保留	0x00	resd	请保持默认值。
位 23: 20	OCLEN	0x0	rw	普通转换序列长度 (Ordinary conversion sequence length) 0000: 1 个转换; 0001: 2 个转换; 1111: 16 个转换。
位 19: 15	OSN16	0x00	rw	普通序列中第 16 个转换通道的编号 (number of 16th conversion in ordinary sequence)
位 14: 10	OSN15	0x00	rw	普通序列中第 15 个转换通道的编号 (number of 15th conversion in ordinary sequence)
位 9: 5	OSN14	0x00	rw	普通序列中第 14 个转换通道的编号 (number of 14th conversion in ordinary sequence)
位 4: 0	OSN13	0x00	rw	普通序列中第 13 个转换通道的编号 (number of 13th conversion in ordinary sequence) 注：编号可设定 0~18，示例：设定为 3 就代表第 13 个转换的是 ADC_IN3 通道。

19.6.10 ADC普通序列寄存器2 (ADC_OSQ2)

访问：字访问

域	简称	复位值	类型	功能
位 31: 30	保留	0x0	resd	请保持默认值。
位 29: 25	OSN12	0x00	rw	普通序列中第 12 个转换通道的编号 (number of 12th conversion in ordinary sequence)
位 24: 20	OSN11	0x00	rw	普通序列中第 11 个转换通道的编号 (number of 11th conversion in ordinary sequence)
位 19: 15	OSN10	0x00	rw	普通序列中第 10 个转换通道的编号 (number of 10th conversion in ordinary sequence)
位 14: 10	OSN9	0x00	rw	普通序列中第 9 个转换通道的编号 (number of 8th conversion in ordinary sequence)
位 9: 5	OSN8	0x00	rw	普通序列中第 8 个转换通道的编号 (number of 8th conversion in ordinary sequence)
位 4: 0	OSN7	0x00	rw	普通序列中第 7 个转换通道的编号 (number of 7th conversion in ordinary sequence) 注：编号可设定 0~18，示例：设定为 8 就代表第 7 个转换的是 ADC_IN8 通道。

19.6.11 ADC普通序列寄存器3 (ADC_OSQ3)

访问：字访问

域	简称	复位值	类型	功能
位 31: 30	保留	0x0	resd	请保持默认值。
位 29: 25	OSN6	0x00	rw	普通序列中第 6 个转换通道的编号 (number of 6th conversion in ordinary sequence)
位 24: 20	OSN5	0x00	rw	普通序列中第 5 个转换通道的编号 (number of 5th conversion in ordinary sequence)
位 19: 15	OSN4	0x00	rw	普通序列中第 4 个转换通道的编号 (number of 4th conversion in ordinary sequence)
位 14: 10	OSN3	0x00	rw	普通序列中第 3 个转换通道的编号 (number of 3rd conversion in ordinary sequence)
位 9: 5	OSN2	0x00	rw	普通序列中第 2 个转换通道的编号 (number of 2nd conversion in ordinary sequence)
位 4: 0	OSN1	0x00	rw	普通序列中第 1 个转换通道的编号 (number of 1st conversion in ordinary sequence) 注：编号可设定 0~18，示例：设定为 17 就代表第 1 个转换的是 ADC_IN17 通道。

19.6.12 ADC 抢占序列寄存器 (ADC_PSQ)

访问：字访问

域	简称	复位值	类型	功能
位 31: 22	保留	0x000	resd	请保持默认值。
位 21: 20	PCLEN	0x0	rw	抢占转换序列长度 (Preempted conversion sequence length) 00: 1 个转换； 01: 2 个转换； 10: 3 个转换； 11: 4 个转换。
位 19: 15	PSN4	0x00	rw	抢占序列中第 4 个转换通道的编号 (number of 4th conversion in Preempted sequence)
位 14: 10	PSN3	0x00	rw	抢占序列中第 3 个转换通道的编号 (number of 3rd conversion in Preempted sequence)
位 9: 5	PSN2	0x00	rw	抢占序列中第 2 个转换通道的编号 (number of 2nd conversion in Preempted sequence)
位 4: 0	PSN1	0x00	rw	抢占序列中第 1 个转换通道的编号 (number of 1st conversion in Preempted sequence) 注：编号可设定 0~18，比如设定为 3 时其代表的就是 ADC_IN3 通道。若 PCLEN 小于 4，则转换的序列顺序是从 (4-PCLEN) 开始。例如：ADC_PSQ[21: 0] = 100110 001010 00100 00011，意味着扫描转换将按下列通道顺序执行：4、5、6，而不是 3、4、5。

19.6.13 ADC 抢占数据寄存器 x (ADC_PDTx) (x=1..4)

访问：字访问

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	请保持默认值。
位 15: 0	PDTx	0x0000	ro	抢占通道的转换数据 (Conversion data of preempted channel)

19.6.14 ADC普通数据寄存器 (ADC_ODT)

访问：字访问

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	请保持默认值。
位 15: 0	ODT	0x0000	ro	普通通道的转换数据 (Conversion data of ordinary channel)

19.6.15 ADC过采样寄存器 (ADC_OVSP)

访问：字访问

域	简称	复位值	类型	功能
位 31: 11	保留	0x0000000	resd	请保持默认值。
位 10	OOSRSEL	0x0	rw	普通过采样重转模式选择 (Ordinary oversampling restart mode select) 当普通过采样中途被抢占通道转换插入后，依据如下设定，选择普通过采样从哪里开始恢复转换。 0: 接续模式（普通过采样缓冲区会被保留）； 1: 重转模式（普通过采样缓冲区会被清零，即当前通道之前的采样次数被清零）。
位 9	OOSTREN	0x0	rw	普通过采样触发模式使能 (Ordinary oversampling trigger mode enable) 0: 关闭（通道的所有过采样转换仅需一次触发）； 1: 开启（通道的每个过采样转换均需进行触发）。
位 8: 5	OSSSEL	0x0	rw	过采样移位选择 (Oversampling shift select) 此位定义应用到最终过采样结果的右移位数。 0000: 不进行移位； 0001: 移 1 位； 0010: 移 2 位； 0011: 移 3 位； 0100: 移 4 位； 0101: 移 5 位； 0110: 移 6 位； 0111: 移 7 位； 1000: 移 8 位； 1001~1111: 未用，禁止配置。
位 4: 2	OSRSEL	0x0	rw	过采样率选择 (Oversampling ratio select) 000: 2x; 001: 4x; 010: 8x; 011: 16x; 100: 32x; 101: 64x; 110: 128x; 111: 256x。
位 1	POSEN	0x0	rw	抢占过采样使能 (Preempted oversampling enable) 0: 关闭； 1: 开启。
位 0	OOSEN	0x0	rw	普通过采样使能 (Ordinary oversampling enable) 0: 关闭； 1: 开启。

19.6.16 ADC通用状态寄存器 (ADC_STS)

访问：字访问

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	请保持默认值。
位 15	TCF2	0x0	ro	ADC2 触发转换失败标志(Trigger convert fail flag of ADC2) 该位是 ADC2_STS 寄存器中 TCF 位的映射。
位 14	RDY2	0x0	ro	ADC2 的准备就绪标志(ADC ready to conversion flag of ADC2) 该位是 ADC2_STS 寄存器中 RDY 位的映射。
位 13	OCCO2	0x0	ro	ADC2 的普通通道转换溢出标志(Ordinary channel conversion overflow flag of ADC2) 该位是 ADC2_STS 寄存器中 OCCO 位的映射。
位 12	OCCS2	0x0	ro	ADC2 的普通通道转换开始标志(Ordinary channel conversion start flag of ADC2) 该位是 ADC2_STS 寄存器中 OCCS 位的映射。
位 11	PCCS2	0x0	ro	ADC2 的抢占通道转换开始标志(Preempted channel conversion start flag of ADC2) 该位是 ADC2_STS 寄存器中 PCCS 位的映射。
位 10	PCCE2	0x0	ro	ADC2 的抢占通道组转换结束标志(Preempted channels conversion end flag of ADC2) 该位是 ADC2_STS 寄存器中 PCCE 位的映射。
位 9	OCCE2	0x0	ro	ADC2 的普通通道转换结束标志(Ordinary channels conversion end flag of ADC2) 该位是 ADC2_STS 寄存器中 OCCE 位的映射。
位 8	VMOR2	0x0	ro	ADC2 的电压检测超出范围标志 (Voltage monitoring out of range flag of ADC2) 该位是 ADC2_STS 寄存器中 VMOR 位的映射。
位 7	TCF1	0x0	ro	ADC1 触发转换失败标志(Trigger convert fail flag of ADC1) 该位是 ADC1_STS 寄存器中 TCF 位的映射。
位 6	RDY1	0x0	ro	ADC1 的准备就绪标志(ADC ready to conversion flag of ADC1) 该位是 ADC1_STS 寄存器中 RDY 位的映射。
位 5	OCCO1	0x0	ro	ADC1 的普通通道转换溢出标志(Ordinary channel conversion overflow flag of ADC1) 该位是 ADC1_STS 寄存器中 OCCO 位的映射。
位 4	OCCS1	0x0	ro	ADC1 的普通通道转换开始标志(Ordinary channel conversion start flag of ADC1) 该位是 ADC1_STS 寄存器中 OCCS 位的映射。
位 3	PCCS1	0x0	ro	ADC1 的抢占通道转换开始标志(Preempted channel conversion start flag of ADC1) 该位是 ADC1_STS 寄存器中 PCCS 位的映射。
位 2	PCCE1	0x0	ro	ADC1 的抢占通道组转换结束标志(Preempted channels conversion end flag of ADC1) 该位是 ADC1_STS 寄存器中 PCCE 位的映射。
位 1	OCCE1	0x0	ro	ADC1 的普通通道转换结束标志(Ordinary channels conversion end flag of ADC1) 该位是 ADC1_STS 寄存器中 OCCE 位的映射。
位 0	VMOR1	0x0	ro	ADC1 的电压检测超出范围标志 (Voltage monitoring out of range flag of ADC1) 该位是 ADC1_STS 寄存器中 VMOR 位的映射。

19.6.17 ADC通用控制寄存器 (ADC_CCTRL)

访问：字访问

域	简称	复位值	类型	功能
位 31: 24	保留	0x0000	resd	请保持默认值。
位 23	ITSRVEN	0x0	rw	内部温度传感器及 V _{INTRV} 使能 (Internal temperature sensor and V _{INTRV} enable) 0: 关闭; 1: 开启。
位 22	VBATEN	0x0	rw	V _{BAT} 使能 (V _{BAT} enable) 0: 关闭; 1: 开启。
位 21: 20	保留	0x0	resd	请保持默认值。
位 19: 16	ADCDIV	0x0	rw	ADC 分频因子 (ADC division) 0000: HCLK 2 分频 0001: HCLK 3 分频 ... 1111: HCLK 17' 分频 注：经此分频后的时钟将被所有 ADC 共用； ADCCLK 额定最大为 80MHz，经分频后 ADCCLK 不能高于 PCLK2。
位 28	MSDMASEL	0x0	rw	主从模式下普通通道数据的 DMA 传输模式选择 (Ordinary channel's DMA transfer mode select for master slave mode) MSDMASEL[2]位于 ADC_CCTRL 寄存器的第 28 位，其中 MSDMASEL[2: 0]定义如下： 000: 禁止 DMA 传输； 001: 使用 DMA 模式 1； 010: 使用 DMA 模式 2； 011: 使用 DMA 模式 3； 100~111: 未用，禁止配置。 注：此位仅作用于设定为主从模式时，具体 DMA 模式 x 的使用规则请参考数据管理章节描述。
位 15: 14				
位 13	MSDRCEN	0x0	rw	主从模式下普通通道数据的 DMA 请求接续使能 (Ordinary channel's DMA request continuation enable for master slave mode) 0: 关闭（传输完 DMA 设定个数后，普通通道转换完毕不会再产生 DMA 请求）； 1: 开启（不关心 DMA 设定的个数，每个普通通道转换完毕均产生 DMA 请求）。 注：此位仅作用于设定为主从模式时，且 MSDMASEL 选择为 DMA 模式 x 时。
位 12	保留	0x0	resd	请保持默认值。
位 11: 8	ASISEL	0x0	rw	普通位移模式下相邻 ADC 间的采样间隔时间选择 (Adjacent ADC sampling interval select for ordinary shifting mode) 0000: 5 * TADCCLK 0001: 6 * TADCCLK 0010: 7 * TADCCLK ... 1111: 20 * TADCCLK 注：这些位用于设定普通位移模式下相邻 ADC 转换间的时间间隔，具体间隔规则请参考普通位移模式章节描述。
位 7: 5	保留	0x0	resd	请保持默认值。
位 4: 0	MSSEL	0x00	rw	主从组合模式选择 (Master slave mode select) 00000: 非组合模式 00001: 混合的普通同时+抢占同时模式； 00010: 混合的普通同时+抢占交错触发模式； 00011~00100: 未用，禁止配置； 00101: 抢占同时模式； 00110: 普通同时模式； 00111: 普通位移模式；

01000: 未用, 禁止配置;

01001: 抢占交错触发模式;

01010~11111: 未用, 禁止配置;

注: 在主从组合模式中, 修改配置会导致主从丢失同步规则。建议在修改前先关闭主从组合模式。

19.6.18 ADC通用普通数据寄存器 (ADC_CODT)

访问: 字访问

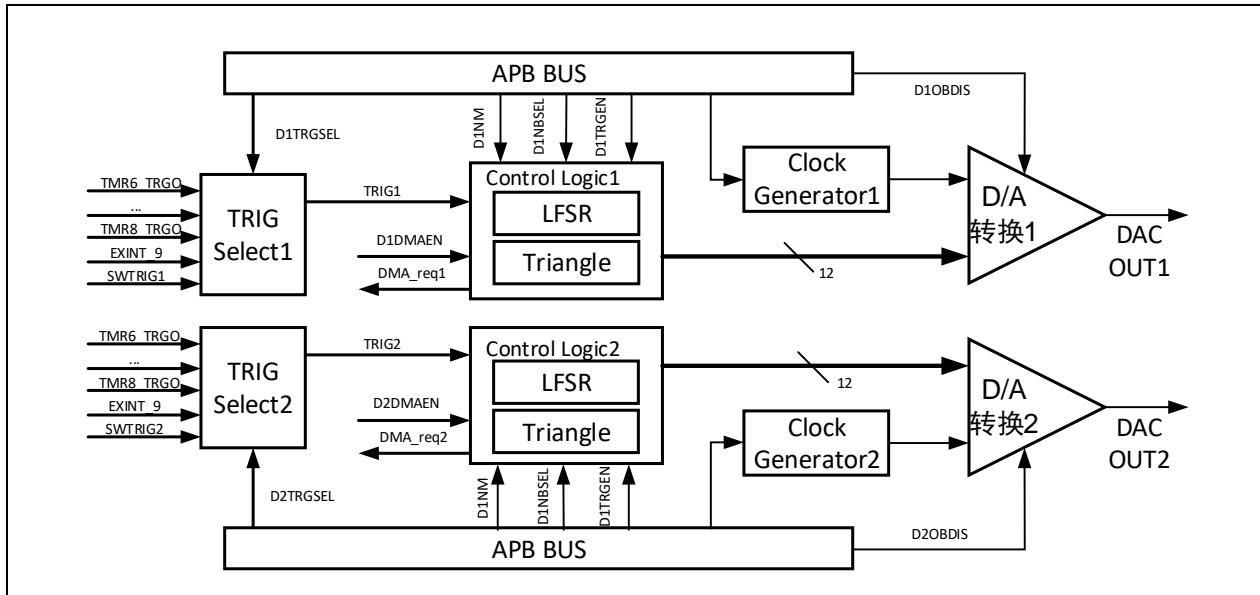
域	简称	复位值	类型	功能
位 31: 16	CODTH	0x0000	rw	主从模式下普通转换的高半字数据 (Ordinary conversion's high halfword data for master slave mode) 注: 在不同的 DMA 模式中, 这些位代表的数据含义存在区别。具体数据存储规则请参考 19.5.1 章节描述。
位 15: 0	CODTL	0x0000	rw	主从模式下普通转换的低半字数据 (Ordinary conversion's low halfword data for master slave mode) 注: 在不同的 DMA 模式中, 这些位代表的数据含义存在区别。具体数据存储规则请参考 19.5.1 章节描述。

20 数字/模拟转换 (DAC)

20.1 简介

数模转换器 (DAC) 采用 12 位数字输入，产生 0 至参考电压之间的模拟输出。数字部分可以配置为 8 位或者 12 位模式，支持单/双 DAC 的左对齐或者右对齐，同时可以与 DMA 配合使用。两个 DAC1/DAC2 各有一个数模转换器，每个 DAC1/DAC2 可以独立进行数模转换，也可以双 DAC 同时触发进行转换，输入参考电压 VINTRV 可以使转换操作更加精确。

图 20-1 DAC1/DAC2模块框图



20.2 主要特性

- 单/双 DAC 8 位或者 12 位数字输入
- 数据支持左对齐或者右对齐模式
- 支持噪声波/三角波产生
- 双 DAC 或者单个 DAC1/DAC2 独立转换
- 每个 DAC1/DAC2 支持 DMA 模式
- 软件触发或者外部触发转换
- 支持输入参考电压 VINTRV

20.3 设计提示

DAC 有以下提示仅供设计参考。

● 模拟模块配置

DAC1/DAC2 的模拟部分由 DAC 控制寄存器 (DAC_CTRL) ENx 位控制开启，数字部分则不受该位控制。另外 DAC 集成了 2 个输出增益，可以用来减少输出阻抗，无需外部运放即可直接驱动外部负载。每个 DAC1/DAC2 输出增益可以通过设置 DAC 控制寄存器 (DAC_CTRL) 的 DxOBDIS 位来使能或关闭。

● DMA 功能

任一 DAC1/DAC2 支持 DMA 功能，通过设置 DAC 控制寄存器 (DAC_CTRL) 的 DxDMAEN 位使能 DMA 请求。当触发使能位 DxTRGEN 有效，触发信号有效时，即产生 DMA 请求。DAC 的 DMA 请求不会累计，未来得及处理的 DMA 请求将被忽略，也不会产生错误信息。

在双 DAC 模式下，程序可以只使用一个 DMA 请求，一个 DMA 通道的情况下，处理工作在双 DAC 模式的 2 个 DAC1/DAC2。

● DMA 下溢处理

当开启 DAC 的 DMA 功能，如果第二个外部触发到达时尚未收到 DMA 反馈的第一个外部

触发的确认信号，即产生 DMA 下溢。此时将不会处理新的外部触发，并且不会发出新的 DMA 请求，同时 DAC 状态寄存器（DAC_SR）DxDMAUDRF 位指示发生 DMA 下溢错误，配置 DAC 控制寄存器（DAC_CTRL）DxDMAUDRIEN 位为‘1’，将产生相应的 DMA 下溢错误中断。

软件通过写入‘1’来清除 DAC 状态寄存器（DAC_SR）DxDMAUDRF 位，在重新开始 DMA 传输之前，应该先清除 DAC 控制寄存器（DAC_CTRL）的 DxDMAEN 位，然后重新初始化 DMA 和 DAC。

- 输入输出配置

数字输入经过 DAC 线性地转换为模拟电压输出，其范围为 0 至参考电压。模拟 DAC 模块采用 VDDA 供电，输入正模拟参考电压大小介于 2.0V 与 VDDA 之间，PA4 或者 PA5 作为模拟输出时，为避免寄生干扰和额外的功耗，需设置为模拟输入。

$$\text{DAC 输出} = \text{VINTRV} \times (\text{DxDOT}[11:0] / 4095)$$

20.4 功能描述

20.4.1 触发事件

如果 DAC_CTRL 寄存器的 DxTRGEN 位被置 1，DAC 转换可以由某外部事件（定时器计数器、外部中断线）或者软件触发，触发事件源由 DxTRGSEL[2:0]进行选择。

表 20-1 触发源选择

触发源	DxTRGSEL [2: 0]	说明
TMR6_TRGOUT	000	片上信号
TMR8_TRGOUT	001	
TMR7_TRGOUT	010	
TMR5_TRGOUT	011	
TMR2_TRGOUT	100	
TMR4_TRGOUT	101	
EXINT_9	110	外部信号
DxSWTRG	111	软件触发

当 DxTRGEN 位被置 1 时，每次 DAC 侦测到有效的的触发事件，存放在 DHRx 中的数据就会被传动到寄存器 DAC_DxDOT 中，当选择软件触发时，触发标志 DxSWTRG 在被软件置‘1’后，硬件自动清零。一旦数据装入 DAC_DxDOT 寄存器，经过一段时间，模拟的数模转换器输出即有效。

当 DxTRGEN 位被清‘0’时，每次写入数据寄存器值时，数据即被传送到寄存器 DAC_DxDOT 中，无需等待触发事件。

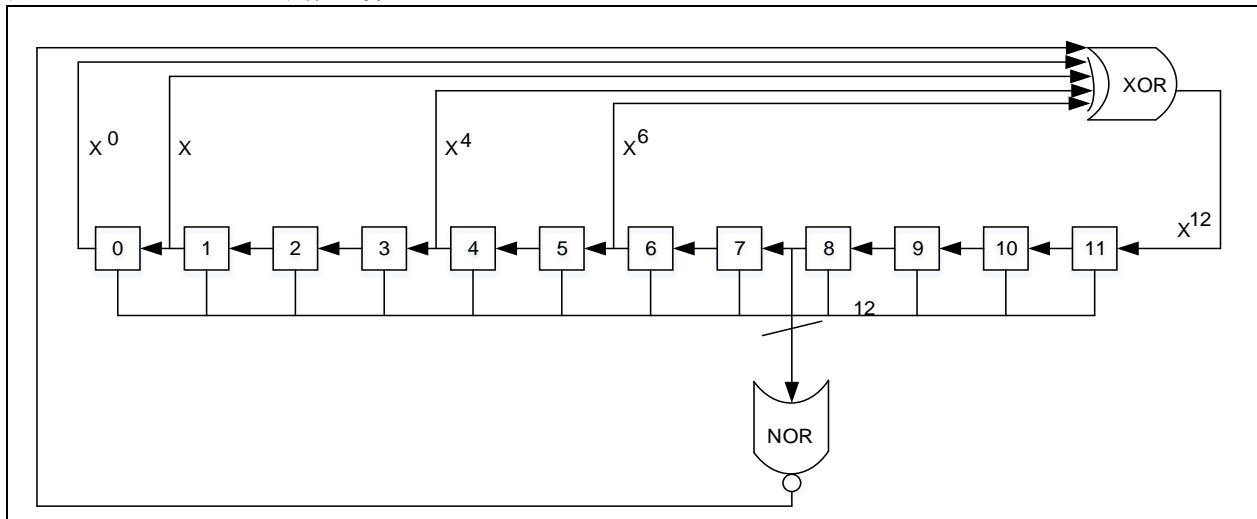
20.4.2 噪声/三角波生成

有噪声波和三角波两种波形可以叠加到 DAC 输出：分别利用线性反馈移位寄存器（Linear Feedback Shift Register LFSR）产生幅度变化的伪噪声和通过三角波发生器（triangle）产生三角波。当设置 DxNM[1:0]位为‘01’使能 LFSR，输出幅度变化的伪噪声。当设置 DxNM[1:0]位为‘1x’使能三角波发生器，输出三角波。

LFSR 原理

寄存器 LFSR 的预装入值为 0xAAA，按照特定算法，在每次触发事件之后更新该寄存器的值。

图 20-2 DAC LFSR 寄存器算法



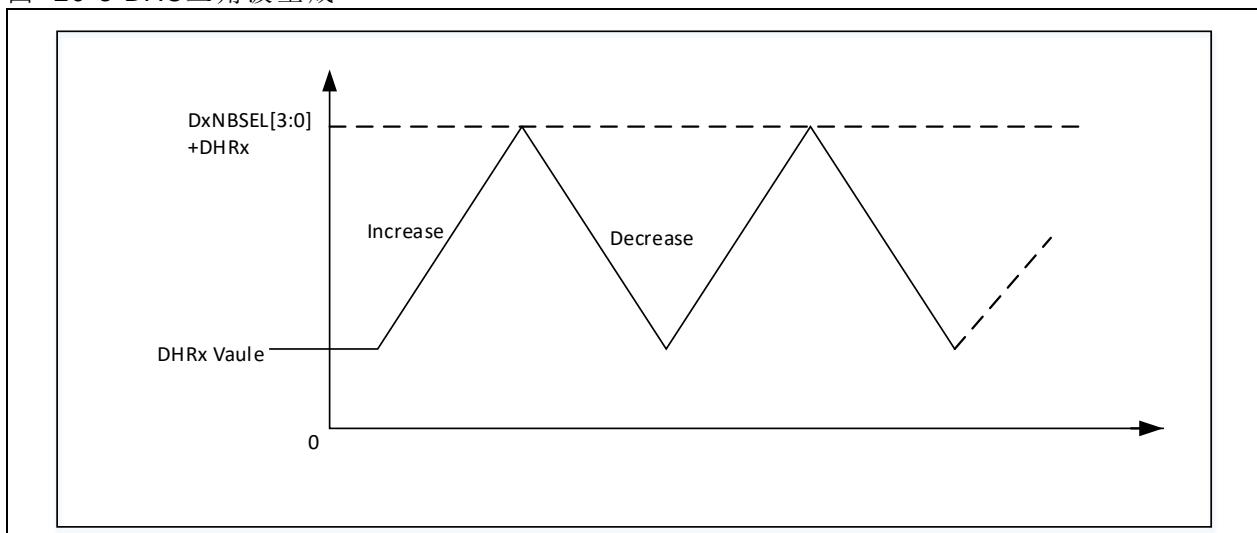
设置 DAC 控制寄存器 (DAC_CTRL) 的 DxNBSEL[3: 0]位可以屏蔽部分或者全部 LFSR 的数据，这样得到的 LSFR 值与 DHRx 的数值相加，去掉溢出位之后即被写入 DAC1/DAC2 数据输出寄存器 (DAC_DxDOT)。将 DxNM[1: 0]位置‘00’可以关掉 LFSR 功能，同时复位 LFSR 波形的生成算法。

三角波原理

当设置 DxNM[1: 0]位为‘1x’，则选择 DAC 的三角波生成功能。三角波的幅度由 DAC 控制寄存器 (DAC_CTRL) 的 DxNBSEL [3: 0]位设置，内部的三角波计数器每检测到一次触发事件累加 1，当达到 DxNBSEL [3: 0]位定义的最大幅度时，则计数器开始递减，达到 0 后再开始累加，周而复始。

同时，计数器的值与 DHRx 寄存器的数值相加并丢弃溢出位后写入 DAC1/DAC2 数据输出寄存器 (DAC_DxDOT)。将 DxNM[1: 0]设置‘00’，可以关掉三角波生成，同时复位三角波计数器。

图 20-3 DAC 三角波生成



20.4.3 数据配置

DAC 支持单 DAC 或者双 DAC 模式，根据模式的不同，数据配置有以下方式：

单 DAC 数据配置方式：采用 8 位数据右对齐，或者 12 位数据左对齐，或者 12 位数据右对齐方式时，对寄存器 DAC_DxDTH8R [7: 0]，或者 DAC_DxDTH12L [15: 4]，或者 DAC_DxDTH12R [11: 0]位写入值。

双 DAC 数据配置方式：采用 8 位数据右对齐，或者 12 位数据左对齐，或者 12 位数据右对齐方式时，对寄存器 DAC_DDTH8R [7: 0]和 DAC_DDTH8R [15: 8]，或者 DAC_DDTH12L [15: 4]和 DAC_DDTH12L [31: 20]，或者 DAC_DDTH12R [11: 0]和 DAC_DDTH12R [27: 16]位写入值。

写入的 8 位数据对应 DHRx[11: 4]位，写入的 12 位数据对应 DHRx[11: 0]位。

20.5 DAC寄存器

下表列出了所有 DAC 寄存器。

必须以字（32 位）的方式操作这些外设寄存器。

表 20-2 DAC寄存器映像和复位值

寄存器简称	基址偏移量	复位值
DAC_CTRL	000h	0x0000 0000
DAC_SWTRG	004h	0x0000 0000
DAC_D1DTH12R	008h	0x0000 0000
DAC_D1DTH12L	00Ch	0x0000 0000
DAC_D1DTH8R	010h	0x0000 0000
DAC_D2DTH12R	014h	0x0000 0000
DAC_D2DTH12L	018h	0x0000 0000
DAC_D2DTH8R	01Ch	0x0000 0000
DAC_DDTH12R	020h	0x0000 0000
DAC_DDTH12L	024h	0x0000 0000
DAC_DDTH8R	028h	0x0000 0000
DAC_D1ODT	02Ch	0x0000 0000
DAC_D2ODT	030h	0x0000 0000
DAC_STS	034h	0x0000 0000

20.5.1 DAC控制寄存器 (DAC_CTRL)

域	简称	复位值	类型	功能
位 31: 30	保留	0x0	resd	请保持默认值。
位 29	D2DMAUDRIEN	0x0	rw	DAC2 的 DMA 传输下溢中断使能 (DAC2 DMA transfer underrun interrupt enable) 该位由软件设置和清除。 0: 关闭 DAC2 DMA 下溢中断; 1: 使能 DAC2 DMA 下溢中断。
位 28	D2DMAEN	0x0	rw	DAC2 的 DMA 传输使能 (DAC2 DMA transfer enable) 该位由软件设置和清除。 0: 关闭 DAC2 DMA 模式; 1: 使能 DAC2 DMA 模式。
位 27: 24	D2NBSEL	0x0	rw	DAC2 噪声位宽选择 (DAC2 noise bit select) 这些位用于在噪声生成模式下选择屏蔽位，在三角波生成模式下选择波形的幅值。 0000: 不屏蔽 LSFR 位 0 / 三角波幅值等于 1; 0001: 不屏蔽 LSFR 位[1: 0] / 三角波幅值等于 3; 0010: 不屏蔽 LSFR 位[2: 0] / 三角波幅值等于 7; 0011: 不屏蔽 LSFR 位[3: 0] / 三角波幅值等于 15; 0100: 不屏蔽 LSFR 位[4: 0] / 三角波幅值等于 31; 0101: 不屏蔽 LSFR 位[5: 0] / 三角波幅值等于 63; 0110: 不屏蔽 LSFR 位[6: 0] / 三角波幅值等于 127; 0111: 不屏蔽 LSFR 位[7: 0] / 三角波幅值等于 255; 1000: 不屏蔽 LSFR 位[8: 0] / 三角波幅值等于 511; 1001: 不屏蔽 LSFR 位[9: 0] / 三角波幅值等于 1023; 1010: 不屏蔽 LSFR 位[10: 0] / 三角波幅值等于 2047; ≥1011: 不屏蔽 LSFR 位[11: 0] / 三角波幅值等于 4095。 注意：这些位在 D2TRGEN = 1 时有效。

				DAC2 噪声/三角波生成选择 (DAC2 noise mode)
位 23: 22 D2NM	0x0	rw		00: 关闭; 01: 开启噪声波形发生器; 1x: 开启三角波发生器。
位 21: 19 D2TRGSEL	0x0	rw		DAC2 的触发事件选择 (DAC2 trigger select) 000: TMR6 TRGOUT 事件; 001: TMR8 TRGOUT 事件; 010: TMR7 TRGOUT 事件; 011: TMR5 TRGOUT 事件; 100: TMR2 TRGOUT 事件; 101: TMR4 TRGOUT 事件; 110: 外部中断线 9; 111: 软件触发。 注意: 这些位在 D2TRGEN = 1 时有效。
位 18 D2TRGEN	0x0	rw		DAC2 触发使能 (DAC2 trigger enable) 0: 关闭; 1: 开启。 注: 关闭触发时, 写入寄存器 DAC_D2DTHx 的数据在 1 个 APB1 时钟周期后传入寄存器 DAC_D2ODT。 开启触发时, 被选择的触发事件发生之后, 寄存器 DAC_D2DTHx 的数据在 3 个 APB1 时钟周期后传入寄存器 DAC_D2ODT。 如果选择软件触发, 写入寄存器 DAC_D2DTHx 的资料只需要 1 个 APB1 时钟周期就可以传入寄存器 DAC_D2ODT。
位 17 D2OBDIS	0x0	rw		关闭 DAC2 输出缓存 (DAC2 output buffer disable) 0: 开启输出缓存; 1: 关闭输出缓存。
位 16 D2EN	0x0	rw		DAC2 使能 (DAC2 enable) 0: 关闭; 1: 开启。
位 15: 14 保留	0x0	resd		请保持默认值。
位 13 D1DMAUDRIEN	0x0	rw		DAC1 的 DMA 传输下溢中断使能 (DAC1 DMA transfer underrun interrupt enable) 该位由软件设置和清除。 0: 关闭 DAC1 DMA 下溢中断; 1: 使能 DAC1 DMA 下溢中断。
位 12 D1DMAEN	0x0	rw		DAC1 的 DMA 传输使能 (DAC1 DMA transfer enable) 该位由软件设置和清除。 0: 关闭 DAC1 DMA 模式; 1: 使能 DAC1 DMA 模式。
位 11: 8 D1NBSEL	0x0	rw		DAC1 屏蔽/幅值选择 (DAC1 noise bit select) 这些位用于在噪声生成模式下选择屏蔽位, 在三角波生成模式下选择波形的幅值。 0000: 不屏蔽 LSFR 位 0 / 三角波幅值等于 1; 0001: 不屏蔽 LSFR 位[1: 0] / 三角波幅值等于 3; 0010: 不屏蔽 LSFR 位[2: 0] / 三角波幅值等于 7; 0011: 不屏蔽 LSFR 位[3: 0] / 三角波幅值等于 15; 0100: 不屏蔽 LSFR 位[4: 0] / 三角波幅值等于 31; 0101: 不屏蔽 LSFR 位[5: 0] / 三角波幅值等于 63; 0110: 不屏蔽 LSFR 位[6: 0] / 三角波幅值等于 127; 0111: 不屏蔽 LSFR 位[7: 0] / 三角波幅值等于 255; 1000: 不屏蔽 LSFR 位[8: 0] / 三角波幅值等于 511; 1001: 不屏蔽 LSFR 位[9: 0] / 三角波幅值等于 1023; 1010: 不屏蔽 LSFR 位[10: 0] / 三角波幅值等于 2047; ≥1011: 不屏蔽 LSFR 位[11: 0] / 三角波幅值等于 4095。 注: 这些位在 D1TRGEN= 1 时有效。
位 7: 6 D1NM	0x0	rw		DAC1 噪声/三角波生成选择 (DAC1 noise mode)

				00: 关闭; 01: 开启噪声波形发生器; 1x: 开启三角波发生器。
位 5: 3	D1TRGSEL	0x0	rw	DAC1 的触发事件选择 (DAC1 trigger select) 000: TMR6 TRGOUT 事件; 001: TMR8 TRGOUT 事件; 010: TMR7 TRGOUT 事件; 011: TMR5 TRGOUT 事件; 100: TMR2 TRGOUT 事件; 101: TMR4 TRGOUT 事件; 110: 外部中断线 9; 111: 软件触发。 注: 这些位在 D1TRGEN= 1 时有效。
位 2	D1TRGEN	0x0	rw	DAC1 触发使能 (DAC1 trigger enable) 0: 关闭; 1: 开启。 注: 关闭触发时, 写入寄存器 DAC_D1DTHx 的数据在 1 个 APB1 时钟周期后传入寄存器 DAC_D1ODT。 开启触发时, 被选择的触发事件发生之后, 寄存器 DAC_D1DTHx 的数据在 3 个 APB1 时钟周期后传入寄存器 DAC_D1ODT。 如果选择软件触发, 写入寄存器 DAC_D1DTHx 的资料只需要 1 个 APB1 时钟周期就可以传入寄存器 DAC_D1ODT。
位 1	D1OBDIS	0x0	rw	关闭 DAC1 输出缓存 (DAC1 output buffer disable) 0: 开启输出缓存; 1: 关闭输出缓存。
位 0	D1EN	0x0	rw	DAC1 使能 (DAC1 enable) 0: 关闭; 1: 开启。

20.5.2 DAC软件触发寄存器 (DAC_SWTRG)

域	简称	复位值	类型	功能
位 31: 2	保留	0x0000 0000	resd	请保持默认值。
位 1	D2SWTRG	0x0	rw	DAC2 软件触发 (DAC2 software trigger) 0: 不触发; 1: 触发。 注: 一旦寄存器 DAC_D2DTH 的数据传入寄存器 DAC_D2ODT, (1 个 APB1 时钟周期后) 该位由硬件清零。
位 0	D1SWTRG	0x0	rw	DAC1 软件触发 (DAC1 software trigger) 0: 不触发; 1: 触发。 注: 一旦寄存器 DAC_D1DTH 的数据传入寄存器 DAC_D1ODT, (1 个 APB1 时钟周期后) 该位由硬件清零。

20.5.3 DAC1的12位右对齐数据保持寄存器 (DAC_D1DTH12R)

域	简称	复位值	类型	功能
位 31: 12	保留	0x00000	resd	请保持默认值。
位 11: 0	D1DT12R	0x000	rw	DAC1 的 12 位右对齐数据 (DAC1 12-bit right-aligned data)

20.5.4 DAC1的12位左对齐数据保持寄存器 (DAC_D1DTH12L)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	请保持默认值。

位 15: 4	D1DT12L	0x000	rw	DAC1 的 12 位左对齐数据 (DAC1 12-bit left-aligned data)
位 3: 0	保留	0x0	resd	请保持默认值。

20.5.5 DAC1的8位右对齐数据保持寄存器 (DAC_D1DTH8R)

域	简称	复位值	类型	功能
位 31: 8	保留	0x000000	resd	请保持默认值。
位 7: 0	D1DT8R	0x00	rw	DAC1 的 8 位右对齐数据 (DAC1 8-bit right-aligned data)

20.5.6 DAC2的12位右对齐数据保持寄存器 (DAC_D2DTH12R)

域	简称	复位值	类型	功能
位 31: 12	保留	0x000000	resd	请保持默认值。
位 11: 0	D2DT12R	0x000	rw	DAC2 的 12 位右对齐数据 (DAC2 12-bit right-aligned data)

20.5.7 DAC2的12位左对齐数据保持寄存器 (DAC_D2DTH12L)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	请保持默认值。
位 15: 4	D2DT12L	0x000	rw	DAC2 的 12 位左对齐数据 (DAC2 12-bit left-aligned data)
位 3: 0	保留	0x0	resd	请保持默认值。

20.5.8 DAC2的8位右对齐数据保持寄存器 (DAC_D2DTH8R)

域	简称	复位值	类型	功能
位 31: 8	保留	0x000000	resd	请保持默认值。
位 7: 0	D2DT8R	0x00	rw	DAC2 的 8 位右对齐数据 (DAC2 8-bit right-aligned data)

20.5.9 双DAC的12位右对齐数据保持寄存器 (DAC_DDTH12R)

域	简称	复位值	类型	功能
位 31: 28	保留	0x0	resd	请保持默认值。
位 27: 16	DD2DT12R	0x000	rw	DAC2 的 12 位右对齐数据 (DAC2 12-bit right-aligned data)
位 15: 12	保留	0x0	resd	请保持默认值。
位 11: 0	DD1DT12R	0x000	rw	DAC1 的 12 位右对齐数据 (DAC1 12-bit right-aligned data)

20.5.10 双DAC的12位左对齐数据保持寄存器 (DAC_DDTH12L)

域	简称	复位值	类型	功能
位 31: 20	DD2DT12L	0x000	rw	DAC2 的 12 位左对齐数据 (DAC2 12-bit left-aligned data)
位 19: 16	保留	0x0	resd	请保持默认值。
位 15: 4	DD1DT12L	0x000	rw	DAC1 的 12 位左对齐数据 (DAC1 12-bit left-aligned data)
位 3: 0	保留	0x0	resd	请保持默认值。

20.5.11 双DAC的8位右对齐数据保持寄存器 (DAC_DDTH8R)

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	请保持默认值。
位 15: 8	DD2DT8R	0x00	rw	DAC2 的 8 位右对齐数据 (DAC2 8-bit right-aligned data)

位 7: 0	DD1DT8R	0x00	rw	DAC1 的 8 位右对齐数据（DAC1 8-bit right-aligned data）
--------	---------	------	----	--

20.5.12 DAC1 数据输出寄存器 (DAC_D1ODT)

域	简称	复位值	类型	功能
位 31: 12	保留	0x00000	resd	请保持默认值。
位 11: 0	D1ODT	0x000	rw	DAC1 输出数据 (DAC1 output data)

20.5.13 DAC2 数据输出寄存器 (DAC_D2ODT)

域	简称	复位值	类型	功能
位 31: 12	保留	0x00000	resd	请保持默认值。
位 11: 0	D2ODT	0x000	rw	DAC2 输出数据 (DAC2 output data)

20.5.14 DAC 状态寄存器 (DAC_STS)

域	简称	复位值	类型	功能
位 31: 30	保留	0x0	resd	请保持默认值。
位 29	D2DMAUDRF	0x0	w1c	DAC2 的 DMA 传输下溢标志 (DAC2 DMA transfer underrun flag) 0: DAC2 未发生 DMA 传输下溢。 1: DAC2 发生 DMA 传输下溢。 注: 软件写 '1' 清除该标志。
位 28: 14	保留	0x0000	resd	请保持默认值
位 13	D1DMAUDRF	0x0	w1c	DAC1 的 DMA 传输下溢标志 (DAC1 DMA transfer underrun flag) 0: DAC1 未发生 DMA 传输下溢。 1: DAC1 发生 DMA 传输下溢。 注: 软件写 '1' 清除该标志。
位 12: 0	保留	0x0000	resd	请保持默认值

21 控制器区域网络 (CAN)

AT32F456 和 AT32F457 额外支持 CANFD 协议，AT32F455 则不支持。

21.1 概述

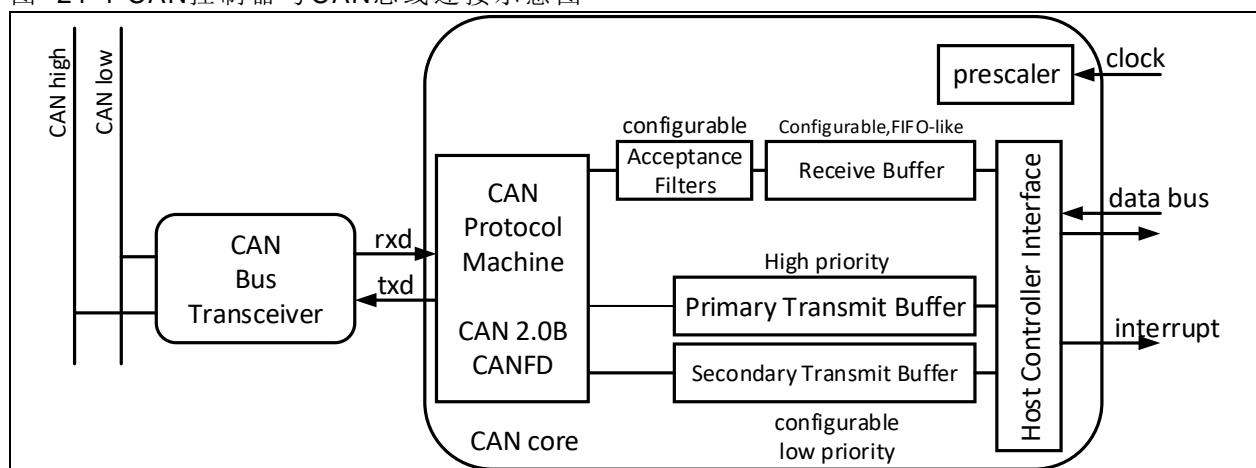
21.1.1 CAN控制器内核

作为串行通信控制器，CAN 控制器内核是基于 CAN 协议实现串行通信功能，并且满足 CAN 规范的所有约束性要求。

- Classic CAN2.0B
- CANFD (ISO 11898-1:2015)

CAN 协议定义了网络节点之间数据帧的传输（即通信对象），以及错误处理的管理机制。CAN 控制器可以实现在各种部件之间建立经济可靠的连接，无需主机控制器参与处理 CAN 协议。对于微控制器而言，CAN IP 核相当于一个内存映射 IO 设备。CPU 访问 CAN 控制器，控制双线 CAN 总线实现帧数据的收发。CAN IP 与 CAN 总线的连接方式如下图所示。

图 21-1 CAN控制器与CAN总线连接示意图



21.1.2 CAN控制器协议

CAN 控制器以数据帧的方式进行通信，所有错误帧和过载帧均由 CAN 控制器自动处理。数据帧用于传输主机应用程序的数据，几种格式的数据帧如图 21-2 和图 21-3 所示。Classic CAN2.0B 的一个数据帧可传输数据高达 8 字节，CANFD 可传输数据高达 64 字节。

帧标识符用于数据寻址，在 CAN 网络中，同时只有一个节点发送带有某个标识符的帧数据，其它节点处于接收状态。主机控制器必须判断其收到的帧是否符合需求，为了减少主机控制器负载，CAN 节点可以使用接收过滤器。过滤器将接收到的帧标识符与用户定义的标识符进行比较，只有通过了接收过滤器的帧才会存放至接收缓存中，并通知主机控制器。

CAN 帧数据的标识符用于总线仲裁，当优先级低的标识符的帧与优先级较高的标识符的帧同时发送时，CAN 控制器将停止发送优先级低的标识符的帧，之后会在总线空闲时自动重新开始发送被中断的帧。

CAN2.0B 定义的数据比特率最高为 1Mbit/s. 对于 CANFD 而言，理论上是没有限制的，但实际上会受到所使用的收发器的电气特性和总线结构的影响。CANFD 的比特率是可以切换的，如果开启了 CANFD 比特率切换功能，那么帧的数据负载将以更高速度传输，而帧头将以较低速度传输。

21.1.3 Classic CAN2.0B 以及 CANFD

CAN 协议历经多年的发展与完善，最早的 CAN 协议为 Classic CAN2.0B，之后进版为 CANFD (ISO11898-1:2015)，CAN 控制器可以接收和发送所有的 CANFD 以及 Classic CAN2.0B 帧，另外，CAN 控制器可以向上兼容。

CAN 控制器可以动态选择当前帧数据是作为 Classic CAN2.0B 帧、CANFD 帧发送，同样接收缓存的帧数据相应位会显示所接收到的帧数据是 Classic CAN2.0B 帧，还是 CANFD 帧。Classic CAN2.0B 帧类型见图 21-3 所示，CANFD 帧类型见图 21-2 所示。

图 21-2 CANFD帧类型

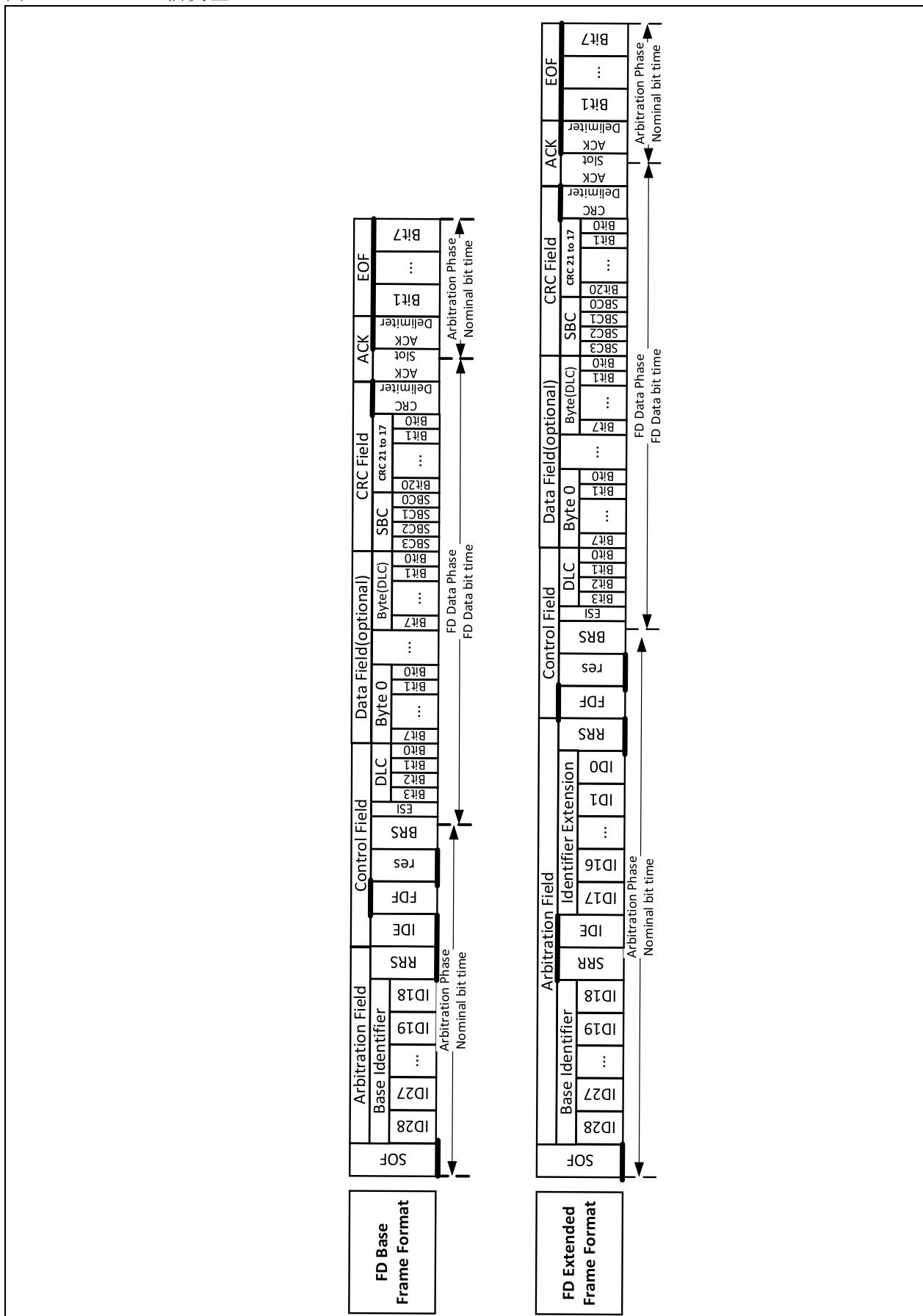
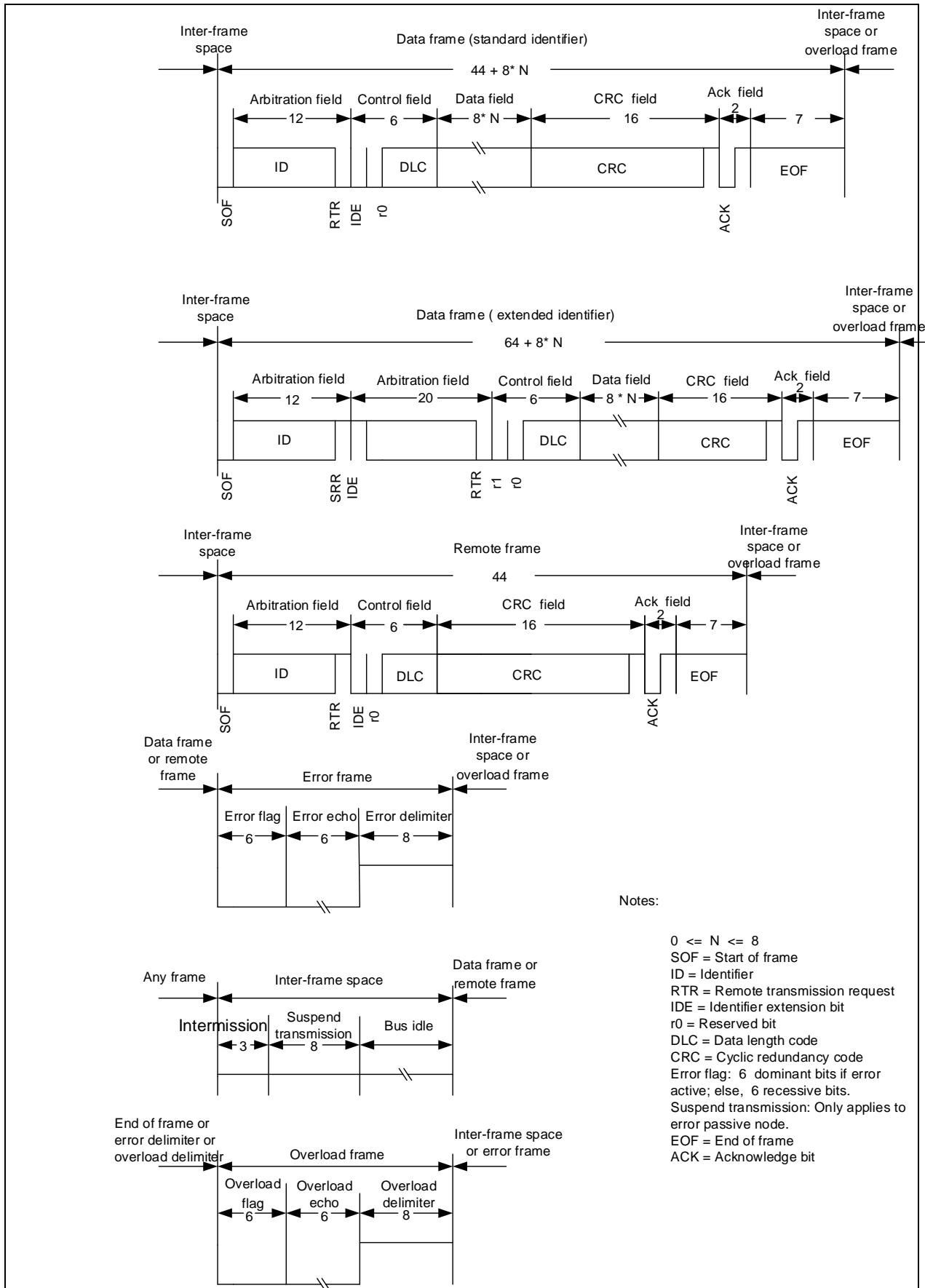


图 21-3 CAN2.0帧类型



21.1.4 向上兼容性和协议例外事件

CAN 规范包含用于协议扩展的保留位，包括在 CAN2.0B 基础上构建 CANFD，如不使用，这些保留位将以低电平(显性)传送，但 CAN2.0B 规范定义了一个保留位高电平(隐性)的行为，以接受此规则并继续帧处理。因此，如果采用此行为的 CAN2.0B 节点接收到一个 CANFD 帧(与 CAN2.0B 帧格式不同)，则会导致 CAN2.0B 节点产生帧错误，从而破坏该 CANFD 帧，这种行为称作“CAN FD intolerant”。

为了向上兼容新的协议规范，节点检测到保留位高时，将发生“协议例外事件”。这一点适用于 CAN2.0B、CANFD 节点，接收器不会对协议例外事件做出任何反应，而是忽略当前帧，不产生 ACK，等待总线空闲然后才发送或接收下一帧。对于 CAN2.0B 节点，这种情况称为“CAN FD tolerant”，允许在一个网络中同时存在 CAN2.0B、CANFD 帧。

旧版 CAN 2.0B 符合性测试会检查“CAN FD intolerant”行为，但建议采用新的“CAN FD tolerant”行为或协议例外事件，以向上兼容任何新版本的 CAN 协议。

目前我们仅会支持新的“CAN FD tolerant”行为或协议例外事件。

21.1.5 时间触发

基于 ISO11898-4 协议，CAN 控制器可以用于时间触发 CAN 通信 (TTCAN)，CAN 控制器提供部分硬件支持，在此模式下，需要主机软件实时交互。

TTCAN 的基本概念是通过定时器记录接收帧数据的时间戳和触发帧数据发送，CAN 网络中存在一个时间主机，时间主机用于发送参考报文，周期时间开始于参考报文，两个参考报文之间为一个基本周期，在基本周期内，报文可以在时间窗口内被发送。TTCAN 系统管理者定义脱机设置期间，每个时间窗的开始时间和持续时间。

总共有 3 种时间窗：

- 专用时间窗 (只允许一个节点传输一个带有特定 ID 的帧)
- 自由时间窗 (保留用作网络系统扩展)
- 仲裁时间窗 (多个节点可以传输一个帧，此时引发仲裁)

当接收到一帧数据，该节点记录实际的周期时间作为时间戳，同时，CAN 控制器提供了一个硬件触发，用于在预先设定的周期内发送一个预先设定的帧。

CAN 控制器在接收时自动检测参考报文并开始周期时间，硬件定时器是以 CAN 位时间为基准的 16 位定时器，满足 ISO11898-4 Level1 协议，CAN 控制器可以用于时间主机。

除了硬件触发帧传输之外，CAN 控制器还提供了一个监视触发器用于监测参考报文的缺失。

所谓的部分硬件支持，它指的是主机控制器需要为每个时间窗预先设置节点操作。比如，主机需要定义下一个待传输的帧，并为之设置触发时间。

21.1.6 CiA 603时间戳

CAN 控制器包含 CAN 总线组织 (CiA) 在 CiA 603 协议中定义的一个最低 16 位的时间戳，CiA 603 时间戳独立于 TTCAN 运行。CiA 603 的基本概念是包含一个自由运行的计数器，该计数器根据时钟周期而非 CAN 位时间进行计数，计数器的精度至少 10us (16 位) 或 1us (32 位或更多)，时间戳在 CAN/CANFD 帧的 SOF 或者 EOF 位置被记录。

CiA 603 支持 AUTOSAR 时间戳和时间同步功能，对于 AUTOSAR 而言，CAN 网络中的一个节点就是一个时间主机，时间主机发送同步报文 (SYNC 报文)，SYNC 报文的时间戳被主机和所有的从机获取，从发出指令传输 SYNC 报文到 SYNC 报文实际传输之间存在时间差，该时间差是由主机以 FUP 报文的方式进行发送的。CiA 603 规定了读取和修改定时器的相关规则，CAN 控制器不包含该定时器，而使用外部定时器。CAN 控制器只包含时间戳运行机制，寄存器存储发送时间戳 (TTS)，缓存存储所有接收帧的接收时间戳。

21.2 特性

21.2.1 特征列表

- 支持CAN协议
 - CAN2.0B: 高达8字节有效数据
 - CANFD: 高达64字节有效载荷, 支持ISO 11898-1:2015或non-ISO Bosch
- 帧速率可自由配置
 - CAN2.0B定义的速率最高为1Mbit/s
 - CANFD受限于收发器和CAN控制器的时钟频率

注意: CAN 协议对时钟部分规定允许振荡器最大容差为 1%, 为避免通讯出现异常, 必须使用外部晶振作为 CAN 的时钟源。
- 可编程的波特率分频器(从1到1/32可选)
- 接收缓冲器(RB)为6级深度
 - 接收缓冲器操作机制类似FIFO
 - 接收到的“不正确”或“不能过滤”的帧不会覆盖已存储的帧
- 两个发送缓冲器
 - 主发送缓冲器(PTB)(一个帧槽)
 - 额外3级深度次发送缓冲器(STB), 运行于FIFO或优先级模式
- 独立的可编程内部接收过滤器
 - 16个接收过滤器
- 扩展特性
 - 重新仲裁和重新发送的限制(1到7次或者“无限”尝试)
 - 只听模式
 - 环回模式(包括内部和外部)
 - 收发器待机模式
- 扩展状态和错误报告
 - 传输帧的状态报告
 - 获取最近一次发生错误的类型和仲裁丢失位置
 - 可编程的错误警告限制次数
- 可设置中断源
- 时间戳
 - ISO 11898-4时间触发CAN, 部分硬件支持
 - CiA 603时间戳
- 兼容AUTOSAR
- 针对SAE J1939进行了优化

21.2.2 中断

CAN 控制器具有 4 个中断向量, 通过配置 CAN 控制器中断使能寄存器 (INTEN), 可以控制相应的中断开启或关闭。

图 21-4 CAN1发送中断的产生

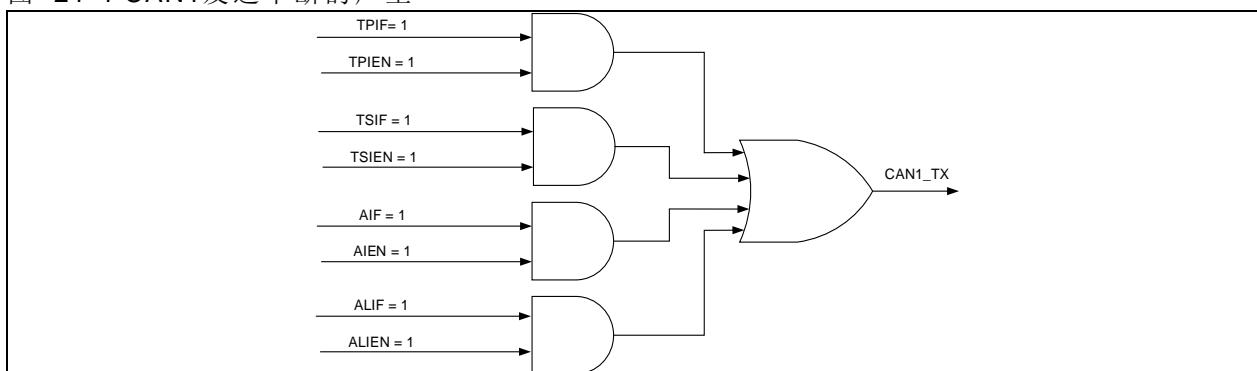


图 21-5 CAN1接收中断的产生

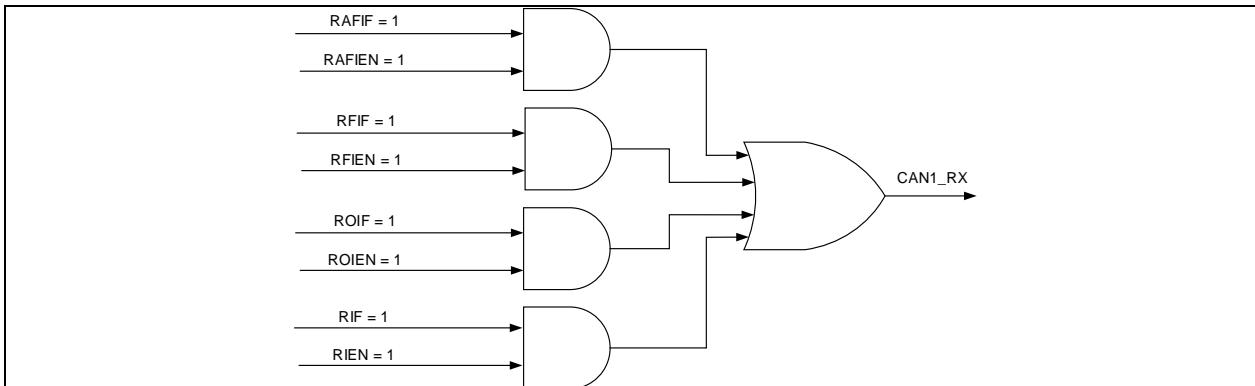


图 21-6 CAN1状态中断的产生

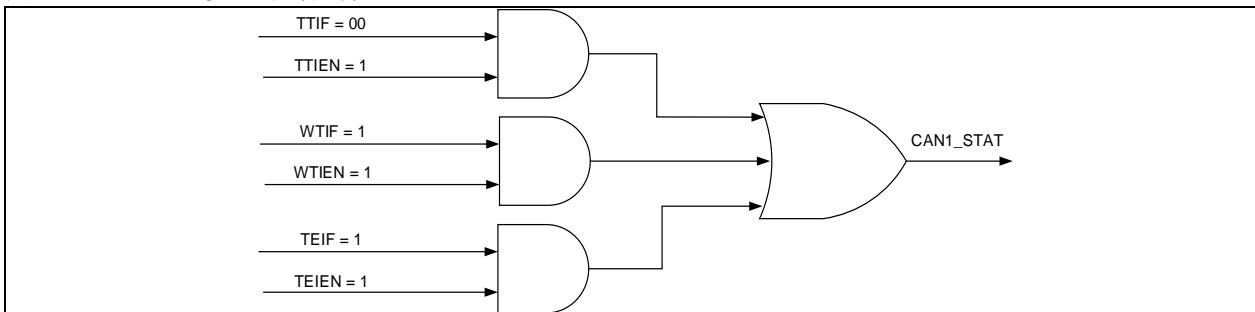
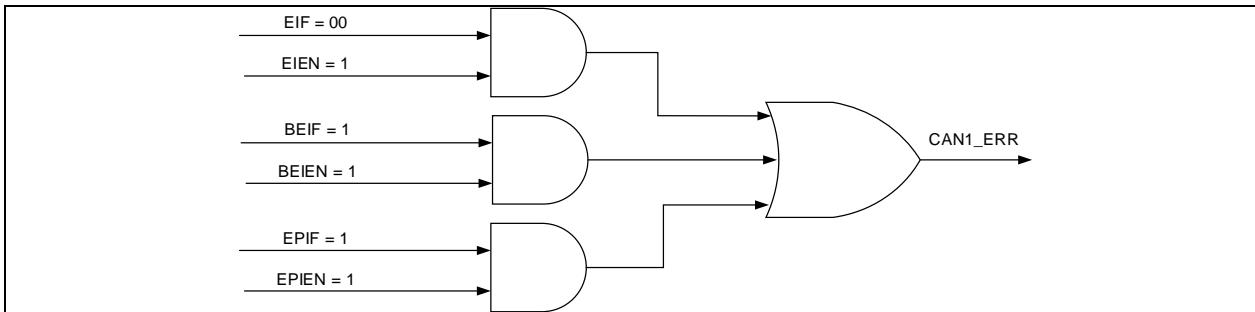


图 21-7 CAN1错误中断的产生



21.2.3 软件操作接口

表 21-1 显示了带有时间戳的逻辑链路控制帧 (LLC 帧) 的定义，主机应用与 CAN 控制器之间按照 LLC 帧格式进行数据交换，这个统一的定义用于往 TBUF 中存储待发送数据、从 RBUF 中读出接收到的帧数据以及配置接收过滤器（包含 ACFC 和 ACFM）。

对于接收过滤器，只有包含标识符、格式、类型的帧头是有意义的，应用于配置 ACFC 和 ACFM 寄存器，负载数据、CiA 603 和 TTCAN 时间戳对于 ACF 不可用。

接收时间戳 (CiA 603 和 TTCAN) 只对于 RBUF 有意义，对于 TBUF 不可用。

接收时间戳 (CiA 603 和 TTCAN) 的地址偏移位置 T 是动态变化的，T 值基于字对齐，通过使用 LLCSIZE 寄存器，应用可以方便的处理 LLC 帧。

表 21-1 LLC帧（逻辑链路控制帧）定义（包含时间戳）

地址	比特位								功能	
偏移值	7	6	5	4	3	2	1	0		
0	ID(7:0)								标识符 (ID)	
1	ID(15:8)									
2	ID(23:16)									
3	TTSEN	-			ID(28:24)					
4					DLC(3:0)				格式 (FMT)	
5	-									
6	-		RMF	-	BRS	FDF	IDE			

7	-	LBF	ESI	KOER(2:0)	
8	-	-	-	-	类型 (TYP)
9	-	-	-	-	
10	-	-	-	-	
11	HANDLE(7:0)	-	-	-	
12	-	-	-	-	保留
13	-	-	-	-	
14	-	-	-	-	
15	-	-	-	-	
16	D1(7:0)	-	-	-	负载数据
17	D2(7:0)	-	-	-	
...	...	-	-	-	
79	D64(7:0)	-	-	-	
T+0	RTS(7:0)	-	-	-	CiA 603 (接收)
...	...	-	-	-	
T+7	RTS(63:56)	-	-	-	
T+8	CYCLE_TIME(7:0)	-	-	-	
T+9	CYCLE_TIME(15:8)	-	-	-	TTCAN (接收)
T+10	-	-	-	-	
T+11	-	-	-	-	

表 21-2 LLC帧缩写定义

比特位	描述		
ID	帧标识符		
	接收: 有效	发送: 有效	过滤器: 有效
TTSEN	发送时间戳 (Time-Stamp) 使能 对于 CiA 603 时间戳, 可以选择是否获取发送时间戳 TTS 0: 当前帧的发送时间戳不更新 1: TTS 更新使能		
	接收: 无效	发送: 有效	过滤器: 无效
DLC	数据长度代码, DLC 定义了一帧数据的负载数据长度		
	接收: 有效	发送: 有效	过滤器: 有效
IDE	扩展标识符标志 0: 标准帧格式, ID(28:18) 1: 扩展帧格式, ID(28:0)		
	接收: 有效	发送: 有效	过滤器: 有效
FDF	CANFD 帧格式 FDF=0: CAN2.0 帧 (高达 8 字节负载) FDF=1: CANFD 帧 (高达 64 字节负载)		
	接收: 有效	发送: 有效	过滤器: 有效
BRS	CANFD 比特率切换使能 0: 整帧采用“慢速”比特率 (标称比特率) 1: CANFD 帧的数据场和 CRC 场切换至“快速”比特率 (数据比特率)		
	接收: 有效	发送: 有效	过滤器: 有效
RMF	远程帧 0: 数据帧		

	1: 远程帧			
	接收: 有效	发送: 有效	过滤器: 有效	
KOER	错误类型 对于接收帧, 当 RBALL 位=1 时, KOER 位变得有意义, 如果 RBALL 位=1, 则接收过滤器通常被关闭。			
	接收: 有效	发送: 无效	接收过滤器: 无效	
ESI	错误状态指示, 协议机自动将正确的 ESI 值嵌入到发送的帧数据中 0: CAN 节点处于主动错误状态 1: CAN 节点处于被动错误状态 ESI 只包含在 CANFD 帧数据中, CAN2.0B 帧该位固定为 0.			
	接收: 有效	发送: 无效	过滤器: 有效	
LBF	环回帧 对于接收帧而言, 如果启用了环回模式, 则 LBF 设置为 1, 并且 CAN 控制器接收自己发送的帧数据。如果 LBME 位=1, 而总线上其他节点也正在传输, LBF 可能很有用。			
	接收: 有效	发送: 无效	过滤器: 有效	
HANDLE	帧标识手柄 帧标识手柄应用于 TSTAT 中, 建议应用程序将软件计数值写入至 HANDLE.			
	接收: 无效	发送: 有效	过滤器: 无效	
Payload Data	帧的负载数据, Classic CAN2.0B 最多为 8 个字节, CANFD 最多为 64 个字节, 参考表 21-3 见具体细节, RBUF 未使用的负载数据填充无效数据, 直接忽略。			
	接收: 有效	发送: 有效	过滤器: N/A	
RTS	CiA 603 时间戳的接收时间戳 每接收一帧数据存储一次 RTS, 与 TTS 相对, RTS 与一个特定的接收帧相关 如果 TSEN 位=0, 则 RTS=0			
	接收: 有效	发送: N/A	过滤器: N/A	
CYCLE_TIME	周期时间 (TTCAN 时间戳) 只有接收帧才存储 CYCLE_TIME, 为帧 SOF 位的周期时间, 参考帧的周期时间恒为 0			
	接收: 有效	发送: N/A	过滤器: N/A	

表 21-3 DLC 的定义

DLC (二进制)	帧类型	负载数据字节
DLC(3:0)=0000 至 1000	Classic CAN2.0B 和 CANFD	0 到 8
DLC(3:0)=1001 至 1111	Classic CAN2.0B	8
DLC(3:0)=1001	CANFD	12
DLC(3:0)=1010	CANFD	16
DLC(3:0)=1011	CANFD	20
DLC(3:0)=1100	CANFD	24
DLC(3:0)=1101	CANFD	32
DLC(3:0)=1110	CANFD	48
DLC(3:0)=1111	CANFD	64

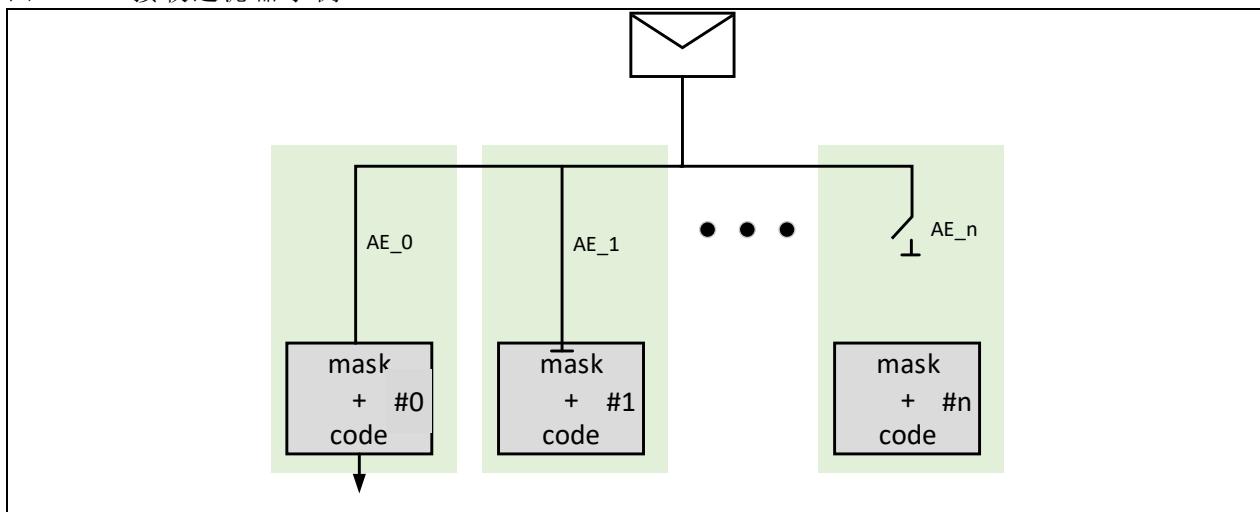
21.3 操作指南

21.3.1 接收过滤器

CAN 控制器使用接收过滤器以减轻主机控制器接收帧的负载，在接收过滤过程中，CAN 控制器会检查帧头。只有通过了过滤器的帧才会被接受，若过滤通过，通过接收过滤器的帧将存放在 RB 内，并置起 RIF 标志。如果过滤失败，则不会置起 RIF 标志，且 RB FIFO 指针也不会增加。未通过接收过滤器的帧会被丢弃，并被下一个帧所覆盖，未通过过滤的帧不会覆盖已接收并存储的有效帧数据。

接收过滤器仅适用于有效帧。(如果在传输过程中出现错误,CAN 控制器将根据 CAN 协议进行错误处理)。

图 21-8 接收过滤器示例



接收屏蔽定义的是需要比较哪些位，而接收代码定义的是相应值。若将验收屏蔽位设置为 0，则使能比较所选的接收代码位与接收帧的相应位。若将接收屏蔽位设置为 1，则会关闭接收过滤器检查，并直接接收该帧。（“屏蔽”意味着“不关心”）。

示例：ID(28)指所有 CAN 帧标识符的最高有效位，是最先发送的。假设 ACFM 的 ID(28)位为 0，其他位为 1，那么若要接收一个接收帧，则该接收帧的 ID(28)的值必须等于 ACFC 的 ID(28)，其他位在进行接收过滤时被忽略。

图 21-8 示例中，通过使用多个过滤器演示如何进行验收过滤，在本例中，通过 ACFCTRL 寄存器的 AE_0 和 AE_1 位使能过滤器 0 和过滤器 1，其他的过滤器处于关闭状态，所以不会接收任何帧。对于这两个已使能的过滤器而言，使用接收屏蔽位和接收代码来定义是否接收帧。在本例中，过滤器 0 接受该帧，而过滤器 1 不接收该帧。

注：设置 AE_x 位=0 将禁用该过滤器，并禁止接收帧。与此相反，ACFM 中的屏蔽位关闭了对应位的接收过滤检查，从而导致接收该帧。

在上电复位后，CAN 控制器配置为接收所有帧（通过设置 AE_0 位=1 使能过滤器 0，并且 ACFM 的所有位为 1，其他过滤器禁用，只有过滤器 0 定义了 ACFC/ACFM 复位值，其他过滤器没有定义复位值）。

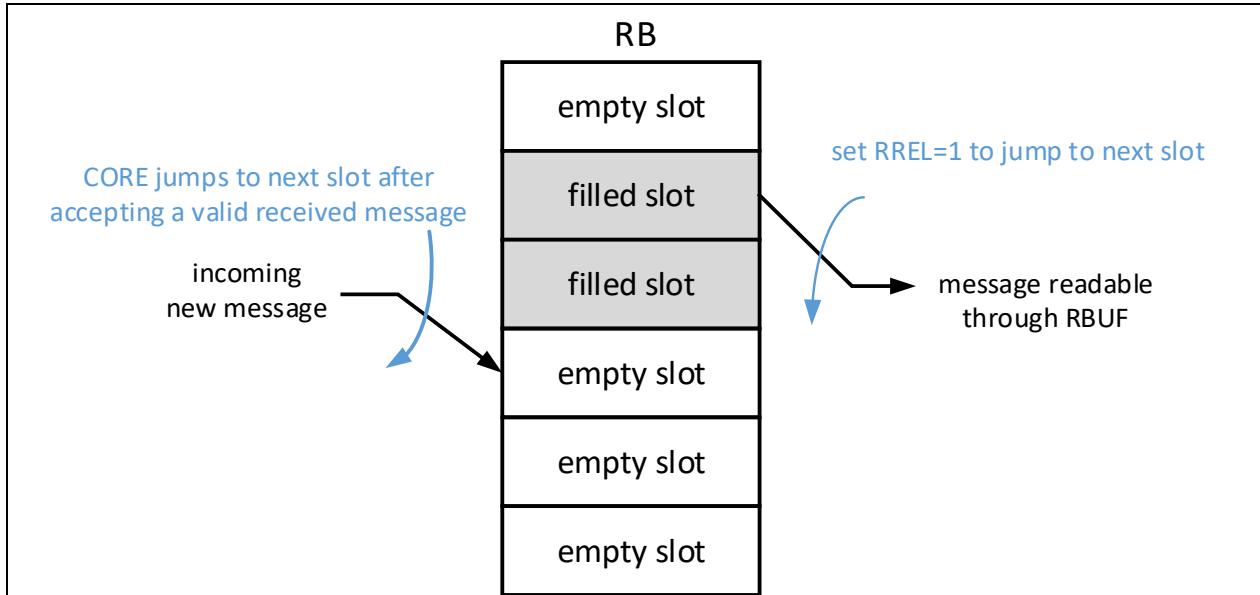
其它注意事项：

- 通过正确设置屏蔽位，一个单独的接收过滤器可以接受多组帧。CAN 控制器最多有 16 个验收过滤器，意味着最多可以定义 16 个过滤器组。
- 接收过滤器的结构是基于 LLC 帧定义进行设计的，目的是提供最灵活的验收过滤组定义。例如，可以通过将 BaseID/PriorityID 的最重要位（即 ID(28)）设置为某个值，即可接受各种类别的 CAN 帧（如 CAN2.0B, CANFD）。
- 有些位并不适用于所有 CAN 帧类型（CAN2.0B, CANFD），或者是强制为固定值。例如，对于 IDE 位，其掩码设置为 0，代码设置为 1，那么就意味着只接受 IDE=1 的帧。
- 主机应用程序负责正确配置接收过滤器，例如，取消屏蔽 LLC 帧的未使用位是没有意义的。CAN 控制器不提供错误配置保护机制（即 CAN 控制器无法防止此类配置错误），因此，除了一些必需的位之外，必须屏蔽所有位（包括“未使用的位”）。

21.3.2 帧接收

如下图所示，所接收到的数据是存放在 RB 内，RB 有 6 级深度，RB 的运作模式类似 FIFO。如果接收到的数据是有效的，且已通过过滤，则会置起 RIF 位=1，RSTAT 位的值取决于填充状态。当所填充的缓冲器的数量等于设定值 AFWL 位，将置起 RAFIF 标志。一旦所有的缓冲器都被填满，则将置位 RFIF 标志。

图 21-9 接收缓冲器 RB 结构图



RBUF 寄存器始终映射到 RB 中存储最早接收帧的槽空间，Classic CAN2.0B 帧的最大有效载荷为 8 字节，CANFD 帧为 64 字节。帧长度由 DLC 设置。RB 提供了槽空间足够大，可以存储最大帧长度。需要通过主机控制器设置 RREL 位跳转至下一个 RB 槽，槽中所有 RBUF 字节可以按照任何顺序读取。

如果 RB 已满，则收到的下一个帧将被临时储存直到它有效接收（第 6 个 EOF 位），之后，当 ROM 位=0 时，则最早接收的帧会被最新接收帧所覆盖；当 ROM 位=1 时，则最新接收帧被丢弃，在这两种情况下，则将置起 ROIF 标志。如果在新接收帧生效之前，主机控制器读取了最早接收的帧并置起 RREL 位，那么就不会丢失任何帧。

21.3.3 帧接收处理

如果没有接收过滤器，则 CAN 控制器会指示接收每个帧，并且要求主机控制器来决定是否对该帧进行寻址，这就给主机控制器带来相当大的负载。

除了通过接收过滤器降低负载之外，还可以关闭中断。基本的操作就是，如果 CAN 控制器已经收到一个有效帧的情况下，RIF 位设置为 1。为减少接收中断数量，可以使用 RAIE/RAFIF 位（RB 几乎满中断）或者 RFIE/RFIF 位（RB 满中断）代替 RIE/RIF 位（接收中断）。“几乎满限制”通过 AFWL 位进行设置。

RB 包含 6 个 RB 槽，读取 RB 的顺序如下：

- 通过 RBUF 寄存器读取 RB FIFO 中最早接收的帧。
- 通过设置 RREL 位=1 释放 RB 槽，该操作是用于选择下一个帧（即下一个 FIFO 槽）。RBUF 将自动更新。
- 重复这些操作直到 RSTAT 位指示 RB 为空。

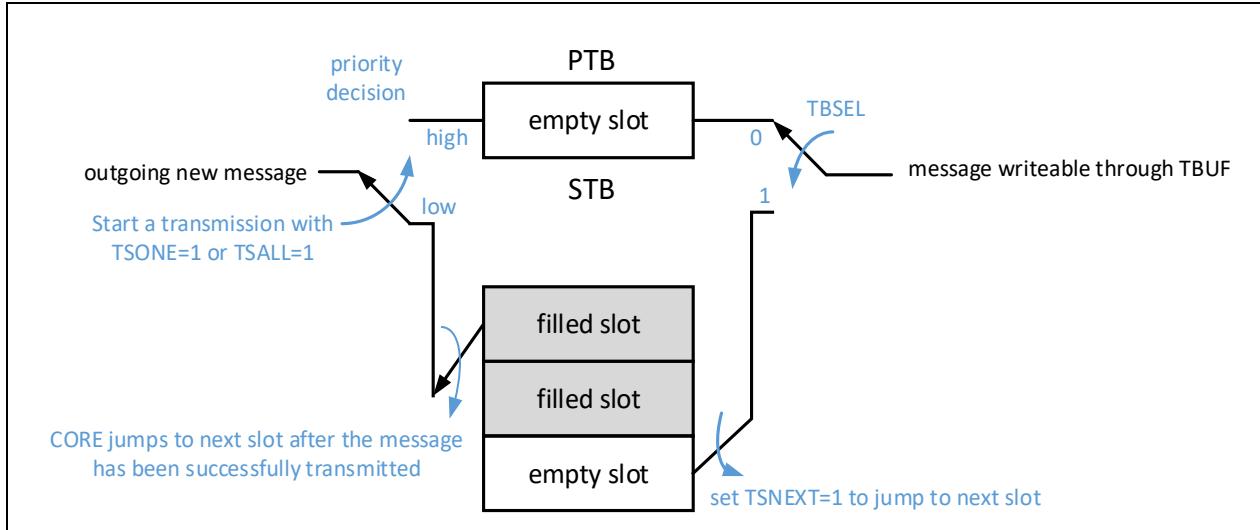
如果 RB FIFO 已满，并且新接收的帧被视为有效（第 6 个 EOF 位），那么会丢失一个帧数据（请参考 ROM 位），在此事件之前，不会丢失任何帧。因此，需要确保在 RB FIFO 已满且所选中断生成之后，主机控制器能够有足够的时问从 RB 读取至少一个帧。

21.3.4 帧发送处理

在启动帧发送之前，必须确保至少其中有一个发送缓冲器（PTB 或者 STB）加载了帧数据，如下图所示，TSSTAT 位指示 STB 填充状态（如果 TPE 位=1，则 PTB 写入被锁定）。TBUF 寄存器用于访问 PTB 和 STB，设置流程建议如下：

- 通过 TBSEL 位选择 PTB 或者 STB
- 向 TBUF 寄存器写入帧数据（所有 TBUF 字节可按照任何顺序写入）
- 对于 STB 而言，设置 TSNEXT 位=1 表示完成 STB 槽的加载
- 对于 PTB 而言，可重复写入，来覆盖之前写入的数据，不可配置 TSNEXT 位=1

图 21-10 PTB 和 STB 在 FIFO 模式下的结构图



CAN2.0B 帧的最大有效载荷的长度为 8 字节，CANFD 为 64 字节，DLC 位用于定义每个帧的长度。对于 CAN2.0B 远程帧（RMF 位），DLC 是无效的，因为 classic CAN 远程帧的数据长度始终是 0 字节。

当使用 PTB 时，应该通过设置 TPE 位启动传输。如果需要使用 STB，则必须设置 TSONE 位启动单个帧，或者设置 TSALL 位启动发送所有帧。

PTB 的优先级始终高于 STB，如果这两个发送缓冲器同时收到发送指令，则始终先发送 PTB 帧，而不受帧标识符的影响。如果 STB 发送已经被激活，那么该发送会继续完成，之后在下一个可发送位置（下一个帧间隙），开始 PTB 的发送。在完成和中止 PTB 发送之后，CAN 控制器回头处理 STB 中挂起未完成的帧。

在完成发送之后，会置起以下传输中断：

- 对于 PTB 来说：将置起 TPIF 标志
- 对于选择了 TSONE 位的 STB 来说，如果已经传输完单个帧，将置起 TSIF 标志
- 对于选择了 TSALL 位的 STB 来说，如果所有帧已传输完成，将置起 TSIF 标志。换言之，如果 STB 为空，将置起 TSIF 标志。因此，在启动 TSALL 位发送之后，如果主机控制器又向 STB 写入一个帧，则该帧也会被发送，然后再置起 TSIF 标志。

21.3.5 中止帧传输

发送缓冲器中的帧因为优先级较低而无法发送时，会较长时间堵塞该缓冲器。为避免此种情况发生，主机控制器可以在传输还未启动之前，通过分别设置 TPA 和 TSA 位撤销传输请求。

TPA 和 TSA 位共用一个中断标志：AIF 标志，CAN 协议机只有在未向 CAN 总线发送任何信息时才会执行中止命令（也就是说，当前正在进行中的发送不会被中止）。因此，应遵循以下规则：

- 总线仲裁期间不会中止发送。
- 如果节点失去仲裁，则会在稍后执行中止发送命令。
- 如果节点赢得仲裁，则继续执行帧发送。
- 帧发送期间不会中止发送。
- 如果一帧数据发送成功，则会向主机控制器指示成功发送的信号。在这种情况下，不会发出中止信号，同时产生相应的中断和状态信息。

- 当因为某种错误导致帧发送失败时，错误计数器会累加，并执行中止发送动作。
- 如果**STB**中至少还有一帧数据，同时主机已命令发送所有帧数据（**TSALL**位=1），那么向主机指示发送完成和中止发送的信号都会被置起。

因为以上这些因素的影响，中止发送可能需要花费一些时间，时间的长短取决于**CAN**通讯速度和帧长度。因此，如果中止发送被执行，将会导致以下情况发生：

- **TPA**位释放**PTB**，导致**TPE**位=0。在释放**PTB**之后，帧数据仍然存放在**PTB**内。
- **TSA**位释放**STB**的单个帧槽或者所有帧槽，具体情况取决于是选择了**TSONE**还是**TSALL**位来启动帧发送。**TSSTAT**位会进行相应的更新，释放**STB**帧将导致该帧被丢弃（原因是主机无法访问该帧）。

另外，不建议同时设置**TPA**和**TSA**位，如果主机控制器同时设置**TPA**和**TSA**位，将置起**AIF**标志，并且**PTB**和**STB**发送会被中止。已经开始发送的帧会在执行中止命令之前继续完成发送，之后会发出成功发送的信号。因此，以下中断标志可能会置起（在使能的情况下）：

- **AIF**（一旦中止**PTB**和**STB**传输，即会置起该标志）
- **TPIF+AIF**
- **TSIF+AIF**
- **TPIF+TSIF**（很少见，只有在主机没有及时处理**TPIF**的情况下，才会发生）
- **TPIF+TSIF+AIF**（很少见，只有当主机没有及时处理**TPIF**和**TSIF**的情况下，才会发生）

若需清除整个**STB**，则需要同时置起**TSALL**和**TSA**位。主机可以使用**ALIF/ALIE**位来检测某个帧是否因为失去仲裁而导致长时间没有发送。

21.3.6 STB满

在向**STB**写入帧之后，如果置起**TSNEXT**位=1，即表示缓冲器槽满，并跳转到下一个可用的帧槽。在完成此操作后，**CAN**控制器自动将**TSNEXT**位复位为0。

如果最后一个帧槽已满，即所有帧槽都被占用，此时**TSNEXT**位会保持置位状态直到有一个新的空闲槽出现，当**TSNEXT**位=1时，**CAN**控制器将禁止写**TBUF**。

当一个槽空间空闲时，**CAN**控制器自动复位**TSNEXT**位为0，如果**STB**内的一帧数据被成功发送，或者主机请求终止发送（**TSA**位=1），或者发送已经达到**RETLIM**或**REALIM**位设定的限值，那么一个槽空间变为空。如果**TSALL**位发送被终止，那么**TSNEXT**位被复位，另外整个**STB**被标记为空。

21.3.7 错误处理

一方面，**CAN**控制器会自动执行错误处理，这就意味着在通常情况下，主机控制器无须关注错误问题，包括自动重新发送帧数据和自动侦测接收帧错误。另一方面，应主机请求，**CAN**控制器可以选择性地提供关于错误的具体信息，并通过中断向主机发出错误信号，该特性可以实现主机运行类似**CAN**总线检测器的应用程序。

每个**CAN**节点有3种错误处理状态：

- 主动错误：节点在监测到错误时会自动发送主动错误帧
- 被动错误：节点在监测到错误时发送被动错误帧，意味着节点不会向总线发送显性位，而是期望其他**CAN**节点发送错误帧。处于被动错误状态的节点可以发送帧数据，但是如果这些节点在之前的帧传输中已经处于发送状态，那么需要在帧间隔之后挂起8个位，才开始新的一帧数据的发送。
- 总线关闭：如果出现太多错误，节点会进入“总线关闭”状态，不再影响总线。

为处理这3种错误状态，每个**CAN**节点配备了2个错误计数器，即发送错误计数器和接收错误计数器（通过**TECNT**和**RECNT**位进行读取），这两个计数器会根据**CAN**协议进行递增和递减。一旦达到**CAN**协议定义的计数器值后，节点会进入相应的错误状态，请参考表21-4。

如果监测到错误发生，错误计数器会递增。由某个节点可能引发的严重错误将会使计数器增加8，而由其他节点可能引发的错误将会使计数器增加1。有效的帧发送和帧接收会使计数器值递减，所有这些均由**CAN**协议进行定义，并由**CAN**控制器自动处理。

错误帧（错误标志产生）与数据帧不相同，错误帧代表的是至少 6 个连续显性位的显性脉冲，这违反了其他节点的位填充规则。如果一个 CAN 节点检测到此违规行为，该 CAN 节点会发送错误标志，该错误标志会产生错误帧，所有节点的错误帧会进行叠加。（因此，错误帧的持续时长为 6 到 12 个位时间）。主机控制器不能命令发送错误帧，这些均由 CAN 控制器自动处理。

如果 CAN 控制器接收命令发送一帧数据，则 CAN 控制器将尽可能地快的尝试重传直到该帧发送成功，或者该节点进入“总线离线”状态。如果 CAN 控制器接收到一帧数据且监测到错误，则该接收帧将被丢弃。因为自动发送的错误帧，发送者将重新发送该帧数据，RBUF 中的帧不会被包含错误的帧所覆盖，只有有效的接收帧才会导致 RBUF 溢出。

重新发送的次数受 TECNT 位所限制（如果节点进入“总线离线”状态），除此之外，还可以使用 RETLIM 和 REALIM 位进行限制。

表 21-4 CAN 节点错误状态

错误状态	TECNT	RECNT	EPASS	BUSOFF
主动错误	两者均小于 128		0	0
被动错误	其中之一大于 128		1	0
离线	大于 255	-	1	1

21.3.8 总线离线

CTRLSTAT 寄存器的 BUSOFF 位用于指示“总线离线”状态，当 CAN 节点的发送错误计数器超过 255 时，CAN 节点将自动进入“总线离线”状态。此时，该节点将不会再进行通讯，直到其重新恢复到错误主动状态。BUSOFF 位被软件配置为 1 时（总线从“总线离线”状态恢复），也能启动 EIF 中断。如果 CAN 节点由上电复位触发复位，或者当该节点接收到 128 次 11 个隐性位时（恢复序列），CAN 节点将恢复到错误主动状态。注意：在各个恢复序列之间，总线可能包含显性位。（注意：两个有效帧之间的最短时间足以被识别为恢复序列）。

在“总线离线”状态下，RECNT 和 TECNT 位计数保持不变。需要注意的是，在进入“总线离线”状态时，TECNT 位会翻转，因此可能会保持为较小的值。因此，建议在节点进入总线离线状态之前，使用 TECNT 位。在进入总线离线状态之后，使用 BUSOFF 标志。节点从“总线离线”状态恢复之后，RECNT 和 TECNT 位自动设置为 0。

如果某个帧被挂起等待发送，但是该 CAN 节点已进入总线离线状态，那么该帧将继续挂起。当节点从“总线离线”状态恢复至错误主动状态，则该挂起的帧将会被重新发送。若不需要重新发送，那么应该通过主机控制器中止帧发送。

21.3.9 扩展状态和错误报告

CAN 总线通讯期间可能会发生错误，下述功能支持错误侦测和错误分析，可用于扩展总线监测。

21.3.9.1 可编程的错误警告界限

RECNT 和 TECNT 计数器负责对收发器期间发生的错误进行计数，寄存器 ERR 包含了一个可编程的错误警告界限 EWL 位，使主机控制器能够灵活处理错误事件。错误警告的界限值以 8 为步长，范围是 8 到 128。

$$\text{错误计数限制} = (\text{EWL}+1)*8$$

如果 EIE 使能情况下，在下列情况下将置起 EIF 中断：

- RECNT 或者 TECNT 位越过错误警告界限，高于或者低于错误警告界限。
- BUSOFF 位硬件置 1 或者清除。

21.3.9.2 仲裁丢失捕获 (ALC)

仲裁丢失不属于错误，因此与 TECNT 和 RECNT 位无关，但是也是不希望发生的。

CAN 控制器能够检测发生仲裁丢失的仲裁位域中具体的位，同时将置起 ALIF 中断标志，用于指示发生仲裁丢失事件。在下一个仲裁丢失事件发生前，ALC 位的值会保持不变。

ALC 位的值的定义：帧从 SOF 位开始，然后发送 ID 的第一位，第一个 ID 位（即 ID(28)）的 ALC 位的值为 0，第二个 ID 位的 ALC 位的值为 1，以此类推，请参考图 21-2 和图 21-3 了解关于各类 CAN 帧的位顺序。

注意事项：

- 只允许在仲裁场进行仲裁，因此，ALC 位的最大值为 31，该位为扩展帧的 RTR/RRS 位。
- 如果标准远程帧与扩展帧发生仲裁，则扩展帧在 IDE 位将丢失仲裁，ALC 位的值为 12。发送标准远程帧的节点不会指示已经发生仲裁，原因是该节点已赢得仲裁。
- 仲裁场外不可能出现仲裁丢失的情况，因为这种情况将被视为位错误。

21.3.9.3 错误类型 (KOER)

CAN 控制器能够识别 CAN 总线上的错误，并将最近一次错误事件存储在 KOER 位，同时监测到 CAN 总线错误时会置起 BEIF 标志。新的错误事件会覆盖 KOER 位之前存放的值，因此，主机控制器必须快速响应错误事件。

每次出现新的错误事件时 KOER 位会随之更新。因此在成功接收或者发送帧时，KOER 位保持不变，这为延迟错误调查提供了机会。当 RBALL 位=1 时，错误帧也会存放在 RB 内，另外 LLC 帧也包含了 KOER 位的信息。

当 RBALL 位=1 时，出错的数据帧也会存放在 RB 内。另外，LLC 帧也包含了 KOER 位的信息。

如果 ROP 位=1 错误事件将导致协议例外。出现这类错误时，BEIF 中断也会置起，同时更新 KOER 位，与之相对的是，向上兼容协议例外不会导致 BEIF 中断，也不会更新 KOER 位。

21.3.9.4 接收所有数据帧 (RBALL)

当 RBALL 位=1 时，所有接收帧（包含出错的数据帧）都会存储在接收缓冲器 RB 内，在环回模式下也是如此，并且还会禁用接收过滤器，接收缓冲器 RB 只存储数据帧，错误帧或者过载帧不存储。

如果 CiA 603 时间戳被使能 (TSEN 位=1) 且时间戳位配置为 EOF 位 (TSPOS 位=1) 时，那么在发生错误的情况下，在错误帧开始处获取时间戳。

大多数错误只发生在节点作为发送器时，在这种情况下，如果启动了环回模式，则数据帧只存储在 RBUF 中。根据错误类型，存储在 RBUF 槽内的帧数据可能部分是有效的，而其它部分是未知的。如下表列出了各种可能的情况：

表 21-5 RBALL 和 KOER 位

KOER	节点状态	描述
无错误	所有	成功接收
位错误	接收器	只发生在应答场，所有存储的数据均有效
	发送器	负载数据始终无效，包含 ID 的帧头信息可能有效。在仲裁场，检测到错误的位属于仲裁过程，所以不属于位错误。但是如果在仲裁场的填充位检测到错误，则归属于位错误，在此情况下，帧头信息无效，但如果该帧头信息与预期的帧头信息相匹配，则需要使用该帧头信息做进一步判断。
格式错误	所有	只包含数据帧中的格式错误，包括 CRC 界定符、ACK 界定符、以及 EOF 位，所有存储的数据均有效。
填充错误	接收器	填充错误的位置未知，所有存储的数据均有效。
	发送器	仅发生在仲裁场，所有存储的数据均有效。
应答错误	接收器	仅发生在节点处于 LOM 模式时，所有存储的数据均有效。
	发送器	仅发生在不含 self-ACK 的环回模式，所有存储的数据均有效。
CRC 错误	接收器	所有存储的数据均有效。

21.3.9.5 传输状态 (TSTAT_1和TSTAT_2)

CAN 控制器包含各种状态和中断位，这些状态和中断位用于跟踪帧的传输状态，例如 TPIF、ALIF、AIF 以及 BEIF 位等，除此之外，发送状态寄存器 TSTAT_1 和 TSTAT_2 位也提供了详细的相关信息。

TSTAT_1 位记录的是当前发送帧的状态信息，所以，TSTAT_1 位中的内容是意失性的。一旦一帧数据成功完成发送，或者因重新仲裁、达到重新发送的限值而停止发送(详见 RETLIM 和 REALIM 寄存器定义)，那么这些信息会存放在 TSTAT_2 位内，换言之，当一帧数据的传输达到其最终状态时，TSTAT_2 位会相应更新。

发送状态包含以下信息：

- 用于标识帧的句柄
- 句柄所指向帧的发送状态
- 句柄所指向的相关时间戳

TTS 寄存器提供了时间戳信息，只有当 LLC 帧中的 TTSEN 位=1 时，才会更新时间戳(见表 21-1)，TTS 寄存器只与 TSTAT_2 位有关。

TSTAT_1 和 TSTAT_2 位根据表 21-6 所述帧传输状态进行编码。

HANDLE 用于标识帧，主机应用程序选择标识符，其包含在 LLC 帧内(表 21-2)，建议使用软件计数器的计数值作为 HANDLE，该 HANDLE 会在每发送一帧新的数据时递增，计数值可能发生翻转。

表 21-6 TSTAT 状态编码

值	标签	描述
000	IDLE	没有正在进行的传输
001	ONGOING	发送正在进行，且无错误
010	LOST_ARBITRATION	仲裁丢失，根据 REALIM 配置重新进行仲裁
011	TRANSMITTED	成功完成发送
100	ABORTED	发送中止 (TPA、TSA)
101	DISTURBED	发送错误，根据 RETLIM 配置重新进行发送
110	REJECTED	在 LLC 帧中，帧格式配置错误
111	-	保留

存在一些可能的情况会导致 TSTAT_1 和 TSTAT_2 位两者的更新，表 21-7 介绍了这些情况的详细信息。

表 21-7 TSTAT 状态事件

事件	TSTAT_1	TSTAT_2
电源上电复位或 RESET 有效	IDLE	IDLE
开始发送 (CAN 协议机从 TBUF 中预取数据)	ONGOING	-unchanged-
成功完成一次发送	IDLE	TRANSMITTED
发送中止 (TPA 或 TSA)	IDLE	ABORTED
仲裁丢失，没有达到 REALIM 配置的重新仲裁限制次数	LOST_ARBITRATION	unchanged
仲裁丢失，达到 REALIM 配置的重新仲裁限制次数	IDLE	LOST_ARBITRATION
发送错误，没有达到 RETLIM 配置的重新发送限制次数	DISTURBED	-unchanged-
发送错误，达到 RETLIM 配置的重新发送限制次数	IDLE	DISTURBED
总线离线恢复	ONGOING	-unchanged-
在发送的起始检测到 LLC 帧的错误配置	IDLE	REJECTED

除了发送中止事件，其它导致 TSTAT_1 或 TSTAT_2 位更新的事件都是一步一步进行的，所以主机应用程序可以跟踪其状态。与之相对的是，发送中止命令则是随时可以执行的。除此之外，一些较复杂的发送中止情况是无法通过 TSTAT_1 和 TSTAT_2 位进行正确指示标识的。

- 如果 STB 内不止一帧数据，且选择了 TSALL 传输模式，同时执行了 TSA 指令，那么指令发送会被中止，而 TSTAT_2 位却只能反映一个帧数据的状态。
- 如果同时是能了 PTB 和 STB，当前正在进行 PTB 的发送和 STB 发送中止，那么可能会同时发生 PTB 发送成功事件和 STB 发送中止事件，这会导致连续两次更改 TSTAT_2 位的内容，导致主机应用程序无法跟踪。
- 如果使用的是 STB，且选择 TSALL 传输模式，TSA 位在前一帧已经发送成功之后，而下一帧还没启动之前短暂置起，这会导致连续两次更改 TSTAT_2 位的内容，导致主机应用程序无法跟踪。

因此，只有在下列条件都满足时，TSTAT_2 位才会置起 ABORTED 状态：

- 在 PTB 传输时置位 TPA 位，或者在 STB 传输时置位 TSA 位。
- TSTAT_1 位标识当前正在进行的发送处于 ONGOING、LOST_ARBITRATION 或者 DISTURBED 状态。

21.3.9.6 帧拒绝发送

如果一帧数据被指定发送，同时该帧数据的帧格式配置错误，则这帧数据将被拒绝，不会予以发送。帧拒绝将会导致 TPIF 或 TSIF 中断（如果相应的中断使能位已置起的话，具体是 TPIF 还是 TSIF 位取决于所使用的是 PTB 还是 STB），并根据 21.3.9.5 章节所述，标识传输状态位“REJECTED”。帧格式配置错误的原因在于上层应用出现严重错误，在正常操作中不应该发生，只会发生在应用开发阶段出现错误时。

21.3.10 比特率切换和收发模式转换

经典 CAN 帧是以一个恒定的比特率（“慢速”标称比特率）进行传输，当 BRS=1 时，CANFD 帧可以切换帧内的比特率，这些帧以“慢速”标称比特率开始传输，然后在 BRS 位采样点切换到“快速”比特率，在 CRC 界定场采样点或者出现错误时返回到原来的“慢速”标称比特率。

要实现更高的比特率取决于总线拓扑结构和收发器的电器特性，对于 CANFD 帧而言，对应的收发器具有改善信号的能力（CANFD SIC 收发器），因此能够达到更高的比特率。

经典 CAN 和 CANFD 收发器采用牵引电阻的概念，逻辑 1 是由牵引电阻引起的，因此称为“隐性”，而逻辑 0 是节点主动驱动的，因此称作“显性”，使用牵引电阻会导致总线的不对称，从而限制最高比特率。

21.3.11 扩展特性

21.3.11.1 限制重传和重新仲裁

因为反复尝试重新传输或重新仲裁会导致严重延迟，增加总线负载，所以有些时候并不需要自动开启重新传输或重新仲裁。通过 RETLIM 和 REALIM 位可以限制这两种行为。这两种限制行为是互相独立的。

重新传输发生在错误之后，而重新仲裁发生在仲裁丢失之后。如果节点进入总线离线状态，则会停止重新传输。

在发生相应事件后，重新传输和重新仲裁的计数器都将递增，并且在开启一次新的传输时（配置 TPE、TSONE 或者 TSALL 位）都将复位至 0，或者配置 TSALL 位开启传输几帧数据并且其中任意一帧数据发送成功，或帧数据没有发送成功但重新发送/重新仲裁达到限定次数时，重新传输和重新仲裁的计数器都将复位至 0。选择 PTB 或者 STB 作为传输数据的来源不会对重新传输和重新仲裁的计数器产生影响，只有错误事件、仲裁丢失或者成功发送才会影响到这两个计数器。下面举例逐步解释这两个计数器的行为逻辑：

- 出现错误导致 STB 传输失败 → 重新传输计数器递增
- STB 传输丢失仲裁 → 重新仲裁计数器递增
- PTB 成功传输 → 两个计数器均复位（STB 传输时两个计数器从 0 开始递增）

如果立即实现了成功传输，这与正常情况没有区别。但是如果传输不成功，则会发生下列行为：

- TPIF 标志会置起，相应的传输缓冲槽会被清除。
- 在报错的情况下，KOER 位和错误计数器会更新，BEIF 标志会置志，并且其他的错误中断

标志也会相应的置位。

因此，如果重新传输或重新仲裁受到限制，那么单纯 TPIF 标志无法指示该帧是否已经发送成功。因此，如果需要收到传输成功的反馈，则重新传输或者重新仲裁限制机制需要配合 BEIF 和 ALIF 标志一起使用。通过 TSTAT_1 和 TSTAT_2 位（查看 21.3.9.5）也可以帮助跟踪事件状态。

如果重新传输或重新仲裁受到限制，TSALL 位=1 置起，并且 STB 内包含不止一帧数据，则计数器会记录每帧数据的尝试次数。在达到限制次数之前，如果有任何一帧数据没有成功传输，则 CAN 控制器会跳到下一帧数据，并且在 STB 变空之后，CAN 控制器停止发送。在此情况下，用户就很难判断发生了什么状况，只有错误计数器 TECNT 位会指示当前状况。如果两个帧中有一个帧出错，主机就没法侦测到底哪一个帧被成功传输了，因此这种情况会导致评估变得很复杂。

限制重新传输和重新仲裁能够保证在时间间隙内进行操作以确保最大限度的报文延迟。

21.3.11.2 只听模式 (LOM)

LOM 能够在不影响总线的情况下监测 CAN 总线。

- LOM 与 ISO 11898-1:2015 定义的总线监控功能类似，但并不完全兼容。不同点在于 ACK，LOM 模式下，CAN 控制器依赖于总线其它的 CAN 节点发出的显性 ACK 信号。
- LOM 与 ISO 11898-1:2015 定义的受限操作模式类似，但并不完全相同，受限操作参考 21.3.11.3 章节。

LOM 模式下可以通过 KOER 位和 BEIF 标志检测总线错误。

在 LOM 模式下，CAN 控制器不能向总线写入显性位（即无法发出主动错误标志、过载帧以及无法产生有效应答 ACK），该行为参考如下规则：

- 如果 LOM 位=1，则协议机类似处于受限操作模式（Restricted Operation Mode），即每个错误会导致协议例外事件（Protocol Exception Event）。
- 如果 LOM 位=1，则协议机不会产生显性 ACK。
- 错误计数器不会产生任何变化。

在 LOM 模式下，有关 ACK 的相关信息：

- 如果节点发送了一帧数据，那么只有当总线上连接了不止一个额外的节点，并且在额外节点生成 ACK 的情况下，才会在总线上生成可见的 ACK，之后，假如没有发生错误，所有节点会接收该帧，否则，该帧数据因 ACK 错误无效。
- 如果发生 ACK 错误，则处于 LOM 模式的节点能够检测到该错误。

注：在传输正在进行时，不能开启 LOM 模式，如果此时开启 LOM 模式，则不会启动传输。

环回模式（外部 LBME）对于 LOM 模式的行为起到非常重要的作用，如果禁用了 LBME 模式，则 LOM 模式的行为如上所述，节点不能向总线写入任何显性位。如果使能 LBME 模式，则允许节点发送包含自应答（self-ACK）信号的帧，但节点不会通过 ACK 来响应从其他节点发送的帧，也不会发出主动错误标志和过载帧。总而言之，LOM 和 LBME 结合起来使用相当于“一个静默接收器，在特殊情况下具有发送功能”。

21.3.11.3 受限操作 (ROP)

在受限操作模式下，除了潜在的时间主机，CAN 节点无法发送帧数据，作为发送主机的节点可以发送时间参考报文以启动网络。

在受限操作模式下，CAN 节点能够接收帧数据，并在接收到有效帧数据时产生 ACK 响应，一旦发生错误或者总线过载情况，处于受限操作模式下（ROP 位=1）的 CAN 节点会将这些视为协议异常，并进入总线集成状态（Bus Intergration），在 ROP 位=1 的情况下，错误计数器保持不变。

注：当 CAN 总线正在进行传输时，无法激活 ROP 位。

21.3.11.4 环回模式（LBMI和LBME）

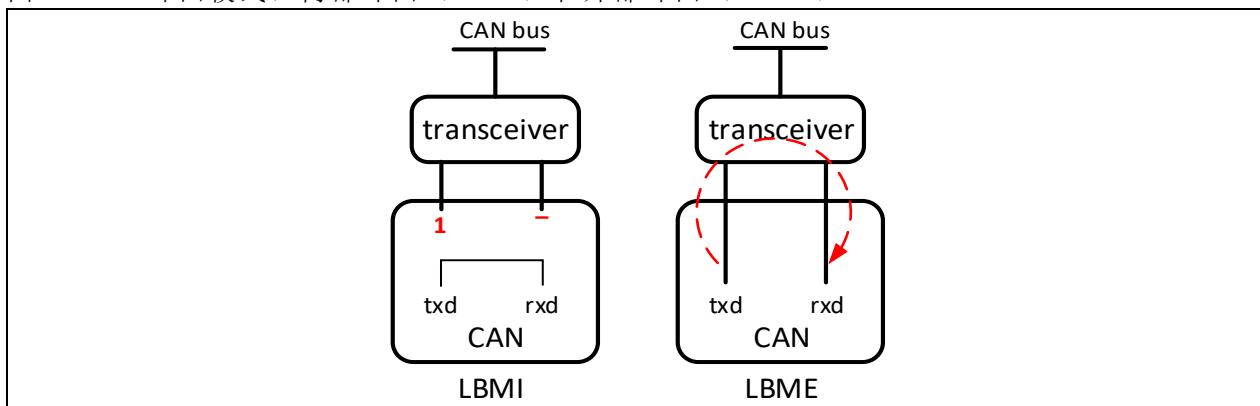
因 CAN 控制器支持两种环回模式：内部环回模式（LBMI）和外部环回模式（LBME），两种模式均可实现接收自己发送的帧数据，这对于自检测非常有用，具体参考图 21-11。

在 LBMI 模式下，CAN 控制器与 CAN 总线断开连接， $\langle\text{txd}\rangle$ 输出隐性电平，输出数据通过内部反馈给输入端，在 LBMI 模式中，节点会产生 self-ACK 以避免产生 ACK 错误。

在 LBME 模式下，CAN 控制器始终与收发器相连，在总线上可以看到节点发送的帧数据，CAN 控制器通过收发器接收自己发送的帧数据。在 LBME 模式下，当 SACK 位=0 时，节点不会产生 self-ACK；当 SACK 位=1 时，节点产生 self-ACK。因此，在 LBME 模式且 SACK 位=0 时，发送帧数据时可能会出现两种结果：

- 其他节点接收帧数据并产生 ACK，从而成功完成发送与接收。
- 没有其他节点与总线相连，导致 ACK 错误，因此建议 RETLIM 位=0，以避免重新传输而引起错误计数器递增。

图 21-11 环回模式：内部环回（LBMI）和外部环回（LBME）



环回模式下，CAN 控制器接收自己发送的帧数据，并将其放在 RBUF 中，置起相应的接收中断标志，当接收到自己发送的帧数据时会置起 LBF。

LBMI 模式可以用于芯片内部测试和软件测试，LBME 可以用于收发器测试，以及节点与收发器的连接测试。

注：LBMI 和 LBME 位不可与 TPE、TSONE 或 TSALL 位同时更改。LBME 可以与 LOM 模式配合使用。

21.3.11.5 收发器待机模式

通过 STBY 配置位可以驱动 $\langle\text{STB}\rangle$ 信号，用于激活收发器的待机模式，这种操作模式与 NXP TJA1049 收发器及其他具有类似功能的收发器是兼容的。

一旦激活待机模式，就不能再进行传输，也不能设置 TPE、TSONE 和 TSALL 位，另外 CAN 控制器不允许在发送过程中（TPE、TSONE 或者 TSALL 位置位时）设置 STBY 模式。

一旦 STBY 位置起，收发器进入低功耗模式。在此模式下，无法以全速接收帧，但是可以监测 CAN 总线的显性状态。如果显性状态在规定时间（即收发器数据特性表定义的时间内）内保持生效状态，那么收发器将拉低 $\langle\text{rxn}\rangle$ 信号。如果 $\langle\text{rxn}\rangle$ 变低，则 CAN 控制器将自动清零 STBY 位，从而关闭收发器的待机模式，并且不会干扰总线上其他主机（退出待机模式后，将收到来自其他节点发出的帧数据）。

收发器需要花一些时间从待机模式切换到活跃模式，所以无法成功接收到初始唤醒帧。因此，当前处于待机模式下的节点不会进行 ACK 响应，如果总线上没有 CAN 节点对唤醒帧进行 ACK 响应，则会导致唤醒帧发送节点产生 ACK 错误，然后该发送节点将自动重新发送该帧。在重新发送时，收发器将回归到活跃模式，CAN 控制器将接收到帧，并以 ACK 进行响应。

21.3.11.6 错误计数器复位

根据 CAN 规范，RECNT 计数接收错误，TECNT 计数发送错误。在发生太多发送错误之后，CAN 节点进入总线离线状态，寄存器 RESET 位不会改变错误计数器或者总线离线状态。CAN 规范制定了从总线离线状态恢复和错误计数器递减的规则，如果只是暂时的错误导致了问题，一个好的节点会自动的从总线离线状态恢复。为避免低级错误，Classic CAN2.0B 规范要求硬件具备上述自动行为，而无需主机控制器的交互操作。

CANFD 规范放宽了上述限制，该规范允许主机控制器对错误计数器的手动控制，但是这种使用方式需要格外小心，建议仅在调试过程中使用。

配置 BUSOFF 位为 1 将复位错误计数器，从而强制节点从离线状态恢复，同时置位 EIF 标志。

21.3.11.7 软件复位

通过将 CTRLSTAT 的 RESET 位置 1，可以启动软件复位。当 RESET 位=1 时，部分寄存器会进入复位状态，其他寄存器则只适用于 IP 的复位，但无论是软件复位还是 IP 的复位，所有的复位值始终一致。

表 21-8 软件复位

寄存器	是否复位	注释
ACFADR	否	-
ACFC	否	当 RESET 位=1 时，寄存器可写，否则写锁定。
ACFM	否	当 RESET 位=1 时，寄存器可写，否则写锁定。
AC_SEG_1	否	当 RESET 位=1 时，寄存器可写，否则写锁定。
AC_SEG_2	否	当 RESET 位=1 时，寄存器可写，否则写锁定。
AC_SJW	否	当 RESET 位=1 时，寄存器可写，否则写锁定。
AE_x	否	-
AFWL	否	-
AIF	是	-
ALC	是	-
ALIF	是	-
BEIF	是	-
BUSOFF	(否)	通过设置 BUSOFF 位=1 复位错误计数器也会复位 BUSOFF 位
EIF	否	-
EPASS	否	-
EPIF	是	-
EWARN	否	-
EWL	是	-
FD_ISO	否	当 RESET 位=1 时，寄存器可写，否则写锁定。
FD_SEG_1	否	当 RESET 位=1 时，寄存器可写，否则写锁定。
FD_SEG_2	否	当 RESET 位=1 时，寄存器可写，否则写锁定。
FD_SWJ	否	当 RESET 位=1 时，寄存器可写，否则写锁定。
FD_SSPOFF	否	当 RESET 位=1 时，寄存器可写，否则写锁定。
KOER	是	-
LBME	是	-
LBMI	是	-
LLCAOT	否	-
LLCFORMAT	否	-
LLCPBYTES	否	-
LOM	否	-
PRESC	否	当 RESET 位=1 时，寄存器可写，否则写锁定。
RAFIF	是	-
RBALL	是	-
RBUF	(是)	所有 RB 槽空间被标记为空，RBUF 存储值未知。
REALIM	是	-
RECNT	否	通过设置 BUSOFF 位=1 可以复位错误计数器。
REF_ID	否	-

REF_IDE	否	-
RETLIM	是	-
RFIF	是	-
RIF	是	-
ROIF	是	-
ROM	否	-
ROP	否	-
ROV	是	所有 RB 槽空间被标记为空。
RREL	是	-
RSTAT	是	-
SACK	是	-
STBY	否	-
TBE	是	-
TBF	是	-
TBPTR	否	-
TBSEL	是	TBUF 固定指向 PTB
TBUF	(是)	所有 STB 槽空间被标记为空, TBUF 指向 PTB。
TECNT	否	通过设置 BUSOFF 位可以复位错误计数器。
TEIF	是	-
TEW	否	-
TPA	是	-
TPE	是	-
TSA	是	-
TSALL	是	-
TSMODE	否	-
TSEN	否	-
TSPOS	否	-
TSNEXT	是	-
TSONE	是	-
TPIF	是	-
TSFF	是	所有 STB 槽空间被标记为空。
TSIF	是	-
TSSTAT	是	所有 STB 槽空间被标记为空。
TTEN	是	-
TTIF	是	-
TTPTR	否	-
TTS	否	-
TTTBM	否	-
TTYPE	否	-
TT_TRIG	否	-
TT_WTRIG	否	-
T_TPRES	否	-

21.4 时间触发CAN (TTCAN)

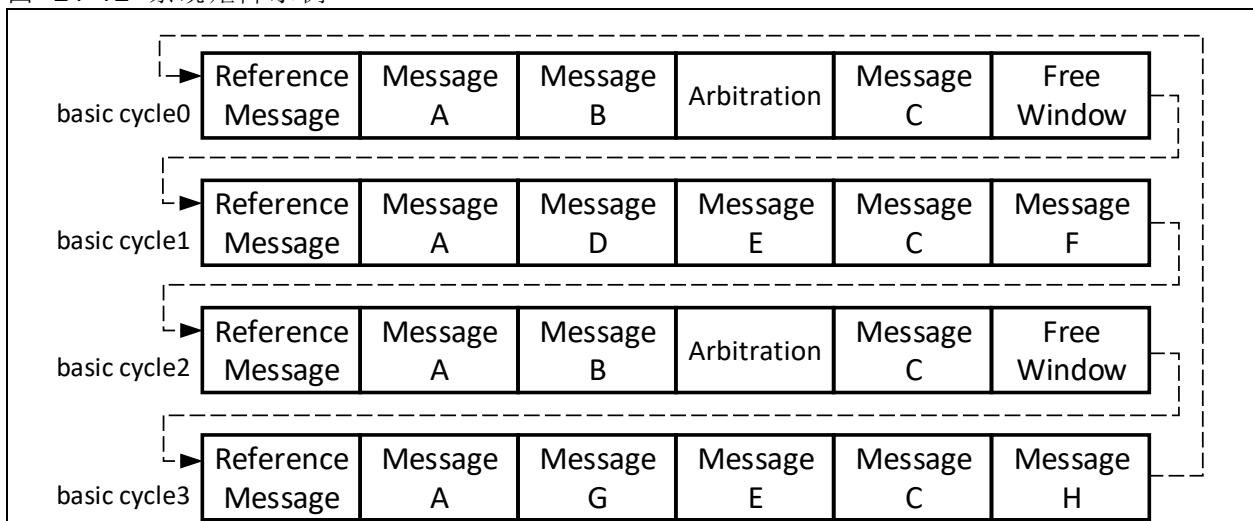
21.4.1 介绍

时间触发 CAN 是基于 ISO11898-4 协议的一种操作模式，所有帧只能在预先设定的时间窗内传输，总共有三种时间窗：

- 专用时间窗（只允许一个节点传输特定 ID 的帧）
- 自由时间窗（保留用作网络系统扩展）
- 仲裁时间窗（多个节点同时传输一帧数据，此时引发仲裁）

时间由 TTCAN 系统管理者离线设置，并按照图 21-12 所示的系统矩阵进行分配，横轴表示的是一个基本周期，开始于参考报文，参考报文由时间主机发送，其它报文由包括时间主机的任意节点发送。

图 21-12 系统矩阵示例



报文的 SOF 时间成为 Sync_Mark，参考报文的 Sync_Mark 称为 Ref_Mark，计时通过一个自由运行的 16 位计时器完成，计时器与 Ref_Mark 之间的时间差称为周期时间。换言之，每个基本周期是从 Ref_Mark 开始计算周期时间，周期时间作为时间戳存储在 RBUF 中。

TTCAN 系统管理者负责定义每个时间窗的长度，时间窗的长度设置要确保能够完整的传输一帧报文，因此，报文必须在单次传输模式下传输，存在一种例外情况：如果多个仲裁时间窗合并，那么可以使能重新传输，但是为了避免影响后续时间窗，需要尽早结束重传。

单次传输的含义：禁用重新传输和重新仲裁功能。

触发器在激活后会通过中断告知主机，然后主机需要为下一个时间窗配置一个触发器，这就要求主机进行实时响应。一个时间窗的时长指的是允许使用主机的时间。

如果需要将 CAN 控制器作为时间主机运行，那么参考报文需要放在 TBUF 槽（像其他的报文一样），然后在硬件触发器激活后进行传输。通过正确设置 REF_MSG_x 寄存器中参考报文的 REF_ID 和 REF_IDE 位，时间主机及所有时间从机将自动检测参考报文。一旦检测到参考报文，CAN 控制器将自动更新 Ref_Mark，启动周期时间。

除了用于报文传输的触发器之外，CAN 控制器还提供了窗口触发器。该窗口触发器用于检查距离上一次参考报文的时间是否过长。主机应该为周期性操作或者事件触发操作配置窗口触发器，在周期性操作中，每一个基本周期之后紧跟着下一个基本周期；在事件触发操作中，两个基本周期之间存在时间间隙，下一个周期开始于事件生成。如果窗口触发器使能，则会窗口产生中断。

除了支持 ISO 11898-4 之外，CAN 控制器还支持事件触发 CAN 通信与接收报文时间戳两种功能配合使用。通过设置寄存器位 TTTBM=0，可以使能该模式。在该模式下，CAN 控制器就类似于事件触发 CAN 通信，只是可以检测到参考报文且提供接收帧的时间戳。此外，该模式仅支持事件触发器和窗口触发

21.4.2 TTCAN模式下TBUF配置

事件 TBUF 的行为取决于寄存器 TTTBM 位，如果 TTTBM 位=1，那么每个 TBUF 槽空间均可由主机控制器寻址，该条件是使用传输触发的必备要求；如果 TTTBM 位=0，则仅支持时间戳和事件触发。

21.4.2.1 当TTTB M位=1时TBUF的行为

在 TTCAN 模式下当 TTTBM 位=1 时，STB 用作报文槽的队列，每个报文槽可由 TB PTR 位寻址。主机通过 TBF 和 TBE 这两个位将报文槽标记为“满”或“空”，已“满”的报文槽是写保护的。在 TTCAN 模式中，TBSEL 和 TSNEXT 两个位没有实际意义而被忽略。

通过设置 TB PTR 位=0 可以寻址 PTB，这使得 PTB 可以作为任何一个 STB 槽使用。实际上，PTB 在 TTCAN 模式中没有特殊性，它与 STB 相关联，因此，发送成功始终是 TSIF 位被标记。

在 TTCAN 模式中，TBUF 既没有 FIFO 模式也没有优先级决定模式。此外，只能选择一帧一帧的数据进行发送。

报文触发定义的是何时需要发送（即时间窗的开始）一帧报文，并通过指针 TT PTR 选择要发送的报文。如果触发事件发生，那么选择的报文随即开始发送。最后，触发中断被置位以便通知主机为下一步行动作准备。

所有的传输只能通过一次触发来启动，在 TTCAN 模式中，TPE, TSONE, TSALL 和 TPA 位固定为 0，并被忽略。

21.4.2.2 当TTTB M位=0时TBUF的行为

设置 TTTBM 位=0 提供了事件触发 CAN 通信和接收报文时间戳的一种组合，在此模式下，PTB 和 STB 的行为就类似于 TTEN 位=0 时的情况。PTB 的优先级始终高于 STB，而 STB 可以运行于 FIFO 模式(即 TSMODE 位=0) 或优先级决定模式 (即 TSMODE 位=1)。

21.4.3 TTCAN操作

上电之后，时间主机需按照 ISO 11898-4 的要求进行初始化，一个 CAN 网络中可能最多有 8 个时间主机，每个时间主机有其自身的参考报文 ID（即 ID 的最后 3 位），潜在的时间主机根据其优先级顺序发送参考报文，优先级较低的时间主机会稍晚发送参考报文。

如果 TTEN 位已置起，则 16 位定时器开始运行，如果检测到参考报文，或时间主机成功地发送了参考报文，则 CAN 控制器将会复制该报文的 Sync_Mark 至 Ref_Mark，同时将周期时间置 0。接收到报文后，会置位 RIF 位，成功发送报文会置位 TPIF 或 TSIF 位，之后主机需要为下一次的动作配置触发。

触发动作可以是一次接收触发，仅仅产生触发中断，该触发动作可以用于检测是否收到所需要的报文。这种触发器也可以用于其它应用，但具体情况取决于主机应用程序。

另一种触发器是发送触发器，此触发器通过 TT PTR 位指向 TBUF 槽，启动 TBUF 槽内的帧发送。如果 TBUF 槽标记为空，则不会启动帧传输，但是会置起中断标志。

如果主机应用程序需要，主机任务可以通过新报文来更新报文槽，比如，一个新的传感器值。随后，如果 TTCAN 需要传输这则报文，则由触发器进行激活。换言之，一个主机任务负责 TTCAN，另一个主机任务负责更新报文槽。因此这就要求一个报文槽专用于一帧报文（比如，槽空间 1 专用于温度传感器）。如果没有足够的 TBUF 槽空间可用，那么不同的报文可以共用槽空间。

TTCAN 主机应用程序需要跟踪系统矩阵，如果触发器被激活，那么主机应用程序需要准备下一次的传输。同时主机还需要处理基本周期，参考报文包括了周期数。

注：ISO 11898-4 协议中的大多数操作都要求单次传输。

21.4.4 TTCAN时序

CAN 控制器支持 ISO 11898-4 等级 1，内置一个 16 位定时器，该定时器以 CAN 位时间（由 PRESC, AC_SEG_1, AC_SEG_2 位定义）为基本时间单元，另外 T-PRESC 位定义了一个额外预分频器，当 TTEN 位=1 时，该定时器持续计数。

在报文的 SOF 位置，该定时器的值为 Sync_Mark。如果该报文是参考报文，则定时器值的 Sync_Mark 值复制到 Ref_mark。周期时间就等于定时器值减去 Ref_mark，这个时间用于接收报文的时间戳，或者用于待发送报文的触发时间。该定时器内置溢出保护，因此周期时间在一个基本周期内，是持续单向递增的。

ISO 11898-4 不支持 CANFD 帧波特率切换功能，因此 CAN 控制器总是按照较慢的正常位速率运行定时器。该定时器是自由运行的，不受同步操作或者波特率切换的影响，所以定时器滴答与 CAN 位的开始不同步。

定时器运行于 CAN 时钟域，而所有的控制位和状态位均位于主机时钟域，因为读取定时器值需要经由跨时钟域同步，所以主机无法读取到绝对的定时器值，需要通过触发事件将主机的动作与该定时器进行同步。

由触发事件，接收和发送引发的中断均需要经由时钟域同步，因此会有几个时钟延迟，但是这种情况仅针对于当主机应用程序在触发后决定是否启动发送时才有相关意义（因此建议使用传输触发）。在其他情形下，主机应用程序有足够的时间准备下一个触发事件（即下一个时间窗）。基本上所有的主机运行都足够快，以确保在 CAN 帧持续期间（即时间窗内）内完成多项任务。

ISO 11898-4 级别 1 的时序也并非完美，因为 CAN 位不能基于 CAN 同步进行缩短或延长，所以周期时间计数值不能与 CAN 位保持同步计数，一个节点的周期时间可能与其他节点的周期时间不同。通常情况下，这个差异是 $+/-1$ 滴答，但也不限于此差值，在发送参考报文时，在新的基本周期的开始，所有的节点会同步。

发送触发器被激活后，适当的发送只能在下一个 CAN 位启动，所以一个帧的 SOF 传输的最早时间点是 TT_TRIGGER+1。

21.4.5 TTCAN触发类型

触发类型由 TTYPE 位配置，TTPTR 指针指向 TB 报文槽，TT_TRIGGER 位定义触发的周期时间。

但是所有触发以及相关的操作必须在达到最大周期时间值“0xFFFF”之前完成，因为这个值定义的是一个基本周期的最大长度值。除了立即触发，所有的触发都会置起 TTIF 位。

当设置 TTTBM 位=0，表示仅支持时间触发，在此模式下启动其他触发均会置起 TEIF 位。

当写 TT_TRIGGER 位 (TTTRIG[15:0]) 时，触发器被激活，此时 TTTRIG[15:0]位的写操作将被锁定，除非触发时间已到达（如果 TTIE 位有使能的话，会置起 TTIF 位）或者检测到错误（TEIF 位置起），才能解锁，所以新的触发操作不会覆盖当前正在进行的触发。通过设置 TTEN 位=0，也可以解锁写访问。

21.4.5.1 立即触发

立即触发立即发送由 TTPTR 位所指向的帧，而且不置起 TTIF 位，如果需要启动立即触发，则对 TTTRIG[15:8]位执行写操作，写入 TTTRIG 寄存器的值对立即触发没有影响。

在 TTCAN 模式中，TPE、TSONE 以及 TSALL 位不可用，取而代之的是立即触发，帧数据的发送只能通过立即触发进行启动，在第一帧数据发送完成之前（发送成功或发送失败），主机不可产生第二个立即触发指令。

对于立即触发而言，可以通过 RETLIM 和 REALIM 寄存器限制重新仲裁和重新发送，TSA 位可以用来中止发送（使用 TPA 位没有意义）。

如果立即触发的 TTPTR 指针指向空槽，则 TEIF 位会置起。

21.4.5.2 时间触发

时间触发只是通过设置 TTIF 位来生成事件并产生中断，不会有其他动作。

时间触发可以用作接收触发。如果节点期望在一个时间窗内接收到一帧报文，那么如果该报文丢失且 RIF 位未置起，则可以使用接收触发发出信号反映此异常。应该在预期能够成功接收报文的最后时刻之后设置接收触发。

如果 TT_TRIGGER 低于实际的周期时间，则会置起 TEIF 标志。

当 TTTBM 位=0 时，可以使用时间触发。这是此模式下唯一可用的触发类型。

21.4.5.3 单次发送触发

单次发送触发用于专用时间窗，在专用时间窗内，报文需要以单次发送模式进行发送，所选报文由 TTPTR 位指定。单次发送模式是自动启动的，不受 RETLIM 和 REALIM 位配置的影响，RETLIM 和 REALIM 寄存器被忽略且保持不变。

单次发送触发用于专用时间窗，为此，ISO 11898-4 定义的一个发送使能窗口，该使能窗口最多为 16 个周期时间滴答，寄存器位 TEW(3:0)+1 用于定义滴答数。如果总线被其他帧占用，则无法启动帧发送。这种情形不应该发生在专用时间窗，但是发送使能窗口可以确保不会发生启动延迟，因为启动延迟会与下一个时间窗冲突。如果发送使能窗口关闭，帧无法启动，则会中止帧发送。帧的 TB 槽会因此被标记为空，并置起 AIF 标志。TB 槽内的帧数据保持不变（不会受影响），如果下次尝试传输该数据，则只需要将 TB 槽标记为满。

如果 TT_TRIGGER 低于实际周期时间，则 TEIF 标志置起，不会执行任何动作。

21.4.5.4 发送启动触发

传输启动触发是专门用于已合并的仲裁时间窗，在这个时间窗内，多个节点可以发送报文，并且会发生 CAN 仲裁，发送的报文由 TTPTR 位指定，RETLIM 和 REALIM 寄存器决定是否进行重新仲裁或者重新发送。

如果所选的报文无法被发送（仲裁丢失，发生错误后的多次发送），那么可以使用发送停止触发来中止报文发送。

如果 TT_TRIG 低于实际周期时间，则 TEIF 标志将置起，不执行任何动作。

21.4.5.5 发送停止触发

发送停止触发专用于中止发送操作，即中止通过发送启动触发启动的发送操作（参考 21.4.5.4 章节）。如果发送中止，则帧数据被丢弃。AIF 标志会置起，并且帧数据的 TB 槽会标记为空。TB 槽中的帧数据不会受影响，所以只需要在下次尝试发送该笔数据时将槽标记为“满”。

发送停止触发类似于上述的单次发送触发所使用的中止发送使能窗口。如果发送中止触发指向空槽（意思是说，帧数据已传输），那么不会执行任何动作，也不会置起 TEIF 标志。

如果 TT_TRIG 低于实际周期时间，则 TEIF 标志将置起，会执行中止动作。

21.4.6 TTCAN窗口触发

窗口触发不同于 21.4.5 章节所讲的通用触发类型，它有一个专用的中断标志 WTIF 位，如果周期计数等于 TT_WTRIF 的设定值，则 WTIF 位将置起。

窗口触发用于判断距离上一次有效参考报文的时间是否过长，参考报文可以在周期性循环中或事件之后进行接收，主机应用程序需要关注此动作，并且对窗口触发进行相应调整。

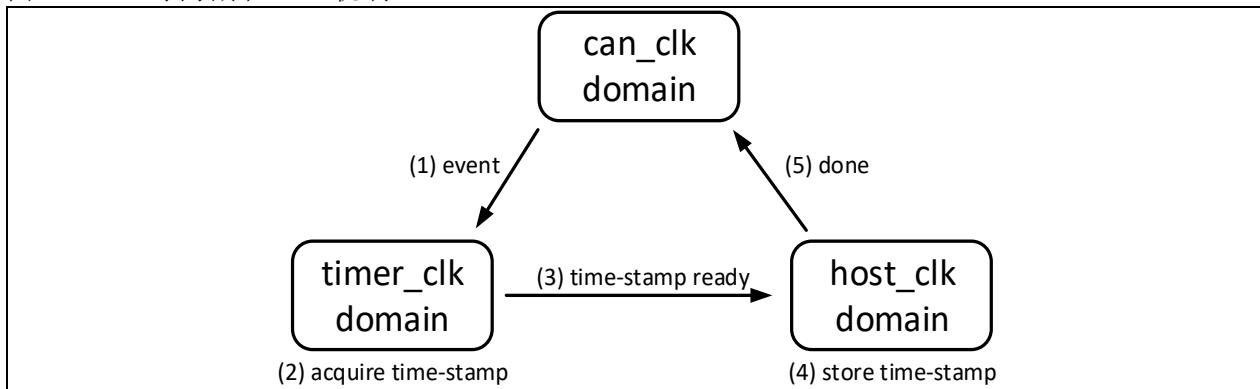
窗口触发默认值“0xFFFF”自动生效。

如果 TT_WTRIG 更新且低于实际周期时间，则将置起 TEIF 标志。

21.5 CiA 603时间戳

CiA 603 协议中的 CAN in Automation (CiA) 定义了生成至少 16 位时间戳的方法，可以选择由 CAN 控制器进行支持。从基本概念来讲，CiA 603 是使用一个自由运行的定时器来计数时钟周期，而不是计数 CAN 位时间，如图 21-13 所示，在发生事件后，定时器值被获取，这个就是时间戳。该时间戳分别存放在 RTS 或者 TTS 位，可以由主机读取，存储时间戳之后，会通过握手信号指示该操作已完成，从而触发 CAN 控制器 3 个时钟域的跨时钟域同步机制 (CDC)。

图 21-13 时间戳和CDC机制



可以通过设置 TSPOS 位，选择 SOF 或 EOF 采样点获取时间戳，ACK 分隔符之后的 7 个隐性位构成 CAN 或者 CANFD 帧的 EOF 位，对于接收器而言，一帧数据在 EOF 的倒数第二位有效，而对于发送器而言，一帧数据在 EOF 的最后一位有效。

在大多数系统中，基于接收和发送中断的软件时间戳被广泛使用，在有效帧的 EOF 处获取时间戳与软件时间戳类似。

CiA 603 支持时间戳和 AUTOSAR 的时间同步。对于 AUTOSAR 来说，CAN 网络中的一个节点就是时间主机。时间主机发送同步报文(SYNC 报文)，时间主机和所有时间从机均可以获取 SYNC 报文的时间戳。从发出 SYNC 报文指令到 SYNC 报文被真正发送之间的时间差将由时间主机通过随后的报文发出。因此，CAN 控制器针对发送帧 (TTS) 仅提供一个时间戳，而针对全部接收帧 (RTS) 提供单独的时间戳是可以满足使用要求的。可以通过 TBUF 中的 TTSEN 位，单独使能和关闭每一个帧的发送帧的时间戳。

CiA 定义的读取和更改定时器的相关规则，CAN 控制器不具有定时器，它依靠的是一个外部定时器，外部定时器的来源可通过 SCFG_CFG2 寄存器的 CAN1_TST_SEL 位来配置。CAN 控制器仅包含时间戳机制，存储 TTS 位的寄存器和存储 RTS 位的寄存器。

寄存器 TNCFG 的 TSEN 位用于关闭和启用时间戳，如果时间戳关闭，则 TTS 和 RTS 位均是 0。

21.6 CAN位时间

21.6.1 位速率

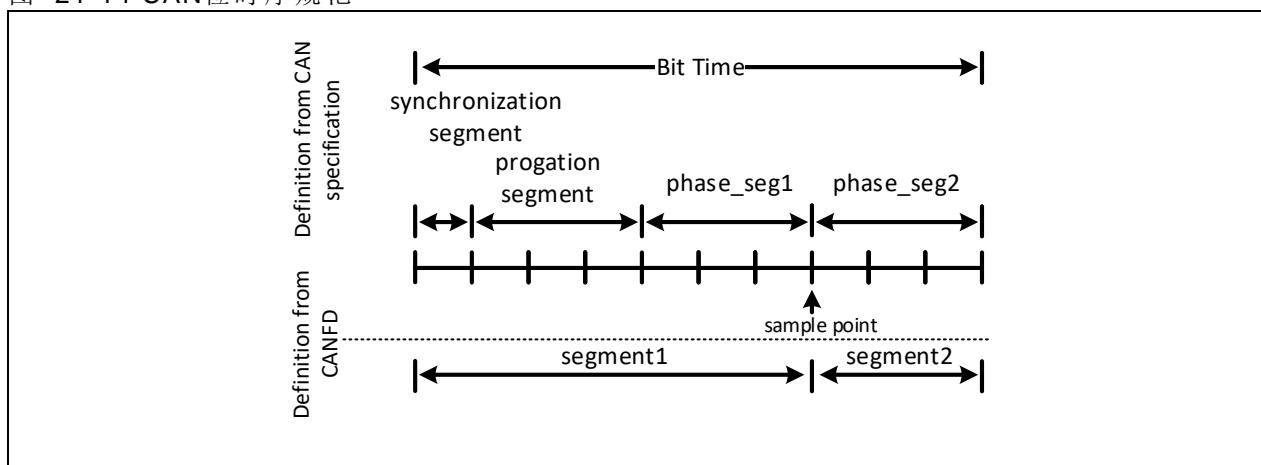
Classic CAN2.0B 定义了高达 1Mbit/s 比特率，对于 CANFD 帧则没有固定限制，对于实际通信系统，数据速率取决于所使用的收发器、所需的 CAN 总线拓扑结构和 CAN 控制器可实现时钟频率的限制。

CANFD 可以被配置为任意比特率，仅受位时序和分频寄存器适当配置范围的限制。

Classic CAN2.0B 在整帧中以一个选定的比特率通信，与之相对的是，CANFD 在帧的开头以“慢速”（标称比特率）通信，在帧的中间切换至“快速”（数据比特率），之后再帧的结尾或者检测到错误之后切换至“慢速”（标称比特率）。

21.6.2 位时间定义

图 21-14 CAN位时序规范



CAN 通信定义了称为“时钟数据恢复”（重同步）的机制，作为 CAN 协议规范的一部分，这些重同步机制基于 CAN 位的定义，如图 21-14 所示。对于 Classic CAN2.0B 和 CANFD，CAN 位时间的定义遵循相同的概念：CAN 位时间 BT 由几个段组成，每段由 n_{TQ} 个时间片单元组成，时间片单元的时间长度 t_{TQ} 为：

$$t_{TQ} = \frac{n_{prescaler}}{f_{can_clk}}$$

n_{TQ} 和 $n_{prescaler}$ 的值需要根据系统时钟频率 f_{can_clk} 来配置，以匹配期望的位时间 BT ，比特率 $BR = 1/BR$ ，其中 x 是以 AC 或 FD 为前缀的占位符。

$$\begin{aligned} BT &= n_{TQ} * t_{TQ} = n_{TQ} * \frac{n_{prescaler}}{f_{can_clk}} = t_{seg_1} + t_{seg_2} \\ t_{seg_1} &= (x_SEG_1 + 2) * t_{TQ} \\ t_{seg_2} &= (x_SEG_2 + 1) * t_{TQ} \\ t_{SJW} &= (x_SJW + 1) * t_{TQ} \\ n_{prescaler} &= PRESC + 1 \end{aligned}$$

CAN 控制器简化了位时间配置，如图所示，同步段、传播段和 phase_seg1 段被组合为段 1，位段 1 的时间长度为 t_{seg_1} ，位段 2 的时间长度为 t_{seg_2} 。

SJW 是同步跳跃宽度，为 TQ 的整数倍，同步的目的是确保接收到的数据信号电平的转换处于同步段中，以将采样点移动到接收信号稳定的位置。如果电平转换提早或延迟到达，CAN 节点可以移除或者增加一些 TQ 来缩短或者延长 CAN 位时间，SJW 限制了一次同步所能移除或者增加 TQ 的最大数量。

信号在 CAN 网络中传播需要一定时间，信号反射（“振铃”）是造成传播延时的物理效应之一，信号传播应该在 CAN 位的传播段内完成。

CAN 节点对接收到的比特流进行重同步，重同步发生在隐性电平到显性电平的边沿，CAN 节点插入填充位，以保证重同步事件发生的已知最大间隔。因此，在重同步之间会接收到几位的数据，由于振

荡器容差的缘故，发送节点和接收节点将发生轻微的漂移，下一次重同步事件将对此进行补偿。**phase_seg1** 和 **phase_seg2** 用于保留发送节点和接收节点之间的相位偏差，因此，**phase_seg1** 和 **phase_seg2** 段持续时间相同。

SJW 的最大值等于 **phase_seg2** 段，换言之，重同步的最大数量与两次重同步事件之间预期的最大相位偏差一致。

配置 CAN 位时间以实现最快的比特率是一个复杂的议题，简单言之：

- 信号在网络中的传播与传播段相关
- 发送节点和接收节点之间的振荡器容差影响段1和段2的相位

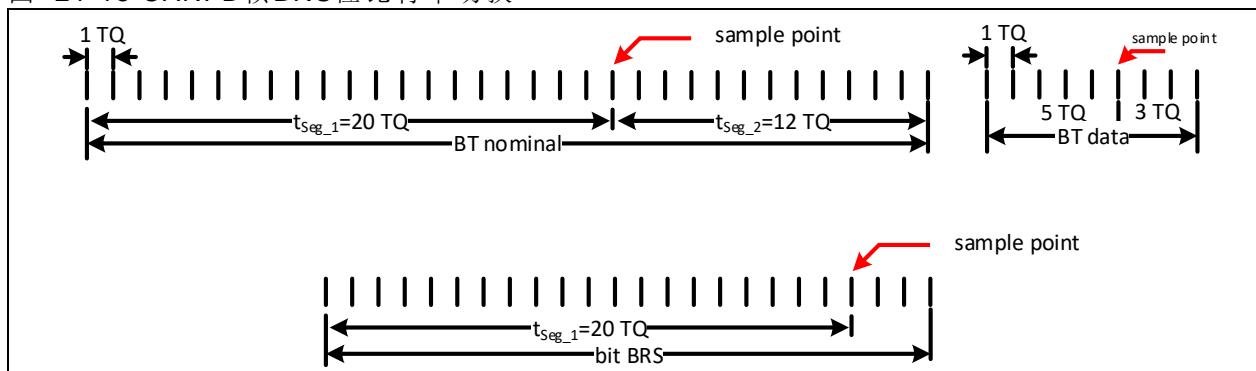
合适的 CAN 位时间配置如下：

- 采样点应该大致在位时间的 60% 到 80% 范围内，通常，将其设置为 75% 到 80% 是最佳选择
- 如果可能的话，选择较低的分频器配置，从而每个位时间可以产生更多的时间片 TQ
- 在大多数情况下，优先设置 **SJW** 等于段 2 的时间长度

21.6.3 CANFD 位速率转换和采样点

在 CANFD 帧的比特率切换有效 (**BRS=1**) 的情况下，对于 CAN 网络中的所有节点，需要具有相同确切位置的采样点。在 **BRS** 位和 **CRC** 界定符的采样点位置，比特率进行切换，所以这些位的时间长度介于“慢速”和“快速”之间。如下图 21-15 所示，采样点的位置对 **BRS** 位的绝对时间长度由很大影响，如果数据比特率远远大于标称比特率，那么较早或较晚的采样点可能会导致快速数据比特率下的错误采样。

图 21-15 CANFD 帧 BRS 位比特率切换



总之，采样点的位置对于比特率切换是非常重要的，所以在标称比特率和 CANFD 数据比特率中，建议在 CAN 通信网络中，所有的 CANFD 帧节点采用相同的时序参数。换言之，所有节点的段 1 和段 2 应该设置为相同的时间长度。此外建议采用相同时间长度的时间片 TQ，以便使所有的节点同步相等。

21.6.4 TDC 和 RDC

对于 CANFD 帧节点，发送器延迟补偿 (TDC) 可以选择性使能，而接收器延迟补偿 (RDC) 自动激活 (Classic CAN2.0 帧节点不支持 TDC 和 RDC 功能)。TDC 和 RDC 机制的应用基于以下背景：采用 CANFD 的数据比特率进行通讯时，发送器的发送延迟或者总线延迟可能会大于一个位时间，这种情况下，需要使用 TDC 进行延时补偿，否则 CANFD 帧数据场的比特率会受到限制，因为如果发送器在比特位的采样点无法检测到自己最新传输的比特位，则发送器会检测到位错误。

图 21-16 发送器延迟

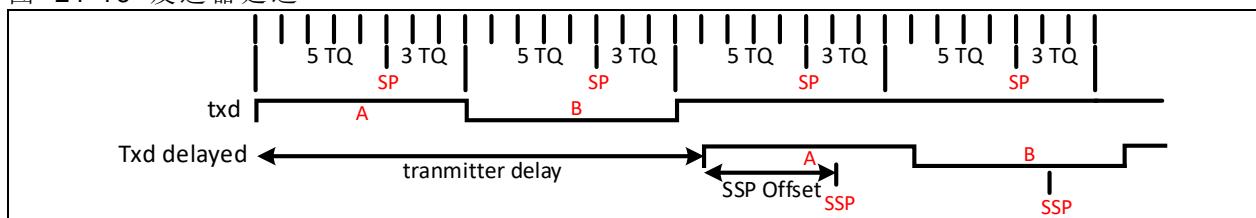


图 21-16 给出了一个发送器延时较大时对总线影响的示例，图中，发送器开始发送 A 和 B 两个比特位，位时间 $t_{seg_1} = 5TQ$, $t_{seg_2} = 3TQ$ ，采样点 (SP) 位于段 1 和段 2 之间。在示例中，发送器的延时大于 2 个位时间，因此原始的采样点 (SP) 无法采到正确的位值，CANFD 规范定义了附加二次采样点 (SSP)。如果 TDC 使能 (SSPOFF 位 $\neq 0$)，CAN 控制器自动确定发送器延时。SSP 的位置采

用发送器的延时加上 SSP 延时（由 SSPOFF 位配置）得到，SSPOFF 位必须配置为 TQ 的整数倍，并建议采用 $t_{Seg_1} + 1$ 。

注：建议 SSPOFF 位采用 $t_{Seg_1} + 1$ 而不是 t_{Seg_1} ，是因为采样一些数据时存在量化误差。在 CANFD 帧，FDF 位与 res 位的下降沿位置，发送器的延时自动进行处理。

ISO 11898-1:2015 规定在 TDC 使能情况下，只能配置分频寄存器 PRESC 位=0 或 1，对于 CAN 控制器而言，在数据比特率下，能够自动检测高达 3 个位时间的发送器延时。RDC 在接收过程中自动完成，不论 TDC 是否开启。

注：RDC 只能在 $n_{prescaler} > 1$ 的情况下起作用。

21.7 CAN 寄存器

除 ACF/RBUF/TBUF 相关寄存器，外设寄存器可以按字节（8 位）的方式进行写操作，但只能按字（32 位）的方式进行读操作。

ACF/RBUF/TBUF 相关寄存器只能按字（32 位）的方式进行读写操作。

表 21-9 CAN 寄存器列表及其复位值

寄存器简称	基址偏移量	复位值
TNCFG	0000h	0x0200 0000
ACTIME	0004h	0x0505 0008
FDTIME	0008h	0x0202 0003
LBTCFG	0010h	0x7700 0000
STS	0014h	0x0000 0000
TSTAT	001Ch	0x0000 0000
TTS	0020h	0x0000 0000
CTRLSTAT	0028h	0x0090 0080
ERR	002Ch	0x0000 0010
REFMSG	0030h	0x0000 0000
TTCFG	0034h	0x0000 0000
TTTRIG	0038h	0xFFFF 0000
ACF	0044h~0064h	
ACFCTRL	0044h	0x0001 0000
FCID	0048h	0XXXXX XXXX
FCFMT	004Ch	0XXXXX XXXX
FCTYP	0050h	0XXXXX XXXX
FMID	0058h	0XXXXX XXXX
FMFMT	005Ch	0XXXXX XXXX
FMTYP	0060h	0XXXXX XXXX
Res 1	0064h	0XXXXX XXXX
RBUF (包括 RTS)	0070h~00CBh	
RBID	0070h	0XXXXX XXXX
RBFMT	0074h	0XXXXX XXXX
RBTYP	0078h	0XXXXX XXXX
RBDAT1	0080h	0XXXXX XXXX
RBDAT2	0084h	0XXXXX XXXX
...
RBCIA1	T+0	0XXXXX XXXX
RBCIA2	T+4	0XXXXX XXXX
RBTCAN	T+8	0XXXXX XXXX

TBUF	00CCh~011Bh	
TBID	00CCh	0XXXXX XXXX
TBFMT	00D0h	0XXXXX XXXX
TBTYP	00D4h	0XXXXX XXXX
Res 2	00D8h	0XXXXX XXXX
TBDATA1	00DCh	0XXXXX XXXX
TBDATA2	00E0h	0XXXXX XXXX
...
LLCFORMAT	0124h	0x0000 0000
LLCSIZE	0128h	0x0010 0000
INTEN	0140h	0x0000 0000

21.7.1 CAN CiA 603时间戳及节点控制寄存器 (TNCFG)

域	简称	复位值	类型	功能
位 31: 26	保留	0x0	resd	请保持默认值。
位 25	TSPOS	0x1	rw	<p>时间戳位置 (Time stamping position) 0: SOF 1: EOF</p> <p>TSPOS 在 TSEN 配置为 0 时可更改，也可以在配置 TSEN 为 1 的同时修改 TSPOS 配置。</p>
位 24	TSEN	0x0	rw	<p>时间戳使能 (Time stamping enable) 0: 禁用 1: 使能</p>
位 23: 18	保留	0x0	resd	请保持默认值。
位 17	ROP	0x0	rw	<p>受限操作 (Restricted operation) 0: 受限操作关闭; 1: 受限操作打开。</p> <p>注：当传输正在进行中时，ROP 禁止修改。如果 ROP 配置为 1，则禁止发送数据。</p>
位 16: 0	保留	0x0	resd	请保持默认置

21.7.2 Classic CAN2.0B位时序寄存器 (ACTIME)

域	简称	复位值	类型	功能
位 31	保留	0x0	resd	请保持默认值。
位 30: 24	AC_SJW	0x5	ro	<p>重新跳跃宽度 $t_{SJW} = (AC_SJW + 1) * t_{TQ}$</p>
位 23	保留	0x0	resd	保持默认值。
位 22: 16	AC_SEG_2	0x5	rw	<p>位时间段 2 设置为 $t_{SEG_2} = (AC_SEG_2 + 1) * t_{TQ}$</p>
位 15: 9	保留	0x0	resd	请保持默认值
位 8: 0	AC_SEG_1	0x8	rw	<p>位时间段 1 采样点设置为 $t_{SEG_1} = (AC_SEG_1 + 2) * t_{TQ}$</p>

21.7.3 CANFD位时序寄存器 (FDTIME)

域	简称	复位值	类型	功能
位 31	保留	0x0	resd	请保持默认值。
位 30: 24	FD_SJW	0x2	rw	重新跳跃宽度 $t_{SJW} = (FD_SJW + 1) * t_{TQ}$
位 23	保留	0x0	resd	请保持默认值。
位 22: 16	FD_SEG_2	0x2	rw	位时间段 2 设置为 $t_{SEG_2} = (FD_SEG_2 + 1) * t_{TQ}$
位 15: 8	保留	0x0	resd	请保持默认值
位 7: 0	FD_SEG_1	0x3	rw	位时间段 1 采样点设置为 $t_{SEG_1} = (FD_SEG_1 + 2) * t_{TQ}$

21.7.4 CAN限制和位时间配置寄存器 (LBTCFG)

域	简称	复位值	类型	功能
位 31	保留	0x0	resd	请保持默认值
位 30: 28	RETLIM	0x7	rw	重发次数限制 (Retransmission limit) 111: 无限制 (只被发送错误计数器 TECNT 限制) 110: 7 次 ... 000: 1 次 (不重发) 如果在发送过程中出错, CAN 节点在总线空闲时立即重新发送。重新发送的次数限制通过 RETLIM 进行配置。 RETLIM 可以在任意时刻进行更新, 但是当 RETLIM 更新后, 需要将该寄存器同步至 can_clk 时钟域供 CAN 协议状态机使用, 在此期间, RETLIM 的写过程被锁定。
位 27	保留	0x0	resd	请保持默认值
位 26: 24	REALIM	0x7	rw	重新仲裁次数限制 (Re-arbitration limit) 111: 无限制 110: 7 次 ... 000: 1 次 (不重新仲裁) 如果两个或者更多 CAN 节点同时发送报文, 低优先级的报文丢失仲裁退出总线竞争。当总线处于空闲状态时, 丢失仲裁的 CAN 节点立即重新参与总线仲裁, 重新仲裁的次数限制由 REALIM 进行配置。REALIM 可以在任意时刻进行更新, 但是当 REALIM 更新后, 需要将该寄存器同步至 can_clk 时钟域供 CAN 协议状态机使用, 在此期间, REALIM 的写过程被锁定。
位 23: 16	保留	0x0	resd	请保持默认值
位 15: 8	FD_SSPOFF	0x0	rw	二次采样点偏移 (Secondary sample point offset) 当 FD_SSPOFF 不等于 0 且 CES 配置为 1 时, 发送延迟补偿 (TDC) 使能。假如 BRS 被激活, TDC 在 CAN FD 帧的数据场起作用。 发送器延迟加上 FD_SSPOFF 定义的时间为 TDC 之后的第二次采样点。FD_SSPOFF 是时间片 TQ 的整数。
位 7: 5	保留	0x0	resd	请保持默认值
位 4: 0	PRESC	0x0	rw	分频器 (Prescaler) 系统时钟根据分频器配置分频得到时间片 tq。 $N_{prescaler} = PRESC + 1$ 或者 $T_{tq} = N_{prescaler}/f_{can_clk}$

21.7.5 CAN状态寄存器 (STS)

域	简称	复位值	类型	功能
位 31	EWARN	0x0	ro	达到错误警告限制 (Error warning limit reached) 0: RECNT 和 TECNT 计数器小于 EWL 1: RECNT 或者 TECNT 其中一个计数器大于等于 EWL
位 30	EPASS	0x0	ro	错误被动状态 (Error passive status) 0: 节点处于错误主动 1: 节点处于错误被动
位 29: 14	保留	0x00	resd	保持默认值。
位 13	WTIF	0x0	rw1c	TTCAN: 窗口触发中断标志 (Watch trigger interrupt flag) 当周期计数达到 TT_WTRIG 配置限制值并且 WTIE 使能情况下, WTIF 被置位。
位 12	TEIF	0x0	rw1c	TTCAN: 触发错误中断标志 (Trigger error interrupt flag) TEIF 的置位条件如文中描述, 没有对应的寄存器用于禁用 TEIF 的状态显示。
位 11	TTIF	0x0	rw1c	TTIF: 时间触发中断标志 (Time trigger interrupt flag) 当 TTIE 被配置时, 周期时间等于触发时间 TT_TRIG, 则 TTIF 被置位。该位写 1 清零, 写 0 无影响。如果 TT_TRIG 不被更新, 那么在下一次的基本周期, TTIF 不会被置位, 即仅置位一次。
位 10	EPIF	0x0	rw1c	错误被动中断使能 (Error passive interrupt flag) 如果 EPIE = 1, 当错误状态从主动错误状态切换至被动错误状态, 则 EPIF 被置位, 反之亦然。
位 9	ALIF	0x0	rw1c	仲裁丢失中断标志 (Arbitration lost interrupt flag)
位 8	BEIF	0x0	rw1c	总线错误中断标志 (Bus error interrupt flag)
位 7	RIF	0x0	rw1c	接收中断标志 (Receiv interrupt flag) 0: 未接收到帧数据 1: 接收到一帧数据帧或者远程帧, 并存储至接收缓存
位 6	ROIF	0x0	rw1c	RB 溢出中断标志 (RB overflow interrupt flag) 0: RB 没有数据覆盖 1: RB 中至少有一帧接收的数据被覆盖
位 5	RFIF	0x0	rw1c	RB 满中断标志 (RB full interrupt flag) 0: RB FIFO 未满 1: 所有 RB 满。如果在下一帧有效数据接收完成之前, 没有 RB 被释放, 则最先存储的帧数据丢失。
位 4	RAFIF	0x0	rw1c	RB 几乎满中断标志 (RB almost full interrupt flag) 0: 填充的 RB 槽数量 < AFWL_i 1: 填充的 RB 槽数量 ≥ AFWL_i
位 3	TPIF	0x0	rw1c	发送主中断标志 (Transmission primary interrupt flag) 0: PTB 没有发送完成 1: PTB 的发送请求已成功执行 在 TTCAN 模式中, TPIF 不会被置位, 只有 TSIF 标志可用。
位 2	TSIF	0x0	rw1c	发送次中断标志 (Transmission secondary interrupt flag) 0: STB 没有发送完成 1: STB 的发送请求已成功执行 在 TTCAN 模式中, 无论帧存储的位置, 所有成功的发送都会置位 TSIF。
位 1	EIF	0x0	rw1c	错误中断标志 (Error interrupt flag) 0: 状态无任何改变 1: 越过错误警告界限, 高于或者低于错误警告界限 BUSOFF 位硬件置 1 或者清除
位 0	AIF	0x0	rw1c	中止中断标志 (Abort interrupt flag) 0: 未中止发送 1: 当在设置 TPA 或者 TSA 之后, 帧数据发送中止。不建议同时将 TPA 和 TSA 置位, 因为两者都会置位 AIF。 AIF 标志位无对应的使能位。

21.7.6 CAN发送状态寄存器 (TSTAT)

域	简称	复位值	类型	功能
位 31: 27	保留	0x0	resd	保持默认值 当前发送状态 (Current transmission status code) 000: 无传输进程 (IDLE) 001: 正在传输进程, 且无任何问题 (ONGOING) 010: 仲裁丢失, 根据 REALIM 配置进行重新仲裁 (LOST_ARBITRATION) 011: 传输成功完成 (TRANSMITTED) 100: 传输中止, 配置 TPA 或者 TSA (ABORTED) 101: 传输错误, 根据 RETLIM 配置进行重新发送 (DISTURBED) 110: LCC 帧格式配置错误 (REJECTED) 111: 保留
位 26: 24	TSTAT_2	0x0	ro	用于帧识别的句柄 (Handle for frame identification) 建议使用软件计数器作为 HANDLE 值, 该值每发送一帧新的数据进行递增。
位 23: 16	HANDLE_2	0x0	ro	最终发送状态 (Final transmission status code) 000: 无传输进程 (IDLE) 001: 正在传输进程, 且无任何问题 (ONGOING) 010: 仲裁丢失, 根据 REALIM 配置进行重新仲裁 (LOST_ARBITRATION) 011: 传输成功完成 (TRANSMITTED) 100: 传输中止, 配置 TPA 或者 TSA (ABORTED) 101: 传输错误, 根据 RETLIM 配置进行重新发送 (DISTURBED) 110: LCC 帧格式配置错误 (REJECTED) 111: 保留
位 15: 11	保留	0x0	resd	保持默认值
位 10: 8	TSTAT_1	0x0	ro	用于帧识别的句柄 (Handle for frame identification) 建议使用软件计数器作为 HANDLE 值, 该值每发送一帧新的数据进行递增。
位 7: 0	HANDLE_1	0x0	ro	最终发送状态 (Final transmission status code) 000: 无传输进程 (IDLE) 001: 正在传输进程, 且无任何问题 (ONGOING) 010: 仲裁丢失, 根据 REALIM 配置进行重新仲裁 (LOST_ARBITRATION) 011: 传输成功完成 (TRANSMITTED) 100: 传输中止, 配置 TPA 或者 TSA (ABORTED) 101: 传输错误, 根据 RETLIM 配置进行重新发送 (DISTURBED) 110: LCC 帧格式配置错误 (REJECTED) 111: 保留

21.7.7 CAN发送时间戳32位 (TTS)

域	简称	复位值	类型	功能
位 31: 0	TTS	0x0	ro	发送时间戳 (Transmission time stamp) TTS 记录上一帧发送数据对应的 CiA 603 时间戳。如果更新使能控制位 TTSEN = 1, 则每发送一帧新的数据, 覆盖 TTS。时间戳为 32 位, 未使用位强制为 0。TTS 用于获取同步信息的时间戳。

21.7.8 CAN控制和状态寄存器 (CTRLSTAT)

域	简称	复位值	类型	功能
位 31	SACK	0x0	rw	自响应 (self-acknowledge) 0: 禁用自响应 1: 当 LBME=1 时自响应
位 30	ROM	0x0	rw	接收缓存溢出策略 (Receive buffer overflow mode) 0: 最先接收的帧数据被覆盖 1: 新接收的帧数据丢弃
位 29	ROV	0x0	ro	接收缓存溢出 (Receive buffer overflow) 0: 没有溢出 1: 溢出, 至少一帧数据丢失 ROV 在配置 RREL=1 时被硬件清除。
位 28	RREL	0x0	rw	接收缓存释放 (Receive buffer release) 上位机 (软件) 读 RB 槽然后释放缓存。之后硬件指针指向下一个 RB 槽, RSTAT 更新。 0: 无释放操作 1: 释放: 上位机 (软件) 已读缓存 RB
位 27	RBALL	0x0	rw	接收缓存存储所有数据帧 (Receive buffer stores all data frames) 0: 正常操作 1: RB 缓存存储所有正确或者错误的帧数据
位 26	保留	0x0	resd	保持默认值
位 25: 24	RSTAT	0x00	ro	接收缓存状态 (Receive buffer status) 00: 空 01: 非空非几乎满 (AFWL 配置) 10: 大于等于几乎满 (AFWL 配置阈值), 非满且无上溢 11: 满 (当上溢时, 维持置位状态, 上溢标志参考 ROV)
位 23	FD_ISO	0x1	rw	CAN FD ISO 模式 (CAN FD ISO mode) 0: Bosch CAN FD (non-ISO) 模式 1: ISO CAN FD 模式 (ISO 11898-1:2015) ISO CAN FD 模式的 CRC 初始值不同, 另外还有额外的填充位计数器。两个模式在同一个 CAN 网络中并不兼容。该位配置对 CAN2.0B 无影响。 该位只能在 RESET=1 时进行改写。
位 22	TSNEXT	0x0	rw	下一个发送缓冲区 (Transmit buffer secondary next) 0: 无动作 1: STB 槽已填充, 选择下一个槽 当所有的帧数据写入 TBUF 寄存器, 上位机 (软件) 需要置位 TSNEXT 来指示当前槽已填充, 之后 TBUF 寄存器指向 STB 需要填充帧数的下一个槽。一旦槽被标记为已填充一帧数据, 配置 TSONE 和 TSALL 寄存器可以开始传输。 TSNEXT 和 TSONE 或者 TSALL 可以在同一个写周期配置, TSNEXT 被软件设置, 之后硬件立马自动清零。 当 TBSEL=0 时, 设置 TSNEXT 无意义, 此时设置 TSNEXT 对硬件无影响, 并被硬件自动清零。 如果 STB 满, TSNEXT 保持置位直到 STB 有一个槽被释放。在 TTCAN 模式, TSNEXT 不可用, 被固定为 0.
位 21	TSMODE	0x0	rw	STB 操作模式 (Transmit buffer secondary operation mode) 0: FIFO 模式 1: 优先级决定模式 FIFO 模式下, 帧数据按写入 STB 的先后顺序依次发送。 优先级决定模式下, STB 拥有最高优先级的帧数据自动最先发送。数据帧的 ID 号决定帧数据的优先级, 帧数据拥有最小 ID 号则优先级最高。PTB 中的帧数据无论 ID 号大小, 总是最高优先级。 TSMODE 只在完成一次软件复位之后 (置位与清除 RESET 位), 并且 STB 为空时进行切换。

				TTCAN 发送缓存模式 (TTCAN transmit buffer mode) 如果 TTEN=0 那么 TTTBM 不起作用, 否则: 0: PTB 和 STB 彼此独立, TSMODE 决定行为 1: 支持 TTCAN: TBPTR 和 TTPTR 选择缓存槽
位 20	TTTBM	0x1	rw	对于事件驱动的 CAN 通讯 (TTEN=0), 系统提供 PTB 和 STB, TSMODE 配置决定 STB 的行为, TTTBM 不起作用。 对于时间触发的 CAN 通讯 (TTEN=1), 支持时间触发通讯的所有特征, TTTBM 需要被配置为 1。TTPTR 和 TBPTR 指向发送缓存槽的指定位置。 对于时间触发的 CAN 通讯 (TTEN=1), 只支持接收报文时间戳, 则 TTTBM 可以被配置为 0。发送缓存采用事件触发模式, TSMODE 决定其行为。 TTTBM 只能在 TBUF 为空时进行更改。
位 19	保留	0x0	resd	保持默认值 如果 TTEN=0 或 TTTBM=0: STB 缓存满标志 (Transmit secondary buffer full flag) 1: STB 已填充最大数量的帧数据 0: STB 未填充最大数量的帧数据
位 18	TSFF	0x0	rw	如果 TTEN=1 且 TTTBM=1: 发送缓存槽满标志 1: TBPTR 指定的缓存槽满 0: TBPTR 指定的缓存槽空 当 TSNEXT=1 之后的一个额外的<host_clk>周期 TSFF 被更新。因此设置 TSNEXT=1 时, TSFF 不可以立马被读。
位 17: 16	TSSTAT	0x0	ro	STB 状态位 (Transmit secondary status bits) 如果 TTEN=0 或 TTTBM=0 00: STB 空 01: STB 小于等于 1/2 满 10: STB 大于 1/2 满 11: STB 满 如果 TTEN=1 且 TTTBM=1: 00: PTB 和 STB 空 01: PTB 和 STB 非空非满 11: PTB 和 STB 满 当 TSNEXT=1 之后的一个额外的<host_clk>周期 TSSTAT 被更新。因此设置 TSNEXT=1 时, TSSTAT 不可以立马被读。
位 15	TBSEL	0x0	rw	传输缓存选择 (Transmit buffer select) 选择发送缓存来导入一帧数据。使用 TBUF 寄存器进行访问。当 TBUF 寄存器正在进行写操作或者 TSNEXT 被置位时, TBSEL 需要保持稳定。 0: PTB (高优先级缓存) 1: STB 当 TTEN=1 和 TTTBM=1 时, 该位被复位成硬件复位值。
位 14	LOM	0x0	rw	只听模式 (Listen only mode) 0: 禁用 1: 使能 当传输过程正在进行时 LOM 不可被更改。当 LOM 使能和 LBME 禁用情况下, 传输启动禁止。 LOM=1 且 LBME=0 禁用所有传输 LOM=1 且 LBME=1 禁止接收帧或者错误帧回复 ACK, 但本节点内部传输。
位 13	STBY	0x0	rw	收发器待机模式 (Transceiver standby mode) 0: 禁用 1: 使能 该寄存器为外部输出信号 stby, 控制收发器进入待机模式。 当 TPE, TSONE 或者 TSALL 等于 1 时, STBY 不能被置位。如果上位机控制 STBY 切换至 0, 主机在请求新的传输之前需要等待收发器启动。

位 12	TPE	0x0	rw	<p>PTB 发送使能 (Transmit primary enable) 0: 无 PTB 的传输 1: 高优先级 PTB 发送帧数据使能 如果 TPE 被置位, PTB 中的帧数据会在下一帧进行发送。在此之前, STB 中正在进行的传输进程会完成, 新的挂起的帧数据会延迟至 PTB 帧数据传输完成之后再进行发送。 TPE 使能在当前帧数据发送成功或者配置 TPA 中止当前帧发送时, 才可被硬件清零。 当 RESET=1, STBY=1, (LOM=1 且 LBME=1), (TTEN=1 且 TTTBM=1) 或者 ROP=1 时, 该位被复位成硬件复位值。</p>
位 11	TPA	0x0	rw	<p>PTB 发送中止 (Transmit primary abort) 0: 无中止 1: 中止 PTB 已经被请求发送 (TPE=1) 但是还没有开始发送的传输进程, PTB 中的帧数据依然保存。 该位被上位机 (软件) 置位, 硬件清零。置位 TPA 将自动清除 TPE 位。 上位机 (软件) 可以配置 TPA 为 1, 但是不可以复位为 0。在硬件复位该位期间, 该位不可以被置位。 当 RESET=1 或者 (TTEN=1 且 TTTBM=1) 时, 该位被复位成硬件复位值。TPA 和 TPE 不应该同时被置位。</p>
位 10	TSONE	0x0	rw	<p>STB 发送一帧数据 (Transmit secondary one frame) 0: 无 STB 的传输 1: STB 发送一帧数据使能。在 FIFO 模式, 最先写入的帧数据被发送; 在优先级模式, 高优先级的一帧数据被发送。在优先级模式, 如果同时 STB 写入新的帧数据, 则并不总是清楚哪一帧数据会被优先发送。 当总线空闲且 PTB 没有请求挂起 (TPE=1) 时, 控制器开始发送数据。 TSONE 使能在当前帧数据发送成功或者配置 TSA 中止当前帧发送时, 才可被硬件清零。 当 RESET=1, STBY=1, (LOM=1 且 LBME=1), (TTEN=1 且 TTTBM=1) 或者 ROP=1 时, 该位被复位成硬件复位值。</p>
位 9	TSALL	0x0	rw	<p>STB 发送所有帧数据 (Transmit secondary all frames) 0: 无 STB 的传输 1: STB 发送所有帧数据使能 当总线空闲且 PTB 没有请求挂起 (TPE=1) 时, 控制器开始发送数据。 TSALL 使能在当前帧数据发送成功或者配置 TSA 中止当前帧发送时, 才可被硬件清零。 当 RESET=1, STBY=1, (LOM=1 且 LBME=1), (TTEN=1 且 TTTBM=1) 或者 ROP=1 时, 该位被复位成硬件复位值。</p>
位 8	TSA	0x0	rw	<p>STB 发送中止 (Transmit secondary abort) 0: 无中止 1: 中止 STB 已经被请求发送但是还没有开始发送的传输进程, 对于 TSONE 传输, 只有一帧数据被中止; 对于 TSALL 传输, 所有帧数据被中止。同时一帧或者所有帧存储被释放, TSSTAT 寄存器状态被更新。 所有被中止的帧数据丢失, 不能再被访问。在优先级模式, TSONE 传输被中止, 则当新的一帧数据同时被写入到 STB 时, 当前中止的帧数据才会被清除。 该位被上位机 (软件) 置位, 硬件清零。置位 TSA 将自动清除 TSONE 和 TSALL 位。 上位机 (软件) 可以设置 TSA 为 1, 但是不可以复位为 0。当 RESET=1 时, 该位被复位为硬件复位值。 TSA 不应该与 TSONE 或者 TSALL 位同时被置位。</p>
位 7	RESET	0x1	rw	<p>RESET 请求位 (RESET request bit) 0: CAN-CTRL 无局部复位</p>

				1: 主控制器产生 CAN-CTRL 局部复位 一些寄存器（如节点配置）只能在 RESET = 1 时进行修改。寄存器 RESET 强制 CAN-CTRL 部分组件进入复位状态，当节点进入离线状态，RESET 被自动置位。 当 RESET 切换至 0，在检测到 11 个连续隐性电平之后，CAN 节点参与到 CAN 总线通信。 如果 RESET 被设置为 1，并马上切换至 0，CAN-CTRL 需要等待一段时间回到非复位状态，因为从<host_clk>时钟域同步至<can_clk>时钟域需要时间。
位 6	LBME	0x0	rw	回环模式，外部 (Loop back mode, external) 0: 禁用 1: 使能 LBME 在传输进行时禁止更改。LBME 和 TPE, TSONE, TSALL 不应该同时更改。
位 5	LBMI	0x0	rw	回环模式，内部 (Loop back mode, internal) 0: 禁用 1: 使能 LBMI 在传输进行时禁止更改。LBMI 和 TPE, TSONE, TSALL 不应该同时更改。
位 4: 1	保留	0x0	resd	保持默认值
位 0	BUSOFF	0x0	rw	总线离线 (Bus off) 0: 控制器处于在线状态 1: 控制器处于离线状态 写寄存器 BUSOFF 为 1 会复位 TECNT 和 RECNT，该操作只能用于 debug 模式。

注：同时设置 TSONE 和 TSALL 无意义，在 TSONE 被置位的情况下，TSALL 不能被置位，反之亦然。如果 TSONE 和 TSALL 被同时置位，TSALL 优先级更高，TSONE 被硬件清除。

21.7.9 CAN错误配置及状态寄存器 (ERR)

域	简称	复位值	类型	功能
位 31: 24	TECNT	0x0	ro	发送错误计数器 (Transmit error count) TECNT 按照 CAN 协议进行递增或递减。 当处于离线状态时，TECNT 可能溢出。
位 23: 16	RECNT	0x0	ro	接收错误计数器 (Receive error count) RECNT 按照 CAN 协议进行递增或递减。 RECNT 不会溢出。
位 15: 13	KOER	0x0	ro	错误类型 (Error code) 000: 无错误 001: 位错误 010: 格式错误 011: 填充错误 100: 应答错误 101: CRC 错误 110: 其它错误 (包括：在错误标志之后检测到显性位，接收显性错误标志过长，在 ACK 错误之后发送隐性错误标志期间检测到显性位。) 111: 保留 每检测到新的错误时 KOER 更新，因此当成功发送接收一帧数据时，该寄存器保持不变。
位 12: 8	ALC	0x0	ro	仲裁丢失捕捉 (指定一帧数据仲裁丢失的位置)
位 7: 4	AFWL	0x1	rw	接收缓存几乎满警戒界限 (Receive buffer almost full warning limit) AFWL 根据可用的 RB 缓存槽数量 NRB 决定内部警戒界限。 如果 RB 缓存槽填充的帧数据数量等于 AFWL 则 RAFIF 被置位。有效的范围 AFWL=[1...6] AFWL 配置为 0 无意义，硬件自动按 0x1 计算。 AFWL 大于 6 时无意义，硬件自动按 6 计算。 AFWL 等于 6 有效，此时 RFIF 也被置位。

位 3:0	EWL	0xB	rw	可编程错误警告界限=(EWL+1)*8 (Programmable error warning limit) 可能的值为: 8, 16, ... 128. EWL 的值决定 EIF 的状态。 EWL 需要经过时钟域同步同步至 CAN 时钟域。在同步过程中, EWL 寄存器的写操作被锁定, 直到同步完成。
-------	-----	-----	----	---

21.7.10 CAN TTCAN: 参考信息 (REFMSG)

域	简称	复位值	类型	功能
位 31	REF_IDE	0x0	rw	参考信息 IDE 位 (Reference message IDE bit)
位 30: 29	保留	0x0	resd	请保持默认值
位 28: 0	REF_ID	0x0	rw	<p>参考信息 ID (Reference message identifier) 如果 REF_IDE 为: 1: REF_ID(28:0)可用 (基础 ID + 扩展 ID) 0: REF_ID(10:0)可用 (基础 ID)</p> <p>REF_ID 用于 TTCAN 模式识别参考信息, 适用于从机 (接收) 和主机 (发送)。如果参考信息被识别且没有错误, 则帧的 Sync_Mark 作为 Ref_Mark。</p> <p>REF_ID(2:0)不会检测, 对应的寄存器位强制设置为 0, 因此最多系统存在 8 个时间主机。</p> <p>硬件只能通过 ID 号识别参考信息, 数据长度无效。</p> <p>另外: 时间主机采用正常帧的方式发送参考信息, REF_ID 的作用是检测成功的传输了一帧参考信息。</p>

21.7.11 CAN TTCAN: 触发配置寄存器 (TTCFG)

域	简称	复位值	类型	功能
位 31: 27	保留	0x0	resd	保持默认值
位 26: 25	T_PRESC	0x0	rw	<p>TTCAN 计时器分频 (TTCAN timer prescaler) 00: 1 01: 2 10: 4 11: 8</p> <p>TTCAN 的时间基准是 CAN 位时间, 该时间被 PRESC, AC_SEG_1 和 AC_SEG_2 进行配置。T_PRESC 是额外的分频因子, 支持 1,2,4 或者 8 分频。</p> <p>T_PRESC 只能在 TTEN=0 时进行修改, T_PRESC 和 TTEN 可以在同一个写周期进行配置。</p>
位 24	TTEN	0x0	rw	<p>时间触发使能 (Time trigger enable) 1: TTCAN 使能, 计时器运行 0: 禁用</p>
位 23	TBE	0x0	rw	<p>指定 TB 槽为空 (Set TB slot to "Empty") 1: TBPTR 选中的槽空间被标记为空 0: 无动作</p> <p>只要槽空间被标记为空且 TSFF=0, TBE 立即被硬件自动复位为 0。如果当前 TB 槽正在进行数据传输, 则 TBE 保持置位状态, 直到当前传输完成, 或者产生传输错误, 或者仲裁丢失, 传输过程不再继续进行。</p> <p>如果 TBF 和 TBE 同时被置位, 则 TBE 优先。</p>
位 22	TBF	0x0	rw	<p>指定 TB 槽为满 (Set TB slot to "Filled") 1: TBPTR 选中的槽空间被标记为满 0: 无动作</p> <p>只要槽空间被标记为满且 TSFF=1, TBF 立即被硬件自动复位为 0。</p> <p>如果 TBF 和 TBE 同时被置位, 则 TBE 优先。</p>
位 21: 16	TB PTR	0x00	rw	<p>指向 TB 帧槽的指针 (Pointer to a TB frame slot) 0x00: 指针指向 PTB Others: 指针指向 STB 的一个槽</p>

位 15: 12 TEW	0x0	rw	TB PTR 指向的帧槽通过 TBUF 寄存器进行读/写操作。只有在 TSFF=0 时，写操作有效。配置 TBF 为 1 标记槽空间满，配置 TBE 为 1 标记槽空间空。 在 TTCAN 模式，TBSEL 和 TSNEXT 配置不可用。 TB PTR 只能指向硬件存在的缓存槽空间，未被使用 TB PTR 位固定为 0. TB PTR 指向的槽空间最大限制为 PTB 和 63 个 STB 槽，TTCAN 模式下，不能使用更多的槽空间。 如果 TB PTR 太大，指向了一个不可用的槽空间，则 TBF 和 TBE 被自动复位，不会进行任何动作。
位 11 保留	0x0	resd	发送使能窗口 (Transmit enable window) 对于单次传输触发，周期时间最多有 16 个时间刻度，指定帧传输的时刻。TWE+1 定义时间刻度的数量。 TEW=0 是可用的配置，指定发送允许窗口包含 1 个时间刻度。
位 10: 8 TTTYPE	0x00	rw	保持默认值 触发类型 (Trigger type) 000: 立即触发，对应于立即传输 001: 时间触发，对应于接收触发 010: 单次发送触发，对应于专用时间窗口 011: 发送起始触发，对应于合并的仲裁窗口 100: 发送结束触发，对应于合并的仲裁窗口 其它：不可用 触发的时间由 TT_TRIG 定义，TTPTR 选择发送触发的 TB 槽。
位 7: 6 保留	0x0	resd	保留默认值 发送触发 TB 槽指针 (Transmit trigger TB slot pointer) 如果 TT PTR 太大，指向不可用的槽空间，则 TEIF 被置位，只有当写入寄存器 TT_TRIG_1 有效时，新的触发才会被激活。 如果 TT PTR 指向一个空的槽空间，则当到达触发时间时，TEIF 被置位。
位 5: 0 TT PTR	0x00	rw	

21.7.12 CAN TTCAN: 触发时间 (TTTRIG)

域	简称	复位值	类型	功能
位 31: 16 TT_WTRIG	0xFFFF	rw	窗口触发时间 (Watch trigger time) TT_WTRIG[15: 0] 定义窗口触发的周期时间。初始的窗口触发采用最大的周期时间 0Xffff.	
位 15: 0 TT_TRIG	0x0000	rw	触发时间 (Trigger time) TT_TRIG[15: 0] 定义触发的周期时间。对于发送触发，最早发送节点发送合适帧的 SOF 时间点是 TT_TRIG+1。	

注：对 TT_TRIG 位 (TTTRIG[15:0]) 位的有效写操作会启动触发配置到 CAN 时钟域的数据同步，同时激活触发器，如果触发器被激活，寄存器 TTCFG[31:24] 被锁定，TT_TRIG 位处于激活状态，当到达触发时间 (TTIE 使能时 TTIF 被置位) 或者发生错误时 (TEIF 被置位)，寄存器写入锁定失效。

对 TT_WTRIG 位 (TTTRIG[31:16]) 位的有效写操作会启动触发配置到 CAN 时钟域的数据同步，同时激活触发器，如果触发器被激活，寄存器 TT_WTRIG 位 (TTTRIG[31:16]) 写入被锁定，为激活一次新的触发，对 TT_WTRIG 位 (TTTRIG[31:16]) 执行一次写操作是必要的。

21.7.13 CAN 接收过滤器控制器 (ACFCTRL)

域	简称	复位值	类型	功能
位 31: 16 AE_x	0x01	rw	接收过滤器使能 (Acceptance filter enable) 1: 接收过滤器使能 0: 接收过滤器禁用 每个接收过滤器 (ACFC/ACFM) 可以被单独使能或禁用。过滤器编号 0 在硬件复位后默认使能。	

位 15: 4	保留	0x0	resd	禁用过滤器将拒收一帧数据。只有在使能过滤器且与 ACFC/ACFM 配置相匹配的情况下，帧数据才能被接收。 保持默认值。
位 3: 0	ACFADR	0x00	rw	接收过滤器地址 (Acceptance filter address) ACFADR 指向一个特定的接收过滤器，选中的过滤器使用寄存器 ACFC 和 ACFM 进行访问。 当 ACFADR>15 时，该值无意义，并被自动按值 15 处理。

21.7.14 CAN过滤器编码ID区（FCID）

参考表 21-1 标识符 (ID)

域	简称	复位值	类型	功能
位 31: 0	FCID_x	0x	rw	过滤器编码 ID 区 (Filter code identifier) 寄存器配置值跟总线上接收到的数据对应的电平完全一致。

21.7.15 CAN过滤器编码格式区（FCFMT）

参考表 21-1 格式 (FMT)

域	简称	复位值	类型	功能
位 31: 0	FCFMT_x	0x	rw	过滤器编码格式区 (Filter code format) 寄存器配置值跟总线上接收到的数据对应的电平完全一致。

21.7.16 CAN过滤器编码类型区（FCTYP）

参考表 21-1 类型 (TYP)

域	简称	复位值	类型	功能
位 31: 0	FCTYP_x	0x	rw	过滤器编码类型区 (Filter code type) 寄存器配置值跟总线上接收到的数据对应的电平完全一致。

21.7.17 CAN过滤器掩码ID区（FMID）

参考表 21-1 标识符 (ID)

域	简称	复位值	类型	功能
位 31: 0	FMID_x	0x	rw	过滤器掩码 ID 区 (Filter mask identifier) 0: 接收标识符的过滤检查被使能 1: 接收标识符的过滤检查被禁用

21.7.18 CAN过滤器掩码格式区（FMFMT）

参考表 21-1 格式 (FMT)

域	简称	复位值	类型	功能
位 31: 0	FMFMT_x	0x	rw	过滤器掩码格式区 (Filter mask format) 0: 接收标识符的过滤检查被使能 1: 接收标识符的过滤检查被禁用

21.7.19 CAN过滤器掩码类型区（FMTYP）

参考表 21-1 类型 (TYP)

域	简称	复位值	类型	功能
位 31: 0	FMTYP_x	0x	rw	过滤器掩码类型区 (Filter mask type) 0: 接收标识符的过滤检查被使能 1: 接收标识符的过滤检查被禁用

21.7.20 CAN保留寄存器1 (Res 1)

域	简称	复位值	类型	功能
位 31: 0	保留	0x	rw	因只对于 CAN XL 帧格式可用, 且本型号不支持, 须软件写 0xFFFFFFFF。

21.7.21 CAN接收缓冲区ID区 (RBID)

参考表 21-1 标识符 (ID)

域	简称	复位值	类型	功能
位 31: 0	RBID	0x	rw	接收缓冲区 ID 区 (RBUF Identifier)

21.7.22 CAN接收缓冲区格式区 (RBFMT)

参考表 21-1 格式 (FMT)

域	简称	复位值	类型	功能
位 31: 0	RBFMT	0x	rw	接收缓冲区格式区 (RBUF Format)

21.7.23 CAN接收缓冲区类型区 (RBTYP)

参考表 21-1 类型 (TYP)

域	简称	复位值	类型	功能
位 31: 0	RBTYP	0x	rw	接收缓冲区类型区 (RBUF Type)

21.7.24 CAN接收缓冲区数据区 (RBDATn)

域	简称	复位值	类型	功能
位 31: 0	RBDATn	0x	rw	接收缓冲区数据区 (RBUF Data n)

21.7.25 CAN接收缓冲区 CiA 603 接收时间戳1 (RBCIA1)

域	简称	复位值	类型	功能
位 31: 0	RBCIA1	0x	rw	接收缓冲区 CIA 603 接收时间戳 1 (RBUF CiA 603)

21.7.26 CAN接收缓冲区 CiA 603 接收时间戳2 (RBCIA2)

域	简称	复位值	类型	功能
位 31: 0	RBCIA2	0x	rw	接收缓冲区 CIA 603 接收时间戳 2 (RBUF CiA 603)

21.7.27 CAN接收缓冲区 TTCAN 周期时间 (RBTTCAN)

域	简称	复位值	类型	功能
位 15: 0	RBTTCAN	0x	rw	接收缓冲区 TTCAN 周期时间 (RBTTCAN)

21.7.28 CAN发送缓冲区ID区 (TBID)

参考表 21-1 标识符 (ID)

域	简称	复位值	类型	功能
位 31: 0	TBID	0x	rw	发送缓冲区 ID 区 (TBUF Identifier)

21.7.29 CAN发送缓冲区格式区 (TBFMT)

参考表 21-1 格式 (FMT)

域	简称	复位值	类型	功能
位 31: 0	TBFMT	0x	rw	发送缓冲区格式区 (TBUF Format)

21.7.30 CAN发送缓冲区类型区 (TBTYP)

参考表 21-1 类型 (TYP)

域	简称	复位值	类型	功能
位 31: 0	TBTYP	0x	rw	发送缓冲区类型区 (TBUF Type)

21.7.31 CAN保留寄存器2 (Res 2)

域	简称	复位值	类型	功能
位 31: 0	保留	0x	rw	因只对于 CAN XL 帧格式可用, 且本型号不支持, 须软件写 0x00000000。

21.7.32 CAN发送缓冲区数据区 (TBDATn)

域	简称	复位值	类型	功能
位 31: 0	TBDATn	0x	rw	发送缓冲区数据区 (TBUF Data n)

21.7.33 CAN LLC帧计算输入 (LLCFORMAT)

域	简称	复位值	类型	功能
位 31: 0	LLCFORMAT	0x00	rw	对 LLCFORMAT 寄存器进行写操作, 硬件会自动计算一帧 LLC 数据的大小, 并存储至 LLCSIZE 中。 LLCFORMAT 的布局与 LLC 帧的格式字的布局是完全相同的。 只有 LLCSIZE 计算相关的位才会存储至 LLCFORMAT 中, 其它的位固定为 0。相关的位是: DLC, FDF, 和 RMF。

21.7.34 CAN LLC帧计算输出 (LLCSIZE)

域	简称	复位值	类型	功能
位 31: 16	LLCAOT	0x10	ro	LLC 时间戳偏移地址 (LLC address offset of time-stamps) 基于 LLCFORMAT 中的值, LLCAOUT 提供 LLC 帧的时间戳偏移地址 (值 T)。 值 T 字对齐。 例如: 对于负载为 1 个字节的帧数据, 值 T: $T=16+4=20$ 。
位 15: 0	LLCPBYTES	0x0	ro	LLC 帧负载字节数 (LLC frame payload bytes) 基于 LLCFORMAT 中的值, LLCPBYTES 提供 LLC 帧数据的负载字节数。

21.7.35 CAN中断使能寄存器 (INTEN)

域	简称	复位值	类型	功能
位 31: 14	保留	0x00	resd	请保持默认值
位 13	WTIEN	0x0	rw	WTIF 标志输出中断使能， 默认关闭。
位 12	TEIEN	0x0	rw	TEIF 标志输出中断使能， 默认关闭。
位 11	TTIEN	0x0	rw	TTIF 标志输出中断使能， 默认关闭。
位 10	EPIEN	0x0	rw	EPIF 标志输出中断使能， 默认关闭。
位 9	ALIEN	0x0	rw	ALIF 标志输出中断使能， 默认关闭。
位 8	BEIEN	0x0	rw	BEIF 标志输出中断使能， 默认关闭。
位 7	RIEN	0x0	rw	RIF 标志输出中断使能， 默认关闭。
位 6	ROIEN	0x0	rw	ROIF 标志输出中断使能， 默认关闭。
位 5	RFIEN	0x0	rw	RFIF 标志输出中断使能， 默认关闭。
位 4	RAFIEN	0x0	rw	RAFIF 标志输出中断使能， 默认关闭。
位 3	TPIEN	0x0	rw	TPIF 标志输出中断使能， 默认关闭。
位 2	TSIEN	0x0	rw	TSIF 标志输出中断使能， 默认关闭。
位 1	EIEN	0x0	rw	EIF 标志输出中断使能， 默认关闭。
位 0	AIEN	0x0	rw	AIF 标志输出中断使能， 默认关闭。

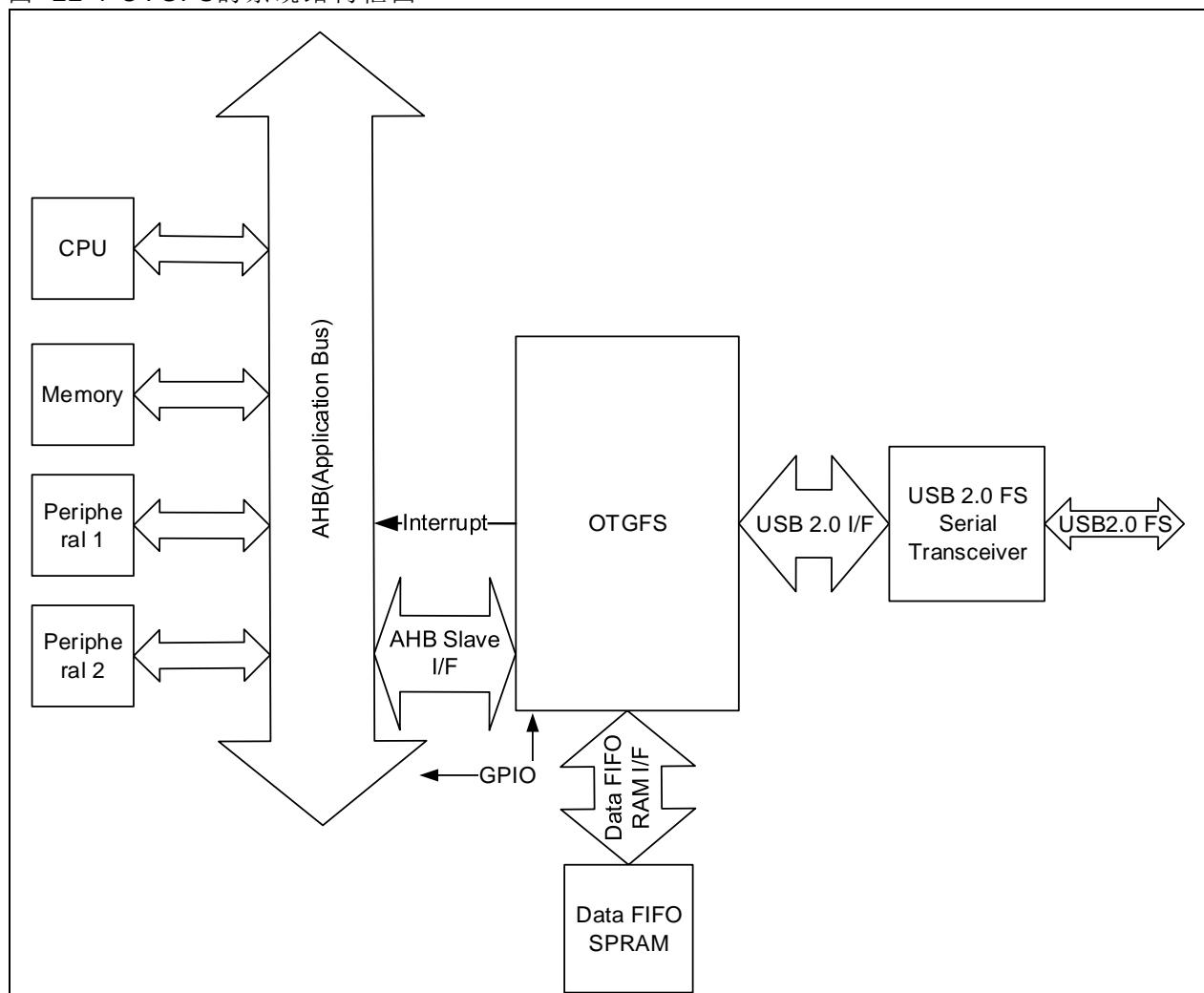
22 USB OTG 全速 (OTGFS)

OTGFS 为全速双重角色设备，符合 Universal Serial Bus Specification, Revision2.0 标准。

22.1 OTGFS系统结构框图

下图为 OTGFS 的系统结构框图，OTGFS 模块挂载到 AHB 上，拥有独立的 1280Byte SRAM。

图 22-1 OTGFS的系统结构框图



22.2 OTGFS 功能概述

OTGFS 在主机模式下支持 FS (12Mb/s) 和 LS (1.5Mb/s)，在设备模式下支持 FS (12Mb/s)。

OTGFS 模块由 OTGFS controller、内置物理层 (PHY) 以及独立 1280 字节 SRAM 组成。

OTGFS 支持控制传输、大容量、中断和同步传输。

OTGFS 是 USB 全速双角色设备 (DRD) 控制器，由 ID 线的状态控制 OTGFS 为主机还是设备：当 ID 线浮空时，OTGFS 作为设备，当 ID 线接地时，OTGFS 为主机。OTG PHY 内置了 1.5KΩ 上拉电阻和 15KΩ 下拉电阻，以满足双角色设备需要。

OTGFS 作为设备时，支持 1 个双向控制端点，7 个 IN 端点、7 个 OUT 端点；做为主机时，支持 16 个主机通道。

OTGFS 支持 SOF 脉冲功能和 OE 脉冲功能：发生 SOF 包时产生 SOF 脉冲，SOF 脉冲可输出到芯片管脚和定时器 2；输出数据时产生 OE 脉冲，OE 脉冲可输出到芯片管脚。

OTGFS 作为设备时，为所有 OUT 端点分配一个 FIFO 缓存，为每个 IN 端点分配各自分配一个 FIFO 缓存；OTGFS 作为主机时，为所有的接收通道分配了一个统一的接收 FIFO，为所有非周期性发送通道分配一个统一的发送 FIFO，为所有周期性发送通道分配一个统一的发送 FIFO。

OTGFS 支持挂起模式，在时钟门控寄存器(OTGFS_PCGCCTL)的 STOPPCLK 位置位后，当连续 3ms 未收到总线信号时，OTGFS 会进入挂起模式；OTGFS 还可以通过配置 OTGFS 通用控制器配置寄存器(OTGFS_GCCFG)的 LP_MODE 位关闭 PHY 的接收功能，从而达到降低功耗效果。

22.3 OTGFS时钟与管脚配置

22.3.1 OTGFS时钟配置

芯片为 OTGFS 提供两个时钟：一个 $48MHz \pm 0.25\%$ 的时钟供 OTGFS 控制器使用，一个 AHB 总线时钟供 OTGFS 控制寄存器访问使用。

USB 全速设备总线速度标准为 $12Mb/s \pm 0.25\%$ 。因此需要给 OTGFS 提供 $48MHz \pm 0.25\%$ 的时钟频率用于 USB 总线采样。

OTGFS 48M 时钟有两种配置来源：

- HICK 48M

使用 HICK 48M 时钟作为 USB 控制时钟时，建议开启 ACC 功能。

- 通过 PLLU

PLLU 需要配置为 $48MHz$ 时钟输出（参考时钟配置寄存器（CRM_PLLCFG））。

注意：使用 OTGFS 时，AHB 时钟频率必须大于 $30Mhz$

22.3.2 OTGFS管脚配置

OTGFS 的输入输出均与 GPIO 复用，当满足以下条件时 GPIO 被用于 OTGFS 功能。

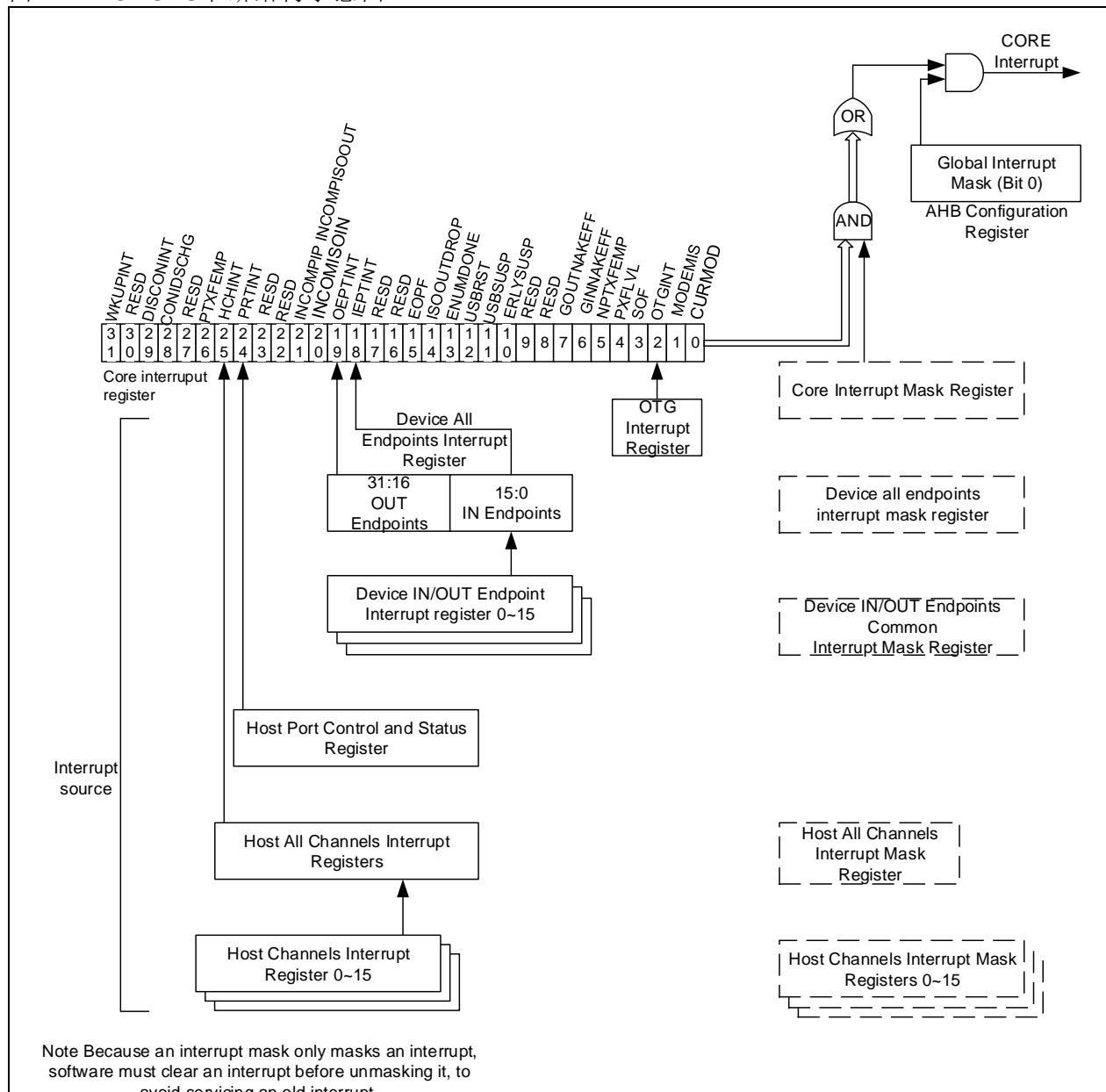
表 22-1 OTGFS输入/输出引脚

信号名称	GPIO	条件
OTGFS_SOF	PA8	在 CRM 中使能 OTGFS 模块，且 PA8 复用功能寄存器配置为 0xa
OTGFS_VBUS	PA9	在 CRM 中使能 OTGFS 模块，且 PA9 复用功能寄存器配置为 0xa
OTGFS_ID	PA10	在 CRM 中使能 OTGFS 模块，且 PA10 复用功能寄存器配置为 0xa
OTGFS_D-	PA11	在 CRM 中使能 OTGFS 模块，且 PWRDOWN 位设置为 1
OTGFS_D+	PA12	在 CRM 中使能 OTGFS 模块，且 PWRDOWN 位设置为 1
	PA4	在 CRM 中使能 OTGFS 模块，且 PA4 复用功能寄存器配置为 0xd
OTGFS_OE	PA13	在 CRM 中使能 OTGFS 模块，且 PA13 复用功能寄存器配置为 0xa
	PC9	在 CRM 中使能 OTGFS 模块，且 PC9 复用功能寄存器配置为 0xb

22.4 OTGFS中断

OTGFS 中断结构示意图如下图所示，功能详见 OTGFS 中断寄存器(OTGFS_GINTSTS)和 OTGFS 中断屏蔽寄存器(OTGFS_GINTMSK)。

图 22-2 OTGFS 中断结构示意图



22.5 OTGFS 功能操作

22.5.1 OTGFS 初始化

如果上电时线缆已连接，那么控制器中断寄存器的当前操作模式位（CURMOD）指示当前工作模式。当连接 A 类插头时，OTGFS 控制器工作在主机模式；当连接 B 类插头时，控制器工作在设备模式。

本节介绍了上电后控制器的初始化操作。无论控制器处于主机模式还是设备模式，应用程序都必须遵循初始化顺序。所有的控制器全局寄存器都须按照控制器的配置进行初始化操作。

1、在全局 AHB 配置寄存器中设置以下位域：

- 全局中断屏蔽位 = 0x1
- 非周期性发送 FIFO 空级别
- 周期性发送 FIFO 空级别

2、设置全局中断屏蔽寄存器的下列位域：

- OTGFS_GINTMSK.RXFLVLMSK = 0x0

3、设置 OTGFS_USB 配置寄存器(OTGFS_GUSBCFG)的下列位域：

- 全速超时标准位
- USB 周转时间位

4、软件必须解除 OTGFS 中断屏蔽寄存器(OTGFS_GINTMSK)中的下列位的中断屏蔽：

- OTG 中断屏蔽
- 模式不匹配中断屏蔽

5、软件通过读取当前运行模式寄存器(OTGFS_GINTSTS.CURMOD)位来确定 OTGFS 控制器是处于主机模式还是设备模式。

22.5.2 OTGFS FIFO 配置

22.5.2.1 设备模式

在上电复位或 USB 复位时，需要进行动态 FIFO 重新分配。在设备模式下，应用程序在更改 FIFO 数据的 SRAM 分配之前，必须先确保满足下列条件：

- OTGFS_DIEPCTLx/ OTGFS_DOEPCTLx.EPENA = 0x0
- OTGFS_DIEPCTLx/ OTGFS_DOEPCTLx.NAKSTS = 0x1

通过发送 FIFO 编号寄存器 (OTGFS_GRSTCTL.TXFNUM) 位刷新控制器中的发送 FIFO。关于如何刷新发送 FIFO，详见“刷新控制器发送 FIFO”章节。

在为 FIFO 分配 SRAM 空间时，必须注意以下几点：

(1) 接收 FIFO SRAM 分配

- 用于 SETUP 包的 SRAM：必须预留 13 个 WORDs 才能接收控制端点发出的 1 个 SETUP 包，这些位置是预留给 SETUP 写数据的，控制器并不会占用。
- 为全局 OUT NAK 预留 1 个 WORD
- 状态信息与每次收到的包一起写入到 FIFO。因而，必须分配至少一个(包最大长度 /4)+1 这样的空间来接收数据包。如果使能了多个同步端点，那么至少需要分配两个(包最大长度 /4)+1 这样的空间来接收连续的数据包。一般情况下，建议分配两个(包最大长度 /4)+1 的空间以确保当前一个数据包正传输给 AHB 时，USB 可以接收到下一个数据包。如果 AHB 延迟较长，必须配置足够的空间来接收多个数据包，以防止丢失同步数据包。
- 传输完成的状态信息，连同每个端点的最后一个数据包，一起被推送到 FIFO。
- 必须为每个端点的禁止状态位预留一个位置。
- 通常建议为每个 OUT 端点配置两个 WORDs。

(2) 发送 FIFO SRAM 分配

每个 IN 端点发送 FIFO 所需的最小 SRAM 空间就是该 IN 端点的最大数据包长度。发送 IN 端点 FIFO 分配的空间越多，USB 性能越好，并且还能避开 AHB 线上的延迟。

表 22-2 OTGFS发送 FIFO SRAM 分配表

FIFO 名称	SRAM 大小
接收 FIFO	rx_fifo_size, 包含 setup 包, OUT 端点控制信息以及数据 OUT 包。
发送 FIFO 0	tx_fifo_size[0]
发送 FIFO 1	tx_fifo_size[1]
发送 FIFO 2	tx_fifo_size[2]
.....
发送 FIFO i	tx_fifo_size[i]

根据以上信息配置下列寄存器：

1. OTGFS 接收 FIFO 长度寄存器(OTGFS_GRXFSIZ)
 - OTGFS_GRXFSIZ.RXFDEP = rx_fifo_size;
2. 端点 0 TX FIFO 长度寄存器(OTGFS_DIEPTXF0)
 - OTGFS_DIEPTXF0.INEPT0TXDEP = tx_fifo_size[0];
 - OTGFS_DIEPTXF0.INEPT0TXSTADDR = rx_fifo_size;
3. 设备 IN 端点发送 FIFO#1 长度寄存器(OTGFS_DIEPTXF1)
 - OTGFS_DIEPTXF1.INEPTXFSTADDR = OTGFS_DIEPTXF0.INEPT0TXSTADDR + tx_fifo_size[0];
4. 设备 IN 端点发送 FIFO#2 长度寄存器(OTGFS_DIEPTXF2)
 - OTGFS_DIEPTXF2.INEPTXFSTADDR = OTGFS_DIEPTXF1.INEPTXFSTADDR + tx_fifo_size[1];
5. 设备 IN 端点发送 FIFO#i 长度寄存器(OTGFS_DIEPTXF*i*)
 - OTGFS_DIEPTXF*i*.INEPTXFSTADDR = OTGFS_DIEPTXF*i*-1.INEPTXFSTADDR + tx_fifo_size[i-1];
6. 完成 SRAM 分配后必须刷新发送 FIFO 和接收 FIFO，以确保 FIFO 正常运行。
 - OTGFS_GRSTCTL.TXFNUM = 0x10
 - OTGFS_GRSTCTL.TXFFLSH = 0x1
 - OTGFS_GRSTCTL.RXFFLSH = 0x1

应用程序必须要等到 TXFFLSH 位和 RXFFLSH 位清除后才能在控制器上执行其它操作。

22.5.2.2 主机模式

在主机模式下，应用程序在更改 FIFO 数据的 SRAM 分配之前，必须先确认下列信息：

- 所有通道已禁用
- 所有 FIFO 为空

重新分配完 FIFO 数据的 SRAM 后，应用程序必须通过发送 FIFO 编号寄存器 (OTGFS_GRSTCTL.TXFNUM) 刷新位来刷新控制器中的所有 FIFO。

在完成重新分配后，必须通过刷新来复位 FIFO 中的指针以确保 FIFO 正常运行。关于刷新发送 FIFO 的详细信息，请参考“刷新控制器中的发送 FIFO”。

(1) 接收 FIFO SRAM 分配

状态信息与每次接收的数据包一起写入 FIFO。因此，必须分配至少一个(包最大长度 / 4)+2 这样的空间来接收数据包。如果使能了多个同步端点，那么至少需要分配两个(包最大长度 / 4)+2 这样的空间来接收背靠背数据包。一般情况下，建议分配两个(包最大长度 / 4)+2 的空间以便当前一个数据包正传输给 AHB 时，USB 可以接收到下一个数据包。如果 AHB 延迟较长，必须配置足够的空间来接收多个数据包。传输完成状态信息和通道中止信息会跟着每个主机通道的最后一个数据包一起推送给 FIFO。为此，必须分配两个 WORDs。

(2) 发送 FIFO SRAM 分配

主机非周期性发送 FIFO 所需要的最小 SRAM 空间就是所有非周期性 OUT 通道的最大包长度。发送非周期性 FIFO 的空间越多，USB 性能越好，并且还能避开 AHB 线上的延迟。通常，建议分配两个最大包长度，以便当前一个数据包正传输给 USB 时，AHB 可以接收到下一个数据包。如果 AHB 延迟较长，必须配置足够的空间来缓冲多个数据包。

主机周期性发送 FIFO 所需要的最小 SRAM 空间就是所有周期性 OUT 通道的最大包长度。

(3) 内部存储空间分配

表 22-3 OTGFS内部存储空间分配表

FIFO Name	Data SRAM Size

接收数据 FIFO	rx_fifo_size
非周期性发送 FIFO	tx_fifo_size[0]
周期性发送 FIFO	tx_fifo_size[1]

根据这些信息来配置下列寄存器：

1. OTGFS 接收 FIFO 长度寄存器(OTGFS_GRXFSIZ)
 - OTGFS_GRXFSIZ.RXFDEP = rx_fifo_size;
2. OTGFS 非周期性 TX FIFO 长度寄存器(OTGFS_GNPTXFSIZ)
 - OTGFS_GNPTXFSIZ.NPTXFDEP = tx_fifo_size[0];
 - OTGFS_GNPTXFSIZ.NPTXFSTADDR = rx_fifo_size;
3. OTGFS 主机周期性发送 FIFO 长度寄存器(OTGFS_HPTXFSIZ)
 - OTGFS_HPTXFSIZ.PTXFSIZE = tx_fifo_size[1];
 - OTGFS_HPTXFSIZ.PTXFSTADDR = OTGFS_GNPTXFSIZ.NPTXFSTADDR + tx_fifo_size[0];
4. 在完成 SRAM 分配之后，必须刷新发送 FIFO 和接收 FIFO 以确保 FIFO 正常运行。
 - OTGFS_GRSTCTL.TXFNUM = 0x10
 - OTGFS_GRSTCTL.TXFFLSH = 0x1
 - OTGFS_GRSTCTL.RXFFLSH = 0x1
 - 应用程序必须等到 TXFFLSH 位和 RXFFLSH 位清除之后才能执行其它操作。

22.5.2.3 刷新控制器发送FIFO

应用程序通过 OTGFS_GRSTCTL.TXFFLSH 来刷新控制器中的所有发送 FIFO，流程如下：

- 如果该位已清除，则置设置全局 IN NAK 寄存器（OTGFS_DCTL.SGNPINNAK）为 0x1。NAK 有效中断置位表示控制器未读取 FIFO。
- 等待 OTGFS_GINTSTS.GINNAKEFF = 0x1，表示针对所有 IN 端点的 NAK 设置已生效。
- 轮询 AHB 主机空闲寄存器（OTGFS_GRSTCTL.AHBIDLE）直到其置为 1，AHBIDLE = H 表示控制器没有写入 FIFO。
- 确认 OTGFS_GRSTCTL.TXFFLSH = 0x0。如果置 0，则将你需要刷新的发送 FIFO 编号写入到发送 FIFO 编号寄存器（OTGFS_GRSTCTL.TXFNUM）。
- 设置 OTGFS_GRSTCTL.TXFFLSH = 0x1，然后等待清除。
- 置起清除全局 IN NAK 寄存器（OTGFS_DCTL.CGNPINNAK）位。

22.5.3 OTGFS主机模式

在 OTGFS 作为主机时，控制器内部不能为 VBUS 提供 5V 电压源，需要外部电压泵持续为 VBUS 供电。

22.5.3.1 主机初始化

应用程序必须遵循以下步骤来初始化控制器：

1. 将主机端口中断屏蔽寄存器（OTGFS_GINTMSK.PRTINTMSK）设置为解除中断屏蔽。
2. 设置 OTGFS 主机模式配置寄存器(OTGFS_HCFG)，选择全速主机或高速主机模式。
3. 设置 OTGFS_HPRT.PRTPOWER 位为 0x1，驱动 USB 线上的 VBUS 供电。
4. 等待端口连接检测寄存器（OTGFS_HPRT0.PRTCONDET）中断，该中断表示设备连接到端口。
5. 设置端口复位寄存器（OTGFS_HPRT.PRTRST）位为 0x1，发起复位操作。
6. 等待至少 10 ms，确保完成复位。
7. 设置端口复位寄存器（OTGFS_HPRT.PRTRST）位为 0x0。
8. 等待端口使能/禁止状态改变寄存器（OTGFS_HPRT.PRTEENCHNG）中断。
9. 读取端口速度寄存器（OTGFS_HPRT.PRTSPD）位，获取枚举速度。
10. 根据已选择的 PHY 时钟值来配置 HFIR 寄存器。
11. 设置 OTGFS 接收 FIFO 长度寄存器(OTGFS_GRXFSIZ)，选择接收 FIFO 的长度。
12. 设置 OTGFS 非周期性 TX FIFO 长度寄存器(OTGFS_GNPTXFSIZ)，选择非周期发送 FIFO 的起始地址和长度。
13. 设置 OTGFS 主机周期性发送 FIFO 长度寄存器(OTGFS_HPTXFSIZ)，选择周期性发送 FIFO 的起始地址和和长度。

为了与设备通信，应用程序必须按照“OTGFS 通道初始化”的要求使能和初始化至少一个通道。

22.5.3.2 OTGFS 通道初始化

为了与设备通信，应用程序必须使能和初始化至少一个通道。应用程序可遵循下列步骤来使能和初始化通道：

1. 设置 OTGFS 中断屏蔽寄存器(OTGFS_GINTMSK)解除下列中断屏蔽：
 - 用于 OUT 传输的非周期性发送 FIFO 空
 - 用于 OUT 传输的非周期性发送 FIFO 半空
2. 设置 OTGFS 主机所有通道中断屏蔽寄存器(OTGFS_HAINTMSK)解除所选通道的中断屏蔽。
3. 设置 OTGFS 主机通道 x 中断屏蔽寄存器(OTGFS_HCINTMSKx)解除主机通道中断寄存器中与传输相关的中断屏蔽。
4. 为所选通道的 OTGFS 主机通道 x 传输长度寄存器(OTGFS_HCTSIZx)配置传输总长度（以字节为单位），以及期望接收的包数目，包括短数据包。应用程序必须根据初始数据 PID (用于首个 OUT 传输的 PID，或者期望从首个 IN 传输收到的数据 PID) 来配置 PID 位。
5. 设置传输长度，确保该通道的传输长度是最大包长度的倍数。
6. 根据设备端点特性，比如类型，速度，方向等来配置所选通道的 OTGFS 主机通道 x 特性寄存器(OTGFS_HCCHARx)（只有当应用程序准备好传输或接收数据包时，才能通过设置通道使能位为 1 来使能通道）。

22.5.3.3 中止通道

应用程序可以通过将 OTGFS 主机通道 x 特性寄存器(OTGFS_HCCHARx)的通道禁止寄存器(HCCHARx.CHDIS) 和通道使能寄存器(OTGFS_HCCHARx.CHENA) 位设置为 0x1 来中止通道。这个操作将使能主机刷新已提交的请求（如果有的话）并生成“通道中止”中断。应用程序必须等待通道中止寄存器(OTGFS_HCINTx.CHHLTD) 中断生成之后，才能重新为其他传输事务分配通道。主机不会中断已经在 USB 线上启动的传输任务。

应用程序在中止通道前，必须确保非周期性请求队列（当中止非周期性通道时）或者周期性请求队列（当中止周期性通道时）中至少有一个可用空间。当请求队列满额时（在中止通道前），应用程序可以通过将 OTGFS 主机通道 x 特性寄存器(OTGFS_HCCHARx)中的通道禁止寄存器(HCCHARx.CHDIS) 位设置为 0x1 以及将通道使能寄存器(OTGFS_HCCHARx.CHENA) 位设置为 0 来刷新已提交的请求。

当请求队列里有传输输入时，控制器会触发 RXFLVL 中断。应用程序必须通过读取/弹出 OTGFS 状态读和 POP 寄存器(OTGFS_GRXSTSP)来生成“通道中止”中断。

应用程序应当在发生下列情况时中止通道：

- 在非周期性 IN 传输期间检测到了传输完成寄存器(OTGFS_HCINTx.XFERC) 中断；
- 当 IN 或 OUT 通道收到了 STALL 响应已收到中断寄存器(OTGFS_HCINTx.STALL)、传输错误寄存器(OTGFS_HCINTx.XACTERR)、Babble Error 寄存器(OTGFS_HCINTx.BBLEERR)、或者数据翻转错误寄存器(OTGFS_HCINTx.DTGLERR) 中断信号时。
- 当收到检测到断开事件产生中断寄存器(OTGFS_GINTSTS.DISCONINT)（设备断开）中断事件时。应用程序必须检查端口连接状态寄存器(OTGFS_HPRT.PRTCONSTS) 信号，这是因为当设备与主机断开时，端口连接状态寄存器(OTGFS_HPRT.PRTCONSTS) 将复位。应用程序必须启动软件复位以确保所有通道已清除。当设备重新连接时，主机必须启动 USB 复位。
- 当应用程序需要在正常传输结束之前中止传输时。

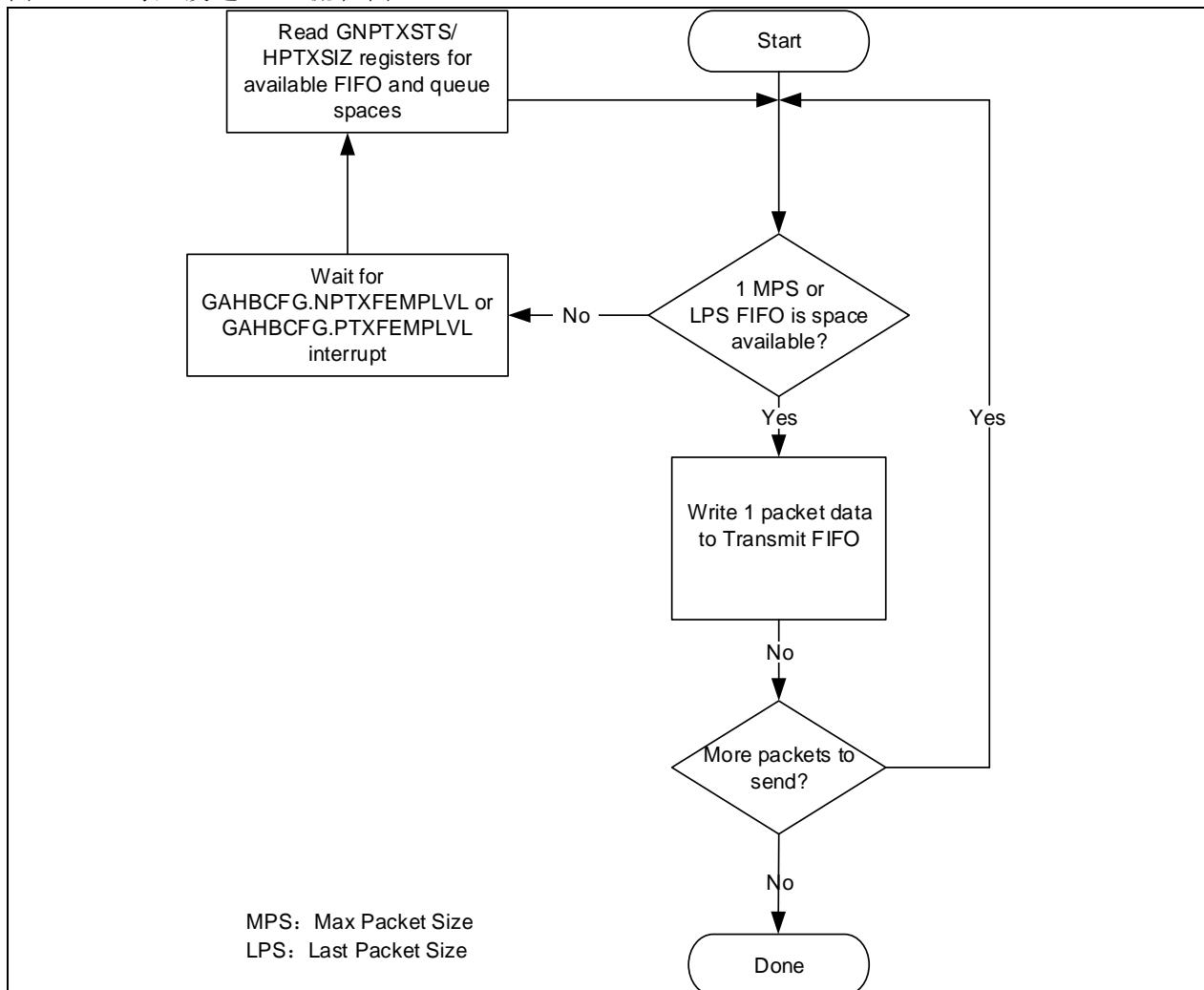
22.5.3.4 选择队列深度

在周期性硬件传输请求队列中支持最多 8 个中断和同步传输请求；在非周期性硬件传输请求队列中支持最多 8 个控制和大容量传输的传输请求。

- 写入发送 FIFO

下图显示写入发送 FIFO 的流程图。OTGFS 主机会在最后一个 WORD 写数据包时，自动将一个条目（OUT 请求）写入周期性/非周期性请求队列。在开始写入 FIFO 之前，应用程序必须保证周期性/非周期性请求队列里至少有一个可用空间。应用程序只能以 WORD 方式写入发送 FIFO。如果包长度没有对齐 WORD，那么应用程序必须填充数据位。OTGFS 主机根据所设置的最大包长度和传输长度来确定实际的包长度。

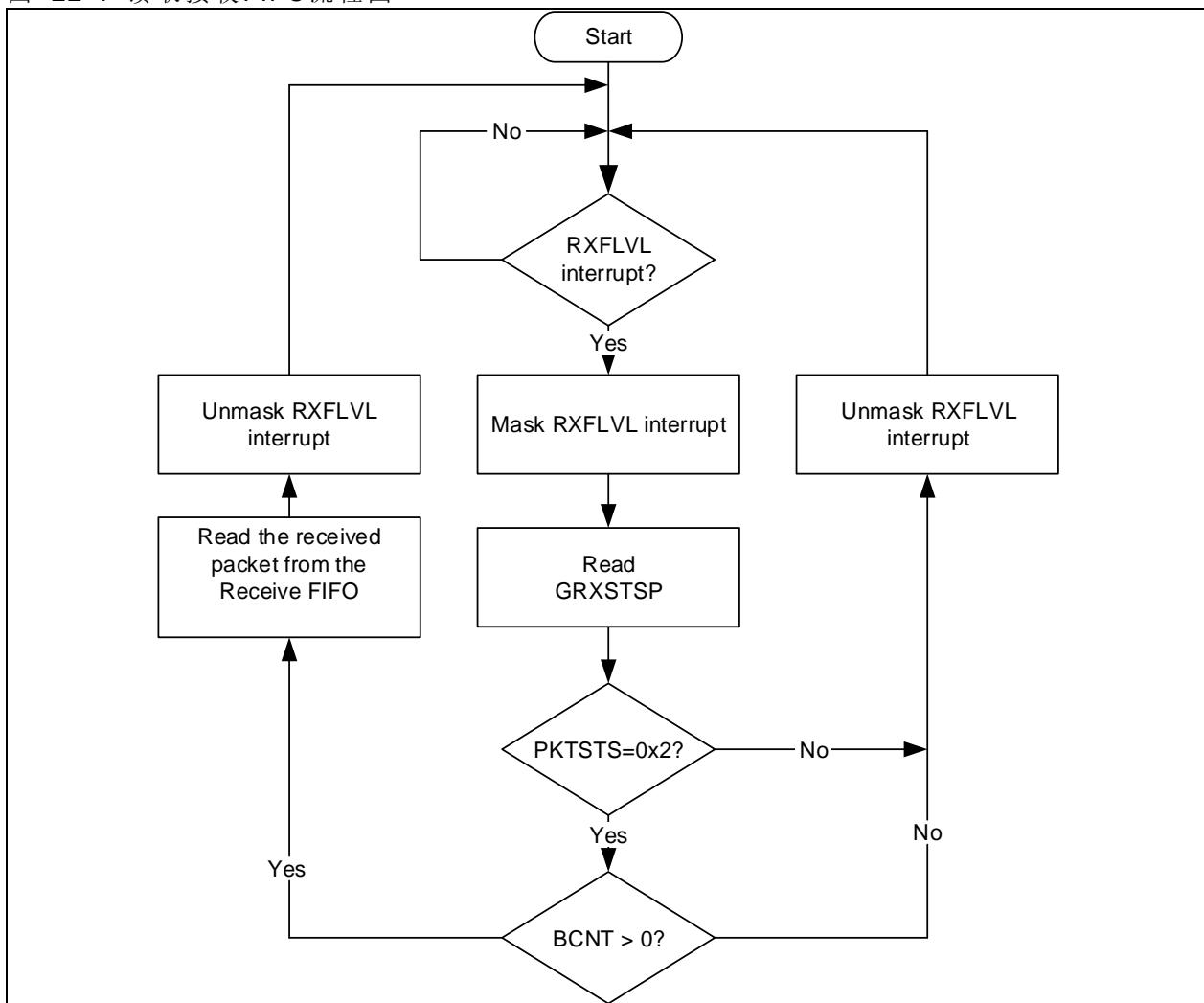
图 22-3 写入发送 FIFO 流程图



- 读取接收 FIFO

下图显示读取接收 FIFO 的流程图。应用程序需要忽略除了 IN 数据包(0x0010)以外的所有包状态。

图 22-4 读取接收 FIFO 流程图



22.5.3.5 特殊情况处理

(1) 处理 Babble 状况

OTGFS 处理两种 babble 情况：即包 babble 和端口 babble。如果设备发出的数据超过该通道的最大包长度时将发生包 babble。如果控制器在 EOF2 (即帧 2 末尾，非常接近 SOF) 时持续接收设备发出的数据时会产生端口 Babble。

当检测到包 babble 时，OTGFS 将停止写入接收缓冲区并等待包结束。当检测到包结束时，OTGFS 将清空已写入接收缓冲区的数据并生成 Babble 中断。

当检测到端口 babble 时，OTGFS 会清空接收 FIFO 并禁用该端口，然后，控制器生成“端口禁用中断”。一旦收到该中断，应用程序需要通过确认端口过流有效位寄存器 (HPRT.PRTOVRCACT) 信号来确定端口中断并不是因为过流引发的（这是导致端口禁用中断的另一个原因），然后进行软件复位。控制器在检测到端口 babble 信号时不再发送令牌。

(2) 处理设备断开

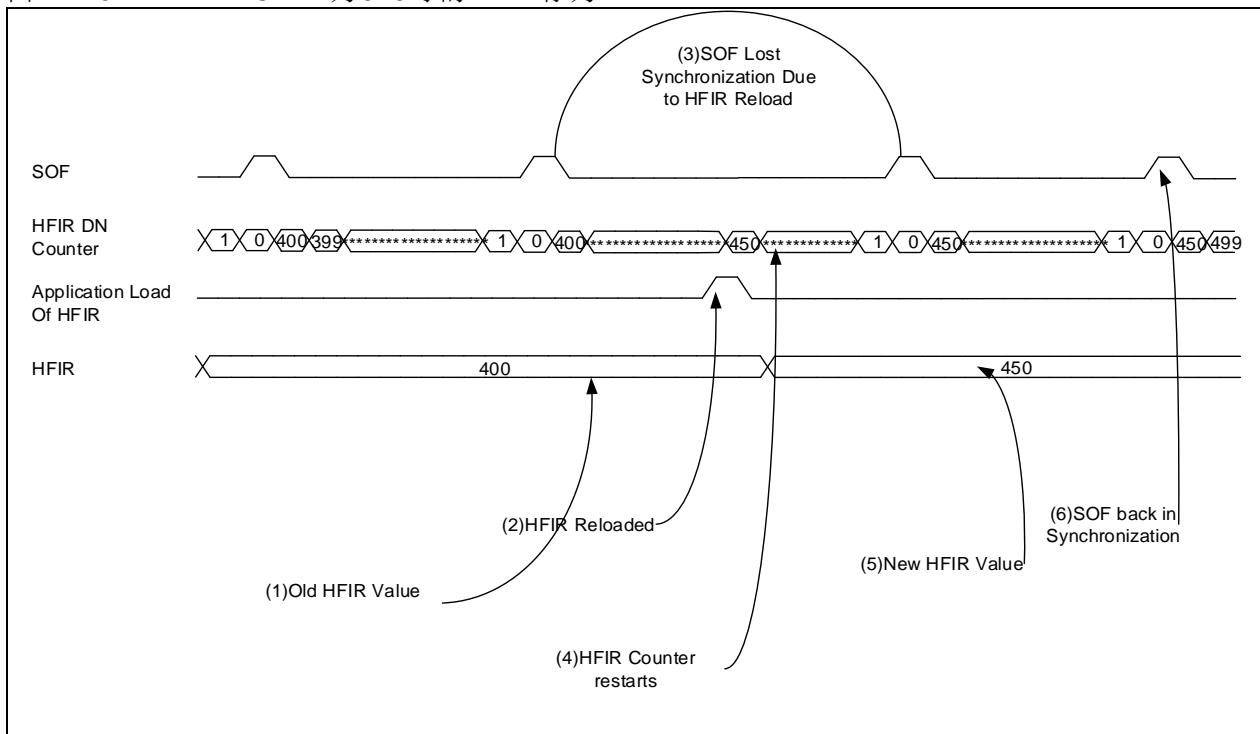
如果设备突然断开，则会生成一个检测到断开事件产生中断寄存器 (OTGFS_GINTSTS.DISCONINT) 中断。应用程序在收到该中断时，必须通过设置控制器软件复位寄存器 (OTGFS_GRSTCTL.CSFTRST) 来启动软件复位。

22.5.3.6 主机HFIR功能

主机帧间隔寄存器(HFIR)定义了两个连续 SOF(全速)或者 Keep-Alive 令牌之间的间隔。此位域包含了构成所需帧间隔的 PHY 时钟数，主要用于根据 PHY 时钟频率来调整 SOF 持续时间。

- 将重新加载控制寄存器 (OTGFS_HFIR.HFIRRLDCTRL) 设置为 0x0 时的 HFIR 行为
本章节使用下图所示波形描述了 OTGFS_HFIR.HFIRRLDCTRL 设置为 0x0 时控制器的行为。

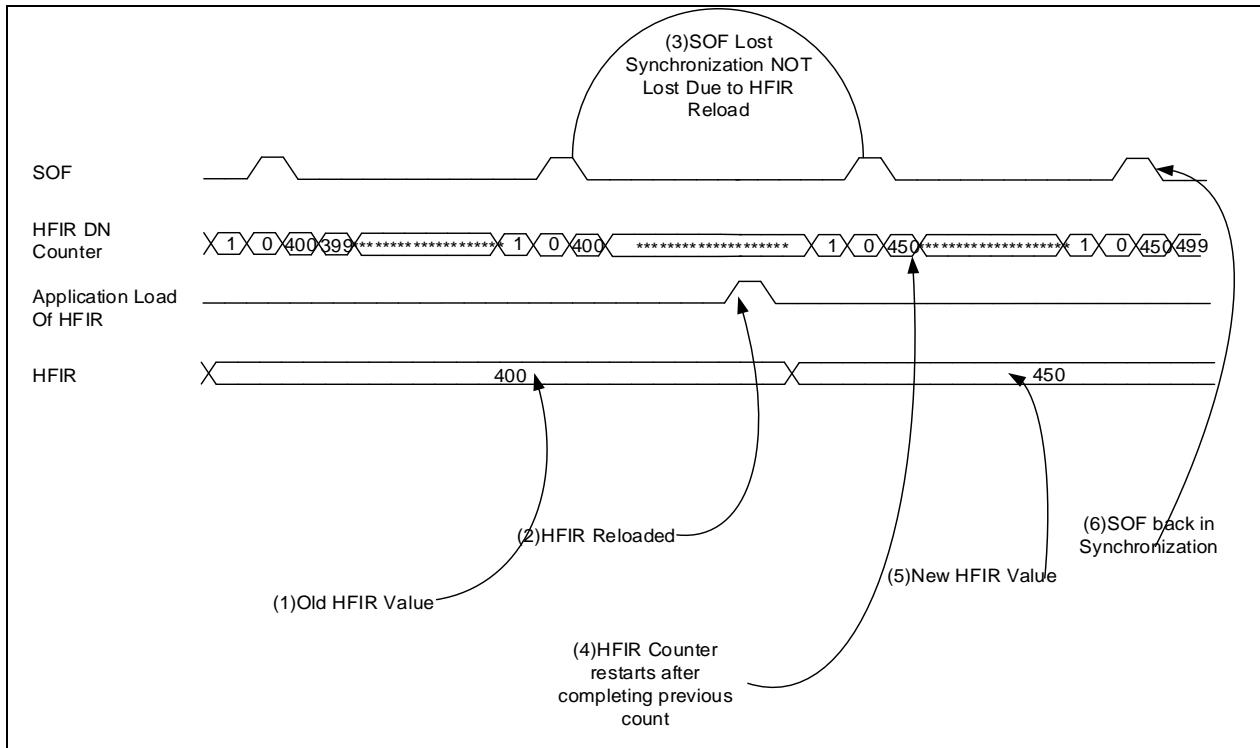
图 22-5 HFIRRLDCTRL 为 0x0 时的 HFIR 行为



步骤如下所示：

1. 上电复位后，显示应用程序当前设定的 HFIR 值
 2. 应用程序加载新值到 HFIR 寄存器
 3. 由于 HFIR 向下计数器重新加载，它会立即开始重新计数，从而丢失 SOF 同步
 4. 重启 HFIR 计数器
 5. HFIR 寄存器接收到新的设定值
 6. 使用 HFIR 新功能生成首个 SOF 后，又再次获得 SOF 同步。
- 当重新加载控制寄存器 (OTGFS_HFIR.HFIRRLDCTRL) 为 0x1 时的 HFIR 行为
本节使用下图所示波形介绍了当 HFIR.HFIRRLDCTRL = 0x1 时控制器的行为。

图 22-6 HFIRRLDCTRL 为 0x1 时的 HFIR 行为



所示的步骤如下：

1. 上电复位后，显示应用程序当前设定的 HFIR 值
 2. 应用程序加载新的 HFIR 值；HFIR 计数器不采用新值，而是继续计数直到计数器达到 0
 3. 当计数器在使用旧 HFIR 值计数到 0 时，生成 SOF
 4. HFIR 计数器采用新值
 5. 新的 HFIR 值生效
- 经过以上步骤 SOF 恢复同步。

22.5.3.7 初始化批量IN传输/控制IN传输

图 22-7 显示典型的批量 IN 传输/控制 IN 传输的操作流程，详见通道 2(ch_2)。假设：

- 应用程序正在尝试接收两个最大长度的数据包（传输长度为 1024 字节）
- 接收 FIFO 包含至少一个最大包长度的数据包以及每个数据包的两个状态 WORDs（对于全速传输有 72 字节）
- 非周期性请求队列深度为 4

(1) 普通批量 IN 传输和控制 IN 传输的操作流程

图 22-7 所示的操作顺序如下：

1. 初始化通道 2（按照“OTGFS 通道初始化”的要求）
2. 置起通道使能寄存器 (OTGFS_HCCHAR2.CHENA) 位，向非周期性请求队列写入一个 IN 请求
3. 控制器在完成当前 OUT 传输后会发送一个 IN 令牌
4. 一旦将收到的数据包写入接收 FIFO，控制器即生成 RXFLVL 中断。
5. 为处理 RXFLVL 中断，需要先屏蔽 RXFLVL 中断，并读取接收数据包状态以确认收到的字节数，然后再读取接收 FIFO。按照这个步骤可以解除 RXFLVL 中断屏蔽。
6. 控制器在传输完成状态输入接收 FIFO 后会产生 RXFLVL 中断。
7. 应用程序必须读取接收的数据包状态，当接收的数据包不是 IN 数据包时，则不理睬它。
8. 控制器在接收到的数据包被读取之后会生成 XFERC 中断。
9. 为处理 XFERC 中断，需要先中止通道（详见“中止通道”），并停止写入 OTGFS 主机通道 2 特性寄存器(OTGFS_HCCHAR2)。控制器会在 OTGFS 主机通道 2 特性寄存器(OTGFS_HCCHAR2)被写入后向非周期性请求队列写入通道中止请求。
10. 控制器在中止状态写入接收 FIFO 后会生成 RXFLVL 中断。
11. 读取接收数据包状态，但不予处理。
12. 一旦从接收 FIFO 中读出中止状态，控制器即会生成 CHHLTD 中断。

13. 如果需要处理 CHHLTD 中断，则需要取消为其它传输分配通道。

(2) 处理中断

下列代码描述了批量传输和 IN 传输流程中与通道相关的中断服务程序：

```
Unmask (XACTERR/XFERC/BBLERR/STALL/DATATGLERR)
if (XFERC)
{
    {
        Reset Error Count
        Unmask CHHLTD
        Disable Channel
        Reset Error Count
        Mask ACK
    }
} else if (XACTERR or BBLERR or STALL)
{
    {
        Unmask CHHLTD
        Disable Channel
        if (XACTERR)
        {
            {
                Increment Error Count
                Unmask ACK
            }
        }
    }
} else if (ChHltd)
{
    {
        Mask CHHLTD
        if (Transfer Done or (Error_count == 3))
        {
            {
                De-allocate Channel
            }
        }
    }
} else if (ACK)
{
    {
        Reset Error Count
        Mask ACK
    }
} else if (DATATGLERR)
{
    {
        Reset Error Count
    }
}
```

22.5.3.8 初始化批量和控制 OUT/SETUP 传输

图 22-7 介绍了典型的批量传输或控制传输的 OUT/SETUP 操作流程。详见通道 1 (ch_1)。需要发送 2 个批量传输的 OUT 数据包。控制传输的 SETUP 操作流程也相同，只是只有一个数据包。假设：

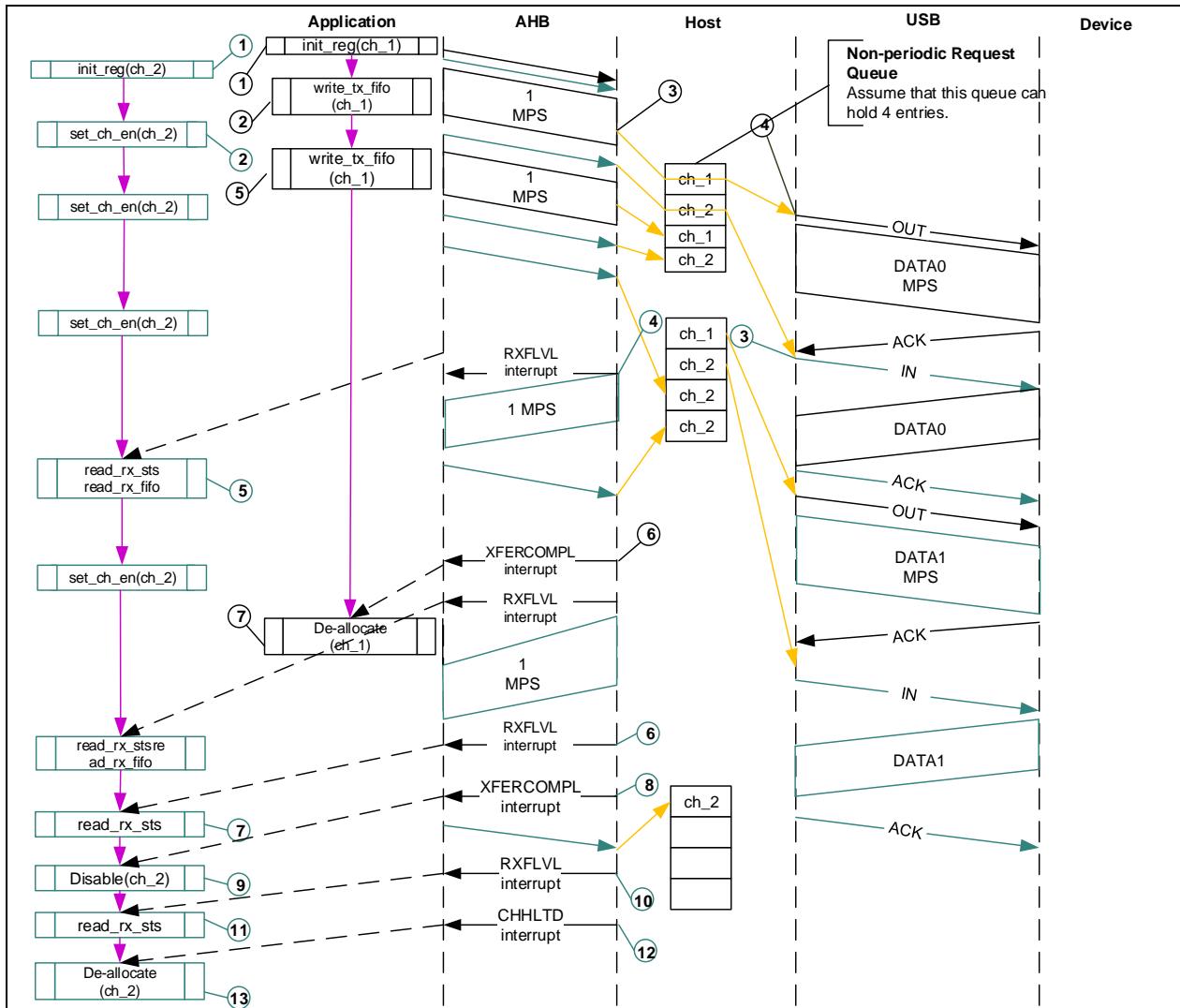
- 应用程序正在尝试发送两个最大包长度的数据包（传输长度为 1024 字节）
- 非周期性发送 FIFO 可以保存 2 个数据包（对于全速传输有 128 字节）
- 非周期性请求队列深度为 4。

(1) 普通批量传输和控制传输的 OUT/SETUP 操作流程

图 22-7 所示的操作流程如下：

1. 初始化通道 1（按照“OTGFS 通道初始化”步骤）
2. 写通道 1 的第一个数据包
3. 随着最后一个 WORD 写入，控制器向非周期性请求队列写入一个条目。
4. 一旦非周期性队列变为非空，控制器就会在当前帧帧内发送一个 OUT 令牌。
5. 向通道 1 写入第二个（最后一个）数据包
6. 在前一个传输成功完成后，控制器就会生成 XFERC 中断。
7. 为响应 XFERC 中断，需要取消为其他传输分配通道。

图 22-7 普通Bulk/Control OUT/SETUP和Bulk/Control IN传输例程示例图



(2) 处理中断

下列代码给出了批量传输、控制传输 OUT/SETUP 流程中与通道相关的中断服务程序：

```
Unmask (NAK/XACTERR/NYET/STALL/XFERC)
if (XFERC)
{
    Reset Error Count
    Mask ACK
    De-allocate Channel
}
else if (STALL)
{
    Transfer Done = 1
    Unmask CHHLTD
    Disable Channel
}
else if (NAK or XACTERR or NYET)
{
    Rewind Buffer Pointers
    Unmask CHHLTD
    Disable Channel
    if (XactErr)
    {
        Increment Error Count
        Unmask ACK
    }
}
```

```
        Reset Error Count
    }
}
else if (CHHLTD)
{
    Mask CHHLTD
    if (Transfer Done or (Error_count == 3))
    {
        De-allocate Channel
    }
    else
    {
        Re-initialize Channel (Do ping protocol for HS)
    }
}
else if (ACK)
{
    Reset Error Count
    Mask ACK
}
```

注意事项：

- 应用程序必须在发送 FIFO 和请求队列里有剩余空间时才能向发送 FIFO 写入数据包。应用程序需要通过非周期性发送 FIFO 空寄存器 (OTGFS_GINTSTS.NPTXFEMP) 中断来检查发送 FIFO 是否有可用空间。
- 应用程序必须在请求队列有可用空间才能写入请求直到收到 XFERC 中断。

22.5.3.9 初始化中断IN传输

图 22-8 显示了典型的中断 IN 传输操作流程。详见通道 2(ch_2)。假设：

- 应用程序正在尝试从奇数帧开始接收最大一个数据包长度的数据包（传输长度为 1024 字节）
- 接收 FIFO 可以保存至少一个最大数据包长度的包以及每个数据包的两个状态 WORDs（对于全速传输有 1031 字节）
- 周期性请求队列深度为 4。

(1) 普通的中断 IN 操作流程

图 22-8 (通道 2) 所描述的操作流程如下：

1. 初始化通道 2 (按照“OTGFS 通道初始化”步骤)。应用程序必须设置奇数帧寄存器 (OTGFS_HCCHAR2.ODDFRM) 位。
2. 设置通道使能寄存器 (OTGFS_HCCHAR2.CHENA) 位，向周期性请求队列写入 IN 请求。
3. 每次置起 OTGFS 主机通道 2 特性寄存器(OTGFS_HCCHAR2)的 CHENA 位时，OTGFS 主机都会向周期性请求队列写入一个 IN 请求。
4. OTGFS 主机尝试在下一个 (奇数) 帧时发送一个 IN 令牌。
5. 一旦收到 IN 数据包，并写入接收 FIFO 后，OTGFS 主机就会生成 RXFLVL 中断。
6. 为了处理 RXFLVL 中断，需要先读取接收的数据包状态以确认接收的字节数，然后再读取接收 FIFO。应用程序必须在读取接收 FIFO 之前先屏蔽 RXFLVL 中断，并在读完整个数据包之后解除中断屏蔽。
7. 控制器在传输完成状态写入接收 FIFO 后会生成 RXFLVL 中断。应用程序必须读取接收包状态并在检测到接收包状态不是 IN 数据包时忽略这个包。
8. 一旦读出接收数据包状态后，控制器即生成 XFERC 中断。
9. 为了处理 XFERC 中断，需要先读取包数目寄存器 (OTGFS_HCTSIZ2.PKTCNT 位)。如果 OTGFS_HCTSIZ2.PKTCNT 不为 0，则需要中止该通道，然后重新初始化该通道进行下次传输。如果 OTGFS_HCTSIZ2.PKTCNT == 0，则需要重新初始化该通道进行下一个传输。此时，应用程序必须复位奇数帧寄存器(OTGFS_HCCHAR2.ODDFRM) 位。

(2) 处理中断

下列代码描述了中断 IN 传输流程中与通道相关的中断服务程序。

```
Unmask (NAK/XACTERR/XFERC/BBLERR/STALL/FRMOVRUN/DATATGLERR)
if (XFERC)
{
    {
        Reset Error Count
        Mask ACK
        if (HCTSIZx.PKTCNT == 0)
        {
            De-allocate Channel
        }
    }
    else
    {
        Transfer Done = 1
        Unmask CHHLTD
        Disable Channel
    }
}
else if (STALL or FRMOVRUN or NAK or DATATGLERR or BBLERR)
{
    Mask ACK
    Unmask CHHLTD
    Disable Channel
    if (STALL or BBLERR)
    {
        Reset Error Count
        Transfer Done = 1
    }
    else if (!FRMOVRUN)
    {
        Reset Error Count
    }
}
else if (XACTERR)
{
    Increment Error Count
    Unmask ACK
    Unmask CHHLTD
    Disable Channel
}
else if (CHHLTD)
{
    Mask CHHLTD
    if (Transfer Done or (Error_count == 3))
    {
        De-allocate Channel
    }
    else Re-initialize Channel (in next b_interval - 1 uF/F)
}
else if (ACK)
{
    Reset Error Count
    Mask ACK
}
```

应用程序必须在请求队列的剩余空间达到 MC 位域指定数目时才能向同一个通道写入请求，然后再切换到其它通道（如果有的话）。

22.5.3.10 初始化中断OUT传输

图 22-8 显示了典型的中断 OUT 操作流程。详见通道 1(ch_1)。假设：

- 应用程序正在尝试从奇数帧开始每个帧发送一个最大包长度的数据包（传输长度为 1024 字节）
- 周期性发送 FIFO 可保存 1 个包（对于全速模式为 1KB）
- 周期性请求队列长度为 4。

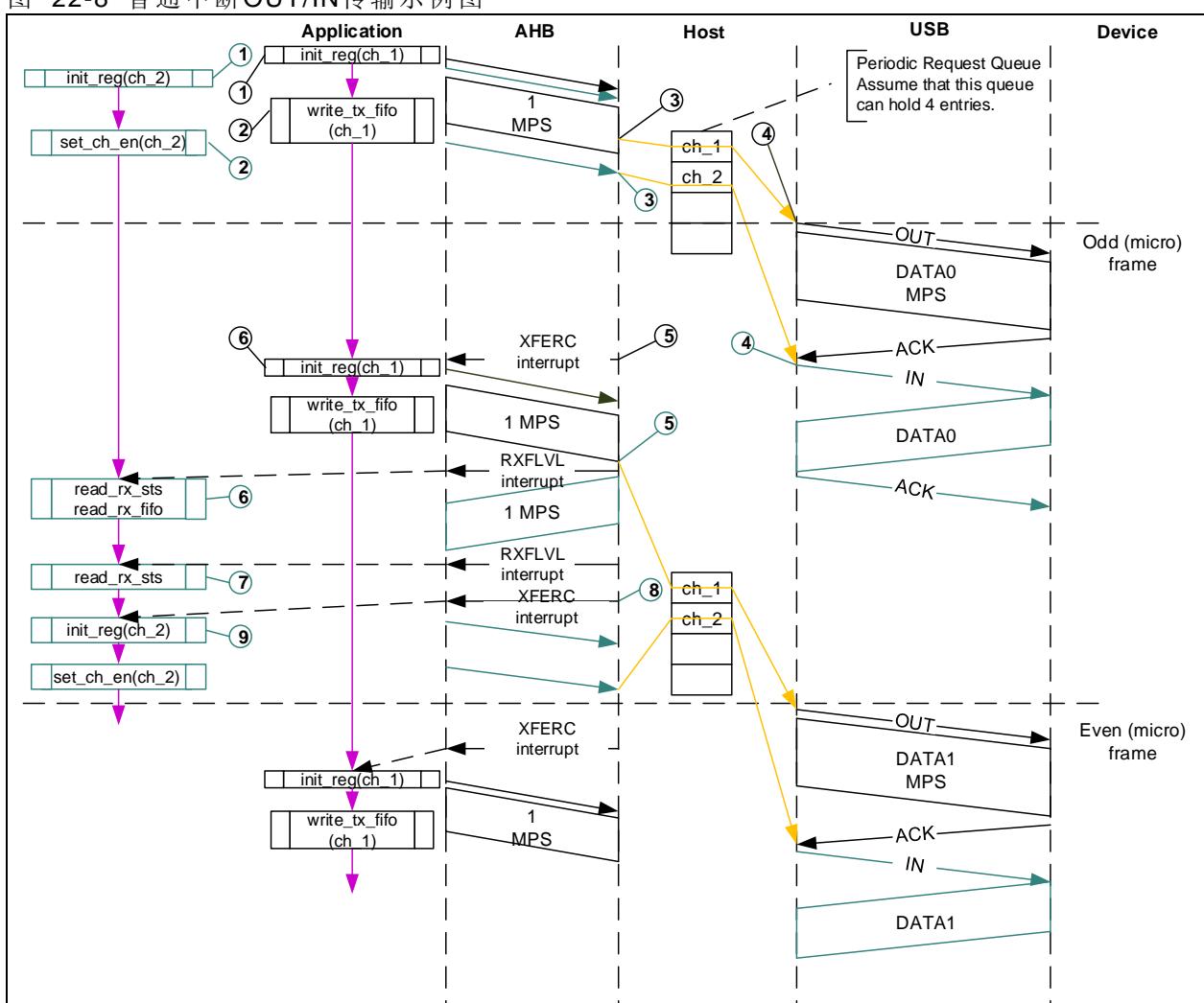
(1) 普通的中断 OUT 操作

图 22-8（通道 1）所描述的操作流程如下：

1. 参考“OTGFS 通道初始化”使能并初始化通道 1。应用程序必须设置奇数帧寄存器(OTGFS_HCCHAR1.ODDFRM)位。
2. 向通道 1 写入第一个数据包。
3. 在写完每个包的最后一个 WORD 时，主机向周期性请求队列写入一个请求。
4. 主机会在下一个帧时(奇数帧)时发送 OUT 令牌。
5. 在最后一个包发送成功后，主机会生成 XFERC 中断。
6. 为响应 XFERC 中断，需要重新初始化该通道进行下次传输。

(2) 处理中断

图 22-8 普通中断 OUT/IN 传输示例图



下列代码示例为中断 OUT 传输流程中与通道相关的中断服务程序

```
Unmask (NAK/XACTERR/STALL/XFERC/FRMOVRUN)
if (XFERC)
{
    Reset Error Count
    Mask ACK
    De-allocate Channel
}
else if (STALL or FRMOVRUN)
{
    Mask ACK
    Unmask CHHLTD
    Disable Channel
    if (STALL)
    {
        Transfer Done = 1
    }
}
else if (NAK or XACTERR)
{
    Rewind Buffer Pointers
    Reset Error Count
    Mask ACK
    Unmask CHHLTD
    Disable Channel
}
else if (CHHLTD)
{
    Mask CHHLTD
    if (Transfer Done or (Error_count == 3))
    {
        De-allocate Channel
    }
    else
    {
        Re-initialize Channel (in next b_interval - 1 uF/F)
    }
}
else if (ACK)
{
    Reset Error Count
    Mask ACK
}
```

在切换到其它通道之前，应用程序需要根据 MC 位域设定的数目，在发送 FIFO 有可用空间时向发送 FIFO 和请求队列写入数据包。应用程序通过非周期性发送 FIFO 空寄存器 (OTGFS_GINTSTS.NPTXFEMP) 中断来确认发送 FIFO 是否有可用空间。

22.5.3.11 初始化同步IN传输

图 22-9 显示了典型的同步 IN 传输流程。详见通道 2(ch_2)。假设：

- 应用程序正在尝试从下一个奇数帧开始每个帧发送一个最大数据包长度的数据包（传输长度为 1024 字节）
- 接收 FIFO 可保存至少一个最长包长度的数据包和两个状态 WORDs（对于全速传输有 1031 字节）
- 周期性请求队列深度为 4。

(1) 普通的同步 IN 传输

图 22-9（通道 2）所描述的操作流程如下：

1. 参考“OTGFS 通道初始化”初始化通道 2。应用程序必须设置奇数帧寄存器(OTGFS_HCCHAR2.ODDFRM) 位。
2. 设置通道使能寄存器(OTGFS_HCCHAR2.CHENA) 位向周期性请求队列写入 IN 请求。
3. 每次设置 OTGFS 主机通道 2 特性寄存器(OTGFS_HCCHAR2)的 CHENA 位，主机就会向周期性请求队列写入一个 IN 请求
4. 主机会在下一个奇数帧时发送 IN 令牌
5. 当接收到 IN 数据包并写入接收 FIFO 后，主机会生成 RXFLVL 中断。
6. 为响应 RXFLVL 中断，需要读取接收的数据包状态以确认接收的字节数，然后读取接收 FIFO。应用程序在读取接收 FIFO 之前必须先屏蔽 RXFLVL 中断，并在读取了整个数据包之后解除中断屏蔽。
7. 控制器在传输完成状态输入到接收 FIFO 之后会生成 RXFLVL 中断。此时，应用程序必须读取接收数据包状态，并在检测到接收数据包状态不是 IN 数据包时丢弃这个包。(GRXSTSR.PKTSTS!= 0x0010)。
8. 控制器在读取了接收数据包状态后会生成 XFERC 中断。
9. 为响应 XFERC 中断，需要读取包数目寄存器(OTGFS_HCTSIZ2.PKTCNT) 位。如果 OTGFS_HCTSIZ2.PKTCNT 不为 0，然后再重新初始化该通道前中止该通道，然后进行下次传输（如果有的话）。如果 OTGFS_HCTSIZ2.PKTCNT == 0，需重新初始化该通道以进行下一次传输。此时，应用程序必须读取奇数帧寄存器(OTGFS_HCCHAR2.ODDFRM) 位。

(2) 处理中断

下列代码示例同步 IN 传输流程中与通道相关的中断服务程序。

```
Unmask (XACTERR/XFERC/FRMOVRUN/BBLERR)
if (XFERC or FRMOVRUN)
{
    if (XFERC and (HCTSIZx.PKTCNT == 0))
    {
        Reset Error Count
        De-allocate Channel
    }
    else
    {
        Unmask CHHLTD
        Disable Channel
    }
}
else if (XACTERR or BBLERR)
{
    Increment Error Count
    Unmask CHHLTD
    Disable Channel
}
else if (CHHLTD)
{
    Mask CHHLTD
    if (Transfer Done or (Error_count == 3))
    {
        De-allocate Channel
    }
    else
    {
        Re-initialize Channel
    }
}
```

22.5.3.12 初始化同步OUT传输

图 22-9 显示了典型的中步 OUT 传输操作流程。详见通道 1(ch_1)。假设：

- 应用程序正在尝试从下一个奇数帧开始每个帧发送一个最大数据包长度的数据包（传输长度为 1024 字节）
- 周期性发送 FIFO 可保存 1 个数据包（对于全速模式为 1KB）
- 周期性请求队列深度为 4。

(1) 普通的同步 OUT 传输操作流程

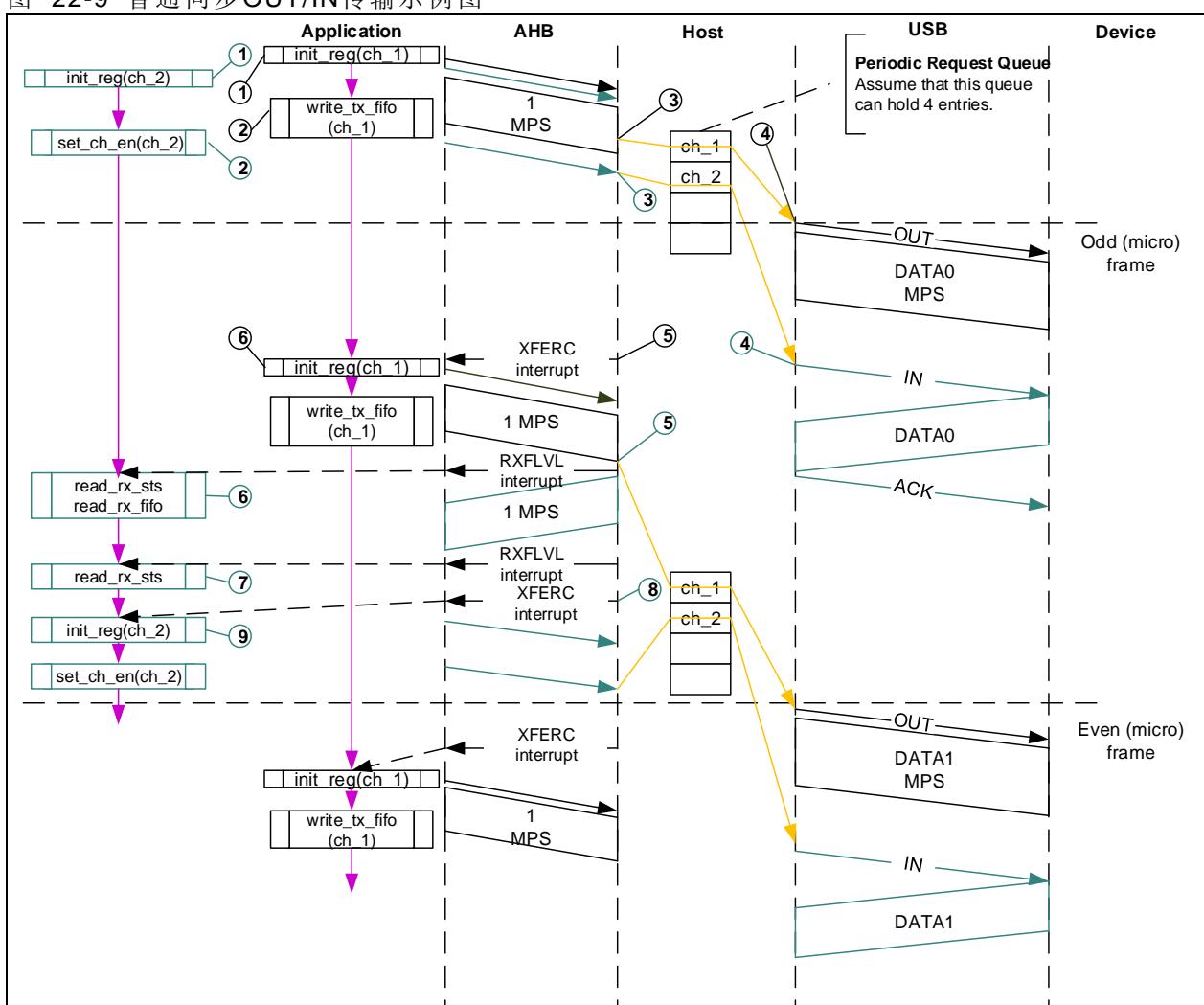
图 22-9（通道 2）所描述的操作流程如下：

1. 参考“OTGFS 通道初始化”初始化通道 1。应用程序必须设置奇数帧寄存器(OTGFS_HCCHAR1.ODDFRM)位。
2. 向通道 1 写入第一个数据包。
3. 在写完每个包的最后一个 WORD 时，主机向周期性请求队列写入一个请求。
4. OTGFS 尝试在下一个帧（奇数帧）发送 OUT 令牌。
5. 一旦发送完成最后一个数据包，OTGFS 主机即会生成 XFERC 中断。
6. 为响应 XFERC 中断，需要重新初始化该通道进行下一个传输。

(2) 处理中断

图 22-9 显示了同步 OUT 传输操作过程中与通道相关的中断服务程序。

图 22-9 普通同步OUT/IN传输示例图



下列示例代码为同步 OUT 传输过程中与通道相关的中断服务程序。

```
Unmask (FRMOVRUN/XFERC)
if (XFERC)
{
    {
        De-allocate Channel
    }
}
else if (FRMOVRUN)
{
    {
        Unmask CHHLTD
        Disable Channel
    }
}
else if (CHHLTD)
{
    {
        Mask CHHLTD
        De-allocate Channel
    }
}
```

22.5.4 OTGFS设备模式

22.5.4.1 设备初始化

在设备开启时，上电时或者从主机模式切换到设备模式时，应用程序需要遵循下列步骤初始化控制器。

1. 配置 OTGFS 设备配置寄存器(OTGFS_DCFG)的相应位
 - 设备速度
 - 非零长度状态 OUT 握手
 - 周期性帧间隔
2. 清除 OTGFS 设备控制寄存器(OTGFS_DCTL.SFTDISCON)位。控制器在清除此位后会启动连接。
3. 设置 OTGFS 中断屏蔽寄存器(OTGFS_GINTMSK)以解除下列中断屏蔽：
 - USB 复位
 - 枚举完成
 - 早期挂起
 - USB 挂起
 - SOF
4. 等待 OTGFS 中断寄存器(OTGFS_GINTSTS.USBRESET)中断，该中断表示检测到 USB 总线上的复位信号已持续 10ms。应用程序一旦接收此中断，就必须按照“USB 复位时初始化”的步骤操作。
5. 等待 OTGFS 中断寄存器(OTGFS_GINTSTS.ENUMDONE)中断。该中断指示 USB 复位结束。应用程序在接收到此中断后，需读取 OTGFS 设备状态寄存器(OTGFS_DSTS)来确认枚举速度，并按照“枚举完成时初始化”的步骤来操作。此时，设备准备接受 SOF 包，并在控制端点 0 执行控制传输。

22.5.4.2 USB复位时初始化

本节介绍了当检测到 USB 复位信号时应用程序必须执行的操作。

1. 置起所有 OUT 端点的 NAK 位
 - OTGFS_DOEPCTLx.SNAK = 0x1(针对所有 OUT 端点)
2. 解除下列位的中断屏蔽
 - OTGFS_DAINTMSK.INEP0 = 0x1(控制 IN 端点 0)
 - OTGFS_DAINTMSK.OUTEP0 = 0x1(控制 OUT 端点 0)
 - OTGFS_DOEPMSK.SETUP = 0x1
 - OTGFS_DOEPMSK.XFERC = 0x1
 - OTGFS_DIEPMSK.XFERC = 0x1
 - OTGFS_DIEPMSK.TIMEOUT = 0x1
3. 为了收发数据，设备必须遵守“设备初始化”流程对寄存器进行初始化
4. 为每个 FIFO 分配 SRAM
 - 设置 OTGFS 接收 FIFO 长度寄存器(OTGFS_GRXFSIZ)以确保能接收控制 OUT 数据和 SETUP 数据。所分配的 SRAM 至少是控制端点 0 的 1 个最大包长度+2 个 WORDs (用于控制 OUT 数据包的状态信息)+10 个 WORDs (用于 setup 包)。
 - 配置 OTGFS 非周期性 TX FIFO 长度寄存器(OTGFS_GNPTXFSIZ) (具体情况取决于所选择的 FIFO 号)，以确保能够发送控制 IN 数据。所分配的 SRAM 至少是控制端点 0 的 1 个最大包长度。

5. 复位“设备配置寄存器”的“设备地址”位。
 6. 设置与端点控制寄存器中的下列位域，确保控制 OUT 端点 0 能够接收 SETUP 数据包。
 - OTGFS_DOEPTSIZ0.SUPCNT = 0x3(可接收高达 3 个连续的 SETUP 数据包)
- 此时，用于接收 SETUP 数据包的所有初始化操作就完成了。

22.5.4.3 枚举完成时初始化

本节介绍了当检测到枚举完成中断信号时应用程序必须执行的操作。

- 检测到枚举完成中断信号时，读取 OTGFS 设备状态寄存器(OTGFS_DSTS)来获取枚举速度。
 - 配置 OTGFS 设备控制 IN 端点 0 控制寄存器 (OTGFS_DIEPCTL0.MPS)位来设置最大包长度。这个操作是用于配置控制端点 0. 控制端点的最大包长度是由枚举速度决定的。
 - 解除 SOF 中断屏蔽。
- 此时，设备准备接收 SOF 包并已经设置好，准备在控制端点 0 执行控制传输。

22.5.4.4 SetAddress 指令时初始化

本节介绍了当 SETUP 包收到了 SetAddress 指令时应用程序必须执行的操作。

- 使用 SetAddress 指令收到的设备地址来配置 OTGFS 设备配置寄存器 (OTGFS_DCFG)。
- 设置控制器，发送 IN 包。

22.5.4.5 SetConfiguration/SetInterface 时初始化

本节介绍了在收到 SetConfiguration / SetInterface 命令时应用程序必须执行的操作。

- 在收到 SetConfiguration 命令时，应用程序需根据新配置定义的有效端点的特性来设置端点寄存器。
- 在收到 SetInterface 命令时，应用程序需针对受到该命令影响的端点配置端点寄存器。
- 有些端点在之前的配置中是有效的，但是在新的配置中可能失效。必须停用这些无效端点。
- 关于如何激活或停用某个端点的，详见“激活端点”以及“USB 端点失效操作”
- 解除每个有效端点的中断屏蔽，并屏蔽 DAINMSK 寄存器中的所有无效端点的中断
- 为每个 FIFO 配置 SRAM。详见“OTGFS FIFO 配置”。
- 在配置完所有端点后，应用程序需要设置控制器来发送状态 IN 包。

此时，设备控制器已准备好接收和发送任何类型的数据包。

22.5.4.6 激活端点

本节介绍了如何激活一个设备端点或者将现有的设备端点配置为新端点类型。

1. 配置 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx)(对于 IN 或双向端点)或 OTGFS 设备 OUT 端点 x 控制寄存器(OTGFS_DOEPCTLx) (对于 OUT 或双向端点) 的以下位：
 - 最大包长度
 - USB 有效端点位 = 1
 - 端点起始数据翻转 (指中断和批量类型的端点)
 - 端点类型
 - 发送 FIFO 号
2. 一旦激活端点，控制器就开始解析发送到该端点的令牌，并针对该端点收到的每一个有效令牌发出一个握手有效信号

22.5.4.7 USB 端点失效操作

本节介绍的是如何使一个现存的端点失效。必须先终止挂起的传输，才能进行端点失效操作。

- 清除 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx) (对于 IN 或双向端点) 或者 OTGFS 设备 OUT 端点 x 控制寄存器(OTGFS_DOEPCTLx) (对于 OUT 或双向端点) 的“USB 有效端点”位。
- 一旦端点失效，控制器就会忽略发送到该端点的令牌，这将导致 USB 超时。

22.5.4.8 控制写传输(SEUP/Data OUT/Status IN)

本节介绍了控制写传输的操作流程。

应用程序配置流程为：

1. OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.SETUP)包中断置起表示已经向应用程序发送了一个有效 SETUP 包，且数据阶段已启动，详见“OUT 数据传输”。在 SETUP 阶段结束时，应用程序需重新向 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DOEPTSIZx.SUPCNT)位写 3 来接收下一个 SETUP 包。
2. 如果在 SETUP 中断生成之前收到的最后一个 SETUP 包指示数据 OUT 阶段，需按照“非同步 OUT 数据传输”配置控制器来执行控制 OUT 传输。
3. 对于控制端点 0 的单次 OUT 数据传输，应用程序最多可接收 64 字节数据。如果应用程序期望在数据 OUT 阶段接收不止 64 字节的数据，那么必须重新使能该端点另外再接收 64 字节数据，而且需要持续此类操作直到接收完数据阶段的所有数据。
4. 在最后一个 OUT 传输时置起 OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.XFERC)中断表示该控制传输的数据 OUT 阶段结束。
5. 一旦完成数据 OUT 阶段，应用程序必须执行如下操作：
 - 如果需要传输一个新的 SETUP 包，应用程序必须重新使能控制 OUT 端点（参考“OUT 数据传输”）
OTGFS_DOEPCTLx.EPENA = 0x1
 - 为了执行接收到的 SETUP 命令，应用程序必须配置控制器中的相应寄存器。这个是可选操作，视所收到的 SETUP 命令类型而定。
6. 在状态 IN 阶段，应用程序必须按照“非周期性（批量和控制）IN 数据传输”来配置寄存器以执行数据 IN 传输。
7. OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.XFERC)中断表示控制传输的状态阶段已启动。当接收 FIFO 包状态寄存器的“数据传输完成模式”和“状态阶段开始”位置起时，控制器即会生成该中断。通过设置 OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.XFERC)来清除“传输完成”中断。重复上述步骤直至检测到该端点上产生 OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.XFERC)中断才表示该控制写传输已完成。

22.5.4.9 控制读传输(SETUP/Data IN/Status OUT)

本节介绍的是控制读传输。应用程序操作流程：

- OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.SETUP)包中断表示已向应用程序发送了一个有效的 SETUP 包，而且数据阶段已启动。详见“OUT 数据传输”。应用程序在 SETUP 阶段结束时，必须重新向 OTGFS 设备 OUT 端点 0 传输长度寄存器(OTGFS_DOEPTSIZ0.SUPCNT)位写 3 来接收下一个 SETUP 包。
- 如果在 SETUP 中断生成之前收到的最后一个 SETUP 包指示数据 IN 阶段，需按照“非周期性 IN 数据传输”配置控制器来执行控制 IN 传输。
- 对于控制端点 0 的单次 IN 数据传输，应用程序最多可接收 64 字节数据。如果应用程序期望在数据 IN 阶段发送不止 64 字节的数据，那么必须重新使能该端点另外再发送 64 字节数据，而且需要持续此类操作直到发送完数据阶段的所有数据。
- 重复上述步骤直至检测到该端点上每个 IN 传输都生成了 OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.XFERC)中断。
- 在最后一个 IN 传输时置起 OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.XFERC)中断表示该控制传输的数据 OUT 阶段结束
- 如需在状态 OUT 阶段执行数据 OUT 传输，应用程序需要按照“OUT 数据传输”来配置控制器。应用程序必须先正确设置 OTGFS 设备配置寄存器(OTGFS_DCFG.NZSTSOUTSHHK)握手位，再发送状态阶段的数据 OUT 传输。
- OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.XFERC)中断表示该控制传输的状态 OUT 阶段结束，标志着控制读传输成功完成。

22.5.4.10 两个阶段的控制传输(SETUP/Status IN)

本节介绍了两个阶段的控制传输操作。应用程序操作流程：

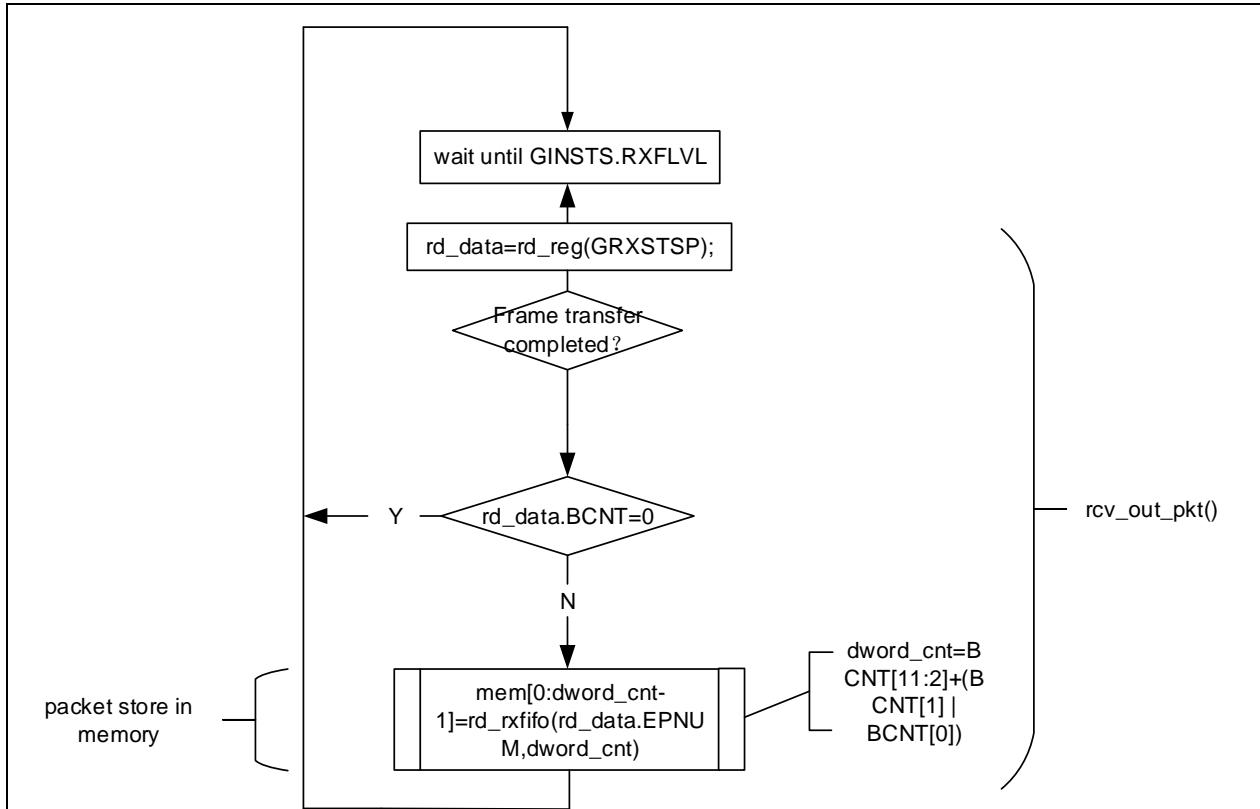
1. OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.SETUP)包中断表示已向应用程序发送了一个有效的 SETUP 包，而且数据阶段已启动。详见“OUT 数据传输”。应用程序在 SETUP 阶段结束时，必须重新向 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DOEPTSIz.x.SUPCNT)位写 3 来接收下一个 SETUP 包。
2. 在 SETUP 中断生成之前对所接收的最后一个 SETUP 包进行解析。如果 SETUP 包指示的两级控制命令，那么应用程序必须执行以下操作：
 - 设置 OTGFS_DOEPCTLx.EPENA = 0x1
 - 根据所收到的 SETUP 命令类型，应用程序需要配置控制器中的寄存器来执行所收到的 SETUP 命令。
3. 对于状态 IN 阶段，应用程序需按照“非周期性（批量和控制）IN 数据传输”配置寄存器来执行数据 IN 传输。
4. OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.XFERC)中断表示该控制传输的状态 IN 阶段已完成。

22.5.4.11 读取 FIFO 包

本节介绍了如何读取接收 FIFO 数据包（OUT 数据和 SETUP 包）

1. 一旦检测到 OTGFS 中断寄存器(OTGFS_GINTSTS.RXFLVL)中断，应用程序必须读取 OTG 状态读和 POP 寄存器(OTGFS_GRXSTSP)
2. 应用程序可以通过设置 OTGFS_GINTMSK.RXFLVL = 0x0 来屏蔽 OTGFS 中断寄存器(OTGFS_GINTSTS.RXFLVL)中断，直到从接收 FIFO 读取到数据包
3. 如果收到的数据包字节数不为 0，那么接收数据 FIFO 会弹出数据字节数并将其储存在存储器。如果所接收的数据包字节数为 0，那么接收数据 FIFO 则不会有数据读出。
4. 接收 FIFO 包状态读数指示发生下列情况：
 5. 全局 OUT NAK 模式：PKTSTS = Global OUT NAK, BCNT = 0x000, EPNUM = Dont Care (0x0), DPID = Dont Care (0x00)，指示全局 OUT NAK 位已生效。
 - SETUP 包模式：PKTSTS = SETUP, BCnt = 0x008, EPNUM = Control EP Num, DPID = D0， 表示可以从接收 FIFO 读取某个指定端点的 SETUP 包。
 - Setup 阶段完成模式：PKTSTS = Setup Stage Done, BCNT = 0x0, EPNUM = Control EP Num, DPID = Don't Care (0x00)， 表示某个指定端点的 Setup 阶段已结束，并开始进入数据阶段。当接收 FIFO 弹出此条目后，控制器会在指定的控制 OUT 端点上触发 Setup 中断。
 - 数据 OUT 包模式：PKTSTS = DataOUT, BCnt = 接收的数据 OUT 包的长度($0 \leq BCNT \leq 1024$), EPNUM = 接收数据包的端点号, DPID = 实际的数据 PID。
 - 数据传输完成模式：PKTSTS = 数据 OUT 传输完成, BCNT = 0x0, EPNUM = 完成数据传输的 OUT 端点号, DPID = Don't Care (0x00)。这些数据表示某个指定的 OUT 端点的 OUT 数据传输已完成。当接收 FIFO 弹出此条目后，控制器会在指定的 OUT 端点上触发传输完成中断。PKTSTS 代码位于本手册的寄存器章节的“接收状态调试读取/状态读和弹出寄存器”
 6. 接收 FIFO 弹出有效数据后，必须解除 OTGFS 中断寄存器(OTGFS_GINTSTS.RXFLVL)中断屏蔽。
 7. 每当应用程序检测到由于 OTGFS 中断寄存器(OTGFS_GINTSTS.RXFLVL)而产生中断线时需要重复第 1-5 步。读取空接收 FIFO 会导致控制器出现意外后果。流程如下图：

图 22-10 读取接收 FIFO



22.5.4.12 OUT数据传输

本节介绍了数据 OUT 传输和 SETUP 传输过程中的内部数据流及应用程序的操作流程。

(1) 执行 Setup 传输

本节介绍了如何处理 SETUP 数据包以及应用程序处理 SETUP 传输的操作流程。上电复位后，应用程序必须遵循“OTGFS 初始化”流程来初始化控制器。应用程序在与主机通信之前，必须按照“设备初始化”流程来初始化端点，并参考“读取 FIFO 包”。

【应用程序要求】

1. 控制 OUT 端点的 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DOEPTSIZx.SUPCNT)位必须设置为非零值才能接收 SETUP 包。当应用程序将 SUPCNT 位设置为非零值时，控制器会接收到 SETUP 包并将其写入接收 FIFO，不受 OTGFS 设备 OUT 端点 x 控制寄存器(OTGFS_DOEPCTLx.NAK)状态位和 OTGFS 设备 OUT 端点 x 控制寄存器(OTGFS_DOEPCTLx.EPENA)位的影响。SUPCNT 位在每次控制端点接收到一个 SETUP 包时会自动递减。如果在接收 SETUP 包之前，SUPCNT 位的设置值不合适，那么虽然控制器仍可以接收 SETUP 包并自动递减 SUPCNT 位，但是应用程序可能就无法确定在控制传输 SETUP 阶段所接收的 SETUP 包有多少个。

- OTGFS_DOEPTSIZx.SUPCNT = 0x3

2. 应用程序必须为接收数据 FIFO 分配一些额外的空间，以确保能够在一个控制端点接收多达 3 个 SETUP 包。

- 预留空间为 13WORDs，其中 4 个 WORDs 空间用于 1 个 SETUP 包，1 个 WORD 空间用于 Setup 阶段，8 个 WORDs 空间用于存入控制端点的两个额外的 SETUP 包。
- 每个 SETUP 包需要 4 个 WORDs 空间用于存放 8 个字节的 SETUP 数据，4 个字节的传输完成状态以及 4 个字节的 SETUP 状态（SETUP 包模式）。控制器需要为接收数据预留此空间。

- FIFO 仅用于写 SETUP 数据，而不会用于数据包

3. 应用程序需要从接收 FIFO 读取 2 个 WORDs 的 SETUP 包。

4. 应用程序必须从接收 FIFO 读取传输完成状态 WORD 并丢弃它。并忽略由于读取而产生的传输完成中断。

【内部数据流】

1. 当收到 SETUP 包时，控制器将接收的数据写入接收 FIFO，不需要确认接收 FIFO 是否有可用空间，也

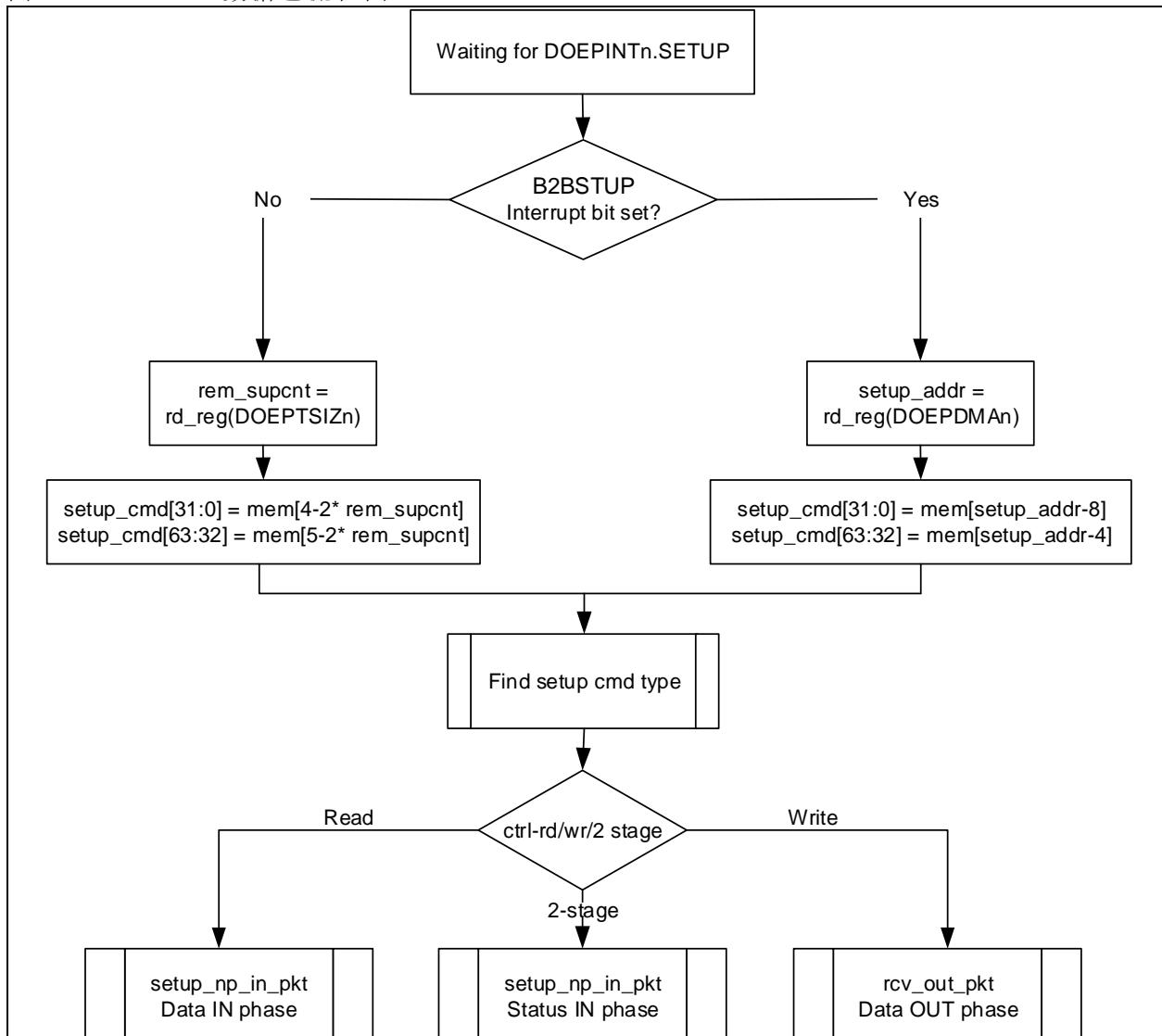
不需要检测控制端点的 NAK 和 Stall 位。

- 控制器会在收到 SETUP 包的控制 IN/OUT 端点上置起 IN NAK 和 OUT NAK 位。
- 2. USB 线上收到每个 SETUP 包后，都会写入 3 个 WORDs 数据到接收 FIFO，SUPCNT 位自动递减 1。
 - 首个 WORD 包含控制器内部使用的控制信息。
 - 第二个 WORD 包含 SETUP 命令的前 4 个字节。
 - 第三个 WORD 包含 SETUP 命令的最后 4 个字节。
- 3. 当从 SETUP 阶段切换到数据 IN/OUT 阶段时，控制器会写入 SETUP 状态完成 WORD 信息到接收 FIFO，指示 SETUP 阶段已结束。
- 4. 应用程序通过 AHB 总线读取 SETUP 包。
- 5. 当应用程序弹出了来自接收 FIFO 的状态阶段完成 WORD 信息时，控制器会生成 OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.SETUP)中断来打断应用程序，指示应用程序可以开始处理接收的 SETUP 包。
- 6. 控制器清除控制 OUT 端点的端点使能位。

【应用程序操作流程】

1. 配置 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DOEPTSI Z_x)。
 - OTGFS_DOEPTSI Z_x .SUPCNT = 0x3
2. 等待 OTGFS 中断寄存器(OTGFS_GINTSTS.RXFLVL)中断并读取刷新接收 FIFO 的数据包（参考“读取 FIFO 包”）。可以多次重复此操作。
3. 宣告 OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.SETUP)中断表示 SETUP 数据传输已成功完成。在收到此中断后，应用程序需要读取 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DOEPTSI Z_x)来确认收到了多少个 SETUP 包，并处理最后接收到的一个 SETUP 包。

图 22-11 SETUP 数据包流程图



(2) 处理 3 个以上的连续的 SETUP 包

根据 USB2.0 规范，通常在 SETUP 包出现错误时，主机不会向同一个端点连续发送 3 个以上的 SETUP 包。然而，USB2.0 规范并没有限制主机可以向同一个端点发送连续 SETUP 包的数目。如果发生此类情况，OTGFS 控制器会生成一个中断(OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.B2BSTUP))。

22.5.4.13 IN数据传输

本节介绍了 IN 数据传输期间的内部数据流和应用程序的操作流程。

1. 应用程序可以选择轮询模式或中断模式。

- 如果选择轮询模式，应用程序将通过读取 OTGFS 设备 IN 端点传输 FIFO 状态寄存器(OTGFS_DTXFSTSx)来监测端点发送数据 FIFO 的状态以确认数据 FIFO 内是否有足够的空间可用。
- 如果选择中断模式，应用程序需要等待 OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.TXFEMP)中断，然后读取 OTGFS 设备 IN 端点传输 FIFO 状态寄存器(OTGFS_DTXFSTSx)来确认数据 FIFO 内是否有足够的空间可用。
- 如果要写一个单独的非零长度的数据包，数据 FIFO 必须要有足够的空间写入整个数据包。
- 如果要写一个零长度的数据包，应用程序不需要查看 FIFO 空间。
- 2. 无论使用上述哪种方式，当应用程序确定了有足够的空间可以写入一个发送数据包时，可以先写入端点控制寄存器，再将数据写入数据 FIFO。通常，除了设置端点使能位之外，应用程序必须在 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx)进行读改写设置以防止该寄存器的内容被更改，如果空间足够大的话，应用程序可以将同一个端点的多个数据包写入发送 FIFO。对于周期性 IN 端点，应用程序只能写入一个帧的数据包。只有在前一个传输发送完成的情况下，才能写入下一个周期性传输。

22.5.4.14 非周期性（批量和控制）IN数据传输

如果要在上电复位后初始化控制器，那么应用程序必须遵循“OTGFS 初始化”的流程。在与主机通讯前，必须先按照“设备初始化”流程对端点进行初始化操作。

【应用程序要求】

1. 对于 IN 传输而言，端点传输长度寄存器中的传输长度位表示的有效数据包含了多个最大包长度的数据包和一个短包。短包在传输结束时被传输。

- 如果需要传输几个最大包长度的数据包和一个短包：
传输长度[epnum] = $n * \text{mps}[epnum] + sp$, (其中 n 是 ≥ 0 的整数，且 $0 \leq sp < \text{mps}[epnum]$)
如果($sp > 0$)，那么包数目[epnum] = $n + 1$ 。否则，包数目[epnum] = n
- 如果需要传输一个单独的零长度的数据包：
传输长度[epnum] = 0x0
包数目[epnum] = 0x1
- 如果要传输几个最大包长度的数据包和一个零长度的数据包（在结束时传输），那么应用程序需要分成两部分进行传输。先发送最大包长度的数据包，再单独发送零长度的数据包。

首次传输：传输长度[epnum] = $n * \text{mps}[epnum]$; 包数目 = n ;

二次传输：传输长度[epnum] = 0x0; 包数目 = 0x1;

2. 如果使能了端点进行数据传输，则控制器会更新传输长度寄存器。在 IN 传输结束时，以端点禁用中断为结束标志，此时应用程序需要读取传输长度寄存器来确认 USB 线上发送了多少 FIFO 数据。

3. 发送 FIFO 获取的数据 = 应用程序设置的首个传输长度 - 控制器更新的最终传输长度

- USB 已发送的数据 = (应用程序设置的首个包数目 - 控制器更新的最终包数目) * $\text{mps}[epnum]$
- USB 待发送的数据 = 应用程序设置的首个传输长度 - USB 已发送的数据

【内部数据流】

1. 应用程序需要设置端点控制寄存器中的传输长度和包数目位，并使能端点来传输数据。

2. 应用程序还需要将所需数据写入该端点的传输 FIFO。

3. 每当应用程序向发送 FIFO 写入一个数据包时，相应端点的传输长度会随着包长度而自动递减。需要持续写入数据直到该端点的传输长度为 0。将数据写入 FIFO 后，“FIFO 中的包数目”会递增（每个 IN 端点发送 FIFO 数据包是 3bit 数目，由内部控制器保存。对于一个 IN 端点 FIFO，任何时候，控制器能保存的最大包数目为 8）。对于非零长度的数据包，每一个 FIFO 会设置一个单独的标志，FIFO 中不会有数据。

4. 当数据写入发送 FIFO 后，控制器在收到 IN 令牌时会读取该数据。对于每个以 ACK 握手信号结束的非同步 IN 数据包传输，该端点的包数目会自动减 1，直到包数目为 0，包数目不会因为超时而递减。
5. 对于零长度数据包（内部会置起一个零长度标志），控制器会根据 IN 令牌发送零长度数据包，并且包数目位会自动递减。
6. 如果收到了 IN 令牌，但 FIFO 没有数据，且该端点的包数目为 0，那么控制器会生成“当 FIFO 为空时收到了 IN 令牌”的中断，同时不设置该端点的 NAK 位。控制器会以 NAK 握手信号来回应 USB 线上的非同步端点。
7. 控制器内部会绕回 FIFO 指针，除了控制 IN 端点外，不会产生超时中断，
8. 当传输长度为 0 并且包数目也为 0 时，会产生传输完成中断，并清除端点使能位。

【应用程序操作流程】

1. 根据传输长度和相应的包数目来设置 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DIEPTSIZx)。
2. 根据端点特性来设置 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx)，并设置 CNAK 和端点使能位。
3. 如果发送非零长度的数据包，则应用程序必须轮询 OTGFS 设备 IN 端点传输 FIFO 状态寄存器(OTGFS_DTXFSTSx)（其中 n 是指与该端点相关的 FIFO 编号）来确认数据 FIFO 是否有足够的可用空间。应用程序也可以使用 OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.TXFEMP)来确定是否写数据。

22.5.4.15 非同步 OUT 数据传输

如果要在上电复位后初始化控制器，那么应用程序必须遵循“OTGFS 初始化”的流程。在与主机通讯前，必须先按照“端点初始化”流程对端点进行初始化操作，并参考“读取 FIFO 包”。本章节介绍了常规非同步 OUT 传输操作（控制传输，批量传输或中断传输）

【应用程序要求】

1. 对于 OUT 传输，端点传输长度寄存器的传输长度位必须是该端点最大包长度的倍数，并调整到 WORD 边界。

```
if (mps[epnum] mod 4) == 0
    transfer size[epnum] = n * (mps[epnum]) //WORD Aligned
else
    transfer size[epnum] = n * (mps[epnum] + 4 - (mps[epnum] mod 4)) //Non WORD
    Aligned
    packet count[epnum] = n
n > 0
```

2. 发生 OUT 端点中断时，应用程序需要读取端点的传输长度寄存器来计算内存中的数据量。所收到的有效数据长度必须小于设定的传输长度。

- 内存中的有效负载=应用程序设置的首个传输长度-控制器更新的最终传输长度
- 收到此有效负载的 USB 数据包数目=应用程序设置的首个包数目—控制器更新的最终包数目

【内部数据流】

1. 应用程序需要设置端点控制寄存器的传输长度和包数目位，清除 NAK 位，并使能接收数据的端点
2. 一旦清除 NAK 位，只要接收 FIFO 里有可用空间，控制器就开始接收数据并写入接收 FIFO。对于 USB 线收到的每个数据包，需要将数据包和其状态写入接收 FIFO。每次向接收 FIFO 中写入数据包（最大包长度或短包），包数目位会自减 1。
 - 接收 FIFO 将自动清空所收到的含有 Bad Data CRC 的 OUT 数据包。
 - 在 USB 发出数据包 ACK 信号后，控制器不会理会主机（因为未检测到 ACK）重新发送的非同步 OUT 数据包。应用程序没有检测到具有相同数据 PID 的同一个端点上的多个连续 OUT 数据包，此时，包数目不会自动递减。
 - 如果接收 FIFO 里没有空间可用，同步或非同步数据包会被忽略，也不会写入接收 FIFO。另外，非同步 OUT 令牌还会收到一个 NAK 握手信号回应。
 - 以上 3 种情况下，包数目不会递减，因为没有数据写入接收 FIFO。
3. 当包数目变为 0 或者当端点收到短包时，该端点会置起 NAK 位。一旦设置 NAK 位，同步或非同步数据包会被忽略，也不会写入接收 FIFO，而且非同步 OUT 令牌还会收到 NAK 握手信号回应。
4. 将数据写入接收 FIFO 后，应用程序会读取接收 FIFO 中的数据并写入外部存储器，每个端点 1 次 1 个包。
5. 在完成了数据包写入外部存储器之后，该端点的传输长度会随着写入包的长度减少而减少。

6. 在发生下列情况时，OUT 端点的 OUT 数据传输完成模式会写入接收 FIFO。

- 传输长度和包数目均为 0。

● 写入接收 FIFO 的最后一个 OUT 数据包是一个短包 ($0 \leq$ 数据包长度 < 最大包长度)

7. 当应用程序弹出该条目（即 OUT 数据传输完成），即会产生“传输完成中断”，且清除该端点使能位。

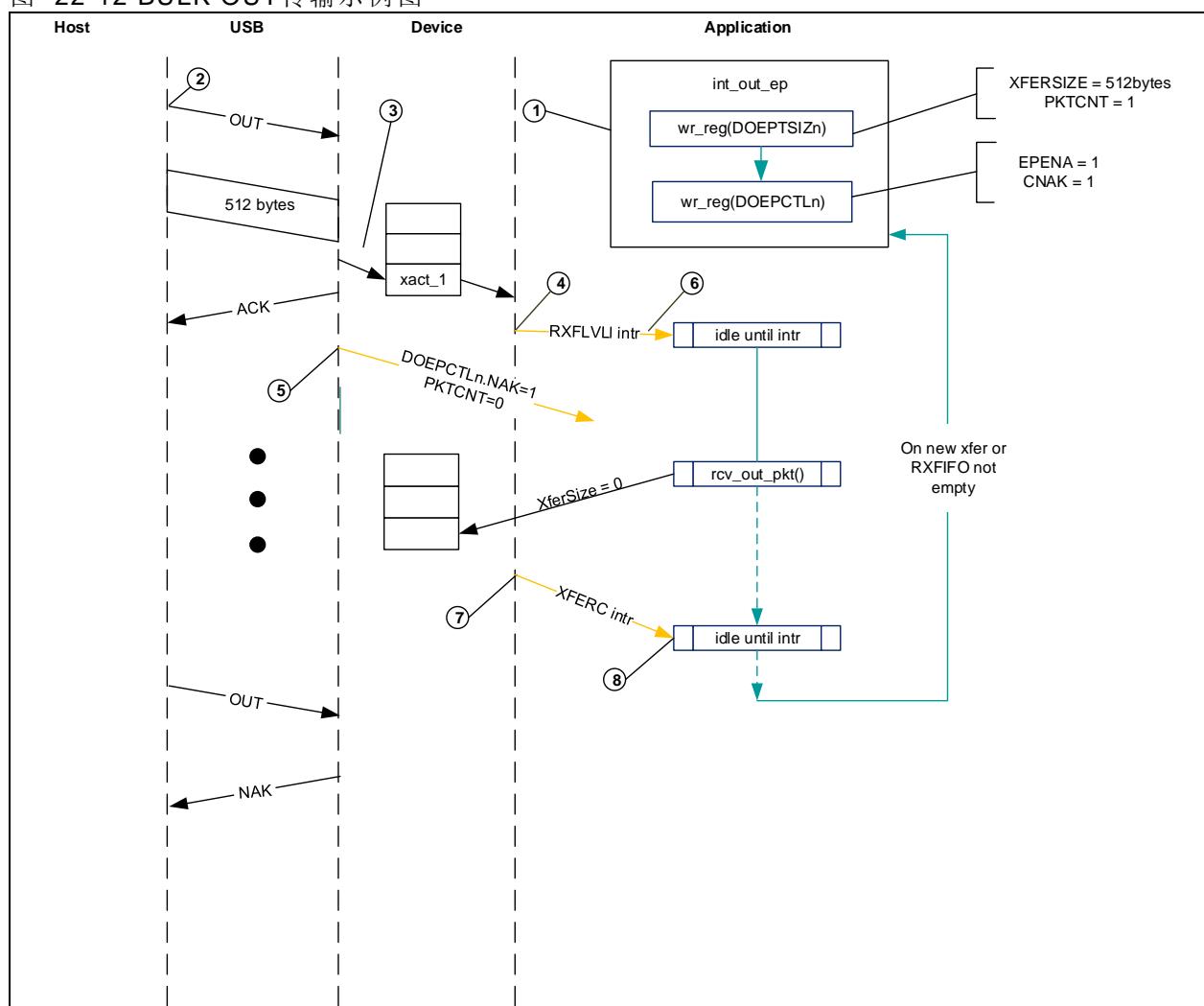
【Application Programming Sequence】应用程序操作流程

1. 根据传输长度和相应的包数目来设置 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DOEPTSIZx)。
2. 根据端点特性设置 OTGFS 设备 OUT 端点 x 控制寄存器(OTGFS_DOEPCTLx)，并设置端点使能位和 ClearNAK 位。
 - OTGFS_DOEPCTLx.EPENA = 0x1
 - OTGFS_DOEPCTLx.CNAK = 0x1
3. 等待 OTGFS 中断寄存器(OTGFS_GINTSTS.RXFLVL)中断，按照“读取 FIFO 包”的流程来读取接收 FIFO 中的所有数据包。
 - 根据传输长度可重复此步骤
4. 置起 OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.XFERC)中断即表示成功完成了非同步 OUT 数据传输。读取 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DOEPTSIZx)来确认收到了多少数据。

【批量 OUT 传输】

下图描绘了从 USB 到 AHB 收到一个单独的批量 OUT 数据包，以及该过程所涉及的相关事件。

图 22-12 BULK OUT 传输示例图



在收到 SetConfiguration/SetInterface 命令后，应用程序将通过设置 OYG_DOEPCTLx.CNAK = 0x1 以及 OTGFS_DOEPCTLx.EPENA = 0x1 来初始化所有的 OUT 端点，并设置 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DOEPTSIZx)中的 XFERSIZE 和 PKTCNT 位。

1. 主机尝试给端点发送数据（OUT 令牌）
2. 当控制器收到 USB 发现的 OUT 令牌时，会将数据存放在接收 FIFO，因为接收 FIFO 里有可用空间。
3. 将完整的数据写入接收 FIFO 之后，控制器会引发 OTGFS 中断寄存器(OTGFS_GINTSTS.RXFLVL)

中断。

4. 在收到 USB 包的包数目后，控制器内部会通过设置该端点的 NAK 位以防止收到更多的数据包。
5. 应用程序处理该中断并从接收 FIFO 读取该数据。
6. 当应用程序读取完所有数据（相当于 XFERSIZE）之后，控制器会生成 OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.XFERC)中断。
7. 应用程序处理该中断，并使用 OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.XFERC)中位后来判断已完成预期的传输。

22.5.4.16 同步 OUT 数据传输

如果需要在上电复位后初始化控制器，应用程序需遵循“OTGFS 初始化”的流程操作。在与主机通讯之前，需要按照“端点初始化”流程对端点进行初始化操作，并参考“读取 FIFO 包”。

本节介绍的是常规同步 OUT 数据传输。

【应用要求】

1. 与非同步 OUT 数据传输的应用程序要求相同。
2. 对于同步 OUT 数据传输，传输长度和包数目位必须设置为一个帧所能接收的最大包长度，不能超过这个数。同步 OUT 数据传输不能跨越超过 1 个帧。
 - $1 \leqslant \text{包数目}[\text{epnum}] \leqslant 3$
3. 如果设备支持同步 OUT 端点，则应用程序必须在周期性帧结束之前（OTGFS 中断寄存器(OTGFS_GINTSTS.EOPF)中断）读取接收 FIFO 中的所有同步 OUT 数据包。
4. 如果要在下一个帧接收数据，必须在产生 OTGFS 中断寄存器(OTGFS_GINTSTS.EOPF)中断以及开始 OTGFS 中断寄存器(OTGFS_GINTSTS.SOF)信号之前使能同步 OUT 端点。

【内部数据流】

1. 同步 OUT 端点的内部数据流与非同步 OUT 端点是一样的，只有细微区别。
2. 如果通过设置端点使能位和清除 NAK 位使能了同步 OUT 端点，则奇/偶帧位也要进行适当设置。只有在满足下列条件时，控制器才能在某个帧的同步 OUT 端点接收数据。
 - OTGFS 设备 OUT 端点 x 控制寄存器(OTGFS_DOEPCTLx.Even)/Odd microframe = OTGFS_DSTS.SOFFN[0]
3. 当应用程序完全读取了接收 FIFO 的同步 OUT 数据包（数据和状态）时，控制器会根据从接收 FIFO 读取的最后一个同步 OUT 数据包的数据 PID 来更新 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DOEPTSIZx.RXDPID)位。

【应用程序操作流程】

1. 设置 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DOEPTSIZx)的传输长度和相应包数目。
2. 根据端点特性设置 OTGFS 设备 OUT 端点 x 控制寄存器(OTGFS_DOEPCTLx)，并设置端点使能位，ClearNAK 和偶/奇帧位。
 - 端点使能 = 0x1
 - CNAK = 0x1
 - 偶/奇帧 = (0x0: 偶; 0x1: 奇)
3. 等待 OTGFS 中断寄存器(OTGFS_GINTSTS.RXFLVL)中断，并读取接收 FIFO 中的所有数据包，详见“读取 FIFO 包”
 - 可以根据传输长度重复此动作。
4. OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.XFERC)中断表示已完成同步 OUT 数据传输。但是此中断并不一定意味着存储器的数据都是好的。
5. 同步 OUT 传输并不一定能检测到该中断信号，但是应用程序可以检测到 OTGFS 中断寄存器(OTGFS_GINTSTS.INCOMPISOOUT)同步 OUT 数据中断，详见“不完整的同步 OUT 数据传输”。
6. 读取 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DOEPTSIZx)来确认所收到的传输长度，并判断帧内所收到的数据是否有效。只有在满足了下列条件时，应用程序才会将存储器收到的数据视为有效数据。
 - OTGFS_DOEPTSIZx.RXDPID = 0xD0 且收到有效数据的 USB 包数目 = 0x1
 - OTGFS_DOEPTSIZx.RXDPID = 0xD1 且收到有效数据的 USB 包数目 = 0x2
 - OTGFS_DOEPTSIZx.RXDPID = 0xD2 且收到有效数据的 USB 包数目 = 0x3

收到有效数据的 USB 包数 = APP 设置的初始包数目 - 控制器更新的最终包数目
应用程序不理会无效数据包。

22.5.4.17 使能同步端点

主机将设置接口控制命令发送给设备后，会使能同步端点。随后，主机就可以发送任意帧内的首个同步 IN

令牌，然后再按照 **BlInterval** 的顺序发送。

但是，在 OTGFS 控制器中，同步支持是基于单个传输级。应用程序必须根据每个帧重新配置控制器。OTGFS 控制会使能将要发生传输的帧之前的那个帧的同步端点。

例如，如果数据要在帧 n 上发送，那么必须使能帧 n-1 的端点。另外，OTGFS 通过设置偶/奇帧位来调度同步传输。

【同步 IN 传输中断】

需要处理好下列中断以确保成功调度同步传输。

- OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.XFERC) (基于端点)
- OTGFS 中断寄存器(OTGFS_GINTSTS.INCOMPISOIN) (全局中断)

【处理同步 IN 传输】

需要按照下列步骤来处理同步 IN 传输：

1. 通过设置 OTGFS 中断屏蔽寄存器(OTGFS_GINTMSK.INCOMISOINMSK)解除 OTGFS 中断寄存器(OTGFS_GINTSTS.incompISOOUT)中断

2. 通过设置 OTGFS 设备 OTGFSIN 端点通用中断屏蔽寄存器(OTGFS_DIEPMSK.XFERCMSK)解除 OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.XFERC)中断

3. 通过执行下列操作来使能同步端点：

- 配置 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DIEPTSIzX)

OTGFS_DIEPTSIzX.XFERSIZE= n * OTGFS_DIEPCTLx.MPS + sp。其中 0 <= n <= 3, 0 <= sp < OTGFS_DIEPCTLx.MPS。当帧内传输的数据长度小于 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx.MPS)，n=0。当帧内传输的数据长度是 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx.MPS)的倍数时，sp=0。

OTGFS_DIEPTSIzX.PKTCNT = 1。

OTGFS 设备端点 x 传输长度寄存器(OTGFS_DIEPTSIzX.MC)的设置值与 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DIEPTSIzX.PKTCNT)一样。

- 配置 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx)

读取 OTGFS 设备状态寄存器(OTGFS_DSTS)来确定当前的帧号

配置 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx.MPS)为最大包长度

配置 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx.USBACTEP)为 0x1

配置 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx.EPTYPE)为 0x1，表示同步

配置 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx.TXFNUM)为端点的 FIFO 号

配置 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx.CNAK)为 0x1

如果 OTGFS_DSTS.SOFFN[0] = 0x0，则 OTGFS_DIEPCTLx.SETEVENFR = 0x1 (否则 OTGFS_DIEPCTLx.SETEVENFR = 0x1)

如果 OTGFS_DSTS.SOFFN[0] = 0x1，则 OTGFS_DIEPCTLx.SETODDFR = 0x1 (否则 OTGFS_DIEPCTLx.SETODDFR = 0x0)

配置 OTGFS_DIEPCTLx.EPENA = 0x1

4. 将端点数据写入相应的发送 FIFO

例如，写入地址范围

- EP1 对应 0x2000 - 0x2FFC
- EP2 对应 0x3000 - 0x3FFC
- EP3 对应 0x3000 - 0x3FFC
- ...

5. 等待中断

● 当 OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.XFERC)中断产生时，清除 OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.XFERC)；对于下一个传输任务，重复步骤 3-5，直到完成传输。

● 当 OTGFS 中断寄存器(OTGFS_GINTSTS.INCOMPISOIN)中断产生时，清除 OTGFS 中断寄存器(OTGFS_GINTSTS.INCOMPISOIN)；对于任何一个同步 IN 端点，当奇/偶位与当前帧号位 0 一致并且在端点保持使能的情况下，控制器在帧结束时会生成该中断。下列情况会导致该中断：

- (1) 帧内没有令牌
- (2) 数据写入接收 FIFO 较晚，数据还没写完，IN 令牌就到了
- (3) IN 令牌出错。

OTGFS 中断寄存器(OTGFS_GINTSTS.INCOMPISOIN)是一个全局中断。所以，当不止一个同步端点处于激活状态时，应用程序必须判断哪一个同步 IN 端点没有完成数据传输。

为了实现这一步，需要读取所有同步端点的 DSTS 和 DIEPCTLx 位。如果当前端点已使能，并且 OTGFS 设备状态寄存器(OTGFS_DSTS.SOFFN)的读取返回值是该端点的目标帧号，那么该端点就未完成传输。应用程序必须准确地跟踪并更新该同步端点的目标帧号。

如果某个端点未完成传输，那么需翻转奇/偶位。

接着：

(1) 当 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx.DPID)位为 1 (奇帧) 时，向 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx.SETD0PID)写 1 使其成为一个偶帧，然后，当下一个帧有 IN 令牌输入时，就会发送数据。

(2) 当 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx.DPID)为 0 (偶帧) 时，向 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx.SETD1PID)写 1 使其变成奇帧，这样，当下一个帧有 IN 令牌输入时，就会发送数据。

22.5.4.18 未完成同步OUT数据传输

若需要在上电复位时初始化控制器，那么应用程序需要根据“OTGFS 初始化”流程进行操作。在与主机通讯之前，必须先按照“端点初始化”的流程初始化端点。本节介绍了当控制器丢失了同步 OUT 数据包时，应用程序的设置流程。

【内部数据流】

1. 对于同步 OUT 端点来说，并不意味着始终会生成 OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.XFERC)中断。如果控制器丢失了同步 OUT 数据包，那么应用程序在遇到下列情况时可能无法检测到 OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.XFERC)中断。

- 当接收 FIFO 无法容纳完整的 ISO OUT 数据包时，控制器会丢失接收到的 ISO OUT 数据。
 - 当接收的同步 OUT 数据包含有 CRC 错误时。
 - 当控制器接收到的同步 OUT 令牌被损坏时。
 - 当应用程序读取接收 FIFO 数据的速度非常慢的时候。
2. 当控制器在传输完成之前检测到所有同步 OUT 端点的周期帧结束时，会生成未完成同步 OUT 数据中断，表示至少有一个同步 OUT 端点没有生成 OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.XFERC)中断。此时，虽然未完成数据传输的这个端点保持使能状态，但是该端点并未进行有效传输。

【应用程序操作流程】

1. 宣告未完成同步 OUT 数据中断表示在当前帧内，至少有一个同步 OUT 端点没有完成数据传输。
2. 如果这是因为该端点的同步 OUT 数据没有完全读取导致的，那么应用程序必须读取接收 FIFO 中的所有同步 OUT 数据（包括数据和状态），再进行下一步处理。
 - 在读取完接收 FIFO 中的所有数据后，应用程序会检测到 OTGFS 设备端点 x 中断寄存器(OTGFS_DOEPINTx.XFERC)中断。此时，应用程序需要按照“控制读传输(SETUP/Data IN/Status OUT)”的流程重新使能该端点来接收下一个帧的同步 OUT 数据。
3. 在收到到未完成同步 OUT 数据中断时，应用程序需要读取所有同步 OUT 端点的控制寄存器来确认当前帧内哪一个端点没有完成传输。如果下列两个条件都满足时，则表示端点就没有完成传输。
 - OTGFS_DOEPCTLx.偶/奇帧位 = OTGFS_DSTS.SOFFN[0]
 - OTGFS_DOEPCTLx.端点使能 = 0x1
4. 必须在检测到 GINTSTS.SOF 中断之前执行上一个步骤，以确保当前帧号不被更改。
5. 对于没有完成传输的同步 OUT 端点，应用程序必须丢弃存储器中的数据，并通过 OTGFS_DOEPCTLx.端点禁用位来禁用此端点。
6. 等待 OTGFS_DOEPINTx.端点禁用中断，并按照“控制读传输(SETUP/Data IN/Status OUT)”的流程使能该端点接收下一个帧的新数据。由于控制器需要花一些时间禁用此端点，所以应用程序在接收到错误数据之后可能无法接收到下一个帧的数据。

22.5.4.19 未完成同步IN数据传输

本节介绍了当同步 IN 数据传输未完成时应用程序该如何操作。

【内部数据流】

1. 在下列情况下，同步 IN 传输被视为未完成。
 - 控制器在不止一个同步 IN 端点接收到已损坏的同步 IN 令牌。此时应用程序会检测到 GINTSTS.未完成同步 IN 传输中断。
 - 应用程序向发送 FIFO 写入完整数据的速度较慢，且还未完成写入就接收到 IN 令牌。此时，应用程序会检测到 OTGFS 设备端点 x 中断寄存器 (OTGFS_DIEPINTx.INTKNTXFEMP) 中断信号。应用程序会忽略此中断，因为这将导致在帧结束时生成 OTGFS_GINTSTS.未完成同步 IN 传输中断。控制器向 USB 发线一个零长度的数据包来回应接收到的 IN 令牌。
2. 无论处于以上哪种情况，应用程序必须尽快停止向发送 FIFO 写数据。
3. 应用程序必须设置该端点的 NAK 位和禁用位。
4. 控制器禁用此端点，清除禁用位，并触发端点禁用中断。

【应用程序操作流程】

1. 当发送 FIFO 为空时，应用程序会忽略任何一个同步 IN 端点上的 OTGFS 设备端点 x 中断寄存器 (OTGFS_DIEPINTx.INTKNTXFEMP) 中断信号，因为这会触发 OTGFS_GINTSTS.未完成同步 IN 传输中断。
2. OTGFS_GINTSTS.未完成同步 IN 传输中断表示至少一个同步 IN 端点没有完成同步 IN 传输。
3. 应用程序必须读取所有同步 IN 端点的端点控制寄存器来确认哪一个端点没有完成 IN 数据传输。
4. 应用程序必须向这些端点的周期发送 FIFO 写入数据。
5. 通过设置 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx)的下列位域来禁用该端点
 - OTGFS_DIEPCTLx.SETNAK = 0x1
 - OTGFS_DIEPCTLx.端点使能 = 0x1
6. DIEPINTx.端点禁用中断表示控制器已禁用了此端点。
7. 此时，应用程序必须通过使能该端点用于传输下一个帧来清空相关发送 FIFO 中的数据或者覆盖 FIFO 中现有的数据。应用程序必须通过 OTGFS 复位寄存器(OTGFS_GRSTCTL)来刷新该数据。

22.5.4.20 周期性IN（中断和同步）数据传输

本节介绍了典型的周期性 IN 数据传输操作。

如果需要在上电复位后初始化控制器，应用程序必须遵循“OTG 初始化”的流程进行操作。在与主机通讯之前，必须按照“端点初始化”流程来初始化端点。

【应用程序要求】

1. “非周期性（批量和控制）IN 数据传输”应用程序要求也适用于周期性 IN 数据传输，除了第 2 点要求略微不同。
 - 应用程序发送的数据仅限于最大包长度的数据包的倍数包，以及短包。只有在满足下列条件时，才能发送几个最大包长度的数据包和短包。
传输长度[epnum] = n * mps[epnum] + sp, (其中 n、i 是 ≥ 0 的整数，且 $0 \leq sp < mps[epnum]$)
如果(sp > 0)，包数[epnum] = n + 1。否则，包数[epnum] = n, mc[epnum] = 包数 [epnum]
● 应用程序不能在传输结束时发送零长度的数据包。但其本身是可以发送一个单独的零长度的数据包，条件是：传输长度[epnum] = 0; 包数目[epnum] = 1; mc[epnum] = 包数目 [epnum]
 - 2. 应用程序一次只能调度传输 1 帧数据。
 - (OTGFS_DIEPTSIzX.MC - 1) * OTGFS_DIEPCTLx.MPS \leq OTGFS_DIEPTSIzX.XFERSIZ \leq OTGFS_DIEPTSIzX.MC * OTGFS_DIEPCTLx.MPS
 - OTGFS_DIEPTSIzX.PKTCNT = OTGFS_DIEPTSIzX.MC
 - 如果 OTGFS_DIEPTSIzX.XFERSIZ < OTGFS_DIEPTSIzX.MC * OTGFS_DIEPCTLx.MPS，最后一个数据包传输是一个短包。
 - 3. 对于周期性 IN 端点，必须在下一个帧传输之前预取 1 个帧数据。可以通过在调度数据传输的帧之前使能周期性 IN 端点 1 帧来完成这个操作。
 - 4. 应用程序在接收周期性 IN 令牌之前必须将帧要传输的完整数据写入发送 FIFO。即使在收到周期 IN 令牌时，发送 FIFO 里缺失了帧需要传输的 1 WORD 数据，控制器将 FIFO 为空来处理。当发送 FIFO 为空时，USB 会针对 ISO IN 端点发送零长度数据包。并针对 INTR IN 端点发送 NAK 握手信号。

【内部数据流】

1. 应用程序必须设置端点控制器的传输长度和包数目位，并使能该端点来传输该数据。
2. 应用程序也可以将所需数据写入相关的发送 FIFO。
3. 每当应用程序将发送 FIFO 写入一个包时，该端点的传输长度会减去包长度。需要持续写入数据直到该端点的传输长度变为 0。
4. 当收到某个周期性端点的 IN 令牌时，应用程序会将数据写入 FIFO（如果有的话）。如果 FIFO 里不存在该帧的完整数据，那么控制器会生成 INTKNTXFEMP 中断。
 - USB 针对同步 IN 端点发送一个零长度的数据包。
 - USB 针对中断 IN 端点发送一个 NAK 握手信号。
5. 端点的包数目在遇到下列情况时会自动减 1：
 - 对于同步端点，发送了长度或非零长度的数据包时
 - 对于中断端点，当发送了 ACK 握手信号
 - 当传输长和包数目都变为 0 时，会生成会该端点的传输完成中断，并清除端点使能位。
6. 在“周期性帧间隔”中（通过设置 OTGFS 设备配置寄存器(OTGFS_DCFG.PERFRINT)），当控制器检测到用于调度当前帧非空的任意一个 IN 端点 FIFO 非空时，控制器会生成 OTGFS 中断寄存器(OTGFS_GINTSTS.INCOMPISOIN)中断。

【应用程序操作流程（帧传输）】

1. 设置 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DIEPTSIZx)
2. 根据端点特性设置 OTGFS 设备端点 x 控制寄存器(OTGFS_DIEPCTLx)并设置 CNAK 和端点使能位。
3. 将下一个帧要传输的数据写入发送 FIFO。
4. 当产生 INTKNTXFEMP 中断表示应用程序尚未将所有待发送的数据写入到发送 FIFO。
5. 如果在检测到该中断时，中断端点已使能，请忽略此中断。如果未使能，请使能该端点以便在下一个 IN 令牌传输数据。如果在检测到该中断时同步端点已使能，详见“未完成同步 IN 数据传输”。
6. 当中断 IN 端点被设置为周期性端点，控制器内部可以处理中断 IN 端点发生的超时情况，并不需要应用程序的介入。因此，应用程序永远无法检测到周期性中断 IN 端点上的 OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.TIMEOUT)中断信号。
7. 产生 OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.XFERC)中断但是没有产生 INTKNTXFEMP 中断表示成功完成了同步 IN 传输。读取 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DIEPTSIZx)时，只有当传输长度为 0 且包数目均为 0 才表示 USB 线上的所有数据均已完成传输。
8. 产生 OTGFS 设备端点 x 中断寄存器(OTGFS_DIEPINTx.XFERC)中断但是不管有没有产生 INTKNTXFEMP 中断表示成功完成了一个中断 IN 传输。读取 OTGFS 设备端点 x 传输长度寄存器(OTGFS_DIEPTSIZx)时，只有当传输长度为 0 且包数目均为 0，才表示 USB 线上的所有数据均已完成传输。
9. 产生了 INCOMPISOIN 中断但没有产生上述所提到的中断即表示控制器没有接收到当前帧发送的至少 1 个周期性 IN 令牌。关于同步 IN 端点，详见“未完成同步 IN 数据传输”。

22.6 OTGFS控制和状态寄存器

应用程序通过 AHB 从接口来读写控制和状态寄存器(CSRx)，从而实现对 OTGFS 控制器的控制。这些寄存器都是 32 位访问的，并且 32 位地址对齐。

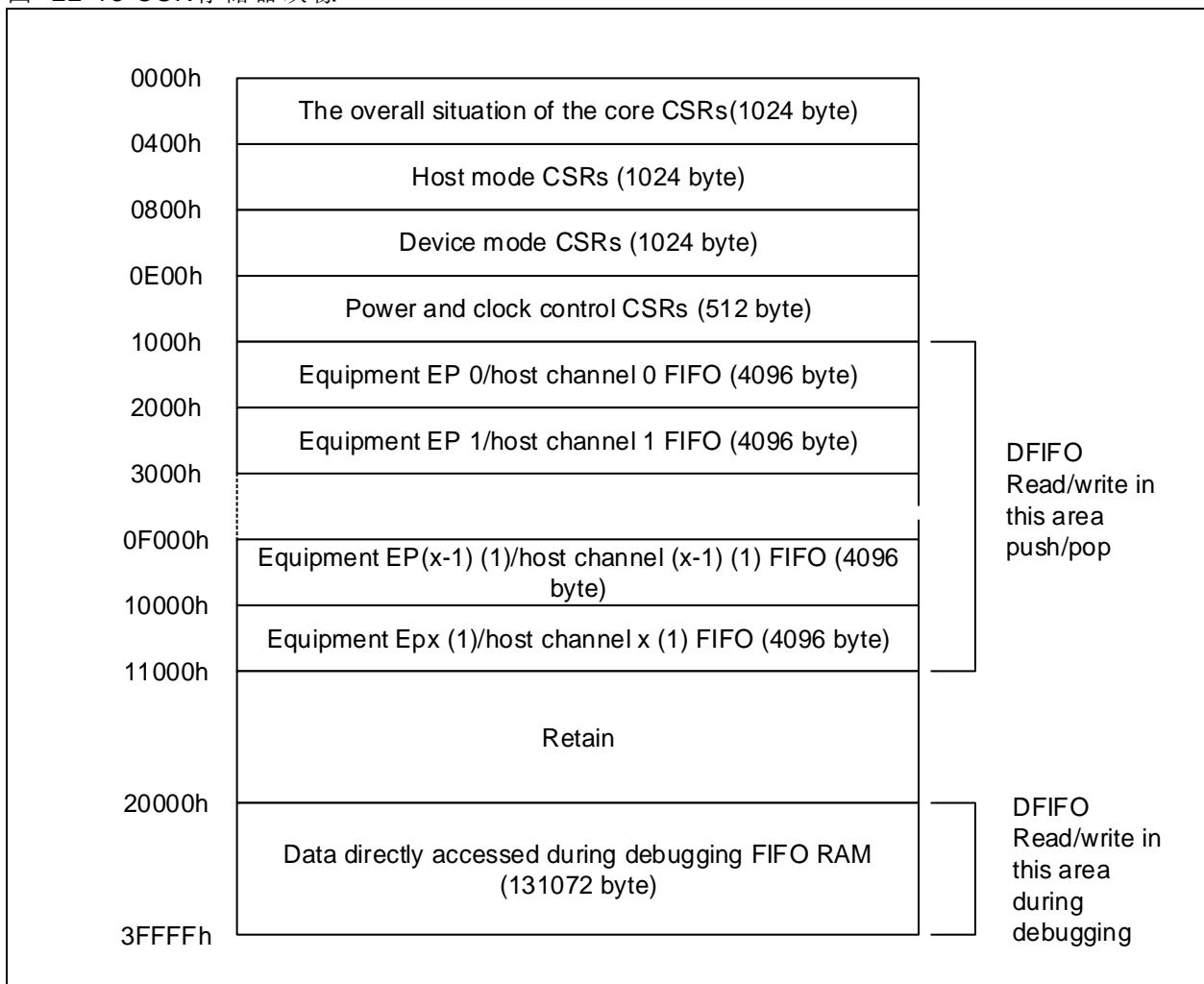
只有控制器全局寄存器，供电和时钟控制寄存器，数据 FIFO 访问寄存器以及主机端口控制和状态寄存器在主机模式和设备模式下都有效。无论 OTGFS 控制器工作在主机模式还是设备模式下，应用程序都不能访问另一种模式下的寄存器组。如果应用程序发生了非法访问，会产生模式不匹配中断并影响控制器中断寄存器的相应位(OTGFS_GINTSTS 寄存器的 MODMIS 位)。

当控制器从一种模式切换到另一种模式时，新模式下的寄存器组都需要和上电复位时一样的重新初始化。必须以字(32 位)的方式操作这些外设寄存器。

22.6.1 CSR寄存器映像

主机模式寄存器和设备模式寄存器占据不同的地址。所有的寄存器都位于 AHB 时钟域。

图 22-13 CSR存储器映像



在设备模式下 x 为 7，在主机模式下 x 为 15

OTGFS 控制和状态寄存器可分为 OTGFS 全局寄存器、主机模式下寄存器、设备模式下寄存器、数据 FIFO(DFIFO)访问寄存器、供电和时钟控制寄存器。

- 1、OTGFS 全局寄存器：这些寄存器在主机模式和设备模式下都有效，寄存器首字母缩写为 G；
- 2、主机模式下寄存器：这些寄存器在每次切换到主机模式时都需要配置，寄存器首字母缩写为 H；
- 3、设备模式下寄存器：这些寄存器在每次切换到设备模式时都需要配置，寄存器首字母缩写为 D；
- 4、数据 FIFO(DFIFO)访问寄存器：这组寄存器列在主机模式和设备模式下都有效，用于读写指定方向的特殊端点或通道的 FIFO。如果一个主机模式下的通道是 IN 类型的，相对应的 FIFO 只能进行读操作。同样地，如果一个主机模式下的通道是 OUT 类型的，相对应的 FIFO 只能进行写操作。
- 5、供电和时钟控制寄存器：只有一个寄存器用来控制供电和时钟控制，此寄存器在设备模式下和主机模式下都有效

22.6.2 OTGFS寄存器地址映象

下表给出了 USB OTG 寄存器映像和复位值。

必须以字（32 位）的方式操作这些外设寄存器。

表 22-4 OTGFS模块的寄存器图及其复位值

寄存器简称	基址偏移量	复位值
OTGFS_GOTGCTL	0x000	0x0001 0000
OTGFS_GOTGINT	0x004	0x0000 0000
OTGFS_GAHBCFG	0x008	0x0000 0000
OTGFS_GUSBCFG	0x00C	0x0000 1440
OTGFS_GRSTCTL	0x010	0x8000 0000
OTGFS_GINTSTS	0x014	0x0400 0020
OTGFS_GINTMSK	0x018	0x0000 0000
OTGFS_GRXSTSR	0x01C	0x0000 0000
OTGFS_GRXSTSP	0x020	0x0000 0000
OTGFS_GRXFISZ	0x024	0x0000 0200
OTGFS_GNPTXFSIZ	0x028	0x0000 0200
OTGFS_GNPTXSTS	0x02C	0x0008 0200
OTGFS_GCCFG	0x038	0x0080 0000
OTGFS_GUID	0x03C	0x0000 1000
OTGFS_HPTXFSIZ	0x100	0x0200 0600
OTGFS_DIEPTXF1	0x104	0x0200 0400
OTGFS_DIEPTXF2	0x108	0x0200 0400
OTGFS_DIEPTXF3	0x10C	0x0200 0400
OTGFS_DIEPTXF4	0x110	0x0200 0400
OTGFS_DIEPTXF5	0x114	0x0200 0400
OTGFS_DIEPTXF6	0x118	0x0200 0400
OTGFS_DIEPTXF7	0x11C	0x0200 0400
OTGFS_DIEPTXF8	0x120	0x0200 0400
OTGFS_DIEPTXF9	0x124	0x0200 0400
OTGFS_DIEPTXF10	0x128	0x0200 0400
OTGFS_DIEPTXF11	0x12C	0x0200 0400
OTGFS_DIEPTXF12	0x130	0x0200 0400
OTGFS_DIEPTXF13	0x134	0x0200 0400
OTGFS_DIEPTXF14	0x138	0x0200 0400
OTGFS_DIEPTXF15	0x13C	0x0200 0400
OTGFS_DIEPTXF16	0x140	0x0200 0400
OTGFS_HCFG	0x400	0x0000 0000
OTGFS_HFIR	0x404	0x0000 EA60
OTGFS_HFNUM	0x408	0x0000 3FFF
OTGFS_HPTXSTS	0x410	0x0008 0100
OTGFS_HAINT	0x414	0x0000 0000
OTGFS_HAINTMSK	0x418	0x0000 0000
OTGFS_HPRT	0x440	0x0000 0000
OTGFS_HCCHAR0	0x500	0x0000 0000
OTGFS_HCINT0	0x508	0x0000 0000

OTGFS_HCINTMSK0	0x50C	0x0000 0000
OTGFS_HCTSIZ0	0x510	0x0000 0000
OTGFS_HCCHAR1	0x520	0x0000 0000
OTGFS_HCINT1	0x528	0x0000 0000
OTGFS_HCINTMSK1	0x52C	0x0000 0000
OTGFS_HCTSIZ1	0x530	0x0000 0000
OTGFS_HCCHAR2	0x540	0x0000 0000
OTGFS_HCINT2	0x548	0x0000 0000
OTGFS_HCINTMSK2	0x54C	0x0000 0000
OTGFS_HCTSIZ2	0x550	0x0000 0000
OTGFS_HCCHAR3	0x560	0x0000 0000
OTGFS_HCINT3	0x568	0x0000 0000
OTGFS_HCINTMSK3	0x56C	0x0000 0000
OTGFS_HCTSIZ3	0x570	0x0000 0000
OTGFS_HCCHAR4	0x580	0x0000 0000
OTGFS_HCINT4	0x588	0x0000 0000
OTGFS_HCINTMSK4	0x58C	0x0000 0000
OTGFS_HCTSIZ4	0x590	0x0000 0000
OTGFS_HCCHAR5	0x5A0	0x0000 0000
OTGFS_HCINT5	0x5A8	0x0000 0000
OTGFS_HCINTMSK5	0x5AC	0x0000 0000
OTGFS_HCTSIZ5	0x5B0	0x0000 0000
OTGFS_HCCHAR6	0x5C0	0x0000 0000
OTGFS_HCINT6	0x5C8	0x0000 0000
OTGFS_HCINTMSK6	0x5CC	0x0000 0000
OTGFS_HCTSIZ6	0x5D0	0x0000 0000
OTGFS_HCCHAR7	0x5E0	0x0000 0000
OTGFS_HCINT7	0x5E8	0x0000 0000
OTGFS_HCINTMSK7	0x5EC	0x0000 0000
OTGFS_HCTSIZ7	0x5F0	0x0000 0000
OTGFS_HCCHAR8	0x600	0x0000 0000
OTGFS_HCINT8	0x608	0x0000 0000
OTGFS_HCINTMSK8	0x60C	0x0000 0000
OTGFS_HCTSIZ8	0x610	0x0000 0000
OTGFS_HCCHAR9	0x620	0x0000 0000
OTGFS_HCINT9	0x628	0x0000 0000
OTGFS_HCINTMSK9	0x62C	0x0000 0000
OTGFS_HCTSIZ9	0x630	0x0000 0000
OTGFS_HCCHAR10	0x640	0x0000 0000
OTGFS_HCINT10	0x648	0x0000 0000
OTGFS_HCINTMSK10	0x64C	0x0000 0000
OTGFS_HCTSIZ10	0x650	0x0000 0000
OTGFS_HCCHAR11	0x660	0x0000 0000
OTGFS_HCINT11	0x668	0x0000 0000
OTGFS_HCINTMSK11	0x66C	0x0000 0000

OTGFS_HCTSIZ11	0x670	0x0000 0000
OTGFS_HCCHAR12	0x680	0x0000 0000
OTGFS_HCINT12	0x688	0x0000 0000
OTGFS_HCINTMSK12	0x68C	0x0000 0000
OTGFS_HCTSIZ12	0x690	0x0000 0000
OTGFS_HCCHAR13	0x6A0	0x0000 0000
OTGFS_HCINT13	0x6A8	0x0000 0000
OTGFS_HCINTMSK13	0x6AC	0x0000 0000
OTGFS_HCTSIZ13	0x6B0	0x0000 0000
OTGFS_HCCHAR14	0x6C0	0x0000 0000
OTGFS_HCINT14	0x6C8	0x0000 0000
OTGFS_HCINTMSK14	0x6CC	0x0000 0000
OTGFS_HCTSIZ14	0x6D0	0x0000 0000
OTGFS_HCCHAR15	0x6E0	0x0000 0000
OTGFS_HCINT15	0x6E8	0x0000 0000
OTGFS_HCINTMSK15	0x6EC	0x0000 0000
OTGFS_HCTSIZ15	0x6F0	0x0000 0000
OTGFS_DCFG	0x800	0x0820 0000
OTGFS_DCTL	0x804	0x0000 0002
OTGFS_DSTS	0x808	0x0000 0010
OTGFS_DIEPMSK	0x810	0x0000 0000
OTGFS_DOEPMSK	0x814	0x0000 0000
OTGFS_DAINT	0x818	0x0000 0000
OTGFS_DAINTMSK	0x81C	0x0000 0000
OTGFS_DIEPEMPMSK	0x834	0x0000 0000
OTGFS_DIEPCTL0	0x900	0x0000 0000
OTGFS_DIEPINT0	0x908	0x0000 0080
OTGFS_DIEPTSIZ0	0x910	0x0000 0000
OTGFS_DTXFSTS0	0x918	0x0000 0200
OTGFS_DIEPCTL1	0x920	0x0000 0000
OTGFS_DIEPINT1	0x928	0x0000 0080
OTGFS_DIEPTSIZ1	0x930	0x0000 0000
OTGFS_DTXFSTS1	0x938	0x0000 0200
OTGFS_DIEPCTL2	0x940	0x0000 0000
OTGFS_DIEPINT2	0x948	0x0000 0080
OTGFS_DIEPTSIZ2	0x950	0x0000 0000
OTGFS_DTXFSTS2	0x958	0x0000 0200
OTGFS_DIEPCTL3	0x960	0x0000 0000
OTGFS_DIEPINT3	0x968	0x0000 0080
OTGFS_DIEPTSIZ3	0x970	0x0000 0000
OTGFS_DTXFSTS3	0x978	0x0000 0200
OTGFS_DIEPCTL4	0x980	0x0000 0000
OTGFS_DIEPINT4	0x988	0x0000 0080
OTGFS_DIEPTSIZ4	0x990	0x0000 0000
OTGFS_DTXFSTS4	0x998	0x0000 0200

OTGFS_DIEPCTL5	0x9A0	0x0000 0000
OTGFS_DIEPINT5	0x9A8	0x0000 0080
OTGFS_DIEPTSIZ5	0x9B0	0x0000 0000
OTGFS_DTXFSTS5	0x9B8	0x0000 0200
OTGFS_DIEPCTL6	0x9C0	0x0000 0000
OTGFS_DIEPINT6	0x9C8	0x0000 0080
OTGFS_DIEPTSIZ6	0x9D0	0x0000 0000
OTGFS_DTXFSTS6	0x9D8	0x0000 0200
OTGFS_DIEPCTL7	0x9E0	0x0000 0000
OTGFS_DIEPINT7	0x9E8	0x0000 0080
OTGFS_DIEPTSIZ7	0x9F0	0x0000 0000
OTGFS_DTXFSTS7	0x9F8	0x0000 0200
OTGFS_DOEPCTL0	0xB00	0x0000 8000
OTGFS_DOEPINT0	0xB08	0x0000 0000
OTGFS_DOEPTSIZ0	0xB10	0x0000 0000
OTGFS_DOEPCTL1	0xB20	0x0000 0000
OTGFS_DOEPINT1	0xB28	0x0000 0000
OTGFS_DOEPTSIZ1	0xB30	0x0000 0000
OTGFS_DOEPCTL2	0xB40	0x0000 0000
OTGFS_DOEPINT2	0xB48	0x0000 0000
OTGFS_DOEPTSIZ2	0xB50	0x0000 0000
OTGFS_DOEPCTL3	0xB60	0x0000 0000
OTGFS_DOEPINT3	0xB68	0x0000 0000
OTGFS_DOEPTSIZ3	0xB70	0x0000 0000
OTGFS_DOEPCTL4	0xB80	0x0000 0000
OTGFS_DOEPINT4	0xB88	0x0000 0000
OTGFS_DOEPTSIZ4	0xB90	0x0000 0000
OTGFS_DOEPCTL5	0xBA0	0x0000 0000
OTGFS_DOEPINT5	0xBA8	0x0000 0000
OTGFS_DOEPTSIZ5	0xBB0	0x0000 0000
OTGFS_DOEPCTL6	0xBC0	0x0000 0000
OTGFS_DOEPINT6	0xBC8	0x0000 0000
OTGFS_DOEPTSIZ6	0xBD0	0x0000 0000
OTGFS_DOEPCTL7	0xBE0	0x0000 0000
OTGFS_DOEPINT7	0xBE8	0x0000 0000
OTGFS_DOEPTSIZ7	0xBF0	0x0000 0000
OTGFS_PCGCCTL	0xE00	0x0000 0000

22.6.3 OTGFS全局寄存器

该寄存器适用于主机模式和设备模式，在两种模式之间切换时不需要重新配置。

22.6.3.1 OTGFS状态控制器(OTGFS_GOTGCTL)

OTG 控制和状态寄存器用于控制 OTG 功能并反映其功能状态。

域	简称	复位值	类型	功能
位 31: 22	保留	0x0000	resd	请保持默认值。
位 21	CURMOD	0x0	ro	当前工作模式(Current Mode of Operation) 适用模式：主机模式和设备模式 该位指示当前模式。 0: 设备模式 1: 主机模式
位 20: 17	保留	0x0000	resd	请保持默认值。
位 16	CONIDSTS	0x1	ro	连接器 ID 状态(Connector ID Status) 表示连接器 ID 状态。 0: OTGFS 控制器处于 A-设备模式 1: OTGFS 控制器处于 B-设备模式
位 15: 0	保留	0x0000	resd	请保持默认值。

22.6.3.2 OTGFS OTG中断状态控制器(OTGFS_GOTGINT)

应用程序可以通过读此寄存器得知发生了何种 OTG 中断，并可以通过写此寄存器清除对应的 OTG 中断。

域	简称	复位值	类型	功能
位 31: 3	保留	0x0000	resd	请保持默认值。
位 2	SESENDDET	0x0	rw1c	适用模式：主机模式和设备模式 会话结束检测(Session End Detected) 当控制器检测到 Bvalid (此芯片即 Vbus) 信号断开时置起此位。该寄存器只能由硬件置 1，软件可通过写 1 清除此位。
位 1: 0	保留	0x0000	resd	请保持默认值。

22.6.3.3 OTGFS AHB配置寄存器(OTGFS_GAHBCFG)

此寄存器用于在上电或改变控制器模式时配置控制器。此寄存器主要配置一些和 AHB 相关的参数。在初始化配置完成后，不要再修改此寄存器。应用程序在开始向 AHB 或 USB 传送数据前必需先配置好此寄存器。

域	简称	复位值	类型	功能
位 31: 9	保留	0x0000000	resd	保持默认值。
位 8	PTXFEMPLVL	0x0	rw	适用模式：仅适用于主机模式 周期性发送 FIFO 空级别(Periodic TxFIFO Empty Level) 表示何时会触发控制器中断寄存器(GINTSTS.PTXFEMP)的周期性发送 FIFO 空中断位。 0: GINTSTS.PTXFEMP 中断表示周期性发送 FIFO 是半空 1: GINTSTS.PTXFEMP 中断表示周期性发送 FIFO 是全空
位 7	NPTXFEMPLVL	0x0	rw	适用模式：主机模式和设备模式 非周期性发送 FIFO 空级别(Non-Periodic TxFIFO Empty Level) 在主机模式下，该位表示何时会触发控制中断寄存器(GINTSTS.NPTXFEMP)的非周期性发送 FIFO 空中断位。 在设备模式下，该位表示何时会触发 IN 端点发送 FIFO 空中断(DIEPINTn.TxFEMP)。 0: DIEPINTn.TxFEMP 中断表示 IN 端点发送 FIFO 是半空 1: DIEPINTn.TxFEMP 中断表示 IN 端点发送 FIFO 是全空

位 6: 1	保留	0x00	resd	保持默认值。 模式: 主机模式和设备模式 全局中断屏蔽(Global Interrupt Mask)
位 0	GLBINTMSK	0x0	rw	应用程序通过该位来屏蔽/取消屏蔽中断线发给自己的中断。无论该位是否置起, 控制器仍会更新中断状态寄存器。 0: 屏蔽对应用程序的中断 1: 不屏蔽对应用程序的中断

22.6.3.4 OTGFS_USB配置寄存器(OTGFS_GUSBCFG)

该寄存器用于在上电后或者在切换主机模式或设备模式时配置控制器。该寄存器包含了 USB 和 USB-PHY 相关的参数。应用程序在处理 AHB 或 USB 事务之前, 必须先设置该寄存器。初始化配置之后, 请不要再修改本寄存器。

域	简称	复位值	类型	功能
位 31	COTXPKT	0x0	rw	适用模式: 主机模式和设备模式 发送包损坏(Corrupt Tx packet) 该位仅用于调试。请勿将该位设置为 1。
位 30	FDEVMODE	0x0	rw	适用模式: 主机模式和设备模式 强制设备模式(Force Device Mode) 无论 ID 输入引脚的状态如何, 向该位写 1 可强制控制器进入设备模式。 0: 普通模式 1: 强制设备模式 将该置起后, 应用程序必须等待至少 25ms 才能使设置生效。
位 29	FHSTMODE	0x0	rw	适用模式: 主机模式和设备模式 强制主机模式(Force Host Mode) 无论 ID 输入引脚的状态如何, 向该位写 1 可强制控制器进入主机模式。 0: 普通模式 1: 强制主机模式 将该置起后, 应用程序必须等待至少 25ms 才能使设置生效。
位 28: 15	保留	0x0000	resd	保持默认值。
位 14	保留	0x0	resd	保持默认值。
位 13: 10	USBTRDTIM	0x5	rw	适用模式: 仅适用设备模式 USB 周转时间(USB Turnaround Time) 以 PHY 时钟为单位设置周转时间。规定了 MAC 向包 FIFO 控制器 (PFC) 发起请求从 DFIFO (SPRAM) 提取数据的响应时间。这些位必须配置为: 0101: 当 MAC 接口为 16-bit UTMI+时 1001: 当 MAC 接口为 8-bit UTMI+时。 注意: 以上数值是基于最低 30MHz 的 AHB 频率计算得出的。USB 周转时间对于用到长线和 5-Hubs 的认证至关重要, 所以如果你需要 AHB 在低于 30 MHz 的频率下运行, 而且 USB 周转时间不重要的话, 可以将这些位域的值设置大一些。
位 9: 3	保留	0x08	resd	保持默认值。
位 2: 0	TOUTCAL	0x0	rw	适用模式: 主机模式和设备模式 全速超时校准(FS Timeout Calibration) 将应用程序在这些位域设置的 PHY 时钟数添加到全速数据包间超时时段内, 以补偿 PHY 引起的额外延迟。这个动作可能是必须的, 因为 PHY 在生成线路状态条件时引起的延迟会因 PHY 不同而有所不同。 全速模式下, USB 标准超时值是 16~18 个 bit(包含 18)时间。应用程序必须要根据枚举的速度来设置此位域。每个 PHY 时钟增加的 bit 时间为 0.25bit 时间

22.6.3.5 OTGFS复位寄存器(OTGFS_GRSTCTL)

应用程序通过本寄存器来复位控制器的各硬件模块。

域	简称	复位值	类型	功能
位 31	AHBIDLE	0x1	ro	适用模式：主机模式和设备模式 AHB 主机空闲(AHB Master Idle) 指示 AHB 主机状态机处于空闲状态。
位 30: 11	保留	0x000	resd	保持默认值。
位 10: 6	TXFNUM	0x00	rw	适用模式：主机模式和设备模式 发送 FIFO 编号(TxFIFO Number) 此位域表示哪个 FIFO 需要通过发送 FIFO 刷新位(TxFIFO Flush)来刷新。除非控制器已清除 TxFIFO Flush 位，否则不允许更改此位域。 00000: - 主机模式下非周期性 Tx FIFO - 设备模式下 Tx FIFO 0 00001: - 主机模式下周期性 Tx FIFO - 设备模式下 Tx FIFO 1 00010: - 设备模式 Tx FIFO 2 ... 01111: - 设备模式下 Tx FIFO 15 10000: - 刷新位于设备模式或主机模式下的所有发送 FIFO
位 5	TXFFLSH	0x0	rw1s	适用模式：主机模式和设备模式 发送 FIFO 刷新(TxFIFO Flush) 该位有选择性地刷新单独一个的或所有的发送 FIFO，但是必须确保控制器没有正在处理的事务。 应用程序必须先确认控制器没有读/写 Tx FIFO，才能对此位进行写操作。 通过这些寄存器来验证： 读：NAK 有效中断(NAK Effective Interrup)确保控制器没有读取 FIFO 写：GRSTCTL.AHBIDLE 确保控制器没有对 FIFO 进行写操作。 在重新配置 FIFO 时，通常建议用户进行刷新操作(Flushing)。 在设备端点禁用期间，也建议使用 FIFO 刷新操作。应用程序必须要等待控制器清除此位之后才能进行其它操作。该位需要 8 个时钟来完成清除（以 phy_clk 或 hclk 的较慢的那个时钟为准来计算）。
位 4	RXFFLSH	0x0	rw1s	适用模式：主机模式和设备模式 接收 FIFO 刷新(RxFIFO Flush) 应用程序可以通过该位来刷新整个接收 FIFO，但必须先确保控制器没有正在处理的事务。应用程序必须先确认控制器没有读/写 Rx FIFO，才能对此位进行写操作。 应用程序必须要等待控制器清除此位之后才能进行其它操作。该位需要 8 个时钟周期来完成清除（以 PHY 或 AHB 最慢的那个时钟为准来计算）。
位 3	保留	0x0	resd	保持默认值。
位 2	FRMCNTRST	0x0	rw1s	适用模式：仅适用主机模式 主机帧计数器复位(Host Frame Counter Reset) 应用程序通过此位来复位控制器的帧数计数器。将帧计数器复位之后，控制器随后发出的 SOF 的帧号为 0。 如果应用程序向该位写 1，它可能无法读取到这个值，因为该位会在几个时钟周期之后被控制器清除。
位 1	PIUSFTRST	0x0	rw1s	适用模式：主机模式和设备模式 PIU 全速专用控制器软件复位(PIU FS Dedicated Controller Soft Reset) 用于复位 PIU 全速专用控制器

位 0	CSFTRST	0x0	rw1s	<p>PIU 全速专用控制器中的所有模块状态机即恢复到空闲状态。在发生 PHY 错误（比如操作中断或 bubble）导致 PHY 保持在接收状态的时长超过 1 个帧时，可使用此位来复位 PIU 的全速专用控制器。</p> <p>该位可自动清除，而且控制器中在所有必要逻辑电路复位之后会清除此位。</p> <p>适用模式：主机模式和设备模式 控制器软件复位(Core Soft Reset) 复位 hclk 和 phy_clock 域，如下所示： 清除所有中断以及 CSR 寄存器，除如下寄存器位之外： - HCFG.FSLSPCS - DCFG.DECSPD - DCTL.SFTDIS 将所有模块状态机（除了 AHB 从机）复位到空闲状态，并清空所有的发送 FIFO 和接收 FIFO。 在完成最后一个阶段的 AHB 数据传输之后，AHB 主机上的所有任务都会尽可能快地结束。USB 上的所有任务会立即停止。 应用程序可以随时对该位进行写操作来复位控制器。此位可自动清除，控制器在将所有必要的逻辑电路复位之后会清除此位，控制器可能需要花几个时钟周期来清除，具体时间取决于控制器的当前所处的状态。清除此位后，应用程序必须要等待至少 3 个 PHY 时钟之后才能访问 PHY 域（同步延迟）。 另外，应用程序在开始其它操作之前，必须要确保本寄存器的位 31 是置 1(AHB 主机是空闲状态)。 通常来讲，使用软件复位的情况为：在进行软件开发，以及当用户对上述所列的 USB 配置寄存器的 PHY 选择位进行了动态修改的情况下，需要通过软件复位。在修改 PHY 时，需要选择相应的 PHY 时钟并运用于 PHY 域。选择了新的时钟之后，必须复位 PHY 域才能进行正常操作。</p>
-----	---------	-----	------	--

22.6.3.6 OTGFS 中断寄存器(OTGFS_GINTSTS)

本寄存器会因当前模式（设备模式或主机模式）发生系统事件而中断应用程序，如图 22-2 所示。本寄存器的一些位仅适用于主机模式，一些位仅适用于设备模式。另外，该寄存器会指示当前位于哪种操作模式。

FIFO 状态寄存器中断位是只读的。一旦软件在处理这些中断时读写了 FIFO，则 FIFO 中断条件会自动被清除。

初始化时，应用程序在使能某个中断位之前，必须先清除 GINTSTS 寄存器，以避免在初始化之前产生中断。

域	简称	复位值	类型	功能
位 31	WKUPINT	0x0	rw1c	<p>适用模式：主机模式和设备模式 检测恢复/远程唤醒信号产生中断(Resume/Remote Wakeup Detected Interrupt)</p> <p>设备模式：仅当 USB 总线上检测到主机触发的恢复信号时，才产生中断</p> <p>主机模式：仅当 USB 总线上检测到设备触发的远程唤醒信号时，才产生中断</p>
位 30	保留	0x0	resd	保持默认值。
位 29	DISCONINT	0x0	rw1c	<p>适用模式：仅适用于主机模式 检测到断开事件产生中断(Disconnect Detected Interrupt)</p> <p>当检测到设备断开时，即产生中断</p>
位 28	CONIDSCHG	0x0	rw1c	<p>适用模式：主机模式和设备模式 连接器 ID 状态变化(Connector ID Status Change)</p> <p>当检测到连接器 ID 状态发生变化时，控制器将此位置起</p>
位 27	保留	0x0	resd	保持默认值。
位 26	PTXFEMP	0x1	ro	<p>适用模式：仅适用于主机模式 周期性发送 FIFO 空(Periodic TxFIFO Empty)</p>

				当周期性发送 FIFO (Periodic Transmit FIFO) 处于半空或全空状态，且在周期性请求队列中有空间可写入一个请求时，即产生中断。具体是半空还是全空，取决于控制器 AHB 配置寄存器的周期性发送 FIFO 空级别位(Periodic TxFIFO Empty Level)。
				主机通道中断(Host Channels Interrupt) 控制器通过将该位置起，来指示控制器（主机模式下）的某一个通道有待处理的中断。应用程序必须读取主机全通道中断寄存器(Host All Channels Interrupt)以确定产生中断的具体通道编号，然后读取相应的主机通道编号中断寄存器(Host Channel-n Interrupt)以获取中断源。 应用程序必须通过清除 HCINTn (Host All Channels Interrupt)寄存器的相应状态位来清除该位。
				主机端口中断(Host Port Interrupt) 控制器通过将该位置起，来指示在主机模式下某个端口状态发生改变。应用程序必须通过读取主机端口控制和状态寄存器(Host Port Control and Status)来确认中断源。应用程序必须通过清除主机端口控制和状态寄存器(Host Port Control and Status)来清除此位。
				位 23: 22 保留 0x0 resd 保持默认值。
				未完成周期性传输(Incomplete Periodic Transfer) 适用模式：仅适用于主机模式 在主机模式下，如果当前帧尚有未完成的周期性传输待处理时，控制器会置起此中断位。
				位 21 INCOMPPIP INCOMPISOOUT 0x0 rw1c 未完成同步 OUT 传输(Incomplete Isochronous OUT Transfer) 适用模式：仅适用于设备模式 在设备模式下，控制器置起此中断位以指示当前帧至少存在一个未完成传输的同步 OUT 端点。该中断会随同本寄存器的周期性帧结束中断位(End of Periodic Frame Interrupt)一同生成。
				位 20 INCOMPISOIN 0x0 rw1c 适用模式：仅适用于设备模式 未完成同步 IN 传输(Incomplete Isochronous IN Transfer) 控制器置起此中断以指示当前帧至少存在一个未完成传输的同步 IN 端点。该中断会随同本寄存器的周期性帧结束中断位(End of Periodic Frame Interrupt)一同生成。
				位 19 OEPTINT 0x0 ro 适用模式：仅适用于设备模式 OUT 端点中断(OUT Endpoints Interrupt) 控制器通过置起此位来指示控制器的某一个 OUT 端点有待处理的中断。应用程序必须通过读取设备全部端点中断寄存器(Device All Endpoints Interrupt)以确定该中断发生在哪一个 OUT 端点号上，接着会读取相应的设备 OUT 端点号中断寄存器(Device OUT Endpoint-n Interrupt)来确定本次中断源。应用程序必须通过清除设备 OUT 端点中断寄存器(Device OUT Endpoint-n Interrupt)的相应状态位来清除此位。
				位 18 IEPTINT 0x0 ro 适用模式：仅适用于设备模式 IN 端点中断(IN Endpoints Interrupt) 控制器通过置起此位来指示控制器的某一个 IN 端点有待处理的中断（设备模式下）。应用程序必须通过读取设备全部端点中断寄存器(Device All Endpoints Interrupt)以确定该中断发生在哪一个 IN 端点号上，接着会读取相应的设备 IN 端点号中断寄存器(Device IN Endpoint-n Interrupt)来确定本次中断源。应用程序必须通过清除相应的 Device IN Endpoint-n Interrupt 寄存器的相应状态位来清除此位。
				位 17: 16 保留 0x0 resd 保持默认值。
				位 15 EOPF 0x0 rw1c 适用模式：仅适用于设备模式 周期性帧结束中断(End of Periodic Frame Interrupt) 该位表示当前帧已经到达了设备配置寄存器(Device Configuration)的周期帧间隔时间位所设置的周期。
				位 14 ISOOUTDROP 0x0 rw1c 适用模式：仅适用于设备模式

				同步 OUT 包丢失中断(Isochronous OUT Packet Dropped Interrupt) 控制器在以下情况会置起该位：控制器因为接收 FIFO 没有足够的空间来容纳同步 OUT 端点所需的最大容量的数据包而导致其无法向接收 FIFO 写入一个同步 OUT 包。
位 13	ENUMDONE	0x0	rw1c	适用模式：仅适用于设备模式 完成枚举(Enumeration Done) 控制器将该位置起以表示已完成速度枚举。 应用程序必须通过读取设备状态寄存器(Device Status)来获取枚举的速度信息。
位 12	USBRST	0x0	rw1c	适用模式：仅适用于设备模式 USB 复位(USB Reset) 控制器将该位置起以表示检测到 USB 总线上的复位信号。
位 11	USBSUSP	0x0	rw1c	适用模式：仅适用于设备模式 USB 挂起(USB Suspend) 控制器将该位置起以表示检测到 USB 总线上的挂起信号。当总线信号在很长一段时间内没有活动时，控制器进入挂起状态。
位 10	ERLYSUSP	0x0	rw1c	适用模式：仅适用于设备模式 早期挂起(Early Suspend) 控制器将该位置起以表示检测到 USB 总线空闲状态已持续 3ms。
位 9: 8	保留	0x0	resd	保持默认值。
位 7	GOUTNAKEFF	0x0	ro	适用模式：仅适用于设备模式 全局 OUT NAK 生效(Global OUT NAK Effective) 该位表示设备控制寄存器(Device Control)设置 Global OUT NAK 位已生效。通过向设备控制寄存器(Device Control)写入 Clear Global OUT NAK 位可以清除该位。
位 6	GINNAKEFF	0x0	ro	适用模式：仅适用于设备模式 全局非周期性 IN NAK 生效(Global IN Non-periodic NAK Effective) 该位表示由应用程序所设置的设备控制寄存器(Device Control)的 Set Global Non-periodic IN NAK 位已生效。即，控制器对应用程序所设置的 Global IN NAK 位进行了采样。通过向设备控制寄存器(Device Control)写入 Clear Global Non-periodic IN NAK 位可以清除该位。此中断并不一定意味着 USB 总线上发送了 NAK 握手信号。STALL 位的优先级高于 NAK 位。
位 5	NPTXFEMP	0x1	ro	适用模式：主机模式和设备模式 非周期性发送 FIFO 空(Non-periodic TxFIFO Empty) 当非周期性发送 FIFO 处于半空或全空状态，并且有足够的空间可以向非周期性发送请求队列(Non-periodic Transmit Request Queue)写入至少一个请求时，即产生中断。具体是半空还是全空，取决于控制器 AHB 配置寄存器(Core AHB Configuration)的非周期性发送 FIFO 空级别位(Non-periodic TxFIFO Empty Level)。
位 4	RXFLVL	0x0	ro	适用模式：主机模式和设备模式 接收 FIFO 非空(RxFIFO Non-Empty) 表示接收 FIFO 中至少有一个包等待读取。
位 3	SOF	0x0	rw1c	适用模式：主机模式和设备模式 帧首包(Start of Frame) 在主机模式下，控制器将该位置起以指示 USB 总线发送了 SOF (全速)或者 Keep-Alive(低速)信号。 应用程序必须将位置 1 才能清除此中断。 在设备模式下，控制器将该位置起以表示 USB 总线接收到 SOF 令牌。应用程序必须读取设备状态寄存器(Device Status register)来获取当前帧号。只有当控制器工作在全速模式时，才能产生此中断。该位只能由控制器设置，应用程序必须写 1 才能清除此位。 注意：上电复位后立即读取该寄存器可能会返回 0x1。如果在上电复位后立即读取寄存器的值为 0x1，并不表示已

				发送 SOF(主机模式下)或已收到 SOF(设备模式下)。只有当主机和设备建立有效连接后，读取的寄存器的值才是有效的。如果在上电复位后将该位置起，那么应用程序可清除此位。
位 2	OTGINT	0x0	ro	适用模式：主机模式和设备模式 OTG 中断(OTG Interrupt) 控制器将该位置起以表示产生了一个 OTG 协议事件。应用程序必须读取 OTG 中断状态寄存器(OTG Interrupt Status)来确定是什么事件引起的中断。应用程序必须通过清除 OTG 中断状态寄存器(OTG Interrupt Status)的相应状态位以清除此位。
位 1	MODEMIS	0x0	rw1c	适用模式：主机模式和设备模式 模式不匹配中断(Mode Mismatch Interrupt) 当应用程序尝试访问以下内容时，控制器会置起此位： 当控制器运行在设备模式下时却尝试访问主机模式下的寄存器 当控制器运行在主机模式下时却尝试访问设备模式下的寄存器 在 AHB 上完成对寄存器的访问之后会出现 OKAY 响应，但控制器内部会忽略此种操作，所以不会影响到控制器的运行。 此位只能由控制器设置，应用程序通过写 1 才能清除此位。
位 0	CURMOD	0x0	ro	适用模式：主机模式和设备模式 当前运行模式(Current Mode of Operation) 此位指示的是当前的运行模式。 0：设备模式 1：主机模式

22.6.3.7 OTGFS 中断屏蔽寄存器(OTGFS_GINTMSK)

本寄存器与“中断寄存器”(Interrupt Register)互相配合来中断应用程序。屏蔽某一个中断位后，就不会生成与该中断位相关的中断。然而与该中断所对应的中断寄存器位仍然是置起状态。

屏蔽中断：0

解除中断：1

域	简称	复位值	类型	功能
位 31	WKUPINTMSK	0x0	rw	适用模式：主机模式和设备模式 屏蔽由恢复/远程唤醒检测到的中断(Resume/Remote Wakeup Detected Interrupt Mask)
位 30	保留	0x0	resd	保持默认值。
位 29	DISCONINTMSK	0x0	rw	适用模式：主机模式和设备模式 屏蔽断开事件检测到的中断(Disconnect Detected Interrupt Mask)
位 28	CONIDSCHGMSK	0x0	rw	适用模式：主机模式和设备模式 屏蔽连接器 ID 状态改变(Connector ID Status Change Mask)
位 27	保留	0x0	resd	保持默认值。
位 26	PTXFEMPMSK	0x0	rw	适用模式：仅适用于主机模式 屏蔽周期性发送 FIFO 空(Periodic TxFIFO Empty Mask)
位 25	HCHINTMSK	0x0	rw	适用模式：仅适用于主机模式 主机通道中断屏蔽(Host Channels Interrupt Mask)
位 24	PRTINTMSK	0x0	ro	适用模式：仅适用于主机模式 主机端口中断屏蔽(Host Port Interrupt Mask)
位 23: 22	保留	0x0	resd	保持默认值。
位 21	INCOMPIPMSK INCOMPISOOUTMSK	0x0	rw	未完成周期性传输屏蔽(Incomplete Periodic Transfer Mask) 适用模式：仅适用于主机模式 未完成同步 OUT 传输屏蔽 (Incomplete Isochronous OUT Transfer Mask) 适用模式：仅适用于设备模式
位 20	INCOMISOINMSK	0x0	rw	适用模式：仅适用于设备模式

				未完成同步 IN 传输屏蔽(Incomplete Isochronous IN Transfer Mask)
位 19	OEPTINTMSK	0x0	rw	适用模式：仅适用于设备模式 OUT 端点中断屏蔽(OUT Endpoints Interrupt Mask)
位 18	IEPTINTMSK	0x0	rw	适用模式：仅适用于设备模式 IN 端点中断屏蔽(IN Endpoints Interrupt Mask)
位 17	保留	0x0	rw	保持默认值。
位 16	保留	0x0	resd	保持默认值。
位 15	EOPFMSK	0x0	rw	适用模式：仅适用于设备模式 周期性帧结束中断屏蔽(End of Periodic Frame Interrupt Mask)
位 14	ISOOUTDROPMSK	0x0	rw	适用模式：仅适用于设备模式下的同步 OUT 包丢失中断屏蔽(Device only Isochronous OUT Packet Dropped Interrupt Mask)
位 13	ENUMDONEMSK	0x0	rw	适用模式：仅适用于设备模式 枚举完成中断屏蔽(Enumeration Done Mask)
位 12	USBRSTMSK	0x0	rw	适用模式：仅适用于设备模式 USB 复位中断屏蔽(USB Reset Mask)
位 11	USBSUSPMSK	0x0	rw	适用模式：仅适用于设备模式 USB 挂起中断屏蔽(USB Suspend Mask)
位 10	ERLYSUSPMSK	0x0	rw	适用模式：仅适用于设备模式 早期挂起中断屏蔽(Early Suspend Mask)
位 9: 8	保留	0x0	resd	保持默认值。
位 7	GOUTNAKEFFMSK	0x0	rw	适用模式：仅适用于设备模式 全局 OUT NAK 有效中断屏蔽(Global OUT NAK Effective Mask)
位 6	GINNAKEFFMSK	0x0	rw	适用模式：仅适用于设备模式 全局非周期性 IN NAK 有效中断屏蔽(Global Non-periodic IN NAK Effective Mask)
位 5	NPTXFEMPMISK	0x0	rw	适用模式：主机模式和设备模式 非周期性发送 FIFO 空中断屏蔽(Non-periodic TxFIFO Empty Mask)
位 4	RXFLVLMSK	0x0	rw	适用模式：主机模式和设备模式 接收 FIFO 非空中断屏蔽(Receive FIFO Non-Empty Mask)
位 3	SOFMSK	0x0	rw	适用模式：主机模式和设备模式 帧开始中断屏蔽(Start of Frame Mask)
位 2	OTGINTMSK	0x0	rw	适用模式：主机模式和设备模式 OTG 中断屏蔽(OTG Interrupt Mask)。
位 1	MODEMISMSK	0x0	rw	适用模式：主机模式和设备模式 模式不匹配中断屏蔽_Mode Mismatch Interrupt Mask)
位 0	保留	0x0	resd	保持默认值。

22.6.3.8 OTGFS接收状态调试读/OTG状态读和POP寄存器 (OTGFS_GRXSTSR / OTGFS_GRXSTSP)

读取“接收状态调试读寄存器(Receive Status Debug Read)”会返回接收 FIFO(Receive FIFO)顶层的数据。读取“接收状态读和 PoP 寄存器(Receive Status Read and Pop)”还会弹出接收 FIFO 顶层的数据。

在主机模式和设备模式下，对接收状态内容的解释并不相同。当接收 FIFO 为空时，控制器将忽略接收状态弹出/读取，并返回 0x0000 0000。当控制器中断寄存器(Core Interrupt register) 的接收 FIFO 非空位被置起时，应用程序才能弹出接收状态 FIFO。

主机模式：

域	简称	复位值	类型	功能
位 31: 21	保留	0x000	resd	保持默认值。
位 20: 17	PKTSTS	0x0	ro	数据包状态(Packet Status) 表示接收到的数据包的状态。 0010: 接收到 IN 数据包 0011: 完成 IN 传输(触发中断) 0101: 数据翻转出错(触发中断) 0111: 通道中止(触发中断) 其它: 保留 复位值: 0
位 16: 15	DPID	0x0	ro	数据 PID(Data PID) 表示接收到的数据包的数据 PID 00: DATA0 10: DATA1 01: DATA2 11: MDATA 复位值: 0
位 14: 4	BCNT	0x000	ro	字节数(Byte Count) 表示已接收到的 IN 数据包的字节数
位 3: 0	CHNUM	0x0	ro	通道数(Channel Number) 表示当前接收到的数据包属于哪一个通道

设备模式：

域	简称	复位值	类型	功能
位 31: 25	保留	0x00	resd	保持默认值。
位 24: 21	FN	0x0	ro	帧号(Frame Number) 表示 USB 总线上所收到的数据包的帧号的最低 4 位。 此位域仅适用于支持同步 OUT 端点功能的情况。
位 20: 17	PKTSTS	0x0	ro	数据包状态(Packet Status) 表示接收到的数据包的状态。 0001: 全局 OUT NAK(触发中断) 0010: 收到 OUT 数据包 0011: 完成 OUT 传输(触发中断) 0100: 完成 SETUP 任务(触发中断) 0110: 收到 SETUP 数据包 其它: 保留
位 16: 15	DPID	0x0	ro	数据 PID(Data PID) 表示接收到的 OUT 数据包的数据 PID 00: DATA0 10: DATA1 01: DATA2 11: MDATA
位 14: 4	BCNT	0x000	ro	字节数(Byte Count) 表示接收到的数据包的字节数
位 3: 0	EPTNUM	0x0	ro	端点号(Endpoint Number) 表示当前所收到的数据包属于哪一个端点号

22.6.3.9 OTGFS接收FIFO长度寄存器(OTGFS_GRXFSIZ)

应用程序可以对必须要分配给接收 FIFO 的 SRAM 长度进行设置。

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	RXFDEP	0x0200	ro/rw	接收 FIFO 深度(RxFifo Depth) 此值以 32 位字为单位。 最小值为 16 最大值为 512 在配置期间，本寄存器的上电复位值被定义为接收数据 FIFO 的最大深度。

22.6.3.10 OTGFS非周期性TX FIFO长度寄存器(OTGFS_GNPTXFSIZ)/端点0 TX FIFO长度寄存器(OTGFS_DIEPTXF0)

应用程序可设置非周期性发送 FIFO 的 SRAM 长度和存储器起始地址，该寄存器的位域随主机模式或设备模式而变化。

主机：

域	简称	复位值	类型	功能
位 31: 16	NPTXFDEP	0x0000	ro/rw	非周期性发送 FIFO 深度(Non-periodic TxFIFO depth) 这个数值的单位是 32 位的字 最小值是 16 最大值是 256。
位 15: 0	NPTXFSTADDR	0x0200	ro/rw	非周期性发送 SRAM 起始地址(Non-periodic Transmit SRAM Start Address) 此位域包含了非周期性发送 FIFO SRAM (Non-periodic Transmit FIFO SRAM) 的存储器起始地址。

设备：

域	简称	复位值	类型	功能
位 31: 16	INEPT0TXDEP	0x0000	ro/rw	IN 端点发送 FIFO 0 深度(IN Endpoint TxFIFO 0 Depth) 这个数值的单位是 32 位的字 最小值是 16 最大值是 256。
位 15: 0	INEPT0TXSTADDR	0x0200	ro/rw	IN 端点 FIFO 0 发送 SRAM 起始地址(IN Endpoint FIFO0 Transmit SRAM Start Address) 此位域包含了 IN 端点发送 FIFO 0 的存储器起始地址

22.6.3.11 OTGFS非周期性TX FIFO/请求队列状态寄存器(OTGFS_GNPTXSTS)

本寄存器仅适用于主机模式，本寄存器为只读，储存了非周期性发送(FIFO Non-periodic TxFIFO) 和非周期性发送请求队列(Non-periodic Transmit Request Queue)的可用空间信息。

域	简称	复位值	类型	功能
位 31	保留	0x0	resd	保持默认值。
位 30: 24	NPTXQTOP	0x00	ro	非周期性发送请求队列的顶部(Top of the Non-periodic Transmit Request Queue) 表示 MAC 正在处理非周期性发送请求队列的条目 位[30: 27]: 通道/端点号 位[26: 25]: 00: IN/OUT 令牌 01: 零长度的发送包 (设备 IN/主机 OUT) 10: PING/CSPLIT 令牌 11: 通道中止指令 位[24]: 结束 (所选通道/端点的最后一个请求)
位 23: 16	NPTXQSPCAVAIL	0x08	ro	非周期性发送请求队列的可用空间(Non-periodic Transmit Request Queue Space Available) 表示非周期性发送请求队列的可用空间。在主机模式下，此队列既支持 IN 请求也支持 OUT 请求。 00: 非周期性发送请求队列的空间已满

				01: 1 个位置可用 02: 2 个位置可用 n: n 个位置可用($0 \leq n \leq 8$) 其它: 保留 复位值: 可设置
位 15: 0	NPTXFSPCAVAIL	0x0200	ro	非周期性发送 FIFO 的可用空间(Non-periodic TxFIFO Space Avail) 表示非周期性发送 FIFO 的可用空间, 此值以 32 位字节为单位。 00: 非周期性发送 FIFO 已满 01: 1 个字可用 02: 2 个字可用 n: n 个字可用 ($0 \leq n \leq 256$) 其它: 保留 复位值: 可设置

22.6.3.12 OTGFS通用控制器配置寄存器(OTGFS_GCCFG)

域	简称	复位值	类型	功能
位 31: 22	保留	0x00	resd	保持默认值。
位 21	VBUSIG	0x0	rw	VBUS 忽略 (VBUS ignored) 当该控制位被置位, OTGFS 并不监测 VBUS 引脚电压, 并且认为在主机和设备模式下, VBUS 电压一直有效, 然后可释放 VBUS 引脚作为其他用途。 0: VBUS 不被忽略 1: VBUS 被忽略, 并认为 VBUS 电压一直有效
位 20	SOFOUTEN	0x0	rw	SOF 输出使能(SOF output enable) 0: 不输出 SOF 脉冲; 1: 输出 SOF 脉冲到引脚上。
位 19: 18	保留	0x0	resd	保持默认值。
位 17	LP_MODE	0x0	rw	低功耗模式(Low-power mode) 作用: 控制 OTG PHY 的功耗。软件置 1 时, OTG PHY 进入低功耗模式; 软件置 0 时, 为正常模式。 0: 非低功耗模式; 1: 低功耗模式。
位 16	PWRDOWN	0x0	rw	掉电(Power down) 用于在发送和接收时激活收发器, 必需预先配置才能允许 USB 通信 0: 使能掉电; 1: 禁止掉电(收发器激活)。
位 15: 0	保留	0x0000	resd	保持默认值。

22.6.3.13 OTGFS控制器ID寄存器(OTGFS_GUID)

此寄存器只读, 保护产品的 ID。

域	简称	复位值	类型	功能
31: 0	USERID	0x0000 1000	rw	产品 ID (Product ID field) 应用程序可以编程此 ID 位。

22.6.3.14 OTGFS主机周期性发送FIFO长度寄存器(OTGFS_HPTXFSIZ)

此寄存器保存着周期性发送 FIFO 的深度和存储器起始地址

域	简称	复位值	类型	功能
位 31: 16	PTXFSIZE	0x02000	ro/rw	主机周期性发送 FIFO 深度 (Host periodic TxFIFO depth) 此位域的值以 32 位字为单位。 最小值是 16 最大值是 512
位 15: 0	PTXFSTADDR	0x0600	ro/rw	主机周期性发送 FIFO 起始地址(Host Periodic TxFIFO Start Address) 该寄存器的上电复位值指的是接收数据 FIFO 最大深度与非周期性发送数据 FIFO 最大深度之和。

22.6.3.15 OTGFS设备IN端点发送FIFO长度寄存器

(OTGFS_DIEPTXFn)(其中n是FIFO的编号, x=1…15)

本寄存器保存着在设备模式下进行的 IN 端点发送 FIFO 的深度以及存储器起始地址。每个 FIFO 包含一个 IN 端点数据。本寄存器可重复用于实例化的 IN 端点 FIFO1~15。通过 GNPTXFSIZ 寄存器来设置 IN 端点 FIFO 0 的深度及内存起始地址。

域	简称	复位值	类型	功能
位 31: 16	INEPTXFDEP	0x0200	ro/rw	IN 端点发送 FIFO 深度(IN Endpoint TxFIFO Depth) 以 32 位字为单位。 最小值为 16 最大值为 512 复位值是最大可能的 IN 端点发送 FIFO 的深度。
位 15: 0	INEPTXFSTADDR	0x0400	ro/rw	IN 端点 FIFO 0 发送 SRAM 起始地址(IN Endpoint FIFO Transmit SRAM Start Address) 此值为 IN 端点 n 发送 FIFO 在 SRAM 中的起始地址。

22.6.4 主机模式下的寄存器

该寄存器影响主机模式下控制器的运行状况。不支持在设备模式下进行访问（因为设备模式下访问结果未定义）。主机模式下的寄存器分别为：

22.6.4.1 OTGFS主机模式配置寄存器(OTGFS_HCFG)

该寄存器用于上电后配置控制器。初始化之后，不得再更改本寄存器。

域	简称	复位值	类型	功能
位 31: 3	保留	0x0000 0000	resd	保持默认值。
位 2	FSLSSUPP	0x0	ro	仅支持全速和低速设备(FS- and LS-Only Support) 应用程序通过此位来控制内核的枚举速度。应用程序可通过此位将控制器以全速主机模式来枚举，即使已连接的设备支持高速通信。在初始化设置之后，不得再更改此位。 0: 全速/低速，具体情况取决于连接设备所支持的最大速度 1: 仅支持全速/低速，即使所连接的设备支持高速模式。
位 1: 0	FSLSPCLKSEL	0x0	rw	全速/低速 PHY 时钟选择(FS/LS PHY Clock Select) 当控制器处于全速主机模式下： 01: PHY 时钟运行频率为 48MHz 其它值：保留 当控制器处于低速主机模式下： 00: 保留； 01: PHY 时钟运行频率为 48MHz。 10: PHY 时钟运行频率为 6MHz。如果选择了 6MHz 时钟，则必须进行软件复位。 11: 保留

22.6.4.2 OTGFS主机帧间隔寄存器(OTGFS_HFIR)

该寄存器用于设置当前枚举速度的帧间隔。

域	简称	复位值	类型	功能
位 31: 17	保留	0x0000	resd	保持默认值。
位 16	HFIERRLDCTRL	0x0	rw	<p>重新加载控制(Reload Control) 该位允许在运行时对主机帧间隔寄存器进行动态重新加载。</p> <p>1: 不能动态重新加载主机帧间隔寄存器 0: 可以在运行时动态重新加载主机帧间隔寄存器 该位需在初始化时设置好，并且在运行期间不得更改其值。</p>
位 15: 0	FRINT	0xEA60	rw	<p>帧间隔(Frame Interval) 应用程序通过此位域来设置是两个连续 SOF (全速) 之间的间隔。 此位域中的 PHY 时钟个数即表示帧间隔。只有当主机端口控制及状态寄存器的端口使能位被设置后，应用程序才可以写入主机帧间隔寄存器。 如果未设定此位域，控制器将根据主机配置寄存器的 FS/LS PHY 时钟选择位所定义的 PHY 时钟频率来计算其值。初始化配置之后，不得再更改其值。</p> <p>1 ms * (全速/低速 PHY 时钟频率)</p>

22.6.4.3 OTGFS主机帧号/帧时间剩余寄存器(OTGFS_HFNUM)

该寄存器指示当前帧号，以及当前帧剩余时间（以 PHY 时钟个数表示）。

域	简称	复位值	类型	功能
位 31: 16	FTREM	0x0000	ro	<p>帧剩余时间(Frame Time Remaining) 指示当前帧（全速/低速）还剩余多少时间，以 PHY 时钟个数表示。此位域的值会随每个 PHY 时钟个数而自动递减。当值减为 0 时，帧间隔寄存器的值会重新加载到此位域，并且向 USB 总线发出一个新 SOF。</p>
位 15: 0	FRNUM	0x3FFF	ro	<p>帧号(Frame Number) 每当向 USB 总线发出一个新 SOF 时，此位域的值就会递增一次；当值增加到 16'h3FFF 时，此位域即复位为 0。</p>

22.6.4.4 OTGFS主机周期性发送FIFO/请求队列寄存器(OTGFS_HPTXSTS)

该寄存器为只读，包含周期性发送 FIFO 和周期性发送请求队列的剩余空间信息。

域	简称	复位值	类型	功能
位 31: 24	PTXQTOP	0x00	ro	<p>周期性发送请求队列的顶层(Top of the Periodic Transmit Request Queue) 表示 MAC 正在处理的周期性发送请求队列的条目。该寄存器用于模块调试。 位[31]: 奇数/偶数帧 0: 偶数帧发送 1: 奇数帧发送 位[30: 27]: 通道/端点号 位[26: 25]: 类型 00: IN/OUT 01: 零长度包 10: 保留 11: 禁止通道指令 位[24]: 终止（所选通道或端点的最后一个条目）</p>
位 23: 16	PTXQSPCAVAIL	0x08	ro	<p>周期性发送请求队列剩余空间(Periodic Transmit Request Queue Space Available) 指示周期性发送请求队列里允许写入的可用空间。此队列包括 IN 和 OUT 请求。 00: 周期性发送请求队列满额</p>

				01: 1 个可用空间 10: 2 个可用空间 n: n 个可用空间($0 \leq n \leq 8$) 其它值: 保留
位 15: 0	PTXFSPCAVAIL	0x0100	rw	周期性发送数据 FIFO 剩余空间(Periodic Transmit Data FIFO Space Available) 指示周期性发送 FIFO 的可用空间, 以 32 位字为单位 0000: 周期性发送 FIFO 满额 0001: 1 个字可用 0010: 2 个字可用 n: n 个字可用(其中 $0 \leq n \leq 512$) 其它值: 保留

22.6.4.5 OTGFS 主机所有通道中断寄存器(OTGFS_HINT)

当某个通道产生了标志事件时, 主机所有通道中断寄存器将通过控制器中断寄存器的主机通道中断位来中断应用程序, 如图 22-2 所示。每个通道都有一个中断位, 共 16 位。应用程序通过设置或清除相应主机通道 n 中断寄存器的相应位来设置或清除此位。

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	HINT	0x0000	ro	通道中断(Channel Interrupts) 每个通道对应一个位: 通道 0 对应位 0, 通道 15 对应位 15。

22.6.4.6 OTGFS 主机所有通道中断屏蔽寄存器(OTGFS_HINTMSK)

主机所有通道中断屏蔽寄存器与主机所有通道中断寄存器配置使用, 用于控制在产生事件时是否打断应用程序。每个通道都有一个相对应的中断屏蔽位, 共有 16 位。

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 0	HINTMSK	0x0000	rw	通道中断屏蔽(Channel Interrupt Mask) 每个通道对应一个位: 位 0 对应通道 0, 位 15 对应通道 15。

22.6.4.7 OTGFS 主机端口控制和状态寄存器(OTGFS_HPRT)

该寄存器仅适用于主机模式。OTG 主机目前仅支持一个端口。

该寄存器包含 USB 端口相关的信息, 比如每个端口的 USB 复位, 使能, 挂起, 唤醒, 连接状态以及测试模式等信息, 如图 22-2 所示。类型为 rw1c 的寄存器可以通过控制器中断寄存器的主机端口中断位来中断应用程序。端口发生中断时, 应用程序必须读取该寄存器, 并且清除导致该中断的位。类型为 rw1c 的寄存器, 应用程序需要写 1 来清除中断。

域	简称	复位值	类型	功能
位 31: 19	保留	0x0000	resd	保持默认值。
位 18: 17	PRTSPD	0x0	ro	端口速度(Port Speed) 表示连上端口的设备的速度。 00: 保留 01: 全速 10: 低速 11: 保留
位 16: 13	PRTTSTCTL	0x0	rw	端口测试控制(Port Test Control) 应用程序通过向此位域写入一个非零值而将此端口置于测试模式, 且端口会发出相应信号。 0000: 测试模式关闭 0001: Test_J 模式 0010: Test_K 模式 0011: Test_SE0_NAK 模式 0100: Test_Packet 模式 0101: Test_Force_Enable 其它值: 保留
位 12	PRTPPWR	0x0	rw	端口供电(Port Power)

				应用程序通过此位来控制对该端口的供电（写 1 或写 0）。 0: 断电 1: 供电 注意：该位与接口无关。应用程序需按照编程手册中的主机编程流程为各个接口设置此位。
位 11: 10 PRTLNSTS	0x0	ro		端口线状态(Port Line Status) 表示当前 USB 数据线的逻辑状态。 位[10]: D+的逻辑电平 位[11]: D- 的逻辑电平
位 9 保留	0x0	resd		保持默认值。
位 8 PRTRST	0x0	rw		端口复位(Port Reset) 当应用程序置起此位后，端口会启动复位序列。应用程序必须计算此次复位序列所需时间，并在复位结束后清除此位。 0: 端口不复位 1: 端口复位 应用程序必须使该位保持置起状态至少达到 USB2.0 规范第 7.1.7.5 章节规定的最短持续时间才能启动端口复位。除了最短持续时间外，应用程序在清除此位之前还可以额外增加 10ms，USB 规范没有设定最长持续时间。
位 7 PRTSUSP	0x0	rw1s		端口挂起(Port Suspend) 通过将该位置起，应用程序可使端口进入挂起模式，此时，控制器只停止发送 SOF。应用程序必须通过设置端口时钟停止位才能关闭 PHY 时钟。 读取该位将返回当前端口的挂起状态。 当控制器器检测到远程唤醒信号时会清除此位，或者当应用程序将该寄存器的端口复位位或端口唤醒位置起，或者将控制器中断寄存器的唤醒/远程唤醒检测中断或断开连接检测中断位置起后，也会清除此位。 即便设备未与主机相连，控制器仍然可以清除此位。 0: 端口未处于挂起模式 1: 端口处于挂起模式
位 6 PRTRES	0x0	rw		端口唤醒(Port Resume) 应用程序通过设置此位来驱动端口发出唤醒信号。控制器会持续触发唤醒信号直到应用程序清除此位。如果控制器检测到 USB 远程唤醒序列（按照控制器中断寄存器的端口唤醒/远程唤醒检测中断位的指示），控制器将不受应用程序干预自动触发唤醒信号。 读取此位时将返回控制器是否正在驱动唤醒信号的信息。 0: 无触发唤醒 1: 触发唤醒
位 5 PRTOVRCCHNG	0x0	rw1c		端口过流状态改变(Port Overcurrent Change) 当该寄存器的端口过电流有效位(位 4)的状态发生改变时，控制器将置起此位。只能通过控制器来设置此位，应用程序必须写 1 才能清除此位。
位 4 PRTOVRCACT	0x0	ro		端口过流有效位(Port Overcurrent Active) 表示端口的过流状况 0: 不存在过电流 1: 存有过电流
位 3 PRTEENCHNG	0x0	rw1c		端口使能/禁止状态改变(Port Enable/Disable Change) 当该寄存器的端口使能位 2 的状态发生改变时，控制器将此位置起。只能通过控制器将该位置起，应用程序必须写 1 才能清除此位。
位 2 PRTEENA	0x0	rw1c		端口使能(Port Enable) 此端口只有在复位序列后才能被控制器使能。在发生过电流，断开连接或者应用程序将此位清除时，即禁用此端口。应用程序不能通过写入寄存器来设置此位，只能通过清除此位来禁用端口。该位不会触发中断。 0: 端口禁用 1: 端口使能

				端口连接检测(Port Connect Detected)
位 1	PRTCONDET	0x0	rw1c	控制器在检测到设备连接端口时，会通过控制器中断寄存器的主机端口中断位来设置该位。该位只能由控制器设置，应用程序必须写 1 才能清除此位。
位 0	PRTCONSTS	0x0	ro	端口连接状态(Port Connect Status) 0: 没有设备连接端口 1: 设备连接端口

22.6.4.8 OTGFS主机通道x特性寄存器(OTGFS_HCCHARx)(此处x代码通道号, x = 0...15)

域	简称	复位值	类型	功能
位 31	CHENA	0x0	rw1s	通道使能(Channel Enable) 此位由应用程序设置，并由 OTG 主机清除。 0: 通道禁止 1: 通道使能
位 30	CHDIS	0x0	rw1s	通道禁止(Channel Disable) 应用程序设置此位来停止发送/接收通道上的数据，即便通道传输还没完成。应用程序必须等到通道禁止中断产生，才视该通道已禁止。
位 29	ODDFRM	0x0	rw	奇数帧(Odd Frame) 应用程序通过设置/复位此位来指示 OTG 主机是否以奇数帧来传输。此位仅适用于周期性（同步和中断）传输。 0: 偶数帧 1: 奇数帧
位 28: 22	DEVADDR	0x00	rw	设备地址(Device Address) 此位域用于选择可作为数据源或接收器的指定设备。
位 21: 20	MC	0x0	rw	多次计数 Multi Count (MC) 此位域通知主机该周期性端口的每个帧要执行的传输数目。 00: 保留，此位域生成未定义的结果 01: 1 个事务 10: 每个帧有 2 个事务 11: 每个帧有 3 个事务 此位域至少应该设置为 0x01。
位 19: 18	EPTYPE	0x0	rw	端点类型(Endpoint Type) 表示所选择的传输类型。 00: 控制传输 01: 同步传输 10: 批传输 11: 中断传输
位 17	LSPDDEV	0x0	rw	低速设备(Low-Speed Device) 应用程序通过设置此位来指示该通道正在与低速设备通信。
位 16	保留	0x0	resd	保持默认值。
位 15	EPTDIR	0x0	rw	端点方向(Endpoint Direction) 指示传输是 IN 还是 OUT 0: OUT 1: IN
位 14: 11	EPTNUM	0x0	rw	端点号(Endpoint Number) 指示设备（用作数据源或接收器）的端点号。
位 10: 0	MPS	0x000	rw	最大包长度(Maximum Packet Size) 指示相应端点的最大包长度。

22.6.4.9 OTGFS主机通道x中断寄存器(OTGFS_HCINTx)(其中x代表通道号, x=0...15)

该寄存器包含着与 USB 和 AHB 相关事件的通道状态, 如图 22-2 所示。当控制器中断寄存器的主机通道中断位置起时, 应用程序必须读取该寄存器。在读寄存器之前, 应用程序必须先读取主机所有通道中断寄存器以获理主机通道 n 中断寄存器的确切通道编号。应用程序必须通过清除该寄存器的相应位来清除 OTGFS_HAIT 和 OTGFS_GINTSTS 寄存器的相应位。

域	简称	复位值	类型	功能
位 31: 11	保留	0x000000	resd	保持默认值。
位 10	DTGLERR	0x0	rw1c	数据翻转错误(Data Toggle Error) 只能由控制器设置此位, 应用程序通过写 1 来清除此位。
位 9	FRMOVRUN	0x0	rw1c	帧溢出(Frame Overrun) 只能由控制器设置此位, 应用程序通过写 1 来清除此位。
位 8	BBLERR	0x0	rw1c	Babble Error 只能由控制器设置此位, 应用程序通过写 1 来清除此位。
传输错误(Transaction Error) 表示 USB 总线发生了下列错误: CRC 校验失败				
位 7	XACTERR	0x0	rw1c	传输超时 位填充错误 EOP 错误 只能由控制器设置此位, 应用程序通过写 1 来清除此位。
位 6	保留	0x0	resd	保持默认值。
位 5	ACK	0x0	rw1c	ACK 响应已收到/已发送中断(ACK Response Received/Transmitted Interrupt) 只能由控制器设置此位, 应用程序通过写 1 来清除此位。
位 4	NAK	0x0	rw1c	NAK 响应已收到中断(NAK Response Received Interrupt) 只能由控制器设置此位, 应用程序通过写 1 来清除此位。
位 3	STALL	0x0	rw1c	STALL 响应已收到中断(STALL Response Received Interrupt) 只能由控制器设置此位, 应用程序通过写 1 来清除此位。
位 2	保留	0x0	resd	保持默认值。
位 1	CHHLTD	0x0	rw1c	通道中止(Channel Halted) 此位指示传输异常结束, 原因是 USB 传输出错或者为响应应用程序因传输完成而发出的中止请求。
位 0	XFERC	0x0	rw1c	传输完成(Transfer Completed) 传输正常结束, 未报错。此位只能由控制器设置, 应用程序通过写 1 来清除此位。

22.6.4.10 OTGFS主机通道x中断屏蔽寄存器(OTGFS_HCINTMSKx)(其中x为通道号, x=0...15)

本寄存器用于屏蔽上一节所描述的各类通道中断。

域	简称	复位值	类型	功能
位 31: 11	保留	0x000000	resd	保持默认值。
位 10	DTGLERRMSK	0x0	rw	数据翻转错误屏蔽(Data Toggle Error Mask)
位 9	FRMOVRUNMSK	0x0	rw	帧溢出屏蔽(Frame Overrun Mask)
位 8	BBLERRMSK	0x0	rw	Babble 错误屏蔽(Babble Error Mask)
位 7	XACTERRMSK	0x0	rw	传输错误屏蔽(Transaction Error Mask)
位 6	NYETMSK	0x0	rw	NYET 响应已收到中断屏蔽(NYET Response Received Interrupt Mask)
位 5	ACKMSK	0x0	rw	ACK 响应已收到/已发送中断屏蔽(ACK Response Received/Transmitted Interrupt Mask)
位 4	NAKMSK	0x0	rw	NAK 响应已收到中断屏蔽(NAK Response Received Interrupt Mask)
位 3	STALLMSK	0x0	rw	STALL 响应已收到中断屏蔽(STALL Response Received Interrupt Mask)
位 2	保留	0x0	resd	保持默认值。
位 1	CHHLTDMSK	0x0	rw	通道中止屏蔽(Channel Halted Mask)
位 0	XFERCMSK	0x0	rw	传输完成屏蔽(Transfer Completed Mask)

22.6.4.11 OTGFS主机通道x传输长度寄存器(OTGFS_HCTSIZx)(其中x为通道号, x=0...15)

域	简称	复位值	类型	功能
位 31	保留	0x0	resd	保持默认值。
位 30: 29 PID		0x0	rw	PID (Pid) 应用程序在此位域设置了用于首次传输的 PID 类型。主机控制着此位域用于后续的传输任务。 00: DATA0 01: DATA2 10: DATA1 11: MDATA(非控制)/SETUP(控制)
位 28: 19 PKTCNT		0x000	rw	包数目(Packet Count) 应用程序在此位域设置了期望发送或期望接收的包数目。主机在每次成功发送/接收 OUT/IN 包时会递减包数目。当数据减为 0 时，应用程序将收到中断以指示传输正常结束。
位 18: 0 XFERSIZE		0x00000	rw	传输长度(Transfer Size) 对于 OUT 传输来说，此位域指示主机在传输过程中发送的数据字节数。 对于 IN 传输来说，此位域指示应用程序为传输预留的缓冲区长度。 对于 IN 传输（周期性和非周期性），应用程序需要将此位域设置为最大包长度的整数倍。

22.6.5 设备模式下的寄存器

这些寄存器仅用于设备模式。主机模式不允许访问，因为访问结果未知。其中有些寄存器会影响到所有端点，而有些寄存器只影响某一个端点。

22.6.5.1 OTGFS设备配置寄存器(OTGFS_DCFG)

在复位后、特殊控制命令后或者枚举后，此寄存器用于配置设备模式下的控制器。在初始化之后，请不要再修改此寄存器。

域	简称	复位值	类型	功能
位 31: 13 保留		0x0110	resd	保持默认值。
位 12: 11 PERFRINT		0x0	rw	周期性帧间隔(Periodic frame interval) 此位域表示产生“周期性帧结束中断”位时占某个帧内的时间比。应用程序可以通过此中断来确认帧内的同步传输是否已完成。 00: 80%的帧时间； 01: 85%的帧时间； 10: 90%的帧时间； 11: 95%的帧时间。
位 10: 4 DEVADDR		0x0	rw	设备地址(Device address) 应用程序在每次收到设置地址 (SetAddress) 控制命令后必须设置此位域。
位 3 保留		0x0	resd	保持默认值。
位 2 NZSTSOUTSHSHK		0x0	rw	非零长度的状态 OUT 握手信号(Non-zero-length status OUT handshake) 在控制传输的状态阶段，如果收到了非零长度的数据包，控制器会发出握手信号，应用程序正是通过此位来选择控制器要发送的握手信号。 1: 向非零长度的状态 OUT 传输发送一个 STALL 握手信号，但不向应用程序传送所收到的 OUT 数据包 0: 向应用程序传递所收到的 OUT 数据包（要么零长度，要么非零长度），并且根据设备端点控制寄存器的 NAK 和 STALL 位选择发送握手信号。
位 1: 0 DEVSPD		0x0	rw	设备速度(Device speed) 此位域指示的是应用程序需要控制器进行枚举的速度，或者应用程序可支持的最大速度。然而，总线实际速度只能

在整个序列完成之后，才能根据控制器与 USB 主机相连的速度来决定。

00: 保留;

01: 保留;

10: 保留;

11: 全速(USB1.1 收发器，时钟为 48MHz)。

22.6.5.2 OTGFS设备控制寄存器(OTGFS_DCTL)

域	简称	复位值	类型	功能
位 31: 12	保留	0x00000	resd	保持默认值。
位 11	PWRPROGDNE	0x0	wo	上电配置结束(Power-on programming done) 应用程序通过此位来指示从断电模式唤醒后，完成了对该寄存器的配置。
位 10	CGOUTNAK	0x0	wo	清除全局 OUT NAK (Clear global OUT NAK) 对此位进行写操作可以清除全局 OUT NAK 状态。
位 9	SGOUTNAK	0x0	wo	设置全局 OUT NAK (Set global OUT NAK) 对此位进行写操作可以设置全局 OUT NAK 状态。 应用程序通过此位向所有 OUT 端点发送一个 NAK 握手信号。只有在确认已清除了控制器中断寄存器的全局 OUT NAK 有效位时，应用程序才需要设置此位。
位 8	CGNPINNAK	0x0	wo	清除全局非周期性 IN NAK (Clear Global Non-periodic IN NAK) 对此位写操作可以清除全局非周期性 IN NAK 状态。
位 7	SGNPINNAK	0x0	wo	设置全局非周期性 IN NAK (Set global Non-periodic IN NAK) 对此位写操作可以设置全局非周期性 IN NAK 状态。 应用程序通过此位向所有非周期性 IN 端点发送一个 NAK 握手信号。应用程序只有在确认了控制器中断寄存器的全局 IN NAK 有效信号已清除的情况下才设置此位。
位 6: 4	TSTCTL	0x0	rw	测试控制(Test control) 000: 测试模式禁止; 001: Test_J 模式; 010: Test_K 模式; 011: Test_SE0_NAK 模式; 100: Test_Packet 模式; 101: Test_Force_Enable; 其他: 保留。
位 3	GOUTNAKSTS	0x0	ro	全局 OUT NAK 状态 (Global OUT NAK status) 0: 根据 FIFO 状态，NAK 和 STALL 位的设定来发送握手信号 1: 不管是否有可用空间，都不会写数据到接收 FIFO。向所有数据包（除了 SETUP 传输）发送一个 NAK 握手信号。丢弃所有同步 OUT 数据包。
位 2	GNPINNAKSTS	0x0	ro	全局非周期性 IN NAK 状态 (Global Non-periodic IN NAK status) 0: 根据发送 FIFO 的数据状况来发送握手信号 1: 不管发送 FIFO 内的数据状况如何，都会向所有的非周期性 IN 端点发送 NAK 握手信号。
位 1	SFTDISCON	0x1	rw	软件断开(Soft disconnect) 应用程序通过此位指示 OTGFS 控制器进行“软件断开”操作。一旦置起此位，那么主机会看到设备已断开，设备也收不收到 USB 总线信号。控制器就始终处于断开状态直至应用程序清除此位。 0: 普通操作。当此位在设备软件断开后清除，控制器会向主机发起一个设备连接事件。USB 主机与设备重新连接后，会重启设备枚举。
位 0	RWKUPSIG	0x0	rw	远程唤醒信号(Remote wakeup signaling) 应用程序置起此位后，控制器会启动远程信号来唤醒 USB 主机。应用程序必须置起此位来指示控制器退出挂起状态。

按照 USB2.0 的规定，应用程序在设置此位 1 – 15 ms 后必须清除该位。

为了使 USB 主机检测到设备断开，软件断开位在各种状态下保持置起的最短持续时间在下表给出。为了适应时钟抖动，建议应用程序在规定的最短持续时间的基础上再添加一些额外的延迟。

表 22-5 软件断开的最短持续时间

操作速度	设备状态	最短持续时间
全速	挂起	1ms + 2.5us
全速	空闲	2.5us
全速	不空闲或挂起(正在执行传输操作)	2.5us

22.6.5.3 OTGFS设备状态寄存器(OTGFS_DSTS)

此寄存器指示控制器与 OTGFS 相关的状态。此寄存器用于在发生设备所有中断(OTGFS_DAINT)寄存器事件时读取。

域	简称	复位值	类型	功能
位 31: 22	保留	0x000	resd	保持默认值。
位 21: 8	SOFFN	0x0000	ro	接收到的 SOF 的帧号(Frame number of the received SOF) 注意：如果在上电复位后立即读取此位域，可能返回非零值。如果在上电复位后立即读取此位域时返回的是非零值，并不表示已经收到主机发出的 SOF。只有当主机和设备进行连接后，此位域的读取值才是有效的。
位 7: 4	保留	0x1	resd	保持默认值。
位 3	ETICERR	0x0	ro	随机误差(Erratic error) 此类错误会导致控制器挂起，并且控制器中断寄存器的早期挂起位会置起，随后产生中断。如果是因为异常错误而触发控制器早期挂起，那么应用程序只能执行软件断开进行修复解决。
位 2: 1	ENUMSPD	0x0	ro	枚举速度(Enumerated speed) 表示控制器在通过序列进行速度检测后所确定的执行速度。 01: 保留； 10: 保留； 11: 全速(PHY 时钟运行在 48MHz)； 其他：保留。
位 0	SUSPSTS	0x0	ro	挂起状态(Suspend status) 在设备模式下，只要检测到 USB 总线上有挂起条件，此位就会置起。当 USB 总线信号长时间不活动时，控制器即会进入挂起状态。 控制器在遇到下列情况时会退出挂起状态： ■USB 总线信号开始活动 ■应用程序对设备控制寄存器的远程唤醒信号位进行写操作

22.6.5.4 OTGFS设备OTGFSIN端点通用中断屏蔽寄存器 (OTGFS_DIEPMSK)

本寄存器与各个端点的设备 IN 端点中断寄存器配合工作以产生 IN 端点中断。可以通过向设备 IN 端点通用中断屏蔽寄存器的相应位进行写操作来屏蔽设备 IN 端点中断寄存器(OTGFS_DIEPINTx)中某个特定状态的 IN 端点中断。状态位是默认屏蔽的。

域	简称	复位值	类型	功能
位 31: 14	保留	0x000000	resd	保持默认值。
位 13	NAKMSK	0x0	rw	NAK 中断屏蔽(NAK interrupt mask) 0: 中断屏蔽; 1: 中断不屏蔽。
位 12: 10	保留	0x0	resd	保持默认值
位 9	BNAINMSK	0x0	rw	BNA 中断屏蔽(BNA interrupt mask) 0: 中断屏蔽; 1: 中断不屏蔽。
位 8	TXFIFOUDRMSK	0x0	rw	FIFO 欠载中断屏蔽(FIFO underrun mask) 0: 中断屏蔽; 1: 中断不屏蔽。
位 7	保留	0x0	resd	保持默认值。
位 6	INEPTNAKMSK	0x0	rw	IN 端点 NAK 状态有效中断屏蔽(IN endpoint NAK effective mask) 0: 中断屏蔽; 1: 中断不屏蔽。
位 5	INTKNEPTMISMSK	0x0	rw	端点收到 IN 命令不匹配中断屏蔽(IN token received with EP mismatch mask) 0: 中断屏蔽; 1: 中断不屏蔽。
位 4	INTKNTXFEMPMSK	0x0	rw	当发送 FIFO 空时收到 IN 命令中断屏蔽(IN token received when TxFIFO empty mask) 0: 中断屏蔽; 1: 中断不屏蔽。
位 3	TIMEOUTMSK	0x0	rw	超时条件中断屏蔽(非同步端点) (Timeout condition mask (Non-isochronous endpoints)) 0: 中断屏蔽; 1: 中断不屏蔽。
位 2	保留	0x0	resd	保持默认值。
位 1	EPTDISMSK	0x0	rw	端点禁用中断屏蔽 (Endpoint disabled interrupt mask) 0: 中断屏蔽; 1: 中断不屏蔽。
位 0	XFERCMSK	0x0	rw	传输完成中断屏蔽(Transfer completed interrupt mask) 0: 中断屏蔽; 1: 中断不屏蔽。

22.6.5.5 OTGFS设备OUT端点通用中断屏蔽寄存器 (OTGFS_DOEPMSK)

此寄存器配合设备 OUT 端点中断寄存器(OTGFS_DOEPINTx)使用，产生 OUT 端点中断。

OTGFS_DOEPINTx 寄存器的每一位都可以通过写此寄存器的相应位来屏蔽。在默认状态下，所有中断都屏蔽。

域	简称	复位值	类型	功能
位 31: 15	保留	0x000000	resd	保持默认值。
位 14	NYETMSK	0x0	rw	NYET 中断屏蔽 (NYET interrupt mask) 0: 中断屏蔽； 1: 中断不屏蔽。
位 13	NAKMSK	0x0	rw	NAK 中断屏蔽 (NAK interrupt mask) 0: 中断屏蔽； 1: 中断不屏蔽。
位 12	BERRMSK	0x0	rw	Babble 错误中断屏蔽 (Babble error interrupt mask) 0: 中断屏蔽； 1: 中断不屏蔽。
位 11:10	保留	0x0	rw	保持默认值。
位 9	BNAOUTMSK	0x0	rw	BNA 中断屏蔽 (BNA interrupt mask) 0: 中断屏蔽； 1: 中断不屏蔽。
位 8	OUTPERRMSK	0x0	rw	OUT 包错误中断屏蔽 (OUT packet error mask) 0: 中断屏蔽； 1: 中断不屏蔽。
位 7	保留	0x0	resd	保持默认值。
位 6	B2BSETUPMSK	0x0	rw	收到连续的 SETUP 包中断屏蔽 (Back-to-back SETUP packets received mask) 0: 中断屏蔽； 1: 中断不屏蔽。
位 5	保留	0x0	resd	保持默认值。
位 4	OUTTEPDMSK	0x0	rw	当端点被禁止时收到 OUT 命令中断屏蔽 (OUT token received when endpoint disabled mask) 0: 中断屏蔽； 1: 中断不屏蔽。
位 3	SETUPMSK	0x0	rw	SETUP 阶段完成中断屏蔽 (SETUP phase done mask) 仅对控制端点有效 0: 中断屏蔽； 1: 中断不屏蔽。
位 2	保留	0x0	resd	保持默认值。
位 1	EPTDISMSK	0x0	rw	端点被禁止中断屏蔽 (Endpoint disabled interrupt mask) 0: 中断屏蔽； 1: 中断不屏蔽。
位 0	XFERCMSK	0x0	rw	传输结束中断屏蔽 (Transfer completed interrupt mask) 0: 中断屏蔽； 1: 中断不屏蔽。

22.6.5.6 OTGFS设备所有端点中断寄存器(OTGFS_DAINT)

当某个端点发生事件时，设备所有端点中断寄存器可以通过设置控制器中断寄存器的设备 IN 端点中断位或设备 OUT 端点中断位来中断应用程序。每个端点有一个中断位，其中 OUT 端点和 IN 端点各自拥有多达 8 个中断位。对于双向端点而言，同时使用对应的 IN 中断位和 OUT 中断位。应用程序通过设置和清除对应的设备端点 x 中断寄存器来设置和清除设备所有端点中断寄存器的相应位。

域	简称	复位值	类型	功能
位 31: 24	保留	0x0000	resd	保持默认值。
位 23: 16	OUTEPTINT	0x0000	ro	OUT 端点中断位 (OUT endpoint interrupt bits) 每个位对应一个 OUT 端点。位 16 对应 OUT 端点 0，位 18 对应 OUT 端点 2。
位 15: 8	保留	0x0000	resd	保持默认值。
位 7: 0	INEPTINT	0x0000	ro	IN 端点中断位 (IN endpoint interrupt bits) 每个位对应一个 IN 端点。位 0 对应 IN 端点 0，位 7 对应 IN 端点 7。

22.6.5.7 OTGFS所有端点中断屏蔽寄存器(OTGFS_DAINTMSK)

当某个设备端点发生事件时，设备端点中断屏蔽寄存器与设备端点中断寄存器共同配合来中断应用程序。尽管如此，与该中断相对应的设备所有端点中断寄存器仍是置起状态。

域	简称	复位值	类型	功能
位 31: 24	保留	0x0000	resd	保持默认值。
位 23: 16	OUTEPTMSK	0x0000	rw	OUT 端点中断屏蔽寄存器 (OUT EP interrupt mask bits) 每个位对应一个 OUT 端点。 位 16 对应 OUT 端点 0，位 18 对应 OUT 端点 2。 0: 屏蔽中断； 1: 不屏蔽中断。
位 15: 8	保留	0x0000	resd	保持默认值。
位 7: 0	INEPTMSK	0x0000	rw	IN 端点中断屏蔽位 (IN EP interrupt mask bits) 每个位对应一个 IN 端点。 位 0 对应 IN 端点 0，位 7 对应 IN 端点 7。 0: 屏蔽中断； 1: 不屏蔽中断。

22.6.5.8 OTGFS设备IN端点FIFO空中断屏蔽寄存器(OTGFS_DIEPEMPMSK)

此寄存器配合 IN 端点 FIFO 空中断寄存器(TXFE_OTGFS_DIEPINTx)产生中断。

域	简称	复位值	类型	功能
位 31: 8	保留	0x0000	resd	保持默认值。
位 7: 0	INEPTXFEMSK	0x0000	rw	IN 端点发送 FIFO 空中断屏蔽位 (IN EP Tx FIFO empty interrupt mask bits) 此位域是设备 IN 端点中断寄存器的屏蔽位。 一个发送 FIFO 空中断位对应一个端点：位 0 对应 IN 端点 0，位 7 对应 IN 端点 7。 0: 屏蔽中断； 1: 不屏蔽中断。

22.6.5.9 OTGFS设备控制IN端点0控制寄存器(OTGFS_DIEPCTL0)

本节介绍了控制 IN 端点 0 控制寄存器。非零控制端点使用端点 1-7 寄存器。

域	简称	复位值	类型	功能
位 31	EPTENA	0x0	rw1s	端点使能 (Endpoint enable) 此应用程序设置此位，以启动在端点 0 上的数据传输。 控制器在产生以下端点中断前会先清除此位： ■端点禁止； ■传输完成。
位 30	EPTDIS	0x0	ro	端点禁用 (Endpoint disable) 应用程序通过设置此位可以在完成传输前停止端点上的数据传输。应用程序必须要等到端点禁用中断产生才认为端点已禁用。控制器在生成端点禁用中断之前先清除此位。 只有端点使能的情况下，应用程序才需要置起此位。
位 29: 28	保留	0x0	resd	保持默认值。
位 27	SNAK	0x0	wo	设置 NAK (Set NAK) 通过对该位写操作可以设置端点的 NAK 位。应用程序可通过此位来控制端点上的 NAK 握手信号的发送。当该端点接收到 SETUP 数据包时，控制器也会置起此位。
位 26	CNAK	0x0	wo	清除 NAK (Clear NAK) 对该位写操作可清除端点的 NAK 位。
位 25: 22	TXFNUM	0x0	rw	发送 FIFO 的编号 (TxFIFO number) 控制端点 0 只能使用 FIFO0
位 21	STALL	0x0	rw1s	STALL 握手 (STALL handshake) 应用程序置起此位，控制器会在收到 SETUP 令牌时清除此位。如果该位连同 NAK 位，全局非周期性 IN NAK 位或者全局 OUT NAK 位同时置起时，STALL 位享有优先级。
位 20	保留	0x0	resd	保持默认值。
位 19: 18	EPTYPE	0x0	ro	端点类型 (Endpoint type) 对于控制端点，由硬件置为 00。
位 17	NAKSTS	0x0	ro	NAK 状态 (NAK status) 表示为： 0：表示控制器根据 FIFO 状态发送非 NAK 握手信号 1：表示控制器正在发送 NAK 握手信号。 当此位置起时（无论是应用程序设置的还是控制器设置的），控制器都会停止发送数据，即使发送 FIFO 有可用空间。不管该位是否置起，控制器始终以 ACK 握手信号来响应 SETUP 数据包。
位 16	保留	0x0	resd	保持默认值。
位 15	USBACCEPT	0x0	ro	USB 有效端点(USB active endpoint) 此位始终置 1，表示不管在什么配置和接口中，控制端点 0 始终有效。
位 14: 2	保留	0x0000	resd	保持默认值。
位 1: 0	MPS	0x0	rw	适用 IN 端点和 OUT 端点。 应用程序通过此位设置当前逻辑端点的最大数据包长度。 00: 64 字节 01: 32 字节 10: 16 字节 11: 8 字节

22.6.5.10 OTGFS设备IN端点x控制寄存器(OTGFS_DIEPCTLx)(其中x为端点号, x=1…7)

应用程序通过这些寄存器控制非0端点的操作特性。

域	简称	复位值	类型	功能
位 31	EPTENA	0x0	rw1s	端点使能 (Endpoint enable) 应用程序设置此位，表示该端点准备发送数据 控制器在产生以下端点中断之前会先清除该位： ■ SETUP 阶段完成 ■ 端点禁用 ■ 传输完成
位 30	EPTDIS	0x0	rw1s	端点禁用 (Endpoint disable) 应用程序可通过设置此位停止某个端点上的数据发送/传输，即使传输还没完成。 应用程序需要等到端点禁用中断产生才视为该端点已被禁用。控制器在设置端点禁用中断前会清除此位。应用程序只有在端点已经使用端点时才能设置此位。
位 29	SETD1PID/ SETODDFR	0x0	wo	设置 DATA1 PID (Set DATA1 PID) 仅适用于中断/批量 IN 端点，写此位可以将该寄存器的端点数据 PID 位设置为 DATA1。
位 28	SETD0PID/ SETEVENFR	0x0	rw	设置奇数帧 (Set odd frame) 仅适用于同步 IN 端点，写此位可以将奇偶帧位设置为奇数帧。 0：关闭 Set DATA1 PID 或者不强制奇数帧 1：使能 Set DATA1 PID 或者强制奇数帧
位 27	SNAK	0x0	wo	设置 DATA0 PID (Set DATA0 PID) 仅适用于中断/批量 IN 端点，写此位可将该寄存器的端点数据 PID 位设置为 DATA0。
位 26	CNAK	0x0	wo	设置偶数帧 (Set Even frame) 仅适用于同步 IN 端点，写此位可以将偶数帧/奇数帧位设置为偶数帧。 0：关闭 Set DATA0 PID 或不强制偶数帧 1：端点数据 PID 设置为 DATA0 或将 EOFNUM 设置为偶数帧
位 25: 22	TXFNUM	0x0	rw	设置 NAK (Set NAK) 通过对该位写操作可以设置端点的 NAK 位。应用程序可通过此位来控制端点上的 NAK 握手信号的发送。控制器在接收到 SETUP 数据包或传输结束中断时置起此位。 数值： 0：未设置 NAK 1：设置 NAK
位 21	STALL	0x0	rw	清除 NAK (Clear NAK) 通过对该位写操作可以清除端点的 NAK 位。 0：不清除 NAK 1：清除 NAK
位 20	保留	0x0	resd	发送 FIFO 的编号 (TxFIFO number)
位 19: 18	EPTYPE	0x0	rw	给对应端点分配 FIFO 编号，必须给每个有效的 IN 端点分配一个独立的 FIFO 编号。此位仅对 IN 端点有效。
位 18	保留	0x0	resd	STALL 握手 (STALL handshake)
位 17	保留	0x0	resd	适用于非控制非同步 IN 端点和 OUT 端点。
位 16	保留	0x0	resd	应用程序通过此位停止向该端点发送 USB 主机的令牌。
位 15	保留	0x0	resd	如果 NAK 位、全局非周期性 IN NAK 位或者全局 OUT NAK 位和该位同时置起时，STALL 位享有优先级。只有应用程序才能清除此位，控制器不能。
位 14	保留	0x0	resd	0：停止发送所有无效令牌 1：停止发送所有有效令牌
位 13	保留	0x0	resd	保持默认值。
位 12	保留	0x0	resd	端点类型 (Endpoint type)
位 11	保留	0x0	resd	表示逻辑端点支持的传输类型

				00: 控制; 01: 同步; 10: 块传输; 11: 中断。
位 17	NAKSTS	0x0	ro	<p>NAK 状态 (NAK status) 指示以下状态: 0: 表示控制器根据 FIFO 状态发送非 NAK 握手信号 1: 表示控制器正在发送 NAK 握手信号。 当此位置起时 (无论是应用程序设置的还是控制器设置的)</p> <ul style="list-style-type: none">■ 控制器停止接收 OUT 端点上的数据, 即使接收 FIFO 有剩余空间可以用来容纳传入的数据包。■ 对于非同步 IN 端点: 控制器停止发送端点上的数据, 即使发送 FIFO 还有待发送的数据。■ 对于同步 IN 端点: 控制器会发出一个零长度的数据包, 即使发送 FIFO 还有剩余空间。 不管该位是否置起, 控制器始终以 ACK 握手信号来响应 SETUP 数据包。
位 16	DPID/ EOFRNUM	0x0	ro	<p>端点数据 PID (Endpoint data PID) 仅适用于中断/批量 OUT 端点。 此位包含的是该端点上要传输或要发送的数据包的 PID 信息。在端点使能后, 应用程序需要设置该端点待发送或接收的首个数据包的 PID。应用程序通过该寄存器的 SetD1PID 和 SetD0PID 来设置 DATA0 或 DATA1 PID。 0: DATA0 1: DATA1</p> <p>奇/偶数帧 (Even/odd frame) 仅适用于同步 OUT 端点。 表示控制器传输或接收该端点的同步数据的帧号。应用程序通过该寄存器的 SETEVNFR 和 SETODDFR 设置希望传输或接收同步数据的偶数帧号/奇数帧号。 0: 偶数帧 1: 奇数帧</p>
位 15	USBACCEPT	0x0	rw	<p>USB 活跃端点 (USB active endpoint) 指示在当前配置和接口下, 该端点是否活跃。控制器在检测到 USB 复位之后会清除此位(除了端点 0)。应用程序在收到 SetConfiguration 和 SetInterface 命令时必须设置端点寄存器并置起此位。 0: 不活跃 1: 活跃</p>
位 14: 11	保留	0x0	resd	保持默认值。
位 10: 0	MPS	0x000	rw	最大包长度 (Maximum packet size) 应用程序通过此位设置当前逻辑端点的最大包长度, 以字节为单位。

22.6.5.11 OTGFS设备控制OUT端点0控制寄存器(OTGFS_DOEPCTL0)

本节介绍了控制 OUT 端点 0 控制寄存器。非零控制端点使用端点 1-7 寄存器。

域	简称	复位值	类型	功能
位 31	EPTENA	0x0	rw1s	端点使能 (Endpoint enable) 应用程序设置此位，以启动端点 0 上的数据传输。 控制器在产生以下端点中断之前会先清除该位： ■ SETUP 阶段完成； ■ 端点禁止； ■ 传输完成。
位 30	EPTDIS	0x0	ro	端点禁用 (Endpoint disable) 应用程序不能禁止控制 OUT 端点 0
位 29: 28	保留	0x0	resd	保持默认值。
位 27	SNAK	0x0	wo	设置 NAK (Set NAK) 通过对该位写操作可以设置端点的 NAK 位。应用程序可通过此位来控制端点上的 NAK 握手信号的发送。控制器在接收到 SETUP 数据包或传输结束中断时置起此位。
位 26	CNAK	0x0	wo	清除 NAK (Clear NAK) 通过对该位写操作可以清除端点的 NAK 位。
位 25: 22	保留	0x0	resd	保持默认值。
位 21	STALL	0x0	rw1s	STALL 握手 (STALL handshake) 应用程序置起此位，控制器会在收到 SETUP 令牌时清除此位。如果该位连同 NAK 位，全局非周期性 OUT NAK 位同时置起时，STALL 位享有优先级。 不管该位是否置起，控制器始终以 ACK 握手信号来响应 SETUP 数据包。
位 20	SNP	0x0	rw	监听模式 (Snoop mode) 此位将端点配置为 Snoop 模式。在 Snoop 模式下，控制器在将 OUT 包传输给应用存储器之前不会检查 OUT 包是否正确。
位 19: 18	EPTYPE	0x0	ro	端点类型 (Endpoint type) 硬件设置将此位设置为 0 用于控制端点类型。
位 17	NAKSTS	0x0	ro	NAK 状态 (NAK status) 表示为： 0： 表示控制器根据 FIFO 状态发送非 NAK 握手信号 1： 表示控制器正在发送 NAK 握手信号。 当此位置起时（无论是应用程序设置的还是控制器设置的），控制器都会停止接收数据，即使接收 FIFO 有可用空间。不管该位是否置起，控制器始终以 ACK 握手信号来响应 SETUP 数据包。
位 16	保留	0x0	resd	保持默认值。
位 15	USBACEPT	0x1	ro	USB 有效端点 (USB active endpoint) 此位始终置 1，表示不管在什么配置和接口中，控制端点 0 始终有效。
位 14: 2	保留	0x0000	resd	保持默认值。
位 1: 0	MPS	0x0	ro	最大包长度 (Maximum packet size) 控制 OUT 端点 0 的最大包长度与控制器 IN 端点 0 的设置是一样的，如下： 00: 64 字节； 01: 32 字节； 10: 16 字节； 11: 8 字节。

22.6.5.12 OTGFS设备OUT端点x控制寄存器(OTGFS_DOEPCTLx)(其中x为端点号, x=1…7)

应用程序通过此寄存器来控制除了端点0的其他端点的操作特性。

域	简称	复位值	类型	功能
位 31	EPTENA	0x0	rw1s	<p>端点使能 (Endpoint enable) 此位表示已设置好描述符结构和准备接收数据的数据缓冲区。控制器在产生以下端点中断之前会先清除该位： ■ SETUP 阶段完成 ■ 端点禁用 ■ 传输完成</p>
位 30	EPTDIS	0x0	ro	<p>端点禁用 (Endpoint disable) 应用程序可通过设置此位来停止某个端点上的数据发送/传输，即使传输还没完成。 应用程序需要等到端点禁用中断产生才视为该端点已被禁用。控制器在设置端点禁用中断前会清除此位。应用程序只有在端点已经使用端点时才能设置此位。 数值： 0：无作用 1：端点禁用</p>
位 29	SETD1PID/ SETODDFR	0x0	rw	<p>设置 DATA1 PID (Set DATA1 PID) 仅适用于中断/批量 OUT 端点，写此位可以将该寄存器的端点数据 PID 位设置为 DATA1。</p>
位 28	SETD0PID/ SETEVENFR	0x0	rw	<p>设置奇数帧 (Set odd frame) 仅适用于同步 OUT 端点，写此位可以将奇偶帧位设置为奇数帧。 0：关闭 Set DATA1 PID 或者不强制奇数帧 1：使能 Set DATA1 PID 或者强制奇数帧</p>
位 27	SNAK	0x0	wo	<p>设置 DATA0 PID (Set DATA0 PID) 仅适用于中断/批量 OUT 端点，写此位可将该寄存器的端点数据 PID 位设置为 DATA0。</p>
位 26	CNAK	0x0	wo	<p>设置偶数帧 (Set Even frame) 仅适用于同步 OUT 端点，写此位可以将偶数帧/奇数帧位设置为偶数帧。 0：关闭 Set DATA0 PID 或不强制偶数帧 1：端点数据 PID 设置为 DATA0 或将 EOFRNUM 设置为偶数帧</p>
位 25: 22	保留	0x0	resd	设置 NAK (Set NAK) 通过对该位写操作可以设置端点的 NAK 位。应用程序可通过此位来控制端点上的 NAK 握手信号的发送。控制器在接收到 SETUP 数据包或传输结束中断时置起此位。 数值： 0：未设置 NAK 1：设置 NAK
位 21	STALL	0x0	rw	清除 NAK (Clear NAK) 通过对该位写操作可以清除端点的 NAK 位。 数值： 0：不清除 NAK 1：清除 NAK
位 20	SNP	0x0	rw	适用于非控制，非同步 IN 端点和 OUT 端点。 应用程序通过此位停止向该端点发送 USB 主机的令牌。 如果 NAK 位，全局非周期性 IN NAK 位或者全局 OUT NAK 位以及该位同时置起时，STALL 位享有优先级。只有应用程序才能清除此位，控制器不能。
				监听模式 (Snoop mode) 设置此位将使端点进入监听模式。在监听模式下，控制器在将 OUT 数据包写入应用程序缓存区前不检查数据的正确性。

				端点类型 (Endpoint type) 表示逻辑端点支持的传输类型。
位 19: 18	EPTYPE	0x0	rw	00: 控制 01: 同步 10: 块传输 11: 中断
位 17	NAKSTS	0x0	ro	NAK 状态 (NAK status) 0: 表示控制器根据 FIFO 状态发送非 NAK 握手信号 1: 表示控制器正在发送 NAK 握手信号。 当此位置起时 (无论是应用程序设置的还是控制器设置的) ■ 控制器停止接收 OUT 端点上的数据，即使接收 FIFO 有剩余空间可以用来容纳传入的数据包。 ■ 对于非同步 IN 端点：控制器停止发送端点上的数据，即使发送 FIFO 还有待发送的数据。 ■ 对于同步 IN 端点：控制器会发出一个零长度的数据包，即使发送 FIFO 还有剩余空间。 不管该位是否置起，控制器始终以 ACK 握手信号来响应 SETUP 数据包。
位 16	DPID/ EOFRNUM	0x0	ro	端点数据 PID (Endpoint data PID) 仅适用于中断/批量 OUT 端点。 此位包含的是该端点上要传输或要发送的数据包的 PID 信息。在端点使能后，应用程序需要设置该端点待发送或接收的首个数据包的 PID。应用程序通过该寄存器的 SetD1PID 和 SetD0PID 来设置 DATA0 或 DATA1 PID。 0: DATA0 1: DATA1
位 15	USBACEPT	0x0	rw	奇/偶数帧 (Even/odd frame) 仅适用于同步 OUT 端点。 表示控制器传输或接收该端点的同步数据的帧号。应用程序通过该寄存器的 SETEVNFR 和 SETODDFR 设置希望传输或接收同步数据的偶数帧号/奇数帧号。 0: 偶数帧 1: 奇数帧
位 14: 11	保留	0x0	resd	USB 活跃端点(USB active endpoint) 指示在当前配置和接口下，该端点是否活跃。控制器在检测到 USB 复位之后会清除此位(除了端点 0)。应用程序在收到 SetConfiguration 和 SetInterface 命令时必须设置端点寄存器并置起此位。 0: 不活跃 1: 活跃
位 10: 0	MPS	0x000	rw	最大包长度(Maximum packet size) 应用程序通过此位设置当前逻辑端点的最大包长度，以字节为单位。

22.6.5.13 OTGFS设备IN端点x中断寄存器(OTGFS_DIEPINTx)(其中x为端点号, x=0…7)

本寄存器指示在发生 USB 及 AHB 相关事件时某个端点的状态，具体请参考图 22-2。当控制器中断寄存器的 IN 端点中断位(OTGFS_GINTSTS 寄存器的 IEPINT 位)为 1 时，程序必需先读设备所有端点中断寄存器(OTGFS_DAINT)来获得发生事件的端点号，然后再读相应端点的中断寄存器获得详细信息。应用程序必需通过清除此位来清除 OTGFS_DAINT 和 OTGFS_GINTST 寄存器的对应位。

域	简称	复位值	类型	功能
位 31: 8	保留	0x000000	resd	保持默认值。
位 7	TXFEMP	0x0	ro	发送 FIFO 空 (Transmit FIFO empty) 当端点的发送 FIFO 为全空或半空时，会生成中断。具体是半空还是全空取决于控制器 AHB 配置寄存器的发送 FIFO 空级别位。
位 6	INEPTNAK	0x0	rw1c	IN 端点 NAK 有效 (IN endpoint NAK effective) 在将对应的 DIEPCTLx.CNAK 设置为 1 之前，需要先写 1 来清除此位。 此中断表示应用程序设置的 IN 端点 NAK 位已经生效。 此中断并不能保证已在 USB 线上发送了 NAK 握手信号。 STALL 位的优先级高于 NAK 位。 此位仅适用于端点已使能的情况。
位 5	保留	0x0	resd	保持默认值。
位 4	INTKNTXFEMP	0x0	rw1c	当发送 FIFO 空时收到 IN 命令 (IN token received when TxFIFO is empty) 此位表示当相关的发送 FIFO(不管是周期性或非周期性)时接收到一个 IN 令牌。在收到 IN 令牌的端点上会生成中断。
位 3	TIMEOUT	0x0	rw1c	超时条件 (Timeout condition) 仅适用于控制 IN 端点，此位表示控制器已经侦测到该端点上最后一个 IN 令牌超时。
位 2	保留	0x0	resd	保持默认值。
位 1	EPTDISD	0x0	rw1c	端点禁用中断 (Endpoint disabled interrupt) 表示已经按照应用程序的请求禁用了端点。
位 0	XFERC	0x0	rw1c	传输完成中断 (Transfer completed interrupt) 表示 AHB 以及 USB 已完成该端点设置的传输任务。

22.6.5.14 OTGFS设备OUT端点x中断寄存器(OTGFS_DOEPINTx)(其中x为端点号, x=0…7)

此寄存器指示相应端点与 USB 和 AHB 相关的事件状态。具体请参考图 22-2。当控制器中断寄存器的 OUT 端点中断位(OTGFS_GINTSTS 寄存器的 OEPINT 位)为 1 时，应用程序必需先读设备所有端点中断寄存器(OTGFS_DAINT)来获得发生事件的端点号，然后再读相应端点的中断寄存器获得详细信息。应用程序必需通过清除此位来清除 OTGFS_DAINT 和 OTGFS_GINTST 寄存器的对应位。

域	简称	复位值	类型	功能
位 31: 7	保留	0x0000001	resd	保持默认值。
位 6	B2BSTUP	0x0	rw1c	收到连续的 SETUP 包 (Back-to-back SETUP packets received) 此位表示接收到不止 3 个连接的 SETUP 包。
位 5	保留	0x0	resd	保持默认值。
位 4	OUTTEPD	0x0	rw1c	端点禁用时接收到 OUT 令牌 (OUT token received when endpoint disabled) 仅对控制 OUT 端点有效。 此位表示在端点还没有使能的情况下接到了 OUT 令牌。在收到 OUT 令牌的端点上会生成中断。
位 3	SETUP	0x0	rw1c	SETUP 阶段完成 (SETUP phase done) 仅适用于控制 OUT 端点，此位表示该控制端点的 SETUP 阶段已完成，当前控制传输不再接收连续的 SETUP 数据包。一旦生成该中断，应用程序就可以解析所收到的 SETUP 数据包。
位 2	保留	0x0	resd	保持默认值。

位 1	EPTDISD	0x0	rw1c	端点禁止中断 (Endpoint disabled interrupt) 表示已经按照应用程序的请求禁用了端点。
位 0	XFERC	0x0	rw1c	传输完成中断 (Transfer completed interrupt) 表示 AHB 以及 USB 已完成该端点设置的传输任务。

22.6.5.15 OTGFS设备IN端点0传输长度寄存器(OTGFS_DIEPTSIZ0)

应用程序必须在使能端点 0 之前设置该寄存器。一旦通过设备控制端点 0 控制寄存器的端点使能位将端点 0 使能后，就只能通过控制器修改本寄存器。一旦控制器清除了端点使能位，则应用程序只能读此寄存器。

域	简称	复位值	类型	功能
位 31: 21	保留	0x000	resd	保持默认值。
位 20: 19	PKTCNT	0x0	rw	包数目(Packet count) 表示 USB 数据包总数目，其构成端点 0 的数据传输长度。
位 18: 7	保留	0x000	resd	每次从发送 FIFO 读取一个数据包时（不管是最大包长度还是短包），此位域就会自动递减。
位 6: 0	XFERSIZE	0x00	rw	保持默认值。 传输长度 (Transfer size) 表示端点 0 上的传输长度（以字节为单位）。当传输长度变为 0 时，控制器就会中断应用程序。在每个包结束后，传输长度可以设置成该端点的最大包长度。 控制器在每次将外部存储器的一个数据包写入发送 FIFO 时，此位域就会自动递减。

22.6.5.16 OTGFS设备OUT端点0传输长度寄存器(OTGFS_DOEPTSIZ0)

应用程序必须在使能端点 0 之前设置该寄存器。一旦通过设备控制端点 0 控制寄存器的端点使能位将端点 0 使能后，就只能通过控制器修改本寄存器。一旦控制器清除了端点使能位，则应用程序只能读此寄存器。

域	简称	复位值	类型	功能
位 31	保留	0x0	resd	保持默认值。
位 30: 29	SUPCNT	0x0	rw	SETUP 包数目 (SETUP packet count) 表示该端点能够收到的连续 SETUP 数据包数。 01: 1 个数据包； 10: 2 个数据包； 11: 3 个数据包。
位 28: 20	保留	0x000	resd	保持默认值。
位 19	PKTCNT	0	rw	包数目 (Packet count) 将数据包写入接收 FIFO 之后，此位自动递减为 0。
位 18: 7	保留	0x000	resd	保持默认值。 传输长度 (Transfer size) 表示端点 0 上的传输长度，以字节为单位。在传输长度变为 0 后，控制器会中断应用程序。传输长度可以设置成该端点的最大包长度，在每个包结束时中断。 控制器在每次将外部存储器的一个数据包写入发送 FIFO 时，此位域就会自动递减。 控制器在每次从接收 FIFO 读取一个数据包并将其写入外部存储器后，此位域就会自动递减。
位 6: 0	XFERSIZE	0x00	rw	

22.6.5.17 OTGFS设备IN端点x传输长度寄存器(OTGFS_DIEPTSIZx)(其中x为端点号, x=1…7)

应用程序必须在使能端点 x 之前设置该寄存器。一旦通过设备控制端点 x 控制寄存器的端点使能位将端点 x 使能后，就只能通过控制器修改本寄存器。一旦控制器清除了端点使能位，则应用程序只能读此寄存器。

域	简称	复位值	类型	功能
位 31	保留	0x0	resd	保持默认值。
位 30: 29 MC		0x0	rw	帧内包数目 (Multi count) 对于周期性 IN 端点来说，此位域表示 USB 总线上每个帧必须传输的包数目。控制器通过此位域计算同步 IN 端点发送的数据 PID。 01: 1 个数据包 10: 2 个数据包 11: 3 个数据包
位 28: 19 PKTCNT		0x000	rw	包数目 (Packet count) 表示 USB 数据包总数（端点上的数据传输长度）。 每次从发送 FIFO 读取一个数据包时（最大包长度和短包），此位域就会自动递减。
位 18: 0 XFERSIZE		0x00000	rw	传输长度 (Transfer Size) 此位域包含当前端点的传输字节数。控制器会在此位为 0 时产生中断并通知应用程序。可以配置此寄存器为端点的最大传输长度，并在每个数据包结束产生中断。 每次将外部存储器的一个数据包写入发送 FIFO 时，控制器就会自动递减此位域。

22.6.5.18 OTGFS设备IN端点传输FIFO状态寄存器(OTGFS_DTXFSTSx)(其中x为端点号, x=0…7)

只读寄存器，储存的是设备 IN 端点发送 FIFO 的可用空间信息。

域	简称	复位值	类型	功能
位 31: 16 保留		0x0000	resd	保持默认值。
位 15: 0 INEPTXFSAV		0x0200	ro	IN 端点发送 FIFO 剩余空间 (IN endpoint TxFIFO space avail) 表示端点发送 FIFO 的可用空间。以 32 位的字为单位。 0x0: 发送 FIFO 满; 0x1: 剩余 1 个字; 0x2: 剩余 2 个字; 0xn: 剩余 n 个字(其中 0 < n < 512); 0x200: 剩余 512 个字; 其他: 保留。

22.6.5.19 OTGFS设备OUT端点x传输长度寄存器

(OTGFS_DOEPTSIZx)(其中x 为端点号, x=1…7)

应用程序必须在使能端点 x 之前设置该寄存器。一旦通过设备控制端点 x 控制寄存器的端点使能位将端点 x 使能后，就只能通过控制器修改本寄存器。一旦控制器清除了端点使能位，则应用程序只能读此寄存器。

域	简称	复位值	类型	功能
位 31	保留	0x0	resd	保持默认值。
位 30: 29	RXDPID	0x0	ro	收到的数据 PID (Received data PID) 此位仅对同步 OUT 端点有效。表示收到的最后一个数据包的数据 PID: 00: DATA0; 01: DATA2; 10: DATA1; 11: MDATA。
位 28: 19	PKTCNT	0x000	rw	SETUP 包数目 (SETUP packet count) 此位仅对控制 OUT 端点有效。表示端点收到的连续的 SETUP 包的数量： 01: 1 个包； 10: 2 个包； 11: 3 个包。
位 18: 0	XFERSIZE	0x00000	rw	包数目 (Packet count) 指示该端点上传输的 USB 包数目。 每次向接收 FIFO 写入一个数据包时（最大包长度和短包），此位域就会自动递减。
				传输长度 (Transfer size) 此位指示该端点要传输的字节数。控制器在此域为 0 时会产生中断通知应用程序。可以配置此域 为端点的最大传输长度，并在每个数据包结束产生中断。 每当从接收 FIFO 读取一个数据包并将其写入外部存储器时，控制器就会自动递减此位域。

22.6.6 供电和时钟控制寄存器

22.6.6.1 OTGFS电源和时钟门控寄存器(OTGFS_PCGCCTL)

此寄存器既适用于主机模式也适用于设备模式。

域	简称	复位值	类型	功能
位 31: 5	保留	0x0000000	resd	保持默认值。
位 4	SUSPENDM	0x0	ro	物理层挂起 (PHY suspend) 指示 PHY 处于挂起状态
位 3: 1	保留	0x0	resd	保持默认值。
位 0	STOPPCLK	0x0	rw	停止 PHY 时钟 (Stop PHY clock) 当 USB 挂起或会话无效或者设备断开时，应用程序通过此位停止 PHY 时钟。当 USB 恢复或者有新的会话请求时，应用程序会清除此位。

23 HICK 自动时钟校准 (ACC)

23.1 简介

HICK 自动时钟校准器 (HICK ACC) 利用 USB 模块产生的 SOF 信号 (周期为 1 毫秒) 作为参考信号，实现对 HICK 时钟的采样和校准。

本模块主要功能就是实现对 USB 设备提供 $48MHz \pm 0.25\%$ 精度的时钟。

它采取“跨越回归”算法，可以将校准后的频率尽可能地靠近目标频率。

23.2 主要特性

- 可配置的中心频率
- 可配置的触发校准功能的边界频率
- 满足中心频率 $\pm 0.25\%$ 的精度要求
- 状态检测标志
 - 校准就绪标志
 - 一个错误检测标志
 - 参考信号丢失错误
- 2 个带标志的中断源
 - 校准就绪标志
 - 参考信号丢失错误
- 两种校验方式：粗校验和精校验。

23.3 中断请求

表 23-1 ACC 中断请求

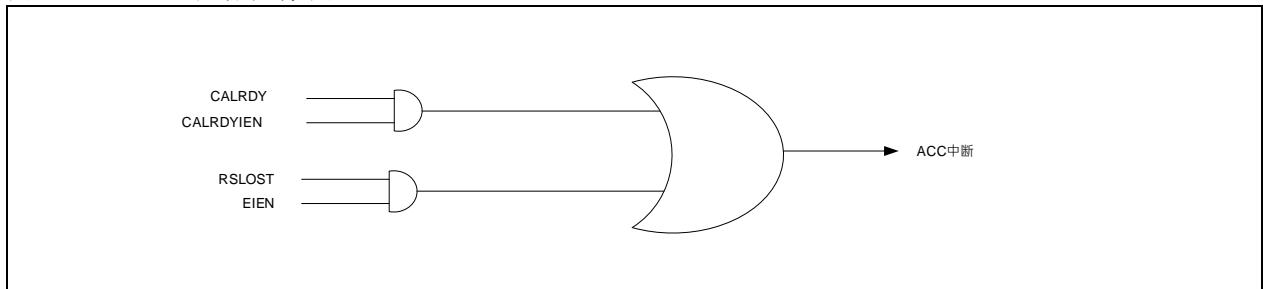
中断事件	事件标志	使能位
校准就绪	CALRDY	CALRDYIEN
参考信号丢失错误	RSLOST	EIEN

ACC 的各种中断事件被连接到同一个中断向量 (见下图)，有以下各种中断事件：

- 校准期间：当校准就绪和参考信号丢失错误。

如果设置了对应的使能控制位，这些事件就可以产生各自的中断。

图 23-1 ACC 中断映像图



23.4 功能概述

ACC 模块的功能：利用 USB 模块产生的 **SOF** 信号（周期为 1 毫秒）作为参考信号，实现对 HICK 时钟的采样和校准。特别的是，可以将 HICK 时钟的频率精度校准到 $\pm 0.25\%$ 以内的精度，从而满足高精度时钟要求的应用场景，例如 **USB** 应用。

本模块的信号均未外接到芯片管脚，而是和芯片内部的 CRM、HICK 等模块相连。

- **CRM_HICKCAL:** 复位和时钟控制（CRM）模块之 HICKCAL。此信号用于 bypass 模式下对内部高速时钟（HICK 模块）的校准，其值的大小由 CRM_CTRL 中的 HICKCAL[7:0] 定义。

- **CRM_HICKTRIM:** 复位和时钟控制（CRM）模块之 HICKTRIM。此信号用于 bypass 模式下对内部高速时钟校准（HICK）的校准，其值的大小由 CRM_CTRL 中的 HICKTRIM[5:0] 定义。

默认数值为 32，可以把 HICK 调整到 $8\text{MHz} \pm 0.25\%$ ；每步 CRM_HICKTRIM 的变化调整 HICK 的频率 20kHz （设计值）。

- **USB_SOF:** USB 设备解析给出的帧起始条件（USB Start-of-Frame）。其高电平宽度为 64 个系统时钟周期，周期为 1 毫秒的脉冲信号。

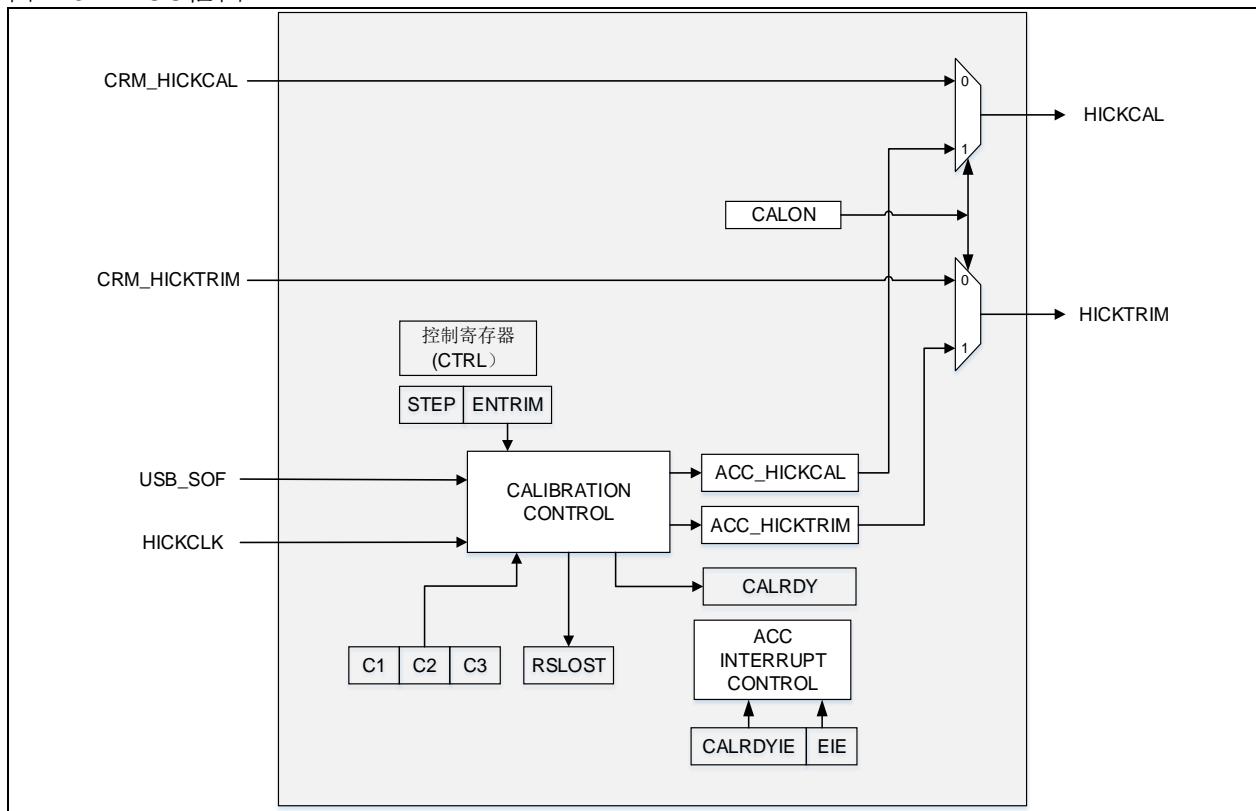
- **HICKCLK:** HICK 时钟。本系列的 HICK 模块输出的原始时钟频率为 48MHz ，但是 HICK 校准模块使用的采样时钟是除频（1/6）电路输出的时钟，频率约 8MHz 。

- **HICKCAL:** HICK 模块的校验信号。对于除频（1/6）后的 HICK 时钟来讲，HICKCAL 每改变一步，除频（1/6）后 HICK 时钟频率改变 40kHz （设计值），且为正相关。换句话说，HICKCAL 每增加一，除频（1/6）后 HICK 时钟频率会增加 40kHz （设计值）；HICKCAL 每减少一，除频（1/6）后 HICK 时钟频率会减少 40kHz 。

- **HICKTRIM:** HICK 模块的校验信号。对于除频（1/6）后的 HICK 时钟来讲，HICKTRIM 每改变一步，除频（1/6）后 HICK 时钟频率改变 20kHz （设计值），且为正相关。

关于以上寄存器中每个位的具体定义，请参考寄存器描述第 23.6 节：寄存器描述。

图 23-2 ACC 框图

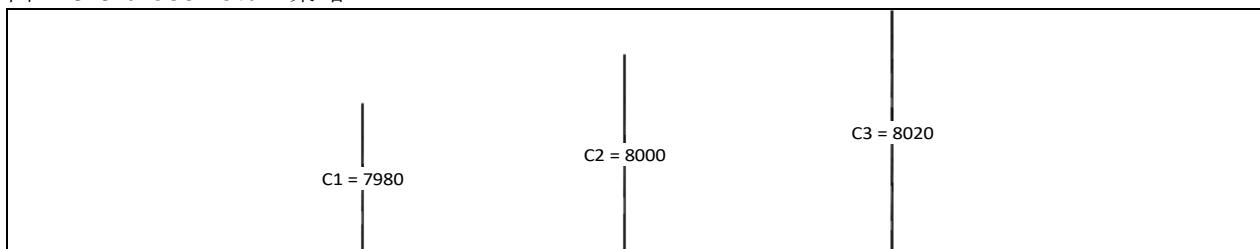


23.5 原理分析

USB_SOF 周期信号：1 毫秒的周期性必须是准确的，是自动校准模块能够正常工作的前提条件。

cross-return 策略：以计算出离理论值最近的校准值。从理论上来说，可以将校准后的实际频率调校到离目标频率（8MHz）约 0.5 个 step 的精度范围以内。

图 23-3 cross-return 策略



如上图所示，一旦触发自动校准的条件满足，自动校准就会按照 step 所规定的步长调整 HICKCAL 或者 HICKTRIM。

跨越 (cross) :

在满足自动校验的条件后的第一个 1 毫秒采样周期内的实际采样值要么小于 $C2$ ，要么大于 $C2$ 。

当这个值小于 $C2$ ，自动校准按照 step 的定义，增加 HICKCAL 或者 HICKTRIM，直到实际采样值比 $C2$ 大，实现实际采样值由小到大对 $C2$ 的跨越。

当这个值大于 $C2$ ，自动校准按照 step 的定义，减少 HICKCAL 或者 HICKTRIM，直到实际采样值比 $C1$ 小，实现实际采样值由大到小对 $C2$ 的跨越。

回归 (return) :

在跨越完成后，比较在跨越前后的实际采样值和 $C2$ 之间的差值（按绝对值计算），得到离 $C2$ 最近的实际采样值，从而得到最佳的校验值 HICKCAL 或者 HICKTRIM。

若跨越后的实际采样值和 $C2$ 之间的差值小于跨越前的实际采样值和 $C2$ 之间的差值，则以跨越后的校验值为准，并结束校验流程，直到满足下一个满足自动校验的条件。

若跨越后的实际采样值和 $C2$ 之间的差值大于跨越前的实际采样值和 $C2$ 之间的差值，则以跨越前的校验值为准，那么校验值会退回到一个 step，并返回到跨越前的那个校验值，并结束校验流程，直到满足下一个满足自动校验的条件。

按照 cross-return 策略，在理论上，可以得到离中心频率约 0.5 个 step 所对应的频率精度。

如下四种情形会启动自动校准：

第一， CALON 的上升沿（从 0 到 1）；

第二， 当 CALON=1 时，参考信号丢失之后又恢复；

第三， 当采样计数器的值小于 $C1$ ；

第四， 当采样计数器的值大于 $C3$ 。

在 CALON 的上升沿，即便采样计数器的值大于 $C1$ 并小于 $C3$ ，也会启动自动校准，其目的在于，在 CALON 之后，能够尽快将 HICK 的频率调整到中心频率的 0.5 个 step 以内。

以上四种情形的自动校准的结果均能将 HICK 的频率调整到中心频率的 0.5 个 step 以内。所以为了获得最佳的校准精度，建议将 step 保持为默认值 1。若将 step 设为 0，则 HICKCAL 或者 HICKTRIM 将无法改变，也即，无法校准。

23.6 寄存器描述

有关寄存器描述里所使用的缩写，请参考“寄存器描述表中使用的缩写列表”。必须用字（32位）的方式操作这些外设寄存器。

23.6.1 ACC寄存器地址映象

表 23-2 ACC寄存器映像和复位值

寄存器简称	基址偏移量	复位值
ACC_STS	0x00	0x0000 0000
ACC_CTRL1	0x04	0x0000 0100
ACC_CTRL2	0x08	0x0000 2080
ACC_C1	0x0C	0x0000 1F2C
ACC_C2	0x10	0x0000 1F40
ACC_C3	0x14	0x0000 1F54

23.6.2 状态寄存器 (ACC_STS)

域	简称	复位值	类型	功能
位 31: 9	保留	0x0000000	resd	保持默认值。
位 1	RSLOST	0x0	ro	参考信号丢失 (Reference Signal Lost) 0: 参考信号未丢失; 1: 参考信号丢失。 注: 在校验过程中, 当校准模块的采样计数器的值为 C2 的 2 倍时, 还未侦测到 SOF 参考信号, 则意味着参考信号丢失。内部状态机回归到 idle 状态, 除非再次侦测到 SOF 信号, 否则内部时钟采样计数器保持为 0。在 CALON 位清零后, 或者向 RSLOST 写入 0, 则 RSLOST 立即被清零。仅仅在 CALON=1 时, 才会检测参考信号。
位 0	CALRDY	0x0	ro	内部高速时钟就绪 (Internal high-speed clock calibration ready) 0: 内部 8MHz 振荡器校验没有就绪; 1: 内部 8MHz 振荡器校验就绪。 注: 由硬件置'1'来指示内部 8MHz 振荡器已经校验到离 8MHz 最近的频率上。在 CALON 位清零后, 或者向 CALRDY 写入 0, 则 CALRDY 立即被清零。

23.6.3 控制寄存器1 (ACC_CTRL1)

域	简称	复位值	类型	功能
位 31: 12	保留位	0x0000000	resd	硬件强制为 0
位 11: 8	STEP	0x1	rw	校准的步长 这 4 位定义了每次校准改变的值。 备注: 为了获得更高的校准精度, 建议将 step 设为 1。 当 ENTRIM=0, 仅校准 HICKCAL, 若 step 改变 1, 对应的 HICKCAL 也改变 1, HICK 频率改变 40KHz (设计值), 为正相关关系。 当 ENTRIM=1, 仅校准 HICKTRIM, 若 step 改变 1, 对应的 HICKTRIM 也改变 1, HICK 频率改变 20KHz (设计值), 为正相关关系。
位 7: 6	保留位	0x0	rw	硬件强制为 0
位 5	CALRDYIEN	0x0	rw	CALRDY 中断使能 (CALRDY interrupt enable) 该位由软件设置或清除。 0: 禁止产生中断; 1: 当 ACC_STS 中的 CALRDY 为'1'时, 产生 ACC 中断。
位 4	EIEN	0x0	rw	RSLOST 中断使能 (RSLOST error interrupt enable)

				该位由软件设置或清除。 0: 禁止产生中断; 1: 当 ACC_STS 中的 RSLOST 为'1' 时, 产生 ACC 中断。
位 3: 2	保留位	0x0	rw	硬件强制为 0
位 1	ENTRIM	0x0	rw	TRIM 使能 (Enable trim) 该位由软件设置或清除。 0: 仅校准 HICKCAL; 1: 仅校准 HICKTRIM。 注: 为了获得更高的校准精度, 建议将 ENTRIM 设为 1。
位 0	CALON	0x0	rw	Calibration 使能 (Calibration on) 该位由软件设置或清除。 0: 禁止校验; 1: 使能校验, 并开始搜寻 USB_SOF 上的脉冲。 注: 如果没有 USB_SOF 参考信号, 则本模块无法使用。 若对 HICK 时钟的精度没有要求, 也无需开启本模块以节省功耗。

23.6.4 控制寄存器2 (ACC_CTRL2)

域	简称	复位值	类型	功能
位 31: 14	保留位	0x00000	resd	硬件强制为 0
位 13: 8	HICKTRIM	0x20	ro	内部高速时钟自动调整 (Internal high-speed auto clock trimming) 该位由软件读取, 不可写。 由 ACC 自动校准模块来调整内部高速时钟, 它们被叠加在 ACC_HICKCAL[7: 0]数值上。这些位在 ACC_HICKCAL[7: 0]的基础上, 让用户可以输入一个调整数值, 根据电压和温度的变化调整内部 HICK RC 振荡器的频率。 默认数值为 32, 可以把 HICK 调整到 8MHz±0.25%; 每步 ACC_HICKTRIM 的变化调整 20kHz (设计值)。
位 7: 0	HICKCAL	0x80	ro	内部高速时钟自动校准 (Internal high-speed auto clock calibration) 该位由软件读取, 不可写。 由 ACC 自动校准模块来调整内部高速时钟, 让用户可以输入一个调整数值, 根据电压和温度的变化调整内部 HICK RC 振荡器的频率。 默认数值为 128, 可以把 HICK 调整到 8MHz±0.25%; 每步 ACC_HICKCAL 的变化调整 40kHz (设计值)。

23.6.5 比较值1 (ACC_C1)

域	简称	复位值	类型	功能
位 31: 16	保留位	0x0000	resd	硬件强制为 0。
位 15: 0	C1	0x1F2C	rw	比较值 1 (Compare 1) 该值是触发校准的下边界, 默认值为 7980。当自动校验模块在 1 毫秒的周期内采样的时钟个数小于或等于 C1, 则会触发自动校验; 当实际采样值 (1 毫秒内的时钟个数) 大于 C1, 且小于 C3, 则不会触发自动校验。

23.6.6 比较值2 (ACC_C2)

域	简称	复位值	类型	功能
位 31: 16	保留位	0x0000	resd	硬件强制为 0。
位 15: 0	C2	0x1F40	rw	比较值 2 (Compare 2) 该值确定了理想频率 (8MHz) 时钟在 1 毫秒为采样周期的时钟个数，默认值为 8000，也是其理论值。 该值是 cross-return 策略的中心点，以计算出离理论值最近的校准值。从理论上来说，可以将校准后的实际频率调教到离目标频率 (8MHz) 约 0.5 个 step 的精度范围内。

23.6.7 比较值3 (ACC_C3)

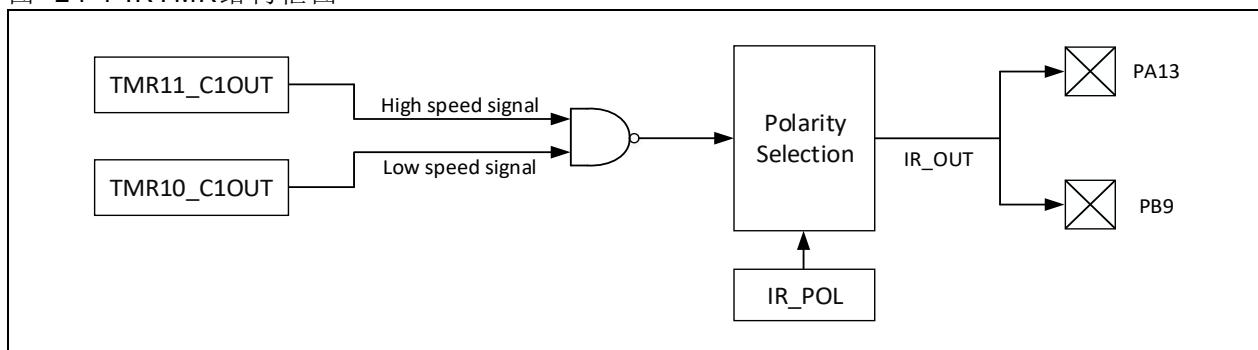
域	简称	复位值	类型	功能
位 31: 16	保留位	0x0000	resd	硬件强制为 0。
位 15: 0	C3	0x1F54	rw	比较值 3 (Compare 3) 该值是触发校准的上边界。当自动校验模块在 1 毫秒的周期内采样的时钟个数大于或等于 C3，则会触发自动校验； 当实际采样值（1 毫秒内的时钟个数）大于 C1，且小于 C3，则不会触发自动校验。

24 红外线接口 (IRTMR)

IRTMR 用于产生驱动红外 LED 的 IR_OUT 信号，进而实现红外控制功能。

IR_OUT 信号由低频调制包络信号和高频载波信号两部分组成，低频调制包络信号由 TMR10_C1OUT 提供；高频载波信号由 TMR11_C1OUT 提供，IR_POL 可控制输出的 IR_OUT 信号是否反向。IR_OUT 通过 PB9 或 PA13 通过复用功能输出（需提前配置为复用模式）。

图 24-1 IRTMR 结构框图



25 外部存储控制器（XMC）

25.1 XMC简介

XMC 是一个将 AHB 传输信号转换与外部存储器信号相互转换的外设。支持的外部存储器有静态随机存储器（SRAM）、NOR 闪存、PSRAM、和同步动态随机存储器 SDRAM。

25.2 XMC主要特征

NOR/PSRAM 界面有以下特征：

- 支持 4 个外部存储器的片选信号，拥有各自的控制寄存器
- 支持静态存储器件，包括：
 - 静态随机存储器（SRAM）
 - NOR 闪存
 - PSRAM
- 支持 8 位与 16 位数据宽度存储器
- 提供多种时序模式选择
 - 读写相同时序的 2 种模式
 - 读写不同时序的 4 种模式
 - 地址数据复用的模式
 - 同步模式
- 具可编程的时序控制寄存器
- 支持将 AHB 数据宽度转换为外部存储器适用的数据宽度

SDRAM 界面有以下特征：

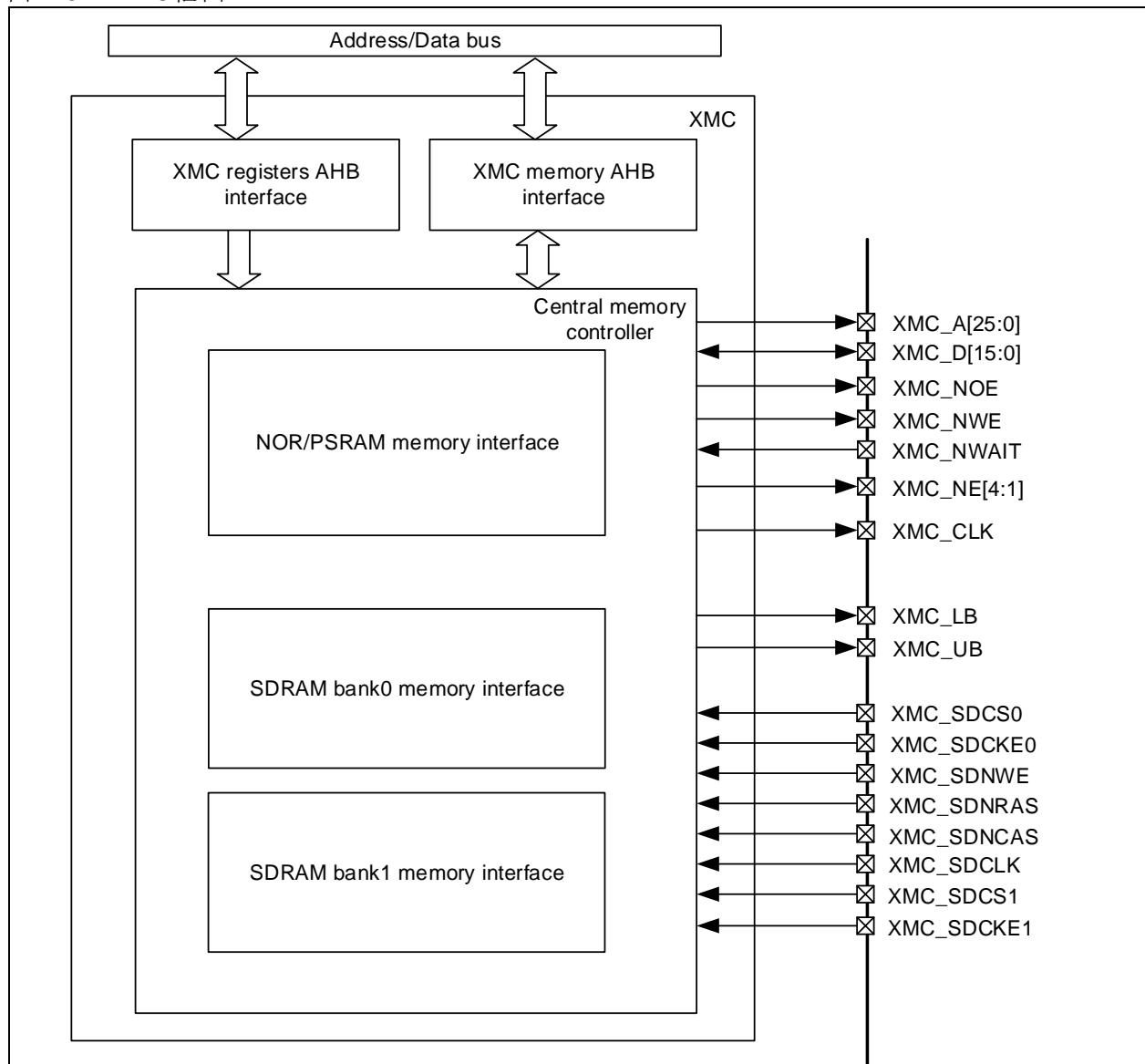
- 可以外接两个独立的 SDRAM 设备
- SDRAM 的数据总线宽度可以是 8 位或者 16 位
- 最大支持 13 位地址行，11 位地址列，4 个内部存储区域：4x16Mx16bit(128 MB)、4x16Mx8bit(64 MB)
- SDRAM 时钟可以是 HCLK 或 HCLK/2 或 HCLK/3 或 HCLK/4
- 支持 AHB byte、halfword 和 word 访问
- 可软件编程的 SDRAM 访问时序参数
- 自动进行行和存储区域边界管理
- 多存储区域乒乓访问
- 支持可软件编程刷新速率的自动刷新操作
- 支持自刷新模式
- 支持掉电模式
- 通过软件进行 SDRAM 上电初始化
- CAS 延迟可配置为 1/2/3 个 SDRAM 时钟周期
- 读 FIFO 可缓存，支持 6 行 x 32 位深度

25.3 XMC构造

25.3.1 框图

XMC 的架构如下所示

图 25-1 XMC 框图



与外部存储器沟通时,透过 NOR/PSRAM 界面与透过 SDRAM 界面所需要使用到的引脚不同,如表 25-1、与表 25-2 所列。

表 25-1 NOR/PSRAM 界面引脚

引脚	方向	介绍
XMC_CLK	输出	时钟
XMC_NE[x], x=1,4	输出	片选
XMC_NADV	输出	地址锁存或地址有效 (NL) 信号
XMC_A[x]	输出	地址总线
XMC_NOE	输出	输出使能信号
XMC_NWE	输出	写使能信号
XMC_LB、XMC_UB	输出	字节选择信号
XMC_D[15: 0]	读输入/写输出	数据总线/地址数据复用 总线
XMC_NWAIT	输入	等待信号

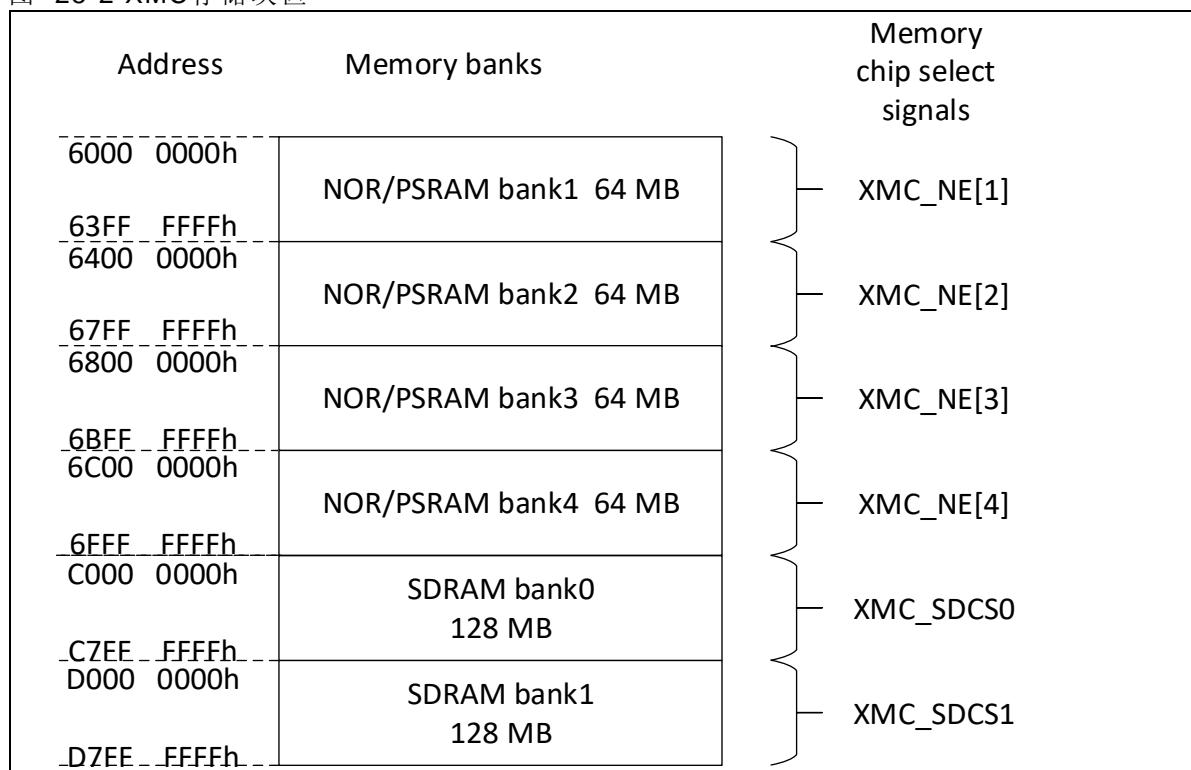
表 25-2 SDRAM 界面引脚

引脚	方向	SDRAM
XMC_SDCKE0	输出	DEVICE0 时钟使能
XMC_SDCKE1	输出	DEVICE1 时钟使能
XMC_SDSCS0	输出	DEVICE0 片选信号
XMC_SDSCS1	输出	DEVICE1 片选信号
XMC_SDCLK	输出	时钟信号
XMC_SDNRAS	输出	行选通信号
XMC_SDNCAS	输出	列选通信号
XMC_SDNWE	输出	写使能
XMC_LB、XMC_UB	输出	字节选择信号
XMC_A[12:0]	输出	地址总线
XMC_A[14]	输出	BANK 地址低位
XMC_A[15]	输出	BANK 地址高位
XMC_D[15: 0]	读输入/写输出	数据总线

25.3.2 地址映射

XMC 地址分为多个存储块区，如下所示。

图 25-2 XMC 存储块区



透过 HADDR 的部份特定位数，选择对哪个存储区块读写，如下表所示。

表 25-3 存储区块选择

HADDR[31: 28]	HADDR[27: 26]
0110: NOR/PSRAM	00: bank1 01: bank2 10: bank3 11: bank4
HADDR[31: 28]	HADDR[26:0]
1100: SDRAM BANK1	BANK 及行列地址映射
HADDR[31: 28]	HADDR[26:0]
1101: SDRAM BANK2	BANK 及行列地址映射

表 25-4 8位宽 SDRAM 地址映射

行大小配置	HADDR (AHB 内部地址线)																											
	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
11 位行大小配置	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
12 位行大小配置	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
13 位行大小配置	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
	保留									Bank [1:0]																		

表 25-5 16位宽 SDRAM 地址映射

行大小配置	HADDR (AHB 地址线)																											
	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
11 位行大小配置	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
12 位行大小配置	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
13 位行大小配置	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
	保留									Bank [1:0]																		
	保留									Bank [1:0]																		

25.4 NOR/PSRAM 界面

NOR/PSRAM 界面提供多种具有不同时序的访问模式，利用这些模式，可驱动多种存储器：NOR 闪存、SRAM、PSRAM 或 Cellular RAM。

4 个存储区块 bank1 到 bank4 有分开的控制寄存器，可使用不同的时序及不同的片选信号访问这 4 个存储区块。

25.4.1 操作方式

引脚使用

不同的外部存储器所需的信号不同，下表列出了典型信号。

表 25-6 NOR闪存与PSRAM典型引脚信号

XMC 引脚信号	NOR 闪存	PSRAM
XMC_CLK	时钟（同步模式）	时钟（同步模式）
XMC_NE[x]	片选信号	片选信号
XMC_NADV	地址锁存或地址有效信号	地址锁存或地址有效信号
XMC_A[25: 0]	地址总线	地址总线
XMC_NOE	输出使能信号	输出使能信号
XMC_NWE	写使能信号	写使能信号
XMC_LB、XMC_UB	无使用	XMC_LB: 低字节选信号 XMC_UB: 高字节选信号
XMC_D[15: 0]	数据总线 地址数据复用总线（复用与同步模式）	数据总线 地址数据复用总线（复用与同步模式）
XMC_NWAIT	NOR 闪存要求等待信号	PSRAM 要求等待信号

注意：若存储器数据宽度为 8 位，典型数据总线为 XMC_D[7: 0]。

访问地址

HADDR 的高地址用来选择存储区块，低地址选择数据存储地址。HADDR 是字节地址，XMC 可支持字节与半字地址的存储器，地址转换如下表所示。只要对特定地址作读写，XMC 即可根据 HADDR 启动片选信号并对外部存储器的地址做读写。

表 25-7 HADDR与外部存储器地址转换

外部存储器数据宽度	地址线连接	最大可访问存储器空间（位）
8 位	HADDR[25: 0]与 XMC_A[25: 0]相连。 复用与同步模式时 HADDR[15: 0]与 XMC_D[15: 0]在地址锁存时间相连。	64M 字节 x8=512 M 位
16 位	HADDR[26: 1]与 XMC_A[25: 0]相连。 复用与同步模式时 HADDR[16: 1]与 XMC_D[15: 0]在地址锁存时间相连。	(64M 字节 x16)/2=512 M 位

访问数据

在 AHB 数据宽度与存储器数据宽度不同时，XMC 针对外部存储器拥有的典型信号可做适度的处理，下表列出 XMC 支持的操作。

表 25-8 访问数据宽度与外部存储器数据宽度对照表

存储器	模式	AHB 数据宽度	存储器数据宽度	说明
SRAM	异步读写	8/16/32	8	1 次、分 2 次或 4 次 XMC 访问
	异步读写	8/16/32	16	使用字节信号 XMC_LB、 XMC_UB，1 次或分 2 次 XMC 访问
PSRAM	异步读	8	16	
	异步写	8	16	使用字节信号 XMC_LB、 XMC_UB
	异步读写	16	16	
	异步读写	32	16	分 2 次 XMC 访问
	同步写	8	16	使用字节信号 XMC_LB、 XMC_UB
	同步读写	16	16	
	同步读写	32	16	分 2 次 XMC 访问
NOR 闪存	异步读	8	16	
	异步读写	16	16	
	异步读写	32	16	分 2 次 XMC 访问
	同步读	16	16	
	同步读	32	16	分 2 次 XMC 访问

25.4.2 访问模式

XMC 提供多种行为不同的访问模式，每种访问会依据时序参数动作，如表 25-9 所示，用户需依照外部存储器的规格与应用需求进行编程。

XMC 提供的访问模式有：

- 读写相同时序的模式：模式1与模式2
- 读写不同时序的模式：模式A、B、C与D
- 地址数据线复用的复用模式
- 有时钟的同步模式

表 25-9 NOR/PSRAM参数寄存器

参数寄存器	意义	访问模式	单位
ADDRST	地址建立时间	1、2、A、B、C、D、复用	HCLK 周期
ADDRHT	地址保持时间	D、复用	HCLK 周期
DTST	数据建立时间	1、2、A、B、C、D、复用	HCLK 周期
DTLAT	数据延迟时间	同步	XMC_CLK 周期
CLKPSC	时钟分频系数	同步	HCLK 周期

时序控制除了时序参数寄存器外，若是开启等待使能位（NWASEN 或 NWSEN），XMC 会在数据建立其间检查 XMC_NWAIT 信号，若是 XMC_NWAIT 信号处在请求等待状态，XMC 便会等待 XMC_NWAIT 回到就绪状态再进行数据传输。

25.4.2.1 读写相同时序的模式

模式 1 与模式 2 读与写的时序皆是参照 XMC_BK1TMG 寄存器的配置。

模式 1

如表 25-10 与图 25-11 配置，XMC 即会使用模式 1 访问外部存储器。读时序如图 25-3 所示，写时序如图 25-4 所示。

表 25-10 模式 1 的 SRAM/NOR 闪存片选控制寄存器配置

域	名称	配置方式
位 31: 20	保留	0x0
位 19	MVMC: 对存储器写操作位	0x0
位 18: 16	CRPGS: CRAM 页大小选择	0x0
位 15	NWASEN: 异步传输等待信号使能	根据存储器规格配置
位 14	RWTD: 读写时序不同控制	0x0
位 13	NWSEN: 同步传输等待信号使能	0x0
位 12	WEN: 写使能	根据需求配置
位 11	NWTCFG: 等待时序配置	0x0
位 10	WRAPEN: 支持非对齐的成组模式	0x0
位 9	NWPOL: 等待信号极性	根据存储器规格配置
位 8	SYNCBEN: 同步突发模式使能	0x0
位 7	保留	0x1
位 6	NOREN: NOR 闪存访问使能	0x0
位 5: 4	EXTMDBW: 外部存储器数据宽度	根据存储器规格配置
位 3: 2	DEV: 存储器类型	根据存储器规格配置，除 0x2 (NOR 闪存) 外有效
位 1	ADMUXEN: 地址/数据复用使能	0x0
位 0	EN: 存储器块使能	0x1

表 25-11 模式 1 的 SRAM/NOR 闪存片选时序寄存器配置

域	名称	配置方式
位 31: 30	保留	0x0
位 29: 28	ASYNCM: 异步访问模式选择	0x0
位 27: 24	DTLAT: 数据延迟时间	0x0
位 23: 20	CLKPSC: 时钟分频系数	0x0
位 19: 16	BUSLAT: 总线延迟时间	XMC_NE[x]由上升沿到下降沿的时间，根据需求与存储器规格配置
位 15: 8	DTST: 数据建立时间	参照图 25-3 与图 25-4，根据需求与存储器规格配置
位 7: 4	ADDRHT: 地址保持时间	0x0
位 3: 0	ADDRST: 地址建立时间	参照图 25-3 与图 25-4，根据需求与存储器规格配置

图 25-3 NOR/PSRAM界面模式1读

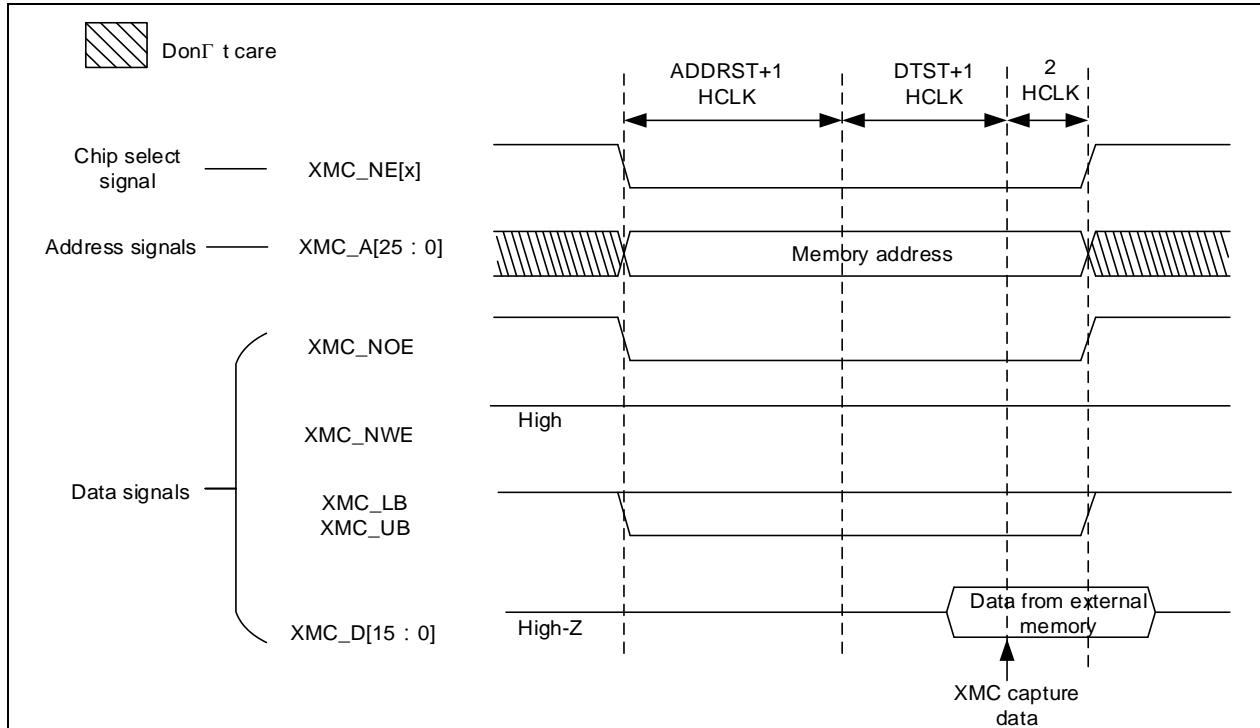
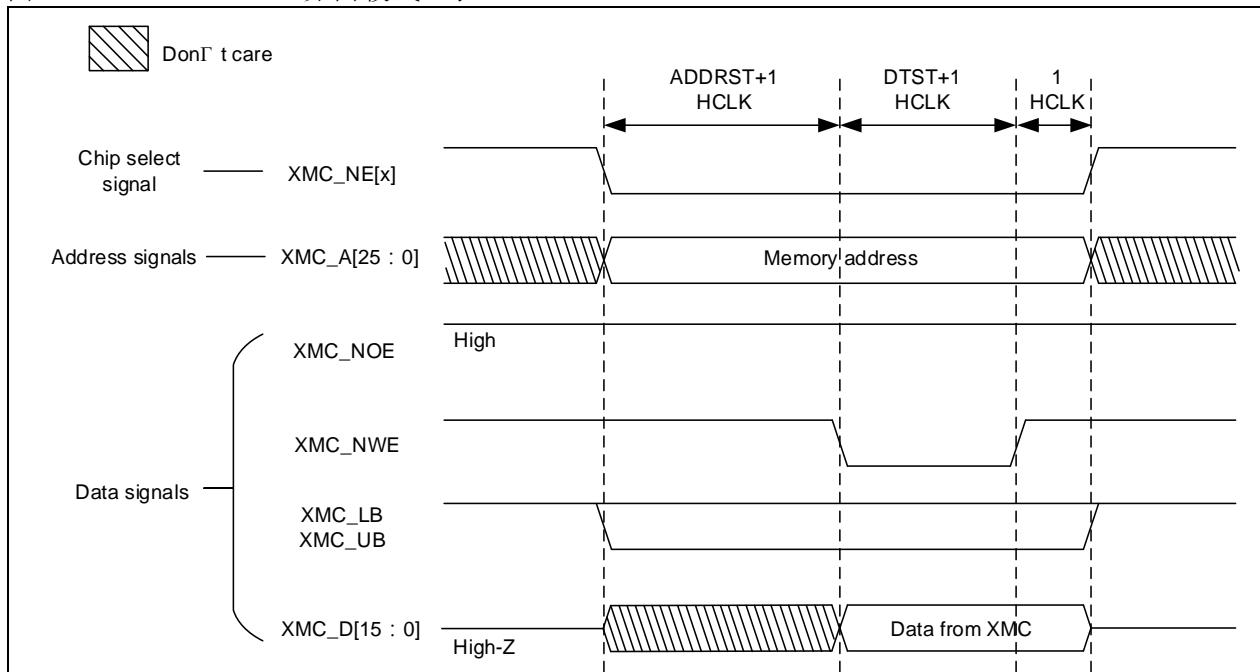


图 25-4 NOR/PSRAM界面模式1写



模式 2

如表 25-12 与表 25-13 配置, XMC 即会使用模式 2 访问外部存储器。读时序如图 25-5 所示, 写时序如图 25-6 所示。

表 25-12 模式 2 的 SRAM/NOR 闪存片选控制寄存器配置

域	名称	配置方式
位 31: 20	保留	0x0
位 19	MWMC: 对存储器写操作位	0x0
位 18: 16	CRPGS: CRAM 页大小选择	0x0
位 15	NWASEN: 异步传输等待信号使能	根据存储器规格配置
位 14	RWTD: 读写时序不同控制	0x0
位 13	NWSEN: 同步传输等待信号使能	0x0
位 12	WEN: 写使能	根据需求配置
位 11	NWTCFG: 等待时序配置	0x0
位 10	WRAPEN: 支持非对齐的成组模式	0x0
位 9	NWPOL: 等待信号极性	根据存储器规格配置
位 8	SYNCBEN: 同步突发模式使能	0x0
位 7	保留	0x1
位 6	NOREN: NOR 闪存访问使能	0x1
位 5: 4	EXTMDBW: 外部存储器数据宽度	根据存储器规格配置
位 3: 2	DEV: 存储器类型	0x2 (NOR 闪存)
位 1	ADMUXEN: 地址/数据复用使能	0x0
位 0	EN: 存储器块使能	0x1

表 25-13 模式 2 的 SRAM/NOR 闪存片选时序寄存器配置

域	名称	配置方式
位 31: 30	保留	0x0
位 29: 28	ASYNCM: 异步访问模式选择	0x0
位 27: 24	DTLAT: 数据延迟时间	0x0
位 23: 20	CLKPSC: 时钟分频系数	0x0
位 19: 16	BUSLAT: 总线延迟时间	XMC_NE[x]由上升沿到下降沿的时间, 根据需求与存储器规格配置
位 15: 8	DTST: 数据建立时间	参照图 25-5 与图 25-6, 根据需求与存储器规格配置
位 7: 4	ADDRHT: 地址保持时间	0x0
位 3: 0	ADDRST: 地址建立时间	参照图 25-5 与图 25-6, 根据需求与存储器规格配置

图 25-5 NOR/PSRAM界面模式2读

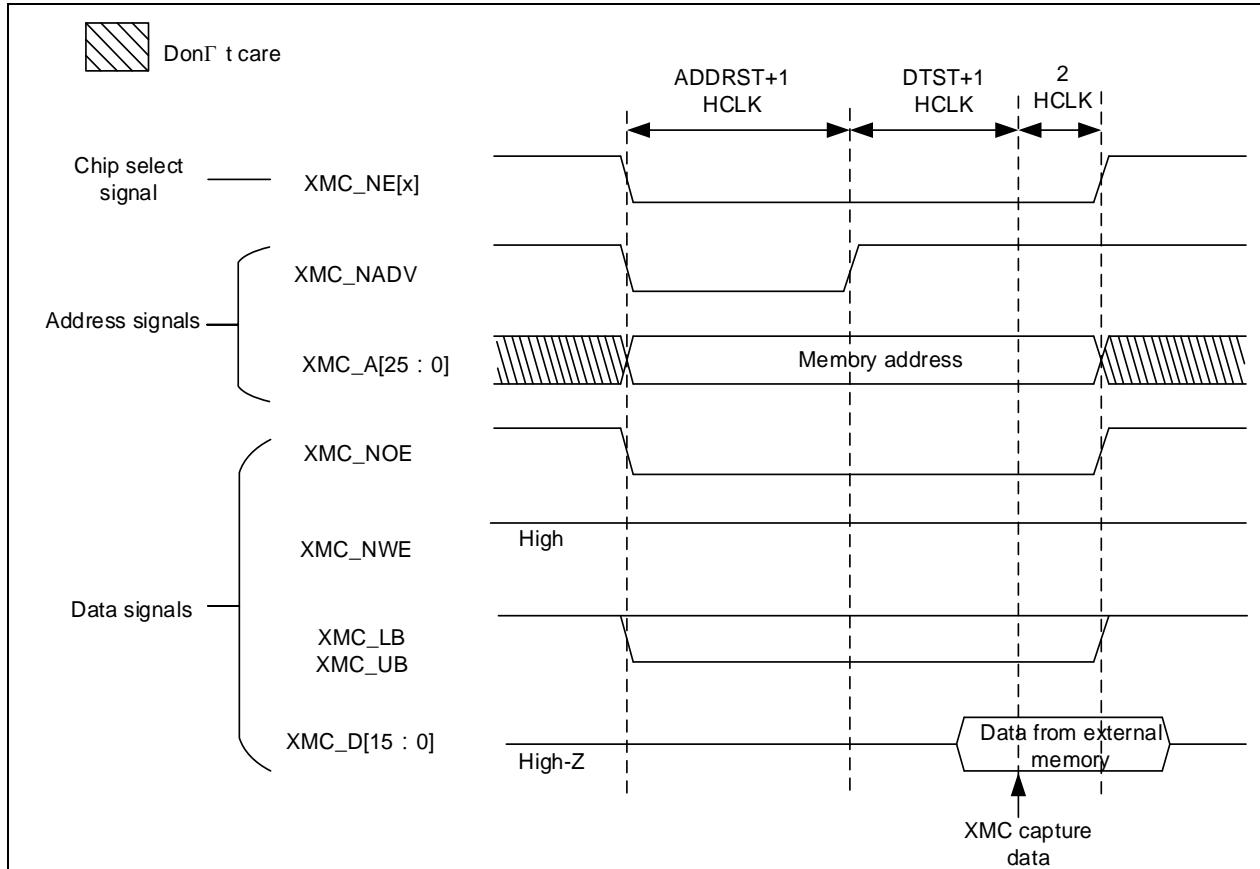
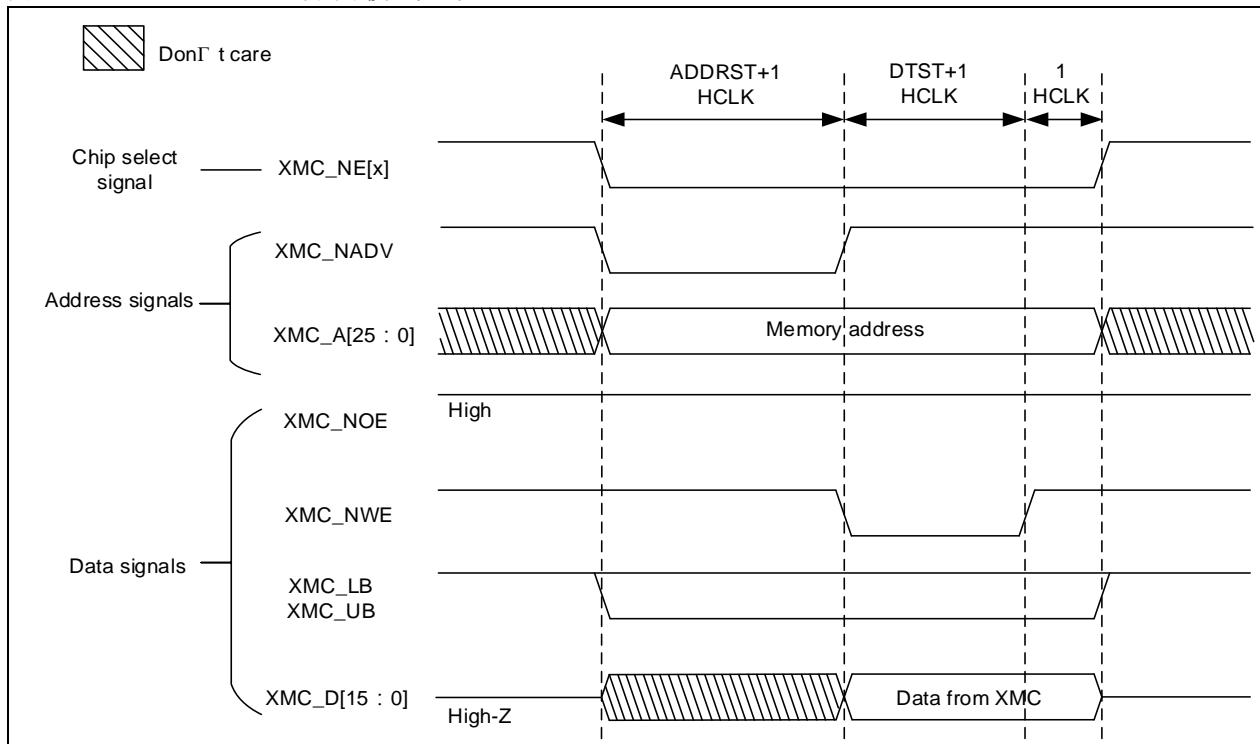


图 25-6 NOR/PSRAM界面模式2写



25.4.2.2 读写不同时序的模式

模式 A、模式 B、模式 C 与模式 D 读时序是参照 XMC_BK1TMGx 寄存器的配置，写时序是参照 XMC_BK1TMGWRx 寄存器的配置。除此之外，读与写可混合搭配不同的模式。

模式 A

如表 25-14、表 25-15、与表 25-16 配置，XMC 即会使用模式 A 访问外部存储器。读时序如图 25-7 所示，写时序如图 25-8 所示。

表 25-14 模式 A 的 SRAM/NOR 闪存片选控制寄存器配置

域	名称	配置方式
位 31: 20	保留	0x0
位 19	MVMC: 对存储器写操作位	0x0
位 18: 16	CRPGS: CRAM 页大小选择	0x0
位 15	NWASEN: 异步传输等待信号使能	根据存储器规格配置
位 14	RWTD: 读写时序不同控制	0x1
位 13	NWSEN: 同步传输等待信号使能	0x0
位 12	WEN: 写使能	根据需求配置
位 11	NWTCFG: 等待时序配置	0x0
位 10	WRAPEN: 支持非对齐的成组模式	0x0
位 9	NWPOL: 等待信号极性	根据存储器规格配置
位 8	SYNCBEN: 同步突发模式使能	0x0
位 7	保留	0x1
位 6	NOREN: NOR 闪存访问使能	0x0
位 5: 4	EXTMDBW: 外部存储器数据宽度	根据存储器规格配置
位 3: 2	DEV: 存储器类型	根据存储器规格配置，除 0x2 (NOR 闪存) 有效
位 1	ADMUXEN: 地址/数据复用使能	0x0
位 0	EN: 存储器块使能	0x1

表 25-15 模式 A 的 SRAM/NOR 闪存片选时序寄存器配置

域	名称	配置方式
位 31: 30	保留	0x0
位 29: 28	ASYNCM: 异步访问模式选择	0x0 (模式 A)
位 27: 24	DTLAT: 数据延迟时间	0x0
位 23: 20	CLKPSC: 时钟分频系数	0x0
位 19: 16	BUSLAT: 总线延迟时间	XMC_NE[x]由上升沿到下降沿的时间，根据需求与存储器规格配置
位 15: 8	DTST: 数据建立时间	参照图 25-7，根据需求与存储器规格配置
位 7: 4	ADDRHT: 地址保持时间	0x0
位 3: 0	ADDRST: 地址建立时间	参照图 25-7，根据需求与存储器规格配置

表 25-16 模式 A 的 SRAM/NOR 闪存写时序寄存器配置

域	名称	配置方式
位 31: 30	保留	0x0
位 29: 28	ASYNCM: 异步访问模式选择	0x0 (模式 A)
位 27: 20	保留	0x0
位 19: 16	BUSLAT: 总线延迟时间	XMC_NE[x]由上升沿到下降沿的时间，根据需求与存储器规格配置
位 15: 8	DTST: 数据建立时间	参照图 25-8，根据需求与存储器规格配置
位 7: 4	ADDRHT: 地址保持时间	0x0
位 3: 0	ADDRST: 地址建立时间	参照图 25-8，根据需求与存储器规格配置

图 25-7 NOR/PSRAM界面模式A读

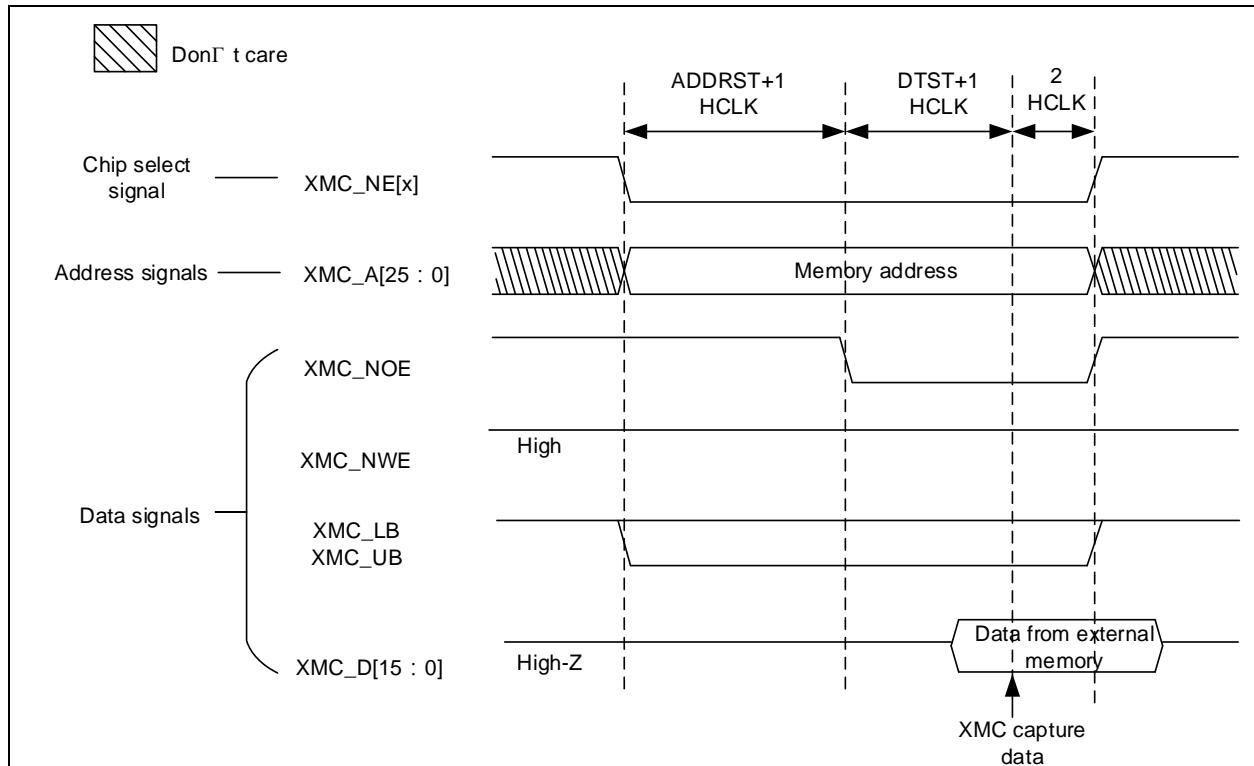
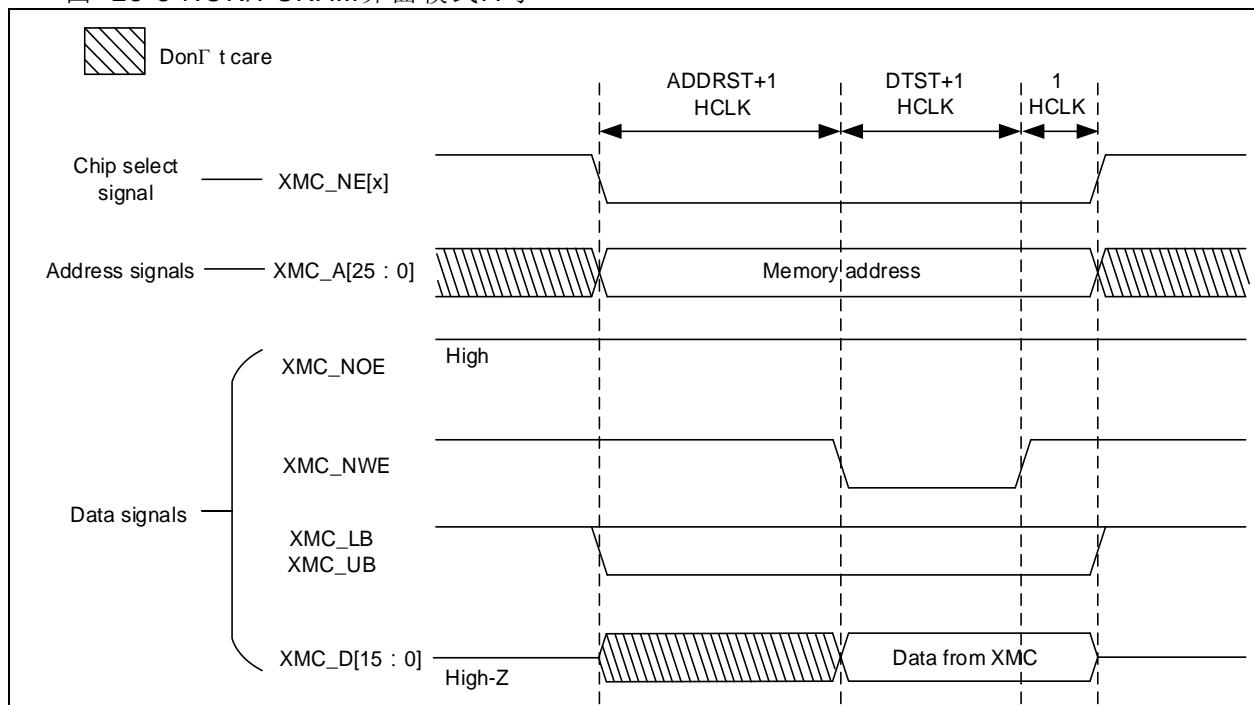


图 25-8 NOR/PSRAM界面模式A写



模式 B

如表 25-17、表 25-18、与表 25-19 配置 XMC 即会使用模式 B 访问外部存储器。读时序如图 25-9 所示，写时序如图 25-10 所示。

表 25-17 模式B的SRAM/NOR闪存片选控制寄存器配置

域	名称	配置方式
位 31: 20	保留	0x0
位 19	MWMC: 对存储器写操作位	0x0
位 18: 16	CRPGS: CRAM 页大小选择	0x0
位 15	NWASEN: 异步传输等待信号使能	根据存储器规格配置
位 14	RWTD: 读写时序不同控制	0x1
位 13	NWSEN: 同步传输等待信号使能	0x0
位 12	WEN: 写使能	根据需求配置
位 11	NWTCFG: 等待时序配置	0x0
位 10	WRAPEN: 支持非对齐的成组模式	0x0
位 9	NWPOL: 等待信号极性	根据存储器规格配置
位 8	SYNCBEN: 同步突发模式使能	0x0
位 7	保留	0x1
位 6	NOREN: NOR 闪存访问使能	0x1
位 5: 4	EXTMDBW: 外部存储器数据宽度	根据存储器规格配置
位 3: 2	DEV: 存储器类型	0x2 (NOR 闪存)
位 1	ADMUXEN: 地址/数据复用使能	0x0
位 0	EN: 存储器块使能	0x1

表 25-18 模式B的SRAM/NOR闪存片选时序寄存器配置

域	名称	配置方式
位 31: 30	保留	0x0
位 29: 28	ASYNCM: 异步访问模式选择	0x1 (模式 B)
位 27: 24	DTLAT: 数据延迟时间	0x0
位 23: 20	CLKPSC: 时钟分频系数	0x0
位 19: 16	BUSLAT: 总线延迟时间	XMC_NE[x]由上升沿到下降沿的时间，根据需求与存储器规格配置
位 15: 8	DTST: 数据建立时间	参照图 25-9，根据需求与存储器规格配置
位 7: 4	ADDRHT: 地址保持时间	0x0
位 3: 0	ADDRST: 地址建立时间	参照图 25-9，根据需求与存储器规格配置

表 25-19 模式B的SRAM/NOR闪存写时序寄存器配置

域	名称	配置方式
位 31: 30	保留	0x0
位 29: 28	ASYNCM: 异步访问模式选择	0x1 (模式 B)
位 27: 20	保留	0x0
位 19: 16	BUSLAT: 总线延迟时间	XMC_NE[x]由上升沿到下降沿的时间，根据需求与存储器规格配置
位 15: 8	DTST: 数据建立时间	参照图 25-10，根据需求与存储器规格配置
位 7: 4	ADDRHT: 地址保持时间	0x0
位 3: 0	ADDRST: 地址建立时间	参照图 25-10，根据需求与存储器规格配置

图 25-9 NOR/PSRAM界面模式B读

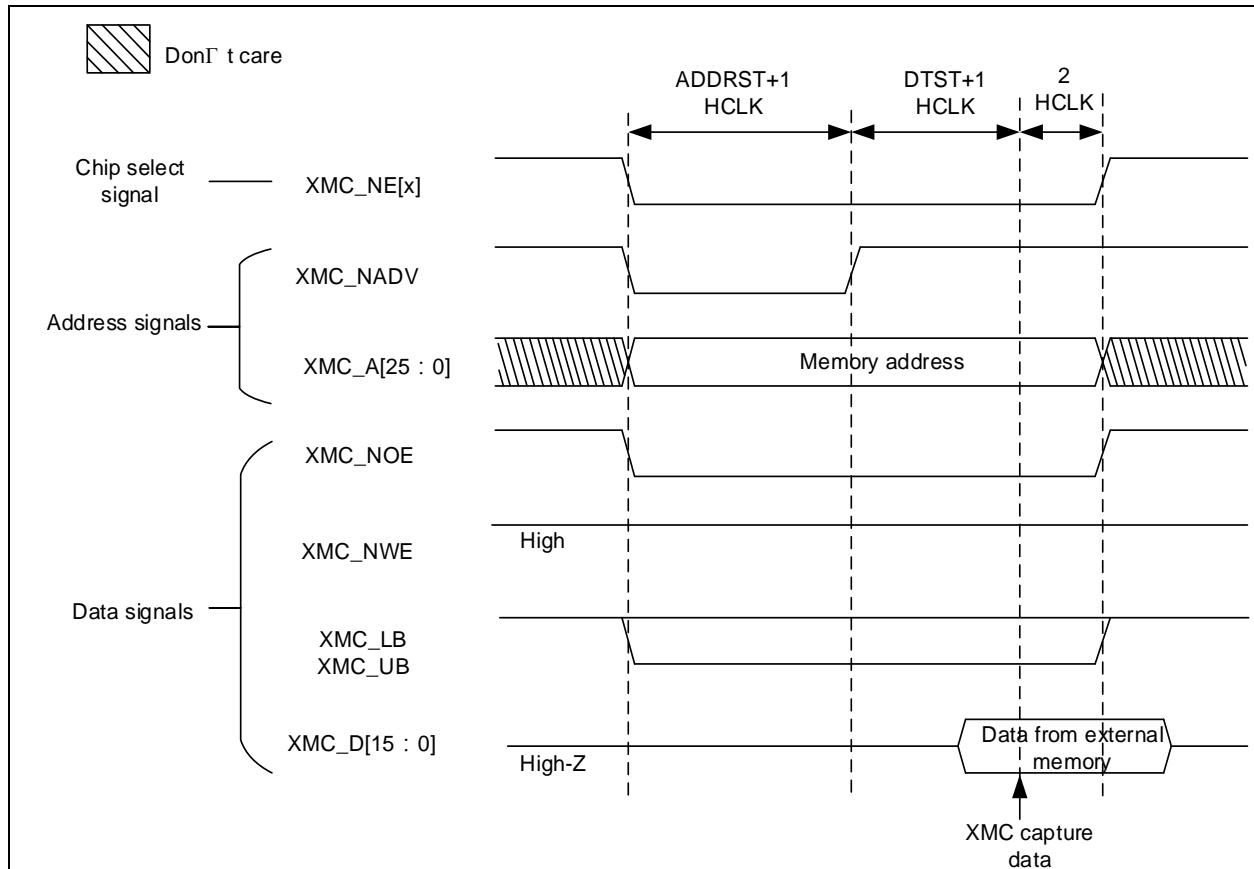
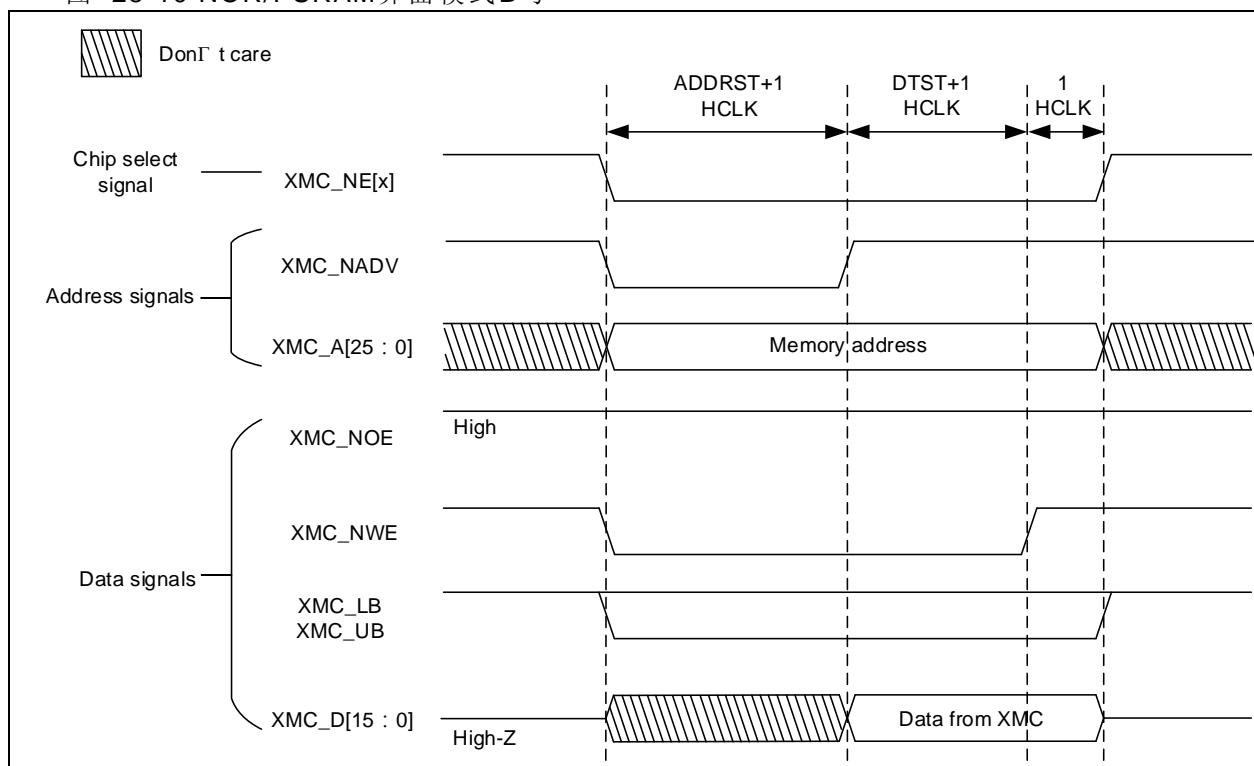


图 25-10 NOR/PSRAM界面模式B写



模式 C

如表 25-20、表 25-21、与表 25-22 配置，XMC 即会使用模式 C 访问外部存储器。读时序如图 25-11 所示，写时序如图 25-12 所示。

表 25-20 模式C的SRAM/NOR闪存片选控制寄存器配置

域	名称	配置方式
位 31: 20	保留	0x0
位 19	MWMC: 对存储器写操作位	0x0
位 18: 16	CRPGS: CRAM 页大小选择	0x0
位 15	NWASEN: 异步传输等待信号使能	根据存储器规格配置
位 14	RWTD: 读写时序不同控制	0x1
位 13	NWSEN: 同步传输等待信号使能	0x0
位 12	WEN: 写使能	根据需求配置
位 11	NWTCFG: 等待时序配置	0x0
位 10	WRAPEN: 支持非对齐的成组模式	0x0
位 9	NWPOL: 等待信号极性	根据存储器规格配置
位 8	SYNCBEN: 同步突发模式使能	0x0
位 7	保留	0x1
位 6	NOREN: NOR 闪存访问使能	0x1
位 5: 4	EXTMDBW: 外部存储器数据宽度	根据存储器规格配置
位 3: 2	DEV: 存储器类型	0x2 (NOR 闪存)
位 1	ADMUXEN: 地址/数据复用使能	0x0
位 0	EN: 存储器块使能	0x1

表 25-21 模式C的SRAM/NOR闪存片选时序寄存器配置

域	名称	配置方式
位 31: 30	保留	0x0
位 29: 28	ASYNCM: 异步访问模式选择	0x2 (模式 C)
位 27: 24	DTLAT: 数据延迟时间	0x0
位 23: 20	CLKPSC: 时钟分频系数	0x0
位 19: 16	BUSLAT: 总线延迟时间	XMC_NE[x]由上升沿到下降沿的时间，根据需求与存储器规格配置
位 15: 8	DTST: 数据建立时间	参照图 25-11，根据需求与存储器规格配置
位 7: 4	ADDRHT: 地址保持时间	0x0
位 3: 0	ADDRST: 地址建立时间	参照图 25-11，根据需求与存储器规格配置

表 25-22 模式C的SRAM/NOR闪存写时序寄存器配置

域	名称	配置方式
位 31: 30	保留	0x0
位 29: 28	ASYNCM: 异步访问模式选择	0x2 (模式 C)
位 27: 20	保留	0x0
位 19: 16	BUSLAT: 总线延迟时间	XMC_NE[x]由上升沿到下降沿的时间，根据需求与存储器规格配置
位 15: 8	DTST: 数据建立时间	参照图 25-12，根据需求与存储器规格配置
位 7: 4	ADDRHT: 地址保持时间	0x0
位 3: 0	ADDRST: 地址建立时间	参照图 25-12，根据需求与存储器规格配置

图 25-11 NOR/PSRAM界面模式C读

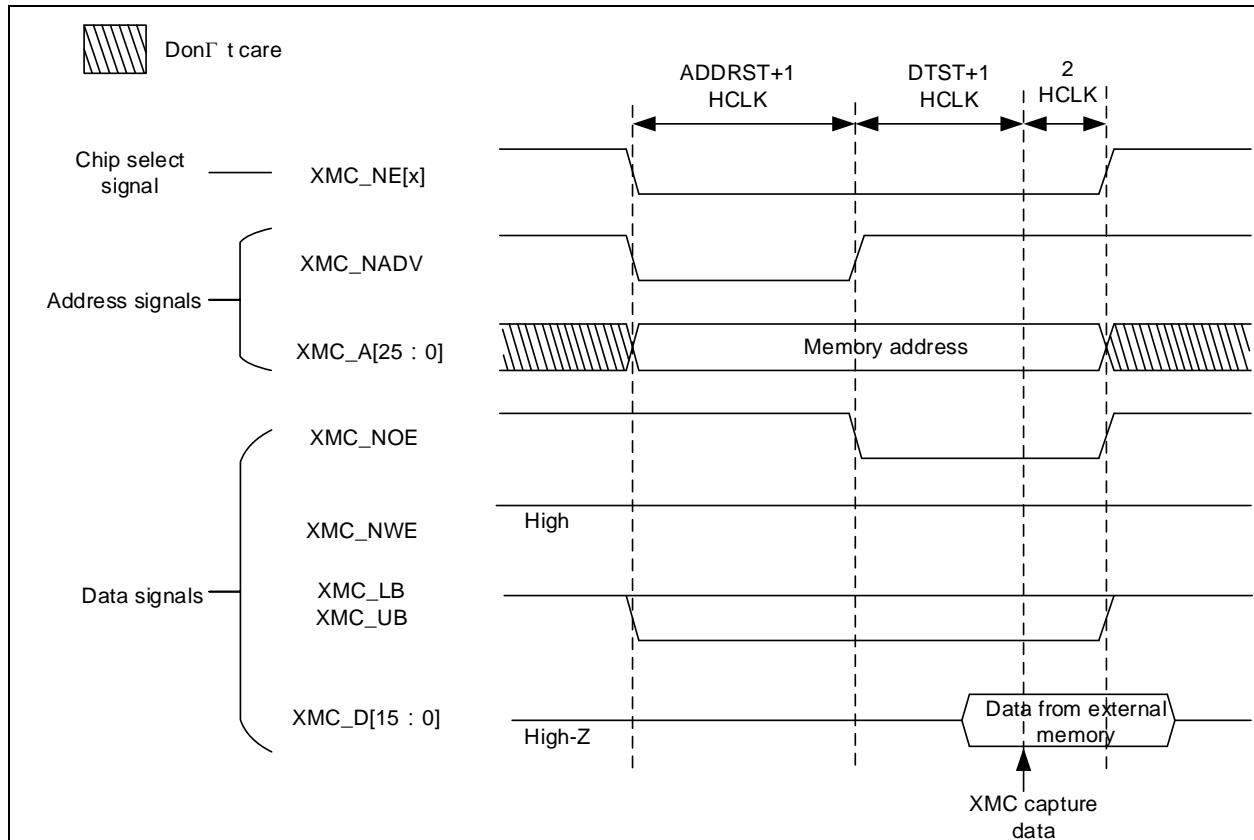
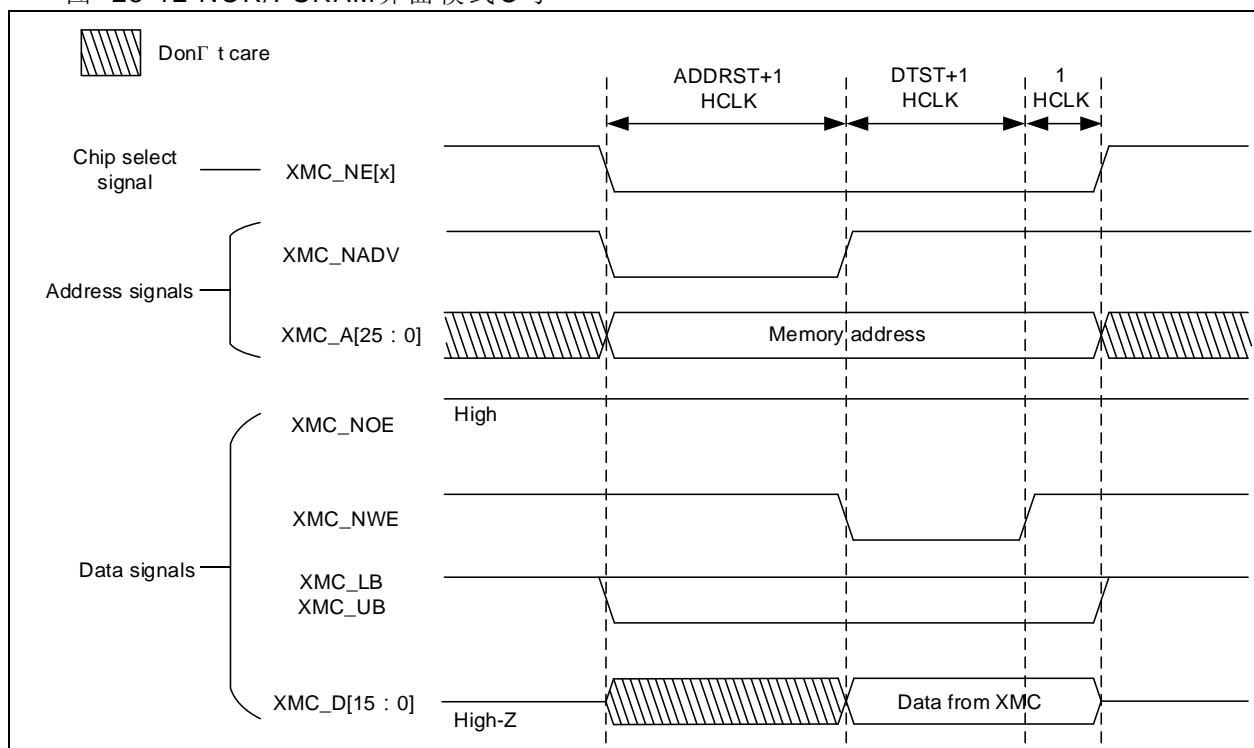


图 25-12 NOR/PSRAM界面模式C写



模式 D

如表 25-23、表 25-24、与表 25-25 配置，XMC 即会使用模式 D 访问外部存储器。读时序如图 25-13 所示，写时序如图 25-14 所示。

表 25-23 模式D的SRAM/NOR闪存片选控制寄存器配置

域	名称	配置方式
位 31: 20	保留	0x0
位 19	MWMC: 对存储器写操作位	0x0
位 18: 16	CRPGS: CRAM 页大小选择	0x0
位 15	NWASEN: 异步传输等待信号使能	根据存储器规格配置
位 14	RWTD: 读写时序不同控制	0x1
位 13	NWSEN: 同步传输等待信号使能	0x0
位 12	WEN: 写使能	根据需求配置
位 11	NWTCFG: 等待时序配置	0x0
位 10	WRAPEN: 支持非对齐的成组模式	0x0
位 9	NWPOL: 等待信号极性	根据存储器规格配置
位 8	SYNCBEN: 同步突发模式使能	0x0
位 7	保留	0x1
位 6	NOREN: NOR 闪存访问使能	根据存储器规格配置
位 5: 4	EXTMDBW: 外部存储器数据宽度	根据存储器规格配置
位 3: 2	DEV: 存储器类型	根据存储器规格配置
位 1	ADMUXEN: 地址/数据复用使能	0x0
位 0	EN: 存储器块使能	0x1

表 25-24 模式D的SRAM/NOR闪存片选时序寄存器配置

域	名称	配置方式
位 31: 30	保留	0x0
位 29: 28	ASYNCM: 异步访问模式选择	0x3 (模式 D)
位 27: 24	DTLAT: 数据延迟时间	0x0
位 23: 20	CLKPSC: 时钟分频系数	0x0
位 19: 16	BUSLAT: 总线延迟时间	XMC_NE[x]由上升沿到下降沿的时间，根据需求与存储器规格配置
位 15: 8	DTST: 数据建立时间	参照图 25-13，根据需求与存储器规格配置
位 7: 4	ADDRHT: 地址保持时间	参照图 25-13，根据需求与存储器规格配置
位 3: 0	ADDRST: 地址建立时间	参照图 25-13，根据需求与存储器规格配置

表 25-25 模式D的SRAM/NOR闪存写时序寄存器配置

域	名称	配置方式
位 31: 30	保留	0x0
位 29: 28	ASYNCM: 异步访问模式选择	0x3 (模式 D)
位 27: 20	保留	0x0
位 19: 16	BUSLAT: 总线延迟时间	XMC_NE[x]由上升沿到下降沿的时间，根据需求与存储器规格配置
位 15: 8	DTST: 数据建立时间	参照图 25-14，根据需求与存储器规格配置
位 7: 4	ADDRHT: 地址保持时间	参照图 25-14，根据需求与存储器规格配置
位 3: 0	ADDRST: 地址建立时间	参照图 25-14，根据需求与存储器规格配置

图 25-13 NOR/PSRAM 界面模式 D 读

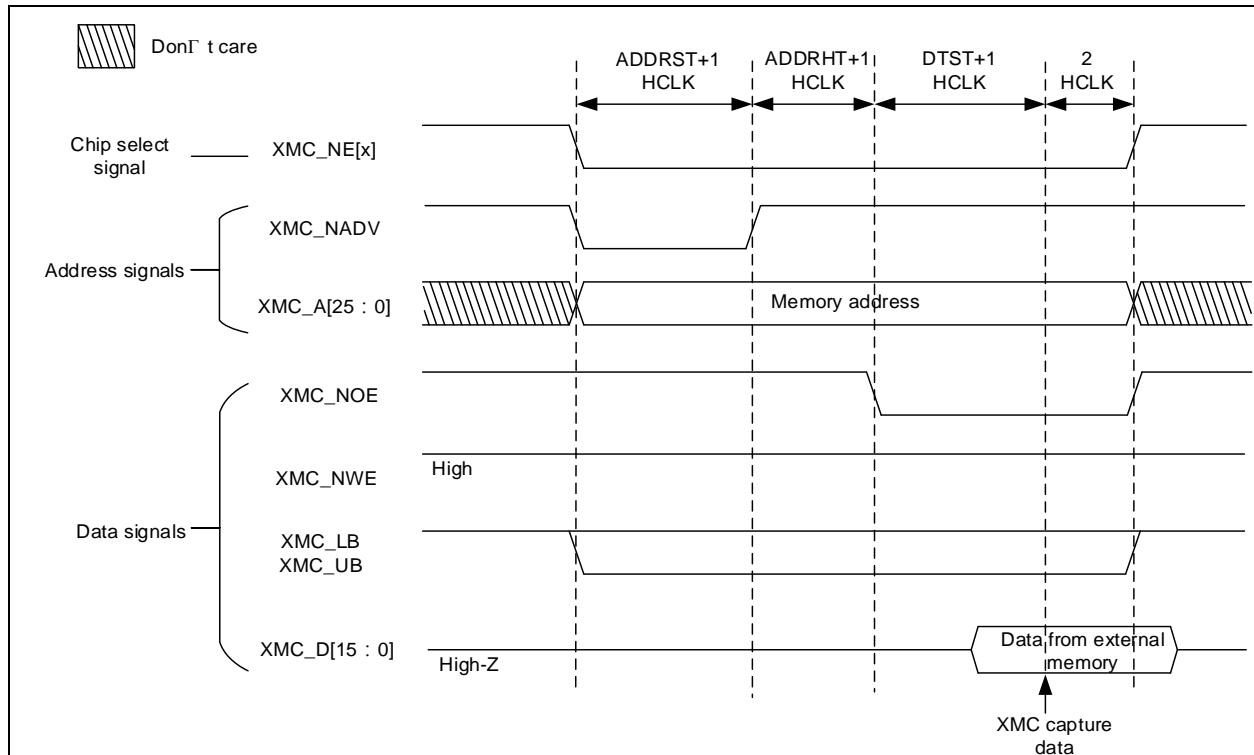
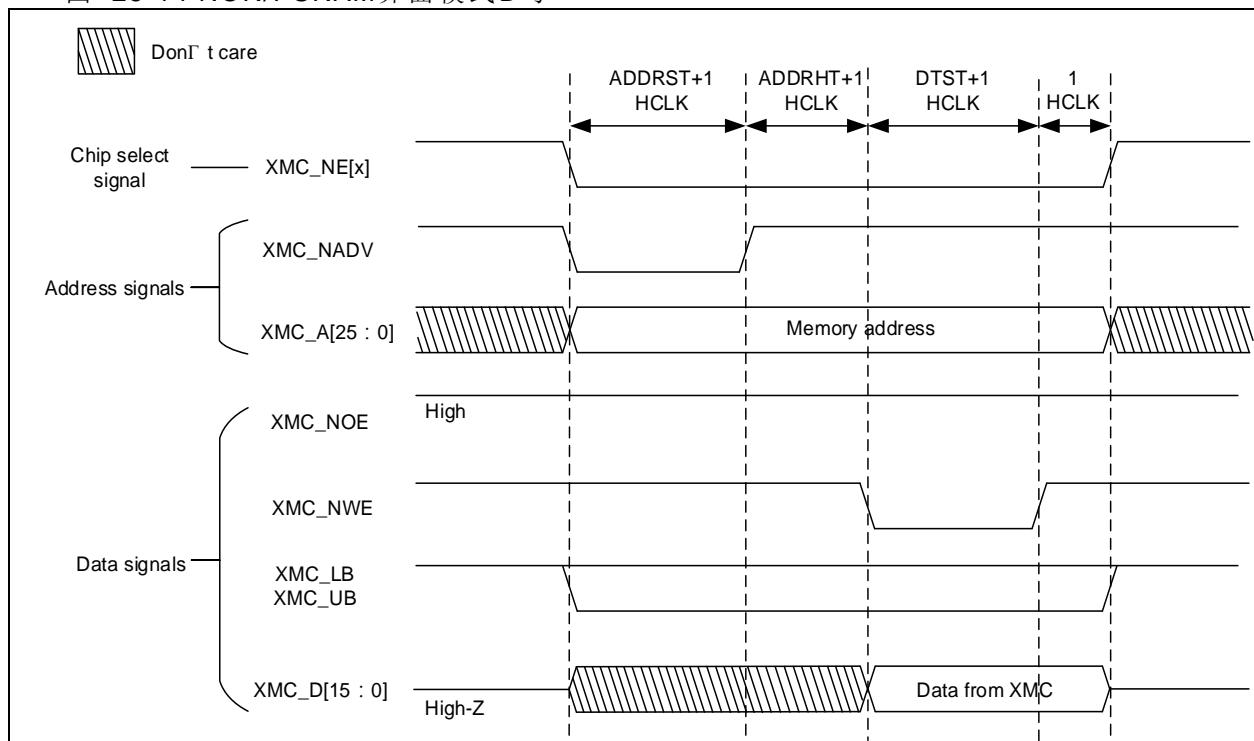


图 25-14 NOR/PSRAM 界面模式 D 写



25.4.2.3 复用模式

如表 25-26 与表 25-27 配置, XMC 即会使用复用模式访问外部存储器。读时序如图 25-15 所示, 写时序如图 25-16 所示。

表 25-26 复用模式的 SRAM/NOR 闪存片选控制寄存器配置

域	名称	配置方式
位 31: 20	保留	0x0
位 19	MWMC: 对存储器写操作位	0x0
位 18: 16	CRPGS: CRAM 页大小选择	0x0
位 15	NWASEN: 异步传输等待信号使能	根据存储器规格配置
位 14	RWTD: 读写时序不同控制	0x0
位 13	NWSEN: 同步传输等待信号使能	0x0
位 12	WEN: 写使能	根据需求配置
位 11	NWTCFG: 等待时序配置	0x0
位 10	WRAPEN: 支持非对齐的成组模式	0x0
位 9	NWPOL: 等待信号极性	根据存储器规格配置
位 8	SYNCBEN: 同步突发模式使能	0x0
位 7	保留	0x1
位 6	NOREN: NOR 闪存访问使能	根据存储器规格配置
位 5: 4	EXTMDBW: 外部存储器数据宽度	根据存储器规格配置
位 3: 2	DEV: 存储器类型	根据存储器规格配置, 除 0x0 (SRAM) 外有效
位 1	ADMUXEN: 地址/数据复用使能	0x1
位 0	EN: 存储器块使能	0x1

表 25-27 复用模式的 SRAM/NOR 闪存片选时序寄存器配置

域	名称	配置方式
位 31: 30	保留	0x0
位 29: 28	ASYNCM: 异步访问模式选择	0x0
位 27: 24	DTLAT: 数据延迟时间	0x0
位 23: 20	CLKPSC: 时钟分频系数	0x0
位 19: 16	BUSLAT: 总线延迟时间	XMC_NE[x]由上升沿到下降沿的时间, 根据需求与存储器规格配置
位 15: 8	DTST: 数据建立时间	参照图 25-15 与图 25-16, 根据需求与存储器规格配置
位 7: 4	ADDRHT: 地址保持时间	参照图 25-15 与图 25-16, 根据需求与存储器规格配置
位 3: 0	ADDRST: 地址建立时间	参照图 25-15 与图 25-16, 根据需求与存储器规格配置

图 25-15 NOR/PSRAM 界面复用模式读

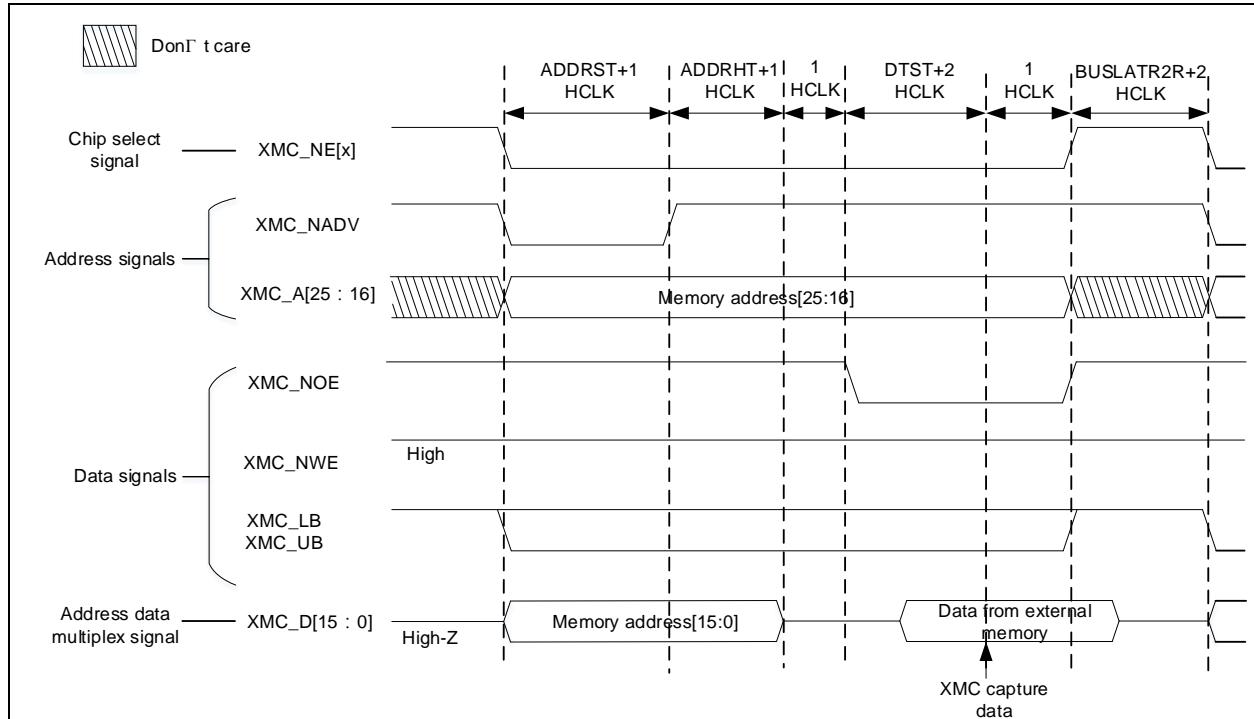
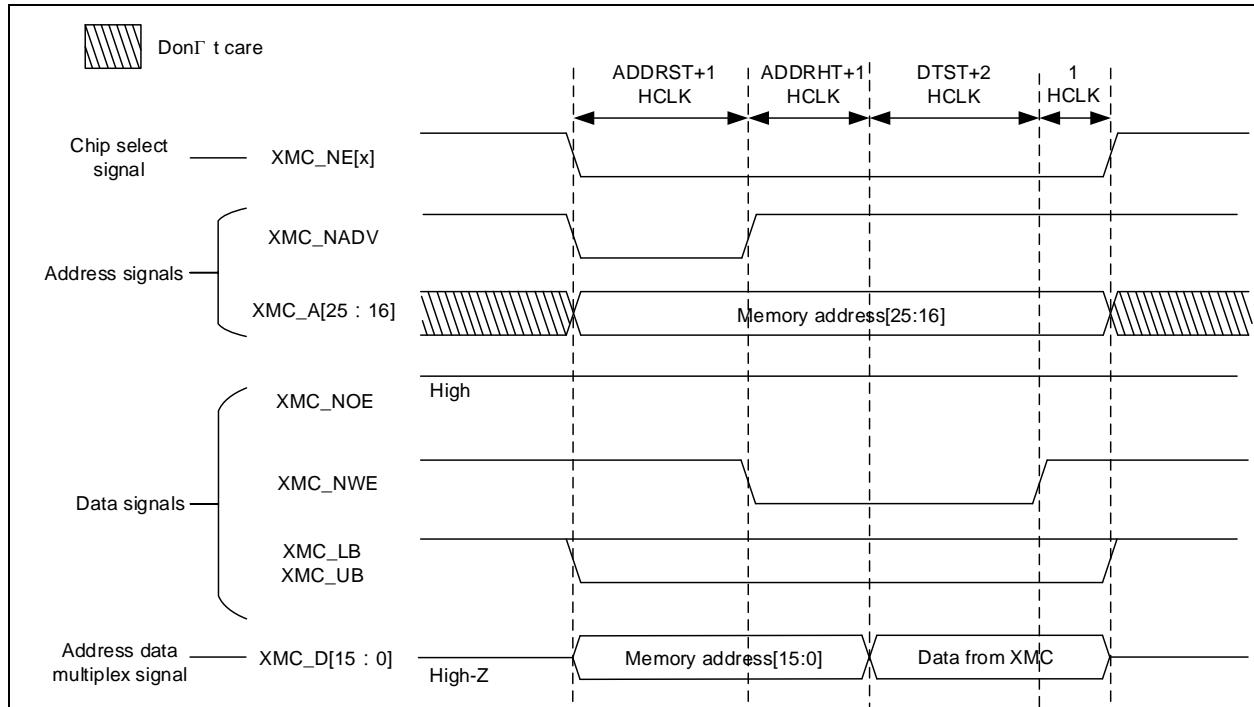


图 25-16 NOR/PSRAM 界面复用模式写



25.4.2.4 同步模式

如表 25-28 与表 25-29 配置, XMC 即会使用同步模式访问外部存储器。

若存储器在地址锁存与数据传输之间插入 XMC_NWAIT 信号, XMC 除了等待 DTLAT+1 个 XMC_CLK 外, 也会根据 XMC_NWAIT 进行等待。在数据传输途中, XMC 会根据 NWTCFG 的配置在 XMC_NWAIT 信号的下一个周期等待或是当个周期等待。

读时序如图 25-17 所示, 写时序如图 25-18 所示。图 25-17 与图 25-18 皆是 XMC_NWAIT 信号的下一个周期等待 (NWTCFG=0) 做示范。

表 25-28 同步模式的SRAM/NOR闪存片选控制寄存器配置

域	名称	配置方式
位 31: 20	保留	0x0
位 19	MWMC: 对存储器写操作位	0x1
位 18: 16	CRPGS: CRAM 页大小选择	根据存储器规格配置
位 15	NWASEN: 异步传输等待信号使能	0x0
位 14	RWTD: 读写时序不同控制	0x0
位 13	NWSEN: 同步传输等待信号使能	根据存储器规格配置
位 12	WEN: 写使能	根据需求配置
位 11	NWTCFG: 等待时序配置	根据存储器规格配置
位 10	WRAPEN: 支持非对齐的成组模式	根据需求配置
位 9	NWPOL: 等待信号极性	根据存储器规格配置
位 8	SYNCBEN: 同步突发模式使能	0x1
位 7	保留	0x1
位 6	NOREN: NOR 闪存访问使能	同步写: 0x0 同步读: 根据存储器规格配置
位 5: 4	EXTMDBW: 外部存储器数据宽度	根据存储器规格配置
位 3: 2	DEV: 存储器类型	同步写: 0x1 同步读: 根据存储器规格配置, 除 0x0 (SRAM) 外有效
位 1	ADMUXEN: 地址/数据复用使能	根据需求配置
位 0	EN: 存储器块使能	0x1

表 25-29 同步模式的SRAM/NOR闪存片选时序寄存器配置

域	名称	配置方式
位 31: 30	保留	0x0
位 29: 28	ASYNCM: 异步访问模式选择	0x0
位 27: 24	DTLAT: 数据延迟时间	参照图 25-17 与图 25-18, 根据需求与存储器规格配置
位 23: 20	CLKPSC: 时钟分频系数	XMC_CLK 周期为 HCLK 周期*(CLKPSC+1)。 参照图 25-17 与图 25-18, 根据需求与存储器规格配置
位 19: 16	BUSLAT: 总线延迟时间	0x0
位 15: 8	DTST: 数据建立时间	0x0
位 7: 4	ADDRHT: 地址保持时间	0x0
位 3: 0	ADDRST: 地址建立时间	0x0

图 25-17 NOR/PSRAM 界面同步模式复用读

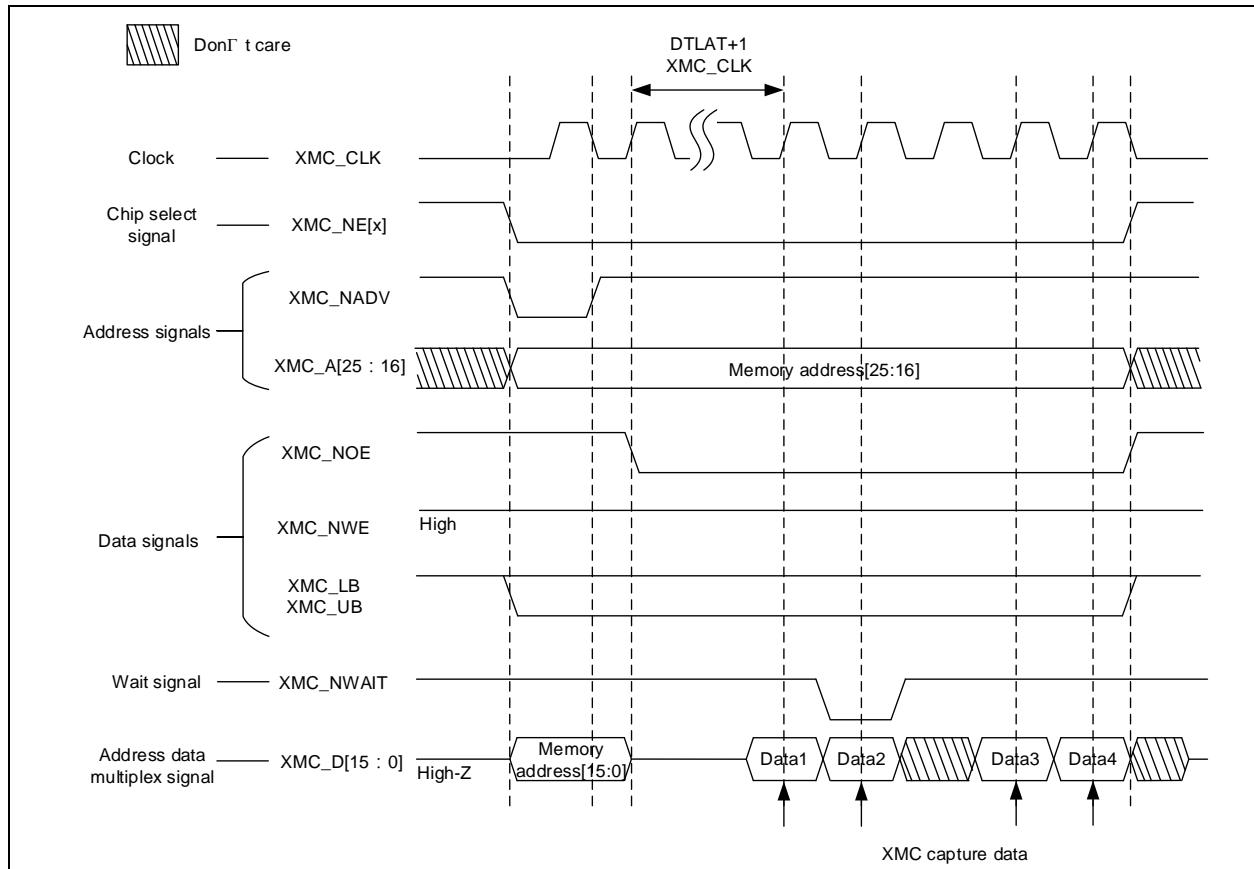
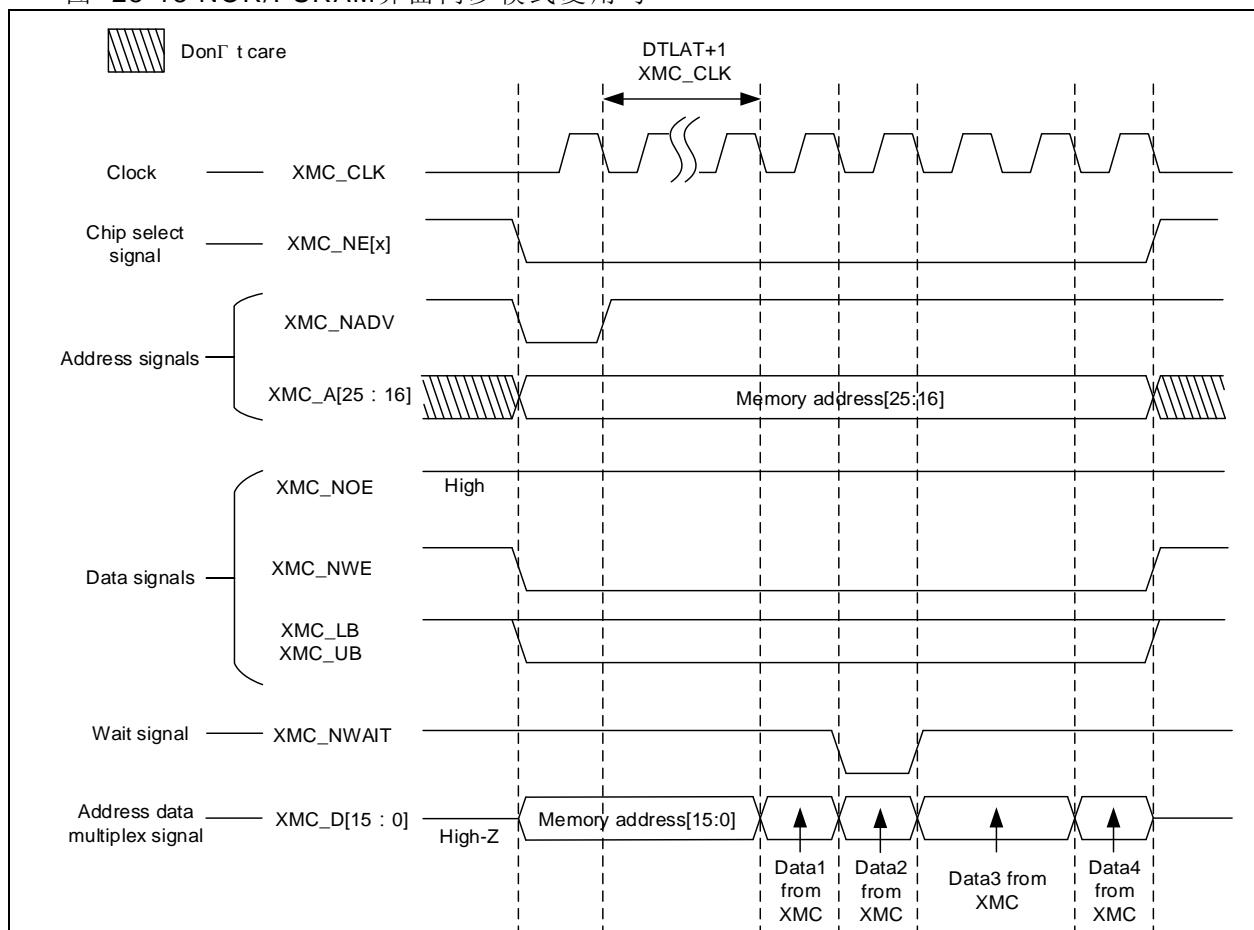


图 25-18 NOR/PSRAM 界面同步模式复用写



25.5 SDRAM界面

25.5.1 SDRAM访问管理

SDRAM 初始化流程

使用 SDRAM 之前必须遵循 JEDEC 规范中定义的 SDRAM 初始化流程。

通过应用程序对 SDRAM 设备进行初始化，如果想对两个 device 进行初始化，需要设置 SDRAM_CMD 寄存器中的 BK1 和 BK2 位为 1，使初始化命令对两个 device 都有效。详细的初始化流程如下：

1. 根据外挂 SDRAM 设备行列以及总线宽度，BANK 数目等信息配置 SDRAM_CTRLx 寄存器。同时控制器对 SDRAM 的读写方式也要配置到 SDRAM_CTRLx 寄存器。
2. 配置访问存储器的时序信息到 SDRAM_TMx 寄存器中。TRP 和 TRC 时序参数必须配置到 SDRAM_TM1 寄存器中。
3. 将 CMD 位置为“001”开始为存储器提供时钟信号。
4. 等待 100us。
5. 将 CMD 位置为“010”发送“全部预充电”命令。
6. 将 CMD 位置为“011”并配置连续自动刷新命令 (ART) 的数量，通常为 8 个。
7. 配置 MRD 字段（controller 只支持 BL=1），将 CMD 位置为“100”发送“加载模式寄存器”命令。如果两个 SDRAM 设备的模式寄存器不同，则此步骤必须重复两次（分别置 BK1 和 BK2 位为 1）。
8. 配置 SDRAM_RCNT 寄存器中的刷新计数器以决定 SDRAM 自动刷新速率。

初始化完成后，SDRAM 设备已做好接受命令的准备。如果 SDRAM 在被访问期间发生系统复位，必须在复位后重新初始化 SDRAM 设备，才能对 SDRAM 进行新的访问。

如果加载模式寄存器命令和自刷新命令同时作用到两个 SDRAM 设备，将按 SDRAM_TM1 寄存器中为 SDRAM 设备 1 配置的时序参数 (TMRD、TRAS 和 TXSR 时序) 发出访问命令。

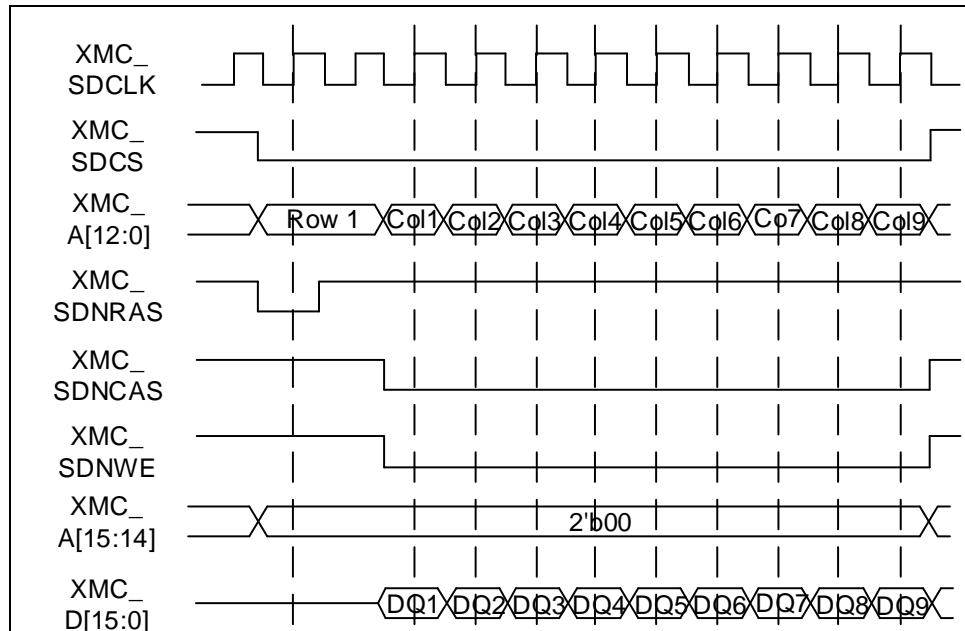
SDRAM 控制器写操作

对 SDRAM 执行写访问前必须将 SDRAM_CTRLx 中 WRP 位清 0，禁止对 SDRAM 的写保护。否则会发生 AHB 错误。

SDRAM 控制器会把接收的单次的或者突发的 AHB 写请求统一转换成单次的 SDRAM 写访问，这是由 BL=1 的设定决定的。控制器总是会尝试把 AHB 连续的或者不连续的写请求转换成连续的单次的 SDRAM 写操作以提高写访问的效率。

SDRAM 控制器始终会标记 SDRAM 各个 BANK 的有效行，以能够对不同的存储区域进行连续的写访问（多存储区域乒乓访问）。当控制器检测到当前接收到的下一个写访问发生在同一行或者其他 BANK 的有效行，则会直接执行写操作。如果下一个写访问针对的是一个无效行，SDRAM 控制器会生成预充电命令（该无效行所在的 BANK 有其它打开的行，如果没有则不需要预充电命令），然后激活要访问的新行并执行写操作。

图 25-19 SDRAM写访问波形 (Trcd=2, 某个写过程中 SDRAM端连续9次写)



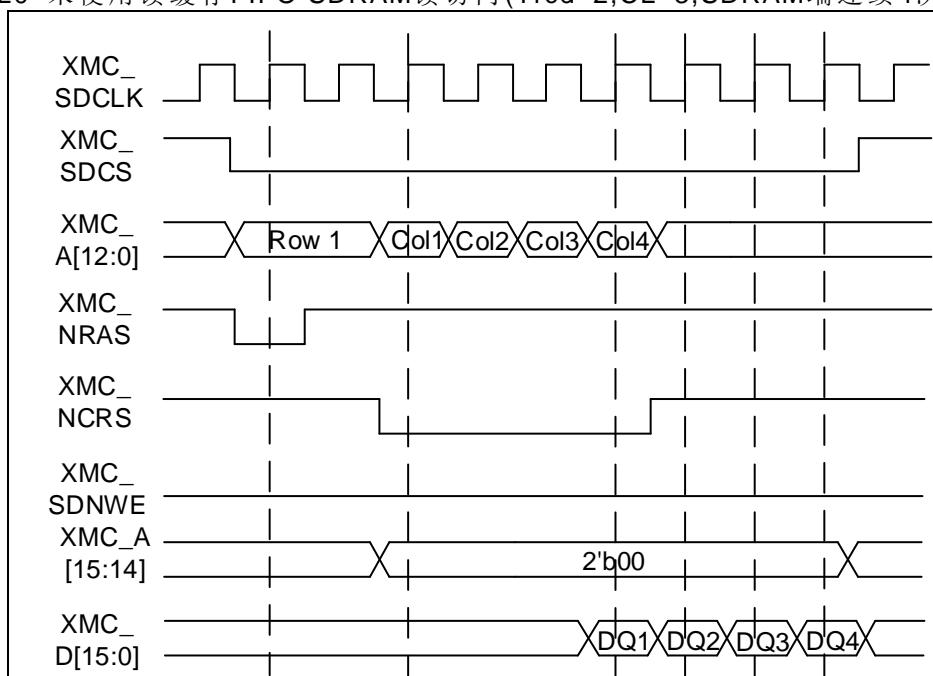
SDRAM 控制器读操作

SDRAM 控制器会把接收的单次的或者突发的 AHB 读请求统一转换成单次的 SDRAM 读访问，这是由 BL=1 的设定决定的。

SDRAM 控制器始终会标记 SDRAM 各个 BANK 的有效行，以能够对不同的存储区域进行连续的读访问（多存储区域乒乓访问）。当控制器检测到当前接收到的下一个读访问发生在同一行或者其他 BANK 的有效行，则会直接执行读操作。如果下一个读访问针对的是一个无效行，SDRAM 控制器会生成预充电命令（该无效行所在的 BANK 有其它打开的行，如果没有则不需要预充电命令），然后激活要访问的新行并执行读操作。

注意上述描述针对的是 SDRAM 控制器未打开连续读功能也即是未使用读缓存 FIFO 的情况。

图 25-20 未使用读缓存 FIFO SDRAM读访问 (Trcd=2, CL=3, SDRAM端连续4次读)



将 SDRAM_CTRL1 中位 BSTR 置 1 时，SDRAM 控制器会在当次读命令发完后紧接着继续发出读命令，多发的读命令的个数取决于 CAS 延迟和 RD 读延时。多读出的数据将会存放到可缓存的读 FIFO 中。每一层读 FIFO 都有相应的地址标记。多发的读命令的个数 M 遵循如下公式：

$$M = \text{CAS 周期} + \text{RD}/2 + 1$$

假如 CAS=3, RD= 3xHCLK，则 FIFO 中将存储 5 个未提交的数据。

每当出现读请求时，SDRAM 控制器将检查 FIFO 地址标记是否命中读请求要访问的地址，如果命中则直接从 FIFO 中取得需要的数据，如果没有命中则向存储器发送读命令，同时更新 FIFO。

行和 BANK 边界管理

当读写访问跨越了行边界时，如果下一个读写访问是连续的并且当前访问已执行到行边界，则 SDRAM 控制器会对当前有效行进行预充电以关闭当前行，然后激活要访问的新行，最后生成针对新的有效行的读写命令。

如果下一个访问是连续的并且当前访问跨越了 BANK 边界，可能存在以下两种情况：

- 如果当前 BANK 不是最后一个存储区域，则 SDRAM 会对下一个 BANK 进行访问。如果下一个 BANK 有有效行且有效行刚好是要访问的行，则直接发出读写命令；如果有有效行但是有效行不是要访问的行则关闭有效行同时激活新行；如果没有有效行则直接激活要访问的新行，然后发出读写命令。
- 如果当前 BANK 是最后一个存储区域，则直接生成 AHB 错误。

SDRAM 控制器自动刷新

由于 SDRAM 芯片本身需要不断刷新的特性，SDRAM 控制器会定期发送自动刷新命令。发送速率取决于 SDRAM_RCNT 寄存器中 RC 值。自动刷新命令的优先级高于读写访问，也即是说当自动刷新命令和读写访问命令同时发生时，读写访问会被暂时挂起，当控制器响应完刷新命令后再执行之前挂起的读写访问。

如果 SDRAM 存在有效活动行，则 SDRAM 控制器将生成全部预充电命令，然后再进行自动刷新。

如果在上一个自动刷新请求尚未完成的情况下又出现了新的自动刷新请求，则状态寄存器 SDRAM_STS 中的 ERR（刷新错误）位将置 1。如果刷新中断位 ERIEN 已配置位 1，将生成中断。

25.5.2 自刷新模式和掉电模式

SDRAM 控制器支持自刷新模式（self refresh）和掉电模式（power down）两种低功耗模式。

自刷新模式

通过将 CMD 配置为“101”并配置 SDRAM_CMD 寄存器中的目标存储区域位（BK1 和/或 BK2）来向 SDRAM 设备发出自刷新命令。

SDRAM 时钟在 TRAS 延迟后停止运行，假如两个 SDRAM 设备都已经初始化而且向两个设备都发出了自刷新命令或者向一个 SDRAM 设备发出了自刷新命令同时另一个设备未初始化则内部刷新定时器会停止计数。

进入自刷新模式后 SDRAM 设备处于此模式的最短时间为 TRAS，这是由 SDRAM 设备器件特性决定的。为了保证这一最短时间，在 SDRAM 刚进入自刷新模式时 BUSY 标志就会置 1，直到 TRAS 时间后，BUSY 才会被自动清 0，用户使用应用程序退出自刷新模式时可以通过此位进行判断。

用户可以对 SDRAM 发出正常模式命令（CMD 配置为 000）或者对 SDRAM 进行读写访问使 SDRAM 退出自刷新模式。

掉电模式

通过将 CMD 位置为“110”并配置 SDRAM_CMD 寄存器中的目标存储区域位（BK1 和/或 BK2）来向 SDRAM 发出掉电模式命令。

SDRAM 设备在掉电模式下也会正常执行自动刷新，当自动刷新命令产生时，SDRAM 会退出掉电模式然后执行自动刷新命令，执行完毕后，SDRAM 会再一次进入掉电模式。

用户可以对 SDRAM 发出正常模式命令（CMD 配置为 000）或者对 SDRAM 进行读写访问使 SDRAM 退出掉电模式。

25.6 XMC寄存器

必须以字（32位）的方式操作这些外设寄存器。

表 25-30 XMC寄存器地址映像

寄存器简称	基址偏移量	复位值
XMC_BK1CTRL1	0x000	0x0000 30DB
XMC_BK1TMG1	0x004	0xFFFF FFFF
XMC_BK1CTRL2	0x008	0x0000 30D2
XMC_BK1TMG2	0x00C	0xFFFF FFFF
XMC_BK1CTRL3	0x010	0x0000 30D2
XMC_BK1TMG3	0x014	0xFFFF FFFF
XMC_BK1CTRL4	0x018	0x0000 30D2
XMC_BK1TMG4	0x01C	0xFFFF FFFF
XMC_BK1TMGWR1	0x104	0xFFFF FFFF
XMC_BK1TMGWR2	0x10C	0xFFFF FFFF
XMC_BK1TMGWR3	0x114	0xFFFF FFFF
XMC_BK1TMGWR4	0x11C	0xFFFF FFFF
XMC_EXT1	0x220	0x0000 0808
XMC_EXT2	0x224	0x0000 0808
XMC_EXT3	0x228	0x0000 0808
XMC_EXT4	0x22C	0x0000 0808
SDRAM_CTRL1	0x140	0x0000 02D0
SDRAM_CTRL2	0x144	0x0000 02D0
SDRAM_TM1	0x148	0xFFFF FFFF
SDRAM_TM2	0x14C	0xFFFF FFFF
SDRAM_CMD	0x150	0x0000 0007
SDRAM_RCNT	0x154	0x0000 0000
SDRAM_STS	0x158	0x0000 0000
SDRAM_CLKDIV1	0x180	0x0000 0000

25.6.1 NOR闪存和PSRAM控制器寄存器

必须以字（32位）的方式操作这些外设寄存器。

25.6.1.1 SRAM/NOR闪存片选控制寄存器1 (XMC_BK1CTRL1)

访问：字访问

域	简称	复位值	类型	功能
位 31: 20	保留	0x000	resd	保持默认值。
位 19	MWMC	0x0	rw	对存储器写操作位 (Memory write mode control) 0: 写操作为异步模式; 1: 写操作为同步模式。
位 18: 16	CRPGS	0x0	rw	CRAM 页大小选择位 (CRAM page size) Cellular RAM 1.5 不允许跨页地址边界的同步访问。同步模式配置这些位时，遇到跨页时，XMC 会自动拆开访问。 000: 当跨页地址边界时不会拆开访问（默认值）; 001: 128 字节; 010: 256 字节; 011: 512 字节; 100: 1024 字节; 其他: 保留。
位 15	NWASEN	0x0	rw	异步传输期间等待信号使能位 (NWAIT in asynchronous transfer enable) 0: 禁止 NWAIT 信号; 1: 使能 NWAIT 信号。
位 14	RWTD	0x0	rw	读写时序不同控制位 (Read-write timing different) 读存储器与写存储器使用不同的时序进行操作，即寄存器 XMC_BK1TMGWR1 被开放。 0: 读写时序相同; 1: 读写时序不同。
位 13	NWSEN	0x1	rw	同步传输期间等待信号使能位 (NWAIT in synchronous transfer enable) 0: 禁用 NWAIT 信号; 1: 使能 NWAIT 信号。
位 12	WEN	0x1	rw	写使能位 (Write enable) 0: 禁止; 1: 使能。
位 11	NWTCFG	0x0	rw	等待时序配置 (NWAIT timing configuration) 仅在同步模式有效。 0: NWAIT 信号在等待状态前的一个数据周期有效; 1: NWAIT 信号在等待状态期间有效。
位 10	WRAPEN	0x0	rw	支持非对齐的成组模式 (Wrapped enable) XMC 于同步模式时是否支持将非对齐的 AHB 成组操作拆成 2 次操作; 0: 不允许直接的非对齐成组操作; 1: 允许直接的非对齐成组操作。
位 9	NWPOL	0x0	rw	等待信号极性 (NWAIT polarity) 在同步模式下，此位设置 NWAIT 信号极性。 0: 低有效; 1: 高有效。
位 8	SYNCBEN	0x0	rw	同步突发模式使能 (Synchronous burst enable) 允许对闪存存储器进行同步模式访问。 0: 禁用; 1: 使能。
位 7	保留	0x1	resd	保持默认值。
位 6	NOREN	0x1	rw	NOR 闪存访问使能 (Nor flash access enable) 0: 禁止 NOR 闪存的访问操作; 1: 使能 NOR 闪存的访问操作。
位 5: 4	EXTMDBW	0x1	rw	外部存储器数据宽度 (External memory data bus width)

				外部存储器的数据总线宽度。 00: 8 位; 01: 16 位; 10: 保留; 11: 保留。
位 3: 2	DEV	0x2	rw	存储器类型 (Memory device type) 00: SRAM/ROM; 01: PSRAM (Cellular RAM 或 CRAM); 10: NOR 闪存; 11: 保留。
位 1	ADMUXEN	0x1	rw	地址/数据复用使能位 (Address/data multiplexing enable) 0: 地址/数据不复用; 1: 地址/数据复用数据总线。
位 0	EN	0x1	rw	存储器块使能位 (Memory bank enable) 0: 禁用存储器块; 1: 启用存储器块。

25.6.1.2 SRAM/NOR闪存片选控制寄存器x (XMC_BK1CTRLx)

x=2,3,4

访问: 字访问

域	简称	复位值	类型	功能
位 31: 20	保留	0x000	resd	保持默认值。
位 19	MWMC	0x0	rw	对存储器写操作位 (Memory write mode control) 0: 写操作为异步模式; 1: 写操作为同步模式。
位 18: 16	CRPGS	0x0	rw	CRAM 页大小选择位 (CRAM page size) Cellular RAM 1.5 不允许跨页地址边界的同步访问。同步模式配置这些位时, 遇到跨页时, XMC 会自动拆开访问。 000: 当跨页地址边界时不会拆开访问 (默认值); 001: 128 字节; 010: 256 字节; 011: 512 字节; 100: 1024 字节; 其他: 保留。
位 15	NWASEN	0x0	rw	异步传输期间等待信号使能位 (NWAIT in asynchronous transfer enable) 0: 禁止 NWAIT 信号; 1: 使能 NWAIT 信号。
位 14	RWTD	0x0	rw	读写时序不同控制位 (Read-write timing different) 读存储器与写存储器使用不同的时序进行操作, 即寄存器 XMC_BK1TMGWRx 被开放。 0: 读写时序相同; 1: 读写时序不同。
位 13	NWSEN	0x1	rw	同步传输期间等待信号使能位 (NWAIT in synchronous transfer enable) 0: 禁用 NWAIT 信号; 1: 使能 NWAIT 信号。
位 12	WEN	0x1	rw	写使能位 (Write enable) 0: 禁止; 1: 使能。
位 11	NWTCFG	0x0	rw	等待时序配置 (NWAIT timing configuration) 仅在同步模式有效。 0: NWAIT 信号在等待状态前的一个数据周期有效; 1: NWAIT 信号在等待状态期间有效。
位 10	WRAPEN	0x0	rw	支持非对齐的成组模式 (Wrapped enable)

				XMC 于同步模式时是否支持将非对齐的 AHB 成组操作拆成 2 次操作； 0: 不允许直接的非对齐成组操作； 1: 允许直接的非对齐成组操作。
位 9	NWPOL	0x0	rw	等待信号极性 (NWAIT polarity) 在同步模式下，此位设置 NWAIT 信号极性。 0: 低有效； 1: 高有效。
位 8	SYNCBEN	0x0	rw	同步突发模式使能 (Synchronous burst enable) 允许对闪存存储器进行同步模式访问。 0: 禁用； 1: 使能。
位 7	保留	0x1	rstd	保持默认值。
位 6	NOREN	0x1	rw	NOR 闪存访问使能 (Nor flash access enable) 0: 禁止 NOR 闪存的访问操作； 1: 使能 NOR 闪存的访问操作。
位 5: 4	EXTMDBW	0x1	rw	外部存储器数据宽度 (External memory data bus width) 外部存储器的数据总线宽度。 00: 8 位； 01: 16 位； 10: 保留； 11: 保留。
位 3: 2	DEV	0x0	rw	存储器类型 (Memory device type) 00: SRAM/ROM； 01: PSRAM (Cellular RAM 或 CRAM)； 10: NOR 闪存； 11: 保留。
位 1	ADMUXEN	0x1	rw	地址/数据复用使能 (Address/data multiplexing enable) 0: 地址/数据不复用； 1: 地址/数据复用数据总线。
位 0	EN	0x0	rw	存储器块使能位 (Memory bank enable) 0: 禁用存储器块； 1: 启用存储器块。

25.6.1.3 SRAM/NOR闪存片选时序寄存器x (XMC_BK1TMGx)

x=1,2,3,4

访问：字访问

域	简称	复位值	类型	功能
位 31: 30	保留	0x0	rstd	保持默认值。
位 29: 28	ASYNCM	0x0	rw	异步访问模式选择位 (Asynchronous mode) 只在 RWTD 位使能时有效。 00: 模式 A； 01: 模式 B； 10: 模式 C； 11: 模式 D。
位 27: 24	DTLAT	0xF	rw	数据延迟 (Data latency) 仅在同步模式有效。 0000: 额外插入 1 个 XMC_CLK 周期； 0001: 额外插入 2 个 XMC_CLK 周期； 1111: 额外插入 16 个 XMC_CLK 周期。
位 23: 20	CLKPSC	0xF	rw	时钟分频系数 (Clock prescale) 仅在同步模式有效，定义 XMC_CLK 时钟的频率。 0000: 保留； 0001: XMC_CLK 周期为 HCLK 周期的 2 倍； 0010: XMC_CLK 周期为 HCLK 周期的 3 倍； 1111: XMC_CLK 周期为 HCLK 周期的 16 倍。
位 19: 16	BUSLAT	0xF	rw	总线延迟时间 (Bus latency)

				为了防止数据总线发生冲突，在复用模式或同步模式时，如果一次读操作之后紧接着写操作 XMC 将在数据总线上插入延迟。
				0000: 插入 1 个 HCLK 周期； 0001: 插入 2 个 HCLK 周期； 1111: 插入 16 个 HCLK 周期。
				数据建立时间 (Data setup time) 00000000: 额外插入 1 个 HCLK 周期； 00000001: 额外插入 2 个 HCLK 周期； 11111111: 额外插入 256 个 HCLK 周期。
位 15: 8	DTST	0xFF	rw	地址保持时间 (Address-hold time) 0000: 额外插入 1 个 HCLK 周期； 0001: 额外插入 2 个 HCLK 周期； 1111: 额外插入 16 个 HCLK 周期。
位 7: 4	ADDRHT	0xF	rw	地址建立时间 (Address setup time) 0000: 额外插入 1 个 HCLK 周期； 0001: 额外插入 2 个 HCLK 周期； 1111: 额外插入 16 个 HCLK 周期。
位 3: 0	ADDRST	0xF	rw	0000: 额外插入 1 个 HCLK 周期； 0001: 额外插入 2 个 HCLK 周期； 1111: 额外插入 16 个 HCLK 周期。

25.6.1.4 SRAM/NOR闪存写时序寄存器x (XMC_BK1TMGWRx)

x=1,2,3,4

访问：字访问

域	简称	复位值	类型	功能
位 31: 30	保留	0x0	resd	保持默认值。
位 29: 28	ASYNCM	0x0	rw	异步访问模式选择位 (Asynchronous mode) 只在 RWTD 位使能时有效。 00: 模式 A; 01: 模式 B; 10: 模式 C; 11: 模式 D。
位 27: 20	保留	0xFF	resd	保持默认值。
位 19: 16	BUSLAT	0xF	rw	总线延迟时间 (Bus latency) 为了防止数据总线发生冲突，在复用模式或同步模式时，如果一次写操作之后紧跟着读操作 XMC 将在数据总线上插入延迟。 0000: 插入 1 个 HCLK 周期； 0001: 插入 2 个 HCLK 周期； 1111: 插入 16 个 HCLK 周期。
位 15: 8	DTST	0xFF	rw	数据建立时间 (Data setup time) 00000000: 额外插入 1 个 HCLK 周期； 00000001: 额外插入 2 个 HCLK 周期； 11111111: 额外插入 256 个 HCLK 周期。
位 7: 4	ADDRHT	0xF	rw	地址保持时间 (Address-hold time) 0000: 额外插入 1 个 HCLK 周期； 0001: 额外插入 2 个 HCLK 周期； 1111: 额外插入 16 个 HCLK 周期。
位 3: 0	ADDRST	0xF	rw	地址建立时间 (Address setup time) 0000: 额外插入 1 个 HCLK 周期； 0001: 额外插入 2 个 HCLK 周期； 1111: 额外插入 16 个 HCLK 周期。

25.6.1.5 SRAM/NOR额外时序寄存器x (XMC_EXTx) x=1,2,3,4

访问：字访问

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	保持默认值。
位 15: 8	BUSLATR2R	0x08	rw	<p>连续读操作恢复时间 (Bus turnaround phase for consecutive read duration) 用于定义连续读操作间总线恢复时间，为了避免总线冲突，在连续的两次读之间将插入延迟时间。 00000000: 连续读操作插入 1 个 HCLK 周期； 00000001: 连续读操作插入 2 个 HCLK 周期； 00001000: 连续读操作插入 9 个 HCLK 周期（默认值）； 11111111: 连续读操作插入 256 个 HCLK 周期。</p>
位 7: 0	BUSLATW2W	0x08	rw	<p>连续写操作恢复时间 (Bus turnaround phase for consecutive write duration) 用于定义连续写操作间总线恢复时间。为了避免总线冲突，在连续的两次写之间将插入延迟时间。 00000000: 连续写操作插入 1 个 HCLK 周期； 00000001: 连续写操作插入 2 个 HCLK 周期； 00001000: 连续写操作插入 9 个 HCLK 周期（默认值）； 11111111: 连续写操作插入 256 个 HCLK 周期。</p>

25.6.2 SDRAM控制器寄存器

25.6.2.1 SDRAM控制寄存器 1, 2 (SDRAM_CTRL1, SDRAM_CTRL2)

此寄存器包含每个 SDRAM 存储器的容量设置以及访问方式配置信息。

域	简称	复位值	类型	功能
位 31: 15	保留	0x000000	resd	必须保持复位时的值。
位 14: 13	RD	0x0	rw	<p>读延时 (Read Delay) 定义在 CAS 延迟后延后多少个 HCLK 时钟周期读取数据 00: 0 个 HCLK 延迟 01: 1 个 HCLK 延迟 10: 2 个 HCLK 延迟 11: 保留，不使用 注意：SDRAM_CTRL2 寄存器中的相应位为无关位。</p>
位 12	BSTR	0x0	rw	<p>连续读 (Burst read) 设置此位为 1，表示将单次 AHB 请求（单次的或者突发的）都作为连续读处理，会预取几笔数据到缓存 FIFO。 0: 不将单次读请求作为连续读处理 1: 始终将单次读请求作为连续读处理 注意：SDRAM_CTRL2 寄存器中的相应位为无关位。</p>
位 11: 10	CLKDIV	0x0	rw	<p>时钟分频配置 (clock division configuration) SDRAM 时钟分频配置。 00: 禁止 SDCLK 时钟 01: 4 分频 HCLK 10: 2 分频 HCLK 11: 3 分频 HCLK 注意：SDRAM_CTRL2 寄存器中的相应位为无关位。 注意：当 CLKDIV1 =1 时 CLKDIV 须配置为 0x0，这时 SDCLK 为 HCLK 的反向。</p>
位 9	WRP	0x0	rw	<p>写保护 (Write protection) 设置此位，使能 SDRAM 写保护。 0: 可以写 1: 禁止写</p>

位 8: 7	CAS	0x0	rw	列地址选通延迟 (CAS Latency) 选择列地址选通延迟。 00: 保留, 不使用。 01: 1 个周期 10: 2 个周期 11: 3 个周期
位 6	INBK	0x0	rw	内部区块 (internal banks) 定义当前 SDRAM 内部 BANK 数目。 0: 2 个内部 BANK 1: 4 个内部 BANK
位 5: 4	DB	0x0	rw	SDRAM 数据总线宽度 (Data Bus)。 可以外接 8 位和 16 位数据总线宽度 SDRAM 设备。 00: 8 位 01: 16 位 10: 保留, 不使用。 11: 保留, 不使用。
位 3: 2	RA	0x0	rw	行地址 (Row address) 行地址位数, 支持 11,12,13 位行地址。 00: 11 位 01: 12 位 10: 13 位 11: 保留, 不使用。
位 1: 0	CA	0x0	rw	列地址 (Column Address) 列地址位数, 支持 8,9,10,11 位列地址 00: 8 位 01: 9 位 10: 10 位 11: 11 位

25.6.2.2 SDRAM时序寄存器 1, 2 (SDRAM_TM1, SDRAM_TM2)

此寄存器包含每个 SDRAM 存储区域的时序参数

域	简称	复位值	类型	功能
位 31: 28	保留	0x0	resd	必须保持复位时的值。
位 27: 24	TRCD	0xF	rw	行激活到列延迟 (Row active to Read/Write delay) 这些位定义激活命令到读/写命令之间的延迟。 0000: 1 个周期 0001: 2 个周期 1111: 16 个周期
位 23: 20	TRP	0xF	rw	预充电到激活延迟 (precharge to active delay) 这些位定义预充电命令与其它命令之间的延迟。 0000: 1 个周期 0001: 2 个周期 1111: 16 个周期
位 19: 16	TWR	0xF	rw	写命令到预充电延迟 (Write Recovery delay) 这些位定义写命令到预充电命令之间的延迟。 0000: 1 个周期 0001: 2 个周期 1111: 16 个周期
位 15: 12	TRC	0xF	rw	刷新命令到激活命令延迟 (refresh to active delay) 这些位定义刷新命令和激活命令之间的延迟, 以及两个相邻刷新命令之间的延迟。 0000: 1 个周期 0001: 2 个周期 1111: 16 个周期
位 11: 8	TRAS	0xF	rw	自刷新周期 (Self refresh)

				这些位定义 SDRAM 在自刷新模式下停留的最短时间。 0000: 1 个周期 0001: 2 个周期 1111: 16 个周期
位 7: 4	TXSR	0xF	rw	退出自刷新延迟 (Exit Self-refresh to active delay) 这些位定义从退出自刷新命令到发出激活命令之间的延迟。 0000: 1 个周期 0001: 2 个周期 1111: 16 个周期
位 3: 0	TMRD	0xF	rw	加载模式寄存器到激活 (Mode register program to active delay) 这些位定义加载模式寄存器命令和激活或刷新命令之间的延迟，按存储器时钟周期计。 0000: 1 个周期 0001: 2 个周期 1111: 16 个周期

注意：如果使用了两个 SDRAM 设备，则必须使用最慢设备的时序配置 TRP 和 TRC 参数。

25.6.2.3 SDRAM命令寄存器 (SDRAM_CMD)

域	简称	复位值	类型	功能
位 31: 22	保留	0x000	resd	必须保持复位时的值。
位 22: 9	MRD	0x0000	rw	模式寄存器数据 (Mode Register data) 模式寄存器各 bit 定义请参考 SDRAM 规范。
位 8: 5	ART	0x0	rw	自刷新次数 (Auto-refresh times) 这些位定义初始化时 MODE = “011” 时所发出的连续自刷新命令个数。 0000: 1 个自刷新 0001: 2 个自刷新 1110: 15 个自刷新 1111: 保留
位 4	BK1	0x0	wo	命令目标存储区域 1 (SDRAM Bank 1) 该位指示是否向 SDRAM 设备 1 发送命令。 0: 不发送命令到 SDRAM 设备 1 1: 发送命令到 SDRAM 设备 1
位 3	BK2	0x0	wo	区块 2 (SDRAM Bank 2) 该位指示是否向 SDRAM 设备 2 发送命令。 0: 不发送命令到 SDRAM 设备 2 1: 发送命令到 SDRAM 设备 2
位 2: 0	CMD	0x0	wo	SDRM 命令 (SDRAM Command) 这些位定义发送到 SDRAM 设备的命令。 000: 正常模式 001: 时钟配置使能 010: 预充电所有存储区域 011: 自动刷新命令 (auto refresh) 100: 加载模式寄存器 101: 自刷新命令 (self refresh) 110: 掉电命令 111: 保留

25.6.2.4 SDRAM刷新定时器寄存器 (SDRAM_RCNT)

该寄存器用来配置 SDRAM 颗粒的刷新速率，计数单位是一个 SDRAM CLK 时钟周期。

必须把 RC 配置为非 0 值，才能执行正确的刷新操作，在初始化后不可修改 RC 值。

刷新操作的优先级高于读写操作，但是如果刷新请求产生的时候正在进行读写操作，则会等到读写操作结束后再执行刷新操作，为了确保预期的刷新速率，建议

RC 的值设置的比计算的值少一些。

此寄存器为 SDRAM 存储区域 1 和存储区域 2 所共用。

域	简称	复位值	类型	功能
位 31: 15	保留	0x000000	resd	必须保持复位时的值。
位 14	ERIEN	0x0	rw	错误中断使能 (error Interrupt Enable) 0: 禁止中断 1: 使能中断
位 13: 1	RC	0x0000	rw	刷新计数器 (Refresh Counter) 该 13 位字段定义 SDRAM 设备的刷新速率，以存储器时钟周期数表示。该字段必须设置为至少 41 个 SDRAM 时钟周期。 刷新速率 = (RC + 1) x SDRAM 频率时钟 RC = (SDRAM 刷新周期/行数) - 20
位 0	ERRC	0x0	wo	清除错误标志 (Error flag clear) 该位用于清除状态寄存器中的错误标志 (ER)。 0: 无操作 1: 清除错误标志

注意： 所编程的 COUNT 值不可等于以下时序之和： TWR+TRP+TRC+TRCD+4 个存储器时钟周期。

25.6.2.5 SDRAM状态寄存器 (SDRAM_STS)

域	简称	复位值	类型	功能
位 31: 6	保留	0x00000000	resd	必须保持复位时的值。
位 5	BUSY	0x0	ro	忙状态 (Busy) SDRAM 控制器当前的状态 0: 空闲 1: 忙状态
位 4: 3	BK2STS	0x0	ro	存储区域 2 的状态 (Bank2 Status) 该位定义 SDRAM 存储区域 2 的状态模式。 00: 正常模式 01: 自刷新模式 10: 掉电模式
位 2: 1	BK1STS	0x0	ro	存储区域 1 的状态 (Bank 1 Status) 该位定义 SDRAM 存储区域 1 的状态模式。 00: 正常模式 01: 自刷新模式 10: 掉电模式
位 0	ERR	0x0	ro	刷新错误标志 (error flag) 0: 未检测到错误 1: 检测到错误

25.6.2.6 SDRAM时钟一分频 (SDRAM_CLKDIV1)

域	简称	复位值	类型	功能
位 31: 5	保留	0x00000000	resd	必须保持复位时的值。
位 4	CLKDIV1	0x0	rw	时钟一分频 (clock one division) SDRAM 时钟一分频。 0: 禁止 SDRAM 时钟一分频； 1: 使能 SDRAM 时钟一分频。 注意：当 CLKDIV1 =1 时 CLKDIV 须配置为 0x0，这时 SDCLK 为 HCLK 的反向。
位 3: 0	保留	0x0	resd	必须保持复位时的值。

26 SDIO 接口

26.1 简介

SD/SDIO MMC 卡主机模块（SDIO）在 AHB 外设总线和多媒体卡（MMC）、SD 存储卡、SDIO 卡间提供了操作接口。

SD 储存卡和 SDI/O 卡的系统规格书可以通过 SD 卡协议网站(www.sdcards.org)

多媒体卡系统规格书由 MMCA 技术委员会发布，可以在多媒体卡协会的网站上(www.mmca.org)获得。

26.2 主要特点

- 与 SD 储存卡 2.0 规格版本全兼容
- 与 SDI/O 卡 2.0 规格版本全兼容并支持 1 位和 4 位数据总线模式
- 与多媒体卡 4.2 规格版本全兼容并支持 1 位、4 位和 8 位数据总线模式
- 与较早的多媒体卡规格版本全兼容
- 支持 DMA 传输
- 8 位总线模式下数据传输速率可达 48 MHz
- 中断请求

注意：SDIO 并不兼容 SPI 的通信模式，并且在同一时间内只能支持一个 SD/SDIO/MMC 4.2 卡

总线上的通信是通过传送命令和数据实现。

- 命令：命令是启动操作的令牌。命令从主机发送到单个卡（寻址命令）或所有连接的卡（广播命令），命令在CMD总线串行传输。
- 响应：响应是从卡发送到主机，作为对先前命令的答复，响应在CMD总线串行传输
- 数据：数据可以从卡传送到主机或是主机传送至卡端，数据通过SDIO_D数据总线进行传递

MMC 卡/SD 卡/SDI/O 卡在总线上的基本操作是命令/响应结构，这样的总线操作在命令或总线机制下实现信息交换；另外，某些操作还具有数据令牌。

在 SD/SD IO 存储器卡上传送的数据是以数据块的形式传输，数据块总是以 CRC 位为后，定义了单个和多个块操作，在 MMC 上传送的数据是以数据块或数据流的形式传输，详细可参考下列图示。

图 26-1 SDIO“无响应”和“有响应”操作

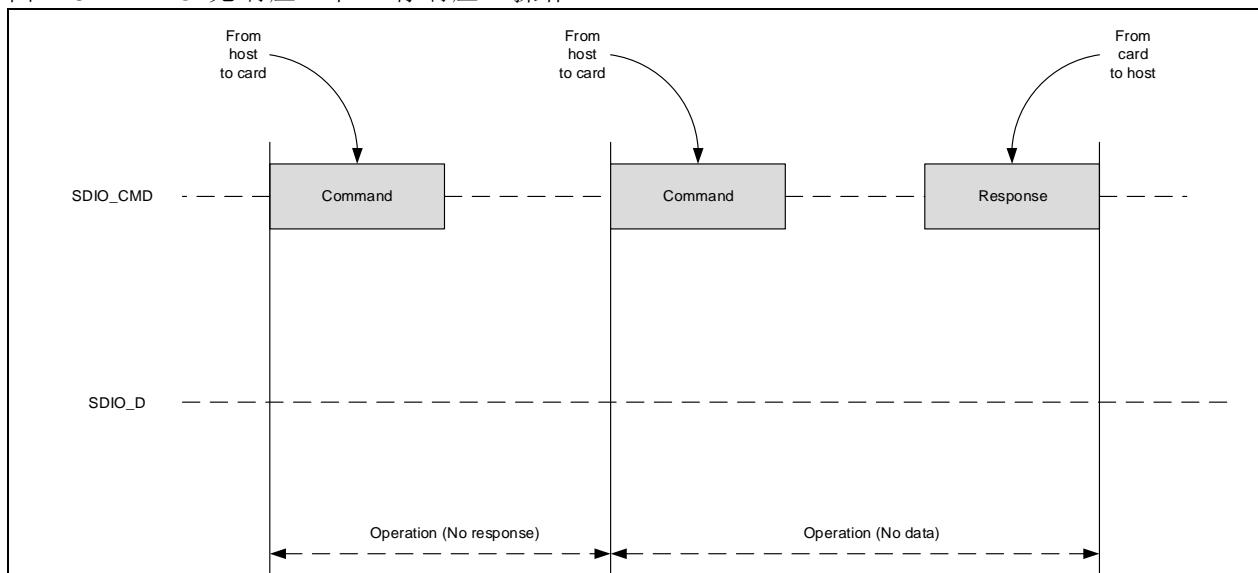


图 26-2 SDIO (多) 数据块读操作

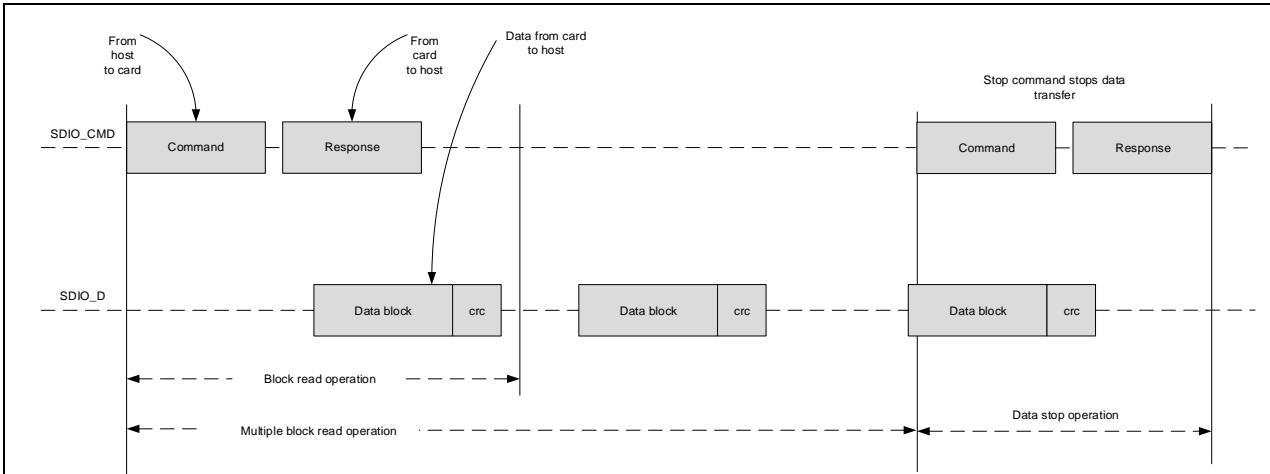
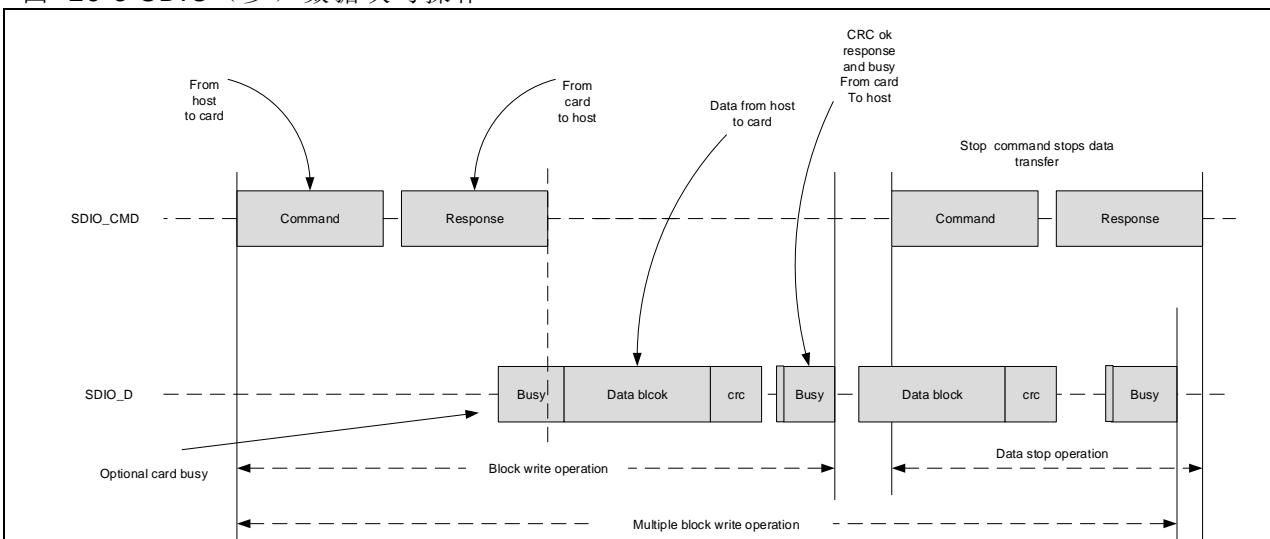


图 26-3 SDIO (多) 数据块写操作



注意：当有 *Busy* (繁忙) 信号时，SDIO (SDIO_D0 被拉低) 将不会发送任何数据。

图 26-4 SDIO MMC 卡数据流读操作

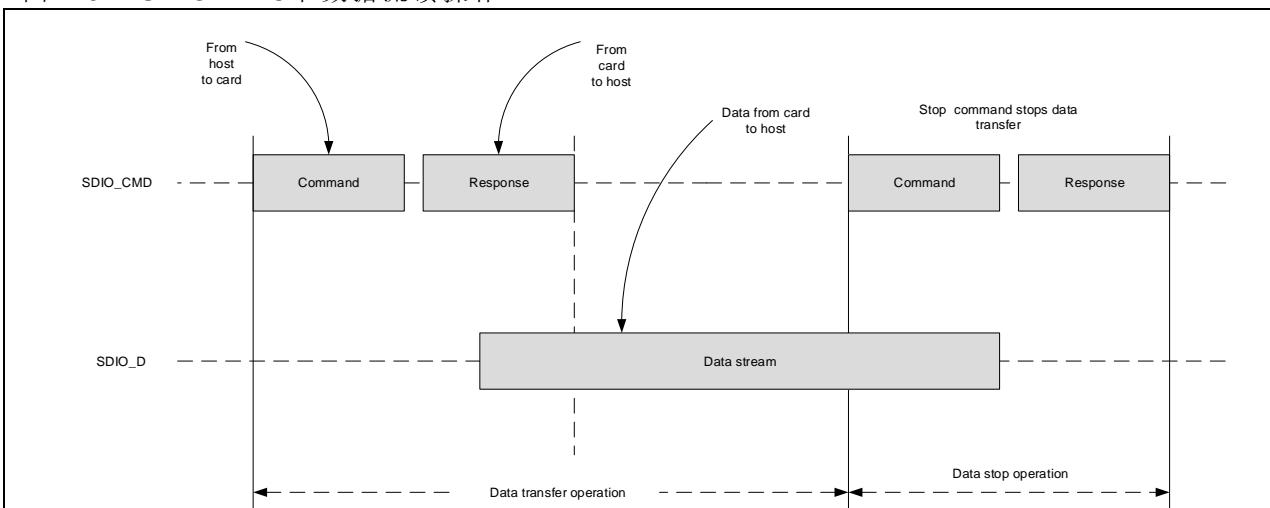
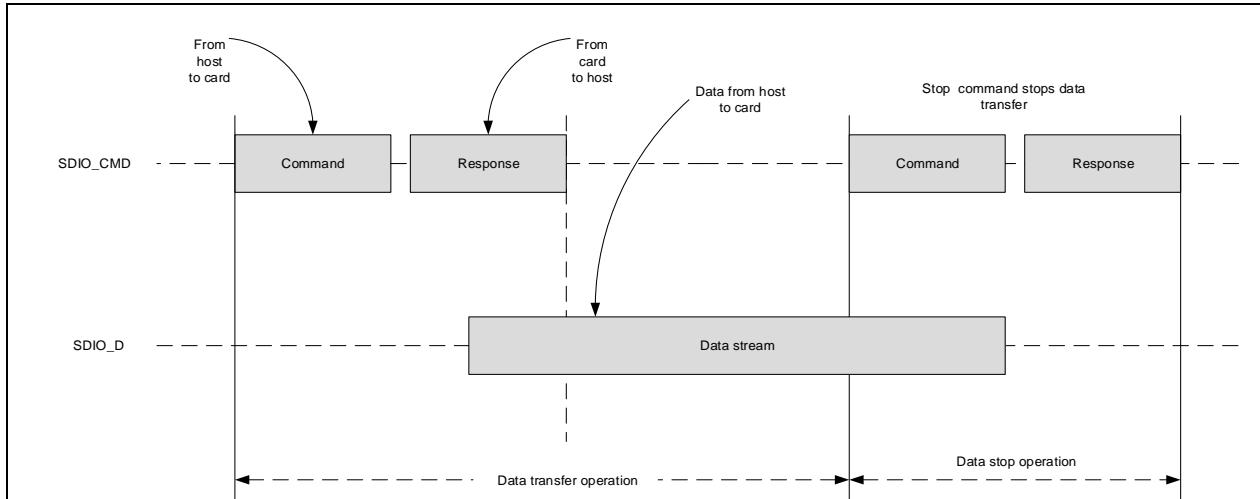


图 26-5 SDIO MMC卡数据流写操作



26.3 功能描述

26.3.1 卡功能描述

主机与卡之间的所有通信都由卡来控制，主机发送两种类型的命令：广播和寻址（点对点）命令。

- 广播命令：适用于所有卡，其中一些命令需要响应
- 寻址命令：发送到寻址的卡，并引起卡的响应

存储卡定义了两种操作模式：

- 卡识别模式
- 数据传输模式

26.3.1.1 卡识别模式

在卡识别模式下，主机会重置处于卡识别模式的所有卡和检测工作电压范围，识别卡并要求它们发布相对卡地址(RCA)，对每个卡在其自己的 CMD 在线分别执行此操作，在卡识别模式下的所有通信都使用命令信号线(CMD)。

卡识别过程

不同的卡有不同的识别过程，主机需要发送不同的命令，卡的类型可以分为 SD 卡、SDI/O 卡和 MMC 卡，要识别卡的类型可以发送 CMD5 命令，如果主机接收到响应，卡的类型就是 SDI/O 卡，若没有响应那接着发送 ACMD41 命令，如果主机接收到响应就是 SD 卡，否则就是 MMC 卡。

以下描述卡的识别过程：

1. 总线被激活，检测卡是否连接，卡识别过程中时钟频率为0-400kHz。
2. SDIO主机发送命令识别卡的类型是SD卡、SDI/O卡或是MMC卡。
3. 根据卡的类型进行初始化
 - SD卡：SDIO主机发送CMD2(ALL_SEND_CID)，以获得其唯一的卡标志(CID号)，卡发送CID号作为响应后主机发送CMD3(SEND_RELATIVE_ADDR)要求卡发布新的相对卡地址(RCA)，该地址比CID短并且用于之后的数据传输模式下寻址卡。
 - SDI/O卡：SDIO主机发送CMD3(SEND_RELATIVE_ADDR)要求卡发布新的相对卡地址(RCA)，该地址比CID短并且用于之后的数据传输模式下寻址卡。
 - MMC卡：SDIO主机发送CMD1(SEND_OP_COND)，接着发送CMD2和CMD3。
4. 如果主机要分配另一个RCA号码，则可以通过向该卡发送另一个CMD3命令来要求该卡发布新的号码，最后发布的RCA是卡的实际RCA编号，主机重复识别过程即系统中每个卡的CMD2和CMD3循环。

26.3.1.2 数据传输模式

主机在识别在总线的所有卡后将进入数据传输模式，在数据传输模式下主机可以在 0 - 48MHz 频率范围内操作卡，主机可以发出 CMD9 (SEND_CSD) 以获取卡特定数据(CSD 寄存器)，例如块长度和卡储存容量等。在数据传输模式下的所有数据通信都是主机与所选卡之间的点对点传输，CMD 总在线的响应会确认所有以寻址的命令，数据传输读写可分为数据块模式和数据流模式，可以在 SDIO 数据控制寄存器 (SDIO_DTCTRL) 的 TFRMODE 位做设置，在数据流模式，数据按字节传输，同时每个数据块后没有 CRC。

宽总线选择/解除

对于 SD 卡可以使用 ACMD6(SET_BUS_WIDTH) 命令选择或解除宽总线(4 位总线宽度)操作模式，上电或 CMD0(GO_IDLE_STATE) 后默认总线宽度为 1 位，ACMD6 命令仅在传输状态时有效也就是在经过 CMD7 选择卡之后才可以改变总线宽度。

数据流读写(只适用于多媒体卡)

读取：

1. 主机发送过 CMD11(READ_DAT_UNTIL_STOP) 进行数据流读取。
2. 直到主机发送 CMD12 (STOP_TRANSMISSION)，由于串行命令的发送，停止命令具有执行延迟，在停止命令的结束位后数据传输停止。

写入：

1. 主机发送 CMD20 (WRITE_DAT_UNTIL_STOP) 进行数据流写入。
2. 直到主机发送 CMD12(STOP_TRANSMISSION)，由于未预先确定要传输的数据量，因此无法使用 CRC，如果主机提供超出范围的地址作为 CMD20 的参数，则卡将拒绝该命令，保持在传输状态，并通过将 ADDRESS_OUT_OF_RANGE 位置 1 进行响应。

数据块读取

在数据块读取的模式下，数据传输的基本单位是块，最大块大小在 CSD(READ_BL_LEN) 定义，其最大大小始终为 512 字节，如果设置了 READ_BL_PARTIAL，可以发送其起始和结束地址完全包含在 512 字节边界内较小的数据块，CRC 会附加到每个块的末尾用以确保数据传输的正确，数据块读取有几个相关的命令操作如下：

- CMD17 (READ_SINGLE_BLOCK)：启动数据块读取，完成传输后卡返回到传输状态。
 - CMD18 (READ_MULTIPLE_BLOCK)：开始传输几个连续的数据块。
- 数据块将连续传输直到主机发出 CMD12(STOP_TRANSMISSION)，由于串行命令的发送，停止命令具有执行延迟，在停止命令的结束位后数据传输停止。

数据块写入

在执行数据块写入命令(CMD24-27)时，一个或多个数据块从主机传输到卡，CRC 会附加到每个数据块的末尾，如果 CRC 检测失败，卡通过 SDIO_D 信号线指示错误，传送数据被丢弃而不写入，并且发送的数据块将被忽略。

如果主机使用的部分块的累积长度未对齐，并且不允许块未对齐(未设置 CSD 参数 WRITE_BLK_MISALIGN)，则卡应检测到块未对齐错误，并在第一个未对齐块开始之前中止编程。卡片应当在 SDIO 状态寄存器 (SDIO_STS) 中设置 ADDRESS_ERROR 错误位，并且在忽略所有进一步的数据传输的同时，在接收数据状态中等待停止命令，如果主机试图在写保护区域上进行写操作，则写操作也应中止。但是，在这种情况下，卡应将 WP_VIOLATION 位置 1。

设置 CID 和 CSD 寄存器不需要事先设置块长度，传送的数据也受 CRC 保护的，如果 CSD 或 CID 寄存器的部分是存储在 ROM 中，则该不可更改的部分应与接收缓冲区的部分匹配，若匹配失败，则卡将报告错误并且不会更改任何寄存器的内容，某些卡可能需要很长且不可预测的时间来写入数据块。接收到数据块并完成 CRC 检查后，如果卡的写缓冲区已满并且无法从新的 WRITE_BLOCK 命令接受新数据，则该卡将开始写并保持 SDIO_D 信号线为低电平，主机可以随时使用 SEND_STATUS 命令 (CMD13) 查询卡的状态，卡将以其状态进行响应。状态位 READY_FOR_DATA 指示卡是否可以接受新数据或写入过程是否仍在进行中，主机可以通过发出 CMD7 (选择另一张卡) 来取消选择卡，这将使卡进入断开状态并释放 SDIO_D 信号线而不会中断写入操作。重新选择卡时，如果编程仍在进行且写缓冲区不可用，它将通过将 SDIO_D 信号线拉至低电平来重新激活忙碌指示。

26.3.1.3 擦除

多媒体卡和 SD 卡的擦除单位是擦除组，以写数据块计算，写数据块是卡的基本写入单位，擦除组的大小是卡的特定参数，在 CSD 中定义。

主机能擦除一个连续范围的擦除组，开始擦除操作有三个步骤，而多媒体卡和 SD 卡发送的命令有所不同。

1. 主机发送命令定义连续范围的开始地址
 - SD卡：发送CMD32 (ERASE_WR_BLK_START)
 - MMC卡：发送CMD35 (ERASE_GROUP_START)
2. 主机发送命令定义连续范围的结束地址
 - SD卡：发送CMD33(ERASE_WR_BLK_END)
 - MMD卡：发送CMD36(ERASE_GROUP_END)
3. 主机发送擦除命令CMD38(ERASE)，开始擦除操作

26.3.1.4 保护管理

SDIO 卡主机模块支持三种保护方式，使主机保护数据不被擦除或改写，如下所示：

机械写保护开关

在卡的侧边有一个机械的滑动开关，使用户设置是否对卡进行写保护，如果滑动平板计算机以窗口打开的方式放置，则表示卡已被写保护。如果窗口关闭，则该卡不受写保护。

卡的内部写保护

卡数据可以受到保护，不被擦除或写入。通过设置 CSD 中的永久或临时写保护位，制造商或内容提供商可以对整个卡进行永久性写保护。支持通过设置扇区组写保护的卡可以设置 CSD 中的 WP_GRP_ENABLE 位以可保护部分数据，并且写保护可由应用程序更改。SET_WRITE_PROT 命令设置寻址的写保护组的写保护，CLR_WRITE_PROT 命令清除寻址的写保护组的写保护。

SEND_WRITE_PROT 命令类似于单个块读取命令。卡应发送一个数据块，该数据块包含 32 个写保护位（代表从指定地址开始的 32 个写保护组），后跟 16 个 CRC 位。写保护命令中的地址域是一个字节为单位的组地址。

密码保护卡锁定

SDIO 卡主机可以使用密码保护功能对卡锁定或解锁，密码储存在 128 位的 PWD 寄存器中，密码长度设置储存在 8 位的 PWD_LEN 寄存器中，这些寄存器是非挥发性的，掉电后不会清除寄存器的内容。

已锁定的卡能够支持基本的命令，主机可以对卡进行复位、初始化和状态查询等操作，但无法获取卡中的数据，当设置了密码后(PWD_LEN 不为 0)，上电后卡自动锁定。

与 CSD 和 CID 寄存器写入命令相似，锁定/解锁命令仅在传输状态下有效，锁定/解锁命令不包含地址参数所以在使用前卡必须要被选中。

卡的锁定/解锁命令具有单数据块写命令的结构和总线操作类型，传输的数据块包含所有命令所需要的信息（密码设置模式、PWD 内容和上锁/解锁指示）。在发送卡的锁定/解锁命令之前，命令数据块的长度由 SDIO 卡主机模块定义，锁定/解锁命令的结构如下表：

表 26-1 锁定/解锁命令的结构

Byte	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
0				保留(需设为 0)	ERASE	LOCK_UNLOCK	CLR_PWD	SET_PWD
1					PWDS_LEN			
2								
...					密码数据			
PWDS_LEN+1								

- ERASE：将该位置 1 将执行强制擦除，所有其它位必须为 0，只发送命令字节
- LOCK_UNLOCK：将该位置 1 锁住卡，置 0 解锁卡，LOCK_UNLOCK 与 SET_PWD 可以同时设置，但不能与 CLR_PWD 同时设置
- CLR_PWD：将该位置 1 清除密码数据
- SET_PWD：将该位置 1 将密码数据保存至存储器
- PWD_LEN：以字节为单位定义密码的长度，在改变密码的情况下，长度应该是新旧密码长度和

- PWD: 密码（依不同的命令，新的密码或正在使用的密码）

数据块大小应由主机在发送卡锁定/解锁命令之前定义。块长度应设置为大于或等于锁定/解锁命令所需的数据结构。

以下几节列出了设置/清除密码、上锁/解锁和强制擦除的命令序列。

设置密码

1. 如果先前未选择卡，先使用CMD7（SELECT/DESELECT_CARD）选择一个卡
2. 使用CMD16(SET_BLOCKLEN)定义要在8位的卡锁定/解锁模式下发送的数据块长度，8位的PWD_LEN，新密码的字节数目。当更换了密码后，发送命令的数据块长度必须同时考虑新旧密码的长度。
3. 在数据线上以合适的数据块长度发送CMD42(LOCK/UNLOCK)命令，并包含16位的CRC码。数据块包含了操作模式(SET_PWD=1)、长度(PWD_LEN)和密码(PWD)。在完成密码替换的情况下，长度数值(PWD_LEN)包含了新旧两个密码的长度，PWD域包含了旧的密码（正在使用的）和新的密码。
4. 当旧的密码匹配后，新的密码和它的长度被分别存储在PWD和PWD_LEN域。如果发送的旧密码不正确（大小和内容不相等），则SDIO状态寄存器（SDIO_STS）中的LOCK_UNLOCK_FAILED错误位将被设置，并且旧密码不会更改。如果发送的旧密码正确（大小和内容相等），则给定的新密码及其大小将分别保存在PWD和PWD_LEN寄存器中。

密码长度域（PWD_LEN）指示当前是否设置了密码，等于0时，未设置密码。如果PWD_LEN的值不等于零，则卡在上电后会自行锁定。如果该域为非零，则表示使用了密码，卡在上电时自动上锁。在不断电的情况下，如果设置了密码，可以通过设置LOCK_UNLOCK位或发送一个额外的锁定命令，立即锁住卡。

清除密码

1. 如果先前未选择卡，先使用CMD7（SELECT/DESELECT_CARD）选择一个卡
2. 使用CMD16(SET_BLOCKLEN)定义要在8位的卡锁定/解锁模式下发送的数据块长度，8位的PWD_LEN，当前密码的字节数目。
3. 在数据线上以合适的数据块长度发送CMD42(LOCK/UNLOCK)命令，并包含16位的CRC码。数据块包含了操作模式(SET_PWD=1)、长度(PWD_LEN)和密码(PWD)。当密码匹配后，PWD域被清除同时PWD_LEN被设为0。如果送出的密码与期望的密码（长度或内容）不吻合，则设置SDIO状态寄存器（SDIO_STS）中的LOCK_UNLOCK_FAILED错误位，同时密码不变。

卡锁定

1. 如果先前未选择卡，先使用CMD7（SELECT/DESELECT_CARD）选择一个卡
2. 使用CMD16(SET_BLOCKLEN)定义要在8位的卡锁定/解锁模式下发送的数据块长度，8位的PWD_LEN，当前密码的字节数目。
3. 在数据线上以合适的数据块长度发送CMD42(LOCK/UNLOCK)命令，并包含16位的CRC码。数据块包含了操作模式(SET_PWD=1)、长度(PWD_LEN)和密码(PWD)。
4. 当密码匹配后，卡被锁定且设置SDIO状态寄存器（SDIO_STS）中的CARD_IS_LOCKED状态位。如果送出的密码与期望的密码（长度或内容）不吻合，则设置SDIO状态寄存器（SDIO_STS）中的LOCK_UNLOCK_FAILED错误位，同时锁定操作失败。

如果曾经设置过密码（PWD_LEN不为0），卡会在上电复位时自动地上锁。对已经上锁的卡执行上锁操作或对没有密码的卡执行上锁操作会导致失败，并设置SDIO状态寄存器（SDIO_STS）中的LOCK_UNLOCK_FAILED错误位。

卡解锁

1. 如果先前未选择卡，先使用CMD7（SELECT/DESELECT_CARD）选择一个卡
2. 使用CMD16(SET_BLOCKLEN)定义要在8位的卡锁定/解锁模式下发送的数据块长度，8位的PWD_LEN，当前密码的字节数目。
3. 在数据线上以合适的数据块长度发送CMD42(LOCK/UNLOCK)命令，并包含16位的CRC码。数据块包含了操作模式(SET_PWD=1)、长度(PWD_LEN)和密码(PWD)。
4. 当密码匹配后，卡锁被解除，同时SDIO状态寄存器（SDIO_STS）中的CARD_IS_LOCKED位被清除。如果送出的密码与期望的密码（长度或内容）不吻合，则设置SDIO状态寄存器（SDIO_STS）中的LOCK_UNLOCK_FAILED错误位，

同时卡仍保持上锁状态。

解锁状态只在当前的供电过程中有效，只要不清除 PWD 域，下次上电后卡会被自动上锁。

试图对已经解了锁的卡执行解锁操作会导致操作失败，并设置 SDIO 状态寄存器（SDIO_STS）中的 LOCK_UNLOCK_FAILED 错误位。

强制擦除

如果用户忘记了密码（PWD 的内容），可以在清除卡中的所有内容后使用卡。强制擦除操作擦除所有卡中的数据和密码。

1. 如果先前未选择卡，先使用CMD7（SELECT/DESELECT_CARD）选择一个卡
2. 使用CMD16(SET_BLOCKLEN)定义要在8位的卡锁定/解锁模式下发送的数据块长度，8位的PWD_LEN，当前密码的字节数目。
3. 在数据线上以合适的数据块长度发送CMD42(LOCK/UNLOCK)命令，并包含16位的CRC码。数据块包含了操作模式（ERASE=1）所有其它位为0。
4. 当ERASE位是数据域中仅有的位时，卡中的所有内容将被擦除，包括PWD和PWD_LEN域，同时卡不再被上锁。如果有任何其它位不为0，则设置SDIO状态寄存器（SDIO_STS）中的LOCK_UNLOCK_FAILED错误位，卡中的数据保持不变，同时卡仍保持上锁状态。

试图对已经解了锁的卡执行擦除操作会导致操作失败，并设置 SDIO 状态寄存器（SDIO_STS）中的 LOCK_UNLOCK_FAILED 错误位。

26.3.2 命令与响应

26.3.2.1 命令

命令类型

四种命令来控制 SD 储存卡：

1. 广播命令：发送到所有卡无响应
2. 带有响应的广播命令：发送到所有卡，收到来自所有卡的同时响应
3. 寻址命令：发送到已选定的卡，SDIO_D 数据在线没有数据传输
4. 已寻址数据传输命令：发送到已选定的卡，SDIO_D 数据在总线有数据传输

详细命令描述

SDIO 主机模块系统是用于提供一个适用于多种应用类型的标准接口，但同时又要兼顾特定用户和应用的功能，因此标准中定义了两类通用命令：通用命令（GEN_CMD）和应用相关命令（ACMD）。

若要使用应用相关命令，SDIO 主机需先发送 CMD55(APP_CMD)，待卡响应给主机指示设置了 APP_CMD 位并等待 ACMD 命令，接着再发送 ACMD 命令。

表 26-2 基于命令

CMD 索引	类型	参数	响应格式	缩写	说明
CMD0	bc	[31: 0]=填充位	-	GO_IDLE_STATE	复位所有的卡到空闲状态
CMD1	bc	[31: 0]=OCR	R3	SEND_OP_COND	在空闲状态请求卡通过 CMD 总线发送 OCR 寄存器的内容
CMD2	bcr	[31: 0]=填充位	R2	ALL_SEND_CID	请求所有卡通过 CMD 总线发送 CID 数据
CMD3	bcr	[31: 0]=填充位	R6	SEND_RELATIVE_ADDR	请求卡发布新的相对卡地址(RCA)
CMD4	bc	[31: 16]=DSR [15: 0]=填充位	-	SET_DSR	设置所有卡的 DSR 寄存器
CMD5	bcr	[31: 24]保留位 [23: 0] I/O OCR	R4	IO_SEND_OP_C	仅用于 SDI/O 卡，查询所需要的 I/O 卡电压范围
CMD6	ac	[31: 26] 设为 0 [25: 24] 访问 [23: 16] 索引	R1b	SWITCH	仅用于 MMC 卡，切换选择卡的操作模式或是修改 EXT_CSD 寄存器

		[15: 8] 值 [7: 3] 设为 0 [2: 0] 命令集			
CMD7	ac	[31: 16]=RCA [15: 0]=填充位	R1b	SELECT/DESELECT_CARD	这个命令用于卡在待机状态和发送状态之间切换，或是编成和断开状态间切换，若要选择该卡则用他自己的相对地址，地址 0 用于取消选择该卡
CMD8 (SD)	bcr	[31: 12]保留位 [11: 8]工作电压 (VHS) [7: 0]检查模式	R7	SEND_IF_COND	向 SD 卡发送主机供电电压讯息和询问卡是否支持电压
CMD8 (MMC)	adtc	[31: 0]=填充位	R1	SEND_EXT_CSD	仅用于 MMC 卡，卡发送自己的 EXT_CSD 寄存器作为数据块
CMD9	ac	[31: 16]=RCA [15: 0]=填充位	R2	SEND_CSD	被选择的卡通过 CMD 总线发送 CSD(卡特定数据)
CMD10	ac	[31: 16]=RCA [15: 0]=填充位	R2	SEND_CID	被选择的卡通过 CMD 总线发送 CID(卡标志)
CMD12	ac	[31: 0]=填充位	R1b	STOP_TRANSMISSION	强制卡停止传输
CMD13	ac	[31: 16]=RCA [15: 0]=填充位	R1	SEND_STATUS	被选择的卡发送状态寄存器
CMD15	ac	[31: 16]=RCA [15: 0]=填充位	-	GO_INACTIVE_STATE	被选择的卡切换到非激活状态

表 26-3 数据块读取命令

CMD 索引	类型	参数	响应格式	缩写	说明
CMD16	ac	[31: 0]=数据块长度	R1	SET_BLOCKLEN	该命令为所有后续块命令设置数据块长度(以字节为单位)，默认是 512 字节
CMD17	adtc	[31: 0]=数据地址	R1	READ_SINGLE_BLOCK	读取由 CMD16 设置大小的数据块
CMD18	adtc	[31: 0]=数据地址	R1	READ_MULTIPLE_BLOCK	不断从卡读取数据到主机，直到收到 STOP_TRANSMISSION 命令

表 26-4 数据流读取和写入命令

CMD 納引	类型	参数	响应格式	縮写	说明
CMD11	adtc	[31: 0]=数据地址	R1	READ_DAT_UNTIL_STOP	从卡中读取数据流，从给定的地址开始，直到收到 STOP_TRANSMISSION 命令
CMD20	adtc	[31: 0]=数据地址	R1	WRITE_DAT_UNTIL_STOP	从主机写数据流，从给定的地址开始，直到收到 STOP_TRANSMISSION 命令

表 26-5 数据块写入命令

CMD 索引	类型	参数	响应格式	缩写	说明
CMD16	ac	[31: 0]=数据块长度	R1	SET_BLOCKLEN	该命令为所有后续块命令设置数据块长度(以字节为单位), 默认是 512 字节
CMD23	ac	[31: 16]=设为 0 [15: 0]=数据块数	R1	SET_BLOCK_COUNT	定义后续数据块读写的块数目
CMD24	adtc	[31: 0]=数据地址	R1	WRITE_BLOCK	写入由 CMD16 设置大小的数据块
CMD25	adtc	[31: 0]=数据地址	R1	WRITE_MULTIP LE_BLOCK	连续写入数据块, 直到收到 STOP_TRANSMISSION 命令
CMD26	adtc	[31: 0]=填充位	R1	PROGRAM_CID	对卡识别寄存器进行编程
CMD27	adtc	[31: 0]=填充位	R1	PROGRAM_CSD	对 CSD 的可编程位编程

表 26-6 基于块传输的写保护命令

CMD 紹引	类型	参数	响应格式	縮写	说明
CMD28	ac	[31: 0]=数据地址	R1b	SET_WRITE_PROT	如果卡具有写保护的功能, 该命令设置指定组的写保护位。写保护的属性设置在卡的特定数据域 (WP_GRP_SIZE)。
CMD29	ac	[31: 0]=数据地址	R1b	CLR_WRITE_PR OT	如果卡具有写保护的功能, 该命令清除指定组的写保护位。
CMD30	adtc	[31: 0]=写保护 数据地址	R1	SEND_WRITE_P ROT	如果卡具有写保护的功能, 该命令要求卡发送写保护位的状态。

表 26-7 擦除命令

CMD 紹引	类型	参数	响应格式	縮写	说明
CMD32		...			保留。为了与旧版本的对媒体卡协议向后兼容, 不能使用这些命令代码。
CMD34					
CMD35	ac	[31: 0]=数据地址	R1	ERASE_GROUP _START	在选择的擦除范围内, 设置第一个擦除组的地址。
CMD36	ac	[31: 0]=数据地址	R1	ERASE_GROUP _END	在选择的连续擦除范围内, 设置最后一个擦除组的地址。
CMD37		...			保留。为了与旧版本的对媒体卡协议向后兼容, 不能使用这个命令代码。
CMD38	ac	[31: 0]=填充位	R1b	ERASE	擦除之前选择的数据块。

表 26-8 I/O 模式命令

CMD 索引	类型	参数	响应格式	缩写	说明
CMD39	ac	[31: 16]=RCA [15]=寄存器写标志 [14: 8]=寄存器地址 [7: 0]=寄存器数据	R4	FAST_IO	用于写入和读取 8 位（寄存器）数据域。该命令指定一个卡和寄存器，如果设置了写标志还提供写入的数据。R4 响应包含从指定寄存器读出的数据。该命令访问未在多媒体卡标准中定义的与应用相关的寄存器。
CMD40	bcr	[31: 0]=填充位	R5	GO_IRQ_STATE	置系统于中断模式。

表 26-9 卡锁定命令

CMD 索引	类型	参数	响应格式	缩写	说明
CMD42	adtc	[31: 0]=填充位	R1	LOCK_UNLOCK	设置/清除密码，又或是对卡锁定/解锁，数据块的长度由 CMD16 定义

表 26-10 应用相关命令

CMD 索引	类型	参数	响应格式	缩写	说明
CMD55	ac	[31: 16]=RCA [15: 0]=填充位	R1	APP_CMD	指示卡下一个命令是应用相关命令而不是一个标准命令。
CMD56	adtc	[31: 1]=填充位 [0]=RD/WR	R1	GEN_CMD	在通用或应用相关命令中，或者用于向卡中传输一个数据块，或者用于从卡中读取一个数据块。数据块的长度由 SET_BLOCK_LEN 命令设置。
CMD57 ... CMD59		保留。			
CMD60 ... CMD63		保留给生产厂商。			

26.3.2.2 响应格式

所有的响应都是通过 CMD 总线发送，响应传输总是从与响应回应字相对应的位串的左位开始，响应字的长度取决于响应类型。

响应总是以起始位(始终为 0)开始，然后是指示传输方向的传输位(卡=0)，下表中标示为 - 的数值表示为可变的部分，除了 R3 响应类型外，所有的响应均受 CRC 保护，每个命令码字都以结束位(始终为 1)终止。

26.3.2.2.1 R1 (普通响应命令)

编码长度为 48 位，位 45 : 40 指示要响应的命令的索引，该值被解释为二进制编码的数字（介于 0 和 63 之间）。卡的状态以 32 位编码。请注意，如果涉及到向卡的数据传输，则在传输每个数据块后，数据在线可能会出现繁忙信号。数据块传输后，主机应检查是否忙碌。

表 26-11 R1 响应

位	47	46	[45: 40]	[39: 8]	[7: 1]	0
域宽度	1	1	6	32	7	1
数值	0	0	-	-	-	1
说明	开始位	传输位	命令索引	卡状态	CRC7	结束位

26.3.2.2.2 R1b

R1b 与 R1 相同，只是在数据在总线传输了可选的忙信号。根据命令接收之前的状态，卡在接收到这些命令后可能会变得很忙。主机应检查响应是否忙碌。

26.3.2.2.3 R2(CID、CSD寄存器)

编码长度为 136 位，CID 寄存器的内容作为对命令 CMD2 和 CMD10 的响应发送。CSD 寄存器的内容作为对 CMD9 的响应发送。仅传送 CID 和 CSD 的位[127 ... 1]，这些寄存器的保留位[0]被响应的结束位替换。

表 26-12 R2响应

位	135	134	[133 : 128]	[127 : 1]	0
域宽度	1	1	6	127	1
数值	1	0	111111	-	1
说明	开始位	传输位	保留	CID 或 CSD 寄存器	结束位

26.3.2.2.4 R3(OCR寄存器)

编码长度为 48 位，该 OCR 寄存器内容作为 ACMD41 的响应被发送。

表 26-13 R3响应

位	47	46	[45: 40]	[39: 8]	[7: 1]	0
域宽度	1	1	6	32	7	1
数值	1	0	111111	-	111111	1
说明	开始位	传输位	保留	OCR 寄存器	保留	结束位

26.3.2.2.5 R4(快速I/O)

编码长度为 48 位，参数域包含指定卡的 RCA、需要读出或写入寄存器的地址、和它的内容。

表 26-14 R4响应

位	47	46	[45: 40]	[39: 8]	[7: 1]	0
域宽度	1	1	6	16	8	8
数值	1	0	100111	-	-	-
说明	开始位	传输位	CMD39	RCA	寄存器地址	读寄存器的内容

26.3.2.2.6 R4b

仅适合 SD I/O 卡：一个 SDIO 卡收到 CMD5 后将返回一个唯一的 SDIO 响应 R4

表 26-15 R4b响应

位	47	46	[45: 40]	[39: 8]	[7: 1]	0
域宽度	1	1	6	1 3 1 3 24	7	1
数值	1	0	- - - -	- - - -	- - - -	1
说明	开始位	传输位	保留	卡就绪 I/O 功当前 填充位 I/O 保留	OCR	结束位

26.3.2.2.7 R5(中断请求)

仅适用于多媒体卡。代码长度=48位。如果这个响应由主机产生，则参数中的 RCA 域为 0x0。

表 26-16 R5响应

位	47	46	[45: 40]	[39: 8]	[7: 1]	0
域宽度	1	1	6	16	16	7 1
数值	1	0	101000	-	-	- 1
说明	开始位	传输位	CMD40	成功的卡或主机的 RCA[31: 16]	未定义可以 作为中断数据。	CRC7 结束位

26.3.2.2.8 R6(中断请求)

仅适用于 SD I/O 卡。这是一个存储器设备对 CMD3 命令的正常响应

表 26-17 R6响应

位	47	46	[45: 40]	[39: 8]	[7: 1]	0
域宽度	1	1	6	16	16	7 1
数值	1	0	000011	-	-	- 1
说明	开始位	传输位	CMD3	成功的卡或主机的 RCA[31: 16]	卡状态	CRC7 结束位

当发送 CMD3 命令到只有 I/O 功能的卡时，卡的状态位[23: 8]会改变；此时，响应中的 16 位将是只有 I/O 功能的 SD 卡中的数值：

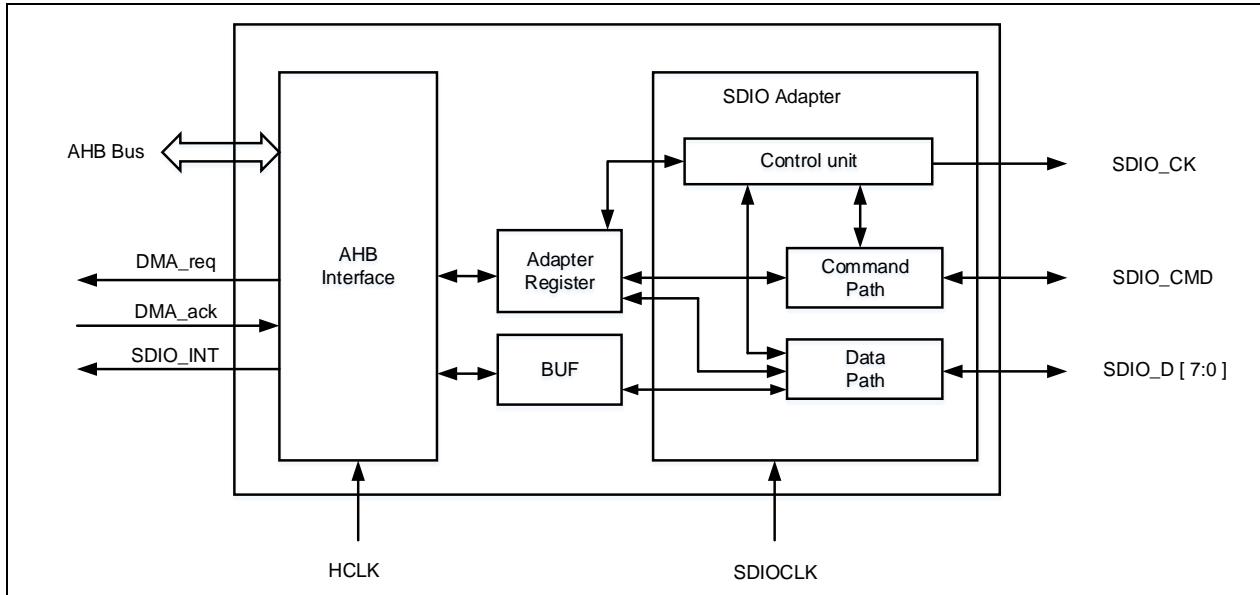
- 位 15=COM_CRC_ERROR
- 位 14=ILLEGAL_COMMAND
- 位 13=ERROR
- 位[12: 0]=保留

26.3.3 SDIO功能描述

SDIO 包含 4 个部分：

- SDIO 适配器模块：由控制单元、命令单元和数据单元所组成，实现所有 MMC/SD/SD I/O 卡的相关功能，如时钟的产生、命令和数据的传送
 - 控制单元：管理并产生时钟信号
 - 命令单元：管理命令的传输
 - 数据单元：管理数据的传输
- AHB 接口：产生中断和 DMA 请求信号
- 适配器寄存器：系统寄存器
- BUF：用于数据传输

图 26-6 SDIO 框图



26.3.3.1 SDIO适配器

SDIO_CK 是主机给多媒体/SD/SDIO 卡的时钟，每个时钟周期在命令和数据在总线传输 1 位命令或数据，不同的卡和协议之间有不同的时钟频率限制

- 多媒体卡
 - V3.31 协议 0 – 20MHz
 - V4.0/4.2 协议 0 – 48MHz
- SD 卡
 - 0 – 48MHz
- SD I/O 卡
 - 0 – 48MHz

SDIO_CMD 信号是双向命令通道，用于卡的初始化和命令传输，主机发送命令至卡端后，卡送出响应至主机端，**SDIO_CMD** 信号有两种操作模式：

- 用于初始化时的开路模式（仅用于 MMC 版本 V3.31 或之前版本）
- 用于命令传输的推挽模式（SD/SD I/O 卡和 MMC V4.2 在初始化时也使用推挽驱动）

SDIO_D [7:0] 信号是双向数据信道，初始化后主机可以改变数据总线的宽度，在复位后默认情况下 SDIO_D0 用于数据传输，MMC 卡在 V3.31 和之前版本的协议只支持一位数据线，只能使用 **SDIO_D0**。下表适用于多媒体卡/SD/SD I/O 卡总线：

表 26-18 SDIO管脚定义

管脚	方向	说明
SDIO_CK	输出	多媒体卡/SD/SDIO 卡时钟。这是从主机至卡的时钟线。
SDIO_CMD	双向	多媒体卡/SD/SDIO 卡命令。这是双向的命令/响应信号线。
SDIO_D[7: 0]	双向	多媒体卡/SD/SDIO 卡数据。这些是双向的数据总线。

控制单元

控制单元包含电源和时钟管理功能，电源管理的部分主要由 **SDIO_PWRCTRL** 寄存器控制，PS 位控制上下电，在电源关闭和电源启动阶段，电源管理子单元会关闭卡总线上的输出信号，时钟管理则是由 SDIO 时钟控制寄存器 (**SDIO_CLKCTRL**) 控制，**CLKDIV** 位定义了 SDIO 时钟(**SDIOCLK**)与 SDIO 输出至卡端的时钟(**SDIO_CK**)间的分频系数关系，若 **BYPSEN** 位为 0，**SDIO_CK** 输出信号由 **SDIOCLK** 依据 **CLKDIV** 位分频后驱动，若 **BYPSEN** 位被置 1，则 **SDIO_CK** 输出信号直接由 **SDIOCLK** 驱动，将 **HFCEN** 位置 1 可以开启硬件流控制功能，避免在发送模式出现下溢和接收模式出现上溢的错误，软件可通过设置 **PWRSVEN** 位开启省电模式，仅有在总线活动时才会输出 **SDIO_CK**。

命令通道

命令通道负责向卡发送和接收命令，将 SDIO_CMD 寄存器的 CCSMEN 位置 1 后，命令传输开始，首先向卡发送一个命令，这个命令共 48 位，通过 SDIO_CMD 发出，SDIO_CMD 上的数据与 SDIO_CK 的上升沿同步，每个 SDIO_CK 传输一笔数据，包含开始位、传输位、由 SDIO_CMD 寄存器 CMDIDX 位定义的命令索引、SDIO 参数寄存器 (SDIO_ARG) 定义的参数、7 位的 CRC 和停止位，然后接收卡端的响应，响应可分为 48 位的短响应和 136 位的长响应，2 种类型都有 CRC 错误检测，收到的响应回存在 SDIO_RSP1 到 SDIO_RSP4 中，命令通道可以产生命令状态标志并由 SDIO 状态寄存器 (SDIO_STS) 定义。

表 26-19 命令格式

位	47	46	[45 : 40]	[39 : 8]	[7 : 1]	0
宽度	1	1	6	32	7	1
数值	0	1	-	-	-	1
说明	开始位	传输位	命令索引	参数	CRC7	结束位

— 响应：响应是由一个被指定地址的卡发送到主机，对于 MMCV3.31 或以前版本所有的卡同时发送响应；响应是对先前接收到命令的一个应答。响应在 CMD 线上串行传送。

表 26-20 短响应格式

位	47	46	[45 : 40]	[39 : 8]	[7 : 1]	0
宽度	1	1	6	32	7	1
数值	0	0	-	-	-	1
说明	开始位	传输位	命令索引	参数	CRC7(或 1111111)	结束位

表 26-21 长响应格式

位	135	134	[133: 128]	[127 : 1]	0
宽度	1	1	6	127	1
数值	0	0	111111	-	1
说明	开始位	传输位	保留	CID 或 CSD(包含 内部 CRC7)	结束位

表 26-22 命令通道状态标志

标志	说明
CMDRSPCMPL	已接受到响应(CRC 检测成功)
CMDFAIL	已收到命令响应(CRC 检测失败)
CMDCMPL	命令 (不需要响应的命令) 已发送
CMDTIMEOUT	命令响应超时(64 个 SDIO_CK 时钟周期)
DOCMD	正在发送命令

命令通道状态机 (CCSM)

当设置 SDIO_CMD 寄存器的 CCSMEN 位，控制器开始发送命令。命令发送完成时，命令通道状态机 (CCSM) 设置命令通道状态标志并在不需要响应时进入空闲状态。当收到响应后，接收到的 CRC 码将会与内部产生的 CRC 码比较，然后设置相应的状态标志。

- CCSM 在空闲状态至少保持 8 个 SDIO_CK 周期以满足 N_{cc}(两个主机命令之间的最小时间间隔) 和 N_{rc}(主机命令与卡响应之间的最长时间间隔) 的时序限制，

- 在发送完成后进入等待状态时，会启动命令通道内的定时器，若进入接收状态前违反 N_{CR}(指令响应时间)时序，超过了 64 个 SDIO_CK 的时间，会设置超时标志 (CMDTIMEOUT) 并回到空闲状态。
- 如果在命令寄存器设置了中断位，则关闭定时器，CCSM 等待某一个卡发出的中断请求。如果命令寄存器中设置挂起位，CCSM 进入挂起 (Pend) 状态并等待数据通道子单元发出的 CmdPend 信号，在检测到 CmdPend 信号时，CCSM 进入发送 (Send) 状态，这将触发数据计数器发送停止命令的功能。

图 26-7 命令通道状态机 (CCSM)

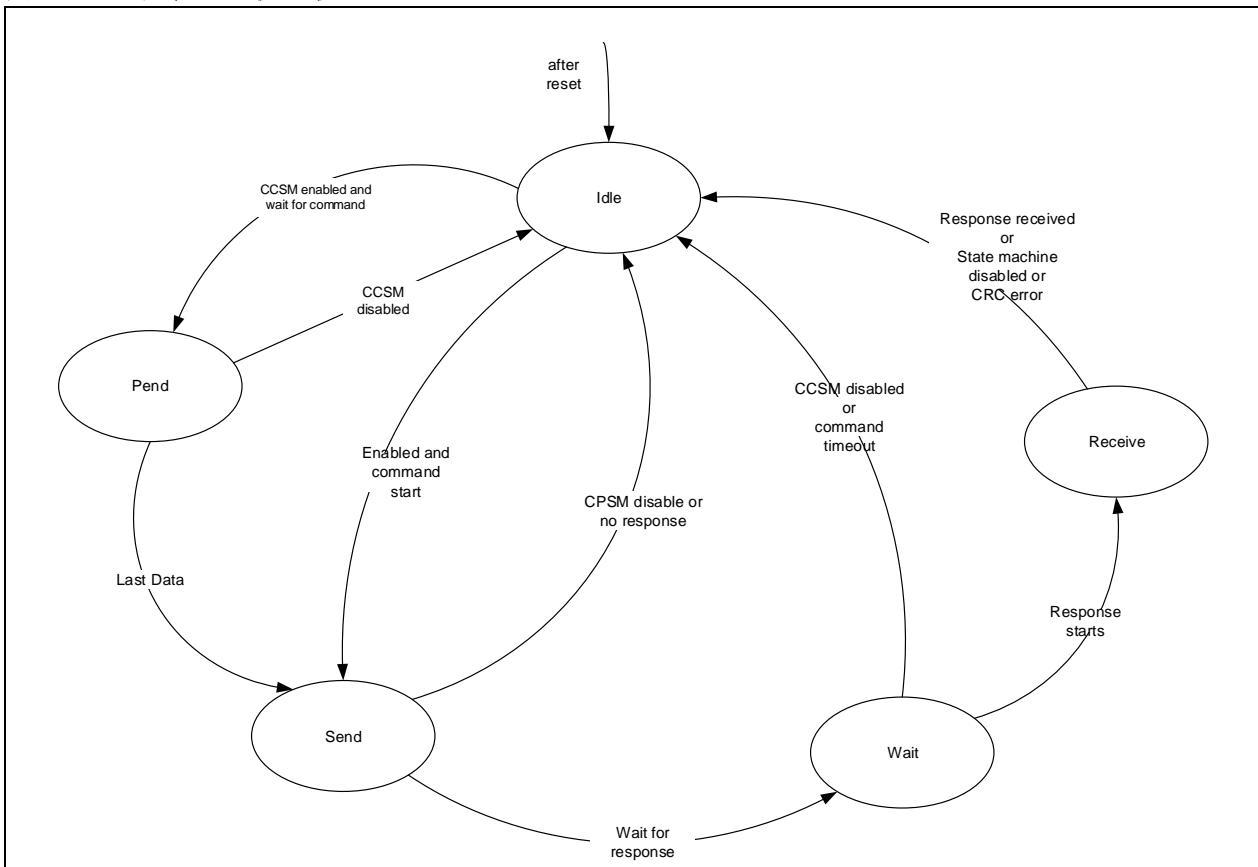
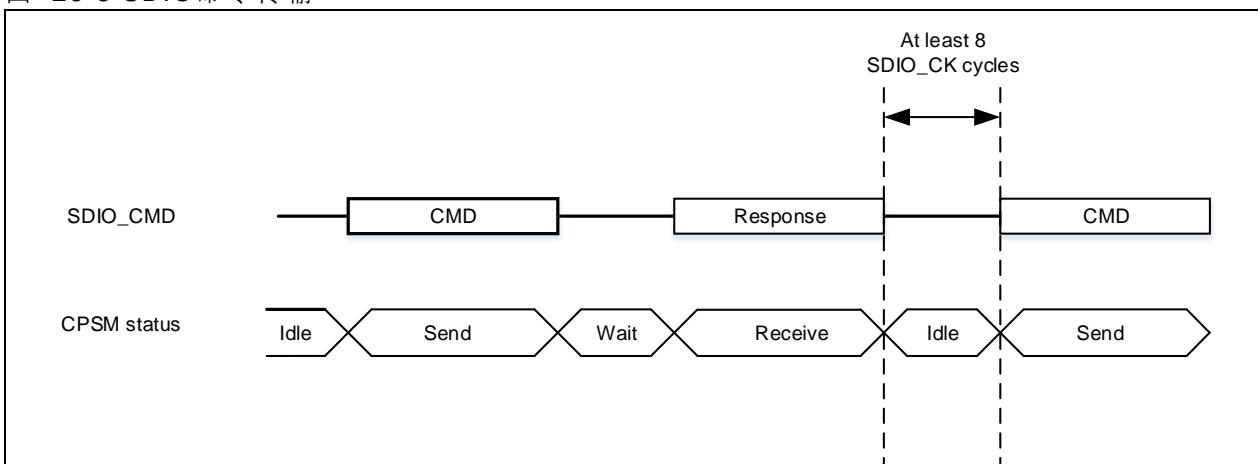


图 26-8 SDIO命令传输



数据通道

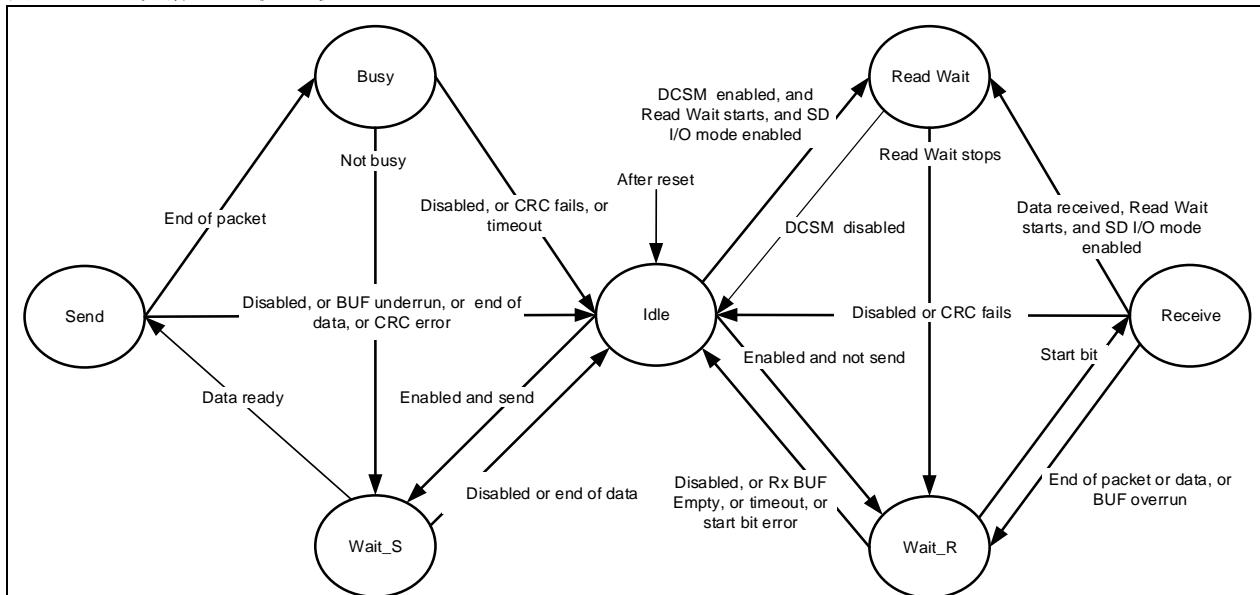
数据通道负责实现主机与卡之间数据传输，可以透过设定 SDIO 时钟控制寄存器 (SDIO_CLKCTRL) 的 BUSWS 位选择数据总线宽度，默认情况下只会使用 SDIO_D0 信号线传输，每个时钟周期传输 1 位的数据，可以选择设定 4 位总线模式，每个时钟周期传输 4 位数据并使用 SDIO_D[3:0] 信号线，若设定 8 位总线模式，每个时钟周期传输 8 位数据并使用 SDIO_D[7:0] 信号线，设定 SDIO_DTCTRL 寄存器的 TFRDIR 位可以传输方向，当 TFRDIR 位为 0 时表示传输方向是从控制器至卡端，若为 1 则是表示传输方向是从卡至控制器，TFRMODE 位可配合多媒体卡选择块数据传输或是流数据传输，如果将 TFREN 位置 1，则开始传输数据，根据 TFRDIR 位决定传输的方向(发送或接收)，使能时数据通道状态机(DCSM)

将进入 Wait_S 或 Wait_R 状态。

数据通道状态机 (DCSM)

DCSM 有 7 个状态，可分为发送和接收模式来看，如下图所示：

图 26-9 数据通道状态机 (DCSM)



发送模式

- 空闲 (Idle)：数据通道不工作，等待发送(进入 Wait_S 状态)或接收数据(进入 Wait_R 状态)
- 等待发送(Wait_S)：等待数据 Buf 为空标志或是数据传输结束，须至少保持两个时钟周期，以满足 N_{WR} 的时序要求，N_{WR} 是接收到卡的响应置主机开始传输数据的间隔。
- 发送(Send)：发送数据到卡，根据 SDIO_DTCTRL 寄存器 TFRMODE 位决定是块数据或是流数据传输，若发生下溢错误则会回到空闲状态。
- 繁忙(Busy)：等待 CRC 标志，若接收正确且卡不繁忙时回到 Wait_S 状态，若不正确或是繁忙状态超时则回到空闲状态，并产生 CRC 失败标志或是超时标志。
- 等待接收(Wait_R)：等待数据接收的起始位，若在检测到起始位前发生超时错误，则会回到空闲状态并产生超时标志。
- 接收(Receive)：接收来自卡端的数据并写入数据 Buf，根据 SDIO_DTCTRL 寄存器 TFRMODE 位决定是块数据或是流数据传输，若发生上溢错误则会回到等待接收状态。

表 26-23 数据令牌格式

说明	开始位	数据	CRC16	结束位
块数据	0	-	有	1
流数据	0	-	无	1

26.3.3.2 数据BUF

数据 BUF 是一个具有发送和接收单元，每字 32 位宽、共 32 个字的数据缓冲区，因为数据 BUF 工作在 AHB 时钟区域 (HCLK)，所有与 SDIO 时钟区域 (SDIOCLK) 连接的信号都进行了重新同步。

- **发送 BUF:** 当使能了 SDIO 的发送功能，数据可以通过 AHB 接口写入发送 BUF。发送 BUF 有 32 个连续的地址。发送 BUF 中有一个数据输出寄存器，包含读指针指向的数据字。当数据通道装填了移位寄存器后，它移动读指针至下个数据并传输出数据。如果未使能发送 BUF，所有的状态标志均处于无效状态。当发送数据时，数据通道设置 DOTX 为有效。
- **接收 BUF:** 当数据通道接收到一个数据字，它会把数据写入 BUF，写操作结束后，写指针自动加一；在另一端，有一个读指针始终指向 BUF 中的当前数据。如果关闭了接收 BUF，所有的状态标志会被清除，读写指针也被复位。在接收到数据时数据通道设置 DORX。

26.3.3.3 SDIO AHB 接口

AHB 接口产生中断和 DMA 请求，并访问 SDIO 接口寄存器和数据 BUF。

SDIO 中断

当有任一个选中的状态标志为高时，中断控制逻辑产生中断请求。SDIO 中断屏蔽寄存器(SDIO_INTEN)可以选择产生中断的条件。

SDIO/DMA 接口：在 SDIO 和存储器之间数据传输的过程

DMA 可设置为弹性映像或固定映像，以下介绍固定映像的配置，弹性映像请参考 DMA 章节配置
在下面的例子中，从主机传送数据到卡端，DMA 控制器用于从存储器向 SDIO 的 BUF 填充数据。

1. 卡识别过程
2. 提高 SDIO_CK 频率
3. 发送 CMD7 命令选择卡
4. 使能 DMA2 控制器并清除所有的中断标志位，设置 DMA2 信道 4 的源地址寄存器为存储器缓冲区的基地址，DMA2 信道 4 的目标地址寄存器为 SDIO_BUF 寄存器的地址，接着设置 DMA2 通道 4 控制寄存器（存储器递增，非外设递增，外设和源的数据宽度为字宽度），最后使能 DMA2 通道 4。
5. 发送 CMD24 (WRITE_BLOCK)，操作如下：
设置 SDIO 数据长度寄存器 (SDIO_DTLEN)，接着设置 SDIO 数据控制寄存器 (SDIO_DTCTRL) 的 BLKSIZE 位，之后设置 SDIO 参数寄存器 (SDIO_ARG) 写入需要传送数据的地址，配置 SDIO 命令寄存器 (SDIO_CMD)，使能 CCSMEN 位，等待 SDIO 状态寄存器 (SDIO_STS) [6]=CMDRSPCMPL 中断，然后设置 SDIO 数据控制寄存器 (SDIO_DTCTRL) TFRREN 置为 1 (使能 SDIO 卡主机发送数据)；TFRDIR 置为 0 (控制器至卡方向)；TFRMODE 置为 0 (块数据传送)；DMAEN 置为 1 (DMA 使能)；BLKSIZE 置为 9 (512 字节)，等待 SDIO 状态寄存器 (SDIO_STS) [10]=DTBLKCMPL。
6. 查询 SDIO 已使能 DMA 通道的状态寄存器，确认没有通道处于传输状态

26.3.3.4 硬件流控制

可以透过设置 SDIO 时钟控制寄存器 (SDIO_CLKCTRL) 的 HFCEN 位开启功能，避免 BUF 下溢和上溢的错误，在流控制功能开启时仍进行读出或写入 BUF 操作。

26.3.4 SDIO I/O 卡特定的操作

当设置了 SDIO 数据控制寄存器 (SDIO_DTCTRL) [11] 位时，SDIO 可支持这些操作(读暂停除外，因为它不需要特殊的硬件操作)

由 SDIO_D2 信号线实现的 SDIO 读等待操作

可选的读等待 (RW) 操作只适用于 SD 卡的 1 位或 4 位模式，读等待操作允许主机正在读多个寄存器 (IO_RW_EXTENDED, CMD53) 时，要求它暂时停止数据传输，同时允许主机发送命令到 SD I/O 设备中的其他功能，以收到第一个数据块之前即可以开始读等待过程，下描述详细过程。

- 使能数据通道 (SDIO 数据控制寄存器 (SDIO_DTCTRL) [0] = 1)
- 使能 SDIO 特定操作 (SDIO 数据控制寄存器 (SDIO_DTCTRL) [11] = 1)
- 开始读等待 (SDIO 数据控制寄存器 (SDIO_DTCTRL) [10]=0 且 SDIO 数据控制寄存器 (SDIO_DTCTRL) [8]=1)
- 数据传输方向是从卡至 SDIO 主机 (SDIO 数据控制寄存器 (SDIO_DTCTRL) [1]=1)
- SDIO 适配器的数据单元将进入读等待状态，待 2 个 SDIO_CK 后驱动 SDIO_D2 为'0'
- 数据单元开始等待从卡里接收数据。在接收数据块时，即使设置了开始读等待，DCSM 也不会进入读等待，读等待过程将在收到 CRC 后开始。必须清除 RDWTSTOP 才能开始新的读等待操作。

在读等待期间，SDIO 主机可以在 SDIO_D1 上监测 SDIO 中断。

通过停止时钟实现的 SDIO 读等待操作

如果 SDIO 卡不能支持前述的读等待操作，SDIO 可以停止 SDIO_CK 进入读等待，详细操作如下：

- 使能数据通道 (SDIO 数据控制寄存器 (SDIO_DTCTRL) [0] = 1)
- 使能 SDIO 特定操作 (SDIO 数据控制寄存器 (SDIO_DTCTRL) [11] = 1)
- 开始读等待 (SDIO 数据控制寄存器 (SDIO_DTCTRL) [10]=且 SDIO 数据控制寄存器 (SDIO_DTCTRL) [8]=1)

在接收当前数据块结束位之后的 2 个 SDIO_CK 周期后，DCSM 停止时钟，在设置了读等待结束位后恢复时钟。

需注意因为 SDIO_CK 停止，SDIO 主机不可以向卡发送任何命令，并且 SDIO 主机可以在 SDIO_D1 上监测 SDIO 中断。

SDIO 暂停/恢复操作（写和读暂停）

为了给其他的功能或者存储器提供更高优先级的传输而释放总线，主机可以暂停某个功能或者存储器的数据传输。一旦高优先级的传输完成后，原来的传输在暂停处重新开始。

在向卡发送数据时，SDIO 可以暂停写操作。设置 SDIO 命令寄存器（SDIO_CMD）[11]位并指示 CCSM 当前的命令是一个暂停命令。CCSM 分析响应，在从卡收到响应时（暂停被接受），它确认在收到当前数据块的 CRC 后 DCSM 进入空闲状态，而暂停读操作的部分，CCSM 会在 Wait_R 状态等待，若再暂停前已发送数据，应用程序会继续读出 BUF 直到 BUF 为空，之后 CCSM 进入 IDLE 状态。

SDIO 中断

为了让 SD I/O 卡能够中断 SDIO 模块，在 SD 接口上有一个具有中断功能的管脚，在 4 位 SD 模式下这个脚是 SDIO_D1，SD I/O 的中断是电平有效，即在被识别并得到 SDIO 模块的响应之前，中断信号线必须保持有效电平（低），在中断过程结束后保持无效电平（高）。

当设置了 SDIO 数据控制寄存器（SDIO_DTCTRL）[11]位，SDIO 主机在 SDIO_D1 信号线上监测 SDIO 中断。

26.4 SDIO 寄存器

设备可以通过在 AHB 上操作的 32 位控制寄存器与系统通信。

必须以字（32 位）的方式操作这些外设寄存器。

下表是 SDIO 寄存器的总结。

表 26-24 SDIO 寄存器映像

寄存器简称	基址偏移量	复位值
SDIO_PWRCTRL	0x00	0x0000 0000
SDIO_CLKCTRL	0x04	0x0000 0000
SDIO_ARG	0x08	0x0000 0000
SDIO_CMD	0x0C	0x0000 0000
SDIO_RSPCMD	0x10	0x0000 0000
SDIO_RSP1	0x14	0x0000 0000
SDIO_RSP2	0x18	0x0000 0000
SDIO_RSP3	0x1C	0x0000 0000
SDIO_RSP4	0x20	0x0000 0000
SDIO_DTTMR	0x24	0x0000 0000
SDIO_DTLEN	0x28	0x0000 0000
SDIO_DTCTRL	0x2C	0x0000 0000
SDIO_DTCNTR	0x30	0x0000 0000
SDIO_STS	0x34	0x0000 0000
SDIO_INTCLR	0x38	0x0000 0000
SDIO_INTEN	0x3C	0x0000 0000
SDIO_BUFCNTR	0x48	0x0000 0000
SDIO_BUF	0x80	0x0000 0000

26.4.1 SDIO电源控制寄存器（SDIO_PWRCTRL）

域	简称	复位值	类型	功能
位 31: 2	保留	0x0000 0000	resd	保持默认值。
位 1: 0	PS	0x0	rw	电源开关位（Power switch） 由软件置起或清零。该位用于定义卡时钟的当前状态。 00: 关闭, 卡的时钟停止; 01: 保留; 10: 保留; 11: 开启, 卡的时钟开启。

注意：写数据后的 7 个 HCLK 时钟周期内，不能写入这个寄存器。

26.4.2 SDIO时钟控制寄存器（SDIO_CLKCTRL）

SDIO 时钟控制寄存器（SDIO_CLKCTRL）控制 SDIO_CK 输出时钟。

域	简称	复位值	类型	功能
位 31: 17	保留	0x0000	resd	保持默认值。
位 16: 15	CLKDIV	0x0	rw	时钟分频系数（Clock division） 由软件置起或清零。该位定义了 SDIO 时钟（SDIOCLK）与 SDIO 总线时钟（SDIO_CK）间的分频系数关系： $SDIO_CK\ 频率 = SDIOCLK / [CLKDIV[9: 0] + 2]$ 。
位 14	HFCEN	0x0	rw	硬件流控制使能（Hardware flow control enable） 由软件置起或清零。 0: 关闭; 1: 开启。 注：当开启硬件流控制后，关于 TXBUF_E 和 RXBUF_F 中断信号的意义请参考 SDIO 状态寄存器（SDIO_STS）的定义。
位 13	CLKEGS	0x0	rw	SDIO_CK 边沿选择（SDIO_CK edge selection） 由软件置起或清零。 0: 在主时钟 SDIOCLK 上升沿产生 SDIO_CK; 1: 在主时钟 SDIOCLK 下降沿产生 SDIO_CK。
位 12: 11	BUSWS	0x0	rw	总线宽度选择（bus width selection） 由软件置起或清零。 00: 默认总线模式，使用 SDIO_D0; 01: 4 位总线模式，使用 SDIO_D[3: 0]; 10: 8 位总线模式，使用 SDIO_D[7: 0]。
位 10	BYPSEN	0x0	rw	旁路时钟分频器（Clock divider bypass enable bit） 由软件置起或清零。关闭表示 SDIO_CK 输出信号由 SDIOCLK 依据 CLKDIV 数值分频后驱动，开启表示 SDIO_CK 输出信号直接由 SDIOCLK 驱动。 0: 关闭; 1: 开启。
位 9	PWRSVEN	0x0	rw	省电模式使能（Power saving mode enable） 由软件置起或清零。关闭表示始终都会输出 SDIO_CK，开启表示仅在有总线活动时才会输出 SDIO_CK。 0: 关闭; 1: 开启。
位 8	CLKOEN	0x0	rw	时钟输出使能（Clock output enable） 由软件置起或清零。 0: 关闭; 1: 开启。
位 7: 0	CLKDIV	0x00	rw	时钟分频系数（Clock division） 由软件置起或清零。该位定义了 SDIO 时钟（SDIOCLK）与 SDIO 总线时钟（SDIO_CK）间的分频系数关系： $SDIO_CK\ 频率 = SDIOCLK / [CLKDIV[9: 0] + 2]$ 。

注意：1. 当 SD/SDIO 卡或多媒体卡在识别模式，SDIO_CK 的频率必须低于 400kHz。

2. 当所有卡都被赋予了相应的地址后，时钟频率可以改变到卡总线允许的最大频率。

3. 写数据后的 7 个 HCLK 时钟周期内不能写入这个寄存器。对于 SD I/O 卡，在读等

待期间可以停止 SDIO_CK，此时 SDIO 时钟控制寄存器（SDIO_CLKCTRL）不控制 SDIO_CK。

26.4.3 SDIO参数寄存器（SDIO_ARG）

SDIO 参数寄存器（SDIO_ARG）包含 32 位命令参数，它将作为命令的一部分发送到卡中。

域	简称	复位值	类型	功能
位 31: 0	ARGU	0x0000 0000	rw	命令参数（Command argument） 命令参数是发送到卡中命令的一部分，如果一个命令包含一个参数，必须在写命令到命令寄存器之前加载这个寄存器。

26.4.4 SDIO命令寄存器（SDIO_CMD）

SDIO 命令寄存器（SDIO_CMD）包含命令索引和命令类型位。命令索引是作为命令的一部分发送到卡中。命令类型位控制命令通道状态机（CCSM）。

域	简称	复位值	类型	功能
位 31: 12	保留	0x00000	resd	保持默认值。
位 11	IOSUSP	0x0	rw	SD I/O 暂停命令（SD I/O suspend command） 由软件置起或清零。如果该位被置起，则将要发送的命令是一个暂停命令（只能用于 SDIO 卡）。 0: 关闭； 1: 开启。
位 10	CCSMEN	0x0	rw	命令通道状态机使能（Command channel state machine (CCSM) enable bit） 由软件置起或清零。 0: 关闭； 1: 开启。
位 9	PNDWT	0x0	rw	CCSM 等待数据传输结束（CmdPending 内部信号） (CCSM Waits for ends of data transfer (CmdPending internal signal)) 由软件置起或清零。如果该位被置起，则 CCSM 在开始发送一个命令之前会等待数据传输结束。 0: 关闭； 1: 开启。
位 8	INTWT	0x0	rw	CCSM 等待中断请求（CCSM waits for interrupt request） 由软件置起或清零。如果该位被置起，则 CCSM 关闭命令超时控制并等待中断请求。 0: 关闭； 1: 开启。
位 7: 6	RSPWT	0x0	rw	等待响应位（Wait for response bits） 由软件置起或清零。该位指示 CCSM 是否需要等待响应，如果需要等待响应，则指示响应类型。 00: 无响应； 01: 短响应； 10: 无响应； 11: 长响应。
位 5: 0	CMDIDX	0x00	rw	命令索引（Command index） 命令索引是作为命令的一部分发送到卡中。

注意：1. 写数据后的 7 个 HCLK 时钟周期内不能写入这个寄存器。

2. 多媒体卡可以发送 2 种响应：48 位长的短响应，或 136 位长的长响应。SD 卡和 SD I/O 卡只能发送短响应，参数可以根据响应的类型而变化，软件将根据发送的命令区分响应的类型。

26.4.5 SDIO命令响应寄存器 (SDIO_RSPCMD)

SDIO 命令响应寄存器 (SDIO_RSPCMD) 包含最后收到的命令响应中的命令索引。如果传输的命令响应不包含命令索引 (长响应或 OCR 响应)，尽管它应该包含 111111b (响应中的保留域值)，但 RSPCMD 域的内容未知。

域	简称	复位值	类型	功能
位 31: 6	保留	0x0000000	resd	保持默认值。
位 5: 0	RSPCMD	0x00	ro	响应的命令索引 (Response command index) 收到的命令响应中的命令索引。

26.4.6 SDIO响应1..4寄存器 (SDIO_RSPx)

SDIO 响应 1..4 寄存器 (SDIO_RSPx) 包含卡的状态，即收到响应的部分信息。

域	简称	复位值	类型	功能
位 31: 0	CARDSTSx	0x0000 0000	ro	见下表

根据响应状态，卡的状态长度是 32 位或 127 位。

表 26-25 响应类型和 SDIO_RSPx 寄存器

寄存器	短响应	长响应
SDIO_RSP1	卡状态[31: 0]	卡状态[127: 96]
SDIO_RSP2	不用	卡状态[95: 64]
SDIO_RSP3	不用	卡状态[63: 32]
SDIO_RSP4	不用	卡状态[31: 1]

总是先收到卡状态的最高位，SDIO_RSP4 寄存器的最低位始终为 0。

26.4.7 SDIO数据定时器寄存器 (SDIO_DTTMR)

SDIO 数据定时器寄存器 (SDIO_DTTMR) 包含以卡总线时钟周期为单位的数据超时时间。

一个计数器从 SDIO 数据定时器寄存器 (SDIO_DTTMR) 加载数值，并在数据通道状态机 (DCSM) 进入 Wait_R 或繁忙状态时进行递减计数，当 DCSM 处在这些状态时，如果计数器减为 0，则设置超时标志。

域	简称	复位值	类型	功能
位 31: 0	TIMEOUT	0x0000 0000	rw	数据超时时间 (Data timeout period) 以卡总线时钟周期为单位的数据超时时间。

注意：在写入 SDIO 数据控制寄存器 (SDIO_DTCTRL) 进行数据传输之前，必须先写入 SDIO 数据计数器寄存器 (SDIO_DTCNTR) 和 SDIO 数据长度寄存器 (SDIO_DTLEN)。

26.4.8 SDIO数据长度寄存器 (SDIO_DTLEN)

SDIO 数据长度寄存器 (SDIO_DTLEN) 包含需要传输的数据字节长度。当数据传输开始时，这个数值被加载到数据计数器中。

域	简称	复位值	类型	功能
位 31: 25	保留	0x00	resd	保持默认值。
位 24: 0	DTLEN	0x0000 000	rw	数据长度 (Data length value) 要传输的数据字节数目。

注意：对于块数据传输，SDIO 数据长度寄存器 (SDIO_DTLEN) 中的数值必须是数据块长度的倍数。在写入 SDIO 数据控制寄存器 (SDIO_DTCTRL) 进行数据传输之前，必须先写入 SDIO 数据定时器寄存器 (SDIO_DTTMR) 和 SDIO 数据长度寄存器 (SDIO_DTLEN)。

26.4.9 SDIO数据控制寄存器 (SDIO_DTCTRL)

SDIO 数据控制寄存器 (SDIO_DTCTRL) 控制数据通道状态机 (DCSM)。

域	简称	复位值	类型	功能
位 31: 12	保留	0x00000	resd	保持默认值。
位 11	IOEN	0x0	rw	SD I/O 使能功能 (SD I/O enable functions) 由软件置起或清零。如果该位被置起，则 DCSM 执行 SD I/O 卡特定的操作。 0: 关闭; 1: 启开。
位 10	RDWTMODE	0x0	rw	读等待模式 (Read wait mode) 由软件置起或清零。关闭表示使用 SDIO_D2 控制读等待，开始表示使用 SDIO_CK 控制读等待。 0: 关闭; 1: 启开。
位 9	RDWTSTOP	0x0	rw	读等待停止 (Read wait stop) 由软件置起或清零。如果设置了 RDWTSTART，关闭表示执行读等待，开启表示关闭读等待。 0: 关闭; 1: 启开。
位 8	RDWTSTART	0x0	rw	读等待开始 (Read wait start) 由软件置起或清零。关闭表示无动作，开启表示开始读等待。 0: 关闭; 1: 启开。
位 7: 4	BLKSIZE	0x0	rw	数据块长度 (Data block size) 由软件置起或清零。当选择了块数据传输模式，该域定义数据块长度。 0000: 块长度= $2^0=1$ 字节; 0001: 块长度= $2^1=2$ 字节; 0010: 块长度= $2^2=4$ 字节; 0011: 块长度= $2^3=8$ 字节; 0100: 块长度= $2^4=16$ 字节; 0101: 块长度= $2^5=32$ 字节; 0110: 块长度= $2^6=64$ 字节; 0111: 块长度= $2^7=128$ 字节; 1000: 块长度= $2^8=256$ 字节; 1001: 块长度= $2^9=512$ 字节; 1010: 块长度= $2^{10}=1024$ 字节; 1011: 块长度= $2^{11}=2048$ 字节; 1100: 块长度= $2^{12}=4096$ 字节; 1101: 块长度= $2^{13}=8192$ 字节; 1110: 块长度= $2^{14}=16384$ 字节; 1111: 保留。
位 3	DMAEN	0x0	rw	DMA 使能位 (DMA enable bit) 由软件置起或清零。 0: 关闭; 1: 启开。
位 2	TFRMODE	0x0	rw	数据传输模式 (Data transfer mode selection) 由软件置起或清零。关闭表示块数据传输，开启表示流数据传输。 0: 关闭; 1: 启开。
位 1	TFRDIR	0x0	rw	数据传输方向 (Data transfer direction selection) 由软件置起或清零。关闭表示控制器至卡，开启表示卡至控制器。 0: 关闭; 1: 启开。
位 0	TFREN	0x0	rw	数据传输使能位 (Data transfer enabled bit) 由软件置起或清零。如果设置该位为 1，则开始数据传输。根据 TFRDIR 方向位，DCSM 进入 Wait_S 或

Wait_R 状态，如果在传输的一开始就设置了 RDWTSTART 位，则 DCSM 进入读等待状态。不需要在数据传输结束后清除使能位，但必须更新 SDIO_DTCTRL 以允许新的数据传输。

- 0: 关闭；
- 1: 开启。

注意：写数据后的 7 个 HCLK 时钟周期内不能写入这个寄存器。

26.4.10 SDIO数据计数器寄存器（SDIO_DTCNTR）

当 DCSM 从空闲状态进入 Wait_R 或 Wait_S 状态时，SDIO 数据计数器寄存器（SDIO_DTCNTR）从 SDIO 数据长度寄存器（SDIO_DTLEN）加载数值，在数据传输过程中，该计数器的数值递减直到减为 0，然后 DCSM 进入空闲状态并设置数据状态结束标志 DTCMPL。

域	简称	复位值	类型	功能
位 31: 25	保留	0x00	resd	保持默认值。
位 24: 0	CNT	0x0000000	ro	数据计数数值（Data count value） 读这个寄存器时返回待传输的数据字节数，写这个寄存器无作用。

注意：只能在数据传输结束时读这个寄存器。

26.4.11 SDIO状态寄存器（SDIO_STS）

SDIO_STS 是一个只读寄存器，它包含两类标志：

- 静态标志（位[23: 22、10: 0]）：写入清除中断寄存器（SDIO_INTCLR），可以清除这些位。
- 动态标志（位[21: 11]）：这些位的状态变化根据它们对应的那部分逻辑而变化（例如：BUF 满和空标志变高或变低随 BUF 的数据写入变化）。

域	简称	复位值	类型	功能
位 31: 23	保留	0x000	resd	保持默认值。
位 22	IOIF	0x0	ro	收到 SD I/O 接口中断（SD I/O interrupt received）
位 21	RXBUF	0x0	ro	在接收 BUF 中的数据可用（Data available in receive BUF）
位 20	TXBUF	0x0	ro	在发送 BUF 中的数据可用（Data available in transmit BUF）
位 19	RXBUFE	0x0	ro	接收 BUF 空（Receive BUF empty）
位 18	TXBUFE	0x0	ro	发送 BUF 空（Transmit BUF empty） 若使用了硬件流控制，当 BUF 包含 2 个字时，TXBUF_E 信号变为有效。
位 17	RXBUFF	0x0	ro	接收 BUF 满（Receive BUF full） 若使用了硬件流控制，当 BUF 还差 2 个字满时，RXBUF_F 信号变为有效。
位 16	TXBUFF	0x0	ro	发送 BUF 满（Transmit BUF full）
位 15	RXBUFH	0x0	ro	接收 BUF 半满（Receive BUF half full）：BUF 中至少还有 8 个字，该标志位可以作为 DMA 请求。
位 14	TXBUFH	0x0	ro	发送 BUF 半空（Transmit BUF half empty）：BUF 中至少还可以写入 8 个字，该标志位可以作为 DMA 请求。
位 13	DORX	0x0	ro	正在接收数据（Data receive in progress）
位 12	DOTX	0x0	ro	正在发送数据（Data transmit in progress）
位 11	DOCMD	0x0	ro	正在传输命令（Command transfer in progress）
位 10	DTBLKCMPL	0x0	ro	已发送/接收数据块（CRC 检测成功）（Data block sent/received (CRC check passed)）
位 9	SBITERR	0x0	ro	在宽总线模式，没有在所有数据信号上检测到起始位（Start bit not detected on all data signals in wide bus mode）
位 8	DTCMPL	0x0	ro	数据结束（数据计数器，SDIO_DTCNTR=0）（Data end (data counter, SDIO CNT, is zero)）
位 7	CMDCMPL	0x0	ro	命令已发送（不需要响应）（Command sent (no response required)）

位 6	CMDRSPCMPL	0x0	ro	已接收到响应 (CRC 检测成功) (Command response)
位 5	RXERRO	0x0	ro	接收 BUF 上溢错误 (Received BUF overrun error)
位 4	TXERRU	0x0	ro	发送 BUF 下溢错误 (Transmit BUF underrun error)
位 3	DTTIMEOUT	0x0	ro	数据超时 (Data timeout)
位 2	CMDTIMEOUT	0x0	ro	命令响应超时 (Command response timeout) 命令超时时间是一个固定的值, 为 64 个 SDIO_CK 时钟周期。
位 1	DTFAIL	0x0	ro	已发送/接收数据块 (CRC 检测失败) (Data block sent/received)
位 0	CMDFAIL	0x0	ro	已收到命令响应 (CRC 检测失败) (Command response received)

26.4.12 SDIO清除中断寄存器 (SDIO_INTCLR)

清除中断寄存器 (SDIO_INTCLR) 是一个只写寄存器, 在对应寄存器位写'1'将清除 SDIO 状态寄存器 (SDIO_STS) 中的对应位。

域	简称	复位值	类型	功能
位 31: 23	保留	0x000	resd	保持默认值。
位 22	IOIF	0x0	rw	SD I/O 接口标志清除位 (SD I/O interface flag clear bit) 由软件置起以清除 IOIF 标志。
位 21: 11	保留	0x000	resd	保持默认值。
位 10	DTBLKCMPL	0x0	rw	DTBLKCMPL 标志清除位 (DTBLKCMPL flag clear bit) 由软件置起以清除 DTBLKCMPL 标志。
位 9	SBITERR	0x0	rw	SBITERR 标志清除位 (SBITERR flag clear bit) 由软件置起以清除 SBITERR 标志。
位 8	DTCMPL	0x0	rw	DTCMPL 标志清除位 (DTCMPL flag clear bit) 由软件置起以清除 DTCMPL 标志。
位 7	CMDCMPL	0x0	rw	CMDCMPL 标志清除位 (CMDCMPL flag clear bit) 由软件置起以清除 CMDCMPL 标志。
位 6	CMDRSPCMPL	0x0	rw	CMDRSPCMPL 标志清除位 (CMDRSPCMPL flag clear bit) 由软件置起以清除 CMDRSPCMPL 标志。
位 5	RXERRO	0x0	rw	RXERRO 标志清除位 (RXERRO flag clear bit) 由软件置起以清除 RXERRO 标志。
位 4	TXERRU	0x0	rw	TXERRU 标志清除位 (TXERRU flag clear bit) 由软件置起以清除 TXERRU 标志。
位 3	DTTIMEOUT	0x0	rw	DTTIMEOUT 标志清除位 (DTTIMEOUT flag clear bit) 由软件置起以清除 DTTIMEOUT 标志。
位 2	CMDTIMEOUT	0x0	rw	CMDTIMEOUT 标志清除位 (CMDTIMEOUT flag clear bit) 由软件置起以清除 CMDTIMEOUT 标志。
位 1	DTFAIL	0x0	rw	DTFAIL 标志清除位 (DTFAIL flag clear bit) 由软件置起以清除 DTFAIL 标志。
位 0	CMDFAIL	0x0	rw	CMDFAIL 标志清除位。 (CMDFAIL flag clear bit) 由软件置起以清除 CMDFAIL 标志。

26.4.13 SDIO中断屏蔽寄存器 (SDIO_INTEN)

在对应位置'1', SDIO 中断屏蔽寄存器 (SDIO_INTEN) 决定哪一个状态位产生中断。

域	简称	复位值	类型	功能
位 31: 23	保留	0x000	resd	保持默认值。
位 22	IOIFIEN	0x0	rw	SD I/O 模式接收中断使能 (SD I/O mode received interrupt enable) 由软件置起或清零。开关 SD I/O 模式接收中断功能。 0: 关闭; 1: 开启。
位 21	RXBUFIEN	0x0	rw	接收 BUF 中的数据有效产生中断 (Data available in RxBUF interrupt enable) 由软件置起或清零。开关接收 BUF 中的数据有效中断。 0: 关闭; 1: 开启。
位 20	TXBUFIEN	0x0	rw	发送 BUF 中的数据有效产生中断 (Data available in TxBUF interrupt enable) 由软件置起或清零。开关发送 BUF 中的数据有效中断。 0: 关闭; 1: 开启。
位 19	RXBUFEIEN	0x0	rw	接收 BUF 空产生中断 (RxBUF empty interrupt enable) 由软件置起或清零。开关接收 BUF 空中断。 0: 关闭; 1: 开启。
位 18	TXBUFEIEN	0x0	rw	发送 BUF 空产生中断 (TxBUF empty interrupt enable) 由软件置起或清零。开关发送 BUF 空中断。 0: 关闭; 1: 开启。
位 17	RXBUFFIEN	0x0	rw	接收 BUF 满产生中断 (RxBUF full interrupt enable) 由软件置起或清零。开关接收 BUF 满中断。 0: 关闭; 1: 开启。
位 16	TXBUFFIEN	0x0	rw	发送 BUF 满产生中断 (TxBUF full interrupt enable) 由软件置起或清零。开关发送 BUF 满中断。 0: 关闭; 1: 开启。
位 15	RXBUFHien	0x0	rw	接收 BUF 半满产生中断 (RxBUF half full interrupt enable) 由软件置起或清零。开关接收 BUF 半满中断。 0: 关闭; 1: 开启。
位 14	TXBUFHien	0x0	rw	发送 BUF 半空产生中断 (TxBUF half empty interrupt enable) 由软件置起或清零。开关发送 BUF 半空中断。 0: 关闭; 1: 开启。
位 13	DORXIEN	0x0	rw	正在接收数据产生中断 (Data receive acting interrupt enable) 由软件置起或清零。开关正在接收数据中断。 0: 关闭; 1: 开启。
位 12	DOTXIEN	0x0	rw	正在发送数据产生中断 (Data transmit acting interrupt enable) 由软件置起或清零。开关正在发送数据中断。 0: 关闭; 1: 开启。
位 11	DOCMDIEN	0x0	rw	正在传输命令产生中断 (Command acting interrupt enable) 由软件置起或清零。开关正在传输命令中断。 0: 关闭; 1: 开启。

				数据块传输结束产生中断 (Data block end interrupt enable)
位 10	DTBLKCMPLIEN	0x0	rw	由软件置起或清零。开关数据块传输结束中断。 0: 关闭; 1: 开启。
位 9	SBITERRIEN	0x0	rw	起始位错误产生中断 (Start bit error interrupt enable) 由软件置起或清零。开关起始位错误中断。 0: 关闭; 1: 开启。
位 8	DTCMPLIEN	0x0	rw	数据传输结束产生中断 (Data end interrupt enable) 由软件置起或清零。开关数据传输结束中断。 0: 关闭; 1: 开启。
位 7	CMDCMPLIEN	0x0	rw	命令已发送产生中断 (Command sent interrupt enable) 由软件置起或清零。开关命令已发送中断。 0: 关闭; 1: 开启。
位 6	CMDRSPCMPLIEN	0x0	rw	接收到响应产生中断 (Command response received interrupt enable) 由软件置起或清零。开关接收到响应中断。 0: 关闭; 1: 开启。
位 5	RXERROIEN	0x0	rw	接收 BUF 上溢错误产生中断 (RxBUF overrun error interrupt enable) 由软件置起或清零。开关接收 BUF 上溢错误中断。 0: 关闭; 1: 开启。
位 4	TXERRUIEN	0x0	rw	发送 BUF 下溢错误产生中断 (TxBUF underrun error interrupt enable) 由软件置起或清零。开关发送 BUF 下溢错误中断。 0: 关闭; 1: 开启。
位 3	DTTIMEOUTIEN	0x0	rw	数据超时产生中断 (Data timeout interrupt enable) 由软件置起或清零。开关数据超时中断。 0: 关闭; 1: 开启。
位 2	CMDTIMEOUTIEN	0x0	rw	命令超时产生中断 (Command timeout interrupt enable) 由软件置起或清零。开关命令超时中断。 0: 关闭; 1: 开启。
位 1	DTFAILIEN	0x0	rw	数据块 CRC 检测失败产生中断 (Data CRC fail interrupt enable) 由软件置起或清零。开关数据块 CRC 检测失败中断。 0: 关闭; 1: 开启。
位 0	CMDFAILIEN	0x0	rw	命令 CRC 检测失败产生中断 (Command CRC fail interrupt enable) 由软件置起或清零。开关命令 CRC 检测失败中断。 0: 关闭; 1: 开启。

26.4.14 SDIOBUF计数器寄存器（SDIO_BUFCNTR）

SDIOBUF 计数器寄存器（SDIO_BUFCNTR）包含还未写入 BUF 或还未从 BUF 读出的数据字数目。当在 SDIO 数据控制寄存器（SDIO_DTCTRL）中设置了数据传输使能位 TFREN，并且 DCSM 处于空闲状态时，BUF 计数器从 SDIO 数据长度寄存器（SDIO_DTLEN）加载数值。如果数据长度未与字对齐（4 的倍数），则最后剩下的 1~3 个字节被当成一个字处理。

域	简称	复位值	类型	功能
位 31: 24	保留	0x00	resd	保持默认值。
位 23: 0	CNT	0x000000	ro	将要写入 BUF 或将要从 BUF 读出数据字的数目。

26.4.15 SDIO数据BUF寄存器（SDIO_BUF）

接收和发送 BUF 是一组可读或可写的 32 位宽的寄存器，它在连续的 32 个地址上包含 32 个寄存器，CPU 可以使用 BUF 读写多个操作数。

域	简称	复位值	类型	功能
位 31: 0	DT	0x0000 0000	rw	接收或发送 BUF 数据（Receive and transmit BUF data） BUF 数据占据 32 个 32 位的字，地址为： (SDIO 基址 + 0x80) 至 (SDIO 基址 + 0xFC)

27 以太网控制器 (EMAC)

仅 AT32F457 支持该模块，AT32F455/456 则不支持。

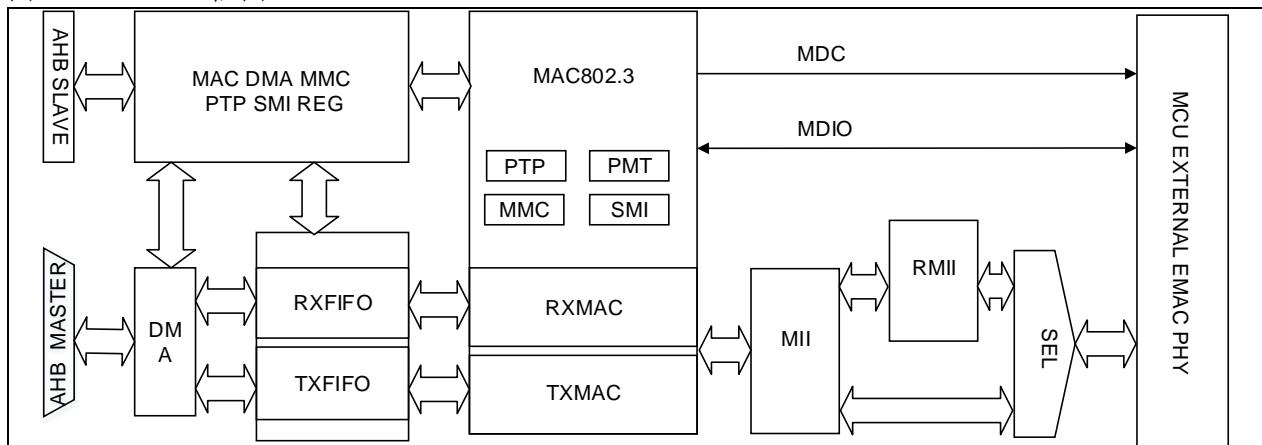
27.1 EMAC简介

AT32F457 的以太网模块支持通过以太网收发数据(10/100Mbps)，符合 IEEE 802.3-2002 标准。

AT32F457 以太网模块支持两种标准接口连接到外接的 PHY：IEEE 802.3 协议定义的独立于媒体的接口(MII)和简化的独立于媒体的接口(RMII)。

27.1.1 EMAC总体结构图

图 27-1 EMAC框图



27.1.2 EMAC主要特征

整个模块包括一个 EMAC CORE 和 DMA 控制器，帧的发送和接收通过 DMA 来调度。

DMA 特征

- AHB 突发传输的类型可通过软件配置
- 支持环结构或者链结构的描述符
- 每个描述符可以传输最高达 8K 字节的数据
- 发送或者接收通过轮询或者固定优先级的方式进行仲裁
- 多种工作状态下的对应的可配置中断
- 每一次传输都会有状态信息报告

EMAC CORE 特征

- 支持 10/100Mbps 的数据传输速率
- 通过兼容 IEEE 802.3 标准的 MII 接口，外接高速以太网 PHY
- 支持流控的全双工操作以及符合 CSMA/CD 协议的半双工操作
- 发送帧时可配置自动计算 CRC 和产生可控制的填充位
- 接收帧时可配置自动去除填充位/CRC
- 帧长度可配，最长为 16K 字节
- 帧间隙可配(40~96 位)
- 支持多种地址过滤模式以及混杂模式
- 支持检测接收到帧的 IEEE 802.1Q VLAN 标签
- 使用 RMON/MIB 计数器(RFC2819/RFC2665)进行强制网络统计
- 检测 LAN 远程唤醒帧和 AMD 的 Magic Packet™ 帧
- 支持对 IPv4 报头校验和以及对 IPv4 或 TCP、UDP 或 ICMP (IPv4 或 IPv6 数据格式封装) 的校验和检查
- 支持由 IEEE 1588-2008 标准定义的以太网帧时间戳，在帧的接收或发送状态中记录 64 位的时间戳。
- 2 个 2K 字节的 FIFO，一个用来发送，一个用来接收，阈值均可独立配置

- 可以滤掉接收到的错误帧，并在存储-转发模式下，不向应用程序转发错误的帧
- 对于 MAC 控制器的数据传输，支持存储-转发机制
- 在延迟冲突、冲突过多、顺延过多和下溢情况下丢弃帧
- 可通过软件清空发送 FIFO
- 在存储-转发模式下，可对发送帧计算并插入 IPv4 的报头校验和及 TCP、UDP 或 ICMP 的校验和
- MII 接口下支持 loopback 模式，方便调试定位
- IEEE 1588-2008 标准定义的精确时间协议，可以设置接收和发送帧的时间戳
- 支持粗调和精调两种校正方法
- 输出秒脉冲（可配置）
- 系统时间达到预定时间时会触发中断

27.2 EMAC模块功能详述

以太网模块由 MAC 802.3(媒体访问控制器)、独立于媒体的接口(MII)和一个专用的 DMA 控制器组成。主要实现以下功能：

- 数据传送和接收
 - 帧的组装(帧间隔和帧同步)
 - 源地址和目的地址管理
 - 错误检测
- 半双工模式下介质访问管理
 - 介质分配(防止冲突)
 - 冲突解决(处理冲突)

通常有两种模式可以操作 MAC 子层：

- 半双工模式：站点通过 CSMA/CD 算法来抢占对物理介质的访问，同一时间只有一个传输方向的两个站点可以进行信息传输
- 全双工模式：不使用 CSMA/CD 算法，但是全双工传输需要满足下面要求：
 - 物理介质支持同时进行收发操作
 - 只有两个站点接入 LAN
 - 接入 LAN 的两个站点都配置为全双工模式

27.2.1 EMAC通信接口介绍

EMAC 支持配置 PHY 的站点管理接口 SMI，以及用于以太网帧通信的媒体接口 MII 和精简的媒体接口 RMII。

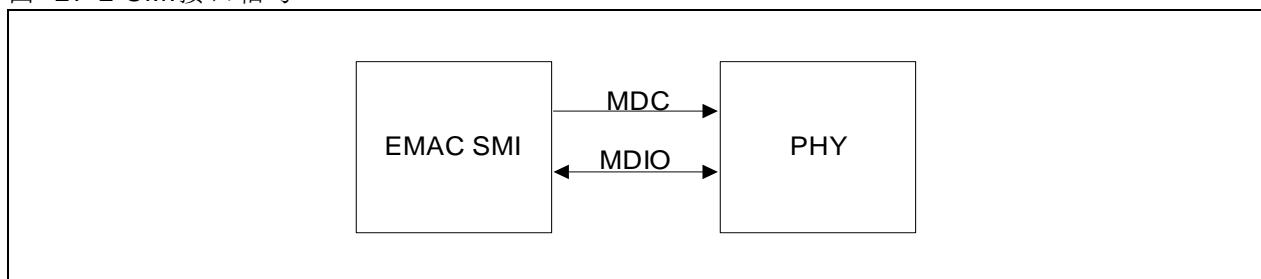
站点管理接口(SMI)

PHY 管理接口(SMI)通过时钟和数据两根线来访问 PHY 寄存器。可以支持最多 32 个 PHY。

MDC：PHY 配置时钟信号，最高频率为 2.5MHz。时钟高电平和低电平的最小维持时间为 160ns，最小周期为 400ns。在空闲状态下 MDC 时钟信号保持在低电平状态。

MDIO：双向口，数据的输入/输出线。

图 27-2 SMI 接口信号



执行写操作前需要先配置 PHY 地址和 MII 寄存器以及以太网 MAC MII 数据寄存器(EMAC_MACMIIDT)，然后设置 MII MW 和 MB 位，SMI 接口会向 PHY 传送 PHY 地址和 PHY 寄存器地址，然后传输数据。传输过程中 MB 位始终为 1，当完成写操作时，SMI 接口将清除 MB 位。需要注意的是，在传输过程中，以太网 MAC MII 地址寄存器(EMAC_MACMIIADDR)和以太网 MAC MII 数据寄存器(EMAC_MACMIIDT)信息是不能被更改的。

执行读操作前需要配置 PHY 地址和 MII 寄存器，然后配置以太网 MAC MII 地址寄存器(EMAC_MACMIIADDR) MB 为 1，MW 位为 0，

SMI 接口则发送 PHY 地址和 PHY 寄存器地址，开始读 PHY 寄存器数据。传输过程中 MB 位始终为 1，当完成读操作时，SMI 接口将清除 MB 位。需要注意的是，在传输过程中，以太网 MAC MII 地址寄存器(EMAC_MACMIIADDR)和以太网 MAC MII 数据寄存器(EMAC_MACMIIDT)信息是不能被更改的（不能被应用程序写，EMAC_MACMIIDT 在传输完成后会被自动更新为 SMI 读回的值）。

SMI 接口的时钟源由 AHB 时钟分频得到。根据 AHB 时钟频率的不同，需要选择不同的分频系数。注意必须要配置合适的分频系数，因为 MDC 的频率不能大于 2.5MHz。

下表列出了这个时钟范围。

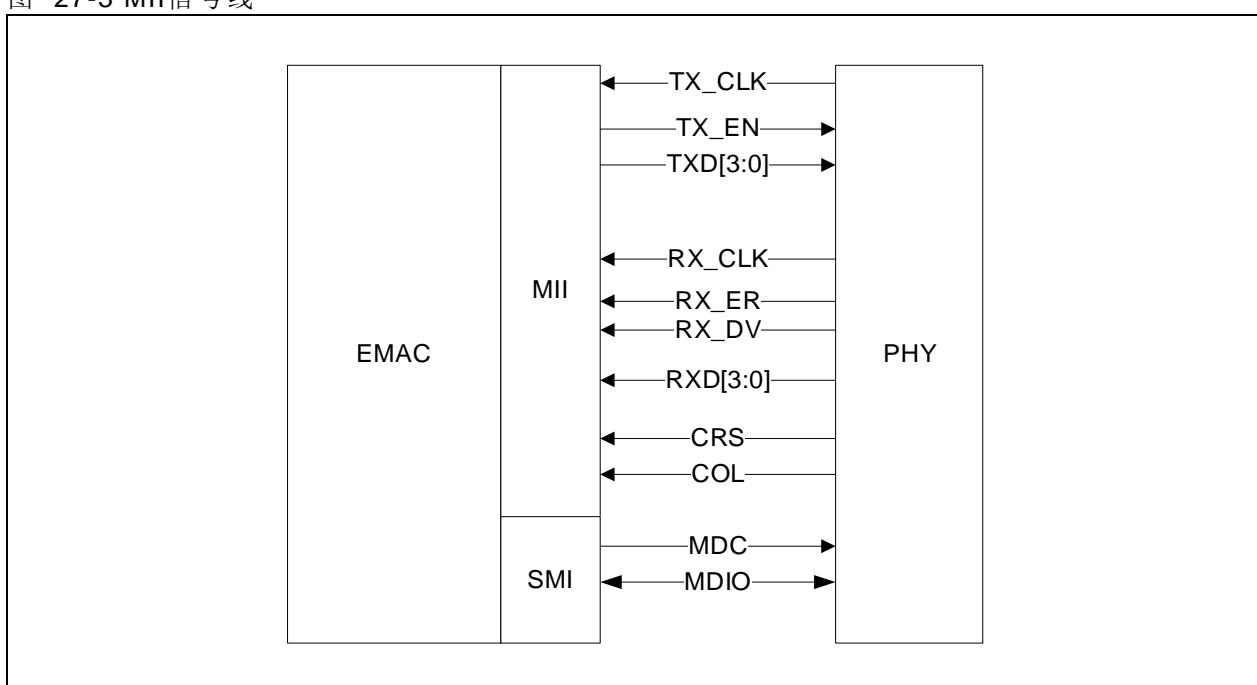
表 27-1 时钟范围

选择位	AHB 时钟	MDC 时钟
0000	60~100MHz	AHB 时钟/42
0001	100~150MHz	AHB 时钟/62
0010	20~35MHz	AHB 时钟/16
0011	35~60MHz	AHB 时钟/26
0100	150~192MHz	AHB 时钟/102
0101,0110,0111	保留	—

媒体接口 MII

独立媒体接口(MII)用于 MAC 子层和 PHY 之间的互联，允许 10M 位/s 和 100M 位/s 数据传输。

图 27-3 MII 信号线



MII_TX_CLK: 发送数据时钟信号, 对于 10Mbps 的数据传输, 此时钟为 2.5MHz, 对于 100Mbps 的数据传输, 此时钟为 25MHz

MII_RX_CLK: 接收数据时钟信号, 对于 10Mbps 的数据传输, 此时钟为 2.5MHz, 对于 100Mbps 的数据传输, 此时钟为 25MHz

MII_TX_EN: 发送使能信号, 此信号必需与数据前导符的起始位同步出现, 并且必需一直保持到所有需要传输的位都传输完为止

MII_TXD[3: 0]: 发送数据, 每次同步地传输 4 位数据, 数据在 MII_TX_EN 信号有效时有效。MII_TXD[0] 是数据的最低位, MII_TXD[3] 是最高位。

MII_CRS: CSMA/CD 定义的载波侦听信号, 由 PHY 控制, 只在半双工模式下有意义。当发送或接收的介质非空闲时, 此信号有效。PHY 必需保证 MII_CRS 信号在发生冲突的整个时间段内都保持有效。不需要此信号与发送/接收的时钟同步。

MII_COL: CSMA/CD 定义的冲突检测信号, 由 PHY 控制, 只在半双工模式下有意义。当检测到介质发生冲突时, 使能此信号, 并且在整个冲突的持续时间内, 保持此信号有效。此信号不需要和发送/接收的时钟同步。

MII_RXD[3: 0]: 由 PHY 控制, 每次同步地发送 4 位需要接收的数据, 数据在 MII_RX_DV 信号有效时有效。MII_RXD[0] 是数据的最低位, MII_RXD[3] 是最高位。当 MII_RX_DV 无效而 MII_RX_ER 有效时, PHY 会传送一组特殊的 MII_RXD[3: 0] 数据来表示一些信息。

MII_RX_DV: 接收数据有效信号, 由 PHY 控制, 当 PHY 准备好数据供 MII 接收时, 该信号有效。此信号必需和帧数据的首位同步(MII_RX_CLK)出现, 并在数据完全传输完毕之前, 都保持有效。在传送最后 4 位数据后的第一个时钟之前, 此信号必需变为无效状态。为了正确的接收一个帧, MII_RX_DV 信号必需在整个帧传输期间内都保持有效, 有效电平不能晚于数据线上的 SFD 位

MII_RX_ER: 接收出错信号, 保持一个或多个时钟周期(MII_RX_CLK)的有效状态, 指示 MAC 在帧内检测到了错误。具体错误信息需配合 MII_RX_DV 的状态以及 MII_RXD[3: 0] 的数据来指示。

表 27-2 发送接口信号编码

MII_TX_EN	MII_TXD[3: 0]	描述
0	0000 到 1111	正常的帧间隔
1	0000 到 1111	正常的数据传输

表 27-3 接收接口信号编码

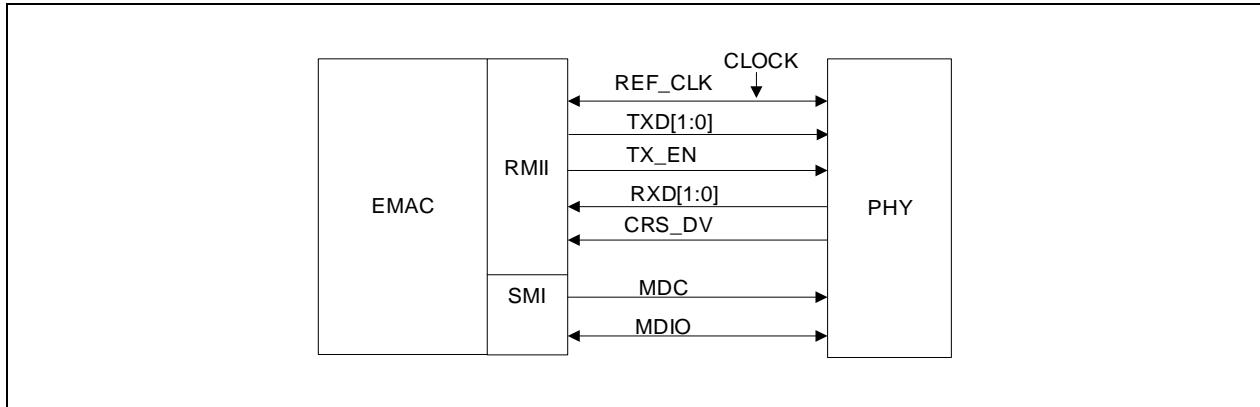
MII_RX_DV	MII_RX_ER	MII_RXD[3: 0]	描述
0	0	0000 到 1111	正常的帧间隔
0	1	0000	正常的帧间隔
0	1	0001 到 1101	保留
0	1	1110	失败的载波指示
0	1	1111	保留
1	0	0000 到 1111	正常的数据接收
1	1	0000 到 1111	数据接收出错

精简的媒体接口: RMII

精简的独立媒体接口(RMII)减少了 AT32F457xx 以太网模块和外部以太网之间的管脚数。根据 IEEE802.3u 标准, MII 接口需要 16 个数据和控制信号管脚, 而 RMII 则将管脚数减少到了 7 个(减少了 62.5% 的管脚数目)。

RMII 接口用于连接 EMAC 和 PHY, 该模块将 MAC 的 MII 信号转换到 RMII 接口上。

图 27-4 精简的独立于媒体的接口信号



MII/RMII 的选择及时钟源

通过 SCFG 配置寄存器 2 (SCFG_CFG2) 的第 23 位 MII_RMII_SEL 位，可以选择使用 MII 或者 RMII 模式。必需在以太网控制器处于复位状态时、或者在使能时钟前，选择好 MII/RMII 模式。

MII 时钟源

EMAC TX_CLK 和 RX_CLK 时钟信号由 PHY 提供，外接的 PHY 模块需要由外部的 25MHz 时钟驱动。这个 25MHz 时钟可以由晶振或者 MCU CLKOUT 脚提供。具体请参照 CRM 章节相关描述。

图 27-5 MII 时钟源 (CLKOUT 脚提供时钟)

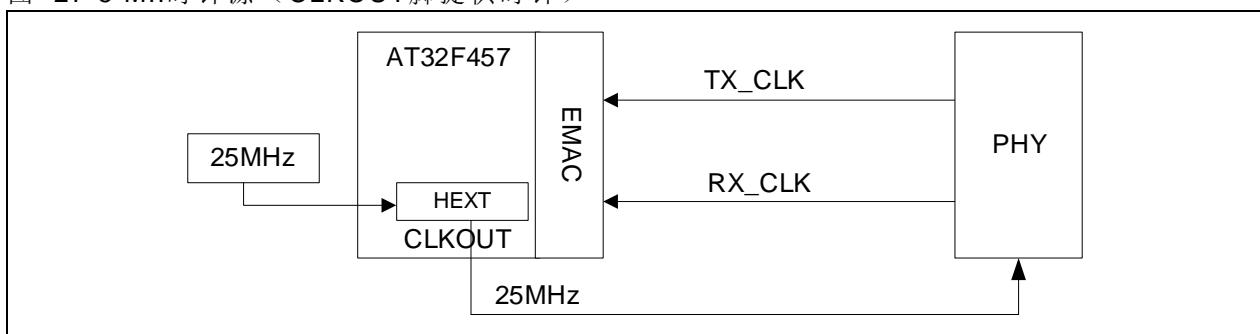
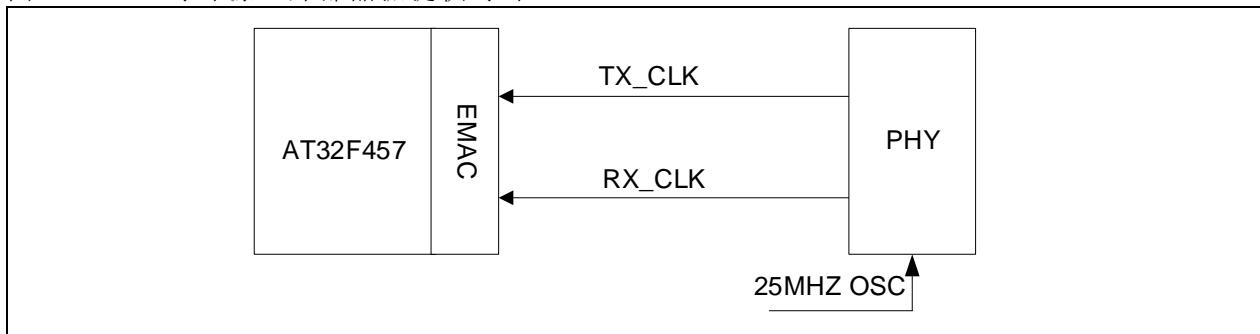


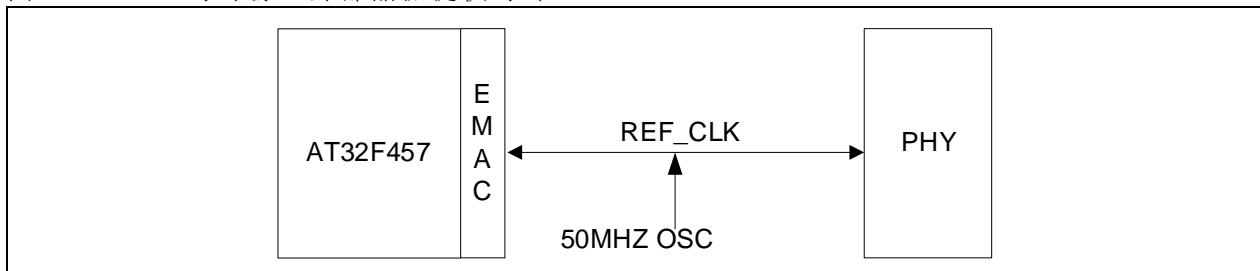
图 27-6 MII 时钟源 (外部晶振提供时钟)



RMII 时钟源

如下图所示，EMAC 和 PHY 都需要 50MHz 时钟源，可以使用外部晶体振荡器提供 50MHz 参考时钟。

图 27-7 RMII 时钟源 (外部晶振提供时钟)



EMAC 管脚分配及复用

表 27-4 以太网模块管脚配置

EMAC 信号类别	名称	管脚	管脚配置
SMI	MDC	PC1	复用功能
	MDIO	PA2	复用功能
	TX_CLK	PC3	复用功能
	TX_EN	PB11/ PG11	复用功能
	TXD3	PB8/PE2	复用功能
	TXD2	PC2	复用功能
	TXD1	PB13/PG14	复用功能
	TXD0	PB12/PG13	复用功能
	RX_CLK	PA1	复用功能
	RX_ER	PB10	复用功能
MII	RX_DV	PA7/PD8	复用功能
	RXD3	PB1/PD12	复用功能
	RXD2	PB0/PD11	复用功能
	RXD1	PC5/PD10	复用功能
	RXD0	PC4/PD9	复用功能
	CRS	PA0	复用功能
	COL	PA3	复用功能
	REF_CLK	PA1	复用功能
	TXD1	PB13/PG14	复用功能
	TXD0	PB12/PG13	复用功能
RMII	TX_EN	PB11/PG11	复用功能
	RXD1	PC5/PD10	复用功能
	RXD0	PC4/PD9	复用功能
	CRS_DV	PA7/PD8	复用功能
	PPS	PPS_OUT	复用功能

27.2.2 EMAC帧通信**帧格式**

MAC 帧格式和带标签的 MAC 帧格式如下图所示(有关详细的 MAC 帧格式定义请见 IEEE 802.C-2002)。
图 27-7 MAC帧格式

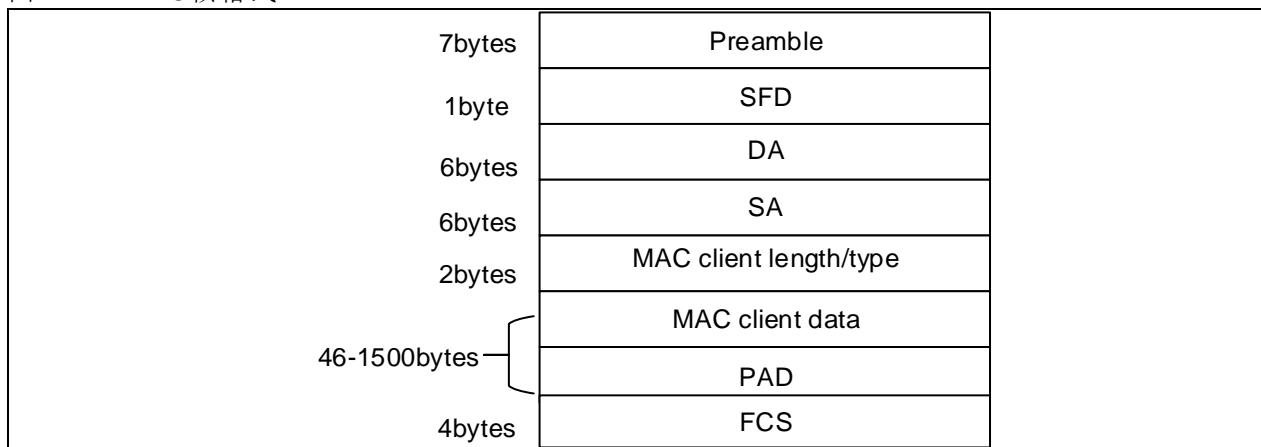
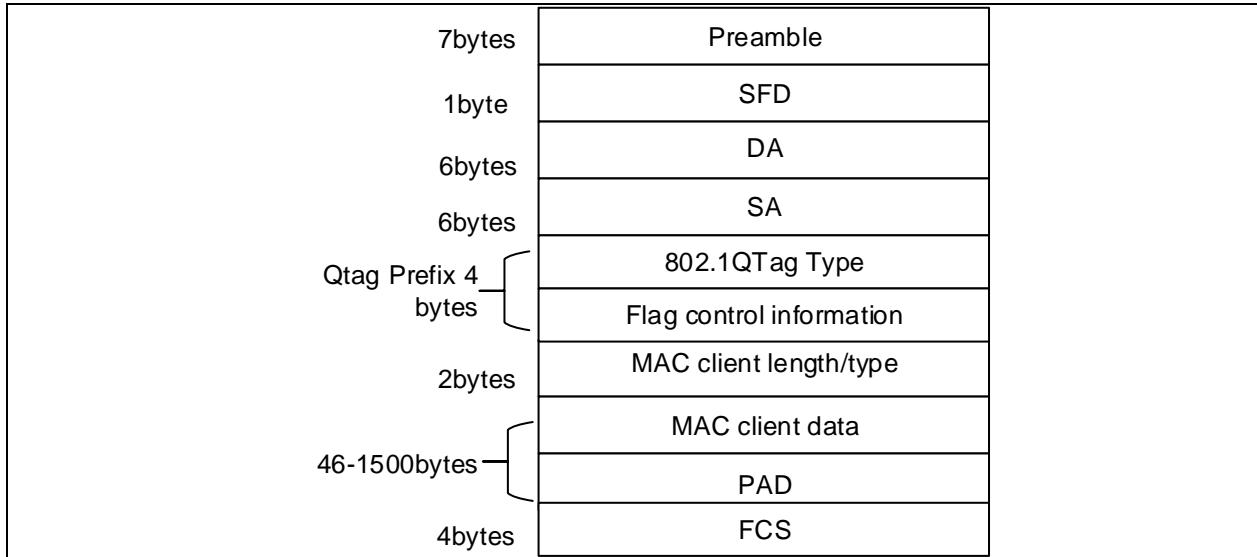


图 27-8 带标签的MAC帧格式



EMAC 帧过滤

EMAC 支持多种源地址和目的地址过滤。

地址过滤

利用 MAC 地址和多播 HASH 列表对接收到的帧进行地址过滤，EMAC 根据应用程序设定的帧过滤寄存器对目的地址和源地址进行检验，并报告相应的地址过滤结果。

单播目的地址滤波器

有完美地址过滤和 HASH 列表过滤两种方式。

- 1: 完美地址过滤：完美地址过滤必须配置帧过滤寄存器 HUC 位 0。有 4 组 MAC 地址可以用来进行完美地址过滤，MACADDR0 始终是使能的，MACADDR1, MACADDR2, MACADDR3 可以配置相应的 AE 位来使能。可以通过配置地址寄存器 MBC 位来屏蔽相应字节的地址比对，如果 MBC 位全为 0，则 EMAC 会被接收到的帧的 48 位单播地址与设定好的 MAC 地址逐位进行比较，必须全部匹配，才能通过过滤。
- 2: HASH 列表过滤模式：需配置帧过滤寄存器 HUC 位为 1。EMAC 利用 64 位的 HASH 列表对单播地址进行不完美过滤。HASH 过滤时，MAC 计算接收到帧的目的地址的 CRC 值(见下面的注释)，并取高 6 位作为索引检索 HASH 列表。CRC 值为'000000'对应 HASH 列表寄存器的位 0, CRC 值为'111111'对应 HASH 列表寄存器的位 63。如果 CRC 值对应的 HASH 列表上的相应位为' 1'，说明该帧能通过 HASH 过滤器，否则该帧不能通过 HASH 过滤器。

多播目的地址滤波器

- 1: 如果帧过滤寄存器 PMC 位为 1，则 MAC 可以接收所有的多播帧
- 2: 如果 PMC 位为 0, HMC 为 0，利用地址 MACADDR0, 1, 2, 3 进行完美地址过滤
- 3: 如果 PMC 位为 0, HMC 位为 1，利用 64 位 HASH 列表进行不完美过滤

广播地址滤波器

在帧过滤寄存器的 DBF 位为 1 的情况下，EMAC 拒收所有广播帧，DBF 为 0 时，EMAC 接收任何广播帧。

单播源地址过滤器

设置 MAC 地址寄存器 1,2,3 的位 30 为' 1'，则在接收帧时过滤器将会比较源地址而不是目的地址。如果帧过滤寄存器的 SAF 位为' 1'，EMAC 会丢弃没能通过源地址(SA)过滤器的帧；而且在 SAF 位为' 1' 时，只有源地址过滤和目的地址过滤都通过，帧才会转发给应用程序。否则，帧将会被丢弃。

颠倒过滤操作

通过设置帧过滤寄存器 DAIF 位和 SAIF 位为 1，可以对目的地址过滤和源地址过滤的输出结果进行颠倒。DAIF 位作用于单播和多播帧的目的地址，SAIF 作用于单播帧的源地址。

表 27-5 目的地址过滤器结果列表

帧类型	PR	HPF	HUC	DAIF	HMC	PMC	DBF	目的地址(DA)过滤器操作
广播帧	1	X	X	X	X	X	X	通过
	0	X	X	X	X	X	0	通过
	0	X	X	X	X	X	1	不通过
单播帧	1	X	X	X	X	X	X	所有帧通过
	0	X	0	0	X	X	X	完美/组过滤器匹配时通过
	0	X	0	1	X	X	X	完美/组过滤器匹配时不通过
	0	0	1	0	X	X	X	HASH 过滤器匹配时通过
	0	0	1	1	X	X	X	HASH 过滤器匹配时不通过
	0	1	1	0	X	X	X	HASH 或者完美/组过滤器匹配时通过
	0	1	1	1	X	X	X	HASH 或者完美/组过滤器匹配时不通过
多播帧	1	X	X	X	X	X	X	通过所有帧
	X	X	X	X	X	1	X	通过所有帧
	0	X	X	0	0	0	X	完美/组过滤器匹配时通过, 如果 PCF=0x, 丢弃暂停帧
	0	0	X	0	1	0	X	HASH 过滤器匹配时通过, 如果 PCF = 0x, 丢弃暂停帧
	0	1	X	0	1	0	X	HASH 或者完美/组过滤器匹配时通过, 如果 PCF = 0x, 丢弃暂停帧
	0	X	X	1	0	0	X	完美/组过滤器匹配时不通过, 如果 PCF = 0x, 丢弃暂停帧
	0	0	X	1	1	0	X	HASH 过滤器匹配时不通过, 如果 PCF = 0x, 丢弃暂停帧
	0	1	X	1	1	0	X	HASH 或者完美/组过滤器匹配时不通过, 如果 PCF = 0x, 丢弃暂停帧

表 27-6 源地址过滤器结果列表

帧类型	PR	SAIF	SAF	源(SA)过滤器操作
单播帧	1	X	X	所有帧通过
	0	0	0	传送完美/组过滤器匹配状态, 但不丢弃不通过的帧
	0	1	0	传送完美/组过滤器不匹配状态, 但不丢弃帧
	0	0	1	完美/组过滤器匹配时通过, 丢弃不通过的帧
	0	1	1	完美/组过滤器匹配时不通过, 丢弃不通过的帧

EMAC 帧的发送

DMA 控制器和 MAC 控制器以太网帧的发送。应用程序发出发送指令后, DMA 将以太网帧从用户数据缓冲区(比如 SRAM)读出并存入 TXFIFO 中, 帧再弹出并传入 MAC 控制器, MAC 控制器通过 MII/RMII 接口传给外部的以太网 PHY 进而和外部的站点进行通信, 发送帧结束后, MAC 控制器会生成发送状态并返回到 DMA 控制器中。

当检测到 SOF 信号后, MAC 接收数据, 并开始传输数据至 MII/RMII。由于各种延迟因素, 比如帧间隙、发送前导码/起始定界符的时间、和半双工模式下由于 CSMA/CD 算法导致的回退等待等, 从应用程序启动传输, 到数据帧发送到接口的时间是不定的。当 EOF 信号传输到 MAC 控制器后, 控制器完成一次传输, 并生成发送状态到 DMA。

从发送 FIFO 弹出数据到 MAC 控制器的操作有两种模式:

- 直通模式：当 FIFO 中的数据达到了设置好的阈值时(或者在达到阈值之前写入了 EOF)，数据会弹出 FIFO 并送入到 MAC 控制器中。直通模式下的阈值可以通过以太网 DMA 工作模式寄存器(EMAC_DMAOPM)的 TTC 位来设置
- 存储一转发模式：只有当一个完整的帧写入 FIFO 之后，数据才会被送入 MAC 控制器。如果发送 FIFO 的长度小于要发送的以太网帧，那么在发送 FIFO 即将全满时，数据也会被送入到 MAC 控制器

可以通过设置 FTF 位即以太网 DMA 工作模式寄存器(EMAC_DMAOPM)的第 20 位来清空发送 FIFO 中的内容。异常情况下如果在帧数据传送到 MAC 控制器的过程中误设置了 FTF 位，FIFO 会被清空，发送过程会被强制终止。EMAC 发送器会标记发生了下溢事件，并生成相应的状态信息给 DMA 控制器。

自动计算 CRC 和填充字节

IEEE802.3 协议规定以太网帧的最小数据长度为 46 字节，如果发送帧的数据长度不够，EMAC 可以配置成自动填充（填充数据是 0），那么在发送短帧时，数据长度会被填充达到 46 字节。

在帧发送时，EMAC 控制器可以配置成自动添加 CRC 到帧校验序列，也可以配置成不添加 CRC。但是如果 EMAC 控制器配置为当帧(DA+SA+LT+数据)少于 60 字节，自动添加填充字节时，不管是否配置了自动添加 CRC，都会添加 CRC 值到填充过的帧结尾。

CSMA/CD 定义的冲突检测

冲突检测仅在半双工模式下会起作用，全双工模式下无需冲突检测（详细原因请查以太网协议）。

在 MII 模式下，如果在帧传送的任意时刻外部 PHY 检测到了(从帧头到 CRC 的末位)冲突，冲突信号会发送给 EMAC 控制器，EMAC 控制器会发送一个 0x5555 5555 的 32 位堵塞信号，通知 LAN 上其他站点发生了冲突。如果冲突发生在传送前导码期间，EMAC 控制器会完成前导码和起始定界符的发送，然后立刻发送堵塞信号。

Jabber 定时器

在一个 LAN 上有可能一个站点会长时间的占用 LAN，为了避免这种情况，可以使能 EMAC jabber 定时器（设置 EMAC_MACCTRL[20]位为 0）。使能后，如果应用程序试图发送超过 2048 字节的帧，EMAC 会停止发送。

帧间隙管理

EMAC 会在符合帧间隙和退让延迟的情况下启动传输。在两个传输的帧之间，MAC 在帧间隙（以太网 MAC 配置寄存器(EMAC_MACCTRL)的 IFG 位）内会保持空闲状态。一旦 MII 的载波信号消失，EMAC 即启动它的 IFG 计数器。如果一个帧早于配置好的 IFG 时间到达，则这个帧将会被延迟发送，直到达到帧间隙时间才能开始传输。一旦 MII 的载波信号消失，MAC 即启动它的 IFG 计数器。全双工模式下，配置好的 IFG 时间到时后，MAC 则使能传输。在半双工模式下，如果 IFG 时间配置为 96 个比特的时间，MAC 将遵循 IEEE 802.3 规范的 4.2.3.2.1 章节定义的规则。如果在整个 IFG 时间间隔的前三分之二(64 个比特时间)的时间段里检测到了载波，MAC 将复位 IFG 计数器。如果在 IFG 时间间隔的后三分之一的时间段里检测到了载波，MAC 将继续 IFG 计数，并在达到 IFG 间隔时间后使能传输。在半双工模式下，MAC 遵循截断二进制指数退让算法。

发送流控

全双工模式下 EMAC 通过 pulse 帧来实现发送流控。有两种情况 EMAC 可以通过发送暂停帧来请求发送端暂停发送以太网帧。

应用程序设置以太网 MAC 流控寄存器(EMAC_MACFCTRL)的 FCB 位为' 1' 时，EMAC 会产生并发送一个单独的 Pause 帧。这个 Pause 帧指定的暂停时间为以太网 MAC 流控寄存器(EMAC_MACFCTRL)中配置好的暂停时间值 PT。EMAC 可以取消之前剩余的暂停时间或者延长暂停时间，只需要再发送一个 pulse 帧即可（取消剩余的暂停时间需要设置 PT 为 0）

在使能流控时 (EMAC_MACFCTRL 位 ETF 为 1)，当 RXFIFO 满时，EMAC 会产生并传输一个 Pause 帧。如果在达到配置好的暂停时间极限(以太网 MAC 流控寄存器(EMAC_MACFCTRL)的 PLT)时，RXFIFO 仍然为满，MAC 会再传输一个 Pause 帧。如果接收 FIFO 一直为满，则将一直重复这个过程。当 RXFIFO 不为满而暂停时间仍未到时，MAC 会传输一个暂停时间为 0 的 Pause 帧，指示远程站点结束暂停，本地缓存区已经准备好接收新的数据帧

遇到冲突时重新发送

半双工模式下，在帧发往 MAC 的时候有可能在 MAC 线路上发生冲突事件。这样 MAC 有可能在 FIFO 接收完这帧数据前就尝试重新发送。这时，如果重新发送开始，就需要再次把数据从 FIFO 中弹出。但是在 FIFO 控制器把 96 字节的数据弹出供 MAC 发送以后，就会释放这些数据的空间，允许 DMA 往那里推进新的数据。这意味着如果帧发出的数据数超过该门限(96 字节)或者 MAC 控制器提示迟到冲突事件时，不能进行重新发送。

清空发送 FIFO 操作

EMAC 可以通过设置以太网 DMA 工作模式寄存器(EMAC_DMAOPM)的 FTF 位(位 20)来清空 TXFIFO。当配置 FTF 位后，EMAC 会立刻执行 TXFIFO 清空操作，复位相应的指针为初始状态，即使 TXFIFO 正在把一帧数据发送给 MAC 控制器。清空操作发生时 EMAC 会中止发送当前帧，标记下溢事件，生成状态信息(TDES0 位 1 和位 13)。在应用程序(DMA)接收到全部被清空帧的状态信息字以后，清空操作完成。随后以太网 DMA 工作模式寄存器(EMAC_DMAOPM)的 FTF 位被清'0'。这时，允许 TXFIFO 从应用程序(DMA)接收新帧。清空操作后，FIFO 会忽略所有不以 SOF 起始符开头的数据。

发送状态信息以及时间戳

EMAC 控制器把以太网帧发送出去以后，会把发送状态信息返回到应用程序。如果使能了 IEEE 1588 时钟功能，一个 64 位的时间戳会和发送状态信息一起写回到发送描述符中。

发送校验和模块

以太网被广泛应用在收发封装有 TCP 或者 UDP 数据的 IP 数据包，因此以太网控制器理所应当具有发送校验和的功能，支持计算校验和，并在发送时插入计算结果，以及在接收时检测校验和错误。下面将详细介绍。

注意：只有在 TXFIFO 设置成存储-转发模式的时候(以太网 DMA 工作模式寄存器(EMAC_DMAOPM)的 TSF 位为'1')才能使能此功能。在数据帧发送到 EMAC 控制器发送端前，必须保证发送 FIFO 的深度足够容纳一个完整的帧。如果 TXFIFO 的深度小于输入的以太网帧长度，即便在存储-转发模式下，校验和功能也是无效的，EMAC 只会计算和修改 IPv4 报头的校验和域。

想要了解 IPv4、TCP、UDP、ICMP、IPv6 和 ICMPv6 报头规范，请分别查阅 IETF 规范 RFC791、RFC 793、RFC 768、RFC 792、RFC 2460 和 RFC 4443。

IP 头校验和

如果以太网帧的类型域值为 0x0800 并且 IP 数据包的版本域值为 0x4，校验和模块就检测到 IPv4 数据包。EMAC 不管输入帧的校验和域内容，直接以计算结果取而代之。IPv6 的报头不包含校验和域，因此校验和模块不会改变 IPv6 报头的值，而是通过发送状态信息的 IP 报头错误状态标志位(TDES0 位 16)反映这个报头的校验和计算结果；在以太网类型域以及 IP 报头版本域取值不匹配，或者以太网帧数据数目不够 IP 报头的数据长度域时，该标志位被置为'1'。换而言之，该位在 IP 报头出现下列错误时被置为'1'：

- 对于 IPv4 数据包
 - 接收到的以太网类型域 0x0800，但 IP 报头的版本域不等于 0x4
 - IPv4 报头长度域取值小于 0x5(20 字节)
 - 帧的总长度小于 IPv4 报头长度域的值
- 对于 IPv6 数据包
 - 接收到的以太网类型域为 0x86DD，但 IP 报头的版本域不等于 0x6
 - 帧在完整接收 IPv6 报头(40 字节)或者扩展报头(扩展报头包含报头长度域)之前结束。即使校验和模块检测到这样的报头错误，如果以太网类型域提示载荷数据为 IPv4 类型，仍然会插入一个 IPv4 的报头校验和

TCP/UDP/ICMP 校验和

TCP/UDP/ICMP 校验和通过分析 IPv4 或 IPv6 报头(包括扩展报头)来判断封装的数据是 TCP 还是 UDP 或者 ICMP。

注意：1：对于非 TCP、UDP 或者 ICMP/ICMPv6 的有效数据，校验和功能无效，不改动帧的数据

2：对于不完整的 IP 帧(IPv4 或者 IPv6)，包含安全功能的 IP 帧(如验证报头或者封装有安全数据)，以及带路由报头的 IPv6 帧，校验和功能无效。

校验和模块会对 TCP、UDP 或者 ICMP 的数据进行计算，并插入报头的相应域。它有以下 2 种工作模

式：

1: 校验和计算不包括 TCP、UDP 或者 ICMPv6 的伪报头，假设报头的校验和值就是输入帧的校验和域的数值。校验和域的内容会被包含在校验和的计算中，并最终被计算的结果取代

2: 不管输入帧的校验和域的内容，校验和的计算包括 TCP、UDP 或者 ICMPv6 的伪报头数据，最终的计算结果将取代校验和域的原有内容

校验和计算完毕后，结果会记录到发送状态信息的校验和错误状态位(TDES0 位 12)。当发生下述情况时，数据校验和错误状态位置‘1’：

1: 在存储-转发模式下，在帧结尾没有写入 TXFIFO 的情况下，帧就被转发给 MAC 发射端

2: 接收到的数据包少于 IP 报头数据长度域指示的字节数目

如果数据包长度大于给出的数据长度，多余的数据会被当成填充字节而丢弃，而不会报告错误。如果检测到第一类错误，则不改变 TCP、UDP 或者 ICMP 报头。如果检测到第二类错误，仍然会把校验和计算结果插入报头的相应域。

EMAC 帧的接收

MAC 接收到的帧都会被送入 RXFIFO 中，然后由 DMA 通过 AHB 接口送出。接收有 2 种模式，分别是直通模式和存储-转发模式。

在直通模式(默认模式)下，如果 RXFIFO 接收到帧数据长度大于门限设定值(可通过以太网 DMA 工作模式寄存器(EMAC_DMAOPM)的 RTC 设置) 数据或者完整的帧，就开始从 RXFIFO 中弹出数据，并通知 DMA 接收。DMA 会持续从 RXFIFO 中弹出数据，直到传输完整个数据。RXFIFO 转发完 EOF 后，就会生成接收状态信息并将其发送给 DMA 控制器。

在存储-转发模式(通过以太网 DMA 工作模式寄存器(EMAC_DMAOPM)的 RSF 位设置)下，只有在 RXFIFO 接收到一个完整的帧后，DMA 才能将其读出。

如果 EMAC 设置成丢弃所有错误帧，那么在两种不同的接收模式下将会有不同的行为：

1: 存储-转发模式下 DMA 只会读出合法的帧，并转发给应用程序。

2: 直通模式下，因为在帧结尾才会收到错误状态信息，因此一部分错误的帧并没有被丢弃。

一旦 EMAC 在 MII 上检测到起始定界符就会启动接收过程。MAC 控制器在处理帧之前会剥离前导码和起始定界符。会通过过滤器检查帧的报头，并用 FCS 域核对帧的 CRC 值。如果帧没能通过地址滤波器，EMAC 控制器就会丢弃该帧。

如果使能了 IEEE1588 时间戳，会在检测到帧的起始定界符的时候记录下系统的当前时间。EMAC 会在接收完当前帧后把这个时间戳转交给应用程序。

如果接收到的“帧长度/类型”域的值小于 0x600，并且把 EMAC 配置成 CRC/填充自动剥离，那么 EMAC 会在向 RXFIFO 发送完“帧长度/类型”域规定数目的帧数据后，开始丢弃剩下的字节(包括 FCS 域)。如果长度/类型域的值大于或等于 0x600，不管自动 CRC 剥离选项怎么设置，MAC 都会把所有接收到的以太网帧数据发送到 RXFIFO。

EMAC 默认打开看门狗定时器，长度大于 2048 字节的帧(DA + SA + LT + 数据 + 填充 + FCS)会被切断。可以通过以太网 MAC 配置寄存器(EMAC_MACCTRL)看门狗使能位(WD)来关闭这项功能。不过如果帧的长度大于 16KB，即使关闭了看门狗定时器，EMAC 仍然会切断当前以太网帧，并报告一个看门狗超时事件。

接收校验和模块

通过设置以太网 MAC 配置寄存器(EMAC_MACCTRL)的 IPC 位，EMAC 会检测 IPv4 和 IPv6 数据帧的完整性。EMAC 通过以太网帧的类型域值来判别当前帧是 IPv4 帧还是 IPv6 帧。接收校验和模块会计算 IPv4 报头的校验和，检查它是否与 IPv4 报头的校验和域的内容匹配。对报头的校验和结果体现在接收描述符 RDES0 的第 7 位，有三种情况会导致 IP 报头错误为被置为 1：

1: 以太网类型域值指示的数据类型与 IP 报头版本域不匹配

2: 接收到的帧长少于 IPv4 报头长度域指示的长度

3: IPv4 或 IPv6 报头少于 20 字节

接收校验和模块还能识别 IP 数据包的数据类型是 TCP、UDP 还是 ICMP，并按照 TCP、UDP 或 ICMP 的规范计算它们的校验和。计算包括了 TCP/UDP/ICMPv6 伪报头的数据，数据的校验结果体现在接收描述符 RDES0 第 0 位。该位在下面两种情况下会被置 1：

1: 收到的 TCP、UDP 或者 ICMP 数据长度与 IP 报头给出的长度不符

2: 计算的校验和与 TCP、UDP 或者 ICMP 校验和域的值不相等

接收流控

在全双工模式下，MAC 能够检测 PAUSE 帧，并按照 PAUSE 帧中的参数，在一定时间内暂停发送数据。设置以太网 MAC 流控寄存器(EMAC_MACFCTRL)的 ERF 位，可以使能或者取消 PAUSE 帧检测功能。如果使能了接收流控功能，EMAC 就会对暂停帧进行解码；

在暂停期间，如果接收到另一个 PAUSE 时间为 0 的 PAUSE 帧，MAC 会清除 PAUSE 的时间，重新开始发送数据。如果不为 0，则帧中的暂停时间会立刻加载到暂停时间计数器中。

多个帧的接收处理

由于帧的状态信息紧随在帧数据之后。因此只要 RXFIFO 未满，就可以存放任意数量的帧。

接收状态信息字

在接收以太网帧结束时，EMAC 会把接收状态信息发给应用程序(DMA)。接收状态信息的具体含义详见 RDES0。

EMAC 自循环模式

EMAC 支持 loopback 模式：发射端把帧发送到自己的接收端上。loopback 模式可以非常方便的用来调试以太网通信，需要通过设置以太网 MAC 配置寄存器(EMAC_MACCTRL)的 LM 位置'1'来打开这个功能。注意 loopback 模式只针对 MII 接口。

EMAC 帧计数器

MMC 是收集发送和接收到帧的统计寄存器。主要包括以太网 MMC 控制寄存器(EMAC_MMCTRL)，以太网 MMC 接收中断寄存器(EMAC_MMCR1)和以太网 MMC 接收中断屏蔽寄存器(EMAC_MMCRIM)，以太网 MMC 发送中断寄存器(EMAC_MMCTI)和以太网 MMC 发送中断屏蔽寄存器(EMAC_MMCTIM)。

如果一个帧的发送过程因为发生错误被中止，则这个帧将不会被统计，错误主要有以下几种：

- 1: 没有载波或者载波丢失
- 2: Jabber 超时
- 3: 迟到冲突
- 4: 冲突过多
- 5: 顺延(Deferral)过多
- 6: 帧溢出

如果一个帧的接收过程因为发生错误被中止，则这个帧将不会被统计，错误主要有以下几种：

- 1: CRC 错误
- 2: 过短帧(少于 64 字节)
- 3: 对齐错误
- 4: 长度错误(长度域值域接收到帧长不符)
- 5: 超出范围(帧长超过上限，非标签帧最大为 1518 字节，标签帧最大为 1522 字节)
- 6: MII_RXER 输入错误

27.2.3 专用 DMA 对以太网帧的传输调度

EMAC 通过内部专用的 DMA 来调度以太网帧的发送和接收。

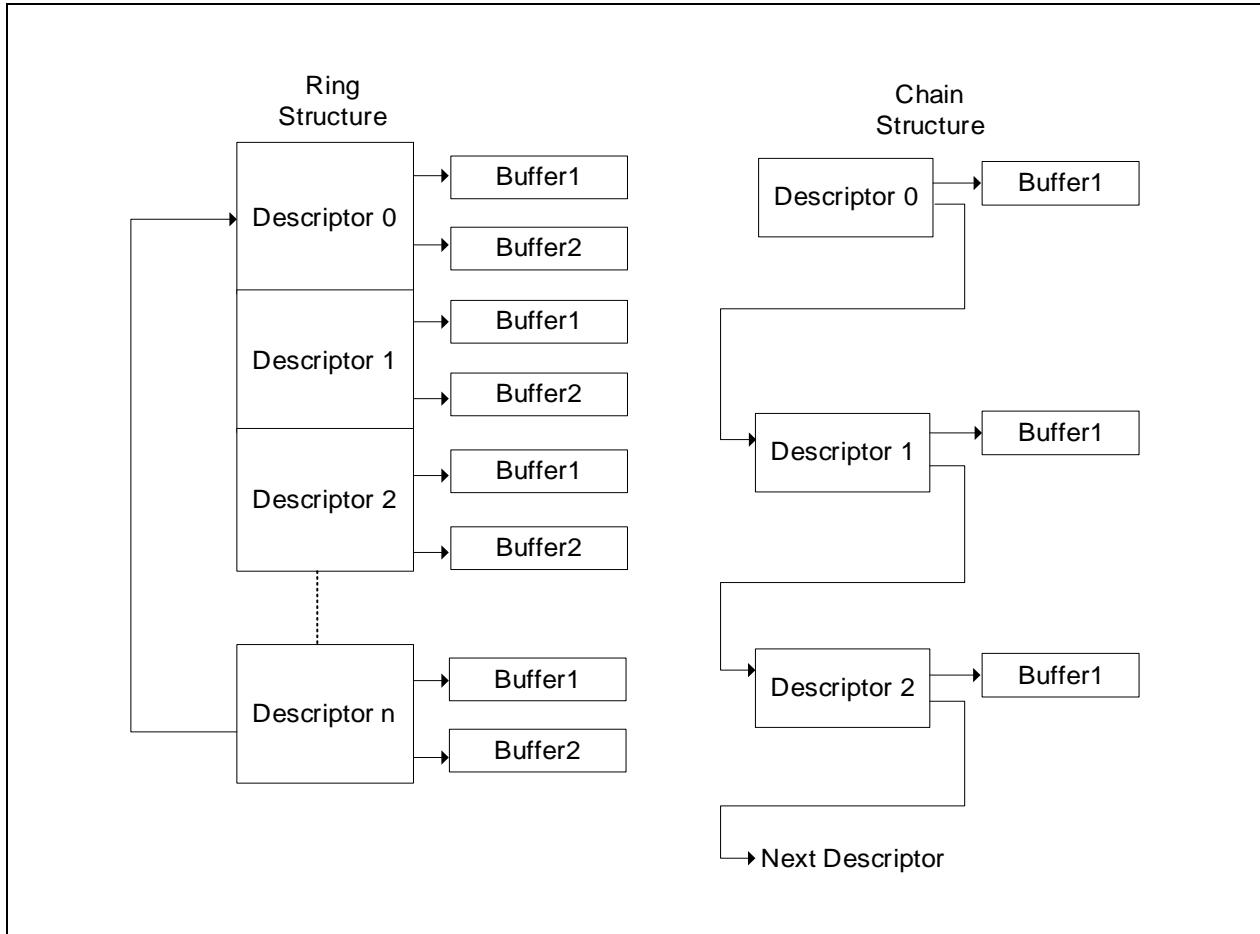
对于发送流程，DMA 会通过 AHB 主接口从用户数据缓冲区(比如 SRAM)读出以太网帧并送给 TXFIFO，EMAC 控制器根据配置把 TXFIFO 的数据成帧的发送到 MII/RMII 接口。

对于接收流程，EMAC 控制器会把从 MII/RMII 接口接收的以太网帧送到 RXFIFO，然后 DMA 会把 RXFIFO 中的以太网帧读出并通过 AHB 接口发送给用户数据缓冲区(比如 SRAM)。

DMA 通过 DMA 控制和状态寄存器以及描述符列表来统筹整个发送和接收流程。发送和接收流程均有对应的描述符列表。描述符列表一般存放在系统缓存中(SRAM)，DMA 在发送和接收启动时会通过发送和接收查询寄存器轮询描述符列表以开始发送和接收过程。发送和接收的描述符列表的基地址存放在发送描述符列表寄存器和接收描述符列表地址寄存器。

描述符有环结构和链结构两种形式。描述符组成环结构时，每个描述符可以指向两个帧的缓冲区。在链结构(对于发送需配置 TDES0[20]为 1，对于接收需配置 RDES1[14]为 1)下，每个描述符只能指向一个缓冲区，当前帧的 TDES3 和 RDES3 的内容代表下一个发送和接收描述符的地址。

图 27-9 环结构描述符和链结构描述符



DMA AHB 主机突发访问

当设置以太网 DMA 总线模式寄存器(EMAC_DMABM)的 FB 位为'1'时, DMA 会在 AHB 主接口上执行固定长度的突发访问。最大突发长度由 PBL 域(以太网 DMA 总线模式寄存器(EMAC_DMABM)位[13: 8])定义。对接收和发送描述符这 16 字节数据的访问总是以 PBL 限定的最大可能突发长度进行。

DMA 会向 AHB 主接口提供起始地址和要传送的字节数量, 然后开始一次传输。

注意, 对于发送流程还要满足下面条件之一:

- 1: TXFIFO 空间大于预设的突发长度
- 2: 帧结束前的数据字节数小于突发长度而且 TXFIFO 能容纳这些字节数

对于接收流程要满足下面条件之一:

- 1: RXFIFO 里的可用数据多于预设的突发长度
- 2: 帧结束之前的数据字节数小于突发长度且在接收 FIFO 里检测到帧结束

AHB 主机数据缓存对齐

DMA 发起传输的时候, 总是从与总线宽度对齐的地址开始, 但是缓存的起始地址可以对齐到 4 个字节中的任意一个。

- 读缓存示例: 如果发送缓存的地址为 0x2000 0AA3, 并需要传输 15 字节。开始读操作后 DMA 实际会从地址 0x2000 0AA0 开始读 5 个字(32 位), 但是在往 TXFIFO 发送数据的时候, 会忽略头 3 个字节和后 2 个字节。除非 DMA 传输的是帧的结尾, 否则 DMA 总是保证向 TXFIFO 传输 32 位的数据。
- 写缓存示例: 如果接收缓存的地址为 0x2000 0BB2, 并需要传输 16 字节长的帧。开始写操作后 DMA 实际会从地址 0x2000 0BB0 开始写 5 个 32 位数据。但是传输的头 2 个字节和末尾的 2 个字节是虚拟字节。

缓冲区大小计算

针对发送过程, 软件需要计算缓存的大小。TXDMA 会传输与 TDES1 缓存大小域相等数目的字节给 EMAC

控制器。如果当前描述符 TDES0 的 FS 位为' 1'，那么 DMA 就标记从这个缓存的第一次发送是帧首。如果当前描述符 TDES0 的 LS 位为' 1'，则 DMA 就标记从这个缓存的最后一次发送是帧尾。

接收帧的过程中，如果接收缓存地址是字对齐的，缓存的有效长度为 RDES1 中配置的值。如果接收缓存地址是非字对齐的，则缓存的有效长度将小于 RDES1 中配置的值。缓存有效长度值应为 RDES1 中配置的值减去缓存地址的低 2 位值。例如，假设缓存的总大小为 1024 字节，缓存地址为 0x2000 0001，地址的低 2 位值为 0x01，那么缓存有效长度为 1023 个字节。

当收到了一个帧起始 SOF，则 DMA 控制器将 FS 位置位，当收到一个帧结束 EOF 时，则 LS 位被置位。如果接收缓存长度域值配置的足够大，能放下整个帧，则 FS 和 LS 位将在同个描述符中被置位。实际接收的帧长度可从 RDES0 的 FL 位域获取。

DMA 仲裁器

有两种仲裁方式来仲裁 DMA 的发送和接收控制器：轮询和固定优先级。选择轮询仲裁方式时(以太网 DMA 总线模式寄存器(EMAC_DMABM)DA 位为' 0')，在发送和 RXDMA 同时要求访问 AHB 总线的时候，按照以太网 DMA 总线模式寄存器(EMAC_DMABM)PR 位设定的比例分配总线。在 DA 位为' 1' 时，RXDMA 总是比 TXDMA 对总线拥有更高的访问优先级。

DMA 错误响应

如果 DMA 在传输过程中接收到了错误的总线响应，那么 DMA 会停止所有操作，并更新以太网 DMA 状态寄存器(EMAC_DMASTS)的错误位和总线致命错误位。要想 DMA 重新工作，必须通过软件或者硬件复位以太网外设并重新初始化 DMA。

DMA 初始化

- 1: 在以太网 DMA 总线模式寄存器(EMAC_DMABM)中设置 AT32F457xx 总线访问相关参数。
- 2: 在以太网 DMA 中断使能寄存器(EMAC_DMAIE)中屏蔽不需要的中断源。
- 3: 应用程序生成发送和接收描述符列表并写入系统缓存，然后把描述符列表的起始地址写入以太网 DMA 接收描述符列表地址寄存器(EMAC_DMARDLADDR)和以太网 DMA 发送描述符列表地址寄存器(EMAC_DMATDLADDR)。
- 4: 配置地址过滤寄存器。
- 5: 在 MAC 的以太网 MAC 配置寄存器(EMAC_MACCTRL)中设置并使能发送和接收操作。根据自动协商(auto-negotiation)的结果，设置 PS 位和 DM 位的值。
- 6: 设置以太网 DMA 工作模式寄存器(EMAC_DMAOPM)的位 13 和位 1 为' 1'，开始发送和接收。
- 7: DMA 发送和接收控制器开始从对应的描述符列表里读取描述符并开始处理接收和发送操作。

TXDMA 操作：非 OSF 模式

在默认模式下，TXDMA 的工作流程如下：

1. 应用程序建立以太网帧数据缓存以及发送描述符列表(TDES0-TDES3)，并置 OWN 位(TDES0[31])为' 1'。
2. 设置 SSTC 位(以太网 DMA 工作模式寄存器(EMAC_DMAOPM)位[13])为' 1' 后，DMA 启动发送。
3. DMA 查询发送描述符来获取待发送的帧，如果 DMA 检测到标志位提示 CPU 正在操作描述符或者发生了错误，就会终止传输进入挂起状态，并设置发送缓存不可用位(以太网 DMA 状态寄存器(EMAC_DMASTS)位2)和正常中断总结位(以太网 DMA 状态寄存器(EMAC_DMASTS)位16)为' 1'。发送控制器操作跳至步骤8。
4. DMA 根据发送描述符的指示从 AT32F457xx 存储器里取数据，并把数据发送给 TXFIFO。
5. 如果以太网帧的数据存放在属于不同描述符的不同缓存里，DMA 会关闭存放中间数据的描述符，并取下一个缓存对应的描述符，重复步骤3、4、5 直到发送完帧结尾的数据。
6. 在帧发送完成以后，如果发送状态信息显示这个帧使能了 IEEE1588 时间戳功能，那么 DMA 在更新发送状态信息给 TDES0 的同时会把时间戳的值会写入存放帧尾的缓存对应的发送描述符(TDES2 和 TDES3)。然后把 OWN 位清' 0'，关闭当前描述符。如果这个帧没有打开时间戳功能，那么 DMA 只会更新状态信息到 TDES0，不会记录时间戳。
7. 在帧发送完成以后，如果描述符的完成中断位(TDES0[30])为' 1'，则发送中断位(以太网 DMA 状态寄存器(EMAC_DMASTS)位[0])会被置' 1'。然后 DMA 控制器返回步骤3，准备发送下一帧数据。

8. 在暂停状态下，如果DMA收到发送查询的请求并且溢出中断标志位为‘0’，DMA就会尝试重新获取描述符(因此会回到步骤3)。

TXDMA 操作：OSF 模式

此模式中需要设置 OSF 位（以太网 DMA 工作模式寄存器(EMAC_DMAOPM)位 2)为‘1’。在这种模式下，DMA 在发送完当前帧数据后，不必等状态信息写回就去直接查询第二帧数据的发送描述符。

1. 按照TXDMA默认模式的步骤1-6操作。
2. DMA不等前一帧最后一个描述符状态信息更新，直接取下一个描述符。
3. 如果DMA占有描述符，那么就解析发送缓存的地址。如果DMA不占有这个描述符，则进入挂起状态并跳到步骤7。
4. DMA从AT32F457xx的存储器中取出发送帧的数据并发送直到帧尾发送完毕，如果帧数据分散在多个缓存中，DMA关闭存放在中间数据缓存对应的描述符。
5. DMA等待前一帧的发送状态信息和时间戳，在接到状态信息后，如果标志位提示记录了帧的时间戳，DMA把时间戳写入TDES2和TDES3。DMA再清零OWN位关闭该描述符。如果这个帧没有打开时间戳功能，那么DMA不会更改TDES2和TDES3的内容。
6. 如果使能中断，发送中断位置‘1’，DMA在状态信息正常的情况下继续取下一个描述符，并跳到步骤3。如果前一个发送状态信息显示有数据下溢错误，DMA进入暂停状态，并跳到步骤7。
7. 在暂停状态下，如果DMA收到一个待处理的状态信息和时间戳，那么如果这个帧使能了时间戳，DMA就把时间戳写入TDES2和TDES3，然后把状态信息写入TDES0。随后，设置相关的中断标志位并回到暂停状态。
8. 只有在收到发送查询请求以太网 DMA 当前发送描述符寄存器(EMAC_DMACTD)后，DMA才会退出暂停状态并进入运行状态，并根据是否有待处理的状态信息跳到步骤1或者步骤2。

发送帧处理

发送缓存里存放的以太网帧数据必须包括目的地址，源地址，正确的类型/长度域值，以及数据。至于是否包括 CRC 值，要看发送描述符的设定。如果描述符要求 EMAC 控制器关闭 CRC 和插入填充功能，那么缓存里还应当包含 CRC。

一个帧可以存放在不同缓存里，用数据链的形式连接起来。传输开始时，第一个描述符的 TDES0 位 28 应为‘1’，DMA 则会把数据从内存送到 TXFIFO 中。如果 TDES0 位 29 是‘1’，表示这个缓存存放的是帧结尾的数据，是这个帧的最后一个缓存。在这最后一个缓存的数据发送完毕以后，DMA 写回发送状态信息到 TDES0。如果发送完成中断位(TDES0[30]为‘1’，则设置发送中断位(以太网 DMA 状态寄存器(EMAC_DMASTS)位 0) 为‘1’。DMA 随后取下一个描述符，重复上述步骤。实际的帧发送还需要看配置的是存储-转发模式或者是阈值模式。在 DMA 传输完帧以后，则会关闭描述符(TDES0[31]清‘0’)。

发送查询暂停

在下列情况下，可以暂停 DMA 查询描述符：

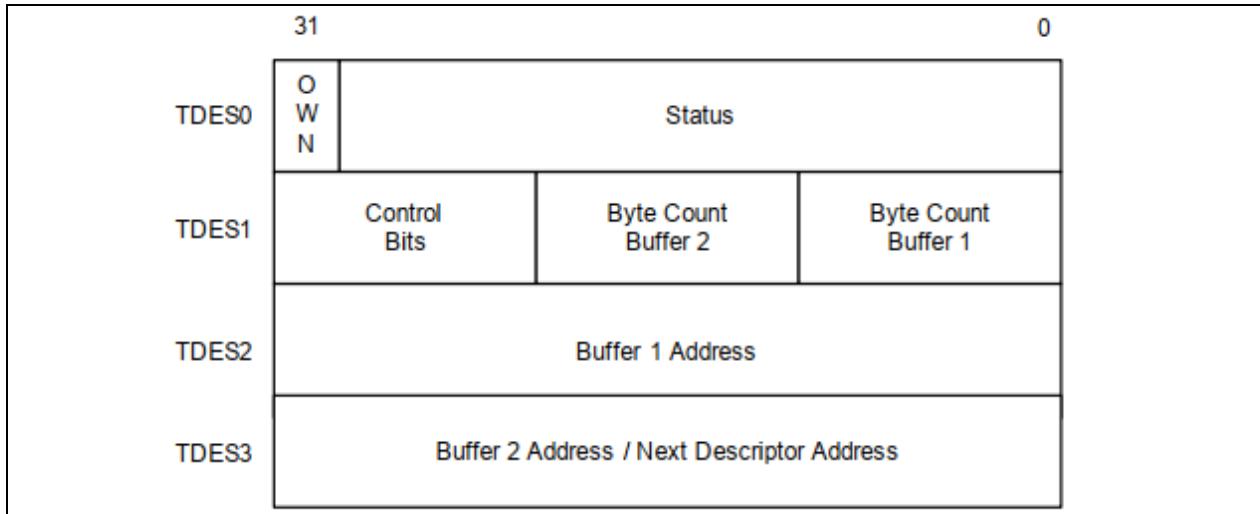
DMA 检测到 CPU 占有描述符(TDES0[31]=0)，DMA 会进入挂起状态。

在检测到数据下溢错误时，DMA 会中止帧的传输，并设置异常中断总结位(以太网 DMA 状态寄存器(EMAC_DMASTS)位 15)和发送数据下溢位 (以太网 DMA 状态寄存器(EMAC_DMASTS)位 5)为‘1’，并更新 TDES0 里的相应错误位为‘1’。

TXDMA 描述符

描述符结构体包含 4 个 32 位字，TDES0、TDES1、TDES2 和 TDES3 的位定义如下图：

图 27-10 传输描述符

**TDES0：发送描述符字 0**

在初始化描述符的时候需要编写控制位[30: 26]+位[23: 20]和 OWN 位。在 DMA 更新，或者说写回描述符的时候，会清除所有的控制位和 OWN 位，只报告状态信息标志位。

域	简称	类型	功能
位 31	OWN	rw	占有位 (Own bit) 0: CPU 占有描述符。 1: DMA 占有描述符。 DMA 在传输完整个帧或者这个缓存里的数据全部读出这两种情况下会把该位清'0'。每个帧的第一个缓存描述符的占有位，必须在后面缓存描述符的占有位全部置'1'以后，才能置'1'。
位 30	IC	rw	完成时中断 (Interrupt on completion) 配置为'1'时，在当前帧发送完成以后，会把传输中断位(以太网 DMA 状态寄存器 (EMAC_DMASTS) 位[0])置'1'。该位仅在 LS 位为 1 时有效。
位 29	LS	rw	最后分块 (Last segment) 置'1'时，表示当前描述符指向的缓存存放着帧的最后一个分块。此位为 1 时，TDES1 中 TBS1 或者 TBS2 的值不能为 0。
位 28	FS	rw	第一分块 (First segment) 置'1'时，表示当前描述符指向的缓存存放着帧的第一个分块。
位 27	DC	rw	不计算 CRC (Disable CRC) 置'1'时，MAC 不再在发送帧结束时插入 CRC 域。该位只有在 FS 位(TDES0[28])为'1'时才有效。
位 26	DP	rw	不填充 (Disable pad) 0: MAC 对帧长不足 64 字节的帧自动填充字节，而且无论 DC 位(TDES0[27])为何值 MAC 都会在帧结尾插入 CRC 数值。该位只在 FS 位(TDES0[28])为'1'时有效。 1: MAC 不对帧长不足 64 字节的帧自动填充字节。
位 25	TTSE	rw	发送时间戳使能 (Transmit time stamp enable) 置'1'时，这个描述符对应的发送帧会打开 IEEE1588 硬件时间戳功能。该位只在 TSE 位(EMAC_PTPTSCTRL[0])和 FS 位(TDES0[28])为'1'时有效。
位 24	保留	resd	请保持默认值。
位 23: 22	CIC	rw	校验和插入控制 (Checksum insertion control) 这 2 位控制校验和的计算和插入，详述如下： 00: 不使能插入校验和功能； 01: 仅使能 IP 报头的校验和计算和插入； 10: 使能 IP 报头和数据域的校验和计算和插入，但是不计算伪报头的校验和； 11: 使能 IP 报头和数据域的校验和计算和插入，同时也计算伪报头的校验和。
位 21	TER	rw	环形发送结束 (Transmit end of ring) 置'1'时，表示到达描述符列表的最后一个描述符。DMA 返回描述符列表的起始地址，形成描述符环。
位 20	TCH	rw	第二地址链表 (Second address chained) 置'1'时，表示 TDES1 中 TBS2 代表的是下一个描述符的地址，而不是第二个缓存的地址。TDES0[21]的功能优先于本位(TDES0[20])。该位只在 FS 位 (TDES0[28])为'1'时才有效。

位 19: 18	保留	resd	请保持默认值。
位 17	TTSS	rw	<p>发送时间戳状态 (Transmit time stamp status)</p> <p>此位作为状态位用来表明发送帧的时间戳是否被记录。置'1'时表示记录下了描述符对应的发送帧的时间戳，记录的时间戳放在 TDES2 和 TDES3 处。该位只在 LS 位 (TDES0[29]) 为'1'时才有效。</p>
位 16	IHE	rw	<p>IP 报头错误 (IP header error)</p> <p>该位置'1'时表示 MAC 发送端检测到了 IP 数据包报头的错误。对于 IPv4 数据包，MAC 会检查数据包中的长度域和接收到的 IPv4 数据字节数是否相等，如果不相等则报错。对 IPv6 数据包，如果主报头长度不是 40 字节时则报错。此外，IPv4 或者 IPv6 帧的长度/类型域值和报头的版本信息必须匹配。对 IPv4 帧，如果报头长度域值小于 0x5，该位也置'1'报错。</p>
位 15	ES	rw	<p>错误汇总 (Error summary)</p> <p>该位为下列位的逻辑“或”</p> <p>TDES0[14]: 嘞嗒(Jabber)超时； TDES0[13]: 帧清空； TDES0[11]: 载波丢失； TDES0[10]: 无载波； TDES0[9]: 迟到冲突； TDES0[8]: 冲突过多； TDES0[2]: 顺延(Deferral)过多； TDES0[1]: 下溢错误； TDES0[16]: IP 报头错误； TDES0[12]: IP 数据错误。</p>
位 14	JT	rw	<p>嘎嗒超时 (Jabber timeout)</p> <p>置'1'时，该位表示 MAC 发送端发生了嘎嗒超时。该位只在以太网 MAC 配置寄存器(EMAC_MACCTRL)的 JAD 位不为'1'时才会在发生嘎嗒超时时被 置'1'。</p>
位 13	FF	rw	<p>帧清空 (Frame flushed)</p> <p>置'1'时，该位表明由于 CPU 发出帧清空命令，DMA 或者 MTL 把帧从 FIFO 中清空。</p>
位 12	IPE	rw	<p>IP 数据错误 (IP payload error)</p> <p>置'1'时，该位表明 MAC 发送端检测到了 TCP、UDP 或者 ICMP 的 IP 数据错误。发送端会把接收到的 IPv4 或者 IPv6 报头中的长度域和实际接收到的 TCP、UDP 和 ICMP 字节数目做比对，如果不相等就置'1'报错。</p>
位 11	LOC	rw	<p>载波丢失 (Loss of carrier)</p> <p>置'1'时，该位表明在帧发送的过程中发生了载波丢失(MII_CRS 信号在发送过程中存在一个或多个以上发送时钟周期的无效状态)。该位只有在半双工模式下且发送帧没有冲突时有效。</p>
位 10	NC	rw	<p>无载波 (No carrier)</p> <p>置'1'时，表示在帧发送的过程中从 PHY 发过来的载波侦听信号没有置起。</p>
位 9	LC	rw	<p>迟到冲突 (Late collision)</p> <p>置'1'时，该位表明帧因为在发送过程中在冲突检测窗口检测到冲突(MII 模式下，包括前导符的 64 个字节时间)而中止发送。在下溢错误位置'1'的情况下该位无效。</p>
位 8	EC	rw	<p>冲突过多 (Excessive collision)</p> <p>置'1'时，该位表明 MAC 在尝试发送当前帧的过程中因为连续发生了 16 次冲突而中止发送。如果以太网 MAC 配置寄存器(EMAC_MACCTRL)的 RD(不进行重试)位为'1'，那么在发生第一次冲突后，该位就置'1'，并中止发送。</p>
位 7	VF	rw	<p>VLAN 帧 (VLAN frame)</p> <p>置'1'时，该位表明发送的帧是 VLAN 类型帧。</p>
位 6: 3	CC	rw	<p>冲突计数 (Collision count)</p> <p>该 4 位值记录了帧发送出去前出现的冲突次数。在 EC 位(TDES0[8])为'1'时，该位无效。这位仅在半双工模式下有效。</p>
位 2	ED	rw	<p>顺延过多 (Excessive deferral)</p> <p>置'1'时，表示在以太网 MAC 配置寄存器(EMAC_MACCTRL)的 DC 位为'1'时，因为顺延超过 24288 位时而结束发送。</p>
位 1	UF	rw	<p>数据下溢错误 (Underflow error)</p> <p>置'1'时，该位表示从系统缓存送到 MAC 的数据过迟，导致 MAC 中止帧发送过程。下溢错误表示 DMA 在发送帧的过程中遇到了空的发送缓存。发送进程进入挂起状态，并设置发送下溢位(以太网 DMA 状态寄存器(EMAC_DMASTS)位 5 TU)和发送中断位(以太网 DMA 状态寄存器(EMAC_DMASTS)位 0 TX)置'1'。</p>
位 0	DB	rw	顺延位 (Deferred bit)

置'1'时，该位表示 MAC 在发送前因为监测到载波信号而推迟帧发送。该位只在半双工模式下有效。

TDES1：发送描述符字 1

域	简称	类型	功能
位 31: 29	保留	resd	请保持默认值。
位 28: 16	TBS2	rw	发送缓存 2 大小 (Transmit buffer 2 size)
位 15: 13	保留	resd	第二个数据缓存的大小(以字节计)。如果 TDES0[20]位为'1'时，这些位表示第二个描述符的地址。
位 12: 0	TBS1	rw	发送缓存 1 大小 (Transmit buffer 1 size)
			第一个数据缓存的大小(以字节计)。如果它的值是 0，那么 DMA 跳过这个缓存，根据 TDES0[20]位配置使用缓存 2 或者下一个缓存。

TDES2：发送描述符字 2

TDES2 包含描述符的第一个缓冲区的地址指针。

域	简称	类型	功能
位 31: 0	TBAP1/T TSL	rw	发送缓存 1 地址指针/发送帧时间戳低 32 位 (Transmit buffer 1 address pointer / Transmit frame time stamp low) 这些位有 2 个功能： 1: 应用程序用它们指示 DMA 以太网帧数据在系统缓存中的位置。 2: 发送帧过程结束时，DMA 可以用它们存放发送帧的时间戳

TBAP1 DMA 占有当前描述符时，这些位表示缓存 1 的物理地址。

TDES3：发送描述符字 3

TDES3 包含描述符的第二个缓冲区的地址指针或下一个描述符。

域	简称	类型	功能
位 31: 0	TBAP2/T TSH	rw	发送缓存 2 地址指针(下个描述符地址)/发送帧时间戳高位 (Transmit buffer 2 address pointer (Next descriptor address) / Transmit frame time stamp high) 这些位有 2 个功能： 1: 程序用它们指示 DMA 以太网数据在系统缓存中的位置。 2: 帧发送结束后，DMA 可以用它们存放帧的时间戳高 32 位

TBAP2 DMA 占有当前描述符时，如果描述符队列为环式时，这些位表示缓存 2 的物理地址。如果描述符为链式的，这些位表示下一个描述符的物理地址。

TXDMA 增强描述符

有如下两种情况必须使用增强的发送描述符：

1. 时间戳被激活(寄存器EMAC_PTPTSCTRL TE位被置1)
2. IPV4校验和被激活(寄存器EMAC_MACCTRL IPC位被置1)

通过将 EMAC_DMABM 寄存器 EDE 位置 1 使能增强描述符功能。增强的 TX 描述符包括 TDES0,TDES1,TDES2,TDES3,TDES4,TDES5,TDES6 和 TDES7，总共 8 个 32 位字。TDES0~3 和常规 TX 描述符的功能是一样的，TDES4~5 未用，TDES6~7 主要用来保存 64bit 时间戳。

TDES6：发送描述符字 6

TDES6 包含时间戳低 32 位数据。

域	简称	类型	功能
位 31: 0	TTSL	rw	DMA 释放描述符给 CPU 前，DMA 会把对应发送帧的时间戳低 32 位写入这些位。 只有在该帧的时间戳使能(TDES0 位 25 的 TTSE 位为'1')和帧的 LS 位为'1'时（表明当前帧传输结束），DMA 才会把时间戳写入这些位。

TDES7：发送描述符字 7

TDES7 包含时间戳数据高 32 位数据。

域	简称	类型	功能
---	----	----	----

位 31: 0	TTSH	rw	发送帧时间戳高 32 位位 (Transmit frame time stamp high) DMA 会把对应帧的时间戳高 32 位写入这些位。只有在帧的时间戳使能(TDES0 位 25 的 TTSE 位为'1')和 LS 位(帧的最后一部分)为'1'时 (表明当前帧发送结束)，DMA 才会把时间戳写入这些位
---------	------	----	--

RXDMA 设置

3. 应用程序建立接收描述符(RDES0~RDES3)，并置 OWN 位为‘1’将描述符释放给 DMA。
4. 配置 SSR 位(EMAC_DMAOPM[1])为‘1’，DMA 进入接收运行状态，DMA 尝试获取接收描述符，如果取到的描述符不可用(被 CPU 占有)，则 DMA 进入暂停状态，并跳到步骤 9。
5. DMA 从取得的接收描述符中解析出接收缓存地址。
6. DMA 将帧数据从 RXFIFO 写入到接收缓存。
7. 如果缓存被填满或者帧传输结束，接收控制器会取描述符队列中的下一个接收描述符。
8. 如果当前帧传输结束，DMA 跳到步骤 7。如果当前帧传输没有结束(未接收到帧尾 EOF)而下一个接收描述符的 OWN 位为 0，这时，如果使能了帧清空功能，则 DMA 置位 RDES0 的描述符错误位并关闭当前描述符(OWN 位清‘0’)同时置位 RDES1 的 LS 位，然后跳到步骤 8 (注意如果没有打开帧清空功能，则不会置位 RDES1 的 LS 位)。如果当前帧传输没有结束，而下一个接收描述符的 OWN 位为 1，那么 DMA 关闭当前描述符，标记其为中间描述符，然后退回步骤 4。
9. 如果使能了 IEEE1588 时间戳功能，DMA 在写回状态信息给 RDES0 的同时也会把时间戳写入当前描述符的 RDES2 和 RDES3。然后清除当前描述符 RDES0 OWN 位同时把 LS 位置‘1’。
10. 接收控制器检查新获取的接收描述符，如果描述符被 DMA 占有则返回到步骤 4。如果描述符被 CPU 占有，RXDMA 会进入挂起状态，并置位接收缓存不可用位，如果使能了接收帧清空功能，控制器会清空接收帧。
11. DMA 收到接收查询命令或者接收 FIFO 里有下一个接收到帧的帧头时，DMA 退出挂起状态，并取下一个描述符，然后跳到步骤 2。

获取接收描述符

RXDMA 总是会尝试获取另一个接收描述符。只要满足下列条件中的任意一个，RXDMA 就会尝试获取接收描述符：

- 在 DMA 进入运行状态 (EMAC_DMAOPM SSR 位被置为‘1’)
- 当前描述符指定的缓存不足以容纳整个帧或者剩余帧数据
- 控制器已经完成当前帧的接收，但还没有关闭当前的接收描述符
- 接收到新的帧但是接收流程由于描述符被 CPU 占有而挂起
- 接到接收轮询命令

接收帧处理

MII/RMII 接口接收帧数据并把帧存入 RXFIFO，当达到设定的条件后 RXDMA 开始把数据转发给当前描述符对应的接收缓存。DMA 会把当前描述符 RDES0 的 FS 位置 1，表示缓存里是帧的第一部分。在 DMA 填满缓存，或者帧接收完毕后，DMA 把 OWN 位清 0 以释放描述符。如果当前描述符缓存足以容纳完整帧数据，则当前描述符 RDES0 LS 位和 FS 位都会被置 1。

DMA 取下一个描述符，把前一个描述符的 LS 位置为‘1’，写回接收状态信息到前一个描述符再关闭前一个描述符。然后，DMA 把接收中断位(EMAC_DMASTS[6])置为‘1’。上面的步骤会一直重复直到 DMA 发现描述符被 CPU 占有，此时接收流程会把接收缓存不可用位(EMAC_DMASTS[7])置为‘1’，并进入挂起状态。在进入挂起状态时会记录描述符列表地址指针当前值，并在退出挂起状态后作为描述符开始的地址。

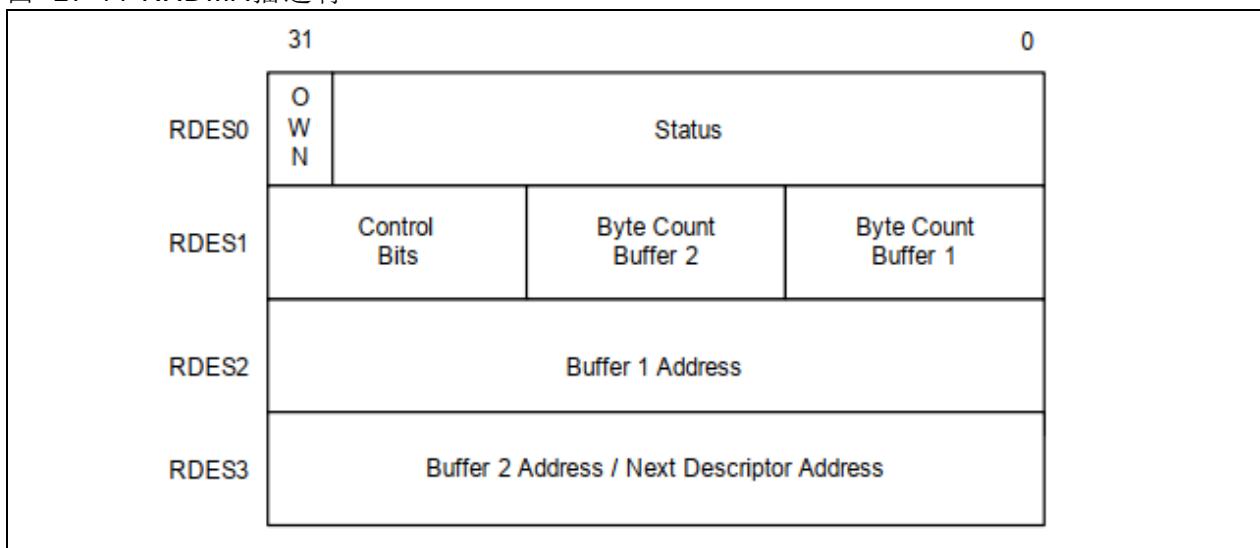
接收流程暂停

在 RXDMA 挂起的情况下，如果 RXFIFO 接收到新的帧，DMA 会重新从 AT32F457xx 内存里取当前描述符。如果取的描述符 OWN 位为 1，接收流程就会重新进入运行模式。如果 OWN 位为 0，那么 DMA 默认会丢弃在接收 FIFO 顶部的这个帧，并且丢失帧计数器加 1。如果 RXFIFO 里有一个以上的帧，则重复上述的步骤。把以太网 DMA 工作模式寄存器(EMAC_DMAOPM)的位 24(DFRF 位)置为‘1’可以阻止清空或者丢弃接收 FIFO 顶部的帧；这时，接收流程会把接收缓存不可用位置为‘1’，并回到挂起状态。

RXDMA 描述符

如下图，描述符结构体包含 4 个 32 位字，RDES0、RDES1、RDES2 和 RDES3。

图 27-11 RXDMA 描述符



RDES0：接收描述符字 0

RDES0 包含了接收帧的状态信息，帧长和描述符的占有信息。

域	简称	类型	功能
位 31	OWN	rw	占有位 (Own bit) 0: 表示 CPU 占有描述符； 1: 表示 DMA 占有描述符。 DMA 在传输完整个帧或者填满描述符对应的缓存以后把该位清‘0’，释放描述符给 CPU。
位 30	AFM	rw	目的地址过滤器未通过 (Destination address filter fail) 置‘1’时，该位表示帧没有通过 MAC 控制器的目的地址(DA)过滤器。
位 29: 16	FL	rw	帧长 (Frame length) 这些位表示 DMA 发送到系统缓存的接收帧的字节数目。只有在 LS 位(RDES0[8])为‘1’且描述符错误位(RDES0[14])为‘0’时，这些位才有效。如果 LS 位为 0，则这些位的值表示帧当前传送到系统缓存里的累计字节数目。帧长是否包括 CRC 部分取决于以太网 MAC 配置寄存器(EMAC_MACCTRL)中 bit7 和 bit25 的配置。
位 15	ES	rw	错误汇总 (Error summary) 该位为下列位的逻辑“或”： RDES0[1]: CRC 错误； RDES0[3]: 接收错误； RDES0[4]: 看门狗超时； RDES0[6]: 迟到冲突； RDES0[7]: 巨大帧或者 IP 校验和错误； RDES0[11]: 溢出错误； RDES0[14]: 描述符错误。 该位在 LS 位(RDES0[8])为‘1’时才有效。
位 14	DE	rw	描述符错误 (Descriptor error) 置‘1’时，该位表示由于描述符的缓存装不下当前接收帧，而 DMA 又不占有下一个描述符，导致当前帧被切断。该位只有在 LS 位(RDES0[8])为‘1’时才有效。
位 13	SAF	rw	源地址过滤器未通过 (Source address filter fail) 置‘1’时，该位表示接收帧没有通过 MAC 控制器的源地址(SA)过滤器。
位 12	LE	rw	长度错误 (Length error) 置‘1’时，表示接收到帧的实际长度与以太网帧头长度/类型域的值不相符。该位只有在 RDES0[5]位为‘0’时才有效。该位在有 CRC 错误时无效。
位 11	OE	rw	溢出错误 (Overflow error) 置‘1’时，表示由于接收 FIFO 溢出，接收到的帧被破坏。
位 10	VLAN	rw	VLAN 标签 (VLAN tag) 置‘1’时，表示当前描述符指向的帧被 MAC 标记为 VLAN 帧。
位 9	FS	rw	第一个描述符 (First descriptor)

			置'1'时，表示这个描述符包含帧的第一个缓冲区，如果缓冲区的长度为 0，则第 2 个缓冲区包含帧的开头；如果第 2 个缓冲区的长度也是 0，则下一个描述符指示的缓冲区包含帧的开头。
位 8	LS	rw	最后一个描述符 (Last descriptor) 置'1'时，表示这个描述符指向的缓存区是帧的最后一个缓冲区。
位 7	IPHCE	rw	IPv 报头校验和错误 (IPv header checksum error) 置'1'时，表示 IPv4 或者 IPv6 的报头错误。错误可能是由于以太网类型域和 IP 版本域值不匹配，IPv4 报头校验和不对或者以太网帧的 IP 报头字节数不足造成的。
位 6	LC	rw	迟到冲突 (Late collision) 置'1'时，该位表示在半双工模式下接收帧的时候发生了迟到冲突。
位 5	FT	rw	帧类型 (Frame type) 置'1'时表示接收到的帧是以太网帧 (LT 域大于等于 1536)。清 '0' 时表示接收到的帧是 IEEE802.3 帧。在接收到帧是小于 14 字节的过短帧时，该位无效。
位 4	RWT	rw	接收看门狗超时 (Receive watchdog timeout) 置'1'时表示在接收当前帧的时候，看门狗超时。看门狗超时后，当前接收帧被截断。
位 3	RE	rw	接收错误 (Receive error) 置'1'时表示在接收帧过程中 RX_DV 有效时，RX_ER 信号有效。
位 2	DE	rw	Dribble 位错误 (Dribble bit error) 置'1'时表示接收到的帧长度不是字节的整数倍(奇数个 4 位数据)。该位只在 MII 模式下有效。
位 1	CE	rw	CRC 错误 (CRC error) 置'1'时表示接收到的帧发生 CRC 错误。该位只在 LS 位(RDES0[8])为'1'时有效。
位 0	PCE	rw	数据校验和错误 (Payload checksum error) 置'1'时，表示 MAC 控制器计算的 TCP、UDP 或 ICMP 校验与接收到帧的 TCP、UDP 或 ICMP 的校验和域值不相符。在接收到以太网帧的数据长度和 IPv4 或 IPv6 数据包长度域的值不符时，该位也会置'1'。

下表显示了位 5、7、0 取值的含义

表 27-7 接收描述符 0

位 5: 帧类型	位 7: 校验和错误	位 0: 数据校验和错误	帧状态
0	0	0	IEEE802.3 类型帧(长度域值小于 0x0600)
1	0	0	IPv4/IPv6 类型帧，未检测到校验和错误
1	0	1	IPv4/IPv6 类型帧，检测到数据校验和错误(PCE 位)
1	1	0	IPv4/IPv6 类型帧，检测到 IP 报头校验和错误(IPC CE 位)
1	1	1	IPv4/IPv6 类型帧，检测到 IP 报头校验和错误和数据校验和错误
0	0	1	IPv4/IPv6 类型帧，未检测到 IP 报头校验和错误；由于不支持的数据格式，未执行数据校验和检测
0	1	1	帧类型即不是 IPv4 也不是 IPv6(校验和模块不执行校验和检验)
0	1	0	保留

RDES1: 接收描述符字 1

域	简称	类型	功能
位 31	DIC	rw	关闭接收完成中断 (Disable interrupt on completion) 置'1'时，在帧接收完成的时候，对于当前描述符指示的帧，不会设置以太网 DMA 状态寄存器(EMAC_DMASTS)的 RECV 位(位 6)。这样也就不会发生 RECV 位触发的中断。该位只在 RDES0[8]为 1 时有效。
位 30: 29	保留	resd	请保持默认值。
位 28: 16	RBS2	rw	接收缓存 2 大小 (Receive buffer 2 size) 这些位的值给出接收缓存 2 的大小，以字节为单位。如果 RDES1[14]位为'1'，这些位取值无意义。
位 15	RER	rw	接收描述符环形结构结尾 (Receive end of ring) 置'1'时，该位表示当前描述符是描述符队列中的最后一个。DMA 会回到描述符队列地址去取下一个描述符，使描述符队列形成环形结构。

			第二地址链表 (Second address chained)
位 14	RCH	rw	置'1'时，该位表示描述符里的第二地址是下一个描述符的地址，而不是第二个缓存的地址。该位为'1'时，RBS2(TDES1[28: 16])的值可以忽略。RDES0[15]的作用优先于本位 RDES0[14]。
位 13	保留	resd	请保持默认值。
位 12: 0	RBS1	rw	接收缓存 1 大小 (Receive buffer 1 size) 这些位的值表示接收缓存 1 的大小，以字节为单位。如果这些位取值为 0，DMA 忽略接收缓存 1，按照 RDES[14]的取值，直接取接收缓存 2 或者下一个描述符。

RDES2: 接收描述符字 2

RDES2 包含描述符第一个数据缓存的地址指针。

域	简称	类型	功能
位 31: 0	RBAP1/R TSL	rw	接收缓存 1 地址指针/接收帧时间戳低位 (Receive buffer 1 address pointer / Receive frame time stamp low) 这些位有 2 个功能：应用程序用它们表示 DMA 数据在内存中的位置。等到数据接收完后，DMA 可以用它们存放帧的时间戳。
	RBAP1		DMA 占有描述符时，这些位表示缓存 1 的物理地址。

RDES3: 接收描述符字 3

RDES3 包含描述符第二个数据缓存的地址指针或者指向下一个描述符。

域	简称	类型	功能
位 31: 0	RBAP2/R TSH	rw	接收缓存 2 地址指针(或者下一个描述符地址指针)/接收帧时间戳高位 (Receive buffer 2 address pointer (next descriptor address) / Receive frame time stamp high) 位 31: 0 这些位有 2 个功能：程序用它们表示 DMA 数据在内存中的位置。等到数据接收完后，DMA 可以用它们存放帧的时间戳
	RBAP2		DMA 占有当前描述符时，在描述符队列为环形结构的时候，这些位表示缓存 2 的物理地址。如果第二地址链表位(RDES0[14])为'1'，并且下一个描述符存在，则这些位指向下一个描述符的物理地址。

RXDMA 增强描述符

有如下两种情况必须使用增强的发送描述符：

1. 时间戳被激活(寄存器EMAC_PTPTSCTRL TE位被置1)
2. IPv4校验和被激活(寄存器EMAC_MACCTRL IPC位被置1)

通过将 EMAC_DMABM 寄存器 EDE 位置 1 使能增强描述符功能。增强的 RX 描述符包括 RDES0,RDES1,RDES2,RDES3,RDES4,RDES5,RDES6 和 RDES7，总共 8 个 32 位字。RDES0~3 和常规 RX 描述符的功能是一样的，RDES4 包含拓展状态，RDES5 保留，RDES6~7 主要用来保存 64bit 时间戳。

RDES4: 接收描述符字 4

域	简称	类型	功能
位 31: 14	保留	resd	请保持默认值。
位 13	PV	ro	PTP 版本 (version of PTP) 0: IEEE 1588 版本 1 1: IEEE 1588 版本 2
位 12	PFT	ro	PTP 帧类型 (PTP frame type) 该位置 1 时，表示 PTP 消息直接通过以太网发送。该位清零且消息类型为非零时，表示 PTP 消息通过 UDP-IPv4 或 UDP-IPv6 发送。可通过位 6 或位 7 获得有关 IPv4 或 IPv6 的信息。
位 11: 8	PMT	ro	PTP 消息类型(PTP message type)。 0000: 未接收到任何 PTP 消息 0001: SYNC 0010: Follow_Up 0011: Delay_Req 0100: Delay_Resp 0101: Pdelay_Req/Announce

			0110: Pdelay_Resp/Management 0111: Pdelay_Resp_Follow_Up/Signaling 1xxx: 保留
位 7	IPV6PR	ro	接收到 IPv6 数据包 (IPv6 packet received) 该位置 1 时, 表示接收到的数据包为 IPv6 数据包。
位 6	IPV4PR	ro	接收到 IPv4 数据包 (IPv4 packet received) 该位置 1 时, 表示接收到的数据包为 IPv4 数据包。
位 5	IPCB	ro	绕过 IP 校验和 (IP checksum bypassed) 该位置 1 时, 表示绕过校验和减荷引擎。
			IP 有效负载错误 (IP payload error)
位 4	IPPE	ro	该位置 1 时, 表示由内核计算的 16 位 IP 有效负载校验和 (即 TCP、UDP 或 ICMP 校验和) 与接收段中对应的校验和字段不匹配。当 TCP、UDP 或 ICMP 段长度与 IP 报头字段中的有效负载长度值不匹配时, 它同样会置 1。
			IP 报头错误 (IP header error)
位 3	IPHE	ro	该位置 1 时, 表示由内核计算的 16 位 IPv4 报头校验和与接收的校验和字节不匹配, 或 IP 数据报版本与以太网类型值不一致。
			IP 有效负载字节 (IP payload type)
位 2: 0	IPPT	ro	如果 IPv4 校验和减荷被激活 (IPC=1, EMAC_MACCTRL 位 10), 则这些位表示 IP 数据报中封装的有效负载类型。当 IP 报头出错或存在分段的 IP 时, 这些位为“00”。 000: 未知, 或未处理 IP 有效负载 001: UDP 010: TCP 011: ICMP 1xx: 保留

RDES6: 接收描述符字 6

RDES6 包含时间戳数据低 32 位数据。

域	简称	类型	功能
位 31: 0	RTSL	rw	DMA 释放描述符之前, DMA 会把帧的时间戳低 32 位写入这些位。只有在使能时间戳功能和 LS 位为'1'(表示缓存存放的是帧的最后分块)时, DMA 才会在这些位写入时间戳。

RDES7: 接收描述符字 7

RDES7 包含时间戳数据高 32 位数据。

域	简称	类型	功能
位 31: 0	RTSH	rw	DMA 释放描述符之前, DMA 会把帧的时间戳高 32 位写入这些位。只有在时间戳使能和 LS 位为'1'(表示缓存存放的是帧的最后分块)时, DMA 才会在这些位写入时间戳。

27.2.4 EMAC掉电模式进入及唤醒

在使能了以太网 MAC PMT 控制和状态寄存器(EMAC_MACPMTCTRLSTS)的 PD 位时, EMAC 会进入掉电模式, 在掉电模式下 EMAC 会丢弃所有的帧而不把它们转发给应用程序。

PMT 模块支持接收远程唤醒帧和 AMD Magic Packet 帧用以把 EMAC 从掉电模式下唤醒, 需要设置相应的使能位 ERWF 和 EMP (以太网 MAC PMT 控制和状态寄存器(EMAC_MACPMTCTRLSTS)位 6 和位 5)。

远程唤醒帧过滤器寄存器

唤醒帧寄存器一共有 8 个, 需要逐一配置帧过滤器寄存器。连续写 8 次唤醒帧过滤器寄存器, 就可以把需要的值分别写入唤醒帧过滤器寄存器 0-7。对这些寄存器的读操作流程和写操作流程一致, 需要连续读 8 次唤醒帧过滤器寄存器, 才能读出唤醒帧过滤器寄存器 0-7 的值。

图 27-12 唤醒帧过滤寄存器

Wkuppkfilter_reg0	Filter 0 Byte Mask											
Wkuppkfilter_reg1	Filter 1 Byte Mask											
Wkuppkfilter_reg2	Filter 2 Byte Mask											
Wkuppkfilter_reg3	Filter 3 Byte Mask											
Wkuppkfilter_reg4	RESD	Filter 3 Cmd	RESD	Filter 2 Cmd	RESD	Filter 1 Cmd	RESD	Filter 0 Cmd				
Wkuppkfilter_reg5	Filter 3 Offset		Filter 2 Offset		Filter 1 Offset		Filter 0 Offset					
Wkuppkfilter_reg6	Filter 1 CRC-16				Filter 0 CRC-16							
Wkuppkfilter_reg7	Filter 3 CRC-16				Filter 2 CRC-16							

过滤器 i 字节屏蔽

该寄存器定义了使用过滤器 i ($i=0\sim3$)的哪些字节检查判断帧是否为唤醒帧。第 31 位必须为' 0'；位 $j[30:0]$ 是字节屏蔽位，如果位 j 为' 1'，则 CRC 模块会处理输入帧的第[过滤器 i 偏移+ j]字节，否则忽略之过滤器 i 命令

4bit 命令。位 3 选择地址类型，如果该位为' 1'，则只检测多播地址；如果该位为' 0'，则只检测单播地址。位 2 和位 1 是保留位。位 0 是过滤器的使能位，如果为' 0'，则该过滤器不工作

过滤器 i 偏移

这个 8bit 寄存器定义了过滤器 i 要检查的首字节在帧内的起始偏移。最小允许取值是 12，代表了帧的第 13 个字节(偏移值为 0 表示帧的第一个字节)

过滤器 i CRC-16

该寄存器包含了过滤器计算出的 CRC_16 值，也包含了唤醒帧过滤器寄存器模块预先写好的字节屏蔽值

远程唤醒帧检测

设置以太网 MAC PMT 控制和状态寄存器(EMAC_MACPMTCTRLSTS)的 RRWF 为' 1' 时可以使能远程唤醒帧检测。

PMT 支持 4 个可编程的过滤器，如果输入帧通过了过滤器的地址过滤，而且过滤器 CRC_16 与被检查的输入帧匹配，则认为接收到唤醒帧。PMT 只会检查唤醒帧是否有长度错误、FCS 错误、Dribble 位错误、MII 错误、冲突，并确保唤醒帧不是过短帧。

当接收到远程唤醒帧后 EMAC 将从睡眠模式时恢复到正常工作状态。同时 EMAC_MACPMTCTRLSTS 位 6 (RRWF) 将会被置 1，表示接收到了远程唤醒帧。如果使能了远程唤醒中断，那么 PMT 在接收到远程唤醒帧时会产生中断。

Magic Packet 检测

通过配置 EMAC_MACPMTCTRLSTS 位 1 (EMP) 可以使能 Magic Packet 帧检测。Magic Packet 帧包含特殊信息的数据包，专门用来唤醒 LAN 上的站点。

Magic Packet 帧的数据格式：6 字节全 1，紧接着重复 16 次的 MAC 地址。比如某设备的 MAC 地址是 0x11aabb22cc33,那么用于唤醒这个设备的 Magic Packet 帧如下：

目的地址 源地址FFFF FFFF FFFF

11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33

11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33

11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33

11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33 11aa bb22 cc33

... CRC

EMAC 睡眠模式下，PMT 模块会持续检测每一个发向本站点的帧，检测其是否符合 Magic Packet 的格式。

接收到 Magic Packet 时会更新以太网 MAC PMT 控制和状态寄存器(EMAC_MACPMTCTRLSTS)的位 5

(RMP)。如果使能了 Magic Packet 中断，则 PMT 在接收到 Magic Packet 时会产生中断。

系统在深度睡眠模式下注意事项

在 MCU 处于深度睡眠模式时，使能外部中断线 19，以太网的 PMT 模块仍能够检测帧。但是要注意以太网 MAC 配置寄存器(EMAC_MACCTRL)的 RE 位需要保持为'1'，因为 EMAC 需要检测 Magic Packet 或者远程唤醒帧。

推荐的进入深度睡眠模式以及唤醒顺序如下：

1. 关闭 TXDMA 并等待之前所有的帧发送完毕。可以通过轮询以太网 DMA 状态寄存器(EMAC_DMASTS)位0 (TI) 来检查发送是否完成。
2. 把以太网MAC配置寄存器(EMAC_MACCTRL)的TE位和RE位清'0'来关闭MAC发送模块和MAC接收模块。
3. 轮询以太网 DMA 状态寄存器(EMAC_DMASTS)位6 (RI)，等待RXDMA把RXFIFO里的所有帧读出，关闭RXDMA。
4. 配置并使能外部中断线19，使其或者能产生事件或者能产生中断。
5. 设置以太网MAC PMT控制和状态寄存器(EMAC_MACPMTCTRLSTS)的EMP/ERWF位为'1'，使能Magic Packet/远程唤醒帧检测。
6. 设置以太网MAC PMT控制和状态寄存器(EMAC_MACPMTCTRLSTS)的PD位为'1'，使能掉电模式。
7. 设置以太网MAC配置寄存器(EMAC_MACCTRL)的RE位为'1'，打开EMAC接收器。
8. MCU进入深度睡眠模式。
9. 接收到Magic Packet/远程唤醒帧后，以太网模块退出掉电模式。
10. 读取以太网MAC PMT控制和状态寄存器(EMAC_MACPMTCTRLSTS)来清除RRWF/RMP，打开EMAC发送状态机，以及发送和RXDMA。
11. 配置MCU系统时钟。

27.2.5 IEEE1588定义的精确时间协议

PTP 协议主要用来同步拥有不同精度，分辨率和稳定度的系统，不局限于以太网。关于详细的 PTP 协议请参考 IEEE 1588 相关文档。

EMAC 的 PTP 模块主要是记录 PTP 包从以太网端口发出和收到的准确时间，并将其返回给应用程序。

基准时钟源

IEEE1588 协议规定，系统需要 64 位格式的基准时间来获得当前时间记录，其中高 32 位提供以秒为单位的时间信息，低 32 位提供以纳秒为单位的时间信息。

PTP 基准时钟主要用来在内部生成系统时间和记录时间戳。这个基准时钟的频率必须大于或等于时间戳计数器的分辨率。主节点和从节点之间的时间同步精度在 100ns 左右。

时间同步的精度主要取决于 PTP 基准时钟输入的频率，晶体振荡器的频漂以及同步流程的执行频度。

带 PTP 功能的帧的发送与接收

使能时间戳功能 (EMAC_PTPTSCTRL 位 0 配置为 1) 以及当前帧的发送时间戳功能也使能时 (发送描述符 TDES0 位 25 配置为 1)，当前帧的起始定界符从 MII 接口输出的时候，EMAC 会记录当前帧的发送时间戳。时间戳高 32 位和低 32 位分别存放在发送描述符 TDES3 和 TDES2 里面，这样时间戳和发送状态信息会一同被返回给应用程序。

使能时间戳功能(EMAC_PTPTSCTRL 位 0 配置为 1)以及接收帧的时间戳使能后(EMAC_PTPTSCTRL 位 8 配置为 1)，EMAC 会在 MII 端记录下所有收到帧的时间戳。时间戳高 32 位和低 32 位分别存放在接收描述符 RDES3 和 RDES2 里面，这样时间戳和接收状态信息会一同被返回给应用程序。

系统时间校准方法

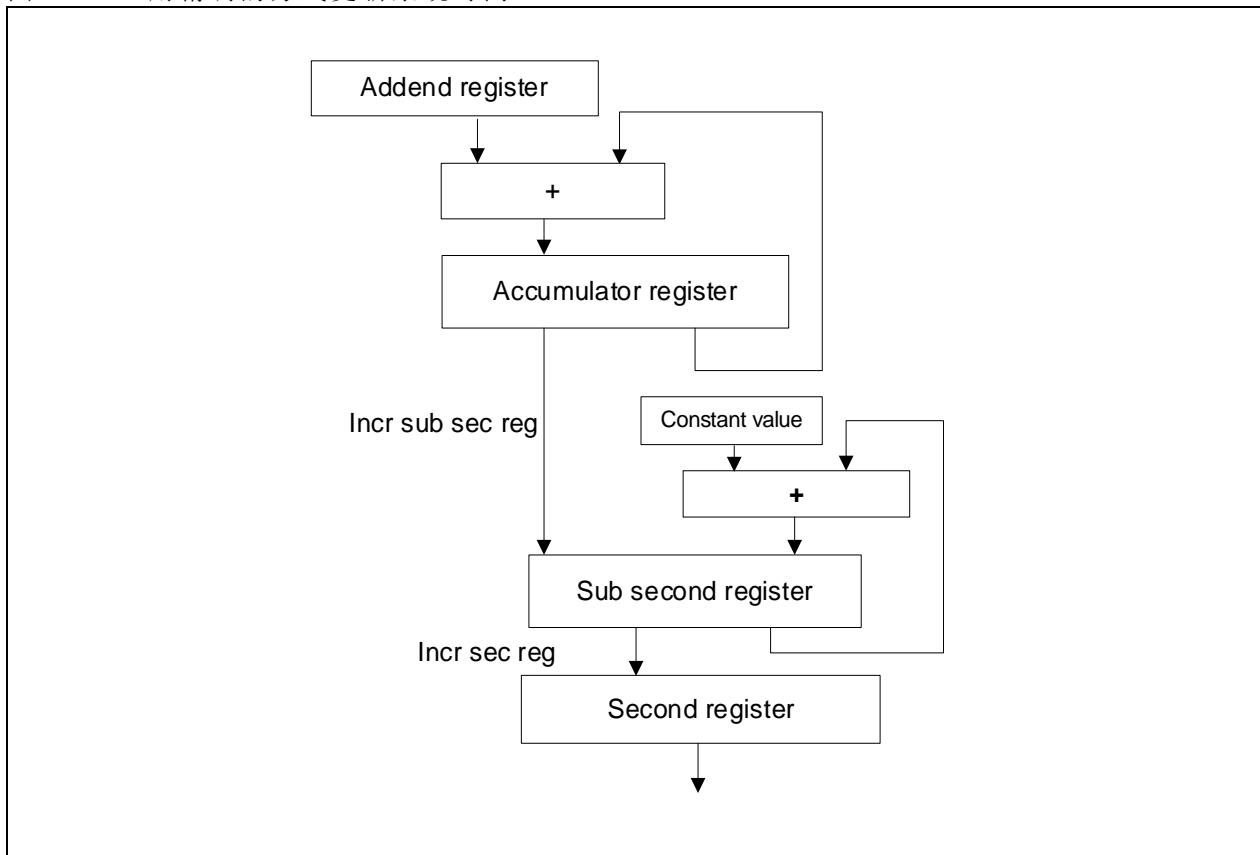
PTP 输入基准时钟是系统时钟 SYSCLK，通过 SYSCLK 来更新发送和接收的以太网帧的 64 位时间戳。系统时间的校准有两种方式：粗略校准和精密校准。

粗略校准：把初始值/偏移值写入时间戳更新寄存器（以太网 PTP 时间戳高更新寄存器(EMAC_PTPTSHUD) 和以太网 PTP 时间戳低更新寄存器(EMAC_PTPTSLUD)）。如果 EMAC_PTPTSCTRL 位 2 (TI) 被设置为 1，则进行初始化流程，时间戳更新寄存器的值会被写入系统时钟计数器。如果 EMAC_PTPTSCTRL 位 3 (TU) 被设置为 1，则进行校准流程，时间戳更新寄存器

值会被作为偏移值，系统时间会加上或者从中减去这个偏移值。

精密校准：在一段时间里纠正从时钟(基准时钟)相对主时钟(如 IEEE1588 定义)的频率偏移。采用这种方法，如下图所示，把加数寄存器的值逐渐加入累加器，累加器的算术进位产生的脉冲令系统时间计数器值增加，加数寄存器的值取决于系统时钟频率。累加器和加数寄存器都是 32 位寄存器。

图 27-13 用精调的方式更新系统时间



假设系统时钟更新电路的精度需要达到 20ns，则亚秒寄存器的更新频率需要是 50MHZ。因此，假设系统时钟频率是 70MHZ，计算频率比得 $70/50=1.4$ 。因此，写入加数寄存器的值是 $2^{32}/1.4$ ，等于 0XB6DB6DB6。加数寄存器中的值要随着系统时钟频率的漂移做相应的改动。亚秒寄存器更新的频率是 50MHZ，每次更新增加的值也即是图中的常量数值为 $2^{31}/(50*10^9)=43$ 。

软件需要利用 Sync 消息计算出频率的漂移，并相应地更新累加寄存器的值。开始时，假设从时间加数寄存器值为 FreqCompensationValue0：

$\text{FreqCompensationValue0} = 2^{32}/\text{频率比}$ 假设对于连续的 Sync 消息，主从节点之间的时延 MasterToSlaveDelay 是恒定的，那么使用下文

描述的算法，在若干个 Sync 周期后，就能锁定频率。从时钟就可以确定精确的 MasterToSlaveDelay 值，并用新的值把从时钟与主时钟同步。算法如下：

在主时钟为 MasterSyncTime_n 时，主节点向从节点发 Sync 消息。从节点在它的时钟为 SlaveClockTime_n 时，收到 Sync 消息，并计算出 MasterClockTime_n 为

$$\text{MasterClockTime}_n = \text{MasterSyncTime}_n + \text{MasterToSlaveDelay}$$

当前 Sync 周期的主时钟计数数目 $\text{MasterClockCount}_n$ 为

$$\text{MasterClockCount}_n = \text{MasterClockTime}_n - \text{MasterClockTime}_{n-1}, \text{ 假设 MasterToSlaveDelay 的数值在 Sync 周期 } n \text{ 与 Sync 周期 } n-1 \text{ 是相同的}$$

当前 Sync 周期的从时钟计数数目 SlaveClockCount_n 为

$$\text{SlaveClockCount}_n = \text{SlaveClockTime}_n - \text{SlaveClockTime}_{n-1}$$

当前 Sync 周期，主时钟计数数目和从时钟计数数目的差别， ClockDiffCount_n 为

$$\text{ClockDiffCount}_n = \text{MasterClockCount}_n - \text{SlaveClockCount}_n$$

从时钟的频率比系数， FreqScaleFactor_n 为

$$\text{FreqScaleFactor}_n = (\text{MasterClockCount}_n + \text{ClockDiffCount}_n) / \text{SlaveClockCount}_n$$

加数寄存器的频率补偿值， $\text{FreqCompensationValue}_n$ 为

$$\text{FreqCompensationValue}_n = \text{FreqScaleFactor}_n \times \text{FreqCompensationValue}_{n-1}$$

该算法具有自校准功能，理论上可以在一个同步周期内锁定频率，不过实际上可能需要多个同步周期才

能同步从设备。

系统时间初始化流程

1. 设置以太网 MAC 中断屏蔽寄存器(EMAC_MAIMR)的位9为' 1'，屏蔽时间戳触发中断。
2. 设置EMAC_PTPTSCTRL位0为' 1'，使能时间戳。
3. 根据系统时间更新精度要求编程亚秒递增寄存器。
4. 如果使用精调校准方式，则设置以太网 PTP 时间戳加数寄存器(EMAC_PTPTSAD)，并设置以太网 PTP 时间戳控制寄存器(EMAC_PTPTSCTRL)的位5为' 1' (更新加数寄存器)，并轮询以太网 PTP 时间戳控制寄存器(EMAC_PTPTSCTRL)直到位5变为' 0'。如果使用粗略校准方式，请忽略此步骤并直接跳到第6步。
5. 如果使用精调校准方式，设置以太网 PTP 时间戳控制寄存器(EMAC_PTPTSCTRL)的位1为' 1'。
6. 把要设置的系统时间值写入以太网 PTP 时间戳高更新寄存器(EMAC_PTPTSHUD)和以太网 PTP 时间戳低更新寄存器(EMAC_PTPTSLUD)。
7. 设置以太网 PTP 时间戳控制寄存器(EMAC_PTPTSCTRL)的位2为' 1'，开始初始化时间戳并轮询此位，此位为0时，初始化完成。
8. 初始化完成，时间戳计数器开始工作。
9. 使能MAC的接收端和发送端，即可以正确地记录时间戳。

用粗调方式更新系统时间的编程步骤

1. 在以太网 PTP 时间戳高更新寄存器(EMAC_PTPTSHUD)和以太网 PTP 时间戳低更新寄存器(EMAC_PTPTSLUD)里写入偏移值(正值或者负值)。
2. 设置以太网 PTP 时间戳控制寄存器(EMAC_PTPTSCTRL)的位3(TU)为' 1'。
3. 在TU位清' 0' 时，系统时间就会加上或者从中减去时间戳更新寄存器的值

用精调方式更新系统时间的编程步骤

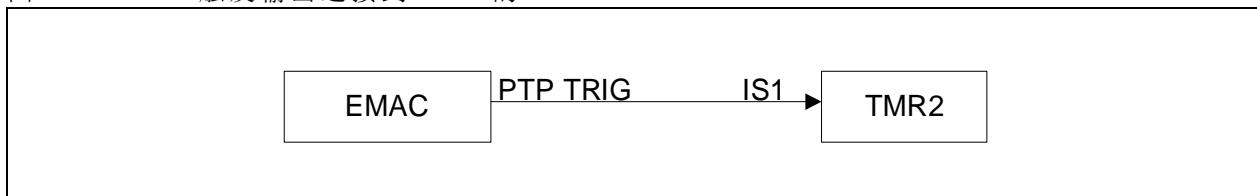
1. 采用前面介绍的算法，计算出加数寄存器的值。
2. 更新加数寄存器。
3. 把要求的目标时间写入目标时间高和目标时间低寄存器，并设置以太网 MAC 中断屏蔽寄存器(EMAC_MAIMR)的位9为' 0' 来允许时间戳中断。
4. 设置以太网 PTP 时间戳控制寄存器(EMAC_PTPTSCTRL)的位4(TITE)为' 1'。
5. 在这个事件产生中断时，读以太网 MAC 中断屏蔽寄存器(EMAC_MAIMR)清除相应的中断标志位。
6. 重新用旧值编写以太网 PTP 时间戳加数寄存器(EMAC_PTPTSAD)并设置以太网 PTP 时间戳控制寄存器(EMAC_PTPTSCTRL)位5为' 1' 来更新加数寄存器。

PTP 触发与 TMR2 的内部连接

EMAC 可以在系统时间大于目标值的时候提供触发中断。使用中断会引入中断时延，为了得到精确的中断延时时间，在系统时间大于目标值的时候，PTP 输出高信号给 TMR2。由于定时器的时钟与 PTP 基准时钟是同步的，因此可以精确的计算中断延时。

设 TMR2 通道输入重映射寄存 (TMR2_RMP) 的位[11:10]为' 01' 可以使能 PTP 输出和 TMR2 IS1 的连接。

图 27-14 PTP 触发输出连接到 TMR2 的 ITR1



PTP 秒脉冲输出信号

PTP 脉冲输出详见以太网 PTP PPS 控制寄存器 (EMAC_PTPPPSCR) 描述，下面只介绍 emac_pps_sel 位 (额外寄存器 3 (CRM_MISC2) 位 9) 为 0 时的情况。

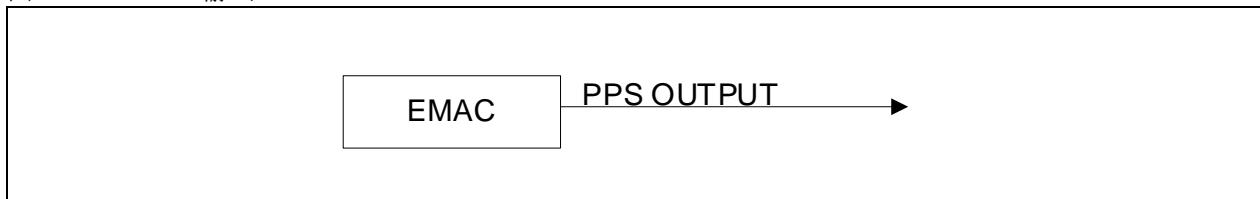
PPS 输出的默认频率为 1 Hz。可使用 PPSFREQ[3: 0] (位于 EMAC_PTPPPSCR 中) 将 PPS 输出

的频率设置为 2PPSFREQ Hz。

如果设置为 1Hz，则使用二进制翻转(TSSSR=0，以太网 PTP 时间戳控制寄存器(EMAC_PTPTSCTRL)中的位 9)时，PPS 脉冲宽度为 125 ms；使用数字翻转(TSSSR=1)时，PPS 脉冲宽度为 100 ms。如果设置为 2 Hz 或更高频率，则使用二进制翻转时，PPS 输出的占空比为 50%。

通过配置 PB5 复用功能，可以使能 PPS 输出功能。

图 27-15 PPS输出



27.2.6 EMAC中断

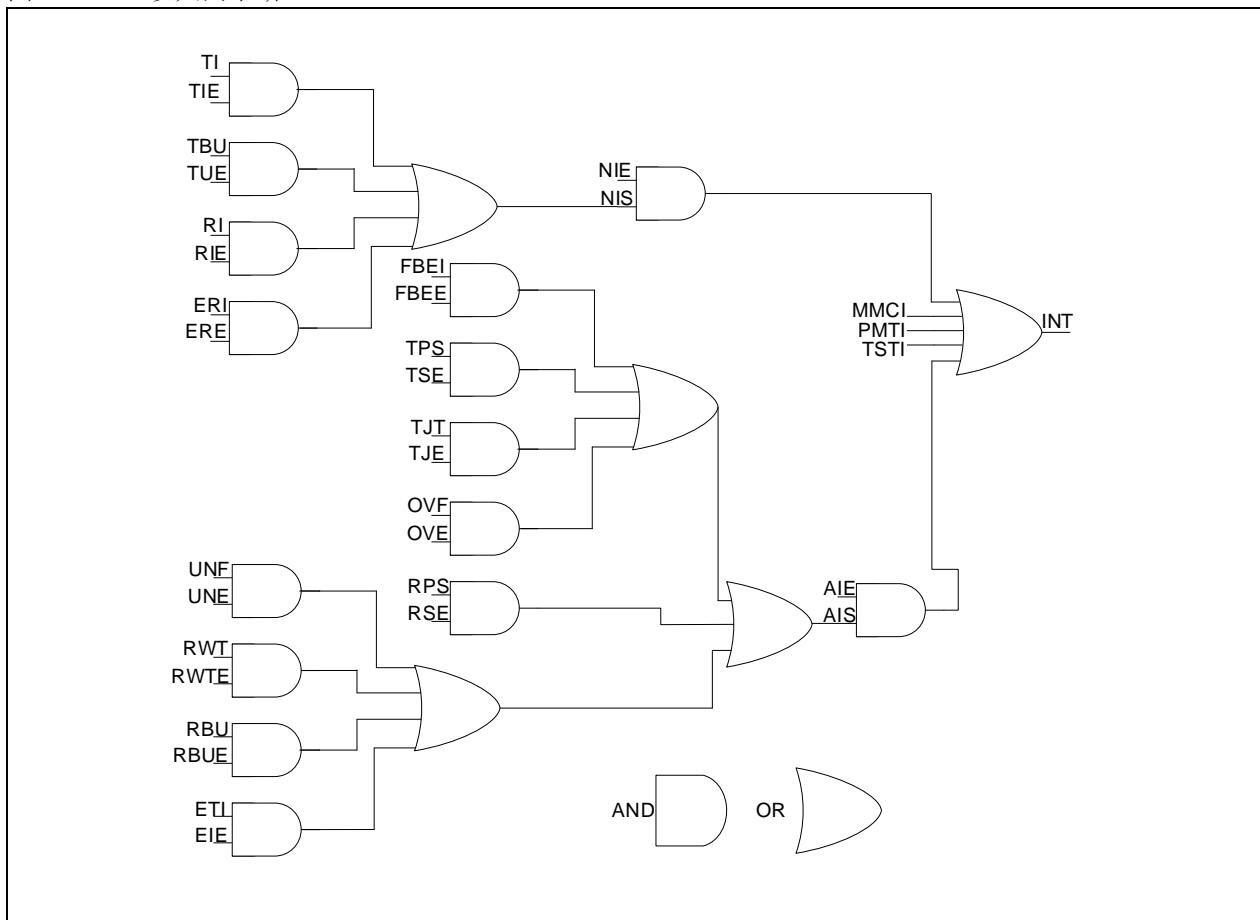
EMAC 有 2 个中断向量，一个用于以太网正常操作，另一个用于映射到 EXINT 线 19 的以太网唤醒事件(检测远程唤醒帧或者 Magic Packet)。

第一个中断向量用于由 MAC 和 DMA 产生的中断。

第二个中断向量用于 PMT 模块在唤醒事件时产生的中断。唤醒事件映射到 EXINT 线 19 上，它可以使 AT32F457xx 微控制器退出低功耗模式，并产生中断。

在以太网唤醒事件映射到 EXINT 线 19 时，一旦发生唤醒事件，如果使能了 EMAC 的 PMT 中断，又使能了 EXINT 线 19 检测到上升沿时产生中断，那么 2 个中断会同时发生。

图 27-16 以太网中断



27.3 EMAC寄存器

下表给出了以太网寄存器映像及其复位值。

可以通过字节(8 位)、半字(16 位)和字(32 位)的形式对本外设的寄存器进行访问。

表 27-8 以太网寄存器映像及其复位值

寄存器简称	基址偏移量	复位值
-------	-------	-----

EMAC_MACCTRL	0x00	0x0000 8000
EMAC_MACFRMF	0x04	0x0000 0000
EMAC_MACHTH	0x08	0x0000 0000
EMAC_MACHTL	0x0C	0x0000 0000
EMAC_MACMIIADDR	0x10	0x0000 0000
EMAC_MACMIIDT	0x14	0x0000 0000
EMAC_MACFCTRL	0x18	0x0000 0000
EMAC_MACVLT	0x1C	0x0000 0000
EMAC_MACRWFF	0x28	0x0000 0000
EMAC_MACPMTCTRLSTS	0x2C	0x0000 0000
EMAC_MACISTS	0x38	0x0000 0000
EMAC_MAIMR	0x3C	0x0000 0000
EMAC_MACA0H	0x40	0x0010 FFFF
EMAC_MACA0L	0x44	0xFFFF FFFF
EMAC_MACA1H	0x48	0x0000 FFFF
EMAC_MACA1L	0x4C	0xFFFF FFFF
EMAC_MACA2H	0x50	0x0000 FFFF
EMAC_MACA2L	0x54	0xFFFF FFFF
EMAC_MACA3H	0x58	0x0000 FFFF
EMAC_MACA3L	0x5C	0xFFFF FFFF
EMAC_MMCTRL	0x100	0x0000 0000
EMAC_MMCR1	0x104	0x0000 0000
EMAC_MMCTI	0x108	0x0000 0000
EMAC_MMCRIM	0x10C	0x0000 0000
EMAC_MMCTIM	0x110	0x0000 0000
EMAC_MMCTFSCC	0x14C	0x0000 0000
EMAC_MMCTFMSCC	0x150	0x0000 0000
EMAC_MMCTFCNT	0x168	0x0000 0000
EMAC_MMCRFECNT	0x194	0x0000 0000
EMAC_MMCRFAECNT	0x198	0x0000 0000
EMAC_MMCRGUFCNT	0x1C4	0x0000 0000
EMAC_PTPTSCTRL	0x700	0x0000 2000
EMAC_PTPSSINC	0x704	0x0000 0000
EMAC_PTPTSH	0x708	0x0000 0000
EMAC_PTPTSL	0x70C	0x0000 0000
EMAC_PTPTSH	0x708	0x0000 0000
EMAC_PTPTSL	0x70C	0x0000 0000
EMAC_PTPTSHUD	0x710	0x0000 0000
EMAC_PTPTSLUD	0x714	0x0000 0000
EMAC_PTPTSAD	0x718	0x0000 0000
EMAC_PTPTTH	0x71C	0x0000 0000

EMAC_PTPTTL	0x720	0x0000 0000
EMAC_PTPTSSR	0x728	0x0000 0000
EMAC_PTPPPSCR	0x72c	0x0000 0000
EMAC_DMABM	0x1000	0x0002 0101
EMAC_DMATPD	0x1004	0x0000 0000
EMAC_DMARPD	0x1008	0x0000 0000
EMAC_DMARDLADDR	0x100C	0x0000 0000
EMAC_DMATDLADDR	0x1010	0x0000 0000
EMAC_DMASTS	0x1014	0x0000 0000
EMAC_DMAOPM	0x1018	0x0000 0000
EMAC_DMAIE	0x101C	0x0000 0000
EMAC_DMAMFBOCNT	0x1020	0x0000 0000
EMAC_DMACTD	0x1048	0x0000 0000
EMAC_DMACRD	0x104C	0x0000 0000
EMAC_DMACTBADDR	0x1050	0x0000 0000
EMAC_DMACRBADDR	0x1054	0x0000 0000

27.3.1 以太网 MAC 配置寄存器(EMAC_MACCTRL)

以太网 MAC 配置寄存器(EMAC_MACCTRL)定义了接收和发送的工作模式。对此寄存器连续的写之间要插入延时，延时值大于 4us。

域	简称	复位值	类型	功能
位 31: 24	保留	0x00	resd	请保持默认值。
位 25	CST	0x0	rw	类型帧 CRC 剥离 (CRC Stripping for Type Frames) 该位被置起后，在传输到应用程序前，剥离以太网类型帧最后 4 字节(FCS)。
位 23	WD	0x0	rw	关闭看门狗 (Watchdog Disable) 关闭看门狗 该位被置起后，MAC 关闭接收器上的看门狗定时器，能够接收多达 16,384 字节的帧。 该位被复位后，MAC 不允许接收超过 2048 字节的帧。
位 22	JD	0x0	rw	关闭超长帧 (Jabber Disable) 该位被置起后，MAC 关闭发送器上的 Jabber (超长帧) 定时器，能够传输多达 16384 字节的帧。 该位被复位后，如果应用程序在传输过程中发送的数据超过 2048 字节时 MAC 会切断发送器。
位 21: 20	保留	0x0	resd	请保持默认值。
位 19: 17	IFG	0x0	rw	帧间间隙 (InterFrame Gap) 这些位是用来控制发送过程中帧与帧之间的最小间隙。 000: 96 bit times 96 位时间 001: 88 bit times 88 位时间 010: 80 bit times 80 位时间 ... 111: 40 bit times 40 位时间 在半双工模式下，IFG 最小值只能设置为 64 位时间(IFG = 100)，不允许更小的值。
位 16	DCS	0x0	rw	关闭载波监听 (Disable Carrier Sense) 该位被置起后，在半双工模式下，MAC 发送器在发送帧过程中会忽略 MII CRS 信号，发送过程中不会因为载波丢失或无载波而报错。 该位复位后，因为有载波监听，MAC 发送器会报错，甚至会中止发送。 在全双工模式配置中，此位保留。

位 15	保留	0x1	resd	请保持默认值。
位 14	FES	0x0	rw	快速 EMAC (Fast EMAC Speed) 此位用于选择 MII, RMII 接口速度。 0: 10 Mbps 1: 100 Mbps
位 13	DRO	0x0	rw	关闭自接收 (Disable Receive Own) 该位被置起后, 如果在半双工模式下将 phy_txen_o 置为有效, 则 MAC 不接受帧。 该位被复位后, MAC 在发送过程中接受 PHY 给出的所有数据包。 该位不适用于 MAC 全双工模式。 当 MAC 被设置为“仅适用于全双工”操作模式时, 该位是保留位 (有默认值的 RO)
位 12	LM	0x0	rw	环回模式 (Loopback Mode) 该位被置起后, MAC 的 MII 运行在环回模式。环回模式需要 MII 接收时钟输入 (即 clk_rx_i) 以确保能正常工作, 因为发送时钟内部没有环回模式。
位 11	DM	0x0	rw	双工模式 (Duplex Mode) 该位被置起后, MAC 运行于全双工模式, 可以同时接收和发送。
位 10	IPC	0x0	rw	IPv4 校验和 (IPv4 Checksum) 该位被置起后, MAC 计算所接收到的所有以太网帧的 16 位补码和, 并检验所收到的帧的 IPv4 报头校验和 (假设是所收到的以太网帧的字节 25-26 或者 29-30 (VLANtagged)) 是否正确, 并在接收状态信息中给出状态。 MAC 还附加了所计算得出的 IP 报头数据包 (即 IPv4header 之后的字节) 的 16 位校验和, 并将这个校验和添加到已传输给应用程序的以太网帧 (当取消选择 Type 2 COE 时)。 当该位被复位后, 此功能关闭。 如果选择了 Type 2 COE, 该位被置起时, 会使能 IPv4 报头检验和检验功能以及 IPv4 或 IPv6 TCP, UDP 或 ICMP 有效载荷检验和检验功能。被复位后, 接收器中的 COE 函数被禁用, 相应的 PCE 和 IP HCE 状态位 (见 138 页的表 3-10) 始终为 0 如果在内核配置过程中, IP 校验和机制未使能, 该位是保留位 (有默认值 RO)
位 9	DR	0x0	rw	关闭重试 (Disable Retry) 该位被置起后, MAC 只尝试发送 1 次。如果在 MII 接口上发生冲突, MAC 忽略当前正在进行的发送帧, 并且在发送帧状态里反馈有过多冲突而引发帧中止。 被复位后, MAC 根据 BL 字段 (位[6: 5]) 的设定尝试重发。该位仅适用于半双工模式, 在“仅适用于全双工模式”配置中, 该位是保留位 (有默认值 RO)。
位 8	保留	0x0	resd	请保持默认值。
位 7	ACS	0x0	rw	自动填充/CRC 剥离 (Automatic pad/CRC Stripping) 该位被置起后, 只有在帧的长度小于 1536 字节时, MAC 才会剥离所接收到的帧的 Pad 和 FCS 域。当接收到的帧的长度等于或大于 1536 字节时, MAC 会保留其 Pad 或 FCS 域并将帧传送给应用程序。 被复位后, MAC 会将所有接收到的帧转发给主机, 而不改变其内容。
位 6: 5	BL	0x0	rw	后退限制 (Back-off Limit) 后退限制定义了 MAC 在发生冲突后, 重新发送前需要等待的随机时间隙整数 (对于 10/100 Mbps 是 512 位时间)。该位仅适用于半双工模式, 在“仅适用于全双工模式”的配置中, 该位是保留位 (RO) 00: k= min (n, 10) 01: k = min (n, 8) 10: k = min (n, 4) 11: k = min (n, 1)

位 4	DC	0x0	rw	其中, n = 重发需要等待的时间间隙数量。r 取的是 0 ≤ r < 2k 之间的随机整数值。
位 3	TE	0x0	rw	递延检验 (Deferral Check) 该位被置起后, MAC 使能递延检验功能。在 10/100 位秒模式下, 如果发送被延迟超过 24288 位时间时, MAC 会发出帧中止状态, 并且在发送帧状态里面将过多延迟错误标志位置起。
位 2	RE	0x0	rw	如果 MAC 在 10 或 100 Mbps 模式中使能了 Jumbo 帧模式, 则递延的阀值就是 155680 位时间。发送器准备发送时就开始递延, 但是当检测到 MII 上有效的载波监听信号时, 递延就不会开始。递延时间不会累计。例如, 如果因为 CRS 信号先是有效的, 而后又变为无效而使发送器延迟了 10000 位时间, 然后发送器才开始发送, 又发生冲突, 因为冲突, 发送器需要后退, 并在完成后退动作之后又重新递延。那么这种情况下, 递延定时器被重置为 0, 并重新启动递延。 当该位被重置后, 递延检验功能被关闭, MAC 会延迟直到 CRS 信号失效。该位仅适用于半双工模式, 在“仅适用于全双工模式”配置中, 该位是保留位 (RO)。
位 1: 0	保留	0x0	resd	使能发送器 (Transmitter Enable) 该位被置起后, MAC 的发送状态机被使能。该位被复位后, MAC 在发送完当前帧后, 关闭发送状态机, 不会再发送任何帧。
位 1: 0	保留	0x0	resd	使能接收器 (Receiver Enable) 该位被置起后, MAC 的接收状态机被使能。 该位被复位后, MAC 在接收完当前帧后, 关闭接收状态机, 不会再接收任何帧。
位 1: 0	保留	0x0	resd	请保持默认值。

27.3.2 以太网 MAC 帧过滤器寄存器(EMAC_MACFRMF)

以太网 MAC 帧过滤器寄存器(EMAC_MACFRMF)包含接收帧的过滤器控制位。一些控制位前往 MAC 的地址检验模块进行第一层级的地址过滤。

第二层级的过滤是根据其它控制位 (比如传递坏帧和传递控制帧) 针对传入的帧执行。

域	简称	复位值	类型	功能
位 31	RA	0x0	rw	接收全部 (Receive All) 该位被置起后, MAC 将接收到的所有帧传递给应用程序, 无论这些帧是否通过了地址过滤。源地址或目的地址的过滤结果(pass or fail)会在接收状态的相应位进行更新 该位被复位后, 接收模块只会将那些通过了源地址/目的地址过滤器的帧传递给应用程序。
位 30: 11	保留	0x00000	resd	请保持默认值。
位 10	HPF	0x0	rw	Hash 或完美过滤器 (Hash or Perfect Filter) 该位被置起后, 如果帧与 HMC 或 HUC 位设定的完美过滤器或 hash 过滤器相符, 地址过滤器就能通过此帧。 该位被复位后, 如果 HUC 或 HMC 位被置起, 则只有当帧符合 hash 过滤器时, 地址过滤器才能通过此帧。
位 9	SAF	0x0	rw	源地址过滤器 (Source Address Filter) 该位被置起后, MAC 会将接收到的帧的源地址域与所使能的源地址寄存器的值进行比较。如果不符, MAC 会丢弃此帧。 该位被复位后, MAC 会将接收到的帧转给应用程序, 并根据源地址比较结果更新接收状态中的源地址过滤器位 (SAF)。
位 8	SAIF	0x0	rw	源地址逆过滤 (Source Address Inverse Filtering) 该位被置起后, 地址检验模块运行于逆过滤模式。源地址与源地址寄存器相符的帧会被标记为未通过源地址过滤器。 该位被复位后, 源地址与源地址寄存器不相符的帧被标记为未通过源地址过滤器。

位 7: 6	PCF	0x0	rw	通过控制帧 (Pass Control Frames) 这些位用于管理所有控制帧的转发 (包括单波和多波 Pause 帧)。 00: MAC 过滤所有控制帧使其无法转到应用程序。 01: MAC 将除 Pause 帧以外的所有控制帧转发给应用程序, 即便这些帧未能通过地址过滤器。 10: MAC 将所有控制帧转给应用程序, 即便这些帧未能通过地址过滤器。 11: MAC 转发通过了地址过滤器的控制帧给应用程序。在处理 Pause 帧时应满足以下条件: 条件 1: MAC 处于全双工模式下, 通过设置寄存器 6 (流控制寄存器) 中的位 2 (REF) 来使能流控。 条件 2: 当寄存器 6 (流控制寄存器) 中的位 3 (UP) 被置起时, 接收的帧的目的地址与特定的多播地址或 MAC 地址 0 相匹配。 条件 3: 接收的帧的类型域 (Type) 是 0x8808, OPCODE 域是 0x0001。
位 5	DBF	0x0	rw	关闭广播帧 (Disable Broadcast Frames) 该位被置起后, 地址过滤模块将阻止所有传入的广播帧。此外, 还会覆盖所有其它的过滤器设置。 当该位被复位后, 地址过滤器模块能允许通过所有接收到的广播帧。
位 4	PMC	0x0	rw	通过多播帧 (Pass MultiCast) 该位被置起后, 所有带多播目的地址 (目的地址的第一位被置 1) 的帧都能通过地址过滤器。 该位被复位后, 多播帧是否能够过滤取决于 HMC 位。
位 3	DAIF	0x0	rw	目的地址逆过滤 (Destination Address Inverse Filtering) 该位被置起后, 地址检验模块运行在逆过滤模式用于比较单播帧和多播帧的目的地址。 该位被复位后, 过滤器将正常工作。
位 2	HMC	0x0	rw	Hash 多播帧 (Hash MultiCast) 该位被置起后, MAC 根据 hash 列表对接收到的多播帧进行目的地址过滤。 该位被复位后, MAC 对多播帧进行目的地址完美过滤, 即将目的地址域与目的地址寄存器的设定值进行比较。如果在内核配置过程中没有选择 Hash 过滤器, 则此位是保留位 (及 RO)
位 1	HUC	0x0	rw	Hash 单播帧 (Hash UniCast) 该位被置起后, MAC 根据 hash 列表对单播帧的目的地址进行过滤。 该位被复位后, MAC 对单播帧的目的地址进行完美过滤, 即将目的地址域与目的地址寄存器的设定值进行比较。
位 0	PR	0x0	rw	混杂模式 (Promiscuous Mode) 该位被置起后, 无论是目的地址还是源地址, 地址过滤器能通过所有传入的帧。当 PR 被置起时, 接收状态的源地址或目的地址错误位总是为 0.

27.3.3 以太网MAC Hash列表高寄存器(EMAC_MACHTH)

64 位的 Hash 列表用于成组的地址过滤。进行 hash 过滤时, 传入帧的目的地址的内容要通过 CRC 逻辑, CRC 寄存器的高 6 位用于检索 Hash 列表。CRC 的最高位决定使用哪个寄存器 (以太网 MAC Hash 列表高寄存器(EMAC_MACHTH)或以太网 MAC Hash 列表低寄存器(EMAC_MACHTL)), 剩余的 5 位决定使用寄存器的哪一位。Hash 值 5b'00000 使用的是所选中寄存器的位 0, 而 Hash 值 5b'11111 使用的是所选中寄存器的位 31.

目的地址的 hash 值按照以下方法计算:

1. 计算目的地址的32位CRC值 (见IEEE 802.3, 参考3.2.8章节获得CRC32计算方法)
2. 对步骤1获取的值进行位反转
3. 从步骤2获取的值中取其高6位

例如, 如果传入帧的目的地址是 0x1F52419CB6AF (MII 接口收到的第一个字节是 0x1F), 那么所计算出

的 6 位 Hash 值是 0x2C，并检查以太网 MAC Hash 列表高寄存器(EMAC_MACHTH)的位 12 是否过滤。如果传入帧的目的地址是 0xA00A98000045，那么所计算的 6 位 Hash 值是 0x07，并检查以太网 MAC Hash 列表低寄存器(EMAC_MACHTL)的位 7 是否过滤。

域	简称	复位值	类型	功能
位 31	HTH	0x0000 0000	rw	该位包含了 Hash 列表的高 32 位。

27.3.4 以太网 MAC Hash 列表低寄存器(EMAC_MACHTL)

以太网 MAC Hash 列表低寄存器(EMAC_MACHTL)包含了 Hash 列表的低 32 位。在内核配置过程中，如果 Hash 过滤器功能关闭或者选择了 128 位或 256 位 Hash 列表，则寄存器 2 和寄存器 3 都是保留位。

域	简称	复位值	类型	功能
位 31	HTL	0x0000 0000	rw	Hash 列表低 (Hash Table Low) 此位包含了 Hash 列表的低 32 位。

27.3.5 以太网 MAC MII 地址寄存器(EMAC_MACMIIADDR)

以太网 MAC MII 地址寄存器(EMAC_MACMIIADDR)通过管理接口控制外部 PHY。

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	请保持默认值。
位 15: 11	PA	0x00	rw	PHY 地址 (PHY Address) 这些位表示的是这 32 个可能的 PHY 设备中的哪一些正在被访问。
位 10: 6	MII	0x00	rw	MII 寄存器 (MII Register) 这些位表示在 PHY 中选择所需的 MII 寄存器。 时钟范围 (Clock Range) CSR 时钟范围选择是指根据用户所使用的 CSR 时钟频率来确定 MDC 时钟频率。 针对每个值（当位 5=0 时）均推荐了 CSR 时钟频率范围，以确保 MDC 时钟频率大致位于 1.0 MHz–2.5 MHz 以内。 0000: CSR 时钟频率为 60 – 100 MHz, MDC 时钟频率为 CSR 时钟除以 42。 0001: CSR 时钟频率为 100 – 150 MHz, MDC 时钟频率为 CSR 时钟除以 62 0010: CSR 时钟频率为 20 – 35 MHz, MDC 时钟频率为 CSR 时钟除以 16. 0011: CSR 时钟频率为 35 – 60 MHz, MDC 时钟频率为 CSR 时钟除以 26. 0100: CSR 时钟频率为 150 – 250 MHz, MDC 时钟频率为 CSR 时钟除以 102. 0101: CSR 时钟频率为 250 – 288 MHz, MDC 时钟频率为 CSR 时钟除以 124. 0110, 0111: 保留。
位 5: 2	CR	0x0	rw	MII 写操作 (MII Write) 当该位被置起时，表示将使用以太网 MAC MII 数据寄存器(EMAC_MACMIIDT)对 PHY 进行写操作。 当该位未被置起时，表示读操作，读的数据将加载到以太网 MAC MII 数据寄存器(EMAC_MACMIIDT)。
位 1	MW	0x0	rw	MII 繁忙 (MII Busy) 在写入以太网 MAC MII 地址寄存器(EMAC_MACMIIADDR)和以太网 MAC MII 数据寄存器(EMAC_MACMIIDT)之前，该位应该先读逻辑 0。在访问 PHY 寄存器期间，软件将该位设置为 1'b1，以此表示正在进行读操作或写操作。 在 MAC 清除该位之前，以太网 MAC MII 数据寄存器(EMAC_MACMIIDT)无效。因此，在对 PHY 写操作时，必须保持 MII 数据于有效状态直到 MAC 清除该位。同样地，在对 PHY 读操作时，以太网 MAC MII 数据寄存器(EMAC_MACMIIDT)的值是无效的，直到该位被清除。
位 0	MB	0x0	rw	

只有在完成前一项操作之后，才能进行随后的读操作或写操作。因为在完成读操作或者写操作后不会有从 PHY 到 MAC 的确认，所以即使 PHY 不存在，该位的功能也不会发生变化。

27.3.6 以太网MAC MII数据寄存器(EMAC_MACMIIDT)

以太网 MAC MII 数据寄存器(EMAC_MACMIIDT)存放的是要写入 PHY 寄存器的数据，该 PHY 寄存器位于以太网 MAC MII 地址寄存器(EMAC_MACMIIADDR)所指定的地址内。该寄存器也存放着从该 PHY 寄存器读出的值。

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	请保持默认值。
位 15: 0	MD	0x0000	rw	MII 数据 (MII Data) 这些位包含了在读操作之后从 PHY 读出的 16 位数据，或者在进行写操作之前要写入 PHY 的 16 位数据。

27.3.7 以太网MAC流控寄存器(EMAC_MACFCTRL)

以太网 MAC 流控寄存器(EMAC_MACFCTRL)通过 MAC 的流控模块管理着控制帧（Pause 命令）的生成和接收。对 Busy 位写 1 会触发流控模块生成 Pause 帧。按照 802.3x 规范选择控制帧的位域，控制帧的 Pause Time 域会使用该寄存器的 Pause Time 值。在控制帧被发送到电缆上之前，Busy 位保持置起状态。主机必须确保在写寄存器之前，Busy 位已经清除。

域	简称	复位值	类型	功能
位 31: 16	PT	0x0000	rw	Pause 时间 (Pause Time) 这些位域存放着要用在控制帧 Pause 时间域的值。如果 Pause 时间位被设置为与 MII 时钟域双重同步，那么至少要 4 个目的时钟域周期后，才允许对该寄存器连续写操作。
位 15: 8	保留	0x00	resd	请保持默认值。
位 7	DZQP	0x0	rw	关闭零值 Pause (Disable Zero-Quanta Pause) 该位被置起后，在撤销 FIFO 层流控信号时，它会关闭零值 Pause 帧的自动生成能力。 当该位被复位后，可正常操作，零值 Pause 帧的自动生成能力被启用。
位 6	保留	0x0	resd	请保持默认值。
位 5: 4	PLT	0x0	rw	Pause 低阀值 (Pause Low Threshold) 这些位设置了 Pause 定时器的阀值。 阀值应该始终小于位[31: 16]定义的 Pause 时间。例如，如果 PT = 100H (256 个时间间隙)，PLT = 01，那么如果在发送第一个 Pause 帧之后的 228 (256-28) 个时间间隙后会自动发送第二个 Pause 帧。 下表给出了不同值所对应的阀值： 00: 阀值是 Pause 时间 - 4 个时间间隙 (PT - 4 个时间间隙). 01: 阀值是 Pause 时间 - 28 个时间间隙 (PT - 28 个时间间隙). 10: 阀值是 Pause 时间 - 144 个时间间隙 (PT - 144 个时间间隙). 11: 阀值是 Pause 时间 - 256 个时间间隙 (PT - 256 个时间间隙). 时间间隙是 MII 接口发送 512 位 (64 字节) 所需要的时间。
位 3	DUP	0x0	rw	单播 Pause 帧监测 (Detect Unicast Pause Frame) 当 pause 帧具有 IEEE 802.3 规定的唯一多播地址时，将对其进行处理。该位被置起后，MAC 也能检测到带有单播地址的 Pause 帧。MAC 地址 0 高寄存器和 MAC 地址 0 低寄存器指定了单播地址。 该位被复位后，MAC 只能检测带有唯一多播地址的 Pause 帧。

				备注：如果接收帧的多播地址与唯一多播地址不一致，MAC 就不会处理 Pause 帧。
位 2	ERF	0x0	rw	使能接收流控 (Enable Receive Flow control) 该位被置起后，MAC 会解译所收到的 Pause 帧，且发送器被关闭一段时间 (Pause)。该位被复位后，Pause 帧的解译功能被关闭。
位 1	ETF	0x0	rw	使能发送流控 (Enable Transmit Flow control) 在全双工模式下，该位被置起后，MAC 使能流控功能而发送 Pause 帧。该位被复位后，MAC 的流控功能被关闭，而且 MAC 不再发送任何 Pause 帧。 在半双工模式下，该位被置起后，MAC 使能背压功能。该位被复位后，背压功能被关闭。
位 0	FCB/BPA	0x0	rw1c/rw	流控繁忙/背压激活 (Flow Control Busy/Back Pressure Activate) 在全双工模式下，该位启动 Pause 帧；在半双工模式下，当 TFE 位被置起后，背压功能被激活。 在全双工模式下，在写以太网 MAC 流控寄存器 (EMAC_MACFCTRL)之前，该位读出值为 1'b0。应用程序必须将该位置为 1'b1 才能启动 Pause 帧。在发送控制帧期间，该位始终处于置起状态，表示正在发送帧。在 Pause 帧发送完成后，MAC 将该位重置为 1'b0。在清除该位之前，不允许写以太网 MAC 流控寄存器 (EMAC_MACFCTRL)， 在半双工模式下，当该位被置起时（以及 TFE 被置起），MAC 激活背压功能。在背压工作期间，如果 MAC 接收到新的帧，发送器就开始发送 JAM 模式，发生冲突。当 MAC 被设置为全双工模式时，背压 (BPA) 功能自动关闭。

27.3.8 以太网 MAC VLAN 标签寄存器(EMAC_MACVLT)

以太网 MAC VLAN 标签寄存器(EMAC_MACVLT)包含了用来识别 VLAN 帧的 IEEE 802.1Q VLAN 标签。MAC 会将接收帧(长度/类型)的第 13 和第 14 字节与 16'h8100 进行比较，再将随后的两个字节与 VLAN 标签进行比较。如果比较结果相符，MAC 会在接收帧的状态栏里置起 VLAN 位。VLAN 帧的合法长度从 1518 字节增加到 1522 字节。

如果以太网 MAC VLAN 标签寄存器(EMAC_MACVLT)被设置成与(G)MII 时钟域双重同步，那么至少在 4 个目的时钟域周期之后才允许连续写入该寄存器。

域	简称	复位值	类型	功能
位 31: 17	保留	0x0000	resd	请保持默认值。
位 16	ETV	0x0	ro	使能 12 位 VLAN 标签比较 (Enable 12-bit VLAN tag comparison) 该位被置起后，使用 12 位而不是 16 位 VLAN 标识符进行比较和过滤。将 VLAN 标签的位[11: 0]与接收到的 VLAN 标签帧的相应域进行比较。同样地，被使能后，只使用接收帧的 12 位 VLAN 标签来进行 hash VLAN 过滤。该位被复位后，使用接收的 VLAN 帧的第 15 和第 16 字节的 16 位进行比较和 VLAN hash 过滤。
位 15: 0	VTI	0x0000	rw	VLAN 标签标识符（针对接收帧） (VLAN Tag Identifier (for receive frames)) 这些位包含了用于识别 VLAN 帧的 802.1Q VLAN 标签，并与接收到的 VLAN 帧的第 15 和第 16 字节进行比较。下表对这些位进行了详细描述： 位[15: 13]: 用户优先级 位 12: 标准格式指示器 (CFI) or 丢弃适当性指示符 (DEI) 位[11: 0]: VLAN 标签的 VLAN 指示符字段 当 ETV 位被置起时，只使用 VID(位[11: 0])来进行比较。如果 VL (如果是 ETV 被置起，则为 VL[11: 0])全为 0，MAC 不会检验用于 VLAN 标签比较的第 15 和第 16 字节，只要帧的类型域值是 0x8100 或 0x88a8，则均被视为 VLAN 帧。

27.3.9 以太网MAC远程唤醒帧过滤器寄存器(EMAC_MACRWFF)

PMT CSR 设置请求唤醒事件并监测唤醒事件。

图 27-17 以太网 MAC 唤醒帧过滤器寄存器(EMAC_MACRWFF)

Wkuppkfilter_reg0	Filter 0 Byte Mask											
Wkuppkfilter_reg1	Filter 1 Byte Mask											
Wkuppkfilter_reg2	Filter 2 Byte Mask											
Wkuppkfilter_reg3	Filter 3 Byte Mask											
Wkuppkfilter_reg4	RESD	Filter 3 Cmd	RESD	Filter 2 Cmd	RESD	Filter 1 Cmd	RESD	Filter 0 Cmd				
Wkuppkfilter_reg5	Filter 3 Offset		Filter 2 Offset		Filter 1 Offset		Filter 0 Offset					
Wkuppkfilter_reg6	Filter 1 CRC-16				Filter 0 CRC-16							
Wkuppkfilter_reg7	Filter 3 CRC-16				Filter 2 CRC-16							

27.3.10 以太网 MAC PMT 控制和状态寄存(EMAC_MACPMTCTRLSTS)

以太网 MAC PMT 控制和状态寄存器(EMAC_MACPMTCTRLSTS)设置并监控唤醒事件。

域	简称	复位值	类型	功能
位 31	RWFFPR	0x0	rw1s	远程唤醒帧过滤器寄存器指针复位 (Remote Wakeup Frame Filter Register Pointer Reset) 被置起后，该位会将远程唤醒帧过滤器寄存器指针重置为 3'b000。该位在 1 个时钟周期后自动被清除。
位 30: 10	保留	0x000000	resd	请保持默认值。
位 9	GUC	0x0	rw	全局单播 (Global UniCast) 该位被置起后，所有通过了 MAC 地址过滤的单播包均被使能为远程唤醒帧。
位 8: 7	保留	0x0	resd	请保持默认值。
位 6	RRWF	0x0	rrc	接收到远程唤醒帧 (Received Remote Wakeup Frame) 被置起后，该位表示的是因为收到远程唤醒帧而生成了电源管理事件。读该寄存器可以将该位清除。
位 5	RMP	0x0	rrc	接收到 Magic Packet (Received Magic Packet) 被置起后，该位表示的是，因为接收到 Magic packet 而生成了电源管理事件。读该寄存器可以将该位清除。
位 4: 3	保留	0x0	resd	请保持默认值。
位 2	ERWF	0x0	rw	使能远程唤醒帧 (Enable Remote Wakeup Frame) 被置起后，表示在收到远程唤醒帧时会生成电源管理事件。
位 1	EMP	0x0	rw	使能 Magic Packet (Enable Magic Packet) 被置起后，表示在收到 magic packet 时会生成电源管理事件
位 0	PD	0x0	rw1s	掉电 (Power Down) 被置起后，MAC 接收器在收到预期的 magic packet 或远程唤醒帧之前会丢弃所有接收到的帧。然后该位会自动清除并关闭掉电模式。在收到预期的 magic packet 或远程唤醒帧之前，软件也会清除该位。该位被清除后，MAC 将接收到的帧转发给应用程序。只能当 Magic Packet 使能位，全局单播位或远程唤醒帧使能位被置高时，才能将该位置起。

27.3.11 以太网 MAC 中断状态寄存器(EMAC_MACISTS)

以太网 MAC 中断状态寄存器(EMAC_MACISTS)用于识别 MAC 中能产生中断的事件。

域	简称	复位值	类型	功能
位 15: 10	保留	0x00	resd	请保持默认值。
位 9	TIS	0x0	rrc	时间戳中断状态 (Timestamp Interrupt Status) 如果该位被置起，表示系统时间值等于或超过目的时间寄存器的设定值。完成读取此位的动作后，该位被清除。
位 8: 7	保留	0x0	resd	请保持默认值。
位 6	MTIS	0x0	ro	MMC 发送中断状态 (MMC Transmit Interrupt Status) 以太网 MMC 发送中断寄存器(EMAC_MMCTI)产生中断事件后，该位被置起。清除发送中断寄存器的所有位即清除该位。
位 5	MRIS	0x0	ro	MMC 接收中断状态 (MMC Receive Interrupt Status) 当以太网 MMC 接收中断寄存器(EMAC_MMCR)产生了中断时，此位被置起。清除接收中断寄存器的所有位即清除此位。
位 4	MIS	0x0	ro	MMC 中断状态 (MMC Interrupt Status) 位[7: 5]中的任一位被置高时，该位被置起。只有当这些位都被置低时，该位才被清除。
位 3	PIS	0x0	ro	PMT 中断状态 (PMT Interrupt Status) 在掉电模式下（详见以太网 MAC PMT 控制和状态寄存器(EMAC_MACPMTCTRLSTS)的位 5 和位 6）接收到 magic packet 或远程唤醒帧时，该位置起。 如果因为读取以太网 MAC PMT 控制和状态寄存器(EMAC_MACPMTCTRLSTS)而清除了位[6: 5]，则该位被清除。
位 2: 0	保留	0x0	resd	请保持默认值。

27.3.12 以太网 MAC 中断屏蔽寄存器(EMAC_MAIMR)

以太网 MAC 中断屏蔽寄存器(EMAC_MAIMR)可以用来屏蔽因以太网 MAC 中断状态寄存器(EMAC_MACISTS)中对应事件而产生的中断信号。

域	简称	复位值	类型	功能
位 15: 10	保留	0x00	resd	请保持默认值。
位 9	TIM	0x0	rw	时间戳中断屏蔽 (Timestamp Interrupt Mask) 置起后，该位会屏蔽以太网 MAC 中断状态寄存器(EMAC_MACISTS)中时间戳中断状态位生成的中断信号。 只有使能了 IEEE1588 时间戳，该位才有效。在其它模式中，该位保留。
位 8: 4	保留	0x00	resd	请保持默认值。
位 3	PIM	0x0	rw	PMT 中断屏蔽 (PMT Interrupt Mask) 置起后，该位会屏蔽以太网 MAC 中断状态寄存器(EMAC_MACISTS)中 PMT 中断状态位生成的中断信号。
位 2: 0	保留	0x0	resd	请保持默认值。

27.3.13 以太网 MAC 地址 0 高寄存器(EMAC_MACA0H)

以太网 MAC 地址 0 高寄存器(EMAC_MACA0H)包含了设备的前 6 字节 MAC 地址的高 16 位。MII 接口收到的第一个 DA 字节对应于 MAC 地址低寄存器的 LS 字节(位[7: 0])。例如，如果将 MII 收到的（第一列的通道 0 中的 0x11）0x112233445566 视为目的地地址，那么 MacAddress0 寄存器[47: 0]用来与 0x665544332211 作比较。

如果 MAC 地址寄存器被设置成与 MII 域双重同步，那么只有向以太网 MAC 地址 0 低寄存器(EMAC_MACA0L)写入位[31: 24]（在小端模式中）或位[7: 0]（在大端模式中），才能触发同步功能。为了进行正确的同步更新，必须至少在 4 个目的时钟域周期之后，才允许对地址低寄存器进行连续写操作。

域	简称	复位值	类型	功能
位 31	AE	0x0	rrc	使能 (Address) 该位始终为 1。
位 30: 16	保留	0x0010	resd	请保持默认值。
位 15: 0	MA0H	0xFFFF	rw	MAC 地址高 (MAC Address0 [47: 32]) 该位域包含了前 6 字节 MAC 地址的高 16 位。 MAC 使用这个位域来过滤接收帧，并将 MAC 地址插入到发送流控帧 (Pause)。

27.3.14 以太网 MAC 地址 0 低寄存器(EMAC_MACA0L)

以太网 MAC 地址 0 低寄存器(EMAC_MACA0L)包含了 6 字节的第一个 MAC 地址的低 32 位。

域	简称	复位值	类型	功能
位 31: 0	MA0L	0xFFFF FFFF	rw	MAC 地址低 (MAC Address0 [31: 0]) 此位域包含了前 6 字节 MAC 地址的低 32 位。 MAC 使用这个位域来过滤所接收帧，并将 MAC 地址插入到发送流控帧 (Pause)。

27.3.15 以太网 MAC 地址 1 高寄存器(EMAC_MACA1H)

以太网 MAC 地址 1 高寄存器(EMAC_MACA1H)包含了第二个 6 字节 MAC 地址的高 16 位。

如果 MAC 地址寄存器被设置成与 MII 时钟域双重同步，只有当以太网 MAC 地址 1 低寄存器(EMAC_MACA1L)的位[31: 24] (在小端模式下) 或位[7: 0] (在大端模式下) 被写入时，才能触发同步功能。为了进行正确的同步更新，必须在至少 4 个目的时钟域周期之后，才允许向地址低寄存器连续写操作。

域	简称	复位值	类型	功能
位 31	AE	0x0	rw	地址使能 (Address Enable) 该位置起后，地址过滤器使用第二个 MAC 地址进行完美过滤。 复位后，地址过滤器将忽略要过滤的地址。
位 30	SA	0x0	rw	源地址 (Source Address) 该位置起后，MAC 地址 1[47: 0]用来与接收帧的源地址域进行比较。 复位后，MAC 地址 1[47: 0]用来与接收帧的目的地址域进行比较。
位 29: 24	MBC	0x00	rw	屏蔽字节控制 (Mask Byte Control) 这些位是指用于比较每个 MAC 地址字节的屏蔽控制位。 置起时，MAC 不会将接收到的目的地址或源地址的对应字节与 MAC 地址 1 寄存器的内容进行比较。每个控制位用来控制字节的屏蔽，如下所示： 位 29: EMAC_MACA1H [15: 8] 位 28: EMAC_MACA1H [7: 0] 位 27: EMAC_MACA1L[31: 24] ... 位 24: EMAC_MACA1L[7: 0] 可以通过屏蔽该地址的一个或多个字节来过滤一组地址 (称为成组地址过滤)
位 23: 16	保留	0x00	resd	请保持默认值。
位 15: 0	MA1H	0xFFFF	rw	MAC Address1 [47: 32] 这些位包含了第二个 6 字节 MAC 地址的高 16 位(47: 32)。

27.3.16 以太网 MAC 地址 1 低寄存器(EMAC_MACA1L)

以太网 MAC 地址 1 低寄存器(EMAC_MACA1L)包含了第二个 6 字节 MAC 地址的低 32 位。

域	简称	复位值	类型	功能
---	----	-----	----	----

位 31: 0 MA1L	0xFFFF FFFF rw	MAC Address1 [31: 0] 这些位包含了第二个 6 字节 MAC 地址的低 32 位。在初始化流程后需要应用程序加载，才会定义这个位域的内容。
--------------	----------------	---

27.3.17 以太网 MAC 地址 2 高寄存器(EMAC_MACA2H)

以太网 MAC 地址 2 高寄存器(EMAC_MACA2H)包含了第二个 6 字节 MAC 地址的高 16 位。如果 MAC 地址寄存器被设置为与 MII 时钟域双重同步，那么当以太网 MAC 地址 2 低寄存器(EMAC_MACA2L)的位[31: 24] (小端模式) 或位[7: 0] (大端模式) 被写入时，才会触发同步功能。为了进行正确地同步更新，至少在 4 个目的时钟域周期之后，才允许对地址低寄存器进行连续写操作。

域	简称	复位值	类型	功能
位 31	AE	0x0	rw	地址使能 (Address Enable) 该位置起后，地址过滤器模块使用第二个 MAC 地址进行完美过滤。 复位后，地址过滤器模块会忽略要过滤的地址。
位 30	SA	0x0	rw	源地址 (Source Address) 该位置起后，使用 MAC 地址 1[47: 0]与接收帧的源地址域进行比较。 复位后，使用 MAC 地址 1[47: 0]与接收帧的目的地址域进行比较。
位 29: 24	MBC	0x00	rw	屏蔽字节控制 (Mask Byte Control) 这些位是指用于比较每个 MAC 地址的屏蔽控制位。置起时，MAC 不会将接收到的源地址或目的地址的对应字节与 MAC 地址 1 寄存器的内容进行比较。每个位用来控制字节的屏蔽，如下所示： 位 29: EMAC_MACA2H [15: 8] 位 28: EMAC_MACA2H [7: 0] 位 27: EMAC_MACA2L [31: 24] ... 位 24: EMAC_MACA2L[7: 0] 可以通过屏蔽该地址的一个或多个字节来过滤一组地址 (称为成组地址过滤)
位 23: 16	保留	0x00	resd	请保持默认值。
位 15: 0	MA2H	0xFFFF	rw	(MAC Address2 High [47: 32]) 这些位包含了第二个 6 字节 MAC 地址的高 16 位 (47: 32)。

27.3.18 以太网 MAC 地址 2 低寄存器(EMAC_MACA2L)

以太网 MAC 地址 2 低寄存器(EMAC_MACA2L)包含了第二个 6 字节 MAC 地址的低 32 位。

域	简称	复位值	类型	功能
位 31: 0	MA2L	0xFFFF FFFF rw		(MAC Address2 Low [31: 0]) 这些位包含了第二个 6 字节 MAC 地址的低 32 位。 在初始化流程之后，需要应用程序加载，才会定义这些位的内容。

27.3.19 以太网 MAC 地址 3 高寄存器(EMAC_MACA3H)

以太网 MAC 地址 3 高寄存器(EMAC_MACA3H)包含了第二个 6 字节 MAC 地址的高 16 位。如果 MAC 地址寄存器被设置为与 MII 时钟域双重同步，那么当以太网 MAC 地址 3 低寄存器(EMAC_MACA3L)的位[31: 24] (小端模式) 或位[7: 0] (大端模式) 被写入时，才会触发同步功能。为了进行正确地同步更新，至少在 4 个目的时钟域周期之后，才允许对地址低寄存器进行连续写操作。

域	简称	复位值	类型	功能
位 31	AE	0x0	rw	地址使能 (Address Enable) 该位置起后，地址过滤器使用第二个 MAC 地址进行完美过滤。

				复位后，地址过滤器将忽略要过滤的地址。
位 30	SA	0x0	rw	源地址 (Source Address) 该位置起后，使用 MAC 地址 1[47: 0]与接收帧的源地址域进行比较。 复位后，使用 MAC 地址 1[47: 0]与接收帧的目的地址域进行比较。
位 29: 24	MBC	0x00	rw	屏蔽字节控制 (Mask Byte Control) 这些位是指用于比较每个 MAC 地址的屏蔽控制位。置起时，MAC 不会将接收到的源地址或目的地址的对应字节与 MAC 地址 1 寄存器的内容进行比较。每个位用来控制字节的屏蔽，如下所示： 位 29: EMAC_MACA3H [15: 8] 位 28: EMAC_MACA3H [7: 0] 位 27: EMAC_MACA3L [31: 24] ... 位 24: EMAC_MACA3L [7: 0] 可以通过屏蔽该地址的一个或多个字节来过滤一组地址 (称为成组地址过滤)
位 23: 16	保留	0x00	resd	请保持默认值。
位 15: 0	MA3H	0xFFFF	rw	(MAC Address3 High [47: 32]) 这些位包含了第二个 6 字节 MAC 地址的低 16 位(47: 32)。

27.3.20 以太网 MAC 地址3低寄存器(EMAC_MACA3L)

以太网 MAC 地址 3 低寄存器(EMAC_MACA3L)包含了第二个 6 字节 MAC 地址的低 32 位。

域	简称	复位值	类型	功能
位 31: 0	MA3L	0xFFFF FFFF	rw	(MAC Address3 Low [31: 0]) 这些位包含了第二个 6 字节 MAC 地址的低 32 位。在初始化流程后需要应用程序加载，才会定义这个位域的内容。

27.3.21 以太网 DMA 总线模式寄存器(EMAC_DMABM)

以太网 DMA 总线模式寄存器(EMAC_DMABM)设定了 DMA 的总线工作模式。

域	简称	复位值	类型	功能
位 31: 26	保留	0x00	resd	请保持默认值。
位 25	AAB	0x0	rw	地址对齐 (Address-Aligned Beats) 当该位被置起且 FB 位等于 1 时，AHB 接口会生成与起始地址 LS 位对齐的突发传输。如果 FB 位等于 0，第一次突发传输（即访问数据缓冲器的起始地址）可能不会与地址对齐，但是随后的突发传输会与地址对齐。 该位仅适用于 GMAC-AHB 和 GMAC-AXI 配置，在其它配置中，该位是保留的。
位 24	PBLx8	0x0	rw	PBLx8 模式 (PBLx8 Mode) 被置起后，该位将设置好的 PBL 值（位 [22: 17] 和位 [13: 8]）乘以 8。所以，DMA 根据 PBL 的值以 8, 16, 32, 64, 128, 和 256 节拍转发数据，
位 23	USP	0x0	rw	使用分散的 PBL (Use Separate PBL) 被置起后，Rx DMA 使用位[22: 17]的设定值作为 PBL。位[13: 8]中的 PBL 值仅适用于 DMA 发送操作。被复位后，位[13: 8]的 PBL 值既适用于 DMA 接收操作也适用于 DMA 发送操作。
位 22: 17	RDP	0x01	rw	接收 DMA PBL (Rx DMA PBL) 这些位域表示的是一次 DMA 接收操作所传输的最大节拍数。这个最大值用于单次读或写操作。
				接收 DMA 每次在主机总线上进行突发传输时，总会尝试按照 RPBL 的设定值进行突发传输。可以使用 1, 2, 4, 8, 16 和 32 来设置 RPBL。除此以外的其它值均会导致意外情况发生。

				该位域仅适用于 USP 被置起时的情况。
位 16	FB	0x0	rw	固定的突发传输 (Fixed Burst) 该位控制着 AHB 主机接口是否进行固定突发传传输。该位被置起后，AHB 接口在开始正常的突发传输过程中，仅使用 SINGLE, INCR4, INCR8 或 INCR16。复位后，AHB 或 AXI 接口使用 SINGLE 和 INCR 突发传输。
位 15: 14	PR	0x0	rw	优先级比率 (Priority Ratio) 这些位控制着接收 DMA 和发送 DMA 之间的加权循环仲裁的优先级比率。这些位仅在位 1 (目的地址) 被复位时才有效。优先级比率为 Rx: Tx 或者 Tx: Rx, 具体取决于 27(TXPR) 是置起还是复位。 00: 优先级比率为 1: 1. 01: 优先级比率为 2: 1. 10: 优先级比率为 3: 1. 11: 优先级比率为 4: 1.
位 13: 8	PBL	0x01	rw	可编程的突发长度 (Programmable Burst Length) 这些位定义了一次 DMA 操作中所传输的最大节拍数。这个最大值用于单次读写操作。 DMA 每次在主机总线上进行突发传输时，总会尝试按照 PBL 的设定值进行突发传输。PBL 允许设置为 1, 2, 4, 8, 16 和 32。除此以外的其它值均会导致意外情况发生。当 USP 置起时，PBL 值仅适用于 DMA 发送操作。 如果要传输的节拍数量大于 32，需按以下步骤操作： 1. 设置 PBLx8 模式。 2. 设置 PBL。 例如，如果待传输的最大值为 64，那么首先需要将 PBLx8 置 1，再将 PBL 设置为 8. PBL 的值有以下限制：可能的最大传输数会受到 MTL 层的发送 FIFO 和接收 FIFO 大小的限制，以及 DMA 上的数据总线宽度的限制。 FIFO 的限制：FIFO 所支持的最大传输次数是 FIFO 深度的一半，除非另有规定
位 7	EDE	0x0	rw	增强描述符使能 (Enhanced descriptor enable) 该位置 1 时，使能增强描述符功能。增强描述符总共有 8 个 WORD。增强描述符的定义请参考 TX 增强描述符和 RX 增强描述符。
位 6: 2	DSL	0x00	rw	描述符跳跃长度 (Descriptor Skip Length) 这些位定义了两个未链接的描述符之间要跳过的字距离。地址跳跃是指从当前描述符的末尾跳到下一个描述的开头。 当 DSL 值等于 0 时，在环形模式下 DMA 将描述符视为连续的。
位 1	DA	0x0	rw	DMA 仲裁 (DMA Arbitration) 该位定义了通道 0 的发送路径和接收路径之间的仲裁方案。 0: Rx: Tx 或者 Tx: Rx 加权循环 通道之间的优先级取决于位[15: 14] (PR) 所定义的优先级以及位 27(TXPR) 所定义的优先级权重。 1: 固定的优先级 当位 27(TXPR) 被置起时，发送路径的优先级高于接收路径。反之，接收路径高于发送路径。
位 0	SWR	0x1	rw	软件复位 (SoftWare Reset) 该位被置起后，MAC DMA 控制器将 MAC 逻辑电路和所有内部寄存器进行复位。在完成了所有复位操作之后，该位将自动清除。

27.3.22 以太网 DMA 发送轮询请求寄存器(EMAC_DMATPD)

以太网 DMA 发送轮询请求寄存器(EMAC_DMATPD)使能发送 DMA 来查询 DMA 是否拥有当前描述符。发送轮询请求可以用来唤醒处于暂停模式下的发送 DMA。发送 DMA 会因为发送的帧出现下溢错误或者未获得可用的描述符而进入暂停模式。可以在任何时刻发出轮询指令，发送 DMA 在重新开始从主机

内存中取回当前描述符时，会将这一轮询指令复位。该寄存器始终读 0.

域	简称	复位值	类型	功能
位 31: 0	TPD	0x0000 0000	rrc	发送轮询请求 (Transmit Poll Demand) 当这些位被写入任意值时，DMA 读取 EMAC_DMACTD 指向的当前描述符。如果描述符不可用（被主机占用），发送动作暂停，状态寄存器的位 2 (TU) 会置起。如果描述符可用，则恢复发送动作。

27.3.23 以太网 DMA 接收轮询请求寄存器(EMAC_DMARPD)

太网 DMA 接收轮询请求寄存器(EMAC_DMARPD)使能接收 DMA 来查询新的描述符。接收轮询请求可以用来唤醒处于暂停模式下的接收 DMA。接收 DMA 会因为描述符不可用而进入暂停模式。该寄存器始终读 0.

域	简称	复位值	类型	功能
位 31: 0	RPD	0x0000 0000	rrc	接收轮询请求 (Receive Poll Demand) 当这些位被写入任意值时，DMA 读取 EMAC_DMACRD 指向的当前描述符。如果描述符不可用（被主机占用），接收动作暂停，状态寄存器的位 7 (RU) 会置起。如果描述符可用，则恢复接收动作。

27.3.24 以太网 DMA 接收描述符列表地址寄存器 (EMAC_DMARDLADDR)

以太网 DMA 接收描述符列表地址寄存器(EMAC_DMARDLADDR)指向接收描述符列表的开头。描述符列表位于主机物理内存内，并且必须与字对齐。DMA 通过将相应的 LS 位置低将地址对齐总线宽度。只有当接收 DMA 停止的时候，才允许对该寄存器进行写操作。在接收 DMA 停止后，必须先写入该寄存器，再发出接收开始指令。

只有在接收 DMA 停止后，才能对该寄存器进行写操作，即将操作模式寄存器的位 1(SR)置 0. 在接收 DMA 停止之后，该寄存器可以写入一个新的描述符列表地址。

当 SR 位被置 1 后，DMA 将使用最新编写的描述符基址。

如果在 SR 位被置 0 后，该寄存器没有更改，那么 DMA 将使用先前 DMA 接收停止时的描述符地址。

域	简称	复位值	类型	功能
位 31: 0	SRL	0x0000 0000	rw	接收描述符基地址 (Start of Receive List) 这些位包含了接收描述符列表中第一个描述符的基地址。 32 位，64 位或 128 位总线宽度的 LSB 位(1: 0, 2: 0 或 3: 0)被忽略，被 DMA 视为 0. 因此，这些 LSB 位是只读的。

27.3.25 以太网 DMA 发送描述符列表地址寄存器 (EMAC_DMATDLADDR)

以太网 DMA 发送描述符列表地址寄存器(EMAC_DMATDLADDR)指向发送描述符列表的开头。描述符列表位于主机物理内存内，并且必须与字对齐。DMA 通过将相应的 LS 位置低将地址对齐总线宽度。

只有当发送 DMA 停止的时候，才允许写入该寄存器，即，将寄存器 6 (操作模式寄存器) 中的位 13(ST) 置为 0. 在发送 DMA 停止后，可以向该寄存器写入一个新的描述符列表地址。

当 SR 位被置 1 后，DMA 将使用最新编写的描述符基址。

如果在 SR 位被置 0 后，该寄存器没有更改，那么 DMA 将使用先前 DMA 发送停止时的描述符地址。

域	简称	复位值	类型	功能
位 31: 0	STL	0x0000 0000	rw	发送描述符基地址 (Start of Transmit List) 这些位包含了发送描述符列表中第一个描述符的基地址。 32 位，64 位或 128 位总线宽度的 LSB 位(1: 0, 2: 0 或 3: 0)被忽略，被 DMA 视为 0. 因此，这些 LSB 位是只读的。

27.3.26 以太网 DMA 状态寄存器(EMAC_DMASTS)

以太网 DMA 状态寄存器(EMAC_DMASTS)包含了 DMA 报告给主机的所有状态位。软件驱动在中断服务程序或者轮询过程中会读取该寄存器。该寄存器中的大多数位都能中断主机。读取该寄存器不能清除其中的位。向该寄存器中的位[16: 0]（非保留位）写入 1'b1 才能清除这些位，写入 1'b0 不起作用。通过中断使能屏蔽寄存器中的相应位，可以屏蔽每一个位（位[16: 0]）。

域	简称	复位值	类型	功能
位 31: 30	保留	0x0	resd	请保持默认值。
位 29	TTI	0x0	ro	时间戳触发中断 (Timestamp Trigger Interrupt) 该位表示时间戳生成模块发生了中断事件。软件必须读取相应寄存器才能获取中断源。 该位仅适用于当 IEEE1588 时间戳功能被使能的情况。反之，该位是保留的。
位 28	MPI	0x0	ro	MAC PMT 中断 (MAC PMT Interrupt) 该位表示 PMT 模块发生了中断事件。软件必须读取以太网 MAC PMT 控制和状态寄存器 (EMAC_MACPMTCTRLSTS) 才能获取中断源并清除中断源，以将此位复位为 1'b0。 该位仅适用于当电源管理功能被使能的情况。反之，该位是保留的。
位 27	MMI	0x0	ro	MAC MMC 中断 (MAC MMC Interrupt) 该位表示 MMC 模块发生了中断事件。软件必须读取对应寄存器才能获取中断源并清除中断源，以将此位复位为 1'b0。 该位仅适用于当 MAC 管理计数器 MMC 被使能的情况。反之，该位是保留的。
位 26	保留	0x0	resd	请保持默认值。
位 25: 23	EB	0x0	ro	错误位 (Error Bits) 这些位表示的是造成总线错误的错误类型。仅适用于当位 13 (FBI) 被置起的情况。该位域不会产生中断。 000: 在 Rx DMA 传输写数据时出错 011: 在 Tx DMA 传输读数据时出错 100: 在 Rx DMA 描述符写访问时出错 101: 在 Tx DMA 描述符写访问时出错 110: 在 Rx DMA 描述符读访问时出错 111: 在 Tx DMA 描述符读访问时出错 注意: 001 和 010 表示保留。
位 22: 20	TS	0x0	ro	发送流程状态 (Transmit Process State) 该位域表示发送 DMA 的 FSM 状态。这些位不会产生中断。 3' b000: 停止; 发出复位或停止发送命令 3' b001: 运行; 正在提取发送描述符 3' b010: 运行; 正在等待状态信息 3' b011: 运行; 正在读取主机内存缓存数据并将其排队到发送缓存(Tx FIFO) 3' b100: 时间戳写状态 3' b101: 保留备用 3' b110: 暂停; 发送描述符不可用或发送缓冲器下溢 3' b111: 运行; 正在关闭发送描述符
位 19: 17	RS	0x0	ro	接收流程状态 (Receive Process State) 该位域表示接收 DMA 的 FSM 状态。这些位不会产生中断。 3' b000: 停止; 发出复位或停止发送命令 3' b001: 运行; 正在提取接收描述符 3' b010: 保留备用 3' b011: 运行; 正在等待接收数据包 3' b100: 暂停; 接收描述符不可用 3' b101: 运行; 正在关闭描述符 3' b110: 时间戳写状态 3' b111: 运行; 将接收缓冲器的数据包转发到主机内存
位 16	NIS	0x0	rw1c	正常中断汇总 (Normal Interrupt Summary)

				当中断使能寄存器中的相应中断位被使能时，正常中断汇总的值是下列位的逻辑或： EMAC_DMASTS[0]: 发送中断 EMAC_DMASTS[2]: 发送缓冲不可用 EMAC_DMASTS[6]: 接收中断 EMAC_DMASTS[14]: 早接收中断 只有未被屏蔽的位才会影响到正常中断汇总位。 该位是粘滞位，必须在每次清除相应位（导致 NIS 被置起）时被清除（向该位写 1）。
位 15	AIS	0x0	rw1c	异常中断汇总 (Abnormal Interrupt Summary) 当中断使能寄存器中的相应中断位被使能时，异常中断汇总的值是下列位的逻辑或 EMAC_DMASTS[1]: 发送流程停止 EMAC_DMASTS[3]: 发送 Jabber 超时 EMAC_DMASTS[4]: 接收 FIFO 溢出 EMAC_DMASTS[5]: 发送数据下溢 EMAC_DMASTS[7]: 接收缓冲不可用 EMAC_DMASTS[8]: 接收流程停止 EMAC_DMASTS[9]: 接收看门狗超时 EMAC_DMASTS[10]: 早发送中断 EMAC_DMASTS[13]: 线致命错误 只有未被屏蔽的位才会影响到异常中断汇总位。 该位是粘滞位，必须在每次清除相应位（导致 AIS 被置起）时被清除（向该位写 1）。
位 14	ERI	0x0	rw1c	早接收中断 (Early Receive Interrupt) 该位表示 DMA 填充了数据包的第一个数据缓存。当软件向该位写 1 或者置起该寄存器的位 6(RI)时（以先发生的为准），即可清除该位。
位 13	FBEI	0x0	rw1c	总线致命错误中断 (Fatal Bus Error Interrupt) 该位表示发生了位[25: 23]所描述的总线错误。该位被置起后，对应的 DMA 将关闭全部的总线访问。
位 12: 11	保留	0x0	resd	请保持默认值。
位 10	ETI	0x0	rw1c	早发送中断 (Early Transmit Interrupt) 该位表示要发送的帧已经全部被转送到 MTL 发送 FIFO。
位 9	RWT	0x0	rw1c	接收看门狗超时 (Receive Watchdog Timeout) 该位被置起后，该位表示在接收看门狗定时器在接收当前帧时到期，并且在看门狗发生超时后，当前帧被截断。
位 8	RPS	0x0	rw1c	接收流程停止 (Receive Process Stopped) 当接收流程进入停止状态时，该位置起。
位 7	RBU	0x0	rw1c	接收缓冲不可用 (Receive Buffer Unavailable) 该位表示主机占用了接收列表中的下一个描述符，使得 DMA 无法获取。因此接收流程暂停。主机需要更改描述符的所有权，并释放接收轮询请求指令才能恢复接收流程。如果未释放接收轮询请求指令，那么 DMA 在收到一个输入帧的时候，会恢复接收流程。只有在 DMA 占用了前一个接收描述符的情况下，该位才会置起。
位 6	RI	0x0	rw1c	接收中断 (Receive Interrupt) 该位表示帧接收已完成。接收完成后，将在最后一个描述符中复位 RDES1(完成后关闭中断)的位 31，具体的帧状态信息会更新到描述符中。 接收流程仍处于运行状态。
位 5	UNF	0x0	rw1c	发送数据下溢 (Transmit Underflow) 该位表示发送缓存在发送帧期间发生数据下溢。发送流程暂停，并且将下溢错误 TDES0[1]置起。
位 4	OVF	0x0	rw1c	接收溢出 (Receive Overflow) 该位表示接收缓存在接收帧期间发生数据溢出。如果部分帧已转发到应用程序，RDES0[11]位会置起溢出状态
位 3	TJT	0x0	rw1c	发送超长帧 (即 Jabber) 超时 (Transmit Jabber Timeout) 该位表示当帧大小超过 2048 字节（若 Jumbo 帧被使能，则为 10240 字节）时会发生发送 Jabber 定时器超时。

				Jabber发生超时后，发送流程中止并进入停止状态。这将导致发送 Jabber 超时标志位 TDES0[14]被置起。
位 2	TBU	0x0	rw1c	发送缓冲不可用 (Transmit Buffer Unavailable) 该位表示主机占用了发送列表中的下一个描述符，使得 DMA 无法获取，因此，发送流程暂停。位[22: 20]解释了发送流程状态。如果需要恢复发送流程，可以通过设置 TDES0[31]更改描述符的所有权并释放发送轮询请求指令。
位 1	TPS	0x0	rw1c	发送流程停止 (Transmit Process Stopped) 当发送流程停止时，该位置起。
位 0	TI	0x0	rw1c	(Transmit Interrupt)发送中断 该位表示帧发送已完成。发送完成后，TDES0 的位 Bit 31 (OWN)会复位，具体的帧状态信息会更新到描述符。

27.3.27 以太网 DMA 工作模式寄存器(EMAC_DMAOPM)

以太网 DMA 工作模式寄存器(EMAC_DMAOPM)定义了发送和接收的工作模式和指令。该寄存器应该是 DMA 初始化过程中最后需要写入的 CSR。该寄存器也适用于 GMAC-MTL 配置，其中，未使用和保留的位是 24, 13, 2 和 1。对此寄存器连续的写之间要插入延时，延时值大于 4us。

域	简称	复位值	类型	功能
位 31: 27	保留	0x00	resd	请保持默认值。
位 26	DT	0x0	rw	不丢弃 TCP/IP 校验和错误帧 (Disable Dropping of TCP/IP Checksum Error Frames) 该位置起后，对于仅由接收校验和减负引擎检测到的有错误的帧，MAC 不会丢弃该帧。这些帧仅在封装数据里会出错，而在 MAC 接收到的以太网帧中不会出错（包含 FCS 错误）。该位复位后，如果 FEF 位复位，则所有错误帧均被丢弃。
位 25	RSF	0x0	rw	接收存储和转发 (Receive Store and Forward) 该位置起后，仅在一个完整的帧写入 Rx FIFO 后，MTL 才会读取 Rx FIFO 的帧，并忽略 RTC 位。该位复位后，Rx FIFO 工作在直通模式下，会受 RTC 指定阀值的限制。
位 24	DFRF	0x0	rw	不清除接收帧 (Disable Flushing of Received Frames) 该位置起后，接收 DMA 不会因为接收描述符或接收缓存不可用而清除接收帧。当该位复位后，遭遇上述情况时，接收 DMA 会清除接收帧。
位 23: 22	保留	0x000	resd	请保持默认值。
位 21	TSF	0x0	rw	发送存储和转发 (Transmit Store and Forward) 该位置起后，当一个完整的帧位于 MTL 发送 FIFO 时，就会开始发送，并且位[16: 14]设定的 TTC 值被忽略。该位的状态仅在发送流程停止时才允许更改。
位 20	FTF	0x0	rw	清空发送 FIFO (Flush Transmit FIFO) 该位被置起后，发送 FIFO 控制器逻辑电路被恢复至默认值，发送 FIFO 中的所有数据要么丢失，要么被清空。在完成清空动作后，该位即被清除。在该位被清除之前，不允许向工作模式寄存器进行写操作。MAC 发送器已接受的数据是不会被清空的，会被安排发送，并导致数据下溢和超短帧传输。
位 19: 17	保留	0x0	resd	请保持默认值。
位 16: 14	TTC	0x0	rw	发送阀值控制 (Transmit Threshold Control) 这些位控制的是 MTL 发送 FIFO 的阀值。当 MTL 发送 FIFO 中的帧超过阀值时，开始发送。另外，长度小于该阀值的完整帧也会被发送。这些位仅适用于当位 21(TSF)复位的情况下。 000: 64 001: 128 010: 192 011: 256 100: 40

				101: 32 110: 24 111: 16
位 13	SSTC	0x0	rw	<p>开始或停止发送指令 (Start or Stop Transmission Command)</p> <p>该位被置起后，发送流程处于运行状态，DMA 会检查当前位置的发送列表，确定待发送的帧。DMA 要么从当前列表位置获取描述符（即发送描述符列表地址寄存器所设定的发送列表基址），要么从之前发送流程中止的位置获取描述符。如果 DMA 未占用当前描述符，发送流程进入暂停状态，并且状态寄存器的位 2（发送缓冲不可用）会被置起。发送开始指令仅在发送停止的时候才有效。如果在未设置发送描述符列表地址寄存器之前就发出了发送开始指令，则 DMA 将会出现意想不到的后果。</p> <p>复位后，在发送完当前帧后，发送流程进入停止状态。保存发送列表中的下一个描述符位置，并在重新开始发送时，将该描述符位置作为当前位置。若要修改列表地址，需要在该位被复位时向发送描述符列表地址寄存器编写一个新值。该位被重新置起时，这个写入的新值会生效。只有在当前帧已发送完成或者发送流程进入暂停状态时，停止发送指令才会生效。</p>
位 12: 8	保留	0x00	resd	请保持默认值。
位 7	FEF	0x0	rw	<p>转发错误帧 (Forward Error Frames)</p> <p>1: 表示除了过短帧以外，所有的帧都会转发给 DMA。 0: 接收 FIFO 会丢弃有错误的帧(CRC 错误、冲突错误、巨人帧、看门狗超时、溢出)。然而，如果在阈值模式下，已经把帧的起始字节指针转发给应用程序，则就不会丢弃该帧。接收 FIFO 会丢弃那些帧的起始字节还没有发送到 AHB 总线的出错帧。</p>
位 6	FUGF	0x0	rw	<p>转发长度偏小的好帧 (Forward Undersized Good Frames)</p> <p>该位被置起后，接收 FIFO 会转发填充字节和 CRC 等长度偏小的好帧（即没有错误且长度小于 64 字节的帧） 复位后，接收 FIFO 会丢弃所有长度小于 64 字节的帧，除非这个帧因为小于接收阀值而被传输给了应用程序，比如 RTC = 01</p>
位 5	保留	0x0	resd	请保持默认值。
位 4: 3	RTC	0x0	rw	<p>接收阀值控制 (Receive Threshold Control)</p> <p>这两个位是用来控制 MTL 接收 FIFO 的阀值。当 MTL 接收 FIFO 里的帧大于阀值时，开始向 DMA 发送请求。此外，长度小于阀值的完整帧也会自动发送。 如果接收 FIFO 大小被设定为 128 字节，则值 11 不适用。这些位仅适用于当 RSF 位等于 0 的情况，当 RSF 位置 1 时，这些位被忽略。</p> <p>00: 64 01: 32 10: 96 11: 128</p>
位 2	OSF	0x0	rw	<p>操作第二帧 (Operate on Second Frame)</p> <p>该位被置起后，它将指导 DMA 在获得第一帧的状态之前就开始处理发送数据的第二帧。</p>
位 1	SSR	0x0	rw	<p>发送或停止接收 (Start or Stop Receive)</p> <p>该位被置起后，接收流程处于运行状态，DMA 会尝试从接收列表获取描述符并处理输入的帧。DMA 要么从当前列表位置获取描述符（即接收描述符列表地址寄存器所设定的地址），要么从之前接收流程中止的位置获取描述符。如果 DMA 未占用当前描述符，接收流程进入暂停状态，并且状态寄存器的位 7（接收缓冲不可用）会被置起。接收开始指令仅在接收停止的时候才有效。如果在未设置接收描述符列表地址寄存器之前就发出了接收开始指令，则 DMA 将会出现意想不到的后果。</p>

该位被清除后，在发送完当前帧后，接收 DMA 操作停止。保存接收列表中的下一个描述符位置，并在重新开始接收时，将该描述符位置作为当前位置。只有在接收流程进入运行状态（正等着接收数据包）或者暂停状态时，停止接收指令才会生效。

位 0	保留	0x0	resd	请保持默认值。
-----	----	-----	------	---------

27.3.28 以太网 DMA 中断使能寄存器(EMAC_DMAIE)

以太网 DMA 中断使能寄存器(EMAC_DMAIE)将使能由状态寄存器反馈的中断。把相应的位置 1'b1 可以使能相应的中断。在软件或硬件复位后，所有的中断关闭。

域	简称	复位值	类型	功能
位 31: 17	保留	0x0000	resd	请保持默认值。
位 16	NIE	0x0	rw	使能正常中断 (Normal Interrupt enable) 该位被置起后，正常中断汇总被启用。该位复位后，正常中断汇总关闭。该位可以使能寄存器 5 (状态寄存器) 中的下列中断： EMAC_DMASTS[0]: 发送中断 EMAC_DMASTS[2]: 发送缓冲不可用 EMAC_DMASTS[6]: 接收中断 EMAC_DMASTS[14]: 早接收中断
位 15	AIE	0x0	rw	使能异常中断 (Abnormal interrupt enable) 该位被置起后，异常中断汇总被启用。复位后，异常中断汇总关闭。该位可以使能状态寄存器中的下列中断： EMAC_DMASTS[1]: 发送流程停止 EMAC_DMASTS[3]: 发送 Jabber 超时 EMAC_DMASTS[4]: 接收溢出 EMAC_DMASTS[5]: 发送数据下溢 EMAC_DMASTS[7]: 发送缓冲不可用 EMAC_DMASTS[8]: 接收流程停止 EMAC_DMASTS[9]: 接收看门够超时 EMAC_DMASTS[10]: 早发送中断 EMAC_DMASTS[13]: 总线致命错误
位 14	ERE	0x0	rw	早接收中断使能 (Early Receive interrupt Enable) 通过正常中断汇总使能 (位 16) 将该位置起后，早接收中断被使能。复位后，早接收中断关闭。
位 13	FBEE	0x0	rw	总线致命错误使能 (Fatal Bus Error Enable) 通过异常中断汇总使能 (位 15) 将该位置起后，总线致命错误被使能。复位后，总线致命错误使能中断关闭。
位 12: 11	保留	0x0	resd	请保持默认值。
位 10	EIE	0x0	rw	早发送中断使能 (Early transmit Interrupt Enable) 通过异常中断汇总使能 (位 15) 将该位置起后，早发送中断被使能。复位后，早发送中断关闭。
位 9	RWTE	0x0	rw	接收看门狗超时使能 (Receive Watchdog Timeout Enable) 通过异常中断汇总使能 (位 15) 将该位置起后，接收看门狗超时中断被使能。复位后，接收看门狗超时中断关闭。
位 8	RSE	0x0	rw	接收停止使能 (Receive Stopped Enable) 通过异常中断汇总使能 (位 15) 将该位置起后，接收停止中断被使能。复位后，接收停止中断关闭。
位 7	RBUE	0x0	rw	接收缓存不可用使能 (Receive Buffer Unavailable Enable) 通过异常中断汇总使能 (位 15) 将该位置起后，接收缓存不可用中断被使能。复位后，接收缓存不可用中断关闭。
位 6	RIE	0x0	rw	接收中断使能 (Receive Interrupt Enable) 通过正常中断汇总使能 (位 16) 将该位置起后，接收中断被使能。复位后，接收中断关闭
位 5	UNE	0x0	rw	下溢中断使能 (Underflow Interrupt Enable)

				通过异常中断汇总使能（位 15）将该位置起后，下溢中断被使能。复位后，下溢中断关闭。
位 4	OVE	0x0	rw	溢出中断使能（Overflow Interrupt Enable） 通过异常中断汇总使能（位 15）将该位置起后，溢出中断被使能。复位后，溢出中断关闭。
位 3	TJE	0x0	rw	发送 Jabber 超时使能（Transmit Jabber Timeout Enable） 通过异常中断汇总使能（位 15）将该位置起后，发送 Jabber 超时中断被使能。复位后，发送 Jabber 超时中断关闭。
位 2	TUE	0x0	rw	发送缓存不可用使能（Transmit Buffer Unavailable Enable） 通过正常中断汇总使能（位 16）将该位置起后，发送缓存不可用中断被使能。复位后，发送缓存不可用中断关闭。
位 1	TSE	0x0	rw	发送停止使能（Transmit Stopped Enable） 通过异常中断汇总使能（位 15）将该位置起后，发送停止中断被使能。复位后，发送停止中断关闭。
位 0	TIE	0x0	rw	发送中断使能（Transmit Interrupt Enable） 通过正常中断汇总使能（位 16）将该位置起后，发送中断被使能。复位后，发送中断关闭。

对于以太网中断，只有在 DMA 状态寄存器的 TST 位或者 PMT 位为‘1’，并且相应中断没有被屏蔽时，或者在 NIS/AIS 位置‘1’，且相应中断被使能时，中断才会发生。

27.3.29 以太网 DMA 丢失帧和缓存溢出计数器寄存器 (EMAC_DMAMFBOCNT)

DMA 包含两个计数器来追踪接收流程中丢失的帧数目。该寄存器反馈计数器的当前值。计数器用于诊断故障。位[15: 0]表示由于主机缓存不可用而丢失的帧数目。位[27: 17]表示由于缓存溢出(MTL 和 MAC) 及被 MTL 丢弃的超短帧(长度小于 64 字节的好帧)而丢失的帧数目。

域	简称	复位值	类型	功能
位 31: 29	保留	0x0	resd	请保持默认值。
位 28	OBFOC	0x0	rrc	FIFO 溢出计数器溢出位（Overflow Bit for FIFO Overflow Counter） 每一次溢出帧计数器(位[27: 17])发生溢出时，该位就会被置起，即接收 FIFO 发生溢出，溢出帧计数器达到最大值。在这种情况下，溢出帧计数器被重置为全 0，该位表示已发生翻转。
位 27: 17	OFC	0x000	rrc	溢出帧计数器（Overflow Frame Counter） 这些位域表示应用程序所丢失的帧数目。
位 16	OBMFC	0x0	rrc	丢失帧计数器溢出位（Overflow Bit for Missed Frame Counter） 每当丢失帧计数器（位[15: 0]）发生溢出时，该位被置起，即 DMA 会忽略由于主机接收缓存不可用而传入的帧，且丢失帧计数器达到最大值。在这种情况下，丢失帧计数器被重置为全 0，该位代表翻转已发生。
位 15: 0	MFC	0x0000	rrc	丢失帧计数器（Missed Frame Counter） 这些位域表示由于主机接收缓存不可用而被控制器丢失的帧数目。每当 DMA 忽略一个传入的帧时，该计数器就会递增一次。

27.3.30 以太网 DMA 当前发送描述符寄存器(EMAC_DMACTD)

当前主机发送描述符寄存器指向 DMA 正在读取的发送描述符的起始地址。

域	简称	复位值	类型	功能
位 31: 0	HTDAP	0x0000 0000	ro	主机发送描述符地址指针（Host Transmit Descriptor Address Pointer） 这些位在复位时被清零，DMA 在运行过程中对指针进行更新。

27.3.31 以太网 DMA 当前接收描述符寄存器(EMAC_DMARD)

当前主机接收描述符寄存器指向 DMA 正在读取的接收描述符的起始地址。

域	简称	复位值	类型	功能
位 31: 0	HRDAP	0x0000 0000	ro	主机接收描述符地址指针 (Host Receive Descriptor Address Pointer) 这些位在复位时被清除, DMA 在运行过程中对指针进行更新。

27.3.32 以太网 DMA 当前发送缓存地址寄存器(EMAC_DMACTBADDR)

当前主机发送缓存地址寄存器指向 DMA 正在读取的发送缓存地址。

域	简称	复位值	类型	功能
位 31: 0	HTBAP	0x0000 0000	ro	主机发送缓存地址指针 (Host Transmit Buffer Address Pointer) 这些位在复位时被清除, DMA 在运行过程中对指针进行更新。

27.3.33 以太网 DMA 当前接收缓存地址寄存器(EMAC_DMCRBADDR)

当前主机接收缓存地址寄存器指向 DMA 正在读取的接收缓存地址。

域	简称	复位值	类型	功能
位 31: 0	HRBAP	0x0000 0000	ro	主机接收缓存地址指针 (Host Receive Buffer Address Pointer) 这些位在复位时被清除, DMA 在运行过程中对指针进行更新。

27.3.34 以太网 MMC 控制寄存器(EMAC_MMCTRL)

以太网 MMC 控制寄存器(EMAC_MMCTRL)定义了各管理计数器的工作模式。

域	简称	复位值	类型	功能
位 31: 4	保留	0x00000000	resd	请保持默认值。
位 3	FMC	0x0	rw	冻结 MMC 计数器 (Freeze MMC Counter) 置起后, 该位会冻结所有 MMC 计数器使它们保持当前值。除非将该位置 0, 否则, 不会因为接收帧或发送帧而更新 MMC 计数器。如果在读取 MMC 计数器时将 Read 位设置成了 Reset, 那么计数器也同样被清除。
位 2	RR	0x0	rw	读操作时复位 (Reset on Read) 该位置起后, 在读(复位后自动清除) MMC 计数器之后, 该计数器被重置为 0. 读取最低字节位(Bits[7: 0])将清除计数器。
位 1	SCR	0x0	rw	计数器停止滚动 (Stop Counter Rollover) 该位置起后, 计数器在计数到最大值后不会滚动为 0
位 0	RC	0x0	rw	计数器复位 (Reset Counter) 该位置起后, 所有计数器被复位。在 1 个时钟周期后, 该位自动清除。

27.3.35 以太网 MMC 接收中断寄存器(EMAC_MMCR1)

以太网 MMC 接收中断寄存器(EMAC_MMCR1)包含着以下情况下所产生的中断。

- 接收统计计数器计数到其最大值的一半时 (32 位计数器对应的是 0x8000_0000, 16 位计数器对应的是 0x8000)
- 接收统计计数器的计数超过其最大值 (32 位计数器对应的是 0xFFFF_FFFF, 16 位计数器对应的是 0xFFFF)

当计数器停止滚动时, 中断被置起, 但是计数器仍为全 1. 以太网 MMC 接收中断寄存器(EMAC_MMCR1)

是一个 32 位寄存器。读取产生中断的 MMC 计数器时会清除中断位。必须读取相应的计数器的低字节 (Bits[7: 0]) 才能清除中断位。

域	简称	复位值	类型	功能
位 31: 18	保留	0x0000	resd	请保持默认值。
位 17	RGUF	0x0	rrc	接收到好的单播帧 (Received Good Unicast Frames) 接收好的单播帧计数器计数达到最大值或者最大值的一半时，该位被置起。
位 16: 7	保留	0x000	resd	请保持默认值。
位 6	RFAE	0x0	rrc	接收帧对齐错误 (Received Frames Alignment Error) 对齐错误接收帧计数器计数到最大值的一半或最大值时，该位被置起。
位 5	RFCE	0x0	rrc	接收帧 CRC 错误 (Received Frames CRC Error) 接收 CRC 错误帧计数器计数到最大值的一半或最大值时，该位被置起。
位 4: 0	保留	0x00	resd	请保持默认值。

27.3.36 以太网 MMC 发送中断寄存器(EMAC_MMCTI)

以太网 MMC 发送中断寄存器(EMAC_MMCTI)包含着以下情况所产生的中断：发送统计计数器计数达到其最大值的一半 (32 位计数器对应的是 0x8000_0000, 16 位计数器对应的是 0x8000)，以及当发送计数器计数超过其最大值时 (32 位计数器对应的是 0xFFFF_FFFF, 16 位计数器对应的是 0xFFFF)。当计数器停止滚动时，中断被置起，但计数器仍为全 1. 以太网 MMC 发送中断寄存器(EMAC_MMCTI)是一个 32 位寄存器。当读取产生中断的 MMC 计数器时，中断位被清除。必须读取相应的计数器的低字节位 (位[7: 0]) 才能清除中断位。

域	简称	复位值	类型	功能
位 31: 22	保留	0x000	resd	请保持默认值。
位 21	TGF	0x0	rrc	发送好的帧 (Transmitted Good Frames) 发送好帧计数器计数达到最大值的一半或最大值时，该位被置起。
位 20: 16	保留	0x00	resd	请保持默认值。
位 15	TGFMSC	0x0	rrc	发送好的帧时遇到不止 1 个冲突 (Transmitted Good Frames More Single Collision) 发送时 1 次以上冲突后好帧计数器计数达到最大值的一半或最大值时，该位被置起。
位 14	TSCGFCI	0x0	rrc	发送单次冲突好帧计数器中断 (Transmit Single Collision Good Frame Counter Interrupt) 发送单次冲突好帧计数器计数达到最大值的一半或最大值时，该位被置起。
位 13: 0	保留	0x0000	resd	请保持默认值。

27.3.37 以太网 MMC 接收中断屏蔽寄存器(EMAC_MMCRIM)

以太网 MMC 接收中断屏蔽寄存器(EMAC_MMCRIM)包含着：当接收统计计数器计数达到其最大值的一半或最大值时所产生的中断的屏蔽。该寄存器是一个 32 位寄存器。

域	简称	复位值	类型	功能
位 31: 18	保留	0x0000	resd	请保持默认值。
位 17	RUGFCIM	0x0	rw	接收的好单播帧计数器中断屏蔽 (Received Unicast Good Frame Counter Interrupt Mask) 接收的好的单播帧计数器计数达到最大值的一半或最大值时，置起该位将屏蔽中断。
位 16: 7	保留	0x000	resd	请保持默认值。
位 6	RAEFACIM	0x0	rw	接收的对齐错误帧对齐计数器中断屏蔽 (Received Alignment Error Frame Alignment Counter Interrupt Mask) 接收的对齐错误帧计数器计数达到最大值的一半或最大值时，置起该位将屏蔽中断。
位 5	RCEFCIM	0x0	rw	接收的 CRC 错误帧计数器中断屏蔽 (Received CRC Error Frame Counter Interrupt Mask)

位 4: 0	保留	0x00	resd	接收的 CRC 错误帧计数器计数达到最大值的一半或最大值时，置起该位将屏蔽中断。 请保持默认值。
--------	----	------	------	---

27.3.38 以太网 MMC 发送中断屏蔽寄存器(EMAC_MMCTIM)

以太网 MMC 发送中断屏蔽寄存器(EMAC_MMCTIM)包含着：当发送统计计数器计数达到其最大值的一半或最大值时所产生的中断的屏蔽。该寄存器是一个 32 位寄存器。

域	简称	复位值	类型	功能
位 31: 22	保留	0x000	resd	请保持默认值。
位 21	TGFCIM	0x0	rw	发送好帧计数器中断屏蔽 (Transmit Good Frame Counter Interrupt Mask) 发送好帧计数器计数达到最大值的一半或最大值时，置起该位将屏蔽中断。
位 20: 16	保留	0x00	resd	请保持默认值。
位 15	TMCGFCIM	0x0	rw	发送多冲突好帧计数器中断屏蔽 (Transmit Multiple Collision Good Frame Counter Interrupt Mask) 发送多冲突好帧计数器计数达到最大值的一半或最大值时，置起该位将屏蔽中断。
位 14	TSCGFCIM	0x0	rw	发送单冲突好帧计数器中断屏蔽 (Transmit Single Collision Good Frame Counter Interrupt Mask) 发送单冲突好帧计数器计数达到最大值的一半或最大值时，置起该位将屏蔽中断。
位 13: 0	保留	0x0000	resd	请保持默认值。

27.3.39 以太网 MMC 1 次冲突后发送“好”帧的计数器寄存器(EMAC_MMCTFSCC)

该寄存器统计在半双工模式下，发送帧成功时只遇到一次冲突的帧的数目。

域	简称	复位值	类型	功能
位 31: 0	TGFSCC	0x0000 0000	ro	发送好帧单冲突计数器 (Transmitted Good Frames Single Collision Counter) 发送好帧计数器，帧在发送时遭遇一次冲突

27.3.40 以太网 MMC 1 次以上冲突后发送“好”帧的计数器寄存器(EMAC_MMCTFSCC)

该寄存器统计在半双工模式下，发送帧成功时遇到一次以上冲突的帧的数目。

域	简称	复位值	类型	功能
位 31: 0	TGFMSCC	0x0000 0000	ro	发送好帧多冲突计数器 (Transmitted Good Frame More Single Collision Counter) 发送好帧计数器，帧在发送时遭遇多次冲突

27.3.41 以太网 MMC 发送“好”帧计数器寄存器(EMAC_MMCTFCNT)

该寄存器统计发送“好”帧的数目。

域	简称	复位值	类型	功能
位 31: 0	TGFC	0x0000 0000	ro	发送好帧计数器 (Transmitted Good Frames Counter) 发送的好帧计数器。

27.3.42 以太网 MMC CRC 错误接收帧计数器寄存器(EMAC_MMCRFCECNT)

该寄存器统计接收到有 CRC 错误帧的数目。

域	简称	复位值	类型	功能
位 31: 0	RFCEC	0x0000 0000	ro	接收帧 CRC 错误计数器 (Received Frames CRC Error Counter)

接收到有 CRC 错误的帧计数器

27.3.43 以太网 MMC 对齐错误接收帧计数器寄存器(EMAC_MMCRFAECNT)

该寄存器统计接收到有对齐错误帧的数目。

域	简称	复位值	类型	功能
位 31: 0	RFAEC	0x0000 0000	ro	接收到帧对齐错误计数器 (Received Frames Alignment Error Counter) 接收到有对齐错误的帧计数器。

27.3.44 以太网 MMC 接收帧“好”单播帧计数器寄存器(EMAC_MMCRGUFCNT)

该寄存器统计接收到“好”单播帧的数目。

域	简称	复位值	类型	功能
位 31: 0	RGUFC	0x0000 0000	ro	接收到好的单播帧计数器 (Received Good Unicast Frames Counter) 接收到好的单播帧的帧计数器。

27.3.45 以太网 PTP 时间戳控制寄存器(EMAC_PTPTSCTRL)

该寄存器控制着接收器内系统时间的生成以及为 PTP 数据包加盖时间戳。

域	简称	复位值	类型	功能
位 31: 19	保留	0x0000	resd	请保持默认值。
位 18	EMAFPFF	0x0	rw	使能 MAC 地址来过滤 PTP 帧 (Enable MAC Address For PTP Frame Filtering) 该位被置起后，当通过以太网直接发送 PTP 时，目的 MAC 地址（与任意一个 MAC 地址寄存器匹配）将用于过滤 PTP 帧。
位 17: 16	SPPFTS	0x0	rw	选择需要拍摄快照的 PTP 数据包 (Select PTP Packets For Taking Snapshot) 00: 普通时钟 01: 边界时钟 10: 端对端透明时钟 11: 点对点透明时钟
位 15	ESFMRTM	0x0	rw	使能与主节点相关的消息的快照拍摄 (Enable Snapshot For Message Relevant To Master) 该位被置起后，仅对与主机节点相关的消息进行拍照。反之，对与从节点相关的消息进行拍照。
位 14	ETSFEM	0x0	rw	使能时间消息的时间戳拍照 (Enable Timestamp Snapshot For Event Messages) 该位被置起后，时间戳拍照仅用于事件消息(SYNC, Delay_Req, Pdelay_Req, or Pdelay_Resp)。该位被复位后，时间戳拍照适用于除了 Announce、Management 和 Signaling 以外的所有消息。
位 13	EPPFSIP4U	0x1	rw	使能处理经由 IPv4-UDP 发送的 PTP 帧 (Enable Processing of PTP Frames Sent over IPv4-UDP) 该位被置起后，MAC 接收器将处理通过 IPv4 数据包封装在 UDP 内的 PTP 数据包。该位被清除后，MAC 将忽略经由 UDP-IPv4 数据包传输的 PTP。该位是默认置起。
位 12	EPPFSIP6U	0x0	rw	使能处理经由 IPv6-UDP 发送的 PTP 帧 (Enable Processing of PTP Frames Sent over IPv6-UDP) 该位被置起后，MAC 接收器将处理经由 IPv6 数据包封装在 UDP 内的 PTP 数据包。该位被清除后，MAC 将忽略经由 UDP-IPv6 数据包发送的 PTP。
位 11	EPPEF	0x0	rw	使能处理 EMAC 帧上的 PTP (Enable Processing of PTP over EMAC Frames)

				该位被置起后，MAC 接收器将处理直接封装在以太网帧内的 PTP 数据包。该位被清除后，MAC 会忽略以太网上的 PTP。
位 10	EPPV2F	0x0	rw	使用 V2 格式处理 PTP 数据包 (Enable PTP packet Processing for Version 2 Format) 该位被置起后，使用 1588 V2 格式来处理 PTP 数据包。反之，使用 V1 格式处理 PTP 数据包。关于 IEEE 1588 V1 和 V2 格式，请参考第 155 页的“PTP 处理和控制”章节。
位 9	TDBRC	0x0	rw	时间戳数字滚动或二进制滚动控制 (Timestamp Digital or Binary Rollover Control) 该位被置起后，时间戳低寄存器将在 0x3B9A_C9FF 值（即 1 纳秒精度）之后滚动，并递增时间戳（高）秒。该位被复位后，亚秒寄存器的滚动值为 0x7FFF_FFFF。必须根据 PTP 参考时钟频率和该位的值来正确设置亚秒递增量。
位 8	ETAF	0x0	rw	使能所有帧的时间戳功能 (Enable Timestamp for All Frames) 该位被置起后，MAC 接收到的所有帧的时间戳快照功能被启用。
位 7: 6	保留	0x0	resd	请保持默认值。
位 5	ARU	0x0	rw	更新加数寄存器 (Addend Reg Update) 该位被置起后，以太网 PTP 时间戳加数寄存器 (EMAC_PTPTSAD) 的内容会更新在 PTP 模块进行精细修正。完成内容更新后，清除此位。该寄存器位在被置起之前应该为 0.
位 4	TITE	0x0	rw	使能时间戳中断触发 (Timestamp Interrupt Trigger Enable) 该位被置起后，当系统时间大于目标时间寄存器中写入的值时，产生时间戳中断。产生了时间戳触发中断后，该位被清除。
位 3	TU	0x0	rw	时间戳更新 (Timestamp Update) 该位被置起后，系统时间将根据系统时间-秒更新寄存器和系统时间-纳秒更新寄存器的设定值进行更新（即加或减）。 在被更新之前，该位应该为 0. 硬件完成更新后，该位被复位。“时间戳高字”寄存器（如果已使能）不会被更新。
位 2	TI	0x0	rw	时间戳初始化 (Timestamp Initialize) 该位被置起后，将根据系统时间-秒更新寄存器（和寄存器系统时间-纳秒更新寄存器）的设定值来初始化（即覆盖）系统时间。 在被更新之前，该位应该为 0. 完成初始化后，该位被复位。“时间戳高字”寄存器（如果已使能）才能被初始化。
位 1	TFCU	0x0	rw	时间戳精细更新或粗略更新 (Timestamp Fine or Coarse Update) 该位被置起后，该位表示使用精细更新方法来更新系统时间戳。被复位后，表示使用粗略的更新方法来更新系统时间戳。
位 0	TE	0x0	rw	使能时间戳 (Timestamp Enable) 该位被置起后，将会发送帧和接收帧添加时间戳。被关闭后，不再为发送帧和接收帧添加时间戳，时间戳生成器也会被挂起。被使能后，需要初始化时间戳（系统时间）。在接收端，只有当该位被置起后，MAC 才会处理 1588 帧。

时间戳快照对寄存器位的相依性

SPPFTS 位 17: 16	ESFMRTM 位 15	ETSFEM 位 14	PTP 消息
00 或 01	X	0	SYNC、Follow_Up、Delay_Req、Delay_Resp
00 或 01	1	1	Delay_Req

00 或 01	0	1	SYNC
10	N/A	0	SYNC、Follow_Up、Delay_Req、Delay_Resp
10	N/A	1	SYNC、Follow_Up
11	N/A	0	SYNC、Follow_Up、Delay_Req、Delay_Resp、Pdelay_Req、Pdelay_Resp
11	N/A	1	SYNC、Pdelay_Req、Pdelay_Resp

1 : N/A=不适用

2 : X=无关

27.3.46 以太网 PTP 亚秒递增寄存器(EMAC_PTPSSINC)

只有在没有外部时间戳输入的情况下选择了 IEEE1588 时间戳功能，该寄存器才会存在。在粗略更新模式下（寄存器中的 TSCFUPDT 位），每个 clk_ptp_ref_i 时间周期都会将该寄存器的值添加到系统时间。在精细更新模式下，每当累加器产生溢出时，就会向系统时间添加该寄存器的值。

域	简称	复位值	类型	功能
位 31: 8	保留	0x000000	resd	请保持默认值。
位 7: 0	SSIV	0x00	rw	亚秒递增值 (Sub-Second Increment Value) 该位域的设定值会在每个时钟周期(of clk_ptp_i)随着亚秒寄存器的值累加。例如，如果 PTP 时钟是 50 MHz (周期 20 ns)，当系统时间纳秒寄存器精度为 1 ns[通过在寄存器 448 即以太网 PTP 时间戳控制寄存器 (EMAC_PTPTSCTRL) 中设置位 9(TSCTRLSSR)]时，则需要将这些位的值设为 20 (0x14)。当 TSCTRLSSR 被清除后，纳秒寄存器分辨精度为~0.465ns。此时，需要将这些位的值设为 43 (0x2B)，即 20ns/0.465。

27.3.47 以太网 PTP 时间戳高寄存器(EMAC_PTPTSH)

系统时间-秒寄存器以及系统时间-纳秒寄存器表示的是由 MAC 维护的系统时间的当前值。该值会持续更新。

域	简称	复位值	类型	功能
位 31: 0	TS	0x0000 0000	ro	时间戳秒 (Timestamp Second) 这些位域的值表示的是由 MAC 维护的当前系统时间的秒值。

27.3.48 以太网 PTP 时间戳低寄存器(EMAC_PTPTSL)

该寄存器包含时间信息的低 32 位。该寄存器为只读，包含了系统时间的亚秒值。

域	简称	复位值	类型	功能
位 31	AST	0x0	ro	时间的加或减 (Add or Subtract Time) 该位被置起后，时间值将与更新寄存器的值相减。被复位后，时间值将与更新寄存器的值相加。
位 30: 0	TSS	0x0000 0000	ro	时间戳亚秒 (Timestamp Sub Seconds) 这个位域的值表示的是亚秒时间，精度为 0.46 ns。将寄存器 448 (以太网 PTP 时间戳控制寄存器 (EMAC_PTPTSCTRL)) 的位 9(TSCTRLSSR) 置起后，每个位代表 1ns，所编程的值不能超过 0x3B9A_C9FF。

27.3.49 以太网 PTP 时间戳高更新寄存器(EMAC_PTPTSHUD)

系统时间-秒更新寄存器和系统时间-纳秒更新寄存器将初始化或更新由 MAC 维护的系统时间。在设置以太网 PTP 时间戳控制寄存器(EMAC_PTPTSCTRL)内的 TSINIT 或 TSUPDT 位之前，必须先向这两个寄存器进行写操作。

域	简称	复位值	类型	功能
位 31: 0	TS	0x0000 0000	rw	时间戳秒 (Timestamp Second) 这个位域表示的是将被初始化或者将被添加到系统时间的秒时间值。

27.3.50 以太网 PTP 时间戳低更新寄存器(EMAC_PTPTSLUD)

只有在没有外部时间戳输入的情况下选择了 IEEE1588 时间戳功能，该寄存器才会存在。

域	简称	复位值	类型	功能
位 31	AST	0x0	rw	时间的加与减 (Add or Subtract Time) 该位被置起后，时间值将与更新寄存器的值相减。被复位后，时间值将与更新寄存器的值相加。
位 30: 0	TSS	0x0000 0000	rw	时间戳亚秒 (Timestamp Sub Seconds) 这个位域的值表示的是亚秒时间，精度为 0.46 ns。将寄存器 448 (以太网 PTP 时间戳控制寄存器 (EMAC_PTPTSCTRL)) 的位 9(TSCTRLSSR) 置起后，每个位代表 1ns，所编程的值不能超过 0x3B9A_C9FF。

27.3.51 以太网 PTP 时间戳加数寄存器(EMAC_PTPTSAD)

该寄存器的值仅用于当系统时间被设置为精细更新模式的情况下。该寄存器的值会在每个时钟周期(of clk_ptp_ref_i)被添加到 32 位累加器，每当累加器溢出时，系统时间就被更新。

域	简称	复位值	类型	功能
位 31: 0	TAR	0x0000 0000	rw	时间戳加数寄存器 (Timestamp Addend Register) 这个位域表示的是为了实现时间同步，将要被添加到累加器的 32 位时间值。

27.3.52 以太网 PTP 目标时间高寄存器(EMAC_PTPTTH)

目标时间秒寄存器和目标时间亚秒寄存器用于调度当系统时间超过寄存器的设定值时的中断事件。

域	简称	复位值	类型	功能
位 31: 0	TTSR	0x0000 0000	rw	目标时间秒寄存器 (Target Time Seconds Register) 该寄存器存放着秒时间值。当时间戳值等于或超过这两个目标时间戳寄存器的值时，MAC 会根据 PPS 控制寄存器的位[6: 5]，开始或中断 PPS 信号输出，并产生中断（如果已使能）

27.3.53 以太网 PTP 目标时间低寄存器(EMAC_PTPTTL)

域	简称	复位值	类型	功能
位 31: 0	TTLR	0x0000 0000	rw	目标时间戳低寄存器 (Target Timestamp Low Register) 该寄存器存放着纳秒时间值(有符号的)。当时间戳的值等于这两个目标时间戳寄存器的值时，MAC 会根据 PPS 控制寄存器的 TRGTMODSEL0(位[6: 5])，开始或中断 PPS 信号输出，并产生中断（如果已使能） 当以太网 PTP 时间戳控制寄存器(EMAC_PTPTSCTRL) 的位 9(TSCTRLSSR) 被置起时，该位域的值不能超过 0x3B9A_C9FF PPS 信号输出的实际开始时间或中断时间可能会有高达 1 个亚秒增量值的误差。

27.3.54 以太网 PTP 时间戳状态寄存器 (EMAC_PTPTSSR)

域	简称	复位值	类型	功能
位 31: 2	保留	0x0000 0000	resd	请保持默认值。
位 1	TTTR	0x0	ro	达到时间戳目标时间 (Timestamp Target Time Reached) 被置起后，该位表示系统时间大于或等于目标时间秒寄存器和目标时间纳秒寄存器的设定值
位 0	TSO	0x0	ro	时间戳秒溢出 (Timestamp Seconds Overflow) 被置起后，该位表示时间戳秒值（支持 V2 格式）已经溢出并超过了 32'hFFFF_FFFF.

27.3.55 以太网 PTP PPS 控制寄存器 (EMAC_PTPPPSCR)

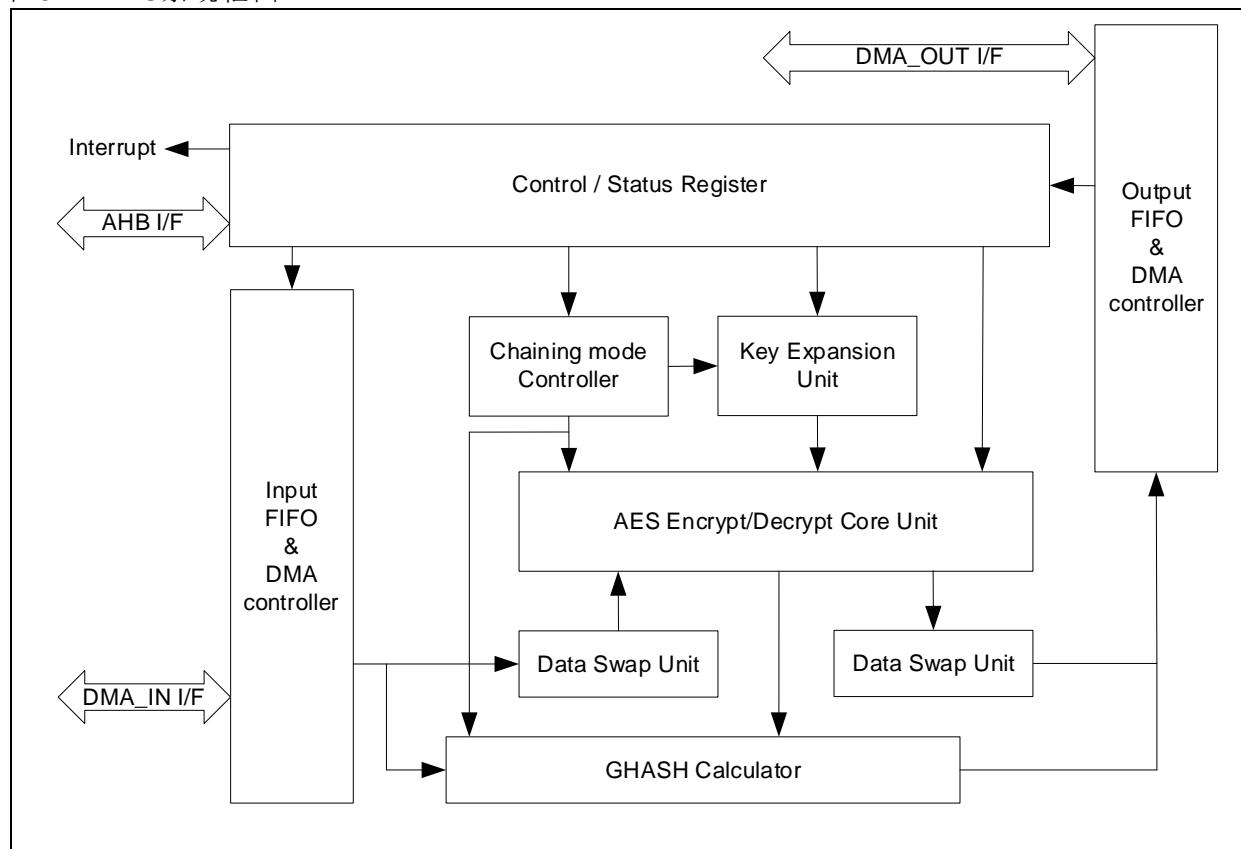
域	简称	复位值	类型	功能
位 31: 4	保留	0x0000000	resd	请保持默认值。 PPS 频率选择 (PPS0 Output Frequency Control) 此位输出受 emac_pps_sel 位控制 (额外寄存器 2 (CRM_MISC2) 位 15) Emac_pps_sel=0: 0000: 1 Hz, 使用二进制翻转时, 脉冲宽度为 125 ms, 使用数字翻转时, 脉冲宽度为 100 ms 0001: 2 hz, 使用二进制翻转时, 占空比为 50% (不建 议使用数字翻转) 0010: 4 hz, 使用二进制翻转时, 占空比为 50% (不建 议使用数字翻转) 0011: 8 hz, 使用二进制翻转时, 占空比为 50% (不建 议使用数字翻转) 0100: 16 hz, 使用二进制翻转时, 占空比为 50% (不建 议使用数字翻转) ... 1111: 32.768 khz, 使用二进制翻转时, 占空比为 50% (不建议使用数字翻转) Emac_pps_sel=1: 0000: 1 Hz, 脉冲宽度为一个 clk_ptp 周期 0001: 二进制翻转是 2hz, 占空比 50%, 数字翻转是 1hz (不建议使用数字翻转) 0010: 二进制翻转是 4hz, 占空比 50%, 数字翻转是 2hz (不建议使用数字翻转) 0011: 二进制翻转是 8hz, 占空比 50%, 数字翻转是 4hz (不建议使用数字翻转) ... 1111: 二进制翻转是 32.768khz, 占空比 50%, 数字翻 转是 16.384khz (不建议使用数字翻转)
位 3: 0	POFC	0x0	rw	在 PPS 为非 0 值时, 不建议使用数字翻转, 因为这些情 况下 PPS 输出的波形会不规则 (尽管其平均频率在任何 一秒窗口内始终正确)

28 AES 硬件加密单元 (AES)

28.1 简介

AES 硬件加密单元(AES)是使用 NIST FIPS PUB-197 Advanced Encryption Standard (AES)之公开算法，对数据进行加解密之加速组件。AES 是对称式的分组密法算法，以 128 比特位为一个数据分组。使用 128, 192, 与 256 位长度之密钥，进行加解密运算。可支持多种常用的工作模式。可配合数据存储格式，进行输入/输出数据之半字，字节，与位交换。直接存储器访问的使用，可在不耗用 CPU 资源的状态下，与存储单元进行数据传输。使用者仅须透过中断监控加密单元运作状态即可。

图 28-1 AES 系统框图



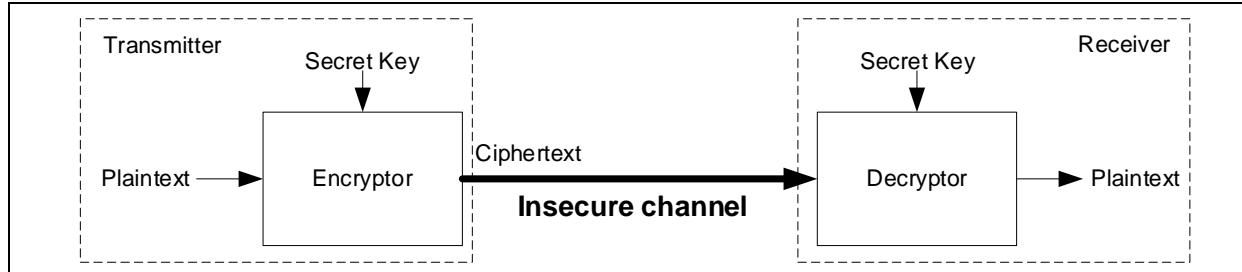
28.2 主要特点

- 符合 NIST FIPS PUB-197 Advanced Encryption Standard (AES) 之设计
- 支持 128, 192, 与 256 位之密钥长度
- 支持多种常用的工作模式：
 - 电子密码本模式 (ECB)
 - 密码块链接模式 (CBC)
 - 计数器模式 (CTR)
 - 伽罗瓦/计数器模式 (GCM)
 - 计数器模式搭配密码块链接消息验证代码 (CCM)
- 支持以伽罗瓦/计数器模式 (GCM) 实现伽罗瓦消息验证代码 (GMAC) 模式
- 支持 4 个字的初始化向量 (IV)，可使用于 CBC, CTR, GCM, CCM
- 可支持以两路直接存储器访问进行数据传输
- 以字为单位，支持输入/输出数据之半字，字节，与位交换
- 以分组为单位，支持由软件打断加密/解密运作，以及恢复先前运作之能力

28.3 密码算法操作

在通讯上，传送方欲于不安全通道上与接收方分享信息，可取用两方已协议共享之私密钥匙，以对称式密码算法将信息明文加密成密文传送。接收方收到后，可使用该私密钥匙将密文解密成原信息明文。而存在于不安全通道上的潜在窃听者，由于窃听者不拥有私密钥匙，故无法取得信息之明文，如图所示意：

图 28-2 信息分享加解密示意图



28.3.1 加密

配置 AES_CTRL 的寄存器 OPR 为 0，可启用加密模式。此模式下，用户通过访问 AES 输入数据寄存器(AES_IDT)输入信息明文进行加密。加密完成之密文存放于输出缓冲中，通过访问 AES 输出数据寄存器(AES_ODT)，用户可取得信息密文。AES 密钥寄存器(AES_KEYx)、AES 初始向量寄存器(AES_IVx)与 AES 控制寄存器(AES_CTRL)需要在 AES 使能前配置完成。若串接模式为伽罗瓦/计数器模式或计数器模式搭配密码块链接消息验证代码(CHN=3 或 CHN=4)，亦须配置清空 AES 停滞讯息寄存器(AES_SIx)。详细操作方式请见 28.4 工作串接模式中各模式之说明。

28.3.2 解密

AES 以迭代方式进行运算，解密所使用的轮密钥为加密轮密钥之反序排列。故欲将密文解密回信息明文，AES 硬件加密单元(AES)需先将私密钥匙扩展成轮密钥后在解密，与加密时的实时扩展有所不同。依据信息的分组数量，AES 硬件加密单元可以两种模式进行解密：

单一分组信息解密

在串接模式为电子密码本模式或密码块链接模式(CHN=0 或 CHN=1)下，欲解密之密文仅只一个分组时，可直接配置 AES_CTRL 的寄存器 OPR 为 3，启用密钥扩展并单分组解密。此模式下，用户访问 AES 输入数据寄存器(AES_IDT)四次，依序输入一个分组的密文后进行解密。解密完成之明文存放于输出缓冲中，用户通过连续访问 AES 输出数据寄存器(AES_ODT)四次，可取得原信息明文。AES 密钥寄存器(AES_KEYx)、AES 初始向量寄存器(AES_IVx)与 AES 控制寄存器(AES_CTRL)需要在 AES 使能前配置完成。

串接分组信息解密

在串接模式为电子密码本模式或密码块链接模式(CHN=0 或 CHN=1)下，欲解密多于一个分组之密文时，需先进行密钥扩展，方可切换模式进行解密。AES 密钥寄存器(AES_KEYx)配置完成后，配置 AES_CTRL 的寄存器 OPR 为 1 并置位 AES 使能(AESEN)，即可启动密钥扩展。密钥扩展完成后，切勿更动 AES 密钥寄存器(AES_KEYx)。接续改配置 OPR 为 2 可启用解密模式。此模式下，用户通过访问 AES 输入数据寄存器(AES_IDT)输入密文进行解密。解密完成之信息明文存放于输出缓冲中，用户可通过访问 AES 输出数据寄存器(AES_ODT)取得。AES 初始向量寄存器(AES_IVx)与 AES 控制寄存器(AES_CTRL)需要在 AES 使能前配置完成。

在串接模式为计数器模式、伽罗瓦/计数器模式或计数器模式搭配密码块链接消息验证代码(CHN=2 或 CHN=3 或 CHN=4)下，欲解密多于一个分组之密文时，无须先进行密钥扩展。直接配置 AES_CTRL 的寄存器 OPR 为 2 启用解密模式。此模式下，用户通过访问 AES 输入数据寄存器(AES_IDT)输入密文进行解密。解密完成之信息明文存放于输出缓冲中，用户可通过访问 AES 输出数据寄存器(AES_ODT)取得。AES 密钥寄存器(AES_KEYx)、AES 初始向量寄存器(AES_IVx)与 AES 控制寄存器(AES_CTRL)需要在 AES 使能前配置完成。若串接模式为伽罗瓦/计数器模式或计数器模式搭配密码块链接消息验证代码(CHN=3 或 CHN=4)，亦须配置清空 AES 停滞讯息寄存器(AES_SIx)。

28.4 工作串接模式

对于信息长度超过一个数据分组（即 16 字节）的明文或密文，AES 支持多种常用的工作串接模式，使用同一组密钥对其进行加密或解密。常用的工作模式介绍如下：

28.4.1 电子密码本模式 (ECB)

电子密码本模式(ECB)是最基本的串接模式，配置 AES_CTRL 的寄存器 CHN 为 0，可选用此模式进行加解密。此模式下，信息以 16 字节拆分分组，各分组独立进行加密或解密，如下图所示：

图 28-3 电子密码本模式加密

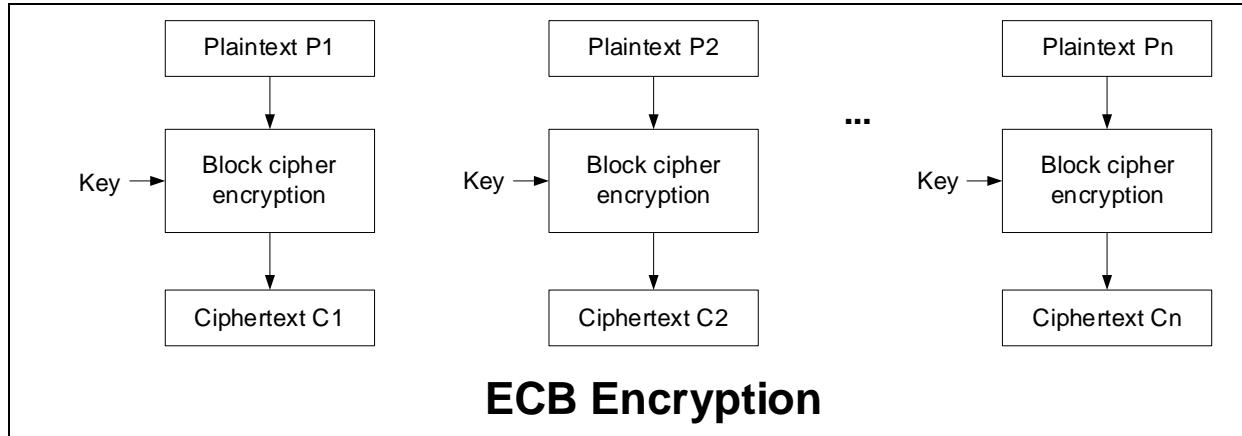
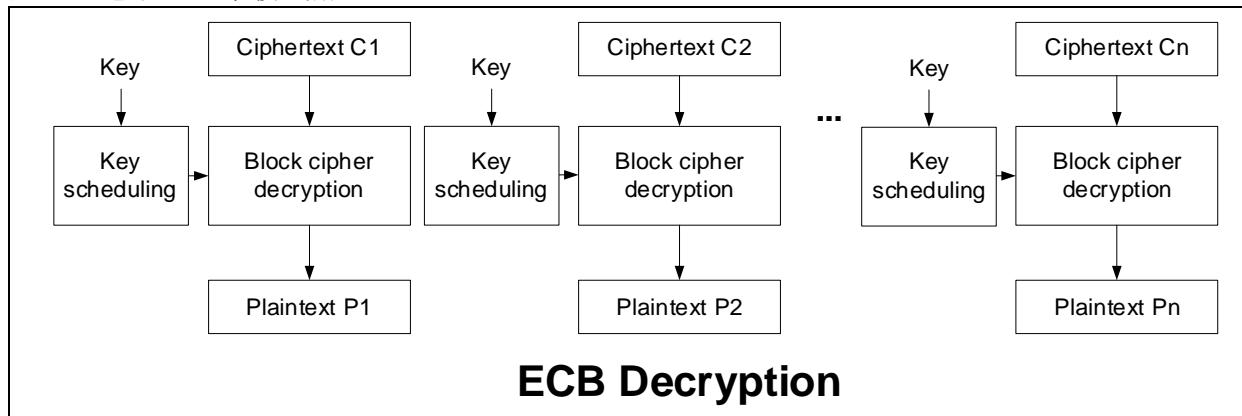


图 28-4 电子密码本模式解密



加解密配置流程

在此模式下，加解密以三个阶段进行：

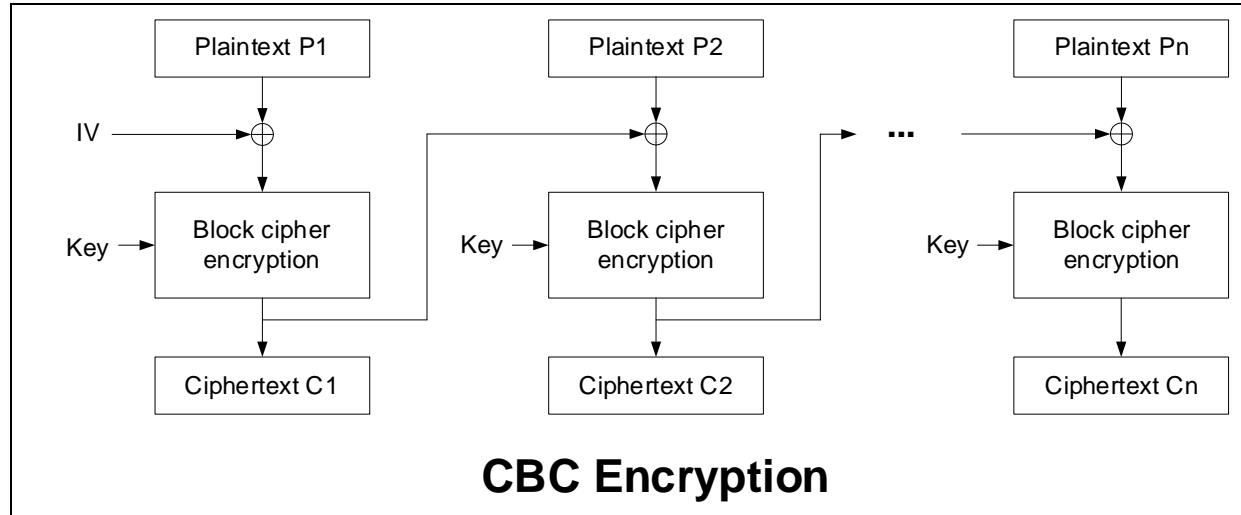
1. 准备阶段
 - 配置 AES_CTRL 的寄存器 CHN 为 0。
 - 配置 AES 密钥寄存器 (AES_KEYx)。
2. 解密密钥扩展
 - 若为加密操作，跳过此阶段。
 - 配置 AES_CTRL 的寄存器 OPR 为 1。
 - 配置 AES_CTRL 的寄存器 AESEN 使能 AES 后，在密钥扩展程序完成后 AESEN 会自动重置。
 - 等待 AES_STS 的 PDFS 置位，置起后对 AES_FCLR 的 PDFC 位写 1 清除 PDFS。
3. 加解密阶段
 - 根据加密或解密，配置 AES_CTRL 的寄存器 OPR 为 0 或 2。
 - 以 16 字节为分组充填欲加密之明文或欲解密之密文，充填动作可透过 CPU 或 DMA 访

- 问完成，操作方法请参考28.5.2数据充填流程。
- 完成所有数据的充填后，重置AESEN。

28.4.2 密码块链接模式 (CBC)

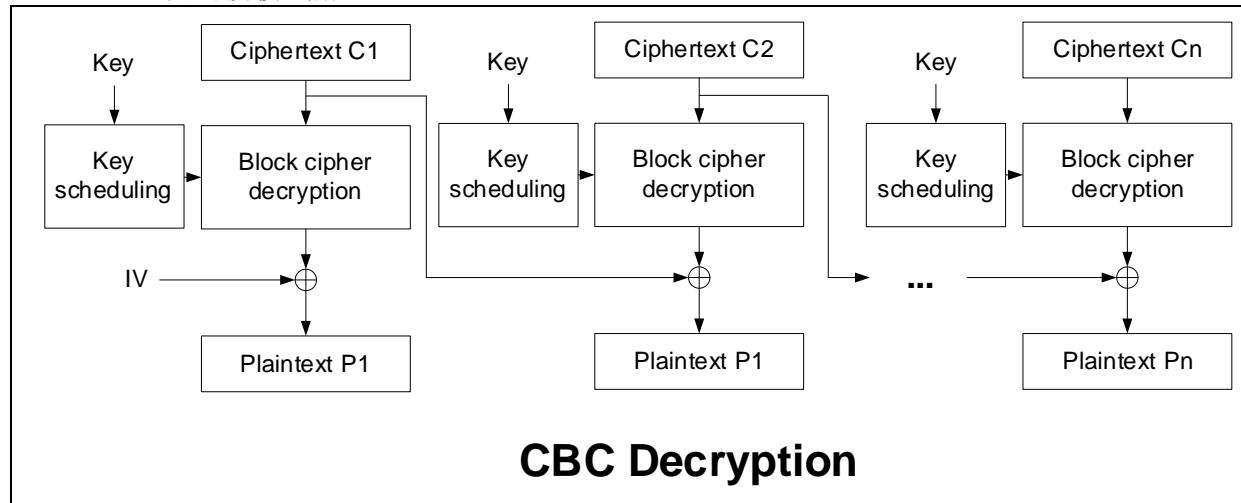
密码块链接模式(CBC)的分组密文为前面所有的分组明文经运算所形成，配置AES_CTRL的寄存器CHN为1，可选用此模式进行加解密。此模式下，信息明文拆分分组后，将前次分组密文与此次分组明文取异或之后，再由加解密核心单元进行加密。为确保信息之唯一性，需在第一个分组上使用初始化向量代替前组密文，初始化向量配置于AES初始向量寄存器(AES_IVx)。加密流程如下图所示：

图 28-5 密码块链接模式加密



此模式解密时，将接收密文拆分分组后，先由加解密核心单元进行解密，在与前次分组密文取异或取得明文。在第一个分组上，需使用与加密相同之初始化向量代替前组密文进行解密。解密流程如下图所示：

图 28-6 密码块链接模式解密



加解密配置流程

在此模式下，加解密以三个阶段进行：

1. 准备阶段

- 配置AES_CTRL的寄存器CHN为1。
- 配置AES密钥寄存器(AES_KEYx)与AES初始向量寄存器(AES_IVx)。

2. 解密密钥扩展

- 若为加密操作，跳过此阶段。

- 配置AES_CTRL的寄存器OPR为1。
- 配置AES_CTRL的寄存器AESEN使能AES后，在密钥扩展程序完成后AESEN会自动重置。
- 等待AES_STS的PDFS置位，置起后对AES_FCLR的PDFC位写1清除PDFS。

3. 加解密阶段

- 根据加密或解密，配置AES_CTRL的寄存器OPR为0或2。
- 以16字节为分组充填欲加密之明文或欲解密之密文，充填动作可透过CPU或DMA访问完成，操作方法请参考28.5.2数据充填流程。
- 完成所有数据的充填后，重置AESEN。

28.4.3 计数器模式 (CTR)

计数器模式以流密码的方式进行加解密，透过加密一次性随机数与逐次累加的计数器值组合产生密钥流分组，将其与信息明文分组取异或即获得密文。由于异或运算之对称性，加密与解密流程完全相同。配置AES_CTRL的寄存器CHN为2可选用此模式，加解密流程如下图所示：

图 28-7 计数器模式加密

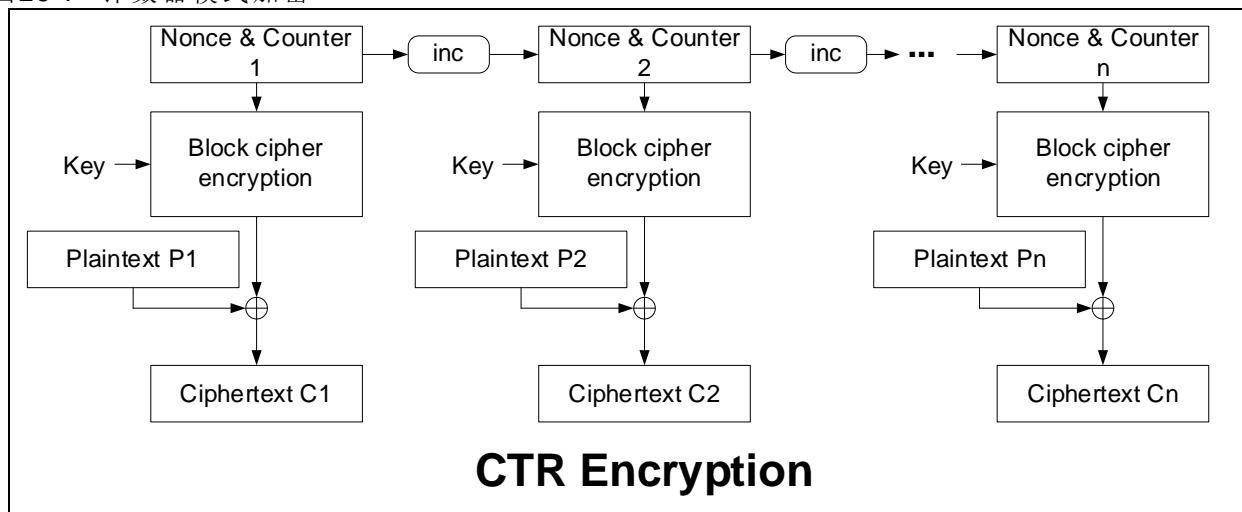
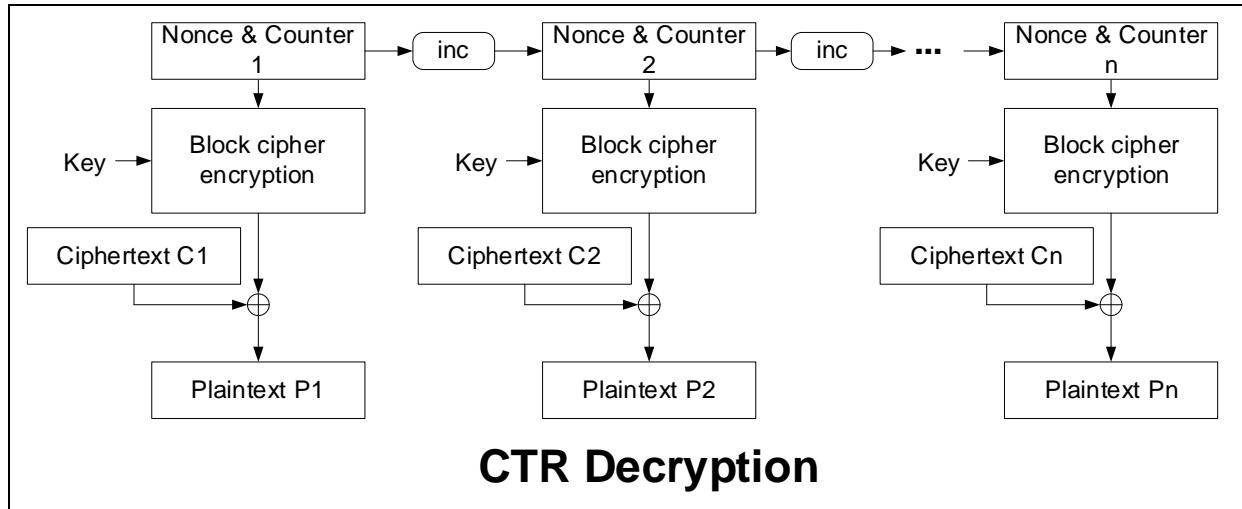


图 28-8 计数器模式解密



加解密配置流程

由于计数器模式是以加密计数器之内容产生密钥流分组，故与电子密码本模式和密码块链接模式不同，计数器模式解密之前，无须进行密钥扩展，加解密以两个阶段进行：

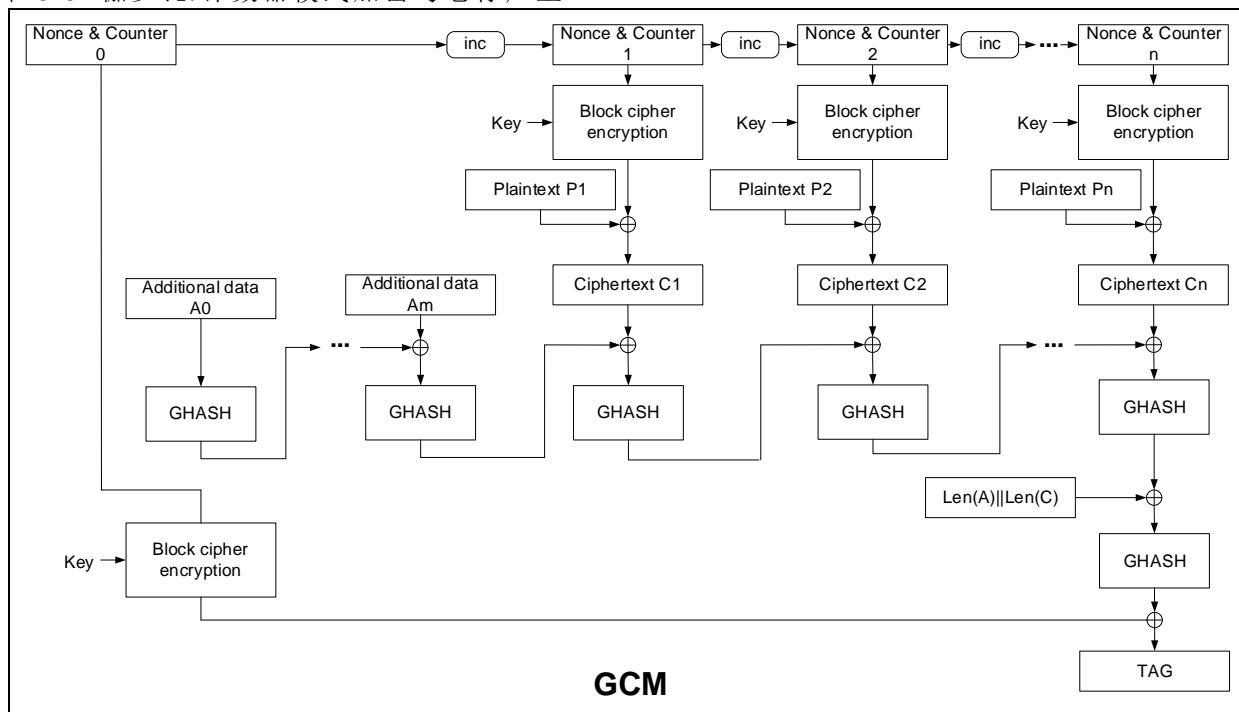
1. 准备阶段

- 配置AES_CTRL的寄存器CHN为2。
 - 配置AES密钥寄存器(AES_KEYx)与AES初始向量寄存器(AES_IVx)。
2. 加解密阶段
- 根据加密或解密，配置AES_CTRL的寄存器OPR为0或2。
 - 以16字节为分组充填欲加密之明文或欲解密之密文，充填动作可透过CPU或DMA访问完成，操作方法请参考28.5.2数据充填流程。
 - 完成所有数据的充填后，重置AESEN。

28.4.4 伽罗瓦/计数器模式 (GCM)

伽罗瓦/计数器模式结合计数器模式与伽罗瓦哈希函数，以同时保证信息的机密性与完整性。信息之加解密以计数器模式完成，如28.4.3之介绍。加密产生的密文与附加身份验证数据以分组为单位，使用伽罗瓦哈希函数的运算产生卷标。配置AES_CTRL的寄存器CHN为3可选用此模式，加密与卷标产生流程如下图所示：

图28-9 伽罗瓦/计数器模式加密与卷标产生



加解密配置流程

在此模式下，加解密与验证以六个阶段进行：

1. 准备阶段
 - 配置AES_CTRL的寄存器CHN为3。
 - 配置AES密钥寄存器(AES_KEYx)与AES初始向量寄存器(AES_IVx)。
 - 配置清空AES停滞讯息寄存器(AES_SIx)。
 - 根据加密或解密，配置AES_CTRL的寄存器OPR为0或2。
2. 伽罗瓦哈希函数初始化
 - 配置AES_CTRL的寄存器PRC为0。
 - 配置AES_CTRL的寄存器AESEN使能AES。
 - 等待AES_STS的PDFS置位，置起后对AES_FCLR的PDFC位写1清除PDFS。
3. 附加身份验证数据阶段
 - 若无附加身份验证数据，跳过此阶段。

- 配置AES_CTRL的寄存器PRC为1。
 - 以16字节为分组充填附加身份验证数据，充填动作可透过CPU或DMA访问完成，操作方法请参考28.5.2数据充填流程，直到完成所有数据的充填。

4. 加解密阶段

- 欲使用伽罗瓦消息验证代码(GMAC), 跳过此阶段。
 - 配置AES_CTRL的寄存器PRC为2。
 - 以16字节为分组充填欲加密之明文或欲解密之密文, 充填动作可透过CPU或DMA访问完成, 操作方法请参考28.5.2数据充填流程, 直到待运算数据不足16字节。

5. 加解密最后分组

- 使用伽罗瓦消息验证代码(GMAC)跳过此阶段。
 - 因待运算数据不足16字节，依据不足之数量，配置AES_CTRL的寄存器NDD，在最后一组数据后面补虚假数据，补足16字节。
 - 配置AES_CTRL的寄存器LST位为1。
 - 输入加解密最后分组，写入AES输入数据寄存器(AES_IDT)四次，硬件会忽略多余的数据(多余的数目写在NDD字段)。
 - 等待AES_STS的PDFS置位，访问AES输出数据寄存器(AES_ODT)四次，取得加解密最后分组的运算结果。
 - 对AES_FCLR的PDFC位写1清除PDFS。

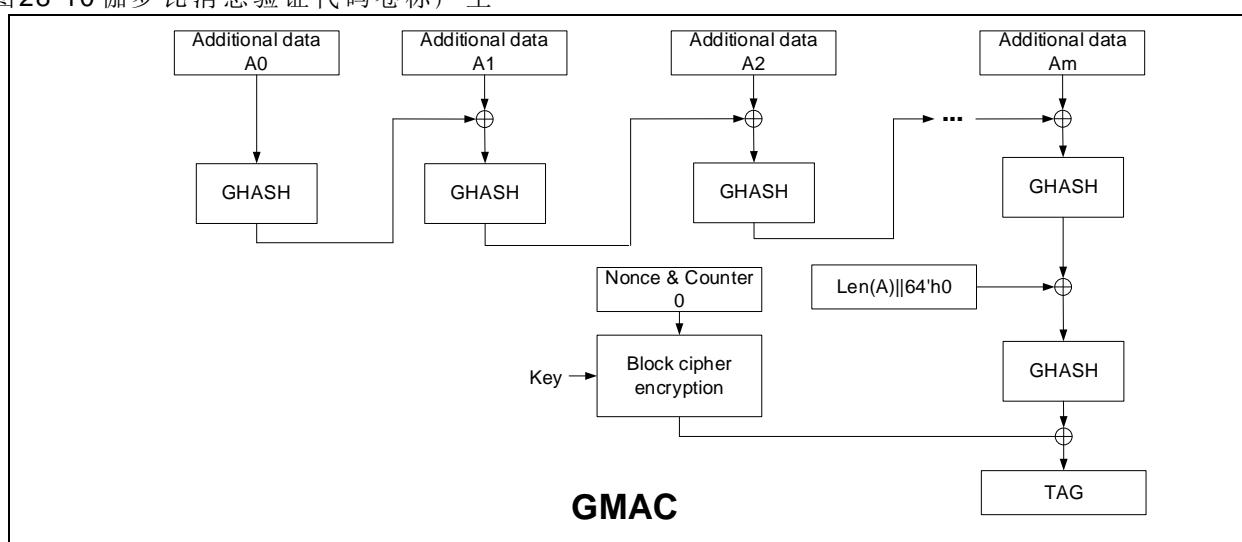
6. 标签阶段

- 配置AES_CTRL的寄存器PRC为3。
 - 访问AES输入数据寄存器(AES_IDT)四次，输入结尾分组。结尾分组由附加身份验证数据长度与密文长度组成，合计16字节。
 - 等待AES_STS的PDFS置位，置起后访问AES输出数据寄存器(AES_ODT)四次，可取得验证标签。
 - 对AES_FCLR的PDFC位写1清除PDFS后，重置AESEN禁止AES硬件加密单元。

28.4.5 伽罗瓦消息验证代码(GMAC)

伽罗瓦消息验证代码(GMAC)是伽罗瓦/计数器模式结合计数器模式的变形体，仅以伽罗瓦哈希函数的运算产生卷标确保信息之完整性，无须进行加密。其卷标产生流程请参考 28.4.4 伽罗瓦/计数器模式(GCM)的加解密配置流程，并跳过阶段 4 的加解密阶段。卷标产生流程如下图所示：

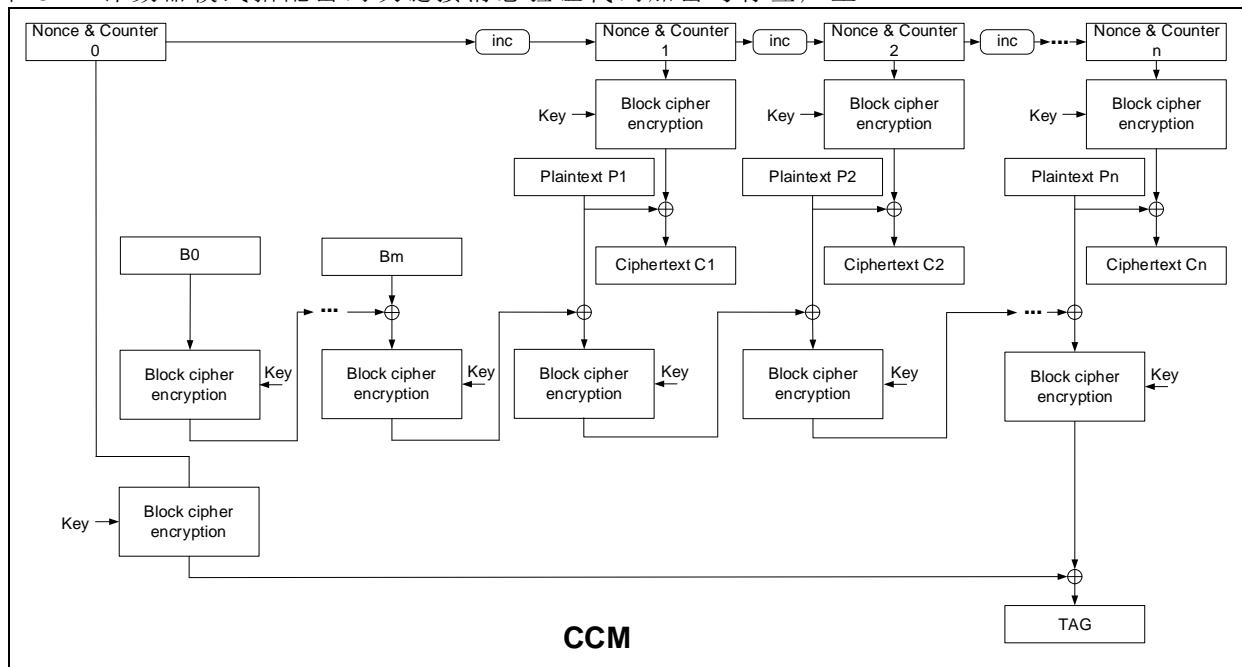
图28-10 伽罗瓦消息验证代码卷标产生



28.4.6 计数器模式搭配密码块链接消息验证代码(CCM)

计数器模式搭配密码块链接消息验证代码结合计数器模式与密码块链接消息验证代码，以同时保证信息的机密性与完整性。信息之加解密以计数器模式完成，如 28.4.3 之介绍。加密产生的密文与附加身份验证数据以分组为单位，使用密码块链接消息验证代码产生标签。配置 AES_CTRL 的寄存器 CHN 为 4 可选用此模式，加密与卷标产生流程如下图所示：

图 28-11 计数器模式搭配密码块链接消息验证代码加密与标签产生



加解密配置流程

在此模式下，加解密与验证以五个阶段进行：

1. 准备阶段

- 配置 AES_CTRL 的寄存器 CHN 为 4。
- 配置 AES 密钥寄存器 (AES_KEYx) 与 AES 初始向量寄存器 (AES_IVx)。
- 配置清空 AES 停滞讯息寄存器 (AES_SIx)。
- 根据加密或解密，配置 AES_CTRL 的寄存器 OPR 为 0 或 2。

2. 附加身份验证数据阶段

- 若无附加身份验证数据，跳过此阶段。
- 配置 AES_CTRL 的寄存器 PRC 为 1。
- 配置 AES_CTRL 的寄存器 AESEN 使能 AES 后，以 16 字节为分组充填附加身份验证数据，充填动作可透过 CPU 或 DMA 访问完成，操作方法请参考 28.5.2 数据充填流程，直到完成所有数据的充填。

3. 加解密阶段

- 配置 AES_CTRL 的寄存器 PRC 为 2。
- 以 16 字节为分组充填欲加密之明文或欲解密之密文，充填动作可透过 CPU 或 DMA 访问完成，操作方法请参考 28.5.2 数据充填流程，直到待运算数据不足 16 字节。

4. 加解密最后分组

- 因待运算数据不足 16 字节，依据不足之数量，配置 AES_CTRL 的寄存器 NDD，在最后一组数据后面补虚假数据，补足 16 字节。
- 配置 AES_CTRL 的寄存器 LST 位为 1。
- 输入加解密最后分组，写入 AES 输入数据寄存器 (AES_IDT) 四次，硬件会忽略多余的数据(多余的数目写在 NDD 字段)。

- 等待AES_STS的PDFS置位，访问AES输出数据寄存器(AES_ODT)四次，取得加解密最后分组的运算结果。
- 对AES_FCLR的PDFC位写1清除PDFS。

5. 标签阶段

- 配置AES_CTRL的寄存器PRC为3。
- 访问AES输入数据寄存器(AES_IDT)四次，输入计数器第零分组(CTR0)。
- 等待AES_STS的PDFS置位，置起后访问AES输出数据寄存器(AES_ODT)四次，可取得验证标签。
- 对AES_FCLR的PDFC位写1清除PDFS后，重置AESEN禁止AES硬件加密单元。

28.5 数据格式与数据填充

28.5.1 数据格式

依据 NIST FIPS PUB-197 所规范，AES 以 128 比特为一个数据分组进行加解密，以字节为单位进行二维矩阵运算。数据于 AES 硬件加密单元(AES)中是以小端格式取用，明文、密文、密钥与初始向量于寄存器之格式如下

图 28-12 明文、密文的数据格式

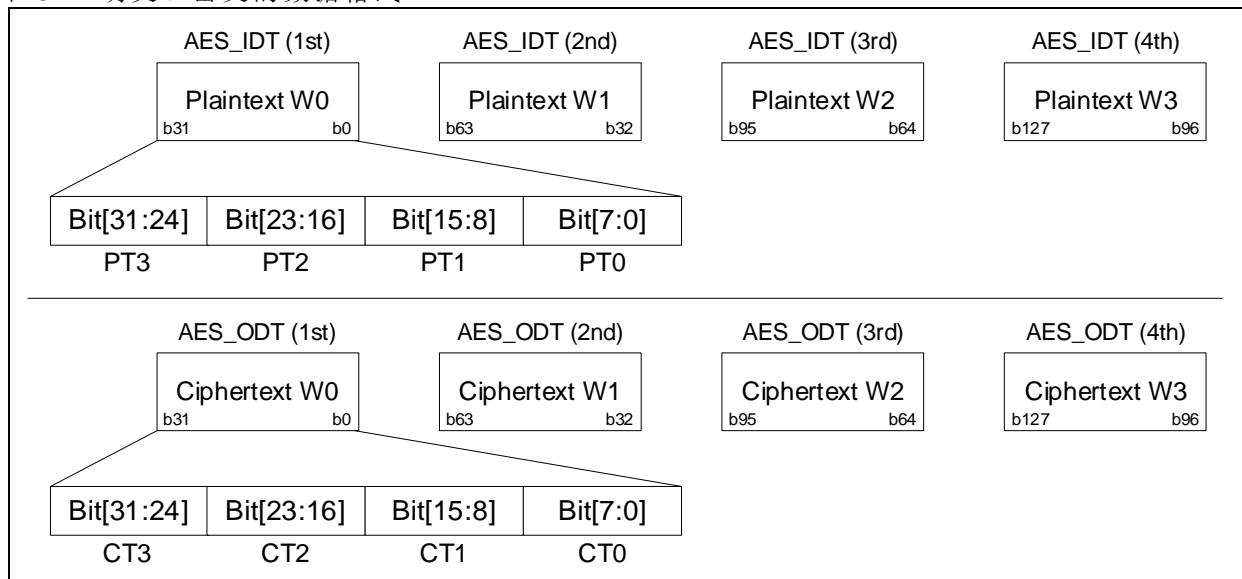
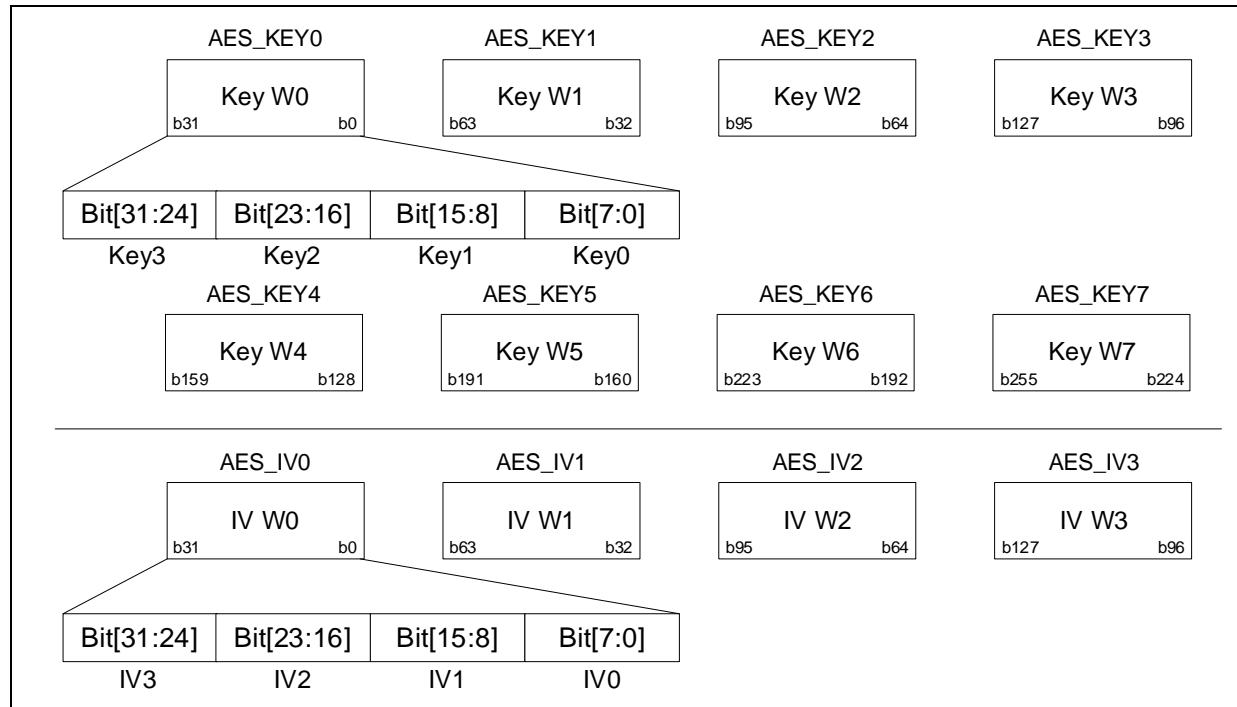
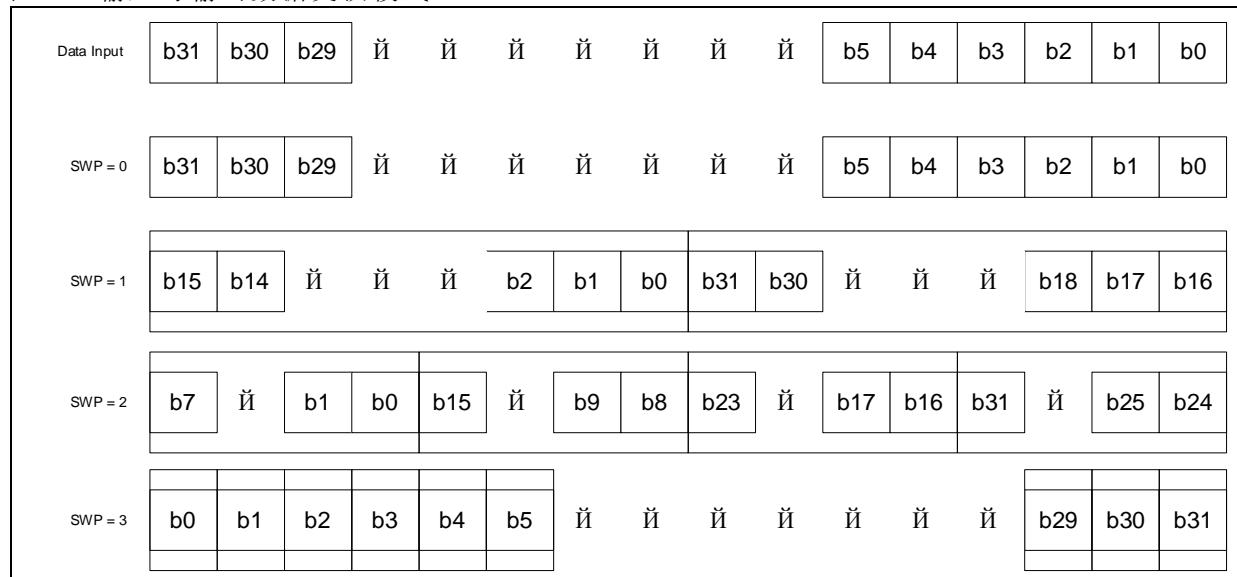


图 28-13 密钥与初始向量的数据格式



为配合不同应用下的数据存储格式，AES 硬件加密单元可支持明文与密文以半字、字节或位做交换。配置 AES_CTRL 的寄存器 SWP，可选择访问 AES 输入数据寄存器(AES_IDT)所输入之数据，以即访问 AES 输出数据寄存器(AES_ODT)所取得之数据，是否进行交换处理，如下图所示：

图 28-14 输入与输出数据交换模式



28.5.2 数据充填流程

在各工作串接模式的加解密阶段，以及伽罗瓦/计数器模式（GCM）与计数器模式搭配密码块链接消息验证码(CCM)的附加身份验证数据阶段，数据以 128 比特为一个数据分组，拆分数据流进行运算。数据分组依序由 AES 输入数据寄存器(AES_IDT)输入，待运算完成，在由 AES 输出数据寄存器(AES_ODT)取出，此过程称为数据充填。数据充填可透过 CPU 或 DMA 进行：

透过 CPU 进行数据充填

CPU 可藉由轮询标志位或是接收中断的方式，控制数据充填的流程，流程如下：

- 1 欲使用接收中断的控制流程，配置 AES_CTRL 的寄存器 PDIE 位与 RWEIE 位为 1。

- 2 配置AES控制寄存器(AES_CTRL)、AES密钥寄存器(AES_KEYx)与AES初始向量寄存器(AES_IVx)完成后，配置AES_CTRL的寄存器AESEN使能AES。
- 3 CPU访问AES输入数据寄存器(AES_IDT)四次，依序输入一个分组的数据。
- 4 等待AES硬件加密单元完成运算，依控制流程有所差异：
 - 如果使用轮询标志位控制流程，CPU反复访问AES_STS，直到PDFS置位。
 - 如果使用中断控制流程，当CPU接收到中断之后，访问AES_STS确认PDFS是否置位。
- 5 若为各工作串接模式的加解密阶段，访问AES输出数据寄存器(AES_ODT)四次，取得运算结果之数据。若为伽罗瓦/计数器模式(GCM)与计数器模式搭配密码块链接消息验证代码(CCM)的附加身份验证数据阶段，则跳过此步骤。
- 6 对AES_FCLR的PDFC位写1，清除PDFS标志位。
- 7 若待运算数据大于16字节，即尚有一个数据分组以上时，重复步骤3至步骤7。
- 8 待运算数据不足16字节时，需由使用者填补零至一个完整数据分组。若为伽罗瓦/计数器模式(GCM)与计数器模式搭配密码块链接消息验证代码(CCM)的加解密阶段，需依补零数量配置AES_CTRL的寄存器虚假数据字节数(NDD)与置位LST位。
- 9 等待AES硬件加密单元完成运算，方式与步骤4相同。
- 10 若有配置AES_CTRL的寄存器NDD，访问AES_STS确认NZDFS位置起。
- 11 若为各工作串接模式的加解密阶段，访问AES输出数据寄存器(AES_ODT)四次，取得运算结果之数据。若为伽罗瓦/计数器模式(GCM)与计数器模式搭配密码块链接消息验证代码(CCM)的附加身份验证数据阶段，则跳过此步骤。
- 12 对AES_FCLR的PDFC位与NZDFC写1，清除PDFS与NZDFS标志位后，重置AESEN。

透过 DMA 进行数据充填

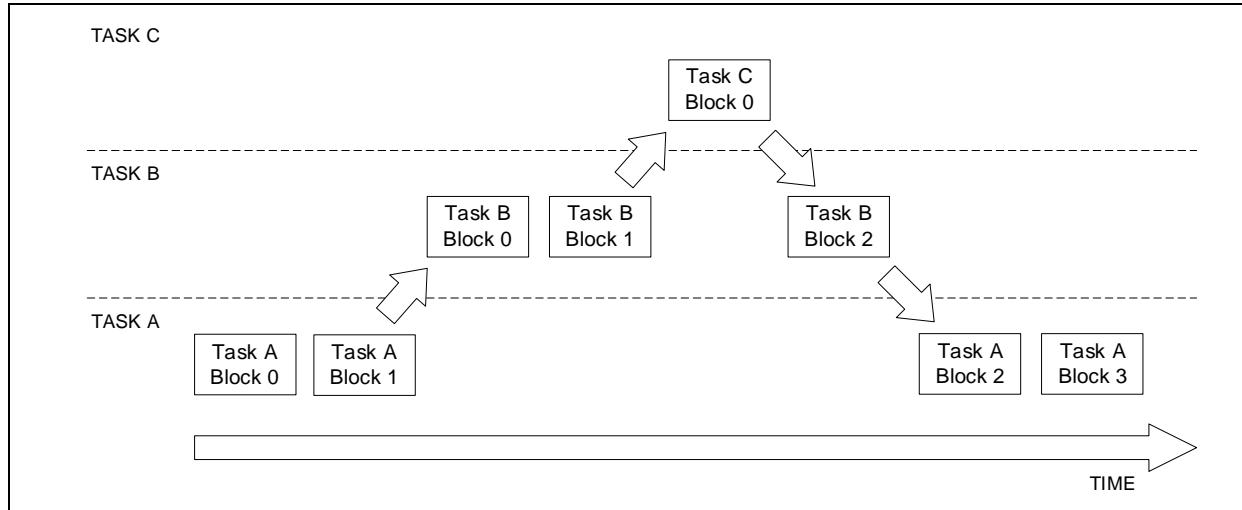
以DMA进行数据充填，可在不耗用CPU资源的状态下进行数据传输。使用者仅须透过中断监控加密单元运作状态即可，其配置流程如下：

- 1 欲运算数据量非16字节之倍数，需由使用者先在数据结尾填补零至完整数据分组。
- 2 配置两组DMA控制器并使能，一组DMA控制器负责由存储单元取出运算数据填入AES_IDT，另一组DMA控制器负责由AES_ODT取出结果数据写入存储单元。需特别注意，若为伽罗瓦/计数器模式(GCM)或计数器模式搭配密码块链接消息验证代码(CCM)的加解密，且运算数据量非16字节之倍数，DMA长度设定需扣除最后分组(加解密最后分组之运算于下一阶段进行)。
- 3 配置AES_CTRL的寄存器DMAWE位与DMARE位为1，使能DMA写信道与读信道。
- 4 配置AES_CTRL的寄存器PDIE位与RWEIE位为1，使能运算完成中断与读写错误中断。
- 5 配置AES控制寄存器(AES_CTRL)、AES密钥寄存器(AES_KEYx)与AES初始向量寄存器(AES_IVx)完成后，配置AES_CTRL的寄存器AESEN使能AES。
- 6 使能AES后，数据传输由DMA与AES自动完成，无须使用者介入。数据充填完成后，AES硬件加密单元会置起AES_STS的PDFS位并发送中断至CPU。
- 7 对AES_FCLR的PDFC位写1，清除PDFS标志位后，重置AESEN。

28.5.3 停滞与恢复流程

对于正在进行数据充填的工作串接模式，AES硬件加密单元(AES)支持以数据分组为单位的停滞操作。停滞操作的目的是暂停并记录当下运算之任务，释放资源以供更高优先权之任务使用，如下图所示意：

图 28-15 停滞与恢复示意图



CPU 充填数据的停滞与恢复流程

1. 依轮询标志位或是接收中断的方式，等待AES_STS的PDFS位置位。
2. 若为各工作串接模式的加解密阶段，访问AES输出数据寄存器(AES_ODT)四次，取得运算结果之数据。若为伽罗瓦/计数器模式(GCM)与计数器模式搭配密码块链接消息验证代码(CCM)的附加身份验证数据阶段，则跳过此步骤。
3. 对AES_FCLR的PDFC位写1，清除PDFS标志位后，重置AESEN禁止AES硬件加密单元。
4. 保存当前配置，内容包含AES控制寄存器(AES_CTRL)、AES密钥寄存器(AES_KEYx)、AES初始向量寄存器(AES_IVx)与AES停滞讯息寄存器(AES_SIx)。
5. 配置并进行高优先权任务。
6. 恢复先前之任务，将步骤4所保存之内容，包含AES控制寄存器(AES_CTRL)、AES密钥寄存器(AES_KEYx)、AES初始向量寄存器(AES_IVx)与AES停滞讯息寄存器(AES_SIx)，重新填回对应之寄存器。
7. 配置AES_CTRL的寄存器AESEN使能AES，即可重回先前之任务。

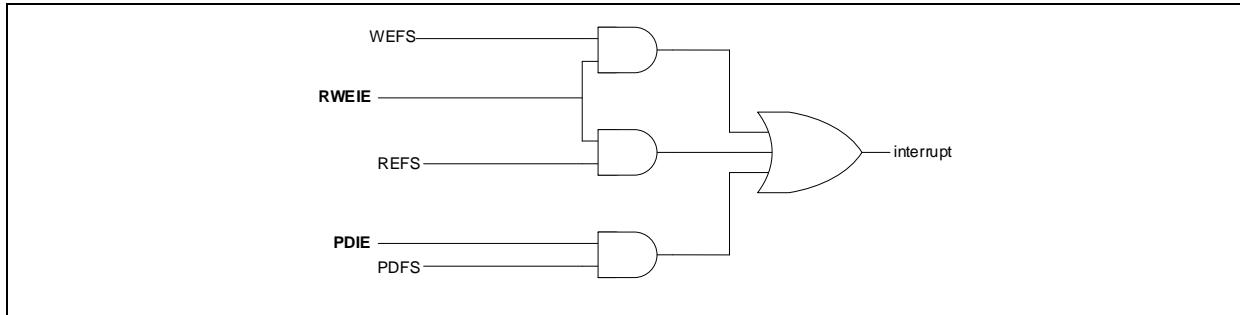
DMA 充填数据的停滞与恢复流程

1. 重置AES_CTRL的寄存器DMAWE位，禁止DMA写通道。
2. 等待中断至CPU，确认AES_STS的PDFS位置位。
3. 重置AES_CTRL的寄存器DMARE位，禁止DMA读通道。
4. 对AES_FCLR的PDFC位写1，清除PDFS标志位后，重置AESEN禁止AES硬件加密单元。
5. 保存当前配置，内容包含AES控制寄存器(AES_CTRL)、AES密钥寄存器(AES_KEYx)、AES初始向量寄存器(AES_IVx)与AES停滞讯息寄存器(AES_SIx)。
6. 若DMA控制器亦须释放给高优先权任务，记录目前剩余的DMA长度。
7. 配置并进行高优先权任务。
8. 恢复先前之任务，将步骤5所保存之内容，包含AES控制寄存器(AES_CTRL)、AES密钥寄存器(AES_KEYx)、AES初始向量寄存器(AES_IVx)与AES停滞讯息寄存器(AES_SIx)，重新填回对应之寄存器。
9. 若DMA控制器亦须恢复，根据原本的DMA起始位置与长度，以及之前记录的剩余的DMA长度，重新配置DMA控制器并使能。
10. 配置AES_CTRL的寄存器AESEN使能AES，即可重回先前之任务。

28.6 中断与中断控制

AES 硬件加密单元(AES)的中断，可由三个标志状态产生，如下图所示：

图 28-16 AES 硬件加密单元的中断控制

**加解密运算完成标志状态(PDFS)**

AES 硬件加密单元完成运算时会置起 AES_STS 的 PDFS 位，若将 AES_CTRL 的寄存器 PDIE 位配置为 1，可发送中断至 CPU。软件对 AES_FCLR 的 PDFC 位写 1，可清除 PDFS 标志位。

读错误标志状态(REFS)

当使用者于输出缓冲处于空状态下访问 AES 输出数据寄存器(AES_ODT)，AES 硬件加密单元会置起 AES_STS 的 REFS 位，若将 AES_CTRL 的寄存器 RWEIE 位配置为 1，可发送错误中断至 CPU。软件对 AES_FCLR 的 REFC 位写 1，可清除 REFS 标志位。

写错误标志状态(WEFS)

当使用者于输入缓冲处于满状态下访问 AES 输入数据寄存器(AES_IDT)，AES 硬件加密单元会置起 AES_STS 的 WEFS 位，若将 AES_CTRL 的寄存器 RWEIE 位配置为 1，可发送错误中断至 CPU。软件对 AES_FCLR 的 WEFC 位写 1，可清除 WEFS 标志位。

另外，曾运算带虚假数据分组标志状态(NZDFS)用于纪录 AES 硬件加密单元曾运算带虚假数据之分组。虚假数据仅只用于数据充填的最终分组不足 16 字节，配置 AES_CTRL 的寄存器 LST 与 NDD，运算结束时 NZDFS 位会置起。此标志位没有中断使能设定，亦即无法产生中断。软件对 AES_FCLR 的 NZDFC 位写 1，可清除 NZDFS 标志位。

28.7 寄存器描述

下表列出了 AES 的寄存器映像和复位值。必须以字（32 位）的方式操作这些外设寄存器。

表 28-1 AES 寄存器映像和复位值

寄存器简称	基址偏移量	复位值
AES_CTRL	0x000	0x0000 0000
AES_STS	0x004	0x0000 0000
AES_IDT	0x008	0x0000 0000
AES_ODT	0x00C	0x0000 0000
AES_KEY0	0x010	0x0000 0000
AES_KEY1	0x014	0x0000 0000
AES_KEY2	0x018	0x0000 0000
AES_KEY3	0x01C	0x0000 0000
AES_IV0	0x020	0x0000 0000
AES_IV1	0x024	0x0000 0000
AES_IV2	0x028	0x0000 0000
AES_IV3	0x02C	0x0000 0000
AES_KEY4	0x030	0x0000 0000
AES_KEY5	0x034	0x0000 0000

AES_KEY6	0x038	0x0000 0000
AES_KEY7	0x03C	0x0000 0000
AES_SI0	0x040	0x0000 0000
AES_SI1	0x044	0x0000 0000
AES_SI2	0x048	0x0000 0000
AES_SI3	0x04C	0x0000 0000
AES_SI4	0x050	0x0000 0000
AES_SI5	0x054	0x0000 0000
AES_SI6	0x058	0x0000 0000
AES_SI7	0x05C	0x0000 0000
AES_FCLR	0x060	0x0000 0000

28.7.1 AES控制寄存器 (AES_CTRL)

域	简称	复位值	类型	功能
位 31: 25	保留	0x00	resd	保持为默认值
位 24	LST	0x0	rw	<p>最后分组 (Processing last block) 0: 运算非最后分组 1: 运算为最后分组</p> <p>该寄存器配置为'1'后，于运算完成时自动清 0</p>
位 23: 20	NDD	0x0	rw	<p>虚假数据字节数 (Number of dummy data) 0: 无虚假数据 1-15: 虚假数据字节数</p> <p>该寄存器只在最后分组设定置起时(LST=1)，方有作用</p>
位 19	保留	0x0	resd	保持为默认值
位 18: 17	KL	0x0	rw	<p>密钥长度 (Key Length) 0: 128 位 (AES-128) 1: 192 位 (AES-192) 2: 256 位 (AES-256) 3: 保留</p>
位 16: 15	保留	0x0	resd	保持为默认值
位 14: 13	PRC	0x0	rw	<p>运行阶段 (Processing stage) 0: 伽罗瓦哈希函数初始化阶段 1: 附加身份验证数据阶段 2: 加解密阶段 3: 标签阶段</p> <p>该设定只在工作串接模式之设定为伽罗瓦/计数器模式或计数器模式搭配密码块链接消息验证代码时(CHN=3 或 CHN=4)，方有作用</p>
位 12	DMARE	0x0	rw	<p>DMA 读通道使能 (DMA read channel enable) 0: DMA 读通道禁止，用户以 CPU 访问 AES 输出数据寄存器(AES_ODT) 取得数据 1: DMA 读通道使能，用户以 DMA 访问 AES 输出数据寄存器(AES_ODT)取得数据</p>
位 11	DMAWE	0x0	rw	<p>DMA 写通道使能 (DMA write channel enable) 0: DMA 写通道禁止，用户以 CPU 访问 AES 输入数据寄存器(AES_IDT)输入数据 1: DMA 写通道使能，用户以 DMA 访问 AES 输入数据寄存器(AES_IDT)输入数据</p>
位 10	RWEIE	0x0	rw	<p>读写错误中断使能 (Read and write error interrupt enable) 0: 读写错误中断禁止</p>

				1: 读写错误中断使能 加解密运算完成中断使能 (Encrypt/Decrypt processing done interrupt enable) 0: 加解密运算完成中断禁止 1: 加解密运算完成中断使能
位 9	PDIE	0x0	rw	保留为默认值
位 8	保留	0x0	resd	工作串接模式 (Chaining mode) 0: 电子密码本模式 (ECB) 1: 密码块链接模式 (CBC) 2: 计数器模式 (CTR) 3: 伽罗瓦/计数器模式 (GCM) 4: 计数器模式搭配密码块链接消息验证代码 (CCM) 5-7: 保留
位 7: 5	CHN	0x0	rw	操作模式 (Encrypt/Decrypt mode) 0: 加密 1: 密钥扩展 2: 解密 3: 密钥扩展开并单分组解密
位 4: 3	OPR	0x0	rw	输入与输出数据交换模式 (Data Swap) 0: 数据不交换 1: 数据以字为处理单位, 实行半字交换 2: 数据以字为处理单位, 实行字节交换 3: 数据以字为处理单位, 实行位交换
位 2: 1	SWP	0x0	rw	AES 使能 (AES Enable) 0: AES 禁止 1: AES 使能
位 0	AESEN	0x0	rw	AES 的寄存器配置, 需在 AES 使能之前完成

28.7.2 AES状态寄存器 (AES_STS)

域	简称	复位值	类型	功能
位 31: 5	保留	0x00000000	resd	保持为默认值 曾运算带虚假数据分组标志状态 (Non-zero dummy data processed flag status)
位 4	NZDFS	0x0	ro	0: 未曾运算带虚假数据之分组 1: 曾经运算带虚假数据之分组 对 AES 标志清除寄存器 (AES_FCLR) 中的 NZDFC 位写入'1'可以清除此状态寄存器
位 3	PBS	0x0	ro	密码运算单元忙碌状态 (Crypto-processor busy status) 0: 密码运算单元运算结束 1: 密码运算单元运算进行中 该寄存器于工作串接模式之设定为伽罗瓦/计数器模式时 (CHN=3), 可能出现加解密运算完成中断状态呈现加解密运算完成(PDFS=1)下, 而密码运算单元忙碌状态呈现密码运算单元运算进行中(PBS=1)。此现象代表数据加解密已完成, 但伽罗瓦哈希计算仍在进行。于其他工作串接模式下, 该寄存器为加解密运算完成中断状态之反码。
位 2	WEFS	0x0	ro	写错误标志状态 (Write error flag status) 0: 未发生写错误 1: 发生写错误 对 AES 标志清除寄存器 (AES_FCLR) 中的 WEFC 位写入'1'可以清除此状态寄存器
位 1	REFS	0x0	ro	读错误标志状态 (Read error flag status) 0: 未发生读错误 1: 发生读错误 对 AES 标志清除寄存器 (AES_FCLR) 中的 REFC 位写入'1'可以清除此状态寄存器
位 0	PDFS	0x0	ro	加解密运算完成标志状态 (Encrypt/Decrypt processing done flag status)

0: 加解密运算未完成
1: 加解密运算完成
对 AES 标志清除寄存器 (AES_FCLR) 中的 PDFC 位写入'1'可以清除此状态寄存器

28.7.3 AES数据输入寄存器 (AES_IDT)

域	简称	复位值	类型	功能
位 31: 0	IDT	0x00000000	wo	数据输入寄存器 (Input Data Port) CPU 或 DMA 可透过访问此寄存器输入数据

28.7.4 AES数据输出寄存器 (AES_ODT)

域	简称	复位值	类型	功能
位 31: 0	ODT	0x00000000	ro	数据输出寄存器 (Output Data Port) CPU 或 DMA 可透过访问此寄存器取得数据

28.7.5 AES密钥寄存器0 (AES_KEY0)

域	简称	复位值	类型	功能
位 31: 0	KEY0	0x00000000	rw	密钥寄存器 0 (Key W0) 私密钥匙第 0-31 比特位

28.7.6 AES密钥寄存器1 (AES_KEY1)

域	简称	复位值	类型	功能
位 31: 0	KEY1	0x00000000	rw	密钥寄存器 1 (Key W1) 私密钥匙第 32-63 比特位

28.7.7 AES密钥寄存器2 (AES_KEY2)

域	简称	复位值	类型	功能
位 31: 0	KEY2	0x00000000	rw	密钥寄存器 2 (Key W2) 私密钥匙第 64-95 比特位

28.7.8 AES密钥寄存器3 (AES_KEY3)

域	简称	复位值	类型	功能
位 31: 0	KEY3	0x00000000	rw	密钥寄存器 3 (Key W3) 私密钥匙第 96-127 比特位

28.7.9 AES初始向量寄存器0 (AES_IV0)

域	简称	复位值	类型	功能
位 31: 0	IV0	0x00000000	rw	初始向量寄存器 0 (Initial Vector W0) 初始向量第 0-31 比特位

28.7.10 AES初始向量寄存器1 (AES_IV1)

域	简称	复位值	类型	功能
位 31: 0	IV1	0x00000000	rw	初始向量寄存器 1 (Initial Vector W1) 初始向量第 32-63 比特位

28.7.11 AES初始向量寄存器2 (AES_IV2)

域	简称	复位值	类型	功能
位 31: 0	IV2	0x00000000	rw	初始向量寄存器 2 (Initial Vector W2) 初始向量第 64-95 比特位

28.7.12 AES初始向量寄存器3 (AES_IV3)

域	简称	复位值	类型	功能
位 31: 0	IV3	0x00000000	rw	初始向量寄存器 3 (Initial Vector W3) 初始向量第 96-127 比特位

28.7.13 AES密钥寄存器4 (AES_KEY4)

域	简称	复位值	类型	功能
位 31: 0	KEY4	0x00000000	rw	密钥寄存器 4 (Key W4) 私密钥匙第 128-159 比特位 该寄存器只在密钥长度之设定为 192 位或 256 位时(KL=1 或 KL=2), 方有作用

28.7.14 AES密钥寄存器5 (AES_KEY5)

域	简称	复位值	类型	功能
位 31: 0	KEY5	0x00000000	rw	密钥寄存器 5 (Key W5) 私密钥匙第 160-191 比特位 该寄存器只在密钥长度之设定为 192 位或 256 位时(KL=1 或 KL=2), 方有作用

28.7.15 AES密钥寄存器6 (AES_KEY6)

域	简称	复位值	类型	功能
位 31: 0	KEY6	0x00000000	rw	密钥寄存器 6 (Key W6) 私密钥匙第 192-223 比特位 该寄存器只在密钥长度之设定为 256 位时(KL=2), 方有作用

28.7.16 AES密钥寄存器7 (AES_KEY7)

域	简称	复位值	类型	功能
位 31: 0	KEY7	0x00000000	rw	密钥寄存器 7 (Key W7) 私密钥匙第 224-255 比特位 该寄存器只在密钥长度之设定为 256 位时(KL=2), 方有作用

28.7.17 AES停滞讯息寄存器0 (AES_SI0)

域	简称	复位值	类型	功能
位 31: 0	SI0	0x00000000	rw	停滞讯息寄存器 0 (Suspend information 0) 停滞纪录讯息 0 该设定只在工作串接模式之设定为伽罗瓦/计数器模式或计数器模式搭配密码块链接消息验证代码时(CHN=3 或 CHN=4), 方有作用

28.7.18 AES停滞讯息寄存器1 (AES_SI1)

域	简称	复位值	类型	功能
位 31: 0	SI1	0x00000000	rw	停滞讯息寄存器 1 (Suspend information 1) 停滞纪录讯息 1

该设定只在工作串接模式之设定为伽罗瓦/计数器模式或计数器模式搭配密码块链接消息验证代码时(CHN=3 或 CHN=4)，方有作用

28.7.19 AES停滞讯息寄存器2 (AES_SI2)

域	简称	复位值	类型	功能
位 31: 0	SI2	0x00000000	rw	停滞讯息寄存器 2 (Suspend information 2) 停滞纪录讯息 2 该设定只在工作串接模式之设定为伽罗瓦/计数器模式或计数器模式搭配密码块链接消息验证代码时(CHN=3 或 CHN=4)，方有作用

28.7.20 AES停滞讯息寄存器3 (AES_SI3)

域	简称	复位值	类型	功能
位 31: 0	SI3	0x00000000	rw	停滞讯息寄存器 3 (Suspend information 3) 停滞纪录讯息 3 该设定只在工作串接模式之设定为伽罗瓦/计数器模式或计数器模式搭配密码块链接消息验证代码时(CHN=3 或 CHN=4)，方有作用

28.7.21 AES停滞讯息寄存器4 (AES_SI4)

域	简称	复位值	类型	功能
位 31: 0	SI4	0x00000000	rw	停滞讯息寄存器 4 (Suspend information 4) 停滞纪录讯息 4 该设定只在工作串接模式之设定为伽罗瓦/计数器模式或计数器模式搭配密码块链接消息验证代码时(CHN=3 或 CHN=4)，方有作用

28.7.22 AES停滞讯息寄存器5 (AES_SI5)

域	简称	复位值	类型	功能
位 31: 0	SI5	0x00000000	rw	停滞讯息寄存器 5 (Suspend information 5) 停滞纪录讯息 5 该设定只在工作串接模式之设定为伽罗瓦/计数器模式或计数器模式搭配密码块链接消息验证代码时(CHN=3 或 CHN=4)，方有作用

28.7.23 AES停滞讯息寄存器6 (AES_SI6)

域	简称	复位值	类型	功能
位 31: 0	SI6	0x00000000	rw	停滞讯息寄存器 6 (Suspend information 6) 停滞纪录讯息 6 该设定只在工作串接模式之设定为伽罗瓦/计数器模式或计数器模式搭配密码块链接消息验证代码时(CHN=3 或 CHN=4)，方有作用

28.7.24 AES停滞讯息寄存器7 (AES_SI7)

域	简称	复位值	类型	功能
位 31: 0	SI7	0x00000000	rw	停滞讯息寄存器 7 (Suspend information 7) 停滞纪录讯息 7

该设定只在工作串接模式之设定为伽罗瓦/计数器模式或计数器模式搭配密码块链接消息验证代码时(CHN=3 或 CHN=4)，方有作用

28.7.25 AES标志清除寄存器 (AES_FCLR)

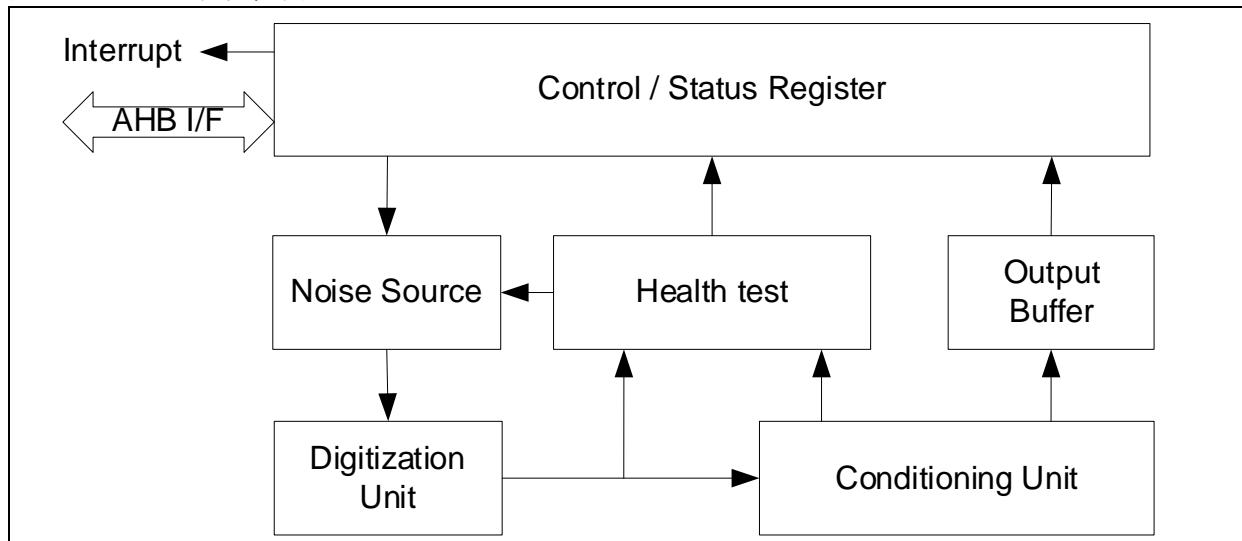
域	简称	复位值	类型	功能
位 31: 5	保留	0x00000000	resd	保持为默认值
位 4	NZDFC	0x0	wo	曾运算带虚假数据分组标志清除 (Non-zero dummy data processed flag clear) 软件对此位写'1'，可清除 AES 状态寄存器 (AES_STS) 的 NZDFS 标志位
位 3	保留	0x0	resd	保持为默认值
位 2	WEFC	0x0	wo	写错误标志清除 (Write error flag clear) 软件对此位写'1'，可清除 AES 状态寄存器 (AES_STS) 的 WEFS 标志位
位 1	REFC	0x0	wo	读错误标志清除 (Read error interrupt clear) 软件对此位写'1'，可清除 AES 状态寄存器 (AES_STS) 的 REFS 标志位
位 0	PDFC	0x0	wo	加解密运算完成标志清除 (Encrypt/Decrypt processing done interrupt clear) 软件对此位写'1'，可清除 AES 状态寄存器 (AES_STS) 的 PDFS 标志位

29 真随机数发生器 (TRNG)

29.1 简介

真随机数发生器(TRNG)用于产生密码或安全应用所需要的随机数据。遵循 NIST SP800-90B 建议的熵源设计与检测方式，TRNG 可以达到最小熵算法之下的最佳熵值，并可透过健康检测机制进行监控，确保生成内容具备不可预测性。每次发生器的运作周期可以产生 128 位的随机数据，透过 AHB 总线，每次可以取得 32 位之随机数据。

图 29-1 TRNG 系统框图



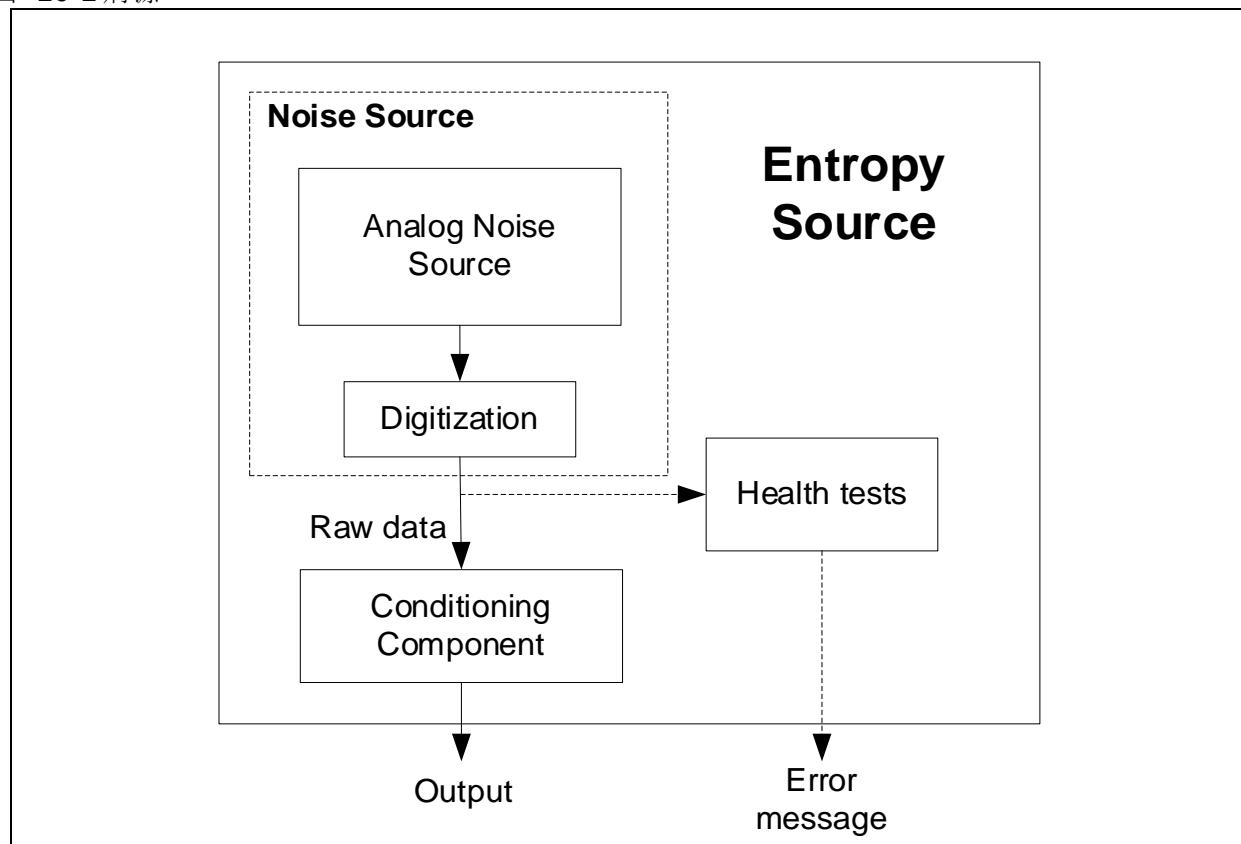
29.2 主要特点

- 遵循 NIST SP800-90B 之随机数发生器设计。
- 内建初始程序之健康检测机制。
 - 对噪声源进行重复次数测试。
 - 对噪声源进行自适应比例测试。
 - 对调节组件进行已知答案测试。
- 内建运行程序之健康检测机制。
 - 对噪声源进行重复次数测试。
 - 对噪声源进行自适应比例测试。
- 内建特定条件之健康检测机制。
 - 对噪声源进行重复翻转测试。
 - 对噪声源时钟进行时钟卡死测试。
- 健康检测电路之测试结果，可反应于错误标志位，以及对应之中断信号上。
- 噪声源自动重置功能，可排除因概率而产生的不良随机数。
- 使用 NIST SP800-90B 所公开，历经审查之调节组件来进行噪声源后处理。
- 透过 AHB 总线，每次可以取得32位之随机数据。
- 每次发生器的运作周期可以产生128位的随机数据。

29.3 熵源

熵源是真随机数发生器的主结构体，按照 NIST SP800-90B 的划分方式，TRNG 所使用的熵源(Entropy source)包含噪声源(Noise source)、健康检测机制(Health test)以及调节组件(Conditioning Component)，如图 1-2 所示。

图 29-2 熵源



29.3.1 噪声源

噪声源是真随机数发生器(TRNG)的根基。多组自主运作的环形振荡器构成模拟噪声源，透过数字化单元转换成一比特码流的原始数据(Raw data)输出，噪声源可提供真随机数发生器所必需的无法预测性。

29.3.2 调节组件

噪声源的输出可能存在着统计特性上的偏移，调节组件之使用是为了改善此偏移现象。TRNG 使用 NIST SP800-90B 所建议，经过审查的 CBC-MAC 算法搭配 AES 区块密码器做为调节组件。本调节组件以 128 比特为单位进行运算处理，完成后将数据存入输出缓冲，并置起 TRNG_STS 的 DTRDY。

29.3.3 健康检测机制

为了确保熵源的正常运行，并未遭受外在环境之蓄意或非蓄意影响，健康检测机制为真随机数发生器以准确并快速的检测方式，确保所产生的随机数据之可靠性。测试结果反应于错误标志位，以及对应之中断信号上，并适时的阻断噪声源输出。依 NIST SP800-90B 之分类，健康检测机制可分为三个形式：

初始程序检测

为了确认电路可以正确运行，在芯片重新上电，或是配置时钟与复位管理(CRM)中的 RNG_RESET 重置 TRNG 模块之后，初次使用真随机数发生器之前，所进行之检测。在检测合格之前，用户无法从数据寄存器取得随机数据(访问数据寄存器仅可取得全零数据)。检测合格之后，TRNG 方可进入运行程序。检测内容有三项：

- 对噪声源进行重复次数测试

重复次数测试(Repetition count test)的目的，是快速地确认噪声源所输出的码流，是否卡死在固定的状态。当噪声源连续输出了超过 42 位的"0"或"1"时，检测机制重置噪声源并重新进行检测。重置噪声源后，当再度连续输出了超过 42 位的"0"或"1"时，此时检测机制判定噪声源卡死，并置起 TRNG_STS 的 ESFES 位。

- 对噪声源进行自适应比例测试

自适应比例测试(Adaptive proportion test)的目的，是判断噪声源所提供的熵值，是否有严重缺损的状态。当噪声源受到不预期之影响，可能造成输出"0"或"1"过度频繁。取噪声源输出前 1024 比特做分析，若"0"或"1"的总数超过 690 位，检测机制即判定噪声源熵值不足，并置起 TRNG_STS 的 ESFES 位。

- 对调节组件进行已知答案测试

已知答案测试(Known answer test)的目的，是确认调节组件的正常运行。检测机制对调节组件输入三组固定内容之数据，再将调节组件之输出与预存的答案进行比对。当比对结果不相符时，检测机制判定调节组件无法正常运行，并置起 TRNG_STS 的 ESFES 位。

运行程序检测

为了防范在 TRNG 运行时遭受外在环境影响，当 TRNG 进入运行程序后，检测机制会以不干扰 TRNG 运行的方式，持续对噪声源进行检测。运行程序检测的检测内容与初始程序检测相仿，当检测发现异常时，检测机制会中止 TRNG 运行，并舍弃异常期间产出的随机数据。检测内容有二项：

- 对噪声源进行重复次数测试

检测机制在 TRNG 运行的过程中对噪声源持续进行检测。当噪声源连续输出了超过 42 位的"0"或"1"时，检测机制重置噪声源，并舍弃当下已取得的原始数据。重置噪声源后，当再度连续输出了超过 42 位的"0"或"1"时，此时检测机制判定噪声源卡死，中止 TRNG 运行并置起 TRNG_STS 的 ESFES 位。

- 对噪声源进行自适应比例测试

在 TRNG 运行的过程中，检测机制以滑动窗户对噪声源当下与之前输出共计 1024 比特进行检测。当噪声源输出"0"或"1"的总数超过 690 位，检测机制即判定噪声源熵值不足，中止 TRNG 运行并置起 TRNG_STS 的 ESFES 位。

其他特定条件检测

除上述检测机制外，为了更精准的判断噪声源是否异常，检测机制还有额外的测试项目如下：

- 对噪声源进行重复翻转测试

检测机制在 TRNG 运行的过程中对噪声源持续进行检测。当噪声源连续输出了超过 21 组的"01"或"10"时，检测机制重置噪声源，并舍弃当下已取得的原始数据。重置噪声源后，当再度连续输出了超过 21 组的"01"或"10"时，此时检测机制判定噪声源异常，中止 TRNG 运行并置起 TRNG_STS 的 ESFES 位。

- 对噪声源之时钟进行时钟卡死测试

检测机制在 TRNG 运行的过程，以 AHB 时钟的 32 分频对噪声源之原始时钟进行侦测，若该时钟于 3 个检测时钟之内未发生翻转行为，检测机制即判定噪声源时钟异常，中止 TRNG 运行并置起 TRNG_STS 的 CKFES 位。

29.4 状态与中断控制

真随机数发生器(TRNG)的寄存器 TRNG_STS，用于纪录 TRNG 的运行状态与错误事件。通过配置 TRNG_CTRL 的 DTRIE 与 FIE，可以对应的事件产生中断发送至 CPU。由于 TRNG 是以全局中断连接 CPU 的 NVIC，使用者需于此寄存器确认中断信息，寄存器信息描述如下：

数据寄存器就绪(Data port ready)

DTRDY 表现 TRNG 的输出缓冲之内尚有可取用之随机数据，用户可以透过访问数据寄存器取得。当噪声源使能(NSEN=1)且调节组件与输出缓冲使能(CUDIS=0)，TRNG 开始运行生成随机数据，并在生成完成，存入输出缓冲后置起该位。当输出缓冲内的四个 32 位的随机数据皆取出后，该位重置。DTRIE 使能下，DTRDY 置起会产生中断发送至 CPU。

时钟错误状态(Clock fail)

CKFES 用于表现噪声源时钟之状态。检测机制判定噪声源时钟异常时会置起该位。若该位置起，请参考 29.5.3 执行错误状态重置流程。若 TRNG 重置后 CKFES 仍处于置位状态，表示噪声源时钟处于损害状态，无法恢复。当 CKFES 置起时，噪声源停止原始数据产生。

FIE 使能下，检测机制判定噪声源时钟异常时会同时置起 CKFIS，并产生中断发送至 CPU。可对该位写'1'清除之。

熵源错误状态(Entropy source fail)

ESFES 用于表现熵源之状态。于下述情况下，检测机制判定熵源异常并置起该位。

- 噪声源于初始与运行程序检测中，无法通过重复次数测试。
- 噪声源于初始与运行程序检测中，无法通过自适应比例测试
- 噪声源于运行程序检测中，无法通过重复翻转测试
- 调节组件于初始程序检测中，无法通过已知答案测试

若该位置起，请参考 29.5.3 执行错误状态重置流程。若 TRNG 重置后 ESFES 仍处于置位状态，表示熵源处于损害状态，无法恢复。当 ESFES 置起时，噪声源停止原始数据产生。

FIE 使能下，检测机制判定熵源异常时会同时置起 ESFIS，并产生中断发送至 CPU。可对该位写'1'清除之。

29.5 硬件运行、配置流程与错误重置

29.5.1 真随机数发生器运行流程

启动真随机数发生器(TRNG)后，硬件的运行流程可以分成初始程序与运行程序两个部分：

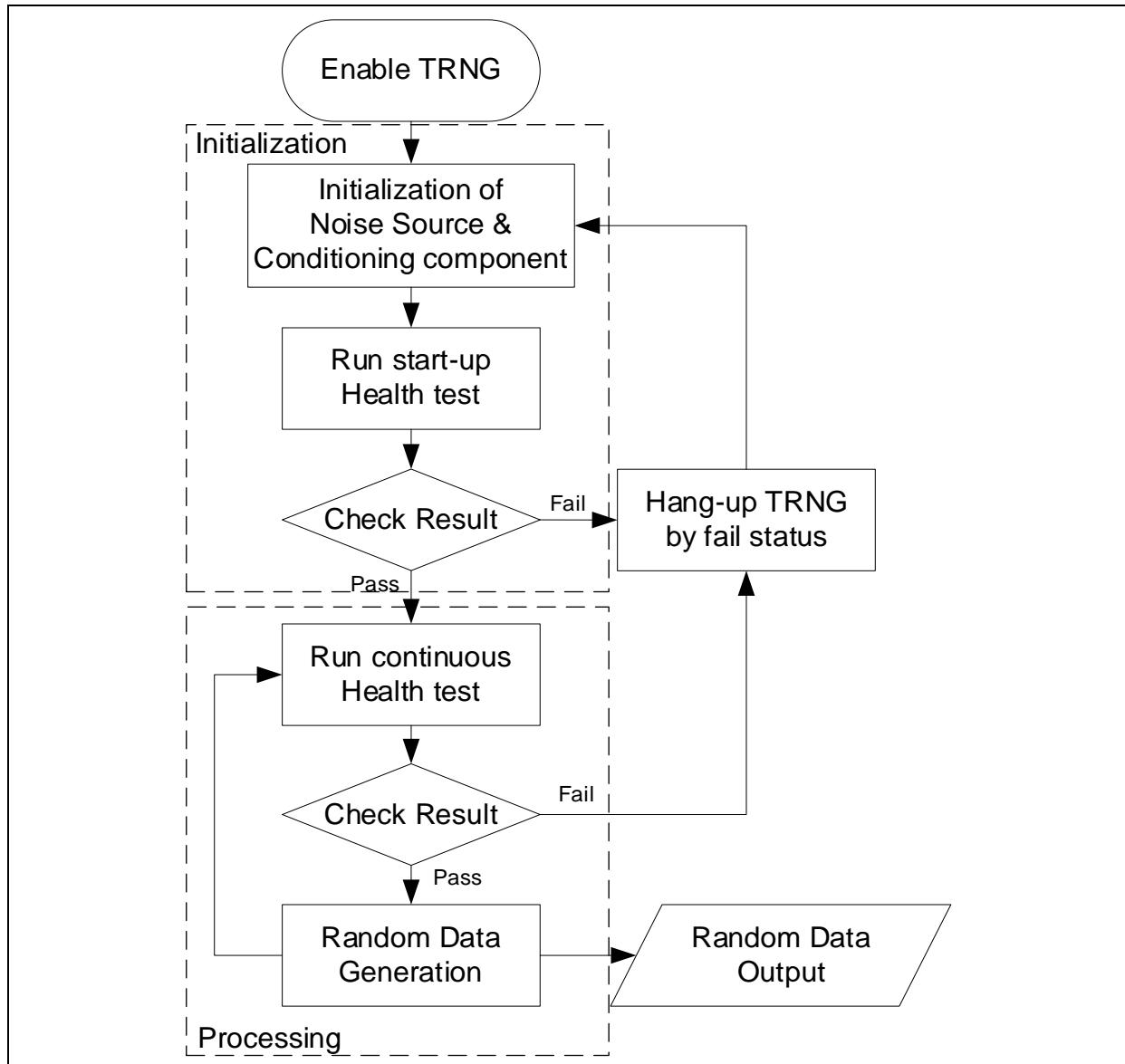
初始程序(Initialization)

当配置噪声源使能(NSEN=1)且调节组件与输出缓冲使能(CUDIS=0)后，即可初始化噪声源与调节组件，启动真随机数发生器。为了确认电路可以正确运行，噪声源与调节组件初始化完成后，检测机制开始进行初始程序检测。依 NIST SP800-90B 之建议，初始程序检测过程中，噪声源所产生之码流皆不被使用。若检测结果无法通过，对应的错误状态位置起，并停滞噪声源之输出。

运行程序(Processing)

通过初始程序检测后，TRNG 进入运行程序生成随机数据，同时检测机制会进行运行程序检测。检测对象是直接针对噪声源，当检测判定异常时，进入调节组件做运算处理的原始数据将直接被舍弃，并中止 TRNG 运行，避免使用者取得异常状态下生成的随机数据。当 TRNG 正常运行下，约每 450 AHB 时钟周期可以生成四个 32 位的随机数据，配置 TRNG_CTRL 的 CLKDIV 将拉长随机数据的生成周期。

图 29-3 流程图



29.5.2 配置流程

TRNG 使用与配置流程如下：

1. 配置TRNG_CTRL的寄存器CUDIS为1后，配置TRNG之设定(ex. CLKDIV)。
2. 若需使用中断讯号，于TRNG_CTRL对应配置DTRIE与FIE。
3. 配置TRNG_CTRL的寄存器CUDIS为0后，配置NSEN为1启动TRNG。
4. 等待TRNG_STS的DTRDY置位，并判断是否有错误状态产生，方式如下：
 - 若使用中断(DTRIE=1)，待CPU的NVIC收到中断讯号，检查ESFIS与CKFIS是否置起，并确认DTRDY是否置位。
 - 若未使用中断(DTRIE=0)，使用者以间隔时间方式反复访问TRNG_STS，检查ESFES与CKFES是否置起，并确认DTRDY是否置位。
5. 确认DTRDY置位且无错误状态，表示TRNG随机数据生成完成并存入输出缓冲中。此时若有使能中断控制，请关闭。
6. DTRDY处于置位状态下，即可透过访问数据寄存器取得一组32位之随机数据，直到输出缓冲中四组32位随机数据全数取出后，DTRDY自动重置。
7. 返回步骤2，进行下一个运作周期。

29.5.3 错误状态重置流程

由于 TRNG 所产生的随机数据常用于密码或安全应用等敏感场景，故须确保生成内容之不可预测性。TRNG 硬件设计本着宁枉勿纵之原则，使用检测机制持续监控，以置起 CKFES 标志位与 ESFES 标志位并停滞噪声源的方式，确保其正常运行，并未遭受外在环境之蓄意或非蓄意影响。且由于影响状况可分为偶发性影响或持续性影响，故当错误状态置起时，用户需执行重置流程并确认错误状态是否能排除。流程如下：

1. 配置 TRNG_CTRL 的 FESCLR，对此位写'1'后，确认 CKFES 或 ESFES 是否清除。无法清除则须执行步骤 2，若可清除，TRNG 将在重新执行初始程序检测后，恢复正常运行程序。
2. 配置噪声源禁止 (NSEN=0) 且调节组件与输出缓冲禁止 (CUDIS=1) 关闭 TRNG 后，重新配置噪声源使能 (NSEN=1) 且调节组件与输出缓冲使能 (CUDIS=0) 启用 TRNG。确认 CKFES 或 ESFES 是否已被清除，若可清除，TRNG 将在初始程序完成后，恢复正常运行程序。
3. 若步骤 1 与步骤 2 皆无法清除，代表 TRNG 受到持续性影响，无法正常提供随机数据，建议停止使用。

29.6 寄存器描述

下表列出了 TRNG 的寄存器映像和复位值。必须以字（32 位）的方式操作这些外设寄存器。

表 29-1 TRNG 寄存器映像和复位值

寄存器简称	基址偏移量	复位值
TRNG_CTRL	0x000	0x0020 0000
TRNG_STS	0x004	0x0000 0000
TRNG_DT	0x008	0x0000 0000
TRNG_HTCTRL	0x00C	0x0004 2ce8

29.6.1 TRNG 控制寄存器 (TRNG_CTRL)

域	简称	复位值	类型	功能
位 31	PRGLK	0x0	rw1s	配置寄存器锁定 (Programming register lock) 0: 允许配置 TRNG_HTCTRL[19:0] 与 TRNG_CTRL[28:5] 1: 禁止配置 TRNG_HTCTRL[19:0] 与 TRNG_CTRL[28:5] 仅可写入 1 置位此寄存器，不允许写 0 清除。 如需解除编写锁定功能，需配置时钟与复位管理(CRM)中的 RNG_RESET，重置 TRNG 模块。
位 30	CUDIS	0x0	rw	调节组件与输出缓冲禁止 (Conditioning unit and output buffer disable) 0: 调节组件与输出缓冲使能。 1: 调节组件与输出缓冲禁止。 配置 TRNG_HTCTRL[19:0] 与 TRNG_CTRL[28:5] 需于调节组件与输出缓冲禁止 (CUDIS=1) 下进行。
位 29	FESCLR	0x0	wo	时钟错误事件与随机数据错误事件清除 (Clock fail event and entropy source fail event status clear) 软件对此位写'1'，可清除 TRNG 状态寄存器 (TRNG_STS) 的 CKFES 标志位以及 ESFES 标志位
位 28: 22	保留	0x0	resd	请保持默认值。
位 21: 20	HT0	0x2	rw	健康测试参数 0 (Health test parameter 0) 请保持默认值。
位 19: 16	CLKDIV	0x0	rw	噪声源时钟分频系数 (Noise source clock divider factor) 0: 噪声源时钟以原始时钟采样 1: 噪声源时钟以原始时钟之 2 分频采样 2: 噪声源时钟以原始时钟之 4 分频采样 3: 噪声源时钟以原始时钟之 8 分频采样 4: 噪声源时钟以原始时钟之 16 分频采样 5: 噪声源时钟以原始时钟之 32 分频采样 ...

				15: 噪声源时钟以原始时钟之 32768 分频采样
位 15: 5	保留	0x00	resd	请保持默认值。
位 4	DTRIE	0x0	rw	数据寄存器就绪中断使能 (Data port ready interrupt enable) 0: 数据寄存器就绪中断禁止 1: 数据寄存器就绪中断使能
位 3	FIE	0x0	rw	错误中断使能 (Fail interrupt enable) 0: 错误中断禁止 1: 错误中断使能
位 2	NSEN	0x0	rw	噪声源使能 (Noise source enable) 0: 噪声源禁止 1: 噪声源使能 配置 TRNG_HTCTRL[19:0]与 TRNG_CTRL[28:5]需于噪声源禁止(NSES=1)下进行。
位 1: 0	保留	0x0	resd	请保持默认值。

29.6.2 TRNG状态寄存器 (TRNG_STS)

域	简称	复位值	类型	功能
位 31: 7	保留	0x00000000	resd	请保持默认值。
位 6	ESFIS	0x0	rw1c	熵源错误中断状态 (Entropy source fail interrupt status) 0: 熵源正常运作 1: 熵源错误中断发生 软件对此位写'1', 可清除此状态寄存器
位 5	CKFIS	0x0	rw1c	时钟错误中断状态 (Clock fail interrupt status) 0: 噪声源时钟正常运作 1: 噪声源时钟错误中断发生 软件对此位写'1', 可清除此状态寄存器
位 4: 3	保留	0x0	resd	请保持默认值。
位 2	ESFES	0x0	ro	熵源错误事件状态 (Entropy source fail event status) 0: 熵源正常运作 1: 熵源错误
位 1	CKFES	0x0	ro	时钟错误事件状态 (Clock fail event status) 0: 噪声源时钟正常运作 1: 噪声源时钟错误
位 0	DTRDY	0x0	ro	数据寄存器就绪 (Data port ready) 0: 随机数据生成未完成 1: 随机数据生成完成

29.6.3 TRNG数据寄存器 (TRNG_DT)

域	简称	复位值	类型	功能
位 31: 0	DT	0x00000000	ro	数据寄存器 (Data Port) 数据寄存器就绪 (DTRDY = 1) 时, 可透过访问此寄存器取得一组 32 位之随机数据。 数据寄存器未就绪 (DTRDY = 0) 时, 访问此寄存器仅可取得全零数据。

29.6.4 TRNG健康测试控制寄存器 (TRNG_HTCTRL)

域	简称	复位值	类型	功能
位 31: 20	保留	0x00000	resd	请保持默认值。
位 19: 0	HT1	0x42ce8	ro	健康测试参数 1 (Health test parameter 1)

30 四线 SPI (QSPI)

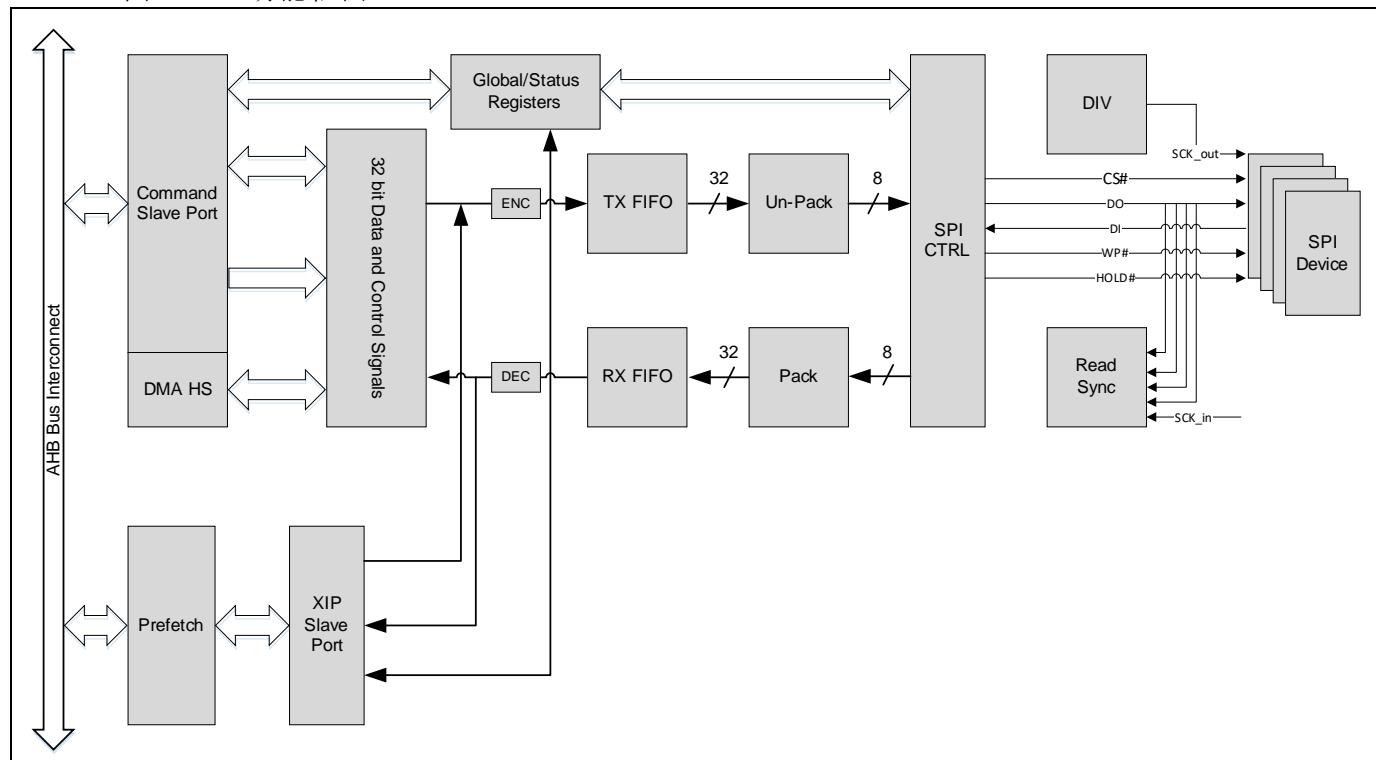
30.1 QSPI简介

QSPI 包含一个基于命令的从端口，一个 XIP（直接地址映射读取）从端口和一个用于执行 SPI Flash 命令的 QSPI 接口控制器。命令从端口用于访问寄存器和数据端口，XIP 从端口用于读取直接地址映射的数据。此外，QSPI 还提供从 AHB 数据端口以 PIO 模式或 DMA 模式访问数据。

30.2 QSPI主要特性

- 支持 DMA 握手模式和 CPU PIO 模式
- 支持 SPI 串行模式，双线/四线输出模式，双线/四线 I/O 模式和 DPI/QPI 模式
- 支持 XIP（直接地址映射读取/写入）端口
- XIP 端口支持预取（具有 2 路关联读取缓存）功能
- TxFIFO/RxFIFO 深度为 128 字节
- 支持可编程分频器
- 支持资料加密

图 30-1 功能框图



30.3 功能描述

30.3.1 命令从端口

QSPI 具有一个命令从端口，包括寄存器和数据端口。用户可以通过该从端口访问寄存器或数据端口。命令字寄存器必须连续依序写入 (CMD_W3 最后写入) 并以字大小访问；而其他寄存器（包括数据端口）可以通过字节，半字和字大小进行访问。

30.3.2 CPU PIO模式

用户可以通过 PIO 模式或 DMA 模式访问数据端口。在 PIO 模式下，用户必须查询 RxFIFO / TxFIFO 准备就绪寄存器 (0x18)。当 RxFIFO 准备就绪时，用户可以读取整个 RxFIFO 数据或将 RxFIFO 的数据保留出去。当 TxFIFO 准备就绪时，用户可以将整个数据写入 TxFIFO。每当在 PIO 模式下运行时，用户都必须确保轮询状态。

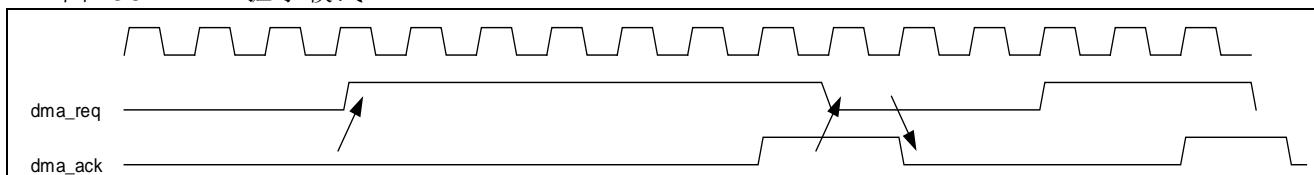
30.3.3 DMA握手模式

DMA 模式是访问数据端口的另一种选择。用户必须将 DMA 控制器寄存器设置为 DMA 握手模式。主机控制器将通过达到接收/发送 FIFO 的阈值触发电平来发出 DMA 请求。接收/发送阈值触发单位为字，当最后一笔传输的数据小于阈值触发电平时，仍触发 DMA 请求。

DMA 设定在 P2M (peripheral to memory) 模式下，也就是 QSPI read data 时，只能以 WORD 为最小传输单位。DMA 的长度需换算成 WORD 单位，数据宽度也必须选择 WORD。

2DMA 设定在 M2P (memory to peripheral) 模式下，也就是 QSPI write data 时，单笔传输最大长度为 128 bytes。如果超过必须分成多次启动 DMA 传输。最小传输单位为 BYTE。

图 30-2 DMA握手模式



30.3.4 XIP（直接地址映射读取/写入）端口

QSPI 具有一个直接地址映射读取/写入 (XIP) 端口。用户可以通过该从端口直接访问系统地址的数据。此外，此时用户只能访问 0x10 寄存器来选择命令从端口或 XIP 端口，或读取除数据寄存器 (0x100) 以外的其他寄存器。请注意，在使用 XIP 端口从 Flash 读取数据时，无法引用中止功能的信息 (0x10，位 [8]) (这意味着 0x10 的位[20]为 1)，并且中止功能不能中止 XIP 端口。用户应按照端口切换顺序进行端口切换。用户系统中必须有另一个主存储器，以便在此版本中，可以先将端口切换代码加载到主存储器中，然后在主存储器上进行端口切换。

30.3.5 XIP端口预取功能

XIP 端口支持具有 2 路关联读取缓存的读取预取功能。

30.3.6 SPI器件操作

命令段

QSPI 的所有操作都需要命令段。用于指示操作目的和 SPI 操作模式 (几线通信)。

注：命令段可以配置为 1/2byte，但是配置为 2byte 的时候发出的命令是 2 个重复的 1byte 命令--只有 1byte 的寄存器位置用于存放命令字。

地址段

在对存储器件进行读/写/擦除操作的时候需要地址段。

命令模式下，地址段长度 0~4byte 可选；XIP 模式下，地址段长度地址段长度 3byte/4byte 可选。

虚拟周期段

用于给被读器件缓冲时间，以准备好后续需要被读的数据。XIP 和命令模式都需要，分别通过 DUM2 和 XIPR_DUM2 配置。另外 XIP 模式的写时序也预留了可配置的虚拟周期时间，通过 XIPW_DUM2 配置，但通常配置为 0

数据段

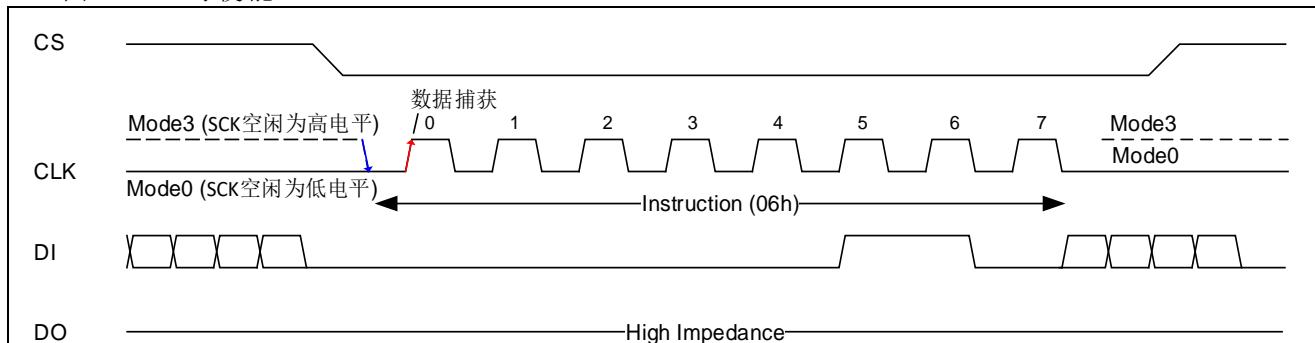
在读取存储器件数据/状态寄存器的时候作为输入；在对存储器件写入的时候作为输出。

串行(1-1-1) 模式

主机控制器支持 SPI 协议（模式 0 和模式 3）。用户必须根据 SPI Device 的规范设置命令队列寄存器。请参考图 30-3 至图 30-10 所示的波形来设置命令队列。

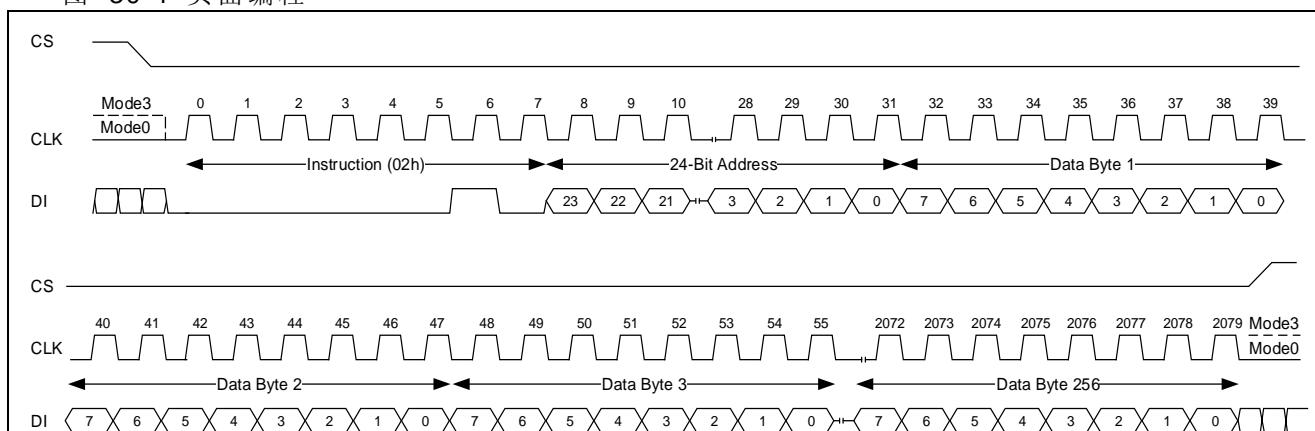
如果用户要执行写使能命令，请将指令代码设置为 06h，写使能，并将指令长度设置为“1”。有关更多详细信息，请参考下图。

图 30-3 写使能



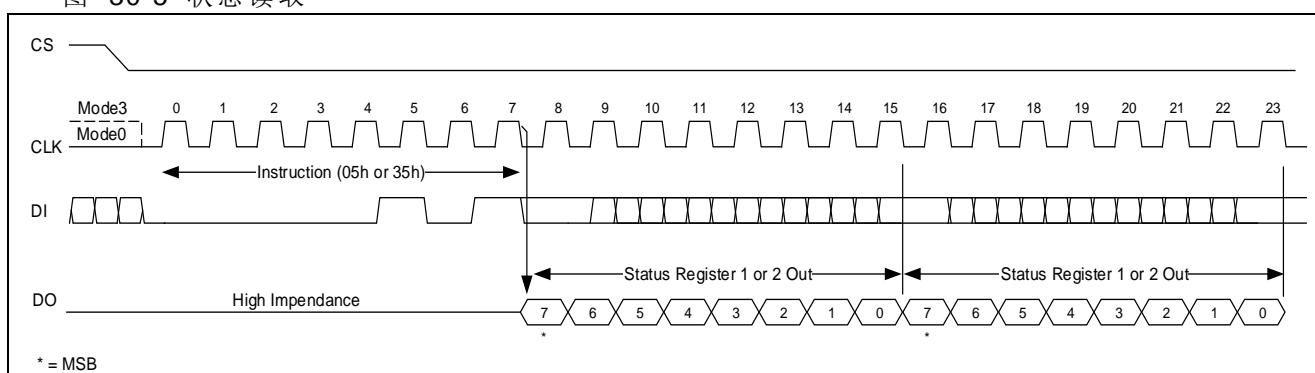
如果用户要执行页面编程命令，请将指令代码设置为 02h，地址寄存器，地址长度，写使能，并将指令长度设置为“1”。

图 30-4 页面编程



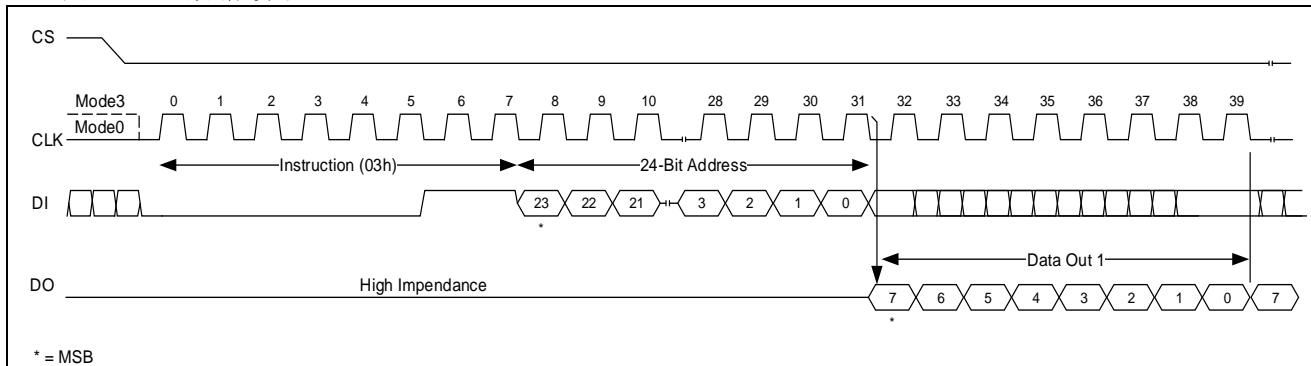
如果用户要执行读取状态命令，请将指令代码设置为 05h / 35h，启用读取状态，通过硬件/软件进行可选的读取状态，并将指令长度设置为“1”。有关更多详细信息，请参见下图。

图 30-5 状态读取



如果用户要执行读取数据命令，请将指令代码/长度设置为 1 个字节，地址/地址长度设置为 3 个字节，并将写入允许设置为“0”。有关更多详细信息，请参见下图。

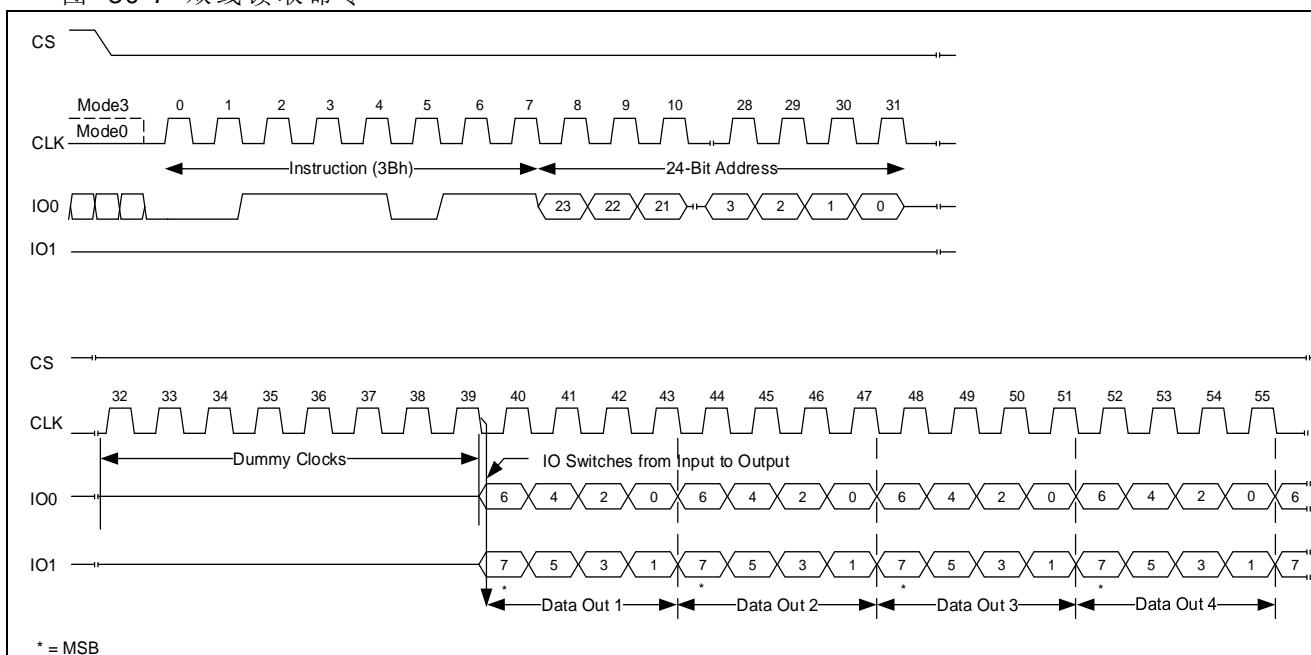
图 30-6 数据读取



双线(1-1-2)模式

如果用户要执行快速读取双线输出命令，请将指令代码/长度设置为“1”，地址/地址长度设置为 3 个字节，第二个虚拟周期为 8，操作模式设置为双线模式。有关更多详细信息，请参见下图。

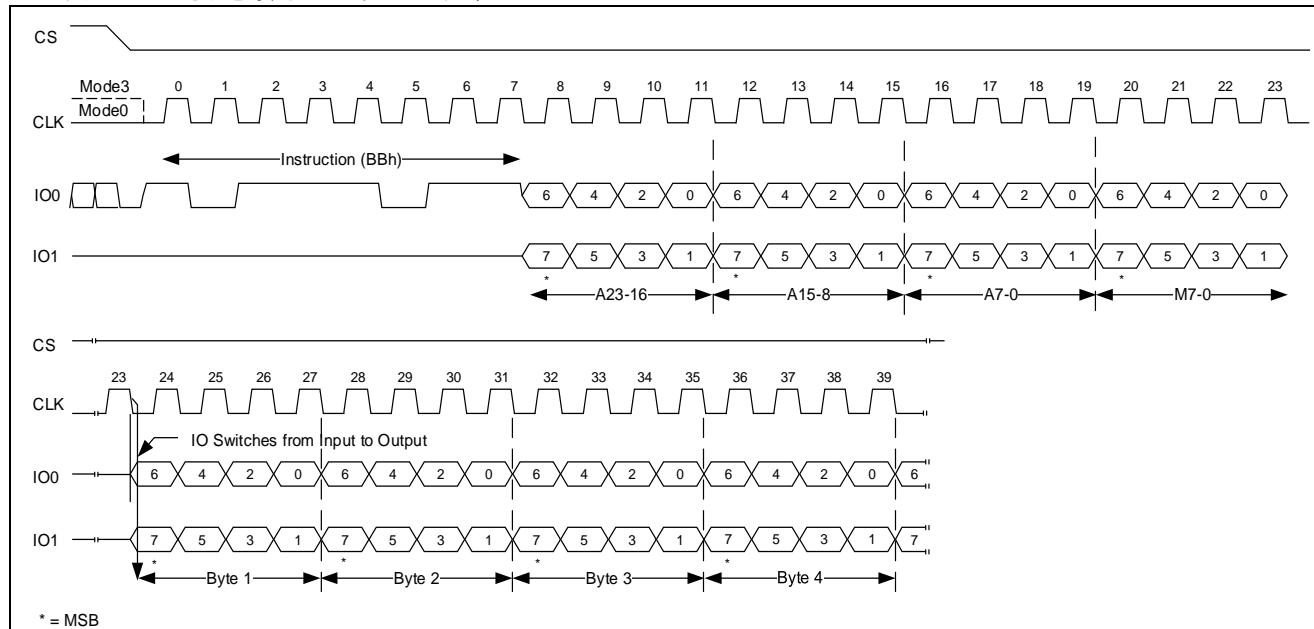
图 30-7 双线读取命令



双线 I/O (1-2-2)模式

如果用户要执行快速读取双线 I/O 命令, 请设置代码/长度, 地址/地址长度, 启用性能增强模式/代码以及对双线 I/O 模式进行操作的指令。有关更多详细信息, 请参见下图。

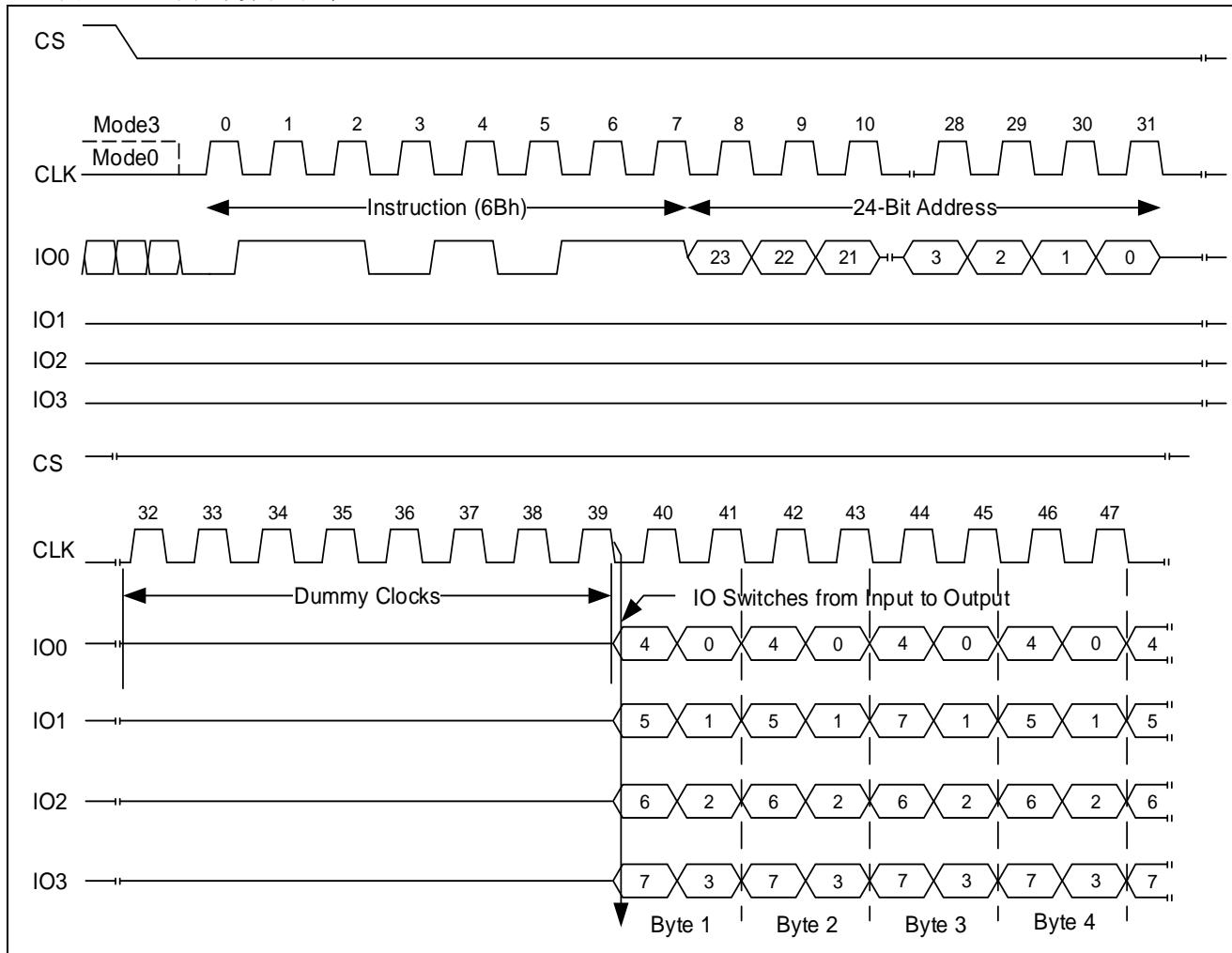
图 30-8 快速读取双线 I/O 命令



四线(1-1-4)模式

如果用户要执行快速读取四线命令，请设置指令代码/长度，地址/地址长度，第二个虚拟周期和操作模式设置为四线模式。请参阅有关更多详细信息，请参见下图。

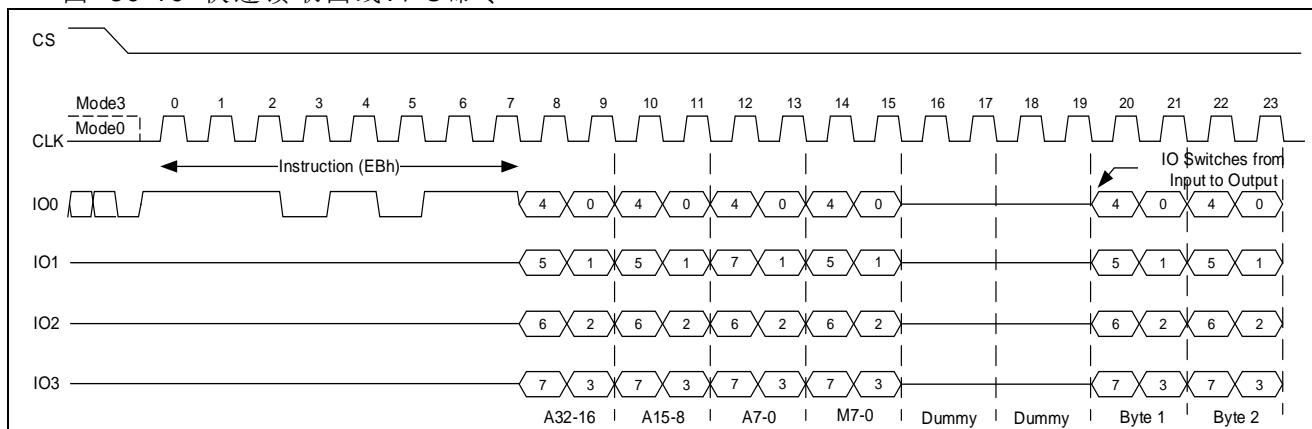
图 30-9 四线读取命令



四线 I/O (1-4-4)模式

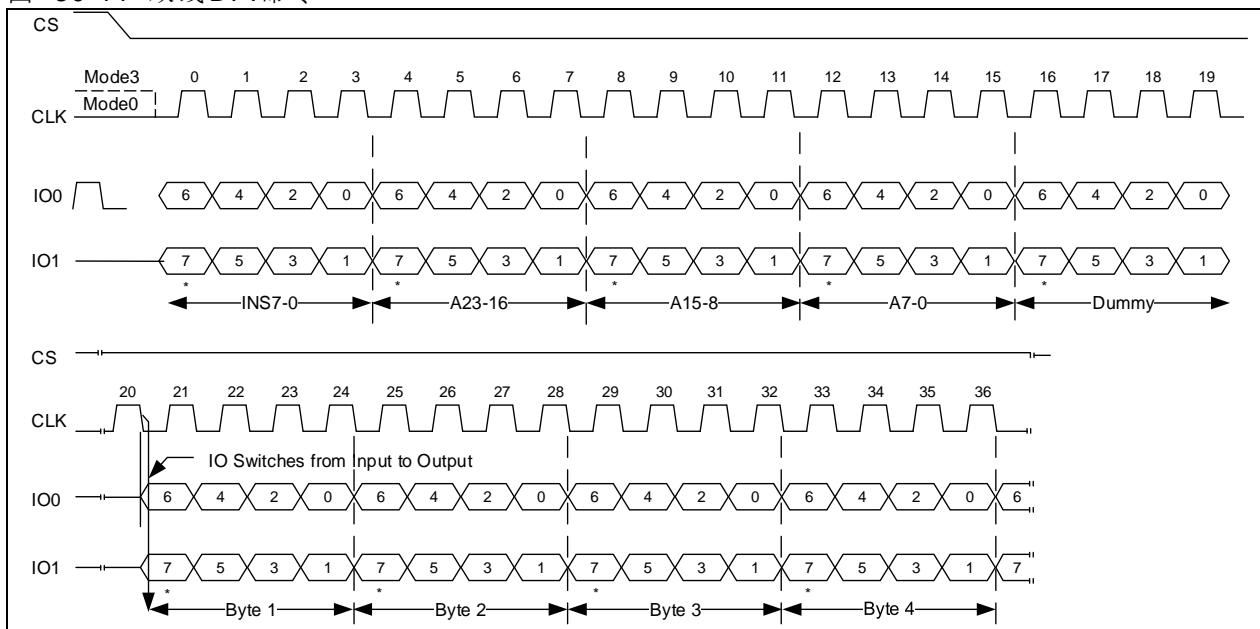
如果用户要执行快速读取四线 I/O 命令，请设置指令代码/长度，地址/地址长度，第二个虚拟周期，性能增强模式/代码和操作模式设置为四线 I/O 模式。有关更多详细信息，请参见下图。

图 30-10 快速读取四线 I/O 命令



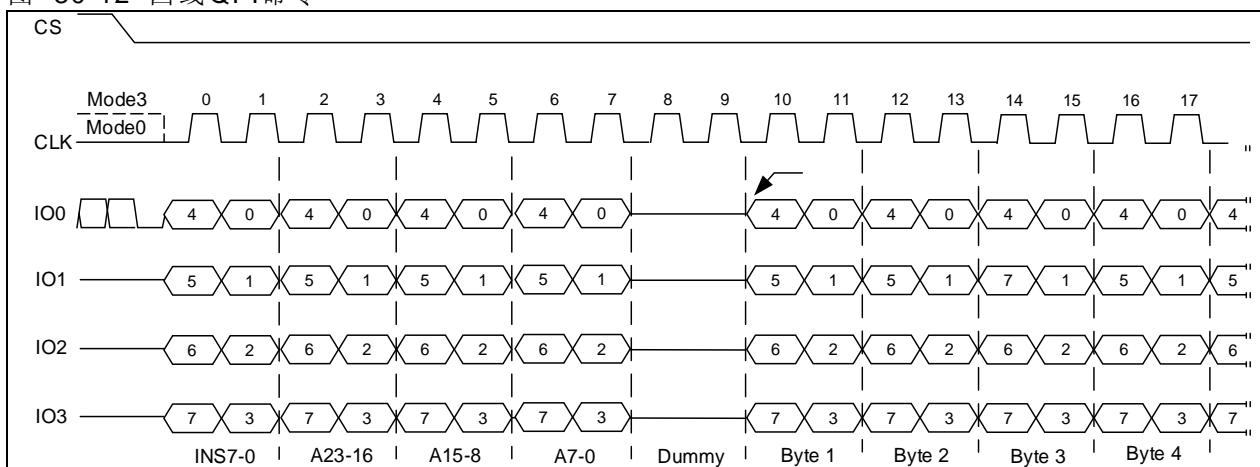
如果用户要执行双线 DPI (2-2-2)模式, 请设置代码/长度, 地址/地址长度, 第二个虚拟周期和双线 DPI 模式进行操作的指令。有关更多详细信息, 请参见下图。

图 30-11 双线 DPI 命令



如果用户要执行四线 QPI (4-4-4)模式, 请设置代码/长度, 地址/地址长度, 第二个虚拟周期和四线 QPI 模式进行操作的指令。有关更多详细信息, 请参见下图。

图 30-12 四线 QPI 命令



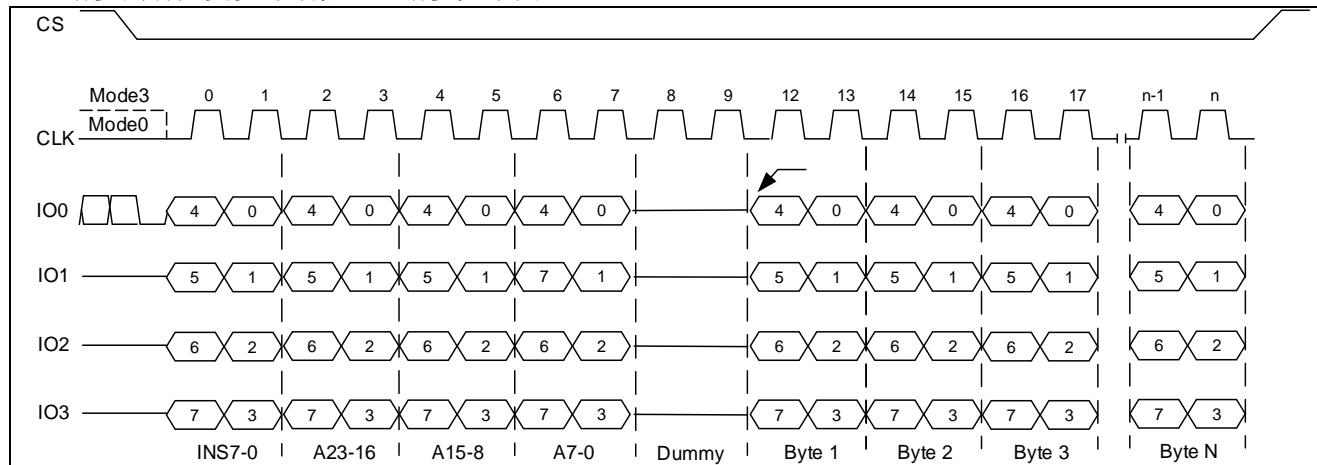
XIP 读取/写入 D 模式

如果用户要执行 XIP 读取/写入 D 模式, 请设置指令代码/长度, 地址/地址长度, 第二个虚拟周期, 选择串行/双线/双线 IO/四线/四线 IO 或 DPI/QPI 模式并设置读取/写入 D 模式和 D 模式数量上限值计数器。

当 XIP 端口读取/写入连续地址时, 会连续读取/写入资料直到数量上限值计数或读取/写入地址不连续。

读取时, $N(\text{byte}) = \text{XIPR_DCNT} * 8$ 。写入时, $N(\text{byte}) = \text{XIPW_DCNT} * 8$ 。

请参阅有关更多详细信息, 请参见下图。



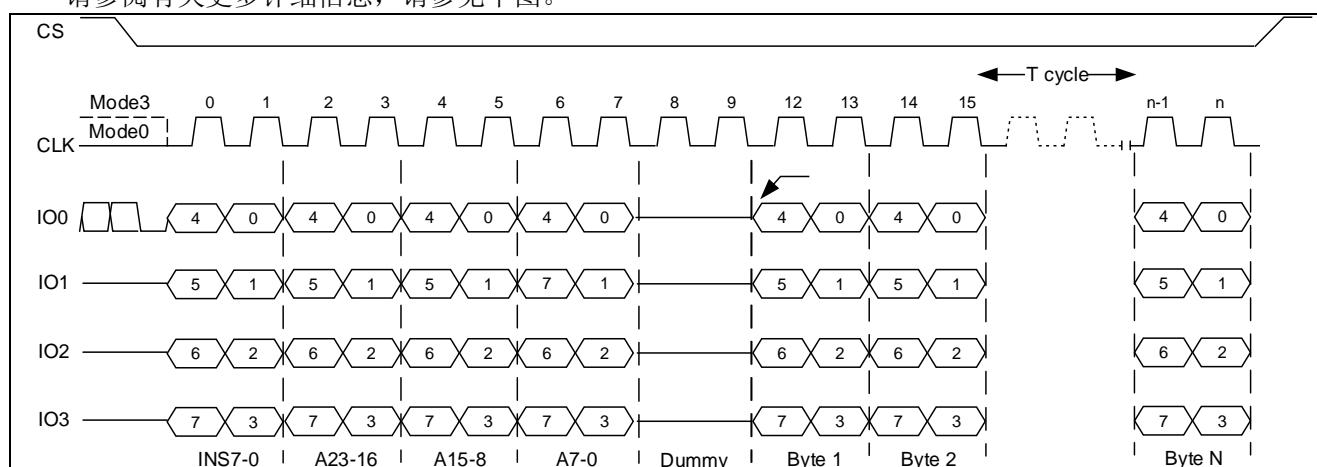
XIP 读取/写入 T 模式

如果用户要执行 XIP 读取/写入 T 模式, 请设置指令代码/长度, 地址/地址长度, 第二个虚拟周期, 选择串行/双线/双线 IO/四线/四线 IO/DPI 或 QPI 模式并设置读取/写入 T 模式和 T 模式时间计数器。

当 XIP 端口间隔时间读取/写入连续地址时, 会在读取/写入资料后关闭 CLK 等待。若等待超过时间上限值计数或下一笔读取 / 写入地址不连续时, 则结束连续读取/写入。

读取时, $T(\text{cycle}) = \text{XIPR_TCNT}$ 。写入时, $T(\text{cycle}) = \text{XIPW_TCNT}$ 。

请参阅有关更多详细信息, 请参见下图。



30.4 QSPI寄存器

可以用字（8位）、半字（16位）或字（32位）的方式操作这些寄存器。

表 30-1 QSPI寄存器映像和复位值

寄存器简称	基址偏移量	复位值
CMD_W0	0x0	0x0000 0000
CMD_W1	0x4	0x0100 0003
CMD_W2	0x8	0x0000 0000
CMD_W3	0xC	0x0000 0000
CTRL	0x10	0x0010 0083
FIFOSTS	0x18	0x0000 0001
CTRL2	0x20	0x0000 0000
CMDSTS	0x24	0x0000 0000
RSTS	0x28	0x0000 0000
FSIZE	0x2C	0xF0000 0000
XIP CMD_W0	0x30	0x0000 3000
XIP CMD_W1	0x34	0x0000 2000
XIP CMD_W2	0x38	0x0F01 0F01
XIP CMD_W3	0x3C	0x0000 0000
CTRL3	0x40	0x0000 0000
REV	0x50	0x0001 0500
DT	0x100	0x0000 0000

30.4.1 命令字0 (CMD_W0)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 0	SPIADR	0x0	rw	SPI 闪存地址 (SPI Flash Address) 该寄存器决定 SPI 闪存地址的值，并将该地址发送给 SPI 闪存。地址的字节由 CMD_W1 寄存器的位 2:0 决定。

30.4.2 命令字1 (CMD_W1)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 29	保留	0x0	resd	保持默认值。
位 28	PEMEN	0x0	rw	性能增强模式使能 (Performance Enhance Mode Enable) 位于地址和第二虚拟状态之间。在性能增强模式下，它可以消除第二条读取命令之后的指令状态。数据计数器 (CMD_W2) 为 0 时，请勿启用此位。 0: 禁用性能增强模式 1: 启用 1 字节性能增强模式
位 27: 26	保留	0x0	resd	保持默认值。

位 25: 24 INSLEN	0x1	rw	指令码长度 (Instruction Code Length) 当用户要执行 SPI Flash 命令时，必须包含指令代码。不同的 SPI Flash 供应商具有不同的指令长度。因此，用户可以设置该寄存器以满足不同的行为。指令代码通常为 1 个字节；但是，如果用户设置 2 字节指令长度，则主机控制器将发出两次此指令代码。 00: 无指令代码。仅在连续读取模式命令完成后才能使用。 01: 1 个字节的指令代码 10: 2 字节指令代码（重复指令代码） 11: 保留
位 23: 16 DUM2	0x0	rw	第二虚拟周期 (Second dummy state cycle) 第二虚拟周期位于地址和数据状态之间，其中不包括性能增强模式状态。用户可以检查 SPI Flash 规范中地址和数据状态之间是否存在虚拟状态。主机控制器将在虚拟周期中发出逻辑 1。 0: 没有第二个虚拟状态 1~32: 1 个虚拟秒周期~32 个虚拟秒周期
位 15: 3 保留	0x0	resd	保持默认值。
位 2: 0 ADRLEN	0x3	rw	SPI 地址长度 (SPI Address Length) 该寄存器决定 SPI Flash 地址的字节数。用户可以设置该寄存器来决定地址字节，范围从 1 字节到 4 字节。 000: 无地址状态 001: 1 个字节的地址 010: 2 字节地址 011: 3 字节地址 100: 4 字节地址 其他: 保留

30.4.3 命令字2 (CMD_W2)

访问: 无等待状态, 字, 半字和字节访问

域	简称	复位值	类型	功能
位 31: 0 DCNT	0x0	rw	读/写数据计数器 (Read/Write Data Counter) 执行读取状态命令时，必须将该寄存器设置为“0”。 0: 无读写数据 1~FFFFFF: 1~FFFFFF 字节数据 请注意，无论是数据读取还是数据写入，都不允许将该寄存器填充为“0”。但是，例如“读取状态”或“写使能”指令，该寄存器必须设置为“0”。	

30.4.4 命令字3 (CMD_W3)

访问: 无等待状态, 字, 半字和字节访问

域	简称	复位值	类型	功能
位 31: 24 INSC	0x00	rw	指令码 (Instruction Code) 用户可以设置此代码以执行 SPI Flash 命令。	
位 23: 16 PEMOPC	0x00	rw	性能增强模式操作代码 (Performance Enhance Mode Operate Code) 需搭配 PEMEN 一起作用，用户可以填充此代码以执行性能增强模式。请参照各家 Flash 的规格文件填写对应的值。	
位 15: 10 保留	0x0	resd	保持默认值。	

				SPI 操作模式 (SPI Operate Mode) 000: 串行 (1-1-1) 模式 001: 双线 (1-1-2) 模式 010: 四线 (1-1-4) 模式 011: 双线 I/O (1-2-2) 模式 100: 四线 I/O (1-4-4) 模式 101: DPI (2-2-2) 模式 110: QPI (4-4-4) 模式 其他: 保留
位 7: 5	OPMODE	0x0	rw	保持默认值。
位 4	保留	0x0	resd	读取 SPI Flash 状态配置 (Read SPI Status Configure) 仅在启用读取状态且写入启用时才可用。 用户必须发出 SPI 读取状态命令。 0: 通过硬件读取状态，控制器将轮询状态直到状态就绪（不忙）并报告状态寄存器。 1: 通过软件读取状态，一次读取状态并报告状态寄存器，直到用户可以读取为止。
位 3	RSTSC	0x0	rw	读取 SPI 状态使能 (Read SPI Status Enable) 它在 WEN = “0” 时可用，且用户必须发出 SPI 读取状态命令。 0: 禁用 1: 启用
位 2	RSTSEN	0x0	rw	写数据使能 (Write Data Enable) 使能 SPI 写数据，但读数据或读状态除外（读数据返回路径）；用户必须为其他 SPI 命令设置写使能=“1”。 请注意，在写入数据或擦除 Flash 命令中，写入启用设置为“1”。仅当在读取数据或读取状态命令中时，必须将写入启用设置为“0”。 0: 禁用 1: 启用
位 1	WEN	0x0	rw	位 0 保留 0x0 resd 保持默认值。
位 0	保留	0x0	resd	读取 SPI Flash 数据，从 0: 命令从端口 1: XIP 端口 当该位被切换时，QSPI 将自动发出中止功能。用户必须等待中止功能完成才能发出命令。

30.4.5 控制寄存器 (CTRL)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 22 保留		0x000	resd	保持默认值。
位 21	KEYEN	0x0	rw	SPI 资料加密钥匙使能 0: 禁用 1: 启用 启用该功能后，QSPIKEY 作为加密算法的秘钥，将原始数据变为密文写入 QSPI 外设，读取时解密成明文取出给到 CPU。
位 20	XIPSEL	0x1	rw	XIP 端口选择 (XIP port selection) 读取 SPI Flash 数据，从 0: 命令从端口 1: XIP 端口 当该位被切换时，QSPI 将自动发出中止功能。用户必须等待中止功能完成才能发出命令。
位 19	XIPRCMDF	0x0	rw	XIP 读取命令刷新 (XIP read command flush)
位 18: 16 BUSY		0x0	rw	SPI 状态的忙位 (Busy bit of the SPI status) 主机轮询此繁忙位，并处于硬件读取状态。 000~111: 位 0~位 7
位 15: 9 保留		0x00	resd	保持默认值。

位 8	ABORT	0x0	rw	刷新所有命令/ FIFO 并重置状态机 发生不正常中止时，用户必须填写该位（该位将自动清除为零）。 0: 无作用。 1: 启用。
位 7	XIPIDLE	0x1	ro	XIP 端口空闲状态 (XIP port idle status) 0: XIP 端口正忙。 1: XIP 端口空闲。
位 6: 5	保留	0x0	resd	保持默认值。
位 4	SCKMODE	0x0	rw	sckout 模式 (sckout mode) 0: 对于模式 0，在空闲状态下 sck_out 为低电平，第一个边沿进行数据捕获。 1: 对于模式 3，在空闲状态下 sck_out 为高电平，第二个边沿进行数据捕获。
位 3: 0	CLKDIV	0x3	Rw	clk 分频器 (clk divider) sck_out 除以 spi_clk，因子列出如下： 0000: 除以 2 0001: 除以 4 0010: 除以 6 0011: 除以 8 0100: 除以 3 0101: 除以 5 0110: 除以 10 0111: 除以 12 1xxx: 除以 1

30.4.6 FIFO状态寄存器 (FIFOSTS)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 2	保留	0x0000 0000	resd	保持默认值。
位 1	RXFIFORDY	0x0	ro	RxFIFO 准备就绪状态 (RxFIFO ready status) 当该位置起时，表明 RxFIFO 已满或 RxFIFO 中的剩余数据小于 RxFIFO 深度，但这是最后一笔数据
位 0	TXFIFORDY	0x1	ro	TxFIFO 准备就绪状态 (TxFIFO ready status) 当该位置起时，表明 TxFIFO 将为空，用户可以将数据传输到 TxFIFO 中，直到数据满为止。

30.4.7 控制寄存器2 (CTRL2)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 14 保留		0x0000 0	resd	保持默认值。
位 13..12 RXFIFO THOD		0x0	rw	该信号用于为 DMA 握手模式设置 RxFIFO 阈值中断的触发电平。单位是 WORD。 触发电平值是 RxFIFO 中的数据。 00: 8 WORD 01: 16 WORD 10: 24 WORD 11: 保留
位 11: 10 保留		0x0	resd	保持默认值。
位 9: 8 TXFIFO THOD		0x0	rw	该信号用于为 DMA 握手模式设置 TxFIFO 阈值中断的触发电平。单位是 WORD。 触发级别值是 TxFIFO 中的可用数据条目。 00: 8 WORD 01: 16 WORD 10: 24 WORD 11: 保留
位 7: 2 保留		0x00	resd	保持默认值。
位 1 CMDIE		0x0	rw	命令完成中断使能 (Command complete Interrupt Enable) 0: 无中断 1: 启用命令完成中断
位 0 DMAEN		0x0	rw	DMA 使能 (DMA Enable) 注意：从基于命令的从端口切换到 XIP 端口之前，请先禁用此位。

30.4.8 命令状态寄存器 (CMDSTS)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 1 保留		0x0000 0000	resd	保持默认值。
位 0 CMDSTS		0x0	rw1c	命令完成状态 (Command complete status) 在命令完成时设置。

30.4.9 读取状态寄存器 (RSTS)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 1 保留		0x0000 00	resd	保持默认值。
位 7: 0 SPISTS		0x00	ro	SPI 读取状态 (SPI Read status) 主机发出读取 SPI Flash 状态命令并将返回数据存储在此寄存器中。用户可以读取该寄存器以检查 SPI 闪存状态。

30.4.10 闪存大小寄存器 (FSIZE)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 0 SPIFSIZE		0xF000 0000	rw	SPI 闪存大小 (SPI Flash Size) 对于直接地址映射功能，系统地址始终大于 SPI Flash 的地址。用户需要屏蔽系统地址的高位以适应 SPI Flash 的大小。

30.4.11 XIP命令字0 (XIP CMD_W0)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 20 保留		0x000	resd	保持默认值。
位 19: 12 XIPR_INSC		0x03	rw	XIP 读取指令代码 (XIP Read Instruction Code) 用户可以设置此代码来执行 SPI Flash 命令以进行 XIP 读取。
位 11 XIPR_ADRLEN		0x0	rw	XIP 读取 SPI 地址长度 (XIP Read Address Length) 该寄存器决定 SPI Flash 地址的字节数。用户可以设置该寄存器来决定 XIP 读取的 3 字节或 4 字节地址。 0: 3 字节地址 1: 4 字节地址
位 10: 8 XIPR_OPMODE		0x0	rw	XIP 读取操作模式 (XIP Read Operate Mode) 000: 串行 (1-1-1) 模式 001: 双线 (1-1-2) 模式 010: 四线 (1-1-4) 模式 011: 双线 IO (1-2-2) 模式 100: 四线 IO (1-4-4) 模式 101: DPI (2-2-2) 模式 110: QPI (4-4-4) 模式 111: 保留
位 7: 0 XIPR_DUM2		0x00	rw	XIP 读取的第二个虚拟周期 (XIP Read second dummy cycle) 第二虚拟状态位于地址和数据状态之间，其中不包括连续读取模式状态。用户可以检查 SPI Flash 规范中地址和数据状态之间是否存在虚拟状态。主机控制器将在虚拟周期中发出逻辑 1。 0: 没有第二个虚拟状态 1~32: 1 个虚拟秒周期~32 个虚拟秒周期

30.4.12 XIP命令字1 (XIP CMD_W1)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 18 保留		0x000	resd	保持默认值。
位 19: 12 XIPW_INSC		0x02	rw	XIP 写入指令代码 (XIP Write Instruction Code) 用户可以设置此代码来执行 SPI Flash 命令以进行 XIP 写入。
位 11 XIPW_ADRLEN		0x0	rw	XIP 写入 SPI 地址长度 (XIP Write Address Length) 该寄存器决定 SPI Flash 地址的字节数。用户可以设置该寄存器来决定 XIP 写入的 3 字节或 4 字节地址。 0: 3 字节地址 1: 4 字节地址
位 10: 8 XIPW_OPMODE		0x0	rw	XIP 写入操作模式 (XIP Write Operate Mode) 000: 串行 (1-1-1) 模式 001: 双线 (1-1-2) 模式 010: 四线 (1-1-4) 模式 011: 双线 IO (1-2-2) 模式 100: 四线 IO (1-4-4) 模式 101: DPI (2-2-2) 模式 110: QPI (4-4-4) 模式 111: 保留

位 7: 0	XIPW_DUM2	0x00	rw	XIP 写入的第二个虚拟状态周期 (XIP Write second dummy cycle) 第二伪状态位于地址和数据状态之间。用户可以检查 SPI Flash 规范中地址和数据状态之间是否存在伪状态。主机控制器将在虚拟周期中发出逻辑 1。 0: 没有第二个虚拟状态 1~32: 1 个虚拟秒周期~32 个虚拟秒周期
--------	-----------	------	----	--

30.4.13 XIP命令字2 (XIP_CMD_W2)

访问: 无等待状态, 字, 半字和字节访问

域	简称	复位值	类型	功能
位 31	XIPW_SEL	0x0	rw	XIP 写入模式选择 0: 选择 模式 D 1: 选择 模式 T XIP 从端口可以优化实际读写的流程, 增加效能。 模式 D: 发生连续地址的写入数据时, 限制单次写入的数量上限值 (DCNT)。 模式 T: 如果连续两笔的写入数据, 地址也连续, 其间隔小于特定的时间 (TCNT), 则合并成一个命令处理。
位 30: 24	XIPW_TCNT	0x0F	rw	选择模式 T 时, 判断间隔时使用的时间计数器 单位是 sck_out 周期。 当 XIPW_SEL 启用模式 T 时, 该时间计数器有效。
位 23: 21	保留	0x0	resd	保持默认值。
位 20: 16	XIPW_DCNT	0x01	rw	选择模式 D 时, 判断数量上限值时使用的计数器 单位是字, 且不能为零 当 XIPW_SEL 启用模式 D, 该数据计数器有效。
位 15	XIPR_SEL	0x0	rw	XIP 读取模式选择 0: 选择 模式 D 1: 选择 模式 T XIP 从端口可以优化实际读写的流程, 增加效能。 模式 D: 发生连续地址的读取数据时, 限制单次读取的数量上限值 (DCNT)。 模式 T: 如果连续两笔的读取数据, 地址也连续, 其间隔小于特定的时间 (TCNT), 则合并成一个命令处理。
位 14..8	XIPR_TCNT	0x0F	rw	选择模式 T 时, 判断间隔时使用的时间计数器 单位是 sck_out 周期。 当 XIPR_SEL 启用模式 T 时, 该时间计数器有效。
位 7: 5	保留	0x0	resd	保持默认值。
位 4: 0	XIPR_DCNT	0x01	rw	选择模式 D 时, 判断数量上限值时使用的计数器 单位是字, 且不能为零 当 XIPR_SEL 启用模式 D, 该数据计数器有效。

30.4.14 XIP命令字3 (XIP CMD_W3)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 4	保留	0x0000 000	resd	保持默认值。
位 3	CSTS	0x0	ro	快取状态 (Cache Status) 0: 缓存验证 1: 缓存失效。
位 2: 1	保留	0x0	resd	保持默认值。
位 0	BYPASSC	0x0	rw	旁路缓存功能 (Bypass Cache Function) 将该位置 1 时，将使高速缓存功能无效，并且所有读取传输都将不查找高速缓存。 缓存功能仅适用于 XIP Read only 应用 (扩展外部 FLASH)，对于 XIP Read/Write 应用 (扩展外部 PSRAM)，必须将该位置 1

30.4.15 控制寄存器 (CTRL3)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	功能
位 31: 9	保留	0x00 00	保持默认值。
位 8	ISPC	0x0	输入采样相位校正使能 (Input sampling phase correction enable) 匹配 SPI Flash 输出时序，依据输入采样相位延迟 (ISPD) 调整采样相位 0: 固定相位 1: 启用相位调整
位 7: 6	保留	0x0	保持默认值。
位 5: 0	ISPD	0x00	输入采样相位延迟 (Input sampling phase delay) 当 ISPC 启用时，该输入采样相位延迟设定有效。

30.4.16 修订寄存器 (REV)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 0	REV	0x0001 0500	ro	表示此 IP 版本

30.4.17 数据端口寄存器 (DT)

访问：无等待状态，字，半字和字节访问

域	简称	复位值	类型	功能
位 31: 0	DT	0x0000 0000	rw	数据端口寄存器 用户可以从数据端口读取/写入数据。

31 调试 (DEBUG)

31.1 简介

Cortex®-M4F 内核具有丰富的调试特性。除了支持暂停和单步等标准的调试特性外，还可以利用跟踪特性查看程序执行的细节。Cortex®-M4F 内核的调试可以通过两种接口实现：串行调试接口。

ARM Cortex®-M4F 内核相关资料，可参考：

- Cortex®-M4 技术参考手册 (TRM)
- ARM 调试接口 V5
- ARM CoreSight 开发工具集 (r1p0 版) 技术参考手册

31.2 调试与跟踪功能

支持不同外设的调试，还可以设置调试时外设的工作状态。对于定时器和看门狗用户可以选择在调试时是否停止或继续计数；对于 CAN，用户可以选择在调试期间是否停止或继续更新接收寄存器；对于 I²C，用户可以选择在调试期间是否停止或继续 SMBUS 超时计数。

另外支持在低功耗模式下调试代码。在睡眠模式下，HCLK 与 FCLK 保持代码配置的时钟继续工作。在深度休眠模式下，HICK 振荡器将开启并为 FCLK 和 HCLK 提供时钟。

MCU 内部有多个 ID 编码，调试器可通过地址为 0xE0042000 的 DEBUG_IDCODE 来访问。它是 DEBUG 的一个组成部分，并且映射到外部 PPB 总线上。使用 SW 调试口或通过用户代码都可以访问此编码。即使当 MCU 处于系统复位状态下这个编码也可以被访问。

31.3 I/O 控制

AT32F405 在所有的封装里都支持 SWJ-DP 调试，该调试共使用 5 个普通 I/O 口。复位以后，SWJ-DP 作为默认功能可立即供调试器使用。

当用户切换调试接口或不使用调试功能时，可配置 GPIO 和 IOMUX 寄存器来切换这些 I/O 口功能。

31.4 DEBUG 寄存器

下面列出了 DEBUG 寄存器映象和复位数值。

必须以字（32 位）的方式操作这些外设寄存器。

表 31-1 DEBUG 寄存器地址和复位值

寄存器简称	基地址	复位值
DEBUG_IDCODE	0xE004 2000	0xXXXX XXXX
DEBUG_CTRL	0xE004 2004	0x0000 0000
DEBUG_APB1_PAUSE	0xE004 2008	0x0000 0000
DEBUG_APB2_PAUSE	0xE004 200C	0x0000 0000
DEBUG_APB3_PAUSE	0xE004 2010	0x0000 0000
DEBUG_SER_ID	0xE004 2020	0x0000 XX0X

31.4.1 DEBUG设备ID (DEBUG_IDCODE)

MCU 集成了 ID code, 通过 ID 可以识别 MCU 的版本编号。DEBUG_IDCODE 寄存器被映射到外部 PPB 总线, 基地址为 0xE0042000。使用 SW 调试口或用户代码都可以访问此编号。

域	简称	复位值	类型	功能
位 31: 0	PID	0xFFFF XXXX	ro	PID 信息

PID [31: 0]	AT32 型号	FLASH 大小	封装
0x7006_32C0	AT32F455ZET7	512KB	144LQFP
0x7005_3241	AT32F455ZCT7	256KB	144LQFP
0x7006_32C2	AT32F455VET7	512KB	100LQFP
0x7005_3243	AT32F455VCT7	256KB	100LQFP
0x7006_32C4	AT32F455RET7	512KB	64LQFP
0x7005_3245	AT32F455RCT7	256KB	64LQFP
0x7006_32C6	AT32F455CET7	512KB	48LQFP
0x7005_3247	AT32F455CCT7	256KB	48LQFP
0x7006_32C8	AT32F455CEU7	512KB	48QFN
0x7005_3249	AT32F455CCU7	256KB	48QFN
0x7006_32CA	AT32F456ZET7	512KB	144LQFP
0x7005_324B	AT32F456ZCT7	256KB	144LQFP
0x7006_32CC	AT32F456VET7	512KB	100LQFP
0x7005_324D	AT32F456VCT7	256KB	100LQFP
0x7006_32CE	AT32F456RET7	512KB	64LQFP
0x7005_324F	AT32F456RCT7	256KB	64LQFP
0x7006_32D0	AT32F456CET7	512KB	48LQFP
0x7005_3251	AT32F456CCT7	256KB	48LQFP
0x7006_32D2	AT32F456CEU7	512KB	48QFN
0x7005_3253	AT32F456CCU7	256KB	48QFN
0x7006_32D4	AT32F457ZET7	512KB	144LQFP
0x7005_3255	AT32F457ZCT7	256KB	144LQFP
0x7006_32D6	AT32F457VET7	512KB	100LQFP
0x7005_3257	AT32F457VCT7	256KB	100LQFP
0x7006_32D8	AT32F457RET7	512KB	64LQFP
0x7005_3259	AT32F457RCT7	256KB	64LQFP

31.4.2 DEBUG控制寄存器 (DEBUG_CTRL)

寄存器由 PORESET 异步复位 (不被系统复位所复位)。当内核处于复位状态下时，调试器可写。

域	简称	复位值	类型	功能
位 31: 3	保留	0x0000 0000	resd	必须保持为 0。
位 2	STANDBY_DEBUG	0x0	rw	待机模式调试控制位。 0: 进入待机模式时，整个 1.2V 数字电路部分都断电； 1: 进入待机模式时，整个 1.2V 数字电路部分不断电，系统时钟由内部 RC 振荡器 (HICK) 提供时钟。
位 1	DEEPSLEEP_DEBUG	0x0	rw	深度睡眠模式调试控制位。 0: 进入深度睡眠模式时，关闭所有 1.2V 域的时钟，退出深度睡眠模式时，系统时钟选择开启内部 RC 振荡器 (HICK)，系统时钟选择 HICK 作为系统时钟源，软件需根据应用需求重新配置系统时钟； 1: 进入深度睡眠模式时，系统时钟由内部 RC 振荡器 (HICK) 提供。退出深度睡眠模式时，系统时钟选择 HICK 作为系统时钟源，软件需根据应用需求重新配置系统时钟。
位 0	SLEEP_DEBUG	0x0	rw	睡眠模式调试控制位 0: 进入睡眠模式时，CPU HCLK 时钟关闭，其他时钟均继续运行，退出睡眠模式时，不需要重新配置时钟系统； 1: 进入睡眠模式时，所有时钟都继续运行。

31.4.3 DEBUG APB1暂停控制寄存器 (DEBUG_APB1_PAUSE)

寄存器由 PORESET 异步复位 (不被系统复位所复位)。当内核处于复位状态下时，调试器可写。

域	简称	复位值	类型	功能
位 31: 29	保留	0x0	resd	请保持为复位值。
位 28	I2C3_SMBUS_TIMEOUT	0x0	rw	I ² C3 暂停控制位。 0: 正常工作； 1: I ² C3 SMBUS 的超时控制停止工作。
位 27	I2C2_SMBUS_TIMEOUT	0x0	rw	I ² C2 暂停控制位。 0: 正常工作； 1: I ² C2 SMBUS 的超时控制停止工作。
位 26	保留	0x0	rw	请保持为复位值。
位 25	保留	0x00	resd	请保持为复位值。
位 24	I2C1_SMBUS_TIMEOUT	0x0	rw	I ² C1 暂停控制位。 0: 正常工作； 1: I ² C1 SMBUS 的超时控制停止工作。
位 23: 13	保留	0x0	rw	请保持为复位值。
位 12	WDT_PAUSE	0x0	rw	看门狗暂停控制位 0: 看门狗正常工作； 1: 看门狗停止工作。
位 11	WWDT_PAUSE	0x0	rw	窗口看门狗暂停控制位。 0: 窗口看门狗正常工作； 1: 窗口看门狗停止工作。
位 10: 9	保留	0x0	rw	请保持为复位值。
位 8	TMR14_PAUSE	0x0	rw	TMR14 暂停控制位。 0: 定时器正常工作； 1: 定时器停止工作。

位 7	TMR13_PAUSE	0x0	rw	TMR13 暂停控制位。 0: 定时器正常工作; 1: 定时器停止工作。
位 6	TMR12_PAUSE	0x0	rw	TMR12 暂停控制位。 0: 定时器正常工作; 1: 定时器停止工作。
位 5	TMR7_PAUSE	0x0	rw	TMR7 暂停控制位。 0: 定时器正常工作; 1: 定时器停止工作。
位 4	TMR6_PAUSE	0x0	rw	TMR6 暂停控制位。 0: 定时器正常工作; 1: 定时器停止工作。
位 3	TMR5_PAUSE	0x0	rw	TMR5 暂停控制位。 0: 定时器正常工作; 1: 定时器停止工作
位 2	TMR4_PAUSE	0x0	rw	TMR4 暂停控制位。 0: 定时器正常工作; 1: 定时器停止工作。
位 1	TMR3_PAUSE	0x0	rw	TMR3 暂停控制位。 0: 定时器正常工作; 1: 定时器停止工作。
位 0	TMR2_PAUSE	0x0	rw	TMR2 暂停控制位。 0: 定时器正常工作; 1: 定时器停止工作。

31.4.4 DEBUG APB2暂停控制寄存器 (DEBUG_APB2_PAUSE)

寄存器由 PORESET 异步复位 (不被系统复位所复位)。当内核处于复位状态下时，调试器可写。

域	简称	复位值	类型	功能
位 31: 19	保留	0x0000	resd	请保持为复位值。
位 18	TMR11_PAUSE	0x0	rw	TMR11 暂停控制位 0: 定时器正常工作; 1: 定时器停止工作。
位 17	TMR10_PAUSE	0x0	rw	TMR10 暂停控制位 0: 定时器正常工作; 1: 定时器停止工作。
位 16	TMR9_PAUSE	00	rw	TMR9 暂停控制位 0: 定时器正常工作; 1: 定时器停止工作。
位 15: 2	保留	0x0000	resd	请保持为复位值。
位 1	TMR8_PAUSE	0x0	rw	TMR8 暂停控制位。 0: 定时器正常工作; 1: 定时器停止工作。
位 0	TMR1_PAUSE	0x0	rw	TMR1 暂停控制位。 0: 定时器正常工作; 1: 定时器停止工作。

31.4.5 DEBUG APB3暂停控制寄存器 (DEBUG_APB3_PAUSE)

寄存器由 PORESET 异步复位 (不被系统复位所复位)。当内核处于复位状态下时，调试器可写。

域	简称	复位值	类型	功能
位 31: 1	保留	0x0000 0000	resd	请保持为复位值。
位 0	ERTC_PAUSE	0x0	rw	ERTC 暂停控制位。 0: ERTC 正常运行; 1: ERTC 停止工作。

31.4.6 DEBUG SERIES ID寄存器 (DEBUG_SER_ID)

DEBUG_SER_ID 寄存器用于识别 MCU 系列和版本号。被映射到外部 PPB 总线，寄存器由 PORESET 异步复位 (不被系统复位所复位)。使用 SW 调试口或用户代码都可以访问此编号。

域	简称	复位值	类型	功能
位 31: 16	保留	0x0000	resd	请保持为复位值。
位 15: 8	SER_ID	0xXX	ro	MCU 型号标识 AT32F455: 0x15 AT32F456: 0x16 AT32F457: 0x17
位 7: 3	保留	0x0X	resd	请保持为复位值。
位 2: 0	REV_ID	0xX	ro	版本标识 0x0: A 版

32 版本历史

日期	版本	变更
2024.08.30	0.00	最初版本
2024.12.16	2.00	初版发布

重要通知 - 请仔细阅读

买方自行负责对本文所述雅特力产品和服务的选择和使用，雅特力概不承担与选择或使用本文所述雅特力产品和服务相关的任何责任。

无论之前是否有过任何形式的表示，本文档不以任何方式对任何知识产权进行任何明示或默示的授权或许可。如果本文档任何部分涉及任何第三方产品或服务，不应被视为雅特力授权使用此类第三方产品或服务，或许可其中的任何知识产权，或者被视为涉及以任何方式使用任何此类第三方产品或服务或其中任何知识产权的保证。

除非在雅特力的销售条款中另有说明，否则，雅特力对雅特力产品的使用和 / 或销售不做任何明示或默示的保证，包括但不限于有关适销性、适合特定用途（及其依据任何司法管辖区的法律的对应情况），或侵犯任何专利、版权或其他知识产权的默示保证。

雅特力产品并非设计或专门用于下列用途的产品：(A) 对安全性有特别要求的应用，例如：生命支持、主动植入设备或对产品功能安全有要求的系统；(B) 航空应用；(C) 航天应用或航天环境；(D) 武器，且/或(E) 其他可能导致人身伤害、死亡及财产损害的应用。如果采购商擅自将其用于前述应用，即使采购商向雅特力发出了书面通知，风险及法律责任仍将由采购商单独承担，且采购商应独立负责在前述应用中满足所有法律和法规要求。

经销的雅特力产品如有不同于本文档中提出的声明和 / 或技术特点的规定，将立即导致雅特力针对本文所述雅特力产品或服务授予的任何保证失效，并且不应以任何形式造成或扩大雅特力的任何责任。

© 2024 雅特力科技 保留所有权利