# GigaDevice Semiconductor Inc.

# GD32A490
# Arm® Cortex®-M4 32-bit MCU

# 用户手册

1.6 版本

(2026 年 7 月)

# 目 录

目 录 ..... 1
图索引 ..... 23
表索引 ..... 31
1. 系统及存储器架构 ..... 35
1.1. Arm® Cortex®-M4 处理器 ..... 35
1.2. 系统架构 ..... 36
1.3. 存储器映射 ..... 39
1.3.1. 位带操作 ..... 42
1.3.2. 片上 SRAM 存储器 ..... 43
1.3.3. 片上 FLASH 存储器概述 ..... 43
1.4. 引导配置 ..... 43
1.5. 系统配置寄存器(SYSCFG) ..... 45
1.5.1. 配置寄存器 0(SYSCFG_CFG0) ..... 45
1.5.2. 配置寄存器 1(SYSCFG_CFG1) ..... 46
1.5.3. EXTI 源选择寄存器 0(SYSCFG_EXTISS0) ..... 47
1.5.4. EXTI 源选择寄存器 1(SYSCFG_EXTISS1) ..... 48
1.5.5. EXTI 源选择寄存器 2(SYSCFG_EXTISS2) ..... 49
1.5.6. EXTI 源选择寄存器 3(SYSCFG_EXTISS3) ..... 51
1.5.7. I/O 补偿控制寄存器(SYSCFG_CPSCTL) ..... 52
1.6. 设备电子签名 ..... 53
1.6.1. 存储容量信息 ..... 53
1.6.2. 设备唯一 ID(96 位/位域) ..... 53
2. 闪存控制器(FMC) ..... 55
2.1. 简介 ..... 55
2.2. 主要特征 ..... 55
2.3. 功能说明 ..... 55
2.3.1. 闪存结构 ..... 55

2.3.2. 读操作 ..... 58
2.3.3. FMC_CTL/FMC_OBCTLx 寄存器解锁 ..... 58
2.3.4. 页擦除 ..... 58
2.3.5. 扇区擦除 ..... 59
2.3.6. 整片擦除 ..... 60
2.3.7. 主存储闪存块编程 ..... 61
2.3.8. OTP 闪存块编程 ..... 63
2.3.9. 选项字节修改 ..... 63
2.3.10. 选项字节说明 ..... 63
2.3.11. 扇区擦除/编程保护 ..... 65
2.3.12. DBUS 读保护 ..... 65
2.3.13. 安全保护 ..... 66

**2.4. FMC 寄存器 ..... 67**

2.4.1. 等待状态寄存器 (FMC_WS) ..... 67
2.4.2. 解锁寄存器 (FMC_KEY) ..... 67
2.4.3. 选项字节解锁寄存器 (FMC_OBKEY) ..... 68
2.4.4. 状态寄存器 (FMC_STAT) ..... 68
2.4.5. 控制寄存器 (FMC_CTL) ..... 69
2.4.6. 选项字节控制寄存器 0 (FMC_OBCTL0) ..... 71
2.4.7. 选项字节控制寄存器 1 (FMC_OBCTL1) ..... 72
2.4.8. 页擦除配置寄存器 (FMC_PECFG) ..... 73
2.4.9. 页擦除功能解锁寄存器 (FMC_PEKEY) ..... 73
2.4.10. 等待状态使能寄存器 (FMC_WSEN) ..... 73
2.4.11. 产品 ID 寄存器 (FMC_PID) ..... 74

**3. 电源管理单元(PMU) ..... 75**

**3.1. 简介 ..... 75**

**3.2. 主要特征 ..... 75**

**3.3. 功能说明 ..... 75**

3.3.1. 电池备份域 ..... 76
3.3.2. 备份 SRAM ..... 77
3.3.3. $V_{DD}$ / $V_{DDA}$ 电源域 ..... 77
3.3.4. $V_{CORE}$ 电源域 ..... 79
3.3.5. 省电模式 ..... 80

**3.4. PMU 寄存器 ..... 83**

3.4.1. 控制寄存器(PMU_CTL) ..... 83
3.4.2. 电源控制和状态寄存器(PMU_CS) ..... 85

**4. 复位和时钟单元(RCU) ..... 87**

**4.1. 复位控制单元(RCTL) ..... 87**
4.1.1. 简介 ..... 87
4.1.2. 功能说明 ..... 87
**4.2. 时钟控制单元(CCTL) ..... 88**
4.2.1. 简介 ..... 88
4.2.2. 主要特性 ..... 90
4.2.3. 功能说明 ..... 90
**4.3. RCU 寄存器 ..... 95**
4.3.1. 控制寄存器(RCU_CTL) ..... 95
4.3.2. PLL 寄存器(RCU_PLL) ..... 97
4.3.3. 时钟配置寄存器 0(RCU_CFG0) ..... 99
4.3.4. 时钟中断寄存器(RCU_INT) ..... 101
4.3.5. AHB1 复位寄存器(RCU_AHB1RST) ..... 104
4.3.6. AHB2 复位寄存器(RCU_AHB2RST) ..... 106
4.3.7. AHB3 复位寄存器(RCU_AHB3RST) ..... 107
4.3.8. APB1 复位寄存器(RCU_APB1RST) ..... 107
4.3.9. APB2 复位寄存器(RCU_APB2RST) ..... 110
4.3.10. AHB1 使能寄存器(RCU_AHB1EN) ..... 112
4.3.11. AHB2 使能寄存器(RCU_AHB2EN) ..... 115
4.3.12. AHB3 使能寄存器(RCU_AHB3EN) ..... 116
4.3.13. APB1 使能寄存器(RCU_APB1EN) ..... 116
4.3.14. APB2 使能寄存器(RCU_APB2EN) ..... 119
4.3.15. AHB1 睡眠模式使能寄存器(RCU_AHB1SPEN) ..... 122
4.3.16. AHB2 睡眠模式使能寄存器(RCU_AHB2SPEN) ..... 125
4.3.17. AHB3 睡眠模式使能寄存器(RCU_AHB3SPEN) ..... 125
4.3.18. APB1 睡眠模式使能寄存器(RCU_APB1SPEN) ..... 126
4.3.19. APB2 睡眠模式使能寄存器(RCU_APB2SPEN) ..... 129
4.3.20. 备份域控制寄存器(RCU_BDCTL) ..... 131
4.3.21. 复位源/时钟寄存器(RCU_RSTSCK) ..... 133
4.3.22. PLL 时钟扩频控制寄存器(RCU_PLLSSCTL) ..... 134
4.3.23. PLLI2S 寄存器(RCU_PLLI2S) ..... 135
4.3.24. PLLSAI 寄存器(RCU_PLLSAI) ..... 136
4.3.25. 时钟配置寄存器 1(RCU_CFG1) ..... 138
4.3.26. 附加时钟控制寄存器(RCU_ADDCTL) ..... 139
4.3.27. 附加时钟中断寄存器(RCU_ADDINT) ..... 140
4.3.28. APB1 附加复位寄存器(RCU_ADDAPB1RST) ..... 141
4.3.29. APB1 附加使能寄存器(RCU_ADDAPB1EN) ..... 141
4.3.30. APB1 附加睡眠模式使能寄存器(RCU_ADDAPB1SPEN) ..... 142
4.3.31. 电源解锁寄存器(RCU_VKEY) ..... 142
4.3.32. 深度睡眠模式电压寄存器(RCU_DSV) ..... 143
**5. 时钟校准控制器(CTC) ..... 144**

5.1. 简介 ..... 144
5.2. 主要特性 ..... 144
5.3. 功能说明 ..... 144
5.3.1. REF 同步脉冲发生器 ..... 145
5.3.2. CTC 校准计数器 ..... 145
5.3.3. 频率评估和自动校准过程 ..... 146
5.3.4. 软件编程指南 ..... 147
5.4. CTC 寄存器 ..... 148
5.4.1. 控制寄存器 0(CTC_CTL0) ..... 148
5.4.2. 控制寄存器 1(CTC_CTL1) ..... 149
5.4.3. 状态寄存器(CTC_STAT) ..... 150
5.4.4. 中断清除寄存器(CTC_INTC) ..... 152
6. 中断/事件控制器(EXTI) ..... 153
6.1. 简介 ..... 153
6.2. 主要特征 ..... 153
6.3. 中断功能描述 ..... 153
6.4. 结构框图 ..... 157
6.5. 外部中断及事件功能概述 ..... 157
6.6. EXTI 寄存器 ..... 160
6.6.1. 中断使能寄存器(EXTI_INTEN) ..... 160
6.6.2. 事件使能寄存器(EXTI_EVEN) ..... 160
6.6.3. 上升沿触发使能寄存器(EXTI_RTEN) ..... 161
6.6.4. 下降沿触发使能寄存器(EXTI_FTEN) ..... 161
6.6.5. 软件中断事件寄存器(EXTI_SWIEV) ..... 161
6.6.6. 挂起寄存器(EXTI_PD) ..... 162
7. 通用和备用输入/输出接口(GPIO 和 AFIO) ..... 163
7.1. 简介 ..... 163
7.2. 主要特性 ..... 163
7.3. 功能描述 ..... 163

7.3.1. GPIO 引脚配置 ..... 164
7.3.2. 外部中断/事件线 ..... 165
7.3.3. 备用功能(AF) ..... 165
7.3.4. 附加功能 ..... 165
7.3.5. 输入配置 ..... 165
7.3.6. 输出配置 ..... 166
7.3.7. 模拟配置 ..... 166
7.3.8. 备用功能(AF)配置 ..... 167
7.3.9. GPIO 锁定功能 ..... 167
7.3.10. GPIO 单周期输出翻转功能 ..... 168
**7.4. GPIO 寄存器 ..... 169**
7.4.1. 端口控制寄存器(GPIOx_CTL, x = A…I) ..... 169
7.4.2. 端口输出模式寄存器(GPIOx_OMODE, x = A…I) ..... 171
7.4.3. 端口输出速度寄存器(GPIOx_OSPD, x = A…I) ..... 172
7.4.4. 端口上拉/下拉寄存器(GPIOx_PUD, x = A…I) ..... 174
7.4.5. 端口输入状态寄存器(GPIOx_ISTAT, x = A…I) ..... 176
7.4.6. 端口输出控制寄存器(GPIOx_OCTL, x = A…I) ..... 176
7.4.7. 端口位操作寄存器(GPIOx_BOP, x = A…I) ..... 177
7.4.8. 端口配置锁定寄存器(GPIOx_LOCK, x = A…I) ..... 177
7.4.9. 备用功能选择寄存器 0(GPIOx_AFSEL0, x = A…I) ..... 178
7.4.10. 备用功能选择寄存器 1(GPIOx_AFSEL1, x = A…I) ..... 179
7.4.11. 位清除寄存器(GPIOx_BC, x = A…I) ..... 180
7.4.12. 端口位翻转寄存器(GPIOx_TG, x = A…I) ..... 180
**8. 循环冗余校验计算单元(CRC) ..... 182**
**8.1. 简介 ..... 182**
**8.2. 主要特征 ..... 182**
**8.3. 功能说明 ..... 183**
**8.4. CRC 寄存器 ..... 184**
8.4.1. 数据寄存器(CRC_DATA) ..... 184
8.4.2. 独立数据寄存器(CRC_FDATA) ..... 184
8.4.3. 控制寄存器(CRC_CTL) ..... 185
**9. 真随机数生成器(TRNG) ..... 186**
**9.1. 简介 ..... 186**
**9.2. 主要特性 ..... 186**
**9.3. 功能描述 ..... 186**

9.3.1. 操作流程 ..... 187
9.3.2. 错误标志 ..... 187
**9.4. TRNG 寄存器 ..... 188**
9.4.1. 控制寄存器 (TRNG_CTL) ..... 188
9.4.2. 状态寄存器 (TRNG_STAT) ..... 188
9.4.3. 数据寄存器 (TRNG_DATA) ..... 189
**10. 直接存储器访问控制器(DMA) ..... 191**
**10.1. 简介 ..... 191**
**10.2. 主要特征 ..... 191**
**10.3. 结构框图 ..... 192**
**10.4. 功能描述 ..... 192**
10.4.1. 外设握手 ..... 193
10.4.2. 数据处理 ..... 195
10.4.3. 地址生成 ..... 199
10.4.4. 循环模式 ..... 200
10.4.5. 存储切换模式 ..... 200
10.4.6. 传输控制器 ..... 201
10.4.7. 传输操作 ..... 201
10.4.8. 传输完成 ..... 202
10.4.9. 通道配置 ..... 203
**10.5. 中断 ..... 204**
10.5.1. 标志 ..... 204
10.5.2. 异常 ..... 205
10.5.3. 错误 ..... 205
**10.6. DMA 寄存器 ..... 207**
10.6.1. 中断标志位寄存器 0(DMA_INTF0) ..... 207
10.6.2. 中断标志位寄存器 1(DMA_INTF1) ..... 208
10.6.3. 中断标志位清除寄存器(DMA_INTC0) ..... 209
10.6.4. 中断标志位清除寄存器 1(DMA_INTC1) ..... 209
10.6.5. 通道 x 控制寄存器(DMA_CHxCTL) ..... 210
10.6.6. 通道 x 计数寄存器(DMA_CHxCNT) ..... 214
10.6.7. 通道 x 外设基地址寄存器(DMA_CHxPADDR) ..... 214
10.6.8. 通道 x 存储器 0 基地址寄存器(DMA_CHxM0ADDR) ..... 215
10.6.9. 通道 x 存储器 1 基地址寄存器(DMA_CHxM1ADDR) ..... 215
10.6.10. 通道 xFIFO 控制寄存器(DMA_CHxFCTL) ..... 216
**11. 图像处理加速器(IPA) ..... 218**

**11.1. 简介 ..... 218**

**11.2. 主要特性 ..... 218**

**11.3. 结构框图 ..... 219**

**11.4. 功能概述 ..... 219**

11.4.1. 传输操作 ..... 220
11.4.2. 前景层和背景层 LUT ..... 220
11.4.3. 前景层和背景层像素通道扩展（PCE） ..... 221
11.4.4. 混合 ..... 223
11.4.5. 目标像素通道压缩（PCC） ..... 224
11.4.6. 内部定时器 ..... 225
11.4.7. 行标记 ..... 226
11.4.8. 传输流 ..... 226
11.4.9. 配置 ..... 227

**11.5. 中断 ..... 230**

**11.6. IPA 寄存器 ..... 233**

11.6.1. 控制寄存器（IPA_CTL） ..... 233
11.6.2. 中断状态寄存器（IPA_INTF） ..... 234
11.6.3. 中断标志清除寄存器（IPA_INTC） ..... 235
11.6.4. 前景层存储区基地址寄存器（IPA_FMADDR） ..... 236
11.6.5. 前景层行偏移寄存器（IPA_FLOFF） ..... 237
11.6.6. 背景层存储区基地址寄存器（IPA_BMADDR） ..... 237
11.6.7. 背景层行偏移寄存器（IPA_BLOFF） ..... 238
11.6.8. 前景层像素控制寄存器（IPA_FPCTL） ..... 238
11.6.9. 前景层像素值寄存器（IPA_FPV） ..... 240
11.6.10. 背景层像素控制寄存器（IPA_BPCTL） ..... 240
11.6.11. 背景层像素值寄存器（IPA_BPV） ..... 242
11.6.12. 前景层 LUT 存储区基地址寄存器（IPA_FLMADDR） ..... 243
11.6.13. 背景层 LUT 存储区基地址寄存器（IPA_BLMADDR） ..... 243
11.6.14. 目标像素控制寄存器（IPA_DPCTL） ..... 244
11.6.15. 目标像素值寄存器（IPA_DPV） ..... 244
11.6.16. 目标存储区基地址寄存器（IPA_DMADDR） ..... 247
11.6.17. 目标行偏移寄存器（IPA_DLOFF） ..... 247
11.6.18. 图像大小寄存器（IPA_IMS） ..... 248
11.6.19. 行标记寄存器（IPA_LM） ..... 248
11.6.20. 内部定时器控制寄存器（IPA_ITCTL） ..... 249

**12. 调试(DBG) ..... 251**

**12.1. 简介 ..... 251**

12.2. JTAG/SW 功能说明 ..... 251
12.2.1. 切换 JTAG/ SW 接口 ..... 251
12.2.2. 引脚分配 ..... 251
12.2.3. JTAG 链状结构 ..... 252
12.2.4. 调试复位 ..... 252
12.2.5. JEDEC-106 ID code ..... 252
12.3. 调试保持功能说明 ..... 252
12.3.1. 低功耗模式调试支持 ..... 252
12.3.2. TIMER, I2C, RTC, WWDGT, FWDGT 和 CAN 外设调试支持 ..... 253
12.4. DBG 寄存器 ..... 254
12.4.1. ID 寄存器 (DBG_ID) ..... 254
12.4.2. 控制寄存器 0(DBG_CTL0) ..... 254
12.4.3. 控制寄存器 1 (DBG_CTL1) ..... 255
12.4.4. 控制寄存器 (DBG_CTL2) ..... 257
13. 可编程参考电流(IREF) ..... 259
13.1. 简介 ..... 259
13.2. 主要特性 ..... 259
13.3. 功能描述 ..... 259
13.3.1. 信号描述 ..... 259
13.3.2. 用户校验 ..... 259
13.4. IREF 寄存器 ..... 260
14. 模数转换器(ADC) ..... 261
14.1. 简介 ..... 261
14.2. 主要特征 ..... 261
14.3. 引脚和内部信号 ..... 262
14.4. 功能说明 ..... 263
14.4.1. 前置校准功能 ..... 263
14.4.2. ADC 时钟 ..... 264
14.4.3. ADCON 使能 ..... 264
14.4.4. 常规序列 ..... 264
14.4.5. 运行模式 ..... 264

14.4.6. 转换结果阈值监测功能 ..... 267
14.4.7. 数据存储模式 ..... 267
14.4.8. 采样时间配置 ..... 268
14.4.9. 外部触发 ..... 268
14.4.10. DMA 请求 ..... 269
14.4.11. 溢出检测 ..... 269
14.4.12. ADC 内部通道 ..... 269
14.4.13. 电池电压监测 ..... 270
14.4.14. 可编程分辨率(DRES) ..... 270
14.4.15. 片上硬件过采样 ..... 270
**14.5. ADC 同步模式 ..... 272**
14.5.1. 独立模式 ..... 273
14.5.2. 常规并行模式 ..... 273
14.5.3. 常规跟随模式 ..... 274
14.5.4. 在 ADC 同步模式中使用 DMA ..... 275
**14.6. 中断 ..... 275**
**14.7. ADC 寄存器 ..... 277**
14.7.1. 状态寄存器 (ADC_STAT) ..... 277
14.7.2. 控制寄存器 0 (ADC_CTL0) ..... 278
14.7.3. 控制寄存器 1 (ADC_CTL1) ..... 279
14.7.4. 采样时间寄存器 0 (ADC_SAMPT0) ..... 281
14.7.5. 采样时间寄存器 1 (ADC_SAMPT1) ..... 282
14.7.6. 看门狗高阈值寄存器 (ADC_WDHT) ..... 283
14.7.7. 看门狗低阈值寄存器 (ADC_WDLT) ..... 283
14.7.8. 常规序列寄存器 0 (ADC_RSQ0) ..... 284
14.7.9. 常规序列寄存器 1 (ADC_RSQ1) ..... 284
14.7.10. 常规序列寄存器 2 (ADC_RSQ2) ..... 285
14.7.11. 常规数据寄存器 (ADC_RDATA) ..... 285
14.7.12. 过采样控制寄存器 (ADC_OVSAMPCTL) ..... 286
14.7.13. 摘要状态寄存器 (ADC_SSTAT) ..... 287
14.7.14. 同步控制寄存器 (ADC_SYNCCTL) ..... 288
14.7.15. 同步常规数据寄存器 (ADC_SYNCDATA) ..... 290
**15. 数模转换器(DAC) ..... 291**
**15.1. 简介 ..... 291**
**15.2. 主要特征 ..... 291**
**15.3. 功能描述 ..... 292**
15.3.1. DAC 使能 ..... 292

15.3.2. DAC 输出缓冲 ..... 292
15.3.3. DAC 数据配置 ..... 292
15.3.4. DAC 触发 ..... 293
15.3.5. DAC 转换 ..... 293
15.3.6. DAC 噪声波 ..... 293
15.3.7. DAC 输出电压 ..... 294
15.3.8. DMA 请求 ..... 294
15.3.9. DAC 并发转换 ..... 295
**15.4. DAC 寄存器 ..... 296**
15.4.1. DACx 控制寄存器(DAC_CTL0) ..... 296
15.4.2. DACx 软件触发寄存器(DAC_SWT) ..... 298
15.4.3. DACx_OUT0 12 位右对齐数据保持寄存器(DAC_OUT0_R12DH) ..... 299
15.4.4. DACx_OUT0 12 位左对齐数据保持寄存器(DAC_OUT0_L12DH) ..... 299
15.4.5. DACx_OUT0 8 位右对齐数据保持寄存器(DAC_OUT0_R8DH) ..... 300
15.4.6. DACx_OUT1 12 位右对齐数据保持寄存器(DAC_OUT1_R12DH) ..... 300
15.4.7. DACx_OUT1 12 位左对齐数据保持寄存器(DAC_OUT1_L12DH) ..... 301
15.4.8. DACx_OUT1 8 位右对齐数据保持寄存器(DAC_OUT1_R8DH) ..... 301
15.4.9. DACx 并发模式 12 位右对齐数据保持寄存器(DACC_R12DH) ..... 302
15.4.10. DACx 并发模式 12 位左对齐数据保持寄存器(DACC_L12DH) ..... 302
15.4.11. DACx 并发模式 8 位右对齐数据保持寄存器(DACC_R8DH) ..... 303
15.4.12. DACx_OUT0 数据输出寄存器(DAC_OUT0_DO) ..... 303
15.4.13. DACx_OUT1 数据输出寄存器(DAC_OUT1_DO) ..... 304
15.4.14. DACx 状态寄存器 0(DAC_STAT0) ..... 304
**16. 看门狗定时器(WDGT) ..... 306**
**16.1. 独立看门狗定时器(FWDGT) ..... 306**
16.1.1. 简介 ..... 306
16.1.2. 主要特征 ..... 306
16.1.3. 功能说明 ..... 306
16.1.4. FWDGT 寄存器 ..... 309
**16.2. 窗口看门狗定时器(WWDGT) ..... 312**
16.2.1. 简介 ..... 312
16.2.2. 主要特征 ..... 312
16.2.3. 功能说明 ..... 312
16.2.4. WWDGT 寄存器 ..... 315
**17. 实时时钟(RTC) ..... 317**
**17.1. 简介 ..... 317**
**17.2. 主要特性 ..... 317**

**17.3. 功能描述 ..... 318**

17.3.1. 结构框图 ..... 318
17.3.2. 时钟源和预分频 ..... 319
17.3.3. 影子寄存器 ..... 319
17.3.4. 位域可屏蔽可配置的闹钟 ..... 319
17.3.5. 可配置周期的自动唤醒定时器 ..... 320
17.3.6. RTC 初始化和配置 ..... 320
17.3.7. 读取日历 ..... 321
17.3.8. RTC 复位 ..... 322
17.3.9. RTC 移位功能 ..... 323
17.3.10. RTC 参考时钟检测 ..... 323
17.3.11. RTC 数字粗校准 ..... 324
17.3.12. RTC 数字平滑校准 ..... 324
17.3.13. 时间戳功能 ..... 326
17.3.14. 侵入检测 ..... 326
17.3.15. 校准时钟输出 ..... 327
17.3.16. 闹钟输出 ..... 328
17.3.17. RTC 省电模式管理 ..... 328
17.3.18. RTC 中断 ..... 328

**17.4. RTC 寄存器 ..... 330**

17.4.1. 时间寄存器(RTC_TIME) ..... 330
17.4.2. 日期寄存器(RTC_DATE) ..... 330
17.4.3. 控制寄存器(RTC_CTL) ..... 331
17.4.4. 状态寄存器(RTC_STAT) ..... 334
17.4.5. 预分频寄存器(RTC_PSC) ..... 335
17.4.6. 唤醒定时器寄存器(RTC_WUT) ..... 336
17.4.7. 粗校准寄存器(RTC_COSC) ..... 337
17.4.8. 闹钟 0 时间日期寄存器(RTC_ALRM0TD) ..... 337
17.4.9. 闹钟 1 时间日期寄存器(RTC_ALRM1TD) ..... 338
17.4.10. 写保护钥匙寄存器(RTC_WPK) ..... 340
17.4.11. 亚秒寄存器(RTC_SS) ..... 340
17.4.12. 移位控制寄存器(RTC_SHIFTCTL) ..... 340
17.4.13. 时间戳时间寄存器(RTC_TTS) ..... 341
17.4.14. 时间戳日期寄存器(RTC_DTS) ..... 342
17.4.15. 时间戳亚秒寄存器(RTC_SSTS) ..... 342
17.4.16. 高精度频率补偿寄存器(RTC_HRFC) ..... 343
17.4.17. 侵入寄存器(RTC_TAMP) ..... 344
17.4.18. 闹钟 0 亚秒寄存器(RTC_ALRM0SS) ..... 346
17.4.19. 闹钟 1 亚秒寄存器(RTC_ALRM1SS) ..... 347
17.4.20. 备份寄存器(RTC_BKPx)(x=0..19) ..... 348

**18. 定时器(TIMERx) ..... 349**

**18.1. 高级定时器(TIMERx,x=0,7) ..... 350**

18.1.1. 简介 ..... 350
18.1.2. 主要特性 ..... 350
18.1.3. 功能描述 ..... 350
18.1.4. TIMERx 寄存器(x=0,7) ..... 376

**18.2. 通用定时器 L0(TIMERx, x=1,2,3,4) ..... 401**

18.2.1. 简介 ..... 401
18.2.2. 主要特性 ..... 401
18.2.3. 功能描述 ..... 401
18.2.4. TIMERx 寄存器(x=1,2,3,4) ..... 416

**18.3. 通用定时器 L1(TIMERx, x=8,11) ..... 441**

18.3.1. 简介 ..... 441
18.3.2. 主要特性 ..... 441
18.3.3. 功能描述 ..... 441
18.3.4. TIMERx 寄存器(x=8,11) ..... 452

**18.4. 通用定时器 L2(TIMERx, x=9,10,12,13) ..... 464**

18.4.1. 简介 ..... 464
18.4.2. 主要特性 ..... 464
18.4.3. 功能描述 ..... 464
18.4.4. TIMERx 寄存器(x=9,10,12,13) ..... 472

**18.5. 基本定时器(TIMERx, x=5,6) ..... 483**

18.5.1. 简介 ..... 483
18.5.2. 主要特性 ..... 483
18.5.3. 功能描述 ..... 483
18.5.4. TIMERx 寄存器(x=5,6) ..... 487

**19. 通用同步异步收发器(USART) ..... 492**

**19.1. 简介 ..... 492**

**19.2. 主要特性 ..... 492**

**19.3. 功能描述 ..... 493**

19.3.1. USART 帧格式 ..... 494
19.3.2. 波特率发生 ..... 495
19.3.3. USART 发送器 ..... 495
19.3.4. USART 接收器 ..... 496
19.3.5. DMA 方式访问数据缓冲区 ..... 497
19.3.6. 硬件流控制 ..... 499

19.3.7. 多处理器通信 ..... 500
19.3.8. LIN 模式 ..... 501
19.3.9. 同步通信模式 ..... 501
19.3.10. 串行红外(IrDA SIR)编解码功能模块 ..... 502
19.3.11. 半双工通信模式 ..... 503
19.3.12. 智能卡(ISO7816-3)模式 ..... 504
19.3.13. USART 中断 ..... 505

**19.4. USART 寄存器 ..... 507**

19.4.1. 状态寄存器 0(USART_STAT0) ..... 507
19.4.2. 数据寄存器(USART_DATA) ..... 509
19.4.3. 波特率寄存器(USART_BAUD) ..... 509
19.4.4. 控制寄存器 0(USART_CTL0) ..... 510
19.4.5. 控制寄存器 1(USART_CTL1) ..... 511
19.4.6. 控制寄存器 2(USART_CTL2) ..... 513
19.4.7. 保护时间和预分频器寄存器(USART_GP) ..... 515
19.4.8. 控制寄存器 3(USART_CTL3) ..... 515
19.4.9. 接收超时寄存器(USART_RT) ..... 517
19.4.10. 状态寄存器 1(USART_STAT1) ..... 517
19.4.11. 兼容性控制寄存器(USART_CHC) ..... 518

**20. 内部集成电路总线接口(I2C) ..... 520**

**20.1. 简介 ..... 520**

**20.2. 主要特征 ..... 520**

**20.3. 功能说明 ..... 521**

20.3.1. SDA 线和 SCL 线 ..... 521
20.3.2. 数据有效性 ..... 522
20.3.3. 开始和停止信号 ..... 522
20.3.4. 时钟同步 ..... 522
20.3.5. 仲裁 ..... 523
20.3.6. I2C 通讯流程 ..... 523
20.3.7. 软件编程模型 ..... 524
20.3.8. SCL 线控制 ..... 532
20.3.9. DMA 模式下数据传输 ..... 533
20.3.10. 报文错误校验 ..... 533
20.3.11. SMBus 支持 ..... 533
20.3.12. SAM_V 支持 ..... 535
20.3.13. 状态、错误和中断 ..... 535

**20.4. I2C 寄存器 ..... 537**

20.4.1. 控制寄存器 0(I2C_CTL0) ..... 537

20.4.2. 控制寄存器 1(I2C_CTL1) ..... 539
20.4.3. 从机地址寄存器 0(I2C_SADDR0) ..... 540
20.4.4. 从机地址寄存器 1(I2C_SADDR1) ..... 540
20.4.5. 传输缓冲区寄存器(I2C_DATA) ..... 541
20.4.6. 传输状态寄存器 0(I2C_STAT0) ..... 541
20.4.7. 传输状态寄存器 1(I2C_STAT1) ..... 543
20.4.8. 时钟配置寄存器(I2C_CKCFG) ..... 545
20.4.9. 上升时间寄存器(I2C_RT) ..... 545
20.4.10. SAM 控制状态寄存器(I2C_SAMCS) ..... 546

**21. 串行外设接口/片上音频接口(SPI/I2S) ..... 548**

**21.1. 简介 ..... 548**

**21.2. 主要特性 ..... 548**

21.2.1. SPI 主要特性 ..... 548
21.2.2. I2S 主要特性 ..... 548

**21.3. SPI 结构框图 ..... 549**

**21.4. SPI 信号线描述 ..... 549**

21.4.1. 常规配置(非 SPI 四线模式) ..... 549
21.4.2. SPI 四线配置 ..... 549

**21.5. SPI 功能描述 ..... 550**

21.5.1. SPI 时序和数据帧格式 ..... 550
21.5.2. NSS 功能 ..... 551
21.5.3. SPI 运行模式 ..... 552
21.5.4. DMA 功能 ..... 559
21.5.5. CRC 功能 ..... 559

**21.6. SPI 中断 ..... 560**

21.6.1. 状态标志位 ..... 560
21.6.2. 错误标志 ..... 560

**21.7. I2S 结构框图 ..... 561**

**21.8. I2S 信号线描述 ..... 562**

**21.9. I2S 功能描述 ..... 562**

21.9.1. I2S 音频标准 ..... 562
21.9.2. I2S 时钟 ..... 569
21.9.3. 运行 ..... 570
21.9.4. DMA 功能 ..... 573

21.10. I2S 中断 ..... 573
21.10.1. 状态标志位 ..... 573
21.10.2. 错误标志 ..... 574
21.11. SPI/I2S 寄存器 ..... 575
21.11.1. 控制寄存器 0(SPI_CTL0) ..... 575
21.11.2. 控制寄存器 1(SPI_CTL1) ..... 577
21.11.3. 状态寄存器(SPI_STAT) ..... 578
21.11.4. 数据寄存器(SPI_DATA) ..... 579
21.11.5. CRC 多项式寄存器(SPI_CRCPOLY) ..... 580
21.11.6. 接收 CRC 寄存器(SPI_RCRC) ..... 580
21.11.7. 发送 CRC 寄存器(SPI_TCRC) ..... 581
21.11.8. I2S 控制寄存器(SPI_I2SCTL) ..... 581
21.11.9. I2S 时钟预分频寄存器(SPI_I2SPSC) ..... 583
21.11.10. SPI5 四线 SPI 控制寄存器(SPI_QCTL) ..... 583
22. 数字摄像头接口(DCI) ..... 585
22.1. 简介 ..... 585
22.2. 主要特性 ..... 585
22.3. 结构框图 ..... 585
22.4. 信号描述 ..... 586
22.5. 功能说明 ..... 586
22.5.1. DCI 硬件同步模式 ..... 586
22.5.2. 内嵌码同步模式 ..... 587
22.5.3. 用快照或连续捕获模式捕获数据 ..... 587
22.5.4. 窗口功能 ..... 587
22.5.5. 像素格式、数据填充和 DMA 接口 ..... 588
22.6. 状态、错误和中断 ..... 589
22.7. DCI 寄存器 ..... 590
22.7.1. 控制寄存器(DCI_CTL) ..... 590
22.7.2. 状态寄存器 0(DCI_STAT0) ..... 591
22.7.3. 状态寄存器 1(DCI_STAT1) ..... 592
22.7.4. 中断使能寄存器(DCI_INTEN) ..... 592
22.7.5. 中断标志寄存器(DCI_INTF) ..... 593
22.7.6. 中断标志清除寄存器(DCI_INTC) ..... 594
22.7.7. 同步码寄存器(DCI_SC) ..... 594

22.7.8. 同步码屏蔽寄存器(DCI_SCUMSK) ..... 595
22.7.9. 剪裁窗口开始位置寄存器(DCI_CWSPOS) ..... 595
22.7.10. 剪裁窗口大小寄存器(DCI_CWSZ) ..... 596
22.7.11. 数据寄存器(DCI_DATA) ..... 596

**23. TFT-LCD 接口(TLI) ..... 598**

**23.1. 简介 ..... 598**

**23.2. 主要特点 ..... 598**

**23.3. 结构框图 ..... 598**

**23.4. 信号线描述 ..... 599**

**23.5. 功能描述 ..... 599**

23.5.1. LCD 显示时序 ..... 599
23.5.2. 像素 DMA 功能 ..... 600
23.5.3. 像素格式 ..... 601
23.5.4. 层窗口和混合功能 ..... 601
23.5.5. Layer 配置重载 ..... 602
23.5.6. 抖动 ..... 602

**23.6. 中断 ..... 602**

**23.7. TLI 寄存器 ..... 604**

23.7.1. 同步脉冲宽度寄存器(TLI_SPSZ) ..... 604
23.7.2. 后沿宽度寄存器(TLI_BPSZ) ..... 604
23.7.3. 有效宽度寄存器(TLI_ASZ) ..... 605
23.7.4. 总宽度寄存器(TLI_TSZ) ..... 605
23.7.5. 控制寄存器(TLI_CTL) ..... 606
23.7.6. 重载层配置寄存器(TLI_RL) ..... 607
23.7.7. 背景色配置寄存器(TLI_BGC) ..... 608
23.7.8. 中断使能寄存器(TLI_INTEN) ..... 608
23.7.9. 中断标志寄存器(TLI_INTF) ..... 609
23.7.10. 中断标志清除寄存器(TLI_INTC) ..... 609
23.7.11. 行标记寄存器(TLI_LM) ..... 610
23.7.12. 当前像素位置寄存器(TLI_CPPOS) ..... 610
23.7.13. 状态寄存器(TLI_STAT) ..... 611
23.7.14. 第 x 层控制寄存器(TLI_LxCTL)(x = 0, 1) ..... 612
23.7.15. 第 x 层水平位置参数寄存器(TLI_LxHPOS)(x = 0, 1) ..... 612
23.7.16. 第 x 层垂直位置参数寄存器(TLI_LxVPOS)(x = 0, 1) ..... 613
23.7.17. 第 x 层色键值寄存器(TLI_LxCKEY)(x = 0, 1) ..... 613
23.7.18. 第 x 层像素格式寄存器(TLI_LxPPF)(x = 0, 1) ..... 614

23.7.19. 第 x 层恒定 Alpha 寄存器(TLI_LxSA)(x = 0, 1) ..... 614
23.7.20. 第 x 层默认颜色寄存器(TLI_LxDC)(x = 0, 1) ..... 615
23.7.21. 第 x 层混合寄存器(TLI_LxBLEND)(x = 0, 1) ..... 615
23.7.22. 第 x 层帧基地址寄存器(TLI_LxFBADDR)(x = 0, 1) ..... 616
23.7.23. 第 x 层行长度寄存器(TLI_LxFLLEN)(x = 0, 1) ..... 617
23.7.24. 第 x 层总行数寄存器(TLI_LxFTLN)(x = 0, 1) ..... 617
23.7.25. 第 x 层颜色查找表寄存器(TLI_LxLUT)(x = 0, 1) ..... 618

**24. SDIO 接口(SDIO) ..... 619**

**24.1. 简介 ..... 619**

**24.2. 主要特性 ..... 619**

**24.3. SDIO 总线拓扑 ..... 619**

**24.4. SDIO 功能描述 ..... 621**

24.4.1. SDIO 适配器 ..... 622
24.4.2. APB2 接口 ..... 626

**24.5. 卡功能描述 ..... 627**

24.5.1. 卡寄存器 ..... 627
24.5.2. 命令 ..... 628
24.5.3. 响应 ..... 636
24.5.4. 数据包格式 ..... 640
24.5.5. 卡的两种状态 ..... 641

**24.6. 编程序列 ..... 647**

24.6.1. 卡识别 ..... 647
24.6.2. 无数据命令 ..... 648
24.6.3. 单个数据块或多个数据块写 ..... 649
24.6.4. 单个数据块或多个数据块读 ..... 650
24.6.5. 数据流写和数据流读 (仅适用于 MMC) ..... 650
24.6.6. 擦除 ..... 652
24.6.7. 总线宽度选择 ..... 652
24.6.8. 保护管理 ..... 652
24.6.9. 卡上锁/解锁操作 ..... 653

**24.7. 特定操作 ..... 655**

24.7.1. SD I/O 特定操作 ..... 655
24.7.2. CE-ATA 特定操作 ..... 658

**24.8. SDIO 寄存器 ..... 659**

24.8.1. 电源控制寄存器 (SDIO_PWRCTL) ..... 659

24.8.2. 时钟控制寄存器（SDIO_CLKCTL） 659
24.8.3. 命令参数寄存器(SDIO_CMDAGMT) 660
24.8.4. 命令控制寄存器（SDIO_CMDCTL） 661
24.8.5. 命令索引响应寄存器（SDIO_RSPCMDIDX） 662
24.8.6. 响应寄存器（SDIO_RESPx x=0..3） 663
24.8.7. 数据超时寄存器（SDIO_DATATO） 663
24.8.8. 数据长度寄存器（SDIO_DATALEN） 664
24.8.9. 数据控制寄存器（SDIO_DATACTL） 664
24.8.10. 数据计数寄存器（SDIO_DATACNT） 666
24.8.11. 状态寄存器（SDIO_STAT） 666
24.8.12. 中断清除寄存器（SDIO_INTC） 668
24.8.13. 中断使能寄存器（SDIO_INTEN） 669
24.8.14. FIFO 计数寄存器（SDIO_FIFOCNT） 670
24.8.15. FIFO 数据寄存器（SDIO_FIFO） 671

**25. 外部存储器控制器(EXMC) 672**

**25.1. 简介 672**

**25.2. 主要特征 672**

**25.3. 功能说明 672**

25.3.1. 结构框图 672
25.3.2. EXMC 访问基本规范 673
25.3.3. 外部设备地址映射 674
25.3.4. NOR/PSRAM 控制器 677
25.3.5. NAND Flash 或 PC Card 控制器 697
25.3.6. SDRAM 控制器 702

**25.4. EXMC 寄存器 712**

25.4.1. NOR/PSRAM 控制器寄存器 712
25.4.2. NAND Flash/PC Card 控制器寄存器 716
25.4.3. SDRAM 控制器寄存器 722
25.4.4. SQPI-PSRAM 控制器寄存器 728

**26. 控制器局域网络(CAN) 732**

**26.1. 简介 732**

**26.2. 主要特征 732**

**26.3. 功能说明 733**

26.3.1. 工作模式 733
26.3.2. 通信模式 734

26.3.3. 数据发送 ..... 735
26.3.4. 数据接收 ..... 737
26.3.5. 过滤功能 ..... 738
26.3.6. 时间触发通信 ..... 741
26.3.7. 通信参数 ..... 741
26.3.8. 错误标志 ..... 743
26.3.9. 中断 ..... 743
**26.4. CAN 寄存器 ..... 746**
26.4.1. 控制寄存器(CAN_CTL) ..... 746
26.4.2. 状态寄存器(CAN_STAT) ..... 747
26.4.3. 发送状态寄存器(CAN_TSTAT) ..... 749
26.4.4. 接收 FIFO0 寄存器(CAN_RFIFO0) ..... 751
26.4.5. 接收 FIFO1 寄存器(CAN_RFIFO1) ..... 752
26.4.6. 中断使能寄存器 (CAN_INTEN) ..... 753
26.4.7. 错误寄存器(CAN_ERR) ..... 754
26.4.8. 位时序寄存器(CAN_BT) ..... 755
26.4.9. 发送邮箱标识符寄存器(CAN_TMIx)(x = 0...2) ..... 756
26.4.10. 发送邮箱属性寄存器(CAN_TMPx)(x = 0...2) ..... 757
26.4.11. 发送邮箱 data0 寄存器(CAN_TMDATA0x)(x = 0...2) ..... 757
26.4.12. 发送邮箱 data1 寄存器(CAN_TMDATA1x)(x = 0...2) ..... 758
26.4.13. 接收 FIFO 邮箱标识符寄存器(CAN_RFIFOMIx)(x = 0,1) ..... 758
26.4.14. 接收 FIFO 邮箱属性寄存器(CAN_RFIFOMPx)(x = 0,1) ..... 759
26.4.15. 接收 FIFO 邮箱 data0 寄存器(CAN_RFIFOMDATA0x)(x=0,1) ..... 760
26.4.16. 接收 FIFO 邮箱 data1 寄存器(CAN_RFIFOMDATA1x)(x=0,1) ..... 760
26.4.17. 过滤器控制寄存器(CAN_FCTL)(仅 CAN0 可用) ..... 760
26.4.18. 过滤器模式配置寄存器(CAN_FMCFG)(仅 CAN0 可用) ..... 761
26.4.19. 过滤器位宽配置寄存器(CAN_FSCFG)(仅 CAN0 可用) ..... 762
26.4.20. 过滤器关联 FIFO 寄存器(CAN_FAFIFO)(仅 CAN0 可用) ..... 762
26.4.21. 过滤器激活寄存器(CAN_FW)(仅 CAN0 可用) ..... 762
26.4.22. 过滤器(x)数据(y)寄存器(CAN_FxDATAy)(x = 0...27, y = 0,1)(仅 CAN0 可用) ..... 763
**27. 以太网(ENET) ..... 764**
**27.1. 简介 ..... 764**
**27.2. 主要特性 ..... 764**
27.2.1. 模块框图 ..... 765
27.2.2. MAC 802.3 以太网数据包描述 ..... 766
27.2.3. 以太网信号描述 ..... 766
**27.3. 功能描述 ..... 767**
27.3.1. 接口配置 ..... 767
27.3.2. MAC 功能简介 ..... 771

27.3.3. DMA 控制器描述 ..... 780
27.3.4. MAC 统计计数器：MSC ..... 802
27.3.5. 唤醒管理：WUM ..... 802
27.3.6. 精确时间协议：PTP ..... 805
27.3.7. 典型的以太网配置流程示例 ..... 809
27.3.8. 以太网中断 ..... 810
**27.4. ENET 寄存器 ..... 812**
27.4.1. MAC 配置寄存器(ENET_MAC_CFG) ..... 812
27.4.2. MAC 帧过滤器寄存器(ENET_MAC_FRMF) ..... 814
27.4.3. MAC hash 列表高寄存器(ENET_MAC_HLH) ..... 816
27.4.4. MAC hash 列表低寄存器(ENET_MAC_HLL) ..... 816
27.4.5. MAC PHY 控制寄存器(ENET_MAC_PHY_CTL) ..... 816
27.4.6. MAC PHY 数据寄存器(ENET_MAC_PHY_DATA) ..... 817
27.4.7. MAC 流控寄存器(ENET_MAC_FCTL) ..... 818
27.4.8. MAC VLAN 标签寄存器(ENET_MAC_VLT) ..... 819
27.4.9. MAC 远程唤醒帧过滤器寄存器(ENET_MAC_RWFF) ..... 820
27.4.10. MAC 唤醒管理寄存器(ENET_MAC_WUM) ..... 820
27.4.11. MAC 调试寄存器(ENET_MAC_DBG) ..... 821
27.4.12. MAC 中断状态寄存器(ENET_MAC_INTF) ..... 823
27.4.13. MAC 中断屏蔽寄存器(ENET_MAC_INTMSK) ..... 824
27.4.14. MAC 地址 0 高寄存器(ENET_MAC_ADDR0H) ..... 825
27.4.15. MAC 地址 0 低寄存器(ENET_MAC_ADDR0L) ..... 825
27.4.16. MAC 地址 1 高寄存器(ENET_MAC_ADDR1H) ..... 826
27.4.17. MAC 地址 1 低寄存器(ENET_MAC_ADDR1L) ..... 826
27.4.18. MAC 地址 2 高寄存器(ENET_ MAC_ADDR2H) ..... 827
27.4.19. MAC 地址 2 低寄存器(ENET_MAC_ADDR2L) ..... 828
27.4.20. MAC 地址 3 高寄存器(ENET_MAC_ADDR3H) ..... 828
27.4.21. MAC 地址 3 低寄存器(ENET_MAC_ADDR3L) ..... 829
27.4.22. MAC 流控阈值寄存器(ENET_MAC_FCTH) ..... 829
27.4.23. MSC 控制寄存(ENET_MSC_CTL) ..... 830
27.4.24. MSC 接收中断状态寄存器(ENET_MSC_RINTF) ..... 831
27.4.25. MSC 发送中断状态寄存器(ENET_MSC_TINTF) ..... 832
27.4.26. MSC 接收中断屏蔽寄存器(ENET_MSC_RINTMSK) ..... 832
27.4.27. MSC 发送中断屏蔽寄存器(ENET_MSC_TINTMSK) ..... 833
27.4.28. MSC 1 次冲突后发送"好"帧的计数器寄存器(ENET_MSC_SCCNT) ..... 834
27.4.29. MSC 1 次以上冲突后发送"好"帧的计数器寄存器(ENET_MSC_MSCCNT) ..... 834
27.4.30. MSC 发送"好"帧计数器寄存器(ENET_MSC_TGFCNT) ..... 835
27.4.31. MSC CRC 错误接收帧计数器寄存器(ENET_MSC_RFCECNT) ..... 835
27.4.32. MSC 对齐错误接收帧计数器寄存器(ENET_MSC_RFAECNT) ..... 836
27.4.33. MSC"好"单播帧接收帧计数器寄存器(ENET_MSC_RGUFCNT) ..... 836
27.4.34. PTP 时间戳控制寄存器(ENET_PTP_TSCTL) ..... 836
27.4.35. PTP 亚秒递增寄存器(ENET_PTP_SSINC) ..... 839
27.4.36. PTP 时间戳高寄存器(ENET_PTP_TSH) ..... 839

27.4.37. PTP 时间戳低寄存器(ENET_PTP_TSL) ..... 840
27.4.38. PTP 时间戳高更新寄存器(ENET_PTP_TSUH) ..... 840
27.4.39. PTP 时间戳低更新寄存器(ENET_PTP_TSUL) ..... 841
27.4.40. PTP 时间戳加数寄存器(ENET_PTP_TSADDEND) ..... 841
27.4.41. PTP 期望时间高寄存器(ENET_PTP_ETH) ..... 842
27.4.42. PTP 期望时间低寄存器(ENET_PTP_ETL) ..... 842
27.4.43. PTP 时间戳标志寄存器(ENET_PTP_TSF) ..... 842
27.4.44. PTP PPS 控制寄存器(ENET_PTP_PPSCTL) ..... 843
27.4.45. DMA 总线控制寄存器(ENET_DMA_BCTL) ..... 844
27.4.46. DMA 发送查询使能寄存器(ENET_DMA_TPEN) ..... 845
27.4.47. DMA 接收查询使能寄存器(ENET_DMA_RPEN) ..... 846
27.4.48. DMA 接收描述符列表地址寄存器(ENET_DMA_RDTADDR) ..... 847
27.4.49. DMA 发送描述符列表地址寄存器(ENET_DMA_TDTADDR) ..... 847
27.4.50. DMA 状态寄存器(ENET_DMA_STAT) ..... 848
27.4.51. DMA 控制寄存器(ENET_DMA_CTL) ..... 851
27.4.52. DMA 中断使能寄存器(ENET_DMA_INTEN) ..... 853
27.4.53. DMA 丢失帧和缓存溢出计数器寄存器(ENET_DMA_MFBOCNT) ..... 855
27.4.54. DMA 接收状态看门狗计数器寄存器(ENET_DMA_RSWDC) ..... 856
27.4.55. DMA 当前发送描述符地址寄存器(ENET_DMA_CTDADDR) ..... 856
27.4.56. DMA 当前接收描述符地址寄存器(ENET_DMA_CRDADDR) ..... 857
27.4.57. DMA 当前发送缓存地址寄存器(ENET_DMA_CTBADDR) ..... 857
27.4.58. DMA 当前接收缓存地址寄存器(ENET_DMA_CRBADDR) ..... 858

**28. 通用串行总线全速接口(USBFS) ..... 859**

**28.1. 概述 ..... 859**

**28.2. 主要特性 ..... 859**

**28.3. 结构框图 ..... 860**

**28.4. 信号线描述 ..... 860**

**28.5. 功能描述 ..... 860**

28.5.1. USBFS 时钟及工作模式 ..... 860
28.5.2. USB 主机功能 ..... 862
28.5.3. USB 设备功能 ..... 864
28.5.4. OTG 功能概述 ..... 865
28.5.5. 数据 FIFO ..... 866
28.5.6. 操作手册 ..... 867
28.5.7. 中断 ..... 871

**28.6. USBFS 寄存器 ..... 872**

28.6.1. USBFS 全局寄存器组 ..... 872

28.6.2. 主机控制和状态寄存器 ..... 892
28.6.3. 设备控制和状态寄存器 ..... 903
28.6.4. 电源和时钟控制寄存器(USBFS_PWRCLKCTL)) ..... 925
**29. 通用串行总线高速接口(USBHS) ..... 926**
**29.1. 概述 ..... 926**
**29.2. 主要特性 ..... 926**
**29.3. 结构框图 ..... 927**
**29.4. 信号线描述 ..... 927**
**29.5. 功能描述 ..... 927**
29.5.1. USBHS PHY 选择、时钟及工作模式 ..... 927
29.5.2. USB 主机功能 ..... 930
29.5.3. USB 设备功能 ..... 932
29.5.4. OTG 功能概述 ..... 933
29.5.5. 数据 FIFO ..... 934
29.5.6. DMA 功能 ..... 936
29.5.7. 操作手册 ..... 937
**29.6. 中断 ..... 941**
**29.7. USBHS 寄存器 ..... 943**
29.7.1. 全局控制与状态寄存器组 ..... 943
29.7.2. 主机控制和状态寄存器 ..... 964
29.7.3. 设备控制和状态寄存器 ..... 976
29.7.4. 电源和时钟控制寄存器(USBHS_PWRCLKCTL) ..... 1003
**30. 版本历史 ..... 1005**

# 图索引

图 1-1. Cortex®-M4 结构框图 ..... 35
图 1-2. GD32A490xx 器件的系统架构示意图 ..... 38
图 2-1. 页擦除操作流程 ..... 59
图 2-2. 扇区擦除操作流程 ..... 60
图 2-3. 整片擦除操作流程 ..... 61
图 2-4. 闪存编程操作流程 ..... 62
图 3-1. 电源域概览 ..... 76
图 3-2. 上电 / 掉电复位波形图 ..... 78
图 3-3. BOR 波形图 ..... 78
图 3-4. LVD 阈值波形图 ..... 79
图 4-1. 系统复位电路 ..... 88
图 4-2. 时钟树 ..... 89
图 4-3. HXTAL 时钟源 ..... 91
图 4-4. 旁路模式下 HXTAL 时钟源 ..... 91
图 5-1. CTC 简介 ..... 145
图 5-2. CTC 校准计数器 ..... 146
图 6-1. EXTI 的结构框图 ..... 157
图 7-1. GPIO 端口位的基本结构 ..... 164
图 7-2. 输入配置的基本结构 ..... 166
图 7-3. 输出配置的基本结构 ..... 166
图 7-4. 模拟配置的基本结构 ..... 167
图 7-5. 备用功能配置的基本结构 ..... 167
图 8-1. CRC 计算单元框图 ..... 182
图 9-1. TRNG 模块框图 ..... 186
图 10-1. 系统架构 ..... 192
图 10-2. 三种传输模式的数据流 ..... 193
图 10-3. 握手机制 ..... 194
图 10-4. PWIDTH 为'2b00'时，数据的打包/解包 ..... 198
图 10-5. PWIDTH 为'2b01'时，数据的打包/解包 ..... 199
图 10-6. PWIDTH 为'2b10'时，数据的打包/解包 ..... 199
图 10-7. 存储切换模式 ..... 200
图 10-8. DMA0 与 DMA1 的系统连接 ..... 206
图 11-1. IPA 模块框图 ..... 219
图 11-2. 从'RGB888'到'ARGB8888'像素格式扩展 ..... 222
图 11-3. 从'RGB565'到'ARGB8888'像素格式扩展 ..... 222
图 11-4. 从'ARGB1555'或'ARGB4444'到'ARGB8888'像素格式扩展 ..... 223
图 11-5. 像素压缩 ..... 225
图 11-6. 内部定时器操作 ..... 226
图 11-7. IPA 的系统连接 ..... 232
图 14-1. ADC 模块框图 ..... 263
图 14-2. 单次运行模式 ..... 264

图 14-3. 连续运行模式 ..... 265
图 14-4. 扫描运行模式，且连续转换模式失能 ..... 266
图 14-5. 扫描运行模式，连续运行模式使能 ..... 266
图 14-6. 间断转换模式 ..... 266
图 14-7. 12 位数据存储模式 ..... 267
图 14-8. 10 位数据存储模式 ..... 267
图 14-9. 8 位数据存储模式 ..... 267
图 14-10. 6 位数据存储模式 ..... 268
图 14-11. 20 位到 16 位的结果截断 ..... 271
图 14-12. 右移 5 位和取整的数例 ..... 271
图 14-13. ADC 同步框图 ..... 273
图 14-14. 基于 16 个通道的常规并行模式 ..... 274
图 14-15. 一个采用连续运行模式通道上的跟随模式 ..... 274
图 15-1. DAC 结构框图 ..... 291
图 15-2. DAC LFSR 算法 ..... 294
图 15-3. DAC 三角噪声模式生成的波形 ..... 294
图 16-1. 独立看门狗定时器框图 ..... 307
图 16-2. 窗口看门狗定时器框图 ..... 312
图 16-3. 窗口看门狗定时器时序图 ..... 313
图 17-1. RTC 结构框图 ..... 318
图 18-1. 高级定时器结构框图 ..... 351
图 18-2. 内部时钟分频为 1 时，计数器的时序图 ..... 352
图 18-3. 当 PSC 数值从 0 变到 2 时，计数器的时序图 ..... 353
图 18-4. 向上计数时序图，PSC=0/2 ..... 354
图 18-5. 向上计数时序图，在运行时改变 TIMERx_CAR 寄存器的值 ..... 354
图 18-6. 向下计数时序图，PSC=0/2 ..... 355
图 18-7. 向下计数时序图，在运行时改变 TIMERx_CAR 寄存器值 ..... 356
图 18-8. 中央计数模式计数器时序图 ..... 357
图 18-9. 中央计数模式下计数器重复时序图 ..... 358
图 18-10. 在向上计数模式下计数器重复时序图 ..... 358
图 18-11. 在向下计数模式下计数器重复时序图 ..... 359
图 18-12. 通道输入捕获原理 ..... 360
图 18-13. 三种输出比较模式 ..... 361
图 18-14. EAPWM 时序图 ..... 362
图 18-15. CAPWM 时序图 ..... 363
图 18-16. 带死区时间的互补输出 ..... 366
图 18-17. 通道响应中止输入(高电平有效)时，输出信号的行为 ..... 367
图 18-18. 在译码器模式 2 且 CI0FE0 极性不反相时计数器行为 ..... 368
图 18-19. 在译码器模式 2 且 CI0FE0 极性反相时计数器行为 ..... 368
图 18-20. 霍尔传感器用在 BLDC 电机控制中 ..... 369
图 18-21. 两个定时器之间的霍尔传感器时序图 ..... 370
图 18-22. 复位模式 ..... 371
图 18-23. 暂停模式 ..... 371
图 18-24. 事件模式 ..... 372

图 18-25. 单脉冲模式，TIMERx_CHxCV = 0x04 TIMERx_CAR=0x60 ..... 372
图 18-26. 定时器 0 主/从模式的例子 ..... 373
图 18-27. 用定时器 2 的 CI0 输入来触发定时器 0 和定时器 2 ..... 374
图 18-28. 通用定时器 L0 结构框图 ..... 402
图 18-29. 内部时钟分频为 1 时，计数器的时序图 ..... 403
图 18-30. 当 PSC 数值从 0 变到 2 时，计数器的时序图 ..... 404
图 18-31. 向上计数时序图，PSC=0/2 ..... 405
图 18-32. 向上计数时序图，在运行时改变 TIMERx_CAR 寄存器的值 ..... 405
图 18-33. 向下计数时序图，PSC=0/2 ..... 406
图 18-34. 向下计数时序图，在运行时改变 TIMERx_CAR 寄存器值 ..... 407
图 18-35. 中央计数模式计数器时序图 ..... 408
图 18-36. 通道输入捕获原理 ..... 409
图 18-37. 三种输出比较模式 ..... 410
图 18-38. EAPWM 时序图 ..... 411
图 18-39. CAPWM 时序图 ..... 412
图 18-40. 复位模式 ..... 413
图 18-41. 暂停模式 ..... 414
图 18-42. 事件模式 ..... 414
图 18-43. 通用定时器 L1 结构框图 ..... 442
图 18-44. 内部时钟分频为 1 时，计数器的时序图 ..... 442
图 18-45. 当 PSC 数值从 0 变到 2 时，计数器的时序图 ..... 443
图 18-46. 向上计数时序图，PSC=0/2 ..... 444
图 18-47. 向上计数时序图，在运行时改变 TIMERx_CAR 寄存器的值 ..... 444
图 18-48. 通道输入捕获原理 ..... 445
图 18-49. 三种输出比较模式 ..... 447
图 18-50. EAPWM 时序图 ..... 448
图 18-51. CAPWM 时序图 ..... 448
图 18-52. 复位模式 ..... 449
图 18-53. 暂停模式 ..... 450
图 18-54. 事件模式 ..... 450
图 18-55. 单脉冲模式，TIMERx_CHxCV = 4 TIMERx_CAR=99 ..... 451
图 18-56. 通用定时器 L2 结构框图 ..... 464
图 18-57. 内部时钟分频为 1 时，计数器的时序图 ..... 465
图 18-58. 当 PSC 数值从 0 变到 2 时，计数器的时序图 ..... 466
图 18-59. 向上计数时序图，PSC=0/2 ..... 467
图 18-60. 向上计数时序图，在运行时改变 TIMERx_CAR 寄存器的值 ..... 467
图 18-61. 通道输入捕获原理 ..... 468
图 18-62. 三种输出比较模式 ..... 470
图 18-63. 基本定时器结构框图 ..... 483
图 18-64. 内部时钟分频为 1 时，计数器的时序图 ..... 484
图 18-65. 当 PSC 数值从 0 变到 2 时，计数器的时序图 ..... 484
图 18-66. 向上计数时序图，PSC=0/2 ..... 485
图 18-67. 向上计数时序图，在运行时改变 TIMERx_CAR 寄存器的值 ..... 486
图 19-1. USART 模块内部框图 ..... 494

图 19-2. USART 字符帧(8 数据位和 1 停止位) ..... 494
图 19-3. USART 发送步骤 ..... 496
图 19-4. 过采样方式接收一个数据位(OSB=0) ..... 497
图 19-5. 采用 DMA 方式实现 USART 数据发送配置步骤 ..... 498
图 19-6. 采用 DMA 方式实现 USART 数据接收配置步骤 ..... 499
图 19-7. 两个 USART 之间的硬件流控制 ..... 499
图 19-8. 硬件流控制 ..... 500
图 19-9. 空闲状态下检测断开帧 ..... 501
图 19-10. 数据传输过程中检测断开帧 ..... 501
图 19-11. 同步模式下的 USART 示例 ..... 502
图 19-12. 8-bit 格式的 USART 同步通信波形(CLEN=1) ..... 502
图 19-13. IrDA SIR ENDEC 模块 ..... 503
图 19-14. IrDA 数据调制 ..... 503
图 19-15. ISO7816-3 数据帧格式 ..... 504
图 19-16. USART 中断映射框图 ..... 506
图 20-1. I2C 模块框图 ..... 521
图 20-2. 数据有效性 ..... 522
图 20-3. 起始和停止信号 ..... 522
图 20-4. 时钟同步 ..... 522
图 20-5. SDA 线仲裁 ..... 523
图 20-6. 7 位地址的 I2C 通讯流程 ..... 523
图 20-7. 10 位地址的 I2C 通讯流程(主机发送) ..... 523
图 20-8. 10 位地址的 I2C 通讯流程(主机接收) ..... 524
图 20-9. 从机发送模式(10 位地址模式) ..... 525
图 20-10. 从机接收模式(10 位地址模式) ..... 526
图 20-11. 主机发送模式(10 位地址模式) ..... 528
图 20-12. 主机接收使用方案 A 模式(10 位地址模式) ..... 530
图 20-13. 主机接收使用方案 B 模式(10 位地址模式) ..... 532
图 21-1. SPI 结构框图 ..... 549
图 21-2. 常规模式下的 SPI 时序图 ..... 550
图 21-3. SPI 四线模式下的 SPI 时序图(CKPL=1, CKPH=1, LF=0) ..... 551
图 21-4. 典型的全双工模式连接 ..... 553
图 21-5. 典型的单工模式连接(主机：接收，从机：发送) ..... 554
图 21-6. 典型的单工模式连接(主机：只发送，从机：接收) ..... 554
图 21-7. 典型的双向线连接 ..... 554
图 21-8. 主机 TI 模式在不连续发送时的时序图 ..... 556
图 21-9. 主机 TI 模式在连续发送时的时序图 ..... 556
图 21-10. 从机 TI 模式时序图 ..... 556
图 21-11. SPI 四线模式写操作时序图 ..... 557
图 21-12. SPI 四线模式读操作时序图 ..... 558
图 21-13. I2S 结构框图 ..... 561
图 21-14. I2S 飞利浦标准时序图(DTLEN=00, CHLEN=0, CKPL=0) ..... 562
图 21-15. I2S 飞利浦标准时序图(DTLEN=00, CHLEN=0, CKPL=1) ..... 563
图 21-16. I2S 飞利浦标准时序图(DTLEN=10, CHLEN=1, CKPL=0) ..... 563

图 21-17. I2S 飞利浦标准时序图(DTLEN=10, CHLEN=1, CKPL=1) ..... 563
图 21-18. I2S 飞利浦标准时序图(DTLEN=01, CHLEN=1, CKPL=0) ..... 563
图 21-19. I2S 飞利浦标准时序图(DTLEN=01, CHLEN=1, CKPL=1) ..... 563
图 21-20. I2S 飞利浦标准时序图(DTLEN=00, CHLEN=1, CKPL=0) ..... 564
图 21-21. I2S 飞利浦标准时序图(DTLEN=00, CHLEN=1, CKPL=1) ..... 564
图 21-22. MSB 对齐标准时序图(DTLEN=00, CHLEN=0, CKPL=0) ..... 564
图 21-23. MSB 对齐标准时序图(DTLEN=00, CHLEN=0, CKPL=1) ..... 564
图 21-24. MSB 对齐标准时序图(DTLEN=10, CHLEN=1, CKPL=0) ..... 564
图 21-25. MSB 对齐标准时序图(DTLEN=10, CHLEN=1, CKPL=1) ..... 565
图 21-26. MSB 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=0) ..... 565
图 21-27. MSB 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=1) ..... 565
图 21-28. MSB 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=0) ..... 565
图 21-29. MSB 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=1) ..... 565
图 21-30. LSB 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=0) ..... 566
图 21-31. LSB 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=1) ..... 566
图 21-32. LSB 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=0) ..... 566
图 21-33. LSB 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=1) ..... 566
图 21-34. PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=0) ..... 567
图 21-35. PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=1) ..... 567
图 21-36. PCM 标准短帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=0) ..... 567
图 21-37. PCM 标准短帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=1) ..... 567
图 21-38. PCM 标准短帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=0) ..... 567
图 21-39. PCM 标准短帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=1) ..... 567
图 21-40. PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=0) ..... 568
图 21-41. PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=1) ..... 568
图 21-42. PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=0) ..... 568
图 21-43. PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=1) ..... 568
图 21-44. PCM 标准长帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=0) ..... 568
图 21-45. PCM 标准长帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=1) ..... 568
图 21-46. PCM 标准长帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=0) ..... 569
图 21-47. PCM 标准长帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=1) ..... 569
图 21-48. PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=0) ..... 569
图 21-49. PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=1) ..... 569
图 21-50. I2S 时钟生成结构框图 ..... 569
图 22-1. DCI 模块示意图 ..... 585
图 22-2. 硬件同步模式 ..... 586
图 22-3. 硬件同步模式之 JPEG 格式 ..... 587
图 23-1. TLI 模块框图 ..... 598
图 23-2. 显示时序图 ..... 600
图 23-3. 混合过程框图 ..... 602
图 24-1. SDIO "无响应" 和 "无数据" 操作 ..... 620
图 24-2. SDIO 多块读操作 ..... 620
图 24-3. SDIO 多块写操作 ..... 621
图 24-4. SDIO 数据流读操作 ..... 621

图 24-5. SDIO 数据流写操作 ..... 621
图 24-6. SDIO 框图 ..... 622
图 24-7. 命令标记格式 ..... 628
图 24-8. 响应令牌格式 ..... 637
图 24-9. 1 位数据总线宽度 ..... 640
图 24-10. 4 位数据总线宽度 ..... 640
图 24-11. 8 位数据总线宽度 ..... 641
图 24-12. 通过停止 SDIO_CLK 的读等待操作 ..... 655
图 24-13. 使用 SDIO_DAT[2]信号线的读等待操作 ..... 656
图 24-14. 在功能 1 的多块读周期期间插入功能 2 读周期 ..... 656
图 24-15. 读中断周期时序 ..... 657
图 24-16. 写中断周期时序 ..... 657
图 24-17. 4 位模式下多块读中断周期时序 ..... 657
图 24-18. 4 位模式下多块写中断周期时序 ..... 658
图 24-19. 命令完成信号关闭操作 ..... 658
图 25-1. 系统架构 ..... 673
图 25-2. EXMC Bank 划分 ..... 674
图 25-3. Bank0 地址映射 ..... 675
图 25-4. NAND/PC Card 地址映射 ..... 676
图 25-5. Bank1 通用空间 ..... 676
图 25-6. SDRAM 地址映射 ..... 677
图 25-7. 模式 1 读时序 ..... 681
图 25-8. 模式 1 写时序 ..... 681
图 25-9. 模式 A 读时序 ..... 682
图 25-10. 模式 A 写时序 ..... 683
图 25-11. 模式 2/B 读时序 ..... 684
图 25-12. 模式 2 写时序 ..... 684
图 25-13. 模式 B 写时序 ..... 685
图 25-14. 模式 C 读时序 ..... 686
图 25-15. 模式 C 写时序 ..... 686
图 25-16. 模式 D 读时序 ..... 688
图 25-17. 模式 D 写时序 ..... 688
图 25-18. 复用模式读时序 ..... 689
图 25-19. 复用模式写时序 ..... 690
图 25-20. 异步等待有效时的读时序 ..... 691
图 25-21. 异步等待有效时的写时序 ..... 692
图 25-22. 同步复用突发传输读时序 ..... 693
图 25-23. 同步复用突发传输写时序 ..... 694
图 25-24. SPI-PSRAM 访问时序 ..... 696
图 25-25. SQPI-PSRAM 访问时序 ..... 697
图 25-26. QPI-PSRAM 访问时序 ..... 697
图 25-27. NAND/PC Card 通用空间操作时序 ..... 699
图 25-28. NCE 敏感 NAND Flash 访问时序 ..... 700
图 25-29. SDRAM 系统架构 ..... 703

图 25-30. 突发读操作 ..... 706
图 25-31. 数据采样时钟延迟模块 ..... 706
图 25-32. 突发写操作 ..... 707
图 25-33. FIFO 未命中时的读访问(BRSTRD=1，CL=2，SDCLK=2，PIPED=2) ..... 708
图 25-34. FIFO 命中时的读访问(BRSTRD=1) ..... 708
图 25-35. 跨边界读操作 ..... 709
图 25-36. 跨边界写操作 ..... 709
图 25-37. 自刷新模式进入和退出的处理 ..... 710
图 25-38. 掉电模式进入和退出的处理 ..... 710
图 26-1. CAN 模块结构框图 ..... 733
图 26-2. 发送寄存器 ..... 735
图 26-3. 发送邮箱状态转换 ..... 736
图 26-4. 接收寄存器 ..... 737
图 26-5. 32-bit 位宽过滤器 ..... 738
图 26-6. 16-bit 位宽过滤器 ..... 738
图 26-7. 32-bit 位宽掩码模式过滤器 ..... 739
图 26-8. 16-bit 位宽掩码模式过滤器 ..... 739
图 26-9. 32-bit 位宽列表模式过滤器 ..... 739
图 26-10. 16-bit 位宽列表模式过滤器 ..... 739
图 26-11. 位时序 ..... 742
图 27-1. 以太网模块框图 ..... 765
图 27-2. MAC/带标签的 MAC 帧格式 ..... 766
图 27-3. 站点管理接口信号 ..... 768
图 27-4. 媒体独立接口(MII)信号线 ..... 769
图 27-5. 精简媒体独立接口(RMII)信号线 ..... 771
图 27-6. 描述符的环结构和链结构 ..... 781
图 27-7. 常规发送描述符 ..... 786
图 27-8. 增强发送描述符 ..... 791
图 27-9. 常规接收描述符 ..... 794
图 27-10. 增强接收描述符 ..... 800
图 27-11. 唤醒帧过滤器寄存器 ..... 805
图 27-12. 系统时钟精细校准方法 ..... 806
图 27-13. MAC 控制器中断示意图 ..... 811
图 27-14. 以太网中断示意图 ..... 811
图 27-15. 远程唤醒帧过滤器寄存器 ..... 820
图 28-1. USBFS 结构框图 ..... 860
图 28-2. 在主机或设备模式下连接示意图 ..... 861
图 28-3. OTG 模式下连接示意图 ..... 862
图 28-4. 主机端口状态转移图 ..... 862
图 28-5. 主机模式 FIFO 空间 ..... 866
图 28-6. 主机模式 FIFO 访问寄存器映射表 ..... 866
图 28-7. 设备模式 FIFO 空间 ..... 867
图 28-8. 设备模式 FIFO 访问寄存器映射表 ..... 867
图 29-1. USBHS 结构框图 ..... 927

图 29-2. 在主机或设备模式下，利用内部 PHY 的连接示意图 ..... 928
图 29-3. OTG 模式下使用内部嵌入式 PHY 连接示意图 ..... 929
图 29-4. 使用外部 ULPI PHY 的连接示意图 ..... 930
图 29-5. 主机端口状态转移图 ..... 930
图 29-6. 主机模式 FIFO 空间 ..... 934
图 29-7. 主机模式 FIFO 访问寄存器映射表 ..... 935
图 29-8. 设备模式 FIFO 空间 ..... 935
图 29-9. 设备模式 FIFO 访问寄存器映射表 ..... 936

# 表索引

表 1-1. AHB 互联矩阵的互联关系列表 ..... 36
表 1-2. GD32A490xx 系列器件的存储器映射表 ..... 39
表 1-3. 引导模式 ..... 43
表 2-1. GD32A490xx 闪存基地址和构成 ..... 55
表 2-2. DBS = 1 时 1M 闪存扇区划分 ..... 57
表 2-3. 选项字节 ..... 63
表 2-4. 扇区保护 WP0/WP1 位 ..... 65
表 3-1. 节电模式总结 ..... 82
表 4-1. 时钟输出 0 的时钟源选择 ..... 93
表 4-2. 时钟输出 1 的时钟源选择 ..... 93
表 4-3. 深度睡眠模式下 1.2V 域电压选择 ..... 94
表 6-1. Cotrex-M4 中的 NVIC 异常类型 ..... 153
表 6-2. 中断向量表 ..... 154
表 6-3. EXTI 触发源 ..... 158
表 7-1. GPIO 配置表 ..... 163
表 10-1. 传输模式 ..... 192
表 10-2. DMA0 外设请求 ..... 194
表 10-3. DMA1 外设请求 ..... 194
表 10-4. CNT 配置 ..... 196
表 10-5. FIFO 计数器临界值配置 ..... 197
表 10-6. DMA 中断事件 ..... 204
表 11-1. IPA 转换模式 ..... 220
表 11-2. 前景层和背景层 CLUT 像素格式 ..... 221
表 11-3. 前景层和背景层像素格式 ..... 222
表 11-4. Alpha 通道值调制 ..... 223
表 11-5. 目标像素格式 ..... 224
表 11-6. IPA 中断事件 ..... 230
表 12-1. 调试引脚分配 ..... 252
表 14-1. ADC 内部输入信号 ..... 262
表 14-2. ADC 输入引脚定义 ..... 262
表 14-3. 外部触发模式 ..... 268
表 14-4. ADC 的外部触发源 ..... 268
表 14-5. 不同分辨率对应的 $t_{CONV}$ 时间 ..... 270
表 14-6. N 和 M 的最大输出值(灰色部分表示截断) ..... 272
表 14-7. ADC 同步模式表 ..... 272
表 15-1. DAC 引脚 ..... 292
表 15-2. DAC 触发与输出 ..... 292
表 15-3. DAC 外部触发 ..... 293
表 16-1. 独立看门狗定时器在 32kHz(IRC32K)时的最小 / 最大超时周期 ..... 307
表 16-2. 在 60MHz ($f_{PCLK1}$) 时的最大/最小超时值 ..... 313
表 17-1. 省电模式管理 ..... 328

表 17-2. 中断控制 ..... 328
表 18-1. 定时器(TIMERx)分为五种类型 ..... 349
表 18-2. 由参数控制的互补输出表 ..... 364
表 18-3. 不同译码器模式下的计数方向 ..... 367
表 18-4. 从模式例子列表和举例 ..... 370
表 18-5. 从模式列表和举例 ..... 413
表 18-6. 从模式列表和举例 ..... 449
表 19-1. USART 重要引脚描述 ..... 493
表 19-2. 停止位配置 ..... 494
表 19-3. USART 中断请求 ..... 505
表 20-1. I2C 总线术语说明(参考飞利浦 I2C 规范) ..... 521
表 20-2. 事件状态标志位 ..... 535
表 20-3. 错误标志位 ..... 535
表 21-1. SPI 信号描述 ..... 549
表 21-2. SPI 四线信号描述 ..... 550
表 21-3. 从机模式 NSS 功能 ..... 551
表 21-4. 主机模式 NSS 功能 ..... 552
表 21-5. SPI 运行模式 ..... 552
表 21-6. SPI 中断请求 ..... 561
表 21-7. I2S 比特率计算公式 ..... 570
表 21-8. 音频采样频率计算公式 ..... 570
表 21-9. 各种运行模式下 I2S 接口信号的方向 ..... 570
表 21-10. I2S 中断 ..... 574
表 22-1. DCI 引脚 ..... 586
表 22-2. 字节填充模式下的存储视图 ..... 588
表 22-3. 半字填充模式下的存储视图 ..... 588
表 22-4. 状态/错误标志 ..... 589
表 23-1. TLI 提供的显示接口的引脚 ..... 599
表 23-2. 八种像素格式 ..... 601
表 23-3. 状态标志 ..... 603
表 23-4. 错误标志 ..... 603
表 24-1. SDIO I/O 定义 ..... 622
表 24-2. 命令格式 ..... 628
表 24-3. 卡命令类 (CCCs) ..... 629
表 24-4. 基本命令(class 0) ..... 631
表 24-5. 面向块的读命令(class 2) ..... 632
表 24-6. 流读取命令(class 1)和流写入命令(class 3) ..... 633
表 24-7. 面向块的写命令(class 4) ..... 633
表 24-8. 擦除命令(class 5) ..... 633
表 24-9. 面向块的写保护命令(class 6) ..... 634
表 24-10. 锁卡命令(class 7) ..... 634
表 24-11. 特定应用命令(class 8) ..... 634
表 24-12. I/O 模式命令(class 9) ..... 635
表 24-13. 切换功能命令(class 10) ..... 636

表 24-14. R1 响应 ..... 637
表 24-15. R2 响应 ..... 638
表 24-16. R3 响应 ..... 638
表 24-17. R4 响应(MMC) ..... 638
表 24-18. R4 响应(SD I/O) ..... 639
表 24-19. R5 响应(MMC) ..... 639
表 24-20. R5 响应(SD I/O) ..... 639
表 24-21. R6 响应 ..... 639
表 24-22. R7 响应 ..... 640
表 24-23. 卡状态 ..... 642
表 24-24. SD 状态 ..... 644
表 24-25. 移动性能字段 ..... 645
表 24-26. AU_SIZE 字段 ..... 645
表 24-27. 最大 AU 大小 ..... 646
表 24-28. 擦除大小字段 ..... 646
表 24-29. 擦除超时字段 ..... 646
表 24-30. 擦除偏移字段 ..... 647
表 24-31. 上锁/解锁数据结构 ..... 653
表 24-32. 不同响应类型对应的 SDIO_RESPx 寄存器 ..... 663
表 25-1. SDRAM 地址映射 ..... 677
表 25-2. NOR Flash 接口信号描述 ..... 678
表 25-3. PSRAM 非复用接口信号描述 ..... 678
表 25-4. SQPI-PSRAM 接口信号描述 ..... 678
表 25-5. EXMC 对 NOR，PSRAM 和 SRAM 支持的访问模式 ..... 679
表 25-6. NOR/PSRAM 控制时序参数 ..... 680
表 25-7. EXMC 时序模型 ..... 680
表 25-8. 模式 1 相关寄存器配置 ..... 681
表 25-9. 模式 A 相关寄存器配置 ..... 683
表 25-10. 模式 2/B 相关寄存器配置 ..... 685
表 25-11. 模式 C 相关寄存器配置 ..... 687
表 25-12. 模式 D 相关寄存器配置 ..... 688
表 25-13. 复用模式相关寄存器配置 ..... 690
表 25-14. 同步复用模式读时序配置 ..... 693
表 25-15. 同步复用模式写时序配置 ..... 695
表 25-16. SPI/QPI 接口 ..... 695
表 25-17. 8 位/16 位 NAND 接口信号描述 ..... 697
表 25-18. 16 位 PC Card 接口信号描述 ..... 698
表 25-19. Bank1/2/3 支持的访问模式 ..... 698
表 25-20. NADN/PC Card 可编程参数 ..... 699
表 25-21. SDRAM 命令真值表 ..... 704
表 25-22. SDRAM IO 定义 ..... 704
表 26-1. 32-bit 过滤序号 ..... 739
表 26-2. 过滤索引 ..... 740
表 26-3. CAN 事件/中断标志 ..... 744

表 27-1. 以太网信号(MII 模式)..... 766
表 27-2.以太网信号(RMII 模式)..... 767
表 27-3. 时钟范围..... 769
表 27-4. 接收接口信号编码..... 770
表 27-5. 目标地址过滤器结果列表..... 775
表 27-6. 源地址过滤器结果列表..... 776
表 27-7. 接收描述符 0 错误状态描述，仅适用于常规描述符(DFM=0)..... 797
表 27-8. 支持的 PTP 时间戳及其寄存器配置..... 838
表 28-1. USBFS 信号线描述..... 860
表 28-2. USBFS 全局中断..... 871
表 29-1. USBHS 信号线描述..... 927
表 29-2. USBHS 支持速度列表..... 928
表 29-3. USBHS 全局中断..... 941
表 30-1. 版本历史..... 1005

# 1. 系统及存储器架构

GD32A490xx系列器件是基于Arm® Cortex®-M4处理器的32位通用微控制器。Arm® Cortex®-M4处理器包括三条AHB总线分别称为I-CODE总线、D-Code总线和系统总线。Cortex®-M4处理器的所有存储访问，根据不同的目的和目标存储空间，都会在这三条总线上执行。存储器的组织采用了哈佛结构，预先定义的存储器映射和高达4 GB的存储空间，充分保证了系统的灵活性和可扩展性。

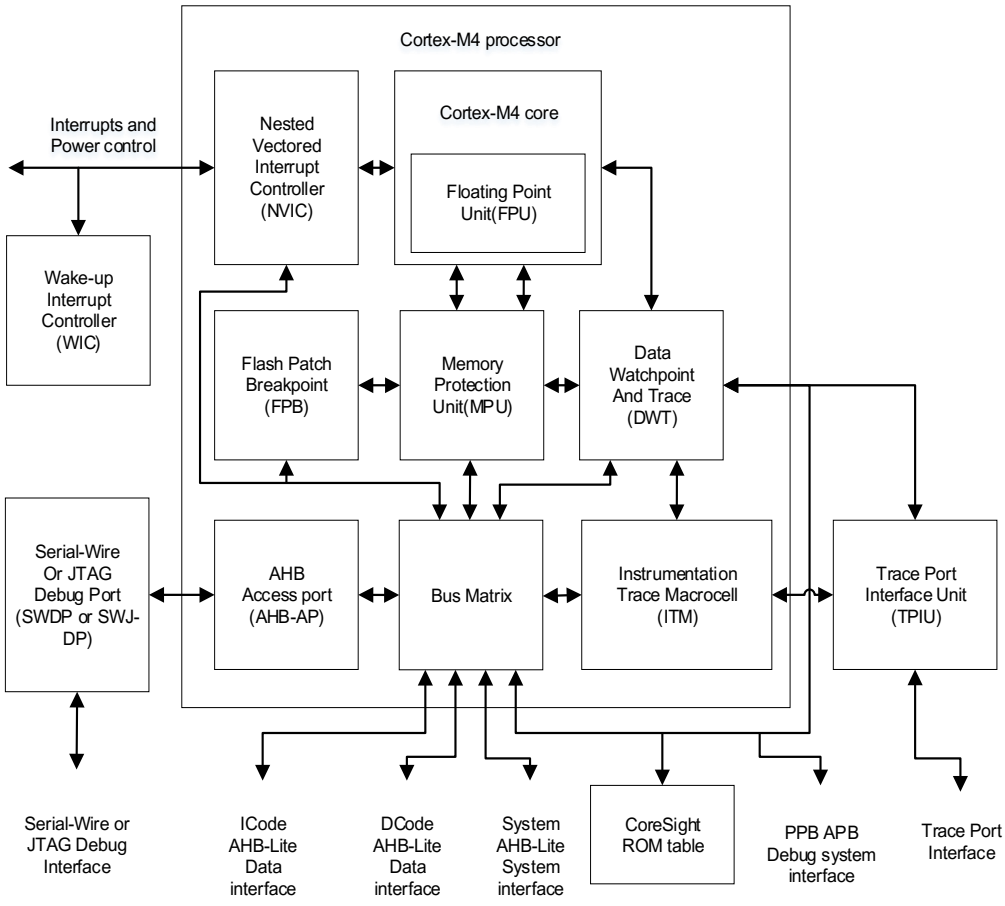
## 1.1. Arm® Cortex®-M4 处理器

Cortex®-M4处理器是一个具有浮点运算功能、低中断延迟时间和低成本调试特性的32位处理器。高集成度和增强的特性使Cortex®-M4处理器适合于那些需要高性能和低功耗微控制器的市场领域。Cortex®-M4处理器基于ARMv7架构，并且支持一个强大且可扩展的指令集，包括通用数据处理I / O控制任务、增强的数据处理位域操作、DSP(数字信号处理)和浮点运算指令。下面列出由Cortex®-M4提供的一些系统外设：

- 内部总线矩阵，用于实现I-Code总线、D-Code总线、系统总线、专用总线(PPB)以及调试专用总线(AHB-AP)的互联。
- 嵌套式向量型中断控制器(NVIC)。
- 闪存地址重载及断点单元(FPB)。
- 数据观测点及跟踪单元(DWT)。
- 指令跟踪宏单元(ITM)。
- 串行线和JTAG调试接口(SWJ-DP)。
- 跟踪端口接口单元(TPIU)。
- 内存保护单元(MPU)。
- 浮点运算单元(FPU)。

*图 1-1. Cortex®-M4 结构框图*显示了 Cortex®-M4 处理器结构框图。欲了解更多信息，请参阅 Arm® Cortex®-M4 技术参考手册。

**图 1-1. Cortex®-M4 结构框图**




## 1.2. 系统架构

GD32A490xx系列器件采用32位多层总线结构，该结构可使系统中的多个主机和从机之间的并行通信成为可能。多层总线结构包括一个AHB互联矩阵、两个AHB总线和两个APB总线。AHB互联矩阵的互联关系接下来将进行说明。在*表1-1. AHB互联矩阵的互联关系列表*中，"1"表示相应的主机可以通过AHB互联矩阵访问对应的从机，空白的单元格表示相应的主机不可以通过AHB互联矩阵访问对应的从机。

**表 1-1. AHB 互联矩阵的互联关系列表**

| | IBUS | DBUS | SBUS | DMA0M | DMA0P | DMA1M | DMA1P | ENET | TLI | USBHS | IPA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **FMC-I** | 1 | | | | | | | | | | |
| **FMC-D** | | 1 | | 1 | | 1 | 1 | 1 | 1 | 1 | 1 |
| **TCMSRAM** | | 1 | | | | | | | | | |

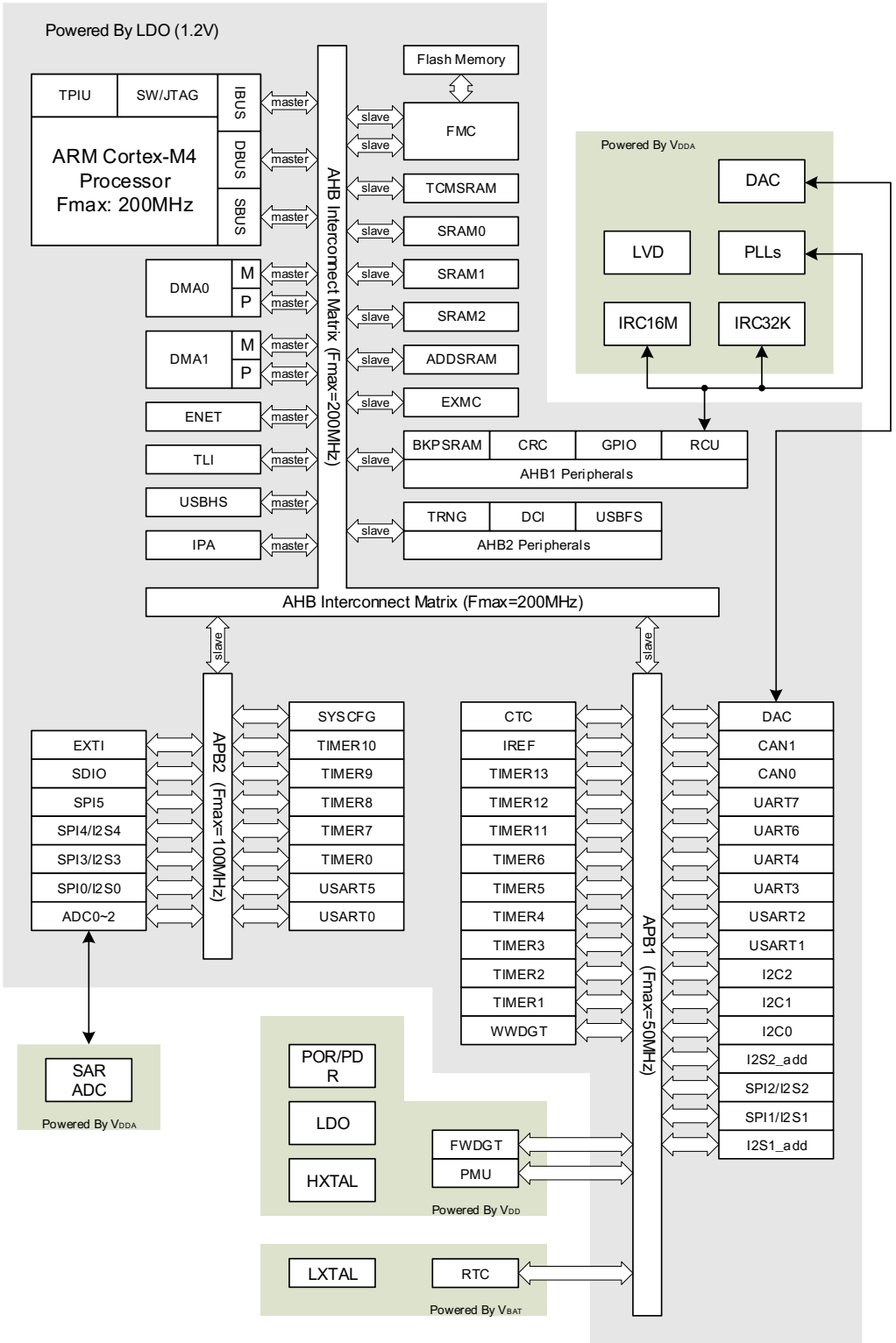
| | IBUS | DBUS | SBUS | DMA0M | DMA0P | DMA1M | DMA1P | ENET | TLI | USBHS | IPA |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **SRAM0** | 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 |
| **SRAM1** | | | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 |
| **SRAM2** | | | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 |
| **ADDSRAM** | 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 |
| **EXMC** | 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | 1 | 1 |
| **AHB1** | | | 1 | | 1 | 1 | 1 | | | | |
| **AHB2** | | | 1 | | | 1 | 1 | | | | |
| **APB1** | | | 1 | | 1 | 1 | 1 | | | | |
| **APB2** | | | 1 | | 1 | 1 | 1 | | | | |

如*表1-1. AHB互联矩阵的互联关系列表*所示，AHB互联矩阵共连接11个主机，分别为：IBUS、DBUS、SBUS、DMA0M、DMA0P、DMA1M、DMA1P、ENET、TLI、USBHS和IPA。IBUS是Cortex®-M4内核的指令总线，用于从代码区域(0x0000 0000-0x1FFF FFFF)中取指令和向量。DBUS是Cortex®-M4内核的数据总线，用于加载和存储数据，以及代码区域的调试访问。同样，SBUS是Cortex®-M4内核的系统总线，用于指令和向量获取、数据加载和存储以及系统区域的调试访问。系统区域包括内部SRAM区域和外设区域。DMA0M和DMA1M分别是DMA0和DMA1的存储器总线。DMA0P和DMA1P分别是DMA0和DMA1的外设总线。ENET是以太网。TLI是TFT LCD接口。USBHS是高速USB。IPA是图像处理加速器。

AHB互联矩阵也连接了12个从机，分别为：FMC-I、FMC-D、TCMSRAM、SRAM0、SRAM1、SRAM2、ADDSRAM、EXMC、AHB1、AHB2、APB1和APB2。FMC-I是闪存存储器控制器的指令总线，而FMC-D是闪存存储器的数据总线。TCMSRAM是紧耦合存储器SRAM，只可通过DBUS访问。SRAM0、SRAM1和SRAM2是片上静态随机存取存储器。ADDSRAM是附加的SRAM，仅在一些特殊的GD32A490xx器件中有效。EXMC是外部存储器控制器。AHB1和AHB2是连接所有AHB从机的两条AHB总线，而APB1和APB2是连接所有APB从机的两条APB总线。

GD32A490xx系列器件的系统架构如*图1-2. GD32A490xx器件的系统架构示意图*所示。

**图 1-2. GD32A490xx 器件的系统架构示意图**



## 1.3. 存储器映射

Arm® Cortex®-M4处理器采用哈佛结构，可以使用相互独立的总线来读取指令和加载/存储数据。指令代码和数据都位于相同的存储器地址空间，但在不同的地址范围。程序存储器，数据存储器，寄存器和I / O端口都在同一个线性的4 GB的地址空间之内。这是Cortex®-M4的最大地址范围，因为它的地址总线宽度是32位。此外，为了降低不同客户在相同应用时的软件复杂度，存储映射是按Cortex®-M4处理器提供的规则预先定义的。在存储器映射表中，一部分地址空间由Arm® Cortex®-M4的系统外设所占用，且不可更改。此外，其余部分地址空间可由芯片供应商定义使用。*表1-2. GD32A490xx系列器件的存储器映射表*显示了GD32A490xx系列器件的存储器映射，包括代码、SRAM、外设和其他预先定义的区域。几乎每个外设都分配了1KB的地址空间，这样可以简化每个外设的地址译码。

**表 1-2. GD32A490xx 系列器件的存储器映射表**

| 预先定义的地址空间 | 总线 | 地址范围 | 外设 |
|---|---|---|---|
| 外部设备 | AHB 互联矩阵 | 0xC000 0000 - 0xDFFF FFFF | EXMC - SDRAM |
| | | 0xA000 1000 - 0xBFFF FFFF | Reserved |
| | | 0xA000 0000 - 0xA000 0FFF | EXMC - SWREG |
| 外部 RAM | | 0x9000 0000 - 0x9FFF FFFF | EXMC - PC CARD |
| | | 0x7000 0000 - 0x8FFF FFFF | EXMC - NAND |
| | | 0x6000 0000 - 0x6FFF FFFF | EXMC -NOR / PSRAM / SRAM |
| 外设 | AHB2 | 0x5006 0C00 - 0x5FFF FFFF | Reserved |
| | | 0x5006 0800 - 0x5006 0BFF | TRNG |
| | | 0x5005 0400 - 0x5006 07FF | Reserved |
| | | 0x5005 0000 - 0x5005 03FF | DCI |
| | | 0x5004 0000 - 0x5004 FFFF | Reserved |
| | | 0x5000 0000 - 0x5003 FFFF | USBFS |
| | AHB1 | 0x4008 0000 - 0x4FFF FFFF | Reserved |
| | | 0x4004 0000 - 0x4007 FFFF | USBHS |
| | | 0x4002 BC00 - 0x4003 FFFF | Reserved |
| | | 0x4002 B000 - 0x4002 BBFF | IPA |
| | | 0x4002 A000 - 0x4002 AFFF | Reserved |
| | | 0x4002 8000 - 0x4002 9FFF | ENET |

| 预先定义的地址空间 | 总线 | 地址范围 | 外设 |
|---|---|---|---|
| | | 0x4002 6800 - 0x4002 7FFF | Reserved |
| | | 0x4002 6400 - 0x4002 67FF | DMA1 |
| | | 0x4002 6000 - 0x4002 63FF | DMA0 |
| | | 0x4002 5000 - 0x4002 5FFF | Reserved |
| | | 0x4002 4000 - 0x4002 4FFF | BKPSRAM |
| | | 0x4002 3C00 - 0x4002 3FFF | FMC |
| | | 0x4002 3800 - 0x4002 3BFF | RCU |
| | | 0x4002 3400 - 0x4002 37FF | Reserved |
| | | 0x4002 3000 - 0x4002 33FF | CRC |
| | | 0x4002 2400 - 0x4002 2FFF | Reserved |
| | | 0x4002 2000 - 0x4002 23FF | GPIOI |
| | | 0x4002 1C00 - 0x4002 1FFF | GPIOH |
| | | 0x4002 1800 - 0x4002 1BFF | GPIOG |
| | | 0x4002 1400 - 0x4002 17FF | GPIOF |
| | | 0x4002 1000 - 0x4002 13FF | GPIOE |
| | | 0x4002 0C00 - 0x4002 0FFF | GPIOD |
| | | 0x4002 0800 - 0x4002 0BFF | GPIOC |
| | | 0x4002 0400 - 0x4002 07FF | GPIOB |
| | | 0x4002 0000 - 0x4002 03FF | GPIOA |
| | APB2 | 0x4001 6C00 - 0x4001 FFFF | Reserved |
| | | 0x4001 6800 - 0x4001 6BFF | TLI |
| | | 0x4001 5800 - 0x4001 67FF | Reserved |
| | | 0x4001 5400 - 0x4001 57FF | SPI5 |
| | | 0x4001 5000 - 0x4001 53FF | SPI4/I2S4 |
| | | 0x4001 4C00 - 0x4001 4FFF | Reserved |
| | | 0x4001 4800 - 0x4001 4BFF | TIMER10 |
| | | 0x4001 4400 - 0x4001 47FF | TIMER9 |
| | | 0x4001 4000 - 0x4001 43FF | TIMER8 |
| | | 0x4001 3C00 - 0x4001 3FFF | EXTI |
| | | 0x4001 3800 - 0x4001 3BFF | SYSCFG |
| | | 0x4001 3400 - 0x4001 37FF | SPI3 / I2S3 |
| | | 0x4001 3000 - 0x4001 33FF | SPI0 / I2S0 |
| | | 0x4001 2C00 - 0x4001 2FFF | SDIO |
| | | 0x4001 2400 - 0x4001 2BFF | Reserved |
| | | 0x4001 2000 - 0x4001 23FF | ADC |
| | | 0x4001 1800 - 0x4001 1FFF | Reserved |
| | | 0x4001 1400 - 0x4001 17FF | USART5 |
| | | 0x4001 1000 - 0x4001 13FF | USART0 |
| | | 0x4001 0800 - 0x4001 0FFF | Reserved |

| 预先定义的地址空间 | 总线 | 地址范围 | 外设 |
|---|---|---|---|
| | | 0x4001 0400 - 0x4001 07FF | TIMER7 |
| | | 0x4001 0000 - 0x4001 03FF | TIMER0 |
| | APB1 | 0x4000 C800 - 0x4000 FFFF | Reserved |
| | | 0x4000 C400 - 0x4000 C7FF | IREF |
| | | 0x4000 8000 - 0x4000 C3FF | Reserved |
| | | 0x4000 7C00 - 0x4000 7FFF | UART7 |
| | | 0x4000 7800 - 0x4000 7BFF | UART6 |
| | | 0x4000 7400 - 0x4000 77FF | DAC |
| | | 0x4000 7000 - 0x4000 73FF | PMU |
| | | 0x4000 6C00 - 0x4000 6FFF | CTC |
| | | 0x4000 6800 - 0x4000 6BFF | CAN1 |
| | | 0x4000 6400 - 0x4000 67FF | CAN0 |
| | | 0x4000 6000 - 0x4000 63FF | Reserved |
| | | 0x4000 5C00 - 0x4000 5FFF | I2C2 |
| | | 0x4000 5800 - 0x4000 5BFF | I2C1 |
| | | 0x4000 5400 - 0x4000 57FF | I2C0 |
| | | 0x4000 5000 - 0x4000 53FF | UART4 |
| | | 0x4000 4C00 - 0x4000 4FFF | UART3 |
| | | 0x4000 4800 - 0x4000 4BFF | USART2 |
| | | 0x4000 4400 - 0x4000 47FF | USART1 |
| | | 0x4000 4000 - 0x4000 43FF | I2S2_add |
| | | 0x4000 3C00 - 0x4000 3FFF | SPI2/I2S2 |
| | | 0x4000 3800 - 0x4000 3BFF | SPI1/I2S1 |
| | | 0x4000 3400 - 0x4000 37FF | I2S1_add |
| | | 0x4000 3000 - 0x4000 33FF | FWDGT |
| | | 0x4000 2C00 - 0x4000 2FFF | WWDGT |
| | | 0x4000 2800 - 0x4000 2BFF | RTC |
| | | 0x4000 2400 - 0x4000 27FF | Reserved |
| | | 0x4000 2000 - 0x4000 23FF | TIMER13 |
| | | 0x4000 1C00 - 0x4000 1FFF | TIMER12 |
| | | 0x4000 1800 - 0x4000 1BFF | TIMER11 |
| | | 0x4000 1400 - 0x4000 17FF | TIMER6 |
| | | 0x4000 1000 - 0x4000 13FF | TIMER5 |
| | | 0x4000 0C00 - 0x4000 0FFF | TIMER4 |
| | | 0x4000 0800 - 0x4000 0BFF | TIMER3 |
| | | 0x4000 0400 - 0x4000 07FF | TIMER2 |
| | | 0x4000 0000 - 0x4000 03FF | TIMER1 |
| SRAM | AHB 互联矩阵 | 0x200B 0000 - 0x3FFF FFFF | Reserved |
| | | 0x2003 0000 - 0x200A FFFF | ADDSRAM (512KB) |

| 预先定义的地址空间 | 总线 | 地址范围 | 外设 |
|---|---|---|---|
| | | 0x2002 0000 - 0x2002 FFFF | SRAM2 (64KB) |
| | | 0x2001 C000 - 0x2001 FFFF | SRAM1 (16KB) |
| | | 0x2000 0000 - 0x2001 BFFF | SRAM0 (112KB) |
| 代码 | AHB 互联矩阵 | 0x1FFF C010 - 0x1FFF FFFF | Reserved |
| | | 0x1FFF C000 - 0x1FFF C00F | Option bytes (Bank 0) |
| | | 0x1FFF 7A10 - 0x1FFF BFFF | Reserved |
| | | 0x1FFF 7800 - 0x1FFF 7A0F | OTP (512B) |
| | | 0x1FFF 0000 - 0x1FFF 77FF | Boot loader (30KB) |
| | | 0x1FFE C010 - 0x1FFE FFFF | Reserved |
| | | 0x1FFE C000 - 0x1FFE C00F | Option bytes (Bank 1) |
| | | 0x1001 0000 - 0x1FFE BFFF | Reserved |
| | | 0x1000 0000 - 0x1000 FFFF | TCMSRAM (64KB) |
| | | 0x0830 0000 - 0x0FFF FFFF | Reserved |
| | | 0x0800 0000 - 0x082F FFFF | Main Flash (3072KB) |
| | | 0x0000 0000 - 0x07FF FFFF | Aliased to the boot device |

### 1.3.1. 位带操作

为了减少"读-改-写"操作的次数，Cortex®-M4处理器提供了一个可以执行单原子比特操作的位带功能。存储器映射包含了两个支持位带操作的区域。其中一个是SRAM区的最低1MB范围，第二个是片内外设区的最低1MB范围。这两个区域中的地址除了普通应用外，还有自己的"位带别名区"。位带别名区把每个比特扩展成一个32位的字。当用户访问位带别名区时，就可以达到访问原始比特的目的。

下面的公式表明了位带别名区中的每个字如何对应位带区的相应比特或目标比特。

$$bit_word_addr = bit_band_base + (byte_offset \times 32) + (bit_number \times 4) \quad (1\text{-}1)$$

其中：

- bit_word_addr指的是位带区目标比特对应在位带别名区的地址；
- bit_band_base指的是位带别名区的起始地址；
- byte_offset指的是位带区目标比特所在的字节的字节地址偏移量；
- bit_number指的是目标比特在对应字节中的位置(0-7)。

例如，要想访问0x2000 0200地址的第7位，可访问的位带别名区地址是：

$$bit_word_addr = 0x2200\ 0000 + (0x200 * 32) + (7 * 4) = 0x2200\ 401C \quad (1\text{-}2)$$

如果对0x2200 401C进行写操作，那么0x2000 0200的第7位将会相应变化；如果对0x2200 401C进行读操作，那么视0x2000 0200的第7位状态而返回0x01或0x00。

### 1.3.2. 片上 SRAM 存储器

GD32A490xx系列微控制器含有高达256KB片上SRAM、4KB备份SRAM和512KB附加SRAM(ADDSRAM)，所有的SRAM均支持字节、半字(16比特)和整字(32比特)访问。片上SRAM可分为4块，分别为SRAM0(112KB)、SRAM1(16KB)、SRAM2(64KB)和TCMSRAM(64KB)。SRAM0、SRAM1和SRAM2可以被所有的AHB主机访问，然而，TCMSRAM(紧耦合存储器SRAM)只可被Cortex®-M4内核的数据总线访问。BKPSRAM(备份SRAM)应用于备份域，即使当VDD供电电源掉电时，该SRAM仍可保持其内容。附加SRAM只在一些特殊的GD32A490xx器件中可用。由于采用AHB互联矩阵，上述SRAM块可以同时被不同的AHB主机访问，例如，即使CPU正在访问SRAM0，USBHS也可以访问SRAM1。

### 1.3.3. 片上 FLASH 存储器概述

GD32A490xx系列微控制器可以提供高密度片上FLASH存储器，按以下分类进行组织：

- 高达3072KB主FLASH存储器。
- 高达30KB引导装载程序(boot loader)信息块存储器。
- 高达512B OTP(一次性可编程)存储器。
- 器件配置的选项字节。

更多详细说明请参考闪存控制器(FMC)章节。

## 1.4. 引导配置

GD32A490xx系列微控制器提供了三种引导源，可以通过BOOT0和BOOT1引脚来进行选择，详细说明见*表1-3. 引导模式*。该两个引脚的电平状态会在复位后的第四个CK_SYS(系统时钟)的上升沿进行锁存。用户可自行选择所需要的引导源，通过设置上电复位和系统复位后的BOOT0和BOOT1的引脚电平。一旦这两个引脚电平被采样，它们可以被释放并用于其他用途。

**表 1-3. 引导模式**

| 引导源选择 | 启动模式选择引脚 | |
|---|---|---|
| | Boot1 | Boot0 |
| 主FLASH存储器 | x | 0 |
| 引导装载程序 | 0 | 1 |
| 片上SRAM | 1 | 1 |

上电序列或系统复位后，ARM® Cortex®-M4处理器先从0x0000 0000地址获取栈顶值，再从

0x0000 0004地址获得引导代码的基地址，然后从引导代码的基地址开始执行程序。

所选引导源对应的存储空间会被映射到引导存储空间，即从0x0000 0000开始的地址空间。如果片上SRAM(开始于0x2000 0000的存储空间)被选为引导源，用户必须在应用程序初始化代码中通过修改NVIC异常向量表和偏移地址将向量表重置到SRAM中。当主FLASH存储器被选择作为引导源，从0x0800 0000开始的存储空间会被映射到引导存储空间。由于主FLASH存储器的Bank0或Bank1均可映射到地址0x0800 0000(通过配置SYSCFG_CFG0寄存器的FMC_SWP控制位，具体参考*1.5.1节*)，所以，微控制器可以使用该方法从Bank0或Bank1中启动。

为了使能引导块功能，选项字节中的BB控制位需要被置位。当该控制位被置位并且主FLASH存储器被选择作为引导源，微控制器从引导装载程序中启动并且引导装载程序跳至主FLASH存储器的Bank1中执行代码。在应用程序初始化代码中，用户必须通过修改NVIC异常向量表和偏移地址将向量表重置到Bank1基地址。

引导装载程序在生产器件的过程中已经被编程，用于通过以下其中一个通信接口重新编程主FLASH存储器：USART0、USART2以及USBFS。

**注意**：为确保系统的可靠性和稳定性，在硬件设计阶段，应查阅对应型号的数据手册的Boot mode章节，确认Bootloader 所使用的引脚上避免连接可能产生干扰信号的外设（如传感器输出、看门狗芯片、按键等），以免影响 Bootloader 的正常功能，和对所连接的外设产生异常输出。

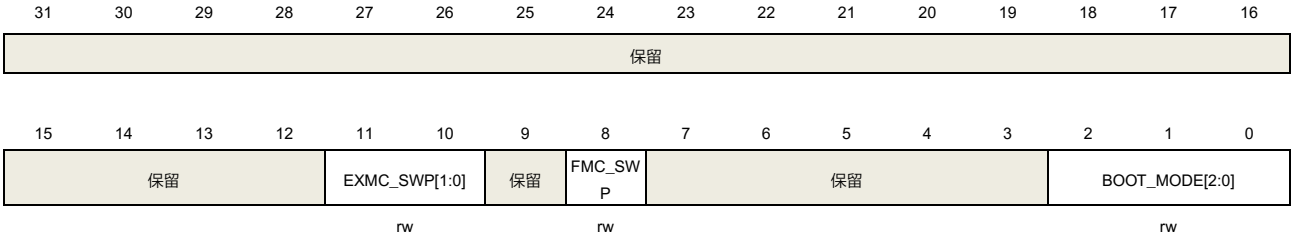
## 1.5. 系统配置寄存器(SYSCFG)

SYSCFG基地址：0x4001 3800

### 1.5.1. 配置寄存器 0(SYSCFG_CFG0)

地址偏移：0x00
复位值：0x0000 000X(根据BOOT0和BOOT1引脚的状态，X表示BOOT_MODE[1:0]，可能为任意值)

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11:10 | EXMC_SWP[1:0] | EXMC存储器映射切换<br>这些位控制在EXMC存储器中地址映射切换功能<br>00：无存储器映射切换；<br>01：SDRAM的Bank0和Bank1与NAND Bank1和PC CARD进行切换，然后，SDRAM的Bank0和Bank1被映射到从0x8000 0000到0x9FFF FFFF的地址范围，NAND的Bank1被映射到从0xC000 0000到0xCFFF FFFF的地址范围，PC CARD被映射到从0xD000 0000到0xDFFF FFFF的地址范围。<br>其他配置保留 |
| 9 | 保留 | 必须保持复位值。 |
| 8 | FMC_SWP | FMC存储器映射切换<br>这些位控制主FLASH存储器的Bank0和Bank1的地址映射切换功能<br>0：主FLASH存储器的Bank0被映射到地址0x0800 0000，主FLASH存储器的Bank1被映射到地址0x0810 0000<br>1：主FLASH存储器的Bank1映射到地址0x0800 0000，主FLASH存储器的Bank0映射到地址0x0810 0000 |
| 7:3 | 保留 | 必须保持复位值。 |
| 2:0 | BOOT_MODE[2:0] | 引导模式(详细请参考*1.4节*Boot配置)<br>这些位选择在地址0x0000 0000的访问值，复位之后，根据BOOT0和BOOT1引脚的配置通过下表获取初始值： |

| BOOT1 | BOOT0 | BOOT_MODE[2:0]的复位值 | 系统启动所选存储器 |
|---|---|---|---|
| x | 0 | 000 | 主FLASH存储器 |
| 0 | 1 | 001 | 引导装载代码所在系统存储器 |
| 1 | 1 | 011 | 片上SRAM |

通过软件配置可以选择更多的器件，一旦这些控制位通过软件写入，BOOT0和BOOT1引脚的电平状态将会被忽略。

000：主FLASH存储器(0x0800 0000~0x08FF FFFF)被映射到地址0x0000 0000

001：引导装载代码所在系统存储器(0x1FFF 0000~0x1FFF 7FFF)被映射到地址0x0000 0000

010：EXMC的SRAM/NOR 0和1(0x6000 0000~0x67FF FFFF)被映射到地址0x0000 0000

011：片上SRAM的SRAM0(0x2000 0000~0x2001 BFFF)被映射到地址0x0000 0000

100：EXMC的SDRAM Device0(0xC000 0000~0xC7FF FFFF)被映射到地址0x0000 0000

其他配置保留

**注意**：即使映射到地址0x0000 0000，相应存储器仍可通过原始存储空间进行访问。

### 1.5.2. 配置寄存器 1(SYSCFG_CFG1)

地址偏移：0x04

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值。 |
| 23 | ENET_PHY_SEL | 以太网PHY选择<br>这些位为以太网MAC选择PHY接口<br>当以太网MAC在复位状态下，且在MAC时钟使能之前，这些控制位必须被配置<br>0：选择MII<br>1：选择RMII |
| 22:0 | 保留 | 必须保持复位值。 |

### 1.5.3. EXTI 源选择寄存器 0(SYSCFG_EXTISS0)

地址偏移：0x08

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EXTI3_SS [3:0] | | | | EXTI2_SS [3:0] | | | | EXTI1_SS [3:0] | | | | EXTI0_SS [3:0] | | | |
| rw | | | | rw | | | | rw | | | | rw | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:12 | EXTI3_SS[3:0] | EXTI 3源选择<br>0000：PA3引脚<br>0001：PB3引脚<br>0010：PC3引脚<br>0011：PD3引脚<br>0100：PE3引脚<br>0101：PF3引脚<br>0110：PG3引脚<br>0111：PH3引脚<br>1000：PI3引脚<br>其他配置保留 |
| 11:8 | EXTI2_SS[3:0] | EXTI 2源选择<br>0000：PA2引脚<br>0001：PB2引脚<br>0010：PC2引脚<br>0011：PD2引脚<br>0100：PE2引脚<br>0101：PF2引脚<br>0110：PG2引脚<br>0111：PH2引脚<br>1000：PI2引脚<br>其他配置保留 |
| 7:4 | EXTI1_SS[3:0] | EXTI 1源选择<br>0000：PA1引脚<br>0001：PB1引脚<br>0010：PC1引脚<br>0011：PD1引脚<br>0100：PE1引脚 |

| | | |
|---|---|---|
| | | 0101：PF1引脚<br>0110：PG1引脚<br>0111：PH1引脚<br>1000：PI1引脚<br>其他配置保留 |
| 3:0 | EXTI0_SS[3:0] | EXTI 0 源选择<br>0000：PA0引脚<br>0001：PB0引脚<br>0010：PC0引脚<br>0011：PD0引脚<br>0100：PE0引脚<br>0101：PF0引脚<br>0110：PG0引脚<br>0111：PH0引脚<br>1000：PI0引脚<br>其他配置保留 |

### 1.5.4. EXTI 源选择寄存器 1(SYSCFG_EXTISS1)

地址偏移：0x0C

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EXTI7_SS[3:0] | | | | EXTI6_SS[3:0] | | | | EXTI5_SS[3:0] | | | | EXTI4_SS[3:0] | | | |
| rw | | | | rw | | | | rw | | | | rw | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:12 | EXTI7_SS[3:0] | EXTI 7源选择<br>0000：PA7引脚<br>0001：PB7引脚<br>0010：PC7引脚<br>0011：PD7引脚<br>0100：PE7引脚<br>0101：PF7引脚<br>0110：PG7引脚<br>0111：PH7引脚<br>1000：PI7引脚<br>其他配置保留 |

| | | |
|---|---|---|
| 11:8 | EXTI6_SS[3:0] | EXTI 6源选择<br>0000：PA6引脚<br>0001：PB6引脚<br>0010：PC6引脚<br>0011：PD6引脚<br>0100：PE6引脚<br>0101：PF6引脚<br>0110：PG6引脚<br>0111：PH6引脚<br>1000：PI6引脚<br>其他配置保留 |
| 7:4 | EXTI5_SS[3:0] | EXTI 5源选择<br>0000：PA5引脚<br>0001：PB5引脚<br>0010：PC5引脚<br>0011：PD5引脚<br>0100：PE5引脚<br>0101：PF5引脚<br>0110：PG5引脚<br>0111：PH5引脚<br>1000：PI5引脚<br>其他配置保留 |
| 3:0 | EXTI4_SS[3:0] | EXTI 4源选择<br>0000：PA4引脚<br>0001：PB4引脚<br>0010：PC4引脚<br>0011：PD4引脚<br>0100：PE4引脚<br>0101：PF4引脚<br>0110：PG4引脚<br>0111：PH4引脚<br>1000：PI4引脚<br>其他配置保留 |

### 1.5.5. EXTI 源选择寄存器 2(SYSCFG_EXTISS2)

地址偏移：0x10

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EXTI11_SS[3:0] | | | | EXTI10_SS[3:0] | | | | EXTI9_SS[3:0] | | | | EXTI8_SS[3:0] | | | |
| rw | | | | rw | | | | rw | | | | rw | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:12 | EXTI11_SS[3:0] | EXTI 11源选择<br>0000：PA11引脚<br>0001：PB11引脚<br>0010：PC11引脚<br>0011：PD11引脚<br>0100：PE11引脚<br>0101：PF11引脚<br>0110：PG11引脚<br>0111：PH11引脚<br>1000：PI11引脚<br>其他配置保留 |
| 11:8 | EXTI10_SS[3:0] | EXTI 10源选择<br>0000：PA10引脚<br>0001：PB10引脚<br>0010：PC10引脚<br>0011：PD10引脚<br>0100：PE10引脚<br>0101：PF10引脚<br>0110：PG10引脚<br>0111：PH10引脚<br>1000：PI10引脚<br>其他配置保留 |
| 7:4 | EXTI9_SS[3:0] | EXTI 9源选择<br>0000：PA9引脚<br>0001：PB9引脚<br>0010：PC9引脚<br>0011：PD9引脚<br>0100：PE9引脚<br>0101：PF9引脚<br>0110：PG9引脚<br>0111：PH9引脚<br>1000：PI9引脚<br>其他配置保留 |
| 3:0 | EXTI8_SS[3:0] | EXTI 8源选择<br>0000：PA8引脚<br>0001：PB8引脚 |

0010：PC8引脚
0011：PD8引脚
0100：PE8引脚
0101：PF8引脚
0110：PG8引脚
0111：PH8引脚
1000：PI8引脚
其他配置保留

### 1.5.6. EXTI 源选择寄存器 3(SYSCFG_EXTISS3)

地址偏移：0x14
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EXTI15_SS[3:0] | | | | EXTI14_SS[3:0] | | | | EXTI13_SS[3:0] | | | | EXTI12_SS[3:0] | | | |
| rw | | | | rw | | | | rw | | | | rw | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:12 | EXTI15_SS[3:0] | EXTI 15源选择<br>0000：PA15引脚<br>0001：PB15引脚<br>0010：PC15引脚<br>0011：PD15引脚<br>0100：PE15引脚<br>0101：PF15引脚<br>0110：PG15引脚<br>0111：PH15引脚<br>其他配置保留 |
| 11:8 | EXTI14_SS[3:0] | EXTI 14源选择<br>0000：PA14引脚<br>0001：PB14引脚<br>0010：PC14引脚<br>0011：PD14引脚<br>0100：PE14引脚<br>0101：PF14引脚<br>0110：PG14引脚<br>0111：PH14引脚 |

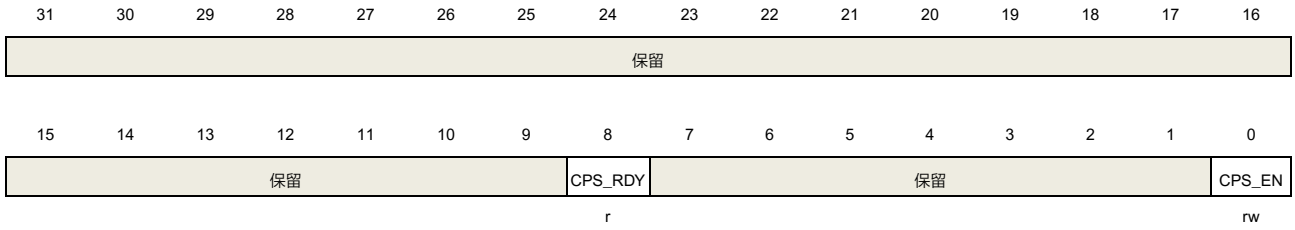
| | | |
|---|---|---|
| | | 其他配置保留 |
| 7:4 | EXTI13_SS[3:0] | EXTI 13源选择<br>0000：PA13引脚<br>0001：PB13引脚<br>0010：PC13引脚<br>0011：PD13引脚<br>0100：PE13引脚<br>0101：PF13引脚<br>0110：PG13引脚<br>0111：PH13引脚<br>其他配置保留 |
| 3:0 | EXTI12_SS[3:0] | EXTI 12源选择<br>0000：PA12引脚<br>0001：PB12引脚<br>0010：PC12引脚<br>0011：PD12引脚<br>0100：PE12引脚<br>0101：PF12引脚<br>0110：PG12引脚<br>0111：PH12引脚<br>其他配置保留 |

### 1.5.7. I/O 补偿控制寄存器(SYSCFG_CPSCTL)

地址偏移：0x20

复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:9 | 保留 | 必须保持复位值。 |
| 8 | CPS_RDY | I / O补偿单元是否准备好<br>该位只读<br>0：I / O补偿单元没有准备好<br>1：I / O补偿单元准备好 |
| 7:1 | 保留 | 必须保持复位值 |

| | | |
|---|---|---|
| 0 | CPS_EN | I / O补偿单元使能<br>0：I / O补偿单元掉电<br>1：I / O补偿单元使能 |

## 1.6. 设备电子签名

设备的电子签名中包含的存储容量信息和96位的唯一设备ID。96位唯一设备ID对于每颗芯片而言都是唯一的。它可以用作序列号，或安全密钥的一部分等等。

### 1.6.1. 存储容量信息

基地址：0x1FFF 7A20
该值是原厂设定的，不能由用户修改。

该寄存器只能按字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | FLASH_DENSITY [15:0] | Flash存储器容量<br>该值表明芯片的片上FLASH存储器容量，以Kbytes为单位<br>例如：0x0020表示32Kbytes。 |
| 15:0 | SRAM_DENSITY [15:0] | SRAM 存储器容量<br>该值表明芯片的片上SRAM容量，以Kbytes为单位<br>例如：0x0008表示8Kbytes。 |

### 1.6.2. 设备唯一 ID(96 位/位域)

基地址：0x1FFF 7A10
该值是原厂设定的，不能由用户修改。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|

| | | |
|---|---|---|
| 31:0 | UNIQUE_ID[31:0] | 设备唯一ID |

基地址：0x1FFF 7A14
该值是原厂设定的，不能由用户修改。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| UNIQUE_ID[63:48] | | | | | | | | | | | | | | | |
| r | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| UNIQUE_ID[47:32] | | | | | | | | | | | | | | | |
| r | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | UNIQUE_ID[63:32] | 设备唯一ID |

基地址：0x1FFF 7A18
该值是原厂设定的，不能由用户修改。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| UNIQUE_ID[95:80] | | | | | | | | | | | | | | | |
| r | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| UNIQUE_ID[79:64] | | | | | | | | | | | | | | | |
| r | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | UNIQUE_ID[95:64] | 设备唯一设备ID |

# 2. 闪存控制器(FMC)

## 2.1. 简介

闪存控制器(FMC)，提供了片上闪存需要的所有功能。MCU执行指令零等待的区域最大支持到1024K字节空间。FMC也提供了页(4KB)擦除操作、扇区擦除和整片擦除，以及32位整字或16位半字、字节编程等闪存操作。

## 2.2. 主要特征

- 高达3072K字节的片上闪存可用于存储指令或数据；
- MCU执行指令零等待区域最大支持到1024K字节空间(在闪存大小等于256K或512K时，闪存全片执行指令零等待)；在此范围外，CPU读取指令存在较长延时；
- 对于GD32A490xx，使用了两片闪存；前1024KB容量在第0片闪存(bank0)中，后续的容量在第1片闪存(bank1)中；
- 支持32位整字或16位半字、字节编程，页(4KB)擦除，扇区擦除和整片擦除操作；
- 2个大小为16字节的选项字节可根据用户需求配置；
- 512字节OTP块和16字节OTP锁定块(一次性编程)，用于存储用户数据；
- 30K字节信息块(一次性编程)，用于引导装载程序；
- 选项字节会在每次系统复位时装载到选项字节控制寄存器；
- 具有安全保护状态，可阻止对代码或数据的非法读访问；
- 具有擦除和编程保护状态，可阻止意外写操作。

## 2.3. 功能说明

### 2.3.1. 闪存结构

对于主存储闪存容量不多于3072KB的GD32A490xx，包含8个16KB的扇区、2个64KB的扇区、14个128KB的扇区、4个256KB的扇区。主存储闪存的每个扇区都可以单独擦除。

闪存结构细节见*表2-1. GD32A490xx闪存基地址和构成*。

**表 2-1. GD32A490xx 闪存基地址和构成**

| 闪存块 | | 名称 | 地址范围 | 大小(字节) |
|---|---|---|---|---|
| 主存储闪存块 | 块0<br>1MB | 扇区0 | 0x0800 0000 - 0x0800 3FFF | 16KB |
| | | 扇区1 | 0x0800 4000 - 0x0800 7FFF | 16KB |
| | | 扇区2 | 0x0800 8000 - 0x0800 BFFF | 16KB |
| | | 扇区3 | 0x0800 C000 - 0x0800 | 16KB |

| 闪存块 | | 名称 | 地址范围 | 大小(字节) |
|---|---|---|---|---|
| | | | FFFF | |
| | | 扇区4 | 0x0801 0000 - 0x0801 FFFF | 64KB |
| | | 扇区5 | 0x0802 0000 - 0x0803 FFFF | 128KB |
| | | 扇区6 | 0x0804 0000 - 0x0805 FFFF | 128KB |
| | | . . . | . . . | . . . |
| | | 扇区11 | 0x080E 0000 - 0x080F FFFF | 128KB |
| | 块1 2816KB | 扇区12 | 0x0810 0000 - 0x0810 3FFF | 16KB |
| | | 扇区13 | 0x0810 4000 - 0x0810 7FFF | 16KB |
| | | 扇区14 | 0x0810 8000 - 0x0810 BFFF | 16KB |
| | | 扇区15 | 0x0810 C000 - 0x0810 FFFF | 16KB |
| | | 扇区16 | 0x0811 0000 - 0x0811 FFFF | 64KB |
| | | 扇区17 | 0x0812 0000 - 0x0813 FFFF | 128KB |
| | | 扇区18 | 0x0814 0000 - 0x0815 FFFF | 128KB |
| | | . . . | . . . | . . . |
| | | 扇区23 | 0x081E 0000 - 0x081F FFFF | 128KB |
| | | 扇区24 | 0x08200000 - 0x0823 FFFF | 256KB |
| | | 扇区25 | 0x0824 0000 - 0x0827 FFFF | 256KB |
| | | . . . | . . . | . . . |
| | | 扇区27 | 0x082C 0000 - 0x082F FFFF | 256KB |
| 信息块 | | 引导装载程序 | 0x1FFF 0000- 0x1FFF | 30KB |

| 闪存块 | 名称 | 地址范围 | 大小(字节) |
|---|---|---|---|
| | | 77FF | |
| 一次性编程块 | 一次性编程区域 | 0x1FFF 7800 - 0x1FFF 7A0F | 528B |
| 选项字节块 | 选项字节块0 | 0x1FFF C000 - 0x1FFF C00F | 16B |
| | 选项字节块1 | 0x1FFEC000 - 0x1FFEC00F | 16B |

**注意：**信息块存储了boot loader，不能被用户编程或擦除。

对于闪存大小为1M字节的类型，将选项字节中DBS位置1后的闪存扇区划分如下图*表2-2. DBS = 1时1M闪存扇区划分*，DBS的详细描述参考*选项字节说明*:

**表 2-2. DBS = 1 时 1M 闪存扇区划分**

| 闪存块 | | 名称 | 地址范围 | 大小(字节) |
|---|---|---|---|---|
| 主存储闪存块 | 块0 512KB | 扇区0 | 0x0800 0000 - 0x0800 3FFF | 16KB |
| | | 扇区1 | 0x0800 4000 - 0x0800 7FFF | 16KB |
| | | 扇区2 | 0x0800 8000 - 0x0800 BFFF | 16KB |
| | | 扇区3 | 0x0800 C000 - 0x0800 FFFF | 16KB |
| | | 扇区4 | 0x0801 0000 - 0x0801 FFFF | 64KB |
| | | 扇区5 | 0x0802 0000 - 0x0803 FFFF | 128KB |
| | | 扇区6 | 0x0804 0000 - 0x0805 FFFF | 128KB |
| | | 扇区7 | 0x0806 0000 - 0x0807 FFFF | 128KB |
| | 块1 512KB | 扇区12 | 0x0808 0000 - 0x0808 3FFF | 16KB |
| | | 扇区13 | 0x0808 4000 - 0x0808 7FFF | 16KB |
| | | 扇区14 | 0x0808 8000 - 0x0808 BFFF | 16KB |
| | | 扇区15 | 0x0808 C000 - 0x0808 FFFF | 16KB |
| | | 扇区16 | 0x0809 0000 - 0x0809 FFFF | 64KB |

| 闪存块 | | 名称 | 地址范围 | 大小(字节) |
|---|---|---|---|---|
| | | 扇区17 | 0x080A 0000 - 0x080B FFFF | 128KB |
| | | 扇区18 | 0x080C 0000 - 0x080D FFFF | 128KB |
| | | 扇区19 | 0x080E 0000 - 0x080F FFFF | 128KB |

### 2.3.2. 读操作

闪存可以像普通存储空间一样直接寻址访问。对闪存取指令和取数据分别使用CPU的IBUS或DBUS总线。

### 2.3.3. FMC_CTL/FMC_OBCTLx 寄存器解锁

复位后，FMC_CTL寄存器进入锁定状态，LK位置为1。通过先后向FMC_KEY寄存器写入0x45670123和0xCDEF89AB，可以使得FMC_CTL解锁。两次写操作后，FMC_CTL寄存器的LK位被硬件清0。可以通过软件设置FMC_CTL寄存器的LK位为1再次锁定FMC_CTL寄存器。任何对FMC_KEY寄存器的错误操作都会将LK位置1，从而锁定FMC_CTL寄存器，并引发一个总线错误。

FMC_OBCTL0寄存器，在FMC_CTL被解锁后仍然处于被保护状态。解锁过程为两次写操作，向FMC_OBKEY寄存器先后写入0x08192A3B和0x4C5D6E7F，然后硬件将FMC_OBCTL0寄存器中的OB_LK位清零。软件可以将FMC_OBCTLx的OB_LK位置1来锁定FMC_OBCTLx。

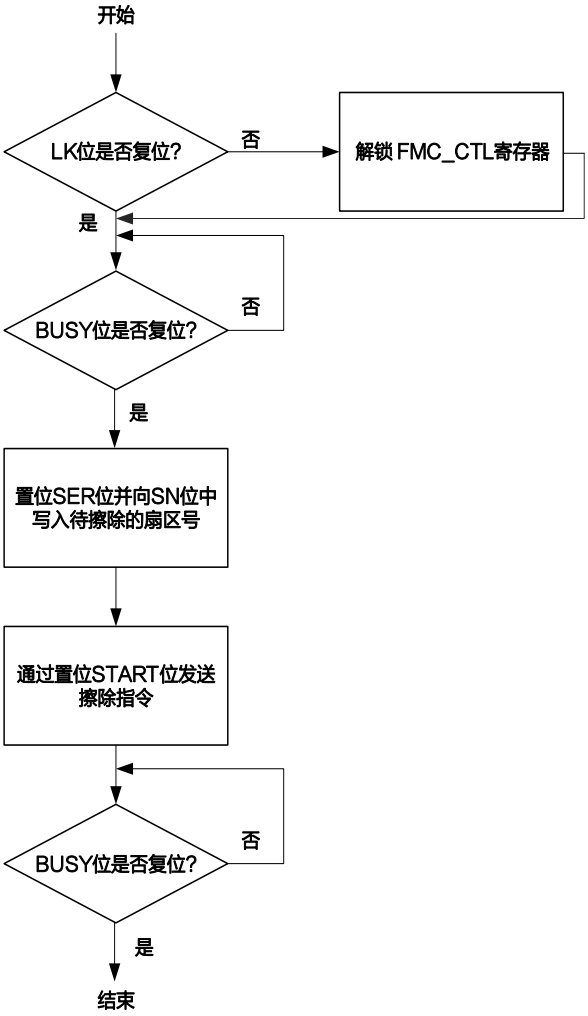
### 2.3.4. 页擦除

在GD3A490xx系类中，FMC额外提供了页擦除功能使得主存储闪存中大小为4K字节的页内容初始化为高电平。每一页都可以被独立擦除，而不影响其他页内容。FMC页擦除操作步骤如下：

1. 确保FMC_CTL寄存器不处于锁定状态；
2. 检查FMC_STAT寄存器的BUSY位来判定闪存是否正处于擦写访问状态，若BUSY位为1，则需等待该操作结束，BUSY位变为0；
3. 向FMC_PEKEY寄存器写入KEY值0xA9B8C7D6用以解锁FMC_PECFG寄存器；
4. 置位FMC_PECFG寄存器中的PE_EN位以使能页擦除功能；
5. 在FMC_PECFG寄存器中的PE_ADDR[28:0]位域中写入待擦除页的首地址，写入的页地址需要4K字节对齐；
6. 确保FMC_CTL寄存器中SN[4:0]位域为0，并置位寄存器中的SER位；
7. 通过将FMC_CTL寄存器的START位置1来发送扇区擦除命令到FMC；
8. 等待擦除指令执行完毕，FMC_STAT寄存器的BUSY位清0；
9. 清除PE_EN位和SER位防止下次误操作；
10. 如果需要，使用DBUS读操作验证该页是否擦除成功。

当页擦除成功执行，且FMC_CTL寄存器中的ENDIE位为1时，FMC_STAT寄存器的END位将置位，同时FMC将产生一个中断。需要注意的是，用户需确保写入的是正确的页地址(4K字节对齐)，否则当待擦除页被用来取指令或访问数据时，软件将会跑飞。该情况下，FMC不会提供任何出错通知。另一方面，对擦除/编程保护的扇区进行页擦除操作将无效。如果FMC_CTL寄存器的ERRIE位被置位，该操作将触发操作出错中断。中断服务程序可通过检测FMC_STAT寄存器的OPERR位来判断该中断是否发生。*图2-1. 页擦除操作流程*显示了扇区擦除操作流程。

**图 2-1. 页擦除操作流程**




## 2.3.5. 扇区擦除

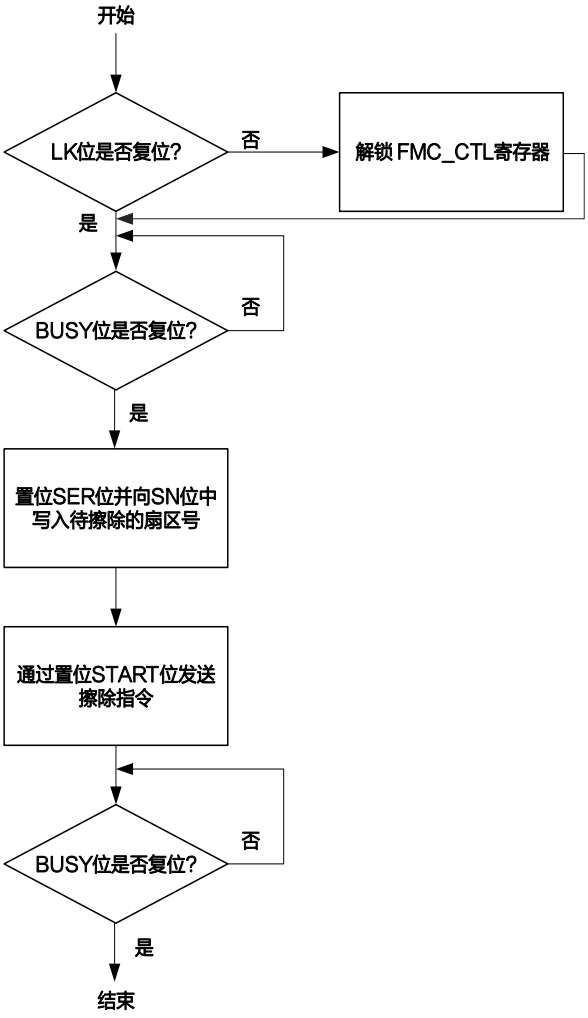
FMC的扇区擦除功能使得主存储闪存的扇区内容初始化为高电平。每一扇区都可以被独立擦除，而不影响其他扇区内容。FMC扇区擦除操作步骤如下：

1. 确保FMC_CTL寄存器不处于锁定状态；
2. 检查FMC_STAT寄存器的BUSY位来判定闪存是否正处于擦写访问状态，若BUSY位为1，则需等待该操作结束，BUSY位变为0；
3. 置位FMC_CTL寄存器的SER位；
4. 将待擦除扇区号写到FMC_CTL寄存器SN位；

5. 通过将FMC_CTL寄存器的START位置1来发送扇区擦除命令到FMC；
6. 等待擦除指令执行完毕，FMC_STAT寄存器的BUSY位清0；
7. 如果需要，使用DBUS读操作验证该扇区是否擦除成功。

当扇区擦除成功执行，且FMC_CTL寄存器中的ENDIE位为1时，FMC_STAT寄存器的END位将置位，同时FMC将产生一个中断。需要注意的是，用户需确保写入的是正确的擦除目标扇区号。否则当待擦除目标扇区被用来取指令或访问数据时，软件将会跑飞。该情况下，FMC不会提供任何出错通知。另一方面，对擦/编程保护的扇区进行扇区擦除操作将无效。如果FMC_CTL寄存器的ERRIE位被置位，该操作将触发操作出错中断。中断服务程序可通过检测FMC_STAT寄存器的OPERR位来判断该中断是否发生。*图2-2. 扇区擦除操作流程*显示了扇区擦除操作流程。

**图 2-2. 扇区擦除操作流程**




## 2.3.6. 整片擦除

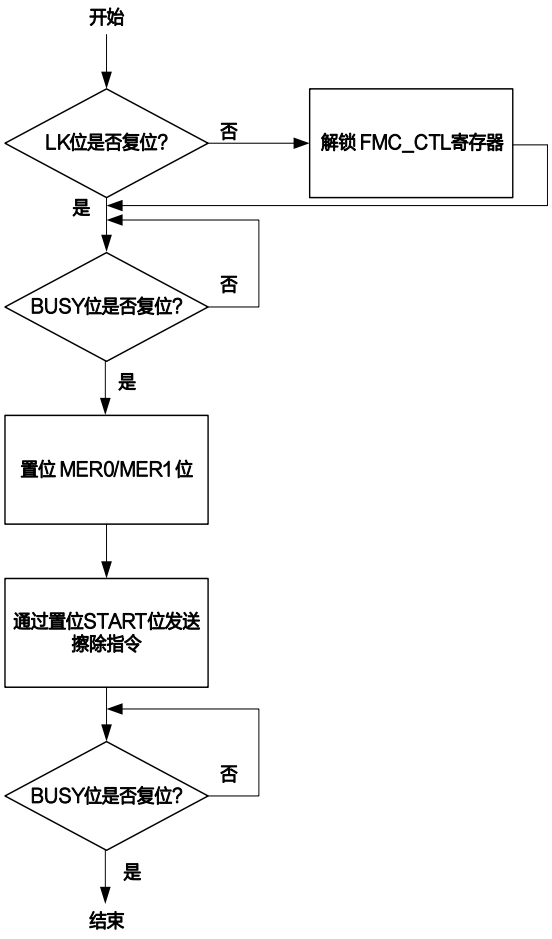
FMC提供了整片擦除功能可以初始化主存储闪存块的内容。当设置MER0为1时，擦除过程仅作用于Bank0，当设置MER1为1时，擦除过程仅作用于Bank1，当设置MER0和MER1为1时，擦除过程作用于整片闪存。FMC整片擦除操作步骤如下：

1. 确保FMC_CTL寄存器不处于锁定状态；
2. 等待FMC_STAT寄存器的BUSY位变为0来确保没有闪存操作在进行，否则等待该操作完成；
3. 置位FMC_CTL寄存器的MER0位，则单独擦除Bank0。置位FMC_CTL寄存器的MER1位，则单独擦除Bank1。同时置位FMC_CTL寄存器的MER0/MER1位，则擦除整片闪存；
4. 通过将FMC_CTL寄存器的START位置1来发送整片擦除命令到FMC；
5. 通过检查FMC_STAT寄存器的BUSY位是否清0，来确定擦除指令执行完毕；
6. 如果需要，使用DBUS读操作验证是否擦除成功。

当整片擦除成功执行，且FMC_CTL寄存器中的ENDIE位为1时，FMC_STAT寄存器的END位置位，同时FMC将产生一个中断。由于所有的闪存数据都将被复位为0xFFFF_FFFF，可以通过运行在SRAM中的程序或使用调试工具直接访问FMC寄存器来实现整片擦除操作。

*图2-3. 整片擦除操作流程*显示了整片擦除操作流程。

**图 2-3. 整片擦除操作流程**



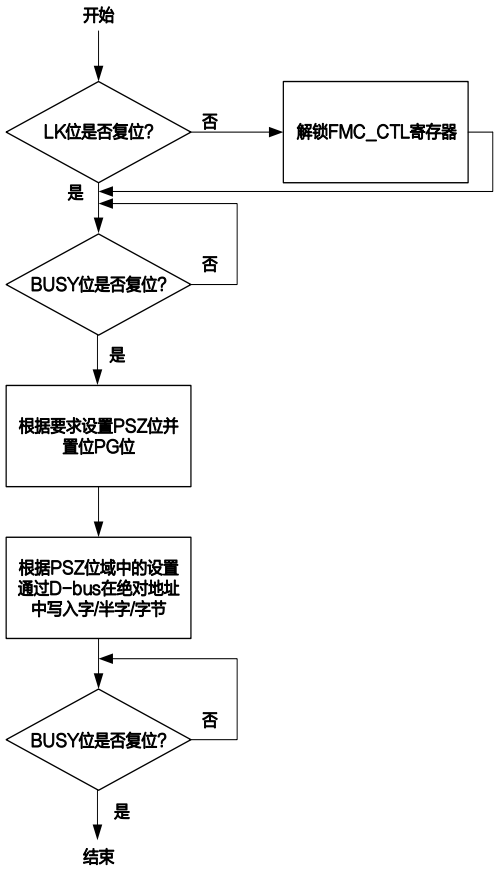

## 2.3.7. 主存储闪存块编程

FMC提供了一个32位整字/16位半字/8位字节编程功能，用来修改主存储闪存块内容。FMC闪存编程操作步骤如下：

1. 确保FMC_CTL寄存器不处于锁定状态；
2. 等待FMC_STAT寄存器的BUSY位变为0来确保没有闪存操作在进行，否则等待该操作完成；
3. 按照需求设置PSZ位域，并置位FMC_CTL寄存器的PG位；
4. DBUS写一个32位整字/16位半字/8位字节(必须与FMC_CTL寄存器中的PSZ位匹配)到目的绝对地址(0x08XX XXXX)；
5. 通过检查FMC_STAT寄存器的BUSY位是否清0，来确定写操作执行完毕；
6. 如果需要，使用DBUS读操作验证是否编程成功。

当主存储块编程成功执行，且FMC_CTL寄存器中的ENDIE位为1时，FMC_STAT寄存器的END位置位，同时FMC将产生一个中断。需要注意的是，执行整字/半字/字节编程操作时需要与FMC_CTL寄存器中的PSZ位匹配。如果不匹配，FMC_STAT寄存器中的PGMERR位被置位。需要注意的是，PG位必须在32位整字/16位半字/8位字节编程开始前进行置位， 否则FMC_STAT寄存器中的PGSERR位会被置位。此外，向被保护擦除/编程扇区进行的编程操作会被忽略，同时FMC_STAT寄存器中的WPERR位被置位。在这些情况下，若FMC_CTL寄存器的ERRIE位被置1时，FMC会触发一个闪存操作错误中断。在中断服务程序中，可以检查FMC_STAT寄存器的PGMERR位、PGSERR位和WPERR位来判断哪一种错误发生了。*图2-4. 闪存编程操作流程*显示了字编程操作流程。

**图 2-4. 闪存编程操作流程**




**注意**：避免在同一个bank中既进行读操作，又进行擦除或编程操作。当CPU进入省电模式时，对闪存的操作将失败。

### 2.3.8. OTP 闪存块编程

FMC提供了一个32位整字/16位半字/8位字节编程功能，用来修改OTP闪存块内容。编程操作顺序同主闪存块编程操作顺序相同。OTP闪存块不可进行擦除操作。

OTP闪存块可以被划分为16个32字节大小的数据块和1个16字节大小的OTP锁定块。OTP锁定块地址从0x1FFF_7A00到0x1FFF_7A0F。OTP数据块地址从0x1FFF_7800到0x1FFF_79FF。OTP锁定块中每一个锁定字节(0x00表示锁定，0xFF表示未锁定)可以锁定相对应的OTP数据块，以阻止在这些数据块上的编程操作。锁定块0用于锁定地址范围0x1FFF_7800到0x1FFF_781F的OTP数据块0。锁定块1用于锁定地址范围0x1FFF_7820到0x1FFF_783F的OTP数据块1，以此类推。

若OTP数据块未被对应的OTP锁定块锁定，这16个OTP数据块可以编程多次，但是无法擦除。OTP数据块中的每一位只能被编程为从1到0，而不能从0到1，OTP锁定块的每个字节仅在被写为0x00时才能锁定对应的数据块。

### 2.3.9. 选项字节修改

FMC提供了一个擦除功能用来修改闪存中的选项字节。选项字节编程操作步骤如下：

1. 确保FMC_OBCTLx寄存器不处于锁定状态；
2. 等待FMC_STAT寄存器的BUSY位变为0来确保没有闪存操作在进行，否则等待该操作完成；
3. 在FMC_OBCTL0寄存器和FMC_OBCTL1寄存器中进行选项字节值写入；
4. 通过将FMC_OBCTL0寄存器的OB_START位置1来发送选项字节编程命令到FMC；
5. 通过检查FMC_STAT寄存器的BUSY位是否清0，来确定编程指令执行完毕；
6. 如果需要，使用DBUS读操作验证是否编程成功。

当选项字节编程成功执行，且FMC_CTL寄存器中的ENDIE位为1时，FMC_STAT寄存器的END位置位，同时FMC将产生一个中断。

### 2.3.10. 选项字节说明

每次系统复位后，闪存的选项字节被重加载到FMC_OBCTL0和FMC_OBCTL1寄存器后，可选字节生效。可选字节详情见*表2-3. 选项字节*。

**表 2-3. 选项字节**

| 地址 | 名称 | 说明 |
|---|---|---|
| 0x1fff c000 | USER | [7]：nRST_STDBY<br>0：进入待机模式时产生复位<br>1：进入待机模式时不产生复位<br>[6]：nRST_DPSLP<br>0：进入深度睡眠模式时产生复位<br>1：进入深度睡眠模式时不产生复位<br>[5]：nWDG_HW<br>0：硬件看门狗 |

| 地址 | 名称 | 说明 |
| --- | --- | --- |
| | | 1：软件自由看门狗(出厂值)<br>[4]：BB<br>0：当配置从主存储块启动(默认值)时，从bank0启动(出厂值)。<br>1:当配置从主存储块启动时,若bank1无启动程序，从bank0启动。否则，从bank1启动。<br>[3:2]：BOR_TH (BOR复位阈值)<br>00：BOR复位阈值3<br>01：BOR复位阈值2<br>10：BOR复位阈值1<br>11：BOR关闭(出厂值)<br>[1:0]：保留 |
| 0x1fff c001 | SPC | 安全保护代码<br>0xAA：无保护(出厂值)<br>除0xAA和0xCC其他值：保护级别低<br>0xCC：保护级别高 |
| 0x1fff c008 | WP0[7:0] | [7:0]：WP0[7:0]<br>扇区擦除/编程保护<br>0：当DRP为1时，无影响。当DRP为0时，擦除/编程保护。<br>1：当DRP为0时，无影响。当DRP为1时，擦除/编程和DBUS读保护(出厂值)。 |
| 0x1fff c009 | WP0 | [7]：DRP<br>DBUS读保护位<br>0：WP0位用于每一个扇区擦除/编程保护(出厂值)<br>1：WP0位用于每一个扇区擦除/编程保护和DBUS读保护<br>[6]：DBS<br>当闪存大小为1M字节时，选择为双块或单块<br>0：当闪存大小为1M字节时，选择为单块<br>1：当闪存大小为1M字节时，选择为双块<br>**注意**:该功能位仅对GD32A490xx 1MB闪存系列有效，出厂值为0<br>[5:4]：保留<br>[3:0]：WP0[11:8]<br>扇区擦除/编程保护<br>0：当DRP为1时，无影响。当DRP为0时，擦除/编程保护。<br>1：当DRP为0时，无影响。当DRP为1时，擦除/编程和DBUS读保护(出厂值)。 |
| 0x1ffec008 | WP1[7:0] | [7:0]：WP1[7:0]<br>Bank1扇区擦除/编程保护<br>0：当DRP为1时，无影响。当DRP为0时，擦除/编 |

| 地址 | 名称 | 说明 |
| --- | --- | --- |
| | | 程保护。<br>1：当DRP为0时，无影响。当DRP为1时，擦除/编程和DBUS读保护(出厂值)。 |
| 0x1ffec009 | WP1 | [7:4]：保留<br>[3:0]：WP1[11:8]<br>Bank1扇区擦除/编程保护<br>0：当DRP为1时，无影响。当DRP为0时，擦除/编程保护。<br>1：当DRP为0时，无影响。当DRP为1时，擦除/编程和DBUS读保护(出厂值)。 |

### 2.3.11. 扇区擦除/编程保护

FMC的扇区擦除/编程保护功能可以阻止对闪存的意外操作。当FMC对被保护扇区进行扇区擦除或编程操作时，操作本身无效且FMC_STAT寄存器的WPERR位将被置1。注意，当MER0/MER1被置位或SN无效时，进行扇区擦除时WPERR仍会被置位。如果WPERR位被置1且ERRIE位也被置1来使能相应的中断，FMC将触发闪存操作出错中断，等待CPU处理。配置选项字节的WP0 [11:0]/WP1[11:0]某位为0可以单独使能某几扇区的保护功能。

**表 2-4. 扇区保护 WP0/WP1 位**

| WP0/WP1 位 | 扇区保护 |
| --- | --- |
| WP0[0] | 扇区0 |
| WP0[1] | 扇区1 |
| WP0[2] | 扇区2 |
| .<br>.<br>. | .<br>.<br>. |
| WP0[10] | 扇区10 |
| WP0[11] | 扇区11 |
| WP1[0] | 扇区12 |
| WP1[1] | 扇区13 |
| WP1[2] | 扇区14 |
| .<br>.<br>. | .<br>.<br>. |
| WP1[10] | 扇区22 |
| WP1[11] | 扇区23~扇区27 |

### 2.3.12. DBUS 读保护

FMC提供了一个DBUS保护功能，当DRP设置为1时禁止对相应扇区进行DBUS读操作。如果DBUS读命令被发送至FMC一个被保护扇区，FMC_STAT寄存器中的RDDERR位会被置位。如果RDDERR位被置1且ERRIE位也被置1来使能相应的中断，FMC将触发闪存操作出错中断，

等待CPU处理。配置选项字节的WP0 [11:0]/WP1[11:0]某位为1并同时设置DRP为1，可以单独使能某几扇区的保护功能。

如果DRP为1，想要修改DRP为0或者将WP0 [11:0]/WP1[11:0]某位值从1变为0时，若芯片处于无安全保护状态，则必须先将芯片设置为低安全保护状态，然后再随着解除低安全保护的过程将DRP或WP0 [11:0]/WP1[11:0]某位值清0。否则，选项字节的修改被忽略并且FMC_STAT寄存器中的WPERR位会被FMC置位。如果WPERR位被置1且ERRIE位也被置1来使能相应的中断，FMC将触发闪存操作出错中断，等待CPU处理。

### 2.3.13. 安全保护

FMC提供了一个安全保护功能来阻止非法读取闪存。此功能可以很好地保护软件和固件免受非法的用户操作。安全保护等级划分三等。

无保护状态：当将SPC字节设置为0xAA，闪存将处于非安全保护状态。主存储块和选项字节可以被所有操作模式访问。

保护等级低：当设置SPC字节为除0xAA或0xCC外的任何值，激活低安全保护等级。主存储闪存块仅能被用户代码访问。在调试模式或者从SRAM中启动或者从boot loader模式启动时，这些模式下对主存储块的操作都被禁止。无论是在调试模式或者从SRAM中启动，还是从boot loader模式启动，如果对主存储块执行一次读操作，将会产生一个总线错误。在调试模式或者从SRAM中启动时或者从boot loader模式启动，如果对主存储块执行一次编程/擦除操作，FMC_STAT寄存器中的WPERR位会被置位。在低安全保护等级下，对于选项字节的所有操作都被允许。如果通过设置SPC字节为0xAA进入无保护状态，主存储闪存块将执行一次整片擦除操作。

保护等级高：当设置SPC字节为0xCC，激活高安全保护等级。当编程选择该保护等级时，调试模式，从SRAM中启动，或者从boot loader启动都被禁止。主存储闪存块可由用户代码的所有操作进行访问。选项字节禁止再次编程。所以，如果高保护等级被激活，将不能再降回到低保护等级或无保护等级。

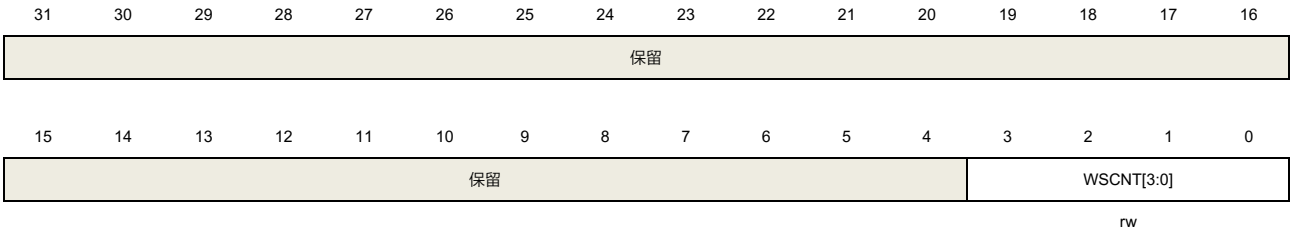
## 2.4. FMC 寄存器

FMC 基地址：0x4002 3C00

### 2.4.1. 等待状态寄存器 (FMC_WS)

地址偏移：0x00
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

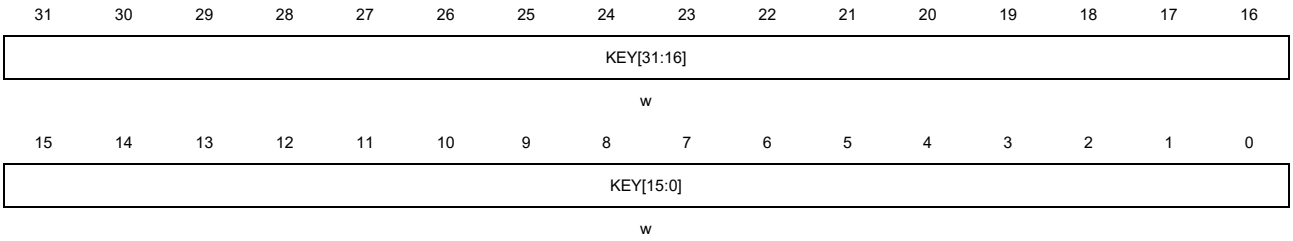



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:4 | 保留 | 必须保持复位值。 |
| 3:0 | WSCNT[3:0] | 等待状态计数寄存器<br>软件置1和清0。FMC_WSEN寄存器的WSEN位被置1时WSCNT位有效。<br>0000：不增加等待状态<br>0001：增加1个等待状态<br>0010：增加2个等待状态<br>…<br>1111：增加15个等待状态 |

### 2.4.2. 解锁寄存器 (FMC_KEY)

地址偏移：0x04
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | KEY[31:0] | FMC_CTL解锁寄存器 |

这些位仅能被软件写。
写解锁值到KEY[31:0]可以解锁 FMC_CTL寄存器。

### 2.4.3. 选项字节解锁寄存器 (FMC_OBKEY)

地址偏移：0x08
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | OBKEY[31:0] | FMC_OBCTLx选项字节解锁寄存器<br>这些位仅能被软件写<br>写解锁值到OBKEY[31:0]解锁FMC_OBCTLx寄存器的选项字节命令 |

### 2.4.4. 状态寄存器 (FMC_STAT)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | RDDERR | PGSERR | PGMERR | 保留 | WPERR | 保留 | | OPERR | END |
| | | | | | | | rc_w1 | rc_w1 | rc_w1 | | rc_w1 | | | rc_w1 | rc_w1 |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:17 | 保留 | 必须保持复位值。 |
| 16 | BUSY | 闪存忙标志位<br>当闪存操作正在进行时，此位被置1。当操作结束或者出错，此位被清0。 |
| 15:9 | 保留 | 必须保持复位值。 |
| 8 | RDDERR | DBUS读保护错误标志位<br>在DBUS读保护扇区进行DBUS读操作时，该位由硬件置位。软件写1清0。 |
| 7 | PGSERR | 编程顺序错误标志位 |

| | | |
|---|---|---|
| | | 当FMC_CTL寄存器中PG位未置位时进行闪存编程，该位由硬件置位。软件写1清0。 |
| 6 | PGMERR | 编程类型不匹配错误标志位<br>当编程写入数据类型(字/半字/字节访问)与FMC_CTL寄存器中PSZ位不匹配时，该位由硬件置位。软件写1清0。 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | WPERR | 擦除/编程保护错误标志位<br>在受保护的页上擦除/编程操作时，此位被硬件置1。软件写1清0。 |
| 3:2 | 保留 | 必须保持复位值。 |
| 1 | OPERR | 闪存操作错误标志位<br>该位由硬件置位，当FMC_CTL寄存器中ERRIE位被置位时闪存操作发生错误(当RDDERR/PGSERR/PGMERR/WPERR位被置位时表示错误发生)。软件写1清0。 |
| 0 | END | 操作结束标志位<br>当操作执行成功，此位被硬件置1。软件写1清0。 |

### 2.4.5. 控制寄存器 (FMC_CTL)

地址偏移：0x10
复位值：0x8000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LK | 保留 | | | | | ERRIE | ENDIE | 保留 | | | | | | | START |
| rs | | | | | | rw | rw | | | | | | | | rs |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| MER1 | 保留 | | | | | PSZ[1:0] | | SN[4:0] | | | | | MER0 | SER | PG |
| rw | | | | | | rw | | rw | | | | | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | LK | FMC_CTL锁定<br>当向FMC_KEY寄存器中写入正确的顺序，该位由硬件清除。软件置1。 |
| 30:26 | 保留 | 必须保持复位值。 |
| 25 | ERRIE | 错误中断使能位<br>软件置1和清0<br>0：无硬件中断产生<br>1：使能错误中断 |
| 24 | ENDIE | 操作结束中断使能位<br>软件置1和清0<br>0：无硬件中断产生<br>1：使能操作结束中断 |

| | | |
|---|---|---|
| 23:17 | 保留 | 必须保持复位值。 |
| 16 | START | 发送擦除命令位<br>该位由软件置位，发送擦除命令到FMC。该位当BUSY位清零时由硬件清零。 |
| 15 | MER1 | 主存储闪存bank1整片擦除命令位<br>软件置1和清0<br>0：无影响<br>1：主存储闪存bank1整片擦除命令 |
| 14:10 | 保留 | 必须保持复位值。 |
| 9:8 | PSZ[1:0] | 编程大小位<br>软件置1和清0<br>00：按字节编程访问<br>01：按半字编程访问<br>10/11：按字编程访问 |
| 7:3 | SN[4:0] | 选择擦除扇区号<br>软件置1和清0<br>00000：选择扇区0<br>00001：选择扇区1<br>…<br>01011：选择扇区11<br>01100：选择扇区24<br>01101：选择扇区25<br>01110：选择扇区26<br>01111：选择扇区27<br>10000：选择扇区12<br>10001：选择扇区13<br>…<br>11011：选择扇区23<br>11100~11111：保留 |
| 2 | MER0 | 主存储块bank0整片擦除命令位<br>软件置1和清0<br>0：无作用<br>1：主存储块bank0整片擦除命令 |
| 1 | SER | 主存储块扇区擦除命令位<br>软件置1和清0<br>0：无作用<br>1：主存储块扇区擦除命令 |
| 0 | PG | 主存储块编程命令位<br>软件置1和清0<br>0：无作用<br>1：主存储块编程命令 |

**注意：**当相应闪存操作完成后，该寄存器需处于复位状态。

### 2.4.6. 选项字节控制寄存器 0 (FMC_OBCTL0)

地址偏移：0x14

复位值：0xXXXX XXXX，初始值为0x2FFF AAED。复位后装载选项字节中的值。

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DRP | DBS | 保留 | | WP0[11:0] | | | | | | | | | | | |
| rw | rw | | | rw | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SPC[7:0] | | | | | | | | nRST_STDBY | nRST_DPSLP | nWDG_HW | BB | BOR_TH[1:0] | | OB_START | OB_LK |
| rw | | | | | | | | rw | rw | rw | rw | rw | | rs | rs |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | DRP | DBUS读保护位<br>0：WPx位用于每一个扇区的擦除/编程保护<br>1：WPx位用于每一个扇区的擦除/编程和DBUS读保护 |
| 30 | DBS | 当闪存大小为1M字节时，选择设置为双块还是单块<br>0：当内存大小为1M字节时，设置为单块<br>1：当内存大小为1M字节时，设置为双块<br>**注意**：该功能位仅对GD32A490xx 1MB闪存系列有效。 |
| 29:28 | 保留 | 必须保持复位值。 |
| 27:16 | WP0[11:0] | 当DRP为0时，每个扇区擦除/编程保护<br>当DRP为1时，每个扇区擦除/编程和DBUS读保护。WP[0]作用于扇区0，WP[1]作用于扇区1，以此类推。<br>0：当DRP为0时，擦除/编程保护。当DRP为1时，无影响。<br>1：当DRP为0时，无影响。当DRP为1时，擦除/编程和DBUS读保护。 |
| 15:8 | SPC[7::0] | 选项字节安全保护代码<br>0xAA：无安全保护<br>0xCC：安全保护高<br>除0xAA或0xCC之外任何值：安全保护低 |
| 7 | nRST_STDBY | 选项字节待机复位值<br>0：产生复位而不进入待机模式<br>1：当进入待机模式时不产生复位 |
| 6 | nRST_DPSLP | 选项字节深度睡眠复位值<br>0：产生复位而不进入深度睡眠模式<br>1：当进入深度睡眠模式时不产生复位 |
| 5 | nWDG_HW | 选项字节看门狗值 |

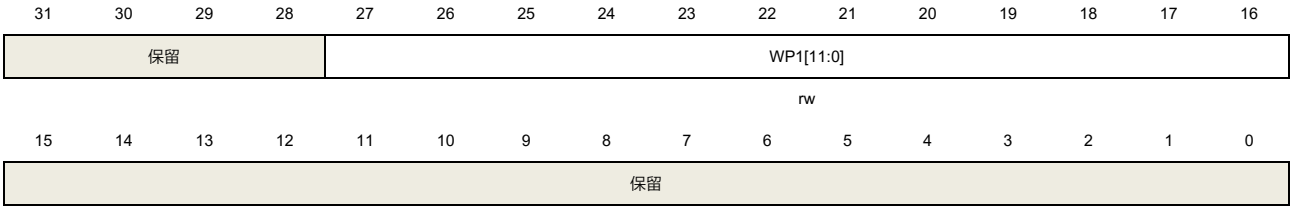
| | | |
|---|---|---|
| | | 如果改变该位，需要系统复位生效<br>0：硬件自由看门狗<br>1：软件自由看门狗 |
| 4 | BB | 选项字节启动块值<br>0：当配置从主存储块启动时，从bank0启动。<br>1：当配置从主存储块启动时，从bank1启动，若bank1无启动程序，从bank0启动。(如果bank0和bank1均无启动程序，芯片不处于高保护等级) |
| 3:2 | BOR_TH[1:0] | 选项字节BOR阈值<br>00：BOR阈值3<br>01：BOR阈值2<br>10：BOR阈值1<br>11：BOR关闭 |
| 1 | OB_START | 发送选项字节命令到FMC<br>该位由软件设置。当BUSY位清0时由硬件清除该位。 |
| 0 | OB_LK | FMC_OBCTLx锁定位<br>当往FMC_OBKEY寄存器写值顺序正确时，该位由硬件清0。软件置位。 |

## 2.4.7. 选项字节控制寄存器 1 (FMC_OBCTL1)

地址偏移：0x18

复位值：0xXXXX XXXX，初始值为0x0FFF 0000(复位后装载选项字节中的值)

该寄存器只能按字(32位)访问。

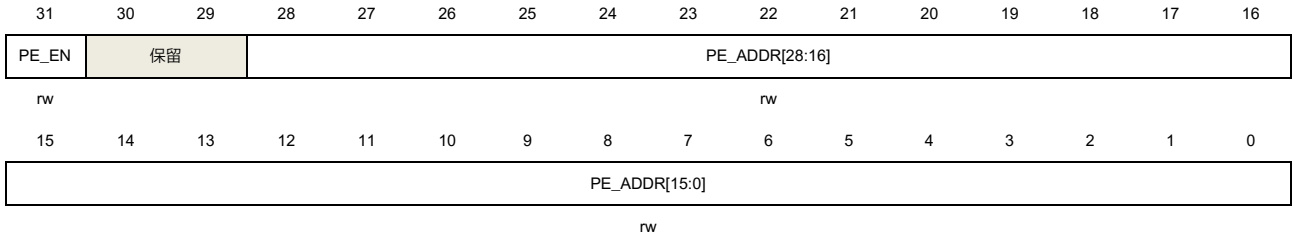



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:16 | WP1[11:0] | 当DRP为0时，每个扇区擦除/编程保护。<br>当DRP为1时，每个扇区擦除/编程和DBUS读保护。WP1[0]作用于扇区12，WP1[1]作用于扇区13，以此类推。特别指出，WP1[11]作用于扇区23~27。<br>0：当DRP为0时，擦除/编程保护。当DRP为1时，无影响。<br>1：当DRP为0时，无影响。当DRP为1时，擦除/编程和DBUS读保护。 |
| 15:0 | 保留 | 必须保持复位值。 |

### 2.4.8. 页擦除配置寄存器 (FMC_PECFG)

地址偏移：0x20
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

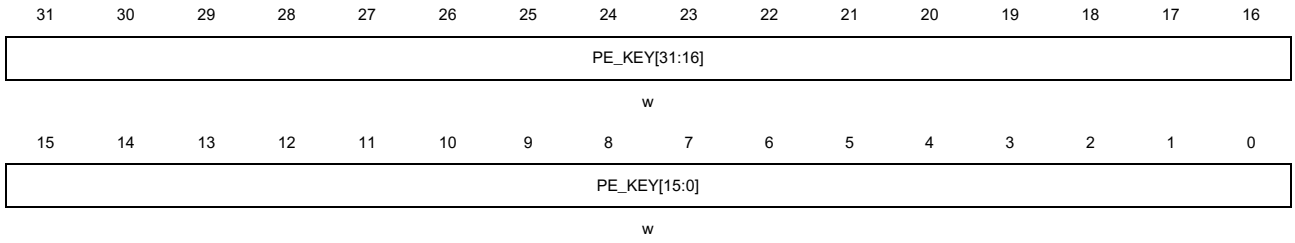



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | PE_EN | 页擦除功能使能位<br>0：禁能页擦除<br>1：使能页擦除 |
| 30:29 | 保留 | 必须保持复位值。 |
| 28:0 | PE_ADDR[28:0] | 待擦除页地址(4K字节对齐) |

### 2.4.9. 页擦除功能解锁寄存器 (FMC_PEKEY)

地址偏移：0x24
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | PE_KEY[31:0] | FMC_PECFG解锁寄存器<br>这些位仅能被软件写。<br>写解锁值0xA9B8C7D6到PE_KEY[31:0]可以解锁FMC_PECFG寄存器。 |

### 2.4.10. 等待状态使能寄存器 (FMC_WSEN)

地址偏移：0xFC
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:1 | 保留 | 必须保持复位值。 |
| 0 | WSEN | 闪存等待状态使能<br>软件写1和清0。该位由FMC_KEY寄存器保护。往FMC_KEY寄存器中写入0x45670123和0XCDEF89AB来解锁该位。<br>0：读取闪存时，无等待状态<br>1：读取闪存时，有等待状态 |

## 2.4.11. 产品 ID 寄存器 (FMC_PID)

地址偏移：0x100
复位值：0xXXXX XXXX

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | PID[31:0] | 产品保留ID寄存器<br>该寄存器为只读<br>上电后这些位始终不会改变，该寄存器在生产过程中被一次性编程。 |

# 3. 电源管理单元(PMU)

## 3.1. 简介

功耗设计是GD32A490xx系列产品比较注重的问题之一。电源管理单元提供了三种省电模式，包括睡眠模式，深度睡眠模式和待机模式。这些模式能减少电源能耗，且使得应用程序可以在CPU运行时间要求、速度和功耗的相互冲突中获得最佳折衷。如*图3-1. 电源域概览*所示，GD32A490xx系列设备有三个电源域，包括$V_{DD}$ / $V_{DDA}$域，$V_{CORE}$域和备份域。$V_{DD}$ / $V_{DDA}$域由电源直接供电。在$V_{DD}$ / $V_{DDA}$域中嵌入了一个LDO，用来为$V_{CORE}$域供电。在备份域中有一个电源切换器，当VDD电源关闭时，电源切换器可以将备份域的电源切换到VBAT引脚，此时备份域由VBAT引脚(电池)供电。

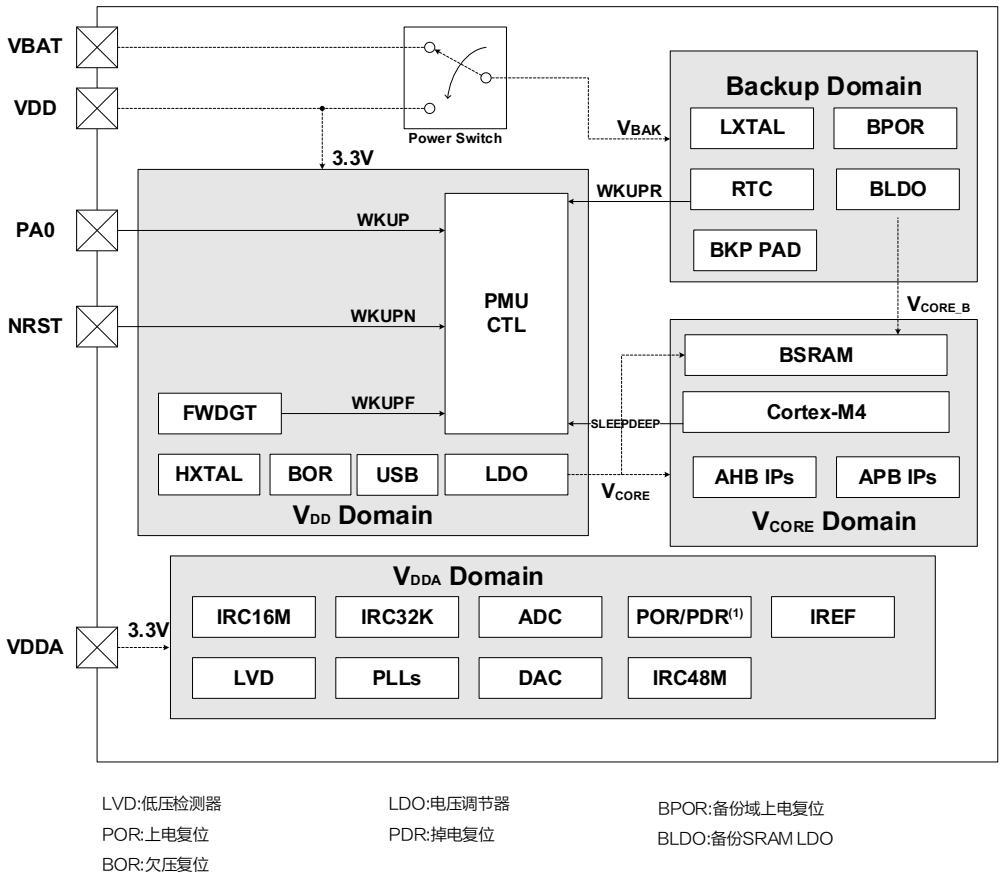
## 3.2. 主要特征

- 三个电源域：备份域、$V_{DD}$ / $V_{DDA}$域和$V_{CORE}$电源域。
- 三种省电模式：睡眠模式、深度睡眠模式和待机模式。
- 内部电压调节器(LDO)提供$V_{CORE}$电源和备份SRAM电压调节器(BLDO)专用于备份SRAM。
- 提供低电压检测器，当电压低于所设定的阈值时能发出中断或事件。
- 当VDD供电关闭时，由VBAT(电池)为备份域供电。
- 低驱动模式用于在深度睡眠模式下超低功耗。高驱动模式用在高频模式中。
- 由$V_{CORE}$(源于$V_{DD}$或$V_{BAK}$)电源供电的4K字节的SRAM，保证在VDD电源断开的情况下，用户数据也不会丢失。
- LDO输出电压用于节约能耗。

## 3.3. 功能说明

*图 3-1. 电源域概览*提供了 PMU 及相关电源域的内部结构框图。

图3-1. 电源域概览




**注意:**

(1)对于产品版本 C，POR / PDR 电路位于 $V_{DD}$ 域，检测 VDD 并在电压低于特定阈值时产生电源复位信号复位除备份域之外的整个芯片。

### 3.3.1. 电池备份域

电池备份域由内部电源切换器来选择VDD供电或VBAT(电池)供电，然后由$V_{BAK}$为备份域供电，该备份域包含RTC(实时时钟)、LXTAL(低速外部晶体振荡器)、BPOR(备份域上电复位)、BLDO(备份SRAM电压调节器)，以及PC13至PC15和PI8共4个BKP PAD。为了确保备份域中寄存器的内容及RTC正常工作，当VDD关闭时，VBAT引脚可以连接至电池或其他电源等备份源供电。电源切换器是由$V_{DD}$ / $V_{DDA}$域掉电复位电路控制的。对于没有外部电池的应用，建议将VBAT引脚通过100nF的外部陶瓷去耦电容连接到VDD引脚上。

备份域的复位源包括备份域上电复位和备份域软件复位。在$V_{BAK}$没有完全上电前，BPOR信号强制设备处于复位状态。应用软件可以通过设置RCU_BDCTL寄存器BKPRST位来触发备份域软件复位。

RTC的时钟源可以是低速内部RC振荡器(IRC32K)或低速外部晶体振荡器(LXTAL)，或高速外部晶体振荡器(HXTAL)时钟2~31分频。当$V_{DD}$被关闭时，RTC只能选择LXTAL作为时钟源。在通

过WFI / WFE指令进入省电模式之前，Cortex®-M4需要通过RTC寄存器设置预期的唤醒时间并启用唤醒功能，以实现RTC定时器唤醒事件。进入省电模式一定时间之后，当经过的时间与预设的唤醒时间匹配时，RTC将唤醒设备。RTC的配置和操作的细节将在*实时时钟(RTC)*来描述。

当备份域由VDD供电($V_{BAK}$连接至$V_{DD}$)时，以下功能可用：

- PC13和PI8可以作为通用I/O口或RTC功能引脚(参见*实时时钟(RTC)*)。
- PC14和PC15可以作为通用I/O口或LXTAL晶振引脚。

当备份域由VBAT电源供电时($V_{BAK}$连接至$V_{BAT}$)，以下功能可用：

- PC13和PI8仅可以作为RTC功能引脚(参见*实时时钟(RTC)*)。
- PC14和PC15仅可作为LXTAL晶振引脚。

**注意：**由于PC13至PC15和PI8引脚是通过电源切换器供电的，电源切换器仅可通过小电流，因此当PC13至PC15和PI8的GPIO口在输出模式时，其工作的速度不能超过2MHz(最大负载为30pF)。

### 3.3.2. 备份 SRAM

在 $V_{CORE}$ 电源域内有 4K 字节备份 SRAM。当 PMU_CS 寄存器中 BLDOON 被置位时，备份 SRAM 用于在待机模式或 $V_{DD}$ 断开时，存储应用数据。在这些情况下，备份 SRAM 由备份电压调节器供电。在其他模式(非待机模式或 $V_{DD}$ 断开)下，系统总线访问备份 SRAM 与访问普通 SRAM 的方式一样，由 $V_{DD}$ 电压调节器供电。

备份 SRAM 只能在 FMC 处于低级安全保护模式时被用户代码访问，以防止不合法的程序或数据访问。如果 FMC 退出低级安全保护模式进入无保护模式，备份 SRAM 被擦除，所有数据丢失。置位 RCU_BDCTL 寄存器的 BKPRST 位，不会复位备份 SRAM 中存储的数据。

### 3.3.3. $V_{DD}$ / $V_{DDA}$ 电源域

$V_{DD}$ / $V_{DDA}$ 域包括 $V_{DD}$ 域和 $V_{DDA}$ 域两部分。$V_{DD}$ 域包括 HXTAL(高速外部晶体振荡器)、LDO(电压调节器)、POR / PDR(上电/掉电复位)、FWDGT(独立看门狗定时器)和除 PC13、PC14、PC15 和 PI8 之外的所有 PAD 等等。$V_{DDA}$ 域包括 ADC / DAC(AD / DA 转换器)、IRC16M(内部 16M RC 振荡器)、IRC48(内部 48M RC 振荡器)、IRC32K(内部 32K RC 振荡器)PLLs(锁相环)和 LVD(低电压检测器)等等。

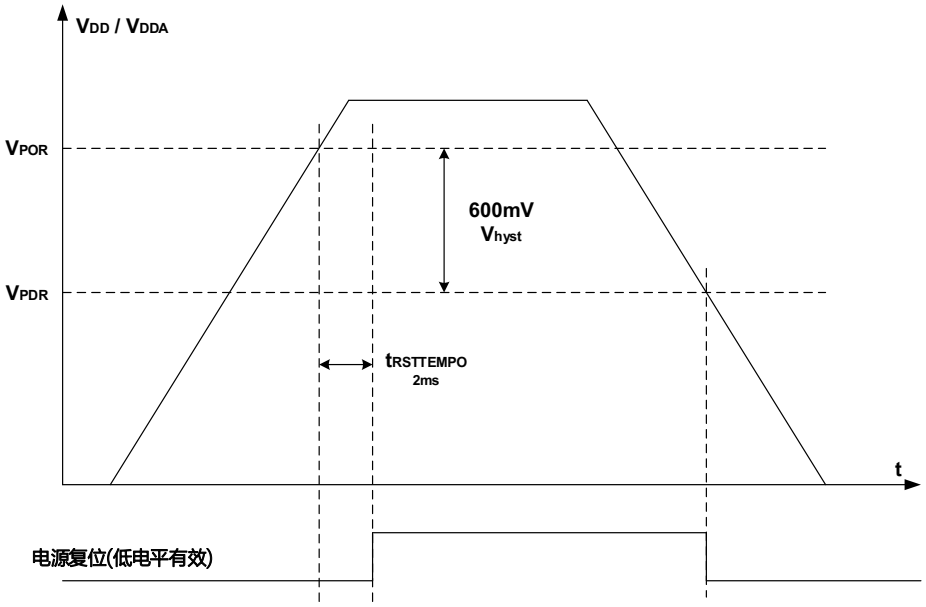
#### $V_{DD}$ 域

为 $V_{CORE}$ 域供电的 LDO(电压调节器)，其复位后保持使能。可以被配置为三种不同的工作状态：包括睡眠模式(全供电状态)、深度睡眠模式(全供电或低功耗状态)和待机模式(关闭状态)。

POR / PDR(上电 / 掉电复位)电路检测VDD / VDDA并在电压低于特定阈值时产生电源复位信号复位除备份域之外的整个芯片*图3-2. 上电 / 掉电复位波形图*显示了供电电压和电源复位信号之间的关系。$V_{POR}$表示上电复位的阈值电压，典型值约为2.40V，$V_{PDR}$表示掉电复位的阈值

电压，典型值约为1.8V。迟滞电压$V_{hyst}$值约为600mV。

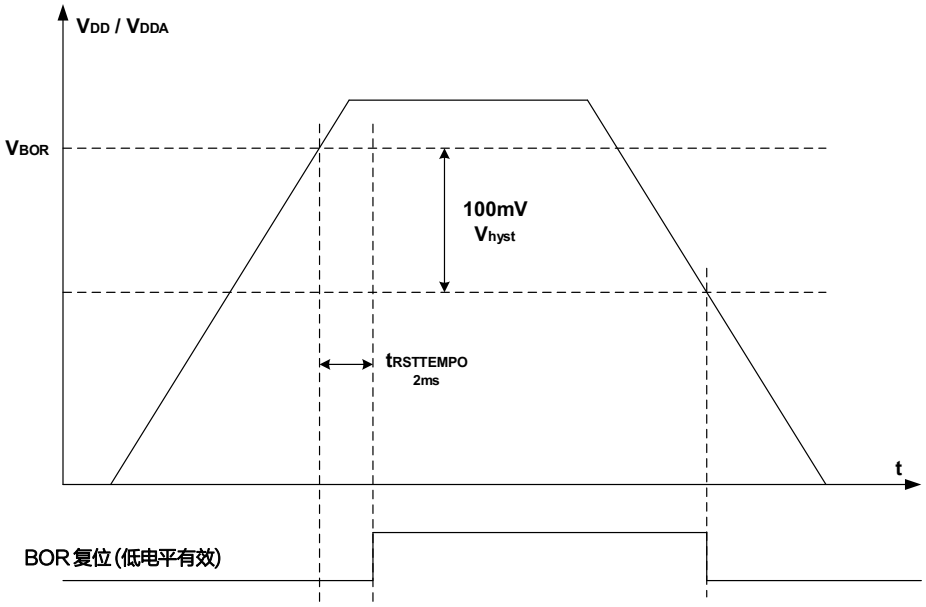
**图3-2. 上电 / 掉电复位波形图**




**注意：**PDR_ON引脚仅存在于不小于144-pin和BGA100-pin封装上。内部POR / PDR电路由PDR_ON引脚使能。为了保证芯片上电和下电阶段POR / PDR的有效性，建议将PDR_ON引脚通过电阻拉到VDD上(具体阻值参考数据手册)。。

BOR电路检测VDD / VDDA并在电压低于选项字节的BOR_TH定义的阈值且该阈值不为0b11(BOR_TH = 0b11，BOR功能关闭)时产生电源复位信号复位除备份域之外的整个芯片。不管选项字节BOR_TH的值是否为0b11，POR / PDR(上电 / 掉电复位)电路会一直处于检测状态。*图3-3. BOR波形图*显示了供电电压和BOR复位信号之间的关系。$V_{BOR}$表示BOR复位的阈值电压，该值在选项字节BOR_TH中定义。迟滞电压$V_{hyst}$值为100mV。

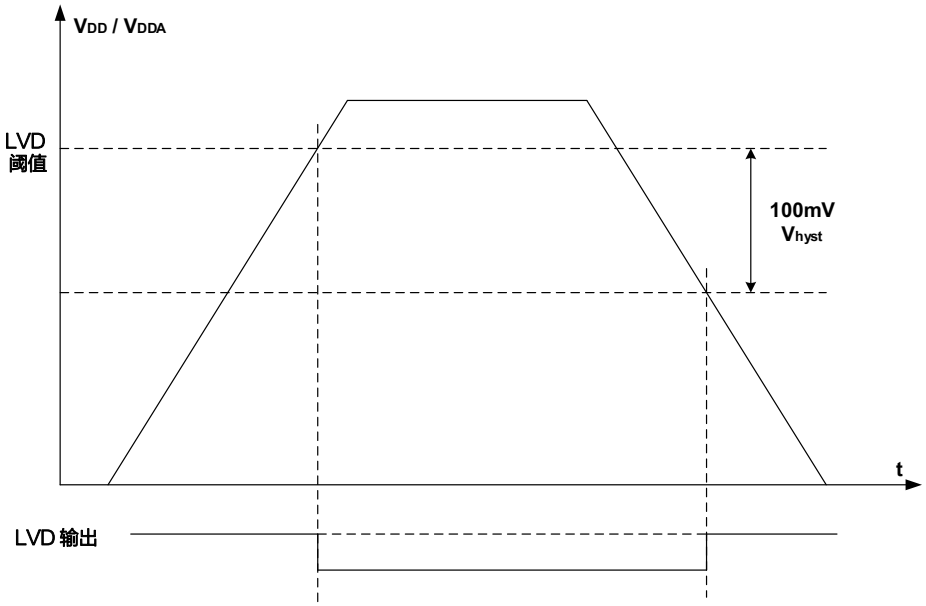
**图3-3. BOR波形图**



### $V_{DDA}$ 域

LVD的功能是检测VDD/VDDA供电电压是否低于低电压检测阈值，该阈值由电源控制寄存器(PMU_CTL)中的LVDT[2:0]位进行配置。LVD通过LVDEN置位使能，位于电源状态寄存器(PMU_CS)中的LVDF位表示低电压事件是否出现，该事件连接至EXTI的第16线，用户可以通过配置EXTI的第16线产生相应的中断。*图3-4. LVD阈值波形图*显示了$V_{DD}$ / $V_{DDA}$供电电压和LVD输出信号的关系。(LVD中断信号依赖于EXTI第16线的上升或下降沿配置)。迟滞电压$V_{hyst}$值为100mV。

**图 3-4. LVD 阈值波形图**




一般来说，数字电路由 VDD 供电，而大多数的模拟电路由 VDDA 供电。为了提高 ADC 和 DAC 的转换精度，为 VDDA 独立供电可使模拟电路达到更好的特性。为避免噪声，VDDA 通过外部滤波电路连接至 VDD，相应的 VSSA 通过特定电路连接至 VSS。否则，如果 VDDA 和 VDD 不同时，$V_{DDA}$ 须高于 $V_{DD}$，但压差不超过 0.3V。

为提高 ADC 和 DAC 的精度，可将独立的外部参考电压连接至 ADC/DAC 引脚 VREFP / VREFN。根据不同的封装，VREFP 可被连接至 VDDA 引脚，或者外部参考电压，外部参考电压的范围

请参考*表 14-2. ADC 输入引脚定义*和*表 15-1. DAC 引脚*. VREFN 须被连接至 VSSA 引脚，

VREFP 引脚仅存在于不小于 100-pin 封装上，而在 64-pin 或更少引脚封装不存在，因其内部已经连接至 VDDA 和 VSSA。VREFN 仅存在于 BGA176-pin 和 BGA100-pin 封装上，在其他封装里，其内部连接至 VSSA。

## 3.3.4. $V_{CORE}$ 电源域

$V_{CORE}$电源域为Cortex®-M4内核逻辑、AHB / APB外设、备份域和$V_{DD}$ / $V_{DDA}$域的APB接口等提供电源。当$V_{CORE}$电压上电后，POR将在$V_{CORE}$域中产生一个复位序列，复位完成后，如果要进入指定的省电模式，须先配置相关的控制位，之后一旦执行WFI或WFE指令，设备便进入该省

电模式。关于这方面的详细内容，将在以下章节予以说明。$V_{CORE}$电压值请参考数据手册。

#### 高驱动模式

如果$V_{CORE}$电源域工作在高频状态下，且打开了多种功能，建议进入高驱动模式。使用高驱动模式需要执行以下步骤：

- 选择系统时钟为IRC16M或HXTAL。
- 将PMU_CTL寄存器的HDEN置1，使能高驱动模式。
- 等待PMU_CS寄存器的HDRF被置位。
- 将PMU_CTL寄存器的HDS置1，将LDO切换到高驱动模式。
- 等待PMU_CS寄存器的HDSRF被置位。进入高驱动模式。
- 工作在高频状态。

在选择IRC16M或HXTAL作为系统时钟后，可以通过将PMU_CTL寄存器的HDEN和HDS清0退出高驱动模式。当系统退出深度睡眠模式，将会自动退出高驱动模式。

### 3.3.5. 省电模式

系统复位或电源复位后，GD32A490xx MCU处于全功能状态且电源域全部处于供电状态。实现较低的功耗的方法有三种：减慢系统时钟(HCLK，PCLK1，PCLK2)，关闭未使用的外设的时钟或通过PMU_CTL寄存器的LDOVS来配置LDO输出电压。LDOVS只有在PLL关闭情况下才可以配置，在PLL打开时，被配置的LDO输出电压才会被用来驱动$V_{CORE}$电源域。当PLL关闭时，LDO输出低电压模式用来驱动$V_{CORE}$域。此外，三种省电模式可以实现更低的功耗，它们是睡眠模式、深度睡眠模式和待机模式。

#### 睡眠模式

睡眠模式与Cortex®-M4的SLEEPING模式相对应。在睡眠模式下，仅关闭Cortex®-M4的时钟。如需进入睡眠模式，只要清除Cortex®-M4系统控制寄存器中的SLEEPDEEP位，并执行一条WFI或WFE指令即可。如果睡眠模式是通过执行WFI指令进入的，任何中断都可以唤醒系统。如果睡眠模式是通过执行WFE指令进入的，任何唤醒事件都可以唤醒系统(如果SEVONPEND为1，任何中断都可以唤醒系统，请参考Cortex®-M4技术手册)。由于无需在进入或退出中断上消耗时间，该模式所需的唤醒时间最短。

根据Cortex®-M4中SCR(系统控制寄存器)的SLEEPONEXIT位，有两种睡眠进入机制可选：

- Sleep-now：如果SLEEPONEXIT位被清零，一旦执行WFI或WFE指令，MCU立即进入睡眠模式。
- Sleep-on-exit：如果SLEEPONEXIT位被置位，当系统从最低优先级的中断处理程序离开后，MCU立即进入睡眠模式。

### 深度睡眠模式

深度睡眠模式与 Cortex®-M4 的 SLEEPDEEP 模式相对应。在深度睡眠模式下，$V_{CORE}$ 域中的所有时钟全部关闭，IRC16M、HXTAL 及 PLLs 也全部被禁用。SRAM 和寄存器中的内容被保留。根据 PMU_CTL 寄存器的 LDOLP 位的配置，可控制 LDO 工作在正常模式或低功耗模式。进入深度睡眠模式之前，先将 Cortex®-M4 系统控制寄存器的 SLEEPDEEP 位置 1，再清除 PMU_CTL 寄存器的 STBMOD 位，然后执行 WFI 或 WFE 指令即可进入深度睡眠模式。如果睡眠模式是通过执行 WFI 指令进入的，任何来自 EXTI 的中断可以将系统从深度睡眠模式中唤醒。如果睡眠模式是通过执行 WFE 指令进入的，任何来自 EXTI 的事件可以将系统从深度睡眠模式中唤醒(如果 SEVONPEND 为 1，任何来自 EXTI 的中断都可以唤醒系统，请参考 Cortex®-M4 技术手册)。刚退出深度睡眠模式时，IRC16M 被选中作为系统时钟。请注意，如果 LDO 工作在低功耗模式，那么唤醒时需额外的延时时间。

在深度睡眠模式下，通过配置 PMU_CTL 寄存器的 LDEN，LDNP，LDLP，LDOLP 位可以进入低驱动模式。低驱动模式具有低驱动能力，低能耗。

正常驱动 / 正常功耗：将 PMU_CTL 寄存器的 LDEN 位配置为 00，深度睡眠模式就工作在正常驱动模式下。将 PMU_CTL 寄存器的 LDOLP 清 0 可以使 LDO 处于正常功耗模式

正常驱动 / 低功耗：将 PMU_CTL 寄存器的 LDEN 位配置为 00，深度睡眠模式就工作在正常驱动模式下。将 PMU_CTL 寄存器的 LDOLP 置 1 可以使 LDO 处于低功耗模式。

低驱动 / 正常功耗：将 PMU_CTL 寄存器的 LDEN 设置为 11，LDNP 置 1 可以进入深度睡眠模式的低驱动模式。将 PMU_CTL 寄存器的 LDOLP 清 0 可以使 LDO 处于正常功耗模式。

低驱动 / 低功耗：将 PMU_CTL 寄存器的 LDEN 设置为 11，LDLP 置 1 可以进入深度睡眠模式的低驱动模式。将 PMU_CTL 寄存器的 LDOLP 置 1 可以使 LDO 处于低功耗模式。

非低驱动：将 PMU_CTL 寄存器的 LDEN 设置为 00，深度睡眠模式将不会处在低驱动模式。

**注意：**为了顺利进入深度睡眠模式，所有 EXTI 线上的挂起状态(在 EXTI_PD 寄存器中)和相关外设标志位必须被复位，参考*表 6-3. EXTI 触发源*。否则，程序将直接跳过深度睡眠模式进入过程而继续执行下面的程序。

### 待机模式

待机模式是基于 Cortex®-M4 的 SLEEPDEEP 模式实现的。在待机模式下，整个 $V_{CORE}$ 域全部停止供电，同时 LDO 和包括 IRC16M、HXTAL 和 PLL 也会被关闭。进入待机模式前，先将 Cortex®-M4 系统控制寄存器的 SLEEPDEEP 位置 1，再将 PMU_CTL 寄存器的 STBMOD 位置 1，再清除 PMU_CS 寄存器的 WUF 位，然后执行 WFI 或 WFE 指令，系统进入待机模式，PMU_CS 寄存器的 STBF 位状态表示 MCU 是否已进入待机模式。待机模式有四个唤醒源，包括来自 NRST 引脚的外部复位，RTC 唤醒事件，包括 RTC 侵入事件、RTC 闹钟事件、RTC 时间戳事件或 RTC 唤醒事件，FWDGT 复位，WKUP 引脚的上升沿。待机模式可以达到最低的功耗，但唤醒时间最长。另外，一旦进入待机模式，SRAM 和 $V_{CORE}$ 电源域寄存器(除了备份 SRAM，当 BLDOON 置位时)的内容都会丢失。退出待机模式时，会发生上电复位，复位之后 Cortex®-M4 将从 0x00000000 地址开始执行指令代码。

**表 3-1. 节电模式总结**

| 模式 | 睡眠 | 深度睡眠 | 待机 |
|---|---|---|---|
| **描述** | 仅关闭 CPU 时钟 | 1、关闭 $V_{CORE}$ 电源域的所有时钟<br>2、关闭 IRC16M、HXTAL 和 PLL | 1、关闭 $V_{CORE}$ 电源域的供电<br>2、关闭 IRC16M、HXTAL 和 PLL |
| **LDO 状态** | 开启(正常功耗，正常驱动模式) | 开启(正常功耗或低功耗模式，正常驱动或低驱动模式) | 关闭 |
| **配置** | SLEEPDEEP = 0 | SLEEPDEEP = 1<br>STBMOD = 0 | SLEEPDEEP = 1<br>STBMOD = 1，WURST=1 |
| **进入指令** | WFI 或 WFE | WFI 或 WFE | WFI 或 WFE |
| **唤醒** | 若通过 WFI 进入，则任何中断均可唤醒；<br>若通过 WFE 进入，则任何事件(或 SEVONPEND = 1 时的中断)均可唤醒 | 若通过 WFI 进入，来自 EXTI 的任何中断可唤醒；若通过 WFE 进入，来自 EXTI 的任何事件(或 SEVONPEND = 1 时的中断)可唤醒 | 1、NRST 引脚<br>2、WKUP 引脚<br>3、FWDGT 复位<br>4、RTC |
| **唤醒延迟** | 无 | IRC16M 唤醒时间<br>如果 LDO 处于低功耗模式，需增加 LDO 唤醒时间 | 上电序列 |

**注意：**在待机模式下，除了 NRST 引脚，配置为 RTC 功能的 PC13 和 PI8，用作 LXTAL 晶振引脚的 PC14 和 PC15，使能的 WKUP 引脚，其他所有 I / O 都处于高阻态。

## 3.4. PMU 寄存器

PMU 基地址：0x4000 7000

### 3.4.1. 控制寄存器(PMU_CTL)

地址偏移：0x00
复位值：0x0000 C000(从待机模式唤醒后复位)

该寄存器可以按半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LDOVS[1:0] | | 保留 | | LDNP | LDLP | 保留 | BKPWEN | LVDT[2:0] | | | LVDEN | STBRST | WURST | STBMOD | LDOLP |
| rs | | | | rw | rw | | rw | rw | | | rw | rc_w1 | rc_w1 | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:20 | 保留 | 必须保持复位值。 |
| 19:18 | LDEN[1:0] | 深度睡眠模式下，低驱动模式使能<br>00：在深度睡眠模式下，禁用低驱动模式<br>01：保留<br>10：保留<br>11：在深度睡眠模式下，使能低驱动模式 |
| 17 | HDS | 高驱动模式切换器<br>选择 IRC16M 或 HXTAL 作为系统时钟，当 HDRF 被置位时，由软件将该位置 1。该位被置位后，系统进入高驱动模式。可由软件清 0，也可在退出深度睡眠模式或 HDEN 被清 0 时由硬件清 0。<br>0：没有高驱动模式切换器<br>1：有高驱动模式切换器 |
| 16 | HDEN | 高驱动模式使能<br>当系统时钟为 IRC16M 或 HXTAL 时，该位由软件置位。当系统退出深度睡眠模式时，该位由软件或硬件清零。<br>0：禁用高驱动模式<br>1：使能高驱动模式 |
| 15:14 | LDOVS[1:0] | 选择 LDO 输出<br>在 PLL 关闭时，这些位由软件配置，在主 PLL 使能后，LDOVS 设置的值生效。如果主 PLL 关闭，LDO 输出低电压模式被选中。<br>00：保留(LDO 输出低电压模式)<br>01：LDO 输出低电压模式<br>10：LDO 输出中电压模式 |

| | | |
|---|---|---|
| | | 11：LDO 输出高电压模式 |
| 13:12 | 保留 | 必须保持复位值。 |
| 11 | LDNP | 使用正常功耗 LDO 时，工作在低驱动模式<br>0：使用正常功耗 LDO 时，工作在正常驱动模式<br>1：使用正常功耗 LDO 且 LDEN 为 11 时，低驱动模式被使能 |
| 10 | LDLP | 使用低功耗 LDO 时，工作在低驱动模式<br>0：使用低功耗 LDO 时，工作在正常驱动模式<br>1：使用低功耗 LDO 且 LDEN 为 11 时，低驱动模式被使能 |
| 9 | 保留 | 必须保持复位值。 |
| 8 | BKPWEN | 备份域写使能<br>0：禁止对备份域寄存器的写访问<br>1：允许对备份域寄存器的写访问<br>复位之后，任何对备份域寄存器的写访问都将被禁止。如需对备份域寄存器做写访问，需先将该位置 1。 |
| 7:5 | LVDT[2:0] | 低电压检测器阈值<br>000：2.1V<br>001：2.3V<br>010：2.4V<br>011：2.6V<br>100：2.7V<br>101：2.9V<br>110：3.0V<br>111：3.1V |
| 4 | LVDEN | 低电压检测器使能<br>0：关闭低电压检测器<br>1：开启低电压检测器 |
| 3 | STBRST | 待机标志复位<br>0：无影响<br>1：复位待机标志<br>读该位，始终返回 0 |
| 2 | WURST | 唤醒标志复位<br>0：无影响<br>1：复位唤醒标志<br>读该位，始终返回 0 |
| 1 | STBMOD | 待机模式<br>0：当 Cortex®-M4 进入 SLEEPDEEP 模式时，系统进入深度睡眠模式<br>1：当 Cortex®-M4 进入 SLEEPDEEP 模式时，系统进入待机模式 |
| 0 | LDOLP | LDO 低功耗模式<br>0：当系统进入深度睡眠模式时，LDO 仍正常工作 |

1：当系统进入深度睡眠模式时，LDO 进入低功耗模式

### 3.4.2. 电源控制和状态寄存器(PMU_CS)

地址偏移：0x04
复位值：0x0000 0000(从待机模式唤醒后不复位)

该寄存器可以按半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | LDRF[1:0] | | HDSRF | HDRF |
| | | | | | | | | | | | | rc_w1 | | r | r |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | LDOVSRF | 保留 | | | | BLDOON | WUPEN | 保留 | | | | BLDORF | LVDF | STBF | WUF |
| | r | | | | | rw | rw | | | | | r | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:20 | 保留 | 必须保持复位值。 |
| 19:18 | LDRF[1:0] | 低驱动模式就绪标志<br>在深度睡眠模式下，且 LDO 处于低驱动模式，这些位由硬件设置。软件对这些位写 11 可以清 0。<br>00：深度睡眠模式下，普通驱动模式<br>01：保留<br>10：保留<br>11：深度睡眠模式下，低驱动模式 |
| 17 | HDSRF | 高驱动切换器就绪标志<br>0：高驱动切换器未就绪<br>1：高驱动切换器就绪 |
| 16 | HDRF | 高驱动准备就绪标志<br>0：高驱动未就绪<br>1：高驱动就绪 |
| 15 | 保留 | 必须保持复位值。 |
| 14 | LDOVSRF | LDO 电压选择就绪标志<br>0：LDO 电压选择未就绪<br>1：LDO 电压选择就绪 |
| 13:10 | 保留 | 必须保持复位值。 |
| 9 | BLDOON | 开启备份 SRAM 电压选择器<br>当该位由软件置位，开启备份 SRAM 电压调节器，用于在断开 $V_{DD}$ 时保护备份 SRAM 中的数据。当断开 $V_{DD}$ 断开，同时该位被清 0，备份 SRAM 中的数据将会丢失。<br>0：关闭备份 SRAM 电压选择器 |

| | | |
|---|---|---|
| | | 1：开启备份 SRAM 电压选择器 |
| 8 | WUPEN | WKUP 引脚唤醒使能<br>0：关闭 WKUP 引脚唤醒功能<br>1：开启 WKUP 引脚唤醒功能<br>如果 WUPEN 在进入省电模式之前置 1，WKUP 引脚的上升沿会将系统从省电模式唤醒。由于 WKUP 引脚为高电平有效，WKUP 引脚内部被配置为输入下拉模式。当置位该控制位后，当输入为高的时候，将会触发一个唤醒事件。 |
| 7:4 | 保留 | 必须保持复位值。 |
| 3 | BLDORF | 备份域电压选择器就绪标志<br>0：备份域电压选择器未就绪<br>1：备份域电压选择器就绪 |
| 2 | LVDF | 低电压状态标志<br>0：低电压事件没出现($V_{DD}$ 高于设定的 LVD 阈值)<br>1：低电压事件出现($V_{DD}$ 等于或低于 LVD 阈值)<br>**注意**：LVD 功能在待机模式被禁用。 |
| 1 | STBF | 待机标志<br>0：设备没进入过待机模式<br>1：设备曾进入过待机模式<br>该位只能由 POR/PDR 或通过设置 PMU_CTL 寄存器的 STBRST 位来清零。 |
| 0 | WUF | 唤醒标志<br>0：没有收到唤醒事件<br>1：收到来自 WKUP 引脚或 RTC 唤醒事件，包括 RTC 侵入事件、RTC 闹钟事件、RTC 时间戳事件或 RTC 唤醒。<br>该位只能由 POR/PDR 或通过设置 PMU_CTL 寄存器的 WURST 位来清零。 |

# 4. 复位和时钟单元(RCU)

## 4.1. 复位控制单元(RCTL)

### 4.1.1. 简介

GD32A490xx复位控制包括三种控制方式：电源复位、系统复位和备份域复位。电源复位又称为冷复位，其复位除了备份域的所有系统。系统复位将复位除了SW-DP控制器和备份域之外的其余部分，包括处理器内核和外设IP。备份域复位将复位备份区域。复位能够被外部信号、内部事件和复位发生器触发。后续章节将介绍关于这些复位的详细信息。

### 4.1.2. 功能说明

#### 电源复位

当以下事件中之一发生时，产生电源复位：1、上电 / 掉电复位(POR / PDR 复位)2、欠压复位(BOR 复位)3、从待机模式中返回后由内部复位发生器产生。电源复位复位所有的寄存器除了备份域。电源复位为低电平有效，当内部LDO电源基准准备好提供1.2V电压时，电源复位电平将变为无效。复位入口向量被固定在存储器映射的地址0x0000_0004。

#### 系统复位

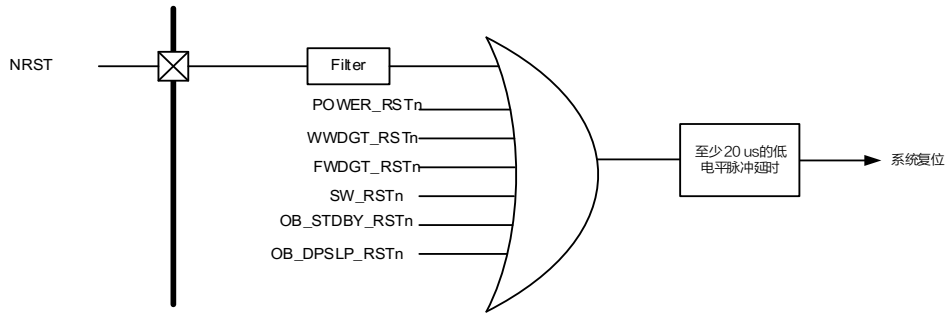
当发生以下任一事件时，产生一个系统复位：

- 上电复位(POWER_RSTn)；
- 外部引脚复位(NRST)；
- 窗口看门狗计数终止(WWDGT_RSTn)；
- 独立看门狗计数终止(FWDGT_RSTn)；
- Cortex®-M4的中断应用和复位控制寄存器中的SYSRESETREQ位置'1'( SW_RSTn)；
- 用户选择字节寄存器nRST_STDBY位设置为0，并且进入待机模式时将产生复位(OB_STDBY_RSTn)；
- 用户选择字节寄存器nRST_DPSLP设置为0，并且进入深度睡眠模式时(OB_DPSLP_RSTn)。

系统复位将复位除了SW-DP控制器和备份域之外的其余部分，包括处理器内核和外设IP。

系统复位脉冲发生器保证每一个复位源(外部或内部)都能有至少20μs的低电平脉冲延时。

**图4-1. 系统复位电路**




### 备份域复位

当以下事件之一发生时，产生备份域复位：
1、设置备份域控制寄存器中的BKPRST位为'1'；
2、备份域电源上电复位(在$V_{DD}$和$V_{BAT}$两者都掉电的前提下，$V_{DD}$或$V_{BAT}$上电)。

**注意：**当备份域复位时，BKPSRAM域不会复位

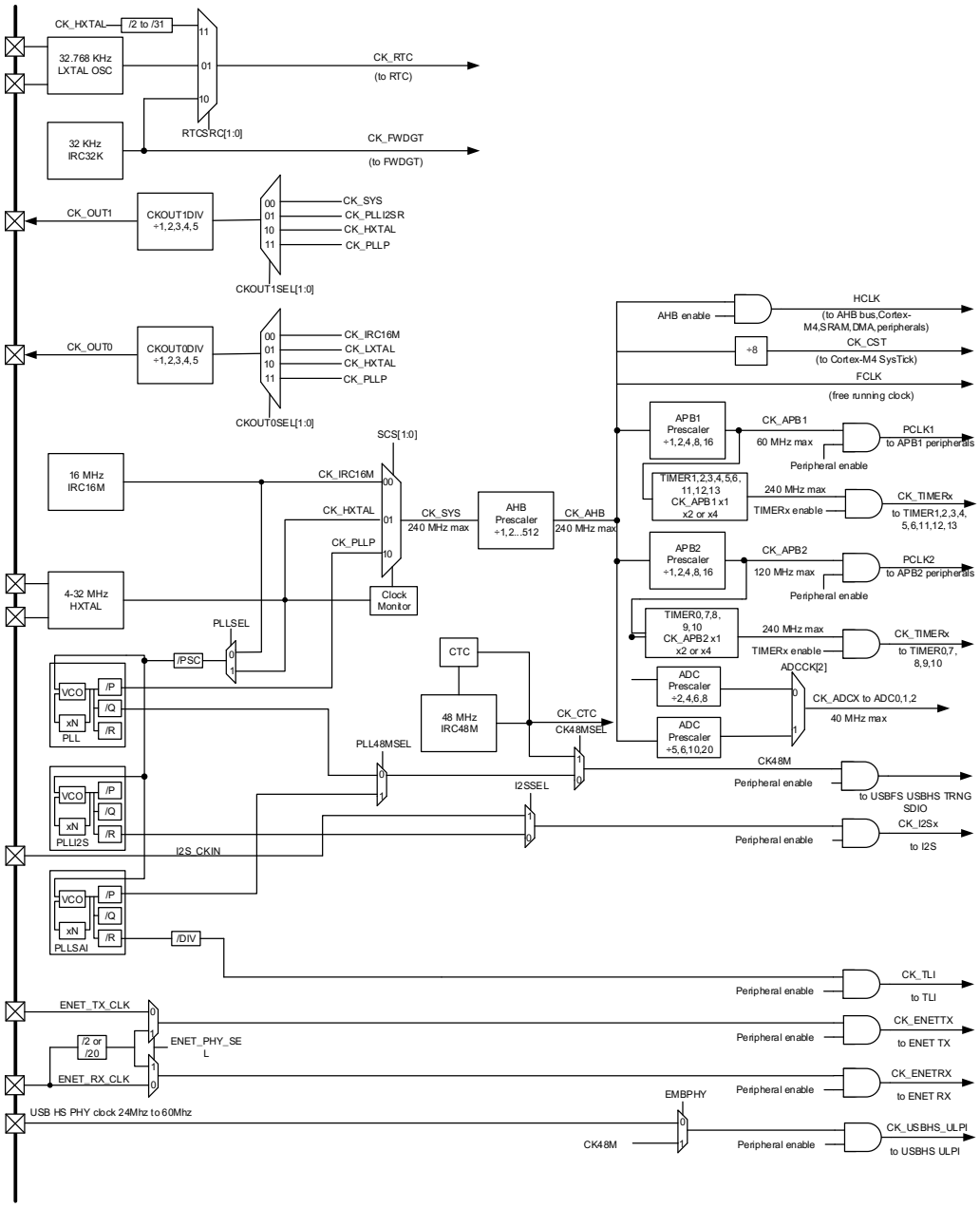
## 4.2. 时钟控制单元(CCTL)

### 4.2.1. 简介

时钟控制单元提供了一系列频率的时钟功能，包括一个内部16M RC振荡器时钟(IRC16M)、一个内部48M RC振荡器时钟(IRC48M)、一个外部高速晶体振荡器时钟(HXTAL)、一个内部32K RC振荡器时钟(IRC32K)、一个外部低速晶体振荡器时钟(LXTAL)、三个锁相环(PLL)、一个HXTAL时钟监视器、时钟预分频器、时钟多路复用器和时钟门控电路。

AHB、APB和Cortex®-M4时钟都源自系统时钟(CK_SYS)，系统时钟的时钟源可以选择IRC16M、HXTAL或PLL。系统时钟的最大运行时钟频率可以达到240MHz。

**图4-2. 时钟树**




预分频器可以配置 AHB、APB2 和 APB1 域的时钟频率。AHB 和 APB2/APB1 域的最高时率分别为 240 MHz/120 MHz/60 MHz。RCU 通过 AHB 时钟(HCLK)8 分频后作为 Cortex 系统定时器(SysTick)的外部时钟。通过对 SysTick 控制和状态寄存器的设置，可选择上述时钟或 AHB(HCLK)时钟作为 SysTick 时钟。

ADC时钟由APB2时钟经2、4、6、8分频或由AHB时钟经5、6、10、20分频获得，它们是通过设置ADC_SYNCCTL寄存器的ADCCK位来选择。

TIMER时钟由AHB时钟分频获得，它的频率可以等于CK_APBx、CK_APBx的两倍或CK_APBx的四倍。详细信息请参考RCU_CFG1寄存器的TIMERSEL位。

USBFS/USBHS/TRNG/SDIO的时钟由CK48M时钟提供。通过配置RCU_ADDCTL寄存器的

CK48MSEL及PLL48MSEL位可以选择PLLQ时钟、PLLSAIP时钟或IRC48M时钟做为CK48M的时钟源。

通过配置USBHS_GUSBCS寄存器的EMBPHY位，USBHS ULPI的时钟可以选择由外部ULPI PHY时钟或CK48M时钟提供。

CTC时钟由IRC48M时钟提供，通过CTC单元，可以实现IRC48M时钟精度的自动调整。

通过设置时钟配置寄存器RCU_CFG0的I2SSEL位，I2S时钟可以选择由PLLI2SR时钟或外部I2S_CKIN引脚输入时钟提供。

通过设置时钟配置寄存器RCU_CFG1的PLLSAIRDIV位域，TLI时钟可以选择由PLLSAIR时钟的2、4、8、16分频提供。

通过配置SYSCFG_CFG1寄存器的ENET_PHY_SEL位，以太网TX/RX时钟可以选择由外部引脚(ENET_TX_CLK / ENET_RX_CLK)输入时钟提供。

通过配置RCU_BDCTL寄存器的RTCSRC位，RTC时钟可以选择由LXTAL时钟、IRC32K时钟或HXTAL时钟的2-31(由RCU_CFG0寄存器的RTCDIV位域值决定)分频提供。RTC时钟选择HXTAL时钟的分频做为时钟源后，当1.2V内核电压域掉电时，时钟将停止。RTC时钟选择IRC32K时钟做为时钟源后，当$V_{DD}$掉电时，时钟将停止。RTC时钟选择LXTAL时钟做为时钟源后，当$V_{DD}$和$V_{BAT}$都掉电时，时钟将停止。

当FWDG启动时，FWDG时钟被强制选择由IRC32K时钟做为时钟源。

### 4.2.2. 主要特性

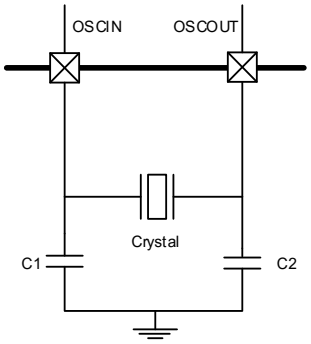
- 4到32 MHz外部高速晶体振荡器(HXTAL)；
- 内部16 MHz RC振荡器(IRC16M)；
- 内部48 MHz RC振荡器(IRC48M)；
- 32,768 Hz外部低速晶体振荡器(LXTAL)；
- 内部32 KHz RC振荡器(IRC32K)；
- PLL时钟源可选HXTAL或IRC16M；
- HXTAL时钟监视器。

### 4.2.3. 功能说明

#### 外部高速晶体振荡器时钟(HXTAL)

4到32MHz的外部高速晶体振荡器可为系统时钟提供更为精确时钟源。带有特定频率的晶体必须靠近两个HXTAL的引脚连接。和晶体连接的外部电阻和电容必须根据所选择的振荡器来调整。

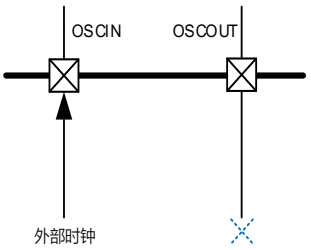
**图4-3. HXTAL时钟源**




HXTAL晶体振荡器可以通过设置控制寄存器RCU_CTL的HXTALEN位来启动或关闭，在控制寄存器RCU_CTL中的HXTALSTB位用来指示外部高速振荡器是否已稳定。在启动时，直到这一位被硬件置'1'，时钟才被释放出来。这个特定的延迟时间被称为振荡器的启动时间。当HXTAL时钟稳定后，如果在中断寄存器RCU_INT中的相应中断使能位HXTALSTBIE位被置'1'，将会产生相应中断。此时，HXTAL时钟可以被直接用作系统时钟源或者PLL输入时钟。

将控制寄存器RCU_CTL的HXTALBPS和HXTALEN位置'1'可以设置外部时钟旁路模式。路输入时，信号接至OSCIN，OSCOUT保持悬空状态，如*图4-4. 旁路模式下HXTAL时钟源*所示。此时，CK_HXTAL等于驱动OSCIN管脚的外部时钟。

**图4-4. 旁路模式下HXTAL时钟源**




## 内部 16M RC 振荡器时钟(IRC16M)

内部16MHz RC振荡器时钟，简称IRC16M时钟，拥有16MHz的固定频率，设备上电后CPU默认选择其做为系统时钟源。IRC16M RC振荡器能够在不需要任何外部器件的条件下为用户提供更低成本类型的时钟源。IRC16M RC振荡器可以通过设置控制寄存器(RCU_CTL)中的IRC16MEN位被启动和关闭。控制寄存器RCU_CTL中的IRC16MSTB位用来指示IRC16M内部RC振荡器是否稳定。IRC16M振荡器的启动时间比HXTAL晶体振荡器要更短。如果中断寄存器RCU_INT中的相应中断使能位IRC16MSTBIE被置'1'，在IRC16M稳定以后，将产生一个中断。IRC16M时钟也可用作系统时钟源或PLL输入时钟。

工厂会校准IRC16M时钟频率的精度，但是它的精度仍然比HXTAL时钟要差。用户可以根据需求、环境条件和成本决定选择哪个时钟作为系统时钟源。

如果HXTAL或者PLL被选择为系统时钟源，为了最大程度减小系统从深度睡眠模式恢复的时间，当系统从深度睡眠模式初始唤醒时，硬件会强制IRC16M时钟作为系统时钟。

### 内部 48M RC 振荡器时钟(IRC48M)

内部48MHz RC振荡器时钟，简称IRC48M时钟，拥有48MHz的固定频率，当使用USBFS/USBHS/TRNG/SDIO模块时，IRC48M振荡器在不需要任何外部器件的条件下为用户提供了一种成本更低的时钟源选择。IRC48M RC振荡器可以通过设置RCU_ADDCTL寄存器中的IRC48MEN位被启动和关闭。RCU_ADDCTL寄存器中的IRC48MSTB位用来指示内部48MHz RC振荡器是否稳定。如果RCU_ADDINT寄存器中的相应中断使能位IRC48MSTBIE被置'1',在IRC48M稳定以后,将产生一个中断。IRC48M时钟可做为USBFS/USBHS/TRNG/SDIO模块时钟。

工厂会校准IRC48M时钟频率的精度，但是它的精度仍然不够精准。因为USB模块需要的时钟频率必须满足48MHz+/–1%。CTC单元提供了一种硬件自动执行动态调整的功能将IRC48M时钟调整到需要的频率。

### 锁相环(PLL)

存在三个内部锁相环，PLL、PLLI2S和PLLSAI。PLLP时钟可做为系统时钟(不超过240MHz),PLLQ时钟可以做为USBFS/USBHS/TRNG/SDIO模块的时钟源。PLLI2S时钟可以做为I2S模块的时钟源。PLLSAI可以做为CK48M或TLI模块的时钟源。

PLL可以通过设置RCU_CTL寄存器中的PLLEN位被启动和关闭。RCU_CTL寄存器中的PLLSTB位用来指示PLL时钟是否稳定。如果RCU_INT寄存器中的相应中断使能位PLLSTBIE被置'1'，在PLL稳定以后，将产生一个中断。

PLLI2S可以通过设置RCU_CTL寄存器中的PLLI2SEN位被启动和关闭。RCU_CTL寄存器中的PLLI2SSTB位用来指示PLLI2S时钟是否稳定。如果RCU_INT寄存器中的相应中断使能位PLLI2SSTBIE被置'1'，在PLLI2S稳定以后，将产生一个中断。

PLLSAI可以通过设置RCU_CTL寄存器中的PLLSAIEN位被启动和关闭。RCU_CTL寄存器中的PLLSAISTB位用来指示PLLSAI时钟是否稳定。如果RCU_INT寄存器中的相应中断使能位PLLSAISTBIE被置'1'，在PLLSAI稳定以后，将产生一个中断。

当进入Deepsleep/Standby模式或者HXTAL监视器检测到时钟阻塞时(HXTAL做为锁相环的输入时钟)，这三路PLL将被关闭。

### 外部低速晶体振荡器时钟(LXTAL)

LXTAL是一个频率为32.768kHz的外部低速晶体或陶瓷谐振器。它为实时时钟电路提供一个低功耗且高精准的时钟源。LXTAL振荡器可以通过设置备份域控制寄存器(RCU_BDCTL)中的LXTALEN位被启动和关闭。备份域控制寄存器RCU_BDCTL中的LXTALSTB位用来指示LXTAL时钟是否稳定。如果中断寄存器RCU_INT中的相应中断使能位LXTALSTBIE被置'1'，在LXTAL稳定以后，将产生一个中断。

将备份域控制寄存器RCU_BDCTL的LXTALBPS和LXTALEN位置'1'可以选择外部时钟旁路模式。CK_LXTAL与连到OSC32IN脚上外部时钟信号一致。

### 内部 32K RC 振荡器时钟(IRC32K)

IRC32K内部RC振荡器时钟担当一个低功耗时钟源的角色，它的时钟频率大约32 kHz，为独立看门狗和实时时钟电路提供时钟。IRC32K提供低成本的时钟源，因为不需要外部器件。IRC32K RC振荡器可以通过设置复位源/时钟寄存器RCU_RSTSCK中的IRC32KEN位被启动和关闭。复位源/时钟寄存器RCU_RSTSCK中的IRC32KSTB位用来指示IRC32K时钟是否已稳定。如果复位源/时钟寄存器RCU_RSTSCK中的相应中断使能位IRC32KSTBIE被置'1'，在IRC32K稳定以后，将产生一个中断。

### 系统时钟(CK_SYS)选择

系统复位后，IRC16M时钟默认做为CK_SYS的时钟源，改变配置寄存器0，RCU_CFG0中的系统时钟变换位SCS可以切换系统时钟源为HXTAL或CK_PLLP。当SCS的值被改变，系统时钟将使用原来的时钟源继续运行直到转换的目标时钟源稳定。当一个时钟源被直接或通过PLL间接作为系统时钟时，它将不能被停止。

### HXTAL 时钟监视器(CKM)

设置控制寄存器RCU_CTL中的HXTAL时钟监视使能位CKMEN，HXTAL可以使能时钟监视功能。该功能必须在HXTAL启动延迟完毕后使能，在HXTAL停止后禁止。一旦监测到HXTAL故障，HXTAL将自动被禁止，中断寄存器RCU_INT中的HXTAL时钟阻塞中断标志位CKMIF将被置'1'，产生HXTAL故障事件。这个故障引发的中断和Cortex®-M4的不可屏蔽中断NMI相连。如果HXTAL被选作系统或PLL的时钟源，HXTAL故障将促使选择IRC16M为系统时钟源且PLL将被自动禁止。

### 时钟输出功能

时钟输出功能输出从32kHz到240MHz的时钟。通过设置时钟配置寄存器0(RCU_CFG0)中的CK_OUT0时钟源选择位域CKOUT0SEL能够选择不同的时钟信号。相应的GPIO引脚应该被配置成备用功能I/O(AFIO)模式来输出选择的时钟信号。CK_OUT1时钟输出源选择通过设置时钟配置寄存器RCU_CFG0中的CKOUT1SEL位域实现。

**表4-1. 时钟输出0的时钟源选择**

| 时钟输出 0 的时钟源选择位域 | 时钟源 |
|---|---|
| 00 | CK_IRC16M |
| 01 | CK_LXTAL |
| 10 | CK_HXTAL |
| 11 | CK_PLLP |

**表4-2. 时钟输出1的时钟源选择**

| 时钟输出 1 的时钟源选择位域 | 时钟源 |
|---|---|
| 00 | CK_SYS |
| 01 | CK_PLLI2SR |
| 10 | CK_HXTAL |

| 时钟输出 1 的时钟源选择位域 | 时钟源 |
|---|---|
| 11 | CK_PLLP |

通过配置RCU_CFG0寄存器的CKOUT0DIV位域，可以将CK_OUT0输出时钟的频率按比例分频，进而降低CK_OUT0的输出频率。

通过配置RCU_CFG0寄存器的CKOUT1DIV位域，可以将CK_OUT1输出时钟的频率按比例分频，进而降低CK_OUT1的输出频率。

### RTC 时钟测量

RTC时钟的三种时钟源：LXTAL、IRC32K和HXTAL时钟的2-31分频(通过配置RCU_CFG0寄存器的RTCDIV位域)，可以通过TIMER模块测量频率。用户可以根据计算得到的时钟频率调整RTC和独立看门狗计数器。详细信息请参考 ***TIMER4_IRMP***寄存器的CI3_RMP位以及 ***TIMER10_IRMP***寄存器的ITI1_RMP位。

### 电压控制

深度睡眠模式电压寄存器(RCU_DSV)中的DSLPVS[2:0]位域可以控制1.2V域在深度睡眠模式下的电压。

**表4-3. 深度睡眠模式下1.2V域电压选择**

| DSLPVS[2:0] | 深度睡眠模式电压(V) |
|---|---|
| 000 | 缺省值 |
| 001 | 缺省值-0.1 |
| 010 | 缺省值-0.2 |
| 011 | 缺省值-0.3 |
| 100 ~ 111 | 保留 |

RCU_DSV寄存器被电源解锁寄存器(RCU_VKEY)保护。只有在写0x1A2B3C4D到RCU_VKEY后，RCU_DSV寄存器才能被写入。

## 4.3. RCU 寄存器

RCU基地址：0x4002 3800

### 4.3.1. 控制寄存器(RCU_CTL)

地址偏移：0x00
复位值：0x0000 xx83 x表示未定义

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | PLLSAISTB | PLLSAIEN | PLLI2SSTB | PLLI2SEN | PLLSTB | PLLEN | 保留 | | | | CKMEN | HXTALBPS | HXTALSTB | HXTALEN |
| | | r | rw | r | rw | r | rw | | | | | rw | rw | r | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IRC16MCALIB[7:0] | | | | | | | | IRC16MADJ[4:0] | | | | | 保留 | IRC16MSTB | IRC16MEN |
| r | | | | | | | | rw | | | | | | r | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:30 | 保留 | 必须保持复位值。 |
| 29 | PLLSAISTB | PLLSAI 时钟稳定标志位<br>硬件置1来表示PLLSAI输出时钟是否稳定待用<br>0：PLLSAI未稳定<br>1：PLLSAI 已稳定 |
| 28 | PLLSAIEN | PLLSAI 使能<br>软件置位或复位，当进入深度睡眠或待机模式时由硬件复位。<br>0：PLLSAI被关闭<br>1：PLLSAI 被打开 |
| 27 | PLLI2SSTB | PLLI2S 时钟稳定标志位<br>硬件置1来表示PLLI2S输出时钟是否稳定待用<br>0：PLLI2S未稳定<br>1：PLLI2S 已稳定 |
| 26 | PLLI2SEN | PLLI2S 使能<br>软件置位或复位，当进入深度睡眠或待机模式时由硬件复位。<br>0：PLLI2S被关闭<br>1：PLLI2S 被打开 |
| 25 | PLLSTB | PLL 时钟稳定标志位<br>硬件置1来表示PLL输出时钟是否稳定待用<br>0：PLL未稳定<br>1：PLL 已稳定 |

| | | |
|---|---|---|
| 24 | PLLEN | PLL 使能<br>软件置位或复位，当PLL时钟做为系统时钟时该位不能被复位。当进入深度睡眠或待机模式时由硬件复位。<br>0：PLL被关闭<br>1：PLL 被打开 |
| 23:20 | 保留 | 必须保持复位值。 |
| 19 | CKMEN | HXTAL时钟监视器使能<br>0：禁止高速4 ~ 32 MHz晶体振荡器(HXTAL)时钟监视器<br>1：使能高速4 ~ 32 MHz晶体振荡器(HXTAL)时钟监视器<br>当硬件检测到HXTAL时钟被阻塞在低或高状态时，内部硬件自动切换系统时钟到IRC16M时钟。恢复原来系统时钟的方式有以下几种：外部复位，上电复位，软件清CKMIF位。<br>注意：使能HXTAL时钟监视器以后，硬件无视控制位IRC16MEN的状态，自动使能IRC16M时钟。 |
| 18 | HXTALBPS | 高速晶体振荡器(HXTAL)时钟旁路模式使能<br>只有在HXTAL位为0时HXTALBPS位才可写<br>0：禁止HXTAL旁路模式<br>1：使能 HXTAL 旁路模式 HXTAL 输出时钟等于输入时钟 |
| 17 | HXTALSTB | 高速晶体振荡器(HXTAL)时钟稳定标志位<br>硬件置'1'来指示HXTAL振荡器时钟是否稳定待用<br>0：HXTAL振荡器未稳定<br>1：HXTAL 振荡器已稳定 |
| 16 | HXTALEN | 高速晶体振荡器(HXTAL)使能<br>软件置位或复位，如果HXTAL时钟作为系统时钟或者当PLL时钟做为系统时钟时，其做为PLL的输入时钟，该位不能被复位。进入深度睡眠或待机模式时硬件自动复位。<br>0：高速4 ~ 32 MHz晶体振荡器被关闭<br>1：高速 4 ~ 32 MHz 晶体振荡器被打开 |
| 15:8 | IRC16MCALIB[7:0] | 内部16MHz RC振荡器校准值寄存器<br>上电时自动加载这些位 |
| 7:3 | IRC16MADJ[4:0] | 内部 16MHz RC 振荡器时钟调整值<br>这些位由软件置位，最终调整值为 IRC16MADJ[4:0]位域的当前值加上IRC16MCALIB[7:0]位域的值。最终调整值应该调整 IRC16M 到 16 MHz +/- 1%。 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | IRC16MSTB | IRC16M内部16MHz RC振荡器稳定标志位<br>硬件置'1'来指示IRC16M振荡器时钟是否稳定待用<br>0：IRC16M振荡器未稳定<br>1：IRC16M 振荡器已稳定 |
| 0 | IRC16MEN | 内部16MHz RC振荡器使能 |

软件置位或复位，如果IRC16M时钟做为系统时钟时，该位不能被复位。当从深度睡眠或待机模式返回，或当CKMEN置位同时用作系统时钟的HXTAL振荡器发生故障时，该位由硬件置1来启动IRC16M振荡器。

0：内部16 MHz RC振荡器被关闭

1：内部 16 MHz RC 振荡器被打开

### 4.3.2. PLL 寄存器(RCU_PLL)

地址偏移：0x04

复位值：0x2400 3010

配置PLL时钟可参考下列公式：

CK_PLLVCOSRC = CK_PLLSRC / PLLPSC

CK_PLLVCO = CK_PLLVCOSRC × PLLN

CK_PLLP = CK_PLLVCO / PLLP

CK_PLLQ = CK_PLLVCO / PLLQ

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | PLLQ[3:0] | | | | 保留 | PLLSEL | 保留 | | | | PLLP[1:0] | |
| | | | | rw | | | | | rw | | | | | rw | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:24 | PLLQ[3:0] | PLLQ 输出频率的分频系数(PLL VCO 时钟做为输入)<br>当 PLL 被关闭时由软件置位或清零。这些位域用做将 PLL VCO 时钟(CK_PLLVCO)分频生成 PLLQ 输出时钟(CK_PLLQ)。CK_PLLQ 时钟可以被用作 UBSFS/USBHS (48MHz)、TRNG(48MHz)或 SDIO(<=48MHz)模块的时钟源。RCU_PLL 寄存器的 PLLN 位域对 CK_PLLVCO 时钟进行了描述。<br>0000：保留<br>0001：保留<br>0010：CK_PLLQ = CK_PLLVCO / 2<br>0011：CK_PLLQ = CK_PLLVCO / 3<br>0100：CK_PLLQ = CK_PLLVCO / 4<br>…<br>1111：CK_PLLQ = CK_PLLVCO / 15 |
| 23 | 保留 | 必须保持复位值。 |
| 22 | PLLSEL | PLL时钟源选择<br>由软件置位或复位，控制PLL时钟源<br>0：IRC16M时钟被选择为PLL、PLLSAI、PLLI2S时钟的时钟源 |

| | | |
|---|---|---|
| | | 1：HXTAL时钟被选择为PLL、PLLSAI、PLLI2S时钟的时钟源 |
| 21:18 | 保留 | 必须保持复位值。 |
| 17:16 | PLLP[1:0] | PLLP 输出频率分频系数(PLL VCO 时钟做为输入)<br>当 PLL 被关闭时由软件置位或清零。这些位域用做将 PLL VCO 时钟(CK_PLLVCO)分频生成 PLLP 输出时钟(CK_PLLP)。CK_PLLP 时钟可以被用作系统时钟(不超过240MHz)。RCU_PLL 寄存器的 PLLN 位域对 CK_PLLVCO 时钟进行了描述。<br>00：CK_PLLP = CK_PLLVCO / 2<br>01：CK_PLLP = CK_PLLVCO / 4<br>10：CK_PLLP = CK_PLLVCO / 6<br>11：CK_PLLP = CK_PLLVCO / 8 |
| 15 | 保留 | 必须保持复位值。 |
| 14:6 | PLLN[8:0] | PLL VCO 时钟倍频因子<br>当 PLL 被关闭时由软件置位或清零(仅支持全字/半字写操作)。这些位域用做将 PLL VCO 源时钟(CK_PLLVCOSRC)倍频生成 PLL VCO 输出时钟(CK_PLLVCO)。RCU_PLL 寄存器的 PLLPSC 位域对 CK_PLLVCOSRC 时钟进行了描述。<br>注意：CK_PLLVCO 时钟频率范围必须在 100MHz 到 500MHz 之间<br>PLLN 的值必须满足：<br>64<=PLLN<=500(当 RCU_PLLSSCTL 寄存器的 SSCGON=0)<br>69<=PLLN<=500(当 RCU_PLLSSCTL 寄存器的 SSCGON=1/SS_TYPE=0)<br>71<=PLLN<=500(当 RCU_PLLSSCTL 寄存器的 SSCGON=1/SS_TYPE=1)<br>000000000：保留<br>000000001：保留<br>…<br>000111111：保留<br>001000000：CK_PLLVCO = CK_PLLVCOSRC x 64<br>001000001：CK_PLLVCO = CK_PLLVCOSRC x 65<br>…<br>111110100：CK_PLLVCO = CK_PLLVCOSRC x 500<br>111110101：保留<br>…<br>111111111：保留 |
| 5:0 | PLLPSC[5:0] | PLL VCO 源时钟分频器<br>当 PLL 被关闭时由软件置位或清零。这些位域用做将 PLL 源时钟(CK_PLLSRC)分频生成 PLL VCO 源时钟(CK_PLLVCOSRC)、PLLSAI VCO 源时钟(CK_PLLSAIVCOSRC)和 PLLI2S VCO 源时钟(CK_PLLI2SVCOSRC)。RCU_PLL 寄存器的 PLLSEL 位对 CK_PLLSRC 时钟进行了描述。<br>VCO 源时钟频率范围必须在 1MHz 到 2MHz 之间<br>000000：保留.<br>000001：保留<br>000010：CK_PLLSRC / 2<br>000011：CK_PLLSRC / 3<br>… |

111111：CK_PLLSRC / 63

### 4.3.3. 时钟配置寄存器 0(RCU_CFG0)

地址偏移：0x08
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CKOUT1SEL[1:0] | | CKOUT1DIV[2:0] | | | CKOUT0DIV[2:0] | | | I2SSEL | CKOUT0SEL[1:0] | | RTCDIV[4:0] | | | | |
| rw | | rw | | | rw | | | rw | rw | | rw | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| APB2PSC[2:0] | | | APB1PSC[2:0] | | | 保留 | | AHBPSC[3:0] | | | | SCSS[1:0] | | SCS[1:0] | |
| rw | | | rw | | | | | rw | | | | r | | rw | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:30 | CKOUT1SEL[1:0] | CKOUT1时钟源选择<br>由软件置位或清零<br>00：选择系统时钟<br>01：选择 CK_PLLI2SR 时钟<br>10：选择高速晶体振荡器时钟(HXTAL)<br>11：选择CK_PLLP时钟 |
| 29:27 | CKOUT1DIV[2:0] | CK_OUT1 分频器，来降低 CK_OUT1 频率<br>CK_OUT1 时钟源的选择参考 RCU_CFG0 寄存器的 31:30 位<br>0xx：CK_OUT1 不分频<br>100：CK_OUT1 被 2 分频<br>101：CK_OUT1 被 3 分频<br>110：CK_OUT1 被 4 分频<br>111：CK_OUT1 被 5 分频 |
| 26:24 | CKOUT0DIV[2:0] | CK_OUT0 分频器，来降低 CK_OUT0 频率<br>CK_OUT0 时钟源的选择参考 RCU_CFG0 寄存器的 22:21 位<br>0xx：CK_OUT0 不分频<br>100：CK_OUT0 被 2 分频<br>101：CK_OUT0 被 3 分频<br>110：CK_OUT0 被 4 分频<br>111：CK_OUT0 被 5 分频 |
| 23 | I2SSEL | I2S时钟源选择<br>由软件置位或复位，控制I2S时钟源<br>0：选择PLLI2S输出时钟作为I2S源时钟<br>1：选择外部I2S_CKIN引脚输入信号作为I2S源时钟 |
| 22:21 | CKOUT0SEL[1:0] | CKOUT0时钟源选择<br>由软件置位或清零 |

| | | |
|---|---|---|
| | | 00：选择内部 16M RC 振荡器时钟<br>01：选择低速晶体振荡器时钟(LXTAL)<br>10：选择高速晶体振荡器时钟(HXTAL)<br>11：选择CK_PLLP时钟 |
| 20:16 | RTCDIV[4:0] | RTC 时钟分频系数<br>由软件置位或清零。这些位用作将 HXTAL 时钟分频生成 RTC 时钟(不超过 1MHz)<br>00000：无时钟<br>00001：无时钟<br>00010：CK_HXTAL / 2<br>00011：CK_HXTAL / 3<br>…<br>11111：CK_HXTAL / 31 |
| 15:13 | APB2PSC[2:0] | APB2 预分频选择<br>由软件置位或清零，控制 APB2 时钟分频因子。<br>0xx：选择 CK_AHB 时钟不分频<br>100：选择 CK_AHB 时钟 2 分频<br>101：选择 CK_AHB 时钟 4 分频<br>110：选择 CK_AHB 时钟 8 分频<br>111：选择 CK_AHB 时钟 16 分频 |
| 12:10 | APB1PSC[2:0] | APB1 预分频选择<br>由软件置位或清零，控制 APB1 时钟分频因子。<br>0xx：选择 CK_AHB 时钟不分频<br>100：选择 CK_AHB 时钟 2 分频<br>101：选择 CK_AHB 时钟 4 分频<br>110：选择 CK_AHB 时钟 8 分频<br>111：选择 CK_AHB 时钟 16 分频 |
| 9:8 | 保留 | 必须保持复位值。 |
| 7:4 | AHBPSC[3:0] | AHB 预分频选择<br>由软件置位或清零，控制 AHB 时钟分频因子。<br>0xxx：选择 CK_SYS 时钟不分频<br>1000：选择 CK_SYS 时钟 2 分频<br>1001：选择 CK_SYS 时钟 4 分频<br>1010：选择 CK_SYS 时钟 8 分频<br>1011：选择 CK_SYS 时钟 16 分频<br>1100：选择 CK_SYS 时钟 64 分频<br>1101：选择 CK_SYS 时钟 128 分频<br>1110：选择 CK_SYS 时钟 256 分频<br>1111：选择 CK_SYS 时钟 512 分频 |
| 3:2 | SCSS[1:0] | 系统时钟选择状态<br>由硬件置位或清零，标识当前系统时钟的时钟源。<br>00：选择 CK_IRC16M 时钟作为 CK_SYS 时钟源 |

01：选择 CK_HXTAL 时钟作为 CK_SYS 时钟源
10：选择 CK_PLLP 时钟作为 CK_SYS 时钟源
11：保留

| | | |
|---|---|---|
| 1:0 | SCS[1:0] | 系统时钟选择<br>由软件配置选择系统时钟源。由于CK_SYS的改变存在固有的延迟，因此软件应当读SCSS位来确保时钟源切换是否结束。在从深度睡眠或待机模式中返回时，以及当HXTAL直接或间接作为系统时钟同时HXTAL时钟监视器检测到HXTAL故障时，强制选择IRC16M作为系统时钟。<br>00：选择 CK_IRC16M 时钟作为 CK_SYS 时钟源<br>01：选择 CK_HXTAL 时钟作为 CK_SYS 时钟源<br>10：选择 CK_PLLP 时钟作为 CK_SYS 时钟源<br>11：保留 |

### 4.3.4. 时钟中断寄存器(RCU_INT)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | CKMIC | PLLSAI STBIC | PLLI2S STBIC | PLL STBIC | HXTAL STBIC | IRC16M STBIC | LXTAL STBIC | IRC32K STBIC |
| | | | | | | | | w | w | w | w | w | w | w | w |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | PLLSAI STBIE | PLLI2S STBIE | PLL STBIE | HXTAL STBIE | IRC16M STBIE | LXTAL STBIE | IRC32K STBIE | CKMIF | PLLSAI STBIF | PLLI2S STBIF | PLL STBIF | HXTAL STBIF | IRC16M STBIF | LXTAL STBIF | IRC32K STBIF |
| | rw | rw | rw | rw | rw | rw | rw | r | r | r | r | r | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值。 |
| 23 | CKMIC | HXTAL时钟阻塞中断清零<br>软件写 1 复位 CKMIF 标志位<br>0：不复位 CKMIF 标志位<br>1：复位 CKMIF 标志位 |
| 22 | PLLSAISTBIC | PLLSAI 时钟稳定中断清零<br>软件写 1 复位 PLLSAISTBIF 标志位<br>0：不复位 PLLSAISTBIF 标志位<br>1：复位 PLLSAISTBIF 标志位 |
| 21 | PLLI2SSTBIC | PLLI2S 时钟稳定中断清零<br>软件写 1 复位 PLLI2SSTBIF 标志位<br>0：不复位 PLLI2SSTBIF 标志位<br>1：复位 PLLI2SSTBIF 标志位 |

| | | |
|---|---|---|
| 20 | PLLSTBIC | PLL 时钟稳定中断清零<br>软件写 1 复位 PLLSTBIF 标志位<br>0：不复位 PLLSTBIF 标志位<br>1：复位 PLLSTBIF 标志位 |
| 19 | HXTALSTBIC | HXTAL 时钟稳定中断清零<br>软件写 1 复位 HXTALSTBIF 标志位<br>0：不复位 HXTALSTBIF 标志位<br>1：复位 HXTALSTBIF 标志位 |
| 18 | IRC16MSTBIC | IRC16M 时钟稳定中断清零<br>软件写 1 复位 IRC16MSTBIF 标志位<br>0：不复位 IRC16MSTBIF 标志位<br>1：复位 IRC16MSTBIF 标志位 |
| 17 | LXTALSTBIC | LXTAL 时钟稳定中断清零<br>软件写 1 复位 LXTALSTBIF 标志位<br>0：不复位 LXTALSTBIF 标志位<br>1：复位 LXTALSTBIF 标志位 |
| 16 | IRC32KSTBIC | IRC32K 时钟稳定中断清零<br>软件写 1 复位 IRC32KSTBIF 标志位<br>0：不复位 IRC32KSTBIF 标志位<br>1：复位 IRC32KSTBIF 标志位 |
| 15 | 保留 | 必须保持复位值。 |
| 14 | PLLSAISTBIE | PLLSAI 时钟稳定中断使能<br>软件置位和复位来使能/禁止PLLSAI时钟稳定中断<br>0：禁止 PLLSAI 时钟稳定中断<br>1：使能 PLLSAI 时钟稳定中断 |
| 13 | PLLI2SSTBIE | PLLI2S 时钟稳定中断使能<br>软件置位和复位来使能/禁止PLLI2S时钟稳定中断<br>0：禁止 PLLI2S 时钟稳定中断<br>1：使能 PLLI2S 时钟稳定中断 |
| 12 | PLLSTBIE | PLL 时钟稳定中断使能<br>软件置位和复位来使能/禁止PLL时钟稳定中断<br>0：禁止 PLL 时钟稳定中断<br>1：使能 PLL 时钟稳定中断 |
| 11 | HXTALSTBIE | HXTAL 时钟稳定中断使能<br>软件置位和复位来使能/禁止HXTAL时钟稳定中断<br>0：禁止HXTAL时钟稳定中断<br>1：使能 HXTAL 时钟稳定中断 |
| 10 | IRC16MSTBIE | IRC16M 时钟稳定中断使能<br>软件置位和复位来使能/禁止IRC16M时钟稳定中断 |

| | | |
|---|---|---|
| | | 0：禁止IRC16M时钟稳定中断<br>1：使能 IRC16M 时钟稳定中断 |
| 9 | LXTALSTBIE | LXTAL 时钟稳定中断使能<br>软件置位和复位来使能/禁止LXTAL时钟稳定中断<br>0：禁止LXTAL时钟稳定中断<br>1：使能 LXTAL 时钟稳定中断 |
| 8 | IRC32KSTBIE | IRC32K 时钟稳定中断使能<br>软件置位和复位来使能/禁止IRC32K时钟稳定中断<br>0：禁止IRC32K时钟稳定中断<br>1：使能 IRC32K 时钟稳定中断 |
| 7 | CKMIF | HXTAL时钟阻塞中断标志位<br>当HXTAL时钟被阻塞时由硬件置位.<br>软件置位 CKMIC 位时清除该位<br>0：时钟正常运行<br>1：HXTAL 时钟阻塞 |
| 6 | PLLSAISTBIF | PLLSAI 时钟稳定中断标志位<br>当PLLSAI时钟稳定且PLLSAISTBIE位被置1时由硬件置1<br>软件置位 PLLSAISTBIC 位时清除该位<br>0：无PLLSAI时钟稳定中断产生<br>1：产生 PLLSAI 时钟稳定中断 |
| 5 | PLLI2SSTBIF | PLLI2SI 时钟稳定中断标志位<br>当PLLI2S时钟稳定且PLLI2SSTBIE位被置1时由硬件置1<br>软件置位 PLLI2SSTBIC 位时清除该位<br>0：无PLLI2S时钟稳定中断产生<br>1：产生 PLLI2S 时钟稳定中断 |
| 4 | PLLSTBIF | PLL 时钟稳定中断标志位<br>当PLL时钟稳定且PLLSTBIE位被置1时由硬件置1<br>软件置位 PLLSTBIC 位时清除该位<br>0：无PLL时钟稳定中断产生<br>1：产生 PLL 时钟稳定中断 |
| 3 | HXTALSTBIF | HXTAL 时钟稳定中断标志位<br>当高速4~32 MHz晶体振荡器时钟稳定且HXTALSTBIE位被置1时由硬件置1<br>软件置位 HXTALSTBIC 位时清除该位<br>0：无HXTAL时钟稳定中断产生<br>1：产生HXTAL时钟稳定中断 |
| 2 | IRC16MSTBIF | IRC16M 时钟稳定中断标志位<br>当内部16MHz RC振荡器时钟稳定且IRC16MSTBIE位被置1时由硬件置1<br>软件置位 IRC16MSTBIC 位时清除该位<br>0：无IRC16M时钟稳定中断产生<br>1：产生 IRC16M 时钟稳定中断 |

| | | |
|---|---|---|
| 1 | LXTALSTBIF | LXTAL 时钟稳定中断标志位<br>当低速晶体振荡器时钟稳定且LXTALSTBIE位被置1时由硬件置1<br>软件置位 LXTALSTBIC 位时清除该位<br>0：无LXTAL时钟稳定中断产生<br>1：产生 LXTAL 时钟稳定中断 |
| 0 | IRC32KSTBIF | IRC32K 时钟稳定中断标志位<br>当内部32kHz RC振荡器时钟稳定且IRC32KSTBIE位被置1时由硬件置1<br>软件置位 IRC32KSTBIC 位时清除该位<br>0：无IRC32K时钟稳定中断产生<br>1：产生IRC32K时钟稳定中断 |

### 4.3.5. AHB1 复位寄存器(RCU_AHB1RST)

地址偏移：0x10
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | USBHSRST | 保留 | | | ENETRST | 保留 | IPARST | DMA1RST | DMA0RST | 保留 | | | | |
| | | rw | | | | rw | | rw | rw | rw | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | CRCRST | 保留 | | | PIRST | PHRST | PGRST | PFRST | PERST | PDRST | PCRST | PBRST | PARST |
| | | | rw | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:30 | 保留 | 必须保持复位值。 |
| 29 | USBHSRST | USBHS 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 USBHS |
| 28:26 | 保留 | 必须保持复位值。 |
| 25 | ENETRST | Ethernet 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 ENET |
| 24 | 保留 | 必须保持复位值。 |
| 23 | IPARST | IPA 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 IPA |

| | | |
|---|---|---|
| 22 | DMA1RST | DMA1 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 DMA1 |
| 21 | DMA0RST | DMA0 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 DMA0 |
| 20:13 | 保留 | 必须保持复位值。 |
| 12 | CRCRST | CRC 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 CRC |
| 11:9 | 保留 | 必须保持复位值。 |
| 8 | PIRST | GPIO 端口 I 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 GPIO 端口 I |
| 7 | PHRST | GPIO 端口 H 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 GPIO 端口 H |
| 6 | PGRST | GPIO 端口 G 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 GPIO 端口 G |
| 5 | PFRST | GPIO 端口 F 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 GPIO 端口 F |
| 4 | PERST | GPIO 端口 E 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 GPIO 端口 E |
| 3 | PDRST | GPIO 端口 D 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 GPIO 端口 D |
| 2 | PCRST | GPIO 端口 C 复位 |

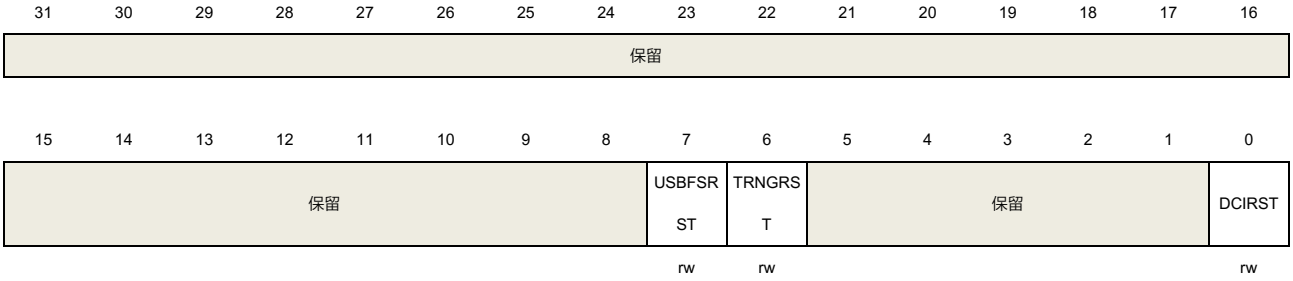
由软件置位或复位
0：无作用
1：复位 GPIO 端口 C

| | | |
|---|---|---|
| 1 | PBRST | GPIO 端口 B 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 GPIO 端口 B |
| 0 | PARST | GPIO 端口 A 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 GPIO 端口 A |

### 4.3.6. AHB2 复位寄存器(RCU_AHB2RST)

地址偏移：0x14
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7 | USBFSRST | USBFS 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 USBFS |
| 6 | TRNGRST | TRNG 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TRNG |
| 5:1 | 保留 | 必须保持复位值。 |
| 0 | DCIRST | DCI 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 DCI |

### 4.3.7. AHB3 复位寄存器(RCU_AHB3RST)

地址偏移：0x18
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:1 | 保留 | 必须保持复位值。 |
| 0 | EXMCRST | EXMC 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 EXMC |

### 4.3.8. APB1 复位寄存器(RCU_APB1RST)

地址偏移：0x20
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| UART7RST | UART6RST | DACRST | PMURST | 保留 | CAN1RST | CAN0RST | 保留 | I2C2RST | I2C1RST | I2C0RST | UART4RST | UART3RST | USART2RST | USART1RST | 保留 |
| rw | rw | rw | rw | | rw | rw | | rw | rw | rw | rw | rw | rw | rw | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SPI2RST | SPI1RST | 保留 | | WWDGTRST | 保留 | | TIMER13RST | TIMER12RST | TIMER11RST | TIMER6RST | TIMER5RST | TIMER4RST | TIMER3RST | TIMER2RST | TIMER1RST |
| rw | rw | | | rw | | | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | UART7RST | UART7 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 UART7 |
| 30 | UART6RST | UART6 复位<br>由软件置位或复位 |

| | | |
|---|---|---|
| | | 0：无作用<br>1：复位 UART6 |
| 29 | DACRST | DAC 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 DAC |
| 28 | PMURST | PMU 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 PMU |
| 27 | 保留 | 必须保持复位值。 |
| 26 | CAN1RST | CAN1 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 CAN1 |
| 25 | CAN0RST | CAN0 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 CAN0 |
| 24 | 保留 | 必须保持复位值。 |
| 23 | I2C2RST | I2C2 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 I2C2 |
| 22 | I2C1RST | I2C1 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 I2C1 |
| 21 | I2C0RST | I2C0 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 I2C0 |
| 20 | UART4RST | UART4 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 UART4 |
| 19 | UART3RST | UART3 复位<br>由软件置位或复位<br>0：无作用 |

| | | |
|---|---|---|
| | | 1：复位 UART3 |
| 18 | USART2RST | USART2 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 USART2 |
| 17 | USART1RST | USART1 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 USART1 |
| 16 | 保留 | 必须保持复位值。 |
| 15 | SPI2RST | SPI2 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 SPI2 |
| 14 | SPI1RST | SPI1 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 SPI1 |
| 13:12 | 保留 | 必须保持复位值。 |
| 11 | WWDGTRST | WWDGT 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 WWDGT |
| 10:9 | 保留 | 必须保持复位值。 |
| 8 | TIMER13RST | TIMER13 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER13 |
| 7 | TIMER12RST | TIMER12 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER12 |
| 6 | TIMER11RST | TIMER11 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER11 |
| 5 | TIMER6RST | TIMER6 复位<br>由软件置位或复位 |

| | | |
|---|---|---|
| | | 0：无作用<br>1：复位 TIMER6 |
| 4 | TIMER5RST | TIMER5 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER5 |
| 3 | TIMER4RST | TIMER4 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER4 |
| 2 | TIMER3RST | TIMER3 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER3 |
| 1 | TIMER2RST | TIMER2 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER2 |
| 0 | TIMER1RST | TIMER1 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER1 |

### 4.3.9. APB2 复位寄存器(RCU_APB2RST)

地址偏移：0x24
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | TLIRST | 保留 | | | | SPI5RST | SPI4RST | 保留 | TIMER10RST | TIMER9RST | TIMER8RST |
| | | | | | rw | | | | | rw | rw | | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | SYSCFGRST | SPI3RST | SPI0RST | SDIORST | 保留 | | ADCRST | 保留 | | USART5RST | USART0RST | 保留 | | TIMER7RST | TIMER0RST |
| | rw | rw | rw | rw | | | rw | | | rw | rw | | | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:27 | 保留 | 必须保持复位值。 |
| 26 | TLIRST | TLI 复位 |

| | | |
|---|---|---|
| | | 由软件置位或复位<br>0：无作用<br>1：复位 TLI |
| 25:22 | 保留 | 必须保持复位值。 |
| 21 | SPI5RST | SPI5 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 SPI5 |
| 20 | SPI4RST | SPI4 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 SPI4 |
| 19 | 保留 | 必须保持复位值。 |
| 18 | TIMER10RST | TIMER10 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER10 |
| 17 | TIMER9RST | TIMER9 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER9 |
| 16 | TIMER8RST | TIMER8 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER8 |
| 15 | 保留 | 必须保持复位值。 |
| 14 | SYSCFGRST | SYSCFG 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 SYSCFG |
| 13 | SPI3RST | SPI3 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 SPI3 |
| 12 | SPI0RST | SPI0 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 SPI0 |

| | | |
|---|---|---|
| 11 | SDIORST | SDIO 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 SDIO |
| 10:9 | 保留 | 必须保持复位值。 |
| 8 | ADCRST | ADC 复位<br>由软件置位或复位<br>0：无作用<br>1：复位所有 ADC |
| 7:6 | 保留 | 必须保持复位值。 |
| 5 | USART5RST | USART5 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 USART5 |
| 4 | USART0RST | USART0 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 USART0 |
| 3:2 | 保留 | 必须保持复位值。 |
| 1 | TIMER7RST | TIMER7 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER7 |
| 0 | TIMER0RST | TIMER0 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 TIMER0 |

### 4.3.10. AHB1 使能寄存器(RCU_AHB1EN)

地址偏移：0x30

复位值：0x0010 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | USBHSULPIEN | USBHSEN | ENETPTPEN | ENETRXEN | ENETTXEN | ENETEN | 保留 | IPAEN | DMA1EN | DMA0EN | TCMSRAMEN | 保留 | BKPSRAMEN | 保留 | |
| | rw | rw | rw | rw | rw | rw | | rw | rw | rw | rw | | rw | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | CRCEN | 保留 | | | PIEN | PHEN | PGEN | PFEN | PEEN | PDEN | PCEN | PBEN | PAEN |

rw rw rw rw rw rw rw rw rw rw

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31 | 保留 | 必须保持复位值。 |
| 30 | USBHSULPIEN | USBHS ULPI 时钟使能<br>由软件置位或复位<br>0：关闭 USBHS ULPI 时钟<br>1：开启 USBHS ULPI 时钟 |
| 29 | USBHSEN | USBHS 时钟使能<br>由软件置位或复位<br>0：关闭 USBHS 时钟<br>1：开启 USBHS 时钟 |
| 28 | ENETPTPEN | 以太网 PTP 时钟使能<br>由软件置位或复位<br>0：关闭以太网 PTP 时钟<br>1：开启以太网 PTP 时钟 |
| 27 | ENETRXEN | 以太网 RX 时钟使能<br>由软件置位或复位<br>0：关闭以太网 RX 时钟<br>1：开启以太网 RX 时钟 |
| 26 | ENETTXEN | 以太网 TX 时钟使能<br>由软件置位或复位<br>0：关闭以太网 TX 时钟<br>1：开启以太网 TX 时钟 |
| 25 | ENETEN | 以太网时钟使能<br>由软件置位或复位<br>0：关闭以太网时钟<br>1：开启以太网时钟 |
| 24 | 保留 | 必须保持复位值。 |
| 23 | IPAEN | IPA 时钟使能<br>由软件置位或复位<br>0：关闭 IPA 时钟<br>1：开启 IPA 时钟 |
| 22 | DMA1EN | DMA1 时钟使能<br>由软件置位或复位<br>0：关闭 DMA1 时钟<br>1：开启 DMA1 时钟 |
| 21 | DMA0EN | DMA0 时钟使能<br>由软件置位或复位 |

| | | |
|---|---|---|
| | | 0：关闭 DMA0 时钟<br>1：开启 DMA0 时钟 |
| 20 | TCMSRAMEN | TCMSRAM 时钟使能<br>由软件置位或复位<br>0：关闭 TCMSRAM 时钟<br>1：开启 TCMSRAM 时钟 |
| 19 | 保留 | 必须保持复位值。 |
| 18 | BKPSRAMEN | BKPSRAM 时钟使能<br>由软件置位或复位<br>0：关闭 BKPSRAM 时钟<br>1：开启 BKPSRAM 时钟 |
| 17:13 | 保留 | 必须保持复位值。 |
| 12 | CRCEN | CRC 时钟使能<br>由软件置位或复位<br>0：关闭 CRC 时钟<br>1：开启 CRC 时钟 |
| 11:9 | 保留 | 必须保持复位值。 |
| 8 | PIEN | GPIO 端口 I 时钟使能<br>由软件置位或复位<br>0：关闭 GPIO 端口 I 时钟<br>1：开启 GPIO 端口 I 时钟 |
| 7 | PHEN | GPIO 端口 H 时钟使能<br>由软件置位或复位<br>0：关闭 GPIO 端口 H 时钟<br>1：开启 GPIO 端口 H 时钟 |
| 6 | PGEN | GPIO 端口 G 时钟使能<br>由软件置位或复位<br>0：关闭 GPIO 端口 G 时钟<br>1：开启 GPIO 端口 G 时钟 |
| 5 | PFEN | GPIO 端口 F 时钟使能<br>由软件置位或复位<br>0：关闭 GPIO 端口 F 时钟<br>1：开启 GPIO 端口 F 时钟 |
| 4 | PEEN | GPIO 端口 E 时钟使能<br>由软件置位或复位<br>0：关闭 GPIO 端口 E 时钟<br>1：开启 GPIO 端口 E 时钟 |
| 3 | PDEN | GPIO 端口 D 时钟使能 |

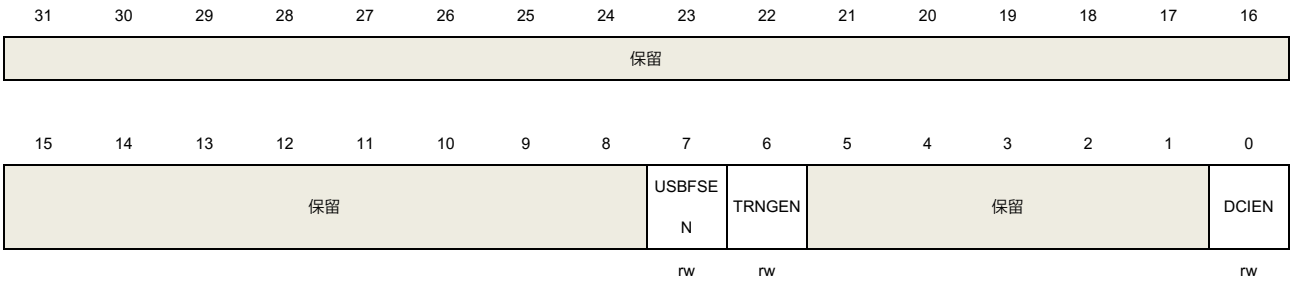
由软件置位或复位
0：关闭 GPIO 端口 D 时钟
1：开启 GPIO 端口 D 时钟

| | | |
|---|---|---|
| 2 | PCEN | GPIO 端口 C 时钟使能<br>由软件置位或复位<br>0：关闭 GPIO 端口 C 时钟<br>1：开启 GPIO 端口 C 时钟 |
| 1 | PBEN | GPIO 端口 B 时钟使能<br>由软件置位或复位<br>0：关闭 GPIO 端口 B 时钟<br>1：开启 GPIO 端口 B 时钟 |
| 0 | PAEN | GPIO 端口 A 时钟使能<br>由软件置位或复位<br>0：关闭 GPIO 端口 A 时钟<br>1：开启 GPIO 端口 A 时钟 |

### 4.3.11. AHB2 使能寄存器(RCU_AHB2EN)

地址偏移：0x34
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7 | USBFSEN | USBFS 时钟使能<br>由软件置位或复位<br>0：关闭 USBFS 时钟<br>1：开启 USBFS 时钟 |
| 6 | TRNGEN | TRNG 时钟使能<br>由软件置位或复位<br>0：关闭 TRNG 时钟<br>1：开启 TRNG 时钟 |
| 5:1 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 0 | DCIEN | DCI 时钟使能<br>由软件置位或复位<br>0：关闭 DCI 时钟<br>1：开启 DCI 时钟 |

### 4.3.12. AHB3 使能寄存器(RCU_AHB3EN)

地址偏移：0x38
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:1 | 保留 | 必须保持复位值。 |
| 0 | EXMCEN | EXMC 时钟使能<br>由软件置位或复位<br>0：关闭 EXMC 时钟<br>1：开启 EXMC 时钟 |

### 4.3.13. APB1 使能寄存器(RCU_APB1EN)

地址偏移：0x40
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| UART7EN | UART6EN | DACEN | PMUEN | 保留 | CAN1EN | CAN0EN | 保留 | I2C2EN | I2C1EN | I2C0EN | UART4EN | UART3EN | USART2EN | USART1EN | 保留 |
| rw | rw | rw | rw | | rw | rw | | rw | rw | rw | rw | rw | rw | rw | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SPI2EN | SPI1EN | 保留 | | WWDGTEN | 保留 | | TIMER13EN | TIMER12EN | TIMER11EN | TIMER6EN | TIMER5EN | TIMER4EN | TIMER3EN | TIMER2EN | TIMER1EN |
| rw | rw | | | rw | | | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | UART7EN | UART7 时钟使能<br>由软件置位或复位 |

| | | |
|---|---|---|
| | | 0：关闭 UART7 时钟<br>1：开启 UART7 时钟 |
| 30 | UART6EN | UART6 时钟使能<br>由软件置位或复位<br>0：关闭 UART6 时钟<br>1：开启 UART6 时钟 |
| 29 | DACEN | DAC 时钟使能<br>由软件置位或复位<br>0：关闭 DAC 时钟<br>1：开启 DAC 时钟 |
| 28 | PMUEN | PMU 时钟使能<br>由软件置位或复位<br>0：关闭 PMU 时钟<br>1：开启 PMU 时钟 |
| 27 | 保留 | 必须保持复位值。 |
| 26 | CAN1EN | CAN1 时钟使能<br>由软件置位或复位<br>0：关闭 CAN1 时钟<br>1：开启 CAN1 时钟 |
| 25 | CAN0EN | CAN0 时钟使能<br>由软件置位或复位<br>0：关闭 CAN0 时钟<br>1：开启 CAN0 时钟 |
| 24 | 保留 | 必须保持复位值。 |
| 23 | I2C2EN | I2C2 时钟使能<br>由软件置位或复位<br>0：关闭 I2C2 时钟<br>1：开启 I2C2 时钟 |
| 22 | I2C1EN | I2C1 时钟使能<br>由软件置位或复位<br>0：关闭 I2C1 时钟<br>1：开启 I2C1 时钟 |
| 21 | I2C0EN | I2C0 时钟使能<br>由软件置位或复位<br>0：关闭 I2C0 时钟<br>1：开启 I2C0 时钟 |
| 20 | UART4EN | UART4 时钟使能<br>由软件置位或复位<br>0：关闭 UART4 时钟 |

| | | |
|---|---|---|
| | | 1：开启 UART4 时钟 |
| 19 | UART3EN | UART3 时钟使能<br>由软件置位或复位<br>0：关闭 UART3 时钟<br>1：开启 UART3 时钟 |
| 18 | USART2EN | USART2 时钟使能<br>由软件置位或复位<br>0：关闭 USART2 时钟<br>1：开启 USART2 时钟 |
| 17 | USART1EN | USART1 时钟使能<br>由软件置位或复位<br>0：关闭 USART1 时钟<br>1：开启 USART1 时钟 |
| 16 | 保留 | 必须保持复位值。 |
| 15 | SPI2EN | SPI2 时钟使能<br>由软件置位或复位<br>0：关闭 SPI2 时钟<br>1：开启 SPI2 时钟 |
| 14 | SPI1EN | SPI1 时钟使能<br>由软件置位或复位<br>0：关闭 SPI1 时钟<br>1：开启 SPI1 时钟 |
| 13:12 | 保留 | 必须保持复位值。 |
| 11 | WWDGTEN | WWDGT 时钟使能<br>由软件置位或复位<br>0：关闭 WWDGT 时钟<br>1：开启 WWDGT 时钟 |
| 10:9 | 保留 | 必须保持复位值。 |
| 8 | TIMER13EN | TIMER13 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER13 时钟<br>1：开启 TIMER13 时钟 |
| 7 | TIMER12EN | TIMER12 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER12 时钟<br>1：开启 TIMER12 时钟 |
| 6 | TIMER11EN | TIMER11 时钟使能<br>由软件置位或复位 |

0：关闭 TIMER11 时钟
1：开启 TIMER11 时钟

| | | |
|---|---|---|
| 5 | TIMER6EN | TIMER6 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER6 时钟<br>1：开启 TIMER6 时钟 |
| 4 | TIMER5EN | TIMER5 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER5 时钟<br>1：开启 TIMER5 时钟 |
| 3 | TIMER4EN | TIMER4 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER4 时钟<br>1：开启 TIMER4 时钟 |
| 2 | TIMER3EN | TIMER3 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER3 时钟<br>1：开启 TIMER3 时钟 |
| 1 | TIMER2EN | TIMER2 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER2 时钟<br>1：开启 TIMER2 时钟 |
| 0 | TIMER1EN | TIMER1 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER1 时钟<br>1：开启 TIMER1 时钟 |

### 4.3.14. APB2 使能寄存器(RCU_APB2EN)

地址偏移：0x44
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | TLIEN | 保留 | | | | SPI5EN | SPI4EN | 保留 | TIMER10EN | TIMER9EN | TIMER8EN |
| | | | | | rw | | | | | rw | rw | | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | SYSCFGEN | SPI3EN | SPI0EN | SDIOEN | ADC2EN | ADC1EN | ADC0EN | 保留 | | USART5EN | USART0EN | 保留 | | TIMER7EN | TIMER0EN |
| | rw | rw | rw | rw | rw | rw | rw | | | rw | rw | | | rw | rw |

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:27 | 保留 | 必须保持复位值。 |
| 26 | TLIEN | TLI 时钟使能<br>由软件置位或复位<br>0：关闭 TLI 时钟<br>1：开启 TLI 时钟 |
| 25:22 | 保留 | 必须保持复位值。 |
| 21 | SPI5EN | SPI5 时钟使能<br>由软件置位或复位<br>0：关闭 SPI5 时钟<br>1：开启 SPI5 时钟 |
| 20 | SPI4EN | SPI4 时钟使能<br>由软件置位或复位<br>0：关闭 SPI4 时钟<br>1：开启 SPI4 时钟 |
| 19 | 保留 | 必须保持复位值。 |
| 18 | TIMER10EN | TIMER10 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER10 时钟<br>1：开启 TIMER10 时钟 |
| 17 | TIMER9EN | TIMER9 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER9 时钟<br>1：开启 TIMER9 时钟 |
| 16 | TIMER8EN | TIMER8 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER8 时钟<br>1：开启 TIMER8 时钟 |
| 15 | 保留 | 必须保持复位值。 |
| 14 | SYSCFGEN | SYSCFG 时钟使能<br>由软件置位或复位<br>0：关闭 SYSCFG 时钟<br>1：开启 SYSCFG 时钟 |
| 13 | SPI3EN | SPI3 时钟使能<br>由软件置位或复位<br>0：关闭 SPI3 时钟<br>1：开启 SPI3 时钟 |

| | | |
|---|---|---|
| 12 | SPI0EN | SPI0 时钟使能<br>由软件置位或复位<br>0：关闭 SPI0 时钟<br>1：开启 SPI0 时钟 |
| 11 | SDIOEN | SDIO 时钟使能<br>由软件置位或复位<br>0：关闭 SDIO 时钟<br>1：开启 SDIO 时钟 |
| 10 | ADC2EN | ADC2 时钟使能<br>由软件置位或复位<br>0：关闭 ADC2 时钟<br>1：开启 ADC2 时钟 |
| 9 | ADC1EN | ADC1 时钟使能<br>由软件置位或复位<br>0：关闭 ADC1 时钟<br>1：开启 ADC1 时钟 |
| 8 | ADC0EN | ADC0 时钟使能<br>由软件置位或复位<br>0：关闭 ADC0 时钟<br>1：开启 ADC0 时钟 |
| 7:6 | 保留 | 必须保持复位值。 |
| 5 | USART5EN | USART5 时钟使能<br>由软件置位或复位<br>0：关闭 USART5 时钟<br>1：开启 USART5 时钟 |
| 4 | USART0EN | USART0 时钟使能<br>由软件置位或复位<br>0：关闭 USART0 时钟<br>1：开启 USART0 时钟 |
| 3:2 | 保留 | 必须保持复位值。 |
| 1 | TIMER7EN | TIMER7 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER7 时钟<br>1：开启 TIMER7 时钟 |
| 0 | TIMER0EN | TIMER0 时钟使能<br>由软件置位或复位<br>0：关闭 TIMER0 时钟<br>1：开启 TIMER0 时钟 |

### 4.3.15. AHB1 睡眠模式使能寄存器(RCU_AHB1SPEN)

地址偏移：0x50

复位值：0x7EEF 91FF

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | USBHSULPISPEN | USBHSSPEN | ENETPTPSPEN | ENETRXSPEN | ENETTXSPEN | ENETSPEN | 保留 | IPASPEN | DMA1SPEN | DMA0SPEN | 保留 | SRAM2SPEN | BKPSRAMSPEN | SRAM1SPEN | SRAM0SPEN |
| | rw | rw | rw | rw | rw | rw | | rw | rw | rw | | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FMCSPEN | 保留 | | CRCSPEN | 保留 | | | PISPEN | PHSPEN | PGSPEN | PFSPEN | PESPEN | PDSPEN | PCSPEN | PBSPEN | PASPEN |
| rw | | | rw | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30 | USBHSULPISPEN | 在睡眠模式下 USBHS ULPI 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 USBHS ULPI 时钟<br>1：在睡眠模式下开启 USBHS ULPI 时钟 |
| 29 | USBHSSPEN | 在睡眠模式下 USBHS 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 USBHS 时钟<br>1：在睡眠模式下开启 USBHS 时钟 |
| 28 | ENETPTPSPEN | 在睡眠模式下以太网 PTP 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭以太网 PTP 时钟<br>1：在睡眠模式下开启以太网 PTP 时钟 |
| 27 | ENETRXSPEN | 在睡眠模式下以太网 RX 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭以太网 RX 时钟<br>1：在睡眠模式下开启以太网 RX 时钟 |
| 26 | ENETTXSPEN | 在睡眠模式下以太网 TX 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭以太网 TX 时钟<br>1：在睡眠模式下开启以太网 TX 时钟 |
| 25 | ENETSPEN | 在睡眠模式下以太网时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭以太网时钟<br>1：在睡眠模式下开启以太网时钟 |

| | | |
|---|---|---|
| 24 | 保留 | 必须保持复位值。 |
| 23 | IPASPEN | 在睡眠模式下 IPA 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 IPA 时钟<br>1：在睡眠模式下开启 IPA 时钟 |
| 22 | DMA1SPEN | 在睡眠模式下 DMA1 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 DMA1 时钟<br>1：在睡眠模式下开启 DMA1 时钟 |
| 21 | DMA0SPEN | 在睡眠模式下 DMA0 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 DMA0 时钟<br>1：在睡眠模式下开启 DMA0 时钟 |
| 20 | 保留 | 必须保持复位值。 |
| 19 | SRAM2SPEN | 在睡眠模式下 SRAM2 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 SRAM2 时钟<br>1：在睡眠模式下开启 SRAM2 时钟 |
| 18 | BKPSRAMSPEN | 在睡眠模式下 BKPSRAM 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 BKPSRAM 时钟<br>1：在睡眠模式下开启 BKPSRAM 时钟 |
| 17 | SRAM1SPEN | 在睡眠模式下 SRAM1 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 SRAM1 时钟<br>1：在睡眠模式下开启 SRAM1 时钟 |
| 16 | SRAM0SPEN | 在睡眠模式下 SRAM0 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 SRAM0 时钟<br>1：在睡眠模式下开启 SRAM0 时钟 |
| 15 | FMCSPEN | 在睡眠模式下 FMC 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 FMC 时钟<br>1：在睡眠模式下开启 FMC 时钟 |
| 14:13 | 保留 | 必须保持复位值。 |
| 12 | CRCSPEN | 在睡眠模式下 CRC 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 CRC 时钟 |

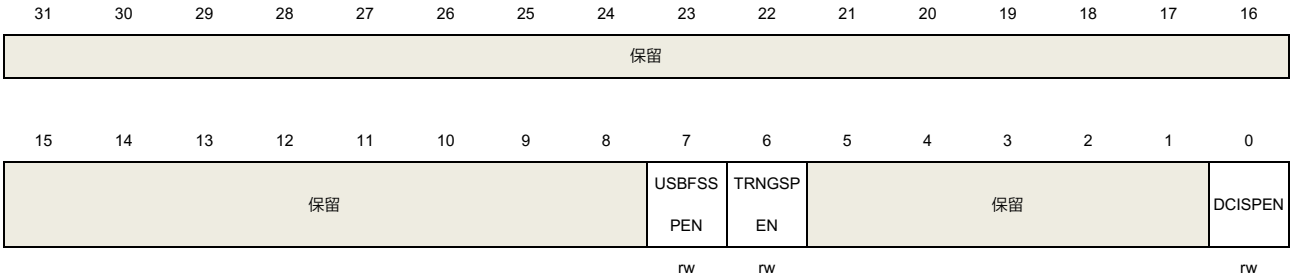
| | | |
|---|---|---|
| | | 1：在睡眠模式下开启 CRC 时钟 |
| 11:9 | 保留 | 必须保持复位值。 |
| 8 | PISPEN | 在睡眠模式下 GPIO 端口 I 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 GPIO 端口 I 时钟<br>1：在睡眠模式下开启 GPIO 端口 I 时钟 |
| 7 | PHSPEN | 在睡眠模式下 GPIO 端口 H 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 GPIO 端口 H 时钟<br>1：在睡眠模式下开启 GPIO 端口 H 时钟 |
| 6 | PGSPEN | 在睡眠模式下 GPIO 端口 G 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 GPIO 端口 G 时钟<br>1：在睡眠模式下开启 GPIO 端口 G 时钟 |
| 5 | PFSPEN | 在睡眠模式下 GPIO 端口 F 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 GPIO 端口 F 时钟<br>1：在睡眠模式下开启 GPIO 端口 F 时钟 |
| 4 | PESPEN | 在睡眠模式下 GPIO 端口 E 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 GPIO 端口 E 时钟<br>1：在睡眠模式下开启 GPIO 端口 E 时钟 |
| 3 | PDSPEN | 在睡眠模式下 GPIO 端口 D 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 GPIO 端口 D 时钟<br>1：在睡眠模式下开启 GPIO 端口 D 时钟 |
| 2 | PCSPEN | 在睡眠模式下 GPIO 端口 C 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 GPIO 端口 C 时钟<br>1：在睡眠模式下开启 GPIO 端口 C 时钟 |
| 1 | PBSPEN | 在睡眠模式下 GPIO 端口 B 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 GPIO 端口 B 时钟<br>1：在睡眠模式下开启 GPIO 端口 B 时钟 |
| 0 | PASPEN | 在睡眠模式下 GPIO 端口 A 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 GPIO 端口 A 时钟<br>1：在睡眠模式下开启 GPIO 端口 A 时钟 |

### 4.3.16. AHB2 睡眠模式使能寄存器(RCU_AHB2SPEN)

地址偏移：0x54

复位值：0x0000 00C1

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



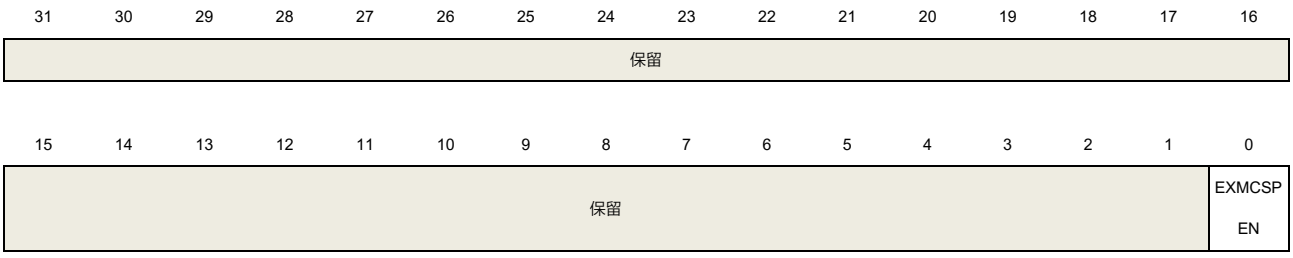

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7 | USBFSSPEN | 在睡眠模式下 USBFS 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 USBFS 时钟<br>1：在睡眠模式下开启 USBFS 时钟 |
| 6 | TRNGSPEN | 在睡眠模式下 TRNG 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TRNG 时钟<br>1：在睡眠模式下开启 TRNG 时钟 |
| 5:1 | 保留 | 必须保持复位值。 |
| 0 | DCISPEN | 在睡眠模式下 DCI 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 DCI 时钟<br>1：在睡眠模式下开启 DCI 时钟 |

### 4.3.17. AHB3 睡眠模式使能寄存器(RCU_AHB3SPEN)

地址偏移：0x58

复位值：0x0000 0001

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



rw

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:1 | 保留 | 必须保持复位值。 |
| 0 | EXMCSPEN | 在睡眠模式下 EXMC 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 EXMC 时钟<br>1：在睡眠模式下开启 EXMC 时钟 |

### 4.3.18. APB1 睡眠模式使能寄存器(RCU_APB1SPEN)

地址偏移：0x60
复位值：0xF6FE C9FF

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| UART7SPEN | UART6SPEN | DACSPEN | PMUSPEN | 保留 | CAN1SPEN | CAN0SPEN | 保留 | I2C2SPEN | I2C1SPEN | I2C0SPEN | UART4SPEN | UART3SPEN | USART2SPEN | USART1SPEN | 保留 |
| rw | rw | rw | rw | | rw | rw | | rw | rw | rw | rw | rw | rw | rw | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SPI2SPEN | SPI1SPEN | 保留 | | WWDGTSPEN | 保留 | | TIMER13SPEN | TIMER12SPEN | TIMER11SPEN | TIMER6SPEN | TIMER5SPEN | TIMER4SPEN | TIMER3SPEN | TIMER2SPEN | TIMER1SPEN |
| rw | rw | | | rw | | | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | UART7SPEN | 在睡眠模式下 UART7 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 UART7 时钟<br>1：在睡眠模式下开启 UART7 时钟 |
| 30 | UART6SPEN | 在睡眠模式下 UART6 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 UART6 时钟<br>1：在睡眠模式下开启 UART6 时钟 |
| 29 | DACSPEN | 在睡眠模式下 DAC 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 DAC 时钟<br>1：在睡眠模式下开启 DAC 时钟 |
| 28 | PMUSPEN | 在睡眠模式下 PMU 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 PMU 时钟<br>1：在睡眠模式下开启 PMU 时钟 |

| | | |
|---|---|---|
| 27 | 保留 | 必须保持复位值。 |
| 26 | CAN1SPEN | 在睡眠模式下 CAN1 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 CAN1 时钟<br>1：在睡眠模式下开启 CAN1 时钟 |
| 25 | CAN0SPEN | 在睡眠模式下 CAN0 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 CAN0 时钟<br>1：在睡眠模式下开启 CAN0 时钟 |
| 24 | 保留 | 必须保持复位值。 |
| 23 | I2C2SPEN | 在睡眠模式下 I2C2 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 I2C2 时钟<br>1：在睡眠模式下开启 I2C2 时钟 |
| 22 | I2C1SPEN | 在睡眠模式下 I2C1 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 I2C1 时钟<br>1：在睡眠模式下开启 I2C1 时钟 |
| 21 | I2C0SPEN | 在睡眠模式下 I2C0 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 I2C0 时钟<br>1：在睡眠模式下开启 I2C0 时钟 |
| 20 | UART4SPEN | 在睡眠模式下 UART4 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 UART4 时钟<br>1：在睡眠模式下开启 UART4 时钟 |
| 19 | UART3SPEN | 在睡眠模式下 UART3 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 UART3 时钟<br>1：在睡眠模式下开启 UART3 时钟 |
| 18 | USART2SPEN | 在睡眠模式下 USART2 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 USART2 时钟<br>1：在睡眠模式下开启 USART2 时钟 |
| 17 | USART1SPEN | 在睡眠模式下 USART1 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 USART1 时钟<br>1：在睡眠模式下开启 USART1 时钟 |

| | | |
|---|---|---|
| 16 | 保留 | 必须保持复位值。 |
| 15 | SPI2SPEN | 在睡眠模式下 SPI2 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 SPI2 时钟<br>1：在睡眠模式下开启 SPI2 时钟 |
| 14 | SPI1SPEN | 在睡眠模式下 SPI1 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 SPI1 时钟<br>1：在睡眠模式下开启 SPI1 时钟 |
| 13:12 | 保留 | 必须保持复位值。 |
| 11 | WWDGTSPEN | 在睡眠模式下 WWDGT 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 WWDGT 时钟<br>1：在睡眠模式下开启 WWDGT 时钟 |
| 10:9 | 保留 | 必须保持复位值。 |
| 8 | TIMER13SPEN | 在睡眠模式下 TIMER13 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER13 时钟<br>1：在睡眠模式下开启 TIMER13 时钟 |
| 7 | TIMER12SPEN | 在睡眠模式下 TIMER12 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER12 时钟<br>1：在睡眠模式下开启 TIMER12 时钟 |
| 6 | TIMER11SPEN | 在睡眠模式下 TIMER11 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER11 时钟<br>1：在睡眠模式下开启 TIMER11 时钟 |
| 5 | TIMER6SPEN | 在睡眠模式下 TIMER6 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER6 时钟<br>1：在睡眠模式下开启 TIMER6 时钟 |
| 4 | TIMER5SPEN | 在睡眠模式下 TIMER5 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER5 时钟<br>1：在睡眠模式下开启 TIMER5 时钟 |
| 3 | TIMER4SPEN | 在睡眠模式下 TIMER4 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER4 时钟 |

| | | |
|---|---|---|
| | | 1：在睡眠模式下开启 TIMER4 时钟 |
| 2 | TIMER3SPEN | 在睡眠模式下 TIMER3 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER3 时钟<br>1：在睡眠模式下开启 TIMER3 时钟 |
| 1 | TIMER2SPEN | 在睡眠模式下 TIMER2 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER2 时钟<br>1：在睡眠模式下开启 TIMER2 时钟 |
| 0 | TIMER1SPEN | 在睡眠模式下 TIMER1 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER1 时钟<br>1：在睡眠模式下开启 TIMER1 时钟 |

## 4.3.19. APB2 睡眠模式使能寄存器(RCU_APB2SPEN)

地址偏移：0x64

复位值：0x0477 7F33

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | TLISPEN | 保留 | | | | SPI5SPEN | SPI4SPEN | 保留 | TIMER10SPEN | TIMER9SPEN | TIMER8SPEN |
| | | | | | rw | | | | | rw | rw | | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | SYSCFGSPEN | SPI3SPEN | SPI0SPEN | SDIOSPEN | ADC2SPEN | ADC1SPEN | ADC0SPEN | 保留 | | USART5SPEN | USART0SPEN | 保留 | | TIMER7SPEN | TIMER0SPEN |
| | rw | rw | rw | rw | rw | rw | rw | | | rw | rw | | | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:27 | 保留 | 必须保持复位值。 |
| 26 | TLISPEN | 在睡眠模式下 TLI 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TLI 时钟<br>1：在睡眠模式下开启 TLI 时钟 |
| 25:22 | 保留 | 必须保持复位值。 |
| 21 | SPI5SPEN | 在睡眠模式下 SPI5 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 SPI5 时钟<br>1：在睡眠模式下开启 SPI5 时钟 |
| 20 | SPI4SPEN | 在睡眠模式下 SPI4 时钟使能 |

| | | |
|---|---|---|
| | | 由软件置位或复位<br>0：在睡眠模式下关闭 SPI4 时钟<br>1：在睡眠模式下开启 SPI4 时钟 |
| 19 | 保留 | 必须保持复位值。 |
| 18 | TIMER10SPEN | 在睡眠模式下 TIMER10 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER10 时钟<br>1：在睡眠模式下开启 TIMER10 时钟 |
| 17 | TIMER9SPEN | 在睡眠模式下 TIMER9 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER9 时钟<br>1：在睡眠模式下开启 TIMER9 时钟 |
| 16 | TIMER8SPEN | 在睡眠模式下 TIMER8 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER8 时钟<br>1：在睡眠模式下开启 TIMER8 时钟 |
| 15 | 保留 | 必须保持复位值 |
| 14 | SYSCFGSPEN | 在睡眠模式下 SYSCFG 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 SYSCFG 时钟<br>1：在睡眠模式下开启 SYSCFG 时钟 |
| 13 | SPI3SPEN | 在睡眠模式下 SPI3 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 SPI3 时钟<br>1：在睡眠模式下开启 SPI3 时钟 |
| 12 | SPI0SPEN | 在睡眠模式下 SPI0 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 SPI0 时钟<br>1：在睡眠模式下开启 SPI0 时钟 |
| 11 | SDIOSPEN | 在睡眠模式下 SDIO 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 SDIO 时钟<br>1：在睡眠模式下开启 SDIO 时钟 |
| 10 | ADC2SPEN | 在睡眠模式下 ADC2 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 ADC2 时钟<br>1：在睡眠模式下开启 ADC2 时钟 |
| 9 | ADC1SPEN | 在睡眠模式下 ADC1 时钟使能<br>由软件置位或复位 |

| | | |
|---|---|---|
| | | 0：在睡眠模式下关闭 ADC1 时钟<br>1：在睡眠模式下开启 ADC1 时钟 |
| 8 | ADC0SPEN | 在睡眠模式下 ADC0 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 ADC0 时钟<br>1：在睡眠模式下开启 ADC0 时钟 |
| 7:6 | 保留 | 必须保持复位值。 |
| 5 | USART5SPEN | 在睡眠模式下 USART5 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 USART5 时钟<br>1：在睡眠模式下开启 USART5 时钟 |
| 4 | USART0SPEN | 在睡眠模式下 USART0 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 USART0 时钟<br>1：在睡眠模式下开启 USART0 时钟 |
| 3:2 | 保留 | 必须保持复位值 |
| 1 | TIMER7SPEN | 在睡眠模式下 TIMER7 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER7 时钟<br>1：在睡眠模式下开启 TIMER7 时钟 |
| 0 | TIMER0SPEN | 在睡眠模式下 TIMER0 时钟使能<br>由软件置位或复位<br>0：在睡眠模式下关闭 TIMER0 时钟<br>1：在睡眠模式下开启 TIMER0 时钟 |

## 4.3.20. 备份域控制寄存器(RCU_BDCTL)

地址偏移：0x70
复位值：0x0000 0000，只能由备份域复位进行复位

**注意：**备份域控制寄存器(RCU_BDCTL)的LXTALEN、LXTALBPS、RTCSRC和RTCEN位仅在备份域复位后才清0。只有在电源控制寄存器(PMU_CTL)中的BKPWEN位置1后才能对这些位进行改动。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RTCEN | 保留 | | | | | RTCSRC[1:0] | | 保留 | | | | LXTALDRI | LXTALBPS | LXTALSTB | LXTALEN |

rw rw rw rw r rw

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:17 | 保留 | 必须保持复位值。 |
| 16 | BKPRST | 备份域复位<br>由软件置位或复位<br>0：无作用<br>1：复位备份域 |
| 15 | RTCEN | RTC 时钟使能<br>由软件置位或复位<br>0：关闭 RTC 时钟<br>1：开启 RTC 时钟 |
| 14:10 | 保留 | 必须保持复位值。 |
| 9:8 | RTCSRC[1:0] | RTC 时钟源选择<br>由软件置位或清零来控制RTC的时钟源。一旦RTC的时钟源选择后，除了将备份域复位否则时钟源不能被改变。<br>00：没有时钟<br>01：选择 CK_LXTAL 时钟作为 RTC 的时钟源<br>10：选择 CK_IRC32K 时钟作为 RTC 的时钟源<br>11：选择 CK_HXTAL / RTCDIV 时钟作为 RTC 的时钟源，请参考 RCU_CFG0 寄存器的 RTCDIV 位域。 |
| 7:4 | 保留 | 必须保持复位值。 |
| 3 | LXTALDRI | LXTAL 驱动能力<br>由软件置位或复位。当备份域复位时将复位该值<br>0：低驱动能力(复位值)<br>1：高驱动能力<br>注意：LXTALDRI 位在旁路模式下无效 |
| 2 | LXTALBPS | LXTAL 旁路模式使能<br>由软件置位或复位<br>0：禁止 LXTAL 旁路模式<br>1：使能 LXTAL 旁路模式 |
| 1 | LXTALSTB | 低速晶体振荡器稳定标志位<br>硬件置'1'来指示LXTAL振荡器时钟是否稳定待用<br>0：LXTAL 未稳定<br>1：LXTAL 已稳定 |
| 0 | LXTALEN | LXTAL 时钟使能<br>由软件置位或复位<br>0：关闭 LXTAL 时钟<br>1：使能 LXTAL 时钟 |

### 4.3.21. 复位源/时钟寄存器(RCU_RSTSCK)

地址偏移：0x74

复位值：0x0E00 0000，所有复位标志位仅在电源复位时被清零，RSTFC/IRC32KEN在系统复位时被清零。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LPRSTF | WWDGTRSTF | FWDGTRSTF | SWRSTF | PORRSTF | EPRSTF | BORRSTF | RSTFC | 保留 | | | | | | | |
| r | r | r | r | r | r | r | rw | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | | | IRC32KSTB | IRC32KEN |
| | | | | | | | | | | | | | | r | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | LPRSTF | 低功耗复位标志位<br>深度睡眠/待机复位发生时由硬件置位<br>向 RSTFC 位写 1 来清除该位<br>0：无低功耗管理复位发生<br>1：发生低功耗管理复位 |
| 30 | WWDGTRSTF | 窗口看门狗定时器复位标志位<br>窗口看门狗定时器复位发生时由硬件置 1<br>向 RSTFC 位写 1 来清除该位<br>0：无窗口看门狗复位发生<br>1：发生窗口看门狗复位 |
| 29 | FWDGTRSTF | 独立看门狗定时器复位标志位<br>独立看门狗复位发生时由硬件置 1<br>向 RSTFC 位写 1 来清除该位<br>0：无独立看门狗定时器复位发生<br>1：发生独立看门狗定时器复位 |
| 28 | SWRSTF | 软件复位标志位<br>软件复位发生时由硬件置 1<br>向 RSTFC 位写 1 来清除该位<br>0：无软件复位发生<br>1：发生软件复位 |
| 27 | PORRSTF | 电源复位标志位<br>电源复位发生时由硬件置 1<br>向 RSTFC 位写 1 来清除该位<br>0：无电源复位发生<br>1：发生电源复位 |

| | | |
|---|---|---|
| 26 | EPRSTF | 外部引脚复位标志位<br>外部引脚复位发生时由硬件置 1<br>向 RSTFC 位写 1 来清除该位<br>0：无外部引脚复位发生<br>1：发生外部引脚复位 |
| 25 | BORRSTF | 欠压复位复位标志位<br>欠压复位复位发生时由硬件置 1<br>向 RSTFC 位写 1 来清除该位<br>0：无欠压复位复位发生<br>1：发生欠压复位复位 |
| 24 | RSTFC | 清除复位标志位<br>由软件置 1 来清除所有复位标志位<br>0：无作用<br>1：清除所有复位标志位 |
| 23:2 | 保留 | 必须保持复位值。 |
| 1 | IRC32KSTB | IRC32K 时钟稳定标志位<br>该位由硬件置 1 指示 IRC32K 输出时钟是否稳定待用<br>0：IRC32K 时钟未稳定<br>1：IRC32K 已稳定 |
| 0 | IRC32KEN | IRC32K 使能<br>由软件置位和复位<br>0：关闭 IRC32K 时钟<br>1：开启 IRC32K 时钟 |

### 4.3.22. PLL 时钟扩频控制寄存器(RCU_PLLSSCTL)

地址偏移：0x80
复位值：0x0000 0000

扩频调制仅适用于主PLL时钟。

仅当PLL被禁止时，RCU_PLLSSCTL寄存器才可写入。
该寄存器用于配置PLL扩频时钟生成，需按照如下公式：

$MODCNT = round(f_{PLLIN}/4/f_{mod})$

$MODSTEP = round(mdamp*PLLN*2^{14}/(MODCNT*100))$

$f_{PLLIN}$表示PLL输入时钟频率，$f_{mod}$表示扩频调制频率，mdamp表示扩频调制振幅(按百分比表示)，PLLN 表示PLL时钟频率倍频因子。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SSCGON | SS_TYPE | 保留 | MODSTEP[14:3] | | | | | | | | | | | | |
| rw | rw | | rw | | | | | | | | | | | | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | SSCGON | PLL 扩频调制使能<br>0：禁止扩频调制<br>1：使能扩频调制 |
| 30 | SS_TYPE | PLL 扩频调制类型选择<br>0：选择中心扩频<br>1：选择向下扩频 |
| 29:28 | 保留 | 必须保持复位值。 |
| 27:13 | MODSTEP | 这些位配置 PLL 扩频调制曲线振幅和频率。必须满足如下条件：MODSTEP*MODCNT<=$2^{15}$-1 |
| 12:0 | MODCNT | 这些位配置 PLL 扩频调制曲线振幅和频率。必须满足如下条件：MODSTEP*MODCNT<=$2^{15}$-1 |

### 4.3.23. PLLI2S 寄存器(RCU_PLLI2S)

地址偏移：0x84
复位值：0x2400 3000

配置PLLI2S时钟可参考下列公式：
CK_PLLI2SVCOSRC = CK_PLLSRC / PLLPSC
CK_PLLI2SVCO = CK_PLLI2SVCOSRC × PLLI2SN
CK_PLLI2SR = CK_PLLI2SVCO / PLLI2SR

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | PLLI2SR[2:0] | | | 保留 | | | | | | | | | | | |
| | rw | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | PLLI2SN[8:0] | | | | | | | | | 保留 | | | | | |
| | rw | | | | | | | | | rw | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 30 | 保留 | 必须保持复位值。 |
| 30:28 | PLLI2SR[2:0] | PLLI2S VCO 时钟的分频因子用于 PLLI2SR 时钟输出频率<br>当 PLLI2S 时钟被关闭时由软件置位或复位。这些位用于通过 PLLI2S VCO 时钟(CK_ PLLI2SVCO)分频生成 PLLI2SR 输出时钟(CK_PLLI2SR)。CK_PLLI2SR 用于生成 I2S 时钟(<=240MHz)。RCU_PLLI2S 寄存器的 PLLI2SN 位域对 CK_PLLI2SVCO 时钟进行了描述。 |

| | | |
|---|---|---|
| | | 000：保留<br>001：保留<br>010：CK_PLLI2SR = CK_PLLI2SVCO / 2.<br>011：CK_PLLI2SR = CK_PLLI2SVCO / 3<br>100：CK_PLLI2SR = CK_PLLI2SVCO / 4<br>…<br>111：CK_PLLI2SR = CK_PLLI2SVCO / 7 |
| 27:15 | 保留 | 必须保持复位值。 |
| 14:6 | PLLI2SN[8:0] | PLLI2S VCO 时钟倍频因子<br>当 PLLI2S 被关闭时由软件置位或清零(仅支持全字/半字写操作)<br>这些位域用做将 PLLI2S VCO 源时钟(CK_PLLI2SVCOSRC)倍频生成 PLLI2S VCO 输出时钟(CK_PLLI2SVCO)。RCU_PLL 寄存器的 PLLPSC 位域对 CK_PLLI2SVCOSRC 时钟进行了描述。<br>注意：CK_ PLLI2SVCO 时钟频率范围必须在 100MHz 到 500MHz 之间<br>PLLI2SN 时钟的值必须为：50<=PLLI2SN<=500<br>000000000：保留<br>000000001：保留<br>…<br>000110001：保留<br>000110010：CK_PLLI2SVCO = CK_PLLI2SVCOSRC x 50.<br>000110011：CK_PLLI2SVCO = CK_PLLI2SVCOSRC x 51.<br>…<br>111110100：CK_PLLI2SVCO = CK_PLLI2SVCOSRC x 500.<br>111110101：保留<br>…<br>111111111：保留 |
| 5:0 | 保留 | 必须保持复位值。 |

## 4.3.24. PLLSAI 寄存器(RCU_PLLSAI)

地址偏移：0x88
复位值：0x2400 3000

配置PLLSAI时钟可参考下列公式：
CK_PLLSAIVCOSRC = CK_PLLSRC / PLLPSC
CK_PLLSAIVCO = CK_ PLLSAIVCOSRC × PLLSAIN
CK_PLLSAIP = CK_ PLLSAIVCO / PLLSAIP
CK_PLLSAIR = CK_ PLLSAIVCO / PLLSAIR

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | PLLSAIR[2:0] | | | 保留 | | | | | | | | | | PLLSAIP[1:0] | |
| | rw | | | | | | | | | | | | | rw | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30:28 | PLLSAIR[2:0] | PLLSAI VCO 时钟的分频因子用于 PLLSAIR 时钟输出频率<br>当 PLLSAI 时钟被关闭时由软件置位或复位。这些位用于通过 PLLSAI VCO 时钟(CK_PLLSAIVCO)分频生成 PLLSAIR 输出时钟(CK_PLLSAIR)。CK_PLLSAIR 用于生成 TLI 时钟(<=216MHz)。RCU_PLLSAI 寄存器的 PLLSAIN 位域对 CK_PLLSAIVCO 时钟进行了描述。<br>000：保留<br>001：保留<br>010：CK_PLLSAIR = CK_PLLSAIVCO / 2<br>011：CK_PLLSAIR = CK_PLLSAIVCO / 3<br>100：CK_PLLSAIR = CK_PLLSAIVCO / 4<br>…<br>111：CK_PLLSAIR = CK_PLLSAIVCO / 7 |
| 27:18 | 保留 | 必须保持复位值。 |
| 17:16 | PLLSAIP[1:0] | PLLSAI VCO 时钟的分频因子用于 PLLSAIP 时钟输出频率<br>当 PLLSAI 时钟被关闭时由软件置位或复位。这些位用于通过 PLLSAI VCO 时钟(CK_PLLSAIVCO)分频生成 PLLSAIP 输出时钟(CK_PLLSAIP)。CK_PLLSAIP 用于生成 UBSFS/USBHS(48MHz)， TRNG(48MHz)或 SDIO(<=48MHz)的时钟。RCU_PLLSAI 寄存器的 PLLSAIN 位域对 CK_PLLSAIVCO 时钟进行了描述。<br>00：CK_PLLSAIP = CK_PLLSAIVCO / 2<br>01：CK_PLLSAIP = CK_PLLSAIVCO / 4<br>10：CK_PLLSAIP = CK_PLLSAIVCO / 6<br>11：CK_PLLSAIP = CK_PLLSAIVCO / 8 |
| 15 | 保留 | 必须保持复位值。 |
| 14:6 | PLLSAIN[8:0] | PLLSAI VCO 时钟倍频因子<br>当 PLLSAI 被关闭时由软件置位或清零(仅支持全字/半字写操作)<br>这些位域用做将 PLLSAI VCO 源时钟(CK_PLLSAIVCOSRC)倍频生成 PLLSAI VCO 输出时钟(CK_PLLSAIVCO)。RCU_PLL 寄存器的 PLLPSC 位域对 CK_PLLVCOSRC 时钟进行了描述。<br>注意：CK_PLLSAIVCO 时钟频率范围必须在 100MHz 到 500MHz 之间<br>PLLI2SN 时钟的值必须为：50<=PLLSAIN<=500<br>000000000：保留<br>000000001：保留<br>…<br>000110001：保留<br>000110010：CK_PLLSAIVCO = CK_PLLSAIVCOSRC x 50<br>000110011：CK_PLLSAIVCO = CK_PLLSAIVCOSRC x 51 |

…
111110100：CK_PLLSAIVCO = CK_PLLSAIVCOSRC x 500
111110101：保留
…
111111111：保留

| | | |
|---|---|---|
| 5:0 | 保留 | 必须保持复位值。 |

## 4.3.25. 时钟配置寄存器 1(RCU_CFG1)

地址偏移：0x8C
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

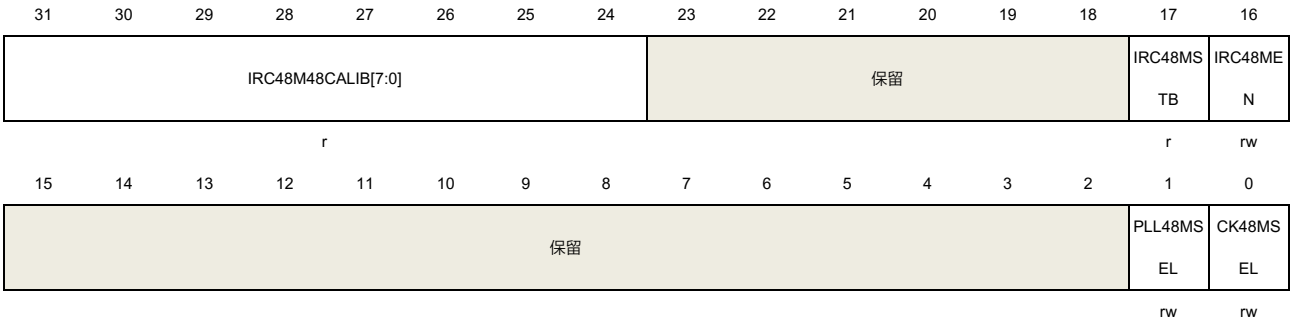
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:25 | 保留 | 必须保持复位值。 |
| 24 | TIMERSEL | TIMER 时钟源选择<br>由软件置位或复位 该位定义了所有定时器的时钟源选择<br>0：如果 RCU_CFG0 寄存器的 APB1PSC/APB2PSC 位域的值为 0b0xx(CK_APBx = CK_AHB) 或 0b100(CK_APBx = CK_AHB/2)，定时器时钟等于 CK_AHB(CK_TIMERx = CK_AHB)，否则定时器时钟等于 APB 时钟的两倍(在 APB1 域的定时器： CK_TIMERx = 2 x CK_APB1，在 APB2 域的定时器：CK_TIMERx = 2 x CK_APB2)。<br>1：如果 RCU_CFG0 寄存器的 APB1PSC/APB2PSC 位域的值为 0b0xx(CK_APBx = CK_AHB)，0b100(CK_APBx = CK_AHB/2)，或 0b101(CK_APBx = CK_AHB/4)，定时器时钟等于 CK_AHB(CK_TIMERx = CK_AHB)。否则定时器时钟等于 APB 时钟的四倍(在 APB1 域的定时器：CK_TIMERx = 4 x CK_APB1；在 APB2 域的定时器：CK_TIMERx = 4 x CK_APB2)。 |
| 23:18 | 保留 | 必须保持复位值。 |
| 17:16 | PLLSAIRDIV[1:0] | PLLSAIR 时钟的分频因子<br>当 PLLSAI 时钟被关闭时由软件置位或复位。该位用于生成 TLI 模块的时钟源。<br>00：CK_PLLSAIR / 2<br>01：CK_PLLSAIR / 4<br>10：CK_PLLSAIR / 8<br>11：CK_PLLSAIR / 16 |

| | | |
|---|---|---|
| 15:0 | 保留 | 必须保持复位值。 |

## 4.3.26. 附加时钟控制寄存器(RCU_ADDCTL)

地址偏移：0xC0
复位值：0xXX00 0000 x表示未定义

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



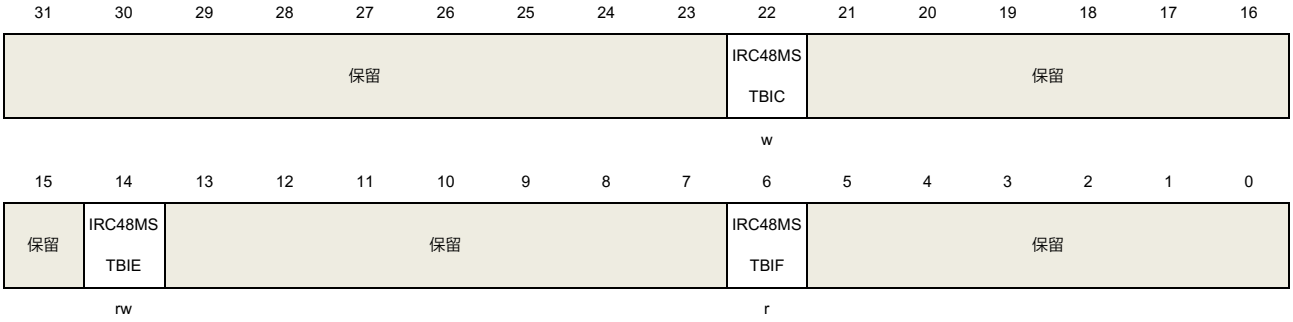

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | IRC48MCALIB[7:0] | 内部 48MHz RC 振荡器校准值寄存器<br>上电时自动加载这些位 |
| 23:18 | 保留 | 必须保持复位值。 |
| 17 | IRC48MSTB | 内部 48MHz RC 振荡器时钟稳定标志位<br>硬件置'1'来指示IRC48M振荡器时钟是否稳定待用<br>0：IRC48M未稳定<br>1：IRC48M 已稳定 |
| 16 | IRC48MEN | 内部 48MHz RC 振荡器使能<br>由软件置位和复位。当进入深度睡眠或待机模式后由硬件复位<br>0：关闭IRC48M时钟<br>1：打开 IRC48M 时钟 |
| 15:2 | 保留 | 必须保持复位值。 |
| 1 | PLL48MSEL | PLL48M时钟源选择<br>由软件置位和复位。该位用于选择CK_PLLQ时钟或CK_PLLSAIP时钟作为PLL48M的时钟源<br>0：选择CK_PLLQ时钟<br>1：选择 CK_PLLSAIP 时钟 |
| 0 | CK48MSEL | 48MHz时钟源选择<br>由软件置位和复位。该位用于选择IRC48M时钟或PLL48M时钟作为CK48M时钟源。CK48M时钟为TRNG/SDIO/USBFS/USBHS模块提供时钟。RCU_ADDCTL寄存器的PLL48MSEL位对PLL48M时钟进行了描述。<br>0：不选择IRC48M时钟(通过PLL48MSEL位选择使用CK_PLLQ时钟或CK_PLLSAIP时钟) |

1：选择 IRC48M 时钟

### 4.3.27. 附加时钟中断寄存器(RCU_ADDINT)

地址偏移：0xCC
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



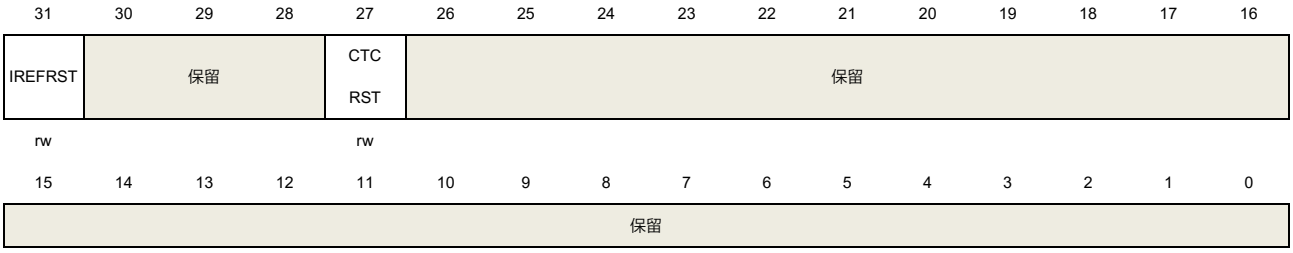

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:23 | 保留 | 必须保持复位值。 |
| 22 | IRC48MSTBIC | 内部 48 MHz RC 振荡器稳定中断清零<br>软件写 1 复位 IRC48MSTBIF 标志位<br>0：不复位 IRC48MSTBIF 标志位<br>1：复位 IRC48MSTBIF 标志位 |
| 21:15 | 保留 | 必须保持复位值。 |
| 14 | IRC48MSTBIE | 内部 48 MHz RC 振荡器稳定中断使能<br>由软件置位和复位来使能/禁止 IRC48M 时钟稳定中断<br>0：禁止IRC48M时钟稳定中断<br>1：使能 IRC48M 时钟稳定中断 |
| 13:7 | 保留 | 必须保持复位值。 |
| 6 | IRC48MSTBIF | IRC48M 时钟稳定中断标志位<br>当内部48 MHz RC振荡器时钟稳定且IRC48MSTBIE位被置1时由硬件置1<br>软件置位 IRC48MSTBIC 位时清除该位<br>0：无IRC48M时钟稳定中断产生<br>1：产生 IRC48M 时钟稳定中断 |
| 5:0 | 保留 | 必须保持复位值。 |

### 4.3.28. APB1 附加复位寄存器(RCU_ADDAPB1RST)

地址偏移：0xE0

复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



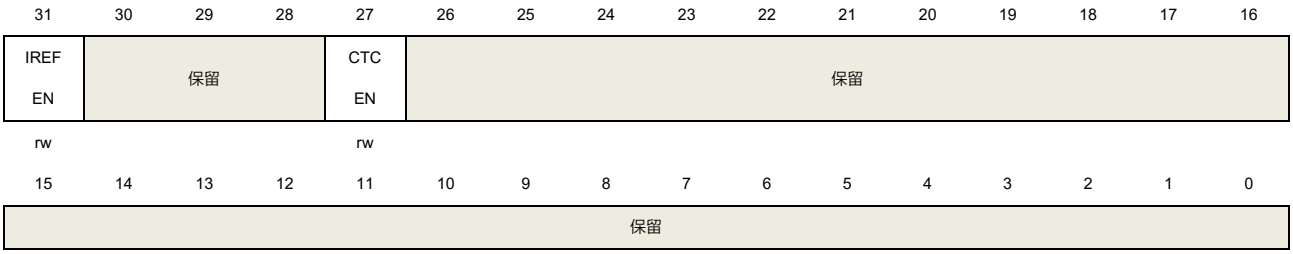

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | IREFRST | IREF 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 IREF |
| 30:28 | 保留 | 必须保持复位值。 |
| 27 | CTCRST | CTC 复位<br>由软件置位或复位<br>0：无作用<br>1：复位 CTC |
| 26:0 | 保留 | 必须保持复位值。 |

### 4.3.29. APB1 附加使能寄存器(RCU_ADDAPB1EN)

地址偏移：0xE4

复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



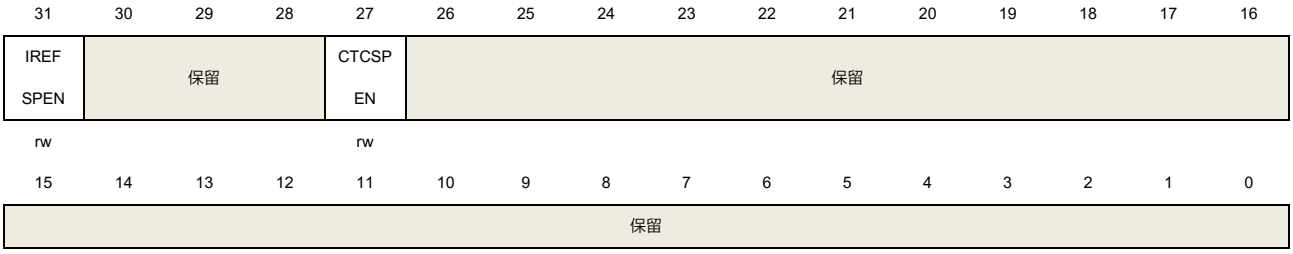

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | IREFEN | IREF 时钟使能<br>由软件置位或复位 |

| | | |
|---|---|---|
| | | 0：关闭 IREF 时钟<br>1：开启 IREF 时钟 |
| 30:28 | 保留 | 必须保持复位值。 |
| 27 | CTCEN | CTC 时钟使能<br>由软件置位或复位<br>0：关闭 CTC 时钟<br>1：开启 CTC 时钟 |
| 26:0 | 保留 | 必须保持复位值。 |

### 4.3.30. APB1 附加睡眠模式使能寄存器(RCU_ADDAPB1SPEN)

地址偏移：0xE8
复位值：0x8800 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | IREFSPEN | 睡眠模式下 IREF 时钟使能<br>由软件置位或复位<br>0：睡眠模式下关闭 IREF 时钟<br>1：睡眠模式下开启 IREF 时钟 |
| 30:28 | 保留 | 必须保持复位值。 |
| 27 | CTCSPEN | 睡眠模式下 CTC 时钟使能<br>由软件置位或复位<br>0：睡眠模式下关闭 CTC 时钟<br>1：睡眠模式下开启 CTC 时钟 |
| 26:0 | 保留 | 必须保持复位值。 |

### 4.3.31. 电源解锁寄存器(RCU_VKEY)

地址偏移：0x100
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | KEY[31:0] | RCU_DSV 寄存器解锁<br>这些位仅能被软件写，若读这些位，则全为 0。只有在向 RCC_VKEY 寄存器写 0x1A2B3C4D 后，RCU_DSV 寄存器才能被写。 |

## 4.3.32. 深度睡眠模式电压寄存器(RCU_DSV)

地址偏移：0x134
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:3 | 保留 | 必须保持复位值。 |
| 2:0 | DSLPVS[2:0] | 深度睡眠模式电压选择<br>由软件置位和清零这些位<br>000：在深度睡眠模式下内核电压为缺省值<br>001：在深度睡眠模式下内核电压为(缺省值-0.1)V(不建议客户使用)<br>010：在深度睡眠模式下内核电压为(缺省值-0.2)V(不建议客户使用)<br>011：在深度睡眠模式下内核电压为(缺省值-0.3)V(不建议客户使用)<br>100~111：保留 |

# 5. 时钟校准控制器(CTC)

## 5.1. 简介

时钟校准控制器(CTC)采用硬件的方式，自动校准内部48MHz RC晶振(IRC48M)。CTC模块基于外部高精度的参考信号源来校准IRC48M的时钟频率，通过自动的或手动的调整校准值，以得到一个精准的IRC48M时钟。

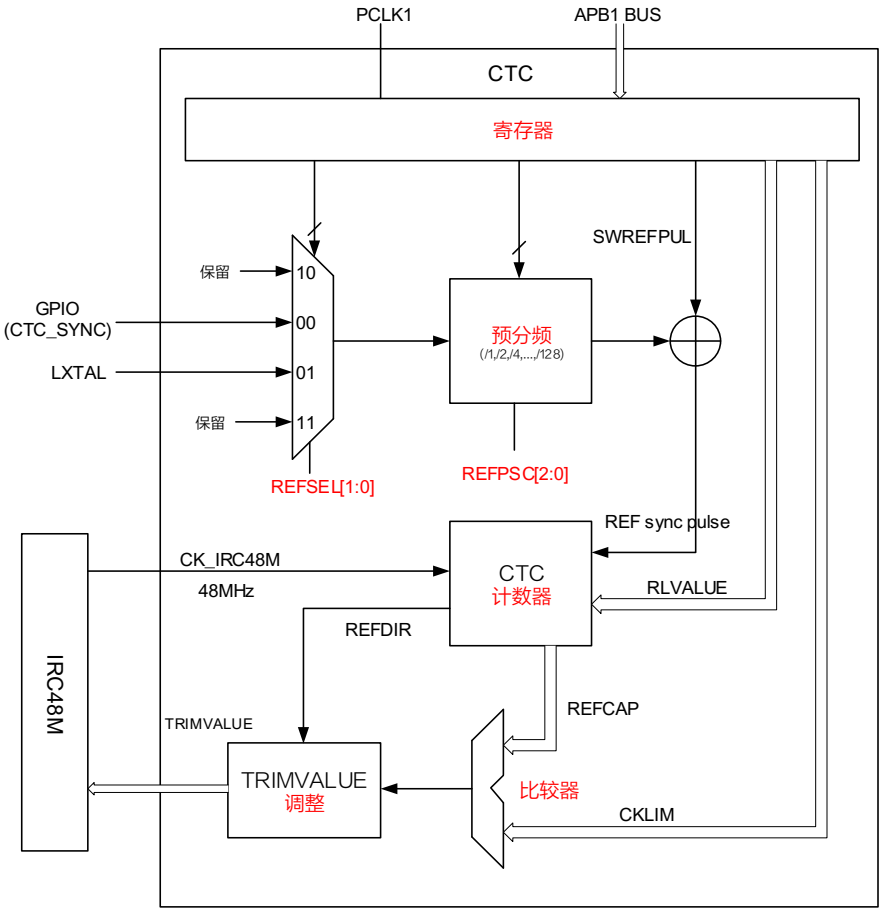
## 5.2. 主要特性

- 两个外部参考信号源：GPIO(CTC_SYNC)，LXTAL时钟；
- 提供软件参考同步脉冲；
- 硬件自动校准，无需软件操作；
- 具有参考信号源捕获和重载功能的16 bits校准计数器；
- 用于频率评估和自动校准的8 bits时钟校准基值；
- 标志位和中断，用于指示时钟校准的状态：校准成功状态(CKOKIF)，警告状态(CKWARNIF)和错误状态(ERRIF)。

## 5.3. 功能说明

CTC 模块的内部结构图如*图 5-1. CTC 简介*。

图5-1. CTC简介




### 5.3.1. REF 同步脉冲发生器

首先，通过设置CTC_CTL1寄存器中的REFSEL位来选择参考信号源：GPIO(CTC_SYNC)，LXTAL时钟输出。

然后，可以通过设置CTC_CTL1寄存器中的REFPOL位来配置参考信号源同步时的信号极性，通过设置CTC_CTL1寄存器中的REFPSC位来产生一个合适的同步时钟频率信号。

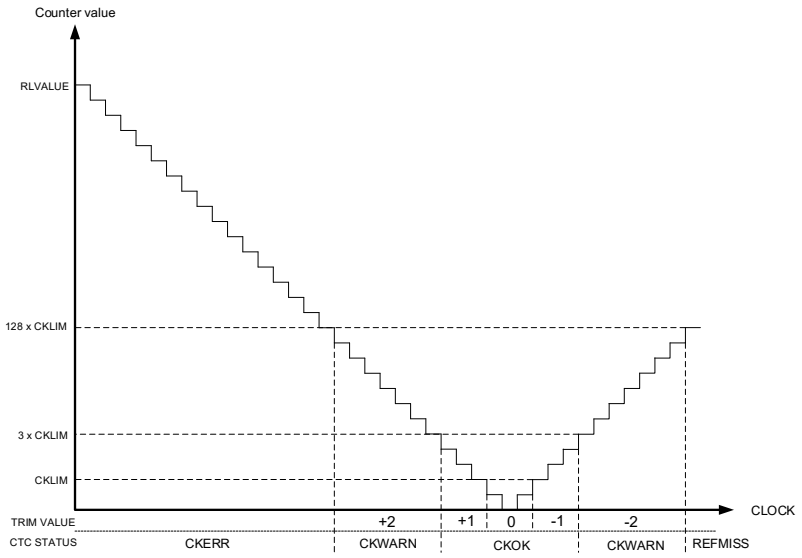
如果需要使用软件参考脉冲信号，则需要设置CTC_CTL0寄存器中的SWREFPUL位为1。软件参考脉冲信号与外部参考脉冲信号最后进行逻辑或操作。

### 5.3.2. CTC 校准计数器

CTC时钟校准计数器由CK_IRC48M提供时钟。在置位CTC_CTL0寄存器中的CNTEN位后，当检测到第一个REF同步脉冲信号，计数器开始从RLVALUE值(RLVALUE在CTC_CTL1寄存器中定义)开始向下计数。每次检测到REF同步脉冲信号时，计数器重载RLVALUE值，同时重新开始向下计数。如果始终检测不到REF同步脉冲信号，计数器会向下计数到零，然后再向上计数到128 x CKLIM(CKLIM在CTC_CTL1中定义)，最后停止，直到检测到下一个REF同步脉冲信号。一旦检测到REF同步脉冲信号，当前CTC校准计数器的计数值被捕获存入CTC_STAT寄存

器中的REFCAP位，同时，当前计数器的计数方向被存入CTC_STAT寄存器的REFDIR位。详细内容如*图5-2. CTC校准计数器*所示。

**图5-2. CTC校准计数器**




### 5.3.3. 频率评估和自动校准过程

当REF同步脉冲信号出现时，时钟频率评估功能开始执行。如果REF同步脉冲信号出现在计数器向下计数的过程中，说明当前时钟频率比期望时钟频率(频率为48M)慢，需要增大CTC_CTL0中的TRIMVALUE值(时钟校准值)。如果REF同步脉冲信号出现在计数器向上计数的过程中，说明当前时钟频率比期望时钟频率快，需要减小TRIMVALUE值。CTC_STAT中的CKOKIF位，CKWARNIF位，CKERR位和REFMISS位反映了频率评估的状态。

如果CTC_CTL0中的AUTOTRIM(硬件自动校准模式)位置1，硬件自动校准模式使能。在这个模式中，如果REF同步脉冲信号出现在计数器向下计数的过程中，说明当前时钟频率比期望时钟频率慢，CTC_CTL0中的TRIMVALUE值会自动增大，来提高当前的时钟频率。反之，如果REF同步脉冲信号出现在计数器向上计数的过程中，说明当前时钟频率比期望时钟频率快，TRIMVALUE值会自动减小，从而减小当前的时钟频率。

- Counter < CKLIM时，检测到REF同步脉冲信号；
  - CTC_STAT中的CKOKIF位(时钟校准成功标志位)被置位，同时，如果CTC_CTL0中的CKOKIE位(时钟校准完成中断使能位)置1，将会产生一个中断。
  - 如果CTC_CTL0中的AUTOTRIM置1，CTC_CTL0中的TRIMVALUE值不变。
- CKLIM <= Counter < 3 x CKLIM时，检测到REF同步脉冲信号；
  - CTC_STAT中的CKOKIF位被置位，同时，如果CTC_CTL0中的CKOKIE位置1，将会产生一个中断。
  - 如果CTC_CTL0中的AUTOTRIM位置1，在计数器向下计数过程中，CTC_CTL0中的TRIMVALUE值将加1，而在向上计数过程中将减1。
- 3 x CKLIM <= Counter < 128 x CKLIM时，检测到REF同步脉冲信号；
  - CTC_STAT中的CKWARNIF位(时钟校准警告中断位)被置位，同时，如果CTC_CTL0

中的CKWARNIE位(时钟校准警告中断使能位)置1，将会产生一个中断。

- 如果CTC_CTL0中的AUTOTRIM位置1，在计数器向下计数过程中，CTC_CTL0中的TRIMVALUE值将加2，而在向上计数过程中将减2。

■ Counter >= 128 x CKLIM，计数器在向下计数过程中，检测到REF同步脉冲信号；

- CTC_STAT中的CKERR位(时钟校准错误位)被置位，同时，如果CTC_CTL0中的ERRIE位(错误中断使能位)置1，将会产生一个中断。
- CTC_CTL0中的TRIMVALUE值不变。

■ Counter = 128 x CKLIM，计数器在向上计数过程中 ；

- CTC_STAT中的REFMISS位(REF同步脉冲丢失位)被置位，同时，如果CTC_CTL0中的ERRIE位置1，将会产生一个中断。
- CTC_CTL0中的TRIMVALUE值不变。

如果CTC_CTL0中的TRIMVALUE的校准值大于63，将会发生上溢事件，同时，若TRIMVALUE的校准值小于0，将会发生下溢事件。TRIMVALUE的取值范围为0~63(上溢事件发生时，TRIMVALUE值为63；下溢事件发生时，TRIMVALUE值为0)。然后，CTC_STAT中的TRIMERR位(校准值错误位)将会被置位，如果CTC_CTL0中的ERRIE位置1，将会产生一个中断。

### 5.3.4. 软件编程指南

CTC_CTL1中RLVALUE位和CKLIM位是时钟频率评估和硬件自动校准的关键。它们的数值由期望时钟的频率(IRC48M：48 MHz)和REF同步脉冲信号的频率计算得到。理想状态是REF同步脉冲信号在CTC计数器计数到零时出现，所以RLVALUE的值为：

RLVALUE=(F_clock/ F_REF )-1 (式5-1)

CKLIM的值由用户根据时钟的精度来设置，一般建议设为步长的一半，所以CKLIM的值为：

$CKLIM=(F_{clock}/ F_{REF}) \times 0.12\% /2$ (式5-2)

典型的步长值是0.12%，$F_{clock}$是期望时钟的频率(IRC48M)，$F_{REF}$是REF同步脉冲信号的频率。

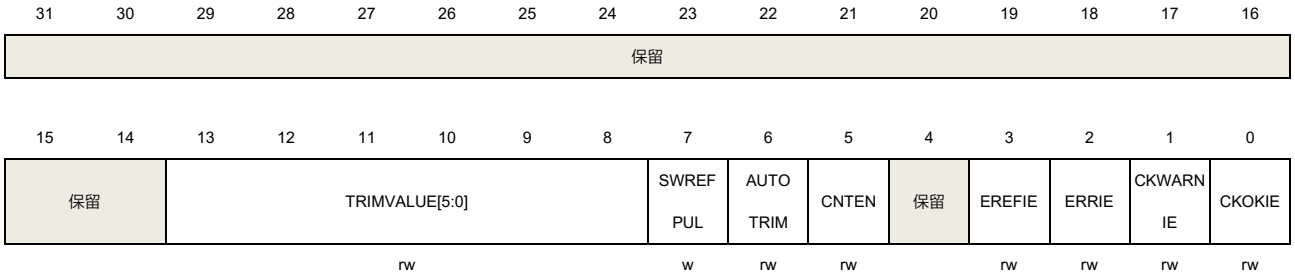
## 5.4. CTC 寄存器

CTC基地址：0x4000 6C00

### 5.4.1. 控制寄存器 0(CTC_CTL0)

地址偏移：0x00
复位值：0x0000 2000

该寄存器只能按字(32 位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值。 |
| 13:8 | TRIMVALUE[5:0] | IRC48M 校准值<br>当 CTC_CTL0 中的 AUTOTRIM 值为 0 时，该位由软件置位和清除，该模式用于软件校准过程。<br>当 CTC_CTL0 中的 AUTOTRIM 值为 1 时，该位只读，由硬件自动修改，该模式用于硬件校准过程。<br>TRIMVALUE 的中间值是 32，当 TRIMVALUE 值加 1 时，IRC48M 时钟频率增加大约 57KHz。当 TRIMVALUE 值减 1 时，IRC48M 时钟频率的减少大约 57KHz。 |
| 7 | SWREFPUL | 软件生成同步参考信号脉冲<br>该位由软件置位，并为 CTC 计数器提供一个同步参考脉冲信号。该位由硬件自动清除，读操作时返回 0。<br>0：没有影响<br>1：软件产生一个同步参考脉冲信号 |
| 6 | AUTOTRIM | 硬件自动校准模式<br>该位由软件置位或清除。当该位置 1 时，硬件自动校准模式使能，通过硬件不断的自动修改 CTC_CTL0 中的 TRIMVALUE 值，直到 IRC48M 的时钟频率达到 48MHz。<br>0：禁止硬件自动校准模式<br>1：使能硬件自动校准模式 |
| 5 | CNTEN | CTC 计数器使能<br>该位由软件置位或清除，用于使能或禁止 CTC 计数器。当该位置 1 时，不能修改 CTC_CTL1 的值。<br>0：禁止 CTC 计数器 |

| | | |
|---|---|---|
| | | 1：使能 CTC 计数器 |
| 4 | 保留 | 必须保持复位值。 |
| 3 | EREFIE | 期望参考信号中断使能<br>0：禁止期望参考信号产生中断<br>1：使能期望参考信号产生中断 |
| 2 | ERRIE | 错误中断使能<br>0：禁止错误中断<br>1：使能错误中断 |
| 1 | CKWARNIE | 时钟校准警告中断使能<br>0：禁止时钟校准警告中断<br>1：使能时钟校准警告中断 |
| 0 | CKOKIE | 时钟校准完成中断使能<br>0：禁止时钟校准完成中断<br>1：使能时钟校准完成中断 |

### 5.4.2. 控制寄存器 1(CTC_CTL1)

地址偏移：0x04
复位值：0x2022 BB7F

该寄存器只能按字(32位)访问。
**注意：**当CNTEN为1时，不能修改该寄存器的值。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| REFPOL | 保留 | REFSEL[1:0] | | 保留 | REFPSC[2:0] | | | CKLIM[7:0] | | | | | | | |
| rw | rw | rw | | | rw | | | rw | | | | | | | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | REFPOL | 参考信号源极性<br>该位由软件置位或清除，用于选择参考信号源的同步极性<br>0：选择上升沿<br>1：选择下降沿 |
| 30 | 保留 | 必须保持复位值。 |
| 29:28 | REFSEL[1:0] | 参考信号源选择<br>该位由软件置位或清除，用于选择参考信号源<br>00：选择 GPIO(CTC_SYNC)输入信号<br>01：选择 LXTAL 时钟<br>10：保留 |

| | | |
|---|---|---|
| | | 11：保留 |
| 27 | 保留 | 必须保持复位值。 |
| 26:24 | REFPSC[2:0] | 参考信号源预分频<br>该位由软件置位或清除<br>000：参考信号不分频<br>001：参考信号 2 分频<br>010：参考信号 4 分频<br>011：参考信号 8 分频<br>100：参考信号 16 分频<br>101：参考信号 32 分频<br>110：参考信号 64 分频<br>111：参考信号 128 分频 |
| 23:16 | CKLIM[7:0] | 时钟校准时基限值<br>该位由软件置位或清除，用于定义时钟校准时基限值。该位用于频率评估和自动校准过程，详细情况请参考*频率评估和自动校准过程*。 |
| 15:0 | RLVALUE[15:0] | CTC 计数器重载值<br>该位由软件置位或清除，用于定义 CTC 计数器的重载值，当检测到一个同步参考脉冲时，该值将重载到 CTC 校准计数器中。 |

### 5.4.3. 状态寄存器(CTC_STAT)

地址偏移：0x08
复位值：0x0000 0000

该寄存器只能按字(32 bit)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| REFDIR | 保留 | | | | TRIM ERR | REF MISS | CKERR | 保留 | | | | EREFIF | ERRIF | CKWARN IF | CKOK IF |
| r | | | | | r | r | r | | | | | r | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | REFCAP[15:0] | CTC 计数器捕获值<br>当检测到一个同步参考脉冲信号时，CTC 校准计数器中的计数值被存入到 REFCAP 位中。 |
| 15 | REFDIR | CTC 校准时钟计数方向<br>当检测到一个同步参考脉冲信号时，CTC 校准计数器的计数方向被存入 REFDIR 位中。<br>0：向上计数 |

| | | |
|---|---|---|
| | | 1：向下计数 |
| 14:11 | 保留 | 必须保持复位值。 |
| 10 | TRIMERR | 校准值错误位<br>当 CTC_CTL0 中的 TRIMVALUE 值发生上溢或下溢时，该位由硬件置位。若 CTC_CTL0 中的 ERRIE 位置 1，则会产生一个中断。通过写 1 到 CTC_INTC 中的 ERRIC 位，可以将 TRIMERR 位清零。<br>0：无校准值错误发生<br>1：发生校准值错误 |
| 9 | REFMISS | 同步参考脉冲信号丢失<br>当同步参考脉冲信号丢失时，该位由硬件置位。当 CTC 校准计数器在增计数的过程中计数到 128 x CKLIM 都没有检测到同步参考脉冲信号时，REFMISS 位置位。说明当前时钟太快,无法校准到期望频率值,或者有其他错误产生。通过写 1 到 CTC_INTC 中的 ERRIC 位，可以将 REFMISS 位清零。<br>0：无同步参考脉冲信号丢失<br>1：同步参考脉冲信号丢失 |
| 8 | CKERR | 时钟校准错误位<br>当时钟校准错误产生时，该位由硬件置位。当 CTC 校准计数器计数值在减计数的过程中大于或等于 128 x CKLIM，并检测到同步参考脉冲信号时，CKERR 置位，说明当前时钟太慢，无法校准到期望频率值。当 CTC_CTL0 中的 ERRIE 置 1 时，产生一个中断。通过写 1 到 CTC_INTC 中的 ERRIC 位，可以将 CKERR 位清零。<br>0：无时钟校准错误发生<br>1：发生时钟校准错误 |
| 7:4 | 保留 | 必须保持复位值。 |
| 3 | EREFIF | 期望参考中断标志位<br>当 CTC 校准时钟计数器计数到 0 时，该位由硬件置位。当 CTC_CTL0 中的 EREFIE 置 1 时，产生一个中断。通过写 1 到 CTC_INTC 中的 EREFIC 位，可以将 EREFIF 位清零。<br>0：无期望参考信号产生<br>1：期望参考信号产生 |
| 2 | ERRIF | 错误中断标志位<br>当发生一个错误时，该位由硬件置位。只要有 TRIMERR，REFMISS 或者 CKERR 错误发生时，该位置位。当 CTC_CTL0 中的 ERRIE 置位时，产生一个中断。通过写 1 到 CTC_INTC 中的 ERRIC 位，可以将 ERRIF 位清零。<br>0：无错误发生<br>1：发生错误 |
| 1 | CKWARNIF | 时钟校准警告中断标志位<br>当时钟校准警告产生时，该位由硬件置位。当 CTC 校准计数器计数值大于或等于 3 x CKLIM 且小于 128 x CKLIM，并检测到同步参考脉冲信号时，CKWARNIF 置位。这说明当前时钟频率太慢或者太快，但可以通过校准达到期望频率值。当时钟校准警告产生时，TRIMVALUE 值加 2 或者减 2。当 CTC_CTL0 中的 CKWARNIE 置 1 时， |

| | | |
|---|---|---|
| | | 产生一个中断。通过写 1 到 CTC_INTC 中的 CKWARNIC 位，可以将 CKWARNIF 位清零。<br>0：无时钟校准警告发生<br>1：有时钟校准警告发生 |
| 0 | CKOKIF | 时钟校准成功中断标志位<br>当时钟校准成功时，该位由硬件置位。若在 CTC 校准计数器计数值小于 3 x CKLIM 时，检测当同步参考脉冲信号，CKOKIF 置位。说明当前时钟频率正常，可以使用，不需要通过 TRIMVALUE 值进行时钟校准。当 CTC_CTL0 中的 CKOKIE 置 1 时，产生一个中断。通过写 1 到 CTC_INTC 中的 CKOKIC 位，可以将 CKOKIF 位清零。<br>0：时钟校准未成功<br>1：时钟校准成功 |

### 5.4.4. 中断清除寄存器(CTC_INTC)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32 bit)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | EREFIC | ERRIC | CKWARNIC | CKOKIC |
| | | | | | | | | | | | | w | w | w | w |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:4 | 保留 | 必须保持复位值。 |
| 3 | EREFIC | EREFIF 中断清除位<br>该位只能由软件写，读操作返回 0。写 1 可以清除 CTC_STAT 中的 EREFIF 位，写 0 没影响。 |
| 2 | ERRIC | ERRIF 中断清除位<br>该位只能由软件写，读操作返回 0。写 1 可以清除 CTC_STAT 中的 ERRIF 位，TRIMERR 位，REFMISS 位和 CKERR 位，写 0 没影响。 |
| 1 | CKWARNIC | CKWARNIF 中断清除位<br>该位只能由软件写，读操作返回 0。写 1 可以清除 CTC_STAT 中的 CKWARNIF 位，写 0 没影响。 |
| 0 | CKOKIC | CKOKIF 中断清除位<br>该位只能由软件写，读操作返回 0。写 1 可以清除 CTC_STAT 中的 CKOKIF 位，写 0 没影响。 |

# 6. 中断/事件控制器(EXTI)

## 6.1. 简介

Cortex®-M4 集成了嵌套式矢量型中断控制器(Nested Vectored Interrupt Controller (NVIC))来实现高效的异常和中断处理。NVIC 实现了低延迟的异常和中断处理，以及电源管理控制。它和内核是紧密耦合的。更多关于 NVIC 的说明请参考《 Cortex®-M4 技术参考手册 》。

EXTI(中断/事件控制器)包括 23 个相互独立的边沿检测电路并且能够向处理器内核产生中断请求或唤醒事件。EXTI 有三种触发类型：上升沿触发、下降沿触发和任意沿触发。EXTI 中的每一个边沿检测电路都可以独立配置和屏蔽。

## 6.2. 主要特征

- Cortex®-M4系统异常；
- 91种可屏蔽的外设中断；
- 4位中断优先级配置位，可提供16个中断优先等级；
- 高效的中断处理；
- 支持异常抢占和咬尾中断；
- 将系统从省电模式唤醒；
- EXTI中有23个相互独立的边沿检测电路；
- 3种触发类型：上升沿触发，下降沿触发和任意沿触发；
- 软件中断或事件触发；
- 可配置的触发源。

## 6.3. 中断功能描述

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

取向量是和当前工作态压栈并行进行的，从而提高了中断入口效率。处理器支持咬尾中断，可实现背靠背中断，大幅削减了反复切换工作态所带来的开销。*表 6-1. Cotrex-M4 中的 NVIC 异常类型*和*表 6-2. 中断向量表*列出了所有的异常类型。

**表 6-1. Cotrex-M4 中的 NVIC 异常类型**

| 异常类型 | 向量编号 | 优先级(a) | 向量地址 | 描述 |
|---|---|---|---|---|
| - | 0 | - | 0x0000_0000 | 保留 |
| 复位 | 1 | -3 | 0x0000_0004 | 复位 |
| NMI | 2 | -2 | 0x0000_0008 | 不可屏蔽中断 |
| 硬件故障 | 3 | -1 | 0x0000_000C | 各种硬件级别的故障 |

| 异常类型 | 向量编号 | 优先级(a) | 向量地址 | 描述 |
|---|---|---|---|---|
| 存储器管理 | 4 | 可编程设置 | 0x0000_0010 | 存储器管理 |
| 总线故障 | 5 | 可编程设置 | 0x0000_0014 | 预取指故障，存储器访问故障 |
| 用法故障 | 6 | 可编程设置 | 0x0000_0018 | 未定义的指令或非法状态 |
| - | 7-10 | - | 0x0000_001C - 0x0000_002B | 保留 |
| SVCall 服务调用 | 11 | 可编程设置 | 0x0000_002C | 通过 SWI 指令实现系统服务调用 |
| 调试监控 | 12 | 可编程设置 | 0x0000_0030 | 调试监视器 |
| - | 13 | - | 0x0000_0034 | 保留 |
| PendSV 挂起服务 | 14 | 可编程设置 | 0x0000_0038 | 可挂起的系统服务请求 |
| 系统节拍 | 15 | 可编程设置 | 0x0000_003C | 系统节拍定时器 |

**表 6-2. 中断向量表**

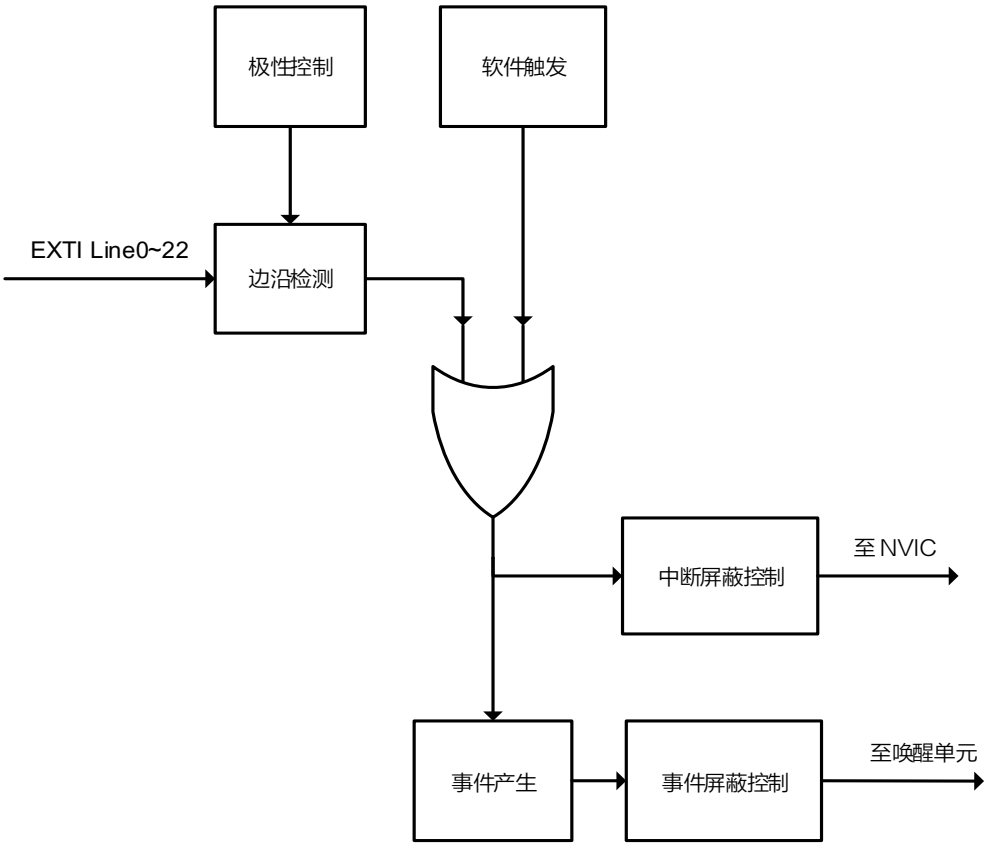
| 中断编号 | 向量编号 | 外设中断描述 | 向量地址 |
|---|---|---|---|
| IRQ 0 | 16 | 窗口看门狗中断 | 0x0000_0040 |
| IRQ 1 | 17 | 连接到 EXTI 线的 LVD 中断 | 0x0000_0044 |
| IRQ 2 | 18 | 连接到 EXTI 线的 RTC 侵入和时间戳中断 | 0x0000_0048 |
| IRQ 3 | 19 | 连接到 EXTI 线的 RTC 唤醒中断 | 0x0000_004C |
| IRQ 4 | 20 | FMC 全局中断 | 0x0000_0050 |
| IRQ 5 | 21 | RCU 和 CTC 中断 | 0x0000_0054 |
| IRQ 6 | 22 | EXTI 线 0 中断 | 0x0000_0058 |
| IRQ 7 | 23 | EXTI 线 1 中断 | 0x0000_005C |
| IRQ 8 | 24 | EXTI 线 2 中断 | 0x0000_0060 |
| IRQ 9 | 25 | EXTI 线 3 中断 | 0x0000_0064 |
| IRQ 10 | 26 | EXTI 线 4 中断 | 0x0000_0068 |
| IRQ 11 | 27 | DMA0 通道 0 全局中断 | 0x0000_006C |
| IRQ 12 | 28 | DMA0 通道 1 全局中断 | 0x0000_0070 |
| IRQ 13 | 29 | DMA0 通道 2 全局中断 | 0x0000_0074 |
| IRQ 14 | 30 | DMA0 通道 3 全局中断 | 0x0000_0078 |
| IRQ 15 | 31 | DMA0 通道 4 全局中断 | 0x0000_007C |
| IRQ 16 | 32 | DMA0 通道 5 全局中断 | 0x0000_0080 |
| IRQ 17 | 33 | DMA0 通道 6 全局中断 | 0x0000_0084 |
| IRQ 18 | 34 | ADC 全局中断 | 0x0000_0088 |
| IRQ 19 | 35 | CAN0 TX 中断 | 0x0000_008C |
| IRQ 20 | 36 | CAN0 RX0 中断 | 0x0000_0090 |
| IRQ 21 | 37 | CAN0 RX1 中断 | 0x0000_0094 |
| IRQ 22 | 38 | CAN0 EWMC 中断 | 0x0000_0098 |
| IRQ 23 | 39 | EXTI 线[9:5]中断 | 0x0000_009C |
| IRQ 24 | 40 | TIMER0 中止中断和 TIMER8 全局中断 | 0x0000_00A0 |
| IRQ 25 | 41 | TIMER0 更新中断和 TIMER9 全局中断 | 0x0000_00A4 |

| 中断编号 | 向量编号 | 外设中断描述 | 向量地址 |
|---|---|---|---|
| IRQ 26 | 42 | TIMER0 触发与通道换相中断和 TIMER10 全局中断 | 0x0000_00A8 |
| IRQ 27 | 43 | TIMER0 捕获比较中断 | 0x0000_00AC |
| IRQ 28 | 44 | TIMER1 全局中断 | 0x0000_00B0 |
| IRQ 29 | 45 | TIMER2 全局中断 | 0x0000_00B4 |
| IRQ 30 | 46 | TIMER3 全局中断 | 0x0000_00B8 |
| IRQ 31 | 47 | I2C0 事件中断 | 0x0000_00BC |
| IRQ 32 | 48 | I2C0 错误中断 | 0x0000_00C0 |
| IRQ 33 | 49 | I2C1 事件中断 | 0x0000_00C4 |
| IRQ 34 | 50 | I2C1 错误中断 | 0x0000_00C8 |
| IRQ 35 | 51 | SPI0 全局中断 | 0x0000_00CC |
| IRQ 36 | 52 | SPI1 全局中断 | 0x0000_00D0 |
| IRQ 37 | 53 | USART0 全局中断 | 0x0000_00D4 |
| IRQ 38 | 54 | USART1 全局中断 | 0x0000_00D8 |
| IRQ 39 | 55 | USART2 全局中断 | 0x0000_00DC |
| IRQ 40 | 56 | EXTI 线[15:10]中断 | 0x0000_00E0 |
| IRQ 41 | 57 | 连接到 EXTI 线的 RTC 闹钟中断 | 0x0000_00E4 |
| IRQ 42 | 58 | 连接到 EXTI 线的 USBFS 唤醒中断 | 0x0000_00E8 |
| IRQ 43 | 59 | TIMER7 中止中断和 TIMER11 全局中断 | 0x0000_00EC |
| IRQ 44 | 60 | TIMER7 更新中断和 TIMER12 全局中断 | 0x0000_00F0 |
| IRQ 45 | 61 | TIMER7 触发与通道换相中断和 TIMER13 全局中断 | 0x0000_00F4 |
| IRQ 46 | 62 | TIMER7 捕获比较中断 | 0x0000_00F8 |
| IRQ 47 | 63 | DMA0 通道 7 全局中断 | 0x0000_00FC |
| IRQ 48 | 64 | EXMC 全局中断 | 0x0000_0100 |
| IRQ 49 | 65 | SDIO 全局中断 | 0x0000_0104 |
| IRQ 50 | 66 | TIMER4 全局中断 | 0x0000_0108 |
| IRQ 51 | 67 | SPI2 全局中断 | 0x0000_010C |
| IRQ 52 | 68 | UART3 全局中断 | 0x0000_0110 |
| IRQ 53 | 69 | UART4 全局中断 | 0x0000_0114 |
| IRQ 54 | 70 | TIMER5 全局中断<br>DAC OUT0，DAC OUT1 下溢错误中断 | 0x0000_0118 |
| IRQ 55 | 71 | TIMER6 全局中断 | 0x0000_011C |
| IRQ 56 | 72 | DMA1 通道 0 全局中断 | 0x0000_0120 |
| IRQ 57 | 73 | DMA1 通道 1 全局中断 | 0x0000_0124 |
| IRQ 58 | 74 | DMA1 通道 2 全局中断 | 0x0000_0128 |
| IRQ 59 | 75 | DMA1 通道 3 全局中断 | 0x0000_012C |
| IRQ 60 | 76 | DMA1 通道 4 全局中断 | 0x0000_0130 |
| IRQ 61 | 77 | 以太网全局中断 | 0x0000_0134 |
| IRQ 62 | 78 | 连接到 EXTI 线的以太网唤醒中断 | 0x0000_0138 |
| IRQ 63 | 79 | CAN1 TX 中断 | 0x0000_013C |
| IRQ 64 | 80 | CAN1 RX0 中断 | 0x0000_0140 |

| 中断编号 | 向量编号 | 外设中断描述 | 向量地址 |
|---|---|---|---|
| IRQ 65 | 81 | CAN1 RX1 中断 | 0x0000_0144 |
| IRQ 66 | 82 | CAN1 EWMC 中断 | 0x0000_0148 |
| IRQ 67 | 83 | USBFS 全局中断 | 0x0000_014C |
| IRQ 68 | 84 | DMA1 通道 5 全局中断 | 0x0000_0150 |
| IRQ 69 | 85 | DMA1 通道 6 全局中断 | 0x0000_0154 |
| IRQ 70 | 86 | DMA1 通道 7 全局中断 | 0x0000_0158 |
| IRQ 71 | 87 | USART5 全局中断 | 0x0000_015C |
| IRQ 72 | 88 | I2C2 事件中断 | 0x0000_0160 |
| IRQ 73 | 89 | I2C2 错误中断 | 0x0000_0164 |
| IRQ 74 | 90 | USBHS 端点 1 输出中断 | 0x0000_0168 |
| IRQ 75 | 91 | USBHS 端点 1 输入中断 | 0x0000_016C |
| IRQ 76 | 92 | 连接到 EXTI 线的 USBHS 唤醒中断 | 0x0000_0170 |
| IRQ 77 | 93 | USBHS 全局中断 | 0x0000_0174 |
| IRQ78 | 94 | DCI 全局中断 | 0x0000_0178 |
| IRQ79 | 95 | 保留 | 0x0000_017C |
| IRQ80 | 96 | TRNG 全局中断 | 0x0000_0180 |
| IRQ 81 | 97 | FPU 全局中断 | 0x0000_0184 |
| IRQ82 | 98 | UART6 全局中断 | 0x0000_0188 |
| IRQ83 | 99 | UART7 全局中断 | 0x0000_018C |
| IRQ84 | 100 | SPI3 全局中断 | 0x0000_0190 |
| IRQ85 | 101 | SPI4 全局中断 | 0x0000_0194 |
| IRQ86 | 102 | SPI5 全局中断 | 0x0000_0198 |
| IRQ87 | 103 | 保留 | 0x0000_019C |
| IRQ88 | 104 | TLI 全局中断 | 0x0000_01A0 |
| IRQ89 | 105 | TLI 错误中断 | 0x0000_01A4 |
| IRQ90 | 106 | IPA 全局中断 | 0x0000_01A8 |

## 6.4. 结构框图

**图 6-1. EXTI 的结构框图**




## 6.5. 外部中断及事件功能概述

EXTI包含23个相互独立的边沿检测电路并且可以向处理器产生中断请求或事件唤醒。EXTI提供3种触发类型：上升沿触发，下降沿触发和任意沿触发。EXTI中每个边沿检测电路都可以分别予以配置或屏蔽。

EXTI触发源包括来自I/O管脚的16根线以及来自内部模块的7根线，具体细节参考*表6-3. EXTI触发源*。通过配置SYSCFG_EXTISSx寄存器，所有的GPIO管脚都可以被选作EXTI的触发源，具体细节请参考*系统配置寄存器(SYSCFG)*。

除了中断，EXTI还可以向处理器提供事件信号。Cortex®-M4内核完全支持等待中断(WFI)，等待事件(WFE)和发送事件(SEV)指令。芯片内部有一个唤醒中断控制器(WIC)，用户可以让处理器和NVIC进入功耗极低的省电模式，由WIC来识别中断和事件以及判断优先级。当某些预期的事件发生时，例如一个特定的I/O管脚电平翻转或者RTC闹钟动作，EXTI能唤醒处理器及整个系统。

### 硬件触发

硬件触发被用来检测外部或内部信号的电压变化。软件需要按如下步骤配置来使用这项功能：

1. 根据应用需要配置 SYSCFG 模块中的 EXTI 触发源；
2. 配置 EXTI_RTEN 寄存器和 EXTI_FTEN 寄存器以使能相应引脚的上升沿或下降沿检测(软件应当同时配置引脚对应的 RTENx 和 FTENx 位以检测该引脚上升沿和下降沿的变化)；
3. 通过配置引脚对应的 EXTI_INTEN 或 EXTI_EVEN 位，使能中断或事件；
4. EXTI 开始检测被配置的引脚上的电平变化，当这些引脚上期望的变化被检测到时，使能的中断或事件将被触发。如果为中断触发，则对应的 PD 位将立刻被置 1；如果为事件触发，则对应的 PD 位不被置 1。软件需要响应该中断或事件并清除相应 PDx 位。

### 软件触发

按照如下步骤软件也可以触发 EXTI 中断或事件：

1. 配置对应的 EXTI_INTEN 或 EXTI_EVEN 位使能中断或事件；
2. 配置 EXTI_SWIEV 寄存器的对应 SWIEVx 位，使能的中断或事件将被立即触发。如果为中断触发，则对应的 PD 位将立刻被置 1；如果为事件触发，则对应的 PD 位不被置 1。软件需要响应该中断或事件并清除相应 PDx 位。

**表 6-3. EXTI 触发源**

| EXTI 线编号 | 触发源 |
|---|---|
| 0 | PA0 / PB0 / PC0 / PD0 / PE0 / PF0 / PG0 / PH0 / PI0 |
| 1 | PA1 / PB1 / PC1 / PD1 / PE1 / PF1 / PG1 / PH1 / PI1 |
| 2 | PA2 / PB2 / PC2 / PD2 / PE2 / PF2 / PG2 / PH2 / PI2 |
| 3 | PA3 / PB3 / PC3 / PD3 / PE3 / PF3 / PG3 / PH3 / PI3 |
| 4 | PA4 / PB4 / PC4 / PD4 / PE4 / PF4 / PG4 / PH4 / PI4 |
| 5 | PA5 / PB5 / PC5 / PD5 / PE5 / PF5 / PG5 / PH5 / PI5 |
| 6 | PA6 / PB6 / PC6 / PD6 / PE6 / PF6 / PG6 / PH6 / PI6 |
| 7 | PA7 / PB7 / PC7 / PD7 / PE7 / PF7 / PG7 / PH7 / PI7 |
| 8 | PA8 / PB8 / PC8 / PD8 / PE8 / PF8 / PG8 / PH8 / PI8 |
| 9 | PA9 / PB9 / PC9 / PD9 / PE9 / PF9 / PG9 / PH9 / PI9 |
| 10 | PA10 / PB10 / PC10 / PD10 / PE10 / PF10 / PG10 / PH10 / PI10 |
| 11 | PA11 / PB11 / PC11 / PD11 / PE11 / PF11 / PG11 / PH11 / PI11 |
| 12 | PA12 / PB12 / PC12 / PD12 / PE12 / PF12 / PG12 / PH12 |
| 13 | PA13 / PB13 / PC13 / PD13 / PE13 / PF13 / PG13 / PH13 |
| 14 | PA14 / PB14 / PC14 / PD14 / PE14 / PF14 / PG14 / PH14 |
| 15 | PA15 / PB15 / PC15 / PD15 / PE15 / PF15 / PG15 / PH15 |
| 16 | LVD |
| 17 | RTC 闹钟 |
| 18 | USBFS 唤醒 |
| 19 | Ethernet 唤醒 |

| EXTI 线编号 | 触发源 |
| --- | --- |
| 20 | USBHS 唤醒 |
| 21 | RTC 侵入和时间戳事件 |
| 22 | RTC 唤醒 |

## 6.6. EXTI 寄存器

EXTI 基地址：0x4001 3C00

### 6.6.1. 中断使能寄存器(EXTI_INTEN)

地址偏移：0x00
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | INTEN22 | INTEN21 | INTEN20 | INTEN19 | INTEN18 | INTEN17 | INTEN16 |
| | | | | | | | | | rw | rw | rw | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INTEN15 | INTEN14 | INTEN13 | INTEN12 | INTEN11 | INTEN10 | INTEN9 | INTEN8 | INTEN7 | INTEN6 | INTEN5 | INTEN4 | INTEN3 | INTEN2 | INTEN1 | INTEN0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:23 | 保留 | 必须保持复位值。 |
| 22:0 | INTENx | 中断使能位x(x = 0…22)<br>0：第x线中断被禁用<br>1：第x线中断被使能 |

### 6.6.2. 事件使能寄存器(EXTI_EVEN)

地址偏移：0x04
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | EVEN22 | EVEN21 | EVEN20 | EVEN19 | EVEN18 | EVEN17 | EVEN16 |
| | | | | | | | | | rw | rw | rw | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EVEN15 | EVEN14 | EVEN13 | EVEN12 | EVEN11 | EVEN10 | EVEN9 | EVEN8 | EVEN7 | EVEN6 | EVEN5 | EVEN4 | EVEN3 | EVEN2 | EVEN1 | EVEN0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:23 | 保留 | 必须保持复位值。 |
| 22:0 | EVENx | 事件使能位x(x = 0…22)<br>0：第x线事件被禁用<br>1：第x线事件被使能 |

### 6.6.3. 上升沿触发使能寄存器(EXTI_RTEN)

地址偏移：0x08
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | RTEN22 | RTEN21 | RTEN20 | RTEN19 | RTEN18 | RTEN17 | RTEN16 |
| | | | | | | | | | rw | rw | rw | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RTEN15 | RTEN14 | RTEN13 | RTEN12 | RTEN11 | RTEN10 | RTEN9 | RTEN8 | RTEN7 | RTEN6 | RTEN5 | RTEN4 | RTEN3 | RTEN2 | RTEN1 | RTEN0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:23 | 保留 | 必须保持复位值。 |
| 22:0 | RTENx | 上升沿触发使能x(x = 0…22)<br>0：第x线上升沿触发无效<br>1：第x线上升沿触发有效(中断/事件请求) |

### 6.6.4. 下降沿触发使能寄存器(EXTI_FTEN)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | FTEN22 | FTEN21 | FTEN20 | FTEN19 | FTEN18 | FTEN17 | FTEN16 |
| | | | | | | | | | rw | rw | rw | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FTEN15 | FTEN14 | FTEN13 | FTEN12 | FTEN11 | FTEN10 | FTEN9 | FTEN8 | FTEN7 | FTEN6 | FTEN5 | FTEN4 | FTEN3 | FTEN2 | FTEN1 | FTEN0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:23 | 保留 | 必须保持复位值。 |
| 22:0 | FTENx | 下降沿触发使能x(x = 0…22)<br>0：第x线下降沿触发无效<br>1：第x线下降沿触发有效(中断/事件请求) |

### 6.6.5. 软件中断事件寄存器(EXTI_SWIEV)

地址偏移：0x10
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | SWIEV22 | SWIEV21 | SWIEV20 | SWIEV19 | SWIEV18 | SWIEV17 | SWIEV16 |
| | | | | | | | | | rw | rw | rw | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SWIEV15 | SWIEV14 | SWIEV13 | SWIEV12 | SWIEV11 | SWIEV10 | SWIEV9 | SWIEV8 | SWIEV7 | SWIEV6 | SWIEV5 | SWIEV4 | SWIEV3 | SWIEV2 | SWIEV1 | SWIEV0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:23 | 保留 | 必须保持复位值。 |
| 22:0 | SWIEVx | 中断/事件软件触发x(x = 0…22)<br>0：禁用EXTI线x软件中断/事件请求<br>1：激活EXTI线x软件中断/事件请求 |

## 6.6.6. 挂起寄存器(EXTI_PD)

地址偏移：0x14

复位值：0xXXXX XXXX X表示未定义

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | PD22 | PD21 | PD20 | PD19 | PD18 | PD17 | PD16 |
| | | | | | | | | | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PD15 | PD14 | PD13 | PD12 | PD11 | PD10 | PD9 | PD8 | PD7 | PD6 | PD5 | PD4 | PD3 | PD2 | PD1 | PD0 |
| rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:23 | 保留 | 必须保持复位值。 |
| 22:0 | PDx | 中断挂起状态x(x = 0…22)<br>0：EXTI线x没有被触发<br>1：EXTI线x被触发<br>对这些位写1，可将其清0。 |

# 7. 通用和备用输入/输出接口(GPIO 和 AFIO)

## 7.1. 简介

最多可支持 140 个通用 I/O 引脚(GPIO)，分别为 PA0 ~ PA15，PB0 ~ PB15，PC0 ~ PC15，PD0 ~ PD15，PE0 ~ PE15，PF0 ~ PF15，PG0 ~ PG15，PH0 ~ PH15 和 PI0 ~ PI11，各片上设备用其来实现逻辑输入/输出功能。每个 GPIO 端口有相关的控制和配置寄存器以满足特定应用的需求。GPIO 引脚上的外部中断在中断/事件控制器(EXTI)中有相关的控制和配置寄存器。

GPIO 端口和其他的备用功能(AFs)共用引脚，在特定的封装下获得最大的灵活性。GPIO 引脚通过配置相关的寄存器可以用作备用功能引脚，备用功能输入/输出都可以。

每个 GPIO 引脚可以由软件配置为输出(推挽或开漏)、输入、外设备用功能或者模拟模式。每个 GPIO 引脚都可以配置为上拉、下拉或无上拉/下拉。除模拟模式外，所有的 GPIO 引脚都具备大电流驱动能力。

## 7.2. 主要特性

- 输入 / 输出方向控制;
- 施密特触发器输入功能使能控制;
- 每个引脚都具有弱上拉 / 下拉功能;
- 推挽 / 开漏输出使能控制;
- 置位 / 复位输出使能;
- 可编程触发沿的外部中断-使用EXTI配置寄存器
- 模拟输入 / 输出配置;
- 备用功能输入 / 输出配置;
- 端口锁定配置;
- 单周期输出翻转功能。

## 7.3. 功能描述

每个通用 I / O 端口都可以通过 32 位控制寄存器(GPIOx_CTL)配置为 GPIO 输入，GPIO 输出，AF 功能或模拟模式。当选择 AF 功能时，引脚 AF 输入 / 输出是通过 AF 功能输出使能来选择。当端口配置为输出(GPIO 输出或 AFIO 输出)时，可以通过 GPIO 输出模式寄存器(GPIOx_OMODE)配置为推挽或开漏模式。输出端口的最大速度可以通过 GPIO 输出速度寄存器(GPIOx_OSPD)配置。每个端口可以通过 GPIO 上 / 下拉寄存器(GPIOx_PUD)配置为浮空(无上拉或下拉)，上拉或下拉功能。

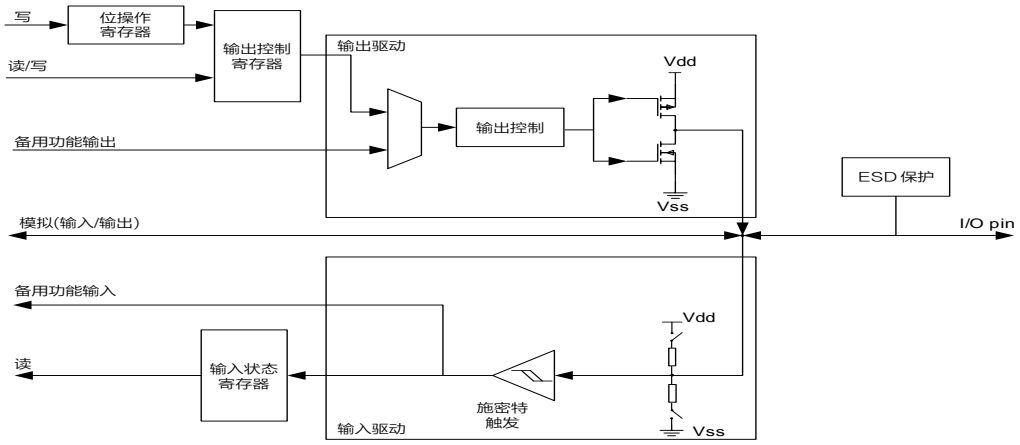
**表 7-1. GPIO 配置表**

| PAD TYPE | | | CTLy | OMy | PUDy |
|---|---|---|---|---|---|
| GPIO 输入 | X | 浮空 | 00 | X | 00 |
| | | 上拉 | | | 01 |

| PAD TYPE | | | CTLy | OMy | PUDy |
|---|---|---|---|---|---|
| | | 下拉 | | | 10 |
| GPIO 输出 | 推挽 | 浮空 | 01 | 0 | 00 |
| | | 上拉 | | | 01 |
| | | 下拉 | | | 10 |
| | 开漏 | 浮空 | | 1 | 00 |
| | | 上拉 | | | 01 |
| | | 下拉 | | | 10 |
| AFIO 输入 | X | 浮空 | 10 | X | 00 |
| | | 上拉 | | | 01 |
| | | 下拉 | | | 10 |
| AFIO 输出 | 推挽 | 浮空 | 10 | 0 | 00 |
| | | 上拉 | | | 01 |
| | | 下拉 | | | 10 |
| | 开漏 | 浮空 | | 1 | 00 |
| | | 上拉 | | | 01 |
| | | 下拉 | | | 10 |
| 模拟 | X | X | 11 | X | XX |

*图 7-1. GPIO 端口位的基本结构*为标准 I / O 端口位的基本结构图。

**图 7-1. GPIO 端口位的基本结构**




## 7.3.1. GPIO 引脚配置

在复位期间或复位之后，备用功能并未激活，所有 GPIO 端口都被配置成输入浮空模式，这种输入模式禁用上拉(PU) / 下拉(PD)电阻。但是复位后，串行线调试端口(JTAG / Serial-Wired Debug pins)为输入 PU / PD 模式：

PA15：JTDI 为上拉模式；
PA14：JTCK / SWCLK 为下拉模式；
PA13：JTMS / SWDIO 为上拉模式；
PB4：NJTRST 为上拉模式。

PB3：JTDO 为浮空模式。

GPIO 引脚可以配置为输入或输出模式，当 GPIO 引脚可配置为输入引脚时，所有的 GPIO 引脚内部都有一个可选择的弱上拉和弱下拉电阻。外部引脚上的数据在每个 AHB 时钟周期时都会装载到数据输入寄存器(GPIOx_ISTAT)。

当 GPIO 引脚配置为输出引脚，用户可以配置端口的输出速度和选择输出驱动模式：推挽或开漏模式。输出寄存器(GPIOx_OCTL)的值将会从相应 I / O 引脚上输出。

当对 GPIOx_OCTL 进行位操作时，不需要先读再写，用户可以通过写'1'到位操作寄存器(GPIOx_BOP，或用于清 0 的 GPIOx_BC，或用于翻转操作的 GPIOx_TG)修改一位或几位，该过程仅需要一个最小的 AHB 写访问周期，而其他位不受影响。

### 7.3.2. 外部中断/事件线

只有在输入模式下配置，端口才能使用外部中断/事件线。

### 7.3.3. 备用功能(AF)

当端口配置为 AFIO(设置 GPIOx_CTL 寄存器中的 CTLy 值为"0b10")时，该端口用作外设备用功能。通过配置 GPIO 备用功能选择寄存器(GPIOx_AFSELz(z=0,1))，每个端口可以配置 16 个备用功能。端口备用功能分配的详细介绍见芯片数据手册。

### 7.3.4. 附加功能

有些引脚具有附加功能，它们优先于标准 GPIO 寄存器中的配置。当用作 ADC 或 DAC 附加功能时，引脚必须配置成模拟模式。当引脚用作 RTC、WKUPx 和振荡器附加功能时，端口类型通过相关的 RTC、PMU 和 RCU 寄存器自动设置。当附加功能禁用时，这些端口可用作普通 GPIO。

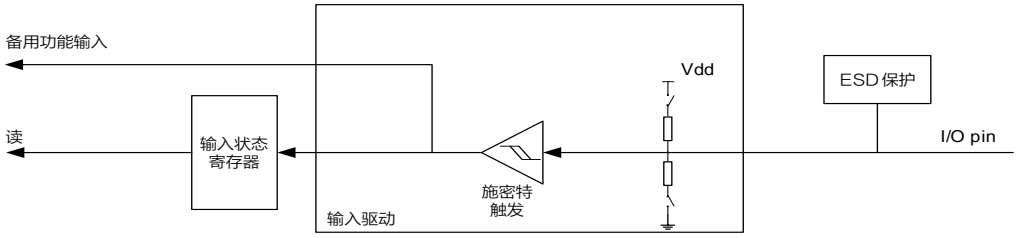
### 7.3.5. 输入配置

当 GPIO 引脚配置为输入时：

- 施密特触发输入使能；
- 可选择的弱上拉和下拉电阻；
- 当前I / O引脚上的数据在每个AHB时钟周期都会被采样并存入端口输入状态寄存器；
- 输出缓冲器禁用。

*图 7-2. 输入配置的基本结构*显示 I/O 引脚的输入配置。

**图 7-2. 输入配置的基本结构**



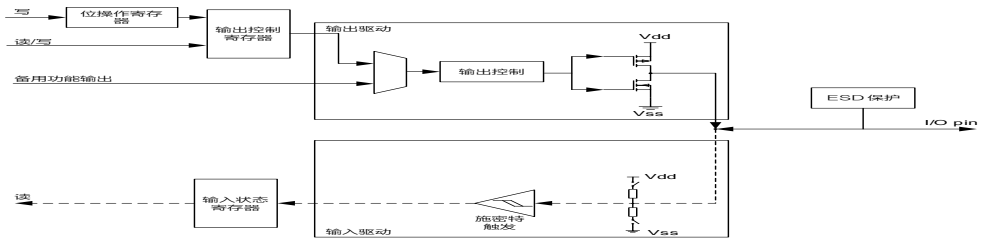

## 7.3.6. 输出配置

当 GPIO 配置为输出时：

- 施密特触发输入使能；
- 可选择的弱上拉和下拉电阻；
- 输出缓冲器使能；
- 开漏模式：输出控制寄存器设置为"0"时，相应引脚输出低电平；输出控制寄存器设置 为"1"，相应管脚处于高阻状态；
- 推挽模式：输出控制寄存器设置为"0"时，相应引脚输出低电平；输出控制寄存器设置为"1"，相应引脚输出高电平；
- 对端口输出控制寄存器进行读操作，将返回上次写入的值；
- 对端口输入状态寄存器进行读操作，将获得当前I/O口的状态。

*图 7-3. 输出配置的基本结构*是 I/O 端口的输出配置。

**图 7-3. 输出配置的基本结构**




## 7.3.7. 模拟配置

当 GPIO 引脚用于模拟模式时：

- 弱上拉和下拉电阻禁用；
- 输出缓冲器禁用；
- 施密特触发输入禁用；
- 端口输入状态寄存器相应位为"0"。

*图 7-4. 模拟配置的基本结构*是 I / O 端口的模拟模式配置。

**图 7-4. 模拟配置的基本结构**




### 7.3.8. 备用功能(AF)配置

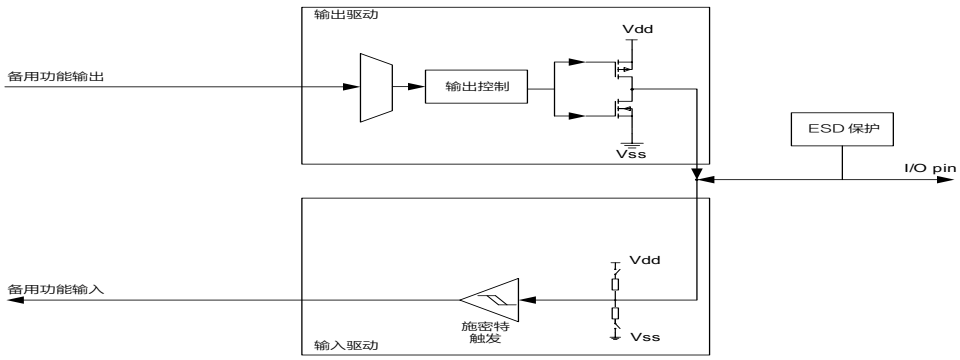
为了适应不同的器件封装，GPIO 端口支持软件配置将一些备用功能应用到其他引脚上。

当引脚配置为备用功能时：

- 使用开漏或推挽功能时，可使能输出缓冲器；
- 输出缓冲器由外设驱动；
- 施密特触发输入使能；
- 在输入配置时，可选择的弱上拉 / 下拉电阻；
- I / O引脚上的数据在每个AHB时钟周期采样并存入端口输入状态寄存器；
- 对端口输入状态寄存器进行读操作，将获得I / O口的状态；
- 对端口输出控制寄存器进行读操作，将返回上次写入的值。

*图 7-5. 备用功能配置的基本结构*是 I / O 端口备用功能配置图。

**图 7-5. 备用功能配置的基本结构**




### 7.3.9. GPIO 锁定功能

GPIO 的锁定机制可以保护 I / O 端口的配置。

被保护的寄存器有： GPIOx_CTL， GPIOx_OMODE， GPIOx_OSPD， GPIOx_PUD 和 GPIOx_AFSELz(z=0,1)。通过配置 32 位锁定寄存器(GPIOx_LOCK)可以锁定 I / O 端口的配置。通过特定的锁定序列配置 GPIOx_LOCK 中的 LKK 位和 LKy 位，相应的端口位被锁定，直到下一个复位前，相应端口位的配置都不能修改。建议在电源驱动模块的配置中使用锁定功能。

### 7.3.10. GPIO 单周期输出翻转功能

通过将 GPIOx_TG 寄存器中对应的位写 1，GPIO 可以在一个 AHB 时钟周期内翻转 I / O 的输出电平。输出信号的频率可以达到 AHB 时钟的一半。

## 7.4. GPIO 寄存器

GPIOA 基地址：0x4002 0000

GPIOB 基地址：0x4002 0400

GPIOC 基地址：0x4002 0800

GPIOD 基地址：0x4002 0C00

GPIOE 基地址：0x4002 1000

GPIOF 基地址：0x4002 1400

GPIOG 基地址：0x4002 1800

GPIOH 基地址：0x4002 1C00

GPIOI 基地址：0x4002 2000

### 7.4.1. 端口控制寄存器(GPIOx_CTL, x = A…I)

地址偏移：0x00

复位值：端口 A 0xA800 0000；端口 B 0x0000 0280；其他端口 0x0000 0000。

该寄存器可以按字节(8 位)、半字(16 位)或字(32 位)访问。

| 31 30 | 29 28 | 27 26 | 25 24 | 23 22 | 21 20 | 19 18 | 17 16 |
|---|---|---|---|---|---|---|---|
| CTL15[1:0] | CTL14[1:0] | CTL13[1:0] | CTL12[1:0] | CTL11[1:0] | CTL10[1:0] | CTL9[1:0] | CTL8[1:0] |
| rw | rw | rw | rw | rw | rw | rw | rw |

| 15 14 | 13 12 | 11 10 | 9 8 | 7 6 | 5 4 | 3 2 | 1 0 |
|---|---|---|---|---|---|---|---|
| CTL7[1:0] | CTL6[1:0] | CTL5[1:0] | CTL4[1:0] | CTL3[1:0] | CTL2[1:0] | CTL1[1:0] | CTL0[1:0] |
| rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:30 | CTL15[1:0] | Pin 15 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 29:28 | CTL14[1:0] | Pin 14 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 27:26 | CTL13[1:0] | Pin 13 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 25:24 | CTL12[1:0] | Pin 12 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |

| | | |
|---|---|---|
| 23:22 | CTL11[1:0] | Pin 11 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 21:20 | CTL10[1:0] | Pin 10 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 19:18 | CTL9[1:0] | Pin 9 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 17:16 | CTL8[1:0] | Pin 8 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 15:14 | CTL7[1:0] | Pin 7 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 13:12 | CTL6[1:0] | Pin 6 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 11:10 | CTL5[1:0] | Pin 5 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 9:8 | CTL4[1:0] | Pin 4 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 7:6 | CTL3[1:0] | Pin 3 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 5:4 | CTL2[1:0] | Pin 2 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 3:2 | CTL1[1:0] | Pin 1 配置位<br>该位由软件置位和清除。<br>参考 CTL0[1:0]的描述 |
| 1:0 | CTL0[1:0] | Pin 0 配置位<br>该位由软件置位和清除。<br>00：GPIO 输入模式(复位值)<br>01：GPIO 输出模式<br>10：备用功能描述 |

11：模拟模式(输入和输出)

### 7.4.2. 端口输出模式寄存器(GPIOx_OMODE, x = A…I)

地址偏移：0x04
复位值：0x0000 0000

该寄存器可以按字节(8 位)、半字(16 位)或字(32 位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OM15 | OM14 | OM13 | OM12 | OM11 | OM10 | OM9 | OM8 | OM7 | OM6 | OM5 | OM4 | OM3 | OM2 | OM1 | OM0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15 | OM15 | Pin 15 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 14 | OM14 | Pin 14 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 13 | OM13 | Pin 13 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 12 | OM12 | Pin 12 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 11 | OM11 | Pin 11 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 10 | OM10 | Pin 10 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 9 | OM9 | Pin 9 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 8 | OM8 | Pin 8 输出模式位<br>该位由软件置位和清除。 |

参考 OM0 的描述

| | | |
|---|---|---|
| 7 | OM7 | Pin 7 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 6 | OM6 | Pin 6 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 5 | OM5 | Pin 5 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 4 | OM4 | Pin 4 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 3 | OM3 | Pin 3 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 2 | OM2 | Pin 2 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 1 | OM1 | Pin 1 输出模式位<br>该位由软件置位和清除。<br>参考 OM0 的描述 |
| 0 | OM0 | Pin 0 输出模式位<br>该位由软件置位和清除。<br>0：端口输出推挽模式(复位值)<br>1：端口输出开漏模式 |

### 7.4.3. 端口输出速度寄存器(GPIOx_OSPD, x = A…I)

地址偏移：0x08

复位值：端口 A 0x0C00 0000；端口 B 0x0000 00C0；其他端口 0x0000 0000。

该寄存器可以按字节(8 位)、半字(16 位)或字(32 位)访问。

| 31 30 | 29 28 | 27 26 | 25 24 | 23 22 | 21 20 | 19 18 | 17 16 |
|---|---|---|---|---|---|---|---|
| OSPD15[1:0] | OSPD14[1:0] | OSPD13[1:0] | OSPD12[1:0] | OSPD11[1:0] | OSPD10[1:0] | OSPD9[1:0] | OSPD8[1:0] |
| rw | rw | rw | rw | rw | rw | rw | rw |
| 15 14 | 13 12 | 11 10 | 9 8 | 7 6 | 5 4 | 3 2 | 1 0 |
| OSPD7[1:0] | OSPD6[1:0] | OSPD5[1:0] | OSPD4[1:0] | OSPD3[1:0] | OSPD2[1:0] | OSPD1[1:0] | OSPD0[1:0] |
| rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:30 | OSPD15[1:0] | Pin 15 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 29:28 | OSPD14[1:0] | Pin 14 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 27:26 | OSPD13[1:0] | Pin 13 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 25:24 | OSPD12[1:0] | Pin 12 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 23:22 | OSPD11[1:0] | Pin 11 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 21:20 | OSPD10[1:0] | Pin 10 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 19:18 | OSPD9[1:0] | Pin 9 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 17:16 | OSPD8[1:0] | Pin 8 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 15:14 | OSPD7[1:0] | Pin 7 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 13:12 | OSPD6[1:0] | Pin 6 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 11:10 | OSPD5[1:0] | Pin 5 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 9:8 | OSPD4[1:0] | Pin 4 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 7:6 | OSPD3[1:0] | Pin 3 输出最大速度位 |

| | | |
|---|---|---|
| | | 该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 5:4 | OSPD2[1:0] | Pin 2 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 3:2 | OSPD1[1:0] | Pin 1 输出最大速度位<br>该位由软件置位和清除。<br>参考 OSPD0[1:0]的描述 |
| 1:0 | OSPD0[1:0] | Pin 0 输出最大速度位<br>该位由软件置位和清除。<br>00：输出速度等级 0(复位值)<br>01：输出速度等级 1<br>10：输出速度等级 2<br>11：输出速度等级 3 |

### 7.4.4. 端口上拉/下拉寄存器(GPIOx_PUD, x = A…I)

地址偏移：0x0C

复位值：端口 A 0x6400 0000；端口 B 0x0000 0100；其他端口 0x0000 0000。

该寄存器可以按字节(8 位)、半字(16 位)或字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PUD15[1:0] | | PUD14[1:0] | | PUD13[1:0] | | PUD12[1:0] | | PUD11[1:0] | | PUD10[1:0] | | PUD9[1:0] | | PUD8[1:0] | |
| rw | | rw | | rw | | rw | | rw | | rw | | rw | | rw | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PUD7[1:0] | | PUD6[1:0] | | PUD5[1:0] | | PUD4[1:0] | | PUD3[1:0] | | PUD2[1:0] | | PUD1[1:0] | | PUD0[1:0] | |
| rw | | rw | | rw | | rw | | rw | | rw | | rw | | rw | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:30 | PUD15[1:0] | Pin 15 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 29:28 | PUD14[1:0] | Pin 14 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 27:26 | PUD13[1:0] | Pin 13 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 25:24 | PUD12[1:0] | Pin 12 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |

| | | |
|---|---|---|
| 23:22 | PUD11[1:0] | Pin 11 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 21:20 | PUD10[1:0] | Pin 10 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 19:18 | PUD9[1:0] | Pin 9 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 17:16 | PUD8[1:0] | Pin 8 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 15:14 | PUD7[1:0] | Pin 7 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 13:12 | PUD6[1:0] | Pin 6 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 11:10 | PUD5[1:0] | Pin 5 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 9:8 | PUD4[1:0] | Pin 4 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 7:6 | PUD3[1:0] | Pin 3 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 5:4 | PUD2[1:0] | Pin 2 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 3:2 | PUD1[1:0] | Pin 1 上拉 / 下拉位<br>该位由软件置位和清除。<br>参考 PUD0[1:0]的描述 |
| 1:0 | PUD0[1:0] | Pin 0 上拉 / 下拉位<br>该位由软件置位和清除。<br>00：浮空模式，无上拉/下拉(复位值)<br>01：端口上拉模式<br>10：端口下拉模式 |

11：保留

### 7.4.5. 端口输入状态寄存器(GPIOx_ISTAT, x = A…I)

地址偏移：0x10
复位值：0x0000 XXXX

该寄存器可以按字节(8 位)、半字(16 位)或字(32 位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ISTAT15 | ISTAT14 | ISTAT13 | ISTAT12 | ISTAT11 | ISTAT10 | ISTAT 9 | ISTAT 8 | ISTAT 7 | ISTAT6 | ISTAT5 | ISTAT4 | ISTAT3 | ISTAT2 | ISTAT1 | ISTAT0 |
| r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | ISTATy | 端口输入状态位(y=0..15)<br>这些位由软件置位和清除。<br>0：引脚输入信号为低电平<br>1：引脚输入信号为高电平 |

### 7.4.6. 端口输出控制寄存器(GPIOx_OCTL, x = A…I)

地址偏移：0x14
复位值：0x0000 0000

该寄存器可以按字节(8 位)、半字(16 位)或字(32 位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OCTL15 | OCTL14 | OCTL13 | OCTL12 | OCTL11 | OCTL10 | OCTL9 | OCTL8 | OCTL7 | OCTL6 | OCTL5 | OCTL4 | OCTL3 | OCTL2 | OCTL1 | OCTL0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | OCTLy | 端口输出控制位(y=0..15)<br>这些位由软件置位和清除。<br>0：引脚输出低电平<br>1：引脚输出高电平 |

### 7.4.7. 端口位操作寄存器(GPIOx_BOP, x = A…I)

地址偏移：0x18
复位值：0x0000 0000

该寄存器可以按字节(8 位)、半字(16 位)或字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CR15 | CR14 | CR13 | CR12 | CR11 | CR10 | CR9 | CR8 | CR7 | CR6 | CR5 | CR4 | CR3 | CR2 | CR1 | CR0 |
| w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BOP15 | BOP14 | BOP13 | BOP12 | BOP11 | BOP10 | BOP9 | BOP8 | BOP7 | BOP6 | BOP5 | BOP4 | BOP3 | BOP2 | BOP1 | BOP0 |
| w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | CRy | 端口清除位 y(y=0..15)<br>这些位由软件置位和清除。<br>0：相应的 OCTLy 位没有改变<br>1：清除相应的 OCTLy 位为 0 |
| 15:0 | BOPy | 端口置位位 y(y=0..15)<br>这些位由软件置位和清除。<br>0：相应的 OCTLy 位没有改变<br>1：设置相应的 OCTLy 位为 1 |

### 7.4.8. 端口配置锁定寄存器(GPIOx_LOCK, x = A…I)

地址偏移：0x1C
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | | | | LKK |
| | | | | | | | | | | | | | | | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| LK15 | LK14 | LK13 | LK12 | LK11 | LK10 | LK9 | LK8 | LK7 | LK6 | LK5 | LK4 | LK3 | LK2 | LK1 | LK0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:17 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 16 | LKK | 锁定序列键<br>该位只能通过使用 Lock Key 写序列设置，始终可读。<br>0：GPIO_LOCK 寄存器和端口配置没有锁定<br>1：直到下一次 MCU 复位前，GPIO_LOCK 寄存器被锁定<br>LOCK Key 写序列：<br>写 1->写 0->写 1-> 读 0->读 1<br>**注意：**在 LOCK Key 写序列期间，LK[15:0]的值必须保持。 |
| 15:0 | LKy | 端口锁定位 y(y=0..15)<br>这些位由软件置位和清除。<br>0：相应的端口位配置没有锁定<br>1：当 LKK 位置 1 时，相应的端口位配置被锁定 |

## 7.4.9. 备用功能选择寄存器 0(GPIOx_AFSEL0, x = A…I)

地址偏移：0x20
复位值：0x0000 0000

该寄存器可以按字节(8 位)、半字(16 位)或字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SEL7[3:0] | | | | SEL6[3:0] | | | | SEL5[3:0] | | | | SEL4[3:0] | | | |
| rw | | | | rw | | | | rw | | | | rw | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SEL3[3:0] | | | | SEL2[3:0] | | | | SEL1[3:0] | | | | SEL0[3:0] | | | |
| rw | | | | rw | | | | rw | | | | rw | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | SEL7[3:0] | Pin 7 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL0 [3:0]的描述 |
| 27:24 | SEL6[3:0] | Pin 6 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL0 [3:0]的描述 |
| 23:20 | SEL5[3:0] | Pin 5 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL0 [3:0]的描述 |
| 19:16 | SEL4[3:0] | Pin 4 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL0 [3:0]的描述 |
| 15:12 | SEL3[3:0] | Pin 3 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL0 [3:0]的描述 |

| | | |
|---|---|---|
| 11:8 | SEL2[3:0] | Pin 2 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL0 [3:0]的描述 |
| 7:4 | SEL1[3:0] | Pin 1 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL0 [3:0]的描述 |
| 3:0 | SEL0[3:0] | Pin 0 备用功能选择<br>该位由软件置位和清除。<br>0000：选择 AF0 功能(复位值)<br>0001：选择 AF1 功能<br>0010：选择 AF2 功能<br>0011：选择 AF3 功能<br>…<br>1111：选择 AF15 功能 |

### 7.4.10. 备用功能选择寄存器 1(GPIOx_AFSEL1, x = A…I)

地址偏移：0x24
复位值：0x0000 0000

该寄存器可以按字节(8 位)、半字(16 位)或字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SEL15[3:0] | | | | SEL14[3:0] | | | | SEL13[3:0] | | | | SEL12[3:0] | | | |
| rw | | | | rw | | | | rw | | | | rw | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SEL11[3:0] | | | | SEL10[3:0] | | | | SEL9[3:0] | | | | SEL8[3:0] | | | |
| rw | | | | rw | | | | rw | | | | rw | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | SEL15[3:0] | Pin 15 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL8[3:0]的描述 |
| 27:24 | SEL14[3:0] | Pin 14 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL8[3:0]的描述 |
| 23:20 | SEL13[3:0] | Pin 13 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL8[3:0]的描述 |
| 19:16 | SEL12[3:0] | Pin 12 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL8[3:0]的描述 |

| | | |
|---|---|---|
| 15:12 | SEL11[3:0] | Pin 11 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL8[3:0]的描述 |
| 11:8 | SEL10[3:0] | Pin 10 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL8[3:0]的描述 |
| 7:4 | SEL9[3:0] | Pin 9 备用功能选择<br>该位由软件置位和清除。<br>参考 SEL8[3:0]的描述 |
| 3:0 | SEL8[3:0] | Pin 8 备用功能选择<br>该位由软件置位和清除。<br>0000：选择 AF0 功能(复位值)<br>0001：选择 AF1 功能<br>0010：选择 AF2 功能<br>0011：选择 AF3 功能<br>…<br>1111：选择 AF15 功能 |

## 7.4.11. 位清除寄存器(GPIOx_BC, x = A…I)

地址偏移：0x28

复位值：0x0000 0000

该寄存器可以按字节(8 位)、半字(16 位)或字(32 位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CR15 | CR14 | CR13 | CR12 | CR11 | CR10 | CR9 | CR8 | CR7 | CR6 | CR5 | CR4 | CR3 | CR2 | CR1 | CR0 |
| w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CRy | 端口清除位 y(y=0..15)<br>这些位由软件置位和清除。<br>0：相应 OCTLy 位没有改变<br>1：清除相应的 OCTLy 位 |

## 7.4.12. 端口位翻转寄存器(GPIOx_TG, x = A…I)

地址偏移：0x2C

复位值：0x0000 0000

该寄存器可以按字节(8 位)、半字(16 位)或字(32 位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TG15 | TG14 | TG13 | TG12 | TG11 | TG10 | TG9 | TG8 | TG7 | TG6 | TG5 | TG4 | TG3 | TG2 | TG1 | TG0 |
| w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | TGy | 端口翻转位 y(y=0..15)<br>这些位由软件置位和清除。<br>0：相应 OCTLy 位没有改变<br>1：翻转相应的 OCTLy 位 |

# 8. 循环冗余校验计算单元(CRC)

## 8.1. 简介

循环冗余校验码是一种用在数字网络和存储设备上的差错校验码，可以校验原始数据的偶然差错。

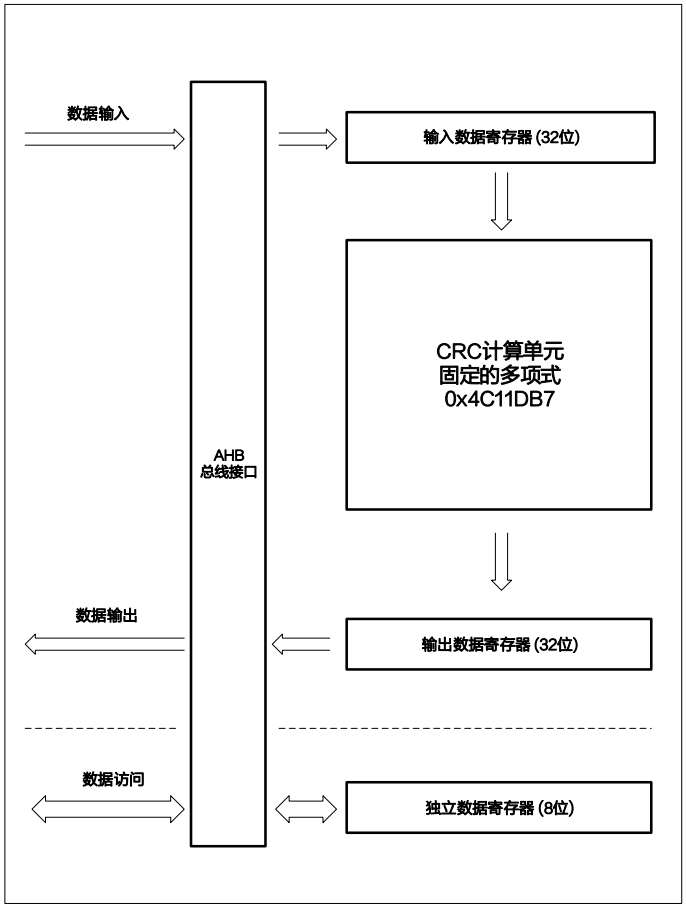
CRC 计算单元使用固定多项式计算 32 位 CRC 校验码。

## 8.2. 主要特征

- 32位数据输入/输出寄存器。对于32位的输入数据，从数据输入到得出计算结果，需要4个AHB的时钟周期；
- 配有与计算无关的独立8位寄存器，可以供其他任何外设使用；
- 固定的计算多项式：0x4C11DB7：
  $X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^{8}+X^{7}+X^{5}+X^{4}+X^{2}+X+1$

该 32 位 CRC 多项式与以太网 CRC 计算多项式相同。

**图 8-1. CRC 计算单元框图**



## 8.3. 功能说明

- CRC计算单元可以用来计算32位的原始数据，CRC_DATA寄存器接收原始数据并存储计算结果。

  如果不通过软件设置CRC_CTL寄存器的方式来清除CRC_DATA寄存器，CRC计算单元将基于新输入的原始数据和前一次CRC_DATA寄存器中的结果进行计算。

  对于32位的数据，CRC的计算需要4个AHB的时钟周期。在此期间，因为32位输入缓存的存在，AHB总线将不会被挂起。

- 此模块提供了一个8位的独立数据寄存器CRC_FDATA。
  CRC_FDATA与CRC计算无关，任何时候都可以进行独立的读写操作。

## 8.4. CRC 寄存器

CRC 基地址：0x4002 3000

### 8.4.1. 数据寄存器(CRC_DATA)

地址偏移：0x00
复位值：0xFFFF FFFF

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | DATA[31:0] | CRC计算结果位<br>软件可读可写。<br>该寄存器用于接收待计算的新数据，直接将其写入即可。刚写入的数据不能被读出来，因为读取该寄存器得到的是上次CRC计算的结果。 |

### 8.4.2. 独立数据寄存器(CRC_FDATA)

地址偏移：0x04
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

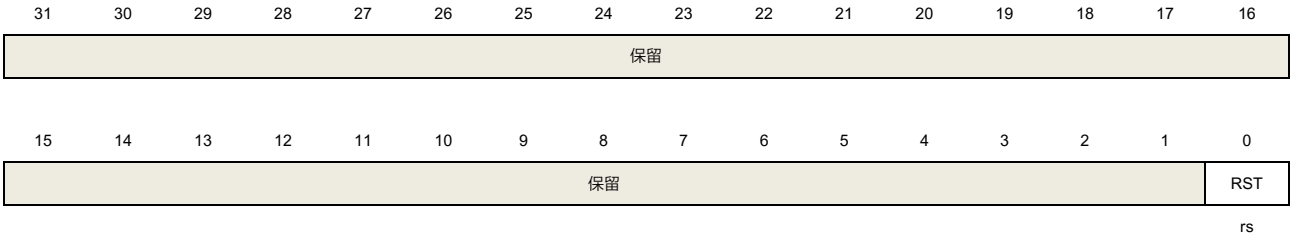
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7:0 | FDATA[7:0] | 独立数据寄存器位<br>软件可读可写。<br>这些位与CRC计算无关。该字节能被任何其他外设用于其他任何目的。该字节不受CRC_CTL寄存器的影响。 |

### 8.4.3. 控制寄存器(CRC_CTL)

地址偏移：0x08

复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:1 | 保留 | 必须保持复位值。 |
| 0 | RST | 将该位置1可以复位CRC_DATA寄存器，并设置其值为0xFFFFFFFF，然后该位被硬件自动清零。该位对CRC_FDATA寄存器没有影响。<br>软件可读可写。 |

# 9. 真随机数生成器(TRNG)

## 9.1. 简介

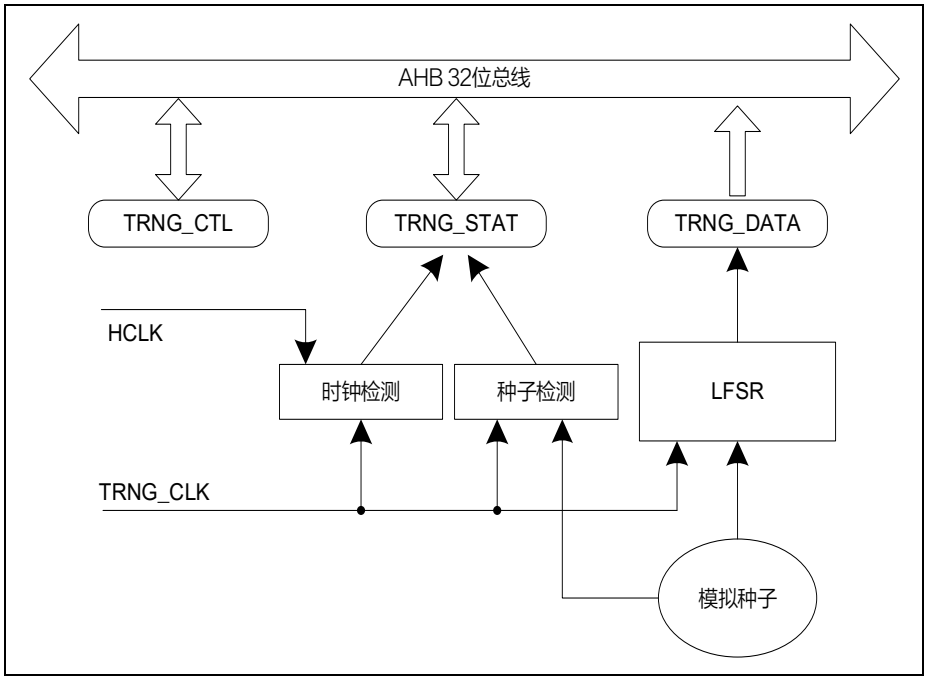
真随机数发生器模块(TRNG)能够通过连续模拟噪声生成一个 32 位的随机数值。

## 9.2. 主要特性

- 两个连续随机数的间隔大约为40个TRNG_CLK时钟周期；
- 32位随机数的种子是由模拟噪声产生的，因此该随机数是一个真随机数值。

## 9.3. 功能描述

**图 9-1. TRNG 模块框图**




随机数种子由模拟电路实现。模拟种子信号输出到一个线性反馈移位寄存器(LFSR)之后在该寄存器中转化成一个 32 位宽度的随机数。

该模拟种子由几个环形振荡器的输出生成。LFSR 由可配置的 TRNG_CLK 时钟(参考 ***RCU*** 相关章节)驱动，因此随机数质量仅与 TRNG_CLK 时钟有关，与 HCLK 频率无关。

当有足够数量的种子被输入 LFSR 之后，LFSR 会输出 32 位数据到 TRNG_DATA 寄存器。同时，系统会监视模拟种子和 TRNG_CLK 时钟。一旦模拟种子发生错误或者时钟产生错误，TRNG_STAT 寄存器的相关状态位将被置 1，如果 TRNG_CTL 寄存器的 TRNGIE 位同时被置 1 还将产生中断。

### 9.3.1. 操作流程

以下步骤为 TRNG 模块的推荐操作流程：

1). 根据需要使能中断，这样当随机数或错误产生时，将会触发一个中断；

2). 使能TRNGEN位；

3). 等待中断产生，检测TRGN_STAT寄存器，如果SEIF=0，CEIF=0并且DRDY=1那么数据寄存器中的随机值可以被读取。

按照 FIPS PUB 140-2 的要求，数据寄存器中的第一个随机数需要保留而不是被使用。每一个新生成的随机数应当与之前的随机数相比较。只有当该随机数与前一个随机数不相等时，该数据才可被使用。

### 9.3.2. 错误标志

(1) 时钟错误

当 TRNG_CLK 时钟频率低于 HCLK 频率的 1/16 时，CECS 和 CEIF 位将被置 1。此时，软件应当检查 TRNG_CLK 和 HCLK 时钟频率配置并清除 CEIF 位。时钟错误对上一个产生的随机数没有影响。

(2) 种子错误

当模拟种子的值在 64 个 TRNG_CLK 时钟周期内不发生变化或连续不断的翻转，SECS 和 SEIF 位将被置位。此时，数据寄存器中的随机数值不应当被使用，并且软件需要清除 SEIF 位。之后将 TRNGEN 位清零并置 1 以便重新启动 TRNG 模块。

## 9.4. TRNG 寄存器

TRNG 基地址：0x5006 0800

### 9.4.1. 控制寄存器 (TRNG_CTL)

地址偏移：0x00
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:4 | 保留 | 必须保持复位值。 |
| 3 | TRNGIE | 中断使能位，当 DRDY，SEIF 或 CEIF 位被置位时该位控制生成一个中断。<br>0：禁止 TRNG 中断<br>1：使能 TRNG 中断 |
| 2 | TRNGEN | TRNG 使能位<br>0：禁止 TRNG 模块<br>1：使能 TRNG 模块 |
| 1:0 | 保留 | 必须保持复位值。 |

### 9.4.2. 状态寄存器 (TRNG_STAT)

地址偏移：0x04
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | SEIF | CEIF | 保留 | | SECS | CECS | DRDY |
| | | | | | | | | | rc_w0 | rc_w0 | | | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:7 | 保留 | 必须保持复位值。 |
| 6 | SEIF | 种子错误中断标志位<br>如果超过 64 个连续位具有相同值或超过 32 组连续交替的 0 和 1 被检测到则此位将置 1。<br>0：未检测到错误<br>1：检测到种子错误。写 0 将清除该位 |
| 5 | CEIF | 时钟错误中断标志位<br>如果 TRNG_CLK 时钟频率低于 HCLK 频率的 1/16 时该位被置位。<br>0：未检测到错误<br>1：检测到时钟错误。写 0 将清除该位 |
| 4:3 | 保留 | 必须保持复位值。 |
| 2 | SECS | 种子错误当前状态<br>0：当前未检测到种子错误。如果 SEIF=1 和 SECS=0，说明之前已经检测到种子错误但现在已恢复正常。<br>1：当前检测到种子错误。如果超过 64 个连续位具有相同值或超过 32 组连续交替的 0 和 1 被检测到时，该位置 1。 |
| 1 | CECS | 时钟错误当前状态<br>0：当前未检测到时钟错误。如果 CEIF=1 和 CECS=0，则意味着之前已检测到时钟错误但现在已恢复正常。<br>1：当前检测到时钟错误。此时 TRNG_CLK 时钟频率低于 1/16 HCLK 频率。 |
| 0 | DRDY | 随机数准备状态位<br>读 TRNG_DATA 寄存器会清零该位，当一个新的随机数产生时被置位。<br>0：TRNG 数据寄存器的内容无效<br>1：TRNG 数据寄存器的内容有效 |

### 9.4.3. 数据寄存器 (TRNG_DATA)

地址偏移：0x08
复位值：0x0000 0000

在读此寄存器之前，软件必须确保 DRDY 位已置 1。

该寄存器只能按字(32 位)访问

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

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

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:0 | TRNGDATA[31:0] | 32 位随机数据 |

# 10. 直接存储器访问控制器(DMA)

## 10.1. 简介

DMA控制器提供了一种硬件的方式在外设和存储器之间或者存储器和存储器之间传输数据，而无需MCU的介入，避免了MCU多次进入中断进行大规模的数据拷贝，最终提高整体的系统性能。

每个DMA控制器包含了两个AHB总线接口和8个4字深度的FIFO，使DMA可以高效的传输数据。DMA控制器(DMA0，DMA1)共有16个通道，每个通道可以被分配给一个或多个特定的外设进行数据传输。两个内置的总线仲裁器用来处理DMA请求的优先级问题。

Cortex®-M4内核与DMA控制器都是通过系统总线来处理数据，引入仲裁机制来处理它们之间的竞争关系。当MCU和DMA指定相同的外设的时候，MCU将会在特定的总线周期挂起。总线矩阵使用了轮询的算法保证MCU至少占用了一半的带宽。

## 10.2. 主要特征

- 两个 AHB 主机接口传输数据，一个 AHB 从机接口配置 DMA。
- 16 个通道(每个 DMA 控制器有 8 个通道)，每个通道连接 8 个特定的外设请求。
- 存储器和外设支持单一传输，4 拍、8 拍和 16 拍增量突发传输。
- 当外设和存储器传输数据时，支持存储器切换。
- 支持软件优先级(低、中、高、超高)和硬件优先级(通道号越低，优先级越高)。
- 存储器和外设的数据传输宽度可配置：字节，半字，字。
- 存储器和外设的数据传输支持固定寻址和增量式寻址。
- 支持循环传输模式。
- 支持三种传输方式：
  - 存储器到外设；
  - 外设到存储器；
  - 存储器到存储器(仅 DMA1 支持)；
- DMA 和外设均可配置为传输控制器：
  - DMA 作为传输控制器：可配置数据传输长度，最大为 65535。
  - 外设作为传输控制器：数据传输的完成取决于外设的最后一个传输请求。
- 支持单数据传输和多数据传输模式：
  - 多数据传输模式：在存储器数据宽度和外设数据宽度不同的时候，自动打包/解包数据；
  - 单数据传输模式：当且仅当 FIFO 空的时候从源地址读取数据，存进 FIFO，然后把 FIFO 的数据写到目标地址。
- 每个通道有 5 种类型的事件标志和独立的中断，支持中断的使能和清除。

## 10.3. 结构框图

**图 10-1. 系统架构**

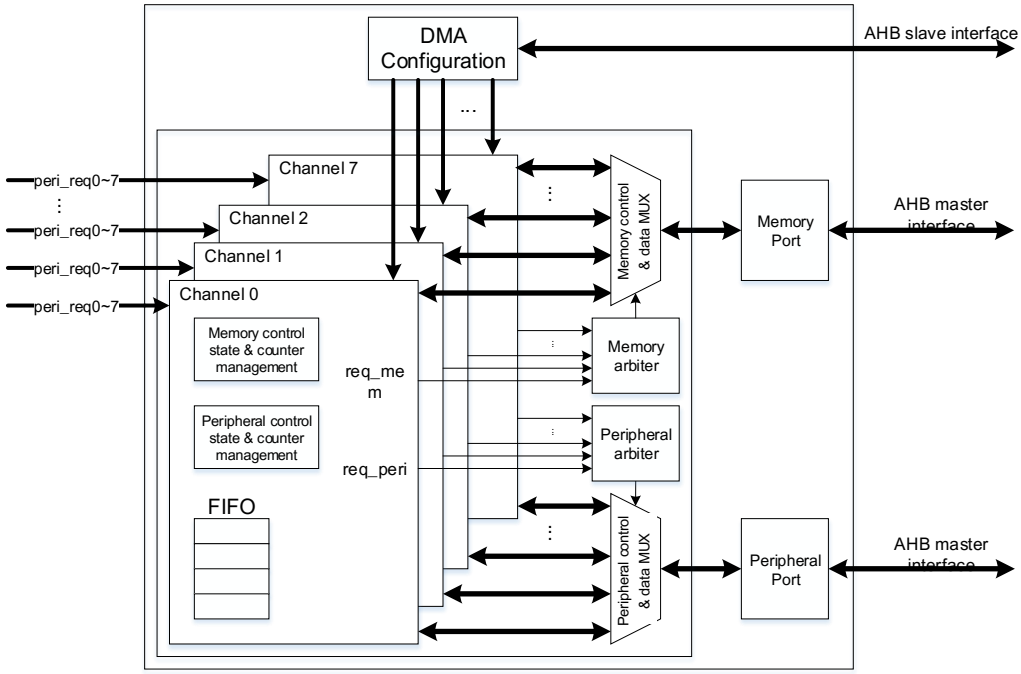



*图 10-1. 系统架构*所示，DMA 控制器由 4 部分组成：

- AHB 从接口配置 DMA；
- 两个 AHB 主接口进行数据传输；
- 两个仲裁器进行 DMA 请求的优先级管理；
- 数据处理和计数。

## 10.4. 功能描述

DMA控制器在没有MCU参与的情况下从一个地址向另一个地址传输数据，它支持多种数据宽度，突发类型，地址生成算法，优先级和传输模式，可以灵活的配置以满足应用的需求。所有的寄存器都可以通过AHB从机接口进行32位的操作。

寄存器DMA_CHxCTL的TM位域决定了DMA的数据传输模式，如*表10-1. 传输模式*所示。

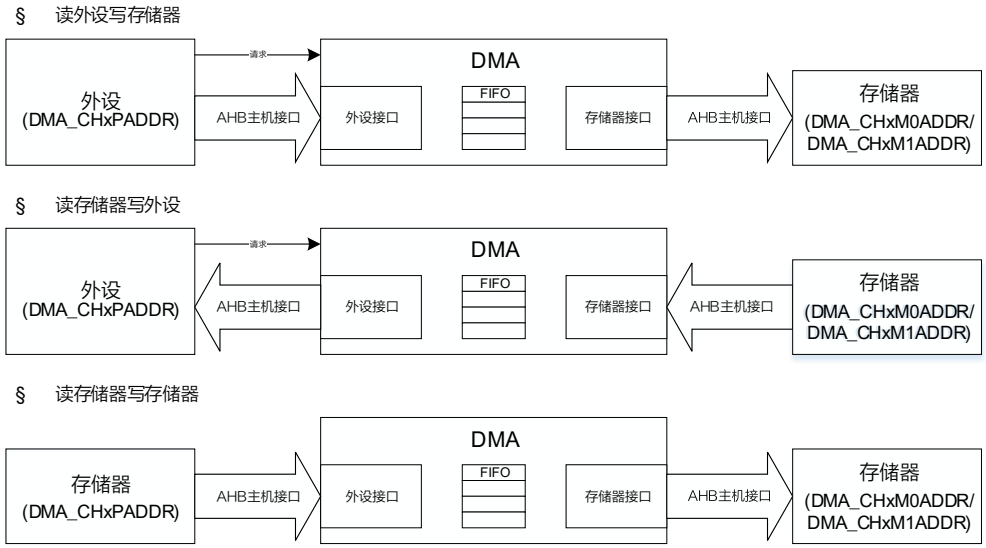
**表 10-1. 传输模式**

| 传输模式 | TM[1:0] | 源地址 | 目的地址 |
|---|---|---|---|
| 外设到存储器 | 00 | DMA_CHxPADDR | DMA_CHxM0ADDR/ DMA_CHxM1ADDR |
| 存储器到外设 | 01 | DMA_CHxM0ADDR/ DMA_CHxM1ADDR | DMA_CHxPADDR |
| 存储器到存储器 | 10 | DMA_CHxPADDR | DMA_CHxM0ADDR/ DMA_CHxM1ADDR |

**注意**：1. 寄存器DMA_CHxCTL的MBS位选择DMA_CHxM0ADDR或者DMA_CHxM1ADDR作为存储器地址。详细请参考*存储切换模式*。

2. 寄存器DMA_CHxCTL的TM位域禁止配置成'0b11'，否则通道将会自动关闭。

**图 10-2. 三种传输模式的数据流**




如*图 10-2. 三种传输模式的数据流*所示，DMA 控制器的两个 AHB 主机接口分别对应存储器和外设的数据访问。

- 外设到存储器：通过 AHB 外设主机接口从外设读取数据，通过 AHB 存储器主机接口向存储器写入数据；
- 存储器到外设：通过 AHB 存储器主机接口从存储器读取数据，通过 AHB 外设主机接口向外设写入数据；
- 存储器到存储器：通过 AHB 外设主机接口从存储器读取数据，通过 AHB 存储器主机接口向存储器写入数据。

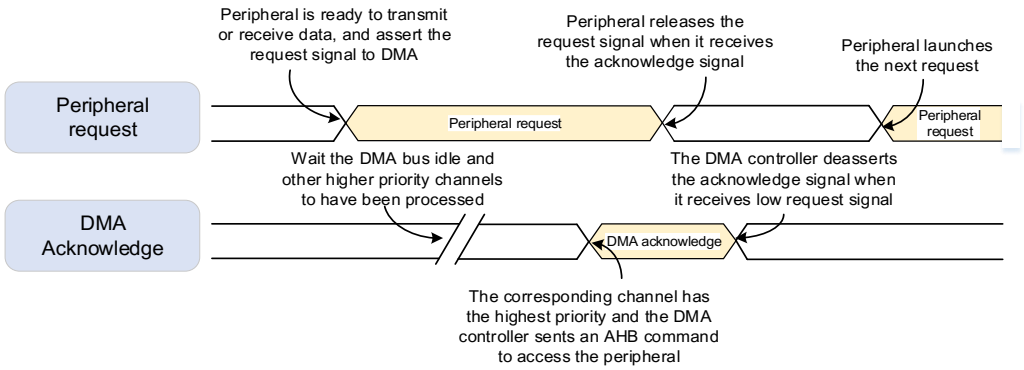
### 10.4.1. 外设握手

为了保证数据的有效传输，DMA控制器中引入了外设和存储器的握手机制，包括请求信号和应答信号：

- 请求信号：由外设发出，表明外设已经准备好发送或接收数据；
- 应答信号：由 DMA 控制器响应，表明 DMA 控制器已经发送 AHB 命令去访问外设。

*图 10-3. 握手机制*中详细描述了 DMA 控制器与外设之间的握手机制。

**图 10-3. 握手机制**




每个 DMA 控制器有 8 个通道，每个通道有多个外设请求。寄存器 DMA_CHxCTL 的 PERIEN 位域决定了 DMA 通道选中的外设请求。DMA0 与 DMA1 的外设请求映射分别列于*表 10-2. DMA0 外设请求*与*表 10-3. DMA1 外设请求*。

如*表 10-2. DMA0 外设请求*，*表 10-3. DMA1 外设请求*所示，同一个外设请求可以连接到两个 DMA 通道上，这里禁止两个 DMA 通道选择相同的外设请求。例如，在 DMA0 控制器中，SPI2_RX 外设请求连接到通道 0 和通道 2。当寄存器 DMA_CH0CTL，DMA_CH2CTL 的 PERIEN 位域同时配置为'0b000'时，使能通道 0 和通道 2，当 SPI2 发出 DMA 请求时，会造成通道 0 和通道 2 的响应混乱，及数据传输错误。

**表 10-2. DMA0 外设请求**

| 通道 | | 通道 0 | 通道 1 | 通道 2 | 通道 3 | 通道 4 | 通道 5 | 通道 6 | 通道 7 |
|---|---|---|---|---|---|---|---|---|---|
| PERIEN[2:0] | 000 | SPI2_RX | • | SPI2_RX | SPI1_RX | SPI1_TX | SPI2_TX | • | SPI2_TX |
| | 001 | I2C0_RX | • | TIMER6_UP | • | TIMER6_UP | I2C0_RX | I2C0_TX | I2C0_TX |
| | 010 | TIMER3_CH0 | • | I2S2_ADD_RX | TIMER3_CH1 | I2S1_ADD_TX | I2S2_ADD_TX | TIMER3_UP | TIMER3_CH2 |
| | 011 | I2S2_ADD_RX | TIMER1_UP<br>TIMER1_CH2 | I2C2_RX | I2S1_ADD_RX | I2C2_TX | TIMER1_CH0 | TIMER1_CH1<br>TIMER1_CH3 | TIMER1_UP<br>TIMER1_CH3 |
| | 100 | UART4_RX | USART2_RX | UART3_RX | USART2_TX | UART3_TX | USART1_RX | USART1_TX | UART4_TX |
| | 101 | UART7_TX | UART6_TX | TIMER2_CH3<br>TIMER2_UP | UART6_RX | TIMER2_CH0<br>TIMER2_TG | TIMER2_CH1 | UART7_RX | TIMER2_CH2 |
| | 110 | TIMER4_CH2<br>TIMER4_UP | TIMER4_CH3<br>TIMER4_TG | TIMER4_CH0 | TIMER4_CH3<br>TIMER4_TG | TIMER4_CH1 | • | TIMER4_UP | • |
| | 111 | • | TIMER5_UP | I2C1_RX | I2C1_RX | USART2_TX | DAC0_OUT0 | DAC0_OUT1 | I2C1_TX |

**表 10-3. DMA1 外设请求**

| 通道 | | 通道 0 | 通道 1 | 通道 2 | 通道 3 | 通道 4 | 通道 5 | 通道 6 | 通道 7 |
|---|---|---|---|---|---|---|---|---|---|
| PERIEN[2:0] | 000 | ADC0 | • | TIMER7_CH0<br>TIMER7_CH1<br>TIMER7_CH2 | • | ADC0 | • | TIMER0_CH0<br>TIMER0_CH1<br>TIMER0_CH2 | • |
| | 001 | • | DCI | ADC1 | ADC1 | • | SPI5_TX | SPI5_RX | DCI |
| | 010 | ADC2 | ADC2 | • | SPI4_RX | SPI4_TX | • | • | • |
| | 011 | SPI0_RX | • | SPI0_RX | SPI0_TX | • | SPI0_TX | • | • |

| 通道 | | 通道 0 | 通道 1 | 通道 2 | 通道 3 | 通道 4 | 通道 5 | 通道 6 | 通道 7 |
|---|---|---|---|---|---|---|---|---|---|
| | 100 | SPI3_RX | SPI3_TX | USART0_RX | SDIO | • | USART0_RX | SDIO | USART0_TX |
| | 101 | • | USART5_RX | USART5_RX | SPI3_RX | SPI3_TX | • | USART5_TX | USART5_TX |
| | 110 | TIMER0_TG | TIMER0_CH0 | TIMER0_CH1 | TIMER0_CH0 | TIMER0_CH3<br>TIMER0_TG<br>TIMER0_CMT | TIMER0_UP | TIMER0_CH2 | • |
| | 111 | • | TIMER7_UP | TIMER7_CH0 | TIMER7_CH1 | TIMER7_CH2 | SPI4_RX | SPI4_TX | TIMER7_CH3<br>TIMER7_TG<br>TIMER7_CMT |

## 10.4.2. 数据处理

### 仲裁

每个DMA控制器有两个分别对应于外设和存储器的仲裁器。当DMA控制器在同一时间接收到多个外设请求时，仲裁器将根据外设请求的优先级来决定响应哪一个外设请求。优先级规则如下：

- 软件优先级：分为4级，低，中，高和超高。可以通过寄存器DMA_CHxCTL的PRIO位域来配置；
- 硬件优先级：当通道具有相同的软件优先级时，编号低的通道优先级高。例：通道0和通道2配置为相同的软件优先级时，通道0的优先级高于通道2。

### 传输宽度，突发传输和计数

#### 传输宽度

寄存器DMA_CHxCTL的PWIDTH和MWIDTH位域决定了外设和存储器的数据传输宽度。DMA控制器支持8位，16位和32位的数据宽度。在多数据传输模式中，如果PWIDTH和MWIDTH不相等，DMA会自动的打包/解包数据来进行完整的数据传输。在单数据传输模式中，MWIDTH在通道使能以后，会被硬件强制设置与PWIDTH相等。

#### 突发传输类型

寄存器DMA_CHxCTL的PBURST和MBURST位域决定了外设和存储器的突发传输方式。DMA控制器的外设和存储器接口均支持单一传输，4拍，8拍，16拍的增量突发传输。对于单数据传输模式，当使能通道后，PBURST和MBURST会被强制设为0，仅支持单一传输。

在外设到存储器或者存储器到外设传输模式中，如果PBURST不为0，在每次外设请求之后，DMA控制器会根据PBURST的值进行4拍，8拍，16拍的增量突发传输。如果剩余的数据不够一次突发传输，剩余的数据将会进行单一传输。

AMBA协议指定突发传输不能超过1KB的地址边界，否则会产生传输错误。对于外设和存储器，当突发传输超过1KB的地址边界，硬件会自动的把4拍，8拍，16拍(由PBURST和MBURST决定)的突发传输拆分为单一传输。

**传输计数**

当DMA作为传输控制器的时候，寄存器DMA_CHxCN的CNT位域决定了需要传输数据的数量，在使能DMA通道之前，数据的传输量必须完成配置。当外设作为传输控制器的时候，在使能通道后，CNT位会被强制设置为'4bFFFF'。在传输过程中，CNT表示剩余需要传输的数据数量。

CNT位的大小与外设的数据传输宽度有关，数据传输总量的字节数等于CNT乘以外设数据传输宽度。举例来说，如果PWIDTH的值设置为'2b10'，则传输的数据总量的字节数等于CNT*4。CNT的值在外设的每次单一传输或者在突发模式中每个节拍传输完成后都会减1。

CNT值的配置需要满足下列要求：

如果关闭循环模式(清除寄存器 DMA_CHxCTL 的 CMEN 位)，CNT 值的配置应该满足*表 10-4. CNT 配置*的要求。传输的数据总量的字节数必须是存储器数据传输宽度的整数倍，以保证完整的存储器传输。

**注意**：如果PBURST和MBURST都不是'2b00'，传输的数据总量不用是存储器和外设的突发传输数据的整数倍。对于不满足一次突发传输的剩余数据，硬件会自动的拆分成多个单一传输。

**表 10-4. CNT 配置**

| PWIDTH | MWIDTH | CNT |
|---|---|---|
| 8 位 | 16 位 | 2 的倍数 |
| 8 位 | 32 位 | 4 的倍数 |
| 16 位 | 32 位 | 2 的倍数 |
| 其他情况 | | 任意值 |

1. 如果开启循环模式(置位寄存器 DMA_CHxCTL 的 CMEN 位)，传输的数据总量必须保证同时是存储器突发传输数据总量和外设突发传输数据总量的整数倍，否则将不能保证数据的正确性：

    a. CNT/PBURST_beats 必须是整数
    b. (CNT*PWIDTH_bytes)/(MURST_beats*MWIDTH_bytes)必须是整数

**注意：**

PWIDTH_bytes是外设的数据传输宽度的字节数。8位是1，16位是2，32位是4。

PBURST_beats是外设突发传输的节拍数，单一传输是1，4拍增量突发传输是4，8拍增量突发传输是8，16拍增量突发传输是16。

MWIDTH_bytes是存储器的数据传输宽度的字节数。8位是1，16位是2，32位是4。

MBURST_beats是存储器突发传输的节拍数，单一传输是1，4拍增量突发传输是4，8拍增量突发传输是8，16拍增量突发传输是16。

**举例：**

如果PWIDTH是16位，PBURST是4拍增量突发传输，MWIDTH是8位，MBURST是16拍增量突发传输，则CNT/4与(CNT*2)/(1*16)必须是整数，所以CNT必须是8的整数倍。

如果PWIDTH是8位，PBURST是16拍增量突发传输，MWIDTH是16位，MBURST是4拍增量突发传输，则CNT/16与(CNT*1)/(2*4)必须是整数，所以CNT必须是16的倍数。

**注意**：如果使能了存储切换模式(置位寄存器DMA_CHxCTL的SBMEN位)，循环模式会被硬件强制打开，所以也必须满足上述要求。

### FIFO

DMA控制器的每个通道都有一个4字深度的FIFO用于缓冲数据，从源地址读取的数据会先暂时保存在FIFO中，再传输到目的地址。根据FIFO的配置，DMA控制器支持两种数据处理模式：单数据传输模式和多数据传输模式。在存储器到存储器模式下，DMA控制器仅支持多数据传输模式。

#### 多数据传输模式

多数据传输模式通过把寄存器DMA_CHxFCTL的MDMEN位置1来实现。

在这个模式中，当FIFO有足够的空间时，DMA控制器响应源端的请求，从源地址读取数据存储进FIFO。如果目的端是外设，当FIFO内的数据量满足外设的一次突发传输时，DMA会响应外设的请求。如果目标端是存储器，寄存器DMA_CHxFCTL的FCCV位设置的FIFO临界值决定DMA控制器何时进行将FIFO中的数据写入存储器，当FIFO计数器达到配置的临界值时，FIFO中的所有数据会被写入目标存储器地址。

为了保证正确的数据传输，FIFO 的临界值必须配置为存储器一次突发传输数据量的整数倍。这样才能保证 FIFO 中有足够的数据可以完成存储器突发传输。FIFO 计数器的临界值的设置与存储器数据传输宽度和存储器突发传输类型有关，具体见*表 10-5. FIFO 计数器临界值配置*。

**表 10-5. FIFO 计数器临界值配置**

| MWIDTH | MBURST | FIFO 计数临界值 | | | |
|---|---|---|---|---|---|
| | | 1-word | 2-word | 3-word | 4-word |
| 8 位 | single | 4 次单一传输 | 8 次单一传输 | 12 次单一传输 | 16 次单一传输 |
| | INCR4 | 1 次突发传输 | 2 次突发传输 | 3 次突发传输 | 4 次突发传输 |
| | INCR8 | 错误 | 1 次突发传输 | 错误 | 2 次突发传输 |
| | INCR16 | 错误 | 错误 | 错误 | 1 次突发传输 |
| 16 位 | single | 2 次单一传输 | 4 次单一传输 | 6 次单一传输 | 8 次单一传输 |
| | INCR4 | 错误 | 1 次突发传输 | 错误 | 2 次突发传输 |
| | INCR8 | 错误 | 错误 | 错误 | 1 次突发传输 |
| | INCR16 | 错误 | 错误 | 错误 | 错误 |
| 32 位 | single | 1 次单一传输 | 2 次单一传输 | 3 次单一传输 | 4 次单一传输 |
| | INCR4 | 错误 | 错误 | 错误 | 1 次突发传输 |
| | INCR8 | 错误 | 错误 | 错误 | 错误 |
| | INCR16 | 错误 | 错误 | 错误 | 错误 |

**注意**：当传输模式是外设到存储器时，如果PBURST_beats*PWIDTH_bytes = 16，FIFO计数器临界值不能设置成'2b10'。如果设置成'2b10'，当接收到外设请求时，DMA控制器从外设中读取数据并填满FIFO，然后DMA会向存储器中写入3个字的数据(这个是由FIFO的临界值决定)，同时剩余一个字的数据。这时当新的外设请求来临时，FIFO中没有足够的空间进行下一次的外设突发传输，同时FIFO中的数据没有达到FIFO临界值也不会进行存储器突发传输，会使通道数据传输冻结。

**单数据传输模式**

单数据传输模式通过把寄存器DMA_CHxFCTL的MDMEN位清0来实现。

在这个模式中，DMA控制器一次只能传输一个数据，FIFO计数器临界值的配置(寄存器DMA_CHxFCTL的FCCV位域)没有意义。在这个模式中，当且仅当FIFO空的时候，DMA会响应源端的请求，从源地址读取数据进入FIFO。当FIFO非空时，DMA响应目的端的请求，把FIFO中的数据写入目的地址。

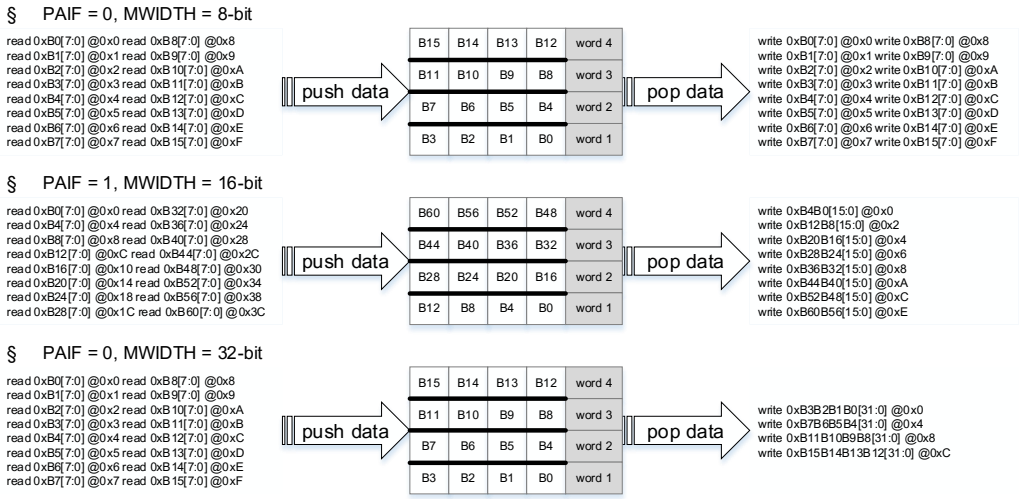
## 打包/解包

在单数据传输模式中，MWIDTH会被硬件强制设置与PWIDTH相等，无需使用数据的打包/解包功能。

在多数据传输模式中，MWIDTH与PWIDTH相互独立，配置更为灵活。当MWIDTH与PWIDTH不相等时，DMA的读写传输宽度不同，DMA会自动的对数据打包/解包操作。在传输过程中，外设和寄存器都只支持小端操作。

当CNT被设置为16，PWIDTH为 '2b00'，PNAGA和MNAGA被置1。对于不同的MWIDTH，DMA的传输操作如*图10-4. PWIDTH为'2b00'时，数据的打包/解包*所示。

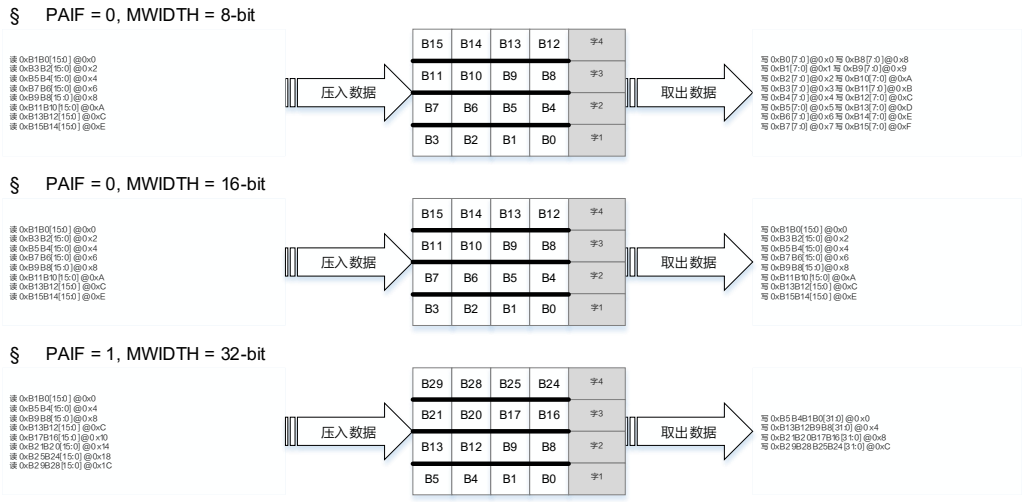
**图 10-4. PWIDTH 为'2b00'时，数据的打包/解包**




当 CNT 被设置为 8，PWIDTH 为'2b01'，PNAGA 和 MNAGA 被置 1。对于不同的 WIDTH，

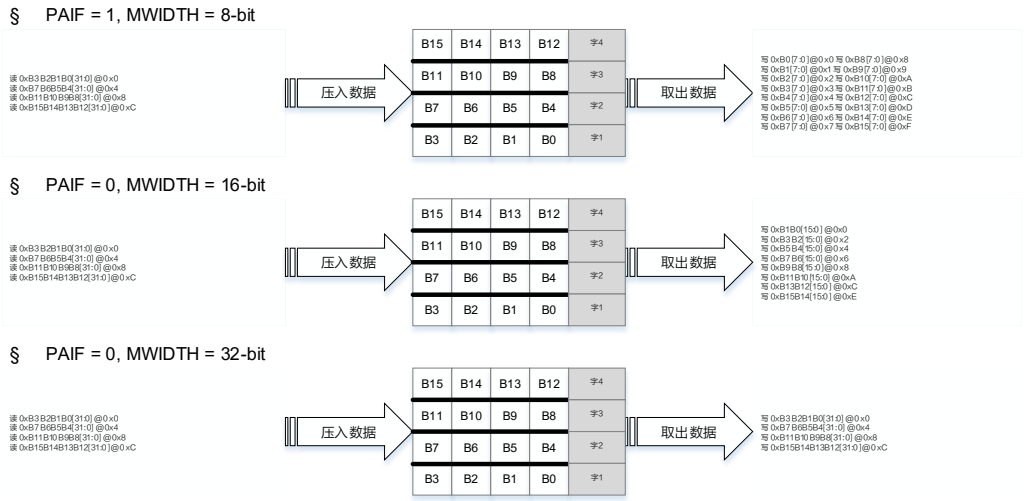
DMA 的传输操作如*图 10-5. PWIDTH 为'2b01'时，数据的打包/解包*所示。

**图 10-5. PWIDTH 为'2b01'时，数据的打包/解包**




当CNT被设置为4，PWIDTH为 '2b10'，PNAGA和MNAGA被置1。对于不同的MWIDTH，DMA的传输操作如*图10-6. PWIDTH为'2b10'时，数据的打包/解包*所示。

**图 10-6. PWIDTH 为'2b10'时，数据的打包/解包**




## 10.4.3. 地址生成

存储器和外设都独立的支持两种地址生成算法：固定模式和增量模式。寄存器DMA_CHxCTL的PNAGA和MNAGA位用来设置存储器和外设的地址生成算法。

在固定模式中，地址一直固定为初始化的基地址(DMA_CHxPADDR，DMA_CHxM0ADDR，DMA_CHxM1ADDR)。

在增量模式中，下一次传输数据的地址是当前地址加1(或者2，4)，这个值取决于数据传输宽度。在多数据传输模式中，若寄存器DMA_CHxCTL的PBURST配置为'2b00'，当寄存器

DMA_CHxCTL的PAIF位置1使能时，外设的下一次传输的地址增量被固定为4，与外设的数据传输宽度无关。PAIF与存储器地址生成无关。

**注意**：若PAIF配置为'1'，外设的基地址(寄存器DMA_CHxPADDR)必须配置为4字节对齐。

### 10.4.4. 循环模式

循环模式用来处理连续的外设请求。可以通过寄存器DMA_CHxCTL的CMEN位置1使能。循环模式只在DMA作为传输控制器时有效。当寄存器DMA_CHxCTL的TFCS位被置1时，外设作为传输控制器，在通道使能后，循环模式会被自动关闭。

在循环模式中，当每次DMA传输完成后，CNT值会被重新载入，且传输完成标志位会被置1。DMA会一直响应外设的请求，直到出现传输错误或者通道使能位被清0。

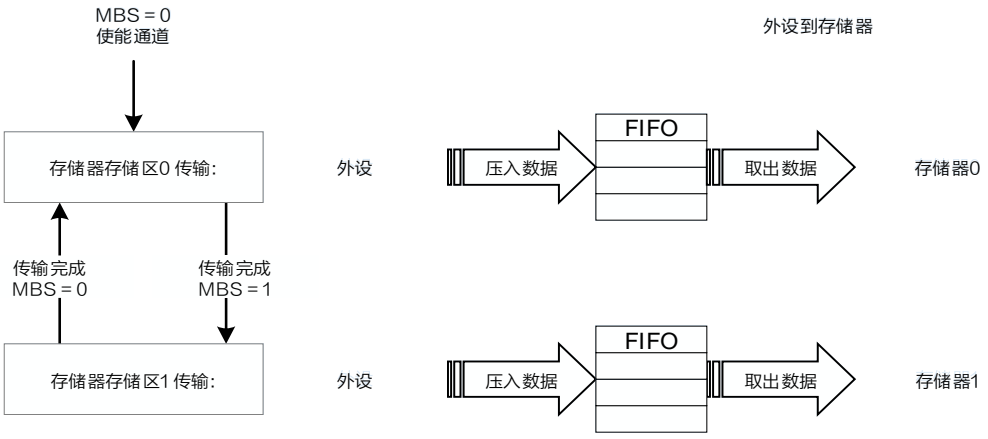
### 10.4.5. 存储切换模式

与循环模式相同，存储切换模式也是用来处理连续的外设请求。可以通过寄存器DMA_CHxCTL的SBMEN位置1使能。若打开了存储切换模式，在通道使能后，硬件会自动打开循环模式。存储切换模式只能应用于外设与存储器之间的数据传输，在存储器到存储器模式中禁止使用。

存储切换模式支持两个存储器缓冲区，两个存储器基地址可以分别在寄存器DMA_CHxM0ADDR和DMA_CHxM1ADDR中配置。在每次DMA传输完成后，存储器指针指向另一个存储器缓冲区。在DMA传输过程中，没有被DMA占用的缓冲区可以被其他的AHB主机接口操作，且其基地址可以改变。

软件可以通过设定寄存器 DMA_CHxCTL 的 MBS 位来指定第一次数据传输 DMA 使用的缓冲区。DMA 通道使能以后，MBS 可以视为 DMA 存储器缓冲区的标志位，它会在每次传输完成后自动在'0'，'1'之间切换，如*图 10-7. 存储切换模式*所示。

**图 10-7. 存储切换模式**



### 10.4.6. 传输控制器

数据传输量的大小由传输控制器决定。寄存器DMA_CHxCTL的TFCS位决定了传输控制器是外设还是DMA。

- DMA 为传输控制器：寄存器 DMA_CHxCNT 的 CNT 位域决定传输数据量的大小，必须在通道使能前配置。
- 外设为传输控制器：在通道使能后寄存器 DMA_CHxCNT 的 CNT 位域为会被硬件强制配置为'FFFF'，因此配置 CNT 没有意义。DMA 数据传输完成由外设发送最后一次传输请求决定。

**注意**：当传输模式是存储器到存储器时，传输控制器只能是DMA。

### 10.4.7. 传输操作

数据传输支持三种操作方式：外设到存储器，存储器到外设，和存储器到存储器。存储器和外设都可以配置为源端和目的端。

#### 存储器端数据传输

- 外设到存储器：
  - 单数据传输模式，当 FIFO 非空时，DMA 启动存储器数据传输，写数据到相应的存储器地址中；
  - 多数据传输模式，当 FIFO 计数器达到临界值时，DMA 启动单一或突发数据传输，把 FIFO 的数据全部写入存储器中。
- 存储器到外设：
  - 单数据传输模式，当通道使能时 DMA 会立刻进行存储器数据传输，读取数据到 FIFO。数据传输过程中，当且仅当 FIFO 为空时，DMA 控制器就会进行存储器读取操作；
  - 多数据传输模式，当通道使能后，不论是否有外设请求，DMA 都会进行单一或突发数据传输填满 FIFO。在数据传输过程中，当 FIFO 有足够的空间进行一次单一或突发传输时，DMA 控制器就会进行存储器读取操作。

存储器到存储器：只支持多数据传输模式。当 FIFO 计数器到达临界值，DMA 进行单一或突发传输把 FIFO 的数据全部写入存储器中。

#### 外设端数据传输

- 外设到存储器：当 DMA 收到外设请求且 FIFO 有足够的空间进行数据传输，DMA 启动外设数据传输从外设读取数据写入 FIFO；
- 存储器到外设：当 DMA 收到外设请求且 FIFO 有足够的数据进行数据传输，DMA 启动外设数据传输从 FIFO 读取数据写入外设；
- 存储器到存储器：只支持多数据传输模式。当通道使能时，DMA 启动单一或突发传输读取数据写满 FIFO。在数据传输过程中，当 FIFO 有足够的空间进行一次单一或突发传输时，DMA 控制器就会进行存储器读取操作。

### 10.4.8. 传输完成

DMA传输由硬件自动完成，寄存器DMA_INTF0与DMA_INTF1位FTFIFx在以下情况下会被置1：

- 传输完成；
- 软件清除；
- 传输错误。

#### 传输完成

当DMA使能以后，数据会在外设和存储器之间传输。当寄存器DMA_CHxCNT的CNT中配置的数据量传输完成或处理完最后一次外设请求以后，DMA传输结束，寄存器DMA_CHxCTL的CHEN位自动清零。

- 外设到存储器：如果 DMA 是传输控制器，CNT 减数到 0 且 FIFO 中的数据完全写入到存储器中，传输完成。如果外设是传输控制器，当外设的最后一个请求完成且 FIFO 中的数据完全写入到存储器中，传输完成；
- 存储器到外设：如果 DMA 是传输控制器，当 CNT 减数到 0 时传输完成。如果外设传输流控制器，当外设的最后一个请求完成时，传输完成；
- 存储器到存储器：只支持 DMA 为传输控制器，CNT 减数到 0 且 FIFO 中的数据完全写入到存储器中，传输完成。

#### 软件清除

DMA传输可以通过对寄存器DMA_CHxCTL的CHEN位清0停止。在清0操作之后，若CHEN仍然为1，代表存储器或者外设仍然处在传输状态，或者FIFO中还有剩余的数据没有传输。

- 外设到存储器：软件清 0 操作后，当前的单次或突发传输完成后，DMA 的外设传输将会停止。为了保证从外设读取的数据完全被写入存储器中，存储器在 FIFO 非空的状态下仍然会进行数据传输，直到 FIFO 中的数据完全被写入存储器中。若 FIFO 中剩余的数据量不满足一次存储器突发传输，这些数据将会被拆分成单一传输。如果 FIFO 总剩余的数据量小于存储器传输宽度，这个数据会被高位补 0，写入存储器中。此时读取 CNT 的值可以计算出存储器中的有效数据量。在 FIFO 中的数据传输完毕之后，CHEN 会被硬件自动清 0，寄存器 DMA_INTF0 或 DMA_INTF1 相应通道标志位 FTFIFx 会被置 1；
- 存储器到外设：软件清 0 操作后，当前的存储器和外设传输完成以后，DMA 传输将会停止。CHEN 会自动清 0，寄存器 DMA_INTF0 或 DMA_INTF1 相应通道标志位 FTFIFx 会被置 1；
- 存储器到存储器：与外设到存储器相同，其中源端存储器的传输通过外设端口来实现。

#### 传输错误

三种类型的错误会关闭DMA传输：

- FIFO 错误：当检测到 FIFO 错误配置，通道会立即关闭且不会进行任何传输。这种情况

下，FTFIFx 不会被置 1。更多 FIFO 错误的信息，请参考章节*错误*。

- 总线错误：当存储器或者外设端口试图访问超出范围的地址时，DMA 控制器会检测到总线错误，通道停止传输且 FTFIFx 不会被置 1。如果错误是由外设端口引起，CNT 仍会进行一次减 1 操作。更多总线错误的信息，请参考章节*错误*。
- 寄存器访问错误：在存储切换模式下，当对 DMA 正在访问的存储器的基地址寄存器进行写操作时，DMA 控制器会检测到寄存器访问错误。发生这个错误后，DMA 控制器的操作与软件清 0 时相同。更多寄存器访问错误的信息，请参考章节*错误*。

### 10.4.9. 通道配置

要启动一次新的DMA数据传输，建议遵循以下步骤进行操作：

1. 读取 CHEN 位，如果为 1(通道已使能)，清 0 或等待 DMA 传输完成。当 CHEN 为 0 时，请按照下列步骤配置 DMA。
2. 清除寄存器 DMA_INTF0 或 DMA_INTF1 相应通道标志位 FTFIFx，否则无法使能 DMA。
3. 配置寄存器 DMA_CHxCTL 的 TM 位选择数据传输方式。
4. 配置寄存器 DMA_CHxCTL 的 PERIEN 位域选择外设。当数据传输方式是存储器到存储器时，PERIEN 没有具体意义，这一步可以跳过。
5. 在寄存器 DMA_CHxCTL 中配置存储器和外设突发类型，目标存储器(memory 0 或 memory1)，存储切换模式，通道优先级，存储器和外设的传输宽度，存储器和外设的地址生成算法，循环模式，传输控制器。
6. 在寄存器 DMA_CHxFCTL 中配置数据处理方式，如果使用多数据传输方式，需要配置 FCCV 位域以设置 FIFO 计数器临界值。
7. 在寄存器 DMA_CHxCTL 中配置传输完成中断，半传输完成中断，传输错误中断，单数据传输方式异常中断的使能位。在寄存器 DMA_CHxFCTL 中配置 FIFO 错误和异常中断的使能位。中断使能位可根据实际需求配置。
8. 在寄存器 DMA_CHxPADDR 中配置外设基地址。
9. 如果使用存储切换模式，在寄存器 DMA_CHxM0ADDR 和 DMA_CHxM1ADDR 中配置两个存储器基地址。如果只使用一个存储器，寄存器 CHxCTL 的 MBS 位决定配置 DMA_CHxM0ADDR 或者 DMA_CHxM1ADDR。
10. 在寄存器 DMA_CHxCNT 中配置数据传输总量。
11. 寄存器 DMA_CHxCTL 的 CHEN 位置 1，使能 DMA 通道。

如果要继续挂起的DMA传输，建议遵循以下步骤进行操作：

1. 读取 CHEN 位，确定 DMA 的挂起操作已经完成。当 CHEN 为 0 时，DMA 处于空闲状态，可以重新配置 DMA 以继续挂起 DMA 传输。
2. 清除寄存器 DMA_INTF0 或 DMA_INTF1 相应通道标志位 FTFIFx，否则 DMA 通道可能无法再使能。
3. 读取寄存器 DMA_CHxCNT 计算出已经发送的数据量与剩余待发的数据量。
4. 在寄存器 DMA_CHxPADDR 中更新外设基地址。
5. 在寄存器 DMA_CHxM0ADDR 或 DMA_CHxM1ADDR 中更新存储器基地址。
6. 在寄存器 DMA_CHxCNT 中配置剩余待发的数据总量。
7. 寄存器 DMA_CHxCTL 的 CHEN 位置 1，重新启动 DMA 通道。

## 10.5. 中断

每个DMA通道都有专有的中断，包括5个中断事件，传输完成中断，半传输完成中断，传输错误中断，单数据传输模式异常中断，FIFO错误和异常中断。任何一个中断事件都可以引发DMA中断。

寄存器 DMA_INTF0 或 DMA_INTF1 包含每个中断事件的标志位，寄存器 DMA_INTC0 或 DMA_INTC1 包含每个中断事件的标志清除位，寄存器 DMA_CHxCTL 和 DMA_CHxFCTL 包含每个中断事件的使能位，如*表 10-6. DMA 中断事件*所示。

**表 10-6. DMA 中断事件**

| 中断事件 | 标志位 | 使能位 | 清除位 |
|---|---|---|---|
| | DMA_INTF0 or DMA_INTF1 | DMA_CHxCTL or DMA_CHxFCTL | DMA_INTC0 or DMA_INTC1 |
| 传输完成 | FTFIF | FTFIE | FTFIFC |
| 半传输完成 | HTFIF | HTFIE | HTFIFC |
| 传输错误 | TAEIF | TAEIE | TAEIFC |
| 单数据模式异常 | SDEIF | SDEIE | SDEIFC |
| FIFO 错误与异常 | FEEIF | FEPIE | FEEIFC |

这5个事件可以分为3种类型：

- 标志：传输完成和半传输完成；
- 异常：单数据传输模式异常和 FIFO 异常；
- 错误：传输错误和 FIFO 错误。

发生异常事件时，正在进行的DMA传输不会被停止，仍将继续传输。发生错误事件时，正在进行的DMA传输会被停止。这三种类型的事件在下一节进行详细描述。

### 10.5.1. 标志

两种标志事件，传输完成事件和半传输完成事件。

发生以下情况时，传输完成标志位将会被置1：

- DMA 作为传输控制器时，CNT 计数到 0。
- 外设作为传输控制器时，响应完外设的最后一个数据传输请求后，(如果是读外设写存储器传输方式，还需满足 FIFO 中的数据全部写入存储器中)传输完成。
- 在数据传输完成之前，通过软件清 0 的方式停止数据传输，当前的存储器和外设数据传输完成后，(如果是外设到存储器或存储器到存储器传输模式，还需满足 FIFO 中的数据全部写入存储器中)传输完成。
- 在数据传输完成之前，由于寄存器访问错误导致停止数据传输，当前的存储器和外设数据传输完成后，(如果是外设到存储器或存储器到存储器传输模式，还需满足 FIFO 中的数据全部写入存储器，)传输完成。

当传输完成标志位置1，且传输完成中断使能时，DMA控制器产生传输完成中断。

当DMA作为传输控制器且CNT减数计数达到初始值的一半时，半传输完成标志位会被置1。

当半传输完成标志位置1，且半传输完成中断使能时，DMA控制器产生半传输完成中断。

### 10.5.2. 异常

两种异常事件，单数据传输模式异常和FIFO异常。异常对于DMA传输无影响。

#### 单数据传输模式异常

这个异常只有在使能单数据传输模式且传输方式为外设到存储器时才会发生。当FIFO非空时，如果外设请求数据传输，DMA在响应外设请求以后，会有多个数据存储在FIFO中，这可能会对存储器后续的数据处理造成影响，此时单数据传输模式异常标志位置1

当单数据传输模式异常标志位置1，且单数据传输模式异常中断使能，DMA控制器产生单数据传输模式异常中断。

#### FIFO 异常

这个异常只有数据在外设和存储器之间传输才会发生，当FIFO发生上溢或下溢时，FIFO异常标志位置1。

当传输模式为外设到存储器时，如果外设请求有效且得到最高优先级时，FIFO的剩余空间不满足单一或突发外设传输，FIFO发生上溢。直到FIFO有足够空间时，DMA控制器才会响应此次外设请求，该异常不会影响到数据传输的正确性。

当传输模式为存储器到外设时，如果外设请求有效且得到最高优先级时，FIFO中的数据不够完成单次或突发外设传输，FIFO发生下溢。直到FIFO有足够数据时，DMA控制器才会响应此次外设请求，该异常不会影响到数据传输的正确性。

当FIFO异常标志位置1，且FIFO异常中断使能时，DMA控制器产生FIFO异常中断。

### 10.5.3. 错误

在数据传输过程中，会发生FIFO错误和传输错误(包含寄存器访问错误和总线错误)，此时数据传输会被中止。

#### FIFO 错误

当使用多数据传输模式时，FIFO计数器临界值设置必须与存储器和外设数据传输宽度匹配，详细见*表10-5. FIFO计数器临界值配置*。错误的配置会引发FIFO错误，此时，通道会立即关闭，并不启动任何传输。

当FIFO错误标志位置1，且FIFO错误中断使能时，DMA控制器产生FIFO错误中断。

### 寄存器访问错误

只有在存储切换模式下才会发生寄存器访问错误。如果软件对DMA正在使用的存储器的基地址寄存器进行写操作，将会发生寄存器访问错误。举例来说，存储器0是DMA控制器正在使用的源端或者目的端地址，如果软件对DMA_CHxM0ADDR寄存器进行写操作，则会产生寄存器访问错误。寄存器访问错误发生后，当前数据传输完成之后(在读外设写存储器传输模式下，FIFO中的数据需要全部写入到memory中)，DMA会被自动停止。

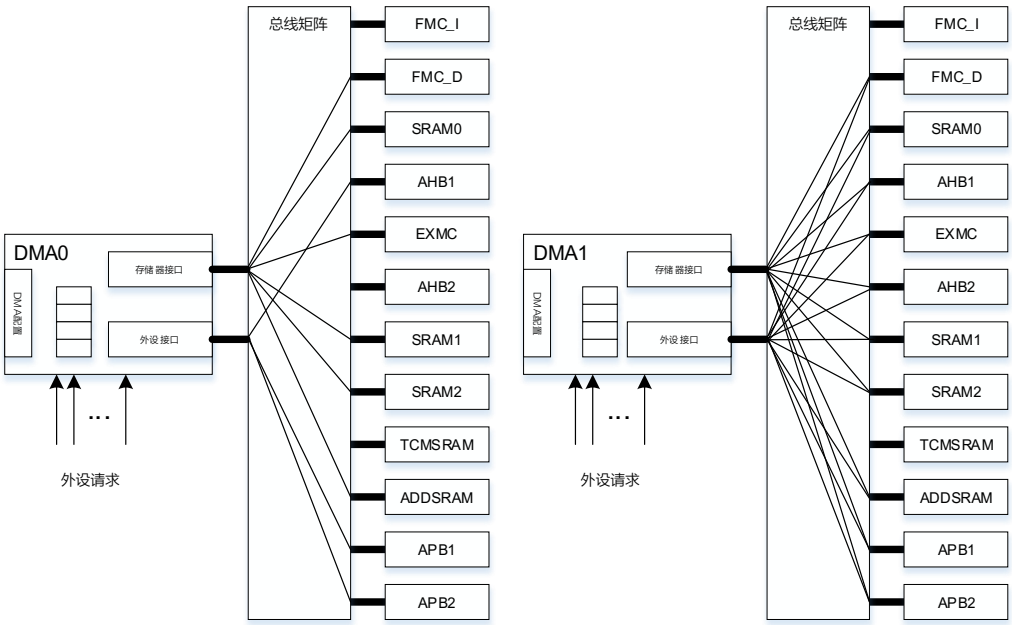
当寄存器访问错误标志位置1，且寄存器访问错误中断使能时，DMA控制器产生寄存器访问错误中断。

### 总线错误

当 DMA 的存储器端或外设端的主机端口访问的地址超出了允许的范围，会发生总线错误，同时 DMA 通道失能。DMA0 和 DMA1 的存储器和外设端口允许访问的地址空间如*图 10-8. DMA0 与 DMA1 的系统连接*所示。

当总线错误标志位置1，且总线错误中断使能时，DMA控制器产生总线错误中断。

**图 10-8. DMA0 与 DMA1 的系统连接**



## 10.6. DMA 寄存器

DMA0 基地址：0x4002 6000

DMA1 基地址：0x4002 6400

### 10.6.1. 中断标志位寄存器 0(DMA_INTF0)

地址偏移：0x00
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | FTFIF3 | HTFIF3 | TAEIF3 | SDEIF3 | 保留 | FEEIF3 | FTFIF2 | HTFIF2 | TAEIF2 | SDEIF2 | 保留 | FEEIF2 |
| | | | | r | r | r | r | | r | r | r | r | r | | r |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | FTFIF1 | HTFIF1 | TAEIF1 | SDEIF1 | 保留 | FEEIF1 | FTFIF0 | HTFIF0 | TAEIF0 | SDEIF0 | 保留 | FEEIF0 |
| | | | | r | r | r | r | | r | r | r | r | r | | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27/21/11/5 | FTFIFx | 通道x的传输完成标志位(x=0…3)<br>硬件置位，软件写DMA_INTC0相应位为1清零。<br>0：通道x传输未完成<br>1：通道x传输完成 |
| 26/20/10/4 | HTFIFx | 通道x的半传输完成标志位(x=0…3)<br>硬件置位，软件写DMA_INTC0相应位为1清零。<br>0：通道x半传输未完成<br>1：通道x半传输完成 |
| 25/19/9/3 | TAEIFx | 通道x的传输错误标志位(x=0…3)<br>硬件置位，软件写DMA_INTC0相应位为1清零。<br>0：通道x未发生传输错误<br>1：通道x发生传输错误 |
| 24/18/8/2 | SDEIFx | 通道x的单数据传输模式异常标志位(x=0…3)<br>硬件置位，软件写DMA_INTC0相应位为1清零。<br>0：通道x未发生单数据传输模式异常<br>1：通道x发生单数据传输模式异常 |
| 23/17/7/1 | 保留 | 必须保持复位值 |
| 22/16/6/0 | FEEIFx | 通道x的FIFO错误与FIFO异常标志位(x=0…3)<br>硬件置位，软件写DMA_INTC0相应位为11清零。<br>0：通道x未发生FIFO错误或FIFO异常 |

1：通道x发生FIFO错误或FIFO异常

### 10.6.2. 中断标志位寄存器 1(DMA_INTF1)

地址偏移：0x04
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | FTFIF7 | HTFIF7 | TAEIF7 | SDEIF7 | 保留 | FEEIF7 | FTFIF6 | HTFIF6 | TAEIF6 | SDEIF6 | 保留 | FEEIF6 |
| | | | | r | r | r | r | | r | r | r | r | r | | r |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | FTFIF5 | HTFIF5 | TAEIF5 | SDEIF5 | 保留 | FEEIF5 | FTFIF4 | HTFIF4 | TAEIF4 | SDEIF4 | 保留 | FEEIF4 |
| | | | | r | r | r | r | | r | r | r | r | r | | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27/21/11/5 | FTFIFx | 通道x的传输完成标志位(x=4…7)<br>硬件置位，软件写DMA_INTC0相应位为1清零。<br>0：通道x传输未完成<br>1：通道x传输完成 |
| 26/20/10/4 | HTFIFx | 通道x的半传输完成标志位(x=4…7)<br>硬件置位，软件写DMA_INTC0相应位为1清零。<br>0：通道x半传输未完成<br>1：通道x半传输完成 |
| 25/19/9/3 | TAEIFx | 通道x的传输错误标志位(x=4…7)<br>硬件置位，软件写DMA_INTC0相应位为1清零。<br>0：通道x未发生传输错误<br>1：通道x发生传输错误 |
| 24/18/8/2 | SDEIFx | 通道x的单数据传输模式异常标志位(x=4…7)<br>硬件置位，软件写DMA_INTC0相应位为1清零。<br>0：通道x未发生单数据传输模式异常<br>1：通道x发生单数据传输模式异常 |
| 23/17/7/1 | 保留 | 必须保持复位值 |
| 22/16/6/0 | FEEIFx | 通道x的FIFO错误与FIFO异常标志位(x=4…7)<br>硬件置位，软件写DMA_INTC0相应位为1清零。<br>0：通道x未发生FIFO错误或FIFO异常<br>1：通道x发生FIFO错误或FIFO异常 |

### 10.6.3. 中断标志位清除寄存器(DMA_INTC0)

地址偏移：0x08
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | FTFIFC3 | HTFIFC3 | TAEIFC3 | SDEIFC3 | 保留 | FEEIFC3 | FTFIFC2 | HTFIFC2 | TAEIFC2 | SDEIFC2 | 保留 | FEEIFC2 |
| | | | | w | w | w | w | | w | w | w | w | w | | w |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | FTFIFC1 | HTFIFC1 | TAEIFC1 | SDEIFC1 | 保留 | FEEIFC1 | FTFIFC0 | HTFIFC0 | TAEIFC0 | SDEIFC0 | 保留 | FEEIFC0 |
| | | | | w | w | w | w | | w | w | w | w | w | | w |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27/21/11/5 | FTFIFCx | 通道x的传输完成标志清除位(x=0…3)<br>0：无影响<br>1：清除传输完成标志位 |
| 26/20/10/4 | HTFIFCx | 通道x的半传输完成标志清除位(x=0…3)<br>0：无影响<br>1：清除半传输完成标志位 |
| 25/19/9/3 | TAEIFCx | 通道x的传输错误标志清除位(x=0…3)<br>0：无影响<br>1：清除传输错误标志位 |
| 24/18/8/2 | SDEIFCx | 通道x的单数据传输模式异常标志清除位(x=0…3)<br>0：无影响<br>1：清除单数据传输模式异常标志位 |
| 23/17/7/1 | 保留 | 必须保持复位值 |
| 22/16/6/0 | FEEIFCx | 通道x的FIFO错误与FIFO异常标志清除位(x=0…3)<br>0：无影响<br>1：清除FIFO错误与FIFO异常标志位 |

### 10.6.4. 中断标志位清除寄存器 1(DMA_INTC1)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | FTFIFC7 | HTFIFC7 | TAEIFC7 | SDEIFC7 | 保留 | FEEIFC7 | FTFIFC6 | HTFIFC6 | TAEIFC6 | SDEIFC6 | 保留 | FEEIFC6 |
| | | | | w | w | w | w | | w | w | w | w | w | | w |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | FTFIFC5 | HTFIFC5 | TAEIFC5 | SDEIFC5 | 保留 | FEEIFC5 | FTFIFC4 | HTFIFC4 | TAEIFC4 | SDEIFC4 | 保留 | FEEIFC4 |
| | | | | w | w | w | w | | w | w | w | w | w | | w |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27/21/11/5 | FTFIFCx | 通道x的传输完成标志清除位(x=4…7)<br>0：无影响<br>1：清除传输完成标志位 |
| 26/20/10/4 | HTFIFCx | 通道x的半传输完成标志清除位(x=4…7)<br>0：无影响<br>1：清除半传输完成标志位 |
| 25/19/9/3 | TAEIFCx | 通道x的传输错误标志清除位(x=4…7)<br>0：无影响<br>1：清除传输错误标志位 |
| 24/18/8/2 | SDEIFCx | 通道x的单数据传输模式异常标志清除位(x=4…7)<br>0：无影响<br>1：清除单数据传输模式异常标志位 |
| 23/17/7/1 | 保留 | 必须保持复位值 |
| 22/16/6/0 | FEEIFCx | 通道x的FIFO错误与FIFO异常标志清除位(x=4…7)<br>0：无影响<br>1：清除FIFO错误与FIFO异常标志位 |

## 10.6.5. 通道 x 控制寄存器(DMA_CHxCTL)

x = 0..7,x为通道编号
地址偏移：0x10 + 0x18*x
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | PERIEN[2:0] | | | MBURST[1:0] | | PBURST[1:0] | | 保留 | MBS | SBMEN | PRIO[1:0] | |
| | | | | rw | | | rw | | rw | | | rw | rw | rw | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PAIF | MWIDTH[1:0] | | PWIDTH[1:0] | | MNAGA | PNAGA | CMEN | TM[1:0] | | TFCS | FTFIE | HTFIE | TAEIE | SDEIE | CHEN |
| rw | rw | | rw | | rw | rw | rw | rw | | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:25 | PERIEN[2:0] | 外设使能 |

| | | |
|---|---|---|
| | | 软件置1与清0<br>000：使能外设0<br>001：使能外设1<br>010：使能外设2<br>011：使能外设3<br>100：使能外设4<br>101：使能外设5<br>110：使能外设6<br>111：使能外设7<br>CHEN为1时不可写入 |
| 24:23 | MBURST[1:0] | 存储器突发类型<br>软件置1与清0<br>00：单一传输<br>01：INCR4 (4拍增量突发传输)<br>10：INCR8 (8拍增量突发传输)<br>11：INCR16 (16拍增量突发传输)<br>CHEN为1时不可写入<br>如果寄存器DMA_CHxFCTL的MDMEN位为0，在使能通道后(CHEN置1)，该位域会被硬件强制清零 |
| 22:21 | PBURST[1:0] | 外设突发类型<br>软件置1与清0<br>00：单一传输<br>01：INCR4 (4拍增量突发传输)<br>10：INCR8 (8拍增量突发传输)<br>11：INCR16 (16拍增量突发传输)<br>CHEN为1时不可写入<br>如果寄存器DMA_CHxFCTL的MDMEN位为0，在使能通道后(CHEN置1)，该位域会被硬件强制清零 |
| 20 | 保留 | 必须保持复位值 |
| 19 | MBS | 存储器缓冲选择<br>硬件置1清0，软件置1清0<br>0：存储器0作为存储器传输区域<br>1：存储器1作为存储器传输区域<br>CHEN为1时不可写入<br>在每次传输完成时，硬件会自动更新该位，以此来表明DMA正在使用哪个存储区 |
| 18 | SBMEN | 存储切换模式使能<br>软件置1与清0<br>0：关闭存储切换模式<br>1：打开存储切换模式<br>CHEN为1时不可写入 |
| 17:16 | PRIO[1:0] | 软件优先级 |

| | | |
|---|---|---|
| | | 软件置1与清0<br>00：低<br>01：中<br>10：高<br>11：超高<br>CHEN为1时不可写入 |
| 15 | PAIF | 外设地址增量固定<br>软件置1与清0<br>0：外设地址增量由PWIDTH决定<br>1：外设地址增量固定为4<br>CHEN为1时不可写入<br>如果PNAGA设置为0，该位无影响<br>如果寄存器DMA_CHxFCTL的MDMEN位为'0'或者PBURST不为'00'，在使能通道后(CHEN置1)，该位域会被硬件强制清零 |
| 14:13 | MWIDTH[1:0] | 存储器传输宽度<br>软件置1与清0<br>00：8位<br>01：16位<br>10：32位<br>11：保留<br>CHEN为1时不可写入<br>如果寄存器DMA_CHxFCTL的MDMEN位为'0'， 在使能通道后(CHEN置1)，该位域会被硬件强制与PWIDTH相等。 |
| 12:11 | PWIDTH[1:0] | 外设传输宽度<br>软件置1与清0<br>00：8位<br>01：16位<br>10：32位<br>11：保留<br>CHEN为1时不可写入 |
| 10 | MNAGA | 存储器地址生成算法<br>软件置1与清0<br>0：固定地址模式<br>1：增量地址模式<br>CHEN为1时不可写入 |
| 9 | PNAGA | 外设地址生成算法<br>软件置1与清0<br>0：固定地址模式<br>1：增量地址模式<br>CHEN为1时不可写入 |
| 8 | CMEN | 循环模式 |

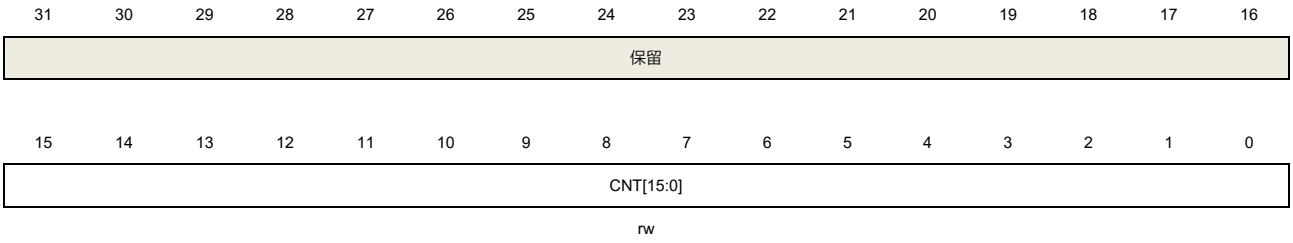
| | | |
|---|---|---|
| | | 软件置1与清0<br>0：关闭循环模式<br>1：打开循环模式<br>CHEN为1时不可写入<br>如果TFCS为'1'， 在使能通道后(CHEN置1)，该位被自动清0<br>如果SBMEN为'1'， 在使能通道后(CHEN置1)，该位被自动置1 |
| 7:6 | TM[1:0] | 传输方式<br>软件置1与清0<br>00：读外设写存储器<br>01：读存储器写外设<br>10：读存储器写存储器<br>11：保留<br>CHEN为1时不可写入 |
| 5 | TFCS | 传输控制器选择<br>软件置1与清0<br>0：DMA作为传输控制器<br>1：外设作为传输控制器<br>CHEN为1时不可写入 |
| 4 | FTFIE | 传输完成中断使能位<br>软件置1与清0<br>0：传输完成中断禁止<br>1：传输完成中断使能 |
| 3 | HTFIE | 半传输完成中断使能位<br>软件置1与清0<br>0：半传输完成中断禁止<br>1：半传输完成中断使能 |
| 2 | TAEIE | 传输错误中断使能位<br>软件置1与清0<br>0：传输错误中断禁止<br>1：传输错误中断使能 |
| 1 | SDEIE | 单数据传输模式异常中断使能位<br>软件置1与清0<br>0：单数据传输模式异常中断禁止<br>1：单数据传输模式异常中断使能 |
| 0 | CHEN | 通道使能<br>软件置1，硬件清0<br>0：通道禁止<br>1：通道使能<br>该位置1，DMA传输开始。发生以下情况该位会被自动清0：<br>■ 数据传输完成<br>– 发生FIFO配置错误或者传输错误 |

软件清0操作后，读该位仍为1代表还有正在进行的数据传输，软件查询该位可以确定DMA通道是否空闲，可以进行新的数据传输。

### 10.6.6. 通道 x 计数寄存器(DMA_CHxCNT)

x = 0...7,x为通道编号
地址偏移：0x14 + 0x18*x
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

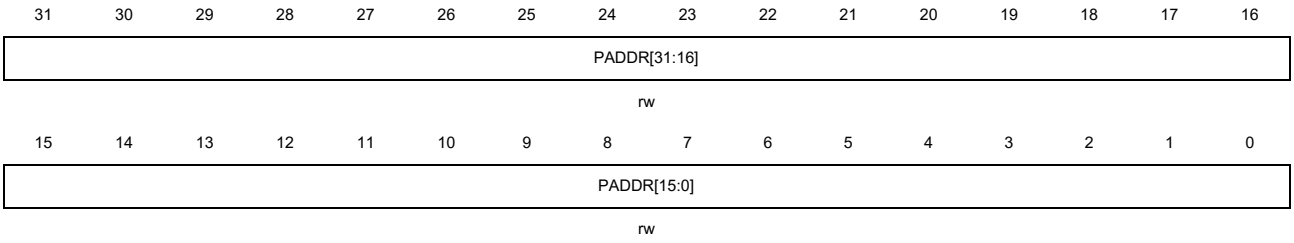



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CNT[15:0] | 传输计数<br>在使能通道后(CHEN置1)，该位域不可写。<br>传输过程中，CNT代表剩余未发的数据量。外设每传输一次数据，CNT减1。如果寄存器DMA_CHxCTL的CMEN位或SBMEN位置1，在每次传输完成时，CNT会由硬件自动重新装载。 |

### 10.6.7. 通道 x 外设基地址寄存器(DMA_CHxPADDR)

x = 0...7,x为通道编号
地址偏移：0x18 + 0x18*x
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | PADDR[31:0] | 外设基地址<br>在使能通道后(CHEN置1)，该位域不可写。<br>当PWIDTH位'01'，最低位被忽略，自动半字对齐 |

当PWIDTH位'10'，最低位两位被忽略，自动字对齐

**注意：**若寄存器DMA_CHxCTL的PAIF位置1，该位域必须配置为4字节对齐。

### 10.6.8. 通道 x 存储器 0 基地址寄存器(DMA_CHxM0ADDR)

x = 0...7,x为通道编号

地址偏移：0x1C + 0x18*x

复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M0ADDR[31:16] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| M0ADDR[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | M0ADDR[31:0] | 存储器0基地址<br>若寄存器DMA_CHxCTL位MBS为0，该位域定义DMA传输过程中存储器的基地址<br>如果寄存器DMA_CHxCTL的CHEN位置1且MBS位为0时，该位域不可写<br>当MWIDTH位'01'，最低位被忽略，自动半字对齐<br>当MWIDTH位'10'，最低位两位被忽略，自动字对齐 |

### 10.6.9. 通道 x 存储器 1 基地址寄存器(DMA_CHxM1ADDR)

x = 0...7,x为通道编号

地址偏移：0x20 + 0x18*x

复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M1ADDR[31:16] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| M1ADDR[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | M1ADDR[31:0] | 存储器1基地址<br>若寄存器DMA_CHxCTL位MBS为1，该位域定义DMA传输过程中存储器的基地址<br>如果寄存器DMA_CHxCTL的CHEN位置1且MBS为1时，该位域不可写<br>当MWIDTH位'01'，最低位被忽略，自动半字对齐 |

当MWIDTH位'10'，最低位两位被忽略，自动字对齐

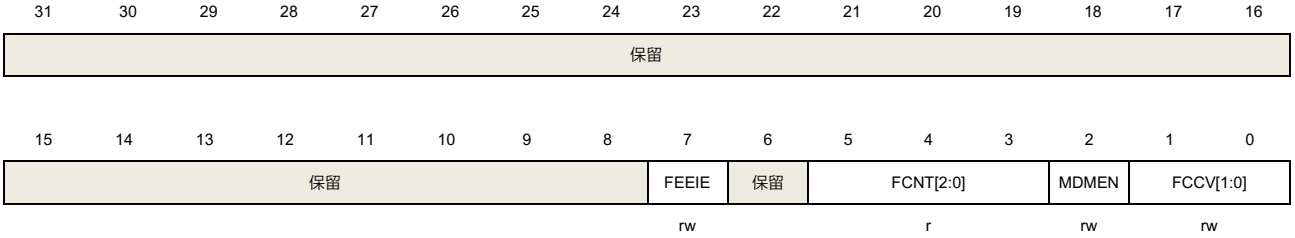
### 10.6.10. 通道 xFIFO 控制寄存器(DMA_CHxFCTL)

x = 0...7,x为通道编号

地址偏移：0x24 + 0x18*x

复位值：0x0000 0000

该寄存器只能按字(32 位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7 | FEEIE | FIFO错误和异常中断使能位<br>软件置1与清0<br>0：FIFO错误和异常中断禁止<br>1：FIFO错误和异常中断使能 |
| 6 | 保留 | 必须保持复位值 |
| 5:3 | FCNT[2:0] | FIFO计数器<br>硬件置位和清零<br>000：FIFO非空并且数据少于1个字<br>001：FIFO数据多于1个字少于2个字<br>010：FIFO数据多于2个字少于3个字<br>011：FIFO数据多于3个字少于4个字<br>100：FIFO空<br>101：FIFO满<br>110~111：保留<br>该位域表明在数据传输过程FIFO中的数据量。若MDMEN为0，则该位域无意义。 |
| 2 | MDMEN | 多数据传输模式使能<br>软件置位与清除<br>0：关闭多数据传输模式<br>1：打开多数据传输模式<br>CHEN为1时不可写入<br>如果寄存器DMA_CHxCTL的TM位域为'10'，在通道使能后，该位由硬件强制置1. |
| 1:0 | FCCV[1:0] | FIFO计数器临界值<br>软件置位与清除<br>00：1个字 |

01：2个字

10：3个字

11：4个字

在通道使能后，该位域不可写。若MDMEN为'0'，该位域无实际意义。

# 11. 图像处理加速器(IPA)

## 11.1. 简介

IPA 提供从某一个或两个源图像到目标图像的可配置的，灵活的图像处理功能。它支持以下四种转换模式：

- 复制某一源图像到目标图像中；
- 复制某一源图像到目标图像中并同时进行特定的格式转换；
- 将两个不同的源图像进行混合，并将得到的结果进行特定的颜色格式转换；
- 用特定的颜色填充目标图像区域。

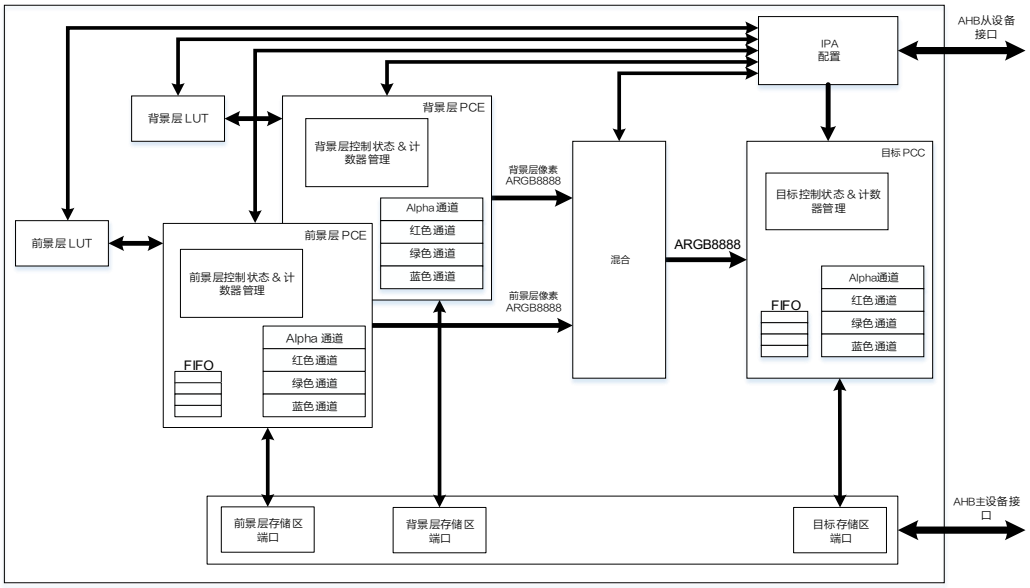
两个源图像支持 11 种像素格式，每像素从 4 位到最高 32 位，对于目标图像支持 5 种像素格式，每像素从 16 位到最高 32 位。采用间接像素模式时，IPA 为两个源图像分别提供了 256*32 的颜色查找表。

## 11.2. 主要特性

- 一个访问存储器的 AHB 主设备接口和一个支持 8 位，16 位，32 位的 IPA 配置的 AHB 从设备接口；
- 3 个 4 个字深度的 32 位 FIFO：两个源图像 FIFO，一个目标图像 FIFO；
- 支持四种像素格式转换模式：
  - 复制某一源图像到目标图像中；
  - 复制某一源图像到目标图像中并同时进行特定的颜色格式转换；
  - 将两个不同的源图像进行混合，并将得到的结果进行特定的颜色格式转换；
  - 用特定的颜色填充目标图像区域。
- 可配置 LUT 的大小；
- 支持两种 LUT 像素格式；
- 支持 LUT 自动加载；
- 支持传输挂起或停止；
- 对于源图像和目标图像，支持独立配置行偏移量；
- 支持预定义像素值；
- 支持 3 种 alpha 通道值计算算法；
- 对于两个源图像，支持 11 种像素格式；
- 对于目标图像，支持 5 种像素格式；
- 支持配置图像大小；
- 支持 AHB 总线带宽自动调节；
- 支持一个带有六种事件标志位的中断：
  - 支持中断使能和清除。

## 11.3. 结构框图

**图 11-1. IPA 模块框图**




如*图 11-1. IPA 模块框图*所示，IPA 包含 6 个主要部分：

- 通过 AHB 从设备接口配置 IPA；
- 通过 AHB 主设备接口访问图像数据；
- 前景层和背景层 LUT；
- 前景层和背景层像素通道扩展(PCE)；
- 前景层和背景层像素混合；
- 目标像素通道压缩(PCC)。

## 11.4. 功能概述

IPA 是一个像素格式转换器，它支持多种转换模式，允许用户通过配置 IPA 对应寄存器的相应位灵活的配置转换模式，前景层，背景层，目标像素格式及其行偏移。除了 LUT 仅支持 32 位访问外，其他所有 IPA 寄存器都可以通过 AHB 从设备接口进行 8 位，16 位或 32 位访问。

IPA 支持 4 种转换模式，它可以通过 IPA_CTL 寄存器的 PFCM 位配置，如*表 11-1. IPA 转换模式*所示：

- 复制前景层图像到目标图像中
  在这种模式中，前景层存储区的像素数据复制到目标存储区而不进行像素转换，所以前景层和目标图像的像素格式没有意义。前景层的像素格式仅定义了每像素的位数。
- 转换前景层图像到目标图像
  在这种模式中，前景层的像素数据从前景层的像素格式转换成目标像素格式，然后写入目标存储区中。如果前景层的像素格式是非直接的(L8, AL44, AL88, L4)，读取前景层存储区域中的数据作为索引从前景层 LUT 获取像素数据。
- 转换和混合前景层和背景层图像到目标图像

在这种模式中，前景层和背景层的像素数据首先由其原来的格式转换为'ARGB8888'，然后前景层和背景层像素数据成对的混合并从'ARGB8888'转换为目标像素格式，写入目标存储区中。

如果前景层的像素格式是非直接的，读取前景层存储区域中的数据作为索引从前景层 LUT 获取像素数据。

如果背景层的像素格式是非直接的，读取背景层存储区域中的数据作为索引从背景层 LUT 获取像素数据。

用特定颜色填充到目标图像中

在这种模式中，目标图像被特定的像素填充，该像素的值被预先定义在对应的目标寄存器中。

**表 11-1. IPA 转换模式**

| PFCM[1:0] | 转换模式 | | 像素转换 | 混合 |
|---|---|---|---|---|
| | 源 | 目的 | | |
| 00 | 前景层图像 | 目标图像 | 否 | 否 |
| 01 | 前景层图像 | 目标图像 | 是 | 否 |
| 10 | 前景层图像和背景层图像 | 目标图像 | 是 | 是 |
| 11 | 在寄存器中预定义的像素值 | 目标图像 | 否 | 否 |

### 11.4.1. 传输操作

一次 IPA 操作包含以下 7 个步骤：

1) 从前景层存储区(基地址配置在 IPA_FMADDR 寄存器中)读取数据，如果前景层像素格式是非直接的，则从前景层 LUT 获取像素数据。
2) 扩展前景层像素数据到一个 32 位的值，并根据 IPA_FPCTL 寄存器的 FAVCA 位计算 alpha 通道的值。
3) 从背景层存储区(基地址配置在 IPA_BMADDR 寄存器中)读取数据，如果背景层像素格式是非直接的，则从背景层 LUT 获取像素数据。
4) 扩展背景层像素数据到一个 32 位的值，并根据 IPA_BPCTL 寄存器的 BAVCA 位计算 alpha 通道的值。
5) 混合扩展后的前景层和背景层像素数据。
6) 压缩像素数据为 IPA_DPCTL 寄存器 DPF 位指定的目标区像素格式。
7) 将转换后的像素数据写到目标存储区(基地址配置在 IPA_DMADDR 寄存器中)。

针对前景层，背景层和目标层像素数据处理，IPA 中提供了 3 个 4 个字深度的 32 位 FIFO。前景层和背景层 FIFO 存储从相对应的源存储区读得的数据，而目标 FIFO 存储处理过的像素数据，当 AHB 总线空闲的时候，这些数据将会被写入目标存储器。

如果 IPA_CTL 寄存器的 PFCM 位域被配置成 '00' 或 '01'，只有前景层 FIFO 和目标层 FIFO 被激活。如果 IPA 的操作为用特定的颜色填充目标图像，则不需要任何一个 FIFO。

### 11.4.2. 前景层和背景层 LUT

IPA 提供了两个 LUT 来存储像素值，以便非直接像素格式使用。如果像素格式是非直接的，使能 IPA 传输之前，像素数据必须已经被写入 LUT 中。LUT 中的像素数据可以通过以下两种方

法更新：

自动加载：
使能 IPA_FPCTL/IPA_BPCTL 寄存器的 FLLEN/BLLEN 位。IPA_FPCTL 或 IPA_BPCTL 寄存器的 FCNP 或 BCNP 位定义了要自动加载的像素的数目，它等于 FCNP+1 或 BCNP+1。
软件编程：
像素数据可直接通过 IPA 从设备接口写入相应的 LUT 存储器地址。前景层 LUT 的基地址偏移是 0x0400，背景层 LUT 的基地址偏移是 0x0800。

LUT 支持两种像素格式，分别为'ARGB8888'和'RGB888'，由 IPA_FPCTL 或 IPA_BPCTL 寄存器的 FLPF 或 BLPF 位决定，如*表 11-2. 前景层和背景层 CLUT 像素格式*所示。

**表 11-2. 前景层和背景层 CLUT 像素格式**

| BLPF/FLPF | LUT 像素格式 | 存储地址 | | | |
|---|---|---|---|---|---|
| | | 基地址+0x3 | 基地址+0x2 | 基地址+0x1 | 基地址+0x0 |
| 0 | ARGB8888 | $A_0$[7:0] | $R_0$[7:0] | $G_0$[7:0] | $B_0$[7:0] |
| 1 | RGB888 | $R_3$[7:0] | $G_3$[7:0] | $B_3$[7:0] | $R_2$[7:0] |
| | | $G_2$[7:0] | $B_2$[7:0] | $R_1$[7:0] | $G_1$[7:0] |
| | | $B_1$[7:0] | $R_0$[7:0] | $G_0$[7:0] | $B_0$[7:0] |

**注意：**如果 LUT 的像素格式是'RGB888'，当自动加载 LUT 的像素数据时，alpha 值为固定的 0xFF。

### 11.4.3. 前景层和背景层像素通道扩展 (PCE)

若 IPA 传输模式需要进行像素格式转换，前景层或背景层像素数据需要由原来的格式扩展为'ARGB8888'格式。

IPA_FPCTL 和 IPA_BPCTL 寄存器的 FPF 和 BPF 位定义了前景层和背景层的像素格式。如*表 11-3. 前景层和背景层像素格式*所示。

一个像素包含以下 5 个通道：

Alpha 通道 A：透明度，0x00：透明的；0xFF：不透明的。
红色通道 R：红色值，0x00：没有红色；0xFF：全红色。
绿色通道 G：绿色值，0x00：无绿色；0xFF：全绿色。
蓝色通道 B：蓝色值，0x00：无蓝色；0xFF：全蓝色。
亮度通道：在 IPA 中，亮度通道的值是索引值，从背景层或前景层的 LUT 中获得像素数据。

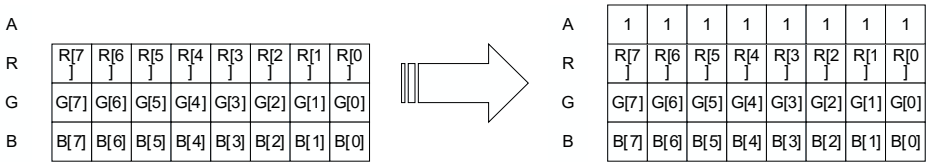
表 11-3. 前景层和背景层像素格式

| BPF[2:0]/FPF[2:0] | 像素格式 | 存储地址 | | | |
|---|---|---|---|---|---|
| | | 基地址+0x3 | 基地址+0x2 | 基地址+0x1 | 基地址+0x0 |
| 0000 | ARGB8888 | $A_0$[7:0] | $R_0$[7:0] | $G_0$[7:0] | $B_0$[7:0] |
| 0001 | RGB888 | $R_3$[7:0] | $G_3$[7:0] | $B_3$[7:0] | $R_2$[7:0] |
| | | $G_2$[7:0] | $B_2$[7:0] | $R_1$[7:0] | $G_1$[7:0] |
| | | $B_1$[7:0] | $R_0$[7:0] | $G_0$[7:0] | $B_0$[7:0] |
| 0010 | RGB565 | $R_1$[4:0]$G_1$[5:3] | $G_1$[2:0]$B_1$[4:0] | $R_0$[4:0]$G_0$[5:3] | $G_0$[2:0]$B_0$[4:0] |
| 0011 | ARGB1555 | $A_1$[0]$R_1$[4:0]$G_1$[4:3] | $G_1$[2:0]$B_1$[4:0] | $A_0$[0]$R_0$[4:0]$G_0$[4:3] | $G_0$[2:0]$B_0$[4:0] |
| 0100 | ARGB4444 | $A_1$[3:0]$R_1$[3:0] | $G_1$[3:0]$B_1$[3:0] | $A_0$[3:0]$R_0$[3:0] | $G_0$[3:0]$B_0$[3:0] |
| 0101 | L8 | $L_3$[7:0] | $L_2$[7:0] | $L_1$[7:0] | $L_0$[7:0] |
| 0110 | AL44 | $A_3$[3:0]$L_3$[3:0] | $A_2$[3:0]$L_2$[3:0] | $A_1$[3:0]$L_1$[3:0] | $A_0$[3:0]$L_0$[3:0] |
| 0111 | AL88 | $A_1$[7:0] | $L_1$[7:0] | $A_0$[7:0] | $L_0$[7:0] |
| 1000 | L4 | $L_7$[3:0]$L_6$[3:0] | $L_5$[3:0]$L_4$[3:0] | $L_3$[3:0]$L_2$[3:0] | $L_1$[3:0]$L_0$[3:0] |
| 1001 | A8 | $A_3$[7:0] | $A_2$[7:0] | $A_1$[7:0] | $A_0$[7:0] |
| 1010 | A4 | $A_7$[3:0]$A_6$[3:0] | $A_5$[3:0]$A_4$[3:0] | $A_3$[3:0]$A_2$[3:0] | $A_1$[3:0]$A_0$[3:0] |

如果像素格式是'RGB888'，当扩展像素数据时，alpha 通道值被设置为 0xFF，如*图 11-2. 从'RGB888'到'ARGB8888'像素格式扩展*所示。

图 11-2. 从'RGB888'到'ARGB8888'像素格式扩展

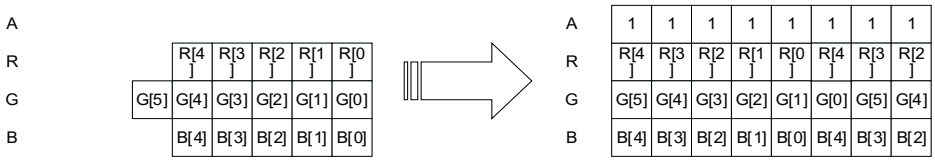
§ RGB888 à ARGB8888




如果像素格式是'RGB565'，当扩展像素数据时，alpha 通道值等于 0xFF。红绿蓝通道值扩展到 8 位，扩展后高位为通道值，通道值的高位值填充到低位。如*图 11-3. 从'RGB565'到'ARGB8888'像素格式扩展*所示。

图 11-3. 从'RGB565'到'ARGB8888'像素格式扩展

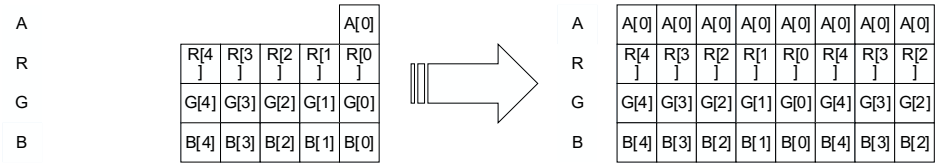
§ RGB565 à ARGB8888



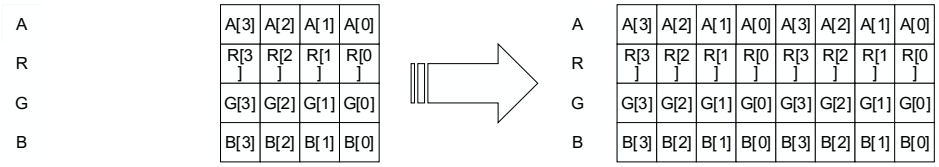

如果像素格式是'ARGB1555'或'ARGB4444'，每通道值将扩展到 8 位，扩展后高位为通道值，通道值的高位值填充到低位。如*图 11-4. 从'ARGB1555'或'ARGB4444'到'ARGB8888'像素格式扩展*所示。

**图 11-4. 从'ARGB1555'或'ARGB4444'到'ARGB8888'像素格式扩展**

§ ARGB1555 à ARGB8888




§ ARGB4444 à ARGB8888




如果像素格式是'L8'或'L4',8 位亮度通道值(当像素格式为'L4'，高位补 0)作为索引值从 LUT 获得像素数据。

如果像素格式是'AL44',8 位亮度通道值(高位补 0)作为索引值从 LUT 获得红、绿、蓝通道值。Alpha 通道值将扩展到 8 位，扩展后高位为通道值，填充通道值的高位值到低位。

如果像素格式是'AL88',只有红，绿，蓝通道值通过 8 位亮度通道从 LUT 获得。

如果像素格式是'A8',红，绿，蓝通道值分别等于 IPA_FPV 寄存器的 FPDRV，FPDGV 位以及 FPDBV 位(或 IPA_BPV 寄存器的 BPDRV 位，BPDGV 位以及 BPDBV 位)。

如果像素格式是'A4'，Alpha 通道值将扩展到 8 位，扩展后高位为通道值，填充通道的高位值到低位。红，绿，蓝通道值分别等于 IPA_FPV 寄存器的 FPDRV, FPDGV 位以及 FPDBV 位(或 IPA_BPV 寄存器的 BPDRV 位, BPDGV 位以及 BPDBV 位)。

IPA 支持通过 3 种算法调制 alpha 通道值，由 IPA_FPCTL 或 IPA_BPCTL 寄存器的 FAVCA 或 BAVCA 位决定。如*表 11-4. Alpha 通道值调制*所述。

**表 11-4. Alpha 通道值调制**

| FAVCA/BAVCA[1:0] | alpha 计算算法 |
|---|---|
| 00/11 | 无影响，等于原来的值 |
| 01 | 等于 IPA_FPCTL 或 IPA_BPCTL 寄存器 FPDAV 或 BPDAV 位的值 |
| 10 | 等于 FPDAV 或 BPDAV 位的值乘以原来 alpha 的值再除以 255 |

### 11.4.4. 混合

若 IPA 的传输模式需要进行像素混合时，扩展之后的前景层和背景层像素数据需要成对的混合并获得一个 32 位像素值。

Alpha 通道值的混合基于下面的公式($A_F$ 是前景层 alpha 值, $A_B$ 是背景层 alpha 值):

$$A_{mix} = \frac{A_F \times A_B}{255}$$

$$A_{blend} = A_F + A_B - A_{mix}$$

红，绿，蓝通道值的混合基于下面的公式($R_F$, $G_F$, $B_F$ 是前景层的红，绿，蓝值; $R_B$, $G_B$, $B_B$ 是背景层的红，绿，蓝值):

$$R_{blend} = \frac{R_F \times A_F + R_B \times A_B - R_B \times A_{mix}}{A_{blend}}$$

$$G_{blend} = \frac{G_F \times A_F + G_B \times A_B - G_B \times A_{mix}}{A_{blend}}$$

$$B_{blend} = \frac{B_F \times A_F + B_B \times A_B - B_B \times A_{mix}}{A_{blend}}$$

**注意:** 1)上述公式中的除法结果是向下取整。2)如果 $A_{blend}$ 等于 0, $R_{blend}$, $G_{blend}$ 和 $B_{blend}$ 等于 '0xFF'。

### 11.4.5. 目标像素通道压缩 (PCC)

如果在 IPA 传输模式需要进行像素转换，在像素数据写入目标存储区之前，需要由'ARGB8888'压缩为目标像素格式。

IPA_DPCTL 寄存器的 DPF 位定义了目标图像的像素格式。如*表 11-5. 目标像素格式*所示。

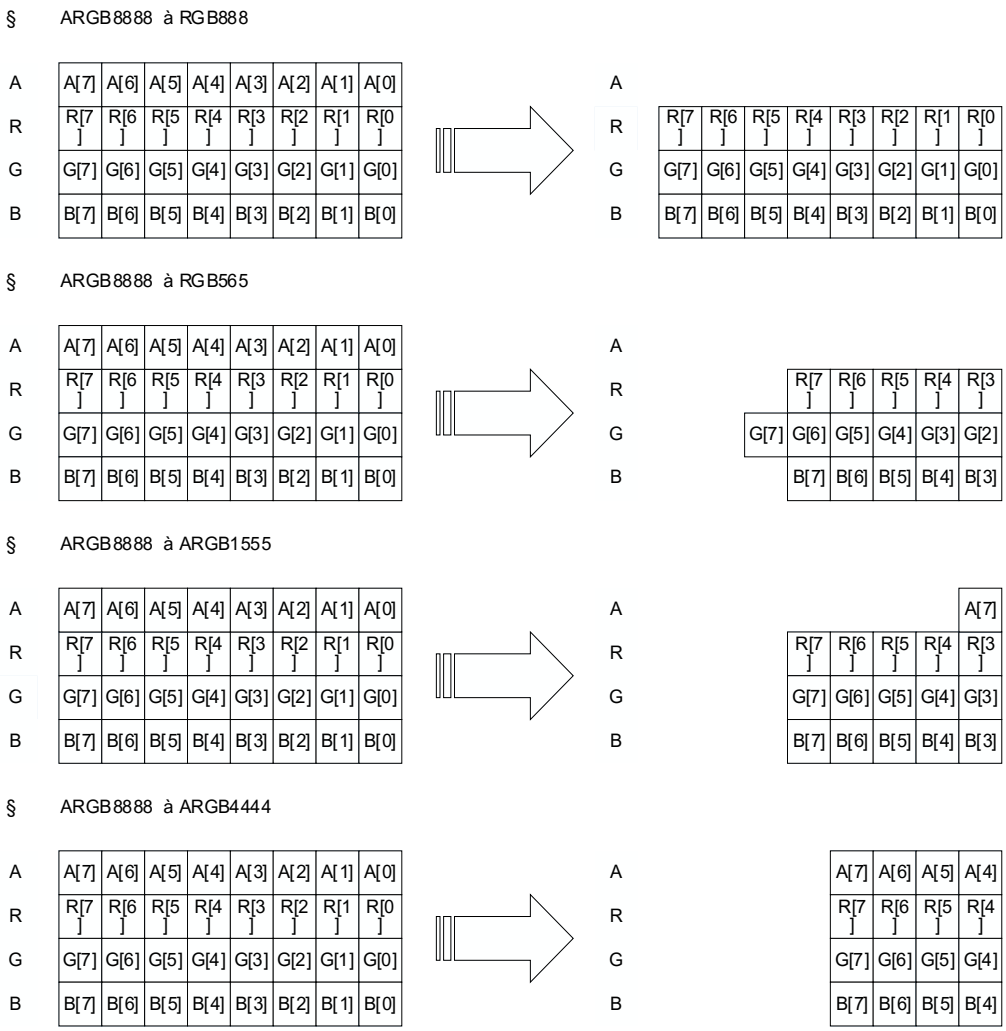
**表 11-5. 目标像素格式**

| DPF[2:0] | 像素格式 | 存储地址 | | | |
|---|---|---|---|---|---|
| | | 基地址+0x3 | 基地址+0x2 | 基地址+0x1 | 基地址+0x0 |
| 000 | ARGB8888 | $A_0$[7:0] | $R_0$[7:0] | $G_0$[7:0] | $B_0$[7:0] |
| 001 | RGB888 | $R_3$[7:0] | $G_3$[7:0] | $B_3$[7:0] | $R_2$[7:0] |
| | | $G_2$[7:0] | $B_2$[7:0] | $R_1$[7:0] | $G_1$[7:0] |
| | | $B_1$[7:0] | $R_0$[7:0] | $G_0$[7:0] | $B_0$[7:0] |
| 010 | RGB565 | $R_1$[4:0]$G_1$[5:3] | $G_1$[2:0]$B_1$[4:0] | $R_0$[4:0]$G_0$[5:3] | $G_0$[2:0]$B_0$[4:0] |
| 011 | ARGB1555 | $A_1$[0]$R_1$[4:0]$G_1$[4:3] | $G_1$[2:0]$B_1$[4:0] | $A_0$[0]$R_0$[4:0]$G_0$[4:3] | $G_0$[2:0]$B_0$[4:0] |
| 100 | ARGB4444 | $A_1$[3:0]$R_1$[3:0] | $G_1$[3:0]$B_1$[3:0] | $A_0$[3:0]$R_0$[3:0] | $G_0$[3:0]$B_0$[3:0] |

**注意:** 如果 IPA_CTL 寄存器的 PFCM 位等于'00' (拷贝前景层图像到目标图像)，DPF 位无意义，IPA_FPCTL 寄存器的 FPF 位决定了源图像和目标图像每像素的位数。

如*图 11-5. 像素压缩*所示，目标像素通道压缩通过丢弃低位实现。

图 11-5. 像素压缩



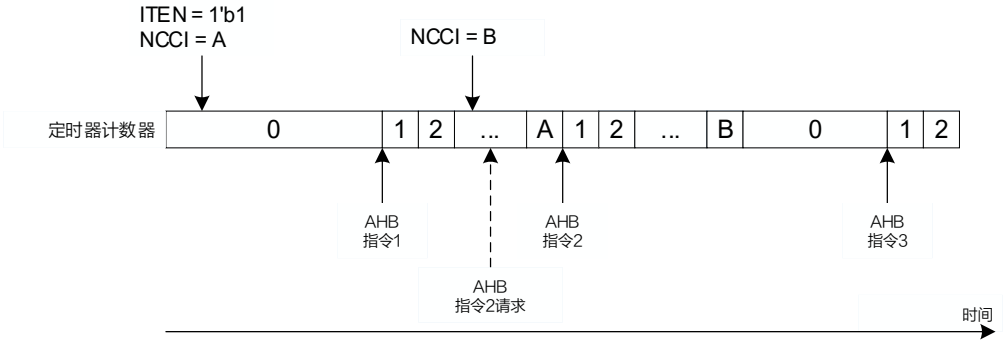

## 11.4.6. 内部定时器

为了减少 IPA 使用的 AHB 总线的带宽，在 IPA 传输与 LUT 自动加载时，IPA 会自动在两个连续的 AHB 请求之间插入若干时钟周期，这个功能通过一个内部定时器实现。

置位 IPA_ITCTL 寄存器的 ITEN 位，内部寄存器使能；IPA_ITCTL 寄存器的 NCCI 位决定了插入的时钟周期数的最小值。若内部定时器没有使能，NCCI 没有意义。

当 IPA 传输或 LUT 自动加载正在进行时，若改变 NCCI 的值，对内部计数器的当前计数没有影响，从下次计数有作用；如*图 11-6. 内部定时器操作*所示。

**图 11-6. 内部定时器操作**




## 11.4.7. 行标记

软件可通过标记行号来了解当前 IPA 传输的进度，被标记的行号可以通过 IPA_LM 寄存器的 LM 位配置。当且仅当标记行的最后一个像素数据被写入目标存储区时，IPA_INTF 寄存器中的 TLMIF 位会被置起。

**注意：**如果 LM 位等于 0，无标志位置位.

## 11.4.8. 传输流

软件置位 IPA_FPCTL/IPA_BPCTL 寄存器的 FLLEN/BLLEN 位，前景层或背景层 LUT 开始自动加载。一旦 LUT 自动加载开始，FLLEN/BLLEN 位变为传输标志位，用于指示 LUT 自动加载是否完成，且软件向其写 0 没有意义；当加载完成时，FLLEN/BLLEN 位会被自动清 0。

软件置位 IPA_CTL 寄存器的 TEN 位，IPA 开始传输。一旦传输开始，TEN 位变为传输标志位，用于指示 IPA 传输是否完成，且软件向其写 0 没有意义；当传输完成时，TEN 位会被自动清 0。

在 IPA 传输或 LUT 自动加载正在工作时，软件可通过置位 IPA_CTL 寄存器的 THU 位挂起当前传输；当软件清除 THU 位后，传输继续。若无前景层/背景层 LUT 自动加载和 IPA 传输使能时，设置 THU 位无影响，读该位的值为 0。

通过置位 IPA_CTL 寄存器的 TST 位可以停止前景层/背景层 LUT 自动加载和 IPA 传输。复位 IPA_FPCTL/IPA_BPCTL 寄存器的 FLLEN/BLLEN 位或 IPA_CTL 寄存器的 TEN 位可以让 LUT 加载或 IPA 传输立即停止，即使 IPA 传输或 LUT 自动加载正在工作或挂起。当当前的传输停止的后，TST 位自动复位。若无前景层/背景层 LUT 自动加载和 IPA 传输使能时，设置 TST 位无影响，读该位的值为 0。

前景层 LUT 自动加载，背景层 LUT 自动加载和 IPA 传输同一时间只能有一个在工作。例如，当 IPA 传输正在进行的时候，若软件置位 FLLEN 或 BLLEN 位，前景层或背景层 LUT 自动加载不会启动，且 FLLEN 或 BLLEN 位自动复位。

### 11.4.9. 配置

开始任何传输之前，软件需要读 TEN，FLLEN 和 BLLEN 位检查是否有 IPA 传输或 LUT 加载正在工作。如果有任何一个正在进行，可以设置 TST 位使其停止或等待其完成。当读取 TEN，FLLEN 和 BLLEN 的值都为 0 时，可以开始一个新的传输。

#### 前景层 LUT 加载

当开始一个新的前景层 LUT 加载的时候，建议按如下步骤进行：

1. 配置 IPA_FLMADDR 寄存器设置前景层 LUT 存储区基地址；
2. 配置 IPA_FPCTL 寄存器的 FLPF 位设置前景层 LUT 像素格式；
3. 配置 IPA_FPCTL 寄存器的 FCNP 位设置前景层 LUT 要加载的像素数目；
4. 配置 IPA_CTL 寄存器的错误配置中断，LUT 加载完成中断，LUT 访问冲突中断和传输访问错误中断使能位；
5. 配置 IPA_FPCTL 寄存器的 FLLEN 位为'1'以使能前景层 LUT 的自动加载。

#### 背景层 LUT 加载

当开始一个新的背景层 LUT 加载的时候，建议按如下步骤进行：

1. 配置 IPA_BLMADDR 寄存器设置背景层 LUT 存储区基地址；
2. 配置 IPA_BPCTL 寄存器的 BLPF 位设置背景层 LUT 像素格式；
3. 配置 IPA_BPCTL 寄存器的 BCNP 位设置背景层 LUT 要加载的像素数目；
4. 配置 IPA_CTL 寄存器的错误配置中断，LUT 加载完成中断，LUT 访问冲突中断和传输访问错误中断使能位；
5. 配置 IPA_FPCTL 寄存器的 FLLEN 位为'1'以使能背景层 LUT 的自动加载。

#### IPA 传输

当开始一个新的 IPA 传输的时候，对应不同像素转换模式的配置步骤如下所示：

#### 复制前景层图像到目标图像

1. 配置 IPA_FMADDR 和 IPA_DMADDR 寄存器设置前景层和目标层存储区基地址；
2. 配置 IPA_FPCTL 寄存器的 FPF 位设置前景层像素格式；
3. 配置 IPA_FLOFF 和 IPA_DLOFF 寄存器的 FLOFF 和 DLOFF 位设置前景层和目标层行偏移；
4. 配置 IPA_LM 寄存器的 LM 位设置行标记；
5. 配置 IPA_IMS 寄存器的 WIDTH 和 HEIGHT 位设置图像大小；
6. 配置 IPA_CTL 寄存器的错误配置中断，LUT 加载完成中断，LUT 访问冲突中断和传输访问错误中断使能位；
7. 配置 IPA_CTL 寄存器的 TEN 位为'1'以使能 IPA 传输。

### 转换前景层图像到目标图像

如果前景层像素格式是非直接的，像素数据应在开始 IPA 传输之前加载到前景层 LUT。LUT 的自动加载过程可参考*前景层 LUT 加载*。

1. 配置 IPA_FMADDR 和 IPA_DMADDR 寄存器设置前景层和目标存储区基地址；
2. 配置 IPA_FPCTL 寄存器的 FAVCA 和 FPF 位设置前景层 alpha 值的计算方法和前景层像素格式；
3. 配置 IPA_FPCTL 和 IPA_FPV 寄存器设置预定义像素值，包括 alpha，红，绿，蓝颜色值；
4. 配置 IPA_DPCTL 寄存器的 DPF 位设置目标像素格式；
5. 配置 IPA_FLOFF 和 IPA_DLOFF 寄存器的 FLOFF 和 DLOFF 位设置前景层和目标层行偏移；
6. 配置 IPA_LM 寄存器的 LM 位设置行标记；
7. 配置 IPA_IMS 寄存器的 WIDTH 和 HEIGHT 位设置图像大小；
8. 配置 IPA_CTL 寄存器的错误配置中断，LUT 加载完成中断，LUT 访问冲突中断和传输访问错误中断使能位；
9. 配置 IPA_CTL 寄存器的 TEN 位为'1'以使能 IPA 传输。

### 转换并混合前景层和背景层图像到目标图像

如果前景层和背景层像素格式是非直接的，开始 IPA 传输之前，像素数据必须被加载到对应的 LUT。前景层和背景层 LUT 的自动加载过程可参考*前景层 LUT 加载和背景层 LUT 加载*。

1. 配置 IPA_FMADDR, IPA_BMADDR 和 IPA_DMADDR 寄存器设置前景层，背景层和目标存储区基地址；
2. 配置 IPA_FPCTL 寄存器的 FAVCA 和 FPF 位设置前景层 alpha 值的计算方法和前景层像素格式；
3. 配置 IPA_FPCTL 和 IPA_FPV 寄存器设置预定义像素值，包括 alpha，红，绿，蓝颜色值；
4. 配置 IPA_BPCTL 寄存器的 BAVCA 和 BPF 位设置背景层 alpha 值的计算方法和背景层像素格式；
5. 配置 IPA_BPCTL 和 IPA_BPV 寄存器设置预定义像素值，包括 alpha，红，绿，蓝颜色值；
6. 配置 IPA_DPCTL 寄存器的 DPF 位设置目标像素格式；
7. 配置 IPA_FLOFF, IPA_BLOFF 和 IPA_DLOFF 寄存器的 FLOFF、BLOFF 和 DLOFF，设置背景层，前景层和目标的行偏移；
8. 配置 IPA_LM 寄存器的 LM 位设置行标记；
9. 配置 IPA_IMS 寄存器的 WIDTH 和 HEIGHT 位设置图像大小；
10. 配置 IPA_CTL 寄存器的错误配置中断，LUT 加载完成中断，LUT 访问冲突中断和传输访问错误中断使能位；
11. 配置 IPA_ CTL 寄存器的 TEN 位为'1'以使能 IPA 传输。

### 用特定的颜色填充目标图像

1. 配置 IPA_DMADDR 寄存器设置目标存储区的基地址；
2. 配置 IPA_DPCTL 寄存器的 DPF 设置目标像素格式；
3. 配置 IPA_DPV 寄存器设置目标区预定义像素值，包括 alpha，红，绿，蓝颜色值；

4. 配置 IPA_DLOFF 寄存器的 DLOFF 设置目标的行偏移；
5. 配置 IPA_LM 寄存器的 LM 位设置行标记；
6. 配置 IPA_IMS 寄存器的 WIDTH 和 HEIGHT 位设置图像大小；
7. 配置 IPA_CTL 寄存器的错误配置中断，LUT 加载完成中断，LUT 访问冲突中断和传输访问错误中断使能位；
8. 配置 IPA_CTL 寄存器的 TEN 位为'1'以使能 IPA 传输。

### 配置规则

IPA 配置必须遵守一些规则，否则在它使能之后，传输或加载将会自动复位，IPA_INTF 寄存器的 WCFIF 位将会立即置位。规则描述如下：

当前景层 LUT 自动加载使能：

- 当 IPA_FPCTL 寄存器的 FLPF 位等于 0 时，寄存器 IPA_FLMADDR 的 FLMADDR 位必须是 32 位对齐的。

当背景层 LUT 自动加载使能：

- 当 IPA_BPCTL 寄存器的 BLPF 位等于 0 时，寄存器 IPA_BLMADDR 的 BLMADDR 位必须是 32 位对齐的。

当 IPA 传输使能：

1) 当 IPA_FPCTL 寄存器的 FPF 位是 'ARGB8888'时，IPA_FMADDR 寄存器的 FMADDR 位必须是 32 位对齐。当 FPF 是 'RGB565', 'ARGB1555', 'ARGB4444' 或 'AL88'，必须是 16 位对齐。
2) 当 IPA_FPCTL 寄存器的 FPF 位是 A4 或 L4 时，IPA_FLOFF 寄存器的 FLOFF 位必须是偶数。
3) 当 IPA_BPCTL 寄存器的 BPF 是 'ARGB8888'时，IPA_BMADDR 寄存器的 BMADDR 位必须是 32 位对齐。当 BPF 是 'RGB565', 'ARGB1555', 'ARGB4444' 或 'AL88'，必须是 16 位对齐。
4) 当 IPA_BPCTL 寄存器的 BPF 位是 A4 或 L4 时，IPA_BLOFF 寄存器的 BLOFF 位必须是偶数。
5) IPA_FPCTL 寄存器的 FPF 位必须小于或等于'0b1010'。
6) IPA_BPCTL 寄存器的 BPF 位必须小于或等于'0b1010'。
7) IPA_DPCTL 寄存器的 DPF 位必须小于或等于'0b100'。
8) 当 IPA_DPCTL 寄存器的 DPF 是 'ARGB8888'时，IPA_ DMADDR 寄存器的 DMADDR 位必须是 32 位对齐。当 DPF 是 'RGB565', 'ARGB1555', 'ARGB4444' 或 'AL88'，必须是 16 位对齐。
9) 当 IPA_FPCTL 寄存器的 FPF 位是 A4 或 L4 时，IPA_DLOFF 寄存器的 DLOFF 位必须是偶数。
10) 当 IPA_FPCTL 寄存器的 FPF 位是 A4 或 L4 时，IPA_IMS 寄存器的 WIDTH 位必须是偶数。
11) 当 IPA_BPCTL 寄存器的 BPF 位是 A4 或 L4 时，IPA_IMS 寄存器的 WIDTH 位必须是偶数。
12) IPA_IMS 寄存器的 WIDTH 位必须大于 0。
13) IPA_IMS 寄存器的 HEIGHT 位必须大于 0。

当 PFCM 位等于'00'，仅考虑 1)、2)、5)、9)、10)、12)、13)。
当 PFCM 位等于'01'，仅考虑 1)、2)、5)、7)、8)、10)、12)、13)。
当 PFCM 位等于'10'，除了 10) 所有的规则都考虑。
当 PFCM 位等于'11'，仅考虑 12)、13)。

## 11.5. 中断

有 6 个中断事件连接到 IPA 中断，包括错误配置中断，LUT 加载完成中断，LUT 访问冲突中断，传输行标记中断，传输完成中断和传输访问错误中断。任何一个中断事件发生都将产生一个 IPA 中断。

每一个中断事件在 IPA_INTF 寄存器有一个专用的状态位，在 IPA_INTC 寄存器有一个专门的清除位，在 IPA_CTL 寄存器有一个专用的使能位。相应位之间的关系如*表 11-6. IPA 中断事件*所述。

**表 11-6. IPA 中断事件**

| 中断事件 | 状态位 | 使能位 | 清除位 |
|---|---|---|---|
| | IPA_INTF | IPA_CTL | IPA_INTC |
| 配置错误中断 | WCFIF | WCFIE | WCFIFC |
| LUT加载完成中断 | LLFIF | LLFIE | LLFIFC |
| LUT访问冲突中断 | LACIF | LACIE | LACIFC |
| 传输行标记中断 | TLMIF | TLMIE | TLMIFC |
| 全传输完成中断 | FTFIF | FTFIE | FTFIFC |
| 传输访问错误中断 | TAEIF | TAEIE | TAEIFC |

### 配置错误中断

当 LUT 加载或 IPA 传输被使能之后，若*配置规则*小节列出的任何一个配置规则被破坏，配置错误中断状态位将立即置位。LUT 加载或 IPA 传输将自动停止且不产生任何访问操作。

当配置错误中断状态位被置位并且相应使能位使能时，将产生一个 IPA 中断。

### LUT 加载完成中断

当最后一个像素数据被写入到前景层或背景层 LUT 后，LUT 加载完成中断状态状态位将立即置位，在加载期间的停止操作不能置位 LUT 加载完成中断状态位。

当 LUT 加载完成中断状态位被置位并且相应使能位使能，将产生一个 IPA 中断。

### LUT 访问冲突中断

当通过软件访问前景层和背景层 LUT 时，必须遵守以下规则：

- 当前景层 LUT 自动加载时，禁止软件访问前景层 LUT；
- 当背景层 LUT 自动加载时，禁止软件访问背景层 LUT；
- 当配置 IPA 传输模式 PFCM 位等于 '01' 或 '10'时，如果前景层像素格式是非直接的，在

IPA 传输正在进行时，禁止软件访问前景层 LUT；

- 当配置 IPA 传输模式 PFCM 位等于 '10'时，如果背景层像素格式是非直接的，在 IPA 传输正在进行时，禁止软件访问背景层 LUT。

当违背上述规则之一时，LUT 访问冲突中断状态位置位，且软件访问无作用(写访问不被执行，读访问返回一个无效值)。

当 LUT 访问冲突中断状态位寄存器被置位并且相应使能位使能时，将产生一个 IPA 中断。

### 传输行标记中断

当标记行的最后一个像素数据被写入目标存储区后，传输行标记中断状态位立即置位。如果 IPA_LM 的 LM 位等于 0，在 IPA 传输期间，传输行标记中断状态位不置位。

当传输行标记中断状态位被置位并且相应使能位使能时，将产生一个 IPA 中断。

### 传输完成中断

当最后一个像素数据被写入目标存储区后，传输完成中断状态位立即置位。在 IPA 传输期间的停止操作不置位传输完成中断状态位。

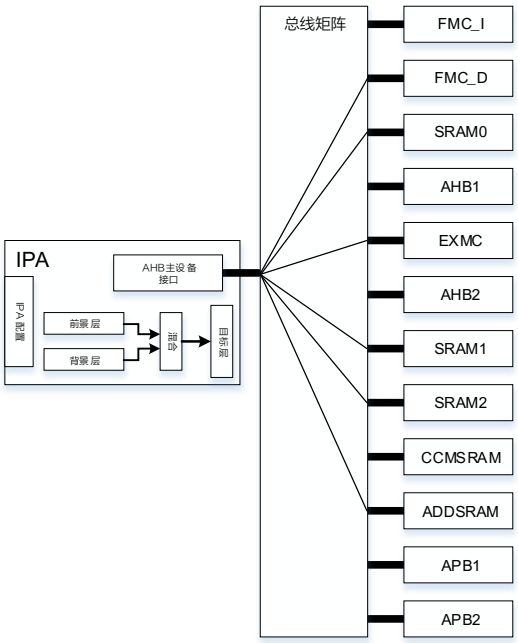
当传输完成中断状态位被置位并且相应使能位使能时，将产生一个 IPA 中断。

### 传输访问错误中断

当 IPA 访问的地址超出允许的地址，IPA 将收到一个错误反馈；传输(LUT 加载或 IPA 传输)将被立即禁止并且没有 LUT 加载完成中断状态位或传输完成中断状态位置位。IPA 允许和禁止的访问区域在*图 11-7. IPA 的系统连接*中列出。

当传输访问错误中断状态位被置位并且相应使能位使能时，将产生一个 IPA 中断。

**图 11-7. IPA 的系统连接**



## 11.6. IPA 寄存器

IPA基地址：0x4002 B000

### 11.6.1. 控制寄存器 (IPA_CTL)

偏移地址：0x00
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | WCFIE | LLFIE | LACIE | TLMIE | FTFIE | TAEIE | 保留 | | | | | TST | THU | TEN |
| | | rw | rw | rw | rw | rw | rw | | | | | | rs | rw | rs |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:18 | 保留 | 必须保持复位值 |
| 17:16 | PFCM[1:0] | 像素格式转换模式<br>软件置位和清除.<br>00：前景层存储区到目标存储区无像素格式转换<br>01：前景层存储区到目标存储区有像素格式转换<br>10：混合前景层和背景层存储区到目标存储区<br>11：用特定的颜色填充目标存储区<br>当TEN为'1'时，该位不可写。 |
| 15：14 | 保留 | 必须保持复位值 |
| 13 | WCFIE | 配置错误中断使能位<br>软件置位和清除<br>0：配置错误中断禁止<br>1：配置错误中断使能 |
| 12 | LLFIE | LUT加载完成中断使能位<br>软件置位和清除<br>0：LUT加载完成中断禁止<br>1：LUT加载完成中断使能 |
| 11 | LACIE | LUT访问冲突中断使能位<br>软件置位和清除<br>0：LUT访问冲突中断禁止<br>1：LUT访问冲突中断使能 |
| 10 | TLMIE | 传输行标记中断使能位<br>软件置位和清除 |

0：传输行标记中断禁止
1：传输行标记中断使能

| | | |
|---|---|---|
| 9 | FTFIE | 传输完成中断使能位<br>软件置位和清除<br>0：传输完成中断禁止<br>1：传输完成中断使能 |
| 8 | TAEIE | 传输访问错误中断使能位<br>软件置位和清除<br>0：传输访问错误中断禁止<br>1：传输访问错误中断使能 |
| 7:3 | 保留 | 必须保持复位值 |
| 2 | TST | 传输停止<br>软件置位，软件和硬件清除<br>0：无影响<br>1：停止当前的传输<br>当该位使能后，当前传输停止。当当前传输停止后，该位立即被硬件自动清0。 |
| 1 | THU | 传输挂起<br>软件置位，软件和硬件清除.<br>0：无影响<br>1：挂起当前传输<br>当该位使能后，当前传输暂停。当该位清0后，当前传输继续。当当前传输被停止时，该位立即被硬件自动清0。 |
| 0 | TEN | 传输使能<br>软件置位，硬件清除<br>0：传输禁止<br>1：传输使能<br>当该位使能后，IPA传输开始。当下述情况之一发生时，该位自动清0。<br>– 使能TST位停止当前传输<br>– 传输完全完成<br>– 检测到配置错误或传输访问错误<br>– 前景层LUT或背景层LUT正在自动加载(IPA_FPCTL寄存器的FLLEN或IPA_BPCTL寄存器的BLLEN位为'1') |

### 11.6.2. 中断状态寄存器 (IPA_INTF)

偏移地址：0x04
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | WCFIF | LLFIF | LACIF | TLMIF | FTFIF | TAEIF |
| | | | | | | | | | | r | r | r | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:6 | 保留 | 必须保持复位值 |
| 5 | WCFIF | 配置错误中断标志<br>硬件置位，软件置位IPA_INTC 寄存器的'WCFIFC'位清除该位。<br>0：当IPA传输完成或LUT加载使能时，没检测到配置错误<br>1：当IPA传输完成或LUT加载使能时，检测到配置错误 |
| 4 | LLFIF | LUT加载完成中断标志LUT<br>硬件置位，软件置位IPA_INTC 寄存器的'LLFIFC'位清除该位。<br>0：没检测到LUT加载完成<br>1：检测到一个LUT加载完成 |
| 3 | LACIF | LUT访问冲突中断标志位<br>硬件置位，软件置位IPA_INTC 寄存器的'LACIFC'位清除该位。<br>0：没检测到LUT访问冲突<br>1：检测到一个LUT访问冲突 |
| 2 | TLMIF | 传输行标记中断标志<br>硬件置位，软件置位IPA_INTC 寄存器的'CTCLIF'位清除该位。<br>0：传输的像素数目，没有准确的达到标记行<br>1：传输的像素数目，准确的达到标记行 |
| 1 | FTFIF | 传输完成中断标志<br>硬件置位，软件置位IPA_INTC 寄存器的'CTFIF'位清除该位。<br>0：没检测到传输完成<br>1：检测到传输完成 |
| 0 | TAEIF | 传输访问错误中断标志<br>硬件置位，软件置位IPA_INTC 寄存器的'CTEIF'位清除该位。<br>0：没检测到传输访问错误<br>1：检测到传输访问错误 |

### 11.6.3. 中断标志清除寄存器 (IPA_INTC)

偏移地址：0x08
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | WCFIFC | LLFIFC | LACIFC | TLMIFC | FTFIFC | TAEIFC |
| | | | | | | | | | | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:6 | 保留 | 必须保持复位值 |
| 5 | WCFIFC | 配置错误中断标志清除位<br>软件置位，硬件清除<br>0：无影响<br>1：清除配置错误中断标志 |
| 4 | LLFIFC | LUT加载完成中断标志清除位<br>软件置位，硬件清除<br>0：无影响<br>1：清除LUT加载完成中断标志 |
| 3 | LACIFC | LUT访问冲突中断标志清除位<br>软件置位，硬件清除<br>0：无影响<br>1：清除LUT访问冲突中断标志 |
| 2 | TLMIFC | 传输行标记中断标志清除位<br>软件置位，硬件清除<br>0：无影响<br>1：清除传输行标记中断标志 |
| 1 | TFIFC | 传输完成中断标志清除位<br>软件置位，硬件清除<br>0：无影响<br>1：清除传输完成中断标志 |
| 0 | TAEIFC | 传输访问错误中断标志清除位<br>软件置位，硬件清除<br>0：无影响<br>1：清除传输访问错误中断标志 |

### 11.6.4. 前景层存储区基地址寄存器 (IPA_FMADDR)

偏移地址：0x0C
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| FMADDR[15:0] |
|---|

rw

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | FMADDR[31:0] | 前景层存储区基地址<br>这些位必须是8位，16位，32位对齐，具体对齐方式与前景层像素格式相对应。如果前景层像素格式是ARGB8888，这些位必须是32位对齐，如果前景层像素格式是RGB565, ARGB1555, ARGB4444或AL88,这些位必须是16位对齐，如果违背以上对齐规则，当传输使能时，将产生配置错误。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |

### 11.6.5. 前景层行偏移寄存器 (IPA_FLOFF)

偏移地址：0x10
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

rw

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值 |
| 13:0 | FLOFF[13:0] | 前景层行偏移<br>该位表明当前行最后一个像素和下一行第一个像素之间的像素数目。如果前景层像素格式是A4或L4，FLOFF 必须被配置成一个偶数，否则当传输使能的时候将检测到一个配置错误。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |

### 11.6.6. 背景层存储区基地址寄存器 (IPA_BMADDR)

偏移地址：0x14
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

rw

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

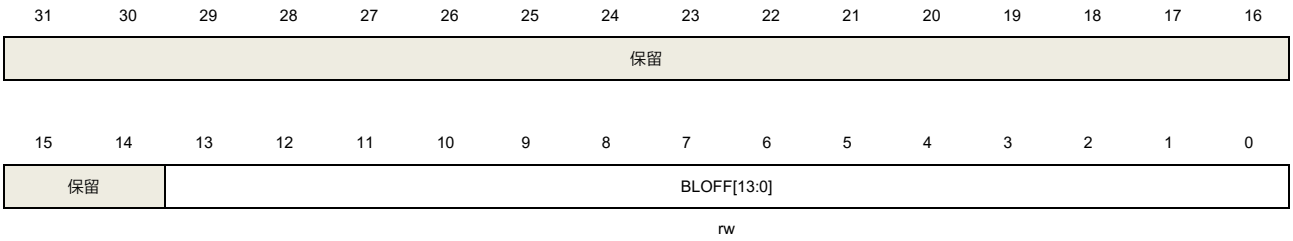
rw

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | BMADDR[31:0] | 背景层存储区基地址<br>这些位必须是8位，16位，32位对齐，具体对齐方式与背景层像素格式相对应。如果背景层像素格式是ARGB8888，这些位必须是32位对齐，如果背景层像素格式是RGB565, ARGB1555, ARGB4444或AL88,这些位必须是16位对齐，如果违背以上对齐规则，当传输使能时，将产生配置错误。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |

## 11.6.7. 背景层行偏移寄存器 (IPA_BLOFF)

偏移地址：0x18
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



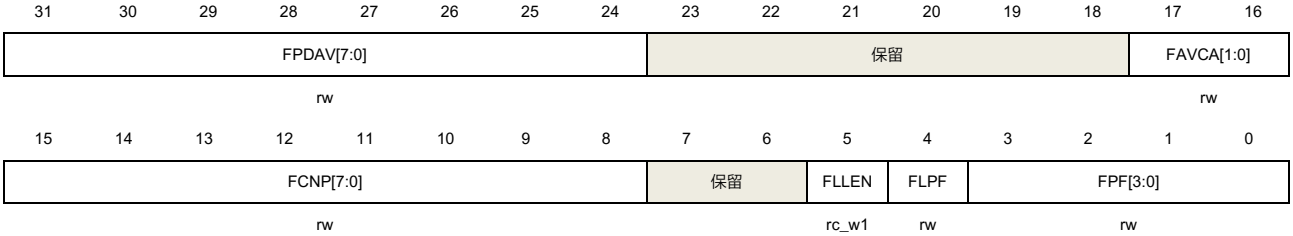

rw

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值 |
| 13:0 | BLOFF[13:0] | 背景层行偏移<br>该位表明当前行最后一个像素和下一行第一个像素之间的像素数目。如果背景层像素格式是A4或L4，BLOFF 必须被配置成一个偶数，否则当传输使能的时候将检测到一个配置错误。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |

## 11.6.8. 前景层像素控制寄存器 (IPA_FPCTL)

偏移地址：0x1C
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




rw rw
rw rc_w1 rw rw

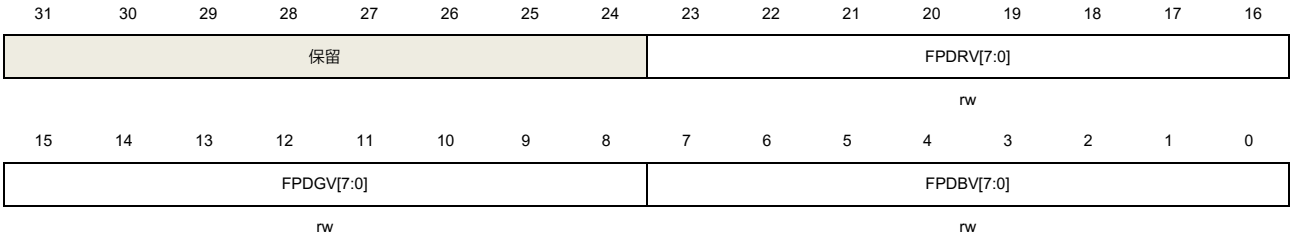
| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:24 | FPDAV[7:0] | 前景层预定义alpha通道值<br>软件置位和清除<br>该位域预定义前景层的alpha通道值。该位域结合从前景层存储区或前景层LUT读取的alpha数据根据前景层alpha计算算法计算前景层的alpha通道值。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |
| 23:18 | 保留 | 必须保持复位值 |
| 17:16 | FAVCA[1:0] | 前景层alpha值计算算法<br>软件置位和清除<br>00：无影响<br>01：FPDAV[7:0] 被选作前景层alpha通道值<br>10：FPDAV[7:0] 乘以从前景层存储区或前景层LUT读取的alpha数据除以255作为前景层alpha通道值。<br>11：保留<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |
| 15:8 | FCNP[7:0] | 前景层 LUT 像素数目<br>软件置位和清除<br>前景层LUT的像素数目等于FCNP + 1.<br>当FLLEN为'1'时，该位不可写。 |
| 7:6 | 保留 | 必须保持复位值 |
| 5 | FLLEN | 前景层 LUT加载使能<br>软件置位，硬件清除<br>0：禁止前景层 LUT 加载<br>1：使能前景层 LUT 加载<br>当该位使能，前景层LUT自动加载开始，当下述情况之一发生时，该位自动清0<br>– 使能TST位停止当前传输<br>– 前景层LUT自动加载完成<br>– 检测到配置错误或传输错误<br>– IPA传输或背景层LUT自动加载正在进行 |
| 4 | FLPF | 前景层LUT像素格式<br>软件置位和清除<br>0：ARGB8888<br>1：RGB888<br>当FLLEN为'1'时，该位不可写。 |
| 3:0 | FPF[3:0] | 前景层像素格式<br>软件置位和清除<br>0000：ARGB8888<br>0001：RGB888<br>0010：RGB565<br>0011：ARGB1555 |

0100：ARGB4444
0101：L8
0110：AL44
0111：AL88
1000：L4
1001：A8
1010：A4
1011 ~ 1111：保留
当IPA_CTL寄存器的TEN位为'1'时，该位不可写。

### 11.6.9. 前景层像素值寄存器 (IPA_FPV)

偏移地址：0x20
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

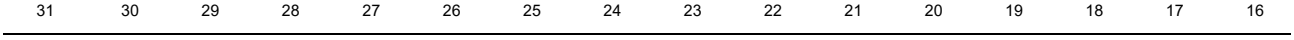


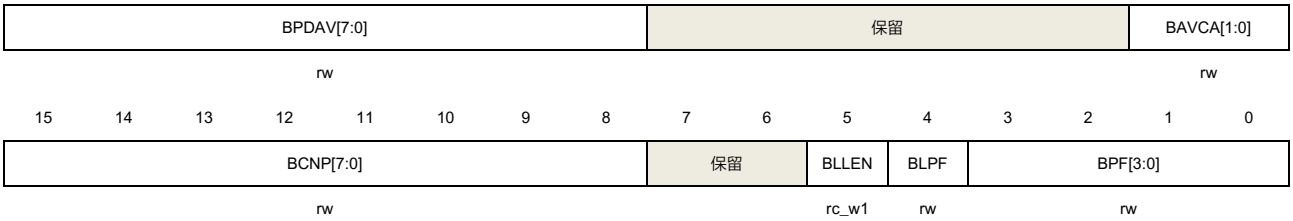

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值 |
| 23:16 | FPDRV[7:0] | 前景层预定义红色值<br>当前景层像素格式是A4或A8时，该位域被用作前景层红色值。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |
| 15:8 | FPDGV[7:0] | 前景层预定义绿色值<br>当前景层像素格式是A4或A8时，该位域被用作前景层绿色值。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |
| 7:0 | FPDBV[7:0] | 前景层预定义蓝色值<br>当前景层像素格式是A4或A8时，该位域被用作前景层蓝色值。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |

### 11.6.10. 背景层像素控制寄存器 (IPA_BPCTL)

偏移地址：0x24
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。






| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | BPDAV[7:0] | 背景层预定义alpha通道值<br>软件置位和清除<br>该位域预定义背景层的alpha通道值。该位域结合从背景层存储区和背景层LUT读取的alpha数据根据背景层alpha计算算法计算背景层alpha通道值。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |
| 23:18 | 保留 | 必须保持复位值 |
| 17:16 | BAVCA[1:0] | 背景层alpha值计算算法<br>软件置位和清除<br>00：无影响<br>01：BPDAV [7:0] 被选作背景层alpha值<br>10：BPDAV [7:0] 乘以从背景层存储区或背景层LUT读取的alpha数据除以255作为背景层alpha值。<br>11：保留<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |
| 15:8 | BCNP[7:0] | 背景层LUT像素数目<br>软件置位和清除<br>背景层LUT的像素数目等于BCNP + 1.<br>当BLLEN为'1'时，该位不可写。 |
| 7:6 | 保留 | 必须保持复位值 |
| 5 | BLLEN | 背景层LUT加载使能<br>软件置位，硬件清除.<br>0：禁止背景层LUT加载<br>1：使能背景层LUT加载<br>当该位使能，背景层LUT加载开始，当下述情况之一发生时，该位自动清0。<br>– 使能TST停止当前传输<br>– 背景层 LUT加载完成<br>– 检测到配置错误或传输错误<br>IPA传输或背景层LUT自动加载正在进行 |
| 4 | BLPF | 背景层LUT像素格式<br>软件置位和清除<br>0：ARGB8888<br>1：RGB888<br>当BLLEN为'1'时，该位不可写。 |

| | | |
|---|---|---|
| 3:0 | BPF[3:0] | 背景层像素格式<br>软件置位和清除<br>0000：ARGB8888<br>0001：RGB888<br>0010：RGB565<br>0011：ARGB1555<br>0100：ARGB4444<br>0101：L8<br>0110：AL44<br>0111：AL88<br>1000：L4<br>1001：A8<br>1010：A4<br>1011 ~ 1111：保留<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |

### 11.6.11. 背景层像素值寄存器 (IPA_BPV)

偏移地址：0x28
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | BPDRV[7:0] | | | | | | | |
| | | | | | | | | rw | | | | | | | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值 |
| 23:16 | BPDRV[7:0] | 背景层预定义红色值<br>当背景层像素格式是A4或A8时，该位域被用作背景层红色值。<br>当IPA_CTL寄存器的TEN为'1'时，该位不可写。 |
| 15:8 | BPDGV[7:0] | 背景层预定义绿色值<br>当背景层像素格式是A4或A8时，该位域被用作背景层绿色值。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |
| 7:0 | BPDBV[7:0] | 背景层预定义蓝色值<br>当背景层像素格式是A4或A8时，该位域被用作背景层蓝色值。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |

### 11.6.12. 前景层 LUT 存储区基地址寄存器 (IPA_FLMADDR)

偏移地址：0x2C
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | FLMADDR[31:0] | 前景层 LUT 存储区基地址<br>软件置位和清除<br>这些位必须是8位，16位，32位对齐，具体对齐方式与前景层LUT像素格式相对应。如果前景层LUT像素格式是ARGB8888，这些位必须是32位对齐。如果违背以上对齐规则，当前景层LUT加载使能时，将产生配置错误。<br>当IPA_FPCTL 寄存器的FLLEN 位为'1'的时候，该位域不可写。 |

### 11.6.13. 背景层 LUT 存储区基地址寄存器 (IPA_BLMADDR)

偏移地址：0x30
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | BLMADDR[31:0] | 背景层 LUT 存储区基地址<br>软件置位和清除<br>这些位必须是8位，16位，32位对齐，具体对齐方式与背景层LUT像素格式相对应。如果背景层LUT像素格式是ARGB8888，这些位必须是32位对齐。如果违背以上对齐规则，当背景层LUT加载使能时，将产生配置错误。<br>当IPA_BPCTL寄存器的BLLEN 位为'1'的时候，该位域不可写。 |

### 11.6.14. 目标像素控制寄存器 (IPA_DPCTL)

偏移地址：0x34
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:3 | 保留 | 必须保持复位值 |
| 2:0 | DPF[2:0] | 目标像素格式<br>软件置位和清除<br>000：ARGB8888<br>001：RGB888<br>010：RGB565<br>011：ARGB1555<br>100：ARGB4444<br>101~111：保留<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |

### 11.6.15. 目标像素值寄存器 (IPA_DPV)

偏移地址：0x38
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DPDAV[7:0] | | | | | | | | DPDRV[7:0] | | | | | | | |
| MEANINGLESS | | | | | | | | DPDRV[7:0] | | | | | | | |
| MEANINGLESS | | | | | | | | | | | | | | | |
| MEANINGLESS | | | | | | | | | | | | | | | |
| MEANINGLESS | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DPDGV[7:0] | | | | | | | | DPDBV[7:0] | | | | | | | |
| DPDGV[7:0] | | | | | | | | DPDBV[7:0] | | | | | | | |
| DPDRV[4:0] | | | | | | DPDGV[5:0] | | | | | | DPDBV[4:0] | | | |
| DPDAV | DPDRV[4:0] | | | | | DPDGV[4:0] | | | | | | DPDBV[4:0] | | | |

| DPDAV[3:0] | DPDRV[3:0] | DPDGV[3:0] | DPDBV[3:0] |
|---|---|---|---|
| rw | rw | rw | rw |

**当目标像素格式是 ARGB8888 时，第一行有效。**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | DPDAV[7:0] | 目标层预定义alpha值<br>软件置位和清除<br>当IPA配置为用特定的颜色填充目标存储区的时候，这些位用作目标层alpha值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 23:16 | DPDRV[7:0] | 目标层预定义红色值<br>软件置位和清除<br>当IPA配置为用特定的颜色填充目标存储区的时候，这些位用作目标层红色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 15:8 | DPDGV[7:0] | 目标层预定义绿色值<br>软件置位和清除<br>当IPA配置为用特定的颜色填充目标存储区的时候，这些位用作目标层绿色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 7:0 | DPDBV[7:0] | 目标层预定义蓝色值<br>软件置位和清除<br>当IPA配置为用特定的颜色填充目标存储区的时候，这些位用作目标层蓝色值。<br>当IPA_CTL寄存器的TEN 的值为'1'时，该位不可写。 |

**当目标像素格式是 RGB888 时，第 2 行有效。**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | Meaningless | 该位域可以软件置位和清除，但当目标像素格式是RGB888时，这些位没有意义。 |
| 23:16 | DPDRV[7:0] | 目标层预定义红色值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层红色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 15:8 | DPDGV[7:0] | 目标层预定义绿色值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层绿色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 7:0 | DPDBV[7:0] | 目标层预定义蓝色值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层蓝色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |

**当目标像素格式是 RGB565 时，第 3 行有效。**

| 位/位域 | 名称 | 描述 |
|---|---|---|

| | | |
|---|---|---|
| 31:16 | Meaningless | 该位域可以软件置位和清除，但当目标像素格式是RGB565时，这些位没有意义。 |
| 15:11 | DPDRV[4:0] | 目标层预定义红色值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层红色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 10:5 | DPDGV[5:0] | 目标层预定义绿色值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层绿色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 4:0 | DPDBV[4:0] | 目标层预定义蓝色值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层蓝色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |

**当目标像素格式是 ARGB1555 时，第 4 行有效。**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | Meaningless | 该位域可以软件置位和清除，但当目标像素格式是ARGB1555时，这些位没有意义。 |
| 15 | DPDAV | 目标层预定义alpha值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层alpha值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 14:10 | DPDRV[4:0] | 目标层预定义红色值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层红色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 9:5 | DPDGV[4:0] | 目标层预定义绿色值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层绿色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 4:0 | DPDBV[4:0] | 目标层预定义蓝色值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层蓝色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |

**当目标像素格式是 ARGB4444 时，第 5 行有效。**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | Meaningless | 该位域可以软件置位和清除，但当目标像素格式是ARGB4444时，这些位没有意义。 |
| 15:12 | DPDAV[3:0] | 目标层预定义alpha值<br>软件置位和清除 |

| | | |
|---|---|---|
| | | 当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层alpha值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 11:8 | DPDRV[3:0] | 目标层预定义红色值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层红色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 7:4 | DPDGV[3:0] | 目标层预定义绿色值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层绿色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |
| 3:0 | DPDBV[3:0] | 目标层预定义蓝色值<br>软件置位和清除<br>当 IPA 配置为用特定的颜色填充目标存储区的时候，这些位用作目标层蓝色值。<br>当IPA_CTL寄存器的TEN位的值为'1'时，该位不可写。 |

### 11.6.16. 目标存储区基地址寄存器 (IPA_DMADDR)

偏移地址：0x3C
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

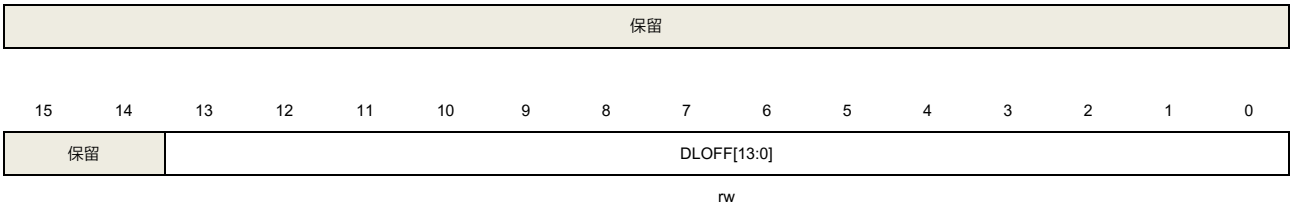
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | DMADDR[31:0] | 目标存储区基地址Destination memory base address<br>软件置位和清除<br>如果目标层像素格式是ARGB8888，这些位必须是32位对齐，如果目标层像素格式是RGB565，ARGB1555或ARGB4444,这些位必须是16位对齐，如果违背以上对齐规则，当传输使能的时候，将检测到一个配置错误。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |

### 11.6.17. 目标行偏移寄存器 (IPA_DLOFF)

偏移地址：0x40
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

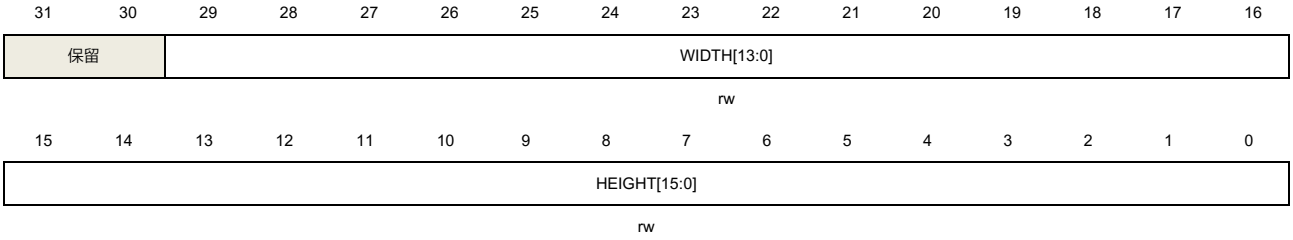



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值 |
| 13:0 | DLOFF[13:0] | 目标行偏移<br>该位表明当前行最后一个像素和下一行第一个像素之间的像素数目。当IPA_CTL寄存器的PFCM配置为"00"时，如果前景层像素格式是A4或L4，DLOFF 必须被配置成一个偶数，否则当传输使能的时候将检测到一个配置错误。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |

### 11.6.18. 图像大小寄存器 (IPA_IMS)

偏移地址：0x44
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



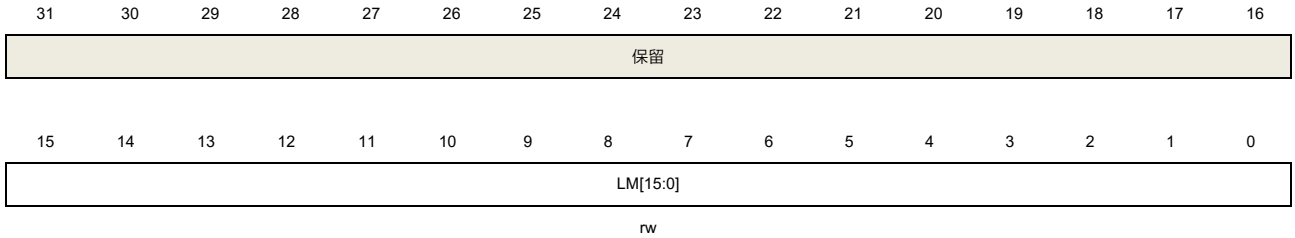

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:30 | 保留 | 必须保持复位值 |
| 29:16 | WIDTH[13:0] | 待处理的图像的宽度<br>软件置位和清除<br>该位域表示待处理的图像每行像素的数目。如果背景层或前景层像素格式是A4或L4，这些位必须配置成偶数，否则当传输使能的时候将检测到一个配置错误。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |
| 15:0 | HEIGHT[15:0] | 待处理图像的高度<br>软件置位和清除<br>该位域表明待处理图像的行数。<br>当IPA_CTL寄存器的TEN为'1'时，该位不可写。 |

### 11.6.19. 行标记寄存器 (IPA_LM)

偏移地址：0x48

复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



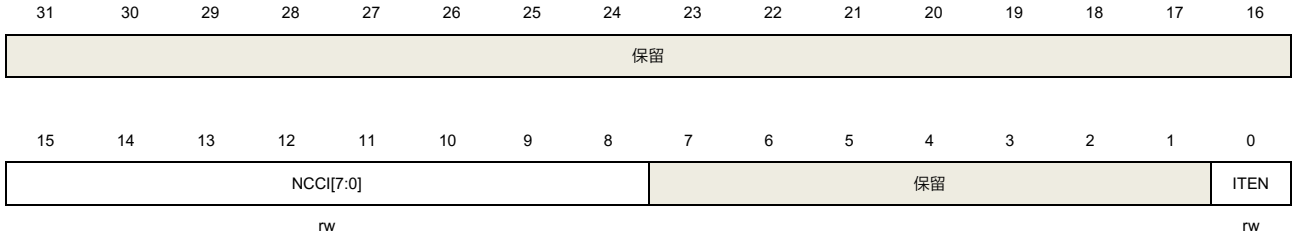

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15:0 | LM[15:0] | 行标记<br>软件置位和清除<br>该位域定义了一个行号以表明传输的进度，当且仅当标记行的最后一个像素已经写入了目标存储区，传输行标记中断标志位将置位。<br>当IPA_CTL寄存器的TEN位为'1'时，该位不可写。 |

## 11.6.20. 内部定时器控制寄存器 (IPA_ITCTL)

偏移地址：0x4C

复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15:8 | NCCI[7:0] | 间隔时钟周期数<br>软件置位和清除<br>如果ITEN等于'0'，该位域没有意义。如果ITEN 等于'1'，该位域表示两个连续的AHB请求之间插入的时钟周期数的最小值。 |
| 7:1 | 保留 | 必须保持复位值 |
| 0 | ITEN | 内部定时器使能<br>IPA使用一个内部定时器用来减少AHB总线使用带宽。<br>0：禁止内部定时器 |

1：使能内部定时器

# 12. 调试(DBG)

## 12.1. 简介

GD32A490xx 系列产品提供了各种各样的调试，跟踪和测试功能。这些功能通过 ARM® CoreSight™组件的标准配置和链状连接的 TAP 控制器来实现的。调试和跟踪功能集成在 ARM® Cortex®-M4 内核中。调试系统支持串行(SW)调试和跟踪功能，也支持 JTAG 调试。调试和跟踪功能请参考下列文档：

- Cortex®-M4技术参考手册；
- ARM®调试接口v5结构规范。

调试系统帮助调试者在低功耗模式下调试。当相应的位被置 1，调试系统会在低功耗模式下提供时钟，或者为一些外设保持当前状态，这些外设包括：TIMER、WWDGT、FWDGT、RTC、I2C 和 CAN。

## 12.2. JTAG/SW 功能说明

调试工具可以通过串行(SW)调试接口或者 JTAG 调试接口来访问调试功能。

### 12.2.1. 切换 JTAG/ SW 接口

默认使用 JTAG 调试接口，可以通过下列软件序列从 JTAG 调试切换到 SW 调试：

- 发送50个以上TCK周期的TMS=1信号；
- 发送16位TMS = 1110011110011110(0xE79E LSB)信号；
- 发送50个以上TCK周期的TMS=1信号。

切换 SW 调试到 JTAG 调试的软件序列：

- 发送50个以上TCK周期的TMS=1信号；
- 发送16位TMS = 1110011100111100 (0xE73C LSB)信号；
- 发送50个以上TCK周期的TMS=1信号。

### 12.2.2. 引脚分配

JTAG 调试提供五个引脚的接口：JTAG 时钟引脚(JTCK)，JTAG 模式选择引脚(JTMS)，JTAG 数据输入引脚(JTDI)，JTAG 数据输出引脚(JTDO)，JTAG 复位引脚(NJTRST，低电平有效)。串行调试(SWD)提供两个引脚的接口：数据输入输出引脚(SWDIO)和时钟引脚(SWCLK)。SW 调试接口的两个引脚与 JTAG 调试接口的两个引脚复用，SWDIO 和 JTMS 复用，SWCLK 和 JTCK 复用。

当异步跟踪功能开启时，JTDO 引脚也用作异步跟踪数据输出(TRACESWO)。

表 12-1. 调试引脚分配

| 引脚 | 调试接口 |
|---|---|
| PA15 | JTDI |
| PA14 | JTCK/SWCLK |
| PA13 | JTMS/SWDIO |
| PB4 | NJTRST |
| PB3 | JTDO |

默认复位后使用五个引脚的 JTAG 调试，用户可以在不使用 NJTRST 引脚情况下正常使用 JTAG 功能，此时 PB4 可以用作普通 GPIO 功能(NJTRST 硬件拉高)。如果切换到 SW 调试模式，PA15/PB4/PB3 释放作为普通 GPIO 功能。如果 JTAG 和 SW 调试功能都没有使用，这五个引脚都释放作为普通 GPIO 功能。五个引脚具体配置请参考*通用和备用输入/输出接口(GPIO 和 AFIO)*。

### 12.2.3. JTAG 链状结构

Cortex®-M4 内核的 JTAG TAP 和 MCU JTAG TAP 串行连接。MCU JTAG 的 IR(指令寄存器)是 5 位，而 Cortex®-M4 内核的 JTAG 的 IR(指令寄存器)是 4 位。所以当 JTAG 进行 IR 移位输入时，首先移位 5 位 BYPASS 指令 MCU JTAG，然后移位 4 位标准指令给 Cortex®-M4 JTAG。当进行数据移位时，数据链只需要额外添加一位，因为 MCU JTAG 已处在 BYPASS 模式。

MCU JTAG ID 代码是 0x790007A3。

### 12.2.4. 调试复位

JTAG-DP 和 SW-DP 寄存器位于上电复位域。系统复位初始化了 Cortex®-M4 的绝大部分组件，除了 NVIC，调试逻辑(FPB，DWT，ITM)。NJTRST 能复位 JTAG TAP 控制器。所以，可以在系统复位下实现调试功能。例如：复位后停止，用户在系统复位后配置相应停止位，系统复位释放后处理器会立即停止。

### 12.2.5. JEDEC-106 ID code

Cortex®-M4 集成了 JEDEC-106 ID 代码。位于 ROM 表中，映射地址为 0xE00FF000_0xE00FFFFF。

## 12.3. 调试保持功能说明

### 12.3.1. 低功耗模式调试支持

当 DBG 控制寄存器 0(DBG_CTL0)的 STB_HOLD 位置 1 并且进入待机模式，AHB 总线时钟和系统时钟由 CK_IRC16M 提供，可以在待机模式下调试。当退出待机模式后，产生系统复位。

当 DBG 控制寄存器 0(DBG_CTL0)的 DSLP_HOLD 位置 1 并且进入深度睡眠模式，AHB 总线时钟和系统时钟由 CK_IRC16M 提供，可以在深度睡眠模式下调试。

当 DBG 控制寄存器 0(DBG_CTL0)的 SLP_HOLD 位置 1 并且进入睡眠模式，AHB 总线时钟没有关闭，可以在睡眠模式下调试。

### 12.3.2. TIMER, I2C, RTC, WWDGT, FWDGT 和 CAN 外设调试支持

当内核停止，并且 DBG 控制寄存器 1(DBG_CTL1)或 DBG 控制寄存器 2(DBG_CTL2)中的相应位置 1。对于不同外设，有不同动作：

对于 TIMER 外设，TIMER 计数器停止并进行调试；

对于 I2C 外设，SMBUS 保持状态并进行调试；

对于 WWDGT 或者 FWDGT 外设，计数器时钟停止并进行调试；

对于 RTC 外设，计数器停止并进行调试；

对于 CAN 外设，接收寄存器停止计数并进行调试。

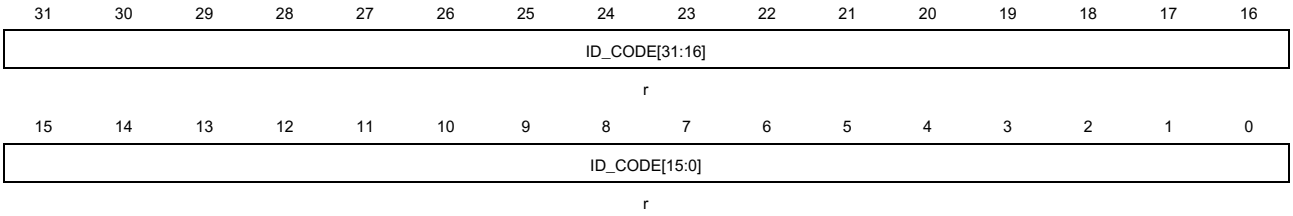
## 12.4. DBG 寄存器

DBG 基地址：0xE0042000U

### 12.4.1. ID 寄存器 (DBG_ID)

地址偏移：0x00
只读寄存器

该寄存器只能按字(32 位)访问

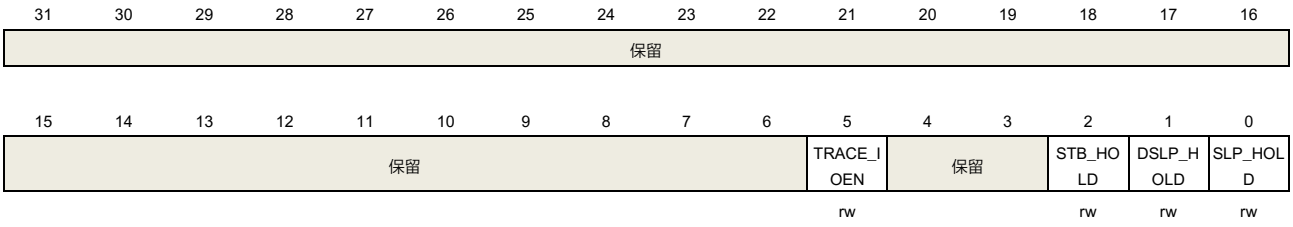



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | ID_CODE[31:0] | DBG ID 寄存器<br>这些位由软件读取，这些位是不变的常数 |

### 12.4.2. 控制寄存器 0(DBG_CTL0)

地址偏移：0x04
复位值：0x0000 0000，仅上电复位

该寄存器只能按字(32 位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:6 | 保留 | 必须保持复位值。 |
| 5 | TRACE_IOEN | 跟踪引脚分配使能<br>该位由软件置位和复位<br>0：跟踪引脚分配禁用<br>1：跟踪引脚分配使能。 |
| 4:3 | 保留 | 必须保持复位值。 |
| 2 | STB_HOLD | 待机模式保持位<br>该位由软件置位和复位 |

0：无影响
1：在待机模式下，系统时钟和 AHB 时钟由 CK_IRC16M 提供，当退出待机模式时，产生系统复位。

| | | |
|---|---|---|
| 1 | DSLP_HOLD | 深度睡眠模式保持位<br>该位由软件置位和复位<br>0：无影响<br>1：在深度睡眠模式下，系统时钟和 AHB 时钟由 CK_IRC16M 提供。 |
| 0 | SLP_HOLD | 睡眠模式保持位<br>该位由软件置位和复位<br>0：无影响<br>1：在睡眠模式下, AHB 时钟继续运行。 |

### 12.4.3. 控制寄存器 1 (DBG_CTL1)

地址偏移：0x08
复位值： 0x0000 0000，仅上电复位

该寄存器只能按字(32 位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | CAN1_HOLD | CAN0_HOLD | 保留. | I2C2_HOLD | I2C1_HOLD | I2C0_HOLD | 保留 | | | | |
| | | | | | rw | rw | | rw | rw | rw | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | FWDGT_HOLD | WWDGT_HOLD | RTC_HOLD | 保留. | TIMER13_HOLD | TIMER12_HOLD | TIMER11_HOLD | TIMER6_HOLD | TIMER5_HOLD | TIMER4_HOLD | TIMER3_HOLD | TIMER2_HOLD | TIMER1_HOLD |
| | | | rw | rw | rw | | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:27 | 保留 | 必须保持复位值。 |
| 26 | CAN1_HOLD | CAN1 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时 CAN1 接收寄存器停止接收数据。 |
| 25 | CAN0_HOLD | CAN0 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时 CAN0 接收寄存器停止接收数据。 |
| 24 | 保留 | 必须保持复位值。 |
| 23 | I2C2_HOLD | I2C2 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持 I2C2 的 SMBUS 状态不变，用于调试。 |
| 22 | I2C1_HOLD | I2C1 保持位 |

| | | |
|---|---|---|
| | | 该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持 I2C1 的 SMBUS 状态不变，用于调试。 |
| 21 | I2C0_HOLD | I2C0 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持 I2C0 的 SMBUS 状态不变，用于调试。 |
| 20:13 | 保留 | 必须保持复位值。 |
| 12 | FWDGT_HOLD | FWDGT 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持 FWDGT 计数器时钟，用于调试。 |
| 11 | WWDGT_HOLD | WWDG 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持 WWDGT 计数器时钟，用于调试。 |
| 10 | RTC_HOLD | RTC 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持 RTC 计数器不变，用于调试 |
| 9 | 保留 | 必须保持复位值。 |
| 8 | TIMER13_HOLD | TIMER13 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持定时器 13 计数器不变，用于调试。 |
| 7 | TIMER12_HOLD | TIMER 12 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持定时器 12 计数器不变，用于调试。 |
| 6 | TIMER11_HOLD | TIMER 11 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持定时器 11 计数器不变，用于调试。 |
| 5 | TIMER6_HOLD | TIMER 6 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持定时器 6 计数器不变，用于调试。 |
| 4 | TIMER5_HOLD | TIMER 5 保持位<br>该位由软件置位和复位 |

0：无影响
1：当内核停止时保持定时器 5 计数器不变，用于调试。

| | | |
|---|---|---|
| 3 | TIMER4_HOLD | TIMER 4 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持定时器 4 计数器不变，用于调试。 |
| 2 | TIMER3_HOLD | TIMER 3 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持定时器 3 计数器不变，用于调试。 |
| 1 | TIMER2_HOLD | TIMER 2 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持定时器 2 计数器不变，用于调试。 |
| 0 | TIMER1_HOLD | TIMER 1 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持定时器 1 计数器不变，用于调试。 |

### 12.4.4. 控制寄存器 (DBG_CTL2)

地址偏移：0x0C
复位值： 0x0000 0000，仅上电复位

该寄存器只能按字(32 位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | | TIMER10_HOLD | TIMER9_HOLD | TIMER8_HOLD |
| | | | | | | | | | | | | | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | | | TIMER7_HOLD | TIMER0_HOLD |
| | | | | | | | | | | | | | | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:19 | 保留 | 必须保持复位值。 |
| 18 | TIMER10_HOLD | TIMER10 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持定时器 10 计数器不变，用于调试。 |
| 17 | TIMER9_HOLD | TIMER9 保持位<br>该位由软件置位和复位<br>0：无影响 |

| | | |
|---|---|---|
| | | 1：当内核停止时保持定时器 9 计数器不变，用于调试。 |
| 16 | TIMER8_HOLD | TIMER8 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持定时器 8 计数器不变，用于调试。 |
| 15:2 | 保留 | 必须保持复位值。 |
| 1 | TIMER7_HOLD | TIMER7 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持定时器 7 计数器不变，用于调试。 |
| 0 | TIMER0_HOLD | TIMER0 保持位<br>该位由软件置位和复位<br>0：无影响<br>1：当内核停止时保持定时器 0 计数器不变，用于调试。 |

# 13. 可编程参考电流(IREF)

## 13.1. 简介

MCU 包含可编程参考电流模块。

在使用参考电流时，有两种不同的运行模式：低功耗模式和大电流模式。

两种模式的区别在于电流步长和最大电流不同。

在低功耗模式下，电流步长为 1uA，最大电流 63uA。

在大电流模式下，电流步长为 8uA，最大电流为 504uA。

## 13.2. 主要特性

参考电流特征：

- 可编程电流；
- 可编程源电流或灌电流；
- 低功耗模式和大电流模式。

## 13.3. 功能描述

### 13.3.1. 信号描述

在使用 IREF 时，相关引脚应该被配置为模拟输入模式。

### 13.3.2. 用户校验

用户可以通过编程 IREF_CTL 寄存器的 CPT 位来校准输出电流。

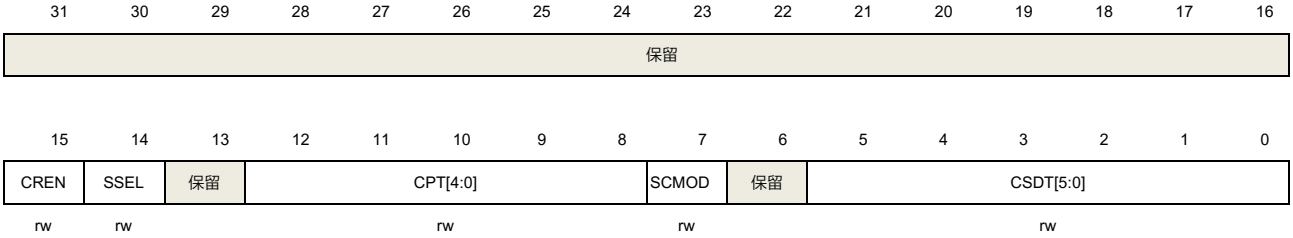
## 13.4. IREF 寄存器

IREF 基地址：0x4000 C400

### 控制寄存器 (IREF_CTL)

偏移地址：0x300
复位值：0x0000 0F00

该寄存器可以按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15 | CREN | 参考电流使能<br>0：禁用参考电流<br>1：使能参考电流 |
| 14 | SSEL | 步长选择<br>0：低功耗，步长 1uA<br>1：大电流，步长 8uA |
| 13 | 保留 | 必须保持复位值 |
| 12:8 | CPT[4:0] | 电流精度校准<br>0x00：-15%<br>….<br>0x1F：+16% |
| 7 | SCMOD | 灌电流模式<br>0：源电流模式<br>1：灌电流模式 |
| 6 | 保留 | 必须保持复位值 |
| 5:0 | CSDT[5:0] | 电流步长设置<br>0x00：默认值<br>0x01：Step * 1<br>….<br>0x3F：Step * 63 |

# 14. 模数转换器(ADC)

## 14.1. 简介

MCU 片上集成了 12 位逐次逼近式模数转换器模块(ADC)，可以采样来自于 16 个外部通道和 2 个内部通道和一个电池电压($V_{BAT}$)通道的模拟信号。这 19 个 ADC 采样通道都支持多种运行模式，采样转换后，转换结果可以按照最低有效位对齐或最高有效位对齐的方式保存在相应的数据寄存器中。片上的硬件过采样机制可以通过减少来自 MCU 的相关计算负担来提高性能。对于电机、电源等对 ADC 有更高需求的应用，可以联系我们的销售，获取更多的 ADC 详细资料。

## 14.2. 主要特征

- 高性能：
  - ADC采样分辨率：12位、10位、8位、或者6位分辨率；
  - ADC采样率：12位分辨率为2.6 MSPs，10位分辨率为3.0 MSPs。分辨率越低，转换越快；
  - 自校准时间：131个ADC时钟周期；
  - 可编程采样时间；
  - 数据存储模式：最高有效位对齐和最低有效位对齐；
  - DMA请求。
- 模拟输入通道：
  - 16个外部模拟输入通道；
  - 1个内部温度传感通道($V_{SENSE}$)；
  - 1个内部参考电压输入通道($V_{REFINT}$)；
  - 1个外部监测电池$V_{BAT}$供电引脚输入通道。
- 转换开始的发起：
  - 软件触发；
  - 硬件触发。
- 运行模式：
  - 转换单个通道，或者扫描一序列的通道；
  - 单次运行模式，每次触发转换一次选择的输入通道；
  - 连续运行模式，连续转换所选择的输入通道；
  - 间断运行模式；
  - 同步模式(适用于具有两个或多个ADC的设备)。
- 转换结果阈值监测器功能：模拟看门狗。
- 中断产生：
  - 常规转换结束；
  - 模拟看门狗事件；
  - 溢出事件。
- 过采样：
  - 16位的数据寄存器；

  - 可调整的过采样率，从2x到256x；
  - 高达8位的可编程数据移位。
- 模块供电要求：2.6V到3.6V，一般电源电压为3.3V。
- 通道输入范围：$V_{REFN}$ <=$V_{IN}$ <=$V_{REFP}$。

## 14.3. 引脚和内部信号

*图 14-1. ADC 模块框图*给出了 ADC 框图。*表 14-1. ADC 内部输入信号*给出了 ADC 内部信号。*表 14-2. ADC 输入引脚定义*给出了 ADC 引脚说明。

**表 14-1. ADC 内部输入信号**

| 内部信号名称 | 说明 |
|---|---|
| $V_{SENSE}$ | 内部温度传感器输出电压 |
| $V_{REFINT}$ | 内部参考输出电压 |

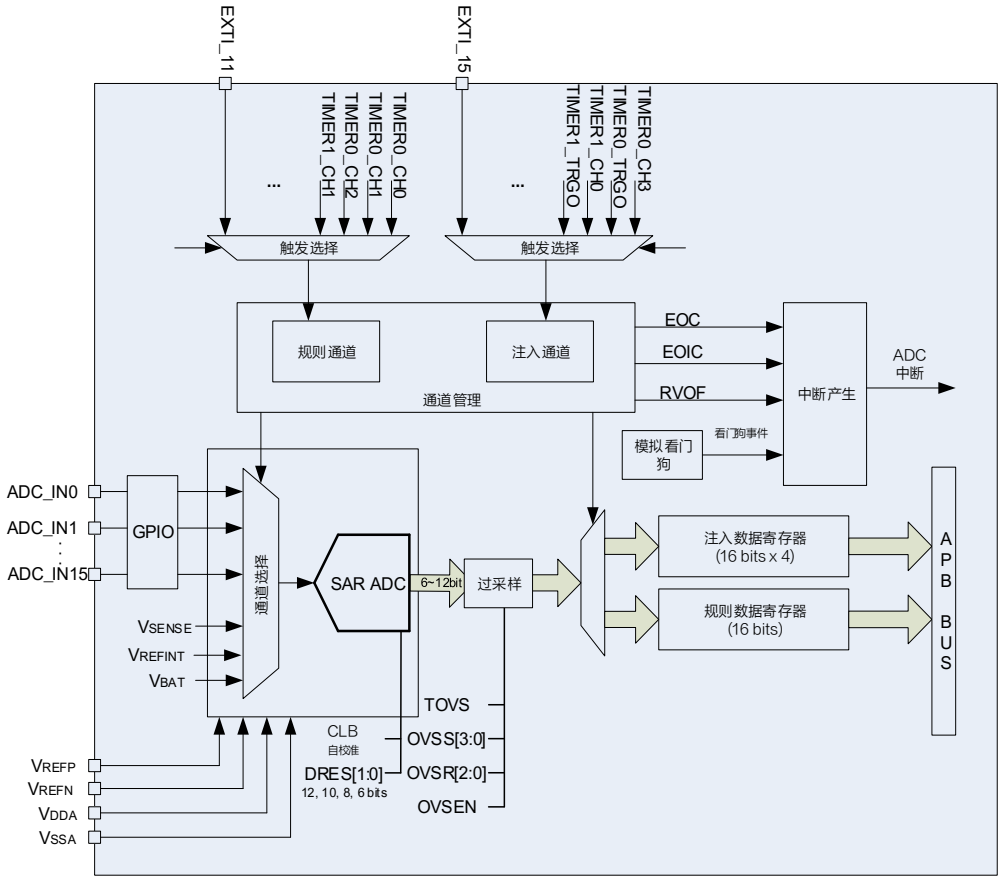
**表 14-2. ADC 输入引脚定义**

| 名称 | 说明 |
|---|---|
| $V_{DDA}$ | 模拟电源输入等于$V_{DD}$，2.6V<=$V_{DDA}$<=3.6V |
| $V_{SSA}$ | 模拟地，等于$V_{SS}$ |
| $V_{REFP}$ | ADC正参考电压，2.6 V <= $V_{REFP}$ <= $V_{DDA}$ |
| $V_{REFN}$ | ADC负参考电压，$V_{REFN}$ = $V_{SSA}$ |
| ADCx_IN[15:0] | 多达16路外部通道 |
| $V_{BAT}$ | 外部电池电压 |

**注意：**$V_{DDA}$和$V_{SSA}$必须分别连接到$V_{DD}$和$V_{SS}$。

## 14.4. 功能说明

**图 14-1. ADC 模块框图**




### 14.4.1. 前置校准功能

在前置校准期间，ADC 计算一个校准系数，这个系数是应用于 ADC 内部的，它直到 ADC 下次掉电才无效。在校准期间，应用不能使用 ADC，它必须等到校准完成。在 A/D 转换前应执行校准操作。通过软件设置 CLB=1 来对校准进行初始化，在校准期间 CLB 位会一直保持 1，直到校准完成，该位由硬件清 0。

当 ADC 运行条件改变(例如，$V_{DDA}$、$V_{REFP}$ 以及温度等)，建议重新执行一次校准操作。

内部的模拟校准通过设置 ADC_CTL1 寄存器的 RSTCLB 位来重置。

软件校准过程：

1. 确保ADCON=1；
2. 延迟14个CK_ADC以等待ADC稳定；
3. 设置RSTCLB (可选的)；
4. 设置CLB=1；
5. 等待直到CLB=0。

### 14.4.2. ADC 时钟

CK_ADC 时钟是由时钟控制器提供的，它和 AHB、APB2 时钟保持同步。ADC 最大的时钟频率为 40MHz。ADC 时钟可以在 RCU 时钟控制器中进行分配和配置。

### 14.4.3. ADCON 使能

ADC_CTL1 寄存器中的 ADCON 位是 ADC 模块的使能开关。如果该位为 0，则 ADC 模块保持复位状态。为了省电，当 ADCON 位为 0 时，ADC 模拟子模块将会进入掉电模式。ADC 使能后需等待 $t_{SU}$ 时间后才能采样，$t_{SU}$ 数值详见芯片数据手册。

### 14.4.4. 常规序列

通道管理电路可以将采样通道组织成一个序列：常规序列。常规序列支持最多 16 个通道，每个通道称为常规通道。

ADC_RSQ0 寄存器的 RL[3:0]位规定了整个常规序列的长度。ADC_RSQ0~ADC_RSQ2 寄存器规定了常规序列的通道选择。

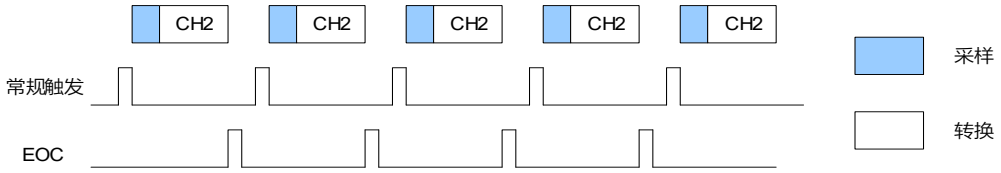
注意：尽管 ADC 支持 19 个通道，但常规序列一次最多转换 16 个通道。

### 14.4.5. 运行模式

#### 单次运行模式

单次运行模式下，ADC_RSQ2 寄存器的 RSQ0[4:0]位规定了 ADC 的转换通道。当 ADCON 位被置 1，一旦相应软件触发或者外部触发发生，ADC 就会采样和转换一个通道。

**图 14-2. 单次运行模式**




常规通道单次转换结束后，转换数据将被存放于 ADC_RDATA 寄存器中，EOC 将会置 1。如果 EOCIE 位被置 1，将产生一个中断。

常规序列单次运行模式的软件流程：

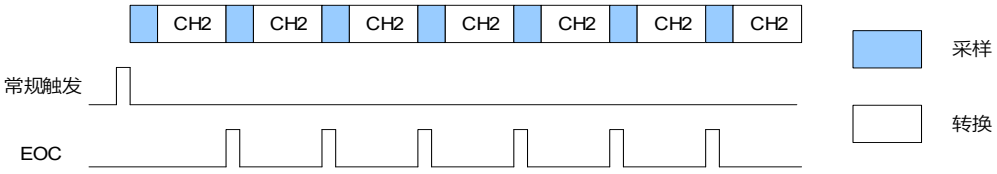
1. 确保ADC_CTL0寄存器的DISRC和SM位以及ADC_CTL1寄存器的CTN位为0；
2. 用模拟通道编号来配置RSQ0；
3. 配置ADC_SAMPTx寄存器；
4. 如果有需要，可以配置ADC_CTL1寄存器的ETMRC和ETSRC位；
5. 设置SWRCST位，或者为常规序列产生一个外部触发信号；
6. 等到EOC置1；

7. 从ADC_RDATA寄存器中读ADC转换结果；
8. 写0清除EOC标志位。

### 连续运行模式

对 ADC_CTL1 寄存器的 CTN 位置 1 可以使能连续运行模式。在此模式下，ADC 执行由 RSQ0[4:0]规定的转换通道。当 ADCON 位被置 1，一旦相应软件触发或者外部触发产生，ADC 就会采样和转换规定的通道。转换数据保存在 ADC_RDATA 寄存器中。

**图 14-3. 连续运行模式**




常规序列连续运行模式的软件流程：

1. 设置ADC_CTL1寄存器的CTN位为1；
2. 根据模拟通道编号配置RSQ0；
3. 配置ADC_SAMPTx寄存器；
4. 如果有需要，配置ADC_CTL1寄存器的ETMRC和ETSRC位；
5. 设置SWRCST位，或者给常规序列产生一个外部触发信号；
6. 等待EOC标志位置1；
7. 从ADC_RDATA寄存器中读ADC转换结果；
8. 写0清除EOC标志位；
9. 只要还需要进行连续转换，重复步骤6~8。

由于要循环查询 EOC 标志位，DMA 可以被用来传输转换数据，软件流程如下：

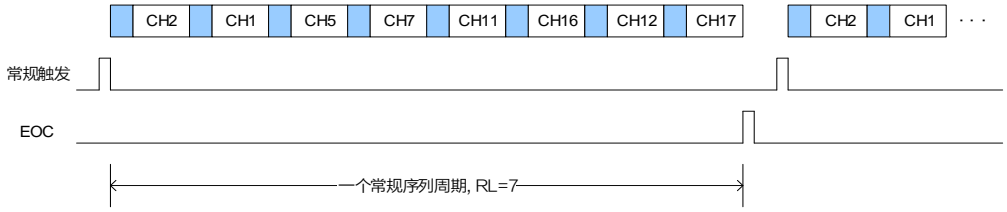
1. 设置ADC_CTL1寄存器的CTN位为1；
2. 根据模拟通道编号配置RSQ0；
3. 配置ADC_SAMPTx寄存器；
4. 如果有需要，配置ADC_CTL1寄存器的ETMRC和ETSRC位；
5. 准备*直接存储器访问控制器(DMA)*模块，用于传输来自ADC_RDATA的数据；
6. 设置SWRCST位，或者给常规序列产生一个外部触发。

### 扫描运行模式

扫描运行模式可以通过将 ADC_CTL0 寄存器的 SM 位置 1 来使能。在此模式下，ADC 扫描转换所有被 ADC_RSQ0~ADC_RSQ2 寄存器选中的所有通道。一旦 ADCON 位被置 1，当相应软件触发或者外部触发产生，ADC 就会一个接一个的采样和转换常规序列通道。转换数据存储在 ADC_RDATA 寄存器中。常规序列转换结束后，EOC 位将被置 1。如果 EOCIE 位被置 1，将产生中断。当常规序列工作在扫描模式下时，ADC_CTL1 寄存器的 DMA 位必须设置为 1。

如果 ADC_CTL1 寄存器的 CTN 位也被置 1，则在常规序列转换完之后，这个转换自动重新开始。

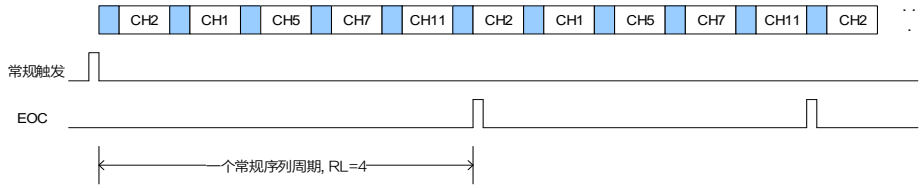
**图 14-4. 扫描运行模式，且连续转换模式失能**




常规序列扫描运行模式的软件流程：

1. 设置 ADC_CTL0 寄存器的 SM 位和 ADC_CTL1 寄存器的 DMA 位为 1；
2. 配置 ADC_RSQx 和 ADC_SAMPTx 寄存器；
3. 如果有需要，配置 ADC_CTL1 寄存器中的 ETMRC 和 ETSRC 位；
4. 准备*直接存储器访问控制器(DMA)*模块，用于传输来自 ADC_RDATA 的数据；
5. 设置 SWRCST 位，或者给常规序列产生一个外部触发。

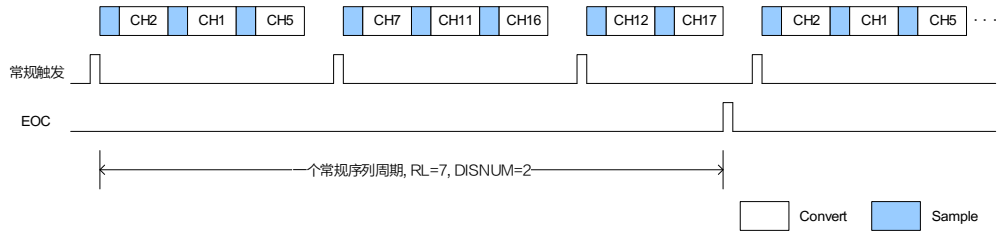
**图 14-5. 扫描运行模式，连续运行模式使能**




## 间断运行模式

当 ADC_CTL0 寄存器的 DISRC 位置 1 时，常规序列使能间断运行模式。该模式下可以执行一次 n 个通道的短序列转换(n 不超过 8)，该序列是 ADC_RSQ0~RSQ2 寄存器所选择的序列的一部分。数值 n 由 ADC_CTL0 寄存器的 DISCNUM[2:0]位配置。当相应的软件触发或外部触发发生，ADC 就会采样和转换在 ADC_RSQ0~RSQ2 寄存器所配置通道中接下来的 n 个通道，直到常规序列中所有的通道转换完成。每个常规序列转换周期结束后，EOC 位将被置 1。如果 EOCIE 位被置 1 将产生一个中断。

**图 14-6. 间断转换模式**




常规序列间断运行模式的软件流程：

1. 设置 ADC_CTL0 寄存器的 DISRC 位和 ADC_CTL1 寄存器的 DMA 位为 1；
2. 配置 ADC_CTL0 寄存器的 DISNUM[2:0]位；
3. 配置 ADC_RSQx 和 ADC_SAMPTx 寄存器；
4. 如果有需要，配置 ADC_CTL1 寄存器中的 ETMRC 和 ETSRC 位；
5. 准备*直接存储器访问控制器(DMA)*模块，用于传输来自 ADC_RDATA 的数据；
6. 设置 SWRCST 位，或者给常规序列产生一个外部触发；

7. 如果需要，重复步骤 6；
8. 等待 EOC 标志位置 1；
9. 写 0 清除 EOC 标志位。

### 14.4.6. 转换结果阈值监测功能

ADC_CTL0 寄存器的 RWDEN 位置 1 将使能常规序列模拟看门狗功能。该功能用于监测转换结果是否超过设定的阈值。如果 ADC 的模拟转换电压低于低阈值或高于高阈值时，ADC_STAT 状态寄存器的 WDE 位将被置 1。如果 WDEIE 位被置 1，将产生中断。ADC_WDHT 和 ADC_WDLT 寄存器用来设定高低阈值。内部数据的比较在对齐之前完成，因此阀值与 ADC_CTL1 寄存器的 DAL 位确定的对齐方式无关。ADC_CTL0 寄存器的 RWDEN，WDSC 和 WDCHSEL[4:0]位可以用来选择模拟看门狗监控单一通道或者多通道。

### 14.4.7. 数据存储模式

ADC_CTL1 寄存器的 DAL 位确定转换后数据存储的对齐方式。

**图 14-7. 12 位数据存储模式**

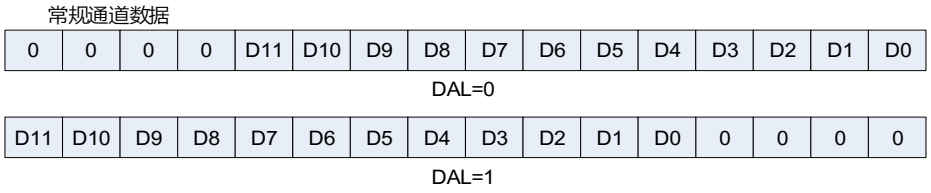



**图 14-8. 10 位数据存储模式**

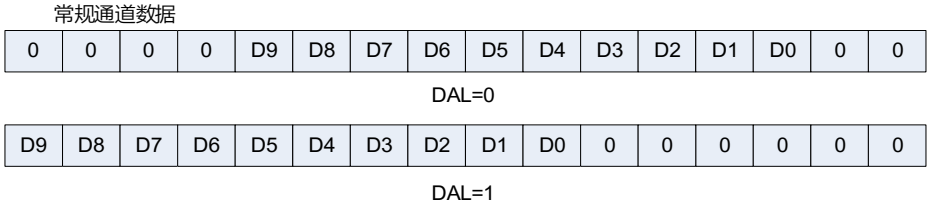
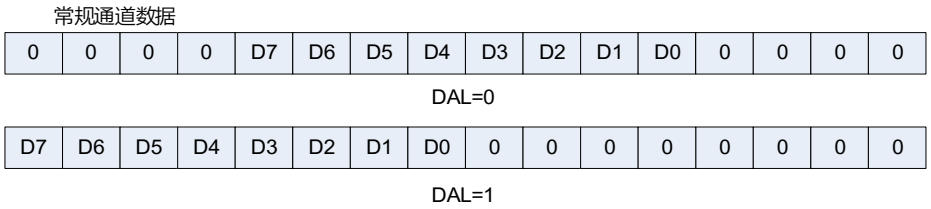
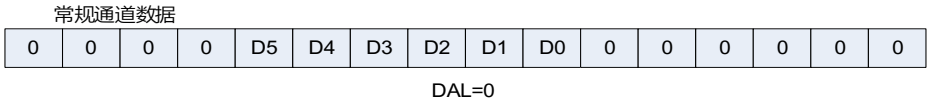



**图 14-9. 8 位数据存储模式**




6 位分辨率的数据存储模式不同于 12 位/10 位/8 位分辨率数据存储模式，如*图 14-10. 6 位数据存储模式*。

图 14-10. 6 位数据存储模式




## 14.4.8. 采样时间配置

ADC 使用多个 CK_ADC 周期对输入电压采样，采样周期数目可以通过 ADC_SAMPT0 和 ADC_SAMPT1 寄存器的 SPTn[2:0]位配置。每个通道可以用不同的采样时间。在 12 位分辨率的情况下，总转换时间=采样时间+12 个 CK_ADC 周期。

**例如:**

CK_ADC = 40MHz ，采样时间为 3 个周期，那么总的转换时间为："3+12"个 CK_ADC 周期，即 0.375us。

## 14.4.9. 外部触发

外部触发输入的上升沿、下降沿可以触发常规序列的转换。ADC_CTL1 寄存器的 ETMRC[1:0]位控制常规序列的触发模式。常规序列的外部触发源由 ADC_CTL1 寄存器的 ETSRC[3:0]位控制。

表 14-3. 外部触发模式

| ETMRC[1:0] | 触发模式 |
|---|---|
| 00 | 外部触发失能 |
| 01 | 外部触发信号上升沿触发使能 |
| 10 | 外部触发信号下降沿触发使能 |
| 11 | 外部触发信号双边沿触发使能 |

表 14-4. ADC 的外部触发源

| ETSRC[3:0] | 触发源 | 触发类型 |
|---|---|---|
| 0000 | TIMER0_CH0 | 硬件触发 |
| 0001 | TIMER0_CH1 | |
| 0010 | TIMER0_CH2 | |
| 0011 | TIMER1_CH1 | |
| 0100 | TIMER1_CH2 | |
| 0101 | TIMER1_CH3 | |
| 0110 | TIMER1_TRGO | |
| 0111 | TIMER2_CH0 | |
| 1000 | TIMER2_TRGO | |
| 1001 | TIMER3_CH3 | |
| 1010 | TIMER4_CH0 | |
| 1011 | TIMER4_CH1 | |
| 1100 | TIMER4_CH2 | |
| 1101 | TIMER7_CH0 | |

| ETSRC[3:0] | 触发源 | 触发类型 |
|---|---|---|
| 1110 | TIMER7_TRGO | |
| 1111 | EXTI_11 | |

可以实时修改外部触发选择，在修改期间不会出现触发事件。

### 14.4.10. DMA 请求

DMA 请求，可以通过设置 ADC_CTL1 寄存器的 DMA 位来使能，它用于常规序列多个通道的转换结果。ADC 在常规序列一个通道转换结束后产生一个 DMA 请求，DMA 接受到请求后可以将转换的数据从 ADC_RDATA 寄存器传输到用户指定的目的地址。

### 14.4.11. 溢出检测

当 DMA 使能的时候或者将 ADC_CTL1 寄存器的 EOCM 位置 1，可以使能溢出检测。如果一个常规转换在上一个常规转换数据读出之前已经完成，则会产生一个溢出事件，相应的 ADC_STAT 状态寄存器的 ROVF 标志位会置位。如果 ADC_CTL0 寄存器的 ROVFIE 置位，溢出中断产生。

为了使得 ADC 从 ROVF 溢出状态中恢复过来，建议对 DMA 模块重新进行初始化。内部状态机复位，以保证常规转换数据正确的传输。ADC 转换将会停止，直到 ROVF 位被清零。

ADC 从 ROVF 状态恢复的软件流程如下：

1. 将 ADC_CTL1 寄存器的 DMA 位清 0；
2. 将 ADC_CTL1 寄存器的 ADON 位清 0；
3. 将 DMA_CHxCTL 寄存器的 CHEN 位清 0，用于重新初始化 DMA 模块；
4. 将 ADC_STAT 寄存器的 ROVF 位清 0；
5. 将 DMA_CHxCTL 寄存器的 CHEN 位置 1；
6. 将 ADC_CTL1 寄存器的 DMA 位置 1；
7. 将 ADC_CTL1 的 ADON 位置 1；
8. 等待 T(setup)；
9. 通过软件或触发开始 ADC 转换。

### 14.4.12. ADC 内部通道

将 ADC_SYNCCTL 寄存器的 TSVREN 位置 1 可以使能温度传感器通道(ADC0_IN16)和 $V_{REFINT}$ 通道(ADC0_IN17)。温度传感器可以用来测量器件周围的温度。传感器输出电压能被 ADC 转换成数字量。建议温度传感器的采样时间至少设置为 $t_{s_temp}$ μs。温度传感器不用时，复位 TSVREN 位可以将其置于掉电模式。

温度传感器的输出电压随温度会发生线性变化，由于芯片生产过程的多样化，温度变化曲线的偏移在不同的芯片上会有不同(最多相差 45°C)。内部温度传感器更适合于检测温度的变化，而不是测量绝对温度。如果需要测量精确的温度，应该使用一个外置的温度传感器来校准这个偏移错误。

内部电压参考($V_{REFINT}$)提供了一个稳定的(带隙基准)电压输出给 ADC 和比较器。$V_{REFINT}$ 内部连

接到 ADC0_IN17 输入通道。

使用温度传感器：

1. 配置温度传感器通道(ADC0_IN16)的转换序列和采样时间为$t_{s_temp}$ us。
2. 置位ADC_CTL1寄存器的TSVREN位，使能温度传感器。
3. 置位ADC_CTL1寄存器的ADCON位，或者由外部触发ADC转换。
4. 读取内部温度传感器输出电压$V_{temperature}$，并由下面公式计算出实际温度：

$$温度\ (°C) = \{(V_{25} - V_{temperature}) / Avg_Slope\} + 25$$

$V_{25}$：内部温度传感器在 25°C 下的电压，典型值请参考相关型号 datasheet。

Avg_Slope：温度与内部温度传感器输出电压曲线的均值斜率，典型值请参考相关型号 datasheet。

### 14.4.13. 电池电压监测

$V_{BAT}$通道由于监测从$V_{BAT}$引脚过来的备份电池电压。当ADC_SYNCCTL寄存器中的VBATEN位置1时，使能$V_{BAT}$通道(ADC_IN18)，同时一个集成在$V_{BAT}$引脚上的4分压桥也随之自动被使能。由于$V_{BAT}$可能比$V_{DDA}$高，所以使用这个4分压桥用来确保ADC正确操作。它将ADC_IN18输入通道连接到$V_{BAT}$/4，所以，ADC_IN18输入通道转换的值是$V_{BAT}$/4。为了防止不必要的电池能量消耗，推荐仅在需要时才使能4分压桥。

### 14.4.14. 可编程分辨率(DRES)

ADC 分辨率可以通过寄存器 ADC_CTL0 中的 DRES[1:0]位进行配置。对于那些不需要高精度数据的应用，可以使用较低的分辨率来实现更快速地转换。只有在 ADCON 比特为 0 时，才能修改 DRES[1:0]的值。ADC 转换的结果只有 12 位，其余没有被用到的低位读出来都是为 0。

较低的分辨率能够减少转换时间。如图*表 14-5. 不同分辨率对应的 tCONV 时间*所示，较低的分辨率能够减少逐次逼近步骤所需的转换时间 $t_{ADC}$。

**表 14-5. 不同分辨率对应的 $t_{CONV}$ 时间**

| DRES[1:0] bits | $t_{CONV}$ (ADC clock cycles) | $t_{CONV}$(ns) at $f_{ADC}$=40MHz | $t_{SMPL}$(min) (ADC clock cycles) | $t_{ADC}$ (ADC clock cycles) | $t_{ADC}$(us) at $f_{ADC}$=40MHz |
|---|---|---|---|---|---|
| 12 | 12 | 300 ns | 3 | 15 | 375 ns |
| 10 | 10 | 250 ns | 3 | 13 | 325 ns |
| 8 | 8 | 200 ns | 3 | 11 | 275 ns |
| 6 | 6 | 150 ns | 3 | 9 | 225 ns |

### 14.4.15. 片上硬件过采样

片上硬件过采样单元执行数据预处理以减轻 CPU 负担。它能够处理多个转换，并将多个转换的结果取平均，得出一个 16 位宽的数据。其结果值根据如下公式计算得出，其中 N 和 M 的值可以被调整，过采样单元可以通过设置 ADC_OVSAMPCTL 寄存器的 OVSEN 位来使能，它是

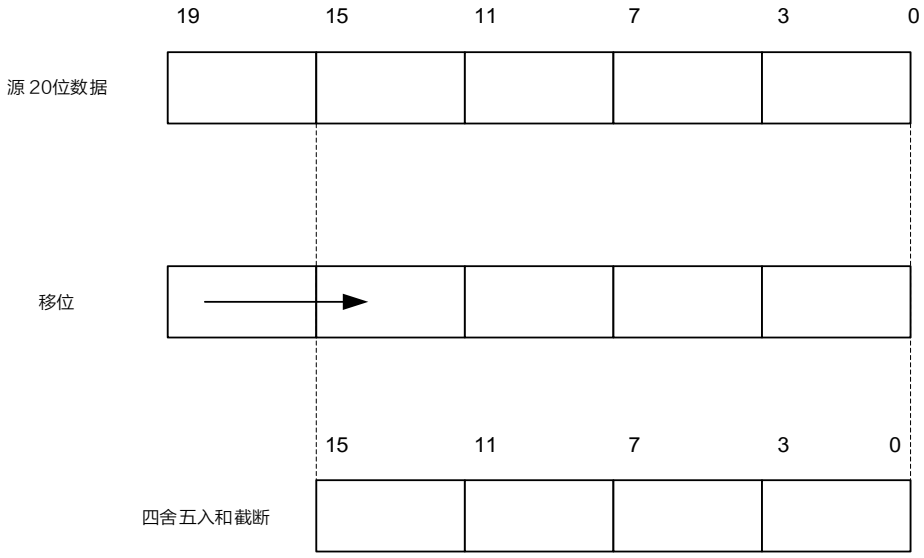
以降低数据输出率为代价，换取较高的数据分辨率。$D_{out}(n)$是指 ADC 输出的第 n 个数字信号：

$$Result=\frac{1}{M}*\sum_{n=0}^{N-1}D_{out}(n) \quad (14\text{-}1)$$

片上硬件过采样单元执行两个功能：求和和位右移。过采样率 N 是在 ADC_OVSAMPCTL 寄存器的 OVSR[2:0]位定义，它的取值范围为 2x 到 256x。除法系数 M 定义一个多达 8 位的右移，它通过 ADC_OVSAMPCTL 寄存器 OVSS[3:0]位进行配置。

求和单元能够生成一个多达 20 位(256*12 位)的值。首先，将这个值要进行右移，将移位后剩余的部分再通过取整转化一个近似值，最后将高位会被截断，仅保留最低 16 位有效位作为最终值传入对应的数据寄存器中。

**图 14-11. 20 位到 16 位的结果截断**




**注意：**如果移位后的中间结果还是超过 16 位，那么该结果的高位就会被直接截掉。

*图 14-11. 20 位到 16 位的结果截断*描述一个从原始 20 位的累积数值处理成 16 位结果值的例子。

**图 14-12. 右移 5 位和取整的数例**

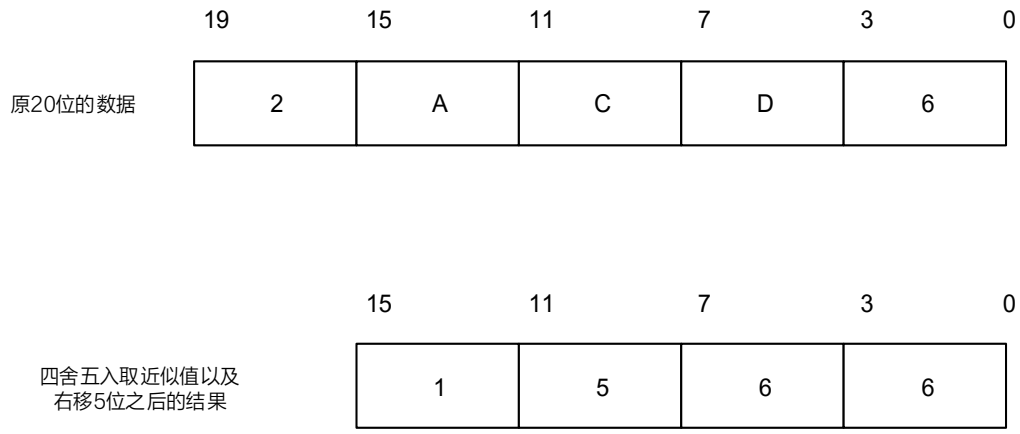


*表 14-6. N 和 M 的最大输出值(灰色部分表示截断)*给出了 N 和 M 各种组合的数据格式，初始转换值为 0xFFF。

**表 14-6. N 和 M 的最大输出值(灰色部分表示截断)**

| Oversampling ratio | Max Raw data | No-shift OVSS=0000 | 1-bit shift OVSS=0001 | 2-bit shift OVSS=0010 | 3-bit shift OVSS=0011 | 4-bit shift OVSS=0100 | 5-bit shift OVSS=0101 | 6-bit shift OVSS=0110 | 7-bit shift OVSS=0111 | 8-bit shift OVSS=1000 |
|---|---|---|---|---|---|---|---|---|---|---|
| 2x | 0x1FFE | 0x1FFE | 0x0FFF | 0x07FF | 0x03FF | 0x01FF | 0x00FF | 0x007F | 0x003F | 0x001F |
| 4x | 0x3FFC | 0x3FFC | 0x1FFE | 0x0FFF | 0x07FF | 0x03FF | 0x01FF | 0x00FF | 0x007F | 0x003F |
| 8x | 0x7FF8 | 0x7FF8 | 0x3FFC | 0x1FFE | 0x0FFF | 0x07FF | 0x03FF | 0x01FF | 0x00FF | 0x007F |
| 16x | 0xFFF0 | 0xFFF0 | 0x7FF8 | 0x3FFC | 0x1FFE | 0x0FFF | 0x07FF | 0x03FF | 0x01FF | 0x00FF |
| 32x | 0x1FFE0 | 0xFFE0 | 0xFFF0 | 0x7FF8 | 0x3FFC | 0x1FFE | 0x0FFF | 0x07FF | 0x03FF | 0x01FF |
| 64x | 0x3FFC0 | 0xFFC0 | 0xFFE0 | 0xFFF0 | 0x7FF8 | 0x3FFC | 0x1FFE | 0x0FFF | 0x07FF | 0x03FF |
| 128x | 0x7FF80 | 0xFF80 | 0xFFC0 | 0xFFE0 | 0xFFF0 | 0x7FF8 | 0x3FFC | 0x1FFE | 0x0FFF | 0x07FF |
| 256x | 0xFFF00 | 0xFF00 | 0xFF80 | 0xFFC0 | 0xFFE0 | 0xFFF0 | 0x7FF8 | 0x3FFC | 0x1FFE | 0x0FFF |

和标准的转换模式相比，过采样模式的转换时间不会改变：在整个过采样序列的过程中采样时间仍然保持相等。每 N 个转换就会产生一个新的数据，一个等价的延迟为：

$$N \times t_{ADC} = N \times (t_{SMPL} + t_{CONV}) \quad (14\text{-}2)$$

## 14.5. ADC 同步模式

在有多个 ADC 模块的产品中，可以使用 ADC 同步模式。在 ADC 同步模式下，根据 ADC_SYNCCTL 寄存器中 SYNCM[4:0]位所选的模式，转换的启动可以是 ADC0/ADC1/ADC2 的交替触发或同步触发。

在 ADC 同步模式中，当转换配置成外部事件触发时，ADC1 和 ADC2 的外部触发必须失能。常规通道的转换结果存储在 ADC 同步常规数据寄存器(ADC_SYNCDATA)中。

ADC 同步模式如*表 14-7. ADC 同步模式表*所示。

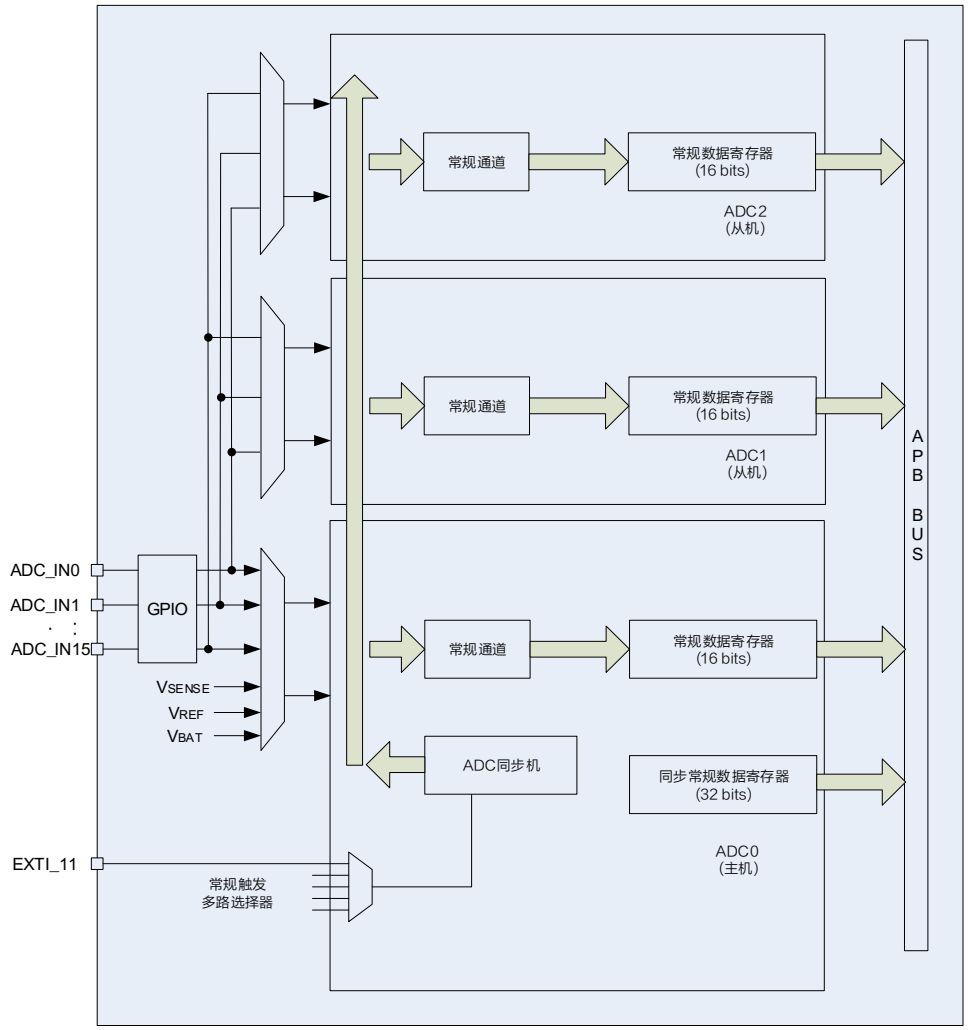
**表 14-7. ADC 同步模式表**

| SYNCM[4: 0] | mode |
|---|---|
| 00000 | 独立模式 |
| 00110 | ADC0和ADC1工作在常规并行模式 |
| 00111 | ADC0和ADC1工作在常规跟随模式 |
| 10110 | 所有ADC工作在常规并行模式 |
| 10111 | 所有ADC工作在常规跟随模式 |

当 ADC 工作在同步模式，而非独立模式时，如果需要再将 ADC 配置成其他同步模式，则需要在配置成其他同步模式前，首先将 ADC 配置成独立模式。

ADC 同步框图如*图 14-13. ADC 同步框图*所示。

图 14-13. ADC 同步框图




### 14.5.1. 独立模式

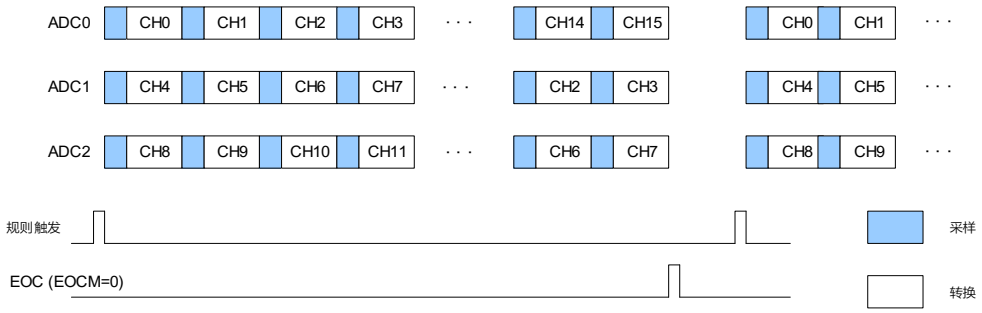
在这种模式下，ADC 同步是忽略的，每个 ADC 都独立工作。

### 14.5.2. 常规并行模式

设置 ADC_SYNCCTL 寄存器的 SYNCM[4:0]位为 5'b00110 或 5'b 10110，使能常规并行模式。在常规并行模式中，根据 ADC0 中选择的外部触发，所有的 ADC 并行的转换常规序列通道。触发选择由 ADC0 的 ADC_CTL1 寄存器 ETSRC[3:0]位进行配置。

根据 ADC_CTL1 寄存器中的 EOCM 位的设置，在转换结束时产生 EOC 中断(如果 ADC 接口使能了该中断)。常规并模式的行为如*图 14-14. 基于 16 个通道的常规并行模式*所示。

**图 14-14. 基于 16 个通道的常规并行模式**




**注意:**

1. 在一个给定的时间，两个ADC不能同时转换同一个通道。(当转换同一通道时，不能覆盖采样时间)
2. 确保在没有任何一个ADC在进行转换的时候才触发ADC。
3. 如果SYNCM=5'b00110，ADC2工作在独立模式。

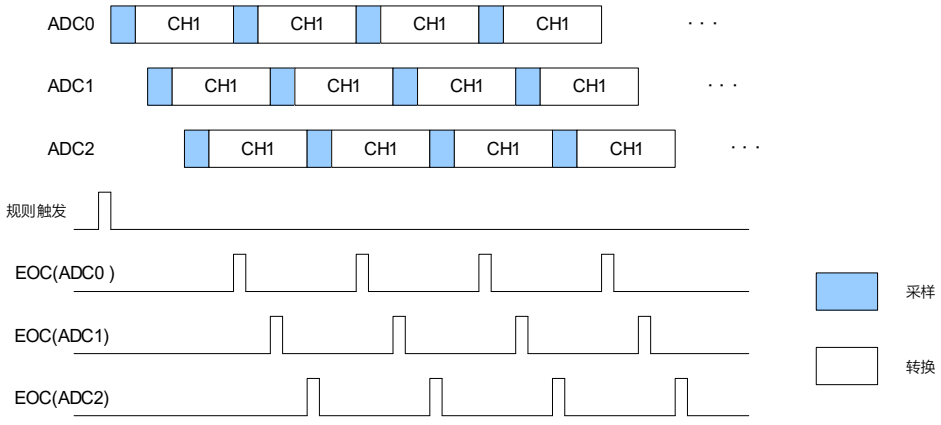
### 14.5.3. 常规跟随模式

设置 ADC_SYNCCTL 寄存器的 SYNCM[4:0]位为 5'b00111 或 5'b10111,使能常规跟随模式。在常规跟随模式中，根据选择的外部触发，ADC0 开始转换常规序列。外部触发选择由 ADC0 的 ADC_CTL1 寄存器 ETSRC[3:0]位进行配置。经过一定的延迟之后，ADC1 开始转换常规序列，再经过另一个延迟之后，ADC2 开始转换常规序列。以上描述中提到的常规序列只能包含一个常规通道。

在两个连续采样阶段之间的延迟时间，由 ADC_SYNCCTL 寄存器的 SYNCDLY[3:0]位进行配置。如果 SYNCDLY[3:0]位配置的延迟时间比采样时间还短，为了避免在一个给定时间，多个 ADC 对同一个通道进行采样，会将(采样时间 + 2)CK_ADC 周期作为实际的延迟时间。

如果 ADC_CTL1 寄存器的 CNT 位置 1，选择的常规序列会被连续的转换。根据 ADC_CTL1 寄存器的 EOCM 的配置，在转换事件结束时产生 EOC 中断(如果 ADC 使能了该中断)。跟随模式的行为如*图 14-15. 一个采用连续运行模式通道上的跟随模式*所示。

**图 14-15. 一个采用连续运行模式通道上的跟随模式**




**注意:**

1. 确保在没有任何一个 ADC 在进行转换的时候才触发 ADC(当有某些转换还没完成时，不触发 ADC0)；
2. 如果 SYNCM=5'b 00111，ADC2 工作在独立模式。

### 14.5.4. 在 ADC 同步模式中使用 DMA

在 ADC 同步模式中，常规序列通道转换的数据存储在 ADC 同步常规数据寄存器(ADC_SYNCDATA)中，DMA 可以用来传输 ADC_SYNCDATA 寄存器的数据。有以下两种 DMA 工作模式，可以和各种 ADC 同步模式很好地配合使用。

#### ADC 同步 DMA 模式 0

在 ADC 同步 DMA 模式 0 中，DMA 传输的位宽为 16。一次 DMA 请求传输一个数据，这个数据轮流的从各 ADC 的常规转换结果中取出。对于每次 DMA 请求，DMA 通道的源地址固定为 ADC_SYNCDATA 寄存器，而这个寄存器的内容会变成 DMA 要被传输的数值。当 ADC0 和 ADC1 工作在同步模式时，DMA 的传输序列为：ADC0_RDATA[15:0] -> ADC1_RDATA[15:0] -> ADC0_RDATA[15:0] -> ADC1_RDATA[15:0]。当所有的 ADC 都工作在同步模式时，DMA 的传输序列为：ADC0_RDATA[15:0] -> ADC1_RDATA[15:0] -> ADC2_RDATA[15:0] -> ADC0_RDATA[15:0] -> ADC1_RDATA[15:0] -> ADC2_RDATA[15:0]。

ADC 同步 DMA 模式 0 适用于：

- ADC0 和 ADC1 工作在常规并行模式(SYNCM=5'b00110)；
- 所有的 ADC 工作在常规并行模式(SYNCM=5'b10110)。

#### ADC 同步 DMA 模式 1

在 ADC 同步 DMA 模式 1 中，DMA 传输的位宽为 32。一次 DMA 请求传输两个数据，这些数据轮流的从各 ADC 的常规转换结果中取出。对于每次 DMA 请求，DMA 通道的源地址固定为 ADC_SYNCDATA 寄存器，而这个寄存器的内容会变成 DMA 要被传输的数值。当 ADC0 和 ADC1 工作在同步模式时，DMA 的数据每次都为：{ADC1_RDATA[15:0], ADC0_RDATA[15:0]}。当所有的 ADC 都工作在同步模式时，DMA 的传输序列为：{ADC1_RDATA[15:0],ADC0_RDATA[15:0]} -> {ADC0_RDATA[15:0],ADC2_RDATA[15:0]} -> {ADC2_RDATA[15:0],ADC1_RDATA[15:0]} -> {ADC1_RDATA[15:0],ADC0_RDATA[15:0]}。

ADC 同步 DMA 模式 1 适用于：

- ADC0 和 ADC1 工作在常规并行模式(SYNCM=5'b 00110)；
- ADC0 和 ADC1 工作在常规跟随模式(SYNCM=5'b 00111)；
- 所有的 ADC 工作在常规跟随模式(SYNCM=5'b 10111)。

## 14.6. 中断

以下任一个事件发生都可以产生中断：

- 常规通道或序列转换结束；
- 模拟看门狗事件；

- 溢出事件。

ADC0、ADC1和ADC2都被映射到同一个中断向量ISR[18]。

## 14.7. ADC 寄存器

ADC0 基地址： 0x4001 2000

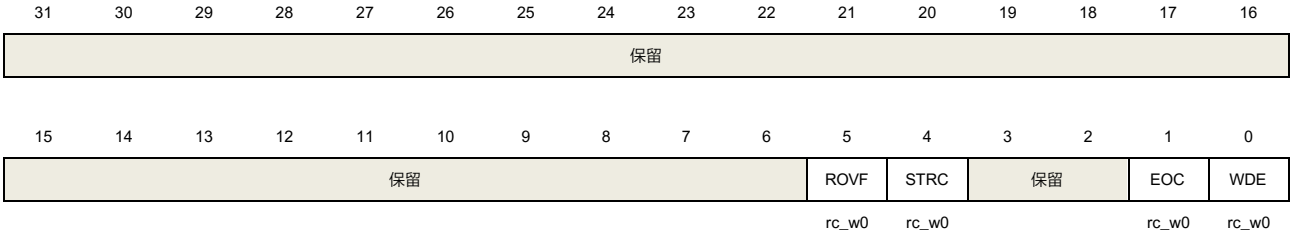
ADC1 基地址:: 0x4001 2100

ADC2 基地址:: 0x4001 2200

### 14.7.1. 状态寄存器 (ADC_STAT)

地址偏移：0x00
复位值：0x0000 0000

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:6 | 保留 | 必须保持复位值。 |
| 5 | ROVF | 常规数据寄存器溢出<br>0：常规数据寄存器没有溢出<br>1：常规数据寄存器溢出<br>在单次或多次模式中，当常规数据寄存器溢出时，该位由硬件置位。只有在 DMA 使能或者转换结束模式被置 1(EOCM=1)时，这个标志位才会置位。如果出现 ROVF 置位，则最后的常规数据会被丢失。<br>软件写'0"清除。 |
| 4 | STRC | 常规序列转换开始标志<br>0：转换没有开始<br>1：转换开始<br>常规序列转换开始时硬件置位，软件写0清除。 |
| 3:2 | 保留 | 必须保持复位值。 |
| 1 | EOC | 常规序列转换结束标志<br>0：常规序列转换没有结束<br>1：常规序列转换结束<br>常规序列转换结束时硬件置位，软件写 0 或读 ADC_RDATA 寄存器清除。 |
| 0 | WDE | 模拟看门狗事件标志<br>0：没有模拟看门狗事件<br>1：产生模拟看门狗事件 |

转换电压超过 ADC_WDLT 和 ADC_WDHT 寄存器设定的阈值时由硬件置 1，软件写 0 清除。

### 14.7.2. 控制寄存器 0 (ADC_CTL0)

地址偏移：0x04
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DISNUM[2:0] | | | 保留 | DISRC | 保留 | WDSC | SM | 保留 | WDEIE | EOCIE | WDCHSEL[4:0] | | | | |
| rw | | | | rw | | rw | rw | | rw | rw | rw | | | | |

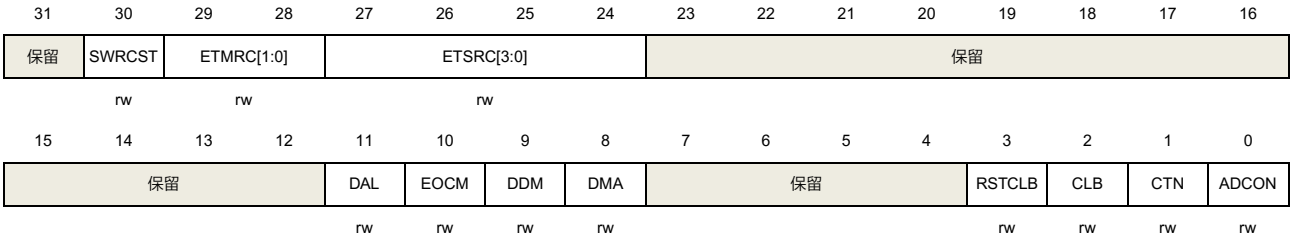
| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:27 | 保留 | 必须保持复位值。 |
| 26 | ROVFIE | ROVF 中断使能<br>0: ROVF 中断失能<br>1: ROVF 中断使能 |
| 25:24 | DRES[1:0] | ADC 数据分辨率<br>00: 12 位<br>01: 10 位<br>10: 8 位<br>11: 6 位 |
| 23 | RWDEN | 常规序列看门狗使能<br>0：常规序列看门狗禁止<br>1：常规序列看门狗使能 |
| 22:16 | 保留 | 必须保持复位值。 |
| 15:13 | DISNUM[2:0] | 间断模式下的转换数目<br>触发后即将被转换的通道数目将变成 DISNUM[2:0]+1 |
| 12 | 保留 | 必须保持复位值。 |
| 11 | DISRC | 常规序列间断模式<br>0：间断运行模式禁止<br>1：间断运行模式使能 |
| 10 | 保留 | 必须保持复位值。 |
| 9 | WDSC | 扫描模式下，模拟看门狗在单通道有效<br>0：模拟看门狗在所有通道有效 |

| | | |
|---|---|---|
| | | 1：模拟看门狗在单通道有效 |
| 8 | SM | 扫描模式<br>0：扫描运行模式禁止<br>1：扫描运行模式使能 |
| 7 | 保留 | 必须保持复位值。 |
| 6 | WDEIE | WDE 中断使能<br>0：中断禁止<br>1：中断使能 |
| 5 | EOCIE | EOC 中断使能<br>0：中断禁止<br>1：中断使能 |
| 4:0 | WDCHSEL[4:0] | 模拟看门狗通道选择<br>00000: ADC 通道 0<br>00001: ADC 通道 1<br>00010: ADC 通道 2<br>00011: ADC 通道 3<br>00100: ADC 通道 4<br>00101: ADC 通道 5<br>00110: ADC 通道 6<br>00111: ADC 通道 7<br>01000: ADC 通道 8<br>01001: ADC 通道 9<br>01010: ADC 通道 10<br>01011: ADC 通道 11<br>01100: ADC 通道 12<br>01101: ADC 通道 13<br>01110: ADC 通道 14<br>01111: ADC 通道 15<br>10000: ADC 通道 16<br>10001: ADC 通道 17<br>10010: ADC 通道 18<br>其他值保留。<br>**注意：** ADC0 的模拟输入通道 16，通道 17 和通道 18 分别连接到温度传感器，$V_{REFINT}$ 和 $V_{BAT}$ 模拟输入。 ADC1 的模拟输入通道 16，通道 17 和通道 18 内部都连接到 $V_{SSA}$。 ADC2 的模拟输入通道 16，通道 17 和通道 18 内部都连接到 $V_{SSA}$。 |

### 14.7.3. 控制寄存器 1 (ADC_CTL1)

地址偏移：0x08
复位值：0x0000 0000

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30 | SWRCST | 常规序列软件启动转换<br>该位置 1 开启常规序列转换。软件置位，软件清零，或转换开始后，立刻由硬件清零。 |
| 29:28 | ETMRC[1:0] | 常规序列外部触发模式<br>00：常规序列外部触发失能<br>01：常规序列外部触发上升沿使能<br>01：常规序列外部触发下降沿使能<br>11：常规序列外部触发双边沿使能 |
| 27:24 | ETSRC[3:0] | 常规序列的外部触发选择<br>0000： 定时器 0 通道 0<br>0001： 定时器 0 通道 1<br>0010： 定时器 0 通道 2<br>0011： 定时器 1 通道 1<br>0100： 定时器 1 通道 2<br>0101： 定时器 1 通道 3<br>0110： 定时器 1TRGO<br>0111： 定时器 2 通道 0<br>1000： 定时器 2 TRGO<br>1001： 定时器 3 通道 3<br>1010： 定时器 4 通道 0<br>1011： 定时器 4 通道 1<br>1100： 定时器 4 通道 2<br>1101： 定时器 7 通道 0<br>1110： 定时器 7 TRGO<br>1111： EXTI 外部中断线 11 |
| 23:12 | 保留 | 必须保持复位值。 |
| 11 | DAL | 数据对齐<br>0：最低有效位对齐<br>1：最高有效位对齐 |
| 10 | EOCM | 转换结束模式<br>0：只有在常规转换序列转换结束时，才将 EOC 置 1。如果不设置 DMA=1，则溢出检测失能。 |

| | | |
|---|---|---|
| | | 1：在每个常规转换结束时，将 EOC 置 1。溢出检测自动使能。 |
| 9 | DDM | DMA 失能模式<br>该位用于在单次 ADC 模式下配置 DMA 失能。<br>0：DMA 机制在 DMA 控制器的传输结束信号之后失能。<br>1：当 DMA=1，在每个常规转换结束时 DMA 机制产生一个 DMA 请求。 |
| 8 | DMA | DMA 请求使能<br>0：DMA 请求禁止<br>1：DMA 请求使能 |
| 7:4 | 保留 | 必须保持复位值。 |
| 3 | RSTCLB | 校准复位<br>在校准寄存器初始化后该位可以软件置位和硬件清零。<br>0：校准寄存器初始化结束.<br>1：校准寄存器初始化开始 |
| 2 | CLB | ADC 校准<br>0：校准结束<br>1：校准开始 |
| 1 | CTN | 连续模式<br>0：禁止连续运行模式<br>1：使能连续运行模式 |
| 0 | ADCON | 开启 ADC。该位从 0 变成 1 将唤醒 ADC。为了省电，当该位为 0 时，模拟子模块将会进入掉电模式。<br>0：失能 ADC，并进入掉电模式<br>1：使能 ADC |

### 14.7.4. 采样时间寄存器 0 (ADC_SAMPT0)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | SPT18[2:0] | | | SPT17[2:0] | | | SPT16[2:0] | | | SPT15[2:1] | |
| | | | | | rw | | | rw | | | rw | | | rw | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SPT15[0] | | SPT14[2:0] | | | SPT13[2:0] | | | SPT12[2:0] | | | SPT11[2:0] | | | SPT10[2:0] | |
| rw | | rw | | | rw | | | rw | | | rw | | | rw | |

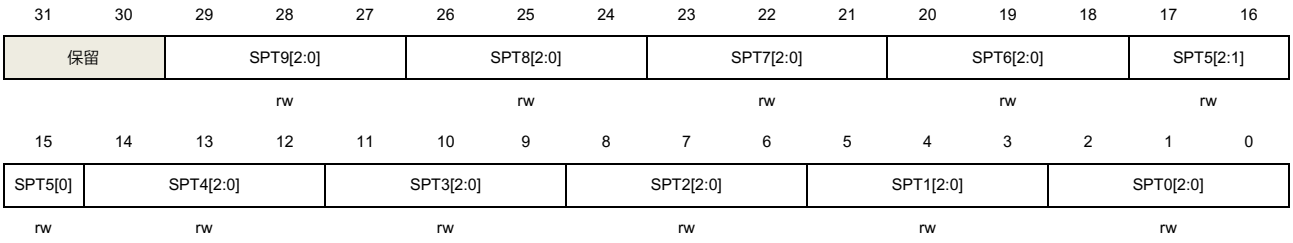
| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值。 |
| 26:24 | SPT18[2:0] | 参考 SPT10[2:0]的描述 |

| | | |
|---|---|---|
| 23:21 | SPT17[2:0] | 参考 SPT10[2:0]的描述 |
| 20:18 | SPT16[2:0] | 参考 SPT10[2:0]的描述 |
| 17:15 | SPT15[2:0] | 参考 SPT10[2:0]的描述 |
| 14:12 | SPT14[2:0] | 参考 SPT10[2:0]的描述 |
| 11:9 | SPT13[2:0] | 参考 SPT10[2:0]的描述 |
| 8:6 | SPT12[2:0] | 参考 SPT10[2:0]的描述 |
| 5:3 | SPT11[2:0] | 参考 SPT10[2:0]的描述 |
| 2:0 | SPT10[2:0] | 通道采样时间<br>000:通道采样时间为 3 个周期<br>001:通道采样时间为 15 个周期<br>010:通道采样时间为 28 个周期<br>011:通道采样时间为 56 个周期<br>100:通道采样时间为 84 个周期<br>101:通道采样时间为 112 个周期<br>110:通道采样时间为 144 个周期<br>111:通道采样时间为 480 个周期 |

### 14.7.5. 采样时间寄存器 1 (ADC_SAMPT1)

地址偏移：0x10
复位值：0x0000 0000

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:30 | 保留 | 必须保持复位值。 |
| 29:27 | SPT9[2:0] | 参考 SPT0[2:0]的描述 |
| 26:24 | SPT8[2:0] | 参考 SPT0[2:0]的描述 |
| 23:21 | SPT7[2:0] | 参考 SPT0[2:0]的描述 |
| 20:18 | SPT6[2:0] | 参考 SPT0[2:0]的描述 |
| 17:15 | SPT5[2:0] | 参考 SPT0[2:0]的描述 |

| | | |
|---|---|---|
| 14:12 | SPT4[2:0] | 参考 SPT0[2:0]的描述 |
| 11:9 | SPT3[2:0] | 参考 SPT0[2:0]的描述 |
| 8:6 | SPT2[2:0] | 参考 SPT0[2:0]的描述 |
| 5:3 | SPT2[2:0] | 参考 SPT0[2:0]的描述 |
| 2:0 | SPT0[2:0] | 通道采样时间<br>000:通道采样时间为 3 个周期<br>001:通道采样时间为 15 个周期<br>010:通道采样时间为 28 个周期<br>011:通道采样时间为 56 个周期<br>100:通道采样时间为 84 个周期<br>101:通道采样时间为 112 个周期<br>110:通道采样时间为 144 个周期<br>111:通道采样时间为 480 个周期 |

### 14.7.6. 看门狗高阈值寄存器 (ADC_WDHT)

地址偏移：0x24
复位值：0x0000 0FFF

该寄存器只能按字(32位)访问

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

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

| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11:0 | WDHT[11:0] | 模拟看门狗高侧阈值<br>这些位定义了模拟看门狗的高侧阈值。 |

### 14.7.7. 看门狗低阈值寄存器 (ADC_WDLT)

地址偏移：0x28
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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

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

| 保留 | WDLT[11:0] |
|---|---|
| | rw |

| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11:0 | WDLT[11:0] | 模拟看门狗低侧阈值<br>这些位定义了模拟看门狗的低侧阈值。 |

### 14.7.8. 常规序列寄存器 0 (ADC_RSQ0)

地址偏移：0x2C
复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | RL[3:0] | | | | RSQ15[4:1] | | | |
| | | | | | | | | rw | | | | rw | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RSQ15[0] | RSQ14[4:0] | | | | | RSQ13[4:0] | | | | | RSQ12[4:0] | | | | |
| rw | rw | | | | | rw | | | | | rw | | | | |

| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值。 |
| 23:20 | RL[3:0] | 常规序列长度<br>常规序列中的总的通道数目为 RL[3:0]+1。 |
| 19:15 | RSQ15[4:0] | 参考 RSQ0[4:0]的描述 |
| 14:10 | RSQ14[4:0] | 参考 RSQ0[4:0]的描述 |
| 9:5 | RSQ13[4:0] | 参考 RSQ0[4:0]的描述 |
| 4:0 | RSQ12[4:0] | 参考 RSQ0[4:0]的描述 |

### 14.7.9. 常规序列寄存器 1 (ADC_RSQ1)

地址偏移：0x30
复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | RSQ11[4:0] | | | | | | RSQ10[4:0] | | | | RSQ9[4:1] | | | |
| | | rw | | | | | | rw | | | | rw | | | |

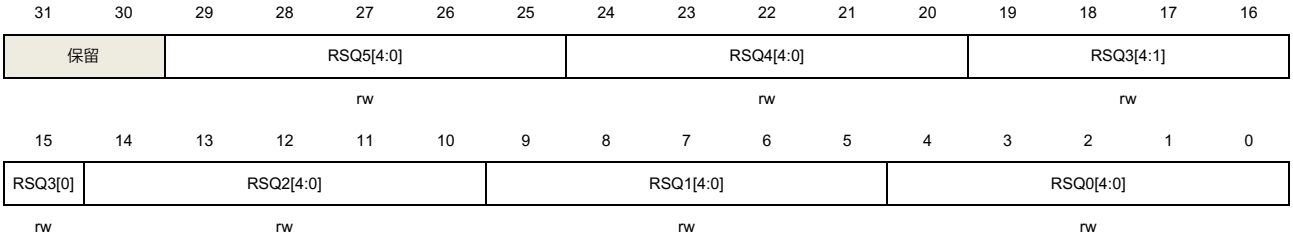
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RSQ9[0] | RSQ8[4:0] | | | | | RSQ7[4:0] | | | | | RSQ6[4:0] | | | | |
| rw | rw | | | | | rw | | | | | rw | | | | |

| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:30 | 保留 | 必须保持复位值。 |
| 29:25 | RSQ11[4:0] | 参考 RSQ0[4:0]的描述 |
| 24:20 | RSQ10[4:0] | 参考 RSQ0[4:0]的描述 |
| 19:15 | RSQ9[4:0] | 参考 RSQ0[4:0]的描述 |
| 14:10 | RSQ8[4:0] | 参考 RSQ0[4:0]的描述 |
| 9:5 | RSQ7[4:0] | 参考 RSQ0[4:0]的描述 |
| 4:0 | RSQ6[4:0] | 参考 RSQ0[4:0]的描述 |

### 14.7.10. 常规序列寄存器 2 (ADC_RSQ2)

地址偏移：0x34
复位值：0x0000 0000

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:30 | 保留 | 必须保持复位值。 |
| 29:25 | RSQ5[4:0] | 参考 RSQ0[4:0]的描述 |
| 24:20 | RSQ4[4:0] | 参考 RSQ0[4:0]的描述 |
| 19:15 | RSQ3[4:0] | 参考 RSQ0[4:0]的描述 |
| 14:10 | RSQ2[4:0] | 参考 RSQ0[4:0]的描述 |
| 9:5 | RSQ1[4:0] | 参考 RSQ0[4:0]的描述 |
| 4:0 | RSQ0[4:0] | 通道编号(0..18)写入这些位来选择常规通道的第 n 个转换的通道 |

### 14.7.11. 常规数据寄存器 (ADC_RDATA)

地址偏移：0x4C
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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

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

| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | RDATA[15:0] | 常规通道数据<br>这些位包含了常规通道的转换结果，只读。 |

## 14.7.12. 过采样控制寄存器（ADC_OVSAMPCTL)

地址偏移：0x80
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | TOVS | OVSS[3:0] | | | | OVSR[2:0] | | | 保留 | OVSEN |
| | | | | | | rw | rw | | | | rw | | | | rw |

| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:10 | 保留 | 必须保持复位值。 |
| 9 | TOVS | 过采样触发<br>该位通过软件置位和清除。<br>0：在一次触发后连续执行过采样通道的所有转换<br>1：对于过采样通道的每次转换都需要一次触发，触发次数由过采样率(OVSR[2:0])决定。<br>注意：只有在 ADCON=0 的时候才允许通过软件对该位进行写(确保没有转换正在执行)。 |
| 8:5 | OVSS[3:0] | 过采样移位<br>该位通过软件置位和清除。<br>0000：不移位<br>0001：移 1 位<br>0010：移 2 位<br>0011：移 3 位<br>0100：移 4 位<br>0101：移 5 位<br>0110：移 6 位 |

0111：移 7 位
1000：移 8 位
其余值都保留
注意： 只有在 ADCON=0 的时候才允许通过软件对该位进行写(确保没有转换正在执行)。

| | | |
|---|---|---|
| 4:2 | OVSR[2:0] | 过采样率<br>这些位定义了过采样率的大小。<br>000：2x<br>001：4x<br>010：8x<br>011：16x<br>100：32x<br>101：64x<br>110：128x<br>111：256x<br>注意：只有在 ADCON=0 的时候才允许通过软件对该位进行写(确保没有转换正在执行)。 |
| 1 | 保留 | 必须保持复位值. |
| 0 | OVSEN | 过采样使能<br>该位通过软件置位和清除。<br>0：过采样失能<br>1：过采样使能<br>注意：只有在 ADCON=0 的时候才允许通过软件对该位进行写(确保没有转换正在执行)。 |

## 14.7.13. 摘要状态寄存器 (ADC_SSTAT)

地址偏移：0x300
复位值：0x0000 0000

该寄存器只能按字(32位)访问

该寄存器是只读的，提供了3个ADC状态的摘要。这个寄存器在ADC1和ADC2中不可用。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | ROVF2 | STRC2 | 保留 | | EOC2 | WDE2 |
| | | | | | | | | | | r | r | | | r | r |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | ROVF1 | STRC1 | 保留 | | EOC1 | WDE1 | 保留 | | ROVF0 | STRC0 | 保留 | | EOC0 | WDE0 |
| | | r | r | | | r | r | | | r | r | | | r | r |

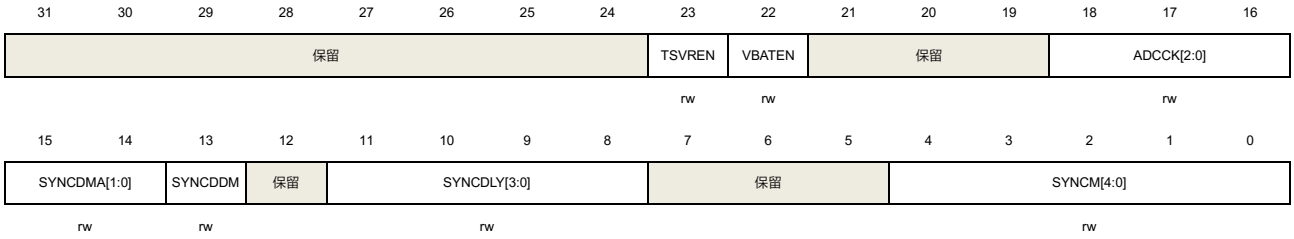
| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:22 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 21 | ROVF2 | 该位是 ADC2 的 ROVF 的镜像 |
| 20 | STRC2 | 该位是 ADC2 的 STRC 的镜像 |
| 19:18 | 保留 | 必须保持复位值。 |
| 17 | EOC2 | 该位是 ADC2 的 EOC 的镜像 |
| 16 | WDE2 | 该位是 ADC2 的 WDE 的镜像 |
| 15:14 | 保留 | 必须保持复位值。 |
| 13 | ROVF1 | 该位是 ADC1 的 ROVF 的镜像 |
| 12 | STRC1 | 该位是 ADC1 的 STRC 的镜像 |
| 11:10 | 保留 | 必须保持复位值。 |
| 9 | EOC1 | 该位是 ADC1 的 EOC 的镜像 |
| 8 | WDE1 | 该位是 ADC1 的 WDE 的镜像 |
| 7:6 | 保留 | 必须保持复位值。 |
| 5 | ROVF0 | 该位是 ADC0 的 ROVF 的镜像 |
| 4 | STRC0 | 该位是 ADC0 的 STRC 的镜像 |
| 3:2 | 保留 | 必须保持复位值。 |
| 1 | EOC0 | 该位是 ADC0 的 EOC 的镜像 |
| 0 | WDE0 | 该位是 ADC0 的 WDE 的镜像 |

### 14.7.14. 同步控制寄存器 (ADC_SYNCCTL)

地址偏移：0x304
复位值：0x0000 0000

该寄存器只能按字(32位)访问
这个寄存器在ADC1和ADC2中不可用




| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值。 |
| 23 | TSVREN | 使能 ADC0 的通道 16 (温度传感器) 和 通道 17 (内部参考电压)。<br>0：ADC0 的通道 16 和通道 17 失能 |

| | | |
|---|---|---|
| | | 1：ADC0 的通道 16 和通道 17 使能 |
| 22 | VBATEN | 使能 ADC0 的通道 18 (外部电池电压的 1/4)<br>0：ADC0 的通道 18 失能<br>1：ADC0 的通道 18 使能 |
| 21:19 | 保留 | 必须保持复位值. |
| 18:16 | ADCCK[2:0] | ADC 时钟<br>这些位配置所有 ADC 的时钟<br>000：PCLK2 2 分频<br>001：PCLK2 4 分频<br>010：PCLK2 6 分频<br>011：PCLK2 8 分频<br>100：HCLK 5 分频<br>101：HCLK 6 分频<br>110：HCLK 10 分频<br>111：HCLK 20 分频 |
| 15:14 | SYNCDMA[1:0] | ADC 同步 DMA 模式选择<br>00：ADC 同步 DMA 失能:<br>01：ADC 同步 DMA 模式 0<br>10：ADC 同步 DMA 模式 1<br>11：保留 |
| 13 | SYNCDDM | ADC 同步 DMA 使能模式<br>该位配置 ADC 同步模式时 DMA 失能模式<br>0：当检测到来自 DMA 控制器的传输结束信号后，DMA 机制失能<br>1：当 SYNCDMA 不为 00 时，根据 SYNCDMA 位来产生 DMA 请求。 |
| 12 | 保留 | 必须保持复位值。 |
| 11:8 | SYNCDLY[3:0] | ADC 同步延迟<br>在 ADC 同步模式中，这些位用于配置两个采样阶段之间的延迟为<br>(5+SYNCDLY)ADC 时钟周期。 |
| 7:5 | 保留 | 必须保持复位值。 |
| 4:0 | SYNCM[4:0] | ADC 同步模式<br>当 ADC 同步模式已经使能，如果要将同步模式修改为其他值，必须先将这些位设置为 00000<br>00000：ADC 同步模式失能。所有的 ADC 都独立工作。<br>00110：ADC0 和 ADC1 工作在常规并行模式。ADC2 独立工作。<br>00111：ADC0 和 ADC1 工作在常规跟随模式。ADC2 独立工作。<br>10110：所有的 ADC 都工作在常规并行模式。<br>10111：所有的 ADC 都工作在常规跟随模式。<br>其他值保留。 |

### 14.7.15. 同步常规数据寄存器 (ADC_SYNCDATA)

地址偏移：0x308
复位值：0x0000 0000

该寄存器只能按字(32位)访问

这个寄存器在ADC1和ADC2中不可用。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SYNCDATA1[15:0] | | | | | | | | | | | | | | | |
| r | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SYNCDATA0[15:0] | | | | | | | | | | | | | | | |
| r | | | | | | | | | | | | | | | |

| 位/位域 | Fields | 说明 |
|---|---|---|
| 31:16 | SYNCDATA1[15:0] | ADC 同步模式中，常规数据 2 |
| 15:0 | SYNCDATA0[15:0] | ADC 同步模式中，常规数据 1 |

# 15. 数模转换器(DAC)

## 15.1. 简介

数字/模拟转换器可以将 12 位的数字数据转换为外部引脚上的电压输出。数据可以采用 8 位或 12 位模式，左对齐或右对齐模式。当使能了外部触发，DMA 可被用于更新输入端数字数据。

在输出电压时，可以利用 DAC 输出缓冲区来获得更高的驱动能力。

每个 DAC 的两个通道可以独立或并发工作。

## 15.2. 主要特征

DAC 的主要特征如下：

- 8 位或 12 位分辨率；
- 数据左对齐或右对齐；
- DMA 功能与欠载检测；
- 同步更新转换；
- 外部事件触发转换；
- 可配置的内部缓冲区；
- 输入参考电压 $V_{REFP}$；
- 噪声波生成(LSFR 噪声模式和三角噪声模式)；
- DACx 双通道并发模式；

*图 15-1. DAC 结构框图*为 DAC 的结构框图，*表 15-1. DAC 引脚*给出了引脚描述。

**图 15-1. DAC 结构框图**

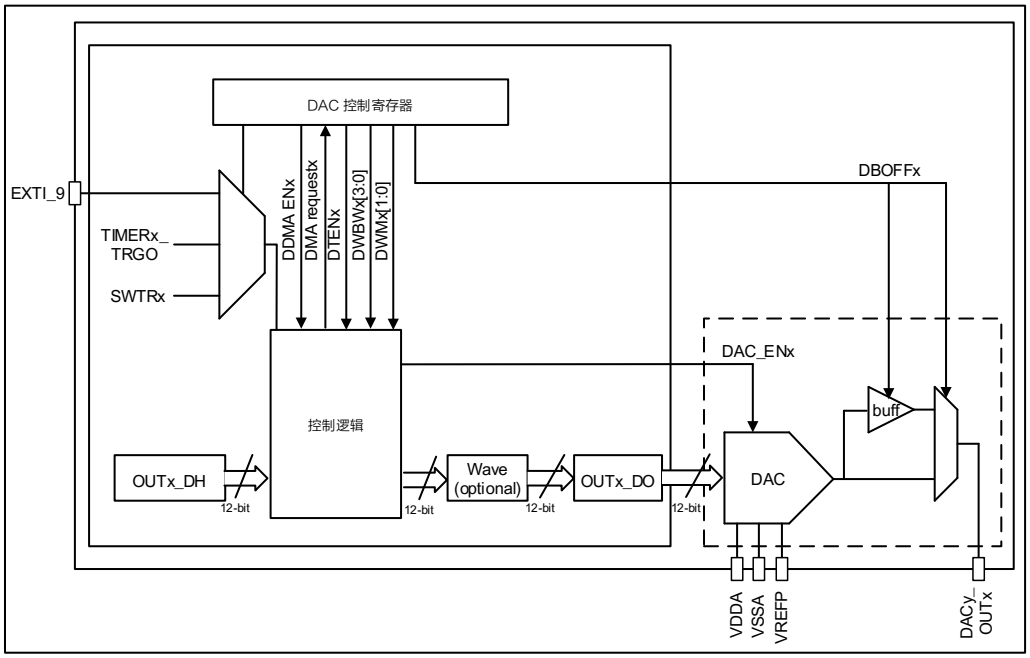


**表 15-1. DAC 引脚**

| 名称 | 描述 | 信号类型 |
|---|---|---|
| $V_{DDA}$ | 模拟电源 | 输入，模拟电源 |
| $V_{SSA}$ | 模拟电源地 | 输入，模拟电源地 |
| $V_{REFP}$ | DAC 正参考电压 | 输入，模拟正参考电压 |
| DACy_OUTx | DAC 模拟输出 | 模拟输出信号 |

下表详细列出了 DAC 的触发与输出信号。

**表 15-2. DAC 触发与输出**

| | DAC0 | |
|---|---|---|
| **通道** | 通道 0 | 通道 1 |
| **DAC 输出 I/O** | PA4 | PA5 |
| **DAC 输出 BUFFER 功能** | • | • |
| **软件触发功能** | • | |
| **EXTI 触发信号** | EXTI_9 | |
| **TIMER 触发信号** | TIMER1_TRGO<br>TIMER3_TRGO<br>TIMER4_TRGO<br>TIMER5_TRGO<br>TIMER6_TRGO<br>TIMER7_TRGO | |

**注意：**在使能 DAC 模块前，GPIO 口(DAC 输出 I/O)应配置为模拟模式。

## 15.3. 功能描述

### 15.3.1. DAC 使能

将 DAC_CTL0 寄存器中的 DENx 位置 1，可以给 DAC 模块上电，DAC 子模块完全启动需要等待 $t_{WAKEUP}$ 时间。

### 15.3.2. DAC 输出缓冲

为了降低输出阻抗，并在没有外部运算放大器的情况下驱动外部负载，每个 DAC 模块内部各集成了一个输出缓冲区。

缺省情况下，输出缓冲区是开启的，可以通过设置 DAC_CTL0 寄存器的 DBOFFx 位来开启或关闭缓冲区。

### 15.3.3. DAC 数据配置

对于 12 位的 DAC 保持数据(OUTx_DH)，可以通过对 DAC_OUTx_R12DH 、

DAC_OUTx_L12DH 和 DAC_OUTx_R8DH 中的任意一个寄存器写入数据来配置。当数据被加载到 DAC_OUTx_R8DH 寄存器时，只有 8 位最高有效位是可配置，4 位最低有效位被强制置为 4'b0000。

### 15.3.4. DAC 触发

DAC 可以通过软件或者外部信号的上升沿触发。外部触发可以通过设置 DAC_CTL0 寄存器中 DTENx 位来使能。触发源可以通过 DAC_CTL0 寄存器中 DTSELx 位来进行选择，如*表 15-3. DAC 外部触发*所示。

**表 15-3. DAC 外部触发**

| DTSELx[2:0] | 触发源 | 触发类型 |
|---|---|---|
| 3b'000 | TIMER5_TRGO | 硬件触发 |
| 3b'001 | TIMER7_TRGO | |
| 3b'010 | TIMER6_TRGO | |
| 3b'011 | TIMER4_TRGO | |
| 3b'100 | TIMER1_TRGO | |
| 3b'101 | TIMER3_TRGO | |
| 3b'110 | EXTI_9 | |
| 3b'111 | SWTR | 软件触发 |

TIMERx_TRGO 信号是由定时器生成的，而软件触发是通过设置 DAC_SWT 寄存器的 SWTRx 位生成的。

### 15.3.5. DAC 转换

如果使能了外部触发(通过设置 DAC_CTL0 寄存器的 DTENx 位)，当已经选择的触发事件发生，DAC 保持数据(OUTx_DH)会被转移到 DAC 数据输出寄存器(DAC_OUTx_DO)。而在外部触发未使能的情况下，DAC 保持数据(OUTx_DH)会被自动转移到 DAC 数据输出寄存器(DAC_OUTx_DO)。

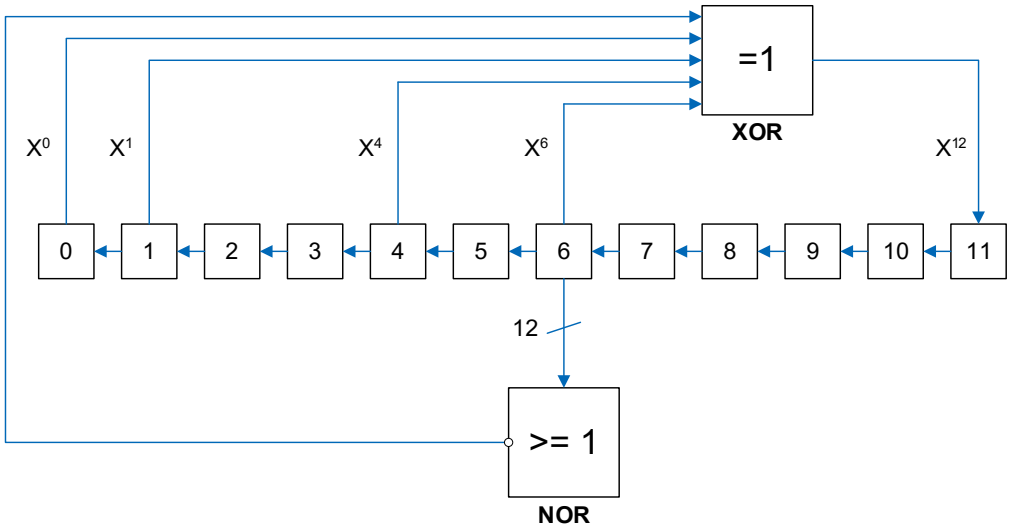
当 DAC 保持数据(OUTx_DH)加载到 DAC_OUTx_DO 寄存器时，经过 $t_{SETTLING}$ 时间之后，模拟输出变得有效，$t_{SETTLING}$ 的值与电源电压和模拟输出负载有关。

### 15.3.6. DAC 噪声波

有两种方式可以将噪声波加载到 DAC 输出数据：LFSR 噪声波和三角波。噪声波模式可以通过 DAC_CTL0 寄存器的 DWMx 位来进行选择。噪声的幅值可以通过配置 DAC_CTL0 寄存器的 DAC 噪声波位宽(DWBWx)位来进行设置。

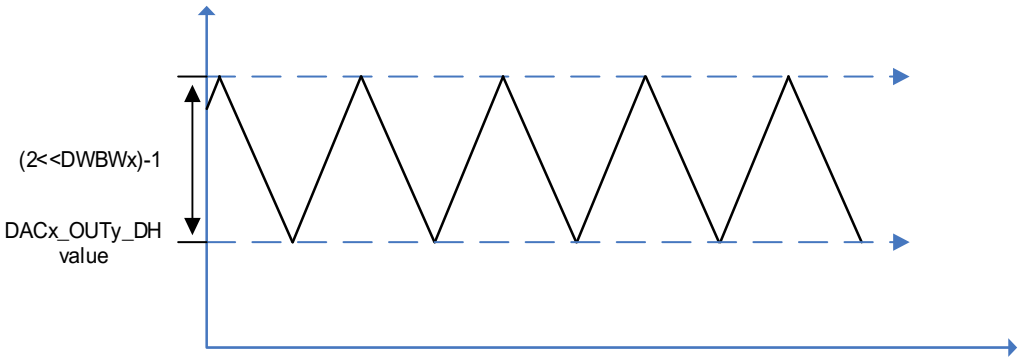
LFSR 噪声模式：在 DAC 控制逻辑中有一个线性反馈移位寄存器(LFSR)。在此模式下，LFSR 的值与 OUTx_DH 值相加后，被写入到 DAC 数据输出寄存器(DAC_OUTx_DO)。当配置的 DAC 噪声波位宽小于 12 时，LFSR 的值等于 LFSR 寄存器最低的 DWBWx 位，高位被屏蔽。

**图 15-2. DAC LFSR 算法**




三角噪声模式：三角波幅值与 OUTx_DH 值相加后，被写入到 DAC 数据输出寄存器(DAC_OUTx_DO)。三角波幅值的最小值为 0，最大值为(2 << DWBWx) - 1。

**图 15-3. DAC 三角噪声模式生成的波形**




## 15.3.7. DAC 输出电压

DAC 引脚上的模拟输出电压取决于下面的等式：

$$V_{DAC_OUT}=V_{REFP}*OUTx_DO/4096 \quad (15\text{-}1)$$

数字输入被线性地转换成模拟输出电压，输出范围为 0 到 $V_{REFP}$。

## 15.3.8. DMA 请求

在外部触发使能的情况下，通过设置 DAC_CTL0 寄存器的 DDMAENx 位来使能 DMA 请求。当有外部硬件触发的时候(不是软件触发)，则产生一个 DMA 请求。

如果在前一个请求响应之前第二个外部触发到达，则不响应新到的触发请求，并且发生欠载错误事件。DAC_STAT0 寄存器中的 DDUDRx 位置 1，如果 DAC_CTL0 寄存器中的 DDUDRIEx

位置 1，则会产生中断。

### 15.3.9. DAC 并发转换

当 DAC 的两个通道同时工作时，为了在特定应用中最大限度利用总线带宽，DAC 的两个通道可以被配置为并发模式。在并发模式中，DAC 的 OUTx_DH 和 OUTx_DO 值将同时被更新。

有 3 个并发寄存器可用于加载 OUTx_DH 的值，分别是：DACC_R8DH、DACC_R12DH 和 DACC_L12DH 寄存器，配置其中的任意一个寄存器都可实现同时驱动 DAC 的两个通道。

当使能了外部触发时，DAC 两个通道的 DTENx 位都需要置 1，需要配置 DTSEL0/1 相同来保证同时触发。

当使能了 DMA 功能时，DAC 任一通道的 DDMAENx 位置 1 即可。

噪声模式和噪声位宽可以根据使用情况配置为相同或不同。

## 15.4. DAC 寄存器

DAC0 基地址：0x4000 7400

### 15.4.1. DACx 控制寄存器(DAC_CTL0)

地址偏移：0x00
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | DDUDRIE1 | DDMAEN1 | DWBW1[3:0] | | | | DWM1[1:0] | | DTSEL1[2:0] | | | DTEN1 | DBOFF1 | DEN1 |
| | | rw | rw | rw | | | | rw | | rw | | | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | DDUDRIE0 | DDMAEN0 | DWBW0[3:0] | | | | DWM0[1:0] | | DTSEL0[2:0] | | | DTEN0 | DBOFF0 | DEN0 |
| | | rw | rw | rw | | | | rw | | rw | | | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:30 | 保留 | 必须保持复位值 |
| 29 | DDUDRIE1 | DACx_OUT1 DMA 欠载中断使能<br>0: DACx_OUT1 DMA 欠载中断禁能<br>1: DACx_OUT1 DMA 欠载中断使能 |
| 28 | DDMAEN1 | DACx_OUT1 DMA 使能<br>0：DACx_OUT1 DMA 模式禁能<br>1：DACx_OUT1 DMA模式使能 |
| 27:24 | DWBW1[3:0] | DACx_OUT1 噪声波位宽<br>这些位指定了 DACx_OUT1 的噪声波信号的位宽。LFSR 噪声模式下，这些位表示不屏蔽 LFSR 的位[n-1, 0]；三角噪声模式下，这些位表示三角波幅值为(2<<(n-1))-1。其中，n 为噪声波位宽。<br>0000：波形信号的位宽为 1<br>0001：波形信号的位宽为 2<br>0010：波形信号的位宽为 3<br>0011：波形信号的位宽为 4<br>0100：波形信号的位宽为 5<br>0101：波形信号的位宽为 6<br>0110：波形信号的位宽为 7<br>0111：波形信号的位宽为 8<br>1000：波形信号的位宽为 9<br>1001：波形信号的位宽为 10<br>1010：波形信号的位宽为 11 |

| | | |
|---|---|---|
| | | >=1011：波形信号的位宽为12 |
| 23:22 | DWM1[1:0] | DACx_OUT1 噪声波模式<br>这些位指定了在 DACx_OUT1 外部触发使能(DTEN1=1)的情况下，DACx_OUT1 的噪声波模式的选择。<br>00：波形生成禁能<br>01：LFSR 噪声模式<br>1x：三角噪声模式 |
| 21:19 | DTSEL1[2:0] | DACx_OUT1 触发选择<br>这些位仅在 DTEN1=1 并选择用于触发 DAC 的外部事件时使用。<br>000：TIMER5 TRGO<br>001：TIMER7 TRGO<br>010：TIMER6 TRGO<br>011：TIMER4 TRGO<br>100：TIMER1 TRGO<br>101：TIMER3 TRGO<br>110：外部中断线9<br>111：软件触发 |
| 18 | DTEN1 | DACx_OUT1 触发使能<br>0：DACx_OUT1 触发禁能<br>1：DACx_OUT1触发使能 |
| 17 | DBOFF1 | DACx_OUT1输出缓冲区关闭<br>0：DACx_OUT1输出缓冲区打开，以降低输出阻抗，提高驱动能力<br>1：DACx_OUT1输出缓冲区关闭 |
| 16 | DEN1 | DACx_OUT1 使能<br>0：DACx_OUT1 禁能<br>1：DACx_OUT1使能 |
| 15:14 | 保留 | 必须保持复位值 |
| 13 | DDUDRIE0 | DACx_OUT0 DMA 欠载中断使能<br>0: DACx_OUT0 DMA 欠载中断禁能<br>1: DACx_OUT0 DMA 欠载中断使能 |
| 12 | DDMAEN0 | DACx_OUT0 DMA 使能<br>0：DACx_OUT0 DMA 模式禁能<br>1：DACx_OUT0 DMA模式使能 |
| 11:8 | DWBW0[3:0] | DACx_OUT0 噪声波位宽<br>这些位指定了 DACx_OUT0 的噪声波信号的位宽。LFSR 噪声模式下，这些位表示不屏蔽 LFSR 的位[n-1, 0]；三角噪声模式下，这些位表示三角波幅值为(2<<(n-1))-1。其中，n 为噪声波位宽。<br>0000：波形信号的位宽为 1<br>0001：波形信号的位宽为 2<br>0010：波形信号的位宽为 3 |

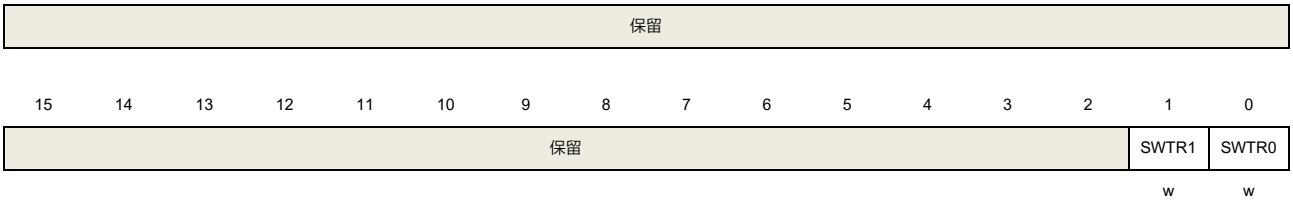
| | | |
|---|---|---|
| | | 0011：波形信号的位宽为 4<br>0100：波形信号的位宽为 5<br>0101：波形信号的位宽为 6<br>0110：波形信号的位宽为 7<br>0111：波形信号的位宽为 8<br>1000：波形信号的位宽为 9<br>1001：波形信号的位宽为 10<br>1010：波形信号的位宽为 11<br>>=1011：波形信号的位宽为12 |
| 7:6 | DWM0[1:0] | DACx_OUT0 噪声波模式<br>这些位指定了在 DACx_OUT0 外部触发使能(DTEN0=1)的情况下，DACx_OUT0 的噪声波模式的选择。<br>00：波形生成禁能<br>01：LFSR 噪声模式<br>1x：三角噪声模式 |
| 5:3 | DTSEL0[2:0] | DACx_OUT0 触发选择<br>这些位仅在 DTEN0=1 并选择用于触发 DAC 的外部事件时使用。<br>000：TIMER5 TRGO<br>001：TIMER7 TRGO<br>010：TIMER6 TRGO<br>011：TIMER4 TRGO<br>100：TIMER1 TRGO<br>101：TIMER3 TRGO<br>110：外部中断线9<br>111：软件触发 |
| 2 | DTEN0 | DACx_OUT0 触发使能<br>0：DACx_OUT0 触发禁能<br>1：DACx_OUT0触发使能 |
| 1 | DBOFF0 | DACx_OUT0输出缓冲区关闭<br>0：DACx_OUT0输出缓冲区打开，以降低输出阻抗，提高驱动能力<br>1：DACx_OUT0输出缓冲区关闭 |
| 0 | DEN0 | DACx_OUT0 使能<br>0：DACx_OUT0 禁能<br>1：DACx_OUT0使能 |

### 15.4.2. DACx 软件触发寄存器(DAC_SWT)

地址偏移：0x04
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

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

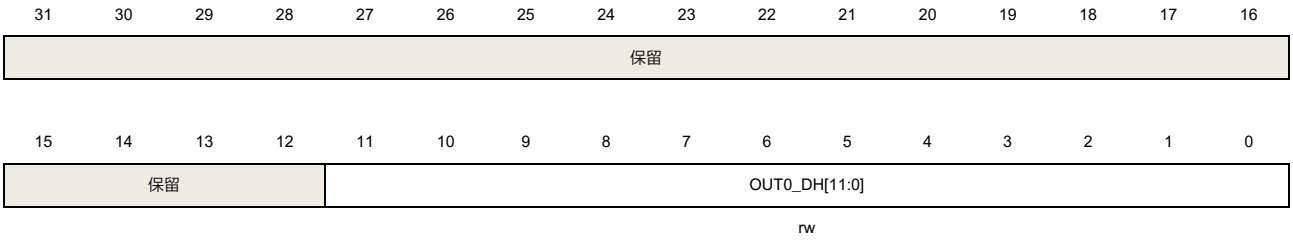



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值 |
| 1 | SWTR1 | DACx_OUT1 软件触发，由硬件清除。<br>0：软件触发禁能<br>1：软件触发使能 |
| 0 | SWTR0 | DACx_OUT0 软件触发，由硬件清除。<br>0：软件触发禁能<br>1：软件触发使能 |

### 15.4.3. DACx_OUT0 12 位右对齐数据保持寄存器(DAC_OUT0_R12DH)

地址偏移：0x08
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

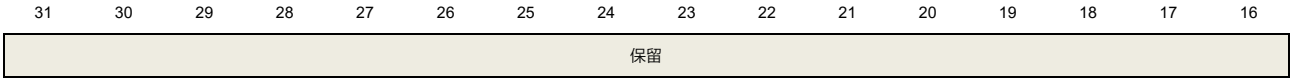


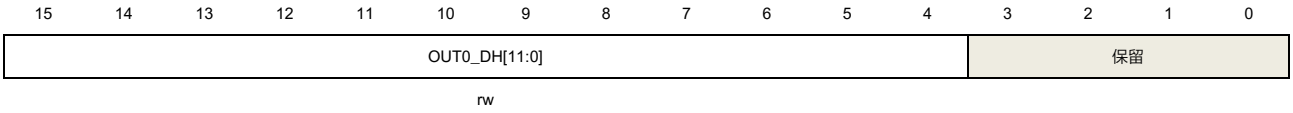

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值 |
| 11:0 | OUT0_DH[11:0] | DACx_OUT0 12 位右对齐数据<br>这些位指定了将由 DACx_OUT0 转换的数据。 |

### 15.4.4. DACx_OUT0 12 位左对齐数据保持寄存器(DAC_OUT0_L12DH)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。



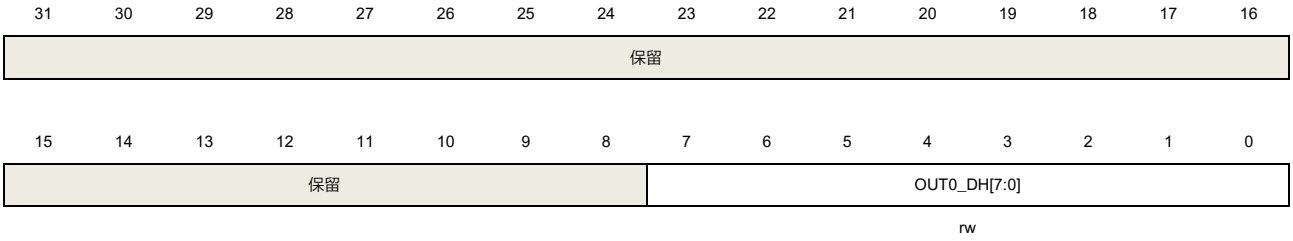



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15:4 | OUT0_DH[11:0] | DACx_OUT0 12 位左对齐数据<br>这些位指定了将由 DACx_OUT0 转换的数据。 |
| 3:0 | 保留 | 必须保持复位值 |

### 15.4.5. DACx_OUT0 8 位右对齐数据保持寄存器(DAC_OUT0_R8DH)

地址偏移：0x10
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

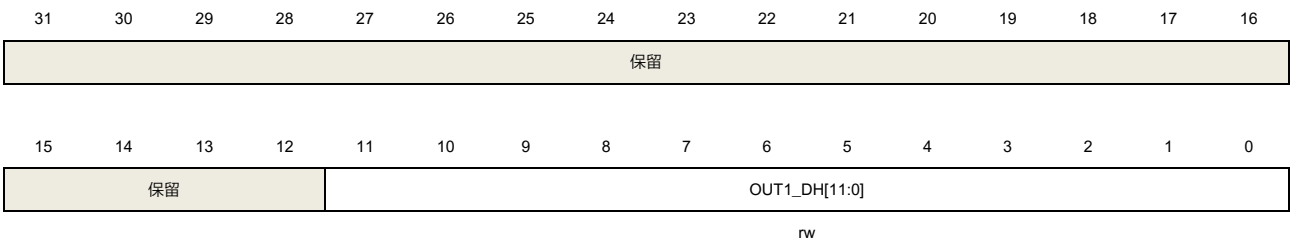



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值 |
| 7:0 | OUT0_DH[7:0] | DACx_OUT0 8 位右对齐数据<br>这些位指定了将由 DACx_OUT0 转换的数据的最高 8 位有效位。 |

### 15.4.6. DACx_OUT1 12 位右对齐数据保持寄存器(DAC_OUT1_R12DH)

地址偏移：0x14
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

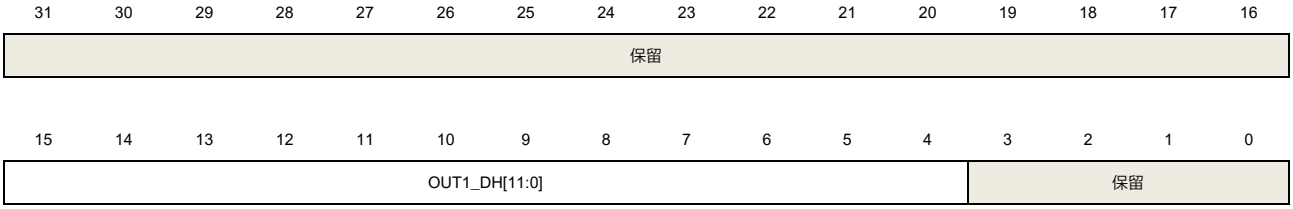


| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值 |
| 11:0 | OUT1_DH[11:0] | DACx_OUT1 12 位右对齐数据<br>这些位指定了将由 DACx_OUT1 转换的数据。 |

### 15.4.7. DACx_OUT1 12 位左对齐数据保持寄存器(DAC_OUT1_L12DH)

地址偏移：0x18
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。



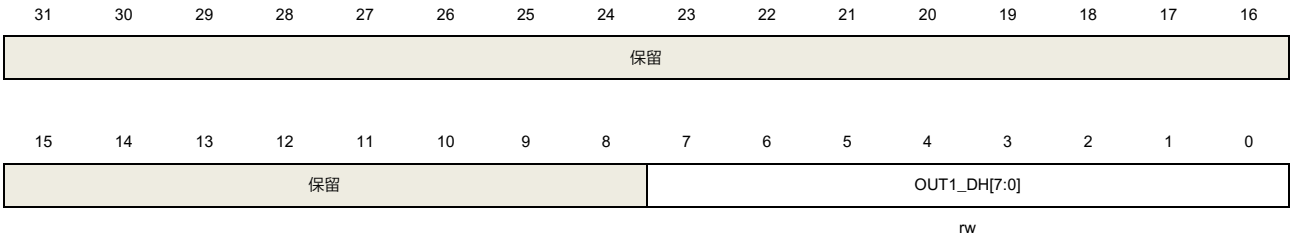

rw

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15:4 | OUT1_DH[11:0] | DACx_OUT1 12 位左对齐数据<br>这些位指定了将由 DACx_OUT1 转换的数据。 |
| 3:0 | 保留 | 必须保持复位值 |

### 15.4.8. DACx_OUT1 8 位右对齐数据保持寄存器(DAC_OUT1_R8DH)

地址偏移：0x1C
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。




rw

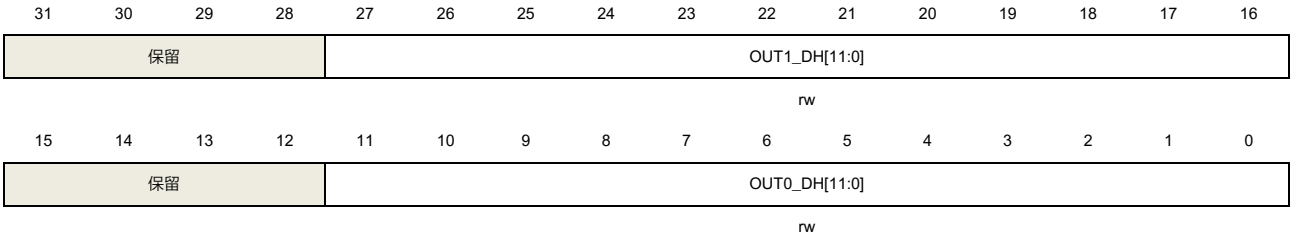
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值 |
| 7:0 | OUT1_DH[7:0] | DACx_OUT1 8 位右对齐数据 |

这些位指定了将由 DACx_OUT1 转换的数据的 8 位最高有效位。

### 15.4.9. DACx 并发模式 12 位右对齐数据保持寄存器(DACC_R12DH)

地址偏移：0x20
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

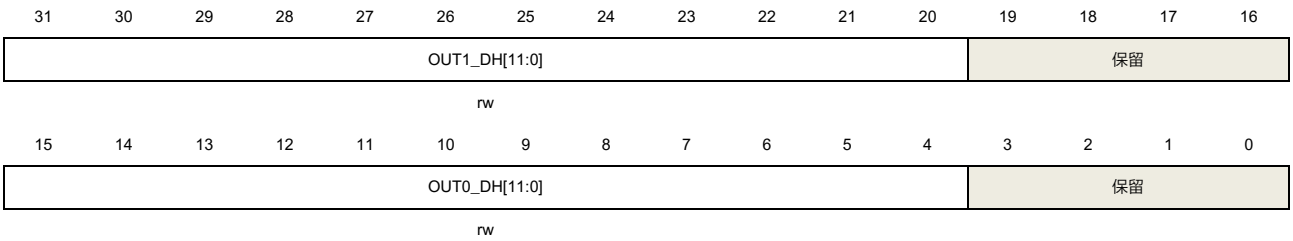



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值 |
| 27:16 | OUT1_DH[11:0] | DACx_OUT1 12 位右对齐数据<br>这些位指定了将由 DACx_OUT1 转换的数据。 |
| 15:12 | 保留 | 必须保持复位值 |
| 11:0 | OUT0_DH[11:0] | DACx_OUT0 12 位右对齐数据<br>这些位指定了将由 DACx_OUT0 转换的数据。 |

### 15.4.10. DACx 并发模式 12 位左对齐数据保持寄存器(DACC_L12DH)

地址偏移：0x24
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:20 | OUT1_DH[11:0] | DACx_OUT1 12 位左对齐数据<br>这些位指定了将由 DACx_OUT1 转换的数据。 |
| 19:16 | 保留 | 必须保持复位值 |

| | | |
|---|---|---|
| 15:4 | OUT0_DH[11:0] | DACx_OUT0 12 位左对齐数据<br>这些位指定了将由 DACx_OUT0 转换的数据。 |
| 3:0 | 保留 | 必须保持复位值 |

### 15.4.11. DACx 并发模式 8 位右对齐数据保持寄存器(DACC_R8DH)

地址偏移：0x28
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

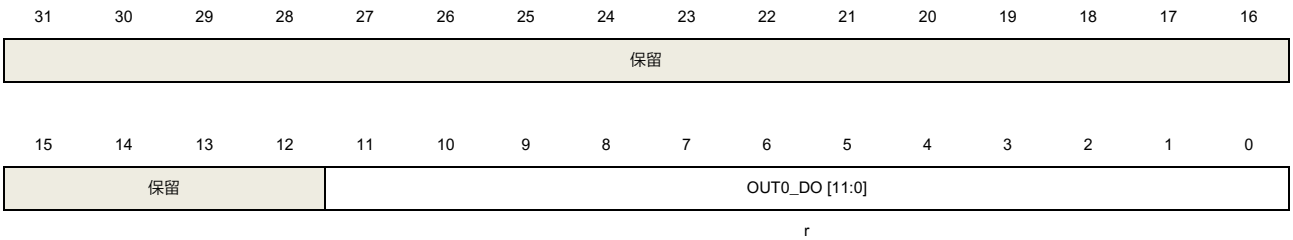



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15:8 | OUT1_DH[7:0] | DACx_OUT1 8 位右对齐数据<br>这些位指定了将由 DACx_OUT1 转换的数据的 8 位最高有效位。 |
| 7:0 | OUT0_DH[7:0] | DACx_OUT0 8 位右对齐数据<br>这些位指定了将由 DACx_OUT0 转换的数据的 8 位最高有效位。 |

### 15.4.12. DACx_OUT0 数据输出寄存器(DAC_OUT0_DO)

地址偏移：0x2C
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。



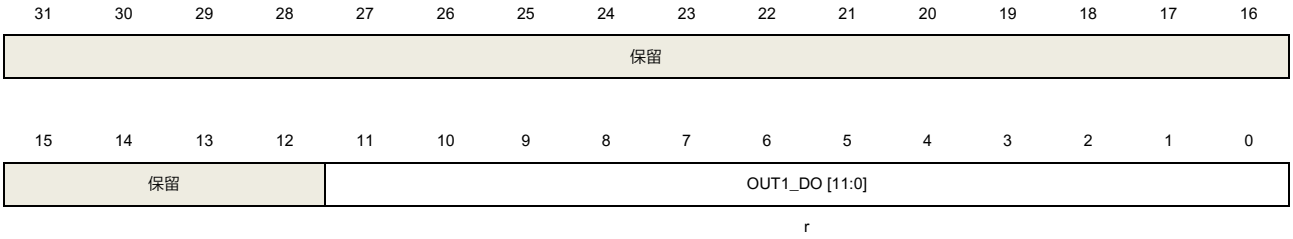

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值 |
| 11:0 | OUT0_DO [11:0] | DACx_OUT0 数据输出。 |

这些位为只读类型，存储由 DACx_OUT0 转换的数据。

### 15.4.13. DACx_OUT1 数据输出寄存器(DAC_OUT1_DO)

地址偏移：0x30
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。

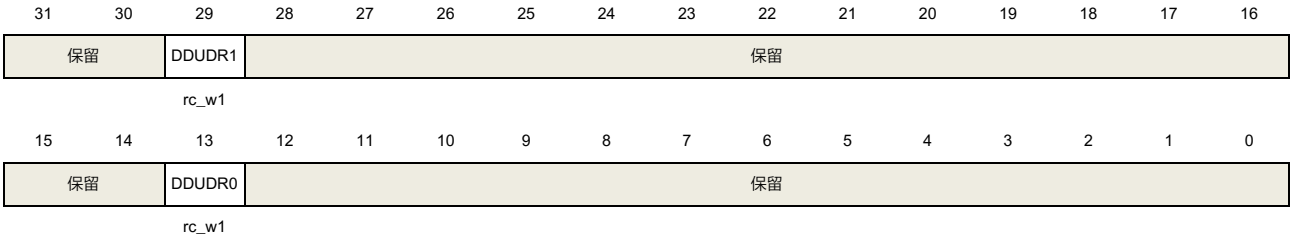



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值 |
| 11:0 | OUT1_DO [11:0] | DACx_OUT1 数据输出。<br>这些位为只读类型，存储由 DACx_OUT1 转换的数据。 |

### 15.4.14. DACx 状态寄存器 0(DAC_STAT0)

地址偏移：0x34
复位值：0x0000 0000

该寄存器只能按字(32 位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:30 | 保留 | 必须保持复位值。 |
| 29 | DDUDR1 | DACx_OUT1 DMA 欠载标志位，硬件置位，软件写 1 清零。<br>0：没有欠载发生<br>1：发生欠载(DAC 触发产生速度快于 DMA 传输速度) |
| 28:14 | 保留 | 必须保持复位值。 |
| 13 | DDUDR0 | DACx_OUT0 DMA 欠载标志位，硬件置位，软件写 1 清零。<br>0：没有欠载发生 |

1：发生欠载(DAC 触发产生速度快于 DMA 传输速度)

| | | |
|---|---|---|
| 12:0 | 保留 | 必须保持复位值。 |

# 16. 看门狗定时器(WDGT)

看门狗定时器(WDGT)是一个硬件计时电路，用来监测由软件故障导致的系统故障。片上有两个看门狗定时器外设，独立看门狗定时器(FWDGT)和窗口看门狗定时器(WWDGT)。它们使用灵活，并提供了很高的安全水平和精准的时间控制。两个看门狗定时器都是用来解决软件故障问题的。

看门狗定时器在内部计数值达到预设门限的时候，会触发一个复位(对于窗口看门狗定时器来说，会产生一个中断)。当处理器工作在调试模式的时候看门狗定时器定时计数器可以停止计数。

## 16.1. 独立看门狗定时器(FWDGT)

### 16.1.1. 简介

独立看门狗定时器(FWDGT)有独立时钟源(IRC32K)。即使主时钟失效，FWDGT依然能保持正常工作状态，适用于需要独立环境且对计时精度要求不高的场合。

当内部向下计数器的计数值达到0，独立看门狗会产生一个系统复位。使能独立看门狗的寄存器写保护功能可以避免寄存器的值被意外的配置篡改。

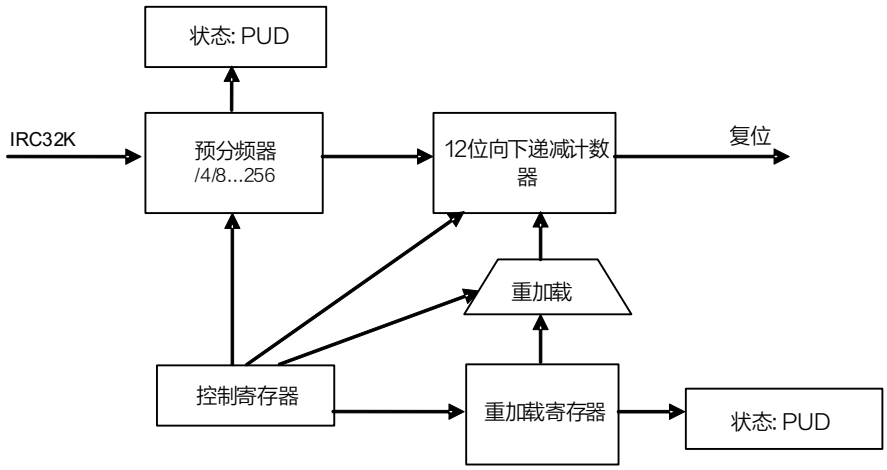
### 16.1.2. 主要特征

- 独立运行的12位向下计数器；
- 使能看门狗定时器，当向下计数器的值达到0时产生系统复位；
- 独立时钟源，独立看门狗定时器在主时钟故障(例如待机和深度睡眠模式下)时仍能工作；
- 独立看门狗定时器硬件控制位，用来控制是否在上电时自动启动独立看门狗定时器；
- 可以配置独立看门狗定时器在调试模式下选择停止还是继续工作。

### 16.1.3. 功能说明

独立看门狗定时器带有一个 8 级预分频器和一个 12 位的向下递减计数器。参考*图 16-1. 独立看门狗定时器框图*的独立看门狗定时器的功能模块。

**图 16-1. 独立看门狗定时器框图**




向控制寄存器(FWDGT_CTL)中写0xCCCC可以开启独立看门狗定时器，计数器开始向下计数。当计数器记到0x000，产生一次系统复位。

在任何时候向FWDGT_CTL中写0xAAAA都可以重装载计数器，重装载值来源于重装载寄存器(FWDGT_RLD)。软件可以在计数器计数值达到0x000之前可以通过重装载计数器来阻止看门狗定时器产生系统复位。

如果在选项字节中打开了"硬件看门狗定时器"功能，那么在上电的时候看门狗定时器就被自动打开。为了避免系统复位，软件应该在计数器达到0x000之前重装载计数器。

预分频寄存器(FWDGT_PSC)和FWDGT_RLD寄存器都有写保护功能。在写数据到这些寄存器之前，需要写0x5555到FWDGT_CTL中。写其他任何值到控制寄存器中将会再次启动对这些寄存器的写保护。当FWDGT_PSC或者FWDGT_RLD更新时，FWDGT_STAT寄存器的状态位被置1。

如果DBG中控制寄存器0(DBG_CTL0)中FWDGT_HOLD位被清0，即使Cortex®-M4内核停止(调试模式下)独立看门狗定时器依然工作。如果FWDGT_HOLD位置1，独立看门狗定时器将在Cortex®-M4内核停止时(调试模式下)停止工作。

**表 16-1. 独立看门狗定时器在 32kHz(IRC32K)时的最小 / 最大超时周期**

| 预分频系数 | PSC[2:0] 位 | 最小超时(ms)<br>RLD[11:0]=0x000 | 最大超时(ms)<br>RLD[11:0]=0xFFF |
|---|---|---|---|
| 1 / 4 | 000 | 0.03125 | 511.90625 |
| 1 / 8 | 001 | 0.03125 | 1023.7812 |
| 1 / 16 | 010 | 0.03125 | 2047.53125 |
| 1 / 32 | 011 | 0.03125 | 4095.03125 |
| 1 / 64 | 100 | 0.03125 | 8190.03125 |
| 1 / 128 | 101 | 0.03125 | 16380.03125 |
| 1 / 256 | 110 or 111 | 0.03125 | 32760.03125 |

通过校准IRC32K可以使自由看门狗定时器超时更加精确。

**注意：** 1. 当执行完喂狗reload操作之后，如需要立即进入deepsleep / standby模式时，必须通过软件设置，在reload命令及deepsleep / standby模式命令中间插入(3个以上)IRC32K时钟间

隔。2.两次连续喂狗之间需插入7个及以上IRC32K时钟间隔。

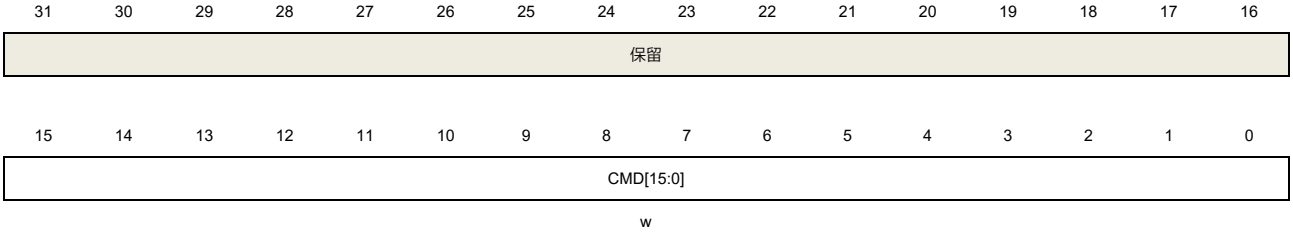
## 16.1.4. FWDGT 寄存器

FWDGT 基地址：0x4000 3000

### 控制寄存器(FWDGT_CTL)

地址偏移：0x00
复位值：0x0000 0000

该寄存器可以按半字(16 位)或字(32 位)访问。

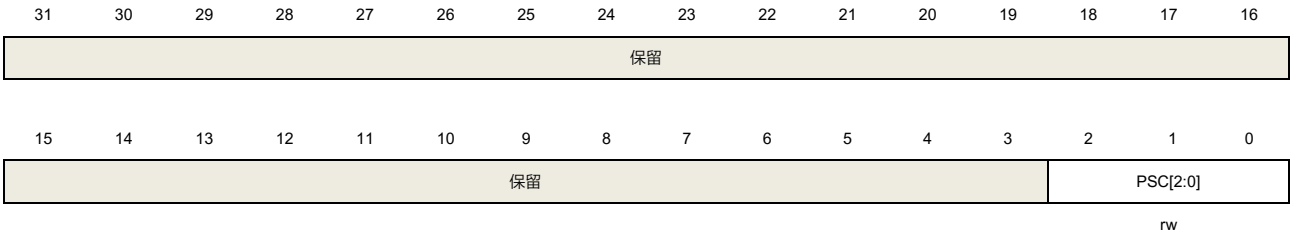



| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CMD[15:0] | 只可写，写入不同的值来产生不同的功能<br>0x5555：关闭FWDGT_PSC和FWDGT_RLD的写保护<br>0xCCCC：开启独立看门狗定时器定时计数器。计数减到0时产生复位<br>0xAAAA：重装载计数器 |

### 预分频寄存器(FWDGT_PSC)

地址偏移：0x04
复位值：0x0000 0000

该寄存器可以按半字(16 位)或字(32 位)访问。




| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:3 | 保留 | 必须保持复位值。 |
| 2:0 | PSC[2:0] | 独立看门狗定时器计时预分频选择。写这些位之前要通过向FWDGT_CTL寄存器写0x5555去除写保护。在改写这个寄存器的过程中，FWDGT_STAT寄存器的PUD位被置1，此时读取此寄存器的值都是无效的。<br>000：1 / 4 |

001：1 / 8
010：1 / 16
011：1 / 32
100：1 / 64
101：1 / 128
110：1 / 256
111：1 / 256

如果应用需要使用不同的预分频系数，改变预分频值之前必须等到PUD位被清0。更新了预分频寄存器中的值后，在代码持续执行之前不必等待PUD值被清零(在进入省电模式前需要等待PUD值清零)。

### 重装载寄存器(FWDGT_RLD)

地址偏移：0x08
复位值：0x0000 0FFF

该寄存器可以按半字(16 位)或字(32 位)访问。

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

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

| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11:0 | RLD[11:0] | 独立看门狗定时器定时计数器重装载值，向FWDGT_CTL寄存器写入0xAAAA的时候，这个值会被更新到看门狗定时器计数器中。<br>这些位有写保护功能。在写这些位之前需向FWDGT_CTL寄存器中写0x5555。在改写这个寄存器的过程中，FWDGT_STAT寄存器的RUD位被置1，从此寄存器中读取的任何值都是无效的。<br>如果应用需要使用不同的重装载值，改变重加载值之前必须等到RUD位被清0。更新了重加载寄存器的值后，在代码持续执行之前不必等待RUD值被清零(在进入省电模式前需要等待RUD值清零)。 |

### 状态寄存器(FWDGT_STAT)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器可以按半字(16 位)或字(32 位)访问。

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

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

| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |
| 1 | RUD | 独立看门狗定时计数器重装载值更新<br>FWDGT_RLD寄存器写操作时，该位被置1，此时读取FWDGT_RLD寄存器的任何值都是无效的。在FWDGT_RLD寄存器更新后，该位由硬件清零。 |
| 0 | PUD | 独立看门狗定时器预分频值更新<br>FWDGT_PSC寄存器写操作时，该位被置1，此时读取FWDGT_PSC寄存器的任何值都是无效的。在FWDGT_PSC寄存器更新后，该位由硬件清零。 |

# 16.2. 窗口看门狗定时器(WWDGT)

## 16.2.1. 简介

窗口看门狗定时器(WWDGT)用来监测由软件故障导致的系统故障。窗口看门狗定时器开启后，7位向下递减计数器值逐渐减小。计数值达到0x3F时会产生系统复位(CNT[6]位被清0)。在计数器计数值达到窗口寄存器值之前，计数器的更新也会产生复位。因此软件需要在给定的区间内更新计数器。窗口看门狗定时器在计数器计数值达到0x40会产生一个提前唤醒标志，如果使能中断将会产生中断。

窗口看门狗定时器时钟是由APB1时钟预分频而来。窗口看门狗定时器适用于需要精确计时的场合。

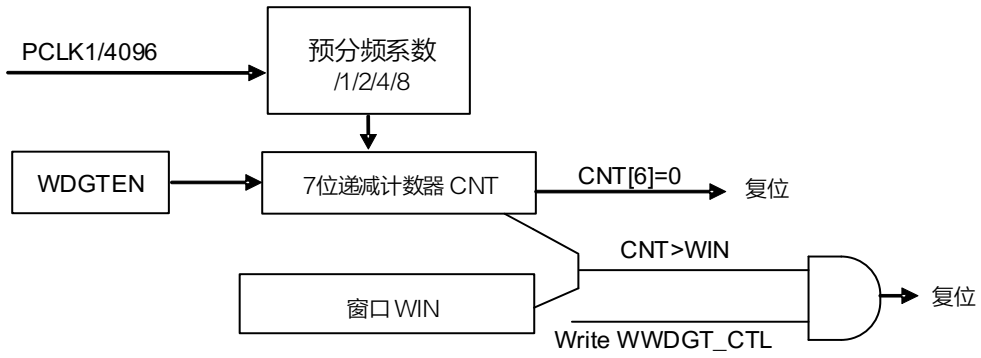
## 16.2.2. 主要特征

- 可编程的7位自由运行向下递减计数器。
- 当窗口看门狗使能后，有以下两种情况会产生复位：
  - 当计数器达到0x3F时产生复位；
  - 当计数器的值大于窗口寄存器的值时，更新计数器会产生复位。
- 提前唤醒中断(EWI)：看门狗定时器打开，中断使能，当计数值达到0x40时将会产生中断。
- 可以配置窗口看门狗定时器在调试模式下选择停止还是继续工作。

## 16.2.3. 功能说明

如果窗口看门狗定时器使能(将WWDGT_CTL寄存器的WDGTEN位置1)，计数值达到0x3F的时候产生系统复位(CNT[6]位被清0)。或是在计数值达到窗口寄存器值之前，更新计数器也会产生系统复位。

**图 16-2. 窗口看门狗定时器框图**



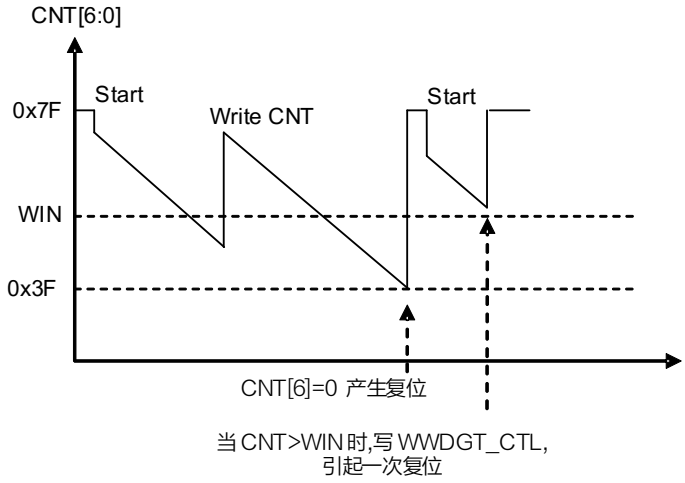

上电复位之后窗口看门狗定时器总是关闭的。软件可以向WWDGT_CTL的WDGTEN写1开启窗口看门狗定时器。窗口看门狗定时器打开后，计数器始终递减计数，计数器配置的值应该大于0x3F，也就是说CNT[6]位应该被置1。CNT[5:0]决定了两次重装载之间的最大间隔时间。计数器的递减速度取决于APB1时钟和预分频器(WWDGT_CFG寄存器的PSC[1:0]位)。

配置寄存器(WWDGT_CFG)中的WIN[6:0]位用来设定窗口值。当计数器的值小于窗口值，且大于0x3F的时候，重装载向下计数器可以避免复位，否则在其他时候进行重加载就会引起复位。

对WWDGT_CFG寄存器的EWIE位置1可以使能提前唤醒中断(EWI)，当计数值达到0x40的时候该中断产生。同时可以用相应的中断服务程序(ISR)来触发特定的行为(例如通信或数据记录)，来分析软件故障的原因以及在器件复位的时候挽救重要数据。此外，在ISR中软件可以重装载计数器来管理软件系统检查等。在这种情况下，窗口看门狗定时器将永远不会复位但是可以用于其他地方。

通过将WWDGT_STAT寄存器的EWIF位写0可以清除EWI中断。

**图 16-3. 窗口看门狗定时器时序图**




窗口看门狗定时器超时的计算公式如下：

$$t_{WWDGT}=t_{PCLK1} \times 4096 \times 2^{PSC} \times (CNT[5:0]+1) \quad (ms) \qquad (式16\text{-}1)$$

其中：

$t_{WWDGT}$：窗口看门狗定时器的超时时间

$t_{PCLK1}$：APB1以ms为单位的时钟周期

$t_{WWDGT}$的最大值和最小值请参考*表 16-2. 在 60MHz ($f_{PCLK1}$) 时的最大/最小超时值*。

**表 16-2. 在 60MHz ($f_{PCLK1}$) 时的最大/最小超时值**

| 预分频系数 | PSC[1:0] | 最小超时<br>CNT[6:0] =0x40 | 最大超时<br>CNT[6:0]=0x7F |
|---|---|---|---|
| 1 / 1 | 00 | 68.27 μs | 4.37 ms |
| 1 / 2 | 01 | 136.54 μs | 8.74 ms |
| 1 / 4 | 10 | 273.08 μs | 17.48 ms |
| 1 / 8 | 11 | 546.16 μs | 34.96 ms |

如果MCU调试模块中的WWDGT_HOLD位被清0，即使Cortex®-M4内核停止工作(调试模式下)，窗口看门狗定时器也可以继续工作。当WWDGT_HOLD位被置1时，窗口看门狗定时器在调试模式下停止。

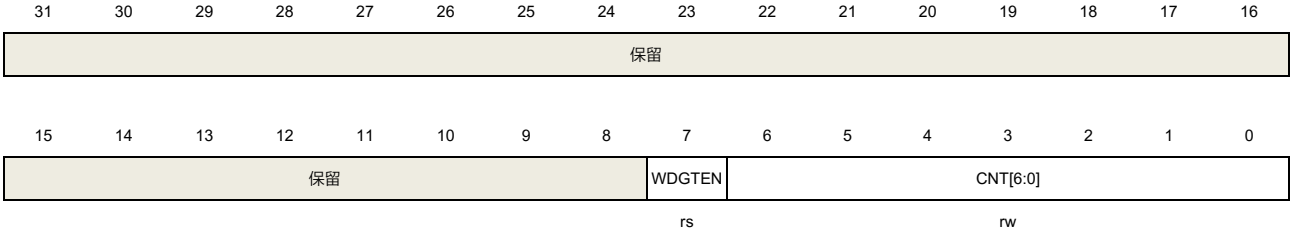
## 16.2.4. WWDGT 寄存器

WWDGT 基地址：0x4000 2C00

### 控制寄存器(WWDGT_CTL)

地址偏移：0x00
复位值：0x0000 007F

该寄存器可以按半字(16 位)或字(32 位)访问。

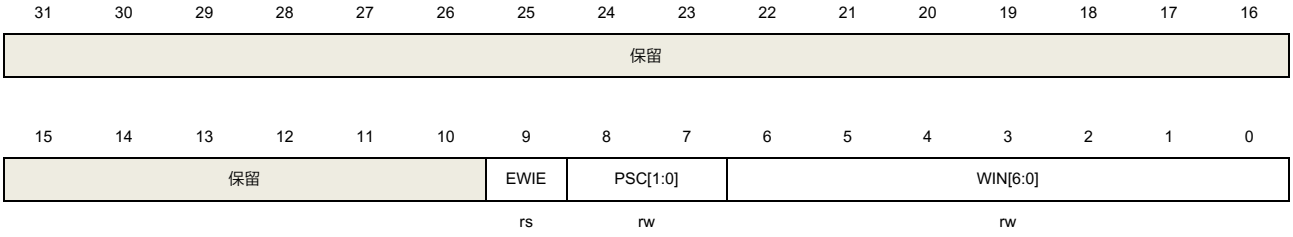



| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7 | WDGTEN | 开启窗口看门狗定时器，硬件复位的时候清0，写0无效。<br>0：关闭窗口看门狗定时器<br>1：开启窗口看门狗定时器 |
| 6:0 | CNT[6:0] | 看门狗定时器计数器的值。当计数值从0x40降到0x3F时，产生看门狗定时器复位。当计数器值高于窗口值的时候，写计数器可以产生看门狗定时器系统复位。 |

### 配置寄存器(WWDGT_CFG)

地址偏移：0x04
复位值：0x0000 007F

该寄存器可以按半字(16 位)或字(32 位)访问。



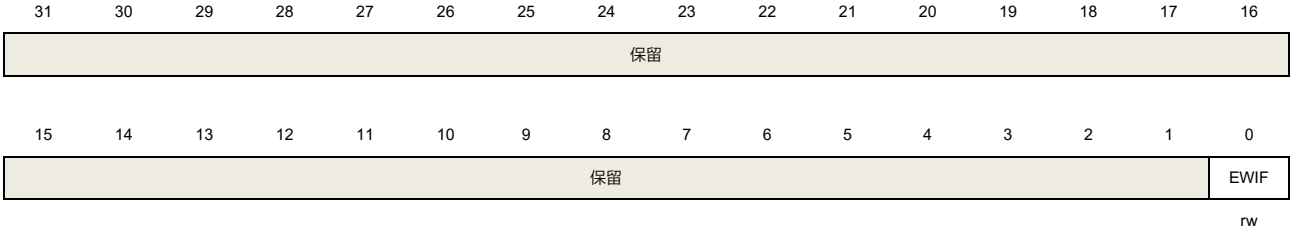

| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:10 | 保留 | 必须保持复位值。 |
| 9 | EWIE | 提前唤醒中断使能。如果该位被置1，计数值达到0x40时触发中断。该位由硬件复位清0，或通过RCU模块的WWDGT软件复位来清0。写0没有任何作用。 |

| | | |
|---|---|---|
| 8:7 | PSC[1:0] | 预分频器，看门狗定时器计数器的时间基准<br>00：PCLK1 / 4096 / 1<br>01：PCLK1 / 4096 / 2<br>10：PCLK1 / 4096 / 4<br>11：PCLK1 / 4096 / 8 |
| 6:0 | WIN[6:0] | 窗口值，当看门狗定时器计数器的值大于窗口值时，写看门狗定时器计数器(WWDGT_CTL的CNT位)会产生复位。 |

### 状态寄存器(WWDGT_STAT)

地址偏移：0x08
复位值：0x0000 0000

该寄存器可以按半字(16 位)或字(32 位)访问。




| 位/位域 | 名称 | 说明 |
|---|---|---|
| 31:1 | 保留 | 必须保持复位值。 |
| 0 | EWIF | 提前唤醒中断标志位。当计数值达到0x40，即使中断没有被使能(WWDGT_CFG中的EWIE位为0)该位也会被硬件置1。这个bit可以通过写0清零，写1无效。 |

# 17. 实时时钟(RTC)

## 17.1. 简介

RTC模块提供了一个包含日期(年/月/日)和时间(时/分/秒/亚秒)的日历功能。除亚秒用二进制码显示外，时间和日期都以BCD码的形式显示。RTC可以进行夏令时补偿。RTC可以工作在省电模式下，并通过软件配置来智能唤醒。RTC支持外接更高精度的低频时钟，用以达到更高的日历精度。

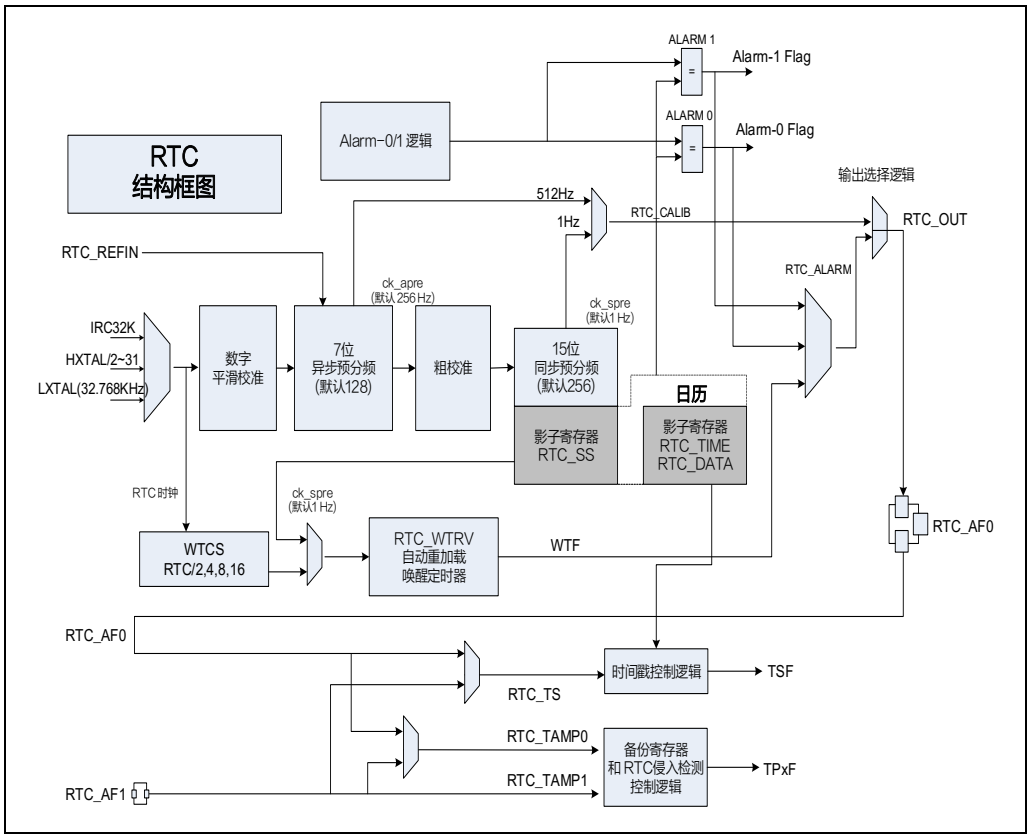
## 17.2. 主要特性

- 通过软件设置来实现夏令时补偿
- 参考时钟检测功能：通过外接更高精度的低频率时钟源(50Hz或60Hz)来提高日历精度
- 数字校准功能：通过调整最小时间单位(最大可调精度0.95ppm)来进行日历校准
- 通过移位功能进行亚秒级调整
- 记录事件时间的时间戳功能
- 两个模式可配置的独立的侵入检测
- 可编程的日历和两个位域可屏蔽的闹钟
- 可屏蔽的中断源
  - 闹钟 0 和闹钟 1
  - 时间戳检测
  - 侵入检测
  - 自动唤醒事件
- 20个32位(共80字节)通用备份寄存器，能够在省电模式下保存数据。当有外部事件侵入时，备份寄存器将会复位

# 17.3. 功能描述

## 17.3.1. 结构框图

**图 17-1. RTC 结构框图**




RTC 单元包括：

- 闹钟事件/中断
- 侵入事件/中断
- 32位备份寄存器
- 可选的RTC输出功能
  - 512Hz(默认预分频值)：(RTC_OUT)PC13
  - 1Hz(默认预分频值)：(RTC_OUT)PC13
  - 闹钟事件(极性可配置)：(RTC_OUT)PC13
  - 自动唤醒事件(极性可配置)：(RTC_OUT)PC13
- 可选的RTC输入功能
  - 时间戳事件检测(RTC_TS)：PC13 和 PI8
  - 侵入事件检测 0(RTC_TAMP0)：PC13 和 PI8
  - 侵入事件检测 1(RTC_TAMP1)：PI8
  - 参考时钟输入 RTC_REFIN(50 或 60Hz)

PC13 和 PI8 的具体配置参考*通用和备用输入/输出接口(GPIO 和 AFIO)。*

### 17.3.2. 时钟源和预分频

RTC单元有三个可选的独立时钟源：LXTAL、IRC32K和HXTAL的2-31(由RCU_CFG寄存器配置)分频后的时钟。

在RTC单元，有两个预分频器用来实现日历功能和其他功能。一个分频器是7位异步预分频器，另一个是15位同步预分频器。异步分频器主要用来降低功率消耗。如果两个分频器都被使用，建议异步分频器的值尽可能大。

两个预分频器的频率计算公式如下：

$$f_{ck_apre} = \frac{f_{rtcclk}}{FACTOR_A + 1} \tag{17-1}$$

$$f_{ck_spre} = \frac{f_{ck_apre}}{FACTOR_S + 1} = \frac{f_{rtcclk}}{(FACTOR_A + 1)*(FACTOR_S + 1)} \tag{17-2}$$

ck_apre 用于为 RTC_SS 亚秒寄存器自减计数器提供时钟，该寄存器值为二进制，表示到达下一秒时间，该寄存器自减到 0 时，自动加载 FACTOR_S 的值。ck_spre 用于为日历寄存器提供时钟，每个时钟增加一秒。

### 17.3.3. 影子寄存器

当APB总线访问RTC日历寄存器RTC_DATE、RTC_TIME和RTC_SS时，BPSHAD位决定是访问影子寄存器还是真实日历寄存器。默认情况下BPSHAD为0，APB总线访问影子日历寄存器。每两个RTC时钟，影子日历寄存器值会更新为真实日历寄存器的值，与此同时RSYNF位也会再次置位。在Deep-sleep和Standby模式下，影子寄存器不会更新。退出这两种模式时，软件必须清除RSYNF位。如果想要在BPSHAD=0的情况下读日历寄存器的值，须等待RSYNF置1(最大的等待时间是2个RTC时钟周期)。

**注意：** 在BPSHAD=0下，读日历寄存器(RTC_SS，RTC_TIME，RTC_DATE)的APB时钟的频率($f_{APB}$)必须至少是RTC时钟频率($f_{RTCCLK}$)的七倍。

系统复位将复位影子寄存器。

### 17.3.4. 位域可屏蔽可配置的闹钟

RTC闹钟功能被划分为多个位域并且每一个位域有一个该域的可屏蔽位。

RTC闹钟功能的使能由RTC_CTL寄存器中的ALRMxEN(x=0，1)位控制。当ALRMxEN=1(x=0，1)并且闹钟所有位域的值与对应的日历时间值匹配，ALRMxF(x=0，1)标志位将会置位。

**注意：** 当秒字段未被屏蔽时(RTC_ALRMxTD 寄存器的 MSKS=0)，为确保正常运行，RTC_PSC寄存器的同步预分频系数(FACTOR_S)应大于等于 3。

如果一个位域被屏蔽，这个位域被认为在逻辑上匹配的。如果所有的位域被屏蔽，在 ALRMxEN 位被置位 3 个 RTC 时钟周期后，ALRMxF 位将置位。

### 17.3.5. 可配置周期的自动唤醒定时器

RTC具有一个16位的自动递减计数器用来周期性产生唤醒标志。

该功能通过WTEN置1来使能，并且可以工作在省电模式。

自动递减计数器有两种可选的时钟来控制：

1) RTC 时钟的 2/4/8/16 分频：
   如果 RTC 时钟为 LXTAL(32.768 KHz)，则唤醒中断周期在 122us 和 32s 之间，分辨率低至 61us。

2) 内部时钟 ck_spre：
   如果 ck_spre 为 1Hz，则唤醒中断周期在 1s 到 36h 之间，分辨率低至 1s。
   - WTCS[2：1] = 0b10，唤醒中断周期在 1s 到 18h
   - WTCS[2：1] = 0b11，唤醒中断周期在 18h 到 36h

该功能使能后，计数器自动递减。当计数器到0时，WTF标志位置1，唤醒计数器自动重载RTC_WUT的值。当WTF置1后，必须软件清除该标志。如果WTIE被置位，计数器到0时，会产生唤醒中断，从而使系统退出省电模式。系统复位对该功能没有影响。WTF标志可以从RTC_ALARM通道输出到PC13。

### 17.3.6. RTC 初始化和配置

#### RTC 寄存器写保护

在默认情况下，PMU_CTL寄存器的BKPWEN位被清0。所以写RTC寄存器前需要软件提前设置BKPWEN位。

上电复位后，大多数RTC寄存器是被写保护的。写入这些寄存器的第一步是解锁这些保护。

通过下面的步骤，可以解锁这些保护：

1. 写'0xCA'到 RTC_WPK 寄存器
2. 写'0x53'到 RTC_WPK 寄存器

写一个错误的值到RTC_WPK会使写保护再次生效。写保护状态不受系统复位的影响。被写保护的寄存器如下：

RTC_TIME，RTC_DATE，RTC_CTL，RTC_STAT，RTC_PSC，RTC_WUT，RTC_COSC，RTC_ALRM0TD，RTC_ALRM1TD，RTC_SHIFTCTL，RTC_HRFC，RTC_ALRM0SS，RTC_ALRM1SS

#### 日历初始化和配置

通过以下步骤可以设置日历和预分频器的值：

1. 设置 INITM 位为 1 进入初始化模式。等待 INITF 位被置 1。
2. 在 RTC_PSC 寄存器中，设置同步和异步预分频器的分频系数。
3. 在影子寄存器(RTC_TIME 和 RTC_DATE)中写初始的日历值，并且通过设置 RTC_CTL 寄

存器的 CS 位来配置时间的格式(12 或 24 小时制)。

4. 清除 INITM 位退出初始化模式。

大约4个RTC时钟周期后，真正的日历寄存器将从影子寄存器载入时间和日期的设定值，同时日历计数器将要重新开始运行。

**注意：**初始化以后如果要读取日历寄存器(BPSHAD=0)，软件应该确保RSYNF位已经置1。

YCM标志表明日历是否完成初始化，该标志会硬件检查日历的年份值。

#### 夏令时

通过S1H，A1H和DSM位配置，RTC模块可以支持夏令时补偿调节功能。

当日历正在运行时，S1H和A1H能使日历减去或加上1小时。S1H和A1H功能可以重复设置，可以软件配置DSM位来记录这个调节操作。设置S1H或A1H位后，减或加1小时将在下一秒钟到来时生效。

#### 闹钟功能操作步骤

为了避免意外的闹钟标记置位和亚稳态，闹钟功能的操作应遵循如下流程：

1. 清除寄存器 RTC_CTL 的 ALRMxEN(x=0，1)位，禁用闹钟
2. 设置 Alarm 寄存器(RTC_ALRMxTD/RTC_ALRMxSS)
3. 设置寄存器 RTC_CTL 的 ALRMxEN 位，使能闹钟功能

### 17.3.7. 读取日历

#### 当 BPSHAD=0 时，读日历寄存器

当BPSHAD=0，从影子寄存器读日历的值。由于同步机制的存在，正常读取日历需要满足一个基本要求：APB1总线时钟频率必须大于或等于RTC时钟频率的7倍。在任何情况下APB1总线时钟的频率都不能低于RTC的时钟频率。

当APB1总线时钟频率不等于或低于7倍RTC时钟频率时，日历的读取应该遵守以下流程：

1. 读取两次日历时间和日期寄存器
2. 如果两次的值相等，那么这个值就是正确的
3. 如果这两次的值不相等，应该再读一次
4. 第三次的值可以认为是正确的

RSYNF每2个RTC时钟周期被置位一次。在这时，影子日历寄存器会更新为真实的日历时间和日期。

为了确保这3个值(RTC_SS，RTC_TIME，RTC_DATE)为同一时间，硬件上采取了如下一致性机制：

1. 读RTC_SS锁定RTC_TIME和RTC_DATE的更新
2. 读RTC_TIME锁定RTC_DATE的更新

3. 读RTC_DATE解锁RTC_TIME和RTC_DATE的更新

如果想在一个很短的时间间隔内(少于2个RTCCLK)读取日历，应先清除RSYNF位并等待其置位后再读取。

下面几种情况，软件须等待RSYNF置位后才能读日历寄存器(RTC_SS，RTC_TIME，RTC_DATE)：

1. 系统复位之后
2. 日历初始化之后
3. 一次移位操作之后

特别是从低功耗模式唤醒后，软件必须清除RSYNF位并等待RSYNF再次置位后才能读取日历寄存器。

#### 当 BPSHAD=1 时，读日历寄存器

当BPSHAD=1，RSYNF位会被硬件清0，读日历寄存器不需考虑RSYNF位。当前真实的日历寄存器值会被直接读取。如此配置的好处是当从低功耗模式(Deep-sleep/Standby模式)唤醒后，软件可以立即获取当前日历寄存器的值而无需加入任何等待延迟(此延迟最大为2个RTC时钟周期)。

由于没有RSYNF位周期性的置位，如果两次读日历寄存器之间出现ck_apre时钟边沿，不同寄存器(RTC_SS/RTC_TIME/RTC_DATE)的值可能并非同一时刻。

另外，如果日历寄存器的值正在发生变化的时刻被APB总线读取，那么有可能APB总线读取的值是不准确的。

为了确保日历值的正确性和一致性，读取时软件须如下操作：连续读取所有日历寄存器的值两次，如果上两次的值是一样的，那么这个值就是一致的且准确的。

### 17.3.8. RTC 复位

在RTC单元，有两个复位源可用：系统复位和备份域复位。

当系统复位有效时，日历影子寄存器和RTC_STAT寄存器的某些位将要复位到默认值。

备份域复位将会影响下面的寄存器，但系统复位不会对它们产生影响：

- RTC 真实的日历寄存器
- RTC 控制寄存器(RTC_CTL)
- RTC 预分频寄存器(RTC_PSC)
- RTC 唤醒定时器寄存器(RTC_WUT)
- RTC 粗校准寄存器(RTC_COSC)
- RTC 高精度频率补偿寄存器(RTC_HRFC)
- RTC 移位控制寄存器(RTC_SHIFTCTL)
- RTC 时间戳寄存器(RTC_SSTS/RTC_TTS/RTC_DTS)
- RTC 侵入寄存器(RTC_TAMP)
- RTC 备份寄存器(RTC_BKPx)

- RTC 闹钟寄存器(RTC_ALRMxSS/RTC_ALRMxTD)

当系统复位或者进入省电模式的时候，RTC单元将会继续运行。但是如果备份域复位，RTC将会停止计数并且所有的寄存器会复位。

### 17.3.9. RTC 移位功能

当用户有一个高精度的远程时钟而且RTC 1hz时钟(ck_spre)和远程时钟只有一个亚秒级的偏差，RTC单元提供一个称作移位的功能去消除这个偏差来提高秒钟的精确性。

以二进制格式显示亚秒值，RTC运行时该值是递减计数。因此通过增加RTC_SHIFTCTL寄存器的SFS[14：0]的值到RTC_SS同步预分频器计数器值SSC[15：0])或通过增加SFS[14：0]的值到同步预分频器计数器SSC[15：0]并且同时置位A1S位，能分别延迟或提前下一秒到达的时间。

RTC_SS的最大值取决于RTC_PSC寄存器的FACTOR_S的值。FACTOR_S越大，调整的精度也就越高。

因为1Hz的时钟(ck_spre)由FACTOR_A和FACTOR_S共同产生，越高的FACTOR_S值就意味着越低的FACTOR_A值，同时越低的FACTOR_A意味着越高的功耗。

**注意：**在使用移位功能之前，软件必须检查 RTC_SS 中 SSC 的第 15 位(SSC[15])并确保该位为 0。写 RTC_SHIFTCTL 寄存器之后，RTC_STAT 寄存器的 SOPF 位将会再次置位。当同步移位操作完成时，SOPF 位被硬件清 0。系统复位不影响 SOPF 位。当 REFEN=0 时，移位操作才能正确的工作。如果 REFEN=1，软件禁止写入 RTC_SHIFTCTL。

### 17.3.10. RTC 参考时钟检测

RTC参考时钟是另外一种提高RTC秒级精度的方法。为了使能这项功能，需要有一个相对于LXTAL有更高精度的外部参考时钟源(50Hz或60Hz)。

使能这项功能之后(REFEN=1)，每一个秒更新的时钟(1Hz)边沿将与最近的RTC_REFIN参考时钟沿进行对比。在大多数情况下，这两个时钟沿是对齐的。但当两个时钟沿由于LXTAL准确度的原因没有对齐的时候，RTC参考时钟的检测功能会偏移1Hz时钟沿一点相位，使得下一个1Hz时钟沿和参考时钟沿对齐。

当REFEN=1，每一秒前后都会有一个进行检测的时间窗，处于不同的检测状态，时间窗时长也不同。当检测状态处于检测第一个参考时钟边沿时，使用7个ck_apre时长的时间窗，当检测状态处于边沿对齐操作时，使用3个ck_apre时长的时间窗。

无论使用哪一种时间窗，当参考时钟在时间窗中被检测到的时候，同步预分频计数器会被强制重载。当两个时钟(ck_spre和参考时钟)边沿是对齐的，这个重载操作对1Hz日历更新没有任何影响。但是当两个时钟边沿没有对齐时，这个重载操作将会移动ck_spre时钟边沿，以使得ck_spre(1Hz)时钟边沿和参考时钟边沿对齐。

当参考检测功能正在运行中但外部参考时钟消失(在3个ck_apre时长时间窗内没有发现参考时钟边沿)，日历也能通过LXTAL继续自动更新。如果这个参考时钟重新恢复，参考时钟检测功能会先用7个ck_apre时长时间窗口去检测参考时钟，然后用3个ck_apre时长时间窗口去调节

ck_spre(1Hz)时钟边沿。

**注意：**使能参考时钟检测功能之前(REFEN=1)，软件必须配置 FACTOR_A 为 0x7F，FACTOR_S 为 0xFF。

待机模式下和数字粗校准时，参考时钟检测功能不可用。

### 17.3.11. RTC 数字粗校准

RTC有两种数字校准方法：数字粗校准和数字平滑校准。两种校准方法不能一起使用。

数字粗校准以异步预分频器输出为源，增加或者减少ck_apre时钟周期。

当COSD=0，在前2xCOSS(COSS：0到31)分钟内，每分钟增加两个ck_apre时钟周期，这样配置会提前更新日历。

当COSD=1，在前2xCOSS(COSS：0到31)分钟内，每分钟减少一个ck_apre时钟周期，这样配置会推迟更新日历。

仅能在初始化模式下配置数字粗校准，并且在清除INITM位后开始校准功能。整个校准过程持续64分钟。在64分钟内的前2xCOSS分钟内调整。

负校准的分辨率约为2PPM，而正校准的分辨率约为4PPM。

**注意：**在 RTC 时钟为 LXTAL 或者 HXTAL 时，可以进行数字粗校准。当 FACTOR_A<6 时，数字粗校准可能无法正确工作。

**例子：**

FACTOR_A 和 FACTOR_S 为默认值。RTC 时钟为 LXTAL，频率为 32.768 KHz。

在校准窗口内(64分钟)，仅在前2xCOSS分钟调整ck_apre(256Hz)时钟周期。

如果COSS=1，表示64分钟内前两分钟需要调整。

这种情况下在每个校准窗口内(64min x 60s/min x 32768周期/s)增加512(两分钟，每分钟两个ck_apre时钟周期)或者减少256(两分钟，每分钟一个ck_apre时钟周期)个RTC时钟周期。也就是说每次校准的分辨率为+4.069PPM或者-2.035PPM。那么每个月的最小校准时间为+10.5或者-5.27s，最大校准时间为+5.45到-2.72分钟。

### 17.3.12. RTC 数字平滑校准

RTC平滑校准是一种用于校准RTC频率的方法，该方法通过调整校准周期内的RTC时钟脉冲个数的方式来实现校准。

完成一次这种校准相当于在一次校准周期内，RTC时钟的脉冲个数增加或者减少了一定的数目。这种校准的分辨率大约为0.954ppm，范围为从-487.1ppm 到+488.5ppm。

校准周期的时间可以配置到 $2^{20}$/$2^{19}$/$2^{18}$ RTC 时钟周期，如果 RTC 的输入频率是 32.768KHz，这些校准周期时间分别代表 32/16/8 秒。

高精度频率补偿寄存器(RTC_HRFC)指定了在校准周期内要屏蔽的RTC时钟数目，CMSK[8：

0]位能屏蔽0到511个RTC时钟，这样RTC的频率最多降低487.1PPM。

为了提高RTC频率可以设置FREQI位。如果FREQI位被置位，将会有512个额外的RTC时钟周期增加到校准周期(32/16/8秒)时间期间，这意味着每$2^{11}/2^{10}/2^{9}$ RTC时钟插入一个RTC时钟周期。

因此使用FREQI可以使RTC频率增加488.5ppm。

同时使用CMSK和FREQI，每个周期时间可以调整-511到+512个RTC时钟周期。这意味着在0.954ppm分辨率的情况下，调整范围为从-487.1ppm到+488.5ppm。

当数字平滑校准功能正在运行时，按如下公式计算输出校准频率：

$$f_{cal} = f_{rtcclk} \times (1 + \frac{FREQI \times 512 - CMSK}{2^{N} + CMSK - FREQI \times 512}) \quad (17\text{-}3)$$

**注意：**N=20/19/18(32/16/8 秒)校准时间周期。

### 当 FACTOR_A < 3 时校准：

当异步预分频器值(FACTOR_A)被设置小于3时，若要使用校准功能，软件不能将FREQI位设置为1。当FACTOR_A<3，FREQI位设置将会被忽略。

当FACTOR_A小于3时，FACTOR_S值应小于标称值。假设RTC时钟频率是正常的32.768KHz，对应的FACTOR_S应该按下面所示设置：

FACTOR_A = 2：FACTOR_S减少2(8189)

FACTOR_A = 1：FACTOR_S减少4(16379)

FACTOR_A = 0：FACTOR_S减少8(32759)

当FACTOR_A小于3，CMSK为0x100，校准频率公式如下：

$$f_{cal} = f_{rtcclk} \times (1 + \frac{256 - CMSK}{2^{N} + CMSK - 256}) \quad (17\text{-}4)$$

**注意：**N=20/19/18(32/16/8 秒)校准时间周期。

### 验证 RTC 校准

提供1Hz校准时钟的输出用于协助软件测量并验证RTC的精度。

在有限的测量周期内测量RTC的频率，最高可能发生2个RTCCLK的测量误差。

为了消除这一测量误差，测量周期应该和校准周期一致。

■ 校准周期设为32秒(默认配置)

用准确的32秒周期去测量1Hz校准输出的准确性能保证这个测量误差在0.477ppm(在32秒周期内0.5个RTCCLK)之内。

■ 校准周期设为16秒(通过设置CWND16位)

使用此配置，CMSK[0]被硬件置0。

用准确的16秒周期去测量1Hz校准输出的准确性能保证这个测量误差在0.954ppm(在16秒周期内0.5个RTCCLK)之内。

■ 校准周期设为8秒(通过设置CWND8位)

使用此配置，CMSK[1：0]被硬件置0。

用准确的8秒周期去测量1Hz校准输出的准确性能保证这个测量误差在1.907ppm(在8秒周期内0.5个RTCCLK)之内。

#### 运行中重校准

当INITF位是0，用下面的步骤，软件可以更新RTC_HRFC：

1) 等待 SCPF 位置 0

2) 写一个新的值到 RTC_HRFC 寄存器

3) 3 个 ck_apre 时钟周期之后，新的校准设置开始生效

### 17.3.13. 时间戳功能

时间戳功能由RTC_TS管脚输入，通过配置TSEN位来使能。

当RTC_TS管脚检测到时间戳事件发生时，会将日历的值保存在时间戳寄存器中(RTC_DTS/RTC_TTS/RTC_SSTS)，同时时间戳标志(TSF)也将由硬件置1。如果时间戳中断使能被启用(TSIE)，时间戳事件会产生一个中断。

时间戳寄存器只会在时间戳事件第一次发生的时刻(TSF=0)记录日历时间，而当TSF=1时，时间戳事件的值不会被记录。

RTC模块提供了一个可选的功能特性，来增加时间戳事件的触发源：设置TPTS=1，使得侵入检测功能的侵入事件同时也作为时间戳事件的输入源。

**注意：**因为同步机制的原因，当时间戳事件发生时，TSF 会延迟 2 个 ck_apre 周期置位。

### 17.3.14. 侵入检测

RTC_TAMPx管脚可以作为侵入事件检测功能输入管脚，检测模式有两种可供用户选择：边沿检测模式或者是带可配置滤波功能的电平检测模式。

#### RTC 备份寄存器(RTC_BKPx)

RTC备份寄存器处于$V_{DD}$备份域中，即使$V_{DD}$电源被切断，该区域的寄存器的电源还可由$V_{BAT}$提供。从待机模式唤醒或系统复位操作都不会影响这些寄存器。

只有当被检测到有侵入事件和备份域复位时，这些寄存器复位。

#### 初始化侵入检测功能

TPxEN位可以独立使能对应于不同管脚上的RTC侵入检测功能。使能TPxEN位启动侵入检测

功能之前，需要设置好侵入检测的配置。当检测到侵入事件，相应的标志位(TPxF)将会置位。如果侵入事件中断使能被启用(TPIE)，侵入事件会产生一个中断。任何侵入事件都会导致备份寄存器(RTC_BKPx)复位。

#### 侵入事件源的时间戳

使能TPTS位，能让侵入检测功能被用作时间戳功能。如果这位被设置为1，当检测到侵入事件时，TSF也将会被置位，如同使能了时间戳功能。当检测到侵入事件时，无论TPTS位的值如何，TPxF位将置位。

#### 侵入事件检测为边沿检测模式

当FLT位为0x0时，侵入检测被设置成边沿检测模式，TPxEG位决定检测沿是上升沿还是下降沿。当侵入检测配置为边沿检测模式时，侵入检测输入管脚上的上拉电阻将会被禁用。

由于检测侵入事件会复位备份寄存器(RTC_BKPx)，因此对备份寄存器写操作时应该确保侵入事件导致的复位和写操作不会同时发生。避免这种情形的推荐方法是先关闭侵入检测功能，在完成写操作后再重新启动该功能。

**注意：** PC13 上的侵入检测功能即使 $V_{DD}$ 电源被关掉也依然可以运行。

#### 侵入事件检测为带可配置滤波功能的电平检测模式

当FLT位没有被设置成0x0时，侵入检测被设置成电平检测模式，FLT位决定有效电平需连续采样的次数(2，4或者8)。

当DISPU被设置成0(默认值)，内部的上拉电阻将会在每一次采样前预充电侵入管脚，这样侵入事件的输入管脚上就允许连接更大的电容。预充电的时间可以通过PRCH位来配置。越大的电容，所需的充电时间越长。

电平检测模式下每次采样之间的时间间隔是可配置的。通过调整采样频率(FREQ)，软件能在功耗和检测延迟之间取得一个平衡。

### 17.3.15. 校准时钟输出

如果COEN位设置为1，PC13管脚会输出参考校准时钟。

当COS位设置为0(默认值)并且异步预分频器(FACTOR_A)设为0x7F时，RTC_CALIB的频率是 $f_{RTCCLK}/64$。因此若RTCCLK的频率为32.768KHz，RTC_CALIB对应的输出为512Hz。因为下降沿存在轻微的抖动，因此推荐使用RTC_CALIB输出的上升沿。

当COS位设置为1时，RTC_CALIB的频率计算公式为：

$$f_{rtc_calib} = \frac{f_{rtcclk}}{(FACTOR_A+1)\times(FACTOR_S+1)} \quad (17\text{-}5)$$

若RTCCLK为32.768KHz，如果预分频器是默认值，那么RTC_CALIB对应的输出是1Hz。

### 17.3.16. 闹钟输出

当OS控制位被设置为0x01时，RTC_ALARM复用输出功能被启用。这个功能将直接输出RTC_STAT寄存器的ALRMxF或者WTF位的值。

RTC_CTL寄存器中的OPOL位可以配置ALRMxF位或者WTF位输出时候的极性，因此RTC_ALARM的输出电平有可能与相应的位值相反。

### 17.3.17. RTC 省电模式管理

**表 17-1. 省电模式管理**

| 模式 | 模式下能否工作 | 退出该模式的方法 |
|---|---|---|
| 睡眠模式 | 是 | RTC中断 |
| 深度睡眠模式 | 当时钟源是LXTAL或IRC32K时可以工作 | RTC闹钟/侵入事件/时间戳事件/唤醒事件 |
| 待机模式 | 当时钟源是LXTAL或IRC32K时可以工作 | RTC闹钟/侵入事件/时间戳事件/唤醒事件 |

### 17.3.18. RTC 中断

所有的 RTC 中断都被连接到 EXTI 控制器。

如果想使用RTC闹钟/侵入事件/时间戳/自动唤醒中断，应按下面步骤操作：

1) 设置并使能对应的 EXTI 中连接到 RTC 闹钟/侵入事件/时间戳/自动唤醒的中断线，然后配置该线为上升沿触发模式；
2) 配置并使能 RTC 闹钟/侵入事件/时间戳/自动唤醒中断；
3) 配置并使能 RTC 闹钟/侵入事件/时间戳/自动唤醒功能。

**表 17-2. 中断控制**

| 中断 | 事件标志 | 控制位 | 退出睡眠模式 | 退出深度睡眠模式 | 退出待机模式 |
|---|---|---|---|---|---|
| 闹钟0 | ALRM0F | ALRM0IE | Y | Y(*) | Y(*) |
| 闹钟1 | ALRM1F | ALRM1IE | Y | Y(*) | Y(*) |
| 唤醒 | WTF | WTIE | Y | Y(*) | Y(*) |
| 时间戳 | TSF | TSIE | Y | Y(*) | Y(*) |
| 侵入0 | TP0F | TPIE | Y | Y(*) | Y(*) |
| 侵入1 | TP1F | TPIE | Y | Y(*) | Y(*) |

**注意:** (*)当RTC时钟源是LXTAL或IRC32K时有效。

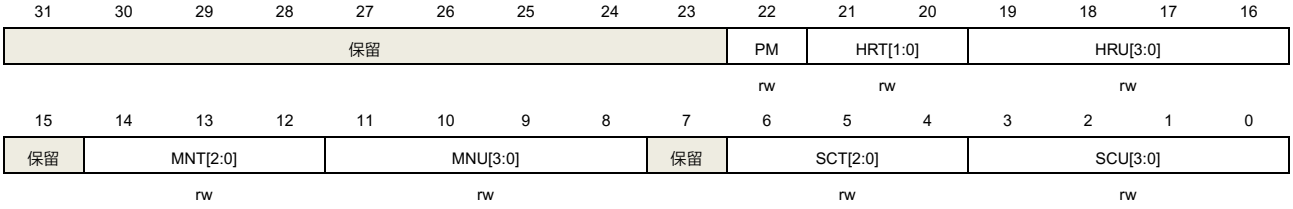
## 17.4. RTC 寄存器

RTC 基地址：0x4000 2800

### 17.4.1. 时间寄存器(RTC_TIME)

偏移地址：0x00
系统复位值：当BPSHAD = 0，0x0000 0000
当BPSHAD = 1，无影响
写保护寄存器，仅在初始化状态可以进行写操作

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:23 | 保留 | 必须保持复位值 |
| 22 | PM | AM/PM 标志<br>0：AM 或 24 小时制<br>1：PM |
| 21:20 | HRT[1:0] | 小时十位值，以 BCD 码形式存储 |
| 19:16 | HRU[3:0] | 小时个位值，以 BCD 码形式存储 |
| 15: | 保留 | 必须保持复位值 |
| 14:12 | MNT[2:0] | 分钟十位值，以 BCD 码形式存储 |
| 11:8 | MNU[3:0] | 分钟个位值，以 BCD 码形式存储 |
| 7 | 保留 | 必须保持复位值 |
| 6:4 | SCT[2:0] | 秒钟十位值，以 BCD 码形式存储 |
| 3:0 | SCU[3:0] | 秒钟个位值，以 BCD 码形式存储 |

### 17.4.2. 日期寄存器(RTC_DATE)

偏移地址：0x04
系统复位值：当BPSHAD = 0，0x0000 2101
当BPSHAD = 1，无影响

写保护寄存器，仅在初始化状态可以进行写操作

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | YRT[3:0] | | | | YRU[3:0] | | | |
| | | | | | | | | rw | | | | rw | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DOW[2:0] | | | MONT | MONU[3:0] | | | | 保留 | DAYT[1:0] | | DAYU[3:0] | | | | |
| rw | | | rw | rw | | | | | rw | | rw | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值 |
| 23:20 | YRT[3:0] | 年份十位值，以 BCD 码形式存储 |
| 19:16 | YRU[3:0] | 年份个位值，以 BCD 码形式存储 |
| 15:13 | DOW[2:0] | 星期<br>0x0：保留<br>0x1：星期一<br>…<br>0x7：星期日 |
| 12 | MONT | 月份十位值，以 BCD 码形式存储 |
| 11:8 | MONU[3:0] | 月份个位值，以 BCD 码形式存储 |
| 7:6 | 保留 | 必须保持复位值 |
| 5:4 | DAYT[1:0] | 日期十位值，以 BCD 码形式存储 |
| 3:0 | DAYU[3:0] | 日期个位值，以 BCD 码形式存储 |

### 17.4.3. 控制寄存器(RTC_CTL)

偏移地址：0x08
系统复位：无影响
备份域复位值：0x0000 0000
写保护寄存器

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | COEN | OS[1:0] | | OPOL | COS | DSM | S1H | A1H |
| | | | | | | | | rw | rw | | rw | rw | rw | w | w |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TSIE | WTIE | ALRM1IE | ALRM0IE | TSEN | WTEN | ALRM1EN | ALRM0EN | CCEN | CS | BPSHAD | REFEN | TSEG | WTCS[2:0] | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | |

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:24 | 保留 | 必须保持复位值 |
| 23 | COEN | 校准输出使能<br>0：关闭校准输出<br>1：使能校准输出 |
| 22:21 | OS[1:0] | 输出选择<br>该位用来选择输出的标志源。<br>0x0：禁用 RTC_ALARM 输出<br>0x1：启用闹钟 0 标志输出<br>0x2：启用闹钟 1 标志输出<br>0x3：启用唤醒标志输出 |
| 20 | OPOL | 输出极性<br>该位用来反转 RTC_ALARM 输出<br>0：禁用反转 RTC_ALARM 输出<br>1：启用反转 RTC_ALARM 输出 |
| 19 | COS | 校准输出选择<br>仅当 COEN=1 并且预分频器是默认值时有效。<br>0：校准输出是 512Hz<br>1：校准输出是 1Hz |
| 18 | DSM | 夏令时屏蔽位<br>该位可以通过软件灵活使用。常用来记录夏令时调整。 |
| 17 | S1H | 减 1 小时(冬季时间变化)<br>当前时间非零的情况下，将当前时间减去一个小时。<br>0：没有影响<br>1：在下一个秒改变时，将减少一个小时 |
| 16 | A1H | 增加 1 小时(夏季时间变化)<br>将当前时间增加一个小时。<br>0：没有影响<br>1：在下一个秒改变时，将增加一个小时 |
| 15 | TSIE | 时间戳中断使能<br>0：禁用时间戳中断<br>1：启用时间戳中断 |
| 14 | WTIE | 自动唤醒定时器中断使能<br>0：禁用自动唤醒定时器中断<br>1：启用自动唤醒定时器中断 |
| 13 | ALRM1IE | RTC 闹钟 1 中断使能<br>0：禁用闹钟中断<br>1：启用闹钟中断 |
| 12 | ALRM0IE | RTC 闹钟 0 中断使能 |

| | | |
|---|---|---|
| | | 0：禁用闹钟中断<br>1：启用闹钟中断 |
| 11 | TSEN | 时间戳功能使能<br>0：禁用时间戳功能<br>1：启用时间戳功能 |
| 10 | WTEN | 自动唤醒定时器功能使能<br>0：禁用自动唤醒定时器<br>1：启用自动唤醒定时器 |
| 9 | ALRM1EN | 闹钟 1 功能使能<br>0：禁用闹钟功能<br>1：启用闹钟功能 |
| 8 | ALRM0EN | 闹钟 0 功能使能<br>0：禁用闹钟功能<br>1：启用闹钟功能 |
| 7 | CCEN | 粗校准使能<br>0：禁用粗校准<br>1：启用粗校准<br>注意：在使能之前 FACTOR_A 必须大于等于 6，仅能在初始化状态进行写入 |
| 6 | CS | 时间格式<br>0：24 小时制<br>1：12 小时制<br>注意：仅能在初始化状态进行写入 |
| 5 | BPSHAD | 禁止影子寄存器<br>0：读取的日历的值来自影子日历寄存器<br>1：读取的日历的值来自真正日历寄存器<br>注意：如果 APB1 时钟的频率小于 RTCCLK 频率的 7 倍，该位必须设为 1 |
| 4 | REFEN | 参考时钟检测功能使能<br>0：禁用参考时钟检测功能<br>1：启用参考时钟检测功能<br>注意：仅能在初始化状态进行写入并且 FACTOR_S 必须为 0x00FF |
| 3 | TSEG | 时间戳事件有效检测边沿<br>0：上升沿是时间戳事件有效检测沿<br>1：下降沿是时间戳事件有效检测沿 |
| 2:0 | WTCS[2:0] | 自动唤醒定时器时钟选择<br>0x0：RTC 时钟的 16 分频<br>0x1：RTC 时钟的 8 分频<br>0x2：RTC 时钟的 4 分频<br>0x3：RTC 时钟的 2 分频<br>0x4，0x5：ck_spre(默认 1Hz)时钟 |

0x6，0x7：ck_spre (默认 1Hz)时钟并且将唤醒计数器值增加 $2^{16}$

### 17.4.4. 状态寄存器(RTC_STAT)

偏移地址：0x0C
系统复位：仅INITM，INITF和RSYNF位被置0，其他位无影响。
备份域复位值：0x0000 0007
写保护寄存器，除RTC_STAT[14：8]外

该寄存器只能按字(32 位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | TP1F | TP0F | TSOVRF | TSF | WTF | ALRM1F | ALRM0F | INITM | INITF | RSYNF | YCM | SOPF | WTWF | ALRM1W | ALRM0W |
| | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rw | r | rc_w0 | r | r | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:17 | 保留 | 必须保持复位值 |
| 16 | SCPF | 平滑校准挂起标志<br>在未进入初始化模式时向 RTC_HRFC 进行软件写操作，该位被硬件置 1。当平滑校准设置开始执行后，该位被硬件清零 0。 |
| 15 | 保留 | 必须保持复位值 |
| 14 | TP1F | RTC_TAMP1 事件标志<br>当在 tamper1 输入管脚检测到侵入事件时，该位硬件置 1。可以通过向该位软件写 0 来清除。 |
| 13 | TP0F | RTC_TAMP0 事件标志<br>当在 tamper0 输入管脚检测到侵入事件时，该位硬件置 1。可以通过向该位软件写 0 来清除。 |
| 12 | TSOVRF | 时间戳事件溢出标志<br>如果 TSF 位已经置位，当再次检测到时间戳事件时，该位会通过硬件置 1。<br>可以通过向该位软件写 0 来清除。 |
| 11 | TSF | 时间戳事件标志<br>当检测到一个时间戳事件时，该位会通过硬件置 1。可以通过向该位软件写 0 来清除。 |
| 10 | WTF | 唤醒定时器标志<br>当唤醒定时器减到 0 时，该位会通过硬件置 1。可以通过向该位软件写 0 来清除。该标志需要在 WTF 位再次置 1 之前的 1.5 个 RTC 时钟周期前完成软件清除该位。 |
| 9 | ALRM1F | Alarm1 发生标志<br>当现在的时间/日期与闹钟 1 设置的时间/日期匹配的时候，该位会通过硬件置 1。可以通过向该位软件写 0 来清除。 |

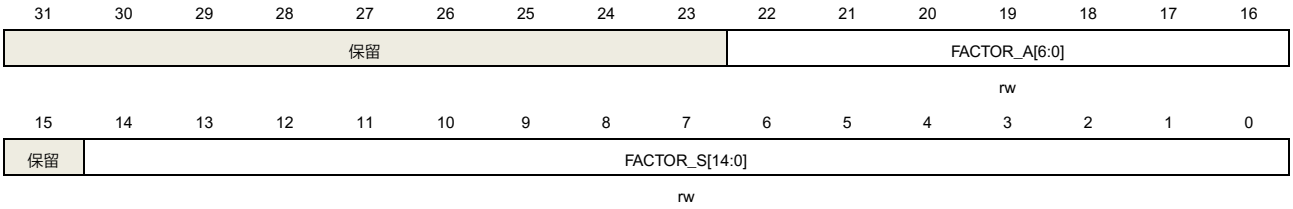
| | | |
|---|---|---|
| 8 | ALRM0F | Alarm0 发生标志<br>当现在的时间/日期与闹钟 0 设置的时间/日期匹配的时候，该位会通过硬件置 1。可以通过向该位软件写 0 来清除。 |
| 7 | INITM | 进入初始化模式<br>0：自由运行模式<br>1：进入初始化模式设置时间/日期和预分频，计数器将停止运行 |
| 6 | INITF | 初始化状态标志<br>该位被硬件置 1，初始化状态时可以设置日历寄存器和预分频器。<br>0：日历寄存器和预分频器的值不能改变<br>1：日历寄存器和预分频器的值可以改变 |
| 5 | RSYNF | 寄存器同步标志<br>每 2 个 RTCCLK 将会由硬件置 1 一次，同时会复制当前日历时间/日期到影子日历寄存器。初始化模式(INITM)，移位操作挂起标志(SOPF)或者禁止影子寄存器模式(BPSHAD = 1)会清除该位。该位也可以通过软件写 0 清除。<br>0：影子寄存器未同步<br>1：影子寄存器已同步 |
| 4 | YCM | 年份配置标志<br>当日历寄存器的年份值不为 0 时硬件置 1<br>0：日历尚未初始化<br>1：日历已经初始化 |
| 3 | SOPF | 移位功能操作挂起标志<br>0：移位操作没有挂起<br>1：移位操作挂起 |
| 2 | WTWF | 唤醒定时器可写标志<br>0：不允许更新唤醒定时器<br>1：允许更新唤醒定时器 |
| 1 | ALRM1WF | Alarm1 配置可写标志<br>硬件置位和清零。ALRM1EN=0 时，标记 alarm 是否可写<br>0：不允许修改 Alarm 寄存器设置<br>1：允许修改 Alarm 寄存器设置 |
| 0 | ALRM0WF | Alarm0 配置可写标志<br>硬件置位和清零。ALRM0EN=0 时，标记 alarm 是否可写<br>0：不允许修改 Alarm 寄存器设置<br>1：允许修改 Alarm 寄存器设置 |

### 17.4.5. 预分频寄存器(RTC_PSC)

偏移地址：0x10
系统复位：无影响
备份域复位值：0x007F 00FF

写保护寄存器，仅在初始化状态可以进行写操作

该寄存器只能按字(32位)访问

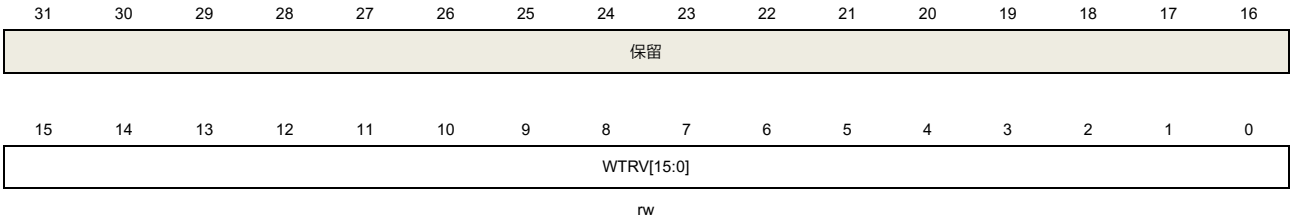



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:23 | 保留 | 必须保持复位值 |
| 22:16 | FACTOR_A[6:0] | 异步预分频系数<br>ck_apre 频率 = RTCCLK 频率/(FACTOR_A+1) |
| 15 | 保留 | 必须保持复位值 |
| 14:0 | FACTOR_S[14:0] | 同步预分频系数<br>ck_spre 频率 = ck_apre 频率/(FACTOR_S+1) |

### 17.4.6. 唤醒定时器寄存器(RTC_WUT)

偏移地址：0x14
系统复位：无影响
备份域复位值：0x0000 FFFF
写保护寄存器

该寄存器只能按字(32 位)访问

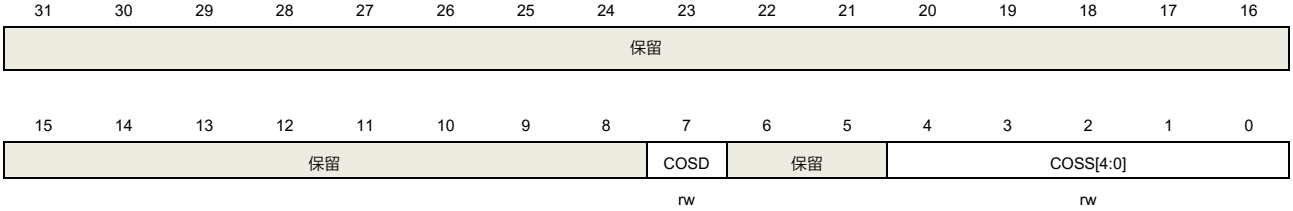



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15:0 | WTRV[15:0] | 自动唤醒定时器重载值<br>当 WTEN 置 1 时，每隔(WTRV[15：0]+1)个 ck_wut 周期，WTF 置 1 一次 。ck_wut 通过 WTCS[2：0]位选择.<br>注意：禁止在 WTCS[2：0]=0b 011 时配置 WTRV=0x0000。<br>该寄存器仅在 WTWF=1 时才能写操作 |

### 17.4.7. 粗校准寄存器(RTC_COSC)

偏移地址：0x18
系统复位：无影响
备份域复位值：0x0000 0000
写保护寄存器，仅在初始化状态可以进行写操作

该寄存器只能按字(32 位)访问

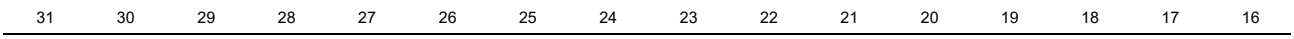



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值 |
| 7 | COSD | 粗校准方向<br>0：增加日历更新频率<br>1：降低日历更新频率 |
| 6;5 | 保留 | 必须保持复位值 |
| 4:0 | COSS[4:0] | 粗校准步伐<br>当 COSD=0：<br>0x00：+0 PPM<br>0x01：+4 PPM(近似值)<br>0x02：+8 PPM(近似值)<br>....<br>0x1F：+126 PPM(近似值)<br>当 COSD=1：<br>0x00：-0 PPM<br>0x01：-2 PPM(近似值)<br>0x02：-4 PPM(近似值)<br>....<br>0x1F：-63 PPM(近似值) |

### 17.4.8. 闹钟 0 时间日期寄存器(RTC_ALRM0TD)

偏移地址：0x1C
系统复位：无影响
备份域复位值：0x0000 0000
写保护寄存器，仅在初始化状态可以进行写操作

该寄存器只能按字(32位)访问



| MSKD | DOWS | DAYT[1:0] | | DAYU[3:0] | | | | MSKH | PM | HRT[1:0] | | HRU[3:0] | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| rw | rw | rw | | rw | | | | rw | rw | rw | | rw | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| MSKM | MNT[2:0] | | | MNU[3:0] | | | | MSKS | SCT[2:0] | | | SCU[3:0] | | | |
| rw | rw | | | rw | | | | rw | rw | | | rw | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | MSKD | 闹钟日期位域屏蔽位<br>0：不屏蔽日期/天位域<br>1：屏蔽日期/天位域 |
| 30 | DOWS | 星期选择<br>0：此时 DAYU[3：0] 代表日期个位值<br>1：此时 DAYU[3：0] 代表星期几，此时 DAYT[1：0]无意义 |
| 29:28 | DAYT[1:0] | 日期十位值，以 BCD 码格式存储 |
| 27:24 | DAYU[3:0] | 日期个位值或星期天数，以 BCD 码格式存储 |
| 23 | MSKH | 闹钟小时位域屏蔽位<br>0：不屏蔽小时位域<br>1：屏蔽小时位域 |
| 22 | PM | AM/PM 标志<br>0：AM 或 24 小时制<br>1：PM |
| 21:20 | HRT[1:0] | 小时十位值，以 BCD 码形式存储 |
| 19:16 | HRU[3:0] | 小时个位值，以 BCD 码形式存储 |
| 15 | MSKM | 闹钟分钟位域屏蔽位<br>0：不屏蔽分钟位域<br>1：屏蔽分钟位域 |
| 14:12 | MNT[2:0] | 分钟十位值，以 BCD 码形式存储 |
| 11:8 | MNU[3:0] | 分钟个位值，以 BCD 码形式存储 |
| 7 | MSKS | 闹钟秒位域屏蔽位<br>0：不屏蔽秒位域<br>1：屏蔽秒位域 |
| 6:4 | SCT[2:0] | 秒钟十位值，以 BCD 码形式存储 |
| 3:0 | SCU[3:0] | 秒钟个位值，以 BCD 码形式存储 |

### 17.4.9. 闹钟 1 时间日期寄存器(RTC_ALRM1TD)

偏移地址：0x20
系统复位：无影响

备份域复位值：0x0000 0000

写保护寄存器，仅在初始化状态可以进行写操作

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MSKD | DOWS | DAYT[1:0] | | DAYU[3:0] | | | | MSKH | PM | HRT[1:0] | | HRU[3:0] | | | |
| rw | rw | rw | | rw | | | | rw | rw | rw | | rw | | | |

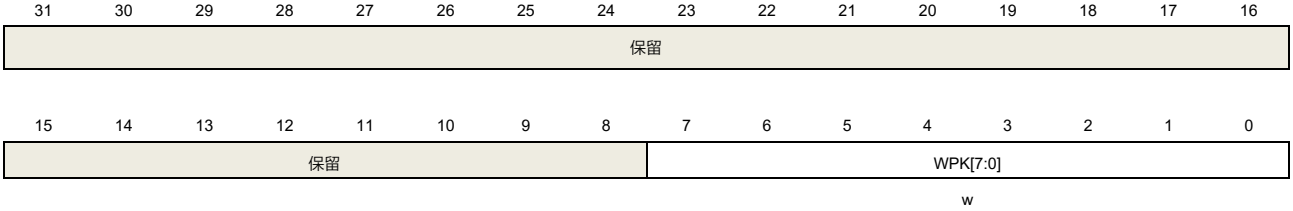
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MSKM | MNT[2:0] | | | MNU[3:0] | | | | MSKS | SCT[2:0] | | | SCU[3:0] | | | |
| rw | rw | | | rw | | | | rw | rw | | | rw | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | MSKD | 闹钟日期位域屏蔽位<br>0：不屏蔽日期/天位域<br>1：屏蔽日期/天位域 |
| 30 | DOWS | 星期选择<br>0：此时 DAYU[3：0] 代表日期个位值<br>1：此时 DAYU[3：0] 代表星期几，此时 DAYT[1：0]无意义 |
| 29:28 | DAYT[1:0] | 日期十位值，以 BCD 码格式存储 |
| 27:24 | DAYU[3:0] | 日期个位值或星期天数，以 BCD 码格式存储 |
| 23 | MSKH | 闹钟小时位域屏蔽位<br>0：不屏蔽小时位域<br>1：屏蔽小时位域 |
| 22 | PM | AM/PM 标志<br>0：AM 或 24 小时制<br>1：PM |
| 21:20 | HRT[1:0] | 小时十位值，以 BCD 码形式存储 |
| 19:16 | HRU[3:0] | 小时个位值，以 BCD 码形式存储 |
| 15 | MSKM | 闹钟分钟位域屏蔽位<br>0：不屏蔽分钟位域<br>1：屏蔽分钟位域 |
| 14:12 | MNT[2:0] | 分钟十位值，以 BCD 码形式存储 |
| 11:8 | MNU[3:0] | 分钟个位值，以 BCD 码形式存储 |
| 7 | MSKS | 闹钟秒位域屏蔽位<br>0：不屏蔽秒位域<br>1：屏蔽秒位域 |
| 6:4 | SCT[2:0] | 秒钟十位值，以 BCD 码形式存储 |
| 3:0 | SCU[3:0] | 秒钟个位值，以 BCD 码形式存储 |

### 17.4.10. 写保护钥匙寄存器(RTC_WPK)

偏移地址：0x24
复位值：0x0000 0000

该寄存器只能按字(32位)访问

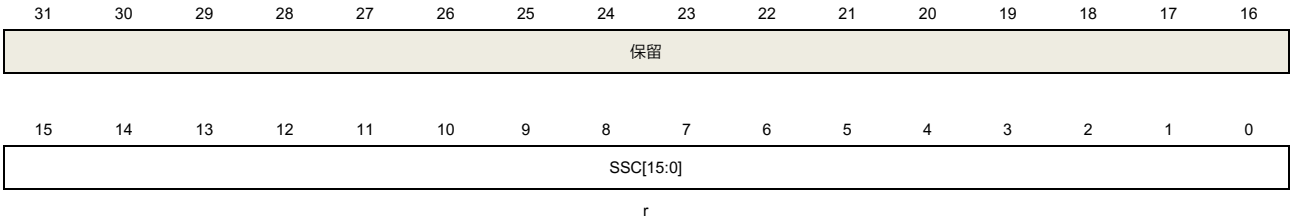



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值 |
| 7:0 | WPK[7:0] | 写保护的解锁值 |

### 17.4.11. 亚秒寄存器(RTC_SS)

偏移地址：0x28
系统复位值：当BPSHAD = 0，0x0000 0000。
当BPSHAD = 1，无影响。

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15:0 | SSC[15:0] | 亚秒值<br>该位值是同步预分频计数器的值。秒的小数部分由下面公式给出：<br>秒的小数部分 = ( FACTOR_S - SSC ) / ( FACTOR_S + 1 ) |

### 17.4.12. 移位控制寄存器(RTC_SHIFTCTL)

偏移地址：0x2C
系统复位：无影响
备份域复位值：0x0000 0000
写保护寄存器，仅当SOPF=0，该寄存器可写
**注意：**写入此寄存器会导致RSYNF位被清0。

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A1S | 保留 | | | | | | | | | | | | | | |
| w | | | | | | | | | | | | | | | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | A1S | 增加一秒<br>0：无影响<br>1：增加一秒到时钟/日历<br>该位与 SFS 位一起使用，增加小于一秒到当前时间。 |
| 30:15 | 保留 | 必须保持复位值 |
| 14:0 | SFS[14:0] | 减去小于一秒的一段时间<br>这位的值将增加到同步预分频计数器<br>当仅用 SFS 时，由于同步预分频器是一个递减计数器，所以时钟将会延迟。<br>延迟(秒) = SFS / ( FACTOR_S + 1 )<br>当 A1S 和 SFS 一起使用时，时钟将会提前<br>提前(秒) = ( 1 - ( SFS / ( FACTOR_S + 1 ) ) ) |

## 17.4.13. 时间戳时间寄存器(RTC_TTS)

偏移地址：0x30
备份域复位值：0x0000 0000
系统复位：无影响
当TSF被置1，该位用来记录日历时间
清除TSF位也会清除此寄存器

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | PM | HRT[1:0] | | HRU[3:0] | | | |
| | | | | | | | | | r | r | | r | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | MNT[2:0] | | | MNU[3:0] | | | | 保留 | SCT[2:0] | | | SCU[3:0] | | | |
| | r | | | r | | | | | r | | | r | | | |

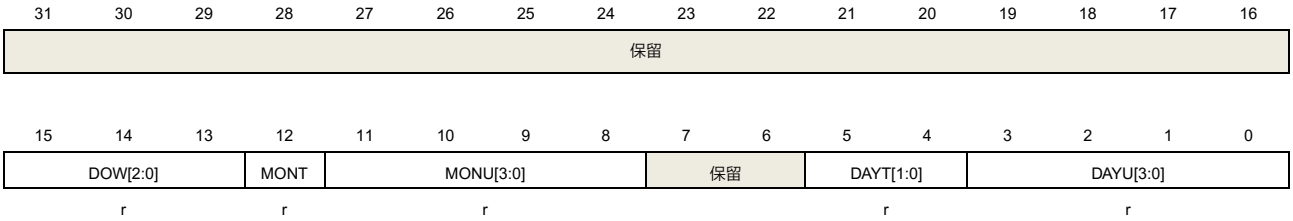
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:23 | 保留 | 必须保持复位值 |
| 22 | PM | AM/PM 标记<br>0：AM 或 24 小时制<br>1：PM |

| | | |
|---|---|---|
| 21:20 | HRT[1:0] | 小时十位值，以 BCD 码形式存储 |
| 19:16 | HRU[3:0] | 小时个位值，以 BCD 码形式存储 |
| 15 | 保留 | 必须保持复位值 |
| 14:12 | MNT[2:0] | 分钟十位值，以 BCD 码形式存储 |
| 11:8 | MNU[3:0] | 分钟个位值，以 BCD 码形式存储 |
| 7 | 保留 | 必须保持复位值 |
| 6:4 | SCT[2:0] | 秒钟十位值，以 BCD 码形式存储 |
| 3:0 | SCU[3:0] | 秒钟个位值，以 BCD 码形式存储 |

### 17.4.14. 时间戳日期寄存器(RTC_DTS)

偏移地址：0x34
备份域复位值：0x0000 0000
系统复位：无影响
当TSF被置1，该位用来记录日历日期
清除TSF位也会清除此寄存器

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15:13 | DOW[2:0] | 星期数 |
| 12 | MONT | 月份十位值，以 BCD 码形式存储 |
| 11:8 | MONU[3:0] | 月份个位值，以 BCD 码形式存储 |
| 7:6 | 保留 | 必须保持复位值 |
| 5:4 | DAYT[2:0] | 日期十位值，以 BCD 码形式存储 |
| 3:0 | DAYU[3:0] | 日期个位值，以 BCD 码形式存储 |

### 17.4.15. 时间戳亚秒寄存器(RTC_SSTS)

偏移地址：0x38
备份域复位：0x0000 0000

系统复位：无影响
当 TSF 被置 1，该位用来记录日历时间
清除 TSF 位也会清除此寄存器

该寄存器只能按字(32 位)访问

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15:0 | SSC[15:0] | 亚秒值<br>TSF 置 1 时记录当时的同步预分频计数器的值。 |

## 17.4.16. 高精度频率补偿寄存器(RTC_HRFC)

偏移地址：0x3C
备份域复位：0x0000 0000
系统复位：无影响
写保护寄存器

该寄存器只能按字(32 位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FREQI | CWND8 | CWND16 | 保留 | | | | CMSK[8:0] | | | | | | | | |
| rw | rw | rw | | | | | rw | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15 | FREQI | RTC 频率增加 488.5ppm<br>0：无影响<br>1：每 $2^{11}$ 个脉冲增加一个 RTCCLK 脉冲<br>该位需与 CMSK 位一起使用。如果输入时钟频率是 32.768KHz，在 32s 校准窗期间，增加的 RTCCLK 脉冲数是(512 * FREQI) - CMSK |
| 14 | CWND8 | 采用 8 秒校准周期<br>0：无影响<br>1：采用 8 秒校准周期<br>注意：当 CWND8=1，CMSK[1：0]被锁定在"00"。 |

| | | |
|---|---|---|
| 13 | CWND16 | 采用 16 秒校准周期<br>0：无影响<br>1：采用 16 秒校准周期<br>注意：当 CWND16=1，CMSK[0]被锁定在"0". |
| 12:9 | 保留 | 必须保持复位值 |
| 8:0 | CMSK[8:0] | 校准周期 RTCCLK 脉冲屏蔽数<br>在 $2^{20}$ RTCCLK 脉冲之内屏蔽的脉冲数<br>此项功能可以以 0.9537ppm 的分辨率来降低日历频率 |

## 17.4.17. 侵入寄存器(RTC_TAMP)

偏移地址：0x40
备份域复位：0x0000 0000
系统复位：无影响

该寄存器只能按字(32 位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | | AOT | TSSEL | TP0SEL |
| | | | | | | | | | | | | | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DISPU | PRCH[1:0] | | FLT[1:0] | | FREQ[2:0] | | | TPTS | 保留 | | TP1EG | TP1EN | TPIE | TP0EG | TP0EN |
| rw | rw | | rw | | rw | | | rw | | | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:19 | 保留 | 必须保持复位值 |
| 18 | AOT | RTC_ALARM 输出类型<br>0：开漏输出<br>1：推挽输出 |
| 17 | TSSEL | 时间戳输入选择：<br>0：时间戳从 PC13 输入<br>1：时间戳从 PI8 输入 |
| 16 | TP0SEL | Tamper 0 输入选择：<br>0：Tamper 0 从 PC13 输入<br>1：Tamper 0 从 PI8 输入<br>注意：当 TP0SEL 改变时 TP0EN 必须复位 |
| 15 | DISPU | RTC_TAMPx 上拉禁用位<br>0：使能内部 RTC_TAMPx 引脚上的上拉电阻并在采样前进行预充电<br>1：禁用预充电功能 |
| 14:13 | PRCH[1:0] | RTC_TAMPx 的预充电时间<br>该位设置决定了每次采样前的预充电时间<br>0x0：1 个 RTC 时钟<br>0x1：2 个 RTC 时钟 |

| | | |
|---|---|---|
| | | 0x2：4 个 RTC 时钟<br>0x3：8 个 RTC 时钟 |
| 12:11 | FLT[1:0] | RTC_TAMPx 过滤器计数设置<br>该位决定了侵入事件检测模式和在电平检测模式下连续采样的次数。<br>0x0：用边沿模式检测侵入事件，预充电功能被自动禁用。<br>0x1：用电平模式检测侵入事件。连续采样到 2 个有效电平时认为发生侵入事件<br>0x2：用电平模式检测侵入事件。连续采样到 4 个有效电平时认为发生侵入事件<br>0x3：用电平模式检测侵入事件。连续采样到 8 个有效电平时认为发生侵入事件 |
| 10:8 | FREQ[2:0] | 侵入事件电平模式检测的采样频率<br>0x0：每次采样间隔 32768 个 RTCCLK(若 RTCCLK=32.768KHz，频率为 1Hz)<br>0x1：每次采样间隔 16384 个 RTCCLK(若 RTCCLK=32.768KHz，频率为 2Hz)<br>0x2：每次采样间隔 8192 个 RTCCLK(若 RTCCLK=32.768KHz，频率为 4Hz)<br>0x3：每次采样间隔 4096 个 RTCCLK(若 RTCCLK=32.768KHz，频率为 8Hz)<br>0x4：每次采样间隔 2048 个 RTCCLK(若 RTCCLK=32.768KHz，频率为 16Hz)<br>0x5：每次采样间隔 1024 个 RTCCLK(若 RTCCLK=32.768KHz，频率为 32Hz)<br>0x6：每次采样间隔 512 个 RTCCLK(若 RTCCLK=32.768KHz，频率为 64Hz)<br>0x7：每次采样间隔 256 个 RTCCLK(若 RTCCLK=32.768KHz，频率为 128Hz) |
| 7 | TPTS | 侵入事件时触发时间戳<br>0：无影响<br>1：当检测到侵入事件时，即使 TSEN=0，TSF 也会被置位 |
| 6:5 | 保留 | 必须保持复位值 |
| 4 | TP1EG | TAMP1 输入管脚的侵入事件检测触发沿<br>如果侵入检测处于边沿模式(FLT = 0)：<br>0：上升沿触发一个侵入检测事件<br>1：下降沿触发一个侵入检测事件<br>如果侵入检测处于电平模式(FLT != 0)：<br>0：低电平触发一个侵入检测事件<br>1：高电平触发一个侵入检测事件 |
| 3 | TP1EN | Tamper1 检测使能位<br>0：禁用 Tamper1 检测功能<br>1：启用 Tamper1 检测功能 |
| 2 | TPIE | 侵入检测中断使能<br>0：禁用侵入中断<br>1：启用侵入中断 |
| 1 | TP0EG | TAMP0 输入管脚的侵入事件检测触发沿<br>如果侵入检测处于边沿模式(FLT = 0)：<br>0：上升沿触发一个侵入检测事件<br>1：下降沿触发一个侵入检测事件<br>如果侵入检测处于电平模式(FLT != 0)：<br>0：低电平触发一个侵入检测事件 |

1：高电平触发一个侵入检测事件

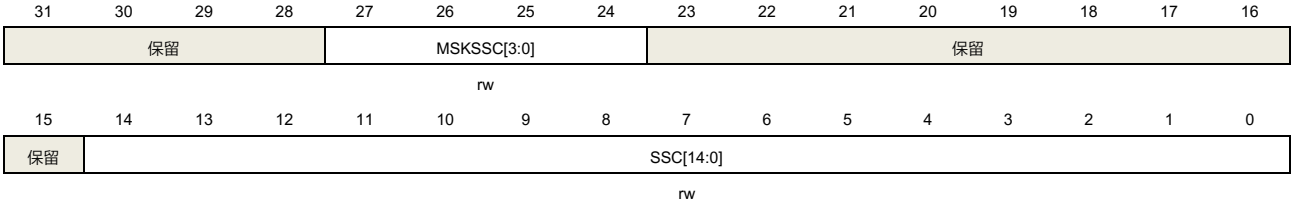
| | | |
|---|---|---|
| 0 | TP0EN | Tamper0 检测使能位<br>0：禁用 Tamper0 检测功能<br>1：启用 Tamper0 检测功能 |

**注意：**强烈建议在改变侵入检测配置之前，应该复位TPxEN位。

## 17.4.18. 闹钟 0 亚秒寄存器(RTC_ALRM0SS)

偏移地址：0x44
备份域复位：0x0000 0000
系统复位：无影响
写保护寄存器，仅当 ALRM0EN=0 或 INITM=1，可以进行写操作

该寄存器只能按字(32 位)访问



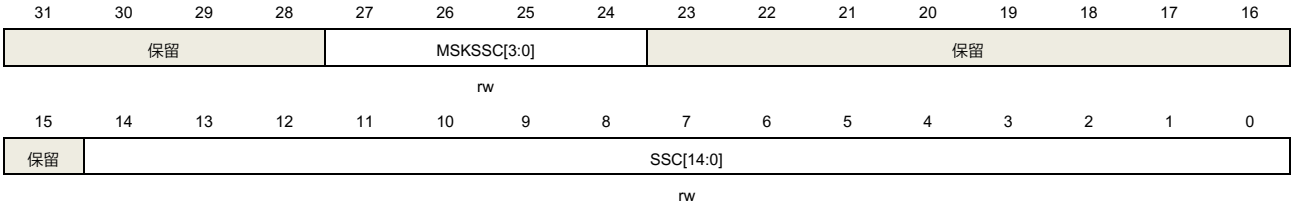

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值 |
| 27:24 | MSKSSC[3:0] | 亚秒位域的屏蔽控制位<br>0x0：屏蔽闹钟亚秒设置。当所有其他的闹钟位域匹配的时候，闹钟将会在每一秒钟到达的时刻置 1。<br>0x1：SSC[0]位用于时间匹配，其他位被忽略。<br>0x2：SSC[1：0] 位用于时间匹配，其他位被忽略。<br>0x3：SSC[2：0] 位用于时间匹配，其他位被忽略。<br>0x4：SSC[3：0] 位用于时间匹配，其他位被忽略。<br>0x5：SSC[4：0] 位用于时间匹配，其他位被忽略。<br>0x6：SSC[5：0] 位用于时间匹配，其他位被忽略。<br>0x7：SSC[6：0] 位用于时间匹配，其他位被忽略。<br>0x8：SSC[7：0] 位用于时间匹配，其他位被忽略。<br>0x9：SSC[8：0] 位用于时间匹配，其他位被忽略。<br>0x10：SSC[9：0] 位用于时间匹配，其他位被忽略。<br>0x11：SSC[10：0] 位用于时间匹配，其他位被忽略。<br>0x12：SSC[11：0] 位用于时间匹配，其他位被忽略。<br>0x13：SSC[12：0] 位用于时间匹配，其他位被忽略。<br>0x14：SSC[13：0] 位用于时间匹配，其他位被忽略。<br>0x15：SSC[14：0] 位用于时间匹配，其他位被忽略。<br>注意：同步预分频计数器的第 15 位(RTC_SS 寄存器中的 SSC[15])从不被匹配。 |
| 23:15 | 保留 | 必须保持复位值 |

| | | |
|---|---|---|
| 14:0 | SSC[14:0] | 闹钟亚秒值<br>该值为闹钟亚秒值，用于与同步预分频计数器匹配。<br>匹配位数由 MSKSSC 位控制。 |

## 17.4.19. 闹钟 1 亚秒寄存器(RTC_ALRM1SS)

偏移地址：0x48
备份域复位：0x0000 0000
系统复位：无影响
写保护寄存器，仅当 ALRM1EN=0 或 INITM=1，可以进行写操作

该寄存器只能按字(32 位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值 |
| 27:24 | MSKSSC[3:0] | 亚秒位域的屏蔽控制位<br>0x0：屏蔽闹钟亚秒设置。当所有其他的闹钟位域匹配的时候，闹钟将会在每一秒钟到达的时刻置 1。<br>0x1：SSC[0]位用于时间匹配，其他位被忽略。<br>0x2：SSC[1：0]位用于时间匹配，其他位被忽略。<br>0x3：SSC[2：0]位用于时间匹配，其他位被忽略。<br>0x4：SSC[3：0]位用于时间匹配，其他位被忽略。<br>0x5：SSC[4：0]位用于时间匹配，其他位被忽略。<br>0x6：SSC[5：0]位用于时间匹配，其他位被忽略。<br>0x7：SSC[6：0]位用于时间匹配，其他位被忽略。<br>0x8：SSC[7：0]位用于时间匹配，其他位被忽略。<br>0x9：SSC[8：0]位用于时间匹配，其他位被忽略。<br>0x10：SSC[9：0]位用于时间匹配，其他位被忽略。<br>0x11：SSC[10：0]位用于时间匹配，其他位被忽略。<br>0x12：SSC[11：0]位用于时间匹配，其他位被忽略。<br>0x13：SSC[12：0]位用于时间匹配，其他位被忽略。<br>0x14：SSC[13：0]位用于时间匹配，其他位被忽略。<br>0x15：SSC[14：0]位用于时间匹配，其他位被忽略。<br>注意：同步预分频计数器的第 15 位(RTC_SS 寄存器中的 SSC[15])从不被匹配。 |
| 23:15 | 保留 | 必须保持复位值 |
| 14:0 | SSC[14:0] | 闹钟亚秒值<br>该值为闹钟亚秒值，用于与同步预分频计数器匹配。 |

匹配位数由 MSKSSC 位控制。

## 17.4.20. 备份寄存器(RTC_BKPx)(x=0..19)

偏移地址：0x50 到 0x9C
备份域复位：0x0000 0000
系统复位：无影响

该寄存器只能按字(32 位)访问

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | DATA[31:0] | 数据<br>软件可读写寄存器。由于此寄存器可由 $V_{BAT}$ 供电，因此寄存器值在省电模式下依然保持有效。当侵入检测标志位 TPxF 置 1，这些寄存器会被复位。 |

# 18. 定时器(TIMERx)

**表 18-1. 定时器(TIMERx)分为五种类型**

| 定时器 | 定时器 0/7 | 定时器 1~4 | 定时器 8/11 | 定时器 9/10/12/13 | 定时器 5/6 |
|---|---|---|---|---|---|
| **类型** | 高级 | 通用(L0) | 通用(L1) | 通用(L2) | 基本 |
| **预分频器** | 16 位 | 16 位 | 16 位 | 16 位 | 16 位 |
| **计数器** | 16 位 | 32 位(定时器 1/4)<br>16 位(定时器 2/3) | 16 位 | 16 位 | 16 位 |
| **计数模式** | 向上，向下，中央对齐 | 向上，向下，中央对齐 | 只有向上 | 只有向上 | 只有向上 |
| **可重复性** | ● | × | × | × | × |
| **捕获/比较通道数** | 4 | 4 | 2 | 1 | 0 |
| **互补和死区时间** | ● | × | × | × | × |
| **中止输入** | ● | × | × | × | × |
| **单脉冲** | ● | ● | ● | × | ● |
| **正交译码器** | ● | ● | × | × | × |
| **主-从管理** | ● | ● | ● | × | × |
| **内部连接** | ●(1) | ●(2) | ●(3) | × | TRGO TO DAC |
| **DMA** | ● | ● | × | × | ●(4) |
| **Debug 模式** | ● | ● | ● | ● | ● |

(1)

| | | | | |
|---|---|---|---|---|
| TIMER0 | **ITI0:**TIMER4_TRGO | **ITI1:** TIMER1_TRGO | **ITI2:** TIMER2_TRGO | **ITI3:**TIMER3_TRGO |
| TIMER7 | **ITI0:** TIMER0_TRGO | **ITI1:** TIMER1_TRGO | **ITI2:** TIMER3_TRGO | **ITI3:**TIMER4_TRGO |

(2)

| | | | | |
|---|---|---|---|---|
| TIMER1 | **ITI0:** TIMER0_TRGO | **ITI1:** TIMER7_TRGO | **ITI2:** TIMER2_TRGO | **ITI3:**TIMER3_TRGO |
| TIMER2 | **ITI0:** TIMER0_TRGO | **ITI1:** TIMER1_TRGO | **ITI2:** TIMER4_TRGO | **ITI3:**TIMER3_TRGO |
| TIMER3 | **ITI0:** TIMER0_TRGO | **ITI1:** TIMER1_TRGO | **ITI2:** TIMER2_TRGO | **ITI3:**TIMER7_TRGO |
| TIMER4 | **ITI0:** TIMER1_TRGO | **ITI1:** TIMER2_TRGO | **ITI2:** TIMER3_TRGO | **ITI3:**TIMER7_TRGO |

(3)

| | | | | |
|---|---|---|---|---|
| TIMER8 | **ITI0:** TIMER1_TRGO | **ITI1:** TIMER2_TRGO | **ITI2:** TIMER9_TRGO | **ITI3:**TIMER10_TRGO |
| TIMER11 | **ITI0:** TIMER3_TRGO | **ITI1:** TIMER4_TRGO | **ITI2:** TIMER12_TRGO | **ITI3:**TIMER13_TRGO |

(4) 只有更新事件可以产生 DMA 请求。但是定时器 5 和定时 6 中没有 DMA 配置寄存器。

# 18.1. 高级定时器(TIMERx,x=0,7)

## 18.1.1. 简介

高级定时器(TIMER0 和 TIMER7)是四通道定时器，支持输入捕获和输出比较。可以产生 PWM 信号控制电机和电源管理。高级定时器含有一个 16 位无符号计数器。

高级定时器是可编程的，可以被用来计数，其外部事件可以驱动其他定时器

高级定时器包含了一个死区时间插入模块，非常适合电机控制。

定时器和定时器之间是相互独立，但是他们可以被同步在一起形成一个更大的定时器，这些定时器的计数器一致地增加。

## 18.1.2. 主要特性

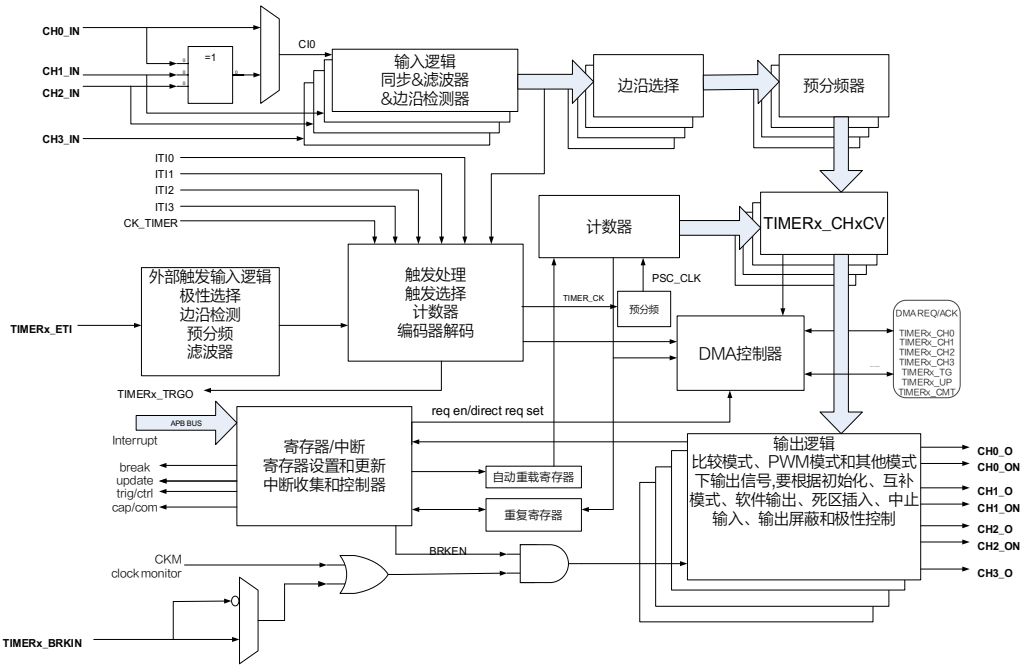
- 总通道数：4；
- 计数器宽度：16位；
- 时钟源可选：内部时钟，内部触发，外部输入，外部触发；
- 多种计数模式：向上计数，向下计数和中央计数；
- 正交译码器接口：被用来追踪运动和分辨旋转方向和位置；
- 霍尔传感器接口：用来做三相电机控制；
- 可编程的预分频器：16位，运行时可以被改变；
- 每个通道可配置：输入捕获模式，输出比较模式，可编程的PWM模式，单脉冲模式；
- 可编程的死区时间；
- 自动重装载功能；
- 可编程的计数器重复功能；
- 中止输入功能使定时器按照用户的配置输出；
- 中断输出和DMA请求：更新事件，触发事件，比较/捕获事件和中止事件；
- 多个定时器的菊链使得一个定时器可以同时启动多个定时器；
- 定时器的同步允许被选择的定时器在同一个时钟周期开始计数；
- 定时器主-从管理。

## 18.1.3. 功能描述

### 结构框图

*图 18-1. 高级定时器结构框图*了高级定时器的内部配置细节。

**图 18-1. 高级定时器结构框图**




## 时钟源配置

高级定时器可以由内部时钟源 CK_TIMER 或者由 SMC(TIMERx_SMCFG 寄存器位[2:0])控制的复用时钟源驱动。

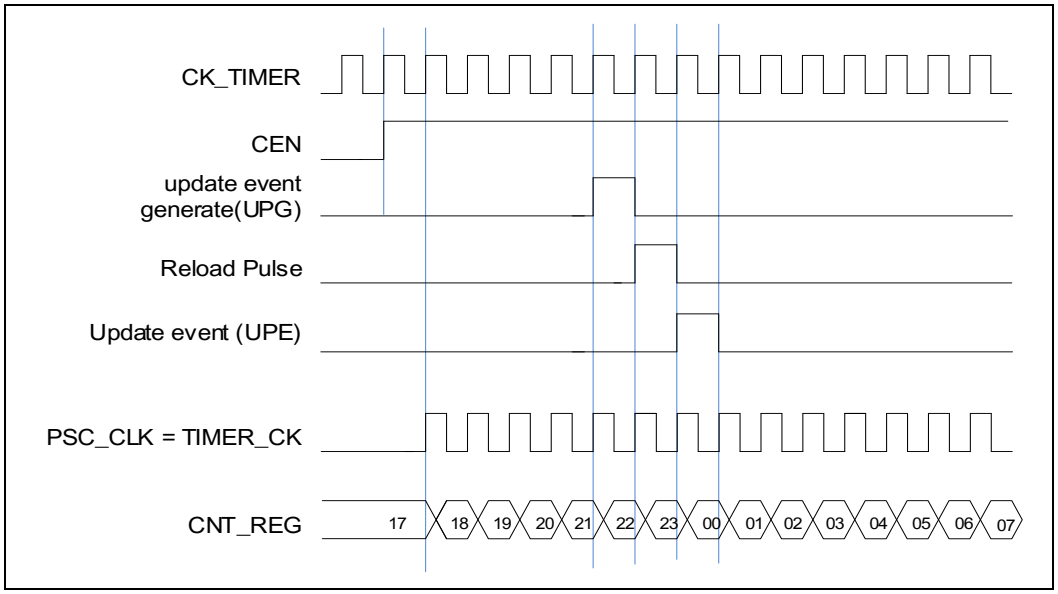
■ SMC[2:0]==3'b000，定时器选择内部时钟源(连接到RCU模块的CK_TIMER)

如果 SMC[2:0]==3'b000，默认用来驱动计数器预分频器的是内部时钟源 CK_TIMER。当 CEN 置位，CK_TIMER 经过预分频器(预分频值由 TIMERx_PSC 寄存器确定)产生 PSC_CLK。

这种模式下，驱动预分频器计数的 TIMER_CK 等于来自于 RCU 模块的 CK_TIMER

如果将 TIMERx_SMCFG 寄存器的 SMC[2:0]设置为 0x1、0x2、0x3 和 0x7，预分频器被其他时钟源(由 TIMERx_SMCFG 寄存器的 TRGS [2:0]区域选择)驱动，在下文说明。当 SMC 位被设置为 0x4、0x5 和 0x6，计数器预分频器时钟源由内部时钟 CK_TIMER 驱动。

**图 18-2. 内部时钟分频为 1 时，计数器的时序图**




■ SMC[2:0]==3'b111(外部时钟模式0)，定时器选择外部输入引脚作为时钟源

计数器预分频器可以在 TIMERx_CI0/ TIMERx_CI1 引脚的每个上升沿或下降沿计数。这种模式可以通过设置 SMC [2:0]为 0x7 同时设置 TRGS[2:0]为 0x4，0x5 或 0x6 来选择。

计数器预分频器也可以在内部触发信号 ITI0/1/2/3 的上升沿计数。这种模式可以通过设置 SMC [2:0]为 0x7 同时设置 TRGS [2:0]为 0x0, 0x1, 0x2 或者 0x3。

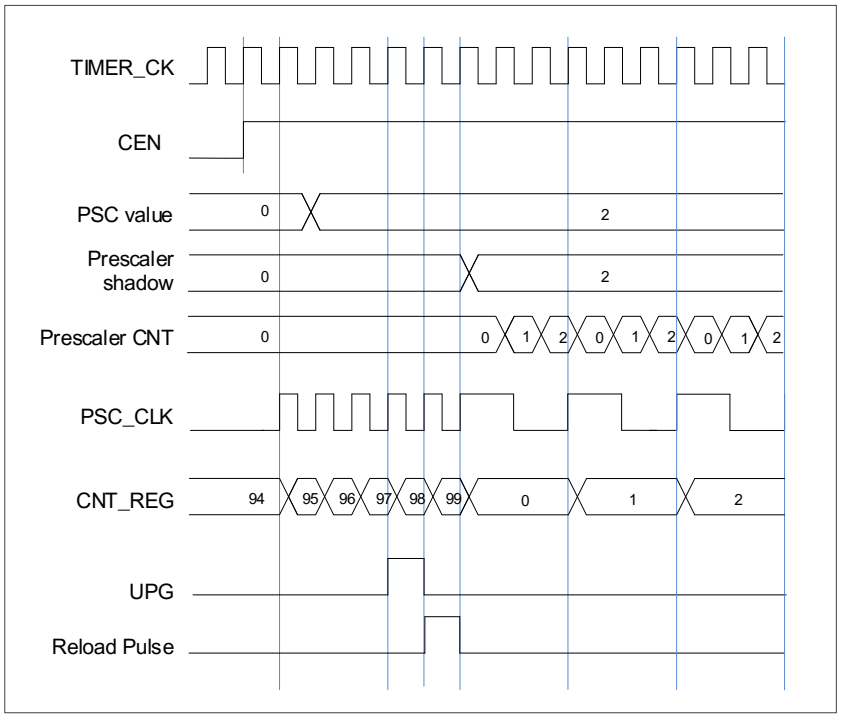
■ SMC1==1'b1(外部时钟模式1)，定时器选择外部输入引脚ETI作为时钟源

计数器预分频器可以在外部引脚 ETI 的每个上升沿或下降沿计数。这种模式可以通过设置 TIMERx_SMCFG 寄存器中的 SMC1 位为 1 来选择。另一种选择 ETI 信号作为时钟源方式是，设置 SMC [2:0]为 0x7 同时设置 TRGS [2:0]为 0x7。注意 ETI 信号是通过数字滤波器采样 ETI 引脚得到的。如果选择 ETI 信号为时钟源，触发控制器包括边沿监测电路将在每个 ETI 信号上升沿产生一个时钟脉冲来为计数器预分频器提供时钟。

### 时钟预分频器

预分频器可以将定时器的时钟(TIMER_CK)频率按 1 到 65536 之间的任意值分频，分频后的时钟 PSC_CLK 驱动计数器计数。分频系数受预分频寄存器 TIMERx_PSC 控制，这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

**图 18-3. 当 PSC 数值从 0 变到 2 时，计数器的时序图**




### 计数器向上计数模式

在这种模式，计数器的计数方向是向上计数。计数器从 0 开始向上连续计数到自动加载值(定义在 TIMERx_CAR 寄存器中)，一旦计数器计数到自动加载值，会重新从 0 开始向上计数。另外，在(TIMERx_CREP+1)次上溢后产生更新事件。在向上计数模式中，TIMERx_CTL0 寄存器中的计数方向控制位 DIR 应该被设置成 0。

当通过 TIMERx_SWEVG 寄存器的 UPG 位置 1 来设置更新事件时，计数值会被清 0，并产生更新事件。

如果 TIMERx_CTL0 寄存器的 UPDIS 置 1，则禁止更新事件。

当发生更新事件时，所有影子寄存器(重复计数器，计数器自动重载寄存器，预分频寄存器)都将被更新。

下面这些图给出了一些例子，当 TIMERx_CAR=0x99 时，计数器在不同预分频因子下的行为。

图 18-4. 向上计数时序图，PSC=0/2

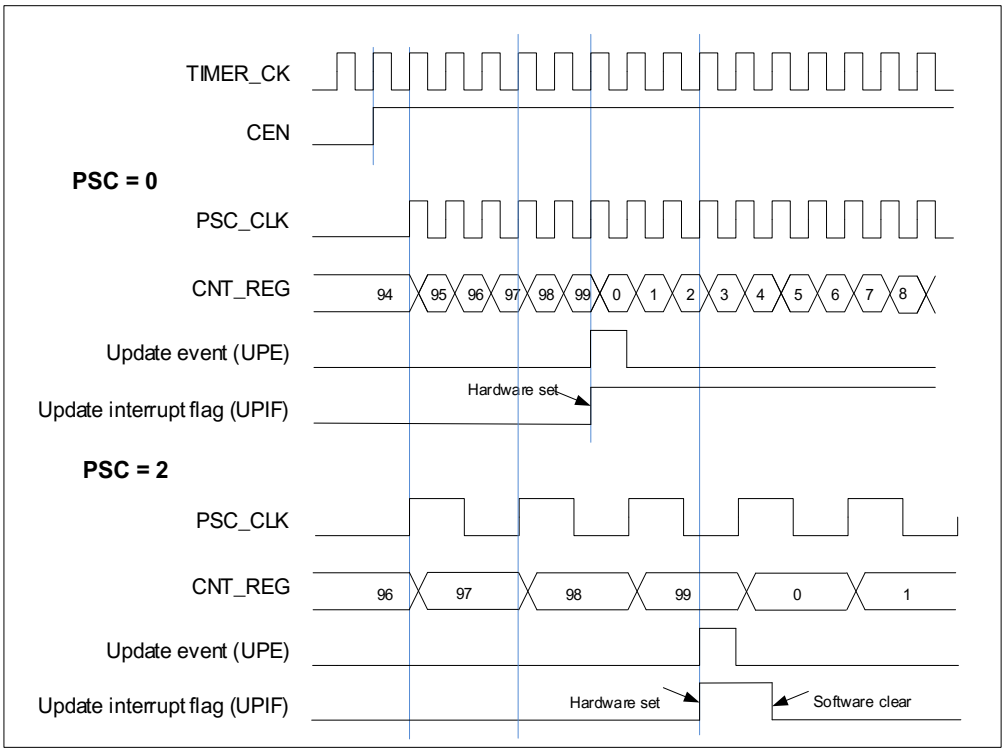
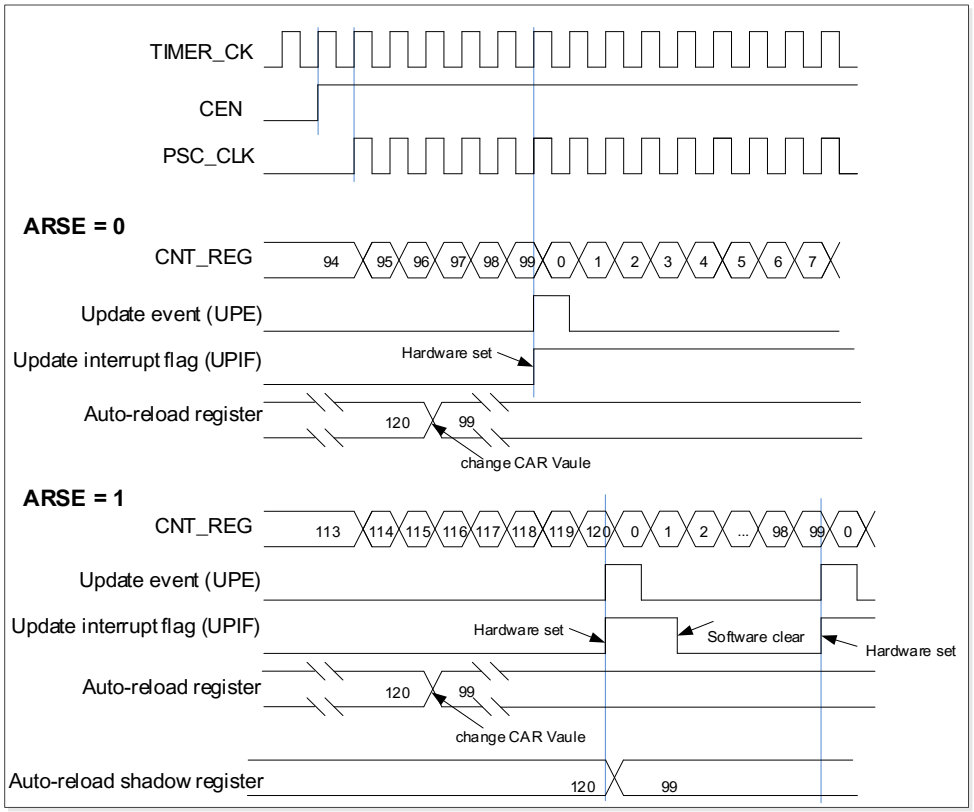



图 18-5. 向上计数时序图，在运行时改变 TIMERx_CAR 寄存器的值



### 计数器向下计数模式

在这种模式，计数器的计数方向是向下计数。计数器从自动加载值(定义在 TIMERx_CAR 寄存器中)向下连续计数到 0。一旦计数器计数到 0，计数器会重新从自动加载值开始计数。另外，在(TIMERx_CREP+1)次下溢后产生更新事件。在向下计数模式中，TIMERx_CTL0 寄存器中的计数方向控制位 DIR 应该被设置成 1。

当通过 TIMERx_SWEVG 寄存器的 UPG 位置 1 来设置更新事件时，计数值会被初始化为自动加载值，并产生更新事件。

如果 TIMERx_CTL0 寄存器的 UPDIS 置 1，则禁止更新事件。

当发生更新事件时，所有影子寄存器(重复计数器，计数器自动重载寄存器，预分频寄存器)都将被更新。

下面这些图给出了一些例子，当 TIMERx_CAR=0x99 时，计数器在不同时钟频率下的行为。

**图 18-6. 向下计数时序图，PSC=0/2**

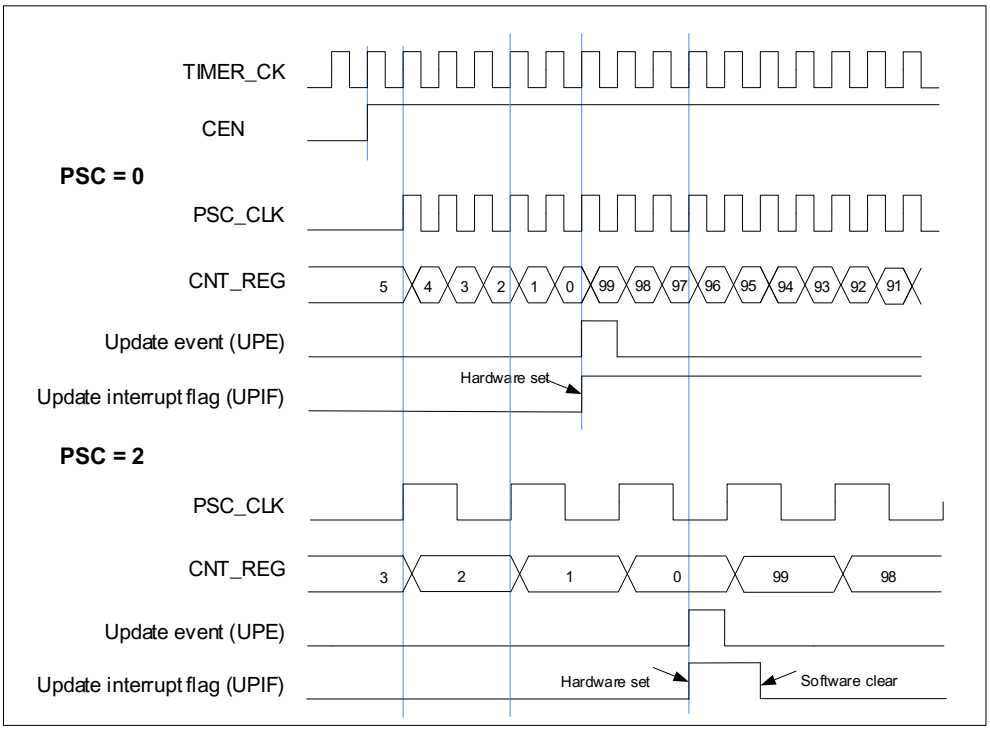
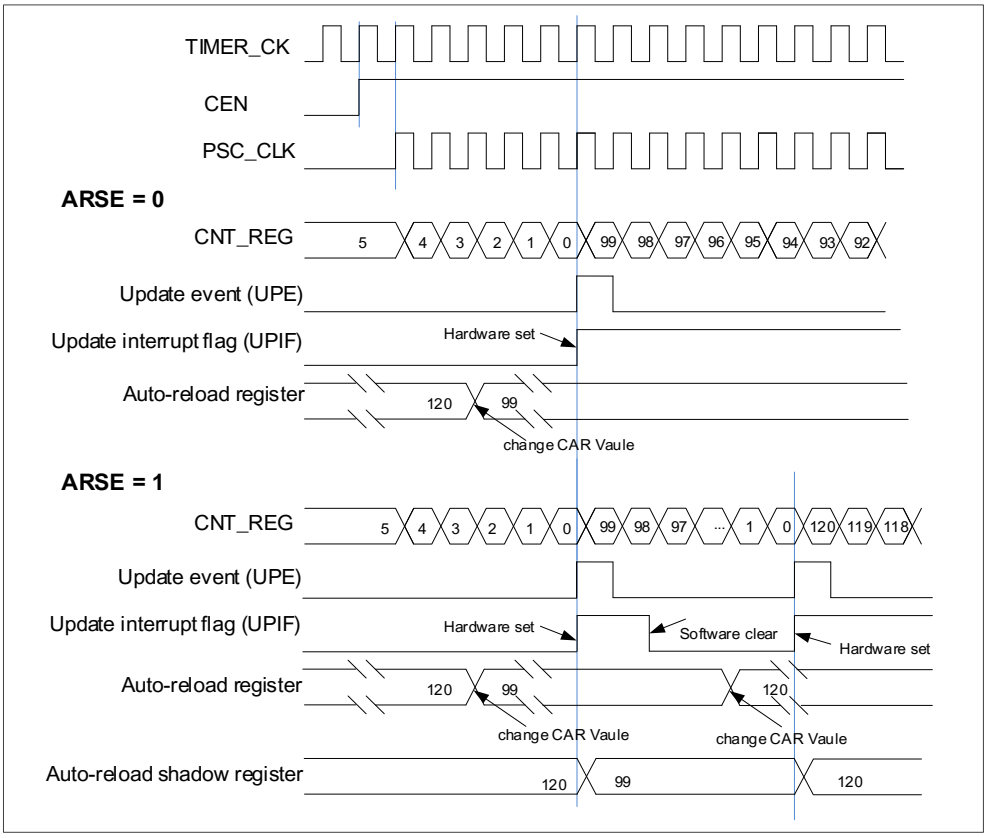


图 18-7. 向下计数时序图，在运行时改变 TIMERx_CAR 寄存器值




### 计数器中央对齐模式

在中央对齐模式下，计数器交替的从 0 开始向上计数到自动加载值，然后再向下计数到 0。。向上计数模式中， 定时器模块在计数器计数到自动加载值-1 产生一个上溢事件；向下计数模式中，定时器模块在计数器计数到 1 时产生一个下溢事件。在中央计数模式中，TIMERx_CTL0 寄存器中的计数方向控制位 DIR 只读，表明了的计数方向。

将 TIMERx_SWEVG 寄存器的 UPG 位置 1 可以初始化计数值为 0，并产生一个更新事件，而无需考虑计数器在中央模式下是向上计数还是向下计数。

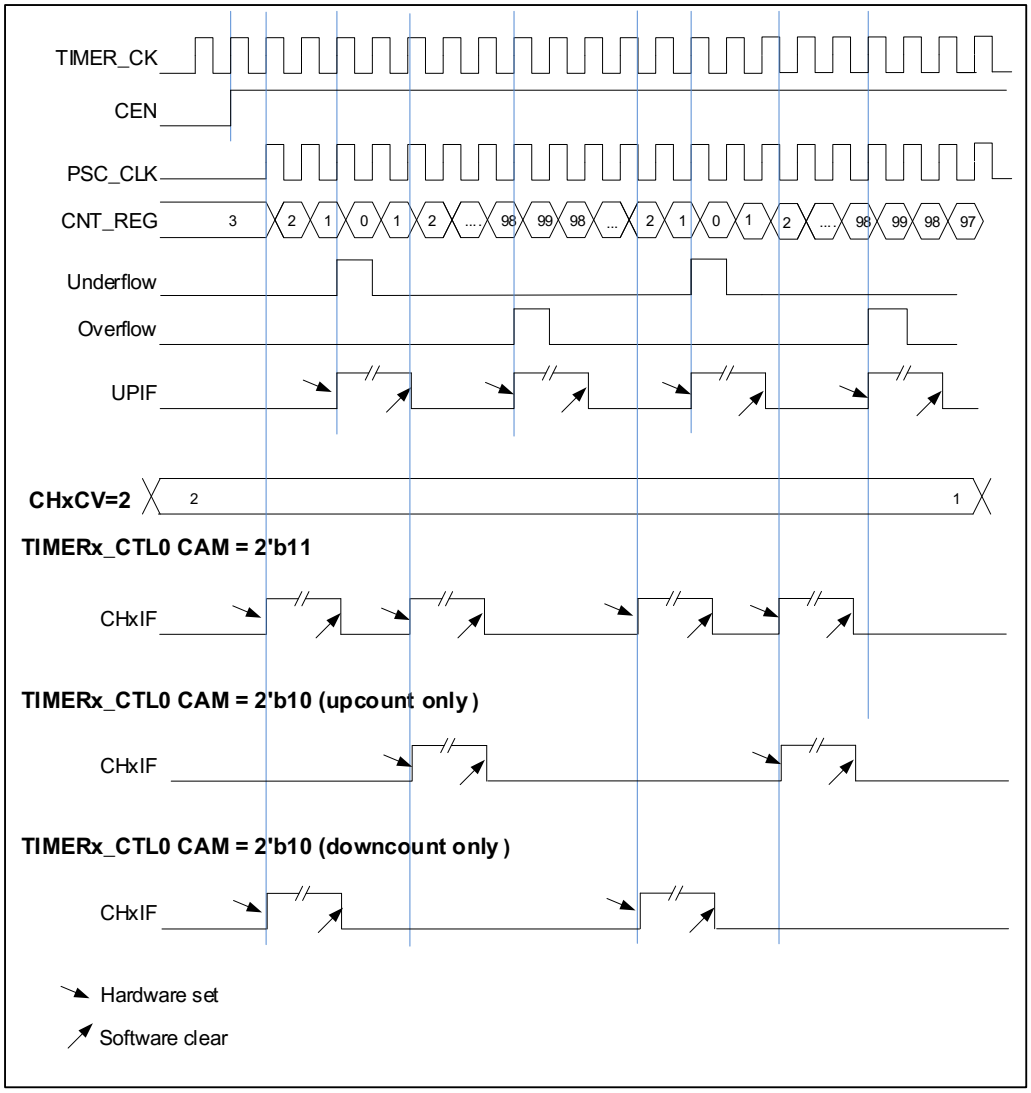
上溢或者下溢时，TIMERx_INTF 寄存器中的 UPIF 位都会被置 1，然而 CHxIF 位置 1 与 TIMERx_CTL0 寄存器中 CAM 的值有关。具体细节参考*图 18-8. 中央计数模式计数器时序图*。

如果 TIMERx_CTL0 寄存器的 UPDIS 置 1，则禁止更新事件。

当发生更新事件时，所有影子寄存器(重复计数器，计数器自动重载寄存器，预分频寄存器)都将被更新。

*图 18-8. 中央计数模式计数器时序图* 给出了一些例子，当 TIMERx_CAR=0x99，TIMERx_PSC=0x0 时，计数器的行为。

图 18-8. 中央计数模式计数器时序图




## 更新事件(来自上溢/下溢)频率配置

更新事件的生成频率(来自上溢和下溢事件)可以通过TIMERx_CREP寄存器进行配置。重复计数器是用来在N+1个计数周期之后产生更新事件，更新定时器的寄存器，N为TIMERx_CREP寄存器的CREP。重复计数器在每次计数器上溢和下溢时递减(向上计数模式中不存在下溢事件；向下计数模式中不存在上溢事件)。

将TIMERx_SWEVG寄存器的UPG位置1可以重载TIMERx_CREP寄存器中CREP的值并产生一个更新事件。

新写入的CREP值将在下一次更新事件到来时生效。当CREP的值为奇数，并且计数器在中央对齐模式下计数时，更新事件发生在上溢或下溢取决于写入的CREP值何时生效。如果在写入奇数到CREP寄存器后由软件生成更新事件(UPG位置1)，则在下溢时产生更新事件。如果在写入奇数到CREP寄存器后下一个更新事件发生在上溢，此后将在上溢时产生更新事件。

图 18-9. 中央计数模式下计数器重复时序图

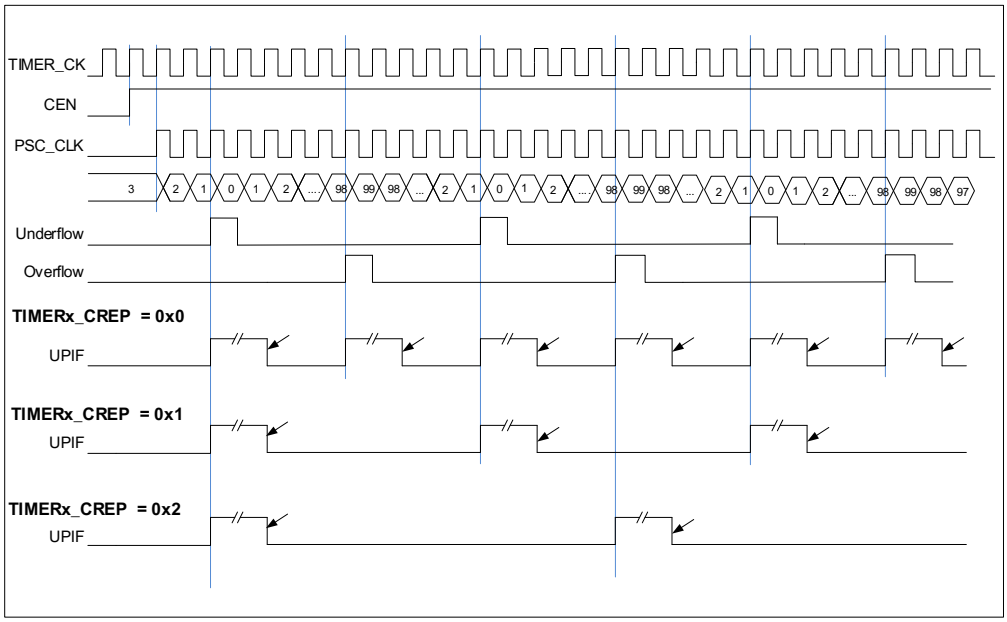



图 18-10. 在向上计数模式下计数器重复时序图

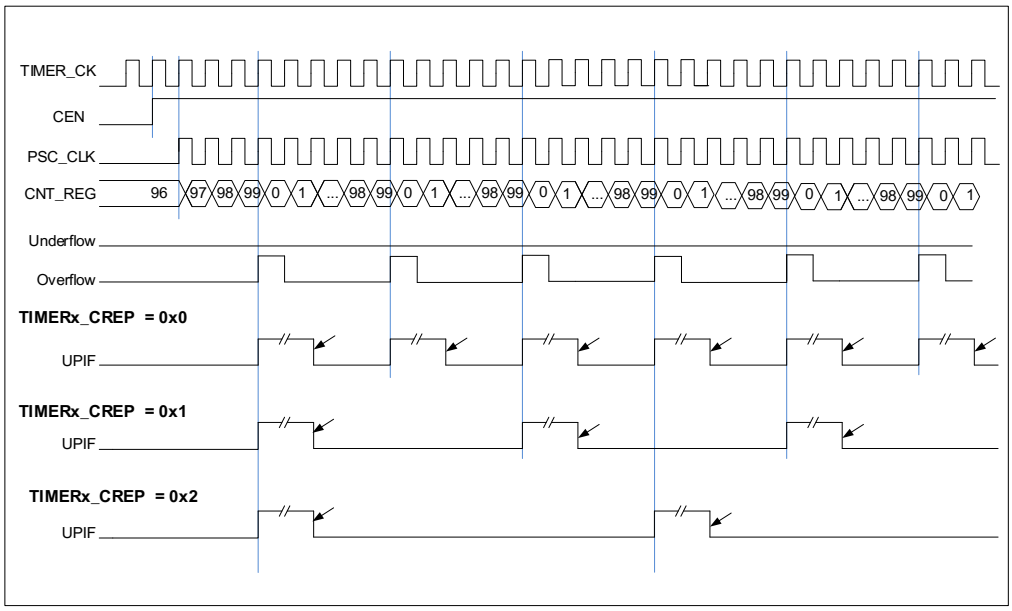
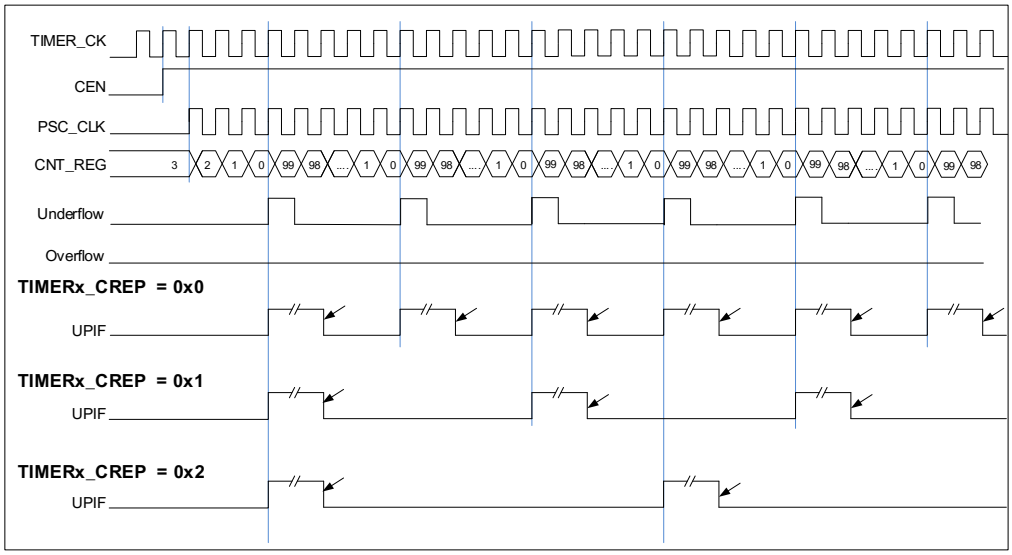


图 18-11. 在向下计数模式下计数器重复时序图



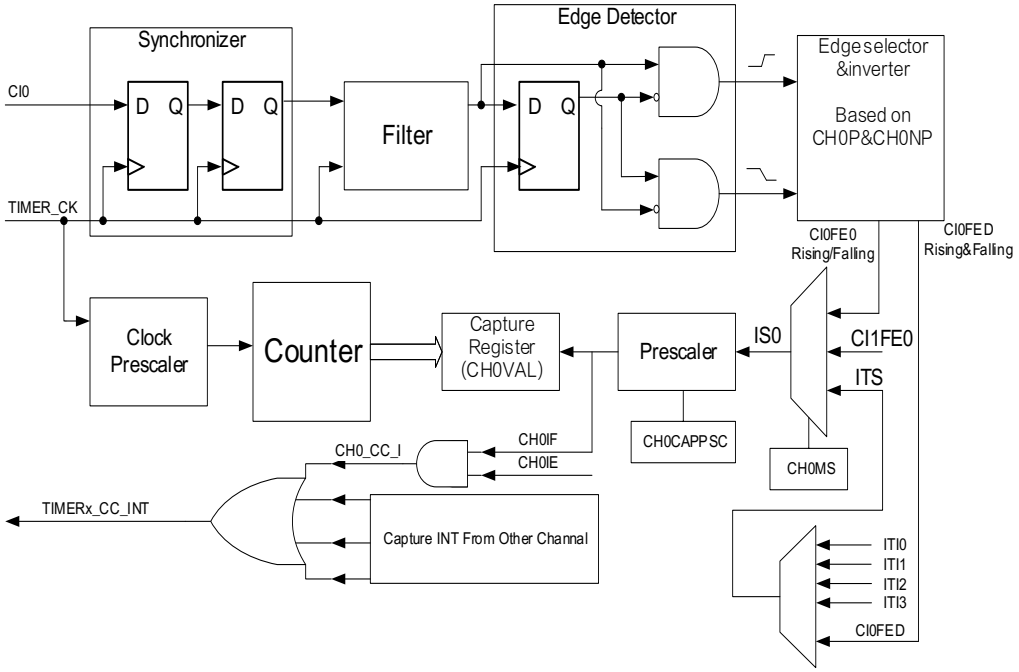

## 输入捕获和输出比较通道

高级定时器拥有四个独立的通道用于捕获输入或比较输出是否匹配。每个通道都围绕一个通道捕获比较寄存器建立，包括一个输入级，通道控制器和输出级。

■ 通道输入捕获功能

通道输入捕获功能允许通道测量一个波形时序，频率，周期，占空比等。输入级包括一个数字滤波器，一个通道极性选择，边沿检测和一个通道预分频器。如果在输入引脚上出现被选择的边沿，TIMERx_CHxCV 寄存器会捕获计数器当前的值，同时 CHxIF 位被置 1，如果 CHxIE = 1 则产生通道中断。

**图 18-12. 通道输入捕获原理**




通道输入信号 CIx 有两种选择，一种是 TIMERx_CHx 信号，另一种是 TIMERx_CH0,TIMERx_CH1 和 TIMERx_CH2 异或之后的信号。通道输入信号 CIx 先被 TIMER_CK 信号同步，然后经过数字滤波器采样，产生一个被滤波后的信号。通过边沿检测器，可以选择检测上升沿或者下降沿。通过配置 CHxP 选择使用上升沿或者下降沿。配置 CHxMS，可以选择其他通道的输入信号，内部触发信号。配置 IC 预分频器，使得若干个输入事件后才产生一个有效的捕获事件。捕获事件发生，CxCV 存储计数器的值。

配置步骤如下：

**第一步：**滤波器配置(TIMERx_CHCTL0寄存器中CHxCAPFLT)：
根据输入信号和请求信号的质量，配置相应的CHxCAPFLT。

**第二步**：边沿选择(TIMERx_CHCTL2寄存器中CHxP/CHxNP)：
配置CHxP/CHxNP选择上升沿或者下降沿。

**第三步：**捕获源选择(TIMERx_CHCTL0寄存器中CHxMS)：
一旦通过配置CHxMS选择输入捕获源，必须确保通道配置在输入模式(CHxMS!=0x0)，而且TIMERx_CHxCV寄存器不能再被写。

**第四步：**中断使能(TIMERx_DMAINTEN寄存器中CHxIE和CHxDEN)：
使能相应中断，可以获得中断和DMA请求。

**第五步：**捕获使能(TIMERx_CHCTL2寄存器中CHxEN)。

**结果：**当期望的输入信号发生时，TIMERx_CHxCV被设置成当前计数器的值，CHxIF位置1。如果CHxIF位已经为1，则CHxOF位置1。根据TIMERx_DMAINTEN寄存器中CHxIE和CHxDEN的配置，相应的中断和DMA请求会被提出。

**直接产生：**软件设置CHxG位，会直接产生中断和DMA请求。

输入捕获模式也可用来测量 TIMERx_CHx 引脚上信号的脉冲波宽度。例如，一个 PWM 波连

接到 CI0。配置 TIMERx_CHCTL0 寄存器中 CH0MS 为 2'b01，选择通道 0 的捕获信号为 CI0 并设置上升沿捕获。配置 TIMERx_CHCTL0 寄存器中 CH1MS 为 2'b10，选择通道 1 捕获信号为 CI0 并设置下降沿捕获。计数器配置为复位模式，在通道 0 的上升沿复位。TIMERX_CH0CV 寄存器测量 PWM 的周期值，TIMERx_CH1CV 寄存器测量 PWM 占空比值。

■ 通道输出比较功能

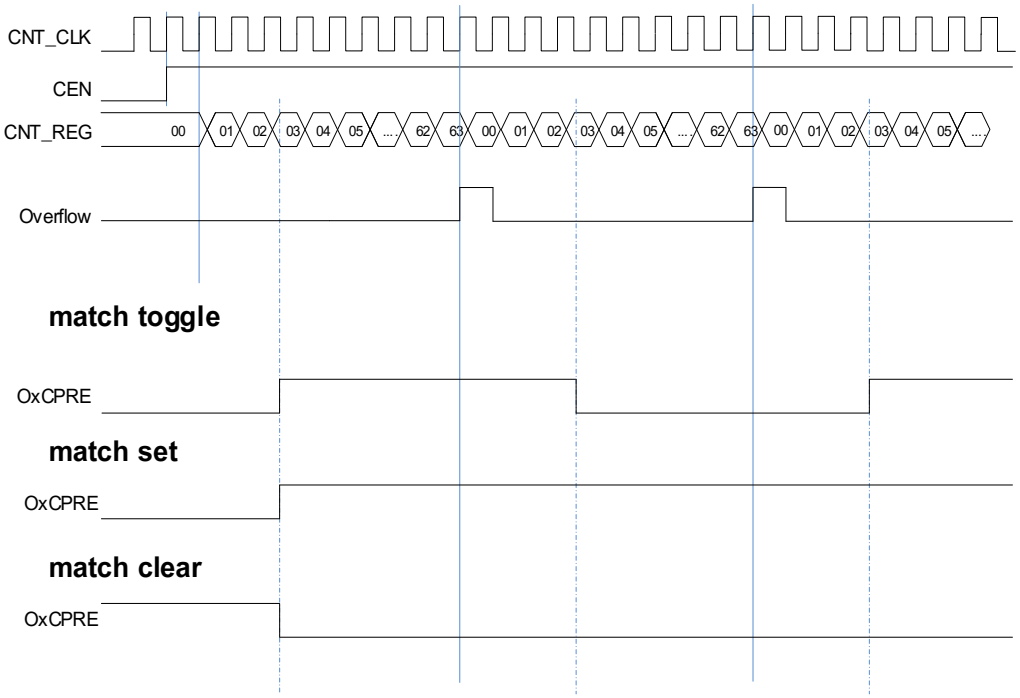
在输出比较模式，TIMERx 可以产生时控脉冲，其位置，极性，持续时间和频率都是可编程的。当一个输出通道的 CHxCV 寄存器与计数器的值匹配时，根据 CHxCOMCTL 的配置，这个通道的输出可以被置高电平，被置低电平或者反转。当计数器的值与 CHxCV 寄存器的值匹配时，CHxIF 位被置 1，如果 CHxIE = 1 则会产生中断，如果 CxCDE=1 则会产生 DMA 请求。

配置步骤如下：

**第一步：** 时钟配置：
配置定时器时钟源，预分频器等。
**第二步：** 比较模式配置：
设置CHxCOMSEN位来配置输出比较影子寄存器；
设置CHxCOMCTL位来配置输出模式(置高电平/置低电平/反转)；
设置CHxP/CHxNP位来选择有效电平的极性；
设置CHxEN使能输出。
**第三步：** 通过CHxIE/CxCDE位配置中断/DMA请求使能。
**第四步：** 通过TIMERx_CAR寄存器和TIMERx_CHxCV寄存器配置输出比较时基：
CHxCVAL可以在运行时根据你所期望的波形而改变。
**第五步：** 设置CEN位使能定时器。

*图 18-13. 三种输出比较模式*显示了三种比较输出模式：反转/置高电平/置低电平，CAR=0x63，CHxVAL=0x3。

**图 18-13. 三种输出比较模式**



### 输出 PWM 功能

在 PWM 输出模式下(PWM 模式 0 是配置 CHxCOMCTL 为 3'b110，PWM 模式 1 是配置 CHxCOMCTL 为 3'b111)，通道根据 TIMERx_CAR 寄存器和 TIMERx_CHxCV 寄存器的值，输出 PWM 波形。

根据计数模式，我们可以分为两种 PWM 波：EAPWM(边沿对齐 PWM)和 CAPWM(中央对齐 PWM)。

EAPWM 的周期由 TIMERx_CAR 寄存器值决定，占空比由 TIMERx_CHxCV 寄存器值决定。*图 18-14. EAPWM 时序图*显示了 EAPWM 的输出波形和中断。

CAPWM 的周期由(2*TIMERx_CAR 寄存器值)决定，占空比由(2*TIMERx_CHxCV 寄存器值)决定。*图 18-15. CAPWM 时序图*显示了 CAPWM 的输出波形和中断。

在 PWM0 模式下(CHxCOMCTL==3'b110)，如果 TIMERx_CHxCV 寄存器的值大于 TIMERx_CAR 寄存器的值，通道输出一直为有效电平。

在 PWM0 模式下(CHxCOMCTL==3'b110)，如果 TIMERx_CHxCV 寄存器的值等于 0，通道输出一直为无效电平。

**图 18-14. EAPWM 时序图**

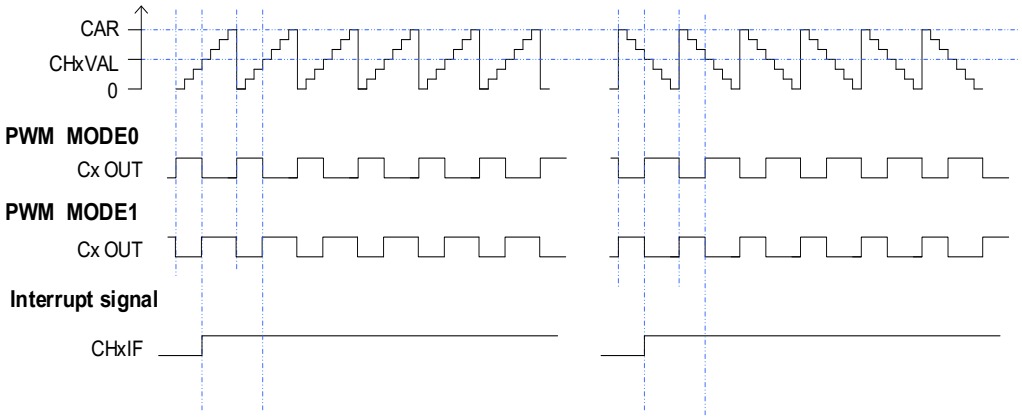
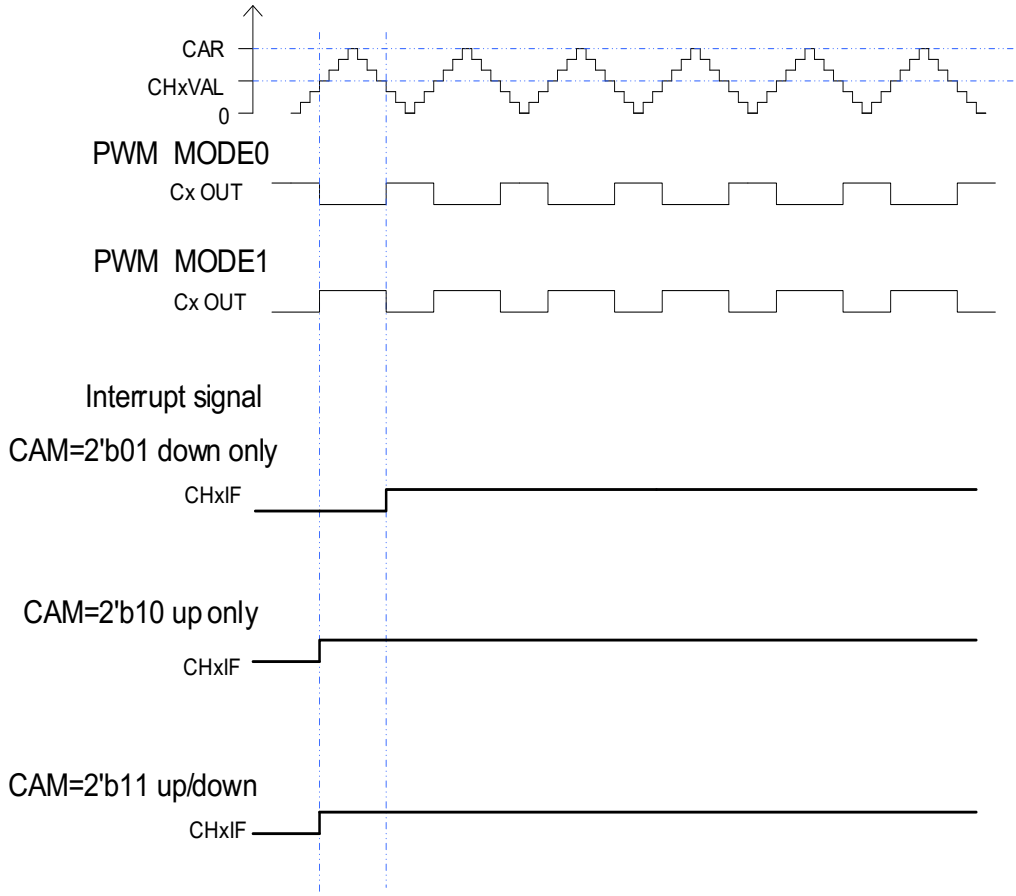


**图 18-15. CAPWM 时序图**




### 通道输出准备信号

当 TIMERx 用于输出匹配比较模式下，设置 CHxCOMCTL 位可以定义 OxCPRE 信号(通道 x 准备信号)类型。OxCPRE 信号有若干类型的输出功能，包括，设置 CHxCOMCTL=0x00 可以保持原始电平；设置 CHxCOMCTL=0x01 可以将 OxCPRE 信号设置为高电平；设置 CHxCOMCTL=0x02 可以将 OxCPRE 信号设置为低电平；设置 CHxCOMCTL=0x03，在计数器值和 TIMERx_CHxCV 寄存器的值匹配时，可以翻转输出信号。

PWM 模式 0 和 PWM 模式 1 是 OxCPRE 的另一种输出类型，设置 CHxCOMCTL 位域位 0x06 或 0x07 可以配置 PWM 模式 0/PWM 模式 1。在这些模式中，根据计数器值和 TIMERx_CHxCV 寄存器值的关系以及计数方向，OxCPRE 信号改变其电平。具体细节描述，请参考相应的位。

设置 CHxCOMCTL =0x04 或 0x05 可以实现 OxCPRE 信号的强制输出功能。输出比较信号能够直接由软件强置为有效或无效状态，而不依赖于 TIMERx_CHxCV 的值和计数器值之间间的比较结果。

设置 CHxCOMCEN=1，当由外部 ETI 引脚信号产生的 ETIFE 信号为高电平时，OxCPRE 被强制为低电平。在下一次更新事件到来时，OxCPRE 信号才会回到有效电平状态。

### 通道输出互补 PWM

CHx_O 和 CHx_ON 是一对互补输出通道，这两个信号不能同时有效。TIMERx 有四路通道，只有前三路有互补输出通道。互补信号 CHx_O 和 CHx_ON 是由一组参数来决定：TIMERx_CHCTL2 寄存器中的 CHxEN 和 CHxNEN 位，TIMERx_CCHP 寄存器中和 TIMERx_CTL1 寄存器中的 POEN，ROS，IOS，ISOx 和 ISOxN 位。输出极性由 TIMERx_CHCTL2 寄存器中的 CHxP 和 CHxNP 位来决定。

**表 18-2. 由参数控制的互补输出表**

<table>
<tr><th colspan="5">互补参数</th><th colspan="2">输出状态</th></tr>
<tr><th>POEN</th><th>ROS</th><th>IOS</th><th>CHxEN</th><th>CHxNEN</th><th>CHx_O</th><th>CHx_ON</th></tr>
<tr><td rowspan="4">0</td><td rowspan="4">0/1</td><td rowspan="3">0</td><td rowspan="2">0</td><td>0</td><td colspan="2">CHx_O / CHx_ON = LOW<br>CHx_O / CHx_ON 输出禁能(1)</td></tr>
<tr><td>1</td><td colspan="2" rowspan="3">CHx_O/CHx_ON输出关闭状态(2):<br>通道先输出无效电平：CHx_O = CHxP，CHx_ON = CHxNP)；如果死区产生时钟未失效，在死区时间之后：<br>CHx_O = ISOx，CHx_ON = ISOxN (3)</td></tr>
<tr><td rowspan="2">1</td><td>0</td></tr>
<tr><td>1</td><td>1</td></tr>
<tr><td>1</td><td>x</td><td>x</td><td colspan="2">CHx_O/CHx_ON输出关闭状态:<br>通道先输出无效电平：CHx_O = CHxP，CHx_ON = CHxNP)；如果死区产生时钟未失效，在死区时间之后：<br>CHx_O = ISOx，CHx_ON = ISOxN</td></tr>
<tr><td rowspan="6">1</td><td rowspan="4">0</td><td rowspan="6">0/1</td><td rowspan="2">0</td><td>0</td><td colspan="2">CHx_O/CHx_ON = LOW<br>CHx_O/CHx_ON输出禁能</td></tr>
<tr><td>1</td><td>CHx_O = LOW<br>CHx_O输出禁能</td><td>CHx_ON=OxCPRE⊕(4)CHxNP<br>CHx_ON输出使能</td></tr>
<tr><td rowspan="2">1</td><td>0</td><td>CHx_O=OxCPRE⊕CHxP<br>CHx_O输出使能</td><td>CHx_ON = LOW<br>CHx_ON输出禁能</td></tr>
<tr><td>1</td><td>CHx_O=OxCPRE⊕CHxP<br>CHx_O输出使能</td><td>CHx_ON=(!OxCPRE)(5)⊕CHxNP<br>CHx_ON输出使能</td></tr>
<tr><td rowspan="2">1</td><td rowspan="2">0</td><td>0</td><td>CHx_O = CHxP<br>CHx_O输出关闭状态</td><td>CHx_ON = CHxNP<br>CHx_ON输出关闭状态</td></tr>
<tr><td>1</td><td>CHx_O = CHxP<br>CHx_O输出关闭状态</td><td>CHx_O=OxCPRE⊕CHxNP<br>CHx_ON输出使能</td></tr>
</table>

| 互补参数 | | | | | 输出状态 | |
|---|---|---|---|---|---|---|
| POEN | ROS | IOS | CHxEN | CHxNEN | CHx_O | CHx_ON |
| | | | 1 | 0 | CHx_O=OxCPRE⊕CHxP<br>CHx_O输出使能 | CHx_ON = CHxNP<br>CHx_ON输出关闭状态 |
| | | | | 1 | CHx_O=OxCPRE⊕CHxP<br>CHx_O输出使能 | CHx_ON=(!OxCPRE)⊕CHx<br>NP<br>CHx_ON输出使能 |

**注意**:

（1） 输出禁能：CHx_O / CHx_ON 输出与对应引脚断开，对应引脚电平受 GPIO 上下拉配置控制，无上下拉时为悬空高阻态；

（2） 输出关闭状态：CHx_O / CHx_ON 输出无效电平(CHx_O = 0⊕CHxP = CHxP)；

（3） 详情见中止模式章节。

（4） ⊕：异或操作；

（5） (!OxCPRE)：OxCPRE 信号的互补信号。

### 互补 PWM 插入死区时间

设置 CHxEN 和 CHxNEN 为 1'b1 同时设置 POEN，死区插入就会被使能。DTCFG 位域定义了死区时间，死区时间对除了通道 3 以外通道有效。死区时间的细节，请参考 TIMERx_CCHP 寄存器。

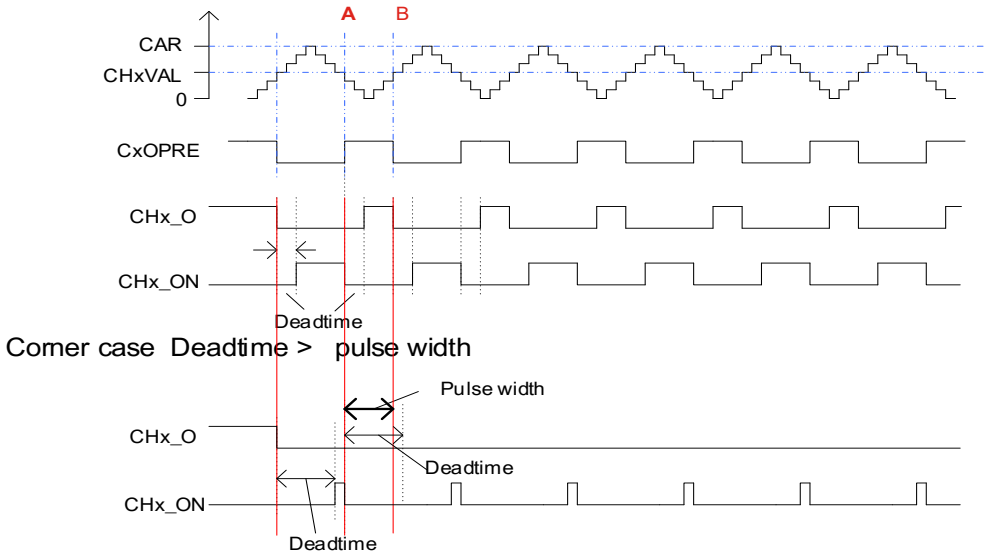
死区时间的插入，确保了通道互补的两路信号不会同时有效。

在 PWM0 模式，当通道 x 匹配发生时(TIMERx 计数器= CxCV)，OxCPRE 反转。在*图 18-16. 带死区时间的互补输出*中的 A 点，CHx_O 信号在死区时间内为低电平，直到死区时间过后才变为高电平，而 CHx_ON 信号立刻变为低电平。同样，在 B 点，计数器再次匹配(TIMERx 计数器= CxCV)，OxCPRE 信号被清 0，CHx_O 信号被立即清零，CHx_ON 信号在死区时间内仍然是低电平，在死区时间过后才变为高电平。

有时会有一些死角事件发生，例如：

- 如果死区延时大于或者等于CHx_O信号的占空比，CHx_O信号一直为无效值(如*图18-16. 带死区时间的互补输出*)。
- 如果死区延时大于或者等于CHx_ON信号的占空比，CHx_ON信号一直为无效值。

**图 18-16. 带死区时间的互补输出**



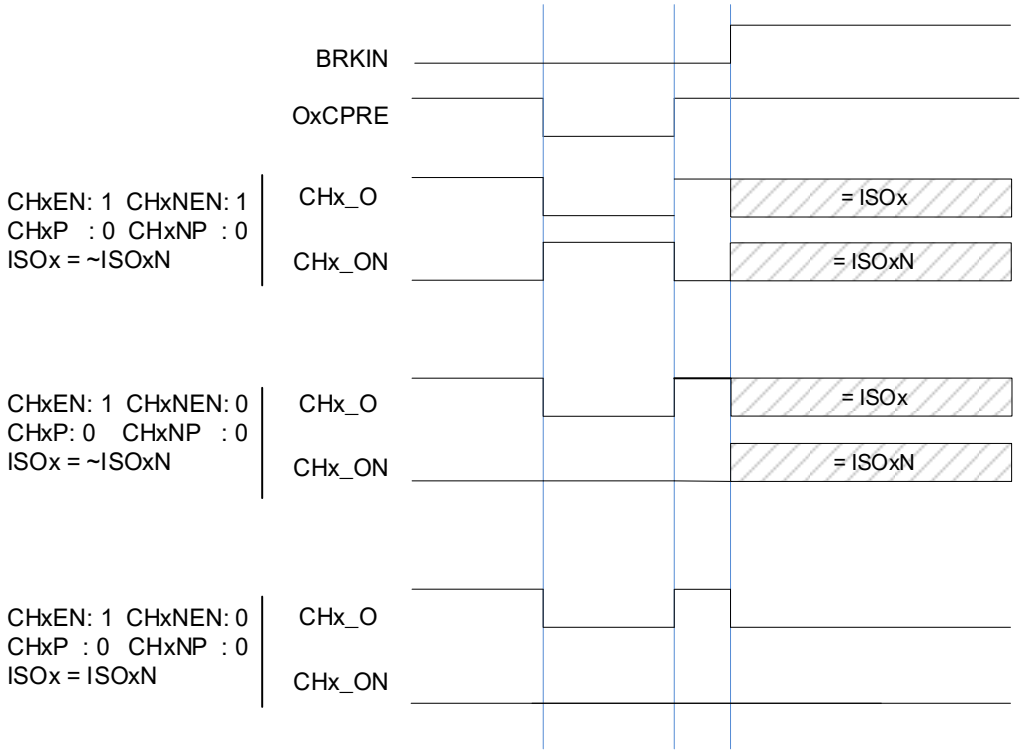

## 中止模式

使用中止模式时，输出 CHx_O 和 CHx_ON 信号电平被以下位控制，TIMERx_CCHP 寄存器的 POEN, IOS 和 ROS 位，TIMERx_CTL1 寄存器的 ISOx 和 ISOxN 位。当中止事件发生时，CHx_O 和 CHx_ON 信号输出不能同时设置为有效电平。中止源可以选择中止输入引脚，也可以选择 HXTAL 时钟失效事件。时钟失败事件由 RCU 中的时钟监视器(CKM)产生。将 TIMERx_CCHP 寄存器的 BRKEN 位置 1 可以使能中止功能。TIMERx_CCHP 寄存器的 BRKP 位决定了中止输入极性。

发生中止时，POEN 位被异步清除，一旦 POEN 位为 0，CHx_O 和 CHx_ON 被 TIMERx_CTL1 寄存器中的 ISOx 位和 ISOxN 驱动。如果 IOS=0，定时器释放输出使能，否则输出使能仍然为高。起初互补输出被置于复位状态，然后死区时间产生器重新被激活，以便在一个死区时间后驱动输出，输出电平由 ISOx 和 ISOxN 位配置。

发生中止时，TIMERx_INTF 寄存器的 BRKIF 位被置 1。如果 BRKIE=1，中断产生。

图 18-17. 通道响应中止输入(高电平有效)时，输出信号的行为




## 正交译码器

正交译码器功能使用由TIMERx_CH0和TIMERx_CH1引脚生成的CI0FE0和CI1FE1正交信号各自相互作用产生计数值。在每个输入源改变期间，DIR位会发生改变。输入源可以是只有CI0FE0，可以只有CI1FE1，或着可以同时有CI0FE0和 CI1FE1，通过设置SMC=0x01, 0x02或0x03来选择使用哪种模式。计数器计数方向改变的机制如*表18-3. 不同译码器模式下的计数方向*所示。正交译码器可以当作一个带有方向选择的外部时钟，这意味着计数器会在0和自动加载值之间连续的计数。因此，用户必须在计数器开始计数前配置TIMERx_CAR寄存器。

表 18-3. 不同译码器模式下的计数方向

| 计数模式 | 电平 | CI0FE0 | | CI1FE1 | |
|---|---|---|---|---|---|
| | | 上升 | 下降 | 上升 | 下降 |
| 译码器模式0 SMC[2:0]=3'b001 | CI1FE1=1 | 向下 | 向上 | - | - |
| | CI1FE1=0 | 向上 | 向下 | - | - |
| 译码器模式1 SMC [2:0]=3'b010 | CI0FE0=1 | - | - | 向上 | 向下 |
| | CI0FE0=0 | - | - | 向下 | 向上 |
| 译码器模式2 SMC [2:0]=3'b011 | CI1FE1=1 | 向下 | 向上 | X | X |
| | CI1FE1=0 | 向上 | 向下 | X | X |
| | CI0FE0=1 | X | X | 向上 | 向下 |
| | CI0FE0=0 | X | X | 向下 | 向上 |

**注意：** "-" 意思是"无计数"; "X" 意思是不可能。"0" 意思是低电平, "1" 意思是高电平

图 18-18. 在译码器模式 2 且 CI0FE0 极性不反相时计数器行为

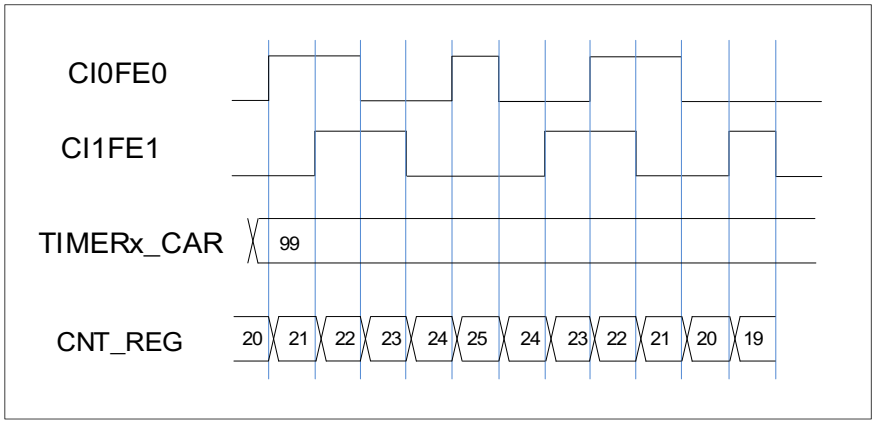
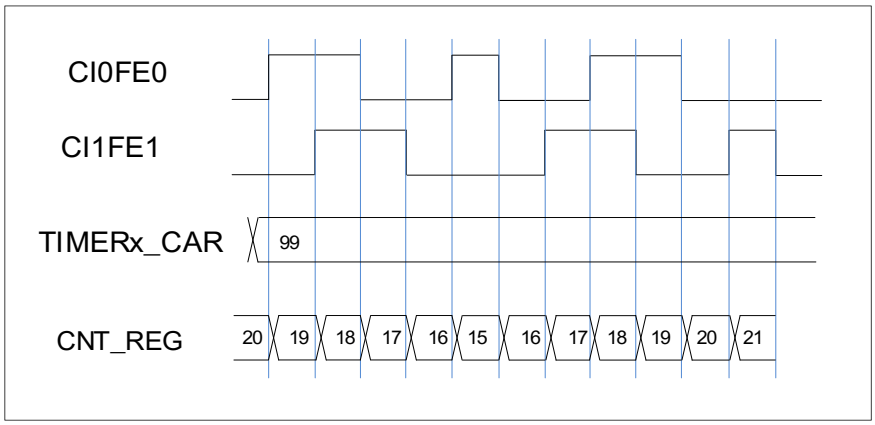



图 18-19. 在译码器模式 2 且 CI0FE0 极性反相时计数器行为




## 霍尔传感器接口功能

高级定时器支持霍尔传感器接口功能，该功能可以用来控制 BLDC 电机。

*图18-20. 霍尔传感器用在BLDC电机控制中*是定时器和电机的连接示意图。TIMER_in定时器(可以是高级定时器或者通用L0定时器)接收霍尔传感器的三路信号。

三个霍尔传感器信号与 TIMER_in 定时器的三路输入捕获引脚一一对应连接，每个霍尔传感器输入一路波形到输入引脚，分析三路霍尔信号可以计算出转子的位置和速度。

通过定时器内部连接，例如 TRGO-ITIx，TIMER_in 定时器和 TIMER_out 定时器可以连接在一起。TIMER_out 定时器根据 ITIx 触发信号输出 PWM 波，驱动 BLDC 电机，控制 BLDC 电机的速度。这样，TIMER_in 定时器和 TIMER_out 定时器的连接形成了一个反馈电路，可以根据需求改变配置。

TIMER_in 定时器需要具备输入异或功能，所以可以选择高级定时器和通用 L0 定时器。

TIMER_out 定时器需要具备互补输出和死区插入功能，所以可以选择高级定时器。另外，根据定时器的内部互连关系，可以选择成对的互连定时器，例如:

TIMER_in (TIMER0) -> TIMER_out (TIMER7 ITI0)

TIMER_in (TIMER1) -> TIMER_out (TIMER0 ITI1)

等等。

选择好合适的互连定时器，定时器和 BLDC 的线路也已经连接好，我们就可以配置定时器了。有以下关键配置：

- 设置TI0S，使能异或功能。三路输入信号的任何一路发生变化，CI0都会反转，CH0VAL此时会捕获计数器的当前值。
- 设置CCUC和CCSE，使能ITIx直接连接到换相功能。
- 根据需求配置PWM参数。

**图 18-20. 霍尔传感器用在 BLDC 电机控制中**

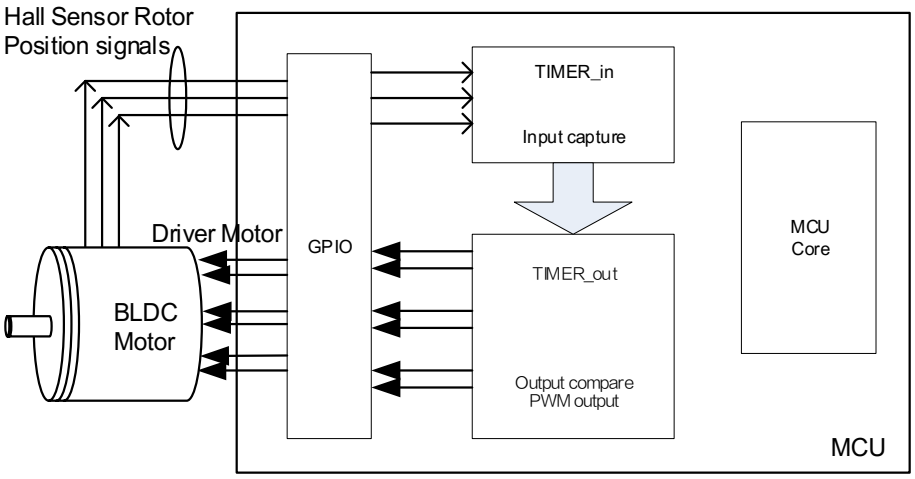
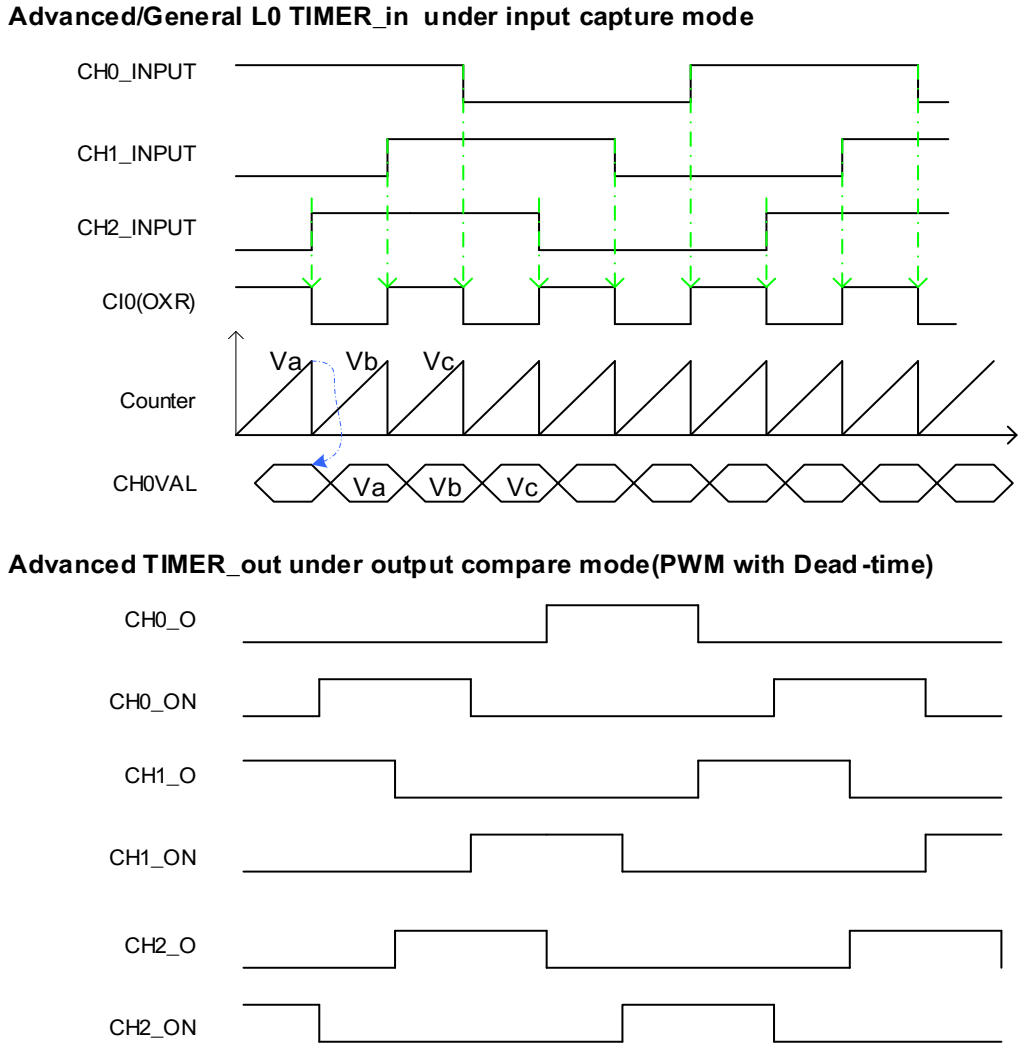


图 18-21. 两个定时器之间的霍尔传感器时序图

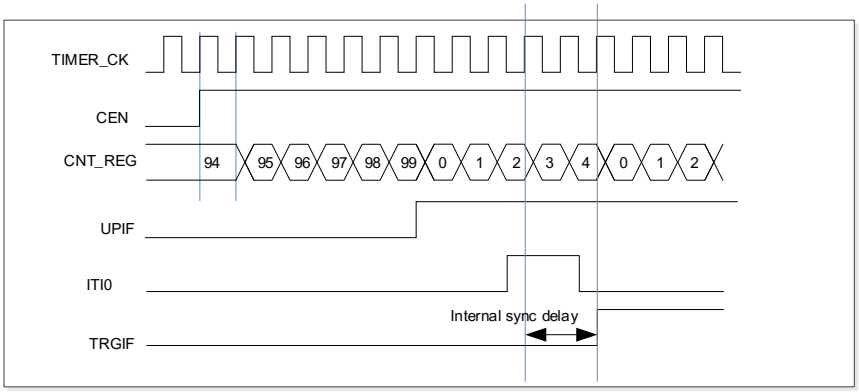
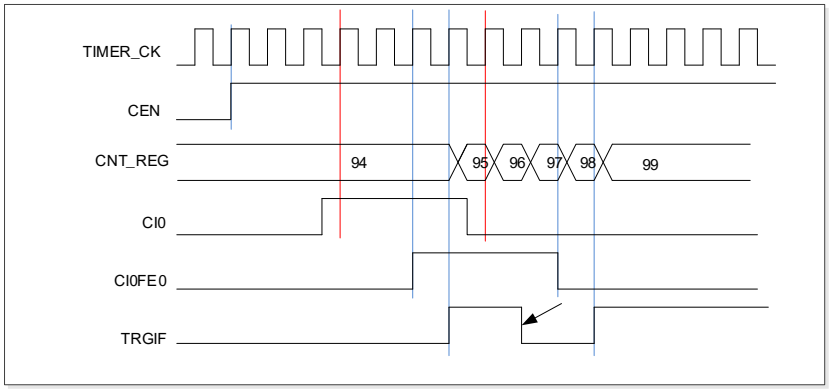



## 主-从管理

TIMERx 能在多种模式下同步外部触发，包括复位模式，暂停模式和事件模式，可以通过设置 TIMERx_SMCFG 寄存器中的 SMC [2:0]配置这些模式。这些模式的输入触发源可以通过设置 TIMERx_SMCFG 寄存器中的 TRGS [2:0]来选择。

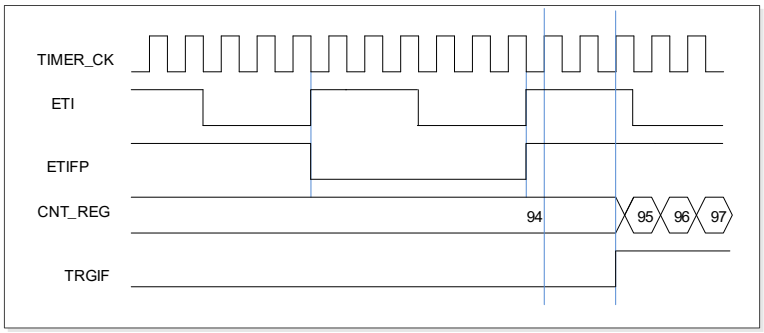
表 18-4. 从模式例子列表和举例

| | 模式选择 | 触发源选择 | 极性选择 | 滤波和预分频 |
|---|---|---|---|---|
| 列举 | SMC[2:0]<br>3'b100(复位模式)<br>3'b101(暂停模式)<br>3'b110(事件模式) | TRGS[2:0]<br>000：ITI0<br>001：ITI1<br>010：ITI2<br>011：ITI3<br>100：CI0F_ED<br>101：CI0FE0 | 如果触发源是CI0FE0或者CI1FE1，配置CHxP和CHxNP来选择极性和反相。<br>如果触发源是ETIFP，配置ETP选择极性和反相。 | 若触发源为ITIx，滤波和预分频不可用。<br>若触发源为CIx，可配置CHxCAPFLT设置滤波，预分频不可用。<br>若触发源为ETIFP，滤波和预分频均可用。 |

<table>
<tr><th></th><th>模式选择</th><th>触发源选择</th><th>极性选择</th><th>滤波和预分频</th></tr>
<tr><td></td><td></td><td>110：CI1FE1<br>111：ETIFP</td><td></td><td></td></tr>
<tr><td rowspan="2">例1</td><td>复位模式<br>当触发输入上升沿到来时，计数器清零重启。</td><td>TRGS[2:0]=3'b000<br>选择ITI0为触发源。</td><td>若触发源是ITI0，极性选择不可用。</td><td>若触发源是ITI0，滤波和预分频不可用。</td></tr>
<tr><td colspan="4">图 18-22. 复位模式<br>
</td></tr>
<tr><td rowspan="2">例2</td><td>暂停模式<br>当触发输入为低的时候，计数器暂停计数，当触发输入为高时，计数器计数。</td><td>TRGS[2:0]=3'b101<br>选择CI0FE0为触发源。</td><td>TI0S=0(非异或)[CH0NP=0，CH0P=0] CI0FE0不反相。捕获发生在上升沿 。</td><td>在这个例子中滤波被旁路。</td></tr>
<tr><td colspan="4">图 18-23. 暂停模式<br>
</td></tr>
<tr><td>例3</td><td>事件模式<br>触发输入的上升沿计数器开始计数。</td><td>TRGS[2:0]=3'b111<br>选择ETIFP为触发源。</td><td>ETP = 0，ETI极性不改变。</td><td>ETPSC = 1，ETI 2分频。<br>ETFC = 0，ETI 无滤波。</td></tr>
</table>

| | 模式选择 | 触发源选择 | 极性选择 | 滤波和预分频 |
|---|---|---|---|---|

图 18-24. 事件模式



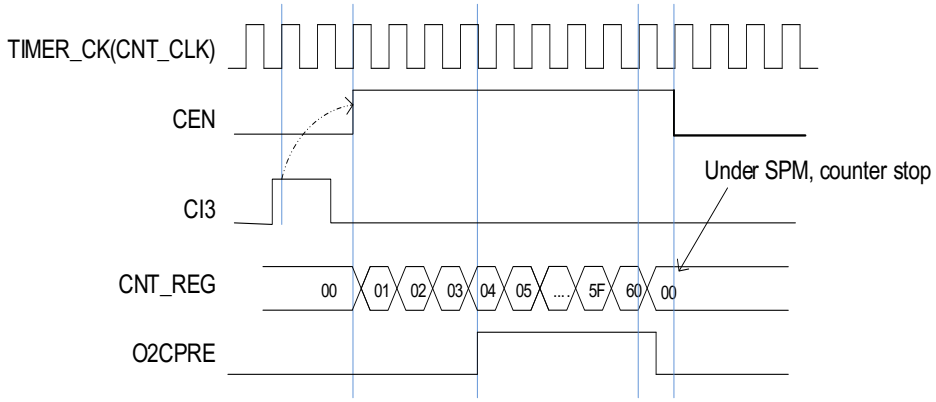

## 单脉冲模式

单脉冲模式与重复模式是相反的，设置 TIMERx_CTL0 寄存器的 SPM 位置 1，则使能单脉冲模式。当 SPM 置 1，计数器在下次更新事件到来后清零并停止计数。为了得到脉冲波，可以通过设置 CHxCOMCTL 配置 TIMERx 为 PWM 模式或者比较模式。

一旦设置定时器运行在单脉冲模式下，没有必要设置 TIMERx_CTL0 寄存器的定时器使能位 CEN=1 来使能计数器。触发信号沿或者软件写 CEN=1 都可以产生一个脉冲，此后 CEN 位一直保持为 1 直到更新事件发生或者 CEN 位被软件写 0。如果 CEN 位被软件清 0，计数器停止工作， 计数值被保持。

在单脉冲模式下，有效的外部触发边沿会将 CEN 位置 1，使能计数器。然而，执行计数值和 TIMERx_CHxCV 寄存器值的比较结果依然存在一些时钟延迟。为了最大限度减少延迟，用户可以将 TIMERx_CHCTL0/1 寄存器的 CHxCOMFEN 位置 1。单脉冲模式下，触发上升沿产生之后， OxCPRE 信号将被立即强制转换为与发生比较匹配时相同的电平，但是不用考虑比较结果。只有输出通道配置为 PWM0 或 PWM1 输出运行模式下时 CHxCOMFEN 位才可用，触发源来源于触发信号

**图 18-25. 单脉冲模式，TIMERx_CHxCV = 0x04 TIMERx_CAR=0x60**

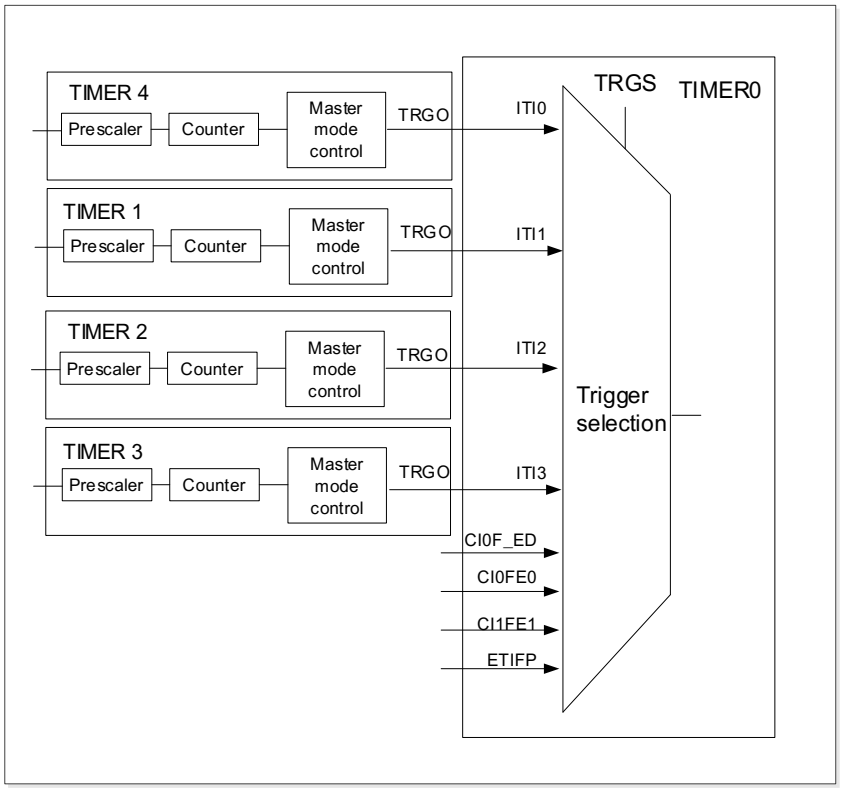


### 定时器互连

定时器之间可配置为内部级联，一个定时器配置为主模式输出TRGO信号，另一个定时器配置为从模式，TRGO信号包括复位事件、使能事件、更新事件、捕获比较脉冲事件、比较事件。从定时器接收到ITIx信号，并执行对应的操作，包括内部时钟模式、正交编码模式、复位模式、暂停模式、事件模式、外部时钟模式。

*图 18-26. 定时器 0 主/从模式的例子* 显示了当定时器 0 配置为从模式时的触发选择。

**图 18-26. 定时器 0 主/从模式的例子**




其他定时器互连的例子：

■ 定时器2作为定时器0的预分频器

参考*图 18-26. 定时器 0 主/从模式的例子*连接配置定时器 2 为定时器 0 的预分频器，步骤如下：

1. 配置定时器 2 为主模式，选择其更新事件(UPE)为触发输出(配置 TIMER2_CTL1 寄存器的 MMC=3'b010)。定时器 2 在每次计数器溢出时，输出一个周期信号；
2. 配置定时器 2 周期(TIMER2_CAR 寄存器)；
3. 选择定时器 0 输入触发源为定时器 2(配置 TIMERx_SMCFG 寄存器的 TRGS=3'b001)
4. 配置定时器 0 在外部时钟模式 0 (配置 TIMERx_SMCFG 寄存器的 SMC=3'b111)；
5. 写 1 到 CEN 位启动定时器 0 (TIMER0_CTL0 寄存器)；
6. 写 1 到 CEN 位启动定时器 2 (TIMER2_CTL0 寄存器)。

■ 使用一个外部触发来同步两个定时器

配置定时器 2 的使能信号触发定时器 0 的开启，配置定时器 2 的 CI0 输入信号上升沿来触发定时器 2。为了确保两个定时器同步开启，定时器 2 必须配置在主/从模式。步骤如下：

1. 配置定时器 2 工作在从模式来获取来自 CI0 的触发输入(配置 TIMER2_SMCFG 寄存器的 TRGS=3'b100)；

2. 配置定时器 2 工作在事件模式(配置 TIMER2_SMCFG 寄存器的 SMC=3'b110)；

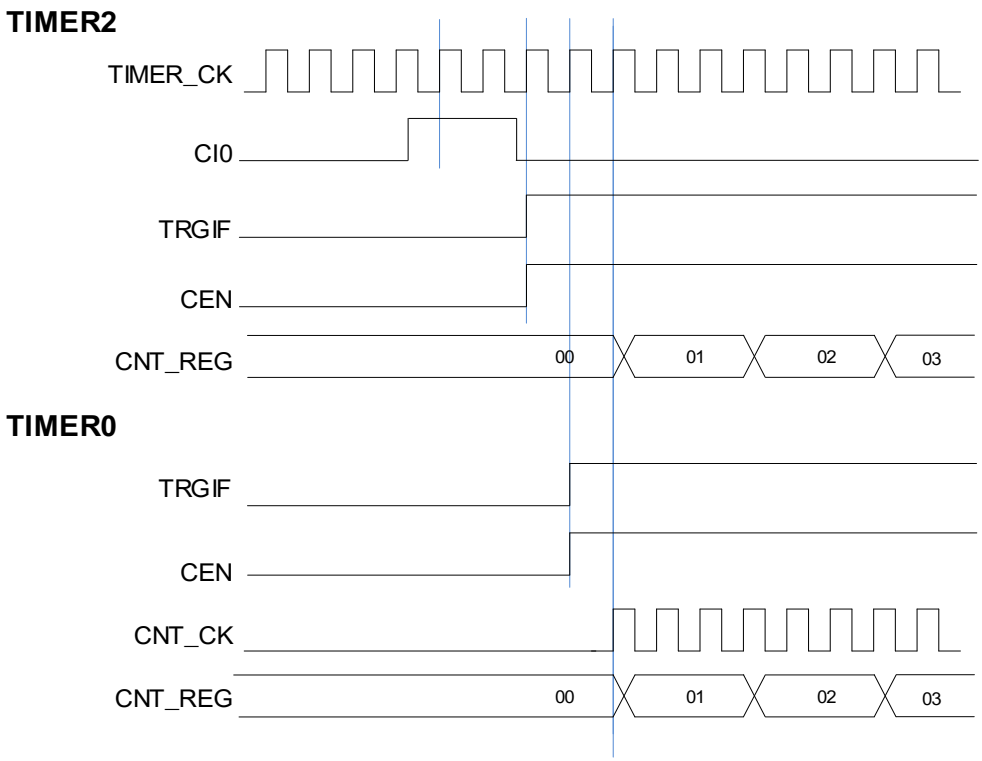
3. 写 MSM=1(TIMER2_SMCFG 寄存器)来配置定时器 2 工作在主/从模式；

4. 配置定时器 0 的触发输入来自定时器 2 (配置 TIMERx_SMCFG 寄存器的 TRGS=3'b010)；

5. 配置定时器 0 工作在事件模式(配置 TIMER0_SMCFG 寄存器的 SMC=3'b110)。

当定时器 2 的 CI0 信号产生上升沿时，两个定时器的计数器在内部时钟下开始同步计数，二者的 TRGIF 标志位都被置 1。

**图 18-27. 用定时器 2 的 CI0 输入来触发定时器 0 和定时器 2**




## 定时器 DMA 模式

定时器 DMA 模式是指通过 DMA 模块配置定时器的寄存器。有两个跟定时器 DMA 模式相关的寄存器：TIMERx_DMACFG 和 TIMERx_DMATB。当然，必须要使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。当中断事件发生，TIMERx 会给 DMA 发送请求。DMA 配置成 M2P 模式，PADDR 是 TIMERx_DMATB 寄存器地址，DMA 就会访问 TIMERx_DMATB 寄存器。实际上，TIMERx_DMATB 寄存器只是一个缓冲，定时器会将 TIMERx_DMATB 映射到一个内部寄存器，这个内部寄存器由 TIMERx_DMACFG 寄存器中的 DMATA 来指定。如果

TIMERx_DMACFG 寄存器的 DMATC 位域值为 0，表示 1 次传输，定时器的发送 1 个 DMA 请求就可以完成。如果 TIMERx_DMACFG 寄存器的 DMATC 位域值不为 1，例如其值为 3，表示 4 次传输，定时器就需要再多发 3 次 DMA 请求。在这 3 次请求下，DMA 对 TIMERx_DMATB 寄存器的访问会映射到访问定时器的 DMATA+0x4, DMATA+0x8, DMATA+0xc 寄存器。总之，发生一次 DMA 内部中断请求，定时器会连续发送(DMATC+1)次请求。

如果再来 1 次 DMA 请求事件，TIMERx 将会重复上面的过程。

### 定时器调试模式

当 Cortex®-M4 内核停止，DBG_CTL2 寄存器中的 TIMERx_HOLD 配置位被置 1，定时器计数器停止。

## 18.1.4. TIMERx 寄存器(x=0,7)

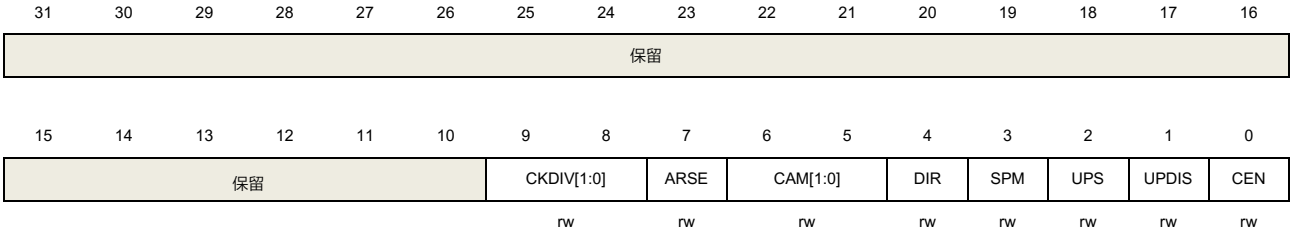
TIMER0 基地址：0x4001 0000

TIMER7 基地址：0x4001 0400

### 控制寄存器 0(TIMERx_CTL0)

地址偏移：0x00
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:10 | 保留 | 必须保持复位值。 |
| 9:8 | CKDIV[1:0] | 时钟分频<br>通过软件配置CKDIV，规定定时器时钟(CK_TIMER) 与死区时间和数字滤波器采样时钟(DTS)之间的分频系数。<br>00：$f_{DTS}=f_{CK_TIMER}$<br>01：$f_{DTS}= f_{CK_TIMER}$ /2<br>10：$f_{DTS}= f_{CK_TIMER}$ /4<br>11：保留 |
| 7 | ARSE | 自动重载影子使能<br>0：禁能 TIMERx_CAR 寄存器的影子寄存器。<br>1：使能 TIMERx_CAR 寄存器的影子寄存器。 |
| 6:5 | CAM[1:0] | 计数器对齐模式选择<br>00：无中央对齐计数模式(边沿对齐模式)。 DIR位指定了计数方向<br>01：中央对齐向下计数置1模式。计数器在中央计数模式计数，通道被配置在输出模式(TIMERx_CHCTL0寄存器中CHxMS=00)，只有在向下计数时，CHxF位置1<br>10：中央对齐向上计数置1模式。计数器在中央计数模式计数，通道被配置在输出模式(TIMERx_CHCTL0寄存器中CHxMS=00)，只有在向上计数时，CHxF位置1<br>11：中央对齐上下计数置1模式。计数器在中央计数模式计数，通道被配置在输出模式(TIMERx_CHCTL0寄存器中CHxMS=00)，在向上和向下计数时，CHxF位都会置1<br>当计数器使能以后，该位不能从 0x00 切换到非 0x00 |
| 4 | DIR | 方向<br>0：向上计数 |

1：向下计数
当计数器配置为中央对齐计数模式或译码器模式时，该位只读。

| | | |
|---|---|---|
| 3 | SPM | 单脉冲模式<br>0：单脉冲模式禁能。更新事件发生后，计数器继续计数<br>1：单脉冲模式使能。在下一次更新事件发生时，计数器停止计数 |
| 2 | UPS | 更新请求源<br>软件配置该位，选择更新事件源.<br>0：以下事件均会产生更新中断或DMA请求：<br>UPG位被置1<br>计数器溢出/下溢<br>复位模式产生的更新<br>1：下列事件会产生更新中断或DMA请求：<br>计数器溢出/下溢 |
| 1 | UPDIS | 禁止更新.<br>该位用来使能或禁能更新事件的产生<br>0：更新事件使能. 更新事件发生时，相应的影子寄存器被装入预装载值，以下事件均会产生更新事件：<br>UPG位被置1<br>计数器溢出/下溢<br>复位模式产生的更新<br>1：更新事件禁能.<br>注意：当该位被置 1 时，UPG 位被置 1 或者复位模式不会产生更新事件，但是计数器和预分频器被重新初始化 |
| 0 | CEN | 计数器使能<br>0：计数器禁能。<br>1：计数器使能。<br>在软件将 CEN 位置 1 后，外部时钟、暂停模式和译码器模式才能工作。 |

### 控制寄存器 1(TIMERx_CTL1)

地址偏移：0x04
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | ISO3 | ISO2N | ISO2 | ISO1N | ISO1 | ISO0N | ISO0 | TI0S | MMC[2:0] | | | DMAS | CCUC | 保留 | CCSE |
| | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | rw | rw | | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|

| | | |
|---|---|---|
| 31:15 | 保留 | 必须保持复位值。 |
| 14 | ISO3 | 通道 3 的空闲状态输出<br>参考 ISO0 位。 |
| 13 | ISO2N | 通道 2 的互补通道空闲状态输出<br>参考 ISO0N 位。 |
| 12 | ISO2 | 通道 2 的空闲状态输出<br>参考 ISO0 位。 |
| 11 | ISO1N | 通道 1 的互补通道空闲状态输出<br>参考 ISO0N 位。 |
| 10 | ISO1 | 通道 1 的空闲状态输出<br>参考 ISO0 位。 |
| 9 | ISO0N | 通道 0 的互补通道空闲状态输出<br>0：当 POEN 复位，CH0_ON 设置低电平。<br>1：当 POEN 复位，CH0_ON 设置高电平。<br>此位只有在 TIMERx_CCHP 寄存器的 PROT [1:0]位为 00 的时候可以被更改。 |
| 8 | ISO0 | 通道 0 的空闲状态输出<br>0：当 POEN 复位，CH0_O 设置低电平。<br>1：当 POEN 复位，CH0_O 设置高电平。<br>如果 CH0_ON 生效，一个死区时间后 CH0_O 输出改变。此位只有在 TIMERx_CCHP 寄存器的 PROT [1:0]位为 00 的时候可以被更改。 |
| 7 | TI0S | 通道 0 触发输入选择<br>0：选择 TIMERx_CH0 引脚作为通道 0 的触发输入。<br>1：选择 TIMERx_CH0, CH1 和 CH2 引脚异或的结果作为通道 0 的触发输入。 |
| 6:4 | MMC[2:0] | 主模式控制<br>这些位控制 TRGO 信号的选择，TRGO 信号由主定时器发给从定时器用于同步功能<br>000：当产生一个定时器复位事件后，输出一个TRGO信号，定时器复位源为：<br>主定时器产生一个复位事件<br>TIMERx_SWEVG寄存器中UPG位置1<br>001：当产生一个定时器使能事件后，输出一个TRGO信号，定时器使能源为：<br>CEN位置1<br>在暂停模式下，触发输入置1<br>010：当产生一个定时器更新事件后，输出一个TRGO信号，更新事件源由UPDIS和UPS位决定<br>011：当通道0在发生一次捕获或一次比较成功时，主模式控制器产生一个TRGO脉冲<br>100：当产生一次比较事件时，输出一个TRGO信号，比较事件源来自O0CPRE<br>101：当产生一次比较事件时，输出一个TRGO信号，比较事件源来自O1CPRE<br>110：当产生一次比较事件时，输出一个TRGO信号，比较事件源来自O2CPRE<br>111：当产生一次比较事件时，输出一个 TRGO 信号，比较事件源来自 O3CPRE |
| 3 | DMAS | DMA请求源选择 |

0：当通道捕获/比较事件发生时，发送通道 x 的 DMA 请求。
1：当更新事件发生，发送通道 x 的 DMA 请求。

| | | |
|---|---|---|
| 2 | CCUC | 换相控制影子寄存器更新控制<br>当换相控制影子寄存器(CHxEN, CHxNEN 和 CHxCOMCTL 位)使能(CCSE=1)，这些影子寄存器更新控制如下：<br>0：CMTG 位被置 1 时更新影子寄存器。<br>1：当 CMTG 位被置 1 或检测到 TRIGI 上升沿时，影子寄存器更新。<br>当通道没有互补输出时，此位无效。 |
| 1 | 保留 | 必须保持复位值。 |
| 0 | CCSE | 换相控制影子使能<br>0：影子寄存器 CHxEN, CHxNEN 和 CHxCOMCTL 位禁能。<br>1：影子寄存器 CHxEN, CHxNEN 和 CHxCOMCTL 位使能。<br>如果这些位已经被写入了，换相事件到来时这些位才被更新。<br>当通道没有互补输出时，此位无效。 |

### 从模式配置寄存器(TIMERx_SMCFG)

地址偏移：0x08
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ETP | SMC1 | ETPSC[1:0] | | ETFC[3:0] | | | | MSM | TRGS[2:0] | | | 保留 | SMC[2:0] | | |
| rw | rw | rw | | rw | | | | rw | rw | | | rw | rw | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15 | ETP | 外部触发极性<br>该位指定 ETI 信号的极性<br>0：ETI 高电平或上升沿有效 .<br>1：ETI 低电平或下降沿有效 . |
| 14 | SMC1 | SMC 的一部分为了使能外部时钟模式 1<br>在外部时钟模式 1，计数器由 ETIF 信号上的任意有效边沿驱动<br>0：外部时钟模式 1 禁能<br>1：外部时钟模式 1 使能<br>当从模式配置为复位模式，暂停模式和事件模式时，定时器仍然可以工作在外部时钟模式 1。但是 TRGS 必须不能为 3'b111。<br>如果外部时钟模式 0 和外部时钟模式 1 同时被被配置，外部时钟的输入是 ETIF |

注意：外部时钟模式 0 使能在寄存器的 SMC[2:0]位域。

13:12 ETPSC[1:0] 外部触发预分频

外部触发信号 ETIFP 的频率不能超过 TIMER_CK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETIFP 的频率。

00：预分频禁能

01：2 分频

10：4 分频

11：8 分频

11:8 ETFC[3:0] 外部触发滤波控制

外部触发信号可以通过数字滤波器进行滤波，该位域定义了数字滤波器的滤波能力。数字滤波器的基本原理是：以 $f_{SAMP}$ 频率连续采样外部触发信号，同时记录采样相同电平的次数。当该次数达到配置的滤波能力时，则认为是一个有效的电平信号。

| EXTFC[3:0] | 次数 | $f_{SAMP}$ |
|---|---|---|
| 4'b0000 | Filter disabled. | |
| 4'b0001 | 2 | $f_{CK_TIMER}$ |
| 4'b0010 | 4 | |
| 4'b0011 | 8 | |
| 4'b0100 | 6 | $f_{DTS_CK}/2$ |
| 4'b0101 | 8 | |
| 4'b0110 | 6 | $f_{DTS_CK}/4$ |
| 4'b0111 | 8 | |
| 4'b1000 | 6 | $f_{DTS_CK}/8$ |
| 4'b1001 | 8 | |
| 4'b1010 | 5 | $f_{DTS_CK}/16$ |
| 4'b1011 | 6 | |
| 4'b1100 | 8 | |
| 4'b1101 | 5 | $f_{DTS_CK}/32$ |
| 4'b1110 | 6 | |
| 4'b1111 | 8 | |

7 MSM 主-从模式

该位被用来同步被选择的定时器同时开始计数。通过 TRIGI 和 TRGO，定时器被连接在一起，TRGO 用做启动事件。

0：主从模式禁能

1：主从模式使能

6:4 TRGS[2:0] 触发选择

该位域用来指定选择哪一个信号作为用来同步计数器的触发输入源

000: ITI0

001: ITI1

010: ITI2

011: ITI3

100: CI0F_ED

101: CI0FE0

| | | |
|---|---|---|
| | | 110: CI1FE1<br>111: ETIFP<br>从模式被使能后这些位不能改 |
| 3 | 保留 | 必须保持复位值 |
| 2:0 | SMC[2:0] | 从模式控制<br>000：关闭从模式. 如果 CEN=1，则预分频器直接由内部时钟驱动<br>001：译码器模式 0. 根据 CI1FE1 的电平，计数器在 CI0FE0 的边沿向上/下计数<br>010：译码器模式 1. 根据 CI0FE0 的电平，计数器在 CI1FE1 的边沿向上/下计数<br>011：译码器模式 2. 根据另一个信号的输入电平，计数器在 CI0FE0 和 CI1FE1 的边沿向上/ 下计数<br>100：复位模式. 选中的触发输入的上升沿重新初始化计数器，并且产生更新事件<br>101：暂停模式. 当触发输入为高时，计数器的时钟开启。一旦触发输入变为低，则计数器时钟停止<br>110：事件模式.计数器在触发输入的上升沿启动。<br>111：外部时钟模式 0. 选中的触发输入的上升沿驱动计数器 |

### DMA 和中断使能寄存器(TIMERx_DMAINTEN)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | TRGDEN | CMTDEN | CH3DEN | CH2DEN | CH1DEN | CH0DEN | UPDEN | BRKIE | TRGIE | CMTIE | CH3IE | CH2IE | CH1IE | CH0IE | UPIE |
| | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:15 | 保留 | 必须保持复位值。 |
| 14 | TRGDEN | 触发 DMA 请求使能<br>0：禁止触发 DMA 请求<br>1：使能触发 DMA 请求 |
| 13 | CMTDEN | 换相 DMA 更新请求使能<br>0：禁止换相 DMA 更新请求<br>1：使能换相 DMA 更新请求 |
| 12 | CH3DEN | 通道 3 比较/捕获 DMA 请求使能<br>0：禁止通道 3 比较/捕获 DMA 请求<br>1：使能通道 3 比较/捕获 DMA 请求 |
| 11 | CH2DEN | 通道 2 比较/捕获 DMA 请求使能 |

0：禁止通道 2 比较/捕获 DMA 请求
1：使能通道 2 比较/捕获 DMA 请求

| | | |
|---|---|---|
| 10 | CH1DEN | 通道 1 比较/捕获 DMA 请求使能<br>0：禁止通道 1 比较/捕获 DMA 请求<br>1：使能通道 1 比较/捕获 DMA 请求 |
| 9 | CH0DEN | 通道 0 比较/捕获 DMA 请求使能<br>0：禁止通道 0 比较/捕获 DMA 请求<br>1：使能通道 0 比较/捕获 DMA 请求 |
| 8 | UPDEN | 更新 DMA 请求使能<br>0：禁止更新 DMA 请求<br>1：使能更新 DMA 请求 |
| 7 | BRKIE | 中止中断使能<br>0：禁止中止中断<br>1：使能中止中断 |
| 6 | TRGIE | 触发中断使能<br>0：禁止触发中断<br>1：使能触发中断 |
| 5 | CMTIE | 换相更新中断使能<br>0：禁止换相更新中断<br>1：使能换相更新中断 |
| 4 | CH3IE | 通道 3 比较/捕获中断使能<br>0：禁止通道 3 中断<br>1：使能通道 3 中断 |
| 3 | CH2IE | 通道 2 比较/捕获中断使能<br>0：禁止通道 2 中断<br>1：使能通道 2 中断 |
| 2 | CH1IE | 通道 1 比较/捕获中断使能<br>0：禁止通道 1 中断<br>1：使能通道 1 中断 |
| 1 | CH0IE | 通道 0 比较/捕获中断使能<br>0：禁止通道 0 中断<br>1：使能通道 0 中断 |
| 0 | UPIE | 更新中断使能<br>0：禁止更新中断<br>1：使能更新中断 |

### 中断标志寄存器(TIMERx_INTF)

地址偏移：0x10

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | CH3OF | CH2OF | CH1OF | CH0OF | 保留 | BRKIF | TRGIF | CMTIF | CH3IF | CH2IF | CH1IF | CH0IF | UPIF |
| | | | rc_w0 | rc_w0 | rc_w0 | rc_w0 | | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:13 | 保留 | 必须保持复位值。 |
| 12 | CH3OF | 通道 3 捕获溢出标志<br>参见 CH0OF 描述。 |
| 11 | CH2OF | 通道 2 捕获溢出标志<br>参见 CH0OF 描述。 |
| 10 | CH1OF | 通道 1 捕获溢出标志<br>参见 CH0OF 描述。 |
| 9 | CH0OF | 通道 0 捕获溢出标志<br>当通道 0 被配置为输入模式时，在 CH0IF 标志位已经被置 1 后，捕获事件再次发生时，该标志位可以由硬件置 1。该标志位由软件清 0。<br>0：无捕获溢出中断发生。<br>1：发生了捕获溢出中断。 |
| 8 | 保留 | 必须保持复位值。 |
| 7 | BRKIF | 中止中断标志位<br>当中止输入有效时，由硬件对该位置'1'。<br>当中止输入无效时，则该位可由软件清'0'。<br>0：无中止事件产生。<br>1：中止输入上检测到有效电平。 |
| 6 | TRGIF | 触发中断标志<br>当发生触发事件时，此标志会置 1，此位由软件清 0。当暂停模式使能时，触发输入的任意边沿都可以产生触发事件。否则，其它模式时，仅在触发输入端检测到有效边沿，产生触发事件。<br>0：无触发事件产生。<br>1：触发中断产生。 |
| 5 | CMTIF | 通道换相更新中断标志<br>当通道换相更新事件发生时此标志位被硬件置 1，此位由软件清 0。 |

0：无通道换相更新中断发生。
1：通道换相更新中断发生。

| | | |
|---|---|---|
| 4 | CH3IF | 通道 3 比较/捕获中断标志<br>参见 CH0IF 描述。 |
| 3 | CH2IF | 通道 2 比较/捕获中断标志<br>参见 CH0IF 描述。 |
| 2 | CH1IF | 通道 1 比较/捕获中断标志<br>参见 CH0IF 描述。 |
| 1 | CH0IF | 通道 0 比较/捕获中断标志<br>此标志由硬件置 1 软件清 0。当通道 0 在输入模式下时，捕获事件发生时此标志位被置 1；当通道 0 在输出模式下时，此标志位在一个比较事件发生时被置 1。<br>0：无通道 0 中断发生。<br>1：通道 0 中断发生。 |
| 0 | UPIF | 更新中断标志<br>此位在任何更新事件发生时由硬件置 1，软件清 0。<br>0：无更新中断发生。<br>1：发生更新中断。 |

### 软件事件产生寄存器(TIMERx_SWEVG)

地址偏移：0x14
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | BRKG | TRGG | CMTG | CH3G | CH2G | CH1G | CH0G | UPG |
| | | | | | | | | w | w | w | w | w | w | w | w |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7 | BRKG | 产生中止事件<br>该位由软件置 1，用于产生一个中止事件，由硬件自动清 0。当此位被置 1 时，POEN 位被清 0 且 BRKIF 位被置 1，若开启对应的中断和 DMA，则产生相应的中断和 DMA 传输。<br>0：不产生中止事件。<br>1：产生中止事件。 |
| 6 | TRGG | 触发事件产生<br>此位由软件置 1，由硬件自动清 0. 当此位被置 1，TIMERx_INTF 寄存器的 TRGIF |

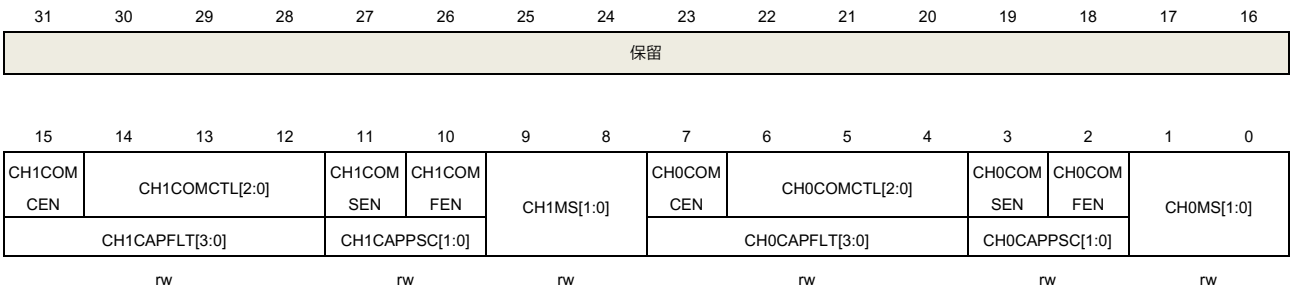
标志位被置 1，若开启对应的中断和 DMA，则产生相应的中断和 DMA 传输。
0：无触发事件产生。
1：产生触发事件。

| | | |
|---|---|---|
| 5 | CMTG | 通道换相更新事件发生<br>此位由软件置 1，由硬件自动清 0. 当此位被置 1，通道捕获/比较控制寄存器(CHxEN, CHxNEN 和 CHxCOMCTL) 的互补输出被更新。<br>0：不产生通道控制更新事件。<br>1：产生通道控制更新事件。 |
| 4 | CH3G | 通道 3 捕获或比较事件发生<br>参见 CH0G 描述。 |
| 3 | CH2G | 通道 2 捕获或比较事件发生<br>参见 CH0G 描述。 |
| 2 | CH1G | 通道 1 捕获或比较事件发生<br>参见 CH0G 描述。 |
| 1 | CH0G | 通道 0 捕获或比较事件发生<br>该位由软件置 1，用于在通道 0 产生一个捕获/比较事件，由硬件自动清 0。当此位被置 1，CH0IF 标志位被置 1，若开启对应的中断和 DMA，则发出相应的中断和 DMA 请求。此外，如果通道 0 配置为输入模式，计数器的当前值被 TIMERx_CH0CV 寄存器捕获，如果 CH0IF 标志位已经为 1，则 CH0OF 标志位被置 1。<br>0：不产生通道 0 捕获或比较事件。<br>1：发生通道 0 捕获或比较事件。 |
| 0 | UPG | 更新事件产生<br>此位由软件置 1，被硬件自动清 0。当此位被置 1，如果选择了中央对齐或向上计数模式，计数器被清 0。否则(向下计数模式)计数器将载入自动重载值，预分频计数器将同时被清除。<br>0：无更新事件产生。<br>1：产生更新事件。 |

### 通道控制寄存器 0(TIMERx_CHCTL0)

地址偏移：0x18
复位值：0x0000 0000

该寄存器只能按字(32位)访问。



**输出比较模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15 | CH1COMCEN | 通道 1 输出比较清 0 使能<br>参见 CH0COMCEN 描述。 |
| 14:12 | CH1COMCTL[2:0] | 通道 1 输出比较模式<br>参见 CH0COMCTL 描述。 |
| 11 | CH1COMSEN | 通道 1 输出比较影子寄存器使能<br>参见 CH0COMSEN 描述。 |
| 10 | CH1COMFEN | 通道 1 输出比较快速使能<br>参见 CH0COMFEN 描述。 |
| 9:8 | CH1MS[1:0] | 通道 1 模式选择<br>这些位定义了通道的方向和输入信号的选择。只有当通道关闭(TIMERx_CHCTL2 寄存器的 CH1EN 位被清 0)时这些位才可以写。<br>00：通道 1 配置为输出<br>01：通道 1 配置为输入，IS1 映射在 CI1FE1 上<br>10：通道 1 配置为输入，IS1 映射在 CI0FE1 上<br>11：通道 1 配置为输入，IS1 映射在 ITS 上<br>注意：当 CH1MS[1:0]=11 时，需要通过 TRGS 位(位于 TIMERx_SMCFG 寄存器)选择内部触发输入。 |
| 7 | CH0COMCEN | 通道 0 输出比较清 0 使能<br>当此位被置 1，当检测到 ETIFP 信号输入高电平时，O0CPRE 参考信号被清 0<br>0：禁止通道 0 输出比较清零<br>1：使能通道 0 输出比较清零 |
| 6:4 | CH0COMCTL[2:0] | 通道 0 输出比较模式<br>此位定义了输出准备信号 O0CPRE 的输出比较模式，而 O0CPRE 决定了 CH0_O、CH0_ON 的值。另外，O0CPRE 高电平有效，而 CH0_O、CH0_ON 通道的极性取决于 CH0P、CH0NP 位。<br>000：时基。输出比较寄存器 TIMERx_CH0CV 与计数器 TIMERx_CNT 间的比较对 O0CPRE 不起作用<br>001：匹配时设置为高。当计数器的值与捕获/比较值寄存器 TIMERx_CH0CV 相同时，强制 O0CPRE 为高。<br>010：匹配时设置为低。当计数器的值与捕获/比较值寄存器 TIMERx_CH0CV 相同时，强制 O0CPRE 为低。<br>011：匹配时翻转。当计数器的值与捕获/比较值寄存器 TIMERx_CH0CV 相同时，强制 O0CPRE 翻转。<br>100：强制为低。强制 O0CPRE 为低电平<br>101：强制为高。强制 O0CPRE 为高电平<br>110：PWM 模式 0。在向上计数时，一旦计数器值小于 TIMERx_CH0CV 时，O0CPRE 为高电平，否则为低电平。在向下计数时，一旦计数器的值大于 TIMERx_CH0CV 时， |

| | | |
|---|---|---|
| | | O0CPRE 为低电平，否则为高电平。<br>111：PWM 模式 1。在向上计数时，一旦计数器值小于 TIMERx_CH0CV 时，O0CPRE 为低电平，否则为高电平。在向下计数时，一旦计数器的值大于 TIMERx_CH0CV 时，O0CPRE 为高电平，否则为低电平。<br>如果配置在 PWM 模式下，只有当输出比较模式从时基模式变为 PWM 模式或者比较结果改变时，O0CPRE 电平才改变。<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 且 CH0MS =00(比较模式)时此位不能被改变。 |
| 3 | CH0COMSEN | 通道 0 输出比较影子寄存器使能<br>当此位被置 1，TIMERx_CH0CV 寄存器的影子寄存器被使能，影子寄存器在每次更新事件时都会被更新。<br>0：禁止通道 0 输出/比较影子寄存器<br>1：使能通道 0 输出/比较影子寄存器<br>仅在单脉冲模式下(SPM =1)，可以在未确认影子寄存器的情况下使用 PWM 模式<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 且 CH0MS =00 时此位不能被改变。 |
| 2 | CH0COMFEN | 通道 0 输出比较快速使能<br>0：通道 0 输出比较快速禁能。仅比较结果输入作为有效边沿时会产生比较匹配，并将 CH0_O 设置为比较电平，激活 CH0_O 输出的最小延时为 5 个时钟周期。<br>1：通道 0 输出比较快速使能。触发输入信号的有效边沿和比较结果都会产生比较匹配，并将 CH0_O 设置为比较电平。当触发信号的输入作为有效边沿时，激活 CH0_O 输出的最小延时为 3 个时钟周期。 |
| 1:0 | CH0MS[1:0] | 通道 0 I/O 模式选择<br>这些位定义了通道的工作模式和输入信号的选择。只有当通道关闭(TIMERx_CHCTL2 寄存器的 CH0EN 位被清 0)时这些位才可写。<br>00：通道 0 配置为输出<br>01：通道 0 配置为输入，IS0 映射在 CI0FE0 上<br>10：通道 0 配置为输入，IS0 映射在 CI1FE0 上<br>11：通道 0 配置为输入，IS0 映射在 ITS 上<br>注意：当 CH0MS[1:0]=11 时，需要通过 TRGS 位(位于 TIMERx_SMCFG 寄存器)选择内部触发输入 |

**输入捕获模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:12 | CH1CAPFLT[3:0] | 通道 1 输入捕获滤波控制<br>参见 CH0CAPFLT 描述。 |
| 11:10 | CH1CAPPSC[1:0] | 通道 1 输入捕获预分频器<br>参见 CH0CAPPSC 描述。 |
| 9:8 | CH1MS[1:0] | 通道 1 模式选择<br>与输出模式相同。 |

7:4 CH0CAPFLT[3:0] 通道 0 输入捕获滤波控制

CI0 输入信号可以通过数字滤波器进行滤波，该位域配置滤波参数。

数字滤波器的基本原理：根据 $f_{SAMP}$ 对 CI0 输入信号进行连续采样，并记录信号相同电平的次数。达到该位配置的滤波参数后，认为是有效电平。

滤波器参数配置如下：

| CH0CAPFLT [3:0] | 采样次数 | $f_{SAMP}$ |
|---|---|---|
| 4'b0000 | 无滤波器 | |
| 4'b0001 | 2 | $f_{CK_TIMER}$ |
| 4'b0010 | 4 | |
| 4'b0011 | 8 | |
| 4'b0100 | 6 | $f_{DTS}/2$ |
| 4'b0101 | 8 | |
| 4'b0110 | 6 | $f_{DTS}/4$ |
| 4'b0111 | 8 | |
| 4'b1000 | 6 | $f_{DTS}/8$ |
| 4'b1001 | 8 | |
| 4'b1010 | 5 | $f_{DTS}/16$ |
| 4'b1011 | 6 | |
| 4'b1100 | 8 | |
| 4'b1101 | 5 | $f_{DTS}/32$ |
| 4'b1110 | 6 | |
| 4'b1111 | 8 | |

3:2 CH0CAPPSC[1:0] 通道 0 输入捕获预分频器

这 2 位定义了通道 0 输入的预分频系数。当 TIMERx_CHCTL2 寄存器中的 CH0EN =0 时，则预分频器复位。

00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获。

01：每 2 个事件触发一次捕获。

10：每 4 个事件触发一次捕获。

11：每 8 个事件触发一次捕获。

1:0 CH0MS[1:0] 通道 0 模式选择

与输出比较模式相同。

### 通道控制寄存器 1(TIMERx_CHCTL1)

地址偏移：0x1C

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH3COM CEN | CH3COMCTL[2:0] | | | CH3COM SEN | CH3COM FEN | CH3MS[1:0] | | CH2COM CEN | CH2COMCTL[2:0] | | | CH2COM SEN | CH2COM FEN | CH2MS[1:0] | |

| CH3CAPFLT[3:0] | CH3CAPPSC[1:0] | | CH2CAPFLT[3:0] | CH2CAPPSC[1:0] | |
|---|---|---|---|---|---|
| rw | rw | rw | rw | rw | rw |

**输出比较模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15 | CH3COMCEN | 通道 3 输出比较清 0 使能<br>参见 CH0COMCEN 描述。 |
| 14:12 | CH3COMCTL[2:0] | 通道 3 输出比较模式<br>参见 CH0COMCTL 描述。 |
| 11 | CH3COMSEN | 通道 3 输出比较影子寄存器使能<br>参见 CH0COMSEN 描述。 |
| 10 | CH3COMFEN | 通道 3 输出比较快速使能<br>参见 CH0COMFEN 描述。 |
| 9:8 | CH3MS[1:0] | 通道 3 模式选择<br>这些位定义了通道的方向和输入信号的选择。只有当通道关闭(TIMERx_CHCTL2 寄存器的 CH3EN 位被清 0)时这些位才可以写。<br>00：通道 3 配置为输出<br>01：通道 3 配置为输入，IS3 映射在 CI3FE3 上<br>10：通道 3 配置为输入，IS3 映射在 CI2FE3 上<br>11：通道 3 配置为输入，IS3 映射在 ITS 上<br>注意：当 CH3MS[1:0]=11 时，需要通过 TRGS 位(位于 TIMERx_SMCFG 寄存器)选择内部触发输入 |
| 7 | CH2COMCEN | 通道 2 输出比较清 0 使能<br>当此位被置 1，当检测到 ETIFP 输入高电平时，O2CPRE 参考信号被清 0<br>0：使能通道 2 输出比较清零<br>1：禁止通道 2 输出比较清零 |
| 6:4 | CH2COMCTL[2:0] | 通道 2 输出比较模式<br>此位定义了输出准备信号 O2CPRE 的输出比较模式，而 O2CPRE 决定了 CH2_O、CH2_ON 的值。另外，O2CPRE 高电平有效，而 CH2_O、CH2_ON 通道的极性取决于 CH2P、CH2NP 位。<br>000：时基。输出比较寄存器 TIMERx_CH2CV 与计数器 TIMERx_CNT 间的比较对 O2CPRE 不起作用<br>001：匹配时设置为高。当计数器的值与捕获/比较值寄存器 TIMERx_CH2CV 相同时，强制 O2CPRE 为高。<br>010：匹配时设置为低。当计数器的值与捕获/比较值寄存器 TIMERx_CH2CV 相同时，强制 O2CPRE 为低。<br>011：匹配时翻转。当计数器的值与捕获/比较值寄存器 TIMERx_CH2CV 相同时，强制 O2CPRE 翻转。<br>100：强制为低。强制 O2CPRE 为低电平<br>101：强制为高。强制 O2CPRE 为高电平 |

| | | |
|---|---|---|
| | | 110：PWM 模式 0。在向上计数时，一旦计数器值小于 TIMERx_CH2CV 时，O2CPRE 为高电平，否则为低电平。在向下计数时，一旦计数器的值大于 TIMERx_CH2CV 时，O2CPRE 为低电平，否则为高电平。<br>111：PWM 模式 1。在向上计数时，一旦计数器值小于 TIMERx_CH2CV 时，O2CPRE 为低电平，否则为高电平。在向下计数时，一旦计数器的值大于 TIMERx_CH2CV 时，O2CPRE 为高电平，否则为低电平。<br>如果配置在 PWM 模式下，只有当输出比较模式从时基模式变为 PWM 模式或者比较结果改变时，O2CPRE 电平才改变。<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 且 CH2MS =00(比较模式)时此位不能被改变。 |
| 3 | CH2COMSEN | 通道 0 输出比较影子寄存器使能<br>当此位被置 1，TIMERx_CH2CV 寄存器的影子寄存器被使能，影子寄存器在每次更新事件时都会被更新。<br>0：禁止通道 2 输出/比较影子寄存器<br>1：使能通道 2 输出/比较影子寄存器<br>仅在单脉冲模式下(SPM =1)，可以在未确认影子寄存器情况下使用 PWM 模式<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 且 CH2MS =00 时此位不能被改变。 |
| 2 | CH2COMFEN | 通道 2 输出比较快速使能<br>0：通道 2 输出比较快速禁能。仅比较结果输入作为有效边沿时会产生比较匹配，并将 CH2_O 设置为比较电平，激活 CH2_O 输出的最小延时为 5 个时钟周期。<br>1：通道 2 输出比较快速使能。触发输入信号的有效边沿和比较结果都会产生比较匹配，并将 CH2_O 设置为比较电平。当触发信号的输入作为有效边沿时，激活 CH2_O 输出的最小延时为 3 个时钟周期。 |
| 1:0 | CH2MS[1:0] | 通道 2 I/O 模式选择<br>这些位定义了通道的工作模式和输入信号的选择。只有当通道关闭(TIMERx_CHCTL2 寄存器的 CH2EN 位被清 0)时这些位才可写。<br>00：通道 2 配置为输出<br>01：通道 2 配置为输入，IS2 映射在 CI2FE2 上<br>10：通道 2 配置为输入，IS2 映射在 CI3FE2 上<br>11：通道 2 配置为输入，IS2 映射在 ITS 上.<br>注意：当 CH2MS[1:0]=11 时，需要通过 TRGS 位(位于 TIMERx_SMCFG 寄存器)选择内部触发输入 |

**输入捕获模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:12 | CH3CAPFLT[3:0] | 通道 3 输入捕获滤波控制<br>参见 CH0CAPFLT 描述。 |
| 11:10 | CH3CAPPSC[1:0] | 通道 3 输入捕获预分频器<br>参见 CH0CAPPSC 描述。 |
| 9:8 | CH3MS[1:0] | 通道 3 模式选择 |

与输出模式相同。

7:4 CH2CAPFLT[3:0] 通道 2 输入捕获滤波控制

CI2 输入信号可以通过数字滤波器进行滤波，该位域配置滤波参数。

数字滤波器的基本原理：根据 $f_{SAMP}$ 对 CI2 输入信号进行连续采样，并记录信号相同电平的次数。达到该位配置的滤波参数后，认为是有效电平。

滤波器参数配置如下：

| CH2CAPFLT [3:0] | 采样次数 | $f_{SAMP}$ |
|---|---|---|
| 4'b0000 | 无滤波器 | |
| 4'b0001 | 2 | $f_{CK_TIMER}$ |
| 4'b0010 | 4 | |
| 4'b0011 | 8 | |
| 4'b0100 | 6 | $f_{DTS}/2$ |
| 4'b0101 | 8 | |
| 4'b0110 | 6 | $f_{DTS}/4$ |
| 4'b0111 | 8 | |
| 4'b1000 | 6 | $f_{DTS}/8$ |
| 4'b1001 | 8 | |
| 4'b1010 | 5 | $f_{DTS}/16$ |
| 4'b1011 | 6 | |
| 4'b1100 | 8 | |
| 4'b1101 | 5 | $f_{DTS}/32$ |
| 4'b1110 | 6 | |
| 4'b1111 | 8 | |

3:2 CH2CAPPSC[1:0] 通道 2 输入捕获预分频器

这 2 位定义了通道 2 输入的预分频系数。当 TIMERx_CHCTL2 寄存器中的 CH2EN =0 时，则预分频器复位。

00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获。

01：每 2 个事件触发一次捕获。

10：每 4 个事件触发一次捕获。

11：每 8 个事件触发一次捕获。

1:0 CH2MS[1:0] 通道 2 模式选择

与输出比较模式相同。

### 通道控制寄存器 2(TIMERx_CHCTL2)

地址偏移：0x20

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 保留 | CH3P | CH3EN | CH2NP | CH2NEN | CH2P | CH2EN | CH1NP | CH1NEN | CH1P | CH1EN | CH0NP | CH0NEN | CH0P | CH0EN |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| Bits | Fields | Descriptions |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值。 |
| 13 | CH3P | 通道 3 极性<br>参考 CH0P 描述。 |
| 12 | CH3EN | 通道 3 使能<br>参考 CH0EN 描述。 |
| 11 | CH2NP | 通道 2 互补输出极性<br>参考 CH0NP 描述 |
| 10 | CH2NEN | 通道 2 互补输出使能<br>参考 CH0NEN 描述。 |
| 9 | CH2P | 通道 2 极性<br>参考 CH0P 描述。 |
| 8 | CH2EN | 通道 2 使能<br>参考 CH0EN 描述。 |
| 7 | CH1NP | 通道 1 互补输出极性<br>参考 CH0NP 描述。 |
| 6 | CH1NEN | 通道 1 互补输出使能<br>参考 CH0NEN 描述。 |
| 5 | CH1P | 通道 1 极性<br>参考 CH0P 描述。 |
| 4 | CH1EN | 通道 1 使能<br>参考 CH0EN 描述。 |
| 3 | CH0NP | 通道 0 互补输出极性<br>当通道 0 配置为输出模式，此位定义了互补输出信号的极性。<br>0：通道0互补输出高电平为有效电平<br>1：通道0互补输出低电平为有效电平<br>当通道 0 配置为输入模式时，此位和 CH0P 联合使用，作为输入信号 CI0 的极性选择控制信号。<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 或 10 时此位不能被更改。 |
| 2 | CH0NEN | 通道 0 互补输出使能<br>当通道 0 配置为输出模式时，将此位置 1 使能通道 0 的互补输出。<br>0：禁止通道 0 互补输出。<br>1：使能通道 0 互补输出。 |
| 1 | CH0P | 通道 0 极性<br>当通道 0 配置为输出模式时，此位定义了输出信号极性。 |

| | | |
|---|---|---|
| | | 0：通道0高电平为有效电平<br>1：通道0低电平为有效电平<br>当通道 0 配置为输入模式时，此位定义了 CI0 信号极性。<br>[CH0NP, CH0P] 将选择 CI0FE0 或者 CI1FE0 的有效边沿或者捕获极性。<br>[CH0NP==0, CH0P==0]：把 CIxFE0 的上升沿作为捕获或者从模式下触发的有效信号，并且 CIxFE0 不会被翻转。<br>[CH0NP==0, CH0P==1]：把 CIxFE0 的下降沿作为捕获或者从模式下触发的有效信号，并且 CIxFE0 会被翻转。<br>[CH0NP==1, CH0P==0]：保留。<br>[CH0NP==1, CH0P==1]：把 CIxFE0 的上升沿和下降沿都作为捕获或者从模式下触发的有效信号，并且 CIxFE0 不会被翻转。<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 或 10 时此位不能被更改。 |
| 0 | CH0EN | 通道 0 捕获/比较使能<br>当通道 0 配置为输出模式时，将此位置 1 使能 CH0_O 信号有效。当通道 0 配置为输入模式时，将此位置 1 使能通道 0 上的捕获事件。<br>0：禁止通道 0。<br>1：使能通道 0。 |

### 计数器寄存器(TIMERx_CNT)

地址偏移：0x24
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

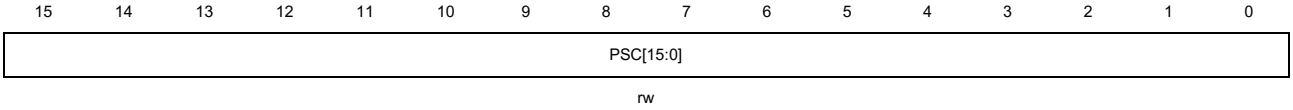
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CNT[15:0] | 这些位是当前的计数值。写操作能改变计数器值。 |

### 预分频寄存器(TIMERx_PSC)

地址偏移： 0x28
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

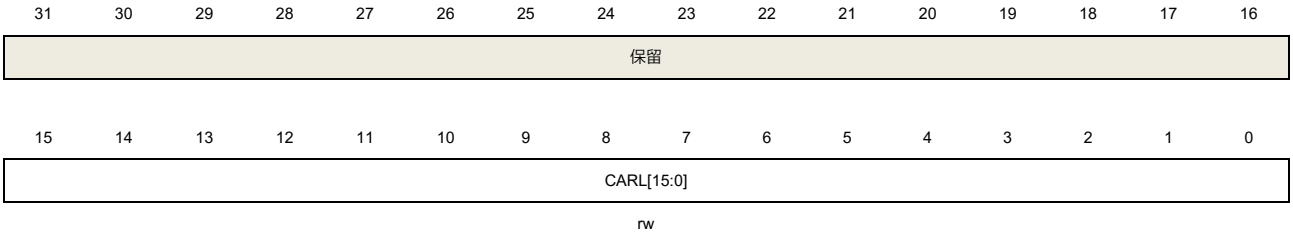



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | PSC[15:0] | 计数器时钟预分频值<br>计数器时钟等于 TIMER_CK 时钟除以(PSC+1)，每次当更新事件产生时，PSC 的值被装入到对应的影子寄存器。 |

### 计数器自动重载寄存器(TIMERx_CAR)

地址偏移：0x2C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

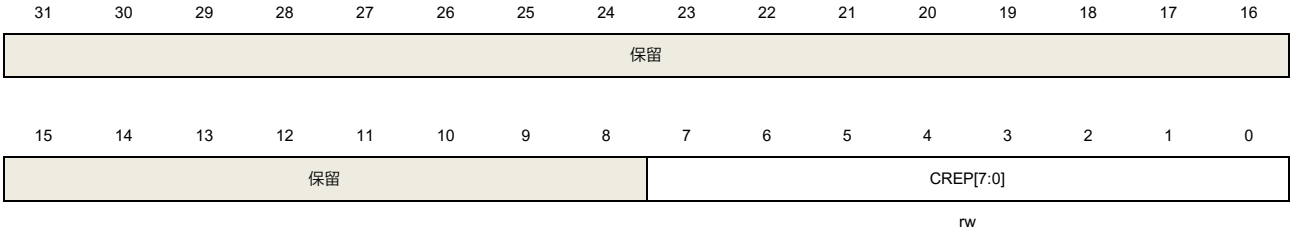



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CARL[15:0] | 计数器自动重载值<br>这些位定义了计数器的自动重载值。 |

### 重复计数寄存器(TIMERx_CREP)

地址偏移：0x30
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|

| | | |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7:0 | CREP[7:0] | 重复计数器的值<br>这些位定义了更新事件的产生速率。重复计数器计数值减为 0 时产生更新事件。影子寄存器的更新速率也会受这些位影响(前提是影子寄存器被使能)。 |

### 通道 0 捕获/比较寄存器(TIMERx_CH0CV)

地址偏移：0x34
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH0VAL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CH0VAL[15:0] | 通道 0 的捕获或比较值<br>当通道 0 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。<br>当通道 0 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。 |

### 通道 1 捕获/比较寄存器(TIMERx_CH1CV)

地址偏移：0x38
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH1VAL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CH1VAL[15:0] | 通道 1 的捕获或比较值<br>当通道 1 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器 |

为只读。
当通道 1 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。

### 通道 2 捕获/比较寄存器(TIMERx_CH2CV)

地址偏移：0x3C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH2VAL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CH2VAL[15:0] | 通道 2 的捕获或比较值<br>当通道 2 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。<br>当通道 2 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。 |

### 通道 3 捕获/比较寄存器(TIMERx_CH3CV)

地址偏移：0x40
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH3VAL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CH3VAL[15:0] | 通道 3 的捕获或比较值<br>当通道 3 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。 |

当通道 3 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。

### 互补通道保护寄存器(TIMERx_CCHP)

地址偏移：0x44
复位值： 0x0000 0000

该寄存器只能按字(32位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| POEN | OAEN | BRKP | BRKEN | ROS | IOS | PROT[1:0] | | DTCFG[7:0] | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | | rw | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |
| 15 | POEN | 所有的通道输出使能<br>该位通过以下方式置 1：<br>-写 1 置位<br>-如果 OAEN=1，则在下一次更新事件发生时置 1.<br>该位通过以下方式清 0：<br>-写 0 清 0<br>-有效的中止输入(异步)<br>如果一个通道配置为输出模式，如果设置了相应的使能位(TIMERx_CHCTL2 寄存器的 CHxEN, CHxNEN 位)，则开启 CHx_O 和 CHx_ON 输出。<br>0：禁止通道输出<br>1：使能通道输出<br>注意：仅当 CHxMS[1:0]=2'b00 时该位有效 |
| 14 | OAEN | 自动输出使能<br>0：POEN 位只能使用软件方式置 1<br>1：如果中止输入无效，下一次更新事件发生时，POEN 位置 1<br>此位只有在 TIMERx_CCHP 寄存器的 PROT [1:0] =00 时才可修改。 |
| 13 | BRKP | 中止极性<br>此位定义了中止输入信号 BRKIN 的极性。<br>0：中止输入低电平有效<br>1：中止输入高电平有效 |
| 12 | BRKEN | 中止使能<br>此位置 1 使能中止事件和 CKM 时钟失败事件输入。<br>0：禁能中止输入 |

1：使能中止输入
此位只有在 TIMERx_CCHP 寄存器的 PROT [1:0] =00 时才可修改。

11 ROS

运行模式下"关闭状态"使能
当 POEN 位被置 1(运行模式)，此位可以被置 1 来使能通道(带有互补输出且配置为输出模式)的输出"关闭状态"。参见*表 18-2. 由参数控制的互补输出表*

0：输出"关闭状态"禁能。当 CHxEN 或者 CHxNEN 位被清零，对应通道为输出"禁能状态"。
1：输出"关闭状态"使能。当 CHxEN 或者 CHxNEN 位被清零，对应通道为输出"关闭状态"。
此位在 TIMERx_CCHP 寄存器的 PROT [1:0]=10 或 11 时不能被更改。

10 IOS

空闲模式下"关闭状态"使能
当 POEN 位被清 0(空闲模式)，此位可以被置 1 来使能通道(带有互补输出且配置为输出模式)的输出"关闭状态"。参见*表 18-2. 由参数控制的互补输出表*。

0：输出"关闭状态"禁能。当 CHxEN 和 CHxNEN 位均被清零，对应通道为输出"禁能状态"。
1：输出"关闭状态"使能。不论 CHxEN 和 CHxNEN 位的值，对应通道为输出"关闭状态"。
此位在 TIMERx_CCHP 寄存器的 PROT [1:0]=10 或 11 时不能被更改。

9:8 PROT[1:0]

互补寄存器保护控制
这两位定义了寄存器的写保护特性。
00：禁能保护模式。无写保护。
01：PROT 模式 0。TIMERx_CTL1 寄存器中 ISOx/ISOxN 位，TIMERx_CCHP 寄存器中 BRKEN/BRKP/OAEN/DTCFG 位写保护。
10：PROT 模式 1。除了 PROT 模式 0 下的寄存器写保护外，还有 TIMERx_CHCTL2 寄存器中 CHxP/CHxNP 位(如果相应通道配置为输出模式)，TIMERx_CCHP 寄存器中 ROS/IOS 位写保护。
11：PROT 模式 2.。除了 PROT 模式 1 下的寄存器写保护外，还有 TIMERx_CHCTLR0/1 中 CHxCOMCTL/ CHxCOMSEN 位(如果相关通道配置为输出模式)写保护。
系统复位后这两位只能被写一次，一旦 TIMERx_CCHP 寄存器被写入，这两位被写保护。

7:0 DTCFG[7:0]

死区时间控制
DTCFG 值和死区时间的关系如下：

| DTCFG[7:5] | The duration of dead-time |
|---|---|
| 3'b0xx | DTCFG[7:0] * $t_{DTS_CK}$ |
| 3'b10x | (64+ DTCFG[5:0]) * $t_{DTS_CK}$ *2 |
| 3'b110 | (32+ DTCFG[4:0]) * $t_{DTS_CK}$ *8 |
| 3'b111 | (32+ DTCFG[4:0]) * $t_{DTS_CK}$ *16 |

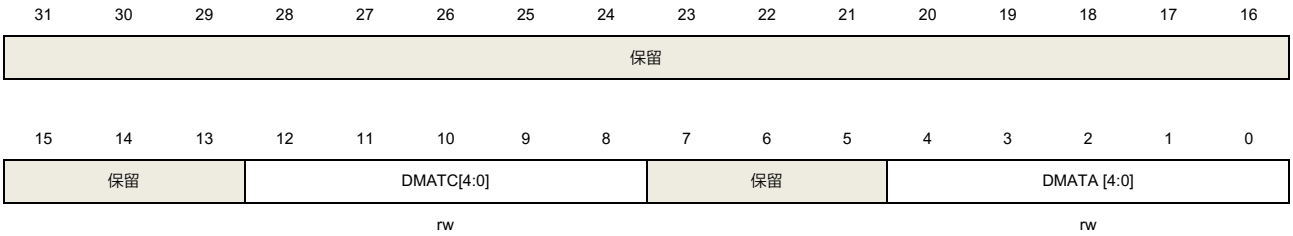
注意：
1. $t_{DTS_CK}$ 是 DTS_CK 的周期，由 TIMERx_CTL0 中的 CKDIC[1:0]定义。

2. 此位只有在 TIMERx_CCHP 寄存器的 PROT [1:0]=00 时才可修改。

### DMA 配置寄存器(TIMERx_DMACFG)

地址偏移：0x48
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

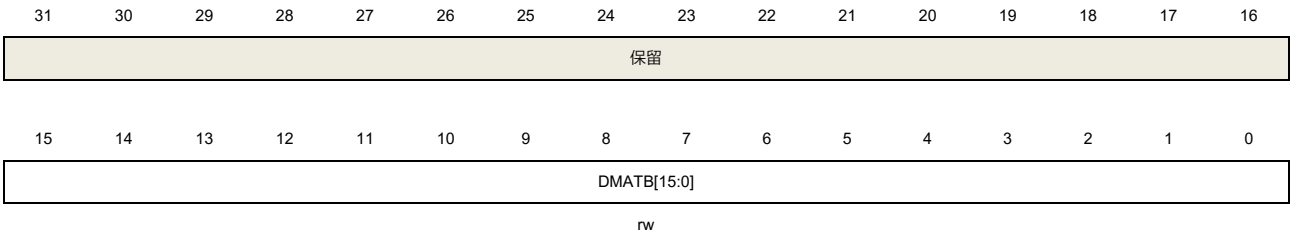



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值。 |
| 12:8 | DMATC [4:0] | DMA 传输计数<br>该位域定义了 DMA 访问(读写)TIMERx_DMATB 寄存器的数量 n，n = (DMATC [4:0] +1). DMATC [4:0] 从 5'b0_0000 到 5'b1_0001. |
| 7:5 | 保留 | 必须保持复位值。 |
| 4:0 | DMATA [4:0] | DMA 传输起始地址<br>该位域定义了 DMA 访问 TIMERx_DMATB 寄存器的第一个地址。当通过 TIMERx_DMA 第一次访问时，访问的就是该位域指定的地址。第二次访问 TIMERx_DMATB 时，将访问起始地址+0x4。 |

### DMA 发送缓冲区寄存器(TIMERx_DMATB)

地址偏移：0x4C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。



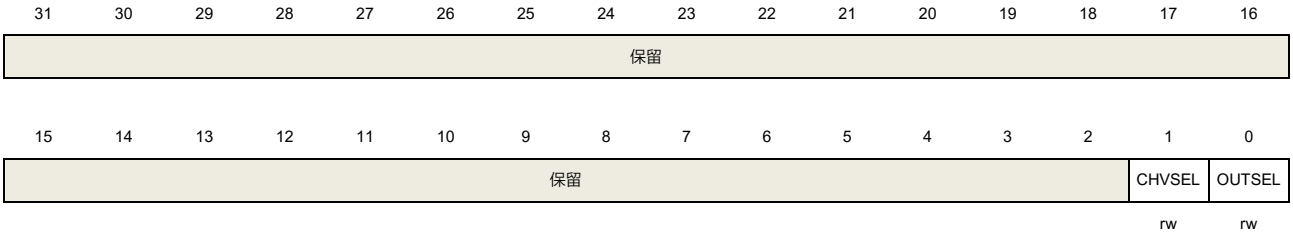

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | DMATB [15:0] | DMA 发送缓冲 |

对这个寄存器的读或写，(起始地址+传输次数*4)地址范围内的寄存器会被访问。传输次数由硬件计算，范围为 0 到 DMATC。

### 配置寄存器(TIMERx_CFG)

地址偏移：0xFC
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |
| 1 | CHVSEL | 写捕获比较寄存器选择位<br>此位由软件写 1 或清 0。<br>1：当写入捕获比较寄存器的值与寄存器当前值相等时，写入操作无效。<br>0：无影响。 |
| 0 | OUTSEL | 输出值选择位<br>此位由软件写 1 或清 0。<br>1：如果 POEN 位与 IOS 位均为 0，则输出无效。<br>0：无影响。 |

# 18.2. 通用定时器 L0(TIMERx, x=1,2,3,4)

## 18.2.1. 简介

通用定时器 L0 是 4 通道定时器，支持输入捕获，输出比较，产生 PWM 信号控制电机和电源管理。通用定时器 L0 计数器是 16 位无符号计数器。

通用定时器是可编程的，可以被用来计数，其外部事件可以驱动其他定时器。

定时器和定时器之间是相互独立，但是他们可以被同步在一起形成一个更大的定时器，这些定时器的计数器一致地增加。

## 18.2.2. 主要特性

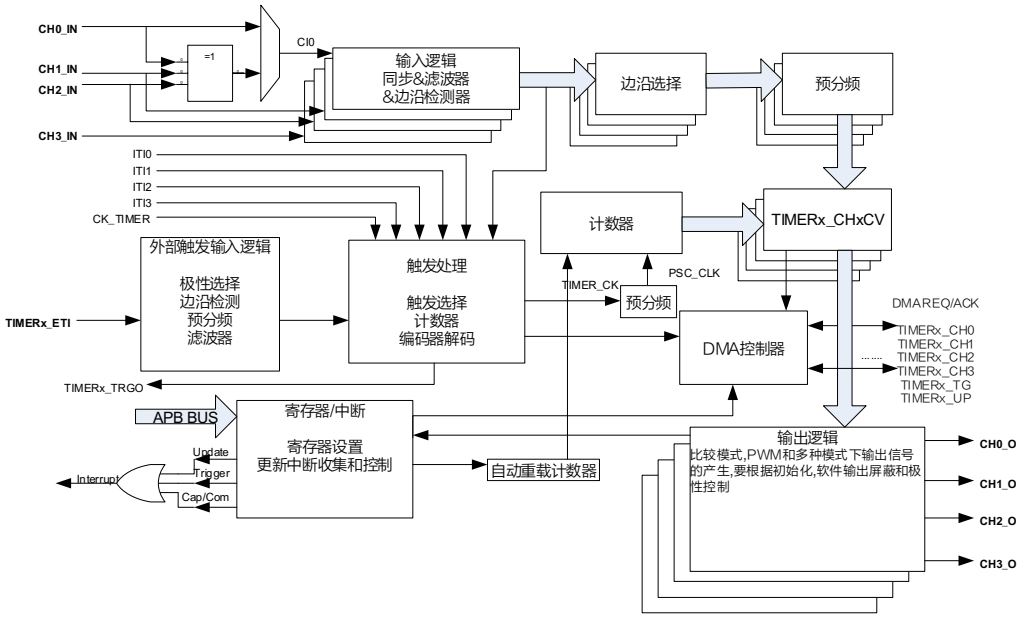
- 总通道数：4；
- 计数器宽度：16位(TIMER2&3)，32位(TIMER1&4)；
- 时钟源可选：内部时钟，内部触发，外部输入，外部触发；
- 多种计数模式：向上计数，向下计数和中央计数；
- 正交译码器接口：被用来追踪运动和分辨旋转方向和位置；
- 霍尔传感器接口：用来做三相电机控制；
- 可编程的预分频器：16位，运行时可以被改变；
- 每个通道可配置：输入捕获模式，输出比较模式，可编程的PWM模式，单脉冲模式；
- 自动重装载功能；
- 中断输出和DMA请求：更新事件，触发事件，比较/捕获事件；
- 多个定时器的菊链使得一个定时器可以同时启动多个定时器；
- 定时器的同步允许被选择的定时器在同一个时钟周期开始计数；
- 定时器主-从管理。

## 18.2.3. 功能描述

### 结构框图

*图 18-28. 通用定时器 L0 结构框图*提供了通用定时器 L0 的内部细节。

**图 18-28. 通用定时器 L0 结构框图**




## 时钟源配置

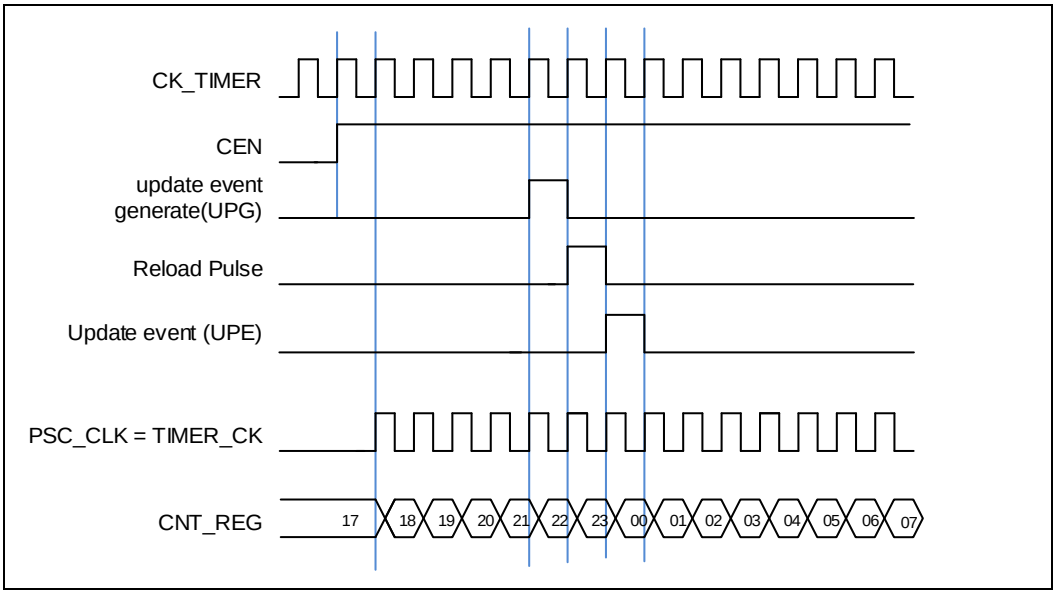
通用定时器 L0 可以由内部时钟源 CK_TIMER 或者由 SMC(TIMERx_SMCFG 寄存器位[2:0])控制的复用时钟源驱动。

■ SMC[2:0]==3'b000，定时器选择内部时钟源(连接到RCU模块的CK_TIMER)

如果 SMC[2:0]==3'b000，默认用来驱动计数器预分频器的是内部时钟源 CK_TIMER。当 CEN 置位，CK_TIMER 经过预分频器(预分频值由 TIMERx_PSC 寄存器确定)产生 PSC_CLK。

如果将 TIMERx_SMCFG 寄存器的 SMC[2:0]设置为 0x1、0x2、0x3 和 0x7，预分频器被其他时钟源(由 TIMERx_SMCFG 寄存器的 TRGS [2:0]区域选择)驱动，在下文说明。当 SMC 位被设置为 0x4、0x5 和 0x6，计数器预分频器时钟源由内部时钟 CK_TIMER 驱动。

**图 18-29. 内部时钟分频为 1 时，计数器的时序图**




■ SMC[2:0]==3'b111(外部时钟模式0)，定时器选择外部输入引脚作为时钟源

计数器预分频器可以在 TIMERx_CI0/ TIMERx_CI1 引脚的每个上升沿或下降沿计数。这种模式可以通过设置 SMC [2:0]为 0x7 同时设置 TRGS [2:0]为 0x4，0x5 或 0x6 来选择。CIx 是 TIMERx_CIx 通过数字滤波器采样后的信号。

计数器预分频器也可以在内部触发信号 ITI0/1/2/3 的上升沿计数。这种模式可以通过设置 SMC [2:0]为 0x7 同时设置 TRGS [2:0]为 0x0, 0x1, 0x2 或者 0x3。

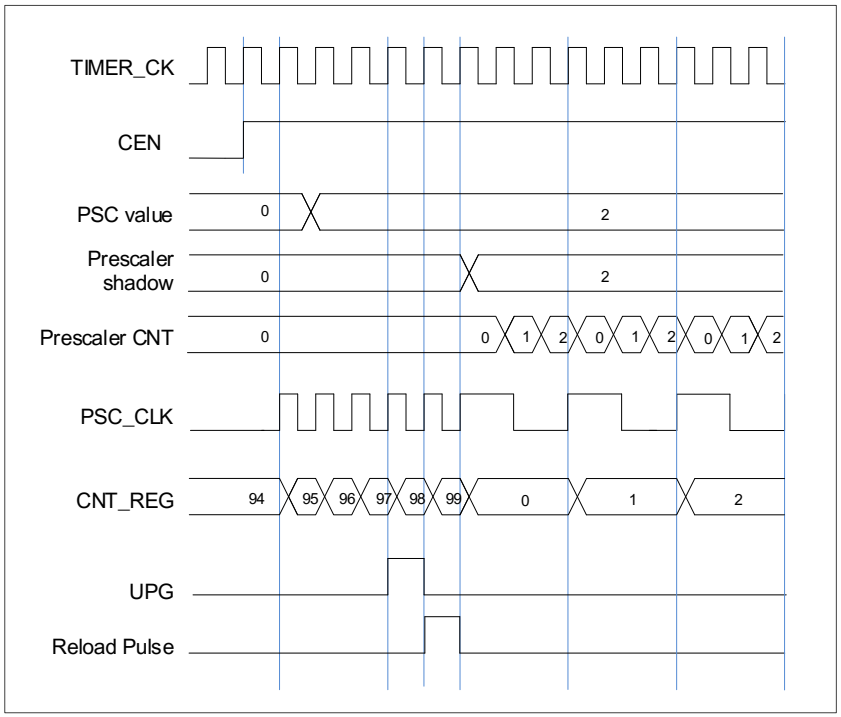
■ SMC==1'b1(外部时钟模式1)，定时器选择外部输入引脚ETI作为时钟源

计数器预分频器可以在外部引脚 ETI 的每个上升沿或下降沿计数。这种模式可以通过设置 TIMERx_SMCFG 寄存器中的 SMC1 位为 1 来选择。另一种选择 ETI 信号作为时钟源方式是，设置 SMC [2:0]为 0x7 同时设置 TRGS [2:0]为 0x7。注意 ETI 信号是通过数字滤波器采样 ETI 引脚得到的。。如果选择 ETIF 信号为时钟源，触发控制器包括边沿监测电路将在每个 ETI 信号上升沿产生一个时钟脉冲来为计数器预分频器提供时钟。

### 时钟预分频器

预分频器可以将定时器的时钟(TIMER_CK)频率按 1 到 65536 之间的任意值分频，分频后的时钟 PSC_CLK 驱动计数器计数。分频系数受预分频寄存器 TIMERx_PSC 控制，这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

**图 18-30. 当 PSC 数值从 0 变到 2 时，计数器的时序图**




### 计数器向上计数模式

在这种模式，计数器的计数方向是向上计数。计数器从 0 开始向上连续计数到自动加载值(定义在 TIMERx_CAR 寄存器中)，一旦计数器计数到自动加载值，会重新从 0 开始向上计数并产生上溢事件。在向上计数模式中，TIMERx_CTL0 寄存器中的计数方向控制位 DIR 应该被设置成 0。

当通过 TIMERx_SWEVG 寄存器的 UPG 位置 1 来设置更新事件时，计数值会被清 0，并产生更新事件。

如果 TIMERx_CTL0 寄存器的 UPDIS 置 1，则禁止更新事件。

当发生更新事件时，所有影子寄存器(计数器自动重载寄存器，预分频寄存器)都将被更新。

下面的这些图给出了一些例子，当 TIMERx_CAR=0x99 时，计数器在不同预分频因子下的行为。

图 18-31. 向上计数时序图，PSC=0/2

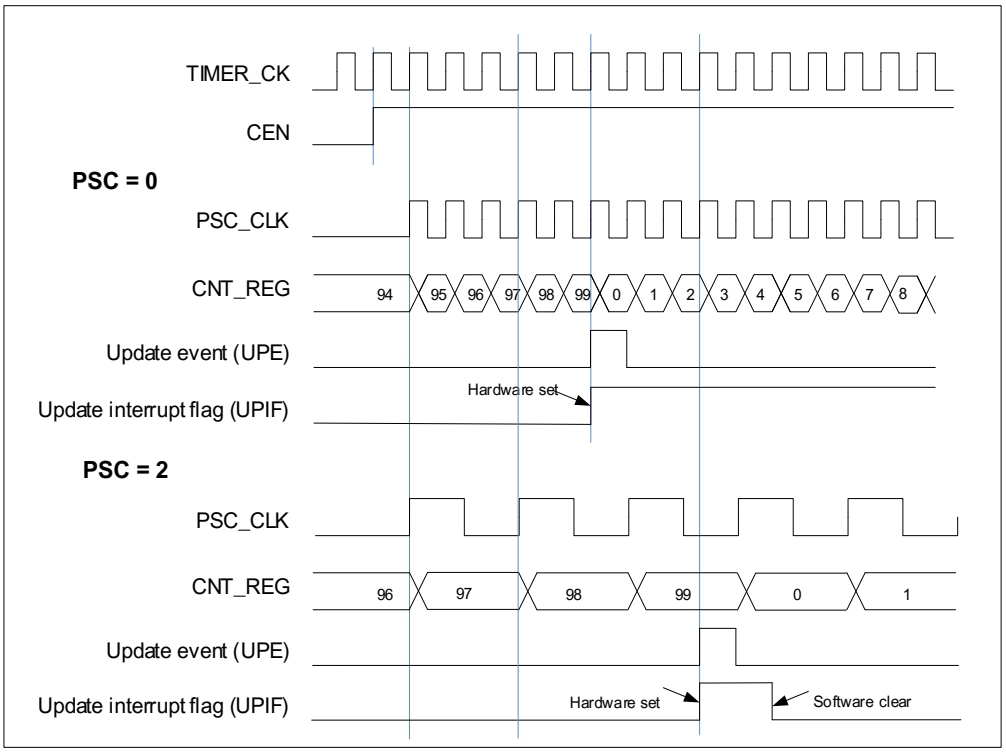
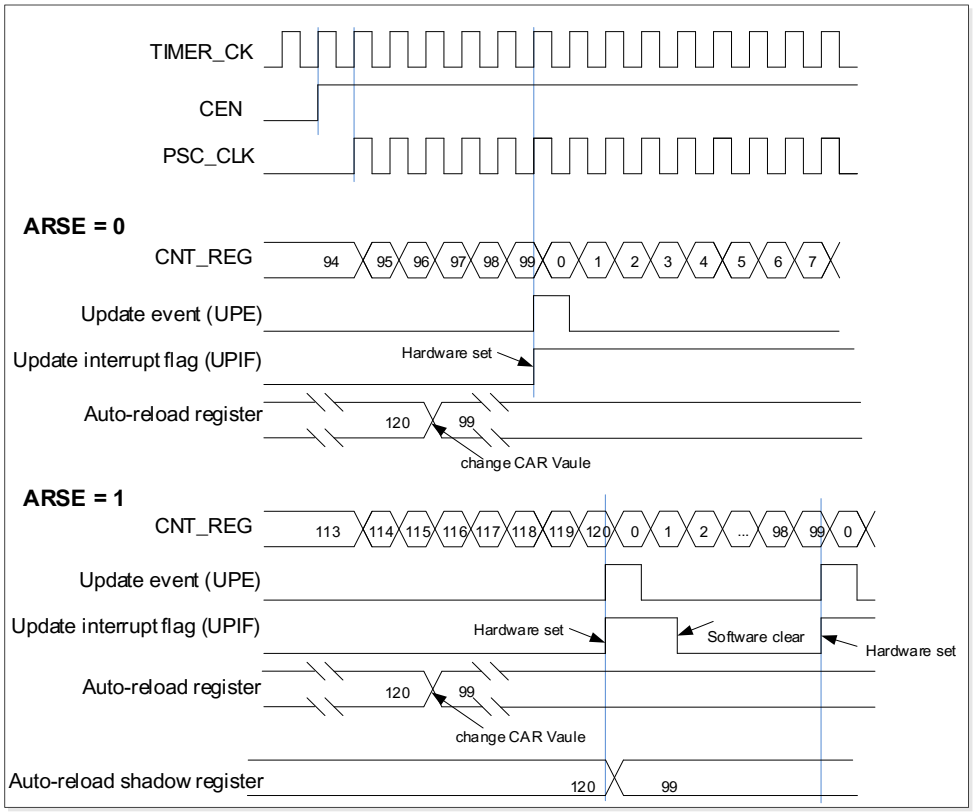



图 18-32. 向上计数时序图，在运行时改变 TIMERx_CAR 寄存器的值



### 计数器向下计数模式

在这种模式，计数器的计数方向是向下计数。计数器从自动加载值(定义在 TIMERx_CAR 寄存器中)向下连续计数到 0。一旦计数器计数到 0，计数器会重新从自动加载值开始计数并产生下溢事件。在向下计数模式中，TIMERx_CTL0 寄存器中的计数方向控制位 DIR 应该被设置成 1。

当通过 TIMERx_SWEVG 寄存器的 UPG 位置 1 来设置更新事件时，计数值会被初始化为自动加载值，并产生更新事件。

如果 TIMERx_CTL0 寄存器的 UPDIS 置 1，则禁止更新事件。

当发生更新事件时，所有影子寄存器(计数器自动重载寄存器，预分频寄存器)都将被更新。

下面这些图给出了一些例子，当 TIMERx_CAR=0x99 时，计数器在不同时钟频率下的行为。

**图 18-33. 向下计数时序图，PSC=0/2**

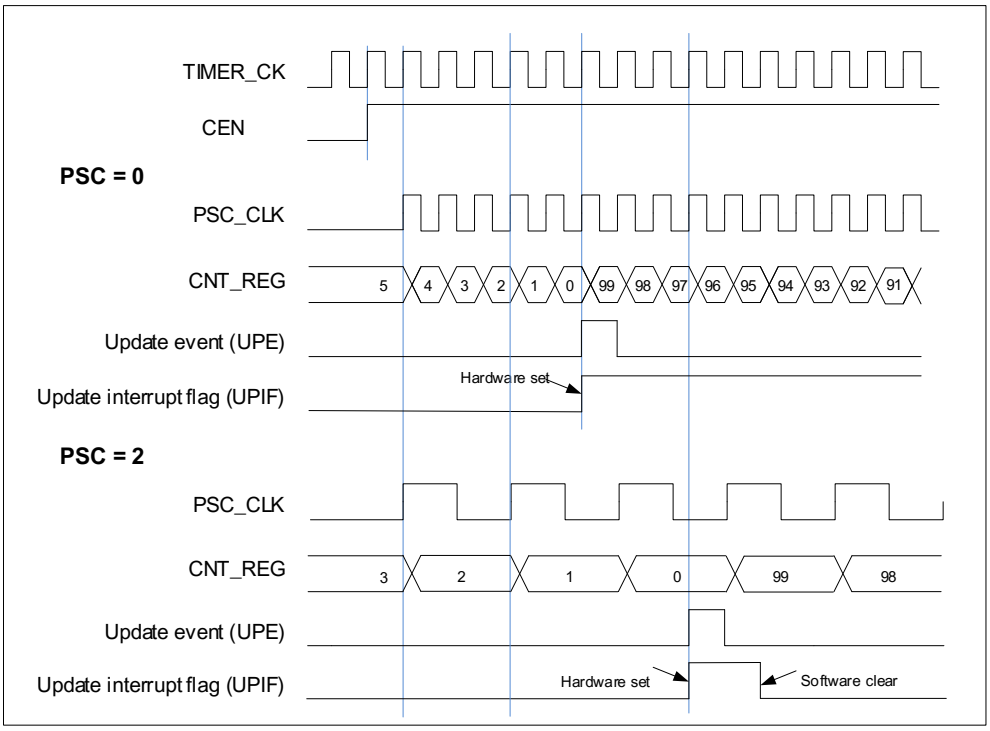
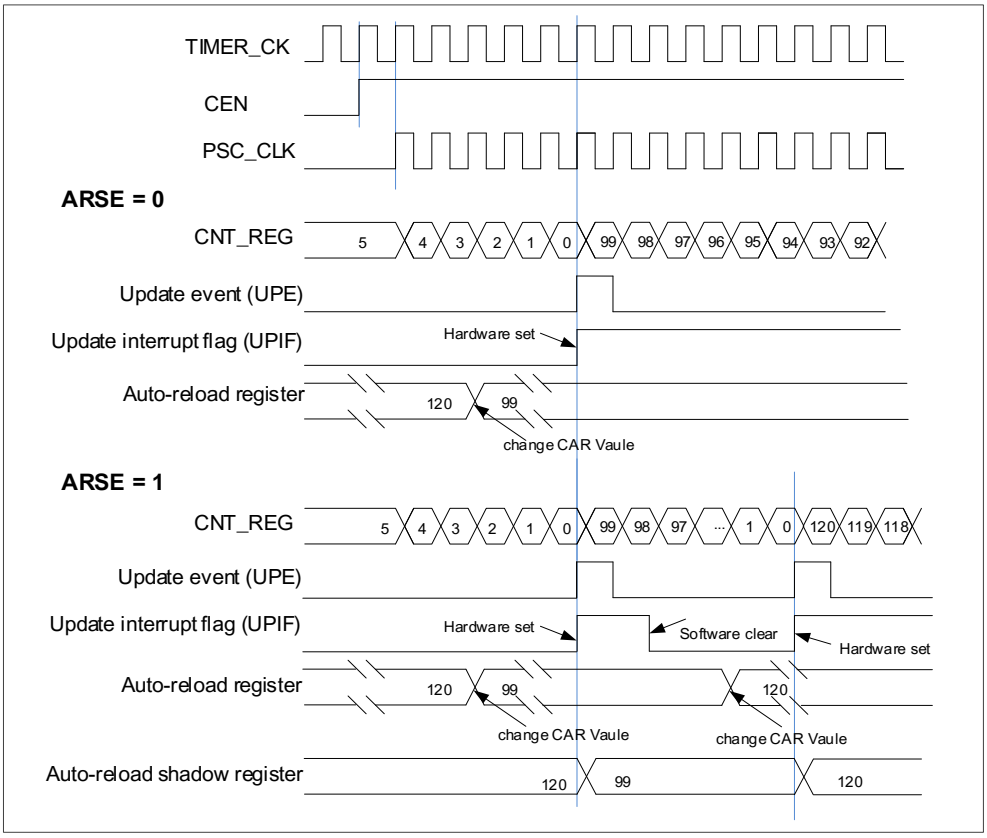


图 18-34. 向下计数时序图，在运行时改变 TIMERx_CAR 寄存器值




## 计数器中央对齐模式

在中央对齐模式下，计数器交替的从 0 开始向上计数到自动加载值，然后再向下计数到 0。。向上计数模式中， 定时器模块在计数器计数到(自动加载值-1)产生一个上溢事件；向下计数模式中，定时器模块在计数器计数到 1 时产生一个下溢事件。在中央计数模式中，TIMERx_CTL0 寄存器中的计数方向控制位 DIR 只读，表明了的计数方向。

将 TIMERx_SWEVG 寄存器的 UPG 位置 1 可以初始化计数值为 0，并产生一个更新事件，而无需考虑计数器在中央模式下是向上计数还是向下计数。

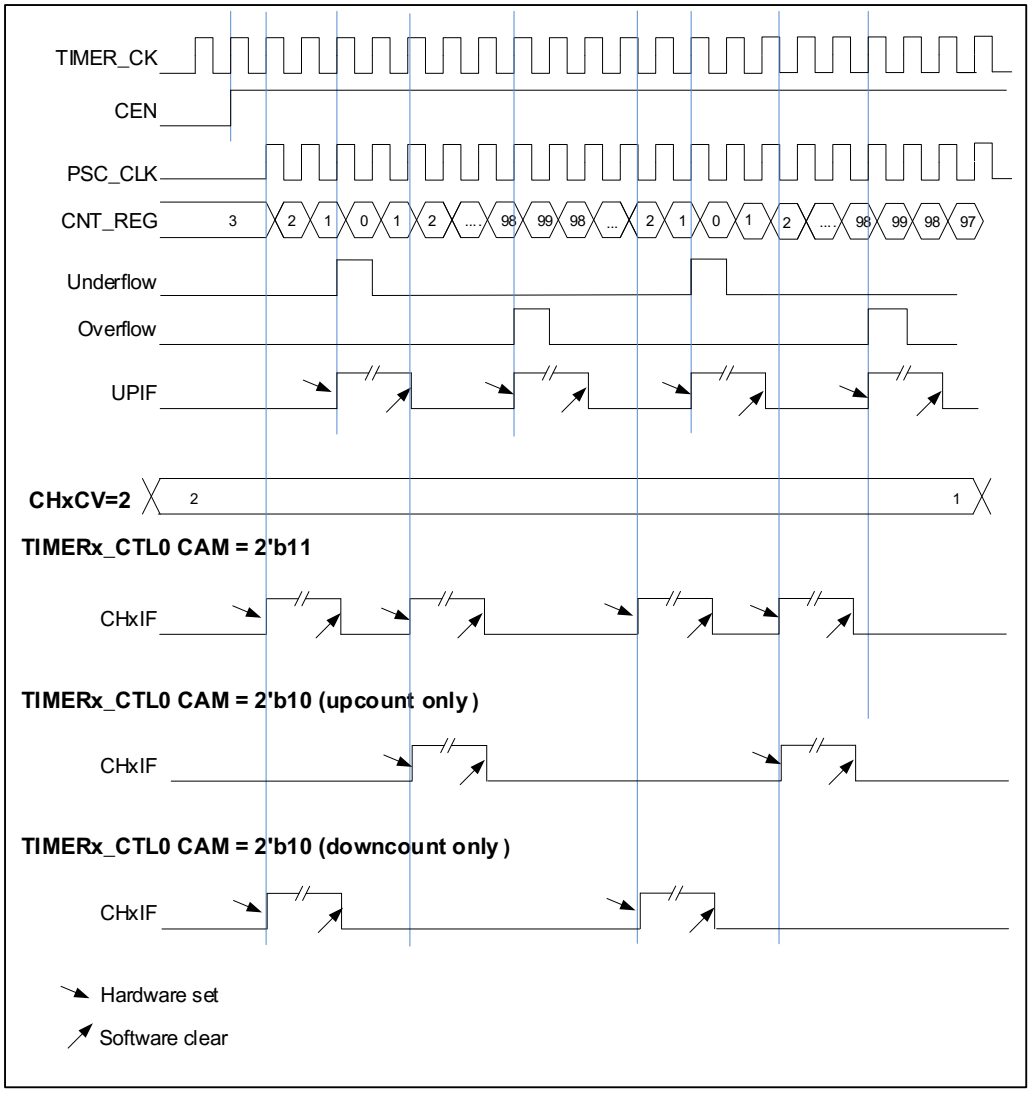
上溢或者下溢时，TIMERx_INTF 寄存器中的 UPIF 位都会被置 1，然而 CHxIF 位置 1 与 TIMERx_CTL0 寄存器中 CAM 的值有关。具体细节参考*图 18-35. 中央计数模式计数器时序图*

如果 TIMERx_CTL0 寄存器的 UPDIS 置 1，则禁止更新事件。

当发生更新事件时，所有影子寄存器(计数器自动重载寄存器，预分频寄存器)都将被更新。

下面这些图给出了一些例子，当 TIMERx_CAR=0x99，TIMERx_PSC=0x0 时，计数器的行为

图 18-35. 中央计数模式计数器时序图



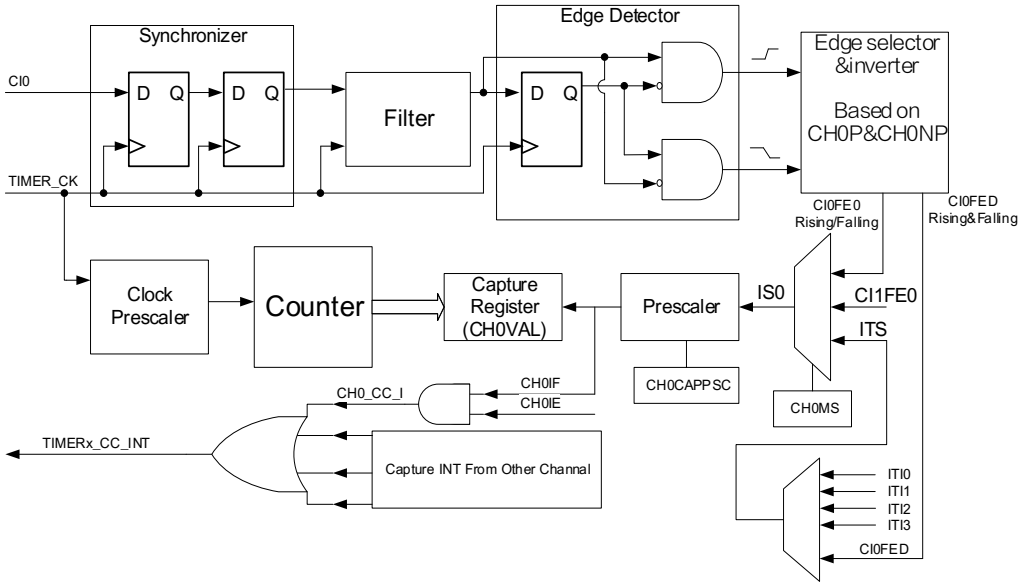

## 输入捕获和输出比较通道

通用定时器 L0 拥有四个独立的通道用于捕获输入或比较输出是否匹配。每个通道都围绕一个通道捕获比较寄存器建立，包括一个输入级，通道控制器和输出级。

■ 通道输入捕获功能

通道输入捕获功能允许通道测量一个波形时序，频率，周期，占空比等。输入级包括一个数字滤波器，一个通道极性选择，边沿检测和一个通道预分频器。如果在输入引脚上出现被选择的边沿，TIMERx_CHxCV 寄存器会捕获计数器当前的值，同时 CHxIF 位被置 1，如果 CHxIE = 1 则产生通道中断。

**图 18-36. 通道输入捕获原理**




通道输入信号 CIx 有两种选择，一种是 TIMERx_CHx 信号，另一种是 TIMERx_CH0,TIMERx_CH1 和 TIMERx_CH2 异或之后的信号。通道输入信号 CIx 先被 TIMER_CK 信号同步，然后经过数字滤波器采样，产生一个被滤波后的信号。通过边沿检测器，可以选择检测上升沿或者下降沿。通过配置 CHxP 选择使用上升沿或者下降沿。配置 CHxMS.，可以选择其他通道的输入信号，内部触发信号。配置 IC 预分频器，使得若干个输入事件后才产生一个有效的捕获事件。捕获事件发生，CxCV 存储计数器的值。

配置步骤如下：

**第一步：**滤波器配置(TIMERx_CHCTL0寄存器中CHxCAPFLT)：
根据输入信号和请求信号的质量，配置相应的CHxCAPFLT。

**第二步：**边沿选择(TIMERx_CHCTL2寄存器中CHxP/CHxNP)：
配置CHxP/CHxNP选择上升沿或者下降沿。

**第三步：**捕获源选择(TIMERx_CHCTL0寄存器中CHxMS)：
一旦通过配置CHxMS选择输入捕获源，必须确保通道配置在输入模式(CHxMS!=0x0)，而且TIMERx_CxCV寄存器不能再被写。

**第四步：**中断使能(TIMERx_DMAINTEN寄存器中CHxIE和CHxDEN)：
使能相应中断，可以获得中断和DMA请求。

**第五步：**捕获使能(TIMERx_CHCTL2寄存器中CHxEN)。

**结果：**当期望的输入信号发生时，TIMERx_CHxCV被设置成当前计数器的值，CHxIF为置1。如果CHxIF位已经为1，则CHxOF位置1。根据TIMERx_DMAINTEN寄存器中CHxIE和CHxDEN的配置，相应的中断和DMA请求会被提出。

**直接产生：**软件设置CHxG位，会直接产生中断和DMA请求。

输入捕获模式也可用来测量 TIMERx_CHx 引脚上信号的脉冲波宽度。例如，一个 PWM 波连接到 CI0。配置 TIMERx_CHCTL0 寄存器中 CH0MS 为 2'b01，选择通道 0 的捕获信号为 CI0 并设置上升沿捕获。配置 TIMERx_CHCTL0 寄存器中 CH1MS 为 2'b10，选择通道 1 捕获信号为 CI0 并设置下降沿捕获。计数器配置为复位模式，在通道 0 的上升沿复位。

TIMERX_CH0CV 寄存器测量 PWM 的周期值，TIMERx_CH1CV 寄存器测量 PWM 占空比值。

■ 通道输出比较功能

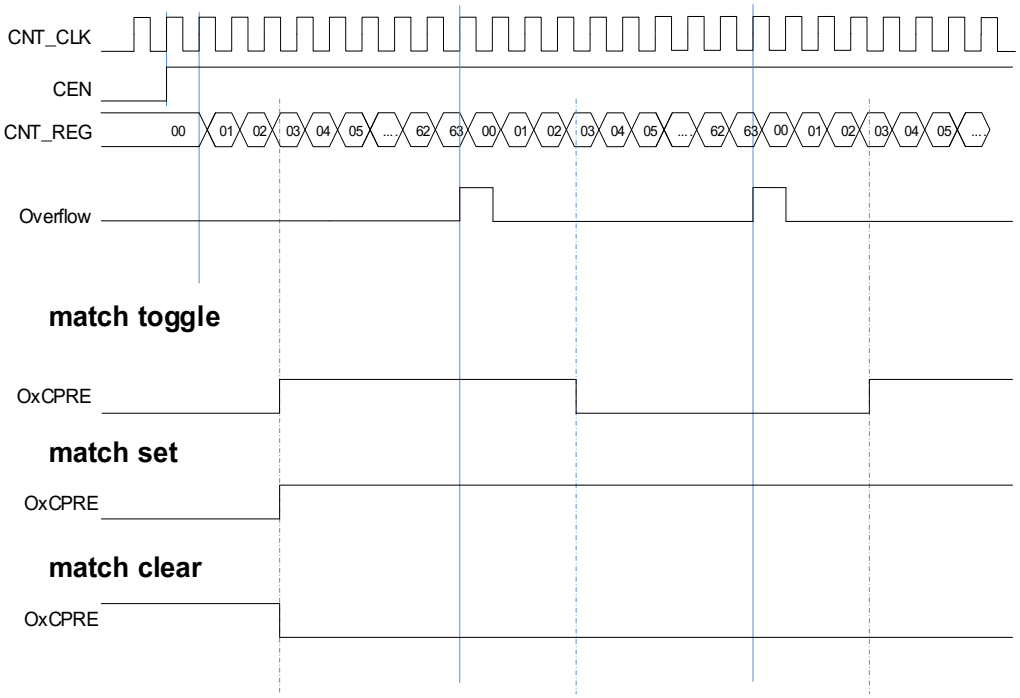
在输出比较模式，TIMERx 可以产生时控脉冲，其位置，极性，持续时间和频率都是可编程的。当一个输出通道的 CxCV 寄存器与计数器的值匹配时，根据 CHxCOMCTL 的配置，这个通道的输出可以被置高电平，被置低电平或者反转。当计数器的值与 CxCV 寄存器的值匹配时，CHxIF 位被置 1，如果 CHxIE = 1 则会产生中断，如果 CxCDE=1 则会产生 DMA 请求。

配置步骤如下：

**第一步：**时钟配置：
配置定时器时钟源，预分频器等。
**第二步：**比较模式配置：
设置CHxCOMSEN位来配置输出比较影子寄存器；
设置CHxCOMCTL位来配置输出模式(置高电平/置低电平/反转)；
设置CHxP/CHxNP位来选择有效电平的极性；
设置CHxEN使能输出。
**第三步**：通过CHxIE/CxCDE位配置中断/DMA请求使能。
**第四步：**通过TIMERx_CAR寄存器和TIMERx_CHxCV寄存器配置输出比较时基：
CxCV可以在运行时根据你所期望的波形而改变。
**第五步：**设置CEN位使能定时器。

*图 18-37. 三种输出比较模式*显示了三种比较输出模式：反转/置高电平/置低电平，CAR=0x63，CxCV=0x3。

**图 18-37. 三种输出比较模式**



### 输出 PWM 功能

在 PWM 输出模式下(PWM 模式 0 是配置 CHxCOMCTL 为 3'b110，PWM 模式 1 是配置 CHxCOMCTL 为 3'b111)，通道根据 TIMERx_CAR 寄存器和 TIMERx_CHxCV 寄存器的值，输出 PWM 波形。

根据计数模式，我们可以分为两种 PWM 波：EAPWM(边沿对齐 PWM)和 CAPWM(中央对齐 PWM)。

EAPWM 的周期由 TIMERx_CAR 寄存器值决定，占空比由 TIMERx_CHxCV 寄存器值决定。

*图 18-38. EAPWM 时序图*显示了 EAPWM 的输出波形和中断。

CAPWM 的周期由(2*TIMERx_CAR 寄存器值)决定，占空比由(2*TIMERx_CHxCV 寄存器值)决定。*图 18-39. CAPWM 时序图*显示了 CAPWM 的输出波形和中断。

在 PWM0 模式下(CHxCOMCTL==3'b110)，如果 TIMERx_CHxCV 寄存器的值大于 TIMERx_CAR 寄存器的值，通道输出一直为有效电平。

在 PWM0 模式下(CHxCOMCTL==3'b110)，如果 TIMERx_CHxCV 寄存器的值等于 0，通道输出一直为无效电平。

**图 18-38. EAPWM 时序图**

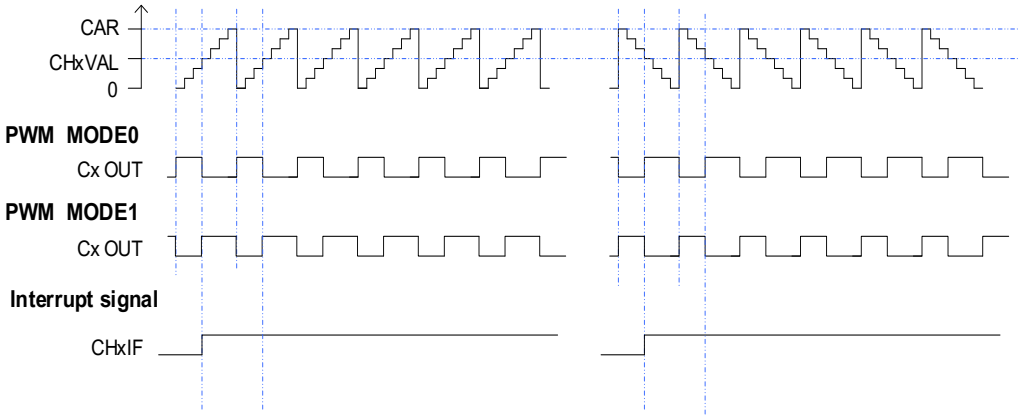
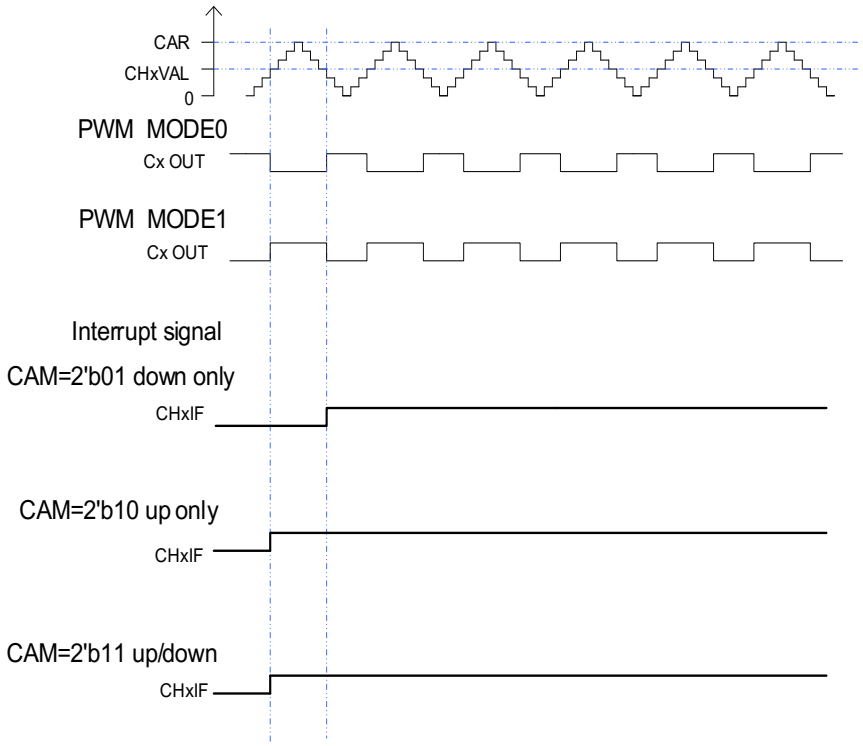


**图 18-39. CAPWM 时序图**




### 通道输出准备信号

当 TIMERx 用于输出匹配比较模式下，设置 CHxCOMCTL 位可以定义 OxCPRE 信号(通道 x 准备信号)类型。OxCPRE 信号有若干类型的输出功能，包括，设置 CHxCOMCTL=0x00 可以保持原始电平；设置 CHxCOMCTL=0x01 可以将 OxCPRE 信号设置为高电平；设置 CHxCOMCTL=0x02 可以将 OxCPRE 信号设置为低电平；设置 CHxCOMCTL=0x03，在计数器值和 TIMERx_CHxCV 寄存器的值匹配时，可以翻转输出信号。

PWM 模式 0 和 PWM 模式 1 是 OxCPRE 的另一种输出类型，设置 CHxCOMCTL 位域位 0x06 或 0x07 可以配置 PWM 模式 0/PWM 模式 1。在这些模式中，根据计数器值和 TIMERx_CHxCV 寄存器值的关系以及计数方向，OxCPRE 信号改变其电平。具体细节描述，请参考相应的位。

设置 CHxCOMCTL =0x04 或 0x05 可以实现 OxCPRE 信号的强制输出功能。输出比较信号能够直接由软件强置为有效或无效状态，而不依赖于 TIMERx_CHxCV 的值和计数器值之间间的比较结果。

设置 CHxCOMCEN=1，当由外部 ETI 引脚信号产生的 ETIFE 信号为高电平时，OxCPRE 被强制为低电平。在下一次更新事件到来时，OxCPRE 信号才会回到有效电平状态。

### 正交译码器

参考*正交译码器*。

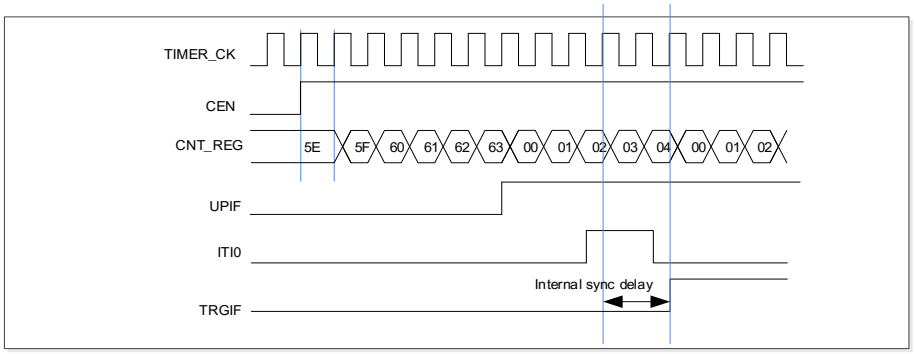
### 霍尔传感器接口功能

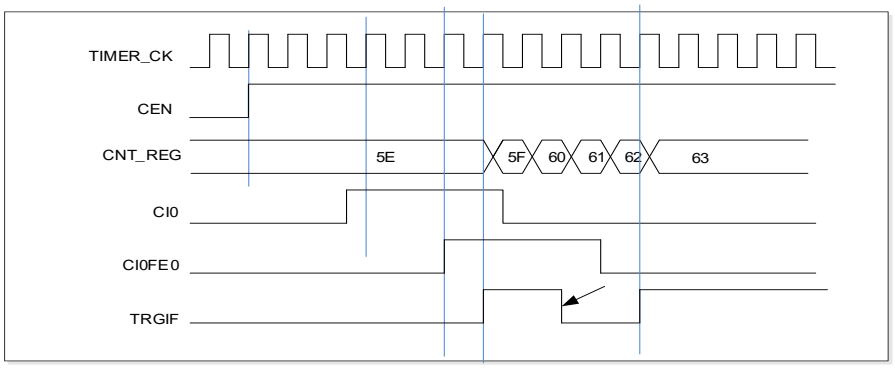
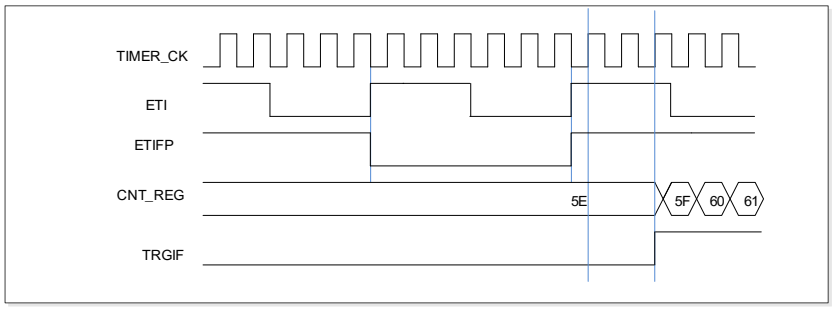
参考*霍尔传感器接口功能*。

### 主-从管理

TIMERx 能在多种模式下同步外部触发，包括复位模式，暂停模式和事件模式，可以通过设置 TIMERx_SMCFG 寄存器中的 SMC [2:0]配置这些模式。这些模式的输入触发源可以通过设置 TIMERx_SMCFG 寄存器中的 TRGS [2:0]来选择。

**表 18-5. 从模式列表和举例**

| | 模式选择 | 触发源选择 | 极性选择 | 滤波和预分频 |
|---|---|---|---|---|
| **列举** | SMC[2:0]<br>3'b100(复位模式)<br>3'b101(暂停模式)<br>3'b110(事件模式) | TRGS[2:0]<br>000：ITI0<br>001：ITI1<br>010：ITI2<br>011：ITI3<br>100：CI0F_ED<br>101：CI0FE0<br>110：CI1FE1<br>111：ETIFP | 如果触发源是CI0FE0或者CI1FE1，配置CHxP和CHxNP来选择极性和反相。<br>如果触发源是ETIFP，配置ETP选择极性和反相。 | 若触发源为ITIx，滤波和预分频不可用。<br>若触发源为CIx，可配置CHxCAPFLT设置滤波，预分频不可用。<br>若触发源为ETIFP，滤波和预分频均可用。 |
| **例1** | **复位模式**<br>当触发输入上升沿到来时，计数器清零重启。 | TRGS[2:0]=3'b000<br>选择ITI0为触发源。 | 若触发源是ITI0，极性选择不可用。 | 若触发源是ITI0，滤波和预分频不可用。 |
| | **图 18-40. 复位模式**<br><br> | | | |
| **例2** | **暂停模式**<br>当触发输入为低的时候，计数器暂停计数，当触发输入为高时，计数器计数。 | TRGS[2:0]=3'b101<br>选择CI0FE0为触发源。 | TI0S=0(非异或)[CH0NP=0，CH0P=0] CI0FE0不反相。捕获发生在上升沿 。 | 在这个例子中滤波被旁路。 |

| | 模式选择 | 触发源选择 | 极性选择 | 滤波和预分频 |
|---|---|---|---|---|
| | 图 18-41. 暂停模式<br><br> | | | |
| 例3 | **事件模式**<br>触发输入的上升沿计数器开始计数。 | TRGS[2:0]=3'b111<br>选择ETIFP为触发源。 | ETP = 0，ETI极性不改变。 | ETPSC = 1，ETI 2分频。<br>ETFC = 0，ETI 无滤波。 |
| | 图 18-42. 事件模式<br><br> | | | |

### 单脉冲模式

参考*单脉冲模式*。

### 定时器互连

参考*高级定时器(TIMERx,x=0,7)*。

### 定时器 DMA 模式

定时器 DMA 模式是指通过 DMA 模块配置定时器的寄存器。有两个跟定时器 DMA 模式相关的寄存器：TIMERx_DMACFG 和 TIMERx_DMATB。当然，必须要使能 DMA 请求，一些内部中断事件可以产生 DMA 请求。当中断事件发生，TIMERx 会给 DMA 发送请求。DMA 配置成 M2P 模式，PADDR 是 TIMERx_DMATB 寄存器地址，DMA 就会访问 TIMERx_DMATB 寄存器。实际上，TIMERx_DMATB 寄存器只是一个缓冲，定时器会将 TIMERx_DMATB 映射到一个内部寄存器，这个内部寄存器由 TIMERx_DMACFG 寄存器中的 DMATA 来指定。如果 TIMERx_DMACFG 寄存器的 DMATC 位域值为 0，表示 1 次传输，定时器的发送 1 个 DMA 请

求就可以完成。如果 TIMERx_DMACFG 寄存器的 DMATC 位域值不为 1，例如其值为 3，表示 4 次传输，定时器就需要再多发 3 次 DMA 请求。在这 3 次请求下，DMA 对 TIMERx_DMATB 寄存器的访问会映射到访问定时器的 DMATA+0x4, DMATA+0x8, DMATA+0xc 寄存器。总之，发生一次 DMA 内部中断请求，定时器会连续发送(DMATC+1)次请求。

如果再来 1 次 DMA 请求事件，TIMERx 将会重复上面的过程。

### 定时器调试模式

当 Cortex®-M4 内核停止，DBG_CTL2 寄存器中的 TIMERx_HOLD 配置位被置 1，定时器计数器停止。

### 18.2.4. TIMERx 寄存器(x=1,2,3,4)

TIMER1 基地址：0x4000 0000

TIMER2 基地址：0x4000 0400

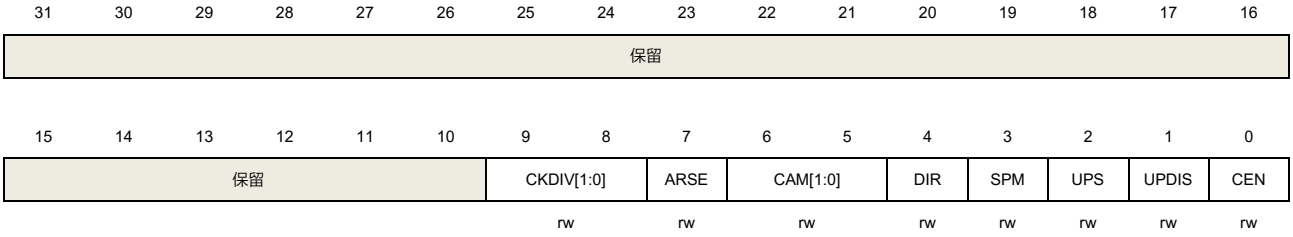
TIMER3 基地址：0x4000 0800

TIMER4 基地址：0x4000 0C00

#### 控制寄存器 0(TIMERx_CTL0)

地址偏移：0x00
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:10 | 保留 | 必须保持复位值。 |
| 9:8 | CKDIV[1:0] | 时钟分频<br>通过软件配置CKDIV，规定定时器时钟(CK_TIMER) 与死区时间和数字滤波器采样时钟(DTS)之间的分频系数。<br>00：$f_{DTS}=f_{CK_TIMER}$<br>01：$f_{DTS}= f_{CK_TIMER} /2$<br>10：$f_{DTS}= f_{CK_TIMER} /4$<br>11：保留 |
| 7 | ARSE | 自动重载影子使能<br>0：禁能 TIMERx_CAR 寄存器的影子寄存器。<br>1：使能 TIMERx_CAR 寄存器的影子寄存器。 |
| 6:5 | CAM[1:0] | 计数器对齐模式选择<br>00：无中央对齐计数模式(边沿对齐模式)。 DIR位指定了计数方向<br>01：中央对齐向下计数置1模式。计数器在中央计数模式计数，通道被配置在输出模式(TIMERx_CHCTL0寄存器中CHxMS=00)，只有在向下计数时，CHxF位置1<br>10：中央对齐向上计数置1模式。计数器在中央计数模式计数，通道被配置在输出模式(TIMERx_CHCTL0寄存器中CHxMS=00)，只有在向上计数时，CHxF位置1<br>11：中央对齐上下计数置1模式。计数器在中央计数模式计数，通道被配置在输出模式(TIMERx_CHCTL0寄存器中CHxMS=00)，在向上和向下计数时，CHxF位都会置1 |

当计数器使能以后，该位不能从 0x00 切换到非 0x00

| | | |
|---|---|---|
| 4 | DIR | 方向<br>0：向上计数<br>1：向下计数<br>当计数器配置为中央对齐计数模式或译码器模式时，该位只读。 |
| 3 | SPM | 单脉冲模式<br>0：单脉冲模式禁能。更新事件发生后，计数器继续计数<br>1：单脉冲模式使能。在下一次更新事件发生时，计数器停止计数 |
| 2 | UPS | 更新请求源<br>软件配置该位，选择更新事件源.<br>0：以下事件均会产生更新中断或DMA请求：<br>UPG位被置1<br>计数器溢出/下溢<br>复位模式产生的更新<br>1：下列事件会产生更新中断或DMA请求：<br>计数器溢出/下溢 |
| 1 | UPDIS | 禁止更新.<br>该位用来使能或禁能更新事件的产生<br>0：更新事件使能. 更新事件发生时，相应的影子寄存器被装入预装载值，以下事件均会产生更新事件：<br>UPG位被置1<br>计数器溢出/下溢<br>复位模式产生的更新<br>1：更新事件禁能.<br>注意：当该位被置 1 时，UPG 位被置 1 或者复位模式不会产生更新事件，但是计数器和预分频器被重新初始化 |
| 0 | CEN | 计数器使能<br>0：计数器禁能。<br>1：计数器使能。<br>在软件将CEN位置1后，外部时钟、暂停模式和译码器模式才能工作。 |

### 控制寄存器 1(TIMERx_CTL1)

地址偏移：0x04
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | TI0S | MMC[2:0] | | | DMAS | 保留. | | |

rw rw rw

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7 | TI0S | 通道0触发输入选择<br>0：选择 TIMERx_CH0 引脚作为通道 0 的触发输入。<br>1：选择 TIMERx_CH0, CH1 和 CH2 引脚异或的结果作为通道 0 的触发输入。 |
| 6:4 | MMC[2:0] | 主模式控制l<br>这些位控制TRGO信号的选择，TRGO信号由主定时器发给从定时器用于同步功能。<br>000：当产生一个定时器复位事件后，输出一个TRGO信号，定时器复位源为：<br>主定时器产生一个复位事件<br>TIMERx_SWEVG寄存器中UPG位置1<br>001：当产生一个定时器使能事件后，输出一个TRGO信号，定时器使能源为：<br>CEN位置1<br>在暂停模式下，触发输入置1<br>010：当产生一个定时器更新事件后，输出一个TRGO信号，更新事件源由UPDIS和UPS位决定<br>011：当通道0在发生一次捕获或一次比较成功时，主模式控制器产生一个TRGO脉冲<br>100：当产生一次比较事件时，输出一个TRGO信号，比较事件源来自O0CPRE<br>101：当产生一次比较事件时，输出一个TRGO信号，比较事件源来自O1CPRE<br>110：当产生一次比较事件时，输出一个TRGO信号，比较事件源来自O2CPRE<br>111：当产生一次比较事件时，输出一个 TRGO 信号，比较事件源来自 O3CPRE |
| 3 | DMAS | DMA 请求源选择<br>0：当通道捕获/比较事件发生时，发送通道 x 的 DMA 请求。<br>1：当更新事件发生，发送通道 x 的 DMA 请求。 |
| 2:0 | 保留 | 必须保持复位值。 |

### 从模式配置寄存器(TIMERx_SMCFG)

地址偏移：0x08
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ETP | SMC1 | ETPSC[1:0] | | ETFC[3:0] | | | | MSM | TRGS[2:0] | | | 保留. | SMC[2:0] | | |
| rw | rw | rw | | rw | | | | rw | rw | | | | rw | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值 |

15 ETP 外部触发极性

该位指定 ETI 信号的极性

0：ETI 高电平或上升沿有效 .

1：ETI 低电平或下降沿有效 .

14 SMC1 SMC 的一部分为了使能外部时钟模式 1

在外部时钟模式 1，计数器由 ETIFP 信号上的任意有效边沿驱动

0：外部时钟模式 1 禁能

1：外部时钟模式 1 使能

当从模式配置为复位模式，暂停模式和事件模式时，定时器仍然可以工作在外部时钟模式 1。但是 TRGS 必须不能为 3'b111。

如果外部时钟模式 0 和外部时钟模式 1 同时被被配置，外部时钟的输入是 ETIF。

注意：外部时钟模式 0 使能在寄存器的 SMC[2:0]位域。

13:12 ETPSC[1:0] 外部触发预分频

外部触发信号 ETIFP 的频率不能超过 TIMER_CK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETIFP 的频率。

00：预分频禁能

01：2 分频

10：4 分频

11：8 分频

11:8 ETFC[3:0] 外部触发滤波控制

外部触发信号可以通过数字滤波器进行滤波，该位域定义了数字滤波器的滤波能力。数字滤波器的基本原理是：以 $f_{SAMP}$ 频率连续采样外部触发信号，同时记录采样相同电平的次数。当该次数达到配置的滤波能力时，则认为是一个有效的电平信号。

| EXTFC[3:0] | 次数 | $f_{SAMP}$ |
|---|---|---|
| 4'b0000 | Filter disabled. | |
| 4'b0001 | 2 | $f_{CK_TIMER}$ |
| 4'b0010 | 4 | |
| 4'b0011 | 8 | |
| 4'b0100 | 6 | $f_{DTS_CK}/2$ |
| 4'b0101 | 8 | |
| 4'b0110 | 6 | $f_{DTS_CK}/4$ |
| 4'b0111 | 8 | |
| 4'b1000 | 6 | $f_{DTS_CK}/8$ |
| 4'b1001 | 8 | |
| 4'b1010 | 5 | $f_{DTS_CK}/16$ |
| 4'b1011 | 6 | |
| 4'b1100 | 8 | |
| 4'b1101 | 5 | $f_{DTS_CK}/32$ |
| 4'b1110 | 6 | |
| 4'b1111 | 8 | |

7 MSM 主-从模式

该位被用来同步被选择的定时器同时开始计数。通过 TRIGI 和 TRGO，定时器被连

| | | |
|---|---|---|
| | | 接在一起，TRGO 用做启动事件。<br>0：主从模式禁能<br>1：主从模式使能 |
| 6:4 | TRGS[2:0] | 触发选择<br>该位域用来指定选择哪一个信号作为用来同步计数器的触发输入源<br>000: ITI0<br>001: ITI1<br>010: ITI2<br>011: ITI3<br>100: CI0F_ED<br>101: CI0FE0<br>110: CI1FE1<br>111: ETIFP<br>从模式被使能后这些位不能改 |
| 3 | 保留 | 必须保持复位值 |
| 2:0 | SMC[2:0] | 从模式控制<br>000：关闭从模式. 如果 CEN=1，则预分频器直接由内部时钟驱动<br>001：译码器模式 0. 根据 CI1FE1 的电平，计数器在 CI0FE0 的边沿向上/下计数<br>010：译码器模式 1. 根据 CI0FE0 的电平，计数器在 CI1FE1 的边沿向上/下计数<br>011：译码器模式 2. 根据另一个信号的输入电平，计数器在 CI0FE0 和 CI1FE1 的边沿向上/ 下计数<br>100：复位模式. 选中的触发输入的上升沿重新初始化计数器，并且产生更新事件.<br>101：暂停模式. 当触发输入为高时，计数器的时钟开启。一旦触发输入变为低，则计数器时钟停止<br>110：事件模式.计数器在触发输入的上升沿启动。<br>111：外部时钟模式 0. 选中的触发输入的上升沿驱动计数器 |

### DMA 和中断使能寄存器(TIMERx_DMAINTEN)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | TRGDEN | 保留 | CH3DEN | CH2DEN | CH1DEN | CH0DEN | UPDEN | 保留 | TRGIE | 保留 | CH3IE | CH2IE | CH1IE | CH0IE | UPIE |
| | rw | | rw | rw | rw | rw | rw | | rw | | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:15 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 14 | TRGDEN | 触发 DMA 请求使能<br>0：禁止触发 DMA 请求。<br>1：使能触发 DMA 请求。 |
| 13 | 保留 | 必须保持复位值。 |
| 12 | CH3DEN | 通道 3 比较/捕获 DMA 请求使能<br>0：禁止通道 3 比较/捕获 DMA 请求。<br>1：使能通道 3 比较/捕获 DMA 请求。 |
| 11 | CH2DEN | 通道 2 比较/捕获 DMA 请求使能<br>0：禁止通道 2 比较/捕获 DMA 请求。<br>1：使能通道 2 比较/捕获 DMA 请求。 |
| 10 | CH1DEN | 通道 1 比较/捕获 DMA 请求使能<br>0：禁止通道 1 比较/捕获 DMA 请求。<br>1：使能通道 1 比较/捕获 DMA 请求。 |
| 9 | CH0DEN | 通道 0 比较/捕获 DMA 请求使能<br>0：禁止通道 0 比较/捕获 DMA 请求。<br>1：使能通道 0 比较/捕获 DMA 请求。 |
| 8 | UPDEN | 更新 DMA 请求使能<br>0：禁止更新 DMA 请求。<br>1：使能更新 DMA 请求。 |
| 7 | 保留 | 必须保持复位值。 |
| 6 | TRGIE | 触发中断使能<br>0：禁止触发中断。<br>1：使能触发中断。 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | CH3IE | 通道 3 比较/捕获中断使能<br>0：禁止通道 3 中断。<br>1：使能通道 3 中断。 |
| 3 | CH2IE | 通道 2 比较/捕获中断使能<br>0：禁止通道 2 中断。<br>1：使能通道 2 中断。 |
| 2 | CH1IE | 通道 1 比较/捕获中断使能<br>0：禁止通道 1 中断。<br>1：使能通道 1 中断。 |
| 1 | CH0IE | 通道 0 比较/捕获中断使能<br>0：禁止通道 0 中断。<br>1：使能通道 0 中断。 |
| 0 | UPIE | 更新中断使能 |

0：禁止更新中断。
1：使能更新中断。

### 中断标志寄存器(TIMERx_INTF)

地址偏移：0x10
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | CH3OF | CH2OF | CH1OF | CH0OF | 保留 | | TRGIF | 保留 | CH3IF | CH2IF | CH1IF | CH0IF | UPIF |
| | | | rc_w0 | rc_w0 | rc_w0 | rc_w0 | | | rc_w0 | | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:13 | 保留 | 必须保持复位值。 |
| 12 | CH3OF | 通道 3 捕获溢出标志<br>参见 CH0OF 描述。 |
| 11 | CH2OF | 通道 2 捕获溢出标志<br>参见 CH0OF 描述。 |
| 10 | CH1OF | 通道 1 捕获溢出标志<br>参见 CH0OF 描述。 |
| 9 | CH0OF | 通道 1 捕获溢出标志<br>当通道 0 被配置为输入模式时，在 CH0IF 标志位已经被置 1 后，捕获事件再次发生时，该标志位可以由硬件置 1。该标志位由软件清 0。<br>0：无捕获溢出中断发生。<br>1：发生了捕获溢出中断。 |
| 8:7 | 保留 | 必须保持复位值。 |
| 6 | TRGIF | 触发中断标志<br>当发生触发事件时，此标志会置 1，此位由软件清 0。当暂停模式使能时，触发输入的任意边沿都可以产生触发事件。否则，其它模式时，仅在触发输入端检测到有效边沿，产生触发事件。<br>0：无触发事件产生。<br>1：触发中断产生。 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | CH3IF | 通道 3 比较/捕获中断标志<br>参见 CH0IF 描述。 |
| 3 | CH2IF | 通道 2 比较/捕获中断标志 |

参见 CH0IF 描述。

| | | |
|---|---|---|
| 2 | CH1IF | 通道 1 比较/捕获中断标志<br>参见 CH0IF 描述。 |
| 1 | CH0IF | 通道 0 比较/捕获中断标志<br>此标志由硬件置 1 软件清 0。当通道 0 在输入模式下时，捕获事件发生时此标志位被置 1；当通道 0 在输出模式下时，此标志位在一个比较事件发生时被置 1。<br>0：无通道 0 中断发生。<br>1：通道 0 中断发生。 |
| 0 | UPIF | 更新中断标志<br>此位在任何更新事件发生时由硬件置 1，软件清 0。<br>0：无更新中断发生。<br>1：发生更新中断。 |

### 软件事件产生寄存器(TIMERx_SWEVG)

地址偏移：0x14
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | TRGG | 保留 | CH3G | CH2G | CH1G | CH0G | UPG |
| | | | | | | | | | w | | w | w | w | w | w |

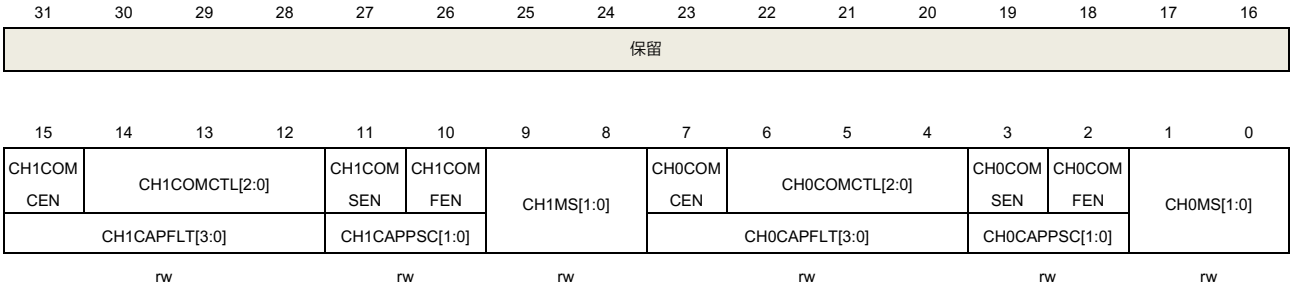
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:7 | 保留 | 必须保持复位值。 |
| 6 | TRGG | 触发事件产生<br>此位由软件置 1，由硬件自动清 0. 当此位被置 1，TIMERx_INTF 寄存器的 TRGIF 标志位被置 1，若开启对应的中断和 DMA，则产生相应的中断和 DMA 传输。<br>0：无触发事件产生。<br>1：产生触发事件。 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | CH3G | 通道 3 捕获或比较事件发生<br>参见 CH0G 描述。 |
| 3 | CH2G | 通道 2 捕获或比较事件发生<br>参见 CH0G 描述。 |
| 2 | CH1G | 通道 1 捕获或比较事件发生<br>参见 CH0G 描述。 |

| | | |
|---|---|---|
| 1 | CH0G | 通道 0 捕获或比较事件发生<br>该位由软件置 1，用于在通道 0 产生一个捕获/比较事件，由硬件自动清 0。当此位被置 1，CH0IF 标志位被置 1，若开启对应的中断和 DMA，则发出相应的中断和 DMA 请求。此外，如果通道 0 配置为输入模式，计数器的当前值被 TIMERx_CH0CV 寄存器捕获，如果 CH0IF 标志位已经为 1，则 CH0OF 标志位被置 1。<br>0：不产生通道 0 捕获或比较事件。<br>1：发生通道 0 捕获或比较事件。 |
| 0 | UPG | 更新事件产生<br>此位由软件置 1，被硬件自动清 0。当此位被置 1，如果选择了中央对齐或向上计数模式，计数器被清 0。否则(向下计数模式)计数器将载入自动重载值，预分频计数器将同时被清除。<br>0：无更新事件产生。<br>1：产生更新事件。 |

### 通道控制寄存器 0(TIMERx_CHCTL0)

地址偏移：0x18
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




**输出比较模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15 | CH1COMCEN | 通道 1 输出比较清 0 使能<br>参见 CH0COMCEN 描述。 |
| 14:12 | CH1COMCTL[2:0] | 通道 1 输出比较模式<br>参见 CH0COMCTL 描述。 |
| 11 | CH1COMSEN | 通道 1 输出比较影子寄存器使能<br>参见 CH0COMSEN 描述。 |
| 10 | CH1COMFEN | 通道 1 输出比较快速使能<br>参见 CH0COMFEN 描述。 |
| 9:8 | CH1MS[1:0] | 通道 1 模式选择<br>这些位定义了通道的方向和输入信号的选择。只有当通道关闭(TIMERx_CHCTL2 寄 |

| | | |
|---|---|---|
| | | 存器的 CH1EN 位被清 0)时这些位才可以写。<br>00：通道 1 配置为输出<br>01：通道 1 配置为输入，IS1 映射在 CI1FE1 上<br>10：通道 1 配置为输入，IS1 映射在 CI0FE1 上<br>11：通道 1 配置为输入，IS1 映射在 ITS 上<br>注意：当 CH1MS[1:0]=11 时，需要通过 TRGS 位(位于 TIMERx_SMCFG 寄存器)选择内部触发输入。 |
| 7 | CH0COMCEN | 通道 0 输出比较清 0 使能<br>当此位被置 1，当检测到 ETIFP 信号输入高电平时，O0CPRE 参考信号被清 0<br>0：禁止通道 0 输出比较清零<br>1：使能通道 0 输出比较清零 |
| 6:4 | CH0COMCTL[2:0] | 通道 0 输出比较模式<br>此位定义了输出准备信号 O0CPRE 的输出比较模式，而 O0CPRE 决定了 CH0_O、CH0_ON 的值。另外，O0CPRE 高电平有效，而 CH0_O、CH0_ON 通道的极性取决于 CH0P、CH0NP 位。<br>000：时基。输出比较寄存器 TIMERx_CH0CV 与计数器 TIMERx_CNT 间的比较对 O0CPRE 不起作用<br>001：匹配时设置为高。当计数器的值与捕获/比较值寄存器 TIMERx_CH0CV 相同时，强制 O0CPRE 为高。<br>010：匹配时设置为低。当计数器的值与捕获/比较值寄存器 TIMERx_CH0CV 相同时，强制 O0CPRE 为低。<br>011：匹配时翻转。当计数器的值与捕获/比较值寄存器 TIMERx_CH0CV 相同时，强制 O0CPRE 翻转。<br>100：强制为低。强制 O0CPRE 为低电平<br>101：强制为高。强制 O0CPRE 为高电平<br>110：PWM 模式 0。在向上计数时，一旦计数器值小于 TIMERx_CH0CV 时，O0CPRE 为高电平，否则为低电平。在向下计数时，一旦计数器的值大于 TIMERx_CH0CV 时，O0CPRE 为低电平，否则为高电平。<br>111：PWM 模式 1。在向上计数时，一旦计数器值小于 TIMERx_CH0CV 时，O0CPRE 为低电平，否则为高电平。在向下计数时，一旦计数器的值大于 TIMERx_CH0CV 时，O0CPRE 为高电平，否则为低电平。<br>如果配置在 PWM 模式下，只有当输出比较模式从时基模式变为 PWM 模式或者比较结果改变时，O0CPRE 电平才改变。<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 且 CH0MS =00(比较模式)时此位不能被改变。 |
| 3 | CH0COMSEN | 通道 0 输出比较影子寄存器使能<br>当此位被置 1，TIMERx_CH0CV 寄存器的影子寄存器被使能，影子寄存器在每次更新事件时都会被更新。<br>0：禁止通道 0 输出/比较影子寄存器<br>1：使能通道 0 输出/比较影子寄存器<br>仅在单脉冲模式下(SPM =1)，可以在未确认影子寄存器的情况下使用 PWM 模式<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 且 CH0MS =00 时此位不能被改变。 |

| | | |
|---|---|---|
| 2 | CH0COMFEN | 通道 0 输出比较快速使能<br>0：通道 0 输出比较快速禁能。仅比较结果输入作为有效边沿时会产生比较匹配，并将 CH0_O 设置为比较电平，激活 CH0_O 输出的最小延时为 5 个时钟周期。<br>1：通道 0 输出比较快速使能。触发输入信号的有效边沿和比较结果都会产生比较匹配，并将 CH0_O 设置为比较电平。当触发信号的输入作为有效边沿时，激活 CH0_O 输出的最小延时为 3 个时钟周期。 |
| 1:0 | CH0MS[1:0] | 通道 0 I/O 模式选择<br>这些位定义了通道的工作模式和输入信号的选择。只有当通道关闭(TIMERx_CHCTL2 寄存器的 CH0EN 位被清 0)时这些位才可写。<br>00：通道 0 配置为输出<br>01：通道 0 配置为输入，IS0 映射在 CI0FE0 上<br>10：通道 0 配置为输入，IS0 映射在 CI1FE0 上<br>11：通道 0 配置为输入，IS0 映射在 ITS 上<br>注意：当 CH0MS[1:0]=11 时，需要通过 TRGS 位(位于 TIMERx_SMCFG 寄存器)选择内部触发输入 |

**输入捕获模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:12 | CH1CAPFLT[3:0] | 通道 1 输入捕获滤波控制<br>参见 CH0CAPFLT 描述。 |
| 11:10 | CH1CAPPSC[1:0] | 通道 1 输入捕获预分频器<br>参见 CH0CAPPSC 描述。 |
| 9:8 | CH1MS[1:0] | 通道 1 模式选择<br>与输出模式相同。 |
| 7:4 | CH0CAPFLT[3:0] | 通道 0 输入捕获滤波控制<br>CI0 输入信号可以通过数字滤波器进行滤波，该位域配置滤波参数。<br>数字滤波器的基本原理：根据 $f_{SAMP}$ 对 CI0 输入信号进行连续采样，并记录信号相同电平的次数。达到该位配置的滤波参数后，认为是有效电平。<br>滤波器参数配置如下： |

| CH0CAPFLT [3:0] | 采样次数 | $f_{SAMP}$ |
|---|---|---|
| 4'b0000 | 无滤波器 | |
| 4'b0001 | 2 | $f_{CK_TIMER}$ |
| 4'b0010 | 4 | |
| 4'b0011 | 8 | |
| 4'b0100 | 6 | $f_{DTS}/2$ |
| 4'b0101 | 8 | |
| 4'b0110 | 6 | $f_{DTS}/4$ |
| 4'b0111 | 8 | |
| 4'b1000 | 6 | $f_{DTS}/8$ |
| 4'b1001 | 8 | |

| | | | | |
|---|---|---|---|---|
| | | 4'b1010 | 5 | $f_{DTS}$/16 |
| | | 4'b1011 | 6 | |
| | | 4'b1100 | 8 | |
| | | 4'b1101 | 5 | $f_{DTS}$/32 |
| | | 4'b1110 | 6 | |
| | | 4'b1111 | 8 | |

3:2 CH0CAPPSC[1:0] 通道 0 输入捕获预分频器

这 2 位定义了通道 0 输入的预分频系数。当 TIMERx_CHCTL2 寄存器中的 CH0EN =0 时，则预分频器复位。

00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获。

01：每 2 个事件触发一次捕获。

10：每 4 个事件触发一次捕获。

11：每 8 个事件触发一次捕获。

1:0 CH0MS[1:0] 通道 0 模式选择

与输出比较模式相同。

### 通道控制寄存器 1(TIMERx_CHCTL1)

地址偏移：0x1C

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH3COMCEN | CH3COMCTL[2:0] | | | CH3COMSEN | CH3COMFEN | CH3MS[1:0] | | CH2COMCEN | CH2COMCTL[2:0] | | | CH2COMSEN | CH2COMFEN | CH2MS[1:0] | |
| CH3CAPFLT[3:0] | | | | CH3CAPPSC[1:0] | | | | CH2CAPFLT[3:0] | | | | CH2CAPPSC[1:0] | | | |
| rw | | | | rw | | rw | | rw | | | | rw | | rw | |

**输出比较模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15 | CH3COMCEN | 通道 3 输出比较清 0 使能<br>参见 CH0COMCEN 描述。 |
| 14:12 | CH3COMCTL[2:0] | 通道 3 输出比较模式<br>参见 CH0COMCTL 描述 |
| 11 | CH3COMSEN | 通道 3 输出比较影子寄存器使能<br>参见 CH0COMSEN 描述。 |
| 10 | CH3COMFEN | 通道 3 输出比较快速使能<br>参见 CH0COMFEN 描述。 |

| | | |
|---|---|---|
| 9:8 | CH3MS[1:0] | 通道 3 模式选择<br>这些位定义了通道的方向和输入信号的选择。只有当通道关闭(TIMERx_CHCTL2 寄存器的 CH3EN 位被清 0)时这些位才可以写。<br>00：通道 3 配置为输出<br>01：通道 3 配置为输入，IS3 映射在 CI3FE3 上<br>10：通道 3 配置为输入，IS3 映射在 CI2FE3 上<br>11：通道 3 配置为输入，IS3 映射在 ITS 上<br>注意：当 CH3MS[1:0]=11 时，需要通过 TRGS 位(位于 TIMERx_SMCFG 寄存器)选择内部触发输入 |
| 7 | CH2COMCEN | 通道 2 输出比较清 0 使能<br>当此位被置 1，当检测到 ETIFP 输入高电平时，O2CPRE 参考信号被清 0<br>0：使能通道 2 输出比较清零<br>1：禁止通道 2 输出比较清零 |
| 6:4 | CH2COMCTL[2:0] | 通道 2 输出比较模式<br>此位定义了输出准备信号 O2CPRE 的输出比较模式，而 O2CPRE 决定了 CH2_O、CH2_ON 的值。另外，O2CPRE 高电平有效，而 CH2_O、CH2_ON 通道的极性取决于 CH2P、CH2NP 位。<br>000：时基。输出比较寄存器 TIMERx_CH2CV 与计数器 TIMERx_CNT 间的比较对 O2CPRE 不起作用<br>001：匹配时设置为高。当计数器的值与捕获/比较值寄存器 TIMERx_CH2CV 相同时，强制 O2CPRE 为高。<br>010：匹配时设置为低。当计数器的值与捕获/比较值寄存器 TIMERx_CH2CV 相同时，强制 O2CPRE 为低。<br>011：匹配时翻转。当计数器的值与捕获/比较值寄存器 TIMERx_CH2CV 相同时，强制 O2CPRE 翻转。<br>100：强制为低。强制 O2CPRE 为低电平<br>101：强制为高。强制 O2CPRE 为高电平<br>110：PWM 模式 0。在向上计数时，一旦计数器值小于 TIMERx_CH2CV 时，O2CPRE 为高电平，否则为低电平。在向下计数时，一旦计数器的值大于 TIMERx_CH2CV 时，O2CPRE 为低电平，否则为高电平。<br>111：PWM 模式 1。在向上计数时，一旦计数器值小于 TIMERx_CH2CV 时，O2CPRE 为低电平，否则为高电平。在向下计数时，一旦计数器的值大于 TIMERx_CH2CV 时，O2CPRE 为高电平，否则为低电平。<br>如果配置在 PWM 模式下，只有当输出比较模式从时基模式变为 PWM 模式或者比较结果改变时，O2CPRE 电平才改变。<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 且 CH2MS =00(比较模式)时此位不能被改变。 |
| 3 | CH2COMSEN | 通道 0 输出比较影子寄存器使能<br>当此位被置 1，TIMERx_CH2CV 寄存器的影子寄存器被使能，影子寄存器在每次更新事件时都会被更新。<br>0：禁止通道 2 输出/比较影子寄存器<br>1：使能通道 2 输出/比较影子寄存器<br>仅在单脉冲模式下(SPM =1)，可以在未确认影子寄存器情况下使用 PWM 模式 |

当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 且 CH2MS =00 时此位不能被改变。

| | | |
|---|---|---|
| 2 | CH2COMFEN | 通道 2 输出比较快速使能<br>0：通道 2 输出比较快速禁能。仅比较结果输入作为有效边沿时会产生比较匹配，并将 CH2_O 设置为比较电平，激活 CH2_O 输出的最小延时为 5 个时钟周期。<br>1：通道 2 输出比较快速使能。触发输入信号的有效边沿和比较结果都会产生比较匹配，并将 CH2_O 设置为比较电平。当触发信号的输入作为有效边沿时，激活 CH2_O 输出的最小延时为 3 个时钟周期。 |
| 1:0 | CH2MS[1:0] | 通道 2 I/O 模式选择<br>这些位定义了通道的工作模式和输入信号的选择。只有当通道关闭(TIMERx_CHCTL2 寄存器的 CH2EN 位被清 0)时这些位才可写。<br>00：通道 2 配置为输出<br>01：通道 2 配置为输入，IS2 映射在 CI2FE2 上<br>10：通道 2 配置为输入，IS2 映射在 CI3FE2 上<br>11：通道 2 配置为输入，IS2 映射在 ITS 上.<br>注意：当 CH2MS[1:0]=11 时，需要通过 TRGS 位(位于 TIMERx_SMCFG 寄存器)选择内部触发输入 |

**输入捕获模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:12 | CH3CAPFLT[3:0] | 通道 3 输入捕获滤波控制<br>参见 CH0CAPFLT 描述。 |
| 11:10 | CH3CAPPSC[1:0] | 通道 3 输入捕获预分频器<br>参见 CH0CAPPSC 描述。 |
| 9:8 | CH3MS[1:0] | 通道 3 模式选择<br>与输出模式相同。 |
| 7:4 | CH2CAPFLT[3:0] | 通道 2 输入捕获滤波控制<br>CI2 输入信号可以通过数字滤波器进行滤波，该位域配置滤波参数。<br>数字滤波器的基本原理：根据 $f_{SAMP}$ 对 CI2 输入信号进行连续采样，并记录信号相同电平的次数。达到该位配置的滤波参数后，认为是有效电平。<br>滤波器参数配置如下： |

| CH2CAPFLT [3:0] | 采样次数 | $f_{SAMP}$ |
|---|---|---|
| 4'b0000 | 无滤波器 | |
| 4'b0001 | 2 | $f_{CK_TIMER}$ |
| 4'b0010 | 4 | |
| 4'b0011 | 8 | |
| 4'b0100 | 6 | $f_{DTS}/2$ |
| 4'b0101 | 8 | |
| 4'b0110 | 6 | $f_{DTS}/4$ |
| 4'b0111 | 8 | |

| | | |
|---|---|---|
| 4'b1000 | 6 | $f_{DTS}$/8 |
| 4'b1001 | 8 | |
| 4'b1010 | 5 | $f_{DTS}$/16 |
| 4'b1011 | 6 | |
| 4'b1100 | 8 | |
| 4'b1101 | 5 | $f_{DTS}$/32 |
| 4'b1110 | 6 | |
| 4'b1111 | 8 | |

3:2 CH2CAPPSC[1:0] 通道 2 输入捕获预分频器

这 2 位定义了通道 2 输入的预分频系数。当 TIMERx_CHCTL2 寄存器中的 CH2EN =0 时，则预分频器复位。

00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获。

01：每 2 个事件触发一次捕获。

10：每 4 个事件触发一次捕获。

11：每 8 个事件触发一次捕获。

1:0 CH2MS[1:0] 通道 2 模式选择

与输出比较模式相同。

### 通道控制寄存器 2(TIMERx_CHCTL2)

地址偏移：0x20

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | C4P | CH3EN | CH2NP | 保留 | CH2P | CH2EN | CH1NP | 保留 | CH1P | CH1EN | CH0NP | 保留 | CH0P | CH0EN |
| | | rw | rw | rw | | rw | rw | rw | | rw | rw | rw | | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值。 |
| 13 | CH3P | 通道 3 极性<br>参考 CH0P 描述。 |
| 12 | CH3EN | 通道 3 使能<br>参考 CH0EN 描述。 |
| 11 | CH2NP | 通道 2 互补输出极性<br>参考 CH0NP 描述。 |
| 10 | 保留 | 必须保持复位值。 |
| 9 | CH2P | 通道 2 极性 |

| | | |
|---|---|---|
| | | 参考 CH0P 描述。 |
| 8 | CH2EN | 通道 2 使能<br>参考 CH0EN 描述。 |
| 7 | CH1NP | 通道 1 互补输出极性<br>参考 CH0NP 描述。 |
| 6 | 保留 | 必须保持复位值。 |
| 5 | CH1P | 通道 1 极性<br>参考 CH0P 描述。 |
| 4 | CH1EN | 通道 1 使能<br>参考 CH0EN 描述。 |
| 3 | CH0NP | 通道 0 互补输出极性<br>当通道 0 配置为输出模式，该位保持 0。<br>当通道 0 配置为输入模式时，此位和 CH0P 联合使用，作为输入信号 CI0 的极性选择控制信号。<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 或 10 时此位不能被更改。 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | CH0P | 通道 0 极性<br>当通道 0 配置为输出模式时，此位定义了输出信号极性。<br>0：通道0高电平为有效电平<br>1：通道0低电平为有效电平<br>当通道 0 配置为输入模式时，此位定义了 CI0 信号极性。<br>[CH0NP, CH0P] 将选择 CI0FE0 或者 CI1FE0 的有效边沿或者捕获极性。<br>[CH0NP==0, CH0P==0]：把 CIxFE0 的上升沿作为捕获或者从模式下触发的有效信号，并且 CIxFE0 不会被翻转。<br>[CH0NP==0, CH0P==1]：把 CIxFE0 的下降沿作为捕获或者从模式下触发的有效信号，并且 CIxFE0 会被翻转。<br>[CH0NP==1, CH0P==0]：保留。<br>[CH0NP==1, CH0P==1]：把 CIxFE0 的上升沿和下降沿都作为捕获或者从模式下触发的有效信号，并且 CIxFE0 不会被翻转。<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 或 10 时此位不能被更改。 |
| 0 | CH0EN | 通道 0 捕获/比较使能<br>当通道 0 配置为输出模式时，将此位置 1 使能 CH0_O 信号有效。当通道 0 配置为输入模式时，将此位置 1 使能通道 0 上的捕获事件。<br>0：禁止通道 0。<br>1：使能通道 0。 |

### 计数器寄存器(TIMERx_CNT)(x=1,4)

地址偏移：0x24

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | CNT[31:0] | 这些位是当前的计数值。写操作能改变计数器值。 |

### 计数器寄存器(TIMERx_CNT)(x=2,3)

地址偏移：0x24
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CNT[15:0] | 这些位是当前的计数值。写操作能改变计数器值。 |

### 预分频寄存器(TIMERx_PSC)

地址偏移： 0x28
复位值： 0x0000 0000

该寄存器只能按字(32位)访问

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

| 位/位域 | 名称 | 描述 |
|---|---|---|

| | | |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | PSC[15:0] | 计数器时钟预分频值<br>计数器时钟等于 TIMER_CK 时钟除以(PSC+1)，每次当更新事件产生时，PSC 的值被装入到对应的影子寄存器。 |

### 计数器自动重载寄存器(TIMERx_CAR)(x=1,4)

地址偏移：0x2C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | CARL[31:0] | 计数器自动重载值<br>这些位定义了计数器的自动重载值。 |

### 计数器自动重载寄存器(TIMERx_CAR)(x=2,3)

地址偏移：0x2C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CARL[15:0] | 计数器自动重载值<br>这些位定义了计数器的自动重载值。 |

### 通道 0 捕获/比较值寄存器(TIMERx_CH0CV)(x=1,4)

地址偏移：0x34

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH0VAL[31:16] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CH0VAL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | CH0VAL[31:0] | 通道 0 的捕获或比较值<br>当通道 0 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。<br>当通道 0 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。 |

### 通道 0 捕获/比较值寄存器(TIMERx_CH0CV)(x=2,3)

地址偏移：0x34
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CH0VAL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CH0VAL[15:0] | 通道 0 的捕获或比较值<br>当通道 0 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。<br>当通道 0 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。 |

### 通道 1 捕获/比较值寄存器(TIMERx_CH1CV)(x=1,4)

地址偏移：0x38
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH1VAL[31:16] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CH1VAL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | CH1VAL[31:0] | 通道 1 的捕获或比较值<br>当通道 1 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。<br>当通道 1 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。 |

### 通道 1 捕获/比较值寄存器(TIMERx_CH1CV)(x=2,3)

地址偏移：0x38
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CH1VAL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CH1VAL[15:0] | 通道 1 的捕获或比较值<br>当通道 1 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。<br>当通道 1 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。 |

### 通道 2 捕获/比较值寄存器(TIMERx_CH2CV)(x=1,4)

地址偏移：0x3C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH2VAL[31:16] | | | | | | | | | | | | | | | |

rw

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH2VAL[15:0] | | | | | | | | | | | | | | | |

rw

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | CH2VAL[31:0] | 通道 2 的捕获或比较值<br>当通道 2 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。<br>当通道 2 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。 |

### 通道 2 捕获/比较值寄存器(TIMERx_CH2CV)(x=2,3)

地址偏移：0x3C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH2VAL[15:0] | | | | | | | | | | | | | | | |

rw

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CH2VAL[15:0] | 通道 2 的捕获或比较值<br>当通道 2 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。<br>当通道 2 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。 |

### 通道 3 捕获/比较值寄存器(TIMERx_CH3CV)(x=1,4)

地址偏移：0x40
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH3VAL[31:16] | | | | | | | | | | | | | | | |

rw

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

| CH3VAL[15:0] |
|---|
| rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | CH3VAL[31:0] | 通道 3 的捕获或比较值<br>当通道 3 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。<br>当通道 3 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。 |

### 通道 3 捕获/比较值寄存器(TIMERx_CH3CV)(x=2,3)

地址偏移：0x40
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH3VAL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CH3VAL[15:0] | 通道 3 的捕获或比较值<br>当通道 3 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。<br>当通道 3 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。 |

### DMA 配置寄存器(TIMERx_DMACFG)

地址偏移：0x48
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值。 |
| 12:8 | DMATC [4:0] | DMA 传输计数<br>该位域定义了 DMA 访问(读写)TIMERx_DMATB 寄存器的数量 n，n = (DMATC [4:0] +1). DMATC [4:0] 从 5'b0_0000 到 5'b1_0001. |
| 7:5 | 保留 | 必须保持复位值。 |
| 4:0 | DMATA [4:0] | DMA 传输起始地址<br>该位域定义了 DMA 访问 TIMERx_DMATB 寄存器的第一个地址。当通过 TIMERx_DMA 第一次访问时，访问的就是该位域指定的地址。第二次访问 TIMERx_DMATB 时，将访问起始地址+0x4。 |

### DMA 发送缓冲区寄存器(TIMERx_DMATB)

地址偏移：0x4C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | DMATB [15:0] | DMA 发送缓冲<br>对这个寄存器的读或写，(起始地址+传输次数*4)地址范围内的寄存器会被访问<br>传输次数由硬件计算，范围为 0 到 DMATC。 |

### 输入重映射寄存器(TIMERx_IRMP)(x=1)

地址偏移：0x50
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

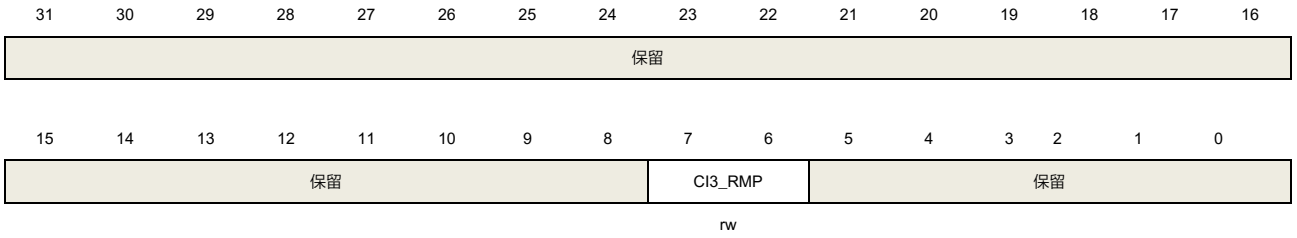
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | ITI1_RMP | | 保留 | | | | | | | | | |
| | | | | rw | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11:10 | ITI1_RMP | 内部触发输入 1 重映射<br>00:TIMER7_TRGO<br>01:Ethernet PTP<br>10:USB FS SOF<br>11:USB HS SOF |
| 9:0 | 保留 | 必须保持复位值。 |

### 输入重映射寄存器(TIMERx_IRMP)(x=4)

地址偏移：0x50
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7:6 | CI3_RMP | 通道 3 输入重映射<br>00：连接到 GPIO 引脚. 参考 GPIO 重映射表<br>01：IRC32K<br>10：LXTAL<br>11:RTC wakeup interrupt |
| 5:0 | 保留 | 必须保持复位值。 |

### 配置寄存器(TIMERx_CFG)

地址偏移：0xFC
复位值：0x0000 0000

该寄存器只能按字(32位)访问。



| 保留 | CHVSEL | 保留 |
|---|---|---|
| | rw | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |
| 1 | CHVSEL | 写捕获比较寄存器选择位<br>此位由软件写 1 或清 0。<br>1：当写入捕获比较寄存器的值与寄存器当前值相等时，写入操作无效<br>0：无影响 |
| 0 | 保留 | 必须保持复位值。 |

# 18.3. 通用定时器 L1(TIMERx, x=8,11)

## 18.3.1. 简介

通用定时器 L1(Timer8，11)是两通道定时器，支持输入捕获和输出比较，可以产生 PEM 信号控制电机和电源管理。通用定时器 L1 含有一个 16 位无符号计数器。

高级定时器是可编程的，可以被用来计数，其外部事件可以驱动其他定时器

定时器和定时器之间是相互独立，但是他们可以被同步在一起形成一个更大的定时器，这些定时器的计数器一致地增加。

## 18.3.2. 主要特性

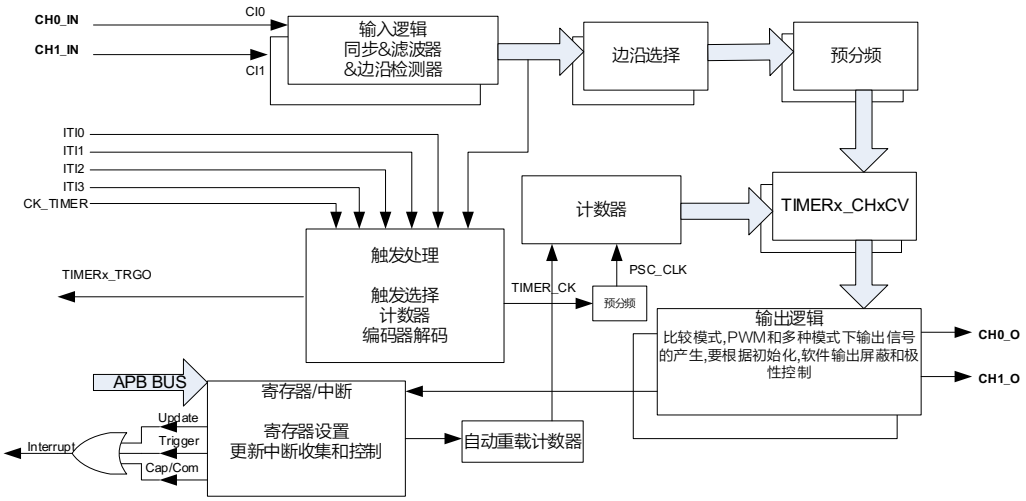
- 总通道数：2；
- 计数器宽度：16位；
- 时钟源可选：内部时钟，内部触发，外部输入，外部触发；
- 计数模式：向上计数；
- 可编程的预分频器：16位，运行时可以被改变；
- 每个通道可配置：输入捕获模式，输出比较模式，可编程的PWM模式，单脉冲模式；
- 自动重装载功能；
- 中断输出：更新事件，触发事件，比较/捕获事件和中止事件；
- 多个定时器的菊链使得一个定时器可以同时启动多个定时器；
- 定时器的同步允许被选择的定时器在同一个时钟周期开始计数；
- 定时器主-从管理。

## 18.3.3. 功能描述

### 结构框图

*图 18-43. 通用定时器 L1 结构框图*提供了通用定时器 L1 的内部配置细节。

**图 18-43. 通用定时器 L1 结构框图**




## 时钟源配置

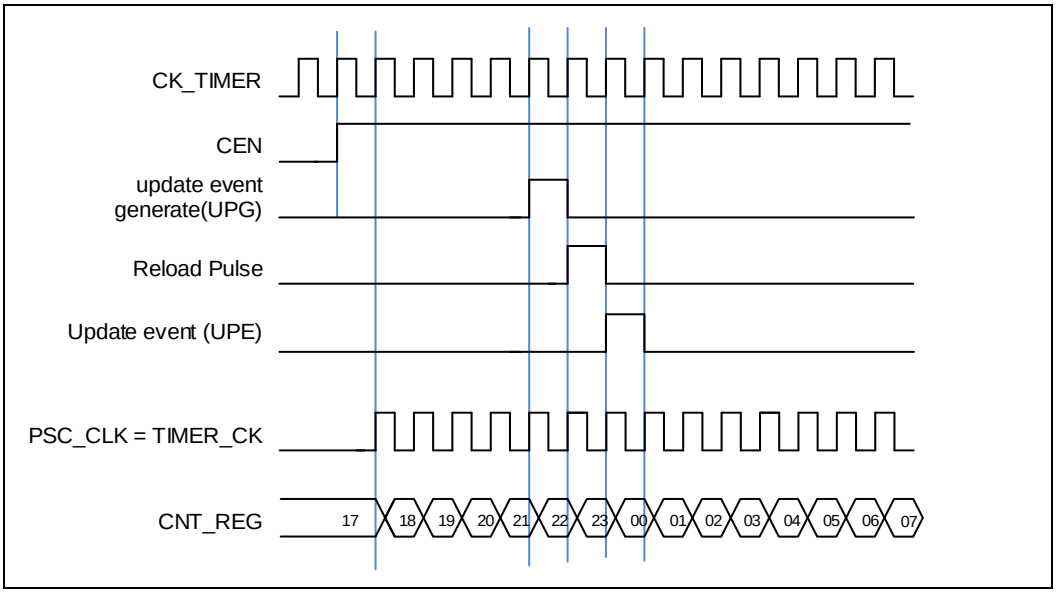
通用定时器 L1 可以由内部时钟源 CK_TIMER 或者由 SMC(TIMERx_SMCFG 寄存器位[2:0])控制的复用时钟源驱动。

■ SMC[2:0]==3'b000，定时器选择内部时钟源(连接到RCU模块的CK_TIMER)

如果 SMC[2:0]==3'b000，默认用来驱动计数器预分频器的是内部时钟源 CK_TIMER。当 CEN 置位，CK_TIMER 经过预分频器(预分频值由 TIMERx_PSC 寄存器确定)产生 PSC_CLK。

如果将 TIMERx_SMCFG 寄存器的 SMC[2:0]设置为 0x1、0x2、0x3 和 0x7，预分频器被其他时钟源(由 TIMERx_SMCFG 寄存器的 TRGS [2:0]区域选择)驱动，在下文说明。当 SMC 位被设置为 0x4、0x5 和 0x6，计数器预分频器时钟源由内部时钟 CK_TIMER 驱动。

**图 18-44. 内部时钟分频为 1 时，计数器的时序图**




■ SMC[2:0]==3'b111(外部时钟模式0)，定时器选择外部输入引脚作为时钟源

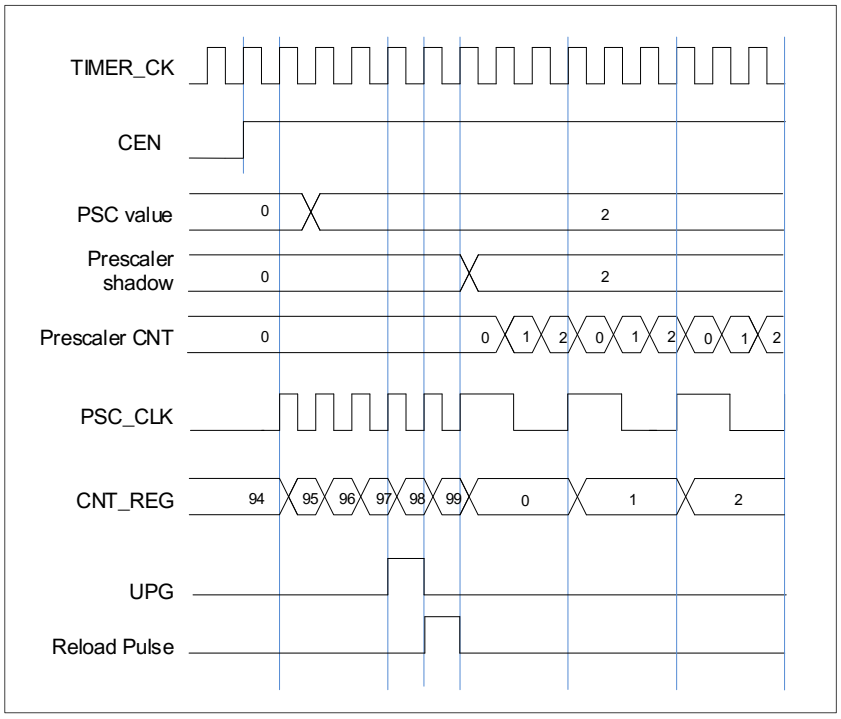
计数器预分频器可以在 TIMERx_CI0/ TIMERx_CI1 引脚的每个上升沿或下降沿计数。这种模式可以通过设置 SMC [2:0]为 0x7 同时设置 TRGS [2:0]为 0x4，0x5 或 0x6 来选择。CIx 是 TIMERx_CIx 通过数字滤波器采样后的信号。

计数器预分频器也可以在内部触发信号 ITI0/1/2/3 的上升沿计数。这种模式可以通过设置 SMC [2:0]为 0x7 同时设置 TRGS [2:0]为 0x0, 0x1, 0x2 或者 0x3。

### 时钟预分频器

预分频器可以将定时器的时钟(TIMER_CK)频率按 1 到 65536 之间的任意值分频，分频后的时钟 PSC_CLK 驱动计数器计数。分频系数受预分频寄存器 TIMERx_PSC 控制，这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

**图 18-45. 当 PSC 数值从 0 变到 2 时，计数器的时序图**




### 计数器向上计数模式

在这种模式，计数器的计数方向是向上计数。计数器从 0 开始向上连续计数到自动加载值(定义在 TIMERx_CAR 寄存器中)，一旦计数器计数到自动加载值，会重新从 0 开始向上计数并产生上溢事件。在向上计数模式中，TIMERx_CTL0 寄存器中的计数方向控制位 DIR 应该被设置成 0。

当通过 TIMERx_SWEVG 寄存器的 UPG 位置 1 来设置更新事件时，计数值会被清 0，并产生更新事件。

如果 TIMERx_CTL0 寄存器的 UPDIS 置 1，则禁止更新事件。

当发生更新事件时，所有影子寄存器(计数器自动重载寄存器，预分频寄存器)都将被更新。

下面一些图给出了一些例子，当 TIMERx_CAR=0x99 时，计数器在不同预分频因子下的行为。

**图 18-46. 向上计数时序图，PSC=0/2**

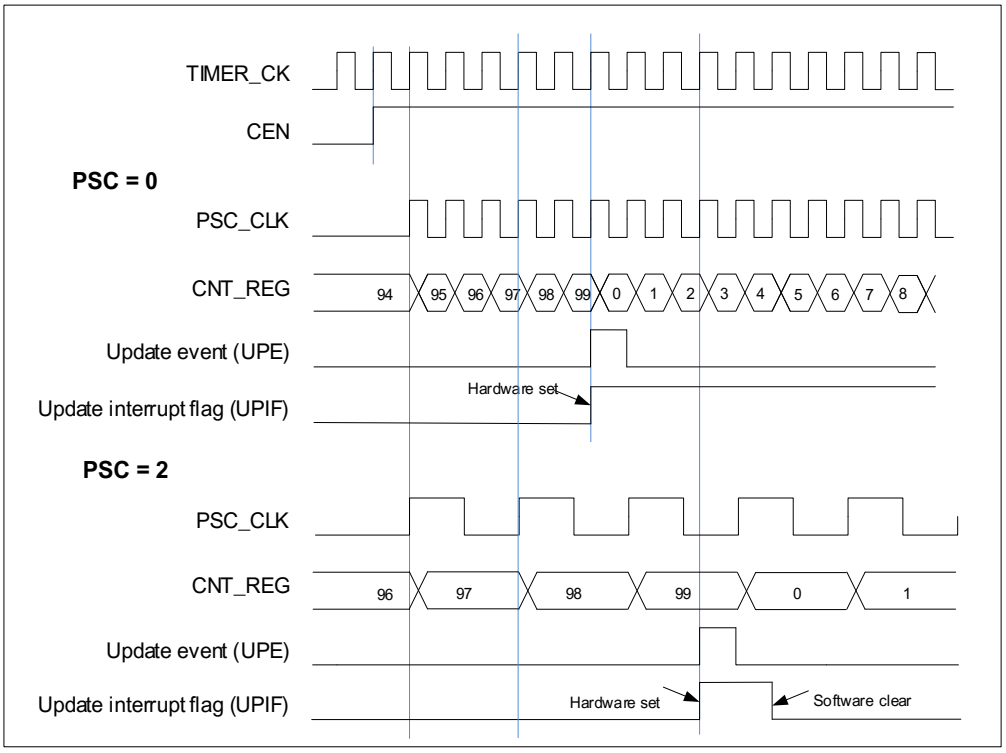
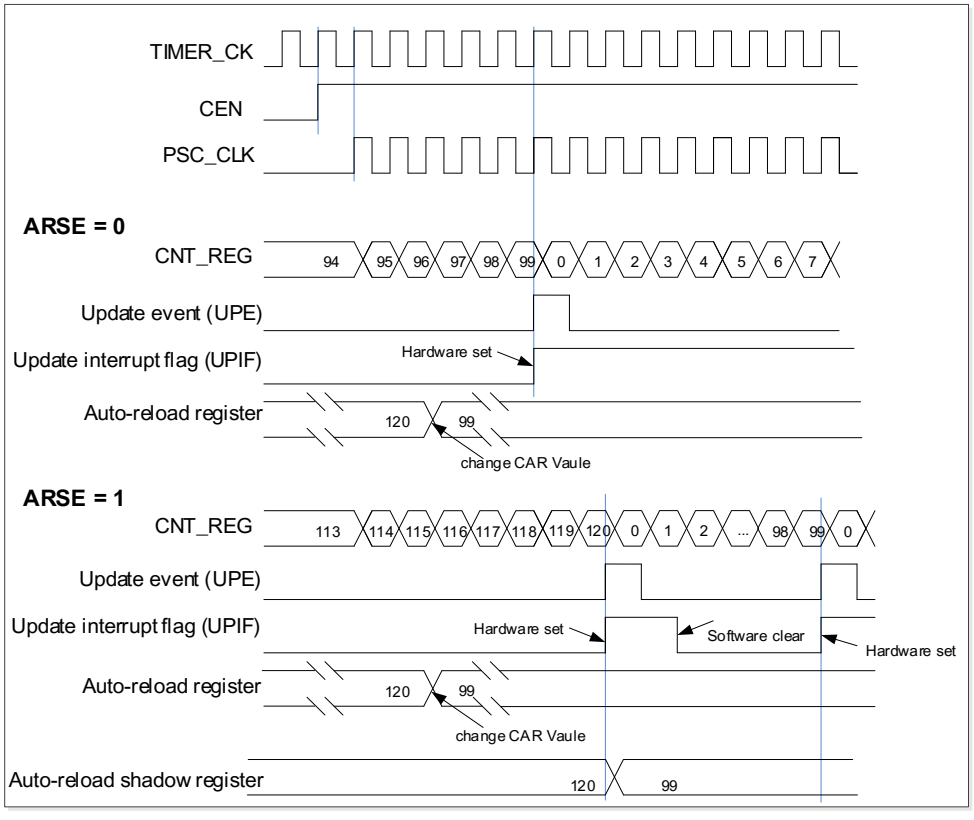



**图 18-47. 向上计数时序图，在运行时改变 TIMERx_CAR 寄存器的值**



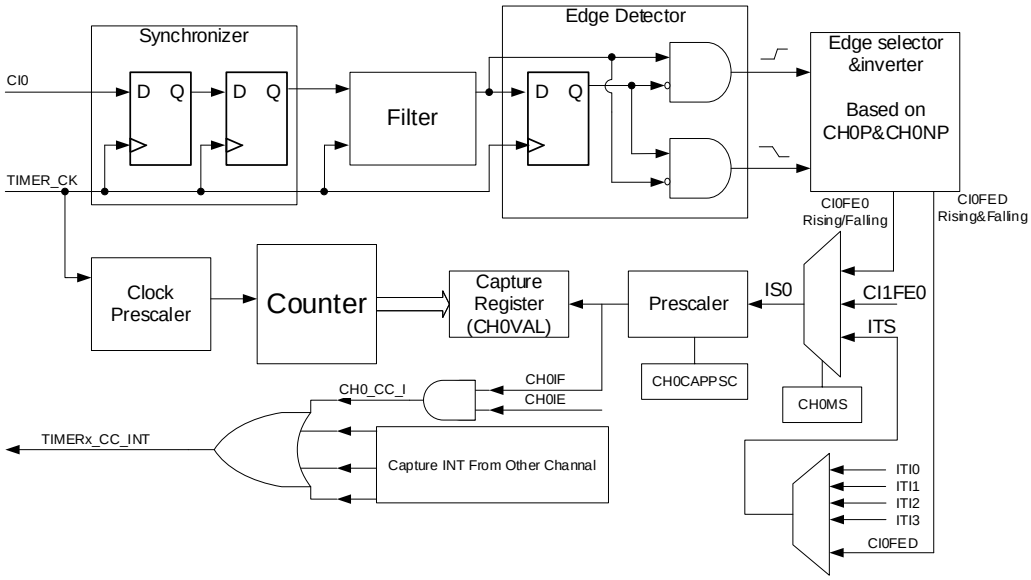
### 输入捕获和输出比较通道

通用定时器 L1 拥有两个个独立的通道用于捕获输入或比较输出是否匹配。每个通道都围绕一个通道捕获比较寄存器建立，包括一个输入级，通道控制器和输出级。

■ 通道输入捕获功能

捕获模式允许通道测量一个波形时序，频率，周期，占空比等。输入级包括一个数字滤波器，一个通道极性选择，边沿检测和一个通道预分频器。如果在输入引脚上出现被选择的边沿，TIMERx_CHxCV 寄存器会捕获计数器当前的值，同时 CHxIF 位被置 1，如果 CHxIE = 1 则产生通道中断。

**图 18-48. 通道输入捕获原理**




通道输入信号 CIx 先被 TIMER_CK 信号同步，然后经过数字滤波器采样，产生一个被滤波后的信号。通过边沿检测器，可以选择检测上升沿或者下降沿。通过配置 CHxP 选择使用上升沿或者下降沿。配置 CHxMS.，可以选择其他通道的输入信号，内部触发信号。配置 IC 预分频器，使得若干个输入事件后才产生一个有效的捕获事件。捕获事件发生，CxCV 存储计数器的值。

**第一步：** 滤波器配置(TIMERx_CHCTL0寄存器中CHxCAPFLT)：
根据输入信号和请求信号的质量，配置相应的CHxCAPFLT。

**第二步：** 边沿选择(TIMERx_CHCTL2寄存器中CHxP/CHxNP)：
配置CHxP/CHxNP选择上升沿或者下降沿。

**第三步：** 捕获源选择(TIMERx_CHCTL0寄存器中CHxMS)：
一旦通过配置CHxMS选择输入捕获源，必须确保通道配置在输入模式(CHxMS!=0x0)，而且TIMERx_CxCV寄存器不能再被写。

**第四步：** 中断使能(TIMERx_DMAINTEN寄存器中CHxIE和CHxDEN)：
使能相应中断，可以获得中断和DMA请求。

**第五步：** 捕获使能(TIMERx_CHCTL2寄存器中CHxEN)。

**结果：**当期望的输入信号发生时，TIMERx_CHxCV被设置成当前计数器的值，CHxIF为置1。如果CHxIF位已经为1，则CHxOF位置1。根据TIMERx_DMAINTEN寄存器中CHxIE和CHxDEN的配置，相应的中断和DMA请求会被提出。

**直接产生：**软件设置CHxG位，会直接产生中断和DMA请求。

通道输入捕获功能也可用来测量 TIMERx_CHx 引脚上信号的脉冲波宽度。例如，一个 PWM 波连接到 CI0。配置 TIMERx_CHCTL0 寄存器中 CH0MS 为 2'b01，选择通道 0 的捕获信号为 CI0 并设置上升沿捕获。配置 TIMERx_CHCTL0 寄存器中 CH1MS 为 2'b10，选择通道 1 捕获信号为 CI0 并设置下降沿捕获。计数器配置为复位模式，在通道 0 的上升沿复位。TIMERX_CH0CV 寄存器测量 PWM 的周期值，TIMERx_CH1CV 寄存器测量 PWM 占空比值。

■ 通道输出比较功能

在输出比较模式，TIMERx 可以产生时控脉冲，其位置，极性，持续时间和频率都是可编程的。当一个输出通道的 CxCV 寄存器与计数器的值匹配时，根据 CHxCOMCTL 的配置，这个通道的输出可以被置高电平，被置低电平或者反转。当计数器的值与 CxCV 寄存器的值匹配时，CHxIF 位被置 1，如果 CHxIE = 1 则会产生中断，如果 CxCDE=1 则会产生 DMA 请求。

配置步骤如下：

**第一步：**时钟配置：
配置定时器时钟源，预分频器等。

**第二步：**比较模式配置：
设置CHxCOMSEN位来配置输出比较影子寄存器；
设置CHxCOMCTL位来配置输出模式(置高电平/置低电平/反转)；
设置CHxP/CHxNP位来选择有效电平的极性；
设置CHxEN使能输出。

**第三步：**通过CHxIE/CxCDE位配置中断/DMA请求使能。

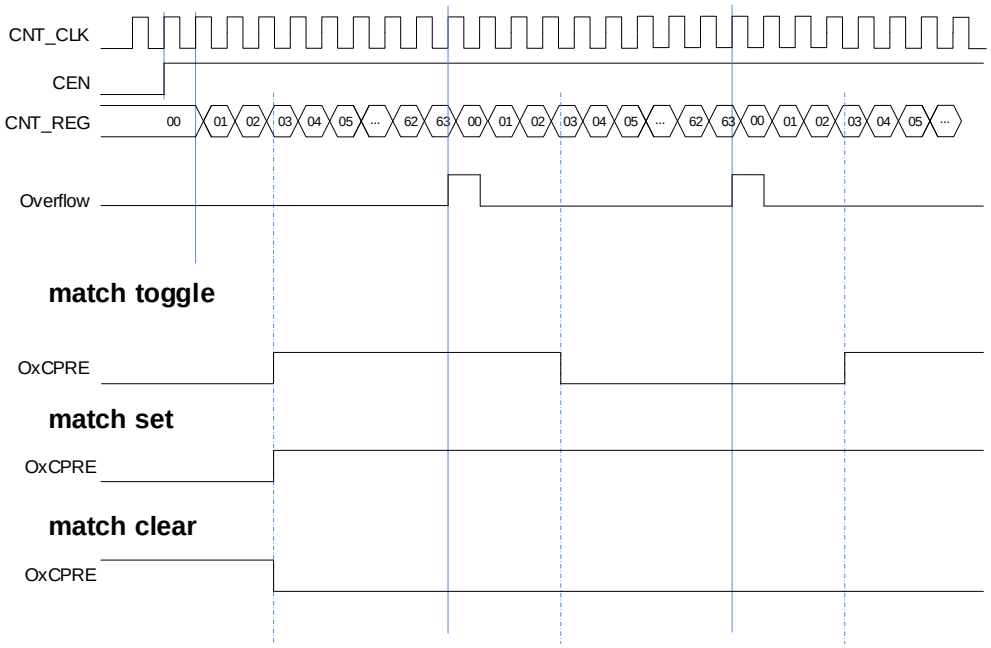
**第四步：**通过TIMERx_CAR寄存器和TIMERx_CHxCV寄存器配置输出比较时基：
CxCV可以在运行时根据你所期望的波形而改变。

**第五步：**设置CEN位使能定时器。

*图 18-49. 三种输出比较模式*显示了三种比较输出模式：反转/置高电平/置低电平，CAR=0x63,

CxCV=0x3。

**图 18-49. 三种输出比较模式**




## 输出 PWM 功能

在 PWM 输出模式下(PWM 模式 0 是配置 CHxCOMCTL 为 3'b110，PWM 模式 1 是配置 CHxCOMCTL 为 3'b111)，通道根据 TIMERx_CAR 寄存器和 TIMERx_CHxCV 寄存器的值，输出 PWM 波形。

根据计数模式，我们可以分为两种 PWM 波：EAPWM(边沿对齐 PWM)和 CAPWM(中央对齐 PWM)。

EAPWM 的周期由 TIMERx_CAR 寄存器值决定，占空比由 TIMERx_CHxCV 寄存器值决定。

*图 18-50. EAPWM 时序图*显示了 EAPWM 的输出波形和中断。

CAPWM 的周期由(2*TIMERx_CAR 寄存器值)决定，占空比由(2*TIMERx_CHxCV 寄存器值)决定。*图 18-51. CAPWM 时序图*显示了 CAPWM 的输出波形和中断。

在 PWM0 模式下(CHxCOMCTL==3'b110)，如果 TIMERx_CHxCV 寄存器的值大于 TIMERx_CAR 寄存器的值，通道输出一直为有效电平。

在 PWM0 模式下(CHxCOMCTL==3'b110)，如果 TIMERx_CHxCV 寄存器的值等于 0，通道输出一直为无效电平。

**图 18-50. EAPWM 时序图**

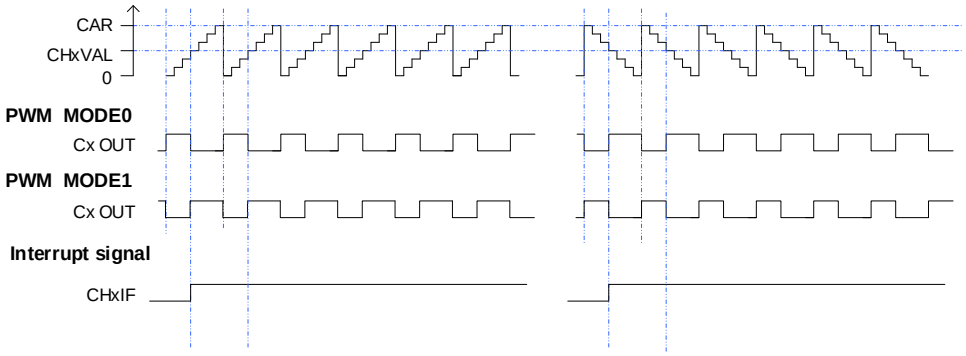
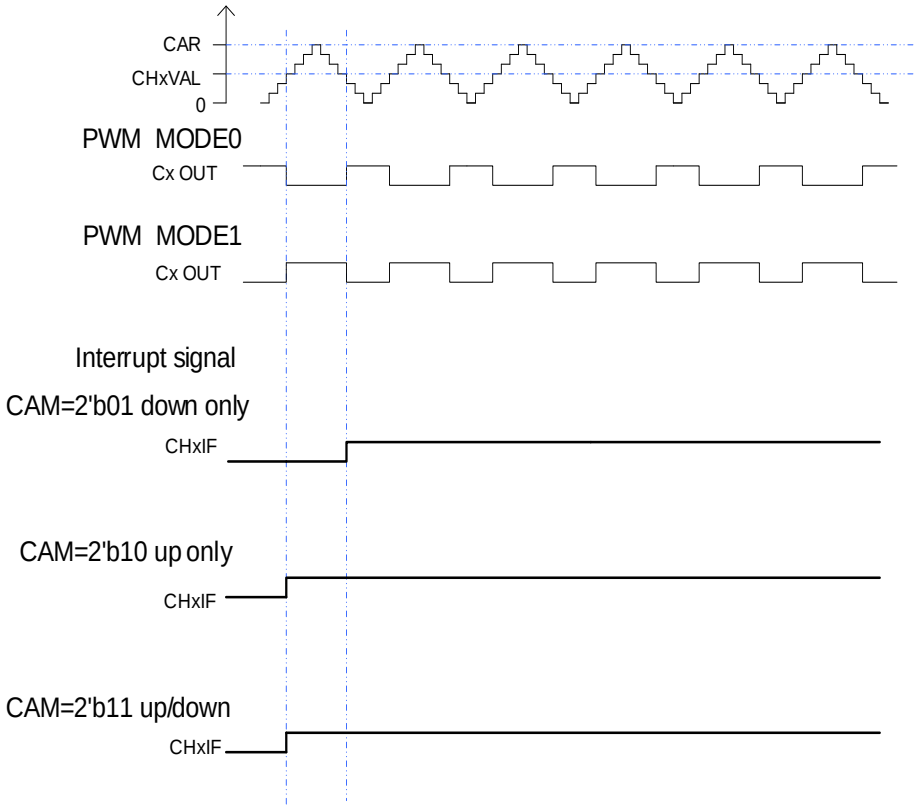



**图 18-51. CAPWM 时序图**




### 通道输出准备信号

当 TIMERx 用于输出匹配比较模式下，设置 CHxCOMCTL 位可以定义 OxCPRE 信号(通道 x 准备信号)类型。OxCPRE 信号有若干类型的输出功能，包括，设置 CHxCOMCTL=0x00 可以保持原始电平；设置 CHxCOMCTL=0x01 可以将 OxCPRE 信号设置为高电平；设置 CHxCOMCTL=0x02 可以将 OxCPRE 信号设置为低电平；设置 CHxCOMCTL=0x03，在计数器值和 TIMERx_CHxCV 寄存器的值匹配时，可以翻转输出信号。

PWM 模式 0 和 PWM 模式 1 是 OxCPRE 的另一种输出类型，设置 CHxCOMCTL 位域位 0x06 或 0x07 可以配置 PWM 模式 0/PWM 模式 1。在这些模式中，根据计数器值和 TIMERx_CHxCV 寄存器值的关系以及计数方向，OxCPRE 信号改变其电平。具体细节描述，请参考相应的位。

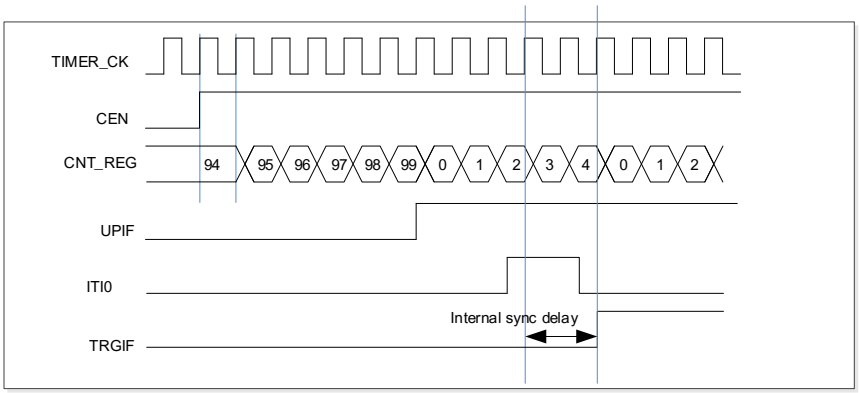
设置 CHxCOMCTL =0x04 或 0x05 可以实现 OxCPRE 信号的强制输出功能。输出比较信号能够直接由软件强置为有效或无效状态，而不依赖于 TIMERx_CHxCV 的值和计数器值之间间的比较结果。

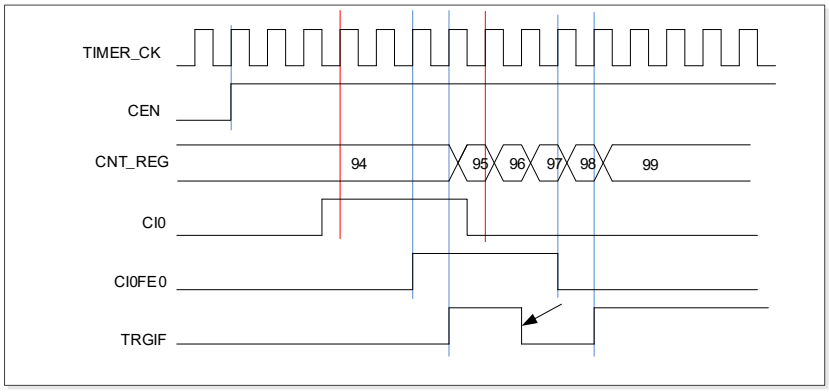
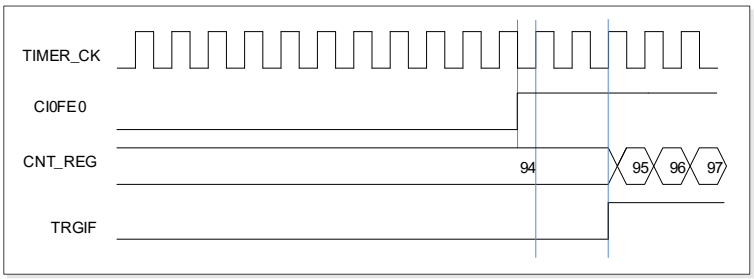
设置 CHxCOMCEN=1，当由外部 ETI 引脚信号产生的 ETIFE 信号为高电平时，OxCPRE 被强制为低电平。在下一次更新事件到来时，OxCPRE 信号才会回到有效电平状态。

## 主-从管理

TIMERx 能在多种模式下同步外部触发，包括复位模式，暂停模式和事件模式，可以通过设置 TIMERx_SMCFG 寄存器中的 SMC [2:0]配置这些模式。这些模式的输入触发源可以通过设置 TIMERx_SMCFG 寄存器中的 TRGS [2:0]来选择。

**表 18-6. 从模式列表和举例**

| | 模式选择 | 触发源选择 | 极性选择 | 滤波和预分频 |
|---|---|---|---|---|
| **列举** | SMC[2:0]<br>3'b100(复位模式)<br>3'b101(暂停模式)<br>3'b110(事件模式) | TRGS[2:0]<br>000：ITI0<br>001：ITI1<br>010：ITI2<br>011：ITI3<br>100：CI0F_ED<br>101：CI0FE0<br>110：CI1FE1<br>111：ETIFP | 如果触发源是CI0FE0或者CI1FE1，配置CHxP和CHxNP来选择极性和反相。<br>如果触发源是ETIFP，配置ETP选择极性和反相。 | 若触发源为ITIx，滤波和预分频不可用。<br>若触发源为CIx，可配置CHxCAPFLT设置滤波，预分频不可用。<br>若触发源为ETIFP，滤波和预分频均可用。 |
| **例1** | **复位模式**<br>当触发输入上升沿到来时，计数器清零重启。 | TRGS[2:0]=3'b000<br>选择ITI0为触发源。 | 若触发源是ITI0，极性选择不可用。 | 若触发源是ITI0，滤波和预分频不可用。 |
| | **图 18-52. 复位模式**<br> | | | |
| **例2** | **暂停模式**<br>当触发输入为低的时 | TRGS[2:0]=3'b101<br>选择CI0FE0为触发 | TI0S=0(非异或)[CH0NP=0， | 在这个例子中滤波被旁路。 |

<table>
<tr><td></td><td>模式选择</td><td>触发源选择</td><td>极性选择</td><td>滤波和预分频</td></tr>
<tr><td rowspan="2"></td><td>候，计数器暂停计数，当触发输入为高时，计数器计数。</td><td>源。</td><td>CH0P=0] CI0FE0不反相。捕获发生在上升沿 。</td><td></td></tr>
<tr><td colspan="4">图 18-53. 暂停模式<br>
</td></tr>
<tr><td rowspan="2">例3</td><td>事件模式<br>触发输入的上升沿计数器开始计数。</td><td>TRGIS[2:0]=3'b101 选择CI0FE0为触发源</td><td>CH0P==0，不反相.在上升沿捕获</td><td>在这个例子中滤波被旁路</td></tr>
<tr><td colspan="4">图 18-54. 事件模式<br>
</td></tr>
</table>

## 单脉冲模式

单脉冲模式与重复模式是相反的，设置 TIMERx_CTL0 寄存器的 SPM 位置 1，则使能单脉冲模式。当 SPM 置 1，计数器在下次更新事件到来后清零并停止计数。为了得到脉冲波，可以通过设置 CHxCOMCTL 配置 TIMERx 为 PWM 模式或者比较模式。

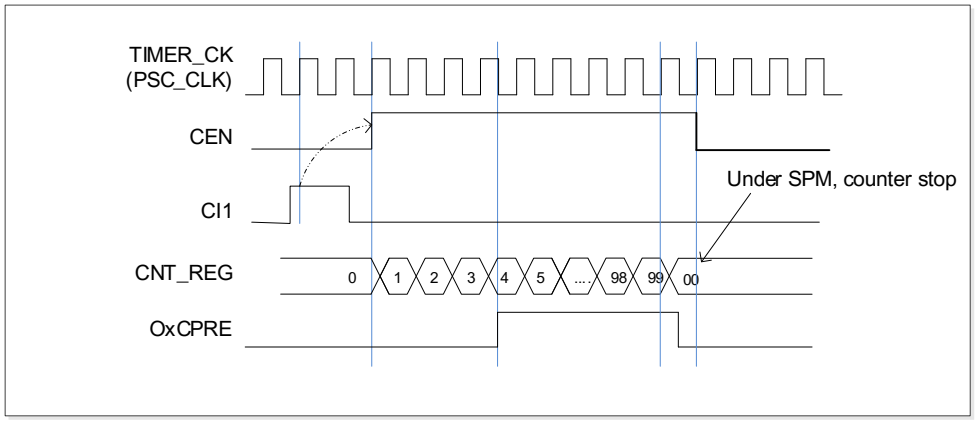
一旦设置定时器运行在单脉冲模式下，没有必要设置 TIMERx_CTL0 寄存器的定时器使能位 CEN=1 来使能计数器。触发信号沿或者软件写 CEN=1 都可以产生一个脉冲，此后 CEN 位一直保持为 1 直到更新事件发生或者 CEN 位被软件写 0。如果 CEN 位被软件清 0，计数器停止工作， 计数值被保持。

在单脉冲模式下，有效的外部触发边沿会将 CEN 位置 1，使能计数器。然而，执行计数值和 TIMERx_CHxCV 寄存器值的比较结果依然存在一些时钟延迟。为了最大限度减少延迟，用户可以将 TIMERx_CHCTL0/1 寄存器的 CHxCOMFEN 位置 1。单脉冲模式下，触发上升沿产生之后， OxCPRE 信号将被立即强制转换为与发生比较匹配时相同的电平，但是不用考虑比较结果。只有输出通道配置为 PWM0 或 PWM1 输出运行模式下时 CHxCOMFEN 位才可用，触

发源来源于触发信号

*图18-55. 单脉冲模式，TIMERx_CHxCV = 4 TIMERx_CAR=99*展示了一个例子。

**图 18-55. 单脉冲模式，TIMERx_CHxCV = 4 TIMERx_CAR=99**




## 定时器互连

参考*高级定时器(TIMERx,x=0,7)*。

## 定时器调试模式

当Cortex®-M4内核停止，DBG_CTL2寄存器中的TIMERx_HOLD配置位被置1，定时器计数器停止

### 18.3.4. TIMERx 寄存器(x=8,11)

TIMER8 基地址:0x4001 4000

TIMER11 基地址:0x4000 1800

#### 控制寄存器 0(TIMERx_CTL0)

地址偏移：0x00
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | CKDIV[1:0] | | ARSE | 保留 | | | SPM | UPS | UPDIS | CEN |
| | | | | | | rw | | rw | | | | rw | rw | rw | rw |

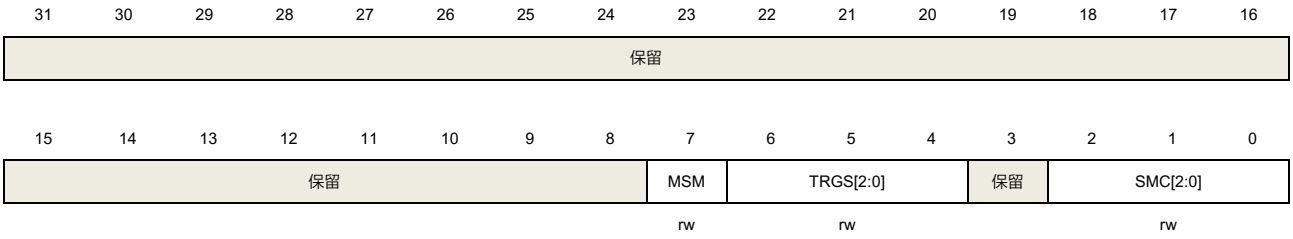
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:10 | 保留 | 必须保持复位值。 |
| 9:8 | CKDIV[1:0] | 时钟分频<br>通过软件配置CKDIV，规定定时器时钟(CK_TIMER) 与死区时间和数字滤波器采样时钟(DTS)之间的分频系数。<br>00：$f_{DTS}=f_{CK_TIMER}$<br>01：$f_{DTS}= f_{CK_TIMER} /2$<br>10：$f_{DTS}= f_{CK_TIMER} /4$<br>11：保留 |
| 7 | ARSE | 自动重载影子使能<br>0：禁能 TIMERx_CAR 寄存器的影子寄存器。<br>1：使能 TIMERx_CAR 寄存器的影子寄存器。 |
| 6:4 | 保留 | 必须保持复位值。 |
| 3 | SPM | 单脉冲模式<br>0：单脉冲模式禁能。更新事件发生后，计数器继续计数<br>1：单脉冲模式使能。在下一次更新事件发生时，计数器停止计数 |
| 2 | UPS | 更新请求源<br>软件配置该位，选择更新事件源.<br>0：以下事件均会产生更新中断或DMA请求：<br>UPG位被置1<br>计数器溢出/下溢<br>复位模式产生的更新<br>1：下列事件会产生更新中断或DMA请求： |

| | | |
|---|---|---|
| | | 计数器溢出/下溢 |
| 1 | UPDIS | 禁止更新.<br>该位用来使能或禁能更新事件的产生<br>0：更新事件使能. 更新事件发生时，相应的影子寄存器被装入预装载值，以下事件均会产生更新事件：<br>UPG位被置1<br>计数器溢出/下溢<br>复位模式产生的更新<br>1：更新事件禁能.<br>注意：当该位被置 1 时，UPG 位被置 1 或者复位模式不会产生更新事件，但是计数器和预分频器被重新初始化 |
| 0 | CEN | 计数器使能<br>0: 计数器禁能。<br>1: 计数器使能。<br>在软件将CEN位置1后，外部时钟、暂停模式和译码器模式才能工作。 |

### 从模式配置寄存器(TIMERx_SMCFG)

地址偏移：0x08
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7 | MSM | 主-从模式<br>该位被用来同步被选择的定时器同时开始计数。通过 TRIGI 和 TRGO，定时器被连接在一起，TRGO 用做启动事件。<br>0: 主从模式禁能。<br>1: 主从模式使能。 |
| 6:4 | TRGS[2:0] | 触发选择<br>该位域用来指定选择哪一个信号作为用来同步计数器的触发输入源。<br>000: ITI0<br>001: ITI1<br>010: ITI2<br>011: ITI3 |

| | | |
|---|---|---|
| | | 100: CI0F_ED<br>101: CI0FE0<br>110: CI1FE1<br>111: 保留<br>从模式被使能后这些位不能改。 |
| 3 | 保留 | 必须保持复位值。 |
| 2:0 | SMC[2:0] | 从模式控制<br>000: 关闭从模式. 如果 CEN=1，则预分频器直接由内部时钟驱动。<br>001: 保留<br>010: 保留<br>011: 保留<br>100: 复位模式. 选中的触发输入的上升沿重新初始化计数器，并且产生更新事件。<br>101:暂停模式. 当触发输入为高时，计数器的时钟开启。一旦触发输入变为低，则计数器时钟停止。<br>110: 事件模式.计数器在触发输入的上升沿启动。<br>111: 外部时钟模式 0. 选中的触发输入的上升沿驱动计数器。 |

### DMA 和中断使能寄存器(TIMERx_DMAINTEN)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | TRGIE | 保留 | | | CH1IE | CH0IE | UPIE |
| | | | | | | | | | rw | | rw | | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:7 | 保留 | 必须保持复位值。 |
| 6 | TRGIE | 触发中断使能<br>0：禁止触发中断。<br>1：使能触发中断。 |
| 5:3 | 保留 | 必须保持复位值。 |
| 2 | CH1IE | 通道 1 比较/捕获中断使能<br>0：禁止通道 1 中断。<br>1：使能通道 1 中断。 |
| 1 | CH0IE | 通道 0 比较/捕获中断使能<br>0：禁止通道 0 中断。 |

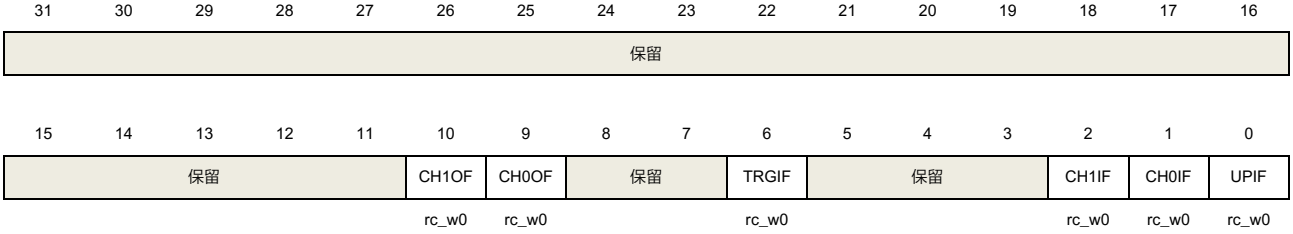
1：使能通道 0 中断。

| | | |
|---|---|---|
| 0 | UPIE | 更新中断使能<br>0：禁止更新中断。<br>1：使能更新中断。 |

### 中断标志寄存器(TIMERx_INTF)

地址偏移：0x10
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:11 | 保留 | 必须保持复位值。 |
| 10 | CH1OF | 通道 1 捕获溢出标志<br>参见 CH0OF 描述。 |
| 9 | CH0OF | 通道 0 捕获溢出标志<br>当通道 0 被配置为输入模式时，在 CH0IF 标志位已经被置 1 后，捕获事件再次发生时，该标志位可以由硬件置 1。该标志位由软件清 0。<br>0: 无捕获溢出中断发生。<br>1: 发生了捕获溢出中断。 |
| 8:7 | 保留 | 必须保持复位值。 |
| 6 | TRGIF | 触发中断标志<br>当发生触发事件时，此标志会置 1，此位由软件清 0。当暂停模式使能时，触发输入的任意边沿都可以产生触发事件。否则，其它模式时，仅在触发输入端检测到有效边沿，产生触发事件。<br>0：无触发事件产生。<br>1：触发中断产生。 |
| 5:3 | 保留 | 必须保持复位值。 |
| 2 | CH1IF | 通道 1 比较/捕获中断标志<br>参见 CH0IF 描述。 |
| 1 | CH0IF | 通道 0 比较/捕获中断标志<br>此标志由硬件置 1 软件清 0。当通道 0 在输入模式下时，捕获事件发生时此标志位被置 1；当通道 0 在输出模式下时，此标志位在一个比较事件发生时被置 1。 |

0：无通道 0 中断发生。
1：通道 0 中断发生。

0 UPIF 更新中断标志
此位在任何更新事件发生时由硬件置 1，软件清 0。
0：无更新中断发生。
1：发生更新中断。

### 软件事件产生寄存器(TIMERx_SWEVG)

地址偏移：0x14
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | TRGG | 保留 | | | CH1G | CH0G | UPG |
| | | | | | | | | | w | | | | w | w | w |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:7 | 保留 | 必须保持复位值。 |
| 6 | TRGG | 触发事件产生<br>此位由软件置 1，由硬件自动清 0. 当此位被置 1，TIMERx_INTF 寄存器的 TRGIF 标志位被置 1，若开启对应的中断和 DMA，则产生相应的中断和 DMA 传输。<br>0：无触发事件产生。<br>1：产生触发事件。 |
| 5:3 | 保留 | 必须保持复位值。 |
| 2 | CH1G | 通道 1 捕获或比较事件发生<br>参见 CH0G 描述。 |
| 1 | CH0G | 通道 0 捕获或比较事件发生<br>该位由软件置 1，用于在通道 0 产生一个捕获/比较事件，由硬件自动清 0。当此位被置 1，CH0IF 标志位被置 1，若开启对应的中断和 DMA，则发出相应的中断和 DMA 请求。此外，如果通道 0 配置为输入模式，计数器的当前值被 TIMERx_CH0CV 寄存器捕获，如果 CH0IF 标志位已经为 1，则 CH0OF 标志位被置 1。<br>0：不产生通道 0 捕获或比较事件。<br>1：发生通道 0 捕获或比较事件。 |
| 0 | UPG | 更新事件产生<br>此位由软件置 1，被硬件自动清 0。当此位被置 1，如果选择了中央对齐或向上计数模式，计数器被清 0。否则(向下计数模式)计数器将载入自动重载值，预分频计数器将同时被清除。 |

0：无更新事件产生。
1：产生更新事件。

### 通道控制寄存器 0(TIMERx_CHCTL0)

地址偏移：0x18
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留. | CH1COMCTL[2:0] | | | CH1COMSEN | CH1COMFEN | CH1MS[1:0] | | 保留. | CH0COMCTL[2:0] | | | CH0COMSEN | CH0COMFEN | CH0MS[1:0] | |
| CH1CAPFLT[3:0] | | | | CH1CAPPSC[1:0] | | | | CH0CAPFLT[3:0] | | | | CH0CAPPSC[1:0] | | | |
| rw | | | | rw | | rw | | rw | | | | rw | | rw | |

**输出比较模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:15 | 保留 | 必须保持复位值。 |
| 14:12 | CH1COMCTL[2:0] | 通道 1 输出比较模式<br>参见 CH0COMCTL 描述。 |
| 11 | CH1COMSEN | 通道 1 输出比较影子寄存器使能<br>参见 CH0COMSEN 描述。 |
| 10 | CH1COMFEN | 通道 1 输出比较快速使能<br>参见 CH0COMFEN 描述。 |
| 9:8 | CH1MS[1:0] | 通道 1 模式选择<br>这些位定义了通道的方向和输入信号的选择。只有当通道关闭(TIMERx_CHCTL2 寄存器的 CH1EN 位被清 0)时这些位才可以写。<br>00：通道 1 配置为输出<br>01：通道 1 配置为输入，IS1 映射在 CI1FE1 上<br>10：通道 1 配置为输入，IS1 映射在 CI0FE1 上<br>11：通道 1 配置为输入，IS1 映射在 ITS 上<br>注意：当 CH1MS[1:0]=11 时，需要通过 TRGS 位(位于 TIMERx_SMCFG 寄存器)选择内部触发输入 |
| 7 | 保留 | 必须保持复位值。 |
| 6:4 | CH0COMCTL[2:0] | 通道 0 输出比较模式<br>此位定义了输出准备信号 O0CPRE 的输出比较模式，而 O0CPRE 决定了 CH0_O、CH0_ON 的值。另外，O0CPRE 高电平有效，而 CH0_O、CH0_ON 通道的极性取决于 CH0P、CH0NP 位。<br>000：时基。输出比较寄存器 TIMERx_CH0CV 与计数器 TIMERx_CNT 间的比较对 |

| | | |
|---|---|---|
| | | O0CPRE 不起作用<br>001：匹配时设置为高。当计数器的值与捕获/比较值寄存器 TIMERx_CH0CV 相同时，强制 O0CPRE 为高。<br>010：匹配时设置为低。当计数器的值与捕获/比较值寄存器 TIMERx_CH0CV 相同时，强制 O0CPRE 为低。<br>011：匹配时翻转。当计数器的值与捕获/比较值寄存器 TIMERx_CH0CV 相同时，强制 O0CPRE 翻转。<br>100：强制为低。强制 O0CPRE 为低电平<br>101：强制为高。强制 O0CPRE 为高电平<br>110：PWM 模式 0。在向上计数时，一旦计数器值小于 TIMERx_CH0CV 时，O0CPRE 为高电平，否则为低电平。在向下计数时，一旦计数器的值大于 TIMERx_CH0CV 时，O0CPRE 为低电平，否则为高电平。<br>111：PWM 模式 1。在向上计数时，一旦计数器值小于 TIMERx_CH0CV 时，O0CPRE 为低电平，否则为高电平。在向下计数时，一旦计数器的值大于 TIMERx_CH0CV 时，O0CPRE 为高电平，否则为低电平。<br>如果配置在 PWM 模式下，只有当输出比较模式从时基模式变为 PWM 模式或者比较结果改变时，O0CPRE 电平才改变。<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 且 CH0MS =00(比较模式)时此位不能被改变。 |
| 3 | CH0COMSEN | 通道 0 输出比较影子寄存器使能<br>当此位被置 1，TIMERx_CH0CV 寄存器的影子寄存器被使能，影子寄存器在每次更新事件时都会被更新。<br>0：禁止通道 0 输出/比较影子寄存器<br>1：使能通道 0 输出/比较影子寄存器<br>仅在单脉冲模式下(SPM =1)，可以在未确认影子寄存器的情况下使用 PWM 模式<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 且 CH0MS =00 时此位不能被改变。 |
| 2 | CH0COMFEN | 通道 0 输出比较快速使能<br>0：通道 0 输出比较快速禁能。仅比较结果输入作为有效边沿时会产生比较匹配，并将 CH0_O 设置为比较电平，激活 CH0_O 输出的最小延时为 5 个时钟周期。<br>1：通道 0 输出比较快速使能。触发输入信号的有效边沿和比较结果都会产生比较匹配，并将 CH0_O 设置为比较电平。当触发信号的输入作为有效边沿时，激活 CH0_O 输出的最小延时为 3 个时钟周期。 |
| 1:0 | CH0MS[1:0] | 通道 0 I/O 模式选择<br>这些位定义了通道的工作模式和输入信号的选择。只有当通道关闭(TIMERx_CHCTL2 寄存器的 CH0EN 位被清 0)时这些位才可写。<br>00：通道 0 配置为输出<br>01：通道 0 配置为输入，IS0 映射在 CI0FE0 上<br>10：通道 0 配置为输入，IS0 映射在 CI1FE0 上<br>11：通道 0 配置为输入，IS0 映射在 ITS 上<br>注意：当 CH0MS[1:0]=11 时，需要通过 TRGS 位(位于 TIMERx_SMCFG 寄存器)选择内部触发输入 |

**输入捕获模式：**

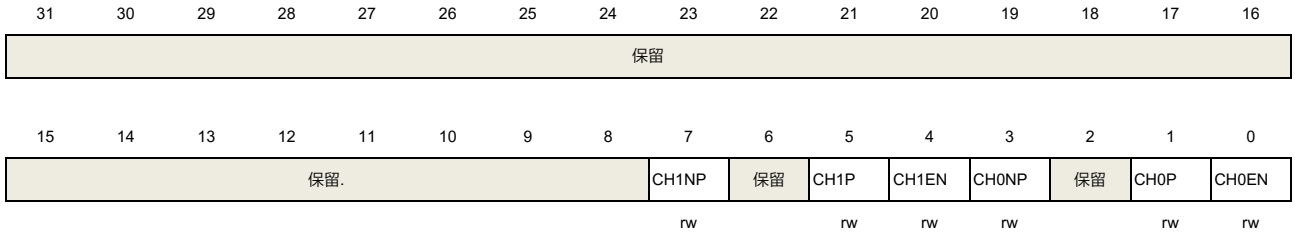
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:12 | CH1CAPFLT[3:0] | 通道 1 输入捕获滤波控制<br>参见 CH0CAPFLT 描述。 |
| 11:10 | CH1CAPPSC[1:0] | 通道 1 输入捕获预分频器<br>参见 CH0CAPPSC 描述。 |
| 9:8 | CH1MS[1:0] | 通道 1 模式选择<br>与输出模式相同。 |
| 7:4 | CH0CAPFLT[3:0] | 通道 0 输入捕获滤波控制<br>CI0 输入信号可以通过数字滤波器进行滤波，该位域配置滤波参数。<br>数字滤波器的基本原理：根据 $f_{SAMP}$ 对 CI0 输入信号进行连续采样，并记录信号相同电平的次数。达到该位配置的滤波参数后，认为是有效电平。<br>滤波器参数配置如下： |
| 3:2 | CH0CAPPSC[1:0] | 通道 0 输入捕获预分频器<br>这 2 位定义了通道 0 输入的预分频系数。当 TIMERx_CHCTL2 寄存器中的 CH0EN=0 时，则预分频器复位。<br>00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获。<br>01：每 2 个事件触发一次捕获。<br>10：每 4 个事件触发一次捕获。<br>11：每 8 个事件触发一次捕获。 |
| 1:0 | CH0MS[1:0] | 通道 0 模式选择<br>与输出比较模式相同。 |

| CH0CAPFLT [3:0] | 采样次数 | $f_{SAMP}$ |
|---|---|---|
| 4'b0000 | 无滤波器 | |
| 4'b0001 | 2 | $f_{CK_TIMER}$ |
| 4'b0010 | 4 | |
| 4'b0011 | 8 | |
| 4'b0100 | 6 | $f_{DTS}/2$ |
| 4'b0101 | 8 | |
| 4'b0110 | 6 | $f_{DTS}/4$ |
| 4'b0111 | 8 | |
| 4'b1000 | 6 | $f_{DTS}/8$ |
| 4'b1001 | 8 | |
| 4'b1010 | 5 | $f_{DTS}/16$ |
| 4'b1011 | 6 | |
| 4'b1100 | 8 | |
| 4'b1101 | 5 | $f_{DTS}/32$ |
| 4'b1110 | 6 | |
| 4'b1111 | 8 | |

### 通道控制寄存器 2(TIMERx_CHCTL2)

地址偏移：0x20
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 名称 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7 | CH1NP | 通道 1 互补输出极性<br>参考 CH0NP 描述。 |
| 6 | 保留 | 必须保持复位值。 |
| 5 | CH1P | 通道 1 极性<br>参考 CH0P 描述。 |
| 4 | CH1EN | 通道 1 使能<br>参考 CH0EN 描述。 |
| 3 | CH0NP | 通道 0 互补输出极性<br>当通道 0 配置为输出模式，此位定义了互补输出信号的极性。<br>0：通道0互补输出高电平为有效电平<br>1：通道0互补输出低电平为有效电平<br>当通道 0 配置为输入模式时，此位和 CH0P 联合使用，作为输入信号 CI0 的极性选择控制信号。<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 或 10 时此位不能被更改。 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | CH0P | 通道 0 极性<br>当通道 0 配置为输出模式时，此位定义了输出信号极性。<br>0：通道0高电平为有效电平<br>1：通道0低电平为有效电平<br>当通道 0 配置为输入模式时，此位定义了 CI0 信号极性。<br>[CH0NP, CH0P] 将选择 CI0FE0 或者 CI1FE0 的有效边沿或者捕获极性。<br>[CH0NP==0, CH0P==0]：把 CIxFE0 的上升沿作为捕获或者从模式下触发的有效信号，并且 CIxFE0 不会被翻转。<br>[CH0NP==0, CH0P==1]：把 CIxFE0 的下降沿作为捕获或者从模式下触发的有效信号，并且 CIxFE0 会被翻转。<br>[CH0NP==1, CH0P==0]：保留。 |

[CH0NP==1, CH0P==1]：把 CIxFE0 的上升沿和下降沿都作为捕获或者从模式下触发的有效信号，并且 CIxFE0 不会被翻转。
当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 或 10 时此位不能被更改。

| | | |
|---|---|---|
| 0 | CH0EN | 通道 0 捕获/比较使能<br>当通道 0 配置为输出模式时，将此位置 1 使能 CH0_O 信号有效。当通道 0 配置为输入模式时，将此位置 1 使能通道 0 上的捕获事件。<br>0：禁止通道 0。<br>1：使能通道 0。 |

### 计数器寄存器(TIMERx_CNT)

地址偏移：0x24
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CNT[15:0] | 这些位是当前的计数值。写操作能改变计数器值。 |

### 预分频寄存器(TIMERx_PSC)

地址偏移： 0x28
复位值： 0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | PSC[15:0] | 计数器时钟预分频值<br>计数器时钟等于 TIMER_CK 时钟除以(PSC+1)，每次当更新事件产生时，PSC 的值 |

被装入到对应的影子寄存器。

### 计数器自动重载寄存器(TIMERx_CAR)

地址偏移：0x2C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CARL[15:0] | 计数器自动重载值<br>这些位定义了计数器的自动重载值。 |

### 通道 0 捕获/比较值寄存器(TIMERx_CH0CV)

地址偏移：0x34
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH0VAL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CH0VAL[15:0] | 通道 0 的捕获或比较值<br>当通道 0 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。<br>当通道 0 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。 |

### 通道 1 捕获/比较值寄存器(TIMERx_CH1CV)

地址偏移：0x38
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CH1VAL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CH1VAL[15:0] | 通道 1 的捕获或比较值<br>当通道 1 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。<br>当通道 1 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。 |

### 配置寄存器(TIMERx_CFG)

地址偏移：0xFC
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |
| 1 | CHVSEL | 写捕获比较寄存器选择位<br>此位由软件写 1 或清 0。<br>1：当写入捕获比较寄存器的值与寄存器当前值相等时，写入操作无效。<br>0：无影响。 |
| 0 | 保留 | 必须保持复位值。 |

# 18.4. 通用定时器 L2(TIMERx, x=9,10,12,13)

## 18.4.1. 简介

通用定时器 L2(TIMERx, x=9, 10, 12, 13)是单通道定时器，支持输入捕获和输出比较，产生 PWM 信号控制电机和电源管理。通用定时器 L2 含有一个 16 位无符号计数器。

高级定时器是可编程的，可以被用来计数，其外部事件可以驱动其他定时器

## 18.4.2. 主要特性

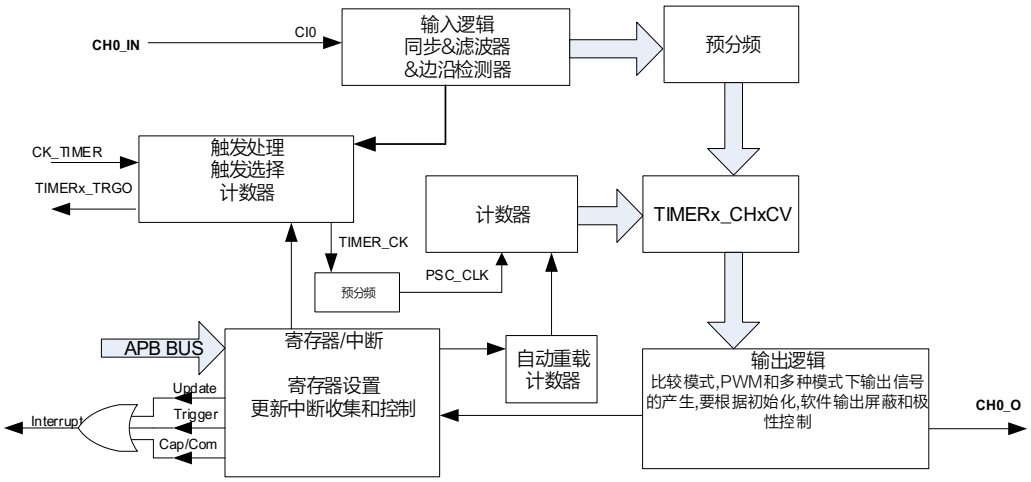
- 总通道数：1；
- 计数器宽度：16位；
- 时钟源只有内部时钟；
- 计数模式：向上计数，向下计数和中央计数；
- 可编程的预分频器：16位，运行时可以被改变；
- 每个通道可配置：输入捕获模式，输出比较模式，可编程的PWM模式，单脉冲模式；
- 自动重装载功能；
- 中断输出：更新事件，触发事件，比较/捕获事件和中止事件。

## 18.4.3. 功能描述

### 结构框图

*图 18-56. 通用定时器 L2 结构框图*提供了通用定时器 L2 的内部配置细节

**图 18-56. 通用定时器 L2 结构框图**



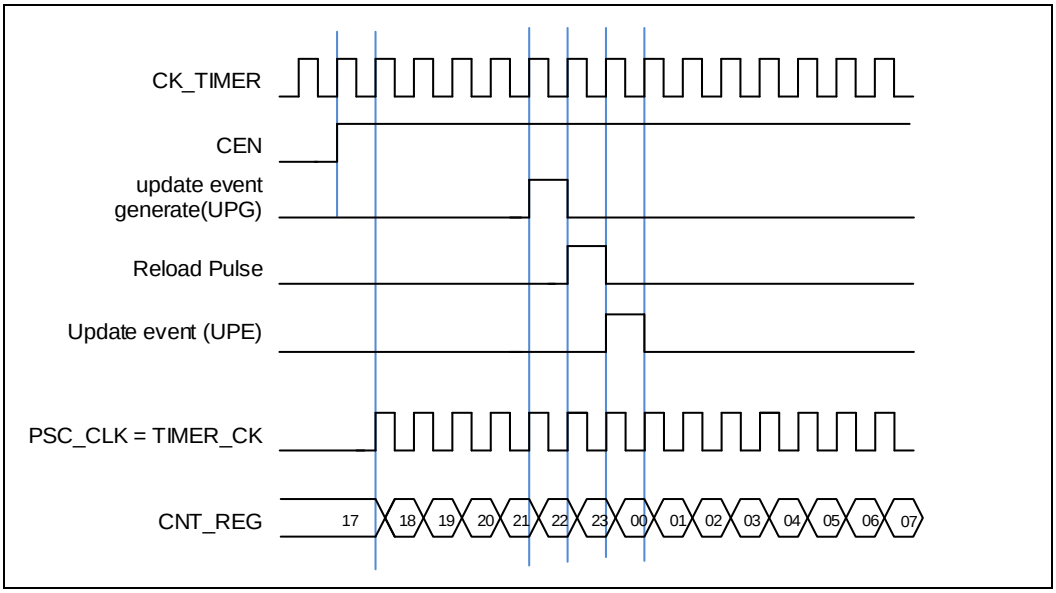
### 时钟源配置

通用定时器 L2 由内部时钟源 CK_TIMER 驱动

定时器时钟 TIMER_CK 连接到 RCU 模块的 CK_TIMER

通用定时器 L2 仅有一个时钟源 CK_TIMER，用来驱动计数器预分频器。当 CEN 置位，CK_TIMER 经过预分频器(预分频值由 TIMERx_PSC 寄存器确定)产生 PSC_CLK。

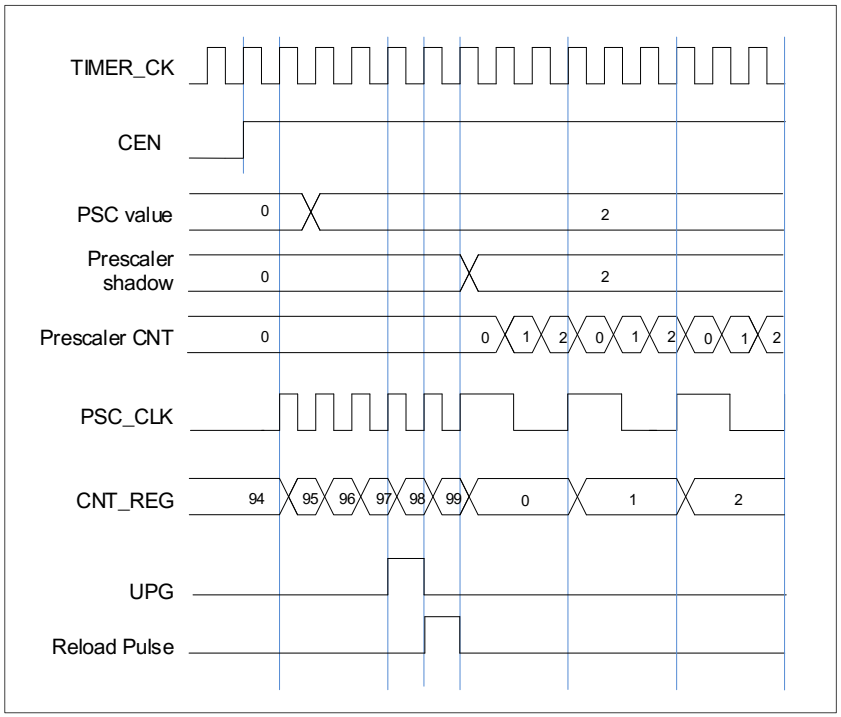
**图 18-57. 内部时钟分频为 1 时，计数器的时序图**




### 时钟预分频器

预分频器可以将定时器的时钟(TIMER_CK)频率按 1 到 65536 之间的任意值分频，分频后的时钟 PSC_CLK 驱动计数器计数。分频系数受预分频寄存器 TIMERx_PSC 控制，这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

图 18-58. 当 PSC 数值从 0 变到 2 时，计数器的时序图




### 计数器向上计数模式

在这种模式，计数器的计数方向是向上计数。计数器从 0 开始向上连续计数到自动加载值(定义在 TIMERx_CAR 寄存器中)，一旦计数器计数到自动加载值，会重新从 0 开始向上计数并产生上溢事件。在向上计数模式中，TIMERx_CTL0 寄存器中的计数方向控制位 DIR 应该被设置成 0。

当通过 TIMERx_SWEVG 寄存器的 UPG 位置 1 来设置更新事件时，计数值会被清 0，并产生更新事件。

如果 TIMERx_CTL0 寄存器的 UPDIS 置 1，则禁止更新事件。

当发生更新事件时，所有影子寄存器(重复计数器，计数器自动重载寄存器，预分频寄存器)都将被更新。

*图 18-59. 向上计数时序图，PSC=0/2* 和*图 18-60. 向上计数时序图，在运行时改变 TIMERx_CAR 寄存器的值*给出了一些例子，当 TIMERx_CAR=0x99 时，计数器在不同预分频

因子下的行为。

**图 18-59. 向上计数时序图，PSC=0/2**

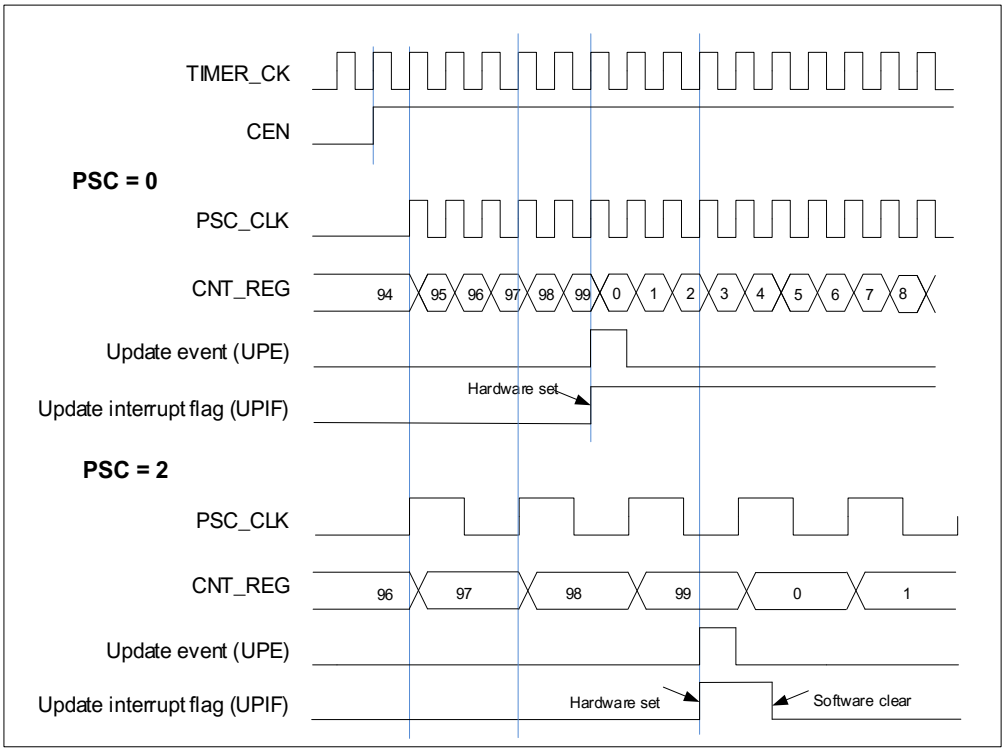
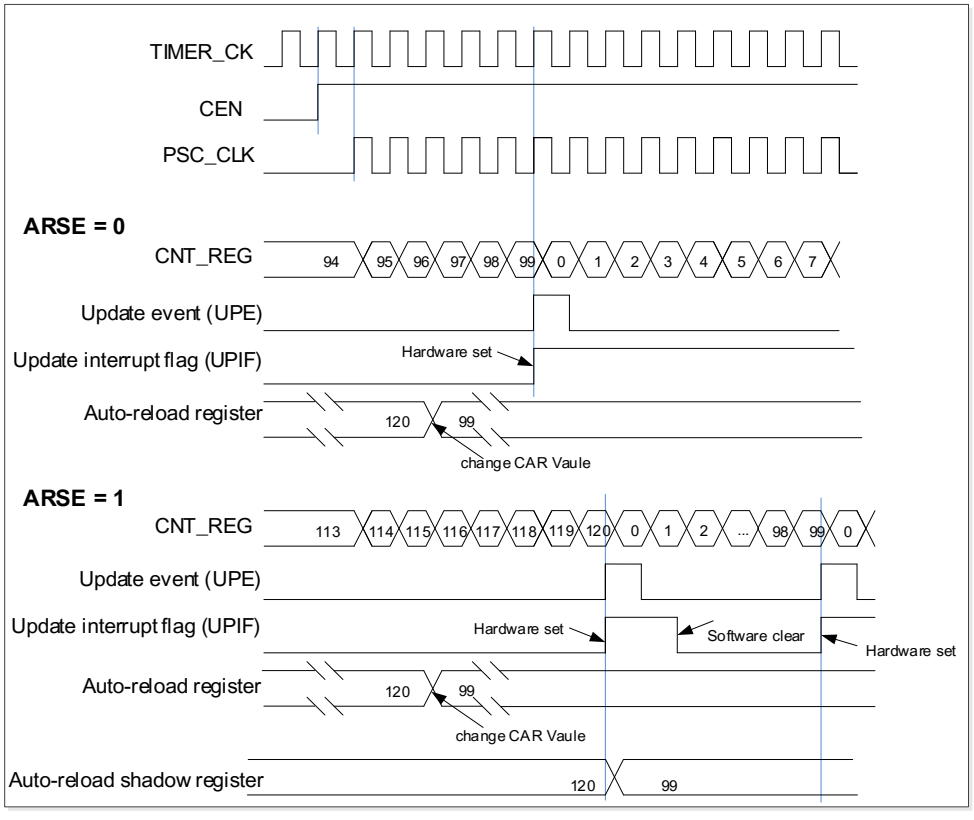



**图 18-60. 向上计数时序图，在运行时改变 TIMERx_CAR 寄存器的值**



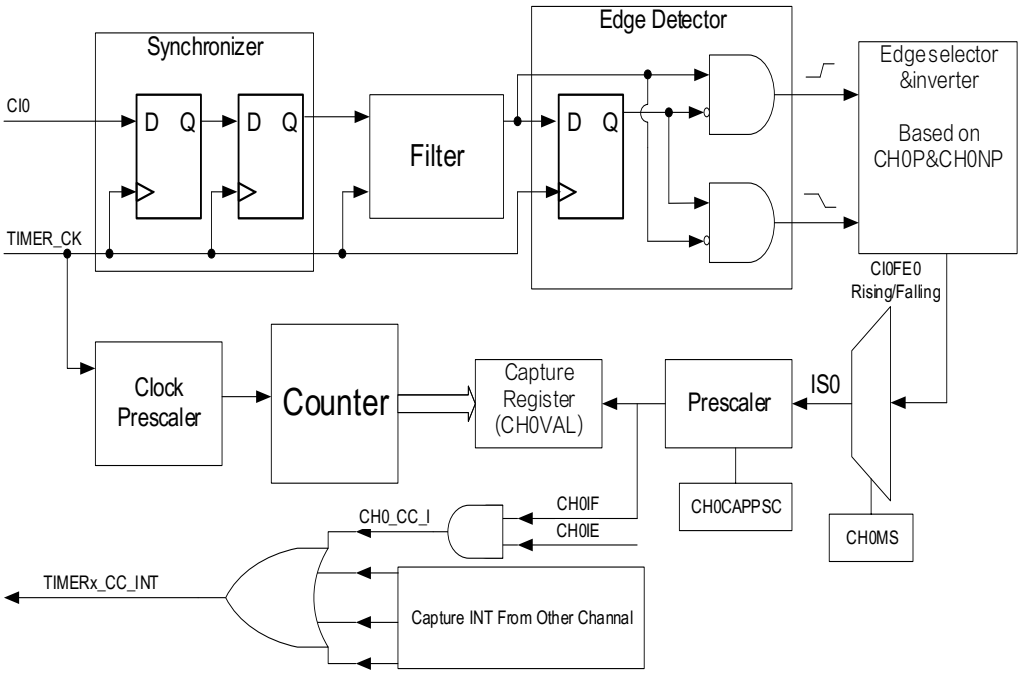
### 输入捕获和输出比较通道

通用定时器 L2 只有一个独立的通道用于捕获输入或比较输出是否匹配。该通道都围绕一个通道捕获比较寄存器建立，包括一个输入级，通道控制器和输出级。

■ 通道输入捕获功能

通道输入捕获功能允许通道测量一个波形时序，频率，周期，占空比等。输入级包括一个数字滤波器，一个通道极性选择，边沿检测和一个通道预分频器。如果在输入引脚上出现被选择的边沿，TIMERx_CHxCV 寄存器会捕获计数器当前的值，同时 CHxIF 位被置 1，如果 CHxIE = 1 则产生通道中断。

**图 18-61. 通道输入捕获原理**




通道输入信号 CIx 先被 TIMER_CK 信号同步，然后经过数字滤波器采样，产生一个被滤波后的信号。通过边沿检测器，可以选择检测上升沿或者下降沿。通过配置 CHxP 选择使用上升沿或者下降沿。配置 IC 预分频器，使得若干个输入事件后才产生一个有效的捕获事件。捕获事件发生，CxCV 存储计数器的值。

配置步骤如下：

**第一步：**滤波器配置(TIMERx_CHCTL0寄存器中CHxCAPFLT)：
根据输入信号和请求信号的质量，配置相应的CHxCAPFLT。
**第二步：**边沿选择(TIMERx_CHCTL2寄存器中CHxP/CHxNP)：
配置CHxP/CHxNP选择上升沿或者下降沿。
**第三步：**捕获源选择(TIMERx_CHCTL0寄存器中CHxMS)：
一旦通过配置CHxMS选择输入捕获源，必须确保通道配置在输入模式(CHxMS!=0x0)，而且TIMERx_CxCV寄存器不能再被写。
**第四步：**中断使能(TIMERx_DMAINTEN寄存器中CHxIE)：

使能相应中断，可以获得中断。

**第五步：**捕获使能(TIMERx_CHCTL2寄存器中CHxEN)。

**结果：**当期望的输入信号发生时，TIMERx_CHxCV被设置成当前计数器的值，CHxIF为置1。如果CHxIF位已经为1，则CHxOF位置1。根据TIMERx_DMAINTEN寄存器中CHxIE的配置，相应的中断会被提出。

**直接产生：**软件设置CHxG位，会直接产生中断。

■ 通道输出比较功能

在输出比较模式，TIMERx 可以产生时控脉冲，其位置，极性，持续时间和频率都是可编程的。当一个输出通道的 CxCV 寄存器与计数器的值匹配时，根据 CHxCOMCTL 的配置，这个通道的输出可以被置高电平，被置低电平或者反转。当计数器的值与 CxCV 寄存器的值匹配时，CHxIF 位被置 1，如果 CHxIE = 1 则会产生中断，如果 CxCDE=1 则会产生 DMA 请求。

配置步骤如下：

**第一步：**时钟配置：
配置定时器时钟源，预分频器等。

**第二步：**比较模式配置：
设置CHxCOMSEN位来配置输出比较影子寄存器；
设置CHxCOMCTL位来配置输出模式(置高电平/置低电平/反转)；
设置CHxP/CHxNP位来选择有效电平的极性；
设置CHxEN使能输出。

**第三步：**通过CHxIE位配置中断使能。

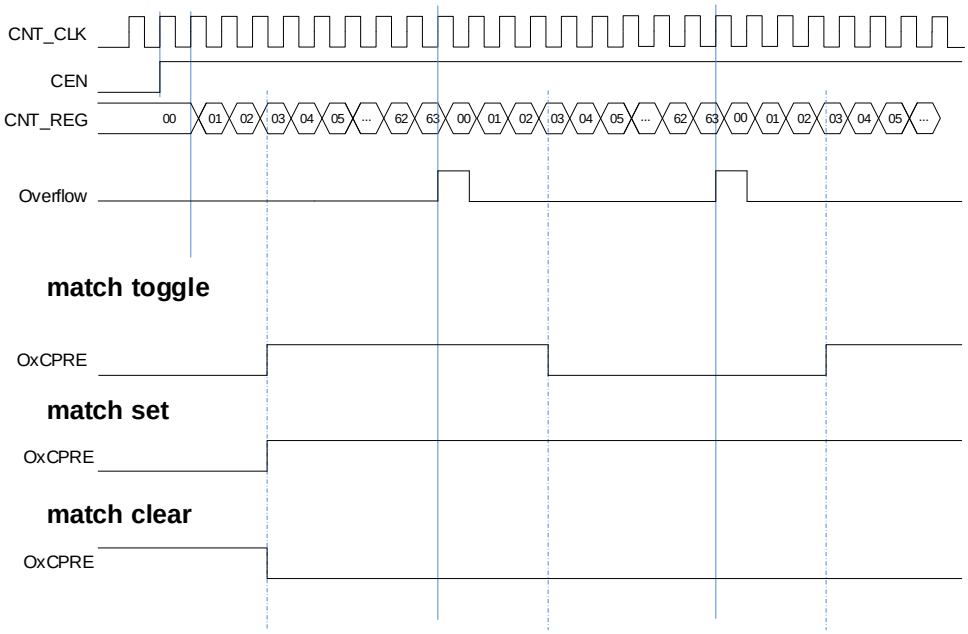
**第四步：**通过TIMERx_CAR寄存器和TIMERx_CHxCV寄存器配置输出比较时基：
CxCV可以在运行时根据你所期望的波形而改变。

**第五步：**设置CEN位使能定时器。

*图 18-62. 三种输出比较模式*显示了三种比较输出模式：反转/置高电平/置低电平，

CAR=0x63, CxCV=0x3。

**图 18-62. 三种输出比较模式**




### 通道输出准备信号

当 TIMERx 用于输出匹配比较模式下，设置 CHxCOMCTL 位可以定义 OxCPRE 信号(通道 x 准备信号)类型。OxCPRE 信号有若干类型的输出功能，包括，设置 CHxCOMCTL=0x00 可以保持原始电平；设置 CHxCOMCTL=0x01 可以将 OxCPRE 信号设置为高电平；设置 CHxCOMCTL=0x02 可以将 OxCPRE 信号设置为低电平；设置 CHxCOMCTL=0x03，在计数器值和 TIMERx_CHxCV 寄存器的值匹配时，可以翻转输出信号。

PWM 模式 0 和 PWM 模式 1 是 OxCPRE 的另一种输出类型，设置 CHxCOMCTL 位域位 0x06 或 0x07 可以配置 PWM 模式 0/PWM 模式 1。在这些模式中，根据计数器值和 TIMERx_CHxCV 寄存器值的关系以及计数方向，OxCPRE 信号改变其电平。具体细节描述，请参考相应的位。

设置 CHxCOMCTL =0x04 或 0x05 可以实现 OxCPRE 信号的强制输出功能。输出比较信号能够直接由软件强置为有效或无效状态，而不依赖于 TIMERx_CHxCV 的值和计数器值之间间的比较结果。

设置 CHxCOMCEN=1，当由外部 ETI 引脚信号产生的 ETIFE 信号为高电平时，OxCPRE 被强制为低电平。在下一次更新事件到来时，OxCPRE 信号才会回到有效电平状态。

### 定时器互连

参考*高级定时器(TIMERx,x=0,7)*。

### 定时器调试模式

当 Cortex®-M4 内核停止，DBG_CTL2 寄存器中的 TIMERx_HOLD 配置位被置 1，定时器计数器停止。

### 18.4.4. TIMERx 寄存器(x=9,10,12,13)

TIMER9 基地址：0x4001 4400

TIMER10 基地址：0x4001 4800

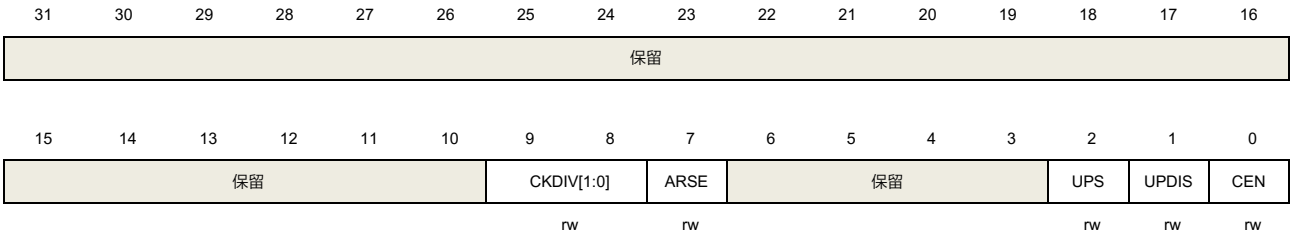
TIMER12 基地址：0x4000 1C00

TIMER13 基地址：0x4000 2000

#### 控制寄存器 0(TIMERx_CTL0)

地址偏移：0x00
复位值：0x0000 0000

该寄存器只能按字(32位)访问。



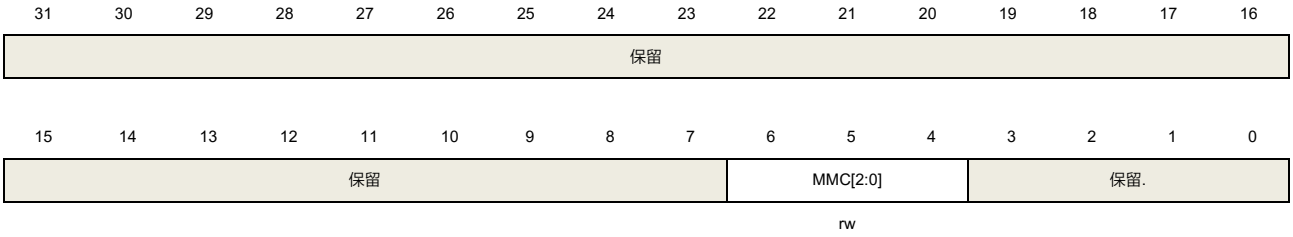

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:10 | 保留 | 必须保持复位值。 |
| 9:8 | CKDIV[1:0] | 时钟分频<br>通过软件配置CKDIV，规定定时器时钟(CK_TIMER) 与死区时间和数字滤波器采样时钟(DTS)之间的分频系数。<br>00：$f_{DTS}=f_{CK_TIMER}$<br>01：$f_{DTS}= f_{CK_TIMER} /2$<br>10：$f_{DTS}= f_{CK_TIMER} /4$<br>11：保留 |
| 7 | ARSE | 自动重载影子使能<br>0：禁能 TIMERx_CAR 寄存器的影子寄存器。<br>1：使能 TIMERx_CAR 寄存器的影子寄存器。 |
| 6:3 | 保留 | 必须保持复位值。 |
| 2 | UPS | 更新请求源<br>软件配置该位，选择更新事件源.<br>0：以下事件均会产生更新中断或DMA请求：<br>UPG位被置1<br>计数器溢出/下溢<br>复位模式产生的更新<br>1：下列事件会产生更新中断或DMA请求：<br>计数器溢出/下溢 |

| | | |
|---|---|---|
| 1 | UPDIS | 禁止更新.<br>该位用来使能或禁能更新事件的产生<br>0：更新事件使能. 更新事件发生时，相应的影子寄存器被装入预装载值，以下事件均会产生更新事件：<br>UPG位被置1<br>计数器溢出/下溢<br>复位模式产生的更新<br>1：更新事件禁能.<br>注意：当该位被置 1 时，UPG 位被置 1 或者复位模式不会产生更新事件，但是计数器和预分频器被重新初始化 |
| 0 | CEN | 计数器使能<br>0：计数器禁能。<br>1：计数器使能。<br>在软件将CEN位置1后，外部时钟、暂停模式和译码器模式才能工作。 |

### 控制寄存器 1(TIMERx_CTL1)

地址偏移：0x04
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:7 | 保留 | 必须保持复位值。 |
| 6:4 | MMC[2:0] | 主模式控制<br>这些位控制TRGO信号的选择，TRGO信号由主定时器发给从定时器用于同步功能。<br>000：当产生一个定时器复位事件后，输出一个TRGO信号，定时器复位源为：<br>主定时器产生一个复位事件<br>TIMERx_SWEVG寄存器中UPG位置1<br>001：当产生一个定时器使能事件后，输出一个TRGO信号，定时器使能源为：<br>CEN位置1<br>在暂停模式下，触发输入置1<br>010：当产生一个定时器更新事件后，输出一个TRGO信号，更新事件源由UPDIS和UPS位决定<br>011：当通道0在发生一次捕获或一次比较成功时，主模式控制器产生一个TRGO脉冲<br>100：当产生一次比较事件时，输出一个TRGO信号，比较事件源来自O0CPRE TRGO。 |

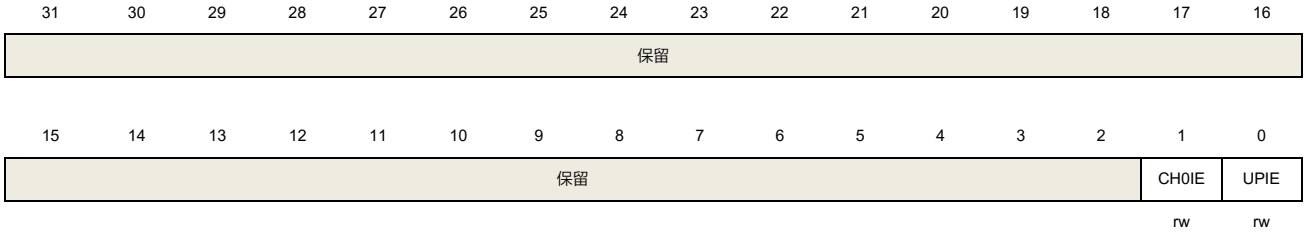
101：保留。
110：保留。
111：保留。

| | | |
|---|---|---|
| 3:0 | 保留 | 必须保持复位值。 |

### DMA 和中断使能寄存器(TIMERx_DMAINTEN)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

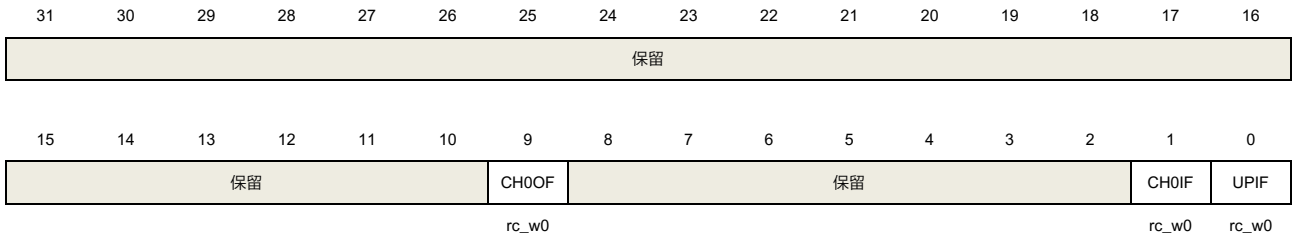



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |
| 1 | CH0IE | 通道 0 比较/捕获中断使能<br>0：禁止通道 0 中断。<br>1：使能通道 0 中断。 |
| 0 | UPIE | 更新中断使能<br>0：禁止更新中断。<br>1：使能更新中断。 |

### 中断标志寄存器(TIMERx_INTF)

地址偏移：0x10
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:10 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 9 | CH0OF | 通道 0 捕获溢出标志<br>当通道 0 被配置为输入模式时，在 CH0IF 标志位已经被置 1 后，捕获事件再次发生时，该标志位可以由硬件置 1。该标志位由软件清 0。<br>0：无捕获溢出中断发生。<br>1：发生了捕获溢出中断。 |
| 8:2 | 保留 | 必须保持复位值。 |
| 1 | CH0IF | 通道 0 比较/捕获中断标志<br>此标志由硬件置 1 软件清 0。当通道 0 在输入模式下时，捕获事件发生时此标志位被置 1；当通道 0 在输出模式下时，此标志位在一个比较事件发生时被置 1。<br>0：无通道 0 中断发生。<br>1：通道 0 中断发生。 |
| 0 | UPIF | 更新中断标志<br>此位在任何更新事件发生时由硬件置 1，软件清 0。<br>0：无更新中断发生。<br>1：发生更新中断。 |

### 软件事件产生寄存器(TIMERx_SWEVG)

地址偏移：0x14
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | | | CH0G | UPG |
| | | | | | | | | | | | | | | w | w |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |
| 1 | CH0G | 通道 0 捕获或比较事件发生<br>该位由软件置 1，用于在通道 0 产生一个捕获/比较事件，由硬件自动清 0。当此位被置 1，CH0IF 标志位被置 1，若开启对应的中断和 DMA，则发出相应的中断和 DMA 请求。此外，如果通道 0 配置为输入模式，计数器的当前值被 TIMERx_CH0CV 寄存器捕获，如果 CH0IF 标志位已经为 1，则 CH0OF 标志位被置 1。<br>0：不产生通道 0 捕获或比较事件。<br>1：发生通道 0 捕获或比较事件。 |
| 0 | UPG | 更新事件产生<br>此位由软件置 1，被硬件自动清 0。当此位被置 1 并且向上计数模式，计数器被清 0，预分频计数器将同时被清除。<br>0：无更新事件产生。 |

1：产生更新事件。

### 通道控制寄存器 0(TIMERx_CHCTL0)

地址偏移：0x18
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




**输出比较模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:7 | 保留 | 必须保持复位值。 |
| 6:4 | CH0COMCTL[2:0] | 通道 0 输出比较模式<br>此位定义了输出准备信号 O0CPRE 的输出比较模式，而 O0CPRE 决定了 CH0_O、CH0_ON 的值。另外，O0CPRE 高电平有效，而 CH0_O、CH0_ON 通道的极性取决于 CH0P、CH0NP 位。<br>000：时基。输出比较寄存器 TIMERx_CH0CV 与计数器 TIMERx_CNT 间的比较对 O0CPRE 不起作用<br>001：匹配时设置为高。当计数器的值与捕获/比较值寄存器 TIMERx_CH0CV 相同时，强制 O0CPRE 为高。<br>010：匹配时设置为低。当计数器的值与捕获/比较值寄存器 TIMERx_CH0CV 相同时，强制 O0CPRE 为低。<br>011：匹配时翻转。当计数器的值与捕获/比较值寄存器 TIMERx_CH0CV 相同时，强制 O0CPRE 翻转。<br>100：强制为低。强制 O0CPRE 为低电平<br>101：强制为高。强制 O0CPRE 为高电平<br>110：PWM 模式 0。在向上计数时，一旦计数器值小于 TIMERx_CH0CV 时，O0CPRE 为高电平，否则为低电平。在向下计数时，一旦计数器的值大于 TIMERx_CH0CV 时，O0CPRE 为低电平，否则为高电平。<br>111：PWM 模式 1。在向上计数时，一旦计数器值小于 TIMERx_CH0CV 时，O0CPRE 为低电平，否则为高电平。在向下计数时，一旦计数器的值大于 TIMERx_CH0CV 时，O0CPRE 为高电平，否则为低电平。<br>如果配置在 PWM 模式下，只有当输出比较模式从时基模式变为 PWM 模式或者比较结果改变时，O0CPRE 电平才改变。<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 且 CH0MS =00(比较模式)时此位不能被改变。 |

| | | |
|---|---|---|
| 3 | CH0COMSEN | 通道 0 输出比较影子寄存器使能<br>当此位被置 1，TIMERx_CH0CV 寄存器的影子寄存器被使能，影子寄存器在每次更新事件时都会被更新。<br>0：禁止通道 0 输出/比较影子寄存器<br>1：使能通道 0 输出/比较影子寄存器<br>仅在单脉冲模式下(SPM =1)，可以在未确认影子寄存器的情况下使用 PWM 模式<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 且 CH0MS =00 时此位不能被改变。 |
| 2 | CH0COMFEN | 通道 0 输出比较快速使能<br>0：通道 0 输出比较快速禁能。仅比较结果输入作为有效边沿时会产生比较匹配，并将 CH0_O 设置为比较电平，激活 CH0_O 输出的最小延时为 5 个时钟周期。<br>1：通道 0 输出比较快速使能。触发输入信号的有效边沿和比较结果都会产生比较匹配，并将 CH0_O 设置为比较电平。当触发信号的输入作为有效边沿时，激活 CH0_O 输出的最小延时为 3 个时钟周期。 |
| 1:0 | CH0MS[1:0] | 通道 0 I/O 模式选择<br>这些位定义了通道的工作模式和输入信号的选择。只有当通道关闭(TIMERx_CHCTL2 寄存器的 CH0EN 位被清 0)时这些位才可写。<br>00：通道 0 配置为输出<br>01：通道 0 配置为输入，IS0 映射在 CI0FE0 上<br>1x：保留 |

**输入捕获模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7:4 | CH0CAPFLT[3:0] | 通道 0 输入捕获滤波控制<br>CI0 输入信号可以通过数字滤波器进行滤波，该位域配置滤波参数。<br>数字滤波器的基本原理：根据 $f_{SAMP}$ 对 CI0 输入信号进行连续采样，并记录信号相同电平的次数。达到该位配置的滤波参数后，认为是有效电平。<br>滤波器参数配置如下： |

| CH0CAPFLT [3:0] | 采样次数 | $f_{SAMP}$ |
|---|---|---|
| 4'b0000 | 无滤波器 | |
| 4'b0001 | 2 | $f_{CK_TIMER}$ |
| 4'b0010 | 4 | |
| 4'b0011 | 8 | |
| 4'b0100 | 6 | $f_{DTS}/2$ |
| 4'b0101 | 8 | |
| 4'b0110 | 6 | $f_{DTS}/4$ |
| 4'b0111 | 8 | |
| 4'b1000 | 6 | $f_{DTS}/8$ |
| 4'b1001 | 8 | |
| 4'b1010 | 5 | $f_{DTS}/16$ |
| 4'b1011 | 6 | |
| 4'b1100 | 8 | |

| | | | |
|---|---|---|---|
| | | 4'b1101 | 5 | $f_{DTS}$/32 |
| | | 4'b1110 | 6 | |
| | | 4'b1111 | 8 | |

3:2 CH0CAPPSC[1:0] 通道 0 输入捕获预分频器

这 2 位定义了通道 0 输入的预分频系数。当 TIMERx_CHCTL2 寄存器中的 CH0EN =0 时，则预分频器复位。

00：无预分频器，捕获输入口上检测到的每一个边沿都触发一次捕获。

01：每 2 个事件触发一次捕获。

10：每 4 个事件触发一次捕获。

11：每 8 个事件触发一次捕获。

1:0 CH0MS[1:0] 通道 0 模式选择

与输出比较模式相同。

### 通道控制寄存器 2(TIMERx_CHCTL2)

地址偏移：0x20

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留.. | | | | | | | | | | | | CH0NP | 保留 | CH0P | CH0EN |
| | | | | | | | | | | | | rw | | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:4 | 保留 | 必须保持复位值。 |
| 3 | CH0NP | 通道 0 互补输出极性<br>当通道 0 配置为输出模式，此位定义了互补输出信号的极性。<br>0：通道0互补输出高电平为有效电平<br>1：通道0互补输出低电平为有效电平<br>当通道 0 配置为输入模式时，此位和 CH0P 联合使用，作为输入信号 CI0 的极性选择控制信号。<br>当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 或 10 时此位不能被更改。 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | CH0P | 通道 0 极性<br>当通道 0 配置为输出模式时，此位定义了输出信号极性。<br>0：通道0高电平为有效电平<br>1：通道0低电平为有效电平<br>当通道 0 配置为输入模式时，此位定义了 CI0 信号极性。<br>[CH0NP, CH0P] 将选择 CI0FE0 或者 CI1FE0 的有效边沿或者捕获极性。 |

[CH0NP==0, CH0P==0]：把 CIxFE0 的上升沿作为捕获或者从模式下触发的有效信号，并且 CIxFE0 不会被翻转。
[CH0NP==0, CH0P==1]：把 CIxFE0 的下降沿作为捕获或者从模式下触发的有效信号，并且 CIxFE0 会被翻转。
[CH0NP==1, CH0P==0]：保留。
[CH0NP==1, CH0P==1]：把 CIxFE0 的上升沿和下降沿都作为捕获或者从模式下触发的有效信号，并且 CIxFE0 不会被翻转。
当 TIMERx_CCHP 寄存器的 PROT [1:0]=11 或 10 时此位不能被更改。

| | | |
|---|---|---|
| 0 | CH0EN | 通道 0 捕获/比较使能<br>当通道 0 配置为输出模式时，将此位置 1 使能 CH0_O 信号有效。当通道 0 配置为输入模式时，将此位置 1 使能通道 0 上的捕获事件。<br>0：禁止通道 0。<br>1：使能通道 0。 |

### 计数器寄存器(TIMERx_CNT)

地址偏移：0x24
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CNT[15:0] | 这些位是当前的计数值。写操作能改变计数器值。 |

### 预分频寄存器(TIMERx_PSC)

地址偏移： 0x28
复位值： 0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | PSC[15:0] | 计数器时钟预分频值<br>计数器时钟等于 TIMER_CK 时钟除以(PSC+1)，每次当更新事件产生时，PSC 的值被装入到对应的影子寄存器。 |

### 计数器自动重载寄存器(TIMERx_CAR)

地址偏移：0x2C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CARL[15:0] | 计数器自动重载值<br>这些位定义了计数器的自动重载值。 |

### 通道 0 捕获/比较值寄存器(TIMERx_CH0CV)

地址偏移：0x34
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| CH0VAL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

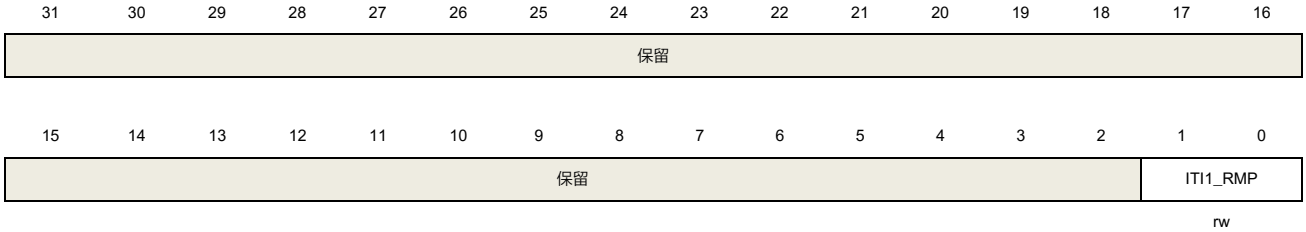
| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CH0VAL[15:0] | 通道 0 的捕获或比较值<br>当通道 0 配置为输入模式时，这些位决定了上次捕获事件的计数器值。并且本寄存器为只读。 |

当通道 0 配置为输出模式时，这些位包含了即将和计数器比较的值。使能相应影子寄存器后，影子寄存器值随每次更新事件更新。

### 输入重映射寄存器(TIMERx_IRMP)(x=10)

地址偏移：0x50
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

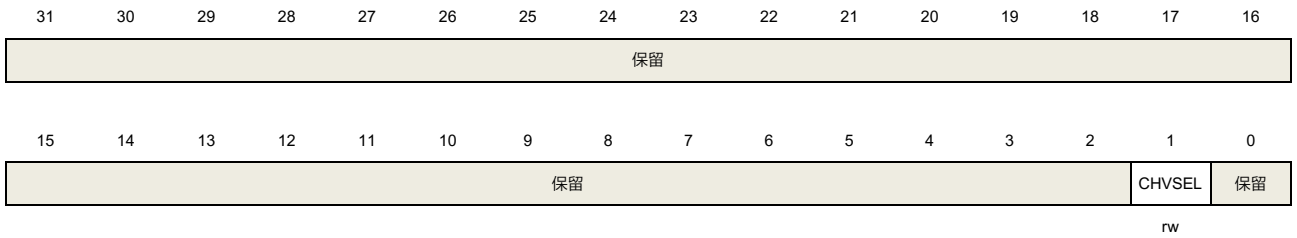



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |
| 1:0 | ITI1_RMP | 内部触发 1 重映射<br>00：根据 GPIO 设置。<br>01：根据 GPIO 设置。<br>10：HXTAL_DIV(RTC 时钟，是 HXTAL 时钟经过 RCU_CFG0 寄存器中 RTCDIV 的位分频后得到)。<br>11：根据 GPIO 设置。 |

### 配置寄存器(TIMERx_CFG)

地址偏移：0xFC
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |
| 1 | CHVSEL | 写捕获比较寄存器选择位<br>此位由软件写 1 或清 0。<br>1：当写入捕获比较寄存器的值与寄存器当前值相等时，写入操作无效。 |

| | | |
|---|---|---|
| | | 0：无影响。 |
| 0 | 保留 | 必须保持复位值。 |

# 18.5. 基本定时器(TIMERx, x=5,6)

## 18.5.1. 简介

基本定时器(Timer5, 6)包含一个无符号 16 位计数器。可以被用作通用定时器和为 DAC(数字到模拟转换器)提供时钟。基本定时器可以配置产生 DMA 请求，TRGO 触发连接到 DAC。

## 18.5.2. 主要特性

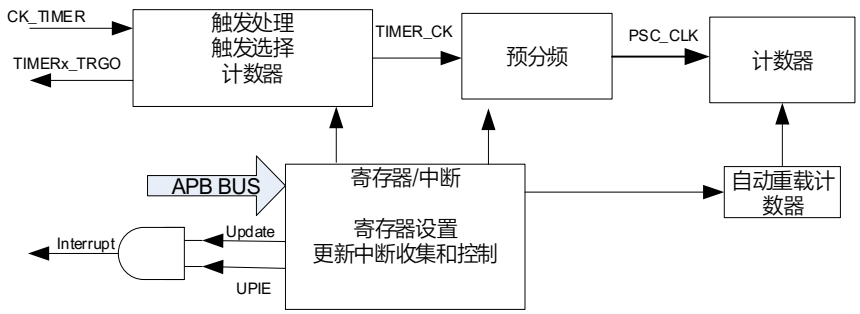
- 计数器宽度：16位；
- 时钟源只有内部时钟；
- 计数模式：向上计数；
- 可编程的预分频器：16位，运行时可以被改变；
- 自动重装载功能；
- 中断输出和DMA请求：更新事件。

## 18.5.3. 功能描述

### 结构框图

*图 18-63. 基本定时器结构框图*提供了基本定时器内部配置的细节。

**图 18-63. 基本定时器结构框图**

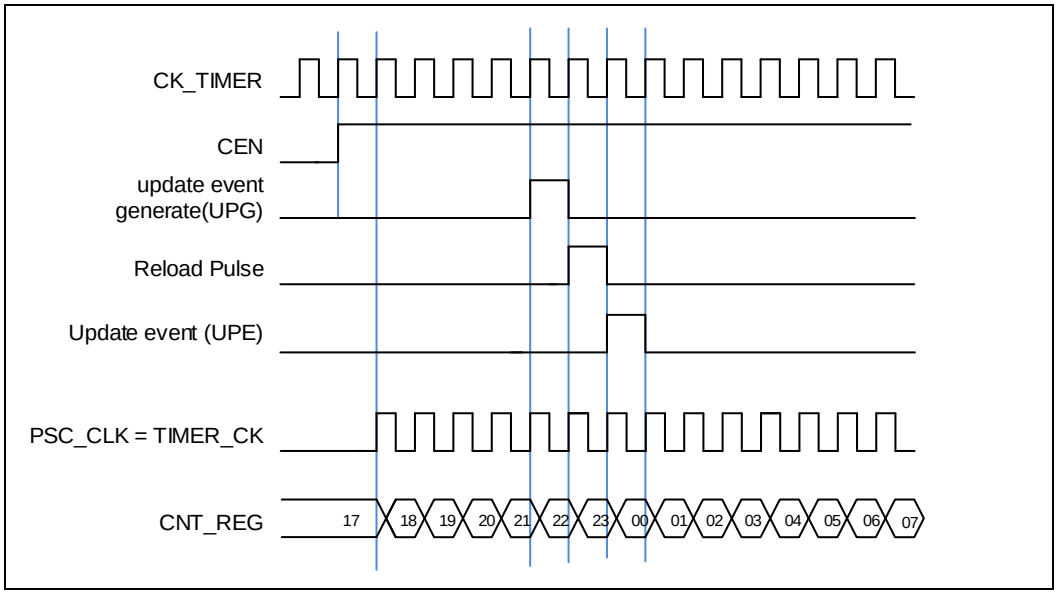



### 时钟源配置

基本定时器可以由内部时钟源 CK_TIMER 驱动 。

基本定时器仅有一个时钟源 CK_TIMER，用来驱动计数器预分频器。当 CEN 置位，CK_TIMER 经过预分频器(预分频值由 TIMERx_PSC 寄存器确定)产生 PSC_CLK。

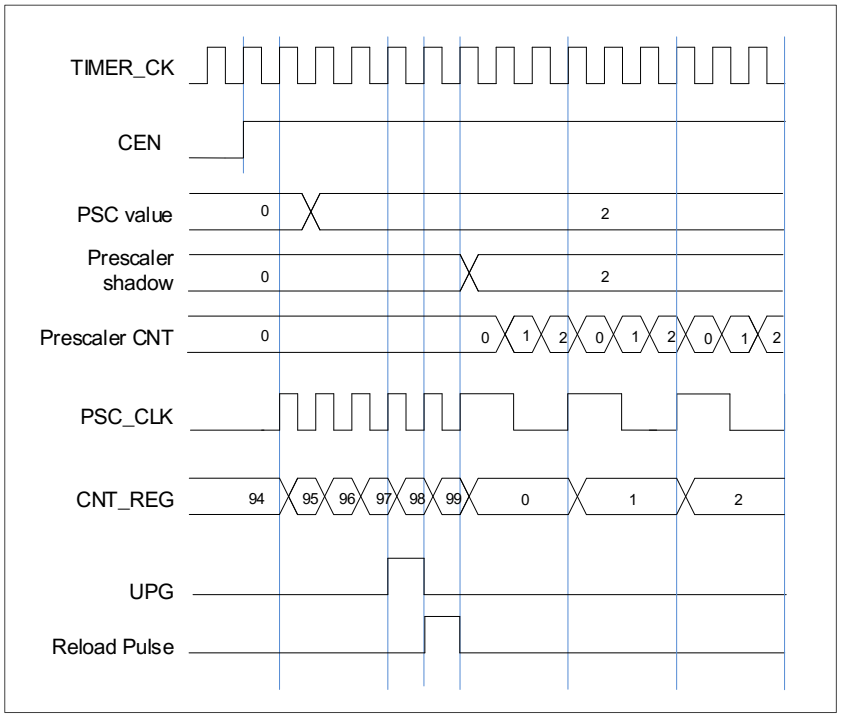
**图 18-64. 内部时钟分频为 1 时，计数器的时序图**




## 时钟预分频器

预分频器可以将定时器的时钟(TIMER_CK)频率按 1 到 65536 之间的任意值分频，分频后的时钟 PSC_CLK 驱动计数器计数。分频系数受预分频寄存器 TIMERx_PSC 控制，这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下一次更新事件到来时被采用。

**图 18-65. 当 PSC 数值从 0 变到 2 时，计数器的时序图**



### 计数器向上计数模式

在这种模式，计数器的计数方向是向上计数。计数器从 0 开始向上连续计数到自动加载值(定义在 TIMERx_CAR 寄存器中)，一旦计数器计数到自动加载值，会重新从 0 开始向上计数并产生上溢事件。在向上计数模式中，TIMERx_CTL0 寄存器中的计数方向控制位 DIR 应该被设置成 0。

当通过 TIMERx_SWEVG 寄存器的 UPG 位置 1 来设置更新事件时，计数值会被清 0，并产生更新事件。

如果 TIMERx_CTL0 寄存器的 UPDIS 置 1，则禁止更新事件。

当发生更新事件时，所有影子寄存器(计数器自动重载寄存器，预分频寄存器)都将被更新。

下面这些图给出了一些例子，当 TIMERx_CAR=0x99 时，计数器在不同预分频因子下的行为。

**图 18-66. 向上计数时序图，PSC=0/2**

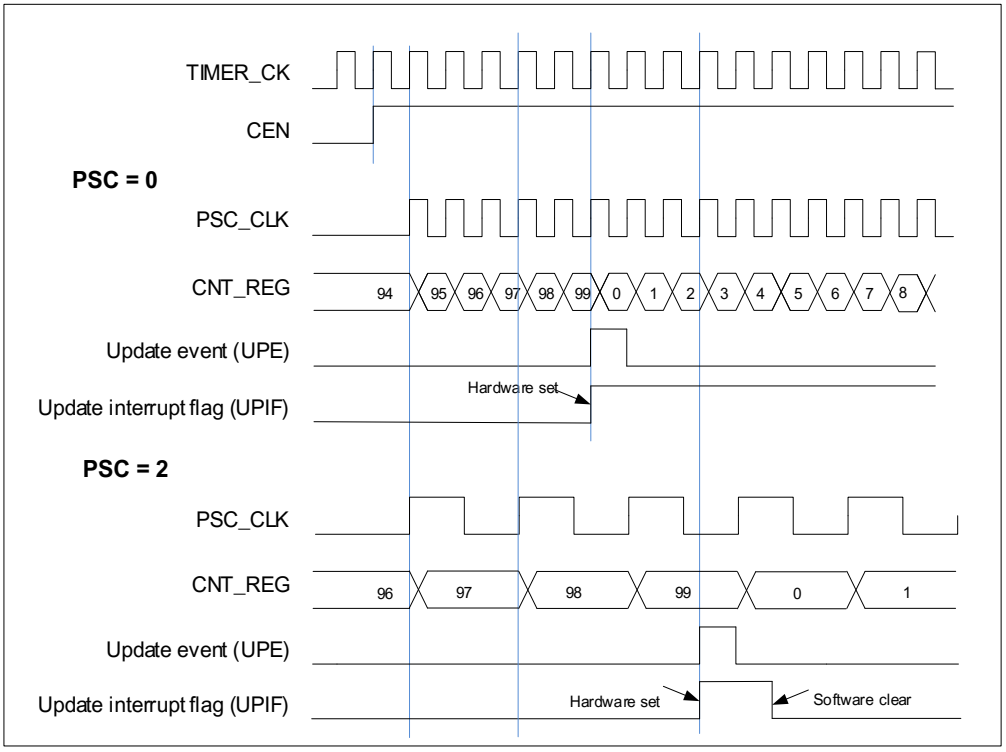
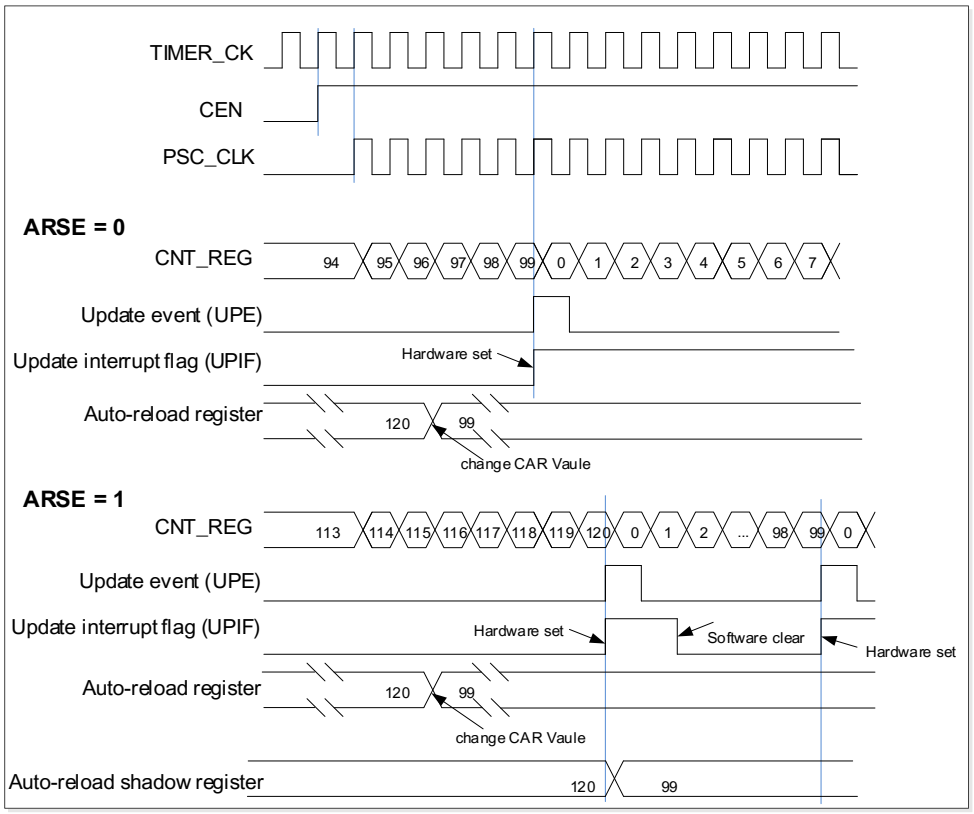


图 18-67. 向上计数时序图，在运行时改变 TIMERx_CAR 寄存器的值




## 单脉冲模式

单脉冲模式与重复模式是相反的，设置TIMERx_CTL0寄存器的SPM位置1，则使能单脉冲模式。当SPM置1，计数器在下次更新事件到来后清零并停止计数。

一旦设置定时器运行在单脉冲模式下，需要设置TIMERx_CTL0寄存器的定时器使能位CEN=1来使能计数器，此后CEN位一直保持为1直到更新事件发生或者CEN位被软件写0。如果CEN位被软件清0，计数器停止工作，计数值被保持。

## 定时器调试模式

当Cortex®-M4内核停止，DBG_CTL2寄存器中的TIMERx_HOLD配置位被置1，定时器计数器停止。

## 18.5.4. TIMERx 寄存器(x=5,6)

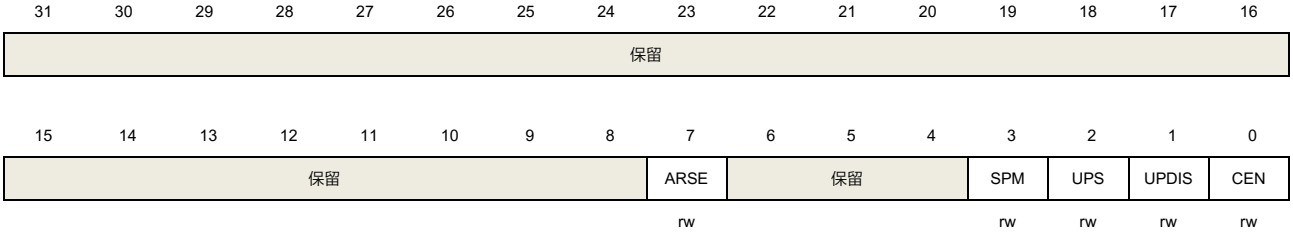
TIMER5 基地址:0x4000 1000

TIMER6 基地址:0x4000 1400

### 控制寄存器 0(TIMERx_CTL0)

地址偏移：0x00
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7 | ARSE | 自动重载影子使能<br>0：禁能 TIMERx_CAR 寄存器的影子寄存器。<br>1：使能 TIMERx_CAR 寄存器的影子寄存器。 |
| 6:4 | 保留 | 必须保持复位值。 |
| 3 | SPM | 单脉冲模式<br>0：单脉冲模式禁能。更新事件发生后，计数器继续计数<br>1：单脉冲模式使能。在下一次更新事件发生时，计数器停止计数 |
| 2 | UPS | 更新请求源<br>软件配置该位，选择更新事件源.<br>0：以下事件均会产生更新中断或DMA请求：<br>UPG位被置1<br>计数器溢出/下溢<br>复位模式产生的更新<br>1：下列事件会产生更新中断或DMA请求：<br>计数器溢出/下溢 |
| 1 | UPDIS | 禁止更新.<br>该位用来使能或禁能更新事件的产生<br>0：更新事件使能. 更新事件发生时，相应的影子寄存器被装入预装载值，以下事件均会产生更新事件：<br>UPG位被置1<br>计数器溢出/下溢 |

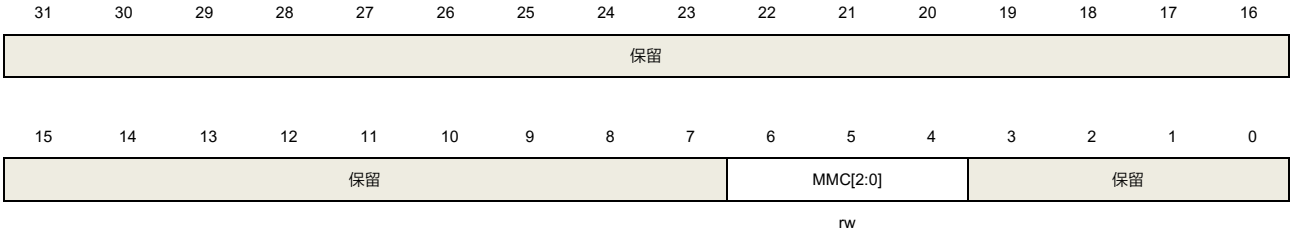
复位模式产生的更新
1：更新事件禁能.
注意：当该位被置 1 时，UPG 位被置 1 或者复位模式不会产生更新事件，但是计数器和预分频器被重新初始化

| | | |
|---|---|---|
| 0 | CEN | 计数器使能。<br>0: 计数器禁能。<br>1: 计数器使能。<br>在软件将 CEN 位置 1 后，外部时钟、暂停模式和译码器模式才能工作。 |

### 控制寄存器 1(TIMERx_CTL1)

地址偏移：0x04
复位值：0x0000 0000

该寄存器只能按字(32位)访问。



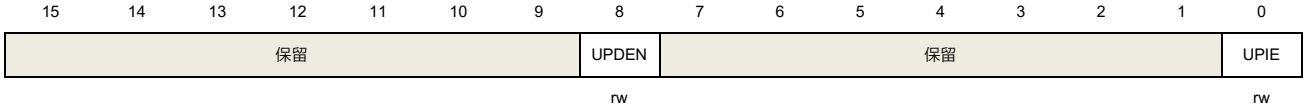

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:7 | 保留 | 必须保持复位值。 |
| 6:4 | MMC[2:0] | 这些位控制 TRGO 信号的选择，TRGO 信号由主定时器发给从定时器用于同步功能<br>000：当产生一个定时器复位事件后，输出一个TRGO信号，定时器复位源为：<br>主定时器产生一个复位事件<br>TIMERx_SWEVG寄存器中UPG位置1<br>001：当产生一个定时器使能事件后，输出一个TRGO信号，定时器使能源为：<br>CEN位置1<br>在暂停模式下，触发输入置1<br>010：当产生一个定时器更新事件后，输出一个TRGO信号，更新事件源由UPDIS和UPS位决定 |
| 3:0 | 保留 | 必须保持复位值。 |

### DMA 和中断使能寄存器(TIMERx_DMAINTEN)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。



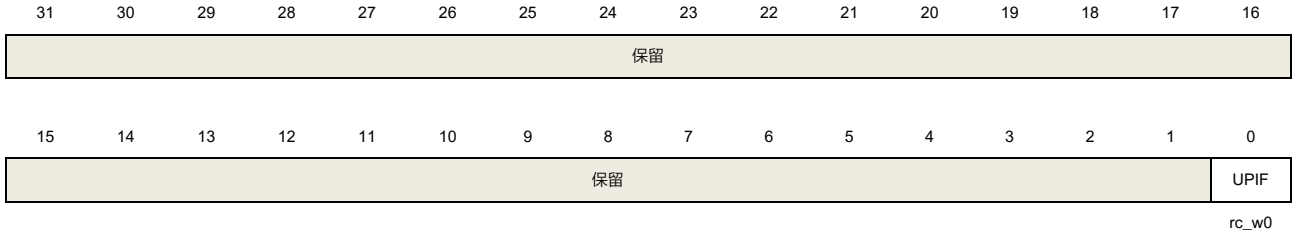



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:9 | 保留 | 必须保持复位值。 |
| 8 | UPDEN | 更新 DMA 请求使能<br>0：禁止更新 DMA 请求。<br>1：使能更新 DMA 请求。 |
| 7:1 | 保留 | 必须保持复位值。 |
| 0 | UPIE | 更新中断使能<br>0：禁止更新中断。<br>1：使能更新中断。 |

### 中断标志寄存器(TIMERx_INTF)

地址偏移：0x10
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:1 | 保留 | 必须保持复位值。 |
| 0 | UPIF | 更新中断标志<br>此位在任何更新事件发生时由硬件置 1，软件清 0。<br>0：无更新中断发生。<br>1：发生更新中断。 |

### 软件事件产生寄存器(TIMERx_SWEVG)

地址偏移：0x14
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 保留 |
|---|

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | | | | UPG |
| | | | | | | | | | | | | | | | w |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:1 | 保留 | 必须保持复位值。 |
| 0 | UPG | 更新事件产生<br>此位由软件置 1，被硬件自动清 0。当此位被置 1 并且向上计数模式，计数器被清 0，预分频计数器将同时被清除。<br>0：无更新事件产生。<br>1：产生更新事件。 |

### 计数器寄存器(TIMERx_CNT)

地址偏移：0x24
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CNT[15:0] | 这些位是当前的计数值。写操作能改变计数器值。 |

### 预分频寄存器(TIMERx_PSC)

地址偏移： 0x28
复位值： 0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | PSC[15:0] | 计数器时钟预分频值<br>计数器时钟等于 TIMER_CK 时钟除以(PSC+1)，每次当更新事件产生时，PSC 的值被装入到对应的影子寄存器。 |

### 计数器自动重载寄存器(TIMERx_CAR)

地址偏移：0x2C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CARL[15:0] | 这些位定义了计数器的自动重载值。 |

# 19. 通用同步异步收发器(USART)

## 19.1. 简介

通用同步异步收发器(USART)提供了一个灵活方便的串行数据交换接口，数据帧可以通过全双工或半双工，同步或异步的方式进行传输。USART提供了可编程的波特率发生器，能对UCLK(PCLK1, PCLK2)进行分频产生USART发送和接收所需的特定频率。

USART不仅支持标准的异步收发模式，还实现了一些其他类型的串行数据交换模式，如红外编码规范，SIR，智能卡协议，LIN，以及同步单双工模式。它还支持多处理器通信和Modem流控操作(CTS/RTS)。数据帧支持从LSB或者MSB开始传输。数据位的极性和TX/RX引脚都可以灵活配置。

所有USART都支持DMA功能，以实现高速率的数据通信。

## 19.2. 主要特性

- NRZ标准格式
- 全双工异步通信
- 半双工单线通信
- 可编程的波特率产生器：
  - 由外设时钟分频产生，其中USART0/5由PCLK2分频得到，USART1/2和UART3/4/6/7由PCLK1分频得到；
  - 8或16倍过采样；
  - 当时钟频率为120M，过采样为8，最高速度可到15MBits/s
- 完全可编程的串口特性：
  - 偶校验位，奇校验位，无校验位的生成/检测
  - 数据位(8或9位)；
  - 产生0.5，1，1.5或者2个停止位
- 发送器和接收器可分别使能。
- 支持硬件Modem流控操作(CTS/RTS)
- DMA访问数据缓冲区
- LIN断开帧的产生和检测
- 支持红外数据协议(IrDA)
- 同步传输模式以及为同步传输输出发送时钟
- 支持兼容ISO7816-3的智能卡接口：
  - 字节模式(T=0)
  - 块模式(T=1)
  - 直接和反向转换
- 多处理器通信：
  - 如果地址不匹配，则进入静默模式
  - 通过线路空闲检测或者地址匹配检测从静默模式唤醒
- 多种状态标志：

- 传输检测标志：接收缓冲区不为空(RBNE)，发送缓冲区为空(TBE)，传输完成(TC)，忙(BSY)
- 错误检测标志：过载错误(ORERR)，噪声错误(NERR)，帧格式错误(FERR)，奇偶校验错误(PERR)
- 硬件流控操作标志：CTS变化(CTSF)
- LIN模式标志：LIN断开检测(LBDF)
- 多处理器通信模式标志：IDLE帧检测(IDLEF)
- 智能卡模式标志：块结束(EBF)和接收超时(RTF)
- 若相应的中断使能，这些事件发生将会触发中断

USART0/1/2/5完全实现上述功能，但是UART3/4/6/7只实现了上面所介绍功能的部分，下面这些功能在UART3/4/6/7中没有实现：

- 智能卡模式
- 同步模式
- 硬件流操作(CTS/RTS)
- 设置数据极性

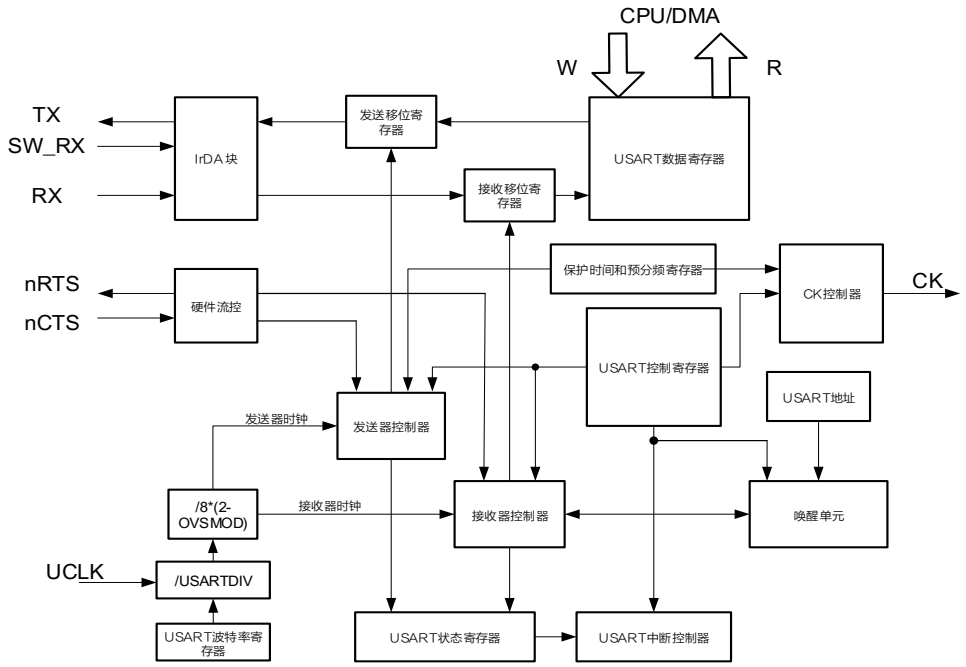
## 19.3. 功能描述

USART接口通过*表19-1. USART重要引脚描述*中主要引脚从外部连接到其他设备。

**表 19-1. USART 重要引脚描述**

| 引脚 | 类型 | 描述 |
|---|---|---|
| RX | 输入 | 接收数据 |
| TX | 输出<br>I/O(单线模式/智能卡模式) | 发送数据。当 USART 使能后，若无数据发送，默认为高电平 |
| CK | 输出 | 用于同步通信的串行时钟信号 |
| nCTS | 输入 | 硬件流控模式发送使能信号 |
| nRTS | 输出 | 硬件流控模式发送请求信号 |

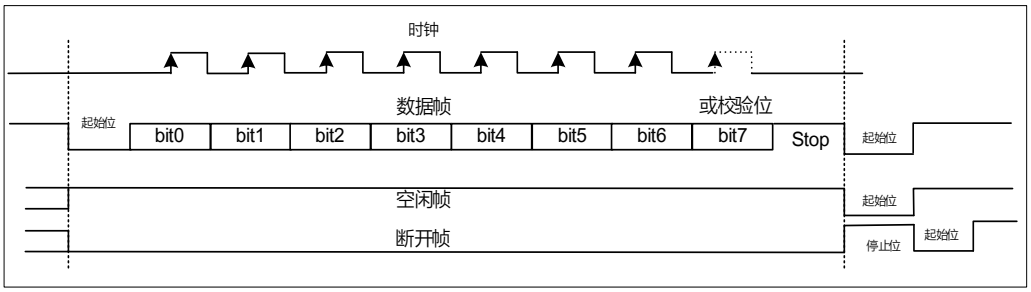
图 19-1. USART 模块内部框图




## 19.3.1. USART 帧格式

USART数据帧开始于起始位，结束于停止位。USART_CTL0寄存器中WL位可以设置数据长度。将USART_CTL0寄存器中PCEN置位，最后一个数据位可以用作校验位。若WL位为0，第七位为校验位。若WL位置1，第八位为校验位。USART_CTL0寄存器中PM位用于选择校验位的计算方法。

图 19-2. USART 字符帧(8 数据位和 1 停止位)




在发送和接收中，停止位可以由USART_CTL1寄存器中STB[1:0]位域配置。

表 19-2. 停止位配置

| STB[1:0] | 停止位长度(位) | 功能描述 |
|---|---|---|
| 00 | 1 | 默认值 |
| 01 | 0.5 | 智能卡模式接收 |
| 10 | 2 | 标准 USART，单线以及调制解调模式 |
| 11 | 1.5 | 智能卡模式发送和接收 |

在一个空闲帧中，所有位都为1。数据帧长度与正常USART数据帧长度相同。

紧随停止位后多个低电平为中断帧。USART数据帧的传输速度由UCLK时钟频率，波特率发生器的配置，以及过采样模式共同决定。

### 19.3.2. 波特率发生

波特率分频系数是一个16位的数字，包含12位整数部分和4位小数部分。波特率发生器使用这两部分组合所得的数值来确定波特率。由于具有小数部分的波特率分频系数，将使USART能够产生所有标准波特率。

波特率分频系数(USARTDIV)与UCLK具有如下关系：

如果过采样率是16，公式为：

$$\text{USARTDIV} = \frac{\text{UCLK}}{16 \times \text{Baud Rate}} \tag{19-1}$$

如果过采样是8，公式为：

$$\text{USARTDIV} = \frac{\text{UCLK}}{8 \times \text{Baud Rate}} \tag{19-2}$$

例如，当过采样是16：

1. 由USART_BAUD寄存器的值得到USARTDIV：
   假设USART_BAUD=0x21D，则INTDIV=33(0x21)，FRADIV=13(0xD)。
   UASRTDIV=33+13/16=33.81。
2. 由USARTDIV得到USART_BAUD寄存器的值：
   假设要求UASRTDIV=30.37，INTDIV=30(0x1E)。
   16*0.37=5.92，接近整数6，所以FRADIV=6(0x6)。
   USART_BAUD=0x1E6。

**注意：**若取整后FRADIV=16(溢出)，则进位必须加到整数部分。

### 19.3.3. USART 发送器

如果USART_CTL0寄存器的发送使能位(TEN)被置位，当发送数据缓冲区不为空时，发送器将会通过TX引脚发送数据帧。TX引脚的极性可以通过USART_CTL3寄存器中TINV位来配置。时钟脉冲通过CK引脚输出。

TEN置位后发送器会发出一个空闲帧。TEN位在数据发送过程中是不可以被复位的。

系统上电后，TBE默认为高电平。在USART_STAT0寄存器中TBE置位时，数据可以在不覆盖前一个数据的情况下写入USART_DATA寄存器。当数据写入USART_DATA寄存器，TBE位将被清0。在数据由USART_DATA移入移位寄存器后，该位由硬件置1。如果数据在一个发送过程正在进行时被写入USART_DATA寄存器，它将首先被存入发送缓冲区，在当前发送过程完成时传输到发送移位寄存器中。如果数据在写入USART_DATA寄存器时，没有发送过程正在进行，TBE位将被清零然后迅速置位，原因是数据将立刻传输到发送移位寄存器。

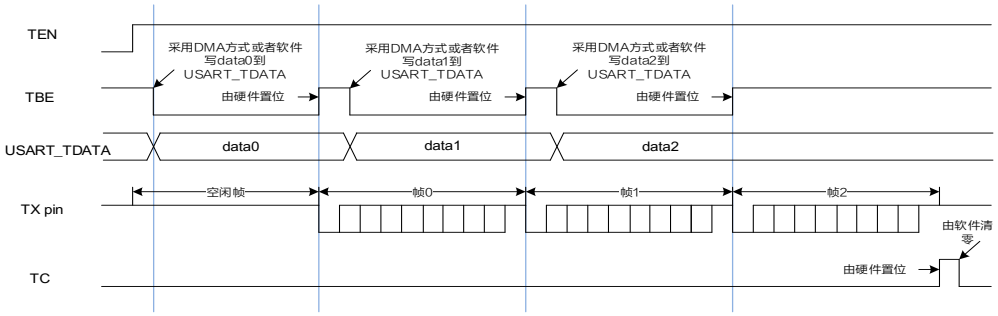
假如一帧数据已经发送出去，并且TBE位已经置位，那么USART_STAT0寄存器中TC位将被置

1。如果USART_CTL0寄存器中的中断使能位(TCIE)为1，将会产生中断。

*图19-3. USART发送步骤*给出了USART发送步骤。软件操作按以下流程进行：

1. 通过USART_CTL0寄存器的WL设置字长；
2. 在USART_CTL1寄存器中写STB[1:0]位来设置停止位的长度；
3. 如果选择了多级缓存通信方式，应该在USART_CTL2寄存器中使能DMA(DENT位)；
4. 在USART_BAUD寄存器中设置波特率；
5. 在USART_CTL0寄存器中置位UEN位，使能USART；
6. 在USART_CTL0寄存器中设置TEN位；
7. 等待TBE置位；
8. 向USART_DATA寄存器写数据；
9. 若DMA未使能，每发送一个字节都需重复步骤7-8；
10. 等待TC=1，发送完成。

**图 19-3. USART 发送步骤**




在禁用USART或进入低功耗状态之前，必须等待TC置位。先读USART_STAT0然后再写USART_DATA可将TC位清0。在多级缓存通信方式(DENT=1)下，直接向TC写0，也能清TC。

### 19.3.4. USART 接收器

上电后，USART接收器使能按以下步骤进行：

1. 在USART_CTL0寄存器中置位UEN位，使能USART；
2. 写USART_CTL0寄存器的WL去设置字长；
3. 在USART_CTL1寄存器中写STB[1:0]位来设置停止位的长度；
4. 如果选择了多级缓存通信方式，应该在USART_CTL2寄存器中使能DMA(DENR位)；
5. 在USART_BAUD寄存器中设置波特率；
6. 在USART_CTL0中设置REN位。

接收器在使能后若检测到一个有效的起始脉冲便开始接收码流。在接收一个数据帧的过程中会检测噪声错误，奇偶校验错误，帧错误和过载错误。

当接收到一个数据帧，USART_STAT0寄存器中的RBNE置位，如果设置了USART_CTL0寄存器中相应的中断使能位RBNEIE，将会产生中断。在USART_STAT0寄存器中可以观察接收状态标志。

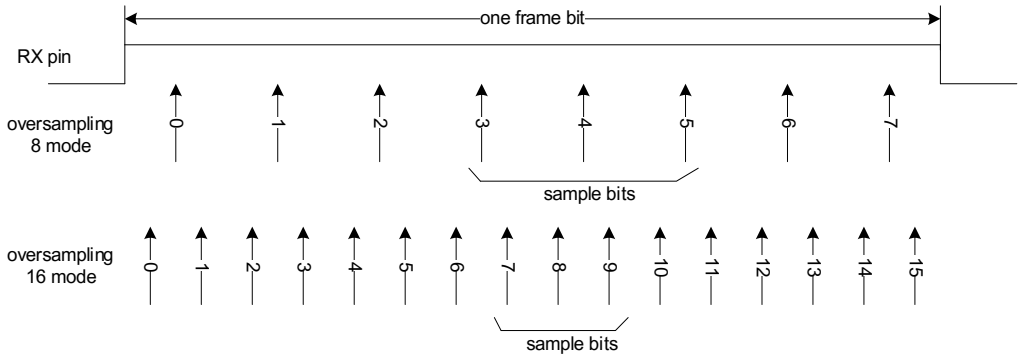
软件可以通过读USART_DATA寄存器或者DMA方式获取接收到的数据。不管是直接读寄存器

还是通过DMA，只要是对USART_DATA寄存器的一个读操作都可以清除RBNE位。

在接收过程中，需使能REN位，不然当前的数据帧将会丢失。

在默认情况下，接收器通过获取三个采样点的值来估计该位的值。如果是8倍过采样模式，选择第3、4、5个采样点；如果是16倍过采样模式，选择第7、8、9个采样点。如果在3个采样点中有2个或3个为0，该数据位被视为0，否则为1。如果3个采样点中有一个采样点的值与其他两个不同，不管是数据位，奇偶校验位或者停止位，都将产生噪声错误(NERR)。如果使能DMA，并置位USART_CTL2寄存器中ERRIE，将会产生中断。如果在USART_CTL2中置位OSB，接收器将仅获取一个采样点来估计一个数据位的值。在这种情况下将不会检测到噪声错误。

**图 19-4. 过采样方式接收一个数据位(OSB=0)**




通过置位USART_CTL0寄存器中的PCEN位使能奇偶校验功能，接收器在接收一个数据帧时计算预期奇偶校验值，并将其与接收到的奇偶校验位进行比较。如果不相等，USART_STAT0寄存器中PERR被置位。如果设置了USART_CTL0寄存器中的PERRIE位，将产生中断。

如果在停止位传输过程中RX引脚为0，将产生帧错误，USART_STAT0寄存器中FERR置位。如果使能DMA并置位USART_CTL2寄存器中ERRIE位，将产生中断。

当接收到一帧数据，而RBNE位还没有被清零，随后的数据帧将不会存储在数据接收缓冲区中。USART_STAT0寄存器中的溢出错误标志位ORERR将置位。如果使能DMA并置位USART_CTL2寄存器中ERRIE位或者置位RBNEIE，将产生中断。

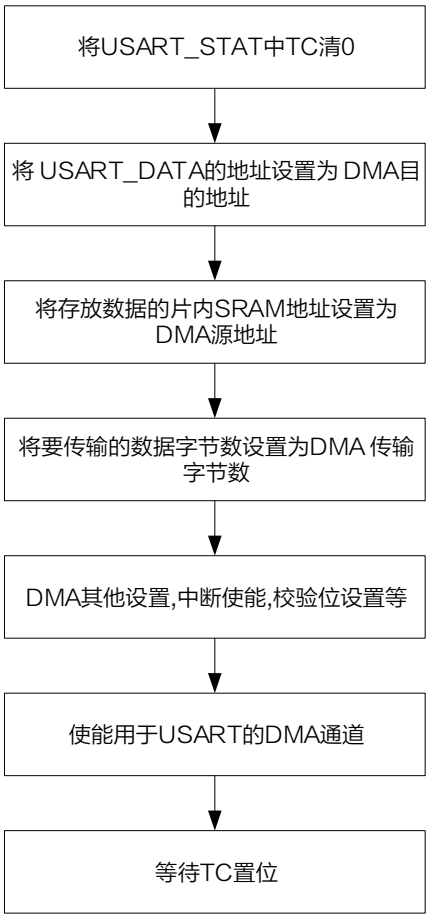
若接收过程中，产生了噪声错误(NERR)、校验错误(PERR)、帧错误(FERR)或溢出错误(ORERR)，则NERR、PERR、FERR或ORERR将和RBNE同时置位。如果没有使能DMA，RBNE中断发生时，软件需检查是否有噪声错误、校验错误、帧错误或溢出错误产生。

### 19.3.5. DMA 方式访问数据缓冲区

为减轻处理器的负担，可以采用DMA访问发送缓冲区或者接收缓冲区。置位USART_CTL2寄存器中DENT位可以使能DMA发送，置位USART_CTL2寄存器中DENR位可以使能DMA接收。

当DMA用于USART发送时，DMA将数据从片内SRAM传送到USART的数据缓冲区。配置步骤如*图19-5. 采用DMA方式实现USART数据发送配置步骤*所示。

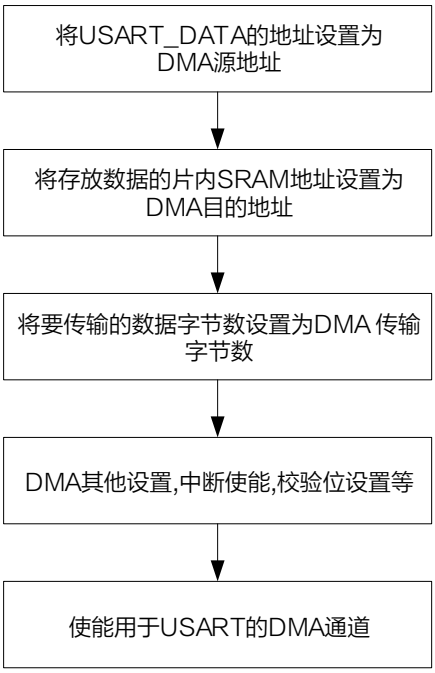
**图 19-5. 采用 DMA 方式实现 USART 数据发送配置步骤**




所有数据帧都传输完成后，USART_STAT0寄存器中TC位置1。如果USART_CTL0寄存器中TCIE置位，将产生中断。

当DMA用于USART接收时，DMA将数据从接收缓冲区传送到片内SRAM。配置步骤如*图 19-6. 采用DMA方式实现USART数据接收配置步骤*所示。如果将USART_CTL2寄存器中ERRIE位置1，USART_STAT0寄存器中的错误标志位(FERR、ORERR和NERR)被置位时将产生中断。

**图 19-6. 采用 DMA 方式实现 USART 数据接收配置步骤**

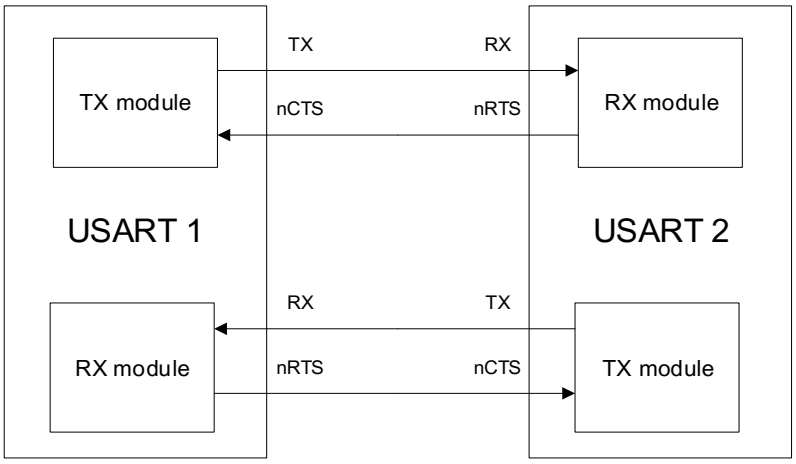



当USART接收到的数据数量达到了DMA传输数据数量，DMA模块将产生传输完成中断。

### 19.3.6. 硬件流控制

硬件流控制功能通过nCTS和nRTS引脚来实现。通过将USART_CTL2寄存器中RTSEN位置1来使能RTS流控，将USART_CTL2寄存器中CTSEN位置1来使能CTS流控。

**图 19-7. 两个 USART 之间的硬件流控制**



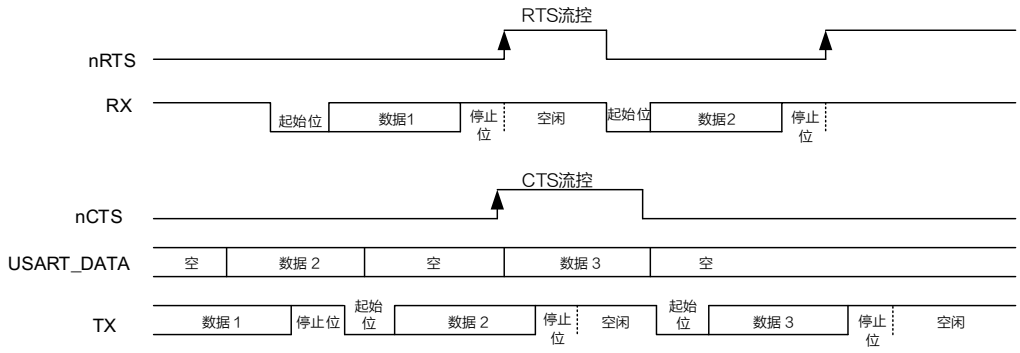

#### RTS 流控

USART接收器输出nRTS，它用于反映接收缓冲区状态。当一帧数据接收完成，nRTS变成高电平，这样是为了阻止发送器继续发送下一帧数据。当接收缓冲区满时，nRTS保持高电平，可以通过读USART_DATA寄存器来清零。

### CTS 流控

USART发送器监视nCTS输入引脚来决定数据帧是否可以发送。如果USART_STAT0寄存器中TBE位是0且nCTS为低电平，发送器发送数据帧。在发送期间，若nCTS信号变为高电平，发送器将会在当前数据帧发送完成后停止发送。

**图 19-8. 硬件流控制**




如果CTS流控制被使能，在nCTS引脚信号发生变化时，USART_STAT0寄存器中CTSF位会置1。如果USART_CTL2寄存器中的CTSIE位被置位，将会产生中断。

## 19.3.7. 多处理器通信

在多处理器通信中，多个USART被连接成一个网络。对于一个设备来说，监视所有来自RX引脚的消息，是一种巨大的负担。为减轻设备负担，软件可以通过将USART_CTL0寄存器中RWU位置1使一个USART进入静默模式。

如果USART处于静默模式，所有的接收状态标志位将不会被置位。软件可以通过对RWU清零来唤醒USART。

此外，USART可以由硬件用以下两种方式中的一种来唤醒：空闲总线检测和地址匹配检测。

设备默认使用空闲总线检测方法唤醒USART。如果RWU位为0，RX引脚检测到空闲帧，USART_STAT寄存器中的IDLEF位会置位。如果RWU位置位，RX引脚检测到空闲帧时，硬件会将RWU清零，从而退出静默模式，当它是被空闲帧唤醒时，USART_STAT寄存器中IDLEF位不会被置1。

当USART_CTL0寄存器中WM被置位，数据最高位会被认为是地址标志位。如果地址标志位为1，该字节被认为是地址字节。如果地址字节的低4位与USART_CTL1寄存器中的ADDR[3:0]相同，硬件会将RWU清零，并退出静默模式。接收到将USART唤醒的数据帧，RBNE将置位。状态标志可以从USART_STAT0寄存器中获取。如果地址字节的低4位与USART_CTL1寄存器中的ADDR[3:0]不相同，硬件会置位RWU并进入静默模式。在这种情况下，RBNE不会被置位。

如果采用地址匹配检测，默认情况下，接收器对地址字节不做奇偶校验。如果USART_CHC寄存器中PCM位以及USART_CTL0寄存器中PCEN位被置位。地址字节最高位被视为校验位，其余位被视为地址。

### 19.3.8. LIN 模式

将USART_CTL1寄存器的LMEN置位即可使能本地互联网络模式。

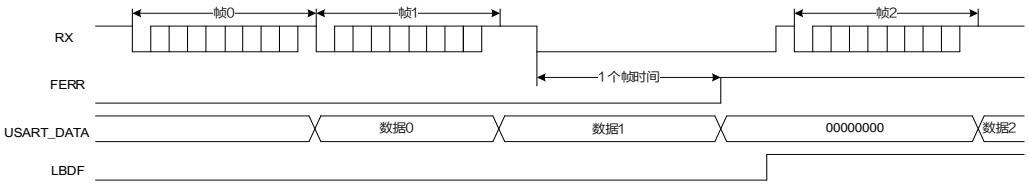
在LIN模式下，USART_CTL1寄存器中CKEN，WL，STB[1:0]以及USART_CTL2的SCEN，HDEN，IREN位都应该被清0。

在发送一个普通数据帧时，LIN发送过程与普通发送过程相同。当USART_CTL0寄存器中SBKCMD置位时，USART在发送完一个停止位后会连续发送13个0。

断开检测功能完全独立于普通USART接收器。因此，断开检测可以是在空闲状态下，也可以在数据传输过程中。USART_CTL1寄存器中LBLEN位可以选择断开帧长度。如果在RX引脚检测到大于或等于与预期断开帧长度相等数量的0(LBLEN=0时，10个0；LBLEN=1时，11个0)，USART_STAT0寄存器中LBDF置位。如果USART_CTL1寄存器中LBDIE被置位，将产生中断。

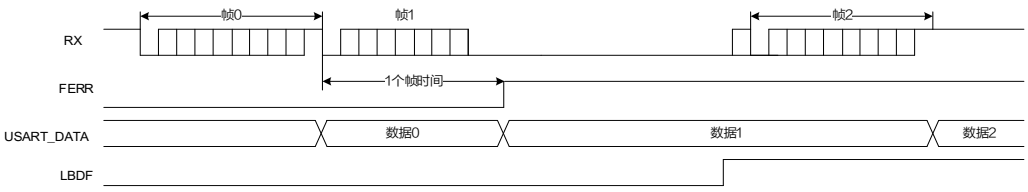
如*图19-9. 空闲状态下检测断开帧*所示，如果断开帧发生在空闲状态下，USART接收器会接收到一个全0数据帧，同时FERR置位。

**图 19-9. 空闲状态下检测断开帧**




如*图19-10. 数据传输过程中检测断开帧*所示，如果断开帧发生在数据传输过程中，当前传输帧发生错误，FERR置位。

**图 19-10. 数据传输过程中检测断开帧**




### 19.3.9. 同步通信模式

USART支持主机模式下的全双工同步串行通信，可以通过置位USART_CTL1的CKEN位来使能。在同步模式下，USART_CTL1的LMEN和USART_CTL2的SCEN，HDEN，IREN位应该被清0。CK引脚作为USART同步发送器的时钟输出，仅当TEN位被使能时，它才被激活。在起始位和停止位传送期间，不会从CK引脚输出时钟脉冲。USART_CTL1的CLEN位用来决定在最低位(地址索引位)发送期间是否有时钟信号输出。USART_CTL1的CPH位用来决定数据在第一个时钟沿被采样还是在第二个时钟沿被采样。USART_CTL1的CPL位用来决定在USART同步模

式空闲状态下，时钟引脚的电平。

CK引脚输出波形由USART_CTL1寄存器中CPL，CPH，CLEN位决定。软件仅在USART禁用(UEN=0)时才可以改变它们的值。

如果USART_CTL0寄存器中REN置位，接收器的工作方式与普通模式下接收方式是不同的。接收器在时钟捕获沿采样数据，并无任何过采样。

**图 19-11. 同步模式下的 USART 示例**

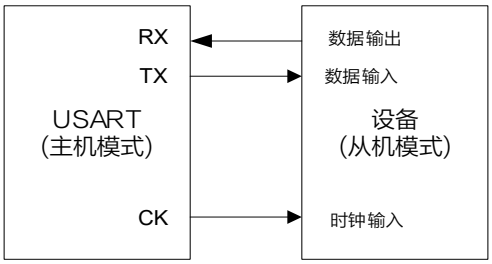
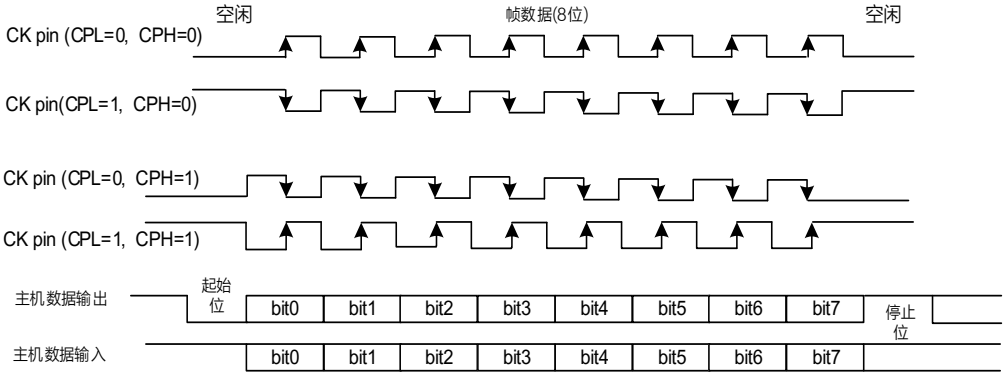



**图 19-12. 8-bit 格式的 USART 同步通信波形(CLEN=1)**

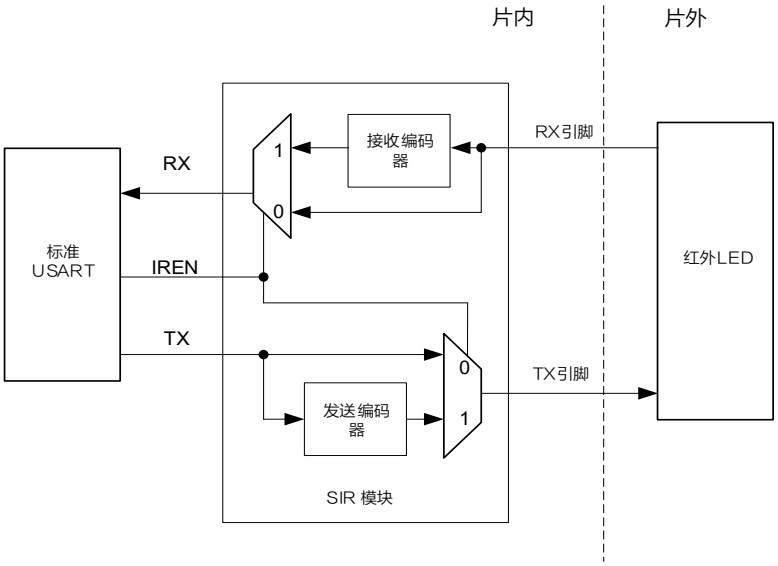



## 19.3.10. 串行红外(IrDA SIR)编解码功能模块

串行红外编解码功能通过置位USART_CTL2寄存器中IREN使能。在IrDA模式下，USART_CTL1寄存器的LMEN，STB[1:0]，CKEN位和USART_CTL2寄存器的HDEN，SCEN位将被清0。

在IrDA模式下，USART数据帧由SIR发送编码器进行调制，调制后的信号经由红外LED进行发送，经解调后将数据发送至USART接收器。对于编码器而言，波特率应小于115200。

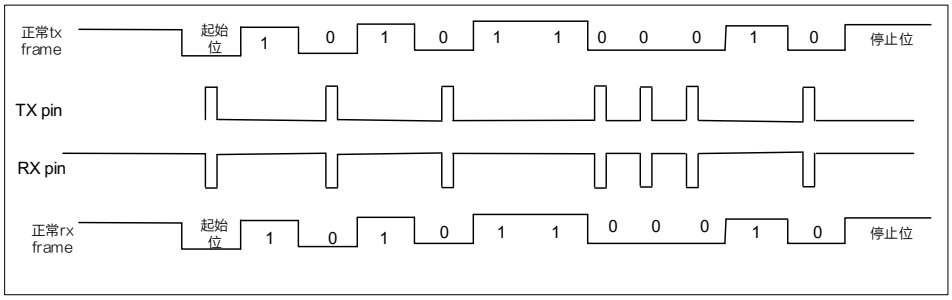
**图 19-13. IrDA SIR ENDEC 模块**




在IrDA模式下，TX引脚电平与RX引脚不同。TX引脚通常为低电平，RX引脚通常为高电平。IrDA引脚电平保持稳定代表逻辑'1'，红外光源脉冲(RTZ信号)代表逻辑'0'。其脉冲宽度通常占一个位时间的3/16。IrDA无法检测到宽度小于一个1个PSC时钟的脉冲。如果脉冲宽度大于1但是小于2倍PSC时钟，IrDA则无法可靠的检测到。

由于IrDA是一种半双工协议，因此在IrDA SIR ENDEC模块中，发送和接收不得同时进行。

**图 19-14. IrDA 数据调制**




将USART_CTL2寄存器中IRLP置位可以使SIR子模块工作在低功耗模式下。发送编码器由PCLK分频得到的低速时钟来驱动。分频系数在USART_GP寄存器中PSC[7:0]位配置，USART_BAUD寄存器需配置为16*PSC[7:0]。TX引脚脉冲宽度可以为低功耗波特率的3倍。接收器解码器工作模式与正常IrDA模式相同。

## 19.3.11. 半双工通信模式

通过设置USART_CTL2寄存器的HDEN位，可以使能半双工模式。

在半双工通信模式下，USART_CTL1寄存器的LMEN，CKEN位和USART_CTL2寄存器的SCEN，IREN位清零。

半双工模式下仅用单线通信，TX引脚和RX引脚将从内部连接到一起，RX引脚不再使用。TX引

脚应该被配置为开漏模式。通信冲突由软件处理。

### 19.3.12. 智能卡(ISO7816-3)模式

智能卡模式是一种异步通信模式，支持ISO7816-3协议。支持字节模式(T=0)和块模式(T=1)。将USART_CTL2寄存器的SCEN位置1，即可使能智能卡模式。在智能卡模式下，USART_CTL1寄存器的LMEN位和USART_CTL2的HDEN，IREN位应该清0。

如果CKEN位被置位，USART通过CK引脚向智能卡提供一个由PCLK分频得到的时钟。分频系数可在USART_GP寄存器中PSC[4:0]配置。CK引脚只为智能卡提供时钟源。

智能卡模式是一种半双工通信协议模式。当与智能卡连接时，TX引脚需要被设置成开漏模式，外接上拉电阻，这个引脚将会与智能卡驱动同一条双向连线。智能卡模式下的帧格式为： 1起始位+9数据位(包括1奇偶校验位)+1.5停止位。其中0.5个停止位被配置为接收器的停止位。

**图 19-15. ISO7816-3 数据帧格式**




#### 字节模式(T=0)

相较于正常操作模式下的时序，从发送移位寄存器到TX引脚的传递时间延迟了半个波特率时钟，并且TC标志的置位将根据USART_GP寄存器的GUAT[7:0]设置延迟某一特定时间。在智能卡模式下，在最后一帧数据的停止位之后，内部保护时间计数器将开始计数，GUAT[7:0]的值配置为ISO7816-3协议的CGT减12。在保护时间寄存器向上计数这段时间TC将被强制拉低，当计数达到设定值时，TC被置位。

在USART发送期间，如果检测到有奇偶校验错误，TX引脚在停止位最后一个位时间内被拉低，智能卡发送一个NACK信号。根据协议，USART会自动重发SCRTNUM次。在重发数据帧前面会插入2位的帧间隔。最后一次重发字节后，TC会立即被置位。如果在最大重发次数后仍然收到NACK信号，USART将会停止发送，帧错误标志被置位。USART不会将NACK信号作为起始位。

在USART接收期间，如果在当前数据帧检测到校验错误，TX引脚在停止位的最后一个位时间内会被拉低。智能卡会接收到NACK信号。然后在智能卡端会产生一个帧错误。如果接收到的字节是错误的，RBNE中断和接收DMA请求都不会被激活。根据协议，智能卡将要重新发送数据。如果在最大的重新发送次数后(这个次数的具体值在SCRTNUM位域)，接收到的字符仍然是错误的，USART停止发送NACK信号和标注这个错误为奇偶校验错误。将USART_CTL2寄存器中的NKEN置位可以使能NACK信号。

空闲帧和断开帧在智能卡模式下不适用。

#### 块模式(T=1)

在T=1(块模式)下，USART_CTL2寄存器的NKEN位应该清零来关闭校验错误发送。

当要从智能卡读取数据时，软件必须将USART_RT寄存器设置成BWT(块等待)-11的值并将RBNEIE置位。这个超时时间体现在波特时间单元。如果这个时间到了，还没有从智能卡收到应答，USART_STAT1寄存器中RTF位被置位。如果设置了USART_CTL3寄存器中RTIE位，将会产生中断。如果在超时之前收到了第一个字节，则会引起RBNE中断。如果用DMA从智能卡读取数据，也只能在第一个字节接收好后再去使能DMA。

第一个字节接收到后，RT[23:0]的值设置成CWT(字节等待时间)-11来使能两个连续字节间最大帧间隔自动校验。如果在RT[23:0]周期内智能卡停止发送字节，USART_STAT1寄存器中RTF将被置位。

USART用一个块长度计数器统计收到的字节数，这个计数器在USART开始发送的时候自动清0(TBE=0)。这个块长度信息位于智能卡发出数据的第三个字节(序言部分)，这个值必须写入USART_RT寄存器BL[7:0]。块长度计数器从0开始计数到最大值BL[7:0]+4。在块计数器计数到最大值时，USART_STAT1寄存器中块结束状态标志位EBF置位。如果设置了USART_CTL3寄存器中的EBIE位，将会产生中断。如果块长度发生错误，RTF置位。

当使用DMA模式接收时，在块开始之前，这个寄存器必须被设定为最小值(0x0)。为了得到这个值，在收到第四个字节后，会引起一个中断。软件可以从接收缓冲区读取第三个字节作为块长度。

如果接收时不使用DMA方式，为避免产生EBF状态标志，BL[7:0]需首先配置为最大值0xFF。在收到第三个字节后，真正的块长度值可以重新写入到BL[7:0]。

#### 直接和反向转换

智能卡协议定义了两种转换方式：直接转换和反向转换。

如果选择直接转换，从数据帧的最低位开始传输，TX引脚高电平代表逻辑'1'，偶校验。在这种情况下，USART_CTL3寄存器中MSBF位和DINV位都为0。

如果选择反向转换，从数据帧的最高位开始传输，TX引脚高电平代表逻辑'0'，偶校验。在这种情况下，USART_CTL3寄存器中MSBF位和DINV位都为1。

### 19.3.13. USART 中断

USART中断事件和标志如*表19-3. USART中断请求*所示：

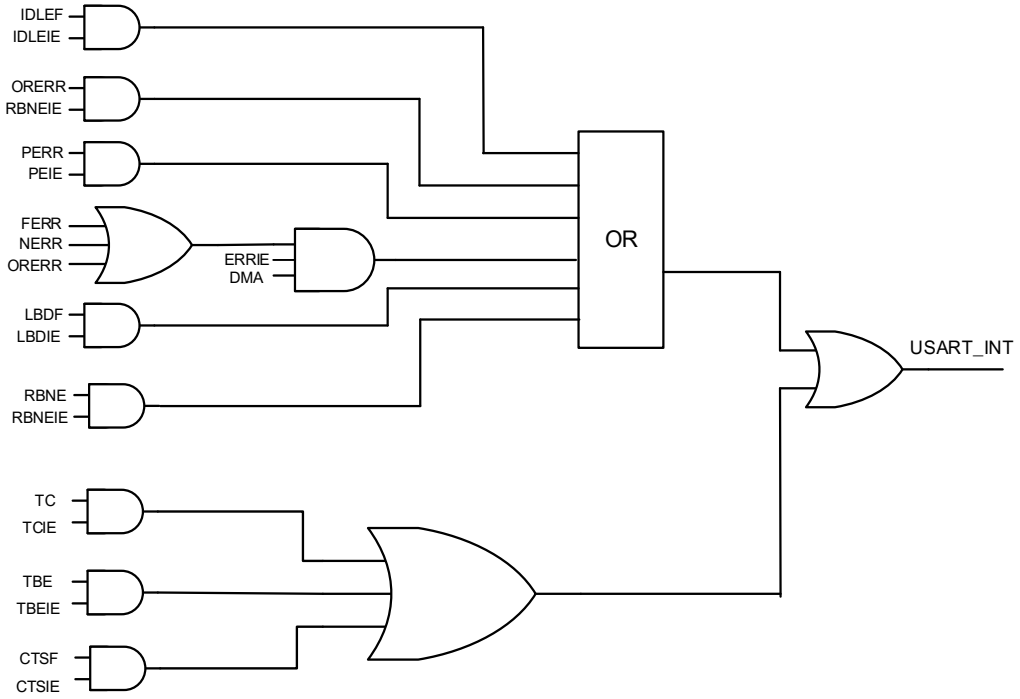
**表 19-3. USART 中断请求**

| 中断事件 | 事件标志 | 使能控制位 |
|---|---|---|
| 发送数据寄存器空 | TBE | TBEIE |
| CTS标志 | CTSF | CTSIE |
| 发送结束 | TC | TCIE |
| 接收到的数据可以读取 | RBNE | RBNEIE |

| 中断事件 | 事件标志 | 使能控制位 |
|---|---|---|
| 检测到过载错误 | ORERR | |
| 检测到线路空闲 | IDLEF | IDLEIE |
| 奇偶校验错误 | PERR | PERRIE |
| LIN模式下，检测到断开标志 | LBDF | LBDIE |
| 接收超时错误 | RTF | RTIE |
| 发现块尾 | EBF | EBIE |
| 接收错误(噪声错误、溢出错误、帧错误)当DMA接收使能时 | NERR或ORERR或FERR | ERRIE |

在发送给中断控制器之前，所有的中断事件是逻辑或的关系。因此在任何时候 USART 只能向控制器产生一个中断请求。不过软件可以在一个中断服务程序里处理多个中断事件。

**图 19-16. USART 中断映射框图**



## 19.4. USART 寄存器

USART0基地址：0x4001 1000
USART1基地址：0x4000 4400
USART2基地址：0x4000 4800
UART3基地址：0x4000 4C00
UART4基地址：0x4000 5000
USART5基地址：0x4001 1400
UART6基地址：0x4000 7800
UART7基地址：0x4000 7C00

### 19.4.1. 状态寄存器 0(USART_STAT0)

地址偏移：0x00
复位值：0x0000 00C0

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | CTSF | LBDF | TBE | TC | RBNE | IDLEF | ORERR | NERR | FERR | PERR |
| | | | | | | rc_w0 | rc_w0 | r | rc_w0 | rc_w0 | r | r | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:10 | 保留 | 必须保持复位值。 |
| 9 | CTSF | CTS变化标志<br>如果设置了USART_CTL2寄存器中CTSEN位，当nCTS输入变化时，该位由硬件置位。如果设置了USART_CTL2寄存器中CTSIE位，将产生中断。<br>该位由软件清0。<br>0：nCTS状态线没有变化。<br>1：nCTS状态线发生变化。<br>该位对UART3/4/6/7无效。 |
| 8 | LBDF | LIN断开检测标志<br>寄存器USART_CTL1寄存器中LMEN置位，说明检测到LIN断开。如果USART_CTL1寄存器中LBDIE被置位时，将产生中断。<br>该位由软件清0。<br>0：没有检测到LIN断开字符。<br>1：检测到LIN断开字符。 |
| 7 | TBE | 发送数据缓冲区空<br>上电复位或待发送数据已发送至移位寄存器后，该位置1。USART_CTL0寄存器中TBEIE被置位将产生中断。 |

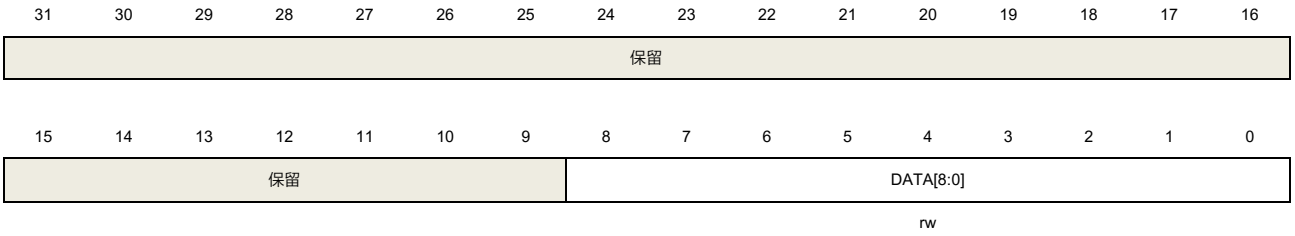
| | | |
|---|---|---|
| | | 该位在软件将待发送数据写入USART_DATA时被清0。<br>0：发送数据缓冲区不为空。<br>1：发送数据缓冲区空。 |
| 6 | TC | 发送完成<br>上电复位后，该位被置1。如果TBE置位，在当前数据发送完成时该位置1。<br>USART_CTL0寄存器中TCIE被置位将产生中断。<br>该位由软件清0。<br>0：发送没有完成<br>1：发送完成 |
| 5 | RBNE | 读数据缓冲区非空。<br>当读数据缓冲区接收到来自移位寄存器的数据时，该位置1。当寄存器USART_CTL0的RBNEIE位被置位，将会有中断产生。<br>软件可以通过对该位写0或读USART_DATA寄存器来将该位清0。<br>0：读数据缓冲区为空。<br>1：读数据缓冲区不为空。 |
| 4 | IDLEF | 空闲线检测标志。<br>在一个帧时间内，在RX引脚检测到空闲状态，该位置1。当寄存器USART_CTL0的IDLEIE位被置位，将会有中断产生。<br>软件先读USART_STAT0，再读USART_DATA可清除该位。<br>0：未检测到空闲帧。<br>1：检测到空闲帧。 |
| 3 | ORERR | 溢出错误<br>在RBNE置位的情况下，如果USART_DATA寄存器接收到来自移位寄存器的数据，该位置1。当寄存器USART_CTL2的ERRIE位被置位，将会有中断产生。<br>软件先读USART_STAT0，再读USART_DATA可清除该位。<br>0：没有检测到溢出错误。<br>1：检测到溢出错误。 |
| 2 | NERR | 噪声错误标志<br>将USART_CTL2寄存器中OSB清0，在接收数据时，如果在RX引脚检测到噪声，该位被置位。当寄存器USART_CTL2的ERRIE位被置位，将会有中断产生。<br>软件先读USART_STAT0，再读USART_DATA可清除该位。<br>0：没检测到噪声错误。<br>1：检测到噪声错误。 |
| 1 | FERR | 帧错误<br>接收数据期间，在停止位传输过程中，RX引脚检测到低电平，该位被置位。当寄存器USART_CTL2的ERRIE位被置位，将会有中断产生。<br>软件先读USART_STAT0，再读USART_DATA可清除该位。<br>0：未检测到帧错误<br>1：检测到帧错误 |
| 0 | PERR | 校验错误<br>当接收到的数据帧校验位与预期校验值不同时，该位置位。 |

软件先读USART_STAT0，再读USART_DATA可清除该位。
0：没检测到校验错误。
1：检测到校验错误。

### 19.4.2. 数据寄存器(USART_DATA)

地址偏移：0x04
复位值：未定义

该寄存器只能按字(32位)访问。

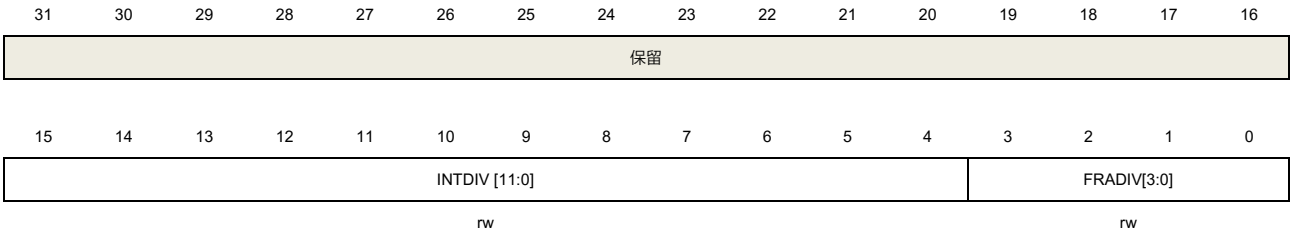



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:9 | 保留 | 必须保持复位值。 |
| 8:0 | DATA[8:0] | 发送或接收的数据值<br>软件可以通过写这些位来改变要发送的数据，或读这些位的值来获取接收到的数据。<br>如果使能了奇偶校验，当发送数据被写入寄存器，数据的最高位(第7位或第8位取决于USART_CTL0寄存器的WL位)将被校验位取代。 |

### 19.4.3. 波特率寄存器(USART_BAUD)

地址偏移：0x08
复位值：0x0000 0000

使能USART(UEN=1)时，不能写该寄存器。
该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:4 | INTDIV[11:0] | 波特率分频器的整数部分。 |
| 3:0 | FRADIV [3:0] | 波特率分频器的小数部分。 |

如果使能了8倍过采样，软件必须保证FRADIV[3]为0。

### 19.4.4. 控制寄存器 0(USART_CTL0)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| OVSMOD | 保留 | UEN | WL | WM | PCEN | PM | PERRIE | TBEIE | TCIE | RBNEIE | IDLEIE | TEN | REN | RWU | SBKCMD |
| rw | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15 | OVSMOD | 采样模式<br>0：16倍采样<br>1：8倍采样<br>如果SCEN=1，IREN=1或者LMEN=1，OVSMOD由硬件强制为0。 |
| 14 | 保留 | 必须保持复位值。 |
| 13 | UEN | USART使能<br>0：USART禁用<br>1：USART使能 |
| 12 | WL | 字长<br>0：8数据位<br>1：9数据位 |
| 11 | WM | 从静默模式唤醒方法<br>0：空闲线<br>1：地址匹配 |
| 10 | PCEN | 校验控制使能<br>0：校验控制禁用<br>1：校验控制被使能 |
| 9 | PM | 校验模式<br>0：偶校验<br>1：奇校验 |
| 8 | PERRIE | 校验错误中断使能。<br>如果该位置1，USART_STAT0寄存器中PERR被置位时产生中断。<br>0：校验错误中断禁用。 |

| | | |
|---|---|---|
| | | 1：校验错误中断使能。 |
| 7 | TBEIE | 发送缓冲区空中断使能。<br>如果该位置1，USART_STAT0寄存器中TBE被置位时产生中断。<br>0：发送缓冲区空中断禁止。<br>1：发送缓冲区空中断使能。 |
| 6 | TCIE | 发送完成中断使能。<br>如果该位置1，USART_STAT0寄存器中TC被置位时产生中断。<br>0：发送完成中断禁用。<br>1：发送完成中断使能。 |
| 5 | RBNEIE | 读数据缓冲区非空中断和过载错误中断使能。<br>如果该位置1，USART_STAT0寄存器中RBNE或ORERR被置位时产生中断。<br>0：读数据缓冲区非空中断和过载错误中断禁用。<br>1：读数据缓冲区非空中断和过载错误中断使能。 |
| 4 | IDLEIE | IDLE线检测中断使能。<br>如果该位置1，USART_STAT0寄存器中IDLEF被置位时产生中断。<br>0：IDLE线检测中断禁用。<br>1：IDLE线检测中断禁用使能。 |
| 3 | TEN | 发送器使能<br>0：发送器禁用<br>1：发送器使能 |
| 2 | REN | 接收器使能<br>0：接收器禁用<br>1：接收器使能 |
| 1 | RWU | 接收器从静默模式中唤醒。<br>软件可以通过将该位置1使得USART进入静默模式，将该位清0唤醒USART。<br>空闲帧唤醒模式下(WM=0)，当检测到空闲帧时，该位由硬件清0。地址匹配模式下(WM=1)，当接收到一个地址匹配帧时，该位由硬件清0；或接收到一个地址非匹配帧时，由硬件置1。<br>0：接收器处于正常工作模式。<br>1：接收器处于静默模式。 |
| 0 | SBKCMD | 发送断开帧<br>软件通过发送断开帧将该位置1。<br>断开帧传输结束由硬件清0。<br>0：没有发送断开帧<br>1：发送断开帧 |

### 19.4.5. 控制寄存器 1(USART_CTL1)

地址偏移：0x10

复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | LMEN | STB[1:0] | | CKEN | CPL | CPH | CLEN | 保留 | LBDIE | LBLEN | 保留 | ADDR[3:0] | | | |
| | rw | rw | | rw | rw | rw | rw | | rw | rw | | rw | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:15 | 保留 | 必须保持复位值。 |
| 14 | LMEN | LIN模式使能<br>0：LIN模式禁用<br>1：LIN模式使能 |
| 13:12 | STB[1:0] | STOP位长<br>00：1停止位<br>01：0.5停止位<br>10：2停止位<br>11：1.5停止位<br>对于UART3/4/6/7，只有1位停止位和两位停止位是有效的。 |
| 11 | CKEN | CK引脚使能<br>0：CK引脚禁用<br>1：CK引脚使能<br>该位对于UART3/4/6/7无效。 |
| 10 | CPL | 时钟极性<br>该位用来设定在同步模式下CK引脚的极性。<br>0：CK引脚不对外发送时保持为低电平。<br>1：CK引脚不对外发送时保持为高电。<br>该位对于UART3/4/6/7无效。 |
| 9 | CPH | 时钟相位<br>该位用来设定在同步模式下CK引脚的相位。<br>0：在首个时钟边沿采样第一个数据。<br>1：在第二个时钟边沿采样第一个数据。<br>该位对于UART3/4/6/7无效。 |
| 8 | CLEN | CK信号长度<br>该位用来设定在同步模式下CK信号的长度。<br>0：8位数据帧中有7个CK脉冲，9位数据帧中有8个CK脉冲<br>1：8位数据帧中有8个CK脉冲，9位数据帧中有9个CK脉冲<br>该位对于UART3/4/6/7无效。 |
| 7 | 保留 | 必须保持复位值。 |
| 6 | LBDIE | LIN断开信号检测中断使能。 |

| | | |
|---|---|---|
| | | 如果该位置1，当USART_STAT0寄存器中LBDF被置位时将产生中断。<br>0：断开信号检测中断禁用。<br>1：断开信号检测中断使能。 |
| 5 | LBLEN | LIN断开帧长度<br>该位用来设定在断开帧长度。<br>0：10位<br>1：11位 |
| 4 | 保留 | 必须保持复位值。 |
| 3:0 | ADDR[3:0] | USART地址<br>地址匹配唤醒模式下(WM=1)，如果接收到的数据帧低四位与ADDR[3:0]值不相等，USART就会进入静默模式；如果接收到的数据帧低四位与ADDR[3:0]值相等，USART就会被唤醒。 |

### 19.4.6. 控制寄存器 2(USART_CTL2)

地址偏移：0x14
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | OSB | CTSIE | CTSEN | RTSEN | DENT | DENR | SCEN | NKEN | HDEN | IRLP | IREN | ERRIE |
| | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11 | OSB | 一个数据位采样一次的方法。<br>该位用于选择采样方法。当该位被置位时，USART对一个数据位取一个采样点，而不是一个数据位取三个采样点。当使用该方法时，噪声错误标志(NERR)需禁用。 |
| 10 | CTSIE | CTS中断使能<br>如果该位置1，当USART_STAT0寄存器中CTSF被置位时将产生中断。<br>0：CTS中断禁用<br>1：CTS中断使能<br>该位对于UART3/4/6/7无效。 |
| 9 | CTSEN | CTS使能<br>该位用于使能CTS硬件流控制功能。<br>0：CTS硬件流控制禁用。<br>1：CTS硬件流控制使能。<br>该位对于UART3/4/6/7无效。 |

| | | |
|---|---|---|
| 8 | RTSEN | RTS使能<br>该位用于使能RTS硬件流控制功能。<br>0：RTS硬件流控制禁用。<br>1：RTS硬件流控制使能。<br>该位对于UART3/4/6/7无效。 |
| 7 | DENT | DMA发送使能<br>0：DMA发送模式禁用。<br>1：DMA发送模式使能。 |
| 6 | DENR | DMA接收使能<br>0：DMA接收模式禁用。<br>1：DMA接收模式使能。 |
| 5 | SCEN | 智能卡模式使能<br>该位用于使能智能卡模式。<br>0：智能卡模式禁用。<br>1：智能卡模式使能。<br>该位对于UART3/4/6/7无效。 |
| 4 | NKEN | 在智能卡模式NACK使能。<br>该位用于智能卡模式在奇偶校验错误发生时使能NACK发送。<br>0：当出现校验错误时不发送NACK。<br>1：当出现校验错误时发送NACK。<br>该位对于UART3/4/6/7无效。 |
| 3 | HDEN | 半双工使能<br>该位用于使能半双工模式。<br>0：半双工模式禁用。<br>1：半双工模式使能。 |
| 2 | IRLP | IrDA低功耗模式<br>该位用于为IrDA模式选择低功耗模式。<br>0：正常模式<br>1：低功耗模式 |
| 1 | IREN | IrDA模式使能<br>0：IrDA禁用<br>1：IrDA使能 |
| 0 | ERRIE | 错误中断使能<br>当DMA接收模式(DENR=1)使能时，如果该位被置1，USART_STAT0寄存器中FERR，ORERR，NERR被置位将产生中断。<br>0：错误中断禁用<br>1：错误中断使能。 |

### 19.4.7. 保护时间和预分频器寄存器(USART_GP)

地址偏移：0x18
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:8 | GUAT[7:0] | 智能卡模式下的保护时间值。<br>TC标志置位时间延时GUAT[7:0]个波特时钟周期。<br>该位对于UART3/4/6/7无效。 |
| 7:0 | PSC[7:0] | 使能USART IrDA低功耗模式，这些位用来设定将外设时钟(PCLK1/PCLK2)分频产生低功耗频率的分频系数。<br>00000000：保留 - 不要写入该值<br>00000001：对源时钟1分频。<br>…<br>11111111：对源时钟255分频。<br>在IrDA正常模式下，PSC只能设置成00000001。<br>在智能卡模式下，PSC[4:0]用于设定外设时钟(APB1/APB2)生成智能卡时钟的分频系数。实际的分频系数为PSC[4:0]设定值的两倍。<br>00000：保留 - 不要写入该值。<br>00001：对源时钟2分频。<br>00010：对源时钟4分频。<br>…<br>11111：对源时钟62分频。<br>在智能卡模式下，PSC[7:5]保留。 |

### 19.4.8. 控制寄存器 3(USART_CTL3)

偏移地址：0x80
复位值：0x0000 0000

UART3/4/6/7未使用该寄存器。
该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | MSBF | DINV | TINV | RINV | 保留 | | EBIE | RTIE | SCRTNUM[2:0] | | | RTEN |
| | | | | rw | rw | rw | rw | | | rw | rw | rw | | | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11 | MSBF | 高位在前<br>该位用于设定数据在发送或接收时的顺序。<br>0：数据发送/接收，采用低位在前。<br>1：数据发送/接收，采用高位在前。<br>USART被使能(UEN=1)时，这一位不能被改写。 |
| 10 | DINV | 数据位反转<br>该位用于设定在发送或接收时数据位的极性。<br>0：数据位信号值没有反转。<br>1：数据位信号值被反转。<br>USART被使能(UEN=1)时，这一位不能被改写。 |
| 9 | TINV | TX 引脚电平反转<br>该位用于设定TX引脚极性。<br>0：TX引脚信号值没有反转。<br>1：TX引脚信号值被反转。<br>USART被使能(UEN=1)时，这一位不能被改写。 |
| 8 | RINV | RX引脚电平反转<br>该位用于设定RX引脚极性。<br>0：RX引脚信号值没有反转。<br>1：RX引脚信号值被反转。<br>USART被使能(UEN=1)时，这一位不能被改写。 |
| 7:6 | 保留 | 必须保持复位值。 |
| 5 | EBIE | 块结束标志中断使能位<br>如果该位置1，USART_STAT1寄存器中EBF被置位时产生中断。<br>0：块中断禁用<br>1：块中断使能 |
| 4 | RTIE | 接收超时标志中断使能位。<br>如果该位置1，USART_STAT1寄存器中RTF被置位时产生中断。<br>0：接收超时中断禁用。<br>1：接收超时中断使能。 |
| 3:1 | SCRTNUM[2:0] | 智能卡自动重试次数寄存器。<br>在智能卡模式下，这些位用来设定在发送和接收时重试的次数。<br>在发送模式下，一帧数据可以重发SCRTNUM次。如果一帧数据发送失败SCRTNUM+1次，FERR被置位。<br>在接收模式下，USART接收一个数据帧可以执行SCRTNUM+1次。如果一个数据帧 |

| | | |
|---|---|---|
| | | 校验位不匹配事件产生SCRTNUM+1次，RBNE位和PERR位被置位。<br>当这些位被设置为0x0时，在发送模式下这些位将不会自动发送。 |
| 0 | RTEN | 接收器超时使能<br>该位用于使能USART接收超时。<br>0：接收器超时检测功能禁用。<br>1：接收器超时检测功能被使能。 |

## 19.4.9. 接收超时寄存器(USART_RT)

偏移地址：0x84
复位值：0x0000 0000
UART3/4/6/7未使用该寄存器。

该寄存器只能按字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | BL[7:0] | 块长度<br>这些位用于设定智能卡T=1的接收时，块的长度。它的值等于信息字节的长度+结束部分的长度(1-LEC/2-CRC)-1。<br>这个值可以在块接收开始去设置(用于需要从块的序言提取块的长度的情形)，这个值在每一个接收时钟周期只能设置一次。在智能卡模式下，当TBE=0时，块的长度计数器被清0。<br>在其他模式下，当REN=0(禁用接收器)或者当USART_STAT1寄存器的EBF位被写0时，块的长度计数器被清0。 |
| 23:0 | RT[23:0] | 接收器超时阈值<br>该位域用于指定接收超时值，单位是波特时钟的时长。<br>标准模式下，如果在最后一个字节接收后，在RT规定的时长内，没有检测到新的起始位，USART_STAT1寄存器中RTF标志被置位。<br>在智能卡模式，这个值被用来实现CWT和BWT。在这种情况下，超时检测是从最后一个接收字节的起始位开始算的。<br>这些位可以在工作时改写。假如一个新数据到来的时间比RT规定的晚，RTF标志会被置位。对于每个接收字符，这个值只能改写一次。<br>**注意：**当16倍过采样时，RT[23:0]不能被配置为0xFFFFFFFF。 |

## 19.4.10. 状态寄存器 1(USART_STAT1)

偏移地址：0x88

复位值：0x0000 0000
UART3/4/6/7未使用该寄存器。

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | EBF | RTF | 保留 | | | | | | | | | | |
| | | | w0 | w0 | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:17 | 保留 | 必须保持复位值。 |
| 16 | BSY | 忙标志<br>USART接收一帧数据时被置位。<br>0：USART接收通道空闲。<br>1：USART接收通道忙。 |
| 15:13 | 保留 | 必须保持复位值。 |
| 12 | EBF | 块结束标志<br>该位在接收字节数(从块开始开始计数，包含序言)等于或者大于BLEN+4时被置位。USART_CTL3寄存器中EBIE被置位将产生中断。<br>软件可以通过写0清除该位。<br>0：块结束事件没有发生<br>1：块结束事件发生 |
| 11 | RTF | 接收超时标志<br>该位在RX引脚空闲时间已经超过RT值时被置位。USART_CTL3寄存器中RTIE被置位将产生中断。<br>软件可以通过写0清除该位。<br>0：接收器超时事件没有发生<br>1：接收器超时事件发生 |
| 10:0 | 保留 | 必须保持复位值。 |

## 19.4.11. 兼容性控制寄存器(USART_CHC)

偏移地址：0xC0
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 保留 | EPERR | 保留 | BCM | PCM | HCM |
|---|---|---|---|---|---|
| | w0c | | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:9 | 保留 | 必须保持复位值。 |
| 8 | EPERR | 校验错误超前检测标志。<br>该标志将在RBNE置位前，校验位被检测到时置位。<br>软件写0可以清除该位。<br>0：没有检测到校验错误<br>1：检测到校验错误 |
| 7:3 | 保留 | 必须保持复位值。 |
| 2 | BCM | 断开帧兼容模式<br>0：当CTS硬件流控制使能时，发送器在发送断开帧之前不会检测nCTS的输入状态<br>1：当CTS硬件流控制使能时，发送器在发送断开帧之前检测nCTS的输入状态<br>该位对于UART3/4/6/7无效。 |
| 1 | PCM | 校验兼容模式<br>0：在地址匹配模式唤醒情况下，数据的最高位被用来识别地址，而不是用作校验位。接收器不会对地址数据进行校验(PERR将不会被置位除非发生校验错误)<br>1：在地址匹配模式唤醒情况下，接收器对地址进行校验，当检测到校验错误时PERR被置位 |
| 0 | HCM | 硬件流控制兼容模式<br>0：nRTS信号与USART_STAT0寄存器中RBNE位相同<br>1：nRTS信号在最后一个数据位被采样后被置位<br>该位对于UART3/4/6/7无效。 |

# 20. 内部集成电路总线接口(I2C)

## 20.1. 简介

I2C(内部集成电路总线)模块提供了符合工业标准的两线串行制接口，可用于 MCU 和外部 I2C 设备的通讯。I2C 总线使用两条串行线：串行数据线 SDA 和串行时钟线 SCL。

I2C 接口模块实现了 I2C 协议的标速模式和快速模式，具备 CRC 计算和校验功能、支持 SMBus(系统管理总线)、PMBus(电源管理总线)和 SAM_V(验证安全控制模块)模式，此外还支持多主机 I2C 总线架构。I2C 接口模块也支持 DMA 模式，可有效减轻 CPU 的负担。

## 20.2. 主要特征

- 并行总线至 I2C 总线协议的转换及接口。
- 同一接口既可实现主机功能又可实现从机功能。
- 主从机之间的双向数据传输。
- 支持 7 位和 10 位的地址模式和广播寻址。
- 支持 I2C 多主机模式。
- 支持标速模式(最高 100 kHz)和快速模式(最高 400 kHz)。
- 从机模式下可配置的 SCL 主动拉低。
- 支持 DMA 模式。
- 兼容 SMBus 2.0 和 PMBus。
- 两个中断：字节成功传输中断和错误事件中断。
- 可选择的 PEC(报文错误校验)生成和校验。
- 支持 SAM_V 模式。

## 20.3. 功能说明

I2C 接口的内部结构如*图 20-1. I2C 模块框图*所示。

**图 20-1. I2C 模块框图**

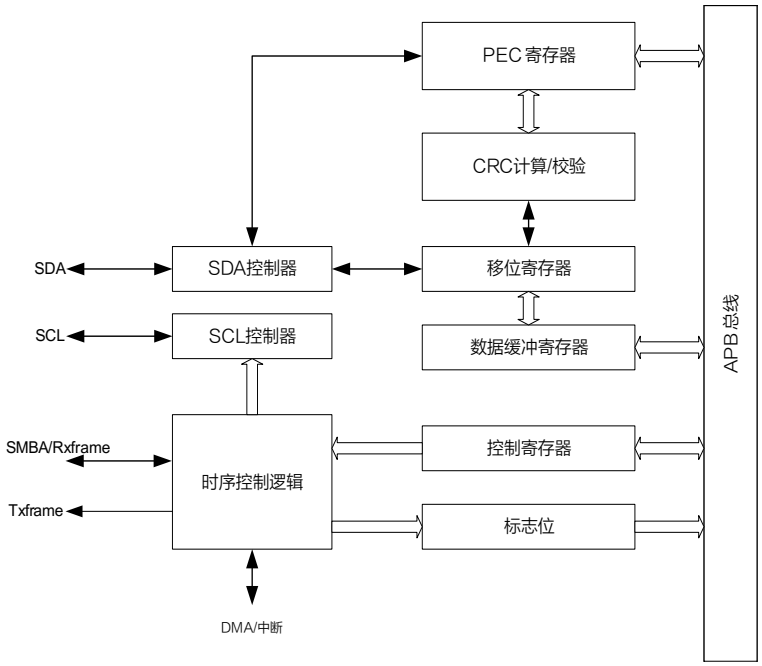



**表 20-1. I2C 总线术语说明(参考飞利浦 I2C 规范)**

| 术语 | 说明 |
|---|---|
| 发送器 | 发送数据到总线的设备 |
| 接收器 | 从总线接收数据的设备 |
| 主机 | 初始化数据传输，产生时钟信号和结束数据传输的设备 |
| 从机 | 由主机寻址的设备 |
| 多主 | 多个主机可以尝试在不破坏信息的前提下同时控制总线 |
| 同步 | 同步两个或更多设备之间的时钟信号的过程 |
| 仲裁 | 如果超过一个主机同时试图控制总线，只有一个主机被允许，且获胜主机的信息不被破坏 |

### 20.3.1. SDA 线和 SCL 线

I2C 模块有两条接口线：串行数据 SDA 线和串行时钟 SCL 线。连接到总线上的设备通过这两根线互相传递信息。

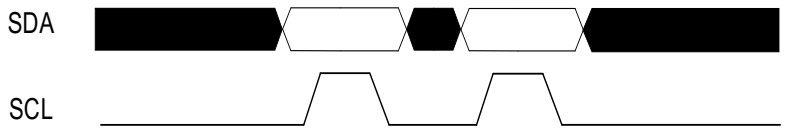
SDA 和 SCL 都是双向线，通过一个电流源或者上拉电阻接到电源正极。当总线空闲时，两条线都是高电平。连接到总线的设备输出极必须是开漏或者开集，以提供线与功能。I2C 总线上的数据在标准模式下可以达到 100 Kbit/s，在快速模式下可以达到 400 Kbit/s。由于 I2C 总线上可能会连接不同工艺的设备(CMOS，NMOS，双极性器件)，逻辑'0'和逻辑'1'的电平并不是

固定的，取决于 $V_{DD}$ 的实际电平。

### 20.3.2. 数据有效性

时钟信号的高电平期间 SDA 线上的数据必须稳定。只有在时钟信号 SCL 变低的时候数据线 SDA 的电平状态才能跳变(如*图 20-2. 数据有效性*)。每个数据比特传输需要一个时钟脉冲。

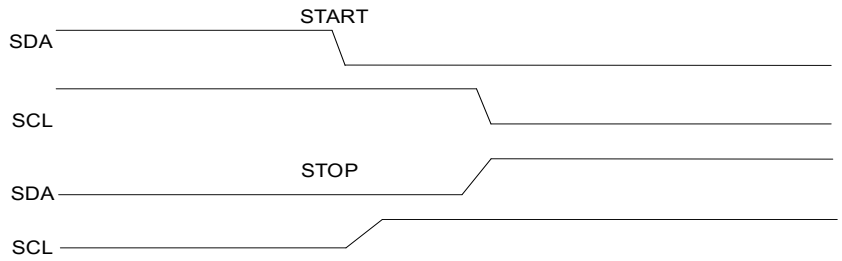
**图 20-2. 数据有效性**




### 20.3.3. 开始和停止信号

所有的数据传输起始于一个 START 结束于一个 STOP(参见*图 20-3. 起始和停止信号*)。START 信号定义为，在 SCL 为高时，SDA 线上出现一个从高到低的电平转换。STOP 信号定义为，在 SCL 为高时，SDA 线上出现一个从低到高的电平转换。

**图 20-3. 起始和停止信号**

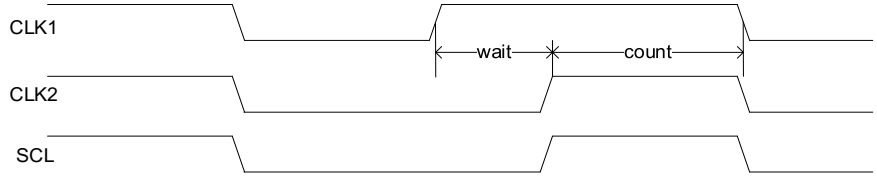



### 20.3.4. 时钟同步

两个主机可以同时在空闲总线上开始传送数据，因此必须通过一些机制来决定哪个主机获取总线的控制权并完成传输，这一般是通过时钟同步和仲裁来完成的。单主机系统下不需要时钟同步和仲裁机制。

时钟同步通过 SCL 线的线与来实现。这就是说 SCL 线的高到低切换会使器件开始计数它们的低电平周期，而且当主机的时钟变低电平时，它会使 SCL 线保持这种状态直到到达时钟的高电平(参见*图 20-4. 时钟同步*)。但是如果另一个时钟仍处于低电平周期，这个时钟的低到高切换不会改变 SCL 线的状态。因此 SCL 线被有最长低电平周期的器件保持低电平。此时低电平周期短的器件会进入高电平的等待状态。

**图 20-4. 时钟同步**



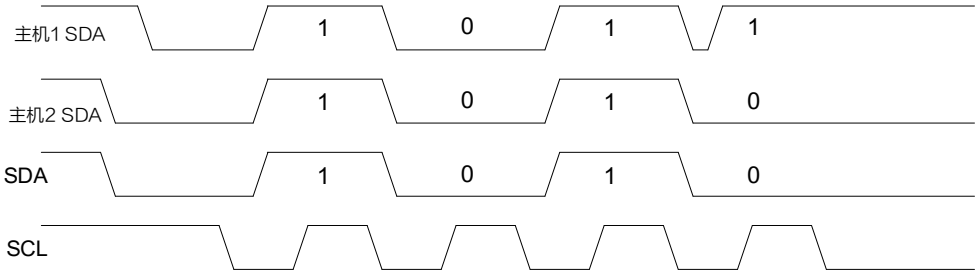
### 20.3.5. 仲裁

仲裁和同步一样，都是为了解决多主机情况下的总线控制冲突。仲裁的过程与从机无关。

只有在总线空闲的时候主机才可以启动传输。两个主机可能在START信号的最短保持时间内在总线上产生一个有效的START信号，这种情况下需要仲裁来决定由哪个主机来完成传输。

仲裁逐位进行，在每一位的仲裁期间，当SCL为高时，每个主机都检查SDA电平是否和自己发送的相同。仲裁的过程需要持续很多位。理论上讲，如果两个主机所传输的内容完全相同，那么它们能够成功传输而不出现错误。如果一个主机发送高电平但检测到SDA电平为低，则认为自己仲裁失败并关闭自己的SDA输出驱动，而另一个主机则继续完成自己的传输。

**图 20-5. SDA 线仲裁**




### 20.3.6. I2C 通讯流程

每个I2C设备(不管是微控制器，LCD驱动，存储器或者键盘接口)都通过唯一的地址进行识别，根据设备功能，他们既可以是发送器也可作为接收器。

I2C从机检测到I2C总线上的START信号之后，就开始从总线上接收地址，之后会把从总线接收到的地址和自身的地址(通过软件编程)进行比较，当两个地址相同时，I2C从机将发送一个确认应答(ACK)，并响应总线的后续命令：发送或接收所需数据。此外，如果软件开启了广播呼叫，则I2C从机始终对一个广播地址(0x00)发送确认应答。I2C模块始终支持7位和10位的地址。

I2C主机负责产生START信号和STOP信号来开始和结束一次传输，并且负责产生SCL时钟。

**图 20-6. 7 位地址的 I2C 通讯流程**

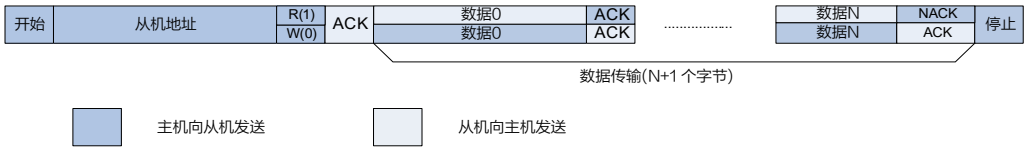



**图 20-7. 10 位地址的 I2C 通讯流程(主机发送)**

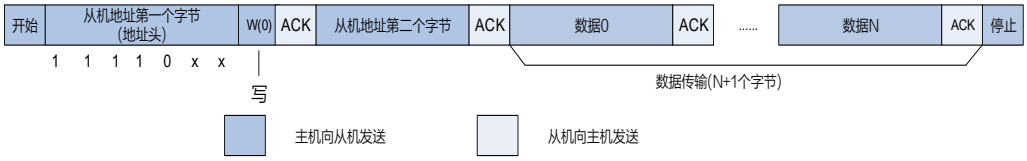
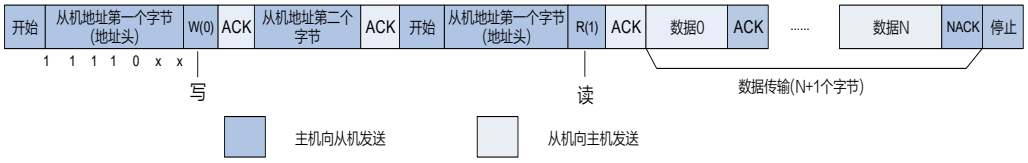


图 20-8. 10 位地址的 I2C 通讯流程(主机接收)




## 20.3.7. 软件编程模型

一个I2C设备例如LCD驱动器可能只是作为一个接收器，但是一个存储器既可以接收数据，也能发送数据。除了按照发送/接收方来区分，I2C设备也分为数据传输的主机和从机。主机是指负责初始化总线上数据的传输并产生时钟信号的设备，此时任何被寻址的设备都是从机。

不管I2C设备是主机还是从机，都可以发送或接收数据，因此，I2C设备有以下4种运行模式：

- 主机发送方
- 主机接收方
- 从机发送方
- 从机接收方

I2C模块支持以上四种模式。系统复位以后，I2C默认工作在从机模式下。通过软件配置使I2C在总线上发送一个START信号之后，I2C变为主机模式，软件配置在I2C总线上发送STOP信号后，I2C又变回从机模式。

### 从机发送模式下的软件流程

如*图20-9. 从机发送模式(10位地址模式)*所示，在从机模式下要发送数据，软件应该按照以下步骤来运行操作：

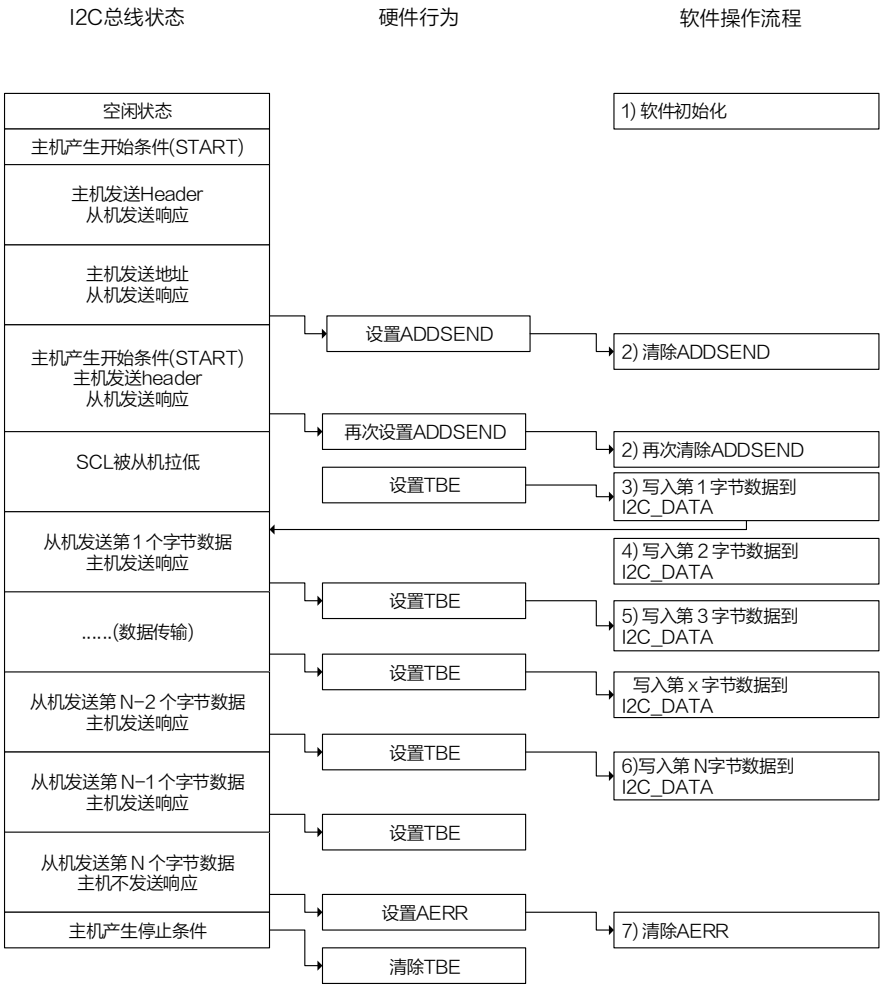
1. 首先，软件应该使能I2C外设时钟，以及配置I2C_CTL1中时钟相关寄存器来确保正确的I2C时序。使能和配置以后，I2C运行在默认的从机模式状态，等待I2C总线上的START信号和地址。
2. 当接收到一个START信号及随后的地址后，地址可以是7位格式也可以是10位格式，I2C硬件将I2C_STAT0寄存器的ADDSEND位置1，此位应该被软件查询或者中断监视，发现置位后，软件应该读I2C_STAT0寄存器然后读I2C_STAT1寄存器来清除ADDSEND位。如果地址是10位格式，I2C主机应该接着再产生一个START并发送一个地址头到I2C总线。从机在检测到START和紧接着的地址头之后会继续将ADDSEND位置1。软件可以通过读I2C_STAT0寄存器和接着读I2C_STAT1寄存器来第二次清除ADDSEND位。
3. 现在I2C进入数据发送状态，由于移位寄存器和数据寄存器I2C_DATA都是空的，硬件将TBE位置1。软件此时可以写入第一个字节数据到I2C_DATA寄存器，但是TBE位并没有被清0，因为写入I2C_DATA寄存器的字节被立即移入内部移位寄存器。当移位寄存器非空的时候，I2C开始发送数据到I2C总线。
4. 第一个字节的发送期间，软件可以写第二个字节到I2C_DATA，此时TBE位被清0，因为I2C_DATA寄存器和移位寄存器都不是空。
5. 第一个字节的发送完成之后，TBE被再次置起，软件可以写第三个字节到I2C_DATA，同

时TBE位被清0。在此之后，任何时候TBE被置1，只要依然有数据待被发送，软件都可以写入一个字节到I2C_DATA寄存器。

6. 倒数第二个字节发送期间，软件写最后一个数据到I2C_DATA寄存器来清除TBE标志位，之后就再不用关心TBE的状态。TBE位会在倒数第二个字节发送完成后置起，直到检测到STOP信号时被清0。

7. 根据I2C协议，I2C主机将不会对接收到的最后一个字节发送应答，所以在最后一个字节发送结束后，I2C从机的AERR(应答错误)会置起以通知软件发送结束。软件写0到AERR位可以清除此位。

**图 20-9. 从机发送模式(10 位地址模式)**



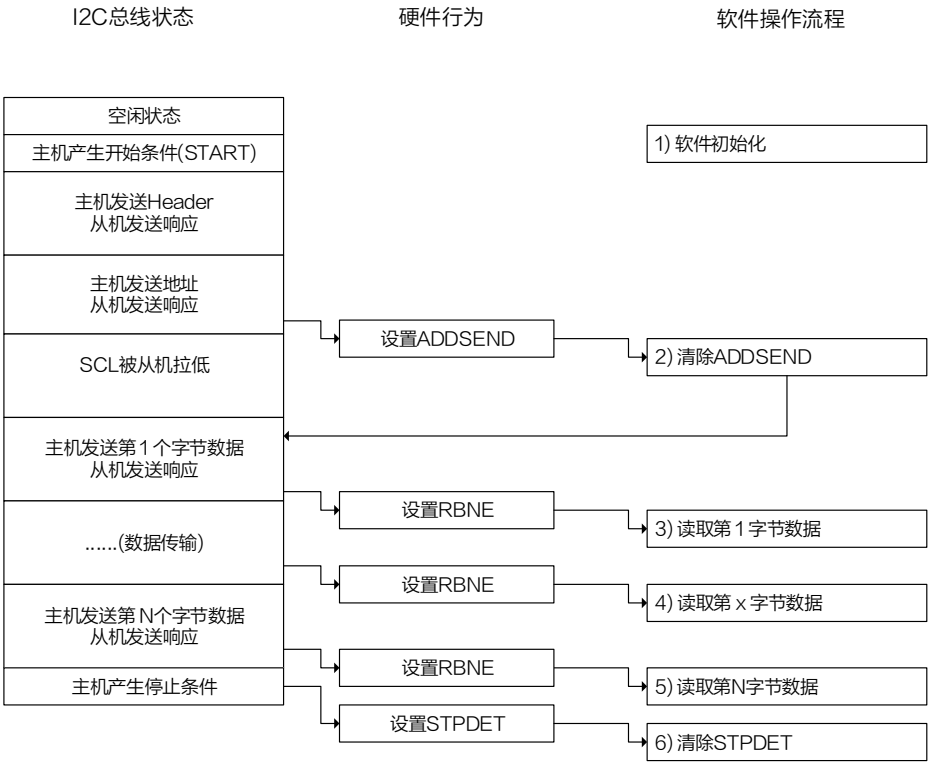

## 从机接收模式下的软件流程

如*图 20-10. 从机接收模式(10 位地址模式)*所示，在从机模式下接收数据时，软件应该遵循这些步骤来操作：

1. 首先，软件应该使能I2C外设时钟，以及配置I2C_CTL1中时钟相关寄存器来确保正确的I2C时序。使能和配置以后，I2C运行在默认的从机模式状态，等待START信号以及地址。

2. 在接收到START起始信号和匹配的7位或10地址之后，I2C硬件将I2C状态寄存器0的ADDSEND位置1，此位应该通过软件轮询或者中断来检测，发现置起后，软件通过先读I2C_STAT0寄存器然后读I2C_STAT1寄存器来清除ADDSEND位。当ADDSEND位被清0时，I2C就开始接收来自I2C总线的数据。
3. 当接收到第一个字节时，RBNE位被硬件置1，软件可以读取I2C_DATA寄存器的第一个字节，此时RBNE位也被清0。
4. 任何时候RBNE被置1，软件可以从I2C_DATA寄存器读取一个字节。
5. 接收到最后一个字节后，RBNE被置1，软件可以读取最后的字节。
6. 当I2C检测到I2C总线上一个STOP信号，STPDET位被置1，软件通过先读I2C_STAT0寄存器再写I2C_CTL0寄存器来清除STPDET位。

**图 20-10. 从机接收模式(10 位地址模式)**




## 主机发送模式下的软件流程

如*图20-11. 主机发送模式(10位地址模式)*所示，在主机模式下发送数据到I2C总线时，软件应该遵循这些步骤来运行I2C模块：

1. 首先，软件应该使能I2C外设时钟，以及配置I2C_CTL1中时钟相关寄存器来确保正确的I2C时序。使能和配置以后，I2C运行在默认的从机模式状态，等待START信号，随后等待I2C总线寻址。
2. 软件将START位置1，在I2C总线上产生一个START信号。

3. 发送一个START信号后，I2C硬件将I2C_STAT0的SBSEND位置1然后进入主机模式。现在软件应该读I2C_STAT0寄存器然后写一个7位地址位或10位地址的地址头到I2C_DATA寄存器来清除SBSEND位。当SBSEND位被清0时，I2C就开始发送地址或者地址头到I2C总线。如果发送的地址是10位地址的地址头，硬件在发送地址头的时候会将ADD10SEND位置1，软件应该通过读I2C_STAT0寄存器然后写10位低地址到I2C_DATA来清除ADD10SEND位。

4. 7位或10位的地址位发送出去之后，I2C硬件将ADDSEND位置1，软件通过读I2C_STAT0寄存器然后读I2C_STAT1寄存器清除ADDSEND位。

5. I2C进入数据发送状态，因为移位寄存器和数据寄存器I2C_DATA都是空的，所以硬件将TBE位置1。此时软件可以写第一个字节数据到I2C_DATA寄存器，但是TBE位此时不会被清零，因为写入I2C_DATA寄存器的字节会被立即移入内部移位寄存器。当移位寄存器非空时，I2C就开始发送数据到总线。

6. 在第一个字节的发送过程中，软件可以写第二个字节到I2C_DATA，此时TBE会被清零，因为I2C_DATA寄存器和移位寄存器都不为空。

7. 任意时刻TBE被置1，软件都可以向I2C_DATA寄存器写入一个字节，只要还有数据待发送。

8. 在倒数第二个字节发送过程中，软件写入最后一个字节数据到I2C_DATA来清除TBE标志位，此后就不用关心TBE位的状态。TBE位会在倒数第二个字节发送完成后被置起，直到发送STOP信号时被清零。

9. 最后一个字节发送结束后，I2C主机将BTC位置起，因为移位寄存器和I2C_DATA寄存器此时都为空。软件此时应该配置STOP来发送一个STOP信号，此后TBE和BTC状态位都将被清0。

**图 20-11. 主机发送模式(10 位地址模式)**




## 主机接收模式下的软件流程

在主机接收模式下，主机需要为最后一个字节接收产生NACK，然后发送STOP信号。因此，需要特别注意以确保最后接收到数据的正确性。下面提供了两种针对主机接收模式的软件编程方案：方案A和B。方案A需要保证软件能对I2C事件进行快速响应，方案B则不需要。

### 方案 A

1. 首先，软件应该使能I2C外设时钟，以及配置I2C_CTL1中时钟相关寄存器来确保正确的I2C时序。使能和配置以后，I2C运行在默认的从机模式状态，等待START信号，随后等待I2C总线寻址。
2. 软件将START位置1，从而在I2C总线上产生一个START信号。
3. 发送一个START信号后，I2C硬件将I2C_STAT0寄存器的SBSEND位置1然后进入主机模式。现在软件应该读I2C_STAT0寄存器然后写一个7位地址位或10位地址的地址头到I2C_DATA寄存器来清除SBSEND位。当SBSEND位被清0时，I2C就开始发送地址或者地址头到I2C总线。如果发送的地址是10位地址的地址头，硬件在发送地址头的时候会先将ADD10SEND位置1，软件应该通过读I2C_STAT0寄存器然后写10位低地址到I2C_DATA

来清除ADD10SEND位。

4. 7位或10位的地址位发送出去之后，I2C硬件将ADDSEND位置1，软件应该通过读I2C_STAT0寄存器然后读I2C_STAT1寄存器清除ADDSEND位。如果地址是10位格式，软件应该再次将START位置1来重新产生一个START。在START产生后，SBSEND位会被置1。软件应该通过先读I2C_STAT0然后写地址头到I2C_DATA来清除SBSEND位，然后地址头被发到I2C总线，ADDSEND再次被置1。软件应该再次通过先读I2C_STAT0然后读I2C_STAT1来清除ADDSEND位。
5. 当接收到第一个字节时，硬件会将RBNE位置1。此时软件可以从I2C_DATA寄存器读取第一个字节，之后RBNE位被清0。
6. 此后任何时候RBNE被置1，软件就可以从I2C_DATA寄存器读取一个字节。
7. 接收完倒数第二个字节(N-1)数据之后，软件应该立即将ACKEN位清0，并将STOP位置1，这一过程需要在最后一个字节接收完毕之前完成，以确保NACK发送给最后一个字节。
8. 最后一个字节接收完毕后，RBNE位被置1，软件可以读取最后一个字节。由于ACKEN已经在前一步骤中被清0，I2C不再为最后一个字节发送ACK，并在最后一个字节发送完毕后产生一个STOP信号。

以上步骤要求字节数目N>1，如果N=1，步骤7应该在步骤4之后就执行，且需要在字节接收完成之前完成。

**图 20-12. 主机接收使用方案 A 模式(10 位地址模式)**




**方案 B**

1. 首先，软件应该使能I2C外设时钟，配置I2C_CTL1中时钟相关寄存器来确保正确的I2C时序。初始化完成之后，I2C运行在默认的从机模式状态，等待START信号和地址。
2. 软件将START位置1，从而在I2C总线上产生一个START信号。
3. 发送一个START信号后，I2C硬件将I2C_STAT0寄存器的SBSEND位置1然后进入主机模式。现在软件应该读I2C_STAT0寄存器然后写一个7位地址位或10位地址的地址头到I2C_DATA寄存器来清除SBSEND位。当SBSEND位被清0时，I2C就开始发送地址或者地址头到I2C总线。如果发送的地址是10位地址的地址头，硬件在发送地址头之后会将ADD10SEND位置1，软件应该通过读I2C_STAT0寄存器然后写10位低地址到I2C_DATA来清除ADD10SEND位。

4. 7位或10位的地址位发送出去之后，I2C硬件将ADDSEND位置1，软件应该通过读I2C_STAT0寄存器然后读I2C_STAT1寄存器清除ADDSEND位。如果地址是10位格式，软件应该接着将START位再次置1来产生一个开始信号，START被发送出去以后SBSEND位被再次置1。软件应该通过先读I2C_STAT0然后写地址头到I2C_DATA来清除SBSEND位，然后地址头被发到I2C总线，ADDSEND再次被置1。软件应该再次通过先读I2C_STAT0然后读I2C_STAT1来清除ADDSEND位。

5. 当第一个字节被接收时，RBNE位会被硬件置1。此时软件可从I2C_DATA寄存器读取出第一个字节，同时RBNE位被清0。

6. 此后任何时刻，只要RBNE位被置1，软件就可以从I2C_DATA寄存器读取一个字节的数据，直到主机接收了N-3个字节。

如*图 20-13. 主机接收使用方案 B 模式(10 位地址模式)*所示，第 N-2 个字节还没被软件读出，之后第 N-1 个字节被接收，此时 BTC 和 RBNE 都被置位，总线就会被主机锁死以阻止最后一个字节的接收。然后软件应该清除 ACKEN 位。

7. 软件从I2C_DATA读出倒数第三个(N-2)字节数据，同时也将BTC位清0。此后第N-1个字节从移位寄存器被移到I2C_DATA，总线得到释放然后开始接收最后一个字节，由于ACKEN已经被清除，因此主机不会给最后一个字节数据发送ACK响应。

8. 最后一个字节接收完毕后，硬件再次把BTC位和RBNE置1，并拉低SCL，软件将STOP位置1，主机发出一个STOP信号。

9. 软件读取第N-1个字节，清除BTC。此后最后一个字节从移位寄存器被移动到I2C_DATA。

10. 软件读取最后一个字节，清除RBNE。

以上步骤需要字节数字N>2，N=1和N=2的情况近似：

**N=1**

在第 4 步，软件应该在清除 ADDSEND 位之前将 ACKEN 位清 0，在清除 ADDSEND 位之后将 STOP 位置 1。当 N=1 时步骤 5 是最后一步。

**N=2**

在第2步，软件应该在START置1之前将POAP置1。在第4步，软件应该在清除ADDSEND位之前将ACKEN位清0。在第5步，软件应该一直等到BTC位被置1然后将STOP位置1且读取I2C_DATA两次。

**图 20-13. 主机接收使用方案 B 模式(10 位地址模式)**




## 20.3.8. SCL 线控制

SCL 线拉低功能是为了避免在接收时发生上溢错误以及在发送时发生下溢错误。如在软件编程模型中所示，在发送模式，当 TBE 和 BTC 被置位，发送器保持 SCL 线为低电平直到下一个发送数据写入传输缓冲区寄存器。在接收模式，当 RBNE 和 BTC 被置位，发送器保持 SCL 线为低电平直到传输缓冲区寄存器里的数据被读出。

当工作在从模式的时候，可以通过置位 I2C_CTL0 寄存器的 SS 位禁止 SCL 线拉低功能。如果该位置位，软件要能足够快的处理 TBE，RBNE 和 BTC 状态，否则上溢或下溢的情况可能会发生。

### 20.3.9. DMA 模式下数据传输

按照前面的软件流程，每当 TBE 位或 RBNE 位被置 1 之后，软件都应该写或读一个字节，这样将导致 CPU 的负荷较重。I2C 的 DMA 功能可以在 TBE 或 RBNE 位置 1 时，自动进行一次写或读操作，从而减轻了 CPU 的负荷，具体 DMA 的配置请参看 DMA 相关章节。

DMA 请求通过 I2C_CTL1 寄存器的 DMAON 位使能。该位应该在清除 ADDSEND 状态位之后被置位。如果一个从机的 SCL 线延长功能被禁止，DMAON 位应该在 ADDSEND 事件前被置位。

参考 DMA 控制器的关于 DMA 的配置方法说明。DMA 必须在 I2C 传输开始之前配置和使能。当指定个数的字节已经传输完成，DMA 会发送一个传输结束(EOT)信号给 I2C 接口，并产生一个 DMA 传输完成中断。

当主机接收两个或两个以上字节时，需将 I2C_CTL1 寄存器的 DMALST 位置位。在接收到最后一个字节之后，I2C 主机发送 NACK。在 DMA 传输完成中断 ISR 中，通过置位 STOP 位，产生一个停止信号。

当主机仅接收一个字节时，清除ADDSEND状态前ACKEN位必须被清除。在清除ADDSEND状态后或在DMA传输完成中断ISR中，通过置位STOP位，产生一个停止信号。

### 20.3.10. 报文错误校验

I2C模块中有一个PEC(包错误检查)模块，它使用CRC-8计算器来执行I2C数据的报文校验，CRC多项式为x8 + x2 + x + 1，和SMBus协议兼容。将PECEN位置1就可以使能PEC功能。PEC将会计算所有通过I2C总线发送的数据(包括地址)。软件可以通过配置PECTRANS来控制I2C在最后一个字节发送完毕后发送PEC值，或者在接收完成后检查接收到的PEC值是否正确。在DMA模式下，如果PECEN位和PECTRANS位被置1，I2C将自动发送或者检查PEC值。

### 20.3.11. SMBus 支持

系统管理总线(System Management Bus，简写为SMBus或SMB)是一种结构简单的单端双线制总线，可实现轻量级的通信需求。一般来说，SMBus最常见于计算机主板，主要用于电源传输ON / OFF指令的通信。SMBus是I2C的一种衍生总线形式，主要用于计算机主板上的低带宽设备间通信，尤其是与电源相关的芯片，例如笔记本电脑的可充电电池子系统(参见Smart Battery Data)。

#### SMBus 协议

SMBus上每个报文交互都遵从SMBus协议中预定义的格式。SMBus是I2C规范中数据传输格式

的子集。只要I2C设备可通过SMBus协议之一进行访问，便视为兼容SMBus规范。不符合这些协议的I2C设备，将无法被SMBus和ACPI规范所定义的标准方法访问。

### 地址解析协议

SMBus是基于I2C硬件实现的，它使用了I2C的硬件寻址方式，但在I2C的基础上增加了二级软件处理，建立自己独特的系统。比较特别的是SMBus规范包含一个地址解析协议，可用于实现动态地址分配。动态识别硬件和软件使得总线设备能够支持热插拔，无需重启系统便能即插即用。总线中的设备将被自动识别并分配唯一地址。这个优点非常有利于实现即插即用的用户接口。在此协议中，系统中的host与设备之间有一个重要的区别，即host具有分配地址的功能。

### 超时特性

SMBus有一种超时特性：假如某个通信耗时太久，便会自动复位设备。这就解释了为什么最小时钟周期为10kHz--为了防止长时间锁死总线。I2C在本质上可以视为一个"直流"总线，也就是说当主机正在访问从机的时候，假如从机正在执行一些子程序无法及时响应，从机可以拉住主机的时钟。这样便可以提醒主机：从机正忙，但并不想放弃当前的通信。从机的当前任务结束之后，将可以继续I2C通信。I2C总线协议中并没有限制这个延时的上限，但在SMBus系统中，这个时间被限定为25ms。按照SMBus协议的假定，如果某个通信耗时太久，就意味着总线出了问题，此时所有设备都应当复位以消除这种问题。这样就并不允许从设备将时钟拉低太长时间。

### 报文错误校验

SMBus 2.0以及1.1都采用了报文错误校验(Packet Error Checking，缩写为PEC)。在这种模式中，每次通信最后都将传输PEC字节。该字节是按照CRC-8校验和的方式计算的，计算范围包括整个报文，包括地址以及读/写位。所采用的多项式为x8+x2+x+1(CRC-8-ATM HEC算法，初始化为0)。

### SMBus 警报

SMBus还有一个额外的中断信号，称为SMBALERT#。从机上发生事件后，可通过这个信号通知主机来访问从机。SMBus中还定义了较少见的"主机提醒协议"，基于I2C多主模式实现类似的提醒功能，但是可以传递更多数据。

**注意:** 当使用SMBus主机模式时，默认检查I2C_SMBA引脚，即使不配置端口配置为AFIO(I2C0对应PB5，I2C1对应PB12，I2C2对应PA9)，当使能错误中断时且I2C_SMBA引脚为低电平时会产生中断。因此当不需要 SMBus 警报时应规避使用I2C_SMBA管脚，且保证相关IO是高电平。

### SMBus 通讯流程

SMBus的通讯流程和标准I2C的流程相似。为了使用SMBus模式，在程序中需要配置几个

SMBus特定的寄存器，响应一些SMBus特定标志位，实现那些在SMBus手册中介绍的上层协议。

1. 在通信之前，需要将I2C_CTL0中SMBEN位置1，并且根据需求，配置SMBSEL和ARPEN的值。
2. 为了支持ARP协议(ARPEN=1)，在SMBus主机模式下(SMBSEL=1)，软件需要响应标志位HSTSMB(在SMBus从机模式下，响应DEFSMB标志位)，并实现ARP协议中的功能。
3. 为了支持SMBus警告模式，软件应该响应SMBALT标志位，并实现相应的功能。

### 20.3.12. SAM_V 支持

为了支持 SAM_V 标准，I2C 模块增加两个附加的引脚：txframe 和 rxframe。Txframe 是一个输出引脚，在主机模式下，当该引脚输出电平为高电平时，表示 I2C 是忙的。Rxframe 是一个输入引脚，应该与 SMBALERT 信号一起多路复用。

SAM_V 模式通过置位 I2C_SAMCS 寄存器的 SAMEN 位使能。txframe 和 rxframe 引脚的状态可以通过 I2C_SAMCS 寄存器的 RFR，RFF，TFR，TFF，RXF 和 TXF 标志反映。如果对应的中断使能位置位，将产生 I2C 中断。

### 20.3.13. 状态、错误和中断

I2C有一些状态、错误标志位，通过设置一些寄存器位，便可以从这些标志触发中断(详情参见*I2C寄存器*)。

**表 20-2. 事件状态标志位**

| 事件标志位名称 | 说明 |
|---|---|
| SBSEND | 主机发送 START 信号 |
| ADDSEND | 地址发送和接收 |
| ADD10SEND | 10 位地址模式中地址头发送 |
| STPDET | 监测到 STOP 信号 |
| BTC | 字节发送结束 |
| TBE | 发送时 I2C_DATA 为空 |
| RBNE | 接收时 I2C_DATA 非空 |
| RFR | SAM_V 模式时检测到 rxframe 上升沿 |
| RFF | SAM_V 模式时检测到 rxframe 下降沿 |
| TFR | SAM_V 模式时检测到 txframe 上升沿 |
| TFF | SAM_V 模式时检测到 txframe 下降沿 |

**表 20-3. 错误标志位**

| 错误名称 | 说明 |
|---|---|
| BERR | 总线错误 |
| LOSTARB | 仲裁丢失 |
| OUERR | 当禁用 SCL 拉低后，发生了上溢或下溢 |
| AERR | 没有接收到应答 |

| 错误名称 | 说明 |
| --- | --- |
| PECERR | CRC 值不相同 |
| SMBTO | SMBus 模式下总线超时 |
| SMBALT | SMBus 警报 |

## 20.4. I2C 寄存器

I2C0基地址：0x4000 5400

I2C1基地址：0x4000 5800

I2C2基地址：0x4000 5C00

### 20.4.1. 控制寄存器 0(I2C_CTL0)

地址偏移：0x00
复位值：0x0000 0000

该寄存器可以按半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SRESET | 保留 | SALT | PECTRANS | POAP | ACKEN | STOP | START | SS | GCEN | PECEN | ARPEN | SMBSEL | 保留 | SMBEN | I2CEN |
| rw | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15 | SRESET | 软件复位 I2C，软件应该在 I2C 总线被释放时复位 I2C。<br>0：I2C 未复位<br>1：I2C 复位 |
| 14 | 保留 | 必须保持复位值。 |
| 13 | SALT | SMBus 警报<br>通过 SMBA 引脚发出警报。<br>软件置 1 和清 0，硬件清 0。<br>0：不通过 SMBA 发布警告<br>1：通过 SMBA 引脚发送警告 |
| 12 | PECTRANS | PEC 传输<br>软件置 1 和清 0，硬件在以下条件下清除此位：PEC 传输完成，或监测到 START / STOP 信号，或 I2CEN=0。<br>0：不传输 PEC 值<br>1：传输的 PEC 值 |
| 11 | POAP | ACK / PEC 的位置含义<br>软件置 1 和清 0，当 I2CEN=0 时，硬件清 0。<br>0：ACKEN 位决定对当前正在接收的字节是否发送 ACK，PECTRANS 位表明正在接收的字节是否为 PEC。 |

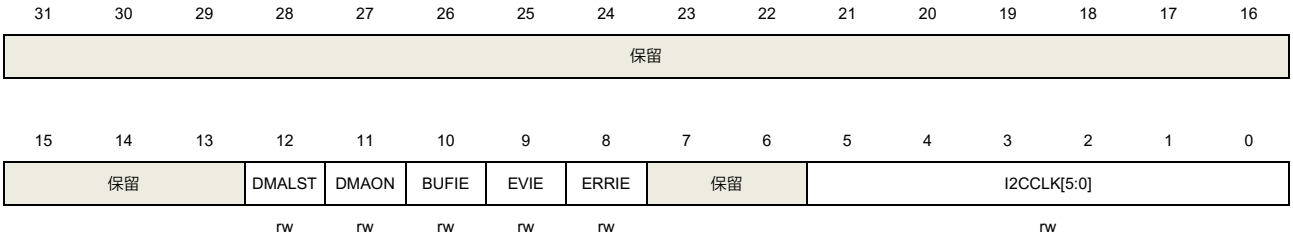
| | | |
|---|---|---|
| | | 1：ACKEN 位决定是否对下一个字节发送 ACK，PECTRANS 位表明下一个即将被接收的字节是 PEC。 |
| 10 | ACKEN | 是否发送 ACK<br>软件置 1 和清 0，当 I2CEN=0 时硬件清 0。<br>0：不发送 ACK<br>1：发送 ACK |
| 9 | STOP | I2C 总线上产生一个 STOP 信号<br>软件置 1 和清 0，SMBus 超时时，硬件置 1，监测到 STOP 信号时，硬件清 0。<br>0：不发送 STOP<br>1：发送 STOP |
| 8 | START | I2C 总线上产生一个 START 信号<br>软件置 1 和清 0，当监测到 START 信号或 I2CEN=0 时由硬件清 0。<br>0：不发送 START<br>1：发送 START |
| 7 | SS | 在从机模式下数据未就绪是否将 SCL 拉低<br>软件置 1 和清 0。<br>0：拉低 SCL<br>1：不拉低 SCL |
| 6 | GCEN | 是否响应对地址(0x00)的广播呼叫<br>0：从机不响应广播呼叫<br>1：从机将响应广播呼叫 |
| 5 | PECEN | PEC 计算使能<br>0：PEC 计算禁能<br>1：PEC 计算使能 |
| 4 | ARPEN | SMBus 下 ARP 协议使能<br>0：ARP 禁能<br>1：ARP 使能 |
| 3 | SMBSEL | SMBus 类型选择<br>0：从机<br>1：主机 |
| 2 | 保留 | 必须保持复位值 |
| 1 | SMBEN | SMBus / I2C 模式开关<br>0：I2C 模式<br>1：SMBus 模式 |
| 0 | I2CEN | I2C 外设使能<br>0：禁能 I2C<br>1：使能 I2C |

### 20.4.2. 控制寄存器 1(I2C_CTL1)

地址偏移：0x04

复位值：0x0000 0000

该寄存器可以按半字(16位)或字(32位)访问。



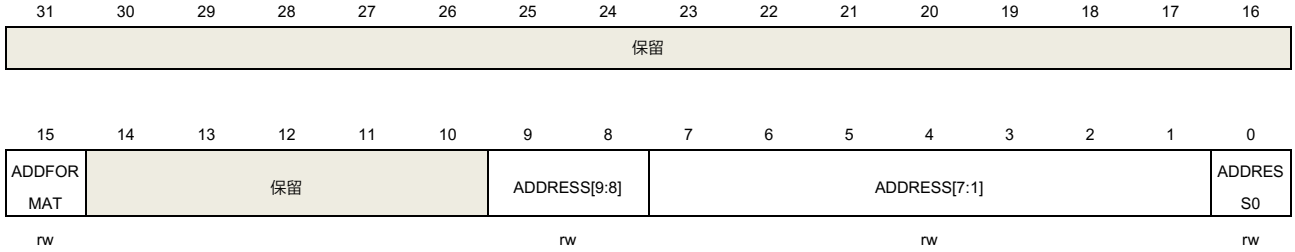

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:13 | 保留 | 必须保持复位值。 |
| 12 | DMALST | DMA 最后传输配置<br>0：下一个 DMA EOT 不是最后传输<br>1：下一个 DMA EOT 是最后传输 |
| 11 | DMAON | DMA 模式开关<br>0：DMA 模式关<br>1：DMA 模式开 |
| 10 | BUFIE | 缓冲区中断使能<br>0：禁用缓冲区中断，如果 EVIE=1，当 TBE = 1 或 RBNE = 1 时不产生中断。<br>1：使能缓冲区中断，如果 EVIE=1，当 TBE = 1 或 RBNE = 1 时产生中断。 |
| 9 | EVIE | 事件中断使能<br>0：禁用事件中断<br>1: 使能事件中断，意味着当 SBSEND、ADDSEND、ADD10SEND、STPDET 或 BTC 标志位有效或当 BUFIE=1 时 TBE=1 或 RBNE=1 时产生中断。 |
| 8 | ERRIE | 错误中断使能<br>0：禁用错误中断<br>1: 使能错误中断，意味着当 BERR、LOSTARB、AERR、OUERR、PECERR、SMBTO 或 SMBALT 标志位生效时产生中断。 |
| 7:6 | 保留 | 必须保持复位值。 |
| 5:0 | I2CCLK[5:0] | I2C 外设时钟频率<br>I2CCLK[5:0]应该是输入 APB1 时钟频率，最低 2MHz。<br>000000 - 000001：无时钟<br>000010 - 111100：2MHz~60MHz<br>111101 - 111111：由于 APB1 时钟限制，无时钟<br>**注意：**<br>在标准模式下，APB1 时钟频率需大于或者等于 2MHz。在快速模式下，APB1 时钟 |

频率需大于或者等于 8MHz。

### 20.4.3. 从机地址寄存器 0(I2C_SADDR0)

地址偏移：0x08
复位值：0x0000 0000

该寄存器可以按半字(16位)或字(32位)访问。

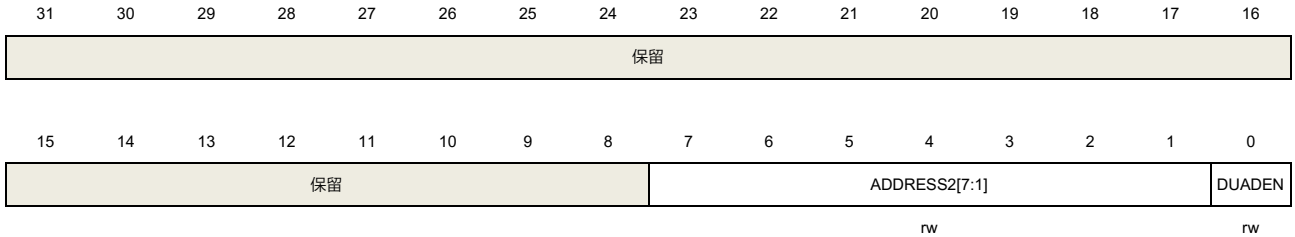



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15 | ADDFORMAT | I2C 从机地址格式<br>0：7 位地址<br>1：10 位地址 |
| 14:10 | 保留 | 必须保持复位值。 |
| 9:8 | ADDRESS[9:8] | 10 位地址的最高两位 |
| 7:1 | ADDRESS[7:1] | 7 位地址或者 10 位地址的第 7-1 位 |
| 0 | ADDRESS0 | 10 位地址的第 0 位 |

### 20.4.4. 从机地址寄存器 1(I2C_SADDR1)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器可以按半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 7:1 | ADDRESS2[7:1] | 从机在双重地址模式下第二个 I2C 地址 |
| 0 | DUADEN | 双重地址模式使能<br>0：禁用双重地址模式<br>1：使能双重地址模式 |

### 20.4.5. 传输缓冲区寄存器(I2C_DATA)

地址偏移：0x10
复位值：0x0000 0000

该寄存器可以按半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7:0 | TRB[7:0] | 数据发送接收缓冲区 |

### 20.4.6. 传输状态寄存器 0(I2C_STAT0)

地址偏移：0x14
复位值：0x0000 0000

该寄存器可以按半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SMBALT | SMBTO | 保留 | PECERR | OUERR | AERR | LOSTARB | BERR | TBE | RBNE | 保留 | STPDET | ADD10SEND | BTC | ADDSEND | SBSEND |
| rc_w0 | rc_w0 | | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 | r | r | | r | r | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15 | SMBALT | SMBus 警报状态<br>硬件置 1，软件写 0 清 0。<br>0：SMBA 引脚未被拉低(从机模式)或未监测到警报(主机模式)<br>1：SMBA 引脚被拉低且接收到警报地址(从机模式)或监测到警报(主机模式) |
| 14 | SMBTO | SMBus 模式下超时信号 |

| | | |
|---|---|---|
| | | 硬件置 1，软件写 0 清 0。<br>0：无超时错误<br>1：超时事件发生(SCL 被拉低达 25ms) |
| 13 | 保留 | 必须保持复位值。 |
| 12 | PECERR | 接收数据时 PEC 错误<br>硬件置 1，软件写 0 清 0。<br>0：接收到 PEC 且校验正确<br>1：接收到 PEC 但检验错误，此时不管 ACKEN 位的值，I2C 将发送 NACK |
| 11 | OUERR | 当禁用 SCL 拉低功能后，在从机模式下发生了上溢或下溢事件。在从机接收模式下，假如 I2C_DATA 中的最后一字节并未被读出，并且后续字节又接收完成，就会发生上溢错误。在从机发送模式下，假如当前字节已经发送完成，而 I2C_DATA 仍然为空，就会发生下溢错误。<br>硬件置 1，软件写 0 清 0。<br>0：无上溢或下溢错误发生<br>1：发生上溢或下溢错误 |
| 10 | AERR | 应答错误<br>硬件置 1，软件写 0 清 0。<br>0：未发生应答错误<br>1：发生了应答错误 |
| 9 | LOSTARB | 主机模式下仲裁丢失<br>硬件置 1，软件写 0 清 0。<br>0：无仲裁丢失<br>1：发生仲裁丢失，I2C 模块返回从机模式。 |
| 8 | BERR | 总线错误，表示 I2C 总线上发生了预料之外的 START 信号或 STOP 信号。<br>硬件置 1，软件写 0 清 0。<br>0：无总线错误<br>1：发生了总线错误 |
| 7 | TBE | 发送期间 I2C_DATA 为空<br>硬件从 I2C_DATA 寄存器移动一个字节到移位寄存器之后将此位置 1，软件写一个字节到 I2C_DATA 寄存器清除该位。如果移位寄存器和 I2C_DATA 寄存器都是空的，写 I2C_DATA 寄存器将不会清除 TBE 位(详见主机/从机发送模式下的软件操作流程)<br>0：I2C_DATA 非空<br>1：I2C_DATA 空，软件可以写 |
| 6 | RBNE | 接收期间 I2C_DATA 非空<br>硬件从移位寄存器移动一个字节到 I2C_DATA 寄存器之后将此位置 1，读 I2C_DATA 可以清除此位。如果 BTC 和 RBNE 都被置 1，读 I2C_DATA 将不会清除 RBNE，因为移位寄存器的字节将被立即移到 I2C_DATA。<br>0：I2C_DATA 为空<br>1：I2C_DATA 非空，软件可以读 |

| | | |
|---|---|---|
| 5 | 保留 | 必须保持复位值。 |
| 4 | STPDET | 从机模式下监测到 STOP 信号<br>此位被硬件置 1，先读 I2C_STAT0 然后写 I2C_CTL0 可以清除此位。<br>0：从机模式下未监测到 STOP 信号<br>1：从机模式下监测到 STOP 信号 |
| 3 | ADD10SEND | 主机模式下 10 位地址的地址头被发送<br>该位由硬件置 1，软件读 I2C_STAT0 和写 I2C_DATA 清除此位。<br>0：主机模式下未发送 10 位地址的地址头<br>1：主机模式下发送 10 位地址的地址头 |
| 2 | BTC | 字节发送结束<br>接收模式下，如果一个字节已经被移位寄存器接收但是此时 I2C_DATA 寄存器仍然是满的；或者发送模式下，一个字节已经被移位寄存器发送但是 I2C_DATA 寄存器仍然是空的，如果使能了 SCL 拉低功能，硬件就会置起 BTC 标志。<br>此位由硬件置 1。<br>可由以下三种方式清除：<br>1、软件清除：读 I2C_STAT0，然后读或写 I2C_DATA 寄存器清除此位<br>2、硬件清除：发送一个 STOP 或 START 信号<br>3、寄存器 I2C_CTL0 中 I2CEN=0<br>0：未发生 BTC<br>1：发生了 BTC |
| 1 | ADDSEND | 主机模式下：成功发送了地址并收到 ACK<br>从机模式下：接收到的地址与自身的地址匹配<br>此位由硬件置 1，软件读 I2C_STAT0 寄存器和读 I2C_STAT1 清 0。<br>0：从机模式下，未收到地址或者收到的地址不匹配；主机模式下，无地址被发送或地址已发送但未收到从机回复的 ACK<br>1：从机模式下，接收到的地址与自身的地址匹配；主机模式下，地址已发送并收到 ACK |
| 0 | SBSEND | 主机模式下发送 START 信号<br>此位由硬件置 1，软件读 I2C_STAT0 和写 I2C_DATA 清 0。<br>0：未发送 START 信号<br>1：START 信号被发送 |

### 20.4.7. 传输状态寄存器 1(I2C_STAT1)

地址偏移：0x18
复位值：0x0000 0000

该寄存器可以按半字(16位)或字(32位)访问。

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

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

| PECV[7:0] | DUMODF | HSTSMB | DEFSMB | RXGC | 保留 | TR | I2CBSY | MASTER |
|---|---|---|---|---|---|---|---|---|
| r | r | r | r | r | | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:8 | PECV[7:0] | 当 PEC 使能后硬件计算出的 PEC 值。 |
| 7 | DUMODF | 从机模式下双标志位表明哪个地址和双地址模式匹配<br>STOP 或 START 信号产生后或 I2CEN=0 时此位由硬件清 0。<br>0：地址和 I2C_SADDR0 匹配<br>1：地址和 I2C_SADDR1 匹配 |
| 6 | HSTSMB | 从机模式下监测到 SMBus 主机地址头<br>STOP 或 START 信号产生后或 I2CEN=0 时此位由硬件清 0。<br>0：未监测到 SMBus 主机地址头<br>1：监测到 SMBus 主机地址头 |
| 5 | DEFSMB | SMBus 设备缺省地址<br>STOP 或 START 信号产生后或 I2CEN=0 时此位由硬件清 0。<br>0：SMBus 设备没有接收到缺省地址<br>1：SMBus 设备接收到缺省地址 |
| 4 | RXGC | 是否接收到广播地址(0x00)<br>STOP 或 START 信号产生后或 I2CEN=0 时此位由硬件清 0。<br>0：未接收到广播呼叫地址<br>1：接收到广播呼叫地址 |
| 3 | 保留 | 必须保持复位值。 |
| 2 | TR | 发送端或接收端<br>该位表明 I2C 作为发送端还是接收端。STOP 或 START 信号产生后或 I2CEN 或 LOSTARB=1 时此位由硬件清 0。<br>0：接收端<br>1：发送端 |
| 1 | I2CBSY | 忙标志<br>STOP 信号后硬件清 0。<br>0：无 I2C 通讯<br>1：I2C 正在通讯 |
| 0 | MASTER | 主机模式<br>表明 I2C 时钟在主机模式还是从机模式的标志位。<br>该位在 START 信号产生后由硬件置 1。<br>该位在 STOP 信号产生后或 I2CEN=0 或 LOSTARB=1 时此位由硬件清 0。<br>0：从机模式<br>1：主机模式 |

### 20.4.8. 时钟配置寄存器(I2C_CKCFG)

地址偏移：0x1C
复位值：0x0000 0000

该寄存器可以按半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15 | FAST | 主机模式下 I2C 速度选择<br>0：标准速度<br>1：快速 |
| 14 | DTCY | 快速模式下占空比<br>0：$T_{low}/T_{high}=2$<br>1：$T_{low}/T_{high}=16/9$ |
| 13:12 | 保留 | 必须保持复位值。 |
| 11:0 | CLKC[11:0] | 主机模式下 I2C 时钟控制<br>标准速度模式下：$T_{high}=T_{low}=CLKC*T_{PCLK1}$<br>如果 DTCY=0，快速模式下：<br>$T_{high}=CLKC*T_{PCLK1}$，$T_{low}=2*CLKC*T_{PCLK1}$<br>如果 DTCY=1，快速模式下：<br>$T_{high}=9*CLKC*T_{PCLK1}$，$T_{low}=16*CLKC*T_{PCLK1}$<br>**注意：**如果 DTCY=0，当 PCLK1 为 3 的整数倍时，波特率会比较准确。如果 DTCY=1，当 PCLK1 为 25 的整数倍时，波特率会比较准确。 |

### 20.4.9. 上升时间寄存器(I2C_RT)

地址偏移：0x20
复位值：0x0000 0002

该寄存器可以按半字(16位)或字(32位)访问。

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

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

rw

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:6 | 保留 | 必须保持复位值。 |
| 5:0 | RISETIME[5:0] | 主机模式下最大上升时间<br>RISETIME 值应该为 SCL 最大上升时间加 1 |

## 20.4.10. SAM 控制状态寄存器(I2C_SAMCS)

地址偏移：0x80
复位值：0x0000 0000

该寄存器可以按半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RFR | RFF | TFR | TFF | 保留 | | RXF | TXF | RFRIE | RFFIE | TFRIE | TFFIE | 保留 | | STOEN | SAMEN |
| rc_w0 | rc_w0 | rc_w0 | rc_w0 | | | r | r | rw | rw | rw | rw | | | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15 | RFR | 接收帧上升沿标志，由软件写 0 清 0 |
| 14 | RFF | 接收帧下降沿标志，由软件写 0 清 0 |
| 13 | TFR | 发送帧上升沿标志，由软件写 0 清 0 |
| 12 | TFF | 发送帧下降沿标志，由软件写 0 清 0 |
| 11:10 | 保留 | 必须保持复位值。 |
| 9 | RXF | 接收帧信号电平 |
| 8 | TXF | 发送帧信号电平 |
| 7 | RFRIE | 接收帧上升沿中断使能<br>0：接收帧上升沿中断失能<br>1：接收帧上升沿中断使能 |
| 6 | RFFIE | 接收帧下降沿中断使能<br>0：接收帧下降沿中断失能<br>1：接收帧下降沿中断使能 |
| 5 | TFRIE | 发送帧上升沿中断使能<br>0：发送帧上升沿中断失能<br>1：发送帧上升沿中断使能 |

| | | |
|---|---|---|
| 4 | TFFIE | 发送帧下降沿中断使能<br>0：发送帧下降沿中断失能<br>1：发送帧下降沿中断使能 |
| 3:2 | 保留 | 必须保持复位值。 |
| 1 | STOEN | SAM_V 接口超时检测使能<br>0：SAM_V 接口超时检测失能<br>1：SAM_V 接口超时检测使能 |
| 0 | SAMEN | SAM_V 接口使能<br>0：SAM_V 接口失能<br>1：SAM_V 接口使能 |

# 21. 串行外设接口/片上音频接口(SPI/I2S)

## 21.1. 简介

SPI/I2S 模块可以通过 SPI 协议或 I2S 音频协议与外部设备进行通信。

串行外设接口(Serial Peripheral Interface，缩写为 SPI)提供了基于 SPI 协议的数据发送和接收功能，可以工作于主机或从机模式。SPI 接口支持具有硬件 CRC 计算和校验的全双工和单工模式。SPI5 还支持 SPI 四线主机模式。

片上音频接口(Inter-IC Sound，缩写为 I2S)支持四种音频标准，分别是 I2S 飞利浦标准，MSB 对齐标准，LSB 对齐标准和 PCM 标准。它可以在四种模式下运行，包括主机发送模式，主机接收模式，从机发送模式和从机接收模式。通过使用附加的 I2S 模块：I2S1_ADD 和 I2S2_ADD，SPI1 和 SPI2 支持 I2S 全双工模式。

## 21.2. 主要特性

### 21.2.1. SPI 主要特性

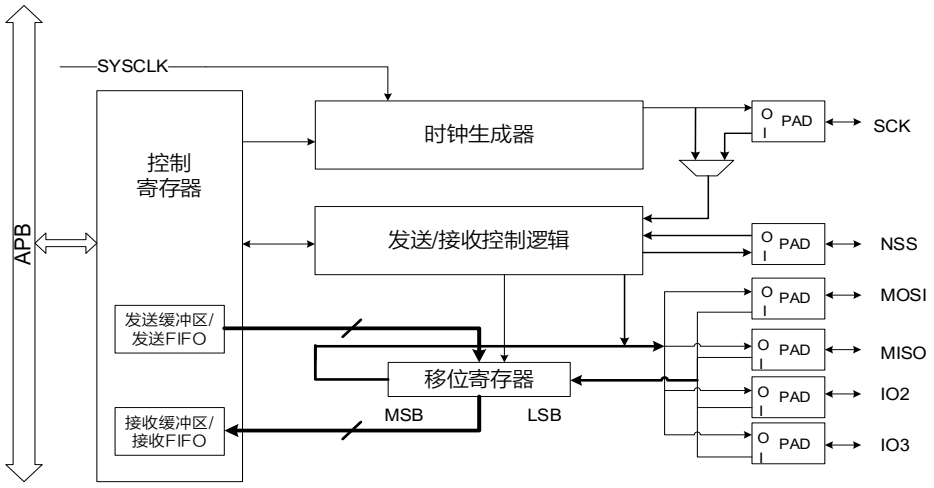
- 具有全双工和单工模式的主从操作；
- 16位宽度，独立的发送和接收缓冲区；
- 8位或16位数据帧格式；
- 低位在前或高位在前的数据位顺序；
- 软件和硬件NSS管理；
- 硬件CRC计算、发送和校验；
- 发送和接收支持DMA模式；
- 支持SPI TI模式；
- 支持SPI四线功能的主机模式(只有SPI5)。

### 21.2.2. I2S 主要特性

- 具有发送和接收功能的主从操作；
- 具有全双工模式的主从操作(仅在SPI1和SPI2中)；
- 支持四种I2S音频标准：飞利浦标准，MSB对齐标准，LSB对齐标准和PCM标准；
- 数据长度可以为16位，24位和32位；
- 通道长度为16位或32位；
- 16位缓冲区用于发送和接收；
- 通过I2S时钟分频器，可以得到8 kHz到192 kHz的音频采样频率；
- 可编程空闲状态时钟极性；
- 可以输出主时钟(MCK)；
- 发送和接收支持DMA功能。

## 21.3. SPI 结构框图

图 21-1. SPI 结构框图




## 21.4. SPI 信号线描述

### 21.4.1. 常规配置(非 SPI 四线模式)

表 21-1. SPI 信号描述

| 引脚名称 | 方向 | 描述 |
|---|---|---|
| SCK | I/O | 主机：SPI 时钟输出<br>从机：SPI 时钟输入 |
| MISO | I/O | 主机：数据接收线<br>从机：数据发送线<br>主机双向线模式：不使用<br>从机双向线模式：数据发送和接收线 |
| MOSI | I/O | 主机：数据发送线<br>从机：数据接收线<br>主机双向线模式：数据发送和接收线<br>从机双向线模式：不使用 |
| NSS | I/O | 软件 NSS 模式：不使用<br>主机硬件 NSS 模式：NSSDRV=1 时，为 NSS 输出，适用于单主机模式；NSSDRV=0 时，为 NSS 输入，适用于多主机模式。<br>从机硬件 NSS 模式：为 NSS 输入，作为从机的片选信号 |

### 21.4.2. SPI 四线配置

SPI 默认配置为单路模式，当 SPI_QCTL 中的 QMOD 位置 1 时，配置为 SPI 四线模式(只适

用于 SPI5)。SPI 四线模式只能工作在主机模式。

在 SPI 四线模式下，SPI 通过以下 6 个引脚与外部设备连接：

**表 21-2. SPI 四线信号描述**

| 引脚名称 | 方向 | 描述 |
|---|---|---|
| SCK | O | SPI 时钟输出 |
| MOSI | I/O | 发送或接收数据 0 |
| MISO | I/O | 发送或接收数据 1 |
| IO2 | I/O | 发送或接收数据 2 |
| IO3 | I/O | 发送或接收数据 3 |
| NSS | O | NSS 输出 |

## 21.5. SPI 功能描述

### 21.5.1. SPI 时序和数据帧格式

SPI_CTL0 寄存器中的 CKPL 位和 CKPH 位决定了 SPI 时钟和数据信号的时序。CKPL 位决定了空闲状态时 SCK 的电平，CKPH 位决定了第一个或第二个时钟跳变沿为有效采样边沿。在 TI 模式下，这两位没有意义。

**图 21-2. 常规模式下的 SPI 时序图**

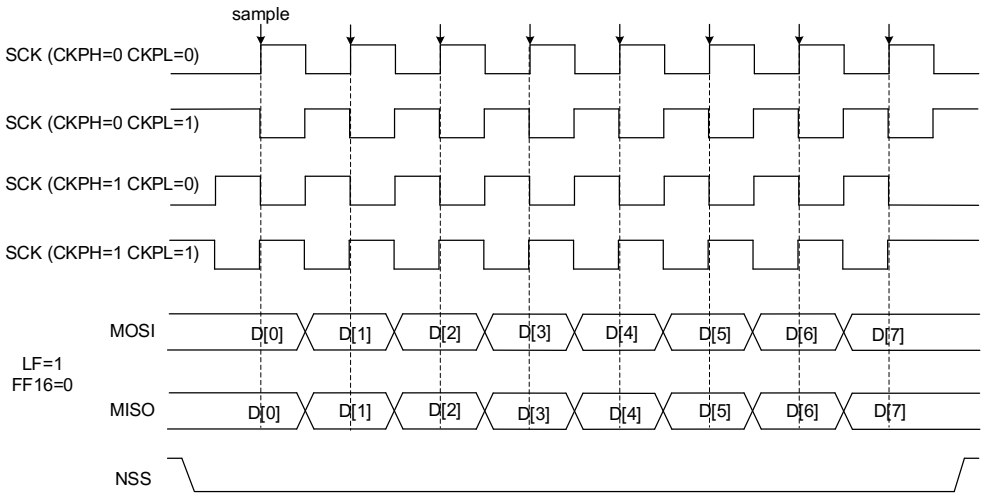
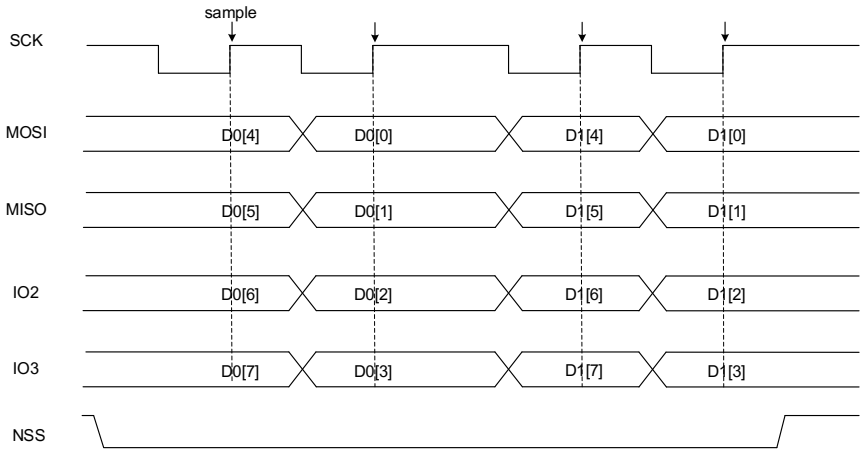


**图 21-3. SPI 四线模式下的 SPI 时序图(CKPL=1, CKPH=1, LF=0)**




在常规模式中，通过 SPI_CTL0 中的 FF16 位配置数据长度，当 FF16=1 时，数据长度为 16 位，否则为 8 位。在 SPI 四线模式下，数据帧长度固定为 8 位。

通过设置 SPI_CTL0 中的 LF 位可以配置数据顺序，当 LF=1 时，SPI 先发送 LSB 位，当 LF=0 时，则先发送 MSB 位。在 TI 模式中，数据顺序固定为先发 MSB 位。

## 21.5.2. NSS 功能

### 从机模式

当配置为从机模式(MSTMOD = 0)时，在硬件 NSS 模式(SWNSSEN = 0)下，SPI 从 NSS 引脚获取 NSS 电平，在软件 NSS 模式(SWNSSEN = 1)下，SPI 根据 SWNSS 位得到 NSS 电平。只有当 NSS 为低电平时，才能发送或接收数据。在软件 NSS 模式下，不使用 NSS 引脚。

**表 21-3. 从机模式 NSS 功能**

| 模式 | 寄存器配置 | 描述 |
|---|---|---|
| 从机硬件 NSS 模式 | MSTMOD = 0<br>SWNSSEN = 0 | SPI 从机 NSS 电平从 NSS 引脚获取。 |
| 从机软件 NSS 模式 | MSTMOD = 0<br>SWNSSEN = 1 | SPI 从机 NSS 电平由 SWNSS 位决定。<br>SWNSS = 0：NSS 电平为低<br>SWNSS = 1：NSS 电平为高 |

### 主机模式

在主机模式(MSTMOD=1)下，如果应用程序使用多主机连接方式，NSS 可以配置为硬件输入模式(SWNSSEN=0, NSSDRV=0)或者软件模式(SWNSSEN=1)。一旦 NSS 引脚(在硬件 NSS 模式下)或 SWNSS 位(在软件 NSS 模式下)被拉低，SPI 将自动进入从机模式，并且产生主机配置错误，CONFERR 位置 1。

如果应用程序希望使用 NSS 引脚控制 SPI 从设备，NSS 应该配置为硬件输出模式(SWNSSEN=0, NSSDRV=1)。使能 SPI 之后，NSS 保持高电平，当发送或接收过程开始时，NSS 变为低电平。

应用程序可以使用一个通用 I/O 口作为 NSS 引脚，以实现更加灵活的 NSS 应用。

**表 21-4. 主机模式 NSS 功能**

| 模式 | 寄存器配置 | 描述 |
|---|---|---|
| 主机硬件 NSS 输出模式 | MSTMOD = 1<br>SWNSSEN = 0<br>NSSDRV=1 | 适用于单主机模式，主机使用 NSS 引脚控制 SPI 从设备，此时 NSS 配置为硬件输出模式。使能 SPI 后 NSS 为低电平。 |
| 主机硬件 NSS 输入模式 | MSTMOD = 1<br>SWNSSEN = 0<br>NSSDRV=0 | 适用于多主机模式，此时 NSS 配置为硬件输入模式，一旦 NSS 引脚被拉低，SPI 将自动进入从机模式，并且产生主机配置错误，CONFERR 位置 1。 |
| 主机软件 NSS 模式 | MSTMOD = 1<br>SWNSSEN = 1<br>SWNSS = 0<br>NSSDRV：不要求 | 适用于多主机模式，一旦 SWNSS = 0，SPI 将自动进入从机模式，并且产生主机配置错误，CONFERR 位置 1。 |
| | MSTMOD = 1<br>SWNSSEN = 1<br>SWNSS = 1<br>NSSDRV：不要求 | 从机可以使用硬件或软件 NSS 模式 |

### 21.5.3. SPI 运行模式

**表 21-5. SPI 运行模式**

| 模式 | 描述 | 寄存器配置 | 使用的数据引脚 |
|---|---|---|---|
| MFD | 全双工主机模式 | MSTMOD = 1<br>RO = 0<br>BDEN = 0<br>BDOEN：不要求 | MOSI：发送<br>MISO：接收 |
| MTU | 单向线连接主机发送模式 | MSTMOD = 1<br>RO = 0<br>BDEN = 0<br>BDOEN：不要求 | MOSI：发送<br>MISO：不使用 |
| MRU | 单向线连接主机接收模式 | MSTMOD = 1<br>RO = 1<br>BDEN = 0<br>BDOEN：不要求 | MOSI：不使用<br>MISO：接收 |
| MTB | 双向线连接主机发送模式 | MSTMOD = 1<br>RO = 0<br>BDEN = 1<br>BDOEN = 1 | MOSI：发送<br>MISO：不使用 |

| 模式 | 描述 | 寄存器配置 | 使用的数据引脚 |
|---|---|---|---|
| MRB | 双向线连接主机接收模式 | MSTMOD = 1<br>RO = 0<br>BDEN = 1<br>BDOEN = 0 | MOSI：接收<br>MISO：不使用 |
| SFD | 全双工从机模式 | MSTMOD = 0<br>RO = 0<br>BDEN = 0<br>BDOEN：不要求 | MOSI：接收<br>MISO：发送 |
| STU | 单向线连接从机发送模式 | MSTMOD = 0<br>RO = 0<br>BDEN = 0<br>BDOEN：不要求 | MOSI：不使用<br>MISO：发送 |
| SRU | 单向线连接从机接收模式 | MSTMOD = 0<br>RO = 1<br>BDEN = 0<br>BDOEN：不要求 | MOSI：接收<br>MISO：不使用 |
| STB | 双向线连接从机发送模式 | MSTMOD = 0<br>RO = 0<br>BDEN = 1<br>BDOEN = 1 | MOSI：不使用<br>MISO：发送 |
| SRB | 双向线连接从机接收模式 | MSTMOD = 0<br>RO = 0<br>BDEN = 1<br>BDOEN = 0 | MOSI：不使用<br>MISO：接收 |

**图 21-4. 典型的全双工模式连接**

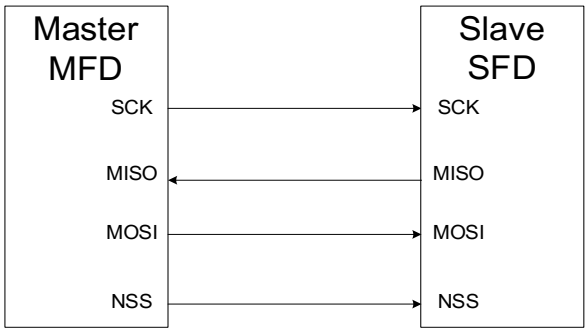


图 21-5. 典型的单工模式连接(主机：接收，从机：发送)

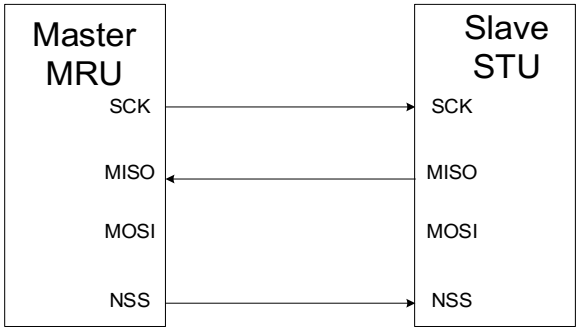



图 21-6. 典型的单工模式连接(主机：只发送，从机：接收)

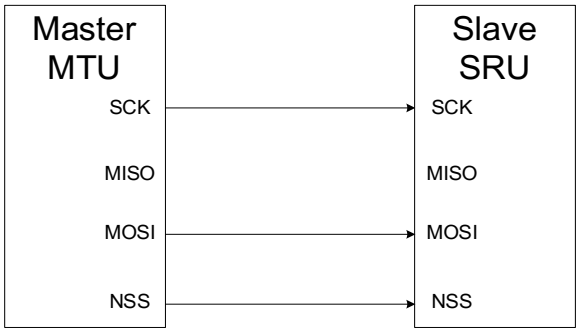
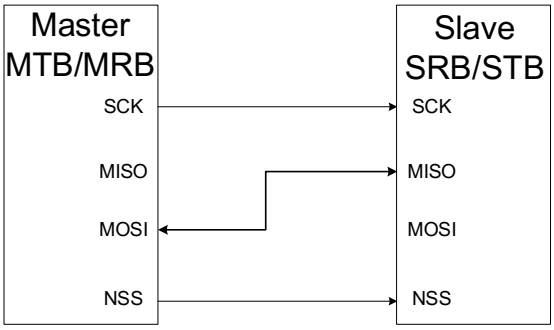



图 21-7. 典型的双向线连接




## SPI 初始化流程

在发送或接收数据之前，应用程序应遵循如下的 SPI 初始化流程：

1. 如果工作在主机模式或从机TI模式，配置SPI_CTL0中的PSC[2:0]位来生成预期波特率的SCK信号，或配置TI模式下的Td时间。否则，忽略此步骤。
2. 配置数据格式(SPI_CTL0中的FF16位)
3. 配置时钟时序(SPI_CTL0中的CKPL位和CKPH位)。
4. 配置帧格式(SPI_CTL0中的LF位)。
5. 按照上文***NSS功能***的描述，根据应用程序的需求，配置NSS模式(SPI_CTL0中的SWNSSEN位和SPI_CTL1中的NSSDRV位)。
6. 如果工作在TI模式，需要将SPI_CTL1中的TMOD位置1，否则，忽略此步骤。
7. 根据***表21-5. SPI运行模式***，配置MSTMOD位、RO位、BDEN位和BDOEN位。

8. 如果工作在SPI四线模式，需要将SPI_QCTL中的QMOD位置1，如果不是，则忽略此步骤。
9. 使能SPI(将SPIEN位置1)。

注意：在通信过程中，不应更改 CKPH、CKPL、MSTMOD、PSC[2:0]、LF 位。

### SPI 基本发送和接收流程

#### 发送流程

在完成初始化过程之后，SPI 模块使能并保持在空闲状态。在主机模式下，当软件写一个数据到发送缓冲区时，发送过程开始。在从机模式下，当 SCK 引脚上的 SCK 信号开始翻转，且 NSS 引脚电平为低，发送过程开始。所以，在从机模式下，应用程序必须确保在数据发送开始前，数据已经写入发送缓冲区中。

当 SPI 开始发送一个数据帧时，首先将这个数据帧从数据缓冲区加载到移位寄存器中，然后开始发送加载的数据。在数据帧的第一位发送之后，TBE(发送缓冲区空)位置 1。TBE 标志位置 1，说明发送缓冲区为空，此时如果需要发送更多数据，软件应该继续写 SPI_DATA 寄存器。

在主机模式下，若想要实现连续发送功能，那么在当前数据帧发送完成前，软件应该将下一个数据写入 SPI_DATA 寄存器中。

#### 接收流程

在最后一个采样时钟边沿之后，接收到的数据将从移位寄存器存入到接收缓冲区，且 RBNE(接收缓冲区非空)位置 1。软件通过读 SPI_DATA 寄存器获得接收的数据，此操作会自动清除 RBNE 标志位。在 MRU 和 MRB 模式中，为了接收下一个数据帧，硬件需要连续发送时钟信号，而在全双工主机模式(MFD)中，当发送缓冲区非空时，硬件才接收下一个数据帧。

### SPI 不同模式下的操作流程(非 SPI 四线模式或 TI 模式)

在全双工模式下，无论是 MFD 模式或者 SFD 模式，应用程序都应该监视 RBNE 标志位和 TBE 标志位，并且遵循上文描述的操作流程。

发送模式(MTU，MTB，STU 或 STB)与全双工模式中的发送流程类似，不同的是需要忽略 RBNE 位和 RXORERR 位。

相比于发送模式的情况，主机接收模式(MRU 或 MRB)与全双工的接收流程大不相同。在 MRU 模式或 MRB 模式下，在 SPI 使能后，SPI 产生连续的 SCK 信号，直到 SPI 停止。所以，软件应该忽略 TBE 标志位，并且在 RBNE 位置 1 后，读出接收缓冲区内的数据，否则，将会产生接收过载错误。

除了忽略 TBE 标志位，且只执行上述的接收流程之外，从机接收模式(SRU 或 SRB)与全双工模式类似。

### SPI TI 模式

SPI TI 模式将 NSS 作为一种特殊的帧头标志信号，它的操作流程与上文描述的常规模式类似。

上文描述的模式(MFD，MTU，MRU，MTB，MRB，SFD，STU，SRU，STB 和 SRB)都支持 TI 模式。但是，在 TI 模式中，SPI_CTL0 中的 CKPL 位和 CKPH 位是没有意义的，SCK 信号的采样边沿为下降沿。

**图 21-8. 主机 TI 模式在不连续发送时的时序图**

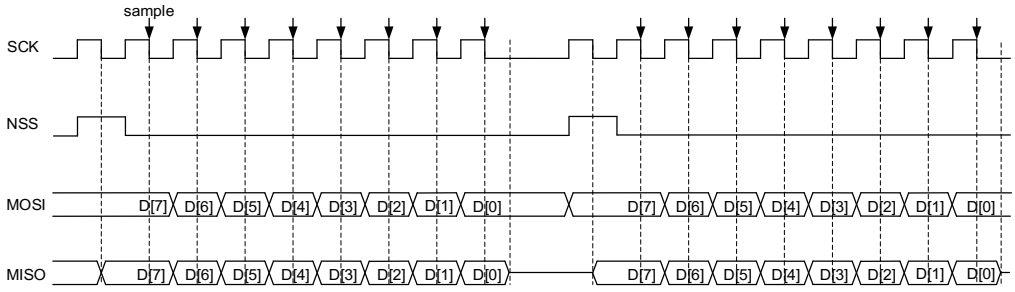
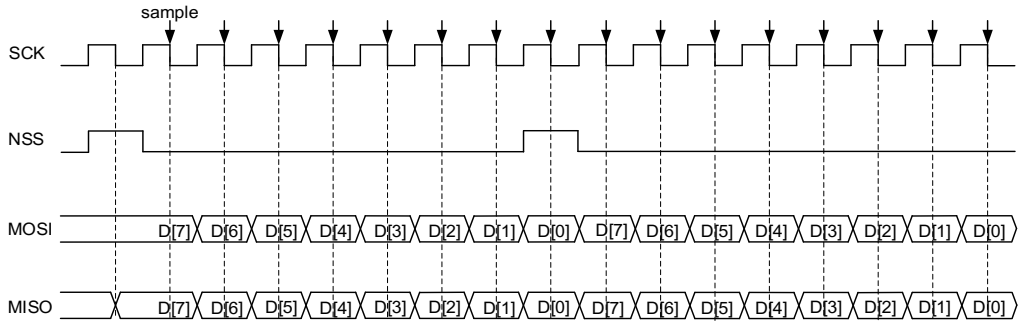
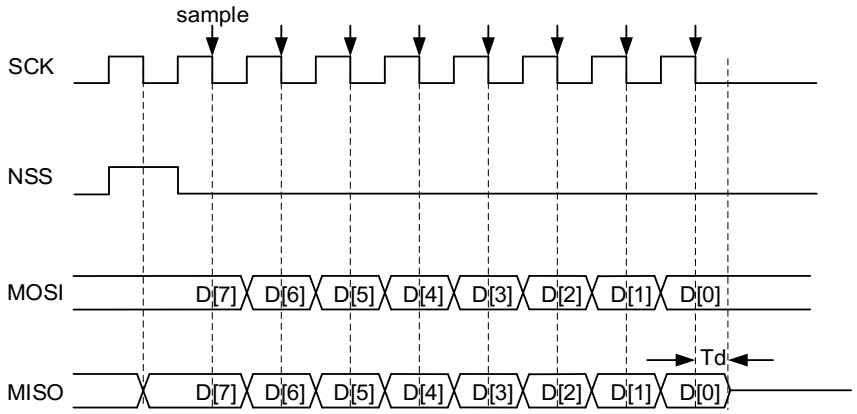



**图 21-9. 主机 TI 模式在连续发送时的时序图**




在主机 TI 模式下，SPI 模块可实现连续传输或者不连续传输。如果主机写 SPI_DATA 的速度很快，那么就是连续传输，否则，为不连续传输。在不连续传输中，在每个字节传输前需要一个额外的时钟周期。但是在连续传输中，额外的时钟周期只存在于第一个字节之前，随后字节的起始时钟周期被前一个字节的最后一位的时钟周期覆盖。

**图 21-10. 从机 TI 模式时序图**




在从机 TI 模式中，在 SCK 信号的最后一个上升沿，从机开始发送最后一个字节的 LSB 位，在半位的时间之后，主机开始采集数据。为了确保主机采集到正确的数据，在释放该引脚之前，从机需要在 SCK 信号的下降沿之后继续驱动该位一段时间，这段时间称为 $T_d$，$T_d$ 通过 SPI_CTL0 寄存器中的 PSC[2:0]位来设置。

$$T_d = \frac{T_{bit}}{2} + 5*T_{pclk} \quad (21\text{-}1)$$

例如，如果 PSC[2:0] = 010，那么$T_d$数值为 9*Tpclk。

在从机模式下，从机需要监视 NSS 信号，如果检测到错误的 NSS 信号，将会置位 FE 标志位。例如，NSS 信号在一个字节的中间位发生翻转。

### SPI 四线模式操作流程

SPI 四线模式用于控制四线 SPI Flash 外设。

要配置成 SPI 四线模式，首先要确认 TBE 位置 1，且 TRANS 位清零，然后将 SPI_QCTL 寄存器中的 QMOD 位置 1。在 SP 四线模式，SPI_CTL0 寄存器中 BDEN 位、BDOEN 位、CRCEN 位、CRCNT 位、FF16 位、RO 位和 LF 位保持清零，且 MSTMOD 位置 1，以保证 SPI 工作于主机模式。SPIEN 位、PSC 位、CKPL 位和 CKPH 位根据需要进行配置。

SPI 四线模式有两种运行模式：四线写模式和四线读模式，通过 SPI_QCTL 寄存器中的 QRD 位进行配置。

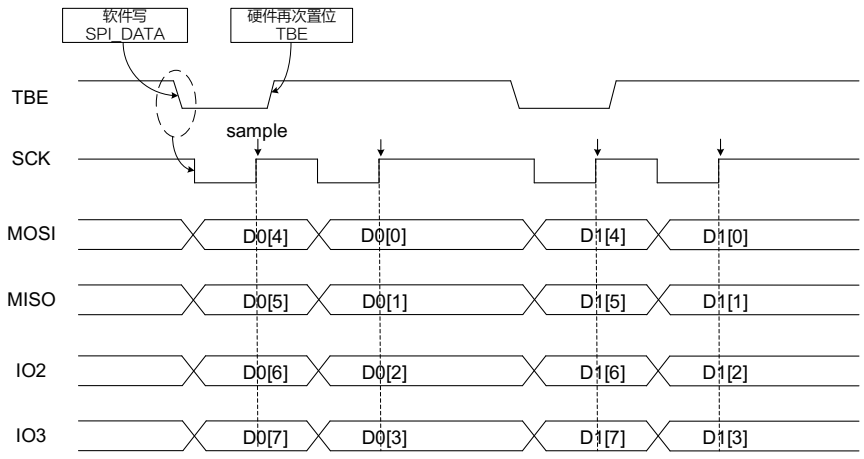
#### 四线写模式

当SPI_QCTL寄存器中的QMOD位置1且QRD位清零时，SPI工作在四线写模式。在四线写模式中，MOSI、MISO、IO2和IO3都用作输出引脚，在SCK产生时钟信号后，一旦数据写入SPI_DATA寄存器(TBE位清零)且SPIEN位置1时，将会通过这四个引脚发送写入的数据。SPI开始数据传输之后，每发送一个数据帧都要检测TBE标志位，若不能满足条件则停止传输。

四路模式下发送操作流程：

1. 根据应用需求，配置SPI_CTL0和SPI_CTL1中的时钟预分频、时钟极性、相位等参数；
2. 将SPI_QCTL中的QMOD位置1，然后将SPI_CTL0中的SPIEN位置1来使能SPI功能；
3. 向SPI_DATA寄存器中写入一个字节的数据，TBE标志位将会清零；
4. 等待硬件将TBE位重新置位，然后写入下一个字节数据。

**图 21-11. SPI 四线模式写操作时序图**



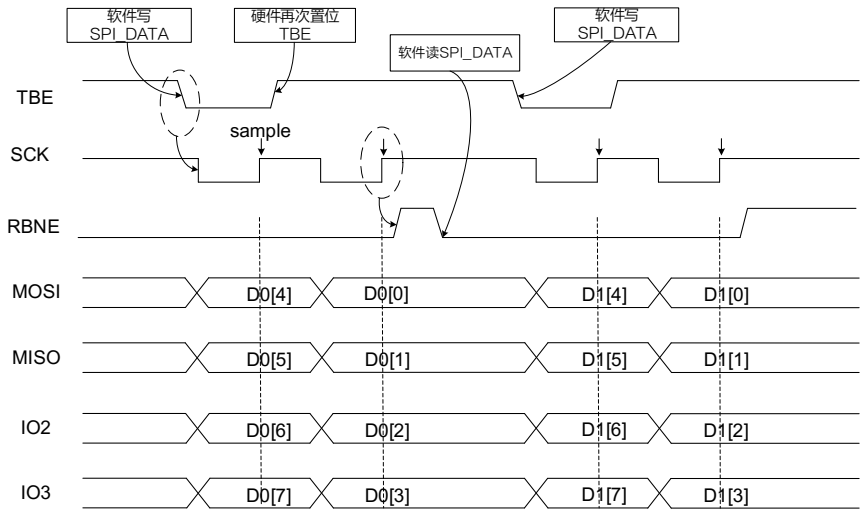
#### 四线读模式

当 SPI_QCTL 寄存器中的 QMOD 位和 QRD 位都置 1 时，SPI 工作在四路读模式。在四路读模式中，MOSI、MISO、IO2 和 IO3 都用作输入引脚。当数据写入 SPI_DATA 寄存器(此时 TBE 位被清零)且 SPIEN 位置 1 时，SPI 开始在 SCK 信号线上产生时钟信号。写数据到 SPI_DATA 寄存器只是为了产生 SCK 时钟信号，所以可以写入任何数据。SPI 开始数据传输之后，每发送一个数据帧都要检测 SPIEN 位和 TBE 位，若条件不满足则停止传输。所以软件需要一直向 SPI_DATA 写空闲数据，以产生 SCK 时钟信号。

四线模式下接收操作流程：

1. 根据应用需求，配置SPI_CTL0和SPI_CTL1中时钟预分频、时钟极性、相位等参数；
2. 将SPI_QCTL中的QMOD位和QRD位置1，然后将SPI_CTL0中的SPIEN位置1来使能SPI功能；
3. 写任意数据(例如0xFF)到SPI_DATA寄存器；
4. 等待RBNE位置1，然后读SPI_DATA寄存器来获取接收的数据；
5. 写任意数据(例如0xFF)到SPI_DATA寄存器，以接收下一个字节数据。

**图 21-12. SPI 四线模式读操作时序图**




### SPI 停止流程

不同运行模式下采用不同的流程来停止 SPI 功能：

#### MFD SFD

等待最后一个 RBNE 置位并接收最后一个数据，等待 TBE=1 和 TRANS=0。最后，通过清零 SPIEN 位关闭 SPI。

#### MTU MTB STU STB

将最后一个数据写入 SPI_DATA 寄存器，等待 TBE 位置 1。然后等待 TRANS 位清零，通过清零 SPIEN 位关闭 SPI。

**MRU MRB**

等待倒数第二个 RBNE 位置 1，从 SPI_DATA 寄存器读数据，等待一个 SCK 时钟周期，然后通过清零 SPIEN 位关闭 SPI。等待最后一个 RBNE 位置 1，并从 SPI_DATA 读数据。

**SRU SRB**

当应用程序不想接收数据时，可以禁用 SPI，然后等待 TRANS=0 以确保正在进行的传输完成。

**TI 模式**

TI 模式的停止流程与上面描述过程相同。

**SPI 四线模式**

在禁用 SPI 四路模式或者关闭 SPI 功能之前，软件应该先检查：TBE 位置 1，TRANS 位清零，SPI_QCTL 中的 QMOD 位和 SPI_CTL0 中的 SPIEN 位清零。

### 21.5.4. DMA 功能

DMA 功能在传输过程中将应用程序从数据读写过程中释放出来，从而提高了系统效率。

通过置位 SPI_CTL1 寄存器中的 DMATEN 位和 DMAREN 位，使能 SPI 模式的 DMA 功能。为了使用 DMA 功能，软件首先应当正确配置 DMA 模块，然后通过初始化流程配置 SPI 模块，最后使能 SPI。

SPI 使能后，如果 DMATEN 位置 1，每当 TBE=1 时，SPI 将会发出一个 DMA 请求，然后 DMA 应答该请求，并自动写数据到 SPI_DATA 寄存器。如果 DMAREN 位置 1，每当 RBNE=1 时，SPI 将会发出一个 DMA 请求，然后 DMA 应答该请求，并自动从 SPI_DATA 寄存器读取数据。

### 21.5.5. CRC 功能

SPI 模块包含两个 CRC 计算单元：分别用于发送数据和接收数据。CRC 计算单元使用 SPI_CRCPOLY 寄存器中定义的多项式。

通过配置 SPI_CTL0 中的 CRCEN 位使能 CRC 功能。对于数据线上每个发送和接收的数据，CRC 单元逐位计算 CRC 值，计算得到的 CRC 值可以从 SPI_TCRC 寄存器和 SPI_RCRC 寄存器中读取。

为了传输计算得到的 CRC 值，应用程序需要在最后一个数据写入发送缓冲区之后，设置 SPI_CTL0 中的 CRCNT 位。在全双工模式(MFD 或 SFD)，当 SPI 发送一个 CRC 值并且准备校验接收到的 CRC 值时，会将最新接收到的数据当作 CRC 值。在接收模式(MRB，MRU，SRU 和 SRB)下，在倒数第二个数据帧被接收后，软件应该把 CRCNT 位置 1。在 CRC 校验失败时，CRCERR 错误标志位将会置 1。

如果使能了 DMA 功能，软件不需要设置 CRCNT 位，硬件将会自动处理 CRC 传输和校验。

**注意：**当 SPI 处于从机模式且 CRC 功能使能时，无论 SPI 是否使能，CRC 计算器都对输入 SCK 时钟敏感。只有当时钟稳定时，软件才能启用 CRC，以避免错误的 CRC 计算。当 SPI 作

为从机工作时，在数据阶段和 CRC 阶段之间，内部 NSS 信号需要保持低电平。

当配置 SPI 为从模式并且使用 CRC 功能时，即使 NSS 引脚为高时仍然会执行 CRC 的计算(当 NSS 信号为高时，只要 SCK 引脚上有时钟脉冲，则 CRC 计算会继续执行)。当主设备交替地与多个从设备进行通信时，将会出现这种情况,此时建议在 NSS 信号为低时重启 CRC 功能)。当从设备未选中(NSS 信号为高)转换到被选中为一个新的从设备(NSS 信号为低)的时候，为了保持主从设备端下次 CRC 计算结果的同步，应该清除主从两端的 CRC 数值。建议按照下述步骤清除 CRC 数值:

1. 关闭 SPI 模块(SPIEN=0);
2. 清除 CRCEN 位(CRCEN=0);
3. 设置 CRCEN 位(CRCEN=1);
4. 使能 SPI 模块(SPIEN=1)。

## 21.6. SPI 中断

### 21.6.1. 状态标志位

■ **发送缓冲区空标志位(TBE)**

当发送缓冲区为空时，TBE 置位。软件可以通过写 SPI_DATA 寄存器将下一个待发送数据写入发送缓冲区。

■ **接收缓冲区非空标志位(RBNE)**

当接收缓冲区非空时，RBNE 置位，表示此时接收到一个数据，并已存入到接收缓冲区中，软件可以通过读 SPI_DATA 寄存器来读取此数据。

■ **SPI通信进行中标志位(TRANS)**

TRANS 位是用来指示当前传输是否正在进行或结束的状态标志位，它由内部硬件置位和清除，无法通过软件控制。该标志位不会产生任何中断。

### 21.6.2. 错误标志

■ **配置错误标志(CONFERR)**

在主机模式中，CONFERR 位是一个错误标志位。在硬件 NSS 模式中，如果 NSSDRV 没有使能，当 NSS 被拉低时，CONFERR 位被置 1。在软件 NSS 模式中，当 SWNSS 位为 0 时，CONFERR 位置 1。当 CONFERR 位置 1 时，SPIEN 位和 MSTMOD 位由硬件清除，SPI 关闭，设备强制进入从机模式。

在 CONFERR 位清零之前，SPIEN 位和 MSTMOD 位保持写保护，从机的 CONFERR 位不能置 1。在多主机配置中，设备可以在 CONFERR 位置 1 时进入从机模式，这意味着发生了系统控制的多主冲突。

■ **接收过载错误(RXORERR)**

在 RBNE 位为 1 时，如果再有数据被接收，RXORERR 位将会置 1。这说明，上一帧数据还未被读出而新的数据已经接收了。接收缓冲区的内容不会被新接收的数据覆盖，所以新接收的数据丢失。

■ **帧错误(FERR)**

在 TI 从机模式下，从机也要监视 NSS 信号，如果检测到错误的 NSS 信号，将会置位 FERR 标志位。例如，NSS 信号在一个字节的中间位发生翻转。

■ **CRC错误(CRCERR)**

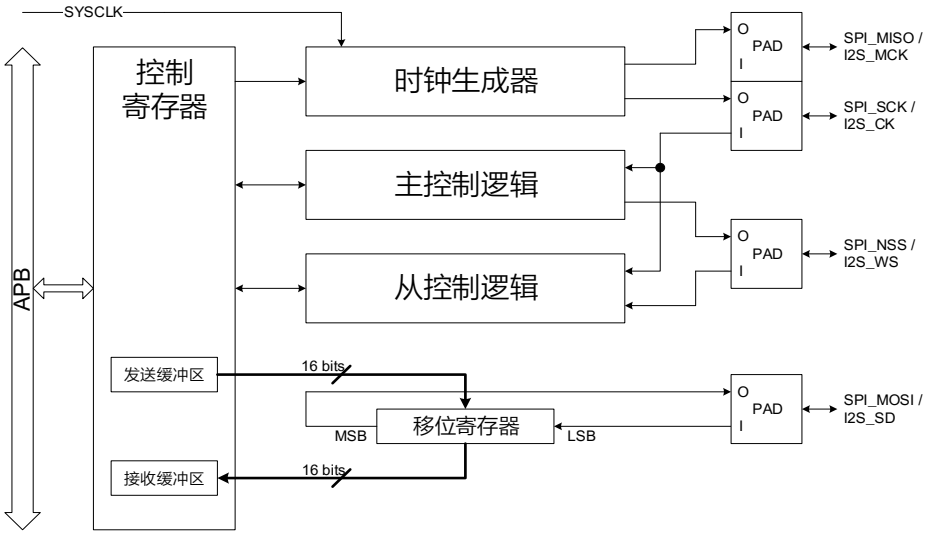
当 CRCEN 位置 1 时，SPI_RCRC 寄存器中接收到的 CRC 值将会和紧随着最后一帧数据接收到的 CRC 值进行比较。当两者不同时，CRCERR 位将会置 1。

**表 21-6. SPI 中断请求**

| 中断事件 | 描述 | 清除方式 | 中断使能位 |
|---|---|---|---|
| TBE | 发送缓冲区空 | 写SPI_DATA寄存器 | TBEIE |
| RBNE | 接收缓冲区非空 | 读SPI_DATA寄存器 | RBNEIE |
| CONFERR | 配置错误 | 读或写 SPI_STAT 寄存器，然后写 SPI_CTL0 寄存器 | ERRIE |
| RXORERR | 接收过载错误 | 读SPI_DATA寄存器，然后读 SPI_STAT寄存器 | |
| CRCERR | CRC错误 | 写0到CRCERR位 | |
| FERR | TI模式帧错误 | 写0到FERR位 | |

## 21.7. I2S 结构框图

**图 21-13. I2S 结构框图**




I2S 功能有 5 个子模块，分别是控制寄存器、时钟生成器、主机控制逻辑、从机控制逻辑和移位寄存器。所有的用户可配置寄存器都在控制寄存器模块实现，其中包括发送缓冲区和接收缓

冲区。时钟生成器用来在主机模式下生成 I2S 通信时钟。主机控制逻辑用来在主机模式下生成 I2S_WS 信号并控制通信。从机控制逻辑根据接收到的 I2SCK 和 I2S_WS 信号来控制从机模式的通信。移位寄存器控制 I2S_SD 上的串行数据发送和接收。

## 21.8. I2S 信号线描述

I2S 接口有 4 个引脚，分别是 I2S_CK、I2S_WS、I2S_SD 和 I2S_MCK。I2S_CK 是串行时钟信号，与 SPI_SCK 共享引脚。I2S_WS 是数据帧控制信号，与 SPI_NSS 共享引脚。I2S_SD 是串行数据信号，与 SPI_MOSI 共享引脚。I2S_MCK 是主时钟信号，在 SPI0，SPI3 和 SPI4 中，与 SPI_MISO 共享引脚，而在 SPI1 和 SPI2 中，I2S_MCK 有一个专用引脚。I2S_MCK 对于 I2S 接口而言是个可选信号，它提供了一个 256 倍于 Fs 的时钟频率，其中 Fs 是音频采样率。

## 21.9. I2S 功能描述

### 21.9.1. I2S 音频标准

I2S 音频标准是通过设置 SPI_I2SCTL 寄存器中的 I2SSTD[1:0]位域来选择的，可以选择四种音频标准：I2S 飞利浦标准，MSB 对齐标准，LSB 对齐标准和 PCM 标准。除 PCM 之外的所有标准都是两个通道(左通道和右通道)的音频数据分时复用 I2S 接口的，并通过 I2S_WS 信号来区分当前数据属于哪个通道。对于 PCM 标准，I2S_WS 信号表示帧同步信息。

数据长度和通道长度可以通过 SPI_I2SCTL 寄存器中的 DTLEN 位和 CHLEN 位来设置。由于通道长度必须大于或等于数据长度，所以有四种数据包类型可供选择。它们分别是：16 位数据打包成 16 位数据帧格式，16 位数据打包成 32 位数据帧格式，24 位数据打包成 32 位数据帧格式，32 位数据打包成 32 位数据帧格式。用于发送和接收的数据缓冲区都是 16 位宽度。所以，要完成数据长度为 24 位或 32 位的数据帧传输，SPI_DATA 寄存器需要被访问 2 次；而要完成数据长度为 16 位的数据帧传输，SPI_DATA 寄存器只需被访问 1 次。如需将 16 位数据打包成 32 位数据帧，硬件会自动插入 16 位 0 将 16 位数据扩展为 32 位格式。

对于所有标准和数据包类型来说，数据的最高有效位总是最先被发送的。对于所有基于两通道分时复用的标准来说，总是先发送左通道，然后是右通道。

#### I2S 飞利浦标准

对于 I2S 飞利浦标准，I2S_WS 和 I2S_SD 在 I2S_CK 的下降沿变化，I2S_WS 在数据的前一个时钟开始有效。各种配置情况的时序图如下所示。

**图 21-14. I2S 飞利浦标准时序图(DTLEN=00, CHLEN=0, CKPL=0)**

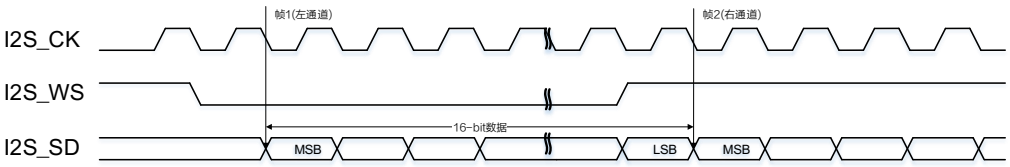
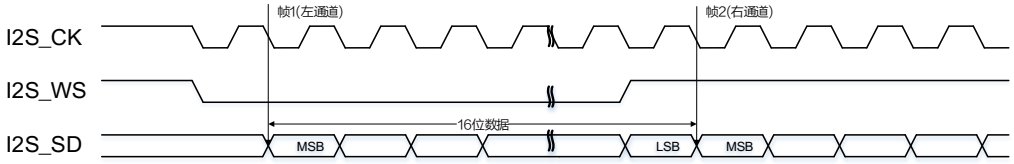


**图 21-15. I2S 飞利浦标准时序图(DTLEN=00, CHLEN=0, CKPL=1)**




当 16 位数据打包成 16 位数据帧时，每完成一帧数据的传输只需要访问 SPI_DATA 寄存器一次。

**图 21-16. I2S 飞利浦标准时序图(DTLEN=10, CHLEN=1, CKPL=0)**

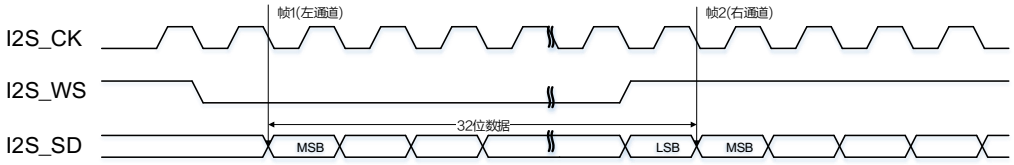
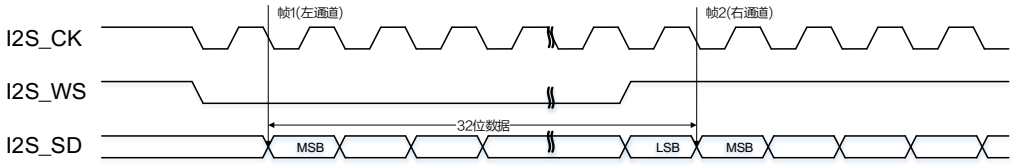



**图 21-17. I2S 飞利浦标准时序图(DTLEN=10, CHLEN=1, CKPL=1)**




当 32 位数据打包成 32 位数据帧的帧格式时，每完成 1 帧数据的传输需要访问 SPI_DATA 寄存器 2 次。在发送模式下，如果要发送一个 32 位数据，第一个写入 SPI_DATA 寄存器的数据应该是高 16 位数据，第二个数据应该是低 16 位数据。在接收模式下，如果要接收一个 32 位数据，第一个从 SPI_DATA 寄存器读到的数据应该是高 16 位数据，第二个数据应该是低 16 位数据。

**图 21-18. I2S 飞利浦标准时序图(DTLEN=01, CHLEN=1, CKPL=0)**

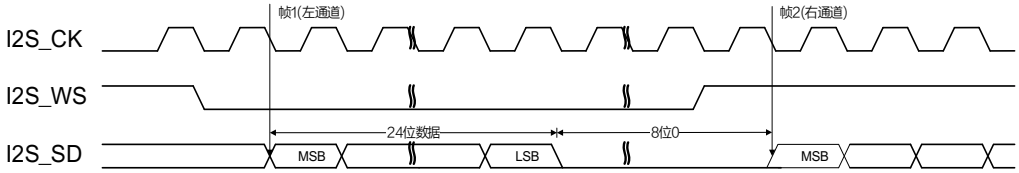
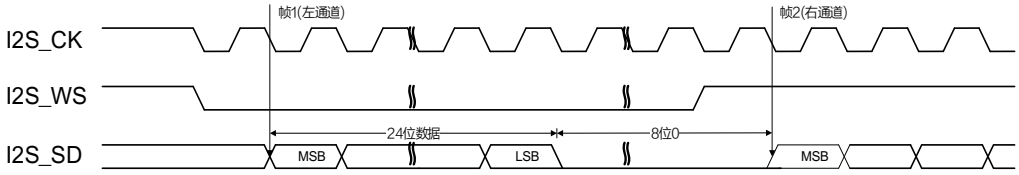



**图 21-19. I2S 飞利浦标准时序图(DTLEN=01, CHLEN=1, CKPL=1)**




当 24 位数据打包成 32 位数据帧的帧格式时，每完成 1 帧数据的传输需要访问 SPI_DATA 寄存器 2 次。在发送模式下，如果要发送一个 24 位数据 D[23:0]，第一个写入 SPI_DATA 寄存器的数据应该是高 16 位数据 D[23:8]，第二个数据应该是一个 16 位数据，该 16 位数据的高 8 位是 D[7:0]，低 8 位数据可以是任意值。在接收模式下，如果要接收一个 24 位数据 D[23:0]，

第一个从 SPI_DATA 寄存器读到的数据应该是高 16 位数据 D[23:8]，第二个数据应该是一个 16 位数据，该 16 位数据的高 8 位是 D[7:0]，低 8 位数据全是 0。

**图 21-20. I2S 飞利浦标准时序图(DTLEN=00, CHLEN=1, CKPL=0)**

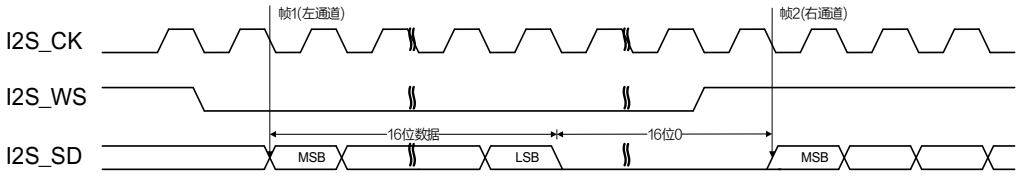
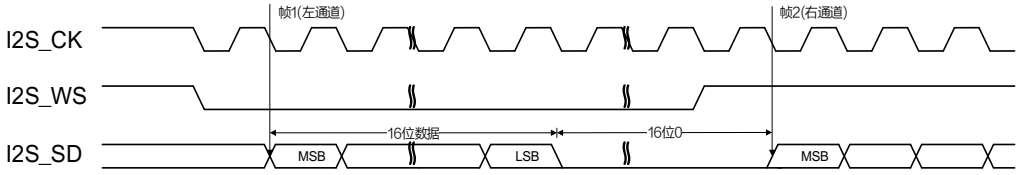



**图 21-21. I2S 飞利浦标准时序图(DTLEN=00, CHLEN=1, CKPL=1)**




当 16 位数据打包成 32 位数据帧时，每完成一帧数据的传输只需要访问 SPI_DATA 寄存器一次。为了将该 16 位数据扩展成 32 位数据，剩下的 16 位被硬件强制填充为 0x0000。

### MSB 对齐标准

对于MSB对齐标准，I2S_WS和I2S_SD在I2S_CK的下降沿变化。SPI_DATA寄存器的处理方式与I2S飞利浦标准完全相同。各个配置情况的时序图如下所示。

**图 21-22. MSB 对齐标准时序图(DTLEN=00, CHLEN=0, CKPL=0)**

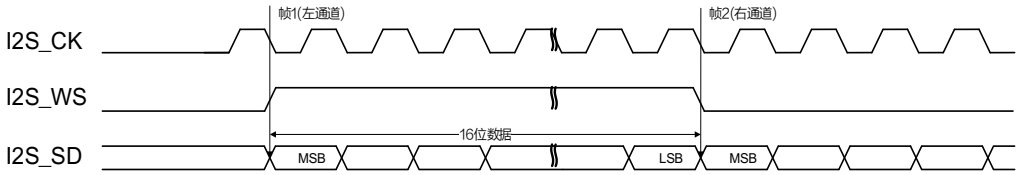



**图 21-23. MSB 对齐标准时序图(DTLEN=00, CHLEN=0, CKPL=1)**

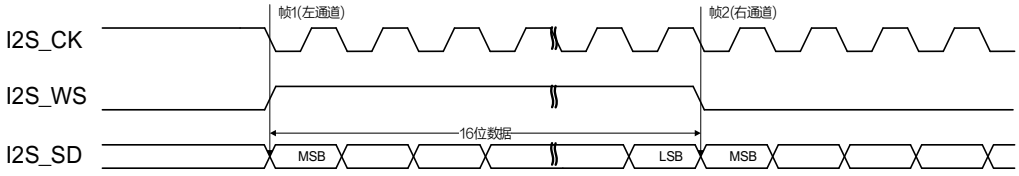



**图 21-24. MSB 对齐标准时序图(DTLEN=10, CHLEN=1, CKPL=0)**

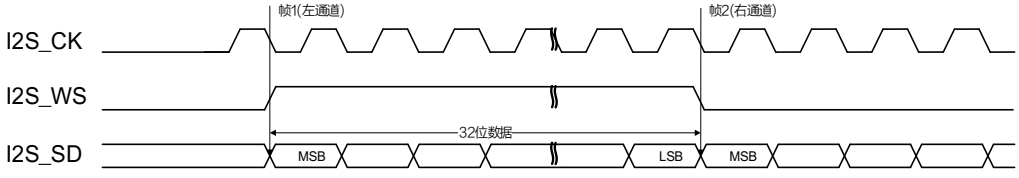


图 21-25. MSB 对齐标准时序图(DTLEN=10, CHLEN=1, CKPL=1)

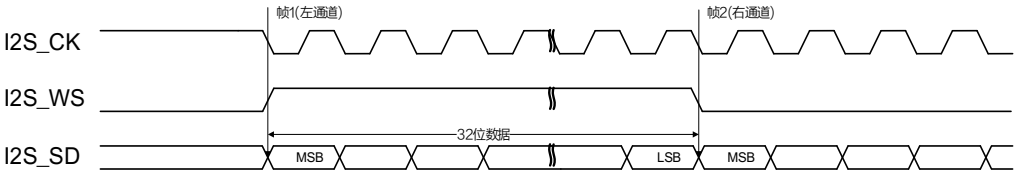



图 21-26. MSB 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=0)

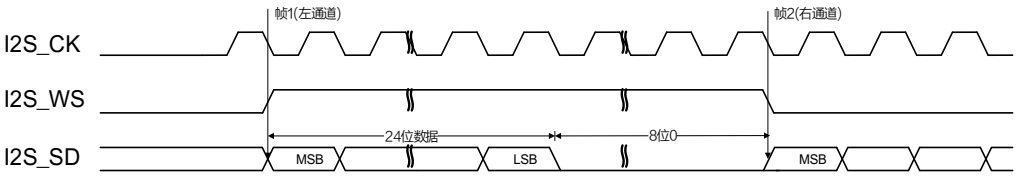



图 21-27. MSB 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=1)

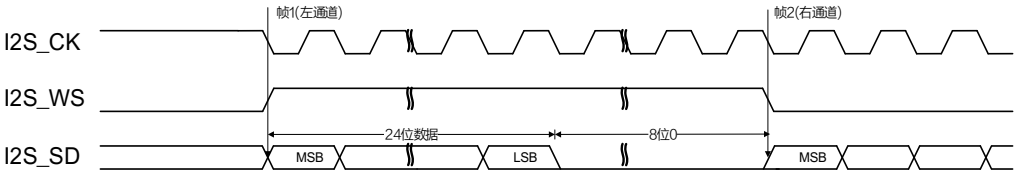



图 21-28. MSB 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=0)

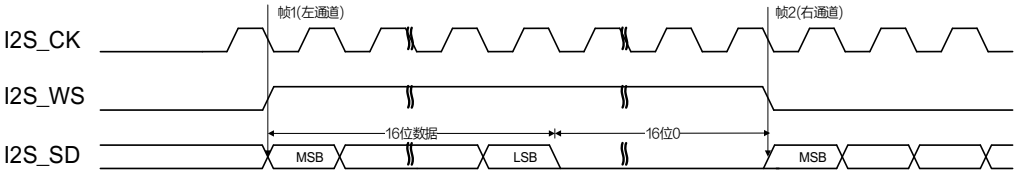
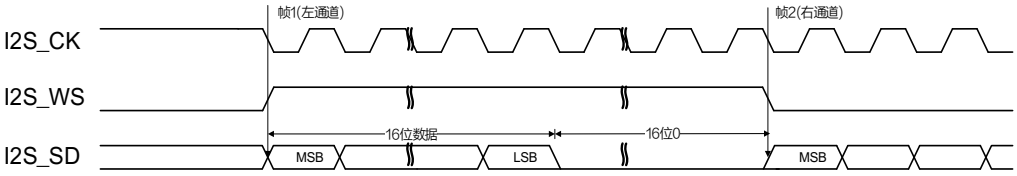



图 21-29. MSB 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=1)




## LSB 对齐标准

对于 LSB 对齐标准，I2S_WS 和 I2S_SD 在 I2S_CK 的下降沿变化。在通道长度与数据长度相同的情况下，LSB 对齐标准和 MSB 对齐标准是完全相同的。对于通道长度大于数据长度的情况，LSB 对齐标准的有效数据与最低位对齐，而 MSB 对齐标准的有效数据与最高位对齐。通道长度大于数据长度的各种配置情况时序图如下所示。

**图 21-30. LSB 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=0)**

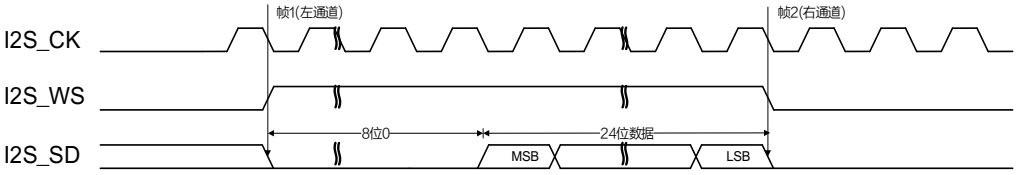
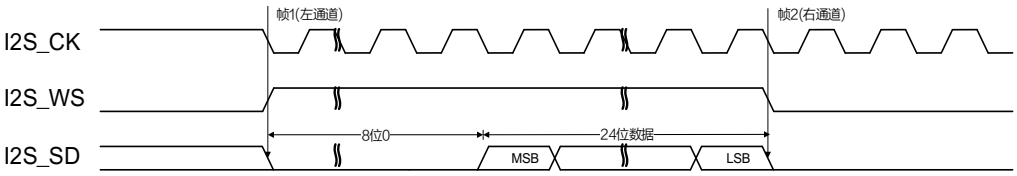



**图 21-31. LSB 对齐标准时序图(DTLEN=01, CHLEN=1, CKPL=1)**




当 24 位数据打包成 32 位数据帧的帧格式时，每完成 1 帧数据的传输需要访问 SPI_DATA 寄存器 2 次。在发送模式下，如果要发送一个 24 位数据 D[23:0]，第一个写入 SPI_DATA 寄存器的数据应该是一个 16 位数据，该 16 位数据的高 8 位可以是任意值，低 8 位是 D[23:16]，第二个数据应该是低 16 位数据 D[15:0]。在接收模式下，如果要接收一个 24 位数据 D[23:0]，第一个从 SPI_DATA 寄存器读到的数据应该是一个 16 位数据，该 16 位数据的高 8 位是 0，低 8 位是 D[23:16]，第二个数据应该是低 16 位数据 D[15:0]。

**图 21-32. LSB 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=0)**

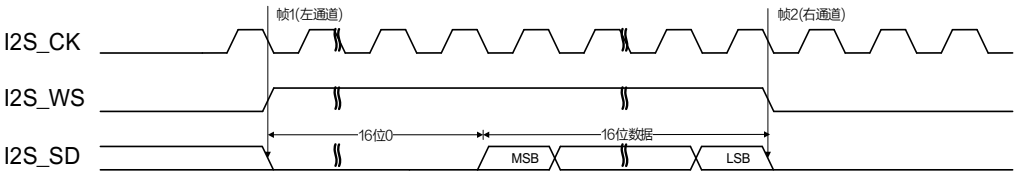
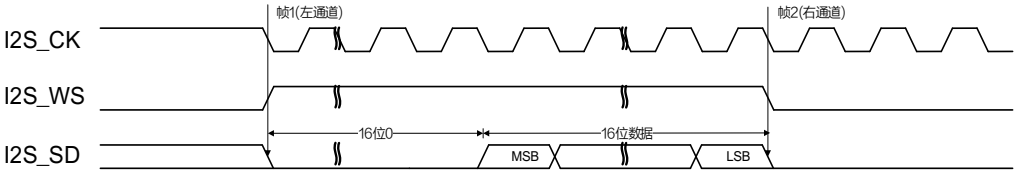



**图 21-33. LSB 对齐标准时序图(DTLEN=00, CHLEN=1, CKPL=1)**




当 16 位数据打包成 32 位数据帧时，每完成一帧数据的传输只需要访问 SPI_DATA 寄存器一次。为了将该 16 位数据扩展成 32 位数据，剩下的 16 位被硬件强制填充为 0x0000。

### PCM 标准

对于 PCM 标准，I2S_WS 和 I2S_SD 在 I2S_CK 的上升沿变化，I2S_WS 信号表示帧同步信息。可以通过 SPI_I2SCTL 寄存器的 PCMSMOD 位来选择短帧同步模式和长帧同步模式。SPI_DATA 寄存器的处理方式与 I2S 飞利浦标准完全相同。短帧同步模式的各种配置情况时序图如下所示。

图 21-34. PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=0)

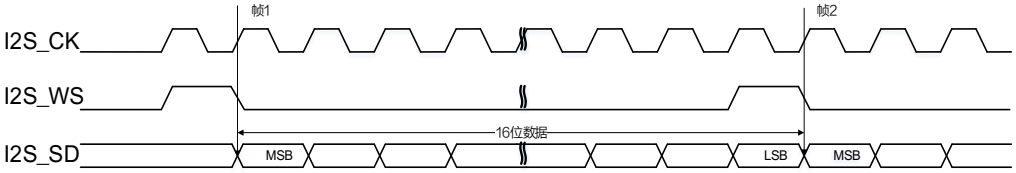



图 21-35. PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=1)

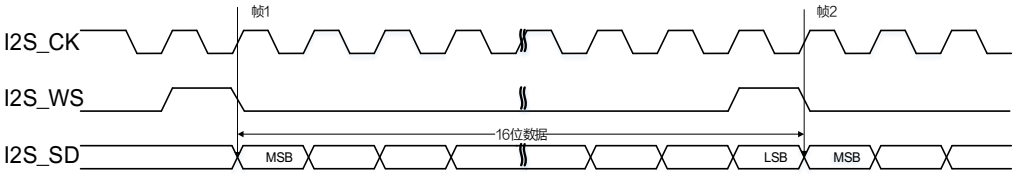



图 21-36. PCM 标准短帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=0)

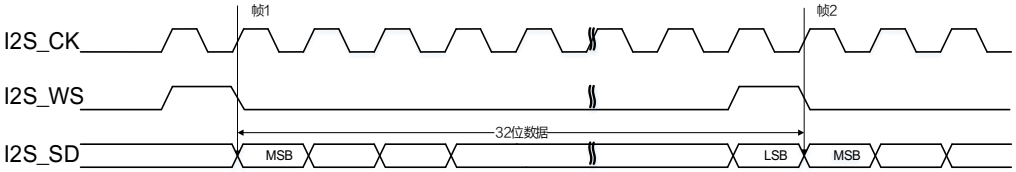



图 21-37. PCM 标准短帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=1)

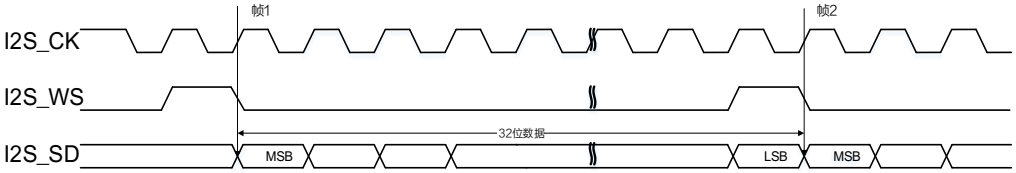



图 21-38. PCM 标准短帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=0)

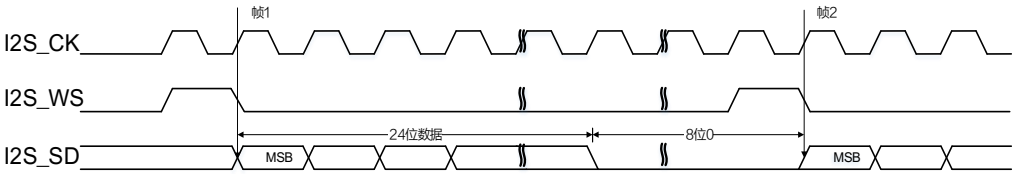



图 21-39. PCM 标准短帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=1)

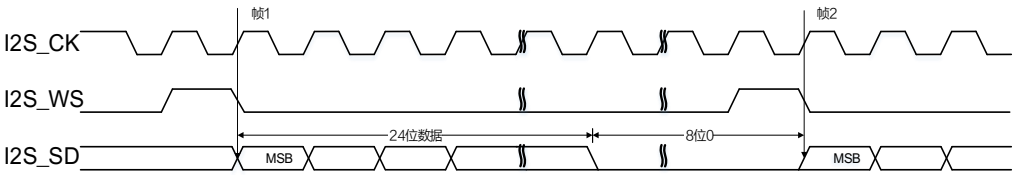


**图 21-40. PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=0)**

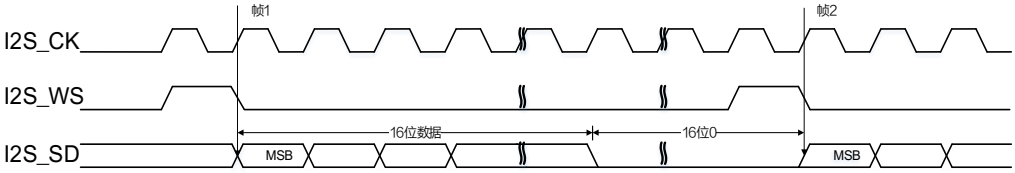
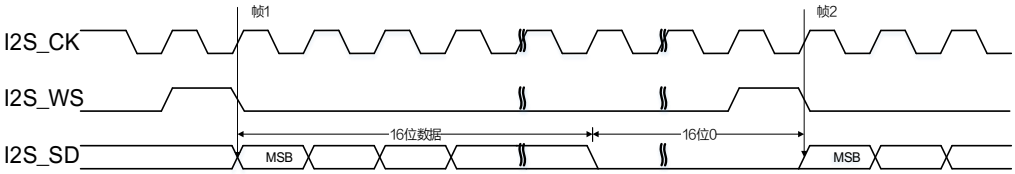



**图 21-41. PCM 标准短帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=1)**




长帧同步模式的各种配置情况时序图如下所示。

**图 21-42. PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=0)**

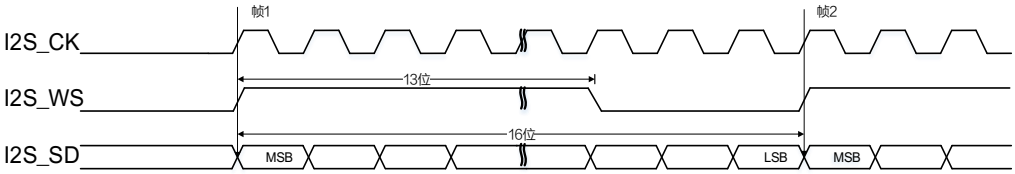



**图 21-43. PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=0, CKPL=1)**

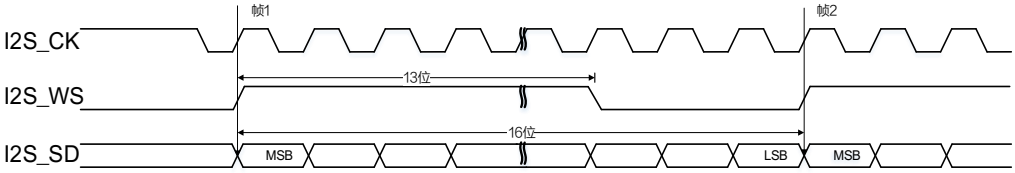



**图 21-44. PCM 标准长帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=0)**

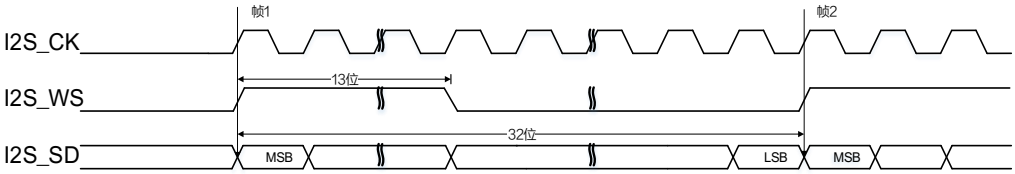



**图 21-45. PCM 标准长帧同步模式时序图(DTLEN=10, CHLEN=1, CKPL=1)**

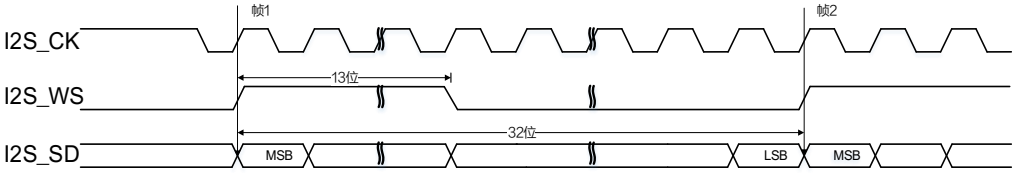


图 21-46. PCM 标准长帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=0)

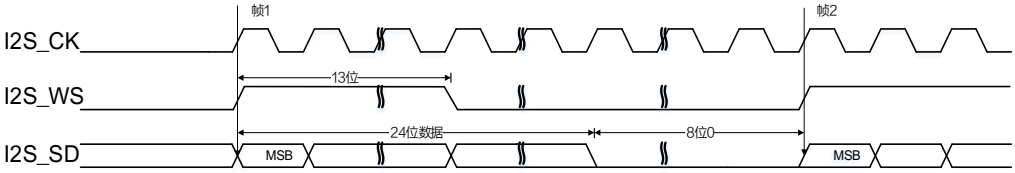



图 21-47. PCM 标准长帧同步模式时序图(DTLEN=01, CHLEN=1, CKPL=1)

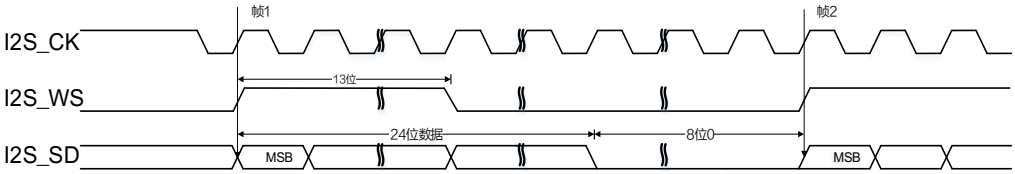



图 21-48. PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=0)

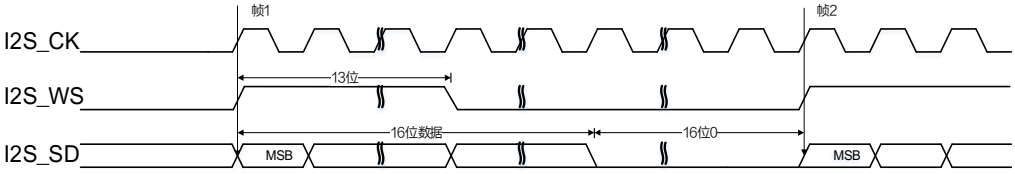
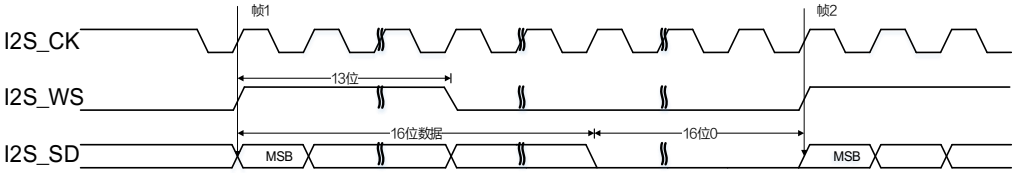
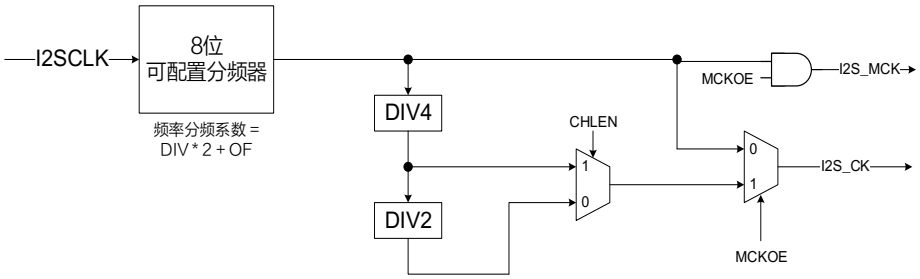



图 21-49. PCM 标准长帧同步模式时序图(DTLEN=00, CHLEN=1, CKPL=1)




## 21.9.2. I2S 时钟

图 21-50. I2S 时钟生成结构框图




I2S 时钟生成器如*图 21-50. I2S 时钟生成结构框图*所示。I2S 接口时钟是通过 SPI_I2SPSC 寄存器的 DIV 位，OF 位和 MCKOEN 位以及 SPI_I2SCTL 寄存器的 CHLEN 位来配置的。I2S 比特率可以通过*表 21-7. I2S 比特率计算公式*所示的公式计算。

**表 21-7. I2S 比特率计算公式**

| MCKOEN | CHLEN | 公式 |
|---|---|---|
| 0 | 0 | I2SCLK / (DIV * 2 + OF) |
| 0 | 1 | I2SCLK / (DIV * 2 + OF) |
| 1 | 0 | I2SCLK / (8 * (DIV * 2 + OF)) |
| 1 | 1 | I2SCLK / (4 * (DIV * 2 + OF)) |

音频采样率(Fs)和 I2S 比特率的关系由如下公式定义：

Fs = I2S 比特率 / (通道长度 * 通道数)

所以，为了得到期望的音频采样率，时钟生成器需要按*表 21-8. 音频采样频率计算公式*所列的公式进行配置。

**表 21-8. 音频采样频率计算公式**

| MCKOEN | CHLEN | 公式 |
|---|---|---|
| 0 | 0 | I2SCLK / (32 * (DIV * 2 + OF)) |
| 0 | 1 | I2SCLK / (64 * (DIV * 2 + OF)) |
| 1 | 0 | I2SCLK / (256 * (DIV * 2 + OF)) |
| 1 | 1 | I2SCLK / (256 * (DIV * 2 + OF)) |

I2S 时钟源可以由 PLLI2S 或外部 I2S_CKIN 引脚提供，在 RCU 模块进行配置。软件需要仔细计算 I2S 分频因子和 PLLI2S 以得到精确的音频采样频率。如果 PLLI2S 精度不能满足应用程序的要求，可以通过 I2S_CKIN 引脚提供精确的外部 I2S 时钟。

**注意：** I2S 串行时钟的配置值需设置为低于 PCLK 时钟的 1/6 倍以下(不包含 1/6)。

## 21.9.3. 运行

### 运行模式

运行模式是通过 SPI_I2SCTL 寄存器的 I2SOPMOD 位来选择的。共有四种运行模式可供选择：主机发送模式，主机接收模式，从机发送模式和从机接收模式。各种运行模式下 I2S 接口信号的方向如*表 21-9. 各种运行模式下 I2S 接口信号的方向*所示。

**表 21-9. 各种运行模式下 I2S 接口信号的方向**

| 运行模式 | I2S_MCK | I2S_CK | I2S_WS | I2S_SD | I2S_ADD_SD(2) |
|---|---|---|---|---|---|
| 主机发送 | 输出或 NU(1) | 输出 | 输出 | 输出 | NU(1) |
| 主机接收 | 输出或 NU(1) | 输出 | 输出 | 输入 | NU(1) |
| 从机发送 | 输入或 NU(1) | 输入 | 输入 | 输出 | NU(1) |
| 从机接收 | 输入或 NU(1) | 输入 | 输入 | 输入 | NU(1) |
| 全双工 | 输出或 NU(1) | 输出 | 输出 | 输出或输入 | 输入或输出 |

1. NU表示该引脚没有被I2S使用，可以用于其他功能。
2. I2S1和I2S2为了支持全双工运行模式，需要两个额外的片上I2S模块：I2S_ADD1和I2S_ADD2。I2S_ADD_SD引脚是I2S_ADD模块的数据引脚，在后面的章节将详细介绍全双工模式。

### I2S 初始化流程

I2S 初始化过程包括以下五个步骤。如果要初始化 I2S 工作在主机模式，五个步骤都要执行。如果要初始化 I2S 工作在从机模式，只需要执行步骤 2、3、4、5。

- 步骤1：配置SPI_I2SPSC寄存器的DIV[7:0]位，OF位和MCKOEN位，定义I2S的比特率和选择是否需要提供I2S_MCK信号。
- 步骤2：配置SPI_I2SCTL寄存器的CKPL位，定义空闲状态的时钟极性。
- 步骤3：配置SPI_I2SCTL寄存器的I2SSEL位，I2SSTD[1:0]位，PCMSMOD位，I2SOPMOD[1:0]位，DTLEN[1:0]位和CHLEN位，定义I2S的特性。
- 步骤4：配置SPI_CTL1寄存器的TBEIE位，RBNEIE位，ERRIE位，DMATEN位和DMAREN位，选择中断源和DMA功能。此步骤可选。
- 步骤5：将SPI_I2SCTL寄存器的I2SEN位置1，来启动I2S。

### I2S 主机发送流程

TBE 标志位被用来控制发送流程。如前文所述，TBE 标志位表示发送缓冲区空，此时，如果 SPI_CTL1 寄存器的 TBEIE 位为 1，将产生中断。首先，发送缓冲区为空(TBE 为 1)，且移位寄存器中没有发送序列。当 16 位数据被写入 SPI_DATA 寄存器时(TBE 变为 0)，数据立即从发送缓冲区装载到移位寄存器中(TBE 变为 1)。此时，发送序列开始。

数据是并行地装载到 16 位移位寄存器中的，然后串行地从 I2S_SD 引脚发出(高位先发)。下一个数据应该在 TBE 为 1 时写入 SPI_DATA 寄存器。数据写入 SPI_DATA 寄存器之后，TBE 变为 0。当前发送序列结束时，发送缓冲区的数据会自动装载到移位寄存器中，然后 TBE 标志变回 1。为了保证连续的音频数据发送，下一个将要发送的数据必须在当前发送序列结束之前写入 SPI_DATA 寄存器。

对于除 PCM 标准外的所有标准，I2SCH 标志用来区别当前传输数据所属的通道。I2SCH 标志在每次 TBE 标志由 0 变 1 的时候更新。刚开始 I2SCH 标志为 0，表示左通道的数据应该被写入 SPI_DATA 寄存器。

为了关闭 I2S，I2SEN 位必须在 TBE 标志为 1 且 TRANS 标志为 0 之后清零。

### I2S 主机接收流程

RBNE 标志被用来控制接收序列。如前文所述，RBNE 标志表示接收缓冲区非空，如果 SPI_CTL1 寄存器的 RBNEIE 位为 1，将产生中断。当 SPI_I2SCTL 寄存器的 I2SEN 位被置 1 时，接收流程立即开始。首先，接收缓冲区为空(RBNE 为 0)。当一个接收流程结束时，接收到的数据将从移位寄存器装载到接收缓冲区(RBNE 变为 1)。当 RBNE 为 1 时，用户应该将数据从 SPI_DATA 寄存器中读走。读操作完成后，RBNE 变为 0。必须在下一次接收结束之前读

走 SPI_DATA 寄存器中的数据，否则将发生接收过载错误。发送接收过载错误时， RXORERR 标志位会被置 1，如果 SPI_CTL1 寄存器的 ERRIE 位为 1，将会产生中断。这种情况下，必须先关闭 I2S 再打开 I2S，然后再恢复通讯。

对于除 PCM 之外的所有标准来说，I2SCH 标志用来区分当前传输数据所属的通道。I2SCH 标志在每次 RBNE 标志由 0 变 1 时更新。

为了关闭 I2S，不同的音频标准，数据长度和通道长度采用不同的操作步骤。每种情况的操作步骤如下所示。

■ 数据长度为16位，通道长度为32位，LSB对齐标准(DTLEN = 00，CHLEN = 1，且I2SSTD = 0b10)：

1. 等待倒数第二个RBNE；
2. 等待17个I2S时钟周期(I2S_CK引脚上的时钟)；
3. 清除I2SEN位。

■ 数据长度为16位，通道长度为32位，除LSB对齐标准之外的其他标准(DTLEN = 00，CHLEN = 1，且 I2SSTD不等于0b10)：

1. 等待最后一个RBNE；
2. 等待1个I2S时钟周期；
3. 清除I2SEN位。

■ 其他所有情况：

1. 等待倒数第二个RBNE；
2. 等待1个I2S时钟周期；
3. 清除I2SEN位。

### I2S 从机发送流程

从机发送流程和主机发送流程相似，不同之处如下：

在从机模式下，从机需要在外部主机开始通讯之前使能。当外部主机开始发送时钟信号且 I2S_WS 信号请求传输数据时，发送流程开始。数据需要在外部主机发起通讯之前写入 SPI_DATA 寄存器。为了确保音频数据的连续传输，必须在当前发送序列结束之前将下一个待发送的数据写入 SPI_DATA 寄存器，否则会产生发送欠载错误。此时 TXURERR 标志会置 1，如果 SPI_CTL1 寄存器的 ERRIE 位为 1，将会产生中断。这种情况下，必须先关闭 I2S 再打开 I2S 来恢复通讯。从机模式下，I2SCH 标志是根据外部主机发送的 I2S_WS 信号而变化的。

为了关闭 I2S，必须在 TBE 标志变为 1 且 TRANS 标志变为 0 之后，才能清除 I2SEN 位。

### I2S 从机接收流程

从机接收流程与主机接收流程类似。不同之处如下。

在从机模式下，从机需要在外部主机开始通讯之前使能。当外部主机开始发送时钟信号且 I2S_WS 信号指示数据开始时，接收流程开始。从机模式下，I2SCH 标志是根据外部主机发送的 I2S_WS 信号而变化的。

为了关闭 I2S，必须在收到最后一个 RBNE 之后立即清除 I2SEN 位。

#### I2S 全双工模式

单个的 I2S 模块只支持单向传输：发送模式或接收模式，通过一个附加的 I2S 模块(I2S_ADD 模块)可以实现 I2S 的全双工模式。I2S_ADD 模块与 I2S 模块功能一样，但只工作在从模式。一共有两个 I2S_ADD 模块：I2S_ADD1 和 I2S_ADD2，所以只有 I2S1 和 I2S2 支持全双工模式。I2S_ADD 模块的 I2S_CK 和 I2S_WS 引脚分别与对应的 I2S 模块的相应引脚内部连接，I2S_ADD 模块的 I2S_SD 引脚映射到对应的 I2S 模块的 SPI_MISO 引脚。

为了工作在全双工模式，需要使能 I2S 模块和相应的 I2S_ADD 模块，I2S 模块支持两种全双工模式：主机模式和从机模式。

在主机全双工模式下，软件必须设置 I2S 为主机，I2S_ADD 为从机，I2S_ADD 模块的 WS 和 SCK 信号都由主机 I2S 模块提供。

在从机全双工模式下，软件必须设置 I2S 和 I2S_ADD 都为从机，I2S 模块和 I2S_ADD 模块的 WS 和 SCK 信号都由外部信号提供。

应用程序可以配置 I2S 模块为发送或接收模式，然后配置 I2S_ADD 为相反的模式。在发送过程中，软件同时操作 I2S 模块和 I2S_ADD 模块的寄存器和中断来实现全双工模式发送。

### 21.9.4. DMA 功能

DMA 功能与 SPI 模式完全一样，唯一不同的地方就是 I2S 模式不支持 CRC 功能。

## 21.10. I2S 中断

### 21.10.1. 状态标志位

SPI_STAT 寄存器中有 4 个可用的标志位，分别是 TBE、RBNE、TRANS 和 I2SCH，用户通过这些标志位可以全面监视 I2S 总线的状态。

- **发送缓冲区空标志(TBE)**

  当发送缓冲区为空时，TBE置位。软件可以通过写SPI_DATA寄存器将下一个数据写入发送缓冲区。

- **接收缓冲区非空标志(RBNE)**

  接收缓冲区非空时，RBNE置位，表示此时接收到一个数据，并已存入接收缓冲区中，软件可以通过读SPI_DATA寄存器来读取此数据。

- **I2S通信进行中标志(TRANS)**

  TRANS是用来指示当前传输是否正在进行或结束的状态标志，它由内部硬件置位和清除，无法通过软件进行操作。该标志位不会产生如何中断。

■ **I2S通道标志(I2SCH)**

I2SCH用来表明当前传输数据的通道信息，对PCM音频标准来说没有意义。在发送模式下，I2SCH标志在每次TBE由0变1时更新，在接收模式下，I2SCH标志在每次RBNE由0变1时更新。该标志位不会产生任何中断。

### 21.10.2. 错误标志

有三个错误标志：

■ **发送欠载错误标志(TXURERR)**

在从发送模式下，当有效的SCK信号开始发送时，如果发送缓冲区为空时，发送欠载错误标志TXURERR将会置位。

■ **接收过载错误标志(RXORERR)**

当接收缓冲区已满且又接收到一个新的数据时，接收过载错误标志RXORERR置位。当接收过载发生时，接收缓冲区中的数据没有更新，新接收的数据丢失。

■ **帧错误(FERR)**

在从I2S模式下，I2S模块监视I2S_WS信号，如果I2S_WS信号在一个错误的位置发生翻转，将会置位FERR帧错误标志位。

*表 21-10. I2S 中断*总结了 I2S 中断事件和相应的使能位。

**表 21-10. I2S 中断**

| 中断事件 | 描述 | 清除方式 | 中断使能位 |
|---|---|---|---|
| TBE | 发送缓冲区空 | 写 SPI_DATA 寄存器 | TBEIE |
| RBNE | 接收缓冲区非空 | 读 SPI_DATA 寄存器 | RBNEIE |
| TXURERR | 发送欠载错误 | 读 SPI_STAT 寄存器 | ERRIE |
| RXORERR | 接收过载错误 | 读 SPI_DATA 寄存器，然后再读 SPI_STAT 寄存器 | |
| FERR | I2S 帧错误 | 读 SPI_STAT 寄存器 | |

## 21.11. SPI/I2S 寄存器

I2S1_add 基地址：0x4000 3400

SPI1/I2S1 基地址：0x4000 3800

SPI2/I2S2 基地址：0x4000 3C00

I2S2_add 基地址：0x4000 4000

SPI0 基地址：0x4001 3000

SPI3 基地址：0x4001 3400

SPI4 基地址：0x4001 5000

SPI5 基地址：0x4001 5400

### 21.11.1. 控制寄存器 0(SPI_CTL0)

地址偏移：0x00
复位值：0x0000 0000

该寄存器可以按字节(8位)或半字(16位)或字(32位)访问。

该寄存器在I2S模式下没有意义。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BDEN | BDOEN | CRCEN | CRCNT | FF16 | RO | SWNSSEN | SWNSS | LF | SPIEN | PSC [2:0] | | | MSTMOD | CKPL | CKPH |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15 | BDEN | 双向数据模式<br>0：2 线单向传输模式<br>1：1 线双向传输模式。数据在主机的 MOSI 引脚和从机的 MISO 引脚之间传输。 |
| 14 | BDOEN | 双向传输输出使能<br>当 BDEN 置位时，该位决定了数据的传输方向。<br>0：工作在只接收模式<br>1：工作在只发送模式 |
| 13 | CRCEN | CRC 计算使能<br>0：CRC 计算禁止<br>1：CRC 计算使能 |
| 12 | CRCNT | 下一次传输 CRC |

| | | |
|---|---|---|
| | | 0：下一次传输值为数据<br>1：下一次传输值为 CRC 值(TCR)<br>当数据传输由 DMA 管理时，CRC 值由硬件传输，该位应该被清零。<br>在全双工和只发送模式下，当最后一个数据写入 SPI_DATA 寄存器后应将该位置 1。在只接收模式下，在接收完倒数第二个数据后应将该位置 1。 |
| 11 | FF16 | 数据帧格式<br>0：8 位数据帧格式<br>1：16 位数据帧格式 |
| 10 | RO | 只接收模式<br>当 BDEN 清零时，该位决定了数据的传输方向。<br>0：全双工模式<br>1：只接收模式 |
| 9 | SWNSSEN | NSS 软件模式选择<br>0：NSS 硬件模式，NSS 电平取决于 NSS 引脚<br>1：NSS 软件模式，NSS 电平取决于 SWNSS 位<br>该位在 SPI TI 模式下没有意义。 |
| 8 | SWNSS | NSS 软件模式下 NSS 引脚选择<br>0：NSS 引脚拉低<br>1：NSS 引脚拉高<br>只有在 SWNSSEN 置位时，该位有效。<br>该位在 SPI TI 模式下没有意义。 |
| 7 | LF | 最低有效位先发模式<br>0：先发送最高有效位<br>1：先发送最低有效位<br>该位在 SPI TI 模式下没有意义。 |
| 6 | SPIEN | SPI 使能<br>0：SPI 设备禁止<br>1：SPI 设备使能 |
| 5:3 | PSC[2:0] | 主时钟预分频选择<br>000：PCLK/2　100：PCLK/32<br>001：PCLK/4　101：PCLK/64<br>010：PCLK/8　110：PCLK/128<br>011：PCLK/16　111：PCLK/256<br>当使用 SPI0，SPI3，SPI4 和 SPI5 时，PCLK=PCLK2，当使用 SPI1 和 SPI2 时，PCLK=PCLK1。 |
| 2 | MSTMOD | 主从模式使能<br>0：从机模式<br>1：主机模式 |
| 1 | CKPL | 时钟极性选择<br>0：SPI 为空闲状态时，CLK 引脚拉低 |

| | | |
|---|---|---|
| | | 1：SPI 为空闲状态时，CLK 引脚拉高 |
| 0 | CKPH | 时钟相位选择<br>0：在第一个时钟跳变沿采集第一个数据<br>1：在第二个时钟跳变沿时钟跳变沿采集第一个数据 |

### 21.11.2. 控制寄存器 1(SPI_CTL1)

地址偏移：0x04
复位值：0x0000 0000

该寄存器可以按字节(8位)或半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 1002 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | TBEIE | RBNEIE | ERRIE | TMOD | 保留. | NSSDRV | DMATEN | DMAREN |
| | | | | | | | | rw | rw | rw | rw | | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7 | TBEIE | 发送缓冲区空中断使能<br>0：TBE 中断禁止<br>1：TBE 中断使能。当 TBE 置位时，产生中断。 |
| 6 | RBNEIE | 接收缓冲区非空中断使能<br>0：RBNE 中断禁止.<br>1：RBNE 中断使能。当 RBNE 置位时，产生中断。 |
| 5 | ERRIE | 错误中断使能<br>0：错误中断禁止<br>1：错误中断使能。当 CRCERR 位，CONFERR 位，RXORERR 位或者 TXURERR 位置 1 时，产生中断。 |
| 4 | TMOD | SPI TI 模式使能<br>0：SPI TI 模式禁止<br>1：SPI TI 模式使能 |
| 3 | 保留 | 必须保持复位值。 |
| 2 | NSSDRV | NSS 输出使能<br>0：NSS 输出禁止<br>1：NSS 输出使能。<br>当 SPI 使能时，如果 NSS 引脚配置为输出模式，NSS 引脚在主模式时被拉低。如果 NSS 引脚配置为输入模式，NSS 引脚在主模式时被拉高，此时该位无效。 |
| 1 | DMATEN | 发送缓冲区 DMA 使能 |

0：发送缓冲区 DMA 禁止
1：发送缓冲区 DMA 使能。当 SPI_STAT 中的 TBE 置位时，将会在相应的 DMA 通道上产生一个 DMA 请求。

0 DMAREN 接收缓冲区 DMA 使能
0：接收缓冲区 DMA 禁止
1：接收缓冲区 DMA 使能。当 SPI_STAT 中的 RBNE 置位时，将会在相应的 DMA 通道上产生一个 DMA 请求。

### 21.11.3. 状态寄存器(SPI_STAT)

地址偏移：0x08
复位值：0x0000 0002

该寄存器可以按字节(8位)或半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | FERR | TRANS | RXORERR | CONFERR | CRCERR | TXURERR | I2SCH | TBE | RBNE |
| | | | | | | | rc_w0 | r | r | r | rc_w0 | r | r | r | r |

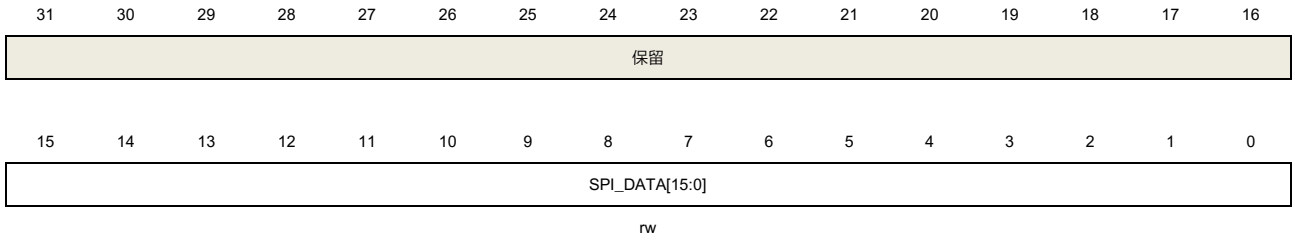
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:9 | 保留 | 必须保持复位值。 |
| 8 | FERR | 帧错误<br>SPI TI 模式：<br>0：没有 TI 模式帧错误发生<br>1：TI 模式帧错误发生<br>I2S 模式：<br>0：没有 I2S 帧错误发生<br>1：I2S 帧错误发生<br>该位由硬件置位，可以通过写 0 清除。 |
| 7 | TRANS | 通信进行中标志<br>0：SPI 或 I2S 空闲<br>1：SPI 或 I2S 当前正在发送或接收数据<br>该位由硬件置位和清除。 |
| 6 | RXORERR | 接收过载错误标志<br>0：没有接收过载错误发生<br>1：接收过载错误发生<br>该位由硬件置位，软件序列清零。软件序列为：先读 SPI_DATA 寄存器，然后读 SPI_STAT 寄存器。 |
| 5 | CONFERR | SPI 配置错误<br>0：无配置错误发生 |

| | | |
|---|---|---|
| | | 1：配置错误发生(主机模式下，在硬件 NSS 模式时 NSS 引脚被拉低，或者软件 NSS 模式时 SWNSS 位为 0，都会产生 CONFERR 错误)<br>该位由硬件置位，软件序列清零。软件序列为：先读或写 SPI_STAT 寄存器，然后写 SPI_CTL0 寄存器。<br>I2S 模式下不使用该位。 |
| 4 | CRCERR | SPI CRC 错误标志<br>0：SPI_RCRC 值等于最后接收到的 CRC 值<br>1：SPI_RCRC 值不等于最后接收到的 CRC 值<br>该位由硬件置位，可以通过写 0 清除。<br>I2S 模式下不使用该位。 |
| 3 | TXURERR | 发送欠载错误标志<br>0：无发送欠载错误发生<br>1：发送欠载错误发生<br>该位由硬件置位，通过读 SPI_STAT 寄存器清除。<br>SPI 模式下不使用该位。 |
| 2 | I2SCH | I2S 通道标志<br>0：下一个将要发送或刚刚接收到的数据属于左通道<br>1：下一个将要发送或刚刚接收到的数据属于右通道<br>该位由硬件置位和清除。<br>SPI 模式下该位无用，I2S PCM 模式下该位没有意义。 |
| 1 | TBE | 发送缓冲区空<br>0：发送缓冲区非空<br>1：发送缓冲区空 |
| 0 | RBNE | 接收缓冲区非空<br>0：接收缓冲区空<br>1：接收缓冲区非空 |

### 21.11.4. 数据寄存器(SPI_DATA)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器可以按字节(8位)或半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|

| | | |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | SPI_DATA[15:0] | 数据传输寄存器值<br>硬件有两个缓冲区：发送缓冲区和接收缓冲区。向 SPI_DATA 写数据将会把数据存入发送缓冲区，从 SPI_DATA 读数据，将从接收缓冲区获得数据。<br>当数据帧格式为 8 位时，SPI_DATA[15:8]强制为 0，SPI_DATA[7:0]用来发送和接收数据，发送和接收缓冲区都是 8 位。如果数据帧格式为 16 位，SPI_DATA[15:0]用于发送和接收数据，发送和接收缓冲区也是 16 位。 |

### 21.11.5. CRC 多项式寄存器(SPI_CRCPOLY)

地址偏移：0x10
复位值：0x0000 0007

该寄存器可以按字节(8位)或半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | CRCPOLY[15:0] | CRC 多项式寄存器值<br>该值包含了 CRC 多项式，用于 CRC 计算，默认值为 0007h。 |

### 21.11.6. 接收 CRC 寄存器(SPI_RCRC)

地址偏移：0x14
复位值：0x0000 0000

该寄存器可以按字节(8位)或半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|

| | | |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | RCRC[15:0] | 接收 CRC 寄存器值<br>当 SPI_CTL0 中的 CRCEN 置位时，硬件计算接收数据的 CRC 值，并保存到 RCR 寄存器中。如果是 8 位数据帧格式，CRC 计算基于 CRC8 标准进行，保存数据到 RCRC[7:0]。如果是 16 位数据帧格式，CRC 计算基于 CRC16 标准进行，保存数据到 RCRC[15:0]。<br>硬件在接收到每个数据位后都会计算 CRC 值，当 TRANS 置位时，读该寄存器将返回一个中间值。<br>当 SPI_CTL0 寄存器中的 CRCEN 位和 SPIEN 位清零时，该寄存器复位。 |

### 21.11.7. 发送 CRC 寄存器(SPI_TCRC)

地址偏移：0x18
复位值：0x0000 0000

该寄存器可以按字节(8位)或半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | TCRC[15:0] | 发送 CRC 寄存器值<br>当 SPI_CTL0 中的 CRCEN 置位时，硬件计算发送数据的 CRC 值，并保存到 TCR 寄存器中。如果是 8 位数据帧格式，CRC 计算基于 CRC8 标准进行，保存数据到 TCRC[7:0]。如果是 16 位数据帧格式，CRC 计算基于 CRC16 标准进行，保存数据到 TCRC[15:0]。<br>硬件在发送出每个数据位后都会计算 CRC 值，当 TRANS 置位时，读该寄存器将返回一个中间值。不同的数据帧格式(SPI_CTL0 中的 LF 位决定)将会得到不同的 CRC 值。<br>当 SPI_CTL0 寄存器中的 CRCEN 位和 SPIEN 位清零时，该寄存器复位。 |

### 21.11.8. I2S 控制寄存器(SPI_I2SCTL)

地址偏移：0x1C
复位值：0x0000 0000

该寄存器可以按字节(8位)或半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | I2SSEL | I2SEN | I2SOPMOD[1:0] | | PCMSMOD | 保留 | I2SSTD[1:0] | | CKPL | DTLEN[1:0] | | CHLEN |
| | | | | rw | rw | rw | | rw | | rw | | rw | rw | | rw |

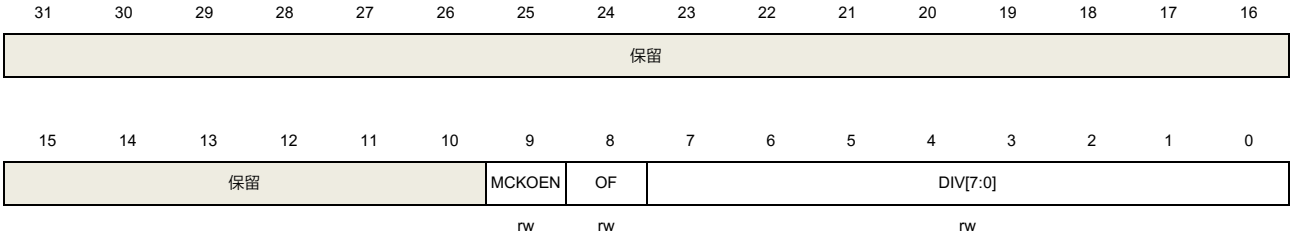
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11 | I2SSEL | I2S 模式选择<br>0：SPI 模式<br>1：I2S 模式<br>当 SPI 模式或 I2S 模式关闭时配置该位。 |
| 10 | I2SEN | I2S 使能<br>0：I2S 禁止<br>1：I2S 使能<br>SPI 模式不使用该位。 |
| 9:8 | I2SOPMOD[1:0] | I2S 运行模式<br>00：从机发送模式<br>01：从机接收模式<br>10：主机发送模式<br>11：主机接收模式<br>当 I2S 模式关闭时配置该位。SPI 模式不使用该位。 |
| 7 | PCMSMOD | PCM 帧同步模式<br>0：短帧同步<br>1：长帧同步<br>只有在 PCM 标准下，该位才有意义。<br>当 I2S 模式关闭时配置该位。SPI 模式不使用该位。 |
| 6 | 保留 | 必须保持复位值 |
| 5:4 | I2SSTD[1:0] | I2S 标准选择<br>00：I2S 飞利浦标准<br>01：MSB 对齐标准<br>10：LSB 对齐标准<br>11：PCM 标准<br>当 I2S 模式关闭时配置该位。SPI 模式不使用该位。 |
| 3 | CKPL | 空闲状态时钟极性<br>0：I2S_CK 空闲状态为低电平<br>1：I2S_CK 空闲状态为高电平<br>当 I2S 模式关闭时配置该位。SPI 模式不使用该位。 |
| 2:1 | DTLEN[1:0] | 数据长度<br>00：16 位 |

| | | |
|---|---|---|
| | | 01：24 位<br>10：32 位<br>11：保留<br>当 I2S 模式关闭时配置该位。SPI 模式不使用该位。 |
| 0 | CHLEN | 通道长度<br>0：16 位<br>1：32 位<br>通道长度必须大于或等于数据长度。<br>当 I2S 模式关闭时配置该位。SPI 模式不使用该位。 |

### 21.11.9. I2S 时钟预分频寄存器(SPI_I2SPSC)

地址偏移：0x20
复位值：0x0000 0002

该寄存器可以按字节(8位)或半字(16位)或字(32位)访问。



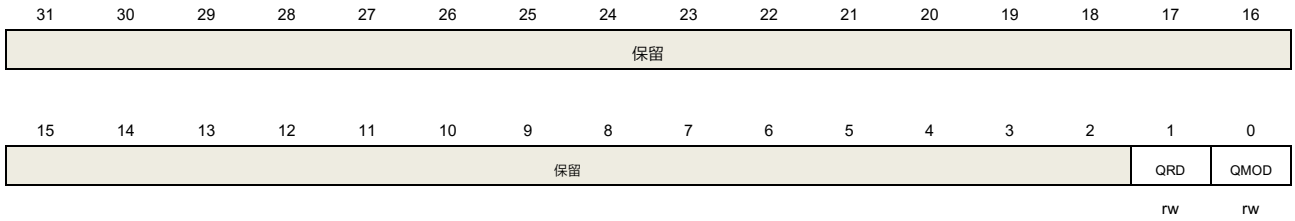

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:10 | 保留 | 必须保持复位值。 |
| 9 | MCKOEN | I2S_MCK 输出使能<br>0：I2S_MCK 输出禁止<br>1：I2S_MCK 输出使能<br>当 I2S 模式关闭时配置该位。<br>SPI 模式不使用该位。 |
| 8 | OF | 预分频器的奇系数<br>0：实际分频系数为 DIV * 2<br>1：实际分频系数为 DIV * 2 + 1<br>当 I2S 模式关闭时配置该位。SPI 模式下不使用该位。 |
| 7:0 | DIV[7:0] | 预分频器的分频系数<br>实际分频系数是 DIV * 2 + OF。<br>DIV 不能为 0.<br>当 I2S 模式关闭时配置该位。SPI 模式下不使用该位。 |

### 21.11.10. SPI5 四线 SPI 控制寄存器(SPI_QCTL)

地址偏移：0x80

复位值：0x0000 0000

该寄存器可以按字节(8位)或半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |
| 1 | QRD | 四线 SPI 模式读选择<br>0：SPI 四线模式写操作<br>1：SPI 四线模式读操作<br>该位仅能在 SPI 未通信时配置(TRANS 位清零)。<br>该位仅适用于 SPI5。 |
| 0 | QMOD | 四线 SPI 模式使能<br>0：SPI 工作在单路模式<br>1：SPI 工作在四线模式<br>该位仅能在 SPI 未通信时配置(TRANS 位清零)。<br>该位仅适用于 SPI5。 |

# 22. 数字摄像头接口(DCI)

## 22.1. 简介

数字摄像头接口是一个同步并行接口，可以从数字摄像头捕获视频和图像信息。它支持不同的颜色空间图像，例如YUV/RGB，另外支持压缩数据的JPEG格式图像。

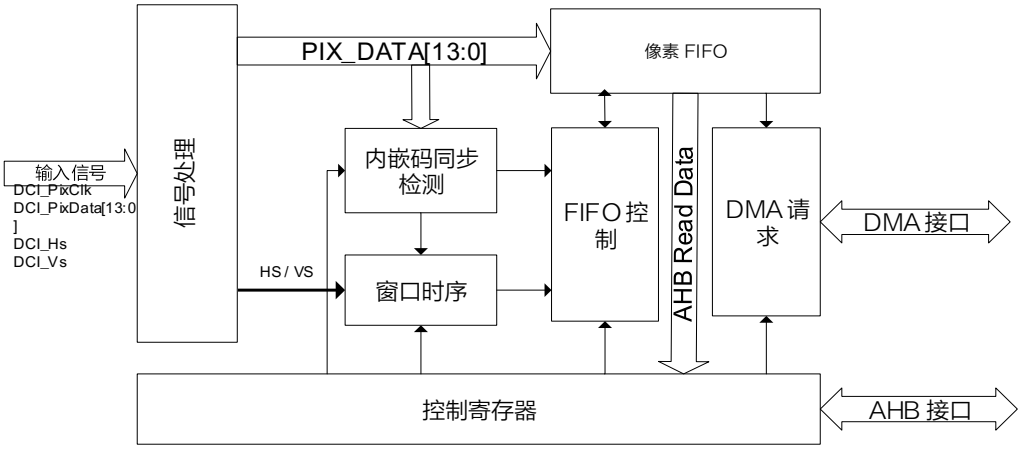
## 22.2. 主要特性

- 数字视频和图像的捕获
- 支持8位、10位、12位或14位并行接口
- DMA高效传输
- 支持视频和图像裁剪
- 支持不同的像素数字编码格式，如YCbCr422/RGB565
- 支持JPEG压缩格式
- 支持内嵌码同步和硬件同步

## 22.3. 结构框图

数字摄像头接口包含以下模块：信号处理单元、像素FIFO、FIFO控制器、窗口时序发生器、内嵌码同步检测器、DMA接口和控制寄存器。

**图 22-1. DCI 模块示意图**




信号处理单元根据外部输入信号，产生有用的信号信息，为其他的内部模块所用。为确保信号处理单元工作正常，HCLK的频率要高于像素时钟频率的2.5倍。

内嵌码同步检测用于内嵌码同步模式。DCI使用内嵌码同步模式时，视频同步信息内嵌于像素数据，并无硬件水平或垂直同步信号(DCI_HSYNC或DCI_VSYNC)。DCI通过内嵌码同步检测器从像素数据提取同步信息，然后根据这些信息重新恢复水平和垂直同步信号。

窗口时序模块具有图片剪裁功能。该模块通过来自DCI接口或内嵌码同步检测器的同步信号计算像素点的位置，然后根据寄存器DCI_CWSPOS和DCI_CWSZ的配置决定是否接收该像素点数据。

DCI用一个4字(32位)FIFO缓存接收到的数据。如果DMA模式使能，当收到一个完整的32位数据的时候，DMA接口置位一个DMA请求。控制寄存器提供DCI和软件之间的接口。

## 22.4. 信号描述

**表 22-1. DCI 引脚**

| 方向 | 引脚名称 | 信号 | 位宽 | 描述 |
|---|---|---|---|---|
| I | DCI_PIXCLK | DCI_PixClk | 1 | DCI 像素时钟 |
| I | DCI_D | DCI_PixData | 14 | DCI 像素数据 |
| I | DCI_HSYNC | DCI_HSYNC | 1 | DCI 水平同步 |
| I | DCI_VSYNC | DCI_VSYNC | 1 | DCI 垂直同步 |

## 22.5. 功能说明

### 22.5.1. DCI 硬件同步模式

在DCI硬件同步模式(DCI_CTL寄存器的ESM为0)，DCI_HSYNC和DCI_VSYNC分别用来表示一行的开始和一帧的开始。DCI在DCI_PIXCLK的上升沿或下降沿(时钟的极性通过DCI_CTL寄存器的CKS位配置)，从DCI_D[13:0]，捕获像素数据。

**图 22-2. 硬件同步模式**

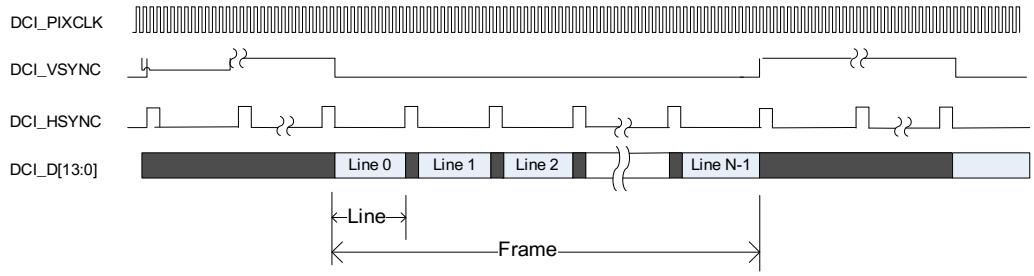
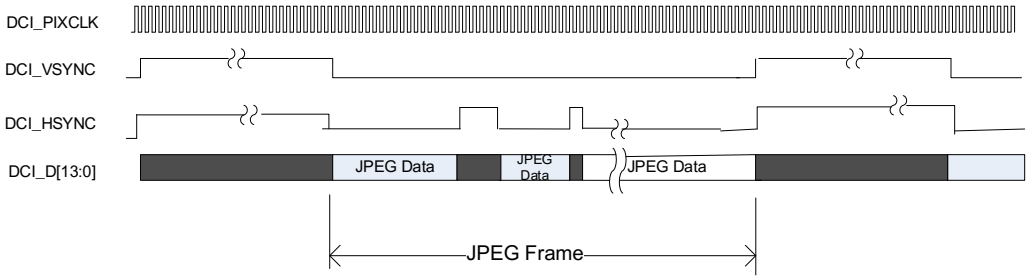



*图 22-2. 硬件同步模式*假设 DCI_HSYNC 和 DCI_VSYNC 消隐期间的极性为高电平，所以 DCI_D 线仅在 DCI_HSYNC 和 DCI_VSYNC 都为低电平期间是有效的。

#### JPEG 模式

DCI在硬件同步模式时，支持JPEG视频/图像压缩格式。在JPEG模式(DCI_CTL寄存器JM置1)，DCI_VSYNC表示一帧的开始，DCI_HSYNC用作数据流有效信号。

图 22-3. 硬件同步模式之 JPEG 格式




### 22.5.2. 内嵌码同步模式

DCI支持内嵌码同步模式。这一模式仅用到DCI接口的DCI_D和DCI_PIXCLK信号，同步信息内嵌在像素数据中。通过置位DCI_CTL寄存器的ESM位，并且清除JM位，使能内嵌码同步模式。

在内嵌码同步模式，行和帧同步信息被编码为同步码并嵌入像素数据中。有4种同步码：行开始(LS)，行结束(LE)，帧开始(FS)和帧结束(FE)。该模式数据宽度强制为8，并且每个同步码由4字节序列组成：FF-00-00-MN，MN在DCI_SC寄存器定义。在内嵌码同步模式，0xFF和0x00不应出现在像素数据中以避免误解。

使能内嵌码同步模式之后，DCI开始检测同步码，并恢复行/帧同步信息。例如，如果DCI检测到一个帧结束码以及一个帧开始码，它开始捕获新的帧。

当检测到一个同步码，通过配置DCI_SCUMSK，可能仅需要比较FF_00_00_MN序列MN字节的几位。DCI仅比较DCI_SCUMSK寄存器的非屏蔽位。例如：DCI_SC寄存器的LS位为A5，DCI_SCUMSK的LSM位是F0，DCI将仅比较LS同步码的高4位，因此FF-00-00-A6也将被检测为LS码。

### 22.5.3. 用快照或连续捕获模式捕获数据

DCI支持两种捕获模式：快照和连续捕获。捕获模式通过DCI_CTL寄存器的SNAP位配置。

正确配置之后，使能DCI并置位DCI_CTL寄存器的CAP位，DCI开始检测帧开始信号。一旦检测到帧开始信号，DCI开始捕获数据。在快照模式(SNAP=1),当一帧被捕获之后，DCI自动停止捕获并清除CAP位，而若在连续模式，DCI将准备捕获下一帧。在连续模式，DCI捕获频率在FR[1:0]位域定义。如果FR[1:0]=00，DCI捕获每一帧，如果FR[1:0]=01，DCI将每隔一帧捕获一次。

在连续模式，当DCI正在捕获数据的时候，软件可以在任意时间清除CAP位，但DCI并不立即停止捕获。它总是在捕获当前帧之后停止。软件应读回CAP位，以确认是否DCI停止生效。

### 22.5.4. 窗口功能

DCI支持窗口功能，该功能能够从捕获到的帧剪裁图像的一部分。该功能通过设置DCI_CTL寄存器的WDEN位，在JPEG子模式使能该功能是禁止的。

在捕获期间，DCI不断的计数和计算像素的水平和垂直位置，并且将该位置与剪裁窗口寄存器(DCI_CWSPOS和DCI_CWSZ)的值进行比较，然后丢弃剪裁窗口外的像素数据，仅将位于窗口内的数据压入数据FIFO。

如果一帧已经结束，但DCI_CWSZ定义的垂直行数还没有到达，这种情况下也将触发帧结束标志并且DCI停止捕获。

### 22.5.5. 像素格式、数据填充和 DMA 接口

DCI支持包含YCbCr422/RGB565等多种像素编码格式，但是DCI只接收这些像素数据，将像素数据补充成全字，并将其压入像素FIFO。DCI不执行任何像素格式转换或数据处理，不关心像素格式细节。

DCI使用32位宽的数据缓冲器在DCI接口和像素FIFO之间传递数据。在这一模块有两种填充方法：字节填充和半字填充，具体使用哪一种取决于DCI接口的数据宽度。数据宽度由DCI_CTL寄存器的DCIF[1:0]配置，在JPEG子模式和内嵌码同步模式，数据宽度固定为8。

当收到一个完整的32位数据的时候，DMA接口发送DMA请求。

#### 字节填充模式

如果DCI接口是8位，使用字节填充模式。在字节填充模式下，四个字节被填充到32位数据缓冲区，在Non-JPEG模式，如果数据缓冲区满或者到达行尾，DCI将压32位数据缓冲区的数据进入像素FIFO。在JPEG子模式，如果数据缓冲区满或者到达帧结束，DCI接口将压32位数据缓冲区的数据进入像素FIFO。

**表 22-2. 字节填充模式下的存储视图**

| | | | |
|---|---|---|---|
| D3[7:0] | D2[7:0] | D1[7:0] | D0[7:0] |
| D7[7:0] | D6[7:0] | D5[7:0] | D4[7:0] |

#### 半字填充模式

如果DCI接口配置为10/12/14位，使用半字填充模式。在该模式下，通过高位填0，每像素数据扩展为16位。所以32位宽的数据缓冲区可以包含两个像素数据。当缓冲区满或行结束的时候，DC将压数据进入像素FIFO。

**表 22-3. 半字填充模式下的存储视图**

| | | | |
|---|---|---|---|
| 2'b00 | D1[13:0] | 2'b00 | D0[13:0] |
| 2'b00 | D3[13:0] | 2'b00 | D2[13:0] |
| 2'b00 | D5[13:0] | 2'b00 | D4[13:0] |
| 2'b00 | D7[13:0] | 2'b00 | D6[13:0] |

## 22.6. 状态、错误和中断

DCI有几个状态和错误标志位，中断可以根据这些标志判断。如果使能DCI_INTEN的相应使能位，这些状态和错误标志触发DCI全局中断。这些标志可以通过写1到DCI_INTC寄存器清除。

**表 22-4. 状态/错误标志**

| 状态标志名 | 解释 |
|---|---|
| ELF | 行结束标志 |
| EFF | 帧结束标志 |
| OVRF | FIFO 溢出标志 |
| VSF | 帧垂直同步消隐标志 |
| ESEF | 内嵌同步错误标志 |

## 22.7. DCI 寄存器

DCI基地址：0x5005 0000

### 22.7.1. 控制寄存器(DCI_CTL)

偏移地址：0x00
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | DCIEN | 保留 | | DCIF[1:0] | | FR[1:0] | | VPS | HPS | CKS | ESM | JM | WDEN | SNAP | CAP |
| | rw | | | rw | | rw | | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:15 | 保留 | 必须保持复位值 |
| 14 | DCIEN | DCI 使能<br>0：DCI 禁止<br>1：DCI 使能 |
| 13:12 | 保留 | 必须保持复位值 |
| 11:10 | DCIF[1:0] | DCI 数据格式<br>00：每个像素时钟捕获 8 位数据<br>01：每个像素时钟捕获 10 位数据<br>10：每个像素时钟捕获 12 位数据<br>11：每个像素时钟捕获 14 位数据 |
| 9:8 | FR[1:0] | 帧频率<br>在连续捕获模式，FR 定义帧捕获频率<br>00：捕获所有帧<br>01：每隔一帧捕获一次<br>10：每隔三帧捕获一次<br>11：保留 |
| 7 | VPS | 垂直同步极性选择<br>0：消隐期间低电平<br>1：消隐期间高电平 |
| 6 | HPS | 水平同步极性选择<br>0：消隐期间低电平<br>1：消隐期间高电平 |

| | | |
|---|---|---|
| 5 | CKS | 时钟极性选择<br>0：下降沿捕获<br>1：上升沿捕获 |
| 4 | ESM | 内嵌码同步模式<br>0：禁止内嵌码同步模式<br>1：使能内嵌码同步模式 |
| 3 | JM | JPEG 子模式<br>0：禁止 JPEG 子模式<br>1：使能 JPEG 子模式 |
| 2 | WDEN | 窗口使能<br>0：禁止窗口功能<br>1：使能窗口功能 |
| 1 | SNAP | 快照模式<br>0：连续捕获模式<br>1：快照模式 |
| 0 | CAP | 使能捕获<br>0：禁止帧捕获<br>1：使能帧捕获 |

### 22.7.2. 状态寄存器 0(DCI_STAT0)

地址偏移：0x04
复位值：0x0000 0003

该寄存器只能按字(32位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | | FV | VS | HS |
| | | | | | | | | | | | | | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:3 | 保留 | 必须保持复位值 |
| 2 | FV | FIFO 有效<br>0：FIFO 没有有效像素数据<br>1：FIFO 中像素数据有效 |
| 1 | VS | VS 引脚状态<br>0：不在垂直消隐期间<br>1：处于垂直消隐期间 |

| | | |
|---|---|---|
| 0 | HS | HS 引脚状态<br>0：不在水平消隐期间<br>1：处于水平消隐期间 |

### 22.7.3. 状态寄存器 1(DCI_STAT1)

地址偏移：0x08
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | ELF | VSF | ESEF | OVRF | EFF |
| | | | | | | | | | | | r | r | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:5 | 保留 | 必须保持复位值 |
| 4 | ELF | 行结束标志<br>0：没有行结束标志<br>1：DCI 捕获到一行 |
| 3 | VSF | 垂直同步标志<br>0：没有垂直同步标志<br>1：检测到垂直同步消隐 |
| 2 | ESEF | 内嵌码同步错误标志<br>0：没有内嵌码同步错误标志<br>1：检测到内嵌码同步错误 |
| 1 | OVRF | FIFO 溢出标志<br>0：没有 FIFO 溢出<br>1：发生 FIFO 溢出 |
| 0 | EFF | 帧结束标志<br>0：没有帧结束标志<br>1：帧被 DCI 捕获 |

### 22.7.4. 中断使能寄存器(DCI_INTEN)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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

| 保留 |
|---|

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | ELIE | VSIE | ESEIE | OVRIE | EFIE |
| | | | | | | | | | | | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:5 | 保留 | 必须保持复位值 |
| 4 | ELIE | 行结束中断使能<br>0：行结束标志不产生中断<br>1：行结束标志产生中断 |
| 3 | VSIE | 垂直同步中断使能<br>0：垂直同步标志不产生中断<br>1：垂直同步标志产生中断 |
| 2 | ESEIE | 内嵌码同步错误中断使能<br>0：内嵌码同步错误标志不产生中断<br>1：内嵌码同步错误标志产生中断 |
| 1 | OVRIE | FIFO 溢出中断使能<br>0：FIFO 溢出不产生中断<br>1：FIFO 溢出产生中断 |
| 0 | EFIE | 帧结束中断使能<br>0：帧结束标志不产生中断<br>1：帧结束标志产生中断 |

### 22.7.5. 中断标志寄存器(DCI_INTF)

地址偏移：0x10
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | ELIF | VSIF | ESEIF | OVRIF | EFIF |
| | | | | | | | | | | | r | r | r | r | r |

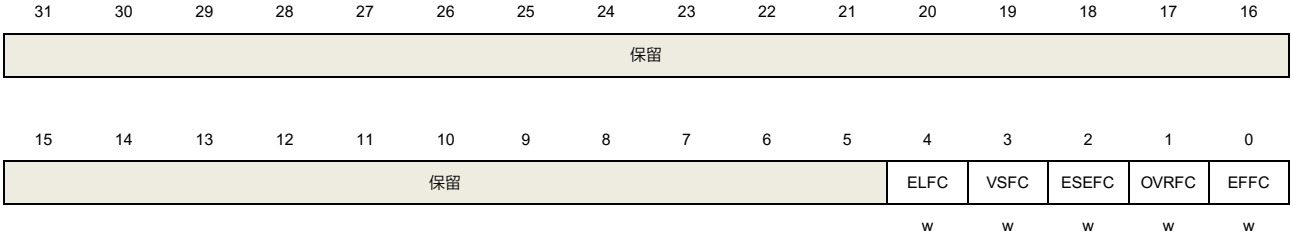
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:5 | 保留 | 必须保持复位值 |
| 4 | ELIF | 行结束中断标志 |

| | | |
|---|---|---|
| 3 | VSMF | 垂直同步中断标志 |
| 2 | ESEMF | 内嵌码同步错误中断标志 |
| 1 | OVRMF | FIFO 溢出中断标志 |
| 0 | EFMF | 帧结束中断标志 |

### 22.7.6. 中断标志清除寄存器(DCI_INTC)

地址偏移：0x14
复位值：0x0000 0000

该寄存器只能按字(32位)访问

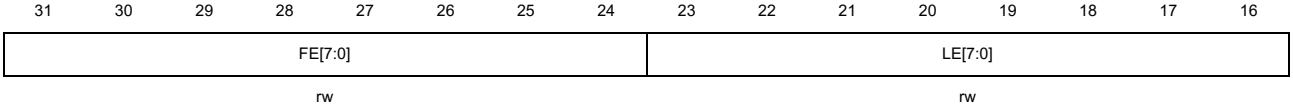


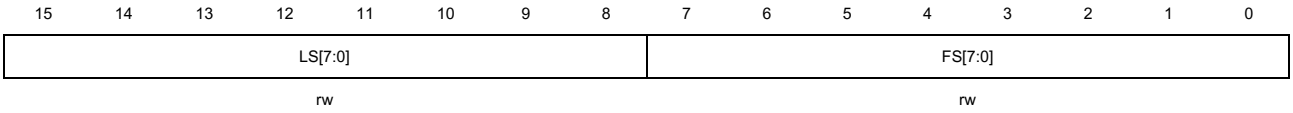

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:5 | 保留 | 必须保持复位值 |
| 4 | ELFC | 行结束中断标志清除<br>写 1 清除行结束中断标志 |
| 3 | VSFC | 垂直同步标志清除<br>写 1 清除垂直同步标志 |
| 2 | ESEFC | 内嵌码同步错误标志清除<br>写 1 清除内嵌码同步错误标志 |
| 1 | OVRFC | FIFO 溢出标志清除<br>写 1 清除 FIFO 溢出标志 |
| 0 | EFFC | 帧结束中断标志清除<br>写 1 清除帧结束中断标志 |

### 22.7.7. 同步码寄存器(DCI_SC)

地址偏移：0x18
复位值：0x0000 0000

该寄存器只能按字(32位)访问






| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | FE[7:0] | 内嵌同步模式的帧结束码 |
| 23:16 | LE[7:0] | 内嵌同步模式的行结束码 |
| 15:8 | LS[7:0] | 内嵌同步模式的行开始码 |
| 7:0 | FS[7:0] | 内嵌同步模式的帧开始码 |

### 22.7.8. 同步码屏蔽寄存器(DCI_SCUMSK)

地址偏移：0x1C
复位值：0x0000 0000

该寄存器只能按字(32位)访问

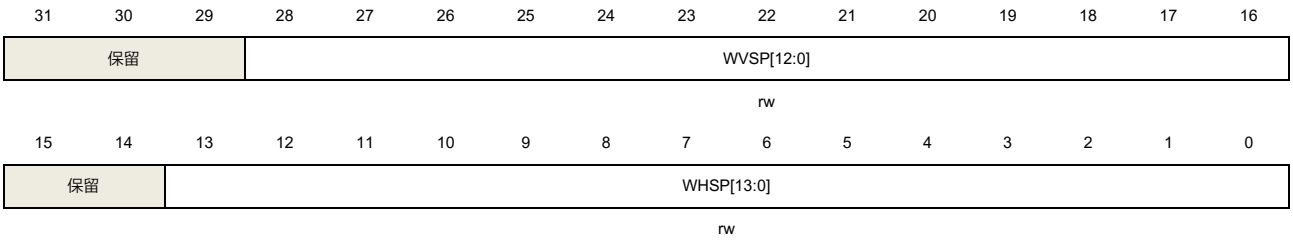



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | FEM[7:0] | 内嵌码同步模式下非屏蔽帧结束码 |
| 23:16 | LEM[7:0] | 内嵌码同步模式下非屏蔽行结束码 |
| 15:8 | LSM[7:0] | 内嵌码同步模式下非屏蔽行开始码 |
| 7:0 | FSM[7:0] | 内嵌码同步模式下非屏蔽帧开始码 |

### 22.7.9. 剪裁窗口开始位置寄存器(DCI_CWSPOS)

地址偏移：0x20
复位值：0x0000 0000

该寄存器只能按字(32位)访问



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:29 | 保留 | 必须保持复位值 |
| 28:16 | WVSP[12:0] | 窗口垂直开始位置<br>值为 0 表示着第一行，以此类推 |
| 15:14 | 保留 | 必须保持复位值 |
| 13:0 | WHSP[13:0] | 窗口水平开始位置<br>值为 0 表示着第一个像素时钟，以此类推 |

## 22.7.10. 剪裁窗口大小寄存器(DCI_CWSZ)

地址偏移：0x24
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:30 | 保留 | 必须保持复位值 |
| 29:16 | WVSZ[13:0] | 窗口垂直大小<br>WVSZ=X 表示 X+1 行 |
| 15:14 | 保留 | 必须保持复位值 |
| 13:0 | WHSZ[13:0] | 窗口水平大小<br>WHSZ=X 表示某一行有 X+1 个像素时钟 |

## 22.7.11. 数据寄存器(DCI_DATA)

地址偏移：0x28
复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DT3[[7:0] | | | | | | | | DT2[7:0] | | | | | | | |
| r | | | | | | | | r | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

| DT1[7:0] | DT0[7:0] |
|---|---|
| r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | DT3[7:0] | 像素字节 3 |
| 23:16 | DT2[7:0] | 像素字节 2 |
| 15:8 | DT1[7:0] | 像素字节 1 |
| 7:0 | DT0[7:0] | 像素字节 0 |

# 23. TFT-LCD 接口(TLI)

## 23.1. 简介

TLI(TFT-LCD接口)连接同步的LCD接口，并且为无源LCD显示屏提供像素数据，时钟以及时序信号。它支持不同的完全可编程的时序参数显示。一个内嵌的DMA不断的从系统存储器搬移数据到TLI然后输出到外部的LCD显示。TLI支持两个独立的显示层，并支持层窗口和层混叠功能。

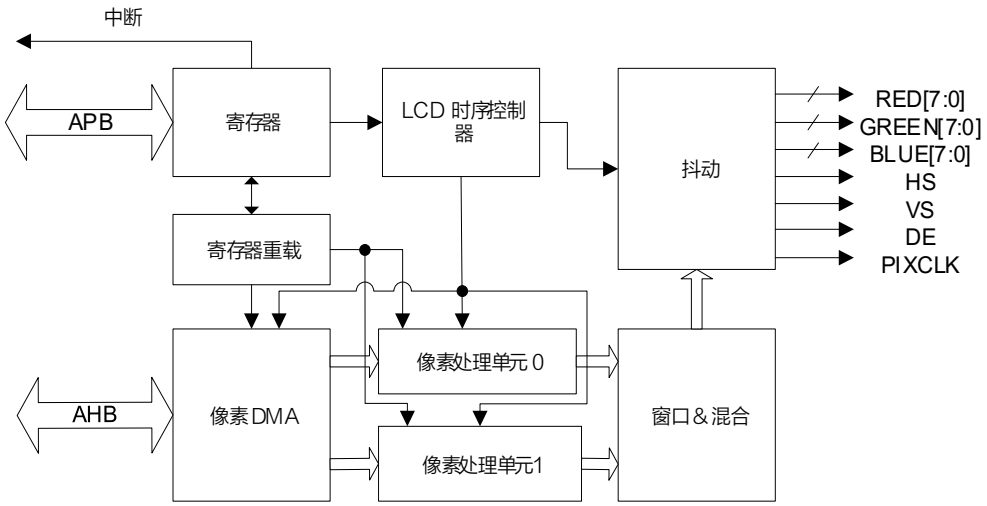
## 23.2. 主要特点

- 每像素最多24位并行数据输出；
- 支持高达2048*2048的分辨率；
- 完全可编程的时序参数；
- 内嵌DMA处理像素数据拷贝；
- 带有窗口和混合功能的两个独立的层；
- 支持多种像素格式：ARGB8888，RGB888，RGB565等；
- 支持CLUT(颜色查找表)和色键格式；
- 像素低位的抖动操作。

## 23.3. 结构框图

*图23-1. TLI模块框图*展示了TLI模块的结构框图。在TLI模块有3个时钟域。寄存器工作在APB时钟域，通过APB总线访问。像素DMA模块工作在AHB时钟域，从系统存储器获取像素数据需要使用AHB总线。剩下的模块工作在TLI时钟域。TLI时钟由PLLSAI-R 分频而得到。PLLSAI参数和分频因子在RCU模块配置。

**图 23-1. TLI 模块框图**



## 23.4. 信号线描述

TLI提供一个24位RGB并行显示接口，如*表23-1. TLI提供的显示接口的引脚*所示。

**表 23-1. TLI 提供的显示接口的引脚**

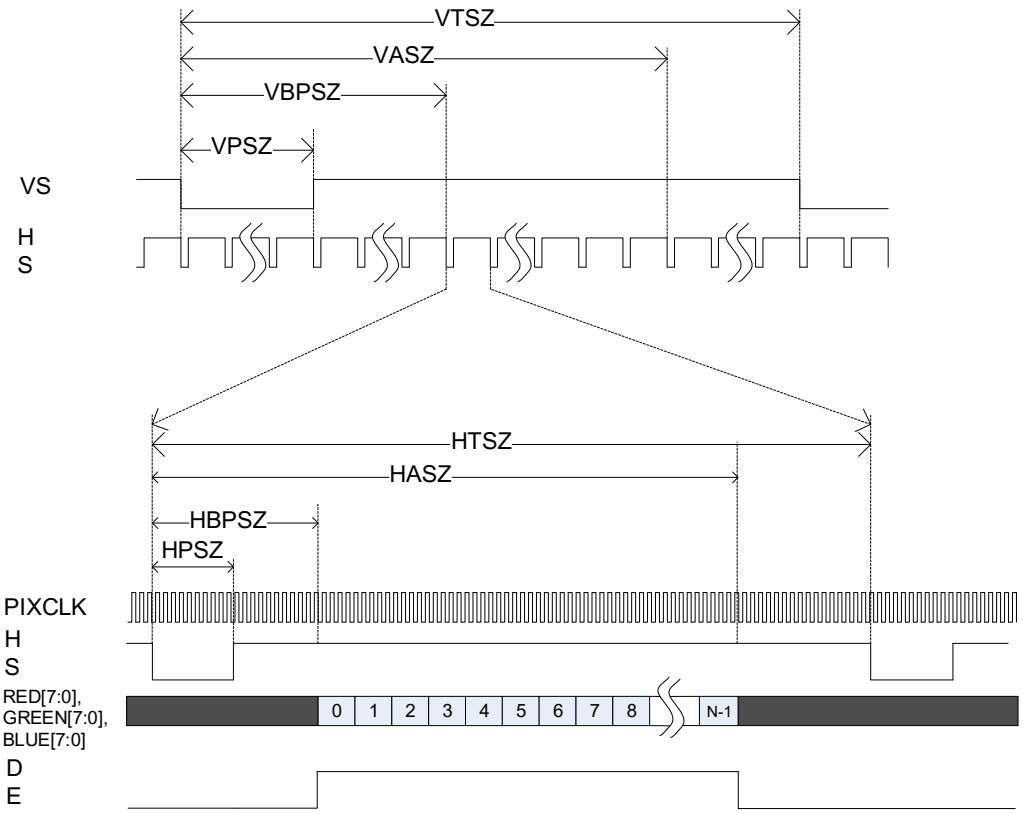
| 方向 | 名称 | 位宽 | 描述 |
|---|---|---|---|
| O | HS | 1 | 水平同步 |
| O | VS | 1 | 垂直同步 |
| O | DE | 1 | 数据使能 |
| O | PIXCLK | 1 | 像素时钟 |
| O | RED[7:0] | 8 | 红色像素数据 |
| O | GREEN[7:0] | 8 | 绿色像素数据 |
| O | BLUE[7:0] | 8 | 蓝色像素数据 |

## 23.5. 功能描述

### 23.5.1. LCD 显示时序

LCD接口是一个同步数据接口，包括像素时钟，像素数据以及水平和垂直同步信号。*图23-2. 显示时序图*展示了一帧的HS和VS信号时序。时序参数在TLI_SPSZ，TLI_BPSZ，TLI_ASZ和TLI_TSZ寄存器中配置。这些时序值寄存器假设第一个点的位置是(0, 0)。

**图 23-2. 显示时序图**




## 23.5.2. 像素 DMA 功能

根据寄存器模块的配置，像素DMA不断的从存储器读像素数据到内部PPU(像素处理单元)像素缓冲区。

使能之后，像素DMA开始从系统取像素数据，只要像素缓冲区未满，就将这些数据压进像素缓冲区。TLI总是用AHB BURST16方式取像素数据。

TLI支持两个独立的帧层，在系统中每层有不同的帧缓冲区地址。像素DMA仅有一个AHB访问端口，所以如果都使能的话，在取像素的时候，它将交替读取两层的数据。

TLI_LxFBADDR寄存器的FBADD位定义了每层的帧缓冲区地址。

TLI_LxFLLEN寄存器的FLL定义了一行的长度，以字节为单位。如果一行的长度是N，FLL=N+3。

在系统存储器里，两行之间可能有一些间隔存储空间，这一间隔空间信息在TLI_LxFLLEN寄存器的STDOFF位域定义。如果某行第一个像素的地址是M，那么下一行第一个像素的地址将是M+STDOFF。如果两行之间无间隔存储空间，则STDOFF=FLL-3。

TLI_LxFTLN寄存器的FTLN位域定义了一帧的行数。

### 23.5.3. 像素格式

像素DMA以字为单位，将像素数据压入PPU，然后由PPU负责将各种像素格式转换成内部ARGB8888格式。如*表23-2. 八种像素格式*所示，TLI支持多达8种像素格式。TLI_LxPPF寄存器的PPF[2:0]位域定义了像素格式。

ARGB8888格式要求每通道(Alpha，Red，Green和Blue)有8位数据。但是ARGB1555和ARGB4444格式的某些通道是少于8位的。 PPU通过拷贝高位填充到低位的方式，将其转换成ARGB8888。当处理RGB888和RGB565格式时，PPU假设Alpha=255，并且如果通道的位数少于8，也将拷贝高位填充到低位。

AL88, AL44和L8格式是LUT(颜色查找表)格式。在这些通道里， L是颜色查找表的地址。 TLI有两个内部颜色查找表：每层各一个。内部颜色查找表的大小是256x24bits (256个节点，每节点存储24位RGB值)。当处理LUT格式像素时，PPU从颜色查找表读出一个节点，并用这个节点值作为RGB值。由于颜色查找表的地址是8位的，如果L通道的位数少于8位，PPU也将拷贝高位填充到低位。颜色查找表的节点在复位后是不会被初始化的，因此在显示一个颜色查找表格式层之前，应用程序应该用TLI_LxLUT 寄存器，写入适当的值初始化颜色查找表。

每层都支持色键模式。TLI_LxCKEY 寄存器定义了一个RGB值。当某层的色键模式使能，PPU将会把该层每一个像素的RGB值与TLI_LxCKEY中的值相比较，如果值匹配，将会置该像素的ARGB值为0。

**表 23-2. 八种像素格式**

| PPF[2:0] | 像素格式 |
|---|---|
| 000 | ARGB8888 |
| 001 | RGB888 |
| 010 | RGB565 |
| 011 | ARGB1555 |
| 100 | ARGB4444 |
| 111 | AL88 |
| 101 | L8 |
| 110 | AL44 |

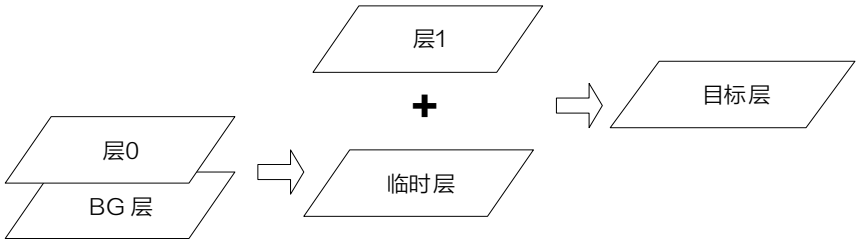
### 23.5.4. 层窗口和混合功能

TLI每层都支持窗口功能以及两层的混合功能。TLI首先执行每层的窗口操作，然后将两层混合成一帧。

窗口功能定义了一个显示窗口，每层有独立的窗口参数定义寄存器TLI_LxHPOS和TLI_LxVPOS。窗口参数定义了一层内的显示窗。窗口内的像素将保持它的原值，但是窗口外的像素值将被TLI_LxDC 寄存器定义的像素值替代。

混合单元首先混合层0和BG层得到一个临时层，然后混合层1和临时层，得到目的层。BG层的ARGB值在TLI_BGC寄存器定义。如果某层被禁止，混合功能使用该层的默认颜色。

**图 23-3. 混合过程框图**




### 混合公式

通用混合公式:

$$BC=BF_1*C+BF_2*C_S \quad (23\text{-}1)$$

- BC 混合的颜色
- BF1 混合因子 1
- C 当前层颜色
- $BF_2$ 混合因子 2
- $C_S$ 下一级层混合的颜色

当前像素的混合因子有两种取值，由寄存器配置。一种是归一化的像素Alpha乘以归一化的恒定Alpha，另一种是归一化的恒定Alpha。

## 23.5.5. Layer 配置重载

如上面所描述的，每层有自己的帧缓冲区，像素格式，窗口，默认颜色配置寄存器并且每个寄存器都有影子寄存器。影子寄存器与真正的寄存器共享相同的地址。每次当应用程序对层相关的寄存器地址执行写操作，相应的影子寄存器立即更新，但是直到一个重载操作前真正寄存器的值是不会改变的，而只有真正寄存器的值才是影响TLI功能的。

应用程序有两种方法触发一个重载操作：请求重载和帧消隐重载。对于请求重载模式，在应用程序设置TLI_RL寄存器的RQR位之后，TLI立即加载影子寄存器的值到真正寄存器。对于帧消隐重载模式，设置TLI_RL寄存器的FBR位之后，TLI等待帧垂直消隐，然后加载影子寄存器的值到真正寄存器。在两种模式下，重载成功完成之后，硬件自动清除RQR或FBR位。

## 23.5.6. 抖动

抖动模块为每一个像素通道加一个2位的伪随机值当18位接口用来显示24位数据的时候，该功能能够使图像更平滑。应用程序可以用TLI_CTL寄存器的DFEN位开启这一功能。

# 23.6. 中断

TLI中有以下错误和状态标志位。中断可以从这些状态判定，状态标志可以触发一个全局中断，

错误标志将触发错误中断。

**表 23-3. 状态标志**

| 状态标志位 | 描述 |
| --- | --- |
| LMF | 行标记标志 |
| LCRF | 层配置重载标志 |

**表 23-4. 错误标志**

| 错误标志位 | 描述 |
| --- | --- |
| TEF | 传输错误标志 |
| FEF | FIFO 错误标志 |

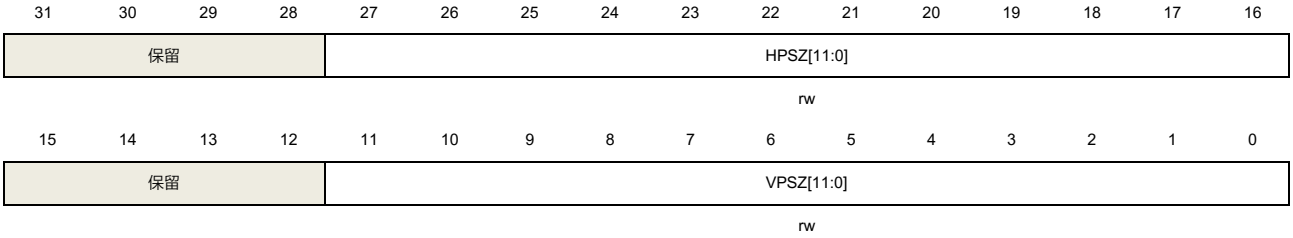
## 23.7. TLI 寄存器

TLI基地址：0x4001 6800

### 23.7.1. 同步脉冲宽度寄存器(TLI_SPSZ)

偏移地址：0x08
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:16 | HPSZ[11:0] | 水平同步脉冲宽度<br>HPSZ 值应该配置成水平同步脉冲像素的个数减 1。 |
| 15:12 | 保留 | 必须保持复位值。 |
| 11:0 | VPSZ[11:0] | 垂直同步脉冲宽度<br>VPSZ 值应该配置成垂直同步脉冲像素的个数减 1。 |

### 23.7.2. 后沿宽度寄存器(TLI_BPSZ)

偏移地址：0x0C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 27:16 | HBPSZ[11:0] | 水平后沿加同步脉冲的宽度<br>HBPSZ 值应该配置成水平后沿像素个数加同步脉冲像素个数减 1。 |
| 15:12 | 保留 | 必须保持复位值。 |
| 11:0 | VBPSZ[11:0] | 垂直后沿加同步脉冲的宽度<br>VBPSZ 值应该配置成垂直后沿像素个数加同步脉冲像素个数减 1。 |

### 23.7.3. 有效宽度寄存器(TLI_ASZ)

偏移地址：0x10
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | HASZ[11:0] | | | | | | | | | | | |
| | | | | rw | | | | | | | | | | | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:16 | HASZ[11:0] | 水平有效宽度加后沿像素和水平同步像素宽度<br>HASZ 值应该配置成水平有效宽度加后沿像素和水平同步像素个数减 1。 |
| 15:12 | 保留 | 必须保持复位值。 |
| 11:0 | VASZ[11:0] | 垂直有效宽度加后沿像素和垂直同步像素宽度<br>VASZ 值应该配置成垂直有效宽度加后沿像素和垂直同步像素个数减 1。 |

### 23.7.4. 总宽度寄存器(TLI_TSZ)

偏移地址：0x14
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | HTSZ[11:0] | | | | | | | | | | | |
| | | | | rw | | | | | | | | | | | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:16 | HTSZ[11:0] | 显示器的水平总宽度，包括有效宽度，后沿,同步脉冲和前沿<br>HTSZ 值应该配置成水平有效宽度像素的个数加后沿像素，前沿像素和同步脉冲像素减 1。 |
| 15:12 | 保留 | 必须保持复位值。 |
| 11:0 | VTSZ[11:0] | 显示器的垂直总宽度，包括有效宽度，后沿,同步脉冲和前沿<br>VTSZ 值应该配置成垂直有效宽度像素的个数加后沿像素，前沿像素和同步脉冲像素减 1. |

### 23.7.5. 控制寄存器(TLI_CTL)

偏移地址：0x18
复位值：0x0000 2220

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| HPPS | VPPS | DEPS | CLKPS | 保留 | | | | | | | | | | | DFEN |
| rw | rw | rw | rw | | | | | | | | | | | | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | RDB[2:0] | | | 保留 | GDB[2:0] | | | 保留 | BDB[2:0] | | | 保留 | | | TLIEN |
| | r | | | | r | | | | r | | | | | | rw |

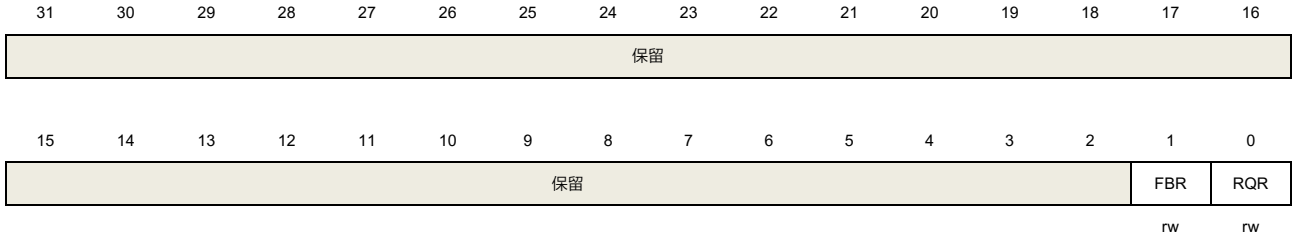
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | HPPS | 水平脉冲极性选择<br>0：水平同步脉冲低电平有效<br>1：水平同步脉冲高电平有效 |
| 30 | VPPS | 垂直脉冲极性选择<br>0：垂直同步脉冲低电平有效<br>1：垂直同步脉冲高电平有效 |
| 29 | DEPS | 非数据使能极性选择<br>0：非数据使能低电平有效<br>1：非数据使能高电平有效 |
| 28 | CLKPS | 像素时钟极性选择<br>0：像素时钟是 TLI 时钟<br>1：像素时钟是 TLI 时钟翻转 |
| 27:17 | 保留 | 必须保持复位值。 |
| 16 | DFEN | 抖动功能使能<br>0：禁止抖动功能 |

| | | |
|---|---|---|
| | | 1：使能抖动功能 |
| 15 | 保留 | 必须保持复位值。 |
| 14:12 | RDB[2:0] | 红色通道抖动位数<br>固定为 2，只读 |
| 11 | 保留 | 必须保持复位值。 |
| 10:8 | GDB[2:0] | 绿色通道抖动位数<br>固定为 2，只读 |
| 7 | 保留 | 必须保持复位值。 |
| 6:4 | BDB[2:0] | 蓝色通道抖动位数<br>固定为 2，只读 |
| 3:1 | 保留 | 必须保持复位值。 |
| 0 | TLIEN | TLI 使能位<br>0：禁止 TLI<br>1：使能 TLI |

### 23.7.6. 重载层配置寄存器(TLI_RL)

偏移地址：0x24
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

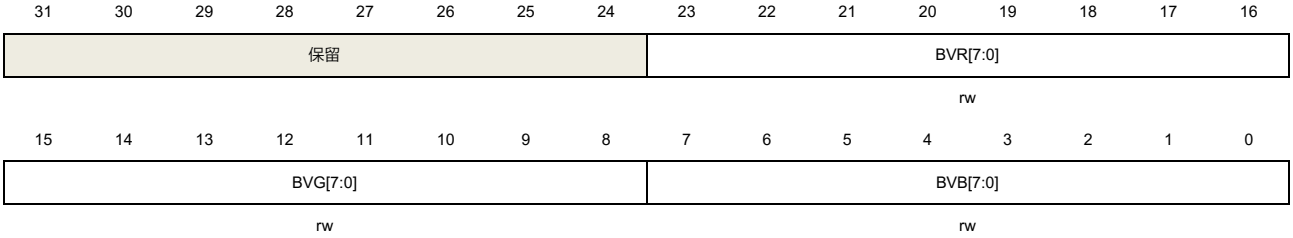



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |
| 1 | FBR | 帧消隐重载请求<br>此位通过软件置位，在重载之后由硬件清除。<br>0：禁止重载<br>1：层配置将在帧消隐时被重载进入真正寄存器。 |
| 0 | RQR | 立即重载请求<br>此位通过软件置位，在重载之后由硬件清除。<br>0：禁止重载<br>1：层配置将在该位置位之后被重载进入真正寄存器。 |

### 23.7.7. 背景色配置寄存器(TLI_BGC)

偏移地址：0x2C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

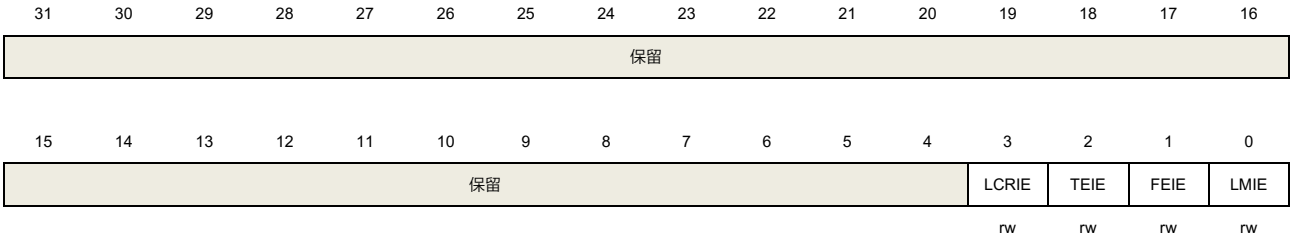



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值。 |
| 23:16 | BVR[7:0] | 背景红色值 |
| 15:8 | BVG[7:0] | 背景绿色值 |
| 7:0 | BVB[7:0] | 背景蓝色值 |

### 23.7.8. 中断使能寄存器(TLI_INTEN)

偏移地址：0x34
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:4 | 保留 | 必须保持复位值。 |
| 3 | LCRIE | 层配置重载中断使能<br>0：层配置重载标志将不产生中断<br>1：层配置重载标志将产生中断 |
| 2 | TEIE | 传输错误中断使能<br>0：传输错误标志将不产生中断<br>1：传输错误标志将产生中断 |

| | | |
|---|---|---|
| 1 | FEIE | FIFO 错误中断使能<br>0：FIFO 错误标志将不产生中断<br>1：FIFO 错误标志将产生中断 |
| 0 | LMIE | 行标记中断使能<br>0：行标记标志将不产生中断<br>1：行标记标志将产生中断 |

### 23.7.9. 中断标志寄存器(TLI_INTF)

地址偏移：0x38
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

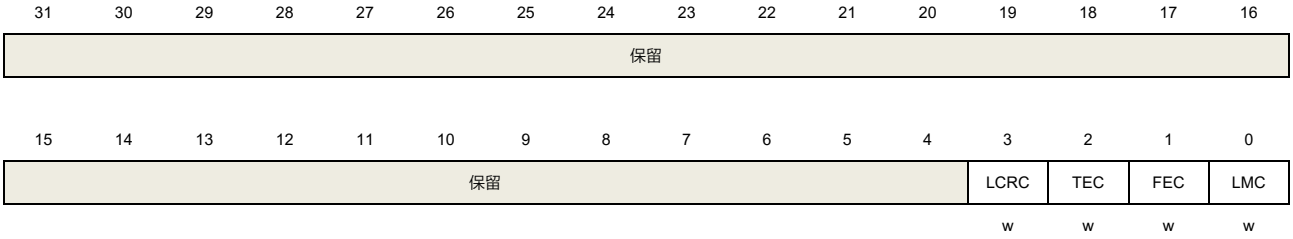
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | LCRF | TEF | FEF | LMF |
| | | | | | | | | | | | | r | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:4 | 保留 | 必须保持复位值。 |
| 3 | LCRF | 层配置重载标志<br>0：无层配置重载标志出现<br>1：由 TLI_RL 寄存器的 FBR 位置位触发了层配置重载 |
| 2 | TEF | 传输错误标志<br>0：无传输错误<br>1：一个传输错误在 AHB 总线上出现 |
| 1 | FEF | FIFO 错误标志<br>0：无 FIFO 错误标志<br>1：出现 FIFO 下溢错误 |
| 0 | LMF | 行标记标志<br>0：没有行标记标志<br>1：行数达到 TLI_LM 寄存器中设置的特定值 |

### 23.7.10. 中断标志清除寄存器(TLI_INTC)

地址偏移：0x3C
复位值：0x0000 0000

该寄存器只能按字(32位)访问。



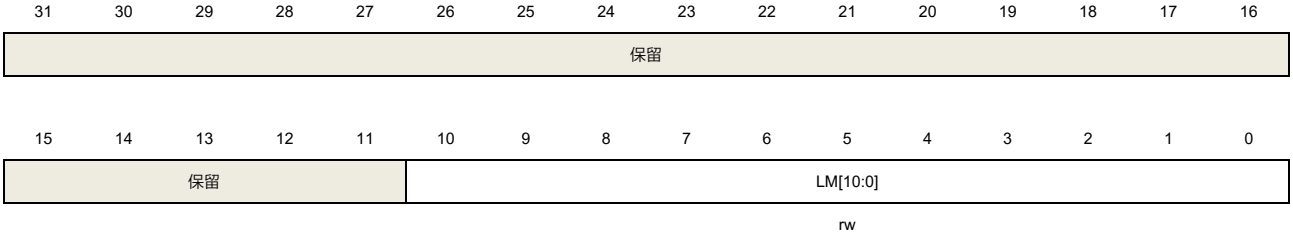

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:4 | 保留 | 必须保持复位值。 |
| 3 | LCRC | 层配置重载标志清除<br>写 1 清除层配置重载标志 |
| 2 | TEC | 传输错误标志清除<br>写 1 清除传输错误标志 |
| 1 | FEC | FIFO 错误标志清除<br>写 1 清除 FIFO 错误标志 |
| 0 | LMC | 行标记标志清除<br>写 1 清除行标记标志 |

## 23.7.11. 行标记寄存器(TLI_LM)

地址偏移: 0x40

复位值：0x0000 0000

该寄存器只能按字(32位)访问。



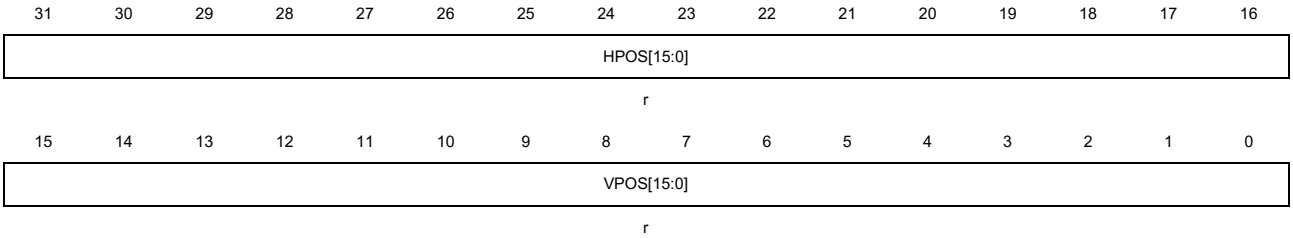

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:11 | 保留 | 必须保持复位值。 |
| 10:0 | LM[10:0] | 行标记值<br>当行数到达该值，TLI_INTF 寄存器的 LMF 位将置位。 |

## 23.7.12. 当前像素位置寄存器(TLI_CPPOS)

地址偏移：0x44

复位值：0x0000 0000

该寄存器只能按字(32位)访问。



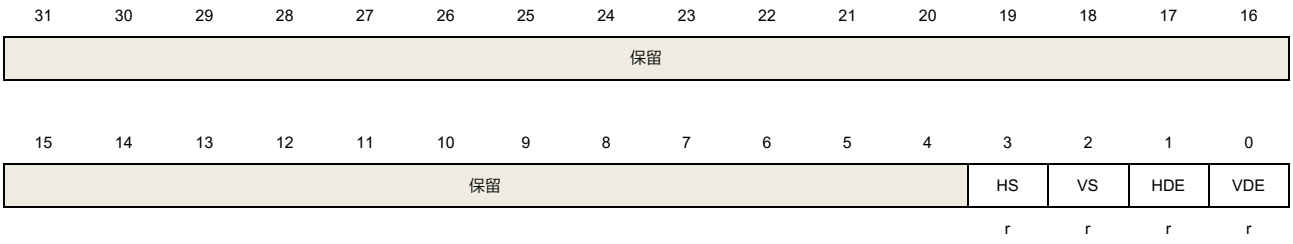

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | HPOS[15:0] | 水平位置<br>当前显示的像素的水平位置 |
| 15:0 | VPOS[15:0] | 垂直位置<br>当前显示的像素的垂直位置 |

## 23.7.13. 状态寄存器(TLI_STAT)

地址偏移：0x48

复位值：0x0000 000F

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:4 | 保留 | 必须保持复位值。 |
| 3 | HS | TLI 当前的 HS 状态 |
| 2 | VS | TLI 当前的 VS 状态 |
| 1 | HDE | 当前的 HDE 状态<br>0：TLI_CPPOS 寄存器 HPOS 并未位于 TLI_BPSZ 寄存器 HBPSZ 与 TLI_ASZ 寄存器 HASZ 之间<br>1：TLI_CPPOS 寄存器 HPOS 位于 TLI_BPSZ 寄存器 HBPSZ 与 TLI_ASZ 寄存器 HASZ 之间 |

| | | |
|---|---|---|
| 0 | VDE | 当前的 VDE 状态<br>0：TLI_CPPOS 寄存器 VPOS 并未位于 TLI_BPSZ 寄存器 VBPSZ 与 TLI_ASZ 寄存器 HASZ 之间<br>1：TLI_CPPOS 寄存器 VPOS 位于 TLI_BPSZ 寄存器 VBPSZ 与 TLI_ASZ 寄存器 VASZ 之间 |

## 23.7.14. 第 x 层控制寄存器(TLI_LxCTL)(x = 0, 1)

地址偏移：0x84+0x80*x x=0 or 1
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | LUTEN | 保留 | | CKEYEN | LEN |
| | | | | | | | | | | | rw | | | rw | rw |

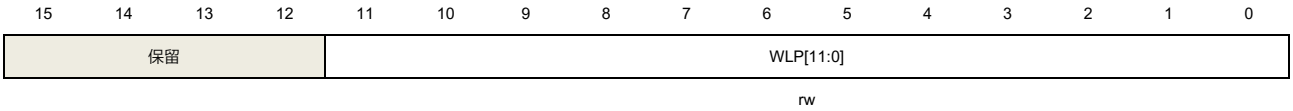
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:5 | 保留 | 必须保持复位值。 |
| 4 | LUTEN | LUT 使能<br>0：禁止 LUT<br>1：使能 LUT |
| 3:2 | 保留 | 必须保持复位值。 |
| 1 | CKEYEN | 色键使能<br>0：禁止色键功能<br>1：使能色键功能 |
| 0 | LEN | 层使能<br>0：禁止层<br>1：使能层 |

## 23.7.15. 第 x 层水平位置参数寄存器(TLI_LxHPOS)(x = 0, 1)

偏移地址：0x88+0x80*x x=0 or 1
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | WRP[11:0] | | | | | | | | | | | |
| | | | | rw | | | | | | | | | | | |

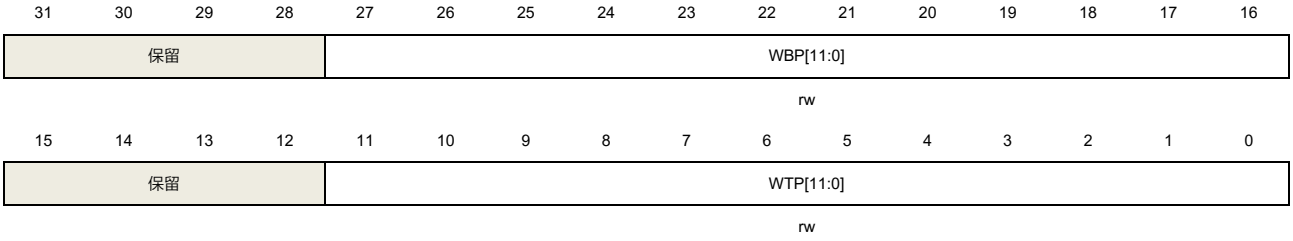



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:16 | WRP[11:0] | 窗口右侧位置 |
| 15:12 | 保留 | 必须保持复位值。 |
| 11:0 | WLP[11:0] | 窗口左侧位置 |

## 23.7.16. 第 x 层垂直位置参数寄存器(TLI_LxVPOS)(x = 0, 1)

地址偏移：0x8C+0x80*x x=0 or 1
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

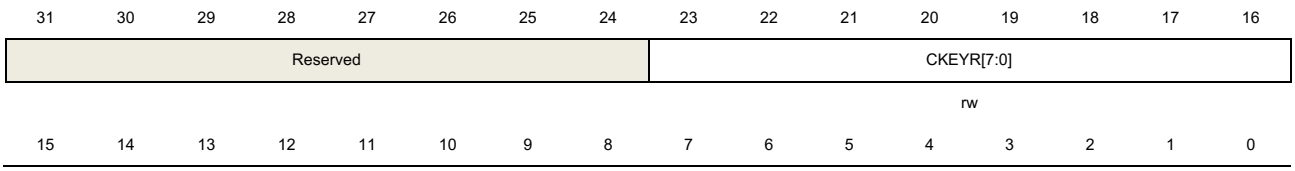



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:16 | WBP[11:0] | 窗口底部位置 |
| 15:12 | 保留 | 必须保持复位值。 |
| 11:0 | WTP[11:0] | 窗口顶部位置 |

## 23.7.17. 第 x 层色键值寄存器(TLI_LxCKEY)(x = 0, 1)

地址偏移：0x90+0x80*x x=0 or 1
复位值：0x0000 0000

该寄存器只能按字(32位)访问。



| CKEYG[7:0] | CKEYB[7:0] |
|---|---|
| rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值。 |
| 23:16 | CKEYR [7:0] | 色键红色值 |
| 15:8 | CKEYG [7:0] | 色键绿色值 |
| 7:0 | CKEYB [7:0] | 色键蓝色值 |

**注意**：如果某层的像素RGB等于TLI_LxCKEY寄存器定义的值，该像素RGB值复位为0。这意味着这些像素对其它层来说是透明的。

## 23.7.18. 第 x 层像素格式寄存器(TLI_LxPPF)(x = 0, 1)

地址偏移：0x94+0x80*x x=0 or 1
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

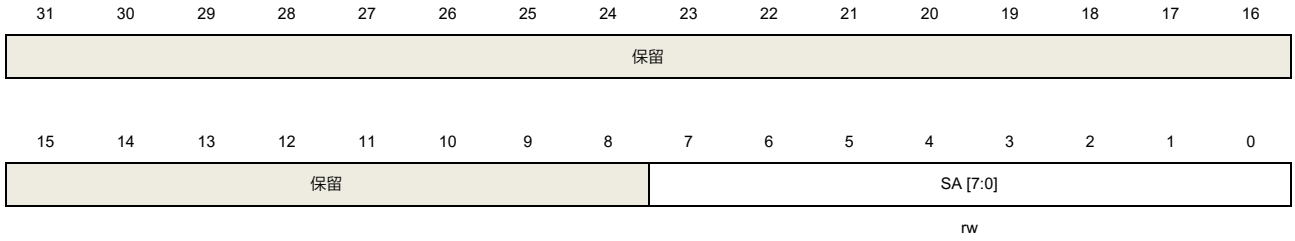
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:3 | 保留 | 必须保持复位值。 |
| 2:0 | PPF[2:0] | 像素格式<br>这些位配置像素格式<br>000：ARGB8888<br>001：RGB888<br>010：RGB565<br>011：ARGB1555<br>100：ARGB4444<br>101：L8<br>110：AL44<br>111：AL88 |

## 23.7.19. 第 x 层恒定 Alpha 寄存器(TLI_LxSA)(x = 0, 1)

地址偏移：0x98+0x80*x x=0 or 1

复位值：0x0000 00FF

该寄存器只能按字(32位)访问。

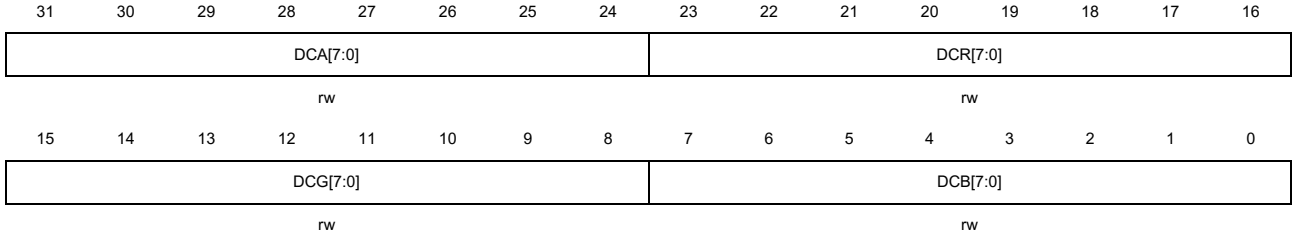



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7:0 | SA[7:0] | 恒定 Alpha<br>可用于计算混合因子。 |

## 23.7.20. 第 x 层默认颜色寄存器(TLI_LxDC)(x = 0, 1)

地址偏移：0x9C+0x80*x x=0 or 1
复位：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | DCA[7:0] | 默认颜色 ALPHA |
| 23:16 | DCR[7:0] | 默认颜色红色 |
| 15:8 | DCG[7:0] | 默认颜色绿色 |
| 7:0 | DCB[7:0] | 默认颜色蓝色 |

**注意**：当该层被禁止或TLI_LxHPOS和TLI_LxVPOS定义的窗口之外，默认颜色值生效。

## 23.7.21. 第 x 层混合寄存器(TLI_LxBLEND)(x = 0, 1)

地址偏移：0xA0+0x80*x x=0 or 1
复位值：0x0000 0607

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | ACF1[2:0] | | | 保留 | | | | | ACF2[2:0] | | |
| | | | | | rw | | | | | | | | | rw | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:11 | 保留 | 必须保持复位值。 |
| 10:8 | ACM1[2:0] | Alpha 混合因子 1 计算方法<br>000：保留<br>001：保留<br>010：保留<br>011：保留<br>100：归一化的恒定 Alpha<br>101：保留<br>110：归一化的像素 Alpha 乘以归一化的恒定 Alpha<br>111：保留 |
| 7:3 | 保留 | 必须保持复位值。 |
| 2:0 | ACM2[2:0] | Alpha 混合因子 2 计算方法<br>000：保留<br>001：保留<br>010：保留<br>011：保留<br>100：保留<br>101：1-归一化的恒定 Alpha<br>110：保留<br>111：1-归一化的像素 Alpha 乘以归一化的恒定 Alpha |

## 23.7.22. 第 x 层帧基地址寄存器(TLI_LxFBADDR)(x = 0, 1)

地址偏移：0xAC+0x80*x x=0 or 1
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

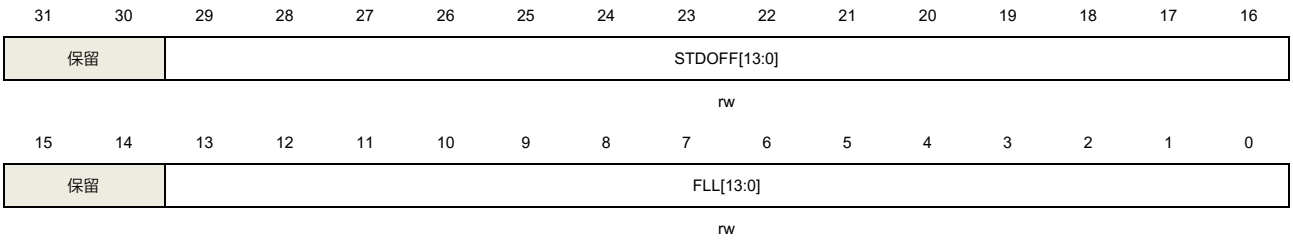
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FBADD[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:0 | FBADD[31:0] | 帧缓冲区基地址<br>帧缓冲区基地址 |

### 23.7.23. 第 x 层行长度寄存器(TLI_LxFLLEN)(x = 0, 1)

地址偏移：0xB0+0x80*x x=0 or 1
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:30 | 保留 | 必须保持复位值。 |
| 29:16 | STDOFF[13:0] | 步幅偏移<br>这个值定义了从某行起始处到下一行起始处之间的字节数 |
| 15:14 | 保留 | 必须保持复位值。 |
| 13:0 | FLL [13:0] | 行长度<br>这个值为一行的字节数+3 |

### 23.7.24. 第 x 层总行数寄存器(TLI_LxFTLN)(x = 0, 1)

地址偏移：0xB4+0x80*x x=0 or 1
复位值：0x0000 0000

该寄存器只能按字(32位)访问。



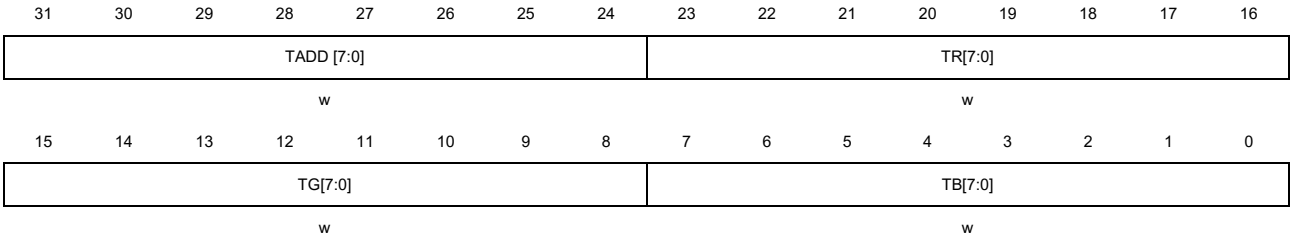

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:11 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 10:0 | FTLN[10:0] | 总行数<br>这个值定义了一帧行数 |

### 23.7.25. 第 x 层颜色查找表寄存器(TLI_LxLUT)(x = 0, 1)

地址偏移：0xC4+0x80*x x=0 or 1
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | TADD[7:0] | 颜色查找表写地址<br>颜色查找表位于该地址的节点的值，将由写入的 TR，TG 和 TB 值更新。 |
| 23:16 | TR[7:0] | LUT 节点的红色值 |
| 15:8 | TG[7:0] | LUT 节点的绿色值 |
| 7:0 | TB[7:0] | LUT 节点的蓝色值 |

# 24. SDIO 接口(SDIO)

## 24.1. 简介

安全的数字输入/输出接口(SDIO)定义了 SD 卡、SD I/O 卡、多媒体卡(MMC)和 CE-ATA 卡主机接口，提供 APB2 系统总线与 SD 存储卡、SD I/O 卡、MMC 和 CE-ATA 设备之间的数据传输。

所支持的 SD 存储卡和 SD I/O 卡系统规格书可以通过 SD 卡协会网站(*www.sdcard.org*)获取。

所支持的多媒体卡(MMC)系统规格书可以通过多媒体卡协会网站(*www.jedec.org*)获取，由 JEDEC 固态技术协会出版。

所支持的 CE-ATA 系统规格书可以通过 CE-ATA 工作组网站(*www.ce-ata.org*)获取。

## 24.2. 主要特性

SDIO 的主要特征如下：

- **MMC：**与多媒体卡系统规格书 V4.2 及之前的版本全兼容。有三种不同的数据总线模式：1 位(默认)、4 位和 8 位；
- **SD 卡：**与 SD 存储卡规格版本 2.0 全兼容；
- **SD I/O：**与 SD I/O 卡规格版本 2.0 全兼容，有两种不同的数据总线模式：1 位(默认)和 4 位；
- **CE-ATA：**与 CE-ATA 数字协议版本 1.1 全兼容；
- 48MHz 数据传输频率和 8 位数据传输模式；
- 中断和 DMA 请求；
- 完成信号使能和失能(CE-ATA)。

**注意：**SDIO 在同一时间仅支持一个 SD、SD I/O、MMC4.2 或 CE-ATA 设备，但可支持多个 MMC4.1 或以前版本的卡。

## 24.3. SDIO 总线拓扑

上电复位之后，主机必须通过特殊的基于消息的总线协议来初始化卡。

每个消息是由以下部分中的一个来表示：

**命令：**命令是启动一个操作的令牌，从主机发送到卡。命令串行传输在 CMD 线上。
**响应：**响应是从卡发送到主机，作为先前接收到的命令的回应。响应串行传输在 CMD 线上。
**数据：**数据可以从卡传输到主机或者从主机传输到卡。数据通过数据线传送。用于数据传输的数据线的数目可以是 1(DAT0)、4(DAT0-DAT3)或 8(DAT0-DAT7)。

命令、响应和数据块的结构在*卡功能描述*章节中介绍。一次数据传输就是一个总线操作。

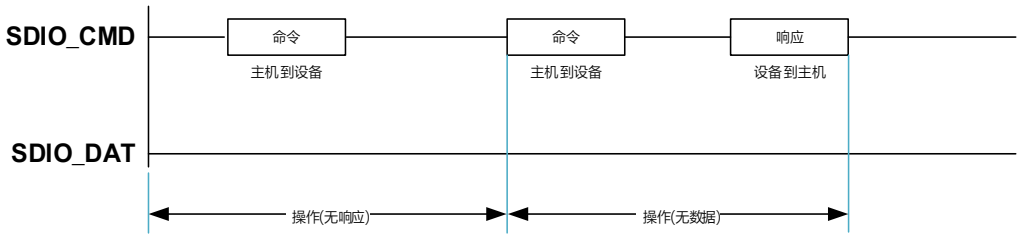
有几种不同类型的操作。一般操作总是包含一个命令和响应。此外，一些操作还有一个数据令牌。还有一些其他操作直接将他们的信息包含在命令或响应结构中。在这种情况下，操作没有数据令牌。在 DAT0-DAT7 和 CMD 信号线上的比特位根据主机时钟同步传输。

两种类型的数据传输命令定义如下：

- 流命令：这些命令发起连续的数据流，只有当 CMD 信号线上出现停止命令时，数据传输终止。该模式将命令的开销减少到最低(仅支持 MMC)。
- 面向块的命令：这些命令成功发送一个数据块后紧跟一个 CRC 校验。读和写操作允许单个或多个块传输。与连续读相同，当 CMD 信号线上出现停止命令时，多块传输终止。

总线上的基本操作是命令/响应操作(参考*图 24-1. SDIO "无响应" 和 "无数据" 操作*)。这种类型的总线事务直接在命令或响应结构中传递它们的信息。此外，有些操作还有数据令牌。卡与设备之间的数据传输通过块完成。

**图 24-1. SDIO "无响应" 和 "无数据" 操作**




多块操作模式比单块操作速度更快。当 CMD 信号线上出现停止命令时，多块传输终止。主机数据传输可以使用单个或多个数据线。多个块的读操作如*图 24-2. SDIO 多块读操作*所示，多个块的写操作如*图 24-3. SDIO 多块写操作*所示。块的写操作在数据(DAT0)信号线上使用忙信号。CE-ATA 设备在准备接收数据之前有一个可选的忙信号。

**图 24-2. SDIO 多块读操作**

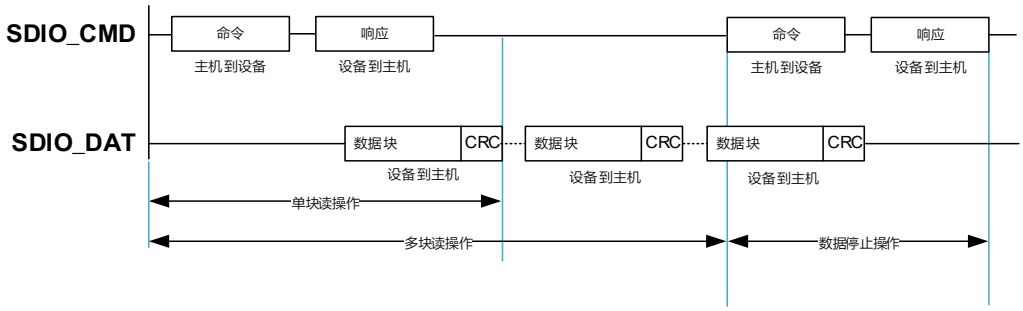
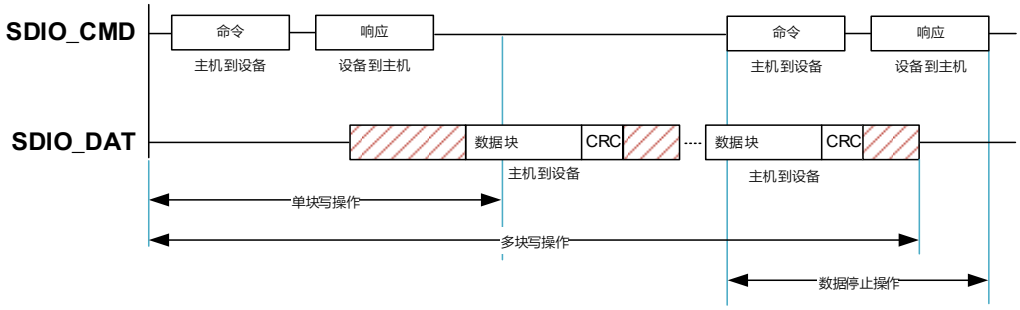


图 24-3. SDIO 多块写操作




SD 存储卡、SD I/O 卡(包括仅 IO 卡和组合卡)和 CE-ATA 设备直接的数据传输是以数据块的方式完成的。MMC 卡以数据块或数据流方式进行数据传输。*图 24-4. SDIO 数据流读操作*和*图 24-5. SDIO 数据流写操作*分别是数据流的读和写操作。

图 24-4. SDIO 数据流读操作

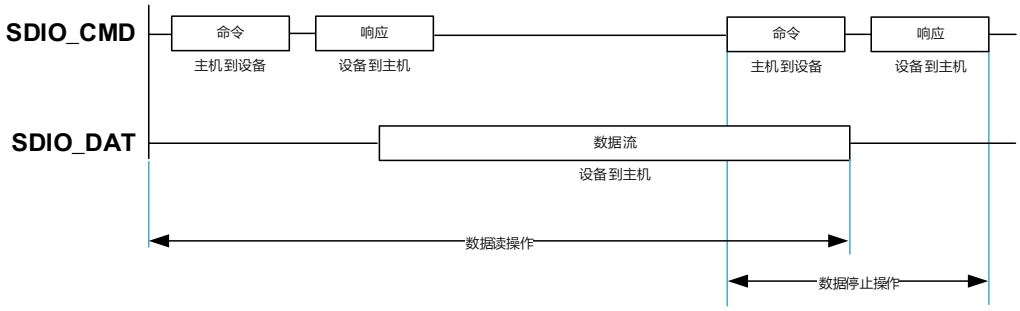
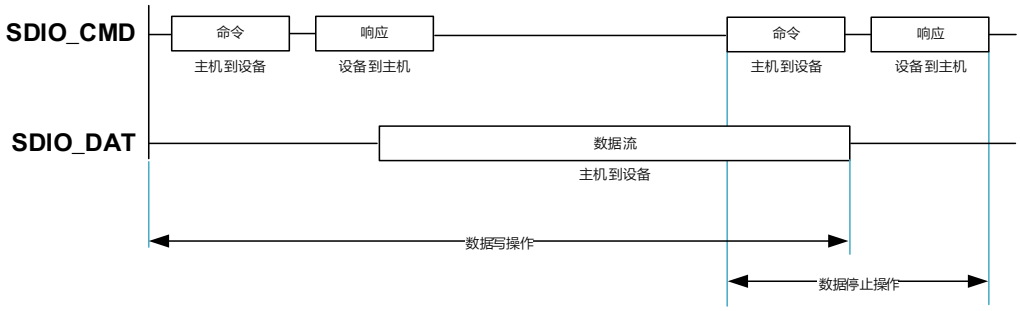



图 24-5. SDIO 数据流写操作

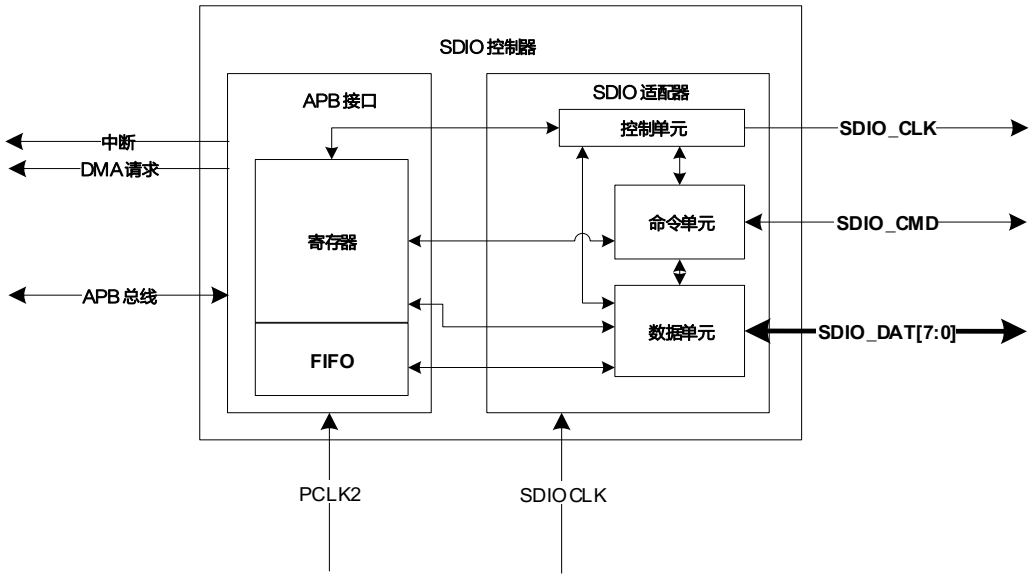



## 24.4. SDIO 功能描述

*图 24-6. SDIO 框图*显示了 SDIO 的结构框图，主要有两大部分：

- SDIO 适配器：由控制单元、命令单元和数据单元组成。控制单元管理时钟信号，命令单元管理命令的传输，数据单元管理数据的传输。
- APB 接口：包括通过 APB2 总线访问的寄存器、用于数据传输的 FIFO 单元以及产生中断和 DMA 请求信号。

图 24-6. SDIO 框图




### 24.4.1. SDIO 适配器

SDIO 适配器包括控制单元、命令单元和数据单元，并且可以向卡生成信号。这些信号的具体描述如下：

**SDIO_CLK**：SDIO 控制器提供给卡的时钟。每个时钟周期在命令线(SDIO_CMD)和所有的数据线(SDIO_DAT)上直接发送一位命令或数据。对于 MMC 卡 V3.31 版本，SDIO_CLK 频率可以在 0 MHz 到 20 MHz 之间，对于 MMC 卡 V4.2 版本可以在 0 MHz 到 48MHz 之间，对于 SD 或 SD I/O 卡可以在 0 MHz 到 25 MHz。
SDIO 使用两个时钟信号：SDIO 适配器时钟(SDIOCLK <= 48MHz)和 APB2 总线时钟(PCLK2)。PCLK2 时钟频率应不低于 SDIO_CLK 的 3/8 倍。

**SDIO_CMD**：该信号是双向命令通道，用于卡的初始化和命令的传输。命令从 SDIO 控制器发送到卡，响应从卡发送到主机。CMD 信号有两种操作模式：用于初始化的开漏模式(仅用于 MMC 卡 V3.31 及之前版本)和用于命令传送的推挽模式(SD 卡/SD I/O 卡和 MMC 卡 4.2 版本初始化时也是用推挽模式)。

**SDIO_DAT[7:0]**：这些信号线都是双向数据通道。数据信号线操作在推挽模式。每次只有卡或者主机会驱动这些信号。默认情况下，上电或者复位后仅 DAT0 用于数据传输。SDIO 适配器可以配置更宽的数据总线用于数据传输，使用 DAT0-DAT3 或者 DAT0-DAT7(仅适用于 MMC V4.2)。SDIO 对数据信号线 DAT1-DAT7 有内部上拉。在进入 4 位模式后，卡断开 DAT1 和 DAT2 的内部上拉(DAT3 内部上拉保持不变是由于 SPI 模式下 CS 片选的使用)。相应地，在进入 8 位模式后，断开 DAT1，DAT2 和 DAT4-DAT7 的内部上拉。

表 24-1. SDIO I/O 定义

| 引脚名称 | 信号 | 方向 | 描述 |
|---|---|---|---|
| SDIO_CK | SDIO_CLK | O | SD/SD I/O /MMC 时钟 |
| SDIO_CMD | SDIO_CMD | I/O | 命令的输入/输出 |

| 引脚名称 | 信号 | 方向 | 描述 |
|---|---|---|---|
| SDIO_D[7:0] | SDIO_DAT[7:0] | I/O | 数据线 DAT[7:0]的数据输入/输出 |

SDIO 适配器是 SD/SD I/O /MMC/CE-ATA 的接口，它由 3 个子单元组成：

### 控制单元

控制单元包含电源管理功能和时钟管理功能用于存储卡时钟。电源管理是由 SDIO_PWRCTL 寄存器控制的，实现电源的掉电和上电。通过设置 SDIO_CLKCTL 的 CLKPWRSAV 位来配置省电模式，实现当总线空闲时，关闭 SDIO_CLK。时钟管理向卡生成 SDIO_CLK 时钟信号。当 SDIO_CLKCTL 寄存器的 CLKBYP 位为 0 时，SDIO_CLK 由 SDIOCLK 分频得到；当 SDIO_CLKCTL 寄存器的 CLKBYP 位为 1 时，SDIO_CLK 直接为 SDIOCLK。

通过设置 SDIO_CLKCTL 寄存器的 HWCLKEN 位使能硬件时钟控制。该功能用于避免 FIFO 下溢和上溢错误，硬件根据系统总线是否繁忙，控制 SDIO_CLK 的开关。当 FIFO 不能接收或发送数据，主机将会关闭 SDIO_CLK 并冻结 SDIO 状态机来避免相关错误。只有状态机能被冻结，但 APB2 接口仍在工作。所以，FIFO 可以通过 APB2 总线访问。

### 命令单元

命令单元实现向卡发送和接收命令。数据传输流由命令状态机(CSM)控制。在对 SDIO_CMDCTL 寄存器进行一次写操作并设置该寄存器的 CSMEN 位为 1 后，命令传输开始。首先向卡发送一个命令，这个命令包含 48 位，通过 SDIO_CMD 线发出，每个 SDIO_CLK 发送一个比特数据。这 48 位命令包含 1 位起始位、1 位传输位、6 位命令索引(由 SDIO_CMDCTL 寄存器的 CMDIDX 位定义)、32 位参数(由 SDIO_CMDAGMT 定义)、7 位 CRC 和 1 位停止位。然后接收来自卡的响应(在 SDIO_CMDCTL 寄存器的 CMDIDX 位不为 0b00 或 0b10 的情况下)，响应分为 48 位的短响应和 136 位的长响应，响应都存在 SDIO_RESP0 - SDIO_RESP3 寄存器中。命令单元同样可以产生命令状态标志(在 SDIO_STAT 寄存器中定义)。

### 命令状态机

| CS_Idle | 复位后准备发送命令 | |
|---|---|---|
| 1.CSM 被使能并且 WAITDEND 使能 | -> | CS_Pend |
| 2.CSM 被使能并且 WAITDEND 失能 | -> | CS_Send |
| 3.CSM 被关闭 | -> | CS_Idle |
| **注意：**命令状态机在空闲状态至少保持 8 个 SDIO_CLK 周期，以满足 $N_{CC}$ 和 $N_{RC}$ 时序限制。$N_{CC}$ 是两个主机命令之间的最小时间间隔，$N_{RC}$ 是主机命令与卡响应之间的最小时间间隔。 | | |

| CS_Pend | 等待数据传输结束 | |
|---|---|---|
| 1.数据传送完成 | -> | CS_Send |
| 2.CSM 被关闭 | -> | CS_Idle |

| CS_Send | 发送命令 |
|---|---|

| | | |
|---|---|---|
| 1.命令发送后有响应 | -> | CS_Wait |
| 2.命令发送后无响应 | -> | CS_Idle |
| 3.CSM 被关闭 | -> | CS_Idle |

| CS_Wait | 等待响应起始位 | |
|---|---|---|
| 1.接收到响应(检测到起始位) | -> | CS_Receive |
| 2.接收响应超时 | -> | CS_Idle |
| 3.CSM 被关闭 | -> | CS_Idle |
| **注意：**命令超时时间固定为 64 个 SDIO_CLK 时钟周期。 | | |

| CS_Receive | 接收响应并检测 CRC | |
|---|---|---|
| 1.在 CE-ATA 模式下收到响应，失能 CE-ATA 中断并且等待 CE-ATA 设备命令完成信号使能 | -> | CS_Waitcompl |
| 2.在 CE-ATA 模式下收到响应，失能 CE-ATA 中断并且等待 CE-ATA 设备命令完成信号失能 | -> | CS_Pend |
| 3.CSM 被关闭 | -> | CS_Idle |
| 4.收到响应 | -> | CS_Idle |
| 5.命令 CRC 检测失败 | -> | CS_Idle |

| CS_Waitcompl | 等待 CE-ATA 设备命令完成信号 | |
|---|---|---|
| 1.收到 CE-ATA 命令完成信号 | -> | CS_Idle |
| 2.CSM 被关闭 | -> | CS_Idle |
| 3.命令 CRC 检测失败 | -> | CS_Idle |

### 数据单元

数据单元实现主机与卡之间的数据传输。当数据宽度为 8 位(SDIO_CLKCTL 寄存器的 BUSMODE 位为 0b10)时，数据传输使用 SDIO_DAT[7:0]信号线；当数据宽度为 4 位(SDIO_CLKCTL 寄存器的 BUSMODE 位为 0b01)时，数据传输使用 SDIO_DAT[3:0]信号线；当数据宽度为 1 位(SDIO_CLKCTL 寄存器的 BUSMODE 位为 0b00)时，数据传输使用 SDIO_DAT[0]信号线。数据传输流由数据状态机(DSM)控制。在对 SDIO_DATACTL 寄存器进行一次写操作并将 SDIO_DATACTL 寄存器的 DATAEN 位为 1，数据传输开始。当 SDIO_DATACTL 寄存器的 DATADIR 位为 0 时，数据是从控制器到卡；当 DATADIR 位为 1 时，数据是从卡到控制器。数据单元同样可以产生数据状态标志(在 SDIO_STAT 寄存器中定义)。

### 数据状态机

| DS_Idle | 数据单元不工作，等待发送和接收数据 | |
|---|---|---|
| 1.DSM 使能并且数据传输方向为主机到卡 | -> | DS_WaitS |
| 2.DSM 使能并且数据传输方向为卡到主机 | -> | DS_WaitR |
| 3.DSM 使能并且读等待已经开始并且使能 SD I/O 模 | -> | DS_Readwait |

| | | |
|---|---|---|
| 式 | | |

| DS_WaitS | 等待数据 FIFO 为空标志无效或者数据传输结束 | |
|---|---|---|
| 1.数据传输结束 | -> | DS_Idle |
| 2.DSM 被关闭 | -> | DS_Idle |
| 3.数据 FIFO 为空标志无效 | -> | DS_Send |

| DS_Send | 发送数据到卡 | |
|---|---|---|
| 1.数据块已发送 | -> | DS_Busy |
| 2.DSM 被关闭 | -> | DS_Idle |
| 3.数据 FIFO 下溢错误发生 | -> | DS_Idle |
| 4.内部 CRC 错误 | -> | DS_Idle |

| DS_Busy | 等待 CRC 状态标志 | |
|---|---|---|
| 1.接收到正确 CRC 状态并且卡不繁忙 | -> | DS_WaitS |
| 2.没有接收到正确 CRC 状态 | -> | DS_Idle |
| 3.DSM 被关闭 | -> | DS_Idle |
| 4.数据超时发生 | -> | DS_Idle |
| **注意：** 命令超时时间设置在数据超时寄存器(SDIO_DATATO)中。 | | |

| DS_WaitR | 等待接收数据的起始位 | |
|---|---|---|
| 1.数据接收结束 | -> | DS_Idle |
| 2.DSM 被关闭 | -> | DS_Idle |
| 3.数据超时 | -> | DS_Idle |
| 4.在超时前收到起始位 | -> | DS_Receive |
| **注意：** 命令超时时间设置在数据超时寄存器(SDIO_DATATO)中。 | | |

| DS_Receive | 接收卡的数据并将其写入数据 FIFO | |
|---|---|---|
| 1.数据块已接收 | -> | DS_WaitR |
| 2.数据传输结束 | -> | DS_WaitR |
| 3.数据 FIFO 下溢发送 | -> | DS_Idle |
| 4.数据已经接收并且读等待开始并且使能 SD I/O 模式 | -> | DS_Readwait |
| 5.DSM 被关闭或 CRC 错误 | -> | DS_Idle |

| DS_Readwait | 等待"读等待停止"指令 | |
|---|---|---|
| 1."读等待停止"使能 | -> | DS_WaitR |
| 2.DSM 被关闭 | -> | DS_Idle |

### 24.4.2. APB2 接口

APB2 接口实现了访问 SDIO 寄存器、数据 FIFO 和生成中断和 DMA 请求。它包括数据 FIFO 单元、寄存器单元和中断/DMA 逻辑。

至少一个已经被选中的状态标志为高时，中断逻辑产生中断。中断使能寄存器允许中断逻辑产生相应的中断。

DMA 接口提供一种方法，可以快速地在 SDIO 数据 FIFO 和存储器直接进行数据传输。下面的例子描述了如何实现这种方法：

1. 完成卡识别的过程。

2. 提高 SDIO_CLK 时钟频率。

3. 发送 CMD7 用于选择卡并配置总线宽度。

4. DMA1 的配置过程如下：
打开 DMA1 控制器并清除任何中断标志。用存储器基地址来配置 DMA1 通道 3 或通道 6 外设请求 4 的源地址寄存器，用 SDIO_FIFO 寄存器的地址来配置 DMA1 通道 3 或通道 6 外设请求 4 的目的地址寄存器。配置 DMA1 通道 3 或通道 6 外设请求 4 的控制寄存器(存储器地址指针递增，外设地址指针固定，存储器和外设的数据宽度为字)。对于外设端，在 DMA1 通道 3 或通道 6 外设请求 4 中，设置突发传输为 4 拍。

5. 写数据块(CMD24)到卡的过程如下：
以字节的形式将数据大小写入到 SDIO_DATALEN 寄存器中。以字节的形式将块大小(BLKSZ)写入到 SDIO_DATACTL 寄存器中，然后主机以每个块 BLKSZ 大小发送数据。向 SDIO_CMDAGMT 中写入数据的地址，该地址是卡中需要传输的数据地址。配置 SDIO 命令控制寄存器(SDIO_CMDCTL)：CMDIDX 置为 24，CMDRESP 置为 1(SDIO 卡主机等待短响应)，CSMEN 置为 1(发送命令使能)。其他字段为其复位值。

当 CMDRECV 标志被置位，配置 SDIO 数据控制寄存器(SDIO_DATACTL)：DATAEN 置为 1(发送数据使能)，DATADIR 置为 0(传输方向从控制器到卡)，TRANSMOD 置为 0(块传输)，DMAEN 置为 1(DMA 使能)，BLKSZ 置为 0x9(512 字节)。其他字段不用设置。

等待 DTBLKEND 标志位置位。通过轮询 DMA 中断标志寄存器，检查没有通道处于使能状态。

它还包括下面两个子单元：

#### 寄存器单元

寄存器单元包含所有的系统寄存器，生成信号用于控制卡与控制器之间的通信。

#### 数据 FIFO

数据 FIFO 单元有一个数据缓冲区，用于发送和接收 FIFO。FIFO 包含一个每个字的宽度为 32 位，深度为 32 字的数据缓冲区。发送 FIFO 被用在当需要写数据到卡上并且 SDIO_STAT 寄存器的 TXRUN 位为 1 时。待传输的数据通过 APB2 总线写入到发送 FIFO 中，SDIO 适配器

中的数据单元从发送 FIFO 中读取数据，然后发送到卡上。接收 FIFO 被用在当需要从卡中读取数据并且 SDIO_STAT 寄存器的 RXRUN 为 1 时。从卡读取数据，然后将待传输的数据写入到接收 FIFO。在需要的时候，通过 APB2 总线读取接收 FIFO 中的数据。这个单元同样可以生成不同的 FIFO 标志(在 SDIO_STAT 寄存器中定义)。

## 24.5. 卡功能描述

### 24.5.1. 卡寄存器

卡内部定义了接口寄存器：OCR，CID，CSD，EXT_CSD，RCA，DSR 和 SCR。这些寄存器只能通过相应的命令来访问。OCR，CID，CSD 和 SCR 寄存器包含卡的一些特定信息，而 RCA 和 DSR 寄存器是配置寄存器，存储实际的配置参数。EXT_CSD 寄存器同时包含卡的特定信息和实际的结构参数。有关具体信息，请参考相关的规范。

**OCR 寄存器**：32 位操作条件寄存器(OCR)储存卡的 $V_{DD}$ 电压描述和存取模式指示(MMC)。另外，该寄存器包括一个状态信息位。如果卡上电过程已经完成该状态位被置位。该寄存器在 MMC 和 SD 卡之间有一点不同。主机可以使用 CMD1(MMC)，ACMD41(SD 存储卡)，CMD5(SD I/O)来获取该寄存器的内容。

**CID 寄存器：**卡识别寄存器(CID)是 128 位宽。它包含在卡识别阶段使用的卡识别信息。每个读/写(RW)卡应具有唯一的标识号。主机可以使用 CMD2 和 CMD10 得到这个寄存器的内容。

**CSD 寄存器：**卡特定数据寄存器提供访问卡中的内容信息。CSD 定义了数据格式、错误校正类型、最大数据访问时间、数据传输速度、DSR 寄存器是否可以使用等。寄存器的可编程部分可通过 CMD27 来修改。主机可以使用 CMD9 得到这个寄存器的内容。

**扩展 CSD 寄存器：**只有 MMC4.2 有该寄存器。扩展 CSD 寄存器定义卡属性和选择模式。它的长度为 512 字节。最高 320 字节为属性段，定义了卡的功能，并且不能由主机修改。最低 192 字节是模式段，定义了卡工作在哪种配置下。这些模式可以由主机通过 SWITCH 命令来修改。主机可以使用 CMD8(仅 MMC 支持这个命令)，以获取该寄存器的内容。

**RCA 寄存器**：可写的 16 位相对卡地址寄存器存放卡地址，该地址在卡的初始化期间由卡向外发布。这个地址用于卡识别过程之后，所寻址的主机和卡通信。主机可以使用 CMD3 要求卡发布一个新的相对地址(RCA)。

**注意**：RCA 的寄存器的缺省值是 0x0001(MMC)或 0x0000(SD/SD I/O)。这个数值是保留值，用于通过 CMD7 设置所有卡到待机(Stand-by)状态。

**DSR 寄存器 (可选)**：16 位驱动阶段寄存器是可选的，可用于在扩展操作条件中提高总线性能(取决于类似于总线长度，传输速率和卡数目这些参数)。CSD 寄存器中有 DSR 寄存器使用情况的信息。DSR 寄存器的默认值是 0x404。主机可以使用 CMD4 得到这个寄存器的内容。

**SCR 寄存器：**仅 SD/ SD I/O(如果有存储模块)有这个寄存器。除了 CSD 寄存器，除了 CSD 寄存器，还有另一种配置寄存器名为 SD 卡配置寄存器(SCR)，它仅用于 SD 卡。SCR 提供了被配置到特定 SD 存储卡的特殊功能的信息。SCR 寄存器的大小是 64 位。该寄存器应在出厂前通过 SD 存储卡制造商进行设置。主机可以使用 ACMD51 得到这个寄存器的内容。

## 24.5.2. 命令

### 命令类型

有四种控制卡的命令：

- 广播命令(bc)，发送到所有卡，没有响应；
- 带响应的广播命令(bcr)，发送到所有卡，同时从所有卡收到响应；
- 寻址(点对点)命令(ac)，发送到寻址的卡上，DAT 信号线没有数据传输；
- 寻址(点对点)的数据传输的命令(adtc)，发送到寻址的卡上，DAT 信号线进行数据传输。

### 命令格式

所有命令都是 48 位的固定码长，如*图 24-7. 命令标记格式*所示，需要 1.92 us(25 MHz)0.96 us(50 MHz)和 0.92us(52 MHz)的发送时间。

**图 24-7. 命令标记格式**

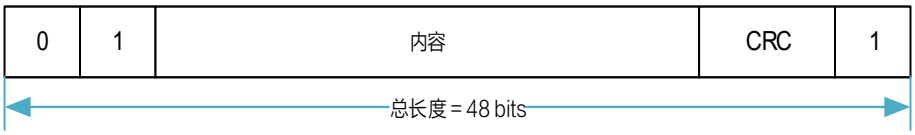



**表 24-2. 命令格式**

| 位 | 47 | 46 | [45:40] | [39:8] | [7:1] | 0 |
|---|---|---|---|---|---|---|
| 宽度 | 1 | 1 | 6 | 32 | 7 | 1 |
| 数值 | '0' | '1' | x | x | x | '1' |
| 描述 | 起始位 | 传输位 | 命令索引 | 参数 | CRC7 | 结束位 |

一个命令总是从一个起始位(始终为 0)开始，随后的位表示传输的方向(主机=1)。接下来的 6 位表示命令的索引，该值被解释为一个二进制编码的数字(0 到 63 之间)。一些命令需要一个参数(例如，一个地址)，由 32 位编码。上面表中的表示为"x"的值表示这个变量依赖于该命令。所有的命令有一个 CRC 7 位校验，由结束位(总是 1)终止。

### 命令分类

卡的命令集分为几类(见*表 24-3. 卡命令类 (CCCs)*)。每类支持一组卡的功能。*表 24-4. 基本命令(class 0)*根据卡支持的命令来决定 CCC 的设置。

对于 SD 卡，类别为 0，2，4，5 和 8 的命令是强制的，应被 SD 卡支持。类别 7 中除了 CMD40 以外都是强制性用于 SDHC。其他类是可选的。所支持的卡命令类(CCC)被编码为参数，设置在每个卡的卡特定数据(CSD)寄存器，提供给主机如何访问该卡信息。

对于 MMC 卡，类别为 0 的命令是强制性的，应被 MMC 卡支持。其他类只对特定类型的卡是强制或是可选的。通过使用不同的类，可以选择几种配置(例如，一个块可写的卡或流可读的卡)。所支持的卡命令类(CCC)被编码为参数，设置在每个卡的卡的特定数据(CSD)寄存器，提

供给主机如何访问该卡信息。

对于 CE-ATA 设备，设备必须支持 MMC 命令，这些命令需要在设备初始化阶段完成传输状态。其它接口配置的设置，如总线宽度，可能需要额外的 MMC 命令来支持，具体请参考 MMC 引用。CE-ATA 利用以下的 MMC 命令：CMD0 - GO_IDLE_STATE，CMD12 - STOP_TRANSMISSION，CMD39 - FAST_IO，CMD60 - RW_MULTIPLE_REGISTER，CMD61 - RW_MULTIPLE_BLOCK。GO_IDLE_STATE(CMD0)，STOP_TRANSMISSION(CMD12)和 FAST_IO(CMD39) 由 MMC 引用定义。 RW_MULTIPLE_REGISTER(CMD60) 和 RW_MULTIPLE_BLOCK(CMD61)是 CE-ATA 协议定义的 MMC 命令。

**表 24-3. 卡命令类 (CCCs)**

| | 卡命令类 (CCC) | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 支持的命令 | 类描述 | basic | Stream read | Block read | Stream write | Block write | erase | write protection | Lock card | application specific | I/O mode | switch | reserved |
| CMD0 | M | + | | | | | | | | | | | |
| CMD1 | M | + | | | | | | | | | | | |
| CMD2 | M | + | | | | | | | | | | | |
| CMD3 | M | + | | | | | | | | | | | |
| CMD4 | M | + | | | | | | | | | | | |
| CMD5 | O | | | | | | | | | | + | | |
| CMD6 | M | | | | | | | | | | | + | |
| CMD7 | M | + | | | | | | | | | | | |
| CMD8 | M | + | | | | | | | | | | | |
| CMD9 | M | + | | | | | | | | | | | |
| CMD10 | M | + | | | | | | | | | | | |
| CMD11 | M | | + | | | | | | | | | | |
| CMD12 | M | + | | | | | | | | | | | |
| CMD13 | M | + | | | | | | | | | | | |
| CMD14 | M | + | | | | | | | | | | | |
| CMD15 | M | + | | | | | | | | | | | |
| CMD16 | M | | | + | | + | | | + | | | | |
| CMD17 | M | | | + | | | | | | | | | |
| CMD18 | M | | | + | | | | | | | | | |
| CMD19 | M | + | | | | | | | | | | | |
| CMD20 | M | | | | + | | | | | | | | |
| CMD23 | M | | | + | | + | | | | | | | |
| CMD24 | M | | | | | + | | | | | | | |
| CMD25 | M | | | | | + | | | | | | | |
| CMD26 | M | | | | | + | | | | | | | |

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **CMD27** | M | | | | | + | | | | | | | |
| **CMD28** | M | | | | | | | + | | | | | |
| **CMD29** | M | | | | | | | + | | | | | |
| **CMD30** | M | | | | | | | + | | | | | |
| **CMD32** | M | | | | | | + | | | | | | |
| **CMD33** | M | | | | | | + | | | | | | |
| **CMD34** | O | | | | | | | | | | | + | |
| **CMD35** | O | | | | | | | | | | | + | |
| **CMD36** | O | | | | | | | | | | | + | |
| **CMD37** | O | | | | | | | | | | | + | |
| **CMD38** | M | | | | | | + | | | | | | |
| **CMD39** | | | | | | | | | | | + | | |
| **CMD40** | | | | | | | | | | | + | | |
| **CMD42** | | | | | | | | | + | | | | |
| **CMD50** | O | | | | | | | | | | | + | |
| **CMD52** | O | | | | | | | | | | + | | |
| **CMD53** | O | | | | | | | | | | + | | |
| **CMD55** | M | | | | | | | | | + | | | |
| **CMD56** | M | | | | | | | | | + | | | |
| **CMD57** | O | | | | | | | | | | | + | |
| **CMD60** | M | | | | | | | | | + | | | |
| **CMD61** | M | | | | | | | | | + | | | |
| **ACMD6** | M | | | | | | | | | + | | | |
| **ACMD13** | M | | | | | | | | | + | | | |
| **ACMD22** | M | | | | | | | | | + | | | |
| **ACMD23** | M | | | | | | | | | + | | | |
| **ACMD41** | M | | | | | | | | | + | | | |
| **ACMD42** | M | | | | | | | | | + | | | |
| **ACMD51** | M | | | | | | | | | + | | | |

**注意：**1. CMD1, CMD11, CMD14, CMD19, CMD20, CMD23, CMD26, CMD39 和 CMD40 仅用于 MMC 卡。CMD5, CMD32-34, CMD50, CMD52, CMD53, CMD57 和 ACMDx 仅用于 SD 存储卡。CMD60,CMD61 仅用于 CE-ATA 设备。

2. 在使用 ACMD 命令之前发送 APP_CMD 命令(CMD55)。

3. CMD8 对于 MMC 卡和 SD 卡有不同的含义。

## 详细的命令描述

下列表详细描述了所有的总线命令。响应 R1-R7 将在*响应*章节说明。寄存器 CID，CSD 和 DSR 在*卡功能描述*介绍。卡应忽略参数中填充位和保留位。

表 24-4. 基本命令(class 0)

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| 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 线发送响应(包含操作条件寄存器的内容)。 |
| CMD2 | bcr | [31:0] 填充位 | R2 | ALL_SEND_CID | 请求任何卡通过CMD线发送发送CID 数据(任何连接到主机的卡都会响应)。 |
| 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<br>[15:0] 填充位 | - | SET_DSR | 设置所有卡的 DSR 寄存器。 |
| CMD5 | bcr | [31:25]保留位<br>[24]S18R<br>[23:0] I/O OCR | R4 | IO_SEND_OP_COND | 仅适用于 I/O 卡。它类似于用于 SD 存储卡的 ACMD41 命令，用于查询所需要的 I/O 卡的电压范围。 |
| CMD6 | ac | [31:26] 设为 0<br>[25:24] 访问<br>[23:16] 索引<br>[15:8] 值<br>[7:3] 设为 0<br>[2:0] 命令集 | R1b | SWITCH | 仅适用于 MMC 卡。切换所选卡的操作模式，或修改 EXT_CSD 寄存器。 |
| CMD7 | ac | [31:16] RCA<br>[15:0] 填充位 | R1b | SELECT/DESELECT_CARD | 这个命令用于卡在待机(stand-by)状态和发送(transfer)状态之间切换，或编程(programming)状态和断开(disconnects)状态之间切换。在两种情况下，要选中该卡用它自己的相对地址，若不选中该卡用任何其他地址。地址 0 用于取消选择该卡。 |
| CMD8 | bcr | [31:12]保留位<br>[11:8]工作电压(VHS)<br>[7:0]检查模式 | R7 | SEND_IF_COND | 向 SD 存储卡发送接口条件，包括主机供电电压信息和询问卡是否支持电压。保留位应设为 0。 |
| CMD8 | 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<br>[15:0] 填充位 | R2 | SEND_CSD | 被选定的卡通过CMD线发送它的卡特定数据(CSD)。 |
| CMD10 | ac | [31:16] RCA<br>[15:0] 填充位 | R2 | SEND_CID | 被选定的卡通过CMD线发送它的卡标识(CID)。 |
| CMD12 | ac | [31:0] 填充位 | R1b | STOP TRANSMISSION | 强制卡停止传输。 |

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| CMD13 | ac | [31:16] RCA<br>[15:0] 填充位 | R1 | SEND_STATUS | 被选定的卡发送它的状态寄存器。 |
| CMD14 | adtc | [31:0] 填充位 | R1 | BUSTEST_R | 主机从卡中读取反向的总线测试数据模式。 |
| CMD15 | ac | [31:16] RCA<br>[15:0] 保留位 | - | GO_INACTIVE_STATE | 将被选定的卡转换到非激活(Inactive)状态。这个命令被用于当主机明确地想停用一张卡的时候。 |
| CMD19 | adtc | [31:0] 填充位 | R1 | BUSTEST_W | 主机向卡发送总线测试模式。 |

**表 24-5. 面向块的读命令(class 2)**

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| CMD16 | ac | [31:0]块长度 | R1 | SET_BLOCKLEN | 在标准容量 SD 卡和 MMC 卡的情况下，该命令为所有后续块命令(读，写，锁)设置块长度(以字节为单位)。默认值是 512 字节。只有在 CSD 中局部块读操作被允许时，设置长度对于存储器访问命令有效。<br>在高容量 SD 存储卡的情况下，块长度是由 CMD16 命令设置，不会影响内存读和写命令。总是使用 512 字节的固定块长度。在这两种情况下，如果块长度设置大于 512 字节，BLOCK_LEN_ERROR 位会被卡置位。 |
| CMD17 | adtc | [31:0]数据地址 | R1 | READ_SINGLE_BLOCK | 在标准容量 SD 卡和 MMC 卡的情况下，通过 SET_BLOCKLEN 命令读取所选择大小的块。<br>在高容量存储卡的情况下，块长度是固定的 512 字节，忽略 SET_BLOCKLEN 命令。 |
| CMD18 | adtc | [31:0]数据地址 | R1 | READ_MULTIPLE_BLOCK | 不断从卡传输数据块到主机，直到收到 STOP_TRANSMISSION 命令才中断。块长度规定和 READ_SINGLE_BLOCK 命令是一样的。 |
| **注意：** 传输的数据不能跨越物理块边界，除非 READ_BLK_MISALIGN 在 CSD 寄存器中被设置。 | | | | | |

表 24-6. 流读取命令(class 1)和流写入命令(class 3)

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| CMD11 | adtc | [31:0]数据地址 | R1 | READ_DAT_UNTIL_STOP | 从卡中读取数据流，起始于给定的地址，直至收到 STOP_TRANSMISSION 命令。 |
| CMD20 | adtc | [31:0]数据地址 | R1 | WRITE_DAT_UNTIL_STOP | 从主机写数据流，起始于给定的地址，直至收到 STOP_TRANSMISSION 命令。 |
| **注意：** 传输的数据不能跨越物理块边界，除非 READ_BLK_MISALIGN 在 CSD 寄存器中被设置。 | | | | | |

表 24-7. 面向块的写命令(class 4)

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| CMD16 | ac | [31:0] 块长度 | R1 | SET_BLOCKLEN | 见 *表 24-5. 面向块的读命令(class 2)*描述。 |
| CMD23 | ac | [31:16] 设为 0<br>[15:0] 块数目 | R1 | SET_BLOCK_COUNT | 定义了将要在后续多个块的读或写命令被传输块的数目。如果参数为全 0，随后的读/写操作将被被认为无终止的。 |
| CMD24 | adtc | [31:0] 数据地址 | R1 | WRITE_BLOCK | 在标准容量 SD 卡的情况下，该命令写入由 SET_BLOCKLEN 命令所选择的块长度。在高容量 SD 卡的情况下，块长度是固定的 512 字节忽略 SET_BLOCKLEN 命令。 |
| CMD25 | adtc | [31:0] 数据地址 | R1 | WRITE_MULTIPLE_BLOCK | 连续写入数据块，直至收到 STOP_TRANSMISSION 命令。块长度是和 WRITE_BLOCK 命令规定一样的。 |
| CMD26 | adtc | [31:0] 填充位 | R1 | PROGRAM_CID | 对卡识别寄存器进行编程。此命令必须一次发出。该编程涉及硬件，以防止首次编程以后的操作。通常情况下这个命令是针对厂家保留。 |
| CMD27 | adtc | [31:0] 填充位 | R1 | PROGRAM_CSD | 对 CSD 的可编程位编程。 |
| **注意：** 1. 传输的数据不得跨越物理块边界。除非是在 CSD 设置 WRITE_BLK_MISALIGN。在写入部分块不支持的情况下，块长度=默认块长度(CSD 中给出)。<br>2. 标准容量 SD 存储卡数据地址以字节为单位，高容量 SD 存储卡数据地址以块(512 字节)为单位。 | | | | | |

表 24-8. 擦除命令(class 5)

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| CMD32 | ac | [31:0]数据地址 | R1 | ERASE_WR_BLK_START | 设置要被擦除数据的第一个块的地址。(SD) |
| CMD33 | ac | [31:0]数据地址 | R1 | ERASE_WR_BLK_EN | 设置要被擦除数据的最后一个块 |

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| | | | | D | 地址。(SD) |
| 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。(MMC) |
| 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。(MMC) |
| CMD38 | ac | [31:0]填充位 | R1b | ERASE | 擦除所有之前选择的数据块. |
| **注意：** 1. CMD34 和 CMD37 被保留，以便保持与旧版本 MMC 的兼容性<br>2. 标准容量 SD 存储卡数据地址以字节为单位，高容量 SD 存储卡数据地址以块(512 字节)为单位。 | | | | | |

**表 24-9. 面向块的写保护命令(class 6)**

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| CMD28 | ac | [31:0] 数据地址 | R1b | SET_WRITE_PROT | 如果卡有写保护功能，该命令将设置地址组的写保护位。写保护的特性被编码在卡的特定数据(WP_GRP_SIZE)中。高容量 SD 存储卡不支持此命令。 |
| CMD29 | ac | [31:0] 数据地址 | R1b | CLR_WRITE_PROT | 如果卡有写保护功能，该命令将清除寻址组的写保护位。 |
| CMD30 | adtc | [31:0] 写保护数据地址 | R1 | SEND_WRITE_PROT | 如果卡有写保护功能，该命令请求卡发送写保护位状态。 |
| **注意：** 1. 高容量 SD 存储卡不支持这三个命令。 | | | | | |

**表 24-10. 锁卡命令(class 7)**

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| CMD16 | ac | [31:0] 块长度 | R1 | SET_BLOCK_LEN | 见*表 24-5. 面向块的读命令(class 2)*描述。 |
| CMD42 | adtc | [31:0] 保留位(所有位设为 0) | R1 | LOCK_UNLOCK | 用于设置/重置密码或者对卡上锁/解锁。数据块长度由命令 SET_BLOCK_LEN 设置。<br>参数及锁卡数据结构里的保留位应设为 0。 |

**表 24-11. 特定应用命令(class 8)**

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| ACMD41 | bcr | [31]保留位<br>[30]HCS<br>[29:24]保留位<br>[23:0]$V_{DD}$ 电压窗口(OCR[23:0]) | R3 | SD_SEND_OP_COND | 发送给主机容量支持信息(HCS)，并请求访问的卡在响应中发送操作条件寄存器(OCR)的内容。当卡接收到 SEND_IF_COND 命令，HCS 是有效的。CCS 位被分配到 |

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| | | | | | OCR[30]。 |
| ACMD42 | ac | [31:1] 填充位<br>[0] set_cd | R1 | SET_CLR_CARD_DETECT | 在卡的 CD/DAT3 (引脚 1)上连接[1]/断开[0] 50K 上拉电阻。 |
| ACMD51 | adtc | [31:0] 填充位 | R1 | SEND_SCR | 读 SD 卡配置寄存器(SCR)。 |
| CMD55 | ac | [31:16] RCA<br>[15:0] 填充位 | R1 | APP_CMD | 表明卡的下一个命令是特定应用命令而不是标准命令。 |
| CMD56 | adtc | [31:1] 填充位<br>[0] RD/WR | R1 | GEN_CMD | 对于通用/特定应用命令，该命令用于向卡传输一个数据块，或从卡读取一个数据块。主机设 RD/WR=1 时是从卡中读数据，RD/WR=0 时啊写数据到卡中。 |
| CMD60 | adtc | [31] WR<br>[23:18] 地址<br>[7:2] 字节数<br>其他位为保留位 | R1(read)/<br>R1b(write) | RW_MULTIPLE_REGISTER | 在地址范围内，读或写寄存器。 |
| CMD61 | adtc | [31] WR<br>[15:0] 数据单元数<br>其他位为保留位 | R1(read)/<br>R1b(write) | RW_MULTIPLE_BLOCK | 在地址范围内，读或写寄存器。 |
| **注意：** 1. ACMDx 是针对 SD 存储卡的特定应用命令<br>2. CMD60, CMD61 针对 CE-ATA 设备的特定应用命令 | | | | | |

**表 24-12. I/O 模式命令(class 9)**

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| CMD39 | ac | [31:16] RCA<br>[15] 寄存器写标志<br>[14:8] 寄存器地址<br>[7:0] 寄存器数据 | R4 | FAST_IO | 用于写入和读取 8 位(寄存器)的数据字段。如果写标志被设置，该命令寻址寄存器，并提供数据写入。如果写标志被清为 0，R4 的响应中包含从寻址寄存器中读取的数据。该命令用于访问未在 MMC 标准定义的应用程序相关的寄存器。 |
| CMD40 | bcr | [31:0] 填充位 | R5 | GO_IRQ_STATE | 设置系统进入中断模式。 |
| CMD52 | adtc | [31] R/W 标志<br>[30:28] 功能数目<br>[27] RAW 标志<br>[26] 填充位<br>[25:9] 寄存器地址<br>[8] 填充位<br>[7:0] 写数据/填充 | R5 | IO_RW_DIRECT | IO_RW_DIRECT 命令提供简单的方式访问任意 I/O 功能的 128K 存储空间的寄存器。此命令可以实现使用单个命令对寄存器的读写。一个常见的用途是初始化寄存器或查询 I/O 功能状态。这个命令是读或写单 I/O 寄 |

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| | | 位 | | | 存器最快的方法，因为它仅需要一对单一的命令/响应。 |
| CMD53 | adtc | [31] R/W 标志<br>[30:28] 功能数目<br>[27] 块模式<br>[26] OP 码<br>[25:9] 寄存器地址<br>[8:0] 字节/块数 | | IO_RW_EXTENDED | 该命令允许用一个简单命令读取或写入大量的 I/ O 寄存器。 |
| **注意：** 1.CMD39, CMD40 仅用于 MMC 卡<br>2. CMD52, CMD53 仅用于 SD I/O 卡 | | | | | |

**表 24-13. 切换功能命令(class 10)**

| 命令索引 | 类型 | 参数 | 响应格式 | 简称 | 描述 |
|---|---|---|---|---|---|
| CMD6 | adtc | [31] 模式<br>0：检测功能<br>1：切换功能<br>[30:24] 保留<br>[23:20] 为功能组 6 保留(0h 或 Fh)<br>[19:16] 为功能组 5 保留(0h 或 Fh)<br>[15:12] 为功能组 4 保留(0h 或 Fh)<br>[11:8] 为功能组 3 保留(0h 或 Fh)<br>[7:4] 功能组 2 命令系统<br>[3:0] 功能组 1 访问模式 | R1 | SWITCH_FUNC | 仅用于 SD 存储卡和 SD I/O 卡。检测可切换功能(模式 0)和切换卡功能(模式 1)。 |

### 24.5.3. 响应

所有的响应都是通过 CMD 信号线发送。响应传输总是从对应响应字串的最左位开始。响应字串的长度依赖于响应类型。

#### 响应类型

响应的类型有七种，分别如下：

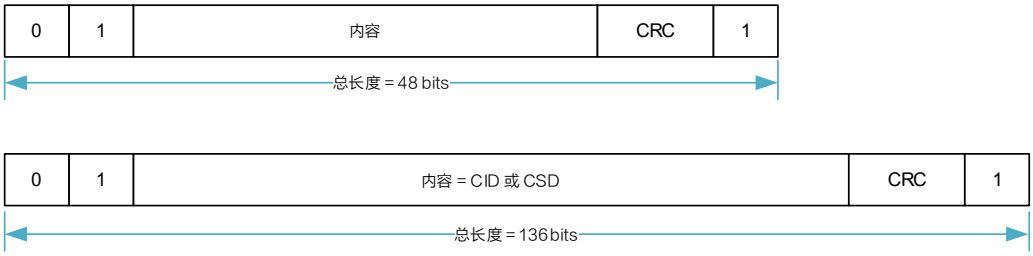
- **R1 / R1b :** 普通命令响应
- **R2 :** CID, CSD 寄存器
- **R3 :** OCR 寄存器
- **R4 :** Fast I/O
- **R5 :** 中断请求
- **R6 :** 发布的 RCA 响应
- **R7 :** 卡接口条件

SD 存储卡支持其中的五种响应，R1 / R1b, R2, R3, R6, R7。SD I/O 卡和 MMC 卡支持支持额外的响应类型，名为 R4 和 R5，但对于 SD I/O 卡和 MMC 卡，这两种响应并不完全相同。

### 响应格式

响应有两种格式，如*图 24-8. 响应令牌格式*所示，所有响应经由 CMD 线发出。代码的长度取决于响应类型。除了 R2 的长度是 136 位，其他的长度均为 48 位。

**图 24-8. 响应令牌格式**




响应总是从一个起始位(始终为 0)开始，随后第二位表示传输的方向(卡= 0)。下面表中的"x"的值表示为可变的部分。除了 R3 类型的所有响应由 CRC 校验。每个响应字段由结束位(总是 1)终止。

### R1 (普通命令响应)

代码长度为 48 位。位 45:40 指示要响应的命令索引，该值被解释为一个二进制编码的数字(0 到 63 之间)。卡的状态被 32 位编码。注意，如果写数据到卡上，在每个数据块传输之后会出现 BUSY 信号，在每个数据块传输完成后主机需要检查 BUSY 信号。卡状态在章节*卡的两种状态*中描述。

**表 24-14. R1 响应**

| 位 | 47 | 46 | [45:40] | [39:8] | [7:1] | 0 |
|---|---|---|---|---|---|---|
| 位宽 | 1 | 1 | 6 | 32 | 7 | 1 |

| 数值 | '0' | '0' | x | x | x | '1' |
|---|---|---|---|---|---|---|
| 描述 | 起始位 | 传输位 | 命令索引 | 卡状态 | CRC7 | 结束位 |

### R1b

R1b 格式与 R1 相同，但可以在数据线 DAT0 上发送忙信号。收到命令后，依据收到命令之前的状态，卡可能变为忙状态。主机应在响应中检查忙状态。

### R2 (CID, CSD 寄存器)

代码长度为 136 位。CID 寄存器的内容作为对命令 CMD2 和 CMD10 的响应被发送。CSD 寄存器的内容将作为以 CMD9 响应被发送。卡只响应发送 CID 和 CSD 的位[127.. 1]，这两个寄存器保留位[0]被替换为响应的结束位。

**表 24-15. R2 响应**

| 位 | 135 | 134 | [133:128] | [127:1] | 0 |
|---|---|---|---|---|---|
| 位宽 | 1 | 1 | 6 | 127 | 1 |
| 数值 | '0' | '0' | '111111' | x | '1' |
| 描述 | 起始位 | 传输位 | 保留 | CID 或 CSD 寄存器，内部 CRC7 | 结束位 |

### R3 (OCR 寄存器)

代码长度为 48 位。该 OCR 寄存器的内容作为 ACMD41(SD 存储卡)，CMD1(MMC)的响应被发送。不同卡的响应可能有一点不同。

**表 24-16. R3 响应**

| 位 | 47 | 46 | [45:40] | [39:8] | [7:1] | 0 |
|---|---|---|---|---|---|---|
| 位宽 | 1 | 1 | 6 | 32 | 7 | 1 |
| 数值 | '0' | '0' | '111111' | x | '1111111' | '1' |
| 描述 | 起始位 | 传输位 | 保留 | OCR 寄存器 | 保留 | 结束位 |

### R4 (Fast I/O)

仅适用于 MMC 卡。代码长度为 48 位。参数域包括选定卡的 RCA，被读取或写入寄存器的地址，和它的内容。如果操作成功，参数域状态位置位。

**表 24-17. R4 响应(MMC)**

| 位 | 47 | 46 | [45:40] | [39:8] 参数域 | | | | [7:1] | 0 |
|---|---|---|---|---|---|---|---|---|---|
| 位宽 | 1 | 1 | 6 | 16 | 1 | 7 | 8 | 7 | 1 |
| 数值 | '0' | '0' | '100111' | x | x | x | x | x | '1' |
| 描述 | 起始位 | 传输位 | CMD39 | RCA [31:16] | 状态 [15] | 寄存器地址 [14:8] | 读寄存器的内容 [7:0] | CRC7 | 结束位 |

### R4b

仅适用于 SD I/O 卡。代码长度为 48 位。SD I/O 卡接收到 CMD5 命令后会返回一个唯一的 SD I/O 卡响应 R4。

**表 24-18. R4 响应(SD I/O)**

| 位 | 47 | 46 | [45:40] | 39 | [38:36] | 35 | [34:32] | 31 | [30:8] | [7:1] | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 位宽 | 1 | 1 | 6 | 1 | 3 | 1 | 3 | 1 | 23 | 7 | 1 |
| 数值 | '0' | '0' | '111111' | x | x | x | '000' | x | x | '1111111' | 1 |
| 描述 | 起始位 | 传输位 | 保留 | C | I/O 功能数目 | 当前存储 | 填充位 | S18A | I/O OCR | 保留 | 结束位 |

### R5 (中断请求)

仅适用于 MMC 卡。代码长度为 48 位。若这个响应由主机产生，参数中 RCA 域为 0x0。

**表 24-19. R5 响应(MMC)**

| 位 | 47 | 46 | [45:40] | [39:8] 参数域 | | [7:1] | 0 |
|---|---|---|---|---|---|---|---|
| 位宽 | 1 | 1 | 6 | 16 | 16 | 7 | 1 |
| 数值 | '0' | '0' | '101000' | x | x | x | '1' |
| 描述 | 起始位 | 传输位 | CMD40 | 成功的卡或主机的 RCA [31:16] | [15:0]未定义，可能作为中断数据 | CRC7 | 结束位 |

### R5b

仅适用于 SD I/O 卡。SD I/O 卡对于 CMD52 和 CMD53 命令的响应是 R5。如果卡和主机之间的通信是在 1 位或 4 位 SD 模式下，响应应是 48 位响应(R5)。

**表 24-20. R5 响应(SD I/O)**

| 位 | 47 | 46 | [45:40] | [39:24] | [23:16] | [15:8] | [7:1] | 0 |
|---|---|---|---|---|---|---|---|---|
| 位宽 | 1 | 1 | 6 | 16 | 8 | 8 | 7 | 1 |
| 数值 | '0' | '0' | '11010X' | '0' | x | x | x | '1' |
| 描述 | 起始位 | 传输位 | CMD52/53 | 填充位 | 响应标志 | 读或写的数据 | CRC7 | 结束位 |

### R6 (发布的 RCA 响应)

代码长度为 48 位。位[45:40]表示对 CMD3 响应的命令索引。参数字段的 16 个最高位比特用于已发布的 RCA 号。

**表 24-21. R6 响应**

| 位 | 47 | 46 | [45:40] | [39:8] 参数域 | | [7:1] | 0 |
|---|---|---|---|---|---|---|---|
| 位宽 | 1 | 1 | 6 | 16 | 16 | 7 | 1 |
| 数值 | '0' | '0' | '000011' | x | x | x | '1' |
| 描述 | 起始位 | 传输位 | CMD3 | 新发布卡的 RCA | 卡的状态位: 23,22,19,12:0 | CRC7 | 结束位 |

### R7 (卡接口条件)

仅适用于 SD 存储卡。代码长度为 48 位。卡支持电压信息由 CMD8 的响应发送。位[19:16]表明该卡支持的电压范围。接受了供电电压的卡返回 R7 响应。在响应中，卡回送的参数设置电压范围和检查模式。

**表 24-22. R7 响应**

| 位 | 47 | 46 | [45:40] | [39:20] | [19:16] | [15:8] | [7:1] | 0 |
|---|---|---|---|---|---|---|---|---|
| 位宽 | 1 | 1 | 6 | 20 | 4 | 8 | 7 | 1 |
| 数值 | '0' | '0' | '001000' | '00000h' | x | x | x | '1' |
| 描述 | 起始位 | 传输位 | CMD8 | 保留位 | 可接受电压 | 回送检查模式 | CRC7 | 结束位 |

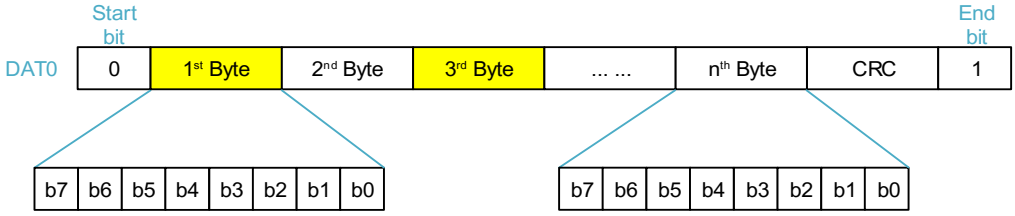
## 24.5.4. 数据包格式

数据总线模式有三种，1 位、4 位和 8 位宽度。1 位模式是强制的，4 位和 8 位模式是可选的。虽然使用 1 位模式，当卡复位和初始化时，DAT3 还需要通知卡当前的工作模式是 SDIO 或 SPI。

### 1 位数据包格式

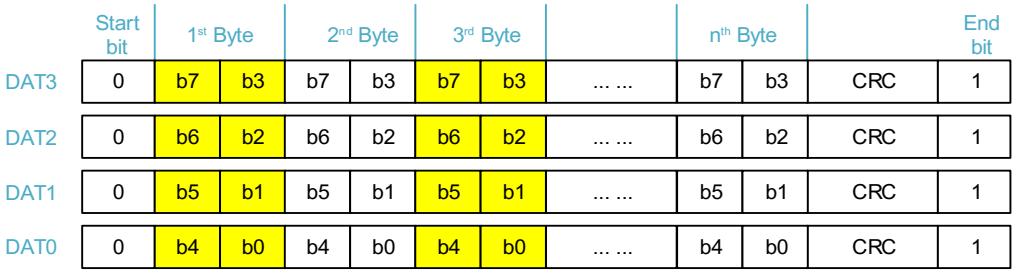
卡复位和初始化之后，只有 DAT0 被用于传输数据。其他引脚可以用于其他用处。*图 24-9. 1 位数据总线宽度*，*图 24-10. 4 位数据总线宽度*和*图 24-11. 8 位数据总线宽度*显示了数据宽度是 1 位，4 位和 8 位时的数据包格式。

**图 24-9. 1 位数据总线宽度**



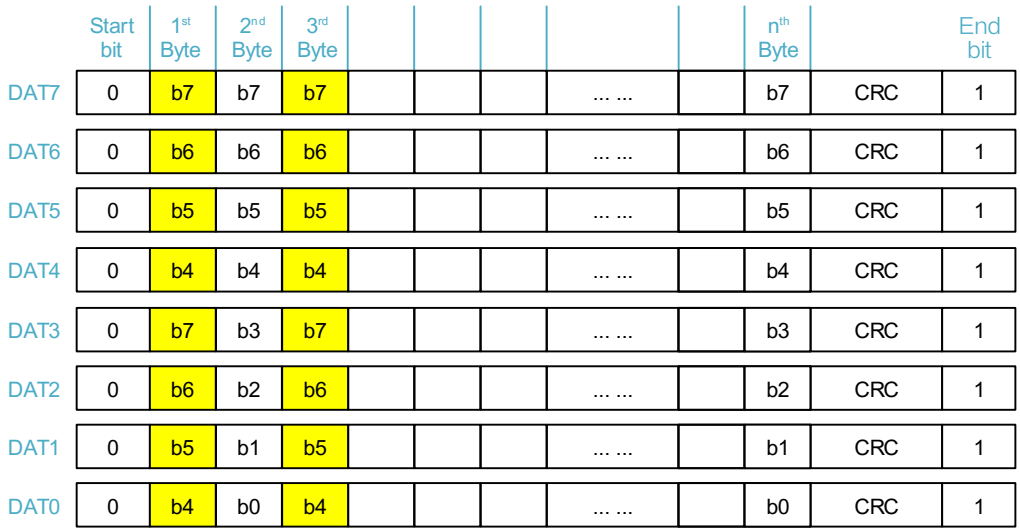

### 4 位数据包格式

**图 24-10. 4 位数据总线宽度**



**8 位数据包格式**

**图 24-11. 8 位数据总线宽度**




## 24.5.5. 卡的两种状态

SD 存储卡支持两种状态字段，而其他的卡只支持第一种：

卡状态：执行命令的错误和状态信息，在响应中指示。
SD 状态：512 位的扩展状态信息，支持特定功能的 SD 存储卡和未来应用特定功能。

### 卡状态

响应格式 R1 包含一个名为卡状态的 32 位字段。该字段用来传送该卡的状态的信息(可以存储在本地状态寄存器)到主机。除非特别说明，卡的状态信息总是与之前发出的命令相关。

表中的类型和清除条件的缩写如下：

**类型**

•E: 错误位。向主机发送错误条件。这些位一旦响应(报告错误)被发出去就会清除。
•S: 状态位。这些位仅作为信息字段，并不因为对命令的响应而改变。这些位是持久性的，它们根据卡状态被设置或被清除。
•R: 卡在命令解释和验证阶段(响应模式)检测到异常。
•X: 卡在命令执行阶段(执行模式)检测到异常。

**清除条件**

•A: 根据卡当前状态。
•B: 始终与之前命令相关。接收到有效命令可清除该状态(有命令延迟)。

•C: 读可清除。

**表 24-23. 卡状态**

| 位 | 标识符 | 类型 | 数值 | 说明 | 清除条件 |
|---|---|---|---|---|---|
| 31 | OUT_OF_RANGE | ERX | '0'= 无错误<br>'1'= 错误 | 命令的参数超出卡的允许范围。 | C |
| 30 | ADDRESS_ERROR | ERX | '0'= 无错误<br>'1'= 错误 | 在命令中使用与块长度不匹配的未对齐地址。 | C |
| 29 | BLOCK_LEN_ERROR | ERX | '0'= 无错误<br>'1'= 错误 | 所传输的块长度是卡不允许的，或者传输的字节数不匹配块的长度。 | C |
| 28 | ERASE_SEQ_ERROR | ER | '0'= 无错误<br>'1'= 错误 | 擦除命令顺序发生错误。 | C |
| 27 | ERASE_PARAM | ERX | '0'= 无错误<br>'1'= 错误 | 擦除时选择了无效的擦除块。 | C |
| 26 | WP_VIOLATION | ERX | '0'= 未保护<br>'1'= 已保护 | 当主机试图写一个受保护的块或暂时或永久写保护卡时置位。 | C |
| 25 | CARD_IS_LOCKED | SX | '0' = 卡未锁<br>'1' = 卡已锁 | 当设置该位，表示卡已经被主机锁住。 | A |
| 24 | LOCK_UNLOCK_FAILED | ERX | '0'= 无错误<br>'1'= 错误 | 在上锁/解锁中有命令的顺序错误或检测到密码错误时置位。 | C |
| 23 | COM_CRC_ERROR | ER | '0'= 无错误<br>'1'= 错误 | 之前命令的 CRC 校验错误。 | B |
| 22 | ILLEGAL_COMMAND | ER | '0'= 无错误<br>'1'= 错误 | 对于当前状态，命令非法。 | B |
| 21 | CARD_ECC_FAILED | ERX | '0'= 成功<br>'1'= 失败 | 卡的内部实施了 ECC 校验，但在更正数据时失败。 | C |
| 20 | CC_ERROR | ERX | '0'= 无错误<br>'1'= 错误 | 卡内部控制器错误。 | C |
| 19 | ERROR | ERX | '0'= 无错误<br>'1'= 错误 | 在操作过程中发生一般的或者未知的错误。 | C |
| 18 | UNDERRUN | ERX | '0'= 无错误<br>'1'= 错误 | 仅针对 MMC。该卡不支持在流读取模式下的数据传输。 | C |
| 17 | OVERRUN | ERX | '0'= 无错误<br>'1'= 错误 | 仅针对 MMC. 该卡不支持在流写入模式下的数据编程。 | C |
| 16 | CID/ CSD_OVERWRITE | ERX | '0'= 无错误<br>'1'= 错误 | 可能是下面两种错误之一:<br>- CSD 的只读部分与卡内容不匹配<br>- 试图进行拷贝或永久写保护的反向操作，即恢复原状或解除写保护 | C |
| 15 | WP_ERASE_SKIP | ERX | '0'= 未保护<br>'1'= 已保护 | 若置位，因为存在写保护数据块仅有部分地址空间被擦除；被暂时或者永久写保护的卡被擦除。 | C |

| 位 | 标识符 | 类型 | 数值 | 说明 | 清除条件 |
|---|---|---|---|---|---|
| 14 | CARD_ECC_DISABLED | SX | '0'= 使能<br>'1'= 失能 | 执行命令时未使用内部 ECC。 | A |
| 13 | ERASE_RESET | SR | '0'= 清除<br>'1'= 设置 | 因为收到一个擦除顺序之外的命令，擦除序列在执行前被清除。 | C |
| [12:9] | CURRENT_STATE | SX | 0 = 空闲<br>1 = 就绪<br>2 = 识别<br>3 = 待机<br>4 = 传输<br>5 = 发送数据<br>6 = 接收数据<br>7 = 编程<br>8 = 断开<br>9-14 = 保留<br>15 = 保留(I/O 模式) | 当收到命令时卡的状态。如果命令的执行导致状态的变化，这个变化将会在下个命令的响应中反映出来。这四个位按十进制数 0 至 15 解释。 | B |
| 8 | READY_FOR_DATA | SX | '0'= 未就绪<br>'1'= 就绪 | 与总线上的缓冲器空的信号一致。 | A |
| 7 | SWITCH_ERROR | EX | '0'= 无错误<br>'1'= 切换错误 | 如果置位，卡没有通过 SWITCH 命令切换到期望的模式。 | B |
| 6 | 保留 | | | | |
| 5 | APP_CMD | SR | '0'= 使能<br>'1'= 失能 | 卡期望 ACMD，或指示命令已经被解释为 ACMD 命令。 | C |
| 4 | 保留 | | | | |
| 3 | AKE_SEQ_ERROR | ER | '0'= 无错误<br>'1'= 错误 | 仅针对 SD 存储卡。验证过程的顺序有错误。 | C |
| 2 | 保留给与应用特定命令。 | | | | |
| [1:0] | 保留给厂商测试模式。 | | | | |

**注意：** 18, 17, 7 位仅适用于 MMC。14, 3 位仅适用于 SD 存储卡。

### SD 状态寄存器

在 SD 状态寄存器中含有与 SD 存储卡的专有特征相关的状态位，并且可以被用于未来的特定应用使用。SD 状态寄存器是大小是一个数据块 512 比特。该寄存器的内容连同一个 16 位 CRC 通过 DAT 总线被发送到主机上。SD 状态通过 DAT 总线被发送到主机上，作为 ACMD13 的响应(CMD55 接着用 CMD13)。ACMD13 只能在"传送状态"被发送到存储卡(卡被选中)。SD 状态结构将在下面描述。

"类型"和"清除条件"的缩写与上述卡状态描述相同。

表 24-24. SD 状态

| 位 | 标识符 | 类型 | 数值 | 描述 | 清除条件 |
|---|---|---|---|---|---|
| [511:510] | DAT_BUS_WIDTH | SR | '00'= 1 (默认)<br>'01'= 保留<br>'10'= 4 位宽<br>'11'= 保留 | 由 SET_BUS_WIDTH 命令显示当前定义的数据总线宽度 | A |
| 509 | SECURED_MODE | SR | '0'= 未处于安全模式<br>'1'= 处于安全模式 | 卡处于操作的安全模式(参考"SD 安全规范")。 | A |
| [508:496] | 保留 | | | | |
| [495:480] | SD_CARD_TYPE | SR | 下列卡目前被定义为:<br>'0000'= 通用 SD 读/写卡<br>'0001'= SD ROM 卡<br>'0002'= OTP | 低 8 位在未来被用来定义 SD 存储卡的不同变种(每个位将定义不同的 SD 卡类型)。高 8 位将被用来定义不符合当前 SD 物理层规范的 SD 卡。 | A |
| [479:448] | SIZE_OF_PROTECTED_AREA | SR | 受保护区域的大小。 | (见下面描述) | A |
| [447:440] | SPEED_CLASS | SR | 卡的速度类型。 | (见下面描述) | A |
| [439:432] | PERFORMANCE_MOVE | SR | 以 1MB/s 为单位的传输性能。 | (见下面描述) | A |
| [431:428] | AU_SIZE | SR | AU 大小 | (见下面描述) | A |
| [427:424] | 保留 | | | | |
| [423:408] | ERASE_SIZE | SR | 一次要被擦除的 AU 数目。 | (见下面描述) | A |
| [407:402] | ERASE_TIMEOUT | SR | UNIT_OF_ERASE_AU 指定的擦除区域的超时时间。 | (见下面描述) | A |
| [401:400] | ERASE_OFFSET | SR | 擦除时间增加固定偏移值。 | (见下面描述) | A |
| [399:312] | 保留 | | | | |
| [311:0] | 保留给生产厂商 | | | | |

**SIZE_OF_PROTECTED_AREA**

对于标准容量卡(SDSC)和高容量卡(SDHC/SDXC)设置该位域不同。

对于标准容量卡(SDSC)，受保护区域容量计算方式如下:

受保护区域 = SIZE_OF_PROTECTED_AREA_* MULT * BLOCK_LEN。
SIZE_OF_PROTECTED_AREA 以 MULT*BLOCK_LEN 为单位。

对于高容量卡(SDHC/SDXC)，受保护区域容量计算方式如下：
受保护区域 = SIZE_OF_PROTECTED_AREA 。
SIZE_OF_PROTECTED_AREA 以字节为单位。

**SPEED_CLASS**

这 8 位字段表示速度等级。

00h: Class 0
01h: Class 2
02h: Class 4
03h: Class 6
04h: Class 10
05h-FFh: 保留

**PERFORMANCE_MOVE**

这 8 位域指示 Pm，该值可被设为以 1MB/秒为单位。如果卡不用 RU 移动数据，应该认为 Pm 是无穷大。设置这个域为 FFh 表示无穷大。Pm 的最小值由*表 24-25. 移动性能字段*中定义。

**表 24-25. 移动性能字段**

| PERFORMANCE_MOVE | 数值定义 |
|---|---|
| 00h | 顺序写入 |
| 01h | 1 [MB/sec] |
| 02h | 2 [MB/sec] |
| ..... | ..... |
| FEh | 254 [MB/sec] |
| FFh | 无穷大 |

**AU_SIZE**

这 4 位字段指示 AU 大小，数值是 16K 字节为单位 2 的幂次的倍数。

**表 24-26. AU_SIZE 字段**

| AU_SIZE | 数值定义 |
|---|---|
| 0h | 未定义 |
| 1h | 16 KB |
| 2h | 32 KB |
| 3h | 64 KB |
| 4h | 128 KB |
| 5h | 256 KB |
| 6h | 512 KB |
| 7h | 1 MB |
| 8h | 2 MB |
| 9h | 4 MB |
| Ah | 8 MB |

| AU_SIZE | 数值定义 |
|---|---|
| Bh | 12 MB |
| Ch | 16 MB |
| Dh | 24 MB |
| Eh | 32 MB |
| Fh | 64 MB |

最大 AU 大小，取决于卡的容量，由*表 24-26. AU_SIZE 字段*中定义。卡可以任意的设置 AU 大小(由*表 24-27. 最大 AU 大小*定义)，只要小于或等于该卡容量所允许的最大 AU 大小。卡应该尽可能小地设置 AU 尺寸。

**表 24-27. 最大 AU 大小**

| 卡容量 | 最大 64MB | 最大 256MB | 最大 512MB | 最大 32GB | 最大 2TB |
|---|---|---|---|---|---|
| 最大 AU 大小 | 512 KB | 1 MB | 2 MB | 4 MB | 64MB |

**ERASE_SIZE**

这 16 位字段表示 $N_{ERASE}$。当 $N_{ERASE}$ 个数的 AU 被擦除，超时时间由 ERASE_TIMEOUT 规定(参考 ERASE_TIMEOUT)。主机应确定在一次操作中要被擦除的 AU 的适当数目，以便主机可以预示擦除操作的进度。如果该字段设置为 0，则不支持擦除的超时计算。

**表 24-28. 擦除大小字段**

| ERASE_SIZE | 数值定义 |
|---|---|
| 0000h | 不支持擦除的超时计算。 |
| 0001h | 1 AU |
| 0002h | 2 AU |
| 0003h | 3 AU |
| ..... | ..... |
| FFFFh | 65535 AU |

**ERASE_TIMEOUT**

这 6 位字段表示 $T_{ERASE}$，当 ERASE_SIZE 指示的多个 AU 被擦除时，这个数值给出了从偏移量算起的擦除超时时间。ERASE_TIMEOUT 的范围可以被定义为最多 63 秒，卡的制造商可以根据具体实现选择 ERASE_SIZE 和 ERASE_TIMEOUT 的任意组合。一旦 ERASE_TIMEOUT 被确定下来，那么 ERASE_SIZE 也确定了。主机可以通过以下公式计算任意数目的 AU 的擦除超时时间：

$$\text{Erase timeout of X AU} = \frac{T_{ERASE}}{N_{ERASE}} * X + T_{OFFSET} \quad (24\text{-}1)$$

**表 24-29. 擦除超时字段**

| ERASE_TIMEOUT | 数值定义 |
|---|---|
| 00 | 不支持擦除的超时计算 |
| 01 | 1 秒 |
| 02 | 2 秒 |
| 03 | 3 秒 |
| ..... | ..... |

| ERASE_TIMEOUT | 数值定义 |
|---|---|
| 63 | 63 秒 |

如果 ERASE_SIZE 字段被设置为 0，则该字段应该设置为 0。

**ERASE_OFFSET**

这 2 位字段表示 $T_{OFFSET}$，可以选择如*表 24-30. 擦除偏移字段*所示的四个数值之一。若 ERASE_SIZE 和 ERASE_TIMEOUT 字段都设为 0，该字段无意义。

**表 24-30. 擦除偏移字段**

| ERASE_OFFSET | 数值定义 |
|---|---|
| 0h | 0 秒 |
| 1h | 1 秒 |
| 2h | 2 秒 |
| 3h | 3 秒 |

## 24.6. 编程序列

### 24.6.1. 卡识别

主机复位后进入卡识别模式，寻找总线上的新卡。在卡识别模式下，主机复位所有的卡，验证工作电压范围，识别卡并询问每个卡的相对卡地址(RCA)。这个操作是在每个卡自己的命令信号线 CMD 上分别完成的。在卡识别模式中的所有数据通信只使用命令信号线(CMD)。

在卡识别过程中，卡应该工作在时钟频率为时钟速率 $F_{OD}$ (400 kHz)的情况下。

#### 卡复位

命令 GO_IDLE_STATE(CMD0)是软件复位命令，并设置 MMC 和 SD 存储卡进入空闲状态(Idle State)，不管当前卡的状态是什么。复位命令(CMD0)仅用于存储器或组合卡的存储器部分。为了重置只有 I/O 卡或组合卡的 I/O 部分，使用 CMD52 写 1 到 CCCR 的 RES 位。在非激活状态(Inactive State)的卡不受此命令的影响。

主机上电后，所有的卡都处于空闲状态(Idle State)，包括之前已在非激活状态(Inactive State)的卡。上电或 CMD0 后，所有卡的 CMD 线处于输入模式，等待下一个命令的起始位。这些卡都是用缺省的相对卡地址(RCA)初始化，并用默认 400 kHz 的时钟频率驱动器。

#### 工作电压范围验证

在主机和卡之间开始通信时，主机可能不知道卡支持的电压，并且卡可能不知道主机能否提供其支持的电压。为了验证电压，下面的命令都在相关规范中定义。

在协议规范中定义的命令包括：SEND_OP_COND (CMD1 用于 MMC)，SD_SEND_OP_COND (ACMD41 用于 SD 存储卡)，IO_SEND_OP_COND (CMD5 用于 SD I/O 卡)，这些命令提供给主机一种机制去识别和拒绝那些不匹配主机所需的 $V_{DD}$ 范围的卡。这

是由主机发送所需的 $V_{DD}$ 电压窗口作为此命令的操作数来实现的。如果卡不能在指定的范围内进行数据传输，必须从总线断开并进入非激活状态(Inactive State)。否则，该卡将响应返回它的 $V_{DD}$ 范围。

如果该卡可以工作在所提供的电压下，响应将返回供电电压和在命令参数中设置的检查模式。

如果该卡不能在提供的电压下工作，它不返回响应，并保持在空闲状态。初始化 SDHC 卡时强制性的在 ACMD41 命令之前发送 CMD8。收到 CMD8 是让该卡知道主机支持物理层 2.00 协议及卡支持高版本的功能。

#### 卡识别过程

对于不同的卡，卡的识别过程不同。这些卡包括 MMC、CE-ATA、SD，或 SD I/O 卡。支持所有类型的 SD I/O 卡，即 SDIO_IO_ONLY 卡、SDIO_MEM_ONLY 卡和 SDIO COMBO 卡。卡识别过程步骤如下：

1. 检测卡是否连接。

2. 识别卡的类型：SD 卡、MMC(CE-ATA)或 SD I/O 卡。
   - 发送 CMD5 命令。如果主机接收到响应，则是 SD I/O 卡；
   - 如果没有响应，发送 ACMD41。如果主机接收到响应，则是 SD 卡；
   - 否则，是 MMC 或者 CE-ATA 设备。

3. 根据卡的类型初始化卡。

使用 $F_{OD}$ (400 KHz)为时钟源，并按照下列命令顺序发送命令：
   - SD 卡 - 发送 CMD0，ACMD41，CMD2，CMD3；
   - SDHC 卡 - 发送 CMD0，CMD8，ACMD41，CMD2，CMD3；
   - SD I/O 卡 - 如果卡没有存储器端口，发送 CMD52，CMD0，CMD5，CMD3；否则，发送 CMD52，CMD0，CMD5，ACMD41，CMD11 (可选)，CMD2，CMD3；
   - MMC/CE-ATA - 发送 CMD0，CMD1，CMD2，CMD3。

4. 识别 MMC/CE-ATA 设备。
   - CPU 应该通过发送 CMD8 查询 EXT_CSD 寄存器的 504 字节(S_CMD_SET)。如果第 4 位被设置为 1，则该设备支持 ATA 模式；
   - 如果支持 ATA 模式，CPU 应通过设置 EXT_CSD 寄存器的 191 字节(CMD_SET)的(第 4 位)ATA 位选择 ATA 模式，以激活使用 ATA 命令集。 CPU 使用 SWITCH(CMD6)命令选择命令集；
   - 如果 CE-ATA 设备存在，FAST_IO(CMD39)和 RW_MULTIPLE_REGISTER(CMD60)命令将会成功，并且返回的数据将会是 CE-ATA 复位签名。

### 24.6.2. 无数据命令

发送任何无数据命令时，软件需要用适当的参数设置 SDIO_CMDCTL 寄存器和 SDIO_CMDAGMT 寄存器。通过这两个寄存器，主机形成命令，并将其发送到命令总线上。主机通过 SDIO_STAT 寄存器的错误标志来反映命令响应的错误。

当接收到响应时，主机设置 SDIO_STAT 寄存器 CMDRECV(CRC 校验通过)位或

CCRCERR(CRC 校验失败)位为 1。短响应被复制到 SDIO_RESP0，而长响应被复制到所有四个响应寄存器。SDIO_RESP3 寄存器的第 31 位代表的长响应的最高位，而 SDIO_RESP0 寄存器的第 0 位表示长响应最低位。

### 24.6.3. 单个数据块或多个数据块写

在发送块写入命令(CMD24 - CMD27)时，一个或多个数据块从主机传到卡。数据块由起始位(1 位或 4 位低电平)，数据块，CRC 和结束位(1 位或 4 位高电平)组成。如果 CRC 失败，则卡通过 SDIO_DAT 线指示传输失败，传送数据被丢弃而不写入，并且后续发送的数据块将被忽略。

如果主机传输的部分数据累积长度不是数据块对齐，并且块错位是不允许的(未设置 CSD 参数 WRITE_BLK_MISALIGN)，卡将在第一个未对齐块的开始之前检测块错位错误(设置状态寄存器的 ADDRESS_ERROR 错误位)，并同时忽略后续的数据传输。如果主机试图写一个写保护区的数据，写操作也将被终止。在这种情况下，卡将设置状态寄存器中 WP_VIOLATION 位。

设置 CID 和 CSD 寄存器不需要先设置块长度，传送的数据也通过 CRC 保护。如果 CSD 或 CID 寄存器的一部分被存储在 ROM 中，那么不可改变部分必须与接收缓冲区的对应部分相匹配。如果匹配失败，卡将报告一个错误同时不改变任何寄存器的内容。

一些卡可能需要很长的或者不可预测的时间写入一个数据块。接收一个数据块并完成 CRC 校验后，卡将开始写操作，如果写缓冲区已满则保持 DAT0 线拉低，并且无法通过新的命令 WRITE_BLOCK 接收新的数据。主机可以在任何时间用 SEND_STATUS 命令(CMD13)查询卡的状态，并且卡将返回当前状态。状态位 READY_FOR_DATA 表示卡是否可以接受新的数据或写入操作是否仍在进行中。主机可以通过发出 CMD7 命令不选中该卡(选择另外的卡)，将该卡置于断开状态(Disconnect State)，并释放 DAT 信号线而不中断写操作。当重新选择卡，如果写操作仍在进行中并且写缓冲区不可用，它会拉低 DAT 信号线重新激活忙指示。

对于 SD 卡。设置一些块被预擦除(ACMD23)操作将使多块写操作比没有 ACMD23 操作更快。主机将使用此命令来定义下一次操作将会有多少个数据块被发送。

单块或多块写操作步骤为：

1. 在 SDIO_DATALEN 寄存器中设置数据大小(以字节为单位)。
2. 在 SDIO_DATACTL 寄存器中设置数据块大小(BLKSZ，以字节为单位)；主机每次发送 BLKSZ 大小的数据块。
3. 在 SDIO_CMDAGMT 寄存器中设置数据应该被写入的地址。
4. 设置 SDIO_CMDCTL 寄存器。对于 SD 存储卡和 MMC 卡，使用 CMD24 命令为单块写和 CMD25 命令为多块写。对于 SD I/O 卡，使用 CMD53 命令来进行单块和多块传输。对于 CE-ATA，先用 CMD60 写 ATA 任务文件，然后使用 CMD61 命令写入数据。在写 CMD 寄存器之后，主机开始执行一个命令，当该命令被发送到总线时，CMDRECV 标志被设置。
5. 将数据写入 SDIO_FIFO。
6. 软件应查询数据错误中断。如果需要，软件可以通过发送停止命令(CMD12)终止数据传输。
7. 当收到 DTEND 中断时，数据传送结束。对于开放式的块传输，如果字节计数为 0，则软件必须发送 STOP 命令。如果字节计数不为 0，则在给定的字节数传送结束时，主机应该发送停止命令。

### 24.6.4. 单个数据块或多个数据块读

读数据块是基于块的数据传输。数据传输的基本单位是块，最大块大小在 CSD(READ_BL_LEN)中被定义，块的大小始终是 512 字节。如果 READ_BL_PARTIAL(在 CSD 中)被设置时，更小的块也可以被传输，其开始和结束地址被完全包含在 512 个字节的边界中。

CMD17(READ_SINGLE_BLOCK)表示开始读一个数据块，完成传输后卡返回发送状态。CMD18(READ_MULTIPLE_BLOCK)开始读连续的数据块。为了确保数据传输的完整性，每个数据块后都有一个 CRC 校验。

块长度由 CMD16 设置，可以设置为 512 字节而忽略 READ_BL_LEN 的设置。

数据块将不断传输，直到主机发出 STOP_TRANSMISSION 命令(CMD12)。由于串行命令传输原因，停止命令有一个执行的延迟。在停止命令的结束位之后停止数据传输。

当使用 CMD18 读到用户区的最后一个块时，主机应该忽略可能会出现的 OUT_OF_RANGE 错误，即使序列是正确的。

如果主机传输的部分块的累积长度不是块对齐并且不允许块错位，卡将在第一个未对齐块的开始检测出块错位，并设置状态寄存器的 ADDRESS_ERROR 错误位，中断传输和等待在数据状态的停止命令。

单块或多块读操作步骤为：

1. 在 SDIO_DATALEN 寄存器中设置数据大小的字节数。
2. 在 SDIO_DATACTL 寄存器中设置块大小(BLKSZ)。主机每次从卡中读取 BLKSZ 大小的数据。
3. 在 SDIO_ CMDAGMT 寄存器中设置需要读取数据的开始地址。
4. 设置 SDIO_ CMDCTL 寄存器。对于 SD 和 MMC 卡，使用 CMD17 用于单块读取和 CMD18 为多块读取。对于 SD I/O 卡，使用 CMD53 用于单块和多块传输。对于 CE-ATA，先用 CMD60 写 ATA 任务文件，然后使用 CMD61 来读取数据。设置 CMD 寄存器之后，主机开始执行该命令，当该命令被发送到总线时，CMDRECV 标志被设置。
5. 软件应查询数据错误中断。如果需要，软件可以通过发送停止命令(CMD12)终止数据传输。
6. 软件应从 FIFO 中读数据，并腾出 FIFO 的空间用于接收更多的数据。
7. 当收到 DTEND 中断时，软件应读出 FIFO 中剩余的数据。

### 24.6.5. 数据流写和数据流读 (仅适用于 MMC)

#### 数据流写

数据流写(CMD20)开始从主机将数据传送到卡，从起始地址开始，直到主机发出停止命令。如果允许部分块传输(如果 CSD 参数 WRITE_BL_PARTIAL 被设置)，数据流可以在卡地址空间内的任何地址启动和停止，否则应仅在块边界启动和停止。由于不预先确定要传输的数据量，CRC 不能使用。

如果主机提供了一个超出范围的地址作为参数传递给 CMD20，卡将拒绝该命令，留在传输状态，并将 ADDRESS_OUT_OF_RANGE 置位。

需要注意的是数据流写命令只适用于 1 位总线配置(DAT0 信号线上)。如果 CMD20 在其它总线配置中发出的，它被认为是非法的命令。

为了使卡保持在流模式的数据传输，接收数据所花费的时间(由总线时钟速率定义)必须比它需要写入到主存储器字段(由卡定义在 CSD 寄存器)的时间少。因此，流写入操作最大的时钟频率由下面给出的公式计算：

$$\max\ \text{write frequence} = \min\left(\text{TRAN_SPEED}, \frac{8*2^{\text{WRITE_BL_LEN}}-100*\text{NSAC}}{\text{TAAC}*\text{R2W_FACTOR}}\right) \quad (24\text{-}2)$$

其中，TRAN_SPEED：最大的总线时钟频
WRITE_BL_LEN：最大写数据块长度
NSAC：以 CLK 周期计算的数据读访问时间 2
TAAC：数据读访问时间 1
R2W_FACTOR：写速度因子

所有的参数在 CSD 寄存器中定义。如果主机试图使用更高频率，卡可能不能够对数据进行处理，并将停止编程，同时忽略所有后续的数据传输并等待(在接收数据状态)一个停止指令。由于主机发送 CMD12，该卡将 TXURE 位置位并返回传输状态。

### 数据流读

由 READ_DAT_UNTIL_STOP(CMD11)控制数据流的数据传输。此命令指示卡从指定地址发送数据，直到主机发送一个 STOP_TRANSMISSION(CMD12)命令。由于串行命令传输停止的原因，命令有一个执行的延迟。停止命令的结束位之后数据传输停止。

如果主机提供了一个超出范围的地址作为参数传递给 CMD11，该卡将拒绝该命令，留在传输状态，并将 ADDRESS_OUT_OF_RANGE 位置位。

需要注意的是数据流读取命令只工作在 1 位总线配置(DAT0 信号线)。如果 CMD11 在其它总线配置中发出的，它被认为是非法的命令。

如果数据传输的地址到达存储范围的结束处时，主机还没有发送停止命令，则后续传输的有效载荷的内容是不确定的。由于主机发送 CMD12 命令，卡将 ADDRESS_OUT_OF_RANGE 位置位并返回传输状态。

为了使卡保持在流模式的数据传输，传输数据所花费的时间(由总线时钟速率定义)必须比它需要从主存储器字段(在 CSD 寄存器中由卡定义)读出的时间少。因此，流读取操作最大的时钟频率由下面给出的公式计算：

$$\max\ \text{read frequence} = \min\left(\text{TRAN_SPEED}, \frac{8*2^{\text{READ_BL_LEN}}-100*\text{NSAC}}{\text{TAAC}*\text{R2W_FACTOR}}\right) \quad (24\text{-}3)$$

其中，TRAN_SPEED: 最大总线时钟频率
READ_BL_LEN: 最大读数据块长度
NSAC: 以 CLK 周期计算的数据读访问时间 2
TAAC: 数据读访问时间 1
R2W_FACTOR: 写速度因子

所有的参数在 CSD 寄存器中定义。如果主机试图使用更高频率，卡可能不能够对数据进行处

理，并将停止编程，同时忽略所有后续的数据传输并等待(在接收数据状态)一个停止指令。由于主机发送 CMD12，该卡将 RXORE 位置位并返回传输状态。

### 24.6.6. 擦除

MMC/ SD 存储卡的可擦除单位是"擦除组"，擦除组是以写数据块计算的，写数据块是卡的基本写入单元。擦除组的大小是一个卡特定的参数，在 CSD 中定义。

主机可以擦除连续范围的擦除组。开始擦除操作有三个步骤。首先，主机使用 ERASE_GROUP_START(CMD35)/ ERASE_WR_BLK_START(CMD32)命令定义了连续范围内的开始地址，然后使用 ERASE_GROUP_END(CMD36)/ ERASE_WR_BLK_END(CMD33)命令定义了连续范围内的结束地址，最后发送 ERASE(CMD38)命令启动擦除操作。在擦除命令中的地址字段是以字节为单位的擦除组地址。卡会舍弃未与擦除组大小对齐的部分，把地址边界对齐到擦除组的边界。

如果未按照定义的步骤接收到擦除命令(CMD35，CMD36 和 CMD38)，卡应设置状态寄存器的 ERASE_SEQ_ERROR 位，并重置整个序列。

如果主机提供了一个超出范围的地址作为参数传递给 CMD35 或 CMD36，卡将拒绝该命令，同时设置 ADDRESS_OUT_OF_RANGE 位，并重置整个擦除序列。

如果收到"非擦除"命令(既不是 CMD35，CMD36，CMD38 也不是 CMD13)，卡应该设置 ERASE_RESET 位，重置擦除序列并执行最后一个命令。

如果擦除范围包括写保护块，它们应不被擦除，只有非保护块被擦除。应设置状态寄存器的 WP_ERASE_SKIP 状态位。

如上所述，对于块写入，卡将通过保持 DAT0 为低来指示擦除过程正在进行。实际擦除时间可能很长，主机可以发送 CMD7 命令以取消选择该卡。

### 24.6.7. 总线宽度选择

在主机已经验证了总线上的功能引脚后，卡初始化后可以改变总线宽度的配置。

对于 MMC 卡，使用 SWITCH 命令(CMD6)。总线宽度的配置是通过在 EXT_CSD 寄存器模式字段的 BUS_WIDTH 字节设置而改变的。上电或软件复位后，BUS_WIDTH 字节的内容为 0x00。如果主机试图写一个无效的值时，BUS_WIDTH 字节不会改变，同时设置 SWITCH_ERROR 位，另外该寄存器是只写的。

对于 SD 存储卡，使用 SET_BUS_WIDTH 命令(ACMD6)改变总线宽度。上电或 GO_IDLE_STATE 命令(CMD0)后默认总线宽度为 1 位。 SET_BUS_WIDTH(ACMD6)仅在传送状态有效，这表明仅在由 SELECT/DESELECT_CARD (CMD7)命令选择卡之后总线宽度才可以改变。

### 24.6.8. 保护管理

为了允许主机保护数据，使得其不被擦除或改写，有三种卡保护方式：

**CSD 寄存器用于卡保护 (可选的)**

通过在 CSD 寄存器中设置永久或临时的写保护位,整个卡可以被写保护。一些卡通过设置 CSD 的 WP_GRP_ENABLE 位支持一组扇区的写保护。它的大小在 CSD 寄存器中的 WP_GRP_SIZE 单元定义。SET_WRITE_PROT 命令设置指定写保护组的写保护，CLR_WRITE_PROT 命令清除指定写保护组的写保护。

高容量 SD 存储卡不支持写保护，不响应写保护命令(CMD28，CMD29 和 CMD30)。

**写保护开关 (SD 存储卡和 SD I/O 卡)**

在卡的侧面有一个机械的滑动开关，提供给用户设置是否对卡进行写保护。如果滑动片处在窗口打开的位置表明该卡被写保护。如果在窗口关闭的位置则卡没有写保护。

**Password Card Lock/Unlock Operation**

卡密码上锁/解锁的保护方式在章节*卡上锁/解锁操作*中描述。

### 24.6.9. 卡上锁/解锁操作

密码保护的功能允许主机使用密码锁住卡，当解锁卡的时候也使用该密码。其中密码存储在 128 位的 PWD 寄存器当中，密码的长度存储在 PWD_LEN 的 8 位寄存器中。这些寄存器是非易失性的，以至于电源开关不会清除他们。

已经上锁的卡支持所有的基本命令(class 0)，ACMD41，CMD16 和锁卡命令(class 7)。因此主机可以对卡进行复位，初始化，选择，状态查询，但是无法获取卡上的数据。如果卡之前被设置过密码(PWD_LEN 的值为 0)，卡在每次上电后会自动上锁。

与存在的 CSD 寄存器写命令相同，上锁/解锁命令也只在卡的传输态有效。这意味着，上锁/解锁命令不包含地址参数，且必须在使用该命令前卡必须被选中。

卡上锁/解锁命令与卡单块写命令有着相同的结构和总线事务类型。传输的数据块包含命令所有需要的信息(密码设置模式，密码本身，卡上锁/解锁等)。*表 24-31. 上锁/解锁数据结构*为上锁/解锁命令的结构。

**表 24-31. 上锁/解锁数据结构**

| Byte | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| 0 | 保留(全设置为 0) | | | | ERASE | LOCK_UNLOCK | CLR_PWD | SET_PWD |
| 1 | PWDS_LEN | | | | | | | |
| 2 | 密码数据(PWD) | | | | | | | |
| …… | | | | | | | | |
| PWDS_LEN+1 | | | | | | | | |

**ERASE**: 该位为 1 时定义了强制擦除操作。字节 0 的位 3 将被设为 1(其他位应为 0)。所有该命令的其他字节将被卡忽略。

**LOCK/UNLOCK**: 1 = 上锁，0 = 解锁。注意，此位可以和 SET_PWD 一起设置，不可以和 CLR_PWD 一起设置。

**CLR_PWD**: 1 = 清除 PWD.

**SET_PWD**: 1 = 设置新的密码到 PWD

**PWDS_LEN**: 定义密码长度(字节)。在改变密码的情况下，这个长度应该是新旧密码长度之和。密码长度可达 16 个字节。在密码替换的情况下，新旧密码长度总和可达 32 个字节。

**密码数据(PWD)**: 在设置一个新密码的情况下，它包含这个新的密码。如果修改密码，它包含旧的密码，后面是设置的新密码。

### 设置密码

- 如果卡之前未被选中，使用 CMD7 选中卡。
- 使用 CMD16 定义数据块长度，8 位卡上锁/解锁模式，8 位密码长度(字节为单位)，新密码的字节数。在密码替换完成的情况下，块的大小应考虑新旧密码都会与命令一起被发送出去。
- 在数据线上，以合适的数据块大小发送卡上锁/解锁命令，包含 16 位的 CRC。数据块应指示模式(SET_PWD)，密码长度(PWDS_LEN)和密码本身。在密码替换完成的情况下，密码长度值(PWDS_LEN)应为新旧密码长度之和，密码数据字段应包括旧的密码(当前使用)，后面是新的密码。需要注意的是卡需要内部处理新密码长度的计算，通过从 PWDS_LEN 字段减去旧密码长度。
- 当发送的旧密码不正确(大小和内容不相同)，状态寄存器中的 LOCK_UNLOCK_FAILED 会被置位，并且旧的密码不会改变。如果发送的旧密码正确(大小和内容相同)，新的密码数据及其长度会分别保存在 PWD 和 PWD_LEN 中。

### 复位密码

- 如果卡之前未被选中，使用 CMD7 选中卡。
- 使用 CMD16 定义数据块长度，8 位卡上锁/解锁模式，8 位密码长度(字节为单位)，当前使用的密码的字节数。
- 在数据线上，以合适的数据块大小发送卡上锁/解锁命令，包含 16 位的 CRC。数据块指示模式(SET_PWD)，密码长度(PWDS_LEN)和密码本身。如果 PWD 和 PWD_LEN 的内容与发送的密码和其大小匹配，PWD 寄存器的内容会被清除，同时 PWD_LEN 被设为 0。如果密码不正确，状态寄存器中的 LOCK_UNLOCK_FAILED 会被置位。

### 卡上锁

- 如果卡之前未被选中，使用 CMD7 选中卡。
- 使用 CMD16 定义数据块长度，8 位卡上锁/解锁模式，8 位密码长度(字节为单位)，当前使用的密码的字节数。
- 在数据线上，以合适的数据块大小发送卡上锁/解锁命令，包含 16 位的 CRC。数据块指示 LOCK 模式，密码长度(PWDS_LEN)和密码本身。

如果 PWD 内容等于发送的密码，卡将会被上锁，并且状态寄存器中卡上锁状态位(CARD_IS_LOCKED)会被置位。如果密码不正确，状态寄存器中 LOCK_UNLOCK_FAILED 会被置位。

### 卡解锁

- 如果卡之前未被选中，使用 CMD7 选中卡。
- 使用 CMD16 定义数据块长度，8 位卡上锁/解锁模式，8 位密码长度(字节为单位)，当前使用的密码的字节数。
- 在数据线上，以合适的数据块大小发送卡上锁/解锁命令，包含 16 位的 CRC。数据块指示 UNLOCK 模式，密码长度(PWDS_LEN)和密码本身。

如果 PWD 内容等于发送的密码，卡将会被解锁，并且状态寄存器中卡上锁状态位(CARD_IS_LOCKED)会被清除。如果密码不正确，状态寄存器中 LOCK_UNLOCK_FAILED 会被置位。

## 24.7. 特定操作

### 24.7.1. SD I/O 特定操作

SD I/O 卡(包括仅 IO 卡和组合卡)支持这些特定操作：

**读等待操作**
**暂停/恢复操作**
**中断**

只有在 SDIO_ DATACTL [11]位被设置时， SD I/O 才支持这些操作，但暂停读操作除外，因为它不需要特定的硬件实现。

#### SD I/O 读等待操作

读等待(RW)操作是可选择的，仅用于 SD I/O 的 1 位和 4 位模式。读等待操作允许一个主机给卡在执行一个读多个块(CMD53)操作时发信号，以暂时停止数据传输，同时允许主机发送命令到 SD I/O 卡内任何功能函数。如果要判断一个卡是否支持读等待协议，主机应测试 CCCR 的卡功能字节的 SRW 功能位。读等待时序是基于中断周期的。如果卡不支持读等待协议，只能表明主机在读取多个命令控制 SDIO_CLK 时已经暂停(不中止)数据。这种方法的局限是，随着时钟停止，主机不能发出任何命令，所以在延迟期间不能执行其他操作。支持读等待的卡是强制性支持暂停和恢复的。*图 24-12. 通过停止 SDIO_CLK 的读等待操作*和*图 24-13. 使用 SDIO_DAT[2]信号线的读等待操作*所示为通过停止 SDIO_CLK 和使用 SDIO_DAT[2]读等待模式。

**图 24-12. 通过停止 SDIO_CLK 的读等待操作**

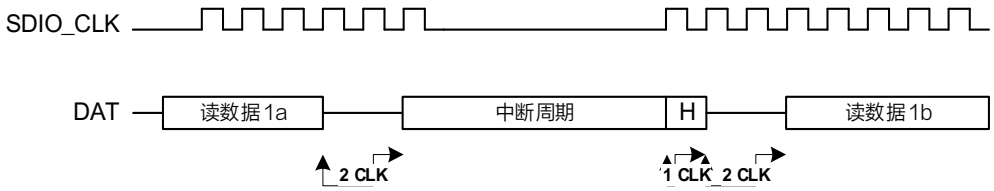
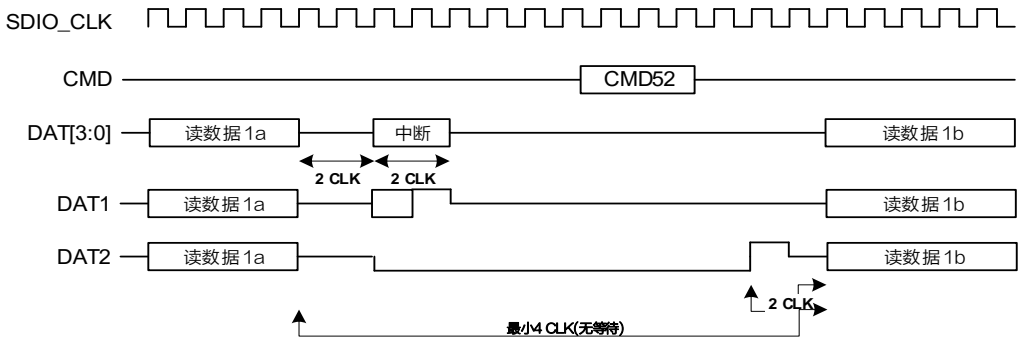


**图 24-13. 使用 SDIO_DAT[2]信号线的读等待操作**



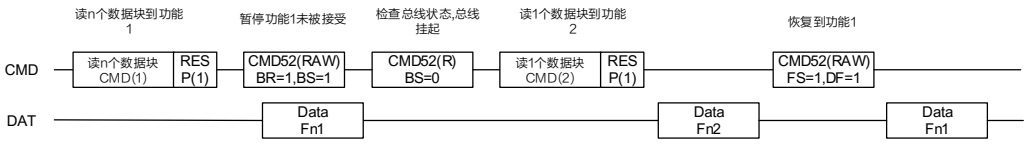

在接收到数据块之前就可以开始读等待：当数据单元使能(设置 SDIO_DATACTL[0]位)，SD I/O 特定操作使能(设置 SDIO_DATACTL[11]位)，开始读等待(SDIO_DATACTL[10] = 0 并且 SDIO_DATACTL[8] = 1)，数据方向为从卡到 SD I/O 主机 (SDIO_DATACTL[1] = 1)，DSM 直接从空闲状态到读等待状态。在读等待时，2 个 SDIO_CLK 时钟周期后，DSM 驱动 SDIO_DAT[2]为 0。在这种状态下，当设置了 RWSTOP 位(SDIO_DATACTL[9])时，DSM 会在等待状态多停留 2 个 SDIO_CLK 时钟周期，并在一个时钟周期中驱动 SDIO_DAT[2]为 1。然后 DSM 再次开始等待直到从卡里接收到数据。在接收数据块时，即使设置了开始读等待，DSM 也不会开始一个读等待间隔，读等待将在收到 CRC 后开始。必须清除 RWSTOP 才能开始新的读等待操作。在读等待期间，SDIO 主机可以在 SDIO_DAT[1]上监测 SD I/O 中断。

## SD I/O 暂停/恢复操作

对于多功能 SD I/O 或组合卡，它们有多个设备(I/O 和存储)共享 SD 总线。为了允许主机同时访问多个设备，SD I/O 和组合卡可以实现可选的暂停/恢复操作。如果卡支持暂停/恢复，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。

*图 24-14. 在功能 1 的多块读周期期间插入功能 2 读周期*显示第一次暂停请求没有立即接受的条件。然后主机检查一个读请求的状态，并确定该总线已被释放(BS = 0)。此时，功能 2 的读操作被启动。一旦读取单个块完成，恢复发送功能，从而恢复数据传输(DF = 1)。

**图 24-14. 在功能 1 的多块读周期期间插入功能 2 读周期**




当主机向卡发送数据时，主机可以暂停写操作。设置 SDIO_CMDCTL[11]位并指示 CSM 当前的命令是一个暂停命令。CSM 分析响应，当从卡收到响应时(暂停被接受)，它确认 DSM 在收到当前数据块的 CRC 后进入空闲状态。

为了暂停读操作，DSM 在 WaitR 状态等待，在停止数据传输之前，当功能被挂起时一个完整的数据包。随后应用程序继续读出接收 FIFO 直到 FIFO 为空，最后 DSM 自动地进入空闲状态。

## 中断

为了允许 SD I/O 卡中断主机，SD 接口增加了一个中断功能的引脚。在 4 位模式下，引脚 8 被用作 SDIO_DAT[1]，它被用于卡到主机的中断信号。对于每张卡中断的功能是可选的。SD I/O 中断"电平敏感"，即中断线应保持有效(低)直到卡要么被主机认可并采取行动，要么或者由于中断周期结束而解除有效状态。一旦主机服务中断，通过函数的唯一 I/O 操作清除中断。

当设置 SDIO_DATACTL[11]位，SD I/O 中断可以在 SDIO_DAT[1]信号线上检测到。

*图 24-15. 读中断周期时序*显示单个数据读周期的中断周期时序。

**图 24-15. 读中断周期时序**

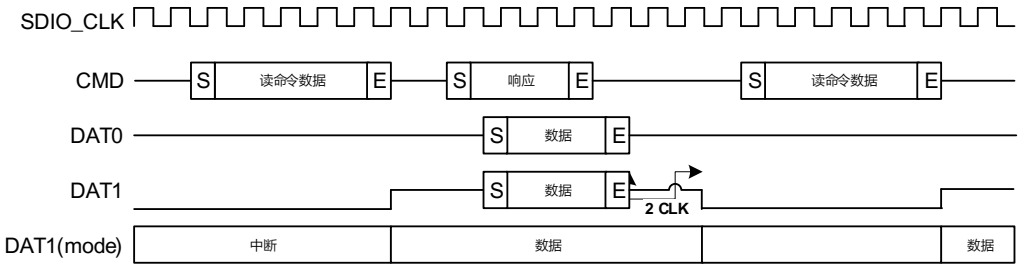
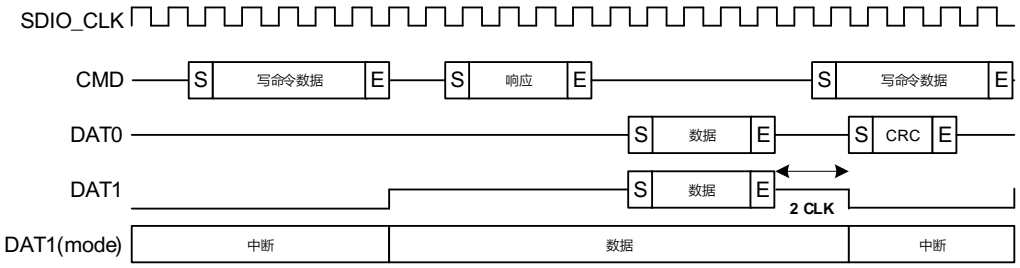



**图 24-16. 写中断周期时序**




当在 4 位 SD 模式传送数据的多个块时，需要中断周期的特定的定义。为了运行通信的最高速度，中断周期限制在 2 个时钟周期。卡如果想向主机发送一个中断信号，应该在第一个时钟周期设置 DAT1 为低，第二个时钟周期设置 DAT1 为高。然后卡应释放 DAT1 进入 Hi-Z 状态。*图 24-17. 4 位模式下多块读中断周期时序*显示了 4 位的多块读取时中断操作，*图 24-18. 4 位模式下多块写中断周期时序*显示了 4 位的多块写入时的中断操作。

**图 24-17. 4 位模式下多块读中断周期时序**

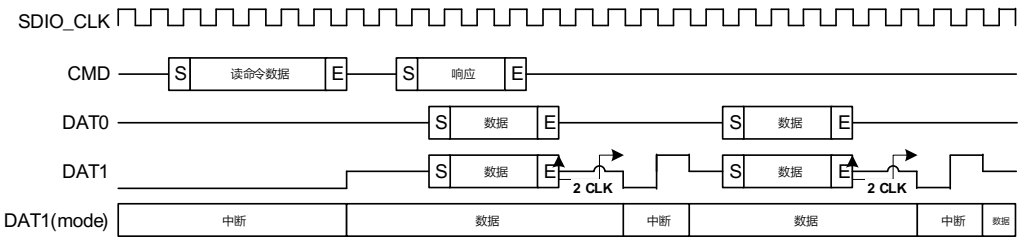
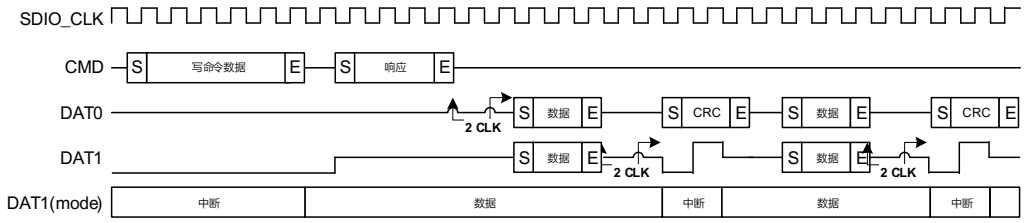


**图 24-18. 4 位模式下多块写中断周期时序**




## 24.7.2. CE-ATA 特定操作

CE-ATA 设备支持下述特定操作：

**接收命令完成信号**
**发送命令完成关闭信号**

只有当设置了 SDIO_CMDCTL[14]位时，SDIO 才支持这些操作。

### 命令完成信号

CE-ATA 定义了命令完成信号，设备使用该信号通知主机正常 ATA 命令完成或者由于设备遇到一个错误条件，ATA 命令终止。

如果"启用 CMD 完成"位 SDIO_CMDCTL[12]被设置并且"不中断使能'位 SDIO_CMDCTL[13]被设置，CSM 等待在 Waitcompl 状态的命令完成信号。

当在 CMD 线上接收到起始位，CSM 进入空闲状态。在 7 位周期之内不能发送新的命令。然后，在 5 个时钟周期内，把 CMD 信号变为 1(推挽模式)。

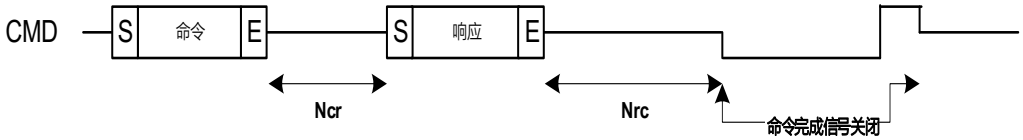
在主机从设备检测到一个命令完成信号之后，应该发送 FAST_IO(CMD39)命令来读取 ATA 状态寄存器以确定 ATA 命令的结束状态。

### 命令完成关闭信号

主机可以通过发送命令完成关闭信号来取消设备返回命令完成信号的功能。只有当主机在发送 RW_MULTIPLE_BLOCK (CMD61)之后接收到 R1b 响应后才能发送命令完成关闭信号。

如果未设置 SDIO_CMDCTL[12]中的"使能命令完成信号"并且重置了 SDIO_CMDCTL[13]中的"非中断使能位"，则在收到一个短响应后的 8 位周期之后，发出命令完成关闭信号。

**图 24-19. 命令完成信号关闭操作**



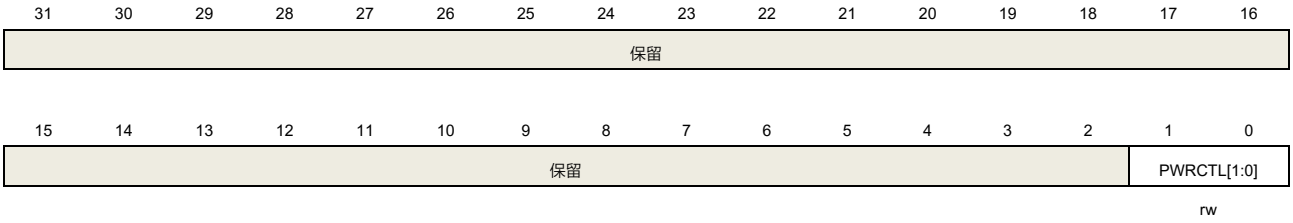
## 24.8. SDIO 寄存器

SDIO 基地址：0x4001 2C00

### 24.8.1. 电源控制寄存器 (SDIO_PWRCTL)

地址偏移：0x00
复位值：0x0000 0000

该寄存器只能按字(32 位)访问



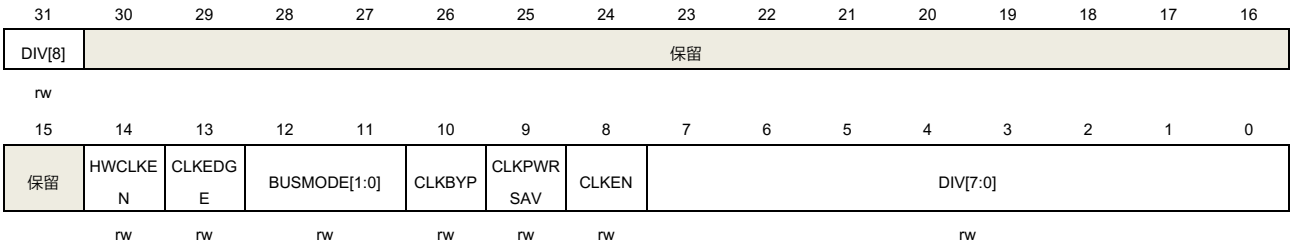

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值 |
| 1:0 | PWRCTL[1:0] | SDIO 电源控制位<br>这些位控制 SDIO 状态，卡输入或输出。<br>00：SDIO 电源关闭：SDIO CSM/DSM 复位到 IDLE，卡的时钟停止，没有命令/数据输出到卡<br>01：保留<br>10：保留<br>11：SDIO 上电 |

**注意：**两次对该寄存器写访问之间，需要至少 3 个 SDIOCLK 和 2 个 PCLK2 时钟周期，用于同步寄存器到 SDIOCLK 时钟域。

### 24.8.2. 时钟控制寄存器 (SDIO_CLKCTL)

地址偏移：0x04
复位值：0x0000 0000

该寄存器控制输出时钟 SDIO_CLK。
该寄存器只能按字(32位)访问



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | DIV[8] | 时钟分频系数的最高位<br>这个域定义了输入时钟(SDIOCLK)与输出时钟间的分频系数的最高位，参考 SDIO_CLKCTL 寄存器的 0 到 7 位。 |
| 30:15 | 保留 | 必须保持复位值 |
| 14 | HWCLKEN | 硬件时钟控制使能位<br>如果该位置位，根据系统总线是否非常忙，硬件控制 SDIO_CLK 开/关。由于硬件可以在快要下溢/上溢时关闭 SDIO_CLK，所以当该位被置位时不会有下溢/上溢错误。<br>0：关闭硬件时钟控制<br>1：开启硬件时钟控制 |
| 13 | CLKEDGE | SDIO_CLK 时钟边沿选择位<br>0：选择 SDIOCLK 的上升沿产生 SDIO_CLK<br>1：选择 SDIOCLK 的下降沿产生 SDIO_CLK |
| 12:11 | BUSMODE[1:0] | SDIO 卡总线模式控制位<br>00：1 位 SDIO 卡总线模式<br>01：4 位 SDIO 卡总线模式<br>10：8 位 SDIO 卡总线模式 |
| 10 | CLKBYP | 旁路时钟使能位<br>该位定义了 SDIO_CLK 直接来自于 SDIOCLK 或是 SDIOCLK 分频。<br>0：无旁路，SDIO_CLK 时钟参考 SDIO_CLKCTL 寄存器的 DIV 位域<br>1：旁路时钟，SDIO_CLK 时钟直接为 SDIOCLK (SDIOCLK/1) |
| 9 | CLKPWRSAV | SDIO_CLK 时钟动态开启/关闭以节省功耗<br>该位在总线空闲的时候，控制 SDIO_CLK 时钟动态开启/关闭以节省功耗。<br>0：SDIO_CLK 时钟总是开启<br>1：SDIO_CLK 时钟在总线空闲时关闭 |
| 8 | CLKEN | SDIO_CLK 时钟输出使能位<br>0：关闭 SDIO_CLK<br>1：开启 SDIO_CLK |
| 7:0 | DIV[7:0] | 时钟分频<br>该个域和 DIV[8]位定义了分频因子来向卡产生 SDIO_CLK 时钟。如果 CLKBYP 位为 0，SDIO_CLK 是由 SDIOCLK 分频得到，并且 SDIO_CLK 频率= SDIOCLK / (DIV[8:0] + 2)。 |

**注意：**两次对该寄存器写访问之间，需要至少 3 个 SDIOCLK 和 2 个 PCLK2 时钟周期，用于同步寄存器到 SDIOCLK 时钟域。

### 24.8.3. 命令参数寄存器(SDIO_CMDAGMT)

地址偏移：0x08
复位值：0x0000 0000

该寄存器定义了 32 位命令参数，这些参数将被用作于命令的一部分(位 39 到位 8)
该寄存器只能按字(32 位)访问

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | CMDAGMT[31:0] | SDIO 卡命令参数<br>这个域定义了将被发送到卡的 SDIO 卡命令参数。这个域是命令消息的位[39:8]。如果命令消息包含一个参数，在发送命令时，这个域应该在写 SDIO_CMDCTL 寄存器前更新。 |

## 24.8.4. 命令控制寄存器 (SDIO_CMDCTL)

地址偏移：0x0C
复位值：0x0000 0000

SDIO_CMDCTL 寄存器包含命令索引和其他命令控制位来控制命令状态机(CSM)。
该寄存器只能按字(32位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | ATAEN | NINTEN | ENCMDC | SUSPEND | CSMEN | WAITDEND | INTWAIT | CMDRESP[1:0] | | CMDIDX[5:0] | | | | | |
| | rw | rw | rw | rw | rw | rw | rw | rw | | rw | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:15 | 保留 | 必须保持复位值 |
| 14 | ATAEN | CE-ATA 命令使能(仅用于 CE-ATA)<br>如果该位置位，主机进入 CE-ATA 模式，并且 CSM 传输 CMD61。<br>0：CE-ATA 失能<br>1：CE-ATA 使能 |
| 13 | NINTEN | 无 CE-ATA 中断(仅用于 CE-ATA)<br>该位定义了有无 CE-ATA 中断。该位仅用于 CE-ATA 卡的情况。<br>0：CE-ATA 中断使能<br>1：CE_ATA 中断失能 |
| 12 | ENCMDC | 使能命令完成信号(仅用于 CE-ATA)<br>该位定义了在 CE-ATA 上有无命令完成信号。<br>0：无命令完成信号 |

1：有命令完成信号

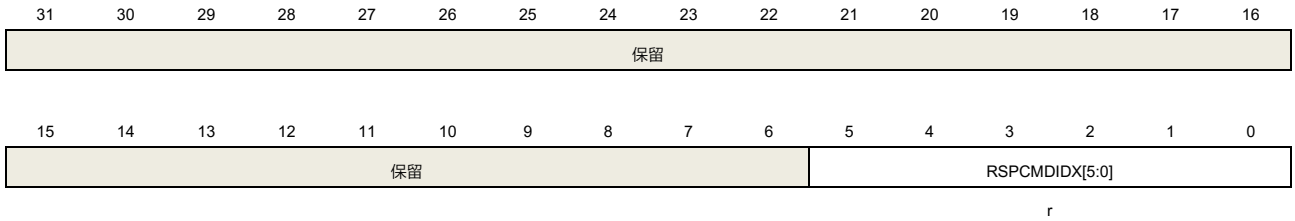
| | | |
|---|---|---|
| 11 | SUSPEND | SD I/O 暂停命令(仅用于 SD I/O)<br>该位定义了 CSM 是否发送了暂停命令。该位仅用于 SDIO 卡。<br>0：无影响<br>1：暂停命令 |
| 10 | CSMEN | 命令状态机(CSM)使能位<br>0：命令状态机失能(停留在 CS_Idle)<br>1：命令状态机使能 |
| 9 | WAITDEND | 等待数据传输结束<br>如果该位置位，命令状态机开始发送命令前需要等待数据传输结束。<br>0：无影响<br>1：等待数据传输结束 |
| 8 | INTWAIT | 中断等待超时<br>该位定义了命令状态机在 CS_Wait 状态等待卡中断。如果该位被置位，无命令等待超时生成。<br>0：无等待中断<br>1：等待中断 |
| 7:6 | CMDRESP[1:0] | 命令响应类型位<br>这些位定义了发送一个命令消息后的响应类型。<br>00：无响应<br>01：短响应<br>10：无响应<br>11：长响应 |
| 5:0 | CMDIDX[5:0] | 命令索引<br>这个域定义了将被发送到 SDIO 卡的命令索引。 |

**注意：**两次对该寄存器写访问之间，需要至少 3 个 SDIOCLK 和 2 个 PCLK2 时钟周期，用于同步寄存器到 SDIOCLK 时钟域。

### 24.8.5. 命令索引响应寄存器 (SDIO_RSPCMDIDX)

地址偏移：0x10
复位值：0x0000 0000

该寄存器只能按字(32位)访问



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:6 | 保留 | 必须保持复位值 |
| 5:0 | RSPCMDIDX[5:0] | 最后响应的命令索引<br>只读位域。这个域包含收到的最后命令响应的命令索引。如果响应没有命令索引(R3的长响应和短响应)，这个寄存器的内容是不未定义的。 |

### 24.8.6. 响应寄存器 (SDIO_RESPx x=0..3)

地址偏移：0x14+(4*x), x=0..3
复位值：0x0000 0000

这些寄存器包含最后收到的卡响应的内容。
该寄存器只能按字(32位)访问

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | RESPx[31:0] | 卡状态。响应内容由*表 24-32. 不同响应类型对应的 SDIO_RESPx 寄存器*所示。 |

短响应为 32 位，长响应为 127 位(位 128 是结束位 0)。

**表 24-32. 不同响应类型对应的 SDIO_RESPx 寄存器**

| 寄存器 | 短响应 | 长响应 |
|---|---|---|
| SDIO_RESP0 | 卡响应 [31:0] | 卡响应 [127:96] |
| SDIO_RESP1 | 保留 | 卡响应 [95:64] |
| SDIO_RESP2 | 保留 | 卡响应 [63:32] |
| SDIO_RESP3 | 保留 | 卡响应 [31:1]，加上位 0 |

### 24.8.7. 数据超时寄存器 (SDIO_DATATO)

地址偏移：0x24
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | DATATO[31:0] | 数据超时时间<br>这些位定义了数据超时时间，由 SDIO_CLK 计数。当 DSM 进入 WaitR 或 BUSY 状态，该寄存器的值加载到内部计数器开始递减。DSM 超时并进入空闲状态，当计数器的值减至 0 时设置 DTTMOUT 标志。 |

**注意：**当需要数据传输时，数据定时器寄存器和数据长度寄存器应在写数据控制寄存器前更新。

## 24.8.8. 数据长度寄存器 (SDIO_DATALEN)

地址偏移：0x28
复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | DATALEN[24:16] | | | | | | | | |
| | | | | | | | rw | | | | | | | | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:25 | 保留 | 必须保持复位值 |
| 24:0 | DATALEN[24:0] | 数据传输长度<br>该寄存器定义了需要传输的字节数。当数据传输开始时，数据计数器加载到这个寄存器并开始递减。 |

**注意：**如果选择了数据块传输，该寄存器的内容应该为块大小的倍数(参考 SDIO_DATACTL 寄存器)。当需要数据传输时，数据定时器寄存器和数据长度寄存器应在写数据控制寄存器前更新。

## 24.8.9. 数据控制寄存器 (SDIO_DATACTL)

地址偏移：0x2C
复位值：0x0000 0000

该寄存器控制 DSM。该寄存器只能按字(32 位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | IOEN | RWTYPE | RWSTOP | RWEN | BLKSZ[3:0] | | | | DMAEN | TRANSMOD | DATADIR | DATAEN |
| | | | | rw | rw | rw | rw | rw | | | | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|

| | | |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值 |
| 11 | IOEN | SD I/O 特定功能使能(仅用于 SD I/O)<br>0：未使能 SD I/O 特定功能<br>1：使能 SD I/O 特定功能 |
| 10 | RWTYPE | 读等待类型(仅用于 SD I/O)<br>0：使用 SDIO_DAT[2] 控制读等待<br>1：通过停止 SDIO_CLK 控制读等待 |
| 9 | RWSTOP | 读等待停止(仅用于 SD I/O)<br>0：无影响<br>1：如果 RWEN 位被置位，停止读等待过程 |
| 8 | RWEN | 读等待模式使能(仅用于 SD I/O)<br>0：读等待模式失能<br>1：读等待模式使能 |
| 7:4 | BLKSZ[3:0] | 数据块大小<br>这些位定义了当数据传输是块传输时数据块的大小。<br>0000：块大小 = $2^0$ = 1 字节<br>0001：块大小 = $2^1$ = 2 字节<br>0010：块大小 = $2^2$ = 4 字节<br>0011：块大小 = $2^3$ = 8 字节<br>0100：块大小 = $2^4$ = 16 字节<br>0101：块大小 = $2^5$ = 32 字节<br>0110：块大小 = $2^6$ = 64 字节<br>0111：块大小 = $2^7$ = 128 字节<br>1000：块大小 = $2^8$ = 256 字节<br>1001：块大小 = $2^9$ = 512 字节<br>1010：块大小 = $2^{10}$ = 1024 字节<br>1011：块大小 = $2^{11}$ = 2048 字节<br>1100：块大小 = $2^{12}$ = 4096 字节<br>1101：块大小 = $2^{13}$ = 8192 字节<br>1110：块大小 = $2^{14}$ = 16384 字节<br>1111：保留 |
| 3 | DMAEN | DMA 使能位<br>0：DMA 失能<br>1：DMA 使能 |
| 2 | TRANSMOD | 数据传输模式<br>0：块传输模式<br>1：流传输或 SDIO 多字节传输模式 |
| 1 | DATADIR | 数据传输方向<br>0：写数据到卡上<br>1：从卡中读取数据 |

| | | |
|---|---|---|
| 0 | DATAEN | 数据传输使能位<br>写 1 到该位开启数据传输不管该位为 0 或 1。如果 RWEN 置位，DSM 进入到读等待状态，或者根据 DATADIR 位 DSM 进入 WaitS 或 WaitR 状态。<br>开始一个新的数据传输，不需要清该位为 0。 |

**注意：**两次对该寄存器写访问之间，需要至少 3 个 SDIOCLK 和 2 个 PCLK2 时钟周期，用于同步寄存器到 SDIOCLK 时钟域。

## 24.8.10. 数据计数寄存器 (SDIO_DATACNT)

地址偏移：0x30
复位值：0x0000 0000

该寄存器为只读类型。当 DSM 从空闲状态进入 WaitR 或者 WaitS 时，该寄存器从数据长度寄存器(SDIO_DATALEN)加载数值。随着数据传输，数值不断递减直至为 0，随后 DSM 进入空闲状态并设置数据结束标志 DTEND。
该寄存器只能按字(32位)访问

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:25 | 保留 | 必须保持复位值 |
| 24:0 | DATACNT[24:0] | 数据计数值<br>只读位域。当读取这些位时，返回待传输剩余数据的字节数。 |

## 24.8.11. 状态寄存器 (SDIO_STAT)

地址偏移：0x34
复位值：0x0000 0000

该寄存器为只读类型。下面描述标志的类型：
位[23:22, 10:0]的标志只能通过向中断清除寄存器(SDIO_INTC)中相应的位写'1'清除。
位[21:11]的标志是根据硬件逻辑而发送变化的。
该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | ATAEND | SDIOINT | RXDTVAL | TXDTVAL | RFE | TFE | RFF | TFF |
| | | | | | | | | r | r | r | r | r | r | r | r |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RFH | TFH | RXRUN | TXRUN | CMDRUN | DTBLKEND | STBITE | DTEND | CMDSEND | CMDRECV | RXORE | TXURE | DTTMOUT | CMDTMOUT | DTCRCERR | CCRCERR |

r r r r r r r r r r r r r r r r

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值 |
| 23 | ATAEND | CE-ATA 命令完成信号已接收(仅用于 CMD61) |
| 22 | SDIOINT | SD I/O 中断已接收 |
| 21 | RXDTVAL | 接收 FIFO 中的数据有效 |
| 20 | TXDTVAL | 发送 FIFO 中的数据有效 |
| 19 | RFE | 接收 FIFO 为空 |
| 18 | TFE | 发送 FIFO 为空，当硬件流控制使能，并且 FIFO 中包含 2 个字时，TFE 信号变得有效。 |
| 17 | RFF | 接收 FIFO 为满，当硬件流控制使能，RFF 信号在 FIFO 差 2 个字就满时变得有效。 |
| 16 | TFF | 发送 FIFO 为满 |
| 15 | RFH | 接收 FIFO 半满：FIFO 中至少还有 8 个字可被读取 |
| 14 | TFH | 发送 FIFO 半空：至少还有 8 个字可被写入到 FIFO 中 |
| 13 | RXRUN | 正在接收数据 |
| 12 | TXRUN | 正在传输数据 |
| 11 | CMDRUN | 正在传输命令 |
| 10 | DTBLKEND | 数据块已发送/已接收(CRC 检测通过) |
| 9 | STBITE | 总线上起始位错误 |
| 8 | DTEND | 数据结束(数据计数器，SDIO_DATACNT 为零) |
| 7 | CMDSEND | 命令已发送(不需响应) |
| 6 | CMDRECV | 命令响应已接收(CRC 检测通过) |
| 5 | RXORE | 接收 FIFO 上溢错误发生 |
| 4 | TXURE | 发送 FIFO 下溢错误发生 |
| 3 | DTTMOUT | 数据超时，数据超时时间取决于 SDIO_DATATO 寄存器。 |
| 2 | CMDTMOUT | 命令响应超时，命令超时时间为 64 个 SDIO_CLK 时钟周期的固定值。 |
| 1 | DTCRCERR | 数据块已发送/已接收(CRC 检测失败) |
| 0 | CCRCERR | 命令响应已接收(CRC 检测失败) |

### 24.8.12. 中断清除寄存器 (SDIO_INTC)

地址偏移：0x38
复位值：0x0000 0000

该寄存器为只读。对该寄存器的位写 1 可以清除 SDIO_STAT 寄存器中相应的状态位。
该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | ATAENDC | SDIOINTC | 保留 | | | | | |
| | | | | | | | | w | w | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | DTBLKENDC | STBITEC | DTENDC | CMDSENDC | CMDRECVC | RXOREC | TXUREC | DTTMOUTC | CMDTMOUTC | DTCRCERRC | CCRCERRC |
| | | | | | w | w | w | w | w | w | w | w | w | w | w |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值 |
| 23 | ATAENDC | ATAEND 标志清除位<br>写 1 清除标志。 |
| 22 | SDIOINTC | SDIOINT 标志清除位<br>写 1 清除标志。 |
| 21:11 | 保留 | 必须保持复位值 |
| 10 | DTBLKENDC | DTBLKEND 标志清除位<br>写 1 清除标志。 |
| 9 | STBITEC | STBITE 标志清除位<br>写 1 清除标志。 |
| 8 | DTENDC | DTEND 标志清除位<br>写 1 清除标志。 |
| 7 | CMDSENDC | CMDSEND 标志清除位<br>写 1 清除标志。 |
| 6 | CMDRECVC | CMDRECV 标志清除位<br>写 1 清除标志。 |
| 5 | RXOREC | RXORE 标志清除位<br>写 1 清除标志。 |
| 4 | TXUREC | TXURE 标志清除位<br>写 1 清除标志。 |
| 3 | DTTMOUTC | DTTMOUT 标志清除位<br>写 1 清除标志。 |

| | | |
|---|---|---|
| 2 | CMDTMOUTC | CMDTMOUT 标志清除位<br>写 1 清除标志。 |
| 1 | DTCRCERRC | DTCRCERR 标志清除位<br>写 1 清除标志。 |
| 0 | CCRCERRC | CCRCERR 标志清除位<br>写 1 清除标志。 |

### 24.8.13. 中断使能寄存器 (SDIO_INTEN)

地址偏移：0x3C
复位值：0x0000 0000

该寄存器使能 SDIO_STAT 寄存器中相应状态位的中断。该寄存器只能按字(32 位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | ATAENDIE | SDIOINTIE | RXDTVALIE | TXDTVALIE | RFEIE | TFEIE | RFFIE | TFFIE |
| | | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RFHIE | TFHIE | RXRUNIE | TXRUNIE | CMDRUNIE | DTBLKENDIE | STBITEIE | DTENDIE | CMDSENDIE | CMDRECVIE | RXOREIE | TXUREIE | DTTMOUTIE | CMDTMOUTIE | DTCRCERRIE | CCRCERRIE |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值 |
| 23 | ATAENDIE | CE-ATA 命令完成信号已接收中断使能<br>写 1 使能中断。 |
| 22 | SDIOINTIE | SD I/O 中断已接收中断使能<br>写 1 使能中断。 |
| 21 | RXDTVALIE | 接收 FIFO 中的数据有效中断使能<br>写 1 使能中断。 |
| 20 | TXDTVALIE | 发送 FIFO 中的数据有效中断使能<br>写 1 使能中断。 |
| 19 | RFEIE | 接收 FIFO 空中断使能<br>写 1 使能中断。 |
| 18 | TFEIE | 发送 FIFO 空中断使能<br>写 1 使能中断。 |
| 17 | RFFIE | 接收 FIFO 满中断使能<br>写 1 使能中断。 |
| 16 | TFFIE | 发送 FIFO 满中断使能<br>写 1 使能中断。 |

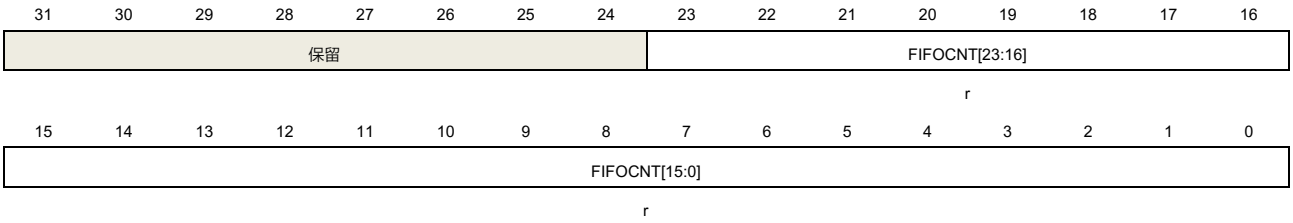
| | | |
|---|---|---|
| 15 | RFHIE | 接收 FIFO 半满中断使能<br>写 1 使能中断。 |
| 14 | TFHIE | 发送 FIFO 半满中断使能<br>写 1 使能中断。 |
| 13 | RXRUNIE | 正在接收数据中断使能<br>写 1 使能中断。 |
| 12 | TXRUNIE | 正在传输数据中断使能<br>写 1 使能中断。 |
| 11 | CMDRUNIE | 正在传输命令中断使能<br>写 1 使能中断。 |
| 10 | DTBLKENDIE | 数据块已发送/已接收中断使能<br>写 1 使能中断。 |
| 9 | STBITEIE | 起始位错误中断使能<br>写 1 使能中断。 |
| 8 | DTENDIE | 数据结束中断使能<br>写 1 使能中断。 |
| 7 | CMDSENDIE | 命令已发送中断使能<br>写 1 使能中断。 |
| 6 | CMDRECVIE | 命令响应已接收中断使能<br>写 1 使能中断。 |
| 5 | RXOREIE | 接收 FIFO 上溢错误中断使能<br>写 1 使能中断。 |
| 4 | TXUREIE | 发送 FIFO 下溢错误中断使能<br>写 1 使能中断。 |
| 3 | DTTMOUTIE | 数据超时中断使能<br>写 1 使能中断。 |
| 2 | CMDTMOUTIE | 命令响应超时中断使能<br>写 1 使能中断。 |
| 1 | DTCRCERRIE | 数据 CRC 错误中断使能<br>写 1 使能中断。 |
| 0 | CCRCERRIE | 命令响应 CRC 错误中断使能<br>写 1 使能中断。 |

### 24.8.14. FIFO 计数寄存器 (SDIO_FIFOCNT)

地址偏移：0x48

复位值：0x0000 0000

该寄存器只能按字(32位)访问

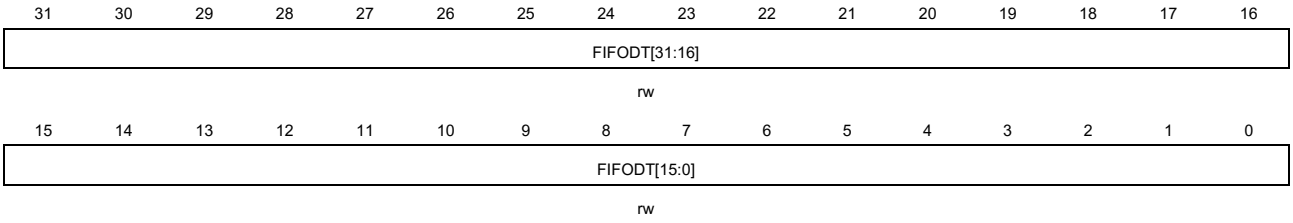



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | 保留 | 必须保持复位值 |
| 23:0 | FIFOCNT[23:0] | FIFO 计数器<br>这些位定义了从 FIFO 中读取或写入到 FIFO 剩余的字数。当 DATAEN 置位时，它加载数据长度寄存器的值(如果 SDIO_DATALEN 是字对齐时，该值为 SDIO_DATALEN[24:2]；如果 SDIO_DATALEN 不是字对齐，该值为 SDIO_DATALEN[24:2]+1)，然后当写一个字到 FIFO 或从 FIFO 中读取一个字时，开始递减计数。 |

## 24.8.15. FIFO 数据寄存器（SDIO_FIFO)

地址偏移：0x80
复位值：0x0000 0000

该寄存器占用了 32 个 32 位的字，地址偏移从 0x80 到 0xFC。
该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | FIFODT[31:0] | 接收 FIFO 数据或发送 FIFO 数据<br>这些位为接收 FIFO 或发送 FIFO 的数据。读或写该寄存器相当于对 FIFO 读或写数据。 |

# 25. 外部存储器控制器(EXMC)

## 25.1. 简介

外部存储器控制器EXMC，用来访问各种片外存储器，通过配置寄存器，EXMC可以把AMBA协议转换为专用的片外存储器通信协议，包括SRAM，ROM，NOR Flash，NAND Flash，PC Card和SDRAM。用户还可以调整相关的时间参数来提高通信效率。EXMC的访问空间被划分为许多个块(Bank)，每个块支持特定的存储器类型，用户可以通过对Bank的控制寄存器配置来控制外部存储器。

## 25.2. 主要特征

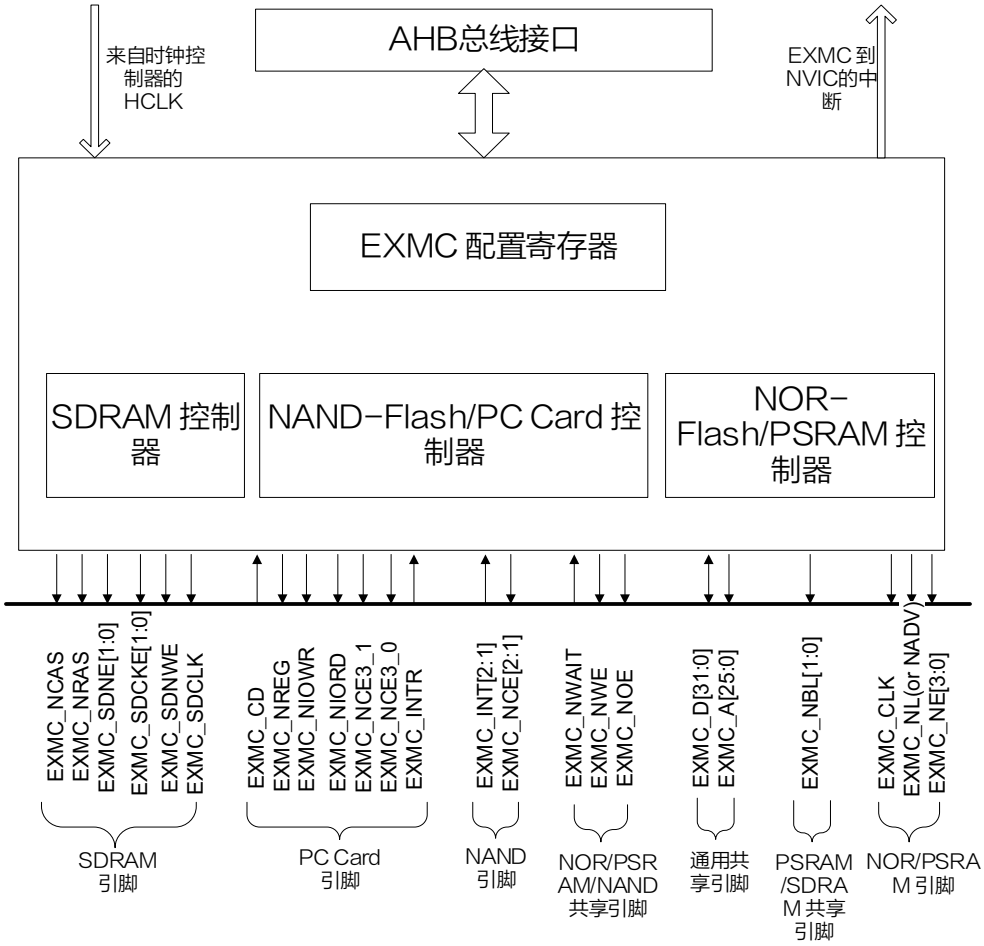
- 支持片外存储器类型：
  - SRAM
  - PSRAM/SQPI-PSRAM
  - ROM
  - NOR Flash
  - 8 位或 16 位 NAND Flash
  - 16 位 PC Card
  - SDRAM
- AMBA协议与各种片外存储器协议转换；
- 时序参数可编程可以满足用户特定需求；
- 每个Bank有独立的片选信号；
- 对于部分存储器类型支持独立的读写时序；
- 对于NAND Flash内置硬件ECC；
- 支持8位，16位，32位总线带宽；
- NOR Flash和PSRAM支持地址总线和数据总线的复用；
- 提供写使能和字节选择信号；
- 当AMBA总线宽度与外部存储器数据宽度不同时，会自动分割操作。

## 25.3. 功能说明

### 25.3.1. 结构框图

EXMC由6个模块组成：AHB总线接口，EXMC配置寄存器，NOR/PSRAM控制器，NAND/PC Card控制器，SDRAM控制器和外部设备接口。AHB时钟(HCLK)是参考时钟。

**图 25-1. 系统架构**



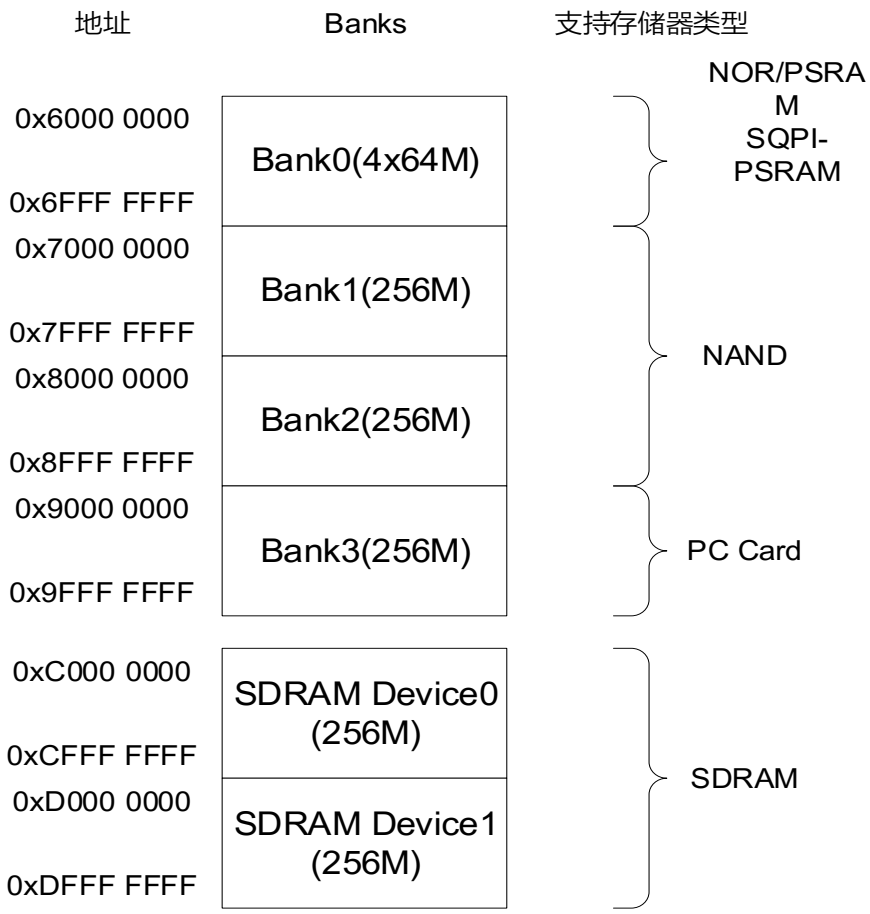

## 25.3.2. EXMC 访问基本规范

EXMC是AHB总线至外部设备协议的转换接口。32位的AHB读写操作可以转化为几个连续的8位或16位读写操作。在数据传输的过程中，AHB数据宽度和存储器数据宽度可能不相同。为了保证数据传输的一致性，EXMC读写访问需要遵从以下规范：

- AHB 访问宽度等于存储器宽度，则没有问题；
- AHB 访问宽度大于存储器宽度，则自动将 AHB 访问分割成几个连续的存储器数据宽度的传输；
- AHB 访问宽度小于存储器宽度。如果外部存储设备具有字节选择功能，如 SRAM、ROM、PSRAM，则可通过它的字节通道 EXMC_NBL[1:0]来访问对应的字节。否则禁止写操作，只允许读操作。

### 25.3.3. 外部设备地址映射

**图 25-2. EXMC Bank 划分**




EXMC将外部存储器分成多个Bank，每个Bank占256M字节，其中Bank0又分为4个Region，每个Region占64M字节。Bank1和Bank2又都被分成2个Section，分别是属性存储空间和通用存储空间。Bank3分成3个Section，分别是属性存储空间，通用存储空间和I/O存储空间。

每个Bank和Region都有独立的片选控制信号，也都能进行独立的配置。

Bank0用于访问NOR、PSRAM设备。

Bank1和Bank2用于连接NAND Flash，且每个Bank连接一个NAND。

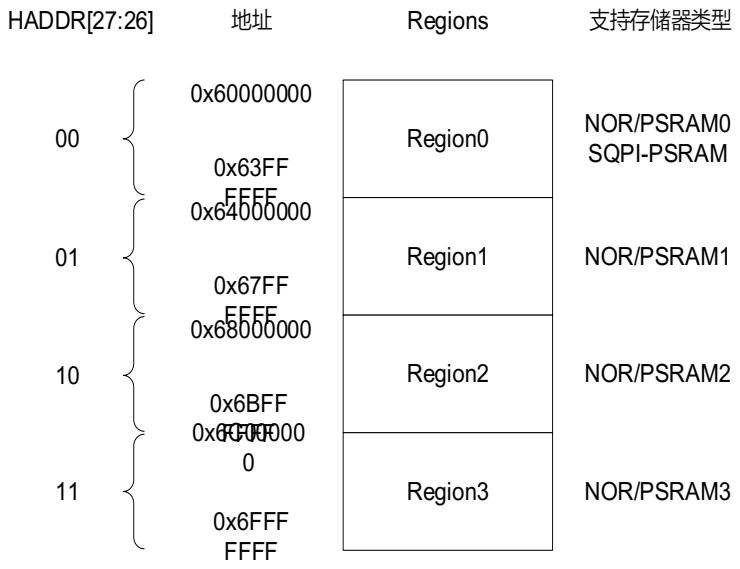
Bank3用于连接PC卡。

SDRAM Device0和SDRAM Device1用于连接SDRAM。

#### NOR 和 PSRAM 的地址映射

*图25-3. Bank0地址映射*是Bank0四个Region的地址映射。AHB地址线HADDR[27：26]作为四个Region的片选信号。

**图 25-3. Bank0 地址映射**



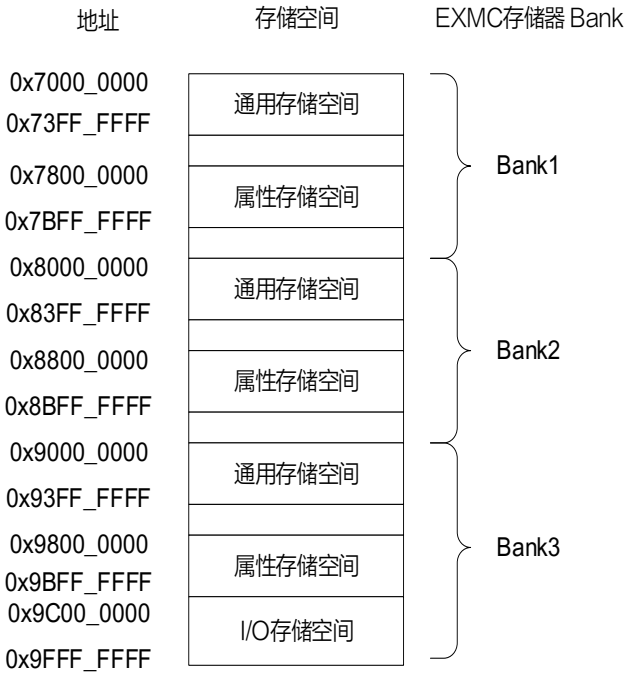

由于HADDR[25:0]是字节地址，而外部存储器访问有可能不是按字节访问的，所以会出现地址不一致的情况，但EXMC能实现对HADDR的调整以适应外部存储器的数据宽度。具体规则如下：

- 如果外部存储器的数据宽度是 8 位按字节对齐，EXMC 内部将 HADDR[25:0]与 EXMC_A[25:0]相连，然后 EXMC_A[25:0]与外部存储器的地址线相连；
- 如果外部存储器的数据宽度是 16 位按半字对齐，就需要将 HADDR 的字节地址转化为半字地址之后再连接外存储器。EXMC 内部将 HADDR[25:1]与 EXMC_A[24:0]相连，然后 EXMC_A[24:0]与外部存储器的地址线相连；
- 如果外部存储器的数据宽度是 32 位按字对齐，就需要将 HADDR 的字节地址转化为字地址之后再连接外存储器。EXMC 内部将 HADDR[25:2]与 EXMC_A[23:0]相连，然后 EXMC_A[23:0]与外部存储器的地址线相连。

### NAND/PC Card 地址映射

Bank1和Bank2用来访问NAND Flash，Bank3用来访问PC Card。每个Bank如*图 25-4. NAND/PC Card地址映射*被分为多个存储空间。

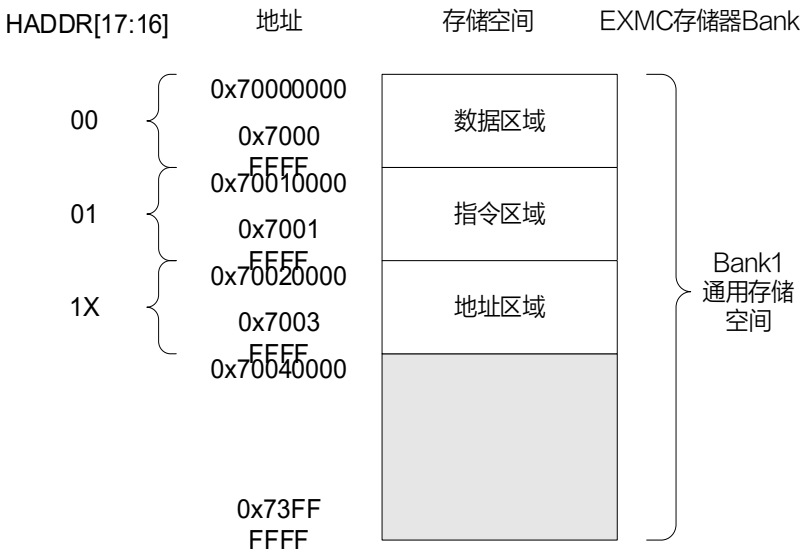
**图 25-4. NAND/PC Card 地址映射**




## NAND 地址映射

对于NAND Flash，通用和属性空间又可以细划分为3个区域。*图25-5. Bank1通用空间*为Bank1通用存储空间的数据区域，指令区域和地址区域的划分。

**图 25-5. Bank1 通用空间**




AHB利用HADDR[17:16]来实现对以上三个区的选择：

- HADDR[17:16]=00，即选择数据区；

- HADDR[17:16]=01，即选择命令区；
- HADDR[17:16]=1X，即选择地址区。

应用软件使用这3个区访问NAND FLASH。操作规则如下：

地址区：指定操作NAND Flash的地址，软件在地址区写入地址。在地址传输过程中，EXMC会使能地址锁存信号(ALE)，ALE映射到EXMC_A[17]。

指令区：指定NAND Flash将要执行的指令，软件在指令区写入指令。在指令传输过程中，EXMC会使能命令锁存信号(CLE)，CLE映射到EXMC_A[16]。

数据区：NAND FLASH读写数据，软件在数据区读出或写入数据。当EXMC在数据发送模式，软件需要在数据区写入数据，当EXMC在数据接收模式，软件需要在数据区读取数据。由于NAND FLASH会自动累加其内部操作地址，故在读写时不需要软件修改操作地址。

### SDRAM 地址映射

HADDR[28]位用来选两个SDRAM Bank，如*图25-6. SDRAM地址映射*所示。

**图 25-6. SDRAM 地址映射**

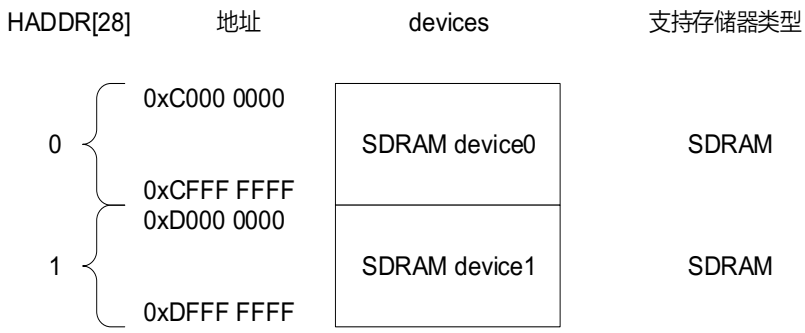



*表25-1. SDRAM地址映射*展示了SDRAM的13位行地址和11位列地址的配置映射：

**表 25-1. SDRAM 地址映射**

| 存储器数据宽度 | 内部bank | 行地址 | 列地址 | 最大存储容量 |
|---|---|---|---|---|
| 8-bit | HADDR[25:24] | HADDR[23:11] | HADDR[10:0] | 64 Mbytes:<br>4 x 8K x 2K |
| 16-bit | HADDR[26:25] | HADDR[24:12] | HADDR[11:1] | 128 Mbytes:<br>4 x 8K x 2K x 2 |
| 32-bit | HADDR[27:26] | HADDR[25:13] | HADDR[12:2] | 256 Mbytes:<br>4 x 8K x 2K x 4 |

## 25.3.4. NOR/PSRAM 控制器

EXMC模块的NOR/PSRAM控制器控制Bank0，它可以支持NOR Flash、PSRAM、SRAM、ROM和CRAM外部存储器。EXMC对Bank0每个Region输出一个唯一的片选信号，NE[x](x=0..3)，用于在4个Region中进行片选，所有其他的信号都是共享的。每个Region 都有专门的寄存器控制。但是只有Region0支持访问SQPI-PSRAM存储器。

**注意**:

在异步模式下，所有控制器输出信号在内部AHB总线时钟(HCLK)的上升沿改变。

在同步模式下，所有控制器输出数据在外部存储器时钟(EXMC_CLK)的下降沿改变。

### NOR/PSRAM 接口描述

**表 25-2. NOR Flash 接口信号描述**

| EXMC 引脚 | 传输方向 | 模式 | 功能描述 |
|---|---|---|---|
| EXMC_CLK | 输出 | 同步 | 同步时钟信号 |
| 非复用 EXMC_A[25:0] | 输出 | 异步/同步 | 地址总线 |
| 复用 EXMC_A[25:16] | | | |
| EXMC_D[15:0] | 输入/输出 | 异步/同步<br>(复用) | 地址/数据总线 |
| | 输入/输出 | 异步/同步<br>(非复用) | 数据总线 |
| EXMC_NE[x] | 输出 | 异步/同步 | 片选， x=0/1/2/3 |
| EXMC_NOE | 输出 | 异步/同步 | 读使能 |
| EXMC_NWE | 输出 | 异步/同步 | 写使能 |
| EXMC_NWAIT | 输入 | 异步/同步 | 等待输入信号 |
| EXMC_NL(NADV) | 输出 | 异步/同步 | 地址有效 |

**表 25-3. PSRAM 非复用接口信号描述**

| EXMC 引脚 | 传输方向 | 模式 | 功能描述 |
|---|---|---|---|
| EXMC_CLK | 输出 | 同步 | 同步时钟信号 |
| EXMC_A[25:0] | 输出 | 异步/同步 | 地址总线 |
| EXMC_D[15:0] | 输入/输出 | 异步/同步 | 数据总线 |
| EXMC_NE[x] | 输出 | 异步/同步 | 片选，x=0/1/2/3 |
| EXMC_NOE | 输出 | 异步/同步 | 读使能 |
| EXMC_NWE | 输出 | 异步/同步 | 写使能 |
| EXMC_NWAIT | 输入 | 异步/同步 | 等待输入信号 |
| EXMC_NL(NADV) | 输出 | 异步/同步 | 地址锁存信号 |
| EXMC_NBL[1] | 输出 | 异步/同步 | 高字节使能 |
| EXMC_NBL[0] | 输出 | 异步/同步 | 低字节使能 |

**表 25-4. SQPI-PSRAM 接口信号描述**

| EXMC 引脚 | 传输方向 | 模式 | 功能描述 |
|---|---|---|---|
| EXMC_CLK | 输出 | 同步 | 时钟信号 |
| EXMC_NE[0] | 输出 | 同步 | 片选，低有效 |
| EXMC_D[0] | 输入/输出 | 同步 | 数据，命令信号 |
| EXMC_D[1] | 输入/输出 | 同步 | 数据信号(SPI/SQPI/QPI 模式) |
| EXMC_D[3:2] | 输入/输出 | 同步 | 数据信号<br>(SQPI/QPI 模式) |

### 支持的存储器访问模式

*表25-5. EXMC对NOR，PSRAM和SRAM支持的访问模式*列出了EXMC对NOR，PSRAM和SRAM支持的访问模式。

**表 25-5. EXMC 对 NOR，PSRAM 和 SRAM 支持的访问模式**

| 存储器类型 | 访问模式 | 读/写 | AHB 传输宽度 | 存储器传输宽度 | 注释 |
|---|---|---|---|---|---|
| NOR Flash | 异步 | R | 8 | 16 | |
| | 异步 | R | 16 | 16 | |
| | 异步 | W | 16 | 16 | |
| | 异步 | R | 32 | 16 | 分成 2 次 EXMC 访问 |
| | 异步 | W | 32 | 16 | 分成 2 次 EXMC 访问 |
| | 同步 | R | 16 | 16 | |
| | 同步 | R | 32 | 16 | |
| PSRAM | 异步 | R | 8 | 16 | |
| | 异步 | W | 8 | 16 | 使用字节信号 NBL[1：0] |
| | 异步 | R | 16 | 16 | |
| | 异步 | W | 16 | 16 | |
| | 异步 | R | 32 | 16 | 分成 2 次 EXMC 访问 |
| | 异步 | W | 32 | 16 | 分成 2 次 EXMC 访问 |
| | 同步 | R | 16 | 16 | |
| | 同步 | R | 32 | 16 | |
| | 同步 | W | 8 | 16 | 使用字节信号 NBL[1：0] |
| | 同步 | W | 16 | 16 | |
| | 同步 | W | 32 | 16 | |
| SRAM and ROM | 异步 | R | 8 | 8 | |
| | 异步 | R | 8 | 16 | |
| | 异步 | R | 16 | 8 | 分成 2 次 EXMC 访问 |
| | 异步 | R | 16 | 16 | |
| | 异步 | R | 32 | 8 | 分成 4 次 EXMC 访问 |
| | 异步 | R | 32 | 16 | 分成 2 次 EXMC 访问 |
| | 异步 | W | 8 | 8 | |
| | 异步 | W | 8 | 16 | 使用字节信号 NBL[1：0] |
| | 异步 | W | 16 | 8 | |
| | 异步 | W | 16 | 16 | |
| | 异步 | W | 32 | 8 | |
| | 异步 | W | 32 | 16 | |

### NOR Flash/PSRAM 控制时序

EXMC为SRAM、ROM、PSRAM、NOR Flash等外部静态存储器提供可编程的时序参数以及多种时序模型以满足不同的需求。

**表 25-6. NOR/PSRAM 控制时序参数**

| 参数 | 功能 | 访问模式 | 单位 | 最小值 | 最大值 |
|---|---|---|---|---|---|
| CKDIV | 同步时钟分频比 | 同步 | HCLK | 2 | 16 |
| DLAT | 数据延迟 | 同步 | EXMC_CLK | 2 | 17 |
| BUSLAT | 总线延迟 | 异步/同步读 | HCLK | 0 | 15 |
| DSET | 数据建立时间 | 异步 | HCLK | 1 | 255 |
| AHLD | 地址保持时间 | 异步(复用) | HCLK | 1 | 15 |
| ASET | 地址建立时间 | 异步 | HCLK | 0 | 15 |

**表 25-7. EXMC 时序模型**

| 时序模型 | | 扩展模式 | 模式描述 | 写时序参数 | 读时序参数 |
|---|---|---|---|---|---|
| 异步 | 模式 1 | 0 | SRAM/PSRAM/CRAM | DSET<br>ASET | DSET<br>ASET |
| | 模式 2 | 0 | NOR Flash | DSET<br>ASET | DSET<br>ASET |
| | 模式 A | 1 | SRAM/PSRAM/CRAM 在数据阶段 EXMC_NOE 翻转 | WDSET<br>WASET | DSET<br>ASET |
| | 模式 B | 1 | NOR Flash | WDSET<br>WASET | DSET<br>ASET |
| | 模式 C | 1 | NOR Flash 在数据阶段 EXMC_NOE 翻转 | WDSET<br>WASET | DSET<br>ASET |
| | 模式 D | 1 | 有地址保持功能 | WDSET<br>WAHLD<br>WASET | DSET<br>AHLD<br>ASET |
| | 模式 AM | 0 | NOR Flash 数据/地址复用 | DSET<br>AHLD<br>ASET<br>BUSLAT | DSET<br>AHLD<br>ASET<br>BUSLAT |
| 同步 | 模式 E | 0 | NOR/PSRAM/CRAM 同步读 PSRAM/CRAM 同步写 | DLAT<br>CKDIV | DLAT<br>CKDIV |
| | 模式 SM | 0 | NOR Flash 数据/地址复用 | DLAT<br>CKDIV | DLAT<br>CKDIV |

如*表25-7. EXMC时序模型*所示，EXMC模块NOR Flash/PSRAM控制器可以提供多种时序模型。用户可以通过修改*表25-6. NOR/PSRAM控制时序参数*中列出的参数来使之适合不同类型外部存储器的时序以及满足用户的要求。当将寄存器EXMC_SNCTLx位EXMODEN置1使能扩展模式后，可以通过寄存器EXMC_SNTCFGx和EXMC_SNWTCFGx将读写配置成独立的时序。

EXMC_CLK可以通过CCK位来设置。如果CCK是0，当NOR Flash使用同步模式时会产生EXMC_CLK；如果CCK是1，当NOR Flash同步模式和异步模式都会产生EXMC_CLK。

### 异步访问时序

模式1 - SRAM/CRAM

图 25-7. 模式 1 读时序

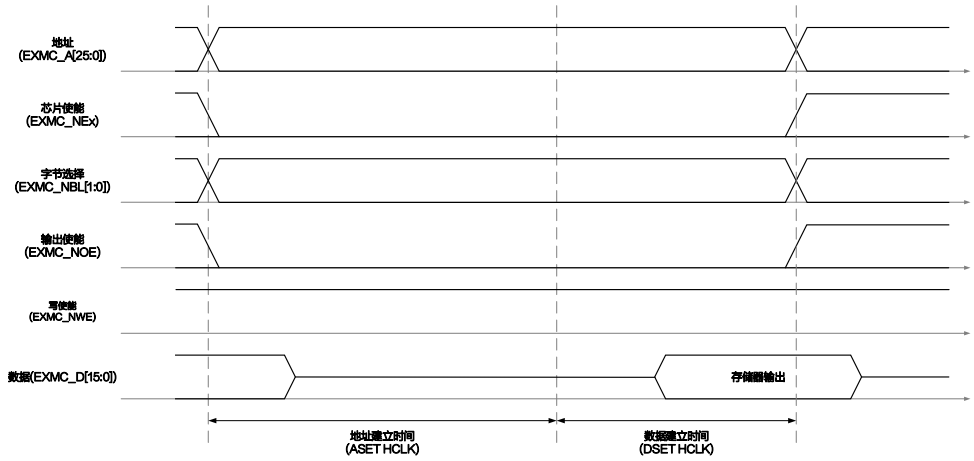



图 25-8. 模式 1 写时序

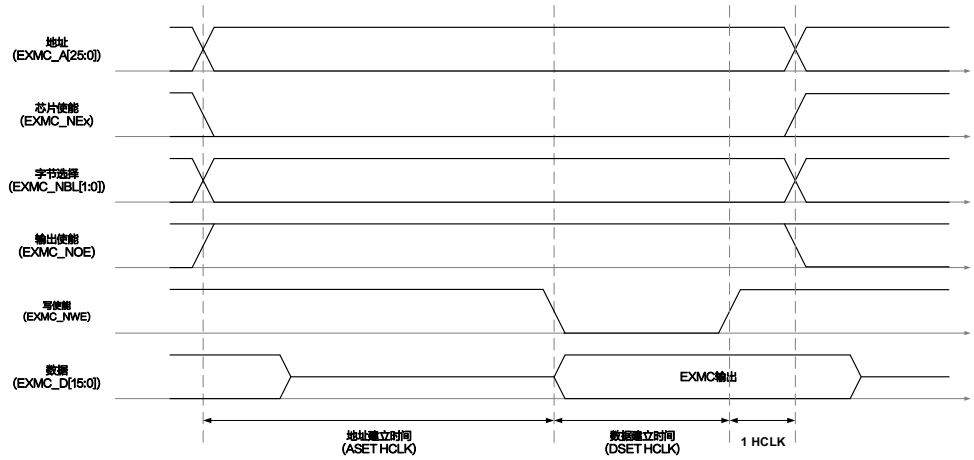



表 25-8. 模式 1 相关寄存器配置

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| EXMC_SNCTLx | | |
| 31-21 | 保留 | 0x000 |
| 20 | CCK | 取决于存储器与用户 |
| 19 | SYNCWR | 0x0 |
| 18-16 | CPS | 取决于存储器 |
| 15 | ASYNCWTEN | 取决于存储器 |
| 14 | EXMODEN | 0x0 |
| 13 | NRWTEN | 0x0 |
| 12 | WEN | 取决于用户 |
| 11 | NRWTCFG | 无影响 |
| 10 | WRAPEN | 0x0 |
| 9 | NRWTPOL | 仅当位 15 为 1 时有效 |
| 8 | SBRSTEN | 0x0 |
| 7 | 保留 | 0x1 |
| 6 | NREN | 无影响 |

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| **EXMC_SNCTLx** | | |
| 5-4 | NRW | 取决于存储器 |
| 3-2 | NRTP | 取决于存储器，Nor Flash：2 |
| 1 | NRMUX | 0x0 |
| 0 | NRBKEN | 0x1 |
| **EXMC_SNTCFGx** | | |
| 31-30 | 保留 | 0x0000 |
| 29-28 | ASYNCMOD | 无影响 |
| 27-24 | DLAT | 无影响 |
| 23-20 | CKDIV | 无影响 |
| 19-16 | BUSLAT | EXMC_NE[x]上升沿到下降沿的时间 |
| 15-8 | DSET | 取决于存储器与用户(写操作为 DSET+1 HCLK 时钟周期，读操作为 DSET HCLK 时钟周期) |
| 7-4 | AHLD | 无影响 |
| 3-0 | ASET | 取决于存储器与用户 |

模式A - SRAM/PSRAM(CRAM) OE翻转

**图 25-9. 模式 A 读时序**

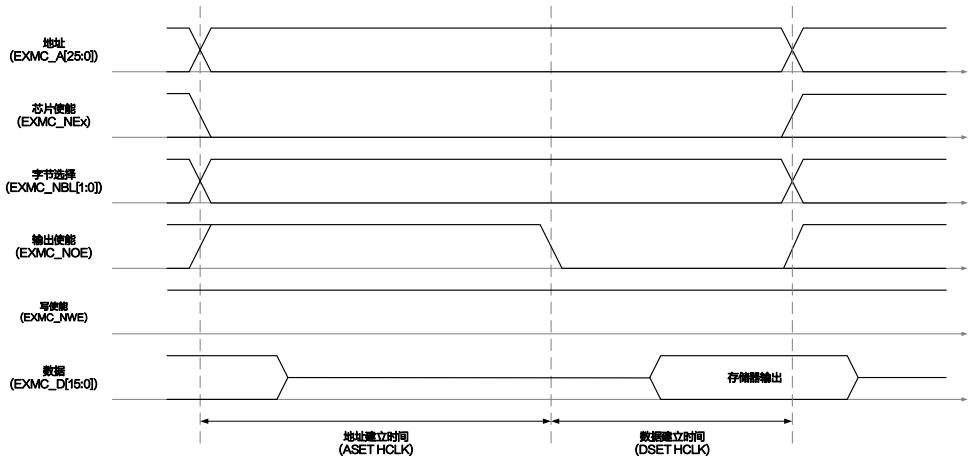
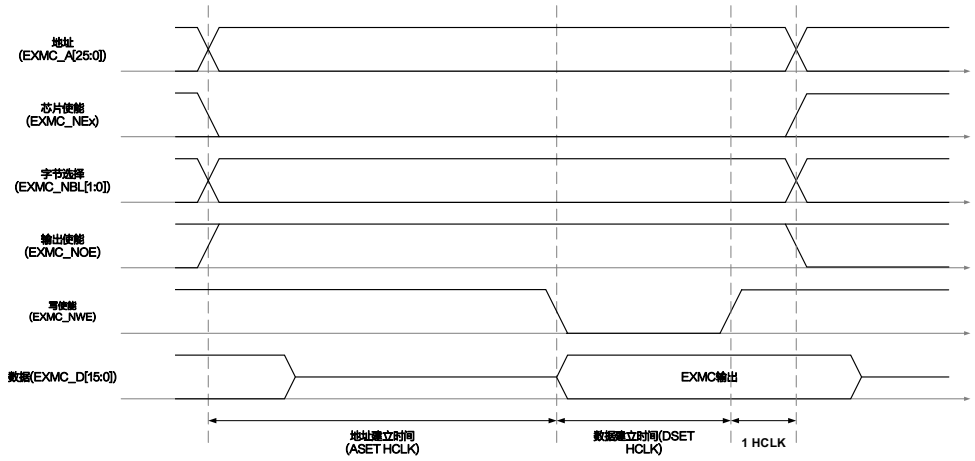


图 25-10. 模式 A 写时序




模式A和模式1的区别在于写时序，当两个模式的寄存器有相同的时序配置时，模式A的写时序独立于读时序。

表 25-9. 模式 A 相关寄存器配置

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| **EXMC_SNCTLx** | | |
| 31-21 | 保留 | 0x000 |
| 20 | CCK | 取决于存储器与用户 |
| 19 | SYNCWR | 0x0 |
| 18-16 | 保留 | 0x0 |
| 15 | ASYNCWTEN | 取决于存储器 |
| 14 | EXMODEN | 0x1 |
| 13 | NRWTEN | 0x0 |
| 12 | WEN | 取决于用户 |
| 11 | NRWTCFG | 无影响 |
| 10 | WRAPEN | 0x0 |
| 9 | NRWTPOL | 仅当位 15 为 1 时有效 |
| 8 | SBRSTEN | 0x0 |
| 7 | 保留 | 0x1 |
| 6 | NREN | 无影响 |
| 5-4 | NRW | 取决于存储器 |
| 3-2 | NRTP | 取决于存储器，Nor Flash：2 |
| 1 | NRMUX | 0x0 |
| 0 | NRBKEN | 0x1 |
| **EXMC_SNTCFGx(读)** | | |
| 31-30 | 保留 | 0x0 |
| 29-28 | ASYNCMOD | 0x0 |
| 27-24 | DLAT | 无影响 |
| 23-20 | CKDIV | 无影响 |
| 19-16 | BUSLAT | EXMC_NE[x]上升沿到下降沿的时间 |

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| 15-8 | DSET | 取决于存储器与用户(读操作为 DSET HCLK 时钟周期) |
| 7-4 | AHLD | 无影响 |
| 3-0 | ASET | 取决于存储器与用户 |
| **EXMC_SNWTCFGx(写)** | | |
| 31-30 | 保留 | 0x0 |
| 29-28 | WASYNCMOD | 0x0 |
| 27-20 | 保留 | 0x00 |
| 19-16 | WBUSLAT | EXMC_NE[x]上升沿到下降沿的时间 |
| 15-8 | WDSET | 取决于存储器与用户(写操作为 WDSET+1 HCLK 时钟周期) |
| 7-4 | WAHLD | 0x0 |
| 3-0 | WASET | 取决于存储器与用户 |

模式2/B - NOR Flash

**图 25-11. 模式 2/B 读时序**

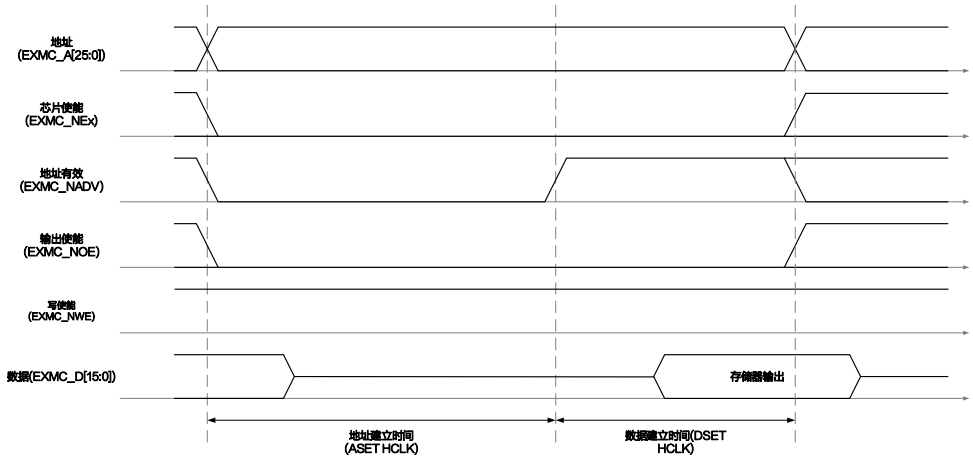



**图 25-12. 模式 2 写时序**

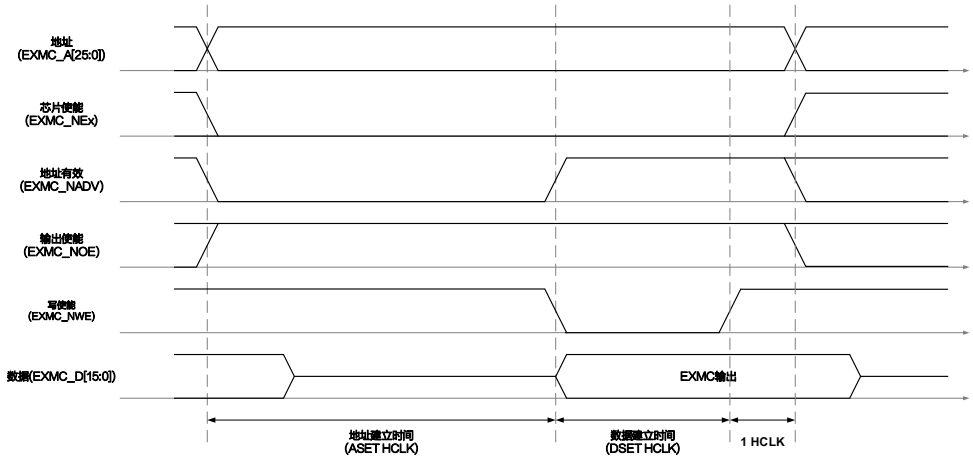


图 25-13. 模式 B 写时序

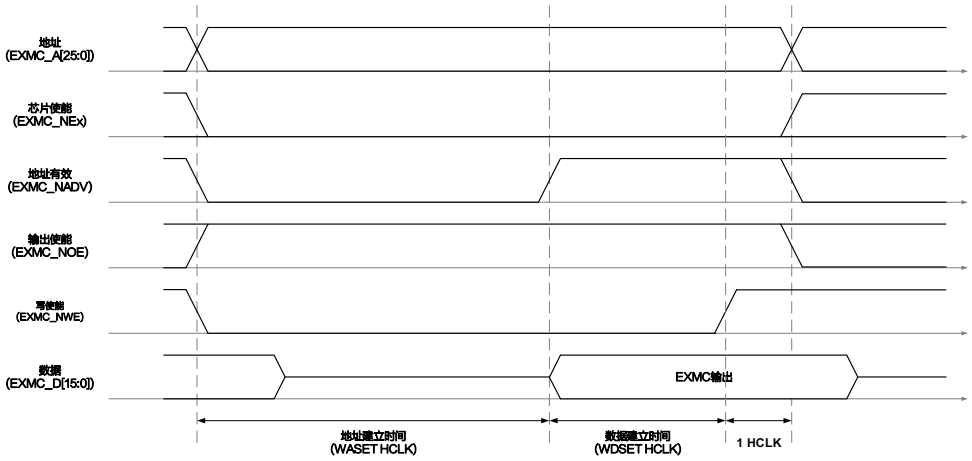



表 25-10. 模式 2/B 相关寄存器配置

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| **EXMC_SNCTLx(模式 2，模式 B)** | | |
| 31-21 | 保留 | 0x000 |
| 20 | CCK | 取决于存储器与用户 |
| 19 | SYNCWR | 0x0 |
| 18-16 | 保留 | 0x0 |
| 15 | ASYNCWTEN | 取决于存储器 |
| 14 | EXMODEN | 模式 2：0x0，模式 B：0x1 |
| 13 | NRWTEN | 0x0 |
| 12 | WEN | 取决于用户 |
| 11 | NRWTCFG | 无影响 |
| 10 | WRAPEN | 0x0 |
| 9 | NRWTPOL | 仅当位 15 为 1 时有效 |
| 8 | SBRSTEN | 0x0 |
| 7 | 保留 | 0x1 |
| 6 | NREN | 0x1 |
| 5-4 | NRW | 取决于存储器 |
| 3-2 | NRTP | Nor Flash：2 |
| 1 | NRMUX | 0x0 |
| 0 | NRBKEN | 0x1 |
| **EXMC_SNTCFGx(模式 2 读/写操作，模式 B 读操作)** | | |
| 31-30 | 保留 | 0x0000 |
| 29-28 | ASYNCMOD | 模式 B：0x1 |
| 27-24 | DLAT | 无影响 |
| 23-20 | CKDIV | 无影响 |
| 19-16 | BUSLAT | EXMC_NE[x]上升沿到下降沿的时间 |
| 15-8 | DSET | 取决于存储器与用户(读操作为 DSET HCLK 时钟周期) |
| 7-4 | AHLD | 0x0 |

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| 3-0 | ASET | 取决于存储器与用户 |
| **EXMC_SNWTCFGx(模式 B 写操作)** | | |
| 31-30 | 保留 | 0x0000 |
| 29-28 | WASYNCMOD | 模式 B：0x1 |
| 27-20 | 保留 | 0x000 |
| 19-16 | WBUSLAT | EXMC_NE[x]上升沿到下降沿的时间 |
| 15-8 | WDSET | 取决于存储器与用户(写操作为 WDSET+1 HCLk 时钟周期) |
| 7-4 | WAHLD | 0x0 |
| 3-0 | WASET | 取决于存储器与用户 |

模式C - NOR Flash OE翻转

**图 25-14. 模式 C 读时序**

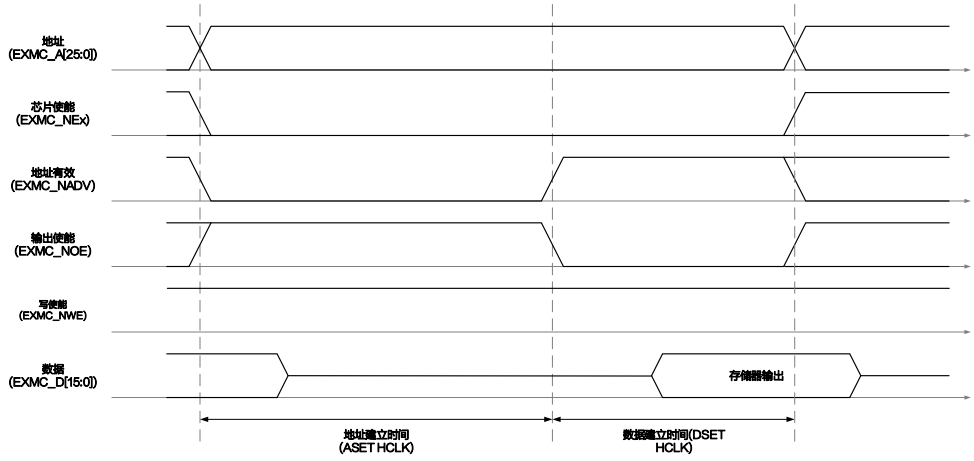
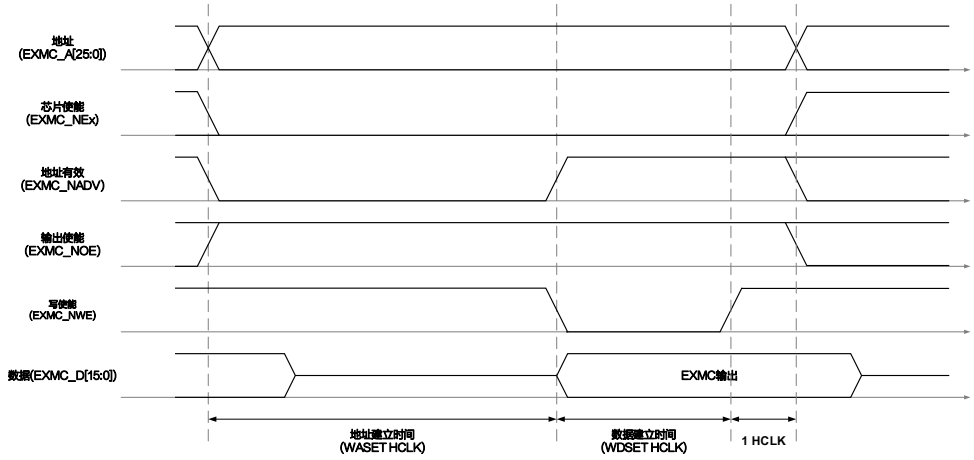



**图 25-15. 模式 C 写时序**




模式C和模式1的区别在于写时序，当两个模式的寄存器有相同的时序配置时，模式C的写时序独立于读时序。

表 25-11. 模式 C 相关寄存器配置

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| **EXMC_SNCTLx** | | |
| 31-21 | 保留 | 0x000 |
| 20 | CCK | 取决于存储器与用户 |
| 19 | SYNCWR | 0x0 |
| 18-16 | 保留 | 0x0 |
| 15 | ASYNCWTEN | 取决于存储器 |
| 14 | EXMODEN | 0x1 |
| 13 | NRWTEN | 0x0 |
| 12 | WEN | 取决于用户 |
| 11 | NRWTCFG | 无影响 |
| 10 | WRAPEN | 0x0 |
| 9 | NRWTPOL | 仅当位 15 为 1 时有效 |
| 8 | SBRSTEN | 0x0 |
| 7 | 保留 | 0x1 |
| 6 | NREN | 0x1 |
| 5-4 | NRW | 取决于存储器 |
| 3-2 | NRTP | Nor Flash：2 |
| 1 | NRMUX | 0x0 |
| 0 | NRBKEN | 0x1 |
| **EXMC_SNTCFGx(读)** | | |
| 31-30 | 保留 | 0x0000 |
| 29-28 | ASYNCMOD | 模式 C：0x2 |
| 27-24 | DLAT | 0x0 |
| 23-20 | CKDIV | 0x0 |
| 19-16 | BUSLAT | EXMC_NE[x]上升沿到下降沿的时间 |
| 15-8 | DSET | 取决于存储器与用户读操作为(DSET HCLk 时钟周期) |
| 7-4 | AHLD | 0x0 |
| 3-0 | ASET | 取决于存储器与用户 |
| **EXMC_SNWTCFGx(写)** | | |
| 31-30 | 保留 | 0x0 |
| 29-28 | WASYNCMOD | 模式 C：0x2 |
| 27-20 | 保留 | 0x000 |
| 19-16 | WBUSLAT | EXMC_NE[x]上升沿到下降沿的时间 |
| 15-8 | WDSET | 取决于存储器与用户(写操作为 WDSET+1 HCLK 时钟周期) |
| 7-4 | WAHLD | 0x0 |
| 3-0 | WASET | 取决于存储器与用户 |

模式D - 带地址扩展的异步操作

图 25-16. 模式 D 读时序

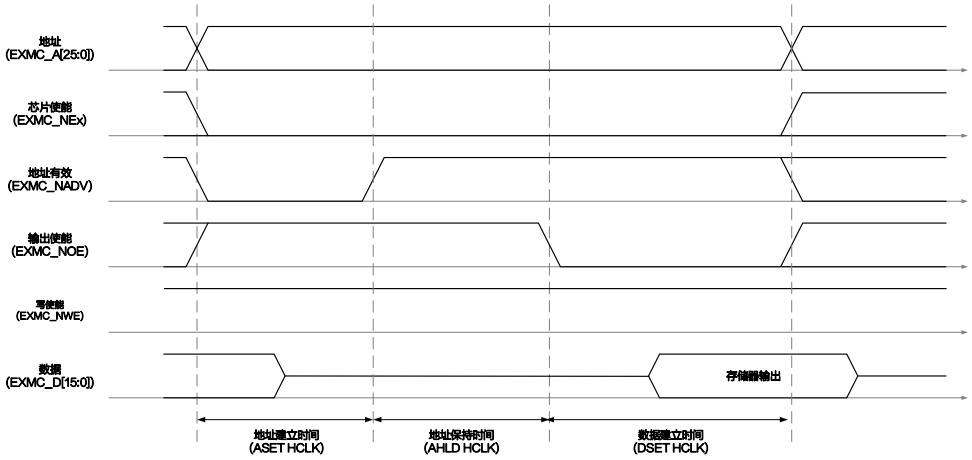



图 25-17. 模式 D 写时序

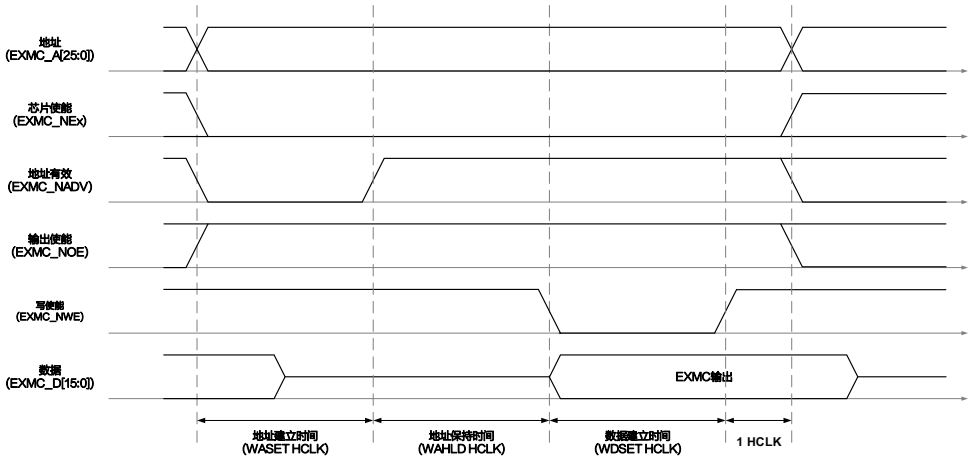



表 25-12. 模式 D 相关寄存器配置

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| **EXMC_SNCTLx** | | |
| 31-21 | 保留 | 0x000 |
| 20 | CCK | 取决于存储器与用户 |
| 19 | SYNCWR | 0x0 |
| 18-16 | 保留 | 0x0 |
| 15 | ASYNCWTEN | 取决于存储器 |
| 14 | EXMODEN | 0x1 |
| 13 | NRWTEN | 0x0 |
| 12 | WEN | 取决于用户 |
| 11 | NRWTCFG | 无影响 |
| 10 | WRAPEN | 0x0 |
| 9 | NRWTPOL | 仅当位 15 为 1 时有效 |
| 8 | SBRSTEN | 0x0 |
| 7 | 保留 | 0x1 |
| 6 | NREN | 取决于存储器 |

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| 5-4 | NRW | 取决于存储器 |
| 3-2 | NRTP | 取决于存储器 |
| 1 | NRMUX | 0x0 |
| 0 | NRBKEN | 0x1 |
| **EXMC_SNTCFGx(读)** | | |
| 31-30 | 保留 | 0x0 |
| 29-28 | ASYNCMOD | 模式 D：0x3 |
| 27-24 | DLAT | 无关 |
| 23-20 | CKDIV | 无影响 |
| 19-16 | BUSLAT | EXMC_NE[x]上升沿到下降沿的时间 |
| 15-8 | DSET | 取决于存储器与用户(读操作为 DSET HCLK 时钟周期) |
| 7-4 | AHLD | 取决于存储器与用户 |
| 3-0 | ASET | 取决于存储器与用户 |
| **EXMC_SNWTCFGx(写)** | | |
| 31-30 | 保留 | 0x0 |
| 29-28 | WASYNCMOD | 模式 D：0x3 |
| 27-20 | 保留 | 0x000 |
| 19-16 | WBUSLAT | EXMC_NE[x]上升沿到下降沿的时间 |
| 15-8 | WDSET | 取决于存储器与用户(写操作为 WSET+1 HCLk 时钟周期) |
| 7-4 | WAHLD | 取决于存储器与用户 |
| 3-0 | WASET | 取决于存储器与用户 |

模式M - NOR Flash地址/数据总线复用

**图 25-18. 复用模式读时序**

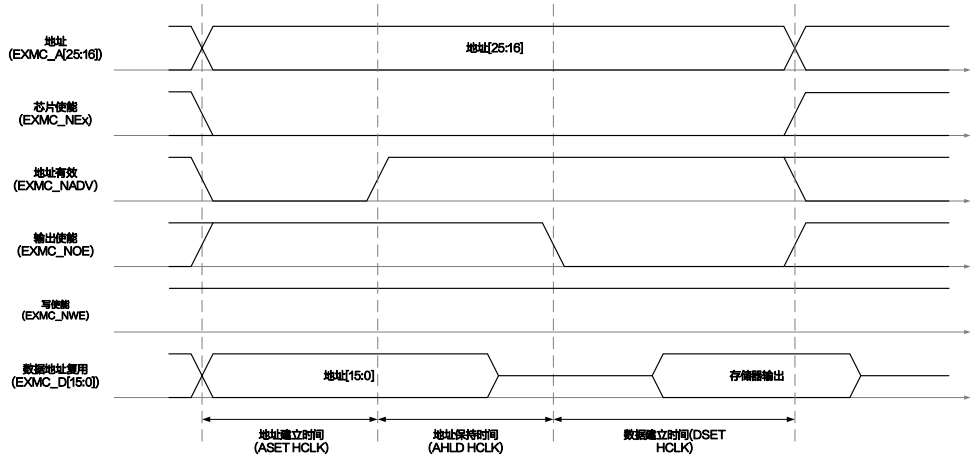


图 25-19. 复用模式写时序

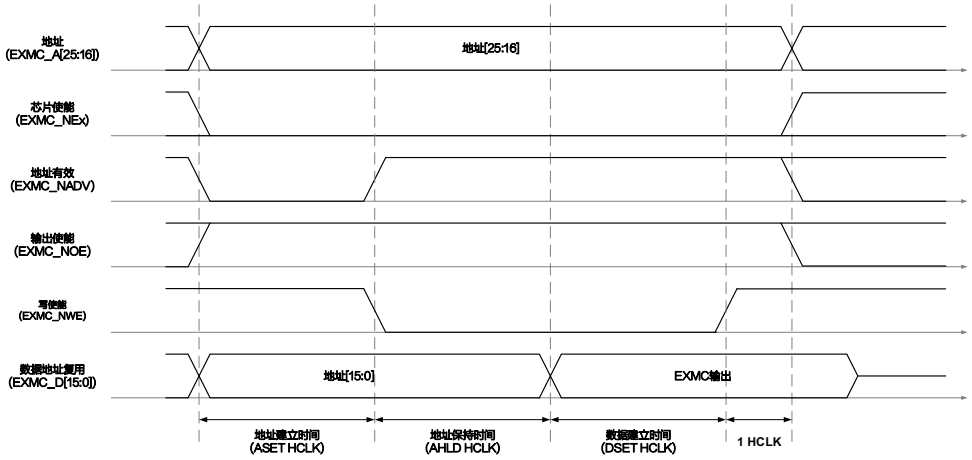



表 25-13. 复用模式相关寄存器配置

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| **EXMC_SNCTLx** | | |
| 31-21 | 保留 | 0x000 |
| 20 | CCK | 取决于存储器 |
| 19 | SYNCWR | 0x0 |
| 18-16 | 保留 | 0x0 |
| 15 | ASYNCWTEN | 取决于存储器 |
| 14 | EXMODEN | 0x0 |
| 13 | NRWTEN | 0x0 |
| 12 | WEN | 取决于存储器 |
| 11 | NRWTCFG | 无影响 |
| 10 | WRAPEN | 0x0 |
| 9 | NRWTPOL | 仅当位 15 为 1 时有效 |
| 8 | SBRSTEN | 0x0 |
| 7 | 保留 | 0x1 |
| 6 | NREN | 0x1 |
| 5-4 | NRW | 取决于存储器 |
| 3-2 | NRTP | Nor Flash：2 |
| 1 | NRMUX | 0x1 |
| 0 | NRBKEN | 0x1 |
| **EXMC_SNTCFGx** | | |
| 31-30 | 保留 | 0x0 |
| 29-28 | ASYNCMOD | 0x0 |
| 27-24 | DLAT | 无影响 |
| 23-20 | CKDIV | 无影响 |
| 19-16 | BUSLAT | EXMC_NE[x]上升沿到下降沿的时间 |
| 15-8 | DSET | 取决于存储器与用户(写操作为 DSET HCLK 时钟周期，读操作为 DSET+1 HCLK 时钟周期) |
| 7-4 | AHLD | 取决于存储器与用户 |

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| 3-0 | ASET | 取决于存储器与用户 |

异步通信的等待时间：

等待功能由寄存器EXMC_SNCTL位ASYNCWTEN控制。在访问外部存储器期间，若使能异步等待功能(ASYNCWTEN=1)，数据建立时间将会自动延长。延长时间的计算如下：

若存储器等待信号与EXMC_NOE/ EXMC_NWE信号对齐：

$$T_{DATA_SETUP} >= maxT_{WAIT_ASSERTION} + 4HCLK \quad (25\text{-}1)$$

若存储器等待信号与EXMC_NE信号对齐：

如果

$$maxT_{WAIT_ASSERTION} >= T_{ADDRES_PHASE} + T_{HOLD_PHASE} \quad (25\text{-}2)$$

则

$$T_{DATA_SETUP} >= (maxT_{WAIT_ASSERTION} - T_{ADDRES_PHASE} - T_{HOLD_PHASE}) + 4HCLK \quad (25\text{-}3)$$

否则

$$T_{DATA_SETUP} >= 4HCLK \quad (25\text{-}4)$$

**图 25-20. 异步等待有效时的读时序**

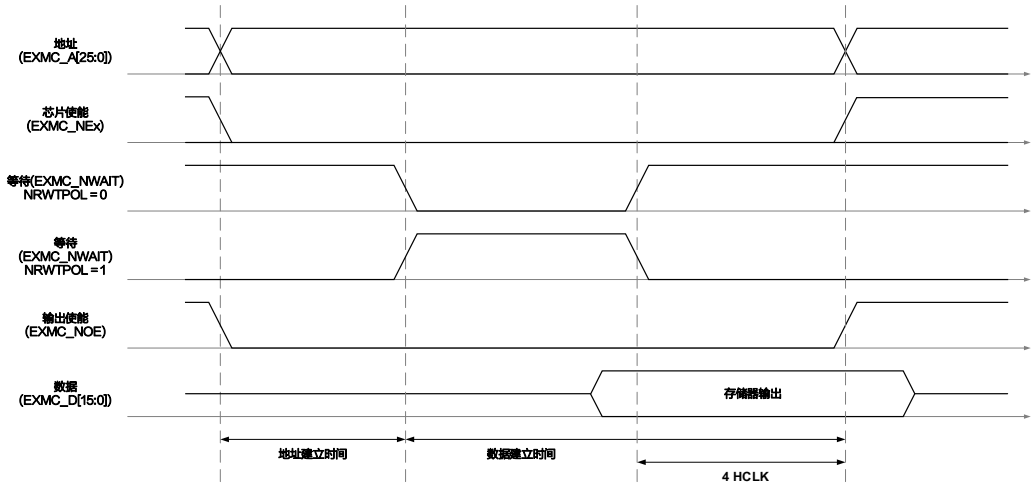
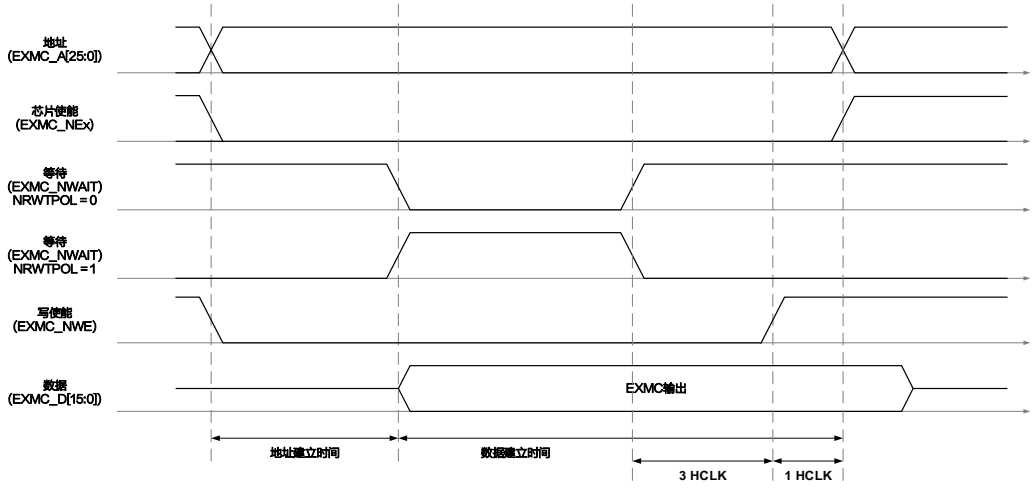


**图 25-21. 异步等待有效时的写时序**




### 同步访问时序

存储器时钟(EXMC_CLK)与系统时钟(HCLK)关系如下：

$$EXMC_CLK=\frac{HCLK}{CKDIV+1} \tag{25-5}$$

其中CKDIV是同步时钟分频比，通过配置寄存器EXMC_SNTCFGx中的CKDIV位来设置不同的值。

1. 数据延迟与 NOR Flash 延迟

数据延迟 DLAT 是指在采样数据之前需要等待的 EXMC_CLK 周期数。它和 NOR 闪存延迟的关系如下：

NOR闪存延迟不包含EXMC_NADV，二者之间的关系为：

$$NOR\ 闪存延迟=DLAT+2 \tag{25-6}$$

NOR闪存延迟包含EXMC_NADV，二者之间的关系为：

$$NOR\ 闪存延迟=DLAT+3 \tag{25-7}$$

2. 数据等待

用户需要保证 EXMC_NWAIT 信号与外部设备一致。该信号通过寄存器 EXMC_SNCTLx 来设置，位 NRWTEN 使能，位 NRWTCFG 决定 EXMC_NWAIT 信号是等待状态同时有效，或者比等待状态提前一个时钟周期有效，位 NRWTPOL 设置 EXMC_NWAIT 信号极性。

在 NOR Flash 的同步突发模式中，当寄存器 EXMC_SNCTLx 位 NRWTEN 置 1，在数据延迟之后后检测到 EXMC_NWAIT 信号。如果 EXMC_NWAIT 有效，在 EXMC_NWAIT 无效之前会一直插入等待时钟。

EXMC_NWAIT有效极性：

NRWTPOL = 1，EXMC_NWAIT 高电平有效

NRWTPOL = 0，EXMC_NWAIT 低电平有效

在同步突发模式中，EXMC_NWAIT 信号有两种配置：

NRWTCFG = 1，EXMC_NWAIT 信号有效时，当前时钟周期数据无效

NRWTCFG = 0，EXMC_NWAIT 信号有效时，下一个时钟周期数据无效，这是复位后的默认配置。

在 EXMC_NWAIT 信号有效的等待周期内，EXMC 会持续的给存储器发送时钟信号，保持片选和输出使能有效，并且忽视总线上的无效数据。

3. CRAM 页边界突发传输的自动分组

CRAM1.5 中禁止突发传输跨越页边界，EXMC 遇到边界会进行传输的自动分组。为了保证正确的突发分组操作，用户需要在寄存器 EXMC_SNCTLx 位 CPS 中需要设定 CRAM 的页大小。

4. 模式 SM - 单次突发传输

对于同步突发传输，如果 AHB 需要的数据为 16 位，则 EXMC 会执行一次长度为 1 的成组传输；如果 AHB 需要的数据为 32 位，则 EXMC 会把这次传输分成 2 次 16 位的传输，即执行一次长度为 2 的突发传输。

同步复用突发读时序 - NOR,PSRAM(CRAM)

**图 25-22. 同步复用突发传输读时序**

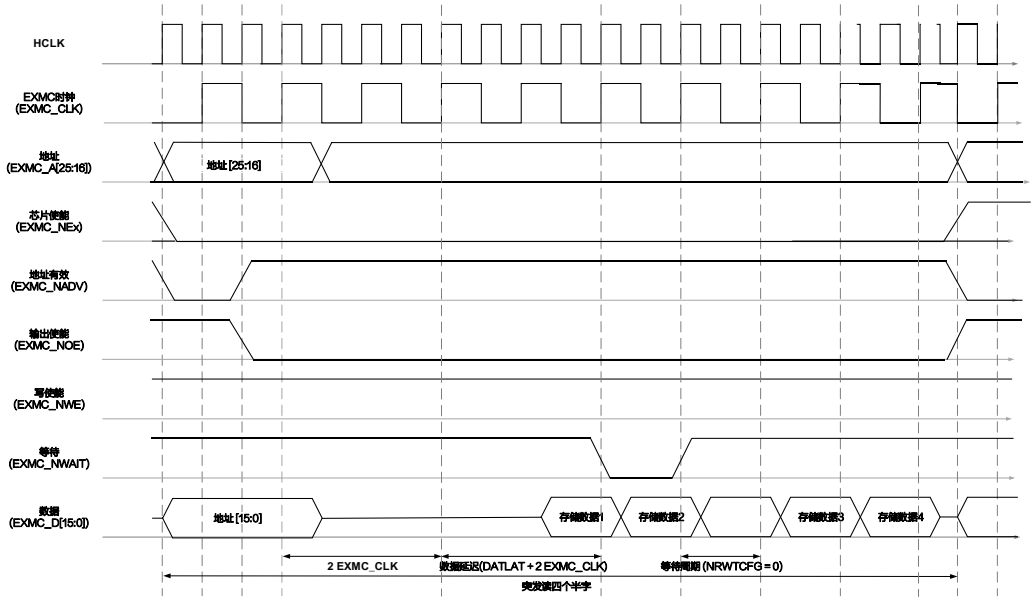



**表 25-14. 同步复用模式读时序配置**

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| **EXMC_SNCTLx** | | |
| 31-21 | 保留 | 0x000 |
| 20 | CCK | 取决于存储器 |
| 19 | SYNCWR | 无影响 |
| 18-16 | 保留 | 0x0 |

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| 15 | ASYNCWTEN | 0x0 |
| 14 | EXMODEN | 0x0 |
| 13 | NRWTEN | 取决于存储器 |
| 12 | WEN | 无影响 |
| 11 | NRWTCFG | 取决于存储器 |
| 10 | WRAPEN | 0x0 |
| 9 | NRWTPOL | 取决于存储器 |
| 8 | SBRSTEN | 0x1，突发读使能 |
| 7 | 保留 | 0x1 |
| 6 | NREN | 取决于存储器 |
| 5-4 | NRW | 0x1 |
| 3-2 | NRTP | 取决于存储器，0x1/0x2 |
| 1 | NRMUX | 0x1，取决于存储器与用户 |
| 0 | NRBKEN | 0x1 |
| **EXMC_SNTCFGx(读)** | | |
| 31-30 | 保留 | 0x0 |
| 29-28 | ASYNCMOD | 0x0 |
| 27-24 | DLAT | 数据延迟 |
| 23-20 | CKDIV | 上图设置：0x1，EXMC_CLK=2HCLK |
| 19-16 | BUSLAT | EXMC_NE[x]上升沿到下降沿的时间 |
| 15-8 | DSET | 无影响 |
| 7-4 | AHLD | 无影响 |
| 3-0 | ASET | 无影响 |

同步复用突发写时序 - NOR,PSRAM(CRAM)

**图 25-23. 同步复用突发传输写时序**

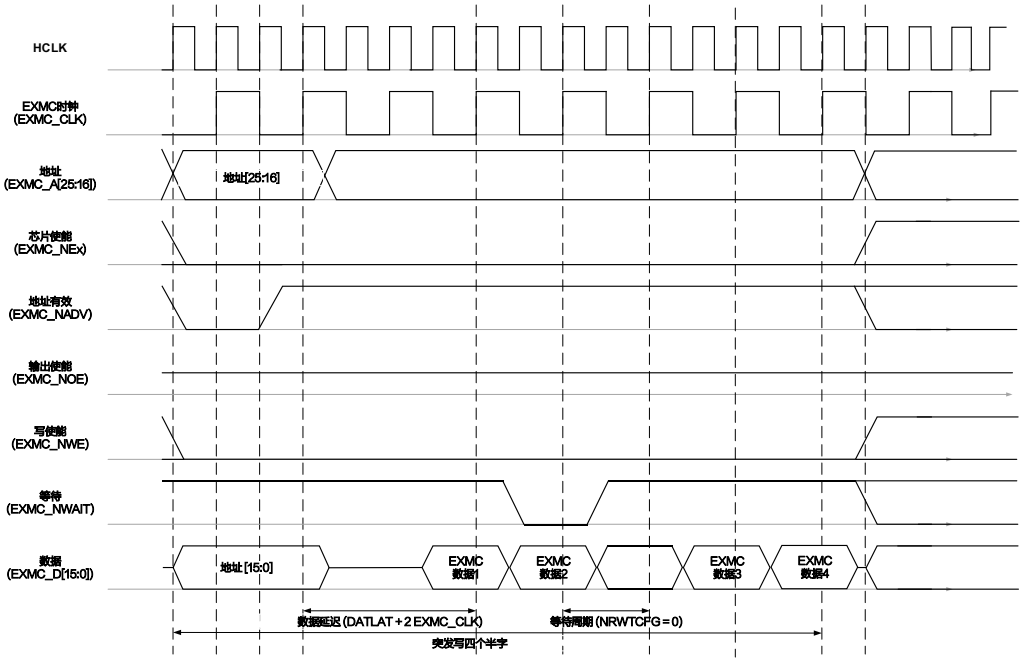


表 25-15. 同步复用模式写时序配置

| 位域/位 | 名称 | 参考设定值 |
|---|---|---|
| EXMC_SNCTLx | | |
| 31-21 | 保留 | 0x000 |
| 20 | CCK | 取决于存储器 |
| 19 | SYNCWR | 0x1，同步写使能 |
| 18-16 | 保留 | 0x0 |
| 15 | AYSNCWAIT | 0x0 |
| 14 | EXMODEN | 0x0 |
| 13 | NRWTEN | 取决于存储器 |
| 12 | WEN | 0x1 |
| 11 | NRWTCFG | 0x0(这里必须为 0) |
| 10 | WRAPEN | 0x0 |
| 9 | NTWTPOL | 取决于存储器 |
| 8 | SBRSTEN | 无影响 |
| 7 | 保留 | 0x1 |
| 6 | NREN | 取决于存储器 |
| 5-4 | NRW | 0x1 |
| 3-2 | NRTP | 0x1 |
| 1 | NRMUX | 0x1，取决于用户 |
| 0 | NRBKEN | 0x1 |
| EXMC_SNTCFGx(写) | | |
| 31-30 | 保留 | 0x0 |
| 29-28 | ASYNCMOD | 0x0 |
| 27-24 | DLAT | 数据延迟 |
| 23-20 | CKDIV | 上图设置：0x1，EXMC_CLK=2HCLK |
| 19-16 | BUSLAT | EXMC_NE[x]上升沿到下降沿的时间 |
| 15-8 | DSET | 无影响 |
| 7-4 | AHLD | 无影响 |
| 3-0 | ASET | 无影响 |

### SPI/QPI-PSRAM 访问时序

SPI/QPI-PSRAM 由 EXMC 的 Bank0，Region0 操作，它是具有 SPI 和 QPI 接口的 PSRAM。具有 6 个 IO 口，包含片选，时钟和 4 个数据 IO 口，详细见*表 25-16. SPI/QPI 接口*。

表 25-16. SPI/QPI 接口

| 信号 | 传输方向 | SPI 模式 | QPI 模式 |
|---|---|---|---|
| EXMC_CLK | 输出 | 串行时钟 | |
| EXMC_NE[0] | 输出 | 芯片使能(低有效) | |
| EXMC_D[0] | 输入/输出 | 串行输出 | 数据 IO[0] |
| EXMC_D[1] | 输入/输出 | 串行输入 | 数据 IO[1] |
| EXMC_D[2] | 输入/输出 | X | 数据 IO[2] |
| EXMC_D[3] | 输入/输出 | X | 数据 IO[3] |

1. 控制器初始化

第一步，用户需要设置 SPI 初始化寄存器 EXMC_SINIT。由 POL 位设置数据采样时钟的有效边沿，IDL 位设置设备 ID 长度，ADRBIT 位设置地址长度，CMDBIT 位设置命令长度。

2. 读写操作

EXMC 支持 SPI，QPI 和 SQPI 三种访问模式，访问模式必须在读写操作之前设置。读写的访问模式由 RMODE 和 WMODE 设置，等待周期由 RWAITCYCLE 和 WWAITCYCLE 位设置，操作指令由 RCMD 和 WCMD 设置。这些位都在 EXMC_SRCMD 和 EXMC_SWCMD 寄存器。

在模式选择完毕以后，读写操作就和普通的 NOR Flash 一样，通过 EXMC 的 Bank0 的 Region 进行。

3. 读设备 ID

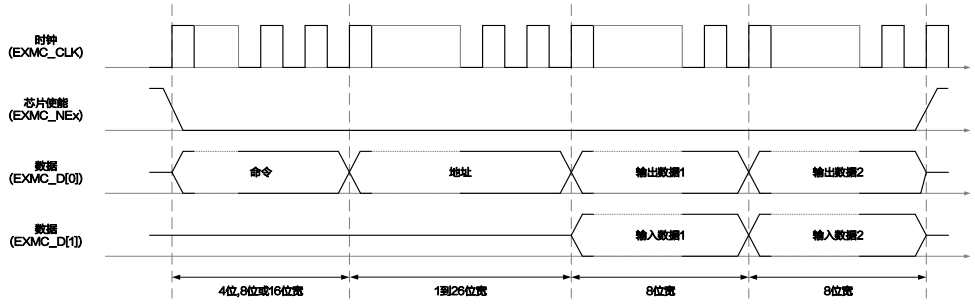
读设备 ID 是一个特殊的指令，需要查询 SC 位直到它为 0，然后把它置 1.然后低 32 位的 ID 会存储在 EXMC_SIDL 寄存器中，高 32 位 ID 会存储在寄存器 EXMC_SIDH 中。

4. SPI-PSRAM 访问时序

在 SPI 模式中，EXMC 可以使用 SPI 协议，通过 4 个 IO 口，时钟，片选和输入输出数据线与外部存储器通信。如*图 25-24. SPI-PSRAM 访问时序*所示，设置外部存储器模式的指令先通过输出数据线串行输出，然后是与 EXMC 设置有关的任意大小的地址，最后是读写的数据。读写操作时，写是通过数据输出线，读是通过数据输入线。

当 SADRBIT[4:0] = 24，CMDBIT[1:0] = 1 时，SPI-PSRAM 通信如下：

**图 25-24. SPI-PSRAM 访问时序**

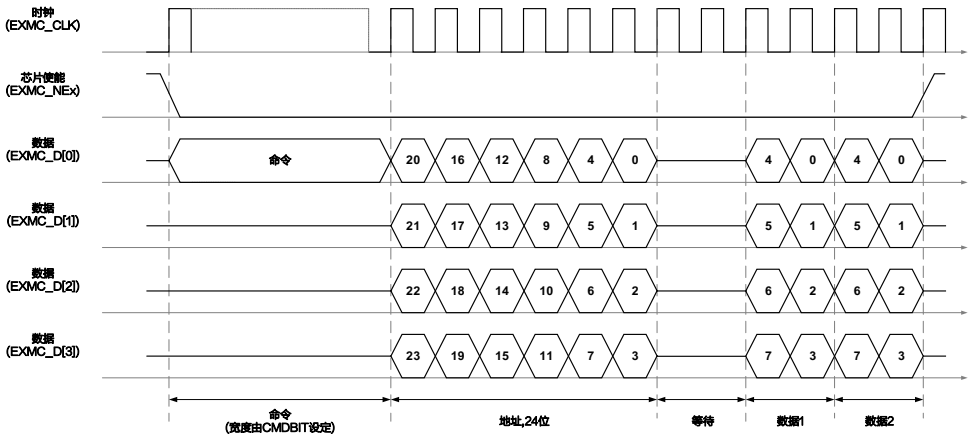



5. SQPI - PSRAM 访问时序

在 SQPI 模式中，EXMC 在指令阶段使用 SPI 协议，地址和数据阶段使用 QSPI 协议，通过 6 个 IO 口，时钟，片选和 4 个数据线与外部存储器通信。如*图 25-25. SQPI-PSRAM 访问时序*所示，设置外部存储器模式的指令先通过数据线 data[0]串行输出，之后是通过 4 个数据线并行输入/输出的地址和数据。

当 ADRBIT[4:0] = 24，CMDBIT[1:0] = 1(可以使用其他值)，RWAITCYCLE[3:0] = WWAITCYCLE[3:0] = 2(可以使用其他值)，SQPI-PSRAM 通信如下：

图 25-25. SQPI-PSRAM 访问时序

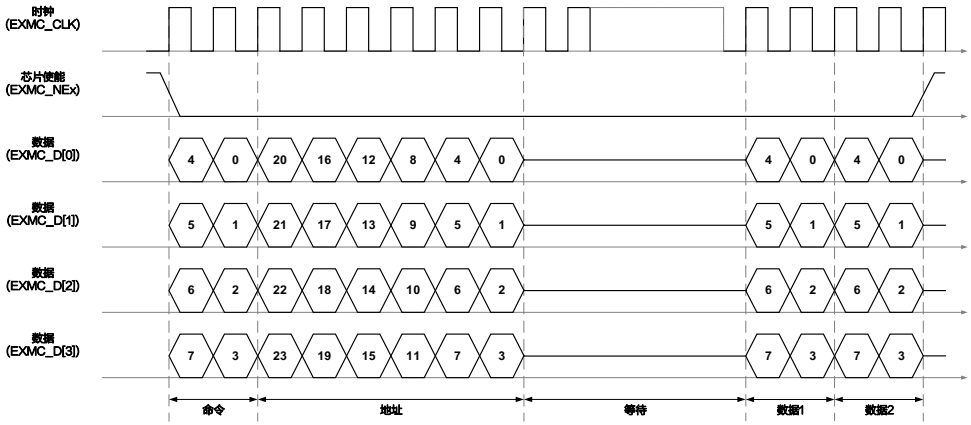



6. QPI - PSRAM 访问时序

QPI 与 SQPI 唯一的区别就是指令也是通过并行 4 个数据线输出的。

当 ADRBIT[4:0] = 24，CMDBIT[1:0] = 1，QPI-PSRAM 通信如下：

图 25-26. QPI-PSRAM 访问时序




## 25.3.5. NAND Flash 或 PC Card 控制器

EXMC模块Bank1、Bank2支持NAND FLASH，Bank3支持PC Card设备。对于每个Bank，EXMC提供独立的寄存器来配置访问时序，支持8位、16位的NAND FLASH以及16位PC卡。对于NAND FLASH，EXMC还提供ECC计算模块，保证数据传输和保存的鲁棒性。

### NAND Flash/PC Card 接口功能

表 25-17. 8 位/16 位 NAND 接口信号描述

| EXMC 引脚 | 传输方向 | 功能描述 |
|---|---|---|
| EXMC_A[17] | 输出 | NAND Flash 地址锁存(ALE) |
| EXMC_A[16] | 输出 | NAND Flash 命令锁存(CLE) |
| | 输入 /输出 | 8 位复用，双向地址/数据总线 |

| EXMC 引脚 | 传输方向 | 功能描述 |
|---|---|---|
| EXMC_D[7:0]/ EXMC_D[15:0] | | 16 位复用，双向地址/数据总线 |
| EXMC_NCE[x] | 输出 | 片选，x = 1,2 |
| EXMC_NOE(NRE) | 输出 | 输出使能 |
| EXMC_NWE | 输出 | 写使能 |
| EXMC_NWAIT/ EXMC_INT[x] | 输入 | NAND Flash 就绪/忙输入信号 EXMC，x=1,2 |

**表 25-18. 16 位 PC Card 接口信号描述**

| EXMC 引脚 | 传输方向 | 功能描述 |
|---|---|---|
| EXMC_A[10:0] | 输出 | 地址总线 |
| EXMC_NIORD | 输出 | I/O 空间输出使能 |
| EXMC_NIOWR | 输出 | I/O 空间写使能 |
| EXMC_NREG | 输出 | 决定访问通用空间还是属性空间 |
| EXMC_D[15:0] | 输入/输出 | 双向数据总线 |
| EXMC_NCE3_x | 输出 | 片选，x = 0,1 |
| EXMC_NOE | 输出 | 输出使能 |
| EXMC_NWE | 输出 | 写使能 |
| EXMC_NWAIT | 输入 | PC Card 等待信号 |
| EXMC_INTR | 输入 | PC Card 中断输入信号 |
| EXMC_CD | 输入 | PC Card 卡存在检测信号，高有效 |

## 支持的存储器访问模式

**表 25-19. Bank1/2/3 支持的访问模式**

| 存储器 | 模式 | 读/写 | AHB 传输宽度 | 注释 |
|---|---|---|---|---|
| 8 位 NAND | 异步 | R | 8 | |
| | 异步 | W | 8 | |
| | 异步 | R | 16 | 分成 2 次 EXMC 访问 |
| | 异步 | W | 16 | |
| | 异步 | R | 32 | 分成 4 次 EXMC 访问 |
| | 异步 | W | 32 | |
| 16 位 NAND/PC Card | 异步 | R | 8 | |
| | 异步 | W | 8 | 不支持此操作 |
| | 异步 | R | 16 | |
| | 异步 | W | 16 | |
| | 异步 | R | 32 | 分成 2 次 EXMC 访问 |
| | 异步 | W | 32 | |

### NAND Flash/PC Card 的控制时序

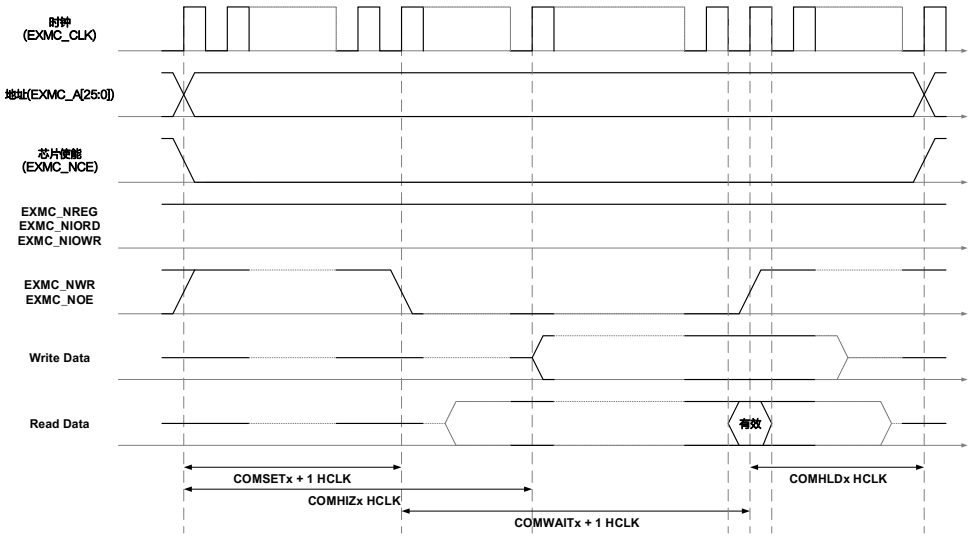
EXMC能够为NAND Flash、PC卡等设备产生合适的时序信号。每个Bank都有相应的寄存器来对外部存储器进行管理和控制，EXMC_NPCTLx、EXMC_NPINTENx、EXMC_NPCTCFGx、EXMC_NPATCFGx、EXMC_PIOTCFG3、EXMC_NECCx，其中寄存器EXMC_NPCTLx、EXMC_NPCTCFGx、EXMC_NPATCFGx都可以配置4个时序参数，可以根据用户需求和外部存储器的特性来进行相应的配置。

**表 25-20. NADN/PC Card 可编程参数**

| 参数 | 读/写 | 单位 | 功能描述 | NAND Flash/ PC Card | |
|---|---|---|---|---|---|
| | | | | 最小值 | 最大值 |
| 存储器数据总线高阻时间(HIZ) | W/R | HCLK | 启动写操作之后保持数据总线为高阻态的时间 | 1 | 255 |
| 存储器保持时间(HLD) | W/R | HCLK | 在发送命令结束后保持地址的时钟(HCLK)周期数目，写操作时也是数据的保持时间 | 1 | 254 |
| 存储器等待时间(WAIT) | W/R | HCLK | 发出命令的最短持续时间时钟(HCLK)周期数目 | 2 | 255 |
| 存储器建立时间(SET) | W/R | HCLK | 发出命令之前建立地址的(HCLK)时钟周期数目 | 1 | 256 |

*图 25-27. NAND/PC Card 通用空间操作时序*给出了在通用存储空间中操作的可编程参数定义，属性存储空间和 I/O 空间(只适用于 PC Card)中操作与此相似。

**图 25-27. NAND/PC Card 通用空间操作时序**



### NAND Flash 操作

EXMC在对NAND Flash发送命令或地址时，需要利用其命令锁存信号(A[16])或地址锁存信号(A[17])这两条地址线，即MCU需要在特定的地址进行写操作。

示例：NAND Flash读操作步骤：

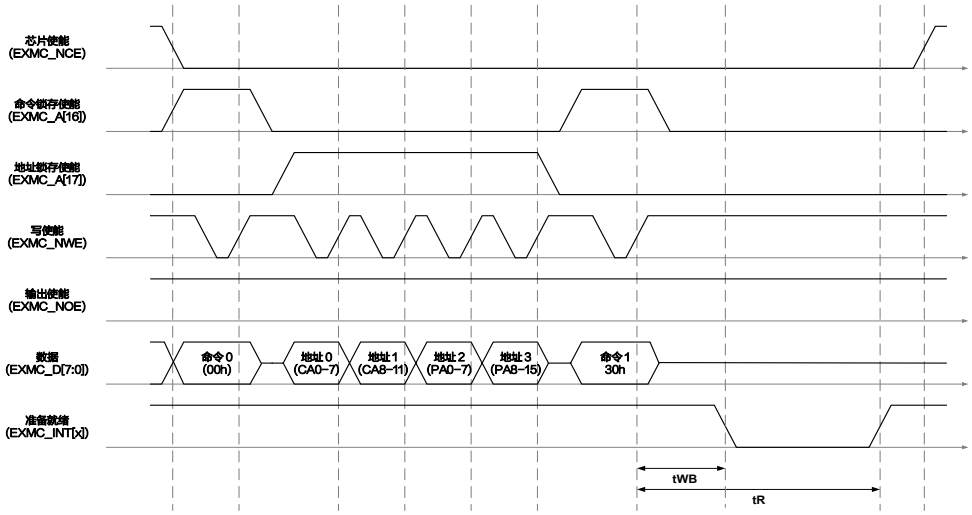
1) 配置 EXMC_NPCTLx、EXMC_NPCTCFGx，若需要预等待功能，还需配置EXMC_NPATCFGx；
2) 往通用空间写入 NAND Flash 读数据命令，即在 EXMC_NCE 和 EXMC_NWE 有效期间，EXMC_CLE(A[16])变为有效电平(高)，则被 NAND 认为写入命令；
3) 往通用空间写入读操作的起始地址，即在 EXMC_NCE 和 EXMC_NWE 有效期间，EXMC_ALE(A[17])变为有效电平(高)，则被 NAND 认为写入地址；
4) 等待 NAND 就绪信号，NAND 控制器会在这期间将和 EXMC_NCE 一直保持有效；
5) 从通用空间的数据区逐字节的读出数据；
6) 在不写入新的命令和地址，可以自动读出 NAND 下一页数据；或转到 3)写入新的地址进行下一页的读取；或转到 2)写入新的命令和地址。

### NAND Flash 预等待功能

某些NAND Flash要求在输入最后一个地址字节后，控制器等待NAND Flash就绪，并且还有一些对EXMC_NCE敏感型的NAND Flash还要求在其就绪前NCE必须保持有效。

下面以TOSHIBA128M*8bit NADN Flash为例：

**图 25-28. NCE 敏感 NAND Flash 访问时序**




1) 往 NAND 的通用空间命令区写入命令 CMD0
2) 往 NAND 的通用空间地址区写入操作地址 ADD0
3) 往 NAND 的通用空间地址区写入操作地址 ADD1
4) 往 NAND 的通用空间地址区写入操作地址 ADD2
5) 往 NAND 的通用空间地址区写入操作地址 ADD3
6) 往 NAND 的属性空间命令区写入命令 CMD1

在 6)中写命令操作，EXMC 使用的是寄存器 EXMC_NPATCFGx 定义的时序。经过 ATTHLD 时间后，NAND Flash 等待 EXMC_INTx 信号，ATTHLD 要大于 $t_{WB}$(EXMC_NWE 高到 EXMC_INTx 低)。

对于那些对片选信号敏感的 NAND Flash，在最后一个地址字节输入后，一直到 B/NB 就绪状态到来的这段时间中，要求片选信号 NCE 一直保持低电平。这里可以通过配置属性存储空间的 ATTHLD 的值来满足 $t_{WB}$ 的时序，这样 MCU 只有在写入最后一个地址字节时才使用属性存储空间的时序，而在其他时候都使用通用存储空间的时序。

### NAND Flash 的 ECC 计数模块

EXMC模块中的Bank1和Bank2各有一个ECC计算的硬件模块，用户可以根据EXMC_NPCTLx中的ECCSZ来选择ECC计算的页面大小，通过ECC计算可以矫正1个bit的错误并且能检测2个bit的错误。

当NAND存储器块使能，ECC模块就会检测D[15:0]以及EXMC_NCE、EXMC_NEW信号。当已经完成ECCSZ大小字节的读写操作时，软件必须读出EXMC_NECCx中的结果值。如果需要再次开始ECC计算，软件需要先将EXMC_NPCTLx中ECCEN清0来清除EXMC_NECCx中的值，再将ECCEN置1来重新启动ECC计算。

### PC/CF Card 访问

EXMC 的 Bank3 用来访问 PC/CF Card，同时支持存储器和 IO 模式。Bank3 分为 3 个子空间，分别为存储空间，属性空间和 IO 空间。

EXMC_NCE3_0 和 EXMC_NCE3_1 是字节选择信号，当仅有 EXMC_NCE3_0 有效时，低字节或高字节的选择取决于 EXMC_A[0]，当仅有 EXMC_NCE3_1 有效时，硬件不支持，当 EXMC_NCE3_0 和 EXMC_NCE3_1 都有效时，16 位操作。复位 NDTP 来选择 PC/CF Card 作为外部存储器，寄存器 EXMC_NPCTLx 位 NDW 必须设置为 01 来保证 EXMC 的正确操作。

下面是对不同空间的访问：

1. 通用空间：EXMC_NCE3_x(x= 0,1)是片选信号，表示同时支持 8 位和 16 位的访问操作。在 EXMC_NREG 位高电平时，EXMC_NWE 为低电平时写操作，EXMC_NOE 为低电平时读操作。
2. 属性空间：EXMC_NCE3_x(x= 0,1)是片选信号，表示同时支持 8 位和 16 位的访问操作。在 EXMC_NREG 位低电平时，EXMC_NWE 为低电平时写操作，EXMC_NOE 为低电平时读操作。
3. IO 空间：EXMC_NCE3_x(x= 0,1)是片选信号，表示同时支持 8 位和 16 位的访问操作。在 EXMC_NREG 位低电平时，EXMC_NIOWR 为低电平时写操作，EXMC_NIORD 为低电平时读操作。

AHB 访问 16 位的 PC/CF Card：

1. 通用空间：数据存储的位置，支持字节和半字访问，奇地址禁止字节访问。当 AHB 进行字访问，EXMC 会自动分成两次连续的半字操作。在 EXMC_NREG 位高电平时，EXMC_NWE 为低电平时写操作，EXMC_NOE 为低电平时读操作。
2. 属性空间：配置信息存储的位置，仅偶地址支持字节访问，半字访问会被转换为单次字节操作，字访问会被转换为两次字节访问。半字与字访问时，只有 EXMC_NCE3_0 有效。

在 EXMC_NREG 位低电平时，EXMC_NWE 为低电平时写操作，EXMC_NOE 为低电平时读操作。

4. IO 空间：同时支持字节和半字访问，EXMC_NREG 位低电平时，EXMC_NIOWR 为低电平时写操作，EXMC_NIORD 为低电平时读操作。

### 25.3.6. SDRAM 控制器

#### 主要特性

- 两个可独立配置的 SDRAM devices；
- 8 位，16 位，32 位数据带宽；
- 多达 13 位行地址、11 位列地址、2 位内部 bank 地址；
- 支持存储器大小：4x16Mx32bit(256 MB), 4x16Mx16bit (128 MB), 4x16Mx8bit (64 MB)；
- AHB 字、半字、字节访问；
- 为每个存储器 bank 提供独立的片选控制；
- 每个存储器 bank 可独立配置；
- 写使能和字节选择输出；
- 自动进行行和 bank 边界管理；
- 多个 bank 的乒乓访问；
- SDRAM 时钟可以为 fHCLK/2 或者 fHCLK/3；
- 可编程的时序参数；
- 可编程的刷新速率的自动刷新操作；
- 通过软件进行上电初始化；
- CAS 延迟可设置为 1、2、3 个时钟周期；
- 具有 16x35 位深度的写数据 FIFO；
- 具有 16x31 位深度的写地址 FIFO；
- 6x32 位深度的可缓存的读数据 FIFO；
- 6x14 位深度的可缓存读地址 FIFO；
- 可调整的读数据采样时钟；
- 自刷新模式；
- 掉电模式。

#### SDRAM 简介

同步动态随机存储器(SDRAM)是通过外部同步时钟刷新的动态随机存储器(DRAM)，它的同步时钟由EXMC的EXMC_SDCLK引脚提供，通过配置寄存器EXMC_SDCTLx位SDCLK时钟频率可设置为fhclk/2或者fhclk/3。指令和数据在时钟的上升沿锁存，在下降沿改变。

SDRAM内部分为多个叫做Bank的区域，允许设备以交错的方式进行访问，以获取更大的并发性和数据传输量。每个Bank可以认为是一个矩阵，其中每个地址对应存储器存储宽度的空间，矩阵由行和列构成，因此存储器的Bank大小可以认为是存储器数据宽度*行数*列数。用户可以通过设置寄存器EXMC_SDCTLx位NBK,SDW,RAW,CAW使EXMC可以与不同的SDRAM进行通信。

由于易失的本征特性，SDRAM需要周期性的刷新。EXMC支持两种刷新模式，自刷新和自动刷新。自刷新是在EXMC挂起的低功耗模式中使用，由SDRAM内部计数提供时钟，内部进行刷新。自动刷新是由EXMC周期性的提供刷新命令，因为此时SDRAM需要进行数据传输，刷新间隔由寄存器EXMC_SDARI位ARINTV决定，连续刷新次数由寄存器EXMC_SDCMD位NARF决定。刷新命令优先级高于其他的包括读写命令，来保证数据的正常存储，当SDRAM同时收到刷新命令与读写命令时，读写命令需要等待刷新命令完成才能进行。如果在前一个刷新命令未完成时，再次接收到刷新命令，寄存器EXMC_SDSTAT刷新错误标志位(REIF)会被置位，同时如果刷新错误中断使能(REIE)，将会发生刷新错误中断。

CAS延迟是读命令和数据线出现第一个可读数据之间的延迟，可以通过寄存器EXMC_SDCTLx位CL设置。

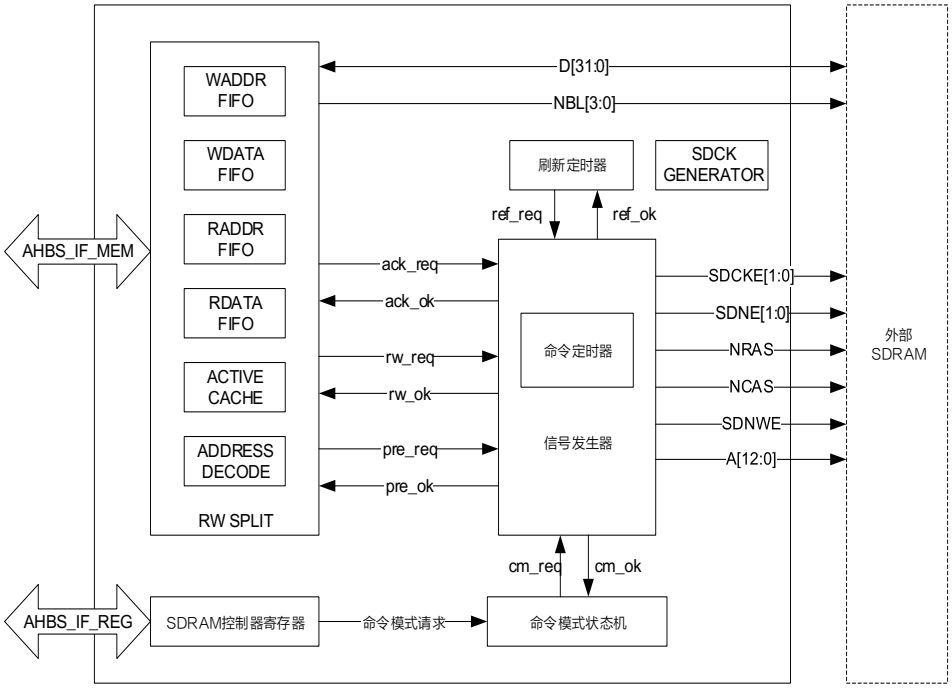
对不同的SDRAM需要参考其手册，使用模式寄存器进行设置，包含突发长度，突发类型，CAS延迟，写模式。在寄存器EXMC_SDCMD位MRC中设置，会通过CMD命令发送给SDRAM。在读写操作之前，需要发送读取模式寄存器命令，否则SDRAM无法工作。

### SDRAM 控制器简介

同步动态随机存储器控制器(SDRAMC)是MCU和SDRAM的接口。它把AHB的操作根据SDRAM协议转换为对SDRAM的操作，同时配置寄存器EXMC_SDTCFG满足时序要求。

SDRAMC 包含 4 个模块，读写预处理模块，控制寄存器，有限状态机和信号发生器。使用两组 FIFO 来提高存储器访问效率，一组用来写地址和数据，另外一组用来读地址数据。SDRAMC模块由*图 25-29. SDRAM 系统架构*所示。

**图 25-29. SDRAM 系统架构**




信号发生器处理状态机，刷新定时器，读写模块产生的请求。

命令定时器由遵守SDRAM时序协议的计数器组成。

SDRAM 命令由 SDRAM 控制器接口发出，可见*表 25-21. SDRAM 命令真值表*:

**表 25-21. SDRAM 命令真值表**

| SDNE | NRAS | NCAS | SDNWE | A[n] | A[10] | A[m] | 命令 |
|---|---|---|---|---|---|---|---|
| H | X | X | X | X | X | X | 命令禁止(无操作) |
| L | H | H | H | X | X | X | 无操作 |
| L | H | H | L | X | X | X | 中止突发传输 |
| L | H | L | H | Bank | L | Col | 突发读选择行 |
| L | H | L | H | Bank | H | Col | 预充电完成后，突发读选择行 |
| L | H | L | L | Bank | L | Col | 突发写选择行 |
| L | H | L | L | Bank | H | Col | 预充电完成后，突发写选择行 |
| L | L | H | H | Bank | Row | Row | 行使能命令，之后可进行读写 |
| L | L | H | L | Bank | L | X | 预充电命令，关闭当前 Bank 的选择行 |
| L | L | H | L | X | H | X | 全局预充电命令，关闭所有 Bank 的选择行 |
| L | L | L | H | X | X | X | SDCKE = 1 时自动刷新模式<br>SDCKE = 0 时自刷新模式 |
| L | L | L | L | L | Mode | Mode | 加载模式寄存器 |

## SDRAM 控制器操作

### IO 配置

SDRAMC 的 IO 口必须在与 SDRAM 通信之前配置，SDRAM 的 IO 配置如*表 25-22. SDRAM IO 定义*所示:

**表 25-22. SDRAM IO 定义**

| 信号 | 传输方向 | 描述 |
|---|---|---|
| EXMC_SDCLK | O | SDRAM 存储器时钟 |
| EXMC_SDCKE[0] | O | SDRAM device 0 的时钟使能信号 |
| EXMC_SDCKE[1] | O | SDRAM device 1 的时钟使能信号 |
| EXMC_SDNE[0] | O | SDRAM device 0 的片选信号，低电平有效 |
| EXMC_SDNE [1] | O | SDRAM device 1 的片选信号，低电平有效 |
| EXMC_NRAS | O | 行地址选通，低电平有效 |
| EXMC_NCAS | O | 列地址选通，低电平有效 |
| EXMC_SDNWE | O | 写使能，低电平有效 |
| EXMC_A[12:0] | O | 地址 |
| EXMC_A[15:14] | O | Bank 地址 |
| EXMC_D[31:0] | I/O | 读/写数据 |
| EXMC_NBL[3:0] | O | 写数据标记(掩码) |

### 控制器初始化

用户需要按照以下步骤来初始化 SDRAM 控制器，初始化序列可以应用于单个 SDRAM，或同时初始化两个 SDRAM，由寄存器 EXMC_SDCMD 位 DS0 和 DS1 决定。为了保证读写的可靠性，必须先进行初始化。

1. 控制参数：控制配置寄存器 EXMC_SDCTLx 指定 SDRAM 的存储器行列数，时钟配置和读写方法。
2. 时序参数：时序配置寄存器 EXMC_SDTCFGx 需要根据 SDRAM 数据手册来配置。RPD 和 ARFD 在 EXMC_SDTCFG0 来配置，EXMC_SDTCFG1 这两位保留。
3. 使能 SDCLK：使用到的 SDRAM 需要给寄存器 EXMC_SDCMD 位域 CMD 写'0b001'使能相应设备的 SDCLK 信号，DS0 和 DS1 决定接收命令的设备。
4. 上电延迟：100us
5. 预充电：命令会对 SDRAM 的所有 Bank 进行复位，并使 SDRAM 回到空闲状态。给寄存器 EXMC_SDCMD 位域 CMD 写'0b010'使能相应设备的 SDCLK 信号，DS0 和 DS1 决定接收命令的设备。
6. 设置自刷新模式：给寄存器 EXMC_SDCMD 位域 CMD 写'0b011'发送自刷新命令。用户也可以通过设置位 NARF 来设置连续刷新次数，DS0 和 DS1 决定接收命令的设备。
7. 模式寄存器配置：模式寄存器通过写寄存器 EXMC_SDCMD 位域 MRC 来设置，其中定义了突发长度，突发类型，CAS 延迟和读写模式，这些需要参考 SDRAM 用户手册。CAS 延迟必须与寄存器 EXMC_SDCTLx 位域 CL 对应，突发长度设为 1 来保证数据正常传输。如果两个 SDRAM 的模式寄存器内容不同，需要通过 DS0 和 DS1 单独选择设备来配置。
8. 设置自刷新频率：自刷新频率与刷新周期相对应，用户必须确保刷新周期满足 SDRAM 的要求。

这里控制器已经完成初始化，可以与 SDRAM 通信。如果发生了复位，初始化需要按照上述步骤重复一遍。在读写操作之前，要保证控制器至少初始化一遍。

### 预充电

若SDRAM控制器在存取时需要进行行切换，那么首先需要将该行地址对应块的读写放大器去使能，使其进入空闲状态，为下一行的读写操作进行准备。这个过程叫做预充电，或者行去使能。预充电可以由控制器的全局预充电命令(Precharge-All)独立激发，或者是在读写完成后自动激发。行预充电延时(RPD)代表SDRAM行切换的最小时间，它是预充电完成到下一次行使能命令的最小时间间隔

### 行使能

行使能命令将行地址所在的块使能，完整的行地址由2比特的块地址EXMC_A[15:14]和13比特的行地址EXMC_A[12:0]组成。行使能会将所选行的16384比特信息读入读写放大器，这个过程也叫做行开启，该命令的一个副作用就是对所选行的存储单元进行了刷新。

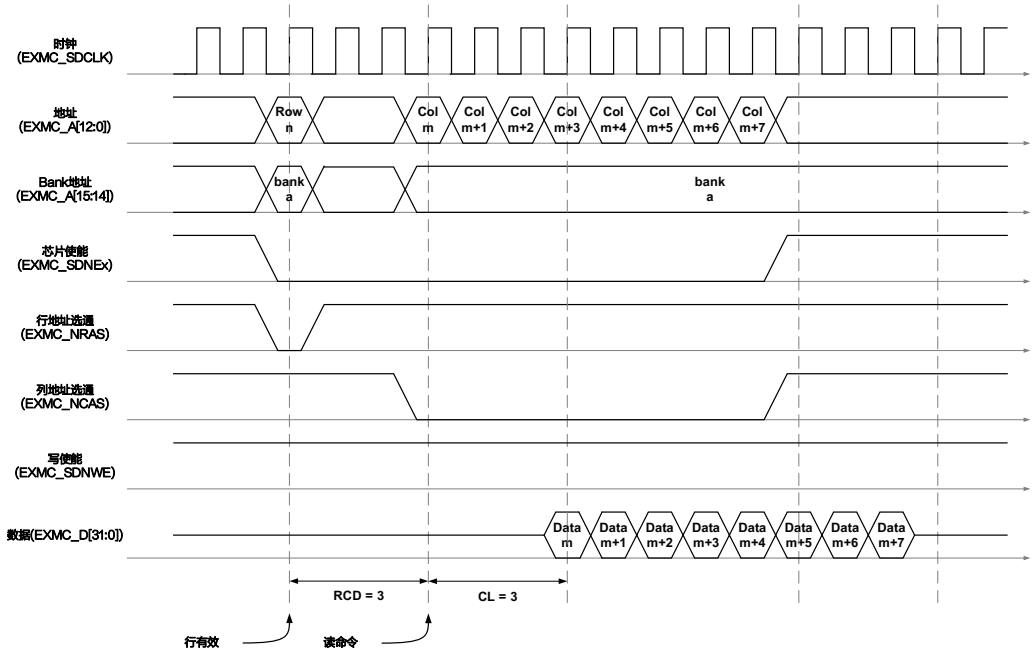
一旦行使能，读写操作就可以顺利的进行，但是行使能需要一定的时间，这个时间间隔叫做行列延时，它是行寻址到列寻址的最小时间间隔。对SDRAM控制器进行配置时的行列延时(RCD)，是包含SDRAM行列延时的最小时钟周期数，它代表了行使能到SDRAM读写间的最小等待时间。在这段时间中，用户可向其它的块地址发出控制命令，因为SDRAM控制器对块的操作是独立进行的。

### 读写访问

控制器可以把AHB的单次或突发读操作转换成单次的存储器读操作。为了连续访问，控制器通常会保存之前操作的行号。若下一次的读取位置是在相同的行号或是已经使能的其他行号，那么读操作会未中断的执行，否则需要先执行取消使能当前行和使能需要操作的行，最后才会执行操作。FIFO设计用来在CAS延迟和管道延迟的时候缓存数据。

*图 25-30. 突发读操作*是对一个未被使能的行突发读操作，在读之前发送了行使能指令。若对一个已经使能的行进行读操作，只需要发送列地址，行地址无需发送。

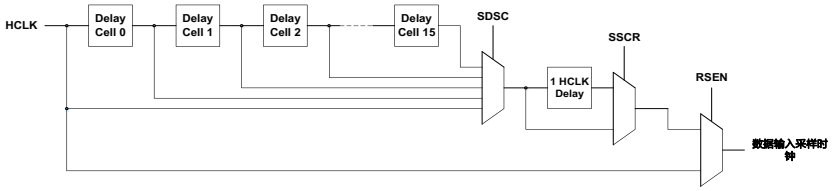
**图 25-30. 突发读操作**




当HCLK采样数据不准的时候，可以使用根据HCLK微调的内部可调时钟。当使能这个时钟的时候，读取的数据在进入AHB总线之前会先存储在异步FIFO中。2-3个HCLK延迟会被添加到读命令过程中。

时钟延迟模块在HCLK输入到信号发生器后添加，这个延迟的时钟作为输入数据的采样时钟。延迟模块可以通过寄存器EXMC_SDRSCTL来控制，其中RSEN位选择是否使用HCLK延迟，SSCR位选择是否额外增加一个HCLK延迟，SDSC选择增加多少个HCLK延迟。

**图 25-31. 数据采样时钟延迟模块**

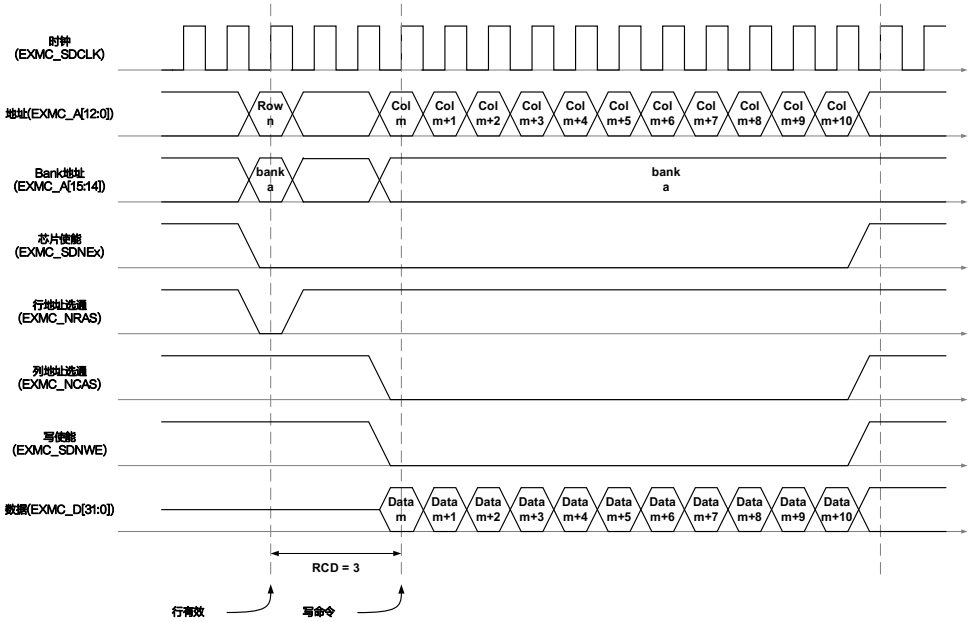



控制器可以把AHB的单次或突发写操作转换成单次的存储器写操作。写操作之前必须失能写保护位(寄存器EXMC_SDCTL位WPEN)。为了连续访问，控制器通常会保存之前操作的行号，若下一次操作是在相同的行号或是已经使能的其他行号，操作会未中断的执行，否则需要先执行

取消使能当前行和使能需要操作的行，最后才会执行操作。

*图 25-32. 突发写操作*是对一个未被使能的行突发写操作，在写之前发送了行使能指令。若对一个已经使能的行写操作，只需要发送列地址，行地址无需发送。

**图 25-32. 突发写操作**




读写命令预处理模块接收AHB命令，然后根据AHB总线和SDRAM接口的数据总线宽度将AHB命令转换成单个的SDRAM访问。

在读写命令预处理模块中，有两个写FIFO，用于缓冲AHB写命令的地址和数据。当两个写FIFO都不为空时，产生写访问。

当寄存器EXMC_SDCTLx位BRSTRD置1时，读写命令预处理模块能够预处理下一个读访问。读FIFO被用来存储在CAS延迟(由EXMC_SDCTLx中的CL位配置)和PIPED延迟(由EXMC_SDCTLx中的PIPED位配置)期间提前读出的数据。

读数据FIFO能够最多缓存6个32位的读数据字，同时地址FIFO携带6个14位的读地址标签，这些标签用来标识6个32位的读数据字中每一个。每个地址标签由11位列地址，2位Bank地址和1位SDRAM设备选择位。

当在AHB总线上出现一个读命令时，读写命令预处理模块将首先检查这个地址是否和某个地址标签匹配，如果匹配，则直接从FIFO中读取数据。否则，向存储器发一个新的读命令，FIFO会被新的数据更新。如果FIFO满了，旧的数据会被丢失。

读FIFO操作，如*图 25-33. FIFO 未命中时的读访问(BRSTRD=1，CL=2，SDCLK=2，PIPED=2)*及*图 25-34. FIFO 命中时的读访问(BRSTRD=1)*所示。

**图 25-33. FIFO 未命中时的读访问(BRSTRD=1，CL=2，SDCLK=2，PIPED=2)**

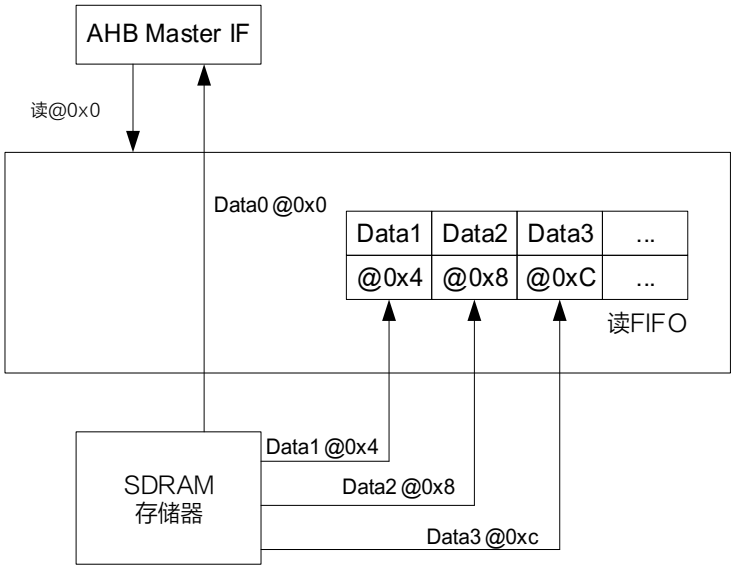
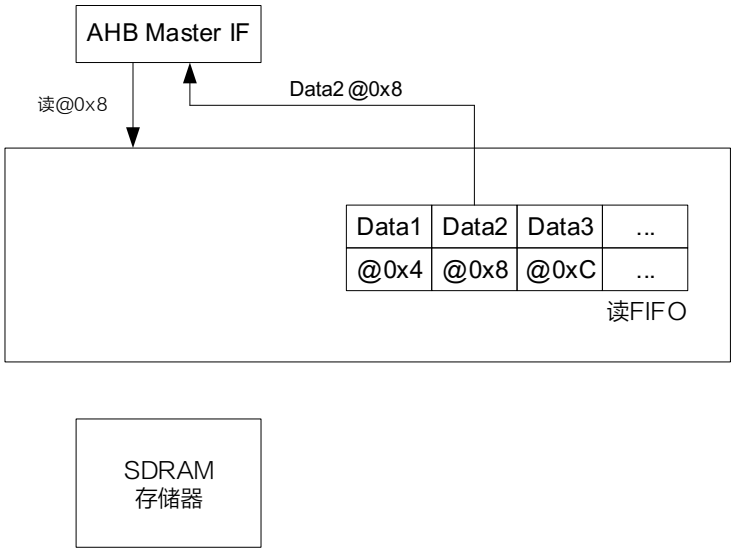



**图 25-34. FIFO 命中时的读访问(BRSTRD=1)**




当一个写访问或者预充电命令出现时，读FIFO缓冲区中的数据就会被清除掉，用以填充新的数据。

地址译码器子模块会根据外部存储器设备的配置将AHB总线地址转化成片选、内部bank地址、行地址和列地址。

使能缓存子模块记录着内部banks是否处于使能状态。当一个内部bank处于使能状态，则相应的行地址也会被记录。当AHB访问或者自动刷新命令出现时，读写命令预处理模块将会查询这个记录，并且决定是否生成使能或预充电命令。

在读/写操作之前，目标行必须被使能，EXMC_A[15:14]选择Bank，EXMC_A[12:0]选择行。被选择的行在预充电命令出现前会一直有效。预充电命令用来取消选择特定Bank或者所有Bank使能的行。预充电命令必须在使能同一个Bank的不同行之前发出。使能和预充电由EXMC自动执行，它的正确性取决于之前描述的存储器的相关配置。有关自动行使能和预充电的读写时序

如下图所示。

**图 25-35. 跨边界读操作**

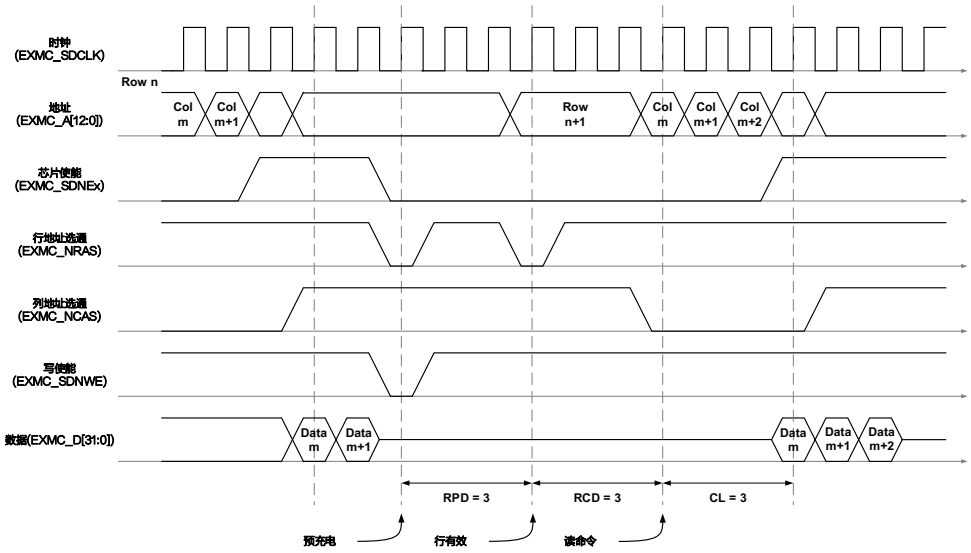
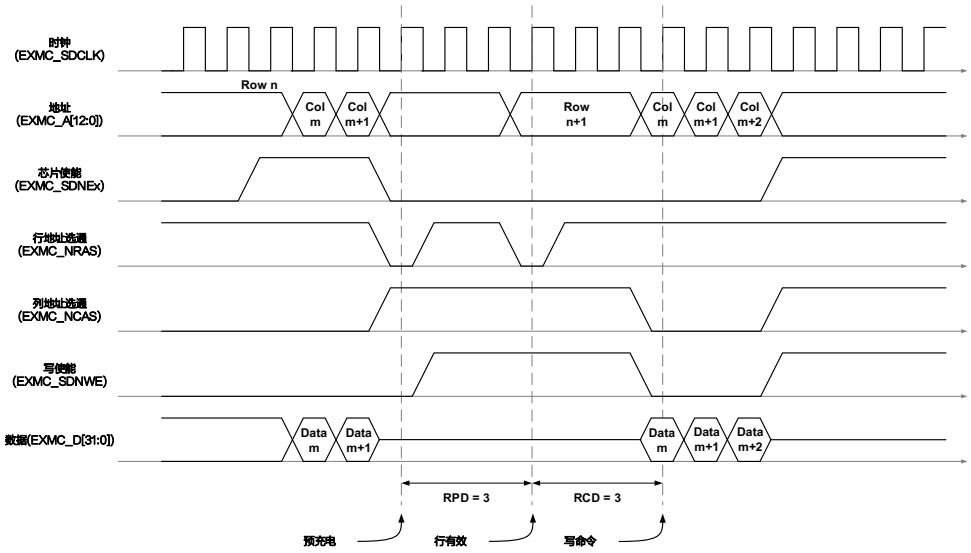



**图 25-36. 跨边界写操作**




上图描述了在跨行边界时的读写操作时序，会按照以下步骤自动执行：

1. 预充电当前行；
2. 使能下一行；
3. 读写操作。

预充电延迟(PRD)和行到列延迟(RCD)会根据寄存器 EXMC_SDTCFGx 设置。其他时序参数必须参照 SDRAM 标准要求。

当读写操作发生在 Bank 边界时，会有以下两种情况：

1. 当前 Bank 不是最后一个 Bank，使能下一个 Bank 的第一行，支持任意的行，列，总线宽度设置。
2. 当前 Bank 是最后一个 Bank，行，列，总线宽度设置为 13 位，11 位，32 位。假设当前操作的 SDRAM 位 device0，控制器会在 device1 上继续操作。

### 低功耗模式

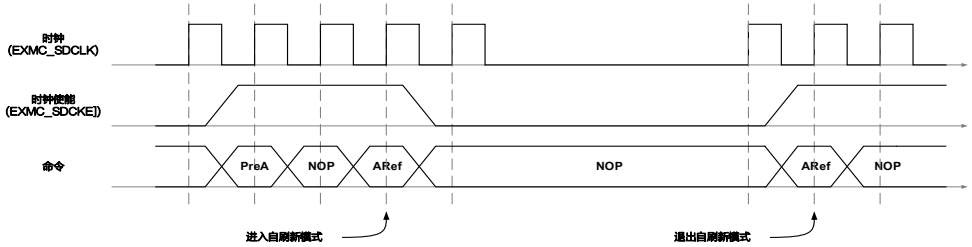
EXMC 支持两种低功耗模式：

1. 自刷新模式：在自刷新模式中，在没有外部时钟(EXMC_CLK)的情况下，刷新由 SDRAM 本身提供，以此来保持数据的完整性。通过往寄存器 EXMC_SDCMD 位域 CMD 写入'0b101'进入自刷新模式，DS0 和 DS1 决定哪个 SDRAM 设备接收到该命令。如果自刷新指令发送给两个 SDRAM 设备或一个未初始化的 SDRAM 设备，则在 RASD 延迟后 EXMC_SDCLK 停止运行。
2. 掉电模式：在掉电模式中，刷新由 SDRAM 控制器提供。通过往寄存器 EXMC_SDCMD 位域 CMD 写入'0b110'进入掉电模式，DS0 和 DS1 决定哪个 SDRAM 设备接收到该命令。如果写数据 FIFO 非空，在掉电模式使能之前，所有数据都会发送给存储器。

命令模式状态机也控制正常模式和低功耗模式(自刷新/掉电)之间的转换过程。

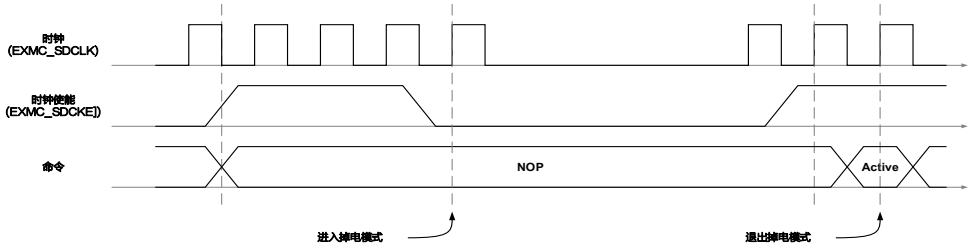
当读写访问出现时，SDRAM 控制器会从自刷新模式退出，返回到正常模式。如果在 SDRAM 控制器进入自刷新模式时出现读写访问，则自刷新的进入过程会被中断，并且在读写访问完成后 SDRAM 控制器会停留在正常模式。

**图 25-37. 自刷新模式进入和退出的处理**




如果在SDRAM控制器处于掉电模式时出现自动刷新请求，那么SDRAM控制器会退出掉电模式并返回到正常模式，发"预充电所有存储区域"命令和自动刷新命令序列，然后再一次自动进入掉电模式。

**图 25-38. 掉电模式进入和退出的处理**




### 状态和中断

寄存器EXMC_SDSTAT的准备未完成状态位NRDY指示SDRAM是否准备完成接受新的命令。在控制器发送新的命令之后，该位会被清除。

寄存器EXMC_SDSTAT的STA0和STA1定义SDRAM的device0和device1的状态，0x00为普通

模式，0b01为自刷新模式，0b10为掉电模式。

若前一个刷新指令未完成时，接收到了新的刷新指令，寄存器EXMC_SDSTAT刷新错误标志位(REIF)会被置位，该位通过寄存器EXMC_SDARI位REC置位来清除。

## 25.4. EXMC 寄存器

EXMC基地址：0xA000 0000

### 25.4.1. NOR/PSRAM 控制器寄存器

#### SRAM/NOR Flash 控制寄存器(EXMC_SNCTLx)(x=0, 1, 2, 3)

偏移地址：0x00 + 8 * x(x = 0, 1, 2, 3)
复位值：0x0000 30DA

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | CCK | SYNCWR | CPS[2:0] | | |
| | | | | | | | | | | | rw | rw | | rw | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ASYNCWTEN | EXMODEN | NRWTEN | WEN | NRWTCFG | WRAPEN | NRWTPOL | SBRSTEN | 保留 | NREN | NRW[1:0] | | NRTP[1:0] | | NRMUX | NRBKEN |
| rw | rw | rw | rw | rw | rw | rw | rw | | rw | rw | | rw | | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:21 | 保留 | 必须保持复位值。 |
| 20 | CCK | 连续时钟配置<br>0：EXMC_CLK只在同步模式产生<br>1：EXMC_CLK无条件产生<br>**注意**：该位只在EXMC_SNCTL0有效, EXMC_SNCTLx (x = 1, 2, 3)没有意义 |
| 19 | SYNCWR | 选择写操作模式<br>0：异步写操作<br>1：同步写操作 |
| 18:16 | CPS[2:0] | CRAM页大小<br>000：页边界自动突发分割<br>001：128字节<br>010：256字节<br>011：512字节<br>100：1024字节<br>其他：保留 |
| 15 | ASYNCWTEN | 异步等待功能使能位<br>0：禁用异步等待功能<br>1：使能异步等待功能 |
| 14 | EXMODEN | 扩展模式使能<br>0：禁用扩展模式，即不使用EXMC_SNWTCFGx |

| | | |
|---|---|---|
| | | 1：使能扩展模式 |
| 13 | NRWTEN | NWAIT信号使能<br>对于存储器的突发模式访问，该位使能/禁用等待状态插入NWAIT信号功能<br>0：成组传输模式时，禁用NWAIT信号<br>1：成组传输模式时，使能NWAIT信号 |
| 12 | WEN | 写操作使能<br>0：禁止EXMC对外部存储器的写操作，否则产生一个AHB错误<br>1：允许EXMC对外部存储器的写操作(复位缺省值) |
| 11 | NRWTCFG | NWAIT信号配置，只在同步模式有效<br>0：NWAIT信号在等待状态前的一个数据周期有效<br>1：NWAIT信号在等待状态期间有效 |
| 10 | WRAPEN | 非对齐成组模式使能<br>0：禁止非对齐成组操作<br>1：允许非对齐成组操作 |
| 9 | NRWTPOL | NWAIT信号极性<br>0：NWAIT低电平有效<br>1：NWAIT高电平有效 |
| 8 | SBRSTEN | 同步突发模式使能<br>0：禁止同步突发模式<br>1：使能同步突发模式 |
| 7 | 保留 | 必须保持复位值。 |
| 6 | NREN | NOR闪存访问使能<br>0：禁止NOR Flash访问<br>1：允许NOR Flash访问 |
| 5:4 | NRW[1:0] | 存储器数据宽度<br>00：8位<br>01：16位(复位缺省值)<br>10/11：保留 |
| 3:2 | NRTP[1:0] | 存储器类型<br>00：SRAM、ROM<br>01：PSRAM(CRAM)<br>10：NOR Flash<br>11：保留 |
| 1 | NRMUX | 数据线/地址线复用<br>0：禁用地址/数据复用功能<br>1：允许地址/数据复用功能 |
| 0 | NRBKEN | 存储块使能<br>0：禁用对应的存储器块 |

1：使能对应的存储器块

### SRAM/NOR Flash 时序寄存器(EXMC_SNTCFGx)(x=0, 1, 2, 3)

偏移地址：0x04 + 8 * x(x = 0, 1, 2, 3)
复位值：0x0FFF FFFF

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | ASYNCMOD[1:0] | | DLAT[3:0] | | | | CKDIV[3:0] | | | | BUSLAT[3:0] | | | |
| | | rw | | rw | | | | rw | | | | rw | | | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:30 | 保留 | 必须保持复位值。 |
| 29:28 | ASYNCMOD[1:0] | 异步访问模式<br>该位只有在扩展模式中使用<br>00：模式A<br>01：模式B<br>10：模式C<br>11：模式D |
| 27:24 | DLAT[3:0] | NOR Flash数据延时，仅在同步模式有效<br>0x0：第一数据的保持时间为2个EXMC_CLK时钟周期<br>0x1：第一数据的保持时间为3个EXMC_CLK时钟周期<br>……<br>0xF：第一数据的保持时间为17个EXMC_CLK时钟周期 |
| 23:20 | CKDIV[3:0] | 同步模式时钟分频比，仅在同步模式有效<br>0x0：保留<br>0x1：EXMC_CLK周期=2个HCLK周期<br>……<br>0xF：EXMC_CLK周期=16个HCLK周期 |
| 19:16 | BUSLAT[3:0] | 总线延迟时间<br>在复用读模式中使用，避免总线冲突，是总线恢复到高阻态的最小时间<br>0x0：总线延迟=0个HCLK周期<br>0x1：总线延迟=1个HCLK周期<br>……<br>0xF：总线延迟=15个HCLK周期 |
| 15:8 | DSET[7:0] | 异步数据建立时间<br>该位域仅在异步模式有效 |

| | | |
|---|---|---|
| | | 0x00：保留<br>0x01：数据建立时间=1个HCLK周期<br>……<br>0xFF：数据建立时间=255个HCLK周期 |
| 7:4 | AHLD[3:0] | 异步地址保持时间<br>该位域设置地址保持时间，仅在模式D与复用模式有效<br>0x0：保留<br>0x1：地址保持时间=1个HCLK<br>……<br>0xF：地址保持时间=15个HCLK |
| 3:0 | ASET[3:0] | 异步地址建立时间<br>该位域设置地址建立时间<br>**注意**：该位域仅在SRAM,ROM,NOR Flash的异步模式有效<br>0x0：地址建立时间= 0个HCLK<br>……<br>0xF：地址建立时间= 15个HCLK |

### SRAM/NOR Flash 写时序寄存器(EXMC_SNWTCFGx)(x=0, 1, 2, 3)

偏移地址：0x104 + 8 * x(x = 0, 1, 2, 3)
复位值：0x0FFF FFFF

该寄存器仅在扩展模式使能(寄存器EXMC_SNCTL位EXMODEN置1)后有效。

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:30 | 保留 | 必须保持复位值。 |
| 29:28 | WASYNCMOD[1:0] | 异步访问模式<br>该位只有在扩展模式中使用<br>00：模式A<br>01：模式B<br>10：模式C<br>11：模式D |
| 27:20 | 保留 | 必须保持复位值。 |
| 19:16 | WBUSLAT[3:0] | 总线延迟时间 |

| | | |
|---|---|---|
| | | 在复用读模式中使用，避免总线冲突，是总线恢复到高阻态的最小时间<br>0x0：总线延迟=0个HCLK周期<br>0x1：总线延迟=1个HCLK周期<br>……<br>0xF：总线延迟=15个HCLK周期 |
| 15:8 | WDSET[7:0] | 异步数据建立时间<br>该位域仅在异步模式有效<br>0x00：保留<br>0x01：数据建立时间=1个HCLK周期<br>……<br>0xFF：数据建立时间=255个HCLK周期 |
| 7:4 | WAHLD[3:0] | 异步地址保持时间<br>该位域设置地址保持时间，仅在模式D与复用模式有效<br>0x0：保留<br>0x1：地址建立时间=1个HCLK<br>……<br>0xF：地址建立时间=15个HCLK |
| 3:0 | WASET[3:0] | 异步地址建立时间<br>该位域设置地址建立时间<br>**注意**：该位域仅在SRAM,ROM,NOR Flash的异步模式有效<br>0x0：地址建立时间= 0个HCLK<br>……<br>0xF：地址建立时间= 15个HCLK |

## 25.4.2. NAND Flash/PC Card 控制器寄存器

### NAND Flash/PC Card 控制寄存器(EXMC_NPCTLx)(x=1, 2, 3)

偏移地址：0x40 + 0x20 * x(x = 1, 2, 3)
复位值：0x0000 0008

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | ECCSZ[2:0] | | | ATR[3] |
| | | | | | | | | | | | | | rw | | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ATR[2:0] | | | CTR[3:0] | | | | 保留 | | ECCEN | NDW[1:0] | | NDTP | NDBKEN | NDWTEN | 保留 |
| rw | | | rw | | | | | | rw | rw | | rw | rw | rw | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:20 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 19:17 | ECCSZ[2:0] | ECC块大小<br>000：256字节<br>001：512字节<br>010：1024字节<br>011：2048字节<br>100：4096字节<br>101：8192字节 |
| 16:13 | ATR[3:0] | ALE至RE的延迟<br>0x0：1个HCLK<br>……<br>0xF：16个HCLK |
| 12:9 | CTR[3:0] | CLE至RE的延迟<br>0x0：1个HCLK<br>0x1：2个HCLK<br>……<br>0xF：16个HCLK |
| 8:7 | 保留 | 必须保持复位值。 |
| 6 | ECCEN | ECC使能<br>0：关闭ECC，并复位EXMC_NECCx<br>1：打开ECC |
| 5:4 | NDW[1:0] | 外部存储器数据宽度<br>00：8位<br>01：16位<br>其他：保留<br>**注意**：对于PC/CF Card，数据宽度必须选择16位 |
| 3 | NDTP | 外部存储器的类型<br>0：PC Card, CF Card, PCMCIA<br>1：NAND Flash |
| 2 | NDBKEN | 存储块使能<br>0：禁用对应的存储器块<br>1：使能对应的存储器块 |
| 1 | NDWTEN | NWAIT信号使能位<br>0：关闭等待功能<br>1：使能等待功能 |
| 0 | 保留 | 必须保持复位值。 |

### NAND Flash/PC Card 中断使能寄存器(EXMC_NPINTENx)(x=1, 2, 3)

偏移地址：0x44 + 0x20 * x(x = 1, 2, 3)

复位值：0x0000 0042 (对于bank1和bank2) 0x0000 0040 (对于bank3)

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | FFEPT | INTFEN | INTHEN | INTREN | INTFS | INTHS | INTRS |
| | | | | | | | | | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:7 | 保留 | 必须保持复位值。 |
| 6 | FFEPT | FIFO空标志位<br>0：FIFO非空<br>1：FIFO空 |
| 5 | INTFEN | 中断下降沿检测使能<br>0：禁用中断下降沿检测<br>1：使能中断下降沿检测 |
| 4 | INTHEN | 中断高电平检测使能<br>0：禁用中断高电平检测<br>1：使能中断高电平检测 |
| 3 | INTREN | 中断上升沿中断检测使能<br>0：禁用中断上升沿检测<br>1：使能中断上升沿检测 |
| 2 | INTFS | 中断下降沿状态<br>0：没有检测到中断下降沿<br>1：检测到中断下降沿 |
| 1 | INTHS | 中断高电平状态<br>0：没有检测到中断高电平<br>1：检测到中断高电平 |
| 0 | INTRS | 中断上升沿状态<br>0：没有检测到中断上升沿<br>1：检测到中断上升沿 |

### NAND Flash/PC Card 通用空间时序寄存器(EXMC_NPCTCFGx)(x=1, 2, 3)

偏移地址：0x48 + 0x20 * x(x = 1, 2, 3)
复位值：0xFFFFFFFF

这些操作适用于以下类型的外部存储器的通用存储空间16位的PC Card，CF card和NAND Flash。

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| COMHIZ[7:0] | | | | | | | | COMHLD[7:0] | | | | | | | |
| rw | | | | | | | | rw | | | | | | | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | COMHIZ[7:0] | 通用空间数据总线的高阻时间<br>定义在通用空间进行写操作后数据总线保持高阻态时间<br>0x00：1个HCLK<br>……<br>0xFE：255个HCLK<br>0xFF：保留 |
| 23:16 | COMHLD[7:0] | 通用空间的保持时间<br>在发送地址后的地址保持时间，在写操作时，也作为数据信号保持的时间<br>0x00：保留<br>0x01：1个HCLK<br>……<br>0xFE：254个HCLK<br>0xFF：保留 |
| 15:8 | COMWAIT[7:0] | 通用空间的等待时间<br>定义了保持命令的最小时间<br>0x00：保留<br>0x01：2个HCLK(加上NWAIT时钟周期)<br>……<br>0xFE：255个HCLK(加上NWAIT时钟周期)<br>0xFF：保留 |
| 7:0 | COMSET[7:0] | 通用空间的建立时间<br>定义地址信号的建立时间<br>0x00：1个HCLK<br>……<br>0xFE：255个HCLK<br>0xFF：保留 |

### NAND Flash/PC Card 属性空间时序寄存器(EXMC_NPATCFGx)(x=1, 2, 3)

偏移地址：0x4C + 0x20 * x(x = 1, 2, 3)
复位值：0x FFFFFFFF

这些操作适用于以下类型的外部存储器的属性存储空间8位的PC Card和NAND Flash。

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ATTHIZ[7:0] | | | | | | | | ATTHLD[7:0] | | | | | | | |
| rw | | | | | | | | rw | | | | | | | |

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

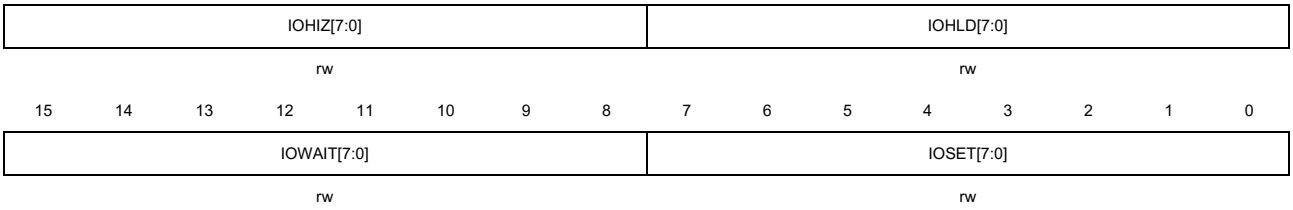
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | ATTHIZ[7:0] | 属性空间数据总线的高阻时间<br>定义在属性空间进行写操作后数据总线保持高阻态时间<br>0x00：0个HCLK<br>……<br>0xFE：254个HCLK<br>0xFF：保留 |
| 23:16 | ATTHLD[7:0] | 属性空间的保持时间<br>在发送地址后的地址保持时间，在写操作时，也作为数据信号保持的时间<br>0x00：保留<br>0x01：1个HCLK<br>……<br>0xFE：254个HCLK<br>0xFF：保留 |
| 15:8 | ATTWAIT[7:0] | 属性空间的等待时间<br>定义了保持命令的最小时间<br>0x00：保留<br>0x01：2个HCLK(加上NWAIT时钟周期)<br>……<br>0xFE：255个HCLK(加上NWAIT时钟周期)<br>0xFF：保留 |
| 7:0 | ATTSET[7:0] | 属性空间的建立时间<br>定义地址信号的建立时间<br>0x00：1个HCLK<br>……<br>0xFE：255个HCLK<br>0xFF：保留 |

### PC Card I/O 空间时序寄存器(EXMC_PIOTCFG3)

偏移地址：0xB0
复位值：0x FFFFFFFF

该寄存器只能按字(32位)访问。

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

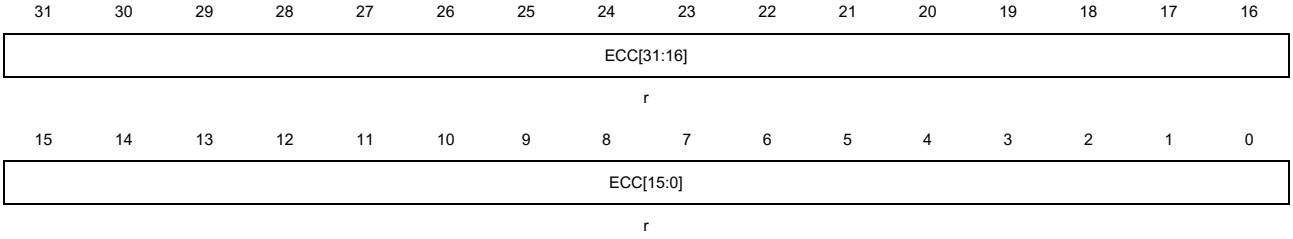



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | IOHIZ[7:0] | I/O空间数据总线的高阻时间<br>定义在IO空间进行写操作后数据总线保持高阻态时间<br>0x00：0个HCLK<br>……<br>0xFF：255个HCLK |
| 23:16 | IOHLD[7:0] | I/O空间的保持时间<br>在发送地址后的地址保持时间，在写操作时，也作为数据信号保持的时间0x00：保留<br>0x01：1个HCLK<br>……<br>0xFF：255个HCLK |
| 15:8 | IOWAIT[7:0] | I/O空间的等待时间<br>定义了保持命令的最小时间<br>0x00：保留<br>0x01：2个HCLK(加上NWAIT时钟周期)<br>……<br>0xFF：256个HCLK(加上NWAIT时钟周期) |
| 7:0 | IOSET[7:0] | I/O空间的建立时间<br>定义地址信号的建立时间<br>0x00：1个HCLK<br>……<br>0xFF：256个HCLK |

### NAND Flash ECC 结果寄存器(EXMC_NECCx)(x=1, 2)

偏移地址：0x54+0x20 * x(x =1, 2)
复位值：0x0000 0000

该寄存器只能按字(32位)访问。



| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:0 | ECC[31:0] | ECC计算结果<br>当ECCSZ[2:0] = 000，页大小为256字节，ECC计算结果存储在ECC[21:0]位域。<br>当ECCSZ[2:0] = 001，页大小为512字节，ECC计算结果存储在ECC[23:0]位域。<br>当ECCSZ[2:0] = 010，页大小为1024字节，ECC计算结果存储在ECC[25:0]位域。<br>当ECCSZ[2:0] = 011，页大小为2048字节，ECC计算结果存储在ECC[27:0]位域。<br>当ECCSZ[2:0] = 100，页大小为4096字节，ECC计算结果存储在ECC[29:0]位域。<br>当ECCSZ[2:0] = 101，页大小为8192字节，ECC计算结果存储在ECC[31:0] ]位域。 |

## 25.4.3. SDRAM 控制器寄存器

### SDRAM 控制寄存器(EXMC_SDCTLx)(x=0, 1)

偏移地址：0x140+4*x(x = 0, 1)
复位值：0x0000 02D0

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 保留 | PIPED[1:0] | | BRSTRD | SDCLK[1:0] | | WPEN | CL[1:0] | | NBK | SDW[1:0] | | RAW[1:0] | | CAW[1:0] | |
| | rw | | rw | rw | | rw | rw | | rw | rw | | rw | | rw | |

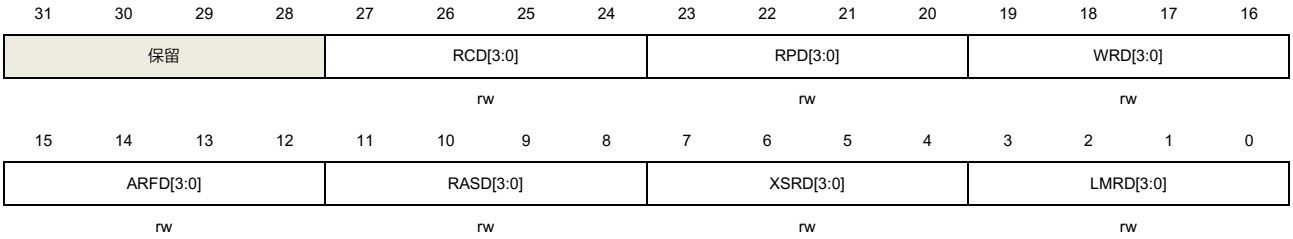
| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:15 | 保留 | 硬件强制清零 |
| 14:13 | PIPED[1:0] | 流水线读数据延迟<br>这些位用于指定在CAS延迟之后再延迟多少个HCLK时钟周期才去读数据<br>00：延迟0个HCLK周期<br>01：延迟1个HCLK周期<br>10：延迟2个HCLK周期<br>11：保留<br>**注意**：寄存器EXMC_SDCTL1相应位保留 |
| 12 | BRSTRD | 突发读开关<br>当该位被置位时，会在CAS延迟期间预期处理下一个读命令，并将数据存储到读FIFO中。<br>0：禁用突发读<br>1：使能突发读<br>**注意**：寄存器EXMC_SDCTL1相应位保留 |
| 11:10 | SDCLK[1:0] | SDRAM时钟配置 |

| | | |
|---|---|---|
| | | 这些位指定了两个SDRAM device的时钟周期。如果需要修改存储器时钟配置，首先需要将存储器时钟禁用，并且在修改配置后将存储器重新初始化。<br>00：SDCLK存储器时钟禁用<br>01：保留<br>10：SDCLK存储器周期为2个HCLK<br>11：SDCLK存储器周期为3个HCLK<br>**注意**：寄存器EXMC_SDCTL1相应位保留 |
| 9 | WPEN | 写保护<br>该位禁用写保护功能<br>0：禁用写保护，允许写访问<br>1：使能写保护，忽略写访问 |
| 8:7 | CL[1:0] | CAS延迟<br>这些位用于设定SDRAM CAS延迟多少个SDRAM存储器时钟周期单元<br>00：保留不使用<br>01：1个周期<br>10：2个周期<br>11：3个周期 |
| 6 | NBK | 内部Bank的个数<br>该位指定内部Bank的个数<br>0：2个内部Banks<br>1：4个内部Banks |
| 5:4 | SDW[1:0] | SDRAM数据总线宽度<br>该位指定SDRAM存储器数据总线宽度<br>00：8位<br>01：16位<br>10：32位<br>11：保留 |
| 3:2 | RAW[1:0] | 行地址位宽<br>这些位用于指定行地址的比特宽度<br>00：11位<br>01：12位<br>10：13位<br>11：保留 |
| 1:0 | CAW[1:0] | 列地址位宽<br>这些位用于指定列地址的比特宽度<br>00：8位<br>01：9位<br>10：10位<br>11：11位 |

### SDRAM 时序寄存器(EXMC_SDTCFGx)(x=0, 1)

偏移地址：0x148+4*x(x = 0, 1)
复位值：0x0FFF FFFF

该寄存器只能按字(32位)访问。



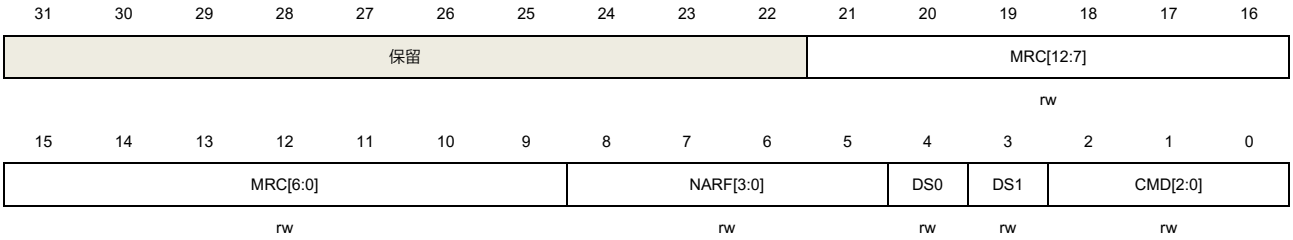

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:24 | RCD[3:0] | 行到列的延迟<br>这些位指定了使能命令与读/写命令之间延迟多少SDRAM时钟周期单元<br>0x0：1个周期<br>0x1：2个周期<br>....<br>0xF：16个周期 |
| 23:20 | RPD[3:0] | 行预充电延迟<br>这些位指定了预充电命令与下一个命令之间延迟多少SDRAM存储器时钟周期单元<br>0x0：1个周期<br>0x1：2个周期<br>....<br>0xF：16个周期<br>**注意**：寄存器EXMC_SDTCFG1相应位保留，如果两个SDRAM存储器都被使用，RPD必须用较慢设备的时序来配置 |
| 19:16 | WRD[3:0] | 写恢复延迟<br>这些位指定写命令和预充电命令之间延迟多少SDRAM存储器时钟周期单元<br>0x0：1个周期<br>0x1：2个周期<br>....<br>0xF：16个周期<br>**注意**：寄存器EXMC_SDTCFG1相应位保留，如果两个SDRAM存储器都被使用，WRD必须用较慢设备的时序来配置 |
| 15:12 | ARFD[3:0] | 自动刷新延迟<br>这些位指定两个连续的刷新命令之间的延迟，在同一个内部bank上两个使能命令之间的延迟，以及刷新命令和使能命令之间的延迟，延迟时间以SDRAM存储器时钟周期为单位<br>0x0：1个周期 |

| | | |
|---|---|---|
| | | 0x1：2个周期<br>....<br>0xF：16个周期<br>**注意**：寄存器EXMC_SDTCFG1相应位保留，如果两个SDRAM存储器都被使用，ARFD必须用较慢设备的时序来配置 |
| 11:8 | RASD[3:0] | 行地址选择延迟<br>这些位指定了使能命令与预充电命令之间延迟多少SDRAM时钟周期单元，也指定了两个连续的自刷新命令之间的最小延迟<br>0x0：1个周期<br>0x1：2个周期<br>....<br>0xF：16个周期 |
| 7:4 | XSRD[3:0] | 退出自刷新延迟<br>这些位指定了从自刷新命令到使能命令之间延迟多少个SDRAM存储器时钟周期单元<br>0x0：1个周期<br>0x1：2个周期<br>....<br>0xF：16个周期 |
| 3:0 | LMRD[3:0] | 加载模式寄存器延迟<br>这些位指定加载模式寄存器命令与刷新或使能命令之间延迟多少SDRAM存储器时钟周期单元<br>0x0：1个周期<br>0x1：2个周期<br>....<br>0xF：16个周期 |

### SDRAM 命令寄存器(EXMC_SDCMD)

偏移地址：0x150
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:22 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 21:9 | MRC[12:0] | 模式寄存器内容<br>这些位指定SDRAM模式寄存器的内容，这些内容在CMD="100"时进行编程 |
| 8:5 | NARF[3:0] | 连续的自动刷新个数<br>这些位指定在CMD="011"时，发出多少个连续自动刷新周期<br>0x0：1个自动刷新周期<br>0x1：2个自动刷新周期<br>....<br>0xE：15个自动刷新周期<br>0xF：保留 |
| 4 | DS0 | 选择SDRAM device 0<br>该位指示SDRAM device 0是否被选择<br>0：SDRAM device 0没有被选择<br>1：SDRAM device 0被选择 |
| 3 | DS1 | 选择SDRAM device 1<br>该位指示SDRAM device 1是否被选择<br>0：SDRAM device 1没有被选择<br>1：SDRAM device 1被选择 |
| 2:0 | CMD[2:0] | 命令<br>这些位指定发送到SDRAM设备上的命令<br>000：正常操作模式<br>001：时钟使能命令<br>010：所有存储区预充电命令<br>011：自动刷新命令<br>100：加载模式寄存器命令<br>101：自刷新命令<br>110：掉电模式进入命令<br>111：保留<br>**注意**：发送命令时，至少需要选择一个设备(设备0或设备1)。如果两个设备同时使用，必须同时选择两个设备发送命令。 |

### SDRAM 自动刷新间隔寄存器(EXMC_SDARI)

偏移地址：0x154
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | REIE | ARINTV[12:0] | | | | | | | | | | | | | REC |
| | rw | rw | | | | | | | | | | | | | w |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:15 | 保留 | 必须保持复位值。 |
| 14 | REIE | 刷新错误中断使能<br>0：中断禁止<br>1：状态寄存器RFE位置1发生中断 |
| 13:1 | ARINTV[12:0] | 自动刷新间隔<br>这些位指定两个连续的自动刷新命令之间间隔多少存储器时钟周期单元<br>ARFITV=(SDRAM刷新周期/行数)-20 |
| 0 | REC | 清除刷新错误标志<br>该位置1会清除状态寄存器REIF位<br>0：没有效果<br>1：清除刷新错误标志 |

### SDRAM 状态寄存器(EXMC_SDSTAT)

偏移地址：0x158
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | NRDY | STA1[1:0] | | STA0[1:0] | | REIF |
| | | | | | | | | | | r | r | | r | | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:6 | 保留 | 必须保持复位值。 |
| 5 | NRDY | 非就绪状态<br>该位指定SDRAM控制器是否已经准备接收一个新的命令<br>0：SDRAM控制器准备好接收新命令<br>1：SDRAM控制器没有准备好接收新命令 |
| 4:3 | STA1[1:0] | device1 状态<br>该位定义SDRAM device1的状态<br>00：正常状态<br>01：自刷新状态<br>10：掉电状态 |
| 2:1 | STA0[1:0] | device 0 状态<br>该位定义SDRAM device 0的状态<br>00：正常状态 |

| | | |
|---|---|---|
| | | 01：自刷新状态<br>10：掉电状态 |
| 0 | REIF | 刷新错误标志<br>0：无刷新错误<br>1：出现刷新错误。若中断使能位置1(REIE)，则产生中断。 |

### SDRAM 读采样控制寄存器(EXMC_SDRSCTL)

偏移地址：0x180
复位值：0x0000 0000

该寄存器只能按字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7:4 | SDSC[3:0] | 选择读数据的采样时钟的延迟单元<br>0x0：0个延迟单元<br>0x1：1个延迟单元<br>……<br>0xF：15个延迟单元 |
| 3:2 | 保留 | 必须保持复位值。 |
| 1 | SSCR | 选择读数据的采样周期<br>0：除延迟之外，为读数据采样时钟增加0个额外的HCLK周期<br>1：除延迟之外，为读数据采样时钟增加1个额外的HCLK周期 |
| 0 | RSEN | 读采样使能<br>0：禁止读采样<br>1：使能读采样 |

## 25.4.4. SQPI-PSRAM 控制器寄存器

### SPI 初始化寄存器(EXMC_SINIT)

偏移地址：0x310
复位值：0x1801 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| POL | IDL[1:0] | | ADRBIT[4:0] | | | | | 保留 | | | | | | CMDBIT[1:0] | |
| rw | rw | | rw | | | | | | | | | | | rw | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | POL | 读数据时的采样极性<br>0：上升沿时进行采样(缺省值)<br>1：下降沿时进行采样 |
| 30:29 | IDL[1:0] | SPI PSRAM ID长度<br>00：64位<br>01：32位<br>10：16位<br>11：8位 |
| 28:24 | ADRBIT[4:0] | SPI PSRAM地址位数<br>范围由1到26(缺省值为24)<br>0x00：保留<br>0x01：1位地址<br>……<br>0x1A：26位地址<br>0x1B：保留<br>……<br>0x1F：保留 |
| 23:18 | 保留 | 必须保持复位值。 |
| 17:16 | CMDBIT[1:0] | SPI PSRAM命令位数<br>00：4位<br>01：8位(缺省值)<br>10：16位<br>11：保留 |
| 15:0 | 保留 | 必须保持复位值。 |

### SPI 读命令寄存器(EXMC_SRCMD)

偏移地址：0x320
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RDID | 保留 | | | | | | | | | RMODE[1:0] | | RWAITCYCLE[3:0] | | | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | RDID | 发送读SPI PSRAM ID的命令，命令码和模式分别通过RCMD和RMODE设置 |
| 30:22 | 保留 | 必须保持复位值。. |
| 21:20 | RMODE[1:0] | SPI PSRAM读命令模式<br>00：非SPI模式<br>01：SPI模式<br>10：SQPI模式<br>11：QPI模式 |
| 19:16 | RWAITCYCLE[3:0] | 读数据时地址阶段结束后等待的周期数 |
| 15:0 | RCMD[15:0] | SPI 读命令的命令码<br>CMDBIT不同时，RCMD有效位不同：<br>CMDBIT=00，RCMD[3:0]有效<br>CMDBIT=01，RCMD[7:0]有效<br>CMDBIT=10，RCMD[15:0]有效 |

**注意**：在向 RDID 位写 1 之前，你必须确保该位已被清除，RDID 置 1 之后，必须等待 RDID 被清除

### SPI 写命令寄存器(EXMC_SWCMD)

偏移地址：0x330
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | SC | 发送SPI PSRAM没有地址和数据阶段的特殊命令，命令码和模式分别由WCMD和WMODE设置 |
| 30:22 | 保留 | 必须保持复位值。 |
| 21:20 | WMODE[1:0] | SPI PSRAM写命令模式<br>00：非SPI模式<br>01：SPI模式 |

| | | |
|---|---|---|
| | | 10：SQPI模式<br>11：QPI模式 |
| 19:16 | WWAITCYCLE[3:0] | 写数据时地址阶段结束后等待的周期数 |
| 15:0 | WCMD[15:0] | SPI 写命令的命令码 |

**注意**：在向 SC 位写 1 之前，你必须确保该位已被清除，SC 置 1 之后，必须等待 SC 被清除

### SPI ID 低位寄存器(EXMC_SIDL)

偏移地址：0x340
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | SIDL[31:0] | ID低位数据<br>当IDL=00或01时，SIDL[31:0]有效<br>当IDL=10时，SIDL[15:0]有效<br>当IDL=11时，SIDL[7:0]有效 |

### SPI ID 高位寄存器(EXMC_SIDH)

偏移地址：0x350
复位值：0x0000 0000

该寄存器只能按字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | SIDH[63:32] | ID高位数据<br>仅在IDL=00时有效 |

# 26. 控制器局域网络(CAN)

## 26.1. 简介

CAN(Controller Area Network)总线是一种可以在无主机情况下实现微处理器或者设备之间相互通信的总线标准。

CAN 总线控制器作为 CAN 网络接口，遵循 CAN 总线协议 2.0A 和 2.0B。CAN 总线控制器可以处理总线上的数据收发并具有 28 个过滤器，过滤器用于筛选并接收用户需要的消息。用户可以通过 3 个发送邮箱将待发送数据传输至总线，邮箱发送的顺序由发送调度器决定。并通过 2 个深度为 3 的接收 FIFO 获取总线上的数据，接收 FIFO 的管理完全由硬件控制。同时 CAN 总线控制器硬件支持时间触发通信(Time-trigger communication)功能。

## 26.2. 主要特征

- 支持 CAN 总线协议 2.0A 和 2.0B；
- 通信波特率最大为 1Mbit/s；
- 支持时间触发通信；
- 中断使能和清除。

**发送功能**

- 3 个发送邮箱；
- 支持优先级发送；
- 支持发送时间戳。

**接收功能**

- 2 个深度为 3 的接收 FIFO；
- 具有 28 个过滤器；
- FIFO 锁定功能。

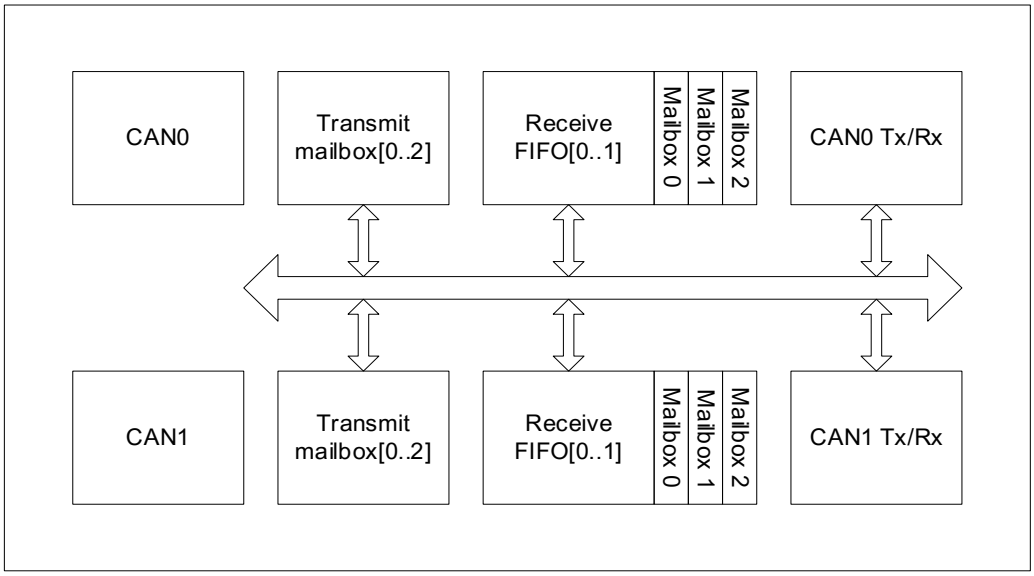
**时间触发通信**

- 在时间触发通信模式下取消自动重传；
- 16 位定时器；
- 接收时间戳；
- 发送时间戳。

## 26.3. 功能说明

CAN 模块结构框图如*图 26-1. CAN 模块结构框图*所示。

**图 26-1. CAN 模块结构框图**




### 26.3.1. 工作模式

CAN 总线控制器有 3 种工作模式：

- 睡眠工作模式；
- 初始化工作模式；
- 正常工作模式。

#### 睡眠工作模式

芯片复位后，CAN总线控制器处于睡眠工作模式。该模式下CAN总线控制器的时钟停止工作并处于一种低功耗状态。

将CAN_CTL寄存器的SLPWMOD位置1，可以使CAN总线控制器进入睡眠工作模式。当CAN进入睡眠工作模式后，CAN_STAT寄存器的SLPWS位将被硬件置1。

将CAN_CTL寄存器的AWU位置1，并当CAN检测到总线活动时，CAN总线控制器将自动退出睡眠工作模式。将CAN_CTL寄存器的SLPWMOD位清0，也可以退出睡眠工作模式。

由睡眠模式进入初始化工作模式：将CAN_CTL寄存器的IWMOD位置1，SLPWMOD位清0。

由睡眠模式进入正常工作模式：将CAN_CTL寄存器的IWMOD位和SLPWMOD位清0。

#### 初始化工作模式

如果需要配置 CAN 总线通信参数，CAN 总线控制器必须进入初始化工作模式。将 CAN_CTL 寄存器的 IWMOD 位置 1，使 CAN 总线控制器进入初始化工作模式，将其清 0 则离开初始化工作模式。在进入初始化工作模式后，CAN_STAT 寄存器的 IWS 位将被硬件置 1。

由初始化模式进入睡眠模式：CAN_CTL 寄存器的 SLPWMOD 位置 1，IWMOD 位清 0。

由初始化模式进入正常工作模式：CAN_CTL 寄存器的 SLPWMOD 位和 IWMOD 位清 0。

#### 正常工作模式

在初始化工作模式中配置完 CAN 总线通信参数后，将 CAN_CTL 寄存器的 IWMOD 位清 0 可以进入正常工作模式并与 CAN 总线网络中的节点进行正常通信。

由正常工作模式进入睡眠模式：CAN_CTL 寄存器的 SLPWMOD 位置 1，并等待当前数据收发过程结束。

由正常工作模式初始化模式：CAN_CTL 寄存器的 IWMOD 位置 1，并等待当前数据收发过程结束。

### 26.3.2. 通信模式

CAN 总线控制器有 4 种通信模式：

- 静默(Silent)通信模式；
- 回环(Loopback)通信模式；
- 回环静默(Loopback and Silent)通信模式；
- 正常(Normal)通信模式。

#### 静默(Silent)通信模式

在静默通信模式下，可以从 CAN 总线接收数据，但不向总线发送任何数据。将 CAN_BT 寄存器中的 SCMOD 位置 1，使 CAN 总线控制器进入静默通信模式，将其清 0 可以退出静默通信模式。

静默通信模式可以用来监控 CAN 网络上的数据传输。

#### 回环(Loopback)通信模式

在回环通信模式下，由 CAN 总线控制器发送的数据可以被自己接收并存入接收 FIFO，同时这些发送数据也送至 CAN 网络。将 CAN_BT 寄存器中的 LCMOD 位置 1，使 CAN 总线控制器进入回环通信模式，将其清 0 可以退出回环通信模式。

回环通信模式通常用来进行 CAN 通信自测。

#### 回环静默(Loopback and Silent)通信模式

在回环静默通信模式下，CAN 的 RX 和 TX 引脚与 CAN 网络断开。CAN 总线控制器既不从 CAN 网络接收数据，也不向 CAN 网络发送数据，其发送的数据仅可以被自己接收。将 CAN_BT 寄存器中的 LCMOD 位和 SCMOD 位置 1，使 CAN 总线控制器进入回环静默通信模式，将它们清 0 可以退出回环静默通信模式。

回环静默通信模式通常用来进行 CAN 通信自测。对外 TX 引脚保持隐性状态(逻辑 1)，RX 引脚保持高阻态。

#### 正常(Normal)通信模式

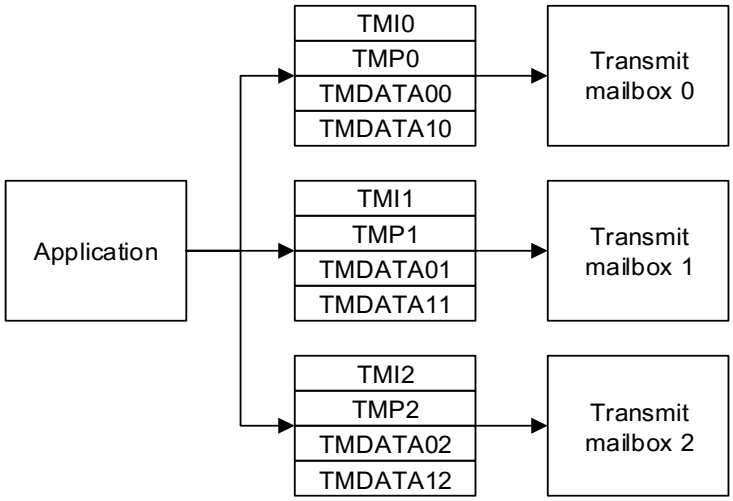
CAN 总线控制器通常工作在正常通信模式下，可以从 CAN 总线接收数据，也可以向 CAN 总线发送数据。这时需要将 CAN_BT 寄存器中的 LCMOD 和 SCMOD 清 0。

### 26.3.3. 数据发送

#### 发送寄存器

数据发送通过 3 个发送邮箱进行，可以通过寄存器 CAN_TMIx，CAN_TMPx，CAN_TMDATA0x 和 CAN_TMDATA1x 对发送邮箱进行配置。如*图 26-2. 发送寄存器*所示。

**图 26-2. 发送寄存器**



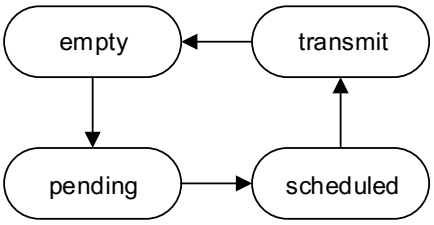

#### 发送邮箱状态转换

当发送邮箱处于 **empty** 状态时，应用程序才可以对邮箱进行配置。当邮箱被配置完成后，

可以将 CAN_TMIx 寄存器 TEN 置 1，从而向 CAN 总线控制器提交发送请求，这时发送

邮箱处于 **pending** 状态。当超过 1 个邮箱处于 **pending** 状态时，需要对多个邮箱进行调度，这时发送邮箱处于 **scheduled** 状态。当调度完成后，发送邮箱中的数据开始向 CAN 总线发送数据，这时发送邮箱处于 **transmit** 状态。当数据发送完成，邮箱变为空闲，可以再次交给应用程序使用，这时发送邮箱重新变为 **empty** 状态。如*图 26-3. 发送邮箱状态转换*所示。

**图 26-3. 发送邮箱状态转换**




### 发送状态和错误信息

CAN_TSTAT寄存器中的MTF，MTFNERR，MAL和MTE用来说明发送状态和错误信息。

- MTF：发送完成标志位。当数据发送完成时，MTF 置 1。
- MTFNERR：无错误发送完成标志位。当数据发送完成且没有错误时，MTFNERR 置 1。
- MAL：仲裁失败标志位。当发送数据过程中出现仲裁失败时，MAL置1。
- MTE：发送错误标志位。当发送过程中检测到总线错误时，MTE置1。

### 数据发送步骤

数据发送步骤如下：

第1步：选择一个空闲发送邮箱；
第2步：根据应用程序要求，配置4个发送寄存器；
第3步：将CAN_TMIx寄存器的TEN置1；
第4步：检测发送状态和错误信息。典型情况是检测到MTF和MTFNERR置1，说明数据被成功发送。

### 发送选项

#### 中止数据发送

将CAN_TSTAT寄存器的MST置1，可以中止数据发送。
当发送邮箱处于**pending**和**scheduled**状态，CAN_TSTAT寄存器的MST置1可以立即中止数据发送。
当发送邮箱处于**transmit**状态，则面临两种情况。一种情况是数据发送被成功地完成，MTF和MTFNERR为1，这时发送邮箱将转换为**empty**状态。相对的，如果数据发送过程中出现了问题，

这时发送邮箱将转换为**scheduled**状态，这时数据发送被中止。

**发送优先级**

当有2个及其以上发送邮箱等待发送时，寄存器CAN_CTL的TFO可以决定发送顺序。

当TFO为1，所有等待发送的邮箱按照先来先发送(FIFO)的顺序进行。

当TFO为0，具有最小标识符(Identifier)的邮箱最先发送。如果所有的标识符(Identifier)相等，具有最小邮箱编号的邮箱最先发送。

## 26.3.4. 数据接收

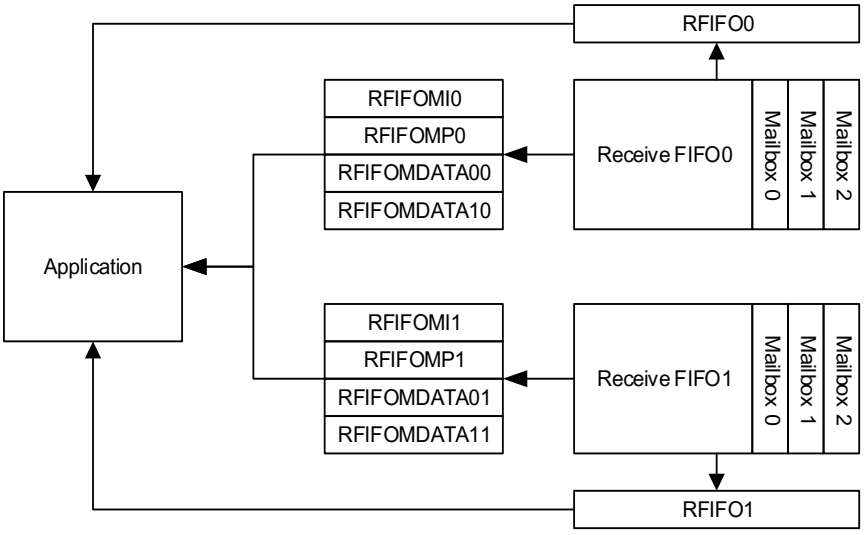
### 接收寄存器

应用程序通过2个深度为3的FIFO接收来自CAN网络的数据。

寄存器CAN_RFIFOx可以操作FIFO，也包含FIFO状态。寄存器CAN_RFIFOMIx，CAN_RFIFOMPx，CAN_RFIFOMDATA0x和CAN_RFIFOMDATA1x用于接收数据帧。

如*图26-4. 接收寄存器*所示。

**图 26-4. 接收寄存器**




### 接收 FIFO

每个接收FIFO包含3个接收邮箱，用来存储接收数据帧。这些邮箱按照先进先出方式进行组织，最早从CAN网络接收的数据，最早被应用程序处理。

寄存器CAN_RFIFOx包含FIFO状态信息和帧的数量。当FIFO中包含数据时，可以通过寄存器CAN_RFIFOMIx，CAN_RFIFOMPx，CAN_RFIFOMDATA0x和CAN_RFIFOMDATA1x读取数据，之后将寄存器CAN_RFIFOx的RFD置1释放邮箱。

### 接收 FIFO 状态信息

接收FIFO状态信息包含在寄存器CAN_RFIFOx中。

RFL：FIFO中包含的帧数量。FIFO为空时，RFL为0；FIFO为满时，RFL为3。

RFF：FIFO满状态标志位。这时RFL为3。

RFO：FIFO溢出标志位。当FIFO已经包含了3个数据帧时，新的数据帧到来使FIFO发生溢出。如果CAN_CTL寄存器的RFOD位被置1，新的数据帧将丢弃。如果该位被清0，新的数据帧将覆盖接收FIFO中最后一帧数据。

### 数据接收步骤

第1步：查看FIFO中帧的数量；
第 2 步 ： 通 过 CAN_RFIFOMIx ， CAN_RFIFOMPx ， CAN_RFIFOMDATA0x 和 CAN_RFIFOMDATA1x读取数据；
第3步：将寄存器CAN_RFIFOx的RFDx置1释放邮箱，并且等待其由硬件自动清0。

## 26.3.5. 过滤功能

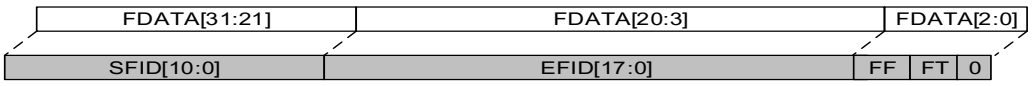
一个待接收的数据帧会根据其标识符(Identifier)进行过滤：硬件会将通过过滤的帧送至接收FIFO，并丢弃没有通过过滤的帧。

### 过滤器位宽

过滤器包含28个单元，它们是bank0到bank27。每一个过滤器单元有2个寄存器CAN_FxDATA0和CAN_ FxDATA1，它们可以配置为2种位宽：32-bit位宽和16-bit位宽。

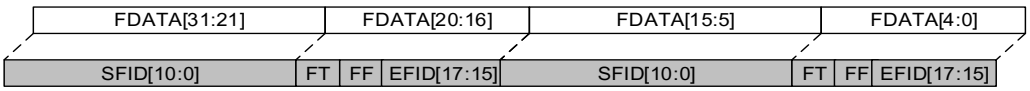
32-bit位宽CAN_FDATA包含字段：SFID[10:0]，EFID[17:0]，FF和FT。如*图26-5. 32-bit位宽过滤器*所示。

**图 26-5. 32-bit 位宽过滤器**




16-bit位宽CAN_FDATA包含字段：SFID[10:0]，FT，FF和EFID[17:15]。如*图26-6. 16-bit位宽过滤器*所示。

**图 26-6. 16-bit 位宽过滤器**



### 掩码模式

对于一个待过滤的数据帧的标识符(Identifier)，掩码模式用来指定哪些位必须与预设的标识符相同，哪些位无需判断。

一个32-bit位宽掩码模式过滤器如*图26-7. 32-bit位宽掩码模式过滤器*所示。

**图 26-7. 32-bit 位宽掩码模式过滤器**

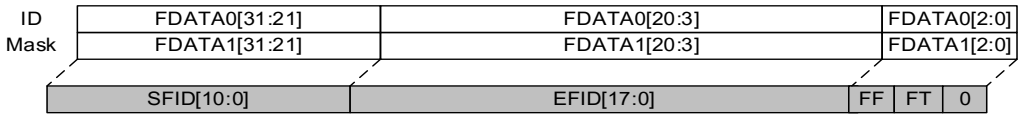
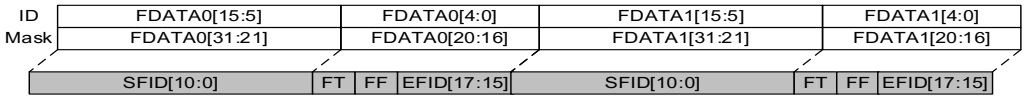



**图 26-8. 16-bit 位宽掩码模式过滤器**




### 列表模式

对于一个接收数据帧的标识符(Identifier)，列表模式用来表示与预设的标识符列表中能够匹配则通过，否则丢弃。

一个32-bit位宽列表模式过滤器如*图26-9. 32-bit位宽列表模式过滤器*所示。

**图 26-9. 32-bit 位宽列表模式过滤器**

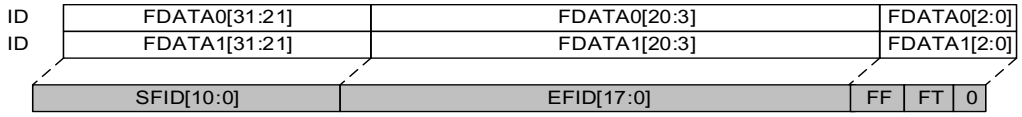
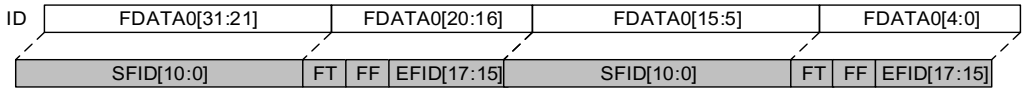



**图 26-10. 16-bit 位宽列表模式过滤器**




### 过滤序号

过滤器由若干过滤单元(Bank)组成，每个过滤单元因为位宽和模式的选择不同，而具有不同的过滤效果。例如*表26-1. 32-bit过滤序号*所示的2个过滤单元，Bank0是32-bit位宽掩码模式，Bank1是32-bit位宽列表模式。

**表 26-1. 32-bit 过滤序号**

| 过滤单元 | 过滤数据寄存器 | 过滤序号 |
|---|---|---|
| 0 | F0DATA0-32bit-ID | 0 |
| | F0DATA1-32bit-Mask | |
| 1 | F1DATA0-32bit-ID | 1 |

| | | |
|---|---|---|
| | F1DATA1-32bit-ID | 2 |

### 过滤器关联的 FIFO

28个过滤单元均可以关联接收FIFO0或接收FIFO1。一旦一个过滤单元关联到接收FIFO，只有通过这个过滤单元的帧才会被传送到接收FIFO中存储。

### 过滤器激活控制

一个过滤单元如果被应用程序用到，就必须激活。通过CAN_FW寄存器可以进行配置。

### 过滤索引

一个包含过滤序号(Fiter Number)N的过滤单元通过了某个帧，则该帧数据的过滤索引(Filtering Index)为N。这时CAN_RFIFOMPx中FI的值为N。*表26-2. 过滤索引*是一个过滤索引的例子。

在*表26-2. 过滤索引*中，如果一个帧通过了FIFO0中过滤序号10(Filter Number=10)的过滤单元，那么该帧的过滤索引为10。这时CAN_RFIFOMPx中FI的值为10。

过滤序号不关心对应的过滤单元(Bank)是否处于工作状态。例如Bank3被关联到FIFO0，且为"不激活"状态，但它仍然包含过滤序号3和4。

**表 26-2. 过滤索引**

| 过滤单元 | FIFO0 | 是否激活 | 过滤序号 | 过滤单元 | FIFO1 | 是否激活 | 过滤序号 |
|---|---|---|---|---|---|---|---|
| 0 | F0DATA0-32bit-ID | Yes | 0 | 2 | F2DATA0[15:0]-16bit-ID | Yes | 0 |
| | F0DATA1-32bit-Mask | | | | F2DATA0[31:16]-16bit-Mask | | |
| 1 | F1DATA0-32bit-ID | Yes | 1 | | F2DATA1[15:0]-16bit-ID | | 1 |
| | F1DATA1-32bit-ID | | 2 | | F2DATA1[31:16]-16bit-Mask | | |
| 3 | F3DATA0[15:0]-16bit-ID | No | 3 | 4 | F4DATA0-32bit-ID | No | 2 |
| | F3DATA0[31:16]-16bit-Mask | | | | F4DATA1-32bit-Mask | | |
| | F3DATA1[15:0]-16bit-ID | | 4 | 5 | F5DATA0-32bit-ID | No | 3 |
| | F3DATA1[31:16]-16bit-Mask | | | | F5DATA1-32bit-ID | | 4 |
| 7 | F7DATA0[15:0]-16bit-ID | No | 5 | 6 | F6DATA0[15:0]-16bit-ID | Yes | 5 |
| | F7DATA0[31:16]-16bit-ID | | 6 | | F6DATA0[31:16]-16bit-ID | | 6 |
| | F7DATA1[15:0]-16bit-ID | | 7 | | F6DATA1[15:0]-16bit-ID | | 7 |
| | F7DATA1[31:16]-16bit-ID | | 8 | | F6DATA1[31:16]-16bit-ID | | 8 |
| 8 | F8DATA0[15:0]-16bit-ID | Yes | 9 | 10 | F10DATA0[15:0]-16bit-ID | No | 9 |
| | F8DATA0[31:16]-16bit-ID | | 10 | | F10DATA0[31:16]-16bit-Mask | | |
| | F8DATA1[15:0]-16bit-ID | | 11 | | F10DATA1[15:0]-16bit-ID | | 10 |
| | F8DATA1[31:16]-16bit-ID | | 12 | | F10DATA1[31:16]-16bit-Mask | | |
| 9 | F9DATA0[15:0]-16bit-ID | Yes | 13 | 11 | F11DATA0[15:0]-16bit-ID | No | 11 |

| 过滤单元 | FIFO0 | 是否激活 | 过滤序号 | 过滤单元 | FIFO1 | 是否激活 | 过滤序号 |
|---|---|---|---|---|---|---|---|
| | F9DATA0[31:16]-16bit-Mask | | | | F11DATA0[31:16]-16bit-ID | | 12 |
| | F9DATA1[15:0]-16bit-ID | | 14 | | F11DATA1[15:0]-16bit-ID | | 13 |
| | F9DATA1[31:16]-16bit-Mask | | | | F11DATA1[31:16]-16bit-ID | | 14 |
| 12 | F12DATA0-32bit-ID | Yes | 15 | 13 | F13DATA0-32bit-ID | Yes | 15 |
| | F12DATA1-32bit-Mask | | | | F13DATA1-32bit-ID | | 16 |

#### 优先级

过滤器优先级顺序如下：

1. 32-bit位宽模式高于16-bit位宽模式；
2. 列表模式高于掩码模式；

较小的过滤序号(Filter Number)具有较高的优先级。

### 26.3.6. 时间触发通信

时间触发通信是CAN数据链路层应用协议。CAN网络中的所有节点都按照一个预先设定的时间序列进行通信，尤其适合于时间周期性应用和时间确定性应用。

在这种通信模式下，一个内部的16-bit计数器开始工作，在每一个CAN位时间(Bit time)增1。这个内部计数器为数据发送和数据接收提供时间戳，这些时间戳存放在寄存器CAN_RFIFOMPx和CAN_TMPx中。

在这种通信模式下，自动重发功能是禁止的。

### 26.3.7. 通信参数

#### 自动重发禁止模式

在时间触发通信模式下，要求自动重发必须是禁止的，可以通过将CAN_CTL寄存器的ARD置1满足要求。

在这种模式下，数据只会被发送一次，如果因为仲裁失败或者总线错误而导致发送失败，CAN总线控制器不会像通常那样进行数据自动重发。

发送结束时，寄存器CAN_TSTAT中的MTF置1，而发送状态信息可以通过MTFNERR，MAL和MTE获得。

#### 位时序(Bit time)

CAN协议采用位同步传输方式。这种方式不仅增大了传输容量，而且意味着需要一种复杂的位同步方法。面向字节传输的位同步方式适用于接收在每个字节前都有起始位的情况，而同步传输协议只要求数据帧的最开始有一个起始位。为保证接收器能正确读取信息，需要不断地进行

重新同步。因此，在相位缓冲段采样点前面和后面都应该插入一个帧间隔。

可以通过位操作仲裁方式访问CAN总线。信号从发送器到接收器，再回到发送器必须在一个位时间内完成。为了达到同步的目的，除了相位缓冲段外，还需要一个传输延时段。在信号传输过程中，传输延时段被视为发送或接收延时。

CAN总线控制器将位时间分为3个部分。

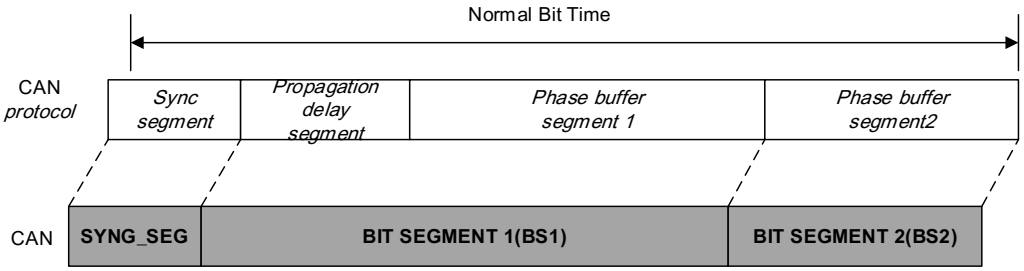
同步段(Synchronization segment)，记为SYNC_SEG。该段占用1个时间单元($1 \times t_q$)。

位段1(Bit segment 1)，记为BS1。该段占用1到16个时间单元。相对于CAN协议而言，BS1相当于传播时间段(Propagation delay segment)和相位缓冲段1(Phase buffer segment 1)。

位段2(Bit segment 2)，记为BS2。该段占用1到8个时间单元。相对于CAN协议而言，BS2相当于相位缓冲段2(Phase buffer segment 2)。

对比与CAN协议，位时序如*图26-11. 位时序*所示。

**图 26-11. 位时序**




再同步补偿宽度SJW(resynchronization Jump Width)对CAN网络节点同步误差进行补偿，占用1到4个时间单元。

有效跳变沿定义为只有当在前一个采样点检测到的总线状态为隐性(recessive)时，一个位时间内从隐性位到显性位的第一次转变。

如果有效跳变在BS1期间被检测到，而不是SYNC_SEG期间，BS1将会最多被延长SJW，因此采样点延时。

相反，如果有效跳变在BS2期间被检测到，而不是SYNC_SEG期间，BS2将会最多被缩短SJW，因此采样点提前。

**注：**有关硬同步和再同步的详细说明，请参考ISO 11898标准。

### 波特率

CAN时钟从APB1总线上获得，波特率计算公式如下：

$$BaudRate=\frac{1}{Normal\ Bit\ Time} \quad (式26\text{-}1)$$

$$Normal\ Bit\ Time=t_{SYNC_SEG}+t_{BS1}+t_{BS2} \quad (式26\text{-}2)$$

其中：

$$t_{SYNC_SEG}=1\times t_q \quad (式26\text{-}3)$$

$$t_{BS1}=(1+BT.BS1)\times t_q \quad (式26\text{-}4)$$

$$t_{BS2}=(1+BT.BS2)\times t_q \quad (式26\text{-}5)$$

$$t_q=(1+BT.BAUDPSC)\times t_{PCLK1} \quad (式26\text{-}6)$$

### 26.3.8. 错误标志

CAN总线的状态可以通过CAN_ERR寄存器的发送错误计数值(Transmit Error Counter，记为TECNT)和接收错误计数值(Receive Error Counter，记为RECNT)反映，其值会根据错误的情况由硬件增加或减少，软件可以通过这些值判断CAN网络的稳定性。关于错误计数值的详细信息请参考CAN协议相关章节。

通过使能CAN_INTEN寄存器中的相应位(ERRIE等)，软件可以在检测到错误时产生相应中断。

#### 离线恢复

当TECNT大于255时，CAN总线控制器进入离线状态，这时寄存器CAN_ERR中的BOERR置1，并且发送和接收失效。

根据寄存器CAN_CTL中的ABOR配置，离线恢复(变为主动错误状态)有2种方式。这两种方式都要求处于离线状态的CAN总线控制器检测到CAN协议所定义的离线恢复序列(在CAN_RX检测到128次连续11个位的隐性位)时，才会自动恢复。

如果ABOR为1，将在检测到离线恢复序列后自动恢复。

如果ABOR为0，则必须先将CAN_CTL中的IWMOD置1进入初始化工作模式，然后进入正常工作模式并在检测到离线恢复序列后恢复。

### 26.3.9. 中断

CAN总线控制器占用4个中断向量，通过寄存器CAN_INTEN进行控制。这4个中断向量对应4类中断源：

- 发送中断；
- FIFO0 中断；
- FIFO1 中断；
- 错误和状态改变中断。

#### 发送中断

在CAN_INTEN寄存器中TMEIE置1时，发送中断可由以下条件之一产生：

- 寄存器CAN_TSTAT中的MTF0置1：发送邮箱0变为空闲；
- 寄存器CAN_TSTAT中的MTF1置1：发送邮箱1变为空闲；
- 寄存器CAN_TSTAT中的MTF2置1：发送邮箱2变为空闲。

### FIFO0 中断

FIFO0中断包括：

- FIFO0中包含待接收数据：寄存器CAN_RFIFO0中的RFL0不为0，CAN_INTEN寄存器中RFNEIE0被置位；
- FIFO0满：寄存器CAN_RFIFO0中的RFF0为1，CAN_INTEN寄存器中RFFIE0被置位；
- FIFO0溢出：寄存器CAN_RFIFO0中的RFO0为1，CAN_INTEN寄存器中RFOIE0被置位。

### FIFO1 中断

FIFO1中断包括：

- FIFO1中包含待接收数据：寄存器CAN_RFIFO1中的RFL1不为0，CAN_INTEN寄存器中RFNEIE1被置位；
- FIFO1满：寄存器CAN_RFIFO1中的RFF1为1，CAN_INTEN寄存器中RFFIE1被置位；
- FIFO1溢出：寄存器CAN_RFIFO1中的RFO1为1，CAN_INTEN寄存器中RFOIE1被置位。

### 错误和工作模式改变中断

错误和工作模式改变中断可由以下条件触发：

- 错误：CAN_STAT寄存器的ERRIF和CAN_INTEN寄存器的ERRIE被置位，请参考CAN_STAT寄存器中ERRIF位描述；
- 唤醒：CAN_STAT寄存器中的WUIF和CAN_INTEN寄存器的WIE被置位；
- 进入睡眠模式：CAN_STAT寄存器中的SLPIF和CAN_INTEN寄存器的SLPWIE被置位。

CAN总线控制器的中断产生条件可参考*表26-3. CAN事件/中断标志*。

**表 26-3. CAN 事件/中断标志**

| 中断事件 | 事件/中断标志 | | 使能控制位 | |
|---|---|---|---|---|
| 发送中断 | 发送邮箱 0 空闲标志 MTF0 | | TMEIE | |
| | 发送邮箱 1 空闲标志 MTF1 | | | |
| | 发送邮箱 2 空闲标志 MTF2 | | | |
| FIFO0 中断 | 接收 FIFO0 中帧的数量 RFL0[1:0] | | RFNEIE0 | |
| | 接收 FIFO0 满 RFF0 | | RFFIE0 | |
| | 接收 FIFO0 溢出 RFO0 | | RFOIE0 | |
| FIFO1 中断 | 接收 FIFO1 中帧的数量 RFL1[1:0] | | RFNEIE1 | |
| | 接收 FIFO1 满 RFF1 | | RFFIE1 | |
| | 接收 FIFO1 溢出 RFO1 | | RFOIE1 | |
| EWMC 中断 | 警告错误 WERR | 错误中断标志 ERRIF | WERRIE | ERRIE |
| | 被动错误 PERR | | PERRIE | |
| | 离线错误 BOERR | | BOIE | |

| 中断事件 | 事件/中断标志 | 使能控制位 |
| --- | --- | --- |
| | 错误种类 1<= ERRN[2:0] <= 6 | ERRNIE |
| | 从睡眠工作模式唤醒的状态改变中断标志WUIF | WIE |
| | 进入睡眠工作模式的状态改变中断标志SLPIF | SLPWIE |

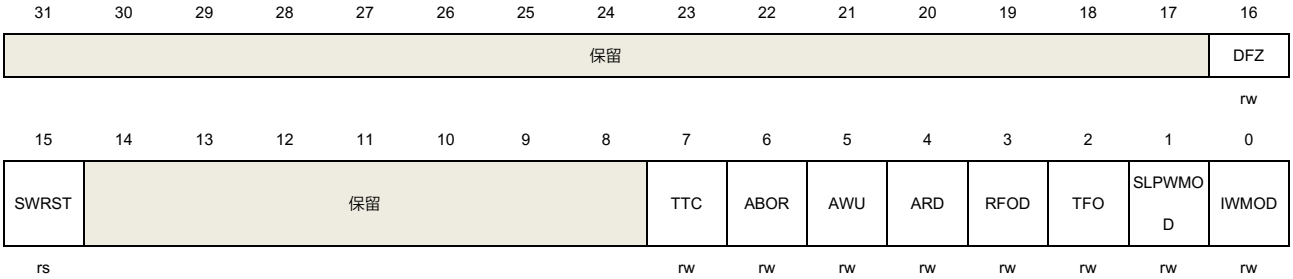
## 26.4. CAN 寄存器

CAN0基地址：0x4000 6400
CAN1基地址：0x4000 6800

### 26.4.1. 控制寄存器(CAN_CTL)

地址偏移：0x00
复位值：0x0001 0002

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:17 | 保留 | 必须保持复位值。 |
| 16 | DFZ | 调试冻结<br>如果 DBG_CTL0 寄存器中 CANx_HOLD 被置位，该位用来定义 CAN 控制器工作在调试冻结或正常工作状态。如果 DBG_CTL0 寄存器中 CANx_HOLD 被清零，该位无效。<br>0：处于 Debug 时，CAN 接收和发送正常工作<br>1：处于 Debug 时，CAN 接收和发送停止 |
| 15 | SWRST | 软件复位<br>0：正常操作<br>1：复位 CAN 并进入睡眠工作模式。该位会自动清 0 |
| 14:8 | 保留 | 必须保持复位值。 |
| 7 | TTC | 时间触发通信<br>0：禁用时间触发通信<br>1：使能时间触发通信 |
| 6 | ABOR | 自动离线恢复<br>0：通过软件手动地从离线状态恢复<br>1：通过硬件自动的从离线状态恢复 |
| 5 | AWU | 自动唤醒<br>一旦自动唤醒后，CAN_CTL 寄存器的 SLPWMOD 位将自动被清 0。<br>0：通过软件手动的从睡眠工作模式唤醒 |

| | | |
|---|---|---|
| | | 1：通过硬件自动的从睡眠工作模式唤醒 |
| 4 | ARD | 自动重发禁止<br>0：使能自动重发<br>1：禁用自动重发 |
| 3 | RFOD | 禁用接收 FIFO 满时覆盖<br>0：使能接收 FIFO 满时覆盖。当接收 FIFO 满时，FIFO 中的数据被新来的数据覆盖<br>1：禁用接收 FIFO 满时覆盖。当接收 FIFO 满时，新来的数据被丢弃，FIFO 中的数据保持不变，不会被覆盖 |
| 2 | TFO | 发送 FIFO 顺序<br>0：标识符(Identifier)较小的帧先发送<br>1：所有等待发送的邮箱按照先进先出(FIFO)的顺序发送 |
| 1 | SLPWMOD | 睡眠工作模式<br>如果软件将该位置 1，CAN 将会在当前发送或接收完成时进入睡眠工作模式。该位可由软件或者硬件清 0。如果 CAN_CTL 寄存器中 AWU 被置位，当检测到 CAN 总线工作时，该位被清 0。<br>0：禁用睡眠工作模式<br>1：使能睡眠工作模式 |
| 0 | IWMOD | 初始化工作模式<br>0：禁用初始化工作模式<br>1：使能初始化工作模式 |

## 26.4.2. 状态寄存器(CAN_STAT)

地址偏移：0x04
复位值：0x0000 0C02

该寄存器只能按字(32位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | RXL | LASTRX | RS | TS | 保留 | | | SLPIF | WUIF | ERRIF | SLPWS | IWS |
| | | | | r | r | r | r | | | | rc_w1 | rc_w1 | rc_w1 | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11 | RXL | RX 引脚电平 |
| 10 | LASTRX | RX 引脚最近一次的采样值 |
| 9 | RS | 接收状态 |

| | | |
|---|---|---|
| | | 0：CAN 当前不是接收器<br>1：CAN 当前是接收器 |
| 8 | TS | 发送状态<br>0：CAN 当前不是发送器<br>1：CAN 当前是发送器 |
| 7:5 | 保留 | 必须保持复位值。 |
| 4 | SLPIF | 进入睡眠工作模式的状态改变中断标志<br>该位在进入睡眠工作模式时由硬件置位。当 CAN 不再处于睡眠工作模式时由硬件清零。该位也可以由软件写 1 清 0。<br>0：CAN 没有进入睡眠工作模式<br>1：CAN 进入睡眠工作模式。如果相应的中断使能位为 1，则发生中断 |
| 3 | WUIF | 从睡眠工作模式唤醒的状态改变中断标志<br>该位在睡眠工作模式时检测到 CAN 总线上的活动时由硬件置位。该位由软件写 1 清 0。<br>0：没有检测到唤醒信号<br>1：发现唤醒信号。如果相应的中断使能位为 1，则发生中断 |
| 2 | ERRIF | 错误中断标志<br>该位由以下事件置位。CAN_ERR 寄存器中 BOERR 位和 CAN_INTEN 寄存器中 BOIE 位都置位。或 CAN_ERR 寄存器中 PERR 位和 CAN_INTEN 寄存器中 PERRIE 位都置位。或 CAN_ERR 寄存器中 WERR 位和 CAN_INTEN 寄存器中 WERRIE 位都置位。或 CAN_ERR 寄存器中 ERRN 位域的值不为 0 且 CAN_INTEN 寄存器中 ERRNIE 位置位。该位由软件写 1 清零。<br>0：没有错误<br>1：发生错误。如果相应的中断使能位为 1，则发生中断 |
| 1 | SLPWS | 睡眠工作状态<br>将 CAN_CTL 寄存器中 SLPWMOD 位置位进入睡眠工作模式后该位由硬件置位。当 CAN 由正常通信模式切换到睡眠工作模式，需等待当前发送过程或者接收过程完成。当 CAN 离开睡眠工作模式(清除 CAN_CTL 寄存器中 SLPWMOD 位或是在 CAN_CTL 寄存器中 AWU 置位时检测到 CAN 总线上的活动)时，该位由硬件清零。如果由睡眠工作模式切换到正常工作模式，该位在 CAN 接收到来自总线的连续 11 个隐性位后被清 0。<br>0：CAN 没有处于睡眠工作状态<br>1：CAN 处于睡眠工作状态 |
| 0 | IWS | 初始化工作状态<br>将 CAN_CTL 寄存器中 IWMOD 位置位进入初始化模式后该位由硬件置位。当 CAN 由正常通信模式切换到初始化工作模式，需等待当前发送过程或者接收过程完成。在清除 CAN_CTL 寄存器中 IWMOD 位离开初始化模式后，该位由硬件清 0。如果由初始化工作模式切换到正常工作模式，该位在 CAN 接收到来自总线的连续 11 个隐性位后被清 0。<br>0：CAN 没有处于初始化工作状态 |

1：CAN 处于初始化工作状态

### 26.4.3. 发送状态寄存器(CAN_TSTAT)

地址偏移：0x08
复位值：0x1C00 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMLS2 | TMLS1 | TMLS0 | TME2 | TME1 | TME0 | NUM[1:0] | | MST2 | 保留 | | | MTE2 | MAL2 | MTFNERR2 | MTF2 |
| r | r | r | r | r | r | r | | rs | | | | rc_w1 | rc_w1 | rc_w1 | rc_w1 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MST1 | 保留 | | | MTE1 | MAL1 | MTFNERR1 | MTF1 | MST0 | 保留 | | | MTE0 | MAL0 | MTFNERR0 | MTF0 |
| rs | | | | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rs | | | | rc_w1 | rc_w1 | rc_w1 | rc_w1 |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | TMLS2 | 在发送 FIFO 中邮箱 2 最后发送<br>该位为 1 表明，当有 2 个及其以上帧等待发送时，发送邮箱 2 具有最后的发送顺序。 |
| 30 | TMLS1 | 在发送 FIFO 中邮箱 1 最后发送<br>该位为 1 表明，当有 2 个及其以上帧等待发送时，发送邮箱 1 具有最后的发送顺序。 |
| 29 | TMLS0 | 在发送 FIFO 中邮箱 0 最后发送<br>该位为 1 表明，当有 2 个及其以上帧等待发送时，发送邮箱 0 具有最后的发送顺序。 |
| 28 | TME2 | 发送邮箱 2 空<br>0：发送邮箱 2 不为空<br>1：发送邮箱 2 空 |
| 27 | TME1 | 发送邮箱 1 空<br>0：发送邮箱 1 不为空<br>1：发送邮箱 1 空 |
| 26 | TME0 | 发送邮箱 0 空<br>0：发送邮箱 0 不为空<br>1：发送邮箱 0 空 |
| 25:24 | NUM[1:0] | 当发送 FIFO 不满时，NUM 表示下一个将要发送的邮箱号。<br>当发送 FIFO 满时，NUM 表示最后一个将要发送的邮箱号。 |
| 23 | MST2 | 邮箱 2 停止发送<br>将其置 1，将停止邮箱 2 的发送过程。 |

| | | |
|---|---|---|
| | | 当邮箱 2 变为 emtpy 状态时，该位被硬件自动清 0。 |
| 22:20 | 保留 | 必须保持复位值。 |
| 19 | MTE2 | 邮箱 2 发送错误<br>当发生发送错误时，该位由硬件置 1。由软件写 1 清 0 或对 CAN_TSTAT 寄存器中 MTF2 写 1 清 0。也可以在下一次发送开始时由硬件清 0。<br>当发生错误时该位被置 1。 |
| 18 | MAL2 | 邮箱 2 仲裁失败<br>当发生仲裁失败时，该位由硬件置 1。由软件写 1 清 0 或对 CAN_TSTAT 寄存器中 MTF2 写 1 清 0。也可以在下一次发送开始时由硬件清 0。<br>当发生仲裁失败时该位被置 1。 |
| 17 | MTFNERR2 | 邮箱 2 无错发送完成<br>当发送结束并且没有错误产生时，该位由硬件置1。由软件写1清0或对CAN_TSTAT 寄存器中MTF2写1清0。也可以在无错传输结束时由硬件清0。<br>0：传输结束时发生了错误<br>1：传输结束且没有错误 |
| 16 | MTF2 | 邮箱 2 发送完成<br>当发送完成或被中止时，该位由硬件置 1。由软件写 1 清 0，或当 CAN_TMI2 寄存器的 TEN 被置位时清 0。<br>0：发送邮箱 2 正在发送<br>1：发送邮箱 2 完成发送 |
| 15 | MST1 | 邮箱 1 停止发送<br>将其置 1，将停止邮箱 1 的发送过程。<br>当邮箱 1 变为 emtpy 状态时，该位被硬件自动清 0。 |
| 14:12 | 保留 | 必须保持复位值。 |
| 11 | MTE1 | 邮箱 1 发送错误<br>当发生发送错误时，该位由硬件置 1。由软件写 1 清 0 或对 CAN_TSTAT 寄存器中 MTF1 写 1 清 0。也可以在下一次发送开始时由硬件清 0。<br>当发生错误时该位被置 1。 |
| 10 | MAL1 | 邮箱 1 仲裁失败<br>当发生仲裁失败时，该位由硬件置 1。由软件写 1 清 0 或对 CAN_TSTAT 寄存器中 MTF1 写 1 清 0。也可以在下一次发送开始时由硬件清 0。<br>当发生仲裁失败时该位被置 1。 |
| 9 | MTFNERR1 | 邮箱 1 无错发送完成<br>当发送结束并且没有错误产生时，该位由硬件置1。由软件写1清0或对CAN_TSTAT 寄存器中MTF1写1清0。也可以在无错传输结束时由硬件清0。<br>0：传输结束时发生了错误<br>1：传输结束且没有错误 |
| 8 | MTF1 | 邮箱 1 发送完成<br>当发送完成或被中止时，该位由硬件置 1。由软件写 1 清 0，或当 CAN_TMI1 寄存 |

| | | |
|---|---|---|
| | | 器的 TEN 被置位时清 0。<br>0：发送邮箱 1 正在发送<br>1：发送邮箱 1 完成发送 |
| 7 | MST0 | 邮箱 0 停止发送<br>将其置 1，将停止邮箱 0 的发送过程。<br>当邮箱 0 变为 emtpy 状态时，该位被硬件自动清 0。 |
| 6:4 | 保留 | 必须保持复位值。 |
| 3 | MTE0 | 邮箱 0 发送错误<br>当发生发送错误时，该位由硬件置 1。由软件写 1 清 0 或对 CAN_TSTAT 寄存器中 MTF0 写 1 清 0。也可以在下一次发送开始时由硬件清 0。<br>当发生错误时该位被置 1。 |
| 2 | MAL0 | 邮箱 0 仲裁失败<br>当发生仲裁失败时，该位由硬件置 1。由软件写 1 清 0 或对 CAN_TSTAT 寄存器中 MTF0 写 1 清 0。也可以在下一次发送开始时由硬件清 0。<br>当发生仲裁失败时该位被置 1。 |
| 1 | MTFNERR0 | 邮箱 0 无错发送完成<br>当发送结束并且没有错误产生时，该位由硬件置1。由软件写1清0或对CAN_TSTAT寄存器中MTF0写1清0。也可以在无错传输结束时由硬件清0。<br>0：传输结束时发生了错误<br>1：传输结束且没有错误 |
| 0 | MTF0 | 邮箱 0 发送完成<br>当发送完成或被中止时，该位由硬件置 1。由软件写 1 清 0，或当 CAN_TMI0 寄存器的 TEN 被置位时清 0。<br>0：发送邮箱 0 正在发送<br>1：发送邮箱 0 完成发送 |

## 26.4.4. 接收 FIFO0 寄存器(CAN_RFIFO0)

地址偏移：0x0C
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | RFD0 | RFO0 | RFF0 | 保留 | RFL0[1:0] | |
| | | | | | | | | | | rs | rc_w1 | rc_w1 | | r | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:6 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 5 | RFD0 | 释放一次 FIFO0 中的数据<br>该位被置 1， 将释放 FIFO0 中的一帧数据。<br>FIFO 释放相应的数据空间后，该位被清 0。 |
| 4 | RFO0 | 接收 FIFO0 溢出<br>当接收 FIFO0 溢出时被置位，由软件写 1 清 0。<br>0：接收 FIFO0 没有溢出<br>1：接收 FIFO0 溢出 |
| 3 | RFF0 | 接收 FIFO0 满<br>当接收 FIFO0 满时被置位，由软件写 1 清 0。<br>0：接收 FIFO0 不满<br>1：接收 FIFO0 满 |
| 2 | 保留 | 必须保持复位值。 |
| 1:0 | RFL0[1:0] | 接收 FIFO0 中帧的数量 |

### 26.4.5. 接收 FIFO1 寄存器(CAN_RFIFO1)

地址偏移：0x10
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | RFD1 | RFO1 | RFF1 | 保留 | RFL1[1:0] | |
| | | | | | | | | | | rs | rc_w1 | rc_w1 | | r | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:6 | 保留 | 必须保持复位值。 |
| 5 | RFD1 | 释放一次 FIFO1 中的数据<br>该位被置 1，将释放 FIFO1 中的一帧数据。<br>FIFO 释放相应的数据空间后，该位被清 0。 |
| 4 | RFO1 | 接收 FIFO1 溢出<br>当接收 FIFO1 溢出时被置位，由软件写 1 清 0。<br>0：接收 FIFO1 没有溢出<br>1：接收 FIFO1 溢出 |
| 3 | RFF1 | 接收 FIFO1 满<br>当接收 FIFO1 满时被置位，由软件写 1 清 0。<br>0：接收 FIFO1 不满<br>1：接收 FIFO1 满 |

| | | |
|---|---|---|
| 2 | 保留 | 必须保持复位值。 |
| 1:0 | RFL1[1:0] | 接收 FIFO1 中帧的数量 |

### 26.4.6. 中断使能寄存器 (CAN_INTEN)

地址偏移：0x14
复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | | | | | SLPWIE | WIE |
| | | | | | | | | | | | | | | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ERRIE | 保留 | | | ERRNIE | BOIE | PERRIE | WERRIE | 保留 | RFOIE1 | RFFIE1 | RFNEIE1 | RFOIE0 | RFFIE0 | RFNEIE0 | TMEIE |
| rw | | | | rw | rw | rw | rw | | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:18 | 保留 | 必须保持复位值。 |
| 17 | SLPWIE | 睡眠中断使能<br>0：禁用睡眠中断<br>1：使能睡眠中断 |
| 16 | WIE | 唤醒中断使能<br>0：禁用唤醒中断<br>1：使能唤醒中断 |
| 15 | ERRIE | 错误中断使能<br>0：禁用错误中断<br>1：使能错误中断 |
| 14:12 | 保留 | 必须保持复位值。 |
| 11 | ERRNIE | 错误种类中断使能<br>0：禁用错误种类中断<br>1：使能错误种类中断 |
| 10 | BOIE | 离线中断使能<br>0：禁用离线中断<br>1：使能离线中断 |
| 9 | PERRIE | 被动错误中断使能<br>0：禁用被动错误<br>1：使能被动错误 |
| 8 | WERRIE | 警告错误中断使能<br>0：禁用警告错误中断 |

| | | |
|---|---|---|
| | | 1：使能警告错误中断 |
| 7 | 保留 | 必须保持复位值。 |
| 6 | RFOIE1 | 接收 FIFO1 溢出中断使能<br>0：禁用接收 FIFO1 溢出中断<br>1：使能接收 FIFO1 溢出中断 |
| 5 | RFFIE1 | 接收 FIFO1 满中断使能<br>0：禁用接收 FIFO1 满中断<br>1：使能接收 FIFO1 满中断 |
| 4 | RFNEIE1 | 接收 FIFO1 非空中断使能<br>0：禁用接收 FIFO1 非空中断<br>1：使能接收 FIFO1 非空中断 |
| 3 | RFOIE0 | 接收 FIFO0 溢出中断使能<br>0：禁用接收 FIFO0 溢出中断<br>1：使能接收 FIFO0 溢出中断 |
| 2 | RFFIE0 | 接收 FIFO0 满中断使能<br>0：禁用接收 FIFO0 满中断<br>1：使能接收 FIFO0 满中断 |
| 1 | RFNEIE0 | 接收 FIFO0 非空中断使能<br>0：禁用接收 FIFO0 非空中断<br>1：使能接收 FIFO0 非空中断 |
| 0 | TMEIE | 发送邮箱空中断使能<br>0：禁用发送邮箱空中断<br>1：使能发送邮箱空中断 |

### 26.4.7. 错误寄存器(CAN_ERR)

地址偏移：0x18
复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RECNT[7:0] | | | | | | | | TECNT[7:0] | | | | | | | |
| r | | | | | | | | r | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | ERRN[2:0] | | | 保留 | BOERR | PERR | WERR |
| | | | | | | | | | rw | | | | r | r | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | RECNT[7:0] | 接收错误计数值 |

| | | |
|---|---|---|
| 23:16 | TECNT[7:0] | 发送错误计数值 |
| 15:7 | 保留 | 必须保持复位值。 |
| 6:4 | ERRN[2:0] | 错误种类<br>ERRN 由硬件更新，可以反映位传输过程中的错误情况。当位传输成功没有错误时，ERRN 为 0。软件可以设置 ERRN 为 0b111。<br>000：无错误<br>001：填充错误<br>010：格式错误<br>011：ACK 错误<br>100：位隐性错<br>101：位显性错误<br>110：CRC 错误<br>111：软件设置值 |
| 3 | 保留 | 必须保持复位值。 |
| 2 | BOERR | 离线错误<br>当 TEC 上溢(超过 255)时，CAN 总线控制器进入离线状态，该位被置 1。 |
| 1 | PERR | 被动错误<br>当 TECNT 或者 RECNT 大于 127 时，该位由硬件置 1。 |
| 0 | WERR | 警告错误<br>当 TECNT 或 RECNT 大于等于 96 时，该位由硬件置 1。 |

### 26.4.8. 位时序寄存器(CAN_BT)

地址偏移：0x1C

复位值：0x0123 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SCMOD | LCMOD | 保留 | | | | SJW[1:0] | | 保留 | BS2[2:0] | | | BS1[3:0] | | | |
| rw | rw | | | | | rw | | | rw | | | rw | | | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | SCMOD | 静默通信模式<br>0：禁用静默通信模式<br>1：使能静默通信模式 |
| 30 | LCMOD | 回环通信模式<br>0：禁用回环通信模式 |

| | | |
|---|---|---|
| | | 1：使能回环通信模式 |
| 29:26 | 保留 | 必须保持复位值。 |
| 25:24 | SJW[1:0] | 再同步补偿宽度<br>再同步补偿占用的时间单元数量= SJW[1:0]+1 |
| 23 | 保留 | 必须保持复位值。 |
| 22:20 | BS2[2:0] | 位段 2<br>位段 2 占用的时间单元数量=BS2[2:0]+1 |
| 19:16 | BS1[3:0] | 位段 1<br>位段 1 占用的时间单元数量=BS1[3:0]+1 |
| 15:10 | 保留 | 必须保持复位值。 |
| 9:0 | BAUDPSC[9:0] | 波特率分频系数 |

### 26.4.9. 发送邮箱标识符寄存器(CAN_TMIx)(x = 0...2)

地址偏移：0x180，0x190，0x1A0
复位值：0xXXXX XXXX(bit0 = 0)

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SFID[10:0]/EFID[28:18] | | | | | | | | | | | EFID[17:13] | | | | |
| rw | | | | | | | | | | | rw | | | | |

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

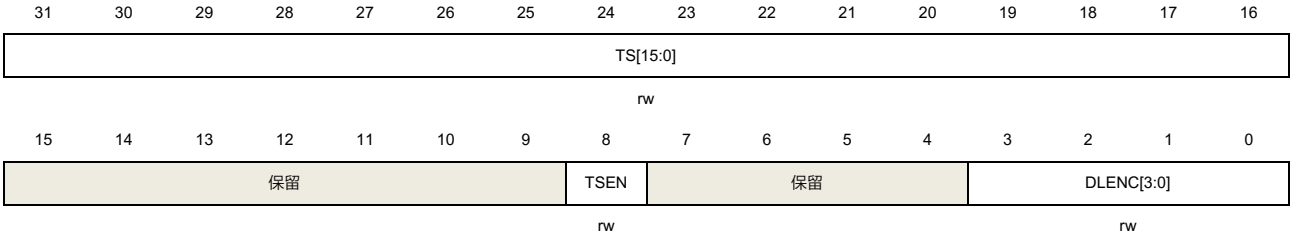
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:21 | SFID[10:0]/EFID[28:18] | 标识符<br>SFID[10:0]：标准格式帧标识符<br>EFID[28:18]：扩展格式帧标识符 |
| 20:16 | EFID[17:13] | 标识符<br>EFID[17:13]：扩展格式帧标识符 |
| 15:3 | EFID[12:0] | 标识符<br>EFID[12:0]：扩展格式帧标识符 |
| 2 | FF | 帧格式<br>0：标准格式帧<br>1：扩展格式帧 |
| 1 | FT | 帧种类<br>0：数据帧 |

| | | |
|---|---|---|
| | | 1：遥控帧 |
| 0 | TEN | 发送使能<br>当应用程序想要发送数据时，该位被置 1 将启动发送过程。当发送结束，发送邮箱为空时，该位由硬件清 0。<br>0：禁用发送<br>1：使能发送 |

## 26.4.10. 发送邮箱属性寄存器(CAN_TMPx)(x = 0...2)

地址偏移：0x184，0x194，0x1A4
复位值：0xXXXX XXXX

该寄存器只能按字(32位)访问

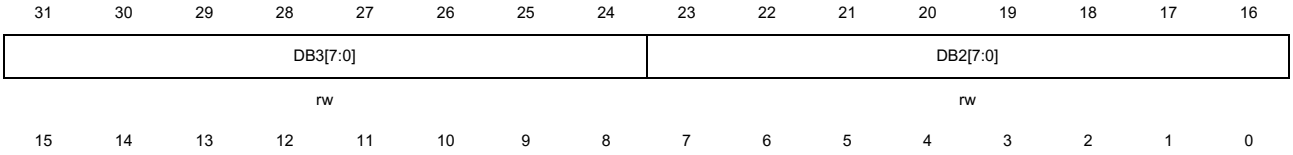



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | TS[15:0] | 时间戳<br>发送时间戳 |
| 15:9 | 保留 | 必须保持复位值。 |
| 8 | TSEN | 时间戳使能<br>0：禁用时间戳<br>1：使能时间戳。时间戳 TS[15:0]将放在寄存器 CAN_TMDATA1 的 DATA6 和 DATA7 中<br>只有当寄存器 CAN_CTL 中的 TTC 为 1 时，该位才有效。 |
| 7:4 | 保留 | 必须保持复位值。 |
| 3:0 | DLENC[3:0] | 数据长度，DLENC[3:0]表示帧内数据长度。 |

## 26.4.11. 发送邮箱 data0 寄存器(CAN_TMDATA0x)(x = 0...2)

地址偏移：0x188，0x198，0x1A8
复位值：0xXXXX XXXX

该寄存器只能按字(32位)访问



| DB1[7:0] | DB0[7:0] |
|---|---|
| rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | DB3[7:0] | 字节 3 |
| 23:16 | DB2[7:0] | 字节 2 |
| 15:8 | DB1[7:0] | 字节 1 |
| 7:0 | DB0[7:0] | 字节 0 |

## 26.4.12. 发送邮箱 data1 寄存器(CAN_TMDATA1x)(x = 0...2)

地址偏移：0x18C，0x19C，0x1AC
复位值：0xXXXX XXXX

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DB7[7:0] | | | | | | | | DB6[7:0] | | | | | | | |
| rw | | | | | | | | rw | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| DB5[7:0] | | | | | | | | DB4[7:0] | | | | | | | |
| rw | | | | | | | | rw | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | DB7[7:0] | 字节 7 |
| 23:16 | DB6[7:0] | 字节 6 |
| 15:8 | DB5[7:0] | 字节 5 |
| 7:0 | DB4[7:0] | 字节 4 |

## 26.4.13. 接收 FIFO 邮箱标识符寄存器(CAN_RFIFOMIx)(x = 0,1)

地址偏移：0x1B0，0x1C0
复位值：0xXXXX XXXX

该寄存器只能按字(32位)访问

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

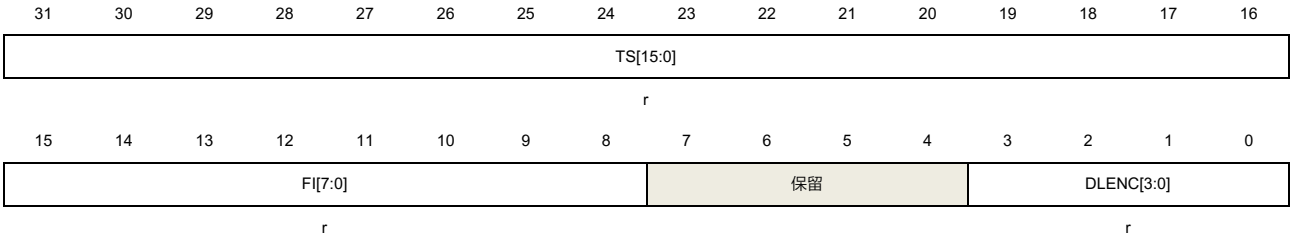
| 位 | 区域 | 说明 |
| --- | --- | --- |
| 31:21 | SFID[10:0]/EFID[28:18] | 标识符<br>SFID[10:0]：标准格式帧标识符<br>EFID[28:18]：扩展格式帧标识符 |
| 20:16 | EFID[17:13] | 标识符<br>EFID[17:13]：扩展格式帧标识符 |
| 15:3 | EFID[12:0] | 标识符<br>EFID[12:0]：扩展格式帧标识符 |
| 2 | FF | 帧格式<br>0：标准格式帧<br>1：扩展格式帧 |
| 1 | FT | 帧种类<br>0：数据帧<br>1：遥控帧 |
| 0 | 保留 | 必须保持复位值。 |

## 26.4.14. 接收 FIFO 邮箱属性寄存器(CAN_RFIFOMPx)(x = 0,1)

地址偏移：0x1B4，0x1C4

复位值：0xXXXX XXXX

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:16 | TS[15:0] | 时间戳<br>接收时间戳 |
| 15:8 | FI[7:0] | 过滤索引<br>帧通过过滤器时的过滤序号 |
| 7:4 | 保留 | 必须保持复位值。 |
| 3:0 | DLENC[3:0] | 数据长度<br>DLENC[3:0]表示帧内数据长度。 |

### 26.4.15. 接收 FIFO 邮箱 data0 寄存器(CAN_RFIFOMDATA0x)(x=0,1)

地址偏移：0x1B8，0x1C8
复位值：0xXXXX XXXX

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | DB3[7:0] | 字节 3 |
| 23:16 | DB2[7:0] | 字节 2 |
| 15:8 | DB1[7:0] | 字节 1 |
| 7:0 | DB0[7:0] | 字节 0 |

### 26.4.16. 接收 FIFO 邮箱 data1 寄存器(CAN_RFIFOMDATA1x)(x=0,1)

地址偏移：0x1BC，0x1CC
复位值：0xXXXX XXXX

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | DB7[7:0] | 字节 7 |
| 23:16 | DB6[7:0] | 字节 6 |
| 15:8 | DB5[7:0] | 字节 5 |
| 7:0 | DB4[7:0] | 字节 4 |

### 26.4.17. 过滤器控制寄存器(CAN_FCTL)(仅 CAN0 可用)

地址偏移：0x200

复位值：0x2A1C 0E01

该寄存器只能按字(32位)访问

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | HBC1F[5:0] | | | | | | 保留 | | | | | | | FLD |
| | | rw | | | | | | | | | | | | | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值。 |
| 13:8 | HBC1F[5:0] | CAN1 过滤器单元起始位置<br>这些位用来定义 CAN1 过滤器起始位置。CAN0 可以用编号为 0~HBC1F-1 过滤器，CAN1 可以用编号为 HBC1F~27 过滤器。当这些位的值为 0，CAN0 将没有过滤器可以使用。当这些位的值为 28 时，CAN1 将没有过滤器可以使用。 |
| 7:1 | 保留 | 必须保持复位值。 |
| 0 | FLD | 过滤器锁禁用<br>0：使能过滤器锁<br>1：禁用过滤器锁 |

## 26.4.18. 过滤器模式配置寄存器(CAN_FMCFG)(仅 CAN0 可用)

地址偏移：0x204

复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | FMOD27 | FMOD26 | FMOD25 | FMOD24 | FMOD23 | FMOD22 | FMOD21 | FMOD20 | FMOD19 | FMOD18 | FMOD17 | FMOD16 |
| | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FMOD15 | FMOD14 | FMOD13 | FMOD12 | FMOD11 | FMOD10 | FMOD9 | FMOD8 | FMOD7 | FMOD6 | FMOD5 | FMOD4 | FMOD3 | FMOD2 | FMOD1 | FMOD0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:0 | FMODx | 过滤器模式<br>0：掩码模式<br>1：列表模式 |

### 26.4.19. 过滤器位宽配置寄存器(CAN_FSCFG)(仅 CAN0 可用)

地址偏移：0x20C
复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | FS27 | FS26 | FS25 | FS24 | FS23 | FS22 | FS21 | FS20 | FS19 | FS18 | FS17 | FS16 |
| | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| FS15 | FS14 | FS13 | FS12 | FS11 | FS10 | FS9 | FS8 | FS7 | FS6 | FS5 | FS4 | FS3 | FS2 | FS1 | FS0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:0 | FSx | 过滤器位宽<br>0：16-bit 位宽<br>1：32-bit 位宽 |

### 26.4.20. 过滤器关联 FIFO 寄存器(CAN_FAFIFO)(仅 CAN0 可用)

地址偏移：0x214
复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | FAF27 | FAF26 | FAF25 | FAF24 | FAF23 | FAF22 | FAF21 | FAF20 | FAF19 | FAF18 | FAF17 | FAF16 |
| | | | | rw | rw | rw | rw | rw | rw | rw | rw | Rw | rw | rw | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| FAF15 | FAF14 | FAF13 | FAF12 | FAF11 | FAF10 | FAF9 | FAF8 | FAF7 | FAF6 | FAF5 | FAF4 | FAF3 | FAF2 | FAF1 | FAF0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | Rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:0 | FAFx | 过滤器关联 FIFO<br>0：关联 FIFO0<br>1：关联 FIFO1 |

### 26.4.21. 过滤器激活寄存器(CAN_FW)(仅 CAN0 可用)

地址偏移：0x21C
复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | FW27 | FW26 | FW25 | FW24 | FW23 | FW22 | FW21 | FW20 | FW19 | FW18 | FW17 | FW16 |
| | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| FW15 | FW14 | FW13 | FW12 | FW11 | FW10 | FW9 | FW8 | FW7 | FW6 | FW5 | FW4 | FW3 | FW2 | FW1 | FW0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:0 | FWx | 过滤器激活<br>0：没有激活<br>1：激活工作 |

## 26.4.22. 过滤器(x)数据(y)寄存器(CAN_FxDATAy)(x = 0...27, y = 0,1)(仅 CAN0 可用)

地址偏移：0x240 + 8 * x + 4 * y, (x = 0..27, y = 0,1)
复位值：0xXXXX XXXX

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FD31 | FD30 | FD29 | FD28 | FD27 | FD26 | FD25 | FD24 | FD23 | FD22 | FD21 | FD20 | FD19 | FD18 | FD17 | FD16 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| FD15 | FD14 | FD13 | FD12 | FD11 | FD10 | FD9 | FD8 | FD7 | FD6 | FD5 | FD4 | FD3 | FD2 | FD1 | FD0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | FDx | 过滤器数据<br>掩码模式下：<br>0：标识符的 Bit(x)不需参与比较<br>1：标识符的 Bit(x)需要参与比较<br>列表模式下：<br>0：标识符的 Bit(x)必须为 0<br>1：标识符的 Bit(x)必须为 1 |

# 27. 以太网(ENET)

## 27.1. 简介

本章描述以太网外设模块。该以太网模块包含10/100Mbps以太网MAC(媒体访问控制器)，采用DMA优化数据帧的发送与接收性能，支持MII(媒体独立接口)与RMII(简化的媒体独立接口)两种与物理层(PHY)通讯的标准接口，实现以太网数据帧的发送与接收。以太网模块遵守IEEE 802.3-2002标准和IEEE 1588-2008标准。

## 27.2. 主要特性

### MAC特性

- 支持10Mbps或100Mbps数据传输速率；
- 支持MII和RMII接口；
- 支持调试用回环模式；
- 支持符合CSMA/CD协议的半双工背压通讯；
- 支持符合IEEE 802.3x的流控通讯。在当前帧发送完毕后，根据接收的暂停帧中暂停时间延迟发送。在全双工/半双工模式下，MAC根据RxFIFO的填充程度自动发送暂停帧/背压信号；
- 支持符合IEEE 802.3x的全双工流控通讯，当输入流控信号失效时，自动发送零时间片暂停帧。支持符合IEEE 802.3x的半双工流控通讯，支持根据RxFIFO的填充程度(直通模式)自动发送背压信号；
- 可选择在发送操作时自动生成校验/填充位；
- 可选择在接收操作时自动去除校验/填充位；
- 帧长度可配置；
- 帧间隙可配置；
- 支持多种模式的接收过滤；
- 支持检测接收帧的IEEE 802.1Q VLAN标签；
- 支持强制网络统计标准(RFC2819/RFC2665)；
- 支持两种唤醒帧检测：LAN远程唤醒帧和AMD的Magic Packet™帧；
- 支持校验和检查(IPv4报头，IPv4或IPv6数据格式封装的TCP、UDP或ICMP)；
- 支持IEEE 1588-2008标准定义的以太网帧时间戳，并将其按64位记录于帧状态中；
- 相互独立的两个FIFO分别用于发送与接收；
- 在延迟冲突、过度冲突、过度顺延和下溢情况下丢弃帧；
- 帧传输时，支持存储转发模式下的硬件校验和的计算和插入。

### DMA特性

- 支持环结构或链结构两种形式的描述符列表；

- 每个描述符可以传输最高为8192字节的数据；
- 中断可配置，适用于多种工作状态；
- 支持轮询或固定优先级两种方式仲裁DMA发送和接收控制器的请求。

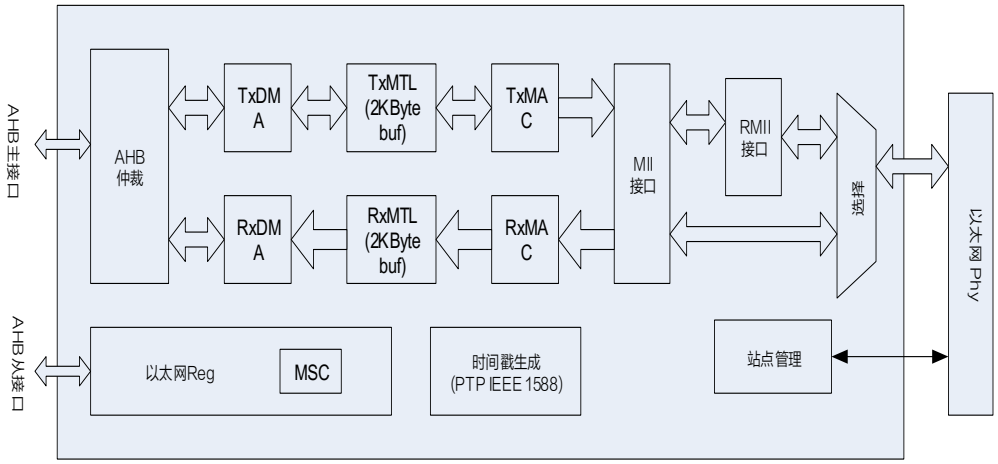
PTP特性

- 支持符合IEEE1588的时间同步功能；
- 支持粗/精调两种校正方法；
- 输出秒脉冲；
- 达到预设目标时间时触发中断。

### 27.2.1. 模块框图

以太网模块由MAC(介质访问控制器)模块、MII/RMII模块和一个以描述符形式控制的DMA模块组成。在使用以太网模块时，用户应确保配置的时钟频率不小于25MHz。

**图 27-1. 以太网模块框图**




MAC模块通过MII或RMII与片外PHY连接。通过对SYSCFG_CFG1寄存器的ENET_PHY_SEL位进行设置，可以选择使用哪种接口。SMI(站点管理接口)用于配置和管理外部PHY。

发送数据模块包括：

- TxDMA控制器，用于从存储器中读取描述符和数据，以及将状态写入存储器；
- TxMTL，用于对发送数据的控制，管理和存储。TxMTL内含TxFIFO，用于缓存待MAC发送的数据；
- MAC发送控制寄存器组，用于管理和控制数据帧的发送。

接收数据模块包括：

- RxDMA控制器，用于从存储器中读取描述符，以及将数据与状态写入存储器；

- RxMTL，用于对接收数据的控制，管理和存储。RxMTL实现了RxFIFO，用于存储待转发到系统存储的帧数据；
- MAC接收控制寄存器组，用于管理数据帧的接收和标示接收状态。MAC内含接收过滤器，采用多种过滤机制，滤除特定的以太网帧。

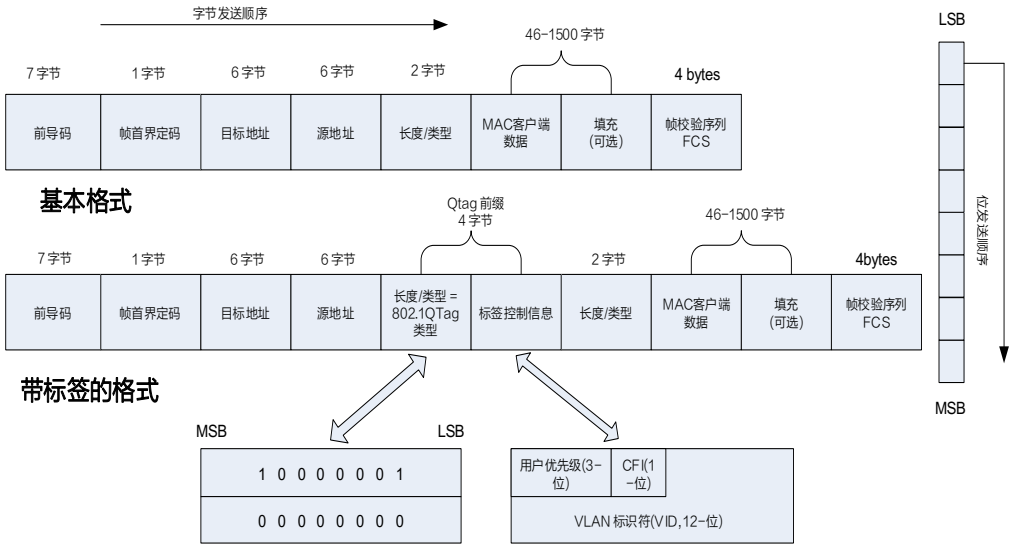
### 27.2.2. MAC 802.3 以太网数据包描述

MAC的数据通信可使用两种帧格式：

- 基本帧格式；
- 带标签的帧格式。

*图27-2. MAC/带标签的MAC帧格式*描述了帧结构(基本的和带标签的)：

**图 27-2. MAC/带标签的 MAC 帧格式**




**注意：**除了帧校验序列，以太网控制器发送每个字节时都按照低位先出的次序进行传输。

CRC计算包括帧数据的所有字节除去前导码和帧首界定码域。以太网帧的32位CRC生成多项式为0x04C11DB7，且此多项式用于以太网模块中所有的32位CRC计算，如下式所示：

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

### 27.2.3. 以太网信号描述

*表27-1. 以太网信号(MII模式)*和*表27-2.以太网信号(RMII模式)*列出了MAC模块所用引脚在MII/RMII模式下默认及重映射的功能和具体配置。

**表 27-1. 以太网信号(MII 模式)**

| 信号 | 引脚1 | 引脚2 | 引脚模式(AF11) |
|---|---|---|---|
| ENET_MDC | PC1 | - | 推挽复用输出 |

| 信号 | 引脚1 | 引脚2 | 引脚模式(AF11) |
|---|---|---|---|
| ENET_MII_TXD2 | PC2 | - | 推挽复用输出 |
| ENET_MII_TX_CLK | PC3 | - | 浮空输入(复位状态) |
| ENET_MII_CRS | PA0 | PH2 | 浮空输入(复位状态) |
| ENET_MII_RX_CLK | PA1 | - | 浮空输入(复位状态) |
| ENET_MDIO | PA2 | - | 推挽复用输出 |
| ENET_MII_COL | PA3 | PH3 | 浮空输入(复位状态) |
| ENET_MII_RX_DV | PA7 | - | 浮空输入(复位状态) |
| ENET_MII_RXD0 | PC4 | - | 浮空输入(复位状态) |
| ENET_MII_RXD1 | PC5 | - | 浮空输入(复位状态) |
| ENET_MII_RXD2 | PB0 | PH6 | 浮空输入(复位状态) |
| ENET_MII_RXD3 | PB1 | PH7 | 浮空输入(复位状态) |
| ENET_PPS_OUT | PB5 | PG8 | 推挽复用输出 |
| ENET_MII_TXD3 | PB8 | PE2 | 推挽复用输出 |
| ENET_MII_RX_ER | PB10 | PI10 | 浮空输入(复位状态) |
| ENET_MII_TX_EN | PB11 | PG11 | 推挽复用输出 |
| ENET_MII_TXD0 | PB12 | PG13 | 推挽复用输出 |
| ENET_MII_TXD1 | PB13 | PG14 | 推挽复用输出 |

**表 27-2.以太网信号(RMII 模式)**

| 信号 | 引脚1 | 引脚2 | 引脚模式(AF11) |
|---|---|---|---|
| ENET_MDC | PC1 | - | 推挽复用输出 |
| ENET_REF_CLK | PA1 | - | 浮空输入(复位状态) |
| ENET_MDIO | PA2 | - | 推挽复用输出 |
| ENET_CRS_DV | PA7 | - | 浮空输入(复位状态) |
| ENET_RMII_RXD0 | PC4 | - | 浮空输入(复位状态) |
| ENET_RMII_RXD1 | PC5 | - | 浮空输入(复位状态) |
| ENET_PPS_OUT | PB5 | PG8 | 推挽复用输出 |
| ENET_RMII_TX_EN | PB11 | PG11 | 推挽复用输出 |
| ENET_RMII_TXD0 | PB12 | PG13 | 推挽复用输出 |
| ENET_RMII_TXD1 | PB13 | PG14 | 推挽复用输出 |

**注意:** 对任意接口模式(MII/RMII)，应用程序都需确保引脚1或引脚2中只有一个被映射到AF11。

## 27.3. 功能描述

### 27.3.1. 接口配置

以太网模块通过MII/RMII接口与片外PHY连接，传送与接收以太网包。MII或RMII模式由软件选择并通过SMI接口对PHY进行管理。

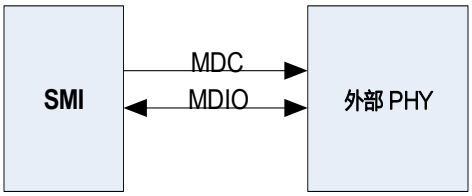
### SMI：站点管理接口

SMI用于访问和设置PHY的配置。

站点管理接口(SMI)通过MDC时钟线与MDIO数据线与外部PHY通讯，可以通过其访问任意PHY的任意寄存器。SMI接口支持的最大PHY数量为32，但在同一时刻只能访问一个PHY的一个寄存器。MDC时钟线和MDIO数据线具体作用如下：

- MDC：最高频率为2.5MHz的时钟信号，在空闲状态下该引脚保持为低电平状态。在传输数据时该信号的高电平和低电平的最短保持时间为160ns，信号的最小周期为400ns；
- MDIO：用于与PHY之间的数据传输，与MDC时钟线配合，接收/发送数据。

**图 27-3. 站点管理接口信号**




### 写操作

应用程序需将要传输的数据写入ENET_MAC_PHY_DATA寄存器中，并对ENET_MAC_PHY_CTL寄存器相关位进行操作：

1. 设置PHY设备地址和将要操作的PHY寄存器地址，并将PW位置为1，使能写模式；
2. 将PB位置1开始传输。在传输过程中PB位一直为高，直到传输完成，硬件将会自动清除PB位。

应用程序可以通过PB位判断传输是否完成。在PB位置1期间，由于操作正在运行，因此不能修改PHY控制寄存器和PHY数据寄存器的内容。在将PB位置位之前，应用程序必须确保该位读出为'0'。

### 读操作

应用程序对ENET_MAC_PHY_CTL寄存器相关位进行操作：

1. 设置PHY设备地址和将要操作的PHY寄存器地址，并将PW位置为0，使能读模式；
2. 将PB位置1开始数据接收。在接收过程中PB位一直为高，直到接收完成，硬件将会自动清除PB位。

应用程序可以通过PB位判断传输是否完成。在PB位置1期间，由于操作正在运行，因此不能修改PHY控制寄存器和PHY数据寄存器的内容。在将PB位置位之前，应用程序必须确保该位读出为'0'。

**注意：**由于PHY寄存器地址16-31的寄存器内容由各厂商自由定义，所以在访问不同PHY设备

的这部分寄存器时，需要根据厂商手册对应用程序进行不同的设置。GD32A490xx固件库当前支持的PHY设备详情请参考固件库相关手册说明。

### 时钟配置

SMI接口的时钟源由AHB时钟分频得到。为了保证MDC时钟频率不超过2.5MHz，需根据AHB时钟频率对PHY控制寄存器中相关位进行设置，选择合适的分频系数。*表27-2. 时钟范围*列出了对应AHB时钟范围的分频系数的选择。

**表 27-3. 时钟范围**

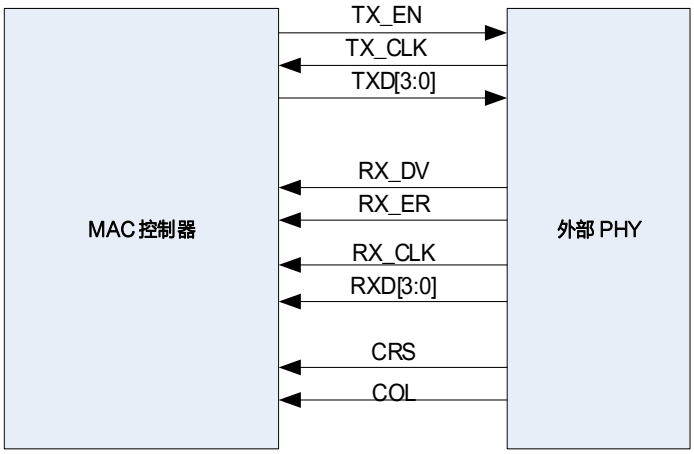
| AHB时钟 | MDC 时钟 | ENET_MAC_PHY_CTL 位 CLR[2:0] |
|---|---|---|
| 150~240MHz | AHB clock/102 | 0x4 |
| 35~60MHz | AHB clock/26 | 0x3 |
| 20~35MHz | AHB clock/16 | 0x2 |
| 100~150 MHz | AHB clock/62 | 0x1 |
| 60~100MHz | AHB clock/42 | 0x0 |

### MII/RMII 的选择

当使能以太网控制器时钟前或以太网控制器处于复位状态时，应用程序可通过配置SYSCFG_CFG1寄存器的第23位ENET_PHY_SEL位来选择使用MII或者RMII模式，默认为MII模式。

### MII：媒体独立接口

**图 27-4. 媒体独立接口(MII)信号线**




- **MII_TX_CLK**：发送数据使用的时钟信号，对于10Mbit/s的数据传输，此时钟为2.5MHz，对于100M bit/s的数据传输，此时钟为25MHz。
- **MII_RX_CLK**：接收数据使用的时钟信号，对于10Mbit/s的数据传输，此时钟为2.5MHz，对于100M bit/s的数据传输，此时钟为25MHz。

- **MII_TX_EN**：发送使能信号，，当数据前导码的起始位出现时，该信号必须有效，并且需要在传输完毕前保持有效。
- **MII_TXD[3:0]**：发送数据线，每次传输4位数据，数据在MII_TX_EN信号有效时有效。当MII_TX_EN信号无效时，PHY将忽略传输的数据。
- **MII_CRS**：载波侦听信号，仅工作在半双工模式下，由PHY控制。该信号不需要与MII_TX_CLK和MII_RX_CLK保持同步。当它处于有效状态时，意味着发送或接收介质不处于空闲状态。MII_CRS信号一直保持有效，直到发送和接收介质都处于空闲状态。
- **MII_COL**：冲突检测信号，仅工作在半双工模式下，由PHY控制。该信号不需要与MII_TX_CLK和MII_RX_CLK保持同步。当检测到介质发生冲突时，此信号有效，并且在整个冲突的持续时间内，保持此信号有效。
- **MII_RXD[3:0]**：接收数据线，每次接收4位数据，数据在MII_RX_DV信号有效时有效。根据MII_RX_DV和MII_RX_ER信号的状态，MII_RXD[3:0]数据值可被用来传达一些特定信息(请参考*表27-4. 接收接口信号编码*)。
- **MII_RX_DV**：接收数据使能信号，由PHY控制，当PHY准备好数据供MAC接收时，该信号有效。当数据帧的第一个4位出现时，该信号必须有效，并且需要在传输完毕前保持有效。在传送最后4位数据后的第一个时钟之前，此信号必须变为无效状态。为确保正确地接收帧，MII_RX_DV信号应该在SFD字段出现之前有效。
- **MII_RX_ER**：接收出错信号，为了表明MAC在接收过程中检测到错误，MII_RX_ER信号必须在一个或多个时钟周期(MII_RX_CLK)内保持有效。具体错误原因需结合MII_RX_DV的状态及MII_RXD[3:0]的数据值，详见*表27-4. 接收接口信号编码*。

**表 27-4. 接收接口信号编码**

| 信号 | 正常的帧间隔 | | 正常的数据接收 | 载波错误指示 | 数据接收出错 |
|---|---|---|---|---|---|
| MII_RX_ER | 0 | 1 | 0 | 1 | 1 |
| MII_RX_DV | 0 | 0 | 1 | 0 | 1 |
| MII_RXD[3:0] | 0000 to 1111 | 0000 | 0000 to 1111 | 1110 | 0000 to 1111 |

#### MII 时钟源

用户需要给外部PHY提供一个外部的25MHz时钟来产生TX_CLK和RX_CLK时钟信号。该时钟不需要与MAC时钟相同。可以使用外部的25MHz晶振或者微控制器的时钟输出引脚CK_OUTx(x=0,1)提供这一时钟。当时钟来源为CK_OUTx(x=0,1)引脚时需配置合适的PLL，保证CK_OUTx(x=0,1)引脚输出的时钟为25MHz。

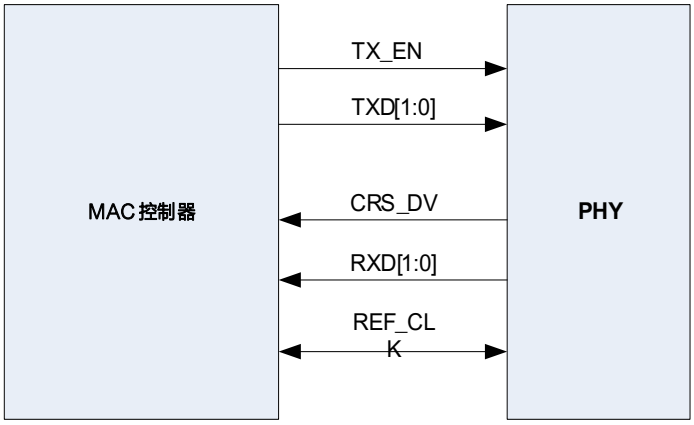
### RMII：精简媒体独立接口

精简媒体独立接口(RMII)规范减少了以太网通信所需要的引脚数。根据IEEE 802.3标准，MII接口需要16个引脚用于数据和控制信号，而RMII标准则将引脚数减少到了7个。

RMII特性：

- 只有一个时钟信号，且该时钟信号需要提高到50MHz；
- MAC和外部的以太网PHY需要使用同样的时钟源；
- 使用2位宽度的数据收发。

图 27-5. 精简媒体独立接口(RMII)信号线




**RMII 时钟源**

通过将相同的时钟源接到MAC和以太网PHY的REF_CLK引脚保证两者时钟源的同步。可以通过外部的50MHz信号或者微控制器的CK_OUTx(x=0,1)引脚提供这一时钟。当时钟来源CK_OUTx(x=0,1)引脚时需配置合适的PLL，保证CK_OUTx(x=0,1)引脚输出的时钟为50MHz。

**MII/RMII 位传输顺序**

不论选择的是MII还是RMII接口，发送接收的次序都是低位先出。

MII和RMII之间的区别主要是数据位数和发送次数的不同。MII上是先发送/接收低4位数据，再发送/接收高4位。在RMII上则是先发送/接收最低2位数据，再次低2位数据，次高2位数据，和最高2位数据。

例如：一个字节数据为10011101b(从左到右顺序：高位到低位)

使用MII发送需2个时钟周期：1101 -> 1001(从左到右顺序：高位到低位，其中1101对应MII_T/RXD[3] - MII_T/RXD[0])

使用RMII发送需4个时钟周期：01 -> 11 -> 01 -> 10(从左到右顺序：高位到低位，其中01对应RMII_T/RXD[1] - RMII_T/RXD[0])

### 27.3.2. MAC 功能简介

MAC 模块可以在两种模式(半双工模式和全双工模式)下工作。半双工模式下，通过 CSMA/CD 算法来抢占对物理介质的访问，在同一时间只有一个传输方向的两个站点有效。全双工模式下，满足以下条件时，可同时进行收发而不发生冲突：1)物理介质支持同时进行收发操作。2)只有两个站点接入 LAN，且两个站点都配置为全双工模式。

MAC 模块能够实现以下功能：1)数据封装(发送和接收)，包括检测/解码帧、帧边界定界、

寻址(管理源地址和目的地址)、错误检测。2)半双工模式下的介质访问管理，包括介质分配和冲突解决。

### MAC 的发送流程

所有的发送均由以太网模块中专用DMA控制器和MAC控制。在收到应用程序发送指令后，DMA将发送帧从系统存储区读出并存入深度为2K的TxFIFO中，之后根据选择的模式(直通或者存储转发模式，具体定义请查看下段)将数据取出到MAC控制器，通过MII/RMII接口发送到以太网PHY，并可以选择配置使MAC控制器自动将硬件计算的CRC值添加到数据帧的帧校验序列中。当MAC控制器收到来自TxFIFO的帧结束信号后，完成整个传输过程。传输完毕后，传输状态信息将会由MAC控制器生成并写回到DMA控制器中，应用程序可以通过DMA当前发送描述符查询发送状态。

TxFIFO取出数据到MAC控制器的操作有两种模式：

- 在直通模式下，一旦FIFO中的数据字节数等于或超过设置的阈值，或者描述符中的帧结束标志被写入，FIFO中的数据就会被送入到MAC控制器中。用户可通过ENET_DMA_CTL中的TTHC[2:0]配置阈值。
- 在存储转发模式下，只有当一个完整的帧写入FIFO之后，FIFO中的数据才会被送入到MAC控制器中。但还有一种情况，帧没有被完整写入FIFO，FIFO也会取出数据。这种情况为TxFIFO的大小小于要发送的以太网帧长度，那么在TxFIFO即将全满时，数据会被送入到MAC控制器。

#### 特殊情况处理

在传输过程中，如果空闲的DMA发送描述符不足，或者误操作了ENET_DMA_CTL的FTF位清空了FIFO(此位置1时将清空TxFIFO中的数据并将FIFO的指针复位，清空操作完成后由硬件将此位清零)，则将导致不能及时连续的发送数据，此时MAC控制器会标识数据下溢状态。对于只收到帧起始信号却没有收到帧结束信号的情况，MAC会忽略第二帧数据的帧起始，而将第二帧作为前一帧的延续。

若被发送的一帧占用两个DMA发送描述符，则第一个描述符的首段位(FSG)和末段位(LSG)应为10b，第二个描述符的应为01b。若第一个描述符与第二个描述符的FSG位都置位了，且第一个描述符的LSG位复位了，则将忽略第二个描述符的FSG位，并认为这两个描述符为只发送一个帧。

若发送MAC帧的数据域长度小于46或者带标签的MAC帧的数据域长度小于42，可以选择配置MAC控制器自动填充内容为0的数据，使帧数据域的长度符合IEEE 802.3规范的相关定义。若执行了自动填充0功能，则MAC将忽略DMA描述符DCRC位的配置，自动计算并添加CRC值到帧的帧校验序列中。

## MAC 的发送管理

### Jabber 定时器

为了防止出现一个站点长时间占用PHY的情况，以太网内置的Jabber定时器会在以太网帧发送超过2048字节后终止发送。默认情况下，Jabber定时器是使能的，因此当以太网帧发送超过2048字节，则MAC将只发送2048字节，并丢弃剩余的帧数据。

### 冲突处理机制：重发

在半双工模式下，MAC发送数据帧时可能会发生冲突。当发生冲突事件的时候如果FIFO中只有不超过96个字节的帧数据被取出到了MAC中，那么帧重发功能将被激活。重发功能激活后，MAC会中止当前的传输，然后重新从FIFO中读取数据并发送。当发生冲突事件的时候如果已有超过96个字节的数据从FIFO中取出到MAC中，那么MAC会中止当前的传输但不会激活重发功能，然后在描述符中置位LCO以通知应用程序。

### 清空 TxFIFO 操作

将ENET_DMA_CTL寄存器的FTF位置1将清空TxFIFO，并将FIFO数据指针复位。无论TxFIFO是否正在取出数据到MAC中，清空操作都会立刻执行。因此这也将导致MAC控制器产生数据下溢事件，并终止发送当前帧，同时返回该帧的状态信息和发送状态信息字到应用程序。并标记数据下溢位和清空位(发送描述符0的FRMF和UFE位)。在应用程序(DMA)接收到所有被清空帧的状态信息字以后，清空操作完成。清空操作完成后，ENET_DMA_CTL寄存器的FTF位将自动清'0'。当收到清空操作指令，所有从FIFO取出到MAC的数据都将被丢弃，直到收到FSG位为1的描述符。

### 帧间隔管理

MAC管理两个帧之间的时间间隔。两个帧之间的时间间隔称为帧间隙时间。在全双工模式下，在完成帧发送后，或者MAC进入空闲状态时，帧间隙计数器开始计数。如果在帧间隙时间未到达ENET_MAC_CFG寄存器中IGBS位所配置的值时，来了新的发送帧，则这个发送帧将被延迟发送直到达到帧间隙时间值。若这个新的发送帧在帧间隙时间之后到达，则会立即发送该帧。在半双工模式下，MAC遵循截断二进制指数退让算法，简要来说，就是在前一个发送帧发送完成之后，或者MAC进入空闲状态时，帧间隙计数器开始计数。在帧间隙时间内，可能会有3种情况会发生：

1. 如果在帧间隙时间的前2/3时间检测到载波信号，帧间隙计数器将复位并重新计数；
2. 如果在帧间隙时间的后1/3时间里检测到载波信号，帧间隙计数器不会复位，将继续计数，当帧间隙时间到达后，MAC发送新的帧；
3. 如果在整个帧间隙时间内都没有检测到载波信号，则在到达帧间隙时间后停止帧时隙计数器，并在之前有帧被延迟的情况下立即发送新的帧。

## 地址过滤模块

MAC过滤分为错误过滤(诸如过短帧、CRC错误以及坏帧的过滤)和地址过滤。此部分主要讨论地址过滤。

地址过滤利用静态物理地址(MAC地址)过滤和多播HASH列表过滤实现。若ENET_MAC_FRMF过滤器寄存器的FAR位为'0'(默认值),则只有通过地址过滤的帧才会被接收。该功能会根据应用程序设定的参数(帧过滤器寄存器)对单播帧或多播帧的目的与/或源地址进行过滤(通过目标地址的I/G位可判断是单播帧还是多播帧)并报告相应的地址过滤结果,所有不能通过过滤器的帧将被丢弃。

**注意：**若ENET_MAC_FRMF过滤器寄存器的FAR位为'1'，则所有帧都会被接收。在这种情况下，帧过滤结果仍会更新到接收描述符中，但帧过滤结果不会影响到帧是否会被过滤。

**单播目标地址过滤器**

通过对ENET_MAC_FRMF寄存器HUF位的设置，可以选择使用静态物理地址(HUF位为'0')或者HASH列表(HUF位为'1')的方式实现单播过滤。

■ 静态物理地址过滤

MAC控制器支持多达4个MAC地址对单播地址进行完美过滤。在这种方式下，MAC会把接收到帧的6个字节单播地址与设好的MAC地址寄存器逐位比较，检查是否相符。对于MAC地址0寄存器始终使能，对于MAC地址1-MAC地址3寄存器分别有对应的使能位。MAC地址1-MAC地址3寄存器的每一个字节都可以通过相应MAC地址的高寄存器的屏蔽字节控制位(MB位)来设置是否与接收帧的目标地址相应字节比较。

■ HASH列表过滤

这种过滤使用一种HASH机制。MAC利用64位的HASH列表对单播地址进行不完美过滤。这种过滤模式遵循以下两个过滤步骤：

1. MAC计算接收帧的目标地址的CRC值
2. 取CRC计算结果高6位作为索引检索HASH列表。如果CRC值对应的HASH列表上的相应位为'1'，则该帧能通过HASH过滤器，反之则该帧不能通过HASH过滤器。

这种类型过滤器的优点是可以仅用一个小表就覆盖任何可能的地址。缺点是过滤不完全，即有时应该丢弃的帧也会被接收。

**单播源地址过滤器**

使能MAC地址1-MAC地址3寄存器，并设置其对应MAC地址高寄存器的SAF位为'1'，MAC可以将MAC地址1-MAC地址3寄存器中设置的物理(MAC)地址与接收帧的源地址进行比较并过滤。MAC也支持对源地址的成组过滤。若设置帧过滤寄存器ENET_MAC_FRMF的SAFLT位为'1'，MAC会丢弃没能通过源地址过滤的帧，同时过滤结果会通过DMA接收描述符0的SAFF位反映出来。当SAFLT位为'1'的同时，目标地址过滤器也在工作，此时MAC控制器以两个滤波器结果的逻辑"与"形式判定帧是否通过。这意味着，只要帧没能通过其中一个过滤器，就会被丢弃。MAC只会把通过全部过滤器的帧转发给应用程序。

**多播目标地址过滤器**

将帧过滤寄存器ENET_MAC_FRMF的MFD位清零，以开启MAC多播地址过滤功能。此时根据帧过滤寄存器ENET_MAC_FRMF的HMF位的取值可以选择类似于单播目标地址过滤的两种方式进行地址过滤。

**广播地址过滤器**

默认情况下，MAC无条件的接收任何广播帧。但当设置帧过滤寄存器ENET_MAC_FRMF的BFRMD位为'1'时，MAC将丢弃接收到的所有广播帧。

**HASH 或者完美地址过滤器**

设置帧过滤器寄存器ENET_MAC_FRMF的HPFLT位为'1'，并设置相应的HUF位(对单播帧)或者HMF位(对多播帧)为'1'，则可以将过滤器配置成只要接收帧的目标地址匹配HASH过滤器或者物理地址过滤器之一，就令帧通过。

**逆转过滤操作**

无论是目标地址过滤还是源地址过滤，都能在过滤器输出端逆转过滤结果。即地址与过滤器匹配时，帧不通过；不匹配时帧通过。通过设置帧过滤寄存器ENET_MAC_FRMF的DAIFLT位和SAIFLT位为'1'可以实现这一功能。DAIFLT位作用于单播和多播帧的目标地址的过滤结果，SAIFLT位作用于单播和多播帧的源地址的过滤结果。

下面*表27-5. 目标地址过滤器结果列表*和*表27-6. 源地址过滤器结果列表*总结了目标地址和源地址过滤器在不同设置下的工作状态。

**表 27-5. 目标地址过滤器结果列表**

| 帧类型 | PM | HPFLT | HUF | DAIFLT | HMF | MFD | BFRMD | 目标地址过滤器操作 |
|---|---|---|---|---|---|---|---|---|
| 广播帧 | 1 | - | - | - | - | - | - | 通过 |
| | 0 | - | - | - | - | - | 0 | 通过 |
| | 0 | - | - | - | - | - | 1 | 不通过 |
| 单播帧 | 1 | - | - | - | - | - | - | 所有帧通过 |
| | 0 | - | 0 | 0 | - | - | - | 匹配完美/组过滤器时通过 |
| | 0 | - | 0 | 1 | - | - | - | 匹配完美/组过滤器时不通过 |
| | 0 | 0 | 1 | 0 | - | - | - | 匹配HASH过滤器时通过 |
| | 0 | 0 | 1 | 1 | - | - | - | 匹配HASH过滤器时不通过 |
| | 0 | 1 | 1 | 0 | - | - | - | 匹配HASH或者完美/组过滤器时通过 |
| | 0 | 1 | 1 | 1 | - | - | - | 匹配HASH或者完美/组过滤器时不通过 |
| 多播帧 | 1 | - | - | - | - | - | - | 所有帧通过 |
| | - | - | - | - | - | 1 | - | 所有帧通过 |
| | 0 | - | - | 0 | 0 | 0 | - | 匹配完美/组过滤器时通过，如果PCFRM = 0x，丢弃暂停控制帧 |
| | 0 | 0 | - | 0 | 1 | 0 | - | 匹配HASH过滤器时通过，如果PCFRM = 0x，丢弃暂停控制帧 |
| | 0 | 1 | - | 0 | 1 | 0 | - | 匹配HASH或者完美/组过滤器时通过，如果PCFRM = 0x，丢弃暂停控制帧 |
| | 0 | - | - | 1 | 0 | 0 | - | 匹配完美/组过滤器时不通过，如果PCFRM = 0x，丢弃暂停控制帧 |
| | 0 | 0 | - | 1 | 1 | 0 | - | 匹配HASH过滤器时不通过，如果PCFRM = 0x，丢弃暂停控制帧 |

| 帧类型 | PM | HPFLT | HUF | DAIFLT | HMF | MFD | BFRMD | 目标地址过滤器操作 |
|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | - | 1 | 1 | 0 | - | 匹配HASH或者完美/组过滤器时不通过，如果PCFRM = 0x，丢弃暂停控制帧 |

**表 27-6. 源地址过滤器结果列表**

| 帧类型 | PM | SAIFLT | SAFLT | 源地址过滤器操作 |
|---|---|---|---|---|
| 单播帧 | 1 | - | - | 所有帧通过 |
| | 0 | 0 | 0 | 匹配完美/组过滤器时返回通过状态，不匹配时状态为不通过，但不丢弃不通过的帧 |
| | 0 | 1 | 0 | 匹配完美/组过滤器时返回不通过状态，但不丢弃帧 |
| | 0 | 0 | 1 | 匹配完美/组过滤器时通过，丢弃不通过的帧 |
| | 0 | 1 | 1 | 匹配完美/组过滤器时不通过，丢弃不通过的帧 |

**混杂模式**

若设置ENET_MAC_FRMF寄存器的PM位为'1'将使能混杂模式，此时地址过滤器无效，所有帧均可通过过滤器。同时接收状态信息的目标地址/源地址错误位总是为'0'。

**暂停控制帧过滤**

MAC会检测接收到的控制帧内的6字节目标地址域，若ENET_MAC_FCTL寄存器的UPFDT位设为0，则判断目标地址域的值是否符合IEEE 802.3规范控制帧的唯一值(0x0180 C200 0001)。若ENET_MAC_FCTL寄存器的UPFDT位设为1，则在与IEEE 802.3规范定义的唯一值比较外，同时与控制器所设置的MAC地址逐位比较。如果目标地址域比较通过且接收流控制被使能(ENET_MAC_FCTL的RFCEN位被置1)，则相应暂停控制帧功能将被触发。这个通过过滤的暂停帧是否会被转发给应用取决于ENET_MAC_FRMF寄存器的PCFRM[1:0]位设置。

## MAC 的接收流程

MAC接收到的帧都会被送入RxFIFO中。MAC接收到帧后会剥离其前导码和帧首界定码，并从帧首界定码后的第一个字节(目标地址)开始向FIFO发送帧数据。如果使能了IEEE 1588时间戳，MAC会在检测到帧的帧首界定码的时候记录下系统的当前时间。如果这个帧通过地址过滤器的检查，MAC会把这个时间戳通过接收描述符一并发给应用程序。

若ENET_MAC_CFG寄存器的APCD位置位，且接收到的帧长度/类型域的值小于0x600时，MAC将自动剥离填充域和帧校验序列。MAC会在向RxFIFO发送完帧长度/类型域规定字节数后，丢弃包括帧校验序列在内的余下字节。如果长度/类型域的值大于或等于0x600，则忽略APCD位，由TFCD位来确定是否自动剥离帧校验序列。

若看门狗定时器被使能(ENET_MAC_CFG寄存器中的WDD位被复位)，当帧长度超过2048字节时将被切断。即使看门狗定时器被禁能，MAC仍然会切断长度大于16384字节的帧。

当RxFIFO工作于直通模式时，如果FIFO中的数据量大于门限值(可通过ENET_DMA_CTL寄存器的RTHC位设置)，就开始从FIFO中取出数据，并通知DMA接收。当FIFO完成取出整个帧后，MAC控制器将接收状态信息字发送给DMA控制器以回写到接收描述符中。在这种模式下，假如一个帧开始由FIFO取出由DMA发送到应用程序，则即使检测到错误，帧也会一直接收直到

整个帧接收完毕。由于错误信息也要等到此时才会发送给DMA控制器，此时帧的前部分已经被DMA接收，所以在这种模式下将MAC设置成将所有错误帧丢弃将无效。

当RxFIFO工作于存储转发模式(通过ENET_DMA_CTL寄存器的RSFD位设置)时，DMA只在RxFIFO完整地收到一帧后，才将其读出。此模式下，如果MAC设置成将所有错误帧丢弃，那么DMA只会读出合法的帧，并转发给应用程序。一旦MAC在接口上检测到帧首界定码就会启动接收过程。MAC控制器在处理帧之前会剥离前导码和帧首界定码。会通过过滤器检查帧的报头，并用帧校验序列核对帧的CRC值。如果帧没能通过地址滤波器，MAC控制器就会丢弃该帧。

## MAC 的接收管理

### 多个帧的接收处理

与TxFIFO不同，由于帧的状态信息紧随在帧数据之后，MAC可以判断接受帧的状态，因此第二个接收帧的传送是紧接着第一个接收帧的数据与状态信息的，只要RxFIFO未满，就可以存放任意数量的帧。

### 错误处理

- 在从MAC接收到EOF之前，RxFIFO已满。则MAC控制器会将整个帧丢弃并返回一个溢出状态。同时将溢出计数器加1；
- 若RxFIFO设置成存储转发模式，MAC可以过滤并丢弃所有的错误帧。但根据ENET_DMA_CTL寄存器的FERF和FUF位的设置，RxFIFO仍可以接收错误帧和长度低于最小帧长的帧；
- 若RxFIFO设置成直通模式，并不能将所有的错误帧都丢弃，仅当DMA从RxFIFO读出帧的SOF时，RxFIFO也已获得了该帧的错误状态时可以丢弃错误帧。

### 流控模块

MAC 控制器主要通过背压(半双工模式)和暂停控制帧(全双工模式)来管理帧的发送流控。

- 半双工模式流控：背压

当 MAC 采用半双工模式进行通讯时，如果设置了发送流控使能位(ENET_MAC_FCTL 寄存器的 TFCEN 位)，有两种情况可以触发背压流控。背压流控是通过发送一个 32 位的堵塞信号 0x5555 5555，通知所有其他站点发生了冲突。两种触发情况中，第一种是通过置位 ENET_MAC_FCTL 寄存器的 FLCB/BKPA 位来使能发送流控。第二种情况在接收帧时发生，MAC 在接收帧的过程中，RxFIFO 中字节数不断增大，当接收数目超过流控激活阈值(ENET_MAC_FCTH 寄存器中的 RFA 位)，MAC 将置位背压挂起标志。若背压挂起标志置位了，且又有新的帧到来，MAC 将发送堵塞信号以延迟一段背压时间再接收帧。在背压时间结束后，PHY 会重新发送这个新的帧。若在背压期间，RxFIFO 中字节数大于等于流控失活阈值(ENET_MAC_FCTH 寄存器中的 RFD 位)，则 MAC 会再次发送背压信号；反之，则 MAC 将复位背压挂起标志，并可以接收新的帧，不再发送堵塞信号。

- 全双工模式流控：暂停帧

对于全双工模式，MAC 控制器使用"暂停帧"进行流控制。这种方式可以使接收端能够命令发送

端暂停一段时间再发送，如当接收缓冲区快要溢出的情况。如果设置了发送流控使能位(ENET_MAC_FCTL 寄存器的 TFCEN 位)，在全双工模式下，MAC 会在以下两种情况下产生并发送暂停帧。两种情况分别为：

1) 应用程序把 ENET_MAC_FCTL 寄存器的 FLCB/BKPA 位置位，将立即发送一个暂停帧。这个暂停帧指定的暂停时间为 ENET_MAC_FCTL 寄存器中 PTM 位配置好的暂停时间值。如果应用程序前面要求了一段时间的暂停，但在这段时间内，应用程序准备好了，可以不需要剩余的暂停时间了，这时应用程序需要发一个零时间片暂停帧来通知发送方可以继续发送了。零时间片暂停帧是通过设置 ENET_MAC_FCTL 寄存器中的 PTM 位为 0，并将 FLCB/BKPA 位置位来发送的；
2) 在 RxFIFO 满足一定的条件下，MAC 会自动发送暂停帧。在接收过程中，RxFIFO 不停地有数据进来，同时 RxFIFO 也取出数据给 RxDMA，如果 RxFIFO 取出数据的频率小于其接收数据的频率，RxFIFO 中的数据就会越来越多。一旦 RxFIFO 中的数据量超过了流控的激活阈值(ENET_MAC_FCTH 寄存器中的 RFA 位)，MAC 将发送一个暂停时间为 PTM 位定义的值的暂停帧。发送暂停帧之后，MAC 将启动一个计数器，计数器的时间由 ENET_MAC_FCTL 寄存器的 PLTS 位定义，当到了计数器规定的时间，MAC 将重新检查 RxFIFO。此时若 RxFIFO 中的数据量仍然大于流控激活阈值，MAC 将再次发送一个暂停帧。若 RxFIFO 中的数据量小于流控失活阈值，并且 ENET_MAC_FCTL 寄存器中的 DZQP 位被复位，则 MAC 将发送一个零时间片暂停帧。这个零时间片暂停帧用于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。

MAC 通过如下方式管理帧的接收流控：

在全双工模式下，MAC 能够检测暂停帧，并按照暂停帧中的暂停时间域参数，在暂停一定时间后再发送数据。可以通过设置 ENET_MAC_FCTL 寄存器的 RFCEN 位，使能或者取消暂停帧检测功能。如果没有使能该功能，则 MAC 会忽略接收到的暂停帧。若使能了该功能，MAC 将能够对接收到的暂停帧进行解码。类型域、操作数域和暂停时间域都将能够被 MAC 识别。在暂停期间，如果收到一个新的暂停帧，则新的暂停时间将立即被加载到暂停时间计数器中。如果接到收的暂停时间域值为 0，则 MAC 会停止暂停时间计数器，恢复数据的发送。通过配置 ENET_MAC_FRMF 寄存器的 PCFRM 位值，来处理这些接收到的控制帧。

### 校验和引擎

以太网控制器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，以及在接收时侦测校验和错误。

如下描述了发送帧的校验和的操作功能。

注意：只有将 ENET_DMA_CTL 寄存器的 TSFD 位置为'1'(TxFIFO 设置成存储转发模式)，同时必须保证 TxFIFO 的深度足够容纳将要发送的完整帧时，才能使能此功能。若 FIFO 深度小于帧长度，则仅仅计算和插入 IPv4 报头的校验和域。

欲了解 IPv4、TCP、UDP、ICMP、IPv6 和 ICMPv6 报头的规范，请分别查阅 IETF 规范 RFC 791、RFC 793、RFC 768、RFC 792、RFC 2460 和 RFC 4443。

■ IP头校验和

若以太网帧的类型域值为 0x0800 同时 IP 数据包的版本域值为 0x4，则校验和模块标记其为

IPv4 数据包并会用计算结果取代帧的校验和域的内容。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。IP 头校验和计算完毕之后，其结果会写到发送描述符 0 的 IPHE 位。当发生下述情况时，IPHE 错误状态位会被硬件置'1'：

1) 对于IPv4数据帧：
   a) 接收到的以太网类型域值为0x0800，但IP报头版本域的值不等于0x4；
   b) IPv4报头长度域的值大于帧的总长度；
   c) IPv4报头长度域值小于IP报头总长0x5(20字节)。

2) 对于IPv6数据帧：
   a) 接收到的以太网类型域值为0x86dd，但IP报头版本域的值不等于0x6；
   b) 帧在完全接收IPv6报头或者扩展报头之前结束。IPv6标准报头长度为40字节，扩展报头包含相应的报头长度字段。

■ TCP/UDP/ICMP校验和

校验和模块通过分析IPv4或IPv6报头(包括扩展报头)来判断帧的类型(TCP、UDP或ICMP)。

当帧发生以下情况时，将绕过校验和功能，校验和模块不对这些帧进行处理：

1) 不完整的IPv4或IPv6帧；
2) 包含安全功能的IP帧(如验证报头或者封装有安全数据)；
3) 非TCP/UDP/ICMPv4/ICMPv6数据的IP帧；
4) 带路由报头的IPv6帧。

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的伪首部。并假定输入帧的校验和字段已有值。校验和字段包含在校验和计算中，在计算完成后插入并替换原校验和域的值；
2) 校验和计算包括TCP、UDP或者ICMPv6的伪首部。将传输帧的校验和字段清零。进行校验和的计算，计算完成后插入传输帧的原校验和域。

校验和计算完毕之后，其结果会写到发送描述符0的IPPE位。当发生下述情况时，IPPE错误状态位会被硬件置'1'：

1) 在存储转发模式下，帧未被完整写入FIFO之前就被转发给MAC控制器；
2) 帧已发送完毕，但MAC从FIFO中取出的数据包字节数小于IP报头中数据长度域标明的字节数。

如果数据包长度大于标明的长度，不会报告错误，之后的数据会被当成填充字节而丢弃。如果检测到第一类错误情况，校验和的值不会插入TCP、UDP或者ICMP报头。如果检测到第二类错误情况，仍然会把校验和计算结果插入报头的相应域。

**注意：**无论采用哪种模式，对于 IPv4 上的 ICMP 数据包，由于这类数据包没有定义伪报头，为正确计算其校验和，校验和域内容必须为 0x0000。

接收帧校验和的操作功能描述如下所述。

置位 ENET_MAC_CFG 寄存器的 IPFCO 位，可以使能接收校验和模块。接收校验和模块可以计算 IPv4 报头的校验和，并检查它是否与 IPv4 报头的校验和域的内容相匹配以外。MAC 可

根据检查接收到的以太网帧类型域是 0x0800 还是 0x86dd，来判别是 IPv4 帧还是 IPv6 帧，这个方法也用于带 VLAN 标签的帧识别。DMA 接收描述符的报头校验和错误位(接收描述符 0 中的 IPHERR 位)反映了对报头的校验和结果，该位在接收到的 IP 报头出现下述错误时被置 1：

- 计算的IPv4报头的校验和值与其校验和域的内容不匹配；
- 以太网类型域值指示的数据类型与IP报头版本域不匹配；
- 接收到的帧长少于IPv4报头长度域指示的长度，或者IPv4/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。计算过程包括TCP/UDP/ICMPv6伪报头的数据。DMA接收描述符(接收描述符0的PCERR位)的数据校验和错误位反映了对数据的校验和结果，该位在接收到的IP数据包数据出现下述错误时被置1：

- 计算的TCP、UDP或ICMP校验和与其帧的TCP、UDP或ICMP校验和域值不匹配；
- 收到的TCP、UDP或者ICMP数据长度与IP报头给出的长度不符。

接收校验和模块不计算下列情况：不完整的IP数据包、带安全功能的IP数据包、IPv6路由报头以及数据类型不是TCP、UDP或者ICMP的数据包。

#### MAC 回环模式

通常地，回环模式用于应用程序对系统硬件和软件的测试与调试。通过将ENET_MAC_CFG寄存器的LBM位置'1'，可以使能MAC回环模式。在该模式下，MAC发射端把帧发送到自身的接收端上。该模式默认为关闭。

### 27.3.3. DMA 控制器描述

为了减少CPU的干预，设计了以太网专用DMA控制器，用于实现FIFO和系统存储之间的帧数据传输。CPU和DMA之间的的通讯通过2种数据结构实现。分别是：1)描述符列表(链结构或环结构)和数据缓存；2)控制和状态寄存器。应用程序需要开辟存储描述符列表及数据缓存用到的物理内存。在存储器里，描述符是以指向缓存的指针的形式存放。有2个描述符队列，一个用作发送，另一个用作接收。两个队列的基地址分别存放在ENET_DMA_TDTADDR寄存器和ENET_DMA_RDTADDR寄存器中。当DFM位为0时，发送描述符由四个描述符字发送描述符0-4)组成，当DFM位为1时，发送描述符由八个描述符字(发送描述符0-8)组成。同样的，当DFM位为0时，接收描述符由四个描述符字(接收描述符0-4)组成，当DFM位为1时，接收描述符由八个描述符字(接收描述符0-8)组成。每个描述符可以指向最多2个缓存用来存储帧的数据。根据描述符列表类型是环结构还是链结构，来决定第二个缓存是被配置为第二个数据存储地址，还是下一个描述符地址。数据缓存存放在MCU的物理内存里，可以存放一个帧的全部或者部分，但是不允许存放不属于同一个帧的数据。描述符队列可以是显性(链结构)或者隐性(环结构)的方式前向连接的。通过设置接收描述符1的RCHM位和发送描述符的TCHM位为'1'，可以实现描述符的显性连接，此时接收描述符2及发送描述符2中将存放缓存地址，接收描述符3及发送描述符3中将存放下一个描述符的地址，这种链接的描述符也可以称为描述符的链结构。通过设置接收描述符1的RCHM位和发送描述符的TCHM位为'0'，可以实现描述符的隐性连接，此时接

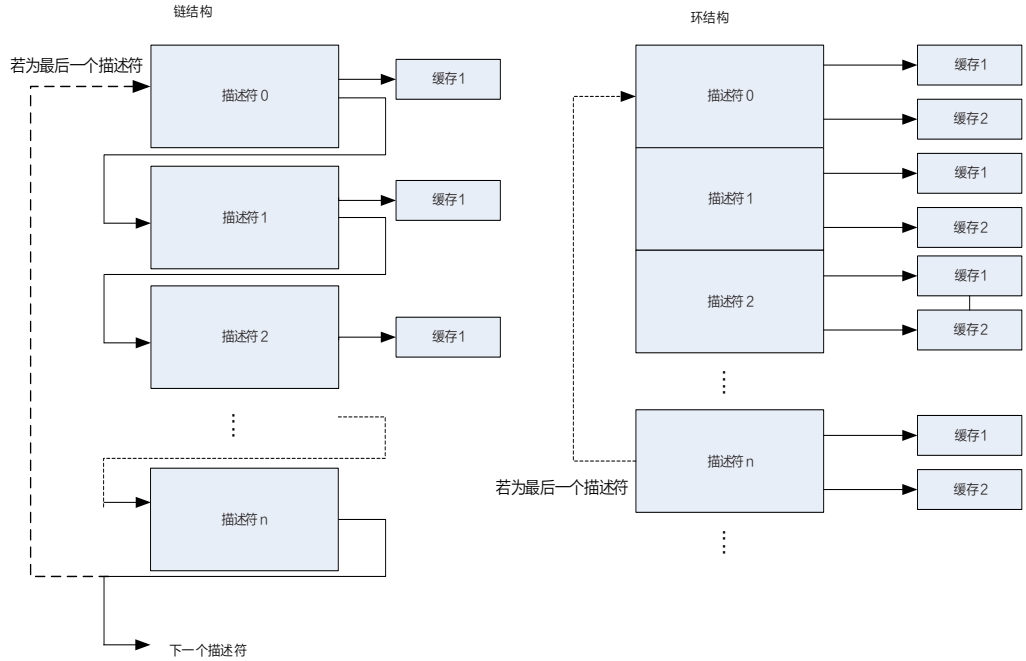
收描述符2/发送描述符2，接收描述符3/发送描述符3中都将存放缓存地址，这种链接的描述符也可以称为描述符的环结构。在使用当前的描述符所指向的缓存地址时，描述符指针就指向下一个描述符。当使用链结构时，描述符指针指向的是第二个缓存。当使用环结构，根据下式计算描述符指针下一个所指向的地址：

DFM=0：下个描述符地址 = 当前描述符地址 + 16 + DPSL * 4

DFM=1：下个描述符地址 = 当前描述符地址 + 32 + DPSL * 4

若当前描述符是描述符列表的最后一个描述符，环结构下必须设置发送描述符0的位21或接收描述符1的位15以标识当前描述符为列表的最后一个。此时下一个描述符又指向描述符列表的第一个。链结构下还可以通过设置发送描述符3或接收描述符3的值指向描述符列表中第一个的地址。DMA一旦检测到帧结束就会跳到下一个帧的缓存。

**图 27-6. 描述符的环结构和链结构**




## 数据缓存地址对齐

以太网DMA控制器支持所有对齐类型：字节对齐，半字对齐，字对齐。这意味着应用程序可将发送和接收数据缓存地址配置到任意地址。但是，在DMA发起传输的时候，总是以字对齐的方式访问地址。对于读和写缓存的访问也不一样。示例如下：

- 读缓存示例：如果发送缓存的地址为0x2000 0AB2，并需要传输15字节。在开始读操作后，DMA实际会从地址0x2000 0AB0，0x2000 0AB4，0x2000 0AB8，0x2000 0ABC和0x2000 0AC0先读5个字，但是在往FIFO发送数据的时候，会丢弃头2个字节(0x2000 0AB0和0x2000 0AB1)和最后3个字节(0x2000 0AC1，0x2000 0AC2和0x2000 0AC3)。

- 写缓存示例：如果接收缓存的地址为0x2000 0CD2，并需要传输16字节。在开始写操作后，DMA实际会从地址0x2000 0CD0到0x2000 0CE0先写5个32位数据。但是头2个字节

(0x2000 0CD0和0x2000 0CD1)和末尾的2个字节(0x2000 0CE2和0x2000 0CE3)会用虚拟字节替代。

**注意：**DMA控制器不会写任何数据到定义的缓存区之外的地址。

### 缓冲区有效长度

发送帧的过程中，TxDMA会传输与发送描述符1中标明的缓存有效长度的字节给MAC控制器。如前所述，一个发送帧可以用多个描述符来描述一个帧，即一个帧的数据可以处于多个不同的缓存中。如果DMA控制器读取的发送描述符0的FSG位为'1，那么DMA就明确了当前缓存存储的是一个新的帧，并标记发送的第一个字节是帧首。如果DMA控制器读取的发送描述符0的LSG位为'1'，则DMA就明确了当前缓存存储的是当前帧的最后一部分数据。通常来说一个帧只存在一个缓存里(因为缓存的大小对于一个正常的帧来说足够大了)，因此FSG和LSG位会在一个相同的描述符中同时置位。

接收帧的过程中，接收帧的缓存长度域值必须是字对齐的。对于字对齐或非字对齐的缓存地址，接收操作与发送操作不大相同。如果接收缓存地址是字对齐的，则与发送流程是类似的，缓存的有效长度为由接收描述符1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接收描述符1中配置的值。缓存有效长度值应为接收描述符1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2048字节，缓存地址为0x2000 0001，地址的低2位值为0b01，那么缓存有效长度为2047个字节，范围从0x2000 0001(帧首)到0x2000 07FF。

当收到了一个帧起始SOF，则DMA控制器将FSG位置位，当收到一个帧结束EOF时，则LSG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FSG和LSG位将在同个描述符中被置位。实际接收的帧长度可从接收描述符0的FRML位域获取。从而应用程序可计算未被使用的缓存空间。RxDMA总是用新的描述符来接收下一帧。

### TxDMA 和 RxDMA 的仲裁器

DMA的仲裁器设计了两种仲裁方式用于提高DMA发送与接收控制器的效率：固定和轮询优先级。设置ENET_DMA_BCTL寄存器DAB位为'0'，选择轮询优先级，在TxDMA和RxDMA同时要求访问数据总线的时候，按照ENET_DMA_BCTL寄存器RTPR位设定的比例对其访问进行分配。设置DAB位为'1'选择固定优先级，此时RxDMA和TxDMA同时要求访问时，RxDMA总是对总线拥有更高的访问优先级。

### DMA 错误状态

若DMA在传输过程中出现了错误的总线响应，那么DMA控制器认为发生了一个致命错误，会立刻停止所有操作，并更新状态寄存器ENET_DMA_STAT。在发生类似的致命错误(响应错误)之后，应用程序必须复位以太网外设并重新初始化DMA，DMA才能恢复操作。

### TxDMA 与 RxDMA 控制器的初始化

在使用DMA控制器之前，必须按如下步骤对DMA进行初始化：

1. 对ENET_DMA_BCTL寄存器进行总线访问参数的相关设置；
2. 对ENET_DMA_INTEN寄存器进行设置，屏蔽不需要的中断源；
3. 将发送描述符列表和接收描述符列表的基地址分别写入ENET_DMA_TDTADDR寄存器与ENET_DMA_RDTADDR寄存器中；
4. 对相关的寄存器进行期望的过滤器配置；
5. 根据从PHY读出的自协商的结果，设置SPD位和DPM位的值，来选择通讯模式(半/全双工)及通讯速度(10Mbit/s或100Mbit/s)。将ENET_MAC_CFG寄存器的TEN和REN位置'1'，使能MAC的发送和接收操作；
6. 设置ENET_DMA_CTL寄存器的位STE和位SRE为'1'，使能DMA发送和接收器。

**注意：**如果HCLK频率过低，应用程序可以先使能DMA接收器，再将ENET_MAC_CFG寄存器的REN位置'1'，以避免RxFIFO在启动的时候溢出。

### DMA 发送帧处理

如前所述，一个帧可以分散在不同缓存内，这意味着需要多个描述符。当 FSG 位置位，表示当前描述符指向的缓存为帧头，当 LSG 位置位，表示当前描述符指向的缓存为帧尾。对于当前帧其他描述符(LSG 位为'0'的描述符)，TxDMA 控制器仅修改清零其 DAV 位。在这最后一个缓存的数据发送完毕以后，DMA 会将整个帧的发送状态信息，写入最后一个的发送描述符 0 并返回。将数据从系统存储传输到 FIFO，开始发送数据，但实际上真正的数据发送是由 TxDMA 模式决定的：直通模式和存储转发模式。直通模式在 FIFO 中的字节数大于所配置的阈值时，数据将取出到 MAC 发送。存储转发模式在整个帧数据都传入 FIFO 后或 FIFO 快要填满时再取出数据给 MAC 进行发送。

### DMA 发送管理

#### 发送缓存区中第二帧操作

如果ENET_DMA_CTL寄存器中OSF位为'0'，则发送顺序为：首先读取发送描述符，然后从系统存储读取数据写到FIFO，再将帧数据通过MAC放到接口上，最后等待数据发送完毕后将发送状态写回描述符。

上述是TxDMA的标准发送流程，但当HCLK远远大于TX_CLK时，在发送两个帧时发送效率将显著降低。

为避免上述提及的情况，应用程序可将OSF位置位。在此情况下，第二帧的数据可以不等待第一帧的描述符状态信息被写回，就先读取内存里的第二帧数据，并把它们送进FIFO。OSF功能仅在两相邻帧之间起作用。

#### TxDMA 操作模式(A)(默认)：非 OSF

在默认模式下，TxDMA控制器的工作流程如下：

1. 初始化帧数据到发送缓存，并对发送描述符0-3进行设置，置发送描述符0的DAV位为'1'；
2. 将ENET_DMA_CTL寄存器的STE位置为'1'，使能TxDMA控制器；
3. TxDMA控制器开始轮询发送描述符列表来获取待发送的帧。如果TxDMA检测到发送描述符0的位DAV为0，或者发生了错误，则控制器就会终止传输进入挂起状态，并设置

ENET_DMA_STAT寄存器的发送缓存不可用位(位2)和正常中断汇总位(位16)为'1'。如果处于挂起状态，则发送控制器操作跳至步骤8；

4. 如果取到的描述符标志位显示该描述符由DMA占有(DAV位被置'1')，那么DMA从描述符中解析出所配置的发送帧以及发送数据缓存的地址；
5. DMA从内存中取出数据并将数据存入TxFIFO；
6. TxDMA控制器会一直轮询描述符列表直到帧结尾被传送出去(LSG位置位)。如果当前描述符的LSG位为'0'，则在所有缓存数据送入TxFIFO之后，将DAV位清零以关闭这个描述符。然后TxDMA控制器等待写回描述符状态，以及IEEE 1588时间戳值(如果使能了时间戳功能)；
7. 在整个帧发送完成以后，仅当发送描述符0位INTC为'1'时，发送状态位(ENET_DMA_STAT寄存器中的TS位)会被置位。此时若使能了DMA中断，将进入相应中断。然后DMA控制器返回步骤3，继续处理下一帧；
8. 在挂起状态下，如果向发送查询使能寄存器ENET_DMA_TPEN写入任意值，并清除发送溢出标志位，TxDMA将重新回到运行状态，尝试重新获取描述符。发送控制器操作回到步骤3。

**TxDMA 操作模式(B)：OSF**

在操作第二帧(OSF)模式下，TxDMA可以不必等到前一帧的状态信息写回，就发送下一帧。如果系统时钟频率远远大于MAC频率(10Mbit/s或100Mbit/s)，这种情况OSF模式可以提高发送效率。设置ENET_DMA_CTL寄存器的位OSF为'1'，进入此模式。DMA在发送完前一帧数据后，不必等到前一阵的状态写回，而是立即查询第二帧的发送描述符，如果第二帧发送描述符的DAV位与FSG位都置1，那么TxDMA立即读取第二帧的帧数据并将其存入MAC FIFO。

在OSF模式下，TxDMA的操作流程如下：

1. 按照TxDMA默认模式的步骤1-6操作；
2. DMA不等到关闭前一帧的最后一个描述符(LSG位为'1')，就直接取下一个描述符；
3. 如果取到的描述符标明被DMA占有(DAV位为'1')，那么就从解析的发送缓存地址中读取下一帧的数据。如果DAV位为'0'即DMA不占有这个描述符，则TxDMA进入挂起状态并跳到步骤7；
4. TxDMA控制器会一直轮询描述符列表直到帧结尾被传送出去。如果一个帧由多个描述符描述，则中间描述符会在获取之后就被关闭；
5. TxDMA等待前一帧的发送状态信息和时间戳(如果使能了时间戳功能)，在接收到状态信息后，DMA会把DAV位为'0'的状态信息写入发送描述符0，将该描述符的占有权交还给CPU进行操作；
6. 在整个帧发送完成以后，仅当发送描述符0位INTC为'1'时，发送状态位(ENET_DMA_STAT寄存器中的TS位)会被置位。此时若使能了DMA中断，将进入相应中断。如果前一个帧返回的状态信息正常则跳到步骤3。若显示有数据下溢错误，TxDMA进入挂起状态，并跳到步骤7；
7. 在挂起状态下，如果TxDMA收到一个发送帧的待处理的状态信息和时间戳(若使能了时间戳)，则TxDMA将这些信息写入发送描述符，并将相应描述符的的DAV位清零。随后设置相关的中断标志位并回到暂停状态；
8. 在挂起状态下，如果向发送查询使能寄存器ENET_DMA_TPEN写入任意值并将溢出中断标志位清零，TxDMA将回到运行状态，尝试重新获取描述符。发送控制器操作根据是否有待处理的状态信息跳到步骤1或者步骤2。

### 发送帧格式

根据前述的IEEE 802.3规范，一个正常的发送帧应该由以下及部分构成：前导码，帧首界定码SFD，目标地址DA，源地址SA，QTAG前缀(可选)，长度/类型域LT，数据，PAD填充域(可选)，和帧校验序列FCS。

前导码和帧首界定码都是由MAC自动生成的，因此应用程序只需要存储目标地址，源地址，QTAG(若需要)，长度/类型，数据，填充域(若需要)，帧校验序列(若需要)。如果帧需要填充位，即缓存中没有存储填充位和帧校验序列部分，则应用程序可配置自动生成帧校验序列和填充位功能。如果帧仅需帧校验序列，即缓存中没有存储帧校验序列部分，则应用程序可配置自动生成帧校验序列。DPAD位和DCRC位用于配置填充位和帧校验序列的自动生成。

### 发送查询挂起后的处理

当传输开始后DMA会不断对发送描述符进行查询，当发生如下情况时，会导致DMA进入挂起状态，并暂停发送。此时当前描述符固定为暂停前的最后一个描述符。

- DMA检测到发送描述符0的DAV位为0，此时CPU占有描述符，则会进入挂起状态，并暂停查询。同时设置ENET_DMA_STAT寄存器的正常中断总结位NI和发送缓存不可用位TBU为'1'；
- 当接口在发送帧的过程中MAC FIFO为空，意味着检测到了数据下溢错误。在此情况下，

  设置ENET_DMA_STAT寄存器的异常中断总结位AI和发送数据下溢位TU为'1'，同时把该信息写入发送描述符0。

### 带 IEEE 1588 时间戳的 TxDMA 描述符格式

如果设置TTSEN位为'1'，则使能了IEEE1588功能。TxDMA控制器会在帧发送完成后，将时间戳写入描述符同时设置TTMSS位为'1'。写时间戳值的地址由ENET_DMA_BCTL寄存器的DFM位决定。如果DFM位为'0'即描述符格式为常规模式，则时间戳将覆盖写到发送描述符2和发送描述符3上。如果DFM位为'1'即描述符格式为增强型模式，则时间戳将写到发送描述符6和发送描述符7上，发送描述符2和发送描述符3中的值保持不变。

### 常规 TxDMA 描述符

常规TxDMA描述符结构体包含4个32位字，发送描述符0~发送描述符3。发送描述符0~发送描述符3的的位定义如下：

**注意：**若一个帧由多个描述符表示，则对于描述符的控制位(除了INTC位)只有第一个描述符的才有效。状态信息和时间戳(若使能了时间戳功能)只写回到最后一个描述符。

**图 27-7. 常规发送描述符**




■ 发送描述符0

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DAV | INTC | LSG | FSG | DCRC | DPAD | TTSEN | 保留 | CM[1:0] | | TERM | TCHM | 保留 | | TTMSS | IPHE |
| rw | rw | rw | rw | rw | rw | rw | | rw | | rw | rw | | | rw | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ES | JT | FRMF | IPPE | LCA | NCA | LCO | ECO | VFRM | COCNT[3:0] | | | | EXD | UFE | DB |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | DAV | DAV位<br>DMA会在将帧完整传输或者描述符指向的缓存里的数据全部被读出以后把该位清'0'。当一个帧位于多个缓存中时，第一个缓存描述符的DAV位，必须在后面缓存描述符的DAV位全部置'1'以后，才能置'1'。<br>0：表示CPU占有描述符<br>1：表示DMA占有描述符 |
| 30 | INTC | 完成时中断位<br>LSG位置位后，此位才有效。<br>0：帧发送完成时，ENET_DMA_STAT寄存器的TS位不被置位<br>1：帧发送完成时，ENET_DMA_STAT寄存器的TS位被置位 |
| 29 | LSG | 最后分块位<br>此位指示缓存是否包含帧的最后一个分块。<br>0：该描述符缓存中没有存放帧的最后一个分块<br>1：该描述符缓存中存放有帧的最后一个分块 |
| 28 | FSG | 第一分块位<br>此位指示缓存是否包含帧的第一个分块。<br>0：该描述符缓存中没有存放帧的第一个分块<br>1：该描述符缓存中存放有帧的第一个分块 |
| 27 | DCRC | 不计算CRC位<br>只有在FSG位置位时，此位才有效。<br>0：MAC 在传输帧末尾自动插入 CRC 域<br>1：MAC不在传输帧末尾自动插入CRC域 |

| | | |
|---|---|---|
| 26 | DPAD | 不填充位<br>只有在 FSG 位置位时，此位才有效。<br>0：DMA 对传输帧自动添加填充字节，并且插入 CRC 数值。发生填充时，CRC 会被插入，忽略 DCRC 位的值。<br>1：MAC 不对传输帧自动填充字节<br>**注意**：此处的传输帧小于64字节。 |
| 25 | TTSEN | 使能发送时间戳位<br>只有在 FSG 位置位时，此位才有效。<br>0：发送时间戳功能失能<br>1：当ENET_PTP_TSCTL寄存器的TMSEN位为'1'时，传输帧的IEEE1588硬件时间戳功能使能。 |
| 24 | 保留 | 必须保持复位值。 |
| 23:22 | CM[1:0] | 校验和插入模式位<br>0x0：不插入校验和<br>0x1：只使能硬件IP报头的校验和计算和插入<br>0x2：使能硬件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<br>0x3：使能硬件IP报头和数据域的校验和计算和插入，也计算伪报头的校验和 |
| 21 | TERM | 环形发送结束模式位<br>该位仅在环模式下使用，且比TCHM位具有更高优先级。<br>0：当前描述符还不是描述符队列的最后一个<br>1：当前描述符到达描述符队列的最后一个，DMA返回列表的基地址 |
| 20 | TCHM | 第二地址链表模式位<br>该位在链模式下使用。该位为'1'时，忽略TB2S[12:0]的值。<br>0：描述符里的第二个地址是第二缓存的地址<br>1：描述符里的第二个地址是下一个描述符的地址，而不是第二个缓存的地址 |
| 19:18 | 保留 | 必须保持复位值。 |
| 17 | TTMSS | 发送时间戳状态位<br>只有在LSG位置位时，此位才有效。<br>0：还未记录帧的时间戳信息<br>1：记录下了描述符对应的帧时间戳，记录的时间戳放在发送描述符2(或发送描述符6，假如DFM=1)和发送描述符3(或发送描述符7，假如DFM=1)处。 |
| 16 | IPHE | IP报头错误位<br>发生下列任意一种情况，则产生IP报头错误：<br>IPv4帧：<br>1)报头长度域值小于0x5。<br>2)报头长度域值与报头的长度不符。<br>3)报头版本域值与帧长度/类型域值不匹配。<br>IPv6帧：<br>1)主报头长度不足40字节。<br>2)报头版本域值与帧长度/类型域值不匹配。 |

| | | |
|---|---|---|
| | | 0：未发现IP数据包报头的错误<br>1：MAC发送端发现了IP数据包报头的错误 |
| 15 | ES | 错误汇总<br>该位为下列位的逻辑"或"：<br>IPHE：IP 报头错误<br>JT：Jabber 超时<br>FRMF：帧清空<br>IPPE：IP 数据错误<br>LCA：载波丢失<br>NCA：无载波<br>LCO：延迟冲突<br>ECO：过度冲突<br>EXD：过度顺延<br>UFE：数据下溢错误 |
| 14 | JT | Jabber超时位<br>该位仅当JBD位复位时才会被置'1'。<br>0：未发生Jabber超时<br>1：MAC发送端发生了Jabber超时 |
| 13 | FRMF | 帧清空位<br>置1时，清空TxFIFO中的数据。 |
| 12 | IPPE | IP数据错误位<br>发送端会核对IPv4或者IPv6报头的数据长度域值与实际收到的TCP、UDP和ICMP数据数目，不符合就置'1'报错。<br>0：未发生IP数据错误<br>1：MAC发送端发现了IP数据包的TCP、UDP或者ICMP的IP数据错误。 |
| 11 | LCA | 载波丢失位<br>在发送时，如果CRS信号在一个或一个以上发送时钟周期中为无效状态，并且没有发生冲突，则载波丢失将概率性发生。<br>该位只有在半双工模式下有效。<br>0：未发生载波丢失<br>1：帧发送的时候发生了载波丢失 |
| 10 | NCA | 无载波位<br>0：PHY的载波侦听信号有效<br>1：帧发送的时候PHY的载波侦听信号无效 |
| 9 | LCO | 延迟冲突位<br>如果冲突在64字节(包括前导符)发送之后发生，则这种情况称作延迟冲突。<br>0：未发生延迟冲突<br>1：发生了延迟冲突<br>**注意：**如果溢出错误位UFE置'1'，该位无效。 |

| | | |
|---|---|---|
| 8 | ECO | 过度冲突位<br>如果MAC设置寄存器的RTD(不进行重试)位为'1'，那么在发生一次冲突后，该位就置'1'。<br>如果MAC设置寄存器的RTD(不进行重试)位为'0'，那么在连续发生16次冲突后，该位置'1'。<br>若该位置位，则中止当前帧的发送。<br>0：未发生过度冲突<br>1：发生了过度冲突 |
| 7 | VFRM | VLAN帧位<br>0：发送帧为普通帧<br>1：发送的帧是VLAN帧 |
| 6:3 | COCNT[3:0] | 冲突计数位<br>只有在 ECO 位为 0 时，此位才有效。<br>该4位计数值记录了帧发送出去前出现的冲突次数。 |
| 2 | EXD | 过度顺延位<br>当MAC设置寄存器的顺延位DFC为'1'时有效。<br>0：未发生过度顺延<br>1：由于顺延超过3036字节的时间而结束发送 |
| 1 | UFE | 数据下溢错误位<br>数据下溢错误表示由于从系统存储传输数据到FIFO的速度过慢，导致DMA在发送帧的时候遇到了空的缓存。发送过程进入挂起状态，并将ENET_DMA_STAT寄存器的发送数据下溢位TU(位5)和发送状态位TS(位0)都置'1'。<br>0：未发生数据下溢错误<br>1：发生了数据下溢错误，MAC中止帧的发送。 |
| 0 | DB | 顺延位<br>该位指示了是否由于载波侦听信号CRS在MAC发送帧之前被占用，而导致发生帧的顺延。<br>该位只在半双工模式下有效。<br>0：未发生发送顺延<br>1：MAC发生了顺延，推迟发送。 |

■ 发送描述符1

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | TB2S[12:0] | | | | | | | | | | | | |
| | | | rw | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | TB1S[12:0] | | | | | | | | | | | | |
| | | | rw | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:29 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 28:16 | TB2S[12:0] | 发送缓存 2 大小<br>第二个数据缓存的大小(以字节记)。 |
| 15:13 | 保留 | 必须保持复位值。 |
| 12:0 | TB1S[12:0] | 发送缓存 1 大小<br>第一个数据缓存的大小(以字节记)。 |

■ 发送描述符2

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TB1AP/TTSL[31:16] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TB1AP/TTSL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | TB1AP/TTSL[31:0] | 发送缓存1地址指针/发送帧时间戳低32位<br>在发送帧之前，应用程序必须对这些位进行配置发生缓存1地址(TB1AP)，等到数据发送完后，DMA可以用它们存放帧的时间戳低32位(若DFM=0)。但若DFM=1，这些位将不会被修改，保持为缓存地址。<br>当这些位的值表示缓存1的物理地址(TTSL)时，对缓存的地址对齐不做限制。<br>当这些位的值表示时间戳低32位(TB1AP)时，当前描述符的TTSEN位和LSG位必须置位。 |

■ 发送描述符3

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TB2AP/TTSH[31:16] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TB2AP/TTSH[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

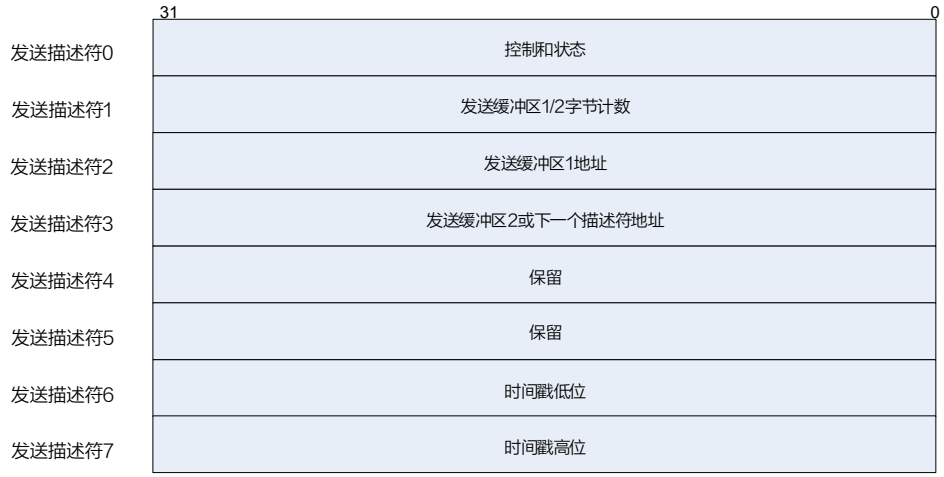
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | TB2AP/TTSH[31:0] | 发送缓存 2 地址指针(下个描述符地址)/发送帧时间戳高 32 位<br>在发送帧之前，应用程序必须对这些位进行配置发生缓存 2 地址(TB2AP)，或者配置下一个描述符地址(由描述符类型是链型还是环型决定)。等到数据发送完后，DMA 可以用它们存放帧的时间戳高 32 位 TTSH(若 DFM=0，且 TTSEN=1)。但若 DFM=1 或 TTSEN=0，这些位将不会被修改。<br>当这些位的值表示缓存 2 的物理地址时(TCHM=0)，对缓存的地址对齐不做限制。<br>当这些位的值表示下个描述符地址时(TCHM=1)，这些位必须是字对齐的。<br>当这些位的值表示时间戳高32位时，当前描述符的TTSEN位和LSG位必须置位。 |

### 增强 TxDMA 描述符

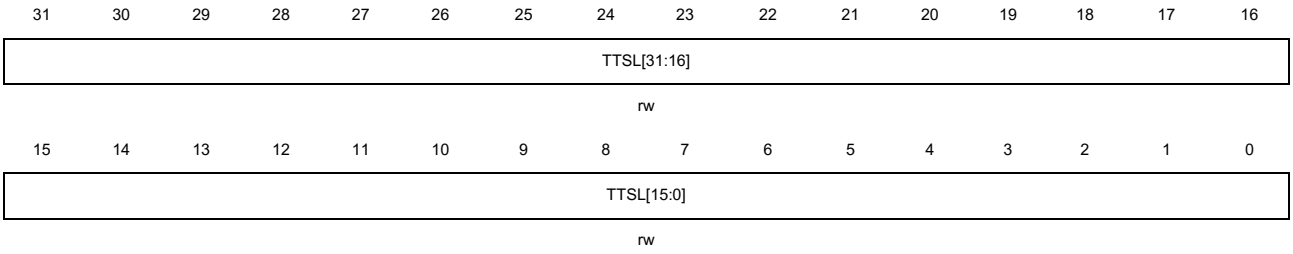
增强TxDMA描述符结构体包含8个32位字：发送描述符0~7。发送描述符0~3的位定义与常规TxDMA描述符相同，发送描述符4~7的位定义如下：

**注意：**若一个帧由多个描述符表示，则对于描述符的控制位(除了INTC位)只有第一个描述符的才有效。状态信息和时间戳(若使能了时间戳功能)只写回到最后一个描述符。

**图 27-8. 增强发送描述符**




- 发送描述符4
  所有位保留。
- 发送描述符5
  所有位保留。
- 发送描述符6




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | TTSL[31:0] | 发送时间戳低 32 位<br>当 TTSEN 位和 LSG 位都为 1 时，这些位用于记录当前发送帧的时间戳低 32 位值。 |

- 发送描述符7

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | TTSH[31:0] | 发送时间戳高 32 位<br>当 TTSEN 位和 LSG 位都为 1 时，这些位用于记录当前发送帧的时间戳高 32 位值。 |

### DMA 接收帧处理

当接口上出现一个帧的时候，MAC开始接收帧。同时，地址过滤模块开始工作，如果这个帧没有通过地址过滤，则MAC RxFIFO将忽略该帧，不会将其通过RxDMA转发给接收缓存。如果这个帧通过了地址过滤，则其在不同的转发条件满足时会被转发给接收缓存。在直通模式下，这个转发条件是指接收的帧长大于等于设好的接收阈值。在存储转发模式下，这个转发条件是指FIFO里存入了完整的帧时。在接收帧的过程中，当发生以下任意一种情况时，将丢弃RxFIFO中的数据，并且不转发数据：

- RxFIFO中数据少于64字节；
- 在接收过程中发生了冲突；
- 提前终止接收帧。

当满足转发条件时，RxDMA控制器开始将数据从RxFIFO中传输到接收缓存中。若当前缓存中包含了帧起始，则在RxDMA控制器写回帧接收状态的时候会将接收描述符0中的FDES位置位，以表明这个描述符中存储的是帧的第一部分。若当前缓存中包含了帧结尾，则在RxDMA控制器写回帧接收状态的时候会将接收描述符0中的LDES位置位，以表明这个描述符中存储的是帧的最后一部分。通常当接收缓存大小大于接收帧的长度时，FDES位和LDES位会在同一个描述符中置位。当缓存接收到了帧结尾，或者当前描述符的缓存不足以存储整个帧时，RxDMA将获取下一个接收描述符，并将上一个描述符的接收描述符0的DAV位清零以关闭上个描述符。当LDES位置位时，描述符其他状态也会更新，并且ENET_DMA_STAT寄存器中的RS位将置位(当DINTC=0时立刻置位，当DINTC=1时延迟置位)。当接收到一个新的帧时，如果描述符的DAV位为'1'，则重复上述的RxDMA控制器操作。如果描述符的DAV位为'0'，则DMA控制器进入挂起状态，并设置ENET_DMA_STAT寄存器的RBU位为'1'。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。

### DMA 接收管理

RxDMA控制器的工作流程如下：

1. DMA接收描述符初始化，置接收描述符0的DAV位为'1'；
2. 将ENET_DMA_CTL寄存器的SRE位置为'1'，使能RxDMA控制器。DMA进入运行状态后，会从ENET_DMA_RDTADDR寄存器配置的描述符列表基地址获取接收描述符。如果获取

的描述符DAV位为1，则当前描述符开始接收帧。但如果检测到取到的描述符正在被CPU操作而不可用(DAV=0)，则DMA进入挂起状态，跳到步骤9；

3. 如果获取的描述符显示描述符由DMA占有(DAV=1)，那么该描述符的控制位和缓存地址就会被DMA所记录；
4. 处理接收到的帧，并从RxFIFO将数据传输到接收缓存；
5. 如果缓存被填满或者帧传输结束，接收控制器会从描述符队列中获取下一个接收描述符；
6. 如果当前帧传输结束，DMA操作跳到步骤7。如果当前帧传输没有结束(未接收到帧尾EOF)，则可能发生两种情况：
   - 下一个描述符的DAV位为'0'。如果接收帧清空功能使能，则RxDMA控制器将接收描述符0的描述符错误位DERR位置位。然后RxDMA控制器将当前描述符的DAV位清零以关闭描述符，并根据帧清空功能是否使能来确定是否置位LSG位(若使能则置位LSG，反之则不置位LSG)。之后DMA操作跳到步骤8；
   - 下一个描述符的DAV位为'1'。那么RxDMA将DAV位清零以关闭当前描述符，之后操作退回步骤4；
7. 如果使能了IEEE 1588时间戳功能，在接收帧完成后DMA控制器会把获取的时间戳的低位和高位(如果接收帧符合需要记录时间戳的帧的条件)，分别写入当前描述符的接收描述符2和接收描述符3(当DFM=0时)，或者写入当前描述符的接收描述符6和接收描述符7(当DFM=1时)。同时DMA把从MAC处返回的接收状态信息写入接收描述符0，并把DAV位清'0'，把LSG位置'1'；
8. 如果新获取的描述符DAV位为'1'，则RxDMA控制器操作跳动步骤4。如果DAV位为'0'，则RxDMA控制器进入挂起状态，并设ENET_DMA_STAT寄存器的RBU位为1。如果使能了接收帧清空功能，则在DMA进入挂起状态之前，控制器会清空接收帧；
9. 在挂起状态下，有两种方法退出该状态。一种方法是向接送查询使能寄存器ENET_DMA_RPEN中写入任意值。另一种方法是RxFIFO收到下一帧数据，这意味着在直通模式下时，帧数据字节数需要大于设置的阈值，或者在存储转发模式下，需要收到整个帧。当DMA退出暂停状态后，DMA会获取下一个描述符，并跳到步骤2。

### 获取接收描述符

只要满足下列条件任意一个或多个，DMA就会尝试获取接收描述符：

- 在寄存器ENET_DMA_CTL的接收开始/停止位SRE从'0'变为'1'，使DMA控制器进入运行状态的时候；
- 当前描述符的整个缓存大小(对于链结构为缓存1，对于环结构为缓存1和2)不足以接收整个帧，也就是说在接收到帧的结尾之前，当前描述符的缓存已满；
- 在一个完整的帧传送到接收缓存之后，并在当前描述符关闭之前；
- 在挂起状态时，MAC接收到新的帧；
- 向接送查询使能寄存器ENET_DMA_RPEN写入任意值。

### 挂起状态时接收到新的帧时的处理

在挂起状态时，当接收到一个新的帧，并且满足转发条件时(转发条件如上所述)，RxDMA将获

取帧的描述符。如果接收描述符0的DAV位为'1'，则RxDMA控制器退出挂起状态，返回运行状态开始接收帧。但当接收描述符0的DAV位为'0'，则应用程序可以通过配置ENET_DMA_CTL寄存器中DAFRF位来选择是否清空RxFIFO中的帧。如果DAFRF=0，则RxDMA控制器将丢弃FIFO所接收的帧数据，并将丢失帧计数器MSFC加1。若DAFRF=1，则可以阻止丢弃RxFIFO顶部的帧，除非RxFIFO满，丢失帧计数器MSFC的值不会增加。在DAV位为'0'时，ENET_DMA_STAT寄存器中的RBU位将被置位，RxDMA控制器仍处于挂起状态。

### 带 IEEE1588 时间戳的 RxDMA 描述符格式

如果使能了IEEE 1588功能，则MAC控制器会在带时间戳的帧接收完成之后，DMA关闭描述符之前(DAV位清'0')，将时间戳写入接收描述符2和接收描述符3(DFM=0)，或接收描述符6和接收描述符7(DFM=1)。

### 常规 RxDMA 描述符

常规RxDMA描述符结构体包含4个32位字，接收描述符0~接收描述符3。接收描述符0~接收描述符3的位定义如下：

**图 27-9. 常规接收描述符**




■ 接收描述符0

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DAV | DAFF | FRML[13:0] | | | | | | | | | | | | | |
| rw | rw | rw | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ERRS | DERR | SAFF | LERR | OERR | VTAG | FDES | LDES | IPHERR/ TSV | LCO | FRMT | RWDT | RERR | DBERR | CERR | PCERR/ EXSV |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | DAV | 描述符转发条件位<br>该位指示了DMA控制器是否可以占有该描述符。DMA会在帧完整接收或者描述符指向的缓存被填满以后把该位清'0'。<br>0：表示CPU占有描述符<br>1：表示DMA占有描述符 |

| | | |
|---|---|---|
| 30 | DAFF | 未通过目标地址过滤器位<br>0：接收帧通过目标地址过滤器<br>1：接收帧没有通过目标地址过滤器 |
| 29:16 | FRML | 帧长位<br>这些位表示了传送到输入缓存的接受帧字节长度(如果输入帧不是类型帧则包括CRC部分，如果输入帧是类型帧，则是否包括CRC部分由ENET_MAC_CFG寄存器的TFCD位决定)。只有LDES位为1且DERR位为0时，此位才有效。若LDES位和错误汇总位ERRS位都为'0'，则这些位表示当前接收帧已经传送到内存里的累计字节数。<br>**注意：**帧长度值为0意味着由于某种原因(例如FIFO溢出或在接收过程中动态地修改了过滤器的值，导致未通过过滤器的情况等)造成写入FIFO的帧不完整。 |
| 15 | ERRS | 错误汇总位<br>该位只在LDES位为'1'时有效。<br>当DFM = 0时，该位为下列位的逻辑"或":<br>DERR：描述符错误<br>OERR：溢出错误<br>LCO：延迟冲突<br>RWDT：看门狗超时<br>RERR：接收错误<br>CERR：CRC错误<br>IPHERR = 0, FRMT = 1且PCERR = 1：数据校验和错误<br>IPHERR = 1, FRMT = 1且PCERR = 0：报头校验和错误<br>IPHERR = 1, FRMT = 1且PCERR = 1：报头及数据校验和错误<br>当DFM = 1时，该位为下列位的逻辑"或":<br>IPPLDERR：IP帧数据错误<br>IPHERR：IP帧报头错误<br>DERR：描述符错误<br>OERR：溢出错误<br>LCO：延迟冲突<br>RWDT：看门狗超时<br>RERR：接收错误<br>CERR：CRC错误 |
| 14 | DERR | 描述符错误位<br>只有在LDES位置位时，此位才有效。<br>当当前描述符的缓存大小小于接收帧帧长同时DMA又无法占有下一个描述符，将发生描述符错误。<br>0：未发生描述符错误<br>1：发生了描述符错误 |
| 13 | SAFF | 未通过源地址过滤器位<br>0：未发生未通过源地址过滤器事件<br>1：帧没有通过源地址过滤器 |
| 12 | LERR | 长度错误位 |

| | | |
|---|---|---|
| | | 只有在 FRMT 位为'0'时，此位才有效。<br>该位指示了接收到的以太网帧头长度/类型域的值是否与接收帧的实际长度匹配。<br>0：未发生长度错误<br>1：发生了长度错误 |
| 11 | OERR | 溢出错误位<br>当RxFIFO发生了溢出，而接收帧已有部分被传送到输入缓存时，溢出错误位置位。<br>0：未发生溢出错误<br>1：发生了RxFIFO溢出，帧数据无效 |
| 10 | VTAG | VLAN标签位<br>0：接收帧非VLAN帧<br>1：当前的接收帧为VLAN帧 |
| 9 | FDES | 第一个描述符<br>该位表示当前描述符是否存放了接受帧的第一部分。<br>0：当前描述符未存放帧的第一部分<br>1：当前描述符存放了帧的第一部分 |
| 8 | LDES | 最后一个描述符<br>该位表示当前描述符是否存放了接受帧最后一部分。<br>0：当前描述符未存放帧的最后一部分<br>1：当前描述符存放了帧的最后一部分 |
| 7 | IPHERR/TSV | IP帧报头错误/时间戳有效位<br>当DFM=0时，则将位7结合位5和位0，查询错误状态表，判断发生了什么错误。<br>当DFM=1时，则表示已获取时间戳并将时间戳值写到了接收描述符6和接收描述符中，该位仅在LDES位置位时有效。 |
| 6 | LCO | 延迟冲突位<br>延迟冲突表示在接收到64字节数据后是否发生了冲突。<br>该位仅在半双工模式下有效 。<br>0：未发生延迟冲突<br>1：接收帧的过程发生了延迟冲突 |
| 5 | FRMT | 帧类型<br>当DFM=0时，则将位7结合位5和位0，查询错误状态表，判断发生了什么错误。<br>当DFM=1时，则表示接受帧为以太网类型帧或标签帧。<br>当接收帧为过短帧时，该位无效。<br>0：接收到的帧是IEEE802.3帧<br>1：接收到的帧是以太网类型的帧(以太网帧头长度/类型域的值大于等于0x0600，或是一个标签帧)。 |
| 4 | RWDT | 接收看门狗超时<br>当WDD=0，该位表示已接收到超过2048字节的帧数据。<br>当WDD=1，该位表示已接收到超过16384字节的帧数据。<br>0：未发生接收看门狗超时<br>1：在接收帧的过程中发生了看门狗超时，当前接收帧将被截断。 |

| | | |
|---|---|---|
| 3 | RERR | 接收错误<br>该位表示帧接收过程中，在RX_DV信号有效时是否收到有效接口信号RX_ER。<br>0：未发生接收错误<br>1：发生了接收错误 |
| 2 | DBERR | Dribble位错误<br>该位表示接收到的数据帧中存在半字节，只有 MII 模式时该位有效。<br>0：未发生Dribble位错误<br>1：发生了Dribble位错误 |
| 1 | CERR | CRC错误<br>该位表示接受帧的帧校验序列是否与硬件计算结果不匹配，只有在 LDES 位为'1'时该位有效。<br>0：未发生CRC错误<br>1：检测到接收帧发生了CRC错误 |
| 0 | PCERR/EXSV | 数据校验和错误/扩展状态有效位<br>当DFM=0时，则将位7结合位5和位0，查询错误状态表，判断发生了什么错误。<br>当DFM=1时，该位表示接收描述符4有效。<br>该位仅在LDES位置位时有效。<br>0：接收描述符4无效<br>1：接收描述符4有效 |

*表27-6. 接收描述符0错误状态描述，仅适用于常规描述符(DFM=0)*显示了接收描述符0中位IPHERR、FRMT、PCERR取值的含义：

**表 27-7. 接收描述符 0 错误状态描述，仅适用于常规描述符(DFM=0)**

| 位7: IPHERR | 位5: FRMT | 位0: PCERR | 帧状态 |
|---|---|---|---|
| 0 | 0 | 0 | IEEE802.3类型帧(长度域值小于0x0600或非标签帧) |
| 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 | 0 | IPv4/IPv6类型帧，未检测到校验和错误 |
| 0 | 1 | 1 | IPv4/IPv6类型帧，检测到数据校验和错误。<br>该错误可能原因：<br>1)计算的校验和值与接收帧中的校验和域值不匹配<br>2)接收帧长度域值与实际接收帧字节数不符 |
| 1 | 0 | 0 | 保留 |
| 1 | 0 | 1 | 类型帧(长度/类型域值大于等于0x600)或标签帧，但非IPv4/IPv6帧，不检测数据校验和 |
| 1 | 1 | 0 | IPv4/IPv6类型帧，检测到报头校验和错误。<br>该错误可能原因：<br>1) 类型域值与数据包中的版本值不一致<br>2) 计算的报头校验和值与报头校验和域值不匹配 |

| | | | |
|---|---|---|---|
| | | | 3)IP报头字节数不足 |
| 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 |

■ 接收描述符1

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DINTC | 保留 | | RB2S[12:0] | | | | | | | | | | | | |
| rw | | | rw | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RERM | RCHM | 保留 | RB1S[12:0] | | | | | | | | | | | | |
| Rw | rw | | rw | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | DINTC | 关闭接收完成中断<br>0：接收完成后ENET_DMA_STAT寄存器的RS位(位6)会立即置1，此时若使能了相应中断则会触发中断。<br>1：在接收帧成功接收完成后，ENET_DMA_STAT寄存器的RS位(位6)不会立即置1，但会在可配置的延时(由ENET_DMA_RSWDC寄存器配置)之后置位。 |
| 30:29 | 保留 | 必须保持复位值。 |
| 28:16 | RB2S[12:0] | 接收缓存2大小<br>表示接收缓存2的大小(以字节为单位)。缓存大小必须被设为4的倍数。这些位在RCHM位为'1'时被忽略。 |
| 15 | RERM | 接收描述符环形结构结尾<br>该位表示到达的是描述符列表中的最后一个描述符，下个描述符自动返回列表的基地址。<br>0：当前描述符不是最后一个描述符<br>1：到达描述符列表的最后一个描述符 |
| 14 | RCHM | 第二地址链表<br>0：描述符里的第二个地址指向第二缓存的地址<br>1：描述符里的第二个地址是下一个描述符的地址，该位仅在在链模式下使用。忽略RB2S [12:0]的值。<br>**注意：**当RERM=1，则即使该位置位，下个描述符也将返回列表基地址。 |
| 13 | 保留 | 必须保持复位值。 |
| 12:0 | RB1S[12:0] | 接收缓存1大小<br>表示接收缓存1的大小(以字节为单位)。缓存大小必须被设为4的倍数。 |

■ 接收描述符2

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RB1AP/RTSL[31:16] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RB1AP/RTSL[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|

| | | |
|---|---|---|
| 31:0 | RBAP1/RTSL[31:0] | 接收缓存1地址指针/发送帧时间戳低32位<br>这些位有2个功能：缓冲区1的地址指针(RB1AP)和时间戳低32位(RTSL)。<br>RB1AP：在DMA控制器获取该描述符之前，配置该位为缓存区1的地址。如果位RB1S不为'0'，则用该位的地址来存储接收的数据帧。对缓存的地址对齐不做限制。<br>RTSL：当使能了时间戳功能同时LDES位为'1'时，如果接收帧通过了地址过滤，并且置位了对应的帧类型使能位，则DMA会将时间戳低32位写入这些位。如果接收帧没有置位对应的帧类型使能位，则这些位会保持RB1AP的值。 |

■ 接收描述符3

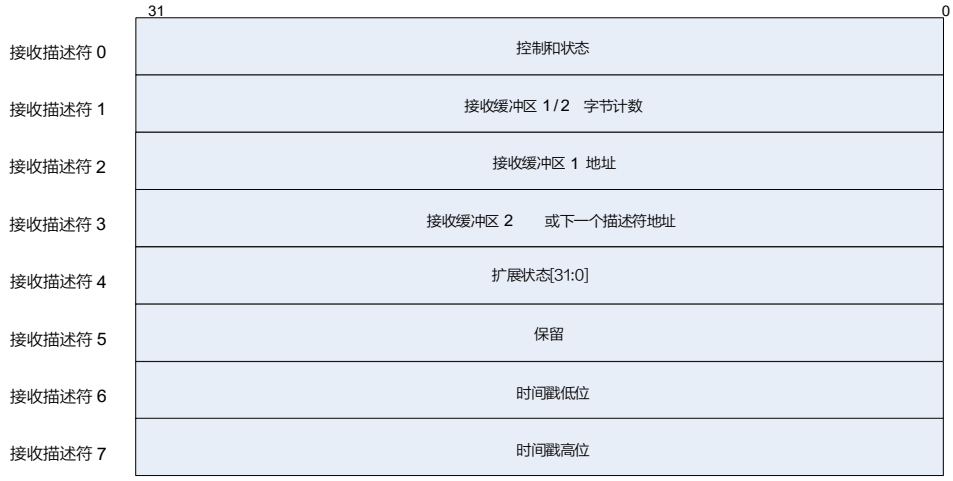
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RB2AP/RTSH[31:16] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RB2AP/RTSH[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | RB2AP/RTSH[31:0] | 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32位<br>这些位有2个功能：数据接收时这些位存放缓冲区2的地址或下个描述符的地址(RB2AP)，或帧时间戳高32位(RTSH)。<br>RB2AP：在DMA控制器获取该描述符之前，配置该位为缓存区2的地址(RCHM=0)或下个描述符地址(RCHM=1)。如果RCHM=1，且RERM=1，则忽略这些位。<br>如果该地址表示下个描述符地址，则地址需为字对齐的，没有其他的限制条件。<br>RTSH：当使能了时间戳功能同时帧的最后部分LDES位为'1'时，如果接收帧通过了地址过滤，并且置位了对应的帧类型使能位，则DMA会将时间戳高32位写入这些位。如果接收帧没有置位对应的帧类型使能位，则这些位会保持RB2AP的值。 |

### 增强 RxDMA 描述符

增强RxDMA描述符结构体包含8个32位字：接收描述符0~接收描述符7。接收描述符0~接收描述符3的位定义与常规RxDMA描述符相同，接收描述符4~接收描述符7的位定义如下：

**图 27-10. 增强接收描述符**




■ 接收描述符4

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | PTPVF | PTPOEF | PTPMT[3:0] | | | | IPF6 | IPF4 | IPCKSB | IPPLDERR | IPHERR | IPPLDT[2:0] | | |
| | | rw | rw | rw | | | | Rw | rw | rw | rw | rw | rw | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值。 |
| 13 | PTPVF | PTP 版本格式位<br>0：版本1格式<br>1：版本2格式 |
| 12 | PTPOEF | 以太网PTP帧<br>0：当PTPMT不为0时，接收的PTP帧是IP-UDP帧。<br>1：接收的PTP帧是IEEE802.3以太网帧 |
| 11:8 | PTPMT[3:0] | PTP消息类型<br>PTP消息类型解码为以下几种：<br>0x0：未收到PTP消息<br>0x1：SYNC<br>0x2：FOLLOW_UP<br>0x3：DELAY_REQ<br>0x4：DELAY_RESP<br>0x5：对于点对点透明时钟：PDELAY_REQ<br>对于普通时钟或边界时钟：ANNOUNCE<br>0x6：对于点对点透明时钟：PDELAY_RESP<br>对于普通时钟或边界时钟：MANAGEMENT<br>0x7：对于点对点透明时钟：PDELAY_RESP_FOLLOW_UP |

对于普通时钟或边界时钟：SIGNALING

| | | |
|---|---|---|
| 7 | IPF6 | IPv6帧位<br>0：接收帧不是IPv6帧<br>1：接收帧是IPv6帧 |
| 6 | IPF4 | IPv4帧位<br>0：接收帧不是IPv4帧<br>1：接收帧是IPv4帧 |
| 5 | IPCKSB | 绕过IP帧校验和<br>该位仅在接收帧为IPv6或IPv4帧时有效。<br>0：没有绕过接收帧校验和功能<br>1：绕过了接收帧校验和功能 |
| 4 | IPPLDERR | IP帧数据错误位<br>在以下任意情形中，该位会被置位：1)硬件计算的校验和域TCP，UDP或ICMP帧校验和域值不匹配 2)IP报头数据长度域值与接收到的帧数据长度不符。<br>0：在接收帧中没有发生帧数据错误<br>1：在接收帧中发生了帧数据错误 |
| 3 | IPHERR | IP报头错误<br>在以下任意情形中，该位会被置位：1)硬件计算的校验和值与IP报头校验和域值不匹配 2)以太网帧类型域值与IP数据包中版本域值不一致(例如类型域值为0x800，但版本域值不为0x4；类型域值为0x86dd，但版本域值不为0x6)。<br>0：没有发生IP报头错误<br>1：发生了IP报头错误 |
| 2:0 | IPPLDT[2:0] | IP帧数据类型位<br>仅在IPFCO=1，IPHERR=0且LDES=1时，这些位有效。<br>0x0：不支持的数据类型或忽略的IP数据类型<br>0x1：数据类型为UDP<br>0x2：数据类型为TCP<br>0x3：数据类型为ICMP<br>0x4~0x7：保留 |

- 接收描述符5
  所有位保留。
- 接收描述符6

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

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

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:0 | RTSL[31:0] | 接收时间戳低 32 位<br>当使能了时间戳功能同时帧的最后部分 LDES 位为'1'时，如果接收帧通过了地址过滤，并且置位了对应的帧类型使能位，则 DMA 会将时间戳低 32 位写入这些位。 |

■ 接收描述符7

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

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:0 | RTSH[31:0] | 接收时间戳高 32 位<br>当使能了时间戳功能同时帧的最后部分 LDES 位为'1'时，如果接收帧通过了地址过滤，并且置位了对应的帧类型使能位，则 DMA 会将时间戳高 32 位写入这些位。 |

### 27.3.4. MAC 统计计数器：MSC

为了了解发送和接收帧的统计情况，利用一组计数器来收集相关的统计数据。这些MAC计数器被称为MAC统计计数器(MSC)。在章节"*ENET寄存器*"中有这些寄存器的详细功能说明。

当发送帧没有出现帧下溢、没有载波、载波丢失、顺延(Deferral)过多、延迟冲突，过度冲突和jabber超时等情况时，可以称为"好帧"，MSC发送计数器会自动更新。

■ 当接收帧没有出现对齐错误、CRC错误、过短帧、长度错误、超出范围和MII_RX_ER引脚上的错误信号有效等情况时，可以称为"好帧"，MSC接收计数器会自动更新。其中，CRC错误是指CRC计算结果与帧校验序列值不一致，过短帧表示帧长少于64字节，长度错误表示长度域值与实际接收到的字节数不符，超出范围表示长度域值超过IEEE 802.3所规定的最大值，即对于非标签帧最大值为1518字节，对于VLAN标签帧最大值为1522字节。

**注意：**当被丢弃的帧是长度小于6字节的过短帧(没有完整接收到目标地址)时，MSC接收计数器也会更新。

### 27.3.5. 唤醒管理：WUM

以太网模块支持两种将系统从深度睡眠模式唤醒的方法。分别为远程唤醒帧和Magic Packet唤醒帧。为了减小功耗，可以使主机系统和以太网模块进入低功耗状态，从而可以停止由HCLK驱动的电路以及发送时钟。但由接收时钟驱动的电路将继续工作，以监听唤醒帧。如果将ENET_MAC_WUM寄存器的PWD位置1，则以太网模块进入低功耗状态。在低功耗状态下，MAC会丢弃所有的帧，直到退出低功耗状态。此时可以采用上述的两种方式能够退出低功耗状态。将ENET_MAC_WUM寄存器的WFEN置1，以设置当收到远程唤醒帧时唤醒以太网模块，

或将ENET_MAC_WUM寄存器的MPEN置1，以设置当收到Magic Packet唤醒帧时唤醒以太网模块。当任一个唤醒功能被使能，一旦MAC接收到相应的唤醒帧，以太网模块将产生一个唤醒中断，并退出低功耗状态。

### 远程唤醒帧检测

将ENET_MAC_WUM寄存器的WFEN置1可以使能远程唤醒检测。当MAC处于低功耗状态，且远程唤醒使能位为'1'时，MAC会进行唤醒帧过滤。如果输入帧通过了过滤器命令的地址过滤，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，随后MAC即恢复正常工作。即便唤醒帧的长度超过了512字节，只要该帧有正确的CRC值，它仍然被认为是有效的。在接收到远程唤醒帧时还会将ENET_MAC_WUM寄存器的WUFR位置1。如果远程唤醒中断没有被屏蔽，那么此时还将产生一个WUM中断。

### Magic Packet 检测

另一种唤醒方法是检测Magic Packet唤醒帧(见AMD公司的"Magic Packet技术")。一个Magic Packet帧是一种特殊构成的数据包，专门用于唤醒。这种包可以被以太网模块接收、分析和识别，并用于触发一个唤醒事件。设置ENET_MAC_WUM寄存器的MPEN位为'1'可以使能此功能。这种类型的帧格式如下：目的和源地址域之后的任何位置连续6字节全1(0xFFFF FFFF FFFF)，接着是在没有任何中断和暂停的情况下有16个重复的MAC地址；如果这16次重复间有任何的间断，则需要重新在输入帧里检测0xFFFF FFFF FFFF。WUM模块会持续监视每一个发向本节点的帧，那些通过地址过滤的Magic Packet帧，MAC会进一步检测其是否符合Magic Packet的格式，一旦通过检测将会使MAC从低功耗状态下唤醒。设备也接受多播帧作为Magic Packet帧。

下面是一个站地址为0xAABB CCDD EEFF的Magic Packet帧实例(MISC表示包内各种附加的数据字节)：

<DESTINATION><SOURCE><MISC>

……FF FF FF FF FF FF

AABB CCDD EEFF AABB CCDD EEFF AABB CCDD EEFF AABB CCDD EEFF

AABB CCDD EEFF AABB CCDD EEFF AABB CCDD EEFF AABB CCDD EEFF

AABB CCDD EEFF AABB CCDD EEFF AABB CCDD EEFF AABB CCDD EEFF

AABB CCDD EEFF AABB CCDD EEFF AABB CCDD EEFF AABB CCDD EEFF

<MISC><FCS>

一旦检测到Magic Packet帧，ENET_MAC_WUM寄存器的位MPKR会被置1。如果使能了Magic Packet中断，此时还将产生对应中断。

### 系统在低功耗期间注意事项

在MCU处于深度睡眠模式时，若使能外部中断线19，则以太网的WUM模块仍能够检测帧。 由

于MAC在低功耗状态也需要进行Magic Packet/远程唤醒帧检测，因此ENET_MAC_CFG寄存器的REN位必须保持为'1'。在低功耗状态时需要把ENET_MAC_CFG寄存器的TEN位清'0'来关闭发送功能。此外，由于不需要把Magic Packet/远程唤醒帧转发给应用，因此在低功耗状态时也要关闭以太网DMA模块，可以通过设置ENET_DMA_CTL寄存器的STE位和SRE位(分别对应TxDMA和RxDMA)为'0'来关闭以太网DMA。

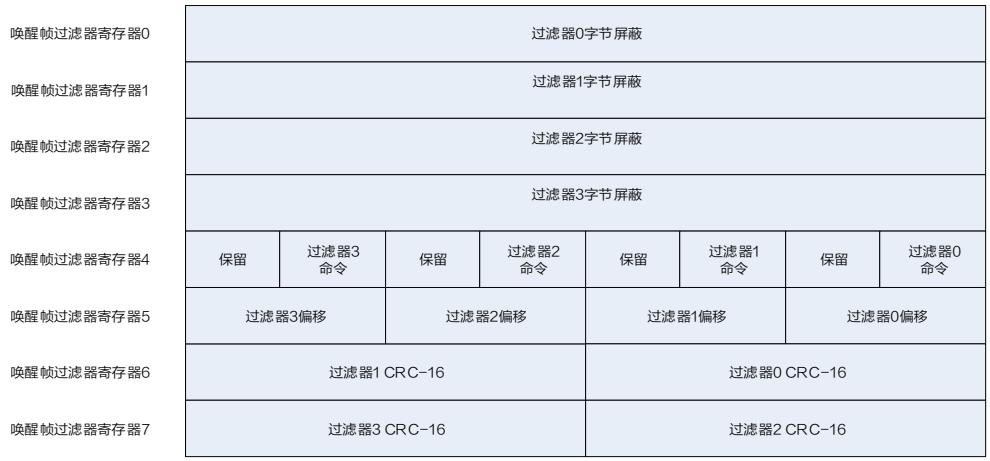
推荐的进入低功耗状态和唤醒步骤如下：

1. 等待当前帧发送完毕，然后将ENET_DMA_CTL寄存器的STE位复位来关闭TxDMA；
2. 把ENET_MAC_CFG寄存器的TEN位和REN位清'0'，来关闭MAC发射器和MAC接收器；
3. 观察ENET_DMA_STAT寄存器位RS，等待RxDMA把RxFIFO里的所有帧读出，再关闭RxDMA；
4. 配置并使能外部中断线19，使其能产生事件或者中断。如果配置了外部中断线19产生中断，则还需要编写中断处理程序ENET_WKUP_IRQ，在其中清除外部中断线19的中断标志位；
5. 设置ENET_MAC_WUM寄存器的MPEN或WFEN位(或两位)为'1'，使能Magic Packet/远程唤醒帧检测(或两种功能)；
6. 设置ENET_MAC_WUM寄存器的PWD位为'1'，使能低功耗模式；
7. 设置ENET_MAC_CFG寄存器的REN位为'1'，打开MAC接收器；
8. 设置使MCU进入深度睡眠模式；
9. 在接收到有效的唤醒帧后，以太网模块退出低功耗状态；
10. 读取ENET_MAC_WUM寄存器来清除电源管理事件标志位，打开MAC发送器，以及TxDMA和RxDMA；
11. 设置系统时钟：使能HXTAL并配置RCU时钟参数。

### 远程唤醒帧过滤器寄存器

唤醒帧过滤器寄存器一共有8个，但这些寄存器共用一个相同的偏移地址。在完成对某个过滤器寄存器的读或写的时候，内部的指针会自动指到下一个过滤器寄存器。不论是读还是写操作，强烈建议连续8次的操作。也就是说，对其设置时需要将设置的值分为8次逐一写入唤醒帧过滤器寄存器地址，读取的时候也是需要连续读8次唤醒帧过滤器寄存器，才能将所有值读出。

**图 27-11. 唤醒帧过滤器寄存器**




■ 过滤器n字节屏蔽

该寄存器定义了过滤器n(n=0，1，2，3)使用帧的哪些字节来检查判断是否为唤醒帧。其第31位必须为'0',位[30:0]是字节屏蔽位。如果过滤器n(n=0，1，2，3)的第m位(m=0~30)为'1'，则唤醒帧检测的CRC模块会处理输入帧的第[过滤器n偏移+m]字节，否则忽略之。

■ 过滤器n命令

共4位控制过滤器n的工作模式。最高位位3为地址类型选择，如果该位为'1'，则只检测多播帧；如果该位为'0'，则只检测单播帧。位2和位1必须保持为0。位0是过滤器n的使能位，置位时使能过滤器n，反之禁能过滤器n。

■ 过滤器n偏移

与过滤器n字节屏蔽配合使用。该寄存器定义了过滤器n要检查的首字节在帧内的偏移量。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1个字节)。

■ 过滤器n CRC-16

该寄存器包含了预先写入的CRC-16码，用于与帧数据计算的CRC-16值进行比较。

## 27.3.6. 精确时间协议：PTP

协议的大部分是通过UDP层之上的应用程序软件实现的。MAC的PTP模块主要是支持记录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。

关于精确时间协议(PTP)的具体内容可参见IEEE 1588™相关文档。

### 基准时钟源

IEEE 1588协议规定，通过一个64位寄存器来获得系统基准时间，其中高32位提供秒级的时间信息，低32位提供亚秒级的时间信息。

PTP基准时钟输入用来生成系统基准时间(也称为系统时间)，以及获取PTP帧的时间戳值。其

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0.1us左右。

### 同步精度

时间同步的精度取决于以下几个因素：

- PTP基准时钟输入的频率；
- 所用晶体振荡器的特性(频漂)；
- 同步流程的执行频度。

### 系统时间校准

64位PTP系统时间由PTP输入基准时钟来更新。这个PTP系统时间用来作为记录发送/接收时间戳的依据。该系统时间的初始化和校准支持两种模式：粗调和精调。校准的目的是纠正频率偏移。

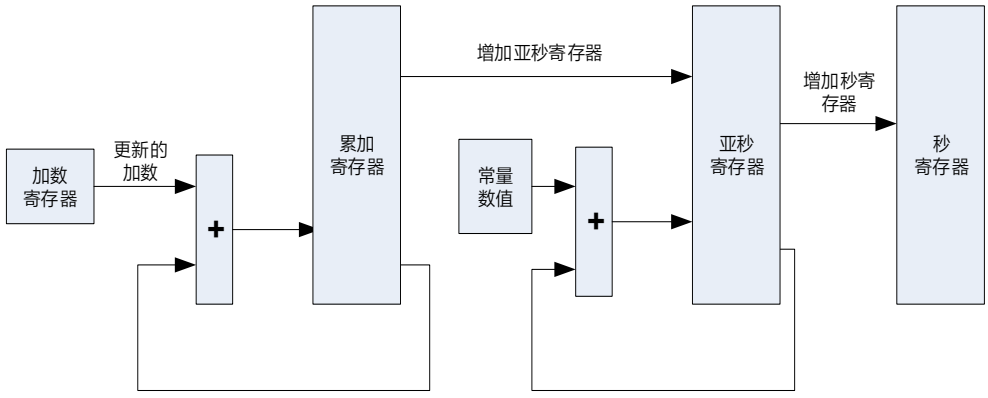
若选择了粗调的方式，则可通过配置PTP时间戳更新寄存器(ENET_PTP_TSUH和ENET_PTP_TSUL)，来进行系统时间初始化和校准。如果TMSSTI位被置位，则PTP时间戳更新寄存器被用于初始化。如果TMSSTU位被置位，则PTP时间戳更新寄存器被用于系统时间的调整，加上或者减去这个寄存器值进行校准。

若选择了精调的方式，则需要一段时间才能完成。由应用程序确定精调的频率，以确保从时钟能线性地同步于主时钟，避免不可预知的大的抖动。

这种方法是指，在每个HCLK周期把加数寄存器ENET_PTP_TSADDEND中的值加入累加器。当累加器溢出时会产生脉冲令时间戳低寄存器ENET_PTP_TSL的值增加。增加的值由亚秒递增寄存器ENET_PTP_SSINC中的值决定。

*图27-7. 系统时钟精细校准方法*演示了精调算法的流程：

**图 27-12. 系统时钟精细校准方法**




下面是一个具体例子用于说明精调方式如何更新系统时间：

假设系统时钟更新电路的精度需要达到25ns，即更新的频率为40MHz。假设基准时钟HCLK是72MHz，计算频率比得72 / 40 = 1.8。写入ENET_PTP_TSADDEND寄存器的值应当是$2^{32}$ / 1.8，

等于0x8E38 E38E。如果基准时钟频率漂低，假设降至68MHz，此时频率比变成68/40 = 1.7，写入ENET_PTP_TSADDEND寄存器的值应当是$2^{32}$ / 1.7 = 0x9696 9697。如果基准时钟漂高，假设升到76MHz，写入到ENET_PTP_TSADDEND寄存器的值应当是$2^{32}$ / 1.9 = 0x86BC A1AF。。初始时，将加数寄存器设为从时钟频率Clock Addend Value(0)，该值按上述进行计算。除了配置加数计数器之外，还需对亚秒递增寄存器进行设置才能保证达到20ns的精度。每次累加寄存器溢出后，该寄存器的值会对时间戳低寄存器进行更新。因为ENET_PTP_TSL寄存器中的STMSS[30:0]位表示系统时间的亚秒值，其精度为$10^9$ns / $2^{31}$ = 0.46ns。所以为了使系统时间精度达到25ns，亚秒递增寄存器的值应该设为25 / 0.46 = 0d54。

**注意：**下文描述的算法是以主从设备之间传输的时延Master-to-Slave-Delay恒定为基础的，通过该算法在若干个Sync周期内确定同步频率比。

算法如下：

- 定义主设备发送一个SYNC消息到从设备时的时间：MSYNCT(n)
  定义从设备的本地时间：SLOCALT(n)
  定义主设备的本地时间：MLOCALT(n)
  计算：MLOCALT(n)= MSYNCT(n)+ Master-to-Slave-Delay(n)

- 定义发送两次SYNC消息之间的主设备时钟计数：MCLOCKC(n)
  计算：MCLOCKC(n)= MLOCALT(n)- MLOCALT(n-1)
  定义接收两次SYNC消息之间的从设备时钟计数：SCLOCKC(n)
  计算：SCLOCKC(n)= SLOCALT(n)- SLOCALT(n-1)

- 定义两个计算之间的差值：DIFFCC(n)
  计算：DIFFCC(n)= MCLOCKC(n)- SCLOCKC(n)

- 定义从时钟频率调整系数：SCFAF(n)
  计算：SCFAF(n)=(MCLOCKC(n)+ DIFFCC(n))/ SCLOCKC(n)

- 定义加数寄存器的时钟加数值：Clock Addend Value(n)
  Clock Addend Value(n)= SCFAF(n)* Clock Addend Value(n-1)

**注意：**实际操作中，可能需要多个SYNC消息来完成主从设备的同步。

### 系统时间初始化流程

设置ENET_PTP_TSCTL寄存器的位TMSEN为'1'，可以使能时间戳功能。不过在把该位置'1'以后，必须首先初始化时间戳计数器来开始时间戳操作。初始化步骤如下：

1. 置位ENET_MAC_INTMSK寄存器的TMSTIM位，以屏蔽时间戳触发中断；
2. 置位ENET_PTP_TSCTL寄存器TMSEN位，以使能时间戳；
3. 根据期望时钟精度配置亚秒递增寄存器；
4. 若希望采用精调校准方式，则配置时间戳加数寄存器，并置位ENET_PTP_TSCTL寄存器的TMSARU位。若希望采用粗调校准方式，则忽略第4-6步，直接跳至第7步；
5. 轮询ENET_PTP_TSCTL寄存器的位TMSARU，直到其变为'0'；
6. 将ENET_PTP_TSCTL寄存器的TMSFCU位置位，来选择使用精调校准方式；
7. 把希望设置的系统时间值写入时间戳更新高寄存器和时间戳更新低寄存器；
8. 置位ENET_PTP_TSCTL寄存器的TMSSTI位，以初始化时间戳；

9. 一旦初始化成功后，时间戳计数器就开始工作。

### 系统时间更新步骤

#### 粗调方式

1. 在时间戳更新高寄存器和时间戳更新低寄存器中写入偏移值(可以是负值)；
2. 置位ENET_PTP_TSCTL寄存器的TMSSTU位，以更新时间戳寄存器；
3. 轮询TMSSTU位，直到其被清'0'后完成。

#### 精调方式

1. 利用前述"*系统时间校准方法*"介绍的算法，计算出期望的系统时钟频率所对应的加数寄存器的值；
2. 将值写入加数计数器，并设置ENET_PTP_TSCTL寄存器的TMSARU位为'1'将该值更新到PTP模块；
3. 把要求的期望时间写入期望时间高和期望时间低寄存器，并设置ENET_MAC_INTMSK寄存器的TMSTIM位为'0'来允许时间戳中断；
4. 设置ENET_PTP_TSCTL寄存器的TMSITEN位为'1'使能时间戳中断；
5. 在这个事件产生中断时，读出ENET_MAC_INTF寄存器的值以清除相应的中断标志位；
6. 重新用旧值编写时间戳加数寄存器，并设置ENET_PTP_TSCTL寄存器的TMSARU位为'1'将值更新到PTP模块。

### 带 PTP 功能的帧的发送与接收

在使能了IEEE 1588(PTP)时间戳功能后，在发送帧的帧首界定码从MAC输出或者MAC接收到帧的帧首界定码的时候，时间戳值被记录。每一个等待发送的帧在DMA发送描述符中都有一个标志，指示是否需要记录这个帧的时间戳，这与发送的帧是否为PTP帧无关。若ENET_PTP_TSCTL寄存器的位ARFSEN为'1'，则所有接收到的帧的时间戳都将被记录。如果ARFSEN位为'0'，则通过地址过滤的接收帧需要与ENET_PTP_TSCTL寄存器的配置进行匹配。换句话说，只有与PTP配置相匹配的帧才标记为一个PTP帧，并且同时记录时间戳值到描述符中。鉴别接收帧是否为PTP帧，接收帧中的PTP版本需要与PFSV位相符，并且对应的帧类型使能位(寄存器ENET_PTP_TSCTL中的位13到位11)要置位。特殊地，对于非IP数据包的PTP帧(Ethernet II型封装的 PTP数据帧)，以太网帧的目标地址域需要为特殊的MAC地址(比如对于PDELAY_REQ/PDELAY_RESP/PDELAY_RESP_FOLLOW_UP消息类型，目标地址应为0x0E00 00C2 8001，对于其他的消息类型，目标地址应为0x0000 0019 1B01，详细信息请参考IEEE1588-2008规范)。同时，为了增强灵活性，如果MAFEN位置位，除了上述两个特殊地址以外，MAC1-3寄存器中SAF位为复位状态的地址也可以认为是特殊地址。

记录下来的时间戳会和帧的发送/接收状态信息一起，存放在相应的发送/接收描述符里。64位的发送帧时间戳写入DMA发送描述符，64位的接受帧时间戳写入DMA接收描述符。具体描述见"*带IEEE 1588时间戳的TxDMA描述符格式*"和"*带IEEE1588时间戳的RxDMA描述符格式*"。

#### 内部连接触发

MAC可以在系统时间大于等于期望时间的时候提供触发中断。使用中断会引入一段已知的中断时延再加上不确定的命令执行时间。为了计算这部分已知的中断时延时间，在系统时间大于期望值的时候，PTP会将一个输出信号置高。将TIMER1_IRMP寄存器的位域[11:10]设为0x1，可将此输出信号内部连接到TIMER1的ITI1输入上。利用这个信号，由于TIMER1的时钟与PTP基准时钟(HCLK)是同步的，因此不再有任何不确定的误差。

#### PPS 输出信号

将ENET_PPS_OUT映射到AF11功能上，可以使能PPS输出功能。该功能可以输出脉冲宽度为默认125ms(其他宽度详见*寄存器设置*)的脉冲，用于检查网络全部节点之间的同步。为了测试本地从时钟和主时钟之间的差别，可以把主从设备的PPS(秒脉冲)输出都连接到示波器，以测量2个时钟之间的差别。

### 27.3.7. 典型的以太网配置流程示例

在上电复位或系统复位之后，应用程序可按以下的典型操作流程来配置并启动以太网模块：

- 使能以太网时钟：
  配置RCU模块来使能HCLK时钟和以太网发送/接收时钟。
- 配置通讯接口：
  配置SYSCFG模块，选择接口模式(MII或RMII)；
  配置GPIO模块，将相应的功能脚映射到复用功能11(AF11)上。
- 等待复位完成：
  轮询 ENET_DMA_BCTL 寄存器直到 SWR 位复位(SWR 位在上电复位后或系统复位后默认置位)。
- 获取并配置PHY寄存器参数：
  根据 HCLK 频率，配置 SMI 时钟频率，并访问 PHY 寄存器获取 PHY 的信息(例如是否支持半/全双工，是否支持 10M/100Mbit 速度等等)。根据外部 PHY 支持的模式，配置 ENET_MAC_CFG 寄存器使与 PHY 寄存器信息一致。
- 初始化以太网DMA模块用于数据传输：
  配置 ENET_DMA_BCTL，ENET_DMA_RDTADDR，ENET_DMA_TDTADDR 和 ENET_DMA_CTL 寄存器，完成 DMA 模块初始化(详细信息请参考 *DMA 控制器描述*章节)。
- 初始化用于存放描述符列表以及数据缓存的物理内存空间：
  根据 ENET_DMA_RDTADDR 和 ENET_DMA_TDTADDR 寄存器中的地址，初始化发送和接收描述符(DAV=1)，以及数据缓存。
- 使能MAC和DMA模块，开始发送和接收：
  置位 ENET_MAC_CFG 寄存器中的 TEN 和 REN 位，开启 MAC 发送器和接收器。置位 ENET_DMA_CTL 寄存器中的 STE 位和 SRE 位，使能 DMA 的发送和接收。
- 如果有帧要发送：

  1. 选择一个或多个描述符发送描述符，将发送帧数据写到发送描述符中指定的缓存地

址中；
2. 将这些发送帧描述符中的 DAV 位置位；
3. 写入任意值到 ENET_DMA_TPEN 寄存器中，使 TxDMA 退出挂起模式，开始发送数据；
4. 有两种方法来确定当前帧是否发送完毕。第一种方法为轮询当前描述符的 DAV 位直到其复位；第二种方法仅适用于当 INTC 位为 1 的情况，应用程序可以轮询 ENET_DMA_STAT 寄存器的 TS 位直到其置位。

■ 如果有帧要接收：
1. 查看描述符列表中的第一个接收描述符(其地址在 ENET_DMA_RDTADDR 寄存器中配置)；
2. 如果接收描述符 0 的 DAV 位复位，则说明描述符已被使用过，且接收缓存空间已存储了接收帧；
3. 处理接收帧数据；
4. 置位当前描述符的 DAV 位，以复用当前描述符接收新的帧；
5. 查看列表中的下一个描述符，跳到步骤 2。

### 27.3.8. 以太网中断

以太网部分一共有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EXTI线19的以太网唤醒事件(检测唤醒帧或者Magic Packet)。

第一个中断向量用于由MAC和DMA产生的中断。关于***MAC中断***和***DMA中断***的详细介绍请看下文。

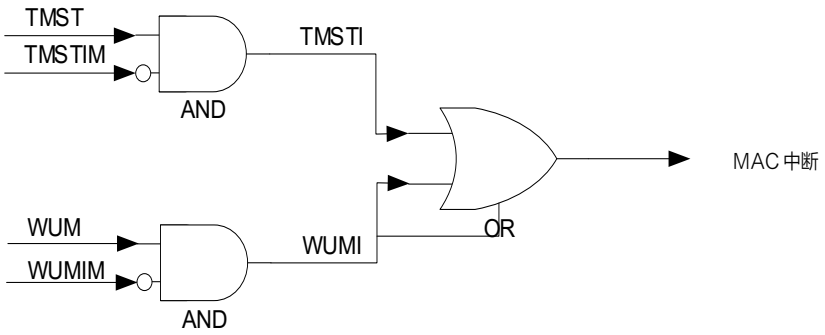
第二个中断向量用于WUM模块在唤醒事件时产生的中断，唤醒事件为远程唤醒帧接收事件或Magic Packet唤醒帧接收事件。唤醒事件映射到EXTI线19上，若使能了EXTI线19的上升沿中断，则唤醒事件可以使微控制器退出深度睡眠模式。此外，若使能了WUM中断，则EXTI线19中断和以太网中断都会被触发。

**注意：**由于WUM寄存器位于RX_CLK域，在应用程序读WUM寄存器后，到这些标志位被清除可能会有可观的延迟(延迟由HCLK和RX_CLK时钟频率之间差异决定)。为避免两次进入同一个中断，强烈建议应用程序在中断里等待唤醒帧接收标志位WUFR和Magic Packet接收标志位MPKR变为0后，再退出中断服务程序。

#### MAC 中断

MAC控制器有多个中断触发源。ENET_MAC_INTF寄存器描述了所有可产生的MAC中断类型，每个位都有对应的中断屏蔽位来防止某一事件引发中断。MAC中断信号为MAC所有中断的逻辑或。

图 27-13. MAC 控制器中断示意图



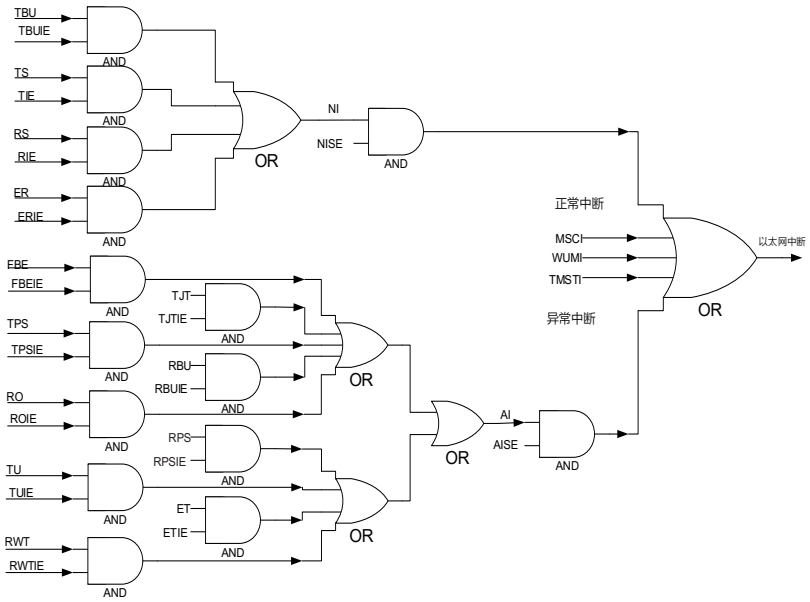

## DMA 中断

DMA控制器有两种类型中断事件：正常类和异常类。

无论什么类型的中断事件，都具有相应的中断使能位(屏蔽位)来控制是否产生中断。当所有中断事件都被清除，或中断使能位被清除，则相应的中断汇总位也被清除。如果正常类和异常类中断都被清除，则DMA中断将被清除。

*图27-14. 以太网中断示意图*示意了以太网模块的中断连接：

图 27-14. 以太网中断示意图



## 27.4. ENET 寄存器

ENET 基地址：0x4002 8000
支持字节，半字和字访问方式。

### 27.4.1. MAC 配置寄存器(ENET_MAC_CFG)

地址偏移：0x0000
复位值：0x0000 8000

MAC配置寄存器是MAC的工作模式寄存器。它定义了接收和发送的工作模式。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | TFCD | 保留 | WDD | JBD | 保留 | | IGBS[2:0] | | | CSD |
| | | | | | | rw | | rw | rw | | | rw | | | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | SPD | ROD | LBM | DPM | IPFCO | RTD | 保留 | APCD | BOL[1:0] | | DFC | TEN | REN | 保留 | |
| | rw | rw | rw | rw | rw | rw | | rw | rw | | rw | rw | rw | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:26 | 保留 | 必须保持复位值。 |
| 25 | TFCD | 类型帧CRC剥离位<br>0：帧的帧校验序列(最后4字节)在转发之前不会被剥离<br>1：帧的帧校验序列(最后4字节)在转发之前会被剥离<br>**注意：**该位仅在帧的LT域大于0x0600时有效。 |
| 24 | 保留 | 必须保持复位值。 |
| 23 | WDD | 关闭看门狗<br>该位表示已接收到了最大字节数的数据，超过的部分将被切断。<br>0：MAC允许接收小于或等于2048字节的帧<br>1：MAC关闭接收看门狗定时器，此时最多可接收16384字节的帧。 |
| 22 | JBD | 不检测Jabber<br>该位表示发送帧最大允许的发送字节数，超过的部分将被截断。<br>0：MAC允许的最大发送字节数为2048字节<br>1：MAC关闭发送Jabber定时器，此时最多可发送16384字节的帧。 |
| 21:20 | 保留 | 必须保持复位值。 |
| 19:17 | IGBS[2:0] | 帧间间隙选择位<br>这些位用于选择2个相邻发送帧之间的最短发送间隙，100Mbps时发送时钟为25M，10Mbps时发送时钟为2.5M。<br>0x0：96个发送时钟周期<br>0x1：88个发送时钟周期 |

| | | |
|---|---|---|
| | | 0x2：80个发送时钟周期<br>0x3：72个发送时钟周期<br>0x4：64个发送时钟周期<br>0x5：56个发送时钟周期(半双工模式下不可用)<br>0x6：48个发送时钟周期(半双工模式下不可用)<br>0x7：40个发送时钟周期(半双工模式下不可用) |
| 16 | CSD | 关闭载波侦听功能<br>0：MAC载波信号错误时会报错，并终止发送。<br>1：在半双工模式下，MAC在发送帧过程中忽略MII的CRS信号，发送过程中载波丢失或者没有载波都不会报错。 |
| 15 | 保留 | 必须保持复位值。 |
| 14 | SPD | 快速以太网速度<br>该位表示快速以太网模式下的速度：<br>0：10 Mbit/s<br>1：100 Mbit/s |
| 13 | ROD | 关闭自接收功能<br>该位在全双工模式下可忽略<br>0：MAC在发送时接收所有来自PHY的数据包<br>1：MAC在半双工模式下不接受帧 |
| 12 | LBM | 回环模式<br>0：MAC在普通模式下工作<br>1：MAC在MII的回环模式下工作 |
| 11 | DPM | 双工模式<br>0：半双工模式使能<br>1：全双工模式使能 |
| 10 | IPFCO | IP帧数据校验和<br>0：禁止接收端TCP/UDP/ICMP报头的校验和检验功能<br>1：使能接收端的帧数据校验和检测功能 |
| 9 | RTD | 不尝试重试<br>全双工模式下该位可被忽略<br>0：MAC会在发生冲突后按照BOL位的设定重发高达16次<br>1：帧仅发送一次 |
| 8 | 保留 | 必须保持复位值。 |
| 7 | APCD | 自动填充/CRC剥离<br>该位仅在非标签帧，且其长度域值小于等于1536时有效。<br>0：MAC会转发所有接收到的帧，而不改变帧的内容。<br>1：MAC会去除帧的填充字节和CRC域 |
| 6:5 | BOL[1:0] | 退后限制<br>在全双工模式下这些位可被忽略 |

| | | |
|---|---|---|
| | | 在发生冲突后，MAC在重发当前帧之前需要延迟一段时间。这个延迟时间(dt)的时基单元称为时间间隙，一个时间间隙为512位时间。这个延迟时间(dt)是由下式计算得的随机整数值：0<=dt <$2^k$。<br>0x0：k = min (n, 10)<br>0x1：k = min (n, 8)<br>0x2：k = min (n, 4)<br>0x3：k = min (n, 1)<br>其中n = 重发次数。 |
| 4 | DFC | 顺延检验<br>在全双工模式下这些位可被忽略<br>0：禁止MAC顺延检验功能。MAC会延迟发送直到CRS信号失效。<br>1：MAC顺延检验功能使能。如果延迟超过24288位时间，则会发生过度顺延错误，并且MAC将中止发送。但如果在顺延时间内检测到有效的CRS(载波侦听)信号，则会将顺延计数器重置为0，重新启动顺延计时。 |
| 3 | TEN | 使能发送器<br>0：MAC关闭发送状态机，若当前帧正在发送则在完成发送后关闭。<br>1：MAC使能发送状态机 |
| 2 | REN | 使能接收器<br>0：MAC关闭接收状态机，若当前帧正在接收则在接收完成后关闭。<br>1：MAC使能接收状态机 |
| 1:0 | 保留 | 必须保持复位值。 |

### 27.4.2. MAC 帧过滤器寄存器(ENET_MAC_FRMF)

地址偏移：0x0004
复位值：0x0000 0000

MAC帧过滤器寄存器包含了接收帧的过滤模式位。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FAR | 保留 | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 保留 | | | | | HPFLT | SAFLT | SAIFLT | PCFRM[1:0] | | BFRMD | MFD | DAIFLT | HMF | HUF | PM |
| | | | | | rw | rw | rw | rw | | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | FAR | 接收所有帧<br>该位控制帧过滤器功能。<br>0：只有通过了地址过滤器的接收帧才会被转发给应用程序<br>1：所有接收到的帧都会被转发给应用程序，但过滤的结果会反映在更新接收描述符状态信息的相应标志位。 |

| | | |
|---|---|---|
| 30:11 | 保留 | 必须保持复位值。 |
| 10 | HPFLT | HASH或者完美过滤<br>0：如果HMF位或者HUF位置'1'，符合HASH过滤器的帧才能通过接收地址过滤。<br>1：如果HMF位或者HUF位置'1'，接收帧通过HASH过滤器或者完美过滤器中任一种，就认为通过接收地址过滤。 |
| 9 | SAFLT | 源地址过滤器<br>除了目标地址过滤之外，使能源地址过滤器。<br>过滤器将接收帧的源地址域值与使能的源地址寄存器中配置的值进行比较。如果源地址值相匹配，则接收描述符中的源地址匹配状态位将置位。<br>0：源地址过滤器关闭<br>1：源地址过滤器使能 |
| 8 | SAIFLT | 源地址过滤结果逆转<br>该位将源地址比较结果逆转。<br>0：仅在源地址过滤器结果逆转<br>1：使能源地址过滤器结果逆转，所有源地址符合源地址寄存器的帧会被标记为未通过源地址过滤。 |
| 7:6 | PCFRM[1:0] | 控制帧转发位<br>这些位用于设置所有控制帧的转发条件(包括单播和多播暂停帧)。<br>对于是否处理暂停控制帧，只取决于RFCEN位(ENET_MAC_FCTL[2])的值。<br>0x0：MAC不转发任何控制帧给应用程序<br>0x1：MAC转发除了暂停帧以外的其他控制帧给应用程序<br>0x2：MAC转发所有的控制帧给应用程序，即使是没通过地址过滤器的控制帧。<br>0x3：MAC转发通过地址过滤器的控制帧给应用程序 |
| 5 | BFRMD | 不接收广播帧<br>0：过滤器接收所有广播帧<br>1：过滤器不接收所有广播帧 |
| 4 | MFD | 关闭多播过滤器<br>0：是否对多播帧进行过滤，取决于HMF位的取值。<br>1：所有的带多播目标地址的帧(帧的目标地址域中第一位为'1'，但不是所有位都为'1')都能通过过滤器。 |
| 3 | DAIFLT | 目标地址过滤结果逆转<br>该位将目标地址过滤结果逆转。<br>0：禁用目标地址过滤结果逆转<br>1：使能目标地址过滤结果逆转 |
| 2 | HMF | 多播HASH过滤器<br>0：MAC会将接收到的多播帧的目标地址域和目标地址寄存器的设定值比较<br>1：MAC根据HASH列表对接收到的多播帧进行目标地址过滤 |
| 1 | HUF | 单播HASH过滤器<br>0：MAC会将接收到的单播帧目标地址域和目标地址寄存器的设定值比较<br>1：MAC根据HASH列表对接收到的单播帧进行目标地址过滤 |
| 0 | PM | 混杂模式<br>该位使地址过滤器无效，这意味着所有帧均可通过过滤器，同时接收描述符中状态信息的目标地址/源地址错误位总是为'0'。 |

0：禁用混杂模式
1：使能混杂模式

### 27.4.3. MAC hash 列表高寄存器(ENET_MAC_HLH)

地址偏移：0x0008
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | HLH[31:0] | HASH列表高位<br>这些位是HASH列表的高32位。 |

### 27.4.4. MAC hash 列表低寄存器(ENET_MAC_HLL)

地址偏移：0x000C
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | HLL[31:0] | HASH列表低位<br>这些位是HASH列表的低32位。 |

### 27.4.5. MAC PHY 控制寄存器(ENET_MAC_PHY_CTL)

地址偏移：0x0010
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 保留 |
|---|

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PA[4:0] | | | | | PR[4:0] | | | | | 保留 | CLR[2:0] | | | PW | PB |
| rw | | | | | rw | | | | | | rw | | | rw | rc_w1 |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:11 | PA[4:0] | PHY地址<br>这些位选择想要访问的PHY地址。 |
| 10:6 | PR[4:0] | PHY寄存器<br>这些位选择想要访问的PHY寄存器。 |
| 5 | 保留 | 必须保持复位值。 |
| 4:2 | CLR[2:0] | 时钟范围<br>根据HCLK的频率来决定MDC的时钟分频系数。<br>0x0：HCLK/42(HCLK范围：60-100 MHz)<br>0x1：HCLK/62(HCLK范围：100-150 MHz)<br>0x2：HCLK/16(HCLK范围：20-35 MHz)<br>0x3：HCLK/26(HCLK范围：35-60 MHz)<br>0x4：HCLK/102(HCLK范围：150-240 MHz)<br>其他：保留。 |
| 1 | PW | PHY写<br>该位指示了PHY的操作模式。<br>0：对PHY进行读操作<br>1：对PHY进行写操作 |
| 0 | PB | PHY忙<br>该位指示了对PHY操作的状态。由应用程序置'1'后开始对PHY的进行读或者写操作，并需等到该位在操作完成后由硬件清'0'。在写ENET_MAC_PHY_CTL寄存器和读ENET_MAC_PHY_DATA寄存器之前，该位应当为'0'。 |

### 27.4.6. MAC PHY 数据寄存器(ENET_MAC_PHY_DATA)

地址偏移：0x0014
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

rw

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | PD[15:0] | PHY数据位<br>对于读操作，这些位为从PHY中读取的数据。对于写操作，这些位为将要写到PHY中的数据。 |

### 27.4.7. MAC 流控寄存器(ENET_MAC_FCTL)

地址偏移：0x0018
复位值：0x0000 0000

该寄存器用于配置控制帧的生成和接收。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | DZQP | 保留 | PLTS[1:0] | | UPFDT | RFCEN | TFCEN | FLCB/BKPA |
| | | | | | | | | rw | | rw | | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | PTM[15:0] | 暂停时间<br>这些位用来设置暂停控制帧时间域的值。 |
| 15:8 | 保留 | 必须保持复位值。 |
| 7 | DZQP | 关闭零时间片暂停功能<br>0：打开零时间片暂停控制帧自动生成功能<br>1：关闭零时间片暂停控制帧的自动生成 |
| 6 | 保留 | 必须保持复位值。 |
| 5:4 | PLTS[1:0] | 暂停低阈值<br>这些位设置了自动重发暂停帧的定时器阈值。这个阈值应当大于0，小于位[31:16]定义的暂停时间。低阈值的计算公式为PTM-PLTS。例如，PTM = 0x80(128个时间间隙)，PLTS = 0x1(28个时间间隙)，那么在第一个暂停帧发出100(128-28)个时间间隙后，将自动重发第二个暂停帧。<br>0x0：暂停时间 - 4个时间间隙<br>0x1：暂停时间 -28个时间间隙<br>0x2：暂停时间 - 144个时间间隙<br>0x3：暂停时间 - 256个时间间隙<br>**注意：**一个时间间隙是指MII接口发送512位(64字节)数据所需要的时间。 |

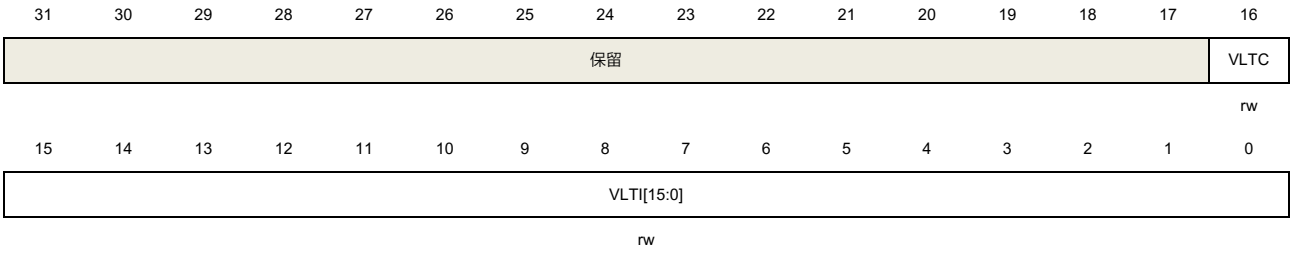
| | | |
|---|---|---|
| 3 | UPFDT | 单播暂停帧检测<br>0：MAC只接收符合IEEE802.3规范定义的唯一多播地址的暂停帧<br>1：除了唯一多播地址的暂停帧，MAC同时还会使用MAC0地址(ENET_MAC_ADDR0H寄存器和ENET_MAC_ADDR0L寄存器)来检测暂停帧。 |
| 2 | RFCEN | 接收流控使能位<br>0：MAC不解析暂停帧<br>1：MAC解析并处理接收到的暂停帧。MAC关闭发送器一段指定的时间(接收帧中的暂停时间域值)。 |
| 1 | TFCEN | 发送流控使能位<br>0：MAC关闭发送流控功能。在全双工模式下，MAC不发送暂停帧；在半双工模式下，MAC关闭背压功能。<br>1：MAC开启发送流控功能。在全双工模式下，MAC使能暂停帧发送；在半双工模式下，MAC使能背压功能。 |
| 0 | FLCB/BKPA | 流控忙/背压激活<br>该位仅在TFCEN位置位时有效。<br>在全双工模式下，该位可发送暂停帧；在半双工模式下，该位可激活背压功能。<br>在全双工模式下，应用程序要确保在写ENET_MAC_FCTL寄存器之前该位为'0'。置位该位后，MAC将发送一个暂停帧到接口，在发送控制帧的过程中，该位始终为'1'，直到暂停控制帧发送完成以后，MAC将该位重置为'0'。<br>在半双工模式下，设置该位为'1'可以激活背压功能。在背压功能有效时，如果MAC接收到新的帧，就会在发送端发送阻塞信号，通知有冲突发生。 |

### 27.4.8. MAC VLAN 标签寄存器(ENET_MAC_VLT)

地址偏移：0x001C
复位值：0x0000 0000

该寄存器包含了用来识别VLAN帧的IEEE802.1Q VLAN标签。MAC把接收到帧的第13，14字节(长度/类型域)与0x8100比较，再把之后的2个字节(第15，16字节)和VLAN标签比较。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:17 | 保留 | 必须保持复位值。 |
| 16 | VLTC | 12位VLAN标签比较位<br>该位选择用12位或16位VLAN标签来进行比较。<br>0：接收到的VLAN帧的全部16位数据(第15和16字节)都用来与VLTI位比对<br>1：仅用VLTI位[11:0]12位数据和接收到VLAN帧的相应域比对 |
| 15:0 | VLTI[15:0] | VLAN标签标识符位 |

这些位用来识别VLAN帧的802.1Q VLAN标签格式。格式如下：
VLTI[15:13]：UP(用户优先级)
VLTI[12]：CFI(标准格式指示符)
VLTI[11:0]：VID(VLAN标识符)
如果比较的位(当VLTC=1，则为VLTI[11:0]；当VLTC=0，则为VLTI[15:0])值是全'0'，则MAC不再比对检验VLAN帧的第15、16字节，并将接收帧的类型域值是0x8100的帧都直接视为VLAN帧。
如果用于比较的位不是全为'0'，则使用VLTI[11:0](VLTC=1)或VLTI[15:0](VLTC=0)进行比较。

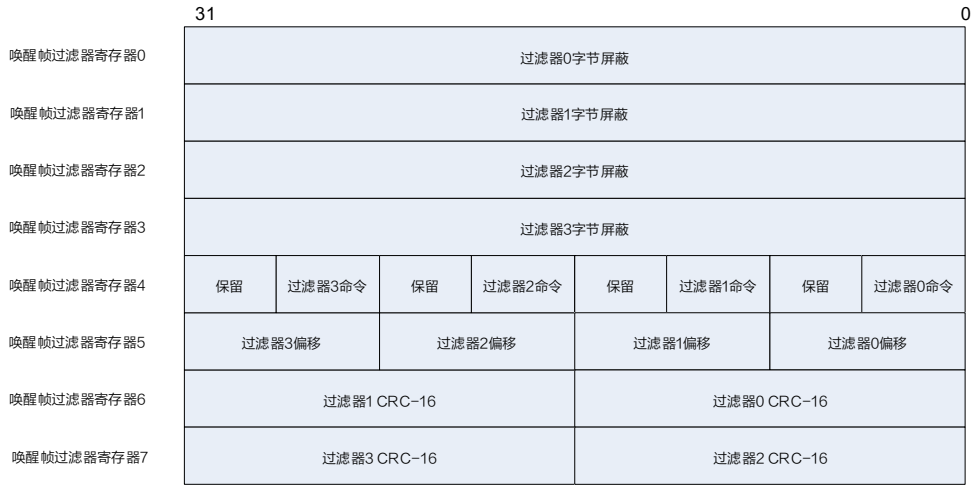
### 27.4.9. MAC 远程唤醒帧过滤器寄存器(ENET_MAC_RWFF)

地址偏移：0x0028
复位值：0x0000 0000

该寄存器实质上是指向8个不透明的唤醒帧过滤器寄存器的指针(使用同一个偏移地址)。对该寄存器地址(偏移为0x0028)的8次连续写操作，可以写入全部8个唤醒帧过滤器寄存器；对该寄存器地址(偏移为0x0028)的8次连续读操作，可以读出全部8个唤醒帧过滤器寄存器。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

**图 27-15. 远程唤醒帧过滤器寄存器**

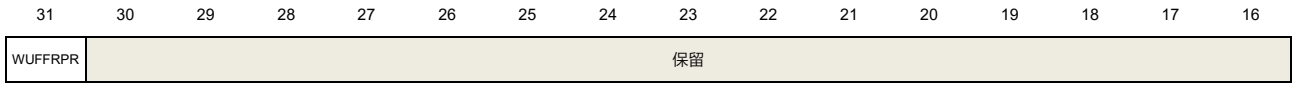



### 27.4.10. MAC 唤醒管理寄存器(ENET_MAC_WUM)

地址偏移：0x002C
复位值：0x0000 0000

该寄存器设置并监控唤醒事件。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



rs

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | GU | 保留 | | WUFR | MPKR | 保留 | | WFEN | MPEN | PWD |
| | | | | | | rw | | | rc_r | rc_r | | | rw | rw | rs |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | WUFFRPR | 唤醒帧过滤器寄存器指针复位<br>向该位写'1'，将会把远程唤醒帧过滤器寄存器指针ENET_MAC_RWFF复位，该位在指针复位完成后自动清'0'。<br>0：无作用<br>1：复位ENET_MAC_RWFF寄存器指针 |
| 30:10 | 保留 | 必须保持复位值。 |
| 9 | GU | 全局单播<br>向该位写1，所有能通过MAC地址过滤器的单播帧，都被认为是唤醒帧。<br>0：不是所有接收的单播帧都被认为是唤醒帧<br>1：所有能通过MAC地址过滤器的单播帧，都被认为是唤醒帧。 |
| 8:7 | 保留 | 必须保持复位值。 |
| 6 | WUFR | 接收到唤醒帧<br>读本寄存器可以清'0'该位。<br>0：没有接收到唤醒帧<br>1：接收到唤醒帧，并发生唤醒事件。 |
| 5 | MPKR | 接收到Magic Packet<br>读本寄存器可以清'0'该位。<br>0：没有接收到Magic Packet<br>1：接收到Magic Packet帧，并发生唤醒事件。 |
| 4:3 | 保留 | 必须保持复位值。 |
| 2 | WFEN | 唤醒帧使能位<br>0：禁能在接收到唤醒帧时产生唤醒事件<br>1：使能在接收到唤醒帧时产生唤醒事件 |
| 1 | MPEN | Magic Packet使能位<br>0：禁能在接收到Magic Packet唤醒帧时产生唤醒事件<br>1：使能在接收到Magic Packet唤醒帧时产生唤醒事件 |
| 0 | PWD | 低功耗位<br>该位由软件置位，由硬件复位。当该位置位，MAC丢弃所有接收到的帧。当发生了唤醒事件，使得退出低功耗模式，硬件会自动将该位清'0'。 |

## 27.4.11. MAC 调试寄存器(ENET_MAC_DBG)

地址偏移：0x0034

复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | TXFF | TXFNE | 保留 | TXFW | TXFRS[1:0] | | PCS | SOMT[1:0] | | MTNI |
| | | | | | | ro | ro | | ro | ro | | ro | ro | | ro |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | RXFS[1:0] | | 保留 | RXFRS[1:0] | | RXFW | 保留 | RXAFS[1:0] | | MRNI |
| | | | | | | ro | | | ro | | ro | | ro | | ro |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:26 | 保留 | 必须保持复位值。 |
| 25 | TXFF | TxFIFO满标志位<br>0：TxFIFO未满<br>1：TxFIFO已满 |
| 24 | TXFNE | TxFIFO非空标志位<br>0: TxFIFO空<br>1: TxFIFO不为空 |
| 23 | 保留 | 必须保持复位值。 |
| 22 | TXFW | 正在写TxFIFO<br>0: 没有向TxFIFO写帧数据<br>1：正在向TxFIFO写帧数据 |
| 21:20 | TXFRS[1:0] | TxFIFO读操作状态<br>0x0：空闲状态<br>0x1：读状态<br>0x2：等待MAC发送器返回Tx状态<br>0x3：写发送描述符状态，或清空TxFIFO。 |
| 19 | PCS | 暂停状态<br>0：MAC发送器不处于暂停状态<br>1：MAC发送器处于暂停状态，并暂停发送帧。 |
| 18:17 | SOMT[1:0] | MAC发送器状态<br>0x0：空闲状态<br>0x1：等待前一帧的状态返回或IFG/BACKOFF周期结束<br>0x2：对于全双工模式，表示正在发送暂停控制帧。<br>0x3：从FIFO读取待发送的帧 |
| 16 | MTNI | MAC发送器不空闲位<br>0：MAC发送器处于空闲状态<br>1：MAC发送器处于非空闲状态 |
| 15:10 | 保留 | 必须保持复位值。 |
| 9:8 | RXFS | RxFIFO状态<br>0x0：RxFIFO空 |

| | | |
|---|---|---|
| | | 0x1：RxFIFO中字节数低于流控下阈值<br>0x2：RxFIFO中字节数高于流控上阈值<br>0x3：RxFIFO满 |
| 7 | 保留 | 必须保持复位值。 |
| 6:5 | RXFRS[1:0] | RxFIFO读操作状态<br>0x0：空闲状态<br>0x1：读数据帧状态<br>0x2：读帧状态(包括时间戳)<br>0x3：清空帧 |
| 4 | RXFW | 正在写RxFIFO<br>0：没有向RxFIFO中写帧数据<br>1：正在向RxFIFO写帧数据 |
| 3 | 保留 | 必须保持复位值。 |
| 2:1 | RXAFS[1:0] | Rx异步FIFO状态<br>RXAFS[1]：Rx异步FIFO在HCLK时钟域进行读操作(MAC取出数据)<br>RXAFS[0]：Rx异步FIFO在MAC RX_CLK时钟域进行写操作(MAC存入数据) |
| 0 | MRNI | MAC接收器不空闲位<br>0：MAC接收器处于空闲状态<br>1：MAC接收器处于非空闲状态 |

## 27.4.12. MAC 中断状态寄存器(ENET_MAC_INTF)

地址偏移：0x0038
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | TMST | 保留 | | MSCT | MSCR | MSC | WUM | 保留 | | |
| | | | | | | rc_r | | | r | r | r | r | | | |

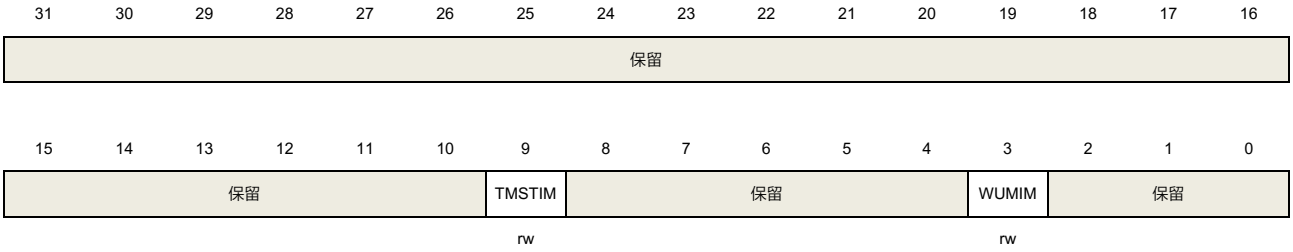
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:10 | 保留 | 必须保持复位值。 |
| 9 | TMST | 时间戳触发状态<br>读 ENET_PTP_TSF 寄存器可以清'0'该位。<br>0：系统时间值小于期望时间值<br>1：系统时间值等于或者超过期望时间值 |

| | | |
|---|---|---|
| 8:7 | 保留 | 必须保持复位值。 |
| 6 | MSCT | MSC 发送状态<br>0：没有产生任一 ENET_MSC_TINTF 寄存器中的中断<br>1：产生任一 ENET_MSC_TINTF 寄存器中的中断 |
| 5 | MSCR | MSC 接收状态<br>0：没有产生任一 ENET_MSC_RINTF 寄存器中的中断<br>1：产生任一 ENET_MSC_RINTF 寄存器中的中断 |
| 4 | MSC | MSC 状态<br>该位为 MSCT 位与 MSCR 位的逻辑或。<br>0：MSCT 位和 MSCR 位均为'0'<br>1：MSCT 位和 MSCR 位中有位为'1' |
| 3 | WUM | WUM 状态<br>该位为 ENET_MAC_WUM 寄存器中的 WUFR 和 MPKR 位的逻辑或。<br>0：未接收到唤醒帧或者 Magic Packet 帧<br>1：在低功耗模式下，接收到唤醒帧或者 Magic Packet。 |
| 2:0 | 保留 | 必须保持复位值。 |

## 27.4.13. MAC 中断屏蔽寄存器(ENET_MAC_INTMSK)

地址偏移：0x003C
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



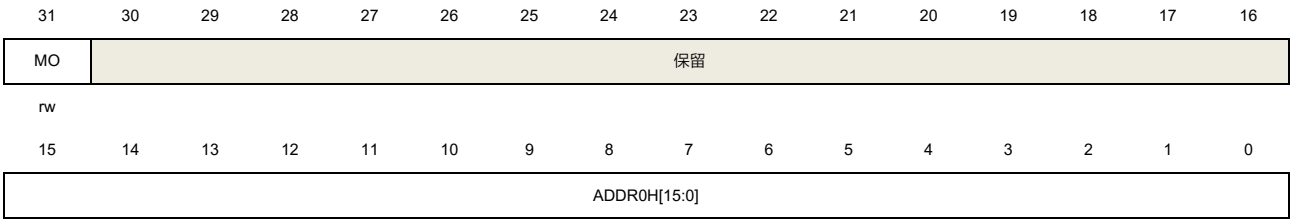

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:10 | 保留 | 必须保持复位值。 |
| 9 | TMSTIM | 时间戳触发中断屏蔽位<br>0：允许产生时间戳中断<br>1：禁止产生时间戳中断 |
| 8:4 | 保留 | 必须保持复位值。 |
| 3 | WUMIM | WUM中断屏蔽位<br>0：允许由于ENET_MAC_INTF寄存器的WUM状态位置位而引发的中断<br>1：禁止由于 ENET_MAC_INTF 寄存器的 WUM 状态位置 1 而引发的中断 |

| | | |
|---|---|---|
| 2:0 | 保留 | 必须保持复位值。 |

### 27.4.14. MAC 地址 0 高寄存器(ENET_MAC_ADDR0H)

地址偏移：0x0040
复位值：0x8000 FFFF

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

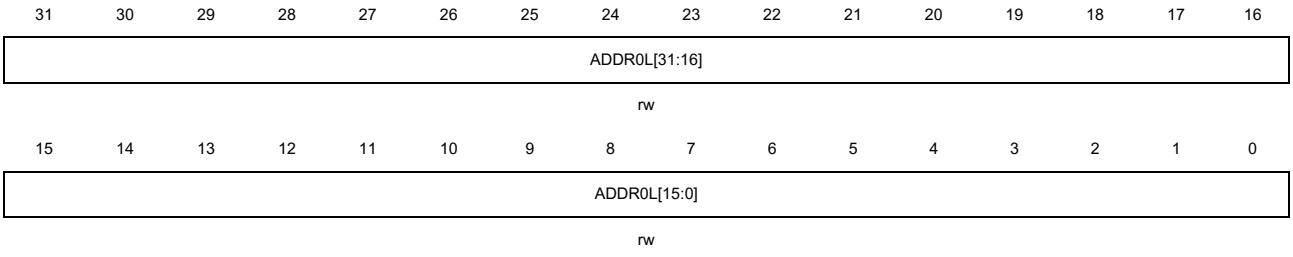



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | MO | 该位总是为'1'。 |
| 30:16 | 保留 | 必须保持复位值。 |
| 15:0 | ADDR0H[15:0] | MAC地址0高16位<br>这些位包含了 6 字节 MAC 地址 0 的高 16 位，这些位用于作为接收帧的地址过滤，还用于发送流控中发送暂停帧时插入作为帧的源地址。 |

### 27.4.15. MAC 地址 0 低寄存器(ENET_MAC_ADDR0L)

地址偏移：0x0044
复位值：0xFFFF FFFF

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | ADDR0L[31:0] | MAC地址0低32位<br>这些位包含了 6 字节 MAC 地址 0 的低 32 位，这些位用于作为接收帧的地址过滤，还用于发送流控中发送暂停帧时插入作为帧的源地址。 |

### 27.4.16. MAC 地址 1 高寄存器(ENET_MAC_ADDR1H)

地址偏移：0x0048
复位值：0x0000 FFFF

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

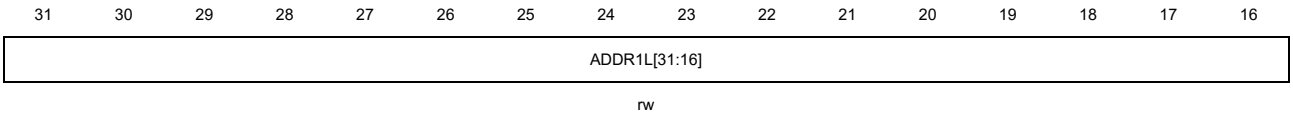



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | AFE | 地址过滤使能<br>0：不使用 MAC 地址 1 进行地址过滤<br>1：地址过滤器使用 MAC 地址 1 来进行完美过滤 |
| 30 | SAF | 源地址过滤器<br>0：MAC 地址 1[47:0]用来和接收帧的目标地址进行比对<br>1：MAC 地址 1[47:0]用来和接收帧的源地址进行比对 |
| 29:24 | MB[5:0] | 屏蔽字节位<br>当某个位置'1'时，MAC 不再把接收帧目标地址/源地址的对应字节与 MAC 地址 1 的相应字节进行比较。每个控制位对应的 MAC 地址字节如下:<br>MB[5]：ENET_MAC_ADDR1H [15:8]<br>MB[4]：ENET_MAC_ADDR1H [7:0]<br>MB[3]：ENET_MAC_ADDR1L [31:24]<br>MB[2]：ENET_MAC_ADDR1L[23:16]<br>MB[1]：ENET_MAC_ADDR1L[15:8]<br>MB[0]：ENET_MAC_ADDR1L [7:0] |
| 23:16 | 保留 | 必须保持复位值。 |
| 15:0 | ADDR1H[15:0] | MAC 地址 1 高[47:32]位<br>这些位包含了 6 字节的 MAC 地址 1 的高 16 位。 |

### 27.4.17. MAC 地址 1 低寄存器(ENET_MAC_ADDR1L)

地址偏移：0x004C
复位值：0xFFFF FFFF

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADDR1L[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | ADDR1L[31:0] | MAC 地址 1 低 32 位<br>这些位包含了 6 字节 MAC 地址 1 的低 32 位。 |

## 27.4.18. MAC 地址 2 高寄存器(ENET_ MAC_ADDR2H)

地址偏移：0x0050
复位值：0x0000 FFFF

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| AFE | SAF | MB[5:0] | | | | | | 保留 | | | | | | | |
| rw | rw | rw | | | | | | | | | | | | | |

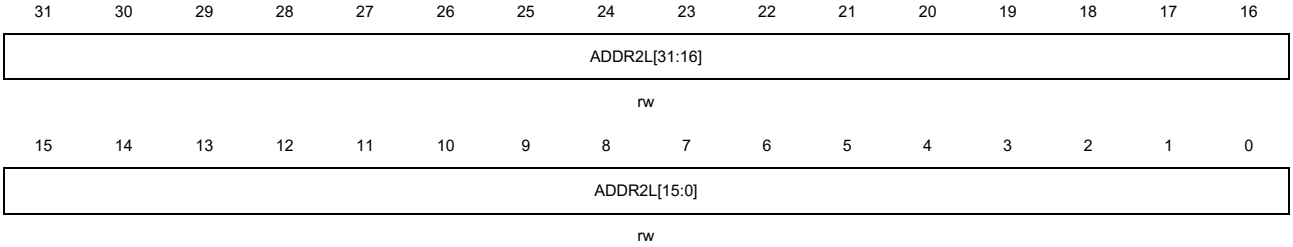
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADDR2H[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | AFE | 地址过滤使能<br>0：不使用 MAC 地址 2 进行地址过滤<br>1：地址过滤器使用 MAC 地址 2 来进行完美过滤 |
| 30 | SAF | 源地址过滤器<br>0：MAC 地址 2[47:0]用来和接收帧的目标地址进行比对<br>1：MAC 地址 2[47:0]用来和接收帧的源地址进行比对 |
| 29:24 | MB[5:0] | 屏蔽字节位<br>当某个位置'1'时，MAC 不再把接收帧目标地址/源地址的对应字节与 MAC 地址 2 的相应字节进行比较。每个控制位对应的 MAC 地址字节如下:<br>MB[5]：ENET_MAC_ADDR2H [15:8]<br>MB[4]：ENET_MAC_ADDR2H [7:0]<br>MB[3]：ENET_MAC_ADDR2L [31:24]<br>MB[2]：ENET_MAC_ADDR2L[23:16]<br>MB[1]：ENET_MAC_ADDR2L[15:8]<br>MB[0]：ENET_MAC_ADDR2L [7:0] |
| 23:16 | 保留 | 必须保持复位值。 |
| 15:0 | ADDR2H[15:0] | MAC 地址 2 高[47:32]位<br>这些位包含了 6 字节的 MAC 地址 2 的高 16 位。 |

### 27.4.19. MAC 地址 2 低寄存器(ENET_MAC_ADDR2L)

地址偏移：0x0054
复位值：0xFFFF FFFF

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

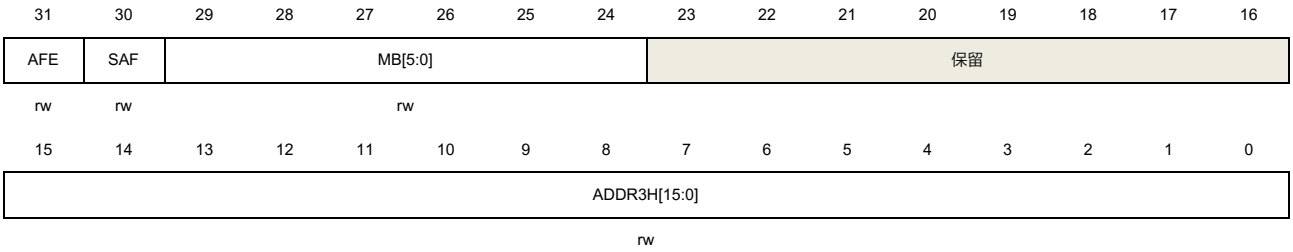



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | ADDR2L[31:0] | MAC 地址 2 低 32 位<br>这些位包含了 6 字节 MAC 地址 2 的低 32 位。 |

### 27.4.20. MAC 地址 3 高寄存器(ENET_MAC_ADDR3H)

地址偏移：0x0058
复位值：0x0000 FFFF

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | AFE | 地址过滤使能<br>0：不使用 MAC 地址 3 进行地址过滤<br>1：地址过滤器使用 MAC 地址 3 来进行完美过滤 |
| 30 | SAF | 源地址过滤器<br>0：MAC 地址 3[47:0]用来和接收帧的目标地址进行比对<br>1：MAC 地址 3[47:0]用来和接收帧的源地址进行比对 |
| 29:24 | MB[5:0] | 屏蔽字节位<br>当某个位置'1'时，MAC 不再把接收帧目标地址/源地址的对应字节与 MAC 地址 3 的相应字节进行比较。每个控制位对应的 MAC 地址字节如下:<br>MB[5]：ENET_MAC_ADDR3H [15:8]<br>MB[4]：ENET_MAC_ADDR3H [7:0]<br>MB[3]：ENET_MAC_ADDR3L [31:24] |

MB[2]：ENET_MAC_ADDR3L[23:16]
MB[1]：ENET_MAC_ADDR3L[15:8]
MB[0]：ENET_MAC_ADDR3L [7:0]

| | | |
|---|---|---|
| 23:16 | 保留 | 必须保持复位值。 |
| 15:0 | ADDR3H[15:0] | MAC 地址 3 高[47:32]位<br>这些位包含了 6 字节的 MAC 地址 3 的高 16 位。 |

## 27.4.21. MAC 地址 3 低寄存器(ENET_MAC_ADDR3L)

地址偏移：0x005C
复位值：0xFFFF FFFF

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADDR3L[31:16] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ADDR3L[15:0] | | | | | | | | | | | | | | | |
| rw | | | | | | | | | | | | | | | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | ADDR3L[31:0] | MAC 地址 3 低 32 位<br>这些位包含了 6 字节 MAC 地址 3 的低 32 位。 |

## 27.4.22. MAC 流控阈值寄存器(ENET_MAC_FCTH)

地址偏移：0x1080
复位值：0x0000 0015

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:7 | 保留 | 必须保持复位值。 |
| 6:4 | RFD[2:0] | 流控失效阈值<br>这些位设置了流控失效的阈值。这个值应当小于位[2:0]定义的流控激活阈值。当 RxFIFO 中未处理的数据低于这些位所设置的值，流控功能将自动失效。 |

0x0：256 字节
0x1：512 字节
0x2：768 字节
0x3：1024 字节
0x4：1280 字节
0x5：1536 字节
0x6，0x7：1792 字节

| | | |
|---|---|---|
| 3 | 保留 | 必须保持复位值。 |
| 2:0 | RFA[2:0] | 流控激活阈值<br>这些位设置了流控激活的阈值。若使能了流控功能，当 RxFIFO 中未处理的数据超过了这些位所设置的值，流控功能将被激活。<br>0x0：256 字节<br>0x1：512 字节<br>0x2：768 字节<br>0x3：1024 字节<br>0x4：1280 字节<br>0x5：1536 字节<br>0x6，0x7：1792 字节 |

## 27.4.23. MSC 控制寄存(ENET_MSC_CTL)

地址偏移：0x0100
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | AFHPM | PMC | MCFZ | RTOR | CTSR | CTR |
| | | | | | | | | | | rw | wo | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:6 | 保留 | 必须保持复位值。 |
| 5 | AFHPM | 近似全值或半值预设模式<br>0：预设 MSC 计数器的值为近似半值(0x7FFF FFF0)<br>1：预设 MSC 计数器的值为近似全值(0xFFFF FFF0)<br>**注意：**该位仅在 PMC 位置位时有效。 |
| 4 | PMC | MAC 计数器预设位<br>0：无作用<br>1：将 MSC 计数器预设为一个预设值。预设值取决于 AFHPM 位。 |

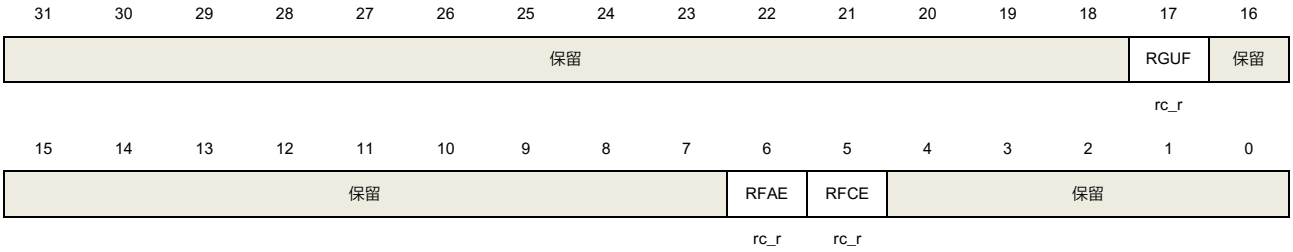
| | | |
|---|---|---|
| 3 | MCFZ | MSC 计数器冻结位<br>0：MSC 计数器正常工作<br>1：冻结 MSC 计数器，保持它们的当前值。RTOR 位可在计数器冻结状态时工作。 |
| 2 | RTOR | 读时复位<br>0：读 MSC 计数器后，计数器不复位。<br>1：读 MSC 计数器后，计数器复位。 |
| 1 | CTSR | 计数器停止回转<br>0：计数器在计数到最大值后，会重新从 0 开始计数。<br>1：计数器在计数到最大值后，不会重新从 0 开始计数。 |
| 0 | CTR | 计数器复位<br>该位置位后，会在 1 个时钟周期后由硬件自动清零。<br>0：无作用<br>1：复位所有计数器 |

### 27.4.24. MSC 接收中断状态寄存器(ENET_MSC_RINTF)

地址偏移：0x0104
复位值：0x0000 0000

该寄存器的中断标志位需要读取导致该中断产生的相应的MSC计数器才能清除

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:18 | 保留 | 必须保持复位值。 |
| 17 | RGUF | 接收到"好"的单播帧<br>0：接收"好"单播帧计数器值小于最大值的一半<br>1：接收"好"单播帧计数器值达到最大值的一半<br>**注意**：需要通过读取 ENET_MSC_RGUFCNT 寄存器来清除该标志位 |
| 16:7 | 保留 | 必须保持复位值。 |
| 6 | RFAE | 接收到帧对齐错误<br>0：对齐错误接收帧计数器值小于最大值的一半<br>1：对齐错误接收帧计数器值达到最大值的一半<br>**注意**：需要通过读取 ENET_MSC_RFAECNT 寄存器来清除该标志位 |
| 5 | RFCE | 接收到帧 CRC 错误 |

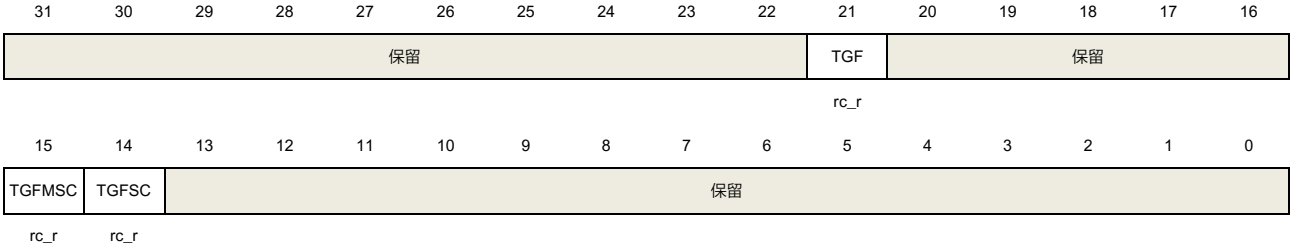
0：CRC 错误接收帧计数器值小于最大值的一半
1：CRC 错误接收帧计数器值达到最大值的一半
**注意**：需要通过读取 ENET_MSC_RFCECNT 寄存器来清除该标志位

| | | |
|---|---|---|
| 4:0 | 保留 | 必须保持复位值。 |

## 27.4.25. MSC 发送中断状态寄存器(ENET_MSC_TINTF)

地址偏移：0x0108
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

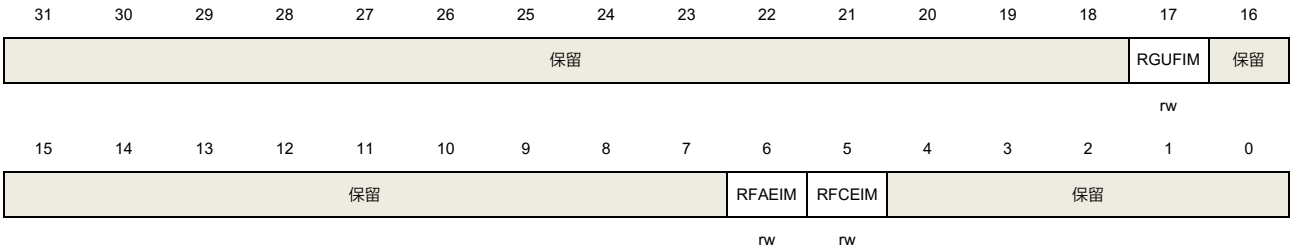



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:22 | 保留 | 必须保持复位值。 |
| 21 | TGF | 发送"好"的帧<br>0：发送"好"单播帧计数器值小于最大值的一半<br>1：发送"好"单播帧计数器值达到最大值的一半<br>**注意**：需要通过读取 ENET_MSC_TGFCNT 寄存器来清除该标志位 |
| 20:16 | 保留 | 必须保持复位值。 |
| 15 | TGFMSC | 发送"好"的帧时遇到 1 个以上冲突<br>0：1 次以上冲突后发送"好"帧计数器值小于最大值的一半<br>1：1 次以上冲突后发送"好"帧计数器值达到最大值的一半<br>**注意**：需要通过读取 ENET_MSC_MSCCNT 寄存器来清除该标志位 |
| 14 | TGFSC | 发送"好"的帧时仅遇到 1 个冲突<br>0：1 次冲突后发送"好"帧计数器值小于最大值的一半<br>1：1 次冲突后发送"好"帧计数器值达到最大值的一半<br>**注意**：需要通过读取 ENET_MSC_SCCNT 寄存器来清除该标志位 |
| 13:0 | 保留 | 必须保持复位值。 |

## 27.4.26. MSC 接收中断屏蔽寄存器(ENET_MSC_RINTMSK)

地址偏移：0x010C
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



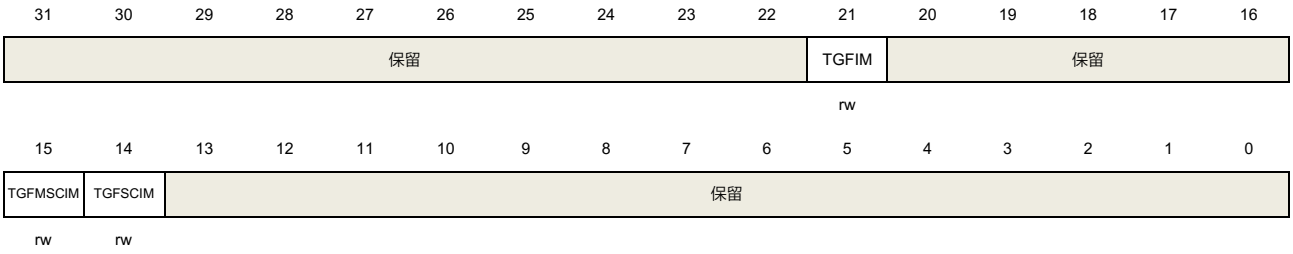

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:18 | 保留 | 必须保持复位值。 |
| 17 | RGUFIM | 接收到"好"的单播帧的中断屏蔽位<br>0：不屏蔽当 RGUF 位为'1'时发生的中断<br>1：屏蔽当 RGUF 位为'1'时发生的中断 |
| 16:7 | 保留 | 必须保持复位值。 |
| 6 | RFAEIM | 接收帧对齐错误中断屏蔽位<br>0：不屏蔽当 RFAE 位为'1'时发生的中断<br>1：屏蔽当 RFAE 位为'1'时发生的中断 |
| 5 | RFCEIM | 接收帧 CRC 错误中断屏蔽位<br>0：不屏蔽当 RFCE 位为'1'时发生的中断<br>1：屏蔽当 RFCE 位为'1'时发生的中断 |
| 4:0 | 保留 | 必须保持复位值。 |

## 27.4.27. MSC 发送中断屏蔽寄存器(ENET_MSC_TINTMSK)

地址偏移：0x0110
复位值：0x0000 0000

该寄存器可以设置相应中断的屏蔽位。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:22 | 保留 | 必须保持复位值。 |
| 21 | TGFIM | 发送"好"的帧的中断屏蔽位<br>0：不屏蔽当 TGF 位为'1'时发生的中断 |

| | | |
|---|---|---|
| | | 1：屏蔽当 TGF 位为'1'时发生的中断 |
| 20:16 | 保留 | 必须保持复位值。 |
| 15 | TGFMSCIM | 遇到 1 个以上冲突后发送"好"帧中断屏蔽位<br>0：不屏蔽当 TGFMSC 位为'1'时发生的中断<br>1：屏蔽当 TGFMSC 位为'1'时发生的中断 |
| 14 | TGFSCIM | 仅遇到 1 个冲突后发送"好"帧中断屏蔽位<br>0：不屏蔽当 TFGSC 位为'1'时发生的中断<br>1：屏蔽当 TFGSC 位为'1'时发生的中断 |
| 13:0 | 保留 | 必须保持复位值。 |

### 27.4.28. MSC 1 次冲突后发送"好"帧的计数器寄存器(ENET_MSC_SCCNT)

地址偏移：0x014C
复位值：0x0000 0000

该寄存器统计在半双工模式下，在只遇到一次冲突后发送帧成功时的帧的数目。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | SCC[31:0] | 1 次冲突后发送好帧计数器<br>这些位是 1 次冲突后发送的"好"帧的计数器。 |

### 27.4.29. MSC 1 次以上冲突后发送"好"帧的计数器寄存器(ENET_MSC_MSCCNT)

地址偏移：0x0150
复位值：0x0000 0000

该寄存器统计在半双工模式下，遇到一次以上冲突后发送帧成功时的帧的数目。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

r

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:0 | MSCC[31:0] | 1 次以上冲突后发送"好"帧计数器<br>这些位是 1 次以上冲突后发送"好"帧的计数器。 |

## 27.4.30. MSC 发送"好"帧计数器寄存器(ENET_MSC_TGFCNT)

地址偏移：0x0168
复位值：0x0000 0000

该寄存器统计发送"好"帧的数目。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:0 | TGF[31:0] | 发送"好"帧计数器<br>这些位是发送"好"帧的计数器。 |

## 27.4.31. MSC CRC 错误接收帧计数器寄存器(ENET_MSC_RFCECNT)

地址偏移：0x0194
复位值：0x0000 0000

该寄存器统计接收帧中有CRC错误的帧的数目。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:0 | RFCER[31:0] | CRC 错误接收帧计数器<br>这些位是接收帧中有 CRC 错误的帧的计数器。 |

### 27.4.32. MSC 对齐错误接收帧计数器寄存器(ENET_MSC_RFAECNT)

地址偏移：0x0198
复位值：0x0000 0000

该寄存器统计接收帧中有对齐错误帧的数目。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | RFAER[31:0] | 对齐错误接收帧计数器<br>这些位是接收帧中有对齐错误的帧的计数器。 |

### 27.4.33. MSC"好"单播帧接收帧计数器寄存器(ENET_MSC_RGUFCNT)

地址偏移：0x01C4
复位值：0x0000 0000

该寄存器统计接收到"好"单播帧的数目。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | RGUF[31:0] | "好"单播帧接收帧计数器<br>这些位是接收到"好"的单播帧的计数器。 |

### 27.4.34. PTP 时间戳控制寄存器(ENET_PTP_TSCTL)

地址偏移：0x0700
复位值：0x0000 2000

该寄存器用于配置时间戳的产生和更新。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MNMSEN | ETMSEN | IP4SEN | IP6SEN | ESEN | PFSV | SCROM | ARFSEN | 保留 | | TMSARU | TMSITEN | TMSSTU | TMSSTI | TMSFCU | TMSEN |
| rw | rw | rw | rw | rw | rw | rw | rw | | | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:19 | 保留 | 必须保持复位值。 |
| 18 | MAFEN | PTP 帧 MAC 地址过滤使能<br>0：无作用<br>1：当接收帧的类型域值为 0x88f7，则使能 MAC 地址 1-3 用于 PTP 帧过滤。 |
| 17:16 | CKNT[1:0] | 时间戳时钟节点类型<br>0x0：普通类型时钟<br>0x1：边界类型时钟<br>0x2：端对端透明类型时钟<br>0x3：点对点透明类型时钟 |
| 15 | MNMSEN | 接收主节点消息时时间戳快照使能<br>该位仅在 CKNT=0x0 或 0x1 时有效。<br>0：从节点消息时间戳快照使能<br>1：主节点消息时间戳快照使能 |
| 14 | ETMSEN | 接收事件类型的消息时时间戳快照使能<br>0：接收到除了 Announce，Management 和 Signaling 以外的所有其他类型的消息时，时间戳快照使能。<br>1：只有接收到事件类型的消息(SYNC，DELAY_REQ，PDELAY_REQ 和 PDELAY_RESP)时，时间戳快照使能。 |
| 13 | IP4SEN | 接收通过 IPv4-UDP 封装的 PTP 数据帧时时间戳使能<br>0：接收到通过 IPv4-UDP 封装的 PTP 数据帧时，时间戳失能。<br>1：接收到通过 IPv4-UDP 封装的 PTP 数据帧时，时间戳使能。 |
| 12 | IP6SEN | 接收通过 IPv6-UDP 封装的 PTP 数据帧时时间戳使能<br>0：接收到通过 IPv6-UDP 封装的 PTP 数据帧时，时间戳失能。<br>1：接收到通过 IPv6-UDP 封装的 PTP 数据帧时，时间戳使能。 |
| 11 | ESEN | 接收通过 Ethernet II 型封装的 PTP 数据帧时时间戳使能<br>0：接收到通过 Ethernet II 型封装的 PTP 数据帧时，时间戳失能。<br>1：接收到通过 Ethernet II 型封装的 PTP 数据帧时，时间戳使能。 |
| 10 | PFSV | 监听 PTP 帧版本<br>0：版本 1(版本为 IEEE STD. 1588-2002/1588-2008)<br>1：版本 2(版本为 IEEE STD. 1588-2008) |

| | | |
|---|---|---|
| 9 | SCROM | 亚秒计数器回转模式<br>0：二进制回转模式，亚秒计数器在达到 0x7FFF FFFF 以后重新从 0 计数。<br>1：十进制回转模式，亚秒计数器在达到 0x3B9A C9FF(0d999 999 999)以后重新从 0 计数。 |
| 8 | ARFSEN | 所有接收帧时间戳快照使能<br>0：不对所有接收帧使能时间戳功能<br>1：对所有接收帧使能时间戳功能 |
| 5 | TMSARU | 时间戳加数寄存器更新位<br>该位在更新完成后清'0'。该位在置位前必须确保读出为'0'。<br>0：不将时间戳加数寄存器的值更新到 PTP 模块进行精调<br>1：将时间戳加数寄存器的值更新到 PTP 模块进行精调 |
| 4 | TMSITEN | 时间戳中断触发使能<br>0：禁止时间戳中断<br>1：使能时间戳中断，当系统时间超过期望时间寄存器的值时将会产生中断。<br>**注意：**产生时间戳中断后，该位将会清'0'。 |
| 3 | TMSSTU | 时间戳系统时间更新位<br>置位该位之前，必须确保 TMSSTU 位和 TMSSTI 位读出为'0'。<br>0：系统时间保持不变<br>1：更新系统时间，在原有系统时间上加上或者减去时间戳高和低更新寄存器的值。完成更新后，硬件将会清除该位。 |
| 2 | TMSSTI | 时间戳系统时间初始化位<br>置位该位之前，必须确保该位读出为'0'。<br>0：系统时间保持不变<br>1：初始化系统时间，将原有系统时间替换为时间戳高和低更新寄存器的值。在初始化完成后，硬件将会清除该位。 |
| 1 | TMSFCU | 时间戳粗调或者精调更新位<br>0：用粗调的方式更新系统时间戳<br>1：用精调的方式更新系统时间戳 |
| 0 | TMSEN | 时间戳使能位<br>0：禁止时间戳功能<br>1：使能接收和发送帧的时间戳功能<br>**注意：**每次设置该位为'1'后，都需要重新初始化系统时间。 |

**表 27-8. 支持的 PTP 时间戳及其寄存器配置**

| **CKNT (位 17:16)** | **0X** | | | **10** | | **11** | |
|---|---|---|---|---|---|---|---|
| **MNMSEN (位 15)** | X(*) | 1 | 0 | X | | | |
| **ETMSEN (位 14)** | 0 | 1 | 1 | 0 | 1 | 0 | 1 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **支持的时间戳消息类型** | SYNC, FOLLOW_UP, DELAY_REQ, DELAY_RESP | DELAY_REQ | SYNC | SYNC, FOLLOW_UP, DELAY_REQ, DELAY_RESP | SYNC, FOLLOW_UP | SYNC, FOLLOW_UP, DELAY_REQ, DELAY_RESP, PDELAY_REQ, PDELAY_RESP | SYNC, PDELAY_REQ, PDELAY_RESP |

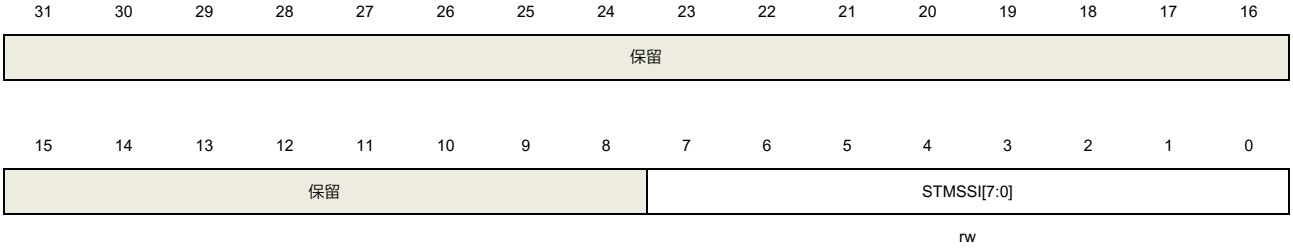
*：指无关值。

## 27.4.35. PTP 亚秒递增寄存器(ENET_PTP_SSINC)

地址偏移：0x0704
复位值：0x0000 0000

该寄存器用于配置亚秒递增寄存器的8位递增值。在粗调模式下，每个HCLK时钟周期，系统时间就加一次该寄存器的值。在精调模式下，在累加器溢出时，系统时间才加一次该寄存器的值。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

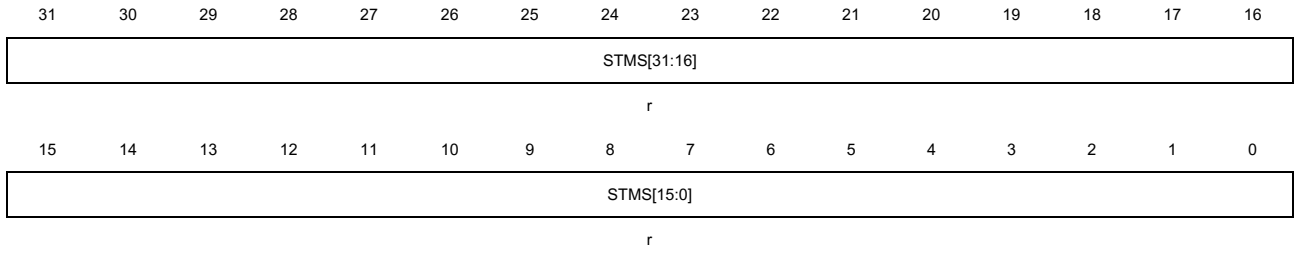



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7:0 | STMSSI[7:0] | 系统时间亚秒递增<br>在每次系统时间递增时，把这些位的值加到系统时间的亚秒值上。 |

## 27.4.36. PTP 时间戳高寄存器(ENET_PTP_TSH)

地址偏移：0x0708
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

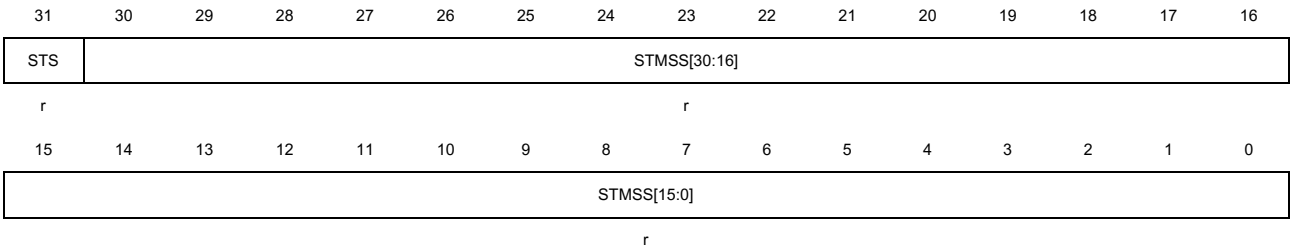


| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:0 | STMS[31:0] | 系统时间秒位<br>这些位表示了当前系统时间的秒值。 |

## 27.4.37. PTP 时间戳低寄存器(ENET_PTP_TSL)

地址偏移：0x070C
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



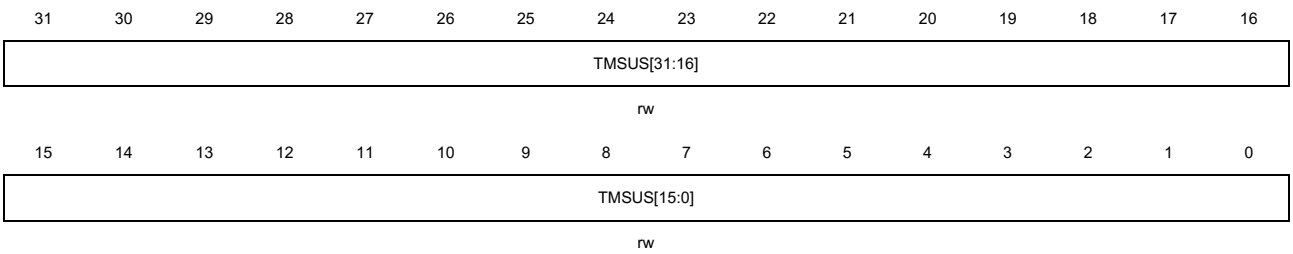

| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31 | STS | 系统时间符号位<br>0：时间值是正的<br>1：时间值是负的 |
| 30:0 | STMSS[30:0] | 系统时间亚秒位<br>这些位表示了当前系统时间的亚秒值，亚秒精度为 0.46ns。 |

## 27.4.38. PTP 时间戳高更新寄存器(ENET_PTP_TSUH)

地址偏移：0x0710
复位值：0x0000 0000

使用该寄存器的值对当前系统时间替换、加或减。时间戳高和低更新寄存器可以用来初始化或更新MAC的当前系统时间。应当先写这2个寄存器，再置位时间戳控制寄存器的TMSSTI位或TMSSTU位。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:0 | TMSUS[31:0] | 时间戳秒更新位 |

这些位表示的值在初始化时用于替换系统时间，在更新时表示在系统时间上加上或减去的秒值。

## 27.4.39. PTP 时间戳低更新寄存器(ENET_PTP_TSUL)

地址偏移：0x0714
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMSUPNS | TMSUSS[30:16] | | | | | | | | | | | | | | |
| rw | rw | | | | | | | | | | | | | | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | TMSUPNS | 时间戳更新正或者负符号位<br>TMSSTI 位置'1'时，该位应当为'0'。<br>0：在系统时间上加上时间戳更新值<br>1：从系统时间中减去时间戳更新值 |
| 30:0 | TMSUSS[30:0] | 时间戳更新亚秒位<br>这些位表示的值在初始化时用于替换系统时间，在更新时表示在系统时间上加上或减去的亚秒值。 |

## 27.4.40. PTP 时间戳加数寄存器(ENET_PTP_TSADDEND)

地址偏移：0x0718
复位值：0x0000 0000

该寄存器只用于系统时间更新方式为精调模式。该寄存器的值在每个时钟周期都会累加到32位累加器上，一旦该累加器溢出就更新系统时间。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | TMSA[31:0] | 时间戳加数 |

这些位用于时钟同步时加到累加器上的值，以实现时间同步。

### 27.4.41. PTP 期望时间高寄存器(ENET_PTP_ETH)

地址偏移：0x071C
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | ETSH[31:0] | 期望时间戳高位<br>这些位表示了期望时间的秒值。 |

### 27.4.42. PTP 期望时间低寄存器(ENET_PTP_ETL)

地址偏移：0x0720
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

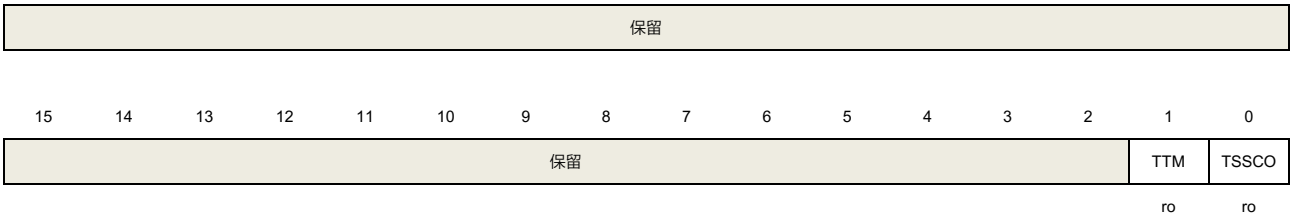
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | ETSL[31:0] | 期望时间戳低位<br>这些位表示了期望时间的纳秒值。 |

### 27.4.43. PTP 时间戳标志寄存器(ENET_PTP_TSF)

地址偏移：0x0728
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

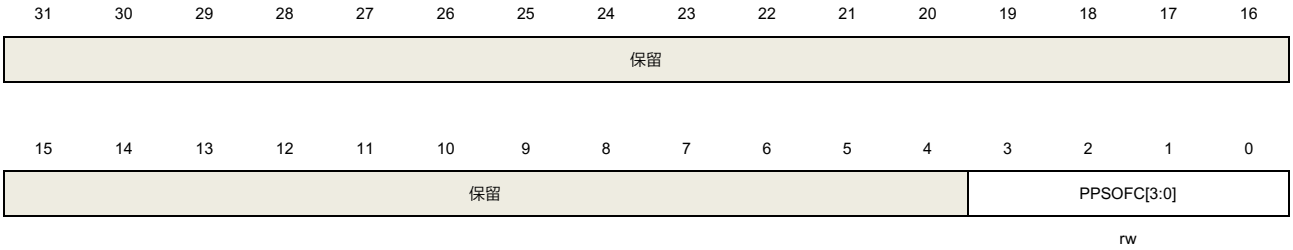



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |
| 1 | TTM | 期望时间比较位<br>0：系统时间小于期望时间<br>1：系统时间大于或等于期望时间<br>**注意：** 读 ENET_PTP_TSF 寄存器将清除该位。 |
| 0 | TSSCO | 时间戳秒计数器上溢位<br>0：时间戳秒计数器没有发生上溢<br>1：时间戳秒计数器值大于 0xFFFF FFFF |

## 27.4.44. PTP PPS 控制寄存器(ENET_PTP_PPSCTL)

地址偏移：0x072C
复位值：0x0000 0000

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

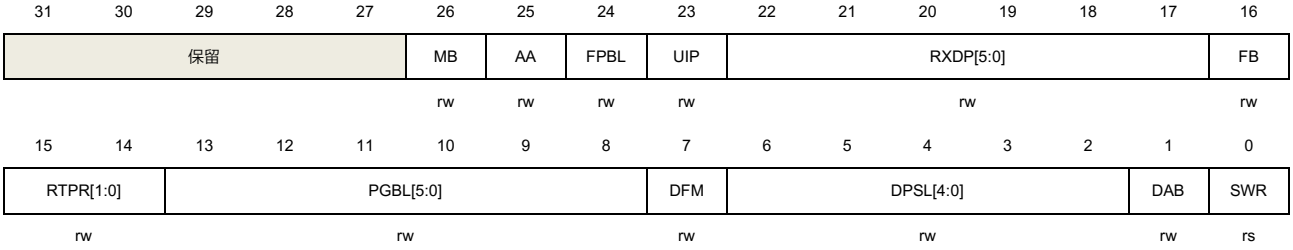



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:4 | 保留 | 必须保持复位值。 |
| 3:0 | PPSOFC[3:0] | PPS 输出频率配置位<br>0x0：1Hz(脉冲宽度：二进制回转模式下为 125ms，十进制回转模式下为 100ms)<br>0x1：2Hz(脉冲宽度：二进制回转模式下 50%占空比)<br>0x2：4Hz(脉冲宽度：二进制回转模式下 50%占空比)<br>….<br>0xF：32768($2^{15}$)Hz(脉冲宽度：二进制回转模式下 50%占空比)<br>**注意：** 如果选择的是十进制回转模式，则建议仅使用 PPSOFC=0。 |

### 27.4.45. DMA 总线控制寄存器(ENET_DMA_BCTL)

地址偏移：0x1000
复位值：0x0002 0101

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:27 | 保留 | 必须保持复位值。 |
| 26 | MB | 混合传输位<br>0：AHB 主接口仅传输小于或等于 16 固定长度的传输<br>1：AHB 主接口将以 INCR 传输大于 16 长度的传输<br>**注意：** MB 和 FB 位应当且必须只有其中一位为'1'。 |
| 25 | AA | 地址对齐<br>0：关闭传输地址对齐功能<br>1：使能传输地址对齐，如果 FB 位为'1'，AHB 接口对齐所有连续传输至起始地址的 LS 位(位 1 到位 0)。如果 FB 位为'0'，除第一次 AHB 访问的地址(访问数据缓存的起始地址)不对齐，后续的传输与地址均对齐。 |
| 24 | FPBL | 4×PGBL 模式<br>0：PGBL 值(位[22:17]和位[13:8])作为 DMA 传输长度值<br>1：PGBL 值(位[22:17]和位[13:8])乘以 4 作为 DMA 传输长度值 |
| 23 | UIP | 使用分散 PGBL<br>0：PGBL 值(位[13:8])对 DMA 接收和发送控制器都有效<br>1：RXDP[5:0]位用于 RxDMA 的传输长度值，PGBL[5:0]位用于 TxDMA 的传输长度值。 |
| 22:17 | RXDP[5:0] | RxDMA PGBL 位<br>如果 UIP=0，则这些位无效。仅当 UIP=1 时，这些位定义了一次 DMA 转发的最大数据传输次数。<br>0x01：最大数据传输次数为 1<br>0x02：最大数据传输次数为 2<br>0x04：最大数据传输次数为 4<br>0x08：最大数据传输次数为 8<br>0x10：最大数据传输次数为 16<br>0x20：最大数据传输次数为 32<br>其他：保留。 |

| | | |
|---|---|---|
| 16 | FB | 固定传输位<br>0：AHB 在连续传输时，只用 SINGLE 和 INCR 数据传输操作。<br>1：AHB 在连续传输时，用 SINGLE，INCR4，INCR8 和 INCR16 数据传输操作。<br>**注意：** MB 和 FB 位应当且必须只有其中一位为'1'。 |
| 15:14 | RTPR[1:0] | 接收发送优先级比率<br>这些位表示 RxDMA 和 TxDMA 之间的访问优先级比率。<br>0x0：RxDMA：TxDMA = 1：1<br>0x1：RxDMA：TxDMA = 2：1<br>0x2：RxDMA：TxDMA = 3：1<br>0x3：RxDMA：TxDMA = 4：1<br>**注意：** 该位只在 DMA 仲裁模式为循环模式(DAB=0)时有效。 |
| 13:8 | PGBL[5:0] | 可编程的数据传输长度位<br>这些位定义了一次 DMA 转发的最大数据传输次数。如果 UIP=1，则这些位仅用于 TxDMA 传输。如果 UIP=0 时，则这些位同时用于 TxDMA 和 RxDMA 传输。<br>0x01：最大数据传输次数为 1<br>0x02：最大数据传输次数为 2<br>0x04：最大数据传输次数为 4<br>0x08：最大数据传输次数为 8<br>0x10：最大数据传输次数为 16<br>0x20：最大数据传输次数为 32<br>其他：保留。 |
| 7 | DFM | 描述符模式<br>0：常规描述符模式<br>1：增强描述符模式 |
| 6:2 | DPSL[4:0] | 描述符跳跃长度<br>这些位仅对于环模式的两个描述符有效，定义了两个无链接的描述符之间从当前描述符的结尾到下一个描述符开头的地址差值，单位为字(32 位)。若 DPSL 域为 0 则 DMA 认为描述符是相邻地连续排列的。 |
| 1 | DAB | DMA 仲裁位<br>该位指示了 TxDMA 和 RxDMA 之间的仲裁模式。<br>0：根据 RTPR 位的值以循环方式仲裁<br>1：固定模式，接收的优先级高于发送 |
| 0 | SWR | 软件复位<br>在所有时钟域的复位操作完成之后，该位将由硬件清零。<br>**注意：** 在写任何 MAC 的寄存器前，应当确保该位为'0'。<br>0：MAC 内部寄存器正常工作<br>1：复位 MAC 所有内核寄存器 |

### 27.4.46. DMA 发送查询使能寄存器(ENET_DMA_TPEN)

地址偏移：0x1004

复位值：0x0000 0000

该寄存器用于TxDMA对发送描述符列表的查询。TxDMA通常因为发送帧的数据下溢错误或者描述符被CPU占有(DAV=0)而进入暂停状态。可以对该寄存器写任意值使能发送查询。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | TPE[31:0] | 发送查询使能位<br>对这些位写任意值，DMA 使能发送查询，将查询当前描述符(描述符地址在ENET_DMA_CTDADDR 寄存器中)是否被 CPU 占有。如果不是(DAV=1)，则描述符可用，DMA 退出暂停状态并恢复工作。如果是(DAV=0)，则 TxDMA 回到暂停状态，并把 ENET_DMA_STAT 的位 TBU 置'1'。 |

## 27.4.47. DMA 接收查询使能寄存器(ENET_DMA_RPEN)

地址偏移：0x1008
复位值：0x0000 0000

该寄存器用于RxDMA对接收描述符列表的查询。对该寄存器写任意值可以使能接收查询。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | RPE[31:0] | 接收查询使能位<br>对这些位写任意值，DMA 使能接收查询，将查询当前描述符(描述符地址在ENET_DMA_CRDADDR 寄存器中)是否被 CPU 占有。如果不是(DAV=1)，则描述符可用，DMA 退出暂停状态并恢复工作。如果是(DAV=0)，则 TxDMA 回到暂停状态，并把 ENET_DMA_STAT 的位 RBU 置'1'。 |

### 27.4.48. DMA 接收描述符列表地址寄存器(ENET_DMA_RDTADDR)

地址偏移：0x100C
复位值：0x0000 0000

接收描述符列表寄存器指向接收描述符队列的开始。描述符队列位于物理内存，并且其地址必须字对齐。只有在接收停止的时候，才允许写该寄存器。在开启RxDMA接收流程之前，必须正确配置该寄存器。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | SRT[31:0] | 接收队列基址<br>这些位包含了接收描述符队列第一个描述符的地址。SRT[1:0]的取值默认为'0'， 因此 SRT[1:0]这两个最低位是只读的。 |

### 27.4.49. DMA 发送描述符列表地址寄存器(ENET_DMA_TDTADDR)

地址偏移：0x1010
复位值：0x0000 0000

该寄存器指向发送描述符队列的起始。描述符队列位于物理内存，并且其地址必须字对齐。只有在发送停止的时候，才允许写该寄存器。在开启TxDMA发送流程之前，必须正确配置该寄存器。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | STT[31:0] | 发送队列基址<br>这些位包含了发送描述符列表第一个描述符的地址。STT[1:0]的取值默认为'0'，因此 STT[1:0]这两个最低位是只读的。 |

### 27.4.50. DMA 状态寄存器(ENET_DMA_STAT)

地址偏移：0x1014
复位值：0x0000 0000

该寄存器表示DMA的状态位。读ENET_DMA_STAT寄存器并不能清除其中的标志位。对寄存器位[16:0](除保留位)需要写'1'才能清除，而写'0'是无效的。通过设置ENET_DMA_INTEN寄存器里的相应位，可以屏蔽位[16:0]触发的中断。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | TST | WUM | MSC | 保留 | EB[2:0] | | | TP[2:0] | | | RP[2:0] | | | NI |
| | | r | r | r | | r | | | r | | | r | | | rc_w1 |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| AI | ER | FBE | 保留 | | ET | RWT | RPS | RBU | RS | TU | RO | TJT | TBU | TPS | TS |
| rc_w1 | rc_w1 | rc_w1 | | | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:30 | 保留 | 必须保持复位值。 |
| 29 | TST | 时间戳触发状态<br>该位指示发生了一个时间戳中断事件。通过清除 TMST 标志，可以清零该位。当该位置'1'时，如果相应中断屏蔽位复位，则产生中断。<br>0：未发生时间戳中断事件<br>1：发生了时间戳中断事件 |
| 28 | WUM | WUM 状态<br>该位指示发生了一个 WUM 事件。当两个事件触发源状态都被清除时，可以清零该位。当该位置'1'时，如果相应中断屏蔽位复位，则产生中断。<br>0：WUM 模块未发生中断事件<br>1：WUM 模块发生了中断事件 |
| 27 | MSC | MSC 状态<br>该位指示发生了一个 MSC 事件。当所有事件触发源状态都被清除时，可以清零该位。当该位置'1'时，如果相应中断屏蔽位复位，则产生中断。<br>0：MSC 模块未发生中断事件<br>1：MSC 模块发生了中断事件 |
| 26 | 保留 | 必须保持复位值。 |
| 25:23 | EB[2:0] | 错误位状态<br>当 FBE=1 时，这些位将对 AHB 总线上的总线响应错误进行错误类型解析。<br>EB[0]<br>1：TxDMA 传输数据时出错<br>0：RxDMA 传输数据时出错<br>EB[1]<br>1：读数据转发时出错 |

| | | |
|---|---|---|
| | | 0：写数据转发时出错<br>EB[2]<br>1：访问描述符时出错<br>0：访问数据缓存时出错 |
| 22:20 | TP[2:0] | 发送流程状态<br>这些位表示 TxDMA 的状态。<br>0x0：停止，接到复位或者停止发送命令。<br>0x1：运行，正在取发送描述符。<br>0x2：运行，正在等待状态信息。<br>0x3：运行，正在读取内存中数据并存入 TxFIFO 中。<br>0x4，0x5：保留。<br>0x6：暂停，发送描述符不可用或发送缓存数据下溢。<br>0x7：运行，正在关闭发送描述符。 |
| 19:17 | RP[2:0] | 接收流程状态<br>这些位表示 RxDMA 的状态。<br>0x0：停止，接到复位或者停止接收命令。<br>0x1：运行，正在取接收描述符。<br>0x2：保留。<br>0x3：运行，正在等待接收数据包。<br>0x4：暂停，接收描述符不可用。<br>0x5：运行，正在关闭接收描述符。<br>0x6：保留。<br>0x7：运行，正在把接收到数据包从 RxFIFO 转发到内存中。 |
| 16 | NI | 正常中断汇总<br>该位是下列位在相应中断使能位(ENET_DMA_INTEN 寄存器)使能了的情况下，其各取值的逻辑或：<br>TS：发送中断<br>TBU：发送缓存不可用<br>RS：接收中断<br>ER：提前接收中断<br>**注意：**该位置'1'后，只有把造成该位置'1'的位清'0'(写'1')，才能把该位清'0'。 |
| 15 | AI | 异常中断汇总<br>该位是下列位在相应中断使能位(ENET_DMA_INTEN 寄存器)使能了的情况下，其各取值的逻辑或：<br>TPS：发送流程停止<br>TJT：发送 Jabber 超时<br>RO：RxFIFO 上溢<br>TU：发送数据下溢<br>RBU：接收缓存不可用<br>RPS：接收流程停止<br>RWT：接收看门狗超时<br>ET：提前发送中断 |

| | | |
|---|---|---|
| | | FBE：总线致命错误<br>**注意：**该位置'1'后，只有把造成该位置'1'的位清'0'(写'1')，才能把该位清'0'。 |
| 14 | ER | 提前接收状态<br>在接收中断位 RS 置'1'时，该位自动清'0'。<br>0：未接收到帧数据<br>1：接收到的数据帧已由 DMA 填满了第一个缓存 |
| 13 | FBE | 总线致命错误状态<br>该位指示发生了一个 AHB 接口响应错误，其错误类型可以由 EB[2:0]位进行解释。<br>0：未发生总线错误<br>1：发生了总线错误，相应的 DMA 控制器停止所有操作。 |
| 12:11 | 保留 | 必须保持复位值。 |
| 10 | ET | 提前发送状态<br>0：发送的帧还未完全传输到 TxFIFO 中<br>1：发送的帧已经完全传输到 TxFIFO 中 |
| 9 | RWT | 接收看门狗超时状态<br>0：接收到的帧长度小于 2048 字节<br>1：接收到的帧长度超过 2048 字节 |
| 8 | RPS | 接收流程停止状态<br>0：接收流程未停止<br>1：接收流程进入停止状态 |
| 7 | RBU | 接收缓存不可用状态<br>0：下一个接收描述符的 DAV 位为'1'<br>1：下一个接收描述符的 DAV 位为'0'，RxDMA 进入暂停状态。 |
| 6 | RS | 接收状态<br>0：帧接收未完成<br>1：帧接收完成 |
| 5 | TU | 发送数据下溢状态<br>0：未发生发送数据下溢错误<br>1：发送帧的过程中发生数据下溢，同时发送进入暂停状态。 |
| 4 | RO | 接收上溢状态<br>0：未发生接收数据上溢错误<br>1：接收帧的过程中发生上溢错误。如果已有一部分帧数据转发到内存，则设置接收描述符 0 的上溢错误位 OERR 为'1'。 |
| 3 | TJT | 发送 Jabber 超时状态<br>0：未发生发送 Jabber 定时器超时事件<br>1：发送 Jabber 定时器超时。此时中止发送进程并进入停止状态，同时设置发送描述符 0 的 Jabber 超时位 JT 为'1'。 |
| 2 | TBU | 发送缓存不可用状态 |

| | | |
|---|---|---|
| | | 0：下一个发送描述符的 DAV 位为'1'<br>1：下一个发送描述符的 DAV 位为'0'，TxDMA 进入暂停状态。 |
| 1 | TPS | 发送流程停止状态<br>0：发送未停止<br>1：发送停止 |
| 0 | TS | 发送状态<br>该位仅在发送描述符 0 中 LSG 和 INTC 位都置位时，才可被置位。<br>0：当前帧发送未完成<br>1：当前帧发送完成 |

## 27.4.51. DMA 控制寄存器(ENET_DMA_CTL)

地址偏移：0x1018
复位值：0x0000 0000

该寄存器设定了接收和发送的工作模式和命令。在整个DMA的初始化流程中，应当最后写该寄存器。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | DTCERFD | RSFD | DAFRF | 保留 | | TSFD | FTF | 保留 | | | TTHC[2] |
| | | | | | rw | rw | rw | | | rw | rs | | | | rw |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TTHC[1:0] | | STE | 保留 | | | | | FERF | FUF | 保留 | RTHC[1:0] | | OSF | SRE | 保留 |
| rw | | rw | | | | | | rw | rw | | rw | | rw | rw | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:27 | 保留 | 必须保持复位值。 |
| 26 | DTCERFD | 不丢弃 TCP/IP 校验和错误帧<br>0：当 FERF 位为'0'，MAC 丢弃所有有错的帧。<br>1：当 FERF 位为'1'，接收到帧仅有校验和错误时，MAC 不会丢弃该帧。 |
| 25 | RSFD | 接收存储转发<br>0：RxFIFO 工作在直通模式，转发阈值由 RTHC 位决定。<br>1：RxFIFO 工作在存储转发模式，只有在帧完整写入 RxFIFO 后，RxDMA 才会把它转发给应用程序，此时 RTHC 位的取值会被忽略。 |
| 24 | DAFRF | 不清空接收帧<br>0：当接收描述符不可用时，RxDMA 就清空 RxFIFO 里的接收帧。<br>1：RxDMA 不清空接收帧，即使接收描述符不可用。 |
| 23:22 | 保留 | 必须保持复位值。 |
| 21 | TSFD | 发送存储转发<br>0：TxFIFO 工作在直通模式，发送阈值由 ENET_DMA_CTL 寄存器中的 TTHC 位决 |

| | | |
|---|---|---|
| | | 定。<br>1：TxFIFO 工作在存储转发模式，只有在帧完整写入 TxFIFO 后，MAC 才会把其发送出去，TTHC 位的取值会被忽略。<br>**注意：**在发送处于停止状态时，可以修改该位。 |
| 20 | FTF | 清空 TxFIFO<br>当此位为 1 时，TxFIFO 控制逻辑电路被复位到初始状态，TxFIFO 里所有的数据被清空/丢失。在清空操作完成后该位被自动清'0'。<br>**注意：**在该位为'0'之前，不允许写 ENET_DMA_CTL 寄存器。 |
| 19:17 | 保留 | 必须保持复位值。 |
| 16:14 | TTHC[2:0] | 发送阈值控制<br>这三位控制直通模式下 TxFIFO 的阈值。<br>当 TSFD=1 时，忽略这些位。<br>0x0：64<br>0x1：128<br>0x2：192<br>0x3：256<br>0x4：40<br>0x5：32<br>0x6：24<br>0x7：16 |
| 13 | STE | 开始/停止发送<br>0：在发送完当前帧或 TxDMA 进入暂停状态后，发送进程进入停止模式。保存发送描述符队列里下一发送描述符的位置，在传输重新开始时，这个描述符就变成当前描述符。<br>1：TxDMA 进入运行状态。DMA 获取当前发送描述符，发送帧描述符可从 ENET_DMA_TDTADDR 基址获取，若上一次发送为停止状态，则也可从发送描述符队列的指针位置获取。如果当前描述符的 DAV 位为'0'，则 TxDMA 进入暂停状态，并设置 TBU 位为'1'。如果在未设置完其他 DMA 寄存器的情况下就置位该位，则会引起不可预料的后果。 |
| 12:8 | 保留 | 必须保持复位值。 |
| 7 | FERF | 转发错误帧<br>0：当 RxFIFO 工作于直通模式(RSFD=0)时，如果在将 RxFIFO 数据转发到内存之前检测到了帧错误(CRC 错误、冲突错误、校验和错误、看门狗超时、溢出)，则 RxFIFO 会丢弃这个错误的帧。但如果在将 RxFIFO 数据转发到内存之后才检测到了帧错误，则就不会丢弃该帧。当 RxFIFO 工作于存储转发模式时，在接收过程中一旦检测到帧错误，就会丢弃该帧。<br>1：除了过短帧外的所有帧都会转发给 DMA |
| 6 | FUF | 转发长度不够的"好"帧<br>0：RxFIFO 丢弃所有长度小于 64 字节的帧，但如果在检测到过短帧之前，帧已开始转发该帧给应用程序(例如在直通模式下，帧长小于接收阈值)，则将转发整个帧。 |

| | | |
|---|---|---|
| | | 1：RxFIFO 把长度不够的"好"帧(帧长小于 64 字节但没有错误)转发给应用程序 |
| 5 | 保留 | 必须保持复位值。 |
| 4:3 | RTHC[1:0] | 接收阈值控制<br>这两位设置了在直通模式下 RxFIFO 的阈值。<br>**注意：**只有在 RSFD 位(位 21)为'0'时，这些位才有效。在 RSFD 位为'1'时忽略这些位。<br>0x0：64<br>0x1：32<br>0x2：96<br>0x3：128 |
| 2 | OSF | 操作第二帧<br>0：TxDMA 仅在接收到前一个帧的发送状态信息后，才开始发送下一个帧的数据。<br>1：TxDMA 在前一帧数据全部存入到 TxFIFO 之后，在接收到前一个帧的发送状态信息前，就开始发送下一个帧的数据。 |
| 1 | SRE | 开始/停止接收<br>0：在转发完当前接收帧后，RxDMA 进入停止模式。保存接收描述符队列里下一接收描述符的位置，在传输重新开始时，这个描述符就变成当前描述符。只有在接收运行时或接收暂停时，可"停止接收"。<br>1：把接收进程置为运行状态，DMA 检查接收描述符队列的当前位置，用来处理下一个收到的帧。接收帧描述符可从 ENET_DMA_RDTADDR 基址获取，若上一次接收为停止状态，则也可从接收描述符队列的指针位置获取。如果获取的描述符 DAV=0，那么接收进程进入暂停状态，并设置 RBU 位为'1'。只有在接收停止时或接收暂停时，"开始接收"命令才有效。在未设置完所有其他 DMA 寄存器之前发出"开始接收"命令，会引起不可预料的后果。 |
| 0 | 保留 | 必须保持复位值。 |

## 27.4.52. DMA 中断使能寄存器(ENET_DMA_INTEN)

地址偏移：0x101C
复位值：0x0000 0000

该寄存器可以使能ENET_DMA_STAT寄存器反映的中断。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| AIE | ERIE | FBEIE | 保留 | | ETIE | RWTIE | RPSIE | RBUIE | RIEN | TUIE | ROIE | TJTIE | TBUIE | TPSIE | TIE |
| rw | rw | rw | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

| 位/位域 | 名称 | 描述 |
|---|---|---|

| | | |
|---|---|---|
| 31:17 | 保留 | 必须保持复位值。 |
| 16 | NIE | 正常中断汇总使能<br>0：屏蔽正常中断<br>1：使能正常中断<br>该位使能下列位：<br>TS：发送中断<br>TBU：发送缓存不可用<br>RS：接收中断<br>ER：提前接收中断 |
| 15 | AIE | 异常中断汇总使能<br>0：屏蔽异常中断<br>1：使能异常中断<br>该位使能下列位：<br>TPS：发送流程停止<br>TJT：发送 Jabber 超时<br>RO：RxFIFO 上溢<br>TU：发送下溢<br>RBU：接收缓存不可用<br>RPS：接收流程停止<br>RWT：接收看门狗超时<br>ET：提前发送中断<br>FBE：总线致命错误 |
| 14 | ERIE | 提前接收中断使能<br>0：屏蔽提前接收中断<br>1：使能早接收中断 |
| 13 | FBEIE | 总线致命错误中断使能<br>0：屏蔽总线致命错误中断<br>1：使能总线致命错误中断 |
| 12:11 | 保留 | 必须保持复位值。 |
| 10 | ETIE | 提前发送中断使能<br>0：屏蔽提前发送中断<br>1：使能提前发送中断 |
| 9 | RWTIE | 接收看门狗超时中断使能<br>0：屏蔽接收看门狗超时中断<br>1：使能接收看门狗超时中断 |
| 8 | RPSIE | 接收流程停止中断使能<br>0：屏蔽接收流程停止中断<br>1：使能接收流程停止中断 |
| 7 | RBUIE | 接收缓存不可用中断使能<br>0：屏蔽接收缓存不可用中断 |

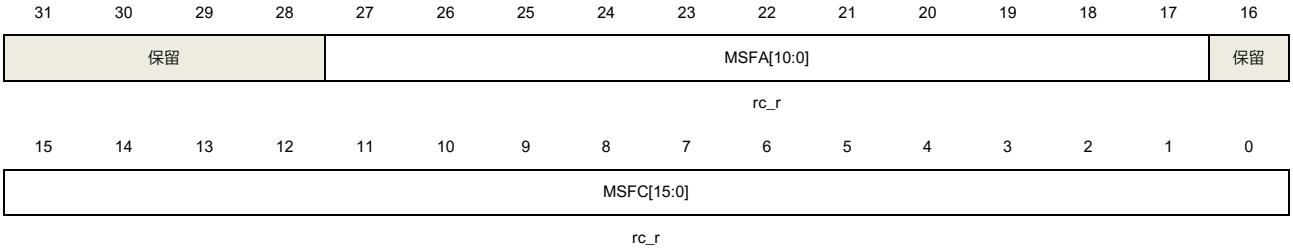
| | | |
|---|---|---|
| | | 1：使能接收缓存不可用中断 |
| 6 | RIE | 接收中断使能<br>0：屏蔽接收中断<br>1：使能接收中断 |
| 5 | TUIE | 发送下溢中断使能<br>0：屏蔽发送数据下溢中断<br>1：使能发送下溢中断 |
| 4 | ROIE | 接收上溢中断使能<br>0：屏蔽接收上溢中断<br>1：使能接收上溢中断 |
| 3 | TJTIE | 发送 Jabber 超时中断使能<br>0：屏蔽发送 Jabber 超时中断<br>1：使能发送 Jabber 超时中断 |
| 2 | TBUIE | 发送缓存不可用中断使能<br>0：屏蔽发送缓存不可用中断<br>1：使能发送缓存不可用中断 |
| 1 | TPSIE | 发送流程停止中断使能<br>0：屏蔽发送流程停止中断<br>1：使能发送流程停止中断 |
| 0 | TIE | 发送中断使能<br>0：屏蔽发送中断<br>1：使能发送中断 |

### 27.4.53. DMA 丢失帧和缓存溢出计数器寄存器(ENET_DMA_MFBOCNT)

地址偏移：0x1020
复位值：0x0000 0000

DMA有2个计数器，用来统计接收过程中丢失帧的数目。可通过读本寄存器来获取计数器的当前值。这个计数器通常用作故障诊断。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。




| 位/位域 | 名称 | 描述 |
|---|---|---|

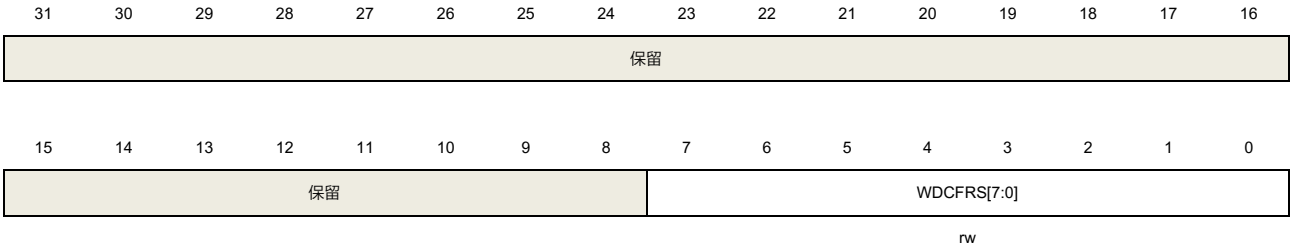
| | | |
|---|---|---|
| 31:28 | 保留 | 必须保持复位值。 |
| 27:17 | MSFA[10:0] | 应用程序丢失的帧<br>这些位指示了 RxFIFO 丢失的帧数目。 |
| 16 | 保留 | 必须保持复位值。 |
| 15:0 | MSFC[15:0] | 控制器丢失的帧<br>这些位表示了由于 MCU 的接收缓存不可用而导致 RxDMA 丢失的帧的数目。每当 DMA 清空一个输入帧时，这个计数器加 1。 |

### 27.4.54. DMA 接收状态看门狗计数器寄存器(ENET_DMA_RSWDC)

地址偏移：0x1024
复位值：0x0000 0000

向该寄存器写入一个值，可在延时一段可配置的时间之后，使能针对RS位(ENET_DMA_STAT寄存器)的看门狗定时器。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



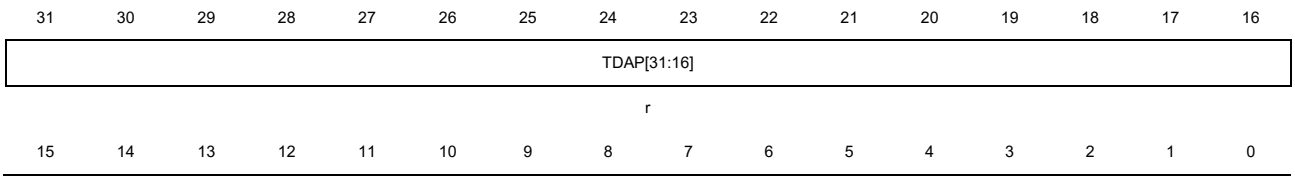

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7:0 | WDCFRS[7:0] | 接收状态看门狗计数器<br>这些位仅在接收描述符 1 的 DINTC 位置位时有效。当 DINTC=1 时，并接收到一个帧，则 RS 位会在接收完毕后延时 WDCFRS*256 个 HCLK 时钟周期再置位。 |

### 27.4.55. DMA 当前发送描述符地址寄存器(ENET_DMA_CTDADDR)

地址偏移：0x1048
复位值：0x0000 0000

当前发送描述符寄存器指向TxDMA正在读取的发送描述符起始地址。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。



| TDAP[15:0] |
|---|

r

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | TDAP[31:0] | 发送描述符地址指针<br>这些位在复位时清'0'，由 TxDMA 在操作过程中自动更新。 |

## 27.4.56. DMA 当前接收描述符地址寄存器(ENET_DMA_CRDADDR)

地址偏移：0x104C
复位值：0x0000 0000

当前接收描述符寄存器指向RxDMA正在读取的接收描述符起始地址。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | RDAP[31:0] | 接收描述符地址指针<br>这些位在复位时清'0'，由 RxDMA 在操作过程中自动更新。 |

## 27.4.57. DMA 当前发送缓存地址寄存器(ENET_DMA_CTBADDR)

地址偏移：0x1050
复位值：0x0000 0000

该寄存器指向TxDMA正在读取的发送缓存的地址。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | TBAP[31:0] | 发送缓存地址指针<br>这些位在复位时清'0'，由 TxDMA 在工作过程中更新。 |

### 27.4.58. DMA 当前接收缓存地址寄存器(ENET_DMA_CRBADDR)

地址偏移：0x1054
复位值：0x0000 0000

该寄存器地址指向RxDMA正在读取的接收缓存地址。

该寄存器可以按字节(8位)、半字(16位)或字(32位)访问。

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

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | RBAP[31:0] | 接收缓存地址指针<br>这些位在复位时清'0'，由 RxDMA 在工作过程中更新。 |

# 28. 通用串行总线全速接口(USBFS)

## 28.1. 概述

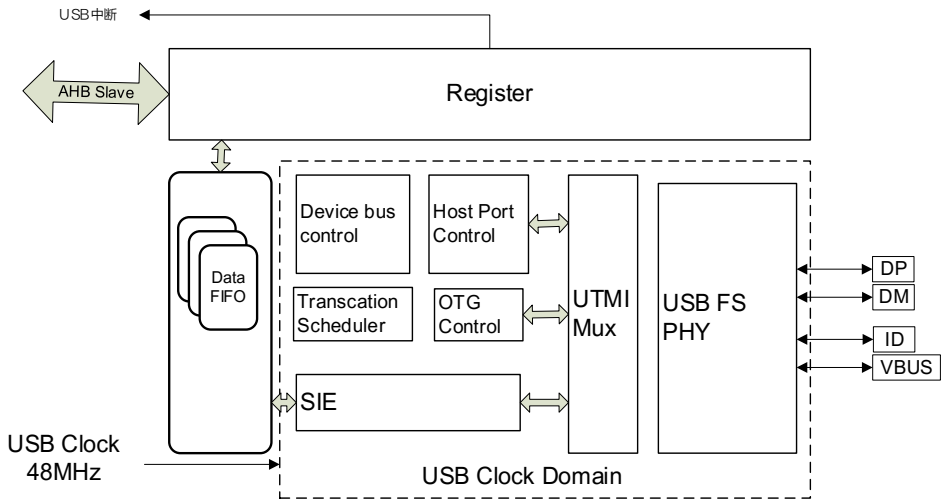
USB全速(USBFS)控制器为便携式设备提供了一套USB通信解决方案。USBFS不仅提供了主机模式和设备模式，也提供了遵循HNP(主机协商协议)和SRP(会话请求协议)的OTG模式。USBFS包含了一个内部的全速USB PHY，并且不再需要外部PHY芯片。USBFS可提供USB 2.0协议所定义的所有四种传输方式(控制传输、批量传输、中断传输和同步传输)。

## 28.2. 主要特性

- 支持USB 2.0全速(12Mb/s)/低速(1.5Mb/s)主机模式；
- 支持USB 2.0全速(12Mb/s)设备模式；
- 支持遵循HNP(主机协商协议)和SRP(会话请求协议)的OTG协议；
- 支持所有的4种传输方式：控制传输、批量传输、中断传输和同步传输；
- 在主机模式下，包含USB事务调度器，用于有效地处理USB事务请求；
- 包含一个1.25KB的FIFO RAM；
- 在主机模式下，支持8个通道；
- 在主机模式下，包含2个发送FIFO(周期性发送FIFO和非周期性发送FIFO)和1个接收FIFO(由所有的通道共享)；
- 在设备模式下，包含4个发送FIFO(每个IN端点一个发送FIFO)和1个接收FIFO(由所有的OUT端点共享)；
- 在设备模式下，支持4个OUT端点和4个IN端点；
- 在设备模式下，支持远程唤醒功能；
- 包含一个支持USB协议的全速USB PHY；
- 在主机模式下，SOF的时间间隔可动态调节；
- 可将SOF脉冲输出到PAD；
- 可检测ID引脚电平和VBUS电压；
- 在主机模式或者OTG A设备模式下，需要外部部件为连接的USB设备提供电源。

## 28.3. 结构框图

图 28-1. USBFS 结构框图




## 28.4. 信号线描述

表 28-1. USBFS 信号线描述

| I/O 端口 | 类型 | 描述 |
|---|---|---|
| VBUS | 输入 | 总线电源端口 |
| DM | 输入/输出 | 差分信号 - 端口 |
| DP | 输入/输出 | 差分信号 + 端口 |
| ID | 输入 | USB 识别：微连接器识别接口 |

## 28.5. 功能描述

### 28.5.1. USBFS 时钟及工作模式

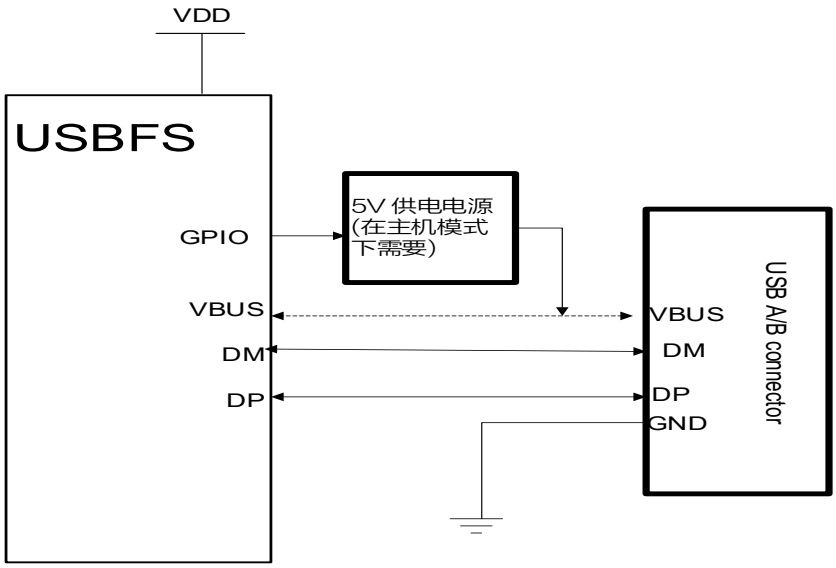
USBFS可以作为一个主机、一个设备或者一个DRD(双角色设备),并且包含一个内部全速PHY。

内部PHY支持主机模式下的全速和低速、设备模式下全速以及具备HNP和SRP的OTG协议。USBFS所使用的USB时钟需要配置为48MHz。该48MHz USB时钟从系统内部时钟产生，并且其时钟源和分频器需要在RCU模块中配置。

上拉或下拉电阻已经集成在内部全速 PHY 的内部，并且 USBFS 可根据当前模式(主机、设备或 OTG 模式)和连接状态进行自动选择。一个利用内部全速 PHY 的典型连接示意图如*图 28-2.*

*在主机或设备模式下连接示意图*所示。

**图 28-2. 在主机或设备模式下连接示意图**



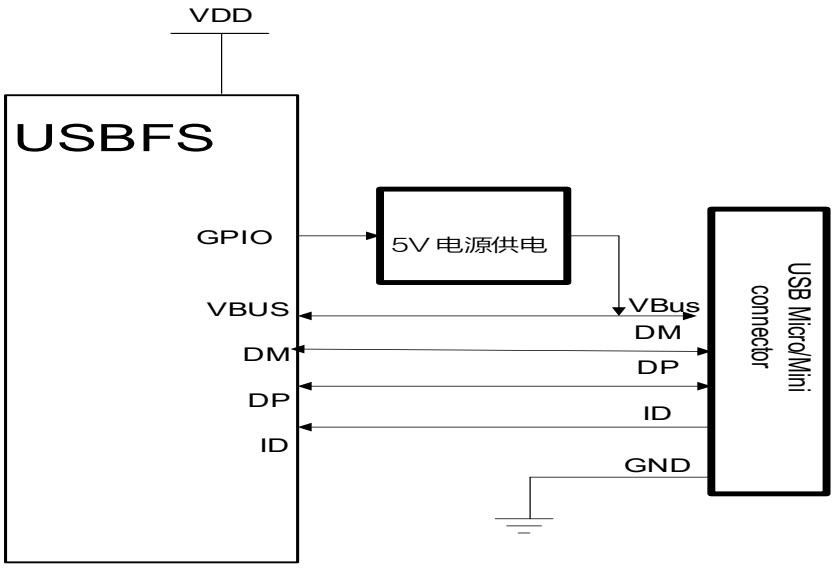

当USBFS工作在主机模式下时(FHM控制位置位、PDM控制位清除)，VBUS为USB协议所定义的5V电源检测引脚。内部PHY不能提供5V VBUS电源，仅在VBUS信号线上具有电压比较器和充电放电电路。所以，如果应用需要提供VBUS电源，那么则需要一个外部的供电电源IC。在主机模式下，USBFS和USB连接头之间的VUBS连接可以被忽略，这是由于USBFS并不检测VBUS引脚的电平状态，并假定5V供电电源一直存在。

当USBFS工作在设备模式下时(FHM控制位清除、FDM控制位置位)，VBUS检测电路由USBFS_GCCFG寄存器中的VBUSIG控制位所配置。所以，如果设备不需要检测VBUS引脚电压，可以置位VBUSIG控制位，并可释放VBUS引脚作为其他用途。否则，VBUS引脚的连接不能够被忽略，并且USBFS需要不断的检测VBUS电平状态，一旦VBUS电压降至所需有效值以下，需要立即关闭DP信号线上的上拉电阻，从而产生一个断开状态。

OTG 模式连接示意图如*图 28-3. OTG 模式下连接示意图*所示。当 USBFS 工作在 OTG 模式下时，USBFS_GUSBCS 寄存器内的 FHM、FDM 控制位应该被清除。在这种模式下，USBFS 需要以下四个引脚：DM、DP、VBUS 和 ID，并且需要使用若干个电压比较器检测这些引脚的电压。USBFS 也包含 VBUS 充电和放电电路，用以完成 OTG 协议中所描述的 SRP 请求。OTG A 设备或 B 设备由 ID 引脚的电平状态所决定。在实现 HNP 协议的过程中，USBFS 控

制上拉和下拉电阻。

**图 28-3. OTG 模式下连接示意图**

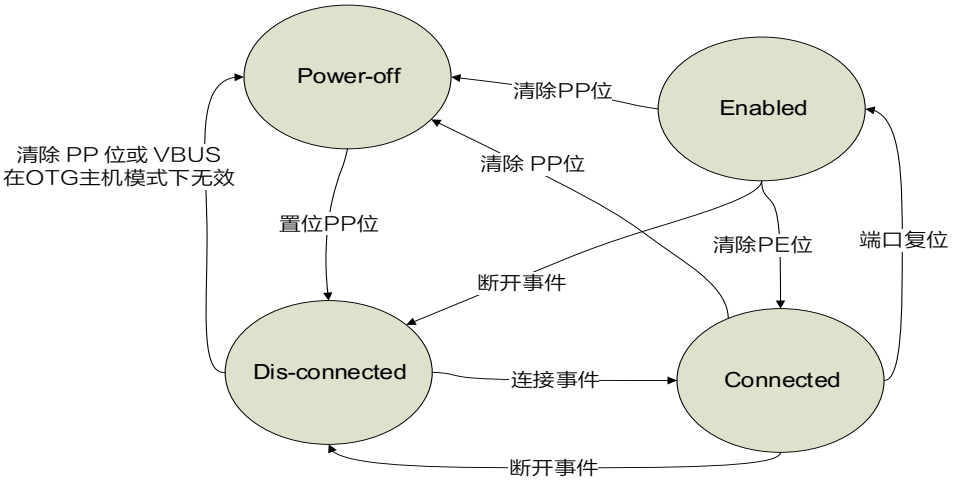



## 28.5.2. USB 主机功能

### USB主机端口状态

主机应用可以通过USBFS_HPCS寄存器控制USB端口状态。系统初始化之后，USB端口保持掉电状态。通过软件置位PP控制位后，USB PHY(内部或外部)将被上电，并且USB端口变为断开状态。检测到连接后，USB端口变为连接状态。在USB总线上产生一个复位后，USB端口将变为使能状态。

**图 28-4. 主机端口状态转移图**




### 连接、复位和速度识别

作为USB主机，在检测到一个连接事件后，USBFS会为应用触发一个连接标志；同样，若检测

到一个断开事件后，将会触发一个断开标志。

PRST控制位用于实现USB复位序列。应用可以置位该控制位以启动一个USB复位序列，或者清除该控制位以结束USB复位序列。仅当端口在连接或使能状态时，该控制位有效。

USBFS在对设备连接和复位时执行速度检测，并且速度检测的结果会反馈在USBFS_HPCS寄存器的PS[1:0]标志位中。USBFS从DM或DP的电平状态决定设备速度。就像USB协议中所描述的那样，全速设备上拉DP信号线，而低速设备上拉DM信号线。

**挂起和复位**

USBFS支持挂起和复位状态，当USBFS端口在使能状态时，向USBFS_HPCS寄存器的PSP控制位写1，USBFS会进入到挂起状态。在挂起状态，USBFS停止在USB总线上发送SOF，并且这样会让连接的USB设备在3ms后进入到挂起状态。应用程序能够置位USBFS_HPCS寄存器中的PREM控制位以启动一个恢复序列，用以唤醒挂起的设备，当清除该控制位可以停止恢复序列。如果主机在挂起状态检测到一个远程唤醒信号，将会置位USBFS_GINTF寄存器的WKUPIF标志位，并且触发USBFS唤醒中断。

**SOF产生器**

在主机模式下，USBFS向USB总线发送SOF令牌包。如USB2.0协议所描述，全速连接下，SOF令牌包每1ms产生一次(由主机控制器或者HUB事务转换器产生)。

每次USBFS进入到使能状态后，它将会按照USB2.0所定义的周期发送SOF令牌包。然而，应用程序可以通过写USBFS_HFT寄存器中的FRI[15:0]位来调整一帧的间隔。FRI控制位定义了在一帧中的USB时钟周期个数，并且应用程序应该基于USBFS所使用的USB时钟频率计算该值。FRT[14:0]位反映了当前帧剩余的时钟周期个数，并且在挂起状态时，该值将停止改变。

USBFS能够在每个SOF令牌包中产生一个脉冲信号，并且将其输出至一个引脚。该脉冲信号长度为12个HCLK周期。如果应用程序希望使用该功能，需要置位USBFS_GCCFG寄存器的SOFOEN控制位，并且配置相应的引脚寄存器为GPIO功能。

**USB通道和事务**

USBFS在主机模式下包含8个独立的通道。每个通道能够与一个USB设备端点通信。通道的传输类型、方向、包长和其他信息都在通道相应的寄存器中配置，例如USBFS_HCHxCTL和USBFS_HCHxLEN寄存器。

USBFS支持所有的四种传输类型：控制、批量、中断和同步。USB2.0协议将这些传输类型划分为两类：非周期性传输(控制和批量)和周期性传输(中断和同步)。基于此，为了有效地进行事务调度，USBFS包含两种请求队列：周期性请求队列和非周期性请求队列。在请求队列上方描述的请求条目可能代表一个USB事务请求或者一个通道操作请求。

若应用程序需要在USB总线上启动一个OUT事务，需要通过AHB寄存器接口向数据FIFO中写入数据包。USBFS硬件会在应用写完整包数据后，自动产生一个事务请求并进入请求队列。

请求队列中的请求条目通过USBFS中的事务控制模块按顺序处理。USBFS通常首先尝试处理周期性请求队列，然后处理非周期性请求队列。

帧起始后，USBFS首先开始处理周期性队列，直到队列为空或者当前周期性请求队列所需时间不够，然后处理非周期性队列。这种做法保证了一帧中周期性传输的带宽。每次USBFS从请求

队列中读取并取出一个请求条目。如果取出的是通道禁用请求，这将直接禁用通道并准备处理下个条目。

如果当前请求是一个事务请求并且USB总线时间能够处理这个请求，USBFS会使用SIE在USB总线上产生该事务。

在当前帧内，当前请求所需的总线时间不足时，如果当前请求为周期性请求，USBFS停止处理该周期性请求队列，并启动处理非周期性请求。如果当前请求为非周期性请求，USBFS会停止处理任何队列，并等待直到当前帧结束。

### 28.5.3. USB 设备功能

**USB设备连接**

在设备模式下，USBFS在初始化后保持掉电状态。利用VBUS引脚上的5V电源连接USB主机后或者置位USBFS_GCCFG寄存器中VBUSIG控制位，USBFS将进入供电状态。USBFS首先打开DP信号线上的上拉电阻，之后主机将会检测到一个连接事件。

**复位和速度识别**

USB主机在检测到设备连接之后，总是会启动一个USB复位序列，并且在设备模式下，检测到USB总线复位事件后，USBFS会为软件触发一个复位中断。

在复位序列后，USBFS将会触发USBFS_GINTF寄存器中的ENUMF中断，并且利用USBFS_DSTAT寄存器内的ES标志位反映当前枚举设备速度，该段位域一直为11(全速)。

如USB2.0协议所需要，USBFS在外设模式下不支持低速。

**挂起和唤醒**

USB总线保持IDLE状态并且数据线3ms无变化，USB设备将会进入挂起状态。当USB设备在挂起状态时，软件能够关闭大部分的时钟以节省电能。USB主机可以通过在USB总线上产生恢复信号，来唤醒挂起的设备。USBFS检测到恢复信号后，将置位USBFS_GINTF寄存器的WKUPIF标志位并且触发USBFS唤醒中断。

在挂起设备模式，USBFS也能够远程唤醒USB总线。软件可以通过置位USBFS_DCTL寄存器的RWKUP控制位来发送一个远程唤醒信号，并且如果USB主机支持远程唤醒，主机会在USB总线上启动发送一个恢复信号。

**软件断开**

USBFS支持软件断开。设备进入到供电状态后，USBFS会打开DP信号线的上拉电阻，并且这样主机会检测到设备连接。然后，软件可以通过置位USBFS_DCTL寄存器中SD控制位进行强制断开。SD控制位置位后，USBFS将会直接关闭上拉电阻。这样，USB主机将会在USB总线上检测到设备断开。

**SOF跟踪**

当USBFS在USB总线上接收到一个SOF令牌包时，将触发一个SOF中断，并且开始利用本地USB时钟计算总线时间。当前帧的帧号将会反应在USBFS_DSTAT寄存器的FNRSOF[13:0]位域中。当USB总线时间达到EOF1或EOF2点(帧结束，在USB2.0协议中描述)，USBFS会触发

USBFS_GINTF寄存器中的EOPFIF中断。软件能够使用这些标志位和寄存器以获得当前总线时间和位置信息。

### 28.5.4. OTG 功能概述

USBFS支持OTG协议1.3中所描述的OTG功能，OTG功能包括SRP和HNP。

**A设备和B设备**

当标准A或微型A插头插入相应的插座时，具有OTG能力的USB设备为A设备。A设备向VBUS供电，并且在会话开始时默认为主机。当标准B、微型B、迷你B插头插入相应的插座或采用一端为标准A插头的不可分离电缆时，具有OTG能力的USB设备为B设备。B设备在会话开始时默认为外设。USBFS使用ID引脚电平状态决定A设备或B设备。ID引脚状态反馈在USBFS_GOTGCS寄存器的IDPS状态位。为了了解A设备和B设备之间传输的详细状态，请参考OTG1.3协议。

**HNP**

主机协商协议(HNP)允许主机功能在两个直接连接的OTG设备之间转换，并且用户不需要为了设备之间通信控制的改变而切换电缆线的连接。典型地，HNP协议是由B设备上的用户或应用启动，HNP只能通过设备上的微型AB插座执行。

一旦OTG设备具有一个微型AB插座，该OTG设备可通过插入的插头类型决定默认为主机或设备(微型A插头插入为主机，微型B插头插入为设备)。通过使用主机协商协议(HNP)，一个默认为外设的OTG设备可以请求成为主机。主机角色切换的过程在下段中描述。此协议使用户不需要为了更改连接设备的角色而切换电缆线的连接。

当USBFS工作在OTG A主机模式时，并且其想放弃主机角色，可以首先置位USBFS_HPCS寄存器的PSP控制位来使USB总线进入挂起状态，然后B设备在3ms后进入挂起状态。如果B设备想要变为主机，软件需要置位USBFS_GOTGCS寄存器的HNPREQ控制位，然后USBFS会开始在总线上执行HNP协议，最后，HNP的结果会反馈在USBFS_GOTGCS寄存器的HNPS状态位。另外，软件总能从USBFS_GINTF寄存器的COPM状态位获取当前设备角色(主机或外设)。

**SRP**

会话请求协议(SRP)允许B设备请求A设备打开VBUS并启动一个会话。该协议允许A设备(或许是电池供电)当总线无活动时通过关闭VBUS以节省电能，并为B设备启动总线活动提供了一种方法。如OTG协议中所描述，OTG设备必须和几个阈值比较VBUS电压，并且将比较结果反馈在USBFS_GOTGCS寄存器的ASV和BSV状态位中。

当USBFS工作在B设备OTG模式时，软件可以通过置位USBFS_GOTGCS寄存器的SRPREQ控制位来启动一个SRP请求，并且如果SRP请求成功，USBFS会在USBFS_GOTGCS寄存器中产生一个成功标志位SRPS。

当USBFS工作在OTG A设备模式且从B设备检测到一个SRP请求时，USBFS将会置位USBFS_GINTF寄存器中的SESIF标志位。软件获取该标志位后，需要准备为VBUS引脚打开5V供电电源。

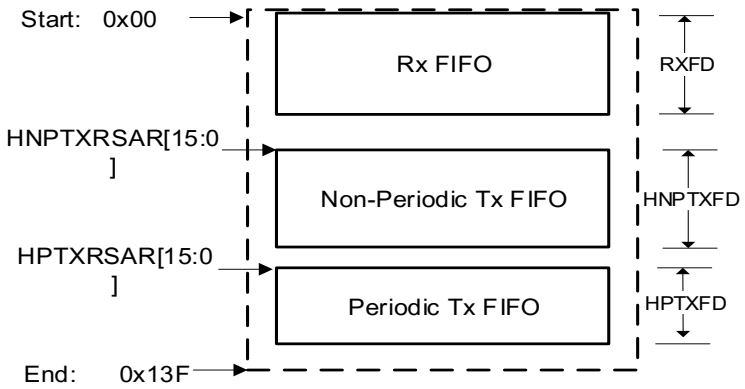
### 28.5.5. 数据 FIFO

USBFS中采用1.25K字节数据FIFO存储包数据，数据FIFO是通过USBFS的内部SRAM实现的。

**主机模式**

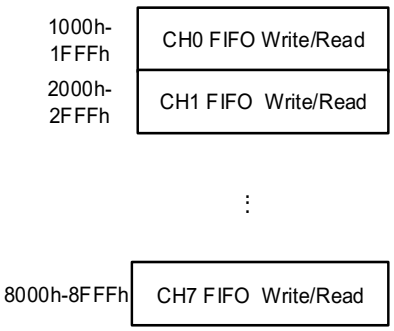
主机模式下，数据 FIFO 空间分为三个部分，分别是：用于接收数据包的 Rx FIFO、用于非周期性发送数据包的非周期性 Tx FIFO 和用于周期性发送数据包的周期性 Tx FIFO。所有的 IN 通道通过共享 Rx FIFO 接收数据。所有的周期性 OUT 通道通过共享周期性 Tx FIFO 来发送数据，所有的非周期性 OUT 通道通过共享非周期性 Tx FIFO 来发送数据。通过寄存器 USBFS_GRFLEN、USBFS_HNPTFLEN 和 USBFS_HPTFLEN，软件可以配置以上数据 FIFO 的大小和起始偏移地址。*图 28-5. 主机模式 FIFO 空间*所描述的是 SRAM 中各 FIFO 的结构，图中的数值是以 32 位为单位。

**图 28-5. 主机模式 FIFO 空间**




USBFS 为程序提供了专有寄存器空间来读写数据 FIFO。*图 28-6. 主机模式 FIFO 访问寄存器映射表*所描述的是数据 FIFO 所访问的寄存器存储空间，图中的数值是以字节为单位寻址。尽管所有的非周期通道共享相同的 FIFO 以及所有的周期通道共享相同的 FIFO，每个通道都拥有它们的 FIFO 访问寄存器空间。对 USBFS 而言，获知当前压入数据包的通道号是非常重要的，通过寄存器 USBFS_GRXTATR/USBFS_GRSTATP 来访问数据包所从属的 Rx FIFO。

**图 28-6. 主机模式 FIFO 访问寄存器映射表**

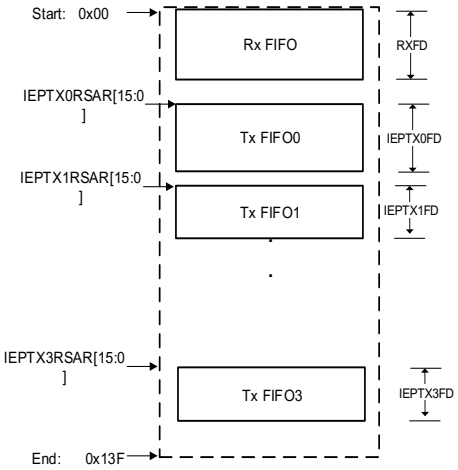



**设备模式**

在设备模式下，数据 FIFO 分为多个部分，其中包含 1 个 Rx FIFO 和 4 个 Tx FIFO，每个 Tx

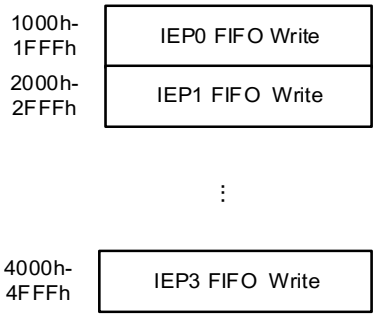
FIFO 对应着一个 IN 端点，所有的 OUT 端点通过共享 Rx FIFO 接收数据包。通过寄存器 USBFS_GRFLEN 和 USBFS_DIEPxTFLEN (x=0…3)，程序可配置数据 FIFO 的大小和起始偏移地址。*图 28-7. 设备模式 FIFO 空间*所描述的是 SRAM 中各 FIFO 的结构，图中的数值是以 32 位为单位。

**图 28-7. 设备模式 FIFO 空间**




USBFS 为程序提供了专有寄存器空间来读写数据 FIFO。*图 28-8. 设备模式 FIFO 访问寄存器映射表*所描述的是数据 FIFO 所访问的寄存器存储空间，图中的数值是以字节为单位寻址。每个端点都拥有它们的 FIFO 访问寄存器空间。通过寄存器 USBFS_GRXTATR/USBFS_GRSTATP 来访问 Rx FIFO。

**图 28-8. 设备模式 FIFO 访问寄存器映射表**




## 28.5.6. 操作手册

该部分描述的是USBFS的操作手册。

### 主机模式

**全局寄存器初始化顺序：**

1. 根据应用的需求，如Tx FIFO的空阈值等，设置寄存器USBFS_GAHBCS，此时，GINTEN位需要保持清零状态。
2. 根据应用的需求，如操作模式(主机、设备或OTG)、某些OTG参数和USB协议，设置寄存

器USBFS_GUSBCS。
3. 根据应用的需求，设置寄存器USBFS_GCCFG。
4. 根据应用的需求，设置寄存器USBFS_GRFLEN、USBFS_HNPTFLEN_DIEP0TFLEN、USBFS_HPTFLEN，配置数据FIFO。
5. 通过设置寄存器USBFS_GINTEN使能模式错误和主机端口中断，置位USBFS_GAHBCS寄存器的GINTEN位使能全局中断。
6. 设置寄存器USBFS_HPCS，置位PP位。
7. 等待设备连接，当设备连接后，触发寄存器USBFS_HPCS的PCD位，然后置位PRST位，执行一次端口复位，等待至少10毫秒后，清除PRST位。
8. 等待USBFS_HPCS寄存器的PEDC中断，然后读取PE位以确认端口被成功地使能，读取PS位以获取连接的设备速度，之后，如果软件需要改变SOF间隔，设置USBFS_HFT寄存器。

**通道初始化和使能顺序：**

1. 根据期望的传输类型、方向、包大小等信息，设置寄存器USBFS_HCHxCTL，在设置期间，要保证位CEN和CDIS保持清除。
2. 设置寄存器USBFS_HCHxINTEN，设置期望的中断使能位。
3. 设置寄存器USBFS_HCHxLEN，PCNT表示一次传输中的包数，TLEN表示一次传输中发送或接收的包数据的总字节数。
   对于OUT通道，如果PCNT为1，单包的大小等于TLEN。如果PCNT大于1，前PCNT-1个包被认定为最大包长度的包，其大小是由寄存器USBFS_HCHxCTL的位MPL所定义。最后一包的大小可通过PCNT、TLEN和MPL计算得到。如果程序想要发出一个零长度的包，应该设定TLEN为0，PCNT位1。
   对于IN通道，因为在IN事务结束之前，程序不知道实际接收的数据大小，程序可将TLEN设定为Rx FIFO所支持的最大值。
4. 置位寄存器USBFS_HCHxCTL中的CEN位以使能通道。

**通道除能顺序：**

程序可以通过同时置位CEN和CDIS除能通道。在寄存器操作后，USBFS将在请求队列中产生一个通道除能请求条目。当这个请求条目到达请求队列的顶部时，USBFS立即进行处理。

对于OUT通道而言，特定的通道将被立即除能。然后，会产生CH标志，USBFS将清除CEN和CDIS位。

对于IN通道而言，USBFS将通道除能状态条目压入Rx FIFO，然后，程序应该处理Rx FIFO非空事件：读和取出该状态条目，然后会产生CH标志，USBFS将清除CEN和CDIS位。

**IN传输操作顺序：**

1. 初始化USBFS全局寄存器。
2. 初始化相应的通道。
3. 使能相应的通道。
4. 通过软件使能IN通道后，USBFS在相应请求队列中生成一个Rx请求条目。
5. 当Rx请求条目到达请求队列的顶部时，USBFS开始执行该请求条目。对于由请求条目所指示的事务而言，如果总线时间足够，USBFS在USB总线上开始IN事务。
6. 当IN事务结束时(收到ACK握手包)，USBFS将接收到的数据包压入Rx FIFO，ACK标志位

被触发，否则，状态标志(NAK)会指示事务结果。

7. 如果步骤5所描述的IN事务完成后，步骤2的PCNT的数值比1大，程序将会返回步骤3，继续接收剩下的数据包。如果步骤5中描述的IN事务没有成功完成，程序将会返回步骤3来再次发送该数据包。
8. 在所有的传输中的所有事务都被成功接收后，USBFS将TF状态条目压入Rx FIFO的最后的数据包的顶部，这样，软件在读取所有接收的数据包后，再读取TF状态条目。USBFS生成TF标志来指示传输成功结束。
9. 除能通道，当通道处于空闲状态，即可为其他传输做准备。

**OUT传输操作顺序：**

1. 初始化USBFS全局寄存器。
2. 初始化及使能相应通道。
3. 将数据包写入通道的Tx FIFO(周期性Tx FIFO或非周期性Tx FIFO)。在所有的数据包都被写入FIFO后，USBFS在相应的请求队列中产生一个Tx请求条目，并且将USBFS_HCHxTLEN中的TLEN值减少，减少的数值等于已写的包大小。
4. 当请求条目到达请求队列的顶部时，USBFS开始执行该请求条目。如果请求条目对应的事务的总线时间足够，USBFS在USB总线上开展OUT事务。
5. 当由请求条目所指示的OUT事务结束时，寄存器USBFS_HCHnTLEN的位PCNT减1。如果该事务完成(收到ACK握手包)，ACK标志位被触发，否则，状态标志(NAK)会指示事务结果。
6. 如果步骤5所描述的OUT事务完成后且步骤2的PCNT的数值比1大，程序将会返回步骤3，继续发送剩下的数据包。如果步骤5中描述的OUT事务没有成功完成，程序将会返回步骤3来再次发送该包。
7. 在所有的传输中的所有事务都被成功送达后，USBFS生成TF标志来指示传输成功结束。
8. 除能通道，当通道处于空闲状态，即可为其他传输做准备。

**注意：**开始写发送FIFO之前，需要保证发送FIFO以及请求队列存在剩余空间。

## 设备模式

**全局寄存器初始化顺序：**

1. 根据应用的需求，如Tx FIFO的空阈值等，设置寄存器USBFS_GAHBCS，此时，GINTEN位需要保持清零状态。
2. 根据应用的需求，如操作模式(主机、设备或OTG)、某些OTG参数、USB协议，设置寄存器USBFS_GUSBCS。
3. 根据应用的需求，设置寄存器USBFS_GCCFG。
4. 根据应用的需求，设置寄存器USBFS_GRFLEN、USBFS_HNPTFLEN_DIEP0TFLEN、USBFS_HPTFLEN，配置数据FIFO。
5. 通过设置寄存器USBFS_GINTEN使能模式错误、挂起、SOF、枚举完成和USB复位中断，置位USBFS_GAHBCS寄存器的GINTEN位使能全局中断。
6. 根据应用的需求，如设备的地址和设备的速度等，设置寄存器USBFS_DCFG。
7. 在设备连接上主机上后，主机在USB总线上执行端口复位，触发寄存器USBFS_GINTF的RST中断。
8. 等待寄存器USBFS_GINTF的ENUMF中断。

**端点初始化和使能顺序：**

1. 根据预期的传输类型、包大小等信息，设置寄存器 USBFS_DIEPnCTL 或 USBFS_DOEPxCTL。
2. 设定寄存器 USBFS_DIEPINTEN 或 USBFS_DOEPINTEN，置位相应中断使能位。
3. 设定寄存器 USBFS_DIEPxLEN 或 USBFS_DOEPxLEN，PCNT 表示一次传输中的包数，TLEN 表示一次传输中发送或接收的数据包的总字节数。
4. 对于 IN 端点，如果 PCNT 等于 1，单数据包的大小等于 TLEN。如果 PCNT 大于 1，前 PCNT-1 个包被认定为最大包长度的包，其大小是由寄存器 USBFS_DIEPxCTL 的位 MPL 所定义。最后一包的大小可通过 PCNT、TLEN 和 MPL 计算得到。如果程序想要发出一个零长度的包，应该设定 TLEN 为 0，PCNT 位 1。
5. 对于 OUT 端点，因为在 IN 事务结束之前，程序不知道实际接收的数据大小，程序可将 TLEN 设定为 Rx FIFO 所支持的最大值。
6. 置位 USBFS_DIEPxCTL 或 USBFS_DOEPxCTL 寄存器 EPEN 位使能端点。

**端点除能顺序：**

当USBFS_DIEPnCTL或USBFS_DOEPnCTL寄存器的EPEN位被清除时，程序可以在任何时候除能端点。

**IN传输操作顺序：**

1. 初始化USBFS全局寄存器。
2. 初始化和使能IN端点。
3. 将数据包写入端点的Tx FIFO，每当包数据写入FIFO，USBFS减少USBFS_DIEPxLEN寄存器的TLEN域的数值，其减少的数值等于已写的包数据大小。
4. 当IN令牌接收后，USBFS发送数据包，在USB总线上的事务完成后，USBFS_DIEPxLEN寄存器的PCNT值减1。如果事务成功完成(接收到ACK握手包)，ACK标志被触发，或者，其他状态标志表示事务的结果。
5. 在一次传输的所有数据包都被成功发送，USBFS生成一个TF标志位表明传输成功结束，除能相应IN端点。

**OUT传输操作顺序：**

1. 初始化USBFS全局寄存器。
2. 初始化和使能端点。
3. 当OUT令牌接收后，USBFS接收包数据或基于Rx FIFO状态和寄存器配置回复NAK握手包。如果事务成功完成(USBFS接收并保存数据到Rx FIFO，发送ACK握手包)，USBFS_DOEPxLEN寄存器的PCNT值减1。如果事务成功完成(接收到ACK握手包)，ACK标志被触发，或者，其他状态标志表示事务的结果。
4. 在一次传输的所有数据包都被成功接收，USBFS将TF状态条目压入Rx FIFO的最后的数据包的顶部，这样，软件在读取所有接收的包数据后，再读取TF状态条目。USBFS生成TF标志来指示传输成功结束。USBFS生成一个TF标志位表明传输成功结束，除能相应OUT端点。

### 28.5.7. 中断

USBFS 有两种中断：全局中断、唤醒中断。

全局中断是软件需要处理的主要中断，全局中断的标志位可在 USBFS_GINTF 寄存器读取，列举在表*表 28-2. USBFS 全局中断*。

**表 28-2. USBFS 全局中断**

| 中断标志 | 描述 | 运行模式 |
|---|---|---|
| SESIF | 会话中断 | 主机或设备模式 |
| DISCIF | 断开连接中断标志 | 主机模式 |
| IDPSC | ID 引脚状态变化 | 主机或设备模式 |
| PTXFEIF | 周期性 Tx FIFO 空中断标志 | 主机模式 |
| HCIF | 主机通道中断标志 | 主机模式 |
| HPIF | 主机端口中断中断 | 主机模式 |
| ISOONCIF/PXNCIF | 周期性传输未完成中断标志 / 同步OUT传输未完成中断标志 | 主机或设备模式 |
| ISOINCIF | 同步 IN 传输未完成中断标志 | 设备模式 |
| OEPIF | OUT 端点中断标志 | 设备模式 |
| IEPIF | IN 端点中断标志 | 设备模式 |
| EOPFIF | 周期性帧尾中断标志 | 设备模式 |
| ISOOPDIF | 同步 OUT 丢包中断标志 | 设备模式 |
| ENUMF | 枚举完成 | 设备模式 |
| RST | USB 复位 | 设备模式 |
| SP | USB挂起 | 设备模式 |
| ESP | 早挂起 | 设备模式 |
| GONAK | 全局OUT NAK有效 | 设备模式 |
| GNPINAK | 全局非周期IN NAK有效 | 设备模式 |
| NPTXFEIF | 非周期Tx FIFO空中断标志 | 主机模式 |
| RXFNEIF | Rx FIFO非空中断标志 | 主机或设备模式 |
| SOF | 帧首 | 主机或设备模式 |
| OTGIF | OTG 中断标志 | 主机或设备模式 |
| MFIF | 模式错误中断标志 | 主机或设备模式 |

唤醒中断可以在 USBFS 处于挂起状态时触发，即使 USBFS 的时钟停止。寄存器 USBFS_GINTF 的位 WKUPIF 是唤醒源。

## 28.6. USBFS 寄存器

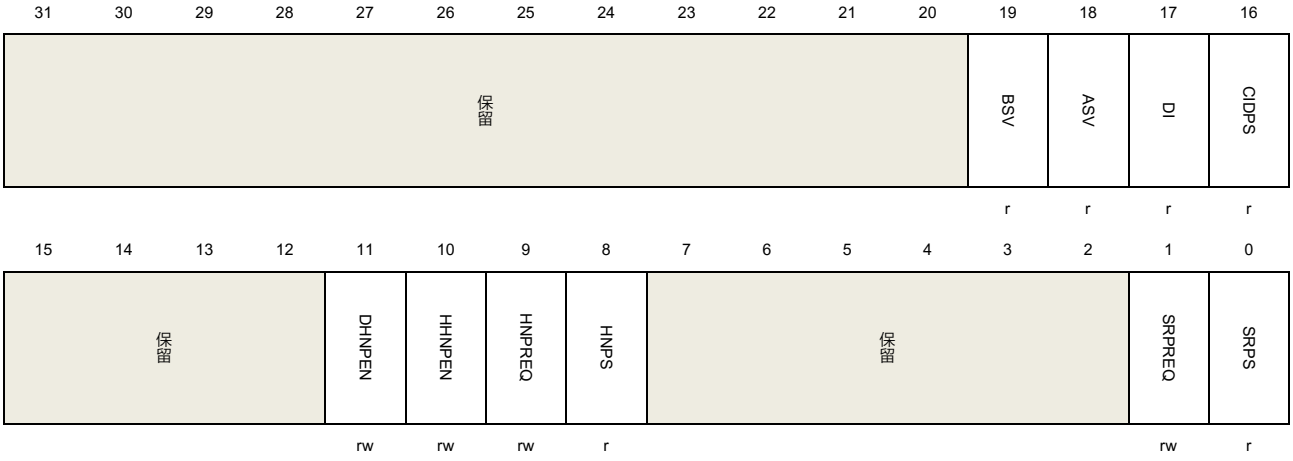
USBFS 基地址：0x5000 0000

### 28.6.1. USBFS 全局寄存器组

#### 全局 OTG 控制和状态寄存器(USBFS_GOTGCS)

地址偏移：0x0000
复位值：0x0000 0800

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:20 | 保留 | 必须保持复位值。 |
| 19 | BSV | B会话有效(在OTG协议中描述)<br>0：OTG B设备Vbus电压水平低于VBSESSVLD<br>1：OTG B设备Vbus电压水平不低于VBSESSVLD<br>**注意**：仅在OTB B设备模式下可访问 |
| 18 | ASV | A会话有效<br>A主机模式收发器状态<br>0：OTG A设备Vbus电压水平低于VASESSVLD<br>1：OTG A设备Vbus电压水平不低于VASESSVLD<br>在会话的开始，A设备默认是主机。<br>**注意**：仅在OTG A设备模式下可访问 |
| 17 | DI | 去抖动间隔<br>检测到连接的去抖动间隔。<br>0：当USB总线上发生插入和连接时，指示长去抖动间隔<br>1：当HNP协议中使用一个软连接时，指示短去抖动间隔 |

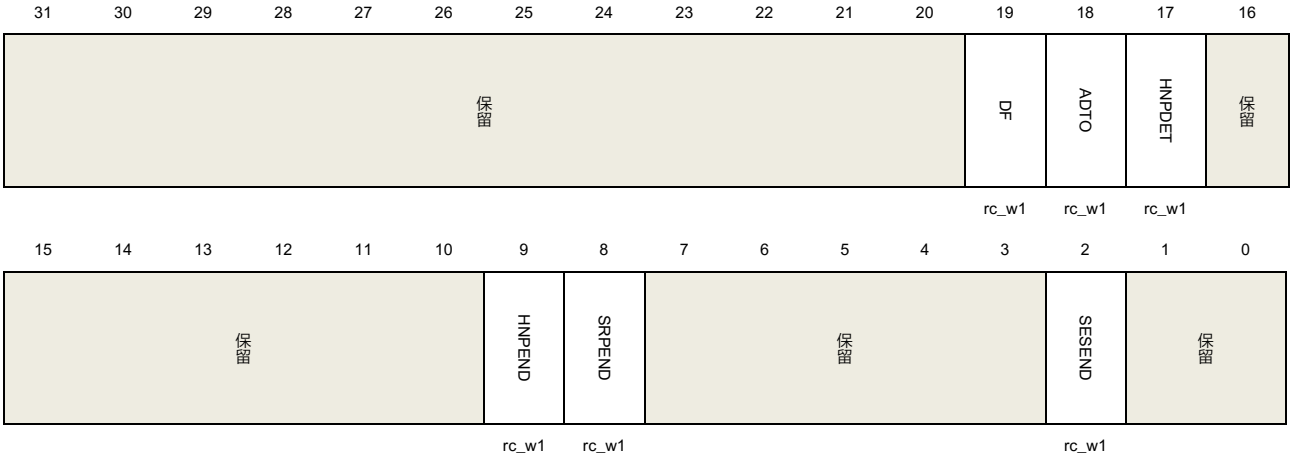
| | | |
|---|---|---|
| | | **注意**：仅在主机模式下可访问 |
| 16 | CIDPS | ID引脚状态<br>连接器ID引脚的电压水平<br>0：USBFS工作在A设备模式<br>1：USBFS工作在B设备模式<br>**注意**：在设备和主机模式下均可访问 |
| 15:12 | 保留 | 必须保持复位值。 |
| 11 | DHNPEN | 设备HNP使能<br>使能B设备HNP功能。如果该控制位清除，当应用置位USBFS_GOTGCS寄存器中的HNPREQ控制位c时，USBFS并不启动HNP协议。<br>0：HNP功能不使能<br>1：HNP功能使能<br>**注意**：仅在设备模式下访问 |
| 10 | HHNPEN | 主机HNP使能<br>使能A设备HNP功能。如果该控制位清除，USBFS不能够响应B设备的HNP请求。<br>0：HNP功能不使能<br>1：HNP功能使能<br>**注意**：仅在主机模式下访问 |
| 9 | HNPREQ | HNP请求<br>软件通过置位该控制位在USB总线上启动一个HNP。当USBFS_GOTGINTF寄存器中HNPEND控制位置位时，软件可以通过向该控制位写0或者清除USBFS_GOTGINTF寄存器中的HNPEND控制位来清除该控制位。<br>0：不发送HNP请求<br>1：发送HNP请求<br>**注意**：仅在设备模式下访问 |
| 8 | HNPS | HNP成功标志位<br>当HNP成功时，该标志位由内核置位。当HNPREQ置位时，该控制位被清除。<br>0：HNP失败<br>1：HNP成功<br>**注意**：仅在设备模式下访问 |
| 7:2 | 保留 | 必须保持复位值。 |
| 1 | SRPREQ | SRP请求<br>软件通过置位该控制位在USB总线上启动一个SRP会话请求。当USBFS_GOTGINTF寄存器中的SRPEND控制位置位时，软件可以通过向该控制位写0或者清除USBFS_GOTGINTF寄存器中的SRPEND控制位来清除该控制位。<br>0：没有会话请求<br>1：会话请求<br>**注意**：仅在设备模式下访问 |
| 0 | SRPS | SRP会话请求成功<br>当SRP会话请求成功时，该标志位由内核置位。当SRPREQ控制位被置位时，该标志 |

位被清除。
0：SRP会话请求失败
1：SRP会话请求成功
**注意**：仅在设备模式下访问

### 全局 OTG 中断状态寄存器(USBFS_GOTGINTF)

地址偏移：0x0004
复位值：0x0000 0000

该寄存器只能按字(32位)访问



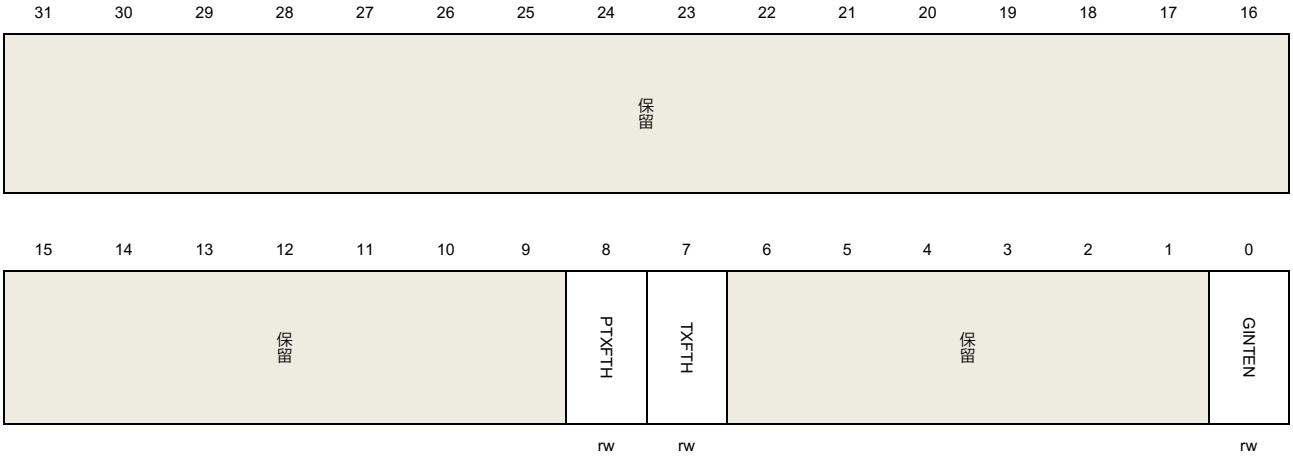

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:20 | 保留 | 必须保持复位值。 |
| 19 | DF | 去抖动完成<br>当设备连接去抖动完成时，USBFS置位该控制位<br>**注意**：仅在主机模式下可访问 |
| 18 | ADTO | A设备超时<br>当A设备等待B设备连接发生超时时，USBFS置位该控制位<br>**注意**：在设备和主机模式下，均可访问 |
| 17 | HNPDET | 检测到主机协商请求<br>当A设备检测到一个HNP请求时，USBFS置位该标志位<br>**注意**：在设备和主机模式下，均可访问 |
| 16:10 | 保留 | 必须保持复位值。 |
| 9 | HNPEND | HNP结束<br>当一个HNP结束时，内核置位该标志位。软件应该读取USBFS_GOTGCS寄存器中HNPS标志位，以获取HNP结果。<br>**注意**：在设备和主机模式下，均可访问。 |
| 8 | SRPEND | SRPEND<br>当一个SRP结束时，内核置位该标志位。软件应该读取USBFS_GOTGCS寄存器中 |

| | | |
|---|---|---|
| | | SRPS标志位，以获取SRP结果。<br>**注意**：在设备和主机模式下，均可访问。 |
| 7:3 | 保留 | 必须保持复位值。 |
| 2 | SESEND | 会话结束<br>当VBUS电压低于Vb_ses_vld时，内核置位该标志位。 |
| 1:0 | 保留 | 必须保持复位值。 |

### 全局 AHB 控制和状态寄存器(USBFS_GAHBCS)

地址偏移：0x0008
复位值：0x0000 0000

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:9 | 保留 | 必须保持复位值。 |
| 8 | PTXFTH | 周期性Tx FIFO阈值<br>0：当周期性发送FIFO半空时，将触发PTXFEIF标志位<br>1：当周期性发送FIFO全空时，将触发PTXFEIF标志位<br>**注意**：只在主机模式下访问 |
| 7 | TXFTH | Tx FIFO阈值<br>设备模式：<br>0：当IN端点发送FIFO半空时，将触发TXFEIF标志位<br>1：当IN端点发送FIFO全空时，将触发TXFEIF标志位<br>主机模式：<br>0：当非周期性发送FIFO半空时，将触发NPTXFEIF标志位<br>1：当非周期性发送FIFO全空时，将触发NPTXFEIF标志位 |
| 6:1 | 保留 | 必须保持复位值。 |
| 0 | GINTEN | 全局中断使能<br>0：全局中断不使能 |

1：全局中断使能

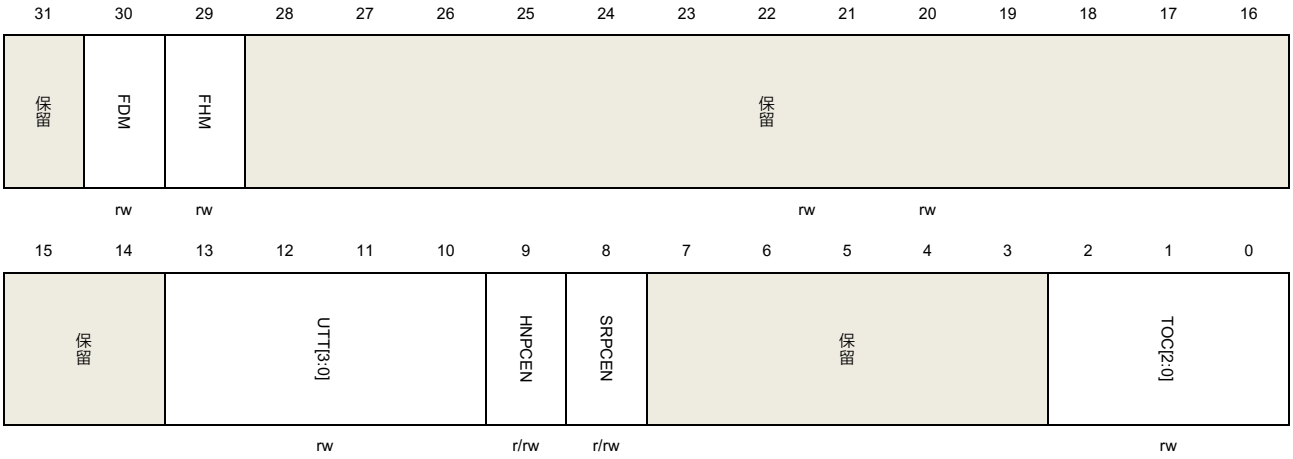
**注意**：在主机和设备模式下，均可访问

### 全局 USB 控制和状态寄存器(USBFS_GUSBCS)

地址偏移：0x000C

复位值：0x0000 0A80

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30 | FDM | 强制设备模式<br>通过置位该控制位，可强制USB内核为设备模式，并且忽略USBFS ID引脚的输入状态<br>0：正常模式<br>1：设备模式<br>设置该控制位后，应用必须等待至少25ms，让变化产生作用。<br>**注意**：在设备和主机模式下，均可访问。 |
| 29 | FHM | 强制主机模式<br>通过置位该控制位，可强制USB内核为主机模式，并且忽略USBFS ID引脚的输入状态<br>0：正常模式<br>1：主机模式<br>设置该控制位后，应用必须等待至少25ms，让变化产生作用。<br>**注意**：在设备和主机模式下，均可访问 |
| 28:14 | 保留 | 必须保持复位值。 |
| 13:10 | UTT[3:0] | USB运转时间<br>以物理时钟数来设定运转时间<br>**注意**：仅在设备模式下访问 |

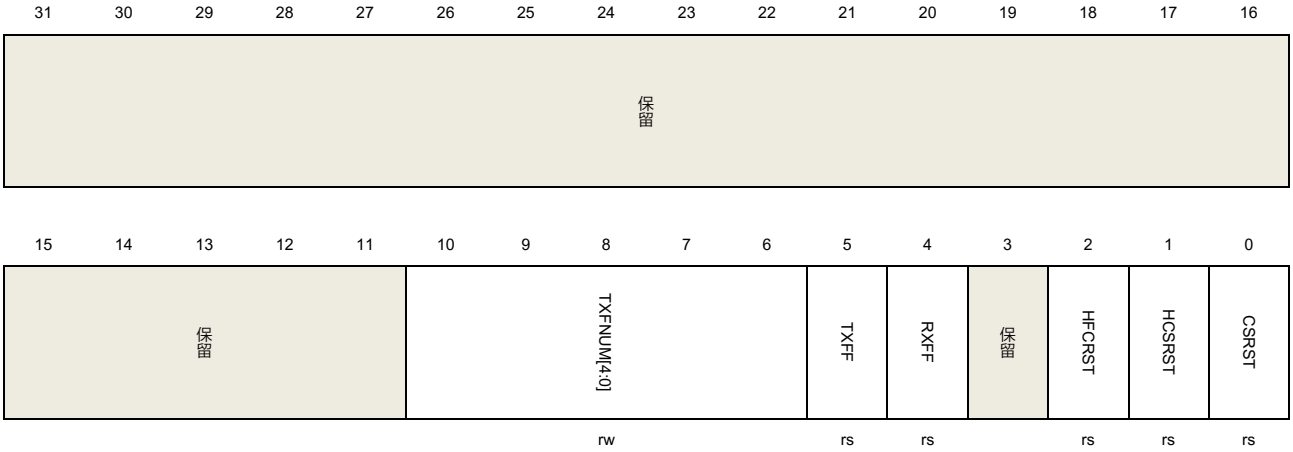
| | | |
|---|---|---|
| 9 | HNPCEN | HNP能力使能<br>控制HNP能力是否使能<br>0：HNP能力禁用<br>1：HNP能力使能<br>**注意**：在设备和主机模式下，均可访问 |
| 8 | SRPCEN | SRP能力使能<br>控制SRP能力是否使能<br>0：SRP能力禁用<br>1：SRP能力使能<br>**注意**：在设备和主机模式下，均可访问 |
| 7:3 | 保留 | 必须保持复位值。 |
| 2:0 | TOC[2:0] | 超时校准<br>当等待一个包时，USBFS需要使用USB2.0协议中需要的超时数值。应用可以使用TOC[2:0]增加该数值(以PHY时钟为单位)。PHY时钟频率为48MHz。 |

### 全局复位控制寄存器(USBFS_GRSTCTL)

地址偏移：0x0010
复位值：0x8000 0000

应用通过该寄存器来复位内核的不同硬件特性。

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:11 | 保留 | 必须保持复位值。 |
| 10:6 | TXFNUM[4:0] | Tx FIFO数目<br>当本寄存器中TXFF控制位置位时，该标志位决定那个Tx FIFO会被冲刷<br>主机模式：<br>00000：仅非周期性Tx FIFO被冲刷<br>00001：仅周期性Tx FIFO被冲刷 |

| | | |
|---|---|---|
| | | 1xxxx：周期性和非周期性Tx FIFO均被冲刷<br>其他：没有数据被冲刷<br>设备模式：<br>00000：仅Tx FIFO0被冲刷<br>00001：仅Tx FIFO1被冲刷<br>…<br>00011：仅Tx FIFO3被冲刷<br>1XXXX：所有的Tx FIFO均被冲刷<br>其他：没有数据被冲刷 |
| 5 | TXFF | Tx FIFO冲刷控制位<br>应用通过置位该控制位来冲刷Tx FIFO数据，并且TXFNUM[4:0]决定冲刷的FIFO数目。当冲刷完成后，硬件自动清除该控制位。置位该控制位后，应用应该等待该控制位清除，并且，在此之前USBFS不应有其他操作。<br>**注意**：在设备和主机模式下，均可访问 |
| 4 | RXFF | Rx FIFO冲刷控制位<br>应用通过置位该控制位来冲刷Rx FIFO数据。当冲刷完成后，硬件自动清除该控制位。置位该控制位后，应用应该等待该控制位清除，并且，在此之前USBFS不应有其他操作。<br>**注意**：在设备和主机模式下，均可访问 |
| 3 | 保留 | 必须保持复位值。 |
| 2 | HFCRST | 主机帧计数器复位<br>应用通过置位该控制位来复位USBFS内的帧计数器。该控制位置位后，接下来SOF的帧计数器将变为0。当复位操作完成后，硬件自动清除该控制位。置位该控制位后，应用应该等待该控制位清除，并且，在此之前USBFS不应有其他操作。<br>**注意**：仅在主机模式下访问 |
| 1 | HCSRST | HCLK软件复位<br>应用通过置位该控制位来复位ABH时钟域电路<br>在复位操作完成后，硬件自动清除该控制位。置位该控制位后，应用应该等待该控制位清除，并且，在此之前USBFS不应有其他操作。<br>**注意**：在设备和主机模式下，均可访问 |
| 0 | CSRST | USB内核软件复位<br>复位AHB和USB时钟域电路，以及大多数的寄存器。 |

### 全局中断标志寄存器(USBFS_GINTF)

地址偏移：0x0014
复位值：0x0400 0021

该寄存器只能按字(32位)访问

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

| WKUPIF | SESIF | DISCIF | IDPSC | 保留 | PTXFEIF | HCIF | HPIF | 保留 | PXNCIF/ISOONCIF | ISOINCIF | OEPIF | IEPIF | 保留 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| rc_w1 | rc_w1 | rc_w1 | rc_w1 | | r | r | r | | rc_w1 | rc_w1 | r | r | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

| EOPFIF | ISOOPDIF | ENUMF | RST | SP | ESP | 保留 | GONAK | GNPINAK | NPTXFEIF | RXFNEIF | SOF | OTGIF | MFIF | COPM |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | | r | r | r | r | rc_w1 | r | rc_w1 | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | WKUPIF | 唤醒中断标志位<br>当在USB总线上检测到一个恢复信号(在设备模式下)或者一个远程唤醒信号(在主机模式下)，硬件将置位该中断标志位。<br>**注意**：在设备和主机模式下，均可访问 |
| 30 | SESIF | 会话中断标志位<br>当在A设备模式下检测到一个SRP会话请求或在B设备模式下B设备的Vbus变为可用时，硬件将置位该中断标志位<br>**注意**：在设备和主机模式下，均可访问 |
| 29 | DISCIF | 断开中断标志位<br>当设备断开后，将触发该标志位。<br>**注意**：仅在主机模式下访问 |
| 28 | IDPSC | ID引脚状态改变中断标志位<br>当ID引脚状态改变时，内核将置位该标志位<br>**注意**：在设备和主机模式下，均可访问 |
| 27 | 保留 | 必须保持复位值。 |
| 26 | PTXFEIF | 周期性Tx FIFO空中断标志位<br>当周期性发送FIFO半空或全空时，将触发该标志位。空阈值由USBFS_GAHBCS寄存器中周期性Tx FIFO空等级控制位(PTXFTH)决定。<br>**注意**：仅在主机模式下访问 |
| 25 | HCIF | 主机通道中断标志位<br>当在主机模式下其中一个通道挂起一个中断时，USBFS将置位该标志位。软件应该首先读取USBFS_HACHINT寄存器以获取通道号，然后读取相应的USBFS_HCHxINTF寄存器以获取产生中断的通道标志位。当产生通道中断的独立通道标志位被清除后，该中断标志位将自动清除。<br>**注意**：仅在主机模式下访问 |
| 24 | HPIF | 主机端口中断标志位<br>当USBFS在主机模式下检测到端口状态改变时，USB内核将置位该标志位。软件应该读取USBFS_HPCS寄存器以获取该中断源。当产生端口中断的标志被清除后，该 |

| | | |
|---|---|---|
| | | 中断标志位将自动清除。<br>**注意**：仅在主机模式下访问 |
| 23:22 | 保留 | 必须保持复位值。 |
| 21 | PXNCIF | 周期性传输未完成中断标志位<br>在当前帧内，当帧结束时，周期性传输未完成，USBFS将置位该标志位(主机模式)。同步OUT传输未完成中断标志位 |
| | ISOONCIF | 在周期性帧结束时(由USBFS_DCFG寄存器的EOPFT控制位定义)，如果仍有同步OUT端点未完成传输，USBFS将置位该标志位(设备模式)。 |
| 20 | ISOINCIF | 同步IN传输未完成中断标志位<br>在周期性帧结束时(由USBFS_DCFG寄存器的EOPFT控制位定义)，如果仍有同步IN端点未完成传输，USBFS将置位该标志位(设备模式)。<br>**注意**：仅在设备模式下访问 |
| 19 | OEPIF | OUT端点中断标志位<br>当在设备模式下，其中一个OUT端点挂起一个中断时，USBFS将置位该中断标志位。软件应该首先读取USBFS_DAEPINT寄存器以获取设备号，然后读取相应的USBFS_ DOEPxINTF寄存器以获取产生中断的端点标志位。当产生中断的相应端点标志位被清除后，该中断标志位被自动清除。<br>**注意**：仅在设备模式下访问 |
| 18 | IEPIF | IN端点中断标志位<br>当在设备模式下，其中一个IN端点挂起一个中断时，USBFS将置位该标志位。软件应该首先读取USBFS_DAEPINT寄存器以获取设备号，然后读取相应的USBFS_DIEPxINTF寄存器以获取产生中断的端点标志位。当相应产生中断的端点标志位被清除后，该中断标志位被自动清除。 |
| 17:16 | 保留 | 必须保持复位值。 |
| 15 | EOPFIF | 周期性帧结束中断标志位<br>当一帧内USB总线时间已经达到USBFS_DCFG寄存器中EOPFT控制位所定义的数值时，USBFS将置位该中断标志位。<br>**注意**：仅在设备模式下访问 |
| 14 | ISOOPDIF | 同步OUT包丢失中断标志位<br>如果USBFS接收到一个同步OUT包，但是Rx FIFO没有足够的空间来接收该OUT包，USBFS将置位该标志位。<br>**注意**：仅在设备模式下访问 |
| 13 | ENUMF | 枚举完成中断标志位<br>在速度枚举完成后，USBFS将置位该中断标志位。软件能够读取USBFS_DSTAT寄存器以获取当前设备速度。<br>**注意**：仅在设备模式下访问 |
| 12 | RST | USB复位中断标志位<br>当USBFS在USB总线上检测到一个USB复位信号后，USBFS将置位该中断标志位。 |

| | | |
|---|---|---|
| | | **注意**：仅在设备模式下访问 |
| 11 | SP | USB挂起中断标志位<br>当USBFS检测到USB总线空闲3ms并且进入挂起状态，USBFS将置位该中断标志位。<br>**注意**：仅在设备模式下访问 |
| 10 | ESP | 早期挂起中断标志位<br>当USBFS检测到USB总线空闲3ms时，USBFS将置位该中断标志位。 |
| 9:8 | 保留 | 必须保持复位值。 |
| 7 | GONAK | 全局OUT NAK有效标志位<br>软件能够向USBFS_DCTL寄存器的SGONAK控制位写1，并且USBFS将会在SGONAK写入有效后，置位GONAK标志位。<br>**注意**：仅在设备模式下可访问 |
| 6 | GNPINAK | 全局非周期性IN NAK有效标志位<br>软件能够向USBFS_DCTL寄存器中的SGINAK控制位写1，并且USBFS将会在SGINAK写入有效后，置位GNPINAK标志位<br>**注意**：仅在设备模式下可访问 |
| 5 | NPTXFEIF | 非周期性Tx FIFO空中断标志位<br>当非周期性Tx FIFO为半空或全空时，将置位该中断标志位。该阈值由USBFS_GAHBCS寄存器中的非周期Tx FIFO空等级控制位(TXFTH)决定。<br>**注意**：仅在主机模式下访问 |
| 4 | RXFNEIF | Rx FIFO非空中断标志位<br>当至少有一个包或状态条目在Rx FIFO中时，USBFS将置位该标志位。<br>**注意**：在主机和设备模式下，均可访问 |
| 3 | SOF | 帧起始中断标志位<br>主机模式：<br>当准备在USB总线上发送一个SOF或保持有效信号，USBFS将置位该中断标志位。软件可以通过写1清除该中断标志位。<br>设备模式：<br>当USBFS接收到一个SOF令牌包后，USBFS置位该标志位。应用可以读取设备状态寄存器以获取当前帧号。软件可以通过写1清除该中断标志位。<br>**注意**：在设备和主机模式下，均可访问 |
| 2 | OTGIF | OTG中断标志位<br>当USBFS_GOTGINTF寄存器中标志位产生一个中断时，USBFS置位该中断标志位。软件应该读取USBFS_GOTGINTF寄存器以获取产生该中断的信号源，当USBFS_GOTGINTF寄存器中产生该中断的标志位被清除后，该中断标志位也被自动清除。<br>**注意**：在设备和主机模式下，均可访问 |
| 1 | MFIF | 模式错误中断标志位<br>如果软件在设备模式下操作仅主机可访问的寄存器或者在主机模式下操作仅设备可 |

访问的寄存器，USBFS将置位该中断标志位。这些错误操作不会产生作用。

**注意**：在主机和设备模式下，均可访问

| | | |
|---|---|---|
| 0 | COPM | 当前操作模式<br>0：设备模式<br>1：主机模式<br>**注意**：在主机和设备模式下，均可访问 |

### 全局中断使能寄存器(USBFS_GINTEN)

地址偏移：0x0018
复位值：0x0000 0000

这个寄存器同全局中断标志寄存器(USBFS_GINTF)一起工作来中断应用程序。当中断使能位被禁止后，相应的中断就不会产生。然而，相应的全局中断标志位依然会被置位。

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WKUPIE | SESIE | DISCIE | IDPSCIE | 保留 | PTXFEIE | HCIE | HPIE | 保留 | | PXNCIE/ISOONCIE | ISOINCIE | OEPIE | IEPIE | 保留 | |
| rw | rw | rw | rw | | rw | rw | r | | | rw | rw | rw | rw | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EOPFIE | ISOOPDIE | ENUMFIE | RSTIE | SPIE | ESPIE | 保留 | | GONAKIE | GNPINAKIE | NPTXFEIE | RXFNEIE | SOFIE | OTGIE | MFIE | 保留 |
| rw | rw | rw | rw | rw | rw | | | rw | rw | rw | rw | rw | rw | rw | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | WKUPIE | 唤醒中断使能<br>0：禁用唤醒中断<br>1：使能唤醒中断<br>**注意**：在主机和设备模式下，均可访问 |
| 30 | SESIE | 会话中断使能<br>0：禁用会话中断<br>1：使能会话中断<br>**注意**：在主机和设备模式下，均可访问 |
| 29 | DISCIE | 断开中断使能<br>0：禁用断开中断<br>1：使能断开中断<br>**注意**：仅在设备模式下使用 |
| 28 | IDPSCIE | ID引脚状态改变中断使能 |

| | | |
|---|---|---|
| | | 0：禁用连接器ID引脚状态中断<br>1：使能连接器ID引脚状态中断<br>**注意**：在主机和设备模式下，均可访问 |
| 27 | 保留 | 必须保持复位值。 |
| 26 | PTXFEIE | 周期性Tx FIFO空中断使能<br>0：禁用周期性Tx FIFO空中断<br>1：使能周期性Tx FIFO空中断<br>**注意**：仅在主机模式下访问 |
| 25 | HCIE | 主机通道中断使能<br>0：禁用主机通道中断<br>1：使能主机通道中断<br>**注意**：仅在主机模式下访问 |
| 24 | HPIE | 主机端口中断使能<br>0：禁止主机端口中断<br>1：使能主机端口中断<br>**注意**：仅在主机模式下访问 |
| 23:22 | 保留 | 必须保持复位值。 |
| 21 | PXNCIE | 周期性传输未完成中断使能<br>0：禁止周期性未完成传输中断<br>1：使能周期性未完成传输中断<br>**注意**：仅在主机模式下访问 |
| | ISOONCIE | 同步OUT传输未完成中断使能<br>0：禁止同步OUT传输未完成中断<br>1：使能同步OUT传输未完成中断<br>**注意**：仅在设备模式下访问 |
| 20 | ISOINCIE | 同步IN传输未完成中断使能<br>0：禁止同步IN传输未完成中断<br>1：使能同步IN传输未完成中断<br>**注意**：仅在设备模式下访问 |
| 19 | OEPIE | OUT端点中断使能<br>0：禁止OUT端点中断<br>1：使能OUT端点中断<br>**注意**：仅在设备模式下访问 |
| 18 | IEPIE | IN端点中断使能<br>0：禁止IN端点中断<br>1：使能IN端点中断<br>**注意**：仅在设备模式下访问 |
| 17:16 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 15 | EOPFIE | 周期性帧结束中断使能<br>0：禁止周期性帧结束中断<br>1：使能周期性帧结束中断<br>**注意**：仅在设备模式下访问 |
| 14 | ISOOPDIE | 同步OUT包丢失中断使能<br>0：禁止同步OUT包丢失中断<br>1：使能同步OUT包丢失中断<br>**注意**：仅在设备模式下访问 |
| 13 | ENUMFIE | 枚举完成中断使能<br>0：禁止枚举完成中断<br>1：使能枚举完成中断<br>**注意**：仅在设备模式下访问 |
| 12 | RSTIE | USB复位中断使能<br>0：禁止USB复位中断<br>1：使能USB复位中断<br>**注意**：仅在设备模式下访问 |
| 11 | SPIE | USB挂起中断使能<br>0：禁止USB挂起中断<br>1：使能USB挂起中断<br>**注意**：仅在设备模式下访问 |
| 10 | ESPIE | 早期挂起中断使能<br>0：禁止早期挂起中断<br>1：使能早期挂起中断<br>**注意**：仅在设备模式下访问 |
| 9:8 | 保留 | 必须保持复位值。 |
| 7 | GONAKIE | 全局OUT NAK有效中断使能<br>0：禁止全局OUT NAK有效中断<br>1：使能全局OUT NAK有效中断<br>**注意**：仅在设备模式下访问 |
| 6 | GNPINAKIE | 全局非周期性IN NAK有效中断使能<br>0：禁止全局非周期性IN NAK有效中断<br>1：使能全局非周期性IN NAK有效中断<br>**注意**：仅在设备模式下访问 |
| 5 | NPTXFEIE | 非周期性发送FIFO空中断使能<br>0：禁止非周期性发送FIFO空中断<br>1：使能非周期性发送FIFO空中断<br>**注意**：仅在主机模式下访问 |
| 4 | RXFNEIE | 接收FIFO非空中断使能<br>0：禁止接收FIFO非空中断 |

| | | |
|---|---|---|
| | | 1：使能接收FIFO非空中断<br>**注意**：在设备模式与主机模式下，均可访问 |
| 3 | SOFIE | 帧首中断使能<br>0：禁止帧首中断<br>1：使能帧首中断<br>**注意**：在设备模式下与主机模式下，均可访问 |
| 2 | OTGIE | OTG中断使能<br>0：禁止OTG中断<br>1：使能OTG中断<br>**注意**：在设备模式下与主机模式下，均可访问 |
| 1 | MFIE | 模式错误中断使能<br>0：禁止模式错误中断<br>1：使能模式错误中断<br>**注意**：在设备模式下与主机模式下，均可访问 |
| 0 | 保留 | 必须保持复位值。 |

### 全局接收状态读取/接收状态读取和弹出寄存器 (USBFS_GRSTATR/USBFS_GRSTATP)

读地址偏移：0x001C
弹出地址偏移：0x0020

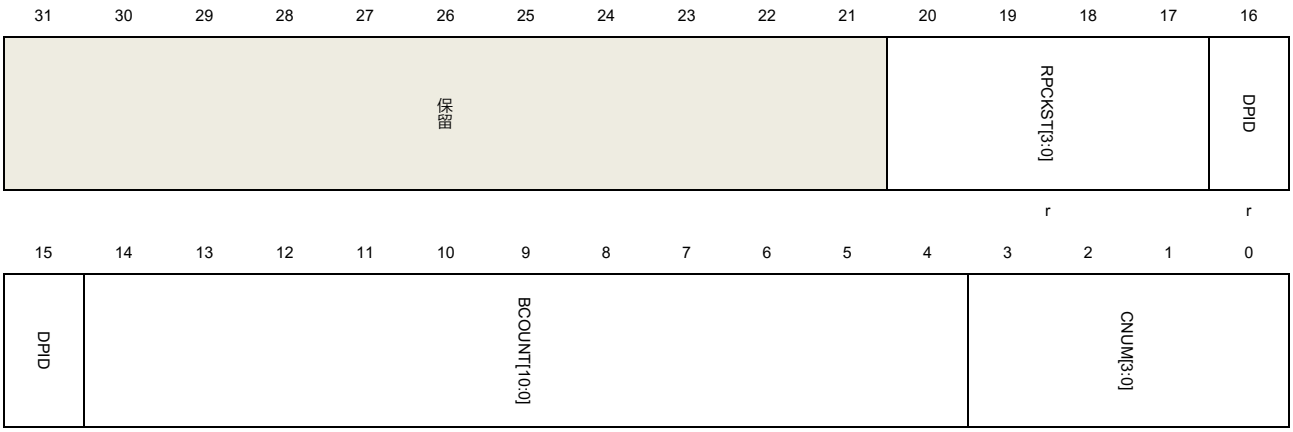
复位值：0x0000 0000

对接收状态读寄存器的读操作，将返回接收FIFO中顶部的条目。对接收状态读取和弹出寄存器的读操作，将额外的弹出Rx FIFO的顶部条目。

在主机模式和设备模式下，Rx FIFO中的条目具有不同的含义。当全局中断标志寄存器(USBFS_GINTF)中的接收FIFO非空中断标志位(RXFNEIF)置位后，软件应该读取该寄存器。

该寄存器只能按字(32位)访问

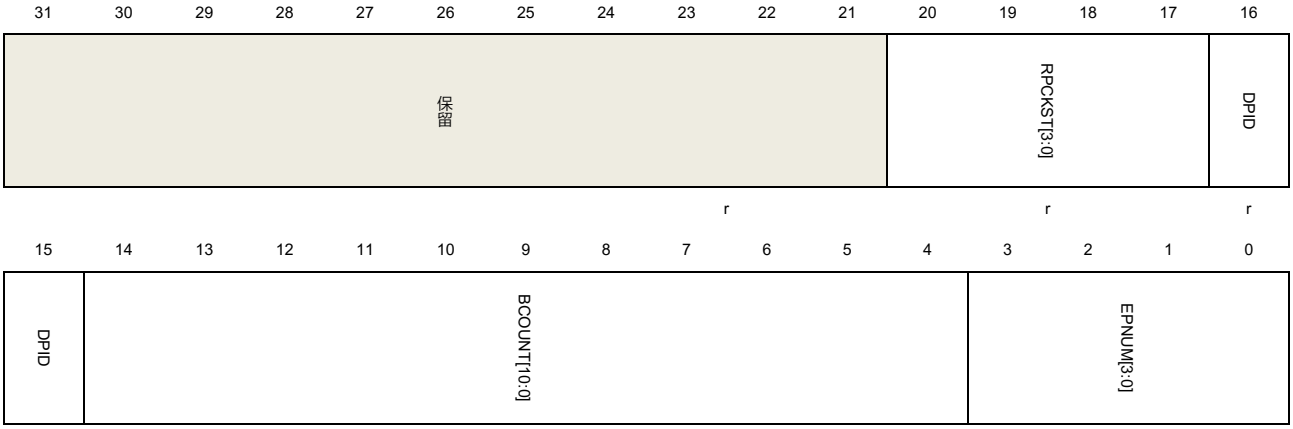
**主机模式：**



r r r

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:21 | 保留 | 必须保持复位值。 |
| 20:17 | RPCKST[3:0] | 接收包状态<br>0010：接收到IN数据包<br>0011：IN传输完成(如果取出，触发一个中断)<br>0101：数据翻转错误(如果取出，触发一个中断)<br>0111：通道中止(如果取出，触发一个中断)<br>其他：保留 |
| 16:15 | DPID[1:0] | 数据PID<br>接收包的数据PID<br>00：DATA0<br>10：DATA1<br>其他：保留 |
| 14:4 | BCOUNT[10:0] | 字节数<br>接收IN数据包字节数。 |
| 3:0 | CNUM[3:0] | 通道数<br>当前接收包所属通道编号。 |

**设备模式：**



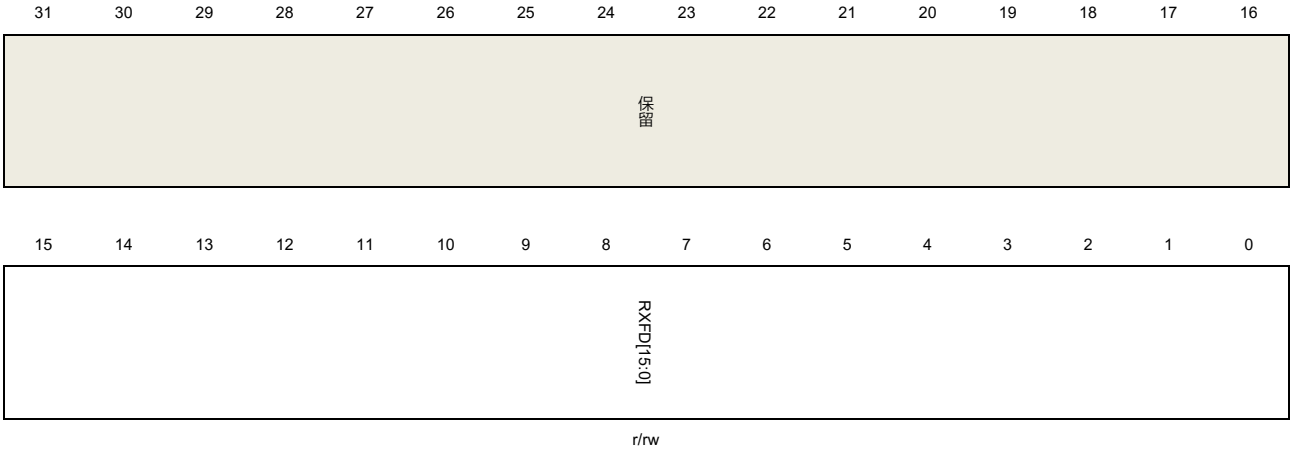

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:21 | 保留 | 必须保持复位值。 |
| 20:17 | RPCKST[3:0] | 接收包状态<br>0001：全局OUT NAK(产生一个中断)<br>0010：接收到OUT数据包<br>0011：OUT传输完成(产生一个中断)<br>0100：SETUP传输完成(产生一个中断)<br>0110：接收到SETUP数据包 |

| | | |
|---|---|---|
| | | 其他：保留 |
| 16:15 | DPID[1:0] | 数据PID<br>接收到OUT数据包的数据PID<br>00：DATA0<br>10：DATA1<br>其他：保留 |
| 14:4 | BCOUNT[10:0] | 字节数<br>接收数据包的字节数 |
| 3:0 | EPNUM[3:0] | 端点号<br>当前接收包所属端点编号 |

### 全局接收 FIFO 长度寄存器(USBFS_GRFLEN)

地址偏移：0x0024
复位值：0x0000 0200

该寄存器只能按字(32位)访问

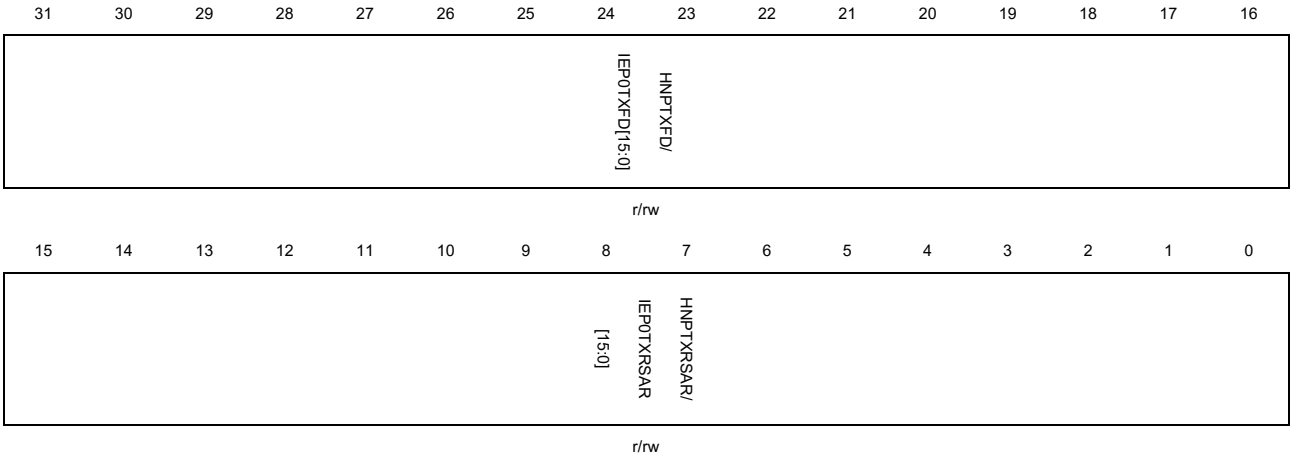



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | RXFD[15:0] | Rx FIFO 深度<br>以32位字计数<br>1<=RXFD<=1024 |

### 主机非周期性发送 FIFO 长度寄存器/设备 IN 端点 0 发送 FIFO 长度寄存器 (USBFS_HNPTFLEN_DIEP0TFLEN)

地址偏移：0x0028
复位值：0x0200 0200

该寄存器只能按字(32位)访问




**主机模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | HNPTXFD[15:0] | 主机非周期性Tx FIFO深度<br>以32位字计数<br>1<=HNPTXFD<=1024 |
| 15:0 | HNPTXRSAR[15:0] | 主机非周期性Tx RAM起始地址<br>非周期性发送FIFO RAM的起始地址 |

**设备模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | IEP0TXFD[15:0] | 输入端点0 Tx FIFO深度<br>以32位字计数<br>16<=IEP0TXFD<=140 |
| 15:0 | IEP0TXRSAR[15:0] | 输入端点0 TX RAM起始地址<br>端点0发送FIFO RAM的起始地址 |

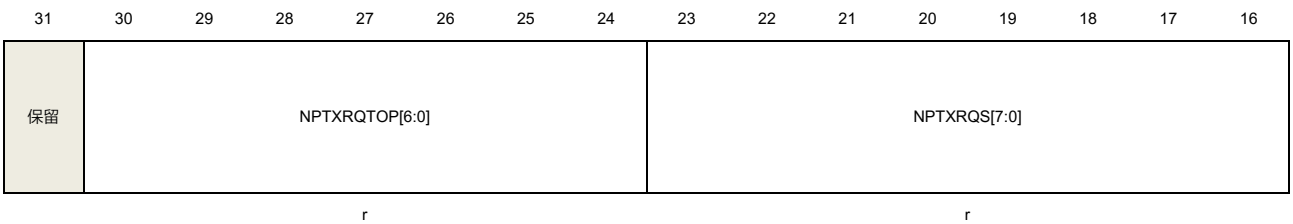
### 主机非周期性发送 FIFO/队列状态寄存器(USBFS_HNPTFQSTAT)

地址偏移：0x002C
复位值：0x0008 0200

该寄存器只能按字(32位)访问

该寄存器反映了非周期性Tx FIFO和请求队列的当前状态。请求队列包括在主机模式下的IN、OUT或其他请求条目。

**注意：**在设备模式下，该寄存器不可用。



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

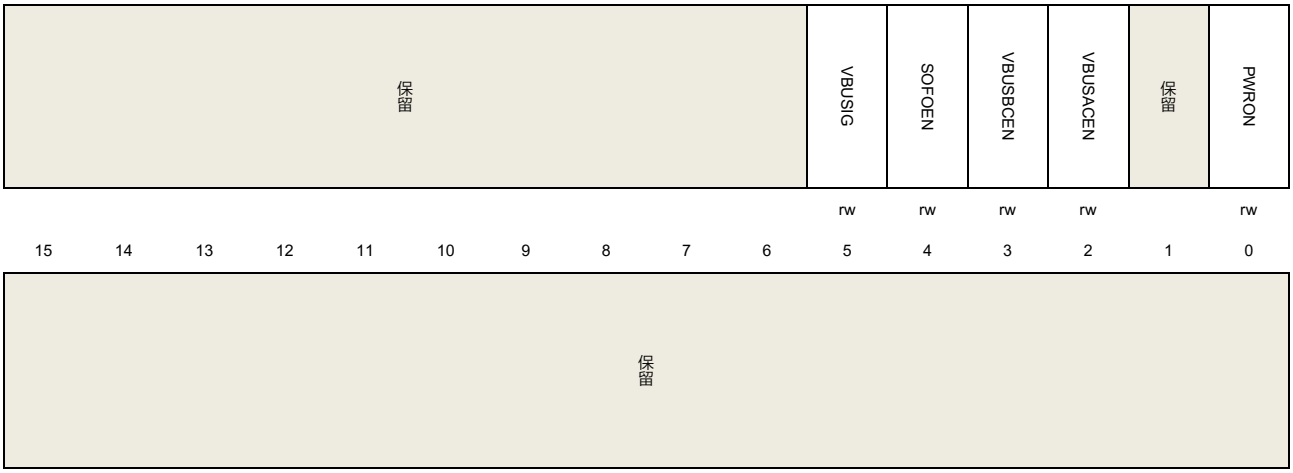
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30:24 | NPTXRQTOP[6:0] | 非周期性发送请求队列的顶部条目<br>在非周期性传输请求队列中的条目。<br>位30:27：通道号<br>位26:25：<br>- 00：IN/OUT令牌<br>- 01：0长度OUT包<br>- 11：通道中止请求<br>位24：结束标志位，表明所选通道的最后一个条目 |
| 23:16 | NPTXRQS[7:0] | 非周期性发送请求队列空间<br>非周期性请求队列的剩余空间<br>0：请求队列空<br>1：1个条目<br>2：2个条目<br>…<br>n：n个条目 (0<=n<=8)<br>其他：保留 |
| 15:0 | NPTXFS[15:0] | 非周期性Tx FIFO空间<br>非周期性发送FIFO剩余空间<br>以32位字计数<br>0：非周期性Tx FIFO为空<br>1：1个字<br>2：2个字<br>n：n个字(0<=n<=NPTXFD)<br>其他：保留 |

### 全局内核配置寄存器(USBFS_GCCFG)

地址偏移：0x0038
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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



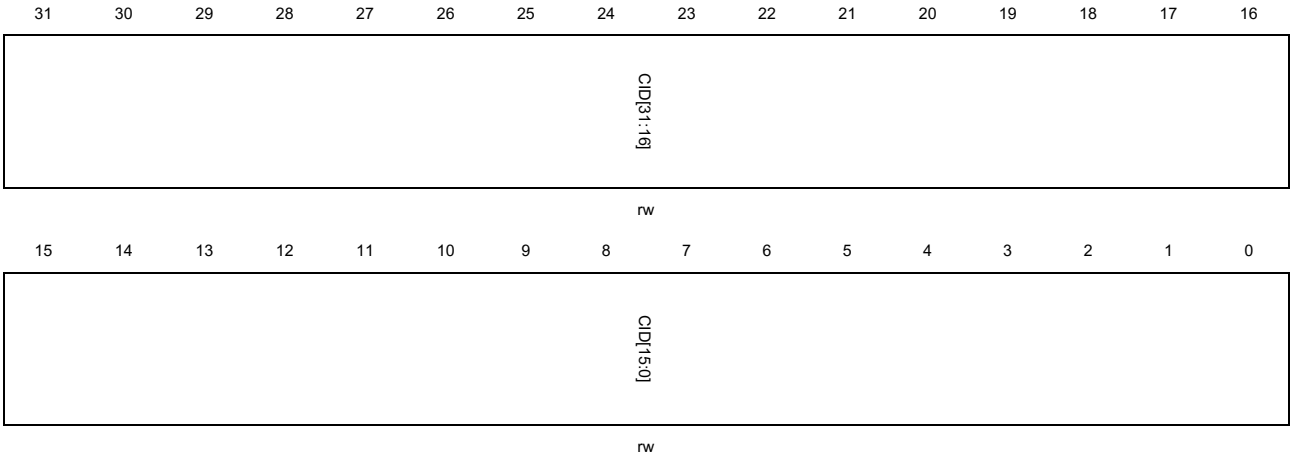

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:22 | 保留 | 必须保持复位值。 |
| 21 | VBUSIG | $V_{BUS}$忽略<br>当该控制位被置位，USBFS并不监测$V_{BUS}$引脚电压，并且认为在主机和设备模式下，$V_{BUS}$电压一直有效，然后可释放$V_{BUS}$引脚作为其他用途。<br>0：$V_{BUS}$不被忽略<br>1：$V_{BUS}$被忽略，并认为$V_{BUS}$电压一直有效 |
| 20 | SOFOEN | SOF输出使能<br>0：SOF脉冲输出禁止<br>1：SOF脉冲输出使能 |
| 19 | VBUSBCEN | $V_{BUS}$ B设备比较器使能<br>0：$V_{BUS}$ B设备比较器禁止<br>1：$V_{BUS}$ B设备比较器使能 |
| 18 | VBUSACEN | $V_{BUS}$ A设备比较器使能<br>0：$V_{BUS}$ A设备比较器禁止<br>1：$V_{BUS}$ A设备比较器使能 |
| 17 | 保留 | 必须保持复位值。 |
| 16 | PWRON | 上电<br>该控制位为内部嵌入式全速PHY的电源开关<br>0：嵌入式全速PHY掉电<br>1：嵌入式全速PHY上电 |
| 15:0 | 保留 | 必须保持复位值。 |

### 内核 ID 寄存器(USBFS_CID)

地址偏移：0x003C
复位值：0x0000 1000

该寄存器包含产品ID

该寄存器只能按字(32位)访问



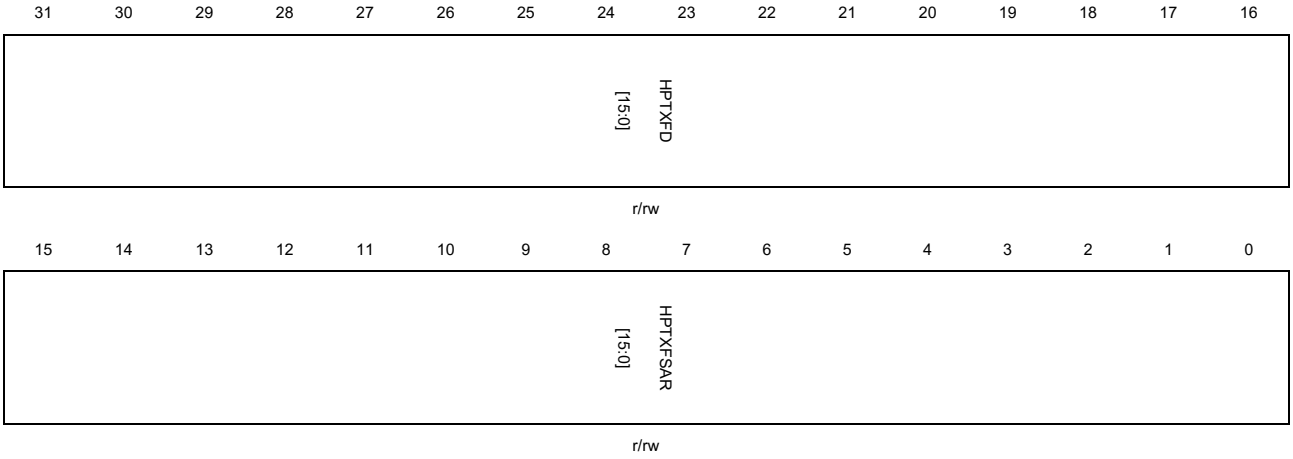

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | CID | 内核ID<br>软件能够写入或读取该域值，并利用该域值为应用产生一个唯一ID。 |

### 主机周期性发送 FIFO 长度寄存器(USBFS_HPTFLEN)

地址偏移：0x0100

复位值：0x0200 0600

该寄存器只能按字(32位)访问

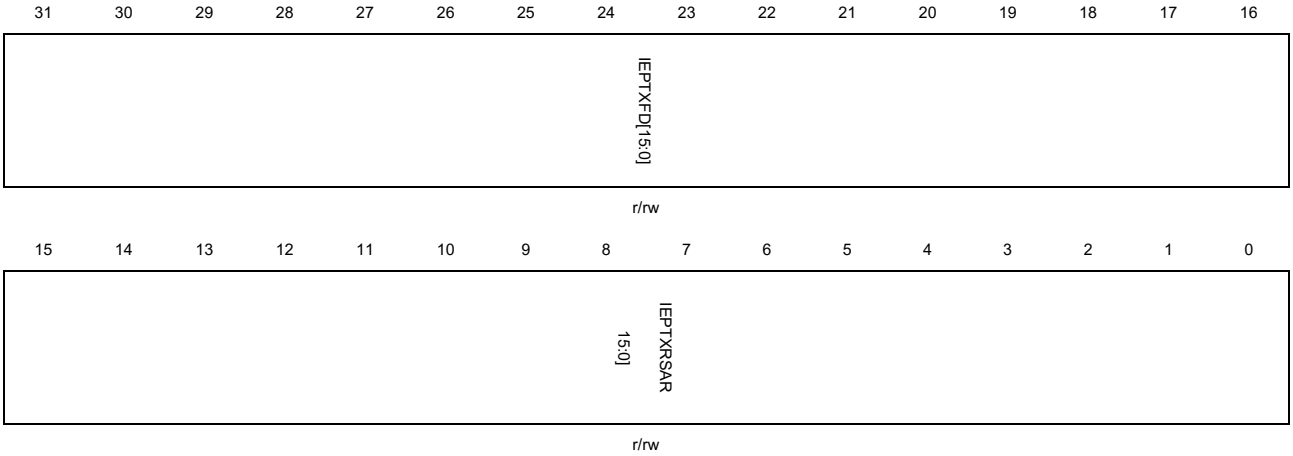



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | HPTXFD[15:0] | 主机周期性Tx FIFO深度<br>以32位字计数<br>1<=HPTXFD<=1024 |
| 15:0 | HPTXFSAR[15:0] | 主机周期性Tx RAM起始地址<br>主机周期性发送FIFO RAM起始地址 |

#### 设备 IN 端点发送 FIFO 长度寄存器(USBFS_DIEPxTFLEN) (x = 1..3，其中 x 为 FIFO 编号)

地址偏移：0x0104 + (FIFO编号 - 1) × 0x04
复位值：0x0200 0400

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | IEPTXFD[15:0] | IN端点Tx FIFO深度<br>以32位字计数<br>1<=HPTXFD<=1024 |
| 15:0 | IEPTXRSAR[15:0] | IN端点FIFOx Tx RAM起始地址<br>以32位字为单位的IN端点发送FIFOx起始地址 |

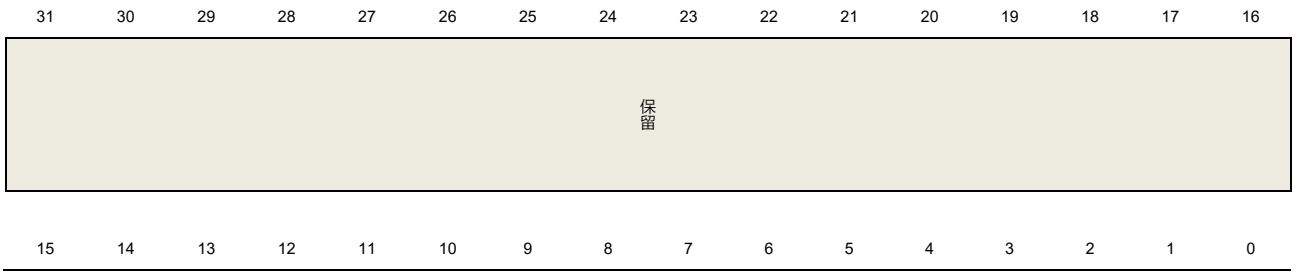
## 28.6.2. 主机控制和状态寄存器

#### 主机帧间隔寄存器(USBFS_HFT)

地址偏移：0x0404
复位值：0x0000 BB80

当USBFS控制器正在枚举中时，该寄存器为当前枚举速度设置帧间隔。

该寄存器只能按字(32位)访问



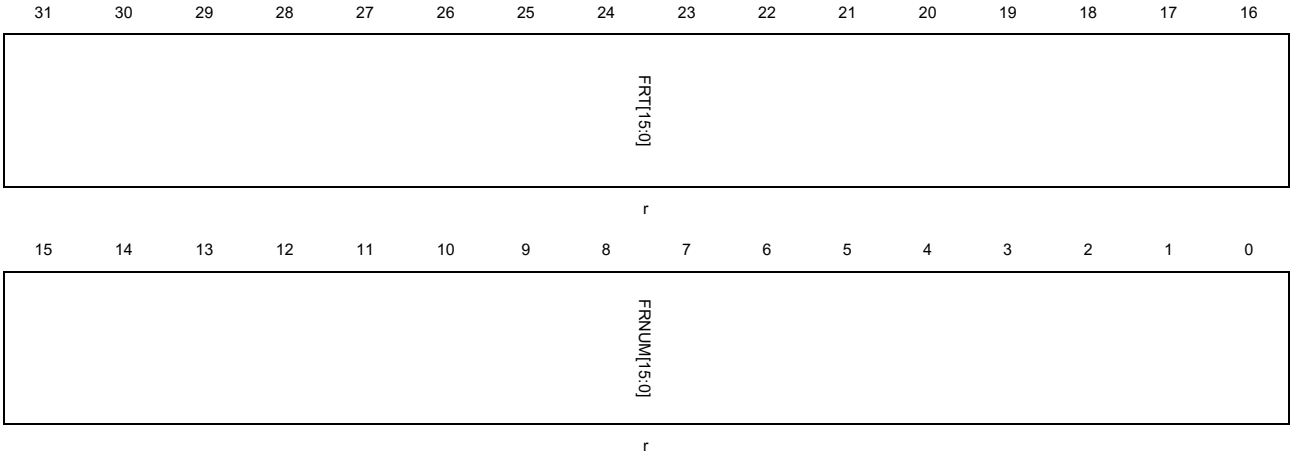



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | FRI[15:0] | 帧间隔<br>该值描述了以PHY时钟为单位的帧周期。每次端口复位操作后，端口被使能，USBFS根据当前速度，采用一个固有值，并且软件可以向该位域写值以改变该固有值。该值需要采用以下描述的频率来进行计算：<br>全速：48MHz<br>低速：6MHz |

### 主机帧信息保持寄存器(USBFS_HFINFR)

地址偏移：0x0408
复位值：0xBB80 0000

该寄存器只能按字(32位)访问



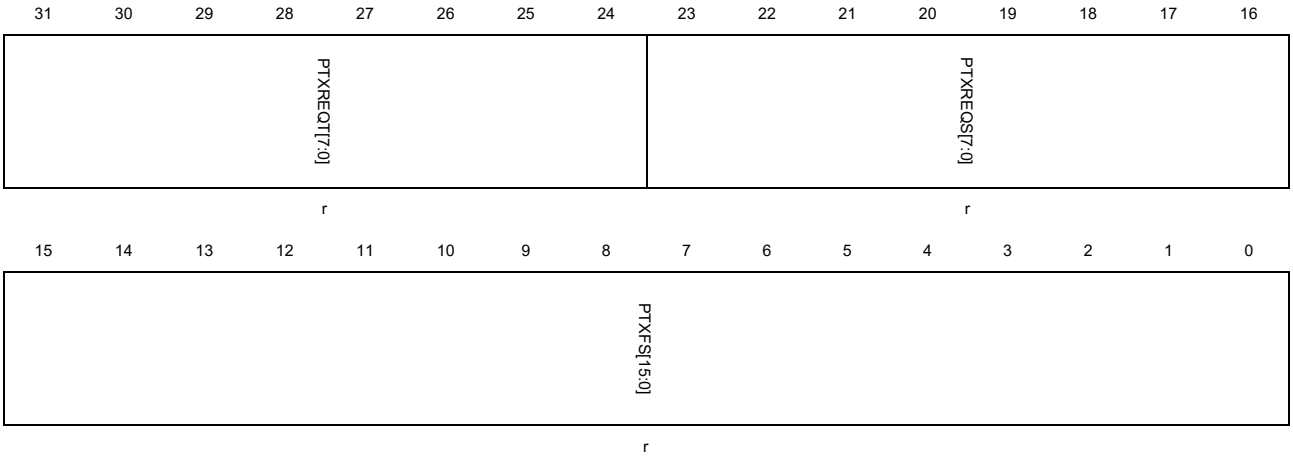

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | FRT[15:0] | 帧剩余时间<br>该位域以PHY时钟为单位反映了当前帧剩余时间。 |
| 15:0 | FRNUM[15:0] | 帧号<br>该位域反映了当前帧的帧号，当其增加到0x3FFF后，其值变为0。 |

### 主机周期性发送 FIFO/队列状态寄存器(USBFS_HPTFQSTAT)

地址偏移：0x0410
复位值：0x0008 0200

该寄存器反映了主机周期性Tx FIFO和请求队列的当前状态。请求队列包括在主机模式下的IN、OUT或其他请求条目。

该寄存器只能按字(32位)访问



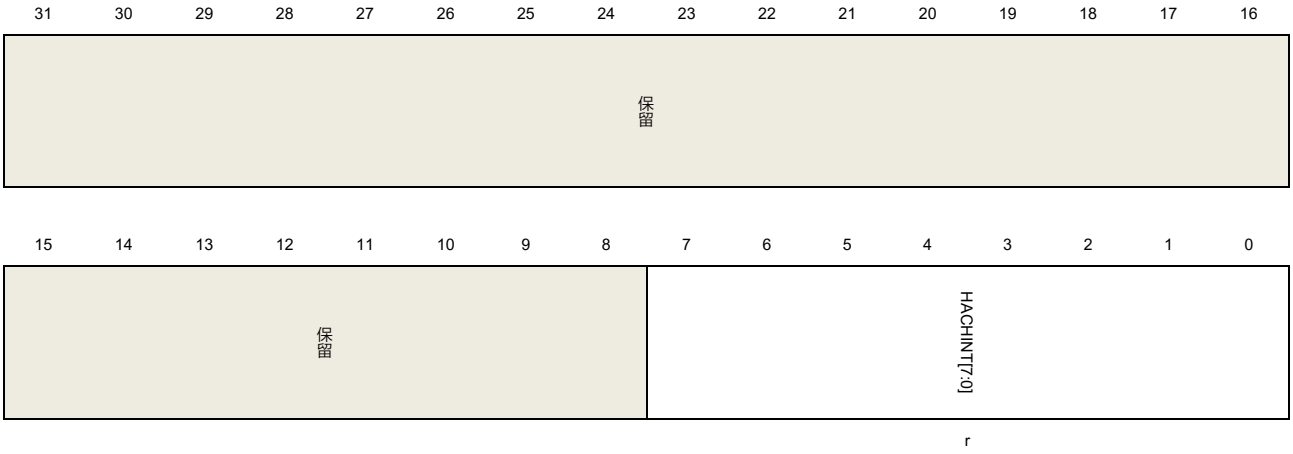

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | PTXREQT[7:0] | 周期性Tx请求队列的顶部条目<br>在周期性发送请求队列中的条目<br>位30:27：通道号<br>位26:25：<br>- 00：IN/OUT 令牌<br>- 01：0长度OUT包<br>- 11：通道中止请求<br>位24：中止标志，指示所选通道的最后一个条目 |
| 23:16 | PTXREQS[7:0] | 周期性发送请求队列空间<br>周期性发送请求队列剩余空间<br>0：请求队列为空<br>1：1个条目<br>2：2个条目<br>…<br>n：n个条目 (0<=n<=8)<br>其他：保留 |
| 15:0 | PTXFS[15:0] | 周期性发送FIFO空间<br>周期性发送FIFO剩余空间<br>以32位字计数<br>0：周期性发送FIFO为空<br>1：1个字<br>2：2个字<br>n：n个字(0<=n<=PTXFD))<br>其他：保留 |

### 主机所有通道中断寄存器(USBFS_HACHINT)

地址偏移：0x0414
复位值：0x0000 0000

当触发一个通道中断时，USBFS在该寄存器中置位相应的位，并且软件可以读取该寄存器以获取产生中断的通道。

该寄存器只能按字(32位)访问



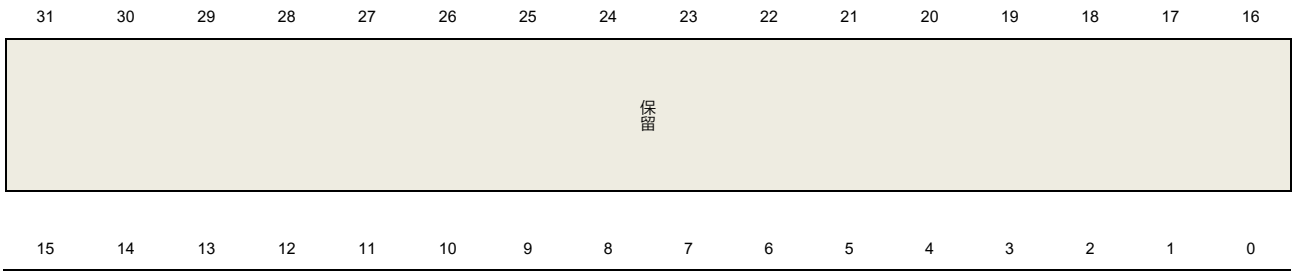

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7:0 | HACHINT[7:0] | 主机所有通道中断<br>每一位表示一个通道：位0代表通道0，位7表示通道7 |

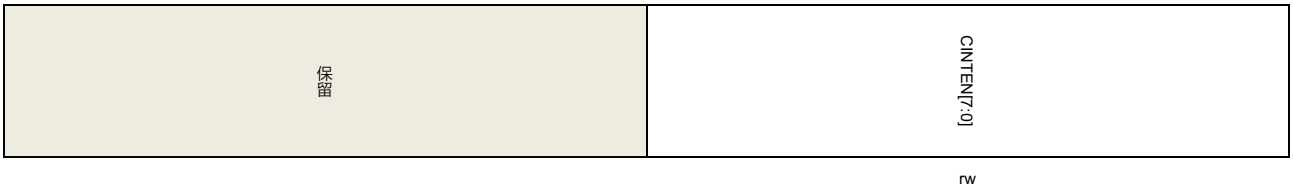
### 主机所有通道中断使能寄存器(USBFS_HACHINTEN)

地址偏移：0x0418
复位值：0x0000 0000

软件可以使用该寄存器使能或禁用一个通道的中断。只有该寄存器中相应通道的中断使能控制位被置位，USBFS_GINTF寄存器中的通道中断标志位HCIF标志位才可产生。

该寄存器只能按字(32位)访问





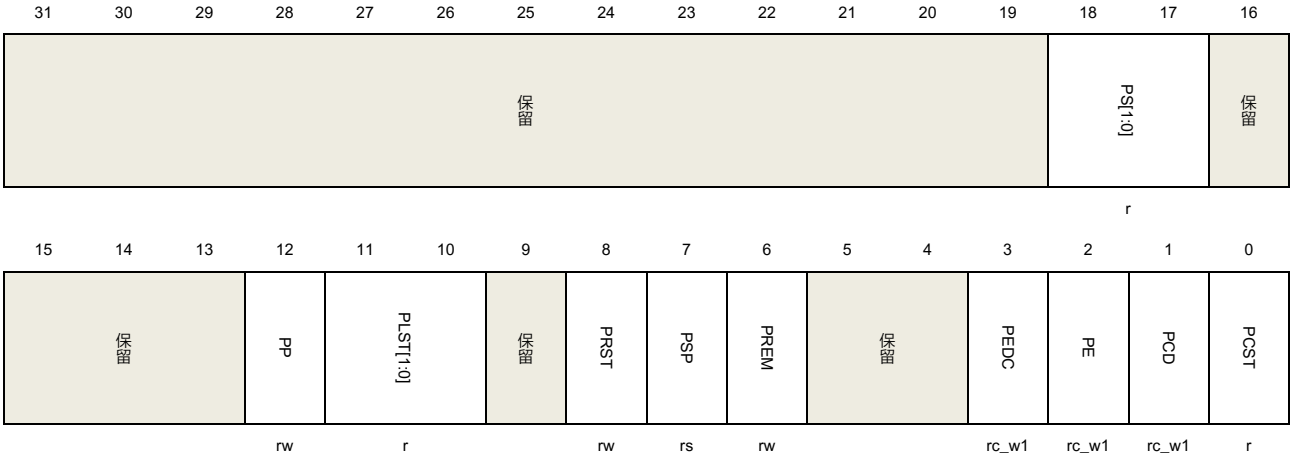

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7:0 | CINTEN[7:0] | 通道中断使能<br>0：禁用通道n中断<br>1：使能通道n中断<br>每一位表示一个通道：位0代表通道0，位7代表通道7 |

### 主机端口控制和状态寄存器(USBFS_HPCS)

地址偏移：0x0440
复位值：0x0000 0000

该寄存器控制端口行为，并且也包含一些反映端口状态的标志位。如果本寄存器中的PRST、PEDC和PCD标志位被USBFS置位的话，USBFS_GINTF寄存器中的HPIF标志位会被置位。

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:19 | 保留 | 必须保持复位值。 |
| 18:17 | PS[1:0] | 端口速度<br>反映连接到该端口的设备的枚举速度。<br>01：全速<br>10：低速<br>其他：保留 |
| 16:13 | 保留 | 必须保持复位值。 |
| 12 | PP | 端口供电 |

| | | |
|---|---|---|
| | | 在端口被使用后，该控制位应该被置位。由于USBFS不具有电源供应能力，它只能使用该控制位以获取该端口是否在供电状态。软件应该在设置该控制位之前，保证在$V_{BUS}$引脚上具有电源供应。<br>0：端口掉电<br>1：端口供电 |
| 11:10 | PLST[1:0] | 端口线状态<br>反映USB数据线当前状态<br>位10：DP线状态<br>位11：DM线状态 |
| 9 | 保留 | 必须保持复位值。 |
| 8 | PRST | 端口复位<br>应用通过设置该控制位以在USB端口上启动一个复位信号。当应用希望停止复位信号时，应用应该清除该控制位。<br>0：端口不在复位状态<br>1：端口处于复位状态 |
| 7 | PSP | 端口挂起<br>应用设置该控制位来将端口进入挂起状态。当该控制位被置位后，端口停止发送SOF令牌包。该控制位只能够通过以下操作清除。<br>– 应用置位该寄存器中的PRST控制位<br>– 置位该寄存器中的PREM控制位<br>– 检测到一个远程唤醒信号<br>– 检测到一个设备断开<br>0：端口不在挂起状态<br>1：端口处于挂起状态 |
| 6 | PREM | 端口恢复<br>应用通过置位该控制位以在USB端口上启动一个恢复信号。当应用希望停止恢复信号时，应用可以清除该控制位。<br>0：无恢复驱动<br>1：恢复驱动 |
| 5:4 | 保留 | 必须保持复位值。 |
| 3 | PEDC | 端口使能/禁止更改<br>当该寄存器中的位2端口使能控制位更改时，USB内核置位该标志位。 |
| 2 | PE | 端口使能<br>当USB复位信号完成后，USBFS自动置位该位，并且该位不可由软件置位。<br>该位可通过以下事件清除：<br>– 一个断开状态<br>– 软件清除该位<br>0：端口禁止<br>1：端口使能 |
| 1 | PCD | 端口连接检测 |

| | | |
|---|---|---|
| | | 当检测到设备连接时，USBFS置位该标志位。可通过向该位写1清除该标志位。 |
| 0 | PCST | 端口连接状态<br>0：设备没有连接到该端口<br>1：设备连接到该端口 |

### 主机通道 x 控制寄存器(USBFS_HCHxCTL) (x = 0…7，其中 x 为主机通道号)

地址偏移：0x0500 + (主机通道号 × 0x20)
复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CEN | CDIS | ODDFRM | DAR[6:0] | | | | | | | 保留 | | EPTYPE[1:0] | | LSD | 保留 |
| rs | rs | rw | rw | | | | | | | | | rw | | rw | |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | CEN | 通道使能<br>由应用设置，并且由USBFS清除<br>0：通道禁止<br>1：通道使能<br>软件应该遵循操作指南来禁用或者使能一个通道 |
| 30 | CDIS | 通道禁止<br>软件可以置位该控制位，来从处理事务中禁用该通道。软件应该遵循操作指南来禁用或者使能一个通道。 |
| 29 | ODDFRM | 奇偶帧控制<br>对于周期性传输(中断或同步传输)，该位控制将要处理的通道事务为奇数帧还是偶数帧。 |
| 28:22 | DAR[6:0] | 设备地址<br>与该通道通信的USB设备地址。 |
| 21:20 | 保留 | 必须保持复位值。 |
| 19:18 | EPTYPE[1:0] | 端点类型<br>与该通道通信的端点的传输类型<br>00：控制 |

| | | |
|---|---|---|
| | | 01：同步<br>10：批量<br>11：中断 |
| 17 | LSD | 低速设备<br>与该通道通信的设备是一个低速设备。 |
| 16 | 保留 | 必须保持复位值。 |
| 15 | EPDIR | 端点方向<br>与该通道通信的端点的传输方向<br>0：OUT<br>1：IN |
| 14:11 | EPNUM[3:0] | 端点号<br>与该通道通信的端点号 |
| 10:0 | MPL[10:0] | 最大包长<br>目标端点的最大包长 |

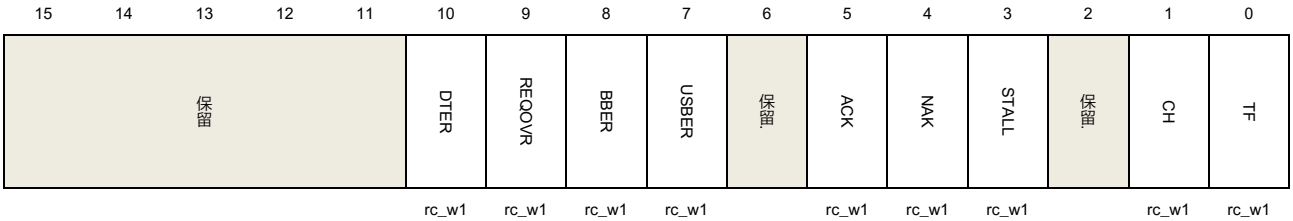
### 主机通道 x 中断标志寄存器(USBFS_HCHxINTF)(x = 0…7，其中 x 为主机通道号)

地址偏移：0x0508 + (主机通道号 × 0x20)
复位值：0x0000 0000

该寄存器包含一个通道的状态和事件，当软件获取一个通道中断时，软件需要为相应通道读取该寄存器以获取产生中断的中断源。该寄存器中的标志位均由硬件置位，并且写1清除。

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:11 | 保留 | 必须保持复位值。 |
| 10 | DTER | 数据切换错误<br>IN事务获取一个数据包，但是该包的PID和USBFS_HCHxLEN寄存器中的DPID[1:0]控制位不匹配。 |

| | | |
|---|---|---|
| 9 | REQOVR | 请求队列上溢<br>当软件启动新的传输时，请求队列上溢。 |
| 8 | BBER | 串扰错误<br>USB总线上发生一个串扰事件。产生串扰事件的典型原因是端点发送了一个数据包，但是数据包长度超过了端点的最大包长。 |
| 7 | USBER | USB总线错误<br>当在接收一个数据包的过程中，发生以下事件时，将置位USB总线错误标志位：<br>– 接收包有一个错误的CRC域<br>– 在USB总线上检测到填充错误<br>– 当等待一个响应包时，超时 |
| 6 | 保留 | 必须保持复位值。 |
| 5 | ACK | ACK<br>接收或者发送一个ACK响应包 |
| 4 | NAK | NAK<br>接收到一个NAK响应包 |
| 3 | STALL | STALL<br>接收到一个STALL响应包 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | CH | 通道中止<br>通道被当前请求所禁用，在当前请求处理的过程中，并不响应其他请求处理。 |
| 0 | TF | 发送完成<br>该通道所有的事务成功完成并且无错误发生。<br>对于IN通道，在USBFS_HCHxLEN寄存器的PCNT位减到0后，该标志位被置位。<br>对于OUT通道，当软件从RxFIFO中读取和取出一个TF状态条目时，该标志位被置位。 |

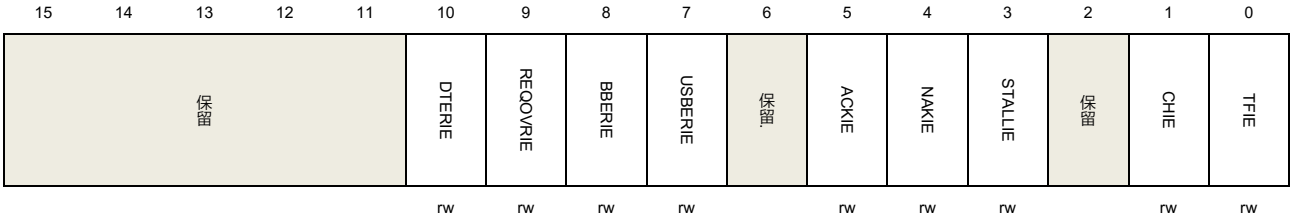
### 主机通道 x 中断使能寄存器(USBFS_HCHxINTEN)(x = 0…7，其中 x 为主机通道号)

地址偏移：0x050C + (主机通道号 × 0x20)
复位值：0x0000 0000

该寄存器包含USBFS_HCHxINTF寄存器内中断标志位的中断使能位。如果该寄存器的某位被软件置位，USBFS_HCHxINTF寄存器内的相应位能够触发一个通道中断。该寄存器内的位可由软件置位和清除。

该寄存器只能按字(32位)访问

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

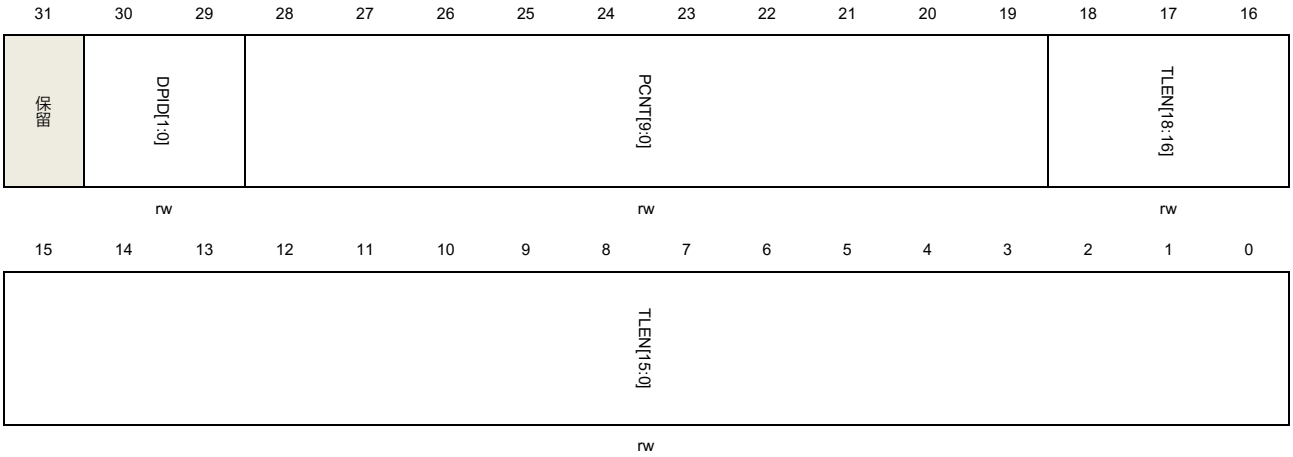



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:11 | 保留 | 必须保持复位值。 |
| 10 | DTERIE | 数据切换错误中断使能<br>0：禁用数据切换错误中断<br>1：使能数据切换错误中断 |
| 9 | REQOVRIE | 请求队列上溢中断使能<br>0：禁用请求队列上溢中断<br>1：使能请求队列上溢中断 |
| 8 | BBERIE | 串扰错误中断使能<br>0：禁用串扰错误中断<br>1：使能串扰错误中断 |
| 7 | USBERIE | USB总线错误中断使能<br>0：禁用USB总线错误中断<br>1：使能USB总线错误中断 |
| 6 | 保留 | 必须保持复位值。 |
| 5 | ACKIE | ACK中断使能<br>0：禁用ACK中断<br>1：使能ACK中断 |
| 4 | NAKIE | NAK中断使能<br>0：禁用NAK中断<br>1：使能NAK中断 |
| 3 | STALLIE | STALL中断使能<br>0：禁用STALL中断<br>1：使能STALL中断 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | CHIE | 通道中止中断使能<br>0：禁用通道中止中断<br>1：使能通道中止中断 |
| 0 | TFIE | 传输完成中断使能<br>0：禁用传输完成中断<br>1：使能传输完成中断 |

### 主机通道 x 长度寄存器(USBFS_HCHxLEN)(x = 0…7，其中 x 为主机通道号)

地址偏移：0x0510 + (主机通道号 × 0x20)
复位值：0x0000 0000

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30:29 | DPID[1:0] | 数据PID<br>软件应该在传输起始之前写该段位域。对于OUT传输，该位域包含第一个传输包的数据PID。对于IN传输，该位域包含第一个接收包的数据PID，并且如果数据PID不匹配的话，将会触发DTER标志位。在传输开始之后，USBFS遵循USB协议自动改变和切换该位域。<br>00：DATA0<br>10：DATA1<br>11：SETUP(仅对于控制传输)<br>01：保留 |
| 28:19 | PCNT[9:0] | 包计数<br>在一个传输中希望发送(OUT)或接收(IN)的数据包个数。<br>软件应该在通道使能之前写该位域。在传输启动之后，该位域在USBFS正确传输每个数据包后，自动减少。 |
| 18:0 | TLEN[18:0] | 传输长度<br>一次传输的总数据字节数。<br>对于OUT传输，该位域为OUT传输中期望发送的所有数据包总数据字节数。软件应该在通道使能之前写该位域。当软件正确向通道的数据FIFO中写入一个包时，该位域以包中字节大小进行减少。<br>对于IN传输，每次软件从RxFIFO中读取一个包后，该位域也以包中字节大小进行减少。 |

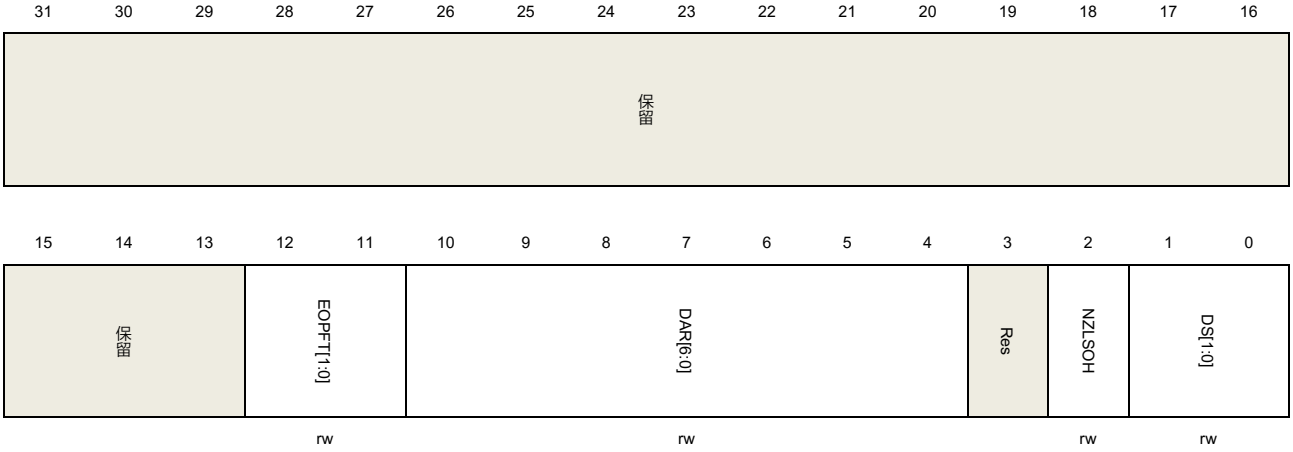
## 28.6.3. 设备控制和状态寄存器

### 设备配置寄存器(USBFS_DCFG)

地址偏移：0x0800
复位值：0x0000 0000

在上电、枚举或执行某些控制命令后，该寄存器配置内核为设备模式。在设备初始化后，不可以改变该寄存器值。

该寄存器采用字(32位)访问



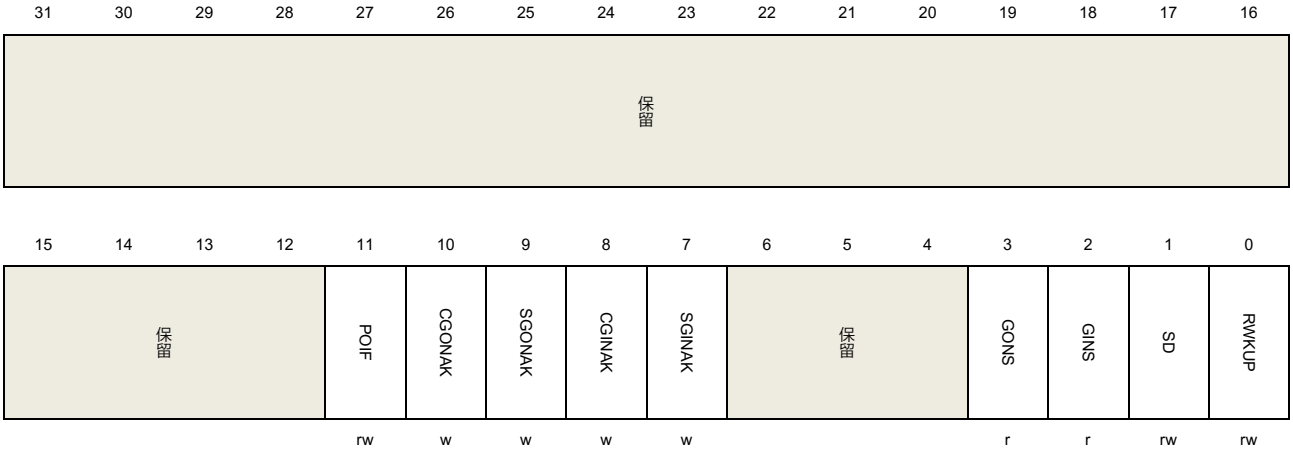

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:13 | 保留 | 必须保持复位值。 |
| 12:11 | EOPFT[1:0] | 周期性帧尾时间<br>该域定义周期性帧时间的帧尾标志触发的时间点<br>00：80%的帧时间<br>01：85%的帧时间<br>10：90%的帧时间<br>11：95%的帧时间 |
| 10:4 | DAR[6:0] | 设备地址<br>该位定义USB设备地址，USBFS采用该位匹配接收的设备令牌地址域，在接收到来自主机的设置地址的命令后，软件设置该域 |
| 3 | 保留 | 必须保持复位值。 |
| 2 | NZLSOH | 非零长度OUT状态阶段握手<br>在控制传输的OUT状态阶段，当USB设备接收到一个非零长度数据包时，该域控制控制USBFS是接收该包，还是用STALL握手信号拒绝该包。<br>0：把该包视为正常包，根据设备OUT端点控制寄存器的NAKS和STALL位，回复握手相应握手包<br>1：发送STALL握手，不保存接收到的OUT数据包 |

| | | |
|---|---|---|
| 1:0 | DS[1:0] | 设备速度<br>该域控制设备连入主机后的设备速度<br>11：全速<br>其他：保留 |

### 设备控制寄存器(USBFS_DCTL)

地址偏移：0x0804
复位值：0x0000 0000

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11 | POIF | 上电初始化完成<br>软件通过设置该位，通知USBFS寄存器在从掉电模式下唤醒，然后完成初始化。 |
| 10 | CGONAK | 清零全局OUT NAK<br>软件设置该位从而清零该寄存器的GONS位 |
| 9 | SGONAK | 设置全局OUT NAK<br>软件设置该位从而实现该寄存器的位GONS置位。<br>当GONS位为零，设置该位会引起USBFS_GINTF寄存器的GONAK标志触发，软件应该在再写该位前清除GONAK标志。 |
| 8 | CGINAK | 清零全局IN NAK<br>软件设置该位从而清零该寄存器的GINS位 |
| 7 | SGINAK | 设置全局IN NAK<br>软件设置该位从而实现该寄存器的位GINS置位<br>当GINS位为零，设置该位会引起USBFS_GINTF寄存器的GINAK标志触发，软件应该在再写该位前清除GINAK标志。 |
| 6:4 | 保留 | 必须保持复位值。 |
| 3 | GONS | 全局OUT NAK状态 |

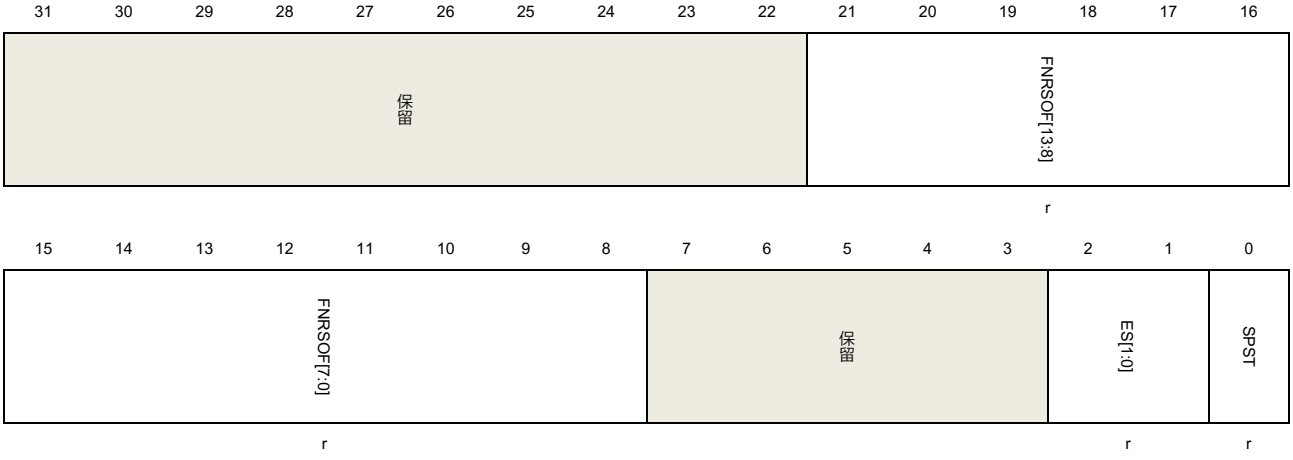
| | | |
|---|---|---|
| | | 0：USBFS回复OUT事务的握手信号以及是否保存OUT数据包由Rx FIFO状态、端点的NAKS、STALL位确定。<br>1：USBFS回复OUT事务NAK握手信号，不保存接收的OUT数据包。 |
| 2 | GINS | 全局IN NAK状态<br>0：USBFS回复IN事务的握手信号由Tx FIFO状态、端点的NAKS、STALL位确定。<br>1：USBFS通常回复IN事务NAK握手信号 |
| 1 | SD | 软断开<br>软件可实现USB总线上的软断开，在置1该位后，关掉DP线上的上拉电阻，从而引起主机检测设备的断开。<br>0：没有软断开生成<br>1：生成软断开 |
| 0 | RWKUP | 远程唤醒<br>在挂起状态，软件可通过该位来生成一个远程唤醒信号来通知主机恢复USB总线<br>0：没有远程唤醒信号生成<br>1：生成远程唤醒信号 |

### 设备状态寄存器(USBFS_DSTAT)

地址偏移：0x0808
复位值：0x0000 0000

该寄存器包含设备模式下的 USBFS 的状态和信息。

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:22 | 保留 | 必须保持复位值。 |
| 21:8 | FNRSOF[13:0] | 所接收的SOF帧编号<br>USBFS会在接收到一个SOF令牌后更新该域。 |
| 7:3 | 保留 | 必须保持复位值。 |

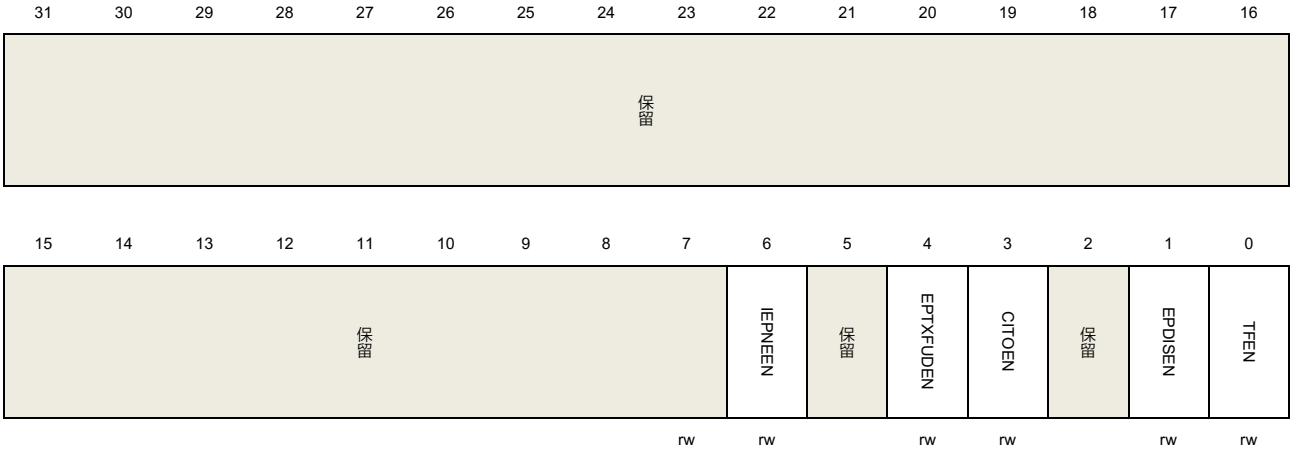
| | | |
|---|---|---|
| 2:1 | ES[1:0] | 枚举速度<br>该域指示所枚举的设备速度，在寄存器USBFS_GINTF的ENUMF标志触发后，软件可以读取该域。<br>01：全速<br>其他：保留 |
| 0 | SPST | 挂起状态<br>该位指示设备是否处于挂起状态。<br>0：设备不在挂起状态<br>1：设备在挂起状态 |

### 设备 IN 端点通用中断使能寄存器(USBFS_DIEPINTEN)

地址偏移：0x810
复位值：0x0000 0000

该寄存器包含寄存器 USBFS_DIEPxINTF 中的标志的中断使能位，如果软件置 1 某位，其在寄存器 USBFS_DIEPxINTF 中对应的位可以触发一个寄存器 USBFS_DAEPINT 端点中断。该位可以通过软件置位和清零。

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:7 | 保留 | 必须保持复位值。 |
| 6 | IEPNEEN | IN端点NAK有效中断使能位<br>0：除能中断<br>1：使能中断 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | EPTXFUDEN | 端点Tx FIFO下溢中断使能位<br>0：除能中断<br>1：使能中断 |
| 3 | CITOEN | 控制IN事务超时中断使能位 |

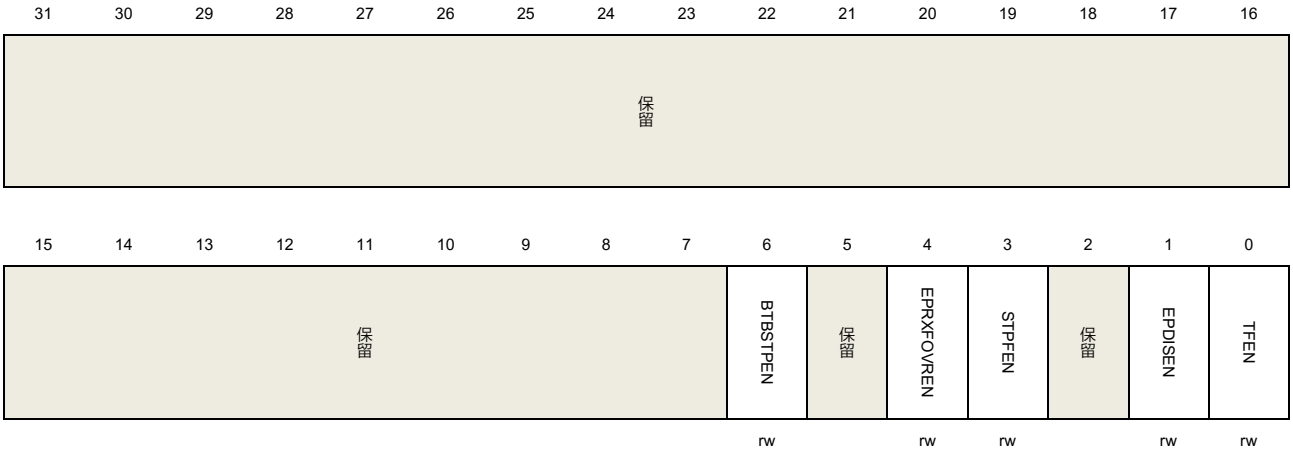
| | | |
|---|---|---|
| | | 0：除能中断<br>1：使能中断 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | EPDISEN | 端点除能中断使能位<br>0：除能中断<br>1：使能中断 |
| 0 | TFEN | 传输完成中断使能位<br>0：除能中断<br>1：使能中断 |

### 设备 OUT 端点通用中断使能寄存器(USBFS_DOEPINTEN)

地址偏移：0x0814
复位值：0x0000 0000

该寄存器包含寄存器 USBFS_DOEPxINTF 中的标志的中断使能位，如果软件置 1 某位，其在寄存器 USBFS_DOEPxINTF 中对应的位可以触发一个寄存器 USBFS_DAEPINT 端点中断。该位可以通过软件置位和清零。

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:6 | 保留 | 必须保持复位值。 |
| 6 | BTBSTPEN | 连续SETUP包中断使能位(仅适用于控制OUT端点)<br>0：除能中断<br>1：使能中断 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | EPRXFOVREN | 端点Rx FIFO上溢中断使能位<br>0：除能中断<br>1：使能中断 |

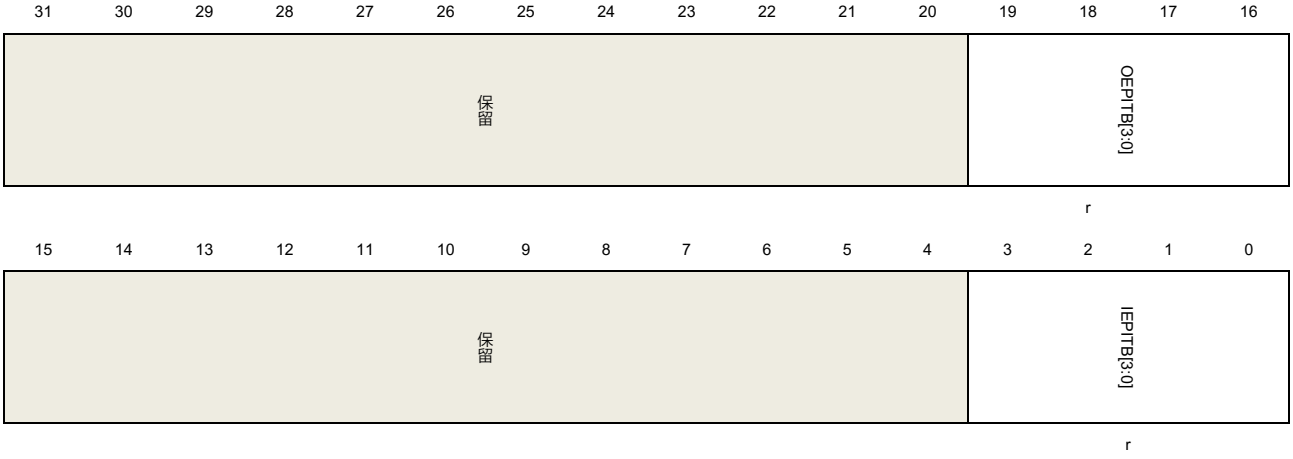
| | | |
|---|---|---|
| 3 | STPFEN | SETUP阶段完成中断使能位(仅适用于控制OUT端点)<br>0：除能中断<br>1：使能中断 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | EPDISEN | 端点除能中断使能位<br>0：除能中断<br>1：使能中断 |
| 0 | TFEN | 传输完成中断使能位<br>0：除能中断<br>1：使能中断 |

### 设备端点中断寄存器(USBFS_DAEPINT)

地址偏移：0x0818
复位值：0x0000 0000

当一个端点的中断被触发，USBFS 置 1 该寄存器的相应位，软件可通过该寄存器知道在本次中断中的端点号。

该寄存器采用字(32位)访问



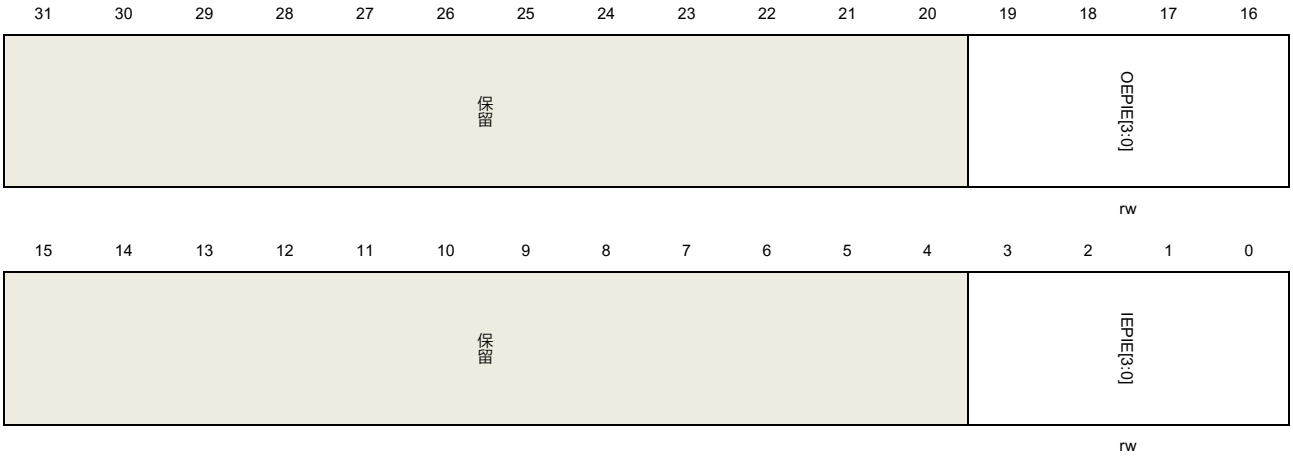

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:20 | 保留 | 必须保持复位值。 |
| 19:16 | OEPITB[3:0] | 设备OUT端点中断位<br>每个位代表一个OUT端点：Bit16代表OUT端点0，Bit19代表OUT端点3 |
| 15:4 | 保留 | 必须保持复位值。 |
| 3:0 | IEPITB[3:0] | 设备IN端点中断位<br>每个位代表一个IN端点：Bit0代表IN端点0，Bit3代表IN端点3 |

### 设备端点中断使能寄存器(USBFS_DAEPINTEN)

地址偏移：0x081C
复位值：0x0000 0000

该寄存器可通过软件使能或除能端点的中断，只有当端点在该寄存器中相应位被置 1 才能触发寄存器 USBFS_GINTF 的端点中断标志 OEPIF 或 IEPIF。

该寄存器采用字(32位)访问



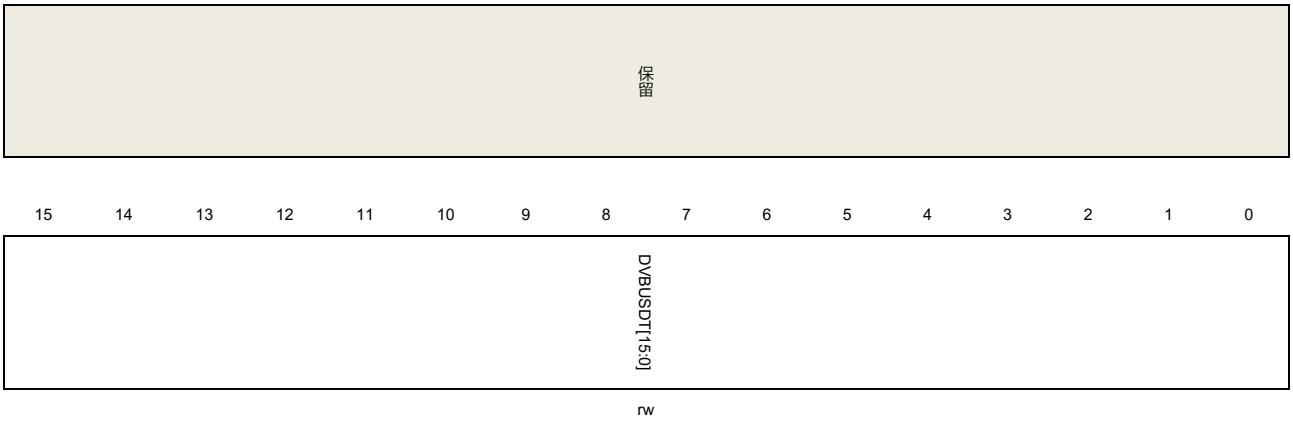

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:20 | 保留 | 必须保持复位值。 |
| 19:16 | OEPIE[3:0] | OUT端点中断使能位<br>0：除能OUT端点n中断<br>1：使能OUT端点n中断<br>每个位代表一个OUT端点：Bit16对应OUT端点0，Bit19对应OUT端点3 |
| 15:4 | 保留 | 必须保持复位值。 |
| 3:0 | IEPIE[3:0] | IN端点中断使能位<br>0：除能IN端点n中断<br>1：使能IN端点n中断<br>每个位代表一个IN端点：Bit0对应IN端点0，Bit3对应IN端点3 |

### 设备 VBUS 放电时间寄存器(USBFS_DVBUSDT)

地址偏移：0x0828
复位值：0x0000 17D7

该寄存器采用字(32位)访问



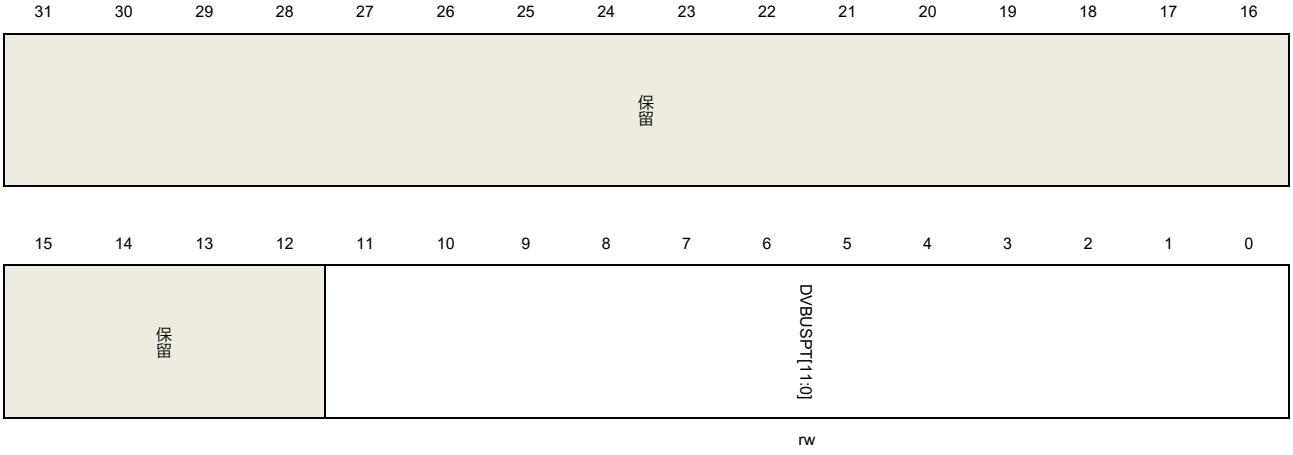



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | DVBUSDT[15:0] | 设备$V_{BUS}$放电时间<br>在SRP协议中，在$V_{BUS}$脉冲产生后，有一个放电过程，该域定义了$V_{BUS}$的放电时间，真正的放电时间是1024*DVBUSDT[15:0] *$T_{USBCLOCK}$，$T_{USBCLOCK}$是USB时钟周期时间。 |

### 设备 VBUS 脉冲时间寄存器(USBFS_DVBUSPT)

地址偏移：0x082C
复位值：0x0000 05B8

该寄存器采用字(32位)访问



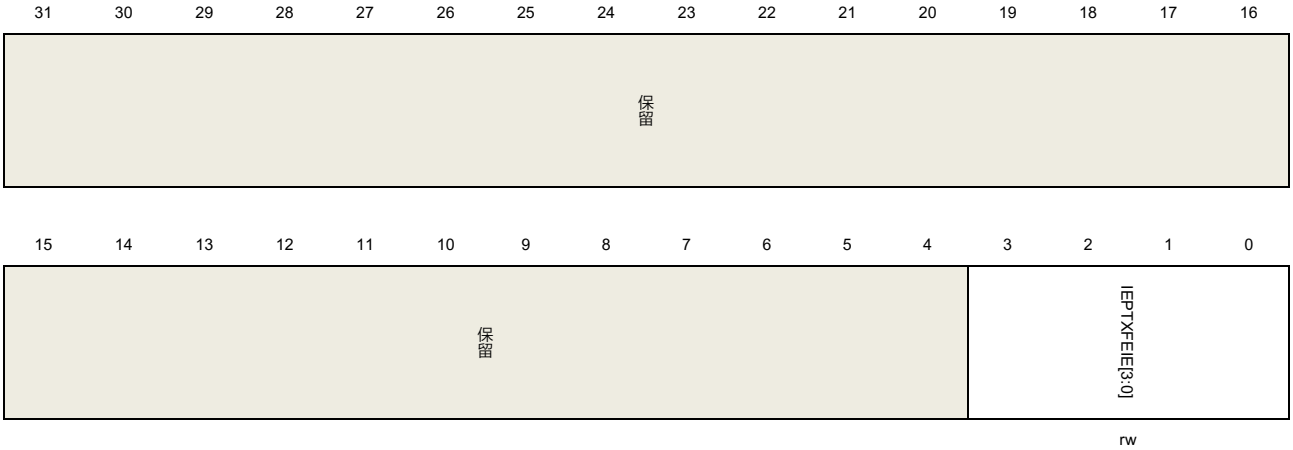

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11:0 | DVBUSPT[11:0] | 设备$V_{BUS}$脉冲时间<br>该域定义$V_{BUS}$的脉冲时间，真正的充电时间是1024*DVBUSPT[15:0] *$T_{USBCLOCK}$，$T_{USBCLOCK}$是USB时钟周期时间 |

### 设备 IN 端点 FIFO 空中断使能寄存器(USBFS_DIEPFEINTEN)

地址偏移：0x0834
复位值：0x0000 0000

该寄存器包含 IN 端点 Tx FIFO 空中断的使能位

寄存器采用字(32位)访问

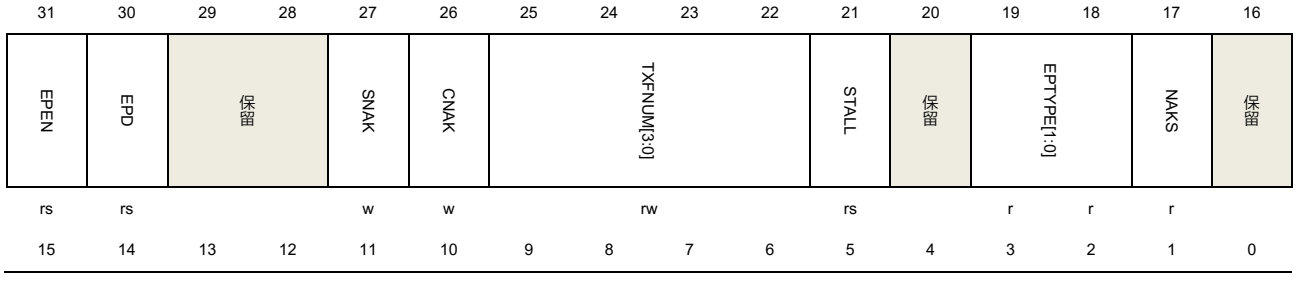



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:4 | 保留 | 必须保持复位值。 |
| 3:0 | IEPTXFEIE[3:0] | IN端点Tx FIFO空中断的使能位<br>该域控制着USBFS_DIEPxINTF寄存器的TXFE位能否生成一个寄存器USBFS_DAEPINT的端点中断位<br>Bit0对应IN端点0，Bit5对应IN端点5<br>0：除能FIFO空中断<br>1：使能FIFO空中断 |

### 设备 IN 端点 0 控制寄存器(USBFS_DIEP0CTL)

地址偏移：0x0900
复位值：0x0000 8000

该寄存器采用字(32位)访问






| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | EPEN | 端点使能<br>软件置位、USBFS清零<br>0：端点除能<br>1：端点使能<br>软件应该按照操作指南使能或除能端点 |
| 30 | EPD | 端点除能<br>软件可通过置位该位从而除能端点，软件应该按照操作指南使能或除能端点。 |
| 29:28 | 保留 | 必须保持复位值。 |
| 27 | SNAK | 置位NAK<br>软件置位该位来设置该寄存器的NAKS位 |
| 26 | CNAK | 清零NAK<br>软件置位该位来清零该寄存器的NAKS位 |
| 25:22 | TXFNUM[3:0] | Tx FIFO编号<br>定义IN端点0的Tx FIFO编号 |
| 21 | STALL | STALL握手<br>当接收IN令牌时，软件可以通过置1该位发送STALL握手包，对于相应的OUT端点0，在接收SETUP令牌后，USBFS清除此位。该位比该寄存器的NAKS位和寄存器USBFS_DCTL的GINS位优先级要高，如果STALL和NAKS位都被置位，STALL位生效。 |
| 20 | 保留 | 必须保持复位值。 |
| 19:18 | EPTYPE[1:0] | 端点类型<br>该域固定为'00',控制端点。 |
| 17 | NAKS | NAK状态<br>当该寄存器的STALL位和寄存器USBFS_DCTL的位GINS被清零，该位控制USBFS的NAK状态。<br>0：根据端点Tx FIFO的状态，USBFS发送数据或握手包<br>1：USBFS总为IN令牌发送NAK握手包<br>该位是只读位，可以通过该寄存器的位CNAK和位SNAK控制该位 |
| 16 | 保留 | 必须保持复位值。 |
| 15 | EPACT | 端点激活<br>对于端点0来说，该域固定为'1' |

| | | |
|---|---|---|
| 14:2 | 保留 | 必须保持复位值。 |
| 1:0 | MPL[1:0] | 最大包长<br>域定义了控制数据包的最大包长，如USB 2.0协议所描述，对控制传输而言，有四种包长度:<br>00：64字节<br>01：32字节<br>10：16字节<br>11：8字节 |

### 设备 IN 端点 x 控制寄存器(USBFS_DIEPxCTL)(x = 1..3，x 是设备端点编号)

地址偏移：0x0900 + (设备端点编号 x 0x20)
复位值：0x0000 0000

该寄存器采用字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EPEN | EPD | SODDFRM/SD1PID | SD0PID/SEVENFRM | SNAK | CNAK | TXFNUM[3:0] | | | | STALL | 保留 | EPTYPE[1:0] | | NAKS | EOFRM/DPID |
| rs | rs | w | w | w | w | rw | | | | rw/rs | | rw | | r | r |

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

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | EPEN | 端点使能<br>软件置位，USBFS清零<br>0：端点除能<br>1：端点使能<br>软件应该按照操作指南使能或除能端点 |
| 30 | EPD | 端点除能<br>软件可通过置位该位从而除能端点，软件应该按照操作指南使能或除能端点。 |
| 29 | SODDFRM | 设置奇数帧(适用于同步IN端点)<br>软件通过置1该位置1该寄存器的EOFRM位 |
| | SD1PID | 设置DATA1 PID(适用于中断和大容量IN端点)<br>软件可通过置1该位置1该寄存器的DPID位 |
| 28 | SEVENFRM | 设置偶数帧(适用于同步IN端点)<br>软件通过置1该位清零该寄存器的EOFRM位 |

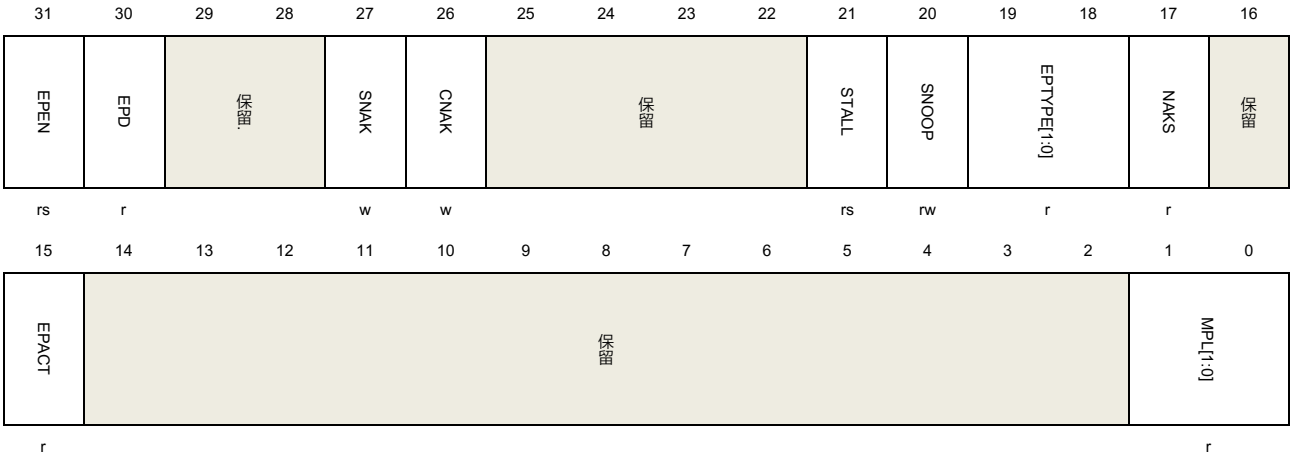
| | | |
|---|---|---|
| | SD0PID | 设置DATA0 PID(适用于中断和大容量IN端点)<br>软件可通过置1该位清零该寄存器的DPID位 |
| 27 | SNAK | 设置NAK<br>软件置1该位置1该寄存器的NAKS位 |
| 26 | CNAK | 清零NAK<br>软件置1该位清零该寄存器的NAKS位 |
| 25:22 | TXFNUM[3:0] | Tx FIFO编号<br>该位定义了IN端点的Tx FIFO编号 |
| 21 | STALL | STALL握手<br>当接收IN令牌时，软件可以通过置1该位发送STALL握手包。该位比该寄存器的NAKS位和寄存器USBFS_DCTL的GINS位优先级要高，如果STALL和NAKS位都被置位，STALL位生效。<br>对于控制IN端点：<br>当对应的OUT端点接收到SETUP令牌时，只有USBFS可以清零此位，软件不可清除此位。<br>对于中断或大容量IN端点：<br>只有软件可以清零此位。 |
| 20 | 保留 | 必须保持复位值。 |
| 19:18 | EPTYPE[1:0] | 端点类型<br>该域定义端点的传输类型：<br>00：控制<br>01：同步<br>10：大容量<br>11：中断 |
| 17 | NAKS | NAK状态<br>当该寄存器的STALL位和寄存器USBFS_DCTL的位GINS被清零，该位控制USBFS的NAK状态：<br>0：根据端点Tx FIFO的状态，USBFS发送数据或握手包<br>1：USBFS总为IN令牌发送NAK握手包<br>该位是只读位，可以通过该寄存器的位CNAK和位SNAK控制该位 |
| 16 | EOFRM | 奇偶帧(适用于同步IN端点)<br>对于同步传输，软件通过使用该位控制USBFS只在奇数帧或偶数帧为IN事务发送数据包，如果当前帧号的奇偶性不匹配该位，USBFS回复一个零长度的包：<br>0：只在偶数帧发送数据<br>1：只在奇数帧发送数据 |
| | DPID | 端点数据PID(适用于中断或大容量IN端点)<br>在端点或大容量传输中，有数据PID翻转机制，在传输开始之前，软件通过设定SD0PID来设置此位，按照USB协议中描述的数据PID翻转机制，USBFS在传输过程中保持该位。<br>0：数据包的PID是DATA0 |

| | | |
|---|---|---|
| | | 1：数据包的PID是DATA1 |
| 15 | EPACT | 端点激活<br>该位控制端点是否激活，当端点没有激活，忽略任何令牌，不做任何回复。 |
| 14:11 | 保留 | 必须保持复位值。 |
| 10:0 | MPL[10:0] | 该域定义最大包长 |

### 设备 OUT 端点 0 控制寄存器(USBFS_DOEP0CTL)

地址偏移: 0x0B00
复位值: 0x0000 8000

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | EPEN | 端点使能<br>软件置位，USBFS清零<br>0：端点除能<br>1：端点使能<br>软件应该按照操作指南使能或除能端点。 |
| 30 | EPD | 端点除能<br>对于OUT端点0，该位固定为0 |
| 29:28 | 保留 | 必须保持复位值。 |
| 27 | SNAK | 设置NAK<br>软件置1该位置1该寄存器的NAKS位 |
| 26 | CNAK | 清零NAK<br>软件置1该位清零该寄存器的NAKS位 |
| 25:22 | 保留 | 必须保持复位值。 |
| 21 | STALL | STALL握手 |

| | | |
|---|---|---|
| | | 在OUT事务中，软件可以通过置1该位发送STALL握手包，对于OUT端点0，在接收SETUP令牌后，USBFS清除此位。该位比该寄存器的NAKS位和寄存器USBFS_DCTL的GINS位优先级要高，即如果STALL和NAKS位都被置位，STALL位生效。 |
| 20 | SNOOP | 调查模式<br>该位控制OUT端点的调查模式，在调查模式中，USBFS不再检查接收数据包的CRC值<br>0：调查模式除能<br>1：调查模式使能 |
| 19:18 | EPTYPE[1:0] | 端点类型<br>对于控制端点，该位固定为"00" |
| 17 | NAKS | NAK状态<br>当该寄存器的STALL位和寄存器USBFS_DCTL的位GINS被清零，该位控制USBFS的NAK状态：<br>0：根据端点Rx FIFO的状态，USBFS发送数据或握手包<br>1：USBFS为OUT事务发NAK握手包<br>该位是只读位，通过该寄存器的CNAK和SNAK位控制该位 |
| 16 | 保留 | 必须保持复位值。 |
| 15 | EPACT | 端点激活<br>对于端点0，该域固定为1 |
| 14:2 | 保留 | 必须保持复位值。 |
| 1:0 | MPL[1:0] | 最大包长<br>该位是只读位，其数值来自于寄存器USBFS_DIEP0CTL的位MPL：<br>00：64字节<br>01：32字节<br>10：16字节<br>11：8字节 |

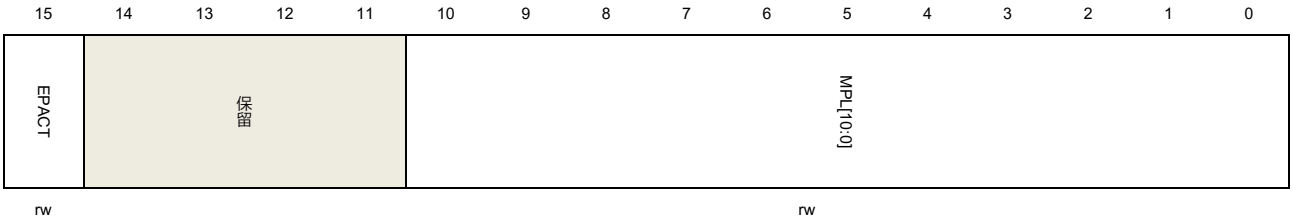
### 设备 OUT 端点 x 控制寄存器(USBFS_DOEPxCTL)(x= 1..3，x 是设备端点编号)

地址偏移：0x0B00 + (设备端点编号 x 0x20)
复位值：0x0000 0000

软件用该寄存器控制 OUT 端点 0 以外的每个逻辑 OUT 端点

该寄存器采用字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EPEN | EPD | SODDFRM/SD1PID | SEVENFRM/SD0PID | SNAK | CNAK | 保留 | | | | STALL | SNOOP | EPTYPE[1:0] | | NAKS | EOFRM/DPID |
| rs | rs | w | w | w | w | | | | | rw/rs | rw | rw | | r | r |




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | EPEN | 端点使能<br>软件置位，USBFS清零<br>0：端点除能<br>1：端点使能<br>软件应该按照操作指南使能或除能端点。 |
| 30 | EPD | 端点除能<br>软件通过置1该位除能端点，软件应该按照操作指南使能或除能端点。 |
| 29 | SODDFRM | 设置奇数帧(适用于同步OUT端点)<br>该位只针对同步OUT端点有效<br>软件置1该位来置位该寄存器的EOFRM位 |
| | SD1PID | 设置DATA1 PID(适用于中断和大容量OUT端点)<br>软件置1该位来置位该寄存器的DPID位 |
| 28 | SEVENFRM | 设置偶数帧(适用于同步OUT端点)<br>软件置1该位来清零该寄存器的EOFRM位 |
| | SD0PID | 设置DATA0 PID(适用于中断和大容量OUT端点)<br>软件置1该位来清零该寄存器的DPID位 |
| 27 | SNAK | 设置NAK<br>软件置1该位从而置1该寄存器的NAKS位 |
| 26 | CNAK | 清零NAK<br>软件置1该位从而清零该寄存器的NAKS位 |
| 25:22 | 保留 | 必须保持复位值。 |
| 21 | STALL | STALL握手<br>在OUT事务中，软件可以通过置1该位发送STALL握手包。该位比该寄存器的NAKS位和寄存器USBFS_DCTL的GINS位优先级要高，如果STALL和NAKS位都被置位，STALL位生效。<br>对于控制OUT端点：<br>当OUT端点接收SETUP令牌时，只有USBFS可以清零该位，软件不可清零此位。<br>对于中断或大容量OUT端点<br>只有软件可以清零该位 |
| 20 | SNOOP | 调查模式<br>该位控制OUT端点的调查模式，在调查模式中，USBFS不再检查接收数据包的CRC |

| | | |
|---|---|---|
| | | 值<br>0：调查模式除能<br>1：调查模式使能 |
| 19:18 | EPTYPE[1:0] | 端点类型<br>该域定义端点的传输类型<br>00：控制<br>01：同步<br>10：大容量<br>11：中断 |
| 17 | NAKS | NAK状态<br>当该寄存器的STALL位和寄存器USBFS_DCTL的位GONS被清零，该位控制USBFS的NAK状态：<br>0：根据端点的Rx FIFO的状态，发送握手包<br>1：USBFS为OUT事务发送NAK握手<br>该位是只读位，通过该寄存器的CNAK和SNAK位控制该位 |
| 16 | EOFRM | 奇偶帧(适用于同步OUT端点)<br>对于同步传输，软件通过使用该位控制USBFS只在奇数帧或偶数帧发送数据包给OUT事务，如果当前帧号的奇偶性不匹配该位，USBFS不保存数据包<br>0：只在偶数帧发送数据<br>1：只在奇数帧发送数据 |
| | DPID | 端点数据PID(适用于中断或大容量端点)<br>在端点或大容量传输中，有数据PID翻转机制，在传输开始之前，软件通过设定SD0PID来设置此位，按照USB协议中描述的数据PID翻转机制，USBFS在传输过程中保持该位。<br>0：数据包PID是DATA0<br>1：数据包PID是DATA1 |
| 15 | EPACT | 端点激活<br>位控制端点是否激活，当端点没有激活，忽略任何令牌，不做任何回复 |
| 14:11 | 保留 | 必须保持复位值。 |
| 10:0 | MPL[10:0] | 该位定义最大包长 |

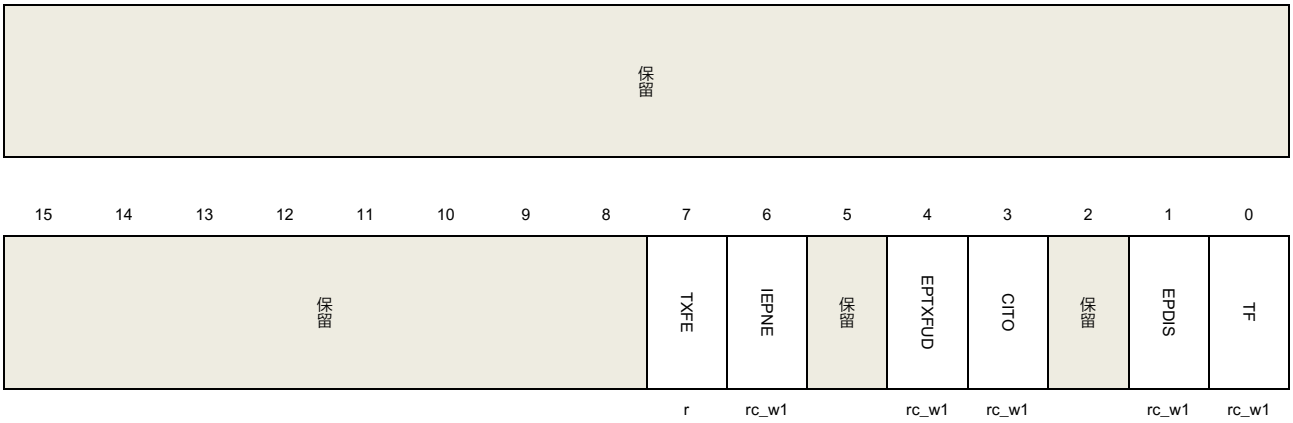
### 设备 IN 端点 x 中断标志寄存器(USBFS_DIEPxINTF)(x = 0..3，x 是设备端点编号)

地址偏移：0x0908 + (设备端点编号 x 0x20)
复位值：0x0000 0080

该寄存器包含 IN 端点的状态和事件，当获得一个 IN 端点的中断时，应该读取该端点的中断标志寄存器，从而获知中断源。该寄存器的标志位通常硬件置位，除了 TXFE 位，各位写 1 清零。

该寄存器采用字(32位)访问

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



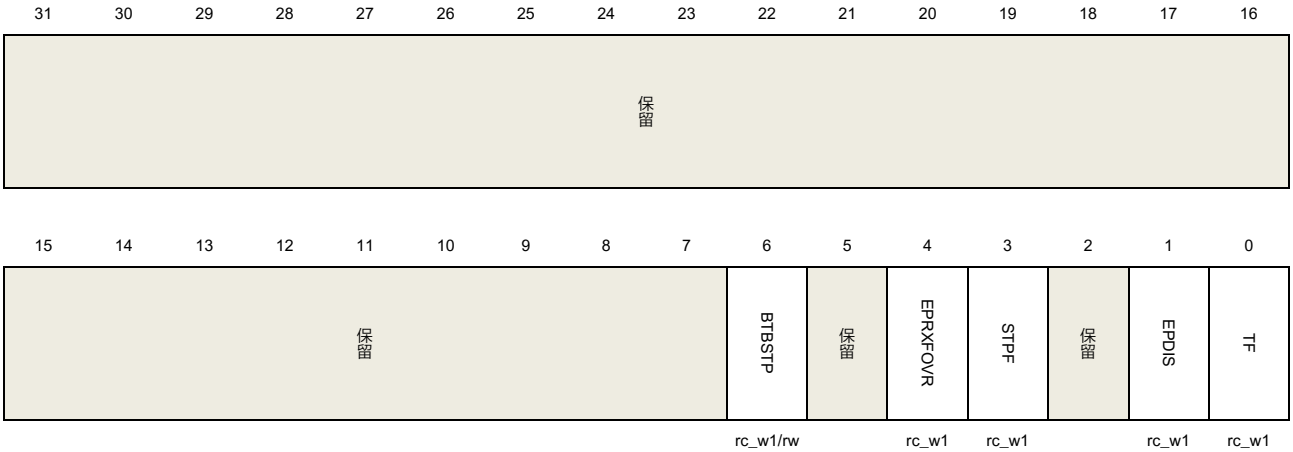

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:8 | 保留 | 必须保持复位值。 |
| 7 | TXFE | 发送FIFO空<br>端点的Tx FIFO达到寄存器USBFS_GAHBCS的位TXFTH定义的空阈值。 |
| 6 | IEPNE | IN端点NAK有效<br>寄存器USBFS_DIEPxCTL的位SNAK的设置生效，该位可以通过写1清零或设置CNAK位 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | EPTXFUD | 端点Tx FIFO下溢<br>如果当IN令牌被接收后，Tx FIFO没有包数据，该标志被触发。 |
| 3 | CITO | 控制IN事务超时中断<br>在控制IN事务中，如果设备等待的握手包超时，该标志位被触发 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | EPDIS | 端点除能<br>端点除能时，该标志位被触发 |
| 0 | TF | 传输完成<br>当该端点的所有IN事务完成，该标志位被触发。 |

### 设备 OUT 端点 x 中断标志寄存器(USBFS_DOEPxINTF)(x = 0..3， x 是设备端点编号)

地址偏移：0x0B08 + (设备端点编号 x 0x20)
复位值：0x0000 0000

该寄存器包含 OUT 端点的状态和事件，当获得一个 OUT 端点的中断时，应该读取该端点的中断标志寄存器，从而获知中断源。该寄存器的标志位通常硬件置位，各位写 1 清零。

该寄存器采用字(32位)访问

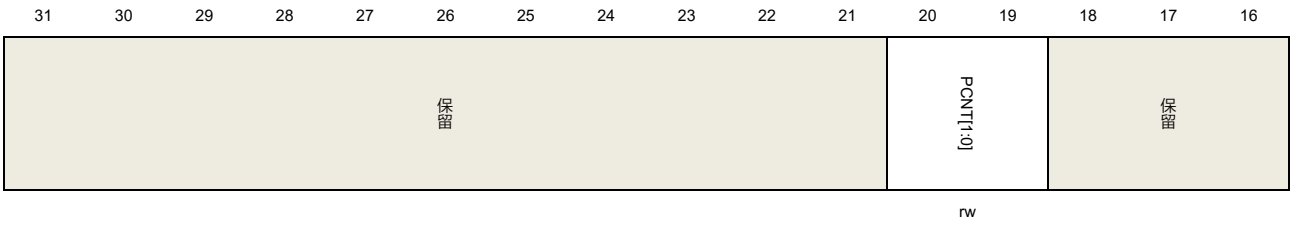


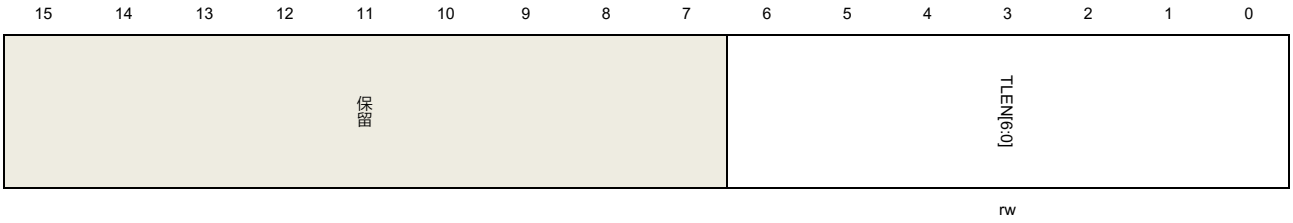

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:7 | 保留 | 必须保持复位值。 |
| 6 | BTBSTP | 连续SETUP包(适用于控制OUT端点)<br>当一个控制OUT端点接收超过连续3个SETUP包时，该标志被触发。 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | EPRXFOVR | 端点Rx FIFO上溢<br>当OUT令牌被接收时，如果OUT端点的Rx FIFO没有足够的空间存放数据包，该位被触发。在这种情况下，USBFS不能接收OUT数据包，发送一个NAK握手包。 |
| 3 | STPF | SETUP阶段完成(适用于控制OUT端点)<br>当一个SETUP阶段完成，也就是USBFS在一个setup令牌后接收了一个IN或OUT令牌，该位被置位。 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | EPDIS | 端点除能<br>端点除能时，该标志位被触发 |
| 0 | TF | 传输完成<br>当该端点的所有OUT事务完成，该标志位被触发 |

### 设备 IN 端点 0 传输长度寄存器(USBFS_DIEP0LEN)

地址偏移：0x0910
复位值：0x0000 0000

该寄存器采用字(32位)访问



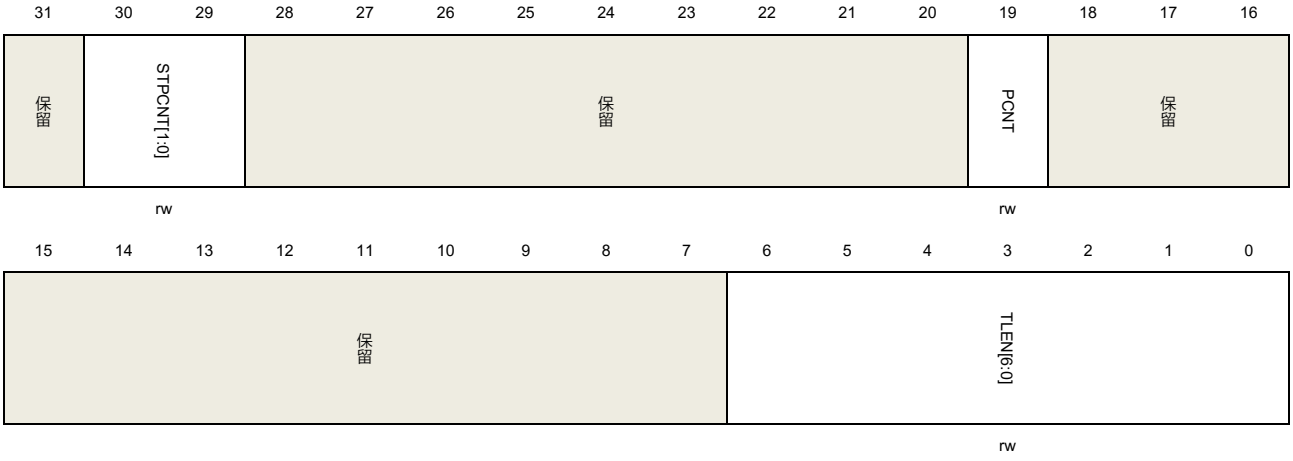



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:21 | 保留 | 必须保持复位值。 |
| 20:19 | PCNT[1:0] | 包数<br>传输中被发送的数据包数量<br>在端点使能之前，软件设置该位，在传输开始后，该域在每次数据包成功发送后自动减少。 |
| 18:7 | 保留 | 必须保持复位值。 |
| 6:0 | TLEN[6:0] | 传输长度<br>一次传输的数据总字节数<br>该域是IN传输中需要发送的包数据的总字节数，在端点使能之前，软件设置该位，在软件成功地将包数据写入端点的Tx FIFO中，该域减少与包数据大小相同的数值。 |

### 设备 OUT 端点 0 传输长度寄存器(USBFS_DOEP0LEN)

地址偏移：0x0B10
复位值：0x0000 0000

该寄存器采用字(32位)访问



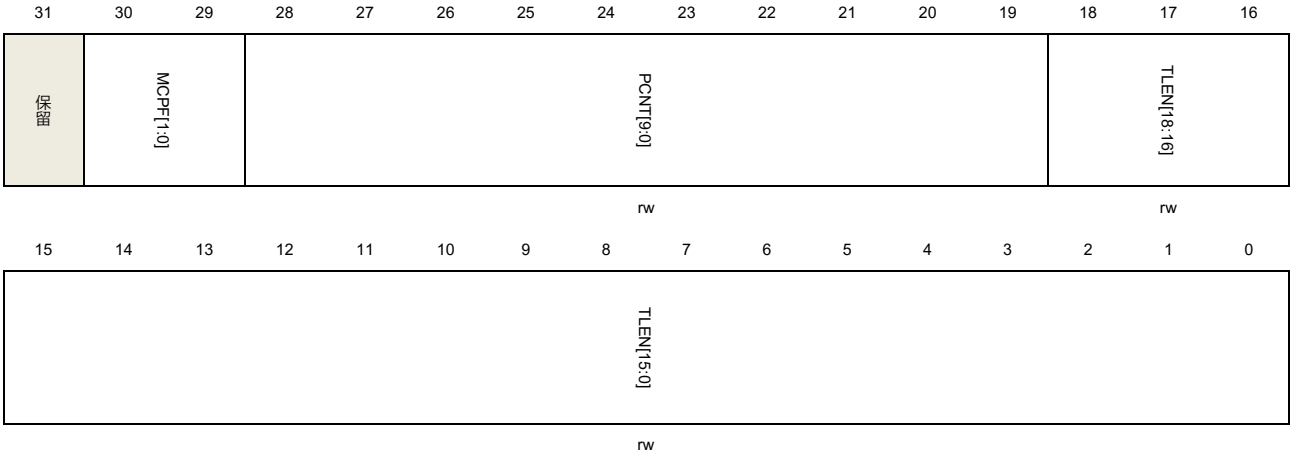

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30:29 | STPCNT[1:0] | SETUP包计数<br>该域定义端点可以接受的最大连续SETUP包数量<br>在SETUP传输之前，设置该域，每当连续SETUP包接收到时，该域值减1，当该域达到0时，寄存器USBFS_DOEP0INTF的BTBSTP标志被触发。 |

| | | |
|---|---|---|
| | | 00：0个包<br>01：1个包<br>10：2个包<br>11：3个包 |
| 28:20 | 保留 | 必须保持复位值。 |
| 19 | PCNT | 包计数<br>一次传输中应该接收到包数量。<br>在端点使能前，软件设置该位，在传输开始后，每当数据包接收到后，该域数值自动减少。 |
| 18:7 | 保留 | 必须保持复位值。 |
| 6:0 | TLEN[6:0] | 传输长度<br>传输中数据总字数。<br>该域是OUT传输中需要接收的包数据的总字节数，在端点使能之前，软件设置该位，在软件成功地将包数据读取端点的Rx FIFO中，该域减少与包数据大小相同的数值。 |

### 设备 IN 端点 x 传输长度寄存器(USBFS_DIEPxLEN)(x = 1..3， x 是设备端点编号)

地址偏移：0x910 + (设备端点编号 x 0x20)
复位值：0x0000 0000

该寄存器采用字(32位)访问



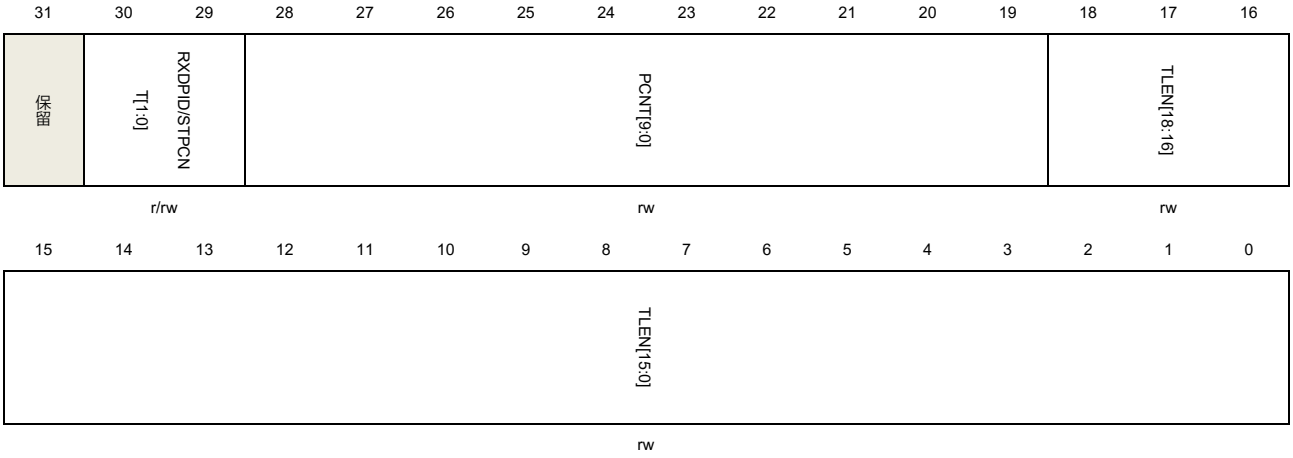

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30:29 | MCPF[1:0] | 每帧多包数目<br>该域是USB周期性IN端点每帧必须发的包数目。用于计算同步IN端点的数据PID。<br>01：1个包<br>10：2个包<br>11：3个包 |
| 28:19 | PCNT[9:0] | 包数量 |

| | | |
|---|---|---|
| | | 传输中被发送的数据包数量<br>在端点使能之前，软件设置该位，在传输开始后，该域在每次数据包成功发送后自动减少。 |
| 18:0 | TLEN[18:0] | 传输长度<br>传输的数据总字节数<br>该域是IN传输中需要发送的包数据的总字节数，在端点使能之前，软件设置该位，在软件成功地将包数据写入端点的Tx FIFO中，该域减少与包数据大小相同的数值。 |

### 设备 OUT 端点 x 传输长度寄存器(USBFS_DOEPxLEN)(x = 1..3， x 是设备端点编号)

地址偏移：0x0B10 + (设备端点编号 x 0x20)
复位值：0x0000 0000

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30:29 | RXDPID[1:0] | 接收数据PID(适用于同步OUT端点)<br>该域保存该端点该数据包所接受的最后一个数据包的PID<br>00：DATA0<br>10：DATA1<br>其他：保留 |
| | STPCNT[1:0] | SETUP包数(适用于控制OUT端点)<br>该位定义该端点可以接受连续SETUP最大包数<br>在SETUP传输之前，设置该域，每当连续SETUP包接收到时，该域值减1，当该域达到0时，寄存器USBFS_DOEP0INTF的BTBSTP标志被触发。<br>00：0个包 |

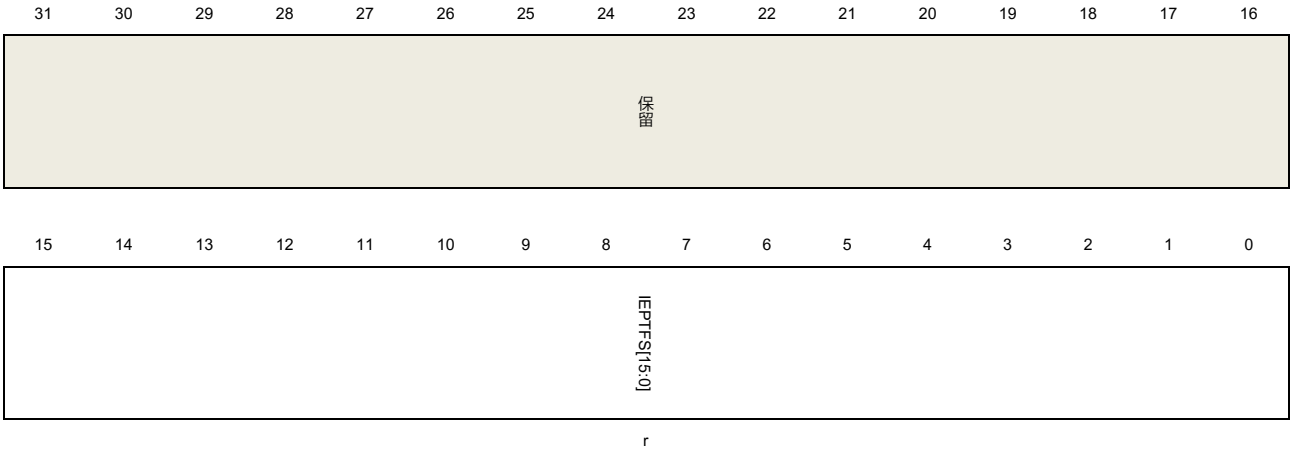
| | | |
|---|---|---|
| | | 01：1个包<br>10：2个包<br>11：3个包 |
| 28:19 | PCNT[9:0] | 包数<br>传输中应该接收到包数量<br>在端点使能前，软件设置该位，在传输开始后，每当数据包接收到后，该域数值自动减少。 |
| 18:0 | TLEN[18:0] | 传输长度<br>传输中数据总字数<br>该域是IN传输中需要接收的包数据的总字节数，在端点使能之前，软件设置该位，在软件成功地将包数据读取端点的Rx FIFO中，该域减少与包数据大小相同的数值。 |

### 设备 IN 端点 x 发送 FIFO 状态寄存器(USBFS_DIEPxTFSTAT)(x = 0..3， x 是设备端点编号)

地址偏移：0x0918 + (设备端点编号 x 0x20)
复位值：0x0000 0200

该寄存器包含每个端点的 Tx FIFO 的信息。

该寄存器采用字(32位)访问



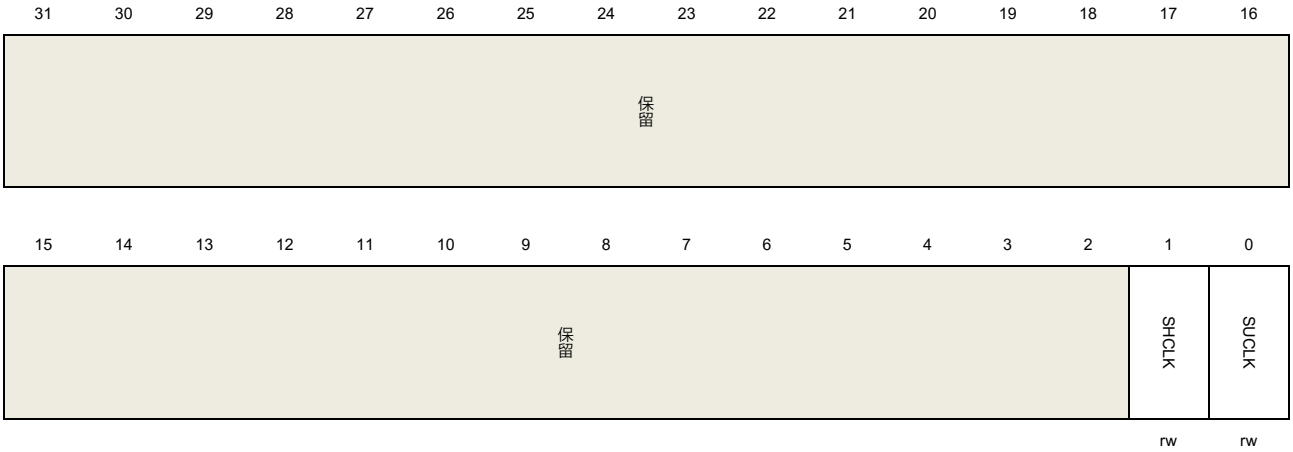

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | IEPTFS[15:0] | IN端点的Tx FIFO可用空间<br>IN端点的Tx FIFO可用空间用32位字为单位<br>0：FIFO是满的<br>1：1字可用<br>…<br>n：n字可用 |

### 28.6.4. 电源和时钟控制寄存器(USBFS_PWRCLKCTL))

地址偏移：0x0E00

复位值：0x0000 0000

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |
| 1 | SHCLK | 停止HCLK<br>停止HCLK，节省电量<br>0：HCLK未停止<br>1：HCLK停止 |
| 0 | SUCLK | 停止USB时钟<br>停止USB时钟，节省电量<br>0：USB时钟未停止<br>1：USB时钟停止 |

# 29. 通用串行总线高速接口(USBHS)

## 29.1. 概述

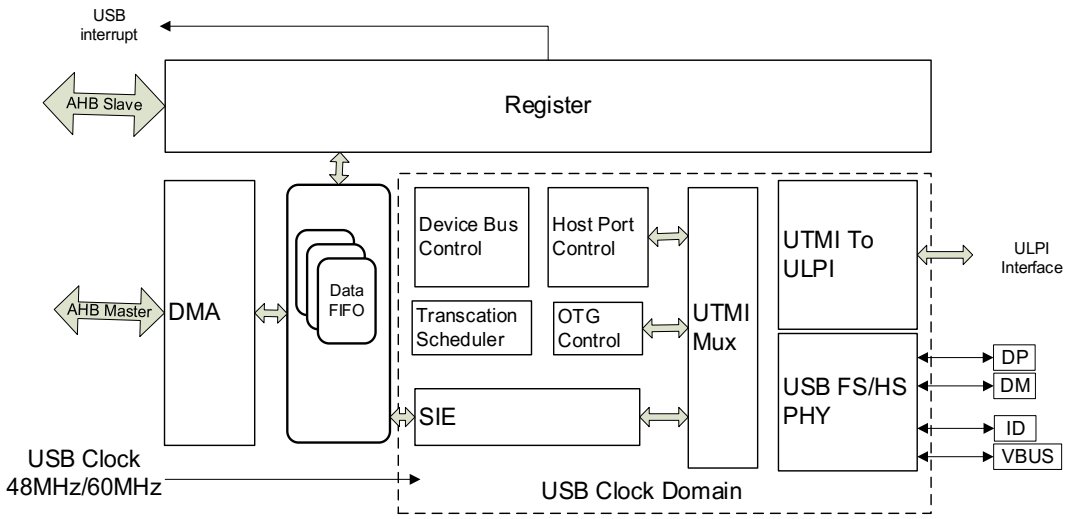
USB高速(USBHS)控制器为便携式设备提供了一套USB互联解决方案。USBHS不仅支持主机模式和设备模式，也支持遵循HNP(主机协商协议)和SRP(会话请求协议)的OTG模式。USBHS为外部USB物理层(PHY)提供了一个ULPI接口，并且其也包含了一个内部的全速USB PHY。所以，对于全速和低速操作，不再需要外部的USB PHY。USBHS可以支持USB 2.0协议所定义的所有四种传输方式(控制传输、批量传输、中断传输和同步传输)。当USBHS操作在高速主机模式下时，USBHS支持HUB连接。另外，在USBHS内部还有一个DMA引擎操作，可作为AHB总线主机在USBHS和系统之间加速数据传输。

## 29.2. 主要特性

- 支持USB 2.0高速(480Mb/s)/全速(12Mb/s)/低速(1.5Mb/s)主机模式；
- 支持USB 2.0高速(480Mb/s)/全速(12Mb/s)设备模式；
- 支持遵循HNP(主机协商协议)和SRP(会话请求协议)的OTG协议；
- 支持所有的4种传输方式：控制传输、批量传输、中断传输和同步传输；
- 支持高带宽中断和同步传输；
- 在主机模式下，包含USB事务调度器，用于有效地处理USB事务请求；
- 包含一个4KB的FIFO RAM；
- 在主机模式下，支持12个通道；
- 在主机模式下，包含2个发送FIFO(周期性发送FIFO和非周期性发送FIFO)和1个接收FIFO(由所有的通道共享)；
- 在设备模式下，包含6个发送FIFO(每个IN端点一个发送FIFO)和1个接收FIFO(由所有的OUT端点共享)；
- 在主机模式下，若在高速模式下操作，支持PING协议；
- 在设备模式下，支持6个OUT端点和6个IN端点；
- 在设备模式下，支持远程唤醒功能；
- 包含一个支持USB协议的全速USB PHY；
- 包含一个内部DMA调度器和引擎，每个应用请求都可在USBHS和系统之间执行数据拷贝；
- 在主机模式下，SOF的时间间隔可动态调节；
- 可将SOF脉冲输出到PAD；
- 可检测ID引脚电平和VBUS电压；
- 在主机模式或者OTG A设备模式下，需要外部部件为连接的USB设备提供电源。

## 29.3. 结构框图

**图 29-1. USBHS 结构框图**




## 29.4. 信号线描述

**表 29-1. USBHS 信号线描述**

| I/O 端口 | 类型 | 描述 | 注意 |
|---|---|---|---|
| VBUS | 输入 | 总线电源端口 | 仅内部 PHY 使用 |
| DM | 输入/输出 | 差分信号线 - 端口 | 仅内部 PHY 使用 |
| DP | 输入/输出 | 差分信号线 + 端口 | 仅内部 PHY 使用 |
| ID | 输入 | USB 识别：微连接器识别接口 | 仅内部 PHY 使用 |
| ULPI_D[7:0] | 输入/输出 | ULPI 数据线 | 外部 ULPI PHY 使用 |
| ULPI_NXT | 输入 | ULPI 下个信号线 | 外部 ULPI PHY 使用 |
| ULPI_DIR | 输入 | ULPI 方向 | 外部 ULPI PHY 使用 |
| ULPI_STP | 输出 | ULPI 停止 | 外部 ULPI PHY 使用 |
| ULPI_CLK | 输入 | ULPI 时钟 | 外部 ULPI PHY 使用 |

## 29.5. 功能描述

### 29.5.1. USBHS PHY 选择、时钟及工作模式

USBHS可以作为一个主机、一个设备或者一个DRD(双角色设备)，并且支持两种连接类型：内部全速PHY和外部ULPI PHY。根据用户需求，应用可以选择两种连接类型的任何一种。

若使用内部 PHY，USBHS 支持的最大速度为全速；若使用外部高速 ULPI PHY，USBHS 支持的最大速度为高速。应用也可以在主机模式下使用 USBHS_HCTL 寄存器内的 SPDFSLS 控

制位和在设备模式下使用 USBHS_DCFG 寄存器内的 DS[1:0]控制位将外部 ULPI PHY 的最大速度限制至全速。具体配置如*表 29-2. USBHS 支持速度列表*所示。

**表 29-2. USBHS 支持速度列表**

| 寄存器配置 | | 主机支持速度 | 设备支持速度 |
|---|---|---|---|
| EMBPHY=1<br>(内部PHY) | | 全速<br>低速 | 全速 |
| EMBPHY=0<br>(外部 ULPI PHY) | DS = 01 (设备模式)<br>SPDFSLS = 1(主机模式) | 全速<br>低速 | 全速 |
| | DS = 00(设备模式)<br>SPDFSLS = 0(主机模式) | 高速<br>全速<br>低速 | 高速<br>全速 |

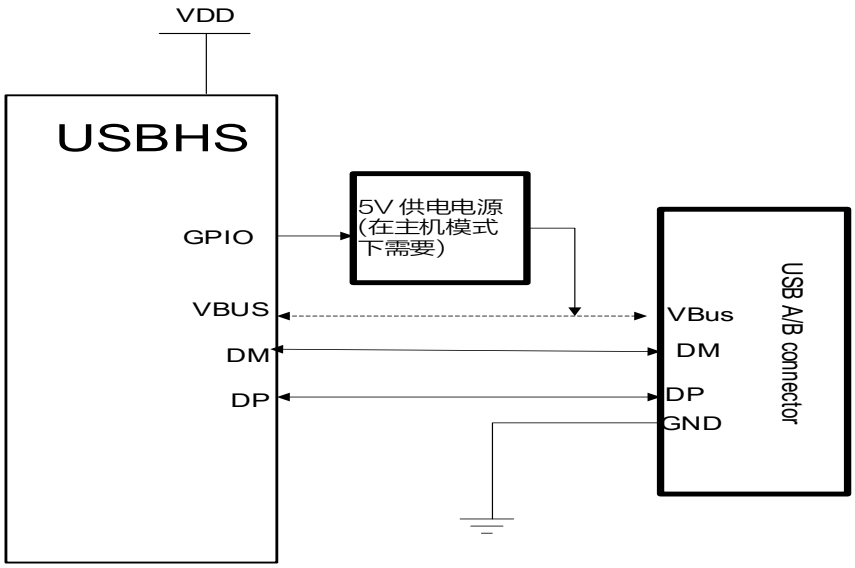
应用可以使用USBHS_GUSBCS寄存器中的FHM和FDM控制位选择USBHS的工作模式：主机模式(FHM=1)或设备模式(FDM=1)。当这两个控制位被清除时，USBHS工作在OTG模式，即系统复位后的默认模式。

**内部嵌入式全速PHY**

USBHS包含一个内部嵌入式全速PHY，该内部嵌入式PHY支持主机模式下的全速和低速、设备模式下全速以及具备HNP和SRP的OTG协议。软件需要置位USBHS_GUSBCS寄存器中的EMBPHY控制位来使用该内部嵌入式全速PHY。如果内部全速PHY被选择，USBHS所使用的USB时钟需要配置为48MHz。该48MHz USB时钟从系统内部时钟产生，并且其时钟源和分频器需要在RCU模块中配置。

上拉或下拉电阻已经集成在内部全速 PHY 的内部，并且 USBHS 可根据当前模式(主机、设备或 OTG 模式)和连接状态进行自动选择。一个利用内部全速 PHY 的典型连接示意图如*图 29-2. 在主机或设备模式下，利用内部 PHY 的连接示意图*所示。

**图 29-2. 在主机或设备模式下，利用内部 PHY 的连接示意图**



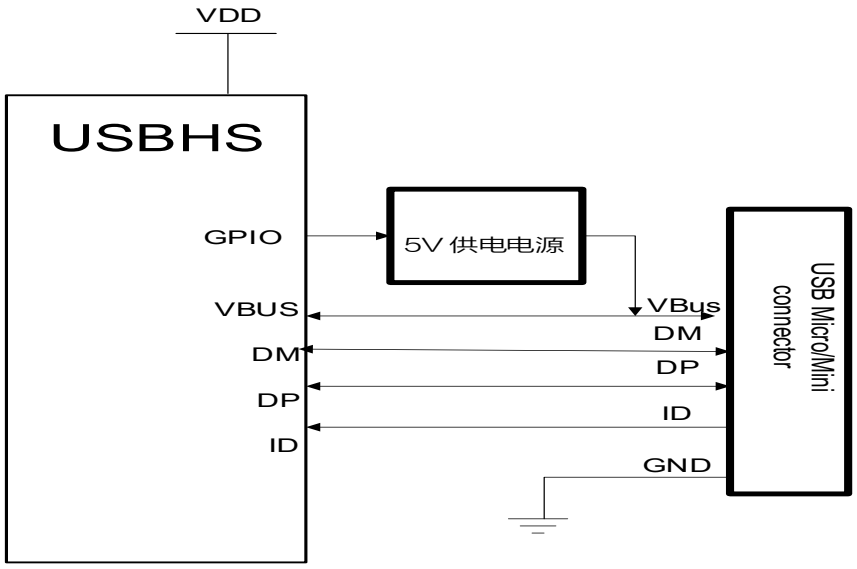

当USBHS工作在主机模式下时(FHM控制位置位、FDM控制位清除)，VBUS为USB协议所定义

的5V电源检测引脚。内部PHY不能提供5V VBUS电源，仅在VBUS信号线上具有电压比较器和充电放电电路。所以，如果应用需要提供VBUS电源，那么则需要一个外部的供电电源IC。在主机模式下，USBHS和USB连接头之间的VBUS连接可以被忽略，这是由于USBHS并不检测VBUS引脚的电平状态，并假定5V供电电源一直存在。

当USBHS工作在设备模式下时(FHM控制位清除、FDM控制位置位)，$V_{BUS}$检测电路由USBHS_GCCFG寄存器中的VBUSIG控制位配置。因此，如果设备不需要检测VBUS引脚电压，可以置位VBUSIG控制位，并可释放VBUS引脚作为其他用途。否则，VBUS引脚的连接不能够被忽略，并且USBHS需要不断的检测VBUS电平状态，一旦VBUS电压降至所需有效值以下，需要立即关闭DP信号线上的上拉电阻，从而产生一个断开状态。

OTG 模式连接示意图如*图 29-3. OTG 模式下使用内部嵌入式 PHY 连接示意图*所示。当 USBHS 工作在 OTG 模式下时，USBHS_GUSBCS 寄存器内的 FHM、FDM 控制位应该被清除。在这种模式下，USBHS 需要以下四个引脚：DM、DP、VBUS 和 ID，并且需要使用若干个电压比较器检测这些引脚的电压。USBHS 也包含 VBUS 充电和放电电路，用以完成 OTG 协议中所描述的 SRP 请求。OTG A 设备或 B 设备由 ID 引脚的电平状态所决定。在实现 HNP 协议的过程中，USBHS 控制上拉和下拉电阻。

**图 29-3. OTG 模式下使用内部嵌入式 PHY 连接示意图**

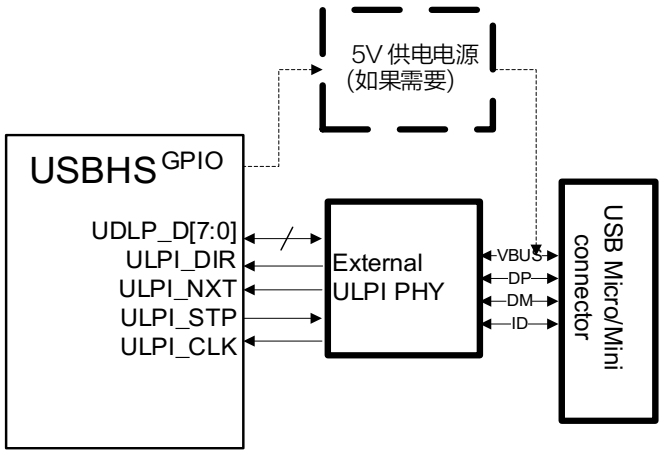



**外部ULPI PHY**

USBHS为外部PHY提供了一个ULPI接口。如果需要使用USBHS模块完成高速USB应用，那么则需要一个外部高速ULPI PHY。结合外部ULPI PHY，USBHS支持高速主机和设备，也支持前文中内部嵌入式全速PHY所描述的所有模式。

软件需要清除USBHS_GUSBCS寄存器中的EMBPHY控制位以使能ULPI接口。当ULPI模式能使，USB时钟需要配置到60MHz，并且需要从ULPI_CLK引脚引入。软件可以在RCU模块中打开或关闭该60MHz ULPI时钟。

**图 29-4. 使用外部 ULPI PHY 的连接示意图**

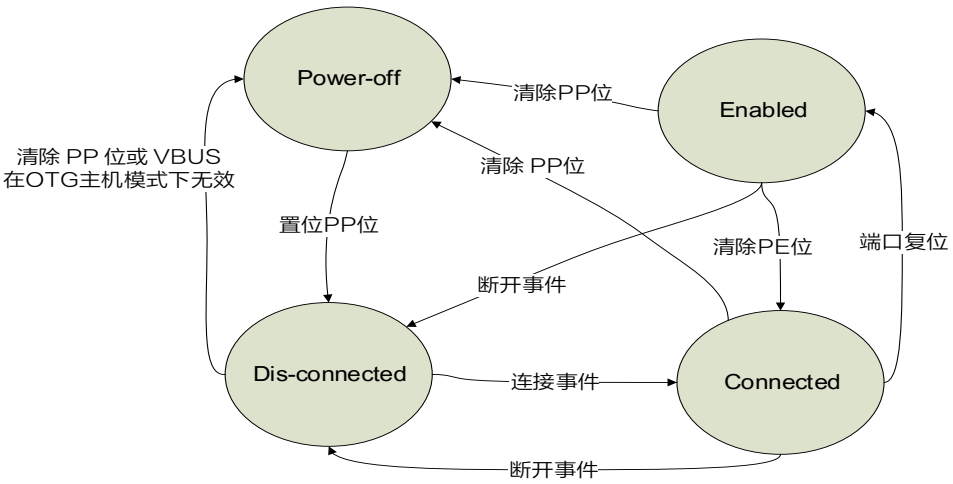



## 29.5.2. USB 主机功能

### USB主机端口状态

主机应用可以通过USBHS_HPCS寄存器控制USB端口状态。系统初始化之后，USB端口保持掉电状态。通过软件置位PP控制位后，USB PHY(内部或外部)将被上电，并且USB端口变为断开状态。检测到连接后，USB端口变为连接状态。在USB总线上产生一个复位后，USB端口将变为使能状态。

**图 29-5. 主机端口状态转移图**




### 连接、复位和速度识别

作为USB主机，在检测到一个连接事件后，USBHS会为应用触发一个连接标志；同样，若检测到一个断开事件后，将会触发一个断开标志。

PRST控制位用于实现USB复位序列。应用可以置位该控制位以启动一个USB复位序列，或者

清除该控制位以结束USB复位序列。仅当端口在连接或使能状态时，该控制位有效。

USBHS在对设备的连接和复位时执行速度检测，并且速度检测的结果会反馈在USBHS_HPCS寄存器的PS[1:0]位域中。

如果最大支持速度被配置为全速(SPDFSLS=1)，USBHS仅仅在设备连接的过程中执行速度识别，并且从DM或DP的电平状态决定设备速度。就像USB协议中所描述的那样，全速设备上拉DP信号线，而低速设备上拉DM信号线。

如果最大支持速度被配置为高速(SPDFSLS=0)，USBHS首先在连接的过程中执行速度检测，如果检测到全速设备连接，USBHS会在连接事件后的每个USB复位序列中，尝试执行高速检测(USB2.0协议中所描述的CHIRP序列)。所以，在主机上的应用应该在一个连接事件后提供一个USB复位，并且再次检查PS[1:0]标志位，以确定其连接的是否为高速设备。

**挂起和复位**

USBHS支持挂起和复位状态，当USBHS端口在使能状态时，向USBHS_HPCS寄存器的PSP控制位写1，USBHS会进入到挂起状态。在挂起状态，USBHS停止在USB总线上发送SOF，并且这样会让连接的USB设备在3ms后进入挂起状态。应用程序能够置位USBHS_HPCS寄存器中的PREM控制位以启动一个恢复序列，从而唤醒挂起的设备，当清除该控制位时，则可以停止恢复序列。如果主机在挂起状态下检测到一个远程唤醒信号，将会置位USBHS_GINTF寄存器的WKUPIF标志位，并且触发USBHS唤醒中断。

**SOF产生器**

在主机模式下，USBHS向USB总线发送SOF令牌包。如USB 2.0协议所描述，全速连接下，SOF令牌包每1ms产生一次(由主机控制器或者HUB事务转换器产生)；高速连接下，SOF令牌包将在接下来的七个125 μs周期后产生。

每当USBHS进入到使能状态后，它将会按照USB2.0所定义的周期发送SOF令牌包。然而，应用程序可以通过写USBHS_HFT寄存器中的FRI[15:0]位来调整一帧或者一微帧的间隔。FRI位定义了在一帧或一微帧中的USB时钟周期个数，并且应用程序应该基于USBHS所使用的USB时钟频率计算该值。FRT[14:0]位显示当前帧或微帧剩余的时钟周期个数，并且在挂起状态时，该值将停止改变。

USBHS能够在每个SOF令牌包中产生一个脉冲信号，并且将其输出至一个引脚。该脉冲信号长度为16个HCLK周期。如果应用程序希望使用该功能，需要置位USBHS_GCCFG寄存器的SOFOEN控制位，并且配置相应的引脚寄存器为GPIO功能。

**USB通道和事务**

USBHS在主机模式下包含12个独立的通道。每个通道能够与一个USB设备端点通信。通道的传输类型、方向、数据包长和其他信息都在通道相应的寄存器中配置，例如USBHS_HCHxCTL和USBHS_HCHxLEN寄存器。

USBHS支持所有的四种传输类型：控制、批量、中断和同步。USB 2.0协议将这些传输类型划分为两类：非周期性传输(控制和批量)和周期性传输(中断和同步)。基于此，为了有效地进行事务调度，USBHS包含两种请求队列：周期性请求队列和非周期性请求队列。在上述请求队列中的请求条目可能代表一个USB事务请求或者一个通道操作请求。

在无DMA模式下，如果应用想要在USB总线上启动一个OUT事务，应用需要通过AHB寄存器

接口向数据FIFO中写入数据包。USBHS硬件会在应用写完整包数据后，自动产生一个事务请求并进入请求队列。在DMA模式下，应用仅需要配置通道属性和通道数据缓冲区地址，USBHS内部的DMA引擎会执行数据包拷贝和请求条目的产生工作。当应用使能IN通道时，USBHS自动产生IN请求条目。

请求队列中的请求条目通过事务控制模块按顺序处理。USBHS通常首先尝试处理周期性请求队列，然后处理非周期性请求队列。

帧起始后，USBHS首先开始处理周期性队列，直到队列为空或者当前周期性请求队列所需时间不够，然后处理非周期性队列。这种做法保证了一帧或微帧中周期性传输的带宽。每次USBHS从请求队列中读取并取出一个请求条目。如果取出的是通道禁用请求，这将直接禁用通道并准备处理下个条目。

如果当前请求是一个事务请求并且USB总线时间能够处理这个请求，USBHS会使用SIE在USB总线上产生该事务。

在当前帧内，当前请求所需的总线时间不足时，如果当前请求为周期性请求，USBHS停止处理该周期性请求队列，并启动处理非周期性请求。如果当前请求为非周期性请求，USBHS会停止处理任何队列，并等待直到当前帧结束。

### 29.5.3. USB 设备功能

#### USB设备连接

在设备模式下，USBHS在初始化后保持掉电状态。利用VBUS引脚上的5V电源连接USB主机后或者置位USBHS_GCCFG寄存器中VBUSIG控制位，USBHS将进入供电状态。USBHS首先打开DP信号线上的上拉电阻，之后主机方将会检测到一个连接事件。

#### 复位和速度识别

USB主机在检测到设备连接之后，总是会启动一个USB复位序列，并且在设备模式下，检测到USB总线复位事件后，USBHS会为软件触发一个复位中断。

如果最大支持速度被配置为全速(USBHS_DCFG寄存器内DS[1:0] = 01)，USBHS会以全速设备操作，然而如果最大支持速度被配置为高速(USBHS_DCFG寄存器内DS[1:0] = 00)，在复位序列中，USBHS设备会尝试和主机启动一个速度识别(USB2.0协议中描述的一个CHIRP序列)。如果和主机的CHIRP序列握手成功，设备将会进入高速模式，否则，仍然停留在全速模式。

在复位序列和速度识别过程完成后，USBHS将会触发USBHS_GINTF寄存器中的ENUMF标志/中断，并且利用USBHS_DSTAT寄存器内的ES标志位反映当前枚举设备速度。所以，如果软件想要实现一个高速设备，必须等待ENUMF中断，然后读取ES[1:0]控制位以获得速度识别结果。

如USB2.0协议所需要，USBHS在外设模式下不支持低速。

#### 挂起和唤醒

USB总线保持IDLE状态并且数据线3ms无变化，USB设备将会进入挂起状态。当USB设备在挂起状态时，软件能够关闭大部分的时钟以节省电能。USB主机可以通过在USB总线上产生恢复信号，来唤醒挂起的设备。USBHS检测到恢复信号后，将置位USBHS_GINTF寄存器的

WKUPIF标志位并且触发USBHS唤醒中断。

在挂起设备模式，USBHS也能够远程唤醒USB总线。软件可以通过置位USBHS_DCTL寄存器的RWKUP控制位来发送一个远程唤醒信号，并且如果USB主机支持远程唤醒，主机会在USB总线上启动发送一个恢复信号。

**软件断开**

USBHS支持软件断开。设备进入到供电状态后，USBHS会打开DP信号线的上拉电阻，并且这样主机会检测到设备连接。然后，软件可以通过置位USBHS_DCTL寄存器中SD控制位进行强制断开。SD控制位置位后，如果当前设备速度为高速，USBHS会首先返回到全速设备，然后关闭DP信号线上的上拉电阻；如果当前设备速度为全速，USBHS将会直接关闭上拉电阻。这样，USB主机将会在USB总线上检测到设备断开。

**SOF跟踪**

当USBHS在USB总线上接收到一个SOF令牌包时，将触发一个SOF标志/中断，并且开始利用本地USB时钟计算总线时间。当前帧的帧号将会反应在USBHS_DSTAT寄存器的FNRSOF[13:0]位域。当USB总线时间达到EOF1或EOF2点(帧结束，在USB 2.0协议中描述)，USBHS会触发USBHS_GINTF寄存器中的EOPFIF标志/中断。软件能够使用这些标志位和寄存器以获得当前总线时间和位置信息。

### 29.5.4. OTG 功能概述

USBHS支持OTG协议1.3中所描述的OTG功能，OTG功能包括SRP和HNP。

**A设备和B设备**

当标准A或微型A插头插入相应的插座时，具有OTG能力的USB设备为A设备。A设备向VBUS供电，并且在会话开始时默认为主机。当标准B、微型B、迷你B插头插入相应的插座或采用一端为标准A插头的不可分离电缆时，具有OTG能力的USB设备为B设备。B设备在会话开始时默认为外设。USBHS使用ID引脚电平状态决定A设备或B设备。ID引脚状态反馈在USBHS_GOTGCS寄存器的IDPS状态位。为了了解A设备和B设备之间传输的详细状态，请参考OTG1.3协议。

**HNP**

主机协商协议(HNP)允许主机功能在两个直接连接的OTG设备之间转换，并且用户不需要为了设备之间通信控制的改变而切换电缆线的连接。典型地，HNP协议是由B设备上的用户或应用启动，HNP只能通过设备上的微型AB插座执行。

一旦OTG设备具有一个微型AB插座，该OTG设备可通过插入的插头类型决定默认为主机或设备(微型A插头插入为主机，微型B插头插入为设备)。通过使用主机协商协议(HNP)，一个默认为外设的OTG设备可以请求成为主机。主机角色切换的过程在下段中描述。此协议使用户不需要为了更改连接设备的角色而切换电缆线的连接。

当USBHS工作在OTG A主机模式时，并且其想放弃主机角色，可以首先置位USBHS_HPCS寄存器的PSP控制位来使USB总线进入挂起状态。然后B设备在3ms后进入挂起状态。如果B设备想要变为主机，软件需要置位USBHS_GOTGCS寄存器的HNPREQ控制位，然后USBHS会开

始在总线上执行HNP协议，最后，HNP的结果会反馈在USBHS_GOTGCS寄存器的HNPS状态位。另外，软件总能从USBHS_GINTF寄存器的COPM状态位获取当前设备角色(主机或外设)。

**SRP**

会话请求协议(SRP)允许B设备请求A设备打开VBUS并启动一个会话。该协议允许A设备(或许是电池供电)当总线无活动时通过关闭VBUS以节省电能，并为B设备启动总线活动提供了一种方法。如OTG协议中所描述，OTG设备必须和几个阈值比较VBUS电压，并且将比较结果反馈在USBHS_GOTGCS寄存器的ASV和BSV状态位中。

当USBHS工作在B设备OTG模式时，软件可以通过置位USBHS_GOTGCS寄存器的SRPREQ控制位来启动一个SRP请求，并且如果SRP请求成功，USBHS会在USBHS_GOTGCS寄存器中产生一个成功标志位SRPS。

当USBHS工作在OTG A设备模式且从B设备检测到一个SRP请求时，USBHS将会置位USBHS_GINTF寄存器中的SESIF标志位。软件获取该标志位后，需要准备为VBUS引脚打开5V供电电源。

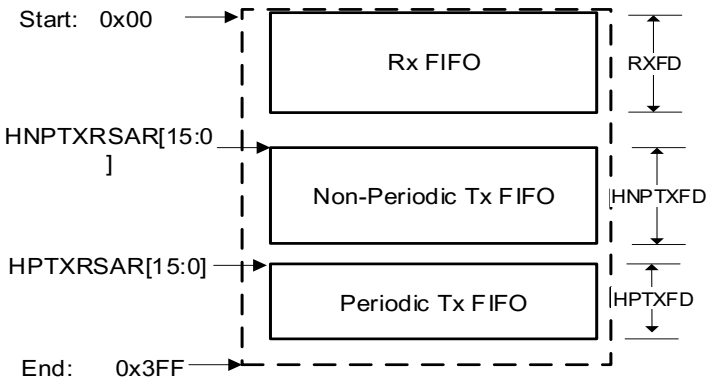
### 29.5.5. 数据 FIFO

USBHS中采用4K字节数据FIFO存储包数据，数据FIFO是通过USBHS的内部SRAM实现的。

**主机模式**

主机模式下，数据 FIFO 空间分为三个部分，分别是：用于接收数据包的 Rx FIFO、用于非周期性发送数据包的非周期性 Tx FIFO 和用于周期性发送数据包的周期性 Tx FIFO。所有的 IN 通道通过共享 Rx FIFO 接收数据。所有的周期性 OUT 通道通过共享周期性 Tx FIFO 来发送数据，所有的非周期性 OUT 通道通过共享非周期性 Tx FIFO 发送数据。通过寄存器 USBHS_GRFLEN、USBHS_HNPTFLEN 和 USBHS_HPTFLEN，软件可以配置以上数据 FIFO 的大小和起始偏移地址。*图 29-6. 主机模式 FIFO 空间*所描述的是 SRAM 中各 FIFO 的结构，图中的数值是以 32 位为单位。

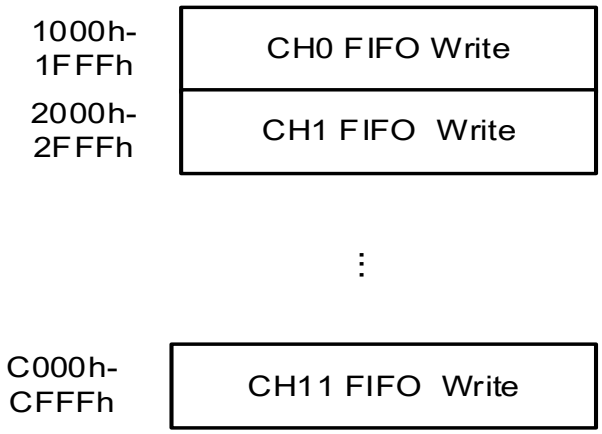
**图 29-6. 主机模式 FIFO 空间**




在 DMA 模式下，DMA 负责系统存储区和数据 FIFO 之间的数据包传输。在非 DMA 模式下，程序将包数据写入数据 FIFO 或从数据 FIFO 读取包数据。USBHS 为程序提供了专有寄存器

空间来读写数据 FIFO。*图 29-7. 主机模式 FIFO 访问寄存器映射表*所描述的是数据 FIFO 所访问的寄存器存储空间，图中的数值是以字节为单位寻址。尽管所有的非周期通道共享相同的 FIFO 以及所有的周期通道共享相同的 FIFO，每个通道都拥有它们的 FIFO 访问寄存器空间。对 USBHS 而言，获知当前压入数据包的通道号是非常重要的，通过寄存器 USBHS_GRSTATR/USBHS_GRSTATP 来访问数据包所从属的 Rx FIFO。

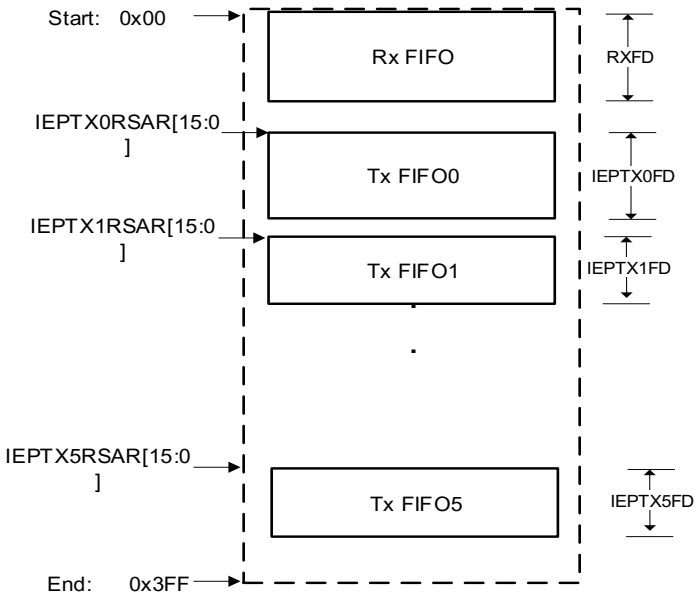
**图 29-7. 主机模式 FIFO 访问寄存器映射表**




**设备模式**

在设备模式下，数据 FIFO 分为多个部分，其中包含 1 个 Rx FIFO 和 6 个 Tx FIFO，每个 Tx FIFO 对应着一个 IN 端点，所有的 OUT 端点通过共享 Rx FIFO 接收数据包。通过寄存器 USBHS_GRFLEN 和 USBHS_DIEPxTFLEN (x=0…5)，程序可配置数据 FIFO 的大小和起始偏移地址。*图 29-8. 设备模式 FIFO 空间*所描述的是 SRAM 中各 FIFO 的结构，图中的数值是以按照 32 位写的。

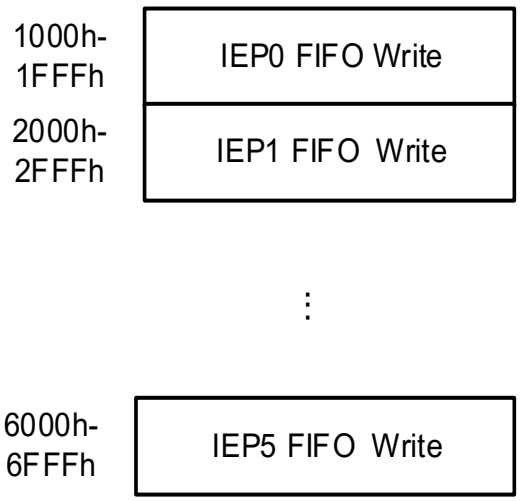
**图 29-8. 设备模式 FIFO 空间**




在 DMA 模式下，DMA 负责系统存储区和数据 FIFO 之间的数据包传输。在非 DMA 模式下，

程序将包数据写入数据 FIFO 或从数据 FIFO 读取包数据。USBHS 为程序提供了专有寄存器空间来读写数据 FIFO。*图 29-9. 设备模式 FIFO 访问寄存器映射表*所描述的是数据 FIFO 所访问的寄存器存储空间，图中的数值是以字节为单位寻址。每个端点都拥有它们的 FIFO 访问寄存器空间。通过寄存器 USBHS_GRSTATR/USBHS_GRSTATP 来访问 Rx FIFO。

**图 29-9. 设备模式 FIFO 访问寄存器映射表**




### 29.5.6. DMA 功能

该部分描述USBHS的DMA调度器和引擎。

**DMA请求和调度器**

DMA功能通过置位寄存器USBHS_GAHBCS的位DMAEN获得使能。当一个IN/OUT通道或IN端点被适当地配置和使能，或Rx FIFO非空，USBHS将生成DMA请求。USBHS的DMA调度器负责应答这些DMA请求。

当同时存在多个请求时，DMA调度器负责仲裁这些请求。这些请求分为三类：Rx FIFO DMA请求、周期性传输DMA请求和非周期性传输DMA请求。在仲裁中，Rx FIFO DMA请求是最高优先级，周期性传输DMA请求是中级优先级，非周期性传输DMA请求是最低优先级。在处理周期性和非周期性传输DMA请求中，DMA调度器实行循环仲裁方法。

综上所述，DMA将自动处理Rx FIFO非空事件，所以，在DMA模式下，程序中可以忽略寄存器USBHS_GINTF的RXFNEIF标志位。

**DMA引擎**

在主机或设备模式下，当Rx FIFO DMA请求获得仲裁后，DMA驱动器开始从Rx FIFO读取包数据或状态条目。对于包数据而言，DMA将数据写到特定的系统地址，该地址配置在寄存器USBHS_HCHxDMAADDR或USBHS_DIEPxDMAADDR/ USBHS_DIEPxDMAADDR。对于表目状态而言，在相关的通道或端点，DMA将生成特定的标志位或中断。

**主机传输**

当一个IN周期性或非周期性通道DMA请求获得仲裁后，DMA将IN请求条目写入周期性或非周

期性请求队列。当一个预期的IN传输完成，或一个AHB/USB总线错误发生后，DMA停止特定的通道，生成寄存器USBHS_HCHxINTF的TF和CH标志位。如上文所述，在Rx FIFO DMA请求生成后，在IN传输的过程中所接受的包数据被复制到系统存储区。

当一个OUT周期性或非周期性通道DMA请求获得仲裁后，DMA从系统存储区读取包数据，或将包数据写到内部的Tx FIFO。当每次完成包数据复制后，DMA总是将OUT请求条目写入请求队列。当一个预期的OUT传输完成，或一个AHB/USB总线错误发生后，DMA停止特定的通道，生成寄存器USBHS_HCHxINTF的TF和CH标志位。

**设备传输**

在设备模式下，当一个IN端点DMA请求获得仲裁后，DMA从系统存储区读取包数据，或将包数据写到端点的Tx FIFO。当USBHS获取IN端点的IN令牌后，将发送DMA引擎所复制的包数据。

### 29.5.7. 操作手册

该部分描述的是USBHS的操作手册。

#### 主机模式

**全局寄存器初始化顺序：**

1. 根据应用的需求，如是否使能DMA、DMA的传输类型、Tx FIFO的空阈值等，设置寄存器USBHS_GAHBCS，此时，GINTEN位需要保持清零状态。
2. 根据应用的需求，如操作模式(主机、设备或OTG)、某些OTG参数、ULPI和USB协议，设置寄存器USBHS_GUSBCS。
3. 根据应用的需求，设置寄存器USBHS_GCCFG。
4. 根据应用的需求，设置寄存器USBHS_GRFLEN、USBHS_HNPTFLEN_DIEP0TFLEN、USBHS_HPTFLEN，配置数据FIFO。
5. 通过设置寄存器USBHS_GINTEN使能模式错误和主机端口中断，置位USBHS_GAHBCS寄存器的GINTEN位使能全局中断。
6. 通过设置寄存器USBHS_HCTL的SPDFSLS位，判断是否将设备速度限制为全速。
7. 设置寄存器USBHS_HPCS，置位PP位。
8. 等待设备连接，当设备连接后，触发寄存器USBHS_HPCS的PCD位，然后置位PRST位，执行一次端口复位，等待至少10毫秒后，清除PRST位。
9. 等待USBHS_HPCS寄存器的PEDC中断，然后读取PE位以确认端口被成功地使能，读取PS位以获取连接的设备速度，之后，如果软件需要改变SOF间隔，设置USBHS_HFT寄存器。

**通道初始化和使能顺序：**

1. 根据期望的传输类型、方向、包大小等信息，设置寄存器USBHS_HCHxCTL，在设置期间，要保证位CEN和CDIS保持清除。
2. 设置寄存器USBHS_HCHxINTEN，设置期望的中断使能位。
3. 在DMA使能的前提下，设置寄存器USBHS_HCHxDMAADDR。
4. 设置寄存器USBHS_HCHxLEN，PCNT表示一次传输中的包数，TLEN表示一次传输中发送或接收的包数据的总字节数。

5. 对于OUT通道，如果PCNT为1，单包的大小等于TLEN。如果PCNT大于1，前PCNT-1个包被认定为最大包长度的包，其大小是由寄存器USBHS_HCHxCTL的位MPL所定义。最后一包的大小可通过PCNT、TLEN和MPL计算得到。如果程序想要发出一个零长度的包，应该设定TLEN为0，PCNT位1。
6. 对于IN通道，因为在IN事务结束之前，程序不知道实际接收的数据大小，程序可将TLEN设定为Rx FIFO所支持的最大值。
7. 置位寄存器USBHS_HCHxCTL中的CEN位以使能通道。

**通道除能顺序：**

程序可以通过同时置位CEN和CDIS除能通道。在寄存器操作后，USBHS将在请求队列中产生一个通道除能请求条目。当这个请求条目到达请求队列的顶部时，USBHS立即进行处理。

对于OUT通道而言，特定的通道将被立即除能。然后，会产生CH标志，USBHS将清除CEN和CDIS位。

对于IN通道而言，USBHS将通道除能状态条目压入Rx FIFO，然后，程序应该处理Rx FIFO非空事件：读和取出该状态条目，然后会产生CH标志，USBHS将清除CEN和CDIS位。

**IN传输操作顺序(DMA除能)：**

1. 初始化USBHS全局寄存器。
2. 初始化相应的通道。
3. 使能相应的通道。
4. 通过软件使能IN通道后，USBHS在相应请求队列中生成一个Rx请求条目。
5. 当Rx请求条目到达请求队列的顶部时，USBHS开始执行该请求条目。对于由请求条目所指示的事务而言，如果总线时间足够，USBHS在USB总线上开始IN事务。
6. 当IN事务结束时(收到ACK握手包)，USBHS将接收到的数据包压入Rx FIFO，ACK标志位被触发，否则，状态标志(NAK)会指示事务结果。
7. 如果步骤5所描述的IN事务完成后，步骤2的PCNT的数值比1大，程序将会返回步骤3，继续接收剩下的数据包。如果步骤5中描述的IN事务没有成功完成，程序将会返回步骤3来再次发送该数据包。
8. 在所有的传输中的所有事务都被成功接收后，USBHS将TF状态条目压入Rx FIFO的最后的数据包的顶部，这样，软件在读取所有接收的包数据后，再读取TF状态条目。USBHS生成TF标志来指示传输成功结束。
9. 除能通道，当通道处于空闲状态，即可为其他传输做准备。

**IN传输操作顺序(DMA使能)：**

1. 初始化USBHS全局寄存器。
2. 初始化并使能相应通道。
3. 在通过软件使能IN通道后，USBHS在相应请求队列中生成一个Rx请求条目。
4. USBHS逐一处理IN请求队列中的请求条目，并将它们所指示的IN事务发到USB总线上。
5. 当一个IN事务获得NAK握手包时，DMA可以自动地再发IN令牌直至USBHS获得预期的数据包的数目。
6. 在USBHS获取寄存器USBHS_HCHxTLEN的位PCNT中期望数据包数目后，USBHS生成TF和CH标志来表示传输成功完成，相应通道除能。如果在这些事务期间发生USB总线错误或DMA取值错误，DMA将触发相关的错误标志，停止该通道的操作，最后除能该通道，

触发CH标志。

**注意**：在DMA模式下，因为DMA将自动处理Rx FIFO，程序不再使能或处理RXFNEIF中断。

**OUT传输操作顺序(DMA除能)：**

1. 初始化USBHS全局寄存器。
2. 初始化及使能相应通道。
3. 将数据包写入通道的Tx FIFO(周期性Tx FIFO或非周期性Tx FIFO)。在所有的数据包都被写入FIFO后，USBHS在相应的请求队列中产生一个Tx请求条目，并且将USBHS_HCHxTLEN中的TLEN值减少，减少的数值等于已写的包大小。
4. 当请求条目到达请求队列的顶部时，USBHS开始执行该请求条目。如果请求条目对应的事务的总线时间足够，USBHS在USB总线上开展OUT事务。
5. 当由请求条目所指示的OUT事务结束时，寄存器USBHS_HCHnTLEN的位PCNT减1。如果该事务完成(收到ACK握手包)，ACK标志位被触发，否则，状态标志(NAK)会指示事务结果。
6. 如果步骤5所描述的OUT事务完成后且步骤2的PCNT的数值比1大，程序将会返回步骤3，继续发送剩下的数据包。如果步骤5中描述的OUT事务没有成功完成，程序将会返回步骤3来再次发送该包。
7. 在所有的传输中的所有事务都被成功送达后，USBHS生成TF标志来指示传输成功结束。
8. 除能通道，当通道处于空闲状态，即可为其他传输做准备。

**注意：**开始写发送FIFO之前，需要保证发送FIFO以及请求队列存在剩余空间。

**OUT传输操作顺序(DMA使能)：**

1. 初始化USBHS全局寄存器。
2. 初始化并使能相应通道。
3. USBHS的DMA开始从寄存器USBHS_HCHxDMAADDR的位DMAADDR中所指示的地址取包数据，并且将数据写入相应通道的Tx FIFO(周期性Tx FIFO或非周期性Tx FIFO)。每当一个完整的包数据被写入FIFO中，USBHS在相应的请求队列中生成一个Tx请求条目，并减少寄存器USBHS_HCHxTLEN的位TLEN的数值，所减少的数值与所完成写操作的包大小相同。
4. USBHS逐一处理请求队列中的请求条目，并将它们所指示的事务发到USB总线上。
5. 当一个事务获得NAK或PING握手包时，DMA可以再取或是再发数据包，在执行PING协议时也会自动像这样执行。
6. 如果所有的事务都被成功发送到USB总线上，USBHS生成TF和CH标志来表示传输成功完成，相应通道除能。如果在这些事务期间发生USB总线错误或DMA取值错误，DMA将触发相关的错误标志，停止该通道的操作，最后除能该通道，触发CH标志。

**注意**：在DMA模式下，因为DMA将自动处理Rx FIFO，程序不再使能或处理RXFNEIF中断。

### 设备模式

**全局寄存器初始化顺序：**

1. 根据应用的需求，如是否使能DMA、DMA的传输类型、Tx FIFO的空阈值等，设置寄存器USBHS_GAHBCS，此时，GINTEN位需要保持清零状态。
2. 根据应用的需求，如操作模式(主机、设备或OTG)、某些OTG参数、ULPI和USB协议，设置寄存器USBHS_GUSBCS。

3. 根据应用的需求，设置寄存器USBHS_GCCFG。
4. 根据应用的需求，设置寄存器USBHS_GRFLEN、USBHS_HNPTFLEN_DIEP0TFLEN、USBHS_HPTFLEN，配置数据FIFO。
5. 通过设置寄存器USBHS_GINTEN使能模式错误、挂起、SOF、枚举完成和USB复位中断，置位USBHS_GAHBCS寄存器的GINTEN位使能全局中断。
6. 根据应用的需求，如设备的地址和设备的速度等，设置寄存器USBHS_DCFG。
7. 在设备连接上主机上后，主机在USB总线上执行端口复位，触发寄存器USBHS_GINTF的RST中断。
8. 等待寄存器USBHS_GINTF的ENUMF中断。

**端点初始化和使能顺序：**

1. 根据预期的传输类型、包大小等信息，设置寄存器 USBHS_DIEPnCTL 或 USBHS_DOEPxCTL。
2. 设定寄存器 USBHS_DIEPINTEN 或 USBHS_DOEPINTEN，置位相应中断使能位。
3. 如果DMA使能，设定寄存器USBHS_DIEPxDMAADDR或USBHS_DOEPxDMAADDR。
4. 设定寄存器 USBHS_DIEPxLEN 或 USBHS_DOEPxLEN，PCNT 表示一次传输中的包数，TLEN 表示一次传输中发送或接收的包数据的总字节数。
   对于IN端点，如果PCNT等于1，单数据包的大小等于TLEN。如果PCNT大于1，前PCNT-1个包被认定为最大包长度的包，其大小是由寄存器USBHS_DIEPxCTL的位MPL所定义。最后一包的大小可通过PCNT、TLEN和MPL计算得到。如果程序想要发出一个零长度的包，应该设定TLEN为0，PCNT位1。
   对于OUT端点，因为在IN事务结束之前，程序不知道实际接收的数据大小，程序可将TLEN设定为Rx FIFO所支持的最大值。
5. 置位 USBHS_DIEPxCTL 或 USBHS_DOEPxCTL 寄存器 EPEN 位使能端点。

**端点除能顺序：**

当USBHS_DIEPnCTL或USBHS_DOEPnCTL寄存器的EPEN位被清除时，程序可以在任何时候除能端点。

**IN传输操作顺序(DMA除能)：**

1. 初始化USBHS全局寄存器。
2. 初始化和使能IN端点。
3. 将数据包写入端点的Tx FIFO，每当数据包写入FIFO，USBHS减少USBHS_DIEPxLEN寄存器的TLEN域的数值，其减少的数值等于已写的数据包大小。
4. 当IN令牌接收后，USBHS发送数据包，在USB总线上的事务完成后，USBHS_DIEPxLEN寄存器的PCNT值减1。如果事务成功完成(接收到ACK握手包)，ACK标志被触发，或者，其他状态标志表示事务的结果。
5. 在一次传输的所有数据包都被成功发送，USBHS生成一个TF标志位以表明传输成功结束，除能相应IN端点。

**IN传输操作顺序(DMA使能)：**

1. 初始化USBHS全局寄存器。
2. 初始化并使能相应端点。
3. 将数据包写入端点的Tx FIFO，每当包数据写入FIFO，USBHS减少USBHS_DIEPxLEN寄

存器的TLEN域的数值，其减少的数值等于已写的包数据大小。

4. 当IN令牌接收后，USBHS发送数据包，在USB总线上的事务完成后，USBHS_DIEPxLEN寄存器的PCNT值减1。如果事务成功完成(接收到ACK握手包)，ACK标志被触发，或者，其他状态标志表示事务的结果。
5. 在一次传输的所有数据包都被成功发送，USBHS生成一个TF和EPDIS标志位表明传输成功结束，除能相应IN端点。如果在事务期间出现USB总线错误或DMA取值错误，DMA将触发相关错误标志。

**注意**：在DMA模式下，因为DMA将自动处理Rx FIFO，程序不再使能或处理RXFNEIF中断。

**OUT传输操作顺序(DMA除能)：**

1. 初始化USBHS全局寄存器。
2. 初始化和使能OUT端点。
3. 当OUT令牌接收后，USBHS接收包数据或基于Rx FIFO状态和寄存器配置回复NAK握手包。如果事务成功完成(USBHS接收并保存数据到Rx FIFO，发送ACK握手包)，USBHS_DOEPxLEN寄存器的PCNT值减1。如果事务成功完成(接收到ACK握手包)，ACK标志被触发，或者，其他状态标志表示事务的结果。
4. 在一次传输的所有数据包都被成功接收，USBHS将TF状态条目压入Rx FIFO的最后的数据包的顶部，这样，软件在读取所有接收的包数据后，再读取TF状态条目。USBHS生成TF标志来指示传输成功结束。USBHS生成一个TF标志位表明传输成功结束，除能相应OUT端点。

**OUT传输操作顺序(DMA使能)：**

1. 初始化USBHS全局寄存器。
2. 初始化并使能相应OUT端点。
3. 当OUT令牌接收后，USBHS接收包数据或基于Rx FIFO状态和寄存器配置回复NAK握手包。如果事务成功完成(USBHS接收并保存数据到Rx FIFO，发送ACK握手包)，USBHS_DOEPxLEN寄存器的PCNT值减1。如果事务成功完成(接收到ACK握手包)，ACK标志被触发，或者，其他状态标志表示事务的结果。
4. 在一次传输的所有数据包都被成功发送，USBHS生成一个TF和EPDIS标志位表明传输成功结束，除能相应端点。如果在事务期间出现USB总线错误或DMA取值错误，DMA将触发相关错误标志。

**注意**：在DMA模式下，因为DMA将自动处理Rx FIFO，程序不再使能或处理RXFNEIF中断。

## 29.6. 中断

USBHS 有四种中断：全局中断、唤醒中断、端点 1IN 中断和端点 1OUT 中断。

全局中断是软件需要处理的主要中断，全局中断的标志位可在 USBHS_GINTF 寄存器读取，列举在*表 29-3. USBHS 全局中断*。

**表 29-3. USBHS 全局中断**

| 中断标志 | 描述 | 运行模式 |
|---|---|---|
| SESIF | 会话中断 | 主机或设备模式 |

| 中断标志 | 描述 | 运行模式 |
|---|---|---|
| DISCIF | 断开连接中断标志 | 主机模式 |
| IDPSC | ID 引脚状态变化 | 主机或设备模式 |
| PTXFEIF | 周期性 Tx FIFO 空中断标志 | 主机模式 |
| HCIF | 主机通道中断标志 | 主机模式 |
| HPIF | 主机端口中断 | 主机模式 |
| ISOONCIF/PXNCIF | 周期性传输未完成中断标志 /<br>同步OUT传输未完成中断标志 | 主机或设备模式 |
| ISOINCIF | 同步 IN 传输未完成中断标志 | 设备模式 |
| OEPIF | OUT 端点中断标志 | 设备模式 |
| IEPIF | IN 端点中断标志 | 设备模式 |
| EOPFIF | 周期性帧尾中断标志 | 设备模式 |
| ISOOPDIF | 同步 OUT 丢包中断标志 | 设备模式 |
| ENUMF | 枚举完成 | 设备模式 |
| RST | USB 复位 | 设备模式 |
| SP | USB挂起 | 设备模式 |
| ESP | 早挂起 | 设备模式 |
| GONAK | 全局OUT NAK有效 | 设备模式 |
| GNPINAK | 全局非周期IN NAK有效 | 设备模式 |
| NPTXFEIF | 非周期Tx FIFO空中断标志 | 主机模式 |
| RXFNEIF | Rx FIFO非空中断标志 | 主机或设备模式 |
| SOF | 帧首 | 主机或设备模式 |
| OTGIF | OTG 中断标志 | 主机或设备模式 |
| MFIF | 模式错误中断标志 | 主机或设备模式 |

唤醒中断可以在 USBHS 处于挂起状态时触发，即使 USBHS 的时钟停止。寄存器 USBHS_GINTF 的位 WKUPIF 是唤醒源。

端点 1 IN/OUT 中断是适用于端点 1 的两个特殊中断，程序可通过这两个中断快速回应端点 1 的事件。这两个是中断通过寄存器 USBHS_DEP1INT 各自使能，这两个中断源来自于寄存器 USBHS_DIEP1INTF 和 USBHS_DOEP1INTF，其中断使能位定义在寄存器 USBHS_DIEP1INTEN 和 USBHS_DOEP1INTEN。

## 29.7. USBHS 寄存器

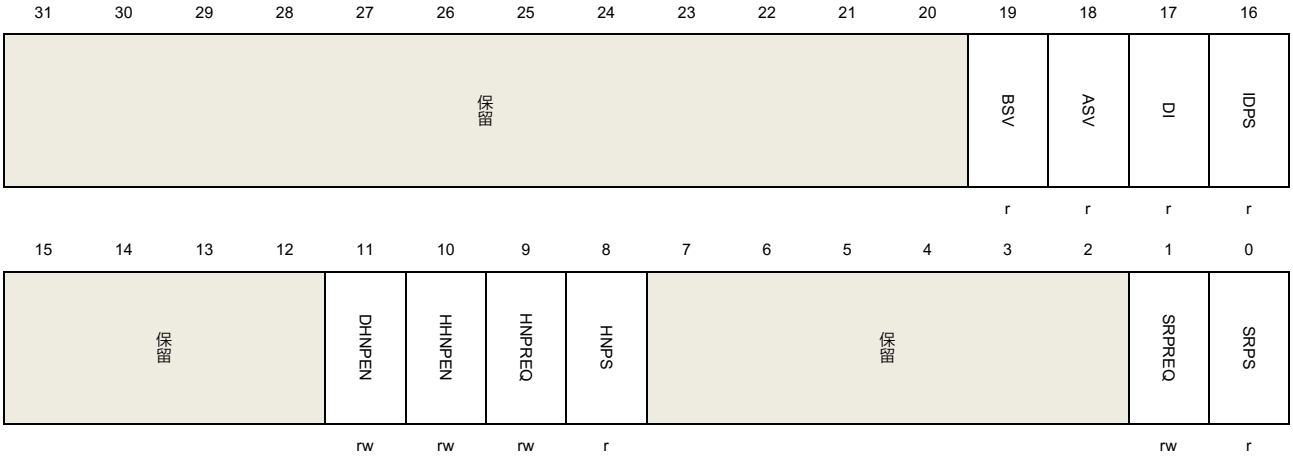
USBHS 基地址：0x4004 0000

### 29.7.1. 全局控制与状态寄存器组

#### 全局 OTG 控制和状态寄存器(USBHS_GOTGCS)

地址偏移：0x0000
复位值：0x0000 0800

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:20 | 保留 | 必须保持复位值。 |
| 19 | BSV | B会话有效(在OTG协议中描述)<br>0：OTG B设备Vbus电压水平低于VBSESSVLD<br>1：OTG B设备Vbus电压水平不低于VBSESSVLD<br>**注意**：仅在OTB B设备模式下可访问 |
| 18 | ASV | A会话有效<br>A主机模式收发器状态<br>0：OTG A设备Vbus电压水平低于VASESSVLD<br>1：OTG A设备Vbus电压水平不低于VASESSVLD<br>在会话的开始，A设备默认是主机。<br>**注意**：仅在OTG A设备模式下可访问 |
| 17 | DI | 去抖动间隔<br>检测到连接的去抖动间隔。<br>0：当USB总线上发生插入和连接时，指示长去抖动间隔<br>1：当HNP协议中使用一个软连接时，指示短去抖动间隔 |

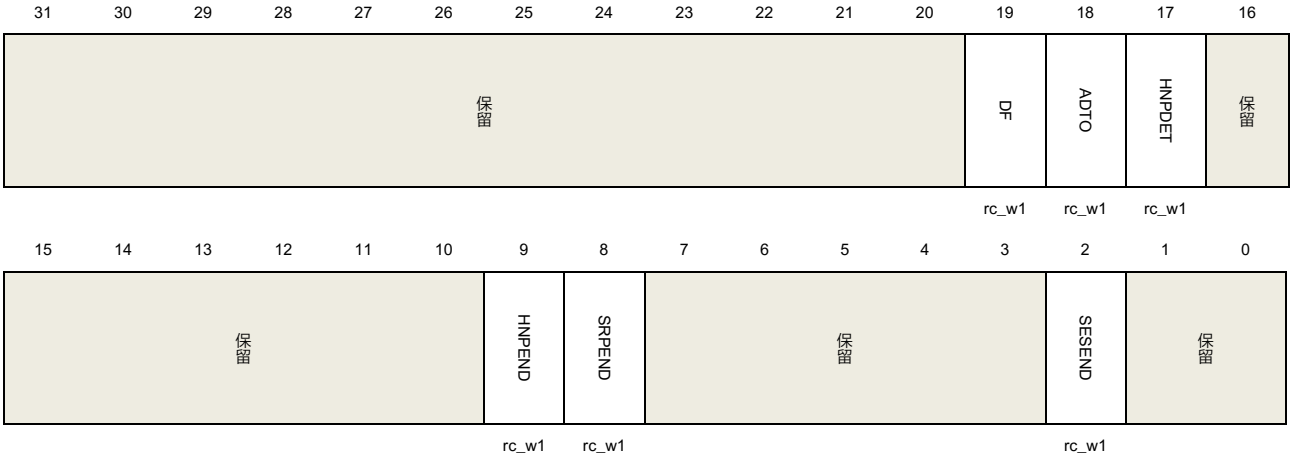
| | | |
|---|---|---|
| | | **注意**：仅在主机模式下可访问 |
| 16 | IDPS | ID引脚状态<br>连接器ID引脚的电压水平<br>0：USBHS工作在A设备模式<br>1：USBHS工作在B设备模式<br>**注意**：在设备和主机模式下均可访问 |
| 15:12 | 保留 | 必须保持复位值。 |
| 11 | DHNPEN | 设备HNP使能<br>使能B设备HNP功能。如果该控制位清除，当应用置位USBHS_GOTGCS寄存器中的HNPREQ控制位时，USBHS并不启动HNP协议。<br>0：HNP功能不使能<br>1：HNP功能使能<br>**注意**：仅在设备模式下访问 |
| 10 | HHNPEN | 主机HNP使能<br>使能A设备HNP功能。如果该控制位清除，USBHS不能够响应B设备的HNP请求。<br>0：HNP功能不使能<br>1：HNP功能使能<br>**注意**：仅在主机模式下访问 |
| 9 | HNPREQ | HNP请求<br>软件通过置位该控制位在USB总线上启动一个HNP。当USBHS_GOTGINTF寄存器中HNPEND控制位置位时，软件可以通过向该控制位写0或者清除USBHS_GOTGINTF寄存器中的HNPEND控制位来清除该控制位。<br>0：不发送HNP请求<br>1：发送HNP请求<br>**注意**：仅在设备模式下访问 |
| 8 | HNPS | HNP成功标志位<br>当HNP成功时，该标志位由内核置位。当HNPREQ置位时，该控制位被清除。<br>0：HNP失败<br>1：HNP成功<br>**注意**：仅在设备模式下访问 |
| 7:2 | 保留 | 必须保持复位值。 |
| 1 | SRPREQ | SRP请求<br>软件通过置位该控制位在USB总线上启动一个SRP会话请求。当USBHS_GOTGINTF寄存器中的SRPEND控制位置位时，软件可以通过向该控制位写0或者清除USBHS_GOTGINTF寄存器中的SRPEND控制位来清除该控制位。<br>0：没有会话请求<br>1：会话请求<br>**注意**：仅在设备模式下访问 |
| 0 | SRPS | SRP会话请求成功<br>当SRP会话请求成功时，该标志位由内核置位。当SRPREQ控制位被置位时，该标志 |

位被清除。
0：SRP会话请求失败
1：SRP会话请求成功
**注意**：仅在设备模式下访问

### 全局 OTG 中断状态寄存器(USBHS_GOTGINTF)

地址偏移：0x0004
复位值：0x0000 0000

该寄存器只能按字(32位)访问



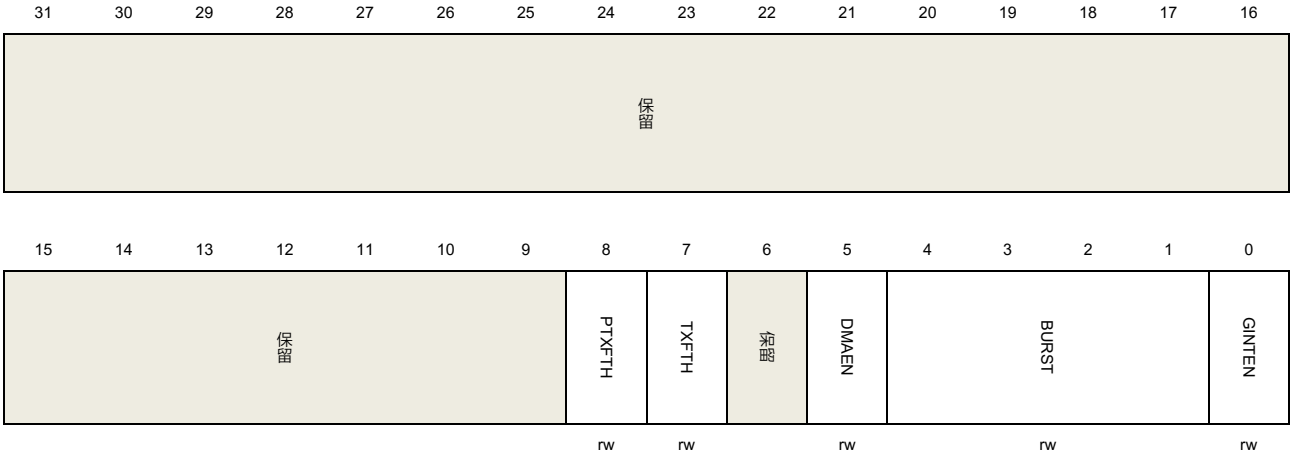

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:20 | 保留 | 必须保持复位值。 |
| 19 | DF | 去抖动完成<br>当设备连接去抖动完成时，USBHS置位该控制位<br>**注意**：仅在主机模式下可访问 |
| 18 | ADTO | A设备超时<br>当A设备等待B设备连接发生超时时，USBHS置位该控制位<br>**注意**：在设备和主机模式下，均可访问 |
| 17 | HNPDET | 检测到主机协商请求<br>当A设备检测到一个HNP请求时，USBHS置位该标志位<br>**注意**：在设备和主机模式下，均可访问 |
| 16:10 | 保留 | 必须保持复位值。 |
| 9 | HNPEND | HNP结束<br>当一个HNP结束时，内核置位该标志位。软件应该读取USBHS_GOTGCS寄存器中HNPS标志位，以获取HNP结果。<br>**注意**：在设备和主机模式下，均可访问。 |
| 8 | SRPEND | SRPEND<br>当一个SRP结束时，内核置位该标志位。软件应该读取USBHS_GOTGCS寄存器中 |

| | | |
|---|---|---|
| | | SRPS标志位，以获取SRP结果。<br>**注意**：在设备和主机模式下，均可访问。 |
| 7:3 | 保留 | 必须保持复位值。 |
| 2 | SESEND | 会话结束<br>当VBUS电压低于Vb_ses_vld时，内核置位该标志位。 |
| 1:0 | 保留 | 必须保留复位值。 |

### 全局 AHB 控制和状态寄存器(USBHS_GAHBCS)

地址偏移：0x0008
复位值：0x0000 0000

该寄存器只能按字(32位)访问



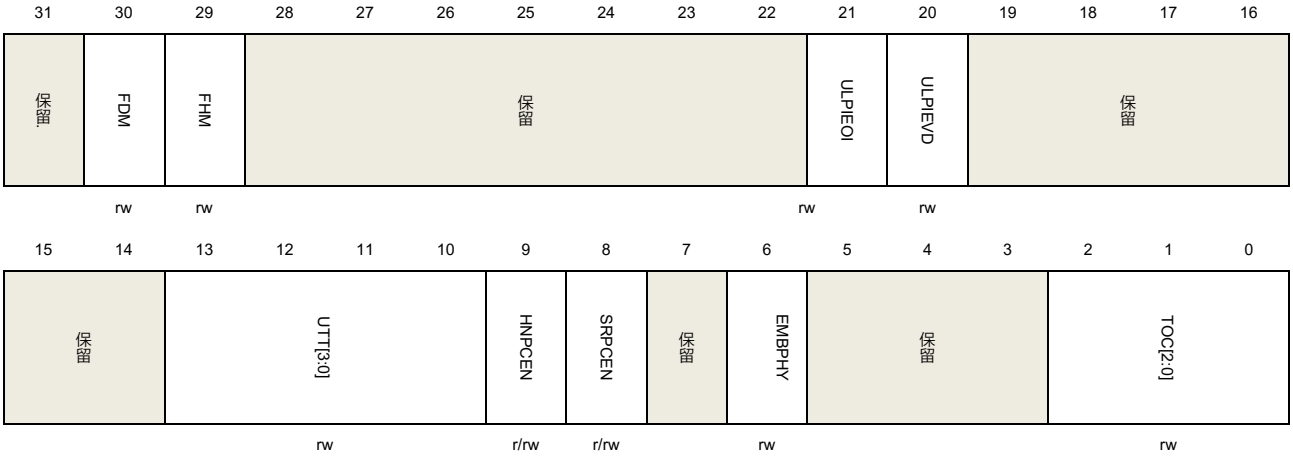

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:9 | 保留 | 必须保持复位值。 |
| 8 | PTXFTH | 周期性Tx FIFO阈值<br>0：当周期性发送FIFO半空时，将触发PTXFEIF标志位<br>1：当周期性发送FIFO全空时，将触发PTXFEIF标志位<br>**注意**：只在主机模式下访问 |
| 7 | TXFTH | Tx FIFO阈值<br>设备模式：<br>0：当IN端点发送FIFO半空时，将触发TXFEIF标志位<br>1：当IN端点发送FIFO全空时，将触发TXFEIF标志位<br>主机模式：<br>0：当非周期性发送FIFO半空时，将触发NPTXFEIF标志位<br>1：当非周期性发送FIFO全空时，将触发NPTXFEIF标志位 |
| 6 | 保留 | 必须保持复位值。 |
| 5 | DMAEN | DMA功能使能 |

| | | |
|---|---|---|
| | | 0：DMA功能禁用<br>1：DMA功能使能 |
| 4:1 | BURST[3:0] | DMA使用的AHB突发类型<br>0000：单次<br>0001：INCR<br>0011：INCR4<br>0101：INCR8<br>0111：INCR16 |
| 0 | GINTEN | 全局中断使能<br>0：全局中断不使能<br>1：全局中断使能<br>**注意**：在主机和设备模式下，均可访问 |

### 全局 USB 控制和状态寄存器(USBHS_GUSBCS)

地址偏移：0x000C
复位值：0x0000 0A00

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30 | FDM | 强制设备模式<br>通过置位该控制位，可强制USB内核为设备模式，并且忽略USBHS ID引脚的输入状态<br>0：正常模式<br>1：设备模式<br>设置该控制位后，应用必须等待至少25ms，让变化产生作用。<br>**注意**：在设备和主机模式下，均可访问。 |
| 29 | FHM | 强制主机模式<br>通过置位该控制位，可强制USB内核为主机模式，并且忽略USBHS ID引脚的输入状 |

| | | |
|---|---|---|
| | | 态<br>0：正常模式<br>1：主机模式<br>设置该控制位后，应用必须等待至少25ms，让变化产生作用。<br>**注意**：在设备和主机模式下，均可访问 |
| 28:22 | 保留 | 必须保持复位值。 |
| 21 | ULPIEOI | ULPI外部过流指示器<br>ULPI PHY使用该控制位决定使用内部或者外部过流指示器。该控制位只在外部ULPI PHY被使用时(本寄存器中EMBPHY控制位为0)，才有效。<br>0：ULPI PHY使用内部过流指示器<br>1：ULPI PHY使用外部过流指示器 |
| 20 | ULPIEVD | ULPI外部VBUS驱动器<br>ULPI PHY使用该控制位决定是由ULPI PHY还是外部电源驱动。该控制位仅在外部ULPI PHY被使用时(本寄存器中EMBPHY控制位为0)，才有效。<br>0：VBUS由ULPI PHY驱动<br>1：VBUS由外部电源驱动 |
| 19:14 | 保留 | 必须保持复位值。 |
| 13:10 | UTT[3:0] | USB运转时间<br>以物理时钟数来设定运转时间<br>**注意**：仅在设备模式下访问 |
| 9 | HNPCEN | HNP能力使能<br>控制HNP能力是否使能<br>0：HNP能力禁用<br>1：HNP能力使能<br>**注意**：在设备和主机模式下，均可访问 |
| 8 | SRPCEN | SRP能力使能<br>控制SRP能力是否使能<br>0：SRP能力禁用<br>1：SRP能力使能<br>**注意**：在设备和主机模式下，均可访问 |
| 7 | 保留 | 必须保持复位值。 |
| 6 | EMBPHY | 嵌入式PHY选择<br>控制选择使用内部嵌入式全速PHY还是外部ULPI PHY<br>0：USBHS使用外部ULPI PHY<br>1：USBHS使用内部嵌入式全速PHY<br>**注意**：在设备和主机模式下，均可访问 |
| 5:3 | 保留 | 必须保持复位值。 |
| 2:0 | TOC[2:0] | 超时校准<br>当等待一个包时，USBHS需要使用USB2.0协议中需要的超时数值。应用可以使用 |

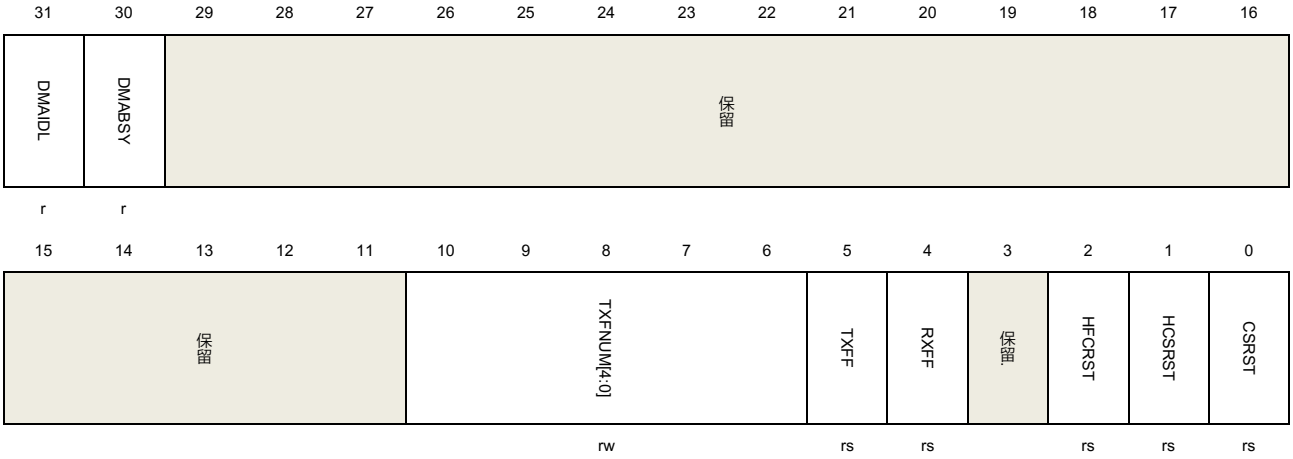
TOC[2:0]增加该数值(以PHY时钟为单位)。PHY时钟频率由使用的PHY所决定：48MHz(内部嵌入式PHY)和60MHz(外部ULPI PHY)。

### 全局复位控制寄存器(USBHS_GRSTCTL)

地址偏移：0x0010
复位值：0x8000 0000

应用通过该寄存器来复位内核的不同硬件特性。

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | DMAIDL | DMA空闲状态<br>该标志位体现了DMA是否在空闲状态<br>0：DMA在空闲状态<br>1：DMA不在空闲状态<br>**注意**：在设备和主机模式下，均可访问 |
| 30 | DMABSY | DMA忙标志位<br>该标志位体现了DMA是否忙<br>0：DMA不忙<br>1：DMA忙<br>**注意**：在设备和主机模式下，均可访问 |
| 29:11 | 保留 | 必须保持复位值。 |
| 10:6 | TXFNUM[4:0] | Tx FIFO数目<br>当本寄存器中TXFF控制位置位时，该标志位决定那个Tx FIFO会被冲刷<br>主机模式：<br>00000：仅非周期性Tx FIFO被冲刷<br>00001：仅周期性Tx FIFO被冲刷<br>1xxxx：周期性和非周期性Tx FIFO均被冲刷<br>其他：没有数据被冲刷<br>设备模式： |

00000：仅Tx FIFO0被冲刷
00001：仅Tx FIFO1被冲刷
…
00101：仅Tx FIFO5被冲刷
1XXXX：所有的Tx FIFO均被冲刷
其他：没有数据被冲刷

| | | |
|---|---|---|
| 5 | TXFF | Tx FIFO冲刷控制位<br>应用通过置位该控制位来冲刷Tx FIFO数据，并且TXFNUM[4:0]决定冲刷的FIFO数目。当冲刷完成后，硬件自动清除该控制位。置位该控制位后，应用应该等待该控制位清除，并且，在此之前USBHS不应有其他操作。<br>**注意**：在设备和主机模式下，均可访问 |
| 4 | RXFF | Rx FIFO冲刷控制位<br>应用通过置位该控制位来冲刷Rx FIFO数据。当冲刷完成后，硬件自动清除该控制位。置位该控制位后，应用应该等待该控制位清除，并且，在此之前USBHS不应有其他操作。<br>**注意**：在设备和主机模式下，均可访问 |
| 3 | 保留 | 必须保持复位值。 |
| 2 | HFCRST | 主机帧计数器复位<br>应用通过置位该控制位来复位USBHS内的帧计数器。该控制位置位后，接下来SOF的帧计数器将变为0。当复位操作完成后，硬件自动清除该控制位。置位该控制位后，应用应该等待该控制位清除，并且，在此之前USBHS不应有其他操作。<br>**注意**：仅在主机模式下访问 |
| 1 | HCSRST | HCLK软件复位<br>应用通过置位该控制位来复位ABH时钟域电路<br>在复位操作完成后，硬件自动清除该控制位。置位该控制位后，应用应该等待该控制位清除，并且，在此之前USBHS不应有其他操作。<br>**注意**：在设备和主机模式下，均可访问 |
| 0 | CSRST | USB内核软件复位<br>复位AHB和USB时钟域电路，以及大多数的寄存器。 |

### 全局中断标志寄存器(USBHS_GINTF)

地址偏移：0x0014
复位值：0x0400 0021

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WKUPIF | SESIF | DISCIF | IDPSC | 保留 | PTXFEIF | HCIF | HPIF | 保留 | | PXNCIF/ ISOONCIF | ISOINCIF | OEPIF | IEPIF | 保留 | |
| rc_w1 | rc_w1 | rc_w1 | rc_w1 | | r | r | r | | | rc_w1 | rc_w1 | r | r | | |

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EOPFIF | ISOOPDIF | ENUMF | RST | SP | ESP | 保留 | | GONAK | GNPINAK | NPTXFEIF | RXFNEIF | SOF | OTGIF | MFIF | COPM |
| rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | rc_w1 | | | r | r | r | r | rc_w1 | r | rc_w1 | r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | WKUPIF | 唤醒中断标志位<br>当在USB总线上检测到一个恢复信号(在设备模式下)或者一个远程唤醒信号(在主机模式下)，硬件将置位该中断标志位。<br>**注意**：在设备和主机模式下，均可访问 |
| 30 | SESIF | 会话中断标志位<br>当在A设备模式下检测到一个SRP会话请求或在B设备模式下B设备的Vbus变为可用时，硬件将置位该中断标志位<br>**注意**：在设备和主机模式下，均可访问 |
| 29 | DISCIF | 断开中断标志位<br>当设备断开后，将触发该标志位。<br>**注意**：仅在主机模式下访问 |
| 28 | IDPSC | ID引脚状态改变中断标志位<br>当ID引脚状态改变时，内核将置位该标志位<br>**注意**：在设备和主机模式下，均可访问 |
| 27 | 保留 | 必须保持复位值。 |
| 26 | PTXFEIF | 周期性Tx FIFO空中断标志位<br>当周期性发送FIFO半空或全空时，将触发该标志位。空阈值由USBHS_GAHBCS寄存器中周期性Tx FIFO空等级控制位(PTXFTH)决定。<br>**注意**：仅在主机模式下访问 |
| 25 | HCIF | 主机通道中断标志位<br>当在主机模式下其中一个通道挂起一个中断时，USBHS将置位该标志位。软件应该首先读取USBHS_HACHINT寄存器以获取通道号，然后读取相应的USBHS_HCHxINTF寄存器以获取产生中断的通道标志位。当产生通道中断的独立通道标志位被清除后，该中断标志位将自动清除。<br>**注意**：仅在主机模式下访问 |
| 24 | HPIF | 主机端口中断标志位<br>当USBHS在主机模式下检测到端口状态改变时，USB内核将置位该标志位。软件应该读取USBHS_HPCS寄存器以获取该中断源。当产生端口中断的标志被清除后，该中断标志位将自动清除。<br>**注意**：仅在主机模式下访问 |
| 23:22 | 保留 | 必须保持复位值。 |
| 21 | PXNCIF | 周期性传输未完成中断标志位 |

| | | |
|---|---|---|
| | | 在当前帧内，当帧结束时，周期性传输未完成，USBHS将置位该标志位(主机模式)。同步OUT传输未完成中断标志位 |
| | ISOONCIF | 在周期性帧结束时(由USBHS_DCFG寄存器的EOPFT控制位定义)，如果仍有同步OUT端点未完成传输，USBHS将置位该标志位(设备模式)。 |
| 20 | ISOINCIF | 同步IN传输未完成中断标志位<br>在周期性帧结束时(由USBHS_DCFG寄存器的EOPFT控制位定义)，如果仍有同步IN端点未完成传输，USBHS将置位该标志位(设备模式)。<br>**注意**：仅在设备模式下访问 |
| 19 | OEPIF | OUT端点中断标志位<br>当在设备模式下，其中一个OUT端点挂起一个中断时，USBHS将置位该中断标志位。软件应该首先读取USBHS_DAEPINT寄存器以获取设备号，然后读取相应的USBHS_ DOEPxINTF寄存器以获取产生中断的端点标志位。当产生中断的相应端点标志位被清除后，该中断标志位被自动清除。<br>**注意**：仅在设备模式下访问 |
| 18 | IEPIF | IN端点中断标志位<br>当在设备模式下，其中一个IN端点挂起一个中断时，USBHS将置位该标志位。软件应该首先读取USBHS_DAEPINT寄存器以获取设备号，然后读取相应的USBHS_DIEPxINTF寄存器以获取产生中断的端点标志位。当相应产生中断的端点标志位被清除后，该中断标志位被自动清除。 |
| 17:16 | 保留 | 必须保持复位值。 |
| 15 | EOPFIF | 周期性帧结束中断标志位<br>当一帧内USB总线时间已经达到USBHS_DCFG寄存器中EOPFT控制位所定义的数值时，USBHS将置位该中断标志位。<br>**注意**：仅在设备模式下访问 |
| 14 | ISOOPDIF | 同步OUT包丢失中断标志位<br>如果USBHS接收到一个同步OUT包，但是Rx FIFO没有足够的空间来接收该OUT包，USBHS将置位该标志位。<br>**注意**：仅在设备模式下访问 |
| 13 | ENUMF | 枚举完成中断标志位<br>在速度枚举完成后，USBHS将置位该中断标志位。软件能够读取USBHS_DSTAT寄存器以获取当前设备速度。<br>**注意**：仅在设备模式下访问 |
| 12 | RST | USB复位中断标志位<br>当USBHS在USB总线上检测到一个USB复位信号后，USBHS将置位该中断标志位。<br>**注意**：仅在设备模式下访问 |
| 11 | SP | USB挂起中断标志位<br>当USBHS检测到USB总线空闲3ms并且进入挂起状态，USBHS将置位该中断标志位。<br>**注意**：仅在设备模式下访问 |

| | | |
|---|---|---|
| 10 | ESP | 早期挂起中断标志位<br>当USBHS检测到USB总线空闲3ms时，USBHS将置位该中断标志位。 |
| 9:8 | 保留 | 必须保持复位值。 |
| 7 | GONAK | 全局OUT NAK有效标志位<br>软件能够向USBHS_DCTL寄存器的SGONAK控制位写1，并且USBHS将会在SGONAK写入有效后，置位GONAK标志位。软件可通过向USBHS_DCTL寄存器的CGONAK控制位写1，清除该标志位。<br>**注意**：仅在设备模式下可访问 |
| 6 | GNPINAK | 全局非周期性IN NAK有效标志位<br>软件能够向USBHS_DCTL寄存器中的SGINAK控制位写1，并且USBHS将会在SGINAK写入有效后，置位GNPINAK标志位。软件可通过向USBHS_DCTL寄存器的CGONAK控制位写1，清除该标志位。<br>**注意**：仅在设备模式下可访问 |
| 5 | NPTXFEIF | 非周期性Tx FIFO空中断标志位<br>当非周期性Tx FIFO为半空或全空时，将置位该中断标志位。该阈值由USBHS_GAHBCS寄存器中的非周期Tx FIFO空等级控制位(TXFTH)决定。<br>**注意**：仅在主机模式下访问 |
| 4 | RXFNEIF | Rx FIFO非空中断标志位<br>当至少有一个包或状态条目在Rx FIFO中时，USBHS将置位该标志位。<br>**注意**：在主机和设备模式下，均可访问 |
| 3 | SOF | 帧起始中断标志位<br>**主机模式:**<br>当准备在USB总线上发送一个SOF或保持有效信号，USBHS将置位该中断标志位。软件可以通过写1清除该中断标志位。<br>**设备模式:**<br>当USBHS接收到一个SOF令牌包后，USBHS置位该标志位。应用可以读取设备状态寄存器以获取当前帧号。软件可以通过写1清除该中断标志位。<br>**注意**：在设备和主机模式下，均可访问 |
| 2 | OTGIF | OTG中断标志位<br>当USBHS_GOTGINTF寄存器中标志位产生一个中断时，USBHS置位该中断标志位。软件应该读取USBHS_GOTGINTF寄存器以获取产生该中断的信号源，当USBHS_GOTGINTF寄存器中产生该中断的标志位被清除后，该中断标志位也被自动清除。<br>**注意**：在设备和主机模式下，均可访问 |
| 1 | MFIF | 模式错误中断标志位<br>如果软件在设备模式下操作仅主机可访问的寄存器或者在主机模式下操作仅设备可访问的寄存器，USBHS将置位该中断标志位。这些错误操作不会产生作用。<br>**注意**：在主机和设备模式下，均可访问 |
| 0 | COPM | 当前操作模式<br>0：设备模式 |

1：主机模式

**注意**：在主机和设备模式下，均可访问

### 全局中断使能寄存器(USBHS_GINTEN)

地址偏移：0x0018

复位值：0x0000 0000

这个寄存器同全局中断标志寄存器(USBHS_GINTF)一起工作来中断应用程序。当中断使能位被禁止后，相应的中断就不会产生。然而，相应的全局中断标志位依然会被置位。

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WKUPIE | SESIE | DISCIE | IDPSCIE | 保留 | PTXFEIE | HCIE | HPIE | 保留 | | PXNCIE/ ISOONCIE | ISOINCIE | OEPIE | IEPIE | 保留 | |
| rw | rw | rw | rw | | rw | rw | r | | | rw | rw | rw | rw | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EOPFIE | ISOOPDIE | ENUMFIE | RSTIE | SPIE | ESPIE | 保留 | | GONAKIE | GNPINAKIE | NPTXFEIE | RXFNEIE | SOFIE | OTGIE | MFIE | 保留 |
| rw | rw | rw | rw | rw | rw | | | rw | rw | rw | rw | rw | rw | rw | |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | WKUPIE | 唤醒中断使能<br>0：禁用唤醒中断<br>1：使能唤醒中断<br>**注意**：在主机和设备模式下，均可访问 |
| 30 | SESIE | 会话中断使能<br>0：禁用会话中断<br>1：使能会话中断<br>**注意**：在主机和设备模式下，均可访问 |
| 29 | DISCIE | 断开中断使能<br>0：禁用断开中断<br>1：使能断开中断<br>**注意**：仅在设备模式下使用 |
| 28 | IDPSCIE | ID引脚状态改变中断使能<br>0：禁用连接器ID引脚状态中断<br>1：使能连接器ID引脚状态中断<br>**注意**：在主机和设备模式下，均可访问 |
| 27 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 26 | PTXFEIE | 周期性Tx FIFO空中断使能<br>0：禁用周期性Tx FIFO空中断<br>1：使能周期性Tx FIFO空中断<br>**注意**：仅在主机模式下访问 |
| 25 | HCIE | 主机通道中断使能<br>0：禁用主机通道中断<br>1：使能主机通道中断<br>**注意**：仅在主机模式下访问 |
| 24 | HPIE | 主机端口中断使能<br>0：禁止主机端口中断<br>1：使能主机端口中断<br>**注意**：仅在主机模式下访问 |
| 23:22 | 保留 | 必须保持复位值。 |
| 21 | PXNCIE | 周期性传输未完成中断使能<br>0：禁止周期性未完成传输中断<br>1：使能周期性未完成传输中断<br>**注意**：仅在主机模式下访问 |
| | ISOONCIE | 同步OUT传输未完成中断使能<br>0：禁止同步OUT传输未完成中断<br>1：使能同步OUT传输未完成中断<br>**注意**：仅在设备模式下访问 |
| 20 | ISOINCIE | 同步IN传输未完成中断使能<br>0：禁止同步IN传输未完成中断<br>1：使能同步IN传输未完成中断<br>**注意**：仅在设备模式下访问 |
| 19 | OEPIE | OUT端点中断使能<br>0：禁止OUT端点中断<br>1：使能OUT端点中断<br>**注意**：仅在设备模式下访问 |
| 18 | IEPIE | IN端点中断使能<br>0：禁止IN端点中断<br>1：使能IN端点中断<br>**注意**：仅在设备模式下访问 |
| 17:16 | 保留 | 必须保持复位值。 |
| 15 | EOPFIE | 周期性帧结束中断使能<br>0：禁止周期性帧结束中断<br>1：使能周期性帧结束中断<br>**注意**：仅在设备模式下访问 |
| 14 | ISOOPDIE | 同步OUT包丢失中断使能 |

| | | |
|---|---|---|
| | | 0：禁止同步OUT包丢失中断<br>1：使能同步OUT包丢失中断<br>**注意**：仅在设备模式下访问 |
| 13 | ENUMFIE | 枚举完成中断使能<br>0：禁止枚举完成中断<br>1：使能枚举完成中断<br>**注意**：仅在设备模式下访问 |
| 12 | RSTIE | USB复位中断使能<br>0：禁止USB复位中断<br>1：使能USB复位中断<br>**注意**：仅在设备模式下访问 |
| 11 | SPIE | USB挂起中断使能<br>0：禁止USB挂起中断<br>1：使能USB挂起中断<br>**注意**：仅在设备模式下访问 |
| 10 | ESPIE | 早期挂起中断使能<br>0：禁止早期挂起中断<br>1：使能早期挂起中断<br>**注意**：仅在设备模式下访问 |
| 9:8 | 保留 | 必须保持复位值。 |
| 7 | GONAKIE | 全局OUT NAK有效中断使能<br>0：禁止全局OUT NAK有效中断<br>1：使能全局OUT NAK有效中断<br>**注意**：仅在设备模式下访问 |
| 6 | GNPINAKIE | 全局非周期性IN NAK有效中断使能<br>0：禁止全局非周期性IN NAK有效中断<br>1：使能全局非周期性IN NAK有效中断<br>**注意**：仅在设备模式下访问 |
| 5 | NPTXFEIE | 非周期性发送FIFO空中断使能<br>0：禁止非周期性发送FIFO空中断<br>1：使能非周期性发送FIFO空中断<br>**注意**：仅在主机模式下访问 |
| 4 | RXFNEIE | 接收FIFO非空中断使能<br>0：禁止接收FIFO非空中断<br>1：使能接收FIFO非空中断<br>**注意**：在设备模式与主机模式下，均可访问 |
| 3 | SOFIE | 帧首中断使能<br>0：禁止帧首中断<br>1：使能帧首中断 |

| | | |
|---|---|---|
| | | **注意**：在设备模式下与主机模式下，均可访问 |
| 2 | OTGIE | OTG中断使能<br>0：禁止OTG中断<br>1：使能OTG中断<br>**注意**：在设备模式下与主机模式下，均可访问 |
| 1 | MFIE | 模式错误中断使能<br>0：禁止模式错误中断<br>1：使能模式错误中断<br>**注意**：在设备模式下与主机模式下，均可访问 |
| 0 | 保留 | 必须保持复位值。 |

### 全局接收状态读取/接收状态读取和弹出寄存器 (USBHS_GRSTATR/USBHS_GRSTATP)

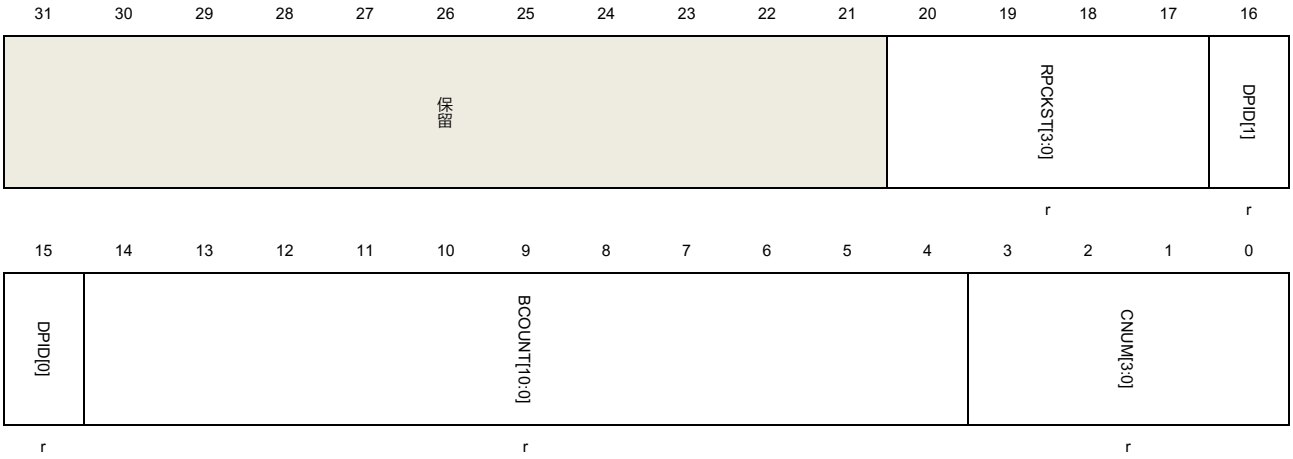
读地址偏移：0x001C
弹出地址偏移：0x0020
复位值：0x0000 0000

对接收状态读寄存器的读操作，将返回接收 FIFO 中顶部的条目。对接收状态读取和弹出寄存器的读操作，将额外的弹出 Rx FIFO 的顶部条目。

在主机模式和设备模式下，Rx FIFO 中的条目具有不同的含义。当全局中断标志寄存器(USBHS_GINTF)中的接收 FIFO 非空中断标志位(RXFNEIF)置位后，软件应该读取该寄存器。

该寄存器只能按字(32位)访问

**主机模式：**




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:21 | 保留 | 必须保持复位值。 |
| 20:17 | RPCKST[3:0] | 接收包状态<br>0010：接收到IN数据包 |

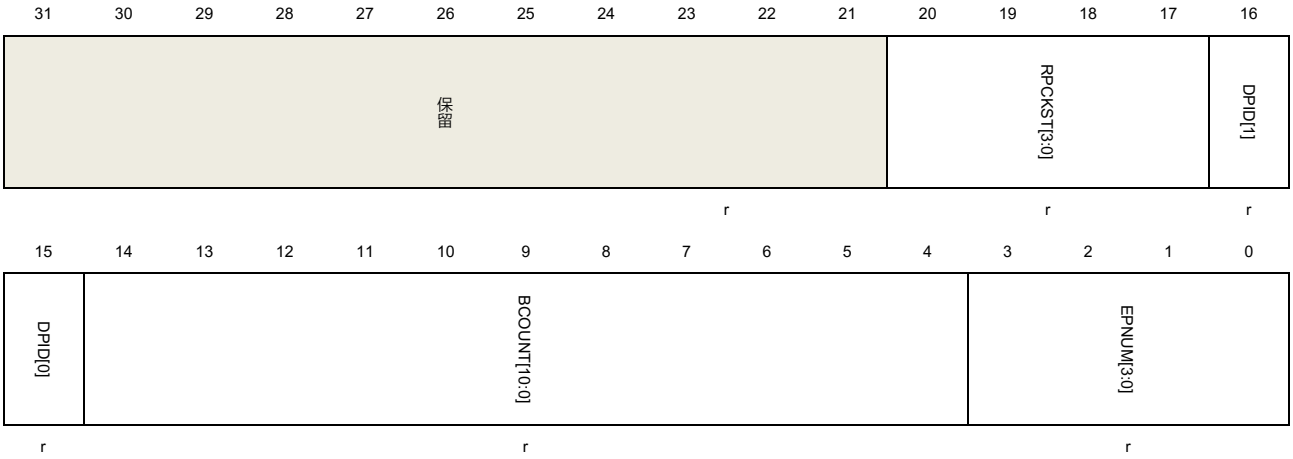
0011：IN传输完成(如果取出，触发一个中断)
0101：数据翻转错误(如果取出，触发一个中断)
0111：通道中止(如果取出，触发一个中断)
其他：保留

16:15 DPID[1:0] 数据PID
接收包的数据PID
00：DATA0
10：DATA1
01：DATA2
11：MDATA

14:4 BCOUNT[10:0] 字节数
接收IN数据包字节数。

3:0 CNUM[3:0] 通道数
当前接收包所属通道编号。

**设备模式：**



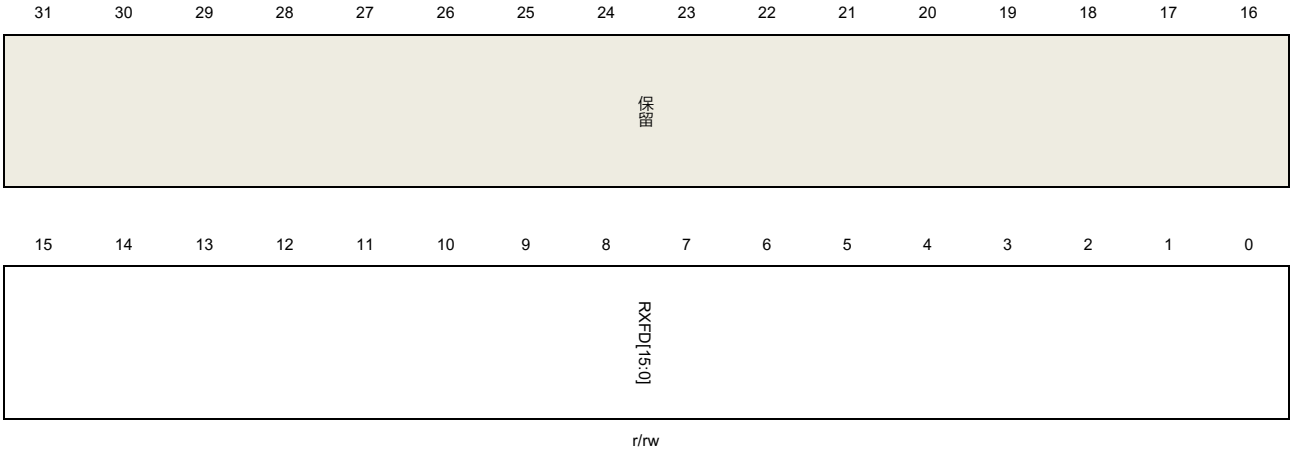

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:21 | 保留 | 必须保持复位值。 |
| 20:17 | RPCKST[3:0] | 接收包状态<br>0001：全局OUT NAK(产生一个中断)<br>0010：接收到OUT数据包<br>0011：OUT传输完成(产生一个中断)<br>0100：SETUP传输完成(产生一个中断)<br>0110：接收到SETUP数据包<br>其他：保留 |
| 16:15 | DPID[1:0] | 数据PID<br>接收到OUT数据包的数据PID<br>00：DATA0<br>10：DATA1<br>01：DATA2 |

| | | |
|---|---|---|
| | | 11：MDATA |
| 14:4 | BCOUNT[10:0] | 字节数<br>接收数据包的字节数 |
| 3:0 | EPNUM[3:0] | 端点号<br>当前接收包所属端点编号 |

### 全局接收 FIFO 长度寄存器(USBHS_GRFLEN)

地址偏移：0x0024
复位值：0x0000 0200

该寄存器只能按字(32位)访问

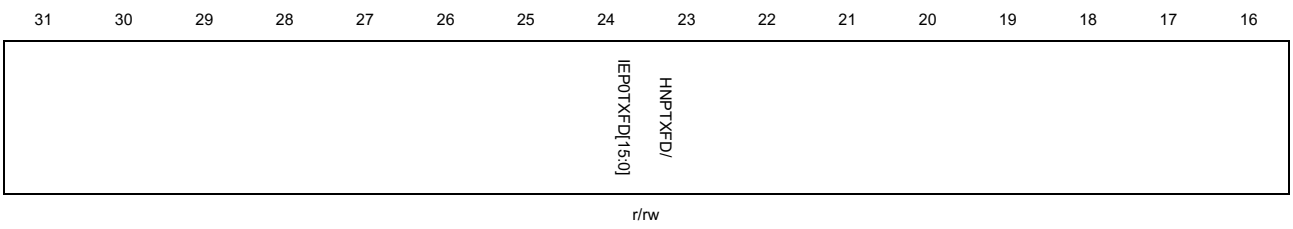


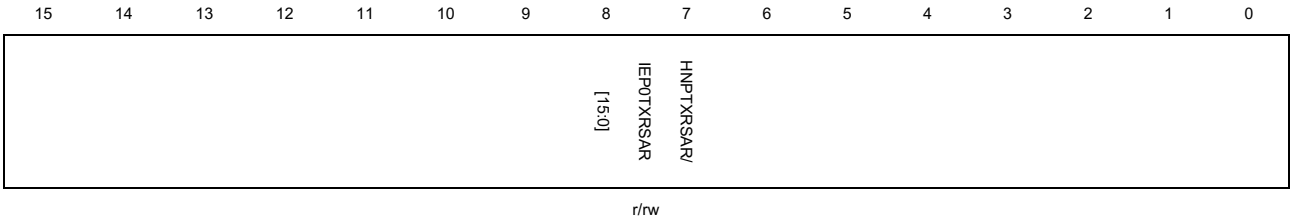

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | RXFD[15:0] | Rx FIFO 深度<br>以32位字计数<br>1<=RXFD<=1024 |

### 主机非周期性发送 FIFO 长度寄存器/设备 IN 端点 0 发送 FIFO 长度寄存器 (USBHS_HNPTFLEN_DIEP0TFLEN)

地址偏移：0x0028
复位值：0x0200 0200

该寄存器只能按字(32位)访问






**主机模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | HNPTXFD[15:0] | 主机非周期性Tx FIFO深度<br>以32位字计数<br>1<=HNPTXFD<=1024 |
| 15:0 | HNPTXRSAR[15:0] | 主机非周期性Tx RAM起始地址<br>非周期性发送FIFO RAM的起始地址 |

**设备模式：**

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | IEP0TXFD[15:0] | 输入端点0 Tx FIFO深度<br>以32位字计数<br>16<=IEP0TXFD<=140 |
| 15:0 | IEP0TXRSAR[15:0] | 输入端点0 TX RAM起始地址<br>端点0发送FIFO RAM的起始地址 |

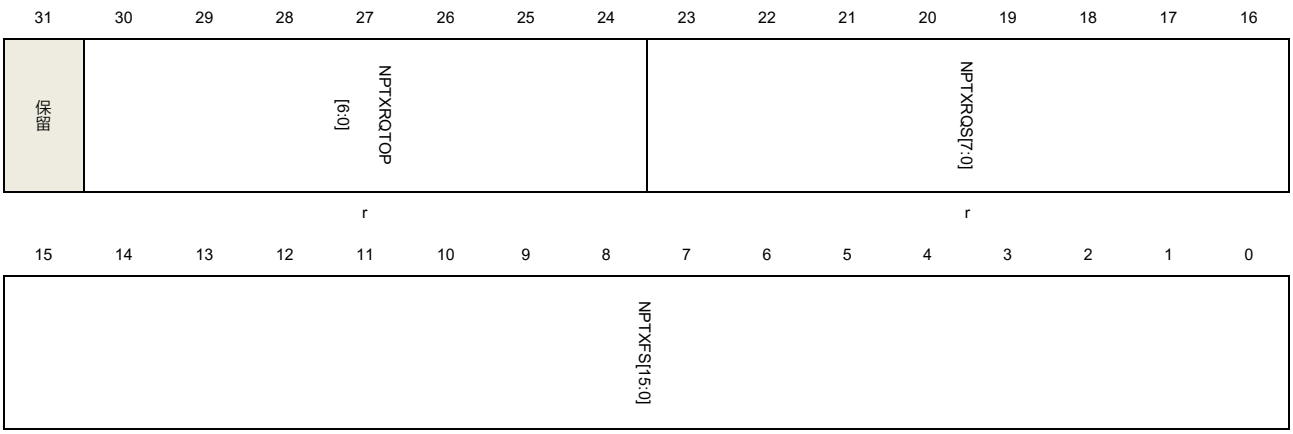
### 主机非周期性发送 FIFO/队列状态寄存器(USBHS_HNPTFQSTAT)

地址偏移：0x002C
复位值：0x0008 0200

该寄存器只能按字(32位)访问

该寄存器反映了非周期性Tx FIFO和请求队列的当前状态。请求队列包括在主机模式下的IN、OUT或其他请求条目。

**注意：**在设备模式下，该寄存器不可用。



r

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30:24 | NPTXRQTOP[6:0] | 非周期性发送请求队列的顶部条目<br>在非周期性传输请求队列中的条目。<br>位30:27：通道号<br>位26:25：<br>- 00：IN/OUT令牌<br>- 01：0长度OUT包<br>- 11：通道中止请求<br>位24：结束标志位，表明所选通道的最后一个条目 |
| 23:16 | NPTXRQS[7:0] | 非周期性发送请求队列空间<br>非周期性请求队列的剩余空间<br>0：请求队列空<br>1：1个条目<br>2：2个条目<br>…<br>n：n个条目 (0<=n<=8)<br>其他：保留 |
| 15:0 | NPTXFS[15:0] | 非周期性Tx FIFO空间<br>非周期性发送FIFO剩余空间<br>以32位字计数<br>0：非周期性Tx FIFO为空<br>1：1个字<br>2：2个字<br>n：n个字(0<=n<=NPTXFD)<br>其他：保留 |

### 全局内核配置寄存器(USBHS_GCCFG)

地址偏移：0x0038
复位值：0x0000 0000

该寄存器只能按字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | | | | | | VBUSIG | SOFOEN | VBUSBCEN | VBUSACEN | 保留 | PWRON |
| | | | | | | | | | | rw | rw | rw | rw | | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

| 保留 |
|---|

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:22 | 保留 | 必须保持复位值。 |
| 21 | VBUSIG | $V_{BUS}$忽略<br>当该控制位被置位，USBHS并不监测$V_{BUS}$引脚电压，并且认为在主机和设备模式下，$V_{BUS}$电压一直有效，然后可释放$V_{BUS}$引脚作为其他用途。<br>0：$V_{BUS}$不被忽略<br>1：$V_{BUS}$被忽略，并认为$V_{BUS}$电压一直有效 |
| 20 | SOFOEN | SOF输出使能<br>0：SOF脉冲输出禁止<br>1：SOF脉冲输出使能 |
| 19 | VBUSBCEN | $V_{BUS}$ B设备比较器使能<br>0：$V_{BUS}$ B设备比较器禁止<br>1：$V_{BUS}$ B设备比较器使能 |
| 18 | VBUSACEN | $V_{BUS}$ A设备比较器使能<br>0：$V_{BUS}$ A设备比较器禁止<br>1：$V_{BUS}$ A设备比较器使能 |
| 17 | 保留 | 必须保持复位值。 |
| 16 | PWRON | 上电<br>该控制位为内部嵌入式全速PHY的电源开关<br>0：嵌入式全速PHY掉电<br>1：嵌入式全速PHY上电 |
| 15:0 | 保留 | 必须保持复位值。 |

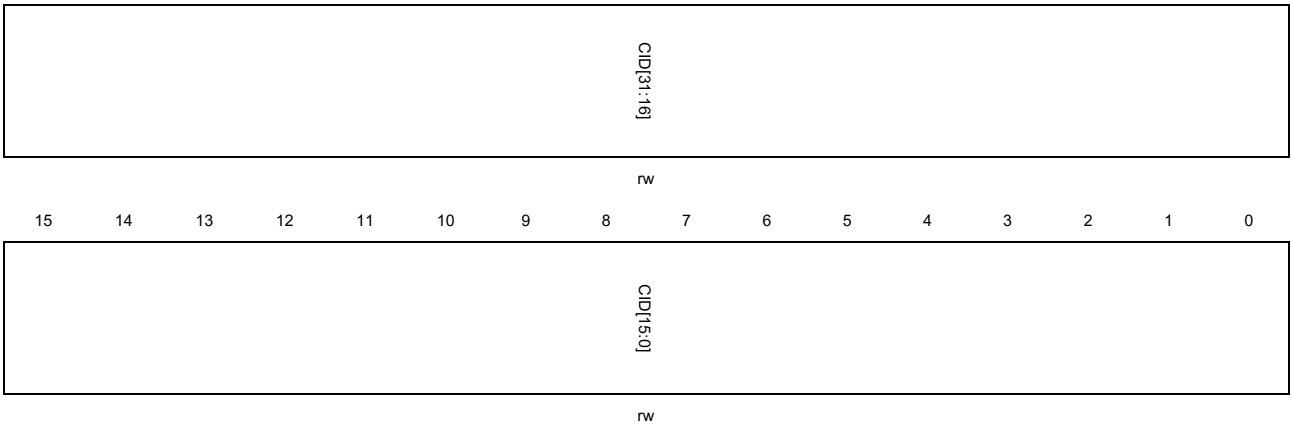
### 内核 ID 寄存器(USBHS_CID)

地址偏移：0x003C
复位值：0x0000 1000

该寄存器包含产品ID

该寄存器只能按字(32位)访问

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

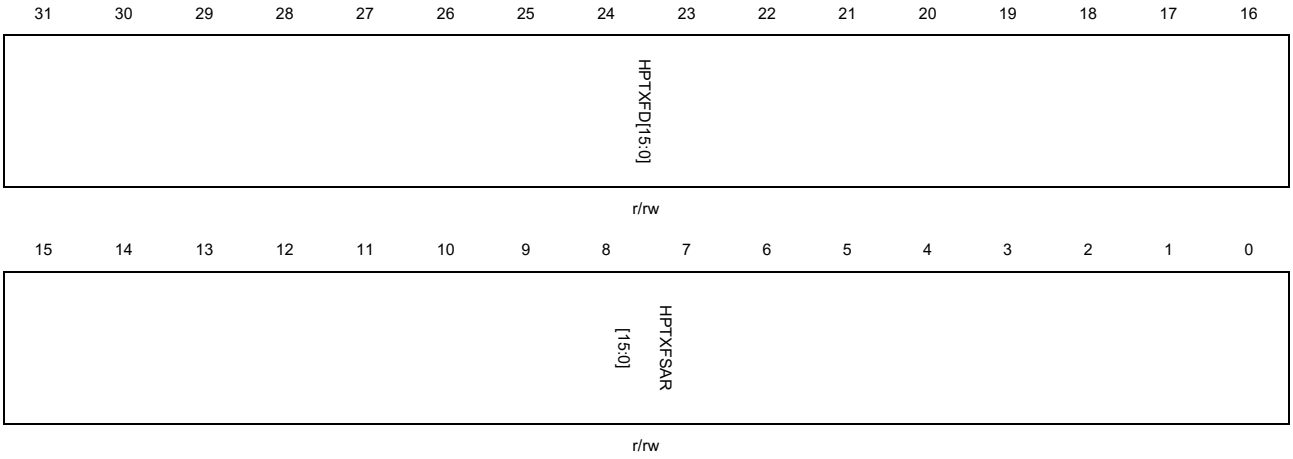



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | CID[31:0] | 内核ID<br>软件能够写入或读取该域值，并利用该域值为应用产生一个唯一ID。 |

### 主机周期性发送 FIFO 长度寄存器(USBHS_HPTFLEN)

地址偏移：0x0100
复位值：0x0200 0600

该寄存器只能按字(32位)访问

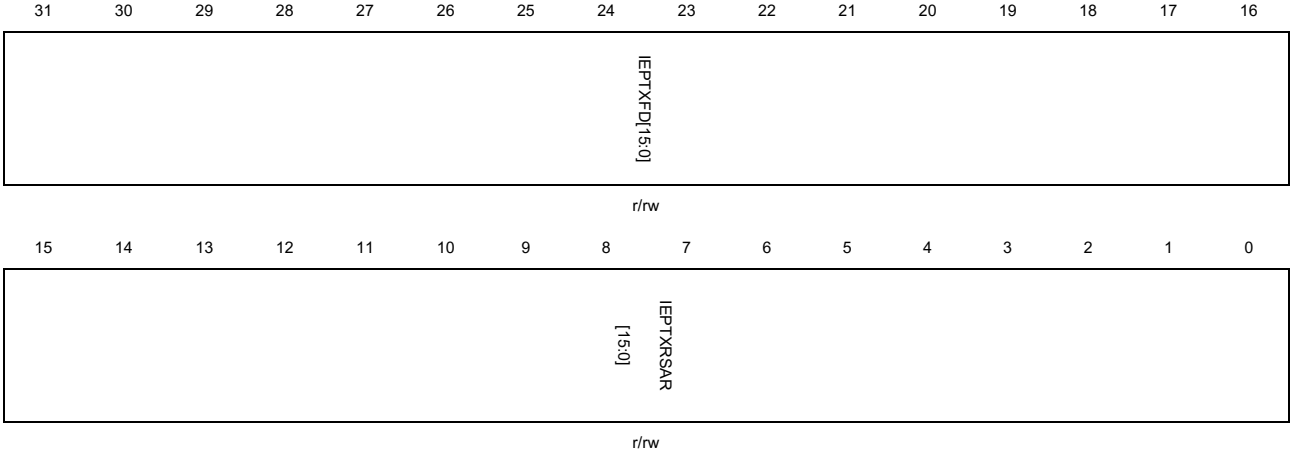



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | HPTXFD[15:0] | 主机周期性Tx FIFO深度<br>以32位字计数<br>1<=HPTXFD<=1024 |
| 15:0 | HPTXFSAR[15:0] | 主机周期性Tx RAM起始地址<br>主机周期性发送FIFO RAM起始地址 |

**设备 IN 端点发送 FIFO 长度寄存器(USBHS_DIEPxTFLEN)(x = 1..3，其中 x 为 FIFO 编号)**

地址偏移：0x0104 + (FIFO编号 - 1) × 0x04
复位值：0x0200 0400

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | IEPTXFD[15:0] | IN端点Tx FIFO深度<br>以32位字计数<br>1<=HPTXFD<=1024 |
| 15:0 | IEPTXRSAR[15:0] | IN端点FIFOx Tx RAM起始地址<br>以32位字为单位的IN端点发送FIFOx起始地址 |

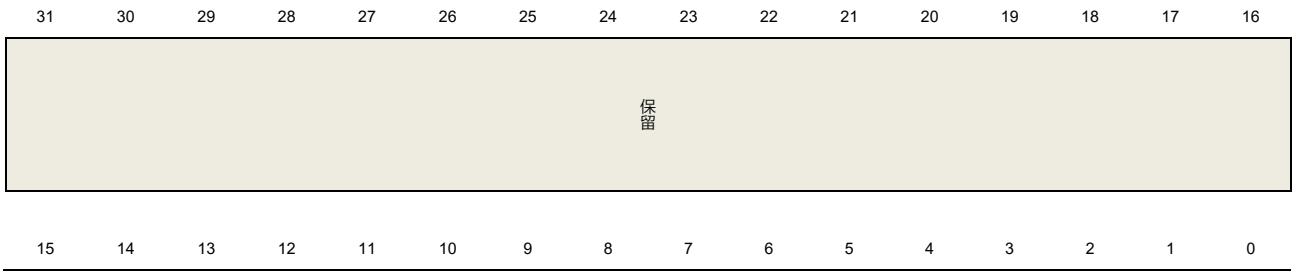
## 29.7.2. 主机控制和状态寄存器

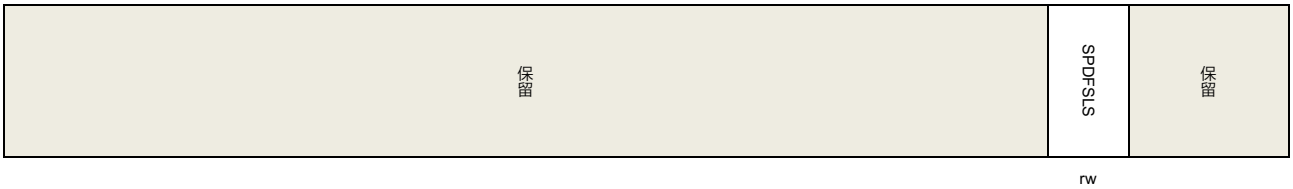
**主机控制寄存器(USBHS_HCTL)**

地址偏移：0x0400
复位值：0x0000 0000

在主机模式下，上电后，该寄存器有USB内核配置。主机初始化后，无需修改。

该寄存器只能按字(32位)访问





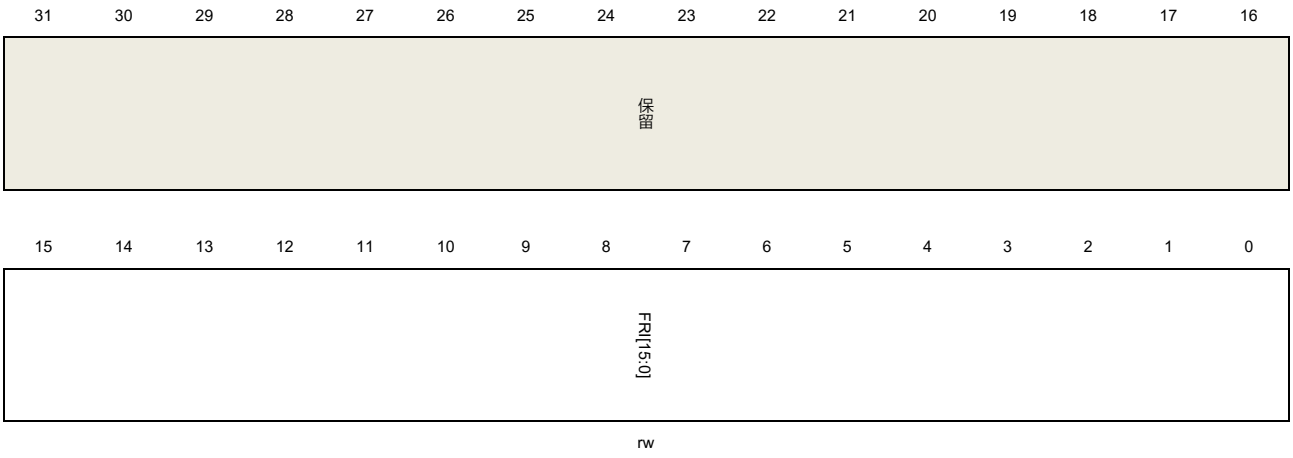

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:3 | 保留 | 必须保持复位值。 |
| 2 | SPDFSLS | 限制速度为FS和LS<br>软件可以利用该控制位限制USBHS的枚举速度为FS/LS，并且使USBHS在复位的过程中不执行高速枚举。该控制位只在使用外部ULPI PHY时有用，因为内部嵌入式PHY只支持全速和低速。<br>0：不限制速度<br>1：限制速度仅为FS/LS |
| 1:0 | 保留 | 必须保持复位值。 |

### 主机帧间隔寄存器(USBHS_HFT)

地址偏移：0x0404
复位值：0x0000 BB80

当USBHS控制器正在枚举中时，该寄存器为当前枚举速度设置帧间隔。

该寄存器只能按字(32位)访问



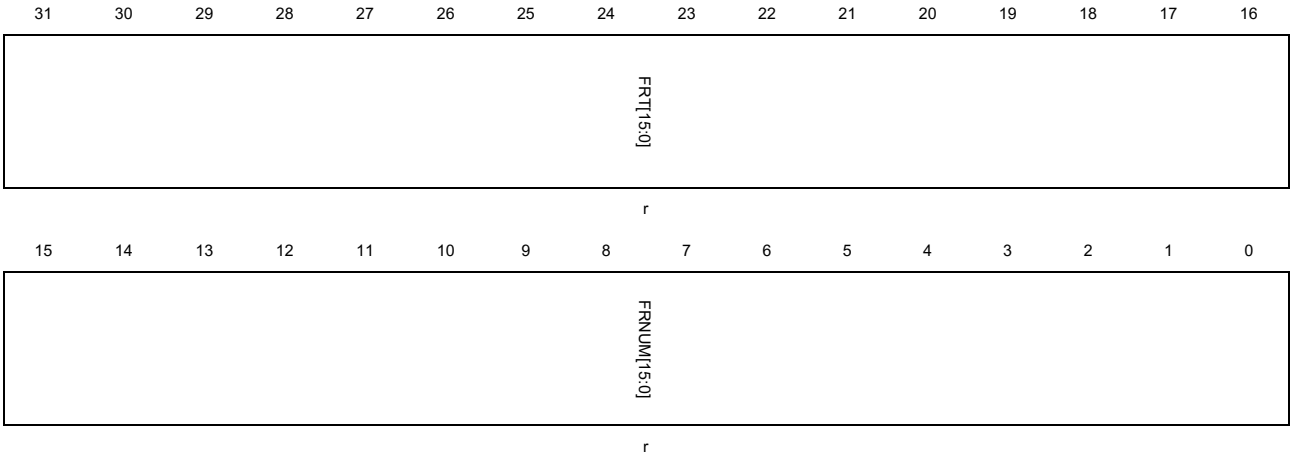

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | FRI[15:0] | 帧间隔<br>该值描述了以PHY时钟为单位的帧周期。每次端口复位操作后，端口被使能，USBHS根据当前速度，采用一个固有值，并且软件可以向该位域写值以改变该固有值。该值需要采用以下描述的频率来进行计算：<br>内部嵌入式全速PHY |

全速：48MHz
低速：6MHz
外部ULPI PHY
60MHz

### 主机帧信息保持寄存器(USBHS_HFINFR)

地址偏移：0x0408
复位值：0xBB80 0000

该寄存器只能按字(32位)访问



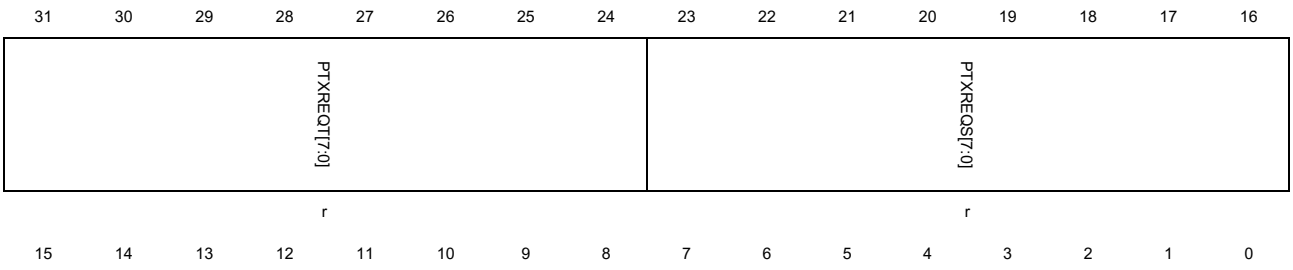

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | FRT[15:0] | 帧剩余时间<br>该位域以PHY时钟为单位反映了当前帧剩余时间。 |
| 15:0 | FRNUM[15:0] | 帧号<br>该位域反映了当前帧的帧号，当其增加到0x3FFF后，其值变为0。 |

### 主机周期性发送 FIFO/队列状态寄存器(USBHS_HPTFQSTAT)

地址偏移：0x0410
复位值：0x0008 0200

该寄存器反映了主机周期性Tx FIFO和请求队列的当前状态。请求队列包括在主机模式下的IN、OUT或其他请求条目。

该寄存器只能按字(32位)访问



| PTXFS[15:0] |
|---|
| r |

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:24 | PTXREQT[7:0] | 周期性Tx 请求队列的顶部条目<br>在周期性发送请求队列中的条目<br>位30:27：通道号<br>位26:25：<br>- 00：IN/OUT 令牌<br>- 01：0长度OUT包<br>- 11：通道中止请求<br>位24：中止标志，指示所选通道的最后一个条目 |
| 23:16 | PTXREQS[7:0] | 周期性发送请求队列空间<br>周期性发送请求队列剩余空间<br>0：请求队列为空<br>1：1个条目<br>2：2个条目<br>…<br>n：n个条目 (0<=n<=8)<br>其他：保留 |
| 15:0 | PTXFS[15:0] | 周期性发送FIFO空间<br>周期性发送FIFO剩余空间<br>以32位字计数<br>0：周期性发送FIFO为空<br>1：1个字<br>2：2个字<br>n：n个字(0<=n<=PTXFD))<br>其他：保留 |

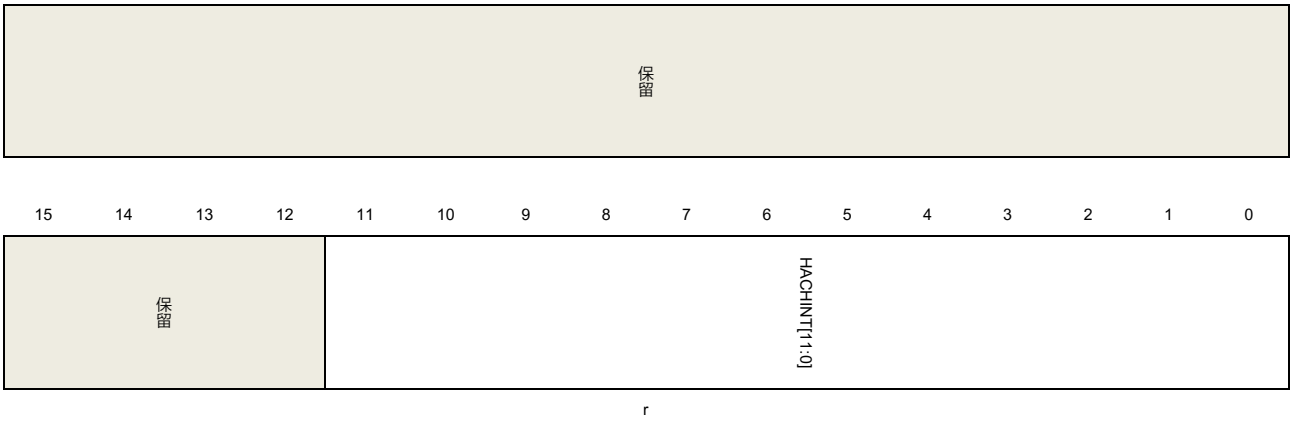
### 主机所有通道中断寄存器(USBHS_HACHINT)

地址偏移：0x0414
复位值：0x0000 0000

当触发一个通道中断时，USBHS在该寄存器中置位相应的位，并且软件可以读取该寄存器以获取产生中断的通道。

该寄存器只能按字(32位)访问

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



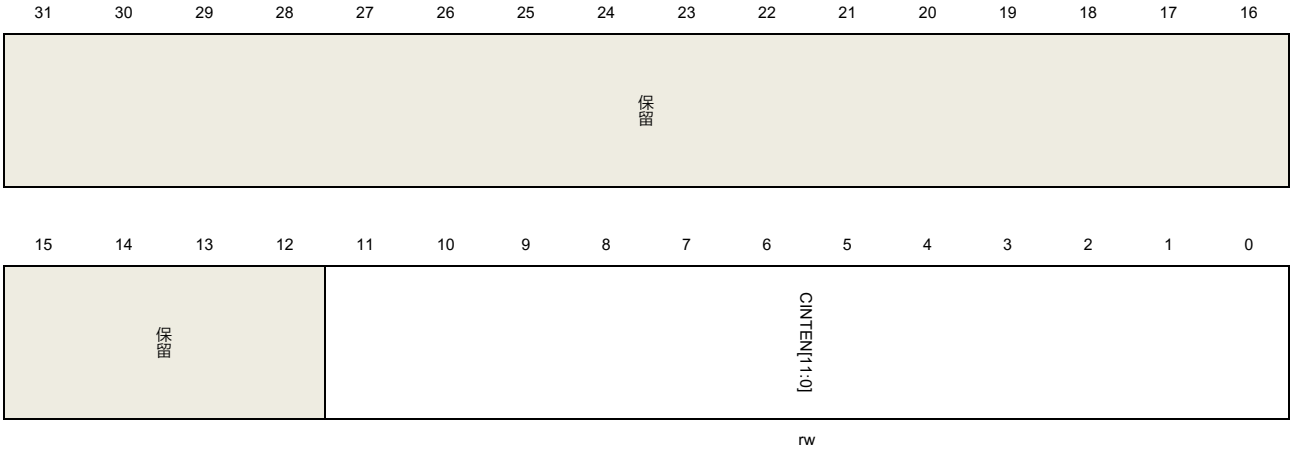

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11:0 | HACHINT[11:0] | 主机所有通道中断<br>每一位表示一个通道：位0代表通道0，位1表示通道1 |

### 主机所有通道中断使能寄存器(USBHS_HACHINTEN)

地址偏移：0x0418
复位值：0x0000 0000

软件可以使用该寄存器使能或禁用一个通道的中断。只有该寄存器中相应通道的中断使能控制位被置位，USBHS_GINTF寄存器中的通道中断标志位HCIF标志位才可产生。

该寄存器只能按字(32位)访问



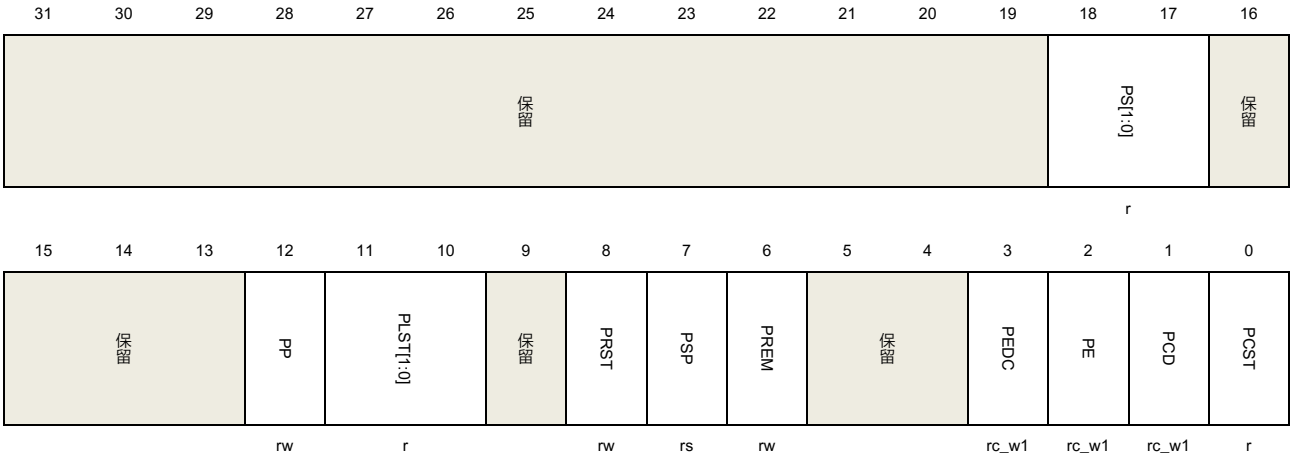

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11:0 | CINTEN[11:0] | 通道中断使能<br>0：禁用通道n中断<br>1：使能通道n中断<br>每一位表示一个通道：位0代表通道0，位11代表通道11 |

### 主机端口控制和状态寄存器(USBHS_HPCS)

地址偏移：0x0440
复位值：0x0000 0000

该寄存器控制端口行为，并且也包含一些反映端口状态的标志位。如果本寄存器中的PRST、PEDC和PCD标志位被USBHS置位的话，USBHS_GINTF寄存器中的HPIF标志位会被置位。

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:19 | 保留 | 必须保持复位值。 |
| 18:17 | PS[1:0] | 端口速度<br>反映连接到该端口的设备的枚举速度。<br>00：高速<br>01：全速<br>10：低速<br>其他：保留 |
| 16:13 | 保留 | 必须保持复位值。 |
| 12 | PP | 端口供电<br>在端口被使用后，该控制位应该被置位。由于USBHS不具有电源供应能力，它只能使用该控制位以获取该端口是否在供电状态。软件应该在设置该控制位之前，保证在$V_{BUS}$引脚上具有电源供应。<br>0：端口掉电<br>1：端口供电 |
| 11:10 | PLST[1:0] | 端口线状态<br>反映USB数据线当前状态<br>位10：DP线状态<br>位11：DM线状态 |
| 9 | 保留 | 必须保持复位值。 |
| 8 | PRST | 端口复位 |

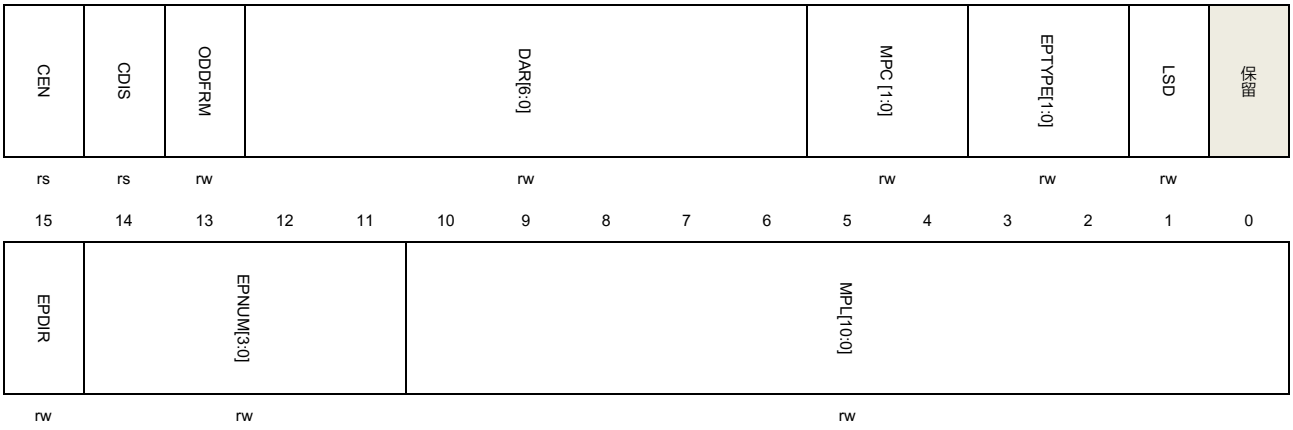
| | | |
|---|---|---|
| | | 应用通过设置该控制位以在USB端口上启动一个复位信号。当应用希望停止复位信号时，应用应该清除该控制位。<br>0：端口不在复位状态<br>1：端口处于复位状态 |
| 7 | PSP | 端口挂起<br>应用设置该控制位来将端口进入挂起状态。当该控制位被置位后，端口停止发送SOF令牌包。该控制位只能够通过以下操作清除。<br>– 应用置位该寄存器中的PRST控制位<br>– 置位该寄存器中的PREM控制位<br>– 检测到一个远程唤醒信号<br>– 检测到一个设备断开<br>0：端口不在挂起状态<br>1：端口处于挂起状态 |
| 6 | PREM | 端口恢复<br>应用通过置位该控制位以在USB端口上启动一个恢复信号。当应用希望停止恢复信号时，应用可以清除该控制位。 |
| 5:4 | 保留 | 必须保持复位值。 |
| 3 | PEDC | 端口使能/禁止更改<br>当该寄存器中的位2端口使能控制位更改时，USB内核置位该标志位。 |
| 2 | PE | 端口使能<br>当USB复位信号完成后，USBHS自动置位该位，并且该位不可由软件置位。<br>该位可通过以下事件清除：<br>– 一个断开状态<br>– 软件清除该位<br>0：端口禁止<br>1：端口使能 |
| 1 | PCD | 端口连接检测<br>当检测到设备连接时，USBHS置位该标志位。可通过向该位写1清除该标志位。 |
| 0 | PCST | 端口连接状态<br>0：设备没有连接到该端口<br>1：设备连接到该端口 |

### 主机通道 x 控制寄存器(USBHS_HCHxCTL)(x = 0..11， 其中 x 为主机通道号)

地址偏移：0x0500 + (主机通道号 × 0x20)
复位值：0x0000 0000

该寄存器只能按字(32位)访问

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




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | CEN | 通道使能<br>由应用设置，并且由USBHS清除<br>0：通道禁止<br>1：通道使能<br>软件应该遵循操作指南来禁用或者使能一个通道 |
| 30 | CDIS | 通道禁止<br>软件可以置位该控制位，来从处理事务中禁用该通道。软件应该遵循操作指南来禁用或者使能一个通道。 |
| 29 | ODDFRM | 奇偶帧控制<br>对于周期性传输(中断或同步传输)，该位控制将要处理的通道事务为奇数帧还是偶数帧。 |
| 28:22 | DAR[6:0] | 设备地址<br>与该通道通信的USB设备地址。 |
| 21:20 | MPC[1:0] | 多包计数<br>对于周期性传输，该位域指定主机每个微帧必须执行的事务数量。对于非周期性传输，该位域指定在内部DMA引擎更改仲裁之前，DMA为此通道获取或写入的包数量。<br>00：保留<br>01：每微帧发出1个事务<br>10：每微帧发出2个事务<br>11：每微帧发出3个事务 |
| 19:18 | EPTYPE[1:0] | 端点类型<br>与该通道通信的端点的传输类型<br>00：控制<br>01：同步<br>10：批量<br>11：中断 |
| 17 | LSD | 低速设备<br>与该通道通信的设备是一个低速设备。 |

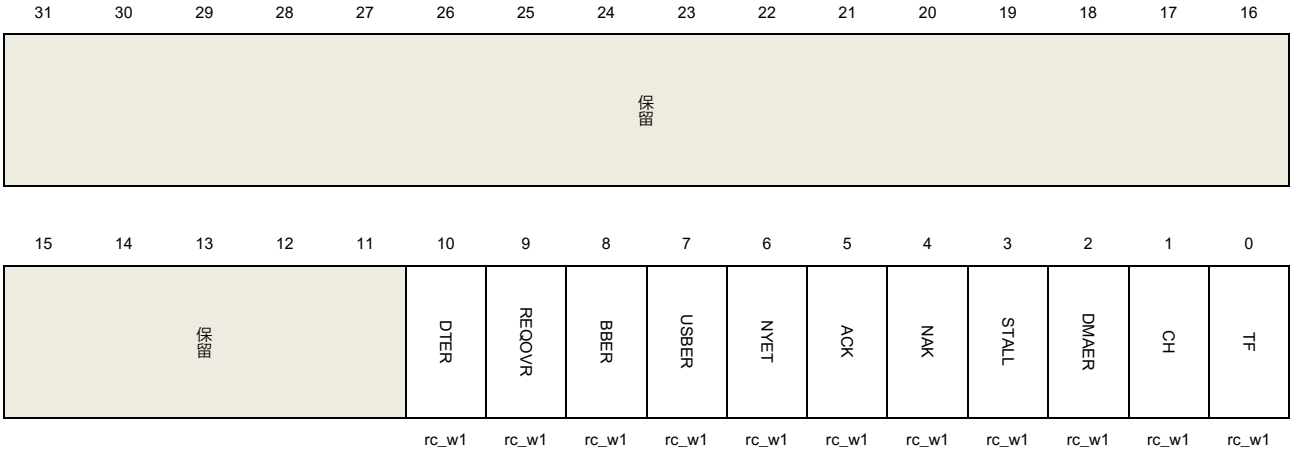
| | | |
|---|---|---|
| 16 | 保留 | 必须保持复位值。 |
| 15 | EPDIR | 端点方向<br>与该通道通信的端点的传输方向<br>0：OUT<br>1：IN |
| 14:11 | EPNUM[3:0] | 端点号<br>与该通道通信的端点号 |
| 10:0 | MPL[10:0] | 最大包长<br>目标端点的最大包长 |

### 主机通道 x 中断标志寄存器(USBHS_HCHxINTF)(x = 0..11， 其中 x 为主机通道号)

地址偏移：0x0508 + (主机通道号 × 0x20)
复位值：0x0000 0000

该寄存器包含一个通道的状态和事件，当软件获取一个通道中断时，软件需要为相应通道读取该寄存器以获取产生中断的中断源。该寄存器中的标志位均由硬件置位，并且写1清除。

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:11 | 保留 | 必须保持复位值。 |
| 10 | DTER | 数据切换错误<br>IN事务获取一个数据包，但是该包的PID和USBHS_HCHxLEN寄存器中的DPID[1:0]控制位不匹配。 |
| 9 | REQOVR | 请求队列上溢<br>当软件启动新的传输时，请求队列上溢。 |
| 8 | BBER | 串扰错误<br>USB总线上发生一个串扰事件。产生串扰事件的典型原因是端点发送了一个数据包，但是数据包长度超过了端点的最大包长。 |

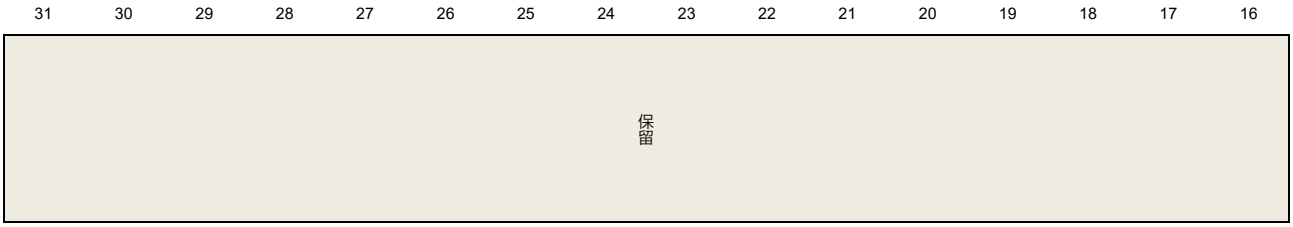
| | | |
|---|---|---|
| 7 | USBER | USB总线错误<br>当在接收一个数据包的过程中，发生以下事件时，将置位USB总线错误标志位：<br>– 接收包有一个错误的CRC域<br>– 在USB总线上检测到填充错误<br>– 当等待一个响应包时，超时 |
| 6 | NYET | NYET<br>接收到一个NYET响应包(在高速模式下) |
| 5 | ACK | ACK<br>接收或者发送一个ACK响应包 |
| 4 | NAK | NAK<br>接收到一个NAK响应包 |
| 3 | STALL | STALL<br>接收到一个STALL响应包 |
| 2 | DMAER | DMA 错误<br>当DMA尝试为当前通道获取或写如包数据时，产生一个错误 |
| 1 | CH | 通道中止<br>当DMA未被使能时：<br>通道被当前请求禁用。<br>当DMA使能时：<br>通道被DMA禁用，可能由于该通道的所有事务成功完成或者发生一个USB错误。 |
| 0 | TF | 发送完成<br>该通道所有的事务成功完成并且无错误发生。<br>对于IN通道，在USBHS_HCHxLEN寄存器的PCNT位减到0后，该标志位被置位。<br>对于OUT通道，当软件从RxFIFO中读取和取出一个TF状态条目时，该标志位被置位。 |

### 主机通道 x 中断使能寄存器(USBHS_HCHxINTEN)(x = 0..11，其中 x 为主机通道号)

地址偏移：0x050C + (主机通道号 × 0x20)
复位值：0x0000 0000

该寄存器包含USBHS_HCHxINTF寄存器内中断标志位的中断使能位。如果该寄存器的某位被软件置位，USBHS_HCHxINTF寄存器内的相应位能够触发一个通道中断。该寄存器内的位可由软件置位和清除。

该寄存器只能按字(32位)访问



| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 保留 | | | | | DTERIE | REQOVRIE | BBERIE | USBERIE | NYETIE | ACKIE | NAKIE | STALLIE | DMAERIE | CHIE | TFIE |
| | | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

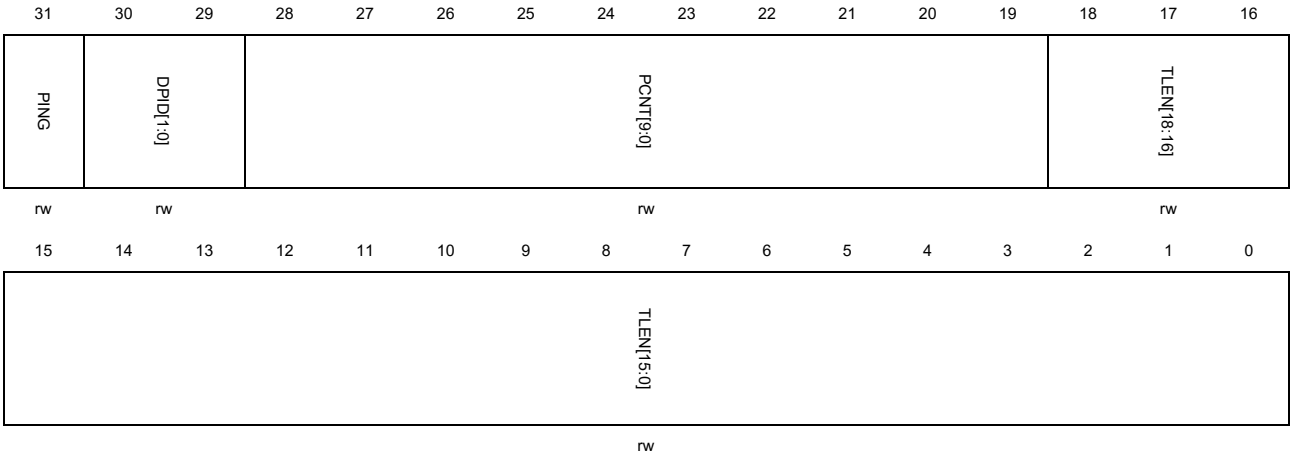
| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:11 | 保留 | 必须保持复位值。 |
| 10 | DTERIE | 数据切换错误中断使能<br>0：禁用数据切换错误中断<br>1：使能数据切换错误中断 |
| 9 | REQOVRIE | 请求队列上溢中断使能<br>0：禁用帧请求队列上溢中断<br>1：使能帧请求队列上溢中断 |
| 8 | BBERIE | 串扰错误中断使能<br>0：禁用串扰错误中断<br>1：使能串扰错误中断 |
| 7 | USBERIE | USB总线错误中断使能<br>0：禁用USB总线错误中断<br>1：使能USB总线错误中断 |
| 6 | NYETIE | NYET中断使能<br>0：禁用NYET中断<br>1：使能NYET中断 |
| 5 | ACKIE | ACK中断使能<br>0：禁用ACK中断<br>1：使能ACK中断 |
| 4 | NAKIE | NAK中断使能<br>0：禁用NAK中断<br>1：使能NAK中断 |
| 3 | STALLIE | STALL中断使能<br>0：禁用STALL中断<br>1：使能STALL中断 |
| 2 | DMAERIE | DMA 错误中断使能<br>0：禁用DMA错误中断<br>1：使能DMA错误中断 |
| 1 | CHIE | 通道中止中断使能<br>0：禁用通道中止中断<br>1：使能通道中止中断 |

| | | |
|---|---|---|
| 0 | TFIE | 传输完成中断使能<br>0：禁用传输完成中断<br>1：使能传输完成中断 |

### 主机通道 x 长度寄存器(USBHS_HCHxLEN)(x = 0..11， 其中 x 为主机通道号)

地址偏移：0x0510 + (主机通道号 × 0x20)
复位值：0x0000 0000

该寄存器只能按字(32位)访问



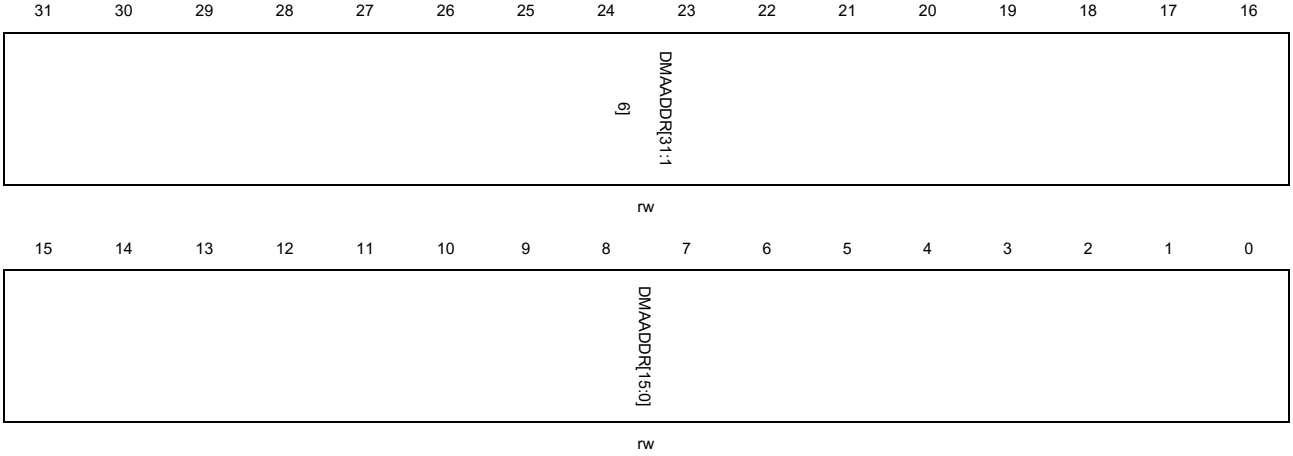

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | PING | PING令牌请求<br>对于OUT传输，如果软件置位该控制位，USBHS会执行PING协议。当OUT事务接收到一个NAK或NYET握手包时，USBHS会自动置位该控制位。不要为IN传输置位该控制位。 |
| 30:29 | DPID[1:0] | 数据PID<br>软件应该在传输起始之前写该段位域。对于OUT传输，该位域包含第一个传输包的数据PID。对于IN传输，该位域包含第一个接收包的数据PID，并且如果数据PID不匹配的话，将会触发DTER标志位。在传输开始之后，USBHS遵循USB协议自动改变和切换该位域。<br>00：DATA0<br>01：DATA2<br>10：DATA1<br>11：MDATA(非控制)/SETUP(控制) |
| 28:19 | PCNT[9:0] | 包计数<br>在一个传输中希望发送(OUT)或接收(IN)的数据包个数。<br>软件应该在通道使能之前写该位域。在传输启动之后，该位域在USBHS正确传输每个数据包后，自动减少。 |
| 18:0 | TLEN[18:0] | 传输长度<br>一次传输的总数据字节数。<br>对于OUT传输，该位域为OUT传输中期望发送的所有数据包总数据字节数。软件应 |

该在通道使能之前写该位域。当软件或DMA正确向通道的数据FIFO中写入一个包时，该位域以包中字节大小进行减少。
对于IN传输，每次软件或DMA从RxFIFO中读取一个包后，该位域也以包中字节大小进行减少。

#### 主机通道 x DMA 地址寄存器(USBHS_HCHxDMAADDR)(x = 0..11， 其中 x 为主机通道号)

地址偏移：0x0514 + (主机通道号 × 0x20)
复位值：0x0000 0000

该寄存器只能按字(32位)访问




| 位/位域 | 名称 | 描述 |
| --- | --- | --- |
| 31:0 | DMAADDR[31:0] | DMA地址<br>该位域定义了端点DMA地址。DMA使用该地址来为该通道取出或写入包数据。 |

### 29.7.3. 设备控制和状态寄存器

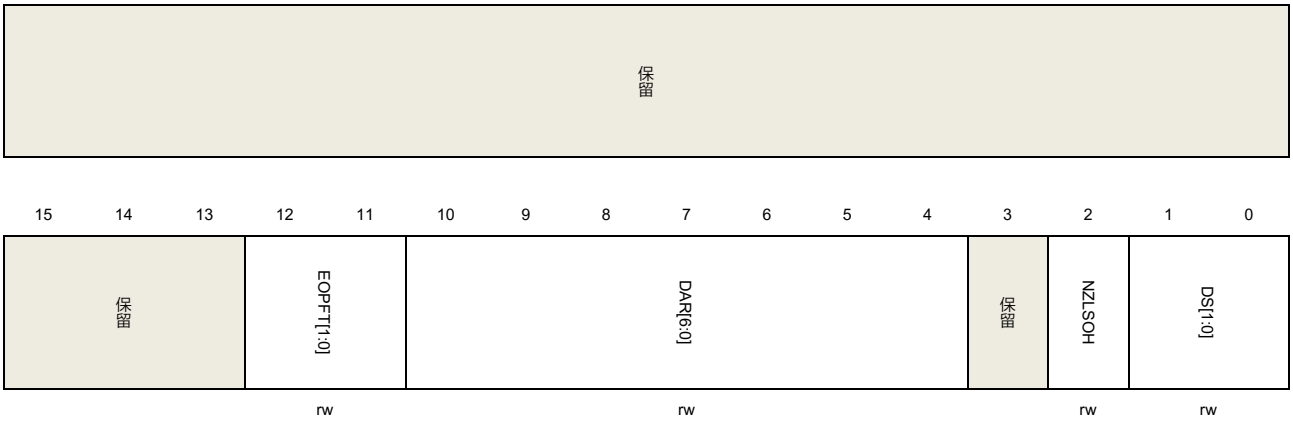
#### 设备配置寄存器(USBHS_DCFG)

地址偏移：0x0800
复位值：0x0000 0000

在上电、枚举或执行某些控制命令后，该寄存器配置内核为设备模式。在设备初始化后，不可以改变该寄存器值。

该寄存器采用字(32位)访问

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

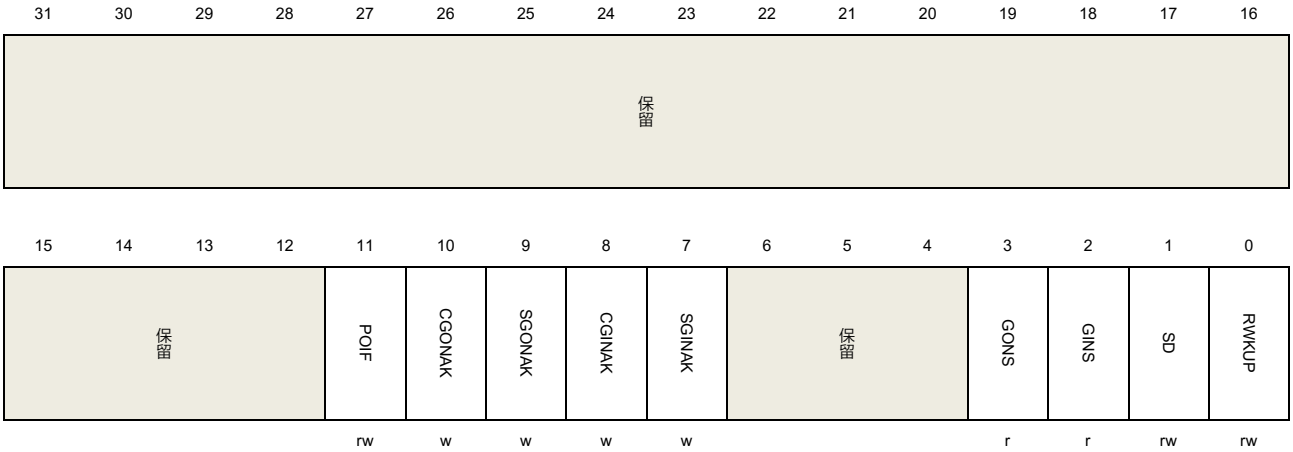



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:13 | 保留 | 必须保持复位值。 |
| 12:11 | EOPFT[1:0] | 周期性帧尾时间<br>该域定义周期性帧时间的帧尾标志触发的时间点<br>00：80%的帧时间<br>01：85%的帧时间<br>10：90%的帧时间<br>11：95%的帧时间 |
| 10:4 | DAR[6:0] | 设备地址<br>该位定义USB设备地址，USBHS采用该位匹配接收的设备令牌地址域，在接收到来自主机的设置地址的命令后，软件设置该域 |
| 3 | 保留 | 必须保持复位值。 |
| 2 | NZLSOH | 非零长度OUT状态阶段握手<br>在控制传输的OUT状态阶段，当USB设备接收到一个非零长度数据包时，该域控制控制USBHS是接收该包，还是用STALL握手信号拒绝该包。<br>0：把该包视为正常包，根据设备OUT端点控制寄存器的NAKS和STALL位，回复握手相应握手包<br>1：发送STALL握手，不保存接收到的OUT数据包 |
| 1:0 | DS[1:0] | 设备速度<br>该域控制设备连入主机后的设备速度<br>00：高速(外部ULPI PHY模式)<br>01：全速<br>其他：保留 |

### 设备控制寄存器(USBHS_DCTL)

地址偏移：0x0804
复位值：0x0000 0000

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11 | POIF | 上电初始化完成<br>软件通过设置该位，通知USBHS寄存器在从掉电模式下唤醒，然后完成初始化。 |
| 10 | CGONAK | 清零全局OUT NAK<br>软件设置该位从而清零该寄存器的GONS位 |
| 9 | SGONAK | 设置全局OUT NAK<br>软件设置该位从而实现该寄存器的位GONS置位。<br>当GONS位为零，设置该位会引起USBHS_GINTF寄存器的GONAK标志触发，软件应该在再写该位前清除GONAK标志。 |
| 8 | CGINAK | 清零全局IN NAK<br>软件设置该位从而清零该寄存器的GINS位 |
| 7 | SGINAK | 设置全局IN NAK<br>软件设置该位从而实现该寄存器的位GINS置位<br>当GINS位为零，设置该位会引起USBHS_GINTF寄存器的GINAK标志触发，软件应该在再写该位前清除GINAK标志。 |
| 6:4 | 保留 | 必须保持复位值。 |
| 3 | GONS | 全局OUT NAK状态<br>0：USBHS回复OUT事务的握手信号以及是否保存OUT数据包由Rx FIFO状态、端点的NAKS、STALL位确定。<br>1：USBHS回复OUT事务NAK握手信号，不保存接收的OUT数据包。 |
| 2 | GINS | 全局IN NAK状态<br>0：USBHS回复IN事务的握手信号由Tx FIFO状态、端点的NAKS、STALL位确定。<br>1：USBHS通常回复IN事务NAK握手信号 |
| 1 | SD | 软断开<br>软件可实现USB总线上的软断开，在置1该位后，如果当前是高速模式，USBHS先退回到全速模式，然后在关掉DP线上的上拉电阻，从而引起主机检测设备的断开。<br>0：没有软断开生成 |

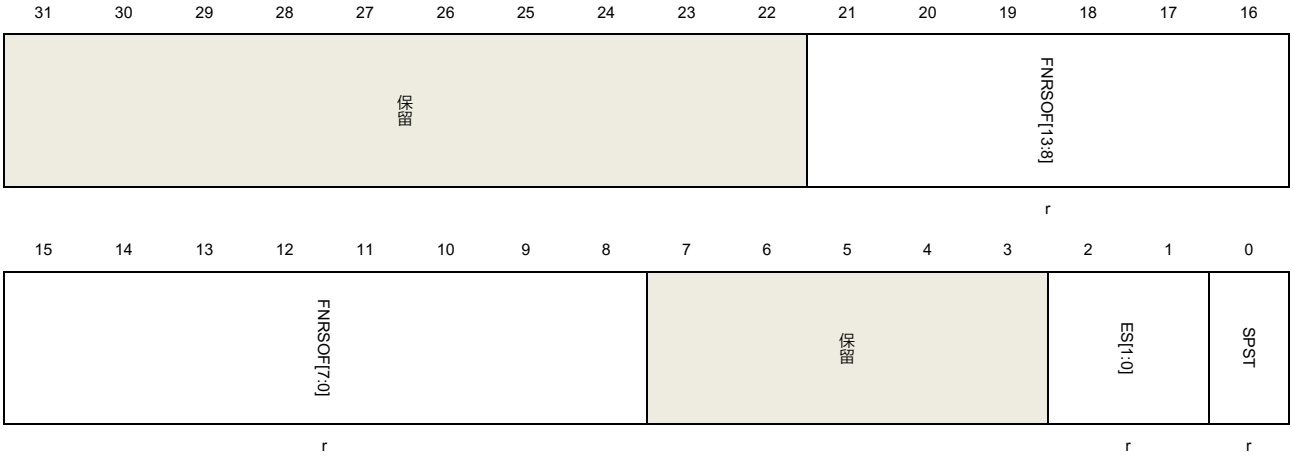
| | | |
|---|---|---|
| | | 1：生成软断开 |
| 0 | RWKUP | 远程唤醒<br>在挂起状态，软件可通过该位来生成一个远程唤醒信号来通知主机恢复USB总线<br>0：没有远程唤醒信号生成<br>1：生成远程唤醒信号 |

### 设备状态寄存器(USBHS_DSTAT)

地址偏移：0x0808
复位值：0x0000 0000

该寄存器包含设备模式下的 USBHS 的状态和信息。

该寄存器采用字(32位)访问



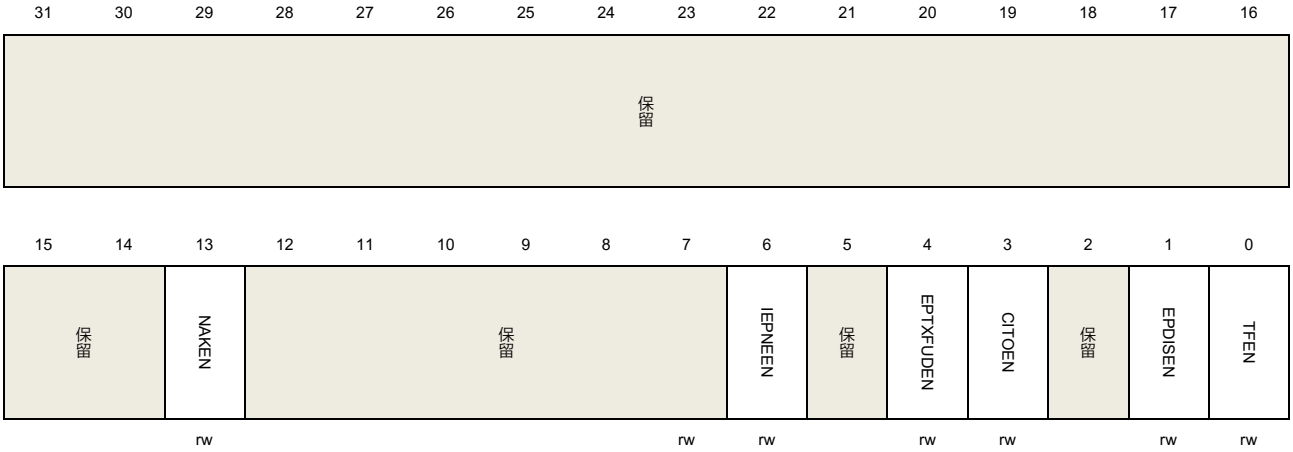

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:22 | 保留 | 必须保持复位值。 |
| 21:8 | FNRSOF[13:0] | 所接收的SOF帧编号<br>USBHS会在接收到一个SOF令牌后更新该域。 |
| 7:3 | 保留 | 必须保持复位值。 |
| 2:1 | ES[1:0] | 枚举速度<br>该域指示所枚举的设备速度，在寄存器USBHS_GINTF的ENUMF标志触发后，软件可以读取该域。<br>00：高速<br>01：全速<br>其他：保留 |
| 0 | SPST | 挂起状态<br>该位指示设备是否处于挂起状态。<br>0：设备不在挂起状态<br>1：设备在挂起状态 |

### 设备 IN 端点通用中断使能寄存器(USBHS_DIEPINTEN)

地址偏移：0x810
复位值：0x0000 0000

该寄存器包含寄存器 USBHS_DIEPxINTF 中的标志的中断使能位，如果软件置 1 某位，其在寄存器 USBHS_DIEPxINTF 中对应的位可以触发一个寄存器 USBHS_DAEPINT 端点中断。该位可以通过软件置位和清零。

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值。 |
| 13 | NAKEN | 发送NAK握手中断使能位<br>0：除能发送NAK握手中断<br>1：使能发送NAK握手中断 |
| 12:7 | 保留 | 必须保持复位值。 |
| 6 | IEPNEEN | IN端点NAK有效中断使能位<br>0：除能IN端点NAK有效中断<br>1：使能IN端点NAK有效中断 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | EPTXFUDEN | 端点Tx FIFO下溢中断使能位<br>0：除能端点Tx FIFO下溢中断<br>1：使能端点Tx FIFO下溢中断 |
| 3 | CITOEN | 控制IN事务超时中断使能位<br>0：除能控制IN事务超时中断<br>1：使能控制IN事务超时中断 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | EPDISEN | 端点除能中断使能位<br>0：除能端点除能中断 |

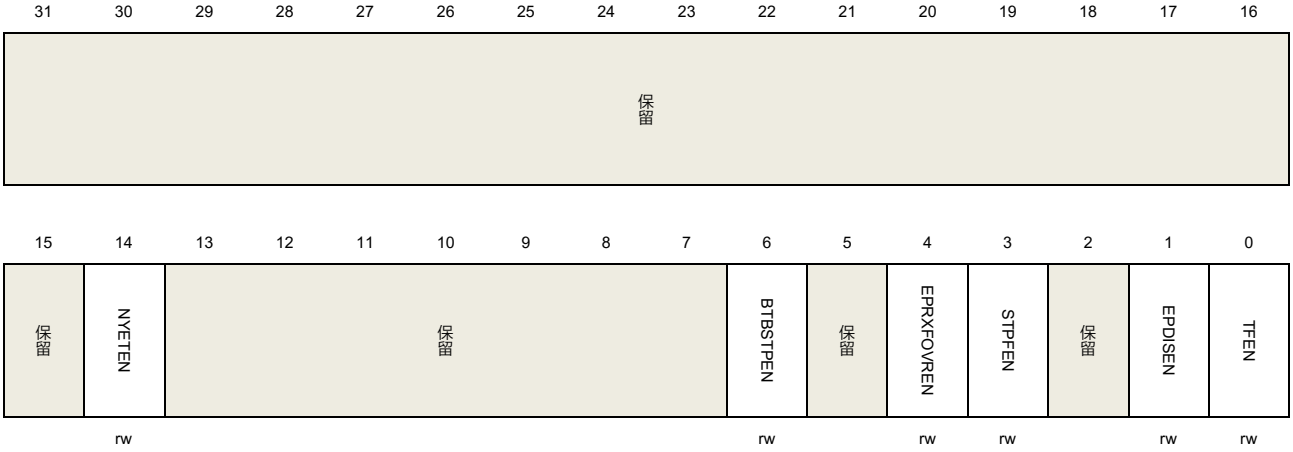
| | | |
|---|---|---|
| | | 1：使能端点除能中断 |
| 0 | TFEN | 传输完成中断使能位<br>0：除能传输完成中断<br>1：使能传输完成中断 |

### 设备 OUT 端点通用中断使能寄存器(USBHS_DOEPINTEN)

地址偏移：0x0814
复位值：0x0000 0000

该寄存器包含寄存器 USBHS_DOEPxINTF 中的标志的中断使能位，如果软件置 1 某位，其在寄存器 USBHS_DOEPxINTF 中对应的位可以触发一个寄存器 USBHS_DAEPINT 端点中断。该位可以通过软件置位和清零。

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:15 | 保留 | 必须保持复位值。 |
| 14 | NYETEN | 发送NYET握手中断使能位<br>0：除能中断<br>1：使能中断 |
| 13:7 | 保留 | 必须保持复位值。 |
| 6 | BTBSTPEN | 连续SETUP包中断使能位(仅适用于控制OUT端点)<br>0：除能中断<br>1：使能中断 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | EPRXFOVREN | 端点Rx FIFO上溢中断使能位<br>0：除能中断<br>1：使能中断 |
| 3 | STPFEN | SETUP阶段完成中断使能位(仅适用于控制OUT端点) |

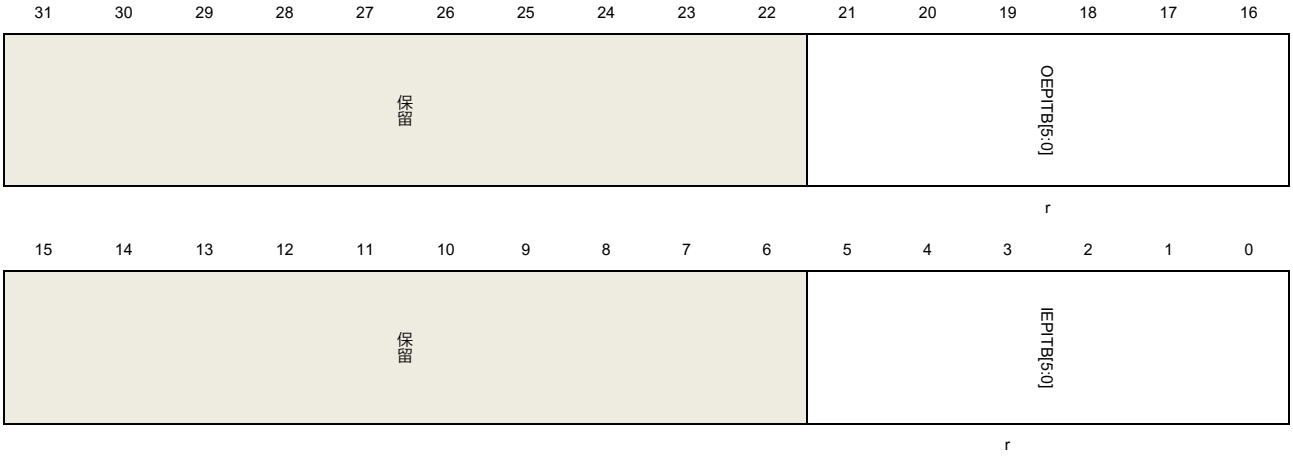
| | | |
|---|---|---|
| | | 0：除能中断<br>1：使能中断 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | EPDISEN | 端点除能中断使能位<br>0：除能中断<br>1：使能中断 |
| 0 | TFEN | 传输完成中断使能位<br>0：除能中断<br>1：使能中断 |

### 设备端点中断寄存器(USBHS_DAEPINT)

地址偏移：0x0818
复位值：0x0000 0000

当一个端点的中断被触发，USBHS 置 1 该寄存器的相应位，软件可通过该寄存器知道在本次中断中的端点号。

该寄存器采用字(32位)访问



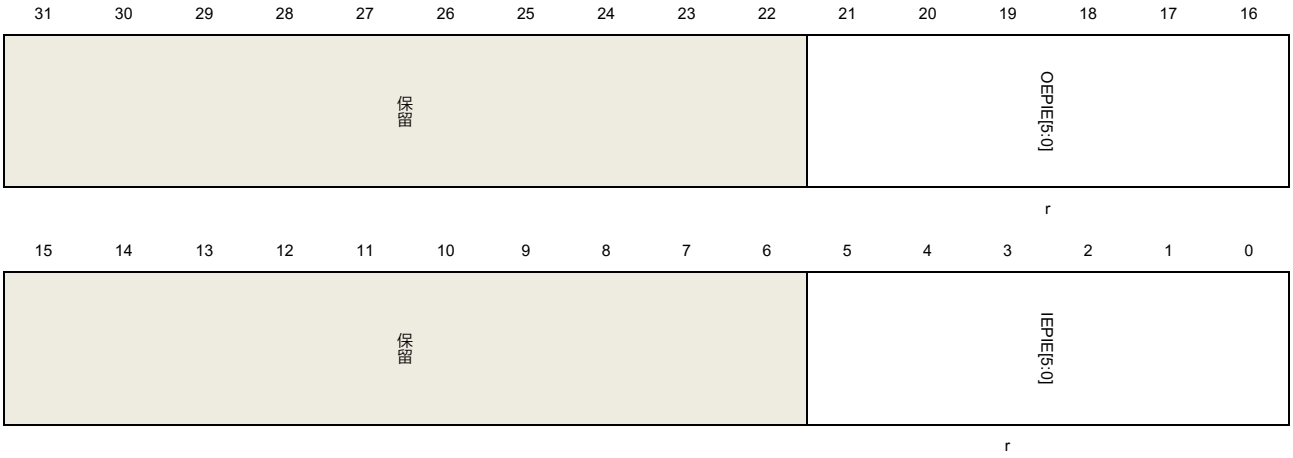

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:22 | 保留 | 必须保持复位值。 |
| 21:16 | OEPITB[5:0] | 设备OUT端点中断位<br>每个位代表一个OUT端点：Bit16代表OUT端点0，Bit21代表OUT端点5 |
| 15:6 | 保留 | 必须保持复位值。 |
| 5:0 | IEPITB[5:0] | 设备IN端点中断位<br>每个位代表一个IN端点：Bit0代表IN端点0，Bit5代表IN端点5 |

### 设备端点中断使能寄存器(USBHS_DAEPINTEN)

地址偏移：0x081C
复位值：0x0000 0000

该寄存器可通过软件使能或除能端点的中断，只有当端点在该寄存器中相应位被置 1 才能触发寄存器 USBHS_GINTF 的端点中断标志 OEPIF 或 IEPIF。

该寄存器采用字(32位)访问



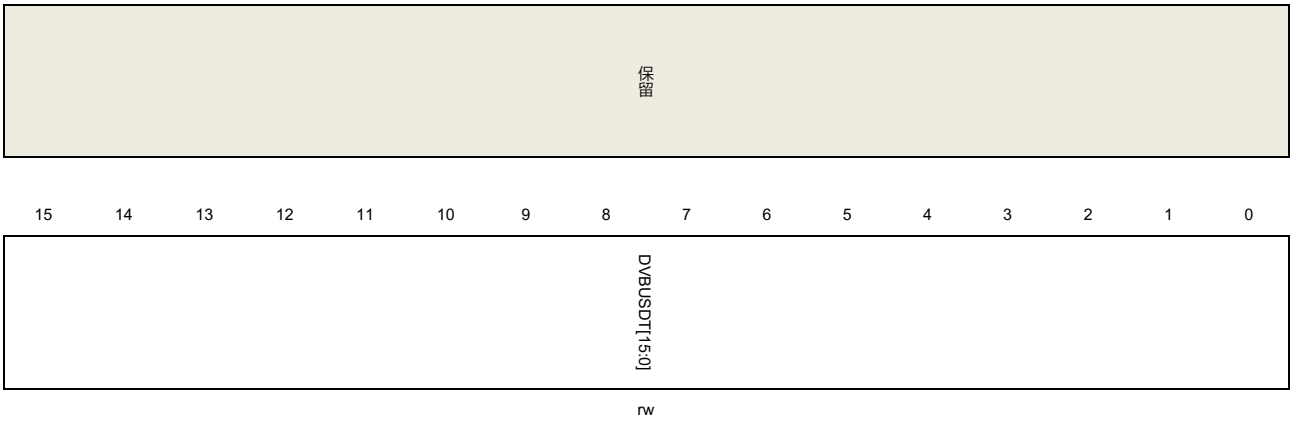

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:22 | 保留 | 必须保持复位值。 |
| 21:16 | OEPIE[5:0] | OUT端点中断使能位<br>0：除能OUT端点n中断<br>1：使能OUT端点n中断<br>每个位代表一个OUT端点：Bit16对应OUT端点0，Bit21对应OUT端点5 |
| 15:6 | 保留 | 必须保持复位值。 |
| 5:0 | IEPIE[5:0] | IN端点中断使能位<br>0：除能IN端点n中断<br>1：使能IN端点n中断<br>每个位代表一个IN端点：Bit0对应IN端点0，Bit5对应IN端点5 |

### 设备 VBUS 放电时间寄存器(USBHS_DVBUSDT)

地址偏移：0x0828
复位值：0x0000 17D7

该寄存器采用字(32位)访问



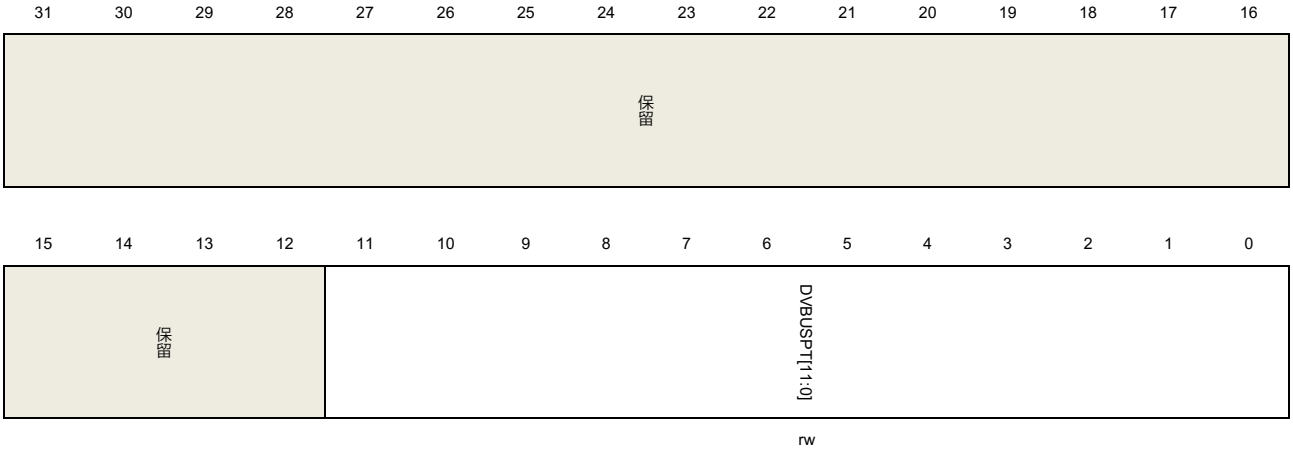



| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | DVBUSDT[15:0] | 设备$V_{BUS}$放电时间<br>在SRP协议中，在$V_{BUS}$脉冲产生后，有一个放电过程，该域定义了$V_{BUS}$的放电时间，真正的放电时间是1024*DVBUSDT[15:0] *$T_{USBCLOCK}$，$T_{USBCLOCK}$是USB时钟周期时间。 |

### 设备 VBUS 脉冲时间寄存器(USBHS_DVBUSPT)

地址偏移：0x082C
复位值：0x0000 05B8

该寄存器采用字(32位)访问



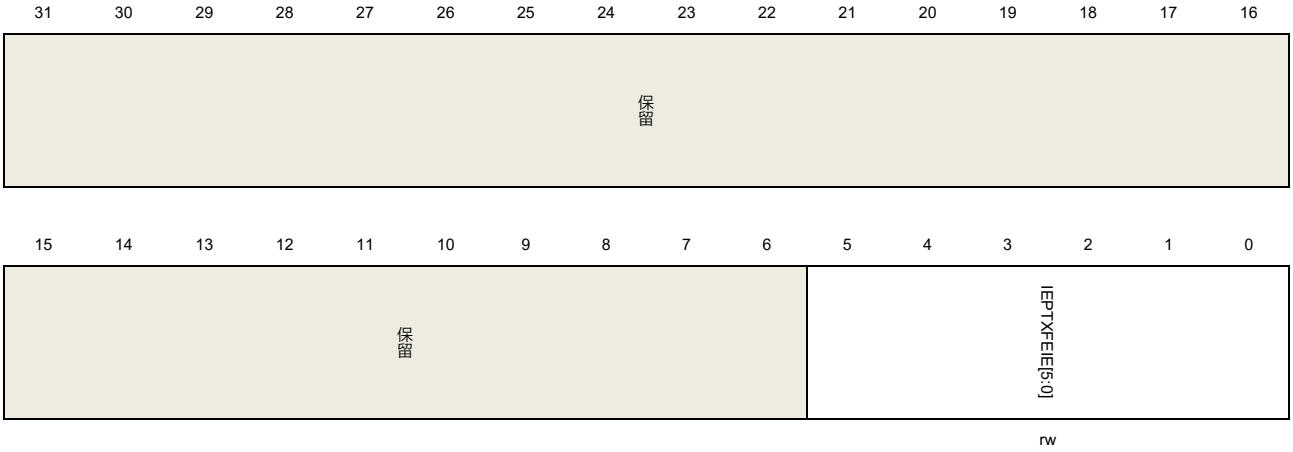

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:12 | 保留 | 必须保持复位值。 |
| 11:0 | DVBUSPT[11:0] | 设备$V_{BUS}$脉冲时间<br>该域定义$V_{BUS}$的脉冲时间，真正的充电时间是1024*DVBUSPT[15:0] *$T_{USBCLOCK}$，$T_{USBCLOCK}$是USB时钟周期时间 |

### 设备 IN 端点 FIFO 空中断使能寄存器(USBHS_DIEPFEINTEN)

地址偏移：0x0834
复位值：0x0000 0000

该寄存器包含 IN 端点 Tx FIFO 空中断的使能位

寄存器采用字(32位)访问



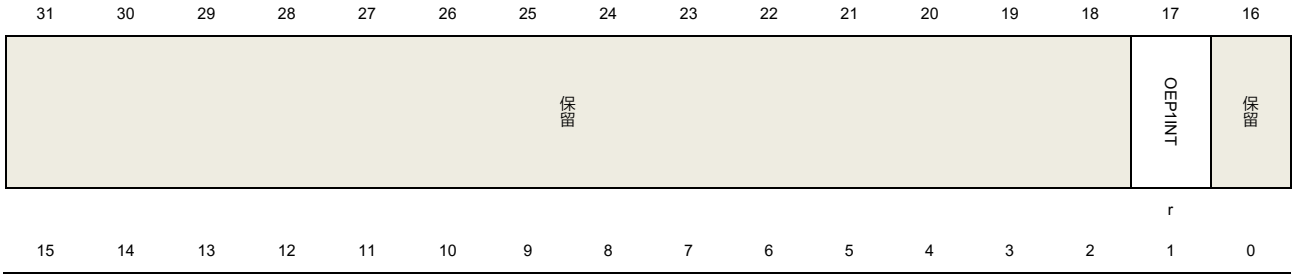

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:6 | 保留 | 必须保持复位值。 |
| 5:0 | IEPTXFEIE[5:0] | IN端点Tx FIFO空中断的使能位<br>该域控制着USBHS_DIEPxINTF寄存器的TXFE位能否生成一个寄存器USBHS_DAEPINT的端点中断位<br>Bit0对应IN端点0，Bit5对应IN端点5<br>0：除能FIFO空中断<br>1：使能FIFO空中断 |

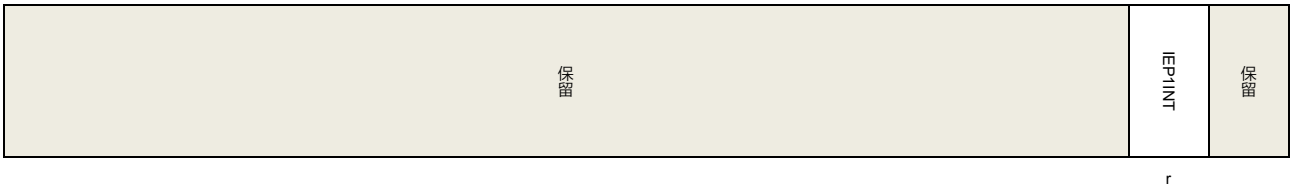
### 设备端点 1 中断寄存器(USBHS_DEP1INT)

地址偏移：0x0838
复位值：0x0000 0000

当端点 1 OUT 或 IN 被触发，USBHS 置位该寄存器的相应位，软件可通过读取该位知道端点 1 的 IN 或 OUT 被触发。

该寄存器采用字(32位)访问





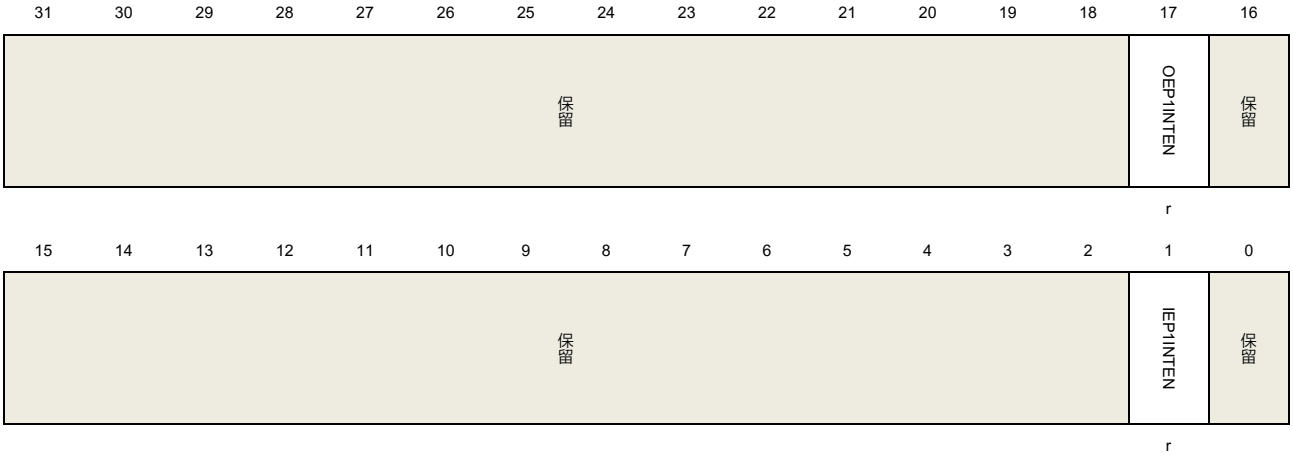

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:18 | 保留 | 必须保持复位值。 |
| 17 | OEP1INT | OUT端点1中断 |
| 16:2 | 保留 | 必须保持复位值。 |
| 1 | IEP1INT | IN端点1中断 |
| 0 | 保留 | 必须保持复位值。 |

### 设备端点 1 中断使能寄存器(USBHS_DEP1INTEN)

地址偏移：0x083C
复位值：0x0000 0000

该寄存器可以用软件使能或除能端点1的中断，只有该寄存器中的相应位被置位才可以引起端点1 IN或者OUT的中断。

该寄存器采用字(32位)访问



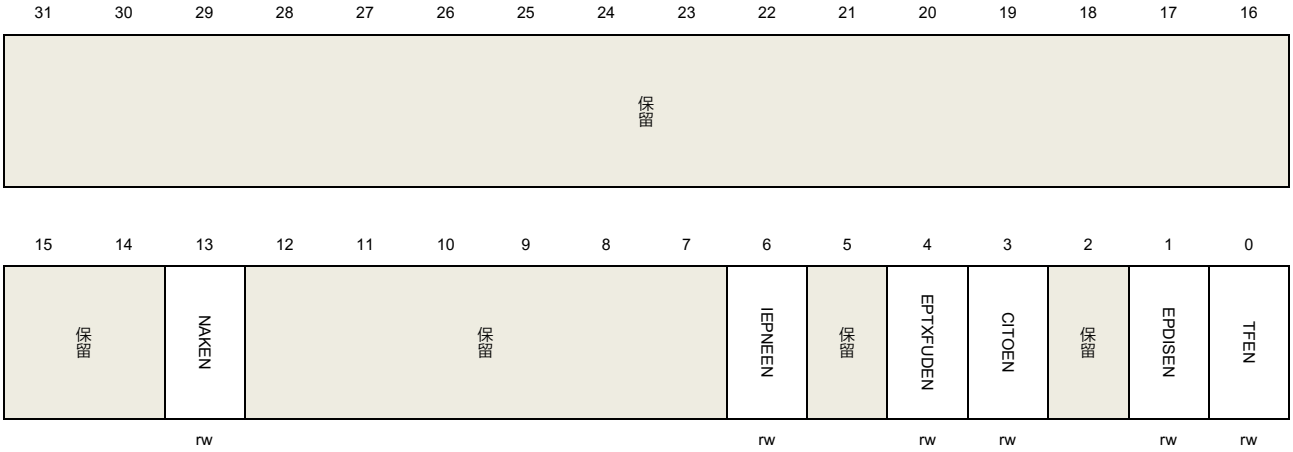

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:18 | 保留 | 必须保持复位值。 |
| 17 | OEP1INTEN | OUT端点1中断使能 |
| 16:2 | 保留 | 必须保持复位值。 |
| 1 | IEP1INTEN | IN端点1中断使能 |
| 0 | 保留 | 必须保持复位值。 |

### 设备 IN 端点 1 中断使能寄存器(USBHS_DIEP1INTEN)

地址偏移：0x844
复位值：0x0000 0000

该寄存器包含对应寄存器 USBHS_DIEP1INTF 标志的中断使能位，如果软件置位该寄存器的某一位,其在寄存器 USBHS_DIEP1INTF 中对应的位会触发一个寄存器 USBHS_DEP1INT 中的端点中断，该寄存器的位可以通过软件置位或清零。

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值。 |
| 13 | NAKEN | 发送NAK握手中断使能位<br>0：除能中断<br>1：使能中断 |
| 12:7 | 保留 | 必须保持复位值。 |
| 6 | IEPNEEN | IN端点NAK有效中断使能位<br>0：除能中断<br>1：使能中断 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | EPTXFUD | 端点Tx FIFO下溢中断使能位<br>0：除能中断<br>1：使能中断 |
| 3 | CITOEN | 控制传输IN超时中断使能位<br>0：除能中断<br>1：使能中断 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | EPDISEN | 端点除能中断使能位<br>0：除能中断 |

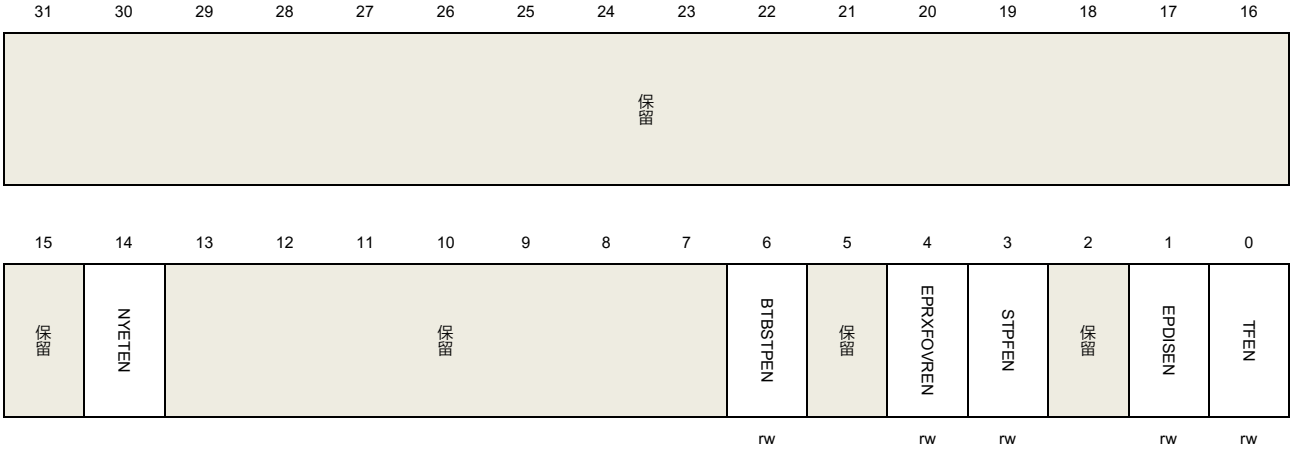
| | | |
|---|---|---|
| | | 1：使能中断 |
| 0 | TFEN | 传输完成中断使能位<br>0：除能中断<br>1：使能中断 |

### 设备 OUT 端点 1 中断使能寄存器(USBHS_DOEP1INTEN)

地址偏移：0x0884
复位值：0x0000 0000

该寄存器包含对应寄存器 USBHS_ DOEP1INTF 标志的中断使能位，如果软件置位该寄存器的某一位，其在寄存器 USBHS_DOEP1INTF 中对应的位会触发一个寄存器 USBHS_DEP1INT 中的端点中断，该寄存器的位可以通过软件置位或清零。

该寄存器采用字(32位)访问



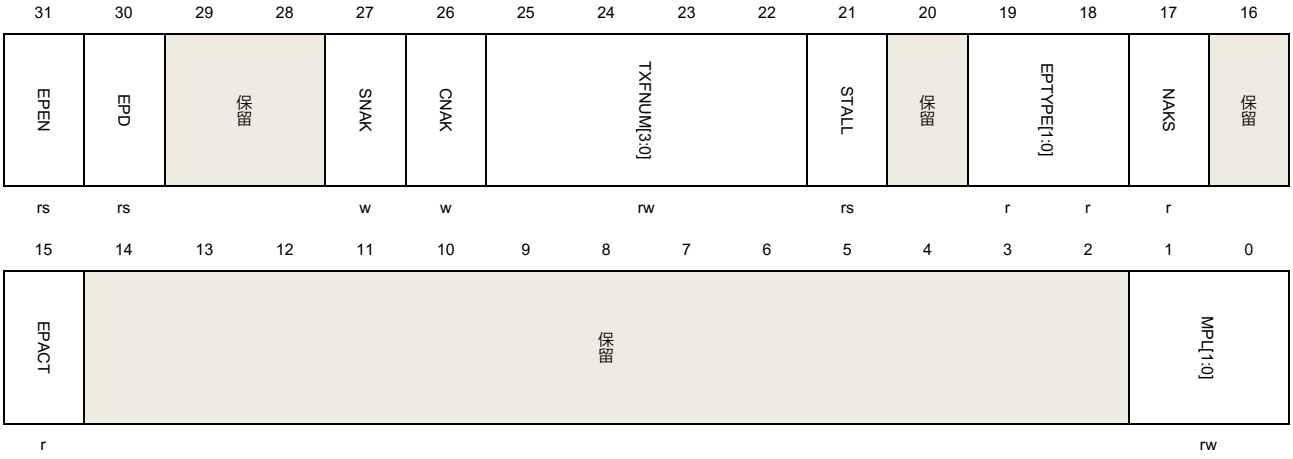

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:15 | 保留 | 必须保持复位值。 |
| 14 | NYETEN | 发送NYET握手中断使能位<br>0：除能中断<br>1：使能中断 |
| 13:7 | 保留 | 必须保持复位值。 |
| 6 | BTBSTPEN | 连续SETUP包中断使能位(仅适用于控制OUT端点)<br>0：除能中断<br>1：使能中断 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | EPRXFOVREN | 端点Rx FIFO上溢中断使能位<br>0：除能中断<br>1：使能中断 |
| 3 | STPFEN | SETUP阶段完成中断使能位(仅适用于控制OUT端点) |

| | | |
|---|---|---|
| | | 0：除能中断<br>1：使能中断 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | EPDISEN | 端点除能中断使能位<br>0：除能中断<br>1：使能中断 |
| 0 | TFEN | 传输完成中断使能位<br>0：除能中断<br>1：使能中断 |

### 设备 IN 端点 0 控制寄存器(USBHS_DIEP0CTL)

地址偏移：0x0900
复位值：0x0000 8000

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | EPEN | 端点使能<br>软件置位、USBHS清零<br>0：端点除能<br>1：端点使能<br>软件应该按照操作指南使能或除能端点 |
| 30 | EPD | 端点除能<br>软件可通过置位该位从而除能端点，软件应该按照操作指南使能或除能端点。 |
| 29:28 | 保留 | 必须保持复位值。 |
| 27 | SNAK | 置位NAK<br>软件置位该位来设置该寄存器的NAKS位 |
| 26 | CNAK | 清零NAK |

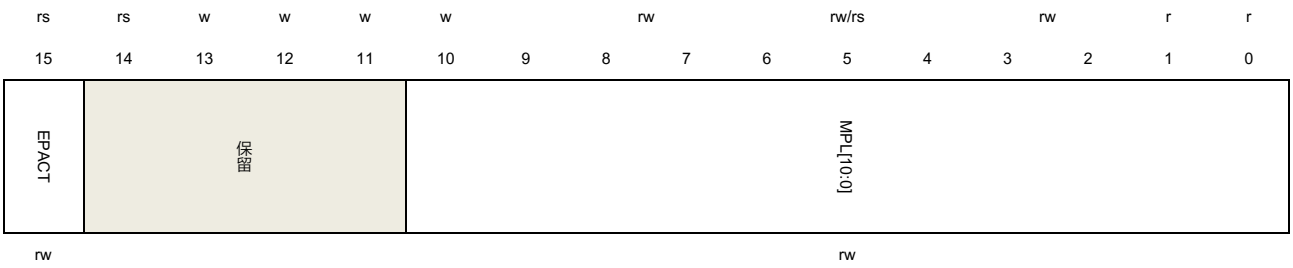
| | | |
|---|---|---|
| | | 软件置位该位来清零该寄存器的NAKS位 |
| 25:22 | TXFNUM[3:0] | Tx FIFO编号<br>定义IN端点0的Tx FIFO编号 |
| 21 | STALL | STALL握手<br>当接收IN令牌时，软件可以通过置1该位发送STALL握手包，对于相应的OUT端点0，在接收SETUP令牌后，USBHS清除此位。该位比该寄存器的NAKS位和寄存器USBHS_DCTL的GINS位优先级要高，如果STALL和NAKS位都被置位，STALL位生效。 |
| 20 | 保留 | 必须保持复位值。 |
| 19:18 | EPTYPE[1:0] | 端点类型<br>该域固定为'00',控制端点。 |
| 17 | NAKS | NAK状态<br>当该寄存器的STALL位和寄存器USBHS_DCTL的位GINS被清零，该位控制USBHS的NAK状态。<br>0：根据端点Tx FIFO的状态，USBHS发送数据或握手包<br>1：USBHS总为IN令牌发送NAK握手包<br>该位是只读位，可以通过该寄存器的位CNAK和位SNAK控制该位 |
| 16 | 保留 | 必须保持复位值。 |
| 15 | EPACT | 端点激活<br>对于端点0来说，该域固定为'1' |
| 14:2 | 保留 | 必须保持复位值。 |
| 1:0 | MPL[1:0] | 最大包长<br>域定义了控制数据包的最大包长，如USB 2.0协议所描述，对控制传输而言，有四种包长度：<br>00：64字节<br>01：32字节<br>10：16字节<br>11：8字节 |

### 设备 IN 端点 x 控制寄存器(USBHS_DIEPxCTL)(x = 1..5， x 是设备端点编号)

地址偏移：0x0900 + (设备端点编号 x 0x20)
复位值：0x0000 0000

该寄存器采用字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EPEN | EPD | SODDFRM/SD1PID | SD0PID/SEVENFRM | SNAK | CNAK | TXFNUM[3:0] | | | | STALL | 保留 | EPTYPE[1:0] | | NAKS | EOFRM/DPID |




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | EPEN | 端点使能<br>软件置位，USBHS清零<br>0：端点除能<br>1：端点使能<br>软件应该按照操作指南使能或除能端点 |
| 30 | EPD | 端点除能<br>软件可通过置位该位从而除能端点，软件应该按照操作指南使能或除能端点。 |
| 29 | SODDFRM | 设置奇数帧(适用于同步IN端点)<br>软件通过置1该位置1该寄存器的EOFRM位 |
| | SD1PID | 设置DATA1 PID(适用于中断和大容量IN端点)<br>软件可通过置1该位置1该寄存器的DPID位 |
| 28 | SEVENFRM | 设置偶数帧(适用于同步IN端点)<br>软件通过置1该位清零该寄存器的EOFRM位 |
| | SD0PID | 设置DATA0 PID(适用于中断和大容量IN端点)<br>软件可通过置1该位清零该寄存器的DPID位 |
| 27 | SNAK | 设置NAK<br>软件置1该位置1该寄存器的NAKS位 |
| 26 | CNAK | 清零NAK<br>软件置1该位清零该寄存器的NAKS位 |
| 25:22 | TXFNUM[3:0] | Tx FIFO编号<br>该位定义了IN端点的Tx FIFO编号 |
| 21 | STALL | STALL握手<br>当接收IN令牌时，软件可以通过置1该位发送STALL握手包。该位比该寄存器的NAKS位和寄存器USBHS_DCTL的GINS位优先级要高，如果STALL和NAKS位都被置位，STALL位生效。<br>对于控制IN端点：<br>当对应的OUT端点接收到SETUP令牌时，只有USBHS可以清零此位，软件不可清除此位。<br>对于中断或大容量IN端点：<br>只有软件可以清零此位。 |

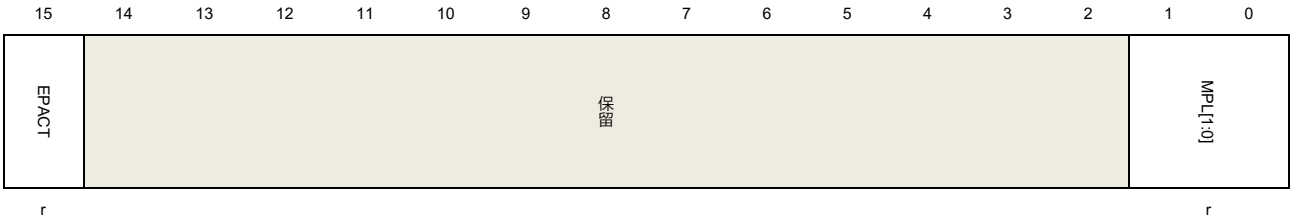
| | | |
|---|---|---|
| 20 | 保留 | 必须保持复位值。 |
| 19:18 | EPTYPE[1:0] | 端点类型<br>该域定义端点的传输类型:<br>00：控制<br>01：同步<br>10：大容量<br>11：中断 |
| 17 | NAKS | NAK状态<br>当该寄存器的STALL位和寄存器USBHS_DCTL的位GINS被清零，该位控制USBHS的NAK状态:<br>0：根据端点Tx FIFO的状态，USBHS发送数据或握手包<br>1：USBHS总为IN令牌发送NAK握手包<br>该位是只读位，可以通过该寄存器的位CNAK和位SNAK控制该位 |
| 16 | EOFRM | 奇偶帧(适用于同步IN端点)<br>对于同步传输，软件通过使用该位控制USBHS只在奇数帧或偶数帧为IN事务发送数据包，如果当前帧号的奇偶性不匹配该位，USBHS回复一个零长度的包:<br>0：只在偶数帧发送数据<br>1：只在奇数帧发送数据 |
| | DPID | 端点数据PID(适用于中断或大容量IN端点)<br>在端点或大容量传输中，有数据PID翻转机制，在传输开始之前，软件通过设定SD0PID来设置此位，按照USB协议中描述的数据PID翻转机制，USBHS在传输过程中保持该位。<br>0：数据包的PID是DATA0<br>1：数据包的PID是DATA1 |
| 15 | EPACT | 端点激活<br>该位控制端点是否激活，当端点没有激活，忽略任何令牌，不做任何回复。 |
| 14:11 | 保留 | 必须保持复位值。 |
| 10:0 | MPL[10:0] | 该域定义最大包长 |

### 设备 OUT 端点 0 控制寄存器(USBHS_DOEP0CTL)

地址偏移：0x0B00
复位值：0x0000 8000

该寄存器采用字(32位)访问

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EPEN | EPD | 保留 | | SNAK | CNAK | 保留 | | | | STALL | SNOOP | EPTYPE[1:0] | | NAKS | 保留 |
| rs | r | | | w | w | | | | | rs | rw | r | | r | |




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | EPEN | 端点使能<br>软件置位，USBHS清零<br>0：端点除能<br>1：端点使能<br>软件应该按照操作指南使能或除能端点。 |
| 30 | EPD | 端点除能<br>对于OUT端点0，该位固定为0 |
| 29:28 | 保留 | 必须保持复位值。 |
| 27 | SNAK | 设置NAK<br>软件置1该位置1该寄存器的NAKS位 |
| 26 | CNAK | 清零NAK<br>软件置1该位清零该寄存器的NAKS位 |
| 25:22 | 保留 | 必须保持复位值。 |
| 21 | STALL | STALL握手<br>在OUT事务中，软件可以通过置1该位发送STALL握手包，对于OUT端点0，在接收SETUP令牌后，USBHS清除此位。该位比该寄存器的NAKS位和寄存器USBHS_DCTL的GINS位优先级要高，即如果STALL和NAKS位都被置位，STALL位生效。 |
| 20 | SNOOP | 调查模式<br>该位控制OUT端点的调查模式，在调查模式中，USBHS不再检查接收数据包的CRC值<br>0：调查模式除能<br>1：调查模式使能 |
| 19:18 | EPTYPE[1:0] | 端点类型<br>对于控制端点，该位固定为"00" |
| 17 | NAKS | NAK状态<br>当该寄存器的STALL位和寄存器USBHS_DCTL的位GINS被清零，该位控制USBHS的NAK状态：<br>0：根据端点Rx FIFO的状态，USBHS发送数据或握手包<br>1：USBHS为OUT事务发送NAK握手包<br>该位是只读位，通过该寄存器的CNAK和SNAK位控制该位 |

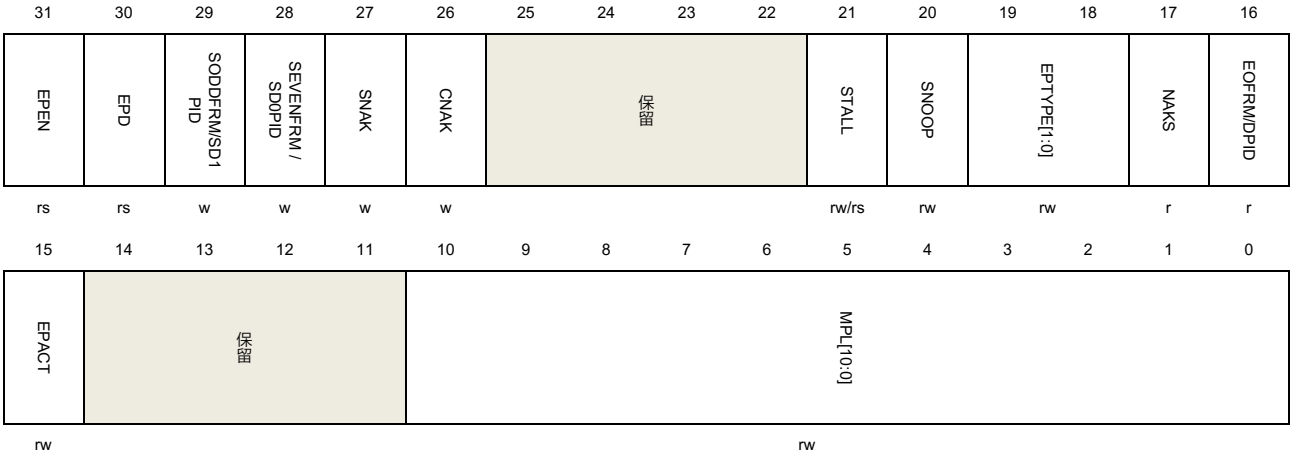
| | | |
|---|---|---|
| 16 | 保留 | 必须保持复位值。 |
| 15 | EPACT | 端点激活<br>对于端点0，该域固定为1 |
| 14:2 | 保留 | 必须保持复位值。 |
| 1:0 | MPL[1:0] | 最大包长<br>该位是只读位，其数值来自于寄存器USBHS_DIEP0CTL的位MPL：<br>00：64字节<br>01：32字节<br>10：16字节<br>11：8字节 |

### 设备 OUT 端点 x 控制寄存器(USBHS_DOEPxCTL)(x= 1..5， x 是设备端点编号)

地址偏移：0x0B00 + (设备端点编号 x 0x20)
复位值：0x0000 0000

软件用该寄存器控制 OUT 端点 0 以外的每个逻辑 OUT 端点

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | EPEN | 端点使能<br>软件置位，USBHS清零<br>0：端点除能<br>1：端点使能<br>软件应该按照操作指南使能或除能端点。 |
| 30 | EPD | 端点除能<br>软件通过置1该位除能端点，软件应该按照操作指南使能或除能端点。 |
| 29 | SODDFRM | 设置奇数帧(适用于同步OUT端点)<br>该位只针对同步OUT端点有效 |

| | | |
|---|---|---|
| | | 软件置1该位来置位该寄存器的EOFRM位 |
| | SD1PID | 设置DATA1 PID(适用于中断和大容量OUT端点)<br>软件置1该位来置位该寄存器的DPID位 |
| 28 | SEVENFRM | 设置偶数帧(适用于同步OUT端点)<br>软件置1该位来清零该寄存器的EOFRM位 |
| | SD0PID | 设置DATA0 PID(适用于中断和大容量OUT端点)<br>软件置1该位来清零该寄存器的DPID位 |
| 27 | SNAK | 设置NAK<br>软件置1该位从而置1该寄存器的NAKS位 |
| 26 | CNAK | 清零NAK<br>软件置1该位从而清零该寄存器的NAKS位 |
| 25:22 | 保留 | 必须保持复位值。 |
| 21 | STALL | STALL握手<br>在OUT事务中，软件可以通过置1该位发送STALL握手包。该位比该寄存器的NAKS位和寄存器USBHS_DCTL的GINS位优先级要高，如果STALL和NAKS位都被置位，STALL位生效。<br>对于控制OUT端点：<br>当OUT端点接收SETUP令牌时，只有USBHS可以清零该位，软件不可清零此位。<br>对于中断或大容量OUT端点：<br>只有软件可以清零该位 |
| 20 | SNOOP | 调查模式<br>该位控制OUT端点的调查模式，在调查模式中，USBHS不再检查接收数据包的CRC值<br>0：调查模式除能<br>1：调查模式使能 |
| 19:18 | EPTYPE[1:0] | 端点类型<br>该域定义端点的传输类型<br>00：控制<br>01：同步<br>10：大容量<br>11：中断 |
| 17 | NAKS | NAK状态<br>当该寄存器的STALL位和寄存器USBHS_DCTL的位GONS被清零，该位控制USBHS的NAK状态：<br>0：根据端点的Rx FIFO的状态，发送握手包<br>1：USBHS发送NAK握手给OUT事务<br>该位是只读位，通过该寄存器的CNAK和SNAK位控制该位 |
| 16 | EOFRM | 奇偶帧(适用于同步OUT端点)<br>对于同步传输，软件通过使用该位控制USBHS只在奇数帧或偶数帧发送数据包给 |

| | | |
|---|---|---|
| | | OUT事务，如果当前帧号的奇偶性不匹配该位，USBHS不保存数据包<br>0：只在偶数帧发送数据<br>1：只在奇数帧发送数据 |
| | DPID | 端点数据PID(适用于中断或大容量端点)<br>在端点或大容量传输中，有数据PID翻转机制，在传输开始之前，软件通过设定SD0PID来设置此位，按照USB协议中描述的数据PID翻转机制，USBHS在传输过程中保持该位。<br>0：数据包PID是DATA0<br>1：数据包PID是DATA1 |
| 15 | EPACT | 端点激活<br>位控制端点是否激活，当端点没有激活，忽略任何令牌，不做任何回复 |
| 14:11 | 保留 | 必须保留为 复位值 |
| 10:0 | MPL[10:0] | 该位定义最大包长 |

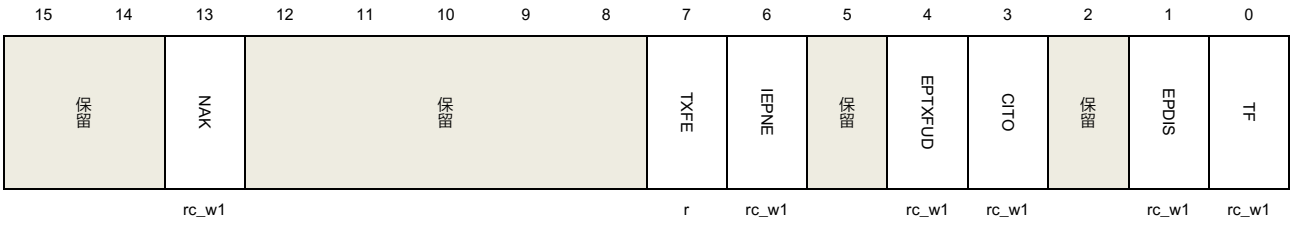
### 设备 IN 端点 x 中断标志寄存器(USBHS_DIEPxINTF)(x = 0..5， x 是设备端点编号)

地址偏移：0x0908 + (设备端点编号 x 0x20)
复位值：0x0000 0080

该寄存器包含 IN 端点的状态和事件，当获得一个 IN 端点的中断时，应该读取该端点的中断标志寄存器，从而获知中断源。该寄存器的标志位通常硬件置位，除了 TXFE 位，各位写 1 清零。

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:14 | 保留 | 必须保持复位值。 |
| 13 | NAK | 发送NAK握手<br>USBHS在发出一个NAK握手包后置该位，因为寄存器USBHS_DIEPxCTL的NAKS位被置位，或端点的Tx FIFO里没有包数据。 |
| 12:8 | 保留 | 必须保持复位值。 |

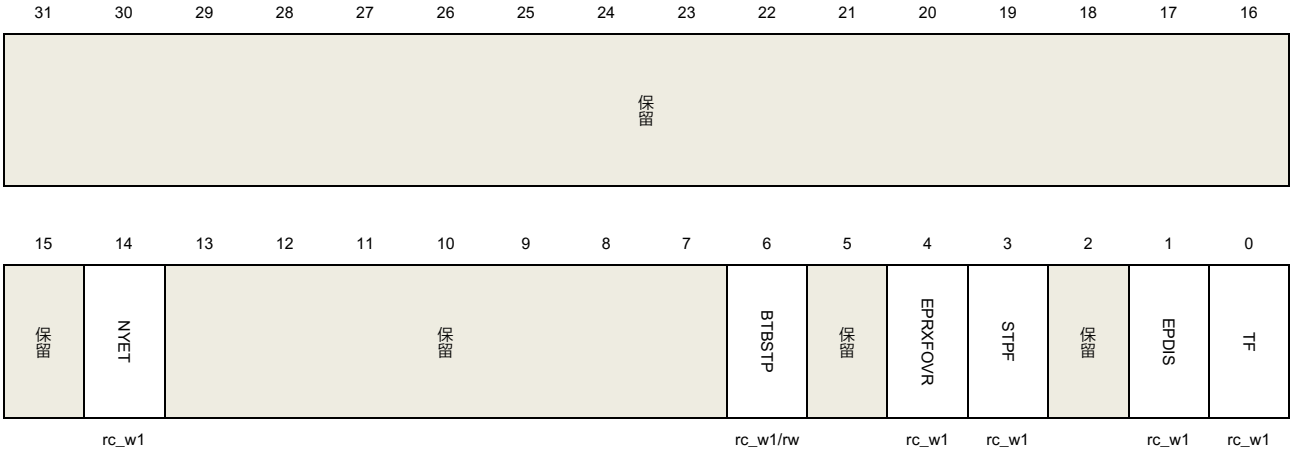
| | | |
|---|---|---|
| 7 | TXFE | 发送FIFO空<br>端点的Tx FIFO达到寄存器USBHS_GAHBCS的位TXFTH定义的空阈值。 |
| 6 | IEPNE | IN端点NAK有效<br>寄存器USBHS_DIEPxCTL的位SNAK的设置生效，该位可以通过写1清零或设置CNAK位 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | EPTXFUD | 端点Tx FIFO下溢<br>如果当IN令牌被接收后，Tx FIFO没有包数据，该标志被触发。 |
| 3 | CITO | 控制IN事务超时中断<br>在控制IN事务中，如果设备等待的握手包超时，该标志位被触发 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | EPDIS | 端点除能<br>端点除能时，该标志位被触发 |
| 0 | TF | 传输完成<br>当该端点的所有IN事务完成，该标志位被触发。 |

### 设备 OUT 端点 x 中断标志寄存器(USBHS_DOEPxINTF)(x = 0..5， x 是设备端点编号)

地址偏移：0x0B08 + (设备端点编号 x 0x20)
复位值：0x0000 0000

该寄存器包含 OUT 端点的状态和事件，当获得一个 OUT 端点的中断时，应该读取该端点的中断标志寄存器，从而获知中断源。该寄存器的标志位通常硬件置位，各位写 1 清零。

该寄存器采用字(32位)访问



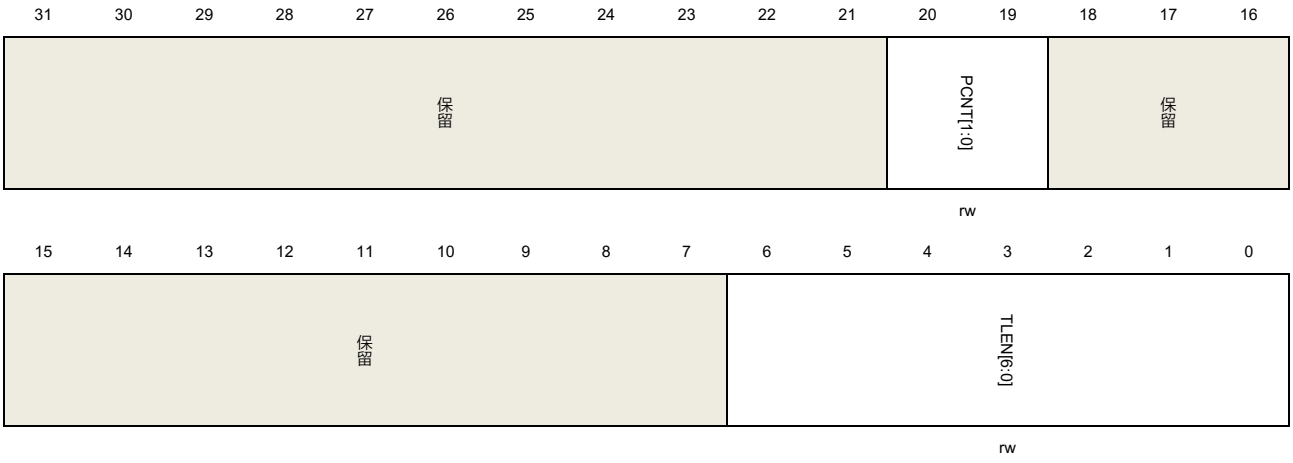

| 位/位域 | 名称 | 描述 |
|---|---|---|

| | | |
|---|---|---|
| 31:15 | 保留 | 必须保持复位值。 |
| 14 | NYET | 发送NYET握手<br>发送NYET握手包时，该标志被触发 |
| 13:7 | 保留 | 必须保持复位值。 |
| 6 | BTBSTP | 连续SETUP包(适用于控制OUT端点)<br>当一个控制OUT端点接收超过连续3个SETUP包时，该标志被触发。 |
| 5 | 保留 | 必须保持复位值。 |
| 4 | EPRXFOVR | 端点Rx FIFO上溢<br>当OUT令牌被接收时，如果OUT端点的Rx FIFO没有足够的空间存放数据包，该位被触发。在这种情况下，USBHS不能接收OUT数据包，发送一个NAK握手包。 |
| 3 | STPF | SETUP阶段完成(适用于控制OUT端点)<br>当一个SETUP阶段完成，也就是USBHS在一个setup令牌后接收了一个IN或OUT令牌，该位被置位。 |
| 2 | 保留 | 必须保持复位值。 |
| 1 | EPDIS | 端点除能<br>端点除能时，该标志位被触发 |
| 0 | TF | 传输完成<br>当该端点的所有OUT事务完成，该标志位被触发 |

### 设备 IN 端点 0 传输长度寄存器(USBHS_DIEP0LEN)

地址偏移：0x0910
复位值：0x0000 0000

该寄存器采用字(32位)访问



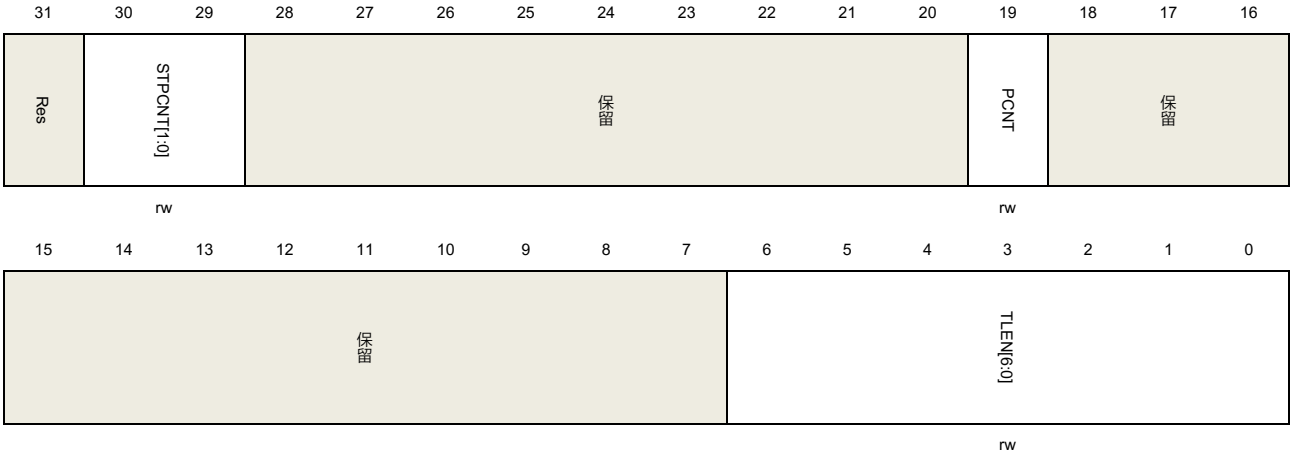

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:21 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 20:19 | PCNT[1:0] | 包数<br>传输中被发送的数据包数量<br>在端点使能之前，软件设置该位，在传输开始后，该域在每次数据包成功发送后自动减少。 |
| 18:7 | 保留 | 必须保持复位值。 |
| 6:0 | TLEN[6:0] | 传输长度<br>一次传输的数据总字节数<br>该域是IN传输中需要发送的包数据的总字节数，在端点使能之前，软件设置该位，在软件或DMA成功地将包数据写入端点的Tx FIFO中，该域减少与包数据大小相同的数值。 |

### 设备 OUT 端点 0 传输长度寄存器(USBHS_DOEP0LEN)

地址偏移：0x0B10
复位值：0x0000 0000

该寄存器采用字(32位)访问



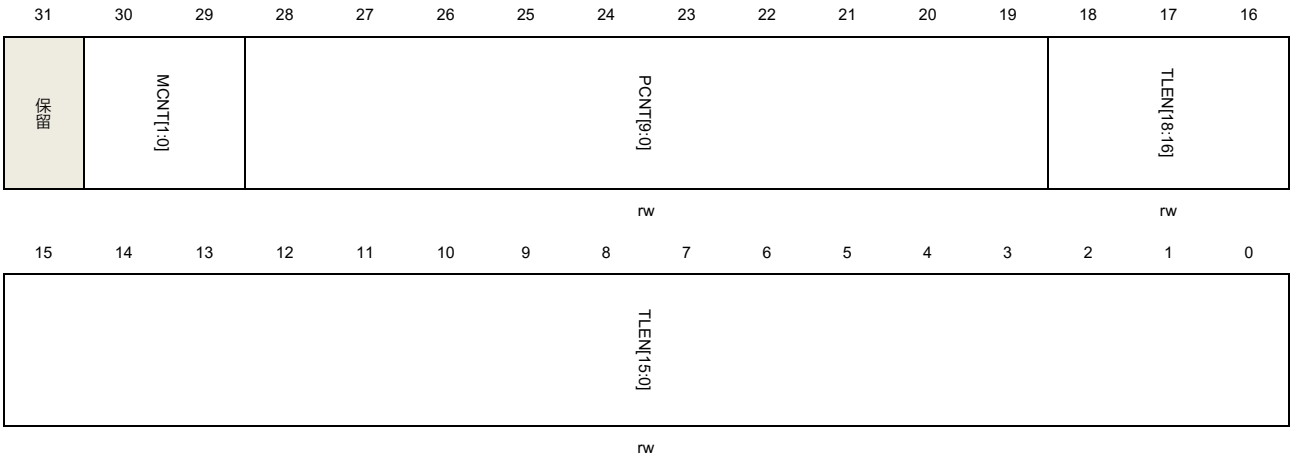

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30:29 | STPCNT[1:0] | SETUP包计数<br>该域定义端点可以接受的最大连续SETUP包数量<br>在SETUP传输之前，设置该域，每当连续SETUP包接收到时，该域值减1，当该域达到0时，寄存器USBHS_DOEP0INTF的BTBSTP标志被触发。<br>00：0个包<br>01：1个包<br>10：2个包<br>11：3个包 |
| 28:20 | 保留 | 必须保持复位值。 |
| 19 | PCNT | 包计数<br>一次传输中应该接收到包数量。 |

| | | |
|---|---|---|
| | | 在端点使能前，软件设置该位，在传输开始后，每当数据包接收到后，该域数值自动减少。 |
| 18:7 | 保留 | 必须保持复位值。 |
| 6:0 | TLEN[6:0] | 传输长度<br>传输中数据总字数。<br>该域是OUT传输中需要接收的包数据的总字节数，在端点使能之前，软件设置该位，在软件或DMA成功地将包数据读取端点的Rx FIFO中，该域减少与包数据大小相同的数值。 |

### 设备 IN 端点 x 传输长度寄存器(USBHS_DIEPxLEN)(x = 1..5， x 是设备端点编号)

地址偏移：0x910 + (设备端点编号 x 0x20)
复位值：0x0000 0000

该寄存器采用字(32位)访问



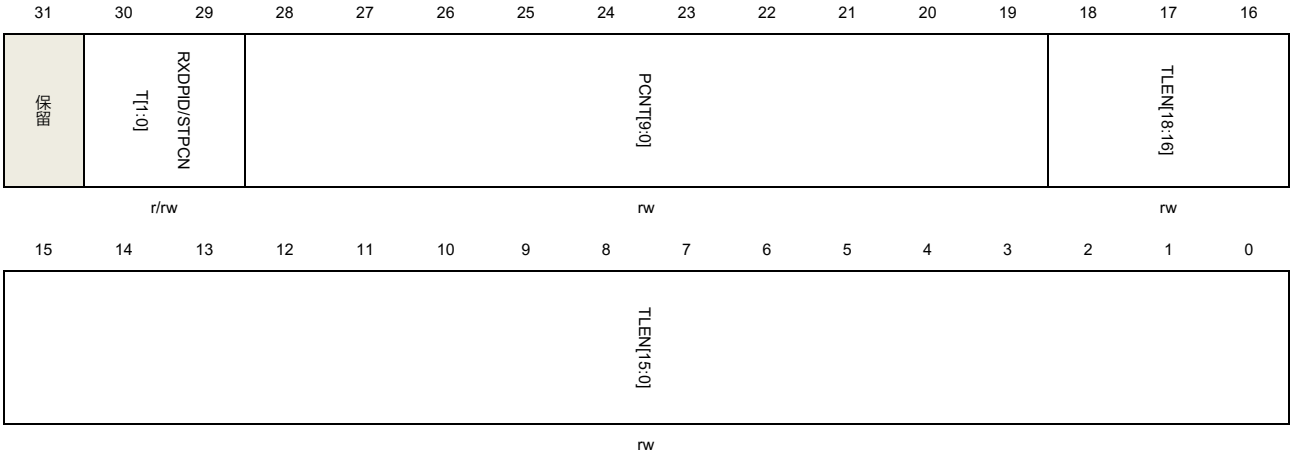

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:29 | 保留 | 必须保持复位值。 |
| 30:29 | MCNT[1:0] | 多包个数<br>该域描述在一帧内需要传输的包的个数<br>01:1个包<br>10:2个包<br>11:3个包 |
| 28:19 | PCNT[9:0] | 包数量<br>传输中被发送的数据包数量<br>在端点使能之前，软件设置该位，在传输开始后，该域在每次数据包成功发送后自动减少。 |

| | | |
|---|---|---|
| 18:0 | TLEN[18:0] | 传输长度<br>传输的数据总字节数<br>该域是IN传输中需要发送的包数据的总字节数，在端点使能之前，软件设置该位，在软件或DMA成功地将包数据写入端点的Tx FIFO中，该域减少与包数据大小相同的数值。 |

### 设备 OUT 端点 x 传输长度寄存器(USBHS_DOEPxLEN)(x = 1..5， x 是设备端点编号)

地址偏移：0x0B10 + (设备端点编号 x 0x20)
复位值：0x0000 0000

该寄存器采用字(32位)访问



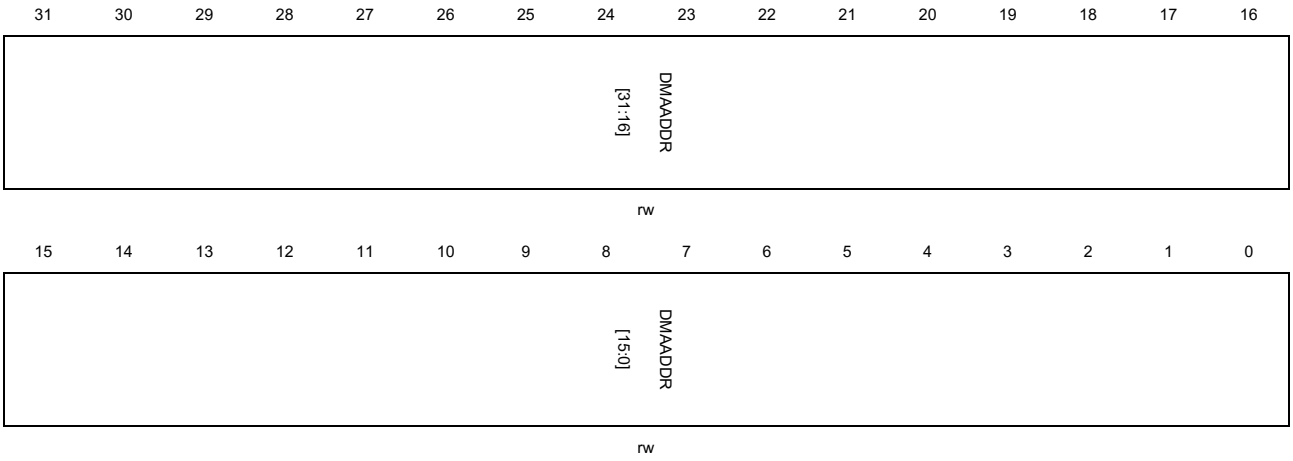

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31 | 保留 | 必须保持复位值。 |
| 30:29 | RXDPID[1:0] | 接收数据PID(适用于同步OUT端点)<br>该域保存该端点该数据包所接受的最后一个数据包的PID<br>00：DATA0<br>01：DATA2<br>10：DATA1<br>11：MDATA |
| | STPCNT[1:0] | SETUP包数(适用于控制OUT端点)<br>该位定义该端点可以接受连续SETUP最大包数<br>在SETUP传输之前，设置该域，每当连续SETUP包接收到时，该域值减1，当该域达到0时，寄存器USBHS_DOEP0INTF的BTBSTP标志被触发。<br>00：0个包<br>01：1个包<br>10：2个包<br>11：3个包 |

| | | |
|---|---|---|
| 28:19 | PCNT[9:0] | 包数<br>传输中应该接收到包数量<br>在端点使能前，软件设置该位，在传输开始后，每当数据包接收到后，该域数值自动减少。 |
| 18:0 | TLEN[18:0] | 传输长度<br>传输中数据总字数<br>该域是IN传输中需要接收的包数据的总字节数，在端点使能之前，软件设置该位，在软件或DMA成功地将包数据读取端点的Rx FIFO中，该域减少与包数据大小相同的数值。 |

### 设备 IN 端点 x DMA 地址寄存器(USBHS_DIEPxDMAADDR)/设备 OUT 端点 x DMA 地址寄存器(USBHS_DOEPxDMAADDR)(x = 0..5， x 是设备端点编号)

地址偏移：
IN 端点：0x0914 + (设备端点编号 x 0x20)
OUT 端点：0x0B14 + (设备端点编号 x 0x20)
复位值：0x0000 0000

该寄存器采用字(32位)访问



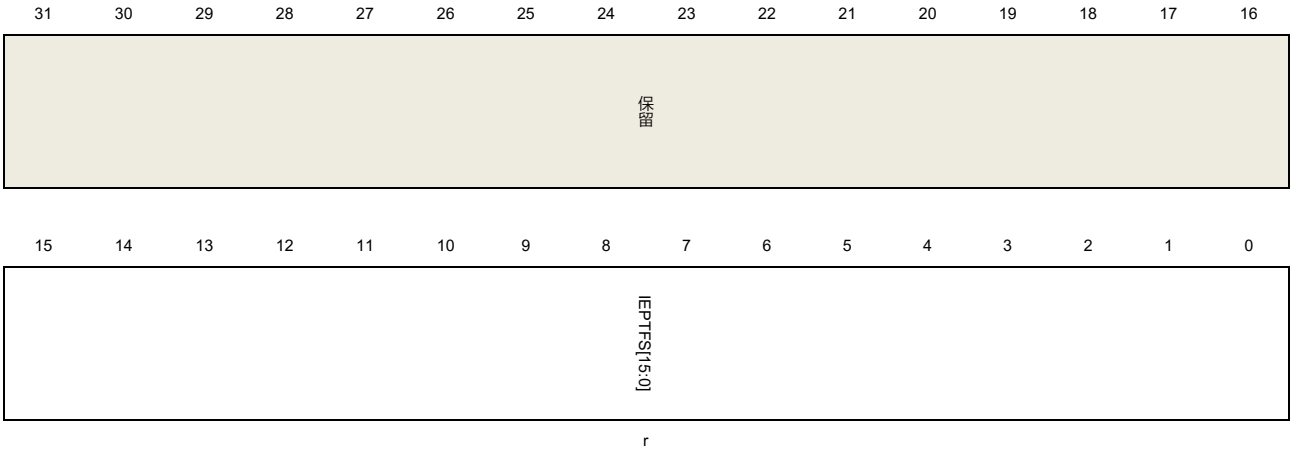

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:0 | DMAADDR[31:0] | DMA地址<br>该域定义端点的DMA地址，DMA使用该地址为IN端点提取包数据，或为OUT端点写入包数据。 |

### 设备 IN 端点 x 发送 FIFO 状态寄存器(USBHS_DIEPxTFSTAT)(x = 0..5， x 是设备端点编号)

地址偏移：0x0918 + (设备端点编号 x 0x20)
复位值：0x0000 0200

该寄存器包含每个端点的 Tx FIFO 的信息。

该寄存器采用字(32位)访问



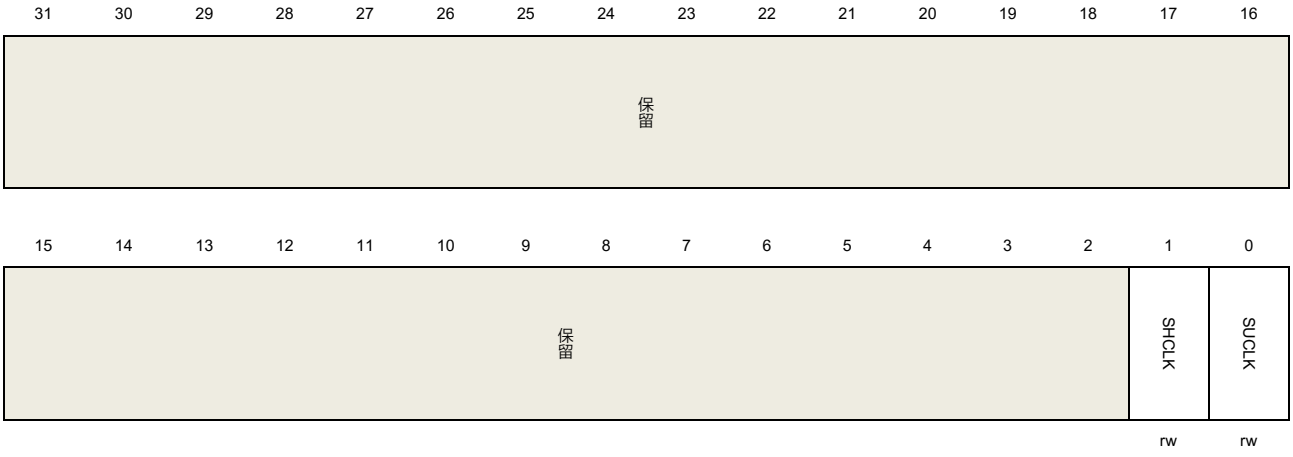

| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:16 | 保留 | 必须保持复位值。 |
| 15:0 | IEPTFS[15:0] | IN端点的Tx FIFO可用空间<br>IN端点的Tx FIFO可用空间用32位字为单位<br>0：FIFO是满的<br>1：1字可用<br>…<br>n：n字可用 |

### 29.7.4. 电源和时钟控制寄存器(USBHS_PWRCLKCTL)

地址偏移：0x0E00

复位值：0x0000 0000

该寄存器采用字(32位)访问




| 位/位域 | 名称 | 描述 |
|---|---|---|
| 31:2 | 保留 | 必须保持复位值。 |

| | | |
|---|---|---|
| 1 | SHCLK | 停止HCLK<br>停止HCLK，节省电量<br>0：HCLK未停止<br>1：HCLK停止 |
| 0 | SUCLK | 停止USB时钟<br>停止USB时钟，节省电量<br>0：USB时钟未停止<br>1：USB时钟停止 |

# 30. 版本历史

**表 30-1. 版本历史**

| 版本号. | 说明 | 日期 |
|---|---|---|
| 1.0 | 初稿发布 | 2023 年 4 月 18 日 |
| 1.1 | 1. FMC 章节：增加选项字节出厂默认值描述；PID1 更正为 PID；选项字节禁止再次修改。<br>2、 USB 章节：GINTF 寄存器的 bit24 描述的 HPCSR 改为 HPCS。 | 2023 年 11 月 29 日 |
| 1.2 | 1. ENET 章节：修改图 27-8 和 27-10 描述符顺序<br>2. 23.7.5 控制寄存器(TLI_CTL)：bit29，将数据使能极性改为非数据使能极性<br>3. USART 章节：Control register 3 (USART_CTL3)寄存器的 EBIE 和 RTIE 描述 0 和 1 写反了，修改为 0：禁用；1：使能；修改 USART 发送器描述，在 USART_CTL0 寄存器中置位 UEN 位，使能 USART；这个应该在 USART 波特率设置以后生效<br>4. PMU 章节：3.3.4 PDR_ON 引脚上拉具体阻值参考数据手册；更新图 3-1 电源域预览 | 2025 年 2 月 19 日 |
| 1.3 | 1. USBFS 章节：28.7.3 和 29.7.3，设备状态寄存器(USBHS_DSTAT)描述修改为：<br>设备状态寄存器(USBHS_DSTAT)<br>0：设备不在挂起状态<br>1：设备在挂起状态 | 2025 年 3 月 13 日 |
| 1.4 | 1. 19.3.11 章节，修改半双工模式描述<br>2. 20.3.11 章节，在 time out feature 中将 35ms 改为 25ms<br>3. 23.5.4 章节，LTI_LxHPOS 和 LTI_LxVPOS 修改为 TLI_LxHPOS 和 TLI_LxVPOS<br>4. 26.3.6 章节，CAN 波特率公式 tBS1 修改<br>5. 21.3 章节，删除 SPI 结构框图中的 FIFO<br>6. 29.1.2 章节，增加注意：I2S 串行时钟的配置值需设置为低于 PCLK 时钟的 1/6 倍以下(不包含 1/6) | 2025 年 8 月 7 日 |
| 1.5 | 1. 在 ***20.3.11 章节***中：增加"注意：当使用 SMBus 主机模式时，默认检查 I2C_SMBA 引脚，即使不配置端口配置为 AFIO(I2C0 对 | 2026 年 1 月 19 日 |

| 版本号. | 说明 | 日期 |
| --- | --- | --- |
| | 应 PB5，I2C1 对应 PB12，I2C2 对应 PA9)，当使能错误中断时且 I2C_SMBA 引脚为低电平时会产生中断。因此当不需要 SMBus 警报时应规避使用 I2C_SMBA 管脚，且保证相关 IO 是高电平。<br>2. 在 ***3 章节***中：1.2V 域改成 VCORE 域;将文档中表示管脚的"VDD""VDDA""VREFP""VREFN""VCORE"都写为无下标形式，表示具体电压时，为带下标写法。<br>3. 在 ***21.4 章节***中：删除描述"通过配置 SPI_QCTL 中的 IO23_DRV 位，在常规非四线 SPI 模式下，软件可以驱动 IO2 引脚和 IO3 引脚为高电平"<br>4. 在 ***21.5.5 章节***中：增加 SPI_CRC 相关描述<br>5. 在 ***22.4 章节***中：修改 DCI 引脚信号名称<br>6. 在 ***19.3.4 章节***中：接收采样删除起始位<br>7. 在 ***26.4.3 章节***中：CAN_TSTAT 寄存器中 CAN_TIM 修改为 CAN_TMI。<br>8. 在 ***12.2.3 章节***中：修改 BSD JTAG TAP 的声明为 MCU JTAG TAP。<br>9. 在 ***19.4.9 章节***中：在 RT[23:0]位增加备注"注意：当 16 倍过采样时，RT[23:0]不能被配置为 0xFFFFFFFF。"<br>10. 在 ***19.3.10 章节***中：红外 IrDA，低功耗分频配高后，无法正确输出波形，增加描述，在配置 PSC[7:0]的同时需要配置波特率寄存器配置为 16*PSC[7:0].<br>11. 在 ***16.1.3 章节***中：添加使用注意<br>12. 在 ***26.3.7 章节***中：修改 CAN 协议同步边沿的描述，并增加符合 ISO11898 的标准 Note。<br>13. 在 ***20.4.6 章节***中：删除 BTC 多余描述。 | |
| 1.6 | 1. 在 ***18.x.4(x=1,..,5) 章节***：修改寄存器 TIMERx_CHCTLy 中 CHxCOMFEN 位域的描述。<br>2. 在 ***24.4.1 章节***：在 SDIO I/O 定义表中增加 SDIO 引脚名称与信号名称的对应关系。<br>3. 在 ***28.5.6 和 29.5.7 章节***：增加主机模式的操作流程注意事项。<br>4. 在 ***28.7.2 章节***：删除 USBFS_HCTL 寄存器。 | 2026 年 7 月 29 日 |

| 版本号. | 说明 | 日期 |
| --- | --- | --- |
| | 5. 在 ***6.3 章节***: 修改 DAC 下溢中断描述。<br>6. 在 ***10.4.1 章节***: 改 DAC0/DAC1 为 DAC0_OUT0/DAC0_OUT1<br>7. 在 ***27.4.24,27.4.25,27.4.34 章节***:修改 MSC 和 ptp 控制寄存器位域描述 | |

# Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the "Product"), is owned by the Company according to the laws of the People's Republic of China and other applicable laws. The Company reserves all rights under such laws and no Intellectual Property Rights are transferred (either wholly or partially) or licensed by the Company (either expressly or impliedly) herein. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

To the maximum extent permitted by applicable law, the Company makes no representations or warranties of any kind, express or implied, with regard to the merchantability and the fitness for a particular purpose of the Product, nor does the Company assume any liability arising out of the application or use of any Product. Any information provided in this document is provided only for reference purposes. It is the sole responsibility of the user of this document to determine whether the Product is suitable and fit for its applications and products planned, and properly design, program, and test the functionality and safety of its applications and products planned using the Product. the Product is designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only, and the Product is not designed or intended for use in (i) safety critical applications such as weapons systems, nuclear facilities, atomic energy controller, aeronautic or aerospace applications, pollution control or hazardous substance management; (ii) life-support systems, other medical equipment or systems (including life support equipment and surgical implants); and/or (iii) other uses where the failure of the device or the Product can reasonably be expected to result in personal injury, death, or severe property or environmental damage (collectively "Unintended Uses"). Customers shall take any and all actions to ensure the Product meets the applicable laws and regulations. The Company is not liable for, in whole or in part, and customers shall hereby release the Company as well as its suppliers and/or distributors from, any claim, damage, or other liability arising from or related to all Unintended Uses of the Product. Customers shall indemnify and hold the Company, and its officers, employees, subsidiaries, affiliates as well as its suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Product.

While the Company has implemented advanced security features, the Product may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on Customer's applications and products, and to the maximum extent permitted by applicable law, the Company accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

Information in this document is provided solely in connection with the Product. The Company reserves the right to make changes, corrections, modifications or improvements to this document and the Product described herein at any time without notice. The Company shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2026 GigaDevice Semiconductor Inc. - All rights reserved