



用户手册

User Manual

**MM32F3270**

基于ARM<sup>®</sup>Cortex<sup>®</sup>-M3内核的32位微控制器

版本：0.9

保留不通知的情况下，更改相关资料的权利

# 目 录

目录.....	I
附图目录 .....	XXI
表格目录 .....	XXVII
1 系统和存储器架构 .....	1
1.1 系统架构简介 .....	1
1.1.1 ICode 总线 (IBUS) .....	3
1.1.2 DCode 总线 (DBUS) .....	3
1.1.3 系统总线 (SBUS) .....	3
1.1.4 DMA 总线 .....	3
1.1.5 总线矩阵 (BUS Matrix) .....	3
1.1.6 AHB 到 APB 桥 (AHB-APB APBx) .....	3
1.2 存储器介绍 .....	3
1.2.1 存储器映像和寄存器编址 .....	3
1.2.2 内置的 SRAM .....	7
1.2.3 闪存存储器概述 .....	7
1.2.4 启动配置 .....	7
1.2.5 引导程序 .....	8
2 嵌入式闪存 (Embedded FLASH) .....	9
2.1 简介 .....	9
2.2 闪存构成与说明 .....	9
2.2.1 闪存构成 .....	9
2.2.2 选项字节说明 .....	10
2.2.3 区块读保护字节说明 .....	13
2.2.4 保密空间说明 .....	13
2.3 闪存操作与流程 .....	13
2.3.1 闪存读操作 .....	13
2.3.2 闪存编程方式与操作流程 .....	14
2.3.3 对闪存块操作限制的解除与使能 .....	17
2.3.4 主闪存块擦除 .....	20
2.3.5 主闪存块编程 .....	22
2.3.6 选项字节区块擦除 .....	24
2.3.7 选项字节区块编程 .....	24
2.3.8 闪存保护 .....	25
2.4 Flash 寄存器描述 .....	28
2.4.1 闪存访问控制寄存器 (FLASH_ACR) .....	28
2.4.2 闪存访问控制寄存器 (FLASH_KEYR) .....	29
2.4.3 闪存 OPTKEY 寄存器 (FLASH_OPTKEYR) .....	29
2.4.4 闪存状态寄存器 (FLASH_SR) .....	30
2.4.5 闪存控制寄存器 (FLASH_CR) .....	31

2.4.6	闪存地址寄存器 (FLASH_AR) .....	32
2.4.7	选项字节寄存器 (FLASH_OBR) .....	33
2.4.8	写保护寄存器 (FLASH_WRPRx) .....	34
3	高速缓冲存储器 (CACHE) .....	36
3.1	基本操作 .....	36
3.1.1	禁用 Cache .....	36
3.1.2	使能 Cache .....	36
3.2	操作流程 .....	36
3.2.1	自动电源和自动失效模式 (APAI) .....	37
3.2.2	手动电源和自动失效模式 (MPAI) .....	37
3.2.3	手动电源和手动失效模式 (MPMI) .....	37
3.3	寄存器描述 .....	38
3.3.1	配置及控制寄存器 (CACHE_CCR) .....	38
3.3.2	状态寄存器 (CACHE_SR) .....	40
3.3.3	中断屏蔽寄存器 (CACHE_IMR) .....	40
3.3.4	中断状态寄存器 (CACHE_ISR) .....	41
3.3.5	Cache 命中统计寄存器 (CACHE_CSHR) .....	42
3.3.6	Cache 丢失统计寄存器 (CACHE_CSMR) .....	42
4	电源控制 (PWR) .....	43
4.1	供电系统 .....	43
4.1.1	模拟模块供电 .....	43
4.1.2	数字模块供电 .....	44
4.1.3	电池备份模块供电 .....	44
4.1.4	5v 电源域 .....	44
4.1.5	1.5v 备份域 .....	44
4.1.6	1.5v 可关断电源域 .....	44
4.2	电源管理器 .....	45
4.2.1	上电复位 (POR) 和掉电复位 (PDR) .....	45
4.2.2	可编程电压监测器 (PWD) .....	45
4.3	功耗控制 .....	46
4.3.1	运行模式降低系统时钟 .....	48
4.3.2	外部时钟的控制 .....	49
4.3.3	低功耗运行模式 (Low Power Run Mode) .....	49
4.3.4	睡眠模式 (Sleep Mode) .....	49
4.3.5	低功耗睡眠模式 (Low Power Sleep Mode) .....	50
4.3.6	停机模式 (Stop Mode) .....	51
4.3.7	深度停机模式 (DeepStop Mode) .....	52
4.3.8	待机模式(Standby Mode) .....	54
4.4	功能描述 .....	54
4.4.1	滤波功能 .....	54
4.4.2	快速唤醒功能 .....	55
4.5	电源控制寄存器 .....	55
4.5.1	电源控制寄存器 1 (PWR_CR1) .....	55

4.5.2	电源控制/状态寄存器 (PWR_CSR).....	56
4.5.3	电源控制寄存器 2 (PWR_CR2) .....	57
4.5.4	电源控制寄存器 3 (PWR_CR3) .....	58
4.5.5	电源控制寄存器 4 (PWR_CR4) .....	59
4.5.6	电源控制寄存器 5 (PWR_CR5) .....	60
4.5.7	电源控制寄存器 6 (PWR_CR6) .....	61
4.5.8	电源状态寄存器 (PWR_SR) .....	62
4.5.9	电源状态清除寄存器(PWR_SCR) .....	62
4.5.10	电源配置寄存器(PWR_CFGR) .....	63
5	时钟和复位(RCC).....	64
5.1	复位单元.....	64
5.1.1	简介 .....	64
5.1.2	功能框图 .....	64
5.1.3	主要特征 .....	64
5.1.4	功能描述 .....	64
5.2	时钟单元.....	65
5.2.1	简介 .....	65
5.2.2	功能框图 .....	66
5.2.3	主要特征 .....	66
5.2.4	功能描述 .....	66
5.3	寄存器描述 .....	73
5.3.1	时钟控制寄存器 (RCC_CR) .....	73
5.3.2	时钟配置寄存器 (RCC_CFGR) .....	77
5.3.3	时钟中断寄存器 (RCC_CIR) .....	79
5.3.4	AHB3 外设复位寄存器 (RCC_AHB3RSTR) .....	83
5.3.5	AHB2 外设复位寄存器 (RCC_AHB2RSTR) .....	84
5.3.6	AHB1 外设复位寄存器 (RCC_AHB1RSTR) .....	84
5.3.7	APB2 外设复位寄存器 (RCC_APB2RSTR) .....	86
5.3.8	APB1 外设复位寄存器 (RCC_APB1RSTR) .....	88
5.3.9	AHB3 外设时钟使能寄存器 (RCC_AHB3ENR) .....	91
5.3.10	AHB2 外设时钟使能寄存器 (RCC_AHB2ENR) .....	92
5.3.11	AHB1 外设时钟使能寄存器 (RCC_AHB1ENR) .....	93
5.3.12	APB2 外设时钟使能寄存器 (RCC_APB2ENR) .....	95
5.3.13	APB1 外设时钟使能寄存器 (RCC_APB1ENR) .....	97
5.3.14	备份域控制寄存器 (RCC_BDCR) .....	101
5.3.15	控制状态寄存器 (RCC_CSR) .....	103
5.3.16	系统配置寄存器 (RCC_SYSCFG) .....	106
5.3.17	时钟配置寄存器 2 (RCC_CFGR2) .....	107
5.3.18	内部时钟校准寄存器 (RCC_ICSCR) .....	109
5.3.19	PLL 配置寄存器 (RCC_PLLCFGR) .....	110
5.3.20	HSI 延迟寄存器 (RCC_HSIDLY) .....	111
5.3.21	HSE 延迟寄存器 (RCC_HSEDLY) .....	112
5.3.22	PLL 延迟寄存器 (RCC_PLLDLY) .....	112

6	系统控制器 (SYSCFG) .....	113
6.1	SYSCFG 简介 .....	113
6.2	寄存器描述 .....	113
6.2.1	SYSCFG 配置寄存器 (SYSCFG_CFGR) .....	113
6.2.2	外部中断配置寄存器 1 (SYSCFG_EXTICR1) .....	115
6.2.3	外部中断配置寄存器 2 (SYSCFG_EXTICR2) .....	116
6.2.4	外部中断配置寄存器 3 (SYSCFG_EXTICR3) .....	116
6.2.5	外部中断配置寄存器 4 (SYSCFG_EXTICR4) .....	117
6.2.6	SYSCFG 配置寄存器 (SYSCFG_CFGR2) .....	118
6.2.7	电源检测配置状态寄存器 (SYSCFG_PDETCSR) .....	119
6.2.8	VOSDLY 配置寄存器 (SYSCFG_VOSDLY) .....	121
7	中断和事件(EXTI) .....	122
7.1	EXTI 简介 .....	122
7.2	功能框图.....	122
7.3	主要特征.....	122
7.4	中断说明&向量表.....	122
7.5	功能描述.....	124
7.5.1	唤醒事件管理 .....	124
7.5.2	中断功能描述 .....	125
7.5.3	硬件中断输出 .....	125
7.5.4	硬件事件输出 .....	125
7.5.5	软件中断事件输出 .....	125
7.5.6	外部中断映射 .....	125
7.6	寄存器描述 .....	126
7.6.1	中断屏蔽寄存器 (EXTI_IMR).....	127
7.6.2	事件屏蔽寄存器 (EXTI_EMR).....	127
7.6.3	上升沿触发选择寄存器 (EXTI_RTSR).....	128
7.6.4	下降沿触发选择寄存器 (EXTI_FTSR).....	129
7.6.5	软件中断事件寄存器 (EXTI_SWIER) .....	129
7.6.6	软件中断事件挂起寄存器 (EXTI_PR).....	130
8	直接存储器访问控制器(DMA).....	132
8.1	DMA 简介 .....	132
8.2	DMA 功能框图 .....	132
8.3	DMA 主要特征 .....	132
8.4	中断.....	133
8.5	DMA .....	133
8.5.1	DMA 请求映像 .....	133
8.6	功能描述.....	134
8.6.1	DMA 处理 .....	134
8.6.2	仲裁器.....	135
8.6.3	DMA 通道 .....	135
8.6.4	可编程的数据传输宽度, 对齐方式和数据大小端 .....	136
8.6.5	错误管理 .....	138

8.7 DMA 寄存器描述.....	139
8.7.1 DMA 中断状态寄存器(DMA_ISR).....	139
8.7.2 DMA 中断标志清除寄存器(DMA_IFCR) .....	140
8.7.3 DMA 通道 x 配置寄存器(DMA_CCRx)(x=1~7).....	142
8.7.4 DMA 通道 x 传输数量寄存器(DMA_CNDTRx)(x=1~7).....	144
8.7.5 DMA 通道 x 外设地址寄存器(DMA_CPARx)(x=1~7).....	145
8.7.6 DMA 通道 x 存储器地址寄存器(DMA_CMARx)(x = 1~7) .....	145
9 备份寄存器(BKP) .....	146
9.1 BKP 简介 .....	146
9.2 主要特征.....	146
9.3 功能描述.....	147
9.3.1 时钟校准.....	147
9.3.2 侵入检测 .....	147
9.4 寄存器描述 .....	147
9.4.1 备份数据寄存器 n(BKP_DRn)(n = 1 .. 10) .....	147
9.4.2 时钟校准寄存器 (BKP_RTCCR) .....	148
9.4.3 备份控制寄存器 (BKP_CR) .....	149
9.4.4 备份控制 I 状态寄存器 (BKP_CSR) .....	150
10 时钟回馈系统 (CRS) .....	151
10.1 CRS 简介.....	151
10.2 CRS 功能框图 .....	152
10.3 CRS 主要特征 .....	152
10.4 中断.....	152
10.5 CRS 功能描述 .....	153
10.5.1 同步输入.....	153
10.5.2 频率误差测量 .....	153
10.5.3 频率误差计算及自动校准 .....	156
10.5.4 CRS 初始化及配置.....	157
10.6 CRS 低功耗模式 .....	158
10.7 CRS 寄存器描述 .....	158
10.7.1 CRS 控制寄存器(CRS_CR) .....	158
10.7.2 CRS 配置寄存器(CRS_CFGR) .....	160
10.7.3 CRS 中断状态寄存器(CRS_ISR).....	161
10.7.4 CRS 中断标志清除寄存器(CRS_ICR) .....	164
11 通用端口(GPIO).....	165
11.1 GPIO 简介 .....	165
11.2 GPIO 功能框图 .....	165
11.3 GPIO 端口配置 .....	165
11.4 主要特征.....	167
11.5 功能描述.....	167
11.5.1 复用功能 .....	167
11.5.2 GPIO 锁定机制 .....	167
11.5.3 输入配置 .....	167

11.5.4	输出配置 .....	168
11.5.5	复用功能配置 .....	169
11.5.6	模拟输入配置 .....	170
11.5.7	外设的 GPIO 配置 .....	170
11.5.8	OSC_IN/OSC_OUT 作为 GPIO 端口 PD0/PD1 .....	172
11.5.9	SWD 复用功能重映射 .....	172
11.6	GPIO 寄存器描述 .....	172
11.6.1	端口配置低寄存器 (GPIOx_CRL)(x = A..D) .....	173
11.6.2	端口配置高寄存器 (GPIOx_CRH)(x = A..D) .....	174
11.6.3	端口输入数据寄存器 (GPIOx_IDR)(x = A..D) .....	175
11.6.4	端口输出数据寄存器 (GPIOx_ODR)(x = A..D) .....	175
Field	175	
ODR	175	
11.6.5	端口设置/清除寄存器 (GPIOx_BSRR)(x = A..D) .....	176
11.6.6	端口位清除寄存器 (GPIOx_BRR)(x = A..D) .....	177
11.6.7	端口配置锁定寄存器 (GPIOx_LCKR)(x = A..D) .....	177
11.6.8	端口输出开漏控制寄存器 (GPIOx_DCR)(x = A..D) .....	178
11.6.9	端口复用功能低位寄存器 (GPIOx_AFRL)(x = A..D) .....	178
11.6.10	端口复用功能高位寄存器 (GPIOx_AFRH)(x = A..D) .....	179
12	高级定时器 (TIM1/8) .....	181
12.1	TIM1 简介 .....	181
12.2	功能框图 .....	181
12.3	主要特征 .....	181
12.4	中断 .....	182
12.5	DMA .....	182
12.6	功能描述 .....	182
12.6.1	时钟 .....	182
12.6.2	输入捕获 .....	190
12.6.3	比较/输出 .....	196
12.6.4	DMA 模式 .....	206
12.6.5	调试模式 .....	206
12.7	寄存器描述 .....	206
12.7.1	控制寄存器 1 (TIM1_CR1) .....	207
12.7.2	控制寄存器 2 (TIM1_CR2) .....	209
12.7.3	从模式控制寄存器 (TIM1_SMCR) .....	212
12.7.4	DMA/中断使能寄存器 (TIM1_DIER) .....	216
12.7.5	状态寄存器 (TIM1_SR) .....	218
12.7.6	事件产生寄存器 (TIM1_EGR) .....	220
12.7.7	捕获/比较模式寄存器 (TIM1_CCMR1) .....	222
12.7.8	捕获/比较模式寄存器 2 (TIM1_CCMR2) .....	226
12.7.9	捕获/比较使能寄存器 (TIM1_CCER) .....	230
12.7.10	计数器 (TIM1_CNT) .....	232
12.7.11	预分频器 (TIM1_PSC) .....	233

12.7.12	自动预装载寄存器 (TIM1_ARR) .....	233
12.7.13	重复计数寄存器 (TIM1_RCR) .....	233
12.7.14	捕获/比较寄存器 1 (TIM1_CCR1) .....	234
12.7.15	捕获/比较寄存器 2 (TIM1_CCR2) .....	235
12.7.16	捕获/比较寄存器 3 (TIM1_CCR3) .....	235
12.7.17	捕获比较寄存器 4 (TIM1_CCR4) .....	235
12.7.18	刹车和死区寄存器 (TIM1_BDTR) .....	236
12.7.19	DMA 控制寄存器 (TIM1_DCR) .....	238
12.7.20	连续模式的 DMA 地址 (TIM1_DMAR) .....	239
12.7.21	捕获/比较模式寄存器 3 (TIM1_CCMR3) .....	240
12.7.22	捕获/比较寄存器 5 (TIM1_CCR5) .....	240
12.7.23	PWM 移相/DMA repeat 更新请求使能寄存器 (TIM1_PDER) .....	241
12.7.24	PWM 移相向下计数捕获/比较寄存器 (TIM1_CCRxFALL) .....	242
13	16 位通用定时器 (TIM3/4) .....	243
13.1	TIM3 简介 .....	243
13.2	功能框图 .....	243
13.3	主要特征 .....	243
13.4	中断 .....	244
13.5	DMA .....	244
13.6	功能描述 .....	244
13.6.1	时钟 .....	244
13.6.2	输入捕获 .....	250
13.6.3	比较/输出 .....	255
13.6.4	DMA 模式 .....	260
13.6.5	调试模式 .....	260
13.7	寄存器描述 .....	261
13.7.1	控制寄存器 1 (TIM3_CR1) .....	261
13.7.2	控制寄存器 2 (TIM3_CR2) .....	263
13.7.3	从模式控制寄存器 (TIM3_SMCR) .....	265
13.7.4	DMA/中断使能寄存器 (TIM3_DIER) .....	268
13.7.5	状态寄存器 (TIM3_SR) .....	270
13.7.6	事件产生寄存器 (TIM3_EGR) .....	272
13.7.7	捕获/比较模式寄存器 1 (TIM3_CCMR1) .....	274
13.7.8	捕获/比较模式寄存器 2 (TIM3_CCMR2) .....	278
13.7.9	捕获/比较使能寄存器 (TIM3_CCER) .....	282
13.7.10	计数器 (TIM3_CNT) .....	284
13.7.11	预分频器 (TIM3_PSC) .....	284
13.7.12	自动预装载寄存器 (TIM3_ARR) .....	285
13.7.13	捕获/比较寄存器 1 (TIM3_CCR1) .....	285
13.7.14	捕获/比较寄存器 2 (TIM3_CCR2) .....	286
13.7.15	捕获/比较寄存器 3 (TIM3_CCR3) .....	286
13.7.16	捕获比较寄存器 4 (TIM3_CCR4) .....	286
13.7.17	DMA 控制寄存器 (TIM3_DCR) .....	287

13.7.18 连续模式的 DMA 地址 (TIM3_DMAR) .....	288
<b>14 32 位定时器 (TIM2/5) .....</b>	<b>289</b>
14.1 TIM2 简介 .....	289
14.2 功能框图 .....	289
14.3 主要特征 .....	289
14.4 中断 .....	290
14.5 DMA .....	290
14.6 功能描述 .....	290
14.6.1 时钟 .....	290
14.6.2 输入捕获 .....	296
14.6.3 比较/输出 .....	301
14.6.4 DMA 模式 .....	306
14.6.5 调试模式 .....	306
14.7 寄存器描述 .....	307
14.7.1 控制寄存器 1 (TIM2_CR1) .....	307
14.7.2 控制寄存器 2 (TIM2_CR2) .....	309
14.7.3 从模式控制寄存器 (TIM2_SMCR) .....	311
14.7.4 DMA/中断使能寄存器 (TIM2_DIER) .....	314
14.7.5 状态寄存器 (TIM2_SR) .....	316
14.7.6 事件产生寄存器 (TIM2_EGR) .....	318
14.7.7 捕获/比较模式寄存器 (TIM2_CCMR1) .....	320
14.7.8 捕获/比较模式寄存器 2 (TIM2_CCMR2) .....	324
14.7.9 捕获/比较使能寄存器 (TIM2_CCER) .....	328
14.7.10 计数器 (TIM2_CNT) .....	330
14.7.11 预分频器 (TIM2_PSC) .....	331
14.7.12 自动预装载寄存器 (TIM2_ARR) .....	331
14.7.13 捕获/比较寄存器 1 (TIM2_CCR1) .....	331
14.7.14 捕获/比较寄存器 2 (TIM2_CCR2) .....	332
14.7.15 捕获/比较寄存器 3 (TIM3_CCR3) .....	332
14.7.16 捕获/比较寄存器 4 (TIM2_CCR4) .....	333
14.7.17 DMA 控制寄存器 (TIM2_DCR) .....	333
14.7.18 连续模式的 DMA 地址 (TIM2_DMAR) .....	334
14.7.19 TIMERx 选项寄存器 (TIMx_OR) .....	335
<b>15 基本定时器 (TIM6/7) .....</b>	<b>336</b>
15.1 TIM6/7 简介 .....	336
15.2 功能框图 .....	336
15.3 主要特征 .....	336
15.4 中断 .....	336
15.5 DMA .....	336
15.6 功能描述 .....	336
15.6.1 时钟 .....	336
15.6.2 调试模式 .....	338
15.7 寄存器描述 .....	338

15.7.1	控制寄存器 1 (TIM6/7_CR1) .....	338
15.7.2	DMA/中断使能寄存器 (TIM6/7_DIER) .....	339
15.7.3	状态寄存器 (TIM6/7_SR) .....	340
15.7.4	事件产生寄存器 (TIM6/7_EGR) .....	340
15.7.5	计数器 (TIM6/7_CNT) .....	341
15.7.6	预分频器 (TIM6/7_PSC) .....	341
15.7.7	自动预装载寄存器 (TIM6/7_ARR) .....	342
16	独立看门狗 (IWDG) .....	343
16.1	IWDG 简介 .....	343
16.2	功能框图 .....	344
16.3	主要特征 .....	345
16.4	功能描述 .....	345
16.5	独立看门狗超时时间 .....	346
16.6	寄存器描述 .....	347
16.6.1	键寄存器 (IWDG_KR) .....	347
16.6.2	预分频寄存器 (IWDG_PR) .....	348
16.6.3	重装载寄存器 (IWDG_RLR) .....	348
16.6.4	状态寄存器 (IWDG_SR) .....	349
16.6.5	控制寄存器 (IWDG_CR) .....	350
16.6.6	中断生成寄存器 (IWDG_IGEN) .....	351
16.6.7	计数寄存器 (IWDG_CNT) .....	352
16.6.8	分频计数寄存器 (IWDG_PS) .....	353
17	窗口看门狗 (WWWDG) .....	354
17.1	WWWDG 简介 .....	354
17.2	功能框图 .....	354
17.3	主要特性 .....	354
17.4	功能描述 .....	355
17.5	窗口看门狗超时时间 .....	356
17.6	寄存器描述 .....	357
17.6.1	控制寄存器 (WWWDG_CR) .....	357
17.6.2	配置寄存器 (WWWDG_CFGR) .....	357
17.6.3	状态寄存器 (WWWDG_SR) .....	358
18	实时时钟器 (RTC) .....	360
18.1	RTC 简介 .....	360
18.2	功能框图 .....	360
18.3	主要特征 .....	360
18.4	功能描述 .....	361
18.4.1	概述 .....	361
18.4.2	模块复位 .....	361
18.4.3	寄存器读取 .....	361
18.4.4	寄存器配置 .....	363
18.4.5	标志位产生 .....	363
18.4.6	RTC 闹钟描述 .....	364

18.4.7 RTC 外部中断事件输出 .....	364
<b>18.5 寄存器描述 .....</b>	<b>364</b>
18.5.1 控制寄存器高位 (RTC_CRH).....	365
18.5.2 控制寄存器低位(RTC_CRL).....	365
18.5.3 预分频装载寄存器高位(RTC_PRLH) .....	367
18.5.4 预分频装载寄存器低位(RTC_PRL).....	368
18.5.5 预分频器分频因子寄存器高位(RTC_DIVH) .....	368
18.5.6 预分频器分频因子寄存器低位(RTC_DIVL) .....	368
18.5.7 计数器寄存器高位(RTC_CNTH).....	369
18.5.8 计数器寄存器低位(RTC_CNTL) .....	369
18.5.9 闹钟寄存器高位(RTC_ALRH).....	370
18.5.10 闹钟寄存器低位(RTC_ALRL) .....	370
18.5.11 毫秒闹钟寄存器高位 (RTC_MSRH).....	370
18.5.12 RTC 毫秒闹钟寄存器低位 (RTC_MSRL) .....	371
18.5.13 RTC LSE 配置寄存器 (RTC_LSE_CFG).....	371
<b>19 通用异步收发器 (UART) .....</b>	<b>372</b>
19.1 UART 简介 .....	372
19.2 UART 功能框图 .....	373
19.3 UART 主要特征 .....	373
19.4 引脚定义及内部信号 .....	374
19.5 中断.....	374
19.6 DMA .....	375
19.7 UART 功能概述 .....	375
19.7.1 UART 特性描述.....	376
19.7.2 分数波特率发生器.....	377
19.7.3 采样 .....	377
19.7.4 容忍度.....	377
19.7.5 校验控制 .....	377
19.7.6 发送器.....	378
19.7.7 接收器.....	379
19.7.8 自动波特率检测 .....	380
19.7.9 9 位数据通信 .....	381
19.7.10 多处理器通信 .....	381
19.7.11 单线半双工通信 .....	382
19.7.12 智能卡.....	382
19.7.13 红外 IrDA 功能 .....	384
19.8 UART 寄存器描述 .....	385
19.8.1 UART 发送数据寄存器(UART_TDR).....	385
19.8.2 UART 接收数据寄存器(UART_RDR).....	386
19.8.3 UART 当前状态寄存器(UART_CSR).....	386
19.8.4 UART 中断状态寄存器(UART_ISR) .....	387
19.8.5 UART 中断使能寄存器(UART_IER) .....	389
19.8.6 UART 中断清除寄存器(UART_ICR) .....	391

19.8.7	UART 全局空制寄存器(UART_GCR) .....	393
19.8.8	UART 通用空制寄存器(UART_CCR).....	395
19.8.9	UART 波特率寄存器(UART_BRR) .....	397
19.8.10	UART 分数波特率寄存器(UART_FRA).....	397
19.8.11	UART 接收地址寄存器(UART_RXADDR) .....	398
19.8.12	UART 接收掩码寄存器(UART_RXMASK) .....	398
19.8.13	UART SCR 寄存器(UART_SCR).....	399
19.8.14	UART IDLE 数据长度寄存器(UART_IDLR) .....	400
19.8.15	UART 自动波特率寄存器(UART_ABRCR) .....	400
19.8.16	UART 红外功能控制寄存器(UART_IRDA).....	401
20	串行外设接口(SPI_I2S).....	403
20.1	SPI_I2S 功能框图 .....	403
20.2	SPI_I2S 简述 .....	403
20.3	SPI 功能描述 .....	403
20.3.1	概述 .....	403
20.3.2	SPI 主要特征.....	407
20.3.3	中断 .....	407
20.3.4	DMA 传输 .....	408
20.3.5	SPI 从模式 .....	409
20.3.6	SPI 主模式 .....	409
20.3.7	波特率设置 .....	410
20.4	I2S 功能描述 .....	410
20.4.1	I2S 主要特征 .....	410
20.4.2	I2S 总线接口 .....	411
20.4.3	数据格式 .....	411
20.4.4	通信标准 .....	411
20.4.5	DMA 传输 .....	413
20.4.6	从模式.....	414
20.4.7	主模式.....	414
20.4.8	中断 .....	415
20.4.9	时钟预分频器 .....	415
20.5	寄存器堆和存储器映射描述 .....	417
20.5.1	发送数据寄存器(SPI_I2S_TXREG) .....	417
20.5.2	接收数据寄存器(SPI_I2S_RXREG).....	418
20.5.3	当前状态寄存器(SPI_I2S_CSTAT) .....	418
20.5.4	中断状态寄存器 (SPI_I2S_INTSTAT) .....	419
20.5.5	中断使能寄存器(SPI_I2S_INTEN).....	421
20.5.6	中断清除寄存器(SPI_I2S_INTCLR) .....	423
20.5.7	全局控制寄存器(SPI_I2S_GCTL).....	424
20.5.8	通用控制寄存器(SPI_I2S_CCTL).....	427
20.5.9	波特率发生器 (SPI_I2S_SPBRG) .....	428
20.5.10	接收数据个数寄存器(SPI_I2S_RXDNR).....	429
20.5.11	从机片选寄存器(SPI_I2S_NSSR) .....	429

20.5.12	数据控制寄存器(SPI_I2S_EXTCTL).....	430
20.5.13	I2S 配置寄存器(SPI_I2S_I2SCFGR).....	431
21	集成电路 I2C 接口(I2C).....	433
21.1	I2C 简介 .....	433
21.2	I2C 功能框图 .....	433
21.3	I2C 主要特征 .....	434
21.4	引脚定义及内部信号 .....	434
21.5	中断.....	434
21.6	DMA 通信 .....	435
21.6.1	利用 DMA 发送 .....	435
21.6.2	利用 DMA 接收 .....	435
21.7	I2C 协议 .....	435
21.7.1	起始和停止条件 .....	435
21.7.2	从机寻址协议 .....	436
21.7.3	发送和接收协议 .....	437
21.7.4	发送缓冲管理以及起始、停止和重复起始条件产生.....	439
21.7.5	多个主机仲裁 .....	440
21.7.6	时钟同步 .....	441
21.7.7	SCL 配置 .....	442
21.8	I2C 工作模式 .....	443
21.8.1	从模式.....	443
21.8.2	主模式.....	444
21.8.3	I2C 中止传输.....	446
21.9	I2C 寄存器描述 .....	447
21.9.1	I2C 控制寄存器(I2C_CR) .....	448
21.9.2	I2C 目标地址寄存器(I2C_TAR).....	451
21.9.3	I2C 从机地址寄存器(I2C_SAR) .....	452
21.9.4	I2C 数据命令寄存器(I2C_DR) .....	452
21.9.5	标准模式 I2C 时钟高电平计数寄存器(I2C_SSHR).....	452
21.9.6	标准模式 I2C 时钟低电平计数寄存器(I2C_SSLR).....	453
21.9.7	快速模式 I2C 时钟高电平计数寄存器(I2C_FSHR) .....	453
21.9.8	快速模式 I2C 时钟低电平计数寄存器(I2C_FSLR) .....	453
21.9.9	I2C 中断状态寄存器(I2C_ISR) .....	454
21.9.10	I2C 中断屏蔽寄存器(I2C_IMR).....	454
21.9.11	I2C RAW 中断寄存器(I2C_RAWISR) .....	455
21.9.12	I2C 接收阈值(I2C_RXTLR) .....	457
21.9.13	I2C 发送阈值(I2C_TXTLR).....	457
21.9.14	I2C 组合和独立中断清除寄存器(I2C_ICR).....	458
21.9.15	I2C 清除 RX_UNDER 中断寄存器(I2C_RX_UNDER).....	458
21.9.16	I2C 清除 RX_OVER 中断寄存器(I2C_RX_OVER) .....	458
21.9.17	I2C 清除 TX_OVER 中断寄存器(I2C_TX_OVER) .....	459
21.9.18	I2C 清除 RD_REQ 中断寄存器(I2C_RD_REQ) .....	460
21.9.19	I2C 清除 TX_ABRT 中断寄存器(I2C_TX_ABRT) .....	460

21.9.20 I2C 清除 RX_DONE 中断寄存器(I2C_RX_DONE) .....	460
21.9.21 I2C 清除 ACTIVITY 中断寄存器(I2C_ACTIV) .....	461
21.9.22 I2C 清除 STOP_DET 中断寄存器(I2C_STOP) .....	461
21.9.23 I2C 清除 START_DET 中断寄存器(I2C_START) .....	461
21.9.24 I2C 清除 GEN_CALL 中断寄存器(I2C_GC) .....	462
21.9.25 I2C 使能寄存器(I2C_ENR) .....	462
21.9.26 I2C 状态寄存器(I2C_SR) .....	463
21.9.27 I2C 发送缓冲水平寄存器(I2C_TXFLR) .....	463
21.9.28 I2C 接收缓冲水平寄存器(I2C_RXFLR) .....	464
21.9.29 I2C SDA 保持时间寄存器(I2C_HOLD) .....	464
21.9.30 I2C DMA 控制寄存器(I2C_DMA) .....	465
21.9.31 I2C SDA 建立时间寄存器(I2C_SETUP) .....	465
21.9.32 I2C 广播呼叫 ACK 寄存器(I2C_GCR) .....	466
21.9.33 I2C 从机地址掩码寄存器(I2C_SLVMASK) .....	466
21.9.34 I2C 从机接收地址寄存器(I2C_SLVRCVADDR) .....	466
<b>22 控制器局域网(CAN) .....</b>	<b>467</b>
<b>22.1 简介 .....</b>	<b>467</b>
<b>22.2 功能框图 .....</b>	<b>467</b>
<b>22.2.1 接口管理逻辑 (IML) .....</b>	<b>469</b>
<b>22.2.2 发送缓冲器 (TX8) .....</b>	<b>469</b>
<b>22.2.3 接收缓冲器 (RX8,RXFIFO) .....</b>	<b>469</b>
<b>22.2.4 验收滤波器 (ACF) .....</b>	<b>469</b>
<b>22.2.5 位流处理器 (8SP) .....</b>	<b>469</b>
<b>22.2.6 位时序逻辑 (8TL) .....</b>	<b>469</b>
<b>22.2.7 错位管理逻辑 (EML) .....</b>	<b>469</b>
<b>22.3 主要特征 .....</b>	<b>469</b>
<b>22.4 中断 .....</b>	<b>470</b>
<b>22.5 功能描述 .....</b>	<b>471</b>
<b>22.5.1 CAN 工作模式 .....</b>	<b>471</b>
<b>22.5.2 Basic CAN 模式 .....</b>	<b>471</b>
<b>22.5.3 Peli CAN 模式 .....</b>	<b>473</b>
<b>22.5.4 发送处理 .....</b>	<b>477</b>
<b>22.5.5 接收管理 .....</b>	<b>477</b>
<b>22.5.6 标识符过滤 .....</b>	<b>478</b>
<b>22.5.7 报文存储 .....</b>	<b>485</b>
<b>22.5.8 出错管理 .....</b>	<b>488</b>
<b>22.5.9 位时序控制 .....</b>	<b>492</b>
<b>22.5.10 仲裁丢失 .....</b>	<b>492</b>
<b>22.6 CAN 寄存器描述 .....</b>	<b>493</b>
<b>22.6.1 CAN 模式寄存器 (CAN_MOD) .....</b>	<b>494</b>
<b>22.6.2 CAN 控制寄存器 (CAN_CR) .....</b>	<b>495</b>
<b>22.6.3 CAN 命令寄存器 (CAN_CMR) .....</b>	<b>496</b>
<b>22.6.4 CAN 状态寄存器 (CAN_SR) .....</b>	<b>498</b>

22.6.5 CAN 中断寄存器 (CAN_IR).....	499
22.6.6 CAN 中断使能寄存器 (CAN_IER).....	501
22.6.7 CAN 验收代码寄存器组 0(GROUP0_ACR).....	503
22.6.8 CAN 验收屏蔽寄存器组 0(GROUP0_AMR).....	503
22.6.9 CAN 总线定时 0(CAN_BTR0) .....	504
22.6.10 CAN 总线定时 1 (CAN_BTR1) .....	505
22.6.11 CAN 发送识别码寄存器 0(CAN_TXID0) .....	506
22.6.12 CAN 发送识别码寄存器 1 (CAN_TXID1) .....	506
22.6.13 CAN 仲裁丢失捕捉寄存器 (CAN_ALC).....	507
22.6.14 CAN 错误代码捕捉寄存器 (CAN_ECC) .....	509
22.6.15 CAN 错误报警限制寄存器 (CAN_EWLR) .....	511
22.6.16 CAN RX 错误计数寄存器 (CAN_RXERR) .....	512
22.6.17 CAN TX 错误计数寄存器 (CAN_TXERR) .....	513
22.6.18 CAN 发送帧信息寄存器 (CAN_SFF) .....	515
22.6.19 CAN 发送识别码寄存器 0(CAN_TXID0) .....	516
22.6.20 CAN 发送识别码寄存器 1(CAN_TXID1) .....	517
22.6.21 CAN 发送数据寄存器 0(CAN_TXDATA0).....	517
22.6.22 CAN 发送数据寄存器 1(CAN_TXDATA1).....	518
22.6.23 CAN 时钟分频寄存器 (CAN_CDR).....	519
22.6.24 CAN 滤波模式寄存器 0(CAN_AFM0) .....	519
22.6.25 CAN 滤波模式寄存器 1(CAN_AFM1) .....	520
22.6.26 CAN 滤波模式寄存器 2(CAN_AFM2) .....	521
22.6.27 CAN 滤波组使能寄存器 0(CAN_FGA0) .....	521
22.6.28 CAN 滤波组使能寄存器 1(CAN_FGA1) .....	522
22.6.29 CAN 滤波组使能寄存器 2(CAN_FGA2) .....	523
22.6.30 CAN 验收代码寄存器组 x(x = 1 ..19) (GROUPx_ACR) .....	523
22.6.31 CAN 验收屏蔽寄存器组 x(x = 1 ..19) (GROUPx_AMR) .....	524
<b>23 USB On-The-Go Full-Speed(OTG-FS).....</b>	<b>526</b>
23.1 OTG-FS 控制器主要功能特征.....	526
23.1.1 USB 功能特征 .....	526
23.1.2 额外 OTG 功能特征.....	526
23.2 OTG-FS 系统框图.....	527
23.3 OTG-FS 引脚说明 .....	527
23.4 OTG-FS 硬件外部配置.....	527
23.5 OTG-FS 软件编程接口.....	530
23.5.1 OTG-FS 数据传输方向 .....	530
23.5.2 缓存描述表 (Buffer Descriptor Table) .....	530
23.5.3 缓冲区描述表地址 .....	532
23.5.4 缓冲区描述符 (BD)格式 .....	532
23.5.5 USB 传输 .....	534
23.6 OTG 与 Host 模式操作方式.....	535
23.7 Host 模式操作示例 .....	535
23.7.1 启用主机模式并发现连接的设备 .....	535

23.7.2	与 USB 设备进行 Control 传输.....	536
23.7.3	与 USB 设备进行 Bulk OUT 传输.....	537
23.8	OTG 操作 .....	537
23.8.1	OTG 双角色 A 设备操作方式.....	537
23.8.2	OTG 双角色 B 设备操作方式.....	539
23.9	寄存器描述 .....	540
23.9.1	OTG 中断状态寄存器 (OTG_ISTAT) .....	541
23.9.2	OTG 中断控制寄存器 (OTG_ICTRL) .....	541
23.9.3	OTG 状态寄存器 (OTG_STAT) .....	542
23.9.4	OTG 控制寄存器 (OTG_CTRL) .....	543
23.9.5	中断状态寄存器 (INT_STAT) .....	544
23.9.6	中断使能寄存器 (INT_ENB) .....	545
23.9.7	错误中断状态寄存器 (ERR_STAT) .....	546
23.9.8	错误中断使能寄存器 (ERR_ENB) .....	547
23.9.9	状态寄存器 (STAT) .....	548
23.9.10	控制寄存器 (CTL) .....	549
23.9.11	地址寄存器 (ADDR) .....	550
23.9.12	缓冲区描述符表页寄存器 1 (BDT_PAGE_01) .....	551
23.9.13	低位帧数寄存器 (FRM_NUML) .....	551
23.9.14	高位帧数寄存器(FRM_NUMH).....	552
23.9.15	令牌寄存器 (TOKEN) .....	552
23.9.16	SOF 阈值寄存器 (SOF_THLD) .....	553
23.9.17	缓冲区描述符表页寄存器 2 (BDT_PAGE_02) .....	553
23.9.18	缓冲区描述符表页寄存器 3 (BDT_PAGE_03) .....	554
23.9.19	端点控制寄存器 (EP_CTL0~15) .....	554
24	以太网 MAC 控制器 (ETHERMAC) .....	556
24.1	ETHERMAC 简介 .....	556
24.2	ETHERMAC 主要特征 .....	556
24.2.1	架构框图 .....	556
24.2.2	功能管脚 .....	559
24.2.3	连接关系 .....	560
24.2.4	802.3 帧格式 .....	561
24.3	ETHERMAC 功能描述 .....	562
24.3.1	MAC 子控制器基本特征 .....	562
24.3.2	MAC 子控制器接收功能 .....	563
24.3.3	MAC 子控制器发送功能 .....	566
24.3.4	MAC 子控制器地址过滤 .....	571
24.3.5	MAC 子控制器回环模式 .....	573
24.3.6	DMA 子控制器基本特征 .....	573
24.3.7	DMA 子控制器接收功能 .....	575
24.3.8	DMA 子控制器发送功能 .....	577
24.3.9	DMA 子控制器初始化 .....	578
24.3.10	DMA 子控制器描述符 .....	579

24.4	ETHERMAC 中断 .....	588
24.5	ETHERMAC 寄存器.....	588
24.5.1	MAC 子控制器寄存器.....	591
24.5.2	DMA 子控制器寄存器.....	604
25	安全数字输入输出接口 (SDIO) .....	622
25.1	SDIO 简介 .....	622
25.1.1	SDIO 功能框图.....	623
25.2	功能描述.....	623
25.3	SD 存储卡系统概念 .....	624
25.3.1	Read-Write 属性 .....	624
25.3.2	电源电压 .....	624
25.3.3	卡容量.....	624
25.3.4	传输速度 .....	624
25.3.5	总线拓扑 .....	625
25.3.6	总线协议 .....	625
25.4	SD 存储卡功能描述 .....	626
25.4.1	简介 .....	626
25.4.2	卡识别模式.....	627
25.4.3	数据传输模式 .....	629
25.5	寄存器描述 .....	630
25.5.1	MMC_CTRL.....	631
25.5.2	MMC_IO .....	633
25.5.3	MMC_BYTECNTL .....	634
25.5.4	MMC_TR_BLOCKCNT .....	634
25.5.5	MMC_CRCCTL .....	635
25.5.6	CMD_CRC.....	636
25.5.7	DAT_CRCL.....	636
25.5.8	DAT_CRCH .....	637
25.5.9	MMC_PORT .....	637
25.5.10	MMC_INT_MASK .....	638
25.5.11	CLR_MMC_INT .....	639
25.5.12	MMC_CARDSEL .....	640
25.5.13	MMC_SIQ .....	641
25.5.14	MMC_IO_MBCTL .....	641
25.5.15	MMC_BLOCKCNT .....	643
25.5.16	MMC_TIMEOUTCNT .....	643
25.5.17	CMD_BUFX (X=0...15) .....	644
25.5.18	BUF_CTL.....	644
25.5.19	DATA_BUF .....	646
26	可变静态存储控制器 (FSMC) .....	648
26.1	设计概述 .....	648
26.1.1	框图 .....	648
26.1.2	接口描述 .....	648

26.1.3	特性 .....	649
26.2	功能描述 .....	650
26.2.1	FSMC 数据位宽 .....	650
26.2.2	FSMC 模式配置 .....	651
26.2.3	NOR Flash/SRAM/PSRAM 模式 .....	654
26.2.4	8080 协议模式 .....	656
26.2.5	6800 协议描述 .....	658
26.3	寄存器描述 .....	662
26.3.1	存储器屏蔽寄存器 (SMSKR0) .....	662
26.3.2	存储器时序寄存器(SMTMGR_SET0) .....	664
26.3.3	存储器时序寄存器(SMTMGR_SET1) .....	665
26.3.4	存储器时序寄存器(SMTMGR_SET2) .....	666
26.3.5	存储器控制寄存器(SMCTRLR) .....	667
27	模拟/数字转换(ADC) .....	669
27.1	简介 .....	669
27.2	功能框图 .....	669
27.3	主要特征 .....	670
27.4	中断 .....	670
27.5	DMA .....	671
27.6	功能描述 .....	671
27.6.1	时钟 .....	671
27.6.2	数据偏差校正 .....	671
27.6.3	数据对齐 .....	671
27.6.4	可编程分辨率 .....	671
27.6.5	可编程采样时间 .....	672
27.6.6	数据通道寄存器 .....	672
27.6.7	通道选择 .....	672
27.7	ADC 开关 .....	672
27.7.1	普通工作模式 .....	672
27.7.2	任意通道工作模式 .....	675
27.7.3	注入通道工作模式 .....	678
27.7.4	ADC 触发信号 .....	680
27.7.5	模拟看门狗 .....	680
27.7.6	内部温度传感器 .....	681
27.7.7	内部电压传感器 .....	681
27.8	寄存器描述 .....	681
27.8.1	A/D 数据寄存器 (ADC_ADDATA) .....	682
27.8.2	A/D 配置寄存器 (ADC_ADCFG) .....	683
27.8.3	A/D 控制寄存器 (ADC_ADCR) .....	685
27.8.4	A/D 通道选择寄存器 (ADC_ADCHS) .....	689
27.8.5	A/D 窗口比较寄存器 (ADC_ADCMPR) .....	691
27.8.6	A/D 状态寄存器 (ADC_ADSTA) .....	691
27.8.7	A/D 数据寄存器 (ADC_ADDR0~15) .....	693

27.8.8 A/D 扩展状态寄存器 (ADC_ADESTA_EXT) .....	694
27.8.9 A/D 任意通道通道选择寄存器 0 (ADC_CHANY0) .....	696
27.8.10 A/D 任意通道通道选择寄存器 1 (ADC_CHANY1) .....	697
27.8.11 A/D 任意通道配置寄存器 (ADC_ANY_CFG) .....	698
27.8.12 A/D 任意通道控制寄存器 (ADC_ANY_CR) .....	698
27.8.13 A/D 采样配置寄存器 (ADC_SMPR1) .....	702
27.8.14 A/D 采样配置寄存器 (ADC_SMPR2) .....	703
27.8.15 A/D 注入通道数据补偿 (ADC_JOFR0~3) .....	704
27.8.16 A/D 注入通道连续寄存器 (ADC_JSQR) .....	704
27.8.17 A/D 注入通道数据寄存器 (ADC_JADDATA) .....	706
27.8.18 A/D 注入通道数据寄存器 (ADC_JDR0~3) .....	707
<b>28 数字/模拟转换器 (DAC) .....</b>	<b>708</b>
28.1 简要介绍 .....	708
28.2 主要特征 .....	708
28.3 功能描述 .....	710
28.3.1 DAC 通道的使能 .....	710
28.3.2 使能 DAC 输出缓存 .....	710
28.3.3 DAC 输出电压 .....	710
28.3.4 DAC 触发源的选择 .....	710
28.3.5 DMA 请求 .....	711
28.3.6 DAC 数据格式 .....	711
28.3.7 带三角波生成的 DAC 转换 .....	712
28.3.8 带噪声生成的 DAC 转换 .....	713
28.3.9 DAC 转换 .....	713
28.3.10 双 DAC 通道转换 .....	713
28.4 DAC 寄存器 .....	715
28.4.1 DAC 控制寄存器 (DAC_CR) .....	716
28.4.2 DAC 软件触发寄存器 (DAC_SWTRIGR) .....	720
28.4.3 DAC 通道 1 的 12 位右对齐数据保持寄存器 (DAC_DHR12R1) .....	721
28.4.4 DAC 通道 1 的 12 位左对齐数据保持寄存器 (DAC_DHR12L1) .....	722
28.4.5 DAC 通道 1 的 8 位右对齐数据保持寄存器 (DAC_DHR8R1) .....	723
28.4.6 DAC 通道 2 的 12 位右对齐数据保持寄存器 (DAC_DHR12R2) .....	723
28.4.7 DAC 通道 2 的 12 位左对齐数据保持寄存器 (DAC_DHR12L2) .....	724
28.4.8 DAC 通道 2 的 8 位右对齐数据保持寄存器 (DAC_DHR8R2) .....	724
28.4.9 双 DAC 的 12 位右对齐数据保持寄存器 (DAC_DHR12RD) .....	725
28.4.10 双 DAC 的 12 位左对齐数据保持寄存器 (DAC_DHR12LD) .....	725
28.4.11 双 DAC 的 8 位右对齐数据保持寄存器 (DAC_DHR8RD) .....	726
28.4.12 DAC 通道 1 数据输出寄存器 (DAC_DOR1) .....	727
28.4.13 DAC 通道 2 数据输出寄存器 (DAC_DOR2) .....	727
<b>29 比较器(COMP) .....</b>	<b>728</b>
29.1 简介 .....	728
29.2 功能框图 .....	728
29.3 主要特征 .....	728

29.4 功能描述 .....	729
29.4.1 时钟复位 .....	729
29.4.2 比较器开关控制 .....	729
29.4.3 比较器输入和输出 .....	729
29.4.4 比较器通道选择 .....	729
29.4.5 中断和唤醒 .....	730
29.4.6 功耗模式 .....	730
29.4.7 比较器锁定机制 .....	730
29.4.8 迟滞电压 .....	730
29.5 比较器寄存器描述 .....	731
29.5.1 比较器控制状态寄存器(COMPx_CSR)(x=1,2) .....	731
29.5.2 比较器外部参考电压寄存器(COMP_CRV) .....	734
29.5.3 比较器轮询寄存器(COMPx_POLL)(x=1,2) .....	735
30 循环冗余校验计算单元(CRC) .....	737
30.1 CRC 简介 .....	737
30.2 CRC 功能框图 .....	737
30.3 主要特征 .....	737
30.4 功能描述 .....	738
30.5 寄存器描述 .....	738
30.5.1 CRC 数据寄存器(CRC_DR) .....	738
30.5.2 CRC 独立数据寄存器(CRC_IDR) .....	739
30.5.3 CRC 控制寄存器(CRC_CTRL) .....	739
30.5.4 CRC 中间数据寄存器(CRC_MIR) .....	740
31 调试支持 (DEBUG) .....	741
31.1 DEBUG 简介 .....	741
31.2 功能框图 .....	741
31.2.1 SWD 内部上拉与下拉 .....	742
31.2.2 SWD 调试端口 .....	742
31.3 ID 代码和锁定机制 .....	742
31.3.1 微控制器设备 ID 编码 .....	742
31.3.2 Cortex JEDEC-106 ID 代码 .....	743
31.4 功能描述 .....	743
31.4.1 SW 协议介绍 .....	743
31.4.2 SW 协议序列 .....	743
31.4.3 SW-DP 状态机 (Reset,idle states,ID code) .....	744
31.4.4 DP 和 AP 读 / 写访问 .....	744
31.4.5 SW-DP 寄存器 .....	745
31.4.6 SW-AP 寄存器 .....	745
31.5 MCU 调试模块 (MCUDBG) .....	746
31.5.1 低功耗模式的调试支持 .....	746
31.5.2 支持定时器、看门狗 .....	746
31.5.3 调试 MCU 配置寄存器 .....	746
31.6 DEBUG 寄存器描述 .....	746

31.6.1 DEBUG 控制寄存器 (DEBUG_CR).....	747
32    器件电子签名(DEVICE).....	749
32.1 DEVICE 简介.....	749
32.2 寄存器描述 .....	750
32.2.1 唯一标识码 (UID1) .....	750
32.2.2 唯一标识码 (UID2) .....	750
32.2.3 唯一标识码 (UID3) .....	751
32.2.4 唯一标识码 (UID4) .....	751

Preliminary

## 附图目录

图 1-1 系统架构框图 .....	2
图 2-1 ISP 方式编程流程图 .....	16
图 2-2 IAP 编程流程 .....	17
图 4-1 电源控制功能框图 .....	43
图 4-2 上电复位和掉电复位的波形图 .....	45
图 4-3 PVD 的阈值 .....	46
图 5-1 复位功能框图 .....	64
图 5-2 时钟树 .....	66
图 5-3 外部时钟 .....	67
图 5-4 晶振/陶瓷谐振器 .....	67
图 7-1 EXTI 结构框图 .....	122
图 8-1 DMA 功能框图 .....	132
图 10-1 CRS 功能框图 .....	152
图 10-2 CRS 计数器状态图 .....	156
图 11-1 标准 I/O 端口 .....	165
图 11-2 输入浮空/上拉/下拉配置 .....	168
图 11-3 输出配置 .....	169
图 11-4 复用功能配置 .....	169
图 12-1 TIM1 结构图 .....	181
图 12-2 时钟选择 .....	182
图 12-3 外部时钟模式 1 下的控制电路 .....	183
图 12-4 外部时钟模式 2 下的控制电路 .....	183
图 12-5 自动预装载 .....	184
图 12-6 递增计数模式 (UDIS=0) .....	184
图 12-7 递增计数模式 (UDIS=0 禁止产生更新事件) .....	184
图 12-8 递减计数模式 (UDIS=0) .....	185
图 12-9 递减计数模式 (UDIS=1 禁止产生更新事件) .....	185
图 12-10 中央计数模式 (UDIS=0) .....	186
图 12-11 中央计数模式 (UDIS=1 禁止产生更新事件) .....	186
图 12-12 复位模式的控制时序图 .....	186
图 12-13 门控模式下的控制时序图 .....	187
图 12-14 触发器模式下的控制时序图 .....	187
图 12-15 外部时钟模式 2+触发模式下的控制时序图 .....	188
图 12-16 定时器间的互联 .....	188
图 12-17 使用 TIM3 作为定时器 1 的预分频器 .....	189
图 12-18 使用 TIM1 使能 TIM2 .....	189
图 12-19 使用 TIM1 的更新事件启动 TIM2 .....	190
图 12-20 TIM1 的 TI1 同步启动两个 TIM1 和 TIM2 .....	190
图 12-21 TIM1 输入捕获结构图 .....	191
图 12-22 PWM 输入模式时序 .....	193
图 12-23 编码器模式下的计数器时序图 .....	194

图 12-24 IC1FP1 反相的编码器接口模式时序图.....	194
图 12-25 (TI1 异或输入) 输入捕获波形图 .....	194
图 12-26 霍尔传感器接口的实例 .....	196
图 12-27 捕获 / 比较的输出部分 .....	196
图 12-28 输出比较模式, OC1 信号在匹配时翻转 .....	198
图 12-29 边沿对齐递增计数时 PWM 模式 1 的波形.....	199
图 12-30 边沿对齐递减计数时 PWM 模式 1 的波形.....	199
图 12-31 中央对齐 PWM 模式 1 的波形 (CMS=11) .....	199
图 12-32 移相功能示意图.....	200
图 12-33 产生六步 PWM, 使用 COM 的例子 (OSSR = 1) .....	201
图 12-34 死区插入 .....	202
图 12-35 响应刹车的输出 (OISx=0, OISxN=0) .....	204
图 12-36 响应刹车的输出 (OISx=0, OISxN=1) .....	204
图 12-37 响应刹车的输出 (OISx=1, OISxN=0) .....	204
图 12-38 响应刹车的输出 (OISx=1, OISxN=1) .....	204
图 12-39 清除 TIM1 的 OCxREF .....	204
图 12-40 单脉冲模式.....	205
图 13-1 TIM3 结构图.....	243
图 13-2 时钟选择 .....	244
图 13-3 外部时钟模式 1 下的控制电路 .....	245
图 13-4 外部时钟模式 2 下的控制电路 .....	245
图 13-5 自动预装载.....	246
图 13-6 递增计数模式 (UDIS=0) .....	246
图 13-7 递增计数模式 (UDIS=0 禁止产生更新事件) .....	246
图 13-8 递减计数模式 (UDIS=0) .....	247
图 13-9 递减计数模式 (UDIS=1 禁止产生更新事件) .....	247
图 13-10 中央计数模式 (UDIS=0) .....	247
图 13-11 中央计数模式 (UDIS=1 禁止产生更新事件) .....	248
图 13-12 复位模式的控制电路图 .....	248
图 13-13 门控模式下的控制电路 .....	249
图 13-14 触发器模式下的控制电路.....	249
图 13-15 外部时钟模式 2+触发模式下的控制电路 .....	250
图 13-16 TIM3 输入捕获结构图.....	250
图 13-17 PWM 输入模式时序.....	252
图 13-18 编码器模式下的计数器时序图 .....	253
图 13-19 IC1FP1 反相的编码器接口模式时序图 .....	253
图 13-20 (TI1 异或输入) 输入捕获波形图 .....	254
图 13-21 霍尔传感器接口的实例 .....	255
图 13-22 捕获 / 比较的输出部分 .....	256
图 13-23 输出比较模式, 翻转 OC1.....	257
图 13-24 边沿对齐递增计数时 PWM 模式 1 的波形.....	258
图 13-25 边沿对齐递减计数时 PWM 模式 1 的波形.....	258
图 13-26 中央对齐 PWM 模式 1 的波形 (CMS=11) .....	259

图 13-27 单脉冲模式的例子 .....	259
图 14-1 TIM2 结构图.....	289
图 14-2 时钟选择 .....	290
图 14-3 外部时钟模式 1 下的控制电路 .....	291
图 14-4 外部时钟模式 2 下的控制电路 .....	291
图 14-5 自动预装载.....	292
图 14-6 递增计数模式 (UDIS=0) .....	292
图 14-7 递增计数模式 (UDIS=0 禁止产生更新事件) .....	292
图 14-8 递减计数模式 (UDIS=0) .....	293
图 14-9 递减计数模式 (UDIS=1 禁止产生更新事件) .....	293
图 14-10 中央计数模式 (UDIS=0) .....	293
图 14-11 中央计数模式 (UDIS=1 禁止产生更新事件) .....	294
图 14-12 复位模式的控制电路图 .....	294
图 14-13 门控模式下的控制电路 .....	295
图 14-14 触发器模式下的控制电路.....	295
图 14-15 外部时钟模式 2+触发模式下的控制电路 .....	296
图 14-16 TIM2 输入捕获结构图.....	296
图 14-17 PWM 输入模式时序.....	298
图 14-18 编码器模式下的计数器时序图 .....	299
图 14-19 IC1FP1 反相的编码器接口模式时序图 .....	299
图 14-20 (TI1 异或输入) 输入捕获波形图 .....	300
图 14-21 霍尔传感器接口的实例 .....	301
图 14-22 捕获 / 比较的输出部分 .....	302
图 14-23 输出比较模式, OC1 信号在匹配时翻转 .....	303
图 14-24 边沿对齐递增计数时 PWM 模式 1 的波形 .....	304
图 14-25 边沿对齐递减计数时 PWM 模式 1 的波形 .....	304
图 14-26 中央对齐 PWM 模式 1 的波形 (CMS=11) .....	305
图 14-27 单脉冲模式的例子 .....	305
图 15-1 TIM6/7 结构图.....	336
图 15-2 自动预装载.....	337
图 15-3 递增计数模式 (UDIS=0) .....	337
图 15-4 递增计数模式 (UDIS=1) .....	337
图 16-1 IWDG 结构框图 .....	344
图 16-2 IWDG 流程框图.....	345
图 17-1 WWDG 结构框图 .....	354
图 18-1 RTC 结构框图 .....	360
图 18-2 RTC 秒和闹钟波形图示例, PR = 0004, ALARM = 002 .....	363
图 18-3 RTC 溢出波形图示例, PR = 0004 .....	364
图 19-1 UART 功能框图.....	373
图 19-2 UART 时序 .....	376
图 19-3 RX 引脚采样方案 .....	377
图 19-4 发送时状态位变化 .....	379
图 19-5 UART 奇偶校验方框图 .....	383

图 19-6 UART 采样 NACK 信号方框图 .....	384
图 19-7 普通模式下 IrDA 发送和接收图 .....	385
图 20-1 SPI_I2S 功能框图 .....	403
图 20-2 单主和单从应用 .....	404
图 20-3 数据时钟时序图 .....	406
图 20-4 飞利浦标准示意图 .....	412
图 20-5 MSB 对齐标准示意图 .....	412
图 20-6 LSB 对齐标准示意图 .....	413
图 20-7 PCM 标准示意图 .....	413
图 20-8 I2S 从模式 .....	414
图 20-9 I2S 主模式 .....	414
图 20-10 I2S 时钟预分频器示意图 .....	416
图 21-1 I2C 功能框图 .....	433
图 21-2 I2C 中断机制 .....	435
图 21-3 起始和停止条件 .....	436
图 21-4 7 位的地址格式 .....	436
图 21-5 10 位的地址格式 .....	437
图 21-6 主发送协议 .....	438
图 21-7 主接收协议 .....	438
图 21-8 带 Restart (SR) 信号的主发送和接收协议 .....	438
图 21-9 起始字节传输 .....	439
图 21-10 DR 寄存器 .....	439
图 21-11 主发送, TX FIFO 为空 .....	440
图 21-12 主接收, TX FIFO 为空 .....	440
图 21-13 多个主机仲裁 .....	441
图 21-14 多个主机时钟同步 .....	441
图 21-15 SCL master clock generation .....	442
图 21-16 SCL master clock synchronization .....	443
图 21-17 I2C 接口主机流程图 .....	446
图 22-1 CAN 网络拓扑结构 .....	467
图 22-2 CAN 结构方框图 .....	468
图 22-3 CAN 标识符接收示例 .....	479
图 22-4 接收标准结构信息时的单个滤波器配置 .....	480
图 22-5 单滤波器配置, 接收扩展帧信息 .....	481
图 22-6 接收标准结构信息时的双个滤波器配置 .....	482
图 22-7 双滤波器配置, 接收扩展帧信息 .....	483
图 22-8 标准帧和扩展帧格式配置在发送缓冲器的列表 .....	487
图 22-9 RXFIFO 中的信息存储举例 .....	488
图 22-10 错误码捕捉功能举例 .....	489
图 22-11 仲裁丢失解释举例 .....	493
图 23-1 OTG-FS 功能框图 .....	527
图 23-2 仅主机框图 .....	528
图 23-3 仅从设备框图 .....	529

图 23-4 双角色设备框图 .....	530
图 23-5 缓冲区描述符表 .....	532
图 23-6 令牌传输 .....	534
图 23-7 OTG 双角色 A 设备操作方式 .....	538
图 23-8 OTG 双角色 B 设备操作方式 .....	539
图 24-1 ETHERMAC 架构框图 .....	557
图 24-2 MII 接口与 Device PHY 连接图 .....	561
图 24-3 RMII 接口与 Device PHY 连接图 .....	561
图 24-4 以太网帧格式 .....	562
图 24-5 MII/RMII 接收时序图 .....	565
图 24-6 无错误时接收时序图 .....	565
图 24-7 有错误时接收时序图 .....	565
图 24-8 假载波指示接收图 .....	566
图 24-9 MII/RMII 发送时序 .....	570
图 24-10 无冲突时发送时序图 .....	570
图 24-11 有冲突时发送时序图 .....	571
图 24-12 描述符结构 .....	574
图 25-1 SDIO 框图 .....	623
图 25-2 SDIO“无响应”和“无数据”操作 .....	625
图 25-3 SDIO（多）数据块读操作 .....	626
图 25-4 SDIO（多）数据块写操作 .....	626
图 25-5 SDIO 连续读操作 .....	626
图 25-6 SDIO 连续写操作 .....	626
图 25-7 SDIO 流程图 .....	628
图 26-1 FSMC 框图 .....	648
图 26-2 NOR FLASH 控制器 .....	649
图 26-3 NOR FLASH 控制器 .....	650
图 26-4 NOR FLASH 16bit 示例 .....	652
图 26-5 SRAM 16bit 示例 .....	653
图 26-6 SRAM 8bit 示例 .....	653
图 26-7 复用 NOR Flash/SRAM/PSRAM 接口 .....	654
图 26-8 NOR FLASH Write .....	655
图 26-9 NOR FLASH Read .....	655
图 26-10 8080 Reg Write .....	657
图 26-11 8080 Reg Read .....	657
图 26-12 8080 Memory Write .....	658
图 26-13 8080 Memory Read .....	658
图 26-14 6800 Reg Write .....	659
图 26-15 6800 Reg Read .....	660
图 26-16 6800 Memory Write .....	661
图 26-17 6800 Memory Read .....	661
图 27-1 ADC 系统框图 .....	669
图 27-2 ADC 框图 .....	670

图 27-3 数据对齐方式 .....	671
图 27-4 单次转换模式时序图 .....	673
图 27-5 单周期扫描下使能通道转换时序图 (通道方向从低到高) .....	674
图 27-6 单周期扫描下使能通道转换时序图 (通道方向从高到低) .....	674
图 27-7 连续扫描模式使能通道转换时序图 (通道方向由低到高) .....	675
图 27-8 连续扫描模式使能通道转换时序图 (通道方向由高到低) .....	675
图 27-9 单次转换模式, 通道转换时序图 .....	676
图 27-10 单周期扫描下通道转换时序图 .....	677
图 27-11 连续扫描模式, 通道转换时序图 .....	678
图 27-12 连续扫描模式, 动态更新配置时序图 .....	678
图 27-13 自动注入模式, 通道转换时序图 .....	679
图 27-14 任意通道转换时注入事件转换通道时序图 .....	679
图 27-15 注入通道转换时任意事件转换通道发生时序图 .....	680
图 28-1 DAC 通道功能框图 .....	709
图 28-2 DAC 单通道模式的数据保持寄存器 .....	711
图 28-3 DAC 双通道模式的数据保持寄存器 .....	712
图 28-4 DAC 三角波生成 .....	712
图 28-5 带三角波生成的 DAC 转换 (使能软件触发) .....	712
图 28-6 DAC LFSR 寄存器算法 .....	713
图 28-7 带 LFSR 波形生成的 DAC 转换 (使能软件触发) .....	713
图 28-8 TEN=0 触发关闭时转换的时间示意图 .....	713
图 29-1 比较器框图 .....	728
图 29-2 比较器的迟滞 .....	731
图 30-1 CRC 计算单元框图 .....	737
图 31-1 MM32 系列级别和 CPU 级别的调试框 .....	741

## 表格目录

表 1-1 存储器映像.....	4
表 1-2 启动模式.....	7
表 2-1 Flash 存储空间.....	9
表 2-2 信息块.....	10
表 2-3 选项字节组织结构.....	10
表 2-4 USER 的位含义 .....	12
表 2-5 区块读保护字节含义 .....	13
表 2-6 Latency 设置关系.....	14
表 2-7 编程方式 .....	14
表 2-8 保护设置的状态变化 .....	18
表 2-9 Flash 读保护状态 .....	26
表 2-10 Flash 解除读保护状态.....	26
表 2-11 写保护区域.....	27
表 2-12 FLASH 寄存器概览 .....	28
表 3-1 Cache 寄存器概览 .....	38
表 4-1 低功耗模式列表 .....	47
表 4-2 低功耗运行模式 .....	49
表 4-3 SLEEP NOW 模式 .....	49
表 4-4 SLEEP ON EXIT 模式.....	50
表 4-5 Low Power SLEEP NOW 模式.....	50
表 4-6 Low Power SLEEP ON EXIT 模式.....	51
表 4-7 停机模式 .....	52
表 4-8 深度停机模式.....	53
表 4-9 待机模式 .....	54
表 4-10 电源控制寄存器概览 .....	55
表 5-1MCO 与时钟源对应关系.....	72
表 5-2 RCC 寄存器概览.....	73
表 6-1 SYSCFG 寄存器概览 .....	113
表 7-1 异常向量表 .....	123
表 7-2 中断向量表 .....	123
表 7-3 EXTI 触发源.....	125
表 7-4 EXTI 寄存器概览.....	126
表 8-1 DMA 中断请求 .....	133
表 8-2 各个通道的 DMA 请求一览 .....	133
表 8-3 可配置的数据传输宽度和大小端操作(当 PINC=MINC=1), 传输数目为 4 .....	136
表 8-4 DMA 寄存器概览 .....	139
表 9-1BKP 寄存器概览 .....	147
表 10-1 中断控制位.....	152
表 10-2 CRS 低功耗模式下的影响 .....	158
表 10-3 CRS 寄存器描述概览 .....	158
表 11-1 端口位配置表 .....	165
表 11-2 高级定时器 TIM1 .....	170

表 11-3 通用/基本定时器 TIM2/3/14/16/17 .....	170
表 11-4 UART .....	171
表 11-5 SPI .....	171
表 11-6 I2C .....	171
表 11-7 CAN .....	172
表 11-8 ADC .....	172
表 11-9 其他 I/O 引脚 .....	172
表 11-10 GPIO 寄存器概览 .....	172
表 12-1 数字滤波器长度与 ICxF 的对应关系表 .....	191
表 12-2 计数方向与编码器信号的关系 .....	193
表 12-3 死区时间计算 .....	201
表 12-4 当 MOE=1, OSSI=0/1, OSSR=0 时: .....	202
表 12-5 当 MOE=1, OSSI=0/1, OSSR=1 时: .....	202
表 12-6 当 MOE=0, OSSI=0, OSSR=0/1 时: .....	202
表 12-7 当 MOE=0, OSSI=0, OSSR=0/1 时: .....	203
表 12-8 TIM1 寄存器概览 .....	206
表 12-9 TIM1 内部触发连接 .....	215
表 12-10 输入模式下, ICx 的极性选择如下表: .....	232
表 13-1 数字滤波器长度与 ICxF 的对应关系表 .....	250
表 13-2 计数方向与编码器信号的关系 .....	253
表 13-3 TIM3 寄存器概览 .....	261
表 13-4 TIM2/3 内部触发连接 .....	268
表 13-5 输入模式下, ICx 的极性选择如下表: .....	284
表 14-1 数字滤波器长度与 ICxF 的对应关系表 .....	296
表 14-2 计数方向与编码器信号的关系 .....	298
表 14-3 TIM2 寄存器概览 .....	307
表 14-4 TIM2/3 内部触发连接 .....	314
表 14-5 输入模式下, ICx 的极性选择如下表: .....	330
表 16-1 IWDG 超时时间 (40kHz 的输入时钟 (LSI)) .....	346
表 16-2 IWDG 寄存器概览 .....	347
表 17-1 超时时间 .....	356
表 17-2 WWDG 寄存器概览 .....	357
表 18-1 RTC 寄存器概览 .....	364
表 19-1 UART 中断请求 .....	374
表 19-2 UART 寄存器概览 .....	385
表 20-1 SPI 状态 .....	408
表 20-2 波特率公式 .....	410
表 20-3 I2S 状态 .....	415
表 20-4 I2S 音频采样率误差表 .....	416
表 20-5 SPI_I2S 寄存器概览 .....	417
表 21-1 中断位的置位和清除 .....	434
表 21-2 I2C 首字节 .....	437
表 21-3 I2C 寄存器描述概览 .....	447

表 21-4 DISSLAVE(bit6)和 MASTER(bit0)配置 .....	451
表 22-1 Basic CAN 模式寄存器权限分配表 .....	472
表 22-2 BasicCAN 模式里的 RX 和 TX 缓冲器 .....	485
表 22-3 接收时可能出现的错误 .....	490
表 22-4 发送时可能出现的错误 .....	490
表 22-5 CAN 寄存器概览 .....	493
表 23-1 BD 结构 .....	533
表 23-2 BD 结构说明 .....	533
表 23-3 OTG 双角色 A 设备操作方式 .....	538
表 23-4 OTG 双角色 B 设备操作方式 .....	539
表 23-5 USB 寄存器概览 .....	540
表 24-1 ETHERMAC 管脚功能表 .....	559
表 24-2 ETHERMAC 寄存器概览表 .....	588
表 25-1 SDIO 寄存器概览 .....	630
表 25-2 MMC_IO 寄存器[7: 0]的详细描述 .....	646
表 25-3 MMC_IO_MBCTL[2:0]和 MMC_IO[7:6]的详细描述表 .....	646
表 26-1 接口描述 .....	648
表 26-2 存储器位宽 .....	650
表 26-3 NOR FLASH 写时序说明 .....	654
表 26-4 NOR FLASH 读时序说明 .....	656
表 26-5 8080 协议接口 .....	656
表 26-6 8080 地址说明 .....	656
表 26-7 8080 读时序说明 .....	657
表 26-8 8080 读时序说明 .....	658
表 26-9 6800 协议接口 .....	658
表 26-10 6800 地址说明 .....	658
表 26-11 6800 读时序说明 .....	659
表 26-12 6800 读时序说明 .....	661
表 26-13 寄存器映射表 .....	662
表 27-1 寄存器概览 .....	681
表 28-1 DAC 管脚 .....	709
表 28-2 外部触发 .....	710
表 28-3 DAC 寄存器概览 .....	715
表 29-1 COMP 寄存器概览 .....	731
表 30-1 CRC 寄存器概览 .....	738
表 31-1 SWJ 调试端口管脚 .....	742
表 31-2 设备 ID .....	742
表 31-3 ID 编码 .....	743
表 31-4 8bit 请求位包 .....	743
表 31-5 3bit 应答包 .....	744
表 31-6 33bit 请求位包 .....	744
表 31-7 SW_DP 寄存器 .....	745
表 31-8 DBG 寄存器概览 .....	746

表 32-1 存储器容量寄存器概览 ..... 750

Preliminary

# 1 系统和存储器架构

## 1.1 系统架构简介

MM32F3270 系列是基于 ARM-Cortex-M3 处理器开发的 32 位微控制器产品，它同时具备了高性能和低功耗的特点。MM32F3270 系列采用矩阵总线结构，该矩阵包括五个 AHB 主机：CPU，DMA1，DMA2，USB 和 Ethernet，从机分别是 SRAM、闪存存储、FSMC、AHB 总线（含 AHB 到 APB 的总线桥）以及连接在 APB 总线的各种设备。

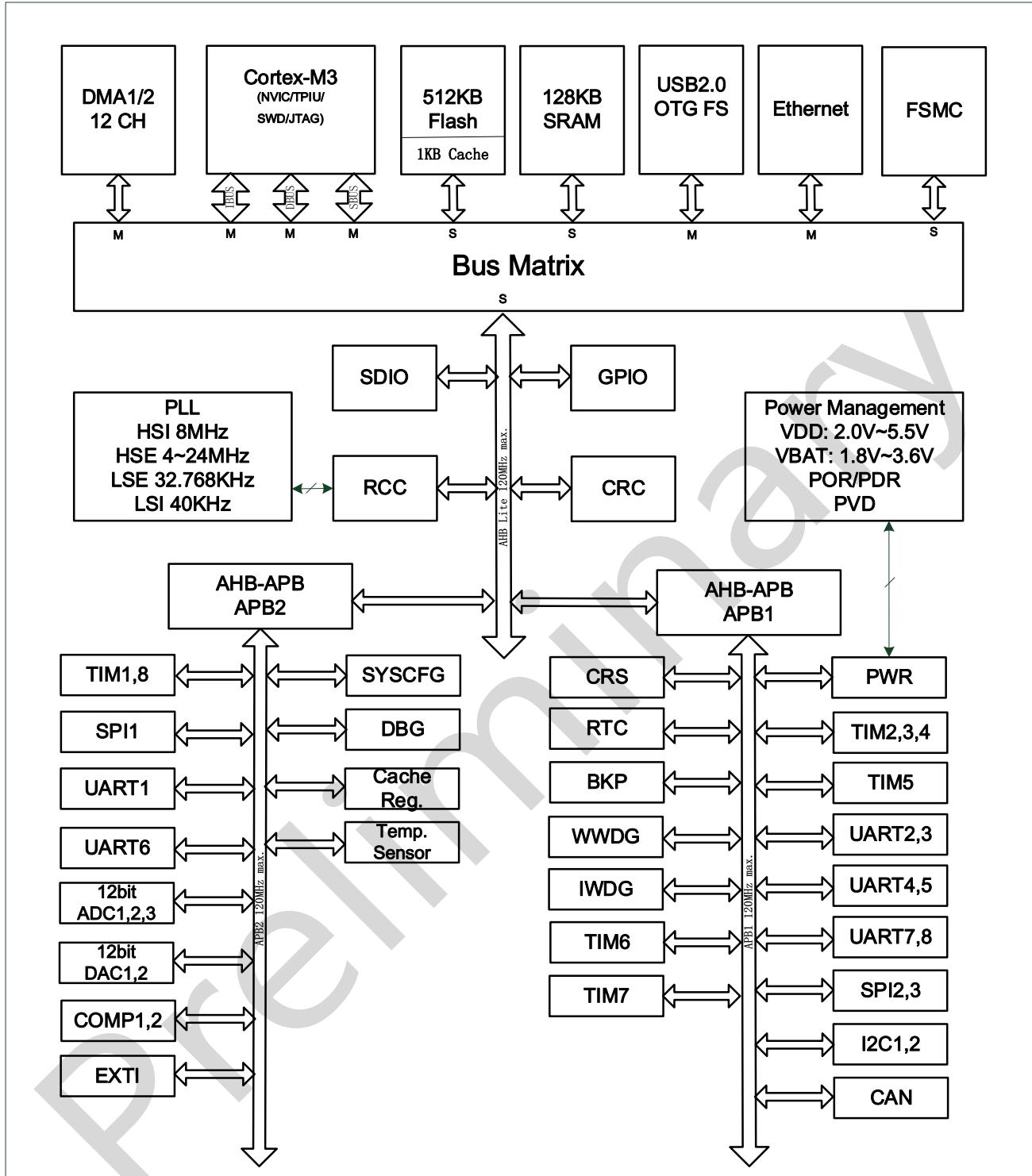


图 1-1 系统架构框图

### **1.1.1 ICode 总线 (IBUS)**

该总线将 CPU 内核的指令总线与闪存指令接口相连接。指令预取在此总线上完成。

### **1.1.2 DCode 总线 (DBUS)**

该总线将 CPU 内核 DCode 总线与闪存存储器的数据接口相连接 (常量加载和调试访问)。

### **1.1.3 系统总线 (SBUS)**

系统总线作用是连接 CPU 内核和总线矩阵，从而达到数据传输的作用。CPU 和 DMA 作为主机驱动总线，总线矩阵会协调 CPU 内核和 DMA 之间的访问。

### **1.1.4 DMA 总线**

DMA 总线作用是连接 DMA 和总线矩阵，从而达到数据传输的作用，总线矩阵协调着主机 DMA 到从机 SRAM，闪存和连接在 APB 线上的各种外设的访问控制。

### **1.1.5 总线矩阵 (BUS Matrix)**

总线矩阵管理着内核系统总线与 DMA 总线的访问仲裁，总线矩阵由主模块总线及从模块总线组成。AHB 外设通过总线矩阵与系统总线相连，允许 DMA 访问。

### **1.1.6 AHB 到 APB 桥 (AHB-APB APBx)**

AHB 到 APB 桥在 AHB 与 APB 总线间提供同步连接。在每次复位之后，所有的外设时钟都关闭 (除了 SRAM 及 Flash 外)。在用一个外设前，用户必须打开相应的 RCC\_AHBXENR 或 RCC\_APBXENR 寄存器中时钟使能位。

注：当对 APB 寄存器进行 8 位或者 16 位访问时，该访问会被自动转换成 32 位的访问：桥会自动将 16 位或者 8 位的数据扩展以配合 32 位的宽度。

## **1.2 存储器介绍**

程序存储器、数据存储器、寄存器和 I/O 接口都位于相同的存储器地址空间（线性 4GB 的地址空间），只是在不同的地址范围。为了降低不同器件供应商重复应用软件的负载度，提供了存储器映射。4GB 的地址空间被分为 8 块，每块为 512MB，分配给片上存储器和外设的存储器空间为固定的，不可更改，其余的地址空间为保留的地址空间，可由芯片供应商定义使用。

### **1.2.1 存储器映像和寄存器编址**

存储器映像请参考各外设对应章节的存储器映像图

表 1-1 存储器映像

总线	编址范围	大小	外设
FLASH	0x0000 0000 - 0x0007 FFFF	512 KB	根据 BOOT0/1 引脚的电平可映射到片内 FLASH 存储区、SRAM 或系统存储区中的一个
	0x0008 0000 - 0x07FF FFFF	~127 MB	Reserved
	0x0800 0000 - 0x0807 FFFF	512 KB	片内 FLASH 存储器
	0x0808 0000 - 0x080F FFFF	512 KB	Reserved
	0x0810 0000 - 0x0810 0FFF	4 KB	Reserved
	0x0810 1000 - 0x0FFF FFFF	~127 MB	Reserved
	0x1000 0000 - 0x1FFD FFFF	~255 MB	Reserved
	0x1FFE 0000 - 0x1FFE 0FFF	4 KB	Reserved
	0x1FFE 1000 - 0x1FFE 1FFF	4 KB	Security memory
	0x1FFE 2000 - 0x1FFF E7FF	114 KB	Reserved
	0x1FFF E800 - 0x1FFF F7FF	4 KB	系统存储区
	0x1FFF F800 - 0x1FFF F9FF	0.5 KB	Option bytes
SRAM	0x1FFF FA00 - 0x1FFF FFFF	1.5 KB	Reserved
	0x2000 0000 - 0x2000 3FFF	16 KB	SRAM-2
	0x2000 4000 - 0x2001 FFFF	112 KB	SRAM-1
AHB3	0x2002 0000 - 0x3FFF FFFF	~511 MB	Reserved
	0x6000 0000 - 0x63FF FFFF	64 MB	FSMC Bank
	0x6400 0000 - 0x67FF FFFF	64 MB	FSMC Bank
	0x6800 0000 - 0x6BFF FFFF	64 MB	FSMC Bank
	0x6C00 0000 - 0x6FFF FFFF	64 MB	FSMC Bank
	0x7000 0000 - 0x9FFF FFFF	768 MB	Reserved
	0xA000 0000 - 0xA000 0FFF	4 KB	FSMC Register
	0xA000 1000 - 0xA000 13FF	1 KB	Reserved

总线	编址范围	大小	外设
AHB2	0x5000 0000 - 0x5003 FFFF	256 KB	USB OTG FS
	0x5006 0000 - 0x5006 03FF	1 KB	Reserved
	0x5006 0800 - 0x5006 0BFF	1 KB	Reserved
AHB1	0x4002 0000 - 0x4002 03FF	1 KB	DMA1
	0x4002 0400 - 0x4002 07FF	1 KB	DMA2
	0x4002 0800 - 0x4002 0FFF	2 KB	Reserved
	0x4002 1000 - 0x4002 13FF	1 KB	RCC
	0x4002 1400 - 0x4002 1FFF	3 KB	Reserved
	0x4002 2000 - 0x4002 23FF	1 KB	Flash memory interface
	0x4002 2400 - 0x4002 2FFF	3 KB	Reserved
	0x4002 3000 - 0x4002 33FF	1 KB	CRC
	0x4002 3400 - 0x4002 7FFF	19 KB	Reserved
	0x4002 8000 - 0x4002 9FFF	8 KB	Ethernet
	0x4002 A000 - 0x4003 FFFF	88 KB	Reserved
	0x4004 0000 - 0x4004 03FF	1 KB	Port A
	0x4004 0400 - 0x4004 07FF	1 KB	Port B
APB2	0x4004 0800 - 0x4004 0BFF	1 KB	Port C
	0x4004 0C00 - 0x4004 0FFF	1 KB	Port D
	0x4004 1000 - 0x4004 13FF	1 KB	Port E
	0x4004 1400 - 0x4004 17FF	1 KB	Port F
	0x4004 1800 - 0x4004 1BFF	1 KB	Port G
	0x4004 1C00 - 0x4004 1FFF	1 KB	Port H
	0x4004 1C00 - 0x47FF FFFF	~127 MB	Reserved
	0x4001 0000 - 0x4001 03FF	1 KB	SYSCFG
	0x4001 0400 - 0x4001 07FF	1 KB	EXTI
	0x4001 0800 - 0x4001 23FF	7 KB	Reserved
	0x4001 2400 - 0x4001 27FF	1 KB	ADC1
	0x4001 2800 - 0x4001 2BFF	1 KB	ADC2
	0x4001 2C00 - 0x4001 2FFF	1 KB	TIM1

总线	编址范围	大小	外设
APB1	0x4001 3000 - 0x4001 33FF	1 KB	SPI1
	0x4001 3400 - 0x4001 37FF	1 KB	TIM8
	0x4001 3800 - 0x4001 3BFF	1 KB	UART1
	0x4001 3C00 - 0x4001 3FFF	1 KB	UART6
	0x4001 4000 - 0x4001 43FF	1 KB	COMP
	0x4001 4400 - 0x4001 4BFF	2 KB	Reserved
	0x4001 4C00 - 0x4001 4FFF	1 KB	ADC3
	0x4001 5000 - 0x4001 5FFF	4 KB	Reserved
	0x4001 6000 - 0x4001 63FF	1 KB	Cache Register
	0x4001 6400 - 0x4001 7FFF	7 KB	Reserved
	0x4001 8000 - 0x4001 83FF	1 KB	SDIO
	0x4001 8400 - 0x4001 FFFF	31 KB	Reserved
	0x4000 0000 - 0x4000 03FF	1 KB	TIM2
	0x4000 0400 - 0x4000 07FF	1 KB	TIM3
	0x4000 0800 - 0x4000 0BFF	1 KB	TIM4
	0x4000 0C00 - 0x4000 0FFF	1 KB	TIM5
	0x4000 1000 - 0x4000 13FF	1 KB	TIM6
	0x4000 1400 - 0x4000 17FF	1 KB	TIM7
	0x4000 1800 - 0x4000 27FF	4 KB	Reserved
	0x4000 2800 - 0x4000 2BFF	1 KB	RTC_BKP
	0x4000 2C00 - 0x4000 2FFF	1 KB	WWDG
	0x4000 3000 - 0x4000 33FF	1 KB	IWDG
	0x4000 3400 - 0x4000 37FF	1 KB	Reserved
	0x4000 3800 - 0x4000 3BFF	1 KB	SPI2
	0x4000 3C00 - 0x4000 3FFF	1 KB	SPI3
	0x4000 4000 - 0x4000 43FF	1 KB	Reserved
	0x4000 4400 - 0x4000 47FF	1 KB	UART2
	0x4000 4800 - 0x4000 4BFF	1 KB	UART3
	0x4000 4C00 - 0x4000 4FFF	1 KB	UART4
	0x4000 5000 - 0x4000 53FF	1 KB	UART5
	0x4000 5400 - 0x4000 57FF	1 KB	I2C1

总线	编址范围	大小	外设
	0x4000 5800 - 0x4000 5BFF	1 KB	I2C2
	0x4000 5C00 - 0x4000 63FF	2 KB	Reserved
	0x4000 6400 - 0x4000 67FF	1 KB	CAN
	0x4000 6800 - 0x4000 6BFF	1 KB	Reserved
	0x4000 6C00 - 0x4000 6FFF	1 KB	CRS
	0x4000 7000 - 0x4000 73FF	1 KB	PWR
	0x4000 7400 - 0x4000 77FF	1 KB	DAC
	0x4000 7800 - 0x4000 7BFF	1 KB	UART7
	0x4000 7C00 - 0x4000 7FFF	1 KB	UART8

## 1.2.2 内置的 SRAM

内置最大可到 128K 字节的静态 SRAM。它可以以字节 (8 位)、半字 (16 位) 或字 (32 位) 进行访问。SRAM 起始地址为 0x2000 0000。

SRAM 可以被 CPU 或者 DMA 用最快的系统时钟且不插入任何等待进行访问。

## 1.2.3 闪存存储器概述

闪存存储器分为两个存储区域：

- 由应用数据和用户数据区组成的主闪存存储块。
- 由选项字节和系统存储器组成的信息块：
  - 选项字节 (Option bytes)：包括硬件和存储保护用户配置选项。
  - 系统存储器 (System memory)：boot loader 代码。

闪存接口基于 AHB 协议执行指令和数据存取。闪存接口的预取缓冲功能可加速 CPU 执行代码的速度。

## 1.2.4 启动配置

在芯片中,可通过 BOOT0 及 BOOT1 脚的配置选择三种不同的启动模式,如下表所示:

表 1-2 启动模式

启动模式	启动模式引脚选择	
	BOOT1	BOOT0
主闪存存储器	x	0
系统存储器	0	1

内置 SRAM	1	1
---------	---	---

用户通过设置 **BOOT0** 引脚值和 **BOOT1** 位来选择三种启动模式，在器件复位后，在 **SYSCLK** 的第 4 个上升沿锁存 **BOOT1** 和 **BOOT0** 的引脚值，芯片根据 **BOOT1** 位和 **BOOT0** 的引脚值，从而确定启动模式。其中在待机模式中需要保持启动模式引脚的设置值，因为，每一次从待机模式唤醒时，CPU 会重新采样 **BOOT1** 和 **BOOT0** 的值来重新确定启动模式。

器件复位后，CPU 先从 0x0000 0000 地址开始获取栈顶值，再从 0x0000 0004 地址获取引导代码 的基地址，并且从基地址开始执行程序。

启动模式主要分为三种： 主闪存存储器， 系统存储器和内置 SRAM。

**主闪存存储器启动 ( $\text{BOOT0}=0$ )：** 主闪存存储器的起始地址是 0x0800 0000，当其被选为启动模式时，被映射到启动存储空间 (0x0000 0000)，但是闪存存储器的内容依旧可以从起始地址 (0x0800 0000) 访问，即当主闪存存储器被选为启动模式，启动地址和起始地址都可以访问闪存存储器。

**系统存储器启动 ( $\text{BOOT1}=0 \& \text{BOOT0}=1$ )：** 系统存储器的起始地址是 0x1FFF F400，当其被选为启动模式时，被映射到启动存储空间 (0x0000 0000)，但是系统存储器的内容依旧可以从起始地址 (0x1FFF F400) 访问，即当系统存储器被选为启动模式，启动地址和起始地址都可以访问系统存储器。

**内置 SRAM 启动 ( $\text{BOOT1}=1 \& \text{BOOT0}=1$ )：** 内置 SRAM 的起始地址是 0x2000 0000，当其被选为启动模式时，被映射到启动存储空间 (0x0000 0000)，但是内置 SRAM 的内容依旧可以从起始地址(0x2000 0000)访问，即当内置 SRAM 被选为启动模式，启动地址和起始地址都可以访问内置 SRAM.

## 1.2.5 引导程序

出厂后引导程序存放在系统存储器中，可以通过串口进行 ISP 编程。

## 2 嵌入式闪存 (Embedded FLASH)

### 2.1 简介

嵌入式闪存支持高达 512K Bytes 的片内 Main Flash，还提供了读保护块，选项字节块与系统启动块（支持芯片 Boot 引导），还有保留的保密空间，提供了特殊应用的场景下的使用。闪存的控制器支持读操作，页擦除，整片擦除，可通过 16 位（半字）方式编程写入闪存，其擦写寿命可达 20000 次。闪存控制器在读取数据时，支持带预取缓冲器的数据接口，以支持 MCU 运行在更高的主频。

### 2.2 闪存构成与说明

#### 2.2.1 闪存构成

闪存空间由 128 位宽的存储单元组成，既可以存代码又可以存数据。

主闪存块按 512 页（每页 1K 字节）或 128 个写保护块（每块 4K 字节）。

主闪存块可按页（每 1K 字节）擦除（Page Erase）和按页设置读保护；

以 4 页（4K 字节）为单位作为 1 个写保护块来设置写保护。

整个片内 Flash 由两部分组成：一部分是主存储块，另一部分是信息存储块。

主存储块用于存储用户代码和数据，用户代码可以对主存储器进行擦除、编程和读取操作。每个 1K 字节在主存储块中称为一页，可以执行最小单位的擦除；另外以 1 个写保护区为单位（4K 字节，4 页=1 个写保护块）进行写保护分配，如表 2-1 所示。

表 2-1 Flash 存储空间

模块	区块名称	页名称	地址	大小（字节）
主存储块	写保护区 0	页 0	0x0800 0000 - 0x0800 03FF	1K
		页 1	0x0800 0400 - 0x0800 07FF	1K
		页 2	0x0800 0800 - 0x0800 0BFF	1K
		页 3	0x0800 0C00 - 0x0800 0FFF	1K
	写保护区 7			
		页 28	0x0800 7000 - 0x0800 73FF	1K
		页 29	0x0800 7400 - 0x0800 77FF	1K
		页 30	0x0800 7800 - 0x0800 7BFF	1K
	写保护区 8	页 31	0x0800 7C00 - 0x0800 7FFF	1K
		页 32	0x0800 8000 - 0x080083FF	1K
		页 508	0x0807 F000 - 0x0807 F3FF	1K
		页 509	0x0807 F400 - 0x0807 F7FF	1K
	写保护区 127	页 510	0x0807 F800 - 0x0807 FBFF	1K
		页 511	0x0807 FC00 - 0x0807 FFFF	1K

信息存储块中，除了“系统存储器 ISP”区域出厂锁定，用户不可写入外；其他区域在一定条件下用户可进行读写操作。信息存储器可分为区块读保护字节，保密空间，系统存储器 ISP 和选项字节四部分，区块读保护字节用于使能读保护（可选），保密空间用于存储需特别保护的数据与代码，ISP Firmware 部分用于固化 ISP 升级的代码。选项字节（Option byte）部分中的前 12 个字节是主存储器的写和读保护信息，剩余字节可用于存放用户特殊的数据。对于信息存储块部分，用户可以通过规定的流程对其擦除、编程和读取。仅 ISP 部分由于用于固化 ISP 升级的代码，不支持用户进行擦除和编程。

表 2-2 信息块

模块	名称	地址	大小（字节）
信息块	区块读保护字节 (可选)	0x1FFE 0000 - 0x1FFE 01FF	0.5K
	保密空间	0x1FFE 1000 - 0x1FFF 1FFF	4K
	系统存储 ISP	0x1FFF E800 - 0x1FFF F7FF	4K
	选项字节	0x1FFF F800 - 0x1FFF F9FF	0.5K

## 2.2.2 选项字节说明

在选项字节页中，内容主要有写保护使能，看门狗使能等。Flash 控制器可以通过选项字节中值的设置，达到使能主存储器禁止写入功能，以避免非法写入；还可以使能硬件看门狗。相关信息存储在选项字节中，修改选项字节中内容后，需要复位或重新上电后才生效，写入时需按补码方式写入，如 nUser，nData 等。每次系统复位后，选项字节会重新装载选项字节信息块的数据，并做相应的判断与状态改变，这些的状态保存在选项字节寄存器（FLASH\_OBR 及 FLASH\_WRPR）中。在信息块中每个选择位都有对应的反码位，在加载选择位时反码位用于验证选择位是否正确，如果在加载过程中发现有差别，将产生一个选项字节错误标志（OPTERR）；如开启中断，将触发中断。

选项字节块中选项字节的组织结构如下表所示（位 15~8 中的值为位 7~0 中选项字节 0 的反码）：

表 2-3 选项字节组织结构

地址	[15: 8]	[7: 0]	默认值
0x1FFF F800	nReserved0	Reserved0	0xFFFF
0x1FFF F802	nUSER	USER	0xFFFF
0x1FFF F804	nData0	Data0	0xFFFF
0x1FFF F806	nData1	Data1	0xFFFF
0x1FFF F808	nWRP0	WRP0	0xFFFF
0x1FFF F80A	nWRP1	WRP1	0xFFFF
0x1FFF F80C	nWRP2	WRP2	0xFFFF
0x1FFF F80E	nWRP3	WRP3	0xFFFF
0x1FFF F810	nWRP4	WRP4	0xFFFF
0x1FFF F812	nWRP5	WRP5	0xFFFF

地址	[15: 8]	[7: 0]	默认值
0x1FFF F814	nWRP6	WRP6	0xFFFF
0x1FFF F816	nWRP7	WRP7	0xFFFF
0x1FFF F818	nWRP8	WRP8	0xFFFF
0x1FFF F81A	nWRP9	WRP9	0xFFFF
0x1FFF F81C	nWRP10	WRP10	0xFFFF
0x1FFF F81E	nWRP11	WRP11	0xFFFF
0x1FFF F820	nWRP12	WRP12	0xFFFF
0x1FFF F822	nWRP13	WRP13	0xFFFF
0x1FFF F824	nWRP14	WRP14	0xFFFF
0x1FFF F826	nWRP15	WRP15	0xFFFF

每次系统复位后，选项字节会被重新加载，并保存在选项字节寄存器（FLASH\_OBR）中，每个选择位都在信息块中有它的反码位；在加载选择位时，反码位用于验证选择位是否正确，如果有任何的差别，将产生一个选项字节错误标志（OPTERR）。

表 2-4 USER 的位含义

	<b>Bit</b>	<b>Field</b>	<b>Type</b>	<b>Default</b>	<b>Description</b>	<b>FLASH_OBR</b>
RDP	7: 0	RDP	rw	0xFF	0xA5	FLASH_OBR. Bit1
nRDP	15:8	nRDP	rw	0xFF	0x5A	
User Byte	0	WDG_SW	rw	0x01	0: 硬件看门狗 1: 软件看门狗	FLASH_OBR. Bit2
	1	nRST_STOP	rw	0x01	0: 当进入停机 (STOP) 模式时产生复位 1: 进入停机 (STOP) 模式时不产生复位	FLASH_OBR. Bit3
	2	nRST_STDBY	rw	0x01	0: 当进入待机模式时产生复位 1: 进入待机模式时不产生复位	FLASH_OBR. Bit4
	3	Reserved	rw	0x01	保留为 0x01	保留
	4	nBOOT_1	rw	0x01	0: nBOOT1=0 1: nBOOT1=1	FLASH_OBR. Bit6
	5	Reserved	rw	0x01	保留为 0x01	保留
	6	Reserved	rw	0x01	保留为 0x01	保留
	7	Reserved	rw	0x01	保留为 0x01	保留
DATA0 Byte	0	DATA0. Bit0	rw	0x01	用户自定义	FLASH_OBR. Bit10
	1	DATA0. Bit1	rw	0x01	用户自定义	FLASH_OBR. Bit11
	2	DATA0. Bit2	rw	0x01	用户自定义	FLASH_OBR. Bit12
	3	DATA0. Bit3	rw	0x01	用户自定义	FLASH_OBR. Bit13
	4	DATA0. Bit4	rw	0x01	用户自定义	FLASH_OBR. Bit14
	5	DATA0. Bit5	rw	0x01	用户自定义	FLASH_OBR. Bit15
	6	DATA0. Bit6	rw	0x01	用户自定义	FLASH_OBR. Bit16
	7	DATA0. Bit7	rw	0x01	用户自定义	FLASH_OBR. Bit17
DATA1 Byte	0	DATA1. Bit0	rw	0x01	用户自定义	FLASH_OBR. Bit18
	1	DATA1. Bit1	rw	0x01	用户自定义	FLASH_OBR. Bit19

	Bit	Field	Type	Default	Description	FLASH_OBR
	2	DATA1. Bit2	rw	0x01	用户自定义	FLASH_OBR. Bit20
	3	DATA1. Bit3	rw	0x01	用户自定义	FLASH_OBR. Bit21
	4	DATA1. Bit4	rw	0x01	用户自定义	FLASH_OBR. Bit22
	5	DATA1. Bit5	rw	0x01	用户自定义	FLASH_OBR. Bit23
	6	DATA1. Bit6	rw	0x01	用户自定义	FLASH_OBR. Bit24
	7	DATA1. Bit7	rw	0x01	用户自定义	FLASH_OBR. Bit25

注意：在写保护值中，一个比特位对应四页，即 4096 Bytes。

### 2.2.3 区块读保护字节说明

在区块读保护字节空间中，Flash 控制器通过对区块读保护字节中的内容做相应的设置，达到使能主存储器读取保护功能，以避免非法访问被保护空间。写入的内容按半字写入，相关信息写入到区块读保护字节中后，需要复位或重新上电后才生效。

区块读保护字节空间的值构成与选项字节相同（位 15~8 中的值为位 7~0 中选项字节 0 的反码）。

表 2-5 区块读保护字节含义

地址	[15: 8]	[7: 0]	默认值	注释
0x1FFE 0000	0x7F	0x80	0xFFFF	
0x1FFE 0002	0xFF	0x00	0xFFFF	
0x1FFE 0004 ~ 0x1FFE 01FF	0xFF	0xFF	0xFFFF	所有的值全保留为 0xFFFF

默认状态下，选项字节块与区块读保护字节始终是可以读且被写保护。要想对选项字节块与区块读保护字节进行写操作（编程与擦除），首先要在 OPTKEYR 中写入正确的键序列，随后允许对选项字节块的写操作，FLASH\_CR 寄存器的 OPTWRE 位表示允许写，清除这位将禁止写操作。

### 2.2.4 保密空间说明

在保密空间中，Flash 控制器可以通过指定的操作字，从而打开对保密空间做直接的读写访问；通过指定的密钥配对操作，达到该块内容的读写保护，以避免非法访问。

## 2.3 闪存操作与流程

### 2.3.1 闪存读操作

用户代码和数据存储于主存储块中，闪存控制器可以按照 8bit/16bit/32bit 位读取数据或取指

令。主闪存模块与普通外设一样统一寻址访问。基于读保护与写保护的要求，任何对主存储块的内容的读写操作都须经过特定的判断过程，以防止非法读取与写入。

闪存按 Flash 访问控制寄存器（FLASH\_ACR）中的设定的方式，通过 AHB 总线执行取指令和取数据。结合 AHB 时钟，设定相应的访问时延（Latency），使能预取指缓冲区后，可提高 CPU 的取指令速度，从而提高 CPU 的运行速度。访问时延（Latency）在 SYSCLK 低于等于 24MHz，可以设定为 0，此后每增加 24MHz，需要增加一个时延。

上电复位后，闪存控制器默认设定预取指缓冲区是打开的。如需要关闭或重新打开预取指缓冲功能，必须设定 SYSCLK 低于 24MHz，并且 AHB 时钟没有经过任何分频的条件下（SYSCLK 必须等于 HCLK）才可以关闭或重新打开预取指缓冲功能。

为了保护对 Flash 的正确读取，必须在 Flash 访问控制寄存器中的 LATENCY[2:0] 中指定预取指控制器的速度比，这个数值等于每次访问 Flash 后到下次访问之间所需插入的等待周期的个数。复位后，这个值默认为零，也就是没有插入等待周期的状态，相应的系统时钟也复位为使用内置时钟 HSI=8MHz。复位后如果需要修改系统时钟，必须先配置好安全的 LATENCY[2:0] 值；而当 AHB 时钟的预分频器大于 1 时，预取指缓冲区也需设定相应的访问时延（Latency）。

表 2-6 Latency 设置关系

SYSCLK	AHB DIV	Latency
0MHz < SYSCLK <= 24MHz	1	0
24MHz < SYSCLK <= 48MHz	1	1
48MHz < SYSCLK <= 72MHz	1	2
72MHz < SYSCLK <= 96MHz	1	3
96MHz < SYSCLK <= 120MHz	1	4

### 2.3.2 闪存编程方式与操作流程

嵌入式闪存支持如下三种编程方式。

表 2-7 编程方式

编程方式	编程说明
在电路编程（ICP）	ICP 是指通过特定烧写器，利用 SWD 接口，改变 Flash 的内容，将用户代码烧录到 MCU 中。
在系统编程（ISP）	ISP 是指通过 ISP Firmware，结合特定指定的 UART，或 I2C，SPI 等接口，改变 Flash 的内容，将用户代码烧录到 MCU 中。
在应用编程（IAP）	与 ICP 和 ISP 的方法不同的是，IAP（在应用编程）能够使用 MCU 支持的任何通信接口（UART，I2C，SPI，CAN，USB 等）下载程序或者数据。IAP 允许用户在运行程序的过程中重写应用程序，前提是一部分应用程序必须预先用 ICP 或 ISP 的方法烧写进去。

烧写和擦除操作在整个产品工作电压范围内都可以完成，在对 Flash 空间做写操作或擦除操作时，内部振荡器（HSI）必须处于开启状态，还需确保 AHB 时钟大于等于 8MHz。

只要 CPU 不去访问 Flash 空间，进行中的 Flash 写操作不会妨碍 CPU 的运行（从 RAM 或 ISP 中运行）。在对 Flash 进行写操作或擦除操作时，对 Flash 的读访问都会遇到总线停顿，直到写操作

或擦除操作完成后才会继续执行；因此在写操作或擦除 Flash 时，不可以对它取指和访问数据。

闪存的编程操作由一系列的动作组合而成，主要包括：

- 对 Flash 操作的解锁与保护
- 对 Flash 擦除（页擦除与整片擦除）
- 对 Flash 编程（半字编程）
- 对信息块中各空间（如选项字节）操作的解锁与保护
- 对信息块中各空间（如选项字节）擦除
- 对信息块中各空间（如选项字节）编程（半字编程）

## ISP、IAP 方式编程流程

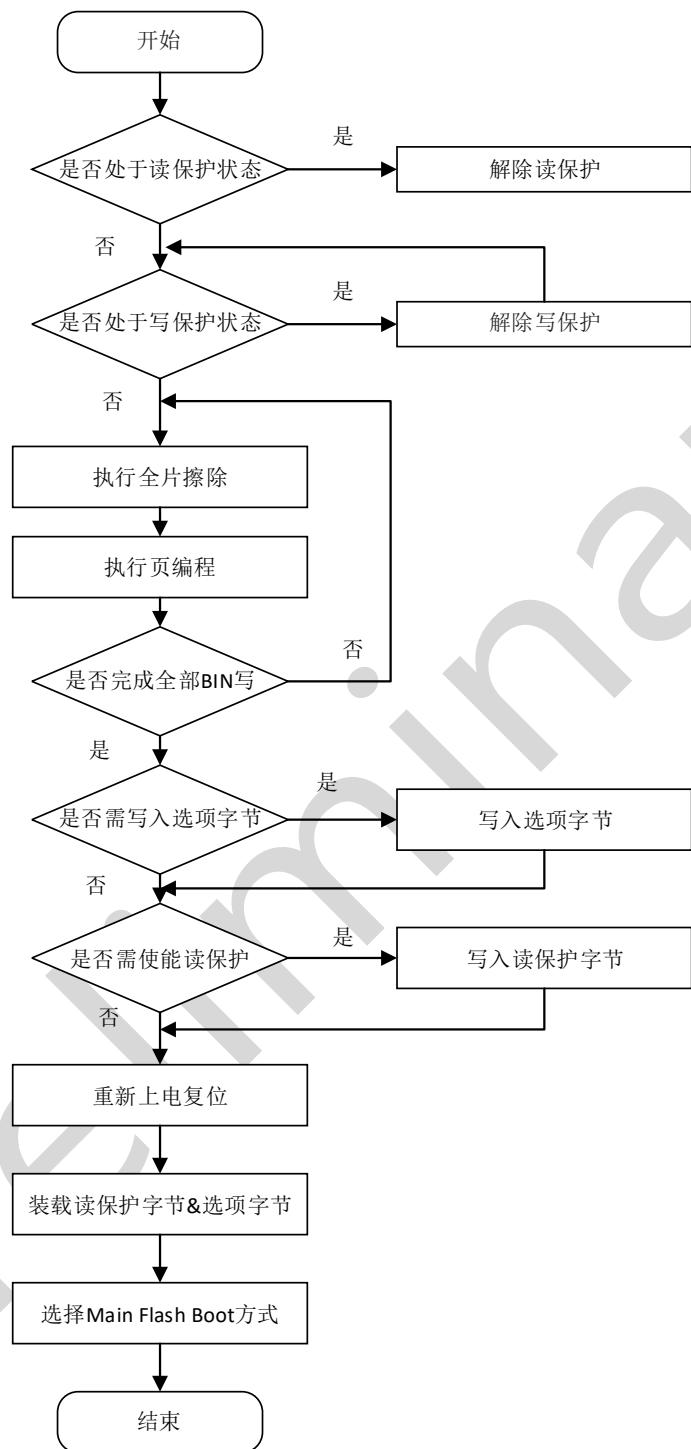


图 2-1 ISP 方式编程流程图

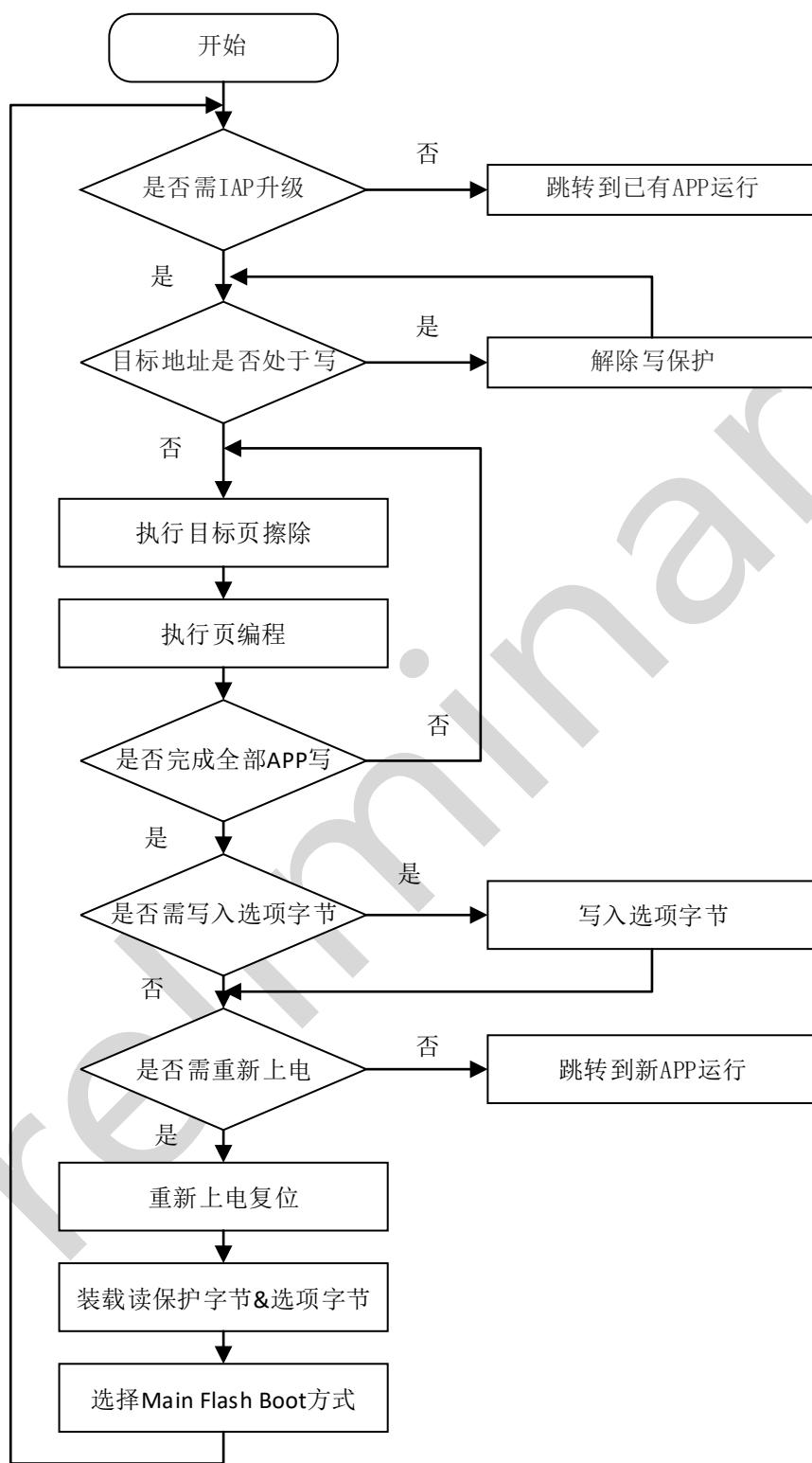


图 2-2 IAP 编程流程

### 2.3.3 对闪存块操作限制的解除与使能

嵌入式闪存在复位后是处于受保护状态的，可避免意外的页擦除、全片擦除和写值等破坏 Flash 存储空间的操作。复位后，FLASH\_CR 寄存器进入锁定状态，FLASH\_CR 的 LOCK 位被控制器模块置为

1。只有通过先后向 FLASH\_KEYR 寄存器写入 0x45670123 和 0xCDEF89AB 做解锁操作后，FLASH\_CR 的 LOCK 位置为 0，才能开启对 FLASH\_CR 的访问权限，否则 FLASH\_CR 寄存器不允许被改写。

可以通过软件设置 FLASH\_CR 的 LOCK 位置为 1 再次锁定，使 Flash 存储器处于受保护状态。

解除保护操作代码：

```
#define FLASH_KEY1      ((unsigned int)0x45670123)  
  
#define FLASH_KEY2      ((unsigned int)0xCDEF89AB)  
  
void FLASH_Unlock()  
{  
    FLASH->KEYR = ((unsigned int)0x45670123);  
    FLASH->KEYR = ((unsigned int)0xCDEF89AB);  
}
```

不符合上述顺序的操作与写入错误的值，将会锁死 FLASH\_CR，并引发一个总线错误，直至下次复位。

使能保护操作代码：

```
#define FLASH_CR_LOCK_Pos      (7)  
  
#define FLASH_CR_LOCK      (0x01U << FLASH_CR_LOCK_Pos)  
  
void FLASH_Lock(void)  
{  
    FLASH->CR |= FLASH_CR_LOCK;  
}
```

对选项字节区块操作限制的解除与使能

闪存控制器在复位后，它的选项字节区块默认是处于写保护的，并且任何时候都是可读的。同样是为了避免对项字节区与读保护设置区做块擦除和写值等破坏性操作，复位后，FLASH\_CR 寄存器进入锁定状态，FLASH\_CR 的 LOCK 位被控制器模块置为 1，而 OPTWRE 位被控制器模块清除为 0；因此需先向 FLASH\_KEYR 寄存器写入 0x45670123 和 0xCDEF89AB 做解锁 FLASH 操作，FLASH\_CR 的 LOCK 位置为 0 后，才做选项字节与区块读保护字节块的解锁。通过向 FLASH\_OPT\_KEYR 寄存器先后写入 0x45670123 和 0xCDEF89AB，从而使硬件将 FLASH\_CR 寄存器的 OPTWRE 位置 1；才能对选项字节区与区块读保护字节区执行块擦除，半字编程操作。可将 FLASH\_CR 寄存器的 OPTWRE 位置 0，从而禁止对选项字节区与区块读保护字节区执行块擦除，半字编程操作。

表 2-8 保护设置的状态变化

设置与状态	主闪存块	信息块	说明
上电复位 闪存控制器状态为 FLASH_CR.LOCK=1 FLASH_CR.OPTWRE=0	保护	保护	使能对主闪存块的操作保护 使能对选项字节与区块读保护字节块的操作保护
设置 FLASH_KEYR=0x45670123 FLASH_KEYR=0xCDEF89AB 闪存控制器状态变为 FLASH_CR.LOCK=0 FLASH_CR.OPTWRE=0	解除保护	保护	解除对主闪存块的操作保护，可对主闪存块执行全片擦除，页擦除，半字编程 还保持使能对选项字节与区块读保护字节块的操作保护，不能对选项字节区与区块读保护字节区执行块擦除，半字编程操作
FLASH_KEYR=0x45670123 FLASH_KEYR=0xCDEF89AB FLASH OTPKEYR=0x45670123 FLASH OTPKEYR=0xCDEF89AB 闪存控制器状态变为 FLASH_CR.LOCK=0 FLASH_CR.OPTWRE=1	解除保护	解除保护	解除对主闪存块的操作保护，可对主闪存块执行全片擦除，页擦除，半字编程 解除对选项字节与区块读保护字节块的操作保护，可对选项字节区与区块读保护字节区执行块擦除，半字编程操作
设置 FLASH_CR.OPTWRE=0 保持 FLASH_CR.LOCK=0	解除保护	使能保护	仍处于解除对主闪存块的操作保护，使能了对选项字节与区块读保护字节块的操作保护
设置 FLASH_CR.OPTWRE=0 设置 FLASH_CR.LOCK=1	使能保护	使能保护	使能了对主闪存块的操作保护与对选项字节与区块读保护字节块的操作保护

解除保护操作代码：

```
#define FLASH_KEY1      ((unsigned int)0x45670123)
#define FLASH_KEY2      ((unsigned int)0xCDEF89AB)

void FLASH_Unlock(void)
{
    FLASH->KEYR = ((unsigned int)0x45670123);
    FLASH->KEYR = ((unsigned int)0xCDEF89AB);
}
```

使能保护操作代码：

```
#define FLASH_CR_LOCK_Pos      (7)
#define FLASH_CR_LOCK          (0x01U << FLASH_CR_LOCK_Pos)

void FLASH_Lock(void)
{
```

```
FLASH->CR |= FLASH_CR_LOCK;  
}
```

解除选项字节与区块读保护字节区保护操作代码:

```
#define FLASH_KEY1      ((unsigned int)0x45670123)  
  
#define FLASH_KEY2      ((unsigned int)0xCDEF89AB)  
  
void FLASH_OPT_Unlock (void)  
{  
    FLASH->OPTKEYR = FLASH_KEY1;  
    FLASH->OPTKEYR = FLASH_KEY2;  
}
```

对选项字节与区块读保护字节区保护使能操作代码:

```
#define FLASH_CR_OPTWRE_Pos      (9)  
#define FLASH_CR_OPTWRE          (0x01U << FLASH_CR_OPTWRE_Pos)  
  
void FLASH_OPT_Lock(void)  
{  
    FLASH->CR &= ~FLASH_CR_OPTWRE;  
}
```

#### 2.3.4 主闪存块擦除

闪存控制器支持整片擦除主闪存块和以页为单位擦除主闪存中的页。

整片擦除功能将初始化主闪存块的所有内容，使所有的值为 0xFFFF。但信息块不会受这个命令影响。

整片擦除操作的寄存器设置，具体步骤如下：

执行步骤	简易流程图
<p>读 FLASH_CR 的 LOCK 位，确保解除 Flash 保护，否则执行 Flash Unlock</p> <p>设置 FLASH_CR 寄存器中的 MER 位为 1，使能对主闪存全片擦除功能</p> <p>设置 FLASH_CR 寄存器中的 STRT 位为 1，开始对主闪存全擦除</p> <p>检查 FLASH_SR 中的 BUSY 位，等待对主闪存全擦除的操作完成</p> <p>可选做 Blank 检查，以确保擦除成功</p>	<pre>graph TD; A[读取FLASH_CR中的LOCK位] --&gt; B{FLASH_CR的LOCK=1}; B -- 是 --&gt; C[执行解锁序列]; B -- 否 --&gt; D[FLASH_CR中的MER位写1]; D --&gt; E[FLASH_CR中的START位写1]; E --&gt; F{FLASH_SR的BSY=1}; F -- 是 --&gt; G[读取用户存储器中的所有地址来检查擦除操作]; F -- 否 --&gt; H[读取用户存储器中的所有地址来检查擦除操作];</pre> The flowchart illustrates the steps for performing a full chip erase. It begins by reading the LOCK bit from the FLASH_CR register. If the LOCK bit is 1 (indicating protection), it executes a unlock sequence. If the LOCK bit is 0, it sets the MER bit in the FLASH_CR register to enable full-chip erase. Then, it sets the START bit in the FLASH_CR register to begin the erase operation. Finally, it checks the BSY bit in the FLASH_SR register. If BSY is 1 (indicating忙), it reads all addresses in user memory to check the erase operation. If BSY is 0, it also performs the final step of reading user memory to check the operation.

页擦除操作的寄存器设置，具体步骤如下：

执行步骤	简易流程图
<p>读 FLASH_CR 的 LOCK 位，确保解除 Flash 保护，否则执行 Flash Unlock</p> <p>设置 FLASH_CR 寄存器中的 PER 位为 1，使能页擦除功能</p> <p>把待擦除页基地址写入 FLASH_AR 寄存器</p> <p>设置 FLASH_CR 寄存器中的 STRT 位为 1，开始页擦除</p> <p>检查 FLASH_SR 中的 BUSY 位，</p> <p>等待页擦除的操作完成 可选对这一页的内容做检查，以确保页擦除成功</p> <p>连续多页擦除，可以重复 2~7 操作；直到完成所有页面擦除</p>	<pre> graph TD     A[读FLASH_CR的LOCK位] --&gt; B{FLASH_CR=的LOCK=1}     B -- 是 --&gt; C[执行解锁序列]     C --&gt; D[写FLASH_CR的PER位]     D --&gt; E[将要擦除的页面地址写入FLASH_AR]     E --&gt; F[将FLASH_CR中的STRT位写入1]     F --&gt; G{FLASH_SR中的BSY=1}     G -- 否 --&gt; H[读取页面中的所有地址 检查页面是否被擦除]     G -- 是 --&gt; F   </pre>

### 2.3.5 主闪存块编程

主闪存只支持以 16 位半字编程，用来修改主存储闪存块内容。如果以 32 位整字或 8 位字节的长度编程，将引起硬件错误中断。当 FLASH\_CR 中的 PG 位为 1 时，直接对相应的地址写一个半字（16 位），就是一次编程操作。

主闪存控制器会预读待编程半字是否为全 1（即是否为 0xFFFF）；如果不是，这次编程操作会自动取消，并且在 FLASH\_SR 寄存器的 PGERR 位上提示编程错误警告。

如果待编程地址所对应的写保护块在 FLASH\_WRPTR 中的写保护位有效，同样也不会有编程动作，同样也会产生编程错误警告。编程动作结束后，FLASH\_SR 寄存器中得 EOP 位会给出提示。

注意：当 CPU 进入省电模式时，通过 SWD 接口，对闪存操作将产生错误。避免在主闪存中运行中断程序，又进行擦除或编程操作。

页擦除操作的寄存器设置，具体步骤如下：

执行步骤	简易流程图
<p>读 FLASH_CR 的 LOCK 位，确保解除 Flash 保护，否则执行 Flash Unlock</p> <p>设置 FLASH_CR 寄存器中的 PG 位为 1，使能半字编程功能</p> <p>以半字为单位向目标地址写入数据，目标地址需以半字对齐</p> <p>检查 FLASH_SR 中的 BUSY 位，等待半字编程操作完成</p> <p>可选读目标地址数据，以确保半字编程成功</p> <p>连续多个半字编程，可以重复 2~5 操作；直到完成所有目标地址的半字编程</p>	<pre>graph TD     A[读FLASH_CR的LOCK位] --&gt; B{FLASH_CR的LOCK=1}     B -- 是 --&gt; C[执行解锁序列]     B -- 否 --&gt; D[FLASH_CR的PG位写1]     D --&gt; E[在目标地址执行半字写入]     E --&gt; F{FLASH_SR的BSY=1}     F -- 是 --&gt; G[读取编程地址来检查编程值]     F -- 否 --&gt; E</pre>

### 2.3.6 选项字节区块擦除

选项字节区块擦除操作的寄存器设置，具体步骤如下：

执行步骤	简易流程图
<p>读 FLASH_CR 的 LOCK 位，确保解除 Flash 保护，否则执行 Flash Unlock</p> <p>读 FLASH_CR 的 OPTWRE 位，确保解除选项区块保护，否则执行选项区块 Unlock</p> <p>把待擦除区块基地址写入 FLASH_AR 寄存器</p> <p>设置 FLASH_CR 寄存器中的 OPTER 位为 1，使能选项字节块擦除功能</p> <p>设置 FLASH_CR 寄存器中的 STRT 位为 1</p> <p>检查 FLASH_SR 中的 BUSY 位，等待块擦除操作完成</p> <p>可选对这一区块的内容做检查，以确保对该区块擦除成功</p> <p>连续多块擦除，可以重复 3~7 操作；直到完成所有选项区块擦除</p>	<pre>graph TD; A[读FLASH_CR的LOCK位] --&gt; B{FLASH_CR的LOCK=1}; B -- 是 --&gt; C[执行解锁序列]; B -- 否 --&gt; D{FLASH_CR的OPTWRE=0}; D -- 是 --&gt; E[执行解锁OPT序列]; D -- 否 --&gt; F[将闪存选项字节块地址写入FLASH_AR]; F --&gt; G[FLASH_CR中的OPTPG位写1]; G --&gt; H[FLASH_CR的STRT位写1]; H --&gt; I{FLASH_SR的BUSY=1}; I -- 是 --&gt; J[通过读取闪存选项存储器中的所有地址来检查闪存选项擦除操作]; I -- 否 --&gt; F;</pre>

### 2.3.7 选项字节区块编程

选项字节区块的编程与主闪存块地址的编程不同，因其写入值受复位后加载到配置选项，需要更加严格的保护。解除对闪存控制器的访问限制后，还需要对选项字节区块与读保护区解除访问限制。完成该操作后，FLASH\_CR 寄存器中的 OPTWRE 位会被置 1，才能允许后续的编程操作。

选项字节有效数据为低 8 位，而高 8 位为低 8 位的反码，从而组成为 16 位数据。在编程过程中，软件将高 8 位设置为低 8 位的反码，保证选项字节的写入值总是对的，然后依次写入 16 位数据。

当可选字节被改变时，需要系统上电复位使之生效。

选项字节区块半字编程操作的寄存器设置，具体步骤如下：

执行步骤	简易流程图
<p>读 FLASH_CR 的 LOCK 位，确保解除 Flash 保护，否则执行 Flash Unlock</p> <p>读 FLASH_CR 的 OPTWRE 位，确保解除选项区块保护，否则执行选项区块 Unlock</p> <p>设置 FLASH_CR 寄存器中的 OPTPG 位为 1，使能选项字节区块半字编程</p> <p>写数据（半字）到目标地址，目标地址需以半字对齐</p> <p>检查 FLASH_SR 中的 BUSY 位，等待半字编程操作完成</p> <p>可选读目标地址数据，以确保半字编程成功</p> <p>连续多个半字编程，可以重复 3~6 操作；直到完成所有目标地址的半字编程</p>	<pre> graph TD     A[读FLASH_CR的LOCK位] --&gt; B{FLASH_CR的LOCK=1}     B -- 是 --&gt; C[执行解锁序列]     B -- 否 --&gt; D{FLASH_CR的OPTWRE=0}     D -- 是 --&gt; E[执行解锁OPT序列]     D -- 否 --&gt; F[FLASH_CR的OPTPG位写1]     F --&gt; G[在所需地址执行半字写入]     G --&gt; H{FLASH_SR的BSY=1}     H -- 是 --&gt; I[通过读取目标地址数据来检查编程值]     H -- 否 --&gt; J[通过读取目标地址数据来检查编程值]   </pre>

在执行选项字节区块半字编程的同时，会自动检查 FLASH\_CR 寄存器中的 OPTWRE 是否为 1，否则相关操作会被取消并且在 FLASH\_SR 中的 PGERR 位提示错误。编程操作结束后，会由 FLASH\_SR 寄存器的 EOP 位给出提示。然后就可以先置位 FLASH\_CR 中的 OPTPG 位，再按半字单位写目标地址。

### 2.3.8 闪存保护

主闪存块使能读保护可以防范主闪存区块的代码被不可信的代码读出，也可以防范在程序跑飞的时候对主闪存块的意外擦除与编程。读保护使能操作，是对整个主存储块起作用；而使能写保护的最小单位是一个写保护块（即 4 页）。

#### 2.3.8.1 主闪存块读保护

主闪存块使能或解除读保护是通过从内置 SRAM 或 ICP、ISP 方式设置 RDP 半字；然后在系统重新上电复位，加载了新的 RDPs 后起作用的。如果在设置了读保护时，需要执行一次上电复位，而不是系统复位，才能起作用。

##### 2.3.8.1.1 使能读保护

按选项字节区域半字编程的操作方式，按顺序写 RDP 二个半字到对应地址

- 1) 设置 FLASH AR 地址值为 0xFFFFF800，执行该选项区块擦除
- 2) 按选项字节区域半字编程的操作方式，按顺序写 0x807F 半字到对应地址
- 3) 进行上电复位以重新加载选项字节，此时读保护被解除。

当 RDP 字包含下列数值时，且被重新上电复位后主闪存块被置于保护状态。

表 2-9 Flash 读保护状态

使能读保护操作	读保护状态
对 0xFFFFF800 选项区块擦除	
写 0x807F 半字到对应地址 0xFFFFF800	保护
重新上电复位，读保护被解除	

当读保护半字被写入相应的值以后：

- 1) 只允许从用户代码中对主闪存存储器的读操作（以非调试方式从主闪存存储器启动）。
- 2) 读保护后，调试模式下（**sram boot** 和 **debug** 模式）禁止对 **flash** 进行操作。
- 3) MCU 可以通过在主闪存存储器中执行的代码进行编程（实现 **IAP** 或数据存储等功能），但不允许在调试模式下或在从内部 **SRAM** 启动后执行写或擦除操作（整片擦除外）。
- 4) 所有通过 **SWD** 向内置 **SRAM** 装载代码并执行代码的功能依然有效，亦可以通过 **SWD** 从内置 **SRAM** 启动，这个功能可以用来解除读保护。
- 5) 通过从内置 **SRAM** 执行代码访问主闪存存储器的操作，通过 **DMA**、**SWV**（串行线观察器）、**SWD**（串行线调试）对闪存的访问都将被禁止。

### 2.3.8.1.2 解除读保护

从内置 **SRAM** 或 **ICP** 方式解除读保护的过程是：

- 1) 设置 **FLASH AR** 地址值为 **0x1FFFF800**，执行该选项区块擦除
- 2) 按选项字节区域半字编程的操作方式，按流程写 **0x5AA5** 半字到对应地址
- 3) 设置 **FLASH AR** 地址值为 **0x08000000**，执行主 **FLASH** 全片擦除
- 4) 进行上电复位以重新加载选项字节，此时读保护被解除。

表 2-10 Flash 解除读保护状态

解除读保护操作	读保护状态
对 0xFFFFF800 选项区块擦除	
写 0x5AA5 半字到对应地址 0xFFFFF800	
对 0x08000000 的主 Flash 全片擦除	解除读保护
重新上电复位，读保护被解除	

注：1.如选项字节块对应的地址值为非 **0xFFFF**，需先执行擦除选项字节块的动作，执行擦除选项字节块的动作不会导致自动的整片擦除操作，不会改变读保护状态。2.必需对 **0x08000000** 的主 **Flash** 全片擦除。

## 2.3.8.2 主闪存块写保护

### 2.3.8.2.1 使能写保护

写保护通过设置选项字节区块中的 WRP0~WRP15 中的 WRP 位为 0，来设置写保护，系统复位后将加载新选项字节，使能写保护。如果试图写入或擦除一个受写保护区的页面，会引起 FLASH\_SR 中的 WRPRTERR 标志位被置位。

表 2-11 写保护区域

地址	[15: 8]	[7: 0]	默认值	注释
0x1FFFF808	nWRP0	WRP0	0xFFFF	
0x1FFF F80A	nWRP1	WRP1	0xFFFF	
0x1FFF F80C	nWRP2	WRP2	0xFFFF	
0x1FFF F80E	nWRP3	WRP3	0xFFFF	
0x1FFF F810	nWRP4	WRP4	0xFFFF	
0x1FFF F812	nWRP5	WRP5	0xFFFF	
0x1FFF F814	nWRP6	WRP6	0xFFFF	
0x1FFF F816	nWRP7	WRP7	0xFFFF	
0x1FFF F818	nWRP8	WRP8	0xFFFF	
0x1FFF F81A	nWRP9	WRP9	0xFFFF	
0x1FFF F81C	nWRP10	WRP10	0xFFFF	
0x1FFF F81E	nWRP11	WRP11	0xFFFF	
0x1FFF F820	nWRP12	WRP12	0xFFFF	
0x1FFF F822	nWRP13	WRP13	0xFFFF	
0x1FFF F824	nWRP14	WRP14	0xFFFF	
0x1FFF F826	nWRP15	WRP15	0xFFFF	

### 2.3.8.2.2 解除写保护

解除写保护有下述 2 种情形：

情形 1：解除写保护，同时解除读保护：

使用闪存控制寄存器（FLASH\_CR）的 OPTER 位擦除整个选项字节区块；写 0x5AA5 半字到对应地址 0x1FFFF800

对 0x08000000 的主 Flash 全片擦除主闪存块；

进行系统复位，重装载选项字节（包含新的 WRP 字节），写保护被解除。

使用这种方法，将解除全片主闪存模块的写保护同时擦除全片主闪存块。

情形 2：解除写保护，同时保持读保护有效，这种情况常见于用户自己的实现在程序中编程的启

动程序：

使用闪存控制寄存器（FLASH\_CR）的 OPTER 位只擦除整个选项字节区域；

进行系统复位，重装载选项字节（包含新的 WRP 字节）；写保护被解除。

使用这种方法，将解除整个主闪存模块的写保护，还保持读保护有效。

## 2.4 Flash 寄存器描述

表 2-12 FLASH 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	FLASH_ACR	闪存访问控制寄存器	0x00000038
0x04	FLASH_KEYR	FPEC 键寄存器	0x00000000
0x08	FLASH_OPTKEYR	闪存 OPTKEY 寄存器	0x00000000
0x0C	FLASH_SR	闪存状态寄存器	0x00000000
0x10	FLASH_CR	闪存控制寄存器	0x00000080
0x14	FLASH_AR	闪存地址寄存器	0x00000000
0x1C	FLASH_OBR	选项字节寄存器	0x03FFFC1C
0x20	FLASH_WRPR	写保护寄存器	0xFFFFFFFF

注意：Flash 寄存器只支持以 32 位的方式访问

### 2.4.1 闪存访问控制寄存器 (FLASH\_ACR)

地址偏移：0x00

复位值：0x0000 0038

Bit	31	30	29	28	27	26	25	24	23	22	21		20		19	18	17	16
Field	Res.																	
Type																		
Bit	15	14	13	12	11	10	9	8	7	6	5		4		3	2	1	0
Field	Res.												PRFTBS	PRFTBE	Res.	LATENCY		
Type													r	rw		rw		

Bit	Field	Type	Reset	Description
31:6	Reserved			保留，始终读为 0。
5	PRFTBS	r	0x01	预取缓冲区状态(Prefetchbufferstatus) 0: 预取缓冲区关闭 1: 预取缓冲区开启

Bit	Field	Type	Reset	Description
4	PRFTBE	rw	0x01	<p>预取缓冲区使能 (Prefetch buffer enable)</p> <p>0: 关闭预取缓冲区 1: 启用预取缓冲区</p>
3	Reserved	r	1	保留, 始终读为 1。
2:0	LATENCY	rw	0x00	<p>时延 (Latency)</p> <p>这些位表示 SYSCLK(系统时钟) 周期与闪存访问时间的比例。</p> <p>000: 零等待状态, 当 <math>0 &lt; \text{SYSCLK} \leq 24\text{MHz}</math>      001: 一个等待状态, 当 <math>24\text{MHz} &lt; \text{SYSCLK} \leq 48\text{MHz}</math>      010: 两个等待状态, 当 <math>48\text{MHz} &lt; \text{SYSCLK} \leq 72\text{MHz}</math>      011: 三个等待状态, 当 <math>72\text{MHz} &lt; \text{SYSCLK} \leq 96\text{MHz}</math>      100: 四个等待状态, 当 <math>96\text{MHz} &lt; \text{SYSCLK} \leq 120\text{MHz}</math></p>

## 2.4.2 闪存访问控制寄存器 (FLASH\_KEYR)

地址偏移: 0x04

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31: 0	FKEYR	w	0x00000000	<p>FPEC 键 (Flash key)</p> <p>这些位用于输入 FPEC 的解锁键。</p>

注: 所有这些位是只写的, 读出时返回 0。

## 2.4.3 闪存 OPTKEY 寄存器 (FLASH\_OPTKEYR)

地址偏移: 0x08

复位值: 0x0000 0000

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

Type	w
Bit	15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Field	OPTKEYR.
Type	w

Bit	Field	Type	Reset	Description
31: 0	OPTKEYR	w	0x00000000	选择字节键 (Option byte key) 这些位用于输入选项字节的键以解除 OPTWRE。

注：所有这些位是只写的，读出时返回 0。

#### 2.4.4 闪存状态寄存器 (FLASH\_SR)

地址偏移: 0x0C

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.										EOP	WRPRTER R	Res.	PGERR	Res.	BSY
Type											rc_w1	rc_w1		rc_w1		r

Bit	Field	Type	Reset	Description
31: 6	Reserved			保留，始终读为 0
5	EOP	rc_w1	0x00	操作结束 (End of operation) 当闪存操作 (编程   擦除) 完成时，硬件设置这位为'1'，写入'1'可以清除这位状态。
4	WRPRTERR	rc_w1	0x00	写保护错误 (Write protection error) 试图对写保护的闪存地址编程时，硬件设置这位为'1'，写入'1'可以清除这位状态。
3	Reserved			保留，始终读为 0。

Bit	Field	Type	Reset	Description
2	PGERR	rc_w1	0x00	<p>编程错误 (Programming error)</p> <p>试图对内容不是'0xFFFF'的地址编程时，硬件设置这位为'1'，写入'1'可以清除这位状态。</p> <p>注：进行编程操作之前，必须先清除 FLASH_CR 寄存器的 STRT 位</p>
1	Reserved			保留，始终读为 0。
0	BSY	r	0x00	<p>忙 (Busy)</p> <p>该位指示闪存操作正在进行。在闪存操作开始时，该位被置为'1'；在操作结束或发生错误时该位被清除为'0'。</p>

## 2.4.5 闪存控制寄存器 (FLASH\_CR)

地址偏移: 0x10

复位值: 0x0000 0080

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.	Res.	Res.	Res.	OPTWRE	Res.	LOCK	STRT	OPTER	OPTPG	Res.	MER	PER	PG		
Type			.		rw		rw.	rw	rw	rw		rw	rw.	rw		

Bit	Field	Type	Reset	Description
31 :13	Reserved			保留，始终读为 0。
12	Reserved			保留，始终读为 0。
11	Reserved			保留，始终读为 0。
10	Reserved			保留，始终读为 0。

Bit	Field	Type	Reset	Description
9	OPTWRE	rc_w0	0x00	允许写选项字节 (Option byte write enable) 当该位为'1'时，允许对选项字节进行编程操作。当在FLASH_OPTKEYR 寄存器写入正确的键序列后，该位被置为'1'。 软件写 0 可清除此位。
8	Reserved			保留，始终读为 0。
7	LOCK	rw	0x01	锁 (Lock) 只能写'1'。当该位为'1'时表示 FPEC 和 FLASH_CR 被锁住。在检测到正确的解锁序列后，硬件自动清除此位为'0'。在一次不成功的解锁操作后，下次系统复位前，该位不能再被改变。
6	STRT	rw	0x00	开始 (Start) 当该位为'1'时将触发一次擦除操作。该位只可由软件置为'1'并在 BSY 变为'1'时自动清'0'。
5	OPTER	rw	0x00	擦除选项字节 (Option byte erase) 擦除选项字节。
4	OPTPG	rw	0x00	烧写选项字节 (Option byte programming) 对选项字节编程。
3	Reserved			保留，始终读为 0。
2	MER	rw	0x00	全擦除 (Mass erase) 选择擦除所有用户页。
1	PER	rw	0x00	页擦除 (Page erase) 选择擦除页。
0	PG	rw	0x00	编程 (Programming) 选择编程操作。

## 2.4.6 闪存地址寄存器 (FLASH\_AR)

地址偏移: 0x014

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31: 0	FAR	w	0x00000000	用户闪存选项字节地址 (Flash Address) 当进行编程时选择要编程的地址，当进行页擦除时选择要擦除的页。注意：当 FLASH_SR 中的 BSY 位为'1'时，不能写这个寄存器。

由硬件修改为当前最后使用的地址。页擦除操作中，必须修改这个寄存器以指定要擦除的页。

## 2.4.7 选项字节寄存器 (FLASH\_OBR)

地址偏移: 0x1C

复位值: 0x03FF FC1C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Res.					Data1											
Type						r											
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Data0					Res.			nBOOT1	Res.	nRST _STDBY	nRST _STOP	WDG _SW	Res.	OPTERR		
Type	r					r				r	r	r	r		r		

Bit	Field	Type	Reset	Description
31 :26	Reserved			保留，始终读为 0。
25 :18	Data1	r	0xFF	Data1
17 :10	Data0	r	0xFF	Data0
9:7	Res.			保留，始终读为 0。

Bit	Field	Type	Reset	Description
6	nBOOT1	r	0x01	nBOOT1
5	Reserved			保留，始终读为 0。
4	nRST_STDBY	r	0x01	进入待机模式时的复位事件 0: 当进入待机模式时产生复位 1: 进入待机模式时不产生复位
3	nRST_STOP	r	0x01	进入停机模式时的复位事件 0: 当进入停机 (STOP) 模式时产生复位 1: 进入停机 (STOP) 模式时不产生复位
2	WDG_SW	r	0x01	选择看门狗事件 0: 硬件看门狗 1: 软件看门狗
1	RDPRT	r	0x00	读保护 (Read protection level status) 当设置为 '1'，表示闪存存储器被读保护。 注：该位为只读。
0	OPTERR	r	0x00	选项字节错误 (Option byte error) 当该位为'1'时表示选项字节和它的反码不匹配。 注意：该位为只读。

#### 2.4.8 写保护寄存器 (FLASH\_WRP Rx)

地址偏移: 0x20,0x24,0x28,0x2C

复位值: 0xFFFF FFFF

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

Bit	Field	Type	Reset	Description
31: 0	WRP	r	0xFFFFFFFF	<p>写保护 (Write protect)</p> <p>该寄存器包含由 OBL 加载的写保护选项字节。</p> <p>0: 写保护生效 1: 写保护失效注意: 这些位为只读。</p>

### 3 高速缓冲存储器 (CACHE)

#### 3.1 基本操作

这一节将描述 cache 在常规操作中的表现。

##### 3.1.1 禁用 Cache

当 cache 被禁用时，所有读写访问都被旁路。在时钟方面没有额外的延迟。

##### 3.1.2 使能 Cache

如果启用了 cache，则可以使用以下操作模式：

###### 1) 旁路

当 cache 使能后，所有跟写、调试和不可高速缓存的访问都会被旁路。每次传输需要一个周期延迟。如果 cache 存在正在进行的预取访问，则延迟可能增加。

###### 2) Cache 命中

如果在 cache 存储器中找到可高速缓存的读事务数据（高速缓存命中），将直接返回结果，不会生成任何事务，也不需要等待状态。

###### 3) Linefill（行填充）

当查找可缓存读事务数据，却在缓存内存中找不到时，会发生 cache 丢失。缓存丢失事件将会触发行填充，并且将行填充的数据存储进 cache 中。每次传输需要一个周期延迟。如果 cache 存在正在进行的预取访问，则延迟可能增加。

###### 4) 性能监控

可以从 CSHR 和 CSMR 寄存器中读取命中和丢失的统计数据。

###### 5) 预取

预取功能是可编程选项。当 CCR 寄存器的 SET\_PREFETCH 位启用预取功能时，在一定条件下，cache 会在行填充之后执行预取。

预取对性能的影响取决于应用程序，需要看具体应用场景。预取不能保证性能一定优于不预取，因为一旦预取开始，将不能被中途停止。那么如果 cache 没有命中，且 cache 又正在从 flash 预取指令，则 CPU 不但不能从 cache 获取指令，也不能马上从 eFLASH 获取指令，导致性能下降。同时，预取操作会导致功耗增加。

###### 6) 中断

当 cache 没有被禁用时，手动使能 cache SRAM 失效会导致手动失效错误；当 cache 使能后工作在手动电源模式下时，未获得 cache SRAM 电源响应将会导致 cache SRAM 电源错误。

#### 3.2 操作流程

主要涉及 CCR 和 SR 寄存器。分三种操作模式，一般选择 APAI mode。

### 3.2.1 自动电源和自动失效模式 (APAI)

注：该模式下，当 cache 使能时，cache 首先会自动使 SRAM 中的值失效，然后再打开 cache。

使能 Cache

- 1) CCR 寄存器中的 SET\_MAN\_POW 位和 SET\_MAN\_INV 位置 0（置 CCR 寄存器为复位值 0x00）。
- 2) 将 CCR 寄存器中的 EN 位置 1（置 CCR 寄存器为 0x01）。
- 3) （可选）等待 SR 寄存器的 CS 位变为 2。

禁用 Cache

- 1) CCR 寄存器中的 EN 位置 0（置 CCR 寄存器为 0x00）。

### 3.2.2 手动电源和自动失效模式 (MPAI)

使能 Cache

- 1) 将 CCR 寄存器中的 SET\_MAN\_POW 位置 1，且 SET\_MAN\_INV 位置 0（置 CCR 寄存器为 0x08）。
- 2) 将 CCR 寄存器中的 POW\_REQ 位置 1（置 CCR 寄存器位 0x0C）。
- 3) 等待 SR 寄存器中的 POW\_STAT 位变为 1。
- 4) 将 CCR 寄存器中的 EN 位置 1（置 CCR 寄存器为 0x0D）。

禁用 Cache

- 1) 将 CCR 寄存器中的 EN 位置 0（置 CCR 寄存器为 0x0C）。
- 2) （可选）将 CCR 寄存器中的 POW\_REQ 位置 0（置 CCR 寄存器为 0x08）。

### 3.2.3 手动电源和手动失效模式 (MPMI)

使能 Cache 并使 SRAM 失效

使能 Cache:

- 1) 将 CCR 寄存器中的 SET\_MAN\_POW 位置 1，且 SET\_MAN\_INV 位置 1（置 CCR 寄存器为 0x18）。
- 2) 将 CCR 寄存器中的 POW\_REQ 位置 1（置 CCR 寄存器位 0x1C）。
- 3) 等待 SR 寄存器中的 POW\_STAT 位变为 1。
- 4) 将 CCR 寄存器中的 INV\_REQ 为置 1（置 CCR 寄存器为 0x1E）。
- 5) 等待 CCR 寄存器中的 INV\_REQ 位变为 0。
- 6) 将 CCR 寄存器中的 EN 位置 1（置 CCR 寄存器为 0x1D）。

使能 Cache 且 SRAM 有效

使能 Cache:

- 1) 将 CCR 寄存器中的 SET\_MAN\_POW 位置 1，且 SET\_MAN\_INV 位置 1（置 CCR 寄存器为 0x18）。
- 2) 将 CCR 寄存器中的 POW\_REQ 位置 1（置 CCR 寄存器位 0x1C）。
- 3) 等待 SR 寄存器中的 POW\_STAT 位变为 1。
- 4) 将 CCR 寄存器中的 EN 位置 1（置 CCR 寄存器为 0x1D）。

禁用 Cache

- 1) 将 CCR 寄存器中的 EN 位置 0（置 CCR 寄存器为 0x1C）。
- 2) （可选）将 CCR 寄存器中的 POW\_REQ 位置 0（置 CCR 寄存器位 0x18）。

使 Cache 失效

- 1) 将 CCR 寄存器中的 EN 位置 0（置 CCR 寄存器为 0x1C）。
- 2) 等待 SR 寄存器中的 CS 位变为 0。
- 3) 将 CCR 寄存器中的 INV\_REQ 位置 1（置 CCR 寄存器为 0x1E）。
- 4) 等待 CCR 寄存器中的 INV\_REQ 位变为 0。
- 5) 将 CCR 寄存器中的 EN 位置 1（置 CCR 寄存器为 0x1D）。

### 3.3 寄存器描述

表 3-1 Cache 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	CACHE_CCR	配置及控制寄存器	0x00000040
0x04	CACHE_SR	状态寄存器	0x00000000
0x08	CACHE_IMR	中断屏蔽寄存器	0x00000000
0x0C	CACHE_ISR	中断状态寄存器	0x00000000
0x14	CACHE_CSHR	命中统计寄存器	0x00000000
0x18	CACHE_CSMR	丢失统计寄存器	0x00000000

#### 3.3.1 配置及控制寄存器 (CACHE\_CCR)

偏移地址: 0x00

复位值: 0x0000 0040

Bit	31	30	29	28	27	26	25	24	23	22		21	20	19	18	17	16
Field	Reserved																
Type																	
Bit	15	14	13	12	11	10	9	8	7	6		5	4	3	2	1	0
Field											STATIST IC_EN	SET_PR EFETCH	SET_MA N_INV	SET_MAN _POW	POW _REQ	INV_ REQ	EN
Type											rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:7	Reserved			保留, 始终读为 0
6	STATISTIC_EN	rw	0x1	<p>统计使能</p> <p>0: 计数器停止</p> <p>1: 计数器使能</p> <p>如果不使用统计 cache 的命中或丢失信息, 则可以禁用, 能够节省功耗。</p>
5	SET_PREFETCH	rw	0x0	<p>预取值功能</p> <p>0: 禁用预载功能</p> <p>1: 使能预载功能</p>
4	SET_MAN_INV	rw	0x0	<p>手动或自动失效设置</p> <p>0: 当 cache 使能后自动 cache 失效模式。</p> <p>1: 手动 cache 失效模式。</p>
3	SET_MAN_POW	rw	0x0	<p>设置手动或自动 SRAM 电源请求</p> <p>0: 自动</p> <p>1: 手动</p>
2	POW_REQ	rw	0x0	手动 SRAM 电源请求
1	INV_REQ	rw	0x0	<p>手动设置失效请求</p> <p>手动设置 cache 数据失效。只有当第 4 位有效, 且 cache 被禁用时该位才起作用。该位只能自动清除, 不能手动清除</p>

Bit	Field	Type	Reset	Description
0	EN	rw	0x0	使能 Cache 0: 禁用 cache 1: 使能 cache

### 3.3.2 状态寄存器 (CACHE\_SR)

偏移地址: 0x04

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved										POW_STAT	AT	Res	INV_STAT	CS	
Type	r										r		r	r	r	

Bit	Field	Type	Reset	Description
31:5	Reserved			保留, 始终读为 0
4	POW_STAT	r	0x0	SRAM 电源响应
3	Reserved			保留, 始终读为 0
2	INV_STAT	r	0x0	无效化状态
1:0	CS	r	0x0	Cache 状态 0: Cache 已被禁用 1: Cache 正在使能 2: Cache 已使能 3: Cache 正在被禁用

### 3.3.3 中断屏蔽寄存器 (CACHE\_IMR)

偏移地址: 0x08

复位值: 0x0000 0000

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

Type																	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Reserved															MAN_IN_V_ERR	POW_ERR
Type																rw	rw

Bit	Field	Type	Reset	Description
31:2	Reserved			保留, 始终读为 0
1	MAN_INV_ERR	rw	0x0	屏蔽手动无效化错误的中断请求 (当 ISR 寄存器的 MAN_INV_ERR 置位时)。
0	POW_ERR	rw	0x0	屏蔽电源错误的中断请求 (当 ISR 寄存器的 POW_ERR 置位时)。

### 3.3.4 中断状态寄存器 (CACHE\_ISR)

偏移地址: 0x0C

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Reserved																
Type																	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Reserved															MAN_IN_V_ERR	POW_ERR
Type																rw	rw

Bit	Field	Type	Reset	Description
31:2	Reserved			保留, 始终读为 0
1	MAN_INV_ERR	rw	0x0	手动无效化错误标记。 当 Cache 没有被禁用的时候, 手动无效化会导致该位错误。写 1 清零。
0	POW_ERR	rw	0x0	当 Cache 使能后工作在手动电源模式的情况下, 未获得 SRAM 的电源响应则导致该位错误。写 1 清零。

### 3.3.5 Cache 命中统计寄存器 (CACHE\_CSHR)

偏移地址: 0x14

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:0	CSHR	rw	0x0	命中次数。写清零。

### 3.3.6 Cache 丢失统计寄存器 (CACHE\_CSMR)

偏移地址: 0x18

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:0	CSMR	rw	0x0	丢失次数。写清零。

## 4 电源控制 (PWR)

电源控制 PWR(Power Control)主要涉及芯片的供电系统、电源管理器、低功耗模式功能。

### 4.1 供电系统

本芯片的电源分配分为以下四个部分:

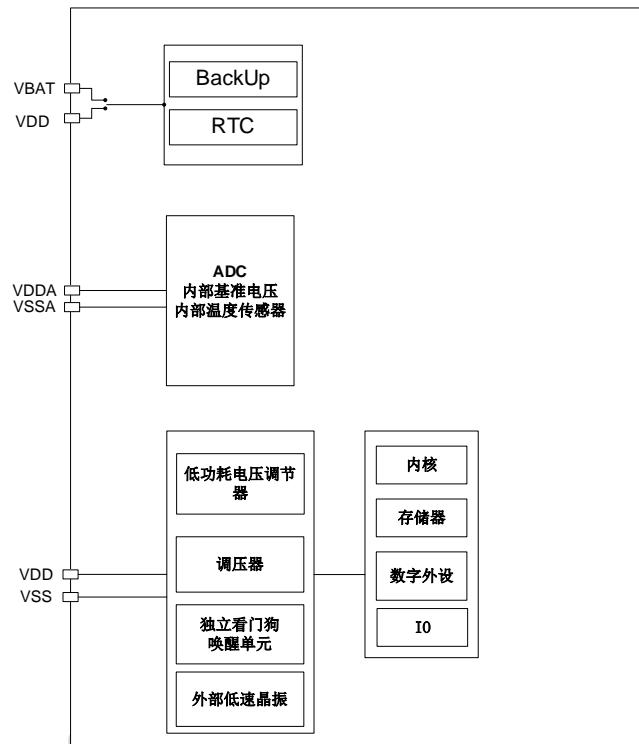


图 4-1 电源控制功能框图

- 1) 由 VDDA 和 VSSA 提供的模拟电源, 为芯片模拟部分工作提供电压, 用于 ADC 模块、内部基准电压、内部温度传感器。
- 2) 由 VDD 和 VSS 提供的数字电源, 用于数字部分和 I/O 引脚工作供电。
- 3) 由 VBAT 提供的电池备份区域电源, 用于 RTC 和备份模块工作供电。
- 4) 在供电系统中要求在相应的电源引脚上外接 10uF 和 100nF 的电容, 并尽量靠近引脚摆放。

注:

- 在有 VDDA 和 VSSA 的封装中, VDDA 和 VSSA 不可悬空, 且 VDD 和 VDDA 电压差要小于 50mV 。
- 在没有 VDDA 和 VSSA 的封装中, 已经在封装内部将 VDD 和 VDDA 连接, VSSA 和 VSS 连接。

#### 4.1.1 模拟模块供电

模拟模块供电主要给 MCU 内部模拟电路提供电源, 主要包括 ADC 模块、复位系统和 PLL 等, 因此电源的稳定性影响模拟模块的工作的性能。

ADC 的精度有一部分取决于 ADC 模块供电的稳定性，针对有需要高精度的 ADC 的应用，为了过滤和屏蔽来自印刷电路板上的毛刺对 ADC 采样的干扰，提高 ADC 的转换精度，ADC 使用一个独立外部电源供电，且提供稳定的外部供电电源。

- ADC 的电源引脚为 VDDA
- ADC 独立的电源地 VSSA

#### 4.1.2 数字模块供电

VDD、VSS 是芯片数字模块供电电源端口，主要为 IO 供电以及通过稳压器为内核、内置数字外设、存储器等供电。

#### 4.1.3 电池备份模块供电

VBAT、VDD 是芯片电池模块供电电源端口，通过选通信号选择供电电源，主要为 RTC 和备份寄存器供电。

#### 4.1.4 5v 电源域

5v 电源域主要给 PMU、IWDG 提供电，在上电后保持开启状态。

#### 4.1.5 1.5v 备份域

VBAT、VDD 是芯片电池模块供电电源端口，通过模拟选通信号选择备份域供电电源，主要为 RTC 和备份寄存器供电。

#### 4.1.6 1.5v 可关断电源域

1.5v 可关断电源域主要给芯片的内核、内存和外设提供供电，在上电后默认是开启状态，在进入低功耗时，芯片会硬件选择关闭该 1.5V 电源域，在唤醒后芯片会自动开启。主要有以下 3 种工作状态：

运转模式(Run Mode): 1.5V 电源域以正常的功耗模式运行，内存、外设都正常工作。

低功耗运转模式(Standby Mode): 1.5V 电源域以低功耗模式运行，内存、外设都以低功耗工作。

睡眠模式(Sleep Mode): 1.5V 电源域以正常的功耗模式工作，CPU 进入睡眠模式，内存、外设都以正常的功耗模式工作。

低功耗睡眠模式(Sleep Mode): 1.5V 电源域以低功耗睡眠模式工作，CPU 进入低功耗睡眠模式，内存、外设都以低功耗工作。

停机模式(Stop Mode): 1.5V 电源域以低功耗模式工作，只保持寄存器和 RAM 的内容。

深度停机模式(DeepStop Mode): 1.5V 电源域以更低功耗模式工作，只保持寄存器和 RAM 的内容。

待机模式(Standby Mode): 1.5V 电源域停止供电。除了备用电路和备份域外，寄存器和 SRAM 的内容全部丢失。

## 4.2 电源管理器

### 4.2.1 上电复位 (POR) 和掉电复位 (PDR)

芯片有一个完整的上电复位 (POR) 和掉电复位 (PDR) 电路，当供电电压达到芯片最低的工作电压系统能正常工作，当供电电压低于芯片最低的工作电压系统处于非工作状态。在对芯片进行上电或者掉电操作时，在上电操作时供电电压达到芯片最低的工作电压，芯片会产生上电复位，同样芯片在掉电操作时，跌落到一个电压，芯片会产生掉电复位。

当 VDD/VDDA 低于指定的限位电压 POR/PDR 时，系统保持为复位状态，NRST 复位引脚处于低电平，关于上电复位和掉电复位的细节请参考数据手册的电气特性部分。

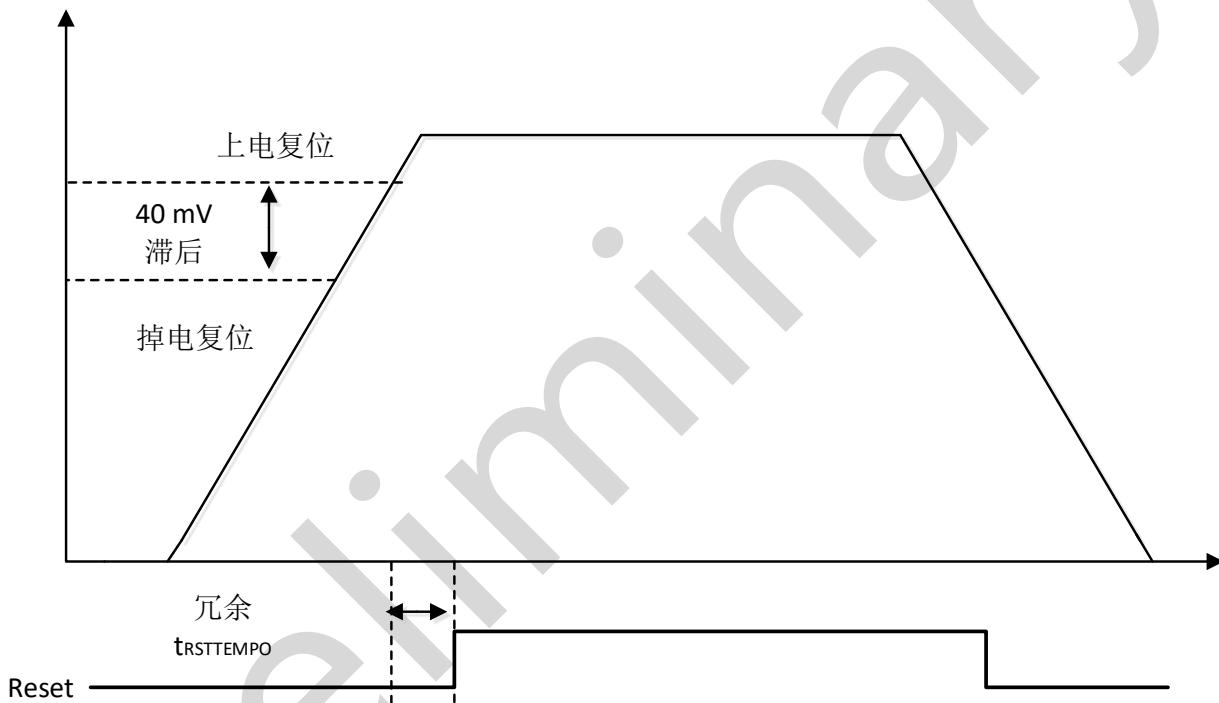


图 4-2 上电复位和掉电复位的波形图

### 4.2.2 可编程电压监测器 (PVD)

可编程电压监测器 PVD(Programmable Voltage Detector)是可以用来监视芯片的供电电压，在供电电压下降到给定的阀值以下时，产生一个中断，软件可以做紧急处理。当供电电压又恢复到给定的阀值以上时，也会产生一个中断，软件处理供电恢复。供电下降的阀值与供电上升的阀值有一个固定的差值，这就是 PVD 迟滞电压，通过列出的 PVD 阀值数据可以看到这个差别。引入这个差值的目的是为了防止电压在阀值上下小幅抖动，而频繁地产生中断。

用户可以通过软件设置电源控制寄存器 (SYSCFG\_PDETCSR) 中的 PLS 位的阈值电压和芯片供电电压进较，用来监控电源。

通过设置电源控制寄存器 (SYSCFG\_PDETCSR) 中的 PVDE 位来使能 PVD。

电源控制/状态寄存器 (SYSCFG\_PDETCSR) 中的 PVDO 标志用来表明 VDD 是高于还是低于电源控制寄存器 (SYSCFG\_PDETCSR) 中设置 PLS 位的阈值电压。

该事件在内部连接到外部中断的第 16 线，如果用户有配置外部中断的第 16 线，该事件就会产生中断，能够进入中断服务函数。当 VDD 下降到 PVD 阈值以下或当 VDD 上升到 PVD 阈值之上时，根据设置的外部中断第 16 线的上升/下降边沿触发，就会产生 PVD 中断（也可以通过软件配置产生 PVD 复位），用户可以在中断中做一些对应的操作。例如：当条件触发时，且掉电的速率慢于中断中处理程序的执行时间，如果系统需要进入特别保护状态，可以执行紧急关闭任务；保存系统一些重要数据，同时可以对外设进行相应的保护操作等。

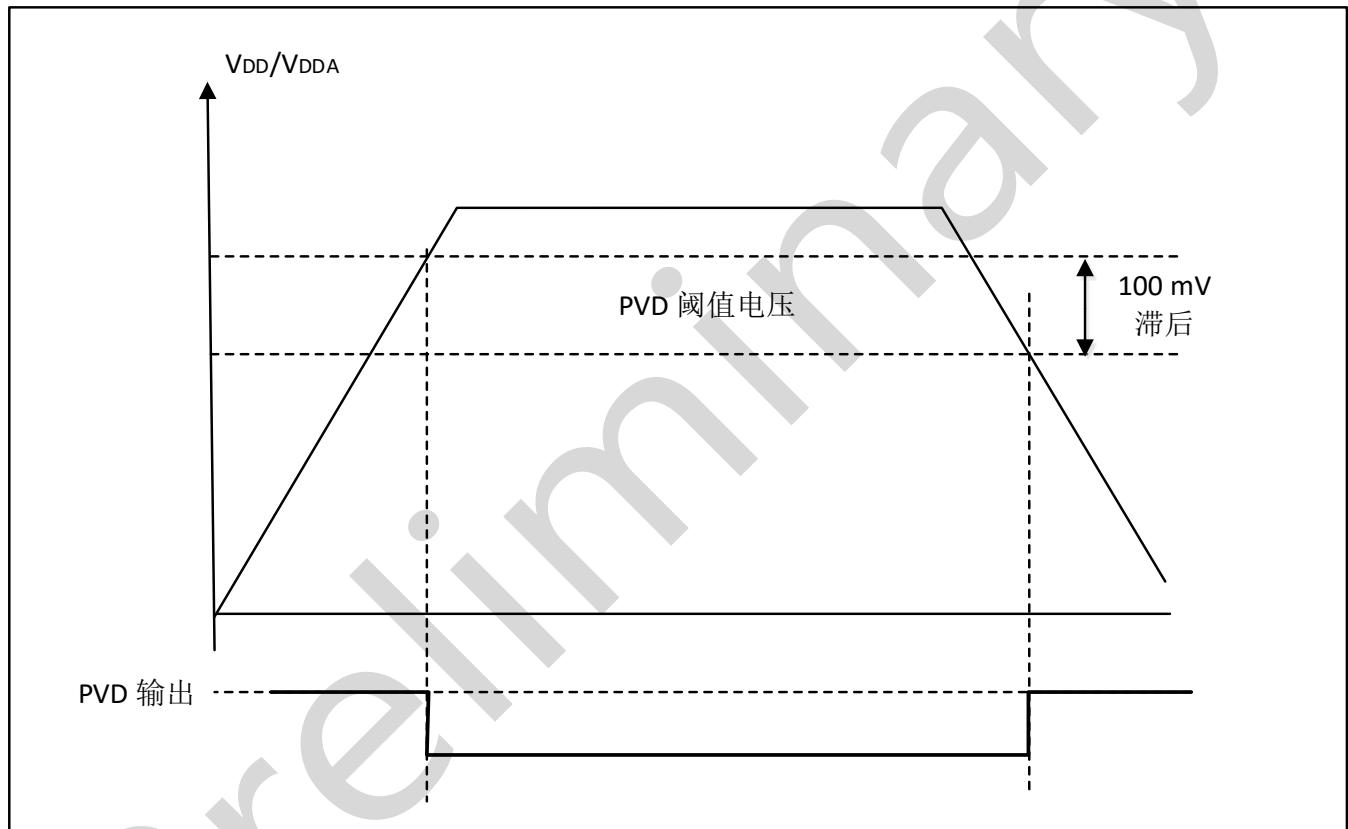


图 4-3 PVD 的阈值

### 4.3 功耗控制

为了延长电池供电类产品寿命，在 MCU 不需要工作时，可以利用 MCU 的多种低功耗模式来节省功耗，当需要 MCU 开始工作时，可以通过外部唤醒源或者低功耗定时器等方式唤醒 MCU 开始工作，从而达到分时工作的目的以节省产品的电流消耗。

芯片有六种低功耗模式，电源消耗不同、唤醒时间不同、唤醒源不同，用户需要根据应用需求，选择最佳的低功耗模式。

芯片有六种低功耗模式：

低功耗运行模式(Low Power Run Mode): CPU,所有芯片外设，如 NVIC、系统时钟 (SysTick) 等

在运行，工作时钟不能高于 2Mhz。

**睡眠模式(Sleep Mode):** CPU 停止，所有芯片外设包括 CPU 的外设，如 NVIC、系统时钟 (SysTick) 等仍在运行。

**低功耗睡眠模式(Low Power Sleep Mode):** CPU 停止，所有芯片外设包括 CPU 的外设，如 NVIC、系统时钟 (SysTick) 等仍在运行，工作时钟不能高于 2Mhz。

**停机模式(Stop Mode):** 1.5V 电源域以低功耗模式工作，只保持寄存器和 RAM 的内容，所有的外设时钟都停止。

**深度停机模式(Deep Stop Mode):** 1.5V 电源域以更低功耗模式工作，只保持寄存器和 RAM 的内容，所有的外设时钟都停止。

**待机模式(Standby Mode):** 1.5V 电源域停止供电，除了备用电路和备份域外，寄存器和 SRAM 的内容全部丢失。

此外，在运行模式下，可以通过以下方式中的一种降低功耗：

- 1) 降低系统时钟频率：在满足应用需求的同时可以选择低速时钟频率或采用高速时钟和低速时钟循环切换的方式来节省功耗。
- 2) 关闭 APB 和 AHB 总线上未被使用的外设时钟：用户只使能应用需要的时钟信号，其他的多余的时钟信号都选择关闭。
- 3) 选择低电压供电：供电电压越高芯片的耗电越大，所以在应用中在芯片安全的供电电压范围内可以选择合适的供电电压。

表 4-1 低功耗模式列表

模式	进入	唤醒方式	对 1.5V 电源域时钟的影响	对 VDD 区域时钟的影响	电压调节器	对数据和寄存器的影响	注意事项
低功耗运行模式 (Low Power Run Mode)	设置 LPR 位	清除 LPR 位	PLL,HSE 振荡器关闭 HIS,LSE 保持工作，系统工作时钟频率不超过 2Mhz	无	开		
睡眠模式 (Sleep Mode)	WFI (Wait for Interrupt)	任一中断	CPU 时钟关，对其他时钟和 ADC 时钟无影响	无	开		外设时钟继续维持，寄存器和 SRAM 的内容保持
	WFE (Wait for Event)	唤醒事件					

模式	进入	唤醒方式	对 1.5V 电源域时钟的影响	对 VDD 区域时钟的影响	电压调节器	对数据和寄存器的影响	注意事项
低功耗睡眠模式 (Low Power Sleep Mode)	设置 LPR 位; WFI (Wait for Interrupt)	任一中断	PLL,HSE 振荡器关闭 HIS,LSI,LSE 保持工作, 系统工作时钟频率不超过 2Mhz	无	开		外设时钟继续维持, 寄存器和 SRAM 的内容保持
	设置 LPR 位; WFE (Wait for Event)	唤醒事件					
停机模式 (Stop Mode)	LPDS 位; SLEEPDE EP 位; WFI 或 WFE;	任一外部中断 (在外部中断寄存器中设置)、RTC 闹钟事件、IWDG 不复位唤醒	所有使用 1.5V 的区域的时钟都已关闭	PLL、HSI 和 HSE 的振荡器关闭	开	寄存器和 SRAM 的内容维持, 所有的外设时钟都停止	进入低功耗前不使用的 GPIO 应该设置模拟输入状态
	PDDS 位;LPDS 位; SLEEPDE EP 位; WFI 或 WFE;	任一外部中断 (在外部中断寄存器中设置)、RTC 闹钟事件、IWDG 不复位唤醒					
深度停机模式 (DeepStop Mode)	PDDS 位; SLEEPDE EP 位; WFI 或 WFE;	WKUP 引脚、RTC 闹钟事件、NRST 引脚上的外部复位、IWDG 复位			开	寄存器和 SRAM 的内容维持, 所有的外设时钟都停止	进入低功耗前不使用的 GPIO 应该设置模拟输入状态
待机模式 (Standby Mode)	PDDS 位; SLEEPDE EP 位; WFI 或 WFE;				关	寄存器和 SRAM 的内容全部丢失, 所有的外设时钟都停止, 唤醒相当于芯片复位	

### 4.3.1 运行模式降低系统时钟

在满足应用需求的同时可以选择低速时钟频率或采用高速时钟和低速时钟循环切换的方式来节省功耗。

芯片的时钟系统可以灵活配置, 用户可以选择不同的的时钟频率作为系统时钟, 可以降低任意一个系统时钟 (SYSCLK、HCLK、PCLK1、PCLK2) 的速度。

进入睡眠模式前, 降低外设的时钟可以有效节省睡眠模式下的功耗。

### 4.3.2 外部时钟的控制

在芯片执行程序过程中，可以通过停止为外设和内存提供时钟总线（HCLK 和 PCLKx）来减少功耗。

使用睡眠模式时，可以在执行 WFI 或 WFE 指令前关闭外设的时钟，有效的降低睡眠模式下外设的电流消耗。

外设的时钟主要挂在 AHB 外设时钟使能寄存器（RCC\_AHBENR）、APB2 外设时钟使能寄存器（RCC\_APB2ENR）和 APB1 外设时钟使能寄存器（RCC\_APB1ENR）总线上，用户可以单独配置寄存器外设控制位关闭外设时钟。

### 4.3.3 低功耗运行模式（Low Power Run Mode）

进一步降低系统在运行模式下的功耗，可以将调节器配置在低功耗模式下。该模式下系统频率不应超过 2Mhz。

#### 进入低功耗运行模式

表 4-2 低功耗运行模式

低功耗运行模式	说明
进入	在以下条件下置位寄存器位 PWR_CR1.LPR: 系统频率低于 2Mhz
退出	清除寄存器位 PWR_CR1.LPR 等待 PWR_CR2.LPF 位清除后，增加系统时钟频率
唤醒延时	调整器从低功耗模式唤醒时间

### 4.3.4 睡眠模式（Sleep Mode）

#### 进入睡眠模式

WFI 表示 Wait for Interrupt，而 WFE 表示 Wait for Event。通过执行 WFI/WFE 指令，请求 MCU 进入睡眠模式。根据 CPU 系统控制寄存器（SCB->SCR）中的 SLEEPONEXIT 位的值，有两种选项可用于选择睡眠模式进入机制：

**SLEEP-NOW:** 如果 SLEEPONEXIT 位被清除，当 WFI 或 WFE 被执行时，微控制器立即进入睡眠模式。

**SLEEP-ON-EXIT:** 如果 SLEEPONEXIT 位被置位，系统从最低优先级的中断处理程序中退出时，微控制器就立即进入睡眠模式。

在睡眠模式下，所有的 I/O 引脚都保持在运行模式时的状态。

表 4-3 SLEEP NOW 模式

SLEEP NOW 模式		说明
进入		在以下条件下执行 WFI(Wait for Interrupt) 或 WFE(Wait for Event) 指令: SLEEPDEEP = 0 SLEEPONEXIT = 0
退出		如果执行 WFI 进入睡眠模式，中断（参考中断向量表） 如果执行 WFE 进入睡眠模式，唤醒事件（参考唤醒事件管理）
唤醒延时		立即唤醒

表 4-4 SLEEP ON EXIT 模式

SLEEP ON EXIT 模式		说明
进入		在以下条件下执行 WFI(Wait for Interrupt)指令： SLEEPDEEP = 0 SLEEPONEXIT = 1。
退出		中断（参考中断向量表）
唤醒延时		立即唤醒

#### 4.3.5 低功耗睡眠模式 (Low Power Sleep Mode)

##### 进入低功耗睡眠模式

WFI 表示 Wait for Interrupt，而 WFE 表示 Wait for Event。通过执行 WFI/WFE 指令，请求 MCU 从低功耗运行模式进入低功耗睡眠模式。根据 CPU 系统控制寄存器 (SCB->SCR) 中的 SLEEPONEXIT 位的值，有两种选项可用于选择低功耗睡眠模式进入机制：

SLEEP-NOW: 如果 SLEEPONEXIT 位被清除，当 WFI 或 WFE 被执行时，微控制器立即进入睡眠模式。

SLEEP-ON-EXIT: 如果 SLEEPONEXIT 位被置位，系统从最低优先级的中断处理程序中退出时，微控制器就立即进入睡眠模式。

在睡眠模式下，所有的 I/O 引脚都保持在运行模式时的状态。

表 4-5 Low Power SLEEP NOW 模式

Low Power SLEEP NOW 模式		说明

进入	在以下条件下执行 WFI(Wait for Interrupt) 或 WFE(Wait for Event) 指令:  SLEEPDEEP = 0  SLEEPONEXIT = 0  LPR=1
退出	如果执行 WFI 进入睡眠模式，中断（参考中断向量表）  如果执行 WFE 进入睡眠模式，唤醒事件（参考唤醒事件管理）
唤醒延时	立即唤醒

表 4-6 Low Power SLEEP ON EXIT 模式

Low Power SLEEP ON EXIT 模式 说明	
进入	在以下条件下执行 WFI(Wait for Interrupt)指令:  SLEEPDEEP = 0  SLEEPONEXIT = 1  LPR=1
退出	中断（参考中断向量表）
唤醒延时	立即唤醒

#### 4.3.6 停机模式 (Stop Mode)

CPU 深度睡眠模式+外设的时钟控制组成了停机模式。在停机模式下，在 1.5V 供电区域的所有时钟都被停止，PLL、HSI 和 HSE 振荡器的功能被禁止，SRAM 和寄存器内容被保留下。

在停机模式下，所有的 I/O 引脚都保持在运行模式时的状态。

##### 进入停机模式

通过对独立的控制位进行编程，停机模式根据唤醒的时钟源不同有两种控制方式，通过如下操作可以配置 MCU 进入停机模式：

- 1) 等待外部中断线 WFI 方式停机模式：配置电源控制寄存器 (PWR\_CR) 的 PDSS = 0 , LPDS = 0 ; 当 WFI 被执行时，微控制器立即进入停机模式。
- 2) 等待外部事件 WFE 方式停机模式：配置电源控制寄存器 (PWR\_CR) 的 PDSS = 0 , LPDS = 0 ; 当 WFE 被执行时，微控制器立即进入停机模式。

进入停机模式时可选择以下时钟源：

- 1) 独立看门狗 (IWDG)：可通过写入独立看门狗的键寄存器或硬件选择来启动独立看门狗，独立看门狗可以选择中断或者复位方式唤醒芯片，中断方式唤醒芯片后 MCU 继续执行进入低功

耗前的程序，复位方式唤醒后 MCU 执行复位；用户可以选择关闭 LSI 时钟源从而关闭独立看门狗。

2) 内部振荡器 (LSI 振荡器): 通过控制/状态寄存器 (RCC\_CSR) 的 LSION 位来设置。

3) 实时时钟(RTC) : 可以通过 RTC 对应的功能作为停机模式的唤醒源。

在停机模式下，如果在进入该模式前 ADC 没有被关闭，那么这些外设仍然消耗电流。通过设置寄存器 ADC\_ADCFG 的 ADEN 位为 0 可关闭这个外设。其他没有使用的 GPIO 需要设置模拟输入模式，否则有电流消耗。

### 退出停机模式

当由一个中断或唤醒事件导致退出停机模式时，HSI 振荡器被选为系统时钟，唤醒后系统时钟需用户重新配置。

当电压调节器处于运行模式下，系统从停机模式退出时，将会有段额外的启动延时。

表 4-7 停机模式

停机模式	说明
进入	<p>在以下条件下执行 WFI(Wait for Interrupt) 或 WFE(Wait for Event) 指令：</p> <p>置位 CPU 系统控制寄存器中的 SLEEPDEEP 位；</p> <p>复位电源控制寄存器 (PWR_CR) 中的 PDDE 位且置位 LPDS；</p> <p>系统时钟切换至 LSI 或 HSI；</p> <p>注：为了进入停机模式，所有的外部中断的请求位 (挂起寄存器 (EXTI_PEND)) 标志都必须被清除，否则停机模式的进入流程将会被跳过，程序继续运行。</p>
退出	<p>在以下条件下执行 WFI(Wait for Interrupt) 指令：</p> <p>任一外部中断引线被设置为中断模式 (相应的外部中断向量在 NVIC 中必须使能)，参见中断向量表 Wait for Event；</p> <p>在以下条件下执行 WFE(Wait for Event) 指令：</p> <p>任一外部中断线被设置为事件模式，例如 RTC 闹钟事情唤醒、看门狗中断；</p>
唤醒延时	LSI 或 HSI 的唤醒时间和电压调节器唤醒产生的额外时间
注意事项	在进入停机模式时需将不使用的 GPIO 设置成模拟输入模式

### 4.3.7 深度停机模式 (DeepStop Mode)

深度停机模式是在 CPU 深度睡眠模式的基础上结合了外设的时钟控制机制。在深度停机模式下，在 1.5V 供电区域的所有时钟都被停止，PLL、HSI 和 HSE 振荡器的功能被禁止，SRAM 和寄存器内容被保留下来。

在深度停机模式下，所有的 I/O 引脚都保持在运行模式时的状态。

#### 进入深度停机模式

通过对独立的控制位进行编程，深度停机模式根据唤醒的时钟源不同有两种控制方式，通过

如下操作可以配置 MCU 进入深度停机模式：

- 1) 等待外部中断线 WFI 方式深度停机模式：配置电源控制寄存器 (PWR\_CR) 的 PDDS = 0 , LPDS = 1 ; 当 WFI 被执行时，微控制器立即进入深度停机模式。
- 2) 等待外部事件 WFE 方式深度停机模式：配置电源控制寄存器 (PWR\_CR) 的 PDDS = 0 , LPDS = 1 ; 当 WFE 被执行时，微控制器立即进入深度停机模式。
- 3) 进入深度停机模式时可选择以下时钟源：
- 4) 独立看门狗 (IWDG)：可通过写入独立看门狗的键寄存器或硬件选择来启动独立看门狗，独立看门狗可以选择中断或者复位方式唤醒芯片，中断方式唤醒芯片后 MCU 继续执行进入低功耗前的程序，复位方式唤醒后 MCU 执行复位；用户可以选择关闭 LSI 时钟源从而关闭独立看门狗。
- 5) 内部振荡器 (LSI 振荡器)：通过控制/状态寄存器 (RCC\_CSR) 的 LSION 位来设置。
- 6) 实时时钟(RTC)：可以通过 RTC 对应的功能作为深度停机模式的唤醒源。

在深度停机模式下，如果在进入该模式前 ADC 没有被关闭，那么这些外设仍然消耗电流。通过设置寄存器 ADC\_ADCFG 的 ADEN 位为 0 可关闭这个外设。其他没有使用的 GPIO 需要设置模拟输入模式，否则有电流消耗。

### 退出深度停机模式

当由一个中断或唤醒事件导致退出深度停机模式时，HSI 振荡器被选为系统时钟，唤醒后系统时钟需用户重新配置。

当电压调节器处于运行模式下，系统从深度停机模式退出时，将会有一段额外的启动延时。

表 4-8 深度停机模式

深度停机模式	说明
进入	<p>在以下条件下执行 WFI(Wait for Interrupt) 或 WFE(Wait for Event) 指令：</p> <p>置位 CPU 系统控制寄存器中的 SLEEPDEEP 位；</p> <p>复位电源控制寄存器 (PWR_CR) 中的 PDDS 和 LPDS 位；</p> <p>系统时钟切换至 LSI 或 HSI；</p> <p>注：为了进入深度停机模式，所有的外部中断的请求位 (挂起寄存器 (EXTI_PEND)) 标志都必须被清除，否则深度停机模式的进入流程将会被跳过，程序继续运行。</p>
退出	<p>在以下条件下执行 WFI(Wait for Interrupt) 指令：</p> <p>任一外部中断引线被设置为中断模式 (相应的外部中断向量在 NVIC 中必须使能)，参见中断向量表 Wait for Event；</p> <p>在以下条件下执行 WFE(Wait for Event) 指令：</p> <p>任一外部中断线被设置为事件模式，例如 RTC 闹钟事情唤醒、看门狗中断；</p>
唤醒延时	LSI 或 HSI 的唤醒时间和电压调节器唤醒产生的额外时间
注意事项	在进入深度停机模式时需将不使用的 GPIO 设置成模拟输入模式

### 4.3.8 待机模式(Standby Mode)

待机模式是在 CPU 深睡眠模式时关闭电压调节器，整个 1.5V 可关断域被断电，PLL、HSI 和 HSE 振荡器也被断电，SRAM 和寄存器内容丢失，只有备份的寄存器和待机电路维持供电。待机模式是芯片的最低功耗模式，唤醒后程序重新开始执行。

#### 进入待机模式

可以通过设置独立的控制位，可选择以下时钟源：

- 1) 独立看门狗 (IWDG): 可通过写入独立看门狗的键寄存器或硬件选择来启动独立看门狗，可以选择独立看门狗的复位唤醒方式来唤醒芯片，唤醒后程序重新开始执行；用户可以选择关闭 LSI 时钟源从而关闭独立看门狗。
- 2) 内部低速振荡器 (LSI 振荡器): 通过控制/状态寄存器 (RCC\_CSR) 的 LSION 位来设置。
- 3) 实时时钟(RTC) : 可以通过 RTC 对应的功能作为停机模式的唤醒源。

表 4-9 待机模式

待机模式	说明
进入	在以下条件下执行 WFI(Wait for Interrupt) 或 WFE(Wait for Event) 指令： 置位 CPU 系统控制寄存器中的 SLEEPDEEP 位； 置位电源控制寄存器 (PWR_CR) 中的 PDSS 位； 复位电源控制/状态寄存器 (PWR_CSR) 中的 WUF 位；
退出	WKUP 引脚上升沿或者下降沿、RTC 闹钟事件、NRST 引脚上外部复位、IWDG 复位
唤醒延时	复位阶段时电压调节器的启动
注意事项	在进入待机模式时需配置正确的唤醒源，否则进入待机模式后且无唤醒源可能无法二次下载

#### 待机模式下的输入/输出端口状态

在待机模式下，所有的 I/O 引脚处于高阻态，除了以下的引脚：

- 1) 复位引脚
- 2) 被使能的唤醒引脚，用户需要正确的配置对应的唤醒边沿信号

#### 调试模式

默认情况下，如果在进行调试微处理器时，使微处理器进入停止或待机模式，将失去调试连接，这是因为 CPU 内核失去了时钟，因此需要设置正确的唤醒源。

## 4.4 功能描述

### 4.4.1 滤波功能

配置 PWR\_CR4 和 PWR\_CR5 寄存器 FILTSEL0、FILTSEL1 选择需要过滤的唤醒引脚，FILTCNT0、FILTCNT1 选择唤醒源过滤宽度，通过配置 FILTE0、FILTE1 使能位打开滤波器。滤波器

将滤除小于配置过滤宽度的唤醒信号。

#### 4.4.2 快速唤醒功能

配置 PWR\_CR6.STDBY\_FS\_WK, 可以控制从唤醒信号产生到唤醒信号有效需要的时钟周期。

### 4.5 电源控制寄存器

表 4-10 电源控制寄存器概览

Offset	Acronym	RegisterName	Reset
0x00	PWR_CR1	电源控制寄存器 1	0x00000000
0x04	PWR_CSR	电源控制状态寄存器	0x00000000
0x08	PWR_CR2	电源控制寄存器 2	0x00000000
0x0C	PWR_CR3	电源控制寄存器 3	0x00000000
0x10	PWR_CR4	电源控制寄存器 4	0x00000000
0x14	PWR_CR5	电源控制寄存器 5	0x00000000
0x18	PWR_CR6	电源控制寄存器 6	0x00000000
0x1C	PWR_SR	电源状态寄存器	0x00000000
0x18	PWR_SCR	电源配置寄存器	0x00000000

#### 4.5.1 电源控制寄存器 1 (PWR\_CR1)

地址偏移: 0x00

复位值: 0x00000000(从待机模式唤醒时清除)

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Res																
Type																	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	VOS	LPR	Res												CS	Re	PD
Type	rw	rw													s	DS	LPD
																	S

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0。

Bit	Field	Type	Reset	Description
15:14	VOS	rw	0x0	<p>调制器电压输出选择</p> <p>00: 保留</p> <p>01: 1.6V</p> <p>10: 1.7V</p> <p>11: 1.8V</p>
13	LPR	rw		<p>低功耗配置位，置位后调节器从正常模式切换到低功耗模式。</p> <p>0: 正常运行模式</p> <p>1: 低功耗运行模式</p>
7:4	Reserved			保留，始终读为 0。
3	CSBF	w	0x0	<p>清除待机位 (Clear Standby Flag)</p> <p>始终读出为 0</p> <p>1: 清除 SBF 待机位 (写)</p> <p>0: 无功效</p>
2	Reserved			保留，始终读为 0。
1	PDDS	rw	0x0	<p>掉电深睡眠(Power down deepsleep)</p> <p>1: CPU 进入深睡眠时进入待机模式</p> <p>0: CPU 进入深睡眠时进入停机模式</p>
0	LPDS	rw	0x0	<p>深睡眠下的低功耗</p> <p>PDDS = 0 时，与 PDDS 位协同操作</p> <p>1: 进入停机模式时，电压调节器处于低功耗模式</p> <p>0: 进入停机模式时，电压调节器处于正常功耗模式</p> <p>当进入停机模式时，LPDS = 1 时的电流小于 LPDS = 0 时的电流。详见该芯片对应的数据手册</p>

#### 4.5.2 电源控制/状态寄存器 (PWR\_CSR)

地址偏移: 0x04

复位值: 0x00000000(从待机模式唤醒时不被清除)

与标准的 APB 读相比，读次寄存器需要额外的 APB 周期

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Res																
Type																	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Res						VOS RDY	Res									
Type							r										

Bit	Field	Type	Reset	Description
31:15	Reserved			保留，始终读为 0
14	VOSRDY	rw	0x0	电压调制器输出准备完成标志位 1: 准备完成 0: 准备未完成
13:2	Reserved			保留，始终读为 0
1	SBF	r	0x0	待机标志 (Standby Flag) 该位由硬件设置，并只能由 POR/PDR (上电/掉电复位) 或设置电源控制寄存器 (PWR_CR) 的 CSBF 位清除。 1: 系统进入待机模式 0: 系统不在待机模式
0	Reserved	r		保留，始终读为 0

#### 4.5.3 电源控制寄存器 2 (PWR\_CR2)

地址偏移: 0x08

复位值: 0x00000000

寄存器在退出待机模式时不进行复位，也不会在 RCC\_APBRSTR1.PWRRST 有效后复位

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

Field	EN WU	Res		EW UP 6	EW UP 5	EW UP 4	EW UP 3	EW UP 2	EW UP1
Type	rw			rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0。
15	ENWU			使能 WKUP (Enable WakeUp Pin) 1: WKUP 唤醒功能有效 0 : WKUP 唤醒功能无效
14:6	Reserved			保留, 始终读为 0。
5:0	EWUP1~6	rw	0x0	使能 WKUP1~6 引脚 (Enable WakeUp Pin1~6) 1: WKUP 引脚用于将 CPU 从待机模式唤醒, WKUP 引脚被强置为输入下拉的配置 (WKUP 引脚上的上升沿将系统从待机模式唤醒) 0 : WKUP 引脚为通用 I/O。WKUP 引脚上的事件不能将 CPU 从待机模式唤醒 注: 在系统复位时清除这一位。

#### 4.5.4 电源控制寄存器 3 (PWR\_CR3)

地址偏移: 0x0C

复位值: 0x00000000

寄存器在退出待机模式时不进行复位, 也不会在 RCC\_APBRSTR1.PWRRST 有效后复位

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

Bit	Field	Type	Reset	Description
31:6	Reserved			保留, 始终读为 0。

Bit	Field	Type	Reset	Description
5:0	WP1~6	rw	0x0	<p>WKUP1~6 唤醒引脚极性 定义了 WKUP1~WKUP6 外部唤醒 pin 上用于事件检测的极性 0 = 高电平检测(上升沿), 唤醒引脚必须先配置为低电平 1 = 低电平检测(下降沿), 唤醒引脚必须先配置为高电平</p>

#### 4.5.5 电源控制寄存器 4 (PWR\_CR4)

地址偏移: 0x10

复位值: 0x00000F00 访问: 写寄存器需要额外的 3 个 APB 周期, 读需要额外的 2 个 APB 周期。

寄存器在退出待机模式时不进行复位, 也不会在 RCC\_APBRSTR1.PWRRST 有效后复位

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	FILTCNT0										Res		FILT F0	FILTE0	FILTSEL0	
Type	rw										w1c		rw	rw		

Bit	Field	Type	Reset	Description
31:6	Reserved			保留, 始终读为 0
15:8	FILTCNT0	rw	0x0	<p>滤波器 0 计数器值: 0、1、2、3...、254、255</p>
7:5	Reserved			保留, 始终读为 0。
4	FILTF0	w1c	0x0	<p>滤波器 0 滤波有效标志位 1: 唤醒源经过滤波器 0 0: 唤醒源不经过滤波器 0</p>

Bit	Field	Type	Reset	Description
3:2	FILTE0	rw	0x0	<p>滤波器 0 使能 (Filter Enable)</p> <p>00: 滤波器 0 无效</p> <p>01: 滤波器 0 上沿滤波</p> <p>10: 滤波器 0 下沿滤波</p> <p>其他: 保留</p>
1:0	FILTSEL0	rw	0x0	<p>WKUP 引脚滤波选择位 (WakeUp Pin Filt Select Bits)</p> <p>00: 选择 WKUP 引脚 1</p> <p>01: 选择 WKUP 引脚 2</p> <p>10: 选择 WKUP 引脚 3</p> <p>其他: 保留</p>

#### 4.5.6 电源控制寄存器 5 (PWR\_CR5)

地址偏移: 0x14

复位值: 0x00000F00 访问: 写寄存器需要额外的 3 个 APB 周期, 读需要额外的 2 个 APB 周期。

寄存器在退出待机模式时不进行复位, 也不会在 RCC\_APBRSTR1.PWR\_RST 有效后复位

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	FILTCNT1										Res		FILT F1	FILTE1	FILTSEL1	
Type	rw										w1c		rw	rw	rw	

Bit	Field	Type	Reset	Description
31:6	Reserved			保留, 始终读为 0
15:8	FILTCNT1	rw	0x0	<p>滤波器 1 计数器值:</p> <p>0、1、2、3、...、254、255</p>
7:5	Reserved			保留, 始终读为 0。
4	FILTF1	w1c	0x0	<p>滤波器 1 滤波有效标志位</p> <p>1: 唤醒源经过滤波器 1</p> <p>0: 唤醒源不经过滤波器 1</p>

Bit	Field	Type	Reset	Description
3:2	FILTE1	rw	0x0	<p>滤波器 1 使能 (Filter Enable)</p> <p>00: 滤波器 1 无效</p> <p>01: 滤波器 1 上沿滤波</p> <p>10: 滤波器 1 下沿滤波</p> <p>其他: 保留</p>
1:0	FILTSEL1	rw	0x0	<p>WKUP 引脚滤波选择位 (WakeUp Pin Filt Select Bits)</p> <p>00: 选择 WKUP 引脚 4</p> <p>01: 选择 WKUP 引脚 5</p> <p>10: 选择 WKUP 引脚 6</p> <p>其他: 保留</p>

#### 4.5.7 电源控制寄存器 6 (PWR\_CR6)

地址偏移: 0x18

复位值: 0x00000000 访问: 写寄存器需要额外的 3 个 APB 周期, 读需要额外的 2 个 APB 周期。

寄存器在退出待机模式时不进行复位, 也不会在 RCC\_APBRSTR1.PWRRST 有效后复位

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

Bit	Field	Type	Reset	Description
31:6	Reserved			保留, 始终读为 0
2:0	STDBY_FS_WK	rw	0x0	<p>STDBY_FS_WK: 快速唤醒 Standby 模式选择位</p> <p>000: 1 个 LSI 周期唤醒</p> <p>001: 2 个 LSI 周期唤醒</p> <p>010: 3 个 LSI 周期唤醒</p> <p>011: 4 个 LSI 周期唤醒</p> <p>000: 5 个 LSI 周期唤醒</p> <p>001: 6 个 LSI 周期唤醒</p> <p>110: 7 个 LSI 周期唤醒</p>

Bit	Field	Type	Reset	Description
				111: 8 个 LSI 周期唤醒

#### 4.5.8 电源状态寄存器 (PWR\_SR)

地址偏移: 0x1C

复位值: 0x00000000

寄存器在退出待机模式时不进行复位，也不会在 RCC\_APBRSTR1.PWRRST 有效后复位

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

Bit	Field	Type	Reset	Description
31:6	Reserved			保留, 始终读为 0。
5:0	WUF1~6	r	0x0	唤醒源 1~6 的唤醒标志位(WakeUp Source1~6 Flag) 1: 检测到 WKUP 引脚有唤醒事件 0: 未检测到 WKUP 引脚有唤醒事件

#### 4.5.9 电源状态清除寄存器(PWR\_SCR)

地址偏移: 0x20

复位值: 0x00000000

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

Bit	Field	Type	Reset	Description
31:6	Reserved			保留, 始终读为 0。
5:0	CWUF1~6	w	0x0	清除唤醒源 1~6 的唤醒标志位 (Clear WakeUp Source1~6 Flag) 1: 清除 PWR_SR 寄存器对应的唤醒标志位 0: 无

#### 4.5.10 电源配置寄存器(PWR\_CFGR)

地址偏移: 0x24

复位值: 0x000000160

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res					LSICAL					LSICALSEL					
Type						rw					w					

Bit	Field	Type	Reset	Description
31:6	Reserved			保留, 始终读为 0。
9:5	LSICAL	rw	0xB	内部低速时钟 LSI 校准值 (Internal Low-speed Calibration ) 系统启动时, 该位初始化为出厂校准值 若 LSICALSEL=0x1F, 可以重新写入校准值校正 LSI 时钟
4:0	LSICALSEL	w	0x0	内部低速时钟 LSI 校准值选择(Internal Low-speed Calibration Select ) 0x1F: 选择寄存器位 LSICAL 的值 其他: 选择出厂校准值

## 5 时钟和复位(RCC)

### 5.1 复位单元

#### 5.1.1 简介

复位共三大类：电源复位、系统复位和备份域复位。

#### 5.1.2 功能框图

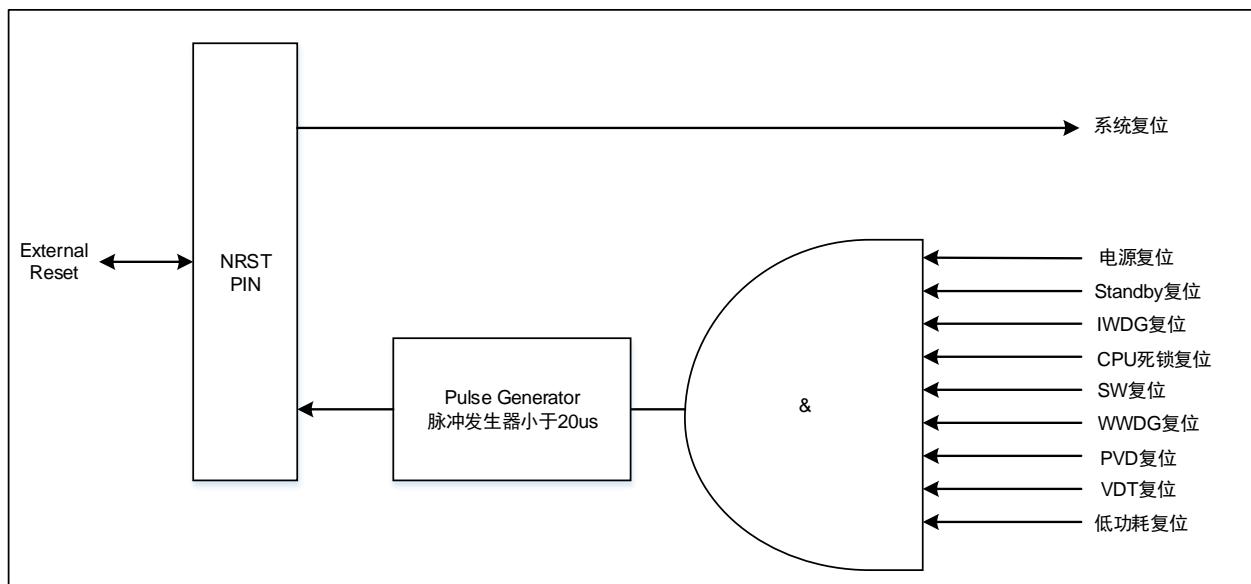


图 5-1 复位功能框图

#### 5.1.3 主要特征

- 1) 复位来源：通过控制状态寄存器（RCC\_CSR）中的复位标志位来进行判断；
- 2) 系统复位：除了时钟控制寄存器（RCC\_CSR）中的复位标志以及内部低速振荡器使能标志、电源控制寄存器（PWR\_CSR）中的待机和唤醒标志、DBG 控制寄存器（DBG\_CR）、备份区域中的寄存器不受系统复位影响，其余寄存器都将被系统复位；
- 3) 上电复位：复位所有寄存器。

#### 5.1.4 功能描述

##### 5.1.4.1 电源复位

电源复位有以下几种：

- 1) 上电复位

##### 5.1.4.2 系统复位

系统复位有以下几种：

- 1) NRST 复位
- 2) WWDG 复位

- 3) IWDG 复位
- 4) Standby 复位
- 5) 软件复位 (SCB\_AIRCR[SYSRESETREQ] = 1)
- 6) CPU 死锁复位
- 7) PVD 复位
- 8) VDT 复位
- 9) 低功耗复位

#### 5.1.4.3 备份域复位

备份域复位有以下几种:

- 1) 上电复位
- 2) 配置备份区域控制寄存器 RCC\_BDCR 中的 BDRST 位

### 5.2 时钟单元

#### 5.2.1 简介

五个都可独立开启或关闭的时钟源:

- 1) 高速内部时钟 (HSI)
- 2) 高速外部时钟 (HSE)
- 3) 锁相环 (PLL)
- 4) 低速内部时钟 (LSI)
- 5) 低速外部时钟 (LSE)

## 5.2.2 功能框图

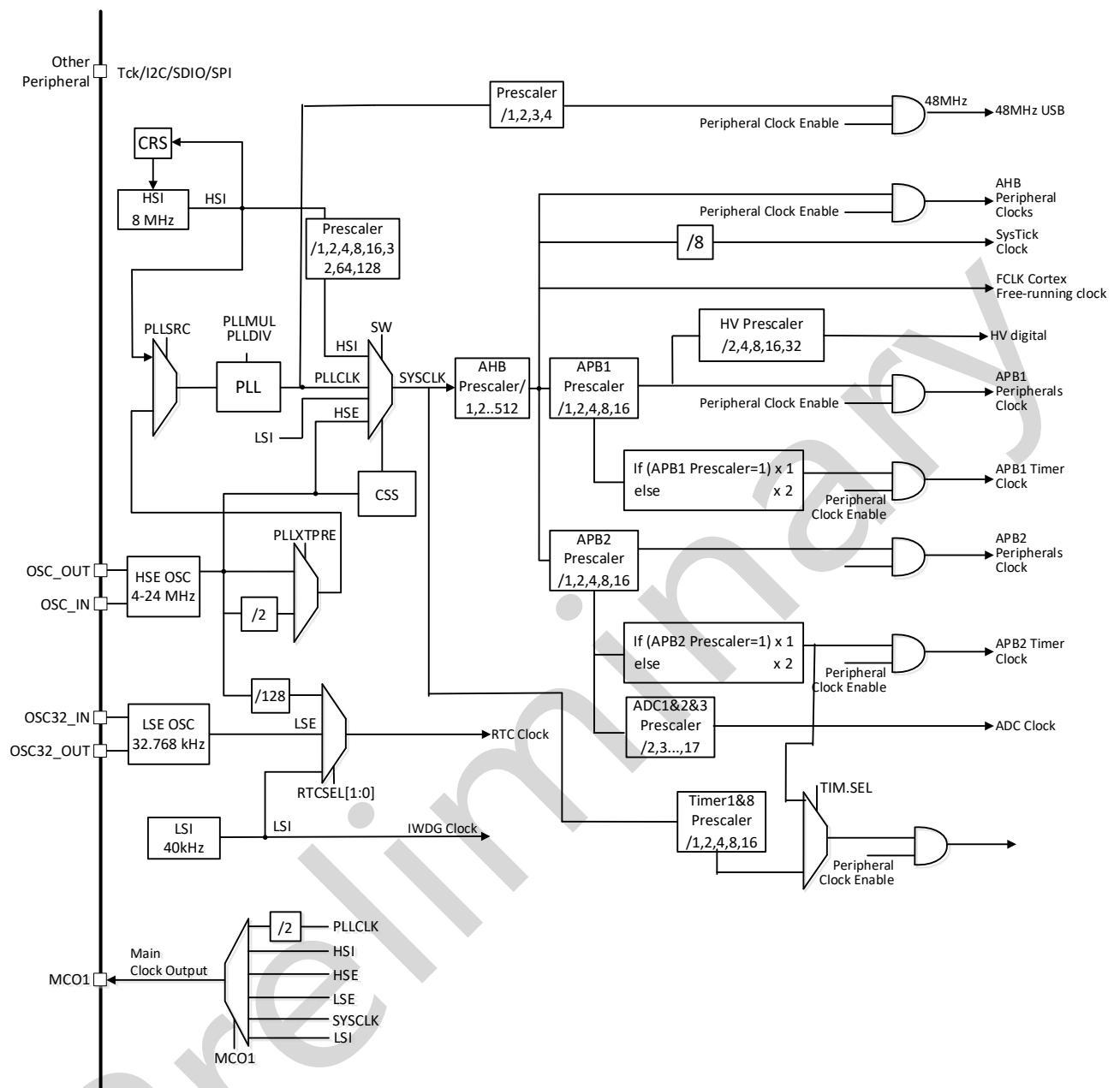


图 5-2 时钟树

## 5.2.3 主要特征

通过时钟配置寄存器（RCC\_CFGR）的预分频器来分别配置 AHB、APB2 和 APB1 域的频率。AHB 和 APB1，APB2 域的最大频率是 120MHz。

## 5.2.4 功能描述

### 5.2.4.1 外部高速时钟 (HSE)

#### 5.2.4.1.1 描述

高速外部时钟的时钟源有以下两种：

### 1) 外部晶振输入

时钟中断：当时钟控制寄存器（RCC\_CR）中的 HSERDY 位由硬件拉高，配置时钟中断寄存器（RCC\_CIR）的 HSERDYIE 位，会产生中断。

优点：外部振荡器时钟非常精确。

### 2) 外部时钟输入

通过配置时钟控制寄存器（RCC\_CR）HSEBYP 和 HSEON 位来选择此模式。

此模式具有以下特征：

- a) 必须提供外部时钟源或晶体振荡器；
- b) 输入频率范围 4~24MHz；
- c) 具有 50% 占空比的外部时钟信号（方波、正弦波）；
- d) 必须驱动 OSC\_IN 管脚，而 OSC\_OUT 引脚在未被使用时应该保持悬空。

注：当使用外部时钟输入时，必须将时钟控制寄存器（RCC\_CR）的 HSEBYP 位置“1”

#### 5.2.4.1.2 局部模块框图

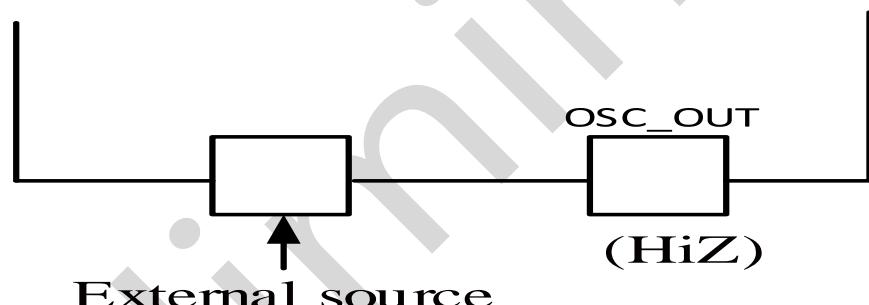


图 5-3 外部时钟

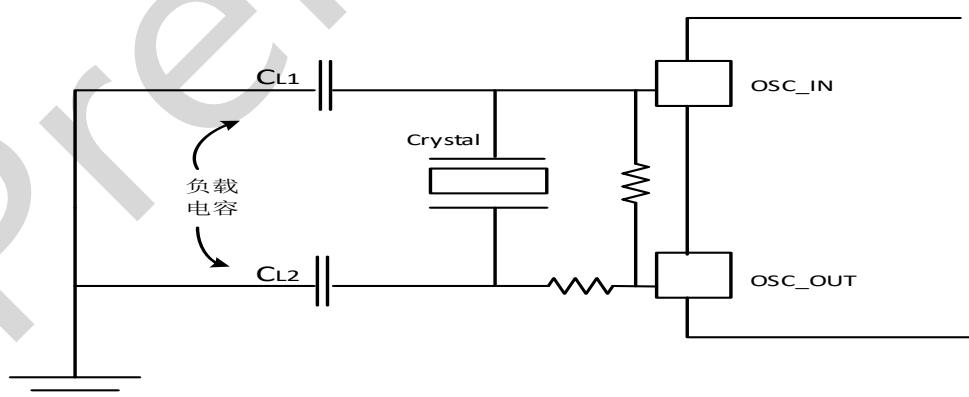


图 5-4 晶振/陶瓷谐振器

上图“外部时钟”为外部时钟输入模块框图；上图“晶振/陶瓷谐振器”为外部晶振输入模块框图。

#### 5.2.4.1.3 配置流程

启用 HSE 配置流程如下：

- 1) 配置时钟控制寄存器（RCC\_CR）中的 HSEON 位置“1”，启用 HSE；

- 2) 等待时钟控制寄存器（RCC\_CR）中的 HSERDY 位被拉高，表示 HSE 准备就绪。只有当 HSERDY 位被拉高，HSE 时钟才会输出有效时钟信号。

### 5.2.4.2 内部高速时钟（HSI）

HSI 时钟信号由内部 8MHz 振荡器产生，HSI 时钟源在芯片上电后默认启用。

时钟中断：当时钟控制寄存器（RCC\_CR）中的 HSIRDY 位由硬件拉高，配置时钟中断寄存器（RCC\_CIR）的 HSIRDYIE 位，会产生中断。

优点：HSI 振荡器能够在不需要任何外部器件的条件下提供系统时钟，它的启动时间也比 HSE 晶体振荡器快。

缺点：即使经过校准，频率也不如外部晶体振荡器或陶瓷谐振器精确。

作用：

- 1) 可直接用作系统时钟作为 PLL 输入；

如果 HSE 晶体振荡器发生故障，HSI 信号也可以用作备用源时钟源，参考时钟安全系统（CSS）章节。

使能 HSI 配置步骤：

- 1) 配置时钟控制寄存器（RCC\_CR）中的 HSION 位置“1”，使能 HSI；
- 2) 等待时钟控制寄存器（RCC\_CR）中的 HSIRDY 位被拉高，表示 HSI 准备就绪，只有当 HSIRDY 位被拉高，HSI 时钟才会释放。

### 5.2.4.3 锁相环（PLL）

PLL 的输入参考时钟有以下三种，可以通过时钟控制寄存器（RCC\_CFGR）中的 PLLXTPRE 和 PLLSRC 进行选择：

- 1) HSI 振荡器输出时钟；
- 2) HSE 晶体输出时钟；
- 3) HSE 晶体输出时钟的两分频。

时钟中断：当时钟控制寄存器（RCC\_CR）中的 PLLRDY 位由硬件拉高，配置时钟中断寄存器（RCC\_CIR）的 PLL\_RDYIE 位，会产生中断。

PLL 的输入时钟  $f_{PLL\_IN}$  和  $f_{PLL\_OUT}$  之间关系为

$$\frac{f_{PLL\_IN}}{\text{PLLDIV}[2:0] + 1} = \frac{f_{PLL\_OUT}}{\text{PLLMUL}[6:0] + 1}$$

PLLMUL[6:0]和 PLLDIV[2:0] 是 PLL 的倍频分频器和输出分频器的分频比设置。

注意事项：

- 1) 在启用 PLL 之前，必须进行 PLL 配置（选择 PLL 输入时钟和选择倍频因子和分频因子）。一旦 PLL 启用，PLL 配置就不能更改。如需更改配置，必须先关闭 PLL 使能位；
- 2) 如果在应用中需要使用 USB 接口，则需要 48MHz 的 USB 时钟，因此 PLL 必须被配置

为 48 或 96MHz。

使能 PLL 配置步骤：（选择 HSI 为 PLL 时钟源）

- 1) 配置时钟控制寄存器 (RCC\_CR) 中的 HSION 位置"1", 使能 HSI;
- 2) 等待时钟控制寄存器 (RCC\_CR) 中的 HSIRDY 位被拉高, 表示 HSI 准备就绪, 只有当 HSIRDY 位被拉高, HSI 时钟才会释放;
- 3) 配置时钟控制寄存器 (RCC\_PLLCFGR) 中 PLLMUL (倍频系数)、PLLDIV (分频系数);
- 4) 配置时钟控制寄存器 (RCC\_CR) 中的 PLLON 位置"1", 使能 PLL;
- 5) 等待时钟控制寄存器 (RCC\_CR) 中的 PLLRDY 位被拉高, 表示 PLL 准备就绪, 只有当 PLLRDY 位被拉高, PLL 时钟才会释放。

使能 PLL 配置步骤：（选择 HSE 为 PLL 时钟源）

- 1) 配置时钟控制寄存器 (RCC\_CR) 中的 HSEON 位置"1", 使能 HSE;
- 2) 等待时钟控制寄存器 (RCC\_CR) 中的 HSERDY 位被拉高, 表示 HSE 准备就绪, 只有当 HSERDY 位被拉高, HSE 时钟才会释放;
- 3) 配置时钟控制寄存器 (RCC\_PLLCFGR) 中的 PLLSRC 位置"1", 选择 HSE 时钟用作 PLL 输入时钟;
- 4) 配置时钟控制寄存器 (RCC\_PLLCFGR) 中的 PLLXTPRE 位, 选择 HSE 或者 HSE2 分频为 PLL 输入时钟;
- 5) 配置时钟控制寄存器 (RCC\_PLLCFGR) 中 PLLMUL (倍频系数)、PLLDIV (分频系数);
- 6) 配置时钟控制寄存器 (RCC\_CR) 中的 PLLON 位置"1", 使能 PLL;
- 7) 等待时钟控制寄存器 (RCC\_CR) 中的 PLLRDY 位被拉高, 表示 PLL 准备就绪, 只有当 PLLRDY 位被拉高, PLL 时钟才会释放。

#### 5.2.4.4 外部低速时钟 (LSE)

LSE 是配合外置 32.768kHz 晶体振荡器产生低速时钟的时钟源。。

时钟中断：当备份域控制寄存器 (RCC\_BDCR) 中的 LSERDY 位由硬件拉高, 配置时钟中断寄存器 (RCC\_CIR) 的 LSE\_RDYIE 位, 会产生中断。

优点：为实时时钟或者其他定时功能提供低功耗且高精度的时钟源。

使能 LSE 配置步骤：

- 1) 配置备份域控制寄存器 (RCC\_BDCR) 的 BDRST 位置"1"备份域软件复位;
- 2) 配置备份域控制寄存器 (RCC\_BDCR) 的 BDRST 位置"0"清除备份域软件复位;
- 3) 配置电源控制寄存器 (RCC\_BDCR) 的 BDP 位置"1", 取消后备区域的写保护;
- 4) 配置备份域控制寄存器 (RCC\_BDCR) 的 LSEON 位置"1", 使能 LSE;
- 5) 等待备份域控制寄存器 (RCC\_BDCR) 的 LSERDY 位被拉高, 表示 LSE 准备就绪, 只有当 LSERDY 位被拉高, LSE 时钟才会释放。

#### 5.2.4.5 内部低速时钟 (LSI)

LSI 作为一个低功耗时钟源, 可以保持在独立看门狗 (IWDG) 和自动唤醒单元 (AWU) 的停止和

待机模式下运行。时钟中心频率在 40kHz 左右。详情请参阅数据手册中有关电气特性部分

时钟中断：当控制状态寄存器（RCC\_CSR）中的 LSIRDY 位由硬件置位，配置时钟中断寄存器（RCC\_CIR）的 LSI\_RDYIE 位，会产生中断。

使能 LSI 配置步骤：

- 1) 配置控制状态寄存器（RCC\_CSR）的 LSION 位置"1"，使能 LSI；
- 2) 等待控制状态寄存器（RCC\_CSR）中的 LSIRDY 位被拉高，表示 LSI 准备就绪，只有当 LSIRDY 位被拉高，LSI 时钟才会释放。

#### 5.2.4.6 时钟选择（SWS）

四个系统时钟源：

- 1) 高速内部时钟（上电后默认）
- 2) 高速外部时钟（HSE）
- 3) 锁相环（PLL）：当 HSI（或 HSE）作为 PLL 时钟源时，HSI（或 HSE）不可关闭
- 4) 低速内部时钟（LSI）

系统时钟配置步骤：

- 1) 使能需要的系统时钟源（HSI, HSE, PLL, LSI），每个时钟使能方式不同，具体方式请查看（HSI, HSE, PLL, LSI 章节）；
- 2) 等待被选择的时钟源 RDY 信号被拉高，表示系统时钟源准备就绪；（当目标时钟源准备就绪后，系统时钟才会发生切换）；
- 3) 通过配置时钟配置寄存器（RCC\_CFGR）的 SW 位来选择系统时钟；
- 4) 通过读取时钟配置寄存器（RCC\_CFGR）的 SWS 位，判断当前系统时钟的时钟源。

#### 5.2.4.7 外设复位（RSTR）

可以通过 APB2 外设复位寄存器（RCC\_APB2RSTR）、APB1 外设复位寄存器（RCC\_APB1RSTR）和 AHB 外设复位寄存器（RCC\_AHB1RSTR）、（RCC\_AHB2RSTR）、（RCC\_AHB3RSTR）来实现相应外设的软件复位。

#### 5.2.4.8 时钟安全系统（CSS）

软件启动：通过时钟控制寄存器（RCC\_CR）的 CSSON 位启动时钟安全系统。时钟检测器在 HSE 振荡器启动后延迟启用，当此振荡器停止时禁用。

触发中断：当检测到 HSE 时钟故障时，CSS 将自行停用 HSE 振荡器，将时钟故障事件发送到高级定时器的刹车输入端，并触发时钟安全系统中断 CSS，请求软件执行救援操作。CSS 中断连接到 CPU 的 NMI 中断。

中断判断标志及清除方式：当时钟中断寄存器（RCC\_CIR）中的 CSSF 位由硬件置位，表示 HSE 时钟失效导致了时钟安全系统中断。通过软件将时钟中断寄存器（RCC\_CIR）中 CSSC 置'1'来清除安全系统中断标志位 CSSF。

**注意事项 1：**一旦启用 CSS，如果 HSE 时钟发生故障，CSS 中断发生，并自动生成 NMI 中断。除非 CSS 中断挂起位被清除，否则 NMI 将被不停地执行。因此，在 NMI 的处理程序中，用户必须通过在时钟中断寄存器 (RCC\_CIR) 中设置 CSSC 位来清除 CSS 中断。

**注意事项 2：**当故障发生时，如果直接使用 HSE 振荡器作为系统时钟，系统时钟将自动切换到高速内部振荡器并禁用 HSE 振荡器；如果 HSE 振荡器用作 PLL 输入时钟，PLL 时钟用作系统时钟，则 PLL 也被禁用。

HSE 当系统时钟配置使用步骤：

- 1) 配置时钟控制寄存器(RCC\_CR)中的 HSEON 位置”1”，使能 HSE;
- 2) 等待时钟控制寄存器(RCC\_CR) 中的 HSERDY 位被拉高，表示 HSE 准备就绪，只有当 HSERDY 位被拉高，HSE 时钟才会释放。
- 3) 通过配置时钟配置寄存器 (RCC\_CFGR)的 SW 位为 2'b01 来选择 HSE 为系统时钟;
- 4) 通过读取时钟配置寄存器 (RCC\_CFGR)的 SWS 位，判断当前系统时钟的时钟源是否为 HSE。
- 5) 配置控制状态寄存器 (RCC\_CSR)的 LSION 位置”1”和 LSI\_OEN\_LV 位置”1”，使能 LSI。
- 6) 等待控制状态寄存器 (RCC\_CSR) 中的 LSIRDY 位被拉高，表示 LSI 准备就绪，只有当 LSIRDY 位被拉高，LSI 时钟才会释放。
- 7) 配置时钟控制寄存器(RCC\_CR)中的 CSSON 位置”1”，时钟安全系统使能。

HSE 间接当系统时钟配置使用步骤：

- 1) 配置时钟控制寄存器 (RCC\_CR) 中的 HSEON 位置”1”，使能 HSE;
- 2) 等待时钟控制寄存器 (RCC\_CR) 中的 HSERDY 位被拉高，表示 HSE 准备就绪，只有当 HSERDY 位被拉高，HSE 时钟才会释放;
- 3) 配置时钟控制寄存器 (RCC\_PLLCFG) 中的 PLLSRC 位置”1”，选择 HSE 时钟用作 PLL 输入时钟;
- 4) 配置时钟控制寄存器 (RCC\_PLLCFG) 中的 PLLXTPRE 位，选择 HSE 或者 HSE2 分频为 PLL 输入时钟;
- 5) 配置时钟控制寄存器 (RCC\_PLLCFG) 中 PLLMUL (倍频系数)、PLLDIV (分频系数) ;
- 6) 配置时钟控制寄存器(RCC\_CR)中的 PLLON 位置”1”，使能 PLL;
- 7) 等待时钟控制寄存器(RCC\_CR) 中的 PLLRDY 位被拉高，表示 PLL 准备就绪，只有当 PLLRDY 位被拉高，PLL 时钟才会释放。
- 8) 通过配置时钟配置寄存器 (RCC\_CFGR)的 SW 位为 “b10” 来选择 PLL 为系统时钟;
- 9) 通过读取时钟配置寄存器 (RCC\_CFGR)的 SWS 位，判断当前系统时钟的时钟源是否为 PLL。
- 10) 配置控制状态寄存器 (RCC\_CSR)的 LSION 位置”1”和 LSI\_OEN\_LV 位置”1”，使能 LSI。
- 11) 等待控制状态寄存器 (RCC\_CSR) 中的 LSIRDY 位被拉高，表示 LSI 准备就绪，只有当 LSIRDY 位被拉高，LSI 时钟才会释放。

12) 配置时钟控制寄存器(RCC\_CR)中的 CSSON 位置 1 , 时钟安全系统使能。

### 5.2.4.9 时钟输出 (MCO)

微控制器时钟输出 (MCO) 能力允许时钟输出到外部 MCO 引脚上。相应 GPIO 端口的配置寄存器必须被配置为相应功能。可以选择以下 5 个时钟信号中的一个作为 MCO 时钟:

表 5-1MCO 与时钟源对应关系

时钟配置寄存器 (RCC_CFGR) 中的 MCO[2:0]位	时钟源
00x	没有时钟输出
010	LSI
011	没有时钟输出
100	SYSCLK
101	HSI
110	HSE
111	PLL/DIV2

### 5.2.4.10 RTC 时钟

RTCCLK 时钟源可以是 HSE/128、LSE 或 LSI 时钟，通过配置备份域控制寄存器 (RCC\_BDCR) 中 RTCSEL[1: 0]位来选择。如果不重置备份域，则无法修改此选择，必须设置 DBP 位（取消后备区域的写保护）为'1'。

LSE 时钟在备份域里，而 HSE 和 LSI 时钟不在备份域。因此：

- 1) 如果 LSE 被选为 RTC 时钟：只要 VBAT 维持供电，即使 VDD 供电被切断，RTC 仍继续工作。
- 2) 如果 LSI 被选为自动唤醒单元 (AWU) 时钟：如果 VDD 电源关闭，AWU 状态不能被保证。
- 3) 如果 HSE 时钟 128 分频后作为 RTC 时钟：如果 VDD 电源关闭或内部电压调压器被关闭 (1.5V 域的供电被切断)，则 RTC 状态不确定。

### 5.2.4.11 定时器时钟

当时钟配置寄存器 (RCC\_CFGR) 的 APB1 预分频系数<4 (不分频) 时，挂在 APB1 上的定时器的时钟频率和 APB1 总线频率一样；否则挂在 APB1 上的定时器的时钟频率是 APB1 总线频率的两倍；挂在 APB2 上的定时器同理。

当时钟配置寄存器 2 (RCC\_CFGR2) 的 TIMADV\_CKSEL 置 1，高级定时器的时钟频率直接由系统时钟经过时钟配置寄存器 2 (RCC\_CFGR2) 的 TIMADV\_CLK 预分频得到。

### 5.2.4.12 独立看门狗时钟

当独立看门狗已经由硬件启动，LSI 振荡器将被自动打开，并且不能被关闭；

当独立看门狗由软件启动，则 LSI 振荡器需通过软件使能打开，在 LSI 振荡器稳定后，时钟供应给 IWDG。

## 5.3 寄存器描述

表 5-2 RCC 寄存器概览

Offset	Acronym	RegisterName	Reset
0x00	RCC_CR	时钟控制寄存器	0x00000001
0x04	RCC_CFGR	时钟配置寄存器	0x00000000
0x08	RCC_CIR	时钟中断寄存器	0x00000000
0x0C	RCC_AHB3RSTR	AHB3 外设复位寄存器	0x00000000
0x10	RCC_AHB2RSTR	AHB2 外设复位寄存器	0x00000000
0x14	RCC_AHB1RSTR	AHB1 外设复位寄存器	0x00000000
0x18	RCC_APB2RSTR	APB2 外设复位寄存器	0x00000000
0x1C	RCC_APB1RSTR	APB1 外设复位寄存器	0x00000000
0x20	RCC_AHB3ENR	AHB3 外设时钟使能寄存器	0x00000000
0x24	RCC_AHB2ENR	AHB2 外设时钟使能寄存器	0x00000000
0x28	RCC_AHB1ENR	AHB1 外设时钟使能寄存器	0x00006000
0x2C	RCC_APB2ENR	APB2 外设时钟使能寄存器	0x00000000
0x30	RCC_APB1ENR	APB1 外设时钟使能寄存器	0x00000000
0x34	RCC_BDCR	备份域控制寄存器	0x00000000
0x38	RCC_CSR	控制状态寄存器	0xC0000000
0x3C	RCC_SYSCFG	系统配置寄存器	0x0C000101
0x40	RCC_CFGR2	时钟配置寄存器 2	0x00031F00
0x44	RCC_ICSCR	内部时钟源校准寄存器	0x20000000
0x48	RCC_PLLCFG	PLL 配置寄存器	0x0018031C
0x80	RCC_HSIDLY	HSI 延迟寄存器	0x00000003
0x84	RCC_HSELDLY	HSE 延迟寄存器	0x000003E8
0x88	RCC_PLLDLY	PLL 延迟寄存器	0x000003E8

### 5.3.1 时钟控制寄存器 (RCC\_CR)

偏移地址： 0x00

复位值: 0x00000001

访问: 无等待状态, 字, 半字和字节访问

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
Field	Res.							PL LR DY	PL LO N	Res.					CS SO N	HS EB YP	HS ER DY	HSE ON
Type								r	rw						rw	rw	r	rw
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Field	Res.		HSIDIV			Res.							HSI RD Y	HSI ON				
Type			rw										r	rw				

Bit	Field	Type	Reset	Description
31:26	Res.			始终读为 0。
25	PLLRDY	r	0x0	PLLRDY: PLL 时钟准备就绪标志 (PLLclockreadyflag) 由硬件设置。 0: PLL 未稳定 1: PLL 已稳定
24	PLLON	rw	0x0	PLLON: PLL 使能 (PLL enable) 通过软件置'1'或清除。 当进入停止或待机模式时, 由硬件清除。如果 PLL 时钟用作系统时钟或被选择将成为系统时钟, 则无法重置此位。 0: 禁用 PLL 1: 使能 PLL
23:20	Reserved			始终读为 0。
19	CSSON	rw	0x0	CSSON: 时钟安全系统使能 (Clock security system enable) 通过软件置'1'或清除。 0: 禁用时钟探测器 1: 如果高速外部振荡器准备就绪, 使能时钟探测器

Bit	Field	Type	Reset	Description
18	HSEBYP	rw	0x0	<p>HSEBYP: 外部高速时钟旁路 (External high-speed clock bypass)</p> <p>在调试模式下由软件置'1'来旁路外部晶体振荡器。</p> <p>只有当 HSE 振荡器被禁用时，才能写入 HSEBYP 位。</p> <p>0: 禁用高速外部晶体振荡器旁路模式</p> <p>1: 使能高速外部晶体振荡器旁路模式</p>
17	HSERDY	r	0x0	<p>HSERDY: 外部高速时钟准备就绪标志(External high-speed clock ready flag)</p> <p>由硬件设置。</p> <p>0: 高速外部晶体振荡器未稳定</p> <p>1: 高速外部晶体振荡器已稳定</p>
16	HSEON	rw	0x0	<p>HSEON: 外部高速时钟开启 (External high-speed clock enable)</p> <p>通过软件置'1'或清除。</p> <p>当进入待机和停机模式时，此位由硬件清零，停止外部时钟。如果外部时钟直接或间接或被选择将要作为系统时钟时，该位不能被清零。</p> <p>0: 禁用高速外部晶体振荡器</p> <p>1: 使能高速外部晶体振荡器</p>
15: 14	Reserved			始终读为 0。

Bit	Field	Type	Reset	Description
13:11	HSIDIV	rw	0x0	<p>HSIDIV: 内部高速时钟分频系数            (Internal high - speed clock division factor)</p> <p>000:HSI 不分频            001:HSI2 分频            010:HSI4 分频            011:HSI8 分频            100:HSI16 分频            101:HSI32 分频            110:HSI64 分频            111:HSI128 分频</p> <p>注: stop 或 deepstop 模式被唤醒后, 系统将强制使用 HSI 作为系统时钟。在 lpstop 或 lpdeepstop 模式下, 需要配置内部高速时钟分频系数以降低系统时钟频率。</p>
10: 2	Reserved			始终读为 0。
1	HSIRDY	r	0x0	<p>HSIRDY: 内部高速时钟就绪标志 (Internal high-speed clock ready flag)</p> <p>由硬件置'1', 表示内部时钟已经稳定。</p> <p>在 HSION 位被清除后, HSIRDY 在 3 个 AHB 时钟周期后变低。</p> <p>0: 内部高速时钟未稳定            1: 内部高速时钟已稳定</p>
0	HSION	rw	0x1	<p>HSION: 内部高速时钟使能 (Internal high-speed clock enable)</p> <p>通过软件置'1'或清除。</p> <p>当离开待机或停机模式或外部振荡器用作系统时钟时发生故障, 此位由硬件置'1', 来迫使内部振荡器打开。当内部高速时钟被直接或间接地用作或被选择将要作为系统时钟时, 该位不能被清零。</p> <p>0: 禁用内部高速时钟            1: 使能内部高速时钟</p>

### 5.3.2 时钟配置寄存器 (RCC\_CFGR)

偏移地址: 0x04

复位值: 0x00000000

访问: 无等待状态, 字, 半字和字节访问

只有当访问发生在时钟切换时, 才会插入 1 或 2 个等待周期。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved					MCO			USBPRE		Reserved					
Type						rw	rw	rw	rw	rw						
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.		PPRE2			PPRE1			HPRE			SWS		SW		
Type			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	r	r	rw	rw

Bit	Field	Type	Reset	Description
31:27	Reserved			始终读为 0。
26:24	MCO	rw	0x0	<p>MCO: 微控制器时钟输出(Micro controller clock output)由软件置 '1' 或清零。</p> <p>00x: 没有时钟输出;</p> <p>010: LSI 时钟输出;</p> <p>011: LSE 时钟输出;</p> <p>100: 系统时钟(SYSCLK)输出;</p> <p>101: HSI 时钟输出;</p> <p>110: HSE 时钟输出;</p> <p>111: PLL 时钟 2 分频后输出。</p> <p>注意: 1.该时钟输出在启动和切换 MCO 时钟源时可能会被停止。2.系统时钟作为输出至 MCO 管脚时, 请保证输出时钟频率不超过 50MHz(IO 口最高频率)</p>

Bit	Field	Type	Reset	Description
23:22	USBPRE	rw	0x0	<p>USBPRE: USB 预分频 (USB prescaler)</p> <p>通过软件指定分频值以输出 48MHz 的 USB 时钟。</p> <p>在使能 RCC_APB1ENR 寄存器中 USB 时钟之前，此为必须有效</p> <p>00: PLL 时钟直接作为 USB 时钟</p> <p>01: PLL 时钟 2 分频作为 USB 时钟</p> <p>10: PLL 时钟 3 分频作为 USB 时钟</p> <p>11: PLL 时钟 4 分频作为 USB 时钟</p>
21:14	Reserved			始终读为 0。
13:11	PPRE2	rw	0x0	<p>PPRE2: APB2 预分频系数</p> <p>通过软件置'1'或清除来控制高速 APB2 时钟 (PCLK2) 预分频系数。</p> <p>0xx: HCLK 不分频</p> <p>100: HCLK2 分频</p> <p>101: HCLK4 分频</p> <p>110: HCLK8 分频</p> <p>111: HCLK16 分频</p>
10:8	PPRE1	rw	0x0	<p>PPRE1: APB1 预分频系数</p> <p>通过软件置'1'或清除来控制低速 APB1 时钟 (PCLK1) 预分频系数。</p> <p>0xx: HCLK 不分频</p> <p>100: HCLK2 分频</p> <p>101: HCLK4 分频</p> <p>110: HCLK8 分频</p> <p>111: HCLK16 分频</p>

Bit	Field	Type	Reset	Description
7:4	HPRE	rw	0x0	<p>HPRE: AHB 预分频系数</p> <p>通过软件置'1'或清除来控制 AHB 时钟的预分频系数。</p> <p>0xxx: SYSCLK 不分频</p> <p>1000: SYSCLK2 分频</p> <p>1001: SYSCLK4 分频</p> <p>1010: SYSCLK8 分频</p> <p>1011: SYSCLK16 分频</p> <p>1100: SYSCLK64 分频</p> <p>1101: SYSCLK128 分频</p> <p>1110: SYSCLK256 分频</p> <p>1111: SYSCLK512 分频</p> <p>注:</p> <p>当 AHB 时钟的预分频系数大于 1 时, 必须开启预取缓冲器。 详见读闪存存储器一节。</p>
3:2	SWS	r	0x0	<p>SWS: 系统时钟选择状态 (System clock switch status)</p> <p>00: 选择 HSI 用作系统时钟;</p> <p>01: 选择 HSE 用作系统时钟;</p> <p>10: 选择 PLL 输出用作系统时钟;</p> <p>11: 选择 LSI 输出用作系统时钟。</p>
1:0	SW	rw	0x0	<p>SW: 系统时钟选择 (System clock switch)</p> <p>通过软件置'1'或清除来选择系统时钟源。</p> <p>当从停止或待机模式中返回时或直接或间接作为系统时钟的 HSE 出现故障时, 硬件会强制选择 HSI 作为系统时钟.</p> <p>00: 选择 HSI 用作系统时钟;</p> <p>01: 选择 HSE 用作系统时钟;</p> <p>10: 选择 PLL 输出用作系统时钟;</p> <p>11: 选择 LSI 输出用作系统时钟。</p>

### 5.3.3 时钟中断寄存器 (RCC\_CIR)

偏移地址: 0x08

复位值: 0x00000000

访问: 无等待周期, 字, 半字和字节访问

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved								CS SC	Reserved		PL LR DY C	HS ER DY C	HSI RD YC	LS ER DY C	LSIR DYC
Type									rc_ w1			rc_ w1	rc_ w1	rc_ w1	rc_w	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved			PLL RD YIE	HSER RDY DYIE	HSI ER DYI E	LS RD YIE	LSI RD YIE	CS SF	Reserved		PL LR DY F	HS ER DY F	HSI RD YF	LSE RD YF	LSI RD YF
Type				rw	rw	rw	rw	rw	r			r	r	r	r	r

Bit	Field	Type	Reset	Description
31:24	Reserved			始终读为 0。
23	CSSC	rc_w1	0x0	<p>CSSC: 时钟安全系统中断清除 (Clock security system interrupt clear)</p> <p>通过软件置'1'来清除安全系统中断标志位 CSSF。</p> <p>0: 无效</p> <p>1: 清除安全系统中断标志位 CSSF</p>
22:21	Reserved			始终读为 0。
20	PLLRDYC	rc_w1	0x0	<p>PLLRDYC: 清除 PLL 就绪中断 (PLL ready interrupt clear)</p> <p>通过软件置'1'来清除 PLL 就绪中断标志位 PLLRDYF。</p> <p>0: 无效</p> <p>1: 清除 PLL 就绪中断标志位 PLLRDYF</p>
19	HSERDYC	rc_w1	0x0	<p>HSERDYC: 清除 HSE 就绪中断 (HSE ready interrupt clear)</p> <p>通过软件置'1'来清除 HSE 就绪中断标志位 HSERDYF。</p> <p>0: 无效</p> <p>1: 清除 HSE 就绪中断标志位 HSERDYF</p>

Bit	Field	Type	Reset	Description
18	HSIRDYC	rc_w1	0x0	<p>HSIRDYC: 清除 HSI 就绪中断 (HSIreadyinterruptclear) 通过软件置'1'来清除 HSI 就绪中断标志位 HSIRDYF。 0: 无效 1: 清除 HSI 就绪中断标志位 HSIRDYF</p>
17	LSERDYC	rc_w1	0x0	<p>LSERDYC: 清除 LSE 就绪中断 (LSE ready interrupt clear) 通过软件置'1'来清除 LSE 就绪中断标志位 LSERDYF。0: 无效 0: 无效 1: 清除 LSE 就绪中断标志位 LSERDYF</p>
16	LSIRDYC	rc_w1	0x0	<p>LSIRDYC: 清除 LSI 就绪中断 (LSI ready interrupt clear) 通过软件置'1'来清除 LSI 就绪中断标志位 LSIRDYF。0: 无 效 0: 无效 1: 清除 LSI 就绪中断标志位 LSIRDYF</p>
15:13	Reserved			始终读为 0。
12	PLL RDYIE	rw	0x0	<p>PLL RDYIE: PLL 就绪中断使能 (PLL ready interrupt enable) 通过软件置'1'来清除来使能或禁用 PLL 就绪中断。 0: 禁用 PLL 就绪中断 1: 使能 PLL 就绪中断</p>
11	HSE RDYIE	rw	0x0	<p>HSE RDYIE: HSE 就绪中断使能 (HSE ready interrupt enable) 通过软件置'1'来清除来使能或禁用外部振荡器就绪中 断。 0: 禁用 HSE 就绪中断 1: 使能 HSE 就绪中断</p>
10	HSI RDYIE	rw	0x0	<p>HSI RDYIE: HSI 就绪中断使能 (HSI ready interrupt enable) 通过软件置'1'来清除来使能或禁用内部振荡器就绪中 断。 0: 禁用 HSI 就绪中断 1: 使能 HSI 就绪中断</p>

Bit	Field	Type	Reset	Description
9	LSERDYIE	rw	0x0	<p>LSERDYIE : LSE 就绪中断使能 (LSE ready interrupt enable)</p> <p>通过软件置'1'来清除来使能或禁用外部 32KHz 振荡器就绪中断。</p> <p>0: 禁用 LSE 就绪中断 1: 使能 LSE 就绪中断</p>
8	LSIRDYIE	rw	0x0	<p>LSIRDYIE: LSI 就绪中断使能 (LSI ready interrupt enable)</p> <p>通过软件置'1'来清除来使能或禁用内部 40KHz 振荡器就绪中断。</p> <p>0: 禁用 LSI 就绪中断 1: 使能 LSI 就绪中断</p>
7	CSSF	r	0x0	<p>CSSF: 时钟安全系统中断标志 (Clock security system interrupt flag)</p> <p>在外部振荡器时钟出现故障时, 由硬件置'1'。</p> <p>通过软件将 CSSC 位置'1'来清除。</p> <p>0: 无 HSE 时钟失效产生的安全系统中断 1: HSE 时钟失效导致了时钟安全系统中断</p>
6:5	Reserved			始终读为 0。
4	PLLRDYF	r	0x0	<p>PLLRDYF: PLL 就绪中断标志 (PLL ready interrupt flag)</p> <p>在 PLL 就绪时, 由硬件置'1'。</p> <p>通过软件将 PLLRDYC 位置'1'来清除。</p> <p>0: 无 PLL 上锁产生的时钟就绪中断 1: PLL 上锁导致时钟就绪中断</p>
3	HSERDYF	r	0x0	<p>HSERDYF: HSE 就绪中断标志 (HSE ready interrupt flag)</p> <p>在外部低速时钟就绪时, 由硬件置'1'。</p> <p>通过软件将 HSERDYC 位置'1'来清除。</p> <p>0: 无外部振荡器产生的时钟就绪中断 1: 外部振荡器导致时钟就绪中断</p>

Bit	Field	Type	Reset	Description
2	HSIRDYF	r	0x0	<p>HSIRDYF: HSI 就绪中断标志 (HSI ready interrupt flag) 在内部高速时钟就绪时，由硬件置'1'。 通过软件将 HSIRDYC 位置'1'来清除。</p> <p>0: 无内部 HSI 振荡器产生的时钟就绪中断 1: 内部 HSI 振荡器导致时钟就绪中断</p>
1	LSERDYF	r	0x0	<p>LSERDYF: LSE 就绪中断标志 (LSEreadyinterruptflag) 在外部低速时钟就绪时，由硬件置'1'。 通过软件将 LSERDYC 位置'1'来清除。</p> <p>0: 无外部 32KHz 振荡器产生的时钟就绪中断; 1: 外部 32KHz 振荡器导致时钟就绪中断。</p>
0	LSIRDYF	r	0x0	<p>LSIRDYF: LSI 就绪中断标志 (LSI ready interrupt flag) 在内部低速时钟就绪时，由硬件置'1'。 通过软件将 LSIRDYC 位置'1'来清除。</p> <p>0: 无内部 40KHz 振荡器产生的时钟就绪中断; 1: 内部 40KHz 振荡器导致时钟就绪中断。</p>

### 5.3.4 AHB3 外设复位寄存器 (RCC\_AHB3RSTR)

偏移地址: 0x0C

复位值: 0x00000000

访问: 无等待周期, 字, 半字和字节访问

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

Bit	Field	Type	Reset	Description
31:1	Reserved			始终读为 0。

Bit	Field	Type	Reset	Description
0	FSMC	rw	0x0	FSMC: FSMC 复位(FSMC reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 FSMC

### 5.3.5 AHB2 外设复位寄存器 (RCC\_AHB2RSTR)

偏移地址: 0x10

复位值: 0x00000000

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

Bit	Field	Type	Reset	Description
31:8	Reserved			始终读为 0。
7	USBOTGFS	rw	0x0	USBOTGFS: USBOTGFS 复位(USBOTGFSreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 USBOTGFS
6:0	Reserved			始终读为 0。

### 5.3.6 AHB1 外设复位寄存器 (RCC\_AHB1RSTR)

偏移地址: 0x14

复位值: 0x00000000

访问: 无等待周期, 字, 半字和字节访问

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.							ET	Reserved		DM	DM	Res.			
								HM			A2	A1				

Type							rw				rw	rw						
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
	Res.			CR C	Res.	SDI O	Res.		GPI OH	GPI OG	GPI OF	GPI OE	GPI OD	GPI OC	GPI OB	GPI OA		
Type				rw		rw			rw									

Bit	Field	Type	Reset	Description
31:26	Reserved			始终读为 0。
25	ETHMAC	rw	0x00	ETHMAC: ETHMAC 复位(EthernetMACreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 ETHMAC
24:23	Reserved			始终读为 0。
22	DMA2	rw	0x00	DMA2: DMA2 复位(DMA2reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 DMA2
21	DMA1	rw	0x00	DMA1: DMA1 复位(DMA1reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 DMA1
20:13	Reserved			始终读为 0。
12	CRC	rw	0x00	CRC: CRC 复位(CRCreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 CRC
11	Reserved			始终读为 0。
10	SDIO	rw	0x00	SDIO: SDIO 复位(SDIOreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 SDIO
9:8	Reserved			始终读为 0。
7	GPIOH	rw	0x0	GPIOH: GPIOH 复位(GPIOHreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 GPIOH

Bit	Field	Type	Reset	Description
6	GPIOG	rw	0x00	GPIOG: GPIOG 复位(GPIOGreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 GPIOG
5	GPIOF	rw	0x00	GPIOF: GPIOF 复位(GPIOFreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 GPIOF
4	GPIOE	rw	0x00	GPIOE: GPIOE 复位(GPIOEreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 GPIOE
3	GPIOD	rw	0x00	GPIOD: GPIOD 复位(GPIODreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 GPIOD
2	GPIOC	rw	0x00	GPIOC: GPIOC 复位(GPIOCreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 GPIOC
1	GPIOB	rw	0x00	GPIOB: GPIOB 复位(GPIOBreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 GPIOB
0	GPIOA	rw	0x00	GPIOA: GPIOA 复位(GPIOAreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 GPIOA

### 5.3.7 APB2 外设复位寄存器 (RCC\_APB2RSTR)

偏移地址: 0x18

复位值: 0x00000000

访问: 无等待周期, 字, 半字和字节访问

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

Field	COMP	SYS CFG	Reserved	SPI 1	Reserved	ADC 3	ADC 2	AD C1	Reserved	UA RT 6	UA RT 1	Reserved	TIM 8	TIM 1
Type	rw	rw		rw		rw	rw	rw					rw	

Bit	Field	Type	Reset	Description
31:16	Reserved			始终读为 0。
15	COMP	rw	0x0	COMP: 比较器复位 (Comparator reset) 通过软件置'1'或清除 0: 无效 1: 重置比较器接口
14	SYSCFG	rw	0x0	SYSCFG: SYSCFG 复位 (SYSCFG reset) 通过软件置'1'或清除 0: 无效 1: 重置 SYSCFG
13	Reserved			始终读为 0。
12	SPI1	rw	0x0	SPI1: SPI1 复位 (SPI1 reset) 通过软件置'1'或清除 0: 无效 1: 重置 SPI1
11	Reserved			始终读为 0。
10	ADC3	rw	0x0	ADC3: ADC3 复位 (ADC3reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 ADC3
9	ADC2	rw	0x0	ADC2: ADC2 复位 (ADC2reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 ADC2

Bit	Field	Type	Reset	Description
8	ADC1	rw	0x0	ADC1: ADC1 复位 (ADC1 interface reset) 通过软件置'1'或清除 0: 无效 1: 重置 ADC1
7:6	Reserved			始终读为 0。
5	UART6	rw	0x0	UART6: UART6 复位(UART6reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 UART6
4	UART1	rw	0x0	UART1: UART1 复位(UART1reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 UART1
3:2	Reserved			始终读为 0。
1	TIM8	rw	0x0	TIM8: TIM8 定时器复位(TIM8timerreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 TIM8 定时器
0	TIM1	rw	0x0	TIM1: TIM1 定时器复位(TIM1timerreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 TIM1 定时器

### 5.3.8 APB1 外设复位寄存器 (RCC\_APB1RSTR)

偏移地址: 0x1C

复位值: 0x00000000

访问: 无等待周期, 字, 半字和字节访问

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	UA RT8	UAR T7	DA C	PW R	BKP	Res.	CA N	CR S	Re s.	I2C 2	I2C 1	UA RT 5	UA RT 4	UA RT 3	UA RT 2	Res.
Type	rw	rw	rw	rw		rw	rw		rw	rw	rw	rw	rw	rw		
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Field	SPI3	SPI2	Reserved	WWDG	Reserved	TIM7	TIM6	TI M5	TI M4	TI 3	TI 2
Type		rw		rw		rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31	UART8	rw	0x0	UART8: UART8 复位(UART8reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 UART8
30	UART7	rw	0x0	UART7: UART7 复位(UART7reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 UART7
29	DAC	rw	0x0	DAC: DAC 复位(DACreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 DAC
28	PWR	rw	0x0	PWR: 电源复位 (Power interface reset) 通过软件置'1'或清除 0: 无效 1: 重置电源
27	BKP	rw	0x0	BKP: 备份接口复位(Backupinterfacereset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位备份接口
26	Reserved			始终读为 0。
25	CAN	rw	0x0	CAN: CAN 复位 (CAN reset) 通过软件置'1'或清除 0: 无效 1: 重置 CAN
24	CRS	rw	0x0	CRS: CRS 复位(CRSreset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 CRS

Bit	Field	Type	Reset	Description
23	Reserved			始终读为 0。
22	I2C2	rw	0x0	I2C2: I2C2 复位(I2C2reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 I2C2
21	I2C1	rw	0x0	I2C1: I2C1 复位 (I2C1 reset) 通过软件置'1'或清除 0: 无效 1: 重置 I2C1
20	UART5	rw	0x0	UART5: UART5 复位(UART5reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 UART5
19	UART4	rw	0x0	UART4: UART4 复位(UART4reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 UART4
18	UART3	rw	0x0	UART3: UART3 复位(UART3reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 UART3
17	UART2	rw	0x0	UART2: UART2 复位 (UART2 reset) 通过软件置'1'或清除 0: 无效 1: 重置 UART2
16	Reserved			始终读为 0。
15	SPI3	rw	0x0	SPI3: SPI3 复位(SPI3reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 SPI3
14	SPI2	rw	0x0	SPI2: SPI2 复位 (SPI2 reset) 通过软件置'1'或清除 0: 无效 1: 重置 SPI2

Bit	Field	Type	Reset	Description
13:12	Reserved			始终读为 0。
11	WWDG	rw	0x0	WWDG: 窗口看门狗复位 (Window watchdog reset) 通过软件置'1'或清除 0: 无效 1: 重置窗口看门狗
10:6	Reserved			始终读为 0。
5	TIM7	rw	0x0	TIM7: 定时器 7 复位(Timer7reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 TIM7 定时器
4	TIM6	rw	0x0	TIM6: 定时器 6 复位(Timer6reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 TIM6 定时器
3	TIM5	rw	0x0	TIM5: 定时器 5 复位(Timer5reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 TIM5 定时器
2	TIM4	rw	0x0	TIM4: 定时器 4 复位(Timer4reset)由软件置 '1' 或清 '0'。 0: 无效 1: 复位 TIM4 定时器
1	TIM3	rw	0x0	TIM3: 定时器 3 复位 (Timer3 reset) 通过软件置'1'或清除 0: 无效 1: 重置 TIM3 定时器
0	TIM2	rw	0x0	TIM2: 定时器 2 复位 (Timer2 reset) 通过软件置'1'或清除 0: 无效 1: 重置 TIM2 定时器

### 5.3.9 AHB3 外设时钟使能寄存器 (**RCC\_AHB3ENR**)

偏移地址: 0x20

复位值: 0x00000000

访问：无等待周期，字，半字和字节访问

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

Bit	Field	Type	Reset	Description
31:1	Reserved			始终读为 0。
0	FSMC	rw	0x0	FSMC: FSMC 时钟使能(FSMCclockenable)由软件置 '1' 或清 '0'。 0: FSMC 时钟关闭 1: FSMC 时钟开启

### 5.3.10 AHB2 外设时钟使能寄存器 (RCC\_AHB2ENR)

偏移地址: 0x24

复位值: 0x00000000

访问：无等待周期，字，半字和字节访问

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

Bit	Field	Type	Reset	Description
31:1	Reserved			始终读为 0。

Bit	Field	Type	Reset	Description
7	USBOTGFS	rw	0x0	USBOTGFS： USBOTGFS(USBOTGFSclockenable)由软件置‘1’或清‘0’。 0: USBOTGFS 时钟关闭 1: USBOTGFS 时钟开启
6:0	Reserved			始终读为 0。

### 5.3.11 AHB1 外设时钟使能寄存器（RCC\_AHB1ENR）

偏移地址: 0x28

复位值: 0x00006000

访问: 无等待周期, 字, 半字和字节访问

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Res.						ET HM AC	Reserved		DM A2	DM A1	Res.					
Type							rw			rw	rw						
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	Res.	SR AM	Flas h	CR C	Res.	SDI O	Res.	GPI OH	GPI OG	GPI OF	GPI OE	GPI OD	GPI OC	GPI OB	GPI OA		
Type			rw		rw			rw									

Bit	Field	Type	Reset	Description
31:26	Reserved			始终读为 0。
25	ETHMAC	rw	0x00	ETHMAC： ETHMAC 时钟使能(EthernetMACclockenable) 由软件置‘1’或清‘0’。 0: ETHMAC 时钟关闭 1: ETHMAC 时钟开启
24:23	Reserved			始终读为 0。

Bit	Field	Type	Reset	Description
22	DMA2	rw	0x00	DMA2: DMA2 时钟使能(DMA2clockenable)由软件置 '1' 或清 '0'。 0: DMA2 时钟关闭 1: DMA2 时钟开启
21	DMA1	rw	0x00	DMA1: DMA1 时钟使能(DMA1clockenable)由软件置 '1' 或清 '0'。 0: DMA1 时钟关闭 1: DMA1 时钟开启
20:15	Reserved			始终读为 0。
14	SRAM	rw	0x01	SRAM: SRAM 时钟使能(SRAMclockenable)由软件置 '1' 或清 '0'。 0: SRAM 时钟关闭 1: SRAM 时钟开启
13	FLASH	rw	0x01	FLASH:FLASH 时钟使能(FLASHclockenable)由软件置 '1' 或清 '0'。 0: FLASH 时钟关闭 1: FLASH 时钟开启
12	CRC	rw	0x00	FLASH:FLASH 时钟使能(FLASHclockenable)由软件置 '1' 或清 '0'。 0: FLASH 时钟关闭 1: FLASH 时钟开启
11	Reserved			始终读为 0。
10	SDIO	rw	0x00	SDIO: SDIO 时钟使能(SDIOclockenable)由软件置 '1' 或清 '0'。 0: SDIO 时钟关闭 1: SDIO 时钟开启
9:8	Reserved			始终读为 0。
7	GPIOH	rw	0x0	GPIOH:GPIOH 时钟使能(GPIOHclockenable)由软件置 '1' 或清 '0'。 0: GPIOH 时钟关闭 1: GPIOH 时钟开启

Bit	Field	Type	Reset	Description
6	GPIOG	rw	0x00	GPIOG: GPIOG 时钟使能(GPIOGclockenable)由软件置 '1' 或清 '0'。 0: GPIOG 时钟关闭 1: GPIOG 时钟开启
5	GPIOF	rw	0x00	GPIOF: GPIOF 时钟使能(GPIOFclockenable)由软件置 '1' 或清 '0'。 0: GPIOF 时钟关闭 1: GPIOF 时钟开启
4	GPIOE	rw	0x00	GPIOE: GPIOE 时钟使能(GPIOEclockenable)由软件置 '1' 或清 '0'。 0: GPIOE 时钟关闭 1: GPIOE 时钟开启
3	GPIOD	rw	0x00	GPIOD: GPIOD 时钟使能(GPIODclockenable)由软件置 '1' 或清 '0'。 0: GPIOD 时钟关闭 1: GPIOD 时钟开启
2	GPIOC	rw	0x00	GPIOC: GPIOC 时钟使能(GPIOCclockenable)由软件置 '1' 或清 '0'。 0: GPIOC 时钟关闭 1: GPIOC 时钟开启
1	GPIOB	rw	0x00	GPIOB: GPIOB 时钟使能(GPIOBclockenable)由软件置 '1' 或清 '0'。 0: GPIOB 时钟关闭 1: GPIOB 时钟开启
0	GPIOA	rw	0x00	GPIOA: GPIOA 时钟使能(GPIOAclockenable)由软件置 '1' 或清 '0'。 0: GPIOA 时钟关闭 1: GPIOA 时钟开启

### 5.3.12 APB2 外设时钟使能寄存器 (RCC\_APB2ENR)

偏移地址: 0x2C

复位值: 0x00000000

访问: 无等待周期, 字, 半字和字节访问

注：当外设时钟没有启动时，软件不能读出外设寄存器的数值

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CO MP	SYS CFG	Res erve d	SPI 1	Rese rved	ADC 3	ADC 2	AD C1	Reserved		UA RT 6	UA RT 1	Reserved		TIM 8	TIM 1

Bit	Field	Type	Reset	Description
31:16	Reserved			始终读为 0。
15	COMP	rw	0x0	COMP: COMP 时钟使能(COMPclockenable)由软件置 '1' 或清 '0'。 0: COMP 时钟关闭 1: COMP 时钟开启
14	SYSCFG	rw	0x0	SYSCFG: SYSCFG 时钟使能由软件置 '1' 或清 '0'。 0: SYSCFG 时钟关闭 1: SYSCFG 时钟开启
13	Reserved			始终读为 0。
12	SPI1	rw	0x0	SPI1: SPI1 时钟使能(SPI1clockenable)由软件置 '1' 或清 '0'。 0: SPI1 时钟关闭 1: SPI1 时钟开启
11	Reserved			始终读为 0。
10	ADC3	rw	0x0	ADC3: ADC3 接口时钟使能(ADC3interfaceclockenable)由软件置 '1' 或清 '0'。 0: ADC3 接口时钟关闭 1: ADC3 接口时钟开启
9	ADC2	rw	0x0	ADC2: ADC2 接口时钟使能(ADC2interfaceclockenable)由软件置 '1' 或清 '0'。 0: ADC2 接口时钟关闭 1: ADC2 接口时钟开启

Bit	Field	Type	Reset	Description
8	ADC1	rw	0x0	ADC1: ADC1 接口时钟使能(ADC1interfaceclockenable)由软件置 '1' 或清 '0'。 0: ADC1 接口时钟关闭 1: ADC1 接口时钟开启
7:6	Reserved			始终读为 0。
5	UART6	rw	0x0	UART6: UART6 时钟使能(UART6clockenable)由软件置 '1' 或清 '0'。 0: UART6 时钟关闭 1: UART6 时钟开启
4	UART1	rw	0x0	UART1: UART1 时钟使能(UART1clockenable)由软件置 '1' 或清 '0'。 0: UART1 时钟关闭 1: UART1 时钟开启
3:2	Reserved			始终读为 0。
1	TIM8	rw	0x0	TIM8: 定时器 8 时钟使能(Timer8clockenable)由软件置 '1' 或清 '0'。 0: TIM8 时钟关闭 1: TIM8 时钟开启
0	TIM1	rw	0x0	TIM1: 定时器 1 时钟使能(Timer1clockenable)由软件置 '1' 或清 '0'。 0: TIM1 时钟关闭 1: TIM1 时钟开启

### 5.3.13 APB1 外设时钟使能寄存器 (RCC\_APB1ENR)

偏移地址: 0x30

复位值: 0x00000000

访问: 无等待周期, 字, 半字和字节访问

注: 当外设时钟没有启动时, 软件不能读出外设寄存器的数值

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	UA RT8	UAR T7	DA C	PW R	BKP	Res.	CA N	CR S	Re s.	I2C 2	I2C 1	UA RT	UA RT	UA RT	UA RT	Res.

												5	4	3	2	
Type	rw	rw	rw	rw	rw		rw	rw		rw	rw	rw	rw	rw	rw	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	SPI3	SPI2	Reserved	WWDG	Reserved					TIM7	TIM6	TI M5	TI M4	TI3	TIM2	
Type		rw		rw						rw	rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
31	UART8	rw	0x0	UART8: UART8 时钟使能(UART8clockenable)由软件置 '1' 或清 '0'。 0: UART8 时钟关闭 1: UART8 时钟开启
30	UART7	rw	0x0	UART7: 时钟使能(UART7clockenable)由软件置 '1' 或清 '0'。 0: UART7 时钟关闭 1: UART7 时钟开启
29	DAC	rw	0x0	DAC: DAC 时钟使能(Powerinterfaceclockenable)由软件置 '1' 或清 '0'。 0: DAC 时钟关闭 1: DAC 时钟开启
28	PWR	rw	0x0	PWR/DBG: 电源接口/DBG 时钟使能(Power interface clock enable)由软件置 '1' 或清 '0'。 0: 电源接口/DBG 时钟关闭 1: 电源接口/DBG 时钟开启
27	BKP	rw	0x0	BKP: 备份时钟使能(Backupinterfaceclockenable)由软件置 '1' 或清 '0'。 0: 备份接口时钟关闭 1: 备份接口时钟开启
26	Reserved			始终读为 0。
25	CAN	rw	0x0	CAN: CAN 时钟使能(CANclockenable)由软件置 '1' 或清 '0'。 0: CAN 时钟关闭 1: CAN 时钟开启

Bit	Field	Type	Reset	Description
24	CRS	rw	0x0	CRS: CRS 时钟使能(CRSclockenable)由软件置 '1' 或清 '0'。0: CRS 时钟关闭 1: CRS 时钟开启
23	Reserved			始终读为 0。
22	I2C2	rw	0x0	I2C2: I2C2 时钟使能(I2C2clockenable)由软件置 '1' 或清 '0'。0: I2C2 时钟关闭 1: I2C2 时钟开启
21	I2C1	rw	0x0	I2C1: I2C1 时钟使能(I2C1clockenable)由软件置 '1' 或清 '0'。0: I2C1 时钟关闭 1: I2C1 时钟开启
20	UART5	rw	0x0	UART5: UART5 时钟使能(UART5clockenable)由软件置 '1' 或清 '0'。 0: UART5 时钟关闭 1: UART5 时钟开启
19	UART4	rw	0x0	UART4: UART4 时钟使能(UART4clockenable))由软件置 '1' 或清 '0'。 0: UART4 时钟关闭 1: UART4 时钟开启
18	UART3	rw	0x0	UART3: UART3 时钟使能(UART3clockenable))由软件置 '1' 或清 '0'。 0: UART3 时钟关闭 1: UART3 时钟开启
17	UART2	rw	0x0	UART2: UART3 时钟使能(UART2clockenable))由软件置 '1' 或清 '0'。 0: UART2 时钟关闭 1: UART2 时钟开启
16	Reserved			始终读为 0。

Bit	Field	Type	Reset	Description
15	SPI3	rw	0x0	SPI3: SPI3 时钟使能(SPI3clockenable)由软件置 '1' 或清 '0'。 0: SPI3 时钟关闭 1: SPI3 时钟开启
14	SPI2	rw	0x0	SPI3: SPI3 时钟使能(SPI3clockenable)由软件置 '1' 或清 '0'。 0: SPI3 时钟关闭 1: SPI3 时钟开启
13:12	Reserved			始终读为 0。
11	WWDG	rw	0x0	WWDG : 窗口看门狗时钟使能(Windowwatchdogclockenable)由软件置 '1' 或清 '0'。 0: 窗口看门狗时钟关闭 1: 窗口看门狗时钟开启
10:6	Reserved			始终读为 0。
5	TIM7	rw	0x0	TIM7: 定时器 7 时钟使能(Timer7clockenable)由软件置 '1' 或清 '0'。 0: TIM7 定时器时钟关闭 1: TIM7 定时器时钟开启
4	TIM6	rw	0x0	TIM6: 定时器 16 时钟使能(Timer6clockenable)由软件置 '1' 或清 '0'。 0: TIM6 定时器时钟关闭 1: TIM6 定时器时钟开启
3	TIM5	rw	0x0	TIM5: 定时器 5 时钟使能(Timer5clockenable)由软件置 '1' 或清 '0'。 0: TIM5 定时器时钟关闭 1: TIM5 定时器时钟开启
2	TIM4	rw	0x0	TIM4: 定时器 4 时钟使能(Timer4clockenable)由软件置 '1' 或清 '0'。 0: TIM4 定时器时钟关闭 1: TIM4 定时器时钟开启

Bit	Field	Type	Reset	Description
1	TIM3	rw	0x0	<p>TIM3: 定时器 3 时钟使能(Timer3clockenable)由软件置 '1' 或清 '0'。</p> <p>0: TIM3 定时器时钟关闭 1: TIM3 定时器时钟开启</p>
0	TIM2	rw	0x0	<p>TIM2: 定时器 2 时钟使能(Timer2clockenable)由软件置 '1' 或清 '0'。</p> <p>0: TIM2 定时器时钟关闭 1: TIM2 定时器时钟开启</p>

### 5.3.14 备份域控制寄存器 (RCC\_BDCR)

偏移地址: 0x34

复位值: 0x00000000, 只能由备份域复位有效复位

支持字, 半字和字节方式访问, 0 到 3 个等待周期, 对该寄存器进行连续访问时需要插入等待状态

注: 备份域控制寄存器中 (RCC\_BDCR) 的 LSEON、LSEBYP、RTCSEL 和 RTCEN 位在备份域内。因此, 在重置之后, 这些位处于写保护的状态, 在修改这些位之前, 必须将 (RCC\_BDCR) DBP 位置 '1'。任何内部或外部重置都不会对这些位产生影响。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Reserved								DBP	Reserved							BDRST
Type								rw								rw	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	RTCEN	Reserved					RTCSEL		Reserved					LSYP	LERDY	LSEON	
Type	rw						rw							rw	r	rw	

Bit	Field	Type	Reset	Description
31:25	Reserved			始终读为 0。

Bit	Field	Type	Reset	Description
24	DBP	rw	0x0	<p>DBP：取消后被区域的写保护，在复位后，后备寄存器处于被保护状态以防意外写入。设置这一位允许写入这些寄存器。由软件置‘1’或清‘0’。</p> <p>0：禁止对 RTC 和后备寄存器的访问</p> <p>1：允许对 RTC 和后备寄存器的访问</p>
23:17	Reserved			始终读为 0。
16	BDRST	rw	0x0	<p>BDRST：备份域软件复位 (Backup domain software reset)</p> <p>通过软件置‘1’或清除</p> <p>0：复位未激活</p> <p>1：复位整个备份域</p>
15	RTCEN	rw	0x0	<p>RTCEN：RTC 时钟使能 (RTC clock enable)</p> <p>通过软件置‘1’或清除</p> <p>0：禁用 RTC 时钟</p> <p>1：使能 RTC 时钟</p>
14:11	Reserved			始终读为 0。
9:8	RTCSEL	rw	0x0	<p>RTCSEL：RTC 时钟源选择 (RTC clock source selection)</p> <p>由软件设置为选择 RTC 的时钟源。一旦选择了 RTC 时钟源，就不能再更改它，除非备份域被重置。可以使用 BDRST 位来重置它们。</p> <p>00：没有时钟</p> <p>01：LSE 振荡器用作 RTC 时钟</p> <p>10：LSI 振荡器用作 RTC 时钟</p> <p>11：HSE 振荡器在 128 分频后用作 RTC 时钟</p>
7:3	Reserved			始终读为 0。
2	LSEBYP	rw	0x0	<p>LSEBYP：外部低速振荡器旁路 (External low-speed oscillator bypass)</p> <p>在调试模式下通过软件置‘1’来旁路 LSE。只有在外部 32KHz 振荡器被禁用时，才能写入此位。</p> <p>0：禁用 LSE 振荡器旁路模式</p> <p>1：使能 LSE 振荡器旁路模式</p>

Bit	Field	Type	Reset	Description
1	LSERDY	rw	0x0	<p>LSERDY: 外部低速 LSE 就绪 (External low-speed oscillator ready)</p> <p>通过软件置'1'或清除来指示外部 32KHz 振荡器是否稳定。</p> <p>在 LSEON 被清除后，在 3 个 AHB 时钟周期后变低。</p> <p>0: 外部 32KHz 振荡器未稳定</p> <p>1: 外部 32KHz 振荡器已稳定</p>
0	LSEON	rw	0x0	<p>LSEON : 外部低速振荡器使能 (External low-speed oscillator enable)</p> <p>通过软件置'1'或清除</p> <p>0: 禁用外部 32KHz 振荡器</p> <p>1: 使能外部 32KHz 振荡器</p>

### 5.3.15 控制状态寄存器 (RCC\_CSR)

偏移地址: 0x38

复位值: 0x0C000000

访问: 0-3 等待周期, 字, 半字和字节访问

当连续对该寄存器进行访问时, 将插入等待状态。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	LPW WW RRS	DG IWD RST	GR SFT STF	SFT RST F	POR RST RSTF	PIN RST F	Re s.	RM VF	LO CK UP	PV DR ST	VD TR ST	Reserved					
Type	r	r	r	r	r	r		rw	r	r	r						
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Reserved								VDT RST NE N	LO CK UP EN	PV DR ST EN	LSI _O _LV	Reserved			LSI RD Y	LSI ON
Type									rw	rw	rw	rw				r	rw

Bit	Field	Type	Reset	Description
31	LPWRRSTF	r	0x0	<p>LPWRRSTF: 低功耗管理复位标志(Low power reset flag) 在低功耗管理复位发生时由硬件置‘1’，且只能由电源复位清除或由软件通过写 RMVF 位清除。</p> <p>0: 在低功耗管理复位发生时由硬件置‘1’。 1: 发生低功耗管理复位</p>
30	WWDGRSTF	r	0x0	<p>WDGRSTF: 窗口看门狗复位标志 (Window watchdog reset flag)</p> <p>在窗口看门狗复位发生时由硬件置‘1’，且只能由电源复位清除或由软件通过写 RMVF 位清除。</p> <p>0: 无窗口看门狗复位发生 1: 发生窗口看门狗复位</p>
29	IWDGRSTF	r	0x0	<p>IWDGRSTF: 独立看门狗复位标志 (Independent watchdog reset flag)</p> <p>在独立看门狗复位发生在 VDD 区域时由硬件置‘1’，且只能由电源复位清除或由软件通过写 RMVF 位清除。</p> <p>0: 无独立看门狗复位发生 1: 发生独立看门狗复位</p>
28	SFTRSTF	r	0x0	<p>SFTRSTF: 软件复位标志 (Software reset flag)</p> <p>在软件复位发生时由硬件置‘1’，且只能由电源复位清除或由软件通过写 RMVF 位清除。</p> <p>0: 无软件复位发生 1: 发生软件复位</p>
27	PORRSTF	r	0x1	<p>PORRSTF: 上电/掉电复位标志 (POR/PDR reset flag)</p> <p>在上电/掉电复位发生时由硬件置‘1’，且只能由电源复位清除或由软件通过写 RMVF 位清除。</p> <p>0: 无上电/掉电复位发生 1: 发生上电/掉电复位</p>
26	PINRSTF	r	0x1	<p>PINRSTF: nRST 管脚复位标志 (PIN reset flag)</p> <p>在 nRST 管脚复位发生时由硬件置‘1’，且只能由电源复位清除或由软件通过写 RMVF 位清除。</p> <p>0: 无 nRST 管脚复位发生 1: 发生 nRST 管脚复位</p>

Bit	Field	Type	Reset	Description
25	Reserved			始终读为 0。
24	RMVF	rw	0x0	<p>RMVF: 清除复位标志 (Remove reset flag) 由软件置'1'来清除复位标志。 0: 无效 1: 清除复位标志</p>
23	LOCKUPF	rw	0x0	<p>LOCKUPF: CPU 死锁复位标志 (CPU lockup reset flag) 在 CPU 发生死锁复位时由硬件置'1', 且只能由电源复位清除或由软件通过写 RMVF 位清除。 0: 无 CPU 死锁复位发生 1: 发生 CPU 死锁复位</p>
22	PVDRSTF	r	0x0	<p>PVDRSTF: PVD 复位标志 (PVD reset flag) 在 PVD 复位发生时由硬件置'1', 且只能由电源复位清除或由软件通过写 RMVF 位清除。 0: 无 PVD 复位发生 1: 发生 PVD 复位</p>
21	VDTRSTF	r	0x0	<p>VDTRSTF: 电源检测复位标志(Voltagedetectresetflag)在 VDT 复位发生时由硬件置 '1', 且只能由电源复位清除或由软件通过写 VDT 位清除。 0: 无 VDT 复位发生 1: 发生 VDT 复位</p>
20:9	Reserved			始终读为 0。
8	VDTRSTNEN	rw	0x0	<p>VDTRSTNEN : 电源检测复位使能 (Voltagedetectresetenable) 0: 禁止 VDT 产生复位 1: 使能 VDT 产生复位</p>
7	LOCKUPEN	rw	0x0	<p>LOCKUPEN: CPU 死锁复位使能 (CPU lockup reset enable) 0: 禁用 CPU 死锁复位 1: 使能 CPU 死锁复位</p>

Bit	Field	Type	Reset	Description
6	PVDRSTEN	rw	0x0	PVDRSTEN: PVD 复位使能 (PVD reset enable) 0: 禁用 PVD 产生复位 1: 使能 PVD 产生复位
5	LSI_OEN_LV	rw	0x0	LSI_OEN_LV: LSI 输出使能低电压(LSI output enable lower voltage) 0: LSI 输出使能低电压禁止 1: LSI 输出使能低电压使能 注: 1.此位可被电源复位清除。2.LSI 作为系统时钟、CSS 时钟安全系统、MCO 时钟、TIMER5 触发时，需要同时使能 LSION 和 LSI_OEN_LV。
4:2	Reserved			始终读为 0。
1	LSIRDY	r	0x0	LSIRDY: 内部低速时钟就绪 (Internal low-speed oscillator ready) 由硬件置'1'或清'0'来指示内部 40KHz 振荡器是否就绪。 在 LSION 清零后，3 个 AHB 时钟后 LSIRDY 被清零。 0: 内部 40KHz 振荡器时钟未就绪 1: 内部 40KHz 振荡器时钟就绪
0	LSION	rw	0x0	LSION: 内部低速振荡器使能 (Internal low-speed oscillator enable) 通过软件置'1'或清除，或由电源复位清除。 0: 禁用内部 40KHz 振荡器 1: 使能内部 40KHz 振荡器

### 5.3.16 系统配置寄存器 (RCC\_SYSCFG)

偏移地址: 0x3C

复位值: 0x00000101

访问: 0-3 等待周期, 字, 半字和字节访问

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

Field	Reserved																	
Type																		
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Field	Rese rvved	OS CLP FEN	Res.	PAD_OSC_TRIM								Reserved				DA TA PR EF ET CH	SE CT OR _1K _C FG	PR OG _CH ECK _EN
Type		rw		rw												rw	rw	rw

Bit	Field	Type	Reset	Description
31:15	Reserved			始终读为 0。
14	OSC_LPF_EN	rw	0x00	OSC_LPF_EN: 外接晶振低通滤波使能 0: 禁止 1: 使能
13	Reserved			始终读为 0。
12: 8	PAD_OSCTRIM	rw	0x01	PAD_OSC_TRIM: 外接晶振校准值
7: 3	Reserved			始终读为 0。
2	DATA_PREFETCH	rw	0x00	DATA_PREFETCH: DATA 预取模块使能位 1: 禁止 0: 打开
1	SECTOR_1K _CFG	rw	0x1	SECTOR_1K_CFG: Flash 页擦除时擦除的大小。 1: 1K 字节 0: 512 字节
0	PROG_CHECK_EN	rw	0x1	PROG_CHECK_EN: 写 Flash 时是否检查 Flash 内的数据是否是 FF。(硬件固定为 1) 1: 检查 0: 不检查

### 5.3.17 时钟配置寄存器 2 (RCC\_CFGR2)

偏移地址: 0x40

复位值: 0x00031F00

访问：无等待状态，字，半字和字节访问

只有当访问发生在时钟切换时，才会插入 1 或 2 个等待周期。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															APB1_CLK_HV_PRE
Type																rw
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved			FSMC_PRE					Reserved					TIMADV_PRE	TIMADV_CK_SEL	
Type														rw		

Bit	Field	Type	Reset	Description
31:20	Reserved			始终读为 0。
19: 16	APB1_CLK_HV_PRE	rw	0x3	APB1_CLK_HV_PRE: APB1_HV 输出时钟分频系数 0011:8 分频 0100:10 分频 ... 1110:30 分频 1111:32 分频 注：配置值必须介于 0x03-0x0F
15: 13	Reserved			始终读为 0。
12: 8	FSMC_PRE	rw	0x0	FSMC_PRE : FSMC_PRE 输出时钟分频系数 0000:1CLK 周期=2HCLK 周期 0001:1CLK 周期=4HCLK 周期 0010:1CLK 周期=6HCLK 周期 ... 1110:1CLK 周期=60HCLK 周期 1111:1CLK 周期=64HCLK 周期 注：在异步访问 NOR 闪存、SRAM 或 ROM 时，此参数不起作用。
7: 4	Reserved			始终读为 0。

Bit	Field	Type	Reset	Description
3:1	TIMADV_PRE	rw	0x0	<p>TIMADV_PRE: timadv_clk 预分频 由软件来控制系统时钟 (SYSCLK) 的预分频系数。</p> <p>0xx: timadv_clk 不分频 100: timadv_clk2 分频 101: timadv_clk4 分频 110: timadv_clk8 分频 111: timadv_clk16 分频</p>
0	TIMADV_CKSEL	rw	0x0	<p>TIMADV_CKSEL: TIMADV 时钟选择 0: TIMADV 时钟选择 timbas_clk (PCLK2 的 2 分频) 1: TIMADV 时钟选择 timadv_clk</p>

### 5.3.18 内部时钟校准寄存器 (RCC\_ICSCR)

偏移地址: 0x44

复位值: 0x0200 0000

访问: 无等待状态, 字, 半字和字节访问

只有当访问发生在时钟切换时, 才会插入 1 或 2 个等待周期。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved						HSI_CAL_SFT									
Type	rw															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	HSI_CAL_SEL						Reserved									
Type	rw															

Bit	Field	Type	Reset	Description
31:26	Reserved			始终读为 0。

Bit	Field	Type	Reset	Description
25: 16	HSI_CAL_SFT	rw	0x200	HSI_CAL_SFT：内部高速时钟校准(Internalhigh-speedclockcalibration)在系统启动时，这些位被自动初始化为出厂校准值，用户可以写入其他校准值，但读出始终为出厂校准值。如果 HSICALSEL=0x1F，写入的值可以重新校正 HSI 频率，否则写入的值不起作用。与原 HSICALSEL 设置一致。
15:11	HSI_CAL_SEL	rw	0x00	HSI_CAL_SEL：选择内部高速时钟校准值初值为 0，写入 1F 后仍读出为 0。1F：选择寄存器 HSI_CAL_SFT 的值其他：选择出厂校准值
10: 1	Reserved			始终读为 0。
0	TRIM_CRS_SEL	rw	0x00	TRIM_CRS_SEL：HSITRIM 值是否使用 CRS 模块作为来源 0:不使用 1:使用

### 5.3.19 PLL 配置寄存器 (RCC\_PLLCFGR)

偏移地址：0x48

复位值：0x0018031C

访问：无等待状态，字，半字和字节访问

只有当访问发生在时钟切换时，才会插入 1 或 2 个等待周期。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															PLLMUL
Type																rw
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved					PLLDIV			Reserved		PLL_LDS		PLLICTRL	PLXLX	PLTPRE	PLLSRC
Type						rw			rw		rw		rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:23	Reserved			始终读为 0。
22:16	PLLMUL	rw	0x18	PLLMUL: PLL 倍频系数 (PLL multiplication factor)
15:11	Reserved			始终读为 0。
10:8	PLLDIV	rw	0x03	PLLDIV: PLL 分频系数 (PLL divide factor)
7:6	Reserved			始终读为 0。
5:4	PLL_LDS	rw	0x01	PLL_LDS:PLL 锁定检测器精度选择(PLL lock detector accuracy select)
3:2	PLL_ICTRL	rw	0x03	PLL_ICTRL:PLL CP 电流控制信号 (PLL CP current control signals)
1	PLLXTPRE	rw	0x00	PLLXTPRE: HSE 分频器用作 PLL 输入 通过软件置'1'或清除来分频 HSE 后用作 PLL 输入时钟。 只有当 PLL 被禁用时，才能写入此位。 0: HSE 不分频 1: HSE2 分频
0	PLLSRC	rw	0x00	PLLSRC: PLL 输入时钟源 (PLL entry clock source) 通过软件置'1'或清除来选择 PLL 输入时钟源。 只有当 PLL 被禁用时，才能写入此位。 0: HSI 时钟用作 PLL 输入时钟 1: HSE 时钟用作 PLL 输入时钟

### 5.3.20 HSI 延迟寄存器 (RCC\_HSIDLY)

偏移地址: 0x80

复位值: 0x00000001E

访问: 无等待状态, 字, 半字和字节访问

只有当访问发生在时钟切换时, 才会插入 1 或 2 个等待周期。

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

Bit	Field	Type	Reset	Description
31:8	Reserved			始终读为 0。
7: 0	HSI_EQUCNT	rw	0x1e	HSI 延迟时间

### 5.3.21 HSE 延迟寄存器 (RCC\_HSEDLY)

偏移地址: 0x84

复位值: 0x0000BB80

访问: 无等待状态, 字, 半字和字节访问

只有当访问发生在时钟切换时, 才会插入 1 或 2 个等待周期。

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

Bit	Field	Type	Reset	Description
31:16	Reserved			始终读为 0。
15:0	HSE_EQUCNT	rw	0xBB80	HSE 延迟时间

### 5.3.22 PLL 延迟寄存器 (RCC\_PLLDLY)

偏移地址: 0x88

复位值: 0x0000003EB

访问: 无等待状态, 字, 半字和字节访问

只有当访问发生在时钟切换时, 才会插入 1 或 2 个等待周期。

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

Bit	Field	Type	Reset	Description
31:8	Reserved			始终读为 0。
7: 0	PLL_EQU_CNT	rw	10'd1000	PLL 延迟时间

## 6 系统控制器 (SYSCFG)

### 6.1 SYSCFG 简介

该芯片具有一组系统配置寄存器。这些寄存器的主要功能如下：

- 1) 管理连接到 GPIO 口的外部中断。
- 2) 重映射存储器到代码起始区域。
- 3) 外部中断引脚配置

### 6.2 寄存器描述

表 6-1 SYSCFG 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	SYSCFG_CFGR	SYSCFG 配置寄存器	0x0000000X
0x08	SYSCFG_EXTICR1	外部中断配置寄存器 1	0x00000000
0x0C	SYSCFG_EXTICR2	外部中断配置寄存器 2	0x00000000
0x10	SYSCFG_EXTICR3	外部中断配置寄存器 3	0x00000000
0x14	SYSCFG_EXTICR4	外部中断配置寄存器 4	0x00000000
0x18	SYSCFG_CFGR	SYSCFG 配置寄存器 2	0x00000000
0x1C	SYSCFG_PDETCSR	电源检测配置状态寄存器	0x00000000
0x20	SYSCFG_VOSDLY	VOSDLY 配置寄存器	0x000001F4

#### 6.2.1 SYSCFG 配置寄存器 (SYSCFG\_CFGR)

该寄存器专门用于配置内存起始区域映射和 DMA 请求重映射。具有两个可配置内存起始 0x00000000 地址存储区类型的控制位，这俩个控制位可软件配置来屏蔽 BOOT 的选择。复位后，这俩个控制位为实际的 BOOT 模式配置。

偏移地址：0x00

复位值：0x20000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved	MODESEL		FC_ODATAEN	Reserved											
Type		rw	rw	rw												

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														MEM_MODE	
Type															rw	rw

Bit	Field	Type	Reset	Description
31	Reserved			保留, 始终读为 0
30:29	MODESEL	rw	0x01	FSMC 模式选择 01: 兼容 8080 协议接口 00: 兼容 NOR FLASH 接口 1x: 保留
28	FC_ODATAEN	rw	0x00	FSMC 地址数据复用引脚只能作为数据使用 1: 仅作为数据引脚使用 0: 允许作为数据地址使用
27:2	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
1:0	MEM_MODE	rw	0x00	<p>存储映射选择位 (Memory selection bit) 由软件设置和清除这些位。它控制存储器内部映射到地址 0x0000 0000。</p> <p>当复位后这些位值由 BOOT0 的引脚配置值和 nBOOT1 bit 值决定。</p> <p>x0: 主闪存存储器映射到 0x0000 0000 01: 系统闪存映射到 0x0000 0000 11: 嵌入式 RAM 映射到 0x0000 0000</p>

## 6.2.2 外部中断配置寄存器 1 (SYSCFG\_EXTICR1)

偏移地址: 0x08

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	EXTI3				EXTI2				EXTI1				EXTI0			
Type	rw	rw	rw	rw												

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0
15:0	EXTIx	rw	0x00	<p>EXTIx 配置 (x=0...3) (EXTIx configuration) 这些位可用于软件读写。用于选择 EXTIx 外部中断的输入源。</p> <p>0000: PA[x] 管脚 0001: PB[x] 管脚 0010: PC[x] 管脚 0011: PD[x] 管脚 0100: PE[x] 管脚 0101: PF[x] 管脚 0110: PG[x] 管脚</p>

### 6.2.3 外部中断配置寄存器 2 (SYSCFG\_EXTICR2)

偏移地址: 0x0C

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	EXTI7				EXTI6				EXTI5				EXTI4			
Type	rw	rw	rw	rw												

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0
15:0	EXTIx	rw	0x00	EXTIx 配置 (x=4...7) (EXTIx configuration) 这些位可用于软件读写。用于选择 EXTIx 外部中断的输入源。 0000: PA[x] 管脚 0001: PB[x] 管脚 0010: PC[x] 管脚 0011: PD[x] 管脚 0100: PE[x] 管脚 0101: PF[x] 管脚 0110: PG[x] 管脚

### 6.2.4 外部中断配置寄存器 3 (SYSCFG\_EXTICR3)

偏移地址: 0x10

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	EXTI11				EXTI10				EXTI9				EXTI8			
Type	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0
15:0	EXTIx	rw	0x00	<p>EXTIx 配置 (<math>x=8\cdots 11</math>) (EXTIx configuration) 这些位可用于软件读写。用于选择 EXTIx 外部中断的输入源。</p> <p>0000: PA[x] 管脚      0001: PB[x] 管脚      0010: PC[x] 管脚      0011: PD[x] 管脚      0100: PE[x] 管脚      0101: PF[x] 管脚      0110: PG[x] 管脚</p>

### 6.2.5 外部中断配置寄存器 4 (SYSCFG\_EXTICR4)

偏移地址: 0x14

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	EXTI15				EXTI14				EXTI13				EXTI12			
Type	rw	rw	rw	rw												

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0
15:0	EXTIx	rw	0x00	<p>EXTIx 配置 (<math>x=12\cdots 15</math>) (EXTIx configuration) 这些位可用于软件读写。用于选择 EXTIx 外部中断的输入源。</p> <p>0000: PA[x] 管脚      0001: PB[x] 管脚      0010: PC[x] 管脚      0011: PD[x] 管脚      0100: PE[x] 管脚      0101: PF[x] 管脚</p>

Bit	Field	Type	Reset	Description
				0110: PG[x] 管脚

## 6.2.6 SYSCFG 配置寄存器 (SYSCFG\_CFGR2)

偏移地址: 0x18

复位值: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Reserved											MA C_S	MII_ RMI	Reserved		I2C 2_M OD E _SE L	I2C1 _MO DE _SE L
Type												rw	rw			rw	rw

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved															
Type															rw	rw

Bit	Field	Type	Reset	Description
31:22	Reserved			保留, 始终读为 0
21	MAC_SPD_SEL	rw	0x00	MAC_SPD_SEL: MAC 简化媒体独立接口速度选择 1: 100Mbps 0: 10Mbps
20	MII_RMII_SEL	rw	0x00	MII_RMII_SEL: Ethernet PHY 接口选择 1: 选择 RMII 接口 0: 选择 MII 接口 注意: 必须在 MAC 处于复位状态下并且在使能 MAC 时钟 之前完成此配置。
19:18	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
17	I2C2_MODE_SEL	rw	0x00	I2C2 端口模式选择位 0: 开漏模式 1: 推挽模式
16	I2C1_MODE_SEL	rw	0x00	I2C1 端口模式选择位 0: 开漏模式 1: 推挽模式
15:0	Reserved			保留, 始终读为 0

### 6.2.7 电源检测配置状态寄存器 (SYSCFG\_PDETCSR)

偏移地址: 0x1C

复位值: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	VBAT_DIV3_EN					VDTLS	VDTE	Res.	VDTO	PVDO	PLS						PVDE
Type					rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
31:12	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
11	VBAT_DIV3_EN	rw	0x00	VBATDIV3_EN: ADC 检测 VBat_DIV3 分压值使能 1: 使用 ADC 检测 VBat_DIV3 分压值 0: 不使用 ADC 检测 VBat_DIV3 分压值
10:9	VDTLS	rw	0x00	VDTLS: VDT 检测阈值选择 11: 选择 0.9V 参考电压 10: 选择 1.0V 参考电压 01: 选择 1.1V 参考电压 00: 选择 1.2V 参考电压
8	VDTE	rw	0x00	VDTE: VDT 使能 1: VDT 使能 0: VDT 禁止
7	Reserved			保留, 始终读为 0
6	VDTO	rw	0x00	VDTO: VDT 输出状态 1: VDT 输出高 0: VDT 输出低
5	PVDO	rw	0x00	PVDO: PVD 输出状态 1: PVD 输出高 0: PVD 输出低 注意: PVDE 必须使能。
4:1	PLS	rw	0x00	PLS: PVD 阈值选择 0000: 1.8 V 0001: 2.1 V 0010: 2.4 V 0011: 2.7 V 0100: 3.0 V 0101: 3.3 V 0110: 3.6 V 0111: 3.9 V 1000: 4.2 V 1001: 4.5 V 1010: 4.8 V

Bit	Field	Type	Reset	Description
0	PVDE	rw	0x00	PVDE: PVD 使能 1: PVD 使能 0: PVD 禁止

## 6.2.8 VOSDLY 配置寄存器 (SYSCFG\_VOSDLY)

偏移地址: 0x20

复位值: 0x000001F4

访问: 无等待状态, 字, 半字和字节访问

只有当访问发生在时钟切换时, 才会插入 1 或 2 个等待周期。

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

Bit	Field	Type	Reset	Description
31:10	Reserved			始终读为 0。
9: 0	VOSDLY_CNT	rw	10'd500	VOSDLY_CNT : VOS 延迟时间

## 7 中断和事件(EXTI)

### 7.1 EXTI 简介

嵌套向量中断控制器（NVIC）连接处理器核，能够有效的处理晚到的中断与延迟较低的中断。

嵌套向量中断控制器管理着包括核异常等中断。NVIC 内部可配置为 8 个不同的优先等级。其它更多的异常和与 NVIC 编程的细节请参考《Cortex-Mx 技术参考手册》。

EXTI（中断/事件唤醒器）边沿检测电路并能够向处理器内核产生中断唤醒事件。主要包括三种触发类型：上升沿触发、下降沿触发和任意边沿触发。

### 7.2 功能框图

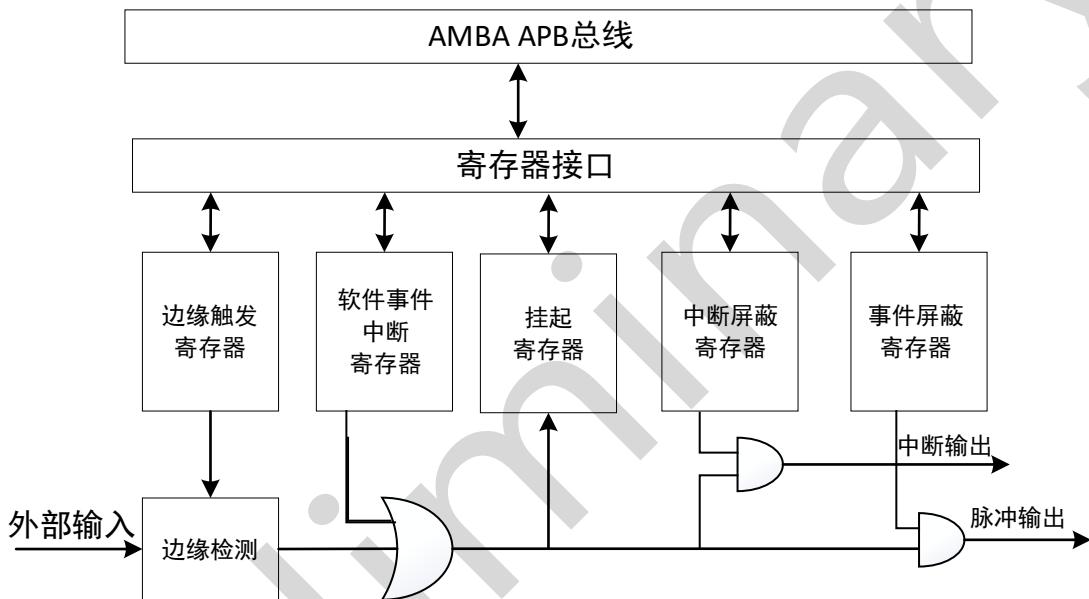


图 7-1 EXTI 结构框图

### 7.3 主要特征

EXTI 控制器的主要特性如下：

- 1) 独立触发与屏蔽每个中断；
- 2) 软件配置中断/事件；
- 3) 状态位保存对应每条中断线的状态；
- 4) 检测低于 APB2 时钟宽度脉冲的外部信号。具体请参考数据手册电气部分参数；

### 7.4 中断说明&向量表

在 Handler 模式下，cortex-m0 处理器与内嵌中断向量控制（NVIC）对所有的异常进行优先级区分处理。当异常发生时，系统会将当前处理的工作压栈，执行完中断服务程序后出栈。

取向量与当前工作的压栈并行进行的，提高了中断的效率，下表分别列出了异常向量与中断向量。

表 7-1 异常向量表

位置	优先级	优先级类型	名称	说明	地址
				保留	0x0000 0000
	-3	固定	Reset	复位	0x0000 0004
	-2	固定	NMI	不可屏蔽中断 RCC 时钟安全系统 (CSS) 联接到 NMI 向量	0x0000 0008
	-1	固定	硬件失效 (HardFault)	所有类型的失效	0x0000 000C
	0	可设置	存储管理 (MemManage)	存储器管理	0x0000 0010
	1	可设置	总线错误 (BusFault)	预取指失败, 存储器访问失败	0x0000 0014
	2	可设置	错误应用 (UsageFault)	未定义的指令或非法状态	0x0000 0018
				保留	0x0000 001C ~ 0x0000 002B

表 7-2 中断向量表

位置	优先级	优先级类型	名称	说明	地址
	3	可设置	SVCALL	通过 SWI 指令的系统服务调用	0x0000 002C
	4	可设置	调试监控 (DebugMonitor)	调试监控器	0x0000 0030
				保留	0x0000 0034
	5	可设置	PendSV	可挂起的系统服务	0x0000 0038
	6	可设置	SysTick	系统嘀嗒定时器	0x0000 003C
0	7	可设置	WWDG IWDG	窗口看门狗与独立看门狗中断	0x0000 0040
1	8	可设置	PVD	连到 EXTI16 的电源电压检测 (PVD) 中断	0x0000 0044
2	9	可设置	RTC BKP	RTC 和 BKP 全局中断	0x0000 0048
3	10	可设置	FLASH	闪存全局中断	0x0000 004C
4	11	可设置	RCC CRS	RCC CRS 全局中断	0x0000 0050
5	12	可设置	EXTI0、1	EXTI 线 [1:0] 中断	0x0000 0054
6	13	可设置	EXTI2、3	EXTI 线 [3:2] 中断	0x0000 0058
7	14	可设置	EXTI4、15	EXTI 线 [15:4] 中断	0x0000 005C

位置	优先级	优先级类型	名称	说明	地址
8	15	可设置	HWDIV	HWDIV 全局中断	0x0000 0060
9	16	可设置	DMA1 通道 1	DMA1 通道 1 全局中断	0x0000 0064
10	17	可设置	DMA1 通道 2、3	DMA1 通道 2、3 全局中断	0x0000 0068
11	18	可设置	DMA1 通道 4、5	DMA1 通道 4、5 全局中断	0x0000 006C
12	19	可设置	ADC1 COMP	ADC1 中断和比较器中断 (与 EXTI19、20 组合)	0x0000 0070
13	20	可设置	TIM1 BRK UP TRG COM	TIM1 刹车、更新、触发、COM 中断	0x0000 0074
14	21	可设置	TIM1 CC	TIM1 捕捉比较中断	0x0000 0078
15	22	可设置	TIM2	TIM2 全局中断	0x0000 007C
16	23	可设置	TIM3	TIM3 全局中断	0x0000 0080
17	24	可设置		保留	0x0000 0084
18	25	可设置		保留	0x0000 0088
19	26	可设置	TIM14	TIM14 全局中断	0x0000 008C
20	27	可设置		保留	0x0000 0090
21	28	可设置	TIM16	TIM16 全局中断	0x0000 0094
22	29	可设置	TIM17	TIM17 全局中断	0x0000 0098
23	30	可设置	I2C1	I2C1 全局中断	0x0000 009C
24	31	可设置		保留	0x0000 00A0
25	32	可设置	SPI1	SPI1 全局中断	0x0000 00A4
26	33	可设置	SPI2	SPI2 全局中断	0x0000 00A8
27	34	可设置	UART1	UART1 全局中断	0x0000 00AC
28	35	可设置	UART2	UART2 全局中断	0x0000 00B0
29	36	可设置	CSM	CSM 全局中断	0x0000 00B4
30	37	可设置	CAN	CAN 全局中断	0x0000 00B8
31	38	可设置	USB	USB 全局中断(EXTI18)	0x0000 00BC

## 7.5 功能描述

### 7.5.1 唤醒事件管理

执行 WFI 指令时 CPU 进入睡眠模式，任意一个由 NVIC 识别的外设中断都可以唤醒 CPU 退出睡眠模式。

执行 WFE 指令时进入了睡眠模式，当任一事件发生时，MCU 退出睡眠模式。

通过如下方式配置产生：

1) 打开外设控制器中的一个中断使能位，关闭对应的 NVIC 中使能，配置 CPU 的系统控制寄存器中的 SEVONPEND 位等于 1。唤醒 CPU 后，在 NVIC 中断清除挂起寄存器中需要清除相应外设的中断挂起位和外设 NVIC 中断通道挂起位。

2) 通过配置内部或外部 EXTI 线为事件模式，CPU 从 WFE 恢复后，由于没有置位对应事件线的挂起位，不用执行清除相应外设的中断挂起位或 NVIC 中断通道挂起位的操作。

关于利用外部 I/O 端口作为唤醒事件的部分，可参考下节的功能描述。

## 7.5.2 中断功能描述

要使能中断功能，产生中断，首先配置好并使能中断线。配置边缘检测触发寄存器为需要的触发类型，打开相应的中断屏蔽寄存器的对应位允许中断请求。对应的外部中断线检测到配置的触发条件时，产生一个中断请求，挂起寄存器对应位置 1，通过对挂起寄存器对应位写 1，将清除中断。

配置产生事件，首先配置好并使能事件线。配置边缘检测触发寄存器为需要的触发类型，打开相应的中断屏蔽寄存器的对应位允许中断请求。对应的外部中断线检测到配置的触发条件时，产生一个中断请求，挂起寄存器对应位置 1，通过对挂起寄存器对应位写 1，将清除中断。

使能软件中断/事件寄存器的对应位，也能够产生中断/事件请求。

## 7.5.3 硬件中断输出

配置硬件中断源的具体步骤如下：

- 1) 打开对应中断线的屏蔽位 (EXTI\_IMR)，使能中断；
- 2) 配置对应中断线的触发寄存器位 (EXTI\_RISE/EXTI\_FALL)；
- 3) 打开对应连接到 NVIC 的中断通道，使得中断请求能够传递到 CPU，被正确的响应；

## 7.5.4 硬件事件输出

配置硬件事件源的具体步骤如下：

- 1) 打开对应事件线的屏蔽位 (EXTI\_EMR)；
- 2) 配置对应事件线的触发寄存器位 (EXTI\_RISE/EXTI\_FALL)；

## 7.5.5 软件中断事件输出

支持通过软件的方式配置产生中断与事件，具体步骤如下：

- 1) 使能事件或中断使能位 (EXTI\_IMR, EXTI\_EMR)
- 2) 配置软件中断事件寄存器对应位为 1 (EXTI\_SWIER)

## 7.5.6 外部中断映射

GPIO 对应的 16 个外部中断/事件映射关系如下表所示：

表 7-3 EXTI 触发源

外部中断线	IO 映射	控制位
EXTI0	PA0;PB0;PC0;PD0	SYSCFG_EXTICR1 寄存器中的 EXTI0
EXTI1	PA1;PB1;PC1;PD1	SYSCFG_EXTICR1 寄存器中的 EXTI1
EXTI2	PA2;PB2;PC2;PD2	SYSCFG_EXTICR1 寄存器中的 EXTI2
EXTI3	PA3;PB3;PC3;PD3	SYSCFG_EXTICR1 寄存器中的 EXTI3
EXTI4	PA4;PB4;PC4;PD4	SYSCFG_EXTICR2 寄存器中的 EXTI4
EXTI5	PA5;PB5;PC5;PD5	SYSCFG_EXTICR2 寄存器中的 EXTI5
EXTI6	PA6;PB6;PC6;PD6	SYSCFG_EXTICR2 寄存器中的 EXTI6
EXTI7	PA7;PB7;PC7;PD7	SYSCFG_EXTICR2 寄存器中的 EXTI7
EXTI8	PA8;PB8;PC8;PD8	SYSCFG_EXTICR3 寄存器中的 EXTI8
EXTI9	PA9;PB9;PC9;PD9	SYSCFG_EXTICR3 寄存器中的 EXTI9
EXTI10	PA10;PB10;PC10;PD10	SYSCFG_EXTICR3 寄存器中的 EXTI10
EXTI11	PA11;PB11;PC11;PD11	SYSCFG_EXTICR3 寄存器中的 EXTI11
EXTI12	PA12;PB12;PC12;PD12	SYSCFG_EXTICR4 寄存器中的 EXTI12
EXTI13	PA13;PB13;PC13;	SYSCFG_EXTICR4 寄存器中的 EXTI13
EXTI14	PA14;PB14;PC14;	SYSCFG_EXTICR4 寄存器中的 EXTI14
EXTI15	PA15;PB15;PC15;	SYSCFG_EXTICR4 寄存器中的 EXTI15

其他的外部中断/事件控制器的连接如下：

- 1) EXTI 线 16 连接到 PVD 输出
- 2) EXTI 线 17 连接到 RTC 阔钟事件
- 3) EXTI 线 18 连接到 USB 总线挂起中断
- 4) EXTI 线 19 连接到比较器 1 输出
- 5) EXTI 线 20 连接到比较器 2 输出
- 6) EXTI 线 24 连接到 IWDG 中断

## 7.6 寄存器描述

表 7-4 EXTI 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	EXTI IMR	中断屏蔽寄存器	0x00000000
0x04	EXTI EMR	事件屏蔽寄存器	0x00000000
0x08	EXTI RTSR	上升沿触发选择寄存器	0x00000000
0x0C	EXTI FTSR	下降沿触发选择寄存	0x00000000
0x10	EXTI SWIER	软件中断事件寄存器	0x00000000

Offset	Acronym	Register Name	Reset
0x14	EXTI PR	挂起寄存器	0x00000000

### 7.6.1 中断屏蔽寄存器 (EXTI\_IMR)

偏移地址: 0x0

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.							IMR 24	Res.			IMR 20	IMR 19	IMR 18	IMR 17	IMR 16
Type								rw				rw	rw	rw	rw	rw
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	IMR 15	IMR 14	IMR 13	IMR 12	IMR 11	IMR 10	R9	IM 8	IM R7	IM R6	IM R5	IMR 4	IMR 3	IMR 2	IMR 1	IMR 0
Type	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:25	Reserved			保留, 始终读为 0
24	IMRx	rw	0x0	线 x 中断使能位 1: 配置该位为 1, 使能线 x 对应的中断 0: 配置该位为 0, 禁止线 x 对应的中断
23:21	Reserved			保留, 始终读为 0
20:0	IMRx	rw	0x0	线 x 中断使能位 1: 配置该位为 1, 使能线 x 对应的中断 0: 配置该位为 0, 禁止线 x 对应的中断

### 7.6.2 事件屏蔽寄存器 (EXTI\_EMR)

偏移地址: 0x04

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.							EM R24	Res.			EM R20	EM R19	EM R18	EM R17	EM R16
Type								rw				rw	rw	rw	rw	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Field	EM R15	EM R14	EM R13	EM R12	EM R11	EM R10	EM R9	EM R8	EM R7	EM R6	EM R5	EM R4	EM R3	EM R2	EM R1	EM R0
Type	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:25	Reserved			保留, 始终读为 0
24	EMRx	rw	0x0	线 x 事件使能位 1: 配置该位为 1, 使能线 x 对应的事件 0: 配置该位为 0, 禁止线 x 对应的事件
23:21	Reserved			保留, 始终读为 0
20:0	EMRx	rw	0x0	线 x 事件使能位 1: 配置该位为 1, 使能线 x 对应的事件 0: 配置该位为 0, 禁止线 x 对应的事件

### 7.6.3 上升沿触发选择寄存器 (EXTI\_RTSR)

偏移地址: 0x08

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.							TR24	Res.			TR20	TR19	TR18	TR17	TR16
Type								rw				rw	rw	rw	rw	rw
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	TR15	TR14	TR13	TR12	TR11	TR10	TR9	TR8	TR7	TR6	TR5	TR4	TR3	TR2	TR1	TR0
Type	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:25	Reserved			保留, 始终读为 0
24	TRx	rw	0x0	线 x 对应中断或事件的触发极性 1: 配置该位为 1, 使能线 x 对应的上升沿触发中断或事件 0: 配置该位为 0, 禁止线 x 对应的上升沿触发中断或事件
23:21	Reserved			保留, 始终读为 0
20:0	TRx	rw	0x0	线 x 对应中断或事件的触发极性

Bit	Field	Type	Reset	Description
				1: 配置该位为 1, 使能线 x 对应的上升沿触发中断或事件 0: 配置该位为 0, 禁止线 x 对应的上升沿触发中断或事件

#### 7.6.4 下降沿触发选择寄存器 (EXTI\_FTSR)

偏移地址: 0x0C

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
Field	Res.								TR24	Res.				TR20	TR19	TR18	TR17	TR16
Type									rw					rw	rw	rw	rw	rw
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Field	TR15	TR14	TR13	TR12	TR11	TR10	TR9	TR8	TR7	TR6	TR5	TR4	TR3	TR2	TR1	TR0		
Type	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		

Bit	Field	Type	Reset	Description
31:25	Reserved			保留, 始终读为 0
24	TRx	rw	0x0	线 x 对应中断或事件的触发极性 1: 配置该位为 1, 使能线 x 对应的下降沿触发中断或事件 0: 配置该位为 0, 禁止线 x 对应的下降沿触发中断或事件
23:21	Reserved			保留, 始终读为 0
20:0	TRx	rw	0x0	线 x 对应中断或事件的触发极性 1: 配置该位为 1, 使能线 x 对应的下降沿触发中断或事件 0: 配置该位为 0, 禁止线 x 对应的下降沿触发中断或事件

#### 7.6.5 软件中断事件寄存器 (EXTI\_SWIER)

偏移地址: 0x10

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Res.								SWIE	Res.				SWIE	SWIE	SWIE	SWIE
Type									rw					rw	rw	rw	rw

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	SWI ER1 5	SW IER 14	SWIE R13	SWIE R12	SWIE R11	SWIE R10	SWIE R9	SWIE R8	SWI ER7	SWI ER6	SWI ER5	SWIE R4	SWIE R3	SWIE R2	SWIE R1	SWIE R0
Type	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:25	Reserved			保留, 始终读为 0
24	SWIERx	rw	0x0	线 x 上的软件中断或事件使能  当该位为 0 时, 写 1 将设置 EXTI_PR 中相应的挂起位, 同时如果配置对应的线 x 的 EXTI_EMR 或 EXTI_IMR 位为 1, 能够产生事件或中断。  注: 向 EXTI_PR 寄存器的对应位写 1, 可以清除该位
23:21	Reserved			保留, 始终读为 0
20:0	SWIERx	rw	0x0	线 x 上的软件中断或事件使能  当该位为 0 时, 写 1 将设置 EXTI_PR 中相应的挂起位, 同时如果配置对应的线 x 的 EXTI_EMR 或 EXTI_IMR 位为 1, 能够产生事件或中断。  注: 向 EXTI_PR 寄存器的对应位写 1, 可以清除该位

## 7.6.6 软件中断事件挂起寄存器 (EXTI\_PR)

偏移地址: 0x14

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Res.							PR2 4	Res.				PR2 0	PR1 9	PR1 8	PR1 7	PR1 6
Type	rc_ w1							rc_ w1	rc_ w1				rc_ w1	rc_ w1	rc_ w1	rc_ w1	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	PR1 5	PR1 4	PR1 3	PR1 2	PR1 1	PR1 0	PR9	PR8	PR7	PR6	PR5	PR4	PR3	PR2	PR1	PR0	
Type	rc_ w1																

Bit	Field	Type	Reset	Description
31:25	Reserved			保留, 始终读为 0
24	PRx	rw	0x0	<p>线 x 触发挂起位            1: 发生了选择的触发请求            0: 没有发生触发请求</p> <p>外部中断线上出现选择的边沿事件时, 该位被置 1, 写 1 清除该位, 也可以通过改变边沿检测的极性清除。</p>
23:21	Reserved			保留, 始终读为 0
20:0	PRx	rw	0x0	<p>线 x 触发挂起位            1: 发生了选择的触发请求            0: 没有发生触发请求</p> <p>外部中断线上出现选择的边沿事件时, 该位被置 1, 写 1 清除该位, 也可以通过改变边沿检测的极性清除。</p>

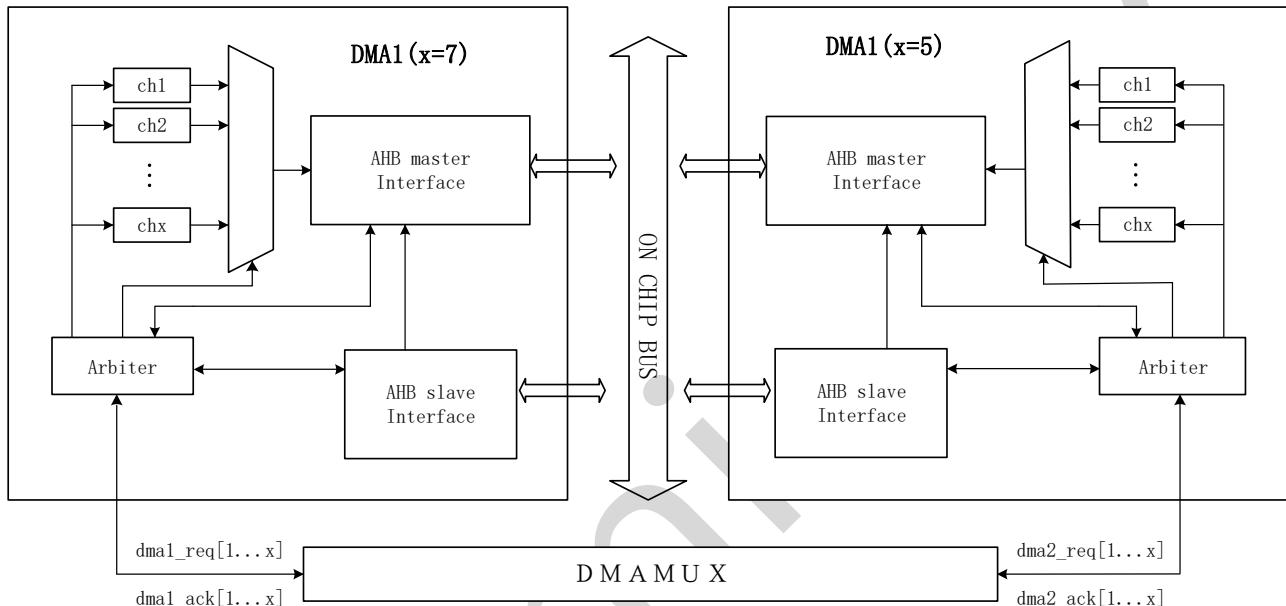
# 8 直接存储器访问控制器(DMA)

## 8.1 DMA 简介

DMA 控制器通过共享系统总线，实现无需 CPU 参与的快速自动数据传输。

有两个 DMA，DMA1 控制器有 7 个通道，DMA2 控制器有 5 个通道，多个外设 DMA 请求发送到对应通道上处理。

## 8.2 DMA 功能框图



## 8.3 DMA 主要特征

- 1) 两个 DMA，DMA1 控制器有 7 个独立的通道，DMA2 控制器有 5 个独立的通道，可通过寄存器配置相关功能。
- 2) 硬件发出的 DMA 请求与对应专用 DMA 通道直连。通过软件配置寄存器的方式也可以触发 DMA 通道请求。
- 3) 可以通过软件的方式配置寄存器决定 12 个通道请求之间的处理优先级(共有四级：很高、高、中等和低)，若优先级相同，则由硬件自动决定处理顺序(低编号通道请求优先处理)。
- 4) 数据源头与目的地的传输宽度可独立配置为字节、半字、全字。
- 5) 独立数据源头的宽度配置进行打包，并在目的地按照目的地的宽度配置进行拆包。要求源和目标地址必须根据各自配置的数据传输宽度对齐。
- 6) 支持循环缓冲器控制。
- 7) 每个通道支持 DMA 半传输，DMA 传输完成和 DMA 传输出错 3 种事件标志。各通道单独的中断请求由这 3 种事件标志逻辑或起来。
- 8) 支持存储器对存储器传输。
- 9) 支持数据传输方向为外设到存储器，存储器到外设。
- 10) 数据访问的源和目标可以是：SRAM、APB1、APB2 和 AHB 总线上的外设。

11) 数据的传输数量可以通过软件配置对应寄存器，最大值为 65536。

## 8.4 中断

DMA 半传输，DMA 传输完成和 DMA 传输出错为每个 DMA 通道都会产生的 3 种事件标志。各通道单独的中断请求由这 3 种事件标志逻辑或起来。

可以配置寄存器的对应位来使能这些中断，以满足程序的不同需求。

表 8-1 DMA 中断请求

中断事件	事件标志位	使能控制位
半传输	HTIF	HTIE
传输结束	TCIF	TCIE
传输出错	TEIF	TEIE

## 8.5 DMA

### 8.5.1 DMA 请求映像

DMA 控制器

从外设 TIMx、ADCx、SPIx、I2S、I2Cx、SDIO 和 UARTx 产生的请求，通过逻辑或输入到 DMA 控制器，为了避免冲突，在一个通道中，同时只能有一个外设 DMA 请求有效。参见下图的 DMA 请求映像。

外设本身的控制寄存器应有对应的 DMA 使能位，来独立控制外设是否发送 DMA 请求。

表 8-2 各个通道的 DMA 请求一览

外设	DMA1 通道 1	DMA1 通道 2	DMA1 通道 3	DMA1 通道 4	DMA1 通道 5	DMA1 通道 6	DMA1 通道 7	DMA2 通道 1	DMA2 通道 2	DMA2 通道 3	DMA2 通道 4	DMA2 通道 5
ADC	ADC1	ADC2										ADC3
SPI/I2S		SPI1_R X	SPI1_T X	SPI2_R X	SPI2_T X			SPI3_R X	SPI3_T X			
UART	UART6 RX	UART3 TX	UART3 RX	UART1 TX	UART1 RX	UART2 RX	UART2 TX	UART5_RX/ UART7_RX	UART5_TX/ UART7_TX	UART4_RX/ UART8_RX	UART6_TX	UART4_TX/ UART8_TX
I2C				I2C2_RX	I2C2_RX	I2C1_RX	I2C1_RX					
SDIO												SDIO
TIM1		TIM1_C C1	TIM1_C C2	TIM1_C C4 TIM1_T RIG TIM1_C OM	TIM1_U P	TIM1_C C3						
TIM2	TIM2_C C3	TIM2_U P			TIM2_C C1		TIM2_C C2 TIM2_C C4					

外设	DMA1 通道 1	DMA1 通道 2	DMA1 通道 3	DMA1 通道 4	DMA1 通道 5	DMA1 通道 6	DMA1 通道 7	DMA2 通道 1	DMA2 通道 2	DMA2 通道 3	DMA2 通道 4	DMA2 通道 5
TIM3		TIM3_C C3	TIM3_C C4 TIM3_U P			TIM3_C C1 TIM3_T RIG						
TIM4	TIM4_C C1			TIM4_C C2	TIM4_C C3		TIM4_U P					
TIM5								TIM5_C C4 TIM5_T RIG	TIM5_C C3 TIM5_U P		TIM5_C C2	TIM5_C C1
TIM6										TIM6_U P/ DAC_C H1		
TIM7											TIM7_U P/ DAC_C H2	
TIM8								TIM8_C C3 TIM8_U P	TIM8_C C4 TIM8_T RIG TIM8_C OM	TIM8_C C1		TIM8_C C2

- 1) 如果 `SYSCFG_CFGR` 寄存器的对应映射位被复位, DMA 请求被映射在这个 DMA 通道。
- 2) 如果 `SYSCFG_CFGR` 寄存器的对应映射位被置位, DMA 请求被映射在这个 DMA 通道。

## 8.6 功能描述

DMA 与 CPU 都是通过系统总线实现对存储器/外设数据的访问。当 CPU 和 DMA 访问冲突时, DMA 请求可能会占用系统总线, 此时 CPU 只能等待 DMA 传输完成释放总线。为了防止总线一直被 DMA 占用导致 CPU 无法工作, 总线仲裁器会执行相关的循环调度, 以此保证 CPU 至少可以获得一半的系统总线控制权。

### 8.6.1 DMA 处理

外设产生一个相关事件后, 会将 DMA 请求信号发送到 DMA 控制器对应通道。按照软件配置的 DMA 通道优先级, 或者硬件默认规则, DMA 控制器依次处理这些请求。DMA 响应外设请求, 通过总线访问外设的同时, DMA 控制器会发送给外设一个应答信号, 告知外设本次请求已响应。外设得到 DMA 的应答信号后, 会立即释放掉本次请求。DMA 侦测到外设请求消失后, 对应的应答信号也会随之释放掉, 本次 DMA 传输完成。

总之, 每个 DMA 传送由 3 个操作组成:

- 1) 加载源地址数据, 地址由软件配置。
- 2) 存储数据到目的地址, 地址由软件配置。
- 3) 执行一次 DMA 传输, 计数器 `DMA_CNDTRx` 从配置的传输数量开始递减, 表示剩余还有多少次 DMA 传输。

## 8.6.2 仲裁器

仲裁器决定 DMA 控制器优先解决哪个 DMA 请求。优先级分软硬件 2 种逻辑控制：

- 1) 软件：4 个等级优先级，每个通道的优先级可在 DMA\_CCRx 寄存器配置：
  - a) 最高优先级
  - b) 高优先级
  - c) 中等优先级
  - d) 低优先级
- 2) 硬件：先处理软件优先级高的请求，软件优先级配置相同则默认更低编号的通道优先。

## 8.6.3 DMA 通道

外设寄存器与存储器的固定地址通过 DMA 通道进行 DMA 传输。数据的传输数量可以通过软件配置对应寄存器，最大值为 65536。从配置的传输数量开始，每次传输后 DMA\_CNDTRx 都会递减，指示剩余还需多少次 DMA 传输。

### 8.6.3.1 可编程数据宽度

配置 DMA\_CCRx 寄存器中的 PSIZE 和 MSIZE 位，可以控制外设和存储器对应的传输数据宽度。

### 8.6.3.2 指针增量

配置 DMA\_CCRx 寄存器中 PINC 和 MINC 标志位，外设与存储器的访问地址可以按照步长累加，不需要每次都去设置访问地址。

清零增量模式寄存器则每次 DMA 传输固定访问同一个地址。

配置为增量模式时，下一个要传输的地址将是前一个地址加上步长，步长取决与所选的数据宽度 1（8 位）、2（16 位）或 4（32 位）。首个传输的地址存放在 DMA\_CPARx / DMA\_CMARx 寄存器中。

通道配置为非循环模式，DMA\_CNDTRx 递减为 0 后，不会继续进行 DMA 传输。

### 8.6.3.3 通道配置

以下为 DMA 通道 x 的配置流程（x 表示通道编号）：

- 1) 操作 DMA\_CPARx 寄存器，配置外设寄存器的地址。DMA 传输时该外设地址为源或目标地址取决于 DMA 传输方向。
- 2) 操作 DMA\_CMARx 寄存器，配置数据存储器的地址。DMA 传输时需要从该存储器地址加载或者存储数据取决于 DMA 传输方向。
- 3) 操作 DMA\_CNDTRx，配置 DMA 传输数量。DMA 传输完成一次，该值减 1，且在 DMA 传输期间该寄存器不可被软件写操作。
- 4) 操作 DMA\_CCRx 寄存器的 PL[1:0] 位，配置通道的优先级。
- 5) 操作 DMA\_CCRx 寄存器，配置数据传输方向、循环模式、外设和存储器的增量模式、外设

和存储器的数据宽度、中断产生种类。

- 6) 操作 DMA\_CCRx 寄存器的 ENABLE 位，使能这个通道。该通道使能后，就可以进行正常的 DMA 工作，响应外设请求，进行 DMA 传输。

半传输标志 (HTIF) 被硬件置‘1’，表示当前 DMA 传输数量为配置传输数量的一半。若想产生中断，则需使能半传输中断位 (HTIE)。

传输完成标志 (TCIF) 被硬件置‘1’，表示当前 DMA 配置的传输数据已全部传输完毕。若想产生中断，则需使能传输完成中断位 (TCIE)。

#### 8.6.3.4 循环模式

如果需要循环读写缓冲区或者是进行连续的数据传输(如 ADC 的扫描模式)，可以进入循环模式。置‘1’DMA\_CCRx 寄存器中的 CIRC 位，使能循环模式。在循环模式下，DMA\_CNDTRx 被递减为 0 时，会自动重新加载先前配置的数值，随后重新进行递减操作，DMA 会继续传输数据。

#### 8.6.3.5 存储器到存储器模式

DMA 支持存储器到存储器的访问，不需要外设的参与。置‘1’DMA\_CCRx 寄存器中的 MEM2MEM 位，同时置‘1’DMA\_CCRx 寄存器中的通道使能位，即可开始 DMA 传输。若 DMA\_CNDTRx 递减为 0，则 DMA 传输结束。

存储器到存储器的访问不支持循环模式。

### 8.6.4 可编程的数据传输宽度，对齐方式和数据大小端

当 PSIZE 和 MSIZE 不相同时，DMA 模块按照下表进行数据对齐。

源数据比特位对齐写入目标地址：

若目标数据传输宽度大于源数据传输宽度，则目标数据宽度多余位补 0 处理。

若目标数据传输宽度小于源数据传输宽度，则源数据宽度多余部分截断处理。

表 8-3 可配置的数据传输宽度和大小端操作(当 PINC=MINC=1)，传输数目为 4

组合类型	传输宽度		传输操作	
	源端	目标	源 (地址   数据)	目标 (地址   数据)
源端传输宽度 等于 目标传输宽度 地址步长、数据 宽度一致	8	8	在 0x0 读 B0[7:0] 在 0x1 读 B1[7:0] 在 0x2 读 B2[7:0] 在 0x3 读 B3[7:0]	在 0x0 写 B0[7:0] 在 0x1 写 B1[7:0] 在 0x2 写 B2[7:0] 在 0x3 写 B3[7:0]

组合类型	传输宽度		传输操作	
	源端	目标	源 (地址 + 数据)	目标 (地址 + 数据)
源端传输宽度 小于 目标传输宽度 地址步长为传输 宽度÷8 目标数据多余 bit 位补 0	16	16	在 0x0 读 B1B0[15:0] 在 0x2 读 B3B2[15:0] 在 0x4 读 B5B4[15:0] 在 0x6 读 B7B6[15:0]	在 0x0 写 B1B0[15:0] 在 0x2 写 B3B2[15:0] 在 0x4 写 B5B4[15:0] 在 0x6 写 B7B6[15:0]
	32	32	在 0x0 读 B3B2B1B0[31:0] 在 0x4 读 B7B6B5B4[31:0] 在 0x8 读 BBBAB9B8[31:0] 在 0xC 读 BFBEBDBC[31:0]	在 0x0 写 B3B2B1B0[31:0] 在 0x4 写 B7B6B5B4[31:0] 在 0x8 写 BBBAB9B8[31:0] 在 0xC 写 BFBEBDBC[31:0]
	8	16	在 0x0 读 B0[7:0] 在 0x1 读 B1[7:0] 在 0x2 读 B2[7:0] 在 0x3 读 B3[7:0]	在 0x0 写 00B0[15:0] 在 0x2 写 00B1[15:0] 在 0x4 写 00B2[15:0] 在 0x6 写 00B3[15:0]
	8	32	在 0x0 读 B0[7:0] 在 0x1 读 B1[7:0] 在 0x2 读 B2[7:0] 在 0x3 读 B3[7:0]	在 0x0 写 000000B0[31:0] 在 0x4 写 000000B1[31:0] 在 0x8 写 000000B2[31:0] 在 0xC 写 000000B3[31:0]
	16	32	在 0x0 读 B1B0[15:0] 在 0x2 读 B3B2[15:0] 在 0x4 读 B5B4[15:0] 在 0x6 读 B7B6[15:0]	在 0x0 写 0000B1B0[31:0] 在 0x4 写 0000B3B2[31:0] 在 0x8 写 0000B5B4[31:0] 在 0xC 写 0000B7B6[31:0]
	16	8	在 0x0 读 B1B0[15:0] 在 0x2 读 B3B2[15:0] 在 0x4 读 B5B4[15:0] 在 0x6 读 B7B6[15:0]	在 0x0 写 B0[7:0] 在 0x1 写 B2[7:0] 在 0x2 写 B4[7:0] 在 0x3 写 B6[7:0]
源端传输宽度 大于 目标传输宽度 地址步长为传输 宽度÷8 目标数据不足 bit 位截断	32	8	在 0x0 读 B3B2B1B0[31:0] 在 0x4 读 B7B6B5B4[31:0] 在 0x8 读 BBBAB9B8[31:0] 在 0xC 读 BFBEBDBC[31:0]	在 0x0 写 B0[7:0] 在 0x1 写 B4[7:0] 在 0x2 写 B8[7:0] 在 0x3 写 BC[7:0]

组合类型	传输宽度		传输操作	
	源端	目标	源 (地址   数据)	目标 (地址   数据)
	32	16	在 0x0 读 B3B2B1B0[31:0] 在 0x4 读 B7B6B5B4[31:0] 在 0x8 读 BBBAB9B8[31:0] 在 0xC 读 BFBEBDBC[31:0]	在 0x0 写 B1B0[15:0] 在 0x2 写 B5B4[15:0] 在 0x4 写 B9B8[15:0] 在 0x6 写 BDBC[15:0]

#### 8.6.4.1 操作一个不支持字节或半字写的 AHB 设备

AHB 总线传输通过 HSIZE 表示传输数据的宽度，目标设备不支持字节/半字操作意味着对应从设置没有处理 HSIZE 的逻辑，目标设备统一认为传输数据为一个字 32bit。

AHB 设备通常都支持字（32bit）操作，而当有的 AHB 设备不支持字节（8bit）或者半字（16bit）写操作时，DMA 会将数据处理扩展为字（32bit）。同时对应目标地址步长应变改与 32bit 对应的 4，目标数据宽度应配置为 32bit。

举例说明，存储器配置为数据源，传输宽度为 8bit，待传输数据为 0xDA，我们希望能够传输到目标设备地址的 0x2 上面，由于对应从设备不支持 8bit/16bit 操作，意味着从设备会认为自己接收的是 32bit 的 0x0000\_00DA，并且写 32bit 的 0x0000\_00DA 到 0x0 地址上（因为是 32bit 操作，地址单位为 0x4），也就是说 0x2 的地址对应的数据为 0x00。

因此需要对数据进行复制扩展操作，0xDA 会被复制扩展为 4 个 8bit 组合为一个 32bit 数据 0xDADA\_DADA，目标地址的 0x2，从设备接收到 32bit 的 0xDADA\_DADA，会写 32bit 数据 0xDADA\_DADA 到 0x0 地址上。这样目的地址 0x2 上的数据即为想要的 0xDA。

这种方法确实会写冗余的数据到目标寄存器上，但是可以保证写入目的地址上的数据是程序想要的，不会发生错误。

源数据宽度 8bit，会被复制扩充为  $4 \times 8\text{bit} = 32\text{bit}$  数据，如 0x12 会被扩充为 0x1212\_1212。

源数据宽度 16bit，会被复制扩充为  $2 \times 16\text{bit} = 32\text{bit}$  数据，如 0x1234 会被扩充为 0x1234\_1234。

只支持 32bit 传输，不支持 8bit/16bit 的从设备数据宽度应配为 32bit。

#### 8.6.5 错误管理

地址空间会存在不允许被访问的保留区域，DMA 传输地址自动递增或者指定地址时有可能会访问到这些保留地址区域。DMA 传输错误标志（TEIF）会在 DMA 操作一个保留的地址空间时置‘1’，同时该 DMA 通道对应的使能位会被硬件清零，以停止该通道上的错误传输。此时，在 DMA\_IFT 寄存器中对该通道的传输错误中断标志位（TEIF）将被置位。若想产生中断，需配置 DMA\_CCRx 寄存器中对应的传输错误中断使能位。

## 8.7 DMA 寄存器描述

表 8-4 DMA 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	DMA_ISR	DMA 中断状态寄存器	0x00000000
0x04	DMA_IFCR	DMA 中断标志清除寄存器	0x00000000
0x08+20×(n-1)	DMA_CCRx	DMA 通道 x 配置寄存器	0x00000000
0x0C+20×(n-1)	DMA_CNDTRx	DMA 通道 x 传输数量寄存器	0x00000000
0x10+20×(n - 1)	DMA_CPARx	DMA 通道 x 外设地址寄存器	0x00000000
0x14+20×(n - 1)	DMA_CMARx	DMA 通道 x 存储器地址寄存器	0x00000000

### 8.7.1 DMA 中断状态寄存器(DMA\_ISR)

偏移地址: 0x00

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24
Field	Reserved				TEIF7	HTIF7	TCIF7	GIF7
Type					r	r	r	r
Bit	23	22	21	20	19	18	17	16
Field	TEIF6	HTIF6	TCIF6	GIF6	TEIF5	HTIF5	TCIF5	GIF5
Type	r	r	r	r	r	r	r	r
Bit	15	14	13	12	11	10	9	8
Field	TEIF4	HTIF4	TCIF4	GIF4	TEIF3	HTIF3	TCIF3	GIF3
Type	r	r	r	r	r	r	r	r
Bit	7	6	5	4	3	2	1	0
Field	TEIF2	HTIF2	TCIF2	GIF2	TEIF1	HTIF1	TCIF1	GIF1
Type	r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31:28	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
27,23,19,15,11,7,3	TEIFx	r	0x00	<p>通道 x 的传输错误标志 (x=1~7) (Channel x transfer error flag)</p> <p>该位软件只读, 由硬件写 1 或清 0。置 '1' DMA_IFCR 寄存器对应位, 可以清 0 此标志位。</p> <p>0: 对应通道 x 的 DMA 传输正常 (TE) 1: 对应通道 x 的 DMA 访问保留地址, 传输错误 (TE)</p>
26,22,18,14,10,6,2	HTIFx	r	0x00	<p>通道 x 的半传输标志 (x=1~7) (Channel x half transfer flag)</p> <p>该位软件只读, 由硬件写 1 或清 0。置 '1' DMA_IFCR 寄存器对应位, 可以清 0 此标志位。</p> <p>0: 对应通道 x 的 DMA 传输未到一半 (HT) 1: 对应通道 x 的 DMA 传输已到一半 (HT)</p>
25,21,17,13,9,5,1	TCIFx	r	0x00	<p>通道 x 的传输完成标志 (x=1~7) (Channel x transfer complete flag)</p> <p>该位软件只读, 由硬件写 1 或清 0。置 '1' DMA_IFCR 寄存器对应位, 可以清 0 此标志位。</p> <p>0: 对应通道 x 的 DMA 传输未完成 (TC) 1: 对应通道 x 的 DMA 传输完毕 (TC)</p>
24,20,16,12,8,4,0	GIFx	r	0x00	<p>通道 x 的全局中断标志 (x=1~7) (Channel x global interrupt flag)</p> <p>该位软件只读, 由硬件写 1 或清 0。置 '1' DMA_IFCR 寄存器对应位, 可以清 0 此标志位。</p> <p>0: 对应通道 x 上 TE、HT、TC 事件都没有产生 1: 对应通道 x 有 TE、HT、TC 事件中的任一事件产生</p>

### 8.7.2 DMA 中断标志清除寄存器(DMA\_IFCR)

偏移地址: 0x04

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	
Field	Reserved					CTEIF7	CHTIF7	CTCIF7	CGIF7
Type						w	w	w	w
Bit	23	22	21	20	19	18	17	16	
Field	CTEIF6	CHTIF6	CTCIF6	CGIF6	CTEIF5	CHTIF5	CTCIF5	CGIF5	

Type	w	w	w	w	w	w	w	w
Bit	15	14	13	12	11	10	9	8
Field	CTEIF4	CHTIF4	CTCIF4	CGIF4	CTEIF3	CHTIF3	CTCIF3	CGIF3
Type	w	w	w	w	w	w	w	w
Bit	7	6	5	4	3	2	1	0
Field	CTEIF2	CHTIF2	CTCIF2	CGIF2	CTEIF1	CHTIF1	CTCIF1	CGIF1
Type	w	w	w	w	w	w	w	w

Bit	Field	Type	Reset	Description
31:28	Reserved			保留, 始终读为 0
27,23,19,15,11,7,3	CTEIFx	w	0x00	<p>清除通道 x 的传输错误标志 (x=1~7) (Channel x transfer error clear)</p> <p>该位由软件置‘1’或清零。</p> <p>0:无效</p> <p>1:清‘0’DMA_ISR 寄存器中的对应 TEIF 标志</p>
26,22,18,14,10,6,2	CHTIFx	w	0x00	<p>清除通道 x 的半传输标志 (x=1~7) (Channel x half transfer clear)</p> <p>该位由软件置‘1’或清零。</p> <p>0:无效</p> <p>1:清‘0’DMA_ISR 寄存器中的对应 HTIF 标志</p>
25,21,17,13,9,5,1	CTCIFx	w	0x00	<p>清除通道 x 的传输完成标志 (x=1~7) (Channel x transfer complete clear)</p> <p>该位由软件置‘1’或清零。</p> <p>0:无效</p> <p>1:清‘0’DMA_ISR 寄存器中的对应 TCIF 标志</p>
24,20,16,12,8,4,0	CGIFx	w	0x00	<p>清除通道 x 的全局中断标志 (x=1~7) (Channel x global interrupt clear)</p> <p>该位由软件置‘1’或清零。</p> <p>0:无效</p> <p>1:清‘0’DMA_ISR 寄存器中的对应的 GIF、TEIF、HTIF 和 TCIF 标志</p>

### 8.7.3 DMA 通道 x 配置寄存器(DMA\_CCRx)(x=1~7)

偏移地址: 0x08+20x(通道编号-1)

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24
Field	Reserved							
Type								
Bit	23	22	21	20	19	18	17	16
Field	Reserved							
Type								
Bit	15	14	13	12	11	10	9	8
Field	ARE	MEM2MEM	PL	MSIZE		PSIZE		
Type	rw	rw	rw	rw		rw		
Bit	7	6	5	4	3	2	1	0
Field	MINC	PINC	CIRC	DIR	TEIE	HTIE	TCIE	EN
Type	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0
15	ARE	rw	0x00	自动重装载 (Auto reload) 该位由软件置 '1' 或清零。 1:使能自动重装载传输数量 0:禁止自动重装载传输数量
14	MEM2MEM	rw	0x00	存储器到存储器模式 (Memory to memory mode) 该位由软件置 '1' 或清零。 0:关闭存储器到存储器模式 1:使能存储器到存储器模式

Bit	Field	Type	Reset	Description
13:12	PL	rw	0x00	<p>通道优先级 (Channel priority level)</p> <p>该位由软件置 ‘1’ 或清零。</p> <p>00:低 01:中 10:高 11:最高</p>
11:10	MSIZE	rw	0x00	<p>存储器数据宽度 (Memory size)</p> <p>该位由软件置 ‘1’ 或清零。</p> <p>00:8 bit 01:16 bit 10:32 bit 11:保留, 未定义</p>
9:8	PSIZE	rw	0x00	<p>外设数据宽度 (Peripheral size)</p> <p>该位由软件置 ‘1’ 或清零。</p> <p>00:8 bit 01:16 bit 10:32 bit 11:保留, 未定义</p>
7	MINC	rw	0x00	<p>存储器地址递增模式 (Memory increment mode)</p> <p>该位由软件置 ‘1’ 或清零。</p> <p>0:关闭存储器地址递增操作 1:使能存储器地址递增操作</p>
6	PINC	rw	0x00	<p>外设地址递增模式 (Peripheral increment mode)</p> <p>该位由软件置 ‘1’ 或清零。</p> <p>0:关闭外设地址递增操作 1:使能外设地址递增操作</p>
5	CIRC	rw	0x00	<p>循环模式 (Circular mode)</p> <p>该位由软件置 ‘1’ 或清零。</p> <p>0:关闭循环操作 1:使能循环操作</p>

Bit	Field	Type	Reset	Description
4	DIR	rw	0x00	<p>数据传输方向 (Date transfer direction)</p> <p>该位由软件置 ‘1’ 或清零。</p> <p>0:从外设读</p> <p>1:从存储器读</p>
3	TEIE	rw	0x00	<p>传输错误中断使能 (Transfer error interrupt enable)</p> <p>该位由软件置 ‘1’ 或清零。</p> <p>0:关闭 TE 中断</p> <p>1:使能 TE 中断</p>
2	HTIE	rw	0x00	<p>半传输中断使能 (Half transfer interrupt enable)</p> <p>该位由软件置 ‘1’ 或清零。</p> <p>0:关闭 HT 中断</p> <p>1:使能 HT 中断</p>
1	TCIE	rw	0x00	<p>传输完成中断使能 (Transfer complete interrupt enable)</p> <p>该位由软件置 ‘1’ 或清零。</p> <p>0:关闭 TC 中断</p> <p>1:使能 TC 中断</p>
0	EN	rw	0x00	<p>通道使能 (Channel enable)</p> <p>该位由软件置 ‘1’ 或清零。</p> <p>0:通道关闭</p> <p>1:通道使能</p>

#### 8.7.4 DMA 通道 x 传输数量寄存器(DMA\_CNDTRx)(x=1~7)

偏移地址: 0x0C+20x(通道编号-1)

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0
15:0	NDT	rw	0x0000	<p>数据传输数量 (Number of data to transfer)            数据传输数量为 0~65535。这个寄存器只能在通道关闭 (DMA_CCRx 的 EN=0) 时写入。通道使能后该寄存器软件变为不可写只读, 表示剩余多少次 DMA 传输。每次 DMA 传输后, 该寄存器数值递减。寄存器数值递减为 0, 表示数据全部传输完毕。此时若通道配置为自动重加载模式时, 寄存器的内容将被自动重新加载为之前配置时的数值。</p> <p>与通道是否使能无关, 只要该寄存器为 0, DMA 就不会传输数据。</p>

### 8.7.5 DMA 通道 x 外设地址寄存器(DMA\_CPARx)(x=1~7)

偏移地址: 0x10+20x(通道编号-1)

复位值: 0x0000 0000

当使能通道(DMA\_CCRx 的 EN=1) 时不能写该寄存器。

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

Bit	Field	Type	Reset	Description
31:0	PA	rw	0x0000 0000	<p>外设地址 (Peripheral address)            外设数据寄存器的基地址, 作为数据传输的源或目标。            当 PSIZE= '01' (16 位), 地址基本单位为 0x2, 最低位 PA[0] 不必使用。操作自动地与半字地址对齐。            当 PSIZE= '10' (32 位), 地址基本单位为 0x4, 倒数 2 位 PA[1:0]不必使用。操作自动地与字地址对齐。</p>

### 8.7.6 DMA 通道 x 存储器地址寄存器(DMA\_CMARx)(x = 1~7)

偏移地址: 0x14+20x(通道编号-1)

复位值: 0x0000 0000

当开启通道(DMA\_CCRx 的 EN=1) 时不能写该寄存器。

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

Bit	Field	Type	Reset	Description
31:0	MA	rw	0x0000 0000	<p>存储器地址 (Memory address)</p> <p>存储器地址作为数据传输的源或目标。</p> <p>当 MSIZE= '01' (16 位), 地址基本单位为 0x2, 最低位 MA[0] 不必使用。操作自动地与半字地址对齐。</p> <p>当 MSIZE= '10' (32 位), 地址基本单位为 0x4, 倒数 2 位 MA[1:0]不必使用。操作自动地与字地址对齐。</p>

## 9 备份寄存器(BKP)

### 9.1 BKP 简介

备份寄存器含有 20 个 16 位的寄存器, 用户端可用于存储 40 个字节的应用程序数据。当发生电源复位或者系统复位, 系统待机模式下唤醒时, BKP\_DPx 不会复位。

BKP 内部含有一组控制寄存器, 可用于侵入事件检测和 RTC 时钟校准。复位后, 禁止访问备份寄存器与 RTC, 保护备份域以防止可能存在的意外的写操作。可以通过如下的两步打开对备份区域以及 RTC 的访问权限。

配置寄存器 RCC\_APB1ENR 的 PWREN 位为 1, 打开电源时钟;

配置电源控制寄存器 (PWR\_CR) 的 DBP 位为 1, 使能对后备寄存器和 RTC 的访问;

### 9.2 主要特征

- 1) 40 字节数据备份寄存器;
- 2) 存储 RTC 校验值的校验寄存器;
- 3) 具有中断使能与管理侵入检测的状态与控制寄存器;
- 4) 当 PC13 管脚不用于侵入检测时, 用于输出 RTC 校准时钟, RTC 闹钟脉冲或者秒脉冲;

## 9.3 功能描述

### 9.3.1 时钟校准

RTC 时钟经 64 分频输出到侵入检测引脚 TAMPER 上，有利于测量。如果配置 RTC 校验寄存器（BKP\_RTCCR）中的 CCO 位为 1，将会开启 RTC 校准功能。

配置 CAL[6: 0]位为不同值用于减慢时钟，最多可以减慢 121ppm。

### 9.3.2 侵入检测

上升沿侵入检测，配置 BKP\_CR 的 TPE 位等于 1，BKP\_CR 的 TPAL 位等于 0，当 TAMPER 引脚出现 0 到 1 的电平翻转，会产生侵入检测事件，此时，会清除备份寄存器中的所有数据。

下降沿侵入检测，配置 BKP\_CR 的 TPE 位等于 1，BKP\_CR 的 TPAL 位等于 1，当 TAMPER 引脚出现 1 到 0 的电平翻转，会产生侵入检测事件，此时，会清除备份寄存器中的所有数据。

检测到侵入事件，同时配置 BKP\_CSR 寄存器的 TPIE 位为 1，产生中断输出。

当正确处理侵入事件后，PC13 管脚的侵入功能应该关闭，如果需要写数据到备份控制寄存器，重新配置 BKP\_CR 的 TPE 位等于 1 开启侵入功能，这样能够避免侵入检测引脚上出现侵入事件时，对备份寄存器进行写操作。

注：当 VDD 掉电时，侵入检测功能仍然开启。为了避免不必要的复位备份寄存器中的数据，保证 TAMPER 引脚连接到正确的电平。

## 9.4 寄存器描述

表 9-1 BKP 寄存器概览

Offset	Acronym	Register Name	Reset
0x50+4x(n-1)	BKP_DRn	备份数据寄存器 x	0x00000000
0x40	BKP_RTCCR	RTC 时钟校准寄存器	0x00000000
0x44	BKP_CR	备份控制寄存器	0x00000000
0x48	BKP_CSR	备份控制 I 状态寄存器	0x00000000

### 9.4.1 备份数据寄存器 n(BKP\_DRn)(n = 1 .. 10)

偏移地址：0x50 + 4 × (备份数据寄存器编号 - 1)

复位值：0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	BKP															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	BKP	rw	0x00	<p>BKP：备份数据 存放用户数据。</p> <p>注：当发生电源复位或者系统复位，系统待机模式下唤醒时，BKP_DPX 不会复位，可以由备份域复位或侵入引脚复位（当侵入引脚功能开启时）</p>

#### 9.4.2 时钟校准寄存器 (BKP\_RTCCR)

偏移地址: 0x40

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.						ASOS	ASOE	CCO	CAL						
Type							rw	rw	rw	rw						

Bit	Field	Type	Reset	Description
15:10	Reserved			保留，始终读为 0。
9	ASOS	rw	0x00	<p>ASOS: 输出闹钟或秒脉冲 (Output alarm clock or second pulse) 配置 ASOE 位为 1, ASOS 位用于选择在 TAMPER 引脚上输出的输出信号。 0: 输出 RTC 闹钟脉冲 1: 输出秒脉冲</p>
8	ASOE	rw	0x00	<p>ASOE: 使能输出闹钟或秒脉冲 (Alarm or second output enable) 根据 ASOS 位的设置，ASOE 位用于使能闹钟或秒脉冲输出到 TAMPER 引脚。 注：输出脉冲的宽度为一个 RTC 时钟的周期。设置了 ASOE 位时不能开启 TAMPER 的功能。只能由后备区域复位清除该位。</p>

7	CCO	rw	0x00	<p>CCO: 校准时钟输出 (Calibration clock output)</p> <p>0: 无影响</p> <p>1: 配置 CCO 位为 1, 在侵入检测引脚输出 64 分频后的 RTC 时钟</p> <p>注: 当 VDD 掉电时, 清除该位。当 CCO 位置 1 时, 必须关闭侵入检测功能以避免检测到无用的侵入信号。</p>
6:0	CAL	rw	0x00	<p>CAL: 校准值 (Calibration value)</p> <p>校准值表示时钟脉冲跳过数量 (每 <math>2^{20}</math> 个时钟脉冲)。用于对 RTC 进行校准, 以 10000001 (<math>2^{20}</math>) ppm 的比例减慢时钟。</p> <p>注: RTC 时钟可以被减慢 0~121ppm。</p>

#### 9.4.3 备份控制寄存器 (BKP\_CR)

偏移地址: 0x44

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.														TPAL	TPE
Type															rw	rw

Bit	Field	Type	Reset	Description
15:2	Reserved			保留, 始终读为 0。
1	TPAL	rw	0x0	<p>TPAL: 侵入检测引脚(TAMPER)0 有效电平(TAMPER pin active level)</p> <p>0: 如果 TPE 位为 1, 侵入检测 TAMPER 引脚上的高电平会清除备份寄存器所有数据</p> <p>1: 如果 TPE 位为 1, 侵入检测 TAMPER 引脚上的低电平会清除备份寄存器所有数据</p>

				TPE: 侵入检测 (TAMPER) 引脚使能 (TAMPER pin enable) 0: TAMPER 引脚作为普通 IO 1: 开启侵入检测引脚
0	TPE	rw	0x0	

#### 9.4.4 备份控制 I 状态寄存器 (BKP\_CSR)

偏移地址: 0x48

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Res.								TIF	TEF	Res.				TPIE	CTI	CTE
Type									r	r					rw	w	w

Bit	Field	Type	Reset	Description
15:9	Reserved		0x0000	保留, 始终读为 0。
8	TIF	r	0x00	<p>TIF: 侵入中断标志 (Tamper interrupt flag)</p> <p>当 TPIE 位为 1 并且检测到侵入事件时, 该位由硬件置'1'。向 CTI 位写'1'来清除此标志位与中断 (如果 TPIE 位被清除, 则此位也会被清除)。</p> <p>0: 无侵入中断 1: 产生侵入中断</p>
7	TEF	r	0x00	<p>TEF: 侵入事件标志 (Tamper event flag)</p> <p>当发生侵入事件时, 此位由硬件置'1'。向 CTE 位写'1'可清除该标志位。</p> <p>0: 无侵入事件 1: 检测到侵入事件</p>
6:3	Reserved			保留, 始终读为 0。

Bit	Field	Type	Reset	Description
2	TPIE	rw	0x00	<p>TPIE: 使能侵入 TAMPER 引脚中断 (TAMPER pin interrupt enable)</p> <p>0: 禁止侵入检测中断</p> <p>1: 允许侵入检测中断 (必须配置 BKP_CR 寄存器的 TPE 位为'1')</p> <p>注 1: 侵入中断无法唤醒处于低功耗模式的内核。</p> <p>注 2: 只有系统复位或由待机模式唤醒后才复位该位。</p>
1	CTI	w	0x00	<p>CTI: 清除侵入检测中断 (Clear tamper interrupt )</p> <p>该位只能写入, 读出值为 0。</p> <p>0: 无效</p> <p>1: 清除侵入检测中断和侵入检测中断标志 (TIF)</p>
0	CTE	w	0x00	<p>CTE: 清除侵入检测事件标志 (Clear tamper event flag),</p> <p>该位只能写入, 读出值为 0。</p> <p>0: 无效</p> <p>1: 清除侵入检测事件标志 (TEF), 同时复位侵入检测器</p>

## 10 时钟回馈系统 (CRS)

### 10.1 CRS 简介

作为一种先进的数字控制器, 时钟回馈系统 (CRS) 用于校准内部高速时钟 (HSI8)。CRS 将实际 HSI 时钟与参考的可选同步源进行比较, 可以计算测出 HSI 时钟的实际频率。同时, 可根据测得的实际频率与理想频率的差值对 HSI 时钟进行微调, CRS 可配置自动微调或手动微调。

USB 外设可通过 CRS 得到精准的时钟。

同步信号根据来源分类为三种:

- 1) 直接来自 GPIO 的外部管脚的同步源
- 2) 来自 USB 外设的 SOF 包, 相邻 SOF 包间隔时长由 USB 主机控制为 1ms
- 3) 来自用户软件配置产生的同步脉冲

## 10.2 CRS 功能框图

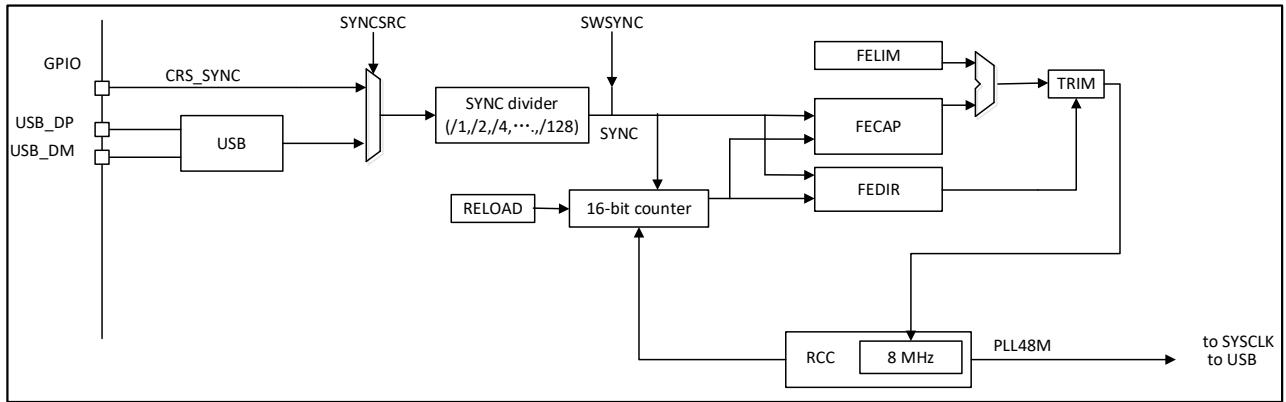


图 10-1 CRS 功能框图

## 10.3 CRS 主要特征

- 1) 多种同步源供选择（包括可配置分频系数的预分频器以及信号极性选择）
  - a) 外部引脚
  - b) USB SOF 包接收

同步脉冲也可以通过软件配置产生

硬件自动校准功能，无需每次软件根据实际差值调整微调值

可通过手动进行相关配置使得频率更快调整到期望值

带有自动误差捕获及数据加载的 16 位频率误差计数器

频率误差分析并产生状态标志的界限值可编程

中断/事件，可通过配置屏蔽

  - a) 期望的 0 误差同步事件（**ESYNC**）
  - b) 同步时误差可接受（**SYNCOK**）
  - c) 同步时误差较大（**SYNCWARN**）
  - d) 同步时误差超出调整范围、同步信号丢失、微调值溢出（**ERR**）

## 10.4 中断

表 10-1 中断控制位

中断事件	事件标志	使能控制位	清除控制位
期望的 0 误差同步事件	ESYNCNF	ESYNCIE	ESYNCIE
同步时误差可接受	SYNCOKF	SYNCOKIE	SYNCOK
同步时误差较大	SYNCWARNF	SYNCWARNIE	SYNCWARNC
同步错误或校准错误 (TRIMOVF, SYNCMISS, SYNCERR)	ERRF	ERRIE	ERRC

## 10.5 CRS 功能描述

### 10.5.1 同步输入

通过 CRS\_CFGR 寄存器配置的 CRS 同步源可以来自外部 CRS\_SYNC 引脚或 USB SOF 包。为满足同步输入源更好的稳健性，一个简单的通过 HSI48 时钟采样的 2 级数字滤波是必不可少的，它可以过滤很多小错误。

信号源拥有一个可配置的极性，且可通过一个可编程的预分频器分频，以保证其处于一个合适的频率范围内（通常为 1KHz）。

CRS 同步源配置方法详见小节：CRS 配置寄存器（CRS\_CFGR）。

### 10.5.2 频率误差测量

频率误差计数器是一个 16 位计数器，可被硬件控制向下或者向上计数。每次同步事件触发时，会使得 RELOAD 值被重新载入该计数器，进行重新计数。在重载后，计数器开始向下重新计数。计数器递减为 0 时，对应 ESYNC 标志位置起。之后计数器自动转换计数方向为向上计数。

在计数器计数的过程中，接收到 SYNC 事件会立即停止计数器计数。此时计数器的值会保存在 CRS\_ISR 的 FECAP 位。此时计数器的计数方向会保存在 CRS\_ISR 的 FEDIR 位。

根据计数器因 SYNC 事件停止时的计数方向，可以定性推测实际时钟频率与期望值的误差：

停止时 FEDIR=0，即向上计数，实际频率大于期望频率；

停止时 FEDIR=1，即向下计数，实际频率小于或等于期望频率。

根据计数器因 SYNC 事件停止时的计数值与界限（界限值可配置，为 CRS\_CFGR 寄存器的 FELIM 位）之间的差值，可以得到微调时的步长：

停止时 FECAP<FELIM，可接受范围，不用微调

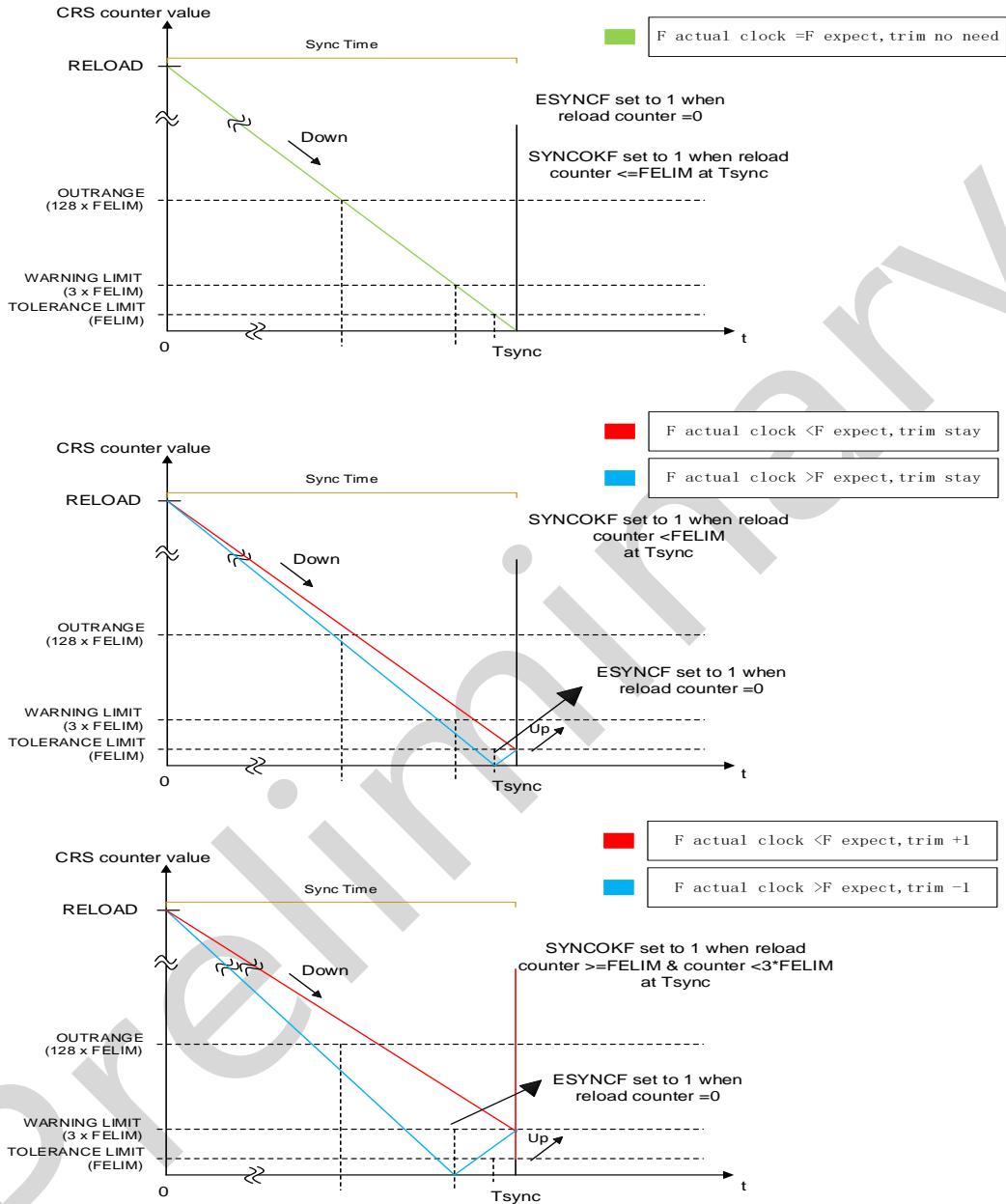
停止时 FELIM<=FECAP<3\*FELIM，误差稍大，微调步长建议为 1

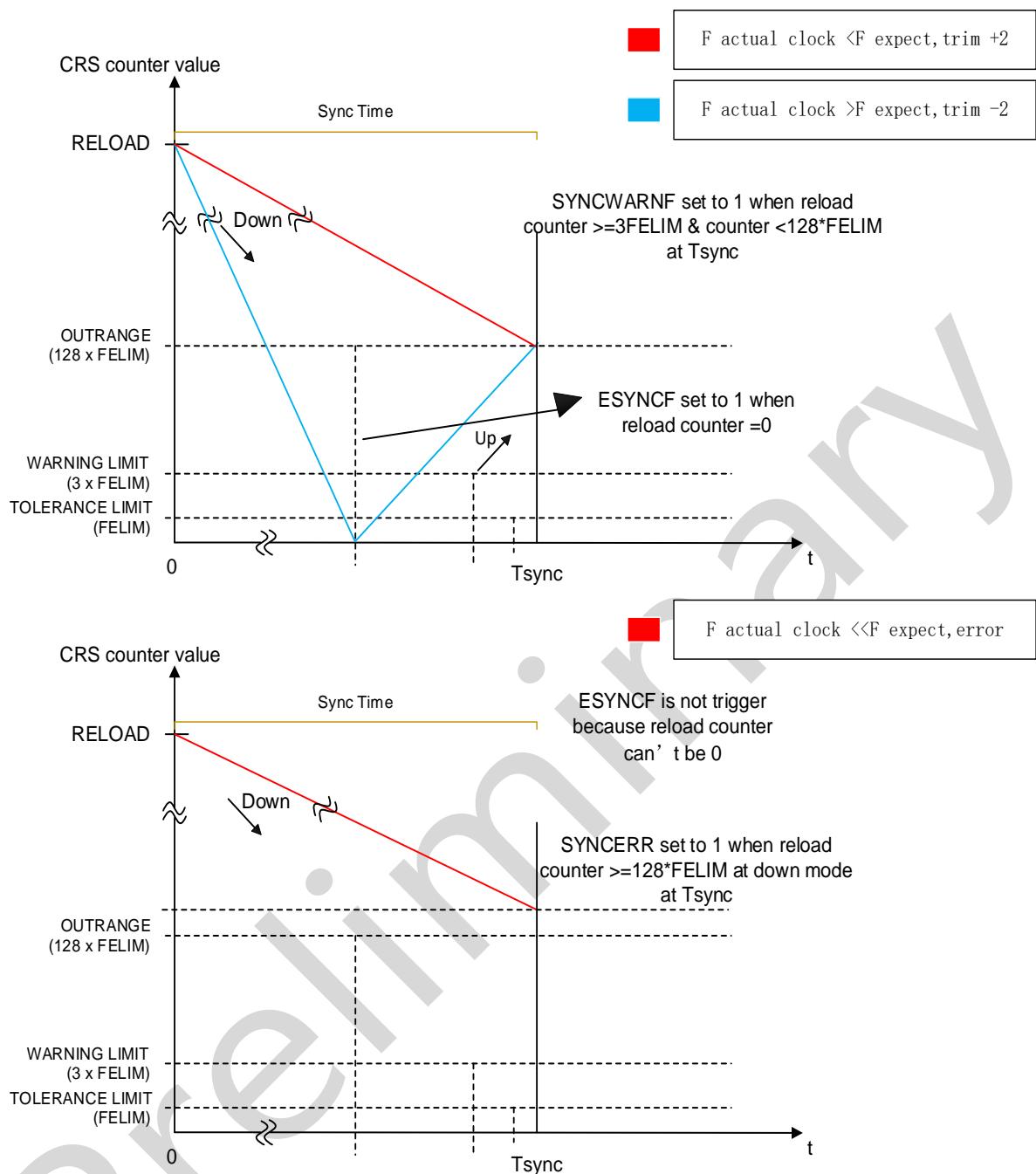
停止时 3\*FELIM<=FECAP<128\*FELIM，误差较大，微调步长建议为 2

停止时 128\*FELIM<=FECAP，误差过大，超出微调范围，微调无意义

在计数器计数的过程中，一直未接收到 SYNC 事件会一直进行计数，直到计数到设定的最大界限值（最大界限值=128\*FELIM），同时标志位 SYNCMISS 置起，表示未接收到 SYNC 事件。

下图列举了计数器测量时可能发生的各种情况。





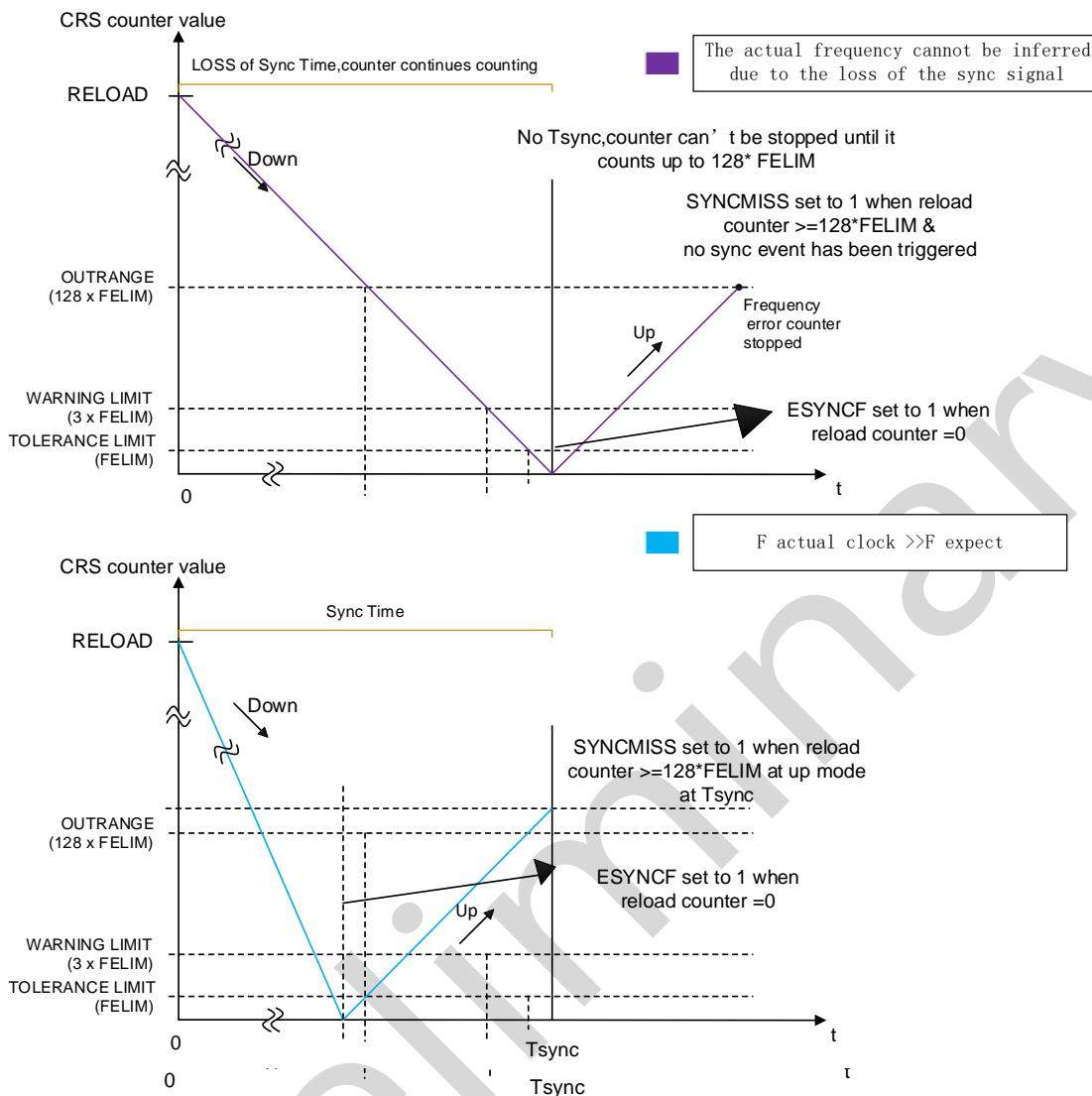


图 10-2 CRS 计数器状态图

### 10.5.3 频率误差计算及自动校准

通过比较频率误差值与定义的界限区间，可以得到实际频率与期望频率的相对关系：

定义  $1 \times FELIM$  为容忍界限 (TOLERANCE LIMIT)

定义  $3 \times FELIM$  为警告界限 (WARNING LIMIT)

定义  $128 \times FELIM$  为溢出界限 (ERROR LIMIT)

通过比较结果 FECAP 在哪个区间，对应状态标志位会置起。同时也会决定自动校准调整的步长（如果 CRS\_CR 寄存器内的 AUTOTRIM 位被使能）：

- 1) 当频率误差低于容忍界限，表明实际校准值位于理想范围内，不需要修正。
  - a) 同步正确标志位 (SYNCOK) 被置位
  - b) 自动校准 (AUTOTRIM) 模式下，不需要修正 TRIM 值

- 2) 当频率误差低于警告界限且大于或等于容忍界限，表明需要些许校准，调节器只需调整 1 档就能达到理想的 TRIM 值。
  - a) 同步正确标志位 (SYNCOK) 被置位
  - b) 自动校准 (AUTOTRIM) 模式下 TRIM 数值每次调节一个档位
- 3) 当误差计数大于或等于警告界限，但小于错误界限，表明需要较大校准并且下个同步周期将不会达到理想的 TRIM 值。
  - a) 同步警告标志位 (SYNCWARN) 被置位
  - b) 自动校准 (AUTOTRIM) 模式下 TRIM 数值每次调节二个档位
- 4) 当误差计数大于或等于错误界限，表明频率误差超出可校准范围。当同步信号未清除或丢失时，也将发生上述情况（如：USB SOF 包丢失）。
  - a) 同步错误 (SYNCERR) 或同步丢失 (SYNCMISS) 标志位被置位
  - b) 自动校准 (AUTOTRIM) 模式下 TRIM 数值不变

注意：如果实际 TRIM 值已接近边界，继续（自动）校准将导致 TRIM 值溢出，此时 TRIM 值将被锁定在边界，且校准溢出 (TRIMOVF) 标志位被置位。

自动校准 (AUTOTRIM) 模式下（配置 CRS\_CR 寄存器 AUTOTRIM 位），TRIM 位仅由硬件配置且软件只读。

## 10.5.4 CRS 初始化及配置

### 10.5.4.1 加载 (RELOAD) 值

RELOAD 值的选定是根据目标频率和分频后的同步源频率比较得出的。RELOAD 值每个 cycle 会减少一从而希望在零值时刚好发生 SYNC 事件，表明实际频率等于期望频率。公式如下：

$$RELOAD = (f_{TARGET} / f_{SYNC}) - 1$$

重置后的 RELOAD 默认值为一个目标频率为 48MHz 的时钟和同步信号频率为 1KHz（来自 USB 的 SOF 信号）计算得到，即 47999（16 进制为 0xBB7F）。

### 10.5.4.2 FELIM 值

FELIM 的选择需要将 HSI48 的特性以及它的典型的调节档位相结合。理想的数值对应于一半的调整步骤，表示为一系列 HSI48 振荡器的计数。公式如下：

$$FELIM = (f_{TARGET} / f_{SYNC}) \times STEP[\%] / 2$$

为了获得更好的修正反馈，应该总是捕获到修正值最近的整数，如果不希望频繁的调整修正值，可以适当的增加 FELIM 数值。

重置的 FELIM 应该匹配  $(f_{TARGET} / f_{SYNC}) = 48000$  和一个典型的调节档位是 0.14%，计算得到值为 33.6（约等于 34.16 进制为 0x22）。

注意：一个错误的 RELOAD 和 FELIM 配置会导致一个不稳定的递减响应，并且没有硬件的保护。理想的运作模式需要适当的 RELOAD 值的设置（根据同步源的频率设置）而且 RELOAD 的值也

是大于 FELIM 值的 128 倍。

## 10.6 CRS 低功耗模式

表 10-2 CRS 低功耗模式下的影响

模式	描述
睡眠	没有影响。 CRS 的中断服务程序会让设备退出睡眠模式。
停止	CRS 寄存器不能被改写。 CRS 停止操作，直到停止或待机模式退出，同时 HSI48 振荡器备用重启。

## 10.7 CRS 寄存器描述

表 10-3 CRS 寄存器描述概览

Offset	Acronym	Register Name	Reset
0x00	CRS_CR	CRS 控制寄存器	0x00001000
0x04	CRS_CFGR	CRS 配置寄存器	0x2022BB7F
0x08	CRS_ISR	CRS 中断状态寄存器	0x00000000
0x0C	CRS_ICR	CRS 中断标志清除寄存器	0x00000000

### 10.7.1 CRS 控制寄存器(CRS\_CR)

地址偏移: 0x00

复位值: 0x0002\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															TRIM[9:8]
Type																rw
Bit	15:8		7	6	5	4	3	2	1	0						
Field	TRIM [7:0]		SW	AUTO	CEN	Res.	ESYNC	ERRIE	SYNC	OK						IE
Type	rw		rw	rw	rw		rw	rw	rw	rw						rw

Bit	Field	Type	Reset	Description
31:13	Reserved			保留, 始终读为 0
17:8	TRIM[9:0]	rw	0x200	<p>TRIM[9: 0]: (HSI 8MHz trimming)</p> <p>默认值是0x200, 这个值是修正档位的中间值。</p> <p>当RCC_ICSCR的TRIM_CRS_SEL(bit[0])置1时, 写入的值可以重新校正HSI频率, 否则写入的值不起作用。</p> <p>当AUTOTRIMEN被设置, 这个域由硬件控制, 且为只读。</p> <p>全0: 频率最小</p> <p>全 1: 频率最大</p>
7	SWSYNC	rw	0x00	<p>产生软件同步行为 (Generate software SYNC event)</p> <p>这位由软件设置, 产生一个软件 SYNC 事件。由硬件自动清除。</p> <p>0: 无操作</p> <p>1: 软件 SYNC 行为被触发 (生成软件 SYNC 事件)</p>
6	AUTOTRIMEN	rw	0x00	<p>自动校准使能 (Automatic trimming enable)</p> <p>这位根据测量两次同步事件之间的频率误差来计算 TRIM 值。</p> <p>如果 AUTOTRIMEN 位被使能, TRIM 位将被锁定为只读。</p> <p>TRIM 值每次可由硬件调整一个或者两个档位 (取决于测量频率误差值)。</p> <p>0: 禁止自动校准, TRIM 位可被用户修改</p> <p>1: 使能自动校准, TRIM 位只读且由硬件控制</p>
5	CEN	rw	0x00	<p>频率误差计数器使能 (Frequency error counter enable)</p> <p>这个位使能了时钟的频率误差计数器。</p> <p>0: 频率误差计数器禁止</p> <p>1: 频率误差计数器使能</p> <p>当被设置, CRS_CFG 寄存器被写保护且不能被改写。</p>
4	Reserved			保留, 始终读为 0
3	ESYNCIE	rw	0x00	<p>Expected SYNC 中断使能(Expected SYNC interrupt enable)</p> <p>0: Expected SYNC (ESYNCF) 中断禁止</p> <p>1: Expected SYNC (ESYNCF) 中断使能</p>

Bit	Field	Type	Reset	Description
2	ERRIE	rw	0x00	<p>同步及修正错误中断使能 (Synchronization or trimming error interrupt enable)</p> <p>0: 同步及修正错误 (ERRF) 中断禁止 1: 同步及修正错误 (ERRF) 中断使能</p>
1	SYNCWARNIE	rw	0x00	<p>SYNC 警告中断使能 (SYNC warning interrupt enable)</p> <p>0: SYNC 警告 (SYNCWARNF) 中断禁止 1: SYNC 警告 (SYNCWARNF) 中断使能</p>
0	SYNCOKIE	rw	0x00	<p>SYNC 行为 OK 中断使能 (SYNC event OK interrupt enable)</p> <p>0: SYNC 行为 OK (SYNCOKF) 中断禁止 1: SYNC 行为 OK (SYNCOKF) 中断使能</p>

## 10.7.2 CRS 配置寄存器(CRS\_CFGR)

地址偏移: 0x04

复位值: 0x2022\_BB7F

Bit	31	30	29	28	27	26	25	24	23:16
Field	SYNCPOL	Reserved		SYNCSRC [1:0]	Reserved		SYNCDIV [2:0]	FELIM[7:0]	
Type	rw				rw		rw		rw
Bit	15	14	13	12	11	10	9	8	7 6 5 4 3 2 1 0
Field	RELOAD[15:0]								
Type	rw								

Bit	Field	Type	Reset	Description
31	SYNCPOL	rw	0x00	<p>SYNC 极性选择 (SYNC polarity selection)</p> <p>这位由软件设置与清除, 用来选择 SYNC 信号源的输入极性。</p> <p>0: SYNC 上升沿有效 (默认值) 1: SYNC 下降沿有效</p>
30	Reserved			保留, 始终读为 0
29:28	SYNCSRC	rw	0x10	<p>SYNC 信号源选择 (SYNC signal source selection)</p> <p>这些位由软件设置与清除, 用来选择信号源。</p>

Bit	Field	Type	Reset	Description
				00: GPIO 选为 SYNC 的信号源 01: 保留 10: USB SOF 选为 SYNC 的信号源 (默认值) 11: 保留
27	Reserved			保留, 始终读为 0
26:24	SYNCDIV	rw	0x00	SYNC 分频 (SYNC divider) 这些位由软件设置清除, 用来控制 SYNC 信号的分频系数。 000: SYNC 不分频 001: SYNC 2 分频 010: SYNC 4 分频 011: SYNC 8 分频 100: SYNC 16 分频 101: SYNC 32 分频 110: SYNC 64 分频 111: SYNC 128 分频
23:16	FELIM	rw	0x22	频率误差限制 (Frequency error limit) FELIM 包含的数值是用来计算捕获的频率误差数值, 频率误差数值存放在 CRS_ISR 寄存器的 FECAP[15:0]域。
15:0	RELOAD	rw	0xBB7F	计数器加载的数值 (Counter reload value) RELOAD 的值在每一次同步行为后都会重新加载进频率误差寄存器中。

### 10.7.3 CRS 中断状态寄存器(CRS\_ISR)

地址偏移: 0x08

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	FECAP[15:0]															
Type	r															
Bit	15	14:11	10	9	8	7:4	3	2	1	0						
Field	FEDIR	Res.	TRIMOVF	SYNC MISS	SYNC ERR	Res.	ESYN CF	ERRF	SYNC WARN	SYNC OKF						

Type	r		r	r	r		r	r	F	
------	---	--	---	---	---	--	---	---	---	--

Bit	Field	Type	Reset	Description
31:16	FECAP	r	0x00	<p>频率误差捕获 ( Frequency error capture)</p> <p>FECAP 是频率误差计数器的值, 在每次的同步行为结束后被锁存。</p>
15	FEDIR	r	0x00	<p>频率错误方向 (Frequency error direction)</p> <p>FEDIR 是频率误差计数器的计数方向在, 在每次的同步行为后被锁存。表明了实际频率是高于还是低于目标频率。</p> <p>0: 向上计数, 实际频率高于目标频率 1: 向下计数, 实际频率低于目标频率</p>
14:11	Reserved			保留, 始终读为 0
10	TRIMOVF	r	0x00	<p>修正上溢出或者下溢出 (Trimming overflow or underflow)</p> <p>当自动修正要上溢出或者下溢出的时候, 这个标志位被硬件设置。如果 CRS_CR 寄存器的 ERRIE 位被置位, 中断产生。通过软件设置 CRS_ICR 寄存器的 ERRC 位来清除这一位。</p> <p>0: 没有修正错误信号 1: 产生修正错误信号</p>
9	SYNCMISS	r	0x00	<p>SYNC 丢失 (SYNC missed)</p> <p>当频率误差计数器达到 FELIM 的 128 倍且没有任何 SYNC 事件被触发, 意味着一个 SYNC 脉冲丢失或者是频率误差太高 (内部频率太高) 以至于不能通过调整 TRIM 值来调节, 需要采取其他的措施, SYNCERR 被硬件置位。在这种情况下, 频率误差计数器停止工作 (等待下一个 SYNC)。如果 CRS_CR 寄存器的 ERRIE 位被置位, 中断产生。通过软件设置 CRS_ICR 寄存器的 ERRC 位清除。</p> <p>0: 没有 SYNC 丢失错误信号 1: 产生 SYNC 丢失错误信号</p>

Bit	Field	Type	Reset	Description
8	SYNCERR	r	0x00	<p>SYNC 错误 (SYNC error)</p> <p>当同步脉冲输入时，在触发 ESYNC 行为之前，测量到的频率误差大于或者等于 FELIM 的 128 倍，SYNCERR 被硬件置位。这说明频率误差太大（内部频率太低）以至于不能通过调整 TRIM 值来调节，需要采取其他的措施。如果 CRS_CR 寄存器的 ERRIE 位被置位，中断产生。通过软件设置 CRS_ICR 寄存器的 ERRC 位清除。</p> <p>0: 没有 SYNC 错误信号 1: 产生 SYNC 错误信号</p>
7:4	Reserved			保留，始终读为 0
3	ESYNCF	r	0x00	<p>期望 SYNC 标志 (Expected SYNC flag)</p> <p>当误差频率计数器到达 0 值的时候，这个标志位被硬件置位。如果 CRS_CR 寄存器的 ESYNCIE 位被置位，中断产生。通过软件设置 CRS_ICR 寄存器的 ESYNCC 位清除。</p> <p>0: 没有期望的 SYNC 信号 1: 产生期望的 SYNC 信号</p>
2	ERRF	r	0x00	<p>错误标志 (Error flag)</p> <p>当产生同步或者修正错误的时候，这个标志位被硬件置位。是由 TRIMOVF、SYNCMISS 和 SYNCERR 组合逻辑决定的。如果 CRS_CR 寄存器的 ERRIE 位被置位，中断产生。通过软件设置 CRS_ICR 寄存器的 ERRC 位清除了 TRIMOVF、SYNCMISS 和 SYNCERR。</p> <p>0: 没有同步或者修正错误信号 1: 产生同步或者修正错误信号</p>
1	SYNCWARNF	r	0x00	<p>SYNC 警告标志 (SYNC warning flag)</p> <p>当测量到的频率误差大于或者等于 FELIM 的 3 倍，但小于 FELIM 的 128 倍，这个标志位被硬件置位。这意味着这必须两个档位甚至更多的档位才能够修正频率误差。如果 CRS_CR 寄存器的 SYNCWARNIE 位被置位，中断产生。通过软件设置 CRS_ICR 寄存器的 SYNCWARNC 位来清除这一位。</p> <p>0: 没有 SYNC 警告信号 1: 产生 SYNC 警告信号</p>

Bit	Field	Type	Reset	Description
0	SYNCOKF	r	0x00	<p>SYNC 行为 OK 标志 (SYNC event OK flag)</p> <p>当测量到的频率误差小于 FELIM 的 3 倍，这个标志位被硬件置位。这意味着不需要有校准的行为，或者一档的调节就足够修正频率误差。如果 CRS_CR 寄存器的 SYNCOKIE 位被置位，中断产生。通过软件设置 CRS_ICR 寄存器的 SYNCOKC 位来清除这一位。</p> <p>0: 没有 SYNC 行为 OK 信号 1: 产生 SYNC 行为 OK 信号</p>

#### 10.7.4 CRS 中断标志清除寄存器(CRS\_ICR)

地址偏移: 0x0C

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15:4		3		2		1		0							
Field	Reserved		ESYNCC		ERRC		SYNCWARNC		SYNCOKC							
Type			w		w		w		w							

Bit	Field	Type	Reset	Description
31:4	Reserved			保留，始终读为 0
3	ESYNCC	w	0x00	<p>Expected SYNC 清除标志 (Expected SYNC clear flag)</p> <p>写 ‘1’ 清除 CRS_ISR 寄存器的 ESYNCF 标志。</p>
2	ERRC	w	0x00	<p>Error 清除标志 (Error clear flag)</p> <p>写 ‘1’ 清除 CRS_ISR 寄存器的 TRIMOVF, SYNCMISS, SYNCERR 位，同样会清除 ERRF 位。</p>
1	SYNCWARNC	w	0x00	<p>SYNC warning 清除标志 (SYNC warning clear flag)</p> <p>写 ‘1’ 清除 CRS_ISR 寄存器的 SYNCWARNF 标志。</p>
0	SYNCOKC	w	0x00	<p>SYNC OK 清除标志 (SYNC event OK clear flag)</p> <p>写 ‘1’ 清除 CRS_ISR 寄存器的 SYNCOKF 标志。</p>

# 11 通用端口(GPIO)

## 11.1 GPIO 简介

每个通用 I/O 端口都可以通过两个 32 位的控制寄存器 (GPIOx\_CRL/GPIOx CRH) 和两个 32 位的复用控制寄存器 (GPIOx\_AFRL, GPIOx\_AFRH) 配置为 8 种模式：模拟输入，浮空输入，上拉输入，下拉输入，GPIO 推挽输出，GPIO 开漏输出，复用推挽输出和复用开漏输出。

可以自由编程控制每个 I/O 端口，不能半字或字节访问，必须按照 32 位字节访问。GPIO 寄存器组有 GPIOx\_BSRR 和 GPIOx\_BRR，通过这两个寄存器可以独立的控制 GPIOx(x=A,B,C,D) 的每一位为 0 或 1。

## 11.2 GPIO 功能框图

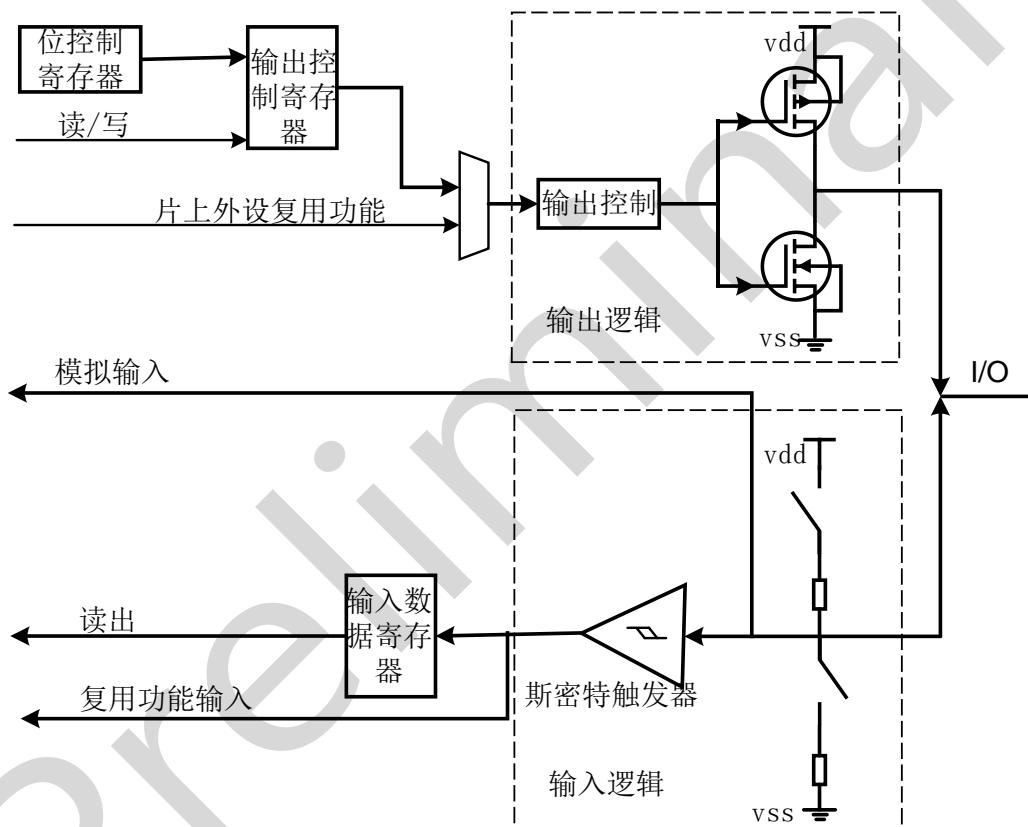


图 11-1 标准 I/O 端口

## 11.3 GPIO 端口配置

表 11-1 端口位配置表

配置模式	CNF1	CNF0	MODE1	MODE0	PxODR 寄存器
通用输出	开漏 (Open-Drain)	0	1	01	0 或 1

配置模式		CNF1	CNF0	MODE1	MODE0	PxODR 寄存器
	推挽 (Push-Pull)		0			0 或 1
复用功能输出	开漏 (Open-Drain)	1	1			不使用
	推挽 (Push-Pull)		0			不使用
输入	浮空输入	0	1	00		不使用
	模拟输入		0			不使用
	上拉输入	1				0
	下拉输入		0			1

输入输出参考配置步骤如下：

1) 通用输入：

用户只需配置 GPIOx\_CRL 中的 CNFx[1:0]选择输入模式

2) 通用输出：

- a) 推挽输出：用户配置 MODE[1:0]选择输出速度，配置 CNFx[1:0]==00；
- b) 开漏输出：用户配置 MODE[1:0]选择输出速度，配置 CNFx[1:0]==01，如果对 pin 上下拉有要求，需要单独配置 GPIOx\_DCR 寄存器，非开漏输出模式，上下拉失效；

3) 复用功能：

配置 AFRLx[3:0]与 AFRHx[3:0]寄存器选择复用功能，

- a) 推挽复用输出：用户配置 MODE[1:0]选择输出速度，配置 CNFx[1:0]==10；
- b) 推挽开漏输出：用户配置 MODE[1:0]选择输出速度，配置 CNFx[1:0]==11；

如果对 IO 上下拉有要求，需要单独配置 GPIOx\_DCR 寄存器，非开漏输出模式，上下拉失效

在复位期间或复位之后，复用功能并未激活，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。

1) PA14:SWCLK 置于下拉模式；

2) PA13:SWDIO 置于上拉模式；

复位后，调试口外的 GPIO 全部置为浮空状态。

配置为输出模式后，写入输出数据寄存器（GPIOx\_ODR）的值会输出到相应的 I/O 引脚。

每个 AHB 时钟周期中，输入数据寄存器（GPIOX\_IDR）捕捉 I/O 引脚上的数据。

每个 GPIO 引脚可以独立控制使能或者关闭内部弱上拉或弱下拉。

## 11.4 主要特征

- 1) 单独的位设置或位清除;

当编程控制 GPIOx\_ODR 的个别位时, 软件不需要禁止中断: 每次 AHB 的写操作, 可以更改 GPIOx\_ODR 对应的一位或多位。

- 2) 外部中断/唤醒线;

每个 IO 端口均可配置中断。当使用外部中断线时, 端口模式配置成输入模式。具体参考外部中断/事件控制器 (EXTI)。

- 3) 支持配置 GPIO 锁定机制;

## 11.5 功能描述

### 11.5.1 复用功能

配置复用功能寄存器打开 IO 对应的复用功能。

- 1) 配置为复用输入功能时, 端口必须配置成输入模式 (浮空、上拉或下拉), 输入管脚驱动由外部控制;
- 2) IO 配置为复用输出功能时, 端口必须配置成复用功能输出模式 (推挽或开漏);
- 3) IO 配置为双向复用功能时, 端口位必须配置复用功能输出模式 (推挽或开漏)。此时, 输入驱动器被配置成浮空输入模式;

当配置端口为复用输出功能时, 端口与片上外设输出信号连接。如果仅仅通过软件方式配置 GPIO 引脚为复用输出功能, 外设没有被激活, 此时输出不确定。

### 11.5.2 GPIO 锁定机制

GPIO 存在锁定机制, 能够保持设定 IO 配置不被改变。当对某一端口执行锁定机制后, 在下一次复位之前, 不能改变端口对应的配置。

### 11.5.3 输入配置

当 I/O 端口配置为输入时:

- 1) 施密特触发输入使能;
- 2) 根据不同的配置 (上拉、下拉, 浮空), 选择弱上拉或弱下拉的电阻连接;
- 3) 出现在 I/O 脚上的数据在每个 AHB 时钟被采样到输入数据寄存器;
- 4) 读访问输入数据寄存器可得到 I/O 状态;
- 5) 输出缓冲被禁用;

下图给出了 I/O 端口位的输入配置:

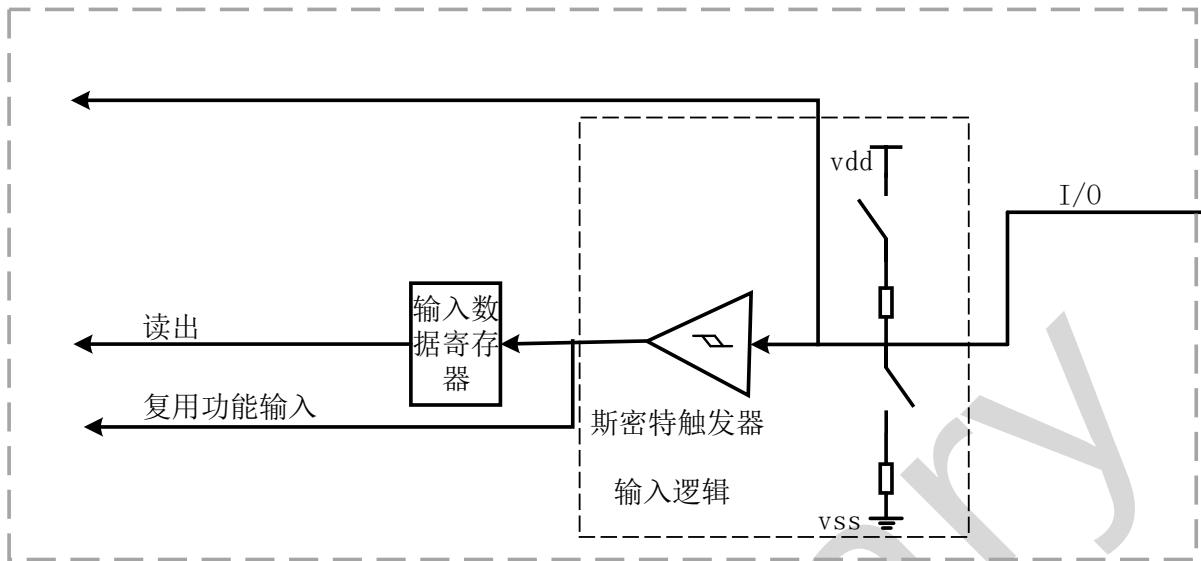


图 11-2 输入浮空/上拉/下拉配置

#### 11.5.4 输出配置

当 GPIO 配置为输出时：

- 1) 施密特触发输入使能；
- 2) 弱上拉和弱下拉电阻被禁用；
- 3) 输出缓冲使能；
- 4) 开漏模式：输出控制器配置为 0 时，对应的引脚输出低电平，输出控制器配置为 1 时，对应的管脚处于高阻态；
- 5) 推挽模式：输出控制器配置为 0 时，对应的引脚输出低电平，输出控制器配置为 1 时，对应的管脚输出高电平；
- 6) 对端口输出控制寄存器读操作，返回上次写入值（在推挽模式时）；
- 7) 对端口输入状态寄存器进行读操作，获得当前 I/O 的状态（开漏模式）；

下图为 I/O 端口位的输出配置：

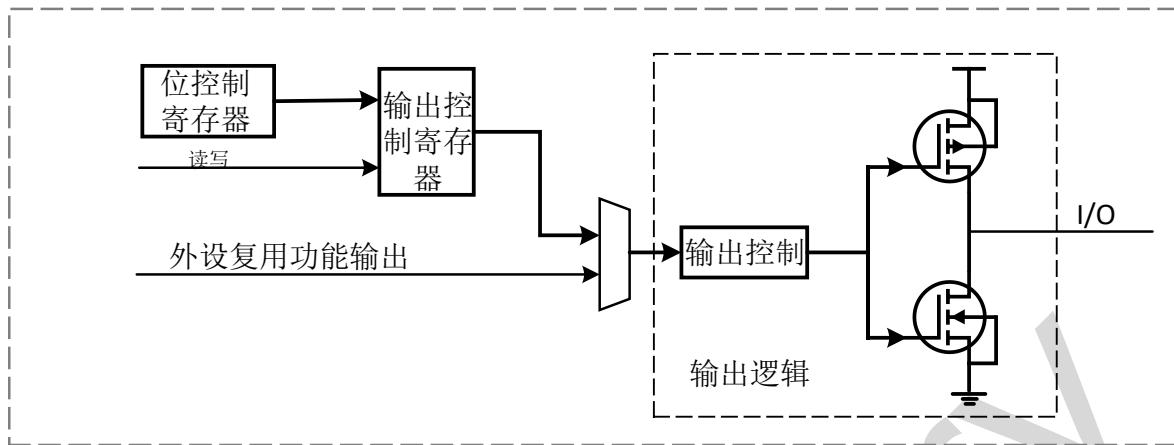


图 11-3 输出配置

### 11.5.5 复用功能配置

当 配置引脚为复用功能时：

- 1) 输出缓冲器由外设驱动
- 2) 配置开漏或推挽时，使能输出缓冲
- 3) 使能施密特触发输入
- 4) 当配置为输入时，可选弱上拉或弱下拉电阻
- 5) I/O 脚上数据在每个 AHB 时钟周期被采样到输入数据寄存器
- 6) 读输入数据寄存器时可得到 I/O 口状态（开漏模式）
- 7) 读输出数据寄存器时可得到最后一次写的值（推挽模式）

下图为 I/O 端口为复用功能的配置。具体见 AFRL 与 AFRH 寄存器描述。

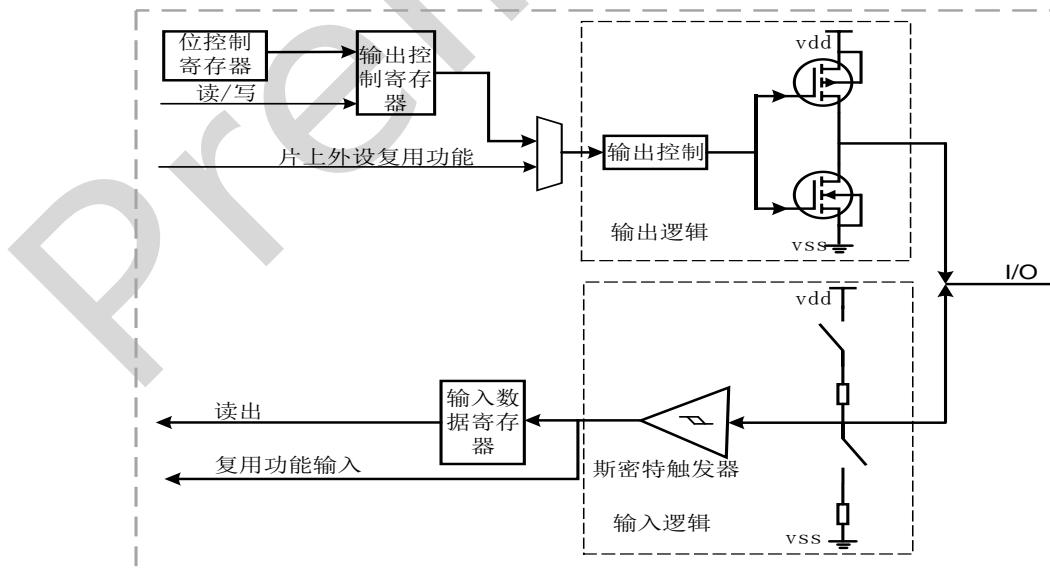


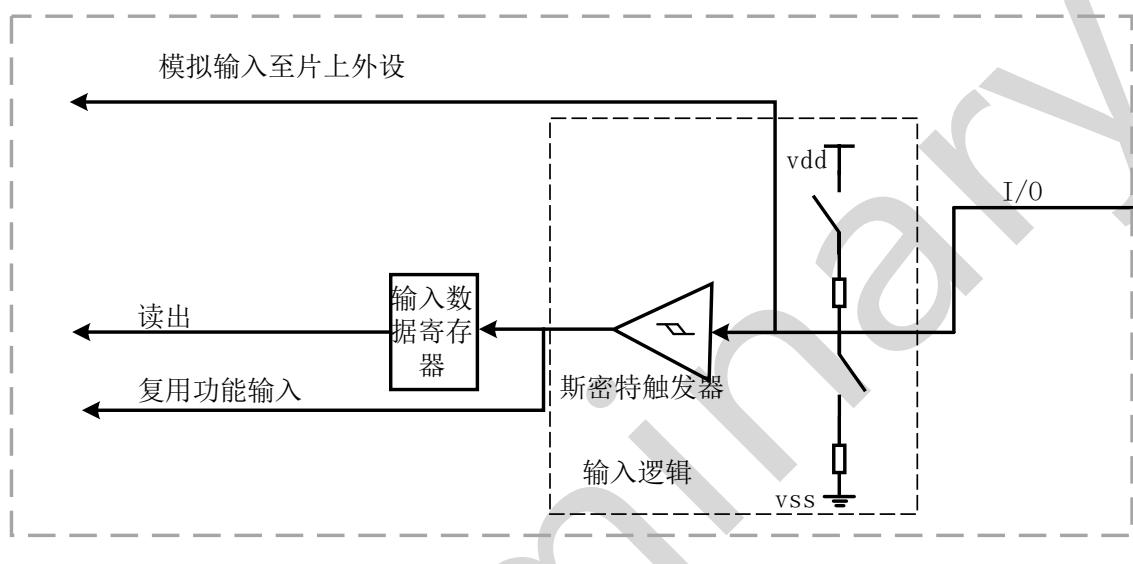
图 11-4 复用功能配置

## 11.5.6 模拟输入配置

当 I/O 端口被配置成模拟输入配置时:

- 1) 输出缓冲器禁用;
- 2) 弱上拉与弱下拉电阻禁用;
- 3) 施密特触发输入禁用;
- 4) 端口输入数据存储器对应位为 0;

下图为 I/O 端口位的高阻抗输入配置



## 11.5.7 外设的 GPIO 配置

下列表格列出了各个外设的引脚配置:

表 11-2 高级定时器 TIM1

TIM1 引脚	配置	GPIO 配置
TIM1_CHx	输入捕获通道 x	浮空输入
	输出比较通道 x	推挽复用输出
TIM1_CHxN	互补输出通道 x	推挽复用输出
TIM1_BKIN	刹车输入	浮空输入
TIM1_ETR	外部触发时钟输入	浮空输入

表 11-3 通用/基本定时器 TIM2/3/14/16/17

TIM2/3/14/16/17	引脚	配置	GPIO 配置
-----------------	----	----	---------

TIM2/3/14/ 16/17_CHx	输入捕获通道 x	浮空输入
	输出比较通道 x	推挽复用输出
TIM2/3_ETR	外部触发时钟输入 x	浮空输入

表 11-4 UART

UART 引脚	配置	GPIO 配置
UARTx_TX	串口发送	推挽复用输出
UARTx_RX	串口接收	浮空输入或带上拉输入
UARTx_RTS	硬件流措控制	推挽复用输出
UARTx_CTS	硬件流措控制	浮空输入或带上拉输入

表 11-5 SPI

SPI 引脚	配置	GPIO 配置
SPIx_SCK	主模式	推挽复用输出
	从模式	浮空输入
SPIx_MOSI	全双工模式/主模式	推挽复用输出
	全双工模式/从模式	浮空输入或带上拉输入
SPIx_MISO	全双工模式/主模式	浮空输入或带上拉输入
	全双工模式/从模式	推挽复用输出
SPIx_NSS	硬件主/从模式	浮空输入或带上拉输入或带下拉输入
	硬件主模式/NSS 输出使能	推挽复用输出
	软件模式	未用,可作为通用 I/O

表 11-6 I2C

I2C 引脚	配置	GPIO 配置
I2Cx_SCL	I2C 时钟	开漏复用输出

I2Cx_SDA	I2C 数据	开漏复用输出
----------	--------	--------

表 11-7 CAN

CAN 引脚	GPIO 配置
CAN_TX	推挽复用输出
CAN_RX	浮空输入或带上拉输入

表 11-8 ADC

ADC 引脚	GPIO 配置
ADC	模拟输入

表 11-9 其他 I/O 引脚

引脚	配置	GPIO 配置
MCO	时钟输出	推挽复用输出
EXTI 输入线	外部中断输入	浮空输入或带上拉输入或下拉输入

### 11.5.8 OSC\_IN/OSC\_OUT 作为 GPIO 端口 PD0/PD1

外部振荡器引脚 OSC\_IN/OSC\_OUT 可以用作 GPIO 的 PD0/PD1 功能，当作为 GPIO 功能时，需先关闭外部高速时钟，再按照正常的 GPIO 功能操作。

注：外部中断/事件功能没有被重映射。

### 11.5.9 SWD 复用功能重映射

SWD 调试接口信号被映射到 GPIO 端口上，如下表所示

复用功能	GPIO 端口
SWDIO	PA13
SWCLK	PA14

为了在调试期间可以使用更多 GPIOs，通过设置复用重映射和复用功能寄存器，可以改变上述重映射配置。

## 11.6 GPIO 寄存器描述

表 11-10 GPIO 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	GPIOx_CRL	端口配置低寄存器	0x44444444

Offset	Acronym	Register Name	Reset
0x04	GPIOx_CRH	端口配置高寄存器	0x44444444
0x08	GPIOx_IDR	端口输入数据寄存器	0x0000XXXX
0x0C	GPIOx_ODR	端口输出数据寄存器	0x00000000
0x10	GPIOx_BSRR	端口设置/清除寄存器	0x00000000
0x14	GPIOx_BRR	端口位清除寄存器	0x00000000
0x18	GPIOx_LCKR	端口配置锁定寄存器	0x00000000
0x1C	GPIOx_DCR	端口输出开漏控制寄存器	0x00000000
0x20	GPIOx_AFRL	端口复用功能低位寄存器	0xFFFFFFFF
0x24	GPIOx_AFRH	端口复用功能高位寄存器	0xFFFFFFFF

### 11.6.1 端口配置低寄存器 (GPIOx\_CRL)(x = A..D)

偏移地址: 0x00

复位值: 0x4444 4444

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	CNF7		MODE7		CNF6		MODE6		CNF5		MODE5		CNF4		MODE4	
Type	rw		rw													
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CNF3		MODE3		CNF2		MODE2		CNF1		MODE1		CNF0		MODE0	
Type	rw		rw													

Bit	Field	Type	Reset	Description
31: 30	CNF7	rw	0x01	端口配置位 (7...0) (Port x configuration bits)
27: 26	CNF6	rw	0x01	配置 MODE==0,端口为输入模式:
23: 22	CNF5	rw	0x01	00: 模拟输入模式
19: 18	CNF4	rw	0x01	01: 浮空输入模式
15: 14	CNF3	rw	0x01	10: 上拉/下拉输入模式
11: 10	CNF2	rw	0x01	11: 保留
7: 6	CNF1	rw	0x01	配置 MODE==1,端口为输出模式
3: 2	CNF0	rw	0x01	00: 通用推挽输出模式
29: 28	MODE7	rw	0x00	01: 通用开漏输出模式
25: 24	MODE6	rw	0x00	10: 复用功能推挽输出模式

Bit	Field	Type	Reset	Description
21: 20	MODE5	rw	0x00	11: 复用功能开漏输出模式 端口输入输出配置 (MODEy) (y = 0...7) 软件配置相应的 I/O 端口；参考端口位配置表 配置表 00: 输入模式；01: 输出模式 10: 保留； 11: 保留
17: 16	MODE4	rw	0x00	
13: 12	MODE3	rw	0x00	
9: 8	MODE2	rw	0x00	
5: 4	MODE1	rw	0x00	
1: 0	MODE0	rw	0x00	

## 11.6.2 端口配置高寄存器 (GPIOx\_CRH)(x = A..D)

偏移地址: 0x04

复位值: 0x4444 4444

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	CNF15		MODE15		CNF14		MODE14		CNF13		MODE13		CNF12		MODE12	
Type	rw		rw													
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CNF11		MODE11		CNF10		MODE10		CNF9		MODE9		CNF8		MODE8	
Type	rw		rw													

Bit	Field	Type	Reset	Description
31: 30	CNF15	rw	0x01	端口配置位 (15...8) (Port x configuration bits) 配置 MODE==0,端口为输入模式： 00: 模拟输入模式 01: 浮空输入模式 10: 上拉/下拉输入模式 11: 保留 配置 MODE==1,端口为输出模式 00: 通用推挽输出模式 01: 通用开漏输出模式 10: 复用功能推挽输出模式
27: 26	CNF14	rw	0x01	
23: 22	CNF13	rw	0x01	
19: 18	CNF12	rw	0x01	
15: 14	CNF11	rw	0x01	
11: 10	CNF10	rw	0x01	
7: 6	CNF9	rw	0x01	
3: 2	CNF8	rw	0x01	
29: 28	MODE15	rw	0x00	
25: 24	MODE14	rw	0x00	

Bit	Field	Type	Reset	Description
21: 20	MODE13	rw	0x00	11: 复用功能开漏输出模式 端口输入输出配置 (MODEy) (y = 15...8) 软件配置相应的 I/O 端口; 参考端口位配置表 配置表 00: 输入模式; 01: 输出模式 10: 保留; 11: 保留
17: 16	MODE12	rw	0x00	
13: 12	MODE11	rw	0x00	
9: 8	MODE10	rw	0x00	
5: 4	MODE9	rw	0x00	
1: 0	MODE8	rw	0x00	

### 11.6.3 端口输入数据寄存器 (GPIOx\_IDR)(x = A..D)

偏移地址: 0x08

复位值: 0x0000 XXXX

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

Bit	Field	Type	Reset	Description
31:16	Reserved			始终读为 0
15:0	IDRy	r	0xXX	端口输入数据位 (y = 0..15) 只能以 16 位 的形式读操作, 读出的值代表了对应的 I/O 口的状态。

### 11.6.4 端口输出数据寄存器 (GPIOx\_ODR)(x = A..D)

偏移地址: 0xC

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31: 16	Reserved			始终读为 0
15: 0	ODRY	rw	0x0000	端口输出数据(y=0..15)(port output data) 这些位为可读, 只能以 16 位的方式读出 注: 对 GPIOx_BSRR(x=A...D), 可以分别独立的对各个 ODR 位独立的设置/清除

### 11.6.5 端口设置/清除寄存器 (GPIOx\_BSRR)(x = A..D)

偏移地址: 0x10

复位值: 0x0000 000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	BR15	BR14	BR13	BR12	BR11	BR10	BR9	BR8	BR7	BR6	BR5	BR4	BR3	BR2	BR1	BR0
Type	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	BS15	BS14	BS13	BS12	BS11	BS10	BS9	BS8	BS7	BS6	BS5	BS4	BS3	BS2	BS1	BS0
Type	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bit	Field	Type	Reset	Description
31:16	BRy	w	0x0000	端口清除位 y (y=0...15) 通过软件置位与清除 0: 相应的 ODRY 位没有改变 1: 清除对应的 ODRY 位为 0
15:0	BSy	w	0x0000	端口置位位 y (y=0...15) 通过软件置位与清除 0: 相应的 ODRY 位没有改变 1: 置位对应的 ODRY 位为 1

## 11.6.6 端口位清除寄存器 (**GPIOx\_BRR**)(*x* = A..D)

偏移地址: 0x14

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31: 16	Reserved			始终读为 0
15: 0	BRy	w	0x0000	端口清除位 y (y=0...15) 通过软件置位与清除 0: 相应的 ODRY 位没有改变 1: 清除对应的 ODRY 位为 0

## 11.6.7 端口配置锁定寄存器 (**GPIOx\_LCKR**)(*x* = A..D)

当此寄存器的 bit16 被 Lock key 写入序列配置后, bit[0:15]便可用于 GPIO 端口的锁定,当端口被锁定后, 只能在系统复位之后才能再次更改端口为的配置, 且在规定的写入操作期间, 不允许改变 bit[0:15], 一个锁定位锁定控制寄存器中的 4 个位。

地址偏移: 0x18

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:17	Reserved			始终读为 0

Bit	Field	Type	Reset	Description
16	LCKK	rw	0x00	<p>锁键 (Lock key) 该位可随时读出,它只可通过锁键写入序列修改。锁键 (Lock key)</p> <p>该位可随时读出,它只可通过锁键写入序列修改。</p> <p>0: 端口配置锁键位激活 1: 端口配置锁键位被激活,下次系统复位前 GPIOx_LCKR 寄存器被锁住</p>
15:0	LCKy	rw	0x00	<p>端口 x 的锁位 y (y = 0...15) 通过软件置位与清除</p> <p>这些位可读可写但只能在 LCKK 位为 0 时写入。</p> <p>0: 不锁定端口的配置 1: 锁定端口的配置</p>

### 11.6.8 端口输出开漏控制寄存器 (GPIOx\_DCR)(x = A..D)

偏移地址: 0x1C

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	PX15		PX14		PX13		PX12		PX11		PX10		PX9		PX8	
Type	rw		rw		rw											
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	PX7		PX6		PX5		PX4		PX3		PX2		PX1		PX0	
Type	rw		rw		rw											

Bit	Field	Type	Reset	Description
31: 2	PX15-PX1	rw	0x00	见 PX0
1: 0	PX0	rw	0x00	<p>PX0[1: 0]:</p> <p>11: 开漏输出模式下,端口上拉 01: 开漏输出模式下,端口下拉 x0: 开漏输出模式下,端口无上下拉</p>

### 11.6.9 端口复用功能低位寄存器 (GPIOx\_AFRL)(x = A..D)

偏移地址: 0x20

复位值: 0xFFFF FFFF

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	AFR7				AFR6				AFR5				AFR4			
Type	rw															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	AFR3				AFR2				AFR1				AFR0			
Type	rw															

Bit	Field	Type	Reset	Description
31:0	AFRy	rw	0xFFFF	端口 x 的位 y (y = 0...7) 的复用功能选择, FFFF 这些位能软件写入配置 IO 复用功能。 0000: AF0 0001: AF1 0010: AF2 0011: AF3 0100: AF4 0101: AF5 0110: AF6 0111: AF7

### 11.6.10 端口复用功能高位寄存器 (GPIOx\_AFRH)(x = A..D)

偏移地址: 0x24

复位值: GPIOA\_AFRH: 0xF00F FFFF ,

GPIOB\_AFRH: 0xFFFF FFFF,

GPIOC\_AFRH: 0xFFFF FFFF,

GPIOD\_AFRH: 0x000F FFFF,

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	AFR15				AFR14				AFR13				AFR12			
Type	rw															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	AFR11				AFR10				AFR9				AFR8			

Type	rw		rw		rw
------	----	--	----	--	----

Bit	Field	Type	Reset	Description
31: 0	AFRy	rw		<p>端口 x 的位 y (y = 8...15) 的复用功能选择这些位能软件写配置 IO 复用功能。</p> <p>0000: AF0      0001: AF1      0010: AF2      0011: AF3      0100: AF4      0101: AF5      0110: AF6      0111: AF7</p>

## 12 高级定时器 (TIM1/8)

### 12.1 TIM1 简介

TIM1 由一个 16 位可实时编程预分频器和一个 16 位计数方向可调的自动重装载计数器组成，可以为用户提供便捷的计数定时功能，计数器时钟由预分频器分频得到。高级定时器具有多种用途，如输入功能（测量输入信号的脉冲宽度、频率，PWM 输入等），输出功能（PWM 输出、死区时间可编程的互补输出、单脉冲模式输出等）。

### 12.2 功能框图

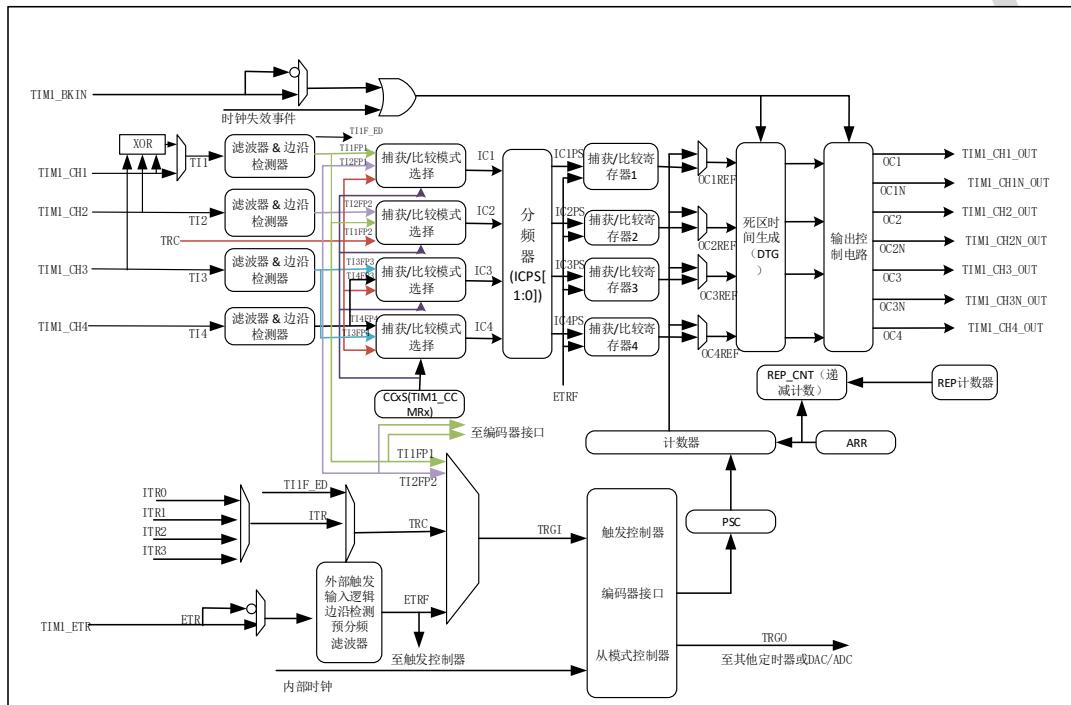


图 12-1 TIM1 结构图

上图为 TIM1 的结构框图，主要由输入单元、输出单元、时基单元、捕获/比较模块、刹车单元等结构组成。

### 12.3 主要特征

- 1) 16 位可实时编程预分频器，分频系数：1–65536 可调。
- 2) 时钟源可选：内部时钟源，外部时钟模式（外部时钟引脚/外部触发输入 ETR），内部触发输入（编码器模式）。
- 3) 16 位自动重装载计数器（计数方向：递增、递减、递增/递减）。
- 4) 16 位可编程重复计数功能，重复计数器可自动重装载（定时器到指定时间后自动更新重复寄存器）。
- 5) 定时器间互连同步电路由外部信号控制。
- 6) 输入捕获：输入信号的脉冲宽度、周期的测量。
- 7) 触发输入可以作为外部时钟或者逐周期管理。
- 8) 支持编码器、霍尔传感器等接口。

- 9) 4 个输出通道，通道 1/2/3 有互补输出通道，通道 4 无互补输出通道。
- 10) 输出比较（控制输出波形或指示定时器已经计时结束）。
- 11) PWM 输出（死区时间可调）（边沿对齐或中央对齐模式）。
- 12) 刹车输入可将计时器的输出信号置于安全状态（复位态或已知态，用户可选）。
- 13) 单脉冲输出。
- 14) 产生中断/DMA 请求的事件：更新事件、触发事件、输入捕获、输出比较或者刹车输入时。

注：

- 更新事件：计数器上溢/下溢，计数器初始化。
- 触发事件：计数器初始化，启动或停止计数。

## 12.4 中断

TIM1 的中断包括：捕获/比较 1 中断、捕获/比较 2 中断、捕获/比较 3 中断、捕获/比较 4 中断、捕获/比较 5 中断、更新中断、COM 中断、触发中断和刹车中断，当相应的中断使能位打开，发生相应的事件时，产生相应的中断。

## 12.5 DMA

TIM1 能够在发生单个事件时生成多个 DMA 请求。主要目的是在没有软件开销的情况下，多次重新编程 TIM1 的一部分寄存器也可以用于按周期读取数个寄存器。

## 12.6 功能描述

### 12.6.1 时钟

#### 12.6.1.1 时钟选择

计数器的时钟源有以下几种：

- 1) 内部时钟 (INT\_CK)。
- 2) 外部时钟模式：外部输入引脚 (**Tix**) 或者外部触发输入 (**ETR**)。
- 3) 内部触发输入（编码器模式）。

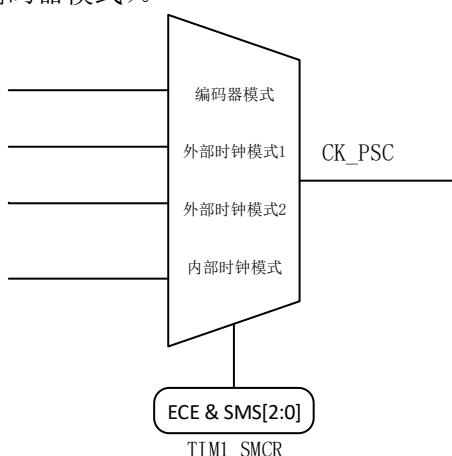


图 12-2 时钟选择

### 12.6.1.1.1 内部时钟源 (INT \_CK)

当 SMS=000，关闭从模式时，计数器使能打开，预分频器的时钟直接由内部时钟驱动。此时计数器时钟为内部时钟分频后的时钟。

### 12.6.1.1.2 外部时钟模式 1 (外部输入引脚 TIx)

当 SMS = 111 时，选择外部时钟模式 1 (外部输入引脚 TIx)。计数器由选定的输入引脚的每个上升沿和下降沿驱动。

例：计数器在 T11 输入端的上升沿递增计数，具体配置如下：

- 1) 配置 TIM1\_CCMR1 寄存器 CC1S=01, IC1F[3:0]=xxxx, TIM1\_CCER 寄存器 CC1P=0，选择通道 1 检测 TI1 输入的上升沿为有效沿，定义输入滤波器带宽。
- 2) 配置 TIM1\_SMCR 寄存器中的 TS=100, SMS=111，选定 TI1 的边沿作为触发输入源，时钟选择外部时钟模式 1。
- 3) 配置 TIM1\_CR1 寄存器的 DIR=0, CEN=1，选择递增计数模式，启动计数器。

当 TI1 出现有效边沿时，计数器递增计数一次且 TIF 标志位由硬件置 1。TI1 的有效边沿和计数器的实际时钟之间的延时取决于预 TI1 输入引脚同步电路设计。

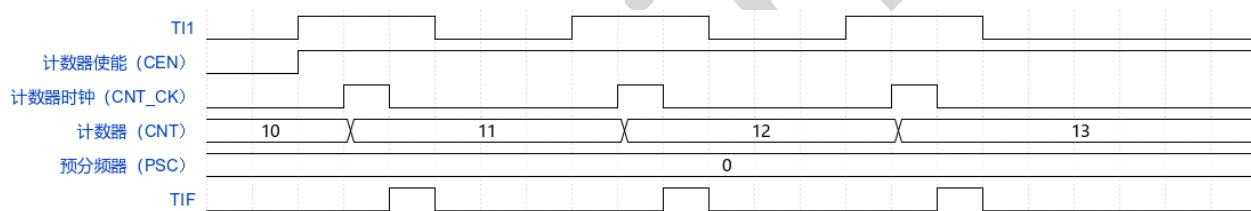


图 12-3 外部时钟模式 1 下的控制电路

### 12.6.1.1.3 外部时钟模式 2 (外部触发输入 ETR)

当 TIM1\_SMCR 寄存器的 ECE=1 时，使能外部时钟模式 2，计数器由 ETRF 信号上的有效边沿驱动。

例：在 ETR 下每 4 个下降沿计数一次的递增计数器，具体配置如下：

- 1) 配置 TIM1\_SMCR 寄存器中的 ETF[3: 0]=0010，每 4 个 ETR 信号的有效边沿驱动计数器计数一次；ETP=1，ETR 被反相，选择下降沿有效；ECE=1，选择外部时钟模式 2。
  - 2) 配置 TIM1\_CR1 寄存器 DIR=0, CEN=1 选择计数模式为递增计数模式，启动计数器。
- 在 ETR 的下降沿和计数器实际时钟之间的延时取决于在 ETR 信号端的重新同步电路。

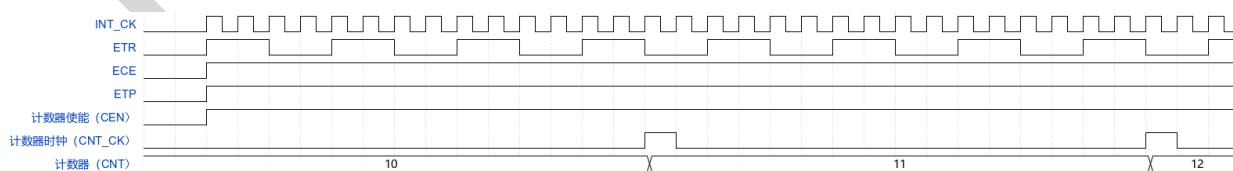


图 12-4 外部时钟模式 2 下的控制电路

### 12.6.1.2 时基单元

TIM1 的时基单元主要包括：计数器寄存器 (TIM1\_CNT)、预分频器寄存器 (TIM1\_PSC)、自动装载寄存器 (TIM1\_ARR) 和重复计数器 (TIM1\_RCR)。

计数器由一个 16 位的计数器和对应的自动装载寄存器组成，可以实现递增计数，递减计数的功能。计数器的时钟由预分频器提供，预分频器由预分频计数器和对应的寄存器组成，分频系数为 1-65536，可以随时写入，在下一次更新事件时生效。

自动预装载寄存器有预装载功能的 16 位影子寄存器，通过设置 TIM1\_CR1 寄存器的 APRE 位选择写入 ARR 寄存器的值立即生效或发生更新事件时载入影子寄存器。

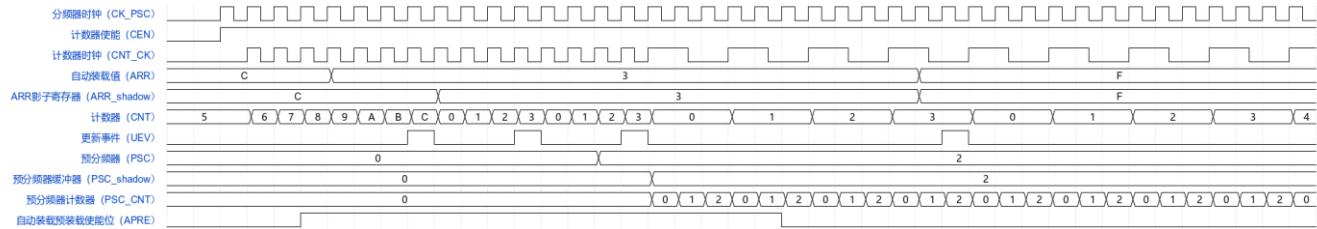


图 12-5 自动预装载

### 12.6.1.3 计数模式

通过配置 TIM1\_CR1 寄存器的 DIR 位和 CMS 位可以选择计数器的计数模式，可以分为三种计数模式，递增计数模式、递减计数模式和中央对齐计数模式（递增/递减计数模式），下面对每个计数模式做详细介绍。

#### 12.6.1.3.1 递增计数模式

配置 TIM1\_CR1 寄存器 CMS=0，DIR=0，选择递增计数模式。

递增计数模式下，在使能 TIM1\_CR1 的 CEN 后计数器由 0 开始递增计数，直至 TIM1\_ARR 的值，并产生一个计数器上溢事件（更新事件），更新事件后计数器从 0 开始重新递增计数。当用户启用了重复计数功能后，重复计数器在每次上溢事件时递减计数，只有当重复计数器从设定值递减到 0 时，才会产生更新事件；设置 TIM1\_EGR 寄存器的 UG=1，同样可以产生一个更新事件。

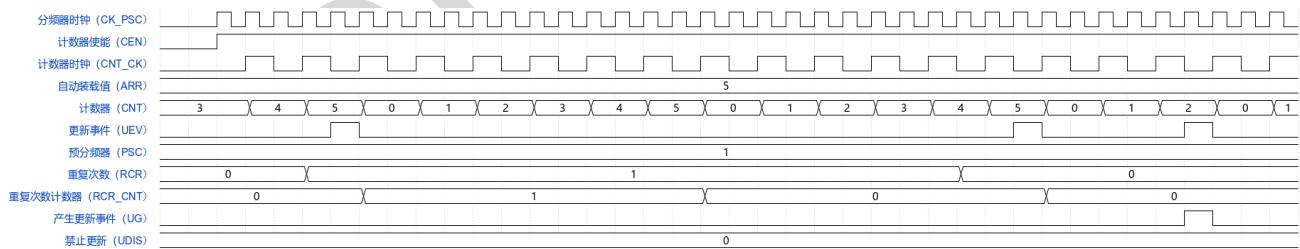


图 12-6 递增计数模式 (UDIS=0)

当配置 TIM1\_CR1 寄存器的 UDIS=1，禁止产生更新事件，当计数器发生上溢事件时，不产生更新事件；配置 UG=1，不产生更新事件，计数器和预分频器计数器会被初始化，从零开始递增计数。

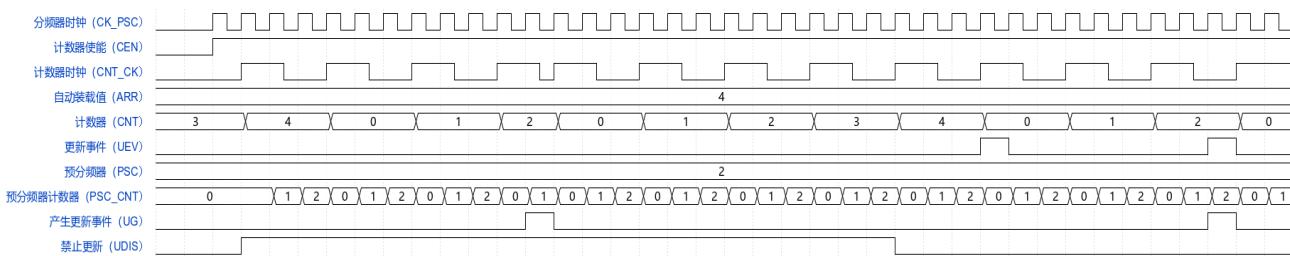


图 12-7 递增计数模式 (UDIS=0 禁止产生更新事件)

注：发生更新事件时：

- 重复计数器被载入 RCR 寄存器中的值，并重新开始递减计数。
  - ARR 寄存器中的值被载入 ARR\_shadow 寄存器中。
  - 预分频器的预装载值生效。

#### 12.6.1.3.2 递减计数模式

配置 TIM1\_CR1 寄存器的 CMS=0, DIR=1, 选择递减计数模式。

递减计数模式下，此时计数器从自动装载值 TIM1\_ARR 开始递减计数，计数到 0 时，产生一个下溢事件（更新事件）。当用户启用了重复计数功能后，重复计数器在每次下溢事件时递减计数，只有当重复计数器从设定值递减到 0 时，才会产生更新事件；设置 TIM1\_EGR 寄存器的 UG=1，同样可以产生一个更新事件，更新事件后计数器从自动装载值 TIM1\_ARR 开始重新递减计数（TIM1\_CR1 寄存器 UDIS=0）。

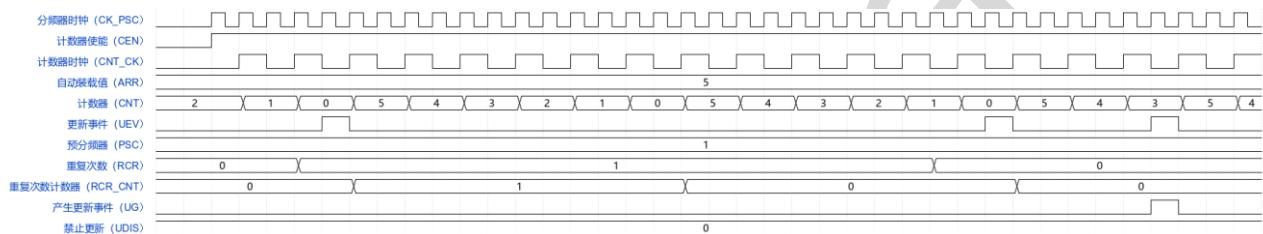


图 12-8 递减计数模式 (UDIS=0)

当配置 TIM1\_CR1 寄存器的 UDIS=1，禁止产生更新事件，当计数器发生下溢事件时，不产生更新事件；配置 UG=1，同样不产生更新事件，但是计数器和预分频器计数器会被初始化，从 TIM1\_ARR 开始计数。

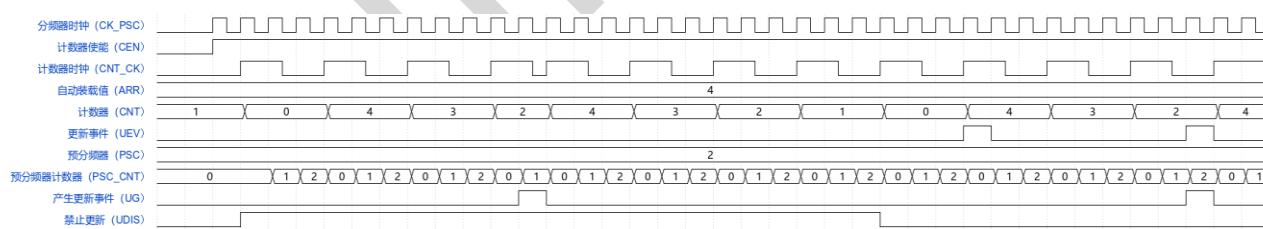


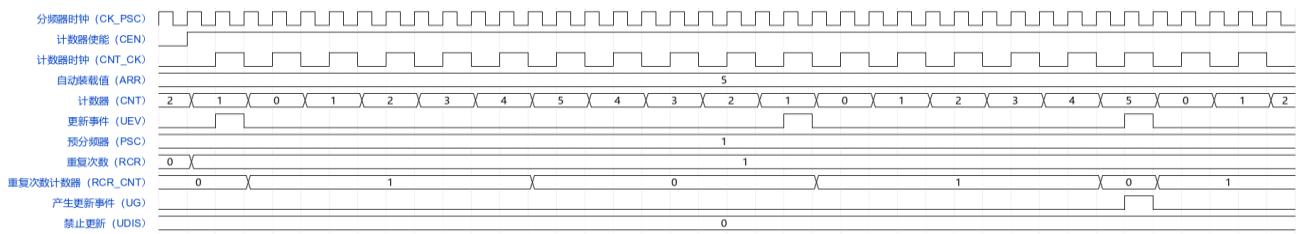
图 12-9 递减计数模式 (UDIS=1 禁止产生更新事件)

### 12.6.1.3.3 中央计数模式（递增/递减计数模式）

配置 TIM1\_CR1 寄存器的 CMS  $\neq 0$  (此时写入 DIR 无效)，选择中央对齐计数模式。

中央对齐计数模式，递增计数和递减计数交替进行。递增计数到 ARR-1 时，产生一个上溢事件，然后从 ARR 先开始递减计数到 1，产生一个下溢事件，再从 0 开始递增计数。

当用户启用了重复计数功能后，重复计数器在每次上溢事件或下溢事件时递减重复计数值，只



有当重复计数器从设定值递减到 0 时，才会产生更新事件；设置 TIM1\_EGR 寄存器的 UG=1，同样可以产生一个更新事件，更新事件后计数器从 0 开始重新递增计数（TIM1\_CR1 寄存器 UDIS=0）。

图 12-10 中央计数模式（UDIS=0）

当配置 TIM1\_CR1 寄存器的 UDIS=1，禁止产生更新事件，当计数器发生上溢或下溢事件时，不产生更新事件；配置 UG=1，同样不产生更新事件，但是计数器和预分频器计数器会被初始化，从零开始重新计数。

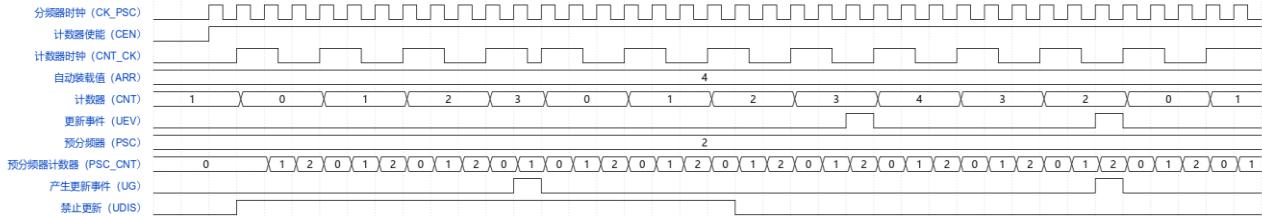


图 12-11 中央计数模式（UDIS=1 禁止产生更新事件）

### 12.6.1.4 定时器同步

不同的 TIMx 定时器在内部连接，通过配置一个定时器工作在主模式，一个定时器工作在从模式（配置 TIMx\_SMCR 寄存器 SMS=100/101/110，从模式选择复位模式、门控模式或触发模式），可以实现定时器之间的级联或同步，实现对处于从模式定时器的计数器进行复位、启动、通知或提供时钟等操作。

#### 12.6.1.4.1 从模式

##### 复位模式

配置 TIM1\_SMCR 寄存器 SMS=100，从模式选择复位模式。此模式下，发生触发输入事件会使计数器清零重启。

例如，TI2 输入端的下降沿触发计数器重启：

- 1) 配置 CC2S=01，CC2 通道被配置为输入模式，IC2 映射在 TI2 上，配置 TIM1\_CCER 寄存器 CC2P=1，检测下降沿。
- 2) 配置 TIM1\_SMCR 寄存器 SMS = 100，从模式选择为复位模式；置 TIM1\_SMCR 寄存器中 TS = 110，ETF=0000，选择滤波后的定时器输入 2 (TI2FP2) 作为同步计数器的触发输入。
- 3) 配置 TIM1\_CR1 寄存器 ARR, PSC=0, DIR=0，配置预分频系数并选择计数方向，配置 CEN=1，使能计数器。

计数器的时钟源由内部时钟提供，当检测到 TI2 的下降沿，计数器被清零重启。此时触发器中断标记被硬件置 1。

下图为复位模式下 TIM1\_ARR = 0x13 的时序图。

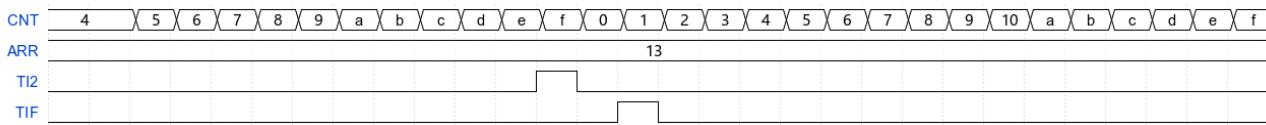


图 12-12 复位模式的控制时序图

## 门控模式

配置 TIM1\_SMCR 寄存器 SMS=101，从模式选择门控模式。此模式下，触发输入为高时，计数器始终开启，否则计数器停止（但不发生复位操作），计数器的开启和停止可控。

例如，计数器只在 TI1 为高时计数：

- 1) 配置 CC1S=01，CC1 通道被配置为输入模式，IC1 映射在 TI1 上，配置 TIM1\_CCER 寄存器 CC1P=0，检测 TI1 上的高电平。
- 2) 配置 TIM1\_SMCR 寄存器 SMS=101，从模式选择为门控模式，配置 TS=101，ETF=0000，选择滤波后的定时器输入 1 (TI1FP1) 作为同步计数器的触发输入。
- 3) 配置 TIM1\_CR1 寄存器 ARR, PSC=0, DIR=0，配置预分频系数并选择计数方向，配置 CEN=1，使能计数器。

计数器的时钟源由内部时钟提供，当检测到 TI1 的高电平，计数器开始计数，当 TI1 为低电平时，计数器停止。计数器开启或停止都会将 TIF 置 1。

下图为门控模式下 TIM1\_ARR=0xf 的时序图。

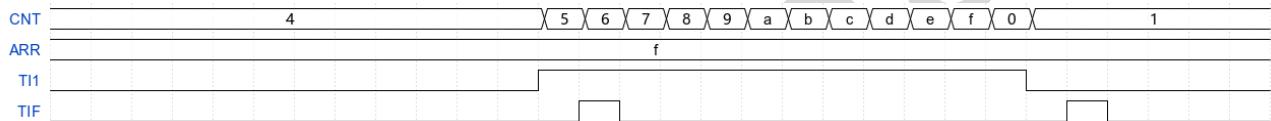


图 12-13 门控模式下的控制时序图

## 触发模式

配置 TIM1\_SMCR 寄存器 SMS=111，从模式选择触发模式。计数器在触发输入的上升沿启动，计数器的启动可控，停止不可控。

例如，计数器在 TI1 输入的上升沿开始计数：

- 1) 配置 CC1S=01，CC1 通道被配置为输入模式，IC1 映射在 TI1 上，配置 TIM1\_CCER 寄存器 CC1P=0，检测上升沿。
- 2) 配置 TIM1\_SMCR 寄存器 SMS = 110，从模式选择为触发模式；配置 TS=101，ETF=0000，选择滤波后的定时器输入 1 (TI1FP1) 作为同步计数器的触发输入。
- 3) 配置 TIM1\_CR1 寄存器 ARR, PSC=0, DIR=0，配置预分频系数并选择计数方向，配置 CEN=1，使能计数器。

计数器的时钟源由内部时钟提供，当检测到 TI1 的上升沿，计数器开始计数。

下图为触发模式下 TIM1\_ARR=0xf 的时序图。

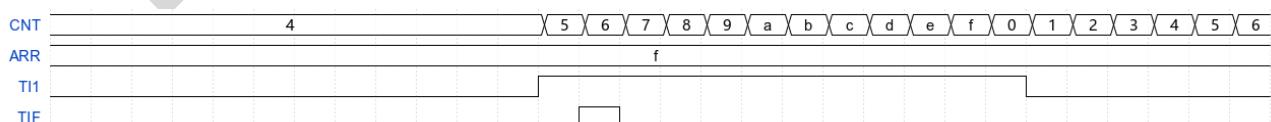


图 12-14 触发器模式下的控制时序图

## 外部时钟模式 2+触发模式

当时钟源选择外部时钟模式 2，ETR 信号被用作外部时钟的输入时，可以与另一种从模式（外部时钟模式 1 和编码器模式除外）一起使用。在复位模式、门控模式或触发模式可以选择另一个输入作

为触发输入 (TRGI)。

例如，计数器在 ETR 的每一个上升沿计数一次：

- 1) 配置 TIM1\_SMCR 寄存器 ETF = 0000, ETPS = 00, ETP = 0 检测 ETR 的上升沿，配置 ECE = 1 选择外部时钟模式 2。
- 2) 配置 TIM1\_CCMR1 寄存器中 CC1S=01，选择输入捕获源，TIM1\_CCER 寄存器中 CC1P=0，选择上升沿有效。
- 3) 配置 TIM1\_SMCR 寄存器中 SMS = 110，从模式选择为触发模式。配置 TIM1\_SMCR 寄存器中 TS = 101，选择 TI1 作为输入源。
- 4) 配置 TIM1\_CR1 寄存器 ARR, PSC=0, DIR=0，配置预分频系数并选择计数方向，配置 CEN=1，使能计数器。

计数器在 ETR 的上升沿开始计数，当检测到 TI1 上的上升沿时，TIF 被硬件置 1。ETR 信号的上升沿和计数器实际复位间的延时取决于 ETRP 输入端的重同步电路。

下图为外部时钟模式 2+触发模式下 TIM1\_ARR=13 时的时序图

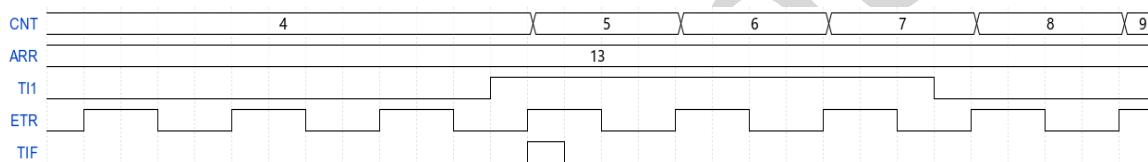


图 12-15 外部时钟模式 2+触发模式下的控制时序图

#### 12.6.1.4.2 定时器间的同步

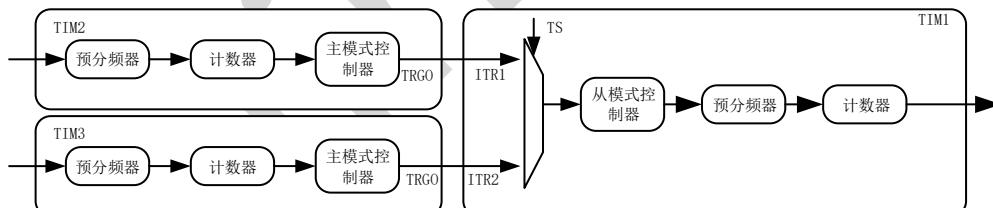


图 12-16 定时器间的互联

使用一个定时器作为另一个定时器的预分频器

例：使用 TIM3 作为定时器 1 的预分频器，流程如上图所示：

- 1) 配置 TIM3\_SMCR 寄存器 MMS=010，TIM3 为主模式，TIM3 的更新事件作为触发输入 (TRGO)，TIM3 在每次更新事件时输出一个周期信号。
- 2) 配置 TIM3\_ARR 寄存器，配置 TIM3 的输出周期。
- 3) 配置 TIM1\_SMCR 寄存器 TS=010，选择 TIM1 的触发源为 TIM3\_TRGO；
- 4) 配置 TIM1\_SMCR 寄存器 SMS=111，从模式选择外部触发模式 1。
- 5) 配置 TIM3\_CR1 寄存器 CEN=1，启动 TIM3。

6) 配置 TIM1\_CR1 寄存器 CEN=1, 启动 TIM1。

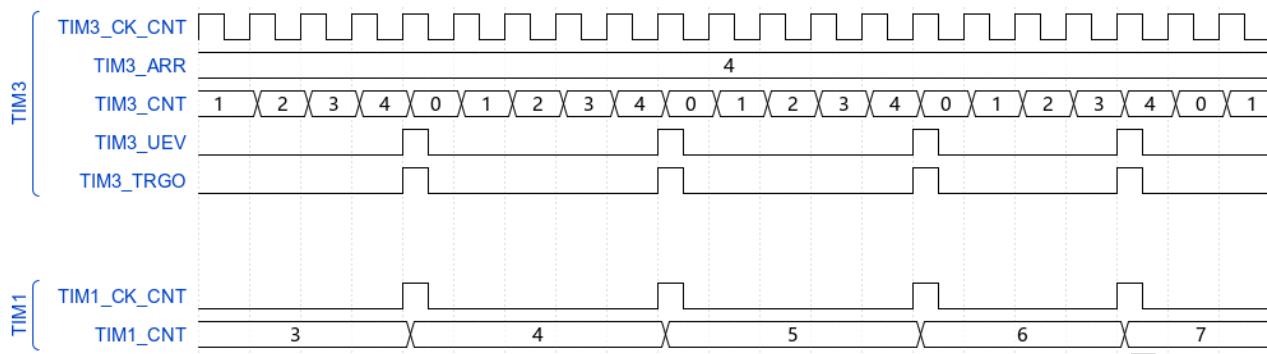


图 12-17 使用 TIM3 作为定时器 1 的预分频器

使用一个定时器使能另一个定时器

例：使用 TIM1 使能 TIM2，波形图如下图所示；当 TIM2 的 OC1REF 为高时，TIM2 的计数器才开始计数。两个 TIM 的时钟频率由 CK\_CNT/3 提供。具体配置如下：

- 1) 配置 TIM1\_SMCR 寄存器 MMS=100，选择 TIM1 为主模式，配置 TIM1 的输出比较参考信号 (OC1REF) 为触发输出 (TRGO)。
- 2) 配置 TIM1\_CCR1 寄存器，TIM1\_ARR 寄存器，TIM2\_CCMR1 寄存器 OC1M 位，CC1S 位等相关控制位，配置 TIM1 输出信号 TRGO 的输出波形。
- 3) 配置 TIM2\_SMCR 寄存器 TS=001，配置 TIM1 的 OC1REF 作为 TIM2 的触发输入。
- 4) 配置 TIM2\_SMCR 寄存器 SMS=101，配置 TIM2 为门控模式。
- 5) 配置 TIM1\_CR1 寄存器 CEN=1，启动 TIM1。
- 6) 配置 TIM2\_CR1 寄存器 CEN=1，启动 TIM2。

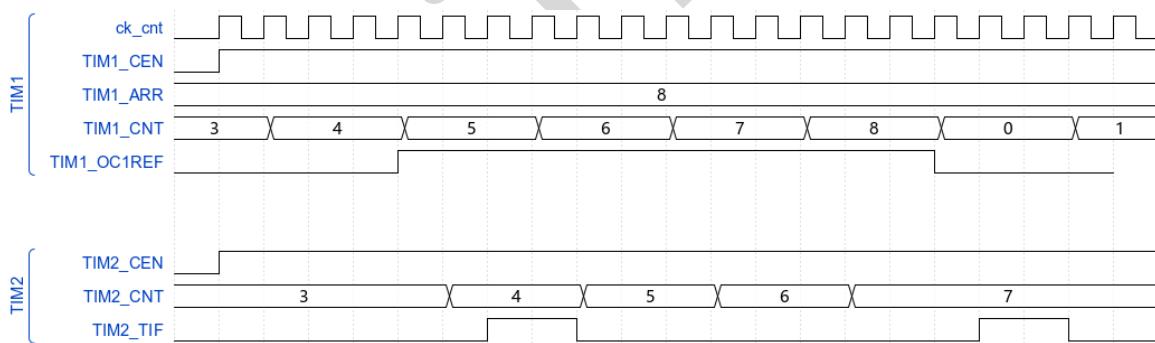


图 12-18 使用 TIM1 使能 TIM2

使用一个定时器启动另一个定时器

例：使用 TIM1 的更新事件启动 TIM2，时序图如下图所示。当 TIM1 产生更新事件时，TIM2 接收到触发信号，TIM2 的 CEN 由硬件自动置 1，TIM2 的计数器开始计数。两个 TIM 的时钟频率由 CK\_CNT/3 提供。具体配置如下：

- 1) 配置 TIM1\_SMCR 寄存器 MMS=010，选择 TIM1 的更新事件为触发输出 (TRGO)。
- 2) 配置 TIM1\_ARR 寄存器，配置更新事件产生周期。
- 3) 配置 TIM2\_SMCR 寄存器 TS=000，配置 TIM1 的 TRGO 作为 TIM2 的触发输入。
- 4) 配置 TIM2\_SMCR 寄存器 SMS=010，配置 TIM2 为触发模式。

5) 配置 TIM1\_CR1 寄存器 CEN=1, 启动 TIM1。

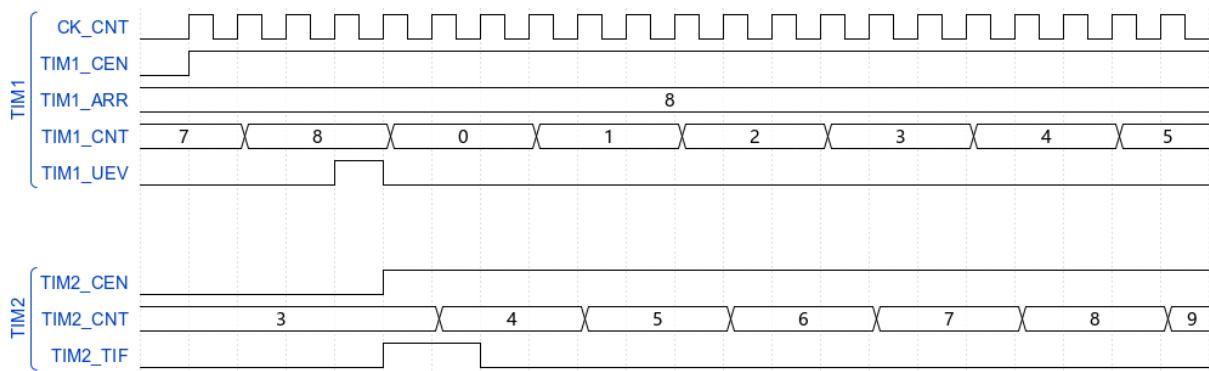


图 12-19 使用 TIM1 的更新事件启动 TIM2

使用一个外部触发同步启动两个定时器

例：配置 TIM1 TI1 上升沿到来时启动 TIM1，启动 TIM1 的同时启动 TIM2，时序图如下图所示。为了确保两个定时器同时开启，TIM1 必须在主/从模式下配置。具体配置如下：

- 1) 配置 TIM1\_CR2 寄存器 MMS=001，配置 TIM1 为主模式，将 TIM1 的使能作为触发输出 (TRGO)。
- 2) 配置 TIM1\_SMCR 寄存器 TS=100，配置 TIM1 为从模式，将 TI1 作为触发输入。
- 3) 配置 TIM1\_SMCR 寄存器 SMS=110，配置 TIM1 为触发模式。
- 4) 配置 TIM2\_SMCR 寄存器 TS=000，配置 TIM1 的触发输出作为 TIM2 的触发输入。
- 5) 配置 TIM2\_SMCR 寄存器 SMS=110，配置 TIM2 为触发模式。

当 TIM1 的 TI1 出现上升沿时，两个定时器同步启动（按照内部时钟），计数器开始计数，两个 TIF 标志也同时置 1。

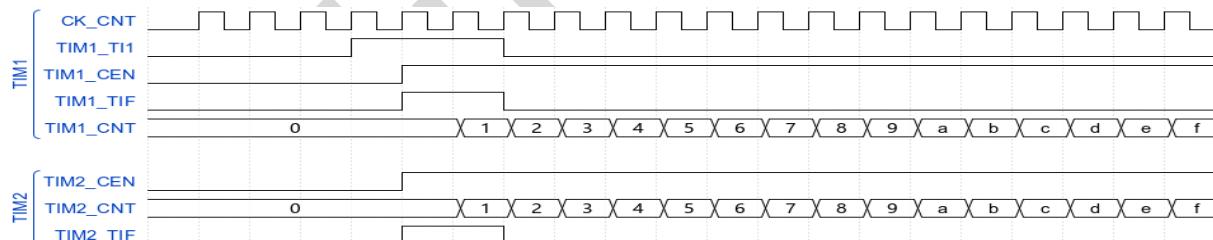


图 12-20 TIM1 的 TI1 同步启动两个 TIM1 和 TIM2

## 12.6.2 输入捕获

### 12.6.2.1 输入捕获

输入捕获部分包括数字滤波器、多路复用、预分频器等，其结构如下图所示：

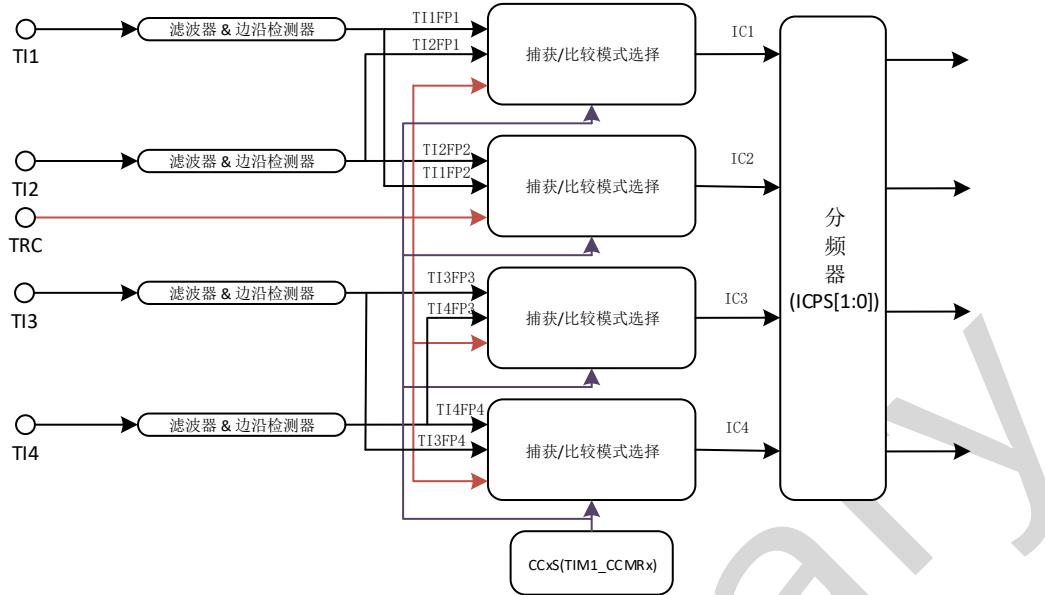


图 12-21 TIM1 输入捕获结构图

通过配置 TIM1\_CCMRx 寄存器 ICxF，可以配置数字滤波器的滤波宽度（滤波器的采样频率及数字滤波器长度如下表所示），当数字滤波器的输入信号带宽大于滤波宽度时，输入信号有效；数字滤波器对输入引脚 TIx 输入信号的采样后，产生一个滤波后的信号 TIxF，然后通过边沿检测器和软件配置选择 TIxF 信号的有效边沿，产生一个有效信号 TIxFPx，这个信号可以用作捕获控制信号或作为从模式控制器的触发输入信号，然后经过预分频产生一个信号 ICxPS，进入捕获/比较寄存器。

表 12-1 数字滤波器长度与 ICxF 的对应关系表

IC1F[3:0]	采样频率和滤波长度	IC1F[3:0]	采样频率和滤波长度
0000	无滤波器，以 $f_{DTS}$ 采样	1000	采样频率 $f_{sampling} = f_{DTS} / 8, N=6$
0001	采样频率 $f_{sampling} = f_{INT\_CK}, N=2$	1001	采样频率 $f_{sampling} = f_{DTS} / 8, N=8$
0010	采样频率 $f_{sampling} = f_{INT\_CK}, N=4$	1010	采样频率 $f_{sampling} = f_{DTS} / 16, N=5$
0011	采样频率 $f_{sampling} = f_{INT\_CK}, N=8$	1011	采样频率 $f_{sampling} = f_{DTS} / 16, N=6$
0100	采样频率 $f_{sampling} = f_{DTS} / 2, N=6$	1100	采样频率 $f_{sampling} = f_{DTS} / 16, N=8$
0101	采样频率 $f_{sampling} = f_{DTS} / 2, N=8$	1101	采样频率 $f_{sampling} = f_{DTS} / 32, N=5$
0110	采样频率 $f_{sampling} = f_{DTS} / 4, N=6$	1110	采样频率 $f_{sampling} = f_{DTS} / 32, N=6$
0111	采样频率 $f_{sampling} = f_{DTS} / 4, N=8$	1111	采样频率 $f_{sampling} = f_{DTS} / 32, N=8$

在输入捕获模式下，输入捕获发生在影子寄存器上，然后影子寄存器的内容载入到捕获/比较寄存器中。

输入捕获模式下，当检测到信号 ICx 上的有效边沿后，计数器的当前值被锁存到对应的影子寄存器(TIM1\_CCRx\_shadow)上，再复制到对应的捕获比较寄存器(TIM1\_CCRx)中。当开启了中断或 DMA 使能，发生捕获事件时，将产生相应的中断或 DMA 请求。发生捕获事件时，会将状态寄存器

(TIM1\_SR) 中的捕获标志位 CCxIF 置 1，通过配置 CCxIF=0 或读取 TIM1\_CCRx 中的数据，清除 CCxIF 标志位。当 CCxIF 未被清零时，发生输入捕获事件，重复捕获标志位将会被置 1，通过配置 CCxOF=0，可以清除 CCxOF 标志位。

例如，通过采样 TI1 输入信号的有效沿，在 TI1 的上升沿来到时捕获当前计数器的值，锁存到 TIM1\_CCR1 寄存器中，步骤如下：

- 1) 配置 TIM1\_CCMR1 寄存器 CC1S=01，CC1 通道被配置为输入，IC1 映射在 TI1 上。
- 2) 配置 TIM1\_CCMRx 寄存器 IC1F[3:0]，配置数字滤波器的滤波宽度（按需配置）。
- 3) 配置 TIM1\_CCER 寄存器 CC1P=0，选择捕获发生在 TI1 信号的上升沿。
- 4) 配置 TIM1\_CCMR1 寄存器的 IC1PS，选择预分频系数。
- 5) 配置 TIM1\_CCER 寄存器的 CC1E = 1，开启输入/捕获通道 1 的捕获使能。
- 6) 配置 TIM1\_DIER 寄存器 CC1IE=1，CC1DE=1，开始通道 1 的捕获/比较通道 1 中断使能和允许捕获/比较通道 1 的 DMA 请求。

注：

- 当通道配置为输入模式时，TIM1\_CCRx 寄存器属性变为只读。
- 为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息，建议在读出捕获溢出标志之前读取数据。
- 设置 TIM1\_EGR 寄存器中相应的 CCxG 位，可以通过软件产生输入捕获中断或 DMA 请求。

### 12.6.2.2 PWM 捕获

PWM 输入模式的操作配置与一般输入捕获有以下不同点：

- 1) 两个边沿有效且极性相反的 ICx 信号被映射至同一个 TIx 输入。
- 2) 配置从模式控制器为复位模式，将其中一路 TIxFP 作为触发输入信号。

例：测量 TI1 的 PWM 信号的长度 (TIM1\_CCR1 寄存器) 和占空比 (TIM1\_CCR2 寄存器)，具体步骤如下 (取决于 INT\_CK 的频率和预分频器的值)。

- 1) 配置 TIM1\_CR1 寄存器 DIR=0，选择计数器计数模式为递增计数模式。
- 2) 配置 TIM1\_CCMR1 寄存器的 CC1S = 01，将 IC1 映射在 TI1 上，选择 TIM1\_CCR1 的有效输入。
- 3) 配置 CC1P =0，选择 TI1FP1 的有效极性(上升沿有效)(将计数器的值捕获到 TIM1\_CCR1 中并清除计数器)。
- 4) 配置 TIM1\_CCMR1 寄存器的 CC2S =10，将 IC2 映射在 TI1 上，选择 TIM1\_CCR2 的有效输入。
- 5) 配置 CC2P =0，选择 TI2FP2 的有效极性 (下降沿有效) (将计数器的值捕获到 TIM1\_CCR2 中)。
- 6) 配置 TIM1\_SMCR 寄存器中的 TS = 101，选择 TI1FP1 为有效的触发输入信号。
- 7) 配置 TIM1\_SMCR 中的 SMS = 100，从模式控制器设置为复位模式。

- 8) 配置 TIM1\_CCER 寄存器中 CC1E=1 且 CC2E = 1。开启 CC1 通道和 CC2 通道的捕获使能。



图 12-22 PWM 输入模式时序

注：由于从模式控制器只连接了 TI1FP1 和 TI2FP2，所以 PWM 输入模式只适用于 TIM1\_CH1/TIM1\_CH2 信号。

### 12.6.2.3 编码器接口

编码器接口模式就是计数器在 TI1 和 TI2 正交信号相互作用下计数，在输入源改变期间，计数方向被硬件自动修改。通过配置 TIM1\_SMCR 寄存器 SMS 位可以选择输入源，根据输入源的不同，可以将编码器接口模式分为 3 种模式，SMS=001，编码器接口模式 1；SMS=010，编码器接口模式 2；SMS=011，编码器接口模式 3；三种模式具体计数操作如下表所示。两个输入 TI1 和 TI2 被用来作为正交编码器的接口。

编码器模式下，计数器开启之前必须先配置好 ARR 寄存器，因为使用编码器接口模式相当于使用了一个带有方向选择的外部时钟。计数器在 0 到 TIM1\_ARR 寄存器的自动装载值之间连续计数（递增计数和递减计数由外部时钟控制）。

注：编码器模式不支持外部时钟模式 2。

编码器接口模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合，假设 TI1 和 TI2 不同时变换。

表 12-2 计数方向与编码器信号的关系

计数模式	相对电平 (TI1FP1 相对于 TI2, TI1FP1 信号)		TI2FP2 信号	
	TI2FP2 相对于 TI1)		上升	下降
编码器接口模式 1 (只在 TI1 计数)	高电平		递减计数	递增计数
	低电平		递增计数	递减计数
编码器接口模式 2 (只在 TI2 计数)	高电平	-	-	递增计数 递减计数
	低电平	-	-	递减计数 递增计数
编码器接口模式 3 (在 TI1 和 TI2 计数)	高电平		递减计数	递增计数 递减计数
	低电平		递增计数	递减计数 递增计数

下例是计数器在编码器接口模式下的配置和时序图，从图中可以看出计数信号的产生和方向控制。具体配置如下：

- 1) 配置 TIM1\_CCMR 寄存器 CC1S=01，将 IC1FP1 映射到 TI1 上。
- 2) 配置 TIM1\_CCMR 寄存器 CC2S =01，将 IC2FP2 映射到 TI2 上。

- 3) 配置 TIM1\_CCER 寄存器 CC1P =0, IC1 不反相, 此时 IC1=TI1。
- 4) 配置 TIM1\_CCER 寄存器 CC2P =0, IC2 不反相, 此时 IC2=TI2。
- 5) 配置 TIM1\_SMCR 寄存器 SMS =011, 选择编码器模式 3, 根据另一个信号的输入电平, 计数器在 TI1FP1 和 TI2FP2 的边沿计数。
- 6) 配置 TIM1\_CR1 寄存器 CEN =1, 开启计数器。

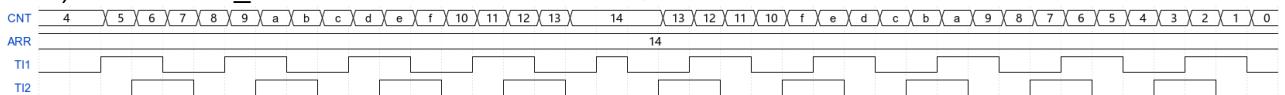


图 12-23 编码器模式下的计数器时序图

下图为当 IC1FP1 反相时计数器的时序图（CC1P = 1, 其他配置不变）

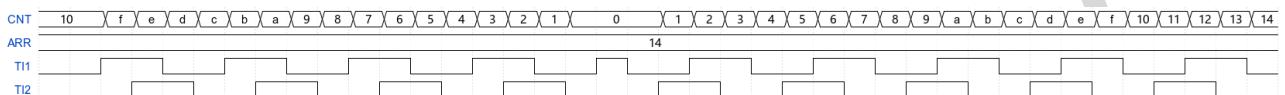


图 12-24 IC1FP1 反相的编码器接口模式时序图

编码器接口模式下，计数器可以提供传感器当前位置的信息。通过使用另一个配置在捕获模式的定时器测量两个编码器事件的间隔周期来获得动态的信息（速度，加速度，减速度）。根据两个编码器事件的间隔周期，可以定期读取计数器。可以通过把计数器的值锁存到第三个输入捕获寄存器（捕获信号必须是周期性的并且可以由另一个定时器产生）来实现。还可以通过 DMA 请求来读取它的值。

#### 12.6.2.4 定时器异或

配置 TIM1\_CR2 寄存器的 TI1S =1, 将 TIM1\_CH1、TIM1\_CH2 和 TIM1\_CH3 引脚经异或后连接到 TI1 的输入端，用于定时器的所有输入模式。

例：TIM1\_CH1、TIM1\_CH2 和 TIM1\_CH3 引脚经异或后连接到 TI1 的输入端，采样 TI1 输入信号的有效沿，在 TI1 的上升沿来到时捕获当前计数器的值，锁存到 TIM1\_CCR1 寄存器中，具体配置如下：

- 1) 配置 TIM1\_CR2 寄存器 TI1S=1, 配置定时器的三个输入经异或后连接到 TI1 输入通道。
- 2) 配置 TIM1\_CCMR1 寄存器 CC1S=01, CC1 通道被配置为输入, IC1 映射在 TI1 上。
- 3) 配置 TIM1\_CCMR1 寄存器 IC1F[3:0], 配置数字滤波器的滤波宽度（按需配置）。
- 4) 配置 TIM1\_CCER 寄存器 CC1P=0, 选择捕获发生在 TI1 信号的上升沿。
- 5) 配置 TIM1\_CCMR1 寄存器的 IC1PS, 选择预分频系数。
- 6) 配置 TIM1\_CCER 寄存器的 CC1E = 1, 开启输入/捕获通道 1 的捕获使能。
- 7) 配置 TIM1\_CR1 寄存器 CEN=1, 启动计数器。



图 12-25 (TI1 异或输入) 输入捕获波形图

#### 12.6.2.4.1 霍尔接口电路

霍尔传感器接口模式是异或功能的一个应用实例，可以用来驱动电机，在使用 TIM1 产生 PWM 信号驱动电机时，可以将另一个计数器 TIM2/TIM3 作为“接口定时器”来连接霍尔传感器，配置 TIM1\_CR2 寄存器 TI1S=1，将定时器的 3 个输入脚（CH1、CH2、CH3）经异或后连接到 TI1 输入通道，“接口定时器”接收这个信号。三个霍尔传感器与“接口定时器”的三路输入捕获引脚对应连接，每个传感器输入一路波形到输入引脚。分析输入捕获信号可以计算电机速度的信息。

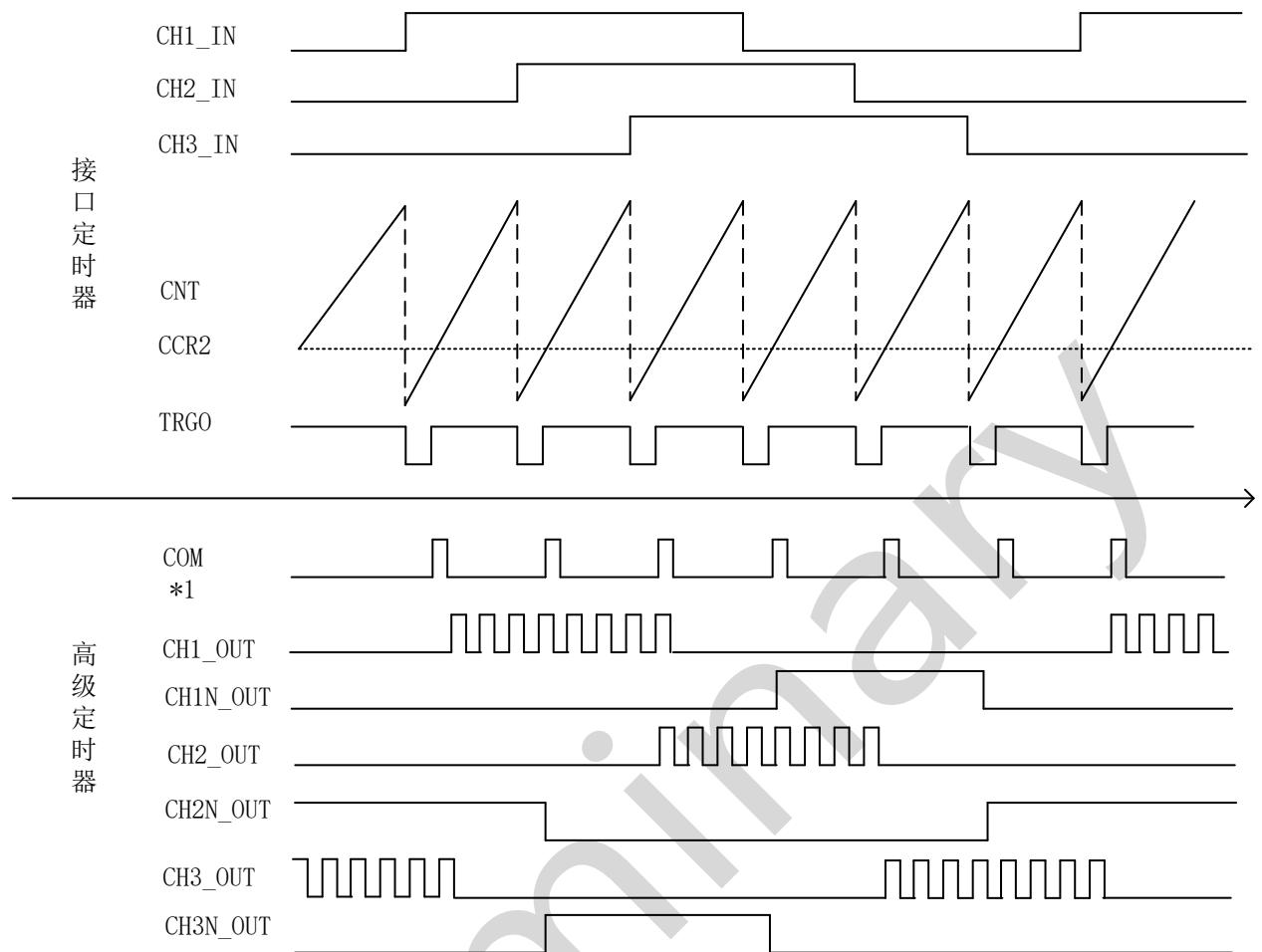
“接口定时器”在输出模式下可以产生一个用来控制 TIM1\_PWM 输出的脉冲，用来驱动电机。所以“接口定时器”在输出比较或 PWM 模式延时一段时间后产生一个正脉冲，然后通过 TRGO 输出被传送到 TIM1。

例：霍尔输入连接到 TIM2 定时器，每次任一霍尔输入上信号变化都会改变 TIM1 的 PWM 配置。

- 1) 配置 TIM2\_CR2 寄存器 TI1S=1，配置三个定时器输入经异或后连接到 TI1 输入通道。
- 2) 配置 TIM2\_ARR 为其最大值（计数器必须通过 TI1 的变化清零）。配置 PSC，设置计数周期大于传感器上的两次变化的时间。
- 3) 配置 TIM2\_CCMR1 寄存器设 CC1S=01，配置通道 1 为捕获模式（选中 TRC）。
- 4) 配置 TIM2\_CCMR1 寄存器 CC2S=00, OC2M=111，配置通道 2 为 PWM2 模式，并具有要求的延时。
- 5) 配置 TIM2\_CR2 寄存器 MMS=101，选择 OC2REF 作为 TRGO 上的触发输出。

TIM1 中，正确的 ITR 输入必须是触发器输入，定时器被编程为产生 PWM 信号，捕获/比较控制信号为预装载的（TIM1\_CR2 寄存器中 CCPC = 1），同时触发输入控制 COM 事件（TIM1\_CR2 寄存器中 CCUS = 1）。在一次 COM 事件后，写入下一步的 PWM 控制位（CCxE、OCxM），这可以在处理 OC2REF 上升沿的中断子程序里实现。

下图显示了这个实例：



\*1: com事件发生，通过中断子程序设定新的CCxE、CCxNE和OCxM来配置下一步动作。

图 12-26 霍尔传感器接口的实例

### 12.6.3 比较/输出

捕获比较通道的比较输出部分由比较器、输出控制电路和捕获/比较寄存器组成，其结构图如下图所示：

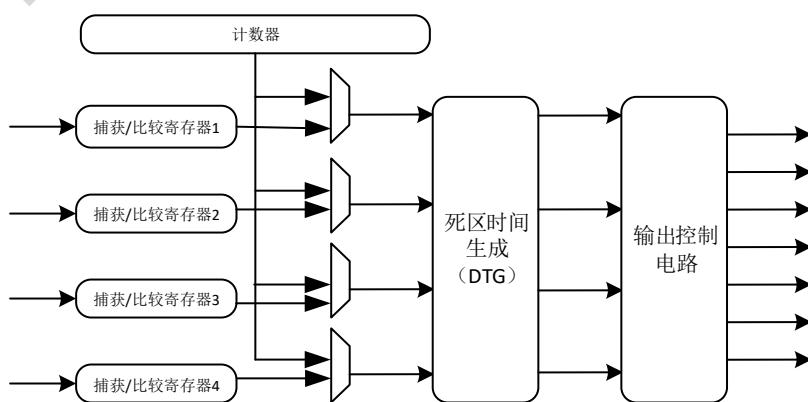


图 12-27 捕获 / 比较的输出部分

在输出比较模式下，捕获比较寄存器的内容被载入到影子寄存器中，然后影子寄存器的内容和计数器当前值进行比较。捕获/比较模块包括一个捕获/比较寄存器（预装载寄存器）和一个影子寄存器，读写过程仅操作捕获/比较寄存器。

### 12.6.3.1 强制输出

配置  $\text{TIM1\_CCMRx}$  寄存器中  $\text{CCxS} = 00$ ，将通道  $\text{CCx}$  设置为输出模式，通过配置  $\text{TIM1\_CCMRx}$  寄存器  $\text{OCxM}$  位，可以直接将输出比较信号直接强制为有效或无效状态，不依赖于输出比较结果。配置  $\text{TIM1\_CCMRx}$  寄存器  $\text{OCxM} = 100$ ，强置输出比较信号为无效状态。此时  $\text{OCxREF}$  被强置为低电平。配置  $\text{TIM1\_CCMRx}$  寄存器  $\text{OCxM} = 101$ ，强置输出比较信号为有效状态。此时  $\text{OCxREF}$  被强置为高电平（ $\text{OCxREF}$  始终为高电平有效）。

注：强制输出模式下，在  $\text{TIM1\_CCRx}$  影子寄存器和计数器之间的输出比较仍在进行，比较结果的相应标志位也会被修改，如果开启了对应的中断和 DMA 请求，仍会产生对应的中断和 DMA 请求。

### 12.6.3.2 输出比较

输出比较模式下，当计数器与捕获比较寄存器值相同时，可以表示计时结束，也可以根据  $\text{TIM1\_CCMRx}$  寄存器的  $\text{OCxM}$  位的配置用来输出不同的波形。

例如，当计数器与捕获/比较寄存器的内容匹配时，输出比较模式下的操作如下：

- 1) 在比较匹配时， $\text{OCxM}$  的值不同，输出通道  $x$  信号  $\text{OCx}$  的操作不同：
  - a)  $\text{OCxM} = 000$ :  $\text{OCx}$  信号的保持它的电平。
  - b)  $\text{OCxM} = 001$ :  $\text{OCx}$  信号被设置成有效电平。
  - c)  $\text{OCxM} = 010$ :  $\text{OCx}$  信号被设置成无效电平。
  - d)  $\text{OCxM} = 011$ :  $\text{OCx}$  信号进行翻转。
- 2) 匹配时设置中断状态寄存器中的标志位（ $\text{TIM1\_SR}$  寄存器中的  $\text{CCxIF}$  位）。
- 3) 当配置了  $\text{TIM1\_DIER}$  寄存器中的  $\text{CCxIE} = 1$ ，匹配时则产生一个中断。
- 4) 当配置了  $\text{TIM1\_DIER}$  寄存器中的  $\text{CCxDE} = 1$ ，匹配时则产生一个 DMA 请求。

输出比较模式也可以用来输出一个单脉冲（单脉冲输出模式）。

通道 1 的输出比较模式的配置步骤如下：

- 1) 配置计数器的时钟（选择时钟源，配置预分频系数）。
- 2) 配置  $\text{TIM1\_ARR}$  和  $\text{TIM1\_CCR1}$  寄存器。
- 3) 配置  $\text{TIM1\_DIER}$  寄存器的  $\text{CC1IE} = 1$ ，使能捕获/比较 1 中断。
- 4) 配置输出模式：
  - a) 配置  $\text{OC1M} = 011$ ， $\text{OC1}$  比较匹配时翻转。
  - b) 配置  $\text{OC1PE} = 0$ ，禁止  $\text{TIM1\_CCRx}$  寄存器的预装载功能。
  - c) 配置  $\text{CC1P} = 1$ ， $\text{OC1}$  低电平有效。
  - d) 配置  $\text{CC1E} = 1$ ，开启输出/比较 1 输出使能， $\text{OC1}$  信号输出到对应的输出引脚。

配置  $\text{TIM1\_CR1}$  寄存器的  $\text{CEN} = 1$ ，启动计数器。

当配置 TIM1\_CCMRx 寄存器中 OCxPE=0，禁止 TIM1\_CCRx 寄存器的预装载功能时，可以随时写入 TIM1\_CCRx 寄存器，并且写入的值立即生效。当配置 TIM1\_CCMRx 寄存器中 OCxPE=1，启用 TIM1\_CCRx 寄存器的预装载功能时，读写仅对预装载寄存器进行操作，TIM1\_CCR1 预装载寄存器的值在下次更新事件到来时生效。下图给出了一个例子。

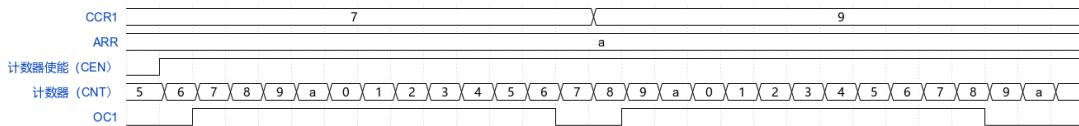


图 12-28 输出比较模式，OC1 信号在匹配时翻转

注：

- 输出比较模式下，更新事件不会对输出结果产生影响。
- 强制输出模式下，在 TIM1\_CCRx 影子寄存器和计数器之间的输出比较仍在进行，比较结果的相应标志位也会被修改，如果开启了对应的中断和 DMA 请求，仍会产生对应的中断和 DMA 请求。

### 12.6.3.3 PWM 输出

在 PWM 模式下，根据 TIM1\_ARR 寄存器和 TIM1\_CCRx 寄存器的值，产生一个频率、占空比可控的 PWM 波形。

配置与通道 x 对应的 TIM1\_CCMRx 寄存器的 OCxM=110 或 OCxM=111，选择通道 x 进入 PWM 模式 1 或 PWM 模式 2。PWM 模式下，计数器和 CCR 会一直进行比较，根据配置和比较结果，通道 x 输出不同的信号，因此 TIM1 可以产生 4 个同频率下独立占空比的 PWM 输出信号。PWM 模式下必须开启 TIM1\_CCRx 的预装载功能和 TIM1\_ARR 寄存器的预装载功能。写入 TIM1\_CCR1 预装载寄存器和 TIM1\_ARR 预装载寄存器的值在发生下个更新事件时，才会生效，载入相应的影子寄存器。所以 PWM 模式下，使能计数器前要先初始化所有的寄存器，也可以设置 UG=1，产生更新事件。

配置 TIM1\_CCER 寄存器的 CCxP 选择 OCx 的有效极性。配置 TIM1\_CCER 寄存器的 CCxE、CCxNE 位和 TIM1\_BDTR 寄存器的 MOE、OSSI、OSSR 位控制 OCx 的输出使能。配置 TIM1\_CR1 寄存器的 CMS 位，可以选择产生边沿对齐或中央对齐的 PWM 信号。

- 1) CMS=00，边沿对齐模式，再进一步配置 DIR，选择计数模式（递增或递减计数模式）。
- 2) CMS=01，中央对齐模式 1。
- 3) CMS=10，中央对其模式 2。
- 4) CMS=11，中央对齐模式 3。

#### 12.6.3.3.1 PWM 边沿对齐模式——递增计数模式

在递增计数模式配置的基础上，配置 TIM1\_CCMRx 寄存器 CCxS=00，选择输出模式，OCxM=110，选择 PWM 模式 1，当 TIM1\_CNT < TIM1\_CCRx 时通道 x (OCxREF) 为有效电平，否则为无效电平。如果 TIM1\_CCRx 中的比较值大于自动重装载值 (TIMx\_ARR)，则 OCxREF 保持为有效电平。如果比较值为 0，则 OCxREF 保持为无效电平。图 14-29 为 CCR1=1，CCR2=2，CCR3=3，

$CCR4=b$ ,  $ARR=a$  时边沿对齐递增计数时 PWM 模式 1 的波形实例。

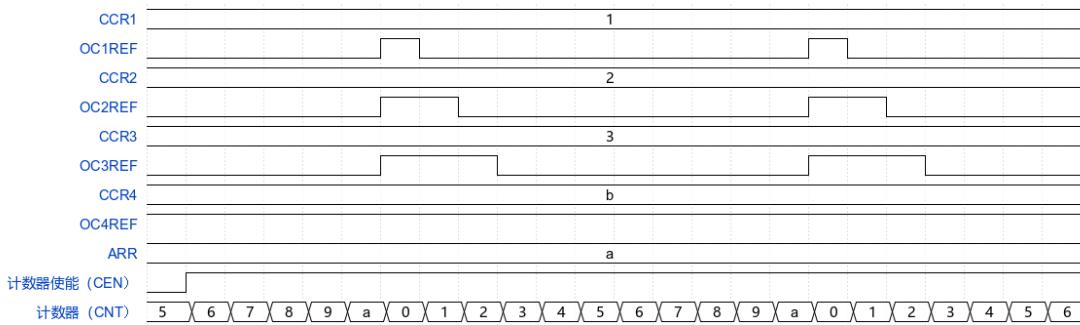


图 12-29 边沿对齐递增计数时 PWM 模式 1 的波形

### 12.6.3.3.2 PWM 边沿对齐模式——递减计数模式

在递减计数模式配置的基础上，配置 TIM1\_CCMRx 寄存器 CCxS=00，选择输出模式， $OCxM=110$ ，选择 PWM 模式 1，当  $TIM1\_CNT > TIM1\_CCRx$  时通道 x (OCxREF) 为无效电平,否则有效电平。图 14-30 为  $CCR1=6$ ,  $CCR2=4$ ,  $CCR3=9$ ,  $CCR4=b$ ,  $ARR=a$  时边沿对齐递减计数时 PWM 模式 1 的波形实例。

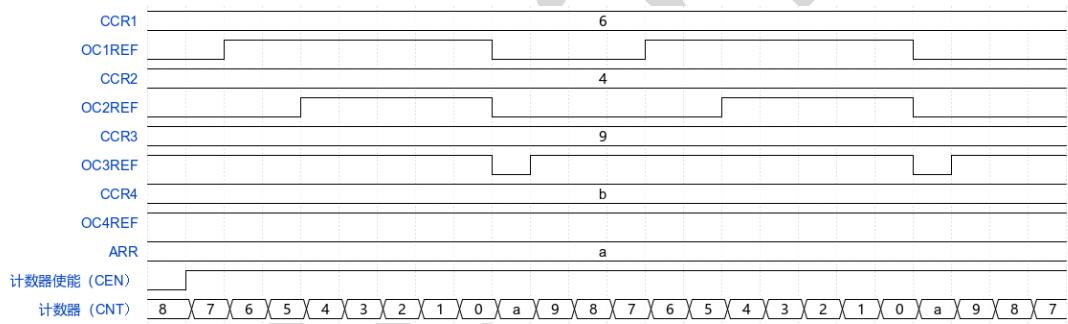


图 12-30 边沿对齐递减计数时 PWM 模式 1 的波形

### 12.6.3.3.3 PWM 中央对齐模式

首先配置 TIM1 计数器为中央对齐计数模式，配置 TIM1\_CCMRx 寄存器 CCxS=00，选择输出模式，根据配置不同的 CMS，输出比较中断标志位在计数器递增计数时被设置 (CMS=01)、在计数器递减计数时被设置 (CMS=10)、或在计数器递增或递减计数时被设置 (CMS=11)。图 14-31 为  $CCR1=6$ ,  $CCR2=4$ ,  $CCR3=9$ ,  $CCR4=b$ ,  $ARR=a$  时中央对齐 PWM 模式 1 (CMS=11) 的波形实例。

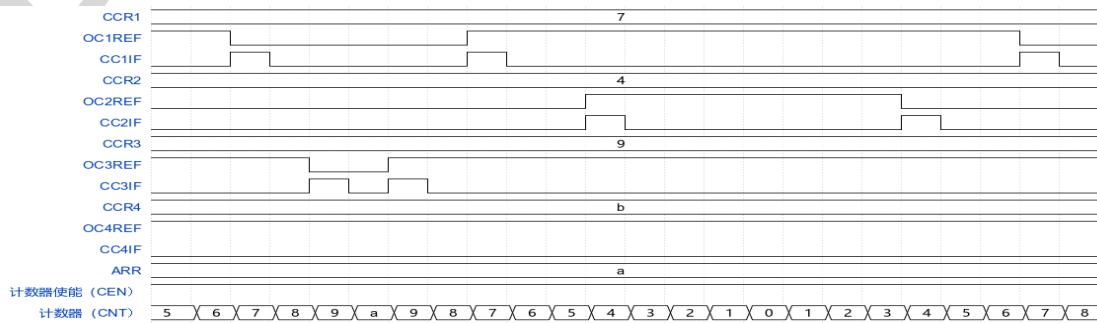


图 12-31 中央对齐 PWM 模式 1 的波形 (CMS=11)

#### 12.6.3.3.4 PWM 中央对齐模式下移相功能

新增 PDER（通道 x 输出 PWM 移相使能位）和 CCRxFALL（通道 x 在 PWM 中央对齐模式向下计数时的捕获/比较值），允许 5 个通道输出 PWM 移相。开启 PDER 寄存器的 PWM 移相使能，根据需要移动相位，配置 CCRxFALL 以及 CCRx，即可实现 PWM 输出可编程的移相波形，可左移或是右移。

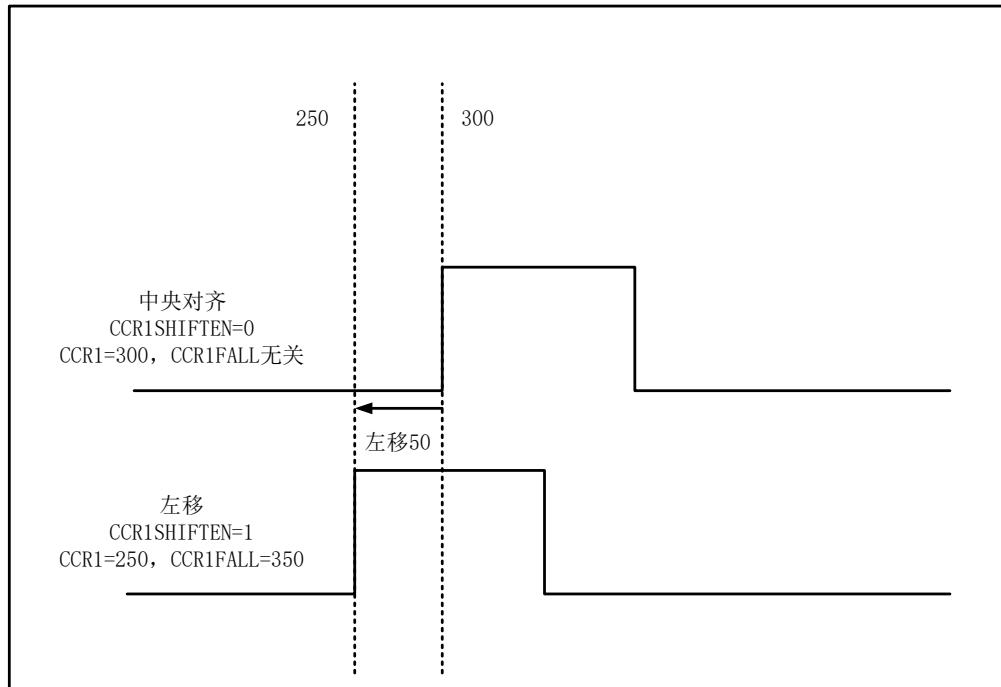


图 12-32 移相功能示意图

注：

- 进入中央对齐模式时，使用当前的递增/递减计数配置，计数方向取决于当前的 DIR 的值。
- 在中央对齐模式下，最好不要修改计数器的值，可能会产生不可预知的结果。当计数器处于递增计数时，写入计数器的值 > TIM1\_ARR，计数器会继续递增计数。直接写入 0 或 ARR，会立即更新计数方向，但是在配置允许时，不会产生更新事件。
- 建议使用中央对齐模式时，在启动计数器之前配置 TIM1\_EGR 位中的 UG=1，产生一个软件更新，更新所有寄存器，启动计数器后不要修改计数器的值。

#### 12.6.3.3.5 六步 PWM 输出

通过配置 OCxM 选择输出模式，CCxE=1 和 CCxNE=1 打开通道 x 和互补通道的输出使能，可以在通道 x 产生互补输出，这几个功能位为预装载位，在发生 COM 换相事件时，这些预装载位被装载到对应的影子寄存器中。这样可以在写入这些位时不会影响现在的输出，并可以同时载入所有通道配置。配置 TIM1\_EGR 寄存器的 COMG=1 或在 TRGI 上升沿都可以产生 COM 事件。

发生 COM 事件时，COM 中断标记会被硬件置 1；当配置了 TIM1\_DIER 寄存器 COMIE=1 和 COMDE = 1，发生 COM 事件会产生一个 COM 中断和一个 DMA 请求。

下图显示当发生 COM 事件时，不同配置下 OCx 和 OCxN 输出。

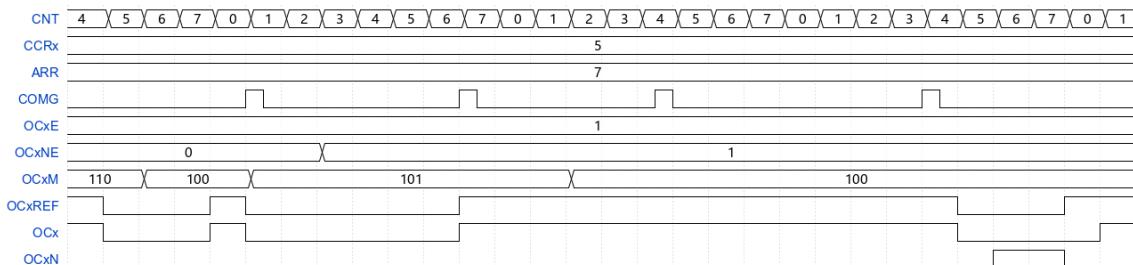


图 12-33 产生六步 PWM，使用 COM 的例子（OSSR = 1）

#### 12.6.3.4 互补输出和死区插入

OCx 和 OCxN 是一对互补输出通道，TIM1 的通道 1/2/3 能够输出三路可以管理瞬时关断和瞬时接通的互补信号。瞬时关断/开启的时间称之为死区，用户根据连接的输出器件和它们的特性（电平转换的延时、电源开关的延时等）来调整死区时间。

TIM1\_BDTR 寄存器 DTG[7:0]位定义了插入互补输出之间的死区持续时间，具体计算方式如下表：

表 12-3 死区时间计算

DTG[7:5]	DT
0xx	$DT=DTG[7:0]\times Tdtg(Tdtg=TDTS)$
10x	$DT=(64+DTG[5:0])\times Tdtg(Tdtg=2\times TDTS)$
110	$DT=32+DTG[4:0]\times Tdtg(Tdtg=8\times TDTS)$
111	$DT=(32+DTG[5:0])\times Tdtg \quad (Tdtg=16\times TDTS)$

例如，如果  $TDTS=125\text{ns}$ ，可能的死区时间为：

- 1) 若步长时间为  $125\text{ns}$ ，死区时间为 0 至  $15875\text{ns}$ 。
- 2) 若步长时间为  $250\text{ns}$ ，死区时间为  $16\mu\text{s}$  至  $31750\text{ns}$ 。
- 3) 若步长时间为  $1\mu\text{s}$ ，死区时间为  $32\mu\text{s}$  至  $63\mu\text{s}$ 。
- 4) 若步长时间为  $2\mu\text{s}$ ，死区时间为  $64\mu\text{s}$  至  $126\mu\text{s}$ 。

当不存在刹车电路时，同时配置 CCxE=1 和 CCxNE=1，开启死区插入，否则还需要配置 MOE=1。

配置 TIM1\_CCER 寄存器中的 CCxP 和 CCxNP 位，可以为每一个输出独立地选择极性（主输出 OCx 或互补输出 OCxN）。

通过配置 TIM1\_CCER 寄存器的 CCxE 和 CCxNE 位，TIM1\_BDTR 和 TIM1\_CR2 寄存器中的 MOE、OISx、OISxN、OSSI 和 OSSR 位的不同组合可以控制互补信号 OCx 和 OCxN 的输出。具体的组合控制配置见表 14-4、表 14-5、表 14-6 和表 14-7 的互补输出通道 OCx 和 OCxN 的控制位。

例：OCx 和 OCxN 都为高有效，PWM 模式下，发生匹配时，输出参考信号 OCxREF 信号翻转，输出信号 OCx 与参考信号相同，但是 OCx 信号的下降沿对于参考信号的下降沿有一个延时；互补输出信号 OCxN 与参考信号相反，但是 OCxN 信号的上升沿对于参考信号的下降沿有一个延时。

注：死区时间不能大于或等于 OCx 或 OCxN 信号的占空比，否则 OCx 或 OCxN 信号一直为无效

值。

下列几张图显示了死区发生器的输出信号和当前参考信号 OCxREF 之间的关系。

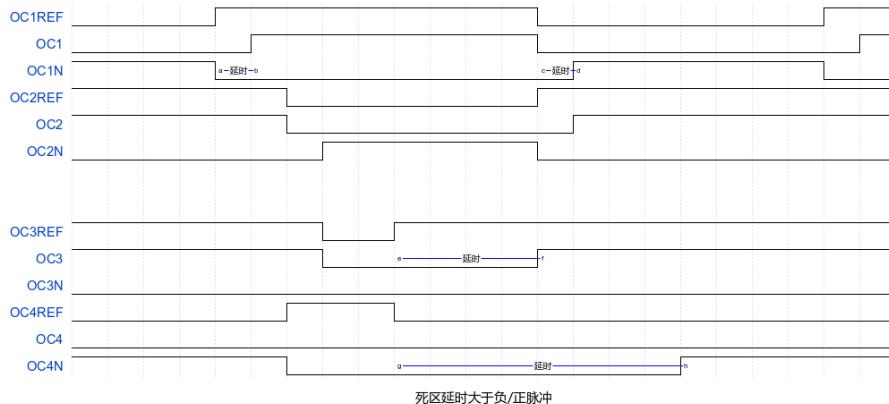


图 12-34 死区插入

### 12.6.3.5 刹车功能

刹车输入管脚和时钟失效事件都可以作为 TIM1 的刹车源。时钟失效事件由复位时钟控制器中的时钟安全系统产生。

使用刹车功能时，OCx 和 OCxN 输出信号电平被以下功能位组合控制：TIM1\_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位，TIM1\_CR2 寄存器中的 OISx 和 OISxN 位。发生刹车事件时，OCx 和 OCxN 输出不能同时有效。具体的带刹车功能的互补输出通道 OCx 和 OCxN 的输出状态如下列表所示。

表 12-4 当 MOE=1, OSSI=0/1, OSSR=0 时：

CCxE	CCxNE	OCx	OCxN
0	0	OCx=0, OCx_EN=0	OCxN=0, OCxN_EN=0
0	1	OCx=0, OCx_EN=0	OCxN=OCxREF+Polarity, OCxN_EN=1
1	0	OCx=OCxREF+Polarity, OCx_EN=1	OCxN=0, OCxN_EN=0
1	1	OCx=OCxREF+Polarity+ 死区时间 , OCx_EN=1	OCxN=OCxREF 反相+Polarity+死区时间, OCxN_EN=1

表 12-5 当 MOE=1, OSSI=0/1, OSSR=1 时：

CCxE	CCxNE	OCx	OCxN
0	0	OCx=0, OCx_EN=1	OCxN=0, OCxN_EN=1
0	1	OCx=CCxP, OCx_EN=1	OCxN=OCxREF+Polarity, OCxN_EN=1
1	0	OCx=OCxREF+Polarity, OCx_EN=1	OCxN=CCxNP, OCxN_EN=0
1	1	OCx=OCxREF+Polarity+ 死区时间 , OCx_EN=1	OCxN=OCxREF 反相+Polarity+死区时间, OCxN_EN=1

表 12-6 当 MOE=0, OSSI=0, OSSR=0/1 时：

CCxE	CCxNE	OCx	OCxN
0	0	OCx_EN=0, OCxN_EN=0	
0	1	异步的: OCx=CCxP, OCxN=CCxNP	
1	0	若时钟存在: 经过一个死区时间后, OCx=OISx, OCxN=OISxN	
1	1	OISx 和 OISxN 都不对应 OCx 和 OCxN 的有效电平	

表 12-7 当 MOE=0, OSSI=0, OSSR=0/1 时:

CCxE	CCxNE	OCx	OCxN
0	0	OCx_EN=1, OCxN_EN=1	
0	1	异步的: OCx=CCxP, OCxN=CCxNP	
1	0	若时钟存在: 经过一个死区时间后, OCx=OISx, OCxN=OISxN	
1	1	OISx 和 OISxN 都不对应 OCx 和 OCxN 的有效电平	

注: 当通道的输出和互补输出都关闭时, OISx, OISxN, CCxP 和 CCxNP 都必须配置为 0。

系统复位后, MOE=0, 刹车功能禁止, 需要配置 TIM1\_BDTR 寄存器的 BKE=1, 使能刹车功能信号。配置 TIM1\_BDTR 寄存器的 BKP 位选择刹车输入信号的极性。BKP 和 BKE 可以同时写入, 且会在一个时钟周期后生效。

由于 MOE 被异步清除, 因此在实际信号和同步控制位间插入了一个再同步电路, 用于在同步信号和异步信号见产生延迟 (当它为低时写 MOE=1, 必须在读它之前输入一个空指令用于延时)。

发生刹车事件时, MOE 被异步清除, 此时根据 OSSI 的配置输出将置于无效状态、空闲状态和复位状态; MOE=0 时, 输出由 TIM1\_CR2 寄存器的 OISx 位决定, OSSI=0 时, 定时器关闭输出使能, 否则打开输出使能时钟; 当使用互补输出时, 输出首先置于复位状态, 然后死区重新生成, 在死区之后输出电平由 OISx 和 OISxN 决定。

配置 TIM1\_BDTR 寄存器的 BIE=1, 当发生刹车事件时, 产生一个刹车中断; 如果发生刹车事件时, 配置了 TIM1\_DIER 寄存器的 BDE=1, 则产生一个刹车 DMA 请求。配置了 TIM1\_BDTR 寄存器 AOE = 1, 则在下一个更新事件到来时自动置位 MOE 位。

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

刹车电路中实现了写保护以保证应用程序的安全, 允许用户锁住死区长度, OCx/OCxN 极性和被禁止的状态, OCxM 配置, 刹车使能和极性等参数。通过 TIM1\_BDTR 寄存器中的 LOCK 位, 可以选择 lock 等级 (总共三级 lock)。Lock 在系统复位后只能修改一次。

下图显示响应刹车的输出实例:

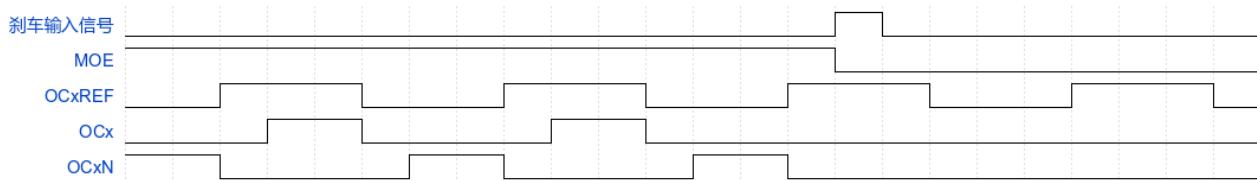


图 12-35 响应刹车的输出 ( $OISx=0, OISxN=0$ )

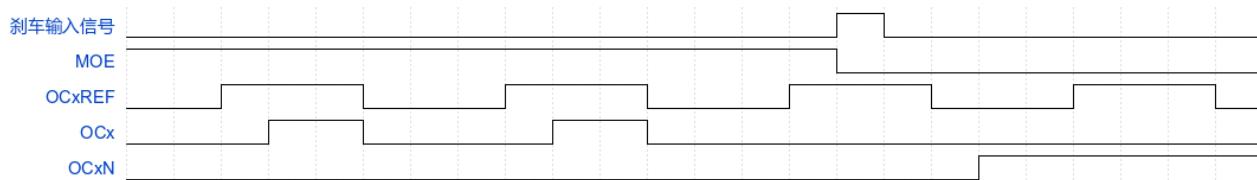


图 12-36 响应刹车的输出 ( $OISx=0, OISxN=1$ )

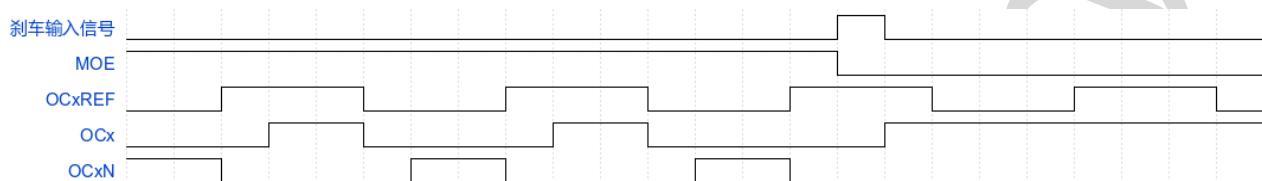


图 12-37 响应刹车的输出 ( $OISx=1, OISxN=0$ )

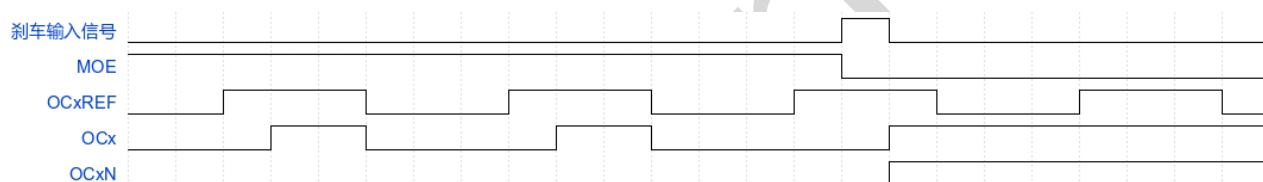


图 12-38 响应刹车的输出 ( $OISx=1, OISxN=1$ )

### 12.6.3.6 外部事件清除 OCxREF

在配置 TIM1\_CCMR 寄存器的 OCxCE=1 时，OCxREF 可以被 ETRF 输入端的高电平拉低直到发生下一次更新事件 (UEV)。此功能只能用于输出模式和 PWM 模式，不能用于强制模式。

例，OCxREF 信号连到一个外部输入时，ETR 配置如下：

- 1) 配置 TIM1\_SMCR 寄存器的 ETPS[1:0]=00，关闭外部触发预分频。
- 2) 配置 TIM1\_SMCR 寄存器 ECE=0，禁用外部时钟模式 2。
- 3) 配置 TIM1\_SMCR 寄存器 ETF[3:0] 和 ETP，配置 ETR 信号的触发极性和滤波宽度。

下图显示了当 ETRF 输入变为高时，对应不同 OCxCE 的值，OCxREF 信号的动作 (PWM 模式)。

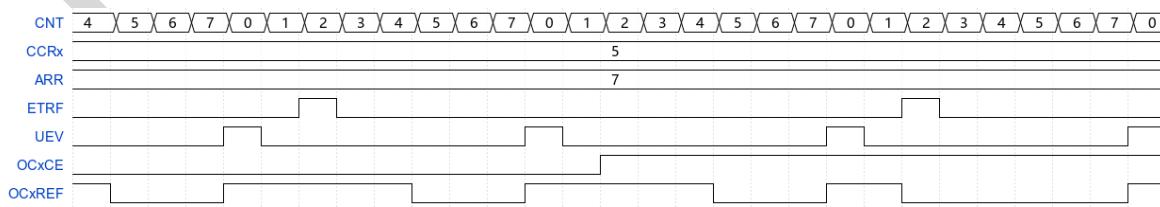


图 12-39 清除 TIM1 的 OCxREF

### 12.6.3.7 单脉冲输出

单脉冲模式 (OPM) 是计数器只响应一个激励，延时后产生一个脉宽可调的脉冲。配置

TIM1\_CR1 寄存器的 OPM=1，选择单脉冲模式，触发信号有效沿或配置 CEN=1 都可以启动计数器，CEN=1 一直保持到下个更新事件发生或配置 CEN=0。

产生脉冲的必要条件是比较值与计数器的初始值不同。所以在计数器启动之前的必要配置如下：

- 1) 递增计数方式：计数器  $CNT < CCRx \leq ARR$ 。
- 2) 递减计数方式：计数器  $CNT > CCRx$ 。

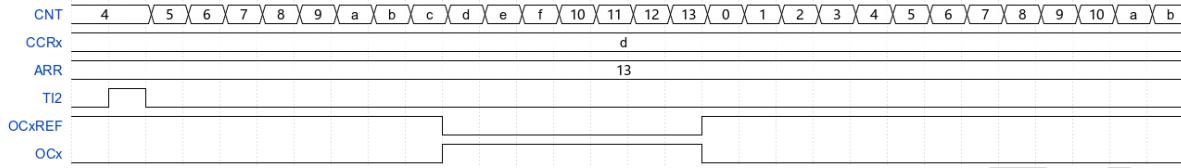


图 12-40 单脉冲模式

例如，在 TI2 检测到上升沿，延迟  $t_{DELAY}$  之后，在 OC2 上产生一个长度为  $t_{PULSE}$  的正脉冲。

假定 TI2FP2 作为触发 1：

- 1) 配置 TIM1\_CCMR1 寄存器中的 CC2S = 01，将 TI2FP2 映射到 TI2。
- 2) 配置 TIM1\_CCER 寄存器中的 CC2P = 0，检测 TI2FP2 的上升沿。
- 3) 配置 TIM1\_SMCR 寄存器中的 TS = 110，TI2FP2 作为从模式控制器的触发 (TRGI)。
- 4) 配置 TIM1\_SMCR 寄存器中的 SMS = 110，选择触发模式，TI2FP2 使能计数器工作。

OPM 的波形由 TIM1\_ARR 和 TIM1\_CCR1 决定（要考虑时钟频率和计数器预分频器）：由 TIM1\_CCR2 寄存器的值和 CNT 初始值决定  $t_{DELAY}$ ；TIM1\_ARR - TIM1\_CCR1 的值为  $t_{PULSE}$ 。

当发生比较匹配时要产生从 1 到 0 的波形，当计数器达到预装载值时要产生一个从 0 到 1 的波形：

- 1) 配置 TIM1\_CCMR1 寄存器 OC1M = 111，选择 PWM 模式 2。
- 2) 配置 TIM1\_CCER 寄存器 CC2P = 1，输出低电平有效。
- 3) 配置 TIM1\_CCMR1 中 OC1PE = 1 和 TIM1\_CR1 寄存器中 ARPE，使能预装载寄存器。
- 4) 配置 TIM1\_CCR1 寄存器和 TIM1\_ARR 寄存器。
- 5) 配置 TIM1\_EGR 寄存器 UG=1 产生一个更新事件。
- 6) 等待在 TI2 上的一个外部触发事件。

此例中，TIM1\_CR1 寄存器中的 DIR=0、CMS=0、OPM=1，在下一个更新事件（当计数器从自动装载值翻转到 0）时停止计数。

### 12.6.3.7.1 OCx 快速使能

OCx 快速使能，是单脉冲模式的一种特殊情况。在单脉冲模式下，通过设置 TIM1\_CCMR 寄存器的 OCxFE=1，强制 OCxREF 直接响应激励而不是依赖计数器和比较值之间的比较结果，输出波形和比较匹配时的波形一样。这样可以去除比较的时间，快速输出比较结果。OCx 快速输出使能只在 PWM 模式下生效。

## 12.6.4 DMA 模式

TIM1 能够在发生单个事件时生成多个 DMA 请求。主要目的是在没有软件开销的情况下，多次重新编程 TIM1 的一部分也可以用于按周期读取数个寄存器。

TIM1\_DCR 和 TIM1\_DMAR 寄存器跟 DMA 模式相关。DMA 控制器的目标是唯一的，必须指向 TIM1\_DMAR 寄存器。开启 DMA 使能后，在给定的 TIM1 事件发生时，TIM1 会给 DMA 发送请求。对 TIM1\_DMAR 寄存器的每次写操作都被重定向到一个 TIM1 寄存器。

TIM1\_DCR 寄存器的 DBL 位定义了 DMA 连续传送的长度，即传输寄存器数量；当对 TIM1\_DMAR 进行读写操作时，定时器识别 DBL，确定传输的寄存器数量。TIM1\_DCR 寄存器的 DBA 位定义了 DMA 传输的基址，定义从 TIM1\_CR1 寄存器地址开始的偏移量（00000 为 TIM1\_CR1；00001 为 TIM1\_CR2；...；00110 为 TIM1\_CCMR1 等）。

例：DMA 模式用于在发生更新事件时更新 CCR1、CCR2、CCR3 寄存器的内容。具体配置如下：

- 1) 配置相应的 DMA 通道。
- 2) 配置 TIM1\_DCR 寄存器 DBA=01101，配置 DMA 的基地址，选择偏移地址为 TIM1\_CCR1 寄存器的地址。
- 3) 配置 TIM1\_DCR 寄存器 DBL=00010，配置传输长度为 3。
- 4) 配置 TIM1\_DIER 寄存器 UDE=1，允许更新事件的 DMA 请求。
- 5) 配置 TIM1\_CR1 寄存器 CEN=1，启动计数器。
- 6) 使能 DMA 通道。

此例中 TIM1 的 CCR1、CCR2、CCR3 更新一次，在第一个 DMA 请求时，数据 1 传到 CCR1 上；在第二个 DMA 请求时，数据 2 传到 CCR2 上；在第三个 DMA 请求时，数据 3 传到 CCR3 上。

## 12.6.5 调试模式

配置 DBG\_CR 寄存器中 DBG\_TIM1\_STOP=1，TIM1 计数器在 CPU 内核停止工作时（调试设置的断点）停止计数。（详见调试章节）

## 12.7 寄存器描述

表 12-8 TIM1 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	TIM1_CR1	控制寄存器 1	0x0000
0x04	TIM1_CR2	控制寄存器 2	0x0000
0x08	TIM1_SMCR	从模式控制寄存器	0x0000
0x0C	TIM1_DIER	DMA/中断使能寄存器	0x0000
0x10	TIM1_SR	状态寄存器	0x0000
0x14	TIM1_EGR	事件产生寄存器	0x0000
0x18	TIM1_CCMR1	捕获/比较模式寄存器 1	0x0000

Offset	Acronym	Register Name	Reset
0x1C	TIM1_CCMR2	捕获/比较模式寄存器 2	0x0000
0x20	TIM1_CCER	捕获/比较使能寄存器	0x0000
0x24	TIM1_CNT	计数器	0x0000
0x28	TIM1_PSC	预分频率器	0x0000
0x2C	TIM1_ARR	自动装载寄存器	0x0000
0x30	TIM1_RCR	重复计数寄存器	0x0000
0x34	TIM1_CCR1	捕获/比较寄存器 1	0x0000
0x38	TIM1_CCR2	捕获/比较寄存器 2	0x0000
0x3C	TIM1_CCR3	捕获/比较寄存器 3	0x0000
0x40	TIM1_CCR4	捕获/比较寄存器 4	0x0000
0x44	TIM1_BDTR	刹车和死区寄存器	0x0000
0x48	TIM1_DCR	DMA 控制寄存器	0x0000
0x4C	TIM1_DMAR	连续模式的 DMA 地址	0x0000
0x54	TIM1_CCMR3	捕获/比较模式寄存器 3	0x0000
0x58	TIM1_CCR5	捕获/比较寄存器 5	0x0000
0x5C	TIM1_PDER	PWM 移相/DMA repeat 更新请求使能寄存器	0x0000
0x60 ~ 0x70	CCRx FALL	PWM 移相向下计数捕获/比较寄存器	0x0000

### 12.7.1 控制寄存器 1 (TIM1\_CR1)

偏移地址: 0x0

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.						CKD		APRE	CMS		DIR	OPM	URS	UDIS	CEN
Type							rw		rw	rw		rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:10	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
9:8	CKD	rw	0x0	<p>时钟分频 (clock division)</p> <p>在定时器时钟 (INT_CK) 频率、死区时间和由死区发生器与数字滤波器 (ETR, TIx) 所用的采样时钟之间的分频比例。</p> <p>00: tDTS = tINT_CK 01: tDTS = 2x tINT_CK 10: tDTS = 4x tINT_CK 11: 保留, 不要使用这个配置</p>
7	APRE	rw	0x0	<p>自动重装载预装载使能 (Auto-reload preload enable)</p> <p>0: 关闭 TIM1_ARR 寄存器的影子寄存器 1: 使能 TIM1_ARR 寄存器的影子寄存器</p>
6:5	CMS	rw	0x0	<p>中央对齐模式选择 (Center-aligned mode selection)</p> <p>00: 边沿对齐模式。计数方向取决于 DIR 位 01: 中央对齐模式 1。计数器交替地递增和递减计数。通道为输出模式, 只在计数器递减计数时比较中断标志位被置 1 10: 中央对齐模式 2。计数器交替地递增和递减计数。通道为输出模式, 只在计数器递增计数时比较中断标志位被置 1 11: 中央对齐模式 3。计数器交替地递增和递减计数。通道为输出模式, 在计数器递增和递减计数时比较中断标志位均被置 1 注: 计数过程中, 不允许对齐模式的更改。</p>
4	DIR	rw	0x0	<p>计数方向 (Direction)</p> <p>0: 计数器递增计数 1: 计数器递减计数 注: 当计数器配置为中央对齐模式或编码器模式时, 该位为只读。</p>
3	OPM	rw	0x0	<p>单脉冲模式 (One pulse mode)</p> <p>0: 禁止单脉冲模式, 在发生更新事件时, 计数器继续计数 1: 使能单脉冲模式, 在发生下一次更新事件 (清除 CEN 位) 时, 计数器停止计数</p>

Bit	Field	Type	Reset	Description
2	URS	rw	0x0	<p>更新请求源 (Update request source) 软件配置该位，选择更新事件源。</p> <p>0: 以下事件可产生一个更新中断或 DMA 请求:</p> <ul style="list-style-type: none"> <li>- 计数器溢出/下溢</li> <li>- 设置 UG 位</li> <li>- 从模式控制器产生的更新</li> </ul> <p>1: 只有计数器溢出/下溢才产生一个更新中断或 DMA 请求</p>
1	UDIS	rw	0x0	<p>禁止更新 (Update disable) 该位用来允许或禁止更新事件的产生</p> <p>0: 允许更新事件 (UEV)。更新事件源:</p> <ul style="list-style-type: none"> <li>- 计数器溢出/下溢</li> <li>- 设置 UG 位为 1</li> <li>- 从模式控制器产生一个更新事件。</li> </ul> <p>1: 禁止更新事件。不产生更新事件，影子寄存器 (ARR、PSC、CCRx) 保持值不变。如果设置了 EGR_UG 位为 1，或者从模式控制器接收到硬件复位，计数器和预分频器被更新为其对应的影子寄存器的值。</p>
0	CEN	rw	0x0	<p>允许计数器 (Counter enable) 0: 禁止计数器 1: 使能计数器。 注: 在软件设置了 CEN 位后, 外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。</p>

## 12.7.2 控制寄存器 2 (TIM1\_CR2)

偏移地址: 0x04

复位值: 0x0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	T11S	MMS			CCDS	CCUS	Res.	CCPC
Type		rw	rw	rw	rw	rw	rw	rw	rw	rw			rw	rw		rw

Bit	Field	Type	Reset	Description
31:15	Reserved			保留, 始终读为 0
14	OIS4	rw	0x0	输出空闲状态 4 (OC4 输出)。参见 OIS1 位。
13	OIS3N	rw	0x0	输出空闲状态 3 (OC3N 输出)。参见 OIS1N 位。
12	OIS3	rw	0x0	输出空闲状态 3 (OC3 输出)。参见 OIS1 位。
11	OIS2N	rw	0x0	输出空闲状态 2 (OC2N 输出)。参见 OIS1N 位。
10	OIS2	rw	0x0	输出空闲状态 2 (OC2 输出)。参见 OIS1 位。
9	OIS1N	rw	0x0	<p>输出空闲状态 1 (OC1N 输出) (Output Idle state 1)</p> <p>0: 当 MOE = 0 时, 死区后 OC1N = 0</p> <p>1: 当 MOE = 0 时, 死区后 OC1N = 1</p> <p>注: 已经设置了 LOCK (TIM1_BKR 寄存器) 级别 1、2 或 3 后, 该位不能被修改。</p>
8	OIS1	rw	0x0	<p>输出空闲状态 1 (OC1 输出) (Output Idle state 1)</p> <p>0: 当 MOE=0, OC1N=1 时则在死区时间后 OC1 = 0</p> <p>1: 当 MOE=0, OC1N=1 时则在死区时间后 OC1 = 1</p> <p>注: 已经设置了 LOCK (TIM1_BKR 寄存器) 级别 1、2 或 3 后, 该位不能被修改。</p>
7	TI1S	rw	0x0	<p>TI1 选择 (TI1 selection)</p> <p>0: TIM1_CH1 管脚连到 TI1 输入</p> <p>1: TIM1_CH1、TIM1_CH2 和 TIM1_CH3 管脚经异或后的结果作为 TI1 输入</p>

Bit	Field	Type	Reset	Description
6:4	MMS	rw	0x0	<p>主模式选择 (Master mode selection)</p> <p>这些位控制 TRGO 信号的选择，用于选择在主模式下送到从定时器的同步信息：</p> <p>000: 复位 TIM1_EGR 寄存器的 UG 位或从模式控制器产生复位触发一次 TRGO 脉冲。从模式控制器产生复位时，TRGO 信号相对实际的复位会有一个延时。</p> <p>001: 使能 用于控制在一定时间内使能从定时器或同时启动多个定时器。计数器使能信号 CNT_EN 被用于作为触发输出 (TRGO)，计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式。</p> <p>010: 更新 更新事件被选为 TRGO。</p> <p>011: 捕获/比较脉冲 发生一次捕获或一次比较成功时，触发输出送出一个 TRGO 信号。</p> <p>100: 比较 OC1REF 信号被用于作为触发输出 (TRGO)</p> <p>101: 比较 OC2REF 信号被用于作为触发输出 (TRGO)</p> <p>110: 比较 OC3REF 信号被用于作为触发输出 (TRGO)</p> <p>111: 比较 OC4REF 信号被用于作为触发输出 (TRGO)</p>
3	CCDS	rw	0x0	<p>DMA 请求源选择 (Capture/compare DMA selection)</p> <p>0: 当 CCx 发生捕获/比较事件时，发送 CCx 的 DMA 请求</p> <p>1: 当 CCx 发生更新事件时，发送 CCx 的 DMA 请求</p>
2	CCUS	rw	0x0	<p>捕获/比较控制更新源选择 (Capture/compare control update selection)</p> <p>0: CCPC=1 时，只能配置 COMG=1 更新它们。</p> <p>1: CCPC=1 时，可以通过配置 COMG=1 或检测到 TRGI 上的一个上升沿更新它们。</p> <p>注：此位只在通道位互补输出时有效。</p>
1	Reserved			保留，始终读为 0

Bit	Field	Type	Reset	Description
0	CCPC	rw	0x0	<p>捕获/比较预装载控制位 ( Capture/compare preloaded control)</p> <p>0: CCxE, CCxNE 和 OCxM 位预装载禁用。</p> <p>1: CCxE, CCxNE 和 OCxM 位是预装载使能。</p> <p>注: 此位只在通道为互补输出时有效。</p>

### 12.7.3 从模式控制寄存器 (TIM1\_SMCR)

偏移地址: 0x08

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ETP	ECE	ETPS	ETF				MSM	TS	SMS						
Type	rw	rw	rw	rw				rw	rw	rw						

Bit	Field	Type	Reset	Description
15	ETP	rw	0x0	<p>外部触发极性 (External trigger polarity)</p> <p>该位选择 ETR 信号的极性。</p> <p>0: 高电平或上升沿有效</p> <p>1: 低电平或下降沿有效</p>
14	ECE	rw	0x0	<p>外部时钟使能位 (External clock enable)</p> <p>该位启用外部时钟模式 2。</p> <p>0: 禁止外部时钟模式 2</p> <p>1: 使能外部时钟模式 2, ETRF 信号上的任意有效沿驱动计数器计数。</p> <p>注 1: 配置 ECE=1 与配置 SMS = 111 和 TS = 111 效果一样。</p> <p>注 2: TS ≠ 111 时, 复位模式, 门控模式和触发模式可以与外部时钟模式 2 同时使用。</p> <p>注 3: 同时使能外部时钟模式 1 和外部时钟模式 2 时, 外部时钟的输入是 ETRF。</p>

Bit	Field	Type	Reset	Description
13: 12	ETPS	rw	0x0	<p>外部触发预分频 (External trigger prescaler)</p> <p>外部触发信号 ETRP 的频率必须低于 TIM1CLK 频率的 1/4。当输入较快的外部时钟时，可以使用预分频降低 ETRP 的频率。</p> <p>00: 关闭预分频</p> <p>01: ETRP 频率除以 2</p> <p>10: ETRP 频率除以 4</p> <p>11: ETRP 频率除以 8</p>
11: 8	ETF	rw	0x0	<p>外部触发滤波 (External trigger filter)</p> <p>这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。</p> <p>0000: 无滤波器，以 <math>f_{DTS}</math> 采样</p> <p>0001: 采样频率 <math>f_{sampling} = f_{INT\_CK}, N=2</math></p> <p>0010: 采样频率 <math>f_{sampling} = f_{INT\_CK}, N=4</math></p> <p>0011: 采样频率 <math>f_{sampling} = f_{INT\_CK}, N=8</math></p> <p>0100: 采样频率 <math>f_{sampling} = f_{DTS}/2, N=6</math></p> <p>0101: 采样频率 <math>f_{sampling} = f_{DTS}/2, N=8</math></p> <p>0110: 采样频率 <math>f_{sampling} = f_{DTS}/4, N=6</math></p> <p>0111: 采样频率 <math>f_{sampling} = f_{DTS}/4, N=8</math></p> <p>1000: 采样频率 <math>f_{sampling} = f_{DTS}/8, N=6</math></p> <p>1001: 采样频率 <math>f_{sampling} = f_{DTS}/8, N=8</math></p> <p>1010: 采样频率 <math>f_{sampling} = f_{DTS}/16, N=5</math></p> <p>1011: 采样频率 <math>f_{sampling} = f_{DTS}/16, N=6</math></p> <p>1100: 采样频率 <math>f_{sampling} = f_{DTS}/16, N=8</math></p> <p>1101: 采样频率 <math>f_{sampling} = f_{DTS}/32, N=5</math></p> <p>1110: 采样频率 <math>f_{sampling} = f_{DTS}/32, N=6</math></p> <p>1111: 采样频率 <math>f_{sampling} = f_{DTS}/32, N=8</math></p>

Bit	Field	Type	Reset	Description
7	MSM	rw	0x0	<p>主/从 模 式 (Master/Slave mode)</p> <p>0: 无作用</p> <p>1: 触发输入 (TRGI) 上的事件被延迟了, 以允许在当前定时器 (通过 TRGO) 与它的从定时器间的完美同步, 这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。</p>
6: 4	TS	rw	0x0	<p>触发选择 (Trigger selection)</p> <p>触发输入源选择。</p> <p>000: 内部触发 0 (ITR0)</p> <p>001: 内部触发 1 (ITR1)</p> <p>010: 内部触发 2 (ITR2)</p> <p>011: 内部触发 3 (ITR3)</p> <p>100: TI1 的边沿检测器 (TI1F_ED)</p> <p>101: 滤波后的定时器输入 1 (TI1FP1)</p> <p>110: 滤波后的定时器输入 2 (TI2FP2)</p> <p>111: 外部触发输入 (ETRF)</p> <p>更多有关 ITRx 的细节, 参见下表。</p> <p>注: 从模式使能后这些位不能修改</p>
3	OCCS	rw	0x0	<p>比较器输出信号清除选择 (Output compare clear selection)</p> <p>在 PWM 模式下, 清除比较器输出</p> <p>0: 外部触发信号作为清除信号</p> <p>1: 比较器输出作为清除信号</p>

Bit	Field	Type	Reset	Description
2: 0	SMS	rw	0x0	<p>从模式选择 (Slave mode selection)</p> <p>当选择了外部信号, 触发信号 (TRGI) 的有效边沿与选中的外部输入极性相关。</p> <p>000: 关闭从模式 - 如果 CEN = 1, 则预分频器直接由内部时钟驱动。</p> <p>001: 编码器模式 1- 根据 TI1FP1 的电平, 计数器在 TI2FP2 的边沿向递增/递减计数。</p> <p>010: 编码器模式 2- 根据 TI2FP2 的电平, 计数器在 TI1FP1 的边沿递增/递减计数。</p> <p>011: 编码器模式 3 - 根据另一个输入的电平, 计数器在 TI1FP1 和 TI2FP2 的边沿递增/递减计数。</p> <p>100: 复位模式 - 选中的触发输入 (TRGI) 的上升沿重新初始化计数器, 并且产生一个更新寄存器的信号。</p> <p>101: 门控模式 - 当触发输入 (TRGI) 为高时, 计数器开始计数。当触发输入变为低时, 计数器停止计数 (但不复位)。计数器的启动和停止都是受控的。</p> <p>110: 触发模式 - 计数器在触发输入 TRGI 的上升沿启动 (但不复位), 只有计数器的启动是受控的。</p> <p>111: 外部时钟模式 1 - 选中的触发输入 (TRGI) 的上升沿驱动计数器。</p> <p>注: 如果 TI1F_EN 被选为触发输入 (TS = 100) 时, 不要使用门控模式。这是因为, TI1F_EO 在每次 TI1F 变化时输出一个脉冲, 然而门控模式是要检查触发输入的电平。</p>

表 12-9 TIM1 内部触发连接

从定时器	ITR0	ITR1	ITR2	ITR3
TIM1	TIM5	TIM2	TIM3	TIM4
TIM2	TIM1	TIM8	TIM3	TIM4
TIM3	TIM1	TIM2	TIM5	TIM4
TIM4	TIM1	TIM2	TIM3	TIM8
TIM5	TIM1	TIM3	TIM4	TIM8
TIM8	TIM1	TIM2	TIM4	TIM5

## 12.7.4 DMA/中断使能寄存器 (TIM1\_DIER)

偏移地址: 0x0C

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															CC5IE
Type																rw
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	R e s .	TD E	COM DE	CC4 DE	CC3 DE	CC2 DE	CC1 DE	UD E	BI E	TI E	COM IE	CC4 IE	CC3 IE	CC2 IE	CC1 IE	UIE
Type	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:17	Reserved			保留, 始终读为 0
16	CC5IE	rw	0x0	允许比较 5 中断 (Compare 5 interrupt enable) 0: 禁止比较 5 中断 1: 允许比较 5 中断
15	Reserved			保留, 始终读为 0
14	TDE	rw	0x0	允许触发 DMA 请求 (Trigger DMA request enable) 0: 禁止触发 DMA 请求 1: 允许触发 DMA 请求
13	COMDE	rw	0x0	允许 COM 的 DMA 请求 (COM DMA request enable) 0: 禁止 COM 的 DMA 请求 1: 允许 COM 的 DMA 请求
12	CC4DE	rw	0x0	允许捕获/比较 4 的 DMA 请求 (Capture/Compare 4 DMA request enable) 0: 禁止捕获/比较 4 的 DMA 请求 1: 允许捕获/比较 4 的 DMA 请求

Bit	Field	Type	Reset	Description
11	CC3DE	rw	0x0	允许捕获/比较 3 的 DMA 请求 (Capture/Compare 3 DMA request enable) 0: 禁止捕获/比较 3 的 DMA 请求 1: 允许捕获/比较 3 的 DMA 请求
10	CC2DE	rw	0x0	允许捕获/比较 2 的 DMA 请求 (Capture/Compare 2 DMA request enable) 0: 禁止捕获/比较 2 的 DMA 请求 1: 允许捕获/比较 2 的 DMA 请求
9	CC1DE	rw	0x0	允许捕获/比较 1 的 DMA 请求 (Capture/Compare 1 DMA request enable) 0: 禁止捕获/比较 1 的 DMA 请求 1: 允许捕获/比较 1 的 DMA 请求
8	UDE	rw	0x0	允许更新 DMA 请求 (Update DMA request enable) 0: 禁止更新 DMA 请求 1: 允许更新 DMA 请求
7	BIE	rw	0x0	允许刹车中断 (Break interrupt enable) 0: 禁止刹车中断 1: 允许刹车中断
6	TIE	rw	0x0	允许触发中断 (Trigger interrupt enable) 0: 禁止触发中断 1: 允许触发中断
5	COMIE	rw	0x0	允许 COM 中断 (COM interrupt enable) 0: 禁止 COM 中断 1: 允许 COM 中断
4	CC4IE	rw	0x0	允许捕获/比较 4 中断 (Capture/Compare 4 interrupt enable) 0: 禁止捕获/比较 4 中断 1: 允许捕获/比较 4 中断
3	CC3IE	rw	0x0	允许捕获/比较 3 中断 (Capture/Compare 3 interrupt enable) 0: 禁止捕获/比较 3 中断 1: 允许捕获/比较 3 中断

Bit	Field	Type	Reset	Description
2	CC2IE	rw	0x0	允许捕获/比较 2 中断 (Capture/Compare 2 interrupt enable) 0: 禁止捕获/比较 2 中断 1: 允许捕获/比较 2 中断
1	CC1IE	rw	0x0	允许捕获/比较 1 中断 (Capture/Compare 1 interrupt enable) 0: 禁止捕获/比较 1 中断 1: 允许捕获/比较 1 中断
0	UIE	rw	0x0	允许更新中断 (Update interrupt enable) 0: 禁止更新中断 1: 允许更新中断

## 12.7.5 状态寄存器 (TIM1\_SR)

偏移地址: 0x10

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.			CC4 OF	CC3 OF	CC2 OF	CC1 OF	Res.	BI F	TI F	COM IF	CC 4IF	CC 3IF	CC2 IF	CC1 IF	UIF
Type				r_w0c					r_w0c							

Bit	Field	Type	Reset	Description
31:17	Reserved			保留, 始终读为 0
16	CC5IF	r_w0c	0x0	比较 5 中断标记 (Compare 5 interrupt flag) 参考 CC1IF 描述。
15:13	Reserved			保留, 始终读为 0
12	CC4OF	r_w0c	0x0	捕获/比较 4 重复捕获标记 (Capture/Compare 4 overcapture flag) 参考 CC1OF 描述。

Bit	Field	Type	Reset	Description
11	CC3OF	r_w0c	0x0	捕获/比较 3 重复捕获标记 (Capture/Compare 3 overcapture flag) 参考 CC1OF 描述。
10	CC2OF	r_w0c	0x0	捕获/比较 2 重复捕获标记 (Capture/Compare 2 overcapture flag) 参考 CC1OF 描述。
9	CC1OF	r_w0c	0x0	捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当通道 1 被配置为输入捕获, CC1F 已经位 1 后, 捕获事件再次发生时, 该标记可由硬件置 1。写 0 可清除该位。 0: 无重复捕获产生; 1: 重复捕获产生。
8	Reserved			保留, 始终读为 0
7	BIF	r_w0c	0x0	刹车中断标记 (Break interrupt flag) 当刹车输入有效, 由硬件对该位置"1"。如果刹车输入无效, 则该位可由软件清"0" 0: 无刹车事件产生 1: 刹车输入上检测到有效电平
6	TIF	r_w0c	0x0	触发器中断标记 (Trigger interrupt flag) 当发生触发事件 (当从模式控制器处于除门控模式外的其它模式时, 在 TRGI 输入端检测到有效边沿, 或门控模式下的任一边沿) 时由硬件对该位置 1。它由软件清 0。 0: 无触发器事件产生 1: 触发器中断产生
5	COMIF	r_w0c	0x0	COM 中断标记 (COM interrupt flag) 当产生 COM 事件 (当捕获/比较控制位: CCxE、CCxNE、OCxM 已被更新) 时该位由硬件置 1。它由软件清 0。 0: 无 COM 事件产生 1: COM 中断等待响应
4	CC4IF	r_w0c	0x0	捕获/比较 4 中断标记 (Capture/Compare 4 interrupt flag) 参考 CC1IF 描述。
3	CC3IF	r_w0c	0x0	捕获/比较 3 中断标记 (Capture/Compare 3 interrupt flag) 参考 CC1IF 描述。
2	CC2IF	r_w0c	0x0	捕获/比较 2 中断标记 (Capture/Compare 2 interrupt flag) 参考 CC1IF 描述。

Bit	Field	Type	Reset	Description
1	CC1IF	r_w0c	0x0	<p>捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag)</p> <p>通道 1 为输出模式:</p> <p>当计数器值与比较值匹配时该位由硬件置 1, 但在中心对称模式下除外。它由软件清 0。</p> <p>0: 无匹配发生 1: TIM1_CNT 的值与 TIM_CCR1 的值匹配。</p> <p>通道 1 为输入模式:</p> <p>当发生捕获事件时该位由硬件置 1, 由软件或读取 TIM1_CCR1 的值清 0。</p> <p>0: 无输入捕获产生 1: 计数器值已被捕获至 TIM1_CCR1</p>
0	UIF	r_w0c	0x0	<p>更新中断标记 (Update interrupt flag)</p> <p>当产生更新事件时该位由硬件置 1。它由软件清 0。</p> <p>0: 无更新中断发生 1: 发生更新中断。</p> <p>当寄存器被更新时该位由硬件置 1:</p> <ul style="list-style-type: none"> <li>- 若 TIM1_CR1 寄存器的 UDIS=0, 且 REP_CNT=0, 当计数器产生上溢/下溢事件时。</li> <li>- 若 TIM1_CR1 寄存器的 UDIS=0、URS=0, 当 TIM1_EGR 寄存器的 UG=1 时产生更新事件。</li> <li>- 若 TIM1_CR1 寄存器的 UDIS=0、URS=0, 当计数器被触发事件重初始化时产生更新事件。</li> </ul>

## 12.7.6 事件产生寄存器 (TIM1\_EGR)

偏移地址: 0x14

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:17	Reserved			保留, 始终读为 0
16	CC5G	w	0x0	产生比较 5 事件 (Compare 5 generation) 参考 CC1G 描述。
16:8	Reserved			保留, 始终读为 0
7	BG	w	0x0	产生刹车事件 (Break generation) 0: 无动作 1: 产生一个刹车事件, 此时 MOE=0, BIF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA, 由硬件清除。
6	TG	w	0x0	产生触发事件 (Trigger generation) 0: 无动作 1: 产生触发事件, TIM1_SR 寄存器的 TIF = 1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA, 由硬件自动清 0。
5	COMG	w	0x0	捕获/比较事件, 产生控制更新 (Capture/Compare control update generation) 0: 无动作 1: 捕获/比较事件控制更新产生, 由硬件自动清 0, 当 CCPC=1, 允许更新 CCxE、CCxNE、OCxM 位。 注: 该位只对拥有互补输出的通道有效。
4	CC4G	w	0x0	产生捕获/比较 4 事件 (Capture/Compare 4 generation) 参考 CC1G 描述。
3	CC3G	w	0x0	产生捕获/比较 3 事件 (Capture/Compare 3 generation) 参考 CC1G 描述。
2	CC2G	w	0x0	产生捕获/比较 2 事件 (Capture/Compare 2 generation) 参考 CC1G 描述。
1	CC1G	w	0x0	产生通道 1 捕获/比较事件 (Capture/Compare 1 generation) 该位由软件置 1, 用于产生一个捕获/比较事件, 由硬件自动清 0。 0: 无动作 1: 在通道 CC1 上产生一个捕获/比较事件: 若通道 CC1 配置为输出: 设置 CC1IF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。 若通道 CC1 配置为输入: 当前的计数器值被捕获至 TIM1_CCR1 寄存器, 设置 CC1IF = 1, 若开启对应的中断和 DMA, 则产生相应

				的中断和 DMA。若 CC1IF 已经为 1，则设置 CC1OF = 1。
0	UG	w	0x0	<p>产生更新事件 (Update generation)</p> <p>0: 无动作</p> <p>1: 重置计数器值，并产生一个更新事件。由硬件自动清 0，如果选择了中央对齐或递增计数模式，计数器被清 0；否则(递减计数模式)计数器将载入自动重载值。预分频计数器将同时被清除。</p>

## 12.7.7 捕获/比较模式寄存器 (TIM1\_CCMR1)

偏移地址: 0x18

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	OC2CE	OC2M			OC2PE	OC2FE	CC2S	OC1CE	OC1M			OC1PE	OC1FE	CC1S		
	IC2F			IC2PSC				IC1F				IC1PSC				
Type	rw	rw			rw	rw	rw	rw	rw			rw	rw	rw	rw	

输出比较模式:

Bit	Field	Type	Reset	Description
15	OC2CE	rw	0x0	通道 2 输出比较清零使能 (Output compare 2 clear enable)
14:12	OC2M	rw	0x0	通道 2 输出比较模式 (Output compare 2 mode)
11	OC2PE	rw	0x0	通道 2 输出比较预装载使能 (Output compare 2 preload enable)
10	OC2FE	rw	0x0	通道 2 输出比较快速使能 (Output compare 2 fast enable)
9:8	CC2S	rw	0x0	<p>通道 2 捕获/比较选择 (Capture/Compare 2 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <p>00: 通道 2 被配置为输出</p> <p>01: 通道 2 被配置为输入，IC2 映射在 TI2 上</p> <p>10: 通道 2 被配置为输入，IC2 映射在 TI1 上</p> <p>11: 通道 2 被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIM1_SMCR 寄存器的 TS 位选择)</p>

Bit	Field	Type	Reset	Description
7	OC1CE	rw	0x0	<p>通道 1 输出比较清 0 使能 (Output compare 1 clear enable)</p> <p>0: OC1REF 不受 ETRF 输入的影响</p> <p>1: 当检测到 ETRF 输入高电平时, 清除 OC1REF = 0</p>
6:4	OC1M	rw	0x0	<p>通道 1 输出比较模式 (Output compare 1 mode)</p> <p>该位定义了输出参考信号 OC1REF 的动作, 而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效, 而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。</p> <p>000: 冻结。TIM1_CCR1 与 TIM1_CNT 间的比较结果对 OC1REF 不起作用</p> <p>001 : 匹配时设置为高。当 TIM1_CNT 的值与 TIM1_CCR1 的值相同时, 强制 OC1REF 为高电平</p> <p>010 : 匹配时设置为低。当 TIM1_CNT 的值与 TIM1_CCR1 的值相同时, 强制 OC1REF 为低电平</p> <p>011: 匹配时翻转。当 TIM1_CCR1=TIM1_CNT 时, 翻转 OC1REF 的电平</p> <p>100: 强制为低。强制 OC1REF 为低电平</p> <p>101: 强制为高。强制 OC1REF 为高电平</p> <p>110: PWM 模式 1。在递增计数时, 当 TIM1_CNT&lt;TIM1_CCR1 时通道 1 为有效电平, 否则为无效电平; 在递减计数时, 当 TIM1_CNT &gt; TIM1_CCR1 时通道 1 为无效电平, 否则为有效电平。</p> <p>111: PWM 模式 2。在递增计数时, 当 TIM1_CNT&lt;TIM1_CCR1 时通道 1 为无效电平, 否则为有效电平; 在递减计数时, 当 TIM1_CNT &gt; TIM1_CCR1 时通道 1 为有效电平, 否则为无效电平</p> <p>注 1: 当 LOCK 级别设为 3 (TIM1_BDTR 寄存器中的 LOCK 位) 并且 CC1S = 00 (该通道配置成输出) 时, 该位不能被修改。</p> <p>注 2: 在 PWM 模式 1 或 PWM 模式 2 中, 只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时, OC1REF 电平才改变。</p>

Bit	Field	Type	Reset	Description
3	OC1PE	rw	0x0	<p>通道 1 输出比较预装载使能 (Output compare 1 preload enable)</p> <p>0: 禁止 TIM1_CCR1 寄存器的预装载功能, 写入 TIM1_CCR1 寄存器的数值立即生效</p> <p>1: 开启 TIM1_CCR1 寄存器的预装载功能, 读写操作仅对预装载寄存器操作, TIM1_CCR1 的预装载值在更新事件到来时生效</p> <p>注 1: 当 LOCK 级别设为 3 (TIM1_BDTR 寄存器中的 LOCK 位) 并且 CC1S = 00 (该通道配置成输出) 时, 该位不能被修改。</p> <p>注 2: 仅在单脉冲模式下 (TIM1_CR1 寄存器的 OPM= 1), 可以在未确认预装载寄存器情况下使用 PWM 模式, 否则其动作不确定。</p>
2	OC1FE	rw	0x0	<p>通道 1 输出比较快速使能 (Output compare 1 fast enable)</p> <p>该位为 1 时, 若通道配置为 PWM 模式, 会加快捕获/比较输出对触发时间的响应。输出通道将触发输入信号的有效边沿的作用等同于发生了一次比较匹配, 因此 OC 被设置为比较电平而与比较结果无关。</p> <p>0: 禁止通道 1 输出比较快速使能, 当检测到触发输入的一个有效边沿时, 激活 OC1 的最小延时需要 5 个时钟周期</p> <p>1: 开启通道 1 输出比较快速使能。当触发输入由一个有效边沿时, 激活 OC1 的最小延时缩短到 3 个周期</p>
1:0	CC1S	rw	0x0	<p>通道 1 捕获/比较选择 (Capture/Compare 1 selection)</p> <p>该位定义通道的方向和输入信号的选择, 只有在通道关闭时这些位才可写入:</p> <p>00: 通道 1 被配置为输出</p> <p>01: 通道 1 被配置为输入, IC1 映射在 TI1 上</p> <p>10: 通道 1 被配置为输入, IC1 映射在 TI2 上</p> <p>11: 通道 1 被配置为输入, IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIM1_SMCR 寄存器的 TS 位选择)</p>

输入捕获模式:

Bit	Field	Type	Reset	Description
15:12	IC2F	rw	0x0	输入捕获 2 滤波器 (Input capture 2 filter)
11:10	IC2PSC	rw	0x0	输入/捕获 2 预分频器 (Input capture 2 prescaler)

Bit	Field	Type	Reset	Description
9:8	CC2S	rw	0x0	<p>捕获/比较 2 选择 (Capture/Compare 2 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <ul style="list-style-type: none"> <li>00: CC2 通道被配置为输出</li> <li>01: CC2 通道被配置为输入，IC2 映射在 TI2 上</li> <li>10: CC2 通道被配置为输入，IC2 映射在 TI1 上</li> <li>11: CC2 通道被配置为输入，IC2 映射在 TRC 上， 此模式仅工作在内部触发器输入被选中时（由 TIM1_SMCR 寄存器的 TS 位选择）</li> </ul>
7:4	IC1F	rw	0x0	<p>通道 1 输入捕获滤波器 (Input capture 1 filter)</p> <p>数字滤波器由一个事件计数器组成，它记录 N 个输入事件后会产生一个输出的跳变。这些位定义了 IC1 输入信号的采样频率和数字滤波器的长度。</p> <ul style="list-style-type: none"> <li>0000: 无滤波器，以 <math>f_{DTS}</math> 采样</li> <li>1000: 采样频率 <math>f_{sampling} = f_{DTS} / 8, N=6</math></li> <li>0001: 采样频率 <math>f_{sampling} = f_{INT\_CK}, N=2</math></li> <li>1001: 采样频率 <math>f_{sampling} = f_{DTS} / 8, N=8</math></li> <li>0010: 采样频率 <math>f_{sampling} = f_{INT\_CK}, N=4</math></li> <li>1010: 采样频率 <math>f_{sampling} = f_{DTS} / 16, N=5</math></li> <li>0011: 采样频率 <math>f_{sampling} = f_{INT\_CK}, N=8</math></li> <li>1011: 采样频率 <math>f_{sampling} = f_{DTS} / 16, N=6</math></li> <li>0100: 采样频率 <math>f_{sampling} = f_{DTS} / 2, N=6</math></li> <li>1100: 采样频率 <math>f_{sampling} = f_{DTS} / 16, N=8</math></li> <li>0101: 采样频率 <math>f_{sampling} = f_{DTS} / 2, N=8</math></li> <li>1101: 采样频率 <math>f_{sampling} = f_{DTS} / 32, N=5</math></li> <li>0110: 采样频率 <math>f_{sampling} = f_{DTS} / 4, N=6</math></li> <li>1110: 采样频率 <math>f_{sampling} = f_{DTS} / 32, N=6</math></li> <li>0111: 采样频率 <math>f_{sampling} = f_{DTS} / 4, N=8</math></li> <li>1111: 采样频率 <math>f_{sampling} = f_{DTS} / 32, N=8</math></li> </ul>

Bit	Field	Type	Reset	Description
3:2	IC1PSC	rw	0x0	<p>通道 1 输入/捕获预分频器 (Input capture 1 prescaler)</p> <p>这 2 位定义了 IC1 的预分频系数。当 CC1E=0 (TIM1_CCER 寄存器中) 时, 预分频器复位。</p> <p>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</p> <p>01: 每 2 个事件触发一次捕获</p> <p>10: 每 4 个事件触发一次捕获</p> <p>11: 每 8 个事件触发一次捕获</p>
1:0	CC1S	rw	0x0	<p>捕获/比较 1 选择 (Capture/Compare 1 selection)</p> <p>该位定义通道的方向和输入信号的选择, 只有在通道关闭时这些位才可写入:</p> <p>00: CC1 通道被配置为输出</p> <p>01: CC1 通道被配置为输入, IC1 映射在 TI1 上</p> <p>10: CC1 通道被配置为输入, IC1 映射在 TI2 上</p> <p>11: CC1 通道被配置为输入, IC1 映射在 TRC 上, 此模式仅工作在内部触发器输入被选中时 (由 TIM1_SMCR 寄存器的 TS 位选择)</p>

## 12.7.8 捕获/比较模式寄存器 2 (TIM1\_CCMR2)

偏移地址: 0x1C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	OC4CE	OC4M		OC4PE	OC4FE		CC4S	OC3CE	OC3M		OC3PE	OC3FE		CC3S		
	IC4F			IC4PSC		IC3F			IC3PSC							
Type	rw	rw		rw	rw		rw	rw		rw	rw		rw	rw		

输出比较模式:

Bit	Field	Type	Reset	Description
15	OC4CE	rw	0x0	通道 4 输出比较清零使能 (Output compare 4 clear enable)
14:12	OC4M	rw	0x0	通道 4 输出比较模式 (Output compare 4 mode)
11	OC4PE	rw	0x0	通道 4 输出比较预装载使能 (Output compare 4 preload enable)
10	OC4FE	rw	0x0	通道 4 输出比较快速使能 (Output compare 4 fast enable)

Bit	Field	Type	Reset	Description
9:8	CC4S	rw	0x0	<p>通道 4 捕获/比较选择 (Capture/Compare 4 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <ul style="list-style-type: none"> <li>00：通道 4 被配置为输出</li> <li>01：通道 4 被配置为输入，IC4 映射在 TI4 上</li> <li>10：通道 4 被配置为输入，IC4 映射在 TI3 上</li> <li>11：通道 4 被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM1_SMCR 寄存器的 TS 位选择）</li> </ul>
7	OC3CE	rw	0x0	<p>通道 3 输出比较清 0 使能 (Output compare 3 clear enable)</p> <p>0：OC1REF 不受 ETRF 输入的影响</p> <p>1：当检测到 ETRF 输入高电平时，清除 OC1REF = 0</p>
6:4	OC3M	rw	0x0	<p>通道 3 输出比较模式 (Output compare 3 mode)</p> <p>该位定义了输出参考信号 OC3REF 的动作，而 OC3REF 决定了 OC3、OC3N 的值。OC3REF 是高电平有效，而 OC3、OC3N 的有效电平取决于 CC3P、CC3NP 位。</p> <ul style="list-style-type: none"> <li>000：冻结。TIM1_CCR3 与 TIM1_CNT 间的比较结果对 OC3REF 不起作用</li> <li>001：匹配时设置为高。当 TIM1_CNT 的值与 TIM1_CCR3 的值相同时，强制 OC3REF 为高电平</li> <li>010：匹配时设置为低。当 TIM1_CNT 的值与 TIM1_CCR3 的值相同时，强制 OC3REF 为低电平</li> <li>011：匹配时翻转。当 TIM1_CCR3=TIM1_CNT 时，翻转 OC3REF 的电平</li> <li>100：强制为低。强制 OC3REF 为低电平</li> <li>101：强制为高。强制 OC3REF 为高电平</li> <li>110：PWM 模式 1。在递增计数时，当 TIM1_CNT&lt;TIM1_CCR3 时通道 3 为有效电平，否则为无效电平；在递减计数时，当 TIM1_CNT &gt; TIM1_CCR3 时通道 1 为无效电平，否则为有效电平。</li> <li>111：PWM 模式 2。在递增计数时，当 TIM1_CNT&lt;TIM1_CCR3 时通道 3 为无效电平，否则为有效电平；在递减计数时，当 TIM1_CNT&gt;TIM1_CCR3 时通道 3 为有效电平，否则为无效电平</li> </ul> <p>注 1：当 LOCK 级别设为 3 (TIM1_BDTR 寄存器中的 LOCK 位) 并且 CC3S=00 (该通道配置成输出) 时，该位不能被修改。</p> <p>注 2：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC3REF 电平才改变。</p>

Bit	Field	Type	Reset	Description
3	OC3PE	rw	0x0	<p>通道 3 输出比较预装载使能 (Output compare 3 preload enable)</p> <p>0: 禁止 TIM1_CCR3 寄存器的预装载功能, 写入 TIM1_CCR3 寄存器的数据立即生效</p> <p>1: 开启 TIM1_CCR3 寄存器的预装载功能, 读写操作仅对预装载寄存器操作, TIM1_CCR3 的预装载值在更新事件到来时生效</p> <p>注 1: 当 LOCK 级别设为 3 (TIM1_BDTR 寄存器中的 LOCK 位) 并且 CC3S =00 (该通道配置成输出) 时, 该位不能被修改。</p> <p>注 2: 仅在单脉冲模式下 (TIM1_CR1 寄存器的 OPM=1), 可以在未确认预装载寄存器情况下使用 PWM 模式, 否则其动作不确定。</p>
2	OC3FE	rw	0x0	<p>通道 3 输出比较快速使能 (Output compare 3 fast enable)</p> <p>该位为 1 时, 若通道配置为 PWM 模式, 会加快捕获/比较输出对触发时间的响应。输出通道将触发输入信号的有效边沿的作用等同于发生了一次比较匹配, 因此 OC 被设置为比较电平而与比较结果无关。</p> <p>0: 禁止通道 3 输出比较快速使能, 当检测到触发输入的一个有效边沿时, 激活 OC1 的最小延时需要 5 个时钟周期</p> <p>1: 开启通道 3 输出比较快速使能。当触发输入由一个有效边沿时, 激活 OC1 的最小延时缩短到 3 个周期</p>
1:0	CC3S	rw	0x0	<p>通道 3 捕获/比较选择 (Capture/Compare 3 selection)</p> <p>该位定义通道的方向和输入信号的选择, 只有在通道关闭时这些位才可写入:</p> <p>00: 通道 3 被配置为输出</p> <p>01: 通道 3 被配置为输入, IC3 映射在 TI3 上</p> <p>10: 通道 3 被配置为输入, IC3 映射在 TI4 上</p> <p>11: 通道 3 被配置为输入, IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIM1_SMCR 寄存器的 TS 位选择)</p>

输入捕获模式:

Bit	Field	Type	Reset	Description
15:12	IC4F	rw	0x0	输入捕获 4 滤波器 (Input capture 4 filter)
11:10	IC4PSC	rw	0x0	输入/捕获 4 预分频器 (Input capture 4 prescaler)

Bit	Field	Type	Reset	Description
9:8	CC4S	rw	0x0	<p>捕获/比较 4 选择 (Capture/Compare 4 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <ul style="list-style-type: none"> <li>00: CC4 通道被配置为输出</li> <li>01: CC4 通道被配置为输入，IC4 映射在 TI4 上</li> <li>10: CC4 通道被配置为输入，IC4 映射在 TI3 上</li> <li>11: CC4 通道被配置为输入，IC4 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时（由 TIM1_SMCR 寄存器的 TS 位选择）</li> </ul>
7:4	IC3F	rw	0x0	<p>通道 3 输入捕获滤波器 (Input capture 3 filter)</p> <p>数字滤波器由一个事件计数器组成，它记录 N 个输入事件后会产生一个输出的跳变。这些位定义了 IC1 输入信号的采样频率和数字滤波器的长度。</p> <ul style="list-style-type: none"> <li>0000: 无滤波器，以 <math>f_{DTS}</math> 采样</li> <li>1000: 采样频率 <math>f_{sampling} = f_{DTS}/8, N=6</math></li> <li>0001: 采样频率 <math>f_{sampling} = f_{INT\_CK}, N=2</math></li> <li>1001: 采样频率 <math>f_{sampling} = f_{DTS}/8, N=8</math></li> <li>0010: 采样频率 <math>f_{sampling} = f_{INT\_CK}, N=4</math></li> <li>1010: 采样频率 <math>f_{sampling} = f_{DTS}/16, N=5</math></li> <li>0011: 采样频率 <math>f_{sampling} = f_{INT\_CK}, N=8</math></li> <li>1011: 采样频率 <math>f_{sampling} = f_{DTS}/16, N=6</math></li> <li>0100: 采样频率 <math>f_{sampling} = f_{DTS}/2, N=6</math></li> <li>1100: 采样频率 <math>f_{sampling} = f_{DTS}/16, N=8</math></li> <li>0101: 采样频率 <math>f_{sampling} = f_{DTS}/2, N=8</math></li> <li>1101: 采样频率 <math>f_{sampling} = f_{DTS}/32, N=5</math></li> <li>0110: 采样频率 <math>f_{sampling} = f_{DTS}/4, N=6</math></li> <li>1110: 采样频率 <math>f_{sampling} = f_{DTS}/32, N=6</math></li> <li>0111: 采样频率 <math>f_{sampling} = f_{DTS}/4, N=8</math></li> <li>1111: 采样频率 <math>f_{sampling} = f_{DTS}/32, N=8</math></li> </ul>

Bit	Field	Type	Reset	Description
3:2	IC3PSC	rw	0x0	<p>通道 3 输入/捕获预分频器 (Input capture 3 prescaler)</p> <p>这 2 位定义了 IC3 的预分频系数。当 CC3E=0 (TIM1_CCER 寄存器中) 时, 预分频器复位。</p> <p>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</p> <p>01: 每 2 个事件触发一次捕获</p> <p>10: 每 4 个事件触发一次捕获</p> <p>11: 每 8 个事件触发一次捕获</p>
1:0	CC3S	rw	0x0	<p>捕获/比较 3 选择 (Capture/Compare 3 selection)</p> <p>该位定义通道的方向和输入信号的选择, 只有在通道关闭时这些位才可写入:</p> <p>00: CC3 通道被配置为输出</p> <p>01: CC3 通道被配置为输入, IC3 映射在 TI3 上</p> <p>10: CC3 通道被配置为输入, IC3 映射在 TI4 上</p> <p>11: CC3 通道被配置为输入, IC3 映射在 TRC 上, 此模式仅工作在内部触发器输入被选中时 (由 TIM1_SMCR 寄存器的 TS 位选择)</p>

### 12.7.9 捕获/比较使能寄存器 (TIM1\_CCER)

偏移地址: 0x20

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CC4	Res	CC													
Type	rw		rw													

Bit	Field	Type	Reset	Description
15	CC4NP	rw	0x0	<p>通道 4 输入 / 捕获互补输出极性 ( Capture/Compare 4 complementary output polarity)</p> <p>参考 CC1NP 的描述。</p>
14	Reserved			保留, 始终读为 0。

Bit	Field	Type	Reset	Description
13	CC4P	rw	0x0	通道 4 输入/捕获输出极性 (Capture/Compare 4 output polarity) 参考 CC1P 的描述。
12	CC4E	rw	0x0	通道 4 输入/捕获输出使能 (Capture/Compare 4 output enable) 参考 CC1E 的描述。
11	CC3NP	rw	0x0	通道 3 输入/捕获互补输出极性 (Capture/Compare 3 complementary output polarity) 参考 CC1NP 的描述。
10	CC3NE	rw	0x0	通道 3 输入 / 捕获互补输出使能 ( Capture/Compare 3 complementary output enable) 参考 CC1NE 的描述。
9	CC3P	rw	0x0	通道 3 输入捕获输出极性 (Capture/Compare 3 output polarity) 参考 CC1P 的描述。
8	CC3E	rw	0x0	通道 3 输入/捕获输出使能 (Capture/Compare 3 output enable) 参考 CC1E 的描述。
7	CC2NP	rw	0x0	通道 2 输入/捕获互补输出极性 (Capture/Compare 2 complementary output polarity) 参考 CC1NP 的描述。
6	CC2NE	rw	0x0	通道 2 输入 / 捕获互补输出使能 ( Capture/Compare 2 complementary output enable) 参考 CC1NE 的描述。
5	CC2P	rw	0x0	通道 2 输入捕获输出极性 (Capture/Compare 2 output polarity) 参考 CC1P 的描述。
4	CC2E	rw	0x0	通道 2 输入/捕获输出使能 (Capture/Compare 2 output enable) 参考 CC1E 的描述。
3	CC1NP	rw	0x0	通道 1 输入 / 捕获互补输出极性 ( Capture/Compare 1 complementary output polarity) 0: OC1N 高电平有效 1: OC1N 低电平有效 注: 当 LOCK 级别 (TIM1_BDTR 寄存器中的 LCCK 位) 设为 3 或 2 且 CC1S = 00 (通道配置为输出) 时, 该位不能被修改。
2	CC1NE	rw	0x0	通道 1 输入 / 捕获互补输出使能 ( Capture/Compare 1 complementary output enable) 0: 关闭通道 1 互补输出。OC1N 禁止输出。 1: 开启通道 1 互补输出。 OC1N 信号输出到对应的输出引脚, 其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。

Bit	Field	Type	Reset	Description
1	CC1P	rw	0x0	<p>通道 1 输入/捕获输出极性 (Capture/Compare 1 output polarity)</p> <p>通道 1 配置为输出时, 此位定义了输出信号极性:</p> <p>0: OC1 高电平有效</p> <p>1: OC1 低电平有效</p> <p>通道 1 配置为输入时, 此位定义了输入信号极性:</p> <p>0: 不反相: 捕获发生在 IC1 的上升沿; 当用作外部触发器时, IC1 不反相。</p> <p>1: 反相: 捕获发生在 IC1 的下降沿; 当用作外部触发器时, IC1 反相。</p> <p>注: 当 LOCK 级别 (TIM1_BDTR 寄存器中的 LCCK 位) 设为 3 或 2 时, 该位不能被修改。</p>
0	CC1E	rw	0x0	<p>通道 1 输入/捕获输出使能 (Capture/Compare 1 output enable)</p> <p>通道 1 配置为输出时,: :</p> <p>0: 关闭。OC1 禁止输出</p> <p>1: 开启。OC1 信号输出到对应的输出引脚.</p> <p>其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1NE 位的值。</p> <p>CC1 通道配置为输入:</p> <p>该位决定了输入捕获功能是否启用。</p> <p>0: 捕获禁止</p> <p>1: 捕获使能</p>

表 12-10 输入模式下, ICx 的极性选择如下表:

CCxP	CCxNP	ICx 极性
0	0	上升沿
1	0	下降沿
1	1	上升沿或下降沿
0	1	保留

## 12.7.10 计数器 (TIM1\_CNT)

偏移地址: 0x24

复位值: 0x0000

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

Field	CNT
Type	rw

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x0000	计数器的值 (Count value)

### 12.7.11 预分频器 (TIM1\_PSC)

偏移地址: 0x28

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	PSC															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	PSC	rw	0x0000	预分频器的值 (Prescaler value) 计数器的时钟频率 ( $ck_{cnt}$ ) = $fCK_{PSC} / (PSC+1)$ 。 当发生更新事件时, PSC 的值装入当前预分频寄存器。

### 12.7.12 自动预装载寄存器 (TIM1\_ARR)

偏移地址: 0x2C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ARR															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	ARR	rw	0x0000	自动预装载的值 (Auto-reload value) 这些位定义了计数器的自动重载值。当自动重装载的值为 0 时, 计数器不工作。

### 12.7.13 重复计数寄存器 (TIM1\_RCR)

偏移地址: 0x30

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	REP_CNT												REP			
Type	rw												rw			

Bit	Field	Type	Reset	Description
15:8	REP_CNT	rw	0x00	<p>重复计数器实时写入的值 (Repetition counter value of real-time writing)</p> <p>用于修改重复计数模式下，实时的将更新中断标志位 (UIF) 的检测点移位，即可将 UIF 左移 (REP_CNT) 个相位 (相减值为负值时右移)。在更新事件后写入这些位 (注意在更新事件前写入 REP_CNT 将会使移位无效)。</p>
7:0	REP	rw	0x00	<p>重复计数器的值 (Repetition counter value)</p> <p>重复计数器的值定义了更新事件的产生速率。重复计数器计数值递减为 0 时产生更新事件。如果允许产生更新中断，则会同时影响产生更新中断的速率。</p> <p>对 REP 值的写入在下次更新事件发生时生效，所以在 PWM 模式中，(REP+1) 对应着：</p> <p>在边沿对齐模式下，PWM 周期的数目</p> <p>在中心对称模式下，PWM 半周期的数目</p>

## 12.7.14 捕获/比较寄存器 1 (TIM1\_CCR1)

偏移地址：0x34

复位值：0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CCR1															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CCR1	rw	0x0000	<p>通道 1 捕获/比较的值 (Capture/Compare 1 value)</p> <p>当通道 1 配置为输入时，CCR1 的值决定了上次捕获事件的计数器值 (此时寄存器为只读)。</p> <p>当通道 1 配置为输出时，CCR1 的值为即将和计数器值比较的值。如果开启了预装载功能，写入的值在更新事件发生时传输到当前捕获/比较寄存器；否则写入的值立即载入捕获/比较寄存器。</p>

### 12.7.15 捕获/比较寄存器 2 (TIM1\_CCR2)

偏移地址: 0x3C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CCR2															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CCR2	rw	0x0000	通道 2 捕获/比较的值 (Capture/Compare 2 value) 参考 CCR1 的描述。

### 12.7.16 捕获/比较寄存器 3 (TIM1\_CCR3)

偏移地址: 0x40

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CCR3															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CCR3	rw	0x0000	通道 3 捕获/比较的值 (Capture/Compare 3 value) 参考 CCR1 的描述。

### 12.7.17 捕获比较寄存器 4 (TIM1\_CCR4)

偏移地址: 0x40

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CCR4															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CCR4	rw	0x0000	通道 4 捕获/比较的值 (Capture/Compare 4 value) 参考 CCR1 的描述。

## 12.7.18 刹车和死区寄存器 (TIM1\_BDTR)

偏移地址: 0x44

复位值: 0x00000000

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

注: 根据锁定设置, AOE、BKP、BKE、OSSI、OSSR 和 DTG 位均可被写保护, 有必要在第一次写入 TIM1\_BDTR 寄存器时对它们进行配置。

Bit	Field	Type	Reset	Description
31:17	Reserved			保留, 始终读为 0。
16	DOE	rw	0x0	<p>直接输出 (Direct output enable)  当刹车有效、MOE 置零后, 有效。  1: 立即输出空闲状态  0: 刹车输入后, 等待一个死区时间后输出空闲状态。</p>
15	MOE	rw	0x0	<p>主输出使能 (Main output enable)  当通道 x 配置为输出时, 根据 AOE 位的设置值, 该位可以由软件清 0 或被自动置 1。当刹车输入有效时, 该位被硬件异步清 0。  0: 禁止 OCx 和 OCxN 输出或强制为空闲状态  1: 如果设置了相应的使能位 (TIM1_CCER 寄存器的 CCxE、CCxNE 位), 则开启 OCx 和 OCxN 输出</p>
14	AOE	rw	0x0	<p>自动输出使能 (AutoMatic output enable)  0: MOE 不能被硬件置 1  1: MOE 能被软件置 1 或刹车无效时在下一个更新事件被硬件自动置 1  注: 当 LOCK 级别 (TIM1_BDTR 寄存器中的 LOCK 位) 设为 1 时, 该位不能被修改。</p>

Bit	Field	Type	Reset	Description
13	BKP	rw	0x0	<p>刹车输入极性 (Break Polarity)</p> <p>0: 刹车输入低电平有效 1: 刹车输入高电平有效</p> <p>注: 当 LOCK 级别 (TIM1_BDTR 寄存器中的 LOCK 位) 设为 1 时, 该位不能被修改。</p>
12	BKE	rw	0x0	<p>刹车功能使能 (Break enable)</p> <p>0: 禁止刹车输入 1: 开启刹车输入</p> <p>注 1: 当 LOCK 级别 (TIM1_BDTR 寄存器中的 LOCK 位) 设为 1 时, 该位不能被修改。</p> <p>注 2: 刹车输入包括 BRK 和 CSS 时钟失效事件。</p>
11	OSSR	rw	0x0	<p>运行模式下“关闭状态”选择 (Off-state selection for Run mode)</p> <p>该位仅适用于当 MOE = 1 且通道为互补输出。</p> <p>0: 当定时器不工作时, 禁止 OC/OCN 输出 1: 当定时器不工作时, 如果 CCxE = 1 或 CCxNE = 1, 首先开启 OC/OCN 并输出无效电平, 然后置 OC/OCN 使能输出信号=1。</p> <p>注: 当 LOCK 级别 (TIM1_BDTR 寄存器中的 LOCK 位) 设为 2 时, 该位不能被修改。</p>
10	OSSI	rw	0x0	<p>空闲模式下“关闭状态”选择 (Off-state selection for Idle mode)</p> <p>该位仅适用于当 MOE = 0 且通道设为输出时。</p> <p>0: 当定时器不工作时, 禁止 OC/OCN 输出 1: 当定时器不工作时, 如果 CCxE = 1 或 CCxNE = 1, 首先 OC/OCN 输出其空闲电平, 然后 OC/OCN 使能输出信号=1。</p> <p>注: 当 LOCK 级别 (TIM1_BDTR 寄存器中的 LOCK 位) 设为 2 时, 该位不能被修改。</p>

Bit	Field	Type	Reset	Description
9:8	LOCK	rw	0x0	<p>锁定设置 (Lock configuration)</p> <p>该位定义了寄存器的写保护功能。</p> <p>00: 写保护功能关闭, 寄存器无写保护</p> <p>01: 锁定级别 1, 不能写入 TIM1_BDTR 寄存器的 OTG、BKE、BKP、AOE 位和 TIM1_CR2 寄存器的 OISx/OISxN 位</p> <p>10: 锁定级别 2, 不能写入锁定级别 1 中的各位, 也不能写入 CC 极性位 (当相关通道通过 CCxS 位设为输出时, CC 极性位是 TIM1_CCER 寄存器的 CCxP/CCNxP 位) 以及 OSSR/OSSI 位</p> <p>11: 锁定级别 3, 不能写入锁定级别 2 中的各位, 也不能写入 CC 控制位 (当相关通道通过 CCxS 位设为输出时, CC 控制位是 TIM1_CCMRx 寄存器的 OCxM/OCxPE 位)</p> <p>注: 在系统复位后, LOCK 位只能写一次, 当写入 TIM1_BDTR 寄存器时, lock 被写保护</p>
7:0	DTG	rw	0x00	<p>死区发生器调整 (Dead-time generator setup)</p> <p>这些位定义了插入互补输出之间的死区持续时间。</p> <p>注: 当 LOCK 级别 (TIM1_BDTR 寄存器中的 LOCK 位) 设为 1、2 或 3 时, 不能修改这些位。</p>

### 12.7.19 DMA 控制寄存器 (TIM1\_DCR)

偏移地址: 0x48

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.			DBL						Res.			DBA			
Type				rw									rw			

Bit	Field	Type	Reset	Description
15:13	Reserved			保留, 始终读为 0。

Bit	Field	Type	Reset	Description
12:8	DBL	w	0x00	<p>DMA 连续传送长度 (DMA burst length)</p> <p>这些位定义了 DMA 在连续模式下的访问寄存器的数量</p> <p>00000: 1 次传输 00001: 2 次传输 00010: 3 次传输 ..... ..... 10001: 18 次传输</p>
7:5	Reserved			保留, 始终读为 0。
4:0	DBA	w	0x00	<p>DMA 基地址 (DMA base address)</p> <p>这些位定义了 DMA 在连续模式下访问 TIM1_DMAR 寄存器的第一个地址。DBA 定义为从 TIM1_CR1 寄存器所在地址开始的偏移值:</p> <p>00000: TIM1_CR1 00001: TIM1_CR2 00010: TIM1_SMCR .....</p>

### 12.7.20 连续模式的 DMA 地址 (TIM1\_DMAR)

偏移地址: 0x4C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	DMAB															
Type	w															

Bit	Field	Type	Reset	Description
-----	-------	------	-------	-------------

15:0	DMAB	w	0x0000	DMA 连续传送寄存器 (DMA register for burst accesses) 对 TIM1_DMAR 寄存器的写操作会导致对以下地址所在寄存器的存取操作: TIM1_CR1 地址 + DBA + DMA 索引, 其中: TIM1_CR1 地址是 TIM1_CR1 寄存器所在的地址;DBA 是 TIM1_DCR 寄存器中定义的基地址; DMA 索引是 DMA 自动控制的偏移量, 它取决于 TIM1_DCR 寄存器中定义的 DBL。
------	------	---	--------	---

### 12.7.21 捕获/比较模式寄存器 3 (TIM1\_CCMR3)

偏移地址: 0x54

复位值: 0x0000

通道仅适用于输出比较模式

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.										OC5M	OC5PE	OC5FE	Res.		
Type											rw	rw	rw			

Bit	Field	Type	Reset	Description
15:7	Reserved			保留, 始终读为 0。
6:4	OC5M	rw	0x0	输出比较 5 模式 (Output compare 5 mode)
3	OC5PE	rw	0x0	输出比较 5 预装载使能 (Output compare 5 preload enable)
2	OC5FE	rw	0x0	输出比较 5 快速使能 (Output compare 5 fast enable)
1:0	Reserved			保留, 始终读为 0。

### 12.7.22 捕获/比较寄存器 5 (TIM1\_CCR5)

偏移地址: 0x58

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CCR5															
Type	rw															

Bit	Field	Type	Reset	Description
-----	-------	------	-------	-------------

15:0	CCR5	rw	0x0000	比较 5 的值 (Compare 5 value) CC5 通道只能配置为输出： 当通道 5 配置为输出时，CCR5 的值为即将和计数器值比较的值。如果开启了预装载功能，写入的值在更新事件发生时传输到当前比较寄存器；否则写入的值立即载入比较寄存器。
------	------	----	--------	--

### 12.7.23 PWM 移相/DMA repeat 更新请求使能寄存器 (TIM1\_PDER)

偏移地址: 0x5C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Res.											CCR5_SHIFT_EN	CCR4_SHIFT_EN	CCR3_SHIFT_EN	CCR2_SHIFT_EN	CCR1_SHIFT_EN	CCDR_EPE
Type												rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
15:6				保留，始终读为 0.
5	CCR5_SHIFT_EN	rw	0x0	允许通道 5 输出 PWM 移相使能位 0: 禁止通道 5 输出 PWM 移相 1: 允许通道 5 输出 PWM 移相 具体见 CCRxFALL 寄存器描述移相操作
4	CCR4_SHIFT_EN	rw	0x0	允许通道 4 输出 PWM 移相使能位 0: 禁止通道 4 输出 PWM 移相 1: 允许通道 4 输出 PWM 移相 具体见 CCRxFALL 寄存器描述移相操作
3	CCR3_SHIFT_EN	rw	0x0	允许通道 3 输出 PWM 移相使能位 0: 禁止通道 3 输出 PWM 移相 1: 允许通道 3 输出 PWM 移相 具体见 CCRxFALL 寄存器描述移相操作

Bit	Field	Type	Reset	Description
2	CCR2_SHIFT_EN	rw	0x0	<p>允许通道 2 输出 PWM 移相使能位 0: 禁止通道 2 输出 PWM 移相 1: 允许通道 2 输出 PWM 移相 具体见 CCRxFALL 寄存器描述移相操作</p>
1	CCR1_SHIFT_EN	rw	0x0	<p>允许通道 1 输出 PWM 移相使能位 0: 禁止通道 1 输出 PWM 移相 1: 允许通道 1 输出 PWM 移相 具体见 CCRxFALL 寄存器描述移相操作</p>
0	CCDREPE	rw	0x0	<p>使能 DMA 再每次 underflow 或是 overflow 时都发出更新请求 0: DAM 发生更新请求需要根据重复计数寄存器的值来产生。 1: 使能 DMA 再每次 underflow 或 overflow 都发出更新请求。</p>

### 12.7.24 PWM 移相向下计数捕获/比较寄存器 (TIM1\_CCRxFALL)

偏移地址: 0x60 ~ 0x70

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CCRx FALL															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CCRx FALL	rw	0x0000	<p>通道 x 在 PWM 中央对齐模式向下计数时的捕获/比较值 PWM 移相功能: 开启 PDER 寄存器的 PWM 移相使能, 根据需要移动相位, 配置 CCRxFALL 以及 CCRx, 即可实现 PWM 输出可编程的移相波形, 可左移或是右移。</p>

# 13 16 位通用定时器 (TIM3/4)

## 13.1 TIM3 简介

TIM3 由一个 16 位可实时编程预分频器和一个 16 位计数方向可调的自动重装载计数器组成，可以为用户提供便捷的计数定时功能，计数器时钟由预分频器分频得到。通用定时器具有多种用途，如输入功能（测量输入信号的脉冲宽度、频率，PWM 输入等），输出功能（PWM 输出、单脉冲模式输出等）。

## 13.2 功能框图

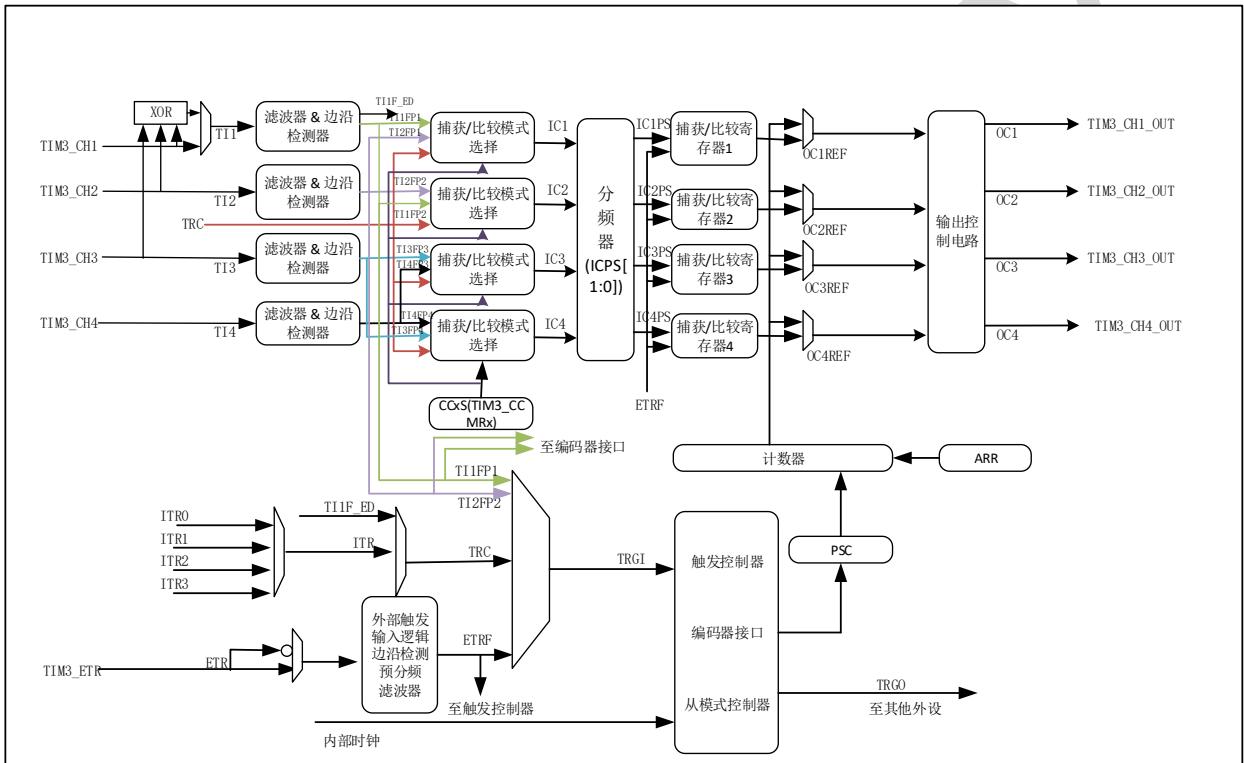


图 13-1 TIM3 结构图

上图为 TIM3 的结构框图，主要由输入单元、输出单元、时基单元、捕获/比较模块等结构组成。

## 13.3 主要特征

- 1) 16 位可实时编程预分频器，分频系数：1–65536 可调。
- 2) 时钟源可选：内部时钟源，外部时钟模式（外部时钟引脚/外部触发输入 ETR），内部触发输入（编码器模式）。
- 3) 16 位自动重装载计数器（计数方向：递增、递减、递增/递减）。
- 4) 输入捕获：输入信号的脉冲宽度、周期的测量。
- 5) 触发输入可以作为外部时钟或者逐周期管理。
- 6) 支持编码器、霍尔传感器等接口。
- 7) 4 个输出通道。
- 8) 输出比较（控制输出波形或指示定时器计时结束）。

- 9) PWM 输出（死区时间可调）（边沿对齐或中央对齐模式）。
- 10) 单脉冲输出。
- 11) 产生中断/DMA 请求的事件：更新事件、触发事件、输入捕获、输出比较或者刹车输入时。

注：

- 更新事件：计数器上溢/下溢，计数器初始化。
- 触发事件：计数器初始化，启动或停止计数。

## 13.4 中断

TIM3 的中断包括：捕获/比较 1 中断、捕获/比较 2 中断、捕获/比较 3 中断、捕获/比较 4 中断、更新中断和触发中断，当相应的中断使能位打开，发生相应的事件时，产生相应的中断。

## 13.5 DMA

TIM3 能够在发生单个事件时生成多个 DMA 请求。主要目的是在没有软件开销的情况下，多次重新编程 TIM3 的一部分寄存器也可以用于按周期读取数个寄存器。

## 13.6 功能描述

### 13.6.1 时钟

#### 13.6.1.1 时钟选择

计数器的时钟源有以下几种：

- 1) 内部时钟 (INT\_CK)。
- 2) 外部时钟模式：外部输入引脚 (TIx) 或者外部触发输入 (ETR)。
- 3) 内部触发输入 (编码器模式)。

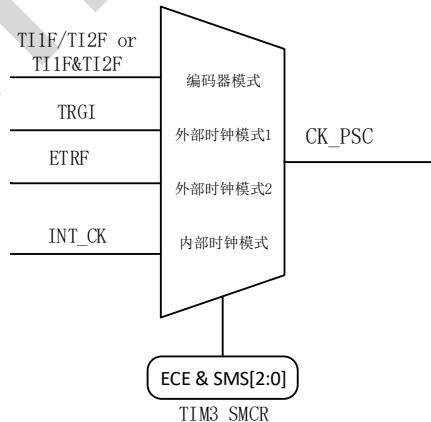


图 13-2 时钟选择

##### 13.6.1.1.1 内部时钟源 (INT\_CK)

当 SMS=000，关闭从模式时，计数器使能打开，预分频器的时钟直接由内部时钟驱动。此时计数器时钟为内部时钟分频后的时钟。

### 13.6.1.1.2 外部时钟模式 1 (外部输入引脚 TIx)

当 SMS = 111 时，选择外部时钟模式 1 (外部输入引脚 TIx)。计数器由选定的输入引脚的每个上升沿和下降沿驱动。

例：计数器在 T11 输入端的上升沿递增计数，具体配置如下：

- 1) 配置 TIM3\_CCMR1 寄存器 CC1S=01, IC1F[3:0]=xxxx, TIM3\_CCER 寄存器 CC1P=0, 选择通道 1 检测 TI1 输入的上升沿为有效沿，定义输入滤波器带宽。
- 2) 配置 TIM3\_SMCR 寄存器中的 TS=100, SMS=111, 选定 TI1 的边沿作为触发输入源，时钟选择外部时钟模式 1。
- 3) 配置 TIM3\_CR1 寄存器的 DIR=0, CEN=1, 选择递增计数模式，启动计数器。

当 TI1 出现有效边沿时，计数器递增计数一次且 TIF 标志位由硬件置 1。TI1 的有效边沿和计数器的实际时钟之间的延时取决于预 TI1 输入引脚同步电路设计。

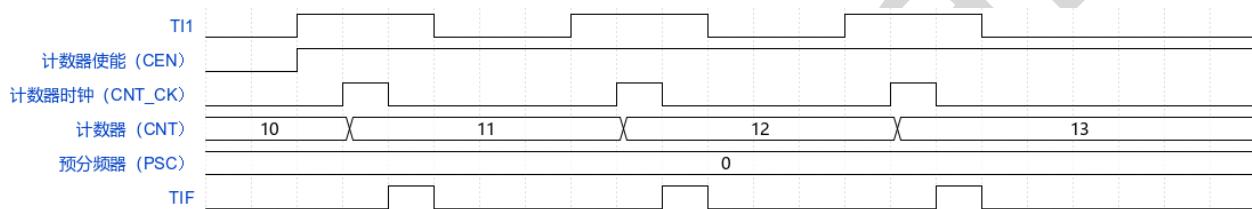


图 13-3 外部时钟模式 1 下的控制电路

### 13.6.1.1.3 外部时钟模式 2 (外部触发输入 ETR)

当 TIM3\_SMCR 寄存器的 ECE=1 时，使能外部时钟模式 2，计数器由 ETRF 信号上的有效边沿驱动。

例：在 ETR 下每 4 个下降沿计数一次的递增计数器，具体配置如下：

- 1) 配置 TIM3\_SMCR 寄存器中的 ETF[3: 0] = 0010，每 4 个 ETR 信号的有效边沿驱动计数器计数一次；ETP=1，ETR 被反相，选择下降沿有效；ECE=1，选择外部时钟模式 2
  - 2) 配置 TIM3\_CR1 寄存器 DIR=0, CEN=1 选择计数方向为递增计数，启动计数器
- 在 ETR 的下降沿和计数器实际时钟之间的延时取决于在 ETRP 信号端的重新同步电路。

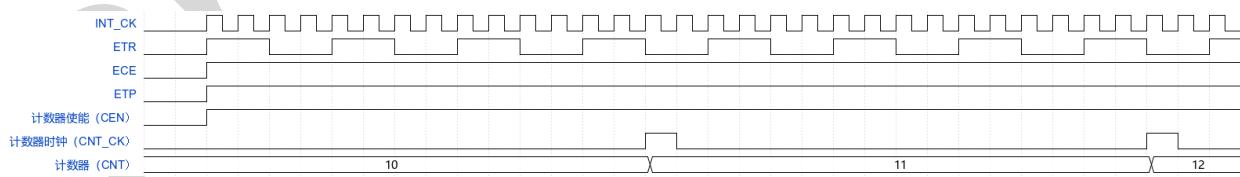


图 13-4 外部时钟模式 2 下的控制电路

## 13.6.1.2 时基单元

TIM3 的时基单元主要包括：计数器寄存器 (TIM3\_CNT)、预分频器寄存器 (TIM3\_PSC) 和自动装载寄存器 (TIM3\_ARR)。

计数器由一个 16 位的计数器和对应的自动装载寄存器组成，可以实现递增计数，递减计数，递增和递减计数的功能。计数器的时钟由预分频器提供，预分频器由预分频计数器和对应的寄存器组成，分频系数为 1-65536，可以随时写入，在下一次更新事件时生效。

自动预装载寄存器有预装载功能的 16 位影子寄存器，通过设置 TIM3\_CR1 寄存器的 APRE 位选择写入 ARR 寄存器的值立即生效或发生更新事件时载入影子寄存器。

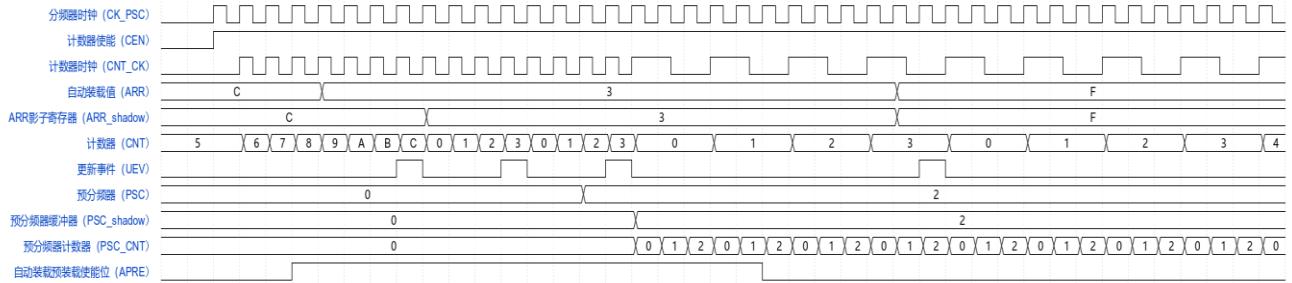


图 13-5 自动预装载

### 13.6.1.3 计数模式

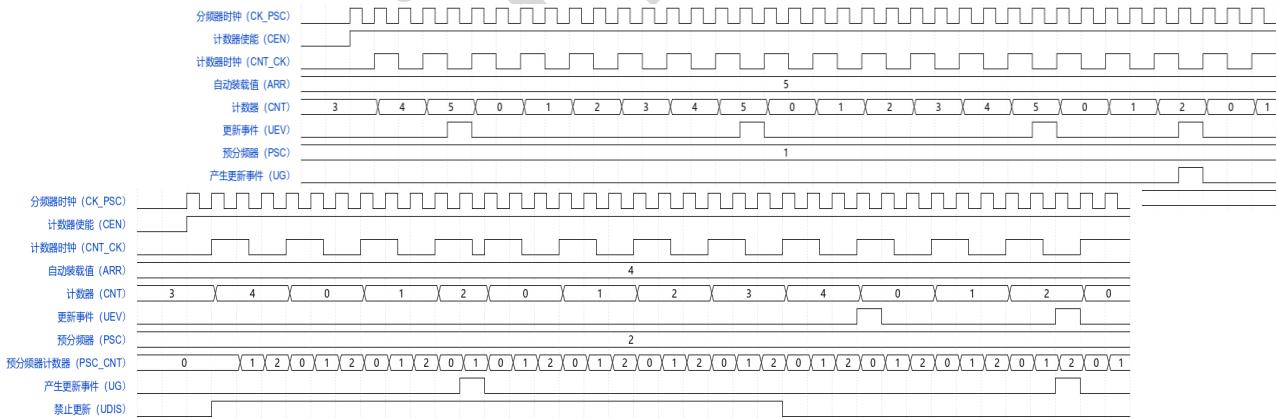
通过配置 TIM3\_CR1 寄存器的 DIR 位和 CMS 位可以选择计数器的计数模式，可以分为三种计数模式，递增计数模式、递减计数模式和中央对齐计数模式（递增/递减计数模式），下面对每个计数模式做详细介绍。

#### 13.6.1.3.1 递增计数模式

配置 TIM3\_CR1 寄存器 CMS=0, DIR=0，选择递增计数模式。

递增计数模式下，在使能 TIM3\_CR1 的 CEN 后计数器由 0 开始递增计数，直至 TIM3\_ARR 的值，并产生一个计数器上溢事件（更新事件），更新事件后计数器重新从 0 开始重新递增计数。设置 TIM3\_EGR 寄存器的 UG=1，同样可以产生一个更新事件。

图 13-6 递增计数模式 (UDIS=0)



当配置 TIM3\_CR1 寄存器的 UDIS=1，禁止产生更新事件，当计数器发生上溢事件时，不产生更新事件；配置 UG=1，不产生更新事件，计数器和预分频器计数器会被初始化，从零开始递增计数。

图 13-7 递增计数模式 (UDIS=0 禁止产生更新事件)

注：发生更新事件时：

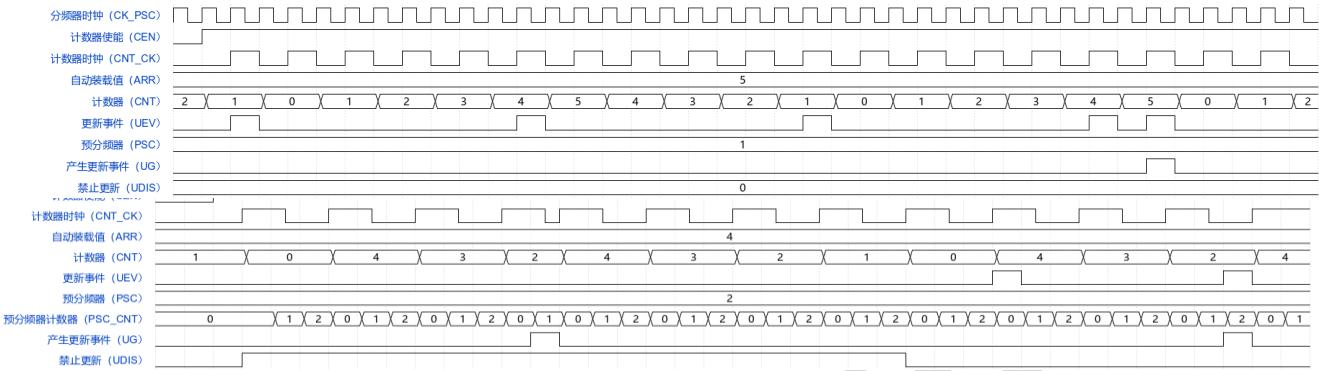
- ARR 寄存器中的值被载入 ARR\_shadow 寄存器中。
- 预分频器的预装载值生效。

### 13.6.1.3.2 递减计数模式

配置 TIM3\_CR1 寄存器的 CMS=0, DIR=1, 选择递减计数模式。

递减计数模式下, 此时计数器从自动装载值 TIM3\_ARR 开始递减计数, 计数到 0 时, 产生一个下溢事件 (更新事件)。设置 TIM3\_EGR 寄存器的 UG=1, 同样可以产生一个更新事件, 更新事件后计数器重新从自动重装载值 TIM3\_ARR 开始重新递减计数 (TIM3\_CR1 寄存器 UDIS=0)。

图 13-8 递减计数模式 (UDIS=0)



当配置 TIM3\_CR1 寄存器的 UDIS=1, 禁止产生更新事件, 当计数器发生下溢事件时, 不产生更新事件; 配置 UG=1, 同样不产生更新事件, 但是计数器和预分频器计数器会被初始化, 从 TIM3\_ARR 开始计数。

图 13-9 递减计数模式 (UDIS=1 禁止产生更新事件)

### 13.6.1.3.3 中央计数模式 (递增/递减计数模式)

配置 TIM3\_CR1 寄存器的 CMS ≠ 0 (此时写入 DIR 无效), 选择中央对齐计数模式。

中央对齐计数模式, 递增计数和递减计数交替进行。递增计数到 ARR-1 时, 产生一个上溢事件, 然后从 ARR 先开始递减计数到 1, 产生一个下溢事件, 再从 0 开始递增计数。

设置 TIM3\_EGR 寄存器的 UG=1, 同样可以产生一个更新事件, 更新事件后计数器从 0 开始重新递增计数 (TIM3\_CR1 寄存器 UDIS=0)。

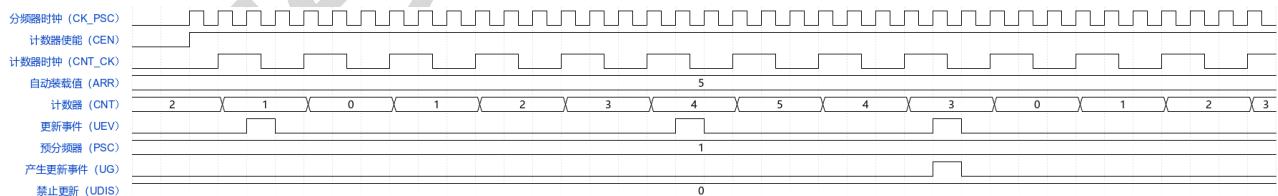


图 13-10 中央计数模式 (UDIS=0)

当配置 TIM3\_CR1 寄存器的 UDIS=1, 禁止产生更新事件, 当计数器发生上溢或下溢事件时, 不产生更新事件; 配置 UG=1, 同样不产生更新事件, 但是计数器和预分频器计数器会被初始化, 从零开始重新计数。

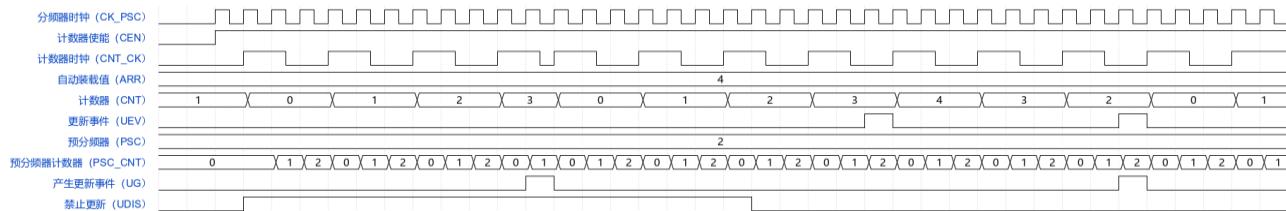


图 13-11 中央计数模式 (UDIS=1 禁止产生更新事件)

### 13.6.1.4 定时器同步

不同的 TIMx 定时器在内部连接，通过配置一个定时器工作在主模式，一个定时器工作在从模式（配置 TIMx\_SMCR 寄存器 SMS=100/101/110，从模式选择复位模式、门控模式或触发模式），可以实现定时器之间的级联或同步，实现对处于从模式定时器的计数器进行复位、启动、通知或提供时钟等操作。

#### 13.6.1.4.1 从模式

##### 复位模式

配置 TIM3\_SMCR 寄存器 SMS=100，从模式选择复位模式。此模式下，发生触发输入事件会使计数器清零重启。

例如，TI2 输入端的下降沿触发计数器重启：

- 1) 配置 CC2S=01，CC2 通道被配置为输入模式，IC2 映射在 TI2 上，配置 TIM3\_CCER 寄存器 CC2P=1，检测下降沿。
- 2) 配置 TIM3\_SMCR 寄存器 SMS = 100，从模式选择为复位模式；置 TIM3\_SMCR 寄存器中 TS = 110，ETF=0000，选择滤波后的定时器输入 2 (TI2FP2) 作为同步计数器的触发输入。
- 3) 配置 TIM3\_CR1 寄存器 ARR, PSC=0, DIR=0，配置预分频系数并选择计数方向，配置 CEN=1，使能计数器。

计数器的时钟源由内部时钟提供，当检测到 TI2 的下降沿，计数器被清零重启。此时触发器中断标记被硬件置 1。

下图为复位模式下 TIM3\_ARR = 0x13 时的时序图：

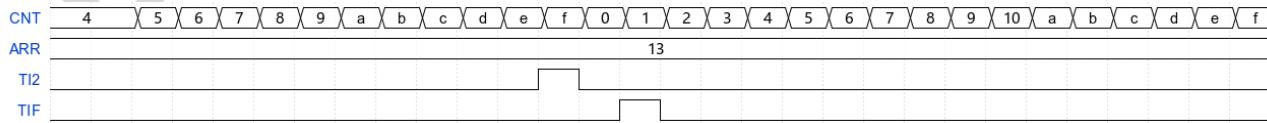


图 13-12 复位模式的控制电路图

##### 门控模式

配置 TIM3\_SMCR 寄存器 SMS=101，从模式选择门控模式。此模式下，触发输入为高时，计数器始终开启，否则计数器停止（但不发生复位操作），计数器的开启和停止可控。

例如，计数器只在 TI1 为高时计数：

- 1) 配置 CC1S=01，CC1 通道被配置为输入模式，IC1 映射在 TI1 上，配置 TIM3\_CCER 寄

存器  $CC1P=1$ , 检测下降沿。

- 2) 配置  $TIM3\_SMCR$  寄存器  $SMS=101$ , 从模式选择为门控模式, 配置  $TS=101$ ,  $ETF=0000$ , 选择滤波后的定时器输入 1 ( $TI1FP1$ ) 作为同步计数器的触发输入。
- 3) 配置  $TIM3\_CR1$  寄存器  $ARR$ ,  $PSC=0$ ,  $DIR=0$ , 配置预分频系数并选择计数方向, 配置  $CEN=1$ , 使能计数器。

计数器的时钟源由内部时钟提供, 当检测到  $TI1$  的低电平, 计数器开始计数, 当  $TI1$  为高电平时, 计数器停止。计数器开启或停止都会将  $TIF$  置 1。

下图为门控模式下  $TIM3\_ARR=0xf$  的时序图。

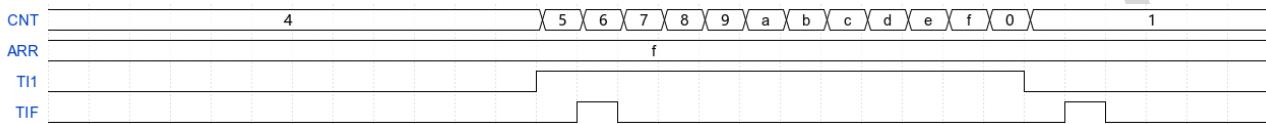


图 13-13 门控模式下的控制电路

#### 触发模式

配置  $TIM3\_SMCR$  寄存器  $SMS=111$ , 从模式选择触发模式。计数器在触发输入的上升沿启动, 计数器的启动可控, 停止不可控。

例如, 计数器在  $TI1$  输入的上升沿开始计数:

- 1) 配置  $CC1S=01$ ,  $CC1$  通道被配置为输入模式,  $IC1$  映射在  $TI1$  上, 配置  $TIM3\_CCER$  寄存器  $CC1P=0$ , 检测上升沿。
- 2) 配置  $TIM3\_SMCR$  寄存器  $SMS=110$ , 从模式选择为触发模式; 配置  $TS=101$ ,  $ETF=0000$ , 选择滤波后的定时器输入 1 ( $TI1FP1$ ) 作为同步计数器的触发输入。
- 3) 配置  $TIM3\_CR1$  寄存器  $ARR$ ,  $PSC=0$ ,  $DIR=0$ , 配置预分频系数并选择计数方向, 配置  $CEN=1$ , 使能计数器。

计数器的时钟源由内部时钟提供, 当检测到  $TI1$  的上升沿, 计数器开始计数。

下图为触发模式下  $TIM3\_ARR=0xf$  的时序图。

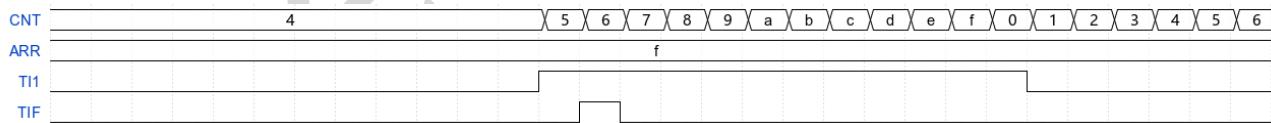


图 13-14 触发器模式下的控制电路

#### 外部时钟模式 2+触发模式

当时钟源选择外部时钟模式 2,  $ETR$  信号被用作外部时钟的输入时, 可以与另一种从模式 (外部时钟模式 1 和编码器模式除外) 一起使用。在复位模式、门控模式或触发模式可以选择另一个输入作为触发输入 ( $TRGI$ )。

例如, 计数器在  $ETR$  的每一个上升沿计数一次:

- 1) 配置  $TIM3\_SMCR$  寄存器  $ETF = 0000$ ,  $ETPS = 00$ ,  $ETP = 0$  检测  $ETR$  的上升沿, 配置  $ECE = 1$  选择外部时钟模式 2。
- 2) 配置  $TIM3\_CCMR1$  寄存器中  $CC1S=01$ , 选择输入捕获源,  $TIM3\_CCER$  寄存器中  $CC1P=0$ , 选择上升沿有效。

- 3) 配置 TIM3\_SMCR 寄存器中 SMS = 110，从模式选择为触发模式。配置 TIM3\_SMCR 寄存器中 TS = 101，选择 TI1 作为输入源。
- 4) 配置 TIM3\_CR1 寄存器 ARR, PSC=0, DIR=0, 配置预分频系数并选择计数方向，配置 CEN=1，使能计数器。

计数器在 ETR 的上升沿开始计数，当检测到 TI1 上的上升沿时，TIF 被硬件置 1。ETR 信号的上升沿和计数器实际复位间的延时取决于 ETRP 输入端的重同步电路。

下图为外部时钟模式 2+触发模式下 TIM3\_ARR=13 时的时序图。

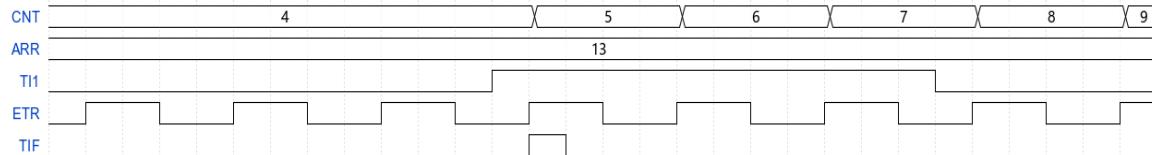


图 13-15 外部时钟模式 2+触发模式下的控制电路

#### 13.6.1.4.2 定时器间的同步

详见 TIM1 相关描述。

### 13.6.2 输入捕获

#### 13.6.2.1 输入捕获

输入捕获部分包括数字滤波器、多路复用、预分频器等，其结构如下图所示：

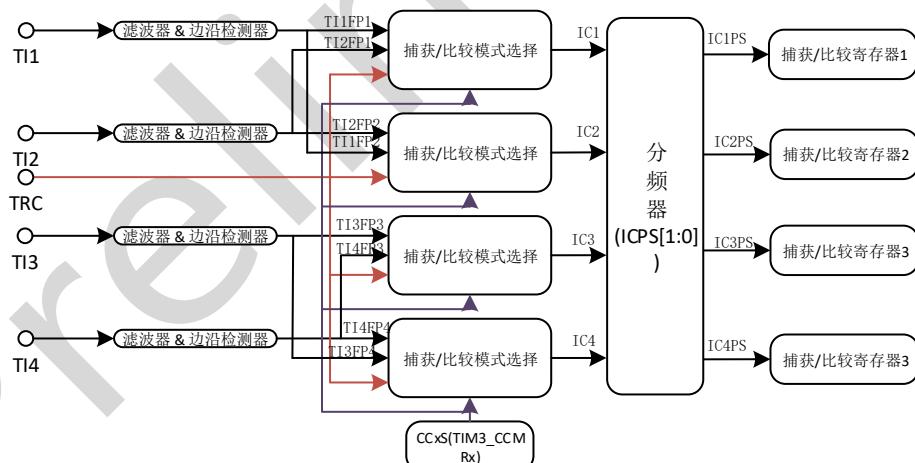


图 13-16 TIM3 输入捕获结构图

通过配置 TIM3\_CCMRx 寄存器 ICxF，可以配置数字滤波器的滤波宽度（滤波器的采样频率及数字滤波器长度如下表所示），当数字滤波器的输入信号带宽大于滤波宽度时，输入信号有效；数字滤波器对输入引脚 TIx 输入信号的采样后，产生一个滤波后的信号 TIxF，然后通过边沿检测器和软件配置选择 TIxF 信号的有效边沿，产生一个有效信号 TIxFPx，这个信号可以用作捕获控制信号或作为从模式控制器的触发输入信号，然后经过预分频产生一个信号 ICxPS，进入捕获/比较寄存器。

表 13-1 数字滤波器长度与 ICxF 的对应关系表

IC1F[3:0]	采样频率和滤波长度	IC1F[3:0]	采样频率和滤波长度
0000	无滤波器, 以 $f_{DTS}$ 采样	1000	采样频率 $f_{sampling} = f_{DTS} / 8, N=6$
0001	采样频率 $f_{sampling} = f_{INT\_CK}, N=2$	1001	采样频率 $f_{sampling} = f_{DTS} / 8, N=8$
0010	采样频率 $f_{sampling} = f_{INT\_CK}, N=4$	1010	采样频率 $f_{sampling} = f_{DTS} / 16, N=5$
0011	采样频率 $f_{sampling} = f_{INT\_CK}, N=8$	1011	采样频率 $f_{sampling} = f_{DTS} / 16, N=6$
0100	采样频率 $f_{sampling} = f_{DTS}/2, N=6$	1100	采样频率 $f_{sampling} = f_{DTS} / 16, N=8$
0101	采样频率 $f_{sampling} = f_{DTS} / 2, N=8$	1101	采样频率 $f_{sampling} = f_{DTS} / 32, N=5$
0110	采样频率 $f_{sampling} = f_{DTS} / 4, N=6$	1110	采样频率 $f_{sampling} = f_{DTS} / 32, N=6$
0111	采样频率 $f_{sampling} = f_{DTS}/4, N=8$	1111	采样频率 $f_{sampling} = f_{DTS} / 32, N=8$

在输入捕获模式下, 输入捕获发生在影子寄存器上, 然后影子寄存器的内容载入到捕获/比较寄存器中。

输入捕获模式下, 当检测到信号 ICx 上的有效边沿后, 计数器的当前值被锁存到对应的影子寄存器(TIM3\_CCRx\_shadow)上, 再复制到对应的捕获比较寄存器(TIM3\_CCRx)中。当开启了中断或 DMA 使能, 发生捕获事件时, 将产生相应的中断或 DMA 请求。发生捕获事件时, 会将状态寄存器 (TIM3\_SR) 中的捕获标志位 CCxIF 置 1, 通过配置 CCxIF=0 或读取 TIM3\_CCRx 中的数据, 清除 CCxIF 标志位。当 CCxIF 未被清零时, 发生输入捕获事件, 重复捕获标志位将会被置 1, 通过配置 CCxOF=0, 可以清除 CCxOF 标志位。

例如, 通过采样 TI1 输入信号的有效沿, 在 TI1 的上升沿来到时捕获当前计数器的值, 锁存到 TIM3\_CCR1 寄存器中, 步骤如下:

- 1) 配置 TIM3\_CCMR1 寄存器 CC1S=01,CC1 通道被配置为输入, IC1 映射在 TI1 上。
- 2) 配置 TIM3\_CCMRx 寄存器 IC1F[3:0], 配置数字滤波器的滤波宽度 (按需配置)。
- 3) 配置 TIM3\_CCER 寄存器 CC1P=0, 选择捕获发生在 TI1 信号的上升沿。
- 4) 配置 TIM3\_CCMR1 寄存器的 IC1PS , 选择预分频系数。
- 5) 配置 TIM3\_CCER 寄存器的 CC1E = 1, 开启输入/捕获通道 1 的捕获使能。
- 6) 配置 TIM3\_DIER 寄存器 CC1IE=1, CC1DE=1, 开始通道 1 的捕获/比较通道 1 中断使能和允许捕获/比较通道 1 的 DMA 请求。

注:

- 当通道配置为输入模式时, TIM3\_CCRx 寄存器属性变为只读。
- 为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息, 建议在读出捕获溢出标志之前读取数据。
- 设置 TIM3\_EGR 寄存器中相应的 CCxG 位, 可以通过软件产生输入捕获中断或 DMA 请求。

### 13.6.2.2 PWM 捕获

PWM 输入模式的操作配置与一般输入捕获有以下不同点：

- 1) 两个边沿有效且极性相反的 IC<sub>x</sub> 信号被映射至同一个 TI<sub>x</sub> 输入。
- 2) 配置从模式控制器为复位模式，将其中一路 TI<sub>x</sub>FP 作为触发输入信号。

例：测量 TI1 的 PWM 信号的长度（TIM3\_CCR1 寄存器）和占空比（TIM3\_CCR2 寄存器），具体步骤如下（取决于 INT\_CK 的频率和预分频器的值）。

- 1) 配置 TIM3\_CR1 寄存器 DIR=0，选择计数器计数模式为递增计数模式。
- 2) 配置 TIM3\_CCMR1 寄存器的 CC1S = 01，将 IC1 映射在 TI1 上，选择 TIM3\_CCR1 的有效输入。
- 3) 配置 CC1P =0，选择 TI1FP1 的有效极性(上升沿有效)(将计数器的值捕获到 TIM3\_CCR1 中并清除计数器)。
- 4) 配置 TIM3\_CCMR1 寄存器的 CC2S =10，将 IC2 映射在 TI1 上，选择 TIM3\_CCR2 的有效输入。
- 5) 配置 CC2P =0，选择 TI2FP2 的有效极性(下降沿有效)(将计数器的值捕获到 TIM3\_CCR2 中)。
- 6) 配置 TIM3\_SMCR 寄存器中的 TS = 101，选择 TI1FP1 为有效的触发输入信号。
- 7) 配置 TIM3\_SMCR 中的 SMS = 100，从模式控制器设置为复位模式。
- 8) 配置 TIM3\_CCER 寄存器中 CC1E=1 且 CC2E = 1。开启 CC1 通道和 CC2 通道的捕获使能。



图 13-17 PWM 输入模式时序

注：由于从模式控制器只连接了 TI1FP1 和 TI2FP2，所以 PWM 输入模式只适用于 TIM3\_CH1/TIM3\_CH2 信号。

### 13.6.2.3 编码器接口

编码器接口模式就是计数器在 TI1 和 TI2 正交信号相互作用下计数，在输入源改变期间，计数方向被硬件自动修改。通过配置 TIM3\_SMCR 寄存器 SMS 位可以选择输入源，根据输入源的不同，可以将编码器接口模式分为 3 种模式，SMS = 001，编码器接口模式 1；SMS = 010，编码器接口模式 2；SMS=011，编码器接口模式 3；三种模式具体计数操作如下表所示。两个输入 TI1 和 TI2 被用来作为正交编码器的接口。

编码器模式下，计数器开启之前必须先配置好 ARR 寄存器，因为使用编码器接口模式相当于使用了一个带有方向选择的外部时钟。计数器在 0 到 TIM3\_ARR 寄存器的自动装载值之间连续计数（递增计数和递减计数由外部时钟控制）。

注：编码器模式不支持外部时钟模式 2。

编码器接口模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合，假设 TI1 和 TI2 不同时变换。

表 13-2 计数方向与编码器信号的关系

计数模式	相对电平 (TI1FP1 相对于 TI2, TI2FP2 相对于 TI1)	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
编码器接口模式 1 (只在 TI1 计数)	高电平	递减计数	递增计数	-	-
	低电平	递增计数	递减计数	-	-
编码器接口模式 2 (只在 TI2 计数)	高电平	-	-	递增计数	递减计数
	低电平	-	-	递减计数	递增计数
编码器接口模式 3 (在 TI1 和 TI2 计数)	高电平	递减计数	递增计数	递增计数	递减计数
	低电平	递增计数	递减计数	递减计数	递增计数

下例是计数器在编码器接口模式下的配置和时序图，从图中可以看出计数信号的产生和方向控制。具体配置如下：

- 1) 配置 TIM3\_CCMR 寄存器 CC1S=01，将 IC1FP1 映射到 TI1 上。
- 2) 配置 TIM3\_CCMR 寄存器 CC2S =01，将 IC2FP2 映射到 TI2 上。
- 3) 配置 TIM3\_CCER 寄存器 CC1P =0，IC1 不反相，此时 IC1=TI1。
- 4) 配置 TIM3\_CCER 寄存器 CC2P =0，IC2 不反相，此时 IC1=TI2。
- 5) 配置 TIM3\_SMCR 寄存器 SMS =011，选择编码器模式 3，根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿计数。
- 6) 配置 TIM3\_CR1 寄存器 CEN =1，开启计数器。

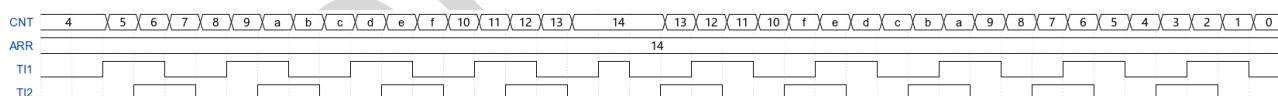


图 13-18 编码器模式下的计数器时序图

下图为当 IC1FP1 极性反相时计数器的时序图（CC1P = 1，其他配置不变）。

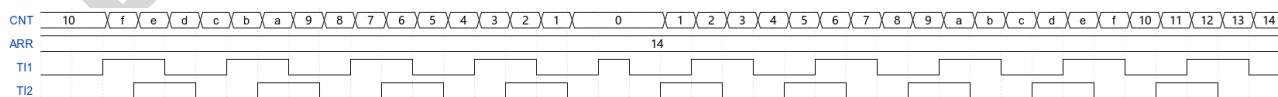


图 13-19 IC1FP1 反相的编码器接口模式时序图

编码器接口模式下，计数器可以提供传感器当前位置的信息。通过使用另一个配置在捕获模式的定时器测量两个编码器事件的间隔周期来获得动态的信息（速度，加速度，减速度）。根据两个编码器事件的间隔周期，可以定期读取计数器。可以通过把计数器的值锁存到第三个输入捕获寄存器（捕获信号必须是周期性的并且可以由另一个定时器产生）来实现。还可以通过 DMA 请求来读取它的值。

### 13.6.2.4 定时器异或

配置 TIM3\_CR2 寄存器的 TI1S =1，将 TIM3\_CH1、TIM3\_CH2 和 TIM3\_CH3 引脚经异或后连接到 TI1 的输入端，用于定时器的所有输入模式。

例：TIM3\_CH1、TIM3\_CH2 和 TIM3\_CH3 引脚经异或后连接到 TI1 的输入端，采样 TI1 输入信号的有效沿，在 TI1 的上升沿来到时捕获当前计数器的值，锁存到 TIM3\_CCR1 寄存器中，具体配置如下：

- 1) 配置 TIM3\_CR2 寄存器 TI1S=1，配置定时器的三个输入经异或后连接到 TI1 输入通道。
- 2) 配置 TIM3\_CCMR1 寄存器 CC1S=01,CC1 通道被配置为输入，IC1 映射在 TI1 上。
- 3) 配置 TIM3\_CCMRx 寄存器 IC1F[3:0],配置数字滤波器的滤波宽度（按需配置）。
- 4) 配置 TIM3\_CCER 寄存器 CC1P=0，选择捕获发生在 TI1 信号的上升沿。
- 5) 配置 TIM3\_CCMR1 寄存器的 IC1PS，选择预分频系数。
- 6) 配置 TIM3\_CCER 寄存器的 CC1E = 1，开启输入/捕获通道 1 的捕获使能。
- 7) 配置 TIM3\_CR1 寄存器 CEN=1，启动计数器。

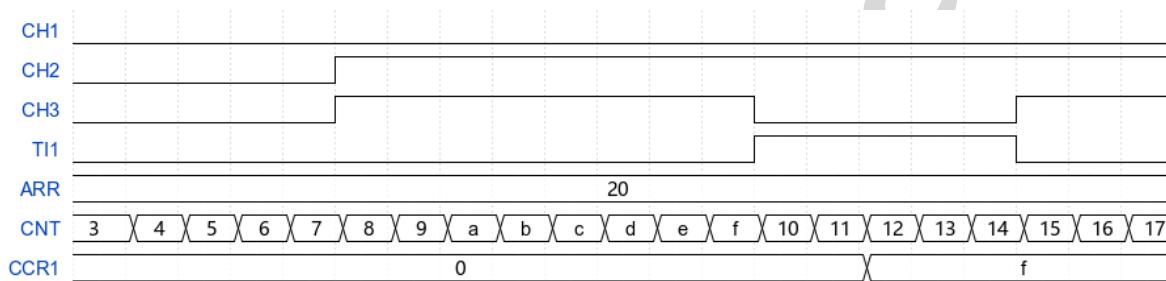


图 13-20 (TI1 异或输入) 输入捕获波形图

#### 13.6.2.4.1 霍尔接口电路

霍尔传感器接口模式是异或功能的一个应用实例，可以用来驱动电机，在使用 TIM1 产生 PWM 信号驱动电机时，可以将另一个计数器 TIM2/TIM3 作为“接口定时器”来连接霍尔传感器，配置 TIM1\_CR2 寄存器 TI1S=1，将定时器的 3 个输入脚（CH1、CH2、CH3）经异或后连接到 TI1 输入通道，“接口定时器”接收这个信号。三个霍尔传感器与“接口定时器”的三路输入捕获引脚对应连接，每个传感器输入一路波形到输入引脚。分析输入捕获信号可以计算电机速度的信息。

“接口定时器”在输出模式下可以产生一个用来控制 TIM1 PWM 输出的脉冲，用来驱动电机。所以“接口定时器”在输出比较或 PWM 模式延时一段时间后产生一个正脉冲，然后通过 TRGO 输出被传送到 TIM1。

例：霍尔输入连接到 TIM3 定时器，要求每次任一霍尔输入上发生变化之后的一个指定的时刻，改变高级控制定时器 TIM1 的 PWM 配置。

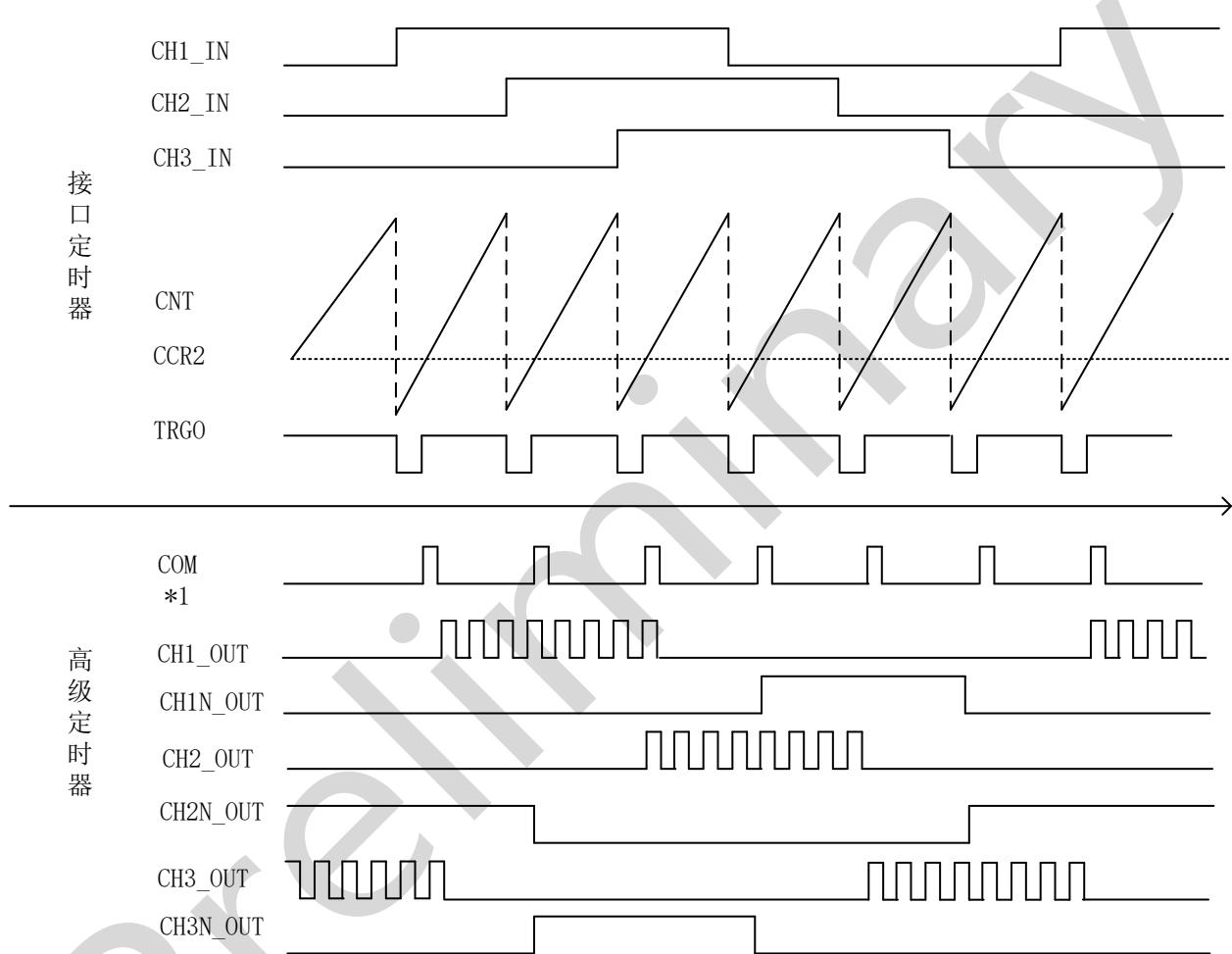
- 1) 配置 TIM3\_CR2 寄存器 TI1S=1，配置三个定时器输入经异或后连接到 TI1 输入通道。
- 2) 配置 TIM3\_ARR 为其最大值（计数器必须通过 TI1 的变化清零）。配置 PSC，设置计数周期大于传感器上的两次变化的时间。
- 3) 配置 TIM3\_CCMR1 寄存器设 CC1S=01，配置通道 1 为捕获模式（选中 TRC）。
- 4) 配置 TIM3\_CCMR1 寄存器 CC2S=00, OC2M=111，配置通道 2 为 PWM2 模式，并具

有要求的延时。

5) 配置 TIM3\_CR2 寄存器 MMS=101, 选择 OC2REF 作为 TRGO 上的触发输出。

TIM3 中, 正确的 ITR 输入必须是触发器输入, 定时器被编程为产生 PWM 信号, 捕获/比较控制信号为预装载的 (TIM1\_CR2 寄存器中 CCPC = 1), 同时触发输入控制 COM 事件 (TIM1\_CR2 寄存器中 CCUS = 1)。在一次 COM 事件后, 写入下一步的 PWM 控制位 (CCxE、OCxM), 这可以在处理 OC2REF 上升沿的中断子程序里实现。

下图显示了这个实例:



\*1:com事件发生, 通过中断子程序设定新的CCxE、CCxNE和OCxM来配置下一步动作。

图 13-21 霍尔传感器接口的实例

### 13.6.3 比较/输出

捕获比较通道的比较输出部分由比较器、输出控制电路和捕获 / 比较寄存器组成, 其结构图如下图所示:

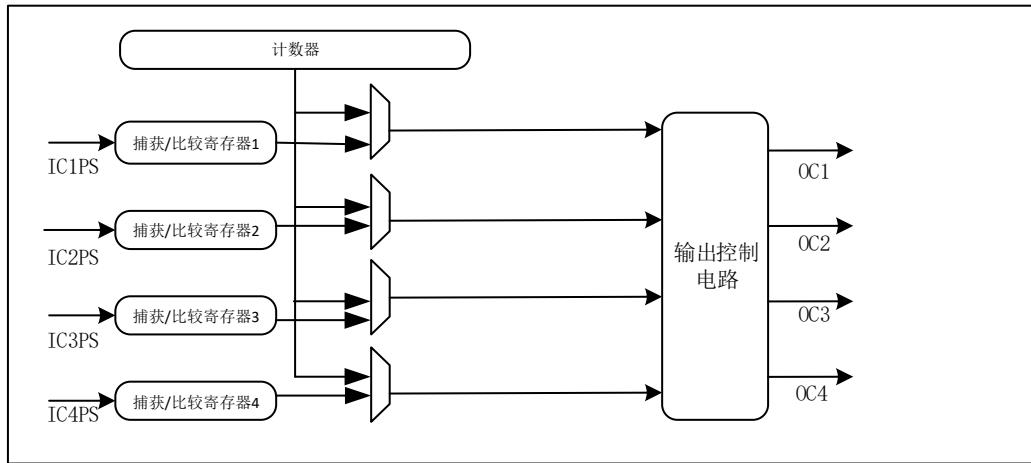


图 13-22 捕获 / 比较的输出部分

在输出比较模式下，捕获比较寄存器的内容被载入到影子寄存器中，然后影子寄存器的内容和计数器当前值进行比较。捕获/比较模块包括一个捕获/比较寄存器（预装载寄存器）和一个影子寄存器，读写过程仅操作捕获/比较寄存器。

### 13.6.3.1 强制输出

配置 TIM3\_CCMRx 寄存器中  $CCxS = 00$ ，将通道  $CCx$  设置为输出模式，通过配置 TIM3\_CCMRx 寄存器  $OCxM$  位，可以直接将输出比较信号直接强制为有效或无效状态，不依赖于输出比较结果。配置 TIM3\_CCMRx 寄存器  $OCxM = 100$ ，强置输出比较信号为无效状态。此时  $OCxREF$  被强置为低电平。配置 TIM3\_CCMRx 寄存器  $OCxM = 101$ ，强置输出比较信号为有效状态。此时  $OCxREF$  被强置为高电平（ $OCxREF$  始终为高电平有效）。

注：强制输出模式下，在 TIM3\_CCRx 影子寄存器和计数器之间的输出比较仍在进行，比较结果的相应标志位也会被修改，如果开启了对应的中断和 DMA 请求，仍会产生对应的中断和 DMA 请求。

### 13.6.3.2 输出比较

输出比较模式下，当计数器与捕获比较寄存器值相同时，可以表示计时结束，也可以根据 TIM3\_CCMRx 寄存器的  $OCxM$  位的配置用来输出不同的波形。

例如，当计数器与捕获/比较寄存器的内容匹配时，输出比较模式下的操作如下：

- 1) 在比较匹配时， $OCxM$  的值不同，输出通道  $x$  信号  $OCx$  的操作不同：
  - a)  $OCxM = 000$ :  $OCx$  信号的保持它的电平。
  - b)  $OCxM = 001$ :  $OCx$  信号被设置成有效电平。
  - c)  $OCxM = 010$ :  $OCx$  信号被设置成无效电平。
  - d)  $OCxM = 011$ :  $OCx$  信号进行翻转。
- 2) 匹配时设置中断状态寄存器中的标志位（TIM3\_SR 寄存器中的  $CCxIF$  位）。
- 3) 当配置了 TIM3\_DIER 寄存器中的  $CCxE = 1$ ，匹配时则产生一个中断。
- 4) 当配置了 TIM3\_DIER 寄存器中的  $CCxDE = 1$ ，匹配时则产生一个 DMA 请求。

输出比较模式也可以用来输出一个单脉冲（单脉冲输出模式）。

通道 1 的输出比较模式的配置步骤如下：

- 1) 配置计数器的时钟（选择时钟源，配置预分频系数）。
- 2) 配置 TIM3\_ARR 和 TIM3\_CCR1 寄存器。
- 3) 配置 TIM3\_DIER 寄存器的 CC1IE =1，使能捕获/比较 1 中断。
- 4) 配置输出模式：
  - a) 配置 OC1M = 011，OC1 比较匹配时翻转。
  - b) 配置 OC1PE = 0，禁止 TIM3\_CCRx 寄存器的预装载功能。
  - c) 配置 CC1P = 1，OC1 低电平有效。
  - d) 配置 CC1E = 1，开启输出/比较 1 输出使能，OC1 信号输出到对应的输出引脚。

配置 TIM3\_CR1 寄存器的 CEN =1，启动计数器。

当配置 TIM3\_CCMRx 寄存器中 OCxPE=0，禁止 TIM3\_CCRx 寄存器的预装载功能时，可以随时写入 TIM3\_CCRx 寄存器，并且写入的值立即生效。当配置 TIM3\_CCMRx 寄存器中 OCxPE=1，启用 TIM3\_CCRx 寄存器的预装载功能时，读写仅对预装载寄存器进行操作，TIM3\_CCR1 预装载寄存器的值在下次更新事件到来时生效。下图给出了一个例子。

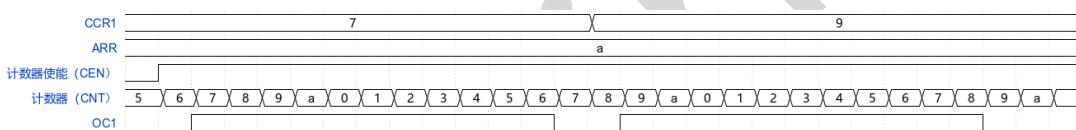


图 13-23 输出比较模式，翻转 OC1

注：

- 输出比较模式下，更新事件不会对输出结果产生影响。
- 强制输出模式下，在 TIM3\_CCRx 影子寄存器和计数器之间的输出比较仍在进行，比较结果的相应标志位也会被修改，如果开启了对应的中断和 DMA 请求，仍会产生对应的中断和 DMA 请求。

### 13.6.3.3 PWM 输出

在 PWM 模式下，根据 TIM3\_ARR 寄存器和 TIM3\_CCRx 寄存器的值，产生一个频率、占空比可控的 PWM 波形。

配置与通道 x 对应的 TIM3\_CCMRx 寄存器的 OCxM=110 或 OCxM=111，选择通道 x 进入 PWM 模式 1 或 PWM 模式 2。PWM 模式下，计数器和 CCR 会一直进行比较，根据配置和比较结果，通道 x 输出不同的信号，因此 TIM3 可以产生 4 个同频率下独立占空比的 PWM 输出信号。PWM 模式下必须开启 TIM3\_CCRx 的预装载功能和 TIM3\_ARR 寄存器的预装载功能。写入 TIM3\_CCR1 预装载寄存器和 TIM3\_ARR 预装载寄存器的值在发生下个更新事件时，才会生效，载入相应的影子寄存器。所以 PWM 模式下，使能计数器前要先初始化所有的寄存器，也可以设置 UG=1，产生更新事件。

配置 TIM3\_CCER 寄存器的 CCxP 选择 OCx 的有效极性。配置 TIM3\_CCER 寄存器的 CCxE、CCxNE 位和 TIM3\_BDTR 寄存器的 MOE、OSSI、OSSR 位控制 OCx 的输出使能。配置 TIM3\_CR1 寄存器的 CMS 位，可以选择产生边沿对齐或中央对齐的 PWM 信号。

- 1) CMS=00, 边沿对齐模式, 再进一步配置 DIR, 选择计数模式(递增或递减计数模式)。
- 2) CMS=01, 中央对齐模式1。
- 3) CMS=10, 中央对其模式2。
- 4) CMS=11, 中央对齐模式3。

### 13.6.3.3.1 PWM 边沿对齐模式——递增计数模式

在递增计数模式配置的基础上, 配置 TIM3\_CCMRx 寄存器 CCxS=00, 选择输出模式, OCxM=110, 选择 PWM 模式1, 当 TIM3\_CNT < TIM3\_CCRx 时通道 x (OCxREF) 为有效电平, 否则为无效电平。如果 TIM3\_CCRx 中的比较值大于自动重装载值(TIM3\_ARR), 则 OCxREF 保持为有效电平。如果比较值为 0, 则 OCxREF 保持为无效电平。图 16-24 为 CCR1=1, CCR2=2, CCR3=3, CCR4=b, ARR=a 时边沿对齐递增计数时 PWM 模式1 的波形实例。

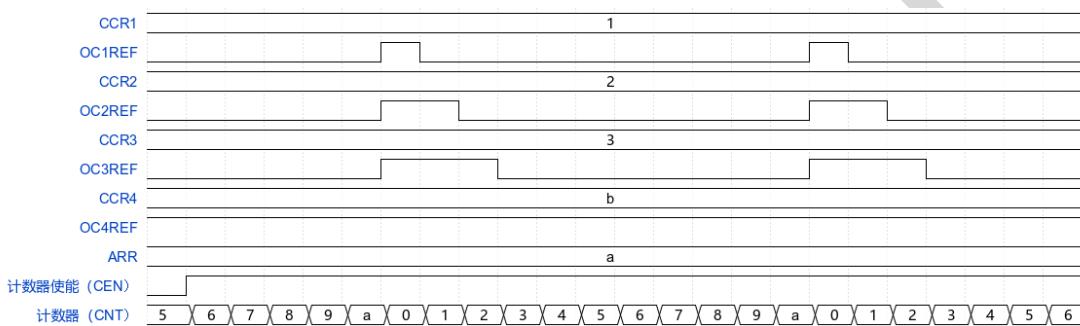


图 13-24 边沿对齐递增计数时 PWM 模式 1 的波形

### 13.6.3.3.2 PWM 边沿对齐模式——递减计数模式

在递减计数模式配置的基础上, 配置 TIM3\_CCMRx 寄存器 CCxS=00, 选择输出模式, OCxM=110, 选择 PWM 模式1, 当 TIM3\_CNT > TIM3\_CCRx 时通道 x (OCxREF) 为无效电平, 否则有效电平。图 16-25 为 CCR1=6, CCR2=4, CCR3=9, CCR4=b, ARR=a 时边沿对齐递减计数时 PWM 模式1 的波形实例。

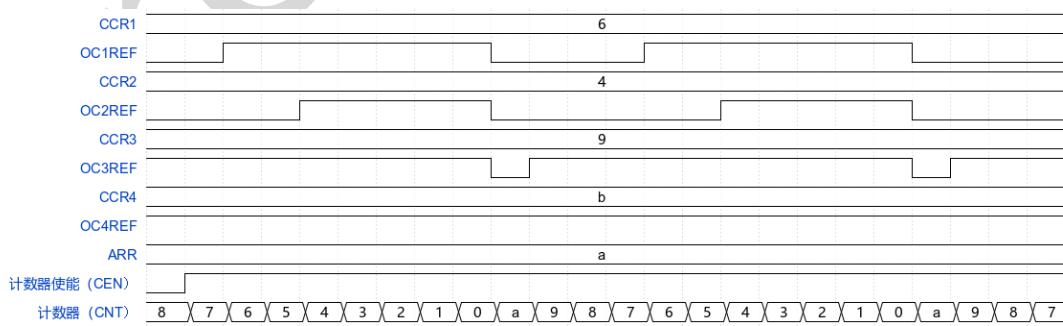


图 13-25 边沿对齐递减计数时 PWM 模式 1 的波形

### 13.6.3.3.3 PWM 中央对齐模式

首先配置 TIM3 计数器为中央对齐计数模式, 配置 TIM3\_CCMRx 寄存器 CCxS=00, 选择输出模式, 根据配置不同的 CMS, 输出比较中断标志位在计数器递增计数时被设置 (CMS=01)、在计数器递减计数时被设置 (CMS=10)、或在计数器递增或递减计数时被设置 (CMS=11)。图 16-26 为

CCR1=6, CCR2=4, CCR3=9, CCR4=b, ARR=a 时中央对齐 PWM 模式 1 (CMS=11) 的波形实例。

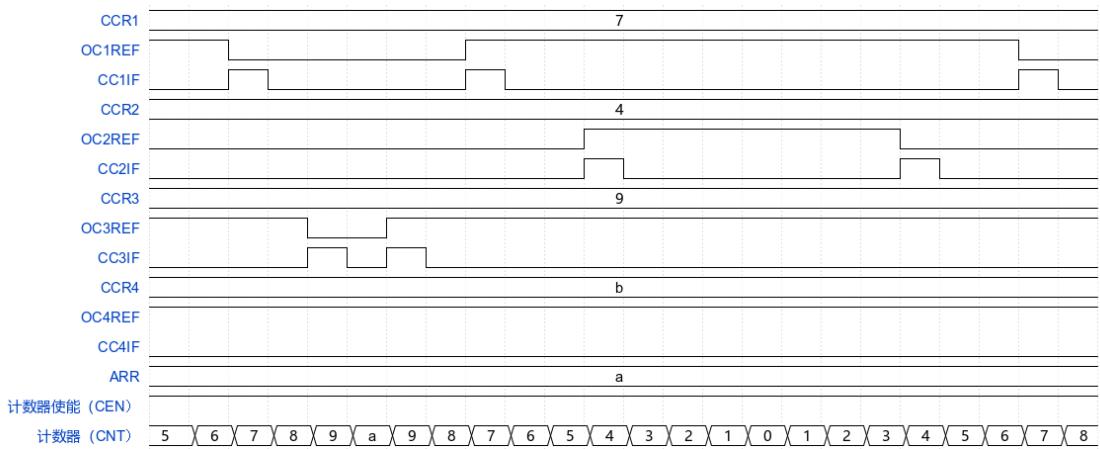


图 13-26 中央对齐 PWM 模式 1 的波形 (CMS=11)

### 13.6.3.4 单脉冲输出

单脉冲模式 (OPM) 是计数器只响应一个激励，延时后产生一个脉宽可调的脉冲。配置 TIM3\_CR1 寄存器的 OPM=1，选择单脉冲模式，触发信号有效沿或配置 CEN=1 都可以启动计数器，CEN=1 一直保持到下个更新事件发生或配置 CEN=0。

产生脉冲的必要条件是比较值与计数器的初始值不同。所以在计数器启动之前的必要配置如下：

- 1) 递增计数方式：计数器 CNT < CCRx  $\leq$  ARR。
- 2) 递减计数方式：计数器 CNT > CCRx。

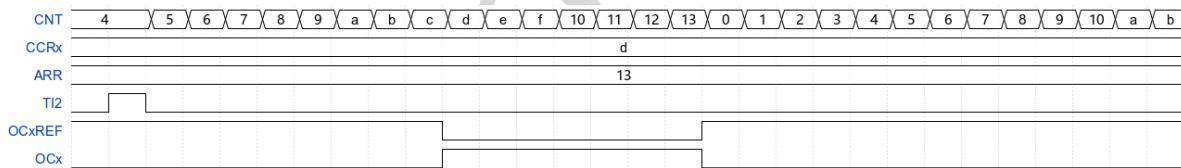


图 13-27 单脉冲模式的例子

例如，在 TI2 检测到上升沿，延迟  $t_{DELAY}$  之后，在 OC2 上产生一个长度为  $t_{PULSE}$  的正脉冲。

假定 TI2FP2 作为触发 1：

- 1) 配置 TIM3\_CCMR1 寄存器中的 CC2S = 01，将 TI2FP2 映射到 TI2。
- 2) 配置 TIM3\_CCER 寄存器中的 CC2P = 0，检测 TI2FP2 的上升沿。
- 3) 配置 TIM3\_SMCR 寄存器中的 TS = 110，TI2FP2 作为从模式控制器的触发 (TRGI)。
- 4) 配置 TIM3\_SMCR 寄存器中的 SMS = 110，选择触发模式，TI2FP2 使能计数器工作。

OPM 的波形由 TIM3\_ARR 和 TIM3\_CCR1 决定（要考虑时钟频率和计数器预分频器）：由 TIM3\_CCR2 寄存器的值和 CNT 初始值决定  $t_{DELAY}$ ；TIM3\_ARR - TIM3\_CCR1 的值为  $t_{PULSE}$ 。

当发生比较匹配时要产生从 1 到 0 的波形，当计数器达到预装载值时要产生一个从 0 到 1 的波形：

- 1) 配置 TIM3\_CCMR1 寄存器 OC1M = 111，选择 PWM 模式 2。
- 2) 配置 TIM3\_CCER 寄存器 CC2P = 1，输出低电平有效。

- 3) 配置 TIM3\_CCMR1 中 OC1PE = 1 和 TIM3\_CR1 寄存器中 ARPE，使能预装载寄存器。
- 4) 配置 TIM3\_CCR1 寄存器和 TIM3\_ARR 寄存器。
- 5) 配置 TIM3\_EGR 寄存器 UG=1 产生一个更新事件。
- 6) 等待在 TI2 上的一个外部触发事件。

此例中，TIM3\_CR1 寄存器中的 DIR=0、CMS=0、OPM=1，在下一个更新事件（当计数器从自动装载值翻转到 0）时停止计数。

#### 13.6.3.4.1 OCx 快速使能

OCx 快速使能，是单脉冲模式的一种特殊情况。在单脉冲模式下，通过设置 TIM3\_CCMR 寄存器的 OCxFE=1，强制 OCxREF 直接响应激励而不是依赖计数器和比较值之间的比较结果，输出波形和比较匹配时的波形一样。这样可以去除比较的时间，快速输出比较结果。OCx 快速输出使能只在 PWM 模式下生效。

### 13.6.4 DMA 模式

TIM3 能够在发生单个事件时生成多个 DMA 请求。主要目的是在没有软件开销的情况下，多次重新编程 TIM3 的一部分也可以用于按周期读取数个寄存器。

TIM3\_DCR 和 TIM3\_DMAR 寄存器跟 DMA 模式相关。DMA 控制器的目标是唯一的，必须指向 TIM3\_DMAR 寄存器。开启 DMA 使能后，在给定的 TIM3 事件发生时，TIM3 会给 DMA 发送请求。对 TIM3\_DMAR 寄存器的每次写操作都被重定向到一个 TIM3 寄存器。

TIM3\_DCR 寄存器的 DBL 位定义了 DMA 连续传送的长度，即传输寄存器数量；当对 TIM3\_DMAR 进行读写操作时，定时器识别 DBL，确定传输的寄存器数量。TIM3\_DCR 寄存器的 DBA 位定义了 DMA 传输的基址，定义从 TIM3\_CR1 寄存器地址开始的偏移量（00000 为 TIM3\_CR1；00001 为 TIM3\_CR2；...；00110 为 TIM3\_CCMR1 等）。

例：DMA 模式用于在发生更新事件时更新 CCR1、CCR2、CCR3 寄存器的内容。具体配置如下：

- 1) 配置相应的 DMA 通道。
- 2) 配置 TIM3\_DCR 寄存器 DBA=01101，配置 DMA 的基址，选择偏移地址为 TIM3\_CCR1 寄存器的地址。
- 3) 配置 TIM3\_DCR 寄存器 DBL=00010，配置传输长度为 3。
- 4) 配置 TIM3\_DIER 寄存器 UDE=1，允许更新事件的 DMA 请求。
- 5) 配置 TIM3\_CR1 寄存器 CEN=1，启动计数器。
- 6) 使能 DMA 通道。

此例中 TIM3 的 CCR1、CCR2、CCR3 更新一次，在第一个 DMA 请求时，数据 1 传到 CCR1 上；在第二个 DMA 请求时，数据 2 传到 CCR2 上；在第三个 DMA 请求时，数据 3 传到 CCR3 上。

### 13.6.5 调试模式

配置 DBG\_CR 寄存器中 DBG\_TIM3\_STOP=1，TIM3 计数器在 CPU 内核停止工作时（调试设置的

断点) 停止计数。(详见调试章节)

## 13.7 寄存器描述

表 13-3 TIM3 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	TIM3_CR1	控制寄存器 1	0x0000
0x04	TIM3_CR2	控制寄存器 2	0x0000
0x08	TIM3_SMCR	从模式控制寄存器	0x0000
0x0C	TIM3_DIER	DMA/中断使能寄存器	0x00000000
0x10	TIM3_SR	状态寄存器	0x00000000
0x14	TIM3_EGR	事件产生寄存器	0x00000000
0x18	TIM3_CCMR1	捕获/比较模式寄存器 1	0x0000
0x1C	TIM3_CCMR2	捕获/比较模式寄存器 2	0x0000
0x20	TIM3_CCER	捕获/比较使能寄存器	0x0000
0x24	TIM3_CNT	计数器	0x0000
0x28	TIM3_PSC	预分频率器	0x0000
0x2C	TIM3_ARR	自动装载寄存器	0x0000
0x34	TIM3_CCR1	捕获/比较寄存器 1	0x0000
0x38	TIM3_CCR2	捕获/比较寄存器 2	0x0000
0x3C	TIM3_CCR3	捕获/比较寄存器 3	0x0000
0x40	TIM3_CCR4	捕获/比较寄存器 4	0x0000
0x48	TIM3_DCR	DMA 控制寄存器	0x0000
0x4C	TIM3_DMAR	连续模式的 DMA 地址	0x0000

### 13.7.1 控制寄存器 1 (TIM3\_CR1)

偏移地址: 0x0

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.						CKD		APRE	CMS		DIR	OPM	URS	UDIS	CEN
Type							rw		rw	rw		rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:10	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
9:8	CKD	rw	0x0	<p>时钟分频 (clock division)</p> <p>在定时器时钟 (INT_CK) 频率、死区时间和由死区发生器与数字滤波器 (ETR, TIx) 所用的采样时钟之间的分频比例。</p> <p>00: tDTS = tINT_CK 01: tDTS = 2x tINT_CK 10: tDTS = 4x tINT_CK 11: 保留, 不要使用这个配置</p>
7	APRE	rw	0x0	<p>自动重装载预装载使能 (Auto-reload preload enable)</p> <p>0: 关闭 TIM3_ARR 寄存器的影子寄存器 1: 使能 TIM3_ARR 寄存器的影子寄存器</p>
6:5	CMS	rw	0x0	<p>中央对齐模式选择 (Center-aligned mode selection)</p> <p>00: 边沿对齐模式。计数方向取决于 DIR 位 01: 中央对齐模式 1。计数器交替地递增和递减计数。通道为输出模式, 只在计数器递减计数时比较中断标志位被置 1 10: 中央对齐模式 2。计数器交替地递增和递减计数。通道为输出模式, 只在计数器递增计数时比较中断标志位被置 1 11: 中央对齐模式 3。计数器交替地递增和递减计数。通道为输出模式, 在计数器递增和递减计数时比较中断标志位均被置 1 注: 计数过程中, 不允许对齐模式的更改。</p>
4	DIR	rw	0x0	<p>计数方向 (Direction)</p> <p>0: 计数器递增计数 1: 计数器递减计数 注: 当计数器配置为中央对齐模式或编码器模式时, 该位为只读。</p>
3	OPM	rw	0x0	<p>单脉冲模式 (One pulse mode)</p> <p>0: 禁止单脉冲模式, 在发生更新事件时, 计数器继续计数 1: 使能单脉冲模式, 在发生下一次更新事件 (清除 CEN 位) 时, 计数器停止计数</p>

Bit	Field	Type	Reset	Description
2	URS	rw	0x0	<p>更新请求源 (Update request source) 软件配置该位，选择更新事件源。</p> <p>0: 以下事件可产生一个更新中断或 DMA 请求:</p> <ul style="list-style-type: none"> <li>- 计数器溢出/下溢</li> <li>- 设置 UG 位</li> <li>- 从模式控制器产生的更新</li> </ul> <p>1: 只有计数器溢出/下溢才产生一个更新中断或 DMA 请求</p>
1	UDIS	rw	0x0	<p>禁止更新 (Update disable) 该位用来允许或禁止更新事件的产生</p> <p>0: 允许更新事件 (UEV)。更新事件源:</p> <ul style="list-style-type: none"> <li>- 计数器溢出/下溢</li> <li>- 设置 UG 位为 1</li> <li>- 从模式控制器产生一个更新事件。</li> </ul> <p>1: 禁止更新事件。不产生更新事件，影子寄存器 (ARR、PSC、CCRx) 保持值不变。如果设置了 EGR_UG 位为 1，或者从模式控制器接收到硬件复位，计数器和预分频器被更新为其对应的影子寄存器的值。</p>
0	CEN	rw	0x0	<p>允许计数器 (Counter enable)</p> <p>0: 禁止计数器</p> <p>1: 使能计数器。 注: 在软件设置了 CEN 位后, 外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。</p>

### 13.7.2 控制寄存器 2 (TIM3\_CR2)

偏移地址: 0x04

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.							TI1S	MMS		CCDS	Res.				
Type								rw	rw		rw					

Bit	Field	Type	Reset	Description
15:8	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
7	TI1S	rw	0x0	<p>TI1 选择 (TI1 selection)</p> <p>0: TIM3_CH1 管脚连到 TI1 输入</p> <p>1: TIM3_CH1、TIM3_CH2 和 TIM3_CH3 管脚经异或后 的结果作为 TI1 输入</p>
6:4	MMS	rw	0x0	<p>主模式选择 (Master mode selection)</p> <p>这些位控制 TRGO 信号的选择, 用于选择在主模式下送到 从定时器的同步信息:</p> <p>000: 复位 TIM3_EGR 寄存器的 UG 位或从模式控制器 产生复位触发一次 TRGO 脉冲。从模式控制器产生复位 时, TRGO 信号相对实际的复位会有一个延时。</p> <p>001: 使能 用于控制在一定时间内使能从定时器或同时 启动多个定时器。计数器使能信号 CNT_EN 被用于作为 触发输出 (TRGO), 计数器使能信号是通过 CEN 控制位 和门控模式下的触发输入信号的逻辑或产生。当计数器使 能信号受控于触发输入时, TRGO 上会有一个延迟, 除非 选择了主/从模式。</p> <p>010: 更新 更新事件被选为 TRGO。</p> <p>011: 捕获/比较脉冲 发生一次捕获或一次比较成功时, 触发输出送出一个 TRGO 信号。</p> <p>100 : 比较 OC1REF 信号被用于作为触发输出 (TRGO)</p> <p>101 : 比较 OC2REF 信号被用于作为触发输出 (TRGO)</p> <p>110 : 比较 OC3REF 信号被用于作为触发输出 (TRGO)</p> <p>111 : 比较 OC4REF 信号被用于作为触发输出 (TRGO)</p>
3	CCDS	rw	0x0	<p>DMA 请求源选择 (Capture/compare DMA selection)</p> <p>0: 当 CCx 发生捕获/比较事件时, 发送 CCx 的 DMA 请 求</p> <p>1: 当 CCx 发生更新事件时, 发送 CCx 的 DMA 请 求</p>
2:0	Reserved			保留, 始终读为 0

### 13.7.3 从模式控制寄存器 (TIM3\_SMCR)

偏移地址: 0x08

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ETP	ECE	ETPS		ETF				MSM	TS			OCCS	SMS		
Type	rw	rw	rw		rw				rw	rw			rw	rw		

Bit	Field	Type	Reset	Description
15	ETP	rw	0x0	外部触发极性 (External trigger polarity) 该位选择 ETR 信号的极性。 0: 高电平或上升沿有效 1: 低电平或下降沿有效
14	ECE	rw	0x0	外部时钟使能位 (External clock enable) 该位启用外部时钟模式 2。 0: 禁止外部时钟模式 2 1: 使能外部时钟模式 2, ETRF 信号上的任意有效沿驱动计数器计数。 注 1: 配置 ECE=1 与配置 SMS = 111 和 TS = 111 效果一样。 注 2: TS ≠ 111 时, 复位模式, 门控模式和触发模式可以与外部时钟模式 2 同时使用。 注 3: 同时使能外部时钟模式 1 和外部时钟模式 2 时, 外部时钟的输入是 ETRF。
13: 12	ETPS	rw	0x0	外部触发预分频 (External trigger prescaler) 外部触发信号 ETRP 的频率必须低于 TIM3_CLK 频率的 1/4。当输入较快的外部时钟时, 可以使用预分频降低 ETRP 的频率。 00: 关闭预分频 01: ETRP 频率除以 2 10: ETRP 频率除以 4 11: ETRP 频率除以 8

Bit	Field	Type	Reset	Description
11: 8	ETF	rw	0x0	<p>外部触发滤波 (External trigger filter)</p> <p>这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。</p> <p>0000: 无滤波器，以 <math>f_{DTS}</math> 采样</p> <p>0001: 采样频率 <math>f_{SAMPLING}=f_{INT\_CK}, N=2</math></p> <p>0010: 采样频率 <math>f_{SAMPLING}=f_{INT\_CK}, N=4</math></p> <p>0011: 采样频率 <math>f_{SAMPLING}=f_{INT\_CK}, N=8</math></p> <p>0100: 采样频率 <math>f_{sampling}=f_{DTS}/2, N=6</math></p> <p>0101: 采样频率 <math>f_{sampling}=f_{DTS}/2, N=8</math></p> <p>0110: 采样频率 <math>f_{sampling}=f_{DTS}/4, N=6</math></p> <p>0111: 采样频率 <math>f_{sampling}=f_{DTS}/4, N=8</math></p> <p>1000: 采样频率 <math>f_{sampling}=f_{DTS}/8, N=6</math></p> <p>1001: 采样频率 <math>f_{sampling}=f_{DTS}/8, N=8</math></p> <p>1010: 采样频率 <math>f_{sampling}=f_{DTS}/16, N=5</math></p> <p>1011: 采样频率 <math>f_{sampling}=f_{DTS}/16, N=6</math></p> <p>1100: 采样频率 <math>f_{sampling}=f_{DTS}/16, N=8</math></p> <p>1101: 采样频率 <math>f_{sampling}=f_{DTS}/32, N=5</math></p> <p>1110: 采样频率 <math>f_{sampling}=f_{DTS}/32, N=6</math></p> <p>1111: 采样频率 <math>f_{sampling}=f_{DTS}/32, N=8</math></p>
7	MSM	rw	0x0	<p>主/从模式 (Master/Slave mode)</p> <p>0: 无作用</p> <p>1: 触发输入 (TRGI) 上的事件被延迟了，以允许在当前定时器 (通过 TRGO) 与它的从定时器间的完美同步，这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。</p>

Bit	Field	Type	Reset	Description
6: 4	TS	rw	0x0	<p>触发选择 (Trigger selection)</p> <p>触发输入源选择。</p> <p>000: 内部触发 0 (ITR0)      001: 内部触发 1 (ITR1)      010: 内部触发 2 (ITR2)      011: 内部触发 3 (ITR3)      100: TI1 的边沿检测器 (TI1F_ED)      101: 滤波后的定时器输入 1 (TI1FP1)      110: 滤波后的定时器输入 2 (TI1FP2)      111: 外部触发输入 (ETRF)</p> <p>更多有关 ITRx 的细节, 参见下表。</p> <p>注: 从模式使能后这些位不能修改</p>
3	OCCS	rw	0x0	<p>比较器输出信号清除选择 (Output compare clear selection)</p> <p>在 PWM 模式下, 清除比较器输出</p> <p>0: 外部触发信号作为清除信号      1: 比较器输出作为清除信号</p>

Bit	Field	Type	Reset	Description
2: 0	SMS	rw	0x0	<p>从模式选择 (Slave mode selection)</p> <p>当选择了外部信号，触发信号 (TRGI) 的有效边沿与选中的外部输入极性相关。</p> <p>000: 关闭从模式 - 如果 CEN = 1，则预分频器直接由内部时钟驱动。</p> <p>001: 编码器模式 1- 根据 TI1FP1 的电平，计数器在 TI2FP2 的边沿递增/递减计数。</p> <p>010: 编码器模式 2- 根据 TI2FP2 的电平，计数器在 TI1FP1 的边沿递增/递减计数。</p> <p>011: 编码器模式 3 - 根据另一个输入的电平，计数器在 TI1FP1 和 TI2FP2 的边沿递增/递减计数。</p> <p>100: 复位模式 - 选中的触发输入 (TRGI) 的上升沿重新初始化计数器，并且产生一个更新寄存器的信号。</p> <p>101: 门控模式 - 当触发输入 (TRGI) 为高时，计数器开始计数。当触发输入变为低时，计数器停止计数 (但不复位)。计数器的启动和停止都是受控的。</p> <p>110: 触发模式 - 计数器在触发输入 TRGI 的上升沿启动 (但不复位)，只有计数器的启动是受控的。</p> <p>111: 外部时钟模式 1 - 选中的触发输入 (TRGI) 的上升沿驱动计数器。</p> <p>注：如果 TI1F_EN 被选为触发输入 (TS = 100) 时，不要使用门控模式。这是因为，TI1F_EO 在每次 TI1F 变化时输出一个脉冲，然而门控模式是要检查触发输入的电平。</p>

表 13-4 TIM2/3 内部触发连接

从定时器	ITR0	ITR1	ITR2	ITR3
TIM2	TIM1	TIM8	TIM3	TIM4
TIM3	TIM1	TIM2	TIM5	TIM4

### 13.7.4 DMA/中断使能寄存器 (TIM3\_DIER)

偏移地址: 0x0C

复位值: 0x0000 0000

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

Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Re s.	TDE	Res.	CC4 DE	CC3 DE	CC2 DE	CC1 DE	UD E	Re s.	TIE s.	Re s.	CC4 IE	CC3 IE	CC2 IE	CC 1IE	UIE
Type		rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:15	Reserved			保留, 始终读为 0
14	TDE	rw	0x0	允许触发 DMA 请求 (Trigger DMA request enable) 0: 禁止触发 DMA 请求 1: 允许触发 DMA 请求
13	Reserved			保留, 始终读为 0
12	CC4DE	rw	0x0	允许捕获/比较 4 的 DMA 请求 (Capture/Compare 4 DMA request enable) 0: 禁止捕获/比较 4 的 DMA 请求 1: 允许捕获/比较 4 的 DMA 请求
11	CC3DE	rw	0x0	允许捕获/比较 3 的 DMA 请求 (Capture/Compare 3 DMA request enable) 0: 禁止捕获/比较 3 的 DMA 请求 1: 允许捕获/比较 3 的 DMA 请求
10	CC2DE	rw	0x0	允许捕获/比较 2 的 DMA 请求 (Capture/Compare 2 DMA request enable) 0: 禁止捕获/比较 2 的 DMA 请求 1: 允许捕获/比较 2 的 DMA 请求
9	CC1DE	rw	0x0	允许捕获/比较 1 的 DMA 请求 (Capture/Compare 1 DMA request enable) 0: 禁止捕获/比较 1 的 DMA 请求 1: 允许捕获/比较 1 的 DMA 请求
8	UDE	rw	0x0	允许更新的 DMA 请求 (Update DMA request enable)

Bit	Field	Type	Reset	Description
				0: 禁止更新 DMA 请求 1: 允许更新 DMA 请求
7	Reserved			保留, 始终读为 0
6	TIE	rw	0x0	允许触发中断 (Trigger interrupt enable) 0: 禁止触发中断 1: 允许触发中断
5	Reserved			保留, 始终读为 0
4	CC4IE	rw	0x0	允许捕获/比较 4 中断 (Capture/Compare 4 interrupt enable) 0: 禁止捕获/比较 4 中断 1: 允许捕获/比较 4 中断
3	CC3IE	rw	0x0	允许捕获/比较 3 中断 (Capture/Compare 3 interrupt enable) 0: 禁止捕获/比较 3 中断 1: 允许捕获/比较 3 中断
2	CC2IE	rw	0x0	允许捕获/比较 2 中断 (Capture/Compare 2 interrupt enable) 0: 禁止捕获/比较 2 中断 1: 允许捕获/比较 2 中断
1	CC1IE	rw	0x0	允许捕获/比较 1 中断 (Capture/Compare 1 interrupt enable) 0: 禁止捕获/比较 1 中断 1: 允许捕获/比较 1 中断
0	UIE	rw	0x0	允许更新中断 (Update interrupt enable) 0: 禁止更新中断 1: 允许更新中断

### 13.7.5 状态寄存器 (TIM3\_SR)

偏移地址: 0x10

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.			CC4 OF	CC3 OF	CC2 OF	CC1 OF	Res.		TIF	Res.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
Type	r_w0c							r_w0c		r_w0c						

Bit	Field	Type	Reset	Description
31:13	Reserved			保留, 始终读为 0
12	CC4OF	r_w0c	0x0	捕获/比较 4 重复捕获标记 (Capture/Compare 4 overcapture flag) 参考 CC1OF 描述。
11	CC3OF	r_w0c	0x0	捕获/比较 3 重复捕获标记 (Capture/Compare 3 overcapture flag) 参考 CC1OF 描述。
10	CC2OF	r_w0c	0x0	捕获/比较 2 重复捕获标记 (Capture/Compare 2 overcapture flag) 参考 CC1OF 描述。
9	CC1OF	r_w0c	0x0	捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当通道 1 被配置为输入捕获, CC1F 已经位 1 后, 捕获事件再次发生时, 该标记可由硬件置 1。写 0 可清除该位。 0: 无重复捕获产生; 1: 重复捕获产生。
8:7	Reserved			保留, 始终读为 0
6	TIF	r_w0c	0x0	触发器中断标记 (Trigger interrupt flag) 当发生触发事件 (当从模式控制器处于除门控模式外的其它模式时, 在 TRGI 输入端检测到有效边沿, 或门控模式下的任一边沿) 时由硬件对该位置 1。它由软件清 0。 0: 无触发器事件产生 1: 触发器中断产生
5	Reserved			保留, 始终读为 0
4	CC4IF	r_w0c	0x0	捕获/比较 4 中断标记 (Capture/Compare 4 interrupt flag) 参考 CC1IF 描述。
3	CC3IF	r_w0c	0x0	捕获/比较 3 中断标记 (Capture/Compare 3 interrupt flag) 参考 CC1IF 描述。

Bit	Field	Type	Reset	Description
2	CC2IF	r_w0c	0x0	<p>捕获/比较 2 中断标记 (Capture/Compare 2 interrupt flag)</p> <p>参考 CC1IF 描述。</p>
1	CC1IF	r_w0c	0x0	<p>捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag)</p> <p>通道 1 为输出模式:</p> <p>当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外。它由软件清 0。</p> <p>0: 无匹配发生</p> <p>1: TIM3_CNT 的值与 TIM3_CCR1 的值匹配。</p> <p>通道 1 为输入模式:</p> <p>当发生捕获事件时该位由硬件置 1，由软件或读取 TIM3_CCR1 的值清 0。</p> <p>0: 无输入捕获产生</p> <p>1: 计数器值已被捕获至 TIM3_CCR1</p>
0	UIF	r_w0c	0x0	<p>更新中断标记 (Update interrupt flag)</p> <p>当产生更新事件时该位由硬件置 1。它由软件清 0。</p> <p>0: 无更新中断发生</p> <p>1: 发生更新中断。</p> <p>当寄存器被更新时该位由硬件置 1:</p> <ul style="list-style-type: none"> <li>-若 TIM3_CR1 寄存器的 UDIS=0、URS=0，当 TIM3_EGR 寄存器的 UG=1 时产生更新事件。</li> <li>- 若 TIM3_CR1 寄存器的 UDIS=0、URS=0，当计数器被触发事件重初始化时产生更新事件。</li> </ul>

### 13.7.6 事件产生寄存器 (TIM3\_EGR)

偏移地址: 0x14

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0
6	TG	w	0x0	<p>产生触发事件 (Trigger generation)</p> <p>0: 无动作</p> <p>1: 产生触发事件, TIM3_SR 寄存器的 TIF = 1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA, 由硬件自动清 0。</p>
5	COMG	w	0x0	<p>捕获/比较事件, 产生控制更新 (Capture/Compare control update generation)</p> <p>0: 无动作</p> <p>1: 捕获/比较事件控制更新产生, 由硬件自动清 0, 当 CCPC=1, 允许更新 CCxE、CCxNE、OCxM 位。</p> <p>注: 该位只对拥有互补输出的通道有效。</p>
4	CC4G	w	0x0	<p>产生捕获/比较 4 事件 (Capture/Compare 4 generation)</p> <p>参考 CC1G 描述。</p>
3	CC3G	w	0x0	<p>产生捕获/比较 3 事件 (Capture/Compare 3 generation)</p> <p>参考 CC1G 描述。</p>
2	CC2G	w	0x0	<p>产生捕获/比较 2 事件 (Capture/Compare 2 generation)</p> <p>参考 CC1G 描述。</p>
1	CC1G	w	0x0	<p>产生通道 1 捕获/比较事件 (Capture/Compare 1 generation)</p> <p>该位由软件置 1, 用于产生一个捕获/比较事件, 由硬件自动清 0。</p> <p>0: 无动作</p> <p>1: 在通道 CC1 上产生一个捕获/比较事件:</p> <p>若通道 CC1 配置为输出: 设置 CC1IF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。</p> <p>若通道 CC1 配置为输入: 当前的计数器值被捕获至 TIM3_CCR1 寄存器, 设置 CC1IF = 1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。若 CC1IF 已经为 1, 则设置 CC1OF = 1。</p>

Bit	Field	Type	Reset	Description
0	UG	w	0x0	<p>产生更新事件 (Update generation)</p> <p>0: 无动作</p> <p>1: 重置计数器值，并产生一个更新事件。由硬件自动清 0，如果选择了中央对齐或递增计数模式，计数器被清 0；否则(递减计数模式)计数器将载入自动重载值。预分频计数器将同时被清除。</p>

### 13.7.7 捕获/比较模式寄存器 (TIM3\_CCMR1)

偏移地址: 0x18

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	OC2CE	OC2M		OC2PE	OC2FE	CC2S	OC1CE	OC1M		OC1PE	OC1FE	CC1S				
IC2F			IC2PSC		IC1F		IC1PSC									
Type	rw	rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式:

Bit	Field	Type	Reset	Description
15	OC2CE	rw	0x0	通道 2 输出比较清零使能 (Output compare 2 clear enable)
14:12	OC2M	rw	0x0	通道 2 输出比较模式 (Output compare 2 mode)
11	OC2PE	rw	0x0	通道 2 输出比较预装载使能 (Output compare 2 preload enable)
10	OC2FE	rw	0x0	通道 2 输出比较快速使能 (Output compare 2 fast enable)
9:8	CC2S	rw	0x0	<p>通道 2 捕获/比较选择 (Capture/Compare 2 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <p>00: 通道 2 被配置为输出</p> <p>01: 通道 2 被配置为输入，IC2 映射在 TI2 上</p> <p>10: 通道 2 被配置为输入，IC2 映射在 TI1 上</p> <p>11: 通道 2 被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIM3_SMCR 寄存器的 TS 位选择)</p>
7	OC1CE	rw	0x0	<p>通道 1 输出比较清 0 使能 (Output compare 1 clear enable)</p> <p>0: OC1REF 不受 ETRF 输入的影响</p> <p>1: 当检测到 ETRF 输入高电平时，清除 OC1REF = 0</p>

Bit	Field	Type	Reset	Description
6:4	OC1M	rw	0x0	<p>通道 1 输出比较模式 (Output compare 1 mode)</p> <p>该位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。</p> <p>000：冻结。TIM3_CCR1 与 TIM3_CNT 间的比较结果对 OC1REF 不起作用</p> <p>001：匹配时设置为高。当 TIM3_CNT 的值与 TIM3_CCR1 的值相同时，强制 OC1REF 为高电平</p> <p>010：匹配时设置为低。当 TIM3_CNT 的值与 TIM3_CCR1 的值相同时，强制 OC1REF 为低电平</p> <p>011：匹配时翻转。当 TIM3_CCR1=TIM3_CNT 时，翻转 OC1REF 的电平</p> <p>100：强制为低。强制 OC1REF 为低电平</p> <p>101：强制为高。强制 OC1REF 为高电平</p> <p>110：PWM 模式 1。在递增计数时，当 TIM3_CNT&lt;TIM3_CCR1 时通道 1 为有效电平，否则为无效电平；在递减计数时，当 TIM3_CNT&gt;TIM3_CCR1 时通道 1 为无效电平，否则为有效电平。</p> <p>111：PWM 模式 2。在递增计数时，当 TIM3_CNT &lt; TIM3_CCR1 时通道 1 为无效电平，否则为有效电平；在递减计数时，当 TIM3_CNT &gt; TIM3_CCR1 时通道 1 为有效电平，否则为无效电平</p> <p>注：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。</p>
3	OC1PE	rw	0x0	<p>通道 1 输出比较预装载使能 (Output compare 1 preload enable)</p> <p>0：禁止 TIM3_CCR1 寄存器的预装载功能，写入 TIM3_CCR1 寄存器的数值立即生效</p> <p>1：开启 TIM3_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIM3_CCR1 的预装载值在更新事件到来时生效</p> <p>注：仅在单脉冲模式下 (TIM3_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。</p>

Bit	Field	Type	Reset	Description
2	OC1FE	rw	0x0	<p>通道 1 输出比较快速使能 (Output compare 1 fast enable)</p> <p>该位为 1 时, 若通道配置为 PWM 模式, 会加快捕获/比较输出对触发时间的响应。输出通道将触发输入信号的有效边沿的作用等同于发生了一次比较匹配, 因此 OC 被设置为比较电平而与比较结果无关。</p> <p>0: 禁止通道 1 输出比较快速使能, 当检测到触发输入的一个有效边沿时, 激活 OC1 的最小延时需要 5 个时钟周期</p> <p>1: 开启通道 1 输出比较快速使能。当触发输入由一个有效边沿时, 激活 OC1 的最小延时缩短到 3 个周期</p>
1:0	CC1S	rw	0x0	<p>通道 1 捕获/比较选择 (Capture/Compare 1 selection)</p> <p>该位定义通道的方向和输入信号的选择, 只有在通道关闭时这些位才可写入:</p> <p>00: 通道 1 被配置为输出</p> <p>01: 通道 1 被配置为输入, IC1 映射在 TI1 上</p> <p>10: 通道 1 被配置为输入, IC1 映射在 TI2 上</p> <p>11: 通道 1 被配置为输入, IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIM3_SMCR 寄存器的 TS 位选择)</p>

输入捕获模式:

Bit	Field	Type	Reset	Description
15:12	IC2F	rw	0x0	输入捕获 2 滤波器 (Input capture 2 filter)
11:10	IC2PSC	rw	0x0	输入/捕获 2 预分频器 (Input capture 2 prescaler)
9:8	CC2S	rw	0x0	<p>捕获/比较 2 选择 (Capture/Compare 2 selection)</p> <p>该位定义通道的方向和输入信号的选择, 只有在通道关闭时这些位才可写入:</p> <p>00: CC2 通道被配置为输出</p> <p>01: CC2 通道被配置为输入, IC2 映射在 TI2 上</p> <p>10: CC2 通道被配置为输入, IC2 映射在 TI1 上</p> <p>11: CC2 通道被配置为输入, IC2 映射在 TRC 上,</p> <p>此模式仅工作在内部触发器输入被选中时 (由 TIM3_SMCR 寄存器的 TS 位选择)</p>

Bit	Field	Type	Reset	Description
7:4	IC1F	rw	0x0	<p>通道 1 输入捕获滤波器 (Input capture 1 filter)</p> <p>数字滤波器由一个事件计数器组成, 它记录 N 个输入事件后会产生一个输出的跳变。这些位定义了 IC1 输入信号的采样频率和数字滤波器的长度。</p> <p>0000: 无滤波器, 以 <math>f_{DTS}</math> 采样</p> <p>1000: 采样频率 <math>f_{SAMPLING}=f_{DTS}/8</math>, <math>N=6</math></p> <p>0001: 采样频率 <math>f_{SAMPLING}=f_{INT\_CK}</math>, <math>N=2</math></p> <p>1001: 采样频率 <math>f_{SAMPLING}=f_{DTS}/8</math>, <math>N=8</math></p> <p>0010: 采样频率 <math>f_{SAMPLING}=f_{INT\_CK}</math>, <math>N=4</math></p> <p>1010: 采样频率 <math>f_{SAMPLING}=f_{DTS}/16</math>, <math>N=5</math></p> <p>0011: 采样频率 <math>f_{SAMPLING}=f_{INT\_CK}</math>, <math>N=8</math></p> <p>1011: 采样频率 <math>f_{SAMPLING}=f_{DTS}/16</math>, <math>N=6</math></p> <p>0100: 采样频率 <math>f_{SAMPLING}=f_{DTS}/2</math>, <math>N=6</math></p> <p>1100: 采样频率 <math>f_{SAMPLING}=f_{DTS}/16</math>, <math>N=8</math></p> <p>0101: 采样频率 <math>f_{SAMPLING}=f_{DTS}/2</math>, <math>N=8</math></p> <p>1101: 采样频率 <math>f_{SAMPLING}=f_{DTS}/32</math>, <math>N=5</math></p> <p>0110: 采样频率 <math>f_{SAMPLING}=f_{DTS}/4</math>, <math>N=6</math></p> <p>1110: 采样频率 <math>f_{SAMPLING}=f_{DTS}/32</math>, <math>N=6</math></p> <p>0111: 采样频率 <math>f_{SAMPLING}=f_{DTS}/4</math>, <math>N=8</math></p> <p>1111: 采样频率 <math>f_{SAMPLING}=f_{DTS}/32</math>, <math>N=8</math></p>
3:2	IC1PSC	rw	0x0	<p>通道 1 输入/捕获预分频器 (Input capture 1 prescaler)</p> <p>这 2 位定义了 IC1 的预分频系数。当 CC1E=0 (TIM3_CCER 寄存器中) 时, 预分频器复位。</p> <p>00: 无预分频器, 捕获输入上检测到的每一个边沿都触发一次捕获</p> <p>01: 每 2 个事件触发一次捕获</p> <p>10: 每 4 个事件触发一次捕获</p> <p>11: 每 8 个事件触发一次捕获</p>

Bit	Field	Type	Reset	Description
1:0	CC1S	rw	0x0	<p>捕获/比较 1 选择 (Capture/Compare 1 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <p>00: CC1 通道被配置为输出</p> <p>01: CC1 通道被配置为输入, IC1 映射在 TI1 上</p> <p>10: CC1 通道被配置为输入, IC1 映射在 TI2 上</p> <p>11: CC1 通道被配置为输入, IC1 映射在 TRC 上, 此模式仅工作在内部触发器输入被选中时 (由 TIM3_SMCR 寄存器的 TS 位选择)</p>

### 13.7.8 捕获/比较模式寄存器 2 (TIM3\_CCMR2)

偏移地址: 0x1C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	OC4CE	OC4M			OC4PE	OC4FE	CC4S	OC3CE	OC3M	OC3PE	OC3FE	CC3S				
Field	IC4F				IC4PSC			IC3F	IC3PSC							
Type	rw	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

输出比较模式:

Bit	Field	Type	Reset	Description
15	OC4CE	rw	0x0	通道 4 输出比较清零使能 (Output compare 4 clear enable)
14:12	OC4M	rw	0x0	通道 4 输出比较模式 (Output compare 4 mode)
11	OC4PE	rw	0x0	通道 4 输出比较预装载使能 (Output compare 4 preload enable)
10	OC4FE	rw	0x0	通道 4 输出比较快速使能 (Output compare 4 fast enable)
9:8	CC4S	rw	0x0	<p>通道 4 捕获/比较选择 (Capture/Compare 4 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <p>00: 通道 4 被配置为输出</p> <p>01: 通道 4 被配置为输入, IC4 映射在 TI4 上</p> <p>10: 通道 4 被配置为输入, IC4 映射在 TI3 上</p> <p>11: 通道 4 被配置为输入, IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIM3_SMCR 寄存器的 TS 位选择)</p>

Bit	Field	Type	Reset	Description
7	OC3CE	rw	0x0	<p>通道 3 输出比较清 0 使能 (Output compare 3 clear enable)</p> <p>0: OC1REF 不受 ETRF 输入的影响</p> <p>1: 当检测到 ETRF 输入高电平时, 清除 OC1REF = 0</p>
6:4	OC3M	rw	0x0	<p>通道 3 输出比较模式 (Output compare 3 mode)</p> <p>该位定义了输出参考信号 OC3REF 的动作, 而 OC3REF 决定了 OC3、OC3N 的值。OC3REF 是高电平有效, 而 OC3、OC3N 的有效电平取决于 CC3P、CC3NP 位。</p> <p>000: 冻结。TIM3_CCR3 与 TIM3_CNT 间的比较结果对 OC3REF 不起作用</p> <p>001 : 匹配时设置为高。当 TIM3_CNT 的值与 TIM3_CCR3 的值相同时, 强制 OC3REF 为高电平</p> <p>010 : 匹配时设置为低。当 TIM3_CNT 的值与 TIM3_CCR3 的值相同时, 强制 OC3REF 为低电平</p> <p>011: 匹配时翻转。当 TIM3_CCR3=TIM3_CNT 时, 翻转 OC3REF 的电平</p> <p>100: 强制为低。强制 OC3REF 为低电平</p> <p>101: 强制为高。强制 OC3REF 为高电平</p> <p>110: PWM 模式 1。在递增计数时, 当 TIM3_CNT&lt;TIM3_CCR3 时通道 3 为有效电平, 否则为无效电平; 在递减计数时, 当 TIM3_CNT&gt;TIM3_CCR3 时通道 3 为无效电平, 否则为有效电平。</p> <p>111: PWM 模式 2。在递增计数时, 当 TIM3_CNT&lt;TIM3_CCR3 时通道 3 为无效电平, 否则为有效电平; 在递减计数时, 当 TIM3_CNT&gt;TIM3_CCR3 时通道 3 为有效电平, 否则为无效电平</p> <p>注: 在 PWM 模式 1 或 PWM 模式 2 中, 只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时, OC3REF 电平才改变。</p>
3	OC3PE	rw	0x0	<p>通道 3 输出比较预装载使能 (Output compare 3 preload enable)</p> <p>0: 禁止 TIM3_CCR3 寄存器的预装载功能, 写入 TIM3_CCR3 寄存器的数值立即生效</p> <p>1: 开启 TIM3_CCR3 寄存器的预装载功能, 读写操作仅对预装载寄存器操作, TIM3_CCR3 的预装载值在更新事件到来时生效</p> <p>注: 仅在单脉冲模式下 (TIM3_CR1 寄存器的 OPM= 1), 可以在未确认预装载寄存器情况下使用 PWM 模式, 否则其动作不确定。</p>

Bit	Field	Type	Reset	Description
2	OC3FE	rw	0x0	<p>通道 3 输出比较快速使能 (Output compare 3 fast enable)</p> <p>该位为 1 时，若通道配置为 PWM 模式，会加快捕获/比较输出对触发时间的响应。输出通道将触发输入信号的有效边沿的作用等同于发生了一次比较匹配，因此 OC 被设置为比较电平而与比较结果无关。</p> <p>0: 禁止通道 3 输出比较快速使能，当检测到触发输入的一个有效边沿时，激活 OC1 的最小延时需要 5 个时钟周期</p> <p>1: 开启通道 3 输出比较快速使能。当触发输入由一个有效边沿时，激活 OC1 的最小延时缩短到 3 个周期</p>
1:0	CC3S	rw	0x0	<p>通道 3 捕获/比较选择 (Capture/Compare 3 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <p>00: 通道 3 被配置为输出</p> <p>01: 通道 3 被配置为输入，IC3 映射在 TI3 上</p> <p>10: 通道 3 被配置为输入，IC3 映射在 TI4 上</p> <p>11: 通道 3 被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIM3_SMCR 寄存器的 TS 位选择)</p>

输入捕获模式：

Bit	Field	Type	Reset	Description
15:12	IC4F	rw	0x0	输入捕获 4 滤波器 (Input capture 4 filter)
11:10	IC4PSC	rw	0x0	输入/捕获 4 预分频器 (Input capture 4 prescaler)
9:8	CC4S	rw	0x0	<p>捕获/比较 4 选择 (Capture/Compare 4 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <p>00: CC4 通道被配置为输出</p> <p>01: CC4 通道被配置为输入，IC4 映射在 TI4 上</p> <p>10: CC4 通道被配置为输入，IC4 映射在 TI3 上</p> <p>11: CC4 通道被配置为输入，IC4 映射在 TRC 上， 此模式仅工作在内部触发器输入被选中时 (由 TIM3_SMCR 寄存器的 TS 位选择)</p>

Bit	Field	Type	Reset	Description
7:4	IC3F	rw	0x0	<p>通道 3 输入捕获滤波器 (Input capture 3 filter)</p> <p>数字滤波器由一个事件计数器组成，它记录 N 个输入事件后会产生一个输出的跳变。这些位定义了 IC1 输入信号的采样频率和数字滤波器的长度。</p> <p>0000: 无滤波器，以 <math>f_{DTS}</math> 采样</p> <p>1000: 采样频率 <math>f_{SAMPLING}=f_{DTS} / 8, N=6</math></p> <p>0001: 采样频率 <math>f_{SAMPLING}=f_{INT\_CK}, N=2</math></p> <p>1001: 采样频率 <math>f_{SAMPLING}=f_{DTS} / 8, N=8</math></p> <p>0010: 采样频率 <math>f_{SAMPLING}=f_{INT\_CK}, N=4</math></p> <p>1010: 采样频率 <math>f_{SAMPLING}=f_{DTS} / 16, N=5</math></p> <p>0011: 采样频率 <math>f_{SAMPLING}=f_{INT\_CK}, N=8</math></p> <p>1011: 采样频率 <math>f_{SAMPLING}=f_{DTS} / 16, N=6</math></p> <p>0100: 采样频率 <math>f_{SAMPLING}=f_{DTS} / 2, N=6</math></p> <p>1100: 采样频率 <math>f_{SAMPLING}=f_{DTS} / 16, N=8</math></p> <p>0101: 采样频率 <math>f_{SAMPLING}=f_{DTS} / 2, N=8</math></p> <p>1101: 采样频率 <math>f_{SAMPLING}=f_{DTS} / 32, N=5</math></p> <p>0110: 采样频率 <math>f_{SAMPLING}=f_{DTS} / 4, N=6</math></p> <p>1110: 采样频率 <math>f_{SAMPLING}=f_{DTS} / 32, N=6</math></p> <p>0111: 采样频率 <math>f_{SAMPLING}=f_{DTS} / 4, N=8</math></p> <p>1111: 采样频率 <math>f_{SAMPLING}=f_{DTS} / 32, N=8</math></p>
3:2	IC3PSC	rw	0x0	<p>通道 3 输入/捕获预分频器 (Input capture 3 prescaler)</p> <p>这 2 位定义了 IC3 的预分频系数。当 CC3E=0 (TIM3_CCER 寄存器中) 时，预分频器复位。</p> <p>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</p> <p>01: 每 2 个事件触发一次捕获</p> <p>10: 每 4 个事件触发一次捕获</p> <p>11: 每 8 个事件触发一次捕获</p>

Bit	Field	Type	Reset	Description
1:0	CC3S	rw	0x0	<p>捕获/比较 3 选择 (Capture/Compare 3 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <ul style="list-style-type: none"> <li>00: CC3 通道被配置为输出</li> <li>01: CC3 通道被配置为输入，IC3 映射在 TI3 上</li> <li>10: CC3 通道被配置为输入，IC3 映射在 TI4 上</li> <li>11: CC3 通道被配置为输入，IC3 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时（由 TIM3_SMCR 寄存器的 TS 位选择）</li> </ul>

### 13.7.9 捕获/比较使能寄存器 (TIM3\_CCER)

偏移地址: 0x20

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CC4N P	Res . .	C C 4P	C C 4E	CC3 NP	Res.	C C 3P	C C 3E	CC2N P	Res . .	C C 2P	CC2 E	CC1N P	Res . .	CC1 P	CC1 E
Type	rw			rw	rw		rw	rw	rw		rw	rw	rw		rw	rw

Bit	Field	Type	Reset	Description
15	CC4NP	rw	0x0	<p>通道 4 输入 / 捕获互补输出极性 (Capture/Compare 4 complementary output polarity)</p> <p>参考 CC1NP 的描述。</p>
14	Reserved			保留，始终读为 0。
13	CC4P	rw	0x0	<p>通道 4 输入/捕获输出极性 (Capture/Compare 4 output polarity)</p> <p>参考 CC1P 的描述。</p>
12	CC4E	rw	0x0	<p>通道 4 输入/捕获输出使能 (Capture/Compare 4 output enable)</p> <p>参考 CC1E 的描述。</p>
11	CC3NP	rw	0x0	<p>通道 3 输入/捕获互补输出极性 (Capture/Compare 3 complementary output polarity)</p> <p>参考 CC1NP 的描述。</p>
10	Reserved			保留，始终读为 0。

Bit	Field	Type	Reset	Description
9	CC3P	rw	0x0	通道 3 输入捕获输出极性 (Capture/Compare 3 output polarity) 参考 CC1P 的描述。
8	CC3E	rw	0x0	通道 3 输入/捕获输出使能 (Capture/Compare 3 output enable) 参考 CC1E 的描述。
7	CC2NP	rw	0x0	通道 2 输入/捕获互补输出极性 (Capture/Compare 2 complementary output polarity) 参考 CC1NP 的描述。
6	Reserved			保留, 始终读为 0。
5	CC2P	rw	0x0	通道 2 输入捕获输出极性 (Capture/Compare 2 output polarity) 参考 CC1P 的描述。
4	CC2E	rw	0x0	通道 2 输入/捕获输出使能 (Capture/Compare 2 output enable) 参考 CC1E 的描述。
3	CC1NP	rw	0x0	通道 1 输入 / 捕获互补输出极性 (Capture/Compare 1 complementary output polarity) 0: OC1N 高电平有效 1: OC1N 低电平有效
2	Reserved			保留, 始终读为 0。
1	CC1P	rw	0x0	通道 1 输入/捕获输出极性 (Capture/Compare 1 output polarity) 通道 1 配置为输出时, 此位定义了输出信号极性: 0: OC1 高电平有效 1: OC1 低电平有效 通道 1 配置为输入时, 此位定义了输入信号极性: 0: 不反相: 捕获发生在 IC1 的上升沿; 当用作外部触发器时, IC1 不反相。 1: 反相: 捕获发生在 IC1 的下降沿; 当用作外部触发器时, IC1 反相。

Bit	Field	Type	Reset	Description
0	CC1E	rw	0x0	<p>通道 1 输入/捕获输出使能 (Capture/Compare 1 output enable)</p> <p>通道 1 配置为输出时, :</p> <p>0: 关闭。OC1 禁止输出</p> <p>1: 开启。OC1 信号输出到对应的输出引脚.</p> <p>CC1 通道配置为输入:</p> <p>该位决定了输入捕获功能是否启用。</p> <p>0: 捕获禁止</p> <p>1: 捕获使能</p>

表 13-5 输入模式下, ICx 的极性选择如下表:

CCxP	CCxNP	ICx 极性
0	0	上升沿
1	0	下降沿
1	1	上升沿或下降沿
0	1	保留

### 13.7.10 计数器 (TIM3\_CNT)

偏移地址: 0x24

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CNT															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x0000	计数器的值 (Count value)

### 13.7.11 预分频器 (TIM3\_PSC)

偏移地址: 0x28

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	PSC															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	PSC	rw	0x0000	预分频器的值 (Prescaler value) 计数器的时钟频率 ( $\text{ck\_cnt} = \text{fCK\_PSC} / (\text{PSC} + 1)$ )。 当发生更新事件时, PSC 的值装入当前预分频寄存器。

### 13.7.12 自动预装载寄存器 (TIM3\_ARR)

偏移地址: 0x2C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ARR															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	ARR	rw	0x0000	自动预装载的值 (Auto-reload value) 这些位定义了计数器的自动重载值。当自动重装载的值为 0 时, 计数器不工作。

### 13.7.13 捕获/比较寄存器 1 (TIM3\_CCR1)

偏移地址: 0x34

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CCR1															
Type	rw															

Bit	Field	Type	Reset	Description
-----	-------	------	-------	-------------

15:0	CCR1	rw	0x0000	通道 1 捕获/比较的值 (Capture/Compare 1 value) 当通道 1 配置为输入时, CCR1 的值决定了上次捕获事件的计数器值 (此时寄存器为只读)。 当通道 1 配置为输出时, CCR1 的值为即将和计数器值比较的值。如果开启了预装载功能, 写入的值在更新事件发生时传输到当前捕获/比较寄存器; 否则写入的值立即载入捕获比较寄存器。
------	------	----	--------	--

### 13.7.14 捕获/比较寄存器 2 (TIM3\_CCR2)

偏移地址: 0x3C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CCR2															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CCR2	rw	0x0000	通道 2 捕获/比较的值 (Capture/Compare 2 value) 参考 CCR1 的描述。

### 13.7.15 捕获/比较寄存器 3 (TIM3\_CCR3)

偏移地址: 0x40

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CCR3															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CCR3	rw	0x0000	通道 3 捕获/比较的值 (Capture/Compare 3 value) 参考 CCR1 的描述。

### 13.7.16 捕获比较寄存器 4 (TIM3\_CCR4)

偏移地址: 0x40

复位值: 0x0000

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

Field	CCR4
Type	rw

Bit	Field	Type	Reset	Description
15:0	CCR4	rw	0x0000	通道 4 捕获/比较的值 (Capture/Compare 4 value) 参考 CCR1 的描述。

### 13.7.17 DMA 控制寄存器 (TIM3\_DCR)

偏移地址: 0x48

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.		DBL				Res.		DBA							
Type					rw								rw			

Bit	Field	Type	Reset	Description
15:13	Reserved			保留, 始终读为 0。
12:8	DBL	w	0x00	DMA 连续传送长度 (DMA burst length) 这些位定义了 DMA 在连续模式下的访问寄存器的数量 00000: 1 次传输 00001: 2 次传输 00010: 3 次传输 ..... ..... 10001: 18 次传输
7:5	Reserved			保留, 始终读为 0。
4:0	DBA	w	0x00	DMA 基址 (DMA base address) 这些位定义了 DMA 在连续模式下访问 TIM3_DMAR 寄存器的第一个地址。DBA 定义为从 TIM3_CR1 寄存器所在地址开始的偏移值： 00000: TIM3_CR1 00001: TIM3_CR2 00010: TIM3_SMCR .....

### 13.7.18 连续模式的 DMA 地址 (TIM3\_DMAR)

偏移地址: 0x4C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	DMAB															
Type	w															

Bit	Field	Type	Reset	Description
15:0	DMAB	w	0x0000	DMA 连续传送寄存器 (DMA register for burst accesses) 对 TIM3_DMAR 寄存器的写操作会导致对以下地址所在寄存器的存取操作: TIM1_CR1 地址 + DBA + DMA 索引, 其中: TIM3_CR1 地址是 TIM3_CR1 寄存器所在的地址;DBA 是 TIM3_DCR 寄存器中定义的基地址; DMA 索引是 DMA 自动控制的偏移量, 它取决于 TIM3_DCR 寄存器中定义的 DBL。

# 14 32位定时器 (TIM2/5)

## 14.1 TIM2 简介

TIM2 由一个 16 位可实时编程预分频器和一个 32 位计数方向可调的自动重装载计数器组成，可以为用户提供便捷的计数定时功能，计数器时钟由预分频器分频得到。通用定时器具有多种用途，如输入功能（测量输入信号的脉冲宽度、频率，PWM 输入等），输出功能（PWM 输出、单脉冲模式输出等）。

## 14.2 功能框图

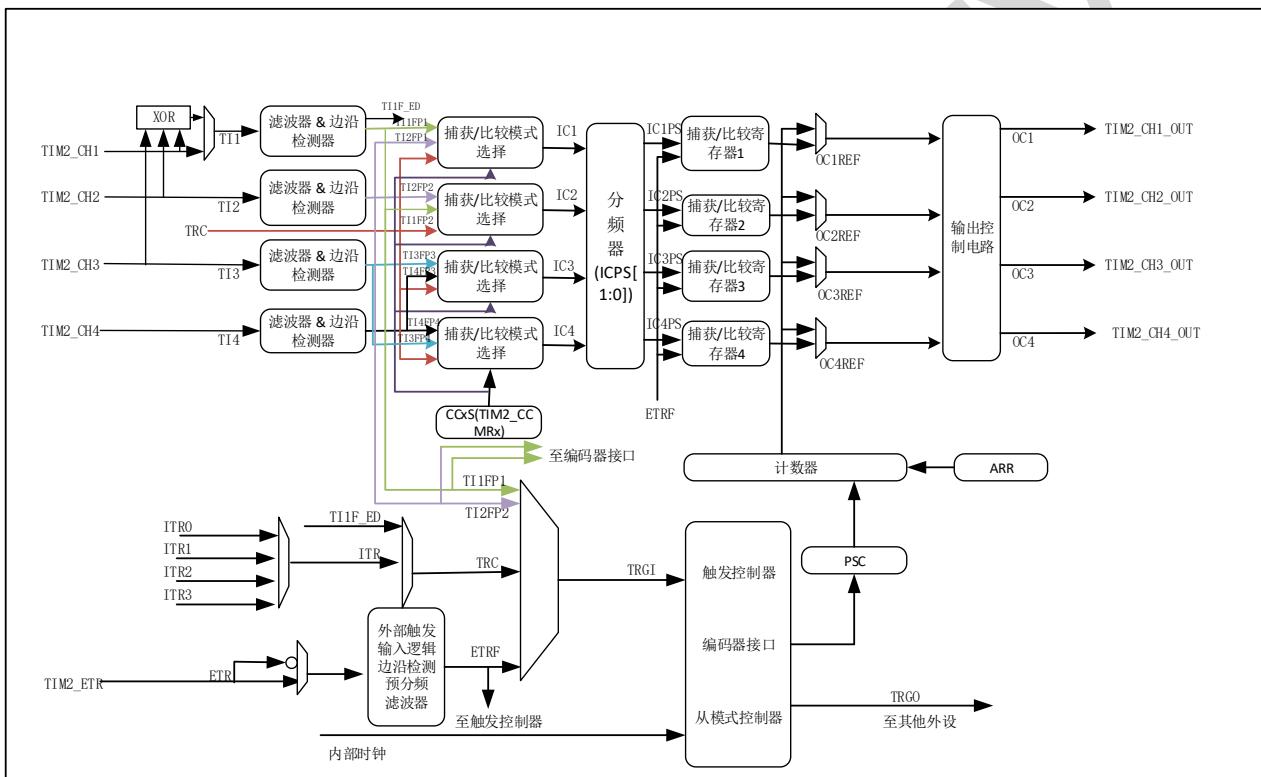


图 14-1 TIM2 结构图

上图为 TIM2 的结构框图，主要由输入单元、输出单元、时基单元、捕获/比较模块等结构组成。

## 14.3 主要特征

- 1) 16 位可实时编程预分频器，分频系数：1–65536 可调。
- 2) 时钟源可选：内部时钟源，外部时钟模式（外部时钟引脚/外部触发输入 ETR），内部触发输入（编码器模式）。
- 3) 32 位自动重装载计数器（计数方向：递增、递减、递增/递减）。
- 4) 输入捕获：输入信号的脉冲宽度、周期的测量。
- 5) 触发输入可以作为外部时钟或者逐周期管理。
- 6) 支持编码器、霍尔传感器等接口。
- 7) 4 个输出通道。

- 8) 输出比较（控制输出波形或指示定时器计时结束）。
- 9) PWM 输出（死区时间可调）（边沿对齐或中央对齐模式）。
- 10) 单脉冲输出。
- 11) 产生中断/DMA 请求的事件：更新事件、触发事件、输入捕获、输出比较或者刹车输入时。

注：

- 更新事件：计数器上溢/下溢，计数器初始化。
- 触发事件：计数器初始化，启动或停止计数。

## 14.4 中断

TIM2 的中断包括：捕获/比较 1 中断、捕获/比较 2 中断、捕获/比较 3 中断、捕获/比较 4 中断、更新中断和触发中断，当相应的中断使能位打开，发生相应的事件时，产生相应的中断。

## 14.5 DMA

TIM2 能够在发生单个事件时生成多个 DMA 请求。主要目的是在没有软件开销的情况下，多次重新编程 TIM2 的一部分寄存器也可以用于按周期读取数个寄存器。

## 14.6 功能描述

### 14.6.1 时钟

#### 14.6.1.1 时钟选择

计数器的时钟源有以下几种：

- 1) 内部时钟 (INT\_CK)。
- 2) 外部时钟模式：外部输入引脚 (TIx) 或者外部触发输入 (ETR)。
- 3) 内部触发输入 (编码器模式)。

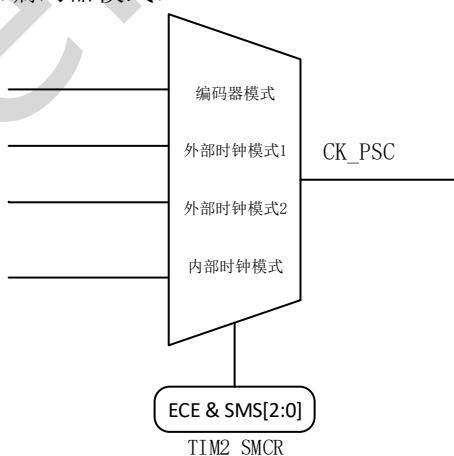


图 14-2 时钟选择

##### 14.6.1.1.1 内部时钟源 (INT \_CK)

当 SMS=000，关闭从模式时，计数器使能打开，预分频器的时钟直接由内部时钟驱动。此时计数器时钟为内部时钟分频后的时钟。

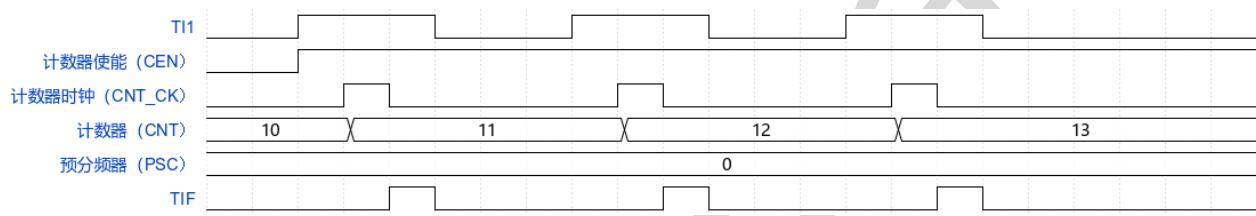
### 14.6.1.1.2 外部时钟模式 1 (外部输入引脚 TIx)

当 SMS = 111 时，选择外部时钟模式 1 (外部输入引脚 TIx)。计数器由选定的输入引脚的每个上升沿和下降沿驱动。

例：计数器在 T11 输入端的上升沿递增计数，具体配置如下：

- 1) 配置 TIM2\_CCMR1 寄存器 CC1S=01, IC1F[3:0]=xxxx, TIM2\_CCER 寄存器 CC1P=0, 选择通道 1 检测 TI1 输入的上升沿为有效沿，定义输入滤波器带宽。
- 2) 配置 TIM2\_SMCR 寄存器中的 TS=100, SMS=111, 选定 TI3 的边沿作为触发输入源，时钟选择外部时钟模式 1。
- 3) 配置 TIM2\_CR1 寄存器的 DIR=0, CEN=1, 选择递增计数模式，启动计数器。

当 TI1 出现有效边沿时，计数器递增计数一次且 TIF 标志位由硬件置 1。TI1 的有效边沿和计数器的实际时钟之间的延时取决于预 TI1 输入引脚同步电路设计。



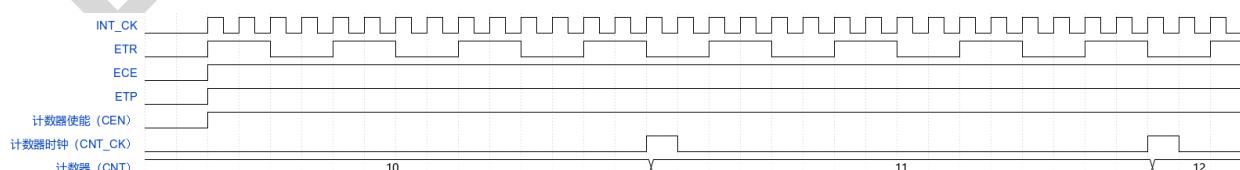
### 14.6.1.1.3 外部时钟模式 2 (外部触发输入 ETR)

当 TIM2\_SMCR 寄存器的 ECE=1 时，使能外部时钟模式 2，计数器由 ETRF 信号上的有效边沿驱动。

例：在 ETR 下每 4 个下降沿计数一次的递增计数器，具体配置如下：

- 1) 配置 TIM2\_SMCR 寄存器中的 ETF[3:0]=0010，每 4 个 ETR 信号的有效边沿驱动计数器计数一次；ETP=1，ETR 被反相，选择下降沿有效；ECE=1，选择外部时钟模式 2。
- 2) 配置 TIM2\_CR1 寄存器 DIR=0, CEN=1 选择计数方向为递增计数，启动计数器。

在 ETR 的下降沿和计数器实际时钟之间的延时取决于在 ETRP 信号端的重新同步电路。



## 14.6.1.2 时基单元

TIM2 的时基单元主要包括：计数器寄存器 (TIM2\_CNT)、预分频器寄存器 (TIM2\_PSC) 和自动装载寄存器 (TIM2\_ARR)。

计数器由一个 32 位的计数器和对应的自动装载寄存器组成，可以实现递增计数，递减计数，递增和递减计数的功能。计数器的时钟由预分频器提供，预分频器由预分频计数器和对应的寄存器组成，分频系数为 1-65536，可以随时写入，在下一次更新事件时生效。

自动预装载寄存器有预装载功能的 32 位影子寄存器，通过设置 TIM2\_CR1 寄存器的 APRE 位选择写入 ARR 寄存器的值立即生效或发生更新事件时载入影子寄存器。

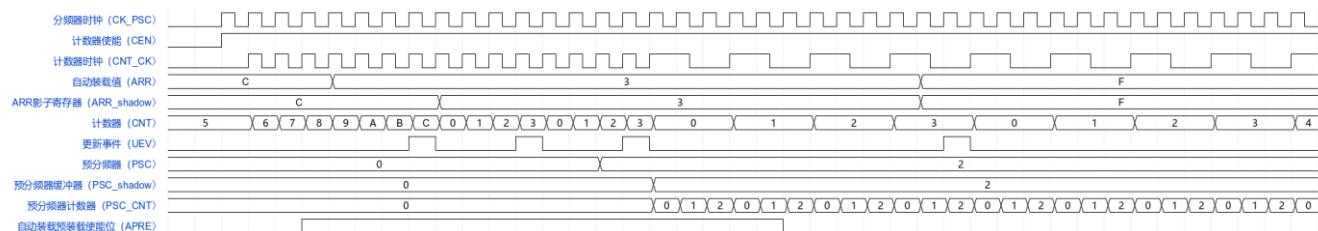


图 14-5 自动预装载

#### 14.6.1.3 计数模式

通过配置 TIM2\_CR1 寄存器的 DIR 位和 CMS 位可以选择计数器的计数模式，可以分为三种计数模式，递增计数模式、递减计数模式和中央对齐计数模式（递增/递减计数模式），下面对每个计数模式做详细介绍。

#### 14.6.1.3.1 递增计数模式

配置 TIM2 CR1 寄存器 CMS=0, DIR=0, 选择递增计数模式。

递增计数模式下，在使能 TIM2\_CR1 的 CEN 后计数器由 0 开始递增计数，直至 TIM2\_ARR 的值，并产生一个计数器上溢事件（更新事件），更新事件后计数器重新从 0 开始重新递增计数。设置 TIM2\_EGR 寄存器的 UG=1，同样可以产生一个更新事件。

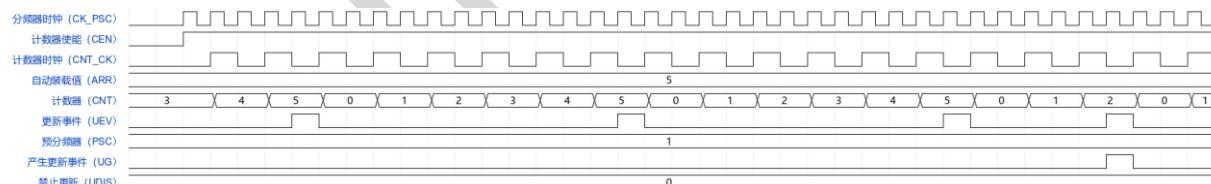


图 14-6 递增计数模式 (UDIS=0)

当配置 TIM2\_CR1 寄存器的 UDIS=1，禁止产生更新事件，当计数器发生上溢事件时，不产生更新事件；配置 UG=1，不产生更新事件，计数器和预分频器计数器会被初始化，从零开始递增计数。

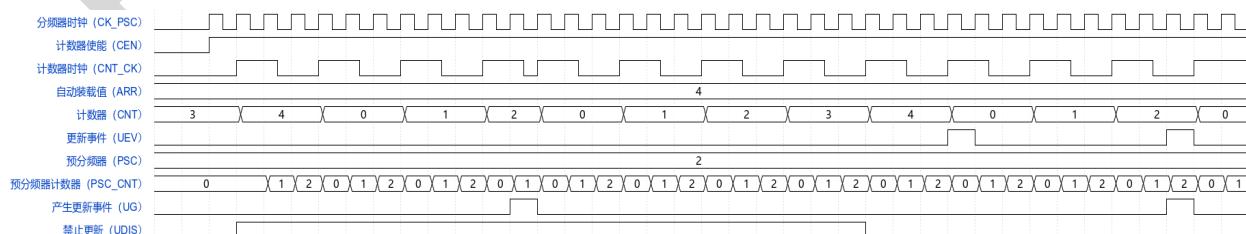


图 14-7 递增计数模式 (UDIS=0 禁止产生更新事件)

注：发生更新事件时：

- ARR 寄存器中的值被载入 ARR shadow 寄存器中。

- 预分频器的预装载值生效。

#### 14.6.1.3.2 递减计数模式

配置 TIM2\_CR1 寄存器的 CMS=0, DIR=1, 选择递减计数模式。

递减计数模式下, 此时计数器从自动装载值 TIM2\_ARR 开始递减计数, 计数到 0 时, 产生一个下溢事件(更新事件)。设置 TIM2\_EGR 寄存器的 UG=1, 同样可以产生一个更新事件, 更新事件后计数器从自动装载值 TIM2\_ARR 开始重新递减计数(TIM2\_CR1 寄存器 UDIS=0)。

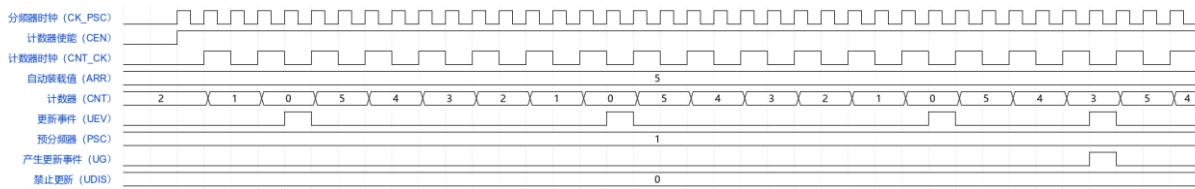


图 14-8 递减计数模式 (UDIS=0)

当配置 TIM2\_CR1 寄存器的 UDIS=1, 禁止产生更新事件, 当计数器发生下溢事件时, 不产生更新事件; 配置 UG=1, 同样不产生更新事件, 但是计数器和预分频器计数器会被初始化, 从 TIM2\_ARR 开始计数。

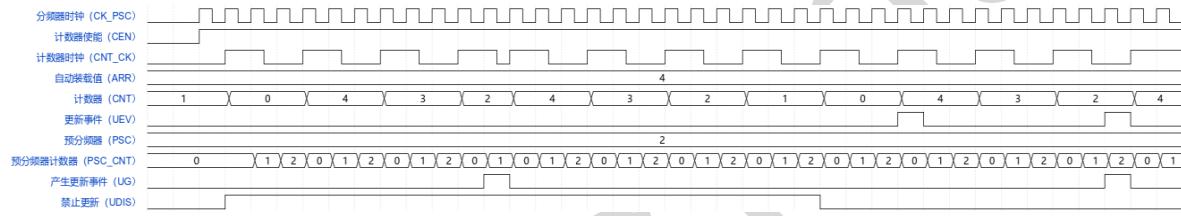


图 14-9 递减计数模式 (UDIS=1 禁止产生更新事件)

#### 14.6.1.3.3 中央计数模式 (递增/递减计数模式)

配置 TIM2\_CR1 寄存器的 CMS ≠ 0 (此时写入 DIR 无效), 选择中央对齐计数模式。

中央对齐计数模式, 递增计数和递减计数交替进行。递增计数到 ARR-1 时, 产生一个上溢事件, 然后从 ARR 先开始递减计数到 1, 产生一个下溢事件, 再从 0 开始递增计数。

设置 TIM2\_EGR 寄存器的 UG=1, 同样可以产生一个更新事件, 更新事件后计数器从 0 开始重新递增计数 (TIM2\_CR1 寄存器 UDIS=0)。

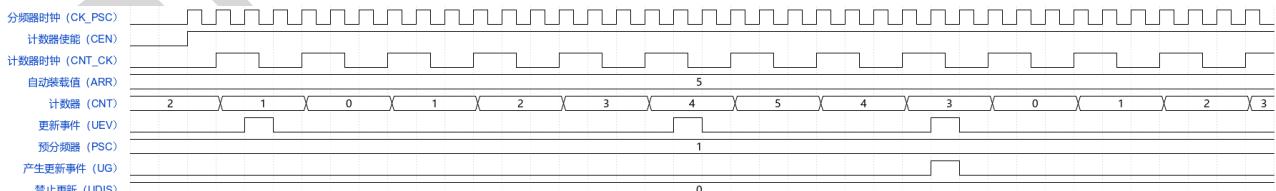


图 14-10 中央计数模式 (UDIS=0)

当配置 TIM2\_CR1 寄存器的 UDIS=1, 禁止产生更新事件, 当计数器发生上溢或下溢事件时, 不产生更新事件; 配置 UG=1, 同样不产生更新事件, 但是计数器和预分频器计数器会被初始化, 从零开始重新计数。

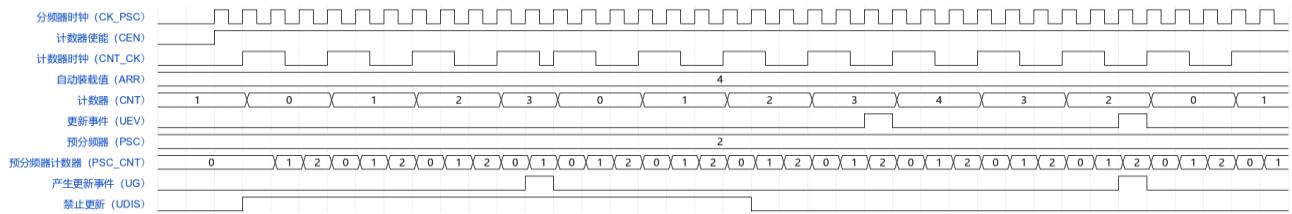


图 14-11 中央计数模式 (UDIS=1 禁止产生更新事件)

#### 14.6.1.4 定时器同步

不同的 TIMx 定时器在内部连接，通过配置一个定时器工作为主模式，一个定时器工作在从模式（配置 TIMx\_SMCR 寄存器 SMS=100/101/110，从模式选择复位模式、门控模式或触发模式），可以实现定时器之间的级联或同步，实现对处于从模式定时器的计数器进行复位、启动、通知或提供时钟等操作。

##### 14.6.1.4.1 从模式

###### 复位模式

配置 TIM2\_SMCR 寄存器 SMS=100，从模式选择复位模式。此模式下，发生触发输入事件会使计数器清零重启。

例如，TI2 输入端的下降沿触发计数器重启：

- 1) 配置 CC2S=01，CC2 通道被配置为输入模式，IC2 映射在 TI2 上，配置 TIM2\_CCER 寄存器 CC2P=1，检测下降沿。
- 2) 配置 TIM2\_SMCR 寄存器 SMS = 100，从模式选择为复位模式；置 TIM2\_SMCR 寄存器中 TS = 110，ETF=0000，选择滤波后的定时器输入 2 (TI2FP2) 作为同步计数器的触发输入。
- 3) 配置 TIM2\_CR1 寄存器 ARR, PSC=0, DIR=0，配置预分频系数并选择计数方向，配置 CEN=1，使能计数器。

计数器的时钟源由内部时钟提供，当检测到 TI2 的下降沿，计数器被清零重启。此时触发器中断标记被硬件置 1。

下图为复位模式下 TIM2\_A RR = 0x13 时的时序图。

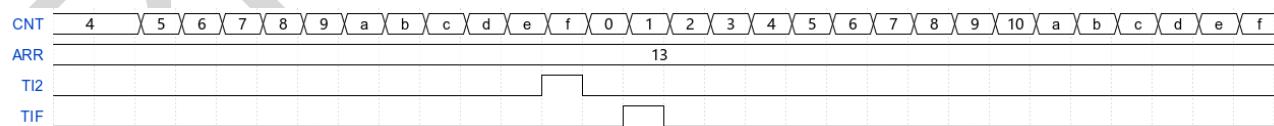


图 14-12 复位模式的控制电路图

###### 门控模式

配置 TIM2\_SMCR 寄存器 SMS=101，从模式选择门控模式。此模式下，触发输入为高时，计数器始终开启，否则计数器停止（但不发生复位操作），计数器的开启和停止可控。

例如，计数器只在 TI1 为高时计数：

- 1) 配置 CC1S=01，CC1 通道被配置为输入模式，IC1 映射在 TI1 上，配置 TIM2\_CCER 寄存器 CC1P=1，检测下降沿。

- 2) 配置 TIM2\_SMCR 寄存器 SMS=101, 从模式选择为门控模式, 配置 TS=101, ETF=0000, 选择滤波后的定时器输入 1 (TI1FP1) 作为同步计数器的触发输入。
- 3) 配置 TIM2\_CR1 寄存器 ARR, PSC=0, DIR=0, 配置预分频系数并选择计数方向, 配置 CEN=1, 使能计数器。

计数器的时钟源由内部时钟提供, 当检测到 TI1 的低电平, 计数器开始计数, 当 TI1 为高电平时, 计数器停止。计数器开启或停止都会将 TIF 置 1。

下图为门控模式下 TIM2\_ARR=0xf 的时序图。

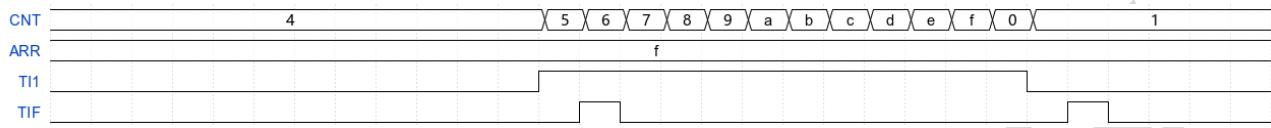


图 14-13 门控模式下的控制电路

#### 触发模式

配置 TIM2\_SMCR 寄存器 SMS=111, 从模式选择触发模式。计数器在触发输入的上升沿启动, 计数器的启动可控, 停止不可控。

例如, 计数器在 TI1 输入的上升沿开始计数:

- 1) 配置 CC1S=01, CC1 通道被配置为输入模式, IC1 映射在 TI1 上, 配置 TIM2\_CCER 寄存器 CC1P=0, 检测上升沿。
- 2) 配置 TIM2\_SMCR 寄存器 SMS = 110, 从模式选择为触发模式; 配置 TS=101, ETF=0000, 选择滤波后的定时器输入 1 (TI1FP1) 作为同步计数器的触发输入。
- 3) 配置 TIM2\_CR1 寄存器 ARR, PSC=0, DIR=0, 配置预分频系数并选择计数方向, 配置 CEN=1, 使能计数器。

计数器的时钟源由内部时钟提供, 当检测到 TI1 的上升沿, 计数器开始计数。

下图为触发模式下 TIM2\_ARR=0xf 的时序图。



图 14-14 触发器模式下的控制电路

#### 外部时钟模式 2+触发模式

当时钟源选择外部时钟模式 2, ETR 信号被用作外部时钟的输入时, 可以与另一种从模式 (外部时钟模式 1 和编码器模式除外) 一起使用。在复位模式、门控模式或触发模式可以选择另一个输入作为触发输入 (TRGI)。

例如, 计数器在 ETR 的每一个上升沿计数一次:

- 1) 配置 TIM2\_SMCR 寄存器 ETF = 0000, ETPS = 00, ETP = 0 检测 ETR 的上升沿, 配置 ECE = 1 选择外部时钟模式 2。
- 2) 配置 TIM2\_CCMR1 寄存器中 CC1S=01, 选择输入捕获源, TIM2\_CCER 寄存器中 CC1P=0, 选择上升沿有效。
- 3) 配置 TIM2\_SMCR 寄存器中 SMS = 110, 从模式选择为触发模式。配置 TIM2\_SMCR 寄

存器中  $TS = 101$ , 选择 TI1 作为输入源。

- 4) 配置 TIM2\_CR1 寄存器 ARR, PSC=0, DIR=0, 配置预分频系数并选择计数方向, 配置 CEN=1, 使能计数器。

计数器在 ETR 的上升沿开始计数, 当检测到 TI1 上的上升沿时, TIF 被硬件置 1。ETR 信号的上升沿和计数器实际复位间的延时取决于 ETRP 输入端的重同步电路。

下图为外部时钟模式 2+触发模式下 TIM2\_ARR=13 时的时序图。

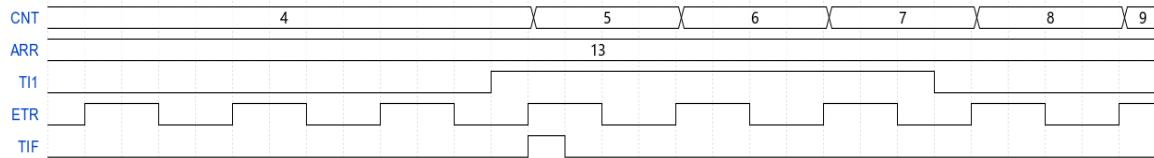


图 14-15 外部时钟模式 2+触发模式下的控制电路

#### 14.6.1.4.2 定时器间的同步

详见 TIM1 相关描述。

### 14.6.2 输入捕获

#### 14.6.2.1 输入捕获

输入捕获部分包括数字滤波器、多路复用、预分频器等, 其结构如下图所示:

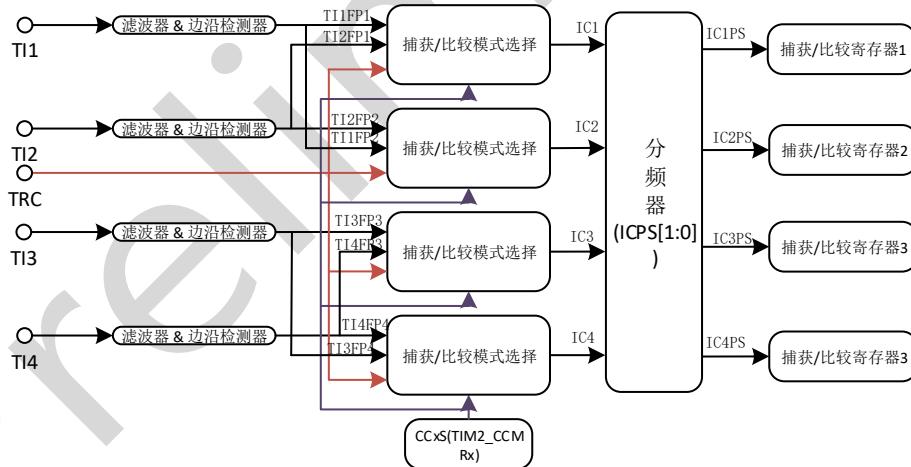


图 14-16 TIM2 输入捕获结构图

通过配置 TIM2\_CCMRx 寄存器 ICxF, 可以配置数字滤波器的滤波宽度 (滤波器的采样频率及数字滤波器长度如下表所示), 当数字滤波器的输入信号带宽大于滤波宽度时, 输入信号有效; 数字滤波器对输入引脚 TIx 输入信号的采样后, 产生一个滤波后的信号 TIxF, 然后通过边沿检测器和软件配置选择 TIxF 信号的有效边沿, 产生一个有效信号 TIxFPx, 这个信号可以用作捕获控制信号或作为从模式控制器的触发输入信号, 然后经过预分频产生一个信号 ICxPS, 进入捕获/比较寄存器。

表 14-1 数字滤波器长度与 ICxF 的对应关系表

IC1F[3:0]	采样频率和滤波长度	IC1F[3:0]	采样频率和滤波长度
-----------	-----------	-----------	-----------

0000	无滤波器, 以 $f_{DTS}$ 采样	1000	采样频率 $f_{sampling} = f_{DTS} / 8, N=6$
0001	采样频率 $f_{sampling} = f_{INT\_CK}, N=2$	1001	采样频率 $f_{sampling} = f_{DTS} / 8, N=8$
0010	采样频率 $f_{sampling} = f_{INT\_CK}, N=4$	1010	采样频率 $f_{sampling} = f_{DTS} / 16, N=5$
0011	采样频率 $f_{sampling} = f_{INT\_CK}, N=8$	1011	采样频率 $f_{sampling} = f_{DTS} / 16, N=6$
0100	采样频率 $f_{sampling} = f_{DTS}/2, N=6$	1100	采样频率 $f_{sampling} = f_{DTS} / 16, N=8$
0101	采样频率 $f_{sampling} = f_{DTS} / 2, N=8$	1101	采样频率 $f_{sampling} = f_{DTS} / 32, N=5$
0110	采样频率 $f_{sampling} = f_{DTS} / 4, N=6$	1110	采样频率 $f_{sampling} = f_{DTS} / 32, N=6$
0111	采样频率 $f_{sampling} = f_{DTS} / 4, N=8$	1111	采样频率 $f_{sampling} = f_{DTS} / 32, N=8$

在输入捕获模式下, 输入捕获发生在影子寄存器上, 然后影子寄存器的内容载入到捕获/比较寄存器中。

输入捕获模式下, 当检测到信号 ICx 上的有效边沿后, 计数器的当前值被锁存到对应的影子寄存器(TIM2\_CCRx\_shadow)上, 再复制到对应的捕获比较寄存器(TIM2\_CCRx)中。当开启了中断或 DMA 使能, 发生捕获事件时, 将产生相应的中断或 DMA 请求。发生捕获事件时, 会将状态寄存器 (TIM2\_SR) 中的捕获标志位 CCxIF 置 1, 通过配置 CCxIF=0 或读取 TIM2\_CCRx 中的数据, 清除 CCxIF 标志位。当 CCxIF 未被清零时, 发生输入捕获事件, 重复捕获标志位将会被置 1, 通过配置 CCxOF=0, 可以清除 CCxOF 标志位。

例如, 通过采样 TI1 输入信号的有效沿, 在 TI1 的上升沿来到时捕获当前计数器的值, 锁存到 TIM2\_CCR1 寄存器中, 步骤如下:

- 1) 配置 TIM2\_CCMR1 寄存器 CC1S=01, CC1 通道被配置为输入, IC1 映射在 TI1 上。
- 2) 配置 TIM2\_CCMRx 寄存器 IC1F[3:0], 配置数字滤波器的滤波宽度 (按需配置)。
- 3) 配置 TIM2\_CCER 寄存器 CC1P=0, 选择捕获发生在 TI1 信号的上升沿。
- 4) 配置 TIM2\_CCMR1 寄存器的 IC1PS , 选择预分频系数。
- 5) 配置 TIM2\_CCER 寄存器的 CC1E = 1, 开启输入/捕获通道 1 的捕获使能。
- 6) 配置 TIM2\_DIER 寄存器 CC1IE=1, CC1DE=1, 开始通道 1 的捕获/比较通道 1 中断使能和允许捕获/比较通道 1 的 DMA 请求。

注:

- 当通道配置为输入模式时, TIM2\_CCRx 寄存器属性变为只读。
- 为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息, 建议在读出捕获溢出标志之前读取数据。
- 设置 TIM2\_EGR 寄存器中相应的 CCxG 位, 可以通过软件产生输入捕获中断或 DMA 请求。

## 14.6.2.2 PWM 捕获

PWM 输入模式的操作配置与一般输入捕获有以下不同点:

- 1) 两个边沿有效且极性相反的 ICx 信号被映射至同一个 TIx 输入。

2) 配置从模式控制器为复位模式，将其中一路 TIxFP 作为触发输入信号。

例：测量 TI1 的 PWM 信号的长度（TIM2\_CCR1 寄存器）和占空比（TIM2\_CCR2 寄存器），具体步骤如下（取决于 INT\_CK 的频率和预分频器的值）。

- 1) 配置 TIM2\_CR1 寄存器 DIR=0，选择计数器计数模式为递增计数模式。
- 2) 配置 TIM2\_CCMR1 寄存器的 CC1S = 01，将 IC1 映射在 TI1 上，选择 TIM2\_CCR1 的有效输入。
- 3) 配置 CC1P = 0，选择 TI1FP1 的有效极性（上升沿有效）（将计数器的值捕获到 TIM2\_CCR1 中并清除计数器）。
- 4) 配置 TIM2\_CCMR1 寄存器的 CC2S = 10，将 IC2 映射在 TI1 上，选择 TIM2\_CCR2 的有效输入。
- 5) 配置 CC2P = 0，选择 TI2FP2 的有效极性（下降沿有效）（将计数器的值捕获到 TIM2\_CCR2 中）。
- 6) 配置 TIM2\_SMCR 寄存器中的 TS = 101，选择 TI1FP1 为有效的触发输入信号。
- 7) 配置 TIM2\_SMCR 中的 SMS = 100，从模式控制器设置为复位模式。
- 8) 配置 TIM2\_CCER 寄存器中 CC1E=1 且 CC2E = 1。开启 CC1 通道和 CC2 通道的捕获使能。

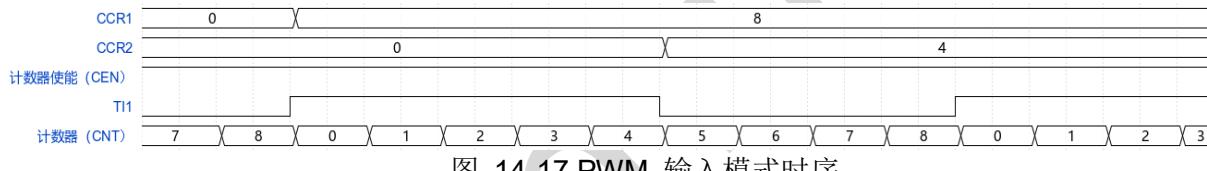


图 14-17 PWM 输入模式时序

注：由于从模式控制器只连接了 TI1FP1 和 TI2FP2，所以 PWM 输入模式只适用于 TIM2\_CH1/TIM2\_CH2 信号。

### 14.6.2.3 编码器接口

编码器接口模式就是计数器在 TI1 和 TI2 正交信号相互作用下计数，在输入源改变期间，计数方向被硬件自动修改。通过配置 TIM2\_SMCR 寄存器 SMS 位可以选择输入源，根据输入源的不同，可以将编码器接口模式分为 3 种模式，SMS=001，编码器接口模式 1；SMS=010，编码器接口模式 2；SMS=011，编码器接口模式 3；三种模式具体计数操作如下表所示。两个输入 TI1 和 TI2 被用来作为正交编码器的接口。

编码器模式下，计数器开启之前必须先配置好 ARR 寄存器，因为使用编码器接口模式相当于使用了一个带有方向选择的外部时钟。计数器在 0 到 TIM2\_ARR 寄存器的自动装载值之间连续计数（递增计数和递减计数由外部时钟控制）。

注：编码器模式不支持外部时钟模式 2。

编码器接口模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指示着编码器的位置。计数方向与相连的传感器旋转的方向对应。下表列出了所有可能的组合，假设 TI1 和 TI2 不同时变换。

表 14-2 计数方向与编码器信号的关系

计数模式	相对电平 (TI1FP1 相对于 TI2, TI2FP2 相对于 TI1)	TI1FP1 信号		TI2FP2 信号	
		上升	下降	上升	下降
编码器接口模式 1 (只在 TI1 计数)	高电平	递减计数	递增计数	-	-
	低电平	递增计数	递减计数	-	-
编码器接口模式 2 (只在 TI2 计数)	高电平	-	-	递增计数	递减计数
	低电平	-	-	递减计数	递增计数
编码器接口模式 3 (在 TI1 和 TI2 计数)	高电平	递减计数	递增计数	递增计数	递减计数
	低电平	递增计数	递减计数	递减计数	递增计数

下例是计数器在编码器接口模式下的配置和时序图，从图中可以看出计数信号的产生和方向控制。具体配置如下：

- 1) 配置 TIM2\_CCMR 寄存器 CC1S=01，将 IC1FP1 映射到 TI1 上。
- 2) 配置 TIM2\_CCMR 寄存器 CC2S =01，将 IC2FP2 映射到 TI2 上。
- 3) 配置 TIM2\_CCER 寄存器 CC1P =0，IC1 不反相，此时 IC1=TI1。
- 4) 配置 TIM2\_CCER 寄存器 CC2P =0，IC2 不反相，此时 IC1=TI2。
- 5) 配置 TIM2\_SMCR 寄存器 SMS =011，选择编码器模式 3，根据另一个信号的输入电平，计数器在 TI1FP1 和 TI2FP2 的边沿计数。
- 6) 配置 TIM2\_CR1 寄存器 CEN =1，开启计数器。

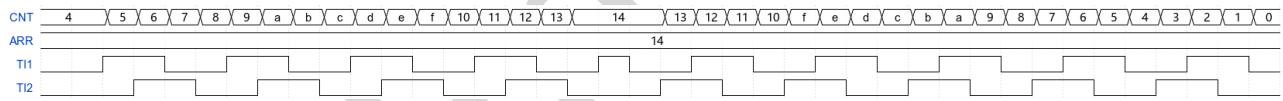


图 14-18 编码器模式下的计数器时序图

下图为当 IC1FP1 极性反相时计数器的时序图 (CC1P = 1, 其他配置不变)

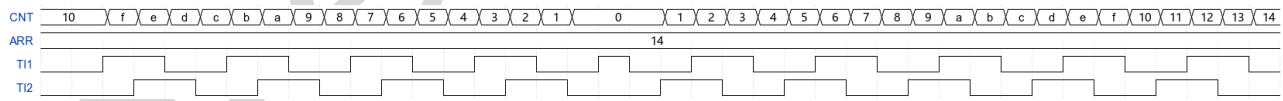


图 14-19 IC1FP1 反相的编码器接口模式时序图

编码器接口模式下，计数器可以提供传感器当前位置的信息。通过使用另一个配置在捕获模式的定时器测量两个编码器事件的间隔周期来获得动态的信息（速度，加速度，减速度）。根据两个编码器事件的间隔周期，可以定期读取计数器。可以通过把计数器的值锁存到第三个输入捕获寄存器（捕获信号必须是周期性的并且可以由另一个定时器产生）来实现。还可以通过 DMA 请求来读取它的值。

#### 14.6.2.4 定时器异或

配置 TIM2\_CR2 寄存器的 TI1S =1，将 TIM2\_CH1、TIM2\_CH2 和 TIM2\_CH3 引脚经异或后连接到 TI1 的输入端，用于定时器的所有输入模式。

例：TIM2\_CH1、TIM2\_CH2 和 TIM2\_CH3 引脚经异或后连接到 TI1 的输入端，采样 TI1 输入信号

的有效沿，在 TI1 的上升沿来到时捕获当前计数器的值，锁存到 TIM2\_CCR1 寄存器中，具体配置如下：

- 1) 配置 TIM2\_CR2 寄存器 TI1S=1，配置定时器的三个输入经异或后连接到 TI1 输入通道。
- 2) 配置 TIM2\_CCMR1 寄存器 CC1S=01,CC1 通道被配置为输入，IC1 映射在 TI1 上。
- 3) 配置 TIM2\_CCMRx 寄存器 IC1F[3:0],配置数字滤波器的滤波宽度（按需配置）。
- 4) 配置 TIM2\_CCER 寄存器 CC1P=0，选择捕获发生在 TI1 信号的上升沿。
- 5) 配置 TIM2\_CCMR1 寄存器的 IC1PS，选择预分频系数。
- 6) 配置 TIM2\_CCER 寄存器的 CC1E = 1，开启输入/捕获通道 1 的捕获使能。
- 7) 配置 TIM2\_CR1 寄存器 CEN=1，启动计数器。

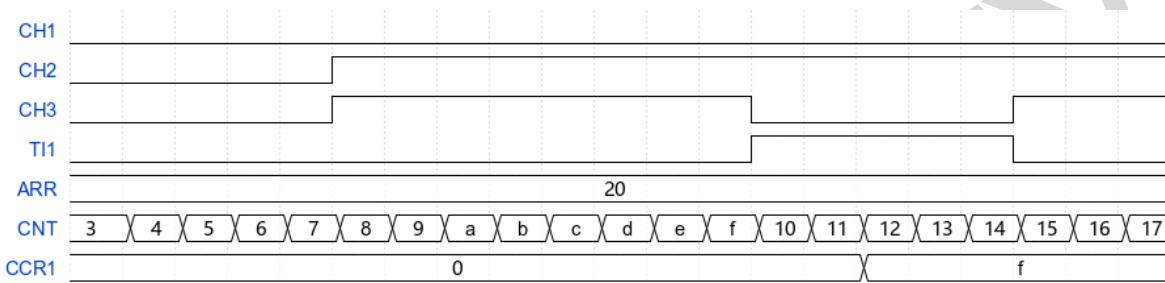


图 14-20 (TI1 异或输入) 输入捕获波形图

#### 14.6.2.4.1 霍尔接口电路

霍尔传感器接口模式是异或功能的一个应用实例，可以用来驱动电机，在使用 TIM1 产生 PWM 信号驱动电机时，可以将另一个计数器 TIM2/TIM3 作为“接口定时器”来连接霍尔传感器，配置 TIM1\_CR2 寄存器 TI1S=1，将定时器的 3 个输入脚 (CH1、CH2、CH3) 经异或后连接到 TI1 输入通道，“接口定时器”接收这个信号。三个霍尔传感器与“接口定时器”的三路输入捕获引脚对应连接，每个传感器输入一路波形到输入引脚。分析输入捕获信号可以计算电机速度的信息。

“接口定时器”在输出模式下可以产生一个用来控制 TIM1 PWM 输出的脉冲，用来驱动电机。所以“接口定时器”在输出比较或 PWM 模式延时一段时间后产生一个正脉冲，然后通过 TRGO 输出被传送到 TIM1。

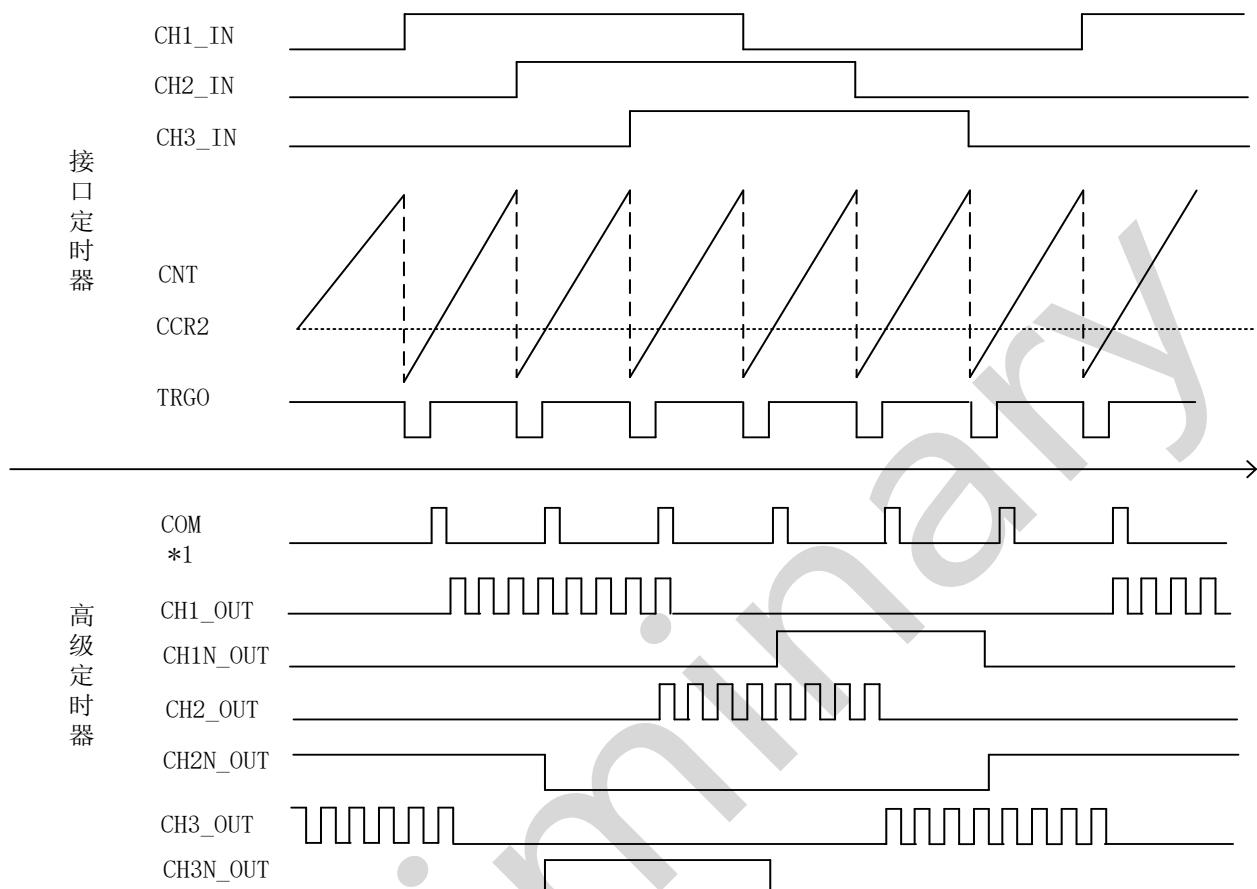
例：霍尔输入连接到 TIM2 定时器，每次任一霍尔输入上信号变化都会改变 TIM1 的 PWM 配置。

- 1) 配置 TIM2\_CR2 寄存器 TI1S=1，配置三个定时器输入经异或后连接到 TI1 输入通道。
- 2) 配置 TIM2\_ARR 为其最大值（计数器必须通过 TI1 的变化清零）。配置 PSC，设置计数周期大于传感器上的两次变化的时间。
- 3) 配置 TIM2\_CCMR1 寄存器设 CC1S=01，配置通道 1 为捕获模式（选中 TRC）。
- 4) 配置 TIM2\_CCMR1 寄存器 CC2S=00, OC2M=111，配置通道 2 为 PWM2 模式，并具有要求的延时。
- 5) 配置 TIM2\_CR2 寄存器 MMS=101，选择 OC2REF 作为 TRGO 上的触发输出。

TIM1 中，正确的 ITR 输入必须是触发器输入，定时器被编程为产生 PWM 信号，捕获/比较控制信号为预装载的 (TIM1\_CR2 寄存器中 CCPC = 1)，同时触发输入控制 COM 事件 (TIM1\_CR2 寄存器中 CCUS = 1)。在一次 COM 事件后，写入下一步的 PWM 控制位 (CCxE、OCxM)，这可以在处理

OC2REF 上升沿的中断子程序里实现。

下图显示了这个实例：



\*1: com事件发生，通过中断子程序设定新的CCxE、CCxNE和OCxM来配置下一步动作。

图 14-21 霍尔传感器接口的实例

### 14.6.3 比较/输出

捕获比较通道的比较输出部分由比较器、输出控制电路和捕获 / 比较寄存器组成，其结构图如下图所示：

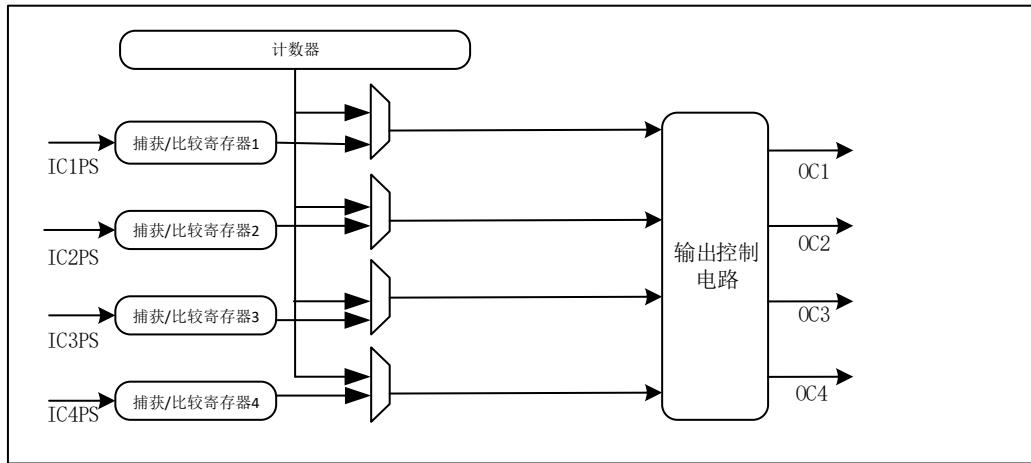


图 14-22 捕获 / 比较的输出部分

在输出比较模式下，捕获比较寄存器的内容被载入到影子寄存器中，然后影子寄存器的内容和计数器当前值进行比较。捕获/比较模块包括一个捕获/比较寄存器（预装载寄存器）和一个影子寄存器，读写过程仅操作捕获/比较寄存器。

#### 14.6.3.1 强制输出

配置  $\text{TIM2\_CCMRx}$  寄存器中  $\text{CCxS} = 00$ ，将通道  $\text{CCx}$  设置为输出模式，通过配置  $\text{TIM2\_CCMRx}$  寄存器  $\text{OCxM}$  位，可以直接将输出比较信号直接强制为有效或无效状态，不依赖于输出比较结果。配置  $\text{TIM2\_CCMRx}$  寄存器  $\text{OCxM} = 100$ ，强置输出比较信号为无效状态。此时  $\text{OCxREF}$  被强置为低电平。配置  $\text{TIM2\_CCMRx}$  寄存器  $\text{OCxM} = 101$ ，强置输出比较信号为有效状态。此时  $\text{OCxREF}$  被强置为高电平（ $\text{OCxREF}$  始终为高电平有效）。

注：强制输出模式下，在  $\text{TIM2\_CCRx}$  影子寄存器和计数器之间的输出比较仍在进行，比较结果的相应标志位也会被修改，如果开启了对应的中断和 DMA 请求，仍会产生对应的中断和 DMA 请求。

#### 14.6.3.2 输出比较

输出比较模式下，当计数器与捕获比较寄存器值相同时，可以表示计时结束，也可以根据  $\text{TIM2\_CCMRx}$  寄存器的  $\text{OCxM}$  位的配置用来输出不同的波形。

例如，当计数器与捕获/比较寄存器的内容匹配时，输出比较模式下的操作如下：

- 1) 在比较匹配时， $\text{OCxM}$  的值不同，输出通道  $x$  信号  $\text{OCx}$  的操作不同：
  - a)  $\text{OCxM} = 000$ :  $\text{OCx}$  信号的保持它的电平。
  - b)  $\text{OCxM} = 001$ :  $\text{OCx}$  信号被设置成有效电平。
  - c)  $\text{OCxM} = 010$ :  $\text{OCx}$  信号被设置成无效电平。
  - d)  $\text{OCxM} = 011$ :  $\text{OCx}$  信号进行翻转。
- 2) 匹配时设置中断状态寄存器中的标志位（ $\text{TIM2\_SR}$  寄存器中的  $\text{CCxIF}$  位）。
- 3) 当配置了  $\text{TIM2\_DIER}$  寄存器中的  $\text{CCxE} = 1$ ，匹配时则产生一个中断。
- 4) 当配置了  $\text{TIM2\_DIER}$  寄存器中的  $\text{CCxDE} = 1$ ，匹配时则产生一个 DMA 请求。

输出比较模式也可以用来输出一个单脉冲（单脉冲输出模式）。

通道 1 的输出比较模式的配置步骤如下：

- 1) 配置计数器的时钟（选择时钟源，配置预分频系数）。
- 2) 配置 TIM2\_ARR 和 TIM2\_CCR1 寄存器。
- 3) 配置 TIM2\_DIER 寄存器的 CC1IE =1，使能捕获/比较 1 中断。
- 4) 配置输出模式：
  - a) 配置 OC1M = 011，OC1 比较匹配时翻转。
  - b) 配置 OC1PE = 0，禁止 TIM2\_CCRx 寄存器的预装载功能。
  - c) 配置 CC1P = 1，OC1 低电平有效。
  - d) 配置 CC1E = 1，开启输出/比较 1 输出使能，OC1 信号输出到对应的输出引脚。

配置 TIM2\_CR1 寄存器的 CEN =1，启动计数器。

当配置 TIM2\_CCMRx 寄存器中 OCxPE=0，禁止 TIM2\_CCRx 寄存器的预装载功能时，可以随时写入 TIM2\_CCRx 寄存器，并且写入的值立即生效。当配置 TIM2\_CCMRx 寄存器中 OCxPE=1，启用 TIM2\_CCRx 寄存器的预装载功能时，读写仅对预装载寄存器进行操作，TIM2\_CCR1 预装载寄存器的值在下次更新事件到来时生效。下图给出了一个例子。

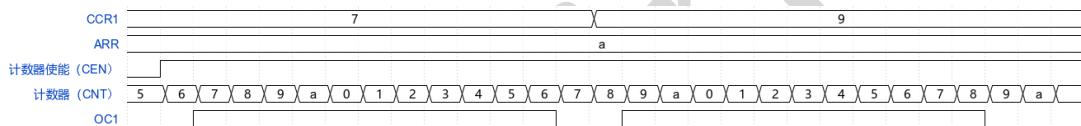


图 14-23 输出比较模式，OC1 信号在匹配时翻转

注：

- 输出比较模式下，更新事件不会对输出结果产生影响。
- 强制输出模式下，在 TIM2\_CCRx 影子寄存器和计数器之间的输出比较仍在进行，比较结果的相应标志位也会被修改，如果开启了 对应的中断和 DMA 请求，仍会产生对应的中断和 DMA 请求。

### 14.6.3.3 PWM 输出

在 PWM 模式下，根据 TIM2\_ARR 寄存器和 TIM2\_CCRx 寄存器的值，产生一个频率、占空比可控的 PWM 波形。

配置与通道 x 对应的 TIM2\_CCMRx 寄存器的 OCxM=110 或 OCxM=111，选择通道 x 进入 PWM 模式 1 或 PWM 模式 2。PWM 模式下，计数器和 CCR 会一直进行比较，根据配置和比较结果，通道 x 输出不同的信号，因此 TIM2 可以产生 4 个同频率下独立占空比的 PWM 输出信号。PWM 模式下必须开启 TIM2\_CCRx 的预装载功能和 TIM2\_ARR 寄存器的预装载功能。写入 TIM2\_CCR1 预装载寄存器和 TIM2\_ARR 预装载寄存器的值在发生下个更新事件时，才会生效，载入相应的影子寄存器。所以 PWM 模式下，使能计数器前要先初始化所有的寄存器，也可以设置 UG=1，产生更新事件。

配置 TIM2\_CCER 寄存器的 CCxP 选择 OCx 的有效极性。配置 TIM2\_CCER 寄存器的 CCxE、CCxNE 位和 TIM2\_BDTR 寄存器的 MOE、OSSI、OSSR 位控制 OCx 的输出使能。配置 TIM2\_CR1 寄存器的 CMS 位，可以选择产生边沿对齐或中央对齐的 PWM 信号。

- 1) CMS=00, 边沿对齐模式, 再进一步配置 DIR, 选择计数模式(递增或递减计数模式)。
- 2) CMS=01, 中央对齐模式1。
- 3) CMS=10, 中央对其模式2。
- 4) CMS=11, 中央对齐模式3。

#### 14.6.3.3.1 PWM 边沿对齐模式——递增计数模式

在递增计数模式配置的基础上, 配置 TIM2\_CCMRx 寄存器 CCxS=00, 选择输出模式, OCxM=110, 选择 PWM 模式1, 当 TIM2\_CNT < TIM2\_CCRx 时通道 x (OCxREF) 为有效电平, 否则为无效电平。如果 TIM2\_CCRx 中的比较值大于自动重装载值(TIM2\_ARR), 则 OCxREF 保持为有效电平。如果比较值为 0, 则 OCxREF 保持为无效电平。图 17-24 为 CCR1=1, CCR2=2, CCR3=3, CCR4=b, ARR=a 时边沿对齐递增计数时 PWM 模式1 的波形实例。

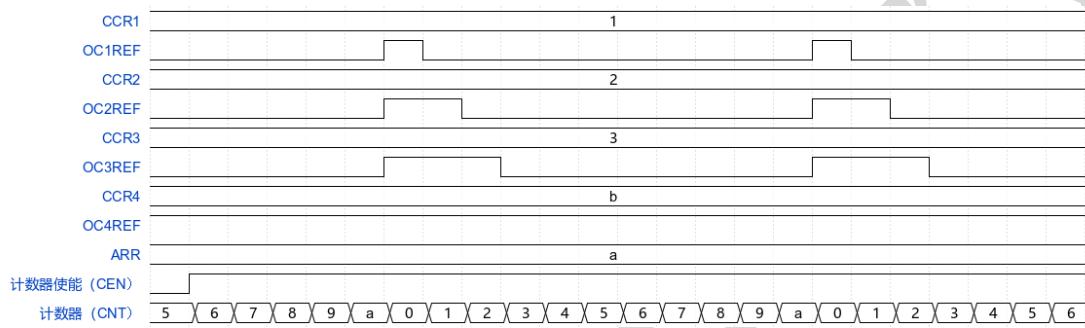


图 14-24 边沿对齐递增计数时 PWM 模式 1 的波形

#### 14.6.3.3.2 PWM 边沿对齐模式——递减计数模式

在递减计数模式配置的基础上, 配置 TIM2\_CCMRx 寄存器 CCxS=00, 选择输出模式, OCxM=110, 选择 PWM 模式1, 当 TIM2\_CNT > TIM2\_CCRx 时通道 x (OCxREF) 为无效电平, 否则有效电平。图 16-25 为 CCR1=6, CCR2=4, CCR3=9, CCR4=b, ARR=a 时边沿对齐递减计数时 PWM 模式1 的波形实例。

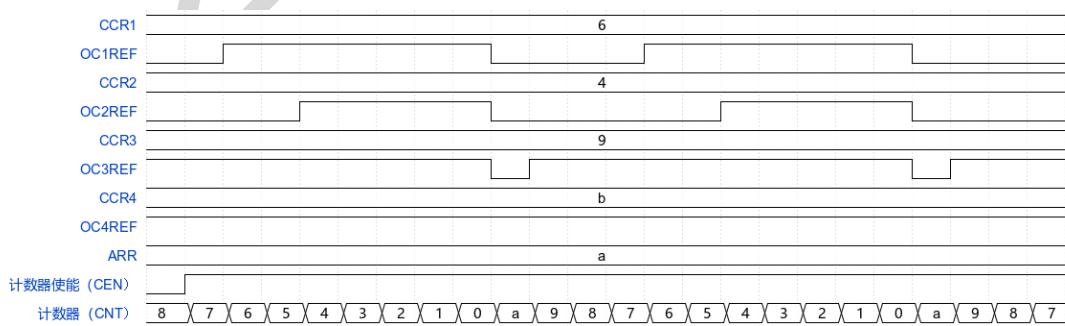


图 14-25 边沿对齐递减计数时 PWM 模式 1 的波形

#### 14.6.3.3.3 PWM 中央对齐模式

首先配置 TIM2 计数器为中央对齐计数模式, 配置 TIM2\_CCMRx 寄存器 CCxS=00, 选择输出模式, 根据配置不同的 CMS, 输出比较中断标志位在计数器递增计数时被设置 (CMS=01)、在计数器递减计数时被设置 (CMS=10)、或在计数器递增或递减计数时被设置 (CMS=11)。图 17-26 为

$\text{CCR1}=6$ ,  $\text{CCR2}=4$ ,  $\text{CCR3}=9$ ,  $\text{CCR4}=b$ ,  $\text{ARR}=a$  时中央对齐 PWM 模式 1 (CMS=11) 的波形实例。

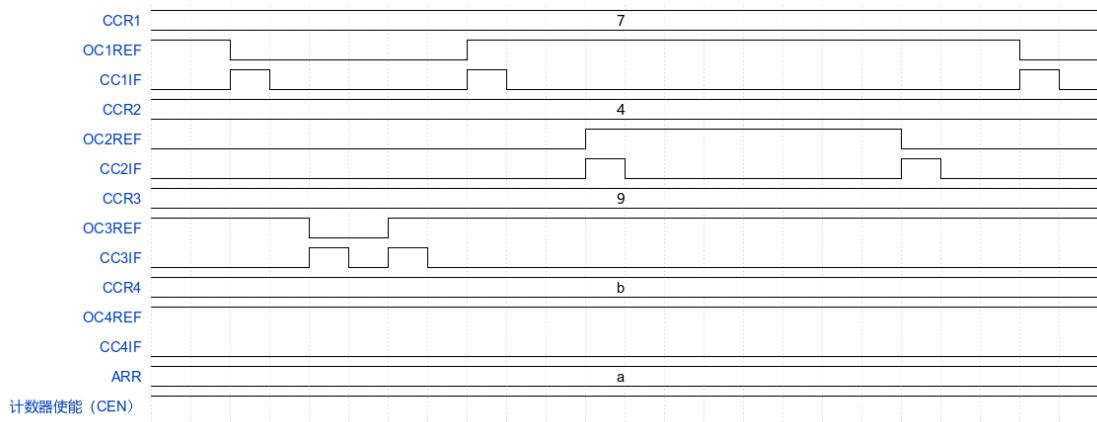


图 14-26 中央对齐 PWM 模式 1 的波形 (CMS=11)

#### 14.6.3.4 单脉冲输出

单脉冲模式 (OPM) 是计数器只响应一个激励，延时后产生一个脉宽可调的脉冲。配置 TIM2\_CR1 寄存器的 OPM=1，选择单脉冲模式，触发信号有效沿或配置 CEN=1 都可以启动计数器，CEN=1 一直保持到下个更新事件发生或配置 CEN=0。

产生脉冲的必要条件是比较值与计数器的初始值不同。所以在计数器启动之前的必要配置如下：

- 1) 递增计数方式：计数器  $\text{CNT} < \text{CCRx} \leq \text{ARR}$ 。
- 2) 递减计数方式：计数器  $\text{CNT} > \text{CCRx}$ 。

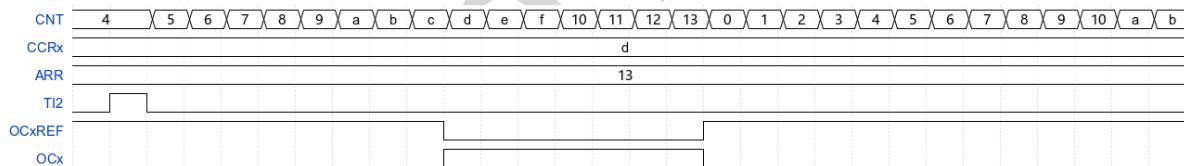


图 14-27 单脉冲模式的例子

例如，在 TI2 检测到上升沿，延迟  $t_{\text{DELAY}}$  之后，在 OC2 上产生一个长度为  $t_{\text{PULSE}}$  的正脉冲。

假定 TI2FP2 作为触发 1：

- 1) 配置 TIM2\_CCMR1 寄存器中的  $\text{CC2S} = 01$ ，将 TI2FP2 映射到 TI2。
- 2) 配置 TIM2\_CCER 寄存器中的  $\text{CC2P} = 0$ ，检测 TI2FP2 的上升沿。
- 3) 配置 TIM2\_SMCR 寄存器中的  $\text{TS} = 110$ ，TI2FP2 作为从模式控制器的触发 (TRGI)。
- 4) 配置 TIM2\_SMCR 寄存器中的  $\text{SMS} = 110$ ，选择触发模式，TI2FP2 使能计数器工作。

OPM 的波形由 TIM2\_ARR 和 TIM2\_CCR1 决定（要考虑时钟频率和计数器预分频器）：由 TIM2\_CCR2 寄存器的值和 CNT 初始值决定  $t_{\text{DELAY}}$ ； $\text{TIM2\_ARR} - \text{TIM2\_CCR1}$  的值为  $t_{\text{PULSE}}$ 。

当发生比较匹配时要产生从 1 到 0 的波形，当计数器达到预装载值时要产生一个从 0 到 1 的波形：

- 1) 配置 TIM2\_CCMR1 寄存器  $\text{OC1M} = 111$ ，选择 PWM 模式 2。
- 2) 配置 TIM2\_CCER 寄存器  $\text{CC2P} = 1$ ，输出低电平有效。
- 3) 配置 TIM2\_CCMR1 中  $\text{OC1PE} = 1$  和 TIM2\_CR1 寄存器中  $\text{ARPE}$ ，使能预装载寄存器。

- 4) 配置 TIM2\_CCR1 寄存器和 TIM1\_ARR 寄存器。
- 5) 配置 TIM2\_EGR 寄存器 UG=1 产生一个更新事件。
- 6) 等待在 TI2 上的一个外部触发事件。

此例中, TIM2\_CR1 寄存器中的 DIR=0、CMS=0、OPM=1, 在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。

#### 14.6.3.4.1 OCx 快速使能

OCx 快速使能, 是单脉冲模式的一种特殊情况。在单脉冲模式下, 通过设置 TIM2\_CCMR 寄存器的 OCxFE=1, 强制 OCxREF 直接响应激励而不是依赖计数器和比较值之间的比较结果, 输出波形和比较匹配时的波形一样。这样可以去除比较的时间, 快速输出比较结果。OCx 快速输出使能只在 PWM 模式下生效。

### 14.6.4 DMA 模式

TIM3 能够在发生单个事件时生成多个 DMA 请求。主要目的是在没有软件开销的情况下, 多次重新编程 TIM2 的一部分也可以用于按周期读取数个寄存器。

TIM2\_DCR 和 TIM2\_DMAR 寄存器跟 DMA 模式相关。DMA 控制器的目标是唯一的, 必须指向 TIM2\_DMAR 寄存器。开启 DMA 使能后, 在给定的 TIM2 事件发生时, TIM2 会给 DMA 发送请求。对 TIM2\_DMAR 寄存器的每次写操作都被重定向到一个 TIM2 寄存器。

TIM2\_DCR 寄存器的 DBL 位定义了 DMA 连续传送的长度, 即传输寄存器数量; 当对 TIM2\_DMAR 进行读写操作时, 定时器识别 DBL, 确定传输的寄存器数量。TIM2\_DCR 寄存器的 DBA 位定义了 DMA 传输的基址, 定义从 TIM2\_CR1 寄存器地址开始的偏移量(00000 为 TIM2\_CR1; 00001 为 TIM2\_CR2; ...; 00110 为 TIM2\_CCMR1 等)。

例: DMA 模式用于在发生更新事件时更新 CCR1、CCR2、CCR3 寄存器的内容。具体配置如下:

- 1) 配置相应的 DMA 通道。
- 2) 配置 TIM2\_DCR 寄存器 DBA=01101, 配置 DMA 的基址, 选择偏移地址为 TIM2\_CCR1 寄存器的地址。
- 3) 配置 TIM2\_DCR 寄存器 DBL=00010, 配置传输长度为 3。
- 4) 配置 TIM2\_DIER 寄存器 UDE=1, 允许更新事件的 DMA 请求。
- 5) 配置 TIM2\_CR1 寄存器 CEN=1, 启动计数器。
- 6) 使能 DMA 通道。

此例中 TIM2 的 CCR1、CCR2、CCR3 更新一次, 在第一个 DMA 请求时, 数据 1 传到 CCR1 上; 在第二个 DMA 请求时, 数据 2 传到 CCR2 上; 在第三个 DMA 请求时, 数据 3 传到 CCR3 上。

### 14.6.5 调试模式

配置 DBG\_CR 寄存器中 DBG\_TIM2\_STOP=1, TIM2 计数器在 CPU 内核停止工作时(调试设置的断点)停止计数。(详见调试章节)

## 14.7 寄存器描述

表 14-3 TIM2 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	TIM2_CR1	控制寄存器 1	0x0000
0x04	TIM2_CR2	控制寄存器 2	0x0000
0x08	TIM2_SMCR	从模式控制寄存器	0x0000
0x0C	TIM2_DIER	DMA/中断使能寄存器	0x00000000
0x10	TIM2_SR	状态寄存器	0x00000000
0x14	TIM2_EGR	事件产生寄存器	0x00000000
0x18	TIM2_CCMR1	捕获/比较模式寄存器 1	0x0000
0x1C	TIM2_CCMR2	捕获/比较模式寄存器 2	0x0000
0x20	TIM2_CCER	捕获/比较使能寄存器	0x0000
0x24	TIM2_CNT	计数器	0x0000
0x28	TIM2_PSC	预分频率器	0x0000
0x2C	TIM2_ARR	自动装载寄存器	0x0000
0x34	TIM2_CCR1	捕获/比较寄存器 1	0x0000
0x38	TIM2_CCR2	捕获/比较寄存器 2	0x0000
0x3C	TIM2_CCR3	捕获/比较寄存器 3	0x0000
0x40	TIM2_CCR4	捕获/比较寄存器 4	0x0000
0x48	TIM2_DCR	DMA 控制寄存器	0x0000
0x4C	TIM2_DMAR	连续模式的 DMA 地址	0x0000
0x50	TIM2_OR	TIMERx 选项寄存器	0x0000

### 14.7.1 控制寄存器 1 (TIM2\_CR1)

偏移地址: 0x0

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Res.						CKD		APRE		CMS		DIR	OPM	URS	UDIS	CEN
Type							rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
15:10	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
9:8	CKD	rw	0x0	<p>时钟分频 (clock division)</p> <p>在定时器时钟 (INT_CK) 频率、死区时间和由死区发生器与数字滤波器 (ETR, TIx) 所用的采样时钟之间的分频比例。</p> <p>00: tDTS = tINT_CK 01: tDTS = 2x tINT_CK 10: tDTS = 4x tINT_CK 11: 保留, 不要使用这个配置</p>
7	APRE	rw	0x0	<p>自动重装载预装载使能 (Auto-reload preload enable)</p> <p>0: 关闭 TIM2_ARR 寄存器的影子寄存器 1: 使能 TIM2_ARR 寄存器的影子寄存器</p>
6:5	CMS	rw	0x0	<p>中央对齐模式选择 (Center-aligned mode selection)</p> <p>00: 边沿对齐模式。计数方向取决于 DIR 位 01: 中央对齐模式 1。计数器交替地递增和递减计数。通道为输出模式, 只在计数器递减计数时比较中断标志位被置 1 10: 中央对齐模式 2。计数器交替地递增和递减计数。通道为输出模式, 只在计数器递增计数时比较中断标志位被置 1 11: 中央对齐模式 3。计数器交替地递增和递减计数。通道为输出模式, 在计数器递增和递减计数时比较中断标志位均被置 1  注: 计数过程中, 不允许对齐模式的更改。</p>
4	DIR	rw	0x0	<p>计数方向 (Direction)</p> <p>0: 计数器递增计数 1: 计数器递减计数  注: 当计数器配置为中央对齐模式或编码器模式时, 该位为只读。</p>
3	OPM	rw	0x0	<p>单脉冲模式 (One pulse mode)</p> <p>0: 禁止单脉冲模式, 在发生更新事件时, 计数器继续计数 1: 使能单脉冲模式, 在发生下一次更新事件 (清除 CEN 位) 时, 计数器停止计数</p>

Bit	Field	Type	Reset	Description
2	URS	rw	0x0	<p>更新请求源 (Update request source) 软件配置该位，选择更新事件源。</p> <p>0: 以下事件可产生一个更新中断或 DMA 请求:</p> <ul style="list-style-type: none"> <li>- 计数器溢出/下溢</li> <li>- 设置 UG 位</li> <li>- 从模式控制器产生的更新</li> </ul> <p>1: 只有计数器溢出/下溢才产生一个更新中断或 DMA 请求</p>
1	UDIS	rw	0x0	<p>禁止更新 (Update disable) 该位用来允许或禁止更新事件的产生</p> <p>0: 允许更新事件 (UEV)。更新事件源:</p> <ul style="list-style-type: none"> <li>- 计数器溢出/下溢</li> <li>- 设置 UG 位为 1</li> <li>- 从模式控制器产生一个更新事件。</li> </ul> <p>1: 禁止更新事件。不产生更新事件，影子寄存器 (ARR、PSC、CCRx) 保持值不变。如果设置了 EGR_UG 位为 1，或者从模式控制器接收到硬件复位，计数器和预分频器被更新为其对应的影子寄存器的值。</p>
0	CEN	rw	0x0	<p>允许计数器 (Counter enable)</p> <p>0: 禁止计数器</p> <p>1: 使能计数器。 注: 在软件设置了 CEN 位后, 外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。</p>

### 14.7.2 控制寄存器 2 (TIM2\_CR2)

偏移地址: 0x04

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.							TI1S	MMS			CCDS	Res.			
Type								rw	rw			rw				

Bit	Field	Type	Reset	Description
15:8	Reserved			保留, 始终读为 0
7	TI1S	rw	0x0	<p>TI1 选择 (TI1 selection)</p> <p>0: TIM2_CH1 管脚连到 TI1 输入</p> <p>1: TIM2_CH1、TIM3_CH2 和 TIM3_CH3 管脚经异或后 的结果作为 TI1 输入</p>
6:4	MMS	rw	0x0	<p>主模式选择 (Master mode selection)</p> <p>这些位控制 TRGO 信号的选择, 用于选择在主模式下送到 从定时器的同步信息:</p> <p>000: 复位 TIM2_EGR 寄存器的 UG 位或从模式控制器 产生复位触发一次 TRGO 脉冲。从模式控制器产生复位 时, TRGO 信号相对实际的复位会有一个延时。</p> <p>001: 使能 用于控制在一定时间内使能从定时器或同时 启动多个定时器。计数器使能信号 CNT_EN 被用于作为 触发输出 (TRGO), 计数器使能信号是通过 CEN 控制 位和门控模式下的触发输入信号的逻辑或产生。当计数器 使能信号受控于触发输入时, TRGO 上会有一个延迟, 除 非选择了主/从模式。</p> <p>010: 更新 更新事件被选为 TRGO。</p> <p>011: 捕获/比较脉冲 发生一次捕获或一次比较成功时, 触发输出送出一个 TRGO 信号。</p> <p>100: 比较 OC1REF 信号被用于作为触发输出 (TRGO)</p> <p>101: 比较 OC2REF 信号被用于作为触发输出 (TRGO)</p> <p>110: 比较 OC3REF 信号被用于作为触发输出 (TRGO)</p> <p>111: 比较 OC4REF 信号被用于作为触发输出 (TRGO)</p>
3	CCDS	rw	0x0	<p>DMA 请求源选择 (Capture/compare DMA selection)</p> <p>0: 当 CCx 发生捕获/比较事件时, 发送 CCx 的 DMA 请 求</p> <p>1: 当 CCx 发生更新事件时, 发送 CCx 的 DMA 请 求</p>
2:0	Reserved			保留, 始终读为 0

### 14.7.3 从模式控制寄存器 (TIM2\_SMCR)

偏移地址: 0x08

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ETP	ECE	ETPS		ETF				MSM	TS			OCCS	SMS		
Type	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15	ETP	rw	0x0	外部触发极性 (External trigger polarity) 该位选择 ETR 信号的极性。 0: 高电平或上升沿有效 1: 低电平或下降沿有效
14	ECE	rw	0x0	外部时钟使能位 (External clock enable) 该位启用外部时钟模式 2。 0: 禁止外部时钟模式 2 1: 使能外部时钟模式 2, ETRF 信号上的任意有效沿驱动计数器计数。 注 1: 配置 ECE=1 与配置 SMS = 111 和 TS = 111 效果一样。 注 2: TS ≠ 111 时, 复位模式, 门控模式和触发模式可以与外部时钟模式 2 同时使用。 注 3: 同时使能外部时钟模式 1 和外部时钟模式 2 时, 外部时钟的输入是 ETRF。
13: 12	ETPS	rw	0x0	外部触发预分频 (External trigger prescaler) 外部触发信号 ETRP 的频率必须低于 TIM3_CLK 频率的 1/4。当输入较快的外部时钟时, 可以使用预分频降低 ETRP 的频率。 00: 关闭预分频 01: ETRP 频率除以 2 10: ETRP 频率除以 4 11: ETRP 频率除以 8

Bit	Field	Type	Reset	Description
11: 8	ETF	rw	0x0	<p>外部触发滤波 (External trigger filter)</p> <p>这些位定义了对 ETRP 信号采样的频率和对 ETRP 数字滤波的带宽。实际上，数字滤波器是一个事件计数器，它记录到 N 个事件后会产生一个输出的跳变。</p> <p>0000: 无滤波器，以 <math>f_{DTS}</math> 采样</p> <p>0001: 采样频率 <math>f_{SAMPLING} = f_{INT\_CK}, N=2</math></p> <p>0010: 采样频率 <math>f_{SAMPLING} = f_{INT\_CK}, N=4</math></p> <p>0011: 采样频率 <math>f_{SAMPLING} = f_{INT\_CK}, N=8</math></p> <p>0100: 采样频率 <math>f_{SAMPLING} = f_{DTS}/2, N=6</math></p> <p>0101: 采样频率 <math>f_{SAMPLING} = f_{DTS}/2, N=8</math></p> <p>0110: 采样频率 <math>f_{SAMPLING} = f_{DTS}/4, N=6</math></p> <p>0111: 采样频率 <math>f_{SAMPLING} = f_{DTS}/4, N=8</math></p> <p>1000: 采样频率 <math>f_{SAMPLING} = f_{DTS}/8, N=6</math></p> <p>1001: 采样频率 <math>f_{SAMPLING} = f_{DTS}/8, N=8</math></p> <p>1010: 采样频率 <math>f_{SAMPLING} = f_{DTS}/16, N=5</math></p> <p>1011: 采样频率 <math>f_{SAMPLING} = f_{DTS}/16, N=6</math></p> <p>1100: 采样频率 <math>f_{SAMPLING} = f_{DTS}/16, N=8</math></p> <p>1101: 采样频率 <math>f_{SAMPLING} = f_{DTS}/32, N=5</math></p> <p>1110: 采样频率 <math>f_{SAMPLING} = f_{DTS}/32, N=6</math></p> <p>1111: 采样频率 <math>f_{SAMPLING} = f_{DTS}/32, N=8</math></p>
7	MSM	rw	0x0	<p>主/从模式 (Master/Slave mode)</p> <p>0: 无作用</p> <p>1: 触发输入 (TRGI) 上的事件被延迟了，以允许在当前定时器 (通过 TRGO) 与它的从定时器间的完美同步，这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。</p>

Bit	Field	Type	Reset	Description
6: 4	TS	rw	0x0	<p>触发选择 (Trigger selection)</p> <p>触发输入源选择。</p> <p>000: 内部触发 0 (ITR0)      001: 内部触发 1 (ITR1)      010: 内部触发 2 (ITR2)      011: 内部触发 3 (ITR3)      100: TI1 的边沿检测器 (TI1F_ED)      101: 滤波后的定时器输入 1 (TI1FP1)      110: 滤波后的定时器输入 2 (TI1FP2)      111: 外部触发输入 (ETRF)</p> <p>更多有关 ITRx 的细节, 参见下表。</p> <p>注: 从模式使能后这些位不能修改</p>
3	OCCS	rw	0x0	<p>比较器输出信号清除选择 (Output compare clear selection)</p> <p>在 PWM 模式下, 清除比较器输出</p> <p>0: 外部触发信号作为清除信号      1: 比较器输出作为清除信号</p>

Bit	Field	Type	Reset	Description
2: 0	SMS	rw	0x0	<p>从模式选择 (Slave mode selection)</p> <p>当选择了外部信号, 触发信号 (TRGI) 的有效边沿与选中的外部输入极性相关。</p> <p>000: 关闭从模式-如果 CEN = 1, 则预分频器直接由内部时钟驱动。</p> <p>001: 编码器模式 1-根据 TI1FP1 的电平, 计数器在 TI2FP2 的边沿递增/递减计数。</p> <p>010: 编码器模式 2-根据 TI2FP2 的电平, 计数器在 TI1FP1 的边沿递增/递减计数。</p> <p>011: 编码器模式 3-根据另一个输入的电平, 计数器在 TI1FP1 和 TI2FP2 的边沿递增/递减计数。</p> <p>100: 复位模式-选中的触发输入 (TRGI) 的上升沿重新初始化计数器, 并且产生一个更新寄存器的信号。</p> <p>101: 门控模式-当触发输入 (TRGI) 为高时, 计数器开始计数。当触发输入变为低时, 计数器停止计数(但不复位)。计数器的启动和停止都是受控的。</p> <p>110: 触发模式-计数器在触发输入 TRGI 的上升沿启动(但不复位), 只有计数器的启动是受控的。</p> <p>111: 外部时钟模式 1-选中的触发输入 (TRGI) 的上升沿驱动计数器。</p> <p>注: 如果 TI1F_EN 被选为触发输入 (TS = 100) 时, 不要使用门控模式。这是因为, TI1F_EO 在每次 TI1F 变化时输出一个脉冲, 然而门控模式是要检查触发输入的电平。</p>

表 14-4 TIM2/3 内部触发连接

从定时器	ITR0	ITR1	ITR2	ITR3
TIM2	TIM1	TIM8	TIM3	TIM4
TIM3	TIM1	TIM2	TIM5	TIM4

#### 14.7.4 DMA/中断使能寄存器 (TIM2\_DIER)

偏移地址: 0x0C

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															

Id																
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Re s.	TDE	Res.	CC4 DE	CC3 DE	CC2 DE	CC1 DE	UD E	Re s.	TIE	Re s.	CC4 IE	CC3 IE	CC2 IE	CC 1IE	UIE
Type		rw		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:15	Reserved			保留, 始终读为 0
14	TDE	rw	0x0	允许触发 DMA 请求 (Trigger DMA request enable) 0: 禁止触发 DMA 请求 1: 允许触发 DMA 请求
13	Reserved			保留, 始终读为 0
12	CC4DE	rw	0x0	允许捕获/比较 4 的 DMA 请求 (Capture/Compare 4 DMA request enable) 0: 禁止捕获/比较 4 的 DMA 请求 1: 允许捕获/比较 4 的 DMA 请求
11	CC3DE	rw	0x0	允许捕获/比较 3 的 DMA 请求 (Capture/Compare 3 DMA request enable) 0: 禁止捕获/比较 3 的 DMA 请求 1: 允许捕获/比较 3 的 DMA 请求
10	CC2DE	rw	0x0	允许捕获/比较 2 的 DMA 请求 (Capture/Compare 2 DMA request enable) 0: 禁止捕获/比较 2 的 DMA 请求 1: 允许捕获/比较 2 的 DMA 请求
9	CC1DE	rw	0x0	允许捕获/比较 1 的 DMA 请求 (Capture/Compare 1 DMA request enable) 0: 禁止捕获/比较 1 的 DMA 请求 1: 允许捕获/比较 1 的 DMA 请求

Bit	Field	Type	Reset	Description
8	UDE	rw	0x0	允许更新的 DMA 请求 (Update DMA request enable) 0: 禁止更新 DMA 请求 1: 允许更新 DMA 请求
7	Reserved			保留, 始终读为 0
6	TIE	rw	0x0	允许触发中断 (Trigger interrupt enable) 0: 禁止触发中断 1: 允许触发中断
5	Reserved			保留, 始终读为 0
4	CC4IE	rw	0x0	允许捕获/比较 4 中断 (Capture/Compare 4 interrupt enable) 0: 禁止捕获/比较 4 中断 1: 允许捕获/比较 4 中断
3	CC3IE	rw	0x0	允许捕获/比较 3 中断 (Capture/Compare 3 interrupt enable) 0: 禁止捕获/比较 3 中断 1: 允许捕获/比较 3 中断
2	CC2IE	rw	0x0	允许捕获/比较 2 中断 (Capture/Compare 2 interrupt enable) 0: 禁止捕获/比较 2 中断 1: 允许捕获/比较 2 中断
1	CC1IE	rw	0x0	允许捕获/比较 1 中断 (Capture/Compare 1 interrupt enable) 0: 禁止捕获/比较 1 中断 1: 允许捕获/比较 1 中断
0	UIE	rw	0x0	允许更新中断 (Update interrupt enable) 0: 禁止更新中断 1: 允许更新中断

### 14.7.5 状态寄存器 (TIM2\_SR)

偏移地址: 0x10

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.			CC4 OF	CC3 OF	CC2 OF	CC1 OF	Res.		TIF	Res.	CC4IF	CC3IF	CC2IF	CC1IF	UIF
Type	r_w0c							r_w0c		r_w0c						

Bit	Field	Type	Reset	Description
31:13	Reserved			保留, 始终读为 0
12	CC4OF	r_w0c	0x0	捕获/比较 4 重复捕获标记 (Capture/Compare 4 overcapture flag) 参考 CC1OF 描述。
11	CC3OF	r_w0c	0x0	捕获/比较 3 重复捕获标记 (Capture/Compare 3 overcapture flag) 参考 CC1OF 描述。
10	CC2OF	r_w0c	0x0	捕获/比较 2 重复捕获标记 (Capture/Compare 2 overcapture flag) 参考 CC1OF 描述。
9	CC1OF	r_w0c	0x0	捕获/比较 1 重复捕获标记 (Capture/Compare 1 overcapture flag) 仅当通道 1 被配置为输入捕获, CC1F 已经位 1 后, 捕获事件再次发生时, 该标记可由硬件置 1。写 0 可清除该位。 0: 无重复捕获产生; 1: 重复捕获产生。
8:7	Reserved			保留, 始终读为 0
6	TIF	r_w0c	0x0	触发器中断标记 (Trigger interrupt flag) 当发生触发事件 (当从模式控制器处于除门控模式外的其它模式时, 在 TRGI 输入端检测到有效边沿, 或门控模式下的任一边沿) 时由硬件对该位置 1。它由软件清 0。 0: 无触发器事件产生 1: 触发器中断产生
5	Reserved			保留, 始终读为 0
4	CC4IF	r_w0c	0x0	捕获/比较 4 中断标记 (Capture/Compare 4 interrupt flag) 参考 CC1IF 描述。
3	CC3IF	r_w0c	0x0	捕获/比较 3 中断标记 (Capture/Compare 3 interrupt flag) 参考 CC1IF 描述。

Bit	Field	Type	Reset	Description
2	CC2IF	r_w0c	0x0	<p>捕获/比较 2 中断标记 (Capture/Compare 2 interrupt flag)</p> <p>参考 CC1IF 描述。</p>
1	CC1IF	r_w0c	0x0	<p>捕获/比较 1 中断标记 (Capture/Compare 1 interrupt flag)</p> <p>通道 1 为输出模式:</p> <p>当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外。它由软件清 0。</p> <p>0: 无匹配发生 1: TIM2_CNT 的值与 TIM2_CCR1 的值匹配。</p> <p>通道 1 为输入模式:</p> <p>当发生捕获事件时该位由硬件置 1，由软件或读取 TIM2_CCR1 的值清 0。</p> <p>0: 无输入捕获产生 1: 计数器值已被捕获至 TIM2_CCR1</p>
0	UIF	r_w0c	0x0	<p>更新中断标记 (Update interrupt flag)</p> <p>当产生更新事件时该位由硬件置 1。它由软件清 0。</p> <p>0: 无更新中断发生 1: 发生更新中断。</p> <p>当寄存器被更新时该位由硬件置 1:</p> <ul style="list-style-type: none"> <li>-若 TIM2_CR1 寄存器的 UDIS=0、URS=0，当 TIM2_EGR 寄存器的 UG=1 时产生更新事件。</li> <li>- 若 TIM2_CR1 寄存器的 UDIS=0、URS=0，当计数器被触发事件重初始化时产生更新事件。</li> </ul>

#### 14.7.6 事件产生寄存器 (TIM2\_EGR)

偏移地址: 0x14

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:7	Reserved			保留, 始终读为 0
6	TG	w	0x0	<p>产生触发事件 (Trigger generation)</p> <p>0: 无动作</p> <p>1: 产生触发事件, TIM2_SR 寄存器的 TIF = 1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA, 由硬件自动清 0。</p>
5	COMG	w	0x0	<p>捕获/比较事件, 产生控制更新 (Capture/Compare control update generation)</p> <p>0: 无动作</p> <p>1: 捕获/比较事件控制更新产生, 由硬件自动清 0, 当 CCPC=1, 允许更新 CCxE、CCxNE、OCxM 位。</p> <p>注: 该位只对拥有互补输出的通道有效。</p>
4	CC4G	w	0x0	<p>产生捕获/比较 4 事件 (Capture/Compare 4 generation)</p> <p>参考 CC1G 描述。</p>
3	CC3G	w	0x0	<p>产生捕获/比较 3 事件 (Capture/Compare 3 generation)</p> <p>参考 CC1G 描述。</p>
2	CC2G	w	0x0	<p>产生捕获/比较 2 事件 (Capture/Compare 2 generation)</p> <p>参考 CC1G 描述。</p>
1	CC1G	w	0x0	<p>产生通道 1 捕获/比较事件 (Capture/Compare 1 generation)</p> <p>该位由软件置 1, 用于产生一个捕获/比较事件, 由硬件自动清 0。</p> <p>0: 无动作</p> <p>1: 在通道 CC1 上产生一个捕获/比较事件:</p> <p>若通道 CC1 配置为输出: 设置 CC1IF=1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。</p> <p>若通道 CC1 配置为输入: 当前的计数器值被捕获至 TIM2_CCR1 寄存器, 设置 CC1IF = 1, 若开启对应的中断和 DMA, 则产生相应的中断和 DMA。若 CC1IF 已经为 1, 则设置 CC1OF = 1。</p>
0	UG	w	0x0	<p>产生更新事件 (Update generation)</p> <p>0: 无动作</p> <p>1: 重置计数器值, 并产生一个更新事件。由硬件自动清 0, 如果选择了中央对齐或递增计数模式, 计数器被清 0; 否则(递减计数模式)计数器将载入自动重载值。预分频计数器将同时被清除。</p>

### 14.7.7 捕获/比较模式寄存器 (TIM2\_CCMR1)

偏移地址: 0x18

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	OC2CE	OC2M			OC2PE	OC2FE	CC2S		OC1CE	OC1M			OC1PE	OC1FE	CC1S	
	IC2F			IC2PSC					IC1F				IC1PSC			
Type	rw	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

输出比较模式:

Bit	Field	Type	Reset	Description
15	OC2CE	rw	0x0	通道 2 输出比较清零使能 (Output compare 2 clear enable)
14:12	OC2M	rw	0x0	通道 2 输出比较模式 (Output compare 2 mode)
11	OC2PE	rw	0x0	通道 2 输出比较预装载使能 (Output compare 2 preload enable)
10	OC2FE	rw	0x0	通道 2 输出比较快速使能 (Output compare 2 fast enable)
9:8	CC2S	rw	0x0	通道 2 捕获/比较选择 (Capture/Compare 2 selection) 该位定义通道的方向和输入信号的选择, 只有在通道关闭时这些位才可写入: 00: 通道 2 被配置为输出 01: 通道 2 被配置为输入, IC2 映射在 TI2 上 10: 通道 2 被配置为输入, IC2 映射在 TI1 上 11: 通道 2 被配置为输入, IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIM2_SMCR 寄存器的 TS 位选择)
7	OC1CE	rw	0x0	通道 1 输出比较清 0 使能 (Output compare 1 clear enable) 0: OC1REF 不受 ETRF 输入的影响 1: 当检测到 ETRF 输入高电平时, 清除 OC1REF = 0

Bit	Field	Type	Reset	Description
6:4	OC1M	rw	0x0	<p>通道 1 输出比较模式 (Output compare 1 mode)</p> <p>该位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。</p> <p>000：冻结。TIM2_CCR1 与 TIM2_CNT 间的比较结果对 OC1REF 不起作用</p> <p>001：匹配时设置为高。当 TIM2_CNT 的值与 TIM2_CCR1 的值相同时，强制 OC1REF 为高电平</p> <p>010：匹配时设置为低。当 TIM2_CNT 的值与 TIM2_CCR1 的值相同时，强制 OC1REF 为低电平</p> <p>011：匹配时翻转。当 TIM2_CCR1=TIM2_CNT 时，翻转 OC1REF 的电平</p> <p>100：强制为低。强制 OC1REF 为低电平</p> <p>101：强制为高。强制 OC1REF 为高电平</p> <p>110：PWM 模式 1。在递增计数时，当 TIM2_CNT&lt;TIM2_CCR1 时通道 1 为有效电平，否则为无效电平；在递减计数时，当 TIM2_CNT &gt; TIM2_CCR1 时通道 1 为无效电平，否则为有效电平。</p> <p>111：PWM 模式 2。在递增计数时，当 TIM2_CNT&lt;TIM2_CCR1 时通道 1 为无效电平，否则为有效电平；在递减计数时，当 TIM2_CNT&gt;TIM2_CCR1 时通道 1 为有效电平，否则为无效电平</p> <p>注：在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时，OC1REF 电平才改变。</p>
3	OC1PE	rw	0x0	<p>通道 1 输出比较预装载使能 (Output compare 1 preload enable)</p> <p>0：禁止 TIM2_CCR1 寄存器的预装载功能，写入 TIM2_CCR1 寄存器的数值立即生效</p> <p>1：开启 TIM2_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，TIM2_CCR1 的预装载值在更新事件到来时生效</p> <p>注：仅在单脉冲模式下 (TIM2_CR1 寄存器的 OPM=1)，可以在未确认预装载寄存器情况下使用 PWM 模式，否则其动作不确定。</p>

Bit	Field	Type	Reset	Description
2	OC1FE	rw	0x0	<p>通道 1 输出比较快速使能 (Output compare 1 fast enable)</p> <p>该位为 1 时, 若通道配置为 PWM 模式, 会加快捕获/比较输出对触发时间的响应。输出通道将触发输入信号的有效边沿的作用等同于发生了一次比较匹配, 因此 OC 被设置为比较电平而与比较结果无关。</p> <p>0: 禁止通道 1 输出比较快速使能, 当检测到触发输入的一个有效边沿时, 激活 OC1 的最小延时需要 5 个时钟周期</p> <p>1: 开启通道 1 输出比较快速使能。当触发输入由一个有效边沿时, 激活 OC1 的最小延时缩短到 3 个周期</p>
1:0	CC1S	rw	0x0	<p>通道 1 捕获/比较选择 (Capture/Compare 1 selection)</p> <p>该位定义通道的方向和输入信号的选择, 只有在通道关闭时这些位才可写入:</p> <p>00: 通道 1 被配置为输出</p> <p>01: 通道 1 被配置为输入, IC1 映射在 TI1 上</p> <p>10: 通道 1 被配置为输入, IC1 映射在 TI2 上</p> <p>11: 通道 1 被配置为输入, IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIM2_SMCR 寄存器的 TS 位选择)</p>

输入捕获模式:

Bit	Field	Type	Reset	Description
15:12	IC2F	rw	0x0	输入捕获 2 滤波器 (Input capture 2 filter)
11:10	IC2PSC	rw	0x0	输入/捕获 2 预分频器 (Input capture 2 prescaler)
9:8	CC2S	rw	0x0	<p>捕获/比较 2 选择 (Capture/Compare 2 selection)</p> <p>该位定义通道的方向和输入信号的选择, 只有在通道关闭时这些位才可写入:</p> <p>00: CC2 通道被配置为输出</p> <p>01: CC2 通道被配置为输入, IC2 映射在 TI2 上</p> <p>10: CC2 通道被配置为输入, IC2 映射在 TI1 上</p> <p>11: CC2 通道被配置为输入, IC2 映射在 TRC 上,</p> <p>此模式仅工作在内部触发器输入被选中时 (由 TIM2_SMCR 寄存器的 TS 位选择)</p>

Bit	Field	Type	Reset	Description
7:4	IC1F	rw	0x0	<p>通道 1 输入捕获滤波器 (Input capture 1 filter)</p> <p>数字滤波器由一个事件计数器组成, 它记录 N 个输入事件后会产生一个输出的跳变。这些位定义了 IC1 输入信号的采样频率和数字滤波器的长度。</p> <p>0000: 无滤波器, 以 <math>f_{DTS}</math> 采样</p> <p>1000: 采样频率 <math>f_{SAMPLING}=f_{DTS} /8, N=6</math></p> <p>0001: 采样频率 <math>f_{SAMPLING}=f_{INT\_CK}, N=2</math></p> <p>1001: 采样频率 <math>f_{SAMPLING}=f_{DTS} /8, N=8</math></p> <p>0010: 采样频率 <math>f_{SAMPLING}=f_{INT\_CK}, N=4</math></p> <p>1010: 采样频率 <math>f_{SAMPLING}=f_{DTS} /16, N=5</math></p> <p>0011: 采样频率 <math>f_{SAMPLING}=f_{INT\_INT}, N=8</math></p> <p>1011: 采样频率 <math>f_{SAMPLING}=f_{DTS} /16, N=6</math></p> <p>0100: 采样频率 <math>f_{SAMPLING}=f_{DTS} /2, N=6</math></p> <p>1100: 采样频率 <math>f_{SAMPLING}=f_{DTS} /16, N=8</math></p> <p>0101: 采样频率 <math>f_{SAMPLING}=f_{DTS} /2, N=8</math></p> <p>1101: 采样频率 <math>f_{SAMPLING}=f_{DTS} /32, N=5</math></p> <p>0110: 采样频率 <math>f_{SAMPLING}=f_{DTS} /4, N=6</math></p> <p>1110: 采样频率 <math>f_{SAMPLING}=f_{DTS} /32, N=6</math></p> <p>0111: 采样频率 <math>f_{SAMPLING}=f_{DTS} /4, N=8</math></p> <p>1111: 采样频率 <math>f_{SAMPLING}=f_{DTS} /32, N=8</math></p>
3:2	IC1PSC	rw	0x0	<p>通道 1 输入/捕获预分频器 (Input capture 1 prescaler)</p> <p>这 2 位定义了 IC1 的预分频系数。当 CC1E=0 (TIM2_CCER 寄存器中) 时, 预分频器复位。</p> <p>00: 无预分频器, 捕获输入上检测到的每一个边沿都触发一次捕获</p> <p>01: 每 2 个事件触发一次捕获</p> <p>10: 每 4 个事件触发一次捕获</p> <p>11: 每 8 个事件触发一次捕获</p>

Bit	Field	Type	Reset	Description
1:0	CC1S	rw	0x0	<p>捕获/比较 1 选择 (Capture/Compare 1 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <ul style="list-style-type: none"> <li>00: CC1 通道被配置为输出</li> <li>01: CC1 通道被配置为输入，IC1 映射在 TI1 上</li> <li>10: CC1 通道被配置为输入，IC1 映射在 TI2 上</li> <li>11: CC1 通道被配置为输入，IC1 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时（由 TIM2_SMCR 寄存器的 TS 位选择）</li> </ul>

#### 14.7.8 捕获/比较模式寄存器 2 (TIM2\_CCMR2)

偏移地址: 0x1C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	OC4CE	OC4M			OC4PE	OC4FE	CC4S	OC3CE	OC3M	OC3PE	OC3FE	CC3S				
	IC4F				IC4PSC			IC3F	IC3PSC							
Type	rw	rw			rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

输出比较模式：

Bit	Field	Type	Reset	Description
15	OC4CE	rw	0x0	通道 4 输出比较清零使能 (Output compare 4 clear enable)
14:12	OC4M	rw	0x0	通道 4 输出比较模式 (Output compare 4 mode)
11	OC4PE	rw	0x0	通道 4 输出比较预装载使能 (Output compare 4 preload enable)
10	OC4FE	rw	0x0	通道 4 输出比较快速使能 (Output compare 4 fast enable)
9:8	CC4S	rw	0x0	<p>通道 4 捕获/比较选择 (Capture/Compare 4 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <ul style="list-style-type: none"> <li>00: 通道 4 被配置为输出</li> <li>01: 通道 4 被配置为输入，IC4 映射在 TI4 上</li> <li>10: 通道 4 被配置为输入，IC4 映射在 TI3 上</li> <li>11: 通道 4 被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时（由 TIM2_SMCR 寄存器的 TS 位选择）</li> </ul>

Bit	Field	Type	Reset	Description
7	OC3CE	rw	0x0	<p>通道 3 输出比较清 0 使能 (Output compare 3 clear enable)</p> <p>0: OC1REF 不受 ETRF 输入的影响</p> <p>1: 当检测到 ETRF 输入高电平时, 清除 OC1REF = 0</p>
6:4	OC3M	rw	0x0	<p>通道 3 输出比较模式 (Output compare 3 mode)</p> <p>该位定义了输出参考信号 OC3REF 的动作, 而 OC3REF 决定了 OC3、OC3N 的值。OC3REF 是高电平有效, 而 OC3、OC3N 的有效电平取决于 CC3P、CC3NP 位。</p> <p>000: 冻结。TIM2_CCR3 与 TIM2_CNT 间的比较结果对 OC3REF 不起作用</p> <p>001 : 匹配时设置为高。当 TIM2_CNT 的值与 TIM2_CCR3 的值相同时, 强制 OC3REF 为高电平</p> <p>010 : 匹配时设置为低。当 TIM2_CNT 的值与 TIM2_CCR3 的值相同时, 强制 OC3REF 为低电平</p> <p>011: 匹配时翻转。当 TIM2_CCR3=TIM2_CNT 时, 翻转 OC3REF 的电平</p> <p>100: 强制为低。强制 OC3REF 为低电平</p> <p>101: 强制为高。强制 OC3REF 为高电平</p> <p>110: PWM 模式 1。在递增计数时, 当 TIM2_CNT&lt;TIM2_CCR3 时通道 3 为有效电平, 否则为无效电平; 在递减计数时, 当 TIM2_CNT&gt;TIM2_CCR3 时通道 3 为无效电平, 否则为有效电平。</p> <p>111: PWM 模式 2。在递增计数时, 当 TIM2_CNT&lt;TIM2_CCR3 时通道 3 为无效电平, 否则为有效电平; 在递减计数时, 当 TIM2_CNT&gt;TIM2_CCR3 时通道 3 为有效电平, 否则为无效电平</p> <p>注: 在 PWM 模式 1 或 PWM 模式 2 中, 只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时, OC3REF 电平才改变。</p>
3	OC3PE	rw	0x0	<p>通道 3 输出比较预装载使能 (Output compare 3 preload enable)</p> <p>0: 禁止 TIM2_CCR3 寄存器的预装载功能, 写入 TIM2_CCR3 寄存器的数据立即生效</p> <p>1: 开启 TIM2_CCR3 寄存器的预装载功能, 读写操作仅对预装载寄存器操作, TIM2_CCR3 的预装载值在更新事件到来时生效</p> <p>注: 仅在单脉冲模式下 (TIM2_CR1 寄存器的 OPM= 1), 可以在未确认预装载寄存器情况下使用 PWM 模式, 否则其动作不确定。</p>

Bit	Field	Type	Reset	Description
2	OC3FE	rw	0x0	<p>通道 3 输出比较快速使能 (Output compare 3 fast enable)</p> <p>该位为 1 时，若通道配置为 PWM 模式，会加快捕获/比较输出对触发时间的响应。输出通道将触发输入信号的有效边沿的作用等同于发生了一次比较匹配，因此 OC 被设置为比较电平而与比较结果无关。</p> <p>0: 禁止通道 3 输出比较快速使能，当检测到触发输入的一个有效边沿时，激活 OC1 的最小延时需要 5 个时钟周期</p> <p>1: 开启通道 3 输出比较快速使能。当触发输入由一个有效边沿时，激活 OC1 的最小延时缩短到 3 个周期</p>
1:0	CC3S	rw	0x0	<p>通道 3 捕获/比较选择 (Capture/Compare 3 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <p>00: 通道 3 被配置为输出</p> <p>01: 通道 3 被配置为输入，IC3 映射在 TI3 上</p> <p>10: 通道 3 被配置为输入，IC3 映射在 TI4 上</p> <p>11: 通道 3 被配置为输入，IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 TIM2_SMCR 寄存器的 TS 位选择)</p>

输入捕获模式：

Bit	Field	Type	Reset	Description
15:12	IC4F	rw	0x0	输入捕获 4 滤波器 (Input capture 4 filter)
11:10	IC4PSC	rw	0x0	输入/捕获 4 预分频器 (Input capture 4 prescaler)
9:8	CC4S	rw	0x0	<p>捕获/比较 4 选择 (Capture/Compare 4 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <p>00: CC4 通道被配置为输出</p> <p>01: CC4 通道被配置为输入，IC4 映射在 TI4 上</p> <p>10: CC4 通道被配置为输入，IC4 映射在 TI3 上</p> <p>11: CC4 通道被配置为输入，IC4 映射在 TRC 上， 此模式仅工作在内部触发器输入被选中时 (由 TIM2_SMCR 寄存器的 TS 位选择)</p>

Bit	Field	Type	Reset	Description
7:4	IC3F	rw	0x0	<p>通道 3 输入捕获滤波器 (Input capture 3 filter)</p> <p>数字滤波器由一个事件计数器组成，它记录 N 个输入事件后会产生一个输出的跳变。这些位定义了 IC1 输入信号的采样频率和数字滤波器的长度。</p> <ul style="list-style-type: none"> <li>0000: 无滤波器，以 <math>f_{DTS}</math> 采样</li> <li>1000: 采样频率 <math>f_{SAMPLING} = f_{DTS} / 8, N=6</math></li> <li>0001: 采样频率 <math>f_{SAMPLING} = f_{INT\_CK}, N=2</math></li> <li>1001: 采样频率 <math>f_{SAMPLING} = f_{DTS} / 8, N=8</math></li> <li>0010: 采样频率 <math>f_{SAMPLING} = f_{INT\_CK}, N=4</math></li> <li>1010: 采样频率 <math>f_{SAMPLING} = f_{DTS} / 16, N=5</math></li> <li>0011: 采样频率 <math>f_{SAMPLING} = f_{INT\_CK}, N=8</math></li> <li>1011: 采样频率 <math>f_{SAMPLING} = f_{DTS} / 16, N=6</math></li> <li>0100: 采样频率 <math>f_{SAMPLING} = f_{DTS} / 2, N=6</math></li> <li>1100: 采样频率 <math>f_{SAMPLING} = f_{DTS} / 16, N=8</math></li> <li>0101: 采样频率 <math>f_{SAMPLING} = f_{DTS} / 2, N=8</math></li> <li>1101: 采样频率 <math>f_{SAMPLING} = f_{DTS} / 32, N=5</math></li> <li>0110: 采样频率 <math>f_{SAMPLING} = f_{DTS} / 4, N=6</math></li> <li>1110: 采样频率 <math>f_{SAMPLING} = f_{DTS} / 32, N=6</math></li> <li>0111: 采样频率 <math>f_{SAMPLING} = f_{DTS} / 4, N=8</math></li> <li>1111: 采样频率 <math>f_{SAMPLING} = f_{DTS} / 32, N=8</math></li> </ul>
3:2	IC3PSC	rw	0x0	<p>通道 3 输入/捕获预分频器 (Input capture 3 prescaler)</p> <p>这 2 位定义了 IC3 的预分频系数。当 CC3E=0 (TIM2_CCER 寄存器中) 时，预分频器复位。</p> <ul style="list-style-type: none"> <li>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</li> <li>01: 每 2 个事件触发一次捕获</li> <li>10: 每 4 个事件触发一次捕获</li> <li>11: 每 8 个事件触发一次捕获</li> </ul>

Bit	Field	Type	Reset	Description
1:0	CC3S	rw	0x0	<p>捕获/比较 3 选择 (Capture/Compare 3 selection)</p> <p>该位定义通道的方向和输入信号的选择，只有在通道关闭时这些位才可写入：</p> <ul style="list-style-type: none"> <li>00: CC3 通道被配置为输出</li> <li>01: CC3 通道被配置为输入，IC3 映射在 TI3 上</li> <li>10: CC3 通道被配置为输入，IC3 映射在 TI4 上</li> <li>11: CC3 通道被配置为输入，IC3 映射在 TRC 上，此模式仅工作在内部触发器输入被选中时（由 TIM2_SMCR 寄存器的 TS 位选择）</li> </ul>

#### 14.7.9 捕获/比较使能寄存器 (TIM2\_CCER)

偏移地址: 0x20

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CC4N P	Res . .	C C 4P	C C 4E	CC3N P	Res . .	C C 3P	C C 3E	CC2N P	Res . .	C C 2P	CC2 E	CC1N P	Res . .	CC1 P	CC1 E
Type	rw			rw	rw		rw	rw	rw		rw	rw	rw		rw	rw

Bit	Field	Type	Reset	Description
15	CC4NP	rw	0x0	<p>通道 4 输入 / 捕获互补输出极性 (Capture/Compare 4 complementary output polarity)</p> <p>参考 CC1NP 的描述。</p>
14	Reserved			保留，始终读为 0。
13	CC4P	rw	0x0	<p>通道 4 输入/捕获输出极性 (Capture/Compare 4 output polarity)</p> <p>参考 CC1P 的描述。</p>
12	CC4E	rw	0x0	<p>通道 4 输入/捕获输出使能 (Capture/Compare 4 output enable)</p> <p>参考 CC1E 的描述。</p>
11	CC3NP	rw	0x0	<p>通道 3 输入/捕获互补输出极性 (Capture/Compare 3 complementary output polarity)</p> <p>参考 CC1NP 的描述。</p>
10	Reserved			保留，始终读为 0。

Bit	Field	Type	Reset	Description
9	CC3P	rw	0x0	通道 3 输入捕获输出极性 (Capture/Compare 3 output polarity) 参考 CC1P 的描述。
8	CC3E	rw	0x0	通道 3 输入/捕获输出使能 (Capture/Compare 3 output enable) 参考 CC1E 的描述。
7	CC2NP	rw	0x0	通道 2 输入/捕获互补输出极性 (Capture/Compare 2 complementary output polarity) 参考 CC1NP 的描述。
6	Reserved			保留, 始终读为 0。
5	CC2P	rw	0x0	通道 2 输入捕获输出极性 (Capture/Compare 2 output polarity) 参考 CC1P 的描述。
4	CC2E	rw	0x0	通道 2 输入/捕获输出使能 (Capture/Compare 2 output enable) 参考 CC1E 的描述。
3	CC1NP	rw	0x0	通道 1 输入 / 捕获互补输出极性 (Capture/Compare 1 complementary output polarity) 0: OC1N 高电平有效 1: OC1N 低电平有效
2	Reserved			保留, 始终读为 0。
1	CC1P	rw	0x0	通道 1 输入/捕获输出极性 (Capture/Compare 1 output polarity) 通道 1 配置为输出时, 此位定义了输出信号极性: 0: OC1 高电平有效 1: OC1 低电平有效 通道 1 配置为输入时, 此位定义了输入信号极性: 0: 不反相: 捕获发生在 IC1 的上升沿; 当用作外部触发器时, IC1 不反相。 1: 反相: 捕获发生在 IC1 的下降沿; 当用作外部触发器时, IC1 反相。

Bit	Field	Type	Reset	Description
0	CC1E	rw	0x0	<p>通道 1 输入/捕获输出使能 (Capture/Compare 1 output enable)</p> <p>通道 1 配置为输出时, :</p> <p>0: 关闭。OC1 禁止输出</p> <p>1: 开启。OC1 信号输出到对应的输出引脚.</p> <p>CC1 通道配置为输入:</p> <p>该位决定了输入捕获功能是否启用。</p> <p>0: 捕获禁止</p> <p>1: 捕获使能</p>

表 14-5 输入模式下, ICx 的极性选择如下表:

CCxP	CCxNP	ICx 极性
0	0	上升沿
1	0	下降沿
1	1	上升沿或下降沿
0	1	保留

### 14.7.10 计数器 (TIM2\_CNT)

偏移地址: 0x24

复位值: 0x0000

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

Bit	Field	Type	Reset	Description
31:16	CNT	rw	0x0000	计数器高 16 位值 (High Count value)

15:0	CNT	rw	0x0000	计数器低 16 位值 (Low Count value)
------	-----	----	--------	------------------------------

### 14.7.11 预分频器 (TIM2\_PSC)

偏移地址: 0x28

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	PSC															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	PSC	rw	0x0000	预分频器的值 (Prescaler value) 计数器的时钟频率 ( $\text{ck\_cnt}$ ) = $f_{CK\_PSC} / (PSC+1)$ 。 当发生更新事件时, PSC 的值装入当前预分频寄存器。

### 14.7.12 自动预装载寄存器 (TIM2\_ARR)

偏移地址: 0x2C

复位值: 0x0000

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

Bit	Field	Type	Reset	Description
31:16	ARR	rw	0x0000	自动重装载高 16 位值 (High Auto-reload value )
15:0	ARR	rw	0x0000	自动预装载低 16 位值 (Low Auto-reload value) 这些位定义了计数器的自动重载值。当自动重装载的值为 0 时, 计数器不工作。

### 14.7.13 捕获/比较寄存器 1 (TIM2\_CCR1)

偏移地址: 0x34

复位值: 0x0000

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

Bit	Field	Type	Reset	Description
31:16	CCR1	rw	0x0000	通道 1 捕获/比较高 16 位值 (High Capture/Compare 1 value)
15:0	CCR1	rw	0x0000	通道 1 捕获/比较低 16 位值 (Low Capture/Compare 1 value) 当通道 1 配置为输入时, CCR1 的值决定了上次捕获事件的计数器值 (此时寄存器为只读)。 当通道 1 配置为输出时, CCR1 的值为即将和计数器值比较的值。如果开启了预装载功能, 写入的值在更新事件发生时传输到当前捕获/比较寄存器; 否则写入的值立即载入捕获/比较寄存器。

#### 14.7.14 捕获/比较寄存器 2 (TIM2\_CCR2)

偏移地址: 0x3C

复位值: 0x0000

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

Bit	Field	Type	Reset	Description
31:16	CCR2	rw	0x0000	通道 2 捕获/比较高 16 位值 (High Capture/Compare 2 value)
15:0	CCR2	rw	0x0000	通道 2 捕获/比较低 16 位值 (Low Capture/Compare 2 value) 详见 CCR1 描述

#### 14.7.15 捕获/比较寄存器 3 (TIM3\_CCR3)

偏移地址: 0x40

复位值: 0x0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	CCR3															
Type	rw															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CCR2															
Type	R3															

Bit	Field	Type	Reset	Description
31:16	CCR3	rw	0x0000	通道 3 捕获/比较高 16 位值 (High Capture/Compare 3value)
15:0	CCR3	rw	0x0000	通道 3 捕获/比较低 16 位值 (Low Capture/Compare 3 value) 详见 CCR1 描述

#### 14.7.16 捕获比较寄存器 4 (TIM2\_CCR4)

偏移地址: 0x40

复位值: 0x0000

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

Bit	Field	Type	Reset	Description
31:16	CCR4	rw	0x0000	通道 4 捕获/比较高 16 位值 (High Capture/Compare 4value)
15:0	CCR4	rw	0x0000	通道 4 捕获/比较低 16 位值 (Low Capture/Compare 4 value) 详见 CCR1 描述

#### 14.7.17 DMA 控制寄存器 (TIM2\_DCR)

偏移地址: 0x48

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.															
	DBL				Res.				DBA							

Type		rw		rw
------	--	----	--	----

Bit	Field	Type	Reset	Description
15:13	Reserved			保留, 始终读为 0。
12:8	DBL	w	0x00	<p>DMA 连续传送长度 (DMA burst length)            这些位定义了 DMA 在连续模式下的访问寄存器的数量            00000: 1 次传输            00001: 2 次传输            00010: 3 次传输            .....            .....            10001: 18 次传输</p>
7:5	Reserved			保留, 始终读为 0。
4:0	DBA	w	0x00	<p>DMA 基地址 (DMA base address)            这些位定义了 DMA 在连续模式下访问 TIM2_DMAR 寄存器的第一个地址。DBA 定义为从 TIM2_CR1 寄存器所在地址开始的偏移值：            00000: TIM2_CR1            00001: TIM2_CR2            00010: TIM2_SMCR            .....</p>

#### 14.7.18 连续模式的 DMA 地址 (TIM2\_DMAR)

偏移地址: 0x4C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	DMAB															
Type	w															

Bit	Field	Type	Reset	Description
-----	-------	------	-------	-------------

15:0	DMAB	w	0x0000	DMA 连续传送寄存器 (DMA register for burst accesses) 对 TIM2_DMAR 寄存器的写操作会导致对以下地址所在寄存器的存取操作: TIM2_CR1 地址 + DBA + DMA 索引, 其中: TIM2_CR1 地址是 TIM2_CR1 寄存器所在的地址;DBA 是 TIM2_DCR 寄存器中定义的基地址; DMA 索引是 DMA 自动控制的偏移量, 它取决于 TIM2_DCR 寄存器中定义的 DBL。
------	------	---	--------	---

#### 14.7.19 TIMERx 选项寄存器 (TIMx\_OR)

偏移地址: 0x50

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.										TI4_RMP	Res.			ETR_RMP	
Type											rw				rw	

Bit	Field	Type	Reset	Description
15:8	Reserved			保留, 始终读为 0。
7:6	TI4_RMP	rw	0x0	Timerx TI4 复用 00: Timerx CH4 GPIO 输入 01: lsi_clk (仅限 TIM5) 10: lse_clk (仅限 TIM5) 11: 保留
5:3	Reserved			保留, 始终读为 0。
2:0	ETR_RMP	rw	0x0	Timerx ETR 复用 00: Timerx ETR GPIO 输入 其他保留

# 15 基本定时器 (TIM6/7)

## 15.1 TIM6/7 简介

TIM6/7 由一个 16 位可实时编程预分频器和一个 16 位的自动重装载计数器组成，可以为用户提供便捷的计数定时功能，计数器时钟由预分频器分频得到。

## 15.2 功能框图

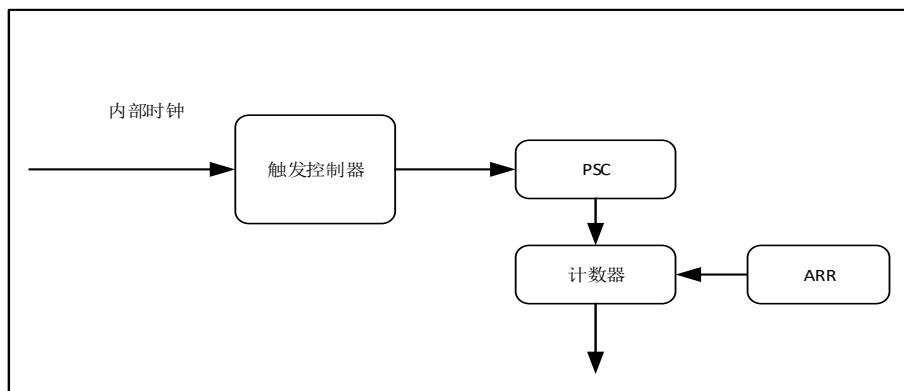


图 15-1 TIM6/7 结构图

上图为 TIM6/7 的结构框图。

## 15.3 主要特征

- 1) 16 位可实时编程预分频器，分频系数：1–65536 可调。
- 2) 16 位自动重装载计数器（计数方向：递增）。
- 3) 产生中断/DMA 请求的事件：更新事件。

注：更新事件：计数器上溢，计数器初始化；

## 15.4 中断

TIM6/7 的中断包括：更新中断，当相应的中断使能位打开，发生相应的事件时，产生相应的中断。

## 15.5 DMA

TIM6/7 能够在发生单个事件时生成多个 DMA 请求。主要目的是在没有软件开销的情况下，多次重新编程 TIM6/7 的一部分寄存器也可以用于按周期读取数个寄存器。

## 15.6 功能描述

### 15.6.1 时钟

#### 15.6.1.1 时钟选择

计数器的时钟源只有一种：内部时钟 (INT\_CK)。

CEN 和 UG 位是事实上的控制位，并且只能被软件修改（除了 UG 位仍被自动清除）。CEN=1，启动计数器。

### 15.6.1.2 时基单元

TIM6/7 的时基单元主要包括：计数器寄存器（TIM6/7\_CNT）、预分频器寄存器（TIM6/7\_PSC）和自动装载寄存器（TIM6/7\_ARR）。

计数器由一个 16 位的计数器和对应的自动装载寄存器组成，可以实现递增计数功能。计数器的时钟由预分频器提供，预分频器由预分频计数器和对应的寄存器组成，分频系数为 1-65536，可以随时写入，在下一次更新事件时生效。

自动预装载寄存器有预装载功能的 16 位影子寄存器，通过设置 TIM6/7\_CR1 寄存器的 APRE 位选择写入 ARR 寄存器的值立即生效或发生更新事件时载入影子寄存器。

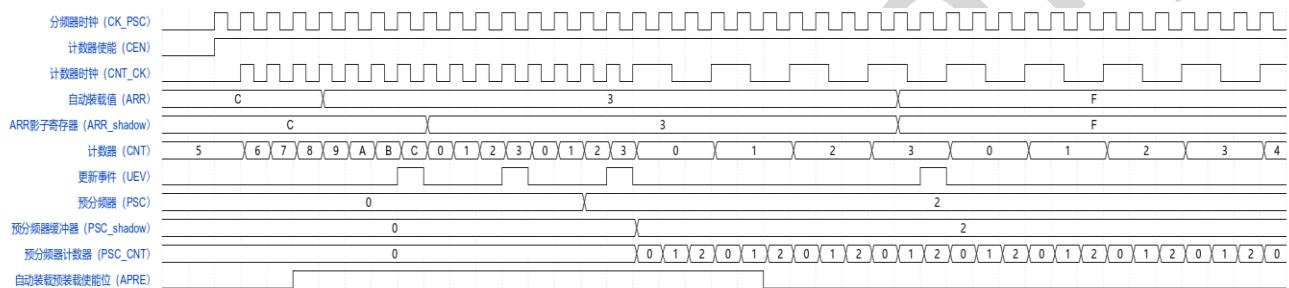


图 15-2 自动预装载

### 15.6.1.3 计数模式--递增计数模式

TIM6/7 的计数模式只有递增计数模式。递增计数模式下，在使能 TIM6/7\_CR1 的 CEN 后计数器由 0 开始递增计数，直至 TIM6/7\_ARR 的值，并产生一个计数器上溢事件（更新事件），更新事件后计数器重新从 0 开始重新递增计数。设置 TIM6/7\_EGR 寄存器的 UG=1，同样可以产生一个更新事件。

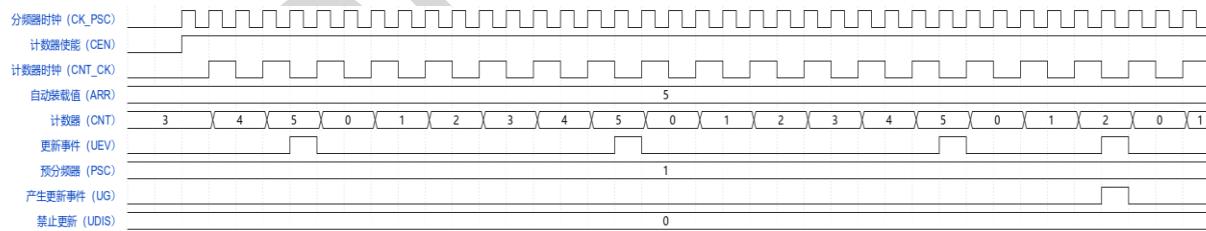


图 15-3 递增计数模式 (UDIS=0)

当配置 TIM6/7\_CR1 寄存器的 UDIS=1，禁止产生更新事件，当计数器发生上溢事件时，不产生更新事件；配置 UG=1，不产生更新事件，计数器和预分频器计数器会被初始化，从零开始递增计数。

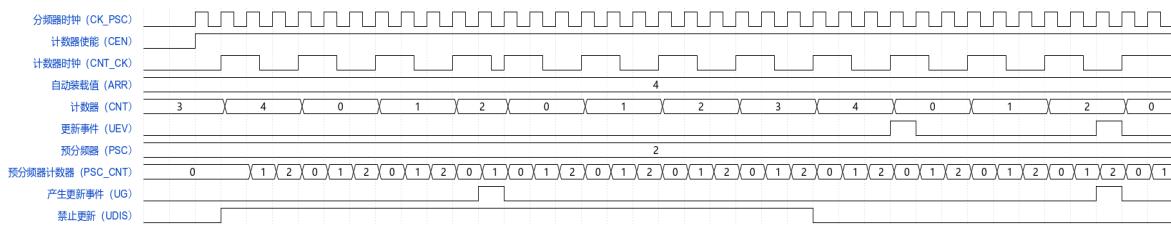


图 15-4 递增计数模式 (UDIS=1)

注：发生更新事件时：

- ARR 寄存器中的值被载入 ARR\_shadow 寄存器中。
- 预分频器的预装载值生效。

### 15.6.2 调试模式

配置 DBG\_CR 寄存器中 DBG\_TIM6/7\_STOP=1, TIM6/7 计数器在 CPU 内核停止工作时（调试设置的断点）停止计数。（详见调试章节）

## 15.7 寄存器描述

表 15-1 TIM6/7 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	TIM6/7_CR1	控制寄存器 1	0x0000
0x0C	TIM6/7_DIER	DMA/中断使能寄存器	0x00000000
0x10	TIM6/7_SR	状态寄存器	0x00000000
0x14	TIM6/7_EGR	事件产生寄存器	0x00000000
0x24	TIM6/7_CNT	计数器	0x0000
0x28	TIM6/7_PSC	预分频率器	0x0000
0x2C	TIM6/7_ARR	自动装载寄存器	0x0000

### 15.7.1 控制寄存器 1 (TIM6/7\_CR1)

偏移地址: 0x0

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								APRE	Res.			OPM	URS	UDIS	CEN
Type									rw				rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15:8	Reserved			保留, 始终读为 0
7	APRE	rw	0x0	自动重装载预装载使能 (Auto-reload preload enable) 0: 关闭 TIM6/7_ARR 寄存器的影子寄存器 1: 使能 TIM6/7_ARR 寄存器的影子寄存器
6:4	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
3	OPM	rw	0x0	<p>单脉冲模式 (One pulse mode)</p> <p>0: 在发生更新事件时, 计数器停止</p> <p>1: 在发生下一次更新事件 (清除 CEN 位) 时, 计时器停止</p>
2	URS	rw	0x0	<p>更新请求源 (Update request source)</p> <p>软件配置该位, 选择更新事件源。</p> <p>0: 以下事件可产生一个更新中断或 DMA 请求:</p> <ul style="list-style-type: none"> <li>- 计数器溢出/下溢</li> <li>- 设置 UG 位</li> <li>- 从模式控制器产生的更新</li> </ul> <p>1: 只有计数器溢出/下溢才产生一个更新中断或 DMA 请求</p>
1	UDIS	rw	0x0	<p>禁止更新 (Update disable)</p> <p>该位用来允许或禁止更新事件的产生</p> <p>0: 允许更新事件 (UEV)。更新事件源:</p> <ul style="list-style-type: none"> <li>- 计数器溢出/下溢</li> <li>- 设置 UG 位为 1</li> <li>- 从模式控制器产生一个更新事件。</li> </ul> <p>1: 禁止更新事件。不产生更新事件, 影子寄存器 (ARR、PSC、CCRx) 保持值不变。如果设置了 EGR_UG 位为 1, 或者从模式控制器接收到硬件复位, 计数器和预分频器被更新为其对应的影子寄存器的值。</p>
0	CEN	rw	0x0	<p>允许计数器 (Counter enable)</p> <p>0: 禁止计数器</p> <p>1: 使能计数器。</p> <p>注: 在软件设置了 CEN 位后, 外部时钟、门控模式和编码器模式才能工作。触发模式可以自动地通过硬件设置 CEN 位。</p>

### 15.7.2 DMA/中断使能寄存器 (TIM6/7\_DIER)

偏移地址: 0x0C

复位值: 0x0000 0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field							UDE	Res.								UIE
Type							rw									rw

Bit	Field	Type	Reset	Description
15:9	Reserved			保留, 始终读为 0
8	UDE	rw	0x0	更新 DMA 请求使能 (Update DMA request enable) 0: 禁止更新 DMA 请求 1: 使能更新 DMA 请求
7:1	Reserved			保留, 始终读为 0
0	UIE	rw	0x0	允许更新中断 (Update interrupt enable) 0: 禁止更新中断 1: 允许更新中断

### 15.7.3 状态寄存器 (TIM6/7\_SR)

偏移地址: 0x10

复位值: 0x0000 0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field																	UIF
Type																	r_w0c

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0
0	UIF	r_w0c	0x0	更新中断标记 (Update interrupt flag) 当产生更新事件时该位由硬件置 1。它由软件清 0。 0: 无更新中断发生 1: 发生更新中断。 当寄存器被更新时该位由硬件置 1: - 若 TIM6/7_CR1 寄存器的 UDIS=0、URS=0, 当 TIM6/7_EGR 寄存器的 UG=1 时产生更新事件。 - 若 TIM6/7_CR1 寄存器的 UDIS=0、URS=0, 当计数器被触发事件初始化时产生更新事件。

### 15.7.4 事件产生寄存器 (TIM6/7\_EGR)

偏移地址: 0x14

复位值: 0x0000 0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.															UG
Type																w

Bit	Field	Type	Reset	Description
31:1	Reserved			保留, 始终读为 0
0	UG	w	0x0	产生更新事件 (Update generation) 0: 无动作 1: 重置计数器值, 并产生一个更新事件。由硬件自动清 0。

### 15.7.5 计数器 (TIM6/7\_CNT)

偏移地址: 0x24

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CNT															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x0000	计数器的值 (Count value)

### 15.7.6 预分频器 (TIM6/7\_PSC)

偏移地址: 0x28

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	PSC															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	PSC	rw	0x0000	预分频器的值 (Prescaler value) 计数器的时钟频率 ( $f_{CK\_cnt}$ ) = $f_{CK\_PSC} / (PSC + 1)$ 。 当发生更新事件时, PSC 的值装入当前预分频寄存器。

### 15.7.7 自动预装载寄存器 (TIM6/7\_ARR)

偏移地址: 0x2C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ARR															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	ARR	rw	0x0000	<p>自动预装载的值 (Auto-reload value) 这些位定义了计数器的自动重载值。当自动重装载的值为 0 时，计数器不工作。</p>

## 16 独立看门狗 (IWDG)

### 16.1 IWDG 简介

独立看门狗的设计初衷是为了检测和解决由软件错误所引起的故障，它的原理可简述为：当独立看门狗（IWDG）计数器不断递减到达给定数值时，产生一个系统复位信号使系统复位，从而提高系统整体安全性能。

独立看门狗适合应用于那些需要看门狗作为一个处于主程序之外，能够完全独立工作，并且对时间精度要求低的场合。

独立看门狗是由内部低速的时钟（LSI）驱动的，保证当主时钟发生故障的时候，独立看门狗依旧可以继续工作。

## 16.2 功能框图

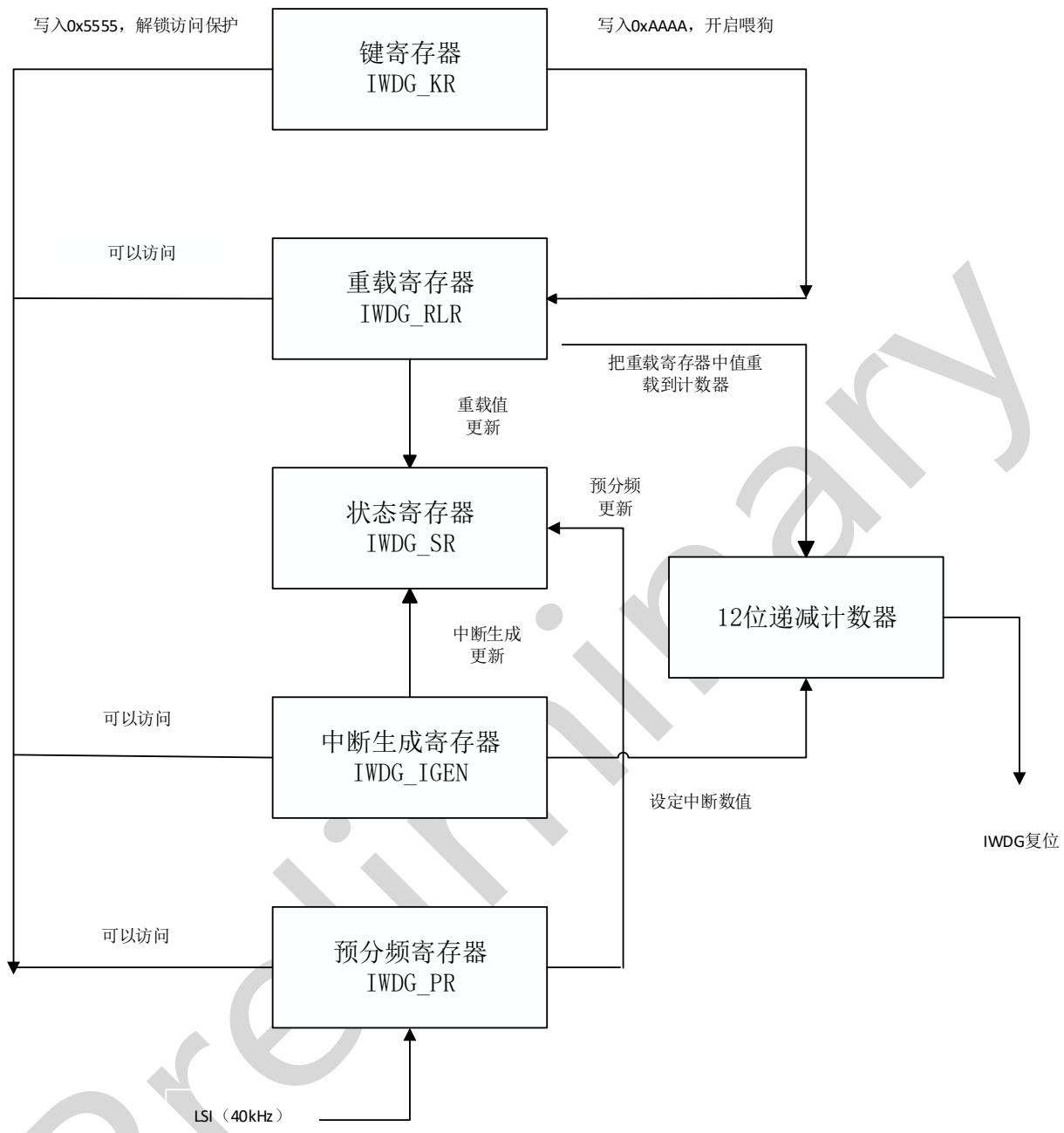


图 16-1 IWDG 结构框图

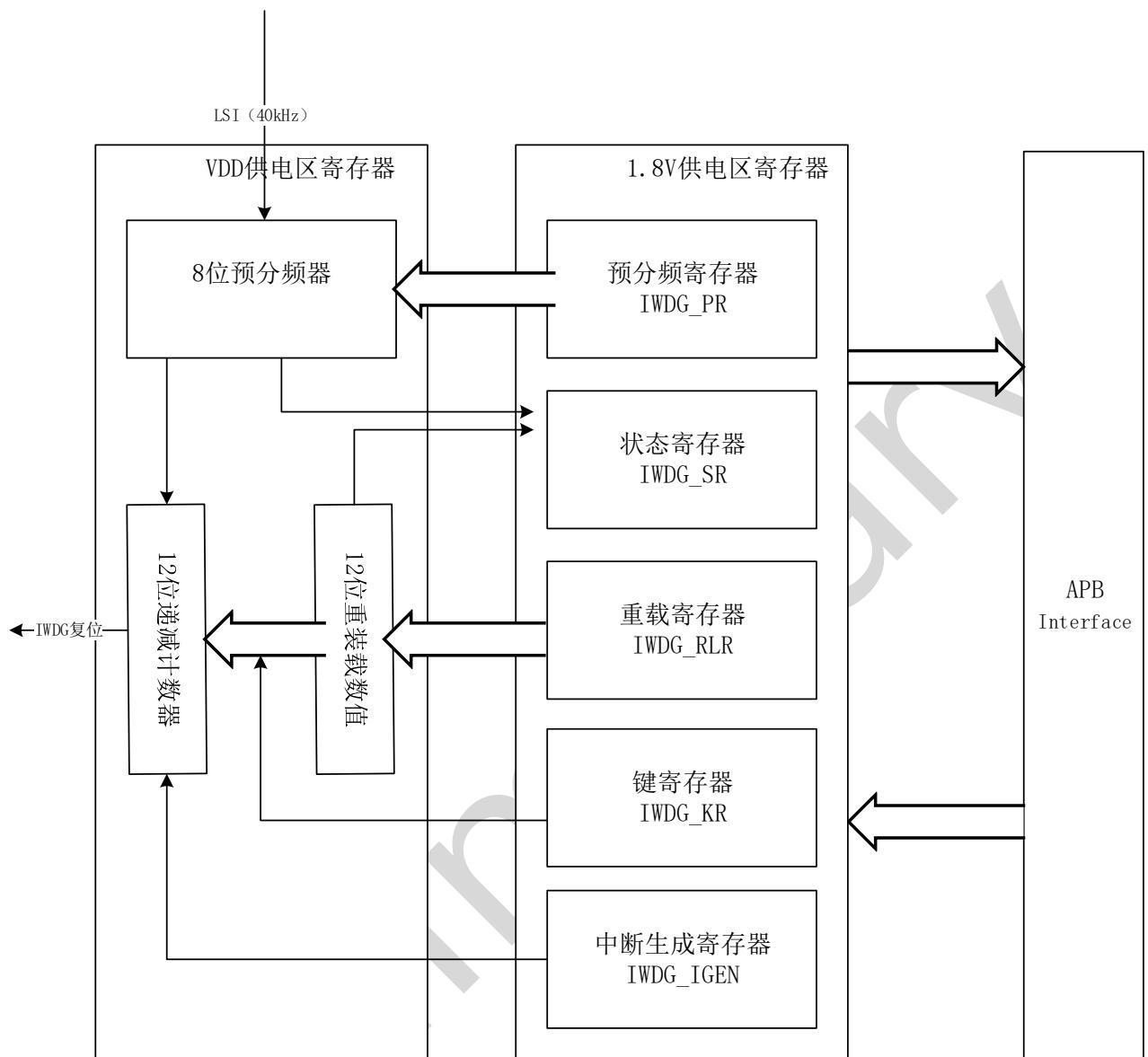


图 16-2 IWDG 流程框图

### 16.3 主要特征

- 1) 芯片默认为软件看门狗模式；
- 2) 通过闪存烧写复位选项字节寄存器中的 WDG\_SW 位可以启动硬件看门狗。硬件看门狗启动后在系统复位上电后自动启动，内部计数器开始递减；
- 3) LSI 可在停止和待机模式下继续进行工作；
- 4) 看门狗内部是自由运行的递减计数器，当计数到达 0x0000 产生一个系统复位或者中断信号。

### 16.4 功能描述

- 1) 在键值寄存器 (IWDG\_KR) 中写入 0xFFFF，开启独立看门狗。与此同时，计数器开始

从其复位值 0xFFFF 开始递减，当递减到达 0x0000 时会产生一个系统复位信号 (IWDG\_RESET) 或者产生中断信号使系统复位或中断。

- 2) 任何时候写入 0xAAAA 到 IWDG\_KR，就会把重载寄存器 (IWDG\_RLR) 中的值重新加载到计数器中（通常说的喂狗），从而避免复位信号或中断信号的产生。
- 3) 比较/输出如果程序异常，无法正常喂狗，就会产生复位信号或中断信号，系统复位或系统中断。
- 4) IWDG\_PR, IWDG\_RLR, IWDG\_IEN 寄存器具有访问保护功能。只有在键值寄存器 (IWDG\_KR) 写入 0x5555，才可以修改以上被保护的寄存器的值。当以其他的值写入键值寄存器，会打乱操作顺序，寄存器依旧处于保护状态。当进行重载操作时，也会处于保护状态。
- 5) 独立看门狗和低功耗模式，因为独立看门狗的时钟由 LSI 提供，因此可以工作在停止和待机模式下。
- 6) 独立看门狗可以在低功耗模式下正常计数，它的复位能够使系统退出 Standby 模式。
- 7) 在低功耗模式下，进入 Stop 模式后，可以选择关闭 LSI 时钟，从而关闭硬件看门狗。

## 16.5 独立看门狗超时时间

表 16-1 IWDG 超时时间 (40kHz 的输入时钟 (LSI))

预分频系数	PR[2:0] 位	最短时间 (ms)	
		RL[11:0]=0x000	最长时间(ms)
/4	0	0.1	409.6
/8	1	0.2	819.2
/16	2	0.4	1638.4
/32	3	0.8	3276.8
/64	4	1.6	6553.6
/128	5	3.2	13107.2
/256	(6 或 7)	6.4	26214.4

超出 (溢出) 时间计算：

$$T_{out} = ((4 \times 2^P) \times RLR) / 40$$

其中：Tout 的单位为毫秒。

时钟频率 LSI=40K，一个看门狗时钟周期就是最短超时时间。最长超时时间 = (IWDG\_RLR 寄存器最大值) X 看门狗时钟周期。

## 16.6 寄存器描述

表 16-2 IWDG 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	IWDG_KR	键寄存器	0x00000000
0x04	IWDG_PR	预分频寄存器	0x00000000
0x08	IWDG_RLR	重装载寄存器	0x00000FFF
0x0C	IWDG_SR	状态寄存器	0x00000000
0x10	IWDG_CR	控制寄存器	0x00000000
0x14	IWDG_IGEN	中断生成寄存器	0x00000FFF
0x18	IWDG_CNT	计数寄存器	0x00000000
0x1c	IWDG_PS	分频计数寄存器	0x00000001

### 16.6.1 键寄存器 (IWDG\_KR)

偏移地址: 0x00

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	KEY															
Type	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Bit	Field	Type	Reset	Description
31: 16	Reserved			保留, 始终读为 0
15: 0	KEY	w	0x0000	<p>键值 (只写寄存器) (Key value)            软件每隔一段时间, 写进 0xFFFF 进行喂狗操作, 否则当计数器递减到 0x0000 时, 会产生一个复位信号, 使系统复位。            软件写入 0xFFFF 表示解除保护, 可以访问其他配置寄存器 (IWDG_PR、IWDG_RLR、IWDG_CR, IWDG_IGEN)</p>

Bit	Field	Type	Reset	Description
				软件写入 0xCCCC，开启看门狗。

## 16.6.2 预分频寄存器 (IWDG\_PR)

偏移地址: 0x04

复位值: 0x0000 0000 (在待机模式复位)

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved															PR
Type																rw

Bit	Field	Type	Reset	Description								
31:3	Reserved			始终读为 0								
2: 0	PR	rw	0x00	<p>预分频因子 (Prescaler divider)</p> <p>通过设置这些位来选择 LSI 时钟的预分频因子</p> <p>要改变预分频因子，需要先解除保护（向 IWDG_KE 中写入 0x5555）才能写入，当预分频因子被更新完成后，PUV 寄存器位会变为 0，此时读出数据才是有效的。</p> <table> <tbody> <tr> <td>000: 预分频因子 = 4</td> <td>100: 预分频因子 = 64</td> </tr> <tr> <td>001: 预分频因子 = 8</td> <td>101: 预分频因子 = 128</td> </tr> <tr> <td>010: 预分频因子 = 16</td> <td>110: 预分频因子 = 256</td> </tr> <tr> <td>011: 预分频因子 = 32</td> <td>111: 预分频因子 = 256</td> </tr> </tbody> </table>	000: 预分频因子 = 4	100: 预分频因子 = 64	001: 预分频因子 = 8	101: 预分频因子 = 128	010: 预分频因子 = 16	110: 预分频因子 = 256	011: 预分频因子 = 32	111: 预分频因子 = 256
000: 预分频因子 = 4	100: 预分频因子 = 64											
001: 预分频因子 = 8	101: 预分频因子 = 128											
010: 预分频因子 = 16	110: 预分频因子 = 256											
011: 预分频因子 = 32	111: 预分频因子 = 256											

## 16.6.3 重装载寄存器 (IWDG\_RLR)

偏移地址: 0x08

复位值: 0x0000 0FFF (在待机模式复位)

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved				RL											
Type																

Bit	Field	Type	Reset	Description
31:12	Reserved			始终读为 0
11: 0	RL	rw	0xFFFF	重装载值 (Watchdog counter reload value) 配置看门狗计数器的重载值，每当喂狗（向 IWDG_KR 寄存器中写入 0xAAAA）时，会把此位数值更新到计时器中，然后从此值开始递减。更改重载值需要解除保护（向 IWDG_KR 寄存器中写入 0x5555），当重载值更新完毕后，RUV 寄存器位会清 0，也就在此时读出值才是有效的。看门狗的超时周期可以通过重装载值和预分频值来计算。

#### 16.6.4 状态寄存器 (IWDG\_SR)

偏移地址: 0x0C

复位值: 0x0000 0000 (待机模式不复位)

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved												UPDATE	IVU	RVU	PVU
Type																

Bit	Field	Type	Reset	Description
31:4	Reserved			保留，始终读为 0

Bit	Field	Type	Reset	Description
3	UPDATE	r	0x00	看门狗重装载值更新标志 当 IWDG_KR 寄存器中写入 0xAAAA 时, update 置位, 当看门狗计数器被更新, 重装载值写入到计数器中, update 自动清零。
2	IVU	r	0x00	看门狗中断生成值更新 (Watchdog Interrupt Generate value update) 此位由硬件置'1' 用来指示中断生成值的更新正在进行中。 当在 VDD 域中的中断生成值更新结束后, 此位由硬件清'0' (最多需要 5 个 40KHz 的振荡器周期) 中断生成值只有在 IVU 位被清'0' 后才可更新。
1	RVU	r	0x00	看门狗计数器重装载值更新 (Watchdog counter reload value update) 重装载值的更新正在进行中, 此位置 1。 当重装载更新结束后, 此位清'0' (最多需要 5 个 40KHz 的振荡器周期) 重装载值只有在 RVU 位被清'0' 后才可更新。
0	PVU	r	0x00	看门狗预分频更新 (Watchdog prescaler value update) 预分频值的更新正在进行中时, 此位置 1。 当预分频值更新结束后, 此位由清'0' (最多需要 5 个 40KHz 的振荡器周期) 预分频值只有在 PVU 位被清'0' 后才可更新。

注: 如果在应用程序中使用多个重装载值、预分频值或中断生成值, 先解除寄存器保护 (向 IWDG\_KR 中写入 0x5555), 然后配置 IWDG\_PR IWDG\_RLR, IWDG\_IGEN 寄存器, 等待对应的状态寄存器位 (PVU, RVU, IVU) 清零, 表示已经配置好重装载值, 预分频值, 和中断生成值, 接下来就可以执行喂狗操作, 或者等待计数器自动递减产生复位或者中断信号。

## 16.6.5 控制寄存器 (IWDG\_CR)

偏移地址: 0x10

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved															
Type																

Bit	Field	Type	Reset	Description
31:2	Reserved			保留, 始终读为 0
1	IRQ_CLR	rw	0x00	IWDG 中断清除 1: 写 1 清除中断 0: 无效操作, 中断标志位依旧挂起
0	IRQ_SEL	rw	0x00	IWDG 溢出操作选择 1: 溢出后产生中断 0: 溢出后产生复位

## 16.6.6 中断生成寄存器 (IWDG\_IGEN)

偏移地址: 0x14

复位值: 0x0000 0FFF

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved				IGEN											
Type					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:12	Reserved			始终读为 0
11: 0	IGEN	rw	0xFFFF	IWDG 中断生成值 (Watchdog Interrupt Generate value) 用于定义看门狗中断生成值, 每当计数器值递减等于该值时, 会产生中断。要改变该位数值, 需要先解除保护。 当改变数值被更新完成后 IWDG_SR 寄存器中的 IVU 位为 清 0 此时, 读出数据才是有效的。

## 16.6.7 计数寄存器 (IWDG\_CNT)

偏移地址: 0x18

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved				IWDG_CNT											
Type					r	r	r	r	r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31: 12	Reserved			保留, 始终读为 0
11: 0	IWDG_CNT	r	0x00	IWDG 计数器 counter 的值

## 16.6.8 分频计数寄存器 (IWDG\_PS)

偏移地址: 0x1C

复位值: 0x0000 0001

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved								IWDG_PS							
Type									r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31:8	Reserved			保留，始终读为 0。
7: 0	IWDG_PS	r	0x01	IWDG 时钟分频计数器的值

# 17 窗口看门狗 (WWDG)

## 17.1 WWDG 简介

窗口看门狗就是其喂狗时间是有上下限的范围的（窗口）看门狗，主要用于检测由外部干扰和不可预测的逻辑条件导致程序跑飞了所引起的软件问题。

对于独立看门狗，程序可以在其产生复位前的任意时刻刷新看门狗，但是有时候程序跑飞了之后又跑回了正常的地方，或者跑飞的程序正好刷新看门狗操作，造成独立看门狗失效。在这种情况下就需要窗口看门狗。

根据程序正常执行的时间设置 刷新看门狗的一个时间窗口，保证不会提前刷新看门狗也不会滞后刷新看门狗，这样就可以检测出程序有没有按照正常的路径运行或者是非正常的跳过某些程序段的情况。

## 17.2 功能框图

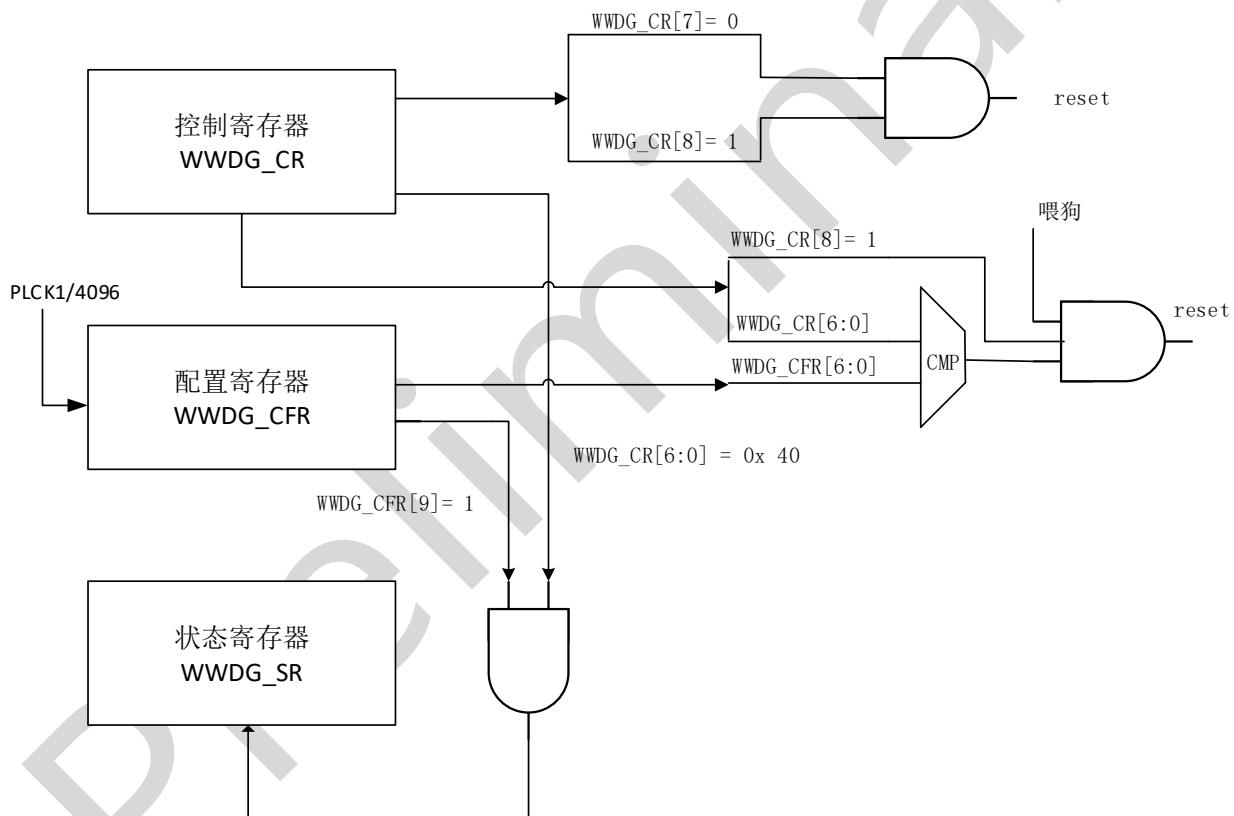


图 17-1 WWDG 结构框图

## 17.3 主要特性

- 1) 可编程的自由运行的递减计数器；
- 2) 窗口外喂狗产生复位

上限：当计数器的值大于配置寄存器（WWDG\_CFR）设定的值时喂狗，将会产生复位；

下限：当计数器的值从下限固定值 0x40 变成 0x3F 时，产生复位；

### 3) 中断防止复位:

如果启动了窗口看门狗并且允许中断, 当递减计数器递减到 0x40 时, 将产生提前唤醒中断 (EWI), 可以在中断处理函数中, 向 **WWDG\_CR** 重载计数器的值来达到喂狗的目的, 从而防止复位。

## 17.4 功能描述

- 1) 当控制寄存器 (**WWDG\_CR**) 中的第 8 位 WDGA 被置 1 时, 启动窗口看门狗, 并且当计数器的值从下限固定值 0x40 翻转到 0x3F (**WWDG\_CR** 中的第 6 位清零), 将会产生一个复位。或者当计数器的值大于配置寄存器 (**WWDG\_CFGR**) 设定的值时喂狗, 将会产生复位。
- 2) 窗口看门狗和独立看门狗一样, 都是在应用程序在正常运行中定时喂狗操作, 来防止 MCU 发生复位, 但区别在于: 窗口看门狗必须在计数器值小于窗口寄存器值的时候操作, 也只有在这种情况下, 喂狗才不会产生复位。其中, 写入控制寄存器 (**WWDG\_CR**) 的值必须在 0xFF 和 0xC0 之间。
- 3) 窗口看门狗在系统复位后, 窗口看门狗不会自动运行, 需要设置控制寄存器 (**WWDG\_CR**) 中 WDGA 位开启窗口看门狗, 因为 WDGA 位只能由硬件复位后清零, 所以一旦软件置位后, 除非复位, 窗口看门狗会一直开启。当窗口看门狗被打开后, 需要配置控制寄存器 (**WWDG\_CR**) 的第 7 位置“1”, 以防止立即产生一个复位。**WWDG\_CR[5:0]** 代表看门狗复位前的计数, 由于预分频值是未知的, 所以复位前的延时时间在一个最小值和最大值之间变化。
- 4) 利用提前唤醒中断 (EWI) 可以避免复位并且重载计数器, 设置配置寄存器 (**WWDG\_CFG**) 中的 EWI 位来开启中断, 当计数器数值到达 0x40 时, 产生中断, 可以在中断处理函数中向 **WWDG\_CR** 重写计数器的值来达到喂狗目的从而防止复位。在状态寄存器 (**WWDG\_SR**) 中写“0”来清除中断。

## 17.5 窗口看门狗超时时间

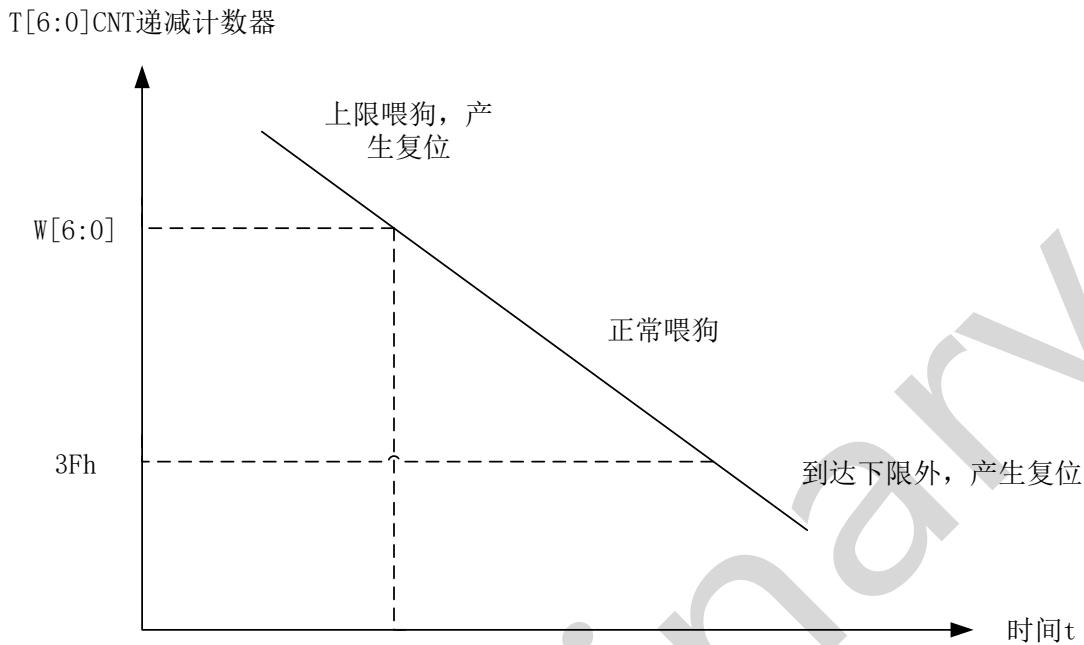


图 17-2 WWDG 超时时间坐标图

表 17-1 超时时间

时基 (WDGTB)	最短时间	最长时间
0	56.5us	3.64ms
1	113.5us	7.28ms
2	227.5us	14.56ms
3	455us	29.12ms

超时计算公式：

$$T = T_{\text{pclk1}} * 4096 * 2^{\text{WDGTB}} * (T[5:0] + 1)$$

WDGTB：分频器的时基

$T_{\text{pclk1}}$ ：APB1 的时钟间隔

## 17.6 寄存器描述

表 17-2 WWDG 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	WWDG_CR	控制寄存器	0x0000007F
0x04	WWDG_CFGR	配置寄存器	0x0000007F
0x08	WWDG_SR	状态寄存器	0x00000000

### 17.6.1 控制寄存器 (WWDG\_CR)

偏移地址: 0x00

复位值: 0x0000 007F

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Reserved								WDGA	T[6:0]							
Type									rw								

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0
7	WDGA	rw	0x0	<p>WDGA: 激活位 (Activation bit) 此位由软件置‘1’，但仅能由硬件在复位后清‘0’。当 WDGA = 1 时，看门狗可以产生复位。</p> <p>0: 禁止看门狗 1: 启动看门狗</p>
6:0	T[6:0]	rw	0x7F	<p>T[6: 0]: 7 位计数器 (MSB 至 LSB) (7 - bit counter) 这些位用来存储看门狗的计数值。每 (4096x2WDGTR) 个 PCLK1 周期减 1。当计数值从 40h 变为 3Fh 时 (T6 变成 0)，产生看门狗复位。</p>

### 17.6.2 配置寄存器 (WWDG\_CFGR)

偏移地址: 0x04

复位值: 0x0000 007F

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved						EWI	WDGTB		W[6:0]						
Type							rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:10	Reserved			保留, 始终读为 0
9	EWI	rw	0x0	EWI: 提前唤醒中断 (Early wakeup interrupt) 当计数器值达到 40h, 并且此位置 1 时, 产生中断。 此中断只能由硬件在复位后清除。
8: 7	WDGTB	rw	0x0	WDGTB[1: 0]: 时基 (Timer base) 预分频器的时基可根据如下修改: 00: CK 计时器时钟 (PCLK1 除以 4096) 除以 1 01: CK 计时器时钟 (PCLK1 除以 4096) 除以 2 10: CK 计时器时钟 (PCLK1 除以 4096) 除以 4 11: CK 计时器时钟 (PCLK1 除以 4096) 除以 8
6: 0	W[6:0]	rw	0x7F	W[6: 0]: 7 位窗口值 (7-bit window value) 窗口看门狗的上限窗口值。

### 17.6.3 状态寄存器 (WWDG\_SR)

偏移地址: 0x08

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved															
Type																

Bit	Field	Type	Reset	Description
31: 1	Reserved			保留, 始终读为 0
0	EWIF	rc_w0	0x0	EWIF: 提前唤醒中断标志 (Early wakeup interrupt flag) 当计数器值达到 40h 时, 此位由硬件置'1'。它必须通过软件写'0'来清除。对此位写'1'无效。若中断未被使能, 此位也会被置'1'。

# 18 实时时钟器 (RTC)

## 18.1 RTC 简介

RTC 模块内部包含一组连续计数的计数器，它作为一个独立的定时器，通过软件配置，可以实时显示时钟。同时可以通过修改计数器的值重置时钟。

当系统从复位或待机模式唤醒后，RTC 会保持设置时间不变（时钟配置模块（RCC\_BDCR 寄存器）与 RTC 模块处于后备区域）。

避免对后备区（BKP）的意外写操作。当系统复位后，禁止访问后备寄存器和 RTC。通过执行下面两步，可以对后备寄存器和 RTC 访问。

- 1) 配置 RCC\_APB1ENR 寄存器中的 PWREN 位为 1 打开电源时钟。

配置 PWR\_CR 寄存器的 DBP 位为 1 打开对后备寄存器和 RTC 的访问权限。

## 18.2 功能框图

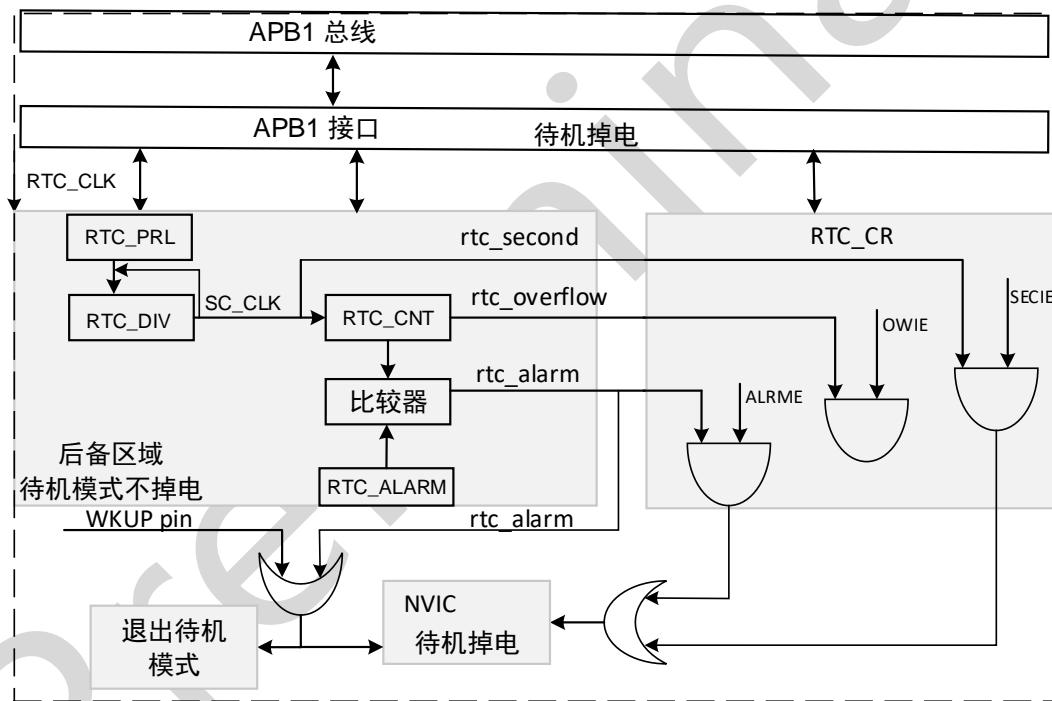


图 18-1 RTC 结构框图

## 18.3 主要特征

- 1) 0 到  $2^{20}$  的可编程控制预分频系数；
- 2) 可编程的 32 位计数器；
- 3) 包含 2 个独立的时钟：分别用于 RTC 时钟与 APB1 接口的 PCLK1，其中作为 RTC 时钟的频率必须小于 PCLK1 时钟频率的四分之一以上；
- 4) RTC 的时钟源有 3 种：

- e) HSE 时钟的 128 分频;
  - f) 外部低速 LSE 振荡器时钟;
  - g) 内部低速 LSI 振荡器时钟;
- 5) 3 个可屏蔽中断:
- a) 可编程控制产生的闹钟中断;
  - b) 可编程控制产生的秒中断（最长时间 1 秒）;
  - c) 计数溢出中断，内部计数器计数到 0 时产生;
- 6) 2 个独立的复位系统:
- a) 预分频器、闹钟、计数器和分频器（RTC 核心部分）只能通过后备域复位;
  - b) APB1 接口部分通过系统复位;

## 18.4 功能描述

### 18.4.1 概述

RTC 主要包含两个部分，APB1 接口部分与可编程计数部分。

APB1 接口部分用于连接 APB 总线。该部分包含一组 16 位寄存器，在 APB1 总线时钟的驱动下，实现对寄存器的读写操作。

可编程预分频器，最大可编程时间基准 TR\_CLK 为 1 秒，如果同时打开 RTC\_CR 控制寄存器中设置相应的中断使能位，会产生中断。

内部 32 位的可编程计数器，可用于配置系统时间。系统时间按 TR\_CLK 周期累加，当设置了 RTC\_CR 控制寄存器中相应的允许位，并且系统时钟的值与存储在 RTC\_ALR 寄存器中的可编程时间比较匹配时，会产生一个闹钟中断。

### 18.4.2 模块复位

系统复位或电源复位的方式可以对所有的寄存器进行复位（RTC\_PRL、RTC\_ALR、RTC\_CNT 和 RTC\_DIV 寄存器，只能通过备份域复位信号复位）。

### 18.4.3 寄存器读取

RTC 作为独立于 RTC APB1 接口的部分，软件可以通过 APB1 接口访问 RTC 的寄存器组的值，包括分频计数以及闹钟的值。但是，可读寄存器的值只有在与 RTC APB1 时钟进行重新同步的 RTC 时钟的上升沿被更新。RTC 标志位也是如此的。

因此，如果之前关闭了 APB1 接口，重新打开后立即进行读操作，那么第一次内部寄存器更新之前，RTC 寄存器数值可能被破坏了，通过 APB1 接口读出错误的数据，通常为 0。

可能造成这种情况的几种情形如下：

- 1) 从待机模式唤醒系统并立即进行读操作；
- 2) 从停机模式唤醒系统并立即进行读操作；
- 3) 发生系统复位或电源复位并立即进行读操作；

Preliminary

当发生以上情况时，即使有复位、无时钟或断点，APB1 接口被禁止，RTC 核仍保持运行状态。

因此，如果 RTC 的 APB1 接口处于被禁止，读取 RTC 寄存器时，则软件必须首先等待硬件置'1' RTC\_CRL 寄存器中的同步标志位（RSF），读操作才能继续。

注：WFI 和 WFE 等低功耗模式不影响 APB1 接口。

#### 18.4.4 寄存器配置

配置 RTC\_CRL 寄存器中的 CNF 位等于 1，RTC 进入配置模式后，开启对 RTC\_PRL、RTC\_CNT、RTC\_ALR 寄存器的写操作。只有当前一次的写操作结束后才能开始下一次对寄存器的写操作。查询 RTC\_CR 寄存器中的 RTOFF 状态位，来判断 RTC 寄存器是否处于更新中。只有当 RTOFF 状态位是'1'时，才能写 RTC 寄存器。

具体配置过程如下：

- 1) 查询 RTOFF 位，等待 RTOFF 的值变为'1'；
- 2) 配置 CNF 位为'1'，RTC 进入配置模式；
- 3) 对 RTC 寄存器进行写操作；
- 4) 配置 CNF 位为 0，RTC 退出配置模式；
- 5) 查询 RTOFF，当 RTOFF 位等于 1 时，写操作完成；

注：只有当 CNF 标志位被清除时，才能进行写操作，该过程至少需要 3 个时钟周期；

#### 18.4.5 标志位产生

RTC 核心的计数过程中，改变 RTC 计数器所设置 RTC 秒标志（SECF）。

- 1) 当满足 RTC\_CNT 计数值等于 RTC\_ALR 值，并且 RTC\_PR 溢出重新从 0 开始计数时，RTC\_ALARM 标志位置 1；
- 2) 当满足 RTC\_CNT 计数值等于 RTC\_MSR 值，并且 RTC\_PR 溢出重新从 0 开始计数时，RTC\_ALARM 标志位置 1；
- 3) 当满足 RTC\_CNT 计数值等于 RTC\_ALR 值，并且 RTC\_PR 溢出，RTC\_PR 计数到 MSR 时，RTC\_ALARM 标志位置 1；

当计数器到达 0x0000 之前的最后一个 RTC 时钟周期时，RTC 溢出标志（OWF）置 1。

RTC\_CNT 开始计数后，在计数到闹钟寄存器值（RTC\_ALR+1）之前的时钟周期中，配置 RTC\_Alarm 与闹钟标志 ALRF，写 RTC 闹钟只能通过如下的两种方式与秒标志同步：

- 1) 进入 RTC 闹钟中断处理程序，修改 RTC 闹钟或 RTC 计数器；
- 2) RTC 控制寄存器中的 SECF 位置 1 时，修改 RTC 闹钟或 RTC 计数器；

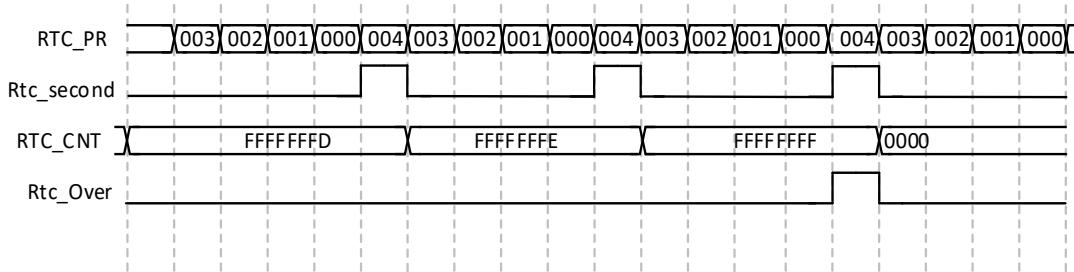


图 18-2 RTC 秒和闹钟波形图示例，PR = 0004，ALARM = 002

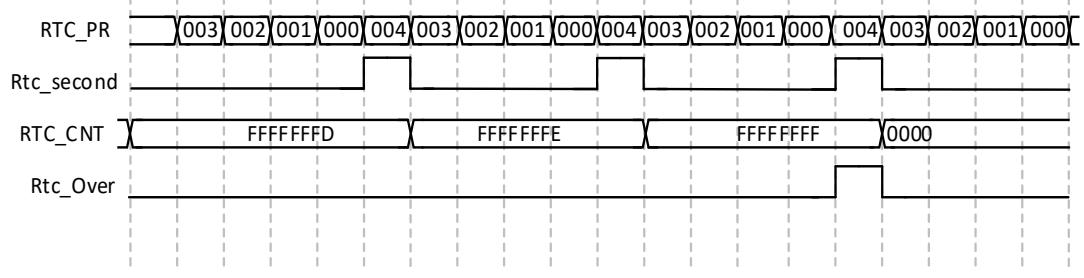


图 18-3 RTC 溢出波形图示例，PR = 0004

#### 18.4.6 RTC 闹钟描述

RTC 闹钟分为两部分：

毫秒计时闹钟：此时需 RTC\_ALR 寄存器为 0，通过配置 RTC\_MSR 来定时毫秒时间，当定时时间到达，闹钟标志位拉起。

秒计时闹钟：此时需 RTC\_MSR 寄存器为 0，通过配置 RTC\_ALR 来定时秒钟时间，当定时时间到达，闹钟标志位拉起。

可同时使用秒和毫秒闹钟用于定时非整秒定时。

RTC 闹钟循环：使用闹钟循环 RTC\_CRL 的 ALPEN 配置闹钟单次/循环发生。

#### 18.4.7 RTC 外部中断事件输出

配置 RTC\_CRL 寄存器中的 ALRF 位为 1，同时打开外部中断线 17，则允许产生 RTC 闹钟中断；配置外部中断线 17 为事件模式，会产生一个事件脉冲信号，不会进入中断。

配置 RTC\_CRH 寄存器中的 ALRIE 与 RTC\_CRL 寄存器中的 ALRF 位为 1，允许产生 RTC 全局中断。同时打开外部中断线 17，则允许产生 RTC 全局中断和 RTC 闹钟中断。

### 18.5 寄存器描述

表 18-1 RTC 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	RTC_CRH	RTC 控制寄存器高位	0x00000000
0x04	RTC_CRL	RTC 控制寄存器低位	0x00000020
0x08	RTC_PRLH	RTC 预分频装载寄存器高位	0x00000000
0x0C	RTC_PRLL	RTC 预分频装载寄存器低位	0x00000000
0x10	RTC_DIVH	RTC 预分频器分频因子寄存器高位	0x00000000
0x14	RTC_DIVL	RTC 预分频器分频因子寄存器低位	0x00000000
0x18	RTC_CNTH	RTC 计数器寄存器高位	0x00000000

Offset	Acronym	Register Name	Reset
0x1C	RTC_CNTL	RTC 计数器寄存器低位	0x00000000
0x20	RTC_ALRH	RTC 闹钟寄存器高位	0x0000FFFF
0x24	RTC_ALRL	RTC 闹钟寄存器低位	0x0000FFFF
0x28	RTC_MSRH	RTC 毫秒寄存器高位	0x00000000
0x2C	RTC_MSRL	RTC 毫秒寄存器低位	0x00000000

### 18.5.1 控制寄存器高位 (RTC\_CRH)

偏移地址: 0x0

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Res.														OWIE	ALRIE	SECIE
Type															rw	rw	rw

Bit	Field	Type	Reset	Description
15:3	Reserved		0x0	保留, 始终读为 0
2	OWIE	rw	0x0	溢出中断使能位 (Overflow interrupt enable) 0: 关闭溢出中断使能 1: 打开溢出中断使能
1	ALRIE	rw	0x0	闹钟中断使能位 (Alarm interrupt enable) 0: 关闭闹钟中断使能 1: 打开闹钟中断使能
0	SECIE	rw	0x0	秒中断使能位 (Second interrupt enable) 0: 关闭秒中断使能 1: 打开秒中断使能

### 18.5.2 控制寄存器低位(RTC\_CRL)

偏移地址: 0x04

复位值: 0x0020

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								ALPEN	RTOFF	CNF	RSF	OWF	ALRF	SECF	
Type									rw	r	rw	rc_w0	rc_w0	rc_w0	rc_w0	

Bit	Field	Type	Reset	Description
15:7	Reserved		0x0	保留, 始终读为 0
6	ALPEN	rw	0x0	<p>RTC 闹钟循环时能 ( RTC Alarm Loop Enable)</p> <p>0: 单次发生闹钟事件 1: 循环发生闹钟事件</p>

Bit	Field	Type	Reset	Description
5	RTOFF	r	0x0	<p>RTC 操作状态 (RTC operation OFF)</p> <p>指示对寄存器最后一次操作是否完成。若此位为'0', 则表示无法对任何的 RTC 寄存器进行读写操作。此位只读。</p> <p>0: 对 RTC 寄存器的写操作仍在进行 1: 完成对 RTC 寄存器的写操作</p>
4	CNF	rw	0x0	<p>CNF: 配置标志 (Configuration flag)</p> <p>只能通过软件置'1'以进入配置模式, 之后可以向 RTC_CNTL/H、RTC_ALRL/H 或 RTC_PRLL/H 寄存器写入数据。只有当此位在被置'1'并重新由软件清'0'后, 才会执行写操作。</p> <p>0: 退出配置模式, 更新 RTC 寄存器 1: 进入配置模式</p>
3	RSF	rc_w0	0x0	<p>寄存器同步标志 (Registers synchronized flag)</p> <p>通过软件更新或清'0' RTC_CNT 与 RTC_DIV 寄存器时, 硬件置'1'该位。当发生 APB1 复位或关闭 APB1 时钟后, 必须由软件清'0'此位</p> <p>用户端必须等待这位被硬件置'1', 才能进行读操作。确保已经同步 RTC_CNT、RTCALR 或 RTC_PRL 寄存器的值。</p> <p>0: 寄存器没有同步 1: 寄存器已经同步</p>
2	OWF	rc_w0	0x0	<p>溢出标志 (Overflow flag)</p> <p>当可编程计数器溢出时, 硬件置'1'</p> <p>同时配置 RTC_CRH 寄存器中的 OWIE 位为 1, 产生中断。</p>

				软件清除该位，硬件写 1 无效 0：无溢出 1：32 位可编程计数器溢出
1	ALRF	rc_w0	0x0	闹钟标志 (Alarm flag) 可编程计数器等于 RTC_ALR 寄存器设定值，硬件置'1' 同时配置 RTC_CRH 寄存器中的 ALRIE=1，产生中断 软件清除该位，硬件写 1 无效 0：无闹钟产生 1：有闹钟产生
0	SECF	rc_w0	0x0	秒标志 (Second flag) 预分频器溢出时，硬件置'1'该位，RTC 计数器加 1。 该标志为可编程的 RTC 计数器提供周期信号（常用 1 秒） 通过配置 RTC_CTH 寄存器中的 SECIE = 1，可以产生中断。 软件清除该位，硬件写 1 无效 0：没有秒标志 1：产生秒标志

注：

- 1) 当写操作状态位 RTOFF=0 时，上一次写操作没有完成，不能对 RTC\_CR 寄存器执行写操作；
- 2) 复位时禁止所有中断，无任何挂起的中断请求，可以写 RTC 寄存器；
- 3) 硬件控制 OWF、ALRF、SECF 和 RSF 位，软件清零；
- 4) 若关闭 APB1 时钟，OWF、ALRF、SECF 和 RSF 位值保持不变；
- 5) 相应标志位保持挂起状态，只有当适当的 RTC\_CR 请求位被软件复位时，表示中断已经响应；

### 18.5.3 预分频装载寄存器高位(RTC\_PRLH)

偏移地址：0x08

复位值：0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	PRL															
Type	w															

Bit	Field	Type	Reset	Description
15:4	Reserved		0x0	保留，始终读为 0
3:0	PRL	w	0x00	RTC 预分频器装载值高位 (RTC prescaler reload value high) 计数器的时钟频率计算公式如下：

Bit	Field	Type	Reset	Description
				$f_{TR\_CLK} = f_{RTCCLK} / (PRL + 1)$ 注：不推荐配置为 0，会产生 错误的 RTC 中断和标志位。

#### 18.5.4 预分频装载寄存器低位(RTC\_PRL)

偏移地址: 0x0C

复位值: 0x0000 0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	PRL															
Type	w															

Bit	Field	Type	Reset	Description
15:0	PRL	w	0x00	RTC 预分频器装载值低位 (RTC prescaler reload value low)

在每个 TR\_CLK 时间基准里，RTC 预分频器中的计数值会更新为 RTC\_PRL 寄存器的值。用户可读取 RTC\_DIV 寄存器获得预分频计数器的当前值，从而获得精确的时间。此寄存器是只读寄存器，其值在 RTC\_PRL 或 RTC\_CNT 寄存器中的值发生改变后，由硬件重新装载。

#### 18.5.5 预分频器分频因子寄存器高位(RTC\_DIVH)

偏移地址: 0x10

复位值: 0x0000 0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.															DIV
Type																r

Bit	Field	Type	Reset	Description
15:4	Reserved		0x0	保留，始终读为 0
3:0	DIV	r	0x00	RTC 时钟分频器分频因子高位 (RTC clock divider high)

#### 18.5.6 预分频器分频因子寄存器低位(RTC\_DIVL)

偏移地址: 0x14

复位值: 0x0000 0000

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

Field	DIV
Type	r

Bit	Field	Type	Reset	Description
15:0	DIV	r	0x0	RTC 时钟分频器分频因子高位 (RTC clock divider low)

RTC 核内部的 32 位可编程的计数器，可分别访问高 16 与低 16 位，预分频器产生的 TR\_CLK 时间基准用于计数。RTC\_CNT 寄存器存放计数器当前值。该寄存器受 RTC\_CR 的位 RTOFF 写保护，只有当 RTOFF 值为'1'时，才能写该寄存器。对高（RTC\_CNTH）或低寄存器（RTC\_CNTL）的写操作，会直接加载到对应的编程计数器，同时重新装载预分频寄存器。执行读操作，直接返回计数器计数值或显示系统时间。

### 18.5.7 计数器寄存器高位(RTC\_CNTH)

偏移地址：0x18

复位值：0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CNT															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x0000	RTC 计数器高位 (RTC counter high) 通过读 RTC_CNTL 寄存器获取 RTC 计数器当前值的高位部分。先进入配置模式。才能对该寄存器执行写操作。

### 18.5.8 计数器寄存器低位(RTC\_CNTL)

偏移地址：0x1C

复位值：0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CNT															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x0000	RTC 计数器低位 (RTC counter low) 通过读 RTC_CNTL 寄存器获取 RTC 计数器当前值的低位部分。先进入配置模式。才能对该寄存器执行写操作。

### 18.5.9 闹钟寄存器高位(RTC\_ALRH)

偏移地址: 0x20

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ALR															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	ALR	rw	0xFFFF	RTC 闹钟值高位 (RTC alarm high) 软件写入的闹钟时间的高位部分。先进入配置模式。才能对该寄存器执行写操作

### 18.5.10 闹钟寄存器低位(RTC\_ALRL)

偏移地址: 0x24

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	ALR															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	ALR	rw	0x0000	RTC 闹钟值低位 (RTC alarm high) 软件写入的闹钟时间的低位部分。先进入配置模式。才能对该寄存器执行写操作

若 RTC\_ALR=0, RTC\_MSR 不为 0: 当可编程计数器的值与 RTC\_MSR 中的 32 位值相等时, 即触发一次毫秒事件, 并发生闹钟中断;

若 RTC\_ALR 不为 0, RTC\_MSR 不为 0: 当秒事件触发后毫秒事件也触发则发生闹钟中断;

若 RTC\_ALR 不为 0, RTC\_MSR=0: 则当秒事件触发后则发生闹钟中断;

此寄存器受 RTC\_CR 寄存器里的 RTOFF 位写保护, 仅当 RTOFF=1 时, 允许写操作。

### 18.5.11 毫秒闹钟寄存器高位 (RTC\_MSRH)

偏移地址: 0x28

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.														MSR	
Type															rw	

Bit	Field	Type	Reset	Description
15:4	Reserved		0x0	保留, 始终读为 0
3:0	MSR	rw	0x00	RTC 闹钟值高位 (RTC msec high) 软件写入的毫秒闹钟时间的高位部分 注: 进入配置模式。才能对该寄存器执行写操作。

### 18.5.12 RTC 毫秒闹钟寄存器低位 (RTC\_MSRL)

偏移地址: 0x2C

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	MSR															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	MSR	rw	0x00	RTC 闹钟值低位 (RTC msec low) 软件写入的毫秒闹钟时间的低位部分 注: 进入配置模式。才能对该寄存器执行写操作。

### 18.5.13 RTC LSE 配置寄存器 (RTC\_LSE\_CFG)

偏移地址: 0x3C

复位值: 0x0250

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res							LSE_IB	LSE_RFB_SEL		LSE_DR	LSE_TEST				
Type								rw	rw		rw	rw				

Bit	Field	Type	Reset	Description
15:10	Reserved		0x00	保留, 始终读为 0 注: 进入配置模式。才能对该寄存器执行写操作。

Bit	Field	Type	Reset	Description
9:8	LSE_IB	rw	0x02	LSE 偏置电流调节： 从 2'b00~2'b11 调节逐渐变大
7:6	LSE_RFB_SEL	rw	0x01	反馈电阻选择： 00: 12M 01: 10M 10: 6M 11: 3M
5:4	LSE_DR	rw	0x01	LSE 驱动能力选择 2'b00~2'b11:驱动能力逐渐变大
3:0	LSE_TEST	rw	0x00	LSE 测试控制信号 LSE_TEST[3]: 0: 正常模式 1: 输出增强模式 LSE_TEST[2]:保留 LSE_TEST[3]: 00: 正常模式 01: 保留 10: 保留 11: 漏电补偿

## 19 通用异步收发器（UART）

### 19.1 UART 简介

对于使用工业标准 NRZ 异步串行数据格式的外设，通用异步收发器（UART）可以灵活地与之进行全双工数据交换。通过分数波特率发生器，UART 可以选择宽范围的波特率。异步单向通信和半双工单线通信，以及调制解调器（CTS/RTS）操作、IrDA 红外功能也能够被支持。另外，UART 也支持多处理器之间的通信。

对于高速数据通信，可以通过使用多缓冲器配置的 DMA 方式来实现。

## 19.2 UART 功能框图

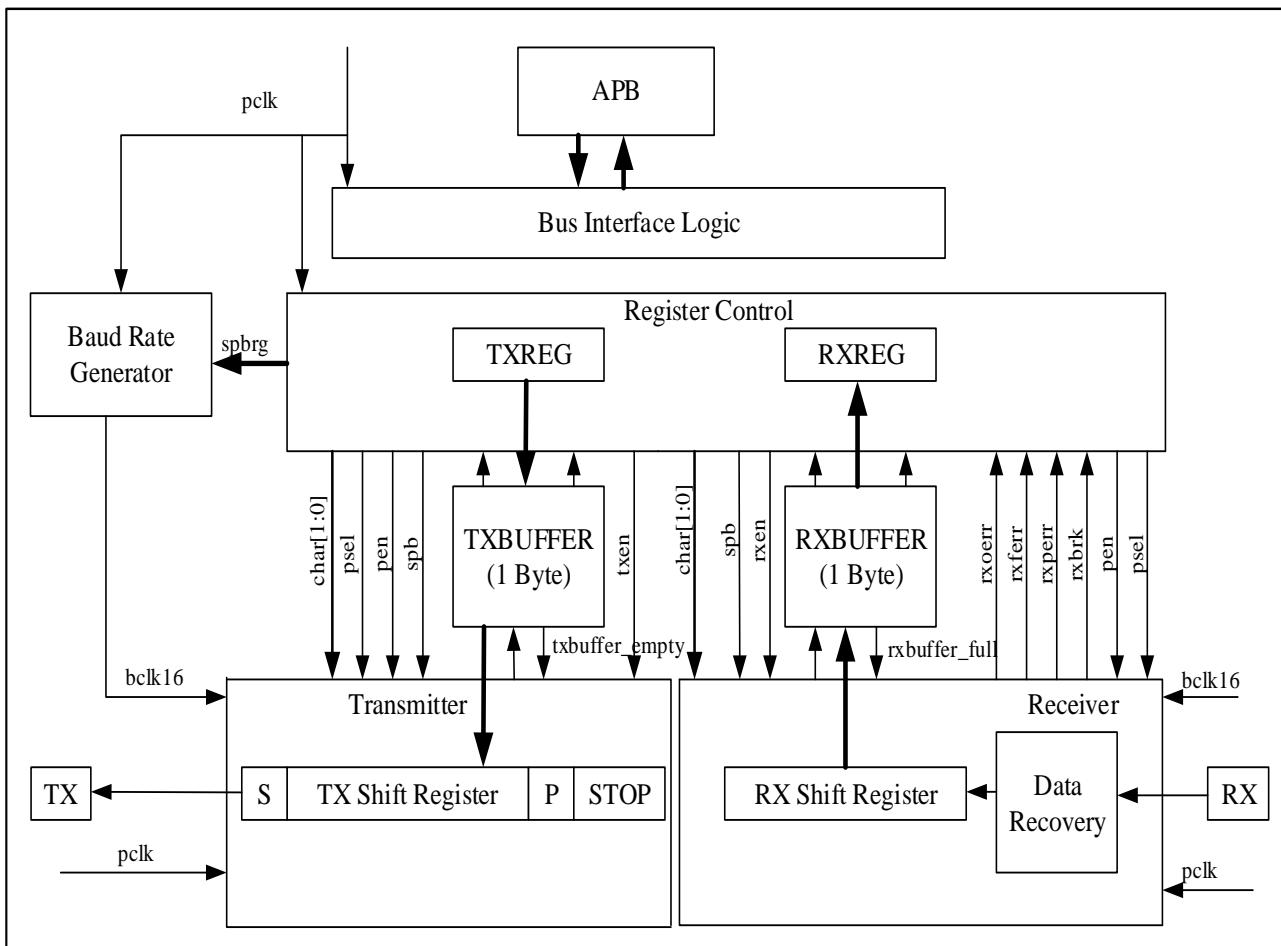


图 19-1 UART 功能框图

## 19.3 UART 主要特征

- 1) 支持异步方式下 RS-232S 协议，符合工业标准 16550
- 2) 支持 DMA 请求
- 3) 全双工异步操作
- 4) 分数波特率发生器
  - 可编程波特率，供发送器和接收器使用，最小分频系数为 1
- 5) 独立的发送和接收缓冲寄存器
- 6) 内置 1 字节发送和 1 字节接收缓冲
- 7) 发送和接收数据低位在前一个起始位开始，后面接数据位，输出的数据长度可为 5 位、6 位、7 位、8 位，最后为停止位。另外可选择是否有加奇偶校验位，奇偶校验位在数据位之后停止位之前。
- 8) 第 9 位可做同步帧配置
- 9) 支持硬件奇数或者偶数校验产生和侦测
- 10) 线断开产生和侦测
- 11) 线空闲产生和侦测
- 12) 支持 LIN 协议下收发 brk
- 13) 支持信号收发互换，接收和发送取反

- 14) 支持波特率自适应功能
- 15) 支持硬件自动流控制
- 16) 支持 IrDA SIR ENDEC 规范的红外功能

支持下面中断源:

- a) 发送端 BUFFER 空
- b) 接收端数据有效
- c) 接收缓冲缓存溢出
- d) 帧错误
- e) 奇偶校验错误
- f) 接收断开帧
- g) 发送移位寄存器完成
- h) 发送断开帧完成
- i) 接收同步帧
- j) 空闲帧完成
- k) 自动波特率结束
- l) 自动波特率错误

## 19.4 引脚定义及内部信号

## 19.5 中断

支持下面中断源:

- 1) 发送端 BUFFER 空
- 2) 接收端数据有效
- 3) 接收缓冲缓存溢出
- 4) 帧错误
- 5) 奇偶校验错误
- 6) 接收断开帧
- 7) 发送移位寄存器完成
- 8) 发送断开帧完成
- 9) 接收同步帧
- 10) 空闲帧完成
- 11) 自动波特率结束
- 12) 自动波特率错误

表 19-1 UART 中断请求

中断事件	中断状态	使能位
发送缓冲空	TX_INTF	TXIEN
接收到有效数据	RX_INTF	RXIEN
发送移位寄存器完成	TXC_INTF	TXC_EN

中断事件	中断状态	使能位
接收溢出错误	RXOERR_INTF	RXOERREN
奇偶校验错误	RXPERR_INTF	RXPERRLEN
帧错误	RXFERR_INTF	RXFERREN
接收断开帧	RXBRK_INTF	RXBRKEN
发送断开帧	TXBRK_INTF	TXBRK_EN
接收同步帧	RXB8_INTF	RXB8_EN
接收空闲帧	RXIDLE_INTF	RXIDLEN
自动波特率结束	ABREND_INTF	ABRENDIEN
自动波特率错误	ABRERR_INTF	ABRERRIEN

如果设置了对应的中断使能控制位，这些设置就可以产生各自对应的中断。

## 19.6 DMA

UART 可以利用 DMA 来搬运数据；使能 UART\_GCR 的 DMAMODE 位激活 DMA 模式；

利用 DMA 发送：DMA 搬送数据到 UART\_TDR

使能 uarten, txen, dmamode 后，只要 TXFIFO 为空，就请求 DMA 发送。

在利用 DMA 发送时，要配置好 DMA 的源地址（存储器地址）和目的地址（UART\_TDR），传输的数据量以及 DMA 通道。

利用 DMA 接收：DMA 将 UART\_RDR 数据搬走

使能 uarten, rxen, dmamode 后，只要 RXFIFO 有数据，即不为空，就请求 DMA 接收。

在利用 DMA 接收时，要配置好 DMA 的源地址（UART\_RDR）和目的地址（存储器地址），传输的数据量以及 DMA 通道。

## 19.7 UART 功能概述

在全双工通信的情况下，至少需要分配两个脚给 UART：接收数据输入（RX）和发送数据输出（TX）。

RX 接收数据串行输入。对于传输过程中产生的噪音，可以使用过采样的技术对其与数据进行区分并剔除，得到原本的数据。

TX 发送数据输出。当发送器被除能时，输出引脚恢复到它的 I/O 端口配置。当发送器被使能，并且无数据发送时，TX 引脚处于高电平。

总线总是从空闲状态跳转到发送或接收状态，起始位为一位，用 0 表示。

一个八位数据（5, 6 或 7 位），发送和接收顺序从最低位到最高位。

停止位用 1 表示一帧的结束，位数可配置为 0.5、1、1.5、2 个。

分数波特率发生器——可配置 16 位的整数以及 4 位的小数，得到的波特率应与给出公式计算的一

致。

通过使能 UART\_GCR 寄存器的 SWAP 位，可以交换接收和发送端的信号。

通过使能 UART\_GCR 寄存器的 RX\_TOG/TX\_TOG 位，可以将接收/发送端的信号取反。

下列引脚在硬件流控模式中需要：

nCTS 发送停止，当其为高电平时，表明当前接收端的 fifo 已满，发送端应停止之后的发送。

nRTS 发送请求，当其为低电平时，表明当前接收端可以接收数据。

### 19.7.1 UART 特性描述

通过配置 UART\_CCR 寄存器中的 CHAR 位，可以选择 5~8 位的数据长度。TX 引脚在起始位发送期间为低电平，在停止位期间为高电平。

空闲符号被视为完全由‘1’组成的一个完整的数据帧，后面跟着包含了数据的下一帧的开始位（‘1’的位数也包括了停止位的位数）。

包括停止位在内，一个完全由‘1’组成的完整数据帧，定义为一个空闲符号。下一个数据帧的起始位跟在空闲符之后。

断开符号被视为在一个帧周期内全部收到‘0’(包括停止位期间，也是‘0’)。在断开帧结束时，发送器再插入 1 个停止位（‘1’）来应答起始位。

包括停止位在内，一个完全由‘0’组成的完整数据帧，定义为一个断开符号。在断开帧结束后，发送端会再发送一个‘1’的停止位，使得下一帧的起始位能够被识别到。（产生下降沿被接收端检测到）

波特率发生器产生的时钟经过发送器和接收器的使能位置位控制之后，供给发送或接收使用。

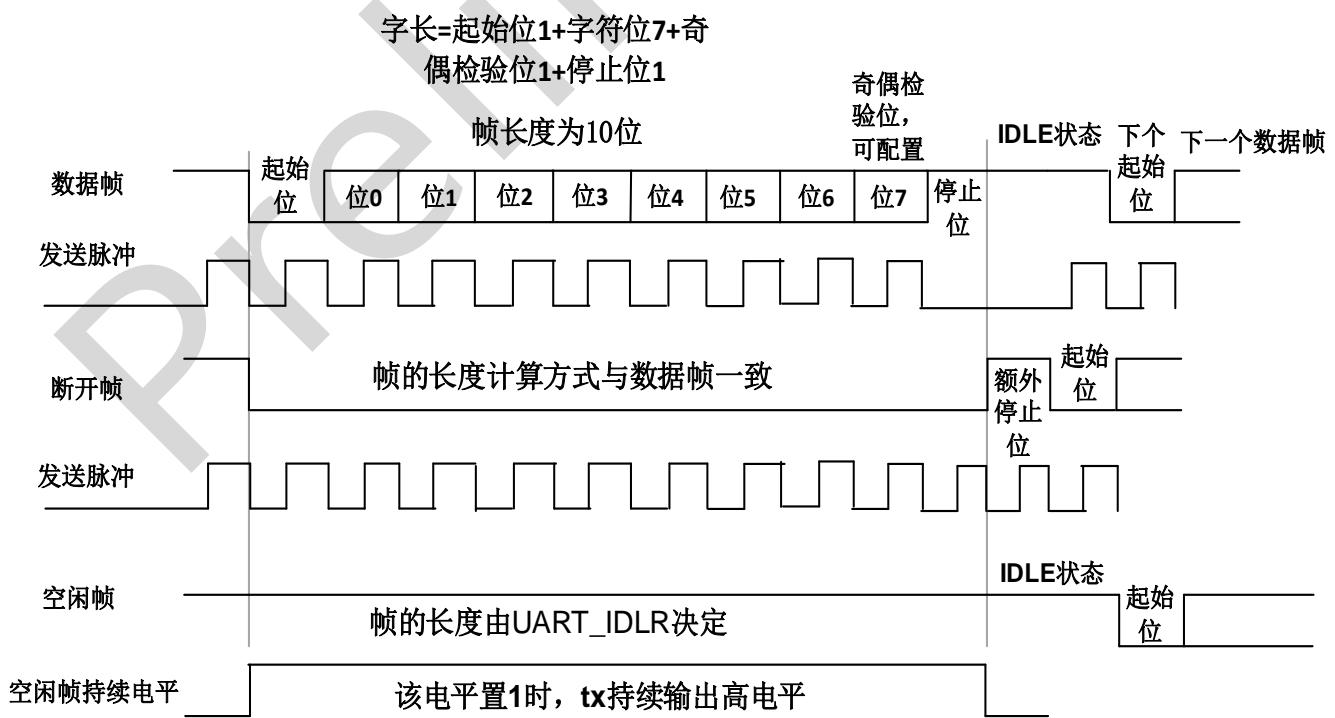


图 19-2 UART 时序

## 19.7.2 分数波特率发生器

设置 BRR 和 FRA 寄存器，可设置相应波特率，参考如下公式：

$$f_{baudrate} = \frac{f_{PCLK}}{16 \times UARTDIV}$$
$$UARTDIV = BRR + \frac{FRA}{16}$$
$$f_{baudrate} = \frac{f_{PCLK}}{16 \times BRR + FRA}$$

BRR 寄存器最小值为 1。

## 19.7.3 采样

由于异步操作没有单独的时钟，接收器需要一个同步于接收器方法。为了能够在接收引脚 RX 获得正确的字符数据，UART 有一个检测电路。UART 采用 16 倍数据波特率‘bclk16’的时钟进行采样 RX 引脚的数据，每个数据有 16 个时钟采样，取中间第 7,8,9 的下降沿的采样值。

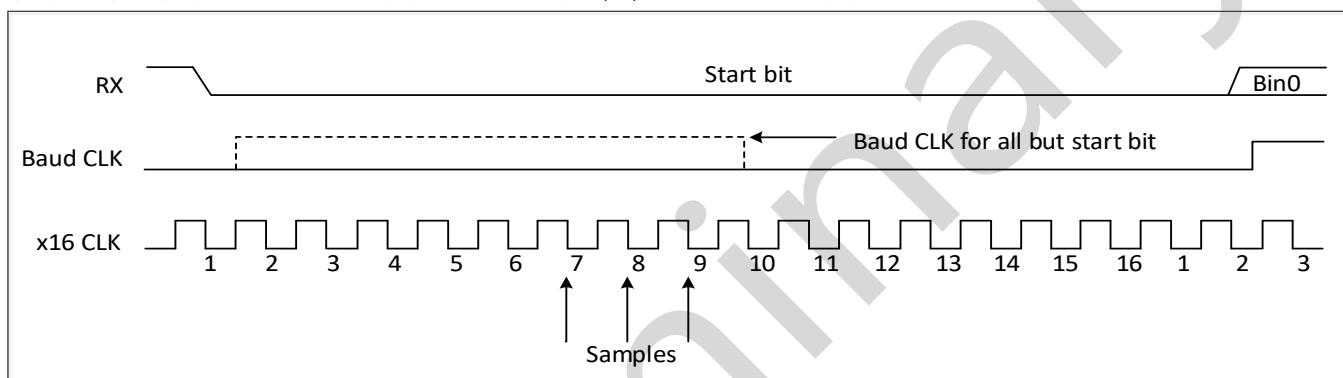


图 19-3 RX 引脚采样方案

## 19.7.4 容忍度

时钟偏差导致收发双方波特率偏差，为了提高接收端对波特率变化的容忍度，在 STOP 状态时，当三次数据采样完成后，接收端就开始准备下一帧数据的接收。当收发双方波特率偏差处于 $\pm 3\%$ 以内时，仍然能保证数据正常传输。

## 19.7.5 校验控制

奇偶控制(发送时生成一个奇偶位，接收时进行奇偶校验)可以通过设置 UART\_CCR 寄存器上的 PEN 位而激活。如果奇偶校验出错，无效数据仍然会从移位寄存器传送到 UART\_RDR 寄存器。设置 UART\_ICR 的 RXPERR 位以清除错误标志 RXPERR\_INTF。

当智能卡自动应答 SCAEN 使能时，在 STOP 状态时检测到奇偶校验错误，会产生 1 位的低电平信号（RXNACK），此时会将智能卡输出端拉低 1 个波特时钟周期，指示帧错误。

偶校验：校验位加上数据总共的 1 的个数为偶数。

奇校验：校验位加上数据总共的 1 的个数为奇数。

待发送数据的 1 的个数是不可改变的，通过配置校验位为 1 或者 0，满足奇偶校验的要求。

例如：数据=1000\_0110，有 3 个‘1’

偶校验：(UART\_CCR 中的 PSEL = 1)，校验位将是‘1’。

奇校验：(UART\_CCR 中的 PSEL = 0)，校验位将是‘0’。

在带奇偶校验的传输中，奇偶校验位的发送将跟在数据发送完最高有效位的后面。如果奇偶校验失败，硬件自动置‘1’UART\_ISR 寄存器中的 RXPERR\_INTF 标志位，若想产生中断，需提前配置 RXPERR 中断使能寄存器。

注：如 9bit 功能使能，奇偶检验无效。

## 19.7.6 发送器

CHAR 位的配置决定发送器发送数据的位数（5~8 位）。发送数据需要先置起发送使能位 (TXEN)，之后数据随着发送移位寄存器串行输出到 TX 引脚上。

### 19.7.6.1 字符发送

在 UART 发送期间，数据从 UART\_TDR 寄存器写入，经过一字节缓冲器缓冲，最后通过发送移位寄存器，以最低字节到最高字节的顺序，串行在 TX 引脚上输出。

一位低电平的起始位会先于字符发送，而在字符发送完成的最后会发送高电平的停止位。停止位的位数可通过寄存器配置。

当前数据传输未完成前不能清零 TXEN 位，否则此时波特率发生器会停止产生时钟，发送器由于没有时钟驱动导致发送数据后部分丢失，数据不完整。

### 19.7.6.2 可配置的停止位

通过 SPB 位进行编程，可以控制发送器每帧输出停止位的位数。

断开帧的总长度由起始位 + 数据位 + (校验位) + 停止位构成，长度不固定，会随着数据位、校验位、停止位设置改变长度。接收到断开帧会置位中断状态寄存器的 RXBRK\_INTF 位。

### 19.7.6.3 配置步骤

- 1) 通过在 UART\_GCR 寄存器上置位 UARTEN 位来使能 UART。
- 2) 编程 UART\_CCR 的 CHAR 位来定义字符长度。
- 3) 在 UART\_CCR 中 SPB 控制停止位的位数。
- 4) 置位 UART\_GCR 中的 TXEN 位。
- 5) 配置 UART\_BRR 和 UART\_FRA，产生需要的波特率时钟。
- 6) 将待发送数据写进入 UART\_TDR 寄存器 (此操作会硬件自动清除 TX\_INTF 位)。因为缓冲器为一字节，每次只能写入一个数据，后续每次传输数据都需要写待发送数据到 UART\_TDR 寄存器。

### 19.7.6.4 单字节通信

TX\_INTF 位的清零操作，由对数据寄存器 UART\_TDR 的写操作硬件自动完成。TX\_INTF 只由硬件写 1 和清 0，当该位被硬件置 1 时，说明：

- 1) 数据发送已经开始，数据已经从缓冲器传输到移位寄存器
- 2) TDR 的缓冲器已被读取，一字节缓冲器为空

3) UART\_TDR 寄存器可写，写入后不会影响前一个正在进行的数据传输。

如果 TXIEN 位被置 1，相应 TX\_INTF 中断也会产生。

如果 UART 处于传输状态，对 UART\_TDR 寄存器的写操作会把数据先存进 TDR 缓冲器，在当前传输结束后，即可传输缓冲器中的数据。

如果 UART 处于空闲状态，写 UART\_TDR 寄存器的同时也会直接把数据放入移位寄存器，进行串行移位传输，TX\_INTF 位会被立即拉高。同时缓冲器立刻清空，对应的寄存器 UART\_CSR 的 TXBUF\_EMPTY 也会被拉高。当停止位发送完成，一帧正式结束后，并且没有后续数据需要传输(TDR 缓冲器为空)，也就是说，所有的传输都已经完成，这时 TXC 会置位。

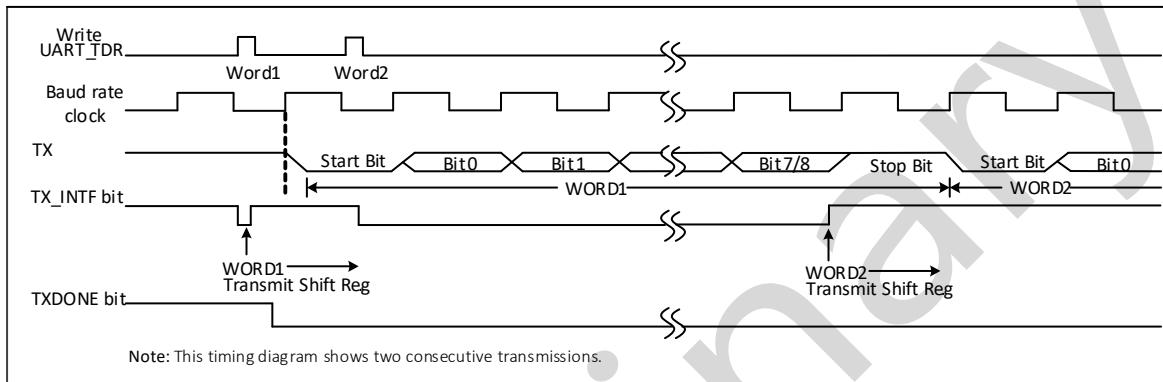


图 19-4 发送时状态位变化

### 19.7.6.5 断开符号

断开符号通过设置 BRK 寄存器发送。如果设置 BRK=1，在当前数据发送完成后，将会发送一个断开符号到 TX 引脚上。断开字符发送完成时(在断开符号的停止位时)，硬件设置 BRK=0。

为了能让相连的下一帧被识别到起始位，UART 会在断开帧停止位后再加一位为‘1’的停止位。

### 19.7.6.6 空闲符号

设置 UART\_IDLR 不为 0，在 TXEN 打开后，会立即发送一帧的空闲帧，然后再进行其他数据传输。空闲帧的长度写入 UART\_IDLR 寄存器中。

## 19.7.7 接收器

接收 FIFO 深度位 1 个字节。

### 19.7.7.1 字符接收

对于 UART 的接收器，RX 串行输入数据，接收顺序从数据最低有效位到最高有效位。在接收器中，内部总线和接收移位寄存器之间插入了 UART\_RDR 寄存器包含的缓冲器。

配置步骤：

- 1) 置‘1’UART\_GCR 寄存器的 UARTEN 位来使能 UART。
- 2) 编程 UART\_CCR 的 CHAR 位来定义字符长度。
- 3) 在 UART\_CCR 中 SPB 控制停止位的位数。
- 4) 配置 UART\_BRR 和 UART\_FRA，产生需要的波特率时钟。

- 5) 设置 **UART\_GCR** 的 **RXEN** 位。使能接收器，等待检测起始位。
- 6) 当接收一个完整帧时，
- 7) 硬件置位 **RX\_INTF**。此时移位寄存器的内容被转移到 **RDR**。数据已经被接收并且可以被读出(包括与之有关的错误标志)。
- 8) 若想产生中断，可置位 **UART\_IER** 的 **RX** 位使能中断。
- 9) 在接收期间如果检测到帧错误，或溢出错误，错误标志将被置起。
- 10) 软件读 **UART\_RDR** 寄存器可以清零 **RX\_INTF**。也可以通过置 1 对应的 **UART\_ICR** 寄存器清零 **RX\_INTF**。为了防止溢出错误，在下一字符接收结束前，**RX\_INTF** 位必须被清零。

在接收数据时，不应该软件清零 **RXEN** 位。如果 在接收时清零 **RXEN** 位，还在 **RX** 引脚上的数据将不再被接收。

### 19.7.7.2 断开符号

**UART** 接收器识别到一个断开帧时，会置位 **RXBRK\_INTF** 中断，数据不会被写入 **RXFIFO**。断开帧由起始位 + 数据位 + (校验位) + 停止位构成，长度不固定，会随着数据位、校验位、停止位设置的不同而不同。

### 19.7.7.3 空闲符号

空闲帧长度由 **UART\_IDLR** 寄存器设置(默认值为 0x0C)，设置 **UART\_IDLR** 不为 0 时，当接收到一个空闲帧时，**UART** 会置位 **RXIDLE\_INTF** 中断，数据不会被写入 **RXFIFO**。

### 19.7.7.4 溢出错误

如果在 **UART\_RDR** 没有读出前又接收到一个字符，则发生溢出错误。当溢出错误产生时：

- 1) **RXOERR\_INTF** 位被置位。
- 2) 未读出的 **RDR** 内容将不会丢失。读 **UART\_RDR** 寄存器仍能得到上一个数据。
- 3) 移位寄存器会移位操作 **RX** 进入的数据，但是数据不会传输到 **RDR** 中。
- 4) 若想产生中断，需配置 **RXOERREN** 位。

### 19.7.7.5 帧错误

当停止位没有在预期的时间上接收和识别出来时检测到帧错误。当帧错误被检测到时：

- 1) **RXFERR\_INTF** 位被硬件置起。
- 2) **UART\_RDR** 寄存器不会更新。
- 3) 如果 **RXFERREN** 位被设置，中断产生。

## 19.7.8 自动波特率检测

**UART** 能够根据接收到的信号，自动检测并设置 **UART\_BRR** 寄存器。自动波特率配置如下：

- 1) 配置 **ABRCR** 寄存器的 **Former\_edge** 和 **Latter\_edge**，选择自动波特率前一个边沿和后一个边沿。
- 2) 配置 **ABRCR** 寄存器的 **Abr\_bitcnt**，选择自动波特率检测位长度。
- 3) 设置 **ABRCR** 寄存器的 **Abr\_en**，打开自动波特率检测功能。

打开 **ABRENDIEN**，自动波特率结束后会产生自动波特率结束中断。

打开 ABRERRIEN，自动波特率位错误或检测溢出错误，会产生自动波特率错误中断。

示例：设置 Former\_edge=1, Latter\_edge=0, Abr\_bitcnt=2, Abr\_en=1

此时当接收到数据 0xEF 时，会自动检测波特率并重置 UART\_BRR 寄存器，RXFIFO 会接收到剩余数据 0x0E。

设置 Former\_edge=0, Latter\_edge=1, Abr\_bitcnt=2, Abr\_en=1

此时当接收到数据 0xF8 时，会自动检测波特率并重置 UART\_BRR 寄存器，RXFIFO 会接收到剩余数据 0x1F。

注：当 Former\_edge=1，并且接收的数据位最低位不为 1，会导致接收出错，RXFIFO 中无数据。

## 19.7.9 9 位数据通信

如果使能 UART\_CCR 寄存器的 B8EN 控制位，UART 使能 9 位数据的发送和接收，可以发送和接收 9 位数据。注意：在 B8EN 使能后，奇偶校验使能位 PEN 不起作用。

数据发送的时候，在写入数据到发送寄存器 UART\_TDR 前，需要先设置 B8TXD。B8TXD 作为发送数据的 MSB 和 UART\_TDR 的值同时发送。如果设置了 B8TOG，如果 B8TXD 与 B8POL 相同时，表示该数据作为地址帧或者同步帧，发送结束后 B8TXD 会自动翻转。在接下来的数据发送过程中，不需要再设置 B8TXD 为无效电平。

数据接收的时候，接收数据的最高位可以从寄存器位 B8RXD 读到。如果接收的 B8RXD 与 B8POL 相同时，中断状态寄存器 UART\_ISR 的 RXB8\_INTF 位会置位。

## 19.7.10 多处理器通信

多个处理器之间可以通过将几个 UART 连在一个网络里实现通信。作为主设备的 UART，它的 TX 输出连接其余从设备的 RX 输入。而其余从设备的 TX 输出逻辑相与之后连接主设备的 RX 输入。

在多处理器的通信中，主机通过寻址方式找到目的从机，而其他非目的从机应该保持在一个较低功耗的模式，以此来避免多余 UART 工作带来资源的浪费。

当从设备不是目的从机时，可以配置进入静默模式来降低功耗。在静默模式里：

- 1) 所有的接收相关状态标志位都不会被硬件置起。
- 2) 所有接收相关的中断不会产生。
- 3) 置‘1’UART\_CCR 寄存器中的 RWU 位表示接收器进入静默模式。软硬件均可读写 RWU 寄存器。

在设置地址标记唤醒时，如果接收 buffer 非空则不能软件修改。

通过配置 UART\_CCR 寄存器中的 WAKE 位，UART 进入或退出静默模式有以下两种方式：

- 1) WAKE=0：进行空闲总线检测。
- 2) WAKE=1：进行地址标记检测。

### 19.7.10.1 空闲总线检测(WAKE=0)

对 RWU 位写 1 使得从设备 UART 进入静默模式。当检测到总线空闲，从设备被唤醒，对应 RWU 被硬件清零。由于并没有正常帧输入，中断状态标志 RX\_INTF 不会置位。通过软件操作的方式也可以清零 RWU。

### 19.7.10.2 地址标记(Address Mark)检测(WAKE=1)

在这个模式里，作为地址帧的字节有效位的最高位等于 B8POL，作为数据帧的字节有效位的最高位为 (~B8POL)。对于一个地址帧，接收器将自己本地寄存器存储的地址与接收到的地址帧相比较，接收器的地址存储在 8 位 UART\_RXADDR 寄存器，另外可以配置 8 位寄存器 UART\_RXMASK，只有当 RXMASK 某一位为 1 时，相应的地址位才会参与比较，配置为 0 则不用比较，直接认为该 bit 位匹配成功。

若接收的地址帧经过 RXMASK 后与本地地址经过 RXMASK 后不匹配，则 UART 进入静默模式。对应硬件将 RWU 位置‘1’。由于 UART 已经在静默模式，本次接收器接收到的地址帧不会触发标志位如 RX\_INTF 置起，也不会产生中断，也不会发出 DMA 请求。

若接收的地址帧经过 RXMASK 后与本地地址经过 RXMASK 后匹配，则 UART 退出静默模式。对应硬件将 RWU 位清零，之后接收器可以正常接收并响应字节。由于 UART 已经退出静默模式，本次接收器收到这个匹配的地址帧也会触发标志位 RX\_INTF 置起。

### 19.7.11 单线半双工通信

配置 UART\_SCR 寄存器的 HDSEL 位进入单线半双工模式。当 UART 处于该模式时，UART\_SCR 寄存器的 SCEN 位必须保持为‘0’。

根据单线半双工协议配置 UART。单线半双工模式下的芯片内部逻辑会将 TX 与 RX 互连。使用 UART\_SCR 中的 HDSEL 位控制 UART 选择半双工或者全双工通信。

当 HDSEL 被配置成 1 时

- 1) RX 引脚悬空，不参与传输。传输时 UART 的 TX 直接连接另一个 UART 的 TX。
- 2) 在传输数据时，TX 一直被占用，直到停止位被发送完。
- 3) 在没有传输数据时，TX 处于被释放状态。因此，它在空闲状态的或接收状态时表现为一个标准 I/O 口。也就是说，TX 对应 I/O 在不被 UART 驱动时，必须配置成悬空输入(或开漏的输出高)。除了单线引脚的配置外，其余配置和正常传输一致。

在没有通信前，两个 UART 的 RXEN 都开启，处于等待接收状态，当需要通信交互数据时，两个 UART 要约定好谁来发送，发送方 RXEN 关闭，TXEN 使能。如果两边 UART 都试图发送数据，即发送冲突，由软件决定哪个 UART 先发送。

特别的是，硬件不会阻碍 UART 的发送。当发送使能位 TXEN 开启，只要数据写到数据寄存器上，TX 就会发送字节。

### 19.7.12 智能卡

进入智能卡模式需配置 UART\_SCR 寄存器的 SCEN 位。开启 SCEN，硬件自动为半双工模式，不用设置 HDSEL 位。

该模式下的 UART 符合 ISO7816-3 标准，支持智能卡异步协议。

以下为 UART 在该模式的配置方式：

- 1) 8 位数据位加校验位：UART\_CCR 寄存器配置 CHAR=11、PEN=1
- 2) 发送和接收时为 1.5 个停止位：UART\_CCR 寄存器配置 SPB1=1、SPB0=1

下图举例说明，有校验错误和无校验错误两种情况下的 TX 线上信号区别。

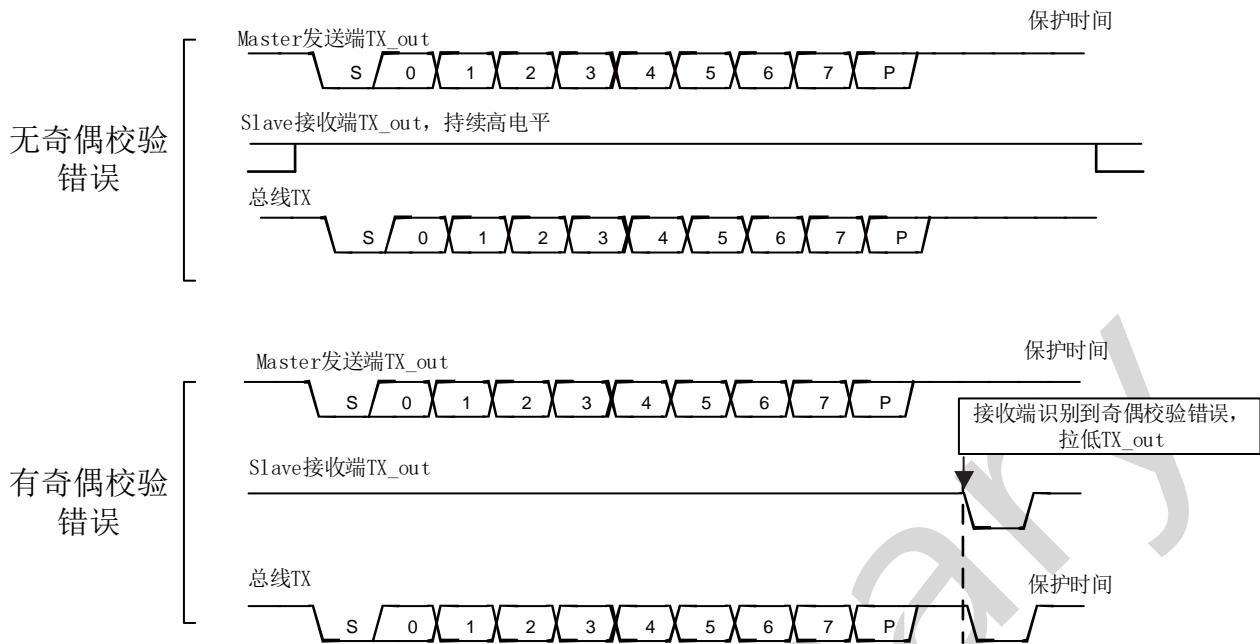


图 19-5 UART 奇偶校验方框图

智能卡端也是一根双向线。UART 作为主机，需要用 TX 单线驱动智能卡端的双向线。也就是说该模式为半双工模式，TX 与 RX 实质上使用的同一个 IO 口。

为了能够识别从机方向反馈回主机的校验错误信号，UART 发送数据时需使能 TXEN，而在停止位发送完毕后应该除能 TXEN，同时 TX 被拉到高电平，也就是说，TX 应被配置成开漏模式。这样当从机识别到数据校验错误之后就可以拉低数据线，使得主机的接收器能够收到发送数据错误的信息。

智能卡是一个单线半双工通信协议

- 1) 从数据被送至移位寄存器到 TX 引脚出现数据，相比于正常模式下的发送时间会多至少  $1/2$  波特时钟周期。正常模式中，一个满的发送移位寄存器将在下一个波特时钟沿开始向外移出数据。在智能卡模式里，发送时间相比正常模式会被延迟  $1/2$  波特时钟。
- 2) 如果在接收一个设置为 0.5 或 1.5 个停止位的数据帧期间，从机在接收数据时检测到了奇偶校验错误，为了将接收信息有误反馈给主机，UART 的接收器会拉低数据线一个波特时钟周期，即 NACK 信号。由于数据线被 UART 拉低，相当于主机发送了一个帧错误（奇偶校验位后本应为停止位，被拉低成 0）。主机收到反馈后，应用程序可以利用这个反馈信号，根据协议处理重发数据。另外该 NACK 信号还需要配置 SCAEN 位才会产生。
- 3) 通过配置保护时间寄存器，可以延长 TXC 标志的置起时间。在正常传输模式下，当发送缓冲器为空且没有后续数据输入到 TDR 时，硬件自动置起 TXC。在智能卡模式下，发送缓冲器一旦为空，对应的保护时间计数器就会开始向上计数，当计数器的值累加到等于配置的保护时间寄存器的值时，TXC 才会被置起。TXC 在计数器累加期间不会置‘1’。
- 4) 智能卡模式不会影响 TXC 标志因 TDR 输入数据而被硬件自动清零。
- 5) 如果主机收到从机的 NACK 信号，主机的接收器不会把 NACK 当作起始位检测。由 ISO 协议规定，接收到的 NACK 可以维持 1 个或 2 个 波特时钟周期。
- 6) 对于从机而言，从机检测到校验错误后，由从机向主机发送 NACK，从机的接收器也不会把

NACK 检测成起始位。

注意：

- a) 智能卡模式中没有断开帧的定义。一个包括停止位在内的全 0 数据帧，将被当成数据帧错误而不是断开帧。
- b) 智能卡模式中没有空闲帧的定义。当来回切换 TXEN 位时，智能卡不会认为接收到空闲帧。

下图详细说明了 UART 对于 NACK 信号的采样逻辑。如图所示，UART 正在发送数据，停止位配置为 1.5 位。使能 UART 的接收功能块，用于检查数据的完整性和 NACK 信号。

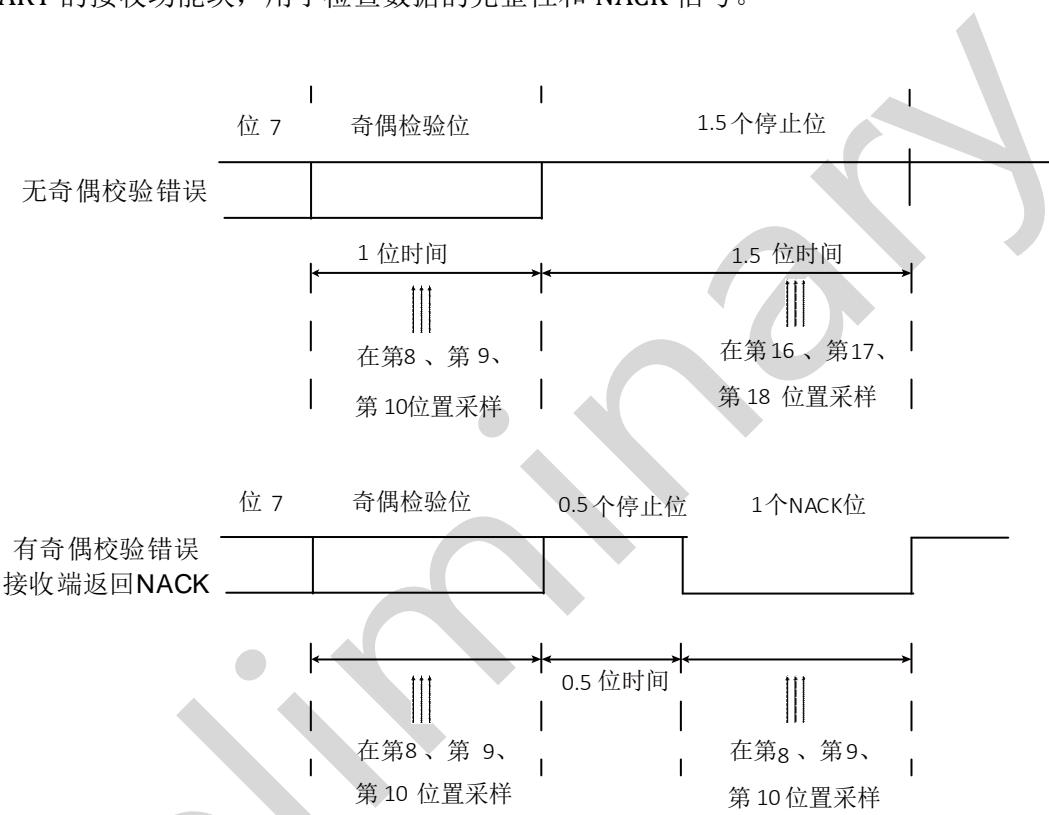


图 19-6 UART 采样 NACK 信号方框图

### 19.7.13 红外 IrDA 功能

UART 模块支持 IrDA(infrared data association 红外数据组织)SIR ENDEC 规范；IrDA SIR 物理层规定使用反相归零调制方案(RZI)，该方案用一个红外光脉冲代表逻辑“0”，正常模式下，“0”的脉宽为  $3/16 \text{ baud}_1 \text{ clk}$ 。低功耗模式下，脉冲的宽度是低功耗波特率  $\text{sirlp\_clk} * 3$ （由 PSC\_REG 分频 pclk 得到），频率范围在  $(1.42 \text{ MHz} < \text{sirlp\_clk} < 2.12 \text{ MHz})$ 。低功耗模式可编程分频器将系统时钟进行分频以达到这个值。

SIR 发送编码器和 SIR 接收解码器实现 UART 比特流与红外脉冲流的转换。

IrDA 是一个半双工通信协议，编解码不可同时进行；在 IrDA 模式里，UART\_CCRL 寄存器上的 STOP 位必须配置成 1 个停止位。

在 IREN 使能时，UART\_TX 输出高电平。

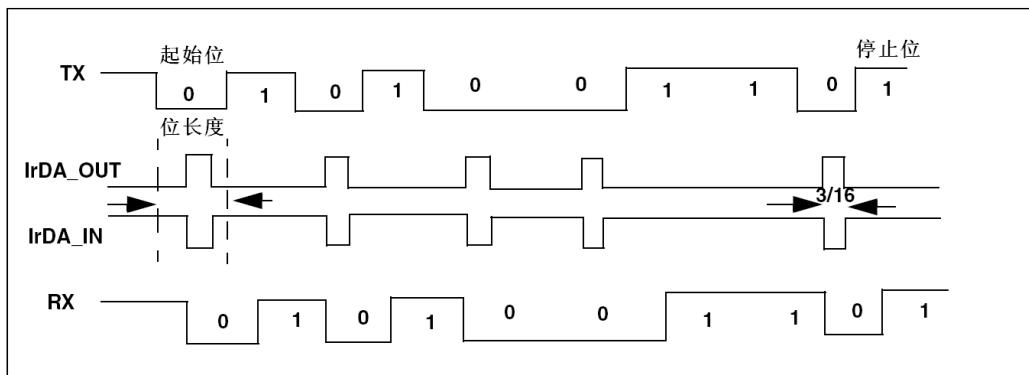


图 19-7 普通模式下 IrDA 发送和接收图

## 19.8 UART 寄存器描述

表 19-2 UART 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	UART_TDR	UART 发送数据寄存器	0x00000000
0x04	UART_RDR	UART 接收数据寄存器	0x00000000
0x08	UART_CSR	UART 当前状态寄存器	0x00000009
0x0C	UART_ISR	UART 中断状态寄存器	0x00000000
0x10	UART_IER	UART 中断使能寄存器	0x00000000
0x14	UART_ICR	UART 中断清除寄存器	0x00000000
0x18	UART_GCR	UART 全局控制寄存器	0x00000000
0x1C	UART_CCR	UART 通用控制寄存器	0x00000000
0x20	UART_BRR	UART 波特率寄存器	0x00000001
0x24	UART_FRA	UART 分数波特率寄存器	0x00000000
0x28	UART_RXADDR	UART 接收地址寄存器	0x00000000
0x2C	UART_RXMASK	UART 接收掩码寄存器	0x000000FF
0x30	UART_SCR	UART SCR 寄存器	0x00000000
0x34	UART_IDLR	UART IDLE 数据长度寄存器	0x0000000C
0x38	UART_ABRCR	UART ABRCR 自动波特率控制寄存器	0x00000000
0x3C	UART_IRDA	UART IRDA 红外功能控制寄存器	0x00000100

### 19.8.1 UART 发送数据寄存器(UART\_TDR)

偏移地址：0x00

复位值：0x0000 0000

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

Bit	Field	Type	Reset	Description
31:9	Reserved			保留, 始终读为 0
8:0	TXREG	rw	0x00	发送数据寄存器(Transmit data register)

## 19.8.2 UART 接收数据寄存器(UART\_RDR)

偏移地址: 0x04

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:9	Reserved			保留, 始终读为 0
8:0	RXREG	r	0x00	接收数据寄存器(Receive data register) 该寄存器只读

## 19.8.3 UART 当前状态寄存器(UART\_CSR)

偏移地址: 0x08

复位值: 0x0000 0009

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved							TXEPT TXFUL RXAVL TXC								

			L		
Type			r	r	r

Bit	Field	Type	Reset	Description
31:4	Reserved			保留, 始终读为 0
3	TXEPT	r	0x01	发送缓冲空标识位(Transmit buffer empty flag bit) 1:发送缓冲为空 0:发送缓冲不为空
2	TXFULL	r	0x00	发送缓冲满标识位(Transmit buffer full flag bit) 1:发送缓冲满 0:发送缓冲不满
1	RXAVL	r	0x00	接收有效字节数据标识位(Receive valid data flag bit) 当接收缓冲接收了一个完整字节的数据时置位该位。 1:接收缓冲接收了一个完整有效的字节数据 0:接收缓冲为空
0	TXC	r	0x01	发送结束标识位(Transmit complete flag bit) 1:发送缓冲和发送移位寄存器都为空 0:发送不为空

#### 19.8.4 UART 中断状态寄存器(UART\_ISR)

偏移地址: 0x0C

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24
Field	Reserved							
Type								
Bit	23	22	21	20	19	18	17	16
Field	Reserved							
Type								
Bit	15	14	13	12	11	10	9	8
Field	Reserved				ABRERR_ INTF	ABREND_ INTF	RXIDLE_ INTF	RXB8_ INTF
Type					r	r	r	r
Bit	7	6	5	4	3	2	1	0

Field	TXBRK_ INTF	RXBRK_ INTF	RXFERR_ INTF	RXPERR_ INTF	RXOERR — INTF	TXC_ INTF	RX_INTF	TX_INTF
Type	r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31:12	Reserved			保留, 始终读为 0
11	ABRERR_INTF	r	0x00	<p>UART 自动波特率错误中断标志位(Auto baud rate error interrupt flag bit)</p> <p>1:自动波特率错误 0:无自动波特率错误</p>
10	ABREND_INTF	r	0x00	<p>UART 自动波特率结束中断标志位(Auto baud rate end interrupt flag bit)</p> <p>1:自动波特率结束 0:自动波特率未结束</p>
9	RXIDLE_INTF	r	0x00	<p>UART 接收空闲帧中断标志位(Receive frame idle interrupt flag bit)</p> <p>在停止位后 RX 引脚在一段时间内接收到若干(UART_IDLR 数量)高电平。</p> <p>1:检测空闲帧 0:没有空闲帧</p>
8	RXB8_INTF	r	0x00	<p>UART 同步帧中断标志位</p> <p>在 9 位通讯模式下, 当接收到到数据的第九位与寄存器 CCR.B8POL 相同时, RXB8_INT 位置。该位可以作为中断请求信号</p> <p>1:接收到的同步帧 0:没有接收到同步帧</p>
7	TXBRK_INTF	r	0x00	<p>UART 断开帧发送完成中断标志位</p> <p>1:移位寄存器断开帧数据发送完成 0:移位寄存器空或正在移位发送备注: 不能连续发送断开帧。</p>

Bit	Field	Type	Reset	Description
6	RXBRK_INTF	r	0x00	<p>UART 接收断开帧中断标志位(Receive frame break interrupt flag bit)</p> <p>在异常停止位后 RX 引脚在一段时间内接收到 10 个或大于 10 位的低电平。</p> <p>1:检测断开帧 0:没有断开帧</p>
5	RXFERR_INTF	r	0x00	<p>帧错误中断标志位(Frame error interrupt flag bit)</p> <p>帧错误发生在当检测到异常停止位。</p> <p>1:检测一个帧错误 0:没有帧错误</p>
4	RXPERR_INTF	r	0x00	<p>奇偶校验错误中断标志位(Parity error interrupt flag bit)</p> <p>1:检测到奇偶校验错误 0:没有奇偶校验错误</p>
3	RXOERR_INTF	r	0x00	<p>接收溢出错误中断标志位(Receive overflow error interrupt flag bit)</p> <p>仅当 autoflowen=0 时会置位。</p> <p>1:接收溢出错误 0:没有溢出错误</p>
2	TXC_INTF	r	0x00	<p>UART 发送移位寄存器完成中断标志位。</p> <p>1:移位寄存器数据发送完成 0:移位寄存器空或正在移位发送注：此标志位与保护时间相关</p>
1	RX_INTF	r	0x00	<p>接收有效数据中断标志位(Receive valid data interrupt flag bit)</p> <p>当接收缓冲接收了一个完整字节的数据时置位该位。</p> <p>1:接收缓冲有效字节数据 0:接收缓冲为空</p>
0	TX_INTF	r	0x00	<p>发送缓冲空中断标志位(Transmit buffer empty interrupt flag bit)</p> <p>1:发送缓冲空 0:发送缓冲不为空</p>

## 19.8.5 UART 中断使能寄存器(UART\_IER)

偏移地址： 0x10

复位值： 0x0000 0000

Bit	31	30	29	28	27	26	25	24
Field	Reserved							
Type								
Bit	23	22	21	20	19	18	17	16
Field	Reserved							
Type								
Bit	15	14	13	12	11	10	9	8
Field	Reserved				ABRERR_ IEN	ABREND_ IEN	RXIDLE_ IEN	RXB8_ IEN
Type					rw	rw	rw	rw
Bit	7	6	5	4	3	2	1	0
Field	TXBRK_ IEN	RXBRK_ IEN	RXFERR_ IEN	RXPERR_ IEN	RXOERR_ IEN	TXC_IEN	RX_IEN	TX_IEN
Type	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:12	Reserved			保留, 始终读为 0
11	ABRERR_IEN	rw	0x00	自动波特率错误中断使能(Auto baud rate error enable bit)。 1:中断使能 0:中断禁止
10	ABREND_IEN	rw	0x00	自动波特率结束中断使能 (Auto baud rate end enable bit)。 1 中断使能 0 中断禁止
9	RXIDLE_IEN	rw	0x00	接收空闲帧中断使能位 (Receive frame idle interrupt enable bit) 1:中断使能 0:中断禁止
8	RXB8_IEN	rw	0x00	UART 同步帧中断使能控制位。 1:使能接收同步帧中断 0:禁止接收同步帧中断
7	TXBRK_IEN	rw	0x00	UART 断开帧发送完成中断使能控制位。 1:使能发送断开帧完成中断 0:禁止发送断开帧完成中断

Bit	Field	Type	Reset	Description
6	RXBRK_IEN	rw	0x00	UART 接收断开帧中断使能位 (Receive frame break interrupt enable bit) 1:中断使能 0 中断禁止
5	RXFERR_IEN	rw	0x00	帧错误中断使能位 (Frame error interrupt enable bit) 1:中断使能 0:中断禁止
4	RXPERR_IEN	rw	0x00	奇偶校验错误中断使能位 (Parity error interrupt enable bit) 1:中断使能 0:中断禁止
3	RXOERR_IEN	rw	0x00	接收溢出错误中断使能位 (Receive overflow error interrupt enable bit) 1:中断使能 0:中断禁止
2	TXC_IEN	rw	0x00	UART 发送移位寄存器完成中断使能控制位。 1:移位寄存器数据发送完成 0:移位寄存器空或正在移位发送
1	RX_IEN	rw	0x00	接收缓冲中断使能位 (Receive buffer interrupt enable bit) 1 中断使能 0 中断禁止
0	TX_IEN	rw	0x00	发送缓冲空中断使能位 (Transmit buffer empty interrupt enable bit) 1 中断使能 0 中断禁止

## 19.8.6 UART 中断清除寄存器(UART\_ICR)

偏移地址: 0x14

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24
Field	Reserved							
Type								
Bit	23	22	21	20	19	18	17	16
Field	Reserved							

Type								
Bit	15	14	13	12	11	10	9	8
Field	Reserved				ABRERR _ICLR	ABREND _ICLR	RXIDLE_ ICLR	RXB8_ ICLR
Type					w	w	w	w
Bit	7	6	5	4	3	2	1	0
Field	TXBRK_ ICLR	RXBRK_ ICLR	RXFERR _ICLR	RXPERR _ICLR	RXOERR _ICLR	TXC_ ICLR	RX_ICL R	TX_ICLR
Type	w	w	w	w	w	w	w	w

Bit	Field	Type	Reset	Description
31:12	Reserved			保留, 始终读为 0
11	ABRERR_ICLR	w	0x00	UART 自动波特率错误中断清除位(Auto baud rate error clear bit) 1:中断清除 0:中断未清除
10	ABREND_ICLR	w	0x00	UART 自动波特率结束中断清除位(Auto baud rate end clear bit) 1:中断清除 0:中断未清除
9	RXIDLE_ICLR	w	0x00	UART 接收空闲帧中断清除位(Receive frame idle interrupt clear bit) 1:中断清除 0:中断未清除
8	RXB8_ICLR	w	0x00	UART 同步帧中断标志清除控制位。 1:清除接收同步帧中断标志 0:无动作
7	TXBRK_ICLR	w	0x00	UART 断开帧发送完成中断标志清除控制位。 1:清除断开帧发送完成中断标志 0:无动作
6	RXBRK_ICLR	w	0x00	UART 接收断开帧中断清除位(Receive frame break interrupt clear bit) 1:中断清除 0:中断未清除

Bit	Field	Type	Reset	Description
5	RXFERR_ICLR	w	0x00	帧错误中断使能位(Frame error interrupt enable bit) 1:中断清除 0:中断未清除
4	RXPERR_ICLR	w	0x00	奇偶校验错误中断清除位(Parity error interrupt clear bit) 1:中断清除 0:中断未清除
3	RXOERR_ICLR	w	0x00	接收溢出错误中断清除位(Receive overflow error interrupt clear bit) 1:中断清除 0:中断未清除
2	TXC_ICLR	w	0x00	发送完成中断清除位(Transmit complete interrupt clear bit) 1:中断清除 0:中断未清除
1	RX_ICLR	w	0x00	接收中断清除位(Receive interrupt clear bit) 1:中断清除 0:中断未清除
0	TX_ICLR	w	0x00	发送缓冲空中断清除位(Transmit buffer empty interrupt clear bit) 1:中断清除 0:中断未清除

### 19.8.7 UART 全局空制寄存器(UART\_GCR)

偏移地址: 0x18

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15:11	10	9	8	7	6:5	4	3	2	1	0					
Field	Res.	TXTO G	RXTO G	SWA P	SELB 8	Res.	TXEN	RXEN	AUTO FLO W EN	DMA MOD E	UART EN					
Type		rw	rw	rw	rw		rw	rw	rw	rw	rw					

Bit	Field	Type	Reset	Description
31:11	Reserved			保留, 始终读为 0
10	TXTOG	rw	0x00	<p>发送取反位</p> <p>1:发送取反</p> <p>0:无效</p>
9	RXTOG	rw	0x00	<p>接收取反</p> <p>1:接收取反</p> <p>0:无效</p>
8	SWAP	rw	0x00	<p>输入与输出交换</p> <p>1:输入输出交换</p> <p>0:无效</p> <p>注: SWAP 置位后, GPIOx_CRL 寄存器的 MODE 需要更改, 如: 原输入模式变为输出模式。</p>
7	SELB8	rw	0x00	<p>选择 B8 数据接收或发送是否有效</p> <p>1:B8 数据收发</p> <p>0:无效</p> <p>UART_CCR 位 B8EN 有效时, 发送时发送 9 位数据; 接收时 接收寄存器位 CHAR + 1 的数据长度。</p>
6:5	Reserved			保留, 始终读为 0
4	TXEN	rw	0x00	<p>发送使能位(Enable transmit)</p> <p>1:发送使能</p> <p>0:发送禁止。可以清除 TX BUFFER</p>
3	RXEN	rw	0x00	<p>接收使能位(Enable receive)</p> <p>1:接收使能</p> <p>0:接收禁止。可以清除 RX BUFFER.</p>
2	AUTOFLOWEN	rw	0x00	<p>自动流控制使能位(AutDMAtic flow control enable bit)</p> <p>1:自动流控制使能</p> <p>0:自动流控制禁止</p>
1	DMAMODE	rw	0x00	<p>DMA 方式选择位(DMA mode selection bit)</p> <p>1:选择 DMA 方式</p> <p>0:选择正常方式</p>
0	UARTEN	rw	0x00	<p>UART 模块选择位(UART mode selection bit)</p> <p>1:UART 模块使能</p> <p>0:UART 模块禁止</p>

## 19.8.8 UART 通用控制寄存器(UART\_CCR)

偏移地址: 0x1C

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24
Field	Reserved							
Type								
Bit	23	22	21	20	19	18	17	16
Field	Reserved							
Type								
Bit	15	14	13	12	11	10	9	8
Field	Reserved	LIN	WAKE	RWU	B8EN	B8TOG	B8POL	B8TXD
Type		rw	rw	rw	rw	rw	rw	rw
Bit	7	6	5	4	3	2	1	0
Field	B8RXD	SPB1	CHAR		BRK	SPB0	PSEL	PEN
Type	r	rw	rw		rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:15	Reserved			保留, 始终读为 0
14	LIN	rw	0x00	UART LIN 协议收发断开帧(UART LIN enable bit) 1:LIN 协议有效 0:LIN 协议无效
13	WAKE	rw	0x00	唤醒方法。 该位决定把 UART 唤醒的方法。 1:地址标记唤醒 0:空闲总线唤醒
12	RWU	rw	0x00	接收静默。 该位用来决定是否把 UART 置于静默模式。该位可以由软件设置或清除。当唤醒序列到来时, 硬件也会自动将其清零。 1:接收器处于静默模式 0:接收器处于正常工作模式 在设置地址标记唤醒时, 如果接收 buffer 非空则不能软件修改。

Bit	Field	Type	Reset	Description
11	B8EN	rw	0x00	<p>UART 同步帧第九位使能控制位。</p> <p>该位使能后校验使能位 PEN 不起作用。</p> <p>1:使能同步帧第九位发送 0:禁止同步帧第九位发送</p>
10	B8TOG	rw	0x00	<p>UART 同步帧发送第九位自动翻转控制位。</p> <p>1:使能第九位自动翻转 0:禁止第九位自动翻转</p> <p>注：在 B8TXD 和 B8POL 的值相同时，在配置完寄存器后传输的第二个数据开始翻转，第一个数据默认为地址位。</p>
9	B8POL	rw	0x00	<p>UART 同步帧第九位极性控制位。</p> <p>1:同步帧第九位高电平有效 0:同步帧第九位低电平有效</p>
8	B8TXD	rw	0x00	<p>UART 同步帧发送数据第九位。</p> <p>1:发送同步帧第九位为高电平 0:发送同步帧第九位为低电平</p>
7	B8RXD	r	0x00	<p>UART 同步帧接收数据第九位。</p> <p>只读。</p> <p>1:接收同步帧第九位为高电平 0:接收同步帧第九位为低电平</p>
6	SPB1	rw	0x00	停止位选择位，与 SPB0 结合设置停止位位数。
5:4	CHAR	rw	0x00	<p>UART 数据宽度位(UART width bit)</p> <p>00 :5 位; 01 :6 位 10 :7 位; 11 :8 位</p>
3	BRK	rw	0x00	<p>UART 发送断开帧 (UART transmit frame break)</p> <p>1:串行强制输出逻辑'0' (断开帧) 0:禁止断开</p>
2	SPB0	rw	0x00	<p>停止位选择(Stop bit selection)</p> <p>设置发送停止位位数。</p> <p>SPB1, SPB0 00:1 个停止位 SPB1, SPB0 01:2 个停止位 SPB1, SPB0 10:0.5 个停止位 SPB1, SPB0 11:1.5 个停止位</p>

Bit	Field	Type	Reset	Description
1	PSEL	rw	0x00	<p>校验选择位(Parity selection bit)</p> <p>当校验使能后，该位用于选择是采用偶校验还是奇校验。</p> <p>1:偶校验 0:奇校验</p>
0	PEN	rw	0x00	<p>校验使能位(Parity enable bit)</p> <p>1:发送接收使能校验 0:禁止校验</p>

### 19.8.9 UART 波特率寄存器(UART\_BRR)

偏移地址: 0x20

复位值: 0x0000 0001

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

Bit	Field	Type	Reset	Description
31:16	Reserved			保留，始终读为 0
15:0	DIV_Mantissa	rw	0x0001	<p>UARTDIV 的整数部分</p> <p>这 16 位定义了 UART 分频器除法因子 (UARTDIV) 的整数部分。</p> <p>DIV_Mantissa 最小值为 1</p>

### 19.8.10 UART 分数波特率寄存器(UART\_FRA)

偏移地址: 0x24

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0
15:0	DIV_Fraction	rw	0x00	UARTDIV 的小数部分 低 4 位定义了 UART 分频器除法因子(UARTDIV)的小数部分。

### 19.8.11 UART 接收地址寄存器(UART\_RXADDR)

偏移地址: 0x28

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0
7:0	RXADDR	rw	0x00	UART 同步帧数据本机匹配地址。 如果 RXMASK =0xFF 时, 接收到的同步帧数据与本机匹配地址相同时, 产生 RXB8_INTF。 地址 0 是广播地址, 收到后都会响应。

### 19.8.12 UART 接收掩码寄存器(UART\_RXMASK)

偏移地址: 0x2C

复位值: 0x0000 00FF

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

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0
7:0	RXMASK	rw	0xFF	数据位全为“0”时, 接收到任何数据都产生同步帧中断请求。 如果数据位为“1”, RDR 和 RXADDR 的相应位匹配时, 产生同步帧中断请求。

### 19.8.13 UART SCR 寄存器(UART\_SCR)

偏移地址: 0x30

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15:13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Field	Res.	HDSEL	SCFCNT						Res.	NACK	SCAE	N	SCEN			
Type		rw	rw							r	rw	rw				

Bit	Field	Type	Reset	Description
31:13	Reserved			保留, 始终读为 0
12	HDSEL	rw	0x00	单线半双工模式选择。 1:使能半双工模式 0:禁止半双工模式
11:4	SCFCNT	rw	0x00	ISO7816 保护计数器。 当发送数据为在保护计数器期间内的低电平, 禁止为下一个数据的起始位。 1~255 个波特率计数时间 注: 当设置计数器为 0 或 FF 时, 均为 255 个波特率计数时间。
3	Reserved			保留, 始终读为 0
2	NACK	r	0x00	主接收帧应答位。 保护期间收到低电平置位
1	SCAEN	rw	0x00	ISO7816 校验自动应答位。 1:使能自动应答 0:禁止自动应答

Bit	Field	Type	Reset	Description
0	SCEN	rw	0x00	ISO7816 使能控制位。 1:使能 ISO7816 功能 0:禁止 ISO7816 功能

### 19.8.14 UART IDLE 数据长度寄存器(UART\_IDLR)

偏移地址: 0x34

复位值: 0x0000 000C

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

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0
15:0	IDLR	rw	0x000C	UART idle 数据长度寄存器(Idle data length register) 数据长度不为 0。

### 19.8.15 UART 自动波特率寄存器(UART\_ABRCR)

偏移地址: 0x38

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15:5				4				3				2:1		0	
Field	Reserved				Latter_edge				Former_edge				Abr_bitcnt		Abren	
Type					rw				rw				rw		rw	

Bit	Field	Type	Reset	Description
31:5	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
4	Latter_edge	rw	0x00	自动波特率后一个边沿选择。 1:上升沿 0:下降沿
3	Former_edge	rw	0x00	自动波特率前一个边沿选择。 1:上升沿 0:下降沿
2:1	Abr_bitcnt	rw	0x00	自动波特率检测长度。 检测前一个边沿和后一个边沿之间的位长。 11:8 位 10:4 位 01:2 位 00:1 位
0	Abren	rw	0x00	自动波特率使能。 只能在 UART 空闲时使能自动波特率，使能后检测接收信号的边沿，完成自动波特率检测后，硬件自动设置 UART_BRR 和 UART_FRA 寄存器。 1:自动波特率使能 0:自动波特率禁止

### 19.8.16 UART 红外功能控制寄存器(UART\_IRDA)

偏移地址: 0x3C

复位值: 0x0000 0100

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15:8						7:2			1		0				
Field	PSC_REG						Reserved			Sirlp		Siren				
Type	rw									rw		rw				

Bit	Field	Type	Reset	Description
31:16	Reserved			保留，始终读为 0

Bit	Field	Type	Reset	Description
15:8	PSC_REG	rw	0x01	<p>预分频寄存器 (Prescaler value)</p> <p>在红外低功耗模式下, 对 UART 源时钟 (pclk) 分频获得低功耗模式下频率 sirlp_clk:</p> <ul style="list-style-type: none"> <li>0000 0000 : 保留, 不能写入该值</li> <li>0000 0001 : 对源时钟 1 分频</li> <li>0000 0010 : 对源时钟 2 分频</li> <li>0000 0011 : 对源时钟 3 分频</li> <li>.....</li> </ul> <p>注: 保证分频后的时钟频率在 (1.42 MHz &lt; sirlp_clk &lt; 2.12 MHz) 之间</p>
7:2	Reserved			保留, 始终读为 0
1	Sirlp	rw	0x00	<p>红外低功耗模式 (IrDA low_power)</p> <p>1 : 低功耗模式 0 : 普通模式</p>
0	Siren	rw	0x00	<p>IrDA 红外模式使能 (IrDA mode enable)</p> <p>1 : 使能红外模式 0 : 不使能红外模式</p>

# 20 串行外设接口(SPI\_I2S)

## 20.1 SPI\_I2S 功能框图

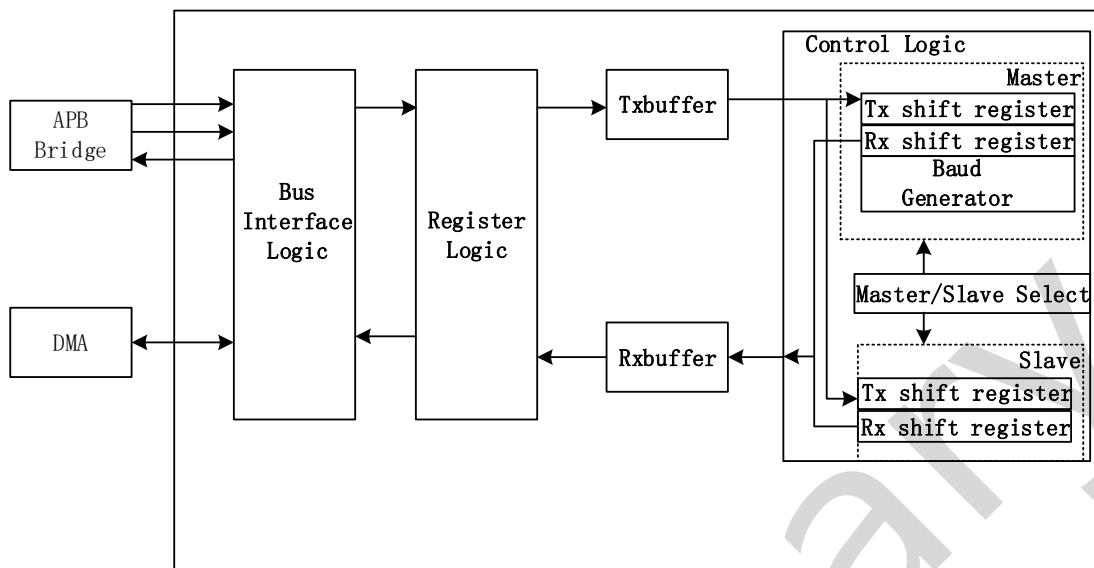


图 20-1 SPI\_I2S 功能框图

## 20.2 SPI\_I2S 简述

SPI (Serial Peripheral Interface) 接口广泛用于不同设备之间的板级通讯，如扩展串行 Flash, ADC 等。许多 IC 制造商生产的器件都支持 SPI 接口。

SPI 允许 MCU 与外部设备以全双工、同步、串行方式通信。应用软件可以通过查询状态或 SPI 中断来通信。

I2S(Inter—IC Sound)总线，又称集成电路内置音频总线，是飞利浦公司为数字音频设备之间的音频数据传输而制定的一种总线标准，该总线专门用于音频设备之间的数据传输，广泛应用于各种多媒体系统。它采用了沿独立的导线传输时钟与数据信号的设计，通过将数据和时钟信号分离，避免了因时差诱发的失真。

## 20.3 SPI 功能描述

### 20.3.1 概述

SPI 支持接收和发送 1~32 位数据同时进行。SPI 可以被配置为从模式或者在一个主机环境下配置为主模式。可以通过配置时钟极性 CPOL 和相位 CPHA 选择四种可能的时序关系。可编程的数据顺序，MSB 在前或者 LSB 在前。

发送和接收部分使用相同的时钟。数据在时钟的上升沿或者下降沿输出，在 SCLK 相反的有效沿锁存数据。因为 SPI 是用于交换数据，因此数据必须在转移结束后读取，即使数据不是有效数据。在 SPI 模式下，主机和与其通信的从机的时钟相位和极性必须相同。

通常 SPI 通过 4 个管脚与外部器件相连：

- 1) MISO: 主设备输入/从设备输出管脚。传输方向为从设备发送到主设备。
- 2) MOSI: 主设备输出/从设备输入管脚。传输方向为主设备发送到从设备。

- 3) SCK:串口时钟，主设备产生，通过该引脚传输供从设备使用。
- 4) NSS:从设备选择。这是一个可选的管脚，用来选择主/从设备。为了避免数据线上设备之间的冲突，通过设置片选管脚 NSS，使得主设备可以和某个从设备一对一的单独通信。当 NSS 引脚功能被激活后，配置作为主设备的 SPI 进入主模式，将会拉低 NSS 引脚，其余连接到主设备 NSS 的 SPI 设备由于检测到了 NSS 拉低的信号，会自动进入从设备模式。

下图例子表示的是主从设备一对互连通信。

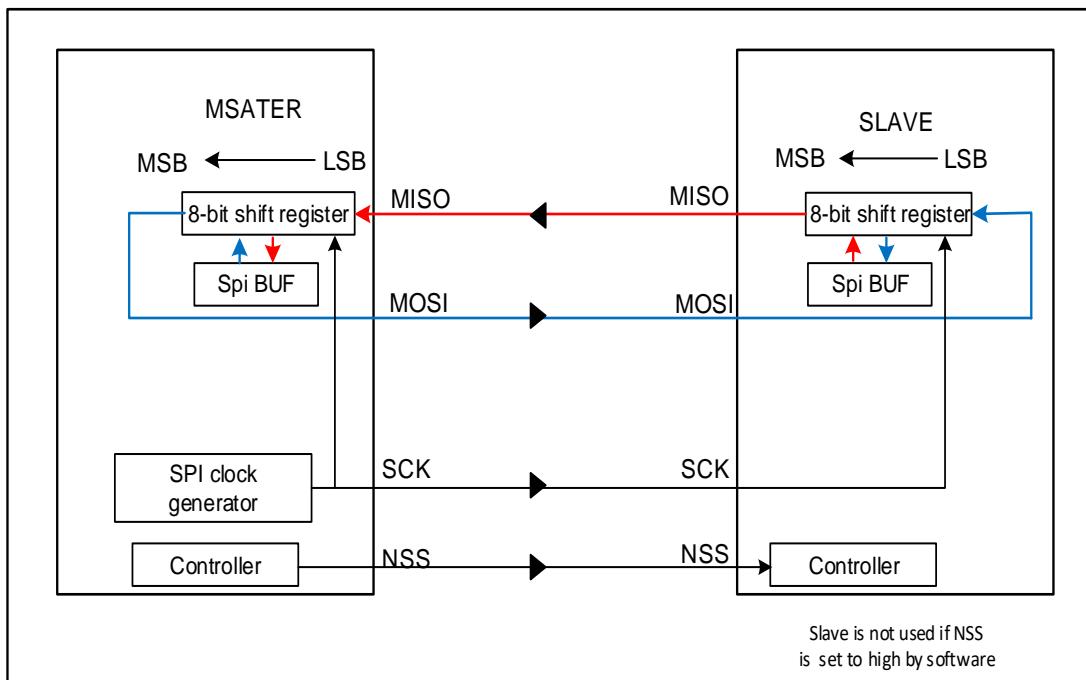


图 20-2 单主和单从应用

SPI 主从设备的同名引脚相互连接。数据传输顺序为从最高有效位到最低有效位串行传输。

主设备负责发起通信请求，从设备负责响应。MOSI 脚将来自主设备的数据输入到从设备，MISO 脚将从设备响应的数据回传给主设备。由于 SPI 的这个特性，主从设备需要使用同一时钟才能保证数据传输的可靠性。通信总是由主设备发起，因此从设备通过 SCK 脚得到主设备提供的时钟信号。

### 20.3.1.1 时钟信号的相位和极性

SPI\_I2S\_CCTL 寄存器的 CPOL 和 CPHA 位分别控制时钟的极性和时钟的相位，通过组合可以得到  $2 \times 2 = 4$  种时序关系。

时钟极性指的是 SCK 时钟空闲状态下的电平保持情况，主设备和从设备都会受到 CPOL 控制位的影响。

如果 CPOL 被配置为‘0’，在空闲状态下，SCK 时钟为持续低电平，即两次传输之间为低电平；如果 CPOL 被配置为‘1’，在空闲状态下，SCK 时钟为持续高电平，即两次传输之间为高电平。

时钟相位决定的是第一个数据位在 SCK 的第一个或第二个时钟沿被采样。

如果 CPHA(时钟相位) 位被置‘1’，第一个数据位在 SCK 时钟的第一个时钟边沿被锁存(CPOL 位为 1 即为下降沿，CPOL 位为 0 即为上升沿)，同时对被接收的第一个数据位进行采样。SPI 在传输的第一个 SCK 时钟转换时改变串行数据 (此时时钟向空闲状态的反方向变动)，在下一个边沿捕捉数据。

如果 CPHA(时钟相位) 位被清‘0’，第一个数据位在 SCK 时钟的第二个时钟边沿被锁存(CPOL 位为 1

即为上升沿，CPOL 位为 0 即为下降沿），同时对被接收的第一个数据位进行采样。SPI 在传输的第一个 SCK 时钟转换时捕捉串行数据（此时时钟向空闲状态的反方向变动），数据在下一个边沿改变。

数据在哪个时钟边沿被采样，需要根据 CPOL 和 CPHA 的组合配置共同决定。

图 20-3 列出了在 SPI 传输下的不同 CPHA 与 CPOL 位组合的所有 4 种情况。此图用来说明主设备和从设备的 SCK 脚、MISO 脚、MOSI 脚、NSS 脚在不同 CPHA、CPOL 配置下的时序关系。

时序配置需要注意以下几点：

- 1) SPI 工作时不能修改 CPOL/CPHA，修改需要关闭 SPI 的使能位 SPIEN。
- 2) 主从设备同步通信，因此双方时序配置应保持一致。
- 3) 空闲状态时 SCK 的电平情况必须和 CPOL 配置对应的极性一致(CPOL 为 1，空闲状态 SCK 保持高电平；CPOL 为 0，空闲状态 SCK 保持低电平)。

### 20.3.1.2 高速传输

针对高速传输模式下对板级延时的敏感，在 SPI\_I2S\_CCTL 寄存器中由 TXEDGE 和 RXEDGE 控制位对发送相位和接收采样进行时间调整。

- 1) 在从模式下，TXEDGE 为 1 时，发送数据立即发送到数据总线，用于高速模式时 (SPBRG<=4)；为 0 时，发送数据在一个有效时钟边沿后发送到数据总线，用于低速模式时 (SPBRG>4)。
- 2) 在主模式下，RXEDGE 为 1 时，在传输数据位的尾时钟沿采样数据(用于高速模式)；为 0 时，在传输数据位的中间采样数据。

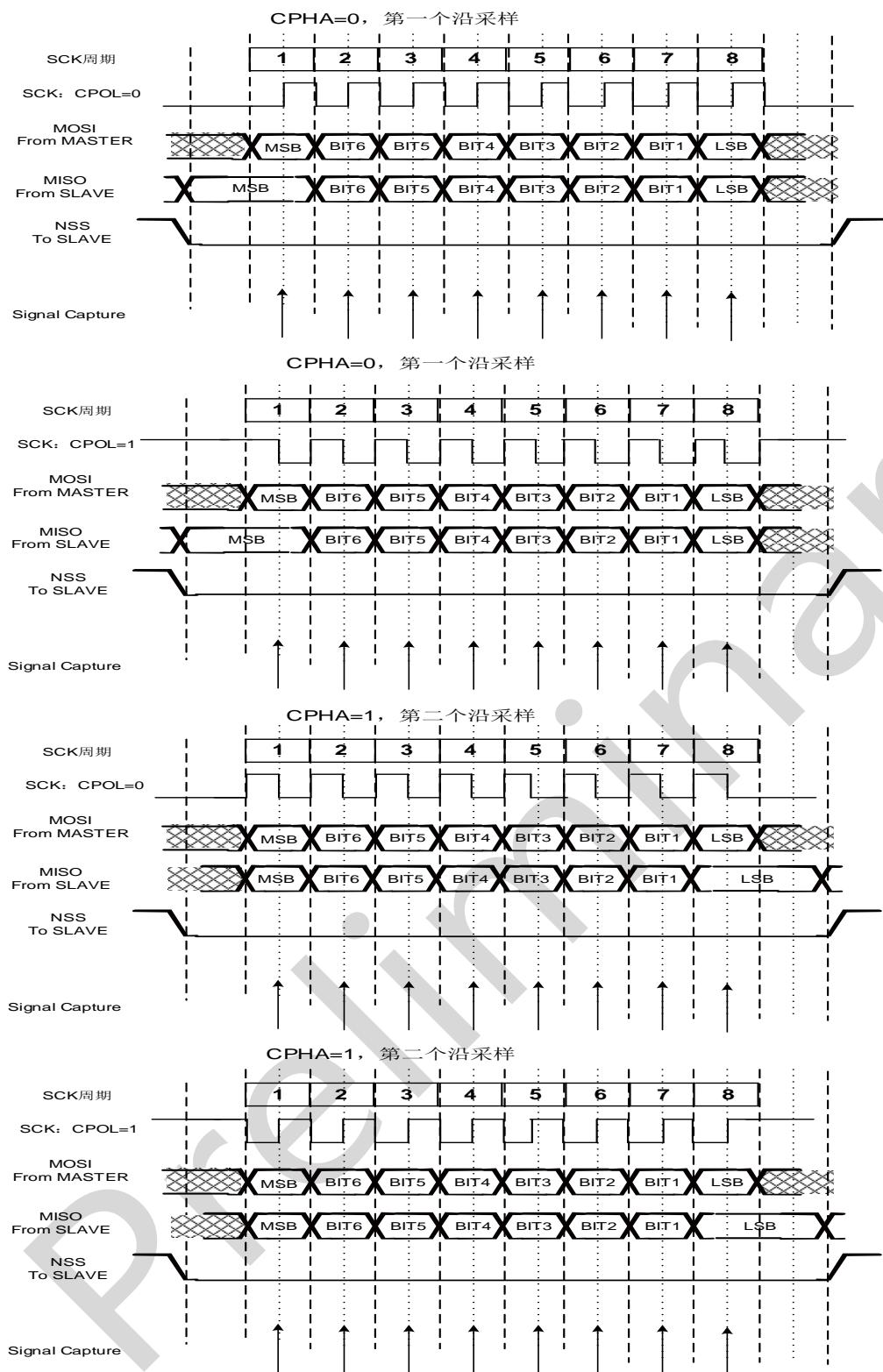


图 20-3 数据时钟时序图

### 20.3.1.3 数据帧格式

配置 SPI\_I2S\_CCTL 寄存器中的 LSBFE 位，决定数据位的输出顺序是从最低有效位到最高有效位，或者从最高有效位到最低有效位。

配置 SPI\_I2S\_CCTL 寄存器的 SPILEN 位，决定数据帧的数据长度为 7 位或者 8 位。

SPI 的发送和接收都受到数据帧格式配置的控制。

另外设置寄存器 SPI\_I2S\_EXTCTL，可以配置数据帧长度为 1~32 位。使用此配置时需要配置：

SPI\_I2S\_GCTL 寄存器的 DW8\_32 位为‘0’，且 SPI\_I2S\_CCTL 寄存器的 LSBFE 位配置为‘1’，SPILEN 位配置为‘1’。在配合 DMA 数据传输是需要将 DMA 的数据长度配置为 8bit。

## 20.3.2 SPI 主要特征

- 1) 完全兼容 Motorola 的 SPI 规格
- 2) 支持 DMA 请求
- 3) 在 3 根线上支持全双工同步传输
- 4) 16 位的可编程波特率生成器
- 5) 支持主机模式和从机模式
- 6) 8 个字节的接收/发送 FIFO
- 7) SPI 作为主机模式下 SPI 的时钟最快可高达 PCLK/2(PCLK 为 APB 时钟)，作为从机模式下 SPI 的时钟最快可高达 PCLK/4
- 8) 可编程的时钟极性和相位
- 9) 可编程的数据顺序，MSB 在前或者 LSB 在前
- 10) 支持一个主机多个从机操作
- 11) 支持 1~32 位的数据位长度同时发送和接收
- 12) 除了 8 位数据收发，其余 1~32 位数据收发只支持 LSB 模式，不支持 MSB 模式。
- 13) 支持各 8 个对应配置数据位(Data size)的发送缓冲器和接收缓冲器
  - a) 中断驱动操作
  - b) 发送缓冲为空
  - c) 发送缓冲和发送移位寄存器同时为空
  - d) 发送端下溢
  - e) 接收到有效字节
  - f) 接收缓冲上溢
  - g) 接收缓冲满
  - h) 主模式下接收到指定字节

## 20.3.3 中断

### 20.3.3.1 状态标志

为了软件操作的方便，应用程序可以通过 4 个当前状态标志和 7 个中断状态标志来监控

SPI 总线的状态。当前状态标志是只读，由硬件自动置位和清除。中断状态标志位在事件发生时置位，并在中断使能时产生 CPU 中断，由软件清除。

SPI 内部分别有一个 8 字节的发送缓冲和接收缓冲，根据 SPI\_I2S\_GCTL 寄存器的 DW8\_32 位的设置，CPU 每次可以读写 1 或者 4 个字节。根据 DW8\_32 的设置，发送和接收缓冲分别有一个字节或者一个有效数据的状态标志。

表 20-1 SPI 状态

分类	状态标志	缓冲器和信号状态
中断状态	TX_INTF	根据 DW8_32 设置, 至少有一个有效数据的空间, 能完成一次发送数据寄存器的写操作
	RX_INTF	根据 DW8_32 设置, 至少有一个有效数据的数据, 能完成一次接收数据寄存器的读操作
	UNDERRUN_INTF	发送缓冲器空且重复发送
	RXOERR_INTF	接收缓冲器非空且被覆盖
	RXMATCH_INTF	非空, 最后一个数据传送到接收缓冲中
	RXFULL_INTF	接收缓冲器满, 不能再接收新的数据
当前状态	TXEPT_INTF	发送缓冲器空, 不能再发送
	RXAVL_4BYTE	接收缓冲器有超过 4 字节有效数据
	TXFULL	发送缓冲器满
	TXEPT	发送缓冲器空
	RXAVL	接收缓冲器非空, 至少还能接收一个字节

当 SPI\_I2S\_GCTL 寄存器的 TXTLF 为 00 时, 发送缓冲器有大于等于 1 个空闲数据空间时 TX\_INTF 置位; TXTLF 为 01 时, 发送缓冲器有超过一半的空闲空间时 TX\_INTF 置位。

当 SPI\_I2S\_GCTL 寄存器的 RXTLF 为 00 时, 接收缓冲器有大于等于 1 个有效数据时, RX\_INTF 置位; RXTLF 为 01 时, 接收缓冲器有超过一半的有效数据时 RX\_INTF 置位。

### 20.3.4 DMA 传输

应用一种简单的请求应答 DMA 机制, 即时监控 SPI 缓冲器的空满状态, 通过 DMA 提高 SPI 的 TXREG 与 RXREG 的读写速率, 加快 SPI 的整体通信速度。

配置 SPI\_I2S\_GCTL 寄存器上的 DMAMODE 位, 实现 SPI 模块与 DMA 之间的信号交互。发送缓冲器有空闲空间, 则请求 DMA 写入 TXREG。接收缓冲器有有效可读数据, 则请求 DMA 读取 RXREG。

- 1) 发送时, 当 SPI\_I2S\_GCTL 寄存器的 TXTLF 为 00 时, 发送缓冲器有大于等于 1 个空闲数据空间时即进行 DMA 传输请求; TXTLF 为 01 时, 发送缓冲器有超过一半的空闲空间时即进行 DMA 请求。每次请求只进行一次 DMA 传输。每次 DMA 传输数据大小以及发送缓冲器每个数据大小由 DW8\_32 为决定。
- 2) 接收时, 当 SPI\_I2S\_GCTL 寄存器的 RXTLF 为 00 时, 接收缓冲器有大于等于 1 个有效数据时即进行 DMA 传输请求; RXTLF 为 01 时, 接收缓冲器有超过一半的有效数据时即进行 DMA 请求。每次请求只进行一次 DMA 传输。每次 DMA 传输数据大小以及接收缓冲器每个数据大小由 DW8\_32 为决定。

## 20.3.5 SPI 从模式

SPI 作为从设备时，时钟驱动来自主设备发送到 SCK 引脚上的串行时钟。因此从设备的波特率发生器不使用，相应配置寄存器 SPI\_I2S\_SPBREG 无效。

### 20.3.5.1 配置步骤

- 1) 设置 SPILEN 位以定义数据帧格式为 7 位或者 8 位。
- 2) 确定时序模式，配置 CPOL 和 CPHA 寄存器。主从设备的时序模式应保持配置一致，保证数据正常传输。
- 3) 确定帧格式数据方向，配置 LSBFE。主从设备的帧格式应保持配置一致，保证数据正常传输。
- 4) MDOE 位清零，SPIEN 位置‘1’，使能 SPI 对应 GPIO 引脚。从模式 SPI 接收 MOSI 引脚数据，并从 MISO 引脚输出回传数据。

### 20.3.5.2 数据发送过程

写数据到发送数据寄存器 TXREG，整个数据被一起传输到发送缓冲器。

当从设备收到 SCK 传来的时钟信号，同时接收到 MOSI 引脚传来的第一个数据位，从设备开始发送，第一个位会被直接发送到 MISO 引脚，而余下的 bit 位会被传输到移位寄存器，后续通过移位寄存器将数据串行发送。SPI\_I2S\_INTSTAT 寄存器里的 TX\_INTF 标志被硬件置‘1’，表示第一位已发送，其余位被传输到移位寄存器。若想产生中断，需将 SPI\_I2S\_INTEN 寄存器上的 TXIEN 位置‘1’。

### 20.3.5.3 数据接收过程

从设备的接收器完整接收 MOSI 引脚传来的数据时：

- 1) 引脚输入的数据通过移位寄存器，在最后一个采样时钟边沿后，数据字节被传输到接收缓冲器中。对应 SPI\_I2S\_INTSTAT 寄存器中的 RX\_INTF 标志被硬件置‘1’。此时读 SPI\_I2S\_RXREG，SPI 设备会返回缓冲器中存储的值。
- 2) 若想产生中断，需将 SPI\_I2S\_INTEN 寄存器中的 RXIEN 位置‘1’。

## 20.3.6 SPI 主模式

SPI 作为主设备时，输出串行时钟到 SCK 引脚上，供从设备使用。

### 20.3.6.1 配置步骤

- 1) 定义串行时钟波特率，配置 SPI\_I2S\_SPBREG 寄存器。
- 2) 确定时序模式，配置 CPOL 和 CPHA 寄存器。主从设备的时序模式应保持配置一致，保证数据正常传输。
- 3) 设置 SPILEN 位来定义 8 或 7 位数据帧格式。
- 4) 确定帧格式数据方向，配置 LSBFE。主从设备的帧格式应保持配置一致，保证数据正常传输。
- 5) 如果只接收而不发送数据，配置 SPI\_I2S\_RNDNR 寄存器，定义需要接收的字节数。
- 6) 必须设置 MDOE 和 SPIEN 位。
- 7) 主模式 SPI 从 MOSI 引脚输出数据，并接收 MISO 回传的数据。NSS 是从设备选择信号输出。

### 20.3.6.2 数据发送过程

写数据到发送数据寄存器 TXREG，数据接着被并行写入发送缓冲器，主设备开始发送。在发送第一个数据位时，整个数据被一起传输到移位寄存器，后续数据通过移位寄存器串行输出到 MOSI 引脚。通过 SPI\_I2S\_CCTL 寄存器中的 LSBFE 位决定数据串行传输顺序。SPI\_I2S\_INTSTAT 寄存器里的 TX\_INTF 标志被硬件置‘1’，表示数据已从发送缓冲器被传输到移位寄存器。若想产生中断，需将 SPI\_I2S\_INTEN 寄存器上的 TXIEN 位置‘1’。

### 20.3.6.3 数据接收过程

从设备的接收器完整接收 MISO 引脚传来的数据时：

- 1) 引脚输入的数据通过移位寄存器，在最后一个采样时钟边沿后，数据字节被传输到接收缓冲器中。对应 SPI\_I2S\_INTSTAT 寄存器中的 RX\_INTF 标志被硬件置‘1’。此时读 SPI\_I2S\_RXREG，SPI 设备会返回缓冲器中存储的值。
- 2) 若想产生中断，需将 SPI\_I2S\_INTEN 寄存器中的 RXIEN 位置‘1’。

如果只接收而不发送数据，在接收完 RXDNR 定义的字节数，RXMATCH\_INTF 位被置‘1’，表示所有的数据接收完毕，主模式下不再发送时钟信号。

### 20.3.7 波特率设置

波特率是生成的 SCLK 的频率，一般是 PCLK 的分频。BRG 是一个 16 位的波特率发生器。

SPBREG 寄存器控制 16 位计数器的计数周期。

提供期望的波特率和 fpclk (APB 模块的频率)，使用下表所示的公式计算出的值近似数赋值给 SPBRG 寄存器。其中下表中的 X 等于 SPBRG 寄存器的值 (2~65535)。

表 20-2 波特率公式

模式	公式
SPI 模式	波特率 = $f_{\text{pclk}}/X$

## 20.4 I2S 功能描述

### 20.4.1 I2S 主要特征

- 1) 半双工通信（仅发射机或接收机）
- 2) 主操作或从操作
- 3) 8 位可编程线性预分频器，以达到精确的音频采样频率 (8KHz 到 192KHz)
- 4) 数据格式可以是 16 位、24 位或 32 位
- 5) 数据包帧固定为 16 位 (16 位数据帧) 或 32 位 (16 位、24 位、32 位数据帧)
- 6) 可编程时钟极性 (稳定状态)
- 7) 发射模式下的下溢标志 (仅从机)，接收模式下的上溢标志 (主和从机) 和接收/发射模式下的帧错误标志 (仅从机)
- 8) 用于传输和接收的 32 位寄存器为两个声道分时复用
- 9) 支持 I2S 协议：
  - 飞利浦标准

- MSB 对齐标准（左对齐）
  - LSB 对齐标准（右对齐）
  - PCM 标准（在 16 位信道帧上具有短帧和长帧同步或扩展到 32 位信道帧的 16 位数据帧）
- 4) 数据方向始终是 **MSB** 优先
  - 5) DMA 传输能力（32 位宽）
  - 6) 可配置输出 **MCLK** 来驱动外部音频组件，比率固定在  $256 \times FS$ （其中 **FS** 为音频采样频率）

## 20.4.2 I2S 总线接口

I2S 与 SPI 共用三个公共管脚：

- 1) **SD**: 串行数据(映射在 **MOSI** 管脚上), 用于发送或接收两次多路数据通道(仅在半双工模式下)。
- 2) **WS**: 声道选择(映射在 **NSS** 引脚上), 是 **master** 中的数据控制信号输出模式和从模式输入。
- 3) **CK**: 串行时钟(映射在 **SCK** 引脚上), 是主模式下的串行时钟输出以及从机模式下的串行时钟输入。
- 4) 当某些外部设备需要主时钟输入时, 可以使用一个附加的管脚输出时钟到音频设备:
- 5) **MCK**: 驱动时钟(映射在 **MISO** 引脚上), 用于驱动外部音频组件, 仅主模式时使用。

## 20.4.3 数据格式

三线总线处理音频数据的线路必须经过分时复用两个声道：右声道和左声道。但是只有一个 32 位寄存器用于传输或接收。所以由软件依次配置寄存器 **TXREG** 为每个声道侧的值, 或依次读取寄存器 **RXREG** 的数据。总是先发送左声道, 然后发送右声道 (**CHSIDE** 对 PCM 协议没有意义)。

数据可采用以下格式发送：

- 1) 16 位数据打包在 16 位帧中
- 2) 16 位数据打包在 32 位帧中
- 3) 24 位数据打包在 32 位帧中
- 4) 32 位数据打包在 32 位帧中

当使用 32 位帧上发送 16 位数据时, 前 16 位 (MSB) 是有效的位, 16 位 LSB 强制为 0, 无需任何软件操作, 通过硬件实现。其他格式相似。

## 20.4.4 通信标准

对于所有数据格式和通信标准, 总是先发送最高位 (MSB 优先)。I2S 接口支持四种音频标准, 可通过配置 **SPI\_I2S\_I2SCFGR** 寄存器的 **I2SSTD[1:0]** 和 **PCMSYNC** 进行切换。

### 20.4.4.1 飞利浦标准

对于本标准, **WS** 信号用于指示正在传输的声音。发射器在 **CK** 的下降沿锁存数据, 接收器并在 **CK** 的上升读取数据。**WS** 信号也在 **CK** 的下降沿被锁定。对于这种标准 I2S 格式的信号, 无论有多少位有效数据, 数据的最高位总是出现在 **WS** 变化 (也就是一帧开始) 后的第 2 个 **CK** 脉冲处。

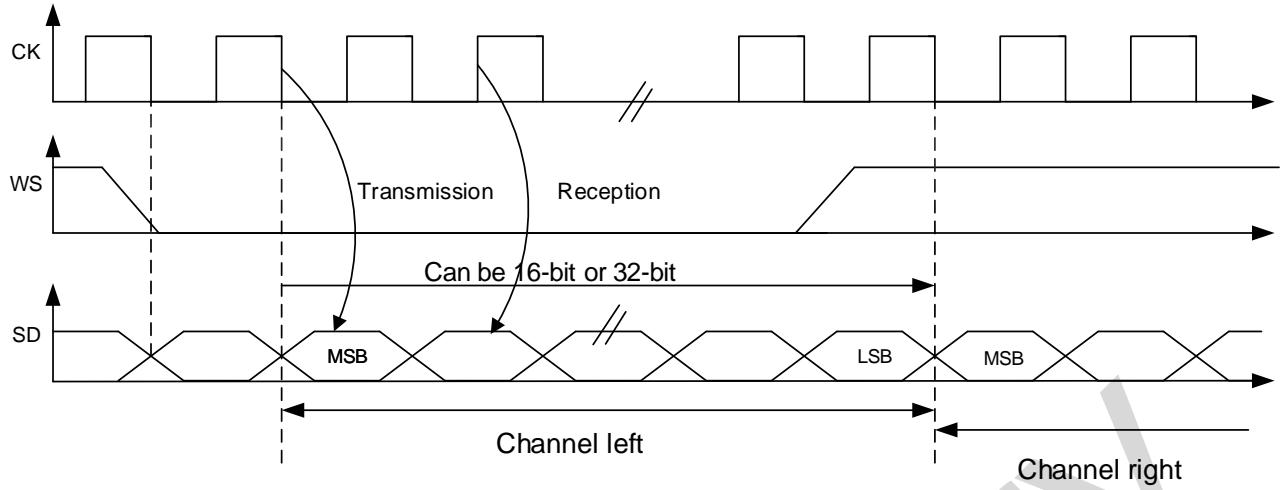


图 20-4 飞利浦标准示意图

#### 20.4.4.2 MSB 对齐标准

对于这个标准，第一个数据在 WS 变化后的第一个沿有效。

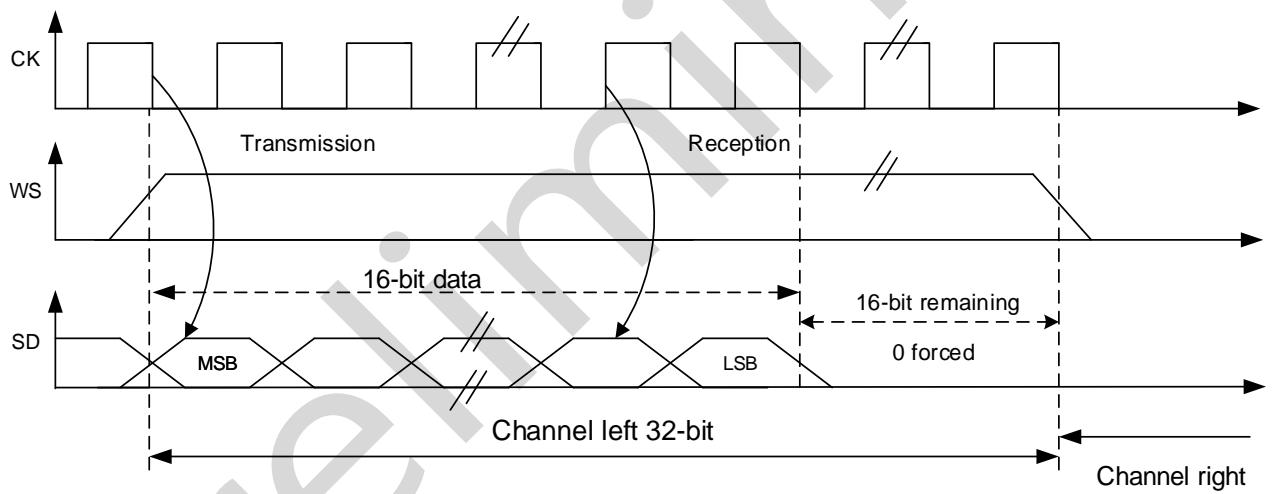


图 20-5 MSB 对齐标准示意图

#### 20.4.4.3 LSB 对齐标准

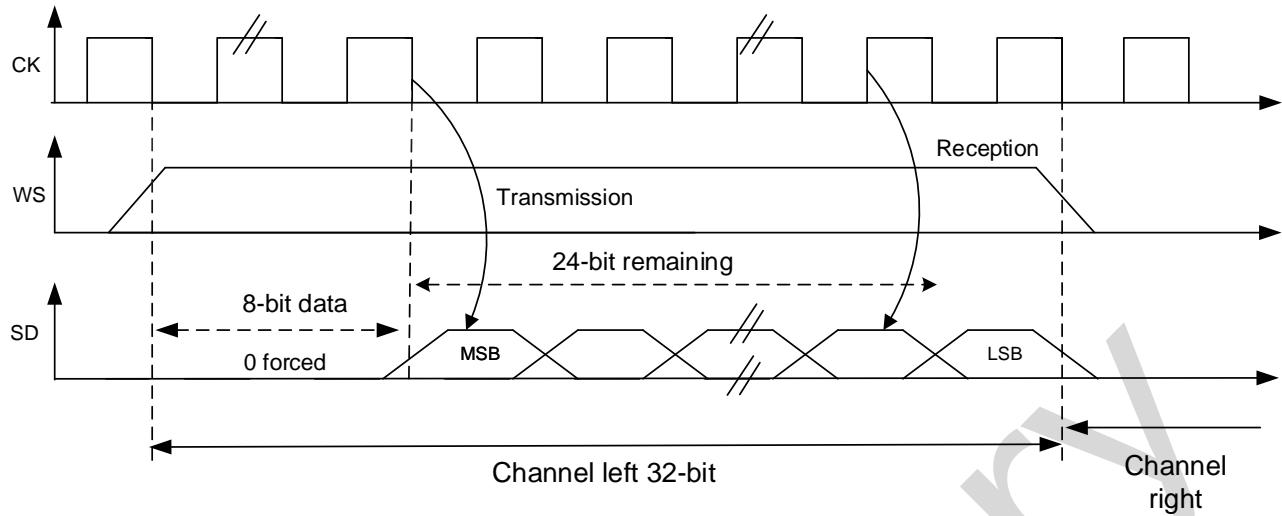


图 20-6 LSB 对齐标准示意图

#### 20.4.4.4 PCM 标准

对于 PCM 标准，不需要使用声道信息。PCM 有两个模式：短帧模式和长帧模式，通过配置 SPI\_I2S\_I2SCFGR 寄存器的 PCMSYNC 位进行切换。在 PCM 模式下，输出信号（WS，SD）在 CK 信号的上升沿进行采样。输入信号（WS，SD）在 CK 下降沿被捕获。注意在主模式下，CK 和 WS 被配置为输出。

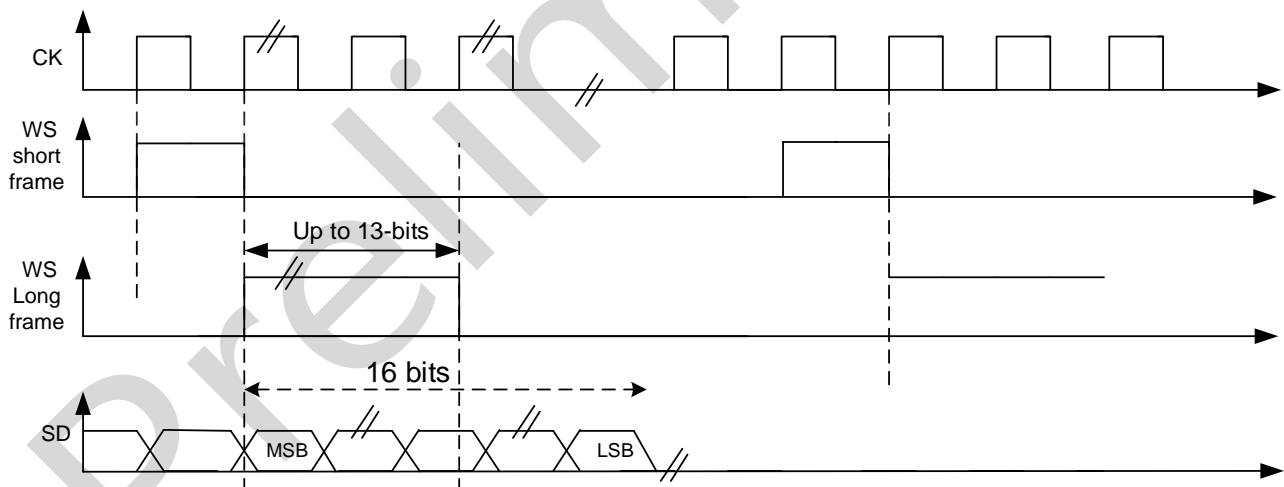


图 20-7 PCM 标准示意图

#### 20.4.5 DMA 传输

在 I2S 模式，DMA 的工作方式与 SPI 模式完全相同。

## 20.4.6 从模式

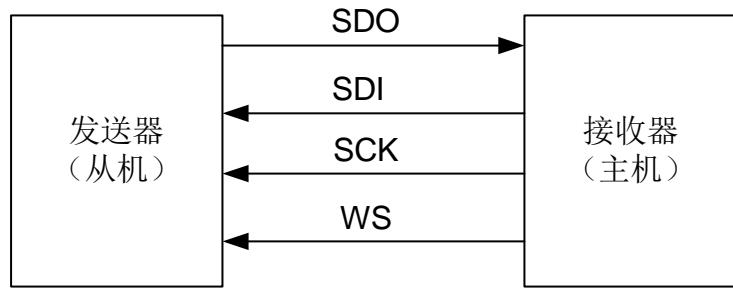


图 20-8 I2S 从模式

模式支持发送和接收，状态机和实现方法同 SPI 从模式，配置流程如下：

- 1) 开启模块使能，配置 SPI\_I2S\_GCTL.SPIEN 为 1；
- 2) 设置模块功能为主模式，配置 SPI\_I2S\_GCTL.MODE 为 0；
- 3) 设置空闲电平，配置 SPI\_I2S\_CCTL.CPOL；
- 4) 设置音频采样频率，配置 SPI\_I2S\_I2SCFGR.I2SDIV、SPI\_I2S\_I2SCFGR.DATLEN 和 SPI\_I2S\_I2SCFGR.CHLEN，计算方法参考章节 1.4.9；
- 5) 设置 I2S 传输，配置 SPI\_I2S\_I2SCFGR.SPI\_I2S；
- 6) 设置传输的通信标准，配置 SPI\_I2S\_I2SCFGR.PCMSSYNC 和 I2SCFGR.I2SSSTD；
- 7) 设置启用 DMA 传输，配置 SPI\_I2S\_GCTL.DMAMODE；
- 8) 开启传输，配置 SPI\_I2S\_GCTL.TXEN 或 SPI\_I2S\_GCTL.RXEN 为 1。

注意，从模式发情况下，需要在检测到 WS 的边沿之前将要发送的数据配置到寄存器 SPI\_I2S\_TXREG。

## 20.4.7 主模式

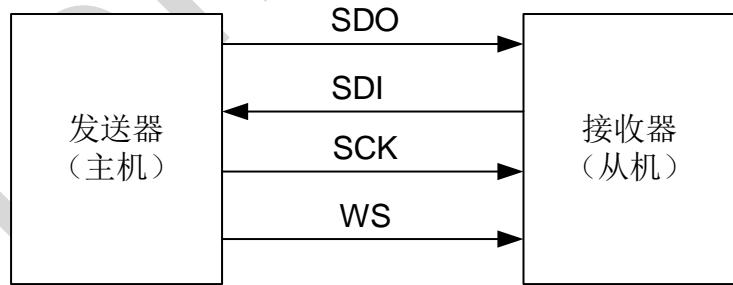


图 20-9 I2S 主模式

主模式支持发送和接收，状态机和实现方法同 SPI 主模式，配置流程如下：

- 1) 开启模块使能，配置 SPI\_I2S\_GCTL.SPIEN 为 1；
- 2) 设置模块功能为主模式，配置 SPI\_I2S\_GCTL.MODE 为 0；
- 3) 设置空闲电平，配置 SPI\_I2S\_CCTL.CPOL；
- 4) 设置是否向外部器件提供 MCK，配置 SPI\_I2S\_I2SCFGR.MCKOE；
- 5) 设置音频采样频率，配置 SPI\_I2S\_I2SCFGR.I2SDIV、SPI\_I2S\_I2SCFGR.DATLEN 和 SPI\_I2S\_I2SCFGR.CHLEN，计算方法参考章节 1.4.9；

- 6) 设置 I2S 传输，配置 SPI\_I2S\_I2SCFGR.SPI\_I2S；
- 7) 设置传输的通信标准，配置 SPI\_I2S\_I2SCFGR.PCMSSYNC 和 SPI\_I2S\_I2SCFGR.I2SSTD；
- 8) 设置启用 DMA 传输，配置 SPI\_I2S\_GCTL.DMAMODE；
- 9) 开启传输，配置 SPI\_I2S\_GCTL.TXEN 或 SPI\_I2S\_GCTL.RXEN 为 1。

其中，主模式收数据情况下，可通过配置 SPI\_I2S\_RXDNR 寄存器设置接收字节数，当接收到指定字节数时，停止传输。

## 20.4.8 中断

### 20.4.8.1 状态标志

与 SPI 共用同一个中断向量

表 20-3 I2S 状态

中断事件	中断寄存器标志
TX BUFFER 空	TXEPT
RX BUFFER 非空	RXVAL
TX BUFFER 向下溢出	UNDERRUN_IEN
RX BUFFER 向上溢出	RXOERR_INTF
帧传输错误	FRE

忙标志 (BSY): 表示 I2S 正在传输中

声道标志 (CHSIDE): 表示正在传输的声音为左声道还是右声道。

TX BUFFER 空 (TXEPT): 表示当前 BUFFER 中的数据已经被读走。

RX BUFFER 非空 (RXVAL): 表示当前 BUFFER 中的数据有效，APB 总线可读。

TX BUFFER 向下溢出 (UNDERRUN\_IEN): 表示从模式发送 BUFFER 中的数据在 I2S 总线上被传输了两次。

RX BUFFER 向上溢出 (RXOERR\_INTF): 表示上一个接收数据未被读走，且已经被覆盖。

帧传输错误 (FRE): 表示 I2S 传输协议不匹配 (仅从模式有效)。

## 20.4.9 时钟预分频器

I2SCLK 可以由外部晶振提供并在内部 PLL 倍频得到。

音频常用采样率为 192kHz, 96 kHz, 48 kHz, 44.1 kHz, 32 kHz, 22.05 kHz, 16 kHz, 11.025 kHz, 8 kHz。

$$F_{I2SCLK} \approx F_s * ((CHLEN+1)*16)*2*4*I2SDIV = F_{MCLK} * I2SDIV$$

其中，当 I2SDIV=4, CHLEN = 1 时，满足  $F_{MCLK} = 256 * F_s$ ;

当  $F_s=192\text{kHz}$ , CHLEN = 1 时， $F_{I2SCLK} = 49.125\text{MHz}$ 。

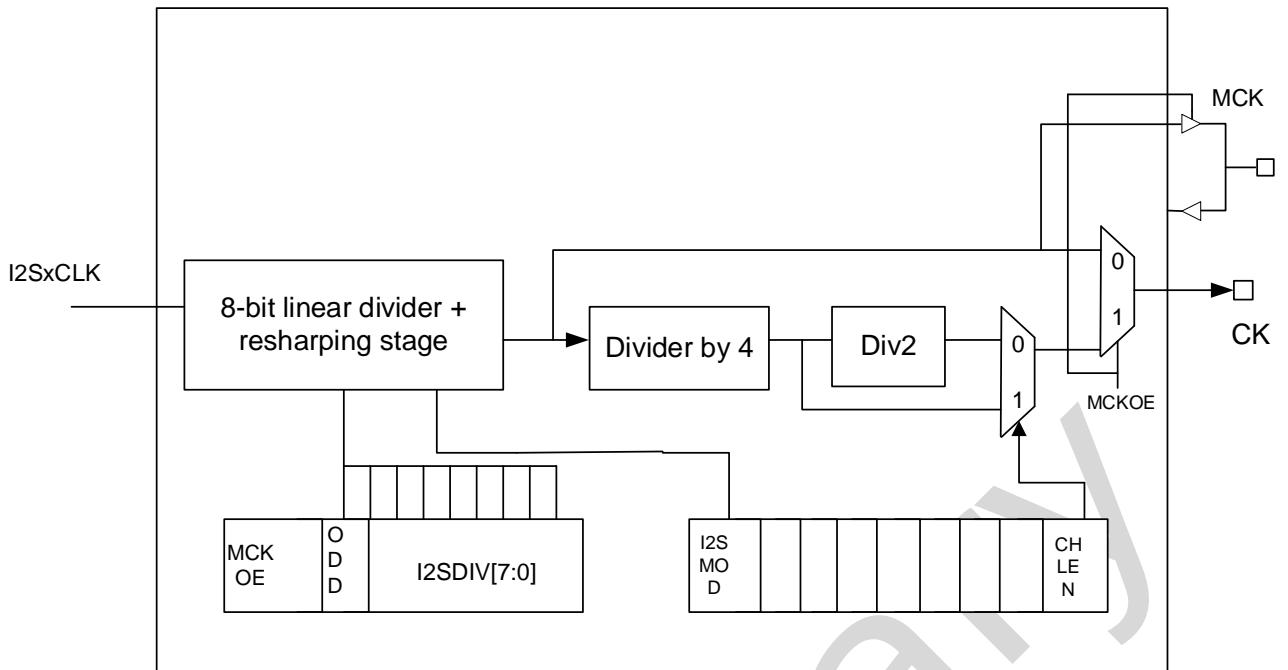


图 20-10 I2S 时钟预分频器示意图

当  $MCKOE=1$  时，芯片需要输出 MCLK 时钟，预分频器需要将 I2SCLK 分频到  $256*Fs$ ，然后再分频到 CK；

当  $MCKOE=0$  时，芯片不需要输出 MCLK 时钟，预分频器直接将 I2SCLK 分频到 CK。

按照  $F_{I2SCLK}=42MHz$  计算，音频采样率误差如下（未考虑 PLL 精度）：

表 20-4 I2S 音频采样率误差表

MCKOE=1				MCKOE=0					
FS	声道位宽	I2SDIV	I2SDIV 真实值	真实 FS	误差	I2SDIV	I2SDIV 真实值	真实 FS	误差
192000	32	0.976563	1	187500	-2.34375	3.90625	4	187500	-2.34375
192000	16	0.976563	1	187500	-2.34375	7.8125	8	187500	-2.34375
96000	32	1.953125	2	93750	-2.34375	7.8125	8	93750	-2.34375
96000	16	1.953125	2	93750	-2.34375	15.625	16	93750	-34375
48000	32	3.90625	4	46875	-2.34375	15.625	16	46875	-2.34375
48000	16	3.90625	4	46875	-2.34375	31.25	31	48387.1	0.806452
44100	32	4.251701	4	46875	6.292517	17.0068	17	44117.65	0.040016
44100	16	4.251701	4	46875	6.292517	34.01361	34	44117.65	0.040016
32000	32	5.859375	6	31250	-2.34375	23.4375	23	32608.7	1.902174
32000	16	5.859375	6	31250	-2.34375	46.875	47	31914.89	-0.26596
22050	32	8.503401	9	20833.33333	-5.51776	34.01361	34	22058.82	0.040016
22050	16	8.503401	9	20833.33333	-5.51776	68.02721	68	22058.82	0.040016

MCKOE=1					MCKOE=0				
16000	32	11.71875	12	15625	-2.34375	46.875	47	15957.45	-0.26596
16000	16	11.71875	12	15625	-2.34375	93.75	94	15957.45	-0.26596
11025	32	17.0068	17	11029.41176	0.040016	68.02721	68	11029.41	0.040016
11025	16	17.0068	17	11029.41176	0.040016	136.0544	136	44029.41	0.040016
8000	32	23.4375	23	8152.173913	1.902174	93.75	94	7978.723	-0.26596
8000	16	23.4375	23	8152.173913	1.902174	187.5	188	7978.723	-0.26596

## 20.5 寄存器堆和存储器映射描述

表 20-5 SPI\_I2S 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	SPI_I2S_TXREG	发送数据寄存器	0x00000000
0x04	SPI_I2S_RXREG	接收数据寄存器	0x00000000
0x08	SPI_I2S_CSTAT	当前状态寄存器	0x00000001
0x0C	SPI_I2S_INTSTAT	中断状态寄存器	0x00000000
0x10	SPI_I2S_INTEN	中断使能寄存器	0x00000000
0x14	SPI_I2S_INTCLR	中断清除寄存器	0x00000000
0x18	SPI_I2S_GCTL	全局控制寄存器	0x00000004
0x1C	SPI_I2S_CCTL	通用控制寄存器	0x00000008
0x20	SPI_I2S_SPBRG	波特率发生器	0x00000002
0x24	SPI_I2S_RXDNR	接收数据个数寄存器	0x00000001
0x28	SPI_I2S_NSSR	从机片选寄存器	0x000000FF
0x2C	SPI_I2S_EXTCTL	数据控制寄存器	0x00000008
0x30	SPI_I2S_I2SCFGR	I2S 配置寄存器	0x00010000

### 20.5.1 发送数据寄存器(SPI\_I2S\_TXREG)

偏移地址:0x00

复位值:0x0000 0000

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

Bit	Field	Type	Reset	Description
31:0	TXREG	rw	0x0000 0000	发送数据寄存器(Transmit data register) 有效数据位由 DW8_32 控制。 0:只有低 8 位有效 1:TXREG[31:0]都有效

## 20.5.2 接收数据寄存器(SPI\_I2S\_RXREG)

偏移地址:0x04

复位值:0x0000 0000

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

Bit	Field	Type	Reset	Description
31:0	RXREG	r	0x0000 0000	接收数据寄存器 (Receive data register) 有效数据位由 DW8_32 控制。 0:只有低 8 位有效 1:RXREG[31:0]都有效 该寄存器可读不可写。

## 20.5.3 当前状态寄存器(SPI\_I2S\_CSTAT)

偏移地址:0x08

复位值:0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15:14	13	12	11:8	7:4		3		2		1		0			
Field	Reser ved	CHSI DE	BUSY	RXFA DDR	TXFADDR		RXAVL_4BYTE		TXFULL		RXAVL		TXEPT			

Type		r	r	r	r	r	r	r	r	r	r
------	--	---	---	---	---	---	---	---	---	---	---

Bit	Field	Type	Reset	Description
31:14	Reserved			始终读为 0。
13	CHSIDE	r	0x01	<p>声道标志 0 表示正在传输的声音为左声道 1 表示正在传输的声音为右声道。 注意： SPI 模式不可用， pcm 模式无意义</p>
12	BUSY	r	0x00	忙标志： 表示 I2S 或 SPI 正在传输中
11:8	RXFADDR	r	0x00	当前接收缓冲器中有效数据个数
7:4	TXFADDR	r	0x00	当前发送缓冲器中有效数据个数
3	RXAVL_4BYTE	r	0x00	<p>接收缓冲器中有效数据达到 4 个字节标志位(Receive available 4 byte data message) 1: 接收缓冲器中有超过 4 个字节 0: 接收缓冲器中数据小于 4 个字节 工作在 I2S 模式下时，该位在接收到一个声道数据后置位。(如 CHLEN=0，在接收到 16bit 后该位将置位)</p>
2	TXFULL	r	0x00	<p>发送缓冲器满标志位(Transmitter FIFO full status bit) 1: 发送缓冲器满 0: 发送缓冲器未满</p>
1	RXAVL	r	0x00	<p>接收有效字节数据信息位(Receive available byte data message) 当接收端缓冲器接收了一个完整字节的数据时置位该位。 1: 接收端缓冲器已经接收了一个有效字节数据 0: 接收端缓冲器空 该位只读，由硬件自动置位和清除。</p>
0	TXEPT	r	0x01	<p>发送端空位(Transmitter empty bit) 1: 发送端缓冲器和发送移位寄存器为空 0: 发送端不为空 该位只读，由硬件自动置位和清除。</p>

## 20.5.4 中断状态寄存器（SPI\_I2S\_INTSTAT）

偏移地址:0x0C

复位值:0x0000 0000

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

Field	Reserved								
Type									
Bit	15:8	7	6	5	4	3	2	1	0
Field	Res erve d	FRE_ INTF	TXEPT_ INTF	RXFULL_ INTF	RX MATCH_ INTF	RXOERR _INTF	UNDERR UN_ INTF	RX_INTF	TX_INTF
Type		r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0
7	FRE_INTF	r	0x00	帧传输错误: 表示 I2S 传输协议不匹配 (仅从模式有效)
6	TXEPT_INTF	r	0x00	发送端空中断标志位(Transmitter empty interrupt flag bit) 硬件自动置位, 写 INTCLR 寄存器 TXEPT_ICLR 位清除。 1: 发送端缓冲器和 TX 移位寄存器为空 0: 发送端缓冲器和 TX 移位寄存器不为空 注意:该位是中断状态信号, TXEPT 是状态信号。
5	RXFULL_INTF	r	0x00	接收端缓冲器满中断标志位(RX FIFO full interrupt flag bit) 硬件自动置位, 写 INTCLR 寄存器 RXFULL_ICLR 位清除。 1: RX 缓冲器满 0: RX 缓冲器未满
4	RXMATCH_INTF	r	0x00	接收指定字节数中断标志位(Receive data match the RXDNR number, the receive process will be completed and generate the interrupt) 硬件自动置位, 写 INTCLR 寄存器 RXMATCH_ICLR 位清除。 1: 接收了 RXDNR 寄存器指定的字节数 0: 未完成 RXDNR 寄存器指定的字节数

Bit	Field	Type	Reset	Description
3	RXOERR_INTF	r	0x00	<p>接收端溢出错误中断标志位(Receive overrun error interrupt flag bit)</p> <p>硬件自动置位, 写 INTCLR 寄存器 RXOERR_ICLR 位清除。</p> <p>1: 溢出错误 0: 没有溢出错误</p>
2	UNDERRUN_INTF	r	0x00	<p>SPI 从机模式下溢标志位(SPI underrun interrupt flag bit)</p> <p>硬件自动置位, 写 INTCLR 寄存器 UNDERRUN_ICLR 位清除。</p> <p>1: 下溢错误 0: 没有下溢错误</p>
1	RX_INTF	r	0x00	<p>接收端数据有效中断标志位(Receive data available interrupt flag bit)</p> <p>硬件自动置位, 写 INTCLR 寄存器 RX_ICLR 位清除。当接收端缓冲器接收了一个完整字节数据。</p> <p>1: 接收端缓冲器有有效字节数据 0: 接收端缓冲器空</p>
0	TX_INTF	r	0x00	<p>发送缓冲器有效中断标志位(发送了一个字节的数据)(Transmit FIFO available interrupt flag bit)</p> <p>硬件自动置位, 发送缓冲器不为空自动清零。</p> <p>1: 发送端缓冲器有效 0: 发送端缓冲器无效</p>

## 20.5.5 中断使能寄存器(SPI\_I2S\_INTEN)

偏移地址:0x10

复位值:0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15:8	7	6	5	4	3	2	1	0							
Field	Reser ved	FRE_ IEN	TXEPT_ IEN	RXFULL_ IEN	RX MATCH_ IEN	RXOERR _IEN	UNDERRU N_IEN	RX_IEN	TX_IEN							
Type		rw	rw	rw	rw	rw	rw	rw	rw							

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0
7	FRE_IEN	rw	0x00	帧传输错误中断使能 (FRE_IEN): 1: 中断使能 0: 中断禁止
6	TXEPT_IEN	rw	0x00	发送端空中断使能位(Transmit empty interrupt enable bit) 1: 中断使能 0: 禁止中断
5	RXFULL_IEN	rw	0x00	接收端缓冲器满中断使能位(Receive FIFO full interrupt enable bit) 1: 中断使能 0: 禁止中断
4	RXMATCH_IEN	rw	0x00	接收指定字节数中断使能位(Receive data complete interrupt enable bit) 1: 中断使能 0: 禁止中断
3	RXOERR_IEN	rw	0x00	接收端溢出错误中断使能位(Overrun error interrupt enable bit) 1: 中断使能 0: 禁止中断
2	UNDERRUN_IEN	rw	0x00	SPI 从机模式下溢中断使能位 (SPI 从机模式)(Transmitterunderrun interrupt enable bit(SPI slave mode only)) 1: 中断使能 0: 禁止中断
1	RX_IEN	rw	0x00	接收端数据中断使能位(Receive FIFO interrupt enable bit) 1: 中断使能 0: 禁止中断
0	TX_IEN	rw	0x00	发送缓冲器空中断使能位(Transmit FIFO empty interrupt enable bit) 1: 中断使能 0: 禁止中断

## 20.5.6 中断清除寄存器(SPI\_I2S\_INTCLR)

偏移地址:0x14

复位值:0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15:8	7	6	5	4	3	2	1	0							
Field	Res erve d	FRE_ ICLR	TXEPT_ ICLR	RXFULL_ ICLR	RX MATCH_ ICLR	RXOERR _ICLR	UNDERR UN_ICLR	RX_ICLR	TX_ICLR							
Type		w	w	w	w	w	w	w	w							

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0
7	FRE_ICLR	w	0x00	帧传输错误中断清除: 1: 写 1 清除中断 0: 写 0 无意义;
6	TXEPT_ICLR	w	0x00	发送端空中断清除位(Transmitter empty interrupt clear bit) 1: 中断清除 0: 中断没有清除
5	RXFULL_ICLR	w	0x00	接收端缓冲器满中断清除位(Receiver buffer full interrupt clear bit) 1: 中断清除 0: 中断没有清除
4	RXMATCH_ICLR	w	0x00	接收指定字节数中断清除位(Receive completed interrupt clear bit) 1: 中断清除 0: 中断没有清除
3	RXOERR_ICLR	w	0x00	接收端溢出错误中断清除位(Overrun error interrupt clear bit) 1: 中断清除 0: 中断没有清除

Bit	Field	Type	Reset	Description
2	UNDERRUN_ICLR	w	0x00	SPI 从机模式下溢中断清除位(SPI 从机模式)(Transmitter underrun interrupt clear bit(SPI slave mode only)) 1: 中断清除 0: 中断没有清除
1	RX_ICLR	w	0x00	接收端数据中断清除位(Receive interrupt clear bit) 1: 中断清除 0: 中断没有清除
0	TX_ICLR	w	0x00	发送缓冲器空中断清除位(Transmitter FIFO empty interrupt clear bit) 1: 中断清除 0: 中断没有清除

## 20.5.7 全局控制寄存器(SPI\_I2S\_GCTL)

偏移地址:0x18

复位值:0x0000 0004

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															PAD_SEL
Type																rw
Bit	15:13	12	11	10	9	8	7	6	5	4	3	2		1	0	
Field	PAD_SEL	NSS TOG	DW8_ 32	NSS	DMA MODE	TXTLF	RXTLF	RX EN	TX EN	MODE	INT EN	SPI EN				
Type	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw	rw	

Bit	Field	Type	Reset	Description
31:18	Reserved			保留, 始终读为 0
17:13	PAD_SEL	rw	0x00	pad0, pad1, pad2, pad3 分别对应 ioassignment 中的 SCK、MOSI、NSS、MISO 管脚。 信号 scl、mosi、ssn、miso 和 PAD 的映射变换如下： pad_sel      scl      mosi      ssn      miso 0              pad0     pad1     pad2     pad3 1              pad0     pad1     pad3     pad2 2              pad0     pad2     pad1     pad3

Bit	Field	Type	Reset	Description
				3 pad0 pad2 pad3 pad1 4 pad0 pad3 pad1 pad2 5 pad0 pad3 pad2 pad1 6 pad1 pad0 pad2 pad3 7 pad1 pad0 pad3 pad2 8 pad1 pad2 pad0 pad3 9 pad1 pad2 pad3 pad0 10 pad1 pad3 pad0 pad2 11 pad1 pad3 pad2 pad0 12 pad2 pad0 pad1 pad3 13 pad2 pad0 pad3 pad1 14 pad2 pad1 pad0 pad3 15 pad2 pad1 pad3 pad0 16 pad2 pad3 pad0 pad1 17 pad2 pad3 pad1 pad0 18 pad3 pad0 pad1 pad2 19 pad3 pad0 pad2 pad1 20 pad3 pad1 pad0 pad2 21 pad3 pad1 pad2 pad0 22 pad3 pad2 pad0 pad1 23 pad3 pad2 pad1 pad0
12	NSSTOG	rw	0x00	从设备选择信号自动翻转(Slave select toggle) 1:NSS 信号在传输完每个数据后自动翻转 0:NSS 信号不翻转 注:该位只在主模式下有效
11	DW8_32	rw	0x00	发送和接收数据寄存器有效数据选择(Valid byte or double-word data select signal) 0:只有低 8 位有效 1:32 位数据都有效 注意: i2s 模式下固定为 1
10	NSS	rw	0x00	硬件或软件控制主模式下的 NSS 输出(NSS select signal that from software or hardware) 0:由 NSSR 寄存器值控制

Bit	Field	Type	Reset	Description
				<p>1:进行数据传输时硬件自动控制 注意：i2s 模式下固定为 0</p>
9	DMAMODE	rw	0x00	<p>DMA 方式 (DMA Mode selection bit) 0: 正常模式 1: 开启 DMA 模式</p>
8:7	TXTLF	rw	0x00	<p>发送缓冲器触发 DMA 请求的边沿选择(TX FIFO trigger level bit) 00:发送缓冲器有大于等于 1 个空闲数据空间时即进行 DMA 请求或发送中断请求 01:发送缓冲器有超过一半的空闲空间时即进行 DMA 请求或发送中断请求 1x:保留 注:当 DW8_32 为 0 时，一个数据空间代表 1 个字节；为 1 时，一个有效数据代表 4 字节。</p>
6:5	RXTLF	rw	0x00	<p>接收缓冲器触发 DMA 请求的边沿选择(RX FIFO trigger level bit) 00:接收缓冲器有大于等于 1 个有效数据时即进行 DMA 请求或接收中断请求 01:接收缓冲器有超过一半的有效数据时即进行 DMA 请求或发送中断请求 1x:保留 注:当 DW8_32 为 0 时，一个数据空间代表 1 个字节；为 1 时，一个有效数据代表 4 字节。</p>
4	RXEN	rw	0x00	<p>接收使能位(Receive enable bit) 1: 接收使能 0: 接收禁止。同时可以清空 RX 缓冲器 注意:当 SPI 只工作在主机接收模式时，txen 必须设置为 0。</p>
3	TXEN	rw	0x00	<p>发送使能位(Transmit enable bit) 1: 发送使能 0: 发送禁止。同时可以清空 TX 缓冲器 注意:当在主机模式下发送和接收同时发生。</p>
2	MODE	rw	0x01	<p>主机模式位(Master mode bit) 1: 主机模式(由内部 BRG 产生串行时钟) 0: 从机模式(串行时钟来自外部主机)</p>

Bit	Field	Type	Reset	Description
1	INTEN	rw	0x00	SPI/I2S 中断使能位(SPI/I2S interrupt enable bit) 1: 使能 SPI/I2S 中断 0: 禁止 SPI/I2S 中断
0	SPIEN	rw	0x00	SPI/I2S 选择位(SPI/I2S select bit) 0 : SPI/I2S 禁止(复位状态) 1 : SPI/I2S 使能

## 20.5.8 通用控制寄存器(SPI\_I2S\_CCTL)

偏移地址:0x1C

复位值:0x0000 0008

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15:7	6	5	4	3	2	1	0								
Field	Res.	CPHASEL	TXEDGE	RXEDGE	SPILEN	LSBFE	CPOL	CPHA								
Type		rw	rw	rw	rw	rw	rw	rw								

Bit	Field	Type	Reset	Description
31:7	Reserved			保留, 始终读为 0
6	CPHASEL	rw	0x00	CPHA 极性取反选择(CPHA polarity invert select) 1 : 将 CPHA 设置值取反。 CPHA 为 1 时, 第一个数据位采样从第二个时钟边沿开始。 CPHA 为 0 时, 第一个数据位采样从第一个时钟边沿开始。 0 : CPHA 设置保持不变。 注意: I2S 模式下固定为 1
5	TXEDGE	rw	0x00	发送数据相位调整位(从模式)(Transmit data edge select) 1: 发送数据立即发送到数据总线 可用于高速模式时(SPBRG = 4)。 0: 发送数据在一个有效时钟边沿后发送到数据总线 可用于低速模式时(SPBRG > 4)。 建议该位配置为 1, 以免使用时不满足使用要求, 导致发送数据不正确

Bit	Field	Type	Reset	Description
4	RXEDGE	rw	0x00	<p>接收数据采样时钟沿选择位(主模式)(Receive data edge select)</p> <p>1: 在传输数据位的尾时钟沿采样数据 (用于高速模式) 0: 在传输数据位的中间采样数据</p> <p>建议该位配置为 1, 以免使用时不满足使用要求, 导致接收数据不正确</p>
3	SPILEN	rw	0x01	<p>SPI 数据宽度位(SPI character length bit)</p> <p>该位在 DW8_32 置位后(DW8_32=0) 配置后起作用。</p> <p>1: 8 位数据(缺省) 0: 7 位数据</p> <p>注意: I2S 模式下固定为 1</p>
2	LSBFE	rw	0x00	<p>LSBFE:LSB 在前使能位(LSB first enable bit)</p> <p>1: 数据传输或接收最低位在前 0: 数据传输或接收最高位在前</p> <p>注意: I2S 模式下固定为 0</p>
1	CPOL	rw	0x00	<p>时钟极性标志位(Clock polarity select bit)</p> <p>1: 时钟在空闲状态为高电平(两次传输之间) 0: 时钟在空闲状态为低电平(两次传输之间)</p>
0	CPHA	rw	0x00	<p>时钟相位选择位(Clock phase select bit)</p> <p>1: 第一个数据位采样从第一个时钟边沿开始 0: 第一个数据位采样从第二个时钟边沿开始</p> <p>注意: I2S 模式下固定为 0</p>

## 20.5.9 波特率发生器 (SPI\_I2S\_SPBRG)

偏移地址:0x20

复位值:0x0000 0002

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

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0
15:0	SPBRG	rw	0x0002	<p>SPI 波特率控制寄存器用于产生波特率(SPI baud rate control register for baud rate)</p> <p>波特率公式:</p> $\text{波特率} = \text{fpclk}/\text{SPBRG}$ <p>(fpclk 是 APB 时钟频率)</p> <p>注意:不要往该寄存器写 0 和 1。</p>

### 20.5.10 接收数据个数寄存器(SPI\_I2S\_RXDNR)

偏移地址:0x24

复位值:0x0000 0001

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

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0
15:0	RXDNR	rw	0x0001	<p>该寄存器用于存储下次接收过程需要接收字节的个数(The register is used to hold a count of to be received bytes in next receive process)</p> <p>该寄存器的值在 SPI 为主机接收模式下有效。缺省值是 1。</p> <p>该寄存器值通过 MCU 写值改变。</p> <p>注意:不要往该寄存器写'0'值。</p>

### 20.5.11 从机片选寄存器(SPI\_I2S\_NSSR)

偏移地址:0x28

复位值:0x0000 00FF

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

Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved										NSS					
Type											rw					

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0
7:0	NSS	rw	0xFF	主模式下片选输出信号。低有效, 从模式下该位无效(Chip select output signal in Master mode). 0:从器件被选中 1:从器件未选中

### 20.5.12 数据控制寄存器(SPI\_I2S\_EXTCTL)

偏移地址:0x2C

复位值:0x0000 0008

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

Bit	Field	Type	Reset	Description
31:5	Reserved			保留, 始终读为 0
4:0	EXTLEN	rw	0x08	<p>控制 SPI 数据长度</p> <p>0 0000:32 bit</p> <p>0 0001:1 bit</p> <p>0 0010:2 bit</p> <p>0 0011:3 bit</p> <p>.....</p> <p>1 1100:28 bit</p> <p>1 1101:29 bit</p> <p>1 1110:30 bit</p> <p>1 1111:31 bit</p> <p>注意: 仅当 SPI_I2S_GCTL 寄存器 DW8_32 位为 '1' 时有效</p> <p>I2S 模式下配置无效:</p> <p>当 CHLEN=1, EXTLEN =0x0;</p> <p>当 CHLEN=0, EXTLEN =0x10000。</p>

### 20.5.13 I2S 配置寄存器(SPI\_I2S\_I2SCFGR)

偏移地址:0x30

复位值:0x00010000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16										
Field	Reserved								I2SDIV																	
Type									rw																	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0										
Field	Reserved				MC KO E	SPI _I2 S	Reserved			PC MS YN	I2SSTD		Reser ved	DATLE N		CHL EN										
Type					rw					rw																
Bit	Field		Type	Reset	Description																					
31:25	Reserved				保留, 始终读为 0																					
24:16	I2SDIV		rw	0x1	<p>I2S 预分频器的分频系数, master 模式下配置有效</p> <p>0 为无意义, 不可配置;</p> <p>当 MCKOE=0 时, 不可配置为 1 (不支持 1 分频)</p>																					

15:12	Reserved			保留, 始终读为 0
11	MCKOE	rw	0x0	主时钟输出使能 0: 主时钟输出禁止 1: 主时钟输出使能
10	SPI_I2S	rw	0x0	模块功能选择 0: 启用 SPI 功能; 1: 启用 I2S 功能。
9:7	Reserved			保留, 始终读为 0
6	PCMSYNC	rw	0x0	PCM 标准帧同步模式 0: 短帧同步 1: 长帧同步
5:4	I2SSTD	rw	0x0	I2S 标准 00: Philips 标准 01: MSB 对齐标准 10: LSB 对齐标准 11: PCM 标准
3	Reserved			保留, 始终读为 0
2:1	DATLEN	rw	0x0	数据长度 00:16 位宽 01:24 位宽 10:32 位宽 11: 禁止设置 注意: 当 CHLEN = 0 ,DATLEN = 00.
0	CHLEN	rw	0x0	声道长度 0 : 16 位宽 1 : 32 位宽

# 21 集成电路 I2C 接口(I2C)

## 21.1 I2C 简介

微控制器通过 I2C 总线接口实现芯片间串行互联。所有 I2C 总线特定的时序、协议、仲裁和定时，都可以通过 I2C 提供的多主机功能来控制。

I2C 总线是一个两线串行接口，其中两线位串行数据（SDA）和串行时钟（SCL）线在连接到总线的器件间传递信息。每个器件都有一个唯一的地址识别，而且都可以作为一个发送或接收器。除了发送器和接收器外，器件在执行数据传输时也可以被看作是主机或者从机。主机是初始化总线的数据传输并产生允许传输的时钟信号的器件。此时，任何被寻址的器件都被认为是从机。

I2C 有两种工作速率模式可供选择：标准模式（数据传输速率为 0 ~ 100 Kbps），快速模式（数据传输速率最大为 400 Kbps）。

## 21.2 I2C 功能框图

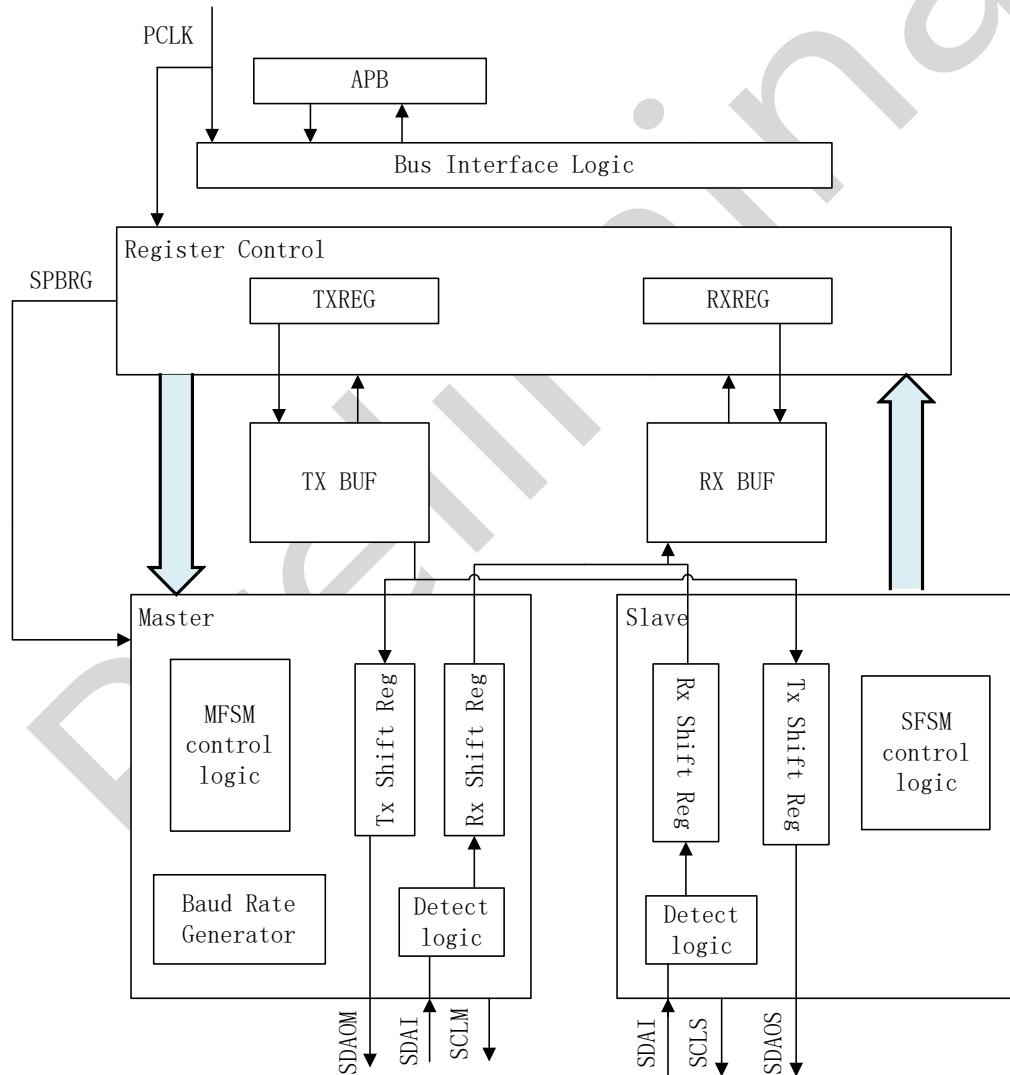


图 21-1 I2C 功能框图

## 21.3 I2C 主要特征

- 1) I2C 总线协议转换器/并行总线
- 2) 半双工同步操作
- 3) 支持主从模式
- 4) 支持 7 位地址和 10 位地址
- 5) 支持标准模式 100Kbps, 快速模式 400Kbps
- 6) 产生 Start、Stop、重新发 Start 和应答 Acknowledge 信号检测
- 7) 在主模式下只支持一个主机
- 8) 分别有 2 字节的发送和接收缓冲
- 9) 在 SCLI 和 SDAI 上增加了无毛刺电路
- 10) 支持 DMA 操作
- 11) 支持中断和查询操作
- 12) 支持 I2C 从机多地址功能（详细见寄存器描述）

## 21.4 引脚定义及内部信号

### 21.5 中断

下表列出了 I2C 的中断位以及它们的设置和清除方式。部分位由硬件置位并由软件清除；另一部分位由硬件置位和清除。

表 21-1 中断位的置位和清除

中断位	硬件置位/软件清除	硬件置位和清除
GEN_CALL	√	✗
START_DET	√	✗
STOP_DET	√	✗
ACTIVITY	√	✗
RX_DONE	√	✗
TX_ABRT	√	✗
RD_REQ	√	✗
TX_EMPTY	✗	√
TX_OVER	√	✗
RX_FULL	✗	√
RX_OVER	√	✗
RX_UNDER	√	✗

下图描述了中断寄存器中，中断位被硬件置位和软件清除的操作。

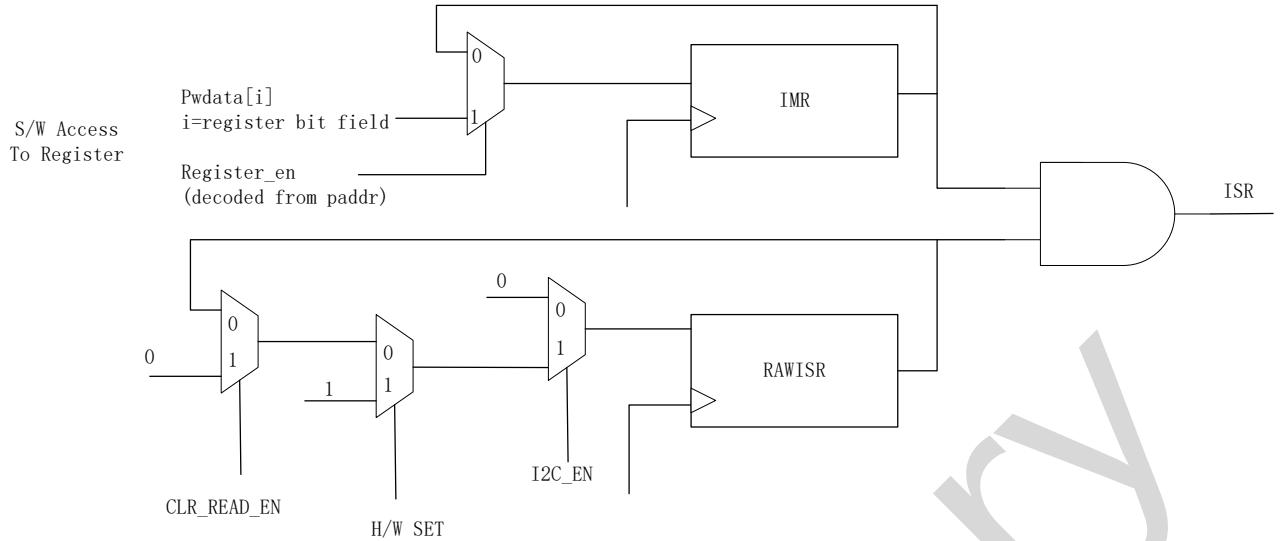


图 21-2 I2C 中断机制

## 21.6 DMA 通信

I2C 接口支持用 DMA 来发送和接收数据。通过设置 DMA 寄存器中的对应位可以单独开启 DMA 发送或者 DMA 接收。发送时数据寄存器变空或接收时数据寄存器变满，则产生 DMA 请求。DMA 请求必须在当前字节传输结束之前被响应。

### 21.6.1 利用 DMA 发送

通过设置 DMA 寄存器的 TXEN 位可以激活 DMA 发送模式。为 I2C 分配好 DMA 通道后，当发送数据时，DMA 控制器会将数据从预置的存储区装载进 DR 寄存器。

### 21.6.2 利用 DMA 接收

通过设置 DMA 寄存器的 RXEN 位可以激活 DMA 接收模式。为 I2C 分配好 DMA 通道后，当每次接收到数据字节时，DMA 控制器会将数据从 DR 寄存器中传送到预置的存储区。

## 21.7 I2C 协议

### 21.7.1 起始和停止条件

当总线处于空闲状态时，SCL 和 SDA 同时被外部上拉电阻拉为高电平。当主机启动数据传输时，必须先产生一个起始条件。在 SCL 线是高电平时，SDA 线从高电平向低电平切换表示起始条件。当主机结束传输时要发送停止条件。在 SCL 线是高电平，SDA 线由低电平向高电平切换表示停止条件。下图显示了起始和停止条件的时序图。数据传输过程中，当 SCL 为 1 时，SDA 必须保持稳定。

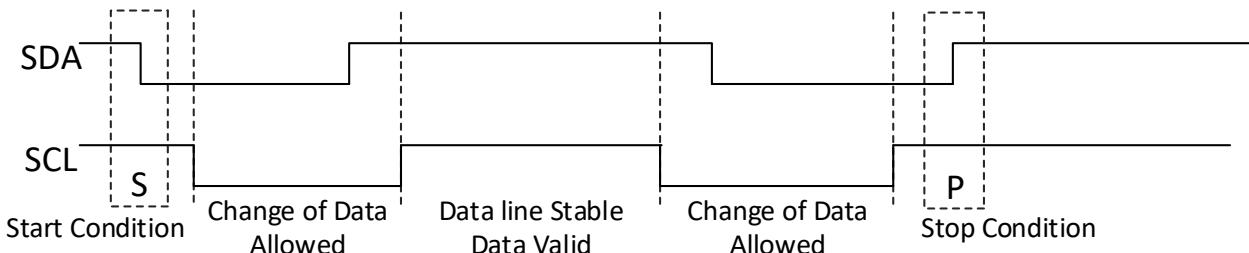


图 21-3 起始和停止条件

## 21.7.2 从机寻址协议

I2C 有两种地址格式：7 位的地址格式和 10 位的地址格式。

### 21.7.2.1 7 位的地址格式

下图中显示在起始条件 (S) 后发送的一个字节的前 7 位 (bit 7:1) 为从机地址，最低位 (bit0) 是数据方向位，当 bit 0 为 0，表示主机写数据到从机，1 表示主机从从机读数据。

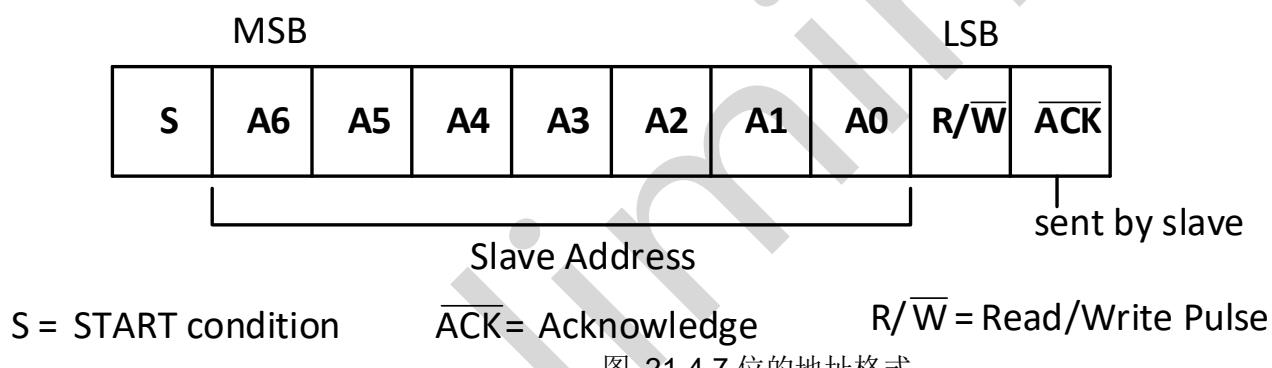
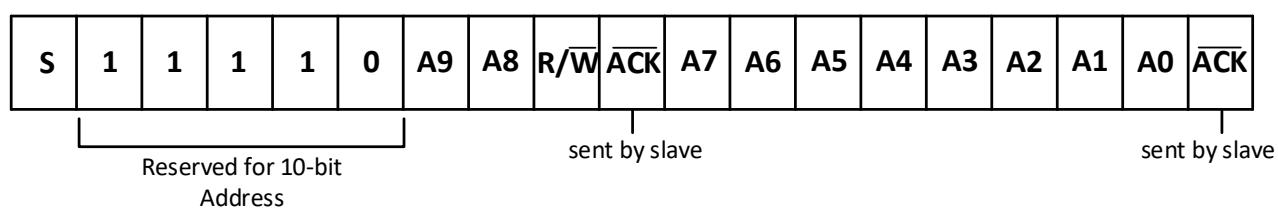


图 21-4 7 位的地址格式

### 21.7.2.2 10 位的地址格式

在 10 位的地址格式中，发送 2 个字节来传输 10 位地址。发送的第一个字节的位的描述如下：第一个 5 位 (bit 7:3) 用于告示从机接下来是 10 位的传输。第一个字节的后两位 (bit 2:1) 是从机地址的 bit 9:8，最低位 (bit 0) 是数据方向位 (R/W)。传输的第二个字节为 10 位地址的低八位。

具体如下图所示：



S = START condition  
 R/W = Read/Write Pulse  
 ACK = Acknowledge

图 21-5 10 位的地址格式

下表定义了 I2C 首字节的特殊用途和保留地址：

表 21-2 I2C 首字节

从机地址	R/W 位	描述
0000 000	0	广播呼叫地址。I2C 将数据放入接收缓冲，并产生一个广播呼叫中断。
0000 000	1	起始字节
0000 001	x	CBUS 地址。I2C 接口忽略该访问。
0000 010	x	保留
0000 011	x	保留
0000 1xx	x	保留
1111 1xx	x	保留
1111 0xx	x	10 位从机寻址

### 21.7.3 发送和接收协议

主机初始化数据传输并且从总线上发送或接收数据，作为主发送或者主接收。从机响应主机的请求来发送或接收数据，作为从发送或从接收器。

#### 21.7.3.1 主发送和从接收

所有数据都是以字节格式传输，且不限制每次传输的字节数。当主机发送完地址和 R/W 位或者主机发送一个字节的数据到从机上，从接收器必须产生一个响应信号（ACK）。当从接收器不能产生 ACK 响应信号，主机将会产生一个停止条件中止传输。从机不能响应时，必须释放 SDA 为高电平才能使得主机产生停止条件。

当主发送器如下图所示传输数据，从接收器在接收到的每个字节后产生一个 ACK 来响应主发送器。

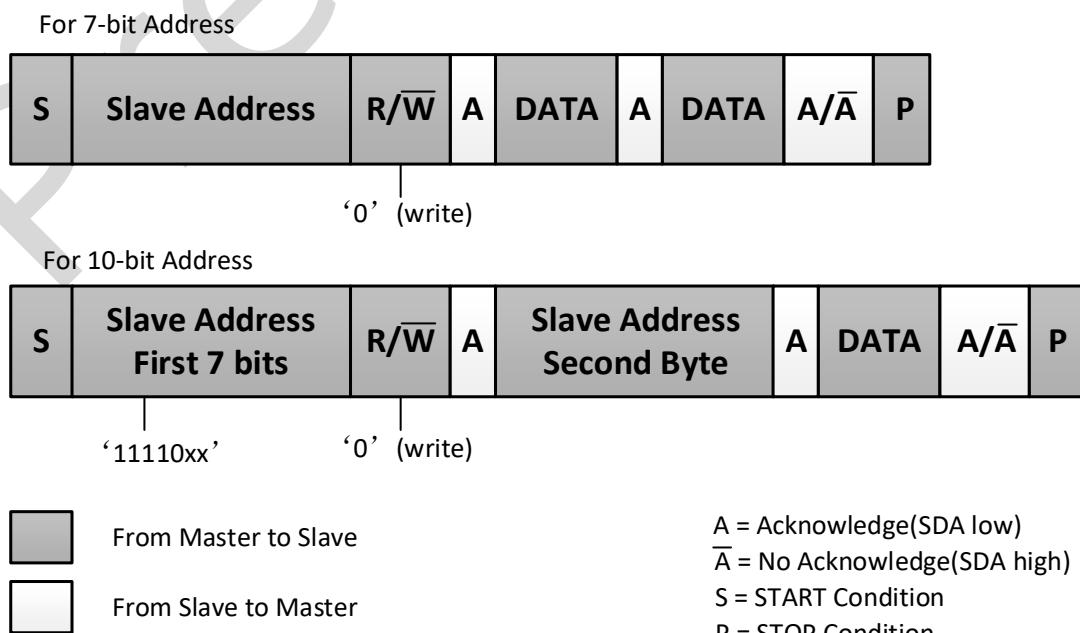


图 21-6 主发送协议

### 21.7.3.2 主接收和从发送

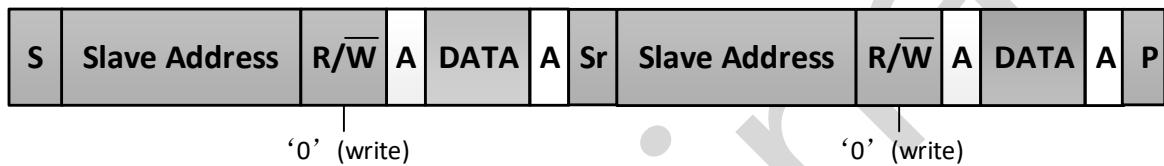
当主机如下图所示接收数据，主机必须在每次接收到一个字节数据后响应从发送器，除了最后一个字节。通过这种方式，主接收器能通知从发送器是否是最后一个字节。从发送器在检测到 NACK 时必须释放 SDA，这样主机可以产生停止条件。

P = STOP Condition

图 21-7 主接收协议

当主机不想产生停止条件而释放总线，可以产生一个重复起始条件。重复起始条件与起始条件相同，只是它是在 ACK 后产生。工作在主机模式下，I2C 接口可以使用不同的传输方向与相同的从机通信。

For 7-bit Address Write



For 7-bit Address Read

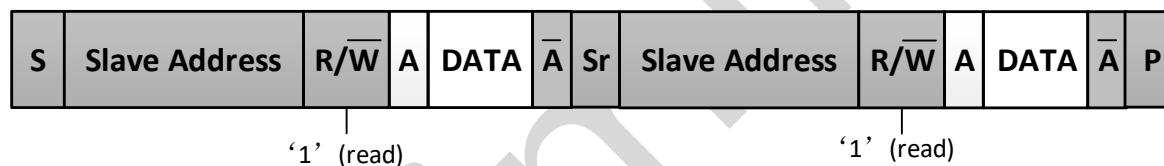


图 21-8 带 Restart (SR) 信号的主发送和接收协议

### 21.7.3.3 起始字节传输协议

起始字节传输协议用来给没有专用的 I2C 硬件模块的系统使用。当 I2C 模块作为主机时，在每次传输开始可以给需要的从机产生起始字节输出。

该协议由 7 个 0 以及一个 1 组成，如下图所示。处理器可以在地址阶段用低速采样 来查询总线。一旦检测到 0，处理器可以从低速采样切换到主机的正常速率。

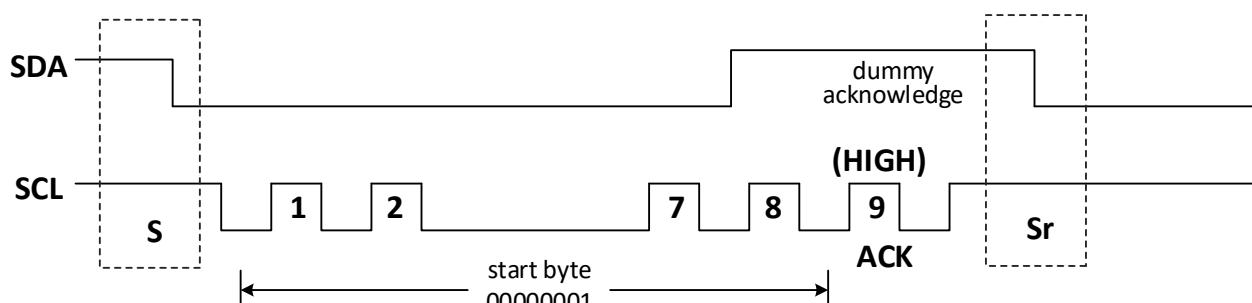


图 21-9 起始字节传输

起始字节程序流程如下：

- 1) 主机产生一个起始条件
- 2) 主机发送起始字节（0000 0001）
- 3) 主机发送 ACK 时钟脉冲（ACK）
- 4) 没有从机响应 ACK 信号
- 5) 主机产生重复起始条件（RESTART）

硬件 I2C 接收器不需要响应开始字节，因为这是一个保留地址，而且地址会在 RESTART 后复位。

#### 21.7.4 发送缓冲管理以及起始、停止和重复起始条件产生

当工作在主机模式，每当发送为空时 I2C 模块就在总线上产生一个停止条件。如果重复起始产生功能使能（RESTART=1），则传输方向从读变为写或者写变为读时产生重复起始条件。如果没有使能重复起始条件，则会在停止条件后产生一个起始条件。

下图显示了 DR 寄存器的位。

DR	CMD	DATA	0
	8 7		0

DATA —Read/Write field; data retrieved from slave is read from this field ; data to be sent to slave is written to this field.

CMD —Write-only field; this bit determines whether transfer to be carried out is read (CMD=1) or Write (CMD=0)

图 21-10 DR 寄存器

下面的时序图描述了 I2C 模块工作在主发送模式下 TX FIFO 变为空时的行为。

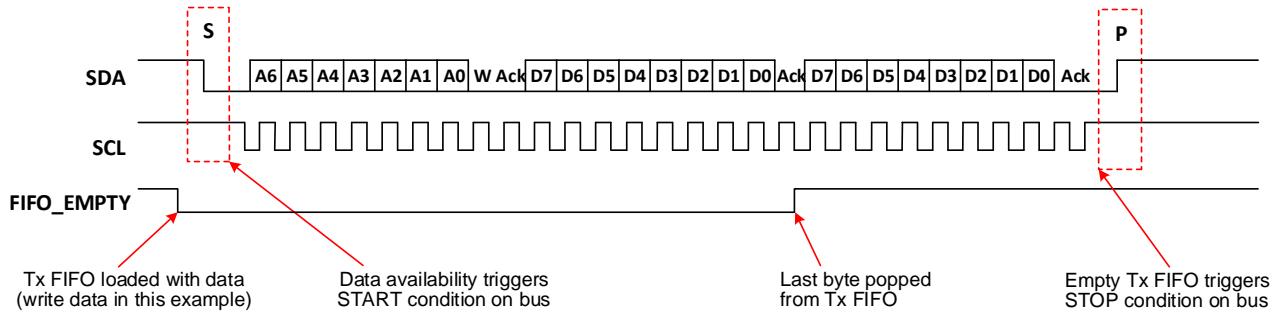


图 21-11 主发送, TX FIFO 为空

下面的时序图描述了 I2C 模块工作在主接收模式下当 TX FIFO 变为空时的行为。

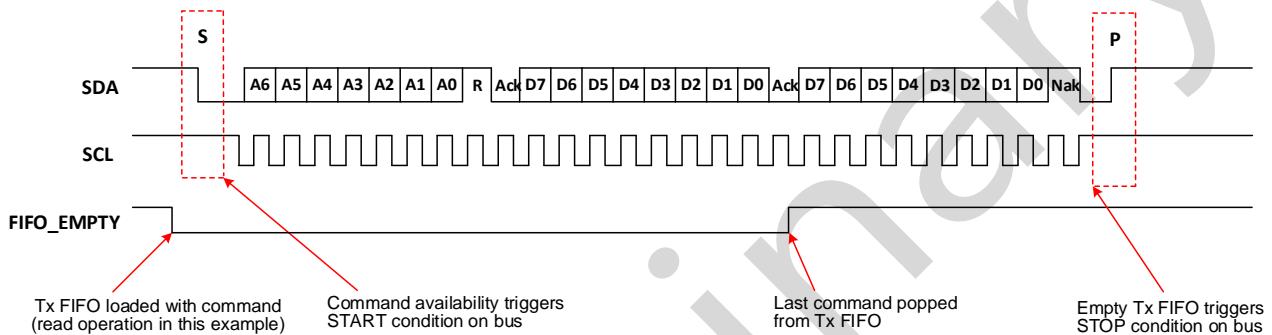


图 21-12 主接收, TX FIFO 为空

## 21.7.5 多个主机仲裁

I2C 总线是一个多主机的总线。仲裁是一个在有多个主机同时尝试控制总线，但只允许其中一个控制总线并使报文不被破坏的过程。一旦其中一个主机已经控制了总线，那么直到主机发送一个停止条件并且将总线释放为空闲状态时，其他主机才能控制总线。

当 SCL 线是高电平时，仲裁在 SDA 线发生。如果两个或多个主机尝试发送信息到总线，在其他主机都产生"0"的情况下，首先产生一个"1"的主机将丢失仲裁。丢失仲裁的主机可以继续产生时钟脉冲直到字节传输结束。如果每个主机都尝试寻址相同的器件，仲裁会继续在数据阶段进行。

检测到丢失仲裁后，I2C 接口会停止产生 SCL 信号。

下图显示了两个主机仲裁的总线时序。

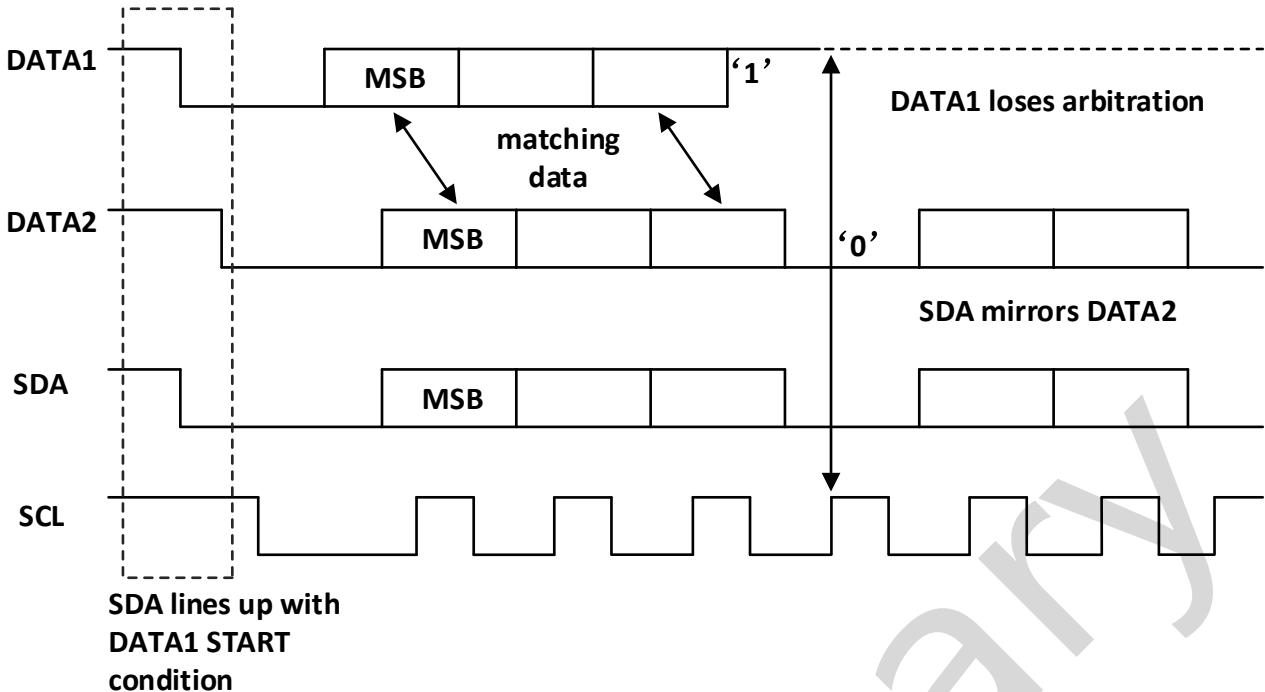


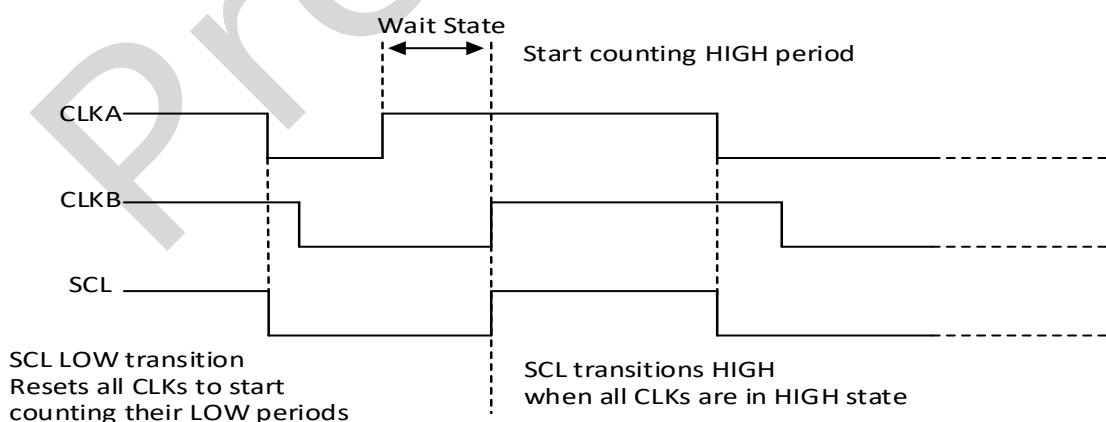
图 21-13 多个主机仲裁

### 21.7.6 时钟同步

当两个或多个主机试图同时在总线传输信息时，必须仲裁和同步 SCL 时钟。所有的主机产生自己的时钟来传输消息。数据只在时钟的高电平有效。时钟同步是通过 SCL 信号的线‘与’连接进行的。当主机把 SCL 时钟变成 0，主机会计算 SCL 低电平的时间，在下一个时钟周期开始把 SCL 时钟变成 1。但是，假如另一个主机把 SCL 保持为 0，那么这个主机会进入等待状态直到 SCL 时钟变为 1。

所有的主机会计算它们的高电平时间，最短高电平时间的主机把 SCL 变为 0。接下来主机会计算低电平时间，最长低电平时间的主机强制其他主机进入等待状态。这样就产生一个同步后的 SCL 时钟，如下图所示。

图 21-14 多个主机时钟同步



## 21.7.7 SCL 配置

I2C 的电平可参考如下配置:

### SCL master clock generation

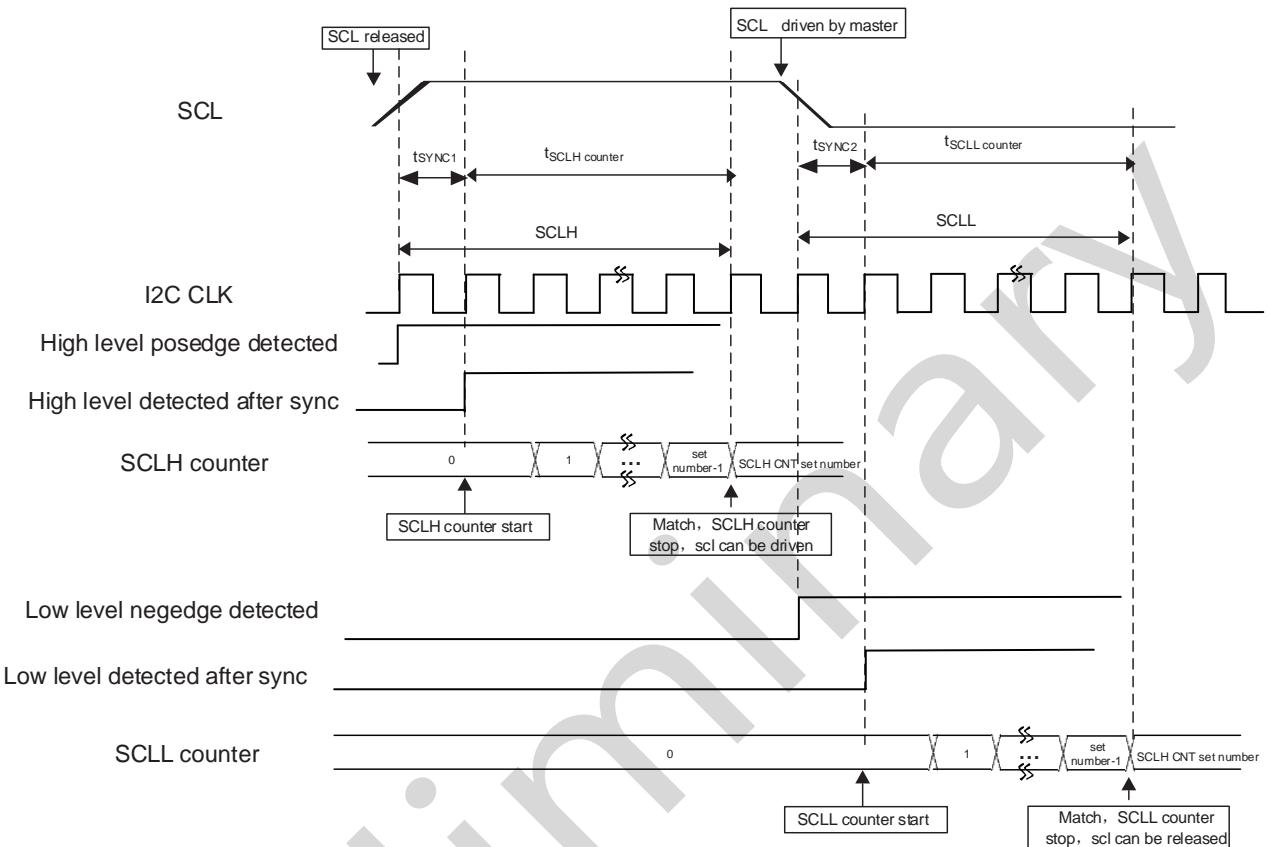


图 21-15 SCL master clock generation

标准模式:

$$SCLH = (SSHR + 12) \times I2C\ CLK + tSYNC1$$

$$SCLL = (SSLR + 1) \times I2C\ CLK + tSYNC2$$

注: tSYNC1 为 0~1 个 I2C CLK, tSYNC2 为 0~1 个 I2C CLK

快速模式:

$$SCLH = (FSHR + 12) \times I2C\ CLK + tSYNC1$$

$$SCLL = (FSLR + 1) \times I2C\ CLK + tSYNC2$$

注: tSYNC1 为 2~3 个 I2C CLK, tSYNC2 为 2~3 个 I2C CLK

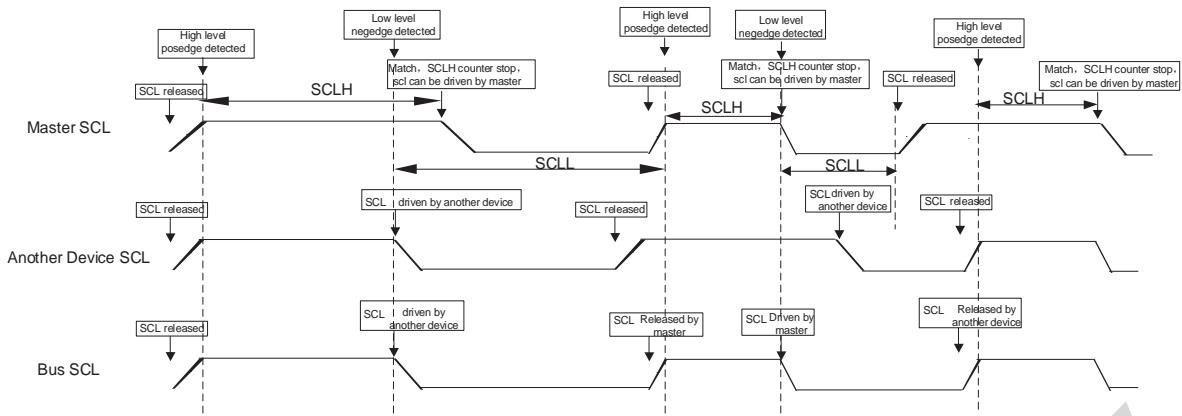


图 21-16 SCL master clock synchronization

## 21.8 I2C 工作模式

I2C 接口可以以下述 4 种方式中的一种运行：

- 从发送器模式
- 从接收器模式
- 主发送器模式
- 主接收器模式

注：I2C 接口模块只能工作在主机模式或者从机模式，但不能同时工作在两种模式下。因此要确保寄存器 CR 中位 6 (DISSLAVE) 和位 0 (MASTER) 不能分别设置为 0 和 1 (或者分别为 1 和 0)。

### 21.8.1 从模式

下面介绍从模式的程序流程图。

初始化配置

- 1) 写 0 到 ENR 寄存器位 0 禁止 I2C。
- 2) 通过初始化 SAR 寄存器来配置从机地址。该地址为 I2C 接口所响应的地址。
- 3) 配置 CR 寄存器指定地址格式 (设置 bit 3 来选择 7 位或 10 位地址格式)。写 0 到 CR 寄存器的位 6 (DISSLAVE) 和写 0 到位 0 (MASTER)。
- 4) 写 1 到 ENR 寄存器中位 0 来使能 I2C 接口模块。

从发送的单字节操作

当 I2C 接口被其他 I2C 主机寻址并请求数据的时候，I2C 接口工作在从发送模式，步骤如下：

- 1) 其他 I2C 主机器件初始化 I2C 传输，发送地址与 SAR 寄存器中的从机地址匹配。
- 2) I2C 接口响应发送的地址，识别传输的方向是工作在从发送模式。
- 3) I2C 接口产生 RD\_REQ 中断 (寄存器 RAWISR 位 5)，并且将 SCL 线拉低。总线一直处于等待状态直到软件响应。

如果 RD\_REQ 中断被屏蔽 (寄存器 IMR[5] = 0)，建议 CPU 定期查询 RAWISR 寄存器。

- a) RAWISR 位 5 置位等效于产生了一个 RD\_REQ 中断。

- b) 软件必须满足 I2C 传输的要求。
  - c) 时间间隔通常在 10 个 SCL 时钟周期左右。例如，对于 400kbps，时间间隔是 25us。
- 4) 如果在接收读请求之前 TX FIFO 仍然有数据，I2C 接口就会产生一个 TX\_ABRT 中断 (RAWISR[6])，清空 TX FIFO 中的数据。
- 5) 软件写数据到 DR 寄存器 (其中位 8 设置为 0)。
- 6) 软件必须先清除 RAWISR 寄存器 RD\_REQ 和 TX\_ABRT 中断 (分别为 bit5, 6)。
- 7) I2C 接口释放 SCL，并发送数据字节。
- 8) 主机器件发送重复起始条件控制总线或者发送停止条件释放总线。

#### 从接收的单字节操作

当其他主机器件寻址 I2C 接口并且发送数据，I2C 接口工作在从接收模式，步骤如下：

- 1) 其他 I2C 主机器件初始化 I2C 传输，发送地址与 SAR 寄存器中的从机地址匹配。
- 2) I2C 接口响应发送的地址，识别传输的方向是工作在从接收模式。
- 3) I2C 接口收到主机发送的数据并将数据存储在接收缓冲中。
- 4) I2C 接口产生 RX\_FULL 中断 (RAWISR[2])。如果 RX\_FULL 中断被屏蔽 (IMR[2] = 0)，建议软件定期查询 SR 寄存器。读到 SR 寄存器位 3 (RFNE) 为 1 时等效于 RX\_FULL 中断产生。
- 5) 软件通过读 DR 寄存器中的 bit 7:0 来获得接收到的数据。
- 6) 主机器件发送重复起始条件控制总线或者发送停止条件释放总线。

#### 从机的块传输操作

标准的 I2C 协议中，所有的数据处理都是单个字节的处理，程序通过写一个字节到从机的 TX FIFO 响应主机的读请求。当一个从机（从发送）接收到主机（主接收）的读请求 (RD\_REQ) 时，最少有一个数据放到从发送的 TX FIFO。这个 I2C 接口模块可以处理 TX FIFO 中有多个数据，所以接下来的读请求不需要再产生中断来取数据。最终，这极大地减少了因为每次数据中断导致的等待时间。

该模式仅存在于当 I2C 接口作为从发送模式时。如果接收到主机发送的 ACK，从机的 TX FIFO 中没有数据，I2C 接口将拉低 I2C 总线的 SCL 线直到读请求中断 (RD\_REQ) 产生并且 TX FIFO 的数据准备好后才释放 SCL 线。

如果 RX\_REQ 中断被屏蔽 (ISR[5] = 0)，软件可以定期查询 RAWISR 寄存器。当读到 RAWISR[5] 返回为 1 等效于产生了 RX\_REQ 中断。

RD\_REQ 中断由于读请求产生，像中断一样必须退出中断服务程序 (ISR) 时清除。在中断服务程序中 (ISR) 可以写一个或多个字节的数据到 TX FIFO。在这些字节传输给主机的过程中，如果主机响应了最后一个字节，从机将必须再次产生 RD\_REQ 中断请求。这是因为主机要求更多的数据。

如果主机接收了来自 I2C 接口的 n 字节，但是程序写到 TX FIFO 中的数据个数大于 n，从机在完成要求的 n 字节的数据发送后，将会清空 TX FIFO 并且忽略额外的字节。

## 21.8.2 主模式

### 初始化配置

- 1) 通过设置 **ENR[0] = 0** 来禁止 I2C 接口。
- 2) 配置 CR 寄存器的 bit 2:1 设置 I2C 工作的速率模式（标准模式、快速模式）。同时确保 bit 6 (**DISSLAVE**) 为 1，且 bit 0 (**MASTER**) 为 1。
- 3) 往 TAR 寄存器写入 I2C 器件地址。设置该寄存器可配置为广播地址或起始字节命令。
- 4) 置位 **ENR[0]** 使能 I2C 接口。
- 5) 将传输的数据以及传输方向写入到 DR 寄存器中。如果在使能 I2C 接口之前配置了 DR 寄存器，数据和命令都会丢失，这是因为在 I2C 接口禁止的情况下缓冲是清空的。

以上的步骤将会使得 I2C 接口产生一个起始条件并发送地址字节数据到 I2C 总线上。

#### 主发送和主接收

I2C 接口支持读写的动态切换。当发送数据时，写数据到 I2C RX/TX 数据缓冲和命令寄存器 (DR) 的低字节中，配置 CMD 位为 0 产生写操作。接下来的读命令，不需要设置 DR 寄存器的低字节，只需要确保 CMD 位为 1。如果发送 FIFO 为空，I2C 模块拉低 SCL 直到下个命令写入到发送 FIFO 中。

#### 程序流程图

下面的流程图为 I2C 接口作为主机的程序示例：

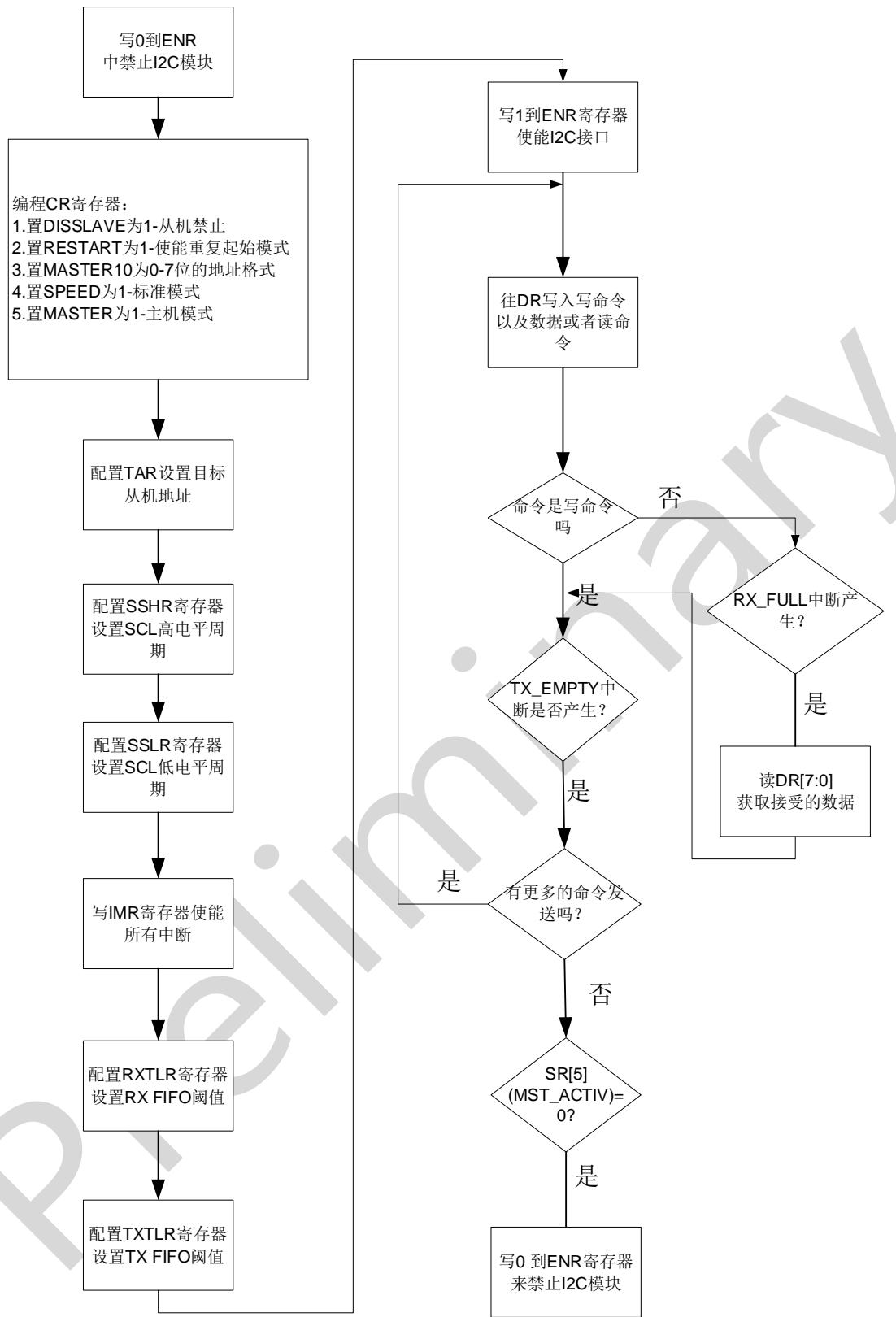


图 21-17 I2C 接口主机流程图

### 21.8.3 I2C 中止传输

ENR 寄存器中的 ABORT 控制位允许软件在完成 TX FIFO 中传输命令之前放弃 I2C 总线。作为 ABORT 请求的响应，I2C 模块发出一个停止条件到 I2C 总线，同时清空 TX FIFO。中止传输操作只允许在主模式下。

程序流程：

- 1) 停止往 TX FIFO (DR) 中写新的命令
- 2) 如果工作在 DMA 模式中，置 TXEN = 0 禁止发送 DMA
- 3) 置 ENR 寄存器 ABORT 位为 1
- 4) 等待 TX\_ABRT 中断

## 21.9 I2C 寄存器描述

表 21-3 I2C 寄存器描述概览

Offset	Acronym	Register Name	Reset
0x00	I2C_CR	I2C 控制寄存器	0x0000007F
0x04	I2C_TAR	I2C 目标地址寄存器	0x00000055
0x08	I2C_SAR	I2C 从机地址寄存器	0x00000055
0x10	I2C_DR	I2C 数据命令寄存器	0x00000000
0x14	I2C_SSHR	标准模式 I2C 时钟高电平计数寄存器	0x00000190
0x18	I2C_SSLR	标准模式 I2C 时钟低电平计数寄存器	0x000001D6
0x1C	I2C_FSHR	快速模式 I2C 时钟高电平计数寄存器	0x0000003C
0x20	I2C_FSLR	快速模式 I2C 时钟低电平计数寄存器	0x00000082
0x2C	I2C_ISR	I2C 中断状态寄存器	0x00000000
0x30	I2C_IMR	I2C 中断屏蔽寄存器	0x000008FF
0x34	I2C_RAWISR	I2C RAW 中断寄存器	0x00000000
0x38	I2C_RXTLR	I2C 接收阈值	0x00000000
0x3C	I2C_TXTLR	I2C 发送阈值	0x00000000
0x40	I2C_ICR	I2C 组合和独立中断清除寄存器	0x00000000
0x44	I2C_RX_UNDER	I2C 清除 RX_UNDER 中断寄存器	0x00000000
0x48	I2C_RX_OVER	I2C 清除 RX_OVER 中断寄存器	0x00000000
0x4C	I2C_TX_OVER	I2C 清除 TX_OVER 中断寄存器	0x00000000
0x50	I2C_RD_REQ	I2C 清除 RD_REQ 中断寄存器	0x00000000
0x54	I2C_TX_ABRT	I2C 清除 TX_ABRT 中断寄存器	0x00000000
0x58	I2C_RX_DONE	I2C 清除 RX_DONE 中断寄存器	0x00000000

Offset	Acronym	Register Name	Reset
0x5C	I2C_ACTIV	I2C 清除 ACTIVITY 中断寄存器	0x00000000
0x60	I2C_STOP	I2C 清除 STOP_DET 中断寄存器	0x00000000
0x64	I2C_START	I2C 清除 START_DET 中断寄存器	0x00000000
0x68	I2C_GC	I2C 清除 GEN_CALL 中断寄存器	0x00000000
0x6C	I2C_ENR	I2C 使能寄存器	0x00000000
0x70	I2C_SR	I2C 状态寄存器	0x00000006
0x74	I2C_TXFLR	I2C 发送缓冲水平寄存器	0x00000000
0x78	I2C_RXFLR	I2C 接收缓冲水平寄存器	0x00000000
0x7C	I2C_HOLD	I2C SDA 保持时间寄存器	0x00000001
0x88	I2C_DMA	I2C DMA 控制寄存器	0x00000000
0x94	I2C_SETUP	I2C SDA 建立时间寄存器	0x00000064
0x98	I2C_GCR	I2C 广播呼叫 ACK 寄存器	0x00000001
0xB0	I2C_SLVMASK	I2C 从机地址掩码寄存器	0x000003FF
0xB4	I2C_SLVRCVADDR	I2C 从机接收地址寄存器	0x00000000

### 21.9.1 I2C 控制寄存器(I2C\_CR)

偏移地址:0x00

复位值:0x007F

Bit	15:12				11	10	9	8
Field	Reserved				SLV_TX_ABRT_DIS	RESTART	STOP	EMPINT
Type					rw	rw	rw	rw
Bit	7	6	5	4	3	2	1	0
Field	STOPINT	DISSLAVE	REPEN	MASTER 10	SLAVE10	SPEED		MASTER
Type	rw	rw	rw	rw	rw	rw		rw

Bit	Field	Type	Reset	Description
15:12	Reserved			保留, 始终读为 0
11	SLV_TX_ABRT_DIS	rw	0x00	I2C 作为从机读时 1:禁止接收到 RD_REQ 信号后清除 TX FIFO 0:接收到 RD_REQ 信号后清除 TX FIFO

Bit	Field	Type	Reset	Description
10	RESTART	rw	0x00	<p>发送或接收之前，是否产生一个 RESTART 信号 仅在 IC_EMPTYFIFO_HOLD_MASTER_EN 的配置为"1" 时有效。</p> <p>1:如果 REPEN 为"1"，数据接收或发送 (根据 CMD 的值) 前产生一个 RESTART 信号，无论前一个命令是否改变数据的传输方向。如果 REPEN 为"0"，STOP 信号将紧跟 START 信号。</p> <p>0:如果 REPEN 信号为"1"，仅在前一个命令改变传输方向时才产生 RESTART 信号。如果 REPEN 为"0"，STOP 信号将紧跟 START 信号。</p>
9	STOP	rw	0x00	<p>发送或接收之后，是否产生一个 STOP 信号 仅在 IC_EMPTYFIFO_HOLD_MASTER_EN 的配置为"1" 时有效。</p> <p>1:当前字节之后产生一个 STOP 信号，无论 TX FIFO 是否为空。如果 TX FIFO 不为空，主机立即发出一个新的传输及总线仲裁信号。</p> <p>0:当前字节之后不产生一个 STOP 信号，无论 TX FIFO 是否为空。主机继续当前传输 (根据 CMD 的值发送或接收数据)。如果 TX FIFO 为空，主机将拉低 SCL 挂起总线直至 TX FIFO 收到新数据。</p>
8	EMPINT	rw	0x00	该位控制 TX_EMPTY 中断产生，细节参考 RAWISR 寄存器。
7	STOPINT	rw	0x00	<p>在从机模式下，是否产生 STOP 中断。</p> <p>1: 当地址匹配时才产生 STOP 中断</p> <p>0: 无论地址是否匹配，都产生 STOP 中断。该位仅适用于从机模式。</p> <p>注：广播地址寻址时，如果该位置位，从机不产生 STOP 中断。</p> <p>STOP 中断仅当发送的地址与从机地址匹配时产生。</p>
6	DISSLAVE	rw	0x01	<p>该位控制 I2C 接口从机禁止(This bit controls whether I2C has its slave disabled)</p> <p>0:从机使能</p> <p>1:从机禁止</p>

Bit	Field	Type	Reset	Description
5	REPEN	rw	0x01	<p>当作为主机时该位控制是否发送 RESTART 条件 (Determines whether RESTART conditions may be sent when acting as a master)</p> <p>0:禁止 1:使能</p> <p>当 RESTART 禁止, I2C 接口作为主机时不能执行以下功能:</p> <ul style="list-style-type: none"> <li>发送起始字节</li> <li>组合格式模式下改变传输方向</li> <li>10 位的地址格式的读操作</li> </ul> <p>可以替换 RESTART 条件为先发送停止条件再发送起始条件。如果上述操作执行会置位 RAWISR 寄存器的位 6(TX_ABRT)。</p>
4	MASTER10	rw	0x01	<p>I2C 作为主机时的地址格式 (Address mode when acting as a master)</p> <p>0:7 位的地址格式 1:10 位的地址格式</p>
3	SLAVE10	rw	0x01	<p>当作为从机时, 该位控制响应 10 位或者 7 位地址(When acting as a slave, this bit controls whether the I2C responds to 7- or 10-bit addresses)</p> <p>0:7 位的寻址地址。I2C 接口忽略处理 10 位的寻址。对于 7 位寻址, 仅比较 SAR 寄存器的低 7 位。</p> <p>1:10 位的寻址地址。I2C 仅响应 10 位的寻址, 接收地址与 SAR 的 10 位比较。</p>
2:1	SPEED	rw	0x03	<p>该两位控制 I2C 接口工作的速率模式(These bits control at which speed the I2C operates)</p> <p>该设置仅当 I2C 接口工作在主机模式下有效。</p> <p>1:标准模式(0~100Kbps) 2:快速模式(400Kbps)</p>
0	MASTER	rw	0x01	<p>该位控制主机模式(This bit controls whether the I2C master is enabled)</p> <p>0:主机禁止 1:主机使能</p>

DISSLAVE(bit 6)和 MASTER(bit 0)配置如下表所列:

表 21-4 DISSLAVE(bit6)和 MASTER(bit0)配置

DISSLAVE CR[6]	MASTER CR[0]	状态
0	0	从机器件
0	1	配置错误
1	0	配置错误
1	1	主机器件

## 21.9.2 I2C 目标地址寄存器(I2C\_TAR)

偏移地址:0x04

复位值:0x0055

Bit	15:12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved	SPECIAL	GC	ADDR									
Type		rw	rw	rw									

Bit	Field	Type	Reset	Description
15:12	Reserved			保留, 始终读为 0
11	SPECIAL	rw	0x00	该位指示软件执行的是是否是特殊命令 (广播呼叫或者起始字节命令) 0:忽略第 10 位 GC, 正常使用 ADDR 位 1:执行特殊 I2C 命令, 如 GC 位描述
10	GC	rw	0x00	如果位 11 置位, 该位显示 I2C 执行的是广播呼叫还是起始字节 (If bit 11(SPECIAL) is set to 1, then this bit indicates whether a General Call or START byte command is to be performed by the I2C) 0:广播呼叫地址。发送广播呼叫地址时只能执行写操作。I2C 接口一直工作在广播地址模式下直到 SPECIAL(bit 11)的值被清零。 1:起始字节命令
9:0	ADDR	rw	0x55	主操作的目标地址 (This is the target address for any master transaction) 当发送一个广播地址, 这些位就可以忽略。 要产生开始字节的命令, CPU 只需要对这些位写一次。

### 21.9.3 I2C 从机地址寄存器(I2C\_SAR)

偏移地址:0x08

复位值:0x0055

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved						ADDR									
Type							rw									

Bit	Field	Type	Reset	Description
15:10	Reserved			保留, 始终读为 0
9:0	ADDR	rw	0x55	当 I2C 接口工作在从机模式下, 这些位存储从机地址。对于 7 位的地址格式, ADDR 只有[6:0]有效。

### 21.9.4 I2C 数据命令寄存器(I2C\_DR)

偏移地址:0x10

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved						CMD	DAT								
Type							w	rw								

Bit	Field	Type	Reset	Description
15:9	Reserved			保留, 始终读为 0
8	CMD	w	0x00	控制在主模式下执行读或写操作 1:读 0:写 当一个命令进入 TX FIFO, 该位用于区分读写命令。在从接收模式下, 该位写值操作被忽略。在从发送模式下, 写 0 表示 DR 寄存器的数据准备发送。
7:0	DAT	rw	0x00	I2C 总线待发送或者接收到的数据

### 21.9.5 标准模式 I2C 时钟高电平计数寄存器(I2C\_SSHR)

偏移地址:0x14

复位值:0x0190

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CNT															

Type	rw
------	----

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x0190	I2C 接口标准模式下 SCL 时钟高电平周期 注意：该寄存器可配置值在 6 和 65525 之间，这是由于 I2C 接口使用了一个 16 位的计数器，该计数器值等于 SSHR+10 时标志 I2C 总线处于空闲状态。

## 21.9.6 标准模式 I2C 时钟低电平计数寄存器(I2C\_SSLR)

偏移地址:0x18

复位值:0x01D6

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CNT															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x01D6	I2C 接口标准模式下 SCL 时钟低电平周期 最小值为 8。

## 21.9.7 快速模式 I2C 时钟高电平计数寄存器(I2C\_FSHR)

偏移地址:0x1C

复位值:0x003C

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CNT															
Type	rw															

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x003C	I2C 接口快速模式下 SCL 时钟高电平周期 当 I2C 工作在标准模式下该寄存器为只读且返回值为 0。最小值为 6。

## 21.9.8 快速模式 I2C 时钟低电平计数寄存器(I2C\_FSLR)

偏移地址:0x20

复位值:0x0082

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CNT															

Type

rw

Bit	Field	Type	Reset	Description
15:0	CNT	rw	0x0082	I2C 接口快速模式下 SCL 时钟低电平周期 当 I2C 工作在标准模式下该寄存器为只读且返回值为 0。最小值为 8。

### 21.9.9 I2C 中断状态寄存器(I2C\_ISR)

偏移地址:0x2C

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	
Field	Reserved					GC	START	STOP	ACTIV
Type						r	r	r	r
Bit	7	6	5	4	3	2	1	0	
Field	RX_DONE	TX_ABRT	RD_REQ	TX_EMPTY	TX_OVER	RX_FULL	RX_OVER	RX_UNDER	
Type	r	r	r	r	r	r	r	r	

Bit	Field	Type	Reset	Description
15:12	Reserved			保留, 始终读为 0
11:0	ISR	r	0x0000	具体每位描述可以参考 RAWISR 寄存器

### 21.9.10 I2C 中断屏蔽寄存器(I2C\_IMR)

偏移地址:0x30

复位值:0x08FF

Bit	15	14	13	12	11	10	9	8	
Field	Reserved					GC	START	STOP	ACTIV
Type						rw	rw	rw	rw
Bit	7	6	5	4	3	2	1	0	
Field	RX_DONE	TX_ABRT	RD_REQ	TX_EMPTY	TX_OVER	RX_FULL	RX_OVER	RX_UNDER	
Type	rw	rw	rw	rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
15:12	Reserved			保留, 始终读为 0
11:0	IMR	rw	0x08FF	每一位屏蔽与 ISR 对应位。

## 21.9.11 I2C RAW 中断寄存器(I2C\_RAWISR)

偏移地址:0x34

复位值:0x0000

Bit	15	14	13	12	11	10	9	8
Field	Reserved				GC	START	STOP	ACTIV
Type					r	r	r	r
Bit	7	6	5	4	3	2	1	0
Field	RX_DONE	TX_ABRT	RD_REQ	TX_EMPTY	TX_OVER	RX_FULL	RX_OVER	RX_UNDER
Type	r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
15:12	Reserved			保留, 始终读为 0
11	GC	r	0x00	广播呼叫(General call) 接收到广播呼叫地址时置位。 禁止 I2C 接口或者当 CPU 读 GC 寄存器时清零。I2C 将接收的数据存储在接收缓冲中。
10	START	r	0x00	起始条件检测(Start condition detection) 无论 I2C 接口工作在主机或者从机, 一旦检测到 I2C 接口上起始或者重复起始条件即置位该位。
9	STOP	r	0x00	停止条件检测(Stop condition detection) 该位状态取决于 CR 寄存器的 STOPINT 的状态: 当 STOPINT = 0: 无论 I2C 接口工作在主机或者从机, 一旦检测到 I2C 接口上停止条件时即置位该位。从机模式下, 无论寻址是否匹配都会产生一个 STOP 中断。 当 STOPINT = 1: 在主机模式下(MASTER = 1), 该位显示 I2C 接口是否发生停止条件; 在从机模式下(MASTER = 0), 仅当从机地址匹配成功时产生一个 STOP 中断。

Bit	Field	Type	Reset	Description
8	ACTIV	r	0x00	<p>I2C 接口激活, 该位用于捕捉 I2C 模块的活动状态。置位后只能由以下四种方式清零:</p> <ul style="list-style-type: none"> <li>禁止 I2C 接口</li> <li>读 ACTIV 寄存器</li> <li>读 ICR 寄存器</li> <li>系统复位</li> </ul> <p>一旦置位后, 只能由上述方式清零, 即使 I2C 处于空闲状态, 该位也仍然保持为高直到被清零。</p>
7	RX_DONE	r	0x00	<p>从发送结束 (Transmit done)</p> <p>当 I2C 作为从发送时, 如果发送一个字节的数据后主机没有响应, 将会置位该位。</p> <p>该情况发生在传输的最后一个字节, 表示传输结束。</p>
6	TX_ABRT	r	0x00	<p>发送中止(Transmit abort)</p> <p>当 I2C 接口作为发送机时, 不能发送完缓冲中的数据时置位。</p> <p>注意: 发送中止会将 I2C 接口中的接收和发送缓冲清空。发送缓冲会处于刷新状态直到读 TX_ABRT 寄存器。一旦该读操作执行后, 发送就可以接收 APB 总线上的新的数据。</p>
5	RD_REQ	r	0x00	<p>读请求(Read request)</p> <p>当 I2C 作为从机, 其他主机试图从 I2C 接口读取数据时置位。</p> <p>I2C 接口会使总线保持等待状态(SCL = 0)直到中断被处理。这就意味着 I2C 接口作为从机时被其他主机寻址成功且要求发送数据。处理器必须响应该中断然后写入数据到 DR 寄存器中。当处理器读 RD_REQ 寄存器该位清零。</p>
4	TX_EMPTY	r	0x00	<p>发送缓冲空(Transmit buffer empty)</p> <p>该位状态取决于 CR 寄存器中的 EMPINT 状态:</p> <ul style="list-style-type: none"> <li>当 EMPINT = 0, 发送缓冲区数据个数小于等于阈值时置位</li> <li>当 EMPINT = 1, 发送缓冲区数据个数小于等于阈值且前一个发送数据的内部移位寄存器发送结束时置位</li> </ul> <p>当发送缓冲非空时由硬件自动清零。</p>
3	TX_OVER	r	0x00	<p>发送缓冲过载(Transmit buffer over)</p> <p>发送缓冲满时处理器写入新的数据导致溢出时置位。</p>

Bit	Field	Type	Reset	Description
2	RX_FULL	r	0x00	接收缓冲非空(Receive buffer not empty) 当接收缓冲非空时置位。 当接收缓冲为空时由硬件清零。
1	RX_OVER	r	0x00	接收缓冲过载(Receive buffer over) 接收缓冲满且又收到新的数据时置位。此时 I2C 接口会响应，但新的数据会丢失。
0	RX_UNDER	r	0x00	接收缓冲欠载 (Receive buffer under) 当 RX FIFO 为空时处理器读 DR 寄存器时置位。

### 21.9.12 I2C 接收阈值(I2C\_RXTLR)

偏移地址:0x38

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														TL	
Type															rw	

Bit	Field	Type	Reset	Description
15:8	Reserved			保留, 始终读为 0
7:0	TL	rw	0x00	接收 FIFO 阈值(Receive FIFO threshold level) 控制 RX_FULL 中断触发。

### 21.9.13 I2C 发送阈值(I2C\_TXTLR)

偏移地址:0x3C

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														TL	
Type															rw	

Bit	Field	Type	Reset	Description
15:8	Reserved			保留, 始终读为 0
7:0	TL	rw	0x00	发送 FIFO 阈值(Transmit FIFO threshold level) 控制 TX_EMPTY 中断触发。

### 21.9.14 I2C 组合和独立中断清除寄存器(I2C\_ICR)

偏移地址:0x40

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Reserved																ICR
Type																	rc_r

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0
0	ICR	rc_r	0x00	读该寄存器将会清除所有组合中断、独立中断 该位不清除硬件可自动清除的中断, 仅清除软件可清除中断。

### 21.9.15 I2C 清除 RX\_UNDER 中断寄存器(I2C\_RX\_UNDER)

偏移地址:0x44

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Reserved																RX_UNDER
Type																	rc_r

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0
0	RX_UNDER	rc_r	0x00	读该寄存器清零 RX_UNDER 中断(RAWISR[0])

### 21.9.16 I2C 清除 RX\_OVER 中断寄存器(I2C\_RX\_OVER)

偏移地址:0x48

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Reserved																RX_OVER
Type																	rc_r

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0
0	RX_OVER	rc_r	0x00	读该寄存器清零 RX_OVER 中断(RAWISR[1])

## 21.9.17 I2C 清除 TX\_OVER 中断寄存器(I2C\_TX\_OVER)

偏移地址:0x4C

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved															TX_OVER
Type																rc_r

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0
0	TX_OVER	rc_r	0x00	读该寄存器清零 TX_OVER 中断 (RAWISR[3])

### 21.9.18 I2C 清除 RD\_REQ 中断寄存器(I2C\_RD\_REQ)

偏移地址:0x50

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														RD_REQ	
Type															rc_r	

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0
0	RD_REQ	rc_r	0x00	读该寄存器清零 RD_REQ 中断(RAWISR[5])

### 21.9.19 I2C 清除 TX\_ABRT 中断寄存器(I2C\_TX\_ABRT)

偏移地址:0x54

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														TX_ABRT	
Type															rc_r	

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0
0	TX_ABRT	rc_r	0x00	读该寄存器清零 TX_ABRT 中断(RAWISR[6]) 同时也将 TX FIFO 从刷新/复位状态中释放, 以便接收写入的数据。

### 21.9.20 I2C 清除 RX\_DONE 中断寄存器(I2C\_RX\_DONE)

偏移地址:0x58

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														RX_DONE	
Type															rc_r	

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
0	RX_DONE	rc_r	0x00	读该寄存器清零 RX_DONE 中断(RAWISR[7])

### 21.9.21 I2C 清除 ACTIVITY 中断寄存器(I2C\_ACTIV)

偏移地址:0x5C

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved															ACTIV
Type																rc_r

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0
0	ACTIV	rc_r	0x00	如果 I2C 总线不活动则读该寄存器清零 ACTIV 中断 (RAWISR[8]) 如果 I2C 仍然活动, 那么 ACTIV 中断将继续置位。当 I2C 模块禁止或者在 I2C 总线不再活动时该位由硬件清零。可以通过读该寄存器得到 RAWISR 中 ACTIV(bit 8) 的状态。

### 21.9.22 I2C 清除 STOP\_DET 中断寄存器(I2C\_STOP)

偏移地址:0x60

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved															STOP
Type																rc_r

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0
0	STOP	rc_r	0x00	读该寄存器清零 STOP 中断(RAWISR[9])

### 21.9.23 I2C 清除 START\_DET 中断寄存器(I2C\_START)

偏移地址:0x64

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved															START

Type	rc_r
------	------

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0
0	START	rc_r	0x00	读该寄存器清零 START 中断(RAWISR[10])

### 21.9.24 I2C 清除 GEN\_CALL 中断寄存器(I2C\_GC)

偏移地址:0x68

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Reserved																GC
Type																	rc_r

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0
0	GC	rc_r	0x00	读该寄存器清零 GC 中断(RAWISR[11])

### 21.9.25 I2C 使能寄存器(I2C\_ENR)

偏移地址:0x6C

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Reserved																ABORT
Type																	rw

Bit	Field	Type	Reset	Description
15:2	Reserved			保留, 始终读为 0
1	ABORT	rw	0x00	<p>I2C 传输中止(I2C transfer abort)            0:中止没有发生或者已经结束            1:中止操作正在进行</p> <p>I2C 模块作为主机时置位可以软件中止 I2C 的传输。一旦置位不能立即清除。置位后 I2C 模块控制逻辑会在完成当前传输之后产生一个 STOP 条件和清空发送缓冲，中止操作之后产生 TX_ABRT 中断。</p> <p>该 ABORT 位会在中止操作结束后自动清零。</p>

Bit	Field	Type	Reset	Description
0	ENABLE	rw	0x00	I2C 模块使能(I2C mode enable) 0:禁止 I2C 模块(发送和接收缓冲保持擦除状态) 1:使能 I2C 模块

### 21.9.26 I2C 状态寄存器(I2C\_SR)

偏移地址:0x70

复位值:0x0006

Bit	15:7	6	5	4	3	2	1	0
Field	Reserved	SLV_ACTIV	MST_ACTIV	RFF	RFNE	TFE	TFNE	ACTIV
Type		r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
15:7	Reserved			保留, 始终读为 0
6	SLV_ACTIV	r	0x00	从机状态机活动状态位 (Slave FSM activity status) 0:从机状态机处于 IDLE 状态, 所以 I2C 从机部分不活动 1:从机状态机不处于 IDLE 状态, 所以 I2C 从机部分活动
5	MST_ACTIV	r	0x00	主机状态机活动状态位 (Master FSM activity status) 0:主机状态机处于 IDLE 状态, 所以 I2C 主机部分不活动 1:主机状态机不处于 IDLE 状态, 所以 I2C 主机部分活动
4	RFF	r	0x00	接收缓冲满 (Receive FIFO completely full) 0:接收缓冲未满 1:接收缓冲满
3	RFNE	r	0x00	接收缓冲非空 (Receive FIFO not empty) 0:接收缓冲空 1:接收缓冲非空
2	TFE	r	0x01	发送缓冲空 (Transmit FIFO completely empty) 0:发送缓冲非空 1:发送缓冲空
1	TFNE	r	0x01	发送缓冲未满 (Transmit FIFO not full) 0:发送缓冲满 1:发送缓冲未满
0	ACTIV	r	0x00	I2C 活动状态 (I2C activity status) MST_ACTIV 位与 SLV_ACTIV 位相或的结果。

### 21.9.27 I2C 发送缓冲水平寄存器(I2C\_TXFLR)

偏移地址:0x74

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														CNT	
Type															r	

Bit	Field	Type	Reset	Description
15:2	Reserved			保留, 始终读为 0
1:0	CNT	r	0x00	发送缓冲中有效数据个数(0~2)

### 21.9.28 I2C 接收缓冲水平寄存器(I2C\_RXFLR)

偏移地址:0x78

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														CNT	
Type															r	

Bit	Field	Type	Reset	Description
15:2	Reserved			保留, 始终读为 0
1:0	CNT	r	0x00	接收缓冲中有效数据个数(0~2)

### 21.9.29 I2C SDA 保持时间寄存器(I2C\_HOLD)

偏移地址:0x7C

复位值:0x0000 0001

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved														RX_HOLD	
Type															rw	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	TX_HOLD															
Type	rw															

Bit	Field	Type	Reset	Description
31:24	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
23:16	RX_HOLD	rw	0x00	当 I2C 器件作为接收时, 配置 SDA 保持时间, 单位为 APB1 系统时钟周期
15:0	TX_HOLD	rw	0x01	当 I2C 器件作为发送时, 配置 SDA 保持时间, 单位为 APB1 系统时钟周期

### 21.9.30 I2C DMA 控制寄存器(I2C\_DMA)

偏移地址:0x88

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														TXEN	RXEN
Type															rw	rw

Bit	Field	Type	Reset	Description
15:2	Reserved			保留, 始终读为 0
1	TXEN	rw	0x00	发送 DMA 使能(Transmit DMA enable) 0:发送 DMA 禁止 1:发送 DMA 使能
0	RXEN	rw	0x00	接收 DMA 使能(Receive DMA enable) 0:接收 DMA 禁止 1:接收 DMA 使能

### 21.9.31 I2C SDA 建立时间寄存器(I2C\_SETUP)

偏移地址:0x94

复位值:0x0064

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														CNT	
Type															rw	

Bit	Field	Type	Reset	Description
15:8	Reserved			保留, 始终读为 0
7:0	CNT	rw	0x64	SDA 建立时间(SDA setup) 如果所需延迟时间为 1000ns, 当 APB1 时钟频率为 10MHz 时, 建议该寄存器设为 11。该寄存器最小值为 2。

### 21.9.32 I2C 广播呼叫 ACK 寄存器(I2C\_GCR)

偏移地址:0x98

复位值:0x0001

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														GC	
Type															rw	

Bit	Field	Type	Reset	Description
15:1	Reserved			保留, 始终读为 0
0	GC	rw	0x01	广播呼叫 ACK(ACK general call) 1:接收到广播呼叫后响应 ACK 0:接收到广播呼叫后不响应, 也不产生中断

### 21.9.33 I2C 从机地址掩码寄存器(I2C\_SLVMASK)

偏移地址:0xB0

复位值:0x03FF

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved						MASK									
Type							rw									

Bit	Field	Type	Reset	Description
15:10	Reserved			保留, 始终读为 0
9:0	MASK	rw	0x03FF	从机地址掩码 1:表示 SAR 寄存器的相应位需要比较 0:表示 SAR 寄存器的相应位被掩码忽略, 不需要比较

### 21.9.34 I2C 从机接收地址寄存器(I2C\_SLVRCVADDR)

偏移地址:0xB4

复位值:0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved						ADDR									
Type							r									

Bit	Field	Type	Reset	Description
15:10	Reserved			保留, 始终读为 0
9:0	ADDR	r	0x0000	从机实际接收的地址

## 22 控制器局域网(CAN)

### 22.1 简介

CAN 是 Controller Area Network 的缩写，属于总线式串行通信网络。目前 CAN 的应用场景中，网络节点数量庞大，报文传输为多主方式工作，面对大量收到的报文，为了减少 CPU 的负荷，报文接收的实时响应程度需要降低。对于发送报文，支持软件配置优先级。

### 22.2 功能框图

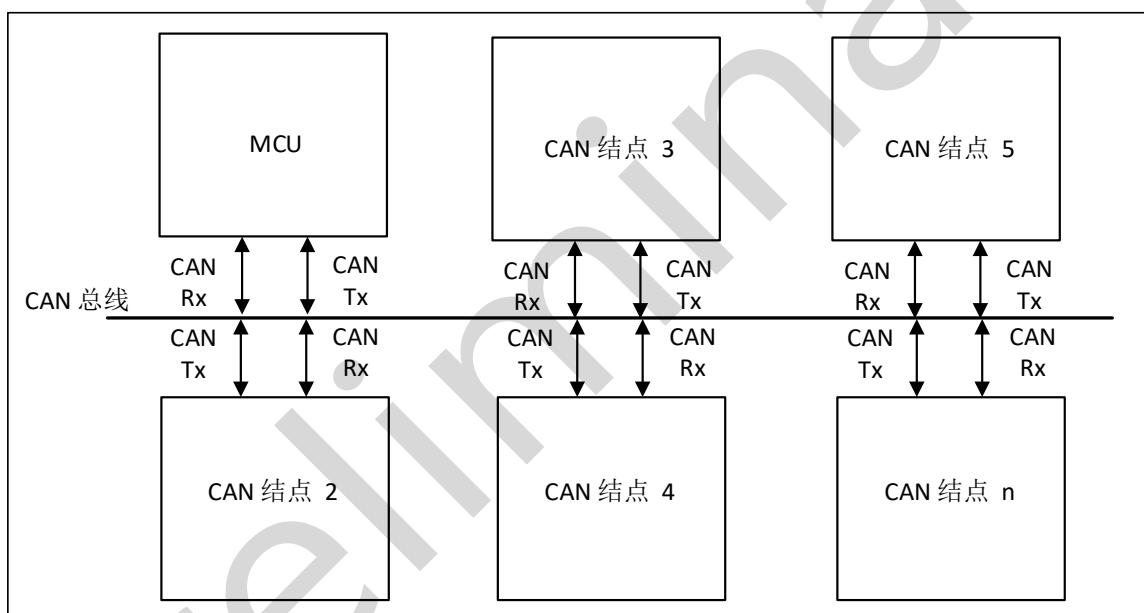


图 22-1 CAN 网络拓扑结构

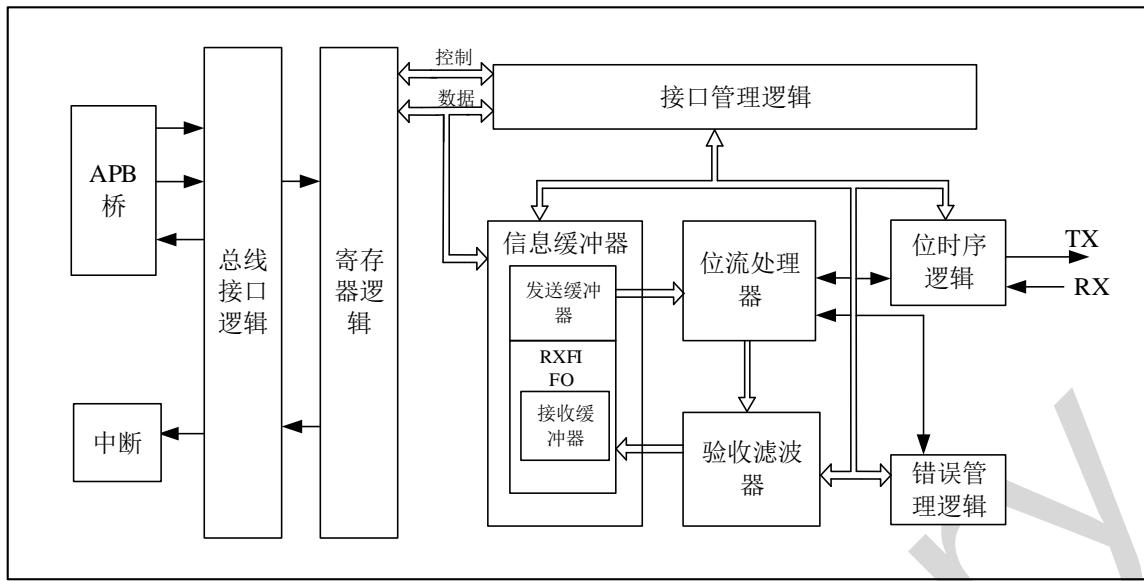


图 22-2 CAN 结构方框图

## **22.2.1 接口管理逻辑 (IML)**

接口管理逻辑解释来自 CPU 的命令，控制 CAN 寄存器的寻址向主控制器提供中断信息和状态信息。

## **22.2.2 发送缓冲器 (TX8)**

发送缓冲器是 CPU 和BSP（位流处理器）之间的接口，能够存储发送到 CAN 网络上的完整信息。缓冲器长 13 个字节，由 CPU 写入、BSP 读出。

## **22.2.3 接收缓冲器 (RX8,RXFIFO)**

接收缓冲器是验收滤波器和 CPU 之间的接口用来储存从 CAN 总线上接收和接收的信息接收缓冲器 (RXB, 13 个字节) 作为接收 FIFO (RXFIFO, 长 64 字节) 的一个窗口，可被 CPU 访问。CPU 在此 FIFO 的支持下可以在处理信息的时候接收其它信息。

## **22.2.4 验收滤波器 (ACF)**

验收滤波器把它其中的数据和接收的识别码的内容相比较，以决定是否接收信息。在纯粹的接收测试中，所有的信息都保存在 RXFIFO 中。

## **22.2.5 位流处理器 (8SP)**

位流处理器是一个在发送缓冲器、RXFIFO 和 CAN 总线之间控制数据流的程序装置。它还在 CAN 总线上执行错误检测、仲裁、填充和错误处理。

## **22.2.6 位时序逻辑 (8TL)**

位时序逻辑监视串口的 CAN 总线和处理与总线有关的位时序。它在信息开头'弱势-支配'的总线传输时同步CAN 总线位流(硬同步)，接收信息时再次同步下一次传送(软同步) BTL 还提供了可编程的时间段来补偿传播延迟时间、相位转换和定义采样点和一位时间内的采样次数。

## **22.2.7 错位管理逻辑 (EML)**

EML 负责传送层模块的错误管制。它接收 BSP 的出错报告，通知 BSP 和 IML 进行错误统计。

## **22.3 主要特征**

- 1) 支持 CAN 协议的 2.0 A 和2.0 B
- 2) 扩展的接收缓冲器 (64 字节、先进先出 FIFO)
- 3) 同时支持 11 位和 29 位识别码
- 4) 位速率可达 1Mbits/s
- 5) PeliCAN 模式扩展功能
  - a) 可读/写访问的错误计数器
  - b) 可编程的错误报警限制
  - c) 最近一次错误代码寄存器
  - d) 对每一个 CAN 总线错误的中断
  - e) 具体控制位控制的仲裁丢失中断
  - f) 单次发送 (无重发)

- g) 只听模式（无确认、无活动的出错标志）
- h) 软件位速率检测
- i) 验收滤波器扩展（4 字节代码，4 字节屏蔽）
- j) 自身信息接收（自接收请求）

## 22.4 中断

1) BasicCAN 模式里有 5 个中断：

- a) 接收中断

当接收 FIFO 不空和接收中断使能（CAN\_CR 寄存器位 1）时产生。CAN\_IR 寄存器 RI 位被置'1'。

- b) 发送中断

发送缓冲器状态从 0 变为 1（释放）和发送中断使能（CAN\_CR 寄存器位 2）时产生。

- c) 错误中断

错误中断使能（CAN\_CR 寄存器位 3）时，错误状态位或总线状态位的变化会置位此位。

- d) 数据溢出中断

大概数据溢出中断使能位（CAN\_CR 寄存器位 4）被置'1'时向数据溢出状态位'0— 1'跳变。

- e) 唤醒中断

退出睡眠模式时产生该中断。

PeliCAN 模式里有 8 个不同的中断：

- a) 接收中断

接收 FIFO 不空且中断寄存器的 RIE 位被置位时产生中断。

- b) 发送中断

发送缓冲器状态从'0— 1'（释放）跳变且中断寄存器的 TIE 位被置位时产生中断。

- c) 错误报警中断

错误状态位和总线状态位的改变和中断寄存器的 EIE 位被置位时产生中断。

- d) 数据溢出中断

数据溢出状态位有'0— 1'跳变且中断寄存器的 DOIE 位被置位时产生中断。

- e) 错误消极中断

当 CAN 控制器到达错误消极状态（至少一个错误计数器超过协议规定的值 127）或从错误消极状态又进入错误活动状态以及中断寄存器的 EPIE 位被置位时产生中断。

- f) 仲裁丢失中断

当 CAN 控制器丢失仲裁，变为接收器和中断使能寄存器的 ALIE 为被置位时产生中断。

- g) 总线错误中断

当 CAN 控制器检测到总线错误且中断使能寄存器中的 BEIE 被置位时产生中断。

h) 唤醒中断

当 CAN 控制器在睡眠模式中检测到总线的活动且中断寄存器的 WUIE 位被置'1'时产生中断。

## 22.5 功能描述

### 22.5.1 CAN 工作模式

1) CAN 控制器有 2 个主要的工作模式:

- a) BasicCAN 模式
- b) PeliCAN 模式

系统复位时默认模式是 BasicCAN 模式。

PeliCAN 模式是新的操作模式, 它能处理所有的 CAN 2.0B 规范的帧类型。而且它还提供一些增强功能使得应用于更宽的领域。

寄存器 CAN\_CDR.7 定义了 CAN 模式。如果 CDR.7 是0, CAN 控制器工作于 BasicCAN 模式。否则, CAN 控制器工作于 PeliCAN 模式。

在 Peli CAN 模式下, CAN 控制器有一个含很多功能的重组寄存器。Peli CAN 模式支持 CAN 2.0B 协议规定的所有功能 (29 字节的识别码)。下面是 PeliCAN 模式的主要新功能:

- a) 标准帧和扩展帧的接收和传送
- b) 接收 FIFO (64 字节)
- c) 在标准和扩展格式中都有单/双验收滤波器 (含屏蔽和代码寄存器)
- d) 读/写访问的错误计数器
- e) 可编程的错误限制报警
- f) 最近一次的误码寄存器
- g) 对每一个 CAN 总线错误的错误中断
- h) 仲裁丢失以及详细的位位置
- i) 一次性发送 (当错误或仲裁丢失时不重发)
- j) 只听模式 (CAN 总线监听, 无应答, 无错误标志)

### 22.5.2 Basic CAN 模式

#### 22.5.2.1 复位模式

复位模式即初始化模式。在硬件启动或总线状态设置为'1' (总线关闭) 时, 复位请求位 (CAN\_CR.0) 被置为'1' (当前)。如果这些位被软件访问, 其值将发生变化, 而且会影响内部时钟的下一个上升沿。复位请求位的变化时内部分频时钟同步的读复位请求位能够反映出这种同步状态。复位模式主要用于 CAN 通讯参数配置, 在不同工作模式下内核对 CAN 寄存器的访问权限不同。

复位请求位被设为'0'后 CAN 控制器将会等待:

- 1) 一个总线空闲信号 (11 个弱势位), 如果前一次复位请求是硬件复位或 CPU 初始复位。

- 2) 128 个总线空闲, 如果前一次复位请求是 CAN 控制器在重新进入总线开启模式前初始化总线造成的; 必须说明的是, 如果复位请求位被置位, 一些寄存器的值会被改变的。

### 22.5.2.2 工作模式

在复位模式完成后, 软件应该让硬件进入正常模式, 以便正常接收和发送报文。复位模式下, 当向复位位传送了'1— 0'的下降沿时, CAN 控制器便返回工作模式, 进行报文的发送和接收。

### 22.5.2.3 睡眠模式

睡眠模式位设为 1 (sleep), CAN 控制器将进入睡眠模式; 没有总线活动和中断等待。至少破坏这两种情况之一时将会导致睡眠模式产生唤醒中断。睡眠模式位设为低 (唤醒) 之后总线进入活动状态或中断被激活。唤醒后, 时钟启动且产生一个唤醒中断。由于总线活动唤醒的直到检测到 11 个连续的隐藏 (弱势) 位 (总线空闲序列) 后才能接收这条信息。注意在复位模式中是不能设置睡眠模式位的。清除复位模式后, 再一次检测到总线空闲时, 睡眠模式位的设置才开始有效。

表 22-1 Basic CAN 模式寄存器权限分配表

Offset	段	工作模式		复位模式	
		读	写	读	写
00	控制	控制 (FFH)	控制 命令	控制	控制
04		状态		状态	
08		(FFH)		中断	
0C		(FFH)		验收代码	验收代码
10		(FFH)		验收屏蔽	验收屏蔽
14		(FFH)		总线定时 0	总线定时 0
18		(FFH)		总线定时 1	总线定时 1
1C		(FFH)			
20		(FFH)			
24		测试	测试	测试	测试
28	数据	识别码 ( 10 ~ 3 )	识别码 ( 10~3 )	(0xFF)	
2C		识别码 ( 2~0 ) RTR 和 DLC	识别码 ( 2~0 ) RTR 和 DLC	(0xFF)	
30		DATA1	DATA1	(0xFF)	
34		DATA2	DATA2	(0xFF)	
38		DATA3	DATA3	(0xFF)	
3C		DATA4	DATA4	(0xFF)	

Offset	段	工作模式		复位模式	
		读	写	读	写
40	发送缓冲器	DATA5	DATA5	(0xFF)	
44		DATA6	DATA6	(0xFF)	
48		DATA7	DATA7	(0xFF)	
4C		DATA8	DATA8	(0xFF)	
50	接收缓冲器	识别码 (10 ~ 3)	识别码 (10 ~ 3)	识别码 (10 ~ 3)	识别码 (10~3)
54		识别码 (2 ~ 0)	识别码 (2 ~ 0)	识别码 (2 ~ 0)	识别码 (2 ~ 0)
58		RTR 和 DLC	RTR 和 DLC	RTR 和 DLC	RTR 和 DLC
5C		DATA1	DATA1	DATA1	DATA1
60		DATA2	DATA2	DATA2	DATA2
64		DATA3	DATA3	DATA3	DATA3
68		DATA4	DATA4	DATA4	DATA4
6C		DATA5	DATA5	DATA5	DATA5
70		DATA6	DATA6	DATA6	DATA6
74		DATA7	DATA7	DATA7	DATA7
78		DATA8	DATA8	DATA8	DATA8
7C		(FFH)		(FFH)	
		时钟分频器	时钟分频器	时钟分频器	时钟分频器

注：'(FFH)'代表读出数据全为1，'-'代表无写操作权限，其余表示可操作。偏移地址为0x7C时钟分频器'用于选择 BasicCAN 和 PeliCAN。

## 22.5.3 Peli CAN 模式

### 22.5.3.1 复位模式

复位模式即初始化模式。在硬件复位或总线状态位为'1'（总线关闭）时复位模式位被置为'1'（当前）。如果通过软件访问这一位，值将发生变化且下一个内部时钟（频率为外部振荡器的 1/2）的上升沿有效。复位请求位的改变和内部分频时钟同步。读复位请求位能够反映出这种同步状态。复位模式位为'0'后，CAN 控制器会等待：

- 1) 一个总线空闲信号（11 个隐藏（弱势）位），如果上一次复位是硬件复位或 CPU 初始复位。
- 2) 128 个总线空闲，如果上一次复位是 CAN 控制器在重新进入总线开启之前初始化复位。

### 22.5.3.2 工作模式

在复位模式完成后，软件应该让硬件进入正常模式，以便正常接收和发送报文。复位模式下，当检测到 CAN\_MOD 寄存器的 RM 位出现了'1—0'的下降沿，CAN 控制器便返回工作模式，进行报文的发送和接收。

### 22.5.3.3 睡眠模式

睡眠模式位（CAN\_MOD.4）设为 1（Sleep），CAN 控制器将进入睡眠模式；没有总线活动和中断等待。至少破坏这两种情况之一时将会导致睡眠模式产生唤醒中断。睡眠模式位设为低（唤醒）之后总线进入活动状态或中断被激活。唤醒后，时钟启动且产生一个唤醒中断。由于总线活动唤醒的直到检测到 11 个连续的隐藏（弱势）位（总线空闲序列）后才能接收这条信息。注意在复位模式中是不能设置睡眠模式位的。清除复位模式后，再一次检测到总线空闲时，睡眠模式位的设置才开始有效。

### 22.5.3.4 自检测模式

此模式主要用于测试。设置自检测模式位（CAN\_MOD.2）为 1，进入自检测模式。此模式可以检测所有节点，没有任何活动的节点使用自接收命令；即使没有应答，CAN 控制器也会成功发送。

### 22.5.3.5 只听模式

此主要用于测试。设置只听模式位（CAN\_MOD.1）为 1 进入只听模式。这种工作模式使 CAN 控制器进入错误消极状态。信息传送是不可能的。以软件驱动的位速检测可使用只听模式。所有其它功能都能象在正常工作模式中一样使用。

在该模式中，CAN 控制器不能在 CAN 总线上写显性位。激活错误标志或超载标志不能都写，成功接收后的应答信号也不会给出。

注：在进入只听模式之前必须进入复位模式。

Offset	工作模式		复位模式	
	读	写	读	写
00	模式	模式	模式	模式
04	(00H)	命令	(00H)	命令
08	状态		状态	
0C	中断		中断	
10	中断使能		中断使能	中断使能
14	(00H)		(00H)	
18	总线定时 0		总线定时 0	总线定时 0
1C	总线定时 1		总线定时 1	总线定时 1
20	保留			
24	检测	检测	检测	检测
28	保留		保留	
2C	仲裁丢失捕捉		仲裁丢失捕获	

Offset	工作模式			复位模式		
	读	写		读	写	
30	错误代码捕捉			错误代码捕捉		
34	错误报警限制			错误报警限制		错误报警限制
38	RX 错误计数器			RX 错误计数器		RX 错误计数器
3C	TX 错误计数器			TX 错误计数器		TX 错误计数器
40	RX 帧信息 SFF	RX 帧错误 EFF	TX 帧错误 SFF	TX 帧错误 EFF	验收代码 0	验收代码 0
44	RX 识别码 1	RX 识别码 1	TX 识别码 1	TX 识别码 1	验收代码 1	验收代码 1
48	RX 识别码 2	RX 识别码 2	TX 识别码 2	TX 识别码 2	验收代码 2	验收代码 2
4C	RX 数据 1	RX 识别码 3	TX 数据 1	TX 识别码 3	验收代码 3	验收代码 3
50	RX 数据 2	RX 识别码 4	TX 数据 2	TX 识别码 4	验收屏蔽 0	验收屏蔽 0
54	RX 数据 3	RX 数据 1	TX 数据 3	TX 数据 1	验收屏蔽 1	验收屏蔽 1
58	RX 数据 4	RX 数据 2	TX 数据 4	TX 数据 2	验收屏蔽 2	验收屏蔽 2
5C	RX 数据 5	RX 数据 3	TX 数据 5	TX 数据 3	验收屏蔽 3	验收屏蔽 3
60	RX 数据 6	RX 数据 4	TX 数据 6	TX 数据 4	保留	
64	RX 数据 7	RX 数据 5	TX 数据 7	TX 数据 5	保留	
68	RX 数据 8	RX 数据 6	TX 数据 8	TX 数据 6	保留	
6C	(FIFO RAM)	RX 数据 7		TX 数据 7	保留	
70	(FIFO RAM)	RX 数据 8		TX 数据 8	保留	

Offset	工作模式				复位模式	
	读		写		读	写
74	RX 信息计数器		RX 信息计数器			
78	RX 缓冲器起始地址		RX 缓冲器起始地址	RX 缓冲器起始地址	78	RX 缓冲器起始地址
7C	时钟分频器	时钟分频器	时钟分频器	时钟分频器	7C	时钟分频器
80	内部 RAM 地址 0 (FIFO)		内部 RAM 地址 0	内部 RAM 地址 0	80	内部 RAM 地址 0 (FIFO)
84	内部 RAM 地址 1 (FIFO)		内部 RAM 地址 1	内部 RAM 地址 1	84	内部 RAM 地址 1 (FIFO)
...	...	...	...	...	...	...
17C	内部 RAM 地址 63 (FIFO)		内部 RAM 地址 63	内部 RAM 地址 63	17C	内部 RAM 地址 63 (FIFO)
180	内部 RAM 地址 64 (TX 缓冲器)		内部 RAM 地址 64	内部 RAM 地址 64	180	内部 RAM 地址 64 (TX 缓冲器)
...	...	...	...	...	...	...
1B0	内部 RAM 地址 76 (TX 缓冲器)		内部 RAM 地址 76	内部 RAM 地址 76	1B0	内部 RAM 地址 76 (TX 缓冲器)
1B4	内部 RAM 地址 77 (空闲)		内部 RAM 地址 77	内部 RAM 地址 77	1B4	内部 RAM 地址 77 (空闲)

Offset	工作模式				复位模式	
	读		写		读	写
1B8	内部 RAM 地址 78 (空闲)		内部 RAM 地址 78	内部 RAM 地址 78	1B8	内部 RAM 地址 78 (空闲)
1BC	内部 RAM 地址 79 (空闲)		内部 RAM 地址 79	内部 RAM 地址 79	1BC	内部 RAM 地址 79 (空闲)
1C0	(00H)		(00H)		1C0	(00H)
...	...	...	...	...	...	...
1FC	(00H)		(00H)		1FC	(00H)

## 22.5.4 发送处理

根据 CAN 协议规范，报文的传输由 CAN 控制器独立完成。微控制器设置标识符，数据长度和待发送数据；然后对命令寄存器的'发送请求'位置'1'来请求发送。当 CAN 控制器正在发送报文时，发送缓冲器被写锁定。所以在防止一个新报文到发送缓冲器之前，微控制器必须检查状态寄存器的'发送缓冲器状态'标志（TBS）。

设置命令位 CMR.0 和 CMR.1 会立即产生一次信息发送，当发送错误或仲裁丢失时是不会重发的（单次发送）。只设置命令位 CMR.0 数据发送失败会重传。在自检测模式下，设置命令位 CMR.4 和 CMR.1 会立即产生一次自接收性质的信息发送。

### 22.5.4.1 中止

一个已经请求发送的报文，可以通过置位命令寄存器位的相应位执行'中止发送'，通过对 CAN\_CMR 寄存器中的 AT 位置'1'，可以中止发送请求。

当 CPU 需要当前请求发送等待时，例如：先发送一条比较紧急的信息时。但当前正在处理的传送是不停止。要想知道源信息是否成功发送，可以通过传送完毕状态位来查看。不过，这应在发送缓冲器状态位置'1'或产生发送中断后。

要注意的是，即使因为发送缓冲器状态位变为'释放'而使信息被中止，也会产生发送中断。

如果前一条指令中发送请求被置为'1'，它不能通过设置发送请求位为'0'来取消，而应通过中止发送位为'0'取消。

## 22.5.5 接收管理

接收到的报文由 CAN 控制器独立完成。收到的报文放在接收缓冲器。可以发送给微控制器的报文，由状态寄存器的接收缓冲器状态标志'RBS'和接收中断标志'RI'标出。

### 22.5.5.1 查询控制接收

微控制器读 CAN 控制器的状态寄存器，检查接收缓冲状态（RBS）查看是否收到一个报文。

当读到 RBS 位为 1，表示收到一个或多个报文，微控制器从 CAN 中取得报文，然后置位命令寄存器的响应标志位'RRB'发送一个释放接收缓冲器命令。

### 22.5.5.2 中断控制接收

中断使能标志位位于 CAN 控制器寄存器里（对于BasicCAN 模式）或位于中断使能寄存器里（对于PeliCAN 模式）。如果 CAN 控制器已接收一个报文，而且报文已经通过验收滤波器并放在接收 FIFO，那么会产生一个接收中断。进入中断服务程序，微控制器取走报文，然后置位命令寄存器的响应标志位'RRB'发送一个释放接收缓冲器命令。

### 22.5.5.3 溢出

在接收 FIFO 满了但还接收其他的报文的时候就会导致溢出，同时置位状态寄存器中的数据超载状态位（如果使能）通知微控制器有数据溢出的情况，CMR.3 位置'1'可清除溢出状态。

CAN\_CMR 寄存器的 ERB 位可用于清除 FIFO 当使能 ERB 后，若接收 FIFO 发生溢出，硬件会自动清除接收 FIFO。

### 22.5.5.4 有效报文

根据CAN 协议，当报文被正确接收（直到EOF 域的最后一位都没有错误），且通过了标识符过滤，那么该报文被认为是有效报文。

CAN\_CMR 寄存器中的 RRB 位是用于清除由数据溢出状态位指出的数据溢出情况。

### 22.5.6 标识符过滤

独立的 CAN 控制器装配了一个多功能的验收滤波器，该滤波器允许自动检查标识符和数据字节。滤波器由验收代码寄存器和屏蔽寄存器根据给定算法来控制。接收到的数据会和验收代码寄存器中的值进行逐位比较。接收屏蔽寄存器定义与比较相关的位置（0 = 相关，1 = 不相关）。

在 CAN 总线上，接收者检测所有的广播报文，只有收到报文的标识符与验收代码寄存器相应的相关位相同，报文才会被接收。使用这些有效的滤波方法，可以防止对于某个节点无效的报文或报文组存储在接收缓冲器里。因此降低了微控制器的处理负载。

### 22.5.6.1 8asicCAN 模式里的验收滤波器

该滤波器是由两个寄存器—验收码寄存器（ACR）和验收屏蔽寄存器（AMR）控制。CAN 报文符的高 8 位和这些寄存器里值相比较。可以定义若干个组的标识符为被任何一个节点接收。

例子：验收码寄存器（ACR）包括：

	MSB				LSB			
验收码寄存器（ACR）包括：	0	1	1	1	0	0	1	0
验收屏蔽寄存器（AMR）包括：	0	0	1	1	1	0	0	0
带有11位标识符信息被接收 (X=无关)	0	1	X	X	X	0	1	0
	ID.10							

### 图 22-3 CAN 标识符接收示例

在验收屏蔽寄存器里是'1'的位置上，标识符相应的位可以是任意值。这对于三个最低位也一样。因此在这个例子里可以接收 64 个不同的标识符。标识符其他的位必须等于验收代码寄存器相应位的值。

#### 22.5.6.2 PeliCAN 模式里的验收滤波器

在验收滤波器的帮助下，只有当接收信息中的识别位和验收滤波器预定义的值相等时，CAN 控制器才允许将已接收信息存入 RXFIFO。PeliCAN 模式中，验收滤波器由验收代码寄存器（ACRn）和验收屏蔽寄存器（AMRn）定义。要接收的信息的位模式在验收代码寄存器中定义。相应的验收屏蔽寄存器允许定义某些位为'不影响'（即可为任意值）。有两种不同的过滤模式可在模式寄存器中的位 3 选择：

- 1) 单滤波器模式 <sup>(1)</sup>
- 2) 双滤波器模式 <sup>(0)</sup>

#### 22.5.6.3 单滤波器配置

这种滤波器配置可以定义一个长滤波器（4 字节）。滤波器字节和信息字节之间位的对应关系取决于当前接收帧格式。

标准帧：如果接收的是标准帧格式的信息，在验收滤波中只使用前两个数据字节来存放包括 RTR 位的完整的识别码，后两个字节存放数据字节信息。如果由于置位 RTR 位而导致没有数据字节，或因为设置相应的数据长度代码而没有或只有一个数据字节，信息也会被接收的。对于一个成功接收的信息，所有单个位的比较后都必须发出接收信号。

注：ACR1 和 AMR1 的低四位是不用的'为了和未来产品兼容'这些位可通过设置 AMR1.3、AMR1.2、AMR1.1、AMR1.0 为 1 而定为'不影响'。

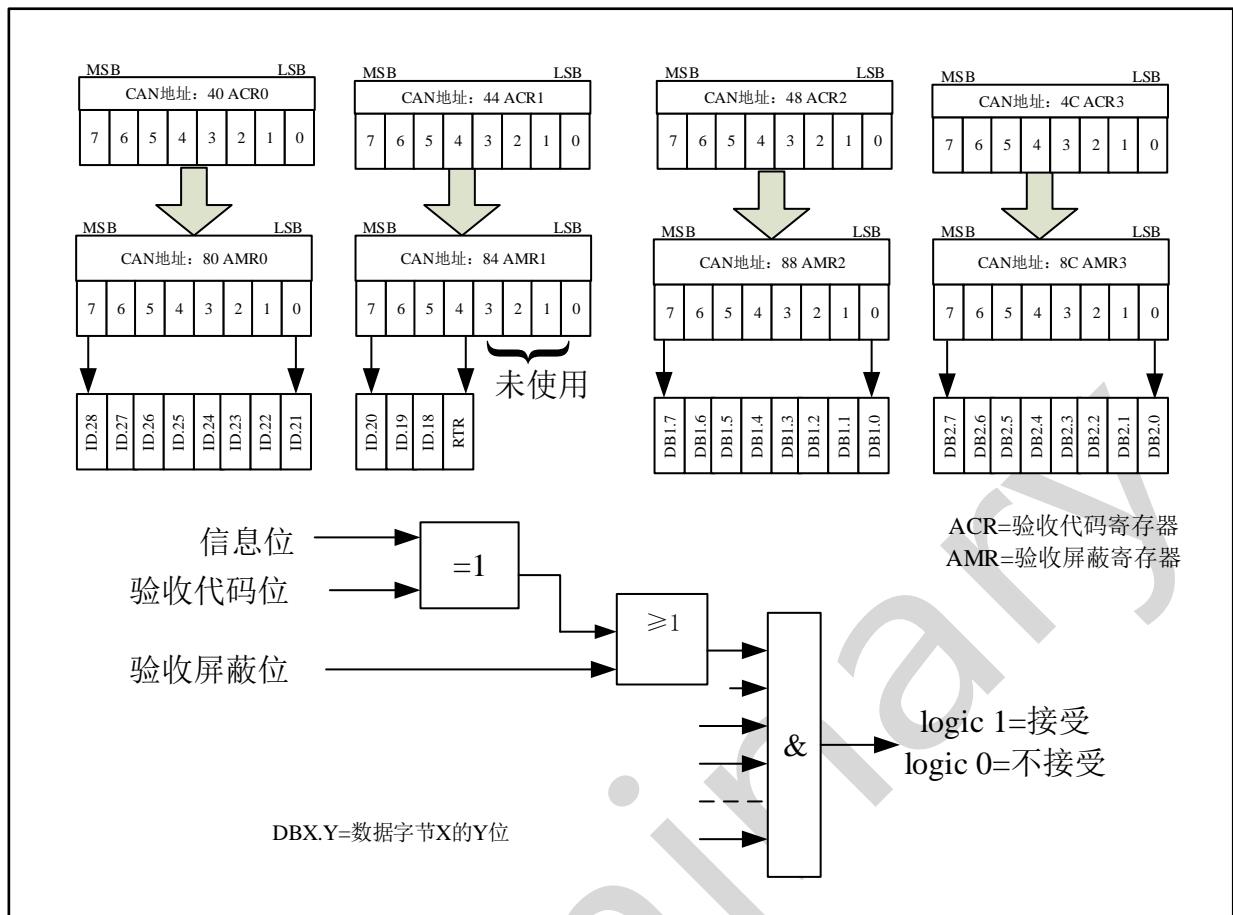
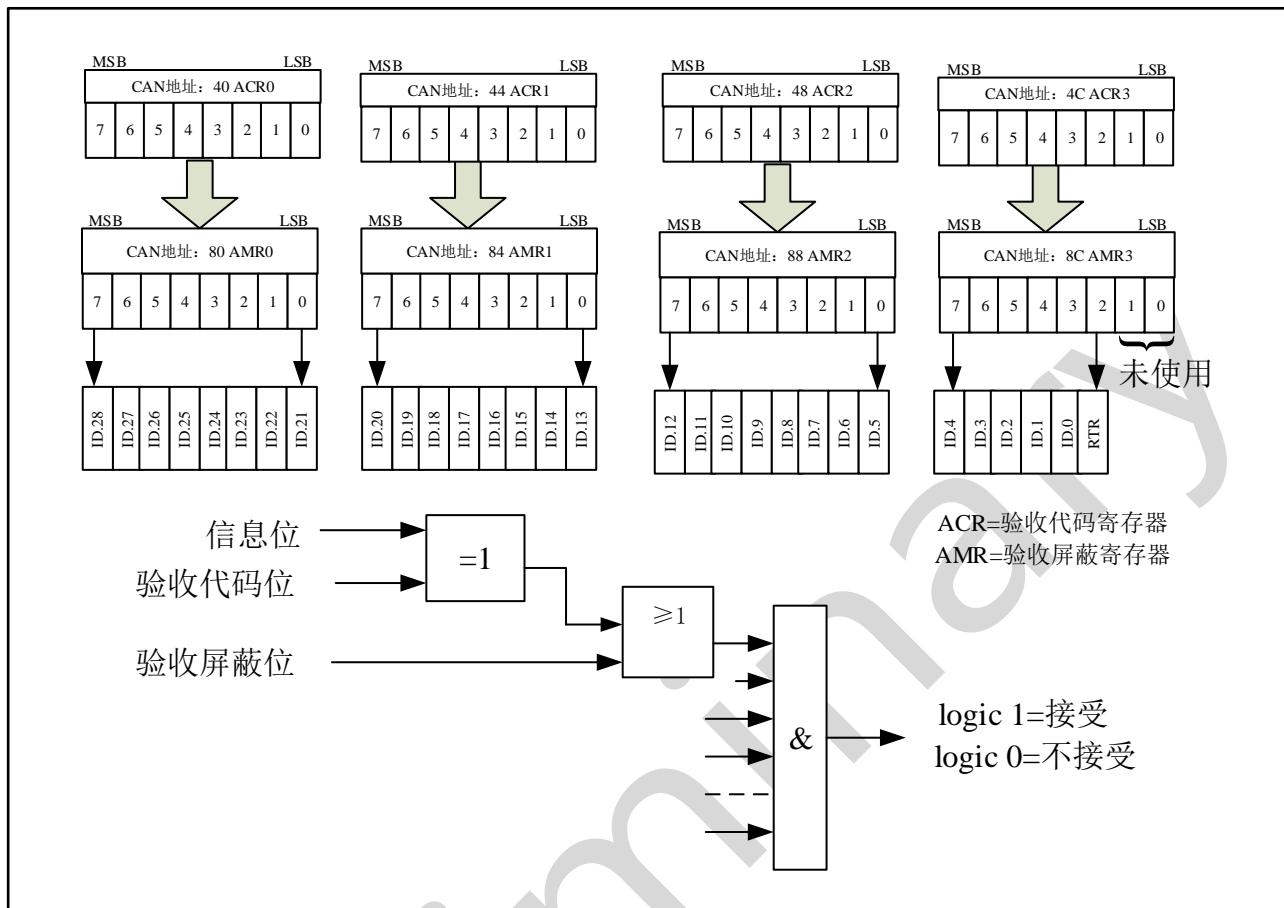


图 22-4 接收标准结构信息时的单个滤波器配置

扩展帧：如果接收的信息是扩展帧格式的，包括 RTR 位的全部识别码将被接收过滤使用。为了成功接收信息，每个位都必须比较通过。

注：AMR3 的最低两位和 ACR3 是不用的。这些位应该通过置位 AMR3.1 和 AMR3.0 来定为“不影响”。

图 22-5 单滤波器配置，接收扩展帧信息



#### 22.5.6.4 双滤波器的配置

这种配置可以定义两个短滤波器。一条接收的信息要和两个滤波器比较来决定是否放入接收缓冲器中。至少有一个滤波器发出接收信号，接收的信息才有效。滤波器字节和信息字节之间位的对应关系取决于当前接收的帧格式。

**标准帧：**如果接收的是标准帧信息，被定义的两个滤波器是不一样的。第一个滤波器比较包括 RTR 位的整个标准识别码和信息的第一个数据字节。第二个滤波器只比较包括 RTR 位的整个标准识别码。

为了成功接收信息，所有单个位的比较时应至少有一个滤波器表示接收。RTR 位置位或数据长度代码是 0 时表示没有数据字节存在。无论怎样，只要从开始到 RTR 位的部分都被表示接收，信息就可以通过滤波器 1。

如果没有向滤波器请求数据字节过滤，AMR1 和 AMR3 的低四位必须被置为'1'(不影响)。当使用包括 RTR 位的整个标准识别码时，两个滤波器都同样工作。

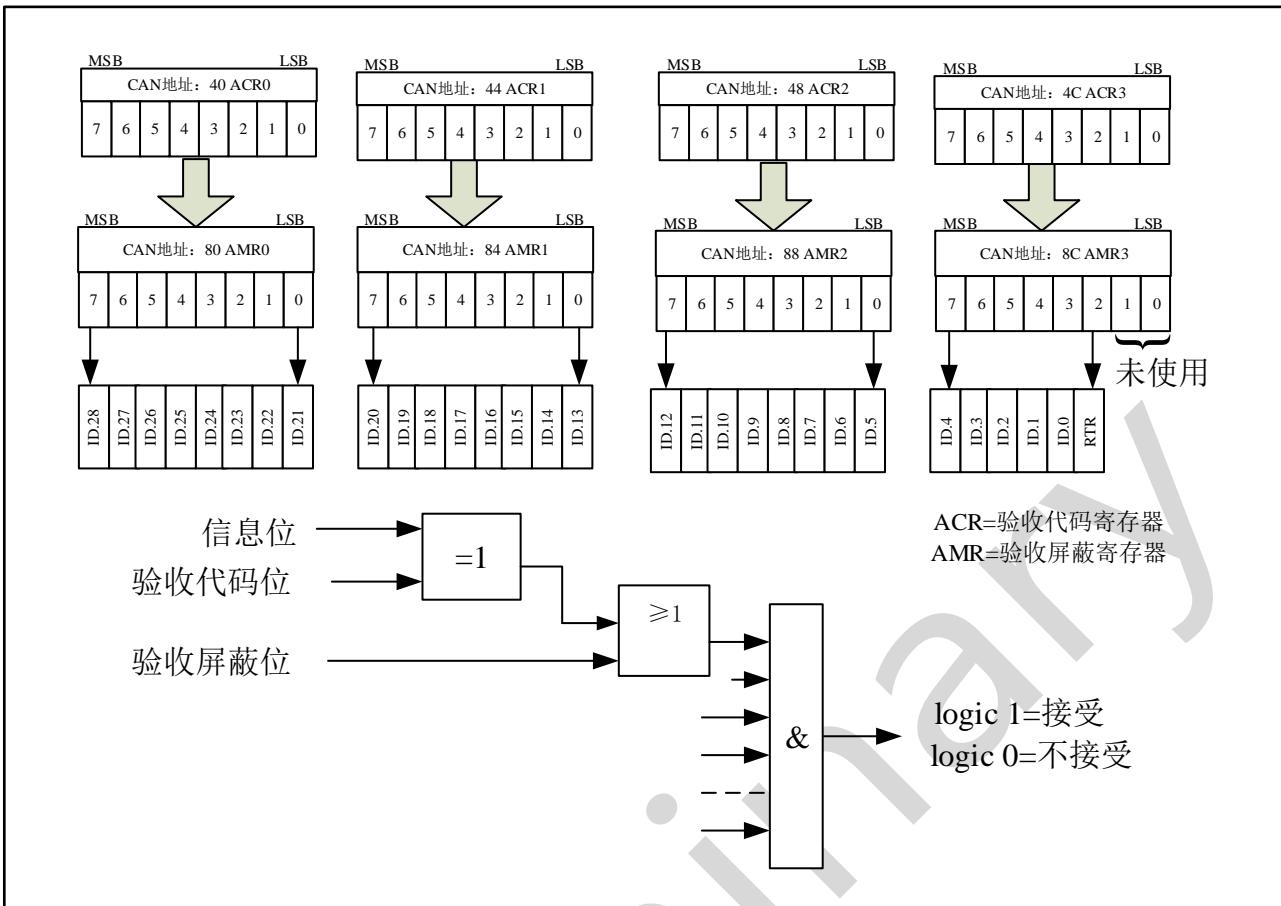


图 22-6 接收标准结构信息时的双个滤波器配置

扩展帧：如果接收到扩展帧信息，定义的两个滤波器是相同的。两个滤波器都只比较扩展识别码的前两个字节。

为了能成功接收信息，所有单个位的比较时至少有一个滤波器表示接收。

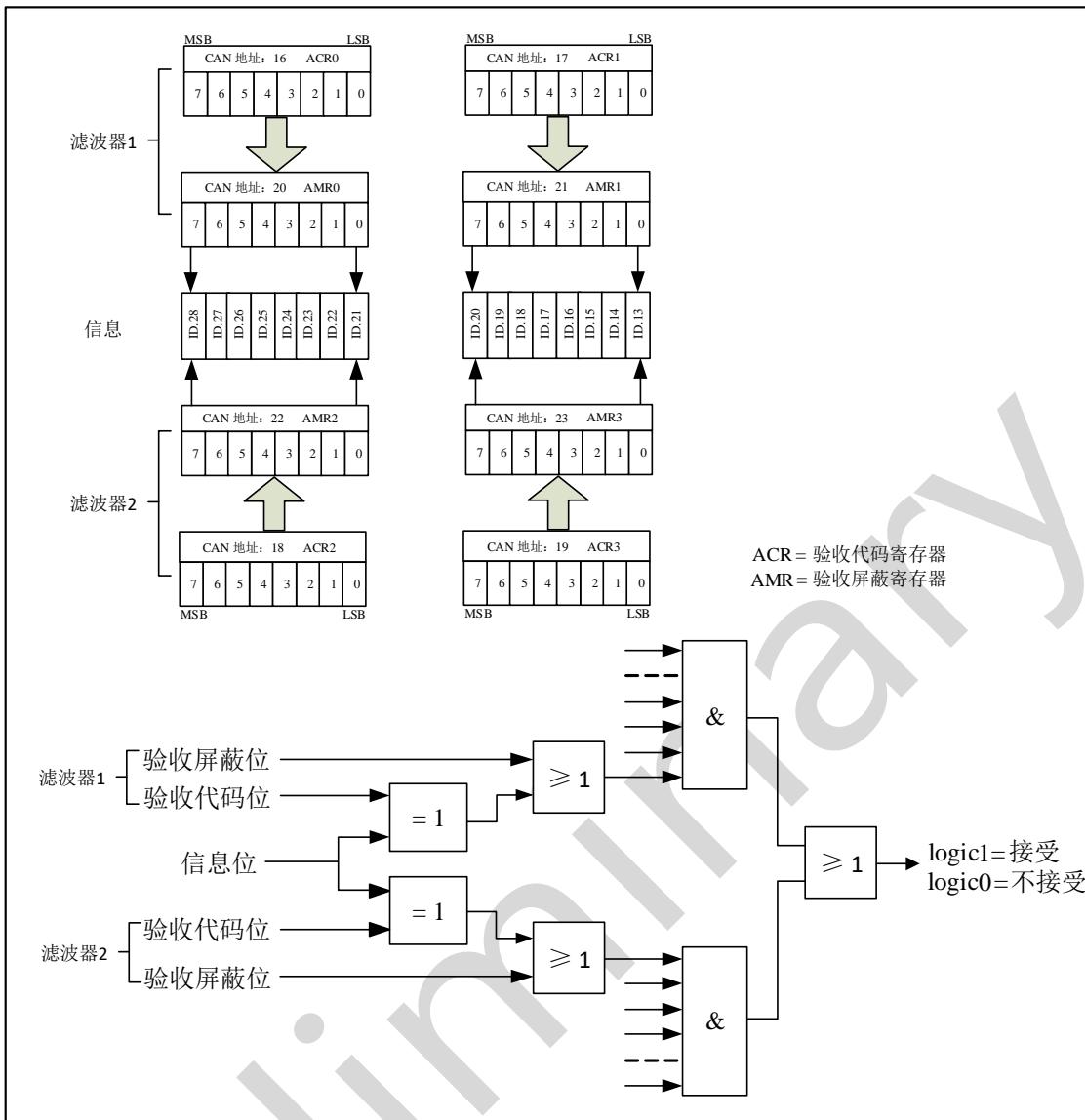


图 22-7 双滤波器配置，接收扩展帧信息

例 1：假设下面 1 个标准帧报文要在 PeliCAN 模式里滤波，可以通过使用一个长滤波器完成（单滤波器模式）。

验收代码寄存器 (ACRn) 和验收屏蔽寄存器 (AMRn) 包括：

n	0	1(高四位)	2	3
ACRn	01XX X010	XXXX	XXXX XXXX	XXXX XXXX
AMRn	0011 1000	1111	1111 1111	1111 1111
接收的报文 (ID.28 ~ ID.18, RTR)	01xx x010 xxxx			

X\* = 不相关 \*x\* = 任意值'只使用了 ACR1 和 AMR1 的高四位。

例2：假设下面 2 个有标准帧标识符的报文在标识符不用进一步译码就被接收。数据和远程帧必须被正确接收。数据字节不要求验收滤波。

报文 1: (ID.28) 1011 1100 101 (ID.18)

报文2: (ID.28) 1111 0100 101 (ID.18)

使用单滤波模式可以接收到四个报文而不仅是要求的两个：

n	0	1(高四位)	2	3	
ACRn	1X11 X100	101X	XXXX XXXX	XXXX XXXX	
AMRn	0100 1000	0001	1111 1111	1111 1111	
接收的报文 (ID28 ~ ID.18, RTR)		1011 0100 101x 1111 0100 101x (报文 2) 1011 1100 101x (报文 1) 1111 1100 101x			

('X'= 不相关 'x'= 任意值'只使用了 ACR1 和 AMR1 的高四位。)

这个结果需要进一步解码才能满足接收两条信息的要求。 使用双滤波器可以得到正确的结果

滤波器 1			滤波器 2		
n	0	1	3 低四位	2	3 高四位
ACRn	1011 1100	101X XXXX	... XXXX	1111 0100	101X ...
AMRn	0000 0000	0001 1111	... 1111	0000 0000	0001 ...
接收的信息 (ID28 ~ ID.18, RTR)		1011 1100 101X (报文 1)			
		1111 0100 101X (报文 2)			

('X'= 不相关 'x'= 任意值)

报文 1 被滤波器 1 接收，报文 2 被滤波器 2 接收。如果报文至少被两个滤波器中的一个接收，报文就被存放到接收 FIFO。这种方法可满足于这种要求。

例 3：在这个例子里，使用一个长的验收滤波器过滤一组带有扩展标识符的报文。

n	0	1	2	3(高六位)	
ACRn	1011 0100	1011 000X	1100 XXXX	0011 0XXX	
AMRn	0000 0000	0001 0001	0000 1111	0000 0111	
接收的信息 (ID28 ~ ID.18, RTR)		1011 0100 101x 000x 1100 xxxx 0011 0x			

('X'= 不相关 'x'= 任意值'只使用了 ACR1 和 AMR1 的高六位。)

例 4：有些使用标准帧系统仅用 11 位标识符和头两个数据字节识别报文。如果报文超过 8 个数据字节，头两个数据字节定义为报文头和使用分段存储协议就会使用像这样的协议。

例如 DeviceNet。对于这种系统类型，CAN 控制器除了 11 位标识符和 RTR 位外，在单滤波器模式里能滤波两个数据字节，在双滤波器模式里能过滤一个数据字节（除了 11 位标识符和 RTR 位）。

下面的例子显示了用双滤波器模式，在这种系统里有效地滤波报文：

滤波器 1			滤波器 2		
n	0	1	3 低四位	2	3 高四位
ACRn	1110 1011	0010 1111	... 1001	1111 0100	XXX0 ...
AMRn	0000 0000	0000 0000	... 0000	0000 0000	1110 ...

滤波器 1		滤波器 2	
接收的信息 (ID28 ~ ID.18, RTR)	1110 1011 0010 + 1111 ... 1001 标识符 RTR + 头一个数据字节	1111 0100 标识符	xxx0 RTR

('X' = 不相关 'x' = 任意值)

1) 滤波器 1 滤波的报文有:

- a) - 标识符'11101011001'
- b) RTR = '0'; 也就是说是数据帧
- c) 数据字节'11111001' (这是指例如DeviceNet: 一个信息的所有段都被过滤)

滤波器 2 用来过滤一组 8 个报文, 其中报文有

- a) - 标识符'11110100 000'到 11110100111'
- b) RTR = '0'; 也即是数据帧

### 22.5.6.5 过滤器组模式设置

过滤器组共 20 组, 可以通过配置相应的 CAN\_FGAx (x=0 ~ 2) 来打开和禁用某些过滤器组, 应用程序中不用的过滤器组, 应该保持在禁用状态;

通过配置 FGAx (x = 0 ~ 19) 选择过滤器组单/双滤波模式;

验收代码寄存器和验收屏蔽寄存器 AMR0, AMR1, AMR2, AMR3, ACR0, ACR1, ACR2, ACR3 共 20 组;

每组滤波器功能的使用方式相同, 可单组使用也可多组使用。

### 22.5.7 报文存储

要在 CAN 总线上发送的数据被载入 CAN 控制器的存储区, 这个存储区叫'发送缓冲器'。从 CAN 总线上收到的数据也存储在 CAN 控制器的存储区, 这个存储区叫'接收缓冲器'。这些缓冲器包括 2, 3 或 5 个字节的标识符和帧信息 (取决于模式和帧类型), 而最多可以包含 8 个数据字节。

#### 22.5.7.1 BasicCAN 模式

缓冲器长达 10 个字节

- 1) 2 个标识符字节
- 2) 8 个数据字节

表 22-2 BasicCAN 模式里的 RX 和 TX 缓冲器

相对 CAN 偏移量		寄存器		组成与注释
TX (16 进制)	RX (16 进制)	TX	RX	
28	50	CAN_TXIDR1	CAN_RXIDR1	8 位标识符
2C	54	CAN_TXIDR2	CAN_RXIDR2	3 位标识符, 1 位远程传输请求位, 4 位数据长度代码, 表示数据字节的数量
30	58	CAN_TXDR1	CAN_RXDR1	由数据长度代码表示, 最多 8 个数据字节
34	5C	CAN_TXDR2	CAN_RXDR2	

相对 CAN 偏移量		寄存器		组成与注释
TX (16 进制)	RX (16 进制)	TX	RX	
38	60	CAN_TXDR3	CAN_RXDR3	由数据长度代码表示，最多 8 个数据字节
3C	64	CAN_TXDR4	CAN_RXDR4	
40	68	CAN_TXDR5	CAN_RXDR5	
44	6C	CAN_TXDR6	CAN_RXDR6	
48	70	CAN_TXDR7	CAN_RXDR7	
4C	74	CAN_TXDR8	CAN_RXDR8	

### 22.5.7.2 Pelican 模式

这些缓冲器是 13 个字节长

- 1) 1 字节帧信息
- 2) 2 个或 4 个标识符字节（标准帧或扩展帧）
- 3) 最多 8 个数据字节

### 22.5.7.3 发送缓冲器

发送缓冲器的全部列表见下图。请务必分清标准帧格式（SFF）和扩展帧格式（EFF）配置。发送缓冲器允许定义长达 8 个数据字节发送信息。



图 22-8 标准帧和扩展帧格式配置在发送缓冲器的列表

帧信息中的 FF 位决定 CAN 控制器将要发送扩展帧格式还是标准帧格式。

#### 22.5.7.4 接收缓冲器

接收缓冲器的列表与发送缓冲器很相似。接收缓冲器是 RXFIFO 的可访问部分，位于 CAN

地址的 40 ~ 70。每条信息都分为描述区和数据区。

注：在帧信息字节中的接收字节长度代码代表实际发送的数据长度代码'它有可能大于 8 (取决于发送器)。无论如何'最大接收数据字节数是 8。这一点在读接收缓冲器中的信息时应当考虑。

见下阳，RXFIFO 共有 64 个信息字节的空间。一次可以存储多少条信息取决于数据的长度。如果 RXFIFO 中没有足够的空间来存储新的信息，CAN 控制器会产生数据溢出条件，此时信息有效且接收检测为肯定。发生数据溢出情况时，已部分写入 RXFIFO 的信息将被删除。这种情况可以通过状态寄存器和数据超限中断（中断允许）反应到 CPU。

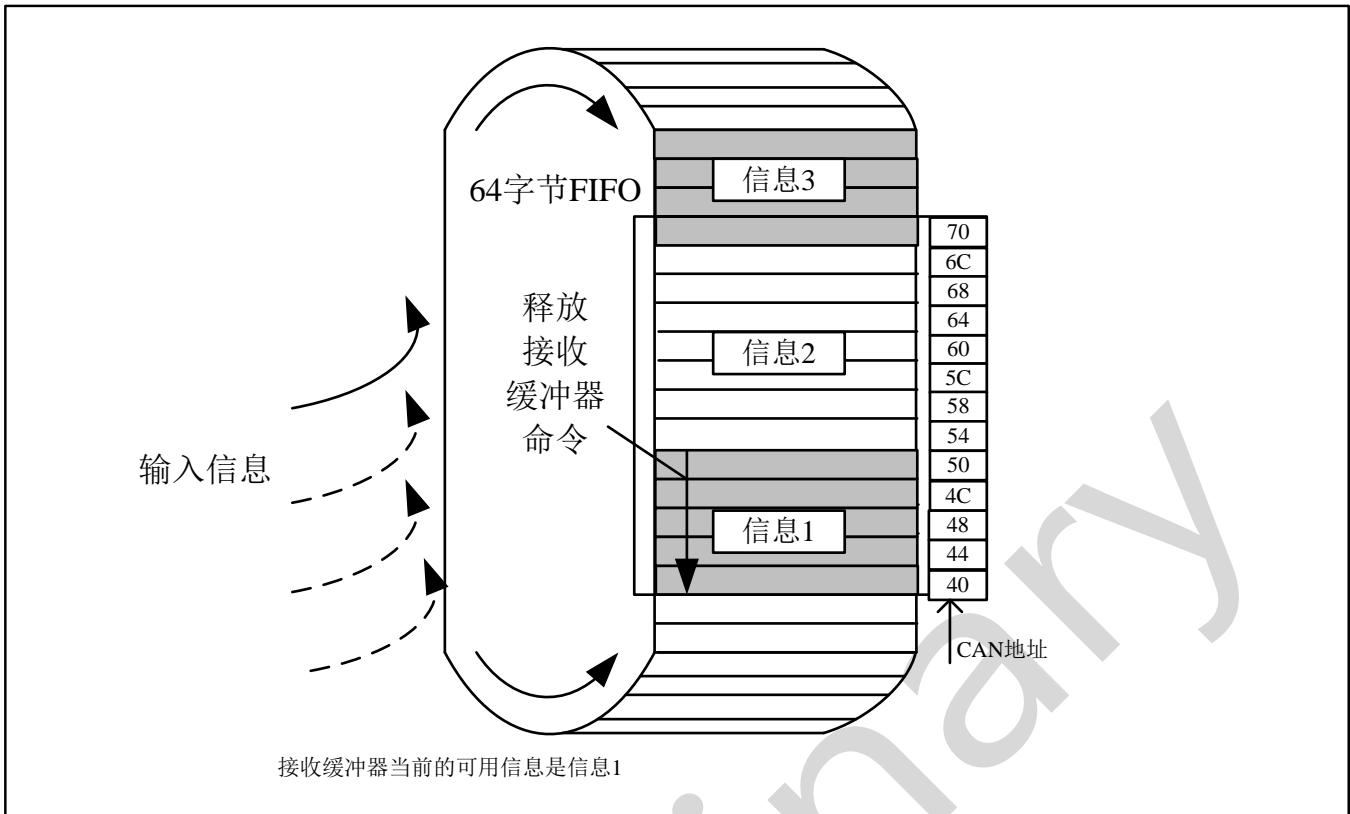


图 22-9 RXFIFO 中的信息存储举例

## 22.5.8 出错管理

基于错误计数器的值，每个 CAN 控制器能够在三种错误状态之一中工作：错误激活、错误认可或总线离线。如果错误计数器的值都在 0 ~ 127 之间，CAN 控制器是错误激活的。此时产生错误激活标志（6 个显性位）。如果一个错误计数器的值在 128 ~ 255 之间，CAN 控制器是错误认可的。此时，在检测到错误前，产生认可错误标志（6 个隐性位）。如果发送错误计数器的值高于 255，则到达总线离线状态。在这种状态下，自动置位复位请求，CAN 控制器对总线没有影响。总线离线状态只能在微控制器用命令‘复位请求 = 0’退出。这将启动总线离线恢复定时器，发送错误计数器计数 128 个总线释放信号。计数结束后，两个错误计数器都是 0，器件再次处于错误激活状态。

### 22.5.8.1 错误计数器

如上面描述，CAN 的错误状态和发送错误计数器和接收错误计数器的值直接相关。

为了仔细研究错误界定，支持 CAN 控制器的增强的错误分析功能，CAN 控制器提供可读的错误计数器。另外，在复位模式，允许对于两个错误计数器进行写访问。

### 22.5.8.2 出错中断

有三个中断源来向微处理器发出错的状态。每个中断都能在中断使能寄存器里分别使能。

- 1) 总线出错中断：在 CAN 总线上检测到任何一个错误都会产生中断
- 2) 出错警告中断 如果超过出错警告界限，产生出错警告中断。而且它在 CAN 控制器进入总线离线状态和再次之前再一次进入错误激活状态也会产生这个中断。CAN 控制器的出错警告界限在复位模式中可编程。复位后的默认值是 96。
- 3) 错误认可中断 如果错误状态从错误激活变成错误认可或相反，将产生错误认可中断。

### 22.5.8.3 错误码捕捉

CAN 控制器可以执行在 CAN2.0B 规范定义的所有错误界定。每个 CAN 控制器处理错误的整个过程是完全自动的。但是，为了向用户提供某个错误的详细信息，CAN 控制器提供了错误代码捕捉功能。无论什么时候发生 CAN 总线错误，它都会强制产生相应的总线出错中断。同时，当前位的位置被捕捉入错误代码捕捉寄存器。在主控制器将捕捉的数据读出前，它都会被保存在寄存器中。然后捕捉机制再次激活。寄存器可以内容区分四种错误类型：格式出错、填充出错、位出错和其他错误。如下图表示，寄存器还另外表明在错误是在报文的接收还是发送期间发生。这个寄存器中的五个位表示 CAN 帧内错误的位位置，更多信息参考下面的表和数据表。

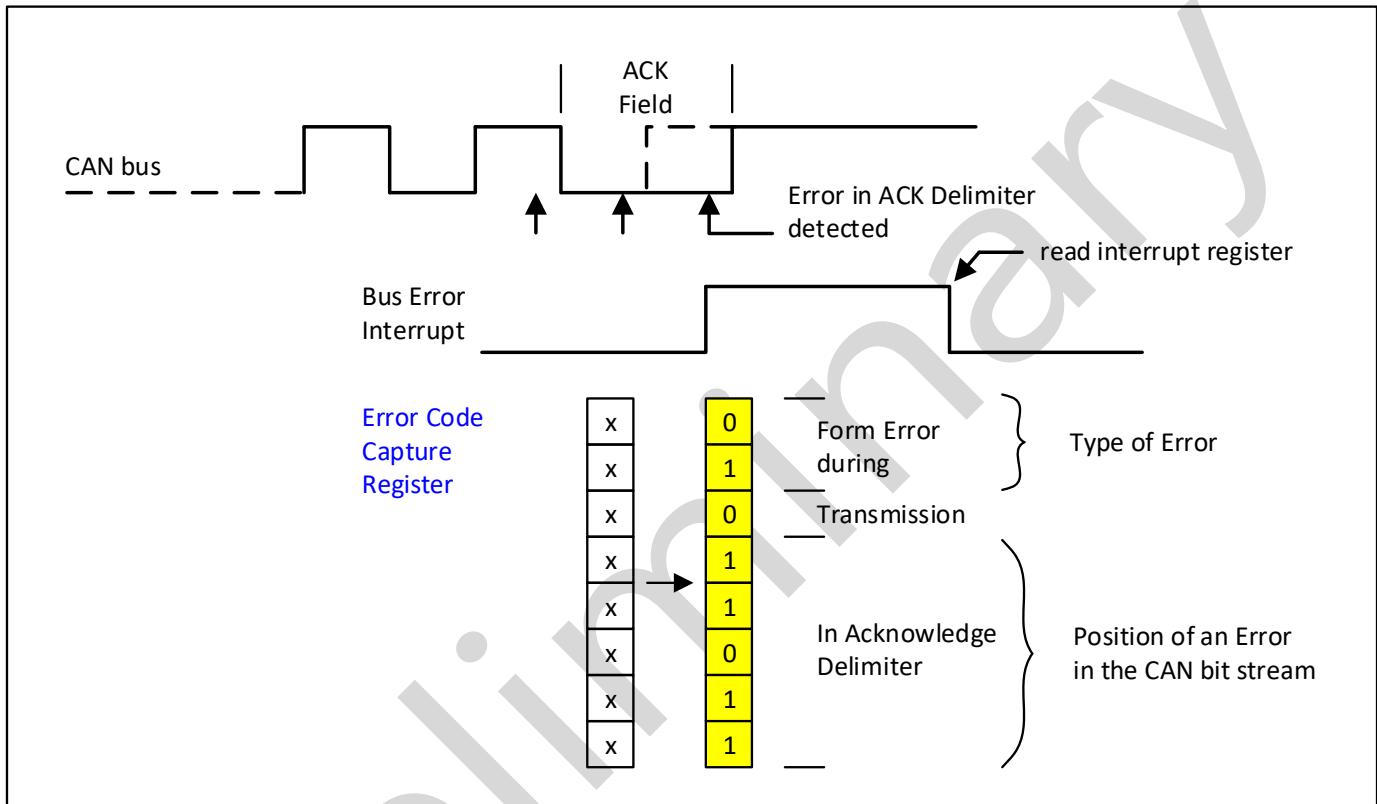


图 22-10 错误码捕捉功能举例

CAN 规范定义了：CAN 总线上的每个位只有特殊类型的错误。下面两张显示了 CAN 报文发送和接收期间可能出现的所有错误。左边的部分包括位置和错误的类型，这些由错误码 捕捉寄存器捕捉。每张表的右边部分是将错误码转换成上层的错误描述，可以直接从寄存 器内容知道其含义。通过使用这些表格，能得到有关错误计数器的变化和在器件发送和接 收管脚的错误状态的更多信息。使用这些表时，例如在错误分析软件里，可以详细地分析 每个错误状态。关于 CAN 错误类型和位置的信息能用于错误统计和系统维护或在系统优化器件进行纠正。

表 22-3 接收时可能出现的错误

CAN 位流里的错误位置	错误类型	RX 错误计数	错误码捕捉	描述
标识符 SRR、IDE 和 RTR 位 保 留 位 数据长度码 数据场 CRC 序列	填充	+1	收到 5 个电平相同的连续位	
CRC 定界符	格式填充	+1 +1	RX = 显性 收到超过 5 个电平相同的连续的位	位必须是隐性
应答位	位	+1	RX = 显性, 或检测到 CRC 错误	临界的总线定时或总线长度 CRC 序列不正确
应答定界符	格式	+1	RX = 显性, 或检测到 CRC 错误	临界的总线定时或总线长度 CRC 序列不正确
帧结束	格式其他	+1 ±0	RX = 头六位是显性 RX = 最后一位的显性	反应: 发出超载标志, 如果发送器重新发送, 数据可能重复
间隔	其他	±0	RX = 显性	反应: 接收器发出超载标志
激活错误标志	位	+8	TX = 显性, 但 RX = 隐性	不能写显性位
容许的显性位	其他	+8	TX = 显性, 但 RX = 隐性	不能写显性位
错误定界符	格式 其他	+8	TX = 显性, 但 RX = 隐性	不能写显性位
超载标志	位	+8	TX = 显性, 但 RX = 隐性	不能写显性位

表 22-4 发送时可能出现的错误

	错误类型	TX 错误计数	错误码捕捉	描述
帧开始	位	+8	TX = 显性, 但 RX = 隐性	不能写显性位
标识符 SRR 位	位 填充	+8 ±0	TX = 显性, 但 RX = 隐性 TX = 隐性, 但 RX = 显性	不能写显性位

	错误类型	TX 错误计数	错误码捕捉	描述
IDE 和 RTR 位	位 填充	+8 ±8	TX = 显性, 但 RX = 隐性 TX = 隐性, 但 RX = 显性	不能写显性位
保 留 位 数据长度码 数据场				
CRC 序列	位	+8	TX = 显性, 但 RX = 隐性	不能写显性位
CRC 定界符	格式	+8	RX = 显性	位必须是隐性
应答隙	其他其他	+8 ±0	RX = 隐性 (错误激活) RX = 隐性 (错误认可)	没有应答 没有应答, 节点可能单独在总线上
应答应界符	格式	+8	RX = 显性	临界的总线定时或总线长度
帧结束	格式其他	+8 +8	RX = 头六位是显性 RX = 最后一位是显性位	帧已经被一些节点接收, 再次发送可能导致接收器里数据重复
间隔	其他	±0	RX = 显性	来自于“旧”CAN 控制器的超载标志
激活错误标志 过载标志	位	+8	TX = 显性, 但 RX = 隐性	不能写显性位
允许显性位	格式	+8	RX = 在激活错误标志或过载标志后有超过 7 个显性位	
错误定界符	格式其他	+8 ±0	RX = 头七位是显性位 RX = 定界符的最后一位是显性位	
认可错误标志	其他	+8	RX = 显性 (错误认可)	没有收到应答, 节点不是单独在总线上。

#### 22.5.8.4 离线恢复

如果发送错误计数器的值高于 255, 则达到总线离线状态。总线状态位被置为'1'。在这种状态下, 自动置位复位请求位, CAN 控制器对总线没有影响。在错误中断允许的情况下, 会产生一个错误中断。这种状态会持续到 CPU 清除复位请求位。所有这些完成后, CAN 控制器将会等待协议规定的最长时间 (128 个总线空闲信号)。

## 22.5.9 位时序控制

位时序控制是通过采样的方式监视串行的 CAN 总线，其采样点可以通过同步帧起始位的边沿，及重新同步后面的边沿来调整。

名义上的每位的时间可以分为 3 段：

- 1) 同步时段 ( $t_{SYNCSEG}$ )：通常期望位的变化发生在该时间段内。其值固定为 1 个时间单元 ( $1 \times t_{CAN}$ )。
- 2) 第一时间段 ( $t_{TSEG1}$ )：定义采样点的位置。它包含 CAN 标准里的 **PROP\_SEG** 和 **PHASE\_SEG1**。其值可以编程为 1 到 16 个时间单元，但也可以被自动延长，以补偿因为网络中不同节点的频率差异所造成的相位的正向漂移。
- 3) 第二时间段 ( $t_{TSEG2}$ ) 定义发送点的位置。它代表 CAN 标准里的 **PHASE\_SEG2**。其值可以编程为 1 到 8 个时间单元，但也可以被自动缩短以补偿相位的负向漂移。

CAN 系统时钟  $t_{SCL}$  的周期是可编程的，而且决定了相应的位时序。

CAN 系统时钟由如下公式计算： $t_{SCL} = 2 \times t_{CLK} \times (BRP + 1)$ 。这里  $t_{CLK} = APB1$  的频率周期同步跳跃宽度 (SJW) 定义了，在每位中可以延长或缩短多少个时间单元的上限。为了补偿在不同总线控制器的时钟振荡器之间的相位偏移，任何总线控制器必须在当前传送的相关信号边沿重新同步。同步跳转宽度定义了每一位周期可以被重新同步缩短或延长的时钟 周期的最大数目  $t_{SJW} = t_{SCL} \times (SJW + 1)$  第一时间段 (TSEG1) 和第二时间段 (TSEG2) 决定了每一位的时钟数目和采样点的位置，这里：

$$\begin{aligned}t_{SYNCSEG} &= 1 \times t_{SCL} \\t_{TSEG1} &= t_{SCL} \times (TSEG1 + 1) \\&= t_{SCL} \times (TSEG2 + 1)\end{aligned}$$

有效跳变被定义为，当 CAN 自己没有发送隐性位时，从显性位到隐性位的第 1 次转变。如果在第一时间段 ( $t_{TSEG1}$ ) 而不是在同步时段 ( $t_{SYNCSEG}$ ) 检测到有效跳变，那么  $t_{TSEG1}$  的时间就被延长最多 SJW 那么长，从而采样点被延迟了。

相反如果在第二时间段 ( $t_{TSEG2}$ ) 而不是在  $t_{SYNCSEG}$  检测到有效跳变，那么  $t_{TSEG2}$  的时间就被缩短最多 SJW 那么长，从而采样点被提前了。

为了避免软件的编程错误，对位时序控制寄存器 (CAN\_BTR) 的设置，只能在 CAN 处于初始化状态下进行。

$$\text{CAN 波特率} = APB1 / (2 * (BRP + 1) * (TSEG1 + 1 + TSEG2 + 1 + 1));$$

## 22.5.10 仲裁丢失

仲裁丢失时，会产生相应的仲裁丢失中断（中断允许）。同时，位流处理器的当前位位置被捕获送入仲裁丢失捕捉寄存器。一直到用户通过软件读这个值，寄存器中的内容都不会变。随后，捕捉机制又被激活了。

读中断寄存器时，中断寄存器中相应的中断标志位被清除。直到仲裁丢失捕捉寄存器被读一次之后，新的仲裁丢失中断才有效。

下图为仲裁丢失位解释:

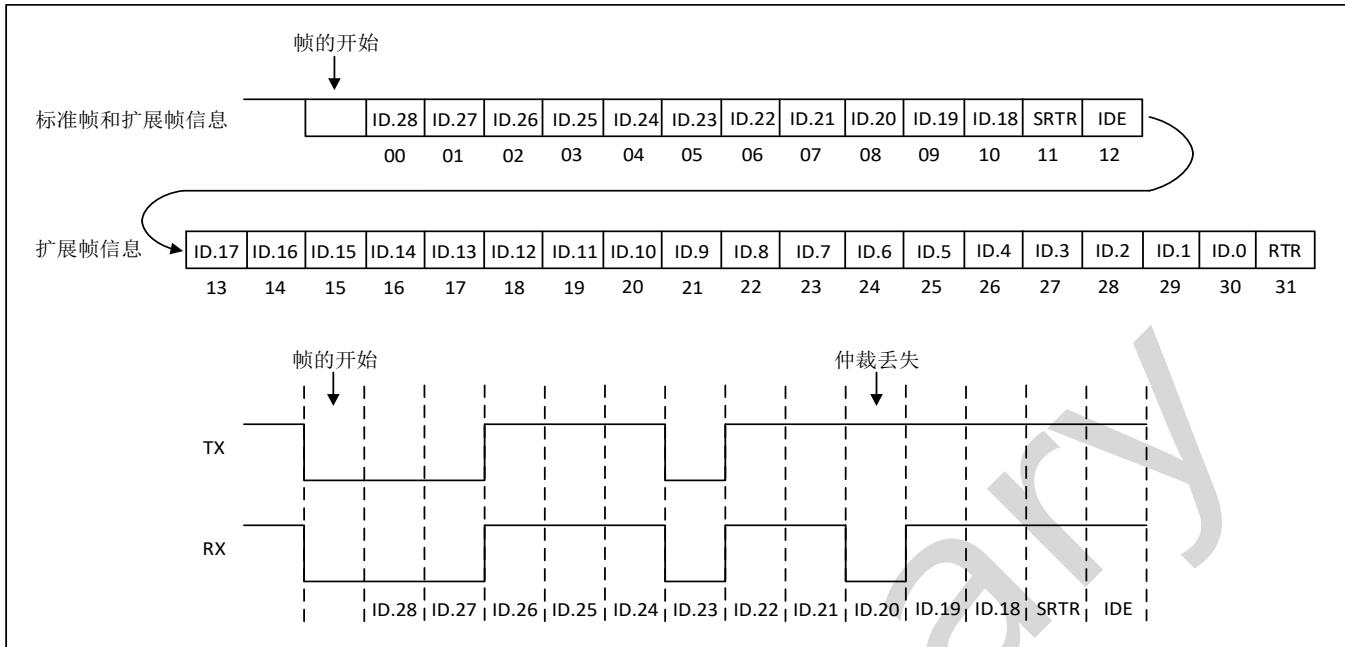


图 22-11 仲裁丢失解释举例

## 22.6 CAN 寄存器描述

表 22-5 CAN 寄存器概览

Offset	Acronym	Register Name	Reset	Section	Note
0x00	CAN_MOD	CAN 模式寄存器	0x00000001	小节 25.6.1	仅 PeliCAN 模式
0x00	CAN_CR	CAN 控制寄存器	0x00000021	小节 25.6.2	仅 BasicCAN 模式
0x04	CAN_CMRR	CAN 命令寄存器	0x000000XX	小节 25.6.3	复位值: BasicCAN 模式: 0x00FF PeliCAN 模式: 0x0000
0x08	CAN_SR	CAN 状态寄存器	0x0000000C	小节 25.6.4	
0x0C	CAN_IR	CAN 中断寄存器	0x000000XX	小节 25.6.5	复位值 BasicCAN 模式 0x00E0 PeliCAN 模式 0x0000
0x10	CAN_IER	CAN 中断使能寄存器	0x00000000	小节 25.6.6	仅存在 PeliCAN 模式
0x10	GROUP0_ACR	CAN 验收代码寄存器组 0	0x000000XX	小节 25.6.7	BasicCAN 模式
0x14	GROUP0_AMR	CAN 验收屏蔽寄存器组 0	0x000000XX	小节 25.6.8	BasicCAN 模式
0x18	CAN_BTR0	CAN 总线定时 0	0x00000000	小节 25.6.9	
0x1C	CAN_BTR1	CAN 总线定时 1	0x00000000	小节 25.6.10	

Offset	Acronym	Register Name	Reset	Section	Note
0x28	CAN_TXID0	CAN 发送识别码寄存器0	0x000000XX	小节25.6.11	仅存在 BasicCAN 模式, 复位模式为 0xFF
0x2C	CAN_TXID1	CAN 发送识别码寄存器1	0x000000XX	小节25.6.12	仅存在 BasicCAN 模式, 复位模式为 0xFF
0x2C	CAN_ALC	CAN 仲裁丢失捕捉寄存器	0x00000000	小节 25.6.13	仅存在 PeliCAN 模式
0x30	CAN_ECC	CAN 错误代码捕捉	0x00000000	小节 25.6.14	仅存在 PeliCAN 模式
0x34	CAN_EWLR	CAN 错误报警限制寄存器	0x00000060	小节 25.6.15	仅存在 PeliCAN 模式
0x38	CAN_RXERR	CAN RX 错误计数寄存器	0x00000000	小节 25.6.16	仅存在 PeliCAN 模式
0x3C	CAN_TXERR	CAN TX 错误计数寄存器	0x00000000	小节 25.6.17	仅存在 PeliCAN 模式
0x4C	CAN_TXDATA 0	CAN 发送数据寄存器 0	0x000000XX	小节 25.6.21	仅存在 PeliCAN 模式
0x50	CAN_TXDATA 1	CAN 发送数据寄存器 1	0x000000XX	小节 25.6.22	仅存在 PeliCAN 模式
0x7C	CAN_CDR	CAN 时钟分频寄存器	0x00000000	小节 25.6.23	
0x80	CAN_AFM0	CAN 滤波模式寄存器 0	0x00000000	小节 25.6.24	
0x84	CAN_AFM1	CAN 滤波模式寄存器 1	0x00000000	小节 25.6.25	
0x88	CAN_AFM2	CAN 滤波模式寄存器 2	0x00000000	小节 25.6.26	
0x98+ (x - 1) *0x20	GROUPx_ACR	CAN 验收代码寄存器组 x (x = 1~19)	0x000000XX	小节25.6.30	
0xA8+ (x - 1) *0x20	GROUPx_AMR	CAN 验收屏蔽寄存器组 x (x = 1~19)	0x000000XX	小节25.6.31	

## 22.6.1 CAN 模式寄存器 (CAN\_MOD)

仅 PeliCAN 模式

偏移地址: 0x00

复位值: 0x0000 0001

Bit	31	30	19	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Field	Res.				AFM	STM	LOM	RM
Type					rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:4	Reserved			保留, 始终读为 0。
3	AFM	rw	0x00	验收滤波器模式 (Acceptance filter mode) 1: 单; 选择单个验收滤波器 (32 位长度) 0: 双; 选择两个验收滤波器 (每个有 16 位激活)
2	STM	rw	0x00	自检测模式 (Self test mode) 1: 自检测; 此模式可以检测所有节点, 没有任何活动的节点使用自接收命令; 即使没有应答, CAN 控制器也会成功发送 0: 正常模式; 成功发送时必须应答信号
1	LOM	rw	0x00	只听模式 (Listen only mode) 1: 只听: 这种模式下即使成功接收信息, CAN 控制器也不向总线发应答信号; 错误计数器停止在当前值 0: 正常模式
0	RM	rw	0x01	复位模式 (Reset mode) 1: 复位; 检测到复位模式位被置位, 中止当前正在接收/发送的信息, 进入复位模式 0: 正常; 复位模式位接收到'1 - 0'的跳变后, CAN 控制器回到工作模式

## 22.6.2 CAN 控制寄存器 (CAN\_CR)

仅 BasicCAN 模式

偏移地址: 0x00

复位值: 0x0000 0021

控制寄存器的内容是用于改变 CAN 控制器的行为的。这些位可以被微控制器设置或置位，微控制器可以对控制寄存器进行读/写操作。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Res.																
Type																	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Res.												OIE	EIE	TIE	RIE	RR
Type													rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:5	Reserved			保留, 位 5 的值始终读为 1。
4	OIE	rw	0x00	溢出中断使能 (Overflow interrupt enable) 1: 使能; 如果置位数据溢出位, 微控制器接收溢出中断信号 0: 禁止; 微控制器不从 CAN 控制器接收溢出中断信号
3	EIE	rw	0x00	错误中断使能 (Error interrupt enable) 1: 使能; 如果出错或总线状态改变, 微控制器接收错误中断信号 0: 禁止; 微控制器不从 CAN 控制器接收错误中断信号
2	TIE	rw	0x00	发送中断使能 (Transmit interrupt enable) 1: 使能; 当信息被成功发送或发送缓冲器又被访问时 (例如, 中止发送命令后), 微控制器接收 CAN 控制器发出的一个发送中断信号 0: 禁止; 微控制器不从 CAN 控制器接收发送中断信号
1	RIE	rw	0x01	接收中断使能 (Receive interrupt enable) 1: 使能; 信息被无错误接收时, CAN 控制器发出的一个接收中断信号到微控制器 0: 禁止; 微控制器不从 CAN 控制器接收接收中断信号
0	RR	rw	0x01	复位请求 (Reset request) 1: 当前; CAN 控制器检测到复位请求后, 中止当前发送/接收的信息, 进入复位模式 0: 空缺; 复位请求位接收到一个下降沿后。CAN 控制器回到工作模式

### 22.6.3 CAN 命令寄存器 (CAN\_CMR)

偏移地址: 0x04

复位值: BasicCAN 模式: 0x0000 00FF

PeliCAN 模式: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.										ERB	SRR	CDO	RRB	AT	TR
Type											W	W	W	W	W	W

Bit	Field	Type	Reset	Description
31:6	Reserved			保留
5	ERB	w		<p>清空所有 RXFIFO 数据 (Empty rxbuffer)</p> <p>1 清除：清除 RXFIFO 所有内容 0：无动作</p> <p>清空 RXFIFO 数据这个命令位是用来清除数据溢出情况的。但不清除溢出状态标志位。如果数据溢出，且置位此位，接收缓存器中的内容将全部被释放。</p>
4	SRR	w		<p>PeliCAN 模式：</p> <p>SRR：自接收请求 (Self reset request) 1：当前信息可被同时发送和接收 0：(空缺)</p>
3	CDO	w		<p>清除数据溢出 (Clear data overrun) 1：清除；清除数据溢出状态位 0：无动作</p> <p>清除数据溢出这个命令位是用来清除由数据溢出状态位指出的数据溢出情况的。如果数据溢出位被置位，不会产生数据溢出中断。释放接收缓冲器命令的同时是可以发出清除数据溢出命令的。</p>
2	RRB	w		<p>释放接收缓冲器 (Release receive buffer) 1：释放；接收缓冲器中存放信息的内存空间将被释放 0：无动作</p> <p>读接收缓冲器之后，微控制器可以通过设置释放接收缓冲器位为 1 来释放 RXFIFO 中当前信息的内存空间。这可能会导致接收缓冲器中的另一条信息立即有效。这样会再产生一次接收中断（使能条件下）。如果没有其它可用信息，就不会再产生接收中断，接收缓冲器状态位被清除</p>
1	AT	w		<p>中止传输 (Abort transmission)</p> <p>1：当前；如果不是在处理过程中，等待处理的发送请求将取消 0：空缺；无动作</p> <p>中止传送位是在 CPU 要求当前传送暂停时使用的，例如，传送一条紧急信息。正在进行的传送是不停止的。要查看原始信息是否被成功发送，可以通过传送成功状态位来检测。不过，这必须在发送缓冲器状态位为 1（释放）或发送中断产生的情况下才能实现。</p>

Bit	Field	Type	Reset	Description
0	TR	w		<p>发送请求 (Transmission request) 1: 当前; 信息被发送 0: 空缺; 无动作</p> <p>如果发送请求在前面的命令中被置位。它就不可以通过直接设置为 0 来取消它了。不过, 可以通过设置中止发送位为 0 来取消。</p>

## 22.6.4 CAN 状态寄存器 (CAN\_SR)

偏移地址: 0x08

复位值: 0x0000 000C

Bi	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bi	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								BS	ES	TS	RS	TCS	TBS	DOS	RBS
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7	BS	rw	0x00	<p>BS: 总线状态 (Bus status) 1: 总线关闭; CAN 控制器退出总线活动 0: 总线开启; CAN 控制器加入总线活动</p>
6	ES	rw	0x00	<p>出错状态 (Error status)</p> <p>1: 出错; 至少出现一个错误计数器满或超过 CPU 报警限制</p> <p>0: ok; 两个错误计数器都在报警限制以下</p> <p>根据 CAN 2.0B 协议说明, 在接收或发送时检测到错误会影响错误计数。当至少有一个错误计数器满或超出 CPU 警告限制 (96) 时, 错误状态位被置位。在允许情况下, 会产生错误中断。</p>
5	TS	rw	0x00	<p>发送状态 (Transmit status)</p> <p>1: 发送; CAN 控制器在传送信息</p> <p>0: 空闲; 没有要发送的信息</p> <p>如果接收状态位和发送状态位都是 0, 则 CAN 总线是空闲的。</p>

Bit	Field	Type	Reset	Description
4	RS	rw	0x00	<p>接收状态 (Receive status)</p> <p>1: 接收; CAN 控制器正在接收信息 0: 空闲; 没有正在接收的信息</p> <p>如果接收状态位和发送状态位都是 0, 则 CAN 总线是空闲的</p>
3	TCS	rw	0x01	<p>发送完毕状态 (Transmission complete status)</p> <p>1: 完毕; 最近一次发送请求被成功处理 0: 未完毕; 当前发送请求未处理完毕</p> <p>无论何时发送请求位被置为'1', 发送完毕位都会被置为'0' (未完毕)。发送完毕位的'0'会一直保持到信息被成功发送。</p>
2	TBS	rw	0x01	<p>发送缓冲器状态 (Transmit buffer status)</p> <p>1: 释放; CPU 可以向发送缓冲器写信息 0: 锁定; CPU 不能访问发送缓冲器; 有信息正在等待发送或正在发送</p> <p>如果 CPU 在发送缓冲器状态位是 0 (锁定) 时试写发送缓冲器, 则写入的字节被拒绝接收且会在无任何提示的情况下丢失。</p>
1	DOS	rw	0x00	<p>数据溢出状态 (Data overrun status)</p> <p>1: 溢出; 信息丢失, 因为 RXFIFO 中没有足够的空间来存储它 0: 空缺; 自从最后一次清除数据溢出命令执行无数据溢出发生</p> <p>当要被接收的信息成功的通过验收滤波器后 (例如, 仲裁后之初), CAN 控制器需要在 RXFIFO 中用一些空间来存储这条信息的描述符。因此必须有足够的空间来存储接收的每一个数据字节。如果没有足够的空间存储信息, 信息将会丢失且只向 CPU 提示数据溢出情况。如果这个接收到的信息除了最后一位之外都无错误, 信息有效。</p>
0	RBS	rw	0x00	接收缓冲器状态 (Receive buffer status) 1: 满; RXFIFO 中有可用信息 0: 空; 无可 用信息在读 RXFIFO 中的信息且用释放接收缓冲器命令来释放内存空间之后, 这一位被 清除。如果 FIFO 中还有可用信息, 此位将在下一位的时限 (tSCL) 中被重新设置。

## 22.6.5 CAN 中断寄存器 (CAN\_IR)

偏移地址: 0x0C

复位值: BasicCAN 模式: 0x0000 00E0

PeliCAN 模式: 0x0000 0000

中断寄存器允许中断源的识别。当寄存器的一位或多们被置位时中断就被激活了。中断寄存器对微控制器来说是只读存储器。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								BEI	ALI	EPI	Res.	DOI	EI	TI	RI
Type									r	r	r		r	r	r	r

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7	BEI	r		<p>Basic 模式: 保留, 读出值为 1</p> <p>PeliCAN 模式: BEI: 总线错误中断 (Bus error interrupt) 1: 置位; 当 CAN 控制器检测到总线错误且中断使能寄存器中的 BEIE 被置位时此位被置位 0: 复位</p>
6	ALI	r		<p>Basic 模式: 保留, 读出值为 1</p> <p>PeliCAN 模式: ALI: 仲裁丢失中断 (Arbitration lost interrupt) 1: 置位; 当 CAN 控制器丢失仲裁, 变为接收器且中断使能寄存器的 ALIE 为被置位时, 此位被置位 0: 复位</p>
5	EPI	r		<p>Basic 模式: 保留, 读出值为 1</p> <p>PeliCAN 模式: EPI: 错误消极中断 (Error passive interrupt) 1: 置位; 当 CAN 控制器到达错误消极状态 (至少一个错误计数器超过协议规定的值 127) 或从错误消极状态又进入错误活动状态以及中断寄存器的 EPIE 位被置位时此位被置'1' 0: 复位</p>
4	Reserved			保留, 始终读为 0。

Bit	Field	Type	Reset	Description
3	DOI	r		<p>数据溢出中断 (Data overrun interrupt)            1: 设置; 当数据溢出中断使能位被置为'1'时向数据溢出状态位'0 - 1'跳变, 此位被置位            0: 复位; 微控制器的任何读访问将清除此位            溢出中断位 (中断允许情况下) 和溢出状态位是同时被置位的。</p>
2	EI	r		<p>错误中断 (Error interrupt)            1: 置位; 错误中断使能时, 错误状态位或总线状态位的变化会置位此位            0: 复位; 微控制器的任何读访问将清除此位</p>
1	TI	r		<p>发送中断 (Transmit interrupt)            1: 置位; 发送缓冲器状态从 0 变为 1 (释放) 和发送中断使能时, 置位此位            0: 复位; 微控制器的任何读访问将清除此位</p>
0	RI	r		<p>接收中断 (Receive interrupt)            1: 置位; 当接收 FIFO 不空和接收中断使能时置位此位            0: 复位; 微控制器的任何读访问将清除此位            接收中断位 (中断允许时) 和接收缓冲器状态位是同时置位的。            必须说明的是接收中断位在读的时候被清除, 即使 FIFO 中还有其它可用信息。当释放接收缓冲器命令执行后, 接收缓冲器中还有其它可用信息, 接收中断 (中断允许时) 会在下一个 tSCL 被重置。</p>

## 22.6.6 CAN 中断使能寄存器 (CAN\_IER)

仅存在 PeliCAN 模式

偏移地址: 0x10

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								BEIE	ALIE	EPIE	Res.	DOIE	EIE	TIE	RIE
Type									rw	rw	rw		rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7	BEIE	rw	0x00	总线错误中断使能 (Bus error interrupt enable) 1: 使能; 如果检测到总线错误, 则 CAN 控制器请求相应的中断 0: 禁止
6	ALIE	rw	0x00	仲裁丢失中断使能 (Arbitration lost interrupt enable) 1: 使能; 如果 CAN 控制器已丢失了仲裁, 则请求相应的中断 0: 禁止
5	EPIE	rw	0x00	错误消极中断使能 (Error passive interrupt enable) 1: 使能; 若 CAN 控制器的错误状态改变 (从消极到活动或反之), 则请求相应的中断 0: 禁止
4	Reserved			保留, 始终读为 0。
3	DOIE	rw	0x00	数据溢出中断使能 (Data overrun interrupt enable) 1: 使能; 如果数据溢出状态位被置位 (见状态寄存器), CAN 控制器请求相应的中断 0: 禁止
2	EIE	rw	0x00	错误报警中断使能 (Error interrupt enable) 1: 使能; 如果错误或总线状态改变 (见状态寄存器), CAN 控制器请求相应的中断 0: 禁止
1	TIE	rw	0x00	发送中断使能 (Transmit interrupt enable) 1: 使能; 当信息被成功发送或发送缓冲器又可访问 (例如, 中止发送命令后) 时, CAN 控制器请求相应的中断 0: 禁止
0	RIE	rw	0x00	接收中断使能 (Receive interrupt enable) 1: 使能; 当接收缓冲器状态是'满'时, CAN 控制器请求相应的中断 0: 禁止

## 22.6.7 CAN 验收代码寄存器组 0(GROUP0\_ACR)

BasicCAN 模式: GROUP0\_ACR

偏移地址: 0x10

复位值: 0x0000 00XX

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								AC							
Type									rw							

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:0	AC	rw	0xXX	(Acceptance code) 复位请求位被置高 (当前) 时, 这个寄存器是可以访问 (读/写) 的。如果一条信息通过了验收滤波器的测试而且接收缓冲器有空间, 那么描述符和数据将被分别顺次写入 RXFIFO。当信息被正确的接收完毕就会: 接收状态位置高 (满) 接收中断使能位置高 (使能) 接收中断置高 (产生中断) 验收代码位 (AC.7 - AC.0) 和信息识别码的高 8 位 (ID.10 - ID.3) 相等, 且与验收屏蔽位 (AM.7 - AM.0) 的相应位相或为 1。即如果满足以下方程的描述, 则被接收: [(ID.10 - ID.3) $\equiv$ (AC.7 - AC.0)] $\vee$ (AM.7 - AM.0) $\equiv$ 11111111

PeliCAN 模式: 第 0 组有四个验收代码寄存器分别是 GROUP0\_ACR0, GROUP0\_ACR1, GROUP0\_ACR2, GROUP0\_ACR3

GROUP0\_ACR0: 偏移地址: 0x40 复位值: 0x0000 00XX

GROUP0\_ACR1: 偏移地址: 0x44 复位值: 0x0000 00XX

GROUP0\_ACR2: 偏移地址: 0x48 复位值: 0x0000 00XX

GROUP0\_ACR3: 偏移地址: 0x4C 复位值: 0x0000 00XX

注: 详细说明见的标识符过滤中 peliCAN 模式介绍。这是第 0 组验收代码寄存器其他组见小节 25.6.30。

## 22.6.8 CAN 验收屏蔽寄存器组 0(GROUP0\_AMR)

BasicCAN 模式: GROUP0\_AMR

偏移地址: 0x14

复位值: 0x0000 00XX

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field																
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								AM							
Type									rw							

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:0	AM	rw	0xXX	(Acceptance mask) 如果复位请求位置高 (当前) 这个寄存器可以被访问 (读/写)。验收屏蔽寄存器定义验收代码寄存器的相应位对验收滤波器是“相关的”或“无影响的”(即可为任意值)。

PeliCAN 模式: 第 0 组有四个验收屏蔽寄存器分别是 GROUP0\_AMR0 , GROUP0\_AMR1 , GROUP0\_AMR2, GROUP0\_AMR3

GROUP0\_AMR0: 偏移地址: 0x50

复位值 : 0x0000 00XX

GROUP0\_AMR1: 偏移地址: 0x54

复位值 : 0x0000 00XX GROUP0\_AMR2: 偏移地址: 0x58

复位值 : 0x0000 00XX

GROUP0\_AMR3: 偏移地址: 0x5C

复位值: 0x0000 00XX

注: 详细说明见的标识符过滤中 peliCAN 模式介绍。这是第 0 组验收屏蔽寄存器'其他组见小节 25.6.31。

## 22.6.9 CAN 总线定时 0(CAN\_BTR0)

偏移地址: 0x18

复位值: 0x0000 0000

总线定时寄存器 0 定义了波特率预设值 (BRP) 和同步跳转宽度 (SJW) 的值。复位模式有效时这个寄存器是可以被访问 (读/写) 的。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								SJW		BRP					
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:6	SJW	rw		<p>同步跳转宽度 (Synchronization Jump width)</p> <p>为了补偿在不同总线控制器的时钟振荡器之间的相位偏移, 任何总线控制器必须在当前传送的相关信号边沿重新同步。同步跳转宽度定义了每一位周期可以被重新同步缩短或延长的时钟周期的最大数目。</p> $t_{SJW} = t_{SCL} \times (S\ JW + 1)$
5:0	BRP	rw	0x00	<p>波特率预设值 (Baud rate prescaler)</p> <p>CAN 系统时钟 <math>t_{SCL}</math> 的周期是可编程的而且决定了相应的位时序。CAN 系统时钟由如下公式计算</p> $t_{SCL} = 2xtCLKx(BRP + 1)$ <p>这里 <math>t_{CLK} = APB1</math> 的时钟周期</p>

## 22.6.10 CAN 总线定时 1 (CAN\_BTR1)

偏移地址: 0x1C

复位值: 0x0000 0000

总线定时寄存器 1 定义了每个位周期的长度、采样点的位置和在每个采样点的采样数目。在复位模式中, 这个寄存器可以被读/写访问。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Res.																
Type																	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Res.										SAM	TESG2		TESG1			
Type											rw	rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7	SAM	rw	0x00	<p>采样 (Sampling)</p> <p>1: 三倍; 总线采样三次; 建议在低/中速总线 (A 和 B 级) 上使用, 这对过滤总线上的毛刺波是有益的</p> <p>0: 单倍; 总线采样一次; 建议使用在高速总线上 (SAE C 级)</p>

Bit	Field	Type	Reset	Description
6:0	TSEG2 TSEG1	rw	0x00	<p>第一时间段 (Time segment 1) 和第二时间段 (Time segment 2)            TSEG1 和 TSEG2 决定了每一位的时钟数目和采样点的位置，这里：  <math>t_{SYNCSEG} = 1 \times t_{SCL}</math>  <math>t_{TSEG1} = t_{SCL} \times (8 \times TSEG1.3 + 4 \times TSEG1.2 + 2 \times TSEG1.1 + TSEG1.0 + 1)</math>  <math>t_{TSEG2} = t_{SCL} \times (4 \times TSEG2.2 + 2 \times TSEG2.1 + TSEG2.1 + 1)</math></p>

## 22.6.11 CAN 发送识别码寄存器 0(CAN\_TXID0)

在 BasicCAN 模式： 偏移地址： 0x28

复位值： 0x0000 00XX

注： 复位模式为 0xFF

发送识别码寄存器 0 定义发送帧的类型与数据长度。仅在工作模式下这个寄存器可以被读/写访问。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留，始终读为 0。
7:0	IDx	rw	0XX	CAN 识别码 10 ~ 3 (CAN identifier byte 10 ~3) 注： 在 Peli 模式下，位 [3: 0] 为 DLC[3: 0]

## 22.6.12 CAN 发送识别码寄存器 1 (CAN\_TXID1)

仅存在 BasicCAN 模式：

偏移地址： 0x2C

复位值： 0x0000 00XX

注： 复位模式为 0xFF

发送识别码寄存器 1 定义发送帧的类型与数据长度。仅在工作模式下这个寄存器可以被读/写访问。

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

Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.							ID2	ID1	ID0	RTR	DLC3	DLC2	DLC1	DLC0	
Type								rw	rw	rw	rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:5	IDx	rw	0x0X	CAN 识别码 2 ~ 0 ( CAN identifier byte 2 ~ 0)
4	RTR	rw	0x0X	帧格式 ( Remote transmission request)
3:0	DLCx	rw	0x0X	发送数据区长度 0 ~ 8 ( Data length code 0 ~ 8)

仅存在 BasicCAN 模式:

偏移地址: 0x30 ~ 0x4C

复位值: 0x0000 0000

发送数据寄存器 CAN\_TXDR0 ~ 7。仅在工作模式下这个寄存器可以被读/写访问。接收缓冲与发送缓冲数据格式一致。

### 22.6.13 CAN 仲裁丢失捕捉寄存器 (CAN\_ALC)

仅存在 PeliCAN 模式:

偏移地址: 0x2C

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.							BITNO								
Type								rw	rw	rw	rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
31: 5	Reserved			保留, 始终读为 0。

4:0	BITNO	rw	0x00	值和功能参考下表 (Bit number) 仲裁丢失时，会产生相应的仲裁丢失中断（中断允许）。同时，位流处理器的当前位位置被捕捉送入仲裁丢失捕捉寄存器。一直到用户通过软件读这个值，寄存器中的内容都不会变。随后，捕捉机制又被激活了读中断寄存器时，中断寄存器中相应的中断标志位被清除。直到仲裁丢失 捕捉寄存器被读一次之后，新的仲裁丢失中断才有效。
-----	-------	----	------	---

位					十进制值	功能
ALC.4	ALC.3	ALC.2	ALC.1	ALC.0		
0	0	0	0	0	0	仲裁丢失在识别码的 bit1
0	0	0	0	1	1	仲裁丢失在识别码的 bit2
0	0	0	1	0	2	仲裁丢失在识别码的 bit3
0	0	0	1	1	3	仲裁丢失在识别码的 bit4
0	0	1	0	0	4	仲裁丢失在识别码的 bit5
0	0	1	0	1	5	仲裁丢失在识别码的 bit6
0	0	1	1	0	6	仲裁丢失在识别码的 bit7
0	0	1	1	1	7	仲裁丢失在识别码的 bit8
0	1	0	0	0	8	仲裁丢失在识别码的 bit9
0	1	0	0	1	9	仲裁丢失在识别码的 bit10
0	1	0	1	0	10	仲裁丢失在识别码的 bit11
0	1	0	1	1	11	仲裁丢失在 SRTR 位；注 2
0	1	1	0	0	12	仲裁丢失在 IDE 位
0	1	1	0	1	13	仲裁丢失在识别码的 bit12；注 3
0	1	1	1	0	14	仲裁丢失在识别码的 bit13；注 3
0	1	1	1	1	15	仲裁丢失在识别码的 bit14；注 3
1	0	0	0	0	16	仲裁丢失在识别码的 bit15；注 3

位					十进制值	功能
ALC.4	ALC.3	ALC.2	ALC.1	ALC.0		
1	0	0	0	1	17	仲裁丢失在识别码的 bit16; 注 3
1	0	0	1	0	18	仲裁丢失在识别码的 bit17; 注 3
1	0	0	1	1	19	仲裁丢失在识别码的 bit18; 注 3
1	0	1	0	0	20	仲裁丢失在识别码的 bit19; 注 3
1	0	1	0	1	21	仲裁丢失在识别码的 bit20; 注 3
1	0	1	1	0	22	仲裁丢失在识别码的 bit21; 注 3
1	0	1	1	1	23	仲裁丢失在识别码的 bit22; 注 3
1	1	0	0	0	24	仲裁丢失在识别码的 bit23; 注 3
1	1	0	0	1	25	仲裁丢失在识别码的 bit24; 注 3
1	1	0	1	0	26	仲裁丢失在识别码的 bit25; 注 3
1	1	0	1	1	27	仲裁丢失在识别码的 bit26; 注 3
1	1	1	0	0	28	仲裁丢失在识别码的 bit27; 注 3
1	1	1	0	1	29	仲裁丢失在识别码的 bit28; 注 3
1	1	1	1	0	30	仲裁丢失在识别码的 bit29; 注 3
1	1	1	1	1	31	仲裁丢失在 RTR 位; 注 3

注：仲裁丢失的二进制编码结构位的号码。标准帧信息的 RTR 位。只使用于扩展帧信息。

## 22.6.14 CAN 错误代码捕捉寄存器 (CAN\_ECC)

仅存在 PeliCAN 模式：

偏移地址: 0x30

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								ERRC		DIR	SEG				
Type									r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:6	ERRC	r	0x00	错误代码 (Error code) 00: 位错 01: 格式错 10: 填充错 11: 其它错误
5	DIR	r	0x00	方向 (Direction) 1: RX; 接收时发生的错误 0: TX; 发送时发生的错误

Bit	Field	Type	Reset	Description
4:0	SEG	r	0x00	段 (Segment) 00010: ID.28 - ID.21 00011: 帧开始 00100: SRTR 位 00101: IDE 位 00110: ID.20 - ID.18 00111: ID.17 - ID.13 01000: CRC 序列 01001: 保留位 0 01010: 数据区 01011: 数据长度代码 01100: RTR 位 01101: 保留位 1 01110: ID.4 - ID.0 01111: ID.12 - ID.5 10001: 活动错误标志 10010: 中止 10011: 支配 (控制) 位误差 10110: 消极错误标志 10111: 错误定义符 11000: CRC 定义符 11001: 应答通道 11010: 帧结束 11011: 应答定义符 11100: 溢出标志 其他: 保留

注：位的设置反映了当前结构段的不同错误事件。

总线发生错误时被迫产生相应的错误中断（中断允许时）。同时，位流处理器的当前位置被捕捉送入错误代码捕捉寄存器。其内容直到用户通过软件读出时都是不变的。读出后，捕捉机制又被激活了。访问中断寄存器期间，中断寄存器中相应的中断标志位被清除。新的总线中断直到捕捉寄存器被读出一次才可能有效。

## 22.6.15 CAN 错误报警限制寄存器 (CAN\_EWLR)

仅存在 PeliCAN 模式：偏移地址：0x34

复位值: 0x0000 0060

错误报警限制在这个寄存器中被定义。默认值（复位值）是 0060。复位模式中，此寄存器对 CPU 来说是可读/写的。工作模式中是只读的。

注：只有之前进入复位模式'EWLR 才有可能被改变。直到复位模式被再次取消后'才有可能发生错误状态的改变（见状态寄存器） 和由新的寄存器内容引起的错误报警中断。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								EWL							
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:0	EWL	rw	0x60	可编程的错误限制报警 (Programmable error warning limit) 当只要一个错误计数器超过错误限制编程值, 错误状态位被置位。

## 22.6.16 CAN RX 错误计数寄存器 (CAN\_RXERR)

仅存在 PeliCAN 模式:

偏移地址: 0x38

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								RXERR							
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。

Bit	Field	Type	Reset	Description
7:0	RXERR	rw	0x00	<p>RX 错误计数寄存器 (RX error counter register) 反应了接收错误计数器的当前值。硬件复位后寄存器被初始化为 0。工作模式中, 对 CPU 来说是只读的。只有在复位模式中才可以写访问此寄存器。如果发生总线关闭, RX 错误计数器就被初始化为 0。总线关闭期间, 写这个寄存器是无效的。</p> <p>注意: 只有之前进入复位模式, 才有可能由 CPU 迫使 RX 错误计数器发生改变。直到复位模式被取消后, 错误状态的改变(见状态寄存器)、错误报警和由新的寄存器内容引起的错误中断才可能有效。</p>

### 22.6.17 CAN TX 错误计数寄存器 (CAN\_TXERR)

仅存在 PeliCAN 模式:

偏移地址: 0x3C

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								TXERR							
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:0	TXERR	rw	0x00	<p>TX 错误计数寄存器 (TX error counter register) 反映了发送错误计数器的当前值。</p> <p>工作模式中, 这个寄存器对 CPU 是只读内存。复位模式中才可以写访问这个寄存器。硬件复位后, 寄存器被初始化为 0。如果总线关闭, TX 错误计数器被初始化为 127 来计算总线定义的最长时间 (128 个总线空闲信号)。这段时间里读 TX 错误计数器将反映出总线关闭恢复的状态信息。如果总线关闭是激活的, 写访问 TXERR 的 0 ~ 254 单元会清除总线关闭标志, 复位模式被清除后控制器会等待一个 11 位的续隐藏 (弱势) 位 (总线空闲)。</p> <p>向 TXERR 写入 255 会初始化 CPU 驱动的总线关闭事件。只有之前进入复位模式, 才有可能发生 CPU 引起的 TX 错误计数器内容的改变。直到复位模式被再次取消, 错误或总线状态的改变 (见状态寄存器)、错误报警和由新的寄存器内容引起的错误中断才有可能有效。离开复位模式后, 就象总线错误引起的一样, 给出新的 TX 计数器内容且总线关闭被同样的执行。这意味着重新进</p>

Bit	Field	Type	Reset	Description
				入复位模式, TX 错误计数器被初始化到 127, RX 计数器被清 0, 所有的相关状态和中断寄存器位被置位。复位模式的清除将会执行协议规定的总线关闭恢复序列 (等待 128 个总线空闲信号)。如果在总线关闭恢复 (TXERR > 0) 之前又进入复位模式, 如果在总线关闭恢复 (TXERR > 0) 之前又进入复位模式, 总线关闭保持有效且 TXERR 被锁定。
6	RTR	rw	0x0X	帧格式 (Remote transmission request) 1: 远程; CAN 将发送/接收远程帧 0: 数据; CAN 将发送/接收数据帧
3:0	DLC	rw	0x0X	数据区长度 (Data length code bit) 发送数据区长度 0 ~ 8。

### 22.6.18 CAN 发送帧信息寄存器 (CAN\_SFF)

仅存在 PeliCAN 模式: 偏移地址: 0x40

复位值: 0x0000 00XX

发送帧信息寄存器设置送帧的类型与数据长度。仅在工作模式下这个寄存器可以被读/写访问, 写时为发送寄存器, 读时为接收寄存器, 数据结构相同。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								FF	RTR	Res.		DLC.3	DLC.2	DLC.1	DLC.0
Type									rw	rw		rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7	FF	rw	0x0X	帧格式 (Frame format) 1: EFF; CAN 将发送/接收扩展帧格式 0: SFF; CAN 将发送/接收标准帧格式
6	RTR	rw	0x0X	帧格式 (Remote transmission request) 1: 远程; CAN 将发送/接收远程帧 0: 数据; CAN 将发送/接收数据帧
5:4	Reserved			保留。
3:0	DLC	rw	0x0X	数据区长度 (Data length code bit) 发送数据区长度 0 ~ 8。

### 22.6.19 CAN 发送识别码寄存器 0(CAN\_TXID0)

仅存在 PeliCAN 模式:

偏移地址: 0x44

复位值: 0x0000 00XX

发送识别码寄存器 0 设置帧的标识符。仅在工作模式下这个寄存器可以被读/写访问, 写时为发送寄存器, 读时为接收寄存器, 数据结构相同。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
Type									rw							

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。

7:0	IDx	rw	0xXX	CAN 标识符 ID28 ~ ID21 (Identifier bit 28 - 21) 标准帧时, 与 ID20 ~ ID18 组成 11 位标识符。
-----	-----	----	------	--

## 22.6.20 CAN 发送识别码寄存器 1(CAN\_TXID1)

仅存在 PeliCAN 模式: 偏移地址: 0x48

复位值: 0x0000 00XX

发送识别码寄存器 1 设置帧的标识符。仅在工作模式下这个寄存器可以被读/写访问, 写时为发送寄存器, 读时为接收寄存器, 数据结构相同。

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

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:5	IDx	rw	0x0X	CAN 标识符 ID20 ~ ID18 (Identifier bit 20 - 18)
4:0	IDx	rw	0x0X	标准帧 无意义 扩展帧 CAN 标识符 ID17 ~ ID13 (Identifier bit 17 - 13)

## 22.6.21 CAN 发送数据寄存器 0(CAN\_TXDATA0)

仅存在 PeliCAN 模式: 偏移地址: 0x4C

复位值: 0x0000 00XX

发送数据寄存器 0, 用于 CAN 数据发送存储。仅在工作模式下这个寄存器可以被读/写访问, 写时为发送寄存器, 读时为接收寄存器, 数据结构相同。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								DATA0							
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:0	DATA0	rw	0xXX	CAN 发送数据 0 (CAN transmit data 0) 扩展帧 ID12 ~ 5

## 22.6.22 CAN 发送数据寄存器 1(CAN\_TXDATA1)

仅存在 PeliCAN 模式：偏移地址：0x50

复位值：0x0000 00XX

发送数据寄存器 1，用于 CAN 数据发送存储。仅在工作模式下这个寄存器可以被读/写访问，写时为发送寄存器，读时为接收寄存器，数据结构相同。

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								DATA1							
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:0	DATA1	rw	0xXX	CAN 发送数据 1 (CAN transmit data 1) 1) 扩展帧 ID4 ~ 0, 低 3 位无意义

仅存在 PeliCAN 模式：

偏移地址: 0x54 ~ 0x70

复位值: 0x0000 00XX

以上偏移地址均为 CAN 数据寄存器，扩展帧时 CAN 发送数据 0 储存在 DATA2 余下数据依次类推。这些寄存器写时为发送寄存器，读时为接收寄存器，数据结构相同。

### 22.6.23 CAN 时钟分频寄存器 (CAN\_CDR)

偏移地址: 0x7C

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Res.																
Type																	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Res.								MODE	Res.							
Type									rw								

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7	MODE	rw	0x00	CAN 模式 ( CAN mode )如果 MODE 是 0, CAN 控制器工作于 BasicCAN 模式。否则, CAN 控制器工作于 PeliCAN 模式。只有在复位模式中是可以写的。
6:0	Reserved			保留, 始终读为 0。

### 22.6.24 CAN 滤波模式寄存器 0(CAN\_AFM0)

偏移地址: 0x80

复位值: 0x0000 0000

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

Field	Res.	AFM7	AFM6	AFM5	AFM4	AFM3	AFM2	AFM1	Res.
Type		rw							

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:1	AFMx	rw	0x00	验收滤波器模式 ( Acceptance filter mode), x = 1~7 1: 单; 选择单个验收滤波器 (32 位长度) 0: 双; 选择两个验收滤波器 (每个有 16 位激活)
0	Reserved			保留, 始终读为 0。

## 22.6.25 CAN 滤波模式寄存器 1(CAN\_AFM1)

偏移地址: 0x84

复位值: 0x0000 0000

Bit	3 1	3 0	2 9	2 8	2 7	2 6	2 5	2 4	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	1 5	1 4	1 3	1 2	11	1 0	9	8	7	6	5	4	3	2	1	0
Field	Res.						AFM1 5	AFM1 4	AFM1 3	AFM1 2	AFM1 1	AFM1 0	AFM 9	AFM 8		
Type							rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。

7:0	AFMx	rw	0x00	验收滤波器模式 (Acceptance filter mode), x= 8~15 1: 单; 选择单个验收滤波器 (32 位长度) 0: 双; 选择两个验收滤波器 (每个有 16 位激活)
-----	------	----	------	--

## 22.6.26 CAN 滤波模式寄存器 2(CAN\_AFM2)

偏移地址: 0x88

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.												AFM19	AFM18	AFM17	AFM16
Type													rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:4	Reserved			保留, 始终读为 0。
3:0	AFMx	rw	0x00	验收滤波器模式 (Acceptance filter mode), x= 16~19 1: 单; 选择单个验收滤波器 (32 位长度) 0: 双; 选择两个验收滤波器 (每个有 16 位激活)

## 22.6.27 CAN 滤波组使能寄存器 0(CAN\_FGA0)

偏移地址: 0x8C

复位值: 0x0000 0001

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	FGA7								FGA6	FGA5	FGA4	FGA3	FGA2	FGA1	Res.	
Type	rw								rw	rw						

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:1	FGAx	rw	0x00	Group 滤波打 (x=1~7) 0 无效 1 有效 Group 滤波
0	Reserved			保留, 始终读为 1。

## 22.6.28 CAN 滤波组使能寄存器 1(CAN\_FGA1)

偏移地址: 0x90

复位值: 0x0000 0000

Bit	3 1	3 0	2 9	2 8	2 7	2 6	2 5	2 4	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	1 5	1 4	1 3	1 2	11	1 0	9	8	7	6	5	4	3	2	1	0
Field	FGA1 5								FGA1 4	FGA1 3	FGA1 2	FGA1 1	FGA1 0	FGA 9	FGA 8	
Type	rw								rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:0	FGAx	rw	0x00	Group 滤波打开 (x=8~15) 0 无效 1 有效 Group 滤波

## 22.6.29 CAN 滤波组使能寄存器 2(CAN\_FGA2)

偏移地址: 0x94

复位值: 0x0000 0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Type	Res.												FGA19	FGA18	FGA17	FGA16
Bit													rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:0	FGAx	rw	0x00	Group 滤波打开 (x=16~19) 0 无效 1 有效 Group 滤波

## 22.6.30 CAN 验收代码寄存器组 x(x = 1 ..19) (GROUPx\_ACR)

BasicCAN 模式: GROUPx\_ACR

偏移地址: 0x098 + (x - 1) \* 0x20

复位值: 0x0000 00XX

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Type	Res.								AC							
Bit									rw							

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:0	AC	rw	0xXX	(Acceptance code) 复位请求位被置高 (当前) 时, 这个寄存器是

Bit	Field	Type	Reset	Description
				<p>可以访问（读/写）的。如果一条信息通过了验收滤波器的测试而且接收缓冲器有空间，那么描述符和数据将被分别顺次写入RXFIFO。当信息被正确的接收完毕就会：接收状态位置高（满）接收中断使能位置高（使能）接收中断置高（产生中断）验收代码位（AC.7 - AC.0）和信息识别码的高8位（ID.10 - ID.3）相等，且与验收屏蔽位（AM.7 - AM.0）的相应位相或为1。即如果满足以下方程的描述，则被接收：</p> $[(ID.10 - ID.3) \equiv (AC.7 - AC.0)] \vee (AM.7 - AM.0) \equiv 11111111$

PeliCAN 模式：每一组有四个验收代码寄存器分别是 GROUPx\_ACR0, GROUPx\_ACR1, GROUPx\_ACR2, GROUPx\_ACR3

GROUPx\_ACR0: 偏移地址: 0x098+ (x - 1) \* 0x20 复位值: 0x0000 00XX  
 GROUPx\_ACR1: 偏移地址: 0x09C + (x - 1) \* 0x20 复位值: 0x0000 00XX  
 GROUPx\_ACR2: 偏移地址: 0x0A0 + (x - 1) \* 0x20 复位值: 0x0000 00XX  
 GROUPx\_ACR3: 偏移地址: 0x0A4 + (x - 1) \* 0x20 复位值: 0x0000 00XX  
 注：第0组验收代码寄存器见小节25.6.7。

### 22.6.31 CAN 验收屏蔽寄存器组 x(x = 1 .. 19) (GROUPx\_AMR)

BasicCAN 模式: GROUPx\_AMR

偏移地址: 0x0A8 + (x - 1) \* 0x20

复位值: 0x0000 00XX

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								AM							
Type									rw							

Bit	Field	Type	Reset	Description
31:8	Reserved			保留, 始终读为 0。
7:0	AM	rw	0xXX	(Acceptance mask ) 如果复位请求位置高 (当前) 这个寄存器 可以被访问 (读/写)。 验收屏蔽寄存器定义 验收代码寄存器的相 应位对验收滤波器是 “相关的”或“无影响的 ” (即可为任意值)。

PeliCAN 模式：每一组有四个验收屏蔽寄存器分别是 GROUPx\_AMR0 , GROUPx\_AMR1 , GROUPx\_AMR2, GROUPx\_AMR3

GROUPx\_AMR0: 偏移地址: 0x0A8 + (x - 1) \* 0x20

复位值 : 0x0000 00XX

GROUPx\_AMR1: 偏移地址: 0x0AC+ (x- 1) \* 0x20

复位值 : 0x0000 00XX GROUPx\_AMR2:

偏移地址: 0x0B0 + (x - 1) \* 0x20

复位值 : 0x0000 00XX GROUPx\_AMR3:

偏移地址: 0x0B4 + (x - 1) \* 0x20

复位值: 0x0000 00XX

注: 第 0 组验收屏蔽寄存器见小节 25.6.8

## 23 USB On-The-Go Full-Speed (OTG-FS)

OTG-FS 控制器支持配置为 OTG 双角色设备（dual-role device），也可仅配置为 USB device 设备或 USB\_HOST 嵌入式主机。OTG-FS 控制器完全符合《On-The-Go Supplement to the USB 2.0 Specification》。

表格 23-1 术语表

英文缩写	说明
FS	全速
LS	低速
USB	通用串行总线
OTG	On-The-Go
PHY	物理层
HNP	主机协商协议
SRP	会话请求协议
BDT	缓冲区描述表
BD	缓冲区描述符
SIE	串行接口号擎

### 23.1 OTG-FS 控制器主要功能特征

#### 23.1.1 USB 功能特征

- 1) 全速设备控制器兼容 USB 1.1 和 2.0 规范
- 2) 16 个双向端点 (End Point)
- 3) FIFO 数据流接口
- 4) DMA 数据传输，无需 CPU 参与完成 USB FIFO 和系统存储器的之间的数据交互

#### 23.1.2 额外 OTG 功能特征

- 1) 支持 USB\_HOST 嵌入式主机模式
- 2) 支持 dual-role device OTG 双角色设备模式
- 3) 通过 ID 线辨认 A-B 设备
- 4) 支持主机协商协议 (HNP) 及会话请求协议 (SRP)
- 5) 在 OTG 应用中，允许主机关闭VBUS 以节省耗电

## 23.2 OTG-FS 系统框图

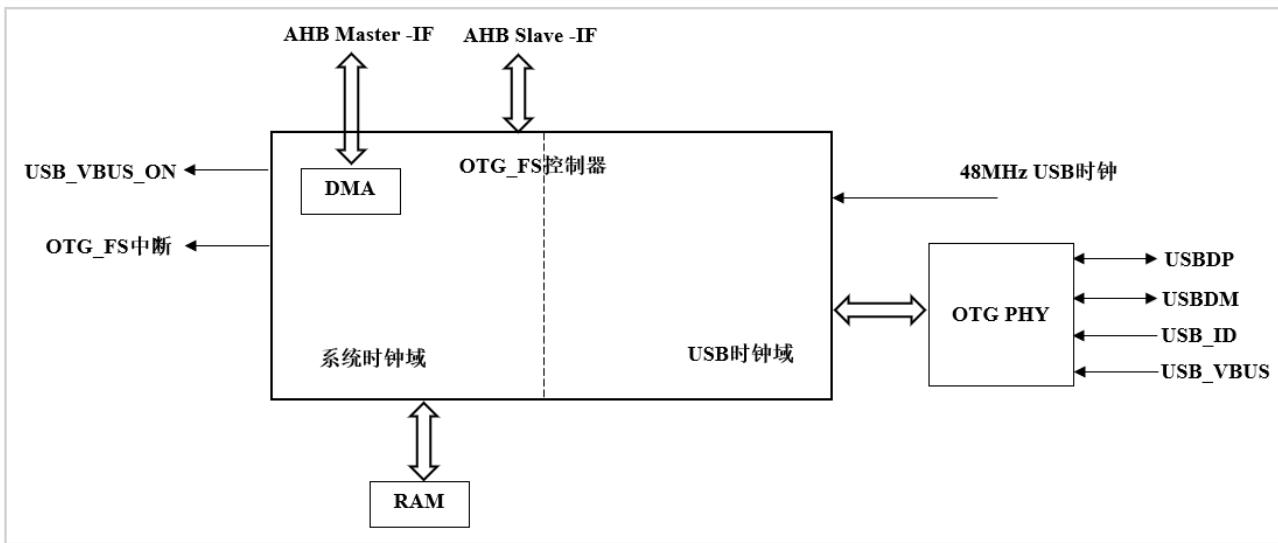


图 23-1 OTG-FS 功能框图

## 23.3 OTG-FS 引脚说明

表格 23-2 BDT USB 引脚说明

引脚名	方向	描述
USBDP	输入输出	差分数据 D+ 信号
USBDM	输入输出	差分数据 D- 信号
USB_ID	输入	USB A-B 器件识别信号
USB_VBUS	输入	USB VBUS 信号
USB_VBUS_ON	输出	外部电源芯片使能信号

## 23.4 OTG-FS 硬件外部配置

下图分别显示仅主机、仅从设备以及双角色连接的概览。

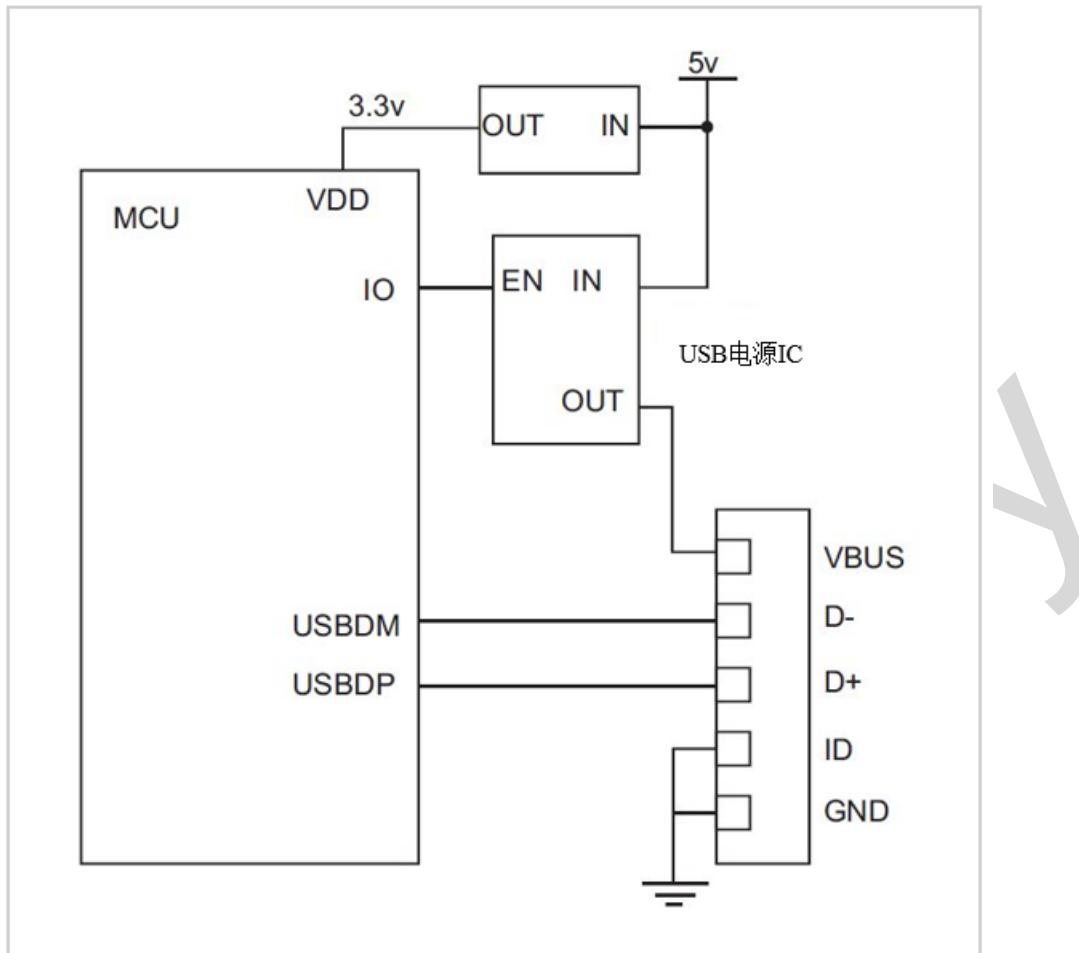


图 23-2 仅主机框图

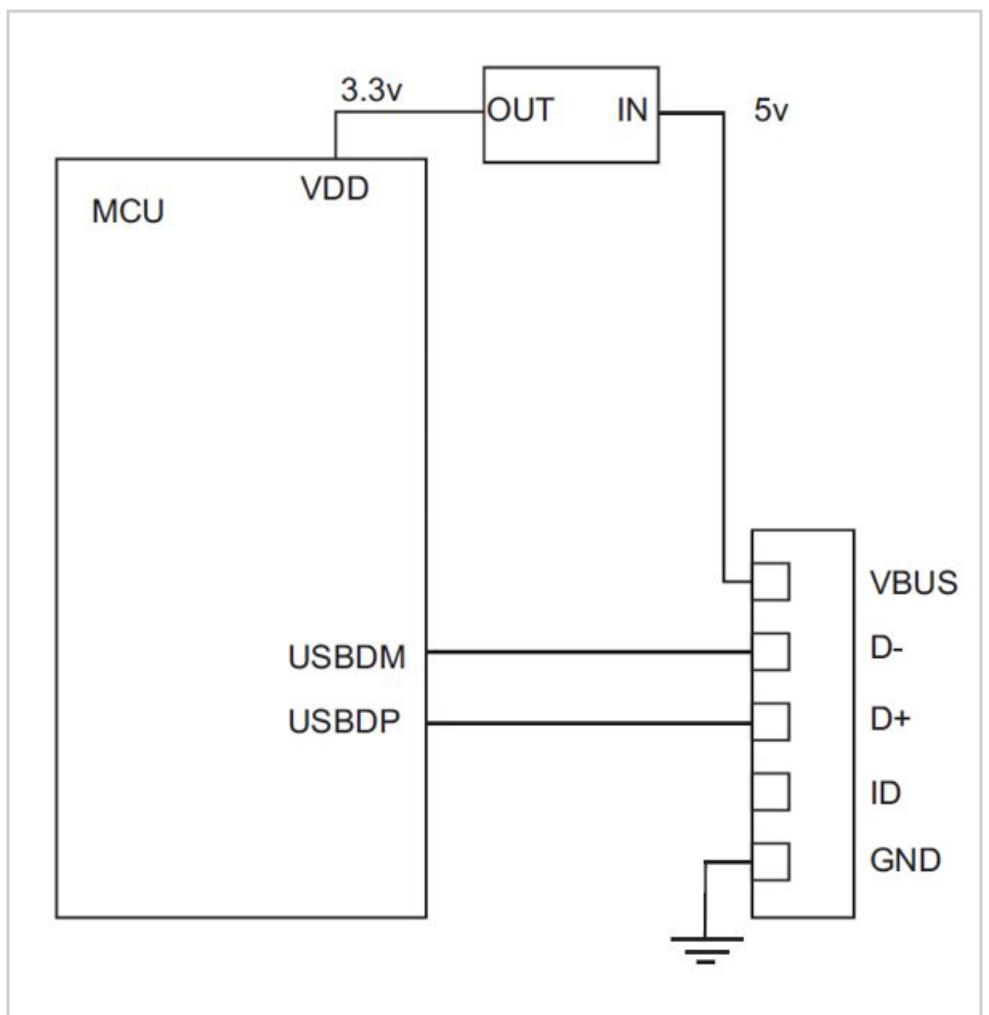


图 23-3 仅从设备框图

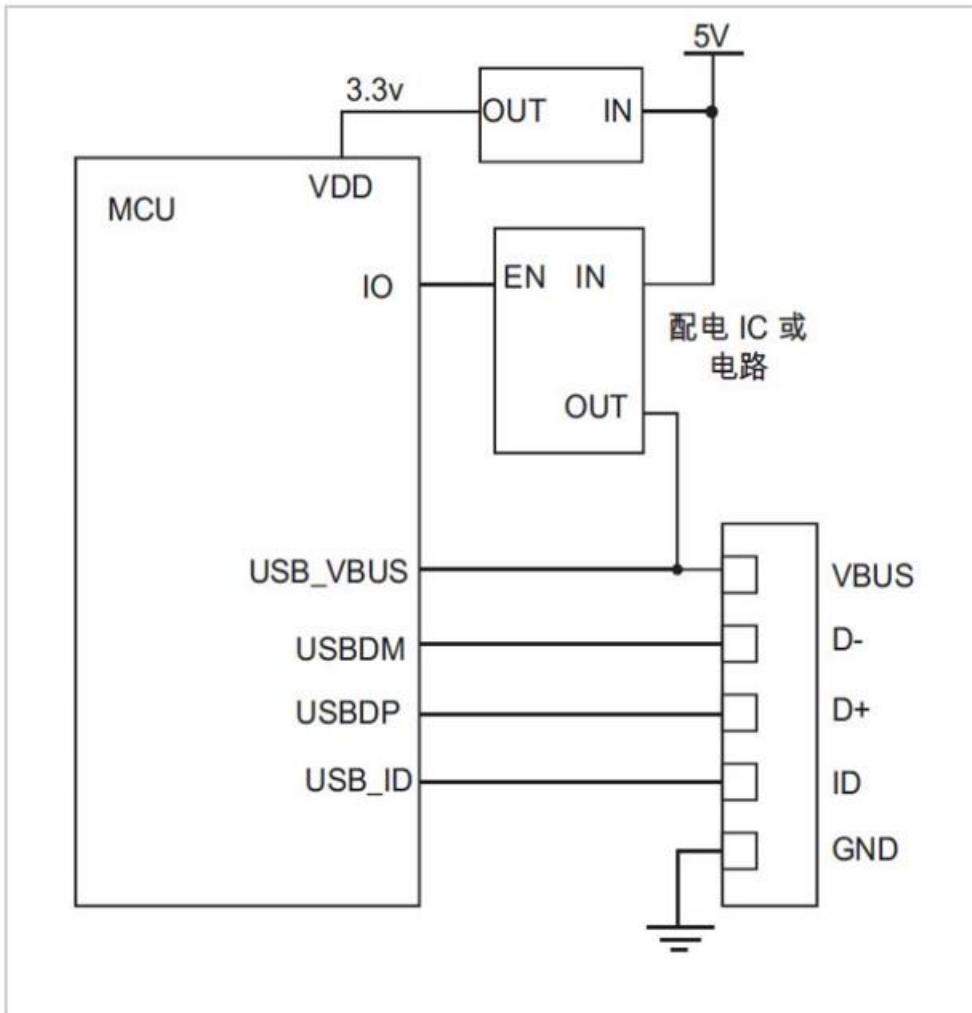


图 23-4 双角色设备框图

## 23.5 OTG-FS 软件编程接口

### 23.5.1 OTG-FS 数据传输方向

Rx (接收) 表示将数据从 USB 移动到内部存储方向, Tx (发送) 表示将数据从内部存储移动到 USB 方向。USB 令牌中数据方向是相对于主机而言, 所以主机的 IN 令牌和设备的 OUT/SETUP 令牌数据方向为 Rx, 主机的 OUT/SETUP 令牌或设备的 IN 令牌的数据方向为 Tx。

### 23.5.2 缓存描述表 (Buffer Descriptor Table)

为了有效地管理 USB 端点通信, OTG-FS 在系统存储器中实现缓冲区描述符表 (BDT)。BDT 起始地址位于系统内存的 512 字节的边界上, 并由 BDT PAGE 寄存器确定地址。每个端点方向部需要两个

---

八字节的缓冲区用来存放描述符条目。每个端点方向由两个缓冲区描述符条目（Buffer Descriptor Entries）组成（EVEN BD 和 ODD BD）。从而实现双缓冲 BD，OTG-FS 可以为 USB 提供的最大吞吐量轻松传输数据。

微处理器和 OTG-FS 都可以访问缓冲区，所以使用一个简单的信号机制来区分谁可以更新系统内存中的 BDT 和缓冲区。当 BD 由微处理器“拥有”时，信号位 OWN 为 0，此时微处理器允许对 BD 条目和系统存储器中的缓冲区进行读写访问。当 OWN 为 1 时，BD 条目和系统存储器中的缓冲区由 OTG-FS 控制，此时 OTG-FS 具有完全的读写访问权，微处理器不应修改 BD 或其相应的数据缓冲区。

BD 还包含间接地址指针，指向实际缓冲区驻留在系统内存中的位置。这个间接寻址机制如下图所示。

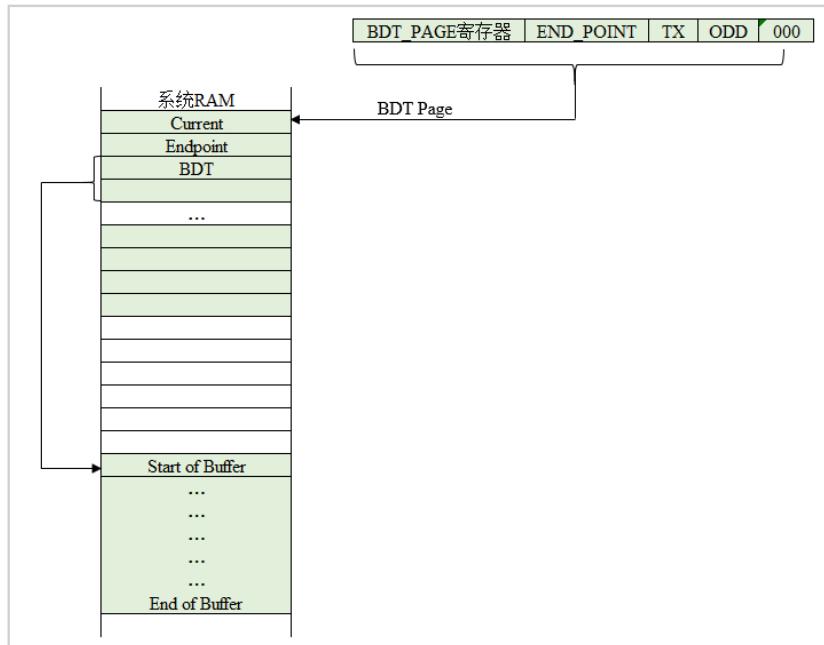


图 23-5 缓冲区描述符表

### 23.5.3 缓冲区描述表地址

要通过 OTG-FS 或微处理器访问端点数据，必须理解缓冲区描述符表的寻址机制。如前所述，每个 USB 端点方向需要 16 个字节。BDT 页面寄存器指向 BDT 的起始位置。BDT 起始地址位于系统内存的 512 字节的边界上。所有启用的 TX 和 RX 端点的 BD 条目部被索引到 BDT 中，以便通过 OTG-FS 或微处理器轻松访问。

当 OTG-FS 在启用的端点上接收到 USB 令牌时，它将使用其集成 DMA 控制器来询问 BDT。OTG-FS 必须读取相应的端点 BD 条目并确定它是否拥有系统存储器中的 BD 和相应的缓冲区的访问权限。为计算 BDT 的入口地址，BDT PAGE 寄存器与当前端点以及 TX 和 ODD 字段连接在一起形成一个 32 位地址。该地址机制如下图所示：

表格 23-2 BDT 地址计算字段结构

31 : 24	23 : 16	15 : 9	8: 5	4	3	2: 0
BDT PAGE 03	BDT PAGE 02	BDT PAGE 01[7:1]	END POINT	IN	ODD	RES.

表格 23-3 BDT 结构说明

BDT PAGE	BDT PAGE 寄存器
END POINT	STAT 寄存器中的 ENDP 位
TX	STAT 寄存器中的 TX 位，1 为 TX 发送传输，0 为 RX 接收传输
ODD	STAT 寄存器中的 ODD 位，该位在 OTG-FS SIE 设置。它对应于当前正在使用的缓冲区。缓冲区以 ping-pong 方式使用。

### 23.5.4 缓冲区描述符 (BD) 格式

缓冲区描述符 (BD) 为 OTG-FS 和微处理器提供端点缓冲区控制信息。基于谁在内存中读取 BD，缓冲区描述符具有不同的含义。

---

OTG-FS 控制器使用存储在 BD 中的数据来确定：

- 1) 谁拥有系统内存中的缓冲区
- 2) Data0 或 Data1 的 PID
- 3) 在数据包完成时释放自己
- 4) 是否有地址增量 (FIFO 模式)
- 5) 数据切换同步启用
- 6) 多少数据传输或接收
- 7) 缓冲区驻留在系统内存中的地址

微处理器使用存储在BD 中的数据来确定：

- 1) 谁拥有系统内存中的缓冲区
- 2) Data0 或 Data1 的 PID
- 3) 收到的 TOKEN PID
- 4) 传输或接收了多少数据
- 5) 缓冲区驻留在系统内存中的位置

BD 的具体格式如下表所示。

表 23-1 BD 结构

31:26	25:16	15:8	7	6	5	4	3	2	1	0
Res.	BC[9:0]	Res.	OWN	DATA0/1	KEEP/ TOK PID[3]	NINC/TOK PID[2]	DTS/ TOK PID[1]	BDT STALL /TOK PID[0]	0	0

Buffer Address (32-bits)

---

表 23-2 BD 结构说明

BC [9:0]	字节计数位。OTG-FS SIE 将在完成 RX 传输后并且计数结束后更改此 字段。
OWN	如果 OWN = 1, OTG-FS 可以独占访问 BD。如果 OWN = 0, 则微处理器可以独占访问 BD。这个 OWN 位决定了谁当前拥有缓冲区。SIE 在完成接收一个令牌时一般将 0 写入该位, 除非 KEEP = 1。当 OWN = 0 时, OTG-FS 忽略 BD 中的所有其他字段。当 OWN = 0 时, 微处理器可以访问整个 BD。BD 的这个字节应该始终是微处理器在初始化 BD 时更新的最后一个字节。一旦 BD 由 OTG-FS 拥有, 微处理器不应该以任何方式改变它。
DATA0/1	发送或接收 DATA0 或 DATA1 包。它不由 OTG-FS 改变。
KEEP/TOK PID[3]	如果 KEEP 等于 1, 一旦 OWN 位被置位, BD 将永远由 OTG-FS 持有。KEEP 必须等于 0 才能允许 OTG-FS 在令牌处理完成后释放 BD。ISO 端点正在填充一个 FIFO 时需设置此位。微处理器不会被告知令牌已被处理, 数据被简单地传送或接收到 FIFO。当 KEEP = 1 时, NINC 位通常也被置位, 以防止地址递增。如果 KEEP = 1, 则该位不会被 OTG-FS 改变, 否则当前令牌 PID 的位 3 被 OTG-FS 写回到 BD 中。

NINC/TOK PID[2]	NINC 位禁用 DMA 地址增量。这将强制 DMA 从相同的地址读取或写入数据。当数据需要从一个单一位置（如 FIFO）读取或写入单个地址时，这对端点很有用。通常情况下，该位与 KEEP 位一同设置，对于连接到 FIFO 的 ISO 端点。如果 KEEP = 1，则该位不会被 OTG-FS 改变，否则当前令牌 PID 的位 2 被 OTG-FS 写回 BD。
DTS/TOK PID[1]	设置该位将使 OTG-FS 执行数据切换同步。当该位为 0 时，不执行数据切换同步。如果 KEEP = 1，则该位不会被 OTG-FS 改变，否则当前令牌 PID 的位 1 被 OTG-FS 写回 BD。
BDT-STALL/TOK PID[0]	如果 SIE 使用此地址的 BDT 接收令牌，则设置此位将导致 OTG-FS 发出 STALL 握手。当设置 BDTSTALL 位时，SIE 不改变 BDT (OWN 保持，BDT 的其余部分保持不变)。 如果 KEEP = 1，则该位不会被 OTG-FS 改变，否则当前令牌 PI 的位 0 被 OTG-FS 写回到 BD
TOK PID[3:0]	当传输完成时，当前令牌 PID 由 OTG-FS 写回 BD。写回的值是来自 USB 规范的令牌 PID 值：OUT 令牌为 0x1，IN 令牌为 0x9，SETUP 令牌为 0xd。 在主机模式下，该字段用于报告上次返回的 PID 或传输状态指示。返回的可能值为：0x3 DATA0, 0xb DATA1, 0x2 ACK, 0xe STALL, 0xa NAK, 0x0 总线超时, 0xf 数据错误。
ADDR [31:0]	地址位表示系统存储器中的 32 位缓冲区地址。由用户设置。

### 23.5.5 USB 传输

当 OTG-FS 发送或接收数据时，首先计算 BDT 地址。一旦 BDT 被读取，并且 OWN 位为 1，SIE 就会将数据通过 DMA 的方式发送或接收到 BD 寄存器 ADDR 所指向的缓冲区。当令牌包传输完成时，如果 KEEP 为 0，则 OTG-FS 将更新 BDT 并将 OWN 位更改为 0。STAT 寄存器更新并设置 TOK\_DNE 中断。当微处理器处理 TOK\_DNE 中断时，首先读取 STAT 寄存器，这给微处理器提供处理端点所需的全部信息。此时，微处理器需要分配一个新的 BD，以便可以为该端点发送或接收额外的 USB 数据，然后处理最后一个 BD。下图显示了一个典型的 USB 令牌如何处理的时间线。

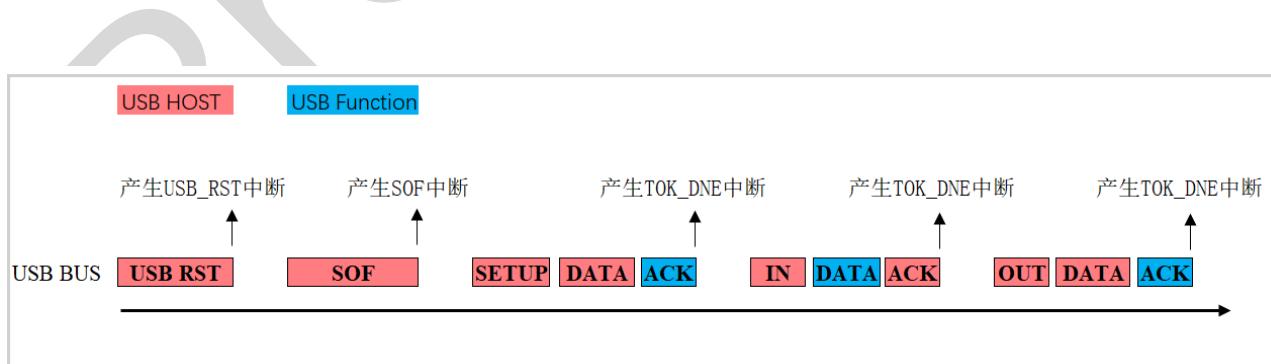


图 23-6 令牌传输

硬件性能问题如内部存储器延迟过大导致 FIFO 溢出，或者软件错误如接收到的数据包超过 Maxpacket 大小设定，将会导致 DMA 上溢错误。

由于硬件性能问题引起的 DMA 上溢错误，USB 将根据传输种类不同响应 NAK 或 BTO。主机模式

或者设备模式，ERR STAT 寄存器的 DMA ERR 位都将被置 1。通过设置 INT ENB 和 ERR ENB 寄存器，允许产生 DMA 错误中断。在设备模式下，BDT 不会被写回，也不会触发 TOK\_DNE 中断。在主机模式下，产生 TOK\_DNE 中断，并且 BDT 的 OK PID 字段将被设置为“1111”，以指示 DMA 延迟错误，另外主机模式软件可以决定重试或者跳转到其它流程。

为了保护内存中的数据，USB 对超过 MaxPacket 大小的数据包会将数据裁剪到 MaxPacket 大小，存入内部存储器，对于非同步传输返回 ACK。ERR STAT 寄存器的 DMA ERR 位将被置 1。通过寄存器设定中断许可，允许产生 DMA 错误中断，并且 TOK\_DNE 中断也将会被触发。回写到 BDT 的数据包长度将是 MaxPacket 值，用于表示实际写入内存的数据长度。软件可以取消传输，禁止断点等操作。

## 23.6 OTG 与 Host 模式操作方式

OTG-FS 支持主机 (Host) 模式和 OTG 操作 (HNP 和 SRP)。

Host 模式旨在用于手持便携式设备，以便于连接简单的 HID 类设备，如打印机和键盘。它不是要执行 PC 主板上的完整 OHCI 或 UHCI 兼容主机控制器的功能。主机模式支持 Control, Bulk, Interrupt 和 Isochronous 四种传输。

设置 CTL 寄存器中的 HOST\_MODE\_EN 位可启用主机模式。OTG-FS 内核只能作为外围设备或主机模式运行。它不能同时在两种模式下运行。当启用 HOST\_MODE 时，只使用端点 0，并且禁用其他端点。

## 23.7 Host 模式操作示例

以下部分说明使用 OTG-FS 内核执行 USB 主机功能所需的步骤。

### 23.7.1 启用主机模式并发现连接的设备

- 1) 使能主机模式 (CTL.HOST\_MODE\_EN = 1)。
- 2) 下拉电阻使能，上拉电阻禁用。并设置 CTL.USB\_EN = 1。
- 3) 等待设备连接
  - a) 使能 ATTACH 中断 (INT\_ENB.ATTACH = 1)。
  - b) 等待设备连接 等待 ATTACH 中断 (INT\_STAT.ATTACH)。
- 4) 获取连接设备速度
  - a) 检查控制寄存器中的 JSTATE 和 SE0 位的状态。如果 JSTATE 位为 0，则连接设备为低速设备。低速设备需设置 ADDR.LS\_EN = 1, EP\_CTL0.HOST\_WO\_HUB = 1。
- 5) 复位设备
  - a) 设定 CTL.RESET = 1。
- 6) 使能 USB
  - a) 设定 CTL.USB\_EN = 1。此时控制器开始发出 SOF，使连接的设备不会进入挂起状态。
- 7) 枚举以及后续操作。

## 23.7.2 与 USB 设备进行 Control 传输

设备连接后，通过以下步骤与设备进行控制传输。

- 1) 设置端点控制寄存器用于双向控制传输
  - a) EP\_CTL0 [4:0] = 0x0d。
- 2) 将需要用到的 Device Framework 相关指令存入内存缓冲区中。
- 3) GET\_DESCRIPTOR 相关设定。
  - a) 初始化当前 TX EP0 BDT（偶或奇），用于 GET\_DESCRIPTOR 命令。将 BD 设置为 0x00080080（BC=8, OWN=1）。将 BD 的 ADDR 地址字段设置为 GET\_DESCRIPTOR 命令缓冲区的起始地址。
- 4) SET\_ADDRESS 相关设定。
  - a) 在地址寄存器（ADDR [6:0]）中设置 USB 设备地址。
- 5) 执行控制传输的 Setup Stage。
  - a) 设置 TOKEN 寄存器为 EP0 的 SETUP 令牌包（TOKEN = 0xD0）。
  - b) SETUP 令牌传输完成后，设备握手将在 BDT PID 字段中返回。
  - c) BDT 写入完成（INT\_STAT [TOK\_DNE]）中断将被置位。
- 6) 执行控制传输的 Data Stage（有些命令不存在 Data Stage）
  - a) 在 RAM 中设置用于要传送的数据缓冲区。
  - b) 初始化当前 TX EP0 BDT（偶或奇）传输数据。将 BD 设置为 0x004000C0（BC=64, OWN=1, DATA0/1 = 1）。将 BD 的 ADDR 地址字段设置为数据缓冲区的起始地址。
  - c) 设置 TOKEN 寄存器为 EP0 的 IN 令牌或者 OUT 令牌，启动 IN 或者 OUT 传输。对于 GET\_DESCRIPTOR 的 Data Stage，设置 TOKEN 寄存器为 EP0 的 IN 令牌。启动 IN 传输。
  - d) 当传输完成时，BDT 被写入并且完成令牌（INT\_STAT [TOK\_DNE]）中断置位。
- 以上步骤完成了一个数据包的传输，如果有多个数据包，遵从 USB 规定的 Data Toggle Synchronization 机制，完成 Data Stage 的操作。
- 7) 执行控制传输的 Status Stage
  - a) 在 RAM 中设置缓冲区以接收或发送零长度的数据包。
  - b) 初始化当前 TX EP0 BDT（偶或奇）传送状态数据。将 BD 设置为 0x00000080（BC=0, OWN=1, DATA0/1 = 0）。将 BD ADDR 地址字段设置为数据缓冲区的起始地址。
  - c) 设置 TOKEN 寄存器为 EP0 的 IN 令牌或者 OUT 令牌，启动 IN 或者 OUT 传输，接收或者发送零长度数据包。对于 GET\_DESCRIPTOR 的 Status Stage，设置 TOKEN 寄存器为 EP0 的 OUT 令牌。启动 OUT 传输。

- 
- d) 当传输完成时, BDT 被写入并且完成令牌 (INT\_STAT [TOK\_DNE]) 中断置位。

### 23.7.3 与 USB 设备进行 Bulk OUT 传输

设备连接并完成配置后, 通过以下步骤与设备进行 BULK OUT 传输。

- 1) 设置端点控制寄存器用于双向传输 EP CTL0 = 0x1d。
- 2) 向 ADDR 寄存器写入设备地址。
- 3) 设置偶数 TX EP0 BDT 可传送多达 64 个字节 (FS 操作时)。
- 4) TOKEN 寄存器设置为所需的端点的 OUT 令牌方式。
- 5) 设置奇数 TX EP0 BDT 可传输多达 64 个字节 (FS 操作时)。
- 6) TOKEN 寄存器设置为所需的端点的 OUT 令牌方式。

步骤 3~6 实现了双缓冲结构, 提升了吞吐量。

- 7) 等待 TOK\_DNE 中断。
  - a) 置位则表示其中一个 BDT 已经被释放给处理器, 并且传输已经完成。
  - b) RETRY\_DIS=0 时, 设备返回 NAK 时, OTG-FS 将继续无限期地在无处理器干预的情况下重试传输。
  - c) RETRY\_DIS=1 时, 重试传输功能被禁止, 则 BDT PID 字段将返回握手 PIN。
  - d) 对于 STALL 相应, 需要将待处理的数据包从队列中移除, 并清除目标设备中的错误条件。
  - e) 如果发生复位中断, 表明设备失去连接状态。

重复步骤 3~7 可以实现多个数据的连续传输。

## 23.8 OTG 操作

处理器通过软件来实现主机协商协议 (HNP) 和会话请求协议 (SRP)。以下状态图显示了 USB 电缆两端的 HNP 和 SRP 协议涉及的 OTG 操作。

### 23.8.1 OTG 双角色 A 设备操作方式

A 设备为将标准 A 或 Mini-A 插头插入其插座的设备。A 设备为 VBUS 供电; 是会话开始时为 host; 在某些条件下, A 设备将放弃主机对双重角色 B 设备的控制。

双重角色“A”设备将按照以下流程图和状态描述表进行操作。

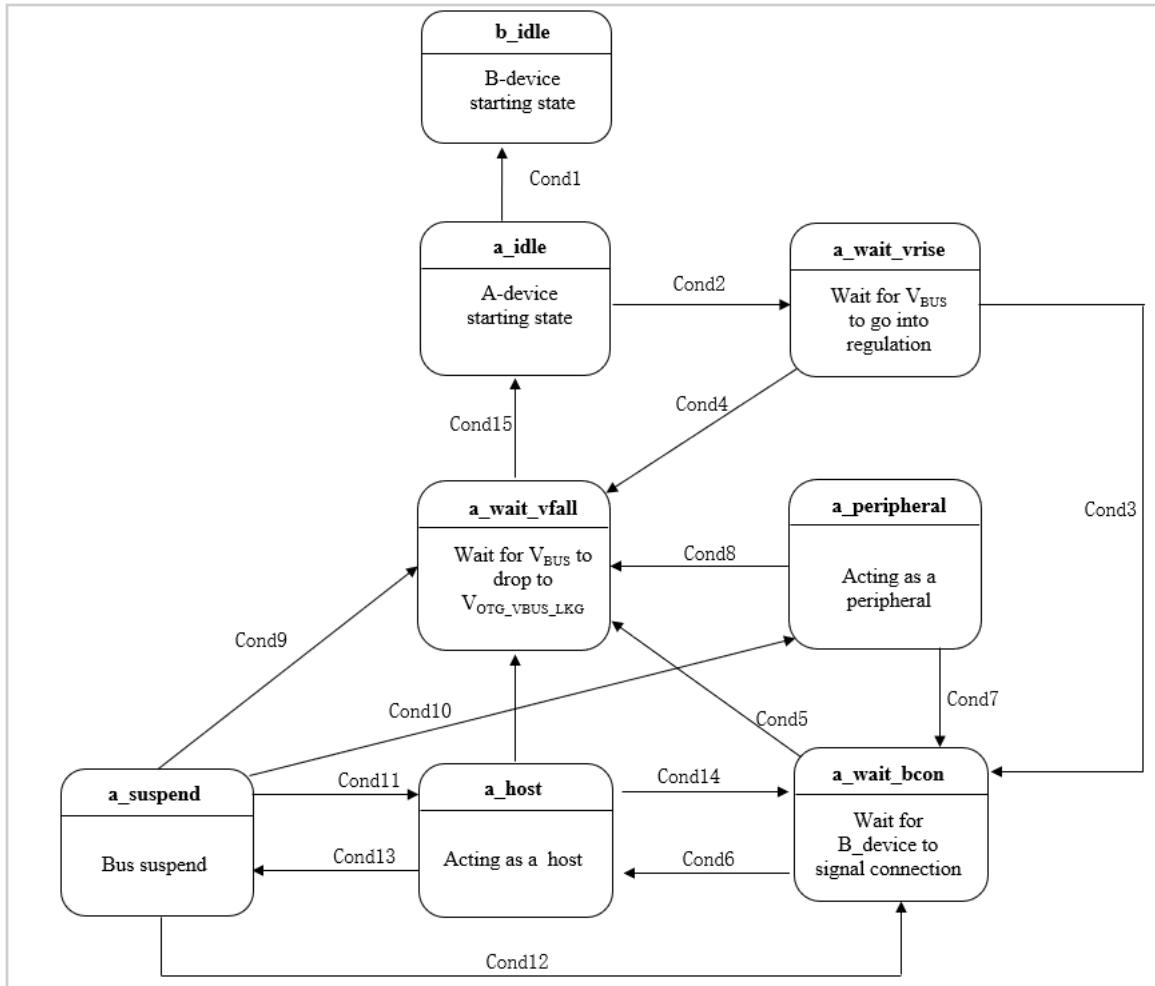


图 23-7 OTG 双角色A 设备操作方式

表 23-3 OTG 双角色 A 设备操作方式

条件	描
Cond1	ID 线中断。电缆未插入或已连接B-device。
Cond2	如果 “A” 应用程序想要使用总线，或者 “B” 设备正在执行一个 SRP，如 A_SESS_VLD 中断或插入或端口状态改变中断响应，检查数据线脉冲 5-10 毫秒。
Cond3	如果 A_VBUS_VLD 中断。
Cond4	如果ID中断响应或者如果100毫秒之后A_VBUS_VLD 错误. 电缆已改变或“A”设备无法支持“B”设备所需的电流。转到a_wait_vfall状态后，关闭DRV_VBUS。
Cond5	200毫秒后没有连接或ID中断。转到a_wait_vfall状态后，关闭DRV_VBUS。
Cond6	A_VBUS_VLD 中断且“B”设备连接上。
Cond7	如果总线空闲3-200毫秒。
Cond8	如果ID中断或者如果A_VBUS_VLD 中断。转到a_wait_vfall状态后，关闭DRV_VBUS。
Cond9	如果ID中断，或者如果150 毫秒“B”断开超时（该超时值可能更长），或者如果A_VBUS_VLD 中断。转到a_wait_vfall状态后，关闭DRV_VBUS。

Cond10	如果启用HNP，并且“B”在150毫秒内断开，则“B”设备正在成为主机。
Cond11	如果“A”想要开启另一个会话。
Cond12	ID 线中断，或者 “B” 未连接。
Cond13	如果“A”设备完成会话，或者“A”设备想要允许“B”设备控制总线。
Cond14	ID中断或“B”设备断开连接。
Cond15	如果ID中断或(A_SESS_VLD / & b conn/)

### 23.8.2 OTG 双角色 B 设备操作方式

B 设备为标准 B 或 Mini-B 插头插入其插座的设备。B 设备是会话开始时的外围设备。如果设备是双重角色，则可以从 A 设备授予主机角色。

双角色 “B” 设备将按照以下流程图和状态描述表所示进行操作。

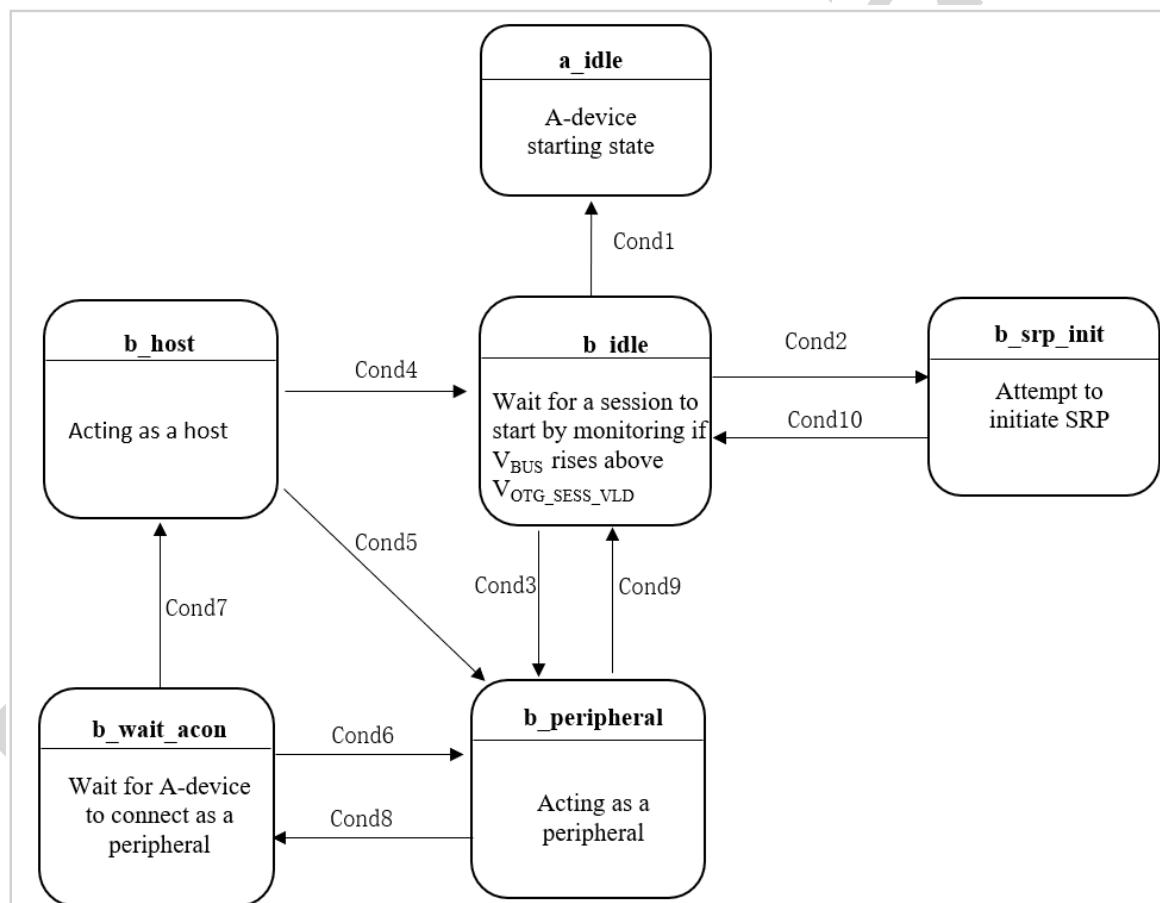


图 23-8 OTG 双角色 B 设备操作方式

表 23-4 OTG 双角色 B 设备操作方式

条件	描
Cond1	如果ID中断。“A”型电缆已经插入，设备现在应该作为“A”型设备

Cond2	如果“B”应用程序想要总线控制权且总线处于空闲状态2 ms 并且B_SESS_END 位置位，则“B”设备可以执行SRP。 在 <b>b_srp_init</b> 状态，脉冲CHRG_VBUS，DP_HIGH产生5~10毫秒的脉冲。
Cond3	如果B_SESS_VLD中断。“A”设备已打开VBUS并开始会话。 在 <b>b_peripheral</b> 状态，打开DP_HIGH。
Cond4	如果ID中断或B_SESS_VLD中断。
Cond5	如果“B”应用程序已完成或“A”断开连接。
Cond6	超过3.125 毫秒或发生resume。
Cond7	如果启用HNP 并且总线挂起，如果“B”想要总线，则“B”设备可以成为主机。
Cond8	如果启用HNP 并且总线挂起，如果“B”想要总线，则“B”设备可以成为主机。 在 <b>b_wait_acon</b> 状态，关闭DP_HIGH
Cond9	如果 ID 中断或 B_SESS_VLD 中断。
Cond10	如果ID中断或SRP完成（SRP 必须在小于100 毫秒内完成）。

## 23.9 寄存器描述

表 23-5 USB 寄存器概览

Offset	Acronym	Register Name	Reset
0x10	OTG_ISTAT	OTG Interrupt Status Register	0x00E8
0x14	OTG_ICTRL	OTG Interrupt Control Register	0x0000
0x18	OTG_STAT	OTG Status Register	0x00A8
0x1C	OTG_CTRL	OTG Control register	0x0000
0x80	INT_STAT	Interrupt status register	0x0001
0x84	INT_ENB	Interrupt enable register	0x0000
0x88	ERR_STAT	Error interrupt status register	0x0000
0x8C	ERR_ENB	Error interrupt enable register	0x0000
0x90	STAT	Status register	0x0000
0x94	CTL	Control register	0x0040
0x98	ADDR	Address register	0x0000
0x9C	BDT_PAGE_01	BDT page register 1	0x0000
0xA0	FRM_NUML	Frame number register	0x0000
0xA4	FRM_NUMH	Frame number register	0x0000
0xA8	TOKEN	Token register	0x0000
0xAC	SOF_THLD	SOF threshold register	0x0000
0xB0	BDT_PAGE_02	BDT page register 2	0x0000

Offset	Acronym	Register Name	Reset
0xB4	BDT_PAGE_03	BDT page register 3	0x0000
0xC0~0xFC	EP_CTLn	Endpoint control register	0x0000

### 23.9.1 OTG 中断状态寄存器 (OTG\_ISTAT)

偏移地址: 0x10

复位值: 0x0000

OTG 中断状态寄存器记录 ID PIN 和 VBUS PIN 的信号变化。 软件读取该寄存器以确定导致中断的事件。 仅置位自上次软件读取后已更改的位。 写 1 清除相应的中断标志。

Bit	1 5	1 4	1 3	1 2	1 1	1 0	9	8	7	6	5	4	3	2	1	0
Field	Reserved							ID_CHG	1_MSEC	LINE_STA	Reseved	SESS_VL	B_SESS_E	Reseved.	A_VBUS_V	LD_CHG
Type								rcw1	rcw1	rcw1		rcw1	rcw1		rcw1	

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留
7	ID_CHG	rcw1	0x00	当检测到USB ID PIN 信号发生变化时，该位置位。
6	1_MSEC	rcw1	0x00	该位在1毫秒定时器到期时置位。该位保持置位直到被软件清零。中断必须每毫秒服务一次，以避免丢失1毫秒的计数值。
5	LINE_STATE_CHG	rcw1	0x00	当USB线路状态（CTL寄存器的 SE0和JSTATE位）稳定，在1毫秒内无变化，并且线路状态的值与线路状态最后一次稳定时的值不同时，该中断被设置。当SE0和J，SE0和K以及J和K之间产生转换，它将被置位。当SE0位为1时，JSTATE的改变不会引起中断。该中断可用于检测复位，恢复。
4	Reserved			保留
2	SESS_VLD_CHG	rcw1	0x00	当检测到VBUS发生变化时，该位置位，表示会话有效或会话不再有效。
2	B_SESS_EN_D_CHG	rcw1	0x00	在“B”设备上检测到VBUS发生变化时，该位置位。
1	Reserved			保留
0	A_VBUS_VL_D_CHG	rcw1	0x00	在“A”设备上检测到VBUS变化时，该位置1。

### 23.9.2 OTG 中断控制寄存器 (OTG\_ICTRL)

偏移地址: 0x14

复位值: 0x0000

OTG 中断控制寄存器使能 OTG 中断状态寄存器中定义的相应中断状态位。

Bit	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0	
Field	5	4	3	2	1	0				ID_EN	1_MSEC_EN	LINE_STABLE_EN	Reserved	SESS_VLD_EN	B_SESS_END_EN	Reserved	A_VBUS_VLD_EN
Type	Reserved									rw	rw	rw		rw	rw		rw

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留
7	ID_EN	rw	0x00	使能 ID 信号中断
6	1_MSEC_EN	rw	0x00	使能 1 毫秒定时器中断。
5	LINE STATE_EN	rw	0x00	使能线路状态变化中断。
4	Reserved			保留
3	SESS_VLD_EN	rw	0x00	使能会话有效中断。
2	B_SESS_END_EN	rw	0x00	使能 B 设备会话结束中断。
1	Reserved			保留
0	A_VBUS_VLD_EN	rw	0x00	使能 A 设备 VBUS 有效中断

### 23.9.3 OTG 状态寄存器 (OTG\_STAT)

偏移地址: 0x18

复位值: 0x0000

OTG 状态寄存器显示 ID PIN 和 VBUS 与比较器的实际输出值。

Bit	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0	
Field	5	4	3	2	1	0				ID	1_MS EC	LINE STATE STABLE	Reser ved.	SESS _VLD	B_SESS _END	Reser ved.	A_VBUS _VLD
Type	Reserved									rw	rw	rw		rw	rw		rw

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留, 始终读为 0
7	ID	rw	0x00	指示 USB 连接器上的 ID 引脚的当前状态。

Bit	Field	Type	Reset	Description
				0 表示 A 型电缆已插入 USB 连接器 1 表示未连接电缆或 B 型电缆已插入 USB 连接器
6	1_MSEC	rw	0x00	该位为 1 毫秒计数器保留，对软件无效。
5	LINE STATE_STABLE	rw	0x00	当线路状态 (CTL 寄存器中的 JSTATE 和 SE0) 在前 1 毫秒内稳定时，该位置 1。该位用于在检测连接，断开和恢复信号时提供线路状态的硬件去抖动。当使用该位时，首先读取 CTL 寄存器中的 SE0 和 JSTATE 的状态。然后读出此位。如果该位为 1，那么 CTL 寄存器的 JSTATE 和 SE0 位的值在 1 毫秒内不变，可以被认为去抖动的。
4	Reserved			保留。
3	SESS_VLD	rw	0x00	当 VBUS 电压高于 B 设备会话的有效阈值时，该位将被置位。
2	B_SESS_END	rw	0x00	当 VBUS 电压低于 B 设备会话结束阈值时，该位将被置位。
1	Reserved			保留
0	A_VBUS_VLD	rw	0x00	当 VBUS 电压高于 A 设备 VBUS 有效阈值时，该位将被置位。

### 23.9.4 OTG 控制寄存器 (OTG\_CTRL)

偏移地址: 0x1C

复位值: 0x0000

OTG 控制寄存器在 OTG\_EN 置位后，控制 VBUS 电压和数据线电阻的上拉及下拉。

Bit	1 5	1 4	1 3	1 2	1 1	1 0	9 8	7	6	5	4	3	2	1	0	
Field	Reserved								DP_HIG H	DM_H IGH	DP_L OW	DM_L OW	VBUS _ON	OTG _EN	VBUS_ CHG	VBUS_D SCHG
Type									rw	rw	rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留。
7	DP_HIGH	rw	0x00	置位时，启用 D+ 数据线的上拉电阻。
6	DM_HIGH	rw	0x00	置位时，启用 D- 数据线的上拉电阻。

Bit	Field	Type	Reset	Description
5	DP_LOW	rw	0x00	置位时，启用 D+数据线的下拉电阻。
4	DM_LOW	rw	0x00	置位时，启用 D-数据线上的下拉电阻。
3	VBUS_ON	rw	0x00	置位时，打开 VBUS 电源信号。
2	OTG_EN	rw	0x00	置位时，这个寄存器中的 D+和 D-上拉和下拉控制位有效。 当 OTG_EN = 0 且 USB_EN.HOST=0 时，为设备模式，D+上拉被使能。 当 OTG_EN = 0 且 USB_EN.HOST=1 时，为主机模式，D+ 和 D-的下拉被使能。
1	VBUS_CHG	rw	0x00	置位时，将通过电阻发送 VBUS 信号。
0	VBUS_DSCHG	rw	0x00	置位时，这将通过电阻释放 VBUS 信号。

### 23.9.5 中断状态寄存器 (INT\_STAT)

偏移地址: 0x80

复位值: 0x0000

中断状态寄存器 (INT\_STAT) 包含 OTG-FS 中每个中断源的位。这些位中的每一位都使用各自的中断使能位进行控制(参见中断使能寄存器)。寄存器的所有位逻辑“或”在一起形成微处理器的单个中断源。中断位置位时，可以通过写 1 清零。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved								STALL	ATTACH	RESUM	SLEEP	TOK_DNE	SOF_TOK	ER_R	US_B_R
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留，始终读为 0
7	STALL	rw	0x00	STALL 中断用于从机模式和主机模式，在从机模式下，SIE 发送 STALL 握手包时置位。 在主机模式下，如果 OTG-FS 在 USB 事务的握手阶段检测到 STALL 信号，则该位置位。该中断用于确定最后的 USB 事务是成功完成还是 STALL 状态
6	ATTACH	rw	0x00	如果 OTG-FS 检测到 USB 外设的连接，则该位置位。该信号仅在 HOST_MODE_EN 为真时有效。此中断对于表示外设已连接并且必须进行配置很有用。如果在 USB 数据线 2.5us 没有状态转换并且当前总线状态不是 SE0，则

Bit	Field	Type	Reset	Description
				ATTACH 中断被置位。
5	RESUME	rw	0x00	主机模式用于检测设备的 RESUME 信号。 设备模式用于发送 Remote Wakeup 信号。 不处于挂起状态时，需禁止该中断。
4	SLEEP	rw	0x00	如果 OTG-FS 在 USB 总线信号上检测到 3ms 空闲，则该位置 1。
3	TOK_DNE	rw	0x00	当正在处理的令牌完成时，该位被置位。微处理器应立即读取 STAT 寄存器以确定相关的端点和 BD。 将该位清零（通过写入 1）会导致 STAT 寄存器被清零或 STAT 保持寄存器被装入 STAT 寄存器。
2	SOF_TOK	rw	0x00	设备模式下，如果 OTG-FS 收到 SOF 令牌，则此位置位。 在主机模式下，当达到 SOF 阈值时，该位被置位，以便软件可以为下一个 SOF 做准备。
1	ERROR	rw	0x00	当 ERR_STAT 寄存器中的任何错误条件发生时，该位置位。微处理器必须读取 ERR_STAT 寄存器来确定错误的来源。
0	USB_RST	rw	0x00	当 OTG-FS 解码有效的 USB 复位时，该位置位。这将通知微处理器将 0x00 写入地址寄存器并使能端点 0。一旦检测到 USB 复位 2.5 微秒，USB_RST 将被置位。在 USB 复位条件被移除之前，它不会被再次置位。

### 23.9.6 中断使能寄存器 (INT\_ENB)

偏移地址：0x84

复位值：0x0000

设置这些位中的任何一位将使能 INT\_STAT 寄存器中的相应中断源。复位后该寄存器值为 0x00 即禁止所有中断。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved						STALL	ATTAC H	RESUM E	SLEEP	TOK_ DNE	SOF_ TOK	ER RO R	US B_R ST		
Type							rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留，始终读为 0
7	STALL	rw	0x00	设置该位使能STALL中断。
6	ATTACH	rw	0x00	设置该位使能ATTACH中断。
5	RESUME	rw	0x00	设置该位使能RESUME中断。
4	SLEEP	rw	0x00	设置该位使能SLEEP中断。
3	TOK_DNE	rw	0x00	设置该位使能TOK_DNE中断。
2	SOF_TOK	rw	0x00	设置该位使能SOF_TOK中断。
1	ERROR	rw	0x00	设置该位使能ERROR中断。
0	USB_RST	rw	0x00	设置该位使能USB_RST中断。

### 23.9.7 错误中断状态寄存器 (ERR\_STAT)

偏移地址: 0x88

复位值: 0x0000

这些位中的每一位都使用各自的错误使能位进行控制（请参见错误使能寄存器页面）。结果进行“或”运算并发送到 INT\_STAT 寄存器的 ERROR 位。通过写 1 清除对应的状态位。一旦检测到错误条件，对应的状态位就会置位。因此，中断通常不会与正在处理的令牌的结尾相对应。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved							BTS_E RR	Reser ved	DMA_E RR	BTO_E RR	DFN8 6	CRC1 F	CR C5/ EO F	PID _ER R	
Type								rw		rw	rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留。
7	BTS_ERR	rw	0x00	检测到 bit stuff 错误时置位。 相应的数据包将被拒绝。
6	Reserved			保留
5	DMA_ERR	rw	0x00	如果 OTG-FS 请求 DMA 访问来读取新的 BDT, 但在 OTG-FS 需要接收或发送数据之前没有给出总线，则该位置 1，如果处理 TX 传输，将导致发送数据下溢，或者如果处理一个 Rx 传输，这会引起一个接收数据溢出的情况。  如果来自主机或发向主机的数据包大于在 BDT 中分配的缓冲区大小，则该位也被设置。在这种情况下，数据包在被放入缓冲存储器时被截断。

Bit	Field	Type	Reset	Description
4	BTO_ERR	rw	0x00	如果发生总线周转超时错误，则该位置位。此 OTG-FS 使用总线周转定时器来跟踪 SETUP 或 OUT TOKEN 的令牌和数据阶段或 IN TOKEN 的数据和握手阶段之间经过的时间。如果在 IDLE 转换之前，从先前的 EOP 中计数，多于 16 位的时间，则会出现总线周转超时错误。
3	DFN8	rw	0x00	收到的数据字段不是 8 位。USB 规范 1.0 规定数据字段必须是整数个字节。如果数据字段不是整数个字节，则该位将被设置。
2	CRC16	rw	0x00	CRC16 校验失败。如果数据包由于 CRC16 错误而被拒绝，则该位置位。
1	CRC5/EOF	rw	0x00	这个错误中断有两个功能。 当 OTG-FS 工作在外设模式 (HOST_MODE_EN = 0) 时，该中断将检测主机生成的令牌数据包中的 CRC5 错误。如果设置令牌数据包由于 CRC5 错误而被拒绝。 当 OTG-FS 工作在主机模式 (HOST_MODE_EN = 1) 时，该中断将检测帧结束 (EOF) 错误情况。当 OTG-FS 正在发送或接收数据并且 SOF 计数器已经达到零时，就会发生这种情况。这个中断在开发 USB 数据包调度软件时很有用，以确保没有 USB 事务跨越下一帧的开始。
0	PID_ERR	rw	0x00	PID 检查字段失败。

### 23.9.8 错误中断使能寄存器 (ERR\_ENB)

偏移地址: 0x8C

复位值: 0x0000

错误中断使能寄存器 (ERR\_ENB) 包含 OTG-FS 中每个错误中断源的使能位。设置任何这些位将使能 ERR\_ISTAT 寄存器中的相应错误中断源。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved								BTS_E	Reser	DMA_E	BTO_E	DFN8	CRC1	CR	PID
Type									RR	ved	RR	RR	6	E0	_ER	R

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留
7	BTS_ERR	rw	0x00	设置该位使能 BTS_ERR 中断。
6	Reserved			保留
5	DMA_ERR	rw	0x00	设置该位使能 DMA_ERR 中断。
4	BTO_ERR	rw	0x00	设置该位使能 BTO_ERR 中断。
3	DFN8	rw	0x00	设置该位使能 DFN8 中断。
2	CRC16	rw	0x00	设置该位使能 CRC16 中断。
1	CRC5/EOF	rw	0x00	设置该位使能 CRC5/EOF 中断。
0	PID_ERR	rw	0x00	设置该位使能 PID_ERR 中断。

### 23.9.9 状态寄存器 (STAT)

地址偏移: 0x90

复位值: 0x0000

当微处理器收到 TOK\_DNE 中断时，应读取状态寄存器以确定先前端点通信的状态。当 TOK\_DNE 中断位置位时，状态寄存器中的数据有效。

STAT 寄存器实际上是 OTG-FS 维护的状态 FIFO 的读取窗口。当 OTG-FS 使用 BD 时，它会更新状态寄存器。如果在 TOK\_DNE 中断服务之前执行了另一个 USB 事务，则 OTG-FS 将把下一个事务的状态存储在 STAT FIFO 中。

因此，STAT 寄存器实际上是一个四字节 FIFO，它允许微处理器在 SIE 处理下一个事务时处理一个事务。

清零 INT\_STAT 寄存器中的 TOK\_DNE 位会导致 SIE 使用下一个 STAT 值的内容更新 STAT 寄存器。如果 STAT 保持寄存器中的数据有效，则 SIE 将立即重新置位 TOK\_DNE 中断。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved								ENDP				TX	ODD	Reserved	
Type									r				r	r		

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留
7:4	ENDP[3:0]	r	0x00	对接收或发送前一个令牌的端点地址进行编码。这允许微处理器确定哪个 BDT 条目由最后的 USB 事务更新。
3	TX	r	0x00	该位指示上次更新的 BDT 是用于的传输类型 TX = 1 表示发送数据传输。 TX = 0 表示接收数据传输。
2	ODD	r	0x00	该位表示更新的最后一个缓冲区描述符位于 BDT 的奇数组中，请参阅前面的部分以获取有关 BDT 地址生成的更多

Bit	Field	Type	Reset	Description
				信息。
1:0	Reserved			保留

### 23.9.10 控制寄存器 (CTL)

地址偏移: 0x94

复位值: 0x0000

控制寄存器为 OTG-FS 提供各种控制和配置信息。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved								JSTAT	SE0	TxdSuspend		HOST_MOD	RESUME	ODD_RST	USB_E_N
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留
7	JSTATE	rw	0x00	USB 差分接收器接收到 JSTATE 信号。该信号的极性受 LS_EN 的当前状态的影响 (参见下面的地址寄存器说明)。
6	SE0	rw	0x00	USB 接收到 SE0 信号。
5	TxdSuspend TokenBusy	rw	0x00	当 OTG-FS 为从机模式时为 TxdSuspend, 当 OTG-FS 处于主机模式时为 TokenBusy。 TxdSuspend 位通知处理器 SIE 已禁用数据包发送和接收。清除该位允许 SIE 继续令牌处理。当接收到 SETUP 令牌包时, 该位由 SIE 置位, 允许软件在恢复令牌处理之前使 BDT 中的任何挂起的数据包事务出队。 TokenBusy 位通知主机处理器 OTG-FS 正忙于执行 USB 令牌, 而不应将更多令牌命令写入令牌寄存器。在将任何令牌命令写入令牌寄存器之前, 软件应检查此位, 以确保令牌命令不会丢失。
4	RESET	rw	0x00	设置该位将使 OTG-FS 产生 USB 复位信号。这将允许 OTG-FS 重置 USB 外设。该控制信号仅在主机模式有效 (即, HOST_MDOE_EN = 1)。软件必须将 RESET 设置为 1 达所需的时间, 然后将其清除为 0 以结束复位

Bit	Field	Type	Reset	Description
				信号。具体请 USB 参考协议规范。
3	HOST_MODE_EN	rw	0x00	设置该位将使 OTG-FS 在主机模式下工作。在主机模式下，OTG-FS 将在主处理器的程序控制下执行 USB 事务。
2	RESUME	rw	0x00	设置该位将允许 OTG-FS 执行恢复信号。这将允许 OTG-FS 执行远程唤醒。软件必须将 RESUME 设置为 1 达所需的时间，然后将其清零。如果 HOST_MODE_EN 位被置位，当 RESUME 位清零时，OTG-FS 将把低速末端数据包追加到 Resume 信号中。有关 RESUME 信号的更多信息，请参阅 USB 规范版本 1.0 的第 7.1.4.5 节。
1	ODD_RST	rw	0x00	设置此位将重置所有的 BDT ODD ping / pong 位为 0，然后指定偶数 BDT 库。
0	USB_EN	rw	0x00	设置该位将使 OTG-FS 工作，清除它将禁用 OTG-FS。将该位置 1 将导致 SIE 将其所有的奇数位复位到 BDT。因此，设置该位会重置 SIE 中的大部分逻辑。当主机模式使能时，清除该位将导致 SIE 停止发送 SOF 令牌。

### 23.9.11 地址寄存器 (ADDR)

地址偏移: 0x98

复位值: 0x0000

地址寄存器 (ADDR) 包含 OTG-FS 在外设模式下解码的唯一 USB 地址 (HOST\_MODE\_EN = 0)。

在主机模式 (HOST\_MODE\_EN = 1) 下运行时，OTG-FS 将使用 TOKEN 数据包发送此地址。这将使 OTG-FS 能够唯一地寻址任何 USB 外设。

在任一模式下，必须设置控制寄存器中的 USB\_EN 位。

在复位输入激活或 USBFS 已解码 USB 复位信号后，寄存器复位为 0x00。这将初始化地址寄存器以解码 USB 规范所要求的地址 0x00。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved										LS_EN	ADDR[6: 0]				
Type											rw	rw				

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留
7	LS_EN	rw	0x00	使能此位将告知 OTG-FS 写入令牌寄存器的下一个令牌命

Bit	Field	Type	Reset	Description
				令必须低速执行。这将使 OTG-FS 能够执行低速数据传输所需的要求。
6:0	ADDR[6:0]	rw	0x00	该 7 位值定义了 OTG-FS 在外设模式下解码的 USB 地址，或者在主机模式下传输的 USB 地址。

### 23.9.12 缓冲区描述符表页寄存器 1 (BDT\_PAGE\_01)

地址偏移: 0x9C

复位值: 0x0000

缓冲区描述符表页寄存器包含一个 8 位值，用于计算当前缓冲区描述符表 (BDT) 驻留在系统存储器中的地址。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														Reserve	
Type	BDT_BA[15: 9]														Reserve	

Bit	Field	Type	Reset	Description
15:8	Reserved			保留。
7:1	BDT_BA[15: 9]	rw	0x00	该7位值提供BDT基地址的地址位15至9，其定义了缓冲器描述符表在系统存储器中的位置。内存中的32位BDT基地址总是与512字节边界对齐。
0	Reserved			保留。

### 23.9.13 低位帧数寄存器 (FRM\_NUML)

地址偏移: 0xA0

复位值: 0x0000

帧编号寄存器包含 11 位帧编号。帧编号寄存器需要两个 8 位寄存器来实现，低位字节包含在 FRM\_NUML 中，高位字节包含在 FRM\_NUMH 中。当接收到 SOF TOKEN 时，这些寄存器用当前帧号更新。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														FRM[7: 0]	
Type	Reserved														rw	

Bit	Field	Type	Reset	Description
15:8	Reserved			保留
7:0	FRM[7: 0]	rw	0x00	这些位表示 11 位帧数的低 8 位。

### 23.9.14 高位帧数寄存器(FRM\_NUMH)

地址偏移: 0xA4

复位值: 0x0000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved														FRM[10: 8]	
Type															rw	

Bit	Field	Type	Reset	Description
15:3	Reserved			保留
2:0	FRM[10: 8]	rw	0x00	这些位表示11位帧数的高3位。

### 23.9.15 令牌寄存器 (TOKEN)

地址偏移: 0xA8

复位值: 0x0000

令牌寄存器用于在主机模式下执行 USB 事务 (HOST\_MODE\_EN = 1)。当主处理器希望对外设执行 USB 事务时，它会将 TOKEN 类型和端点写入该寄存器。一旦写入该寄存器，OTG-FS 就会开始指定的 USB 事务

在执行对令牌寄存器的写操作之前，主机处理器应始终检查控制寄存器中的 TOKEN\_BUSY 位是否未置 1。这将确保令牌命令在执行之前不会被覆盖。地址寄存器和端点控制寄存器 0 也在执行令牌命令时使用，因此也必须在令牌寄存器写入之前写入。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved							TOKEN_PID[3: 0]	TOKEN_ENDPT[3: 0]							
Type								rw								

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留，始终读为 0
7:4	TOKEN_PID[3: 0]	rw	0x00	此 4 位值是由 VSUB 执行的令牌类型，有效令牌是： 0001: OUT 令牌 OTG-FS 将执行 OUT (TX) 事务 1001: IN 令牌 OTG-FS 将执行 IN (RX) 事务 1101: SETUP 令牌 OTG-FS 将执行 SETUP (TX) 事 务
3:0	TOKEN_ENDPT[3: 0]	rw	0x00	此 4 位值决定了令牌命令的端点地址。写入的四位值必 须是有效的端点。

### 23.9.16 SOF 阈值寄存器 (SOF\_THLD)

地址偏移: 0xAC

复位值: 0x0000

SOF 阈值寄存器仅在 HOST\_MODE 中使用。

当 HOST\_MODE 使能时, 14 位 SOF 计数器计算 SOF 帧之间的间隔。

SOF 必须每 1ms 发送一次, 因此 SOF 计数器加载值为 12000. 当 SOF 计数器达到零时, 发送帧起始 (SOF) 令牌。

SOF 阈值寄存器用于在 SOF 停止启动令牌包事务之前编程 USB 字节时间的数量。

必须将此寄存器设置为一个值, 以确保在 SOF 计时器计数到零时不会主动传输其他数据包。

当 SOF 计数器达到阈值时, 在发送 SOF 之前不再发送令牌。

编程到阈值寄存器中的值必须保留足够的时间以确保最坏情况下的事务将完成。

通常, 最坏情况事务是 IN 令牌后跟来自目标的数据包, 然后是来自主机的响应。

所需的实际时间是总线上最大数据包大小的函数。

SOF 阈值的典型值为: 64 字节数据包= 74; 32 字节包= 42; 16 字节包= 26; 8 字节包= 18。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved								CNT[7: 0]							
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留。
7:0	CNT[7: 0]	rw	0x00	这些位代表字节时间的 SOF 计数阈值。

### 23.9.17 缓冲区描述符表页寄存器 2 (BDT\_PAGE\_02)

地址偏移: 0xB0

复位值: 0x0000

缓冲区描述符表页寄存器包含一个 8 位值, 用于计算当前缓冲区描述符表 (BDT) 驻留在系统存储器中的地址。有关缓冲区描述符表格式的更多信息, 请参见前面的部分。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Reserved								BDT_BA[23: 16]								
Type									rw								

Bit	Field	Type	Reset	Description
15: 8	Reserved			保留。
7:0	BDT_BA[23: 16]	rw	0x00	该 8 位值提供 BDT 基地址的地址位 23 至 16, 其定义了缓冲器描述符表在系统存储器中的位置。内存中的 32 位

Bit	Field	Type	Reset	Description
				BDT 基地址总是与 512 字节边界对齐。

### 23.9.18 缓冲区描述符表页寄存器 3 (BDT\_PAGE\_03)

地址偏移: 0xB4

复位值: 0x0000

缓冲区描述符表页寄存器包含一个 8 位值，用于计算当前缓冲区描述符表 (BDT) 驻留在系统存储器中的地址。有关缓冲区描述符表格式的更多信息，请参见前面的部分。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved															BDT_BA[31: 24]
Type																rw

Bit	Field	Type	Reset	Description
15:8	Reserved			保留。
7:0	BDT_BA[31: 24]	rw	0x00	该 8 位值提供了 BDT 基地址的地址位 31 至 24，其定义了缓冲器描述符表在系统存储器中的位置。内存中的 32 位 BDT 基地址总是与 512 字节边界对齐。

### 23.9.19 端点控制寄存器 (EP\_CTL0~15)

地址偏移: 0xC0–0xFC

复位值: 0x0000

在收到 USB\_RST 中断后，微处理器应将 ENDPT0 设置为包含 0x0D。

在主机模式下，ENDPT0 用于确定主机传输的握手，重试和低速特性。对于主机模式控制，批量和中断传输，EP\_HSHK 位应设置为 1，对于同步传输，应将其设置为 0.在主机模式下用于 ENDPT0 的公共值为 0x4D 用于控制，批量和中断传输，0x4C 用于同步传输。

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved					HOST_WO_H	RETR_Y_DIS	Res.	EP_CT_L_DIS	EP_R_X_EN	EP_T_X_EN	EP_STA_LL	EP_HS_HK			
Type						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description																								
15: 8	Reserved			保留, 始终读为 0																								
7	HOST_WO_HUB	rw	0x00	这是一个仅用于主机模式的位, 仅存在于端点 0 的控制寄存器中。 设置此位允许主机与直接连接的低速设备进行通信。 当被清除时, 主机将产生 PRE_PID, 然后根据需要通过集线器向低速设备发送令牌给低速设备时切换到低速信号。																								
6	RETRY_DIS	rw	0x00	这是一个仅用于主机模式的位, 仅存在于端点 0 的控制寄存器中。 当设置这个位时, 主机不会重试 NAK 的事务。 当事务处于 NAK 状态时, 将使用 NAK PID 更新 BDT PID 字段, 并设置令牌完成中断。 当这个位被清 0 后, 被 NAK 的事务将在硬件中重试。 当主机试图查询中断端点时, 必须设置该位。																								
5	Reserved			保留																								
4: 2	EP_CTL_DIS /EP_RX_EN/EP_TX_EN	rw	0x00	<p>EP_CTL_DIS /EP_RX_EN/EP_TX_EN:</p> <p>这三位定义了端点是否被使能以及端点的方向。 端点使能/方向控制定义如下:</p> <table border="1"> <thead> <tr> <th>EP_CTL_DIS</th> <th>EP_RX_EN</th> <th>EP_TX_EN</th> <th>使能及方向</th> </tr> </thead> <tbody> <tr> <td>X</td> <td>0</td> <td>0</td> <td>关闭端点</td> </tr> <tr> <td>X</td> <td>0</td> <td>1</td> <td>仅 TX 传输</td> </tr> <tr> <td>X</td> <td>1</td> <td>0</td> <td>仅 RX 传输</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>TX 及 RX 传输</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>控制传输及 TX 及 RX</td> </tr> </tbody> </table>	EP_CTL_DIS	EP_RX_EN	EP_TX_EN	使能及方向	X	0	0	关闭端点	X	0	1	仅 TX 传输	X	1	0	仅 RX 传输	1	1	1	TX 及 RX 传输	0	1	1	控制传输及 TX 及 RX
EP_CTL_DIS	EP_RX_EN	EP_TX_EN	使能及方向																									
X	0	0	关闭端点																									
X	0	1	仅 TX 传输																									
X	1	0	仅 RX 传输																									
1	1	1	TX 及 RX 传输																									
0	1	1	控制传输及 TX 及 RX																									
1	EP_STALL	rw	0x00	当该位置位时, 表示端点已经 STALL。 该位优先于 EPCTL 寄存器中的所有其他控制位, 但仅在 EP_TX_EN = 1 或 EP_RX_EN = 1 时有效。 对此端点的任何访问都将导致 OTG-FS 返回一个 STALL 握手。 一旦端点停止, 它需要主机控制器的干预。																								
0	EP_HSHK	rw	0x00	设置该位定义了端点是否在事务处理期间执行到该端点的握手。 除非是同步端点, 否则应当置位。																								

## 24 以太网 MAC 控制器（ETHERMAC）

### 24.1 ETHERMAC 简介

以太网是一种计算机局域网技术。IEEE 组织的 IEEE802.3 标准制定了以太网的技术标准，它规定了包括物理层的连线、电气信号和介质访问层协议的相关内容。以太网是目前应用最普遍的局域网技术。

以太网 MAC 控制器（ETHERMAC）用于在以太网网络中发送和接收数据，它有多种应用领域，如交换机、网络接口卡等。本产品中以太网 MAC 控制器支持与外部物理层（PHY）相连的两个工业标准接口：介质独立接口（MII）和简化介质独立接口（RMII）。

### 24.2 ETHERMAC 主要特征

#### 24.2.1 架构框图

下图为以太网 MAC 控制器的架构框图。

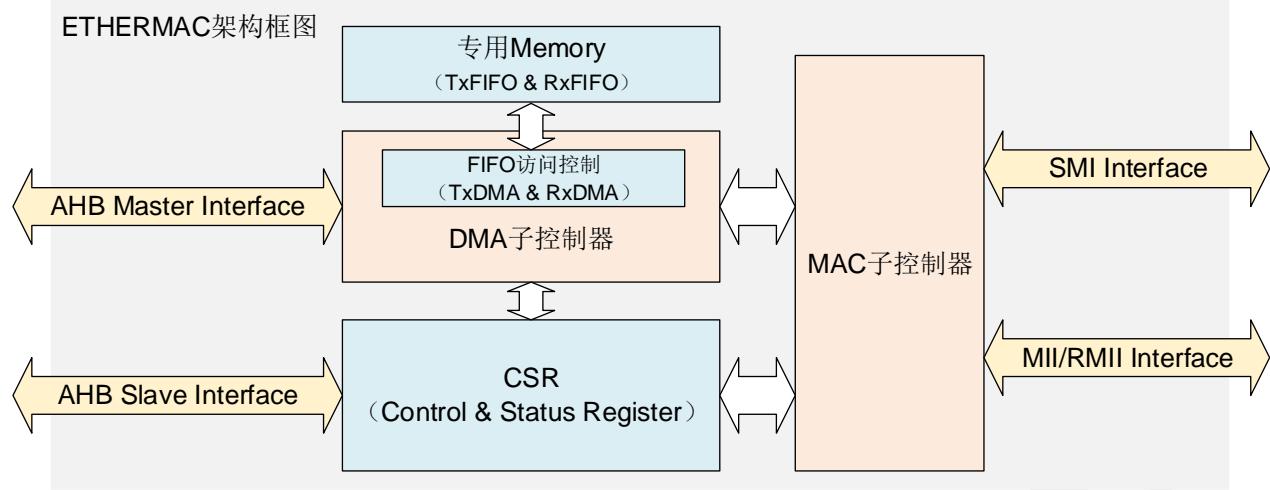


图 24-1 ETHERMAC 架构框图

以太网 MAC 控制器包括 AHB Master 接口、AHB Slave 接口、DMA 子控制器、MAC 子控制器、接口控制逻辑等。DMA 子控制器通过 AHB 接口将 MAC 子控制器和系统存储器相连。

AHB Master 接口用于数据传输，AHB Slave 接口用于该控制器的基本寄存器访问配置等。

在发送数据时，将数据从系统存储器由 DMA 子控制器内的 TxDMA 引擎送至专用 Memory (Tx FIFO)，经 MAC 子控制器后通过 MII 接口或 RMII 接口送至外部 PHY。

在接收数据时，经过 MII 接口或 RMII 接口接收进来的数据，经 MAC 子控制器后送至专用 Memory (Rx FIFO)，DMA 子控制器的 RxDMA 引擎再将接收数据传送到系统存储器。

以太网 MAC 控制器内部各个主要模块的特性如下：

- MAC 子控制器主要特性

- 支持外部 PHY 接口实现 10/100Mbps 数据传输速率
- 为应用程序提供单独的发送、接收和控制接口
- 使用 SMI 接口配置和管理最多 32 个 PHY 设备
- 支持通过 MII 进行内部回送的 LoopBack 模式
- 支持半双工操作的 CSMA/DA 协议
- 支持半双工操作的背压流量控制
- 全双工操作时可以将接收的暂停控制帧转发到应用程序
- 全双工操作中如果流量控制输入信号消失，将自动发送零时间片暂停帧
- 接收以太网帧时，自动去除长度字段小于 1536 的帧的 PAD 和 FCS 字段
- 接收以太网帧时，自动去除类型帧的 FCS 字段
- 接收以太网帧时，自动进行接收帧的 CRC 计算
- 发送以太网帧时，对小于 60 字节的帧自动生成 PAD 填充
- 发送以太网帧时，处理冲突帧的自动重新发送
- 发送以太网帧时，支持可编程帧间隔（40-96 位时间，以 8 为步长）

- 
- 支持高达 32 个 48 位完美 (DA) 地址过滤器, 对每个字节进行掩码操作
  - 支持高达 31 个 48 位完美 (SA) 地址过滤器, 对每个字节进行掩码操作
  - 支持 64 位 Hash 过滤器, 适用于单播和多播目标地址过滤
  - 可传送所有多播地址帧
  - 可传送所有帧, 无需为网络监视进行过滤
  - 传送所有传入数据包时 (每次过滤时) 均附有一份状态报告
  - 支持一组 VLAN 标记过滤
  - 在接收帧时, 可选择丢弃无 TCP/UDP 字段的帧
  - 对接收到的 IPv4 报的 Header Checksum 字段进行校验
  - 对接收到的 TCP 报、UDP 报、ICMP 报的 Checksum 字段进行校验
  - 对发送的 IPv4 报进行 Checksum 计算, 将计算结果插入到 Header Checksum 字段
  - 对发送的 TCP 报、UDP 报、ICMP 报进行 Checksum 计算, 将计算结果插入到 Checksum 字段
  - 具有一个可编程阈值的 512Byte 专用发送 Memory (TxFIFO) 和一个可配置阈值的 256Byte 专用接收 Memory (RxFIFO)
  - RxFIFO 和 TxFIFO 均支持存储转发模式
  - 软件控制刷新 TxFIFO
  - RxFIFO 在存储转发模式下, 可选择过滤所有的错误帧, 不将这些错误帧转发给应用
  - RxFIFO 可根据填充 (阈值可配置) 级别自动生成要发送给 MAC 子控制器的暂停控制帧或背压信号
  - RxFIFO 中丢失或损坏的帧可进行数据统计
- . DMA 子控制器主要特性
    - 支持软件设定 AHB Master 接口的 AHB 突发类型 (固定或不确定突发)
    - AHB Slave 接口中支持所有 AHB 突发类型
    - AHB Master 接口可选择地址对齐突发传输
    - 支持 RxDMA 和 TxDMA 引擎单独编程突发长度, 以充分利用总线
    - 支持对数据缓冲区进行字节对齐寻址
    - 支持环式和链式两种描述符链接方式
    - 每个描述符可传输高达 8KB 的数据
    - TxDMA 和 RxDMA 均支持存储转发模式
    - 报告正常工作和传输错误时的综合状态
    - 可编程中断选项, 根据不同的应用场景选择不同的中断
    - 按帧控制发送/接收完成中断

- 
- 接收引擎和发送引擎间采用循环调度仲裁或固定优先级仲裁

## 24.2.2 功能管脚

下表显示了 MAC 信号和 MII/RMII 接口的映射。所有 MAC 信号均映射到 AF11，一些信号映射到不同的 I/O 引脚，这些应在复用功能模式下进行配置。

表 24-1 ETHERMAC 管脚功能表

名称	型号类型		注释
	MII 接口	RMII 接口	
PA0_WKP0	ETH_MII_CRS	-	载波侦听,当发送或接收介质处于非空闲状态时,由 PHY 使能该信号;发送和接收介质均处于空闲状态时,由 PHY 禁止该信号。PHY 必须确保 ETH_MII_CRS 信号在冲突条件下保持有效状态。该信号无需与发送和接收时钟保持同步。在全双工模式下,该信号没意义。
PA1	ETH_MII_RX_CLK	ETH_RMII_REF_CLK	提供进行 RX 数据传输时的参考时序。MII 接口时,速率为 10Mbps 时为 2.5MHz、速率为 100Mbps 时为 25MHz ; RMII 接口时,为 50MHz。
PA2	ETH_MDIO	ETH_MDIO	数据输入/输出比特流,用于通过 ETH_MDC 时钟信号与 PHY 设备进行数据信息传输。
PA3	ETH_COL	-	冲突检测,检测到介质上存在冲突后,PHY 必须立即使能冲突检测信号,并且只要存在冲突条件,冲突检测信号必须保持有效状态。该信号无需与发送和接收时钟保持同步。在全双工模式下,该信号没意义。
PA7	ETH_MII_RX_DV	ETH_RMII_CRS_DV	接收数据有效,表示 PHY 当前正对 MII 接口或 RMII 接口发送数据。为正确地接收帧,该信号必须在时间范围上涵盖要接收的帧,其开始时间不得迟于 SFD 字段出现的时间。
PB0	ETH_MII_RXD2	-	
PB1	ETH_MII_RXD3	-	
PB8	ETH_MII_TXD3	-	
PB10	ETH_MII_RX_ER	-	接收错误,该信号必须保持一个或多个发送时钟周期,从而向 MAC 子层指示在帧的某处检测到错误。

名称	型号类型		注释
	MII 接口	RMII 接口	
PB11	ETH_MII_TX_EN	ETH_RMII_TX_EN	发送使能，表示 MAC 当前正通过 MII 接口或 RMII 接口发送数据。
PB12	ETH_MII_TXD0	ETH_RMII_TXD0	发送数据，由 MAC 子层同步驱动。 在 ETH_MII_TX_EN/ETH_RMII_TX_EN 信号有效时，才为有效信号；在 ETH_MII_TX_EN/ETH_RMII_TX_EN 信号无效时，发送数据不会对 PHY 产生任何影响。
PB13	ETH_MII_TXD1	ETH_RMII_TXD1	
PC1	ETH_MDC	ETH_MDC	周期性时钟，提供最大频率 2.5MHz 传输数据时的参考时序。MDC 的最短高电平和最短低电平必须均为 160ns。MDC 的最小周期必须为 400 ns。在空闲状态下，SMI 将 ETH_MDC 时钟信号驱动为低电平。
PC2	ETH_MII_TXD2	-	
PC3	ETH_MII_TX_CLK	-	提供进行 TX 数据传输时的参考时钟。速率为 10Mbps 时为 2.5MHz；速率为 100Mbps 时为 25MHz。
PC4	ETH_MII_RXD0	ETH_RMII_RXD0	接收数据，由外部 PHY 同步驱动。 在 ETH_MII_RX_DV/ETH_RMII_VRS_DV 信号有效时，才为有效信号。
PC5	ETH_MII_RXD1	ETH_RMII_RXD1	
PE2	ETH_MII_TXD3	-	

### 24.2.3 连接关系

MII 接口和 PHY 芯片的连接关系如下图：

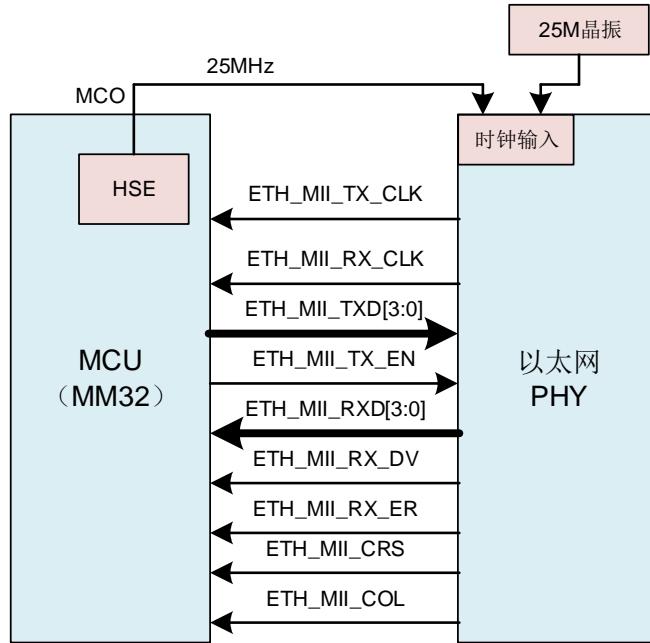


图 24-2 MII 接口与 Device PHY 连接图

要生成 ETH\_MII\_TX\_CLK 和 ETH\_MII\_RX\_CLK 时钟信号，必须向外部 PHY 提供 25MHz 时钟，如图所示。除了使用外部 25 MHz 石英晶体提供该时钟，还可以通过本产品的 MCO 引脚输出该信号。

RMII 接口和 PHY 芯片的连接关系如下图：

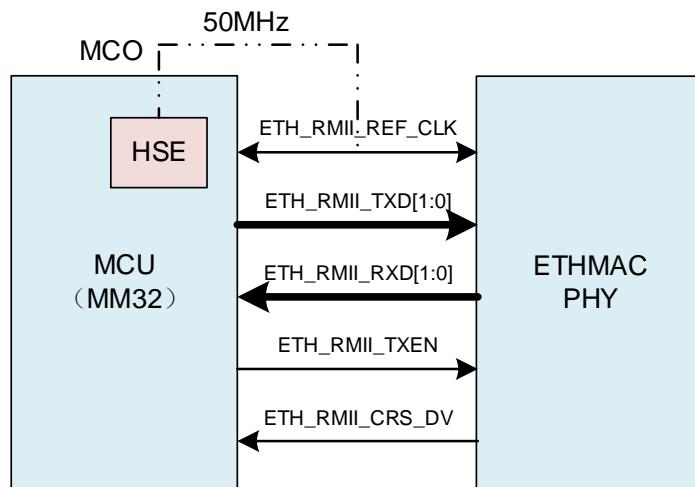


图 24-3 RMII 接口与 Device PHY 连接图

要生成 ETH\_RMII\_REF\_CLK 可以通过本产品的 MCO 引脚输出该信号。这种情况下，必须对产品内部 PLL 倍频进行配置，从而在 MCO 引脚上获得所需频率。

## 24.2.4 802.3 帧格式

以太网 MAC 层的数据通信可使用两种帧格式：

- 1) 基本MAC帧格式
- 2) 标识MAC帧格式（扩展了基本MAC帧格式）

其基本的帧格式如下图所示。

基本MAC帧格式

前导码 PR (7 Byte)	帧首定界码 SFD (1 Byte)	目标地址 DA (6 Byte)	源地址 SA (6 Byte)	长度/类型 LEN/TYPE (2 Byte)	数据 DATA (46~1500字节)	填充PAD	帧校验序列 FCS (4 Byte)
-----------------------	--------------------------	------------------------	-----------------------	-------------------------------	---------------------------	-------	--------------------------

标记 MAC帧格式

前导码 PR (7 Byte)	帧首定界码 SFD (1 Byte)	目标地址 DA (6 Byte)	源地址 SA (6 Byte)	802.1Q VLAN Tag (4 Byte)	长度/类型 LEN/TYPE (2 Byte)	数据 DATA (46~1500字节)	填充PAD	帧校验序列 FCS (4 Byte)
Tag Protocol ID “0x8100” (2byte)      User Priority (3bits)      CFI (1bit)      VLAN ID (12bits)								

图 24-4 以太网帧格式

除了帧校验序列 FCS，以太网 MAC 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 32CRC 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：

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

通过以下条件之一定义一个帧为无效 MAC 帧：

- 1) 帧长度与长度/类型字段指定的预期值不一致。如果长度/类型字段包含类型值，则认为帧长度与此字段一致（没有无效帧）
- 2) 帧长度不是字节的整数倍（额外位）
- 3) 根据传入帧计算出的CRC值与包含的FCS不匹配

## 24.3 ETHERMAC 功能描述

### 24.3.1 MAC 子控制器基本特征

MAC 子控制器执行以下与数据链路控制步骤相关的功能：

- 数据封装（发送和接收）
  - 组帧（帧边界界定、帧同步）
  - 寻址（处理源地址和目标地址）
  - 错误检测
- 介质访问管理
  - 介质分配（冲突避免）
  - 竞争解决（冲突处理）

MAC 子控制器主要有两个工作模式：

- 半双工模式：站点使用CSMA/CD算法争用物理介质
- 全双工模式：满足以下条件时，无需解决竞争问题（CSMA/CD算法不是必需的）便可同时发送和接收数据：

- 
- 物理介质能够支持同步发送和接收
  - 正好有2个站点与LAN相连
  - 两个站均配置为全双工工作模式

## 24.3.2 MAC 子控制器接收功能

MAC 子控制器接收的帧将送入 RxFIFO，在接收达到一定数量条件时，就会指示 RxDMA 进行数据传送，RxDMA 可向 AHB 接口发起预配置的突发传输。

以太网 MAC 控制器搭载 256Byte 的 RxFIFO。RxDMA 将 RxFIFO 中的数据输送到系统存储器有两种操作模式：阈值模式和存储转发模式。

### 阈值模式

在阈值模式下，当 RxFIFO 接收收到的字节数大于配置的接收阈值时（使用 ETH\_DMAMDR 寄存器中的 RTC 位配置）或完整的数据包时，数据将发起输送，其可用性将通知给 RxDMA。RxDMA 向 AHB 接口发起传输后，数据传输将从 RxFIFO 持续进行，直到传输完成整个数据包。完成 EOF 帧的传输后，接收状态字将返回并发送到 RxDMA。在该模式下，某些错误帧不会被丢弃，因为在帧结束时接收到错误状态，而此时已从 RxFIFO 将数据传送进系统存储器。

### 存储转发模式

在存储转发模式下，当帧完全写入 RxFIFO 后才可读出帧。在此模式下，将丢弃所有错误帧（如果配置为执行此操作），这样只会读出有效帧并将其转发到应用。

接收数据后状态立即可用，因此只要 RxFIFO 未满，就可以向 RxFIFO 中存储帧。

当 MAC 在 MII 上检测到 SFD 时，将启动接收操作。MAC 子控制器将去除报头和 SFD，然后再继续处理帧。检查报头字段以进行过滤，FCS 字段用于验证帧的 CRC。如果帧未通过地址过滤器，则丢弃该帧。

### 24.3.2.1 接收协议

MAC 接收数据时，首先去除接收的帧的报头和 SFD。检测到 SFD 后，开始向 RxFIFO 发送以太网帧数据，从 SFD 后面的第一个字节（目标地址）开始发送。

如果接收的帧长度/类型字段小于 0x600 并且为 MAC 编程了自动去除 CRC/PAD 选项，则 MAC 将向 RxFIFO 发送帧数据（数据量不超过长度/类型字段中指定的数量），然后开始丢弃字节（包括 FCS 字段）。如果长度/类型字段大于或等于 0x600，则不管编程的自动 CRC 去除选项的值如何，MAC 都会向 RxFIFO 发送所有接收到的以太网帧数据。

默认情况下，使能 MAC 看门狗定时器，即超过 2048 个字节（DA+SA+LT+数据+PAD+FCS）的帧会被切断。可通过对 MAC 配置寄存器（ETH\_MACCR）中的看门狗禁止（WTD）位编程来禁止此功能。但是，即使禁止看门狗定时器，仍将切断大于 16KB 的帧并给出看门狗超时状态。

### 24.3.2.2 错误处理

如果在从 MAC 接收 EOF 数据之前 RxFIFO 已满，则将声明上溢并丢弃整个帧，同时帧丢失统计寄存器（ETH\_DMAFLCR）中的上溢计数器将递增。由于上溢，状态将指示这是一个部分帧。如果使能相应功能（ETH\_DMAMDR 中的 FEF 和 FUF 位），RxFIFO 可过滤错误帧和过小帧。

在阈值模式下，从 RxFIFO 读取帧 SOF 时，可获取当前帧的基本信息，通过与 Rx 描述符的基本设定比对，则可判定该帧是否为错误帧，并丢弃整个错误帧；在存储转发模式下，则可在数据传送之前过滤并丢弃所有错误帧。

### 24.3.2.3 CRC&PAD 处理

MAC 子控制器将计算接收的帧（包括目标地址字段到 FCS 字段）的 32 位 CRC，并检查接收帧中的任何 CRC 错误。不管是否自动去除 CRC/PAD（通过 ETH\_MACCR 寄存器 APCS 位控制），MAC 都将接收整个帧来计算所接收帧的 CRC 校验，并将校验的结果通过接收描述符 RDES0 的 CRE 位反馈给应用。

### 24.3.2.4 状态字

以太网帧接收结束时，MAC 输出接收状态通过 RxDMA 送到接收描述符 RDES0 中以供应用参考查询。

### 24.3.2.5 暂停帧

在帧发送过程中，MAC 将检测接收暂停帧并暂停帧发送，暂停时间为接收的暂停帧内指定的延迟（仅限全双工模式）。可通过 ETH\_MACFCR 中的 FRE 位使能或禁止暂停帧检测功能。使能接收流量控制后，将开始监视接收帧的目标地址是否与控制帧的多播地址（0x0180C2000001）匹配。如果检测到匹配（接收的帧的目标地址与保留的控制帧的目标地址匹配），MAC 将根据 ETH\_MACAFR 中的 PCF 位来决定是否将接收的控制帧传输到应用。

MAC 还将对接收到的控制帧的类型、操作码和暂停定时器字段进行解码。如果状态的字节计数指示 64 个字节，并且不存在任何 CRC 错误，则 MAC 发送器将暂停任何数据帧的发送，暂停时间为解码的暂停时间值乘以时隙（对于 10/100Mbps 模式，均为 64 字节时间）。同时，如果检测到另一个零暂停时间值的暂停帧，MAC 将复位暂停时间并管理新的暂停请求。如果接收到的控制帧与类型字段（0x8808）、操作码（0x00001）以及字节长度（64 字节）均不匹配，或存在 CRC 错误，则 MAC 不会生成暂停。

对于具有多播目标地址的暂停帧，MAC 将根据地址匹配来过滤帧；对于具有单播目标地址的暂停帧，MAC 将根据 DA 字段是否与 MAC 地址寄存器 0 的内容匹配以及 ETH\_MACFCR 中的 UP 位是否置 1（检测具有单播目标地址的暂停帧）来进行过滤。ETH\_MACAFR 寄存器中的 PCF 位可对控制帧的过滤以及地址过滤进行控制。

### 24.3.2.6 COE 引擎

以太网 MAC 控制器支持对接收的以太网帧中的 IPv4 和 IPv6 帧进行检测和处理，以确保数据完整性。通过将 ETH\_MACCR 寄存器中的 IPCO 位置 1 来使能接收 COE 引擎。MAC 通过检查所接收的以太网帧的类型字段中是否存在值 0x0800 或 0x86DD，来识别 IPv4 或 IPv6 帧。此识别方法也适用于带 VLAN 标记的帧。

接收 COE 引擎可计算 IPv4 报头校验和，并检查其是否与接收的 IPv4 报头校验和匹配。如果指示的有效负载类型（以太网类型字段）与 IP 报头版本之间有任何不匹配，或接收的帧的字节数小于 IPv4 报头的长度字段中指示的数量（IPv4 或 IPv6 报头中的可用字节数少于 20），IP 报头错误位将置 1。

接收 COE 引擎还可以识别接收的 IP 数据报（IPv4 或 IPv6）中的 TCP、UDP 或 ICMP 有效负载，并正确计算此类有效负载的校验和字段，包括用于校验和计算的 TCP/UDP/ICMPv6 伪报头字节，并检查接收的校验和字段是否与计算的值匹配。

COE 引擎将绕过分段 IP 数据报的有效负载、带安全功能的 IP 数据报、IPv6 路由报头以及除 TCP、UDP 或 ICMP 以外的有效负载。

### 24.3.2.7 接收时序

在 RMII 接口时，MAC 内部的每个半字节都从接收自 RMII 的双位发送到 MII，半字节发送顺序如下图所示。先接收位序较低的位（D0 和 D1），再接收位序较高的位（D2 和 D3）。

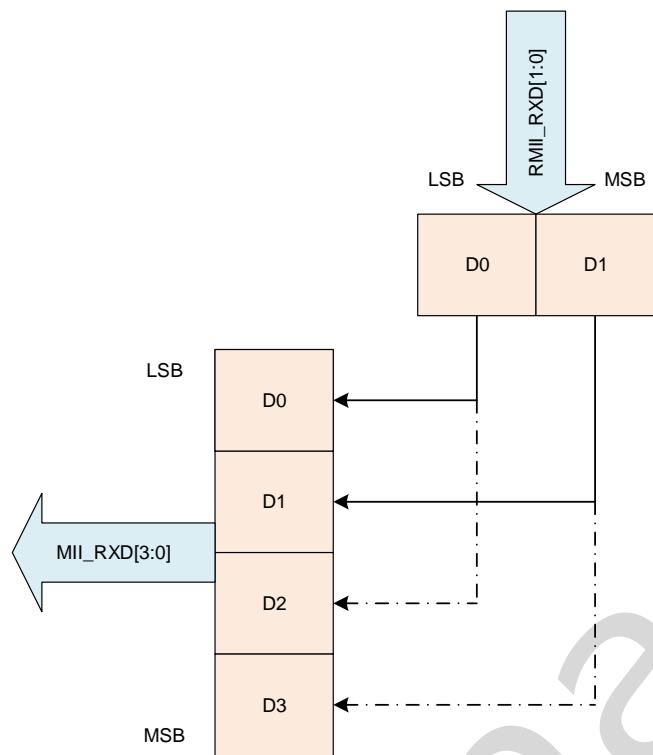


图 24-5 MII/RMII 接收时序图

下面所示为基本的接收时序图。

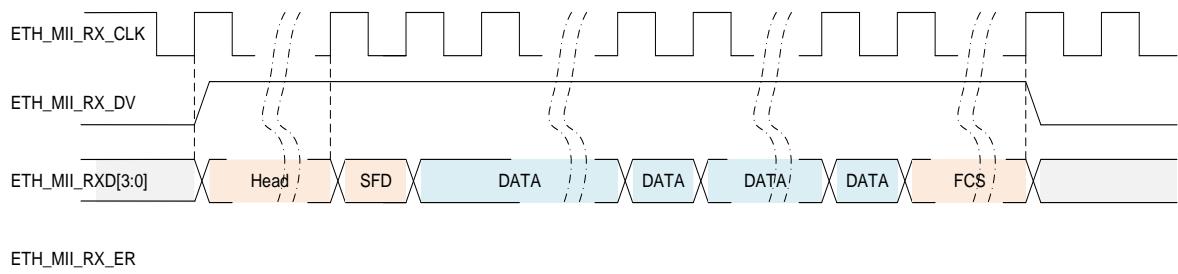


图 24-6 无错误时接收时序图

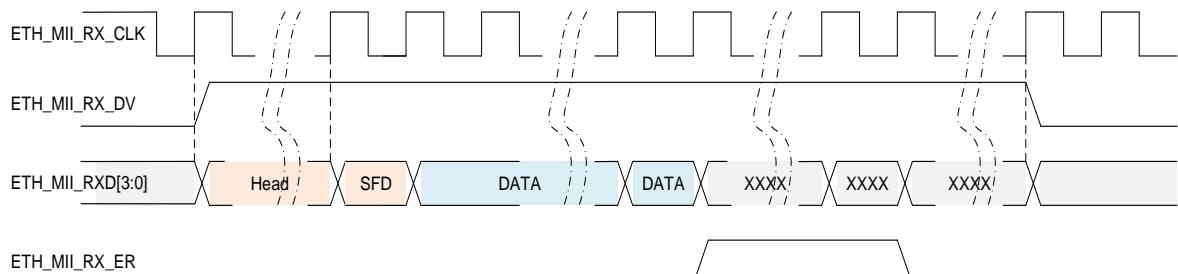


图 24-7 有错误时接收时序图

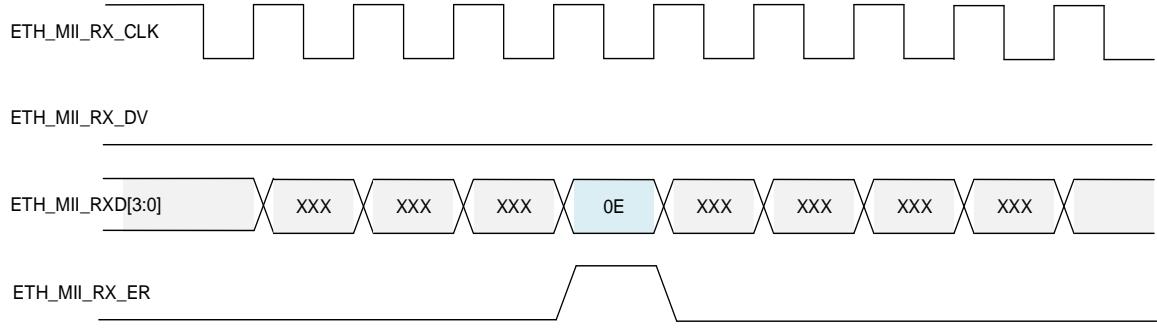


图 24-8 假载波指示接收图

### 24.3.3 MAC 子控制器发送功能

TxDMA 控制发送路径的所有事务。从系统存储器读取的以太网帧由 TxDMA 送入 TxFIFO，然后传输至 MAC 子控制器。帧传输结束时，从 MAC 子控制器获取发送状态并传回 TxDMA。

以太网 MAC 控制器搭载 512Byte 的 Tx FIFO。TxFIFO 填充级别将指示给 DMA，以便 TxDMA 可通过 AHB 接口在所需的系统存储器启动数据传送。来自 AHB Master 接口的数据将送入 Tx FIFO。

检测到 SOF 时，MAC 接收数据并开始向 MII 发送。在应用程序启动发送后，向 MII 发送帧数据所需的时间是可变的，具体取决于 IFG 延迟、发送报头/SFD 的时间以及半双工模式的任意回退延迟等延迟因素。EOF 传输到 MAC 子控制器后，MAC 子控制器将完成正常的发送，然后将发送的状态返回给 TxDMA。如果在发送过程中发生常规冲突（在半双工模式下），MAC 子控制器将使发送状态有效，然后接受并丢弃所有后续数据，直至收到下一个 SOF。检测到来自 MAC 的重试请求（在状态中）时，应从 SOF 重新发送同一帧。

如果发送期间未连续提供数据，MAC 将发出下溢状态。在帧的正常传输期间，如果 MAC 在未获得前一帧的 EOF 的情况下接收到 SOF，则将忽略该 SOF 并将新的帧视为前一帧的延续。

向 MAC 子控制器输送数据有两种操作模式：阈值模式和存储转发模式。

#### 阈值模式

在阈值模式下，只要 FIFO 中的字节数超过配置的阈值（或在超过阈值前写入帧结束），数据就准备好并转发到 MAC 子控制器。该阈值可使用 ETH\_DMAMDR 的 TTC 位配置。

#### 存储转发模式

在存储转发模式下，仅当在 FIFO 中存储完整的帧后，才会向 MAC 内核输送帧。如果 Tx FIFO 的大小小于要发送的以太网帧，则在 Tx FIFO 接近填满时向 MAC 内核输送帧。

应用可通过将 FTF 位（ETH\_DMAMDR 寄存器[20]）置 1 来清空 Tx FIFO 的所有内容。此位自行清零，并将 FIFO 指针初始化为默认状态。如果向 MAC 内核传输帧时将该位置 1，则传输将停止，此时 Tx FIFO 被视为空，MAC 发送器将出现下溢事件并且相应的状态字将转发给 TxDMA。

### 24.3.3.1 发送协议

MAC 子控制器控制以太网帧的发送操作。它执行下列功能以满足 IEEE802.3/802.3z 规范。包括：

- 1) 生成报头和SFD
- 2) 在半双工模式下生成Jam阻塞信号
- 3) 控制Jabber超时

---

#### 4) 控制半双工模式下的流量（背压）

#### 5) 生成发送帧状态

当请求发送新的帧时，MAC 将发送报头和 SFD，紧接着发送数据。报头定义为“10101010”样式的 7 个字节，SFD 定义为“10101011”样式的 1 个字节。

冲突窗口定义为 1 个时隙（对于 10/100 Mbps 以太网，为 512 个位时间）。阻塞信号生成仅适用于半双工模式，不适用于全双工模式。在 MII 模式下，如果在开始传输帧到 CRC 字段结束之间的任何时间发生冲突，MAC 将在 MII 上发送 0x55555555 的 32 位 Jam 阻塞信号，通知所有其它站已发生冲突。如果在报头发送阶段发生冲突，MAC 将完成报头和 SFD 的发送，然后发送 Jam 阻塞信号。

MAC 子控制器内使用一个 Jabber 定时器，用于在传输的字节超过 2048 字节时切断以太网帧的发送。在半双工模式下，MAC 使用延迟机制进行流量控制（背压）。当上层应用请求停止接收帧时，如果已使能发送流量控制，则只要 MAC 检测到接收帧，就会发送一个 32 字节的 Jam 信号。这会导致冲突并使远程站回退。应用程序通过将 ETH\_MACFCR 寄存器中的 FCBBPA 位置 1 来请求流量控制。如果应用请求发送帧，则即使激活背压功能，也将按调度计划发送。如果背压功能长时间保持激活（发生的连续冲突事件超过 16 个），则远程站将由于冲突过多而中止发送。

### 24.3.3.2 调度算法

MAC 子控制器负责调度 MII 上的帧发送。通过 IFG 设定或截断二进制指数回退算法来保持两个发送帧之间的帧间隔。在满足 IFG 和回退条件后，MAC 使能发送。

IFG 设定可确保两个发送帧之间的空闲时段，即配置的帧间隔（ETH\_MACCR 寄存器中的 IFG 位）。如果要发送的帧在配置的 IFG 时间之前到达，则 MII 会等待来自 MAC 的使能信号，然后再开始发送。只要 MII 的载波信号进入无效状态，MAC 就会启动 IFG 计数器。在 IFG 计数值与设定值相等时，MAC 将以全双工模式使能发送。

在半双工模式下，当 IFG 配置为 96 个位时间时，MAC 将遵循 IEEE802.3 规范指定的顺从规则。如果在 IFG 间隔的前三分之二时间内（对于所有 IFG 值都为 64 位时间）检测到载波，MAC 将复位其 IFG 计数器。如果在 IFG 间隔的后三分之一时间内检测到载波，MAC 将继续执行 IFG 计数并在 IFG 间隔结束后使能发送器。

在半双工模式下，实施截断二进制指数回退算法，具体 ETH\_MACCR 寄存器的 BL 位设定控制。

### 24.3.3.3 冲突重发

在半双工模式下，向 MAC 子控制器传输帧时，可能在 MAC 接口上发生冲突事件。MAC 子控制器甚至会在接收到帧结束状态之前就指示重试，然后将使能重新发送并再次将帧从 TxFIFO 中送出。

当超过 96 个字节送到 MAC 子控制器后，TxFIFO 控制器将释放该空间，使 DMA 可送入更多数据。这意味着超过阈值后或 MAC 子控制器指示延迟冲突事件时，无法重新发送。

### 24.3.3.4 TxFIFO 刷新

MAC 子控制器可通过操作模式寄存器（ETH\_DMAMDR）中的 FTF 位来清空 TxFIFO。清空操作是立即操作，即使 TxFIFO 正在向 MAC 内核传输帧，TxFIFO 和相应的指针也会清零到初始状态。这将导致 MAC 发送器中生成下溢事件，并且帧发送将中止，此时帧的状态将同时标记下溢和清空事件（TDES0 的 UDE 位和 FFF 位）。清空操作期间，没有数据从 TxDMA 传输到 TxFIFO，TxDMA 根据清空的帧数（包括局部帧），将相应数量的传输发送状态字传输到应用。完全清空的帧的帧清空状态位（TDES0 的 FFF 位）将置 1。当 TxDMA 接受所有已清空的帧的状态字后，清空操作完成，FTF 位随后将清零。此时，将接受来自 TxDMA 的新帧。清空操作完成后，将丢弃所有提交的数据，除非数据以 SOF 标记开头。

### 24.3.3.5 CRC&PAD 处理

当从应用接收的字节数少于 60 (DA+SA+LT+数据) 时, 可通过发送描述符 TDES0 的 DPAD 位复位, 向发送帧附加零 (PAD), 使数据长度正好为 46 字节, 以满足 IEEE802.3 的最小数据字段要求, 也可选择不附加任何填充值。

MAC 子控制器还会计算帧检查序列 (FCS) 字段的循环冗余校验 (CRC), 根据发送描述符 TDES0 的 DCRC 位和 CRCR 位的设定可选择将校验值插入或替换到正在发送的数据中, 具体请参考常规型 Tx 描述符。

如果将 MAC 编程为不将 CRC 值附加到以太网帧的末尾, 则不发送计算出的 CRC。但当 TDES0 的 DPAD 位为 0 时, 小于 60 字节 (DA+SA+LT+数据) 的帧会在数据后附加填充 PAD, CRC 结果也将自动附加在填充帧的末尾。

### 24.3.3.6 状态字

在向 MAC 子控制器传输以太网帧结束时, 以及子控制器完成帧的发送后, 发送状态会通过发送描述符提供给应用。发送状态的详细说明参见 TDES0。

### 24.3.3.7 暂停帧

在全双工模式下, 当发送流量控制使能位 (ETH\_MACFCR 中的 FTE 位) 置 1 时, MAC 子控制器将生成暂停帧并根据需要发送暂停帧。暂停帧与计算出的 CRC 附加在一起一并发送。可以通过两种方式启动暂停帧的生成。

通过将 ETH\_MACFCR 寄存器中的 FCBBPA 位置 1 来请求流量控制, MAC 子控制器将生成并发送单个暂停帧。生成的帧中的暂停时间值为 ETH\_MACFCR 中编程的暂停时间值。要在先前发送的暂停帧中指定的时间之前延长或结束暂停时间, 用户必须在应用程序中以适当的值编程 ETH\_MACFCR 寄存器中的 PSET 暂停时间值, 然后请求另一次暂停帧发送。

RxFIFO 填满时并且应用程序已请求流量控制, 则 MAC 将生成并发送暂停帧。生成的帧中的暂停时间值为 ETH\_MACFCR 中编程的暂停时间值。如果在此暂停时间结束前, RxFIFO 在可配置的时隙数 (ETH\_MACFCR 中的 PLT 位) 期间保持填满状态, 将发送第二个暂停帧。只要 RxFIFO 保持填满状态, 该过程将一直重复下去。如果在暂停时间结束之前不再满足此条件, MAC 子控制器将发送一个暂停时间为零的暂停帧, 向远程端口表明 RxFIFO 已准备好接收新数据帧。

### 24.3.3.8 COE 引擎

在网络互联过程中, 基本的通信协议 (例如 TCP 和 UDP) 都实施校验和, 并配备校验和字段, 这有助于确定通过网络发送的数据的完整性。由于以太网最广泛的用途是通过 IP 数据报封装 TCP 和 UDP, 因此以太网 MAC 子控制器具有校验和减荷功能 (COE 引擎), 该功能支持发送路径中的校验和计算、插入以及接收路径中的校验和计算、检测。

COE 引擎通过完整的帧来计算 TCP、UDP 或 ICMP 的校验和, 然后将其插入报头的校验和字段。由于此要求, 仅当 TxFIFO 配置为存储转发模式 (ETH\_DMAMDR 寄存器中的 TSF 位置 1) 时, 才使能此功能。如果内核配置为阈值模式, 则将绕过 COE 引擎。

必须确保 TxFIFO 足够深, 使其能存储一个完整的帧, 以便将该帧传输到 MAC 子控制器发送器。如果 FIFO 的深度小于输入以太网帧的大小, 则将绕过 TCP/UDP/ICMP 校验和插入功能, 仅支持 IPv4 报头校验和插入功能, 即使在存储转发模式下也是如此。

发送校验和减荷功能的控制可通过将 CIC 位 (TDES1 中的[28:27]) 设定来实现每个帧的校验和控制。

。

---

- IP报头校验和

在 IPv4 数据报中，报头字段的完整性由 16 位头校验和字段（IPv4 数据报的第 11 字节和第 12 字节）指示。当以太网帧的类型字段值为 0x0800 且 IP 数据报的版本字段值为 0x4 时，COE 引擎将检测到 IPv4 数据报。计算期间，将忽略输入帧的校验和字段并将其替换为计算出的值。IPv6 报头没有校验和字段，因此，COE 不修改 IPv6 报头字段。IPv4 报头校验和计算的结果由发送状态中的 IP 报头错误状态位 (TDES0 的 IHE 位) 指示。只要以太网类型字段的值和 IP 报头版本字段的值不一致，或当以太网帧没有足够的数据 (如 IP 报头长度字段所指示) 时，此状态位将置 1。

因此，总结下来，在以下情况下会发生 IP 报头错误：

- 1) 对于IPv4数据报：
  - a) 以太网类型为0x0800，但IP报头版本字段不等于0x4
  - b) IPv4报头长度字段指示小于0x5 (20字节) 的值
  - c) 总的帧长度小于IPv4报头长度字段给定的值
- 2) 对于IPv6数据报：
  - a) 以太网类型为0x86DD，但IP报头版本字段不等于0x6
  - b) 帧在IPv6报头 (40字节) 之前结束，或已完全接收到扩展报头 (如扩展报头中相应的报头长度字段中给定)

如果以太网类型字段指示为 IPv4 有效负载，即使 COE 引擎测到 IP 报头错误，也会插入 IPv4 报头校验和。

- TCP/UDP/ICMP校验和

TCP/UDP/ICMP 校验和功能会对 IPv4 或 IPv6 报头 (包括扩展报头) 进行处理，并确定封装的有效负载是 TCP、UDP 还是 ICMP。

在下面两种情况下，COE 引擎会无效：

- 1) 非TCP、非UDP或非ICMP/ICMPv6的有效负载。
- 2) 分段的IP帧 (IPv4或IPv6)、带安全功能 (例如，验证报头或封装的安全有效负载) 的IP帧和带路由报头的IPv6帧。

在 COE 引擎有效的情况下，它会计算 TCP、UDP 或 ICMP 有效负载的校验和，然后将其插入报头中相应的字段。它可工作在以下两种方式：

- 1) TCP、UDP或ICMPv6伪报头并未包含在校验和计算中，并假定其存在于输入帧的校验和字段中。校验和字段包含在校验和计算中，然后替换为最终计算出的校验和。
- 2) TCP、UDP或ICMPv6伪报头包含在校验和计算中，将忽略校验和字段，并使用最终计算出的值覆盖校验和字段。

注：对于 ICMP-over-IPv4 数据包，由于没有为其定义伪报头，因此在这两种模式下 ICMP 数据包中的校验和字段都必须始终为 0x0000。如果不等于 0x0000，可能向数据包插入不正确的校验和。

此操作的结果由发送描述符中的有效负载错误状态位 (TDES0 的 TPCE 位) 指示。当检测到下列情况之一时，TPCE 位置 1：

- 1) 在存储转发模式下，帧已转发到MAC发送器，但帧结束未写入Rx FIFO
- 2) 在接收到IP报头中的有效负载长度字段指示的字节数前，数据包已结束

注：当数据包的长度大于指示的有效负载长度时，将字节作为填充字节忽略，并且不报告错误。检测到第一种类型的错误时，不修改 TCP、UDP 或 ICMP 报头。对于第二种错误类型，计算的校验和仍将插入相应

的报头字段。

### 24.3.3.9 发送时序

在 RMII 接口时，来自 MII 的每个半字节都在 RMII 上发送，一次发送双位，双位的发送顺序如下图所示。首先发送位序较低的位 (D0 和 D1)，再发送位序较高的位 (D2 和 D3)。

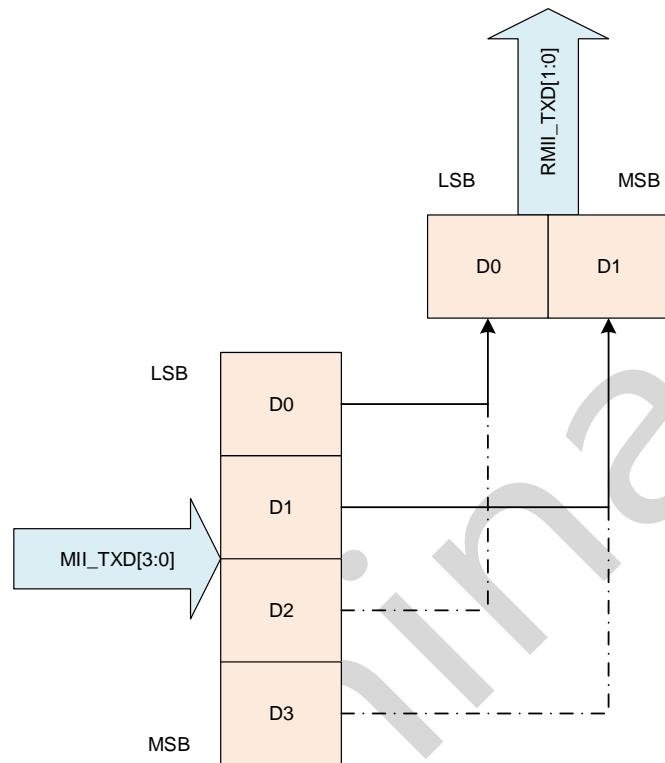


图 24-9 MII/RMII 发送时序

下面所示为基本的接收时序图。

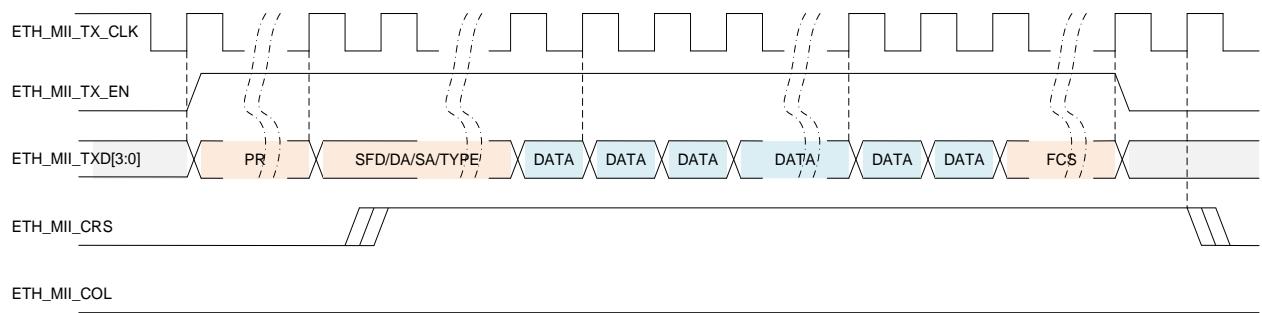


图 24-10 无冲突时发送时序图

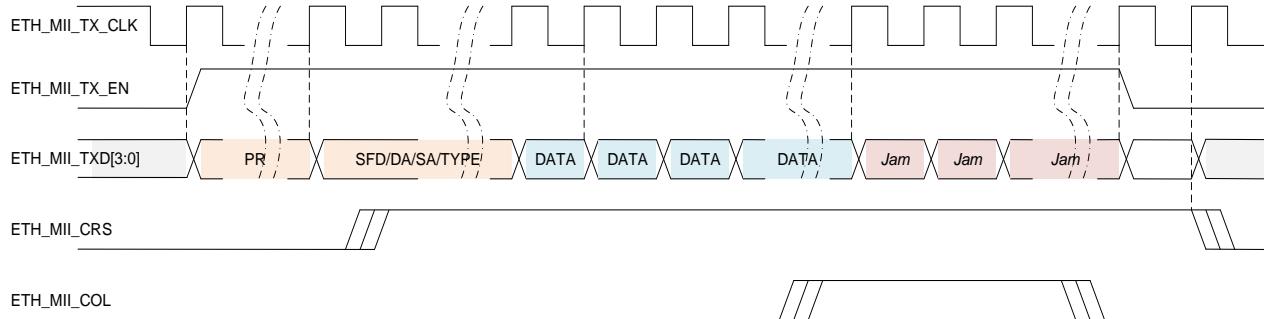


图 24-11 有冲突时发送时序图

## 24.3.4 MAC 子控制器地址过滤

### 24.3.4.1 完美地址过滤

#### 源地址过滤

MAC 子控制器可根据接收的帧的源地址字段来执行完美过滤。默认情况下，MAC 将 SA 字段与 MAC 地址寄存器中编程的值进行比较。可通过将 MAC 过滤地址寄存器中的位 30 (SA) 置 1，来将 MAC 过滤地址寄存器 1~31 配置为 SA 字段进行比较。如果地址过滤控制寄存器 (ETH\_MACAFR) 中的 SAF 位置 1，则 MAC 子控制器将丢弃未通过 SA 过滤的帧。否则，SA 过滤的结果将通过接收状态字中的状态位给出（参见接收描述符字 RDES0）。

**注：**SAF 位置 1 时，对 SA 过滤和 DA 过滤的结果进行与运算，以决定是否需要转发帧。这意味着任何一个过滤未通过都将丢弃帧。两个过滤必须都通过，才能将帧转发到应用。

MAC 子控制器还可以将 MAC 过滤地址寄存器 1~31 中地址值的各个字节与接收的相应 SA 字节进行比较时，可以将寄存器中相应的屏蔽字节控制位置 1 来屏蔽该字节，从而实现 SA 的组地址过滤。

对于源地址过滤，可在最终输出时选择反转过滤匹配结果，即在 SA 地址比较匹配时认定为过滤失败。该功能由地址过滤控制寄存器 (ETH\_MACAFR) 的 SAIF 位控制。

#### 目标地址过滤-单播

MAC 支持多达 32 个用于单播过滤的 MAC 目标地址。如果选择完美过滤（地址过滤控制寄存器中的 HU 位设为 0），MAC 将接收的单播地址的所有 48 位目标地址与编程的 MAC 过滤地址进行比较来确定是否匹配。默认情况下，始终使能 MAC 来过滤地址寄存器 0，其它 MAC 过滤地址寄存器 1~31 则通过单独的使能位进行选择。

MAC 子控制器还可以将 MAC 过滤地址寄存器 1~31 中地址值的各个字节与接收的相应 DA 字节进行比较时，可以将寄存器中相应的屏蔽字节控制位置 1 来屏蔽该字节，从而实现 DA 的组地址过滤。

#### 目标地址过滤-多播

通过将地址过滤控制寄存器中的 PMF 位置 1，将 MAC 子控制器编程为通过所有多播帧。如果 PMF 位复位，MAC 子控制器将根据帧过滤寄存器中的 HM 位执行对多播地址的过滤。在完美过滤模式下，将多播地址与编程的 MAC 过滤地址寄存器 1~31 进行比较。该过滤方式也支持组地址过滤。

#### 目标地址过滤-广播

在默认模式下，MAC 子控制器不过滤任何广播帧。但是，如果将地址过滤控制寄存器中的 DBF 位置 1 来将 MAC 编程为拒绝所有广播帧，则会丢弃任何广播帧。

---

对于目标地址过滤（单播、多播、广播），也可在最终输出时选择反转过滤匹配结果，即在 DA 地址比较匹配时认定为过滤失败。该功能由地址过滤控制寄存器（ETH\_MACAFR）的 DAIF 位控制，同样适用于 Hash 过滤结果。

#### 24.3.4.2 HASH 地址过滤

对于单播目标地址和多播目标地址的过滤也支持 Hash 过滤方式。在 Hash 过滤模式（HU 位/HM 位置 1）下，MAC 将使用 64 位 Hash 表执行对单播地址或多播地址的不完美过滤，即 Hash 过滤。对于 Hash 过滤，MAC 将使用接收的目标地址的 6 个高 CRC 位来索引 Hash 表的内容。值为 000000 时，选取所选寄存器中的位 0；值为 111111 时，选取散列表寄存器中的位 63。如果相应位（由 6 位 CRC 指示）已置 1，将认为单播帧或多播帧已通过 Hash 过滤，否则认为帧未能通过 Hash 过滤。

注：CRC 是使用下列多项式编码的 32 位值：

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

#### 24.3.4.3 VLAN 标识过滤

当接收帧为 VLAN 帧时，MAC 子控制器可通过设定地址过滤控制寄存器（ETH\_MACAFR）的 VTFE 位来使能对 VLAN 标识符字段的过滤。过滤用比较值设置在 MAC VLAN 标签寄存器（ETH\_MACVLTR）的 VLANTAG 位。

### 24.3.5 MAC 子控制器回环模式

MAC 子控制器支持对发送到其接收器的帧进行回送。默认情况下，禁止 MAC 回送功能，可通过编程 MAC 配置寄存器（ETH\_MACCR）中的 LM 位使能该功能。

### 24.3.6 DMA 子控制器基本特征

ETHERMAC 的 DMA 子控制器具有独立的发送引擎（TxDMA）和接收引擎（RxDMA）。发送引擎将数据从系统存储器传送到 TxFIFO，而接收引擎将数据从 RxFIFO 传送到系统存储器。DMA 可以在 CPU 完全不干预的情况下，通过描述符有效地将数据在 FIFO 和系统存储器之间传送。该 DMA 控制器可经过应用编程，在完成帧发送和接收操作时以及其它正常/错误条件下产生 CPU 中断。因此可以看出，ETHERMAC 的 DMA 和 MCU 内核之间通过两种数据结构进行通信：

- 1) 控制和状态寄存器
- 2) 描述符列表和数据缓冲区

DMA 子控制器既可将 MAC 接收到的数据帧传送系统的数据缓冲区，也可以将系统数据缓冲区中的数据发送给 MAC，描述符中包含指向这些数据缓冲区的指针。DMA 子控制器共有两个描述符列表：一个用于接收的 Rx 描述符列表，一个用于发送的 Tx 描述符列表。Rx 描述符列表和 Tx 描述符列表的地址分别通过描述符地址寄存器（ETH\_DMARXDSAR 和 ETH\_DMATXDSAR）设定。该描述符列表是一种前向链表（无论是隐式还是显式），最后一个描述符会指向第一个描述符以构成环形结构。描述符列表位于系统的物理存储空间。

每个描述符最多可指向两个缓冲区，这样便能使用两个物理寻址的缓冲区替代存储器中两个连续的缓冲区。通过配置 Rx 描述符和 Tx 描述符（RDES1[24]和 TDES0[24]）中链接的第二个地址来完成描述符的显式链接。

数据缓冲区同样位于系统的物理存储空间，通常由整个帧或部分帧组成，但不会超过单个帧。数据缓冲区中仅包含数据，数据通信的状态保存在描述符中。检测到帧结束时，DMA 子控制器会跳到下一个帧缓冲区，数据链接可实现跨越多个数据缓冲区存储帧，可以使能或禁止数据链接。描述符有两种结构方式：环形结构与链接结构，其实现方式如下图所示。

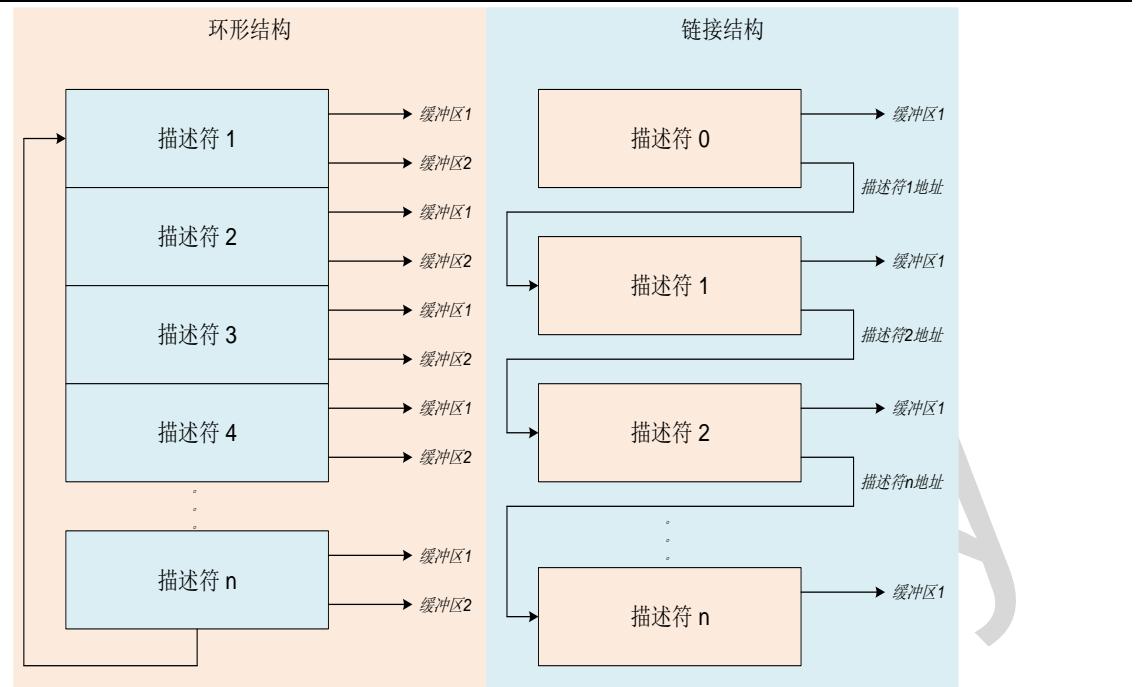


图 24-12 描述符结构

#### 24.3.6.1 突发访问

DMA 子控制器会尝试在 AHB Master 接口上执行固定长度的突发传送（当 ETH\_DMABSR 中的 FBST 位进行相应配置时）。突发的最大长度由 ETH\_DMABSR 的 PBL 指示和限制。对于要读取的 16 个字节，接收和发送描述符始终采用可能的最大突发大小进行访问。

仅当 TxFIFO 中的空间足以容纳配置的突发或帧结束之前的字节数时（当帧短于配置的突发长度时），TxDMA 才会启动数据传送。TxDMA 会向 AHB Master 接口指示起始地址和所需的传送次数。当 AHB 接口配置为固定长度突发时，将使用 INCR4、INCR8、INCR16 和 SINGLE 的最佳搭配来传送数据。否则（非固定长度突发）会使用 INCR（未定义长度）与 SINGLE 传送数据。

仅当 RxFIFO 中有足够的数据用于配置的突发时，或者在 RxFIFO 中检测到帧结束时（当帧短于配置的突发长度时），RxDMA 才会启动数据传送。RxDMA 会向 AHB Master 接口指示起始地址和所需的传送次数。当 AHB 接口配置为固定长度突发时，将使用 INCR4、INCR8、INCR16 和 SINGLE 的最佳搭配来传送数据，如果在 AHB 接口上的固定突发结束前已到达帧结束，将执行空传送以完成固定长度的突发传送。否则（非固定长度突发）会使用 INCR（未定义长度）和 SINGLE 传送数据。

当 AHB 接口配置为地址对齐的节拍 (ETH\_DMABSR.AAL=1) 时，两个 DMA 引擎会确保 AHB 启动的第一次突发传送小于或者等于已配置 PBL 的大小。这样，后续的所有节拍都会从与已配置 PBL 对齐的地址开始。由于 AHB 接口不支持多于 INCR16 的传送，DMA 只能对齐节拍最大为 16 (PBL>16) 的地址。

#### 24.3.6.2 对齐访问

发送和接收数据缓冲区在起始地址对齐方面没有任何限制。缓冲区的起始地址可与四个字节中的任意一个对齐。但是，DMA 子控制器始终在地址与总线宽度对齐时启动传输，并且在不需要的字节通道上传输空数据。这通常发生在以太网帧的开始或结束传送期间。

##### 缓冲区读操作示例：

如果发送缓冲区地址为 0x00000FF2，并且需要传送 15 个字节，则 TxDMA 将从地址 0x00000FF0

---

读取 5 个全字，但在将数据传送到 TxFIFO 时，丢弃或忽略额外的字节（前两个字节）。同样地，还将忽略最后一次传送的最后 3 个字节。TxDMA 始终确保向 TxFIFO 传送的是全 32 位数据，除非是帧结束。

#### 缓冲区写操作示例：

如果接收缓冲区地址为 0x00000FF2，并且需要传送接收到的帧的 16 个字节，则 RxDMA 将从地址 0x00000FF0 开始写入 5 个全 32 位数据。但是，在第一次传送的前 2 个字节与第三次传送的最后 2 个字节将包含空数据。

### 24.3.6.3 缓冲器计算

DMA 子控制器不会更新发送和接收描述符中的缓冲区大小字段，只更新描述符的状态字段。应用必须用驱动程序计算缓冲区的大小。

TxDMA 会向 MAC 内核传送准确的字节数（由 TDES1 中的缓冲区大小字段指示）。如果将描述符标记为第一个描述符（将 TDES0 中的 TFS 位置 1），则 DMA 会将缓冲区的第一次传送标记为帧起始。如果将描述符标记为最后一个描述符（将 TDES0 中的 TLS 位置 1），则 DMA 会将数据缓冲区的最后一次传送标记为帧结束。

RxDMA 持续将数据传送至缓冲区，直到缓冲区已满或接收到帧结束为止。当描述符的 RFS 位置 1 时，如果未将描述符标记为最后一个描述符（RDES0 中的 RLS 位），则该描述符对应的缓冲区会填满，有效数据量由 RDES1 中的缓冲区大小指示；如果将描述符标记为最后一个描述符，则缓冲区不会填满，有效数据量将通过缓冲区大小字段减去数据缓冲指针偏移（RDES0 中的 FL 位）得到的结果表示，当数据缓冲区指针与数据总线宽度对齐时，偏移为零。

注：即使当接收缓冲区的起始地址与系统数据总线宽度未对齐时，系统也应分配一个大小与系统总线宽度对齐的接收缓冲区。例如，如果系统分配一个起始地址为 0x1000、大小为 1024 字节 (1KB) 的接收缓冲区，则软件可将接收描述符中的缓冲区起始地址编程为具有 0x1002 偏移。RxDMA 将帧写入该缓冲区，其中前两个单元 (0x1000 和 0x1001) 中为空数据。实际帧从单元 0x1002 开始写入。因此，尽管已将缓冲区大小编程为 1024 字节，但由于存在起始偏移地址，因此该缓冲区的实际有用空间为 1022 字节。

### 24.3.6.4 DMA 仲裁

DMA 子控制器内的仲裁器会在发送和接收通道对 AHB Master 接口进行的访问之间进行仲裁。可以使用两类仲裁：循环优先级和固定优先级。如果选择循环优先级仲裁（ETH\_DMABSR.DMAA=0），仲裁器会在发送和接收 DMA 同时请求访问时，按照 ETH\_DMABSR.FTPR 位设置的优先比率分配数据总线；如果选择固定优先仲裁（ETH\_DMABSR.DMAA=1），仲裁器会设定接收 DMA 总是比发送 DMA 对总线拥有更高的访问优先级线。

### 24.3.6.5 错误响应

对于由 DMA 子控制器发起的任何数据传送，如果从机给出错误响应，则相应 DMA 将停止所有操作并更新状态寄存器（ETH\_DMASR）中的错误位和致命总线错误位。此时，该 DMA 子控制器只能在软复位或硬复位外设以及重新初始化 DMA 之后才能恢复操作。

## 24.3.7 DMA 子控制器接收功能

### 24.3.7.1 DMA 操作

RxDMA 引擎的操作顺序如下：

- 1) 用户设置接收描述符（RDES0-RDES3）并将 OWN 位（RDES0[31]）置 1

- 
- 2) 操作模式寄存器ETH\_DMAMDR.STR位置1后，RxDMA进入运行状态。在运行状态下，RxDMA轮询接收描述符列表，尝试获取空闲描述符。如果获得的描述符不空闲（由CPU所拥有），则RxDMA进入挂起状态并跳转到步骤9
  - 3) RxDMA将所获取描述符中的接收数据缓冲区地址解码
  - 4) 处理传入帧并将其放入所获取描述符的数据缓冲区
  - 5) 当缓冲区已满或帧传输完成时，RxDMA引擎将获取下一个描述符
  - 6) 如果当前的帧传输完成，RxDMA将继续执行第7步。如果RxDMA未拥有下一个接收描述符且帧传输尚未完成（尚未传输EOF字段），则RxDMA会将RDES0中的描述符错误位DPE置1（除非禁止刷新）。RxDMA关闭当前描述符（将OWN位清零）并通过将最后一个描述符位（RDES0.RLS）清零来将其标记为中间描述符（如果禁止刷新，则将其标记为最后一个描述符），随后继续执行第8步；如果RxDMA已拥有下一个描述符，但当前的帧传输尚未完成，则RxDMA将关闭当前描述符作为中间值并返回第4步
  - 7) RxDMA随后获取所接收帧的状态并将该状态字写入当前描述符的RDES0，同时OWN位清零且最后一个描述符位置1
  - 8) RxDMA引擎检查最新描述符的OWN位。如果CPU拥有该描述符（OWN位为0），则接收缓冲区不可用位（ETH\_DMASR.RUS）置1且RxDMA引擎进入挂起状态（第9步）；如果RxDMA拥有该描述符，引擎将返回第4步并等待下一个帧
  - 9) 在RxDMA引擎进入挂起状态前，将从Rx FIFO中刷新部分帧（可使用ETH\_DMAMDR寄存器的DFRF位控制刷新）
  - 10) 当收到接收轮询要求命令或者可以从Rx FIFO获得下一个帧的起点时，RxDMA将退出挂起状态。引擎继续执行第2步并重新获取下一个描述符

#### 24.3.7.2 帧获取处理

RxDMA引擎始终会尝试为即将传入的帧获取一个额外的描述符。只要满足以下任一条件，即尝试获取描述符：

- 1) RxDMA进入运行状态后，接收启动/停止位（ETH\_DMAMDR.STR）立即置1
- 2) 在当前传输的帧结束前，当前描述符的数据缓冲区已满
- 3) MAC子控制器已完成数据帧接收，但当前的接收描述符尚未关闭
- 4) 接收过程因描述符由CPU所拥有（RDES0.OWN=0）而挂起，并且接收到新帧
- 5) 已发出接收轮询要求命令

当帧通过地址过滤且其大小大于或等于为 Rx FIFO 所设置的可配置阈值字节数时，或者在存储转发模式下将整个帧写入 Rx FIFO 时，RxDMA 才会将接收的帧传输到系统存储器；当帧未通过地址过滤时，将丢弃该帧（除非接收所有位 ETH\_MACAFR.RALL 置 1）。不足 64 字节的帧由于会发生冲突或过早结束，因而可从 Rx FIFO 中清除。接收 64（可配置阈值）个字节后，RxDMA 开始将帧数据传输到当前描述符所指定的接收缓冲区中。DMAAHB 接口准备好接收数据传输后，如果 RxDMA 当前未从存储器获取发送数据，则将第一个描述符位（RDES0.RFS）置 1，以分隔该帧。数据缓冲区已满或者帧的末段已传输到接收缓冲区时，OWN（RDES0）位将复位为 0，此时释放描述符。如果帧包含在单个描述符中，则最后一个描述符位（RDES0.RLS）和第一个描述符位（RDES0.RFS）均置 1。

RxDMA 获取下一个帧的描述符时，将最后一个描述符位（RDES0.RLS）置 1，并释放前一个帧描述符中的状态位，然后将接收中断位（ETH\_DMASR 寄存器 RIS）置 1。该过程将重复执行，直至 RxDMA 遇到被标记为由 CPU 所有的描述符。这种情况时，接收过程将 ETH\_DMASR 寄存器中接收缓冲区不可用位 RUS 置 1，随后进入挂起状态，但其在接收列表中的位置仍然保留。

### 24.3.7.3 过程挂起

如果在接收过程处于挂起状态时有新的接收帧到达，则 RxDMA 将重新获取系统存储器中的当前描述符。如果该描述符现在由 RxDMA 所拥有，接收过程将重新进入运行状态并开始接收帧；如果该描述符仍由主机所拥有，则默认情况下，RxDMA 将丢弃 RxFIFO 顶部的当前帧并将丢失帧计数器递增。如果 RxFIFO 中存储了多个帧，将重复执行该过程。

若将 DMA 工作模式寄存器 (ETH\_DMAMDR) 的 DFRF 位置 1 后，可避免丢弃或刷新 RxFIFO 顶部的帧。在这种情况下，接收过程将动作状态寄存器 (ETH\_DMASR) 的接收缓冲区不可用位 RUS 置 1 并返回到挂起状态。

## 24.3.8 DMA 子控制器发送功能

### 24.3.8.1 DMA 操作

在连续帧传送过程中，TxDMA 根据操作模式寄存器 ETH\_DMAMDR 的 OSF 位的设定不同，对应的第二帧处理时序不同。当 OSF 位置 1 时，发送过程完成第一个帧的传送后，会立即轮询第二个帧的发送描述符列表，如果第二个帧有效，发送过程会在写入第一个帧的状态信息前发送第二帧。这种发送过程无需关闭第一个帧的状态描述符便可同时获取两个帧的方式称之为 OSF 模式，反之为非 OSF 模式。

在 ETH\_DMAMDR.OSF=0 下的操作顺序如下：

- 1) 在数据缓冲区设置好相应的以太网帧数据后，用户配置发送描述符 (TDES0-TDES3) 并将 OWN 位 (TDES0[31]) 置 1
- 2) STT 位 (ETH\_DMAMDR 寄存器[13]) 置 1 后，TxDMA 立即进入运行状态
- 3) 在运行状态下，TxDMA 为需要传送的帧轮询发送描述符列表。轮询启动后，TxDMA 会以连续的描述符环形顺序或链接顺序持续进行。如果 TxDMA 检测到标记为 CPU 所有的描述符 (TDES0.OWN=0)，或者发生错误条件，则会挂起传送并将发送缓冲区不可用位 (ETH\_DMASR 寄存器 TUS 位) 与正常中断汇总状态位 (ETH\_DMASR 寄存器 AIS 位) 置 1。发送引擎继续执行步骤 9
- 4) 如果获得的描述符标记为 TxDMA 所有 (TDES0.OWN=1)，则 TxDMA 将根据取得的描述符对发送数据缓冲区地址进行解码
- 5) TxDMA 从系统存储器中获取发送数据并传送这些数据
- 6) 如果以太网帧保存在多个描述符的数据缓冲区中，则 TxDMA 将关闭中间描述符并获取下一个描述符。重复执行步骤 3、4 和 5，直到完成以太网帧数据的传送
- 7) 完成帧传送后，状态信息将写入该发送描述符的各状态位 (TDES0)。此步骤中会清零 OWN 位，此时该描述符变为由 CPU 所有
- 8) 帧发送完成后，在其最后一个描述符位 (TDES1.TLS) 置 1 时，DMA 动作状态寄存器的 TIS 位将置 1。随后 TxDMA 引擎返回步骤 3
- 9) 在挂起状态下，TxDMA 会在接收到发送轮询要求时尝试重新获取描述符 (返回步骤 3)，并将下溢中断状态位清零

在 ETH\_DMAMDR.OSF=1 下的操作顺序如下：

- 1) TxDMA 的操作过程和 OSF=0 模式下 TxDMA 的步骤 1~6 相同
- 2) 无需关闭前一帧的最后一个描述符，TxDMA 便可获取下一个描述符
- 3) 如果 TxDMA 拥有所需描述符，便会对该描述符中的发送缓冲区地址进行解码。如果 TxDMA 未拥有描述符，则会进入挂起模式并跳到步骤 7

- 4) TxDMA从系统存储器中取得发送帧并发送该帧，直到帧数据发送结束，如果该帧拆分到多个描述符中，则会同时关闭中间描述符
  - 5) TxDMA等待前一个帧的发送状态。当状态可用时，TxDMA将状态写入相应TDES0，进而关闭描述符
  - 6) TxDMA获取下一个描述符，然后继续执行步骤3（状态正常时）。如果上一个发送状态显示下溢错误，则TxDMA会进入挂起模式（步骤7）
  - 7) 在挂起模式下，如果TxDMA接收到挂起状态，会将状态写入相应的TDES0。之后，将相关中断置1并返回挂起模式
  - 8) 只有在接收到发送轮询要求（ETH\_DMATXPDR寄存器）后，TxDMA才会退出挂起模式并进入运行状态（转到步骤1或步骤2，具体取决于挂起状态）
- 其整个过程如下图所示：

#### 24.3.8.2 帧获取处理

系统数据缓冲区须包含完整的以太网帧，不包括报头、PAD 和 FCS 字段，DA 字段、SA 字段和类型/长度字段须包含在有效数据中。如果发送描述符指示 MAC 子控制器必须禁止插入 CRC 或填充字节，则缓冲区必须具有包括 CRC 字节的完整以太网帧（不包括报头）。帧可以采用数据链接形式，也可以跨越多个缓冲区。帧必须由第一个描述符位（TDES1.TFS）和最后一个描述符位（TDES1.TLS）定界。传送启动时，必须将 TDES1.TFS 置 1，之后帧数据便从存储器缓冲区传送到 TxFIFO。同时，如果当前帧的 TDES1.TLS 为 0，发送过程将尝试获取下一个描述符。TDES1.TLS 为 0，则指示中间缓冲区，TDES1.TLS 为 1，则指示帧的最后一个缓冲区。当帧的最后一个缓冲区完成传送后，TxDMA 会将最终状态信息写回描述符的（TDES0）相关状态位。此时，如果 TDES0.DIC 位置 1，则发送状态寄存器的 TIS 位将置 1，并获取下一个描述符，然后重复执行该过程。

根据 ETH\_DMAMDR 寄存器 TSF 位的设定，实际的帧传送过程在 TxFIFO 达到可编程的发送阈值时（ETH\_DMAMDR 寄存器 TTC 位），或者 TxFIFO 中包含完整的帧时启动传送过程。TxDMA 完成帧的传送后会释放描述符（清零 TDES0.OWN 位）。

#### 24.3.8.3 过程挂起

可通过以下任意条件暂停发送轮询：

- 1) TxDMA检测到CPU所有的描述符（TDES0[31]=0），并且ETH\_DMASR寄存器的发送缓冲区不可用标志TUS置1。如要恢复，驱动程序必须将描述符的所有权交给TxDMA，然后发出轮询查询要求命令
- 2) 检测到由下溢导致的传送错误时，将中止帧的传送。相应的发送描述符（TDES0）位将置1。

如果 TxDMA 由于第一个条件而进入挂起状态，则发送状态寄存器的正常中断汇总位与发送缓冲区不可用位（ETH\_DMASR 寄存器位 NIS 和 TUS 位）均置 1；如果发生第二个条件，异常中断汇总位与发送下溢位（ETH\_DMASR 寄存器位 AIS 位和 UNS 位）都将置 1，并且信息将写入发送描述符，从而导致挂起。两种情况下，发送列表中的位置都会保留。保留的位置是 TxDMA 关闭的最后一个描述符后面的描述符位置。纠正挂起原因后，驱动程序必须明确发出发送轮询要求命令。

#### 24.3.9 DMA 子控制器初始化

DMA 的初始化步骤如下：

- 1) 对ETH\_DMABSR执行写操作以设置总线访问参数
- 2) 对ETH\_DMAIR寄存器执行写操作以屏蔽不必要的中断源

- 3) 创建描述符列表, 对ETH\_DMARXDSAR和ETH\_DMATXDSAR寄存器执行写操作, 为DMA提供各列表的起始地址
- 4) 对MAC寄存器执行写操作设定所需的过滤选项
- 5) 对ETH\_MACCR寄存器执行写操作以配置和使能发送与接收工作模式
- 6) 对ETH\_DMAMDR寄存器执行写操作, 将STT位和STR位置1以启动发送和接收
- 7) 发送与接收引擎进入运行状态, 并尝试从相应描述符列表中获取描述符, 这两个引擎随后开始处理接收和发送操作。发送和接收处理过程彼此独立, 可单独进行启动或停止

### 24.3.10 DMA 子控制器描述符

#### 24.3.10.1 接收描述符

RxDMA 的接收描述符有 4 个, 由四个 32 位字组成, 如下表所示, 分别定义为 RDES0、RDES1、RDES2 和 RDES3。

<i>RDES0</i>	OWN	状态位 [30:0]				
<i>RDES1</i>	控制位 [31]	Res.	控制位 [25:24]	Res.	字节计数缓冲区 2 [21:11]	字节计数缓冲区 1 [10:0]
<i>RDES2</i>	缓冲地址 1					
<i>RDES3</i>	缓冲地址 2					

下面分别对 RDES0~RDES3 的具体位做说明。

1) RDES0的各位功能如下:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
OWN	DAF	FL[13:0]													
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
ERSM	DPE	SAF	LEE	OVE	VLAT	RFS	RLS	IPCE	LCOE	FRAT	WTE	REE	DBE	CRE	DAS

位	标记	位名	功能
b31	OWN	所有关系位	0: 该描述符为 CPU 所有 1: 该描述符为 DMA 所有 DMA 在帧接收完成或此描述符的关联缓冲区已满时将该位清零
b30	DAF	目标地址过滤失败	该位指示帧未能通过 MAC 内核中的 DA 过滤

位	标记	位名	功能
b29~b16	FL	帧长统计	该位指示传输到主机存储器中的接收帧字节长度（含 CRC） 只有在最后一个描述符位（RDES0.RLS）置 1，并且描述符错误位（RDES0.DPE）和上溢错误位（RDES0.OVE）复位时，该字段才有效 当最后一个描述符位和错误汇总位均未置 1 时，该字段指示为当前帧传输的累计字节数
b15	ERSM	接收错误汇总	该描述符的 B[14]、B[11]、B[7]、B[6]、B[4]、B[3]、B[1]位及 REDS4 描述符的 B[4]、B[3]位中有一个被置位，该位也随之被置位
b14	DPE	描述符错误	该位指示某个帧因超过当前描述符缓冲区的大小而被截断以及 DMA 未拥有下一个描述符 注：该位只有在最后一个描述符（RDES0.RLS）置 1 时才有效
b13	SAF	源地址过滤失败	该位指示帧的 SA 字段未能通过 MAC 内核中的 SA 过滤
b12	LEE	长度错误	该位指示接收帧的实际长度与长度/类型字段的值不符 注 1：该位仅在帧类型位（RDES0.FRAT）复位后有效 注 2：在 CRC 错误位（RDES0.CRE）有效时，该位无效
b11	OVE	上溢错误	该位指示接收帧因缓冲区上溢而损坏
b10	VLAT	VLAN 标识符	该位指示描述符所指向的帧是由 MAC 内核标记的 VLAN 帧
b9	RFS	第一个描述符	该位指示此描述符包含帧的第一个缓冲区。如果第一个缓冲区的大小为 0，则第二个缓冲区将包含帧的帧头；如果第二个缓冲区的大小为 0，则下一个描述符将包含帧的帧头
b8	RLS	最后一个描述符	该位指示此描述符指向的缓冲区为帧的最后一个缓冲区

位	标记	位名	功能
b7	IPCE	COE 错误/巨帧	<p>当 Checksum Offload 功能有效时:</p> <p>该位指示 IPv4 或 IPv6 报头中存在错误 导致此错误的原因可能是:</p> <ol style="list-style-type: none"> <li>1) 以太网类型字段与 IP 报头版本字段值不一致</li> <li>2) IPv4 中报头的校验和不匹配</li> <li>3) 以太网帧缺少所需的 IP 报头字节数</li> </ol> <p>当 Checksum Offload 功能无效时:</p> <p>该位指示 MAC 接收到一个巨帧</p>
b6	LCOE	延迟冲突错误	该位指示在以半双工模式接收帧时发生了延迟冲突
b5	FRAT	帧类型	<p>0: MAC 接收到以太网帧 1: MAC 接收到 PTP 帧</p> <p>注: 当接收到一个矮帧时, 该位无效</p>
b4	WTE	看门狗错误	该位指示接收看门狗计时器在接收当前帧时超时, 且当前帧在看门狗超时后被截断
b3	REE	接收错误	该位指示在帧接收期间, PHY 发出 RX_DV 信号时, 发出了 RX_ER 信号
b2	DBE	Dribble 位错误	<p>该位指示接收的帧具有非整数倍数的字节 (奇数半字节)</p> <p>注: 该位仅在 MII 模式下有效</p>
b1	CRE	CRC 错误	<p>该位指示接收的帧发生循环冗余校验 (CRC) 错误</p> <p>注: 该位只在最后一个描述符 (RDES0.RLS) 置 1 有效</p>

位	标记	位名	功能
b0	DAS	地址过滤成功/状态位扩展	<p>当 COE 功能无效时：</p> <p>0：接收帧通过了 MAC 过滤地址寄存器 0 的 DA 地址过滤</p> <p>1：接收帧通过了 MAC 过滤地址寄存器 1~31 的 DA 地址过滤</p> <p>注：当该描述符的 DAF 位置位时，该位无效</p> <p>当 COE 功能有效时，以下情况该位置 1：</p> <ol style="list-style-type: none"> <li>1) 当内核计算的 16 位 IP 有效负载校验和（即 TCP、UDP 或 ICMP 校验和）与接收帧中对应的校验和字段不匹配</li> <li>2) 当 TCP、UDP 或 ICMP 段长度与 IP 报头字段中的有效负载长度值不匹配</li> </ol>

2) RDES1的各位功能如下：

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
DIC	Res.						RER	RCH	Res.		RBS2[10:5]					
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
RBS2[4:0]						RBS1[10:0]										

位	标记	位名	功能
b31	DIC	完成时禁止中断	<p>该位置 1 时，在接收帧在由该描述符所指示的缓冲区中结束时禁止引发主机的中断，同时也阻止动作状态寄存器 (ETH_DMCSR) 的 RIS 位置 1</p> <p>注：该位仅在最后一个描述符位 (RDES0.RLS) 置 1 的情况下有效</p>
b30~b26	Reserved	-	-
b25	RER	环接收结束	该位指示描述符列表已到达其最后一个描述符，DMA 会返回描述符列表的首地址，形成一个描述符环
b24	RCH	第二个地址链接	<p>该位置 1 时，描述符中的第二个地址是下一个描述符地址，而非第二个缓冲区地址，此时该描述符的 RBS2 位为无效</p> <p>注：RER 位优先级高于 RCH 位</p>
b23~b22	Reserved	-	-
b21~b11	RBS2	缓冲区 2 大小	<p>该位以字节为单位指示第二个数据缓冲区的大小</p> <p>注：该位在该描述符的 RCH 位置 1 时无效</p>

位	标记	位名	功能
b10~b0	RBS1	缓冲区 1 大小	该位以字节为单位指示第一个数据缓冲区的大小。如果该字段为 0, DMA 将忽略该缓冲区并使用缓冲区 2 或下一个描述符, 具体取决于该描述符的 RCH 的设定

3) RDES2的各位功能如下:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RBA1[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RBA1[15:0]															

位	标记	位名	功能
b31~b0	RBA1	接收缓冲区 1 地址	该位向 DMA 指示数据在存储器中的位置, 当软件为 DMA 提供此描述符 (RDES0 中的 OWN 位置 1) 时, 这些位将指示缓冲区 1 的物理地址

4) RDES3的各位功能如下:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
RBA2[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
RBA2[15:0]															

位	标记	位名	功能
b31~b0	RBA2	接收缓冲区 2 地址/下一个描述符地址	该位向 DMA 指示数据在存储器中的位置, 当软件为 DMA 提供此描述符 (RDES0 中的 OWN 位置 1) 并且使用描述符环结构时, 这些位将指示缓冲区 2 的物理地址; 如果 RDES1.RSAC 位置 1, 则该地址包含下一个描述符所在物理寄存器的指针  注: 只有在 RDES1.RSAC 位置 1 时, 缓冲区地址指针才必须与总线宽度相符

#### 24.3.10.2 发送描述符

TxDMA 的发送描述符有 4 个, 由四个 32 位字组成, 如下表所示, 分别定义为 TDES0、TDES1、TDES2 和 TDES3。

<i>TDES0</i>	OWN	Res.								状态位 [16:0]							
<i>TDES1</i>	控制位 [31:23]						Res.	字节计数缓冲区 2 [21:11]				字节计数缓冲区 1 [10:0]					
<i>TDES2</i>	缓冲地址 1																
<i>TDES3</i>	缓冲地址 2																

下面分别对 TDES0~TDES3 的具体位做说明。

### 1) TDES0的各位功能如下：

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
OWN	Res.													IHE		
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
ETSM	JTE	FFF	TPCE	LCAE	NCE	TLCE	ECE	VLF	COC[3:0]					EDE	UDE	DEE

位	标记	位名	功能
b31	OWN	所有关系位	0: 该描述符为 CPU 所有 1: 该描述符为 DMA 所有  DMA 在完成帧发送时或在描述符分配的缓冲区全部读取完成后将该位清零  属于同一个帧的所有后续描述符置 1 后，也应该将帧的第一个描述符的 OWN 置 1
b30~b17	Reserved	-	-
b16	IHE	IP Header 错误	该位置 1 时指示 MAC 发送器在 IP 数据报头中检测到错误  注：当 COE 引擎检测出一个 IP Header 错误时，它仍然会将计算好的 Checksum 值插入到 IPv4 Header Checksum 字段
b15	ETSM	发送错误汇总	该描述符的 B[16]、B[14]、B[13]、B[12]、B[11]、B[10]、B[9]、B[8]、B[2]、B[1]位中有一个被置位，该位也随之被置位
b14	JTE	Jabber 超时错误	该位置 1 时，指示 MAC 发送器经历了 Jabber 超时  注：只有在 MAC 配置寄存器 (ETH_MAC_CONFIGR) 的 MJD 位未置 1 时，该位才会置 1

位	标记	位名	功能
b13	FFF	帧刷新	该位指示 DMA 已依照 CPU 发出的软件刷新命令刷新帧
b12	TPCE	有效负载错误	该位指示 COE 引擎在 TCP、UDP 或 ICMP 报有效负载中检测到错误，且不会更新原帧中的 Checksum 字段
b11	LCAE	载波丢失错误	该位指示帧发送期间丢失载波，即，帧发送期间有一个或多个发送时钟周期的 MII_CRS 信号无效 注：该位仅对在 MAC 处于半双工模式时实现无冲突发送的帧有效
b10	NCE	无载波错误	该位指示在发送期间未由 PHY 触发载波监听信号
b9	TLCE	延迟冲突错误	该位指示帧发送过程因冲突窗口（MII 模式下为 64 个字节时间，含报头）后出现冲突而中止 注：如果该描述符的 UDE 错误位置 1，则该位无效
b8	ECE	过度冲突错误	该位指示尝试发送当前帧时因出现 16 个连续冲突而中止发送 注：如果 MAC 配置寄存器（ETH_MACCR）中的 RETY（禁止重试）位置 1，则此位会在出现首个冲突后置 1 并且帧发送过程将中止
b7	VLF	VLAN 帧	该位指示所发送的帧为 VLAN 帧
b6~b3	COC	冲突计数	该位指示发送帧之前出现的冲突个数 注：过度冲突位（TDES0.ECE）置 1 时，该计数无效
b2	EDE	过度延迟错误	该位指示因出现超过 24288 个位时间的过度延迟而中止发送 注：该位在 MAC 配置寄存器（ETH_MACCR）中的延迟检查（DLYC）位置 1 时有效
b1	UDE	下溢错误	该位指示 MAC 因发送缓冲区的数据未及时到达而中止了帧发送下溢错误表示 DMA 在帧发送期间遇到发送缓冲区为空的情况

位	标记	位名	功能
b0	DEE	延迟错误	该位指示 MAC 在发送前因存在载波而延迟 注：该位仅在半双工模式下有效

2) TDES1的各位功能如下：

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16	
DIC	TLS	TFS	CIC[1:0]		DCRC	TER	TCH	DPAD	-	TBS2[10:5]						
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0	
TBS2[4:0]					TBS1[10:0]											

位	标记	位名	功能
b31	DIC	完成时禁止中断	该位置 1 时，当前帧发送完毕后，DMA 状态寄存器的 TIS 位置 1 注：该位仅在最后一个描述符位 (TDES1.TLS) 置 1 的情况下有效
b30	TLS	最后一个描述符	该位指示此描述符指向的缓冲区为帧的最后一个缓冲区
b29	TFS	第一个描述符	该位指示此描述符包含帧的第一个缓冲区。如果第一个缓冲区的大小为 0，则第二个缓冲区将包含帧的帧头；如果第二个缓冲区的大小为 0，则下一个描述符将包含帧的帧头
b28~27	CIC	Checksum 插入控制	该位控制校验和的计算与插入，如下所示： 00：禁止插入校验和 01：仅使能 IP 报头校验和的计算与插入 10：使能 IP 报头校验和以及 TCP/UCP/ICMP 校验和的计算与插入，但不会在硬件中计算伪报头校验和 11：使能 IP 报头校验和以及 TCP/UDP/ICMP 校验和的计算与插入，并在硬件中计算伪报头校验和 注：该位仅在最后一个描述符位 (TDES1.TFS) 置 1 的情况下有效
b26	DCRC	无效 CRC	该位置 1 时，MAC 不会将循环冗余校验 (CRC) 附加到所发送帧的末尾 注：该位仅在 TDES1.TFS 置 1 的情况下有效

位	标记	位名	功能
b25	TER	环发送结束	该位指示描述符列表已到达其最后一个描述符，DMA 会返回描述符列表的首地址，形成一个描述符环
b24	TCH	第二个地址链接	该位置 1 时，描述符中的第二个地址是下一个描述符地址，而非第二个缓冲区地址，此时该描述符的 TBS2 位为无效 注：TER 位优先级高于 TCH 位
b23	DPAD	无效 DPAD	0: DMA 会自动为不足 64 字节的帧添加补位项和 CRC，不管该描述符的 DCRC 位的设定 1: MAC 不会自动为不足 64 字节的帧添加补位项 注：该位仅在第一个描述符位 (TDES1.TFS) 置 1 的情况下有效
b22	Reserved	-	-
b21~b11	TBS2	缓冲区 2 大小	该位以字节为单位指示第二个数据缓冲区的大小 注：该位在该描述符的 TCH 位置 1 时无效
b10~b0	TBS1	缓冲区 1 大小	该位以字节为单位指示第一个数据缓冲区的大小。如果该字段为 0，DMA 将忽略该缓冲区并使用缓冲区 2 或下一个描述符，具体取决于该描述符的 TCH 的设定

3) TDES2的各位功能如下：

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TBA1[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TBA1[15:0]															

位	标记	位名	功能
b31~b0	TBA1	发送缓冲区 1 地址	该位向 DMA 指示数据在存储器中的位置，当软件为 DMA 提供此描述符 (TDES0 中的 OWN 位置 1) 时，这些位将指示缓冲区 1 的物理地址

4) TDES3的各位功能如下:

b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
TBA2[31:16]															
b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
TBA2[15:0]															

位	标记	位名	功能
b31~b0	TBA2	发送缓冲区 2 地址/下一个描述符地址	<p>该位向 DMA 指示数据在存储器中的位置，当软件为 DMA 提供此描述符 (TDES0 中的 OWN 位置 1) 并且使用描述符环结构时，这些位将指示缓冲区 2 的物理地址；如果 TDES1.TSAC 位置 1，则该地址包含下一个描述符所在物理寄存器的指针</p> <p>注：只有在 TDES1.TSAC 位置 1 时，缓冲区地址指针才必须与总线宽度相符</p>

## 24.4 ETHERMAC 中断

ETHERMAC 的 DMA 子控制器在发送数据和接收数据过程中产生的各种正常事件会触发普通中断、各种异常事件会触发异常中断，两类中断汇总成一个中断输出给 CPU。各具体事件的中断控制可通过 DMA 中断寄存器 (ETH\_DMAIR) 的各个位控制。参见 DMA 中断寄存器章节。

## 24.5 ETHERMAC 寄存器

表 24-2 ETHERMAC 寄存器概览表

Offset	Acronym	Register Name	Reset
0x0010	ETH_MACSMIAR	SMI 地址寄存器	0x00000000
0x0014	ETH_MACSMIDR	SMI 数据寄存器	0x00000000
0x0000	ETH_MACCCR	MAC 配置寄存器	0x00008000
0x0024	ETH_MACSR	MAC 状态寄存器	0x00000000
0x0018	ETH_MACFCR	MAC 流控寄存器	0x00000000
0x0004	ETH_MACAFR	MAC 地址过滤控制寄存器	0x00000000
0x0040	ETH_MACAHR0	MAC 过滤地址高位寄存器 0	0x8000FFFF
0x0044	ETH_MACALR0	MAC 过滤地址低位寄存器 0	0xFFFFFFFF
0x0048	ETH_MACAHR1	MAC 过滤地址高位寄存器 1	0x0000FFFF
0x004C	ETH_MACALR1	MAC 过滤地址低位寄存器 1	0xFFFFFFFF
0x0050	ETH_MACAHR2	MAC 过滤地址高位寄存器 2	0x8000FFFF
0x0054	ETH_MACALR2	MAC 过滤地址低位寄存器 2	0xFFFFFFFF
0x0058	ETH_MACAHR3	MAC 过滤地址高位寄存器 3	0x0000FFFF

Offset	Acronym	Register Name	Reset
0x005C	ETH_MACALR3	MAC 过滤地址低位寄存器 3	0xFFFFFFFF
0x0060	ETH_MACAHR4	MAC 过滤地址高位寄存器 4	0x0000FFFF
0x0064	ETH_MACALR4	MAC 过滤地址低位寄存器 4	0xFFFFFFFF
0x0068	ETH_MACAHR5	MAC 过滤地址高位寄存器 5	0x0000FFFF
0x006C	ETH_MACALR5	MAC 过滤地址低位寄存器 5	0xFFFFFFFF
0x0070	ETH_MACAHR6	MAC 过滤地址高位寄存器 6	0x0000FFFF
0x0074	ETH_MACALR6	MAC 过滤地址低位寄存器 6	0xFFFFFFFF
0x0078	ETH_MACAHR7	MAC 过滤地址高位寄存器 7	0x0000FFFF
0x007C	ETH_MACALR7	MAC 过滤地址低位寄存器 7	0xFFFFFFFF
0x0080	ETH_MACAHR8	MAC 过滤地址高位寄存器 8	0x0000FFFF
0x0084	ETH_MACALR8	MAC 过滤地址低位寄存器 8	0xFFFFFFFF
0x0088	ETH_MACAHR9	MAC 过滤地址高位寄存器 9	0x0000FFFF
0x008C	ETH_MACALR9	MAC 过滤地址低位寄存器 9	0xFFFFFFFF
0x0090	ETH_MACAHR10	MAC 过滤地址高位寄存器 10	0x0000FFFF
0x0094	ETH_MACALR10	MAC 过滤地址低位寄存器 10	0xFFFFFFFF
0x0098	ETH_MACAHR11	MAC 过滤地址高位寄存器 11	0x0000FFFF
0x009C	ETH_MACALR11	MAC 过滤地址低位寄存器 11	0xFFFFFFFF
0x00A0	ETH_MACAHR12	MAC 过滤地址高位寄存器 12	0x0000FFFF
0x00A4	ETH_MACALR12	MAC 过滤地址低位寄存器 12	0xFFFFFFFF
0x00A8	ETH_MACAHR13	MAC 过滤地址高位寄存器 13	0x0000FFFF
0x00AC	ETH_MACALR13	MAC 过滤地址低位寄存器 13	0xFFFFFFFF
0x00B0	ETH_MACAHR14	MAC 过滤地址高位寄存器 14	0x0000FFFF
0x00B4	ETH_MACALR14	MAC 过滤地址低位寄存器 14	0xFFFFFFFF
0x00B8	ETH_MACAHR15	MAC 过滤地址高位寄存器 15	0x0000FFFF
0x00BC	ETH_MACALR15	MAC 过滤地址低位寄存器 15	0xFFFFFFFF
0x0800	ETH_MACAHR16	MAC 过滤地址高位寄存器 16	0x0000FFFF
0x0804	ETH_MACALR16	MAC 过滤地址低位寄存器 16	0xFFFFFFFF
0x0808	ETH_MACAHR17	MAC 过滤地址高位寄存器 17	0x0000FFFF
0x080C	ETH_MACALR17	MAC 过滤地址低位寄存器 17	0xFFFFFFFF
0x0810	ETH_MACAHR18	MAC 过滤地址高位寄存器 18	0x0000FFFF
0x0814	ETH_MACALR18	MAC 过滤地址低位寄存器 18	0xFFFFFFFF
0x0818	ETH_MACAHR19	MAC 过滤地址高位寄存器 19	0x0000FFFF
0x081C	ETH_MACALR19	MAC 过滤地址低位寄存器 19	0xFFFFFFFF
0x0820	ETH_MACAHR20	MAC 过滤地址高位寄存器 20	0x0000FFFF
0x0824	ETH_MACALR20	MAC 过滤地址低位寄存器 20	0xFFFFFFFF
0x0828	ETH_MACAHR21	MAC 过滤地址高位寄存器 21	0x0000FFFF
0x082C	ETH_MACALR21	MAC 过滤地址低位寄存器 21	0xFFFFFFFF
0x0830	ETH_MACAHR22	MAC 过滤地址高位寄存器 22	0x0000FFFF
0x0834	ETH_MACALR22	MAC 过滤地址低位寄存器 22	0xFFFFFFFF

Offset	Acronym	Register Name	Reset
0x0838	ETH_MACAHR23	MAC 过滤地址高位寄存器 23	0x0000FFFF
0x083C	ETH_MACALR23	MAC 过滤地址低位寄存器 23	0xFFFFFFFF
0x0840	ETH_MACAHR24	MAC 过滤地址高位寄存器 24	0x0000FFFF
0x0844	ETH_MACALR24	MAC 过滤地址低位寄存器 24	0xFFFFFFFF
0x0848	ETH_MACAHR25	MAC 过滤地址高位寄存器 25	0x0000FFFF
0x084C	ETH_MACALR25	MAC 过滤地址低位寄存器 25	0xFFFFFFFF
0x0850	ETH_MACAHR26	MAC 过滤地址高位寄存器 26	0x0000FFFF
0x0854	ETH_MACALR26	MAC 过滤地址低位寄存器 26	0xFFFFFFFF
0x0858	ETH_MACAHR27	MAC 过滤地址高位寄存器 27	0x0000FFFF
0x085C	ETH_MACALR27	MAC 过滤地址低位寄存器 27	0xFFFFFFFF
0x0860	ETH_MACAHR28	MAC 过滤地址高位寄存器 28	0x0000FFFF
0x0864	ETH_MACALR28	MAC 过滤地址低位寄存器 28	0xFFFFFFFF
0x0868	ETH_MACAHR29	MAC 过滤地址高位寄存器 29	0x0000FFFF
0x086C	ETH_MACALR29	MAC 过滤地址低位寄存器 29	0xFFFFFFFF
0x0870	ETH_MACAHR30	MAC 过滤地址高位寄存器 30	0x0000FFFF
0x0874	ETH_MACALR30	MAC 过滤地址低位寄存器 30	0xFFFFFFFF
0x0878	ETH_MACAHR31	MAC 过滤地址高位寄存器 31	0x0000FFFF
0x087C	ETH_MACALR31	MAC 过滤地址低位寄存器 31	0xFFFFFFFF
0x0008	ETH_MACHTHR	MAC HASH 表高位寄存器	0x00000000
0x000C	ETH_MACHTLR	MAC HASH 表低位寄存器	0x00000000
0x001C	ETH_MACVLTR	MAC VLAN 标签寄存器	0x00000000
0x0018	ETH_DMAMDR	DMA 工作模式寄存器	0x00000000
0x1000	ETH_DMABSR	DMA 总线控制寄存器	0x00020101
0x101C	ETH_DMAIR	DMA 中断寄存器	0x00000000
0x1014	ETH_DMASR	DMA 状态寄存器	0x00000000
0x1020	ETH_DMAFLCR	DMA 帧丢失统计寄存器	0x00000000
0x1024	ETH_DMAWDTR	DMA 看门狗定时寄存器	0x00000000
0x1008	ETH_DMARXPDR	DMA 接收轮询寄存器	0x00000000
0x1004	ETH_DMATXPDR	DMA 发送轮询寄存器	0x00000000
0x100C	ETH_DMARXDSAR	DMA 接收描述符地址寄存器	0x00000000
0x1010	ETH_DMATXDSAR	DMA 发送描述符地址寄存器	0x00000000
0x104C	ETH_DMACUTRXDSAR	DMA 当前接收描述符地址寄存器	0x00000000
0x1048	ETH_DMACUTTXDSAR	DMA 当前发送描述符地址寄存器	0x00000000
0x1054	ETH_DMACUTRXBFAR	DMA 当前接收缓冲区地址寄存器	0x00000000
0x1050	ETH_DMACUTTXBFAR	DMA 当前发送缓冲区地址寄存器	0x00000000

## 24.5.1 MAC 子控制器寄存器

### 24.5.1.1 SMI 地址寄存器 (ETH\_MACSMIAR)

偏移地址: 0x10

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	Res.															
Type																
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	SMI_ADD[4:0]				SMI_REG[4:0]				SMI_CLK[3:0]				SMI_WR	SMI_BY		
Type	rw				rw				rw				rw	rc_w1		

Bit	Field	Type	Reset	Description															
31:16	Reserved			保留, 始终读为 0															
15:11	SMI_ADD	rw	0x00	指示要访问 32 个可能的 PHY 器件中的哪一个															
10:6	SMI_REG	rw	0x00	指示在所选 PHY 器件中要选择寄存器															
5:2	SMI_CLK	rw	0x00	基于系统频率决定 MDC 时钟频率: <table> <thead> <tr> <th>选项</th> <th>系统频率</th> <th>MDC 时钟</th> </tr> </thead> <tbody> <tr> <td>0000</td> <td>60-100MHz</td> <td>系统频率/42</td> </tr> <tr> <td>0001</td> <td>100-120MHz</td> <td>系统频率/62</td> </tr> <tr> <td>0010</td> <td>20-35MHz</td> <td>系统频率/16</td> </tr> <tr> <td>0011</td> <td>35-60MHz</td> <td>系统频率/26</td> </tr> </tbody> </table> 请不要设定其它值	选项	系统频率	MDC 时钟	0000	60-100MHz	系统频率/42	0001	100-120MHz	系统频率/62	0010	20-35MHz	系统频率/16	0011	35-60MHz	系统频率/26
选项	系统频率	MDC 时钟																	
0000	60-100MHz	系统频率/42																	
0001	100-120MHz	系统频率/62																	
0010	20-35MHz	系统频率/16																	
0011	35-60MHz	系统频率/26																	
1	SMI_WR	rw	0x00	0: SMI 读操作 1: SMI 写操作															

Bit	Field	Type	Reset	Description
0	SMI_BY	rc_w1	0x00	<p>向 ETH_MACSMIAR 和 ETH_MACSMIDR 写入前，此位应读取逻辑 0</p> <p>向 ETH_MACSMIAR 写入过程中，此位也必须复位为 0</p> <p>在 PHY 寄存器访问过程中，此位由应用程序设为 1，指示读或写访问正在进行中</p> <p>在对 PHY 进行写操作过程中，ETH_MACSMIDR 应始终有效，直到 MAC 将此位清零；在对 PHY 进行读操作过程中，ETH_MACSMIDR 始终无效，直到 MAC 将此位清零。此位清零后，才可以向 ETH_MACSMIDR 写入值</p>

### 24.5.1.2 SMI 数据寄存器 (ETH\_MACSMIDR)

偏移地址: 0x14

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	Res.															
Type																
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	SMI_DATA[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:16	Reserved			保留，始终读为 0
15:0	SMI_DATA	rw	0x00	站点管理读操作之后从 PHY 中读取的 16 位数据值，或站点管理写操作之前要写入 PHY 的 16 位数据值

### 24.5.1.3 MAC 配置寄存器 (ETH\_MACCR)

偏移地址: 0x00

复位值: 0x0000\_8000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	Res.						CST	Res.	WTD	JD	Res.	Res.	IFG[2:0]			MCRS

Type							rw		rw	rw			rw			rw
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	Res.	ROD	LM	DM	IPCO	RETY	Res.	APCS	BL[1:0]		DLYC	TE	RE	Res.	Res.	
Type		rw	rw	rw	rw	rw		rw	rw		rw	rw	rw			

Bit	Field	Type	Reset	Description	
31:26	Reserved			保留, 始终读为 0	
25	CST	rw	0x0	0: 该功能无效 1: 在将帧转发到应用之前, 去除并丢弃所有 TYPE (类型字段大于 0x0600) 帧的最后 4 个字节 (FCS 字段)	
24	Reserved			保留, 始终读为 0	
23	WTD	rw	0x0	0: MAC 允许接收的帧不超过 2048 字节, 并会截断超过此限制接收的任何字节 1: MAC 禁止接收器上的看门狗定时器, 并可接收多达 16384 字节的帧	
22	JD	rw	0x0	0: 发送期间应用发送超过 2048 字节的数据, MAC 会截断发送器 1: MAC 禁止发送器上的 Jabber 定时器, 并可发送多达 16384 字节的帧	
21:20	Reserved			保留, 始终读为 0	
19:17	IFG	rw	0x0	发送期间帧间的最小间隙 000: 96 位时间 001: 88 位时间 010: 80 位时间 ..... 111: 40 位时间 注: 在半双工模式下, 最小 IFG 仅能配置为 64 位时间 (IFG=100) , 不考虑更低的值	

Bit	Field	Type	Reset	Description
16	MCRS	rw	0x0	<p>0: 半双工模式下帧发送期间的 MII_CRS 信号有效时, MAC 发送器会生成载波侦听错误, 甚至中止发送</p> <p>1: 半双工模式下帧发送期间的 MII_CRS 信号被忽略, 不会因在此发送期间载波丢失或无载波而生成错误</p> <p>注: 该位只在半双工模式时有效, 全双工模式时无效</p>
15:14	Reserved			保留, 始终读为 10
13	ROD	rw	0x0	<p>0: 在半双工模式下, MAC 接收来自 PHY 的所有包</p> <p>1: 在半双工模式下, MAC 禁止接收帧</p> <p>注: 该位只在半双工模式时有效, 全双工模式时无效</p>
12	LM	rw	0x0	<p>0: Loopback 模式无效</p> <p>1: Loopback 模式有效</p>
11	DM	rw	0x0	<p>0: 半双工模式</p> <p>1: 全双工模式</p>
10	IPCO	rw	0x0	<p>0: IP Checksum Offload 功能无效</p> <p>1: IP Checksum Offload 功能有效</p>
9	RETY	rw	0x0	<p>0: 在 MII 模式下发生冲突时, MAC 会尝试根据该寄存器的 BL 位的设置进行重试</p> <p>1: 在 MII 模式下发生冲突时, MAC 会忽略当前帧发送并以发送帧状态下过度冲突错误报告帧中止, 即 MAC 仅尝试一次发送</p> <p>注: 该位只在半双工模式时有效, 全双工模式时无效</p>
8	Reserved			保留, 始终读为 0
7	APCS	rw	0x0	<p>0: MAC 不管接收到的帧的长度, 传送所有接收帧到应用</p> <p>1: MAC 在长度字段值小于 1536 字节时自动去除接收帧上的 PAD/FCS 字段, 在长度字段大于或等于 1536 字节时不去除 PAD/FCS 字段</p>

Bit	Field	Type	Reset	Description
6:5	BL	rw	0x0	<p>后退限制决定发生冲突后重试期间 MAC 在重新安排一次发送之前等待的随机整数 (r) 个时间片延迟 (512 位时间)</p> <p>00: k = min (n, 10) 01: k = min (n, 8) 10: k = min (n, 4) 11: k = min (n, 1)</p> <p>其中 n=重新发送尝试的次数。随机整数 r 的取值范围为: 0&lt;= r&lt;2^k</p> <p>注: 该位只在半双工模式时有效, 全双工模式时无效</p>
4	DLYC	rw	0x0	<p>0: 禁止延迟检查功能。MAC 发生延迟, 直到 CRS 信号变为无效信号 1: 使能延迟检查功能。当发送状态机在延迟超过 24288 位时间时, MAC 将指示帧中止状态, 同时在发送帧状态中将过度延迟错误位置 1</p> <p>注 1: 延迟的产生: 当发送器准备好发送但因 MII 上含有有效 CRS (载波侦听) 信号而被阻止时延迟计数就开始。延迟时间是非累积性的, 如果发送器延迟 10000 位时间, 则重新执行一遍发送、冲突、后退, 然后在完成后退后复位延迟计数器为 0 并重新开始计数</p> <p>注 2: 该位只在半双工模式时有效, 全双工模式时无效</p>
3	TE	rw	0x0	0: 发送无效 1: 发送使能
2	RE	rw	0x0	0: 接收无效 1: 接收使能
1:0	Reserved			保留, 始终读为 0

#### 24.5.1.4 MAC 状态寄存器 (ETH\_MACSR)

偏移地址: 0x24

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	Res.						TXFF	TXFNE	Res.	TXFWA	TXFRS[1:0]		MTP	MTS[1:0]		MTE
Type							r	r		r	r		r	r		r
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	Res.						RXFFL[1:0]		Res.	RXFRS[1:0]		RXFWA	Res.	MRS[1:0]		MRE
Type							r			r		r		r		r

Bit	Field	Type	Reset	Description
31:26	Reserved			保留, 始终读为 0
25	TXFF	r	0x0	0: TxFIFO 未满 1: TxFIFO 已满, 无法再接收发送的帧
24	TXFNE	r	0x0	0: TxFIFO 空 1: TxFIFO 非空, 还有未发送的帧
23	Reserved			保留, 始终读为 0
22	TXFWA	r	0x0	0: TxFIFO 写无效 1: TxFIFO 写控制器有效且正在将数据传输到 TxFIFO
21:20	TXFRS	r	0x0	00: 空闲态 01: 读状态 (数据从 TxFIFO 往 MAC 发送器传送) 10: 等待来自 MAC 发送器的 Tx Status 11: 写入收到的 Tx Status 或刷新 TxFIFO
19	MTP	r	0x0	0: MAC 发送器处于发送数据状态 1: MAC 发送器处于暂停状态 (全双工模式时), 不安排任何帧发送
18:17	MTS	r	0x0	00: 空闲态 01: 等待前一个帧的状态或 IFG/回退阶段结束 10: 生成并发送暂停控制帧 (在全双工模式下) 11: 传输要发送的输入帧
16	MTE	rw	0x00	0: MAC MII 发送引擎处于空闲态 1: MAC MII 发送引擎正在主动发送数据而未处于空闲状态
15:10	Reserved			保留, 始终读为 0
9:8	RXFLL	r	0x0	00: RxFIFO 为空 01: RxFIFO 填充级别低于流控制, 取消激活阈值 10: RxFIFO 填充级别高于流控制, 激活阈值 11: RxFIFO 已满

Bit	Field	Type	Reset	Description
7	Reserved			保留, 始终读为 0
6:5	RXFRS	r	0x0	00: 空闲态 01: 读取帧数据 10: 读取帧状态或时间戳 11: 刷新帧数据和状态
4	RXFWA	r	0x0	0: RxFIFO 写无效 1: RxFIFO 写有效, 并正在将收到的帧传输到 RxFIFO
3	Reserved			保留, 始终读为 0
2:1	MRS	r	0x0	00: 空闲 01: MAC 接收器中的 FIFO 写控制器在工作 10: MAC 接收器中的 FIFO 读控制器在工作 11: MAC 接收器中的 FIFO 读和写控制器在工作
0	MRE	r	0x0	0: MAC MII 接收引擎处于空闲态 1: MAC MII 接收引擎正在主动接收数据而未处于空闲状态

#### 24.5.1.5 MAC 流控寄存器 (ETH\_MACFCR)

偏移地址: 0x18

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	PSET[15:0]															
Type	rw															
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	Res.							DZPQ	Res.	PLT[1:0]		UP	FRE	FTE	FCBBPA	
Type	rw							rw		rw		rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
31:16	PSET	rw	0x00	该位表示发送控制帧中暂停时间字段要使用的值
15:8	Reserved			保留, 始终读为 0
7	DZPQ	rw	0x0	0: 当流控制信号有效时, 正常生成零时间片暂停控制帧 1: 当流控制信号有效时, 禁止生成零时间片暂停控制帧
6	Reserved			保留, 始终读为 0
5:4	PLT	rw	0x0	配置暂停定时器的阈值, 达到该值时会自动重新传输 PAUSE 帧 00: 暂停时间减去 4 个时隙 01: 暂停时间减去 28 个时隙 10: 暂停时间减去 144 个时隙 11: 暂停时间减去 256 个时隙 注 1: 时隙定义为 MII 接口每发送 512 位 (64 字节) 所需的时间 注 2: 该阈值应始终小于暂停时间。例如, 如果暂停时间 PSET=100H (256 个时隙), 而 PLT=01, 则在第一个 PAUSE 帧发送完成后, 第二个 PAUSE 帧会在第 228 (256-28) 个时隙启动时自动发送
3	UP	rw	0x0	0: MAC 仅检测具有 802.3x 标准中指定的唯一多播地址的暂停帧 1: MAC 除了检测具有唯一多播地址的暂停帧外, 还会检测具有 ETH_MACA0HR 和 ETH_MACA0LR 寄存器所指定的站单播地址的暂停帧
2	FRE	rw	0x0	0: 禁止暂停帧的解码功能 1: MAC 对接收到的暂停帧进行解码, 并禁止其在指定的暂停时间内发送
1	FTE	rw	0x0	全双工模式时: 0: 禁止 MAC 传送任何暂停帧 1: 使能流控制操作, MAC 发送暂停帧 半双工模式时: 0: 禁止背压功能 1: 使能背压操作

Bit	Field	Type	Reset	Description
0	FCBBPA	rw	0x0	<p>向流控制寄存器写入数据前此位应读为 0</p> <p>全双工模式时：</p> <p>此位置 1 时 MAC 子控制器启动暂停控制帧。在控制帧发送过程中，此位保持置 1 以指示帧发送正在进行中。当暂停控制帧发送完成后，MAC 会将此位复位为 0。此位清零后，才可以对流控制寄存器执行写操作</p> <p>在半双工模式时：</p> <p>此位置 1 时，MAC 子控制器将置位背压功能。在背压操作期间，当 MAC 接收到新帧时，发送器会开始发送一个导致冲突的 Jam 信号</p>

#### 24.5.1.6 MAC 地址过滤控制寄存器 (ETH\_MACAFR)

偏移地址：0x04

复位值：0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	RALL	Res.														VTFE
Type	rw															rw
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	Res.					HPF	SAF	SAIF	PCF[1:0]		DBF	PMF	DAIF	HM	HU	PM
Type						rw	rw	rw	rw		rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31	RALL	rw	0x0	<p>0: MAC 接收器将通过地址过滤的接收帧传送给应用</p> <p>1: MAC 接收器将所有的接收帧传送到应用，不管它们是否已通过地址过滤</p>
30:17	Reserved			保留，始终读为 0
16	VTFE	rw	0x0	<p>0: MAC 接收器将所有接收的 VLAN 帧传送到应用，不管它们是否已通过 VLAN 标记过滤</p> <p>1: MAC 接收器丢弃与 VLAN 标记设定的过滤值比较不匹配的 VLAN 帧</p>

Bit	Field	Type	Reset	Description
15:11	Reserved			保留, 始终读为 0
10	HPF	rw	0x0	<p>0: 若该寄存器的 HU 位或 HM 位置 1, 则地址过滤器只传递与 Hash 过滤器匹配的帧</p> <p>1: 若该寄存器的 HU 位或 HM 位置 1, 则地址过滤器传递与地址过滤器或 Hash 过滤器匹配的帧</p>
9	SAF	rw	0x0	<p>0: MAC 接收器将所有接收的帧传递到应用, 不管它们是否已通过源地址过滤</p> <p>1: MAC 接收器丢弃与源地址设定的过滤值比较不匹配的帧</p>
8	SAIF	rw	0x0	<p>0: MAC 接收器接收到的帧 SA 字段与设定的过滤值不匹配时, 认定为过滤失败</p> <p>1: MAC 接收器接收到的帧 SA 字段与设定的过滤值匹配时, 认定为过滤失败</p>
7:6	PCF	rw	0x0	<p>对控制帧（包括单播暂停帧和多播暂停帧）的转发做如下操作：</p> <p>00: MAC 阻止所有控制帧到达应用</p> <p>01: MAC 将除了暂停控制帧以外的所有控制帧转发到应用, 即使这些帧的地址过滤失败</p> <p>10: MAC 将所有控制帧转发到应用, 即使这些帧的地址过滤失败</p> <p>11: MAC 转发通过地址过滤的控制帧</p> <p>关于暂停控制帧的几点说明:</p> <p>注 1: 在全双工模式时, 暂停控制帧的处理由流控寄存器 (ETH_MACFCR) 的 FRE 位决定</p> <p>注 2: 在流控寄存器 (ETH_MACFCR) 的 UP 位置 1 时, 接收到的帧的目标地址和 MAC 地址寄存器 0 设定的值匹配时, 该帧被认为是单播暂停帧</p> <p>注 3: 接收到的帧的 TYPE 段值为 0x8808 或 OPCODE 段值为 0x0001 时, 该帧被认为是暂停控制帧</p>
5	DBF	rw	0x0	<p>0: 地址过滤器不对接收到的广播帧进行过滤</p> <p>1: 地址过滤器对所有传入的广播帧进行过滤</p> <p>注: 该位置 1 时, 其优先级高于其它过滤设定</p>
4	PMF	rw	0x0	<p>0: 对接收到的带多播目标地址（目标地址字段的第一位是 1）的帧进行地址过滤, 过滤方式由该寄存器的 HM 位决定</p> <p>1: 对接收到的带多播目标地址（目标地址字段的第一位是 1）的帧不进行地址过滤</p>

Bit	Field	Type	Reset	Description
3	DAIF	rw	0x0	0: MAC 接收器接收到的单播帧或多播帧的目标地址与设定的过滤值不匹配时，认定为过滤失败 1: MAC 接收器接收到的单播帧或多播帧的目标地址与设定的过滤值匹配时，认定为过滤失败
2	HM	rw	0x0	0: MAC 对多播帧执行完美目标地址过滤，即将 DA 字段与设定的过滤值进行比较 1: MAC 对多播帧执行 Hash 目标地址过滤
1	HU	rw	0x0	0: MAC 对单播帧执行完美目标地址过滤，即将 DA 字段与设定的过滤值进行比较 1: MAC 对单播帧执行 Hash 目标地址过滤
0	PM	rw	0x0	0: 地址过滤器进行正常的地址过滤 1: 地址过滤器不管目标或源地址，都传送所有传入的帧

#### 24.5.1.7 MAC 过滤地址高位寄存器 (ETH\_MACAHRx) (x=0~31)

偏移地址: 0x40/0x48/0x50/0x58...0xB0/0xB8、0x800/0x808/0x810/0x818...0x870/0x878

复位值: 0x8000\_FFFF(x=0)/0x0000\_FFFF(x=1~31)

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	ADDE	SELE	MBYTEC[5:0]										Res.			
Type	rw	rw	rw													
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	MACADDH[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31	ADDE	rw	0x0	0: 忽略 MAC 地址 x(x=0~31)地址过滤 1: 使用 MAC 地址 x(x=0~31)进行地址过滤 注: 当 x=0 时, 该位固定为 1, 为只读位

Bit	Field	Type	Reset	Description
30	SELE	rw	0x0	0: 用 MAC 地址 $x(x=1\sim31)$ 对接收帧的 DA 进行过滤 1: 用 MAC 地址 $x(x=1\sim31)$ 对接收帧的 SA 进行过滤 注: 该位只在 $x=1\sim31$ 时可设定, $x=0$ 时固定为 0
29:24	MBYTEC	rw	0x00	地址过滤时屏蔽对应的字节 b29: 屏蔽 ADDRH[15:8]位 b28: 屏蔽 ADDRH[7:0]位 b27: 屏蔽 ADDRL[31:24]位 b26: 屏蔽 ADDRL[23:16]位 b25: 屏蔽 ADDRL[15:8]位 b24: 屏蔽 ADDRL[7:0]位 注: 该位只在 $x=1\sim31$ 时可设定, $x=0$ 时固定为 0
23:16	Reserved			保留, 始终读为 0
15:0	MACADDH	rw	0x00	当 $x=0$ 时, MAC 使用此字段过滤所接收的帧以及将 MAC 地址插入到发送流控制(暂停)帧中 当 $x=1\sim31$ 时, MAC 使用此字段过滤所接收的帧

#### 24.5.1.8 MAC 过滤地址低位寄存器 (ETH\_MACALRx) ( $x=0\sim31$ )

偏移地址: 0x44/0x4C/0x54/0x5C...0xB4/0xBC、0x804/0x80C/0x814/0x81C...0x874/0x87C

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	MACADDL[31:16]															
Type	rw															
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	MACADDL[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:0	MACADDL	rw	0x00	<p>当 x=0 时, MAC 使用此字段过滤所接收的帧以及将 MAC 地址插入到发送流控制(暂停)帧中</p> <p>当 x=1~31 时, MAC 使用此字段过滤所接收的帧</p>

#### 24.5.1.9 MAC HASH 表高位寄存器 (ETH\_MACHTHR)

偏移地址: 0x08

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	MACHTABH[31:16]															
Type	rw															
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	MACHTABH[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:0	MACHTABH	rw	0x00	Hash Table 的高 32 位值

#### 24.5.1.10 MAC HASH 表地位寄存器 (ETH\_MACHTLR)

偏移地址: 0x0C

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	MACHTABL[31:16]															
Type	rw															
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	MACHTABL[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:0	MACHTABLE	rw	0x00	Hash Table 的低 32 位值

#### 24.5.1.11 MAC VLAN 标签寄存器 (ETH\_MACVLTR)

偏移地址: 0x1C

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	Res.															
Type																
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	VLANTAG[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0
15:0	VLANTAG	rw	0x00	此字段包含用于过滤 VLAN 帧的标记字段的比较值, 该位与接收的 VLAN 帧的第 15 和第 16 字节进行比较

#### 24.5.2 DMA 子控制器寄存器

### 24.5.2.1 DMA 工作模式寄存器 (ETH\_DMAMDR)

偏移地址: 0x1018

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	Res.				DTCOE	RSF	DFRF	Res.	Res.	TSF	FTF	Res.			TTC[2]	
Type					rw	rw				rw	rw				rw	
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	TTC[1:0]		STT	Res.				FEF	FUF	DGF	RTC[1:0]		OSF	STR	Res.	
Type	rw		rw					rw	rw	rw	rw		rw	rw		

Bit	Field	Type	Reset	Description
31:27	Reserved			保留, 始终读为 0
26	DTCOE	rw	0x0	0: 如果该寄存器的 FEF 位置位 1, 则丢弃所有错误帧 1: 如果接收帧中仅存在由接收校验和减荷引擎检测出来的错误, 则内核不会丢弃该帧 <small>注: 这类帧在 MAC 接收到的以太网帧中没有任何错误 (包括 CRC 错误), 而仅在封装的有效负载中有错误</small>
25	RSF	rw	0x0	0: Rx FIFO 中的帧达到该寄存器的 RTC 位指定的值时, 才进行 Rx FIFO 的读动作 1: Rx FIFO 写入完整帧后, 就执行读动作, 同时忽略 RTC 位
24	DFRF	rw	0x0	该位置 1 时, Rx DMA 不会因为接收描述符/缓冲区不可用而刷新任何帧, 参见“接收过程挂起”章节
23:22	Reserved			保留, 始终读为 0
21	TSF	rw	0x0	0: Tx FIFO 中的帧达到该寄存器的 TTC 位指定的值时, 才进行 Tx FIFO 的读动作, 即 MAC 发送才启动 1: Tx FIFO 中有一个完整帧, 则 MAC 发送会启动, 同时忽略 TTC 位 <small>注: 该位只有在已停止传输时才能更改</small>

Bit	Field	Type	Reset	Description																
20	FTF	rw	0x0	<p>该位置 1 时, TxFIFO 复位为默认值, 此时 TxFIFO 中的所有数据均会丢失/刷新。刷新操作结束时该位自动清零</p> <p>注: 此位清零之前不得对工作模式寄存器执行写操作</p>																
19:17	Reserved			保留, 始终读为 0																
16:14	TTC	rw	0x0	<p>该位用于控制 TxFIFO 的阈值级别。当 TxFIFO 中的帧大小大于阈值时启动 DMA 发送。此外, 长度小于阈值的完整帧会自动传输</p> <table> <tr><td>000:</td><td>64</td></tr> <tr><td>001:</td><td>128</td></tr> <tr><td>010:</td><td>192</td></tr> <tr><td>011:</td><td>256</td></tr> <tr><td>100:</td><td>40</td></tr> <tr><td>101:</td><td>32</td></tr> <tr><td>110:</td><td>24</td></tr> <tr><td>111:</td><td>16</td></tr> </table> <p>注: 该位只有在 TSF 位为 0 时才使用</p>	000:	64	001:	128	010:	192	011:	256	100:	40	101:	32	110:	24	111:	16
000:	64																			
001:	128																			
010:	192																			
011:	256																			
100:	40																			
101:	32																			
110:	24																			
111:	16																			
13	STT	rw	0x0	<p>0: 发送过程在完成发送当前帧的任务之后进入停止状态, 并保存发送列表中的下一个描述符位置, 该描述符位置在重启发送后会成为当前位置</p> <p>1: 启动发送过程, DMA 检查发送列表中的当前位置, 即发送描述符地址列表寄存器 (ETH_DMA_TXDLADR) 设定的地址, 或者上一次停止发送时的保留位置尝试获取描述符, 来查找待发送的帧</p> <p>如果当前描述符不属于 TxDMA, 则发送过程会进入挂起状态; 如果该命令在设置 ETH_DMA_TXDLADR 寄存器之前发出, 则 TxDMA 行为无法预知</p> <p>注: 启动发送命令只有在传输已停止时才有效; 停止发送命令只有在当前帧发送过程结束或发送过程处于挂起状态时才有效</p>																
12:8	Reserved			保留, 始终读为 0																

Bit	Field	Type	Reset	Description
7	FEF	rw	0x0	<p>0: RxFIFO 会丢弃带错误状态 (CRC 错误、冲突错误、巨帧、看门狗超时、上溢) 的帧 如果某个帧的起始字节指针已传输到读控制器端 (阈值模式下), 则不会丢弃该帧; 如果总线上未传输帧的起始字节, 则 RxFIFO 会丢弃此错误帧</p> <p>1: 除矮帧错误帧之外的所有帧都会转发</p>
6	FUF	rw	0x0	<p>0: RxFIFO 丢弃所有不足 64 字节的帧, 除非因接收阈值下限更低 (例如 RTC=01) 而导致此类帧已传输</p> <p>1: RxFIFO 转发包含 PAD 字节和 CRC 的过小帧 (无错误但长度不足 64 字节的帧)</p>
5	DGF	rw	0x0	<p>0: RxFIFO 不丢弃巨帧</p> <p>1: RxFIFO 丢弃巨帧</p>
4:3	RTC	rw	0x0	<p>该位用于控制 RxFIFO 的阈值级别。当 RxFIFO 中的帧大小大于阈值时启动 DMA 传输 (请求)。此外, 长度小于阈值的完成帧会自动传输</p> <p>00: 64 01: 32 10: 96 11: 128</p> <p>注: 该位只有在 RSF 位为 0 时才使用</p>
2	OSF	rw	0x0	该位置 1 时会命令 DMA 处理第二个发送数据帧, 即使尚未获得首个帧的状态
1	SR	rw	0x0	<p>0: RxDMA 会在传输当前帧之后停止操作, 并保存接收列表中的下一个描述符位置, 该描述符位置在接收过程重启后会成为当前位置</p> <p>1: 启动接收过程, DMA 尝试从接收列表中的当前位置, 即, 接收描述符地址列表寄存器 (ETH_DMA_RDLADR) 设定的地址, 或者上一次停止接收时的保留位置尝试获取描述符, 并处理传入帧</p> <p>如果 RxDMA 未占据任何描述符, 则接收过程会进入挂起状态; 如果该命令在设置 ETH_DMA_RXLADR 寄存器之前发出, 则 RxDMA 行为无法预知</p> <p>注: 启动接收命令只有在已停止接收时才有效; 停止接收命令只有在当前接收过程处于运行 (等待接收数据包) 或挂起状态时才有效。</p>
0	Reserved			保留, 始终读为 0

## 24.5.2.2 DMA 总线控制寄存器 (ETH\_DMABSR)

偏移地址: 0x1000

复位值: 0x0002\_0101

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16		
Field	Res.						ALL	Res.										FBST
Type							rw											rw
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0		
Field	FTP[1:0]		PBL[5:0]						Res.	DSL[4:0]				DMAA	SR			
Type	rw		rw							rw				rw	rw			

Bit	Field	Type	Reset	Description
31:26	Reserved			保留, 始终读为 0
25	ALL	rw	0x0	<p>该位置 1 时:</p> <p>1) 若 FBST=1 时, AHB 接口会生成与起始地址 LS 位对齐的所有突发</p> <p>2) 若 FBST=0 时, 则第一个突发(访问数据缓冲器的起始地址)不对齐, 但后续的突发与地址对齐</p>
24:18	Reserved			保留, 始终读为 0
17	Reserved			保留, 始终读为 1
16	FBST	rw	0x0	<p>0: DMA Master 接口只使用 SINGLE 和 INCR 访问类型</p> <p>1: DMA Master 接口使用 SINGLE 和 INCR4、INCR8、INCR16 访问类型</p>

Bit	Field	Type	Reset	Description
15:14	FTPR	rw	0x0	<p>Rx DMA 和 Tx DMA 对 BUS 的抢占优先级比</p> <p>00: 1: 1 11: 2: 1 10: 3: 1 11: 4: 1</p> <p>注 1: 该位只在该寄存器的 <b>DMAA</b> 位为 0 时有效</p>
13:8	PBL	rx	0x01	<p>该位指示要在一个 TxDMA 事务或 RxDMA 事务中传输的最大节拍数，这是在单个块读/写操作中使用的最大值。DMA 每次在主机总线上开始突发传输时，始终尝试按 PBL 中指定的方式进行突发。允许使用值 1、2、4、8、16 和 32 对 PBL 进行编程，任何其它值都会产生未定义的行为</p> <p>注: <b>PBL</b> 值有以下限制:</p> <ol style="list-style-type: none"> <li>1) 可能的最大节拍数(PBL)受 <b>TxFIFO</b> 和 <b>RxFIFO</b> 大小的限制</li> <li>2) 请不要编程超出范围的 <b>PBL</b> 值</li> </ol>
7	Reserved			保留，始终读为 0
6:2	DSL	rw	0x0	该位指定两个不以链式结构连接的描述符之间跳过的 Word 数，地址从当前描述符结束处开始跳到下一个描述符起始处。当 DSL 值等于零时，在环形模式下，DMA 会将描述符表视为连续的
1	DMAA	rw	0x0	<p>0: 循环优先级，仲裁方式由该寄存器的 <b>FTPR</b> 位决定 1: 固定优先级，接收优先级高于发送</p>
0	SR	rw	0x0	该位置 1 时，DMA 控制器复位所有的 MAC 子系统的内部寄存器和逻辑。在所有内核时钟域完成复位操作后，该位自动清零；在重新写 MAC 寄存器，应确保该位为 0

### 24.5.2.3 DMA 中断寄存器 (ETH\_DMAIR)

偏移地址: 0x101C

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	Res.															NIE
Type																rw
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	AIE	ERE	FBE	Res.	Res.	ETE	RWE	RSE	RUE	RIE	UNE	OVE	TJE	TUE	TSE	TIE
Type	rw	rw	rw			rw										

Bit	Field	Type	Reset	Description
31:17	Reserved			保留, 始终读为 0
16	NIE	rw	0x0	<p>0: 普通事件中断无效 1: 普通事件中断有效 注: 普通事件是指动作状态寄存器 <i>ETH_DMADMASR</i> 的 <i>B0</i>、<i>B2</i>、<i>B6</i>、<i>B14</i> 位所指示的事件</p>
15	AIE	rw	0x0	<p>0: 异常事件中断无效 1: 异常事件中断有效 注: 异常事件是指动作状态寄存器 <i>ETH_DMA_DMASTR</i> 的 <i>B1</i>、<i>B3</i>、<i>B4</i>、<i>B5</i>、<i>B7</i>、<i>B8</i>、<i>B9</i>、<i>B10</i>、<i>B14</i> 位所指示的事件</p>
14	ERE	rw	0x0	<p>0: 提前接收中断无效 1: 提前接收中断使能 注: 该位在 <i>NIE</i> 位置 1 后有效</p>
13	FBE	rw	0x0	<p>0: 致命总线错误中断无效 1: 致命总线错误中断有效 注: 该位在 <i>AIE</i> 位置 1 后有效</p>
12:11	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
10	ETE	rw	0x0	<p>0: 提前发送中断无效 1: 提前发送中断有效 注: 该位在 <i>AIE</i> 位置 1 后有效</p>
9	RWE	rw	0x0	<p>0: 接收看门狗溢出中断无效 1: 接收看门狗溢出中断有效 注: 该位在 <i>AIE</i> 位置 1 后有效</p>
8	RSE	rw	0x0	<p>0: 接收停止中断无效 1: 接收停止中断有效 注: 该位在 <i>AIE</i> 位置 1 后有效</p>
7	RUE	rw	0x0	<p>0: 接收缓冲不可用中断无效 1: 接收缓冲不可用中断有效 注: 该位在 <i>AIE</i> 位置 1 后有效</p>
6	RIE	rw	0x0	<p>0: 接收中断无效 1: 接收中断有效 注: 该位在 <i>NIE</i> 位置 1 后有效</p>
5	UNE	rw	0x0	<p>0: 发送下溢中断无效 1: 发送下溢中断有效 注: 该位在 <i>AIE</i> 位置 1 后有效</p>
4	OVE	rw	0x0	<p>0: 接收上溢中断无效 1: 接收上溢中断有效 注: 该位在 <i>AIE</i> 位置 1 后有效</p>

Bit	Field	Type	Reset	Description
3	TJE	rw	0x0	0: 发送 Jabber 超时中断无效 1: 发送 Jabber 超时中断有效 注: 该位在 AIE 位置 1 后有效
2	TUE	rw	0x0	0: 发送缓冲不可用中断无效 1: 发送缓冲不可用中断有效 注: 该位在 NIE 位置 1 后有效
1	TSE	rw	0x0	0: 发送停止中断无效 1: 发送停止中断有效 注: 该位在 AIE 位置 1 后有效
0	TIE	rw	0x0	0: 发送中断无效 1: 发送中断有效 注: 该位在 NIE 位置 1 后有效

#### 24.5.2.4 DMA 状态寄存器 (ETH\_DMASR)

偏移地址: 0x1014

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	Res.						EBUS[2:0]			TPS[2:0]			RPS[2:0]			NIS
Type							r			r			r			rc_w1
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	AIS	ERS	FBS	Res.	Res.	ETS	RWS	RSS	RUS	RIS	UNS	OVS	TJS	TUS	TSS	TIS
Type	rc_ w1	rc_ w1	rc_ w1			rc_ w1										

Bit	Field	Type	Reset	Description
31:26	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
25:23	EBUS	r	0x0	<p>该位指示导致总线错误（AHB 接口上的错误响应）的错误类型，不会产生中断</p> <p>000: RxDMA 写数据时发生错误 011: TxDMA 读数据时发生错误 100: RxDMA 写描述符时发生错误 101: TxDMA 写描述符时发生错误 110: RxDMA 读描述符时发生错误 111: TxDMA 读描述符时发生错误 请不要写入其它值</p> <p>注：该位仅在该寄存器的 <b>FBES</b> 位置 1 时有效</p>
22:20	TPS	r	0x0	<p>该位指示 TxDMA FSM 的状态，不会产生中断</p> <p>000: 停止，发出复位或停止发送命令 001: 运行中，正在获取发送传输描述符 010: 运行中，正在等待状态 011: 运行中，正在读取主机存储器缓冲区中的数据并将其加入发送缓冲区（TxFIFO）队列 100: 时间戳写入 101: Reserved 110: 挂起，发送描述符不可用或发送缓冲区下溢 111: 运行中，正在关闭发送描述符</p>
19:17	RPS	r	0x0	<p>位指示 RxDMA FSM 的状态，不会产生中断。</p> <p>000: 停止，发出复位或停止接收命令 001: 运行中，正在获取接收传输描述符 010: Reserved 011: 运行中，正在等待接收数据包 100: 挂起，接收描述符不可用 101: 运行中，正在关闭接收描述符 110: 时间戳写入 111: 运行中，将接收数据包的数据从接收缓冲区传输到主机存储器</p>

Bit	Field	Type	Reset	Description
16	NIS	rc_w1	0x0	<p>下列任何一个条件满足，该位就被置 1：</p> <ul style="list-style-type: none"> <li>1) TIS=1 &amp;&amp; ETH_DMAIR.TIE=1</li> <li>2) TUS=1 &amp;&amp; ETH_DMAIR.TUE=1</li> <li>3) RIS=1 &amp;&amp; ETH_DMAIR.RIE=1</li> <li>4) ERS=1 &amp;&amp; ETH_DMAIR.ERE=1</li> </ul> <p>注 1：只有未屏蔽的中断使能位会影响该普通中断状态汇总位</p> <p>注 2：每当导致该位置 1 的对应位被清零时，该位也必须清零（通过向此位写入 1）</p>
15	AIS	rc_w1	0x0	<p>下列任何一个条件满足，该位就被置 1：</p> <ul style="list-style-type: none"> <li>1) TSS=1 &amp;&amp; ETH_DMAIR.TSE=1</li> <li>2) TJS=1 &amp;&amp; ETH_DMAIR.TJE=1</li> <li>3) OVS=1 &amp;&amp; ETH_DMAIR.OVE=1</li> <li>4) UNS=1 &amp;&amp; ETH_DMAIR.UNE=1</li> <li>5) RUS=1 &amp;&amp; ETH_DMAIR.RUE=1</li> <li>6) RSS=1 &amp;&amp; ETH_DMAIR.RSE=1</li> <li>7) RWS=1 &amp;&amp; ETH_DMAIR.RWE=1</li> <li>8) ETS=1 &amp;&amp; ETH_DMAIR.ETE=1</li> <li>9) FBS=1 &amp;&amp; ETH_DMAIR.FBE=1</li> </ul> <p>注 1：只有未屏蔽的中断使能位会影响该异常中断状态汇总位</p> <p>注 2：每当导致该位置 1 的对应位被清零时，该位也必须清零（通过向此位写入 1）</p>
14	ERS	rc_w1	0x0	该位指示 DMA 已填满数据包的首个数据缓冲区
13	FBS	rc_w1	0x0	该位指示发生了总线错误，具体错误类型参见该寄存器的 EBUS 位，当该位置 1 后，对应的 DMA 引擎会禁止其所有的总线访问
12:11	Reserved			保留，始终读为 0

Bit	Field	Type	Reset	Description
10	ETS	rc_w1	0x0	该位指示要发送的帧已完全传输到 TxFIFO
9	RWS	rc_w1	0x0	当接收到的帧的长度大于 2048 个字节时，该位置 1
8	RSS	rc_w1	0x0	当接收过程进入停止状态时，该位置 1
7	RUS	rc_w1	0x0	<p>该位指示接收列表中的下一个描述符由主机所拥有，DMA 无法获取，接收过程进入挂起状态</p> <p>要恢复处理接收描述符，主机应更改描述符的拥有关系，然后发出接收轮询要求命令</p> <p>如果未发出接收轮询要求命令，则当接收到下一个识别的输入帧时，接收过程会恢复</p> <p>仅当上一个接收描述符由 DMA 所拥有时，该位置 1</p>
6	RIS	rc_w1	0x0	该位指示帧接收已完成，特定的帧状态信息已发布在描述符中，接收保持运行状态
5	UNS	rc_w1	0x0	该位指示在帧发送期间，发送缓冲区发生下溢，发送会进入挂起状态，且描述符中下溢错误 TDES0[1]标志位置 1
4	OVS	rc_w1	0x0	该位指示在帧接收期间，接收缓冲区发生上溢，如果部分帧已传输到应用，则描述符中上溢错误 RDES0[11]位置 1
3	TJS	rc_w1	0x0	该位指示发送 Jabber 定时器已过期，这意味着发送器过度有效，发送过程会中止并将其置于停止状态，且描述符中发送 Jabber 超时 TDES0[14]标志位置 1

Bit	Field	Type	Reset	Description
2	TUS	rc_w1	0x0	<p>该位指示发送列表中的下一个描述符由主机所拥有, DMA 无法获取, 发送会进入挂起状态</p> <p>要恢复处理发送描述符, 主机应更改描述符的拥有关系, 然后发出发送轮询要求命令</p>
1	TSS	rc_w1	0x0	当发送过程进入停止状态时, 该位置 1
0	TIS	rc_w1	0x0	该位指示帧发送已完成, 特定的帧状态信息已发布在描述符中

#### 24.5.2.5 DMA 帧丢失统计寄存器 (ETH\_DMAFLCR)

偏移地址: 0x1020

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	Res.			OVFF	OVFC[10:0]							BNAF				
Type				rw	rw							rw				
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	BNAC[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:29	Reserved			保留, 始终读为 0
28	OVFF	rw	0x0	<p>该寄存器的 OVFC 位统计丢失帧计数器上溢时, 该位置 1</p> <p>注: 该位在读出该寄存器 OVFC 位后自动清零</p>
27:17	OVFC	rw	0x00	<p>该位指示因 Rx FIFO 上溢情况以及帧长度过短(不足 64 字节的好帧)而丢失的帧数量</p> <p>注: 该位在读出后自动清零</p>

Bit	Field	Type	Reset	Description
16	BNAF	rw	0x0	该寄存器的 BNAC 位统计丢失帧计数器上溢时，该位置 1 注：该位在读出该寄存器 BNAC 位后自动清零
15:0	BNAC	rw	0x00	该位指示因缓冲区不可用（无可用的接收描述符）而丢失的帧数量 注：该位在读出后自动清零

#### 24.5.2.6 DMA 看门狗定时寄存器 (ETH\_DMAWDTR)

偏移地址：0x1024

复位值：0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	Res.															
Type																
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	Res.								RIWT[7:0]							
Type									rw							

Bit	Field	Type	Reset	Description
31:8	Reserved			保留，始终读为 0
7:0	RIWT	rw	0x00	该位指示系统时钟周期数乘以 256 之后的时间值，即看门狗定时器的设定时间值 看门狗定时器会在 RxDMA 结束帧传输之后由编程设定的值触发，此时，动作状态寄存器 ETH_DMASR 的 RIS 位因相对应的描述符中 RDES1.RIS 位置 1 而未被置 1，直到看门狗定时器计时结束时，RIS 位才置 1 且看门狗定时器停止。当动作状态寄存器 ETH_DMASR 的 RIS 位设为高电平时，看门狗定时器复位

#### 24.5.2.7 DMA 接收轮询寄存器 (ETH\_DMARXPDR)

偏移地址：0x1008

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	RXPD[31:16]															
Type	rw															
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	RXPD[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:0	RXPD	rw	0x00	<p>向该位写入任何值时, DMA 都会读取 ETH_DMARXDSAR 寄存器指向的当前描述符, 如果该描述符不可用(由主机所有), 则 RxDMA 会返回到挂起状态, 且不会将 ETH_DMASR 寄存器 RUS 位进行置位; 如果该描述符可用, 则 RxDMA 将返回到活动状态</p> <p>使用该寄存器来指示 RxDMA 轮询接收描述符列表, 检查新描述符, 用于将 RxDMA 从挂起状态唤醒, 仅当 RxDMA 所拥有的描述符不可用时, 它才会进入挂起状态</p> <p>注: 该位在读出时始终读出零</p>

#### 24.5.2.8 DMA 发送轮询寄存器 (ETH\_DMATXPDR)

偏移地址: 0x1004

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	TXPD[31:16]															
Type	rw															
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	TXPD[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:0	TXPD	rw	0x00	<p>向该位写入任何值时，DMA 都会读取 ETH_DMATXDSAR 寄存器指向的当前描述符，如果该描述符不可用（由主机所有），则 TxDMA 会返回到挂起状态，并将 ETH_DMASR 寄存器 TUS 位进行置位；如果该描述符可用，则发送会继续进行</p> <p>使用该寄存器来指示 TxDMA 轮询发送描述符列表，检查当前描述符是否为 DMA 所有。如果 TxDMA 处于挂起模式，则发出发送轮询要求命令将其唤醒；如果发送帧中出现下溢错误或 TxDMA 所拥有的描述符不可用，则 TxDMA 会进入挂起模式</p> <p>用户可以随时发出此命令，当该命令开始重新获取主机存储器的当前描述符后，TxDMA 即会对其进行复位</p> <p>注：该位在读出时始终读出零</p>

#### 24.5.2.9 DMA 接收描述符地址寄存器 (ETH\_DMARXDSAR)

偏移地址：0x100C

复位值：0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	RXDSA[31:16]															
Type	rw															
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	RXDSA[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:0	RXDSA	rw	0x00	<p>该位表示接收描述符列表中的首个描述符的地址</p> <p>注：最低 2 位在读出时始终读出零</p>

#### 24.5.2.10 DMA 发送描述符地址寄存器 (ETH\_DMATXDSAR)

偏移地址：0x1010

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	TXDSA[31:16]															
Type	rw															
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	TXDSA[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:0	TXDSA	rw	0x00	该位表示发送描述符列表中的首个描述符的地址 注: 最低 2 位在读出时始终读出零

#### 24.5.2.11 DMA 当前接收描述符地址寄存器 (ETH\_DMACUTRXDSAR)

偏移地址: 0x104C

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	CUTRXDSA[31:16]															
Type	rw															
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	CUTRXDSA[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:0	CUTRXDSA	rw	0x00	该位指向 DMA 所读取的当前接收描述符的起始地址

#### 24.5.2.12 DMA 当前发送描述符地址寄存器 (ETH\_DMACUTTXDSAR)

偏移地址: 0x1048

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	CUTTXDSA[31:16]															
Type	rw															
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	CUTTXDSA[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:0	CUTTXDSA	rw	0x00	该位指向 DMA 所读取的当前发送描述符的起始地址

#### 24.5.2.13 DMA 当前接收缓冲区地址寄存器 (ETH\_DMACUTRXBUFR)

偏移地址: 0x1054

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	CUTRXBUF[31:16]															
Type	rw															
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	CUTRXBUF[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:0	CUTRXBUF	rw	0x00	该位指向 DMA 所读取的当前接收缓冲区地址

#### 24.5.2.14 DMA 当前发送缓冲区地址寄存器 (ETH\_DMACUTTXBUFR)

偏移地址: 0x1050

复位值: 0x0000\_0000

Bit	b31	b30	b29	b28	b27	b26	b25	b24	b23	b22	b21	b20	b19	b18	b17	b16
Field	CUTTXBUF[31:16]															
Type	rw															
Bit	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
Field	CUTTXBUF[15:0]															
Type	rw															

Bit	Field	Type	Reset	Description
31:0	CUTTXBUF	rw	0x00	该位指向 DMA 所读取的当前发送缓冲区地址

## 25 安全数字输入输出接口（SDIO）

### 25.1 SDIO 简介

SD/MMC/SDIO 控制器是 AMBA AHB 从外设，用于控制外部 SD/MMC/SDIO 卡，并支持 DSP/MCU 的读/写访问。它作为主机与连接的 SD/MMC/SDIO 卡进行通信。

所述控制器是基于 AMBA HCLK 域的完全同步设计。

## 25.1.1 SDIO 功能框图

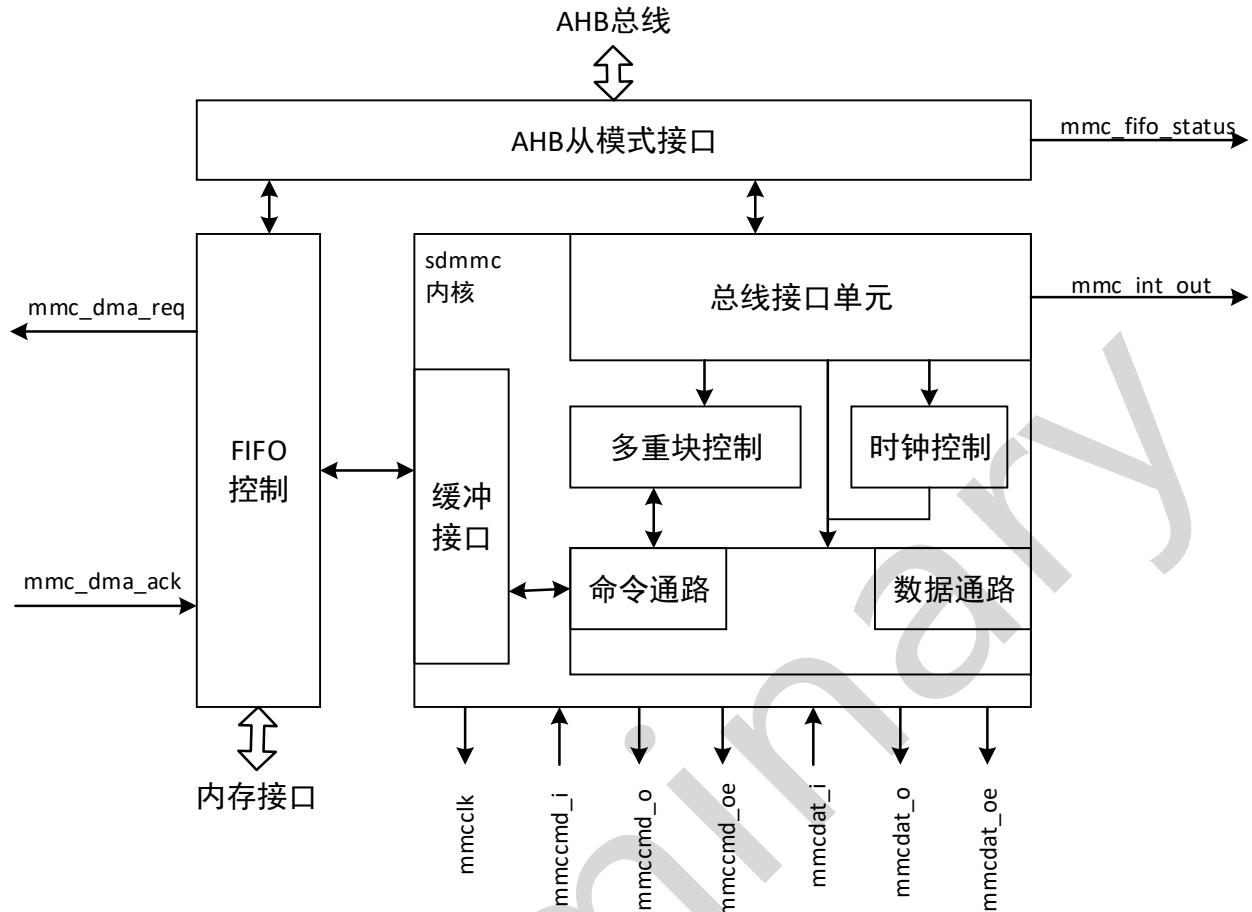


图 25-1 SDIO 框图

**AHB 从模式接口**: 为 32 位 AHB 总线提供接口。

**FIFO 控制**: 产生握手信号到 DMA 硬件接口，并控制对外部数据 FIFO (128x32) 的读/写访问。

**总线接口单元**: 包括控制寄存器和命令缓冲单元。

**多重块控制**: 控制多块数据的读写。

**时钟控制**: 通用时钟基于寄存器中定义的分频值。

**命令通路**: 从总线接口单元或 irq 响应中加载新的命令，然后发送命令，并接收响应 crc7 检查和 8 个空时钟。

**数据通路**: 发送和接收数据，用 crc16 检查。

**缓冲接口**: 控制对数据 FIFO 的读写控制信号。

## 25.2 功能描述

- 1) 完全兼容 SD 记忆卡规格 1.0
- 2) 完全兼容 SD 存储卡规格 1.1 (高速)
- 3) 完全兼容 SD 记忆卡规格 2.0 (SDHC)

- 
- 4) 完全兼容 MMC 系统规格 2.0~4.2
  - 5) 完全兼容 SDIO 存储卡规格 1.1.0
  - 6) 标准的 MMC 模式接口支持
  - 7) 可编程时钟速率
  - 8) 自动命令/响应 CRC 生成/检查
  - 9) 自动数据 CRC 生成/检查
  - 10) 可编程超时检测
  - 11) AMBA 2.0 32 位 AHB 接口
  - 12) 用于 AHB 数据访问的总共外部 128\*32 数据 FIFO
  - 13) 32 位 DMA 硬件接口，用于更快的 DMA 访问
  - 14) DMA 接口可以配置为启用/禁用
  - 15) DMA 请求是可配置的
  - 16) 组合中断输出

## 25.3 SD 存储卡系统概念

### 25.3.1 Read-Write 属性

根据 Read-Write 属性不同可分为两种类型的 SD 存储卡：

- 1) 可读可写 (RW) 卡 (FLASH, OTP, MTP)。这些卡通常作为空白媒介来售卖，用于海量终端用户的视频、音频或数字图像记录的存储。
- 2) 只读存储卡 (ROM)。这些卡是用固定的数据内容生产出来的，它们通常用作软件、音频、视频等的传播媒介。

### 25.3.2 电源电压

根据工作电源电压不同可分为两种类型的 SD 存储卡：

- 1) 高电压 SD 存储卡，可以在 2.7-3.6V 的电压范围内工作；
- 2) 双电压 SD 存储卡，可以在 1.6-3.6V 的电压范围内工作。

### 25.3.3 卡容量

根据卡容量不同可分为两种类型的 SD 存储卡：

- 1) 标准容量的 SD 存储卡支持最大 2G 字节的容量。所有版本的物理规格都定义了标准容量 SD 存储卡；
- 2) 高容量 SD 存储卡支持超过 2G 字节的容量，此版本规范限制容量高达 32GB。高容量 SD 存储卡是物理层规范版本 2.00 中新定义的。

### 25.3.4 传输速度

定义了四种速度等级，表示 SD 卡的最低性能：

- 1) Class0：这类卡不指定性能；
- 2) Class2：速度不低于 2MB/s；
- 3) Class4：速度不低于 4MB/s；
- 4) Class6：速度不低于 6MB/s；

大容量 SD 存储卡应支持速度等级规范，性能大于或等于 2 级。

### 25.3.5 总线拓扑

SD 卡系统定义了两种通信协议：SD 和 SPI。

主机系统可以选择任意一种。当收到 reset 命令的时候，SD 卡通过主机的信息来决定使用何种模式，并且之后的通讯都会使用相同模式。不推荐多卡槽用共同的总线信号。一个单独的 SD 总线应该连接一个单独的 SD 卡。SD 总线包含下面的信号：

- 1) CLK: 时钟信号；
- 2) CMD: 双向命令/响应信号；
- 3) DAT0-DAT3: 双向数据信号；
- 4) Vdd, Vss1, Vss2: 电源和地信号。

### 25.3.6 总线协议

SD 总线：

- 1) Command: 命令是一次操作开始的令牌，从主机发送到一个卡片（编址命令）或者连接到主机的所有卡片（广播命令）。命令在 CMD 线上连续传输。
- 2) Response: 响应是从已寻址的卡或从所有连接上的卡发送到主机的令牌，作为对先前接收到指令的应答。响应在 CMD 线上连续传输。
- 3) Data: 数据可以通过 DATA 线双向传输。

卡片寻址通过使用会话地址来实现，会话地址会在初始化阶段分配给卡。SD 总线上的基本交互是命令/响应交互。这种总线交互直接在命令或者响应的结构里面传输他们的信息。此外，某些操作还有数据令牌。SD 卡发送或接收的数据在块（block）中完成。数据块以 CRC 位来保证传输成功。目前有单块和多块操作。

注：多块操作模式在快速写操作时更好一点。多块传输在 CMD 线上产生 stop 命令时结束。主机端可以配置数据传输是单线还是多线。

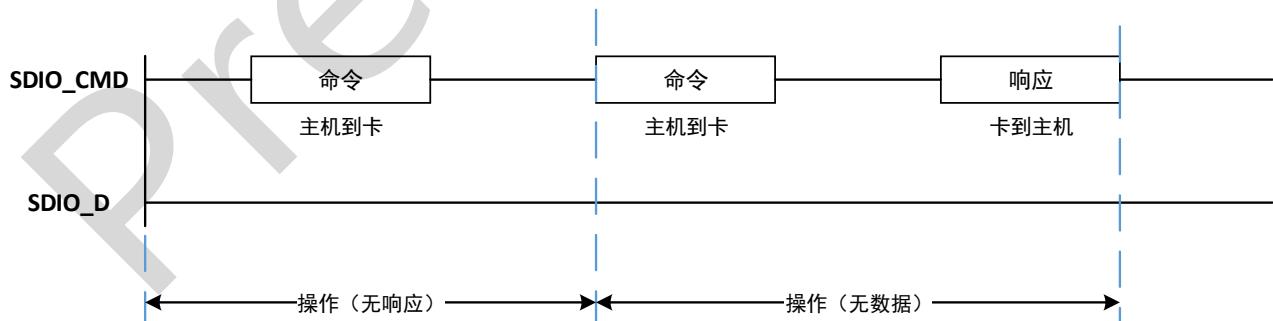
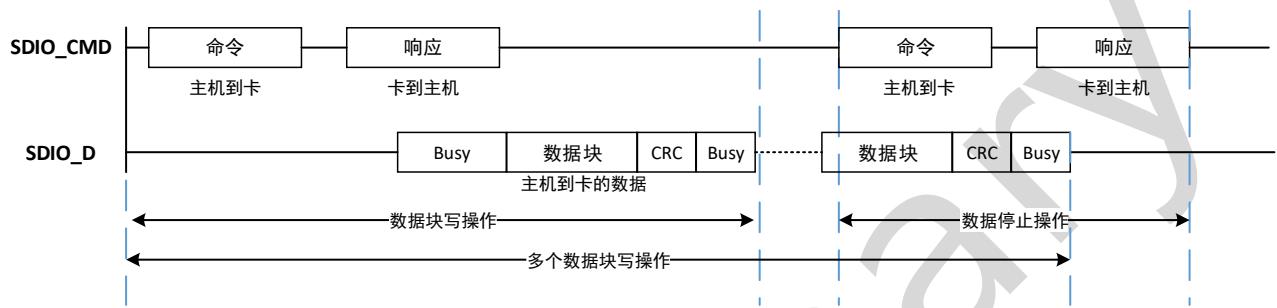
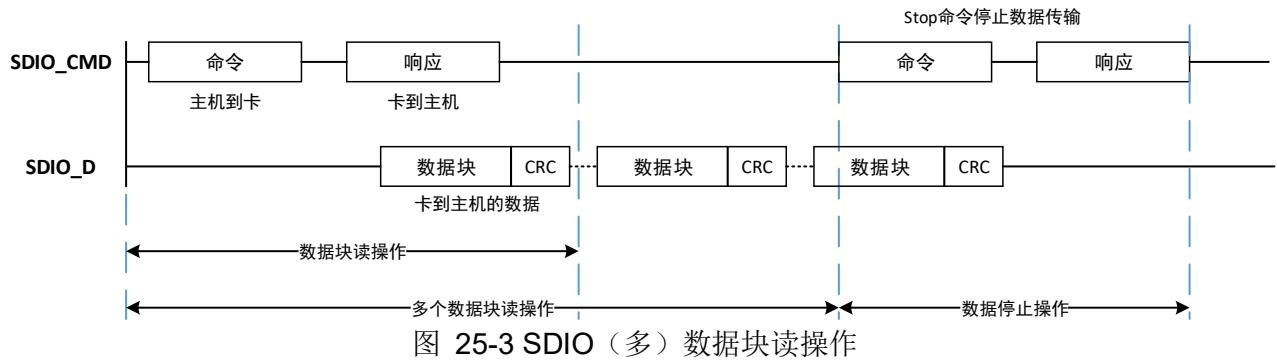
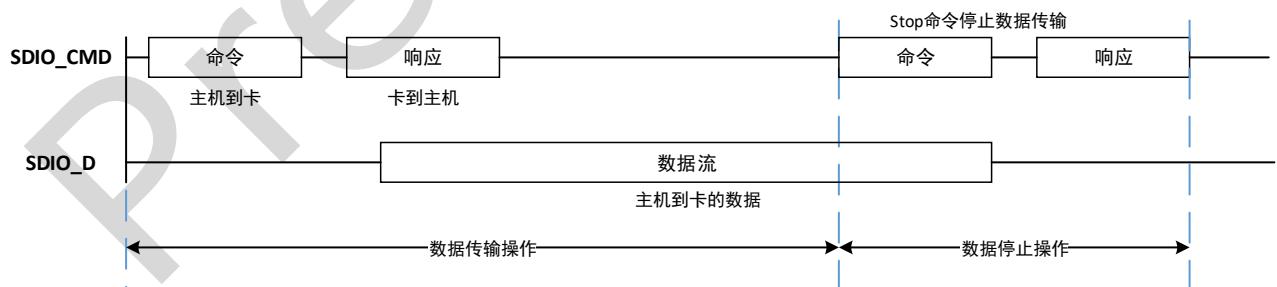


图 25-2 SDIO “无响应”和“无数据”操作



注：当有 Busy 信号时，SDIO DAT0 被拉低，SDIO 将不会发送任何数据。



## 25.4 SD 存储卡功能描述

### 25.4.1 简介

主机和卡之间的交互都是主机控制的。主机发送两种命令：广播命令，寻址（点对点）命令。

- 1) 广播命令用于所有的卡，部分命令需要响应。
- 2) 寻址命令是发送给对应地址的卡的，并且会引起这张卡的响应。

---

SD 卡系统定义了两种操作模式：卡识别模式和数据传输模式。

## 25.4.2 卡识别模式

在卡识别模式下，主机复位所有处于该模式的卡，确认工作电压范围，识别卡，并且要求这些卡发布相对卡地址（RCA）。在卡识别模式下，所有数据通信都只使用命令信号线（CMD）。

### 25.4.2.1 卡复位

`GO_IDLE_STATE` 命令（CMD0）是一个软件复位命令，无论卡当前是什么状态，都将其设为空闲（Idle）状态。主机上电或执行 CMD0 后，所有卡的 CMD 线都为输入模式，等待下个命令的起始位。卡初始化的时候，会有一个默认的相对卡地址（RCA=0x0000）和默认的驱动级寄存器设置（最低的速度，最大的电流驱动能力）。

### 25.4.2.2 工作条件确认

主机首先会发送 CMD8 去确认卡片的工作电压。`SEND_IF_COND`（CMD8）用于验证 SD 卡接口操作条件。如果卡片不支持当前电压，则不会返回并保持在空闲状态；如果卡片能够在当前电压工作，则会返回卡片的支持电压以及检测模式，还有相应的 CRC 检验码。`SD_SEND_OP_COND`（ACMD41）是用来提供给主机一种机制来识别或拒绝那些不匹配它期望的 Vdd 范围的卡。不能支持指定电压的卡自动放弃后续总线操作，并且进入无效（Inactive）状态。通过设置 ACMD41 的参数里面 OCR（工作电压范围）的值为 0，主机可以查询每张卡，并且确定通用电压范围，进而使超出范围的卡进入 Inactive 状态。这种查询模式在需要选择通用电压范围或得到应用中不可用卡信息时使用。如果 ACMD41 是作为查询命令发送，卡无法启动初始化。之后，主机要再选择一个工作电压，重新发送 ACMD41。在初始化过程中，主机不可以改变工作电压。

### 25.4.2.3 卡的初始化以及识别过程

初始化进程以命令 ACMD41 作为开始，通过设置工作条件和 OCR 来进行。HCS（HighCapacitySupport）位为 1 表示主机支持高容量 SD 卡。卡通过 OCR 的 busy 位来通知主机 ACMD41 的初始化完成了。busy 位为 0 表示卡仍然在初始化；为 1 表示已经完成初始化。主机会重复发送 ACMD41，直到 busy 位被置 1。卡片旨在第一个 ACMD41 的命令时，检查工作条件和 OCR 里面的 HCS 位。当重复 ACMD41 的时候，除了 CMD0，主机不再发其他命令。接着主机会发送命令 `ALL_SEND_CID`（CMD2），来获得卡的 CID 号。未识别的卡（处于 Ready 状态的）发送自己的 CID 作为响应。当卡发送 CID 后，进入卡识别（Identification）状态。之后主机发送 `SEND_RELATIVE_ADDR`（CMD3）命令要求卡发布新的相对地址（RCA），一旦收到 RCA，卡就会变为等待（Stand-by）状态。主机会重复识别进程，为系统中每个卡循环发送 CMD2 和 CMD3。对于 SDI/O 卡而言，总线被激活后 SDIO 卡主机先发送 `IO_SEND_OP_COND`（CMD5）命令，得到的响应是卡的工作条件寄存器的内容，之后再同上发送 CMD3 命令，执行后续操作。

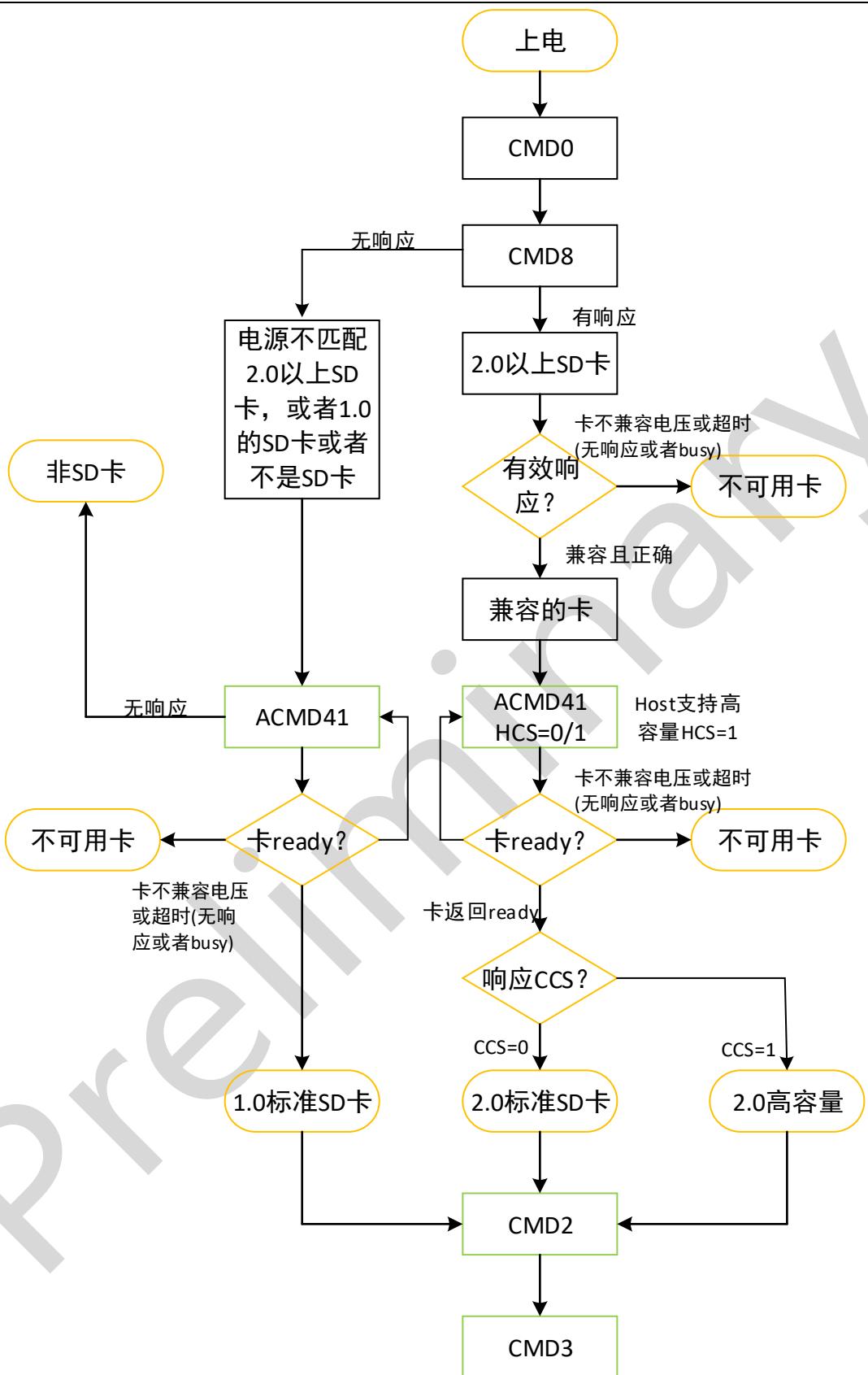


图 25-7 SDIO 流程图

### 25.4.3 数据传输模式

CMD7 用于选择一张卡进入传输模式，每次只能有一张卡处于传输模式。当主机发送 CMD7，RCA=0x0000 时，所有卡都会回到等待状态。如果新的 CMD7 命令中的 RCA 不是当前卡片的，则会释放连接并且当前卡就会回到等待状态，如果同一条 CMD 线上有多个卡，则会自动去匹配新卡。

各种数据传输模式的关系总结如下：

- 1) 所有读取数据的命令在任何时间都能被 CMD12 终止，同时卡回到传输状态。读取数据的命令包括：读取单块数据（CMD17）、读取多块数据（CMD18）、读取写保护状态位（CMD30）、读取 SCR（ACMD51）、读模式通用命令（ACMD56）。
- 2) 所有写入数据的命令在任何时间也能被 CMD12 终止，在发送 CMD7 取消选定卡之前，应该先停止写命令。写入数据的命令包括：写单块数据（CMD24）、写多块数据（CMD25）、设置 CSD 寄存器（CMD27）、加锁解锁（CMD42）、写模式通用命令（ACMD56）。
- 3) 一旦数据传输完成，卡会推出写状态，并且进入编程状态（传输成功）或传输状态（传输失败）。
- 4) 如果写操作停止，并且块长度和最后一个块的 CRC 校验码是有效的，那么数据会被写入存储介质。
- 5) 卡支持写缓存，这意味着当前块在进行编程操作时下一块也可以进行传输。如果写缓存满了，卡片还在编程状态，则卡片会将 DAT0 一直拉低（busy）。
- 6) 写 CSD，写保护和擦除没有缓存操作，这意味着执行这些处理时，其他传输命令都不接受。DAT0 也会保持低电平，保持编程状态。
- 7) 当卡正在处理命令/数据（编程状态）时，读命令和参数设置命令将不被执行。参数设置命令包括：设置块长度（CMD16）、设置擦除起始块数（CMD32）、设置擦除结束块数（CMD33）。
- 8) CMD7 不会终止擦除和编程操作。
- 9) 处于断开连接状态的卡可以通过 CMD7 重新被选定。这是，卡会进入编程模式，并重新使能 busy 标记。
- 10) 复位卡（CMD0 和 CMD15）会终止任何等候或执行的编程操作，这可能会损坏卡的内容，

#### 25.4.3.1 读数据

当总线上没有数据传输时，DAT 电平被拉高。一个传输的数据块包含了起始位（1 位或 4 位，低电平）和后面连续的数据流。数据流包含了有效数据和 ECC 值。数据流之后是结束位（1 位或 4 位，高电平）。数据传输与时钟信号同步。

块读：块读，是以块为单位的数据传输。数据传输的基本单元是一个块（block），最大为 512 字节。

每个块的后面都会有一个 CRC 值，用来确保数据传输的有效性。READ\_SINGLE\_BLOCK（CMD17）代表一个块的内容，传输结束后，回到传输状态。READ\_MULTIPLE\_BLOCK（CMD18）代表读取多个连续的块。块会连续的传输，直到 STOP\_TRANSMISSION（CMD12）命令发出。因为连续数据传输，停止命令会有些许执行延迟。

#### 25.4.3.2 写数据

写数据流程与读数据流程类似。CRC 检查位在每一个数据块之后。在写之前，卡会检测每一个收到

的数据块的 CRC 值。这样传输数据的错误可以避免。如果发生 BLOCK\_LEN\_ERROR 和 ADDRESS\_ERROR 的错误，写命令就无法执行了。

块写：在块写（CMD24-27, 42, 56）期间，一个或多个数据块从主机传递到卡，每个块都有 CRC。

### 25.4.3.3 宽总线选择和解除选择

宽总线（4Bit 宽）操作模式可以通过命令 ACMD6 来选定和取消。上电或者 GO\_IDLE（CMD0）命令后，默认总线宽度是 1Bit。如果想要改变总线宽度，需要具备以下两个条件：

- 1) 卡处于传输状态；
- 2) 卡没有被锁定（锁定的卡会认为 ACMD6 是无效命令）。

### 25.4.3.4 擦除

擦除数据的流程和读取多块数据的流程类似。CMD32 命令指定擦除的开始块，CMD33 命令指定擦除的结束块，CMD38 命令启动擦除。如果将要擦除的块是写保护的则跳过。在擦除过程中 DATA0 保持低电平。在写或者擦除的过程中可以通过 CMD7 命令去操作别的卡片。

### 25.4.3.5 写保护管理

SD 卡支持三种写保护方法：

- 1) 物理写保护开关（主机支持）；
- 2) 卡内部写保护（卡支持）；
- 3) 密码保护卡锁定操作。

### 25.4.3.6 卡锁定/解锁操作

SD 存储卡支持加锁功能，密码以及长度保存在 128bits 的 PWD 和 8bits 的 PWD\_LEN 寄存器。被加锁的卡片可以响应 class0、CMD16、ACMD41、LOCKCARD（class7）命令，但是不能访问数据。如果卡片被加锁，则 PWD\_LEN 不为 0，卡片在上电后就会进入锁定状态。加锁解锁的命令为 CMD42。

## 25.5 寄存器描述

表 25-1 SDIO 寄存器概览

Offset	Acronym	RegisterName	Reset
0x00	MMC_CTRL	MMC_CTRL	0x00000045
0x04	MMC_IO	MMC_IO	0x00000000
0x08	MMC_BYTECNTL	MMC_BYTECNTL	0x00000200
0x0C	MMC_TR_BLOCKCNT	MMC_TR_BLOCKCNT	0x00000000
0x10	MMC_CRCCTL	MMC_CRCCTL	0x00000000
0x14	CMD_CRC	CMD_CRC	0x00000000
0x18	DAT_CRCL	DAT_CRCL	0x00000000
0x1C	DAT_CRCH	DAT_CRCH	0x00000000

Offset	Acronym	RegisterName	Reset
0x20	MMC_PORT	MMC_PORT	0x00000007F
0x24	MMC_INT_MASK	MMC_INT_MASK	0x000000000
0x28	CLR_MMC_INT	CLR_MMC_INT	0x000000000
0x2C	MMC_CARDSEL	MMC_CARDSEL	0x000000040
0x30	MMC_SIQ	MMC_SIQ	0x0000000FF
0x34	MMC_IO_MBCTL	MMC_IO_MBCTL	0x000000010
0x38	MMC_BLOCKCNT	MMC_BLOCKCNT	0x000000001
0x3C	MMC_TIMEOUTCNT	MMC_TIMEOUTCN	0x000000040
0x40-0x7C	CMD_BUFX (0..15)	CMD_BUFX (0..15)	0x000000000
0x80	BUF_CTL	BUF_CTL	0x000000002
0x100-0x2FF	DATA_BUF	DATA_BUF	0x000000000

## 25.5.1 MMC\_CTRL

偏移地址: 0x00

复位值: 0x000000045

Bit	31	30	29	28	27	26	25	24	23	22	2 1	20	19	18	17	16	
Field	Res.																
Bit	15	14	13	12	11	10	9	8	7	6	5 4	3	2	1	0		
Field	Res.						RDW TEN	INT EN	MD EN	DAT WT	SelPT SM	CLKSP			OU TM	Sel SM	OPM Sel
Type					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
31:11	Reserved			始终读为 0。
10	RDWTEN	rw	0x0	SDIO 读等待使能信号 (SDIO read wait enable signal) 1: SDIO 读等待使能 0: SDIO 读等待禁用
9	INTEN	rw	0x0	SDIO 中断使能信号 (SDIO interrupt enable signal) 1: SDIO 中断使能 0: SDIO 中断禁用
8	MDEN	rw	0x0	SDIO 模式使能 (SDIO mode enable) 1: SDIO

Bit	Field	Type	Reset	Description
				0: SD/MMC
7	DATWT	rw	0x0	<p>定义 SD/MMC/SDIO 端口 DAT 线的总线宽度 (Define the bus width of SD/MMC/SDIO port DAT line)</p> <p>1: 4 位</p> <p>0: 1 位</p>
6	SelPTSM	rw	0x1	<p>选择 SD/MMC/SDIO 端口传输速度模式 (Select SD/MMC/SDIO port transfer speed mode)</p> <p>1: SD/MMC/SDIO 端口高速传输模式</p> <p>0: SD/MMC/SDIO 端口低速传输模式</p> <p>注:</p> <p>当高速传输模式时, 基准时钟为频率 hclk;</p> <p>当低速传输模式时, 基准时钟是 clk1m 的频率。</p> <p>1MHz=Fhclk/ ((mmc_cardsel[5:0]+1) *2)</p>
5:3	CLKSP	rw	0x0	<p>SD/MMC/SDIO 端口 CLK 线速度选择位 (SD/MMC/SDIO port CLK line speed selection)</p> <p>000: 1/2 基础时钟</p> <p>001: 1/4 基础时钟</p> <p>010: 1/6 基础时钟</p> <p>011: 1/8 基础时钟</p> <p>100: 1/10 基础时钟</p> <p>101: 1/12 基础时钟</p> <p>110: 1/14 基础时钟</p> <p>111: 1/16 基础时钟</p> <p>注: 是为 SDIO/MMC 端口速度选择而定义的。</p>
2	OUTM	rw	0x1	<p>SD/MMC/SDIO 端口 CMD 输出驱动选择位 (SD/MMC/SDIO port CMD line output driver mode selection)</p> <p>1: 开漏输出</p> <p>0: 推挽输出</p>
1	SelSM	rw	0x0	<p>选择信号模式位 (Select Signal mode)</p> <p>1: SD/MMC/SDIO 端口自动传输模式</p> <p>0: SD/MMC/SDIO 端口使用 mmc_port 寄存器</p>
0	OPMSel	rw	0x1	<p>SD/MMC/SDIO 端口操作模式选择位 (SD/MMC/SDIO port operation mode select)</p> <p>1: SD/MMC/SDIO 模式</p>

Bit	Field	Type	Reset	Description
				0: SPI 模式

## 25.5.2 MMC\_IO

偏移地址: 0x04

复位值: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.						CMD AF	CMD CH	AUT OC LKG	ENRR ESP	PCL KG	CID/ CSD RD	RES PC MD SEL	AU TO TR	TR AN SF DIR	AUTO DATT R
Type					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:10	Reserved			始终读为 0。
9	CMDAF	rw	0x0	SDIO cmd12/IO 中止标志 1: 标记当前的命令是 cmd12/IO 中止命令 0: 标记当前命令不是 cmd12/IO 中止命令
8	CMDCH	rw	0x0	SDIO 命令特征位 1: 标志当前的命令后跟数据块 0: 标志当前命令后面既不是数据块也不是响应
7	AUTOCLKG	rw	0x0	在响应/命令或单个数据块后启用自动转换 8 空时钟 1: 启用 0: 禁用
6	ENRRESP	rw	0x0	在命令后启用自动接收响应 1: 启用 0: 禁用
5	PCLKG	rw	0x0	SD/MMC/SDIO 端口 CLK 线 8 个空时钟产生 1: 8 个空时钟产生 0: 通过位[3]选择接收响应/发送命令
4	CID/CSDRD	rw	0x0	这是为 CID 和 CSD 读取而设计的。发出读取 CID 或 CSD 命令时，SD/MMC/SDIO 卡将响应 CMD 线中的 136 位 CID 或 CSD 数据。

Bit	Field	Type	Reset	Description
				将该位置 1， 将得到命令缓冲区[135: 8]中的 CID 或 CSD 数据。
3	RESPCMDSEL	rw	0x0	位[5]为“0” 时的响应/命令选择 1: 接收回响。 0: 发送命令。
2	AUTOTR	rw	0x0	设置自动 8 位/命令/响应传输。 1: 启用自动 8null/命令/响应传输。 0: 禁用自动 8null/命令/响应传输。 根据[5]和[3]位，生成 8 个空时钟/接收响应/发送命令，当传输完成后，该位将自动清除。
1	TRANSFDIR	rw	0x0	设置数据传输方向。 1: 读取数据。 0: 写入数据。
0	AUTODATTR	rw	0x0	设置自动数据传输。 1: 启用自动数据传输。 0: 禁止自动数据传输。 数据传输完成后，该位将自动清零。 有关 SD/MMC/SDIO 操作控制的更多信息，请参考 MMC_IO 寄存器[7:0]的详细描述表。

### 25.5.3 MMC\_BYTECNTL

偏移地址: 0x08

复位值: 0x00000200

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

Bit	Field	Type	Reset	Description
31:16	Reserved			始终读为 0。
15:0	CNT	rw	0x0200	数据传输字节计数寄存器

### 25.5.4 MMC\_TR\_BLOCKCNT

偏移地址: 0x0C

复位值: 0x00000000

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

Bit	Field	Type	Reset	Description
31:16	Reserved			始终读为 0。
15:0	CNT	r	0x0	当多块传送时, 传输完成的计数器值。

## 25.5.5 MMC\_CRCCTL

偏移地址: 0x10

复位值: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								CMD_C RCEN	DAT_C RCEN	ENC HK	ENR DMB	DAT_C RCS		CMD_C CRCE	DAT_C CE
Type									rw	rw	rw	rw	rw	rw	r	r

Bit	Field	Type	Reset	Description
31:8	Reserved			始终读为 0。
7	CMD_CRCEN	rw	0x0	SD/MMC/SDIO 端口 CMD 线路 CRC 电路使能。 1: 启用。 0: 禁用。
6	DAT_CRCEN	rw	0x0	SD/MMC/SDIO 端口 DAT 线路 CRC 电路使能。 1: 启用。 0: 禁用。
5	ENCHK	rw	0x0	启用自动检查 crc_status[2: 0]。 1: 启用。 如果 crc_status[2: 0]! =3' b010, crcstatuserr 中断产生, 写数据传输将通过停止命令来终止并清除 MMC_IO[0]或 MMC_IO_MBCTL[2: 0]

Bit	Field	Type	Reset	Description
				0: 禁用。 1: 启用 0: 禁用
4	ENRDMB	rw	0x0	在响应之前启用读取多个块数据
3:2	DAT_CRCS	rw	0x0	DAT CRC 选择。 00: SD/MMC/SDIO DAT0 或 MMC DAT 线路 CRC 结果 01: SD/MMC/SDIO DAT1 线路 CRC 结果 10: SD/MMC/SDIO DAT2 线路 CRC 结果 11: SD/MMC/SDIO DAT3 线路 CRC 结果 注: 是为 DAT CRC 结果显示设置定义的
1	CMD_CRCE	r	0x0	CMD CRC 错误。只读。
0	DAT_CRCE	r	0x0	DAT CRC 错误。只读。

## 25.5.6 CMD\_CRC

偏移地址: 0x14

复位值: 0x00000000

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

Bit	Field	Type	Reset	Description
31:7	Reserved			始终读为 0。
6:0	CMD_CRCV	r	0x0	CMD CRC 寄存器值

## 25.5.7 DAT\_CRCL

偏移地址: 0x18

复位值: 0x00000000

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

Type		r	r	r	r	r	r	r
------	--	---	---	---	---	---	---	---

Bit	Field	Type	Reset	Description
31:7	Reserved			始终读为 0。
6:0	DAT_CRCLO	r	0x0	DAT CRC 低寄存器值

## 25.5.8 DAT\_CRCLO

偏移地址: 0x1C

复位值: 0x00000000

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

Bit	Field	Type	Reset	Description
31:7	Reserved			始终读为 0。
6:0	DAT_CRCCHV	r	0x0	DAT CRC 高寄存器值

## 25.5.9 MMC\_PORT

偏移地址: 0x20

复位值: 0x0000007F

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

Bit	Field	Type	Reset	Description
31:8	Reserved			始终读为 0。
7	PCLKS	rw	0x0	SD/MMC/SDIO 端口 CLK 线信号。
6	PCMDs	rw	0x1	SD/MMC/SDIO 端口 CMD 线信号
5	PDATs	rw	0x1	SD/MMC/SDIO 端口 DAT 线路信号。

Bit	Field	Type	Reset	Description
4	AUTONTEN	rw	0x1	自动 Ncr 定时器输出使能 1: 启用自动检查 Ncr 超时。 0: 禁用自动检查 Ncr 超时。
3:0	NTCR	rw	0xF	Ncr 超时计数寄存器 (SD/MMC/SDIO 时钟号)。

### 25.5.10 MMC\_INT\_MASK

偏移地址: 0x24

复位值: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Res.																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Res.								D1I	CR	CR	MB	MB	CM	DAT	DAT	CM
Type								NT	CIN	TIN	TIN	DIN	DEI	EIN	DIN	DDI	
								M	TM	TM	TM	TM	NT	T	T	NT	
Type								rw	rw	rw	rw	rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
31:9	Reserved			始终读为 0。
8	D1INTM	rw	0x0	SDIO 数据 1 线路中断屏蔽 (SDIO data1 line interrupt mask) 1:开放请求 0:屏蔽请求
7	CRCINTM	rw	0x0	CRC 状态令牌错误中断屏蔽 (CRC status token err interrupt mask) 1:开放请求 0:屏蔽请求
6	CRTINTM	rw	0x0	Cmd 和 Resp Ncr 超时中断屏蔽(Cmd and RespNcr Time out interrupt mask.) 1:开放请求 0:屏蔽请求
5	MBTINTM	rw	0x0	多块超时中断屏蔽 (Multi Block Timeout interrupt mask) 1:开放请求 0:屏蔽请求
4	MBDINTM	rw	0x0	多块完成中断屏蔽 (Multi Block done interrupt mask) 1:开放请求 0:屏蔽请求

Bit	Field	Type	Reset	Description
3	CMDEINT	rw	0x0	CMD CRC 错误中断屏蔽 (CMD CRC error interrupt mask) 1:开放请求 0:屏蔽请求
2	DATDINT	rw	0x0	DAT CRC 完成中断屏蔽 (DAT CRC error interrupt mask) 1:开放请求 0:屏蔽请求
1	CMDDINT	rw	0x0	DAT 完成中断屏蔽 (DAT done interrupt tmask) 1:开放请求 0:屏蔽请求
0	CMDDINT	rw	0x0	CMD 完成中断屏蔽 (CMD done interrupt mask) 1:开放请求 0:屏蔽请求 注: 在其他中断生成时, CRC 状态令牌, Ncr 超时, CMD CRC 错误和 DAT CRC 错误中断不影响其他中断的产生

### 25.5.11 CLR\_MMC\_INT

偏移地址: 0x28

复位值: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.							D1 MC	CR CE MC	CR NT MC	MB TM C	MB DM C	CM DE MC	DAT EM C	DAT DM C	CM DD MC
Type								rc_ w1	rc_ w1	rc_ w1	rc_ w1	rc_ w1	rc_ w1	rc_ w1	rc_ w1	

Bit	Field	Type	Reset	Description
31:9	Reserved			始终读为 0。
8	D1MC	rc_w1	0x0	SDIO data1 线中断标志/清除位 W: 清除 SDIO data1 线路中断 R: SDIO data1 线路中断标志
7	CRCEMC	rc_w1	0x0	CRC 状态错误标志中断屏蔽位 W: 清除 CRC 状态错误标志中断标志

Bit	Field	Type	Reset	Description
				R: CRC 状态错误标志中断标志 当该位为 1 时，判断 mmc_sig[6: 4]。
6	CRNTMC	rc_w1	0x0	命令和回应 Ncr 超时中断屏蔽位 W: 清除命令和回应 Ncr 超时中断标志。 R: Cmd 和 Resp Ncr 超时中断标志
5	MBTMC	rc_w1	0x0	多块传输超时中断屏蔽位 W: 清除多块传输超时中断标志。 R: 多块超时中断标志 (读取和写入)
4	MBDMC	rc_w1	0x0	多块传输完成中断屏蔽位 W: 清除多段完成中断标志。 R: 多块传输完成中断标志。
3	CMDEMC	rc_w1	0x0	CMD CRC 错误中断屏蔽位 W: 清除 CMD CRC 错误中断标志。 R: CMD CRC 错误原始中断
2	DATEMC	rc_w1	0x0	DAT CRC 错误中断屏蔽位 W: 清除 DAT CRC 错误中断标志。 R: DAT CRC 错误中断标志。
1	DATDMC	rc_w1	0x0	DAT 完成中断屏蔽位 W: 清除 DAT 完成中断标志。 R: DAT 完成中断标志。
0	CMDDMC	rc_w1	0x0	CMD 完成中断屏蔽位 W: 清除 CMD 完成中断标志。 R: CMD 完成中断标志。 清除响应后，这些位将自动复位为低电平打断。

### 25.5.12 MMC\_CARDSEL

偏移地址: 0x2C

复位值: 0x00000040

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.										CTR EN	ENPC LK	TSCALE			

Type		rw								
------	--	----	----	----	----	----	----	----	----	----

Bit	Field	Type	Reset	Description
31:8	Reserved			始终读为 0。
7	CTREN	rw	0x0	SD/MMC/SDIO 控制器使能位。
6	ENPCLK	rw	0x1	使能卡的 SD/MMC/SDIO 端口 CLK 时钟。
5:0	TSCALE	rw	0x0	SD/MMC/SDIO 时钟分频系数 (基于 1MHz)。 使用这些位来构建 1Mhz 时钟。 1MHz=Fhclk/ ((mmc_cardsel[5:0]+1) ×2)。

### 25.5.13 MMC\_SIQ

偏移地址: 0x30

复位值: 0x000000FF

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.															
Type									PCMDS	CRC_status			PDAT3S	PDAT2S	PDAT1S	PDAT0S

Bit	Field	Type	Reset	Description
31:8	Reserved			始终读为 0。
7	PCMDS	r	0x01	SD/MMC/SDIO 端口 CMD 线路信号。
6:4	CRC_status	r	0x07	写入数据 CRC 状态令牌时的 CRC 状态[2: 0]
3	PDAT3S	r	0x01	SD/MMC/SDIO 端口 DAT3 线路信号。
2	PDAT2S	r	0x01	SD/MMC/SDIO 端口 DAT2 线路信号。
1	PDAT1S	r	0x01	SD/MMC/SDIO 端口 DAT1 线路信号。
0	PDAT0S	r	0x01	SD/MMC/SDIO 端口 DAT0 线路信号。 当主机读取寄存器时, SD/MMC/SDIO 控制器将在 SD/MMC/SDIO 端口 CLK 线上产生一个时钟脉冲。当 SD/MMC/SDIO 端口 CLK 线上升时, SD/MMC/SDIO 端口的信号状态将被锁存到寄存器中。

### 25.5.14 MMC\_IO\_MBCTL

偏移地址: 0x34

复位值: 0x00000010

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								NTSSel		BTSSel		PCL KP	PAUT OTR	SMB DTD	SPMB DTR
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			始终读为 0。
7:6	NTSSel	rw	0x00	SD/MMC/SDIO NAC 超时级别选择位。 00: 1uS 01: 100uS 10: 10mS 11: 1S 注: 是用于 SD/MMC/SDIO 端口 NAC 超时刻度选择
5:4	BTSSel	rw	0x01	SD/MMC/SDIO BUSY 超时级别选择位。 00: 1uS 01: 100uS 10: 10mS 11: 1S 注: 是用于 SD/MMC/SDIO 端口繁忙超时刻度选择
3	PCLKP	rw	0x00	SD/MMC/SDIO 端口 CLK 线极性选择位 1: 时钟下降沿 push, 时钟上升沿 pull 0: 时钟上升沿 push, 时钟下降沿 pull
2	PAUTOTR	rw	0x00	设置 SD/MMC/SDIO 端口全自动命令和多块数据传输。 1: 使能。 0: 禁用。 设置此位为 1 (MMC_IO[7: 6]==11) 将触发 SD/MMC/SDIO 命令, 响应 8 个空时钟, 多块数据传输。数据传输完成后, 该位将自动清零。
1	SMBDTD	rw	0x00	多块数据传送方向选择位。 1: 读取数据。 0: 写入数据。
0	SPMBDTR	rw	0x00	设置 SD/MMC/SDIO 端口自动多块数据传输位。

Bit	Field	Type	Reset	Description
				<p>1: 启用 0: 禁用</p> <p>将该位置 1 将触发 SD/MMC/SDIO 多块数据传输。块计数由 mmc_blockcnt 寄存器定义。数据传输完成后，该位将自动清零。</p> <p>MMC_IO_MBCTL[2:0]和MMC_IO[7:6]为 SD/MMC/SDIO 多块数据操作控制，请参考 MMC_IO_MBCTL[2:0]和MMC_IO[7:6]的详细描述表。</p>

### 25.5.15 MMC\_BLOCKCNT

偏移地址: 0x38

复位值: 0x00000001

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	BCNT															
Type	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved			始终读为 0。
15:0	BCNT	rw	0x01	<p>数据块计数寄存器</p> <p>在多块传输模式下，配置这些位定义将要传输的数据块数量</p>

### 25.5.16 MMC\_TIMEOUTCNT

偏移地址: 0x3C

复位值: 0x00000040

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								DTCNT							
Type										rw						

Bit	Field	Type	Reset	Description
31:8	Reserved			始终读为 0。
7:0	DTCNT	rw	0x40	<p>数据传输超时计数寄存器</p> <p>Time=Scale×bit[7:0]</p>

Bit	Field	Type	Reset	Description
				注: Scale 根据 MMC_IO_MBCTL[7:6]/[5:4]进行定义

### 25.5.17 CMD\_BUFX (X=0...15)

偏移地址: 0x40-0x7C

复位值: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.								DAT							
Type									rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:8	Reserved			始终读为 0。
7:0	DAT	rw	0x00	<p>cmd_buf 字节 x, 映射到命令[ (15+8x): 8 (x+1) ]位</p> <p>cmd_buf 字节 0, 映射到命令[15: 8]位</p> <p>cmd_buf 字节 1, 映射到命令[23: 16]位</p> <p>cmd_buf 字节 2, 映射到命令[31: 24]位</p> <p>cmd_buf 字节 3, 映射到命令[39: 32]位</p> <p>cmd_buf 字节 4, 映射到命令[47: 40]位</p> <p>cmd_buf 字节 5, 映射到命令[55: 48]位</p> <p>cmd_buf 字节 6, 映射到命令[63: 56]位</p> <p>cmd_buf 字节 7, 映射到命令[71: 64]位</p> <p>cmd_buf 字节 8, 映射到命令[79: 72]位</p> <p>cmd_buf 字节 9, 映射到命令[87: 80]位</p> <p>cmd_buf 字节 10, 映射到命令[95: 88]位</p> <p>cmd_buf 字节 11, 映射到命令[103: 96]位</p> <p>cmd_buf 字节 12, 映射到命令[111: 104]位</p> <p>cmd_buf 字节 13, 映射到命令[119: 112]位</p> <p>cmd_buf 字节 14, 映射到命令[127: 120]位</p> <p>cmd_buf 字节 15, 映射到命令[135: 128]位</p>

### 25.5.18 BUF\_CTL

偏移地址: 0x80

复位值: 0x00000002

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	DBF EN	DR M	Res.	DFIF OSM	SBAD	DMA HEN	DBML								DBE	DBF
Type	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	r	r

Bit	Field	Type	Reset	Description
31:16	Reserved			始终读为 0。
15	DBFEN	rw	0x00	<p>数据 Buf 清空使能位</p> <p>1: 触发清空数据 Buf</p> <p>0: 没有效果</p> <p>当写入 1 时, 1 个时钟后自动清零。</p>
14	DRM	rw	0x00	<p>DMA 请求屏蔽</p> <p>0: 没有屏蔽</p> <p>1: 屏蔽</p> <p>注意: 请在重新配置 DMA 之前, 屏蔽此位, 启用 DMA 后, 取消屏蔽此位, DMA 请求才启动</p>
13	Reserved			始终读为 0。
12	DFIFOSM	rw	0x00	<p>数据 FIFO 状态信号屏蔽位。</p> <p>1: 激活显示数据 FIFO 状态</p> <p>0: 默认值, 屏蔽数据 FIFO 状态。</p> <p>注: 数据 FIFO 状态信号, 高电平有效。FIFO 满读 SD 卡访问, FIFO 空写 SD 卡访问。</p>
11	SBAD	rw	0x00	<p>设置 buff 的访问方向。</p> <p>1: 写</p> <p>0: 读</p>
10	DMAHEN	rw	0x00	<p>DMA 硬件接口使能。</p> <p>1: DMA 硬件接口握手。</p> <p>0: 正常的 APB 访问 buff 数据。</p> <p>当使用 DMA 接口时, 当块 (单块传输) 或多块 (多块传输) 传输完成时, 该位将自动复位。</p>
9:2	DBML	rw	0x00	数据 buff 标记。只有当 buf_ctl[10]=1 时该位才有效。
1	DBE	r	0x01	数据 buff 为空。只读。

Bit	Field	Type	Reset	Description
0	DBF	r	0x00	数据缓存已满。只读。

### 25.5.19 DATA\_BUF

偏移地址: 0x100-0x2FF

复位值: 0x00000000

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

Bit	Field	Type	Reset	Description
31:0	DB	rw	0x0000_0000	数据 buff 注: 在这个地址范围内的所有访问将被视为地址 AMBA 读访问

表 25-2 MMC\_IO 寄存器[7: 0]的详细描述

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	操作描述	Transfer
X	X	X	X	X	0	X	0	没有操作	N/A
X	X	1	X	0	Trig	X	0	生成 8 个空时钟	N/A
0	0	0	X	0	Trig	X	0	发送命令	6
0	0	0	0	1	Trig	X	0	接收响应	6
0	0	0	1	1	Trig	X	0	重申响应	17
1	0	0	0	0	Trig	X	0	发送命令+8 个空时钟	N/A
0	1	0	0	0	Trig	X	0	发送命令+接收响应	N/A
1	1	0	0	0	Trig	X	0	发送命令, 接收响应+8 个空时钟	N/A
X	X	X	X	X	0	1	Trig	从卡中读取单个数据+8 空时钟	mmc_bytectn
X	X	X	X	X	0	0	Trig	将单个数据到卡写入卡+8 空时钟	mmc_bytectn

MMC\_IO 的 bit[7:0]定义为 SDIO/MMC 命令, 响应和单块数据操作控制。

注:

- 在上表中, 除了最后两行, 其他都会产生 CMD 完成中断 (前提中断未被屏蔽);
- 在上表中, 只有最后两行会产生 DAT 完成中断 (前提中断未被屏蔽)。

表 25-3 MMC\_IO\_MBCTL[2:0]和 MMC\_IO[7:6]的详细描述表

操作说明						传输字节
MMC_IO		MMC_IO_MBCTL				
1	1	Trig	0	0	写入多个块命令+响应+8 个时钟+数据 response+8null clocks+data	mmc_blockcnt
1	1	Trig	1	0	读取多个块命令+响应+8 个时钟+数据 response+8null clocks+data	mmc_blockcnt
X	X	0	0	Trig	只写入多个块数据	mmc_blockcnt
X	X	0	1	Trig	只读取多个块数据	mmc_blockcnt

MMC\_IO\_MBCTL 的 Bit[2:0]和 MMC\_IO 的 Bit[7:6]是为 SD/MMC/SDIO 多块数据操作控制定义的。

注:

- 在上表中, 如果中断未被屏蔽, 前两个都将产生多块传输完成中断, 每个 block 都会产生 DAT 完成中断和 CMD 完成中断;
- 在上表中, 如果中断未被屏蔽, 当发生超时事件时, 产生多块传输完成中断, 但会产生多块传输超时中断。

其他注意事项:

- 1) 使用 DMA 硬件接口时, 应先启用 DMA;
- 2) 如果没有使用中断(中断被屏蔽), 当命令/响应/8 个空时钟被传输时, 可以检查 MMC\_IO[2]; 当数据被传输时, 可以检查 MMC\_IO[0]; 当应该传输多个块数据时, 可以检查 MMC\_IO\_MBCTL[2] / MMC\_IO\_MBCTL[0];
- 3) 如果产生 Ncr 超时或多个块数据超时, 传输将被暂停。软件应该重新开始新的控制。

## 26 可变静态存储控制器 (FSMC)

### 26.1 设计概述

FSMC 是一种灵活的存储器控制接口，通过配置不同模式和相关寄存器，以达到满足不同外接设备的协议要求。

#### 26.1.1 框图

下图展示了 FSMC 的框图：

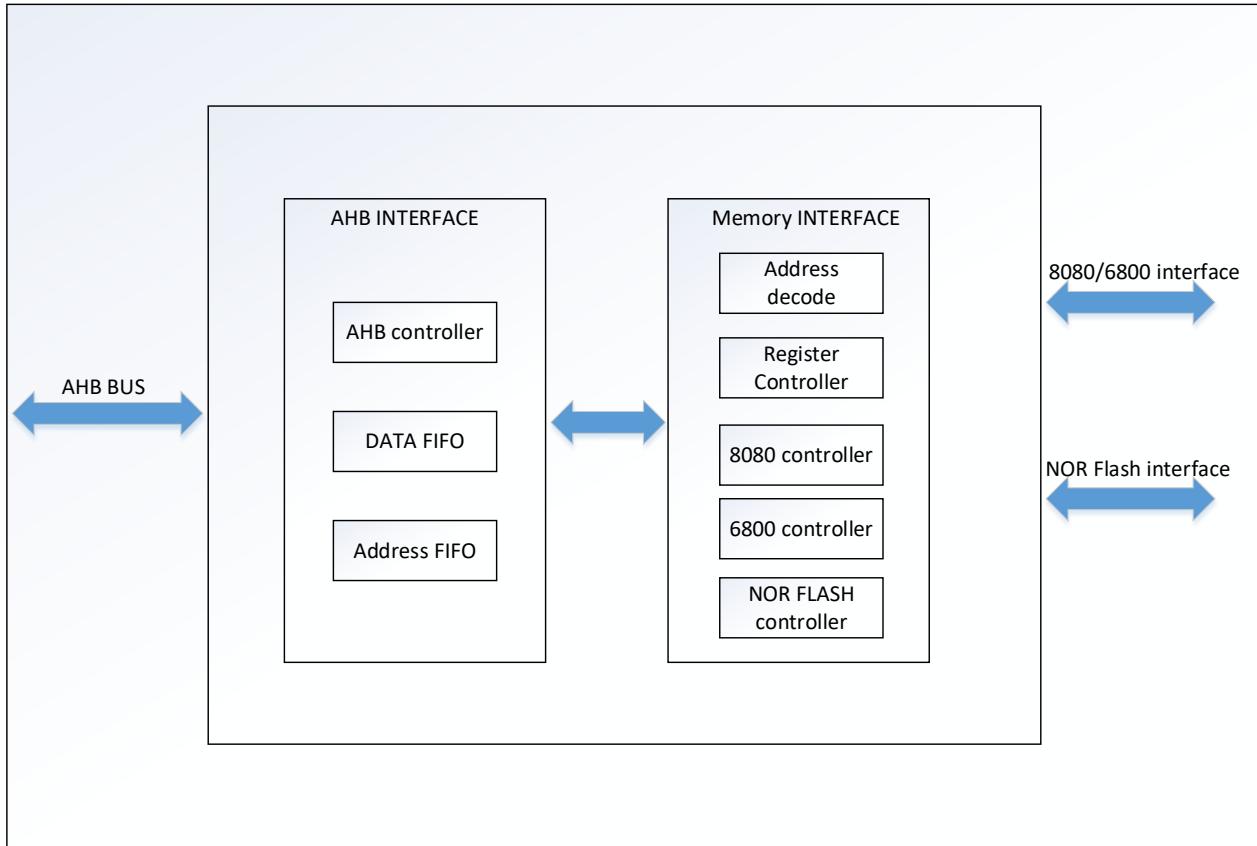


图 26-1 FSMC 框图

#### 26.1.2 接口描述

表 26-1 接口描述

Signal Name	Direction	Description
FSMC_A[25:0]	OUTPUT	地址信号
FSMC_DATA_OUT[15:0]	OUTPUT	16 位输出数据，可选择输出数据或者低 16 位地址
FSMC_NCS[4:1]	OUTPUT	BANK1 子 BANK4 位片选信号
FSMC_NBS[1:0]	OUTPUT	BYTE 选择信号
FSMC_NOE	OUTPUT	读使能信号
FSMC_NWE	OUTPUT	写使能信号
FSMC_OUT_EN	OUTPUT	输出使能
FSMC_NADV	OUTPUT	地址有效信号

Signal Name	Direction	Description
FSMC_NWAIT	INPUT	NOR FLASH 输入等待信号
FSMC_DATA_IN	INPUT	16 位输入数据

### 26.1.3 特性

- 1) 可配置的静态存储器接口包括:
  - a) SRAM
  - b) PSRAM
  - c) NOR FLASH
- 2) 支持 Intel 8080 协议
- 3) 支持 moto 6800 协议
- 4) 8 位,16 位,32 位可配置的数据总线
- 5) BANK1 分为 4 块子 BANK, 每块 64M 空间
- 6) 时序可编程以满足不同的需求
  - a) 等待周期可编程
  - b) 总线恢复周期可编程
  - c) 写, 读控制周期可编程
- 7) 可将 32 位的 AHB 访问请求, 转换为对外接设备连续的 8 位,16 位的访问

内部控制器	BANK号	地址范围	设备类型
NOR FLASH控制器	BANK1	0xX000_0000~0xFFFF_FFFF	SRAM/ROM NOR Flash PSRAM

图 26-2 NOR FLASH 控制器

所选BANK	片选信号	地址范围
BANK1.sub_bank1	FSMC_NCS1	0xX000_0000~0xX3FF_FFFF
BANK1.sub_bank2	FSMC_NCS2	0xX400_0000~0xX7FF_FFFF
BANK1.sub_bank3	FSMC_NCS3	0xX800_0000~0xXBFF_FFFF
BANK1.sub_bank4	FSMC_NCS4	0xXC00_0000~0xXFFF_FFFF

图 26-3 NOR FLASH 控制器

## 26.2 功能描述

本章节主要描述 FSMC 的功能操作。

### 26.2.1 FSMC 数据位宽

AHB 接口为内部 CPU 和 DMA 总线控制设备访问外部存储器等提供了通道，可支持 8、16、32 位接口，当选择的外部存储器的数据通道是 8 位或 16 位时，在 AHB 上的 32 位数据会被分割成连续的 8 位或 16 位数据操作。

寄存器的 SMCTRLR 的 sm\_data\_width[2:0]，定义了外部存储器的数据宽度，需根据实际数据宽度配置为 8 位,16 位, 32 位，此时需要保障实现数据传输的一致性。

表 26-2 存储器位宽

外部存储器宽度		
CPU/DMA 支持 AHB 访问宽度		
8 位	8 位	16 位
8 位	模式一	模式三
16 位	模式二	模式一
32 位	模式二	模式二

---

FSMC 执行操作规则如下：

模式一、AHB 操作的数据宽度与存储器数据宽度相同，无数据传输一致性的问题。

模式二、AHB 操作的数据宽度大于存储器的数据宽度。

存储器的数据宽度，将根据寄存器 SMCTRLR 的 sm\_data\_width\_set0/1/2 来设置，

例如当 sm\_data\_width\_set0 = 3'h0，表示外部存储器是 16bits 数据宽度；

此时，AHB 接口将对 hwdata[15:0],hwdatabit[31:16]进行连续写操作，以适应外部设备的数据宽度；

读操作时，hrdata[31:0]的低 16 位是有效数据。

模式三、AHB 操作的数据宽度小于存储器的数据宽度

若存储设备没有高低字节片选，不允许进行写操作，若存储设备有高低字节选择，通过 BL 控制访问对应字节。

可以进行读操作，但有效数据需要用户自己处理。

### 26.2.2 FSMC 模式配置

FSMC 通过寄存器 SYSCFG\_CFGR1[30:29]: mode\_sel 来配置不同模式，默认值为 01

00: 兼容 NOR FLASH 接口

01: 兼容 8080 协议接口

10: 兼容 6800 协议接口

11: 保留

NOR Flash 支持的传输模式（16bit NOR Flash 示例）：

模式	读/写	AHB传输宽度	存储器宽度	备注
异步访问	读	8	16	支持
	写	8	16	不支持
	读	16	16	支持
	写	16	16	支持
	读	32	16	支持
	写	32	16	支持

图 26-4 NOR FLASH 16bit 示例

PSRAM 支持的传输模式 (16bit PSRAM 示例):

模式	读/写	AHB传输宽度	存储器宽度	备注
异步访问	读	8	16	支持
	写	8	16	支持, 使用NBL选择高低字节
	读	16	16	支持
	写	16	16	支持
	读	32	16	支持
	写	32	16	支持, 分解成连续两次读写访问

SRAM 支持的传输模式 (16bit SRAM 示例):

模式	读/写	AHB传输宽度	存储器宽度	备注
异步访问	读	8	16	支持, 使用NBL选择高低字节
	写	8	16	
	读	16	16	支持
	写	16	16	支持
	读	32	16	支持, 分解成连续两次读写访问
	写	32	16	

图 26-5 SRAM 16bit 示例

模式	读/写	AHB传输宽度	存储器宽度	备注
异步访问	读	8	8	支持
	写	8	8	支持
	读	16	8	支持, 分解成连续两次读写访问
	写	16	8	
	读	32	8	支持, 分解成连续4次读写访问
	写	32	8	

图 26-6 SRAM 8bit 示例

寄存器 SMSKR0[10:8]用来选择三组不同的寄存器 register set0/set1/set2, 以配置不同的时序;

### 26.2.3 NOR Flash/SRAM/PSRAM 模式

NOR Flash/SRAM/PSRAM 控制器外部信号复用模式的接口如下：

信号名称	信号方向	NOR		PSRAM/SRAM		功能
		复用	非复用	复用	非复用	
A[25:0]	输出	x	x	x	x	地址总线
D[15:0]	输入/输出	x	x	x	x	双向数据总线
NE[3:0]	输出	x	x	x	x	四个子BANK片选
NOE	输出	x	x	x	x	读使能
NWE	输出	x	x	x	x	写使能
NADV	输出	x		x		数据/地址复用时的地址锁存
NBL[1:0]	输出			x	x	高低字节选择
NWAIT	输入	x	x	x	x	外部存储器等待输入信号

图 26-7 复用 NOR Flash/SRAM/PSRAM 接口

注意：复用模式下，地址低 16 位和 16 位的数据线复用，通过 NADV 信号区别地址还是数据；

地址高 10 位 A[25:16]独立

下面是接口操作时序的配置：

表 26-3 NOR FLASH 写时序说明

写操作阶段	周期(SMTMGR_SETn)	说明
addr setup	t_as	地址建立时间, NADV 有效周期
addr hold	固定为 1 个周期	地址保持时间
write pulse	t_wp + 1	写使能周期, NWE 有效周期
write recovery	t_wr	写使能释放周期

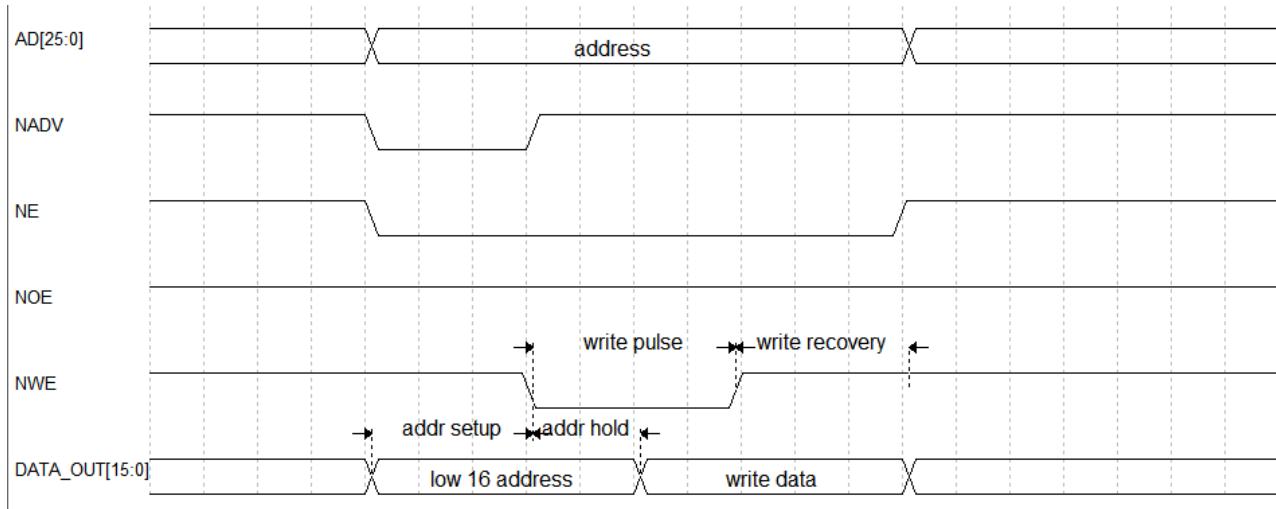


图 26-8 NOR FLASH Write

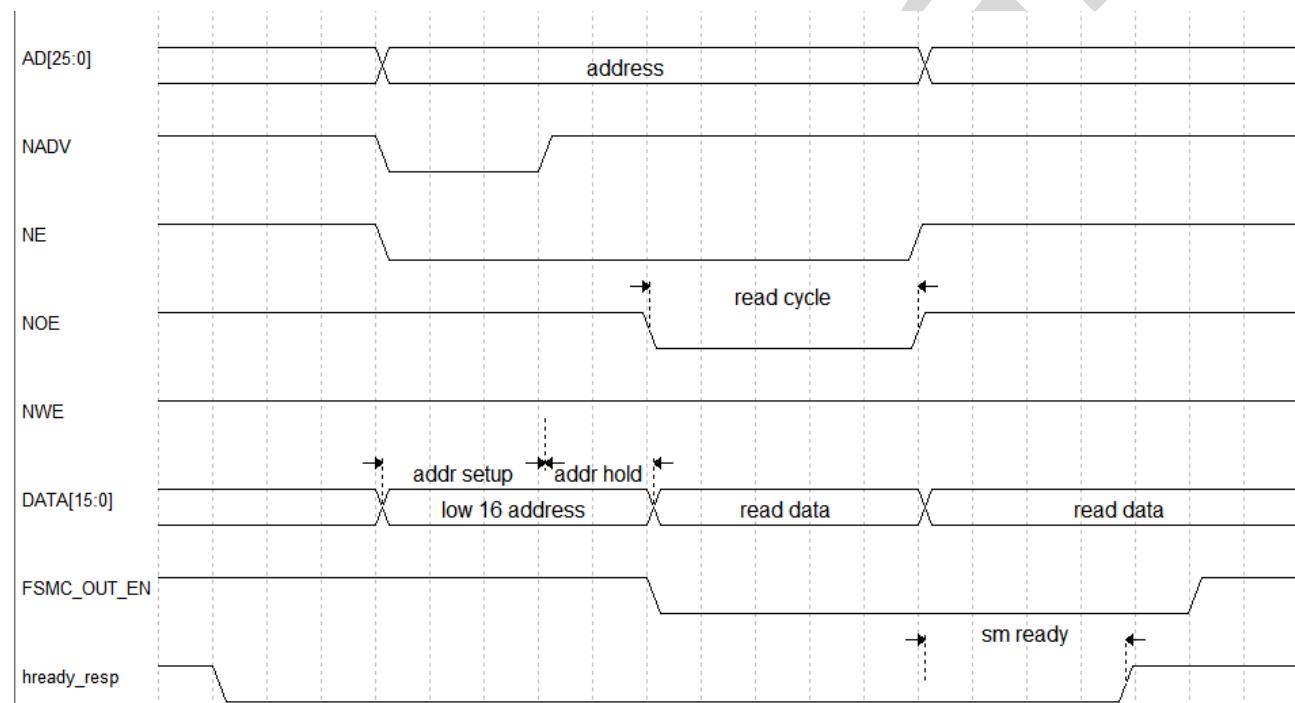


图 26-9 NOR FLASH Read

表 26-4 NOR FLASH 读时序说明

读操作阶段	周期(SMTMGR_SETn 值)	说明
addr setup	固定为 1 个周期	地址建立时间, NADV 有效周期
addr hold	固定为 1 个周期	地址保持时间
read cycle	t_rc - 1	读使能周期, NOE 有效周期
fsmc_out_en (低电平周期)	read pulse + 5*hclk	FSMC_AD[15:0]输入/输出控制信号 0: 输入模式 1: 输出模式
sm_ready	sm_read_pipe +2*hclk	NOE 无效到 hready_resp 拉高的周期 在 hready_resp 上升沿处采样数据

## 26.2.4 8080 协议模式

8080 协议接口只用到了部分 PIN, 接口如下:

表 26-5 8080 协议接口

信号名称	信号方向	功能
A[25:16]	输出	高 10 位地址总线, 只作 RS 信号使用
NE[3:0]	输出	片选信号, 低有效
NOE	输出	读使能信号, 低有效
NWE	输出	写使能信号, 低有效
AD[15:0]	输入/输出	低 16 位地址/数据

在 8080 模式下, FSMC\_A[25:16]中的一个 bit 作为 RS 控制信号。下面对地址信号之间的关系进行简单的描述:

$$\text{FSMC\_A}[25:16] = \text{sm\_addr}[25:16]$$

表 26-6 8080 地址说明

sm_data_width	data bit	address
100	8	sm_addr[25:0] = haddr[25:0]
001	16	sm_addr[25:0] = {1'h0,haddr[25:1]}
010	32	sm_addr[25:0] = {2'h0,haddr[25:2]}

例如:

若要写指令, RS 信号需要拉高(根据实际外接 DEVICE 的时序标准),

sm\_data\_with=100 时, ( $\text{FSMC\_BANK1\_BASE} + (0x1 \ll 23)$ ) == CMD

则  $\text{FSMC\_A}[23]$  作为 RS 控制信号,

sm\_data\_with=001 时, ( $\text{FSMC\_BANK1\_BASE} + (0x1 \ll 23)$ ) == CMD,

则  $\text{FSMC\_A}[21]$  作为 RS 控制信号,

若要写数据, RS 信号需要拉低(根据实际外接 DEVICE 的时序标准),

sm\_data\_with=100 时, ( $\text{FSMC\_BANK1\_BASE}$ ) = DATA,

则  $\text{FSMC\_A}[23]$  作为 RS 控制信号,

sm\_data\_with=001 时, (FSMC\_BANK1\_BASE)= DATA,  
则 FSMC\_A[21]作为 RS 控制信号。

1) 下面是 8080 REG 写操作时序:

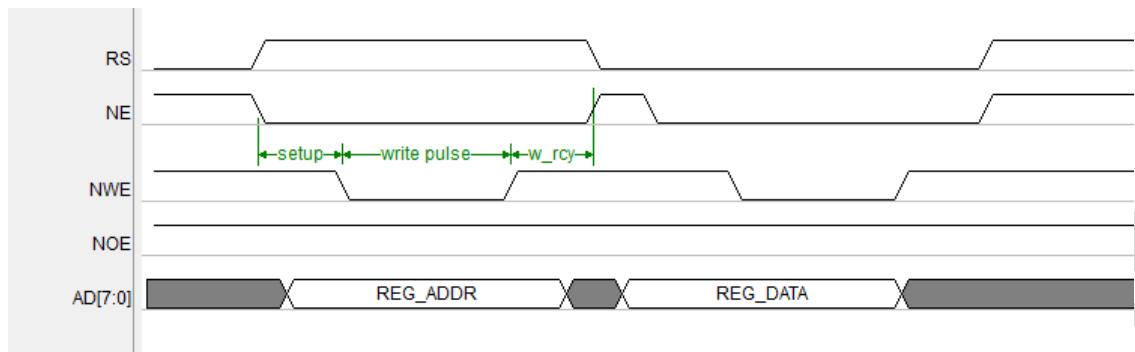


图 26-10 8080 Reg Write

表 26-7 8080 读时序说明

REG WRITE	周期(SMTMGR_SETn)	说明
setup	t_as	建立时间
write pulse	t_wp + 1	写使能周期, NWE 有效周期
w_rcy	t_wr	写使能释放周期

下面是 8080 REG 读操作时序:

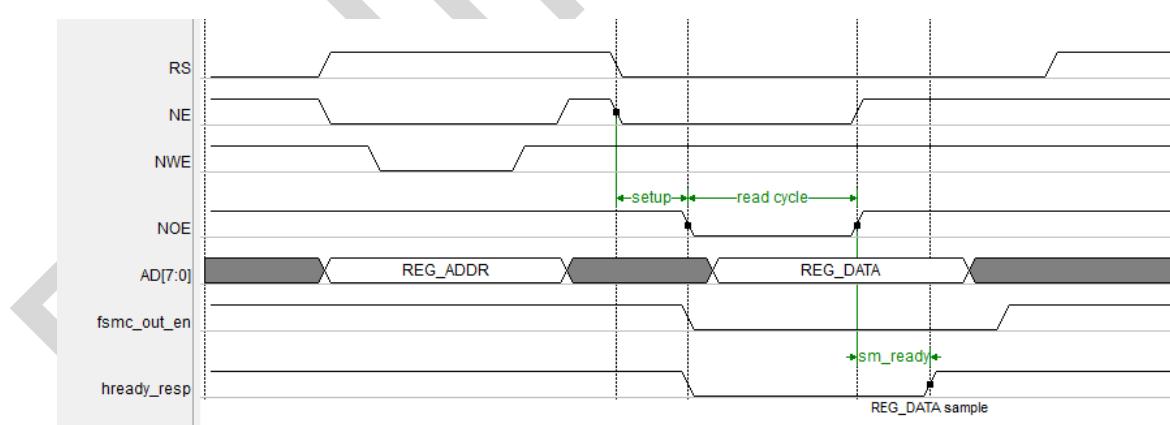


图 26-11 8080 Reg Read

表 26-8 8080 读时序说明

REG READ	周期(SMTMGR_SETn 值)	说明
setup	固定为 1 个周期	建立时间
read cycle	t_rc	读使能周期, NOE 有效周期
fsmc_out_en (低电平周期)	read pulse + 5*hclk	FSMC_AD[15:0]输入/输出控制信号 0: 输入模式 1: 输出模式
sm_ready	sm_read_pipe + 2*hclk	NOE 无效到 hready_resp 拉高的周期 在 hready_resp 上升沿处采样数据

下面是 8080 MEM 操作时序:

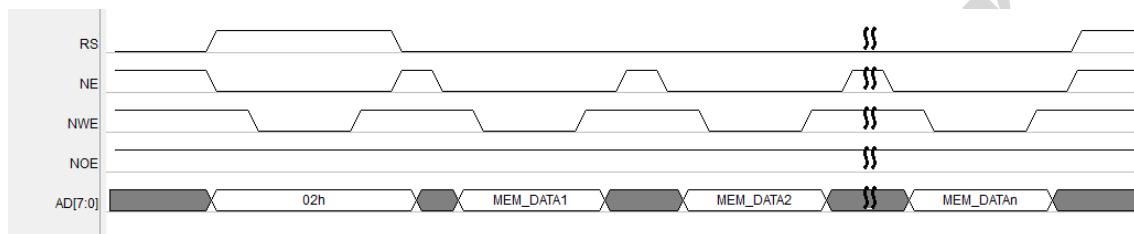


图 26-12 8080 Memory Write

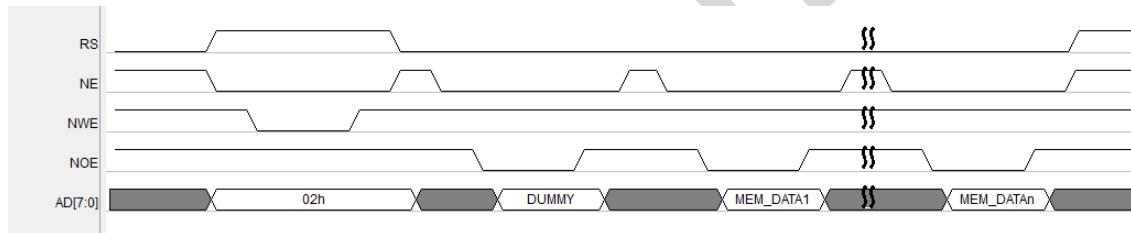


图 26-13 8080 Memory Read

MEM 操作时序设置与 REG 操作时序一致。

## 26.2.5 6800 协议描述

6800 协议接口只用到了部分 PIN, 接口如下:

表 26-9 6800 协议接口

信号名称	信号方向	功能
A[25:0]	输出	地址总线, 高 10 位只作 RS 信号使用
NE[3:0]	输出	片选信号, 低有效
OUT_EN	输出	使能信号, 高为写, 低为读
DATA[15:0]	输入/输出	16 位数据

在 6800 模式下, FSMC\_A[25:16]中的一个 bit 作为 RS 控制信号。

下面对地址信号之间的关系进行简单的描述:  $FSMC\_A[25:16] = sm\_addr[25:16]$

表 26-10 6800 地址说明

sm_data_width	data bit	address
---------------	----------	---------

100	8	sm_addr[25:0] = haddr[25:0]
001	16	sm_addr[25:0] = {1'h0,haddr[25:1]}
010	32	sm_addr[25:0] = {2'h0,haddr[25:2]}

例如：

若要写指令，RS 信号需要拉高(根据实际外接 DEVICE 的时序标准)，

sm\_data\_with=100 时, (FSMC\_BANK1\_BASE+ (0x1«23))==CMD 则 FSMC\_A[23]作为 RS 控制信号,

sm\_data\_with=001 时, (FSMC\_BANK1\_BASE+ (0x1«23))==CMD, 则 FSMC\_A[21]作为 RS 控制信号,

若要写数据，RS 信号需要拉低(根据实际外接 DEVICE 的时序标准)，

sm\_data\_with=100 时, (FSMC\_BANK1\_BASE)= DATA, 则 FSMC\_A[23]作为 RS 控制信号,

sm\_data\_with=001 时, (FSMC\_BANK1\_BASE)= DATA, 则 FSMC\_A[21]作为 RS 控制信号。

2) 下面是 6800 REG 写操作时序：

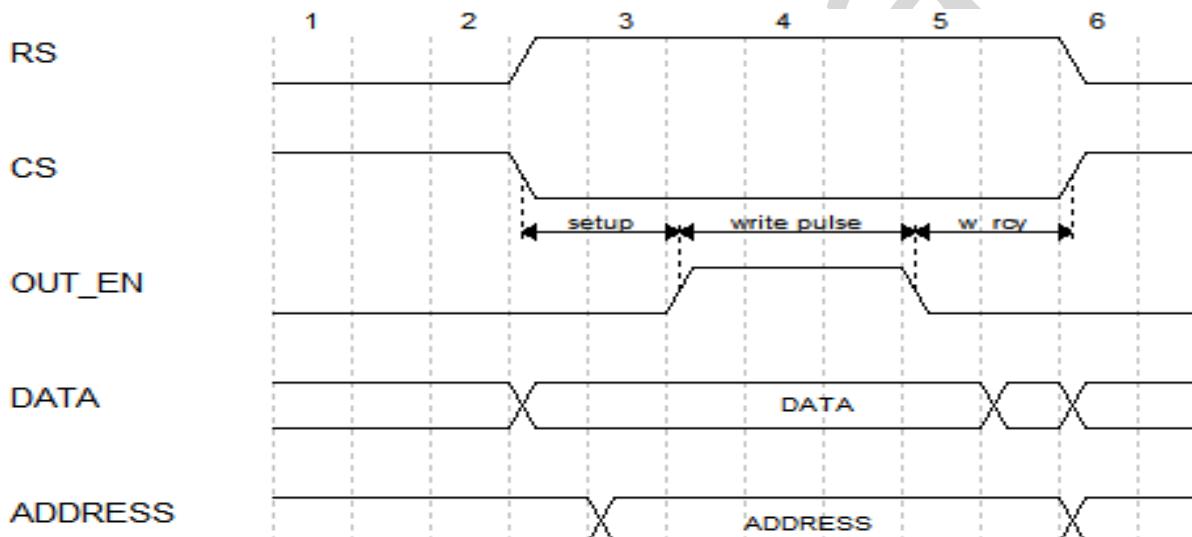


图 26-14 6800 Reg Write

表 26-11 6800 读时序说明

REG WRITE	周期(SMTMGR_SETn)	说明
setup	t_as	建立时间
write pulse	t_wp + 1	写使能周期, NWE 有效周期
w_rcy	t_wr	写使能释放周期

下面是 6800 REG 读操作时序：

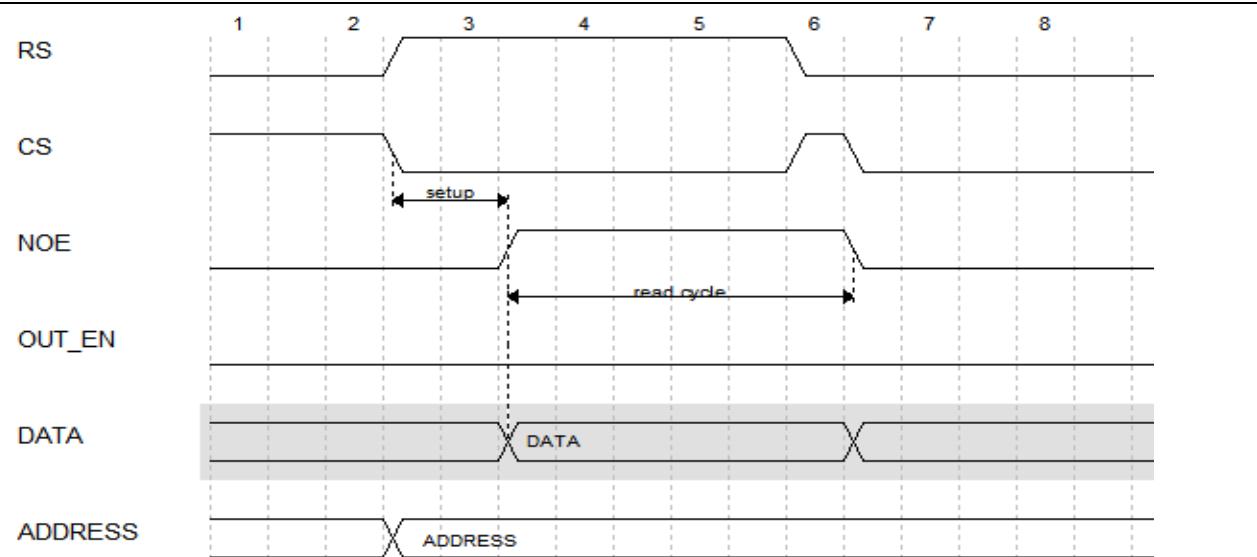


图 26-15 6800 Reg Read

表 26-12 6800 读时序说明

REG READ	周期(SMTMGR_SETn 值)	说明
setup	固定为 1 个周期	建立时间
read cycle	t_rc	读使能周期, NOE 有效周期

下面是 6800 MEM 操作时序:

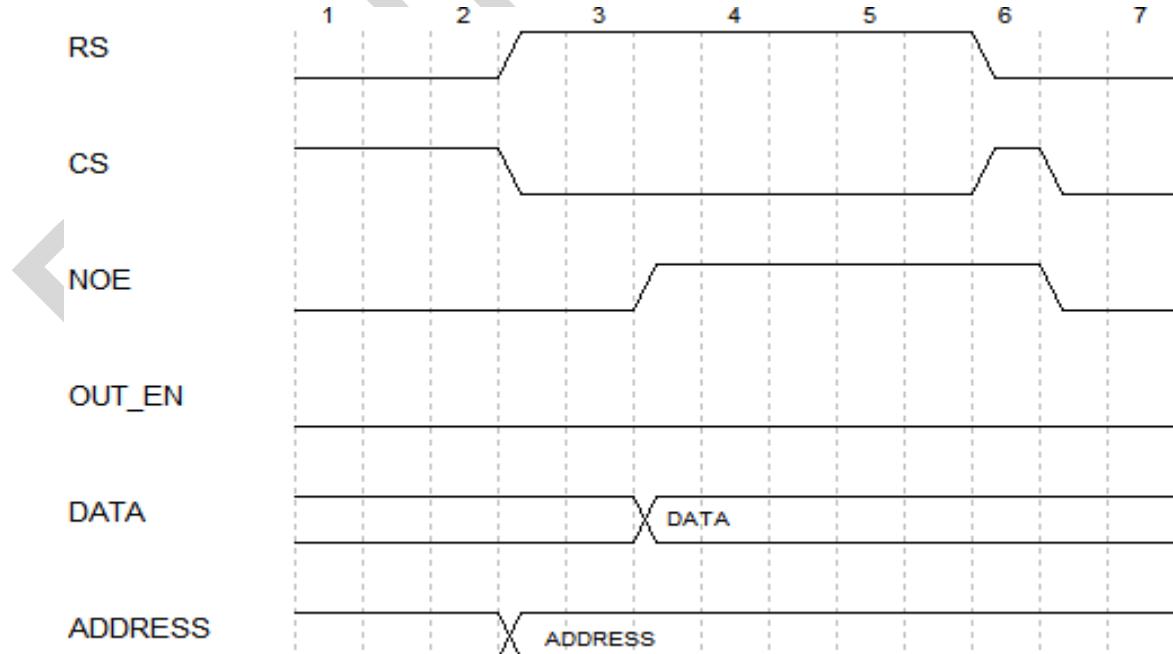
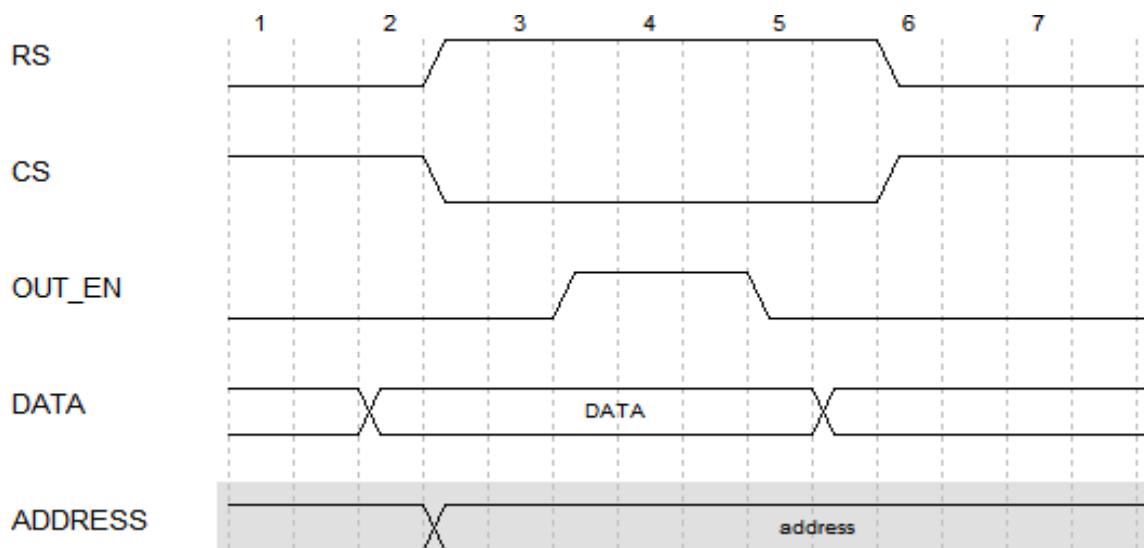


图 26-17 6800 Memory Read

MEM 操作时序设置与 REG 操作时序一致。

## 26.3 寄存器描述

表 26-13 寄存器映射表

Offset	Acronym	Register Name	Reset
0x54	SMSKR0	存储器屏蔽寄存器	0x0000024d
0x94	SMTMGR_SET0	存储器时序寄存器 0	0x04010441
0x98	SMTMGR_SET1	存储器时序寄存器 1	0x007c4f5b
0x9C	SMTMGR_SET2	存储器时序寄存器 2	0x001c194e
0xA4	SMCTRL	存储器控制寄存器	0x00000081

### 26.3.1 存储器屏蔽寄存器 (SMSKR0)

地址偏移: 0x54

复位值: 0x24d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved					Reg select			Mem type			Mem size				
Type						rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:11	Reserved			保留, 始终读为 0
10:8	Reg_select	rw	0x2	reg_select[2:0]: 时序参数配置寄存器组选择 0 - register set0 - register set1 - register set2
7:5	Mem_type	rw	0x2	Mem_type[2:0]: 外接设备类型 1-SRAM, 2-FLASH, 3-PSRAM, 其他值保留
4:0	Mem_siez	rw	0xd	mem_size[4:0]: 外接 DEVICE 的容量大小 0 - 未连接 DEVICE 1 - 64KB 2 - 128KB 3 - 256KB

Bit	Field	Type	Reset	Description
				4 - 512KB - 1MB - 2MB - 4MB - 8MB 9 - 16MB 32MB 64MB 12- 128MB 13- 256MB 14- 512MB 15- 1GB 2GB 4GB

Bit	Field	Type	Reset	Description
31:11	Reserved			保留, 始终读为 0
10:8	Reg_select	rw	0x2	<b>reg_select[2:0]: 时序参数配置寄存器组选择</b> 0 - register set0 - register set1 - register set2
7:5	Mem_type	rw	0x2	<b>Mem_type[2:0]:外接设备类型</b> 1-SRAM, 2-FLASH, 3-PSRAM, 其他值保留
4:0	Mem_siez	rw	0xd	<b>mem_size[4:0]:外接 DEVICE 的容量大小</b> 0 - 未连接 DEVICE 1 - 64KB 2 - 128KB 3 - 256KB 4 - 512KB - 1MB - 2MB

Bit	Field	Type	Reset	Description
				- 4MB - 8MB 9 - 16MB 32MB 64MB 12- 128MB 13- 256MB 14- 512MB 15- 1GB 2GB 4GB

### 26.3.2 存储器时序寄存器(SMTMGR\_SET0)

地址偏移: 0x94

复位值: 0x4010441

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
Field			Sm_read_pipe			Read y_mo de												
Type			rw			rw												
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Field	T_wp						T_wr		T_as		T_rc							
Type	rw						rw		rw		rw							

Bit	Field	Type	Reset	Description
31:30	Reserved			保留, 始终读为 0
29:28	Sm_read_pipe	rw	0x0	sm_read_pipe[1:0] 锁存读数据的周期, 即 hready_resp 拉高的周期
27	reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
26	Ready_mode	rw	0x1	ready_mode: 选择 hready_resp 信号是来自 FSMC IP 内部, 还是外部 DEVICE, 只针对写,读外部 DEVICE 操作。 0: 内部 FSMC 1: 外部 DEVICE(即来自 FSMC_NWAIT)
25:16	reserved			保留, 始终读为 0
15:10	T_wp	rw	0x1	写操作周期, 取值范围 0~63, 对应写脉冲宽度 1~64 个时钟周期
9:8	T_wr	rw	0x0	写操作时地址/数据的保持时间,取值范围 0~3, 对应写操作时地址/数据的保持时间 0~3 个时钟周期
7:6	T_as	rw	0x1	写操作时地址的建立时间, 取值范围 0~3, 对应写操作时地址的建立时间 0~3 个时钟周期, 0 值只在 SSRAM 时有效
5:0	T_rc	rw	0x1	读操作周期, 取值范围 0~63, 对应读操作周期 1~64 个时钟周期

### 26.3.3 存储器时序寄存器(SMTMGR\_SET1)

地址偏移: 0x98

复位值: 0x7c4f5b

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
Field	Sm_read_pipe					Read_y_mode												
Type	rw					rw												
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Field	T_wp						T_wr		T_as		T_rc							
Type	rw						rw		rw		rw							

Bit	Field	Type	Reset	Description
31:30	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
29:28	Sm_read_pipe	rw	0x0	sm_read_pipe[1:0] 锁存读数据的周期, 即 hready_resp 拉高的周期
27	reserved			保留, 始终读为 0
26	Ready_mode	rw	0x1	ready_mode: 选择 hready_resp 信号是来自 FSMC IP 内部, 还是外部 DEVICE, 只针对写, 读外部 DEVICE 操作。 0: 内部 FSMC 1: 外部 DEVICE(即来自 FSMC_NWAIT)
25:16	reserved			保留, 始终读为 0
15:10	T_wp	rw	0x13	写操作周期, 取值范围 0~63, 对应写脉冲宽度 1~64 个时钟周期
9:8	T_wr	rw	0x3	写操作时地址/数据的保持时间, 取值范围 0~3, 对应写操作时地址/数据的保持时间 0~3 个时钟周期
7:6	T_as	rw	0x1	写操作时地址的建立时间, 取值范围 0~3, 对应写操作时地址的建立时间 0~3 个时钟周期, 0 值只在 SSRAM 时有效
5:0	T_rc	rw	0x1b	读操作周期, 取值范围 0~63, 对应读操作周期 1~64 个时钟周期

### 26.3.4 存储器时序寄存器(SMTMGR\_SET2)

地址偏移: 0x9c

复位值: 0x1c194e

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

Field	T_wp		T_wr	T_as	T_rc
Type	rw		rw	rw	rw

Bit	Field	Type	Reset	Description
31:30	Reserved			保留, 始终读为 0
29:28	Sm_read_pipe	rw	0x0	sm_read_pipe[1:0] 锁存读数据的周期, 即 hready_resp 拉高的周期
27	reserved			保留, 始终读为 0
26	Ready_mode	rw	0x1	ready_mode: 选择 hready_resp 信号是来自 FSMC IP 内部, 还是外部 DEVICE, 只针对写, 读外部 DEVICE 操作。 0: 内部 FSMC 1: 外部 DEVICE(即来自 FSMC_NWAIT)
25:16	reserved			保留, 始终读为 0
15:10	T_wp	rw	0x6	写操作周期, 取值范围 0~63, 对应写脉冲宽度 1~64 个时钟周期
9:8	T_wr	rw	0x1	写操作时地址/数据的保持时间, 取值范围 0~3, 对应写操作时地址/数据的保持时间 0~3 个时钟周期
7:6	T_as	rw	0x1	写操作时地址的建立时间, 取值范围 0~3, 对应写操作时地址的建立时间 0~3 个时钟周期, 0 值只在 SRAM 时有效
5:0	T_rc	rw	0xe	读操作周期, 取值范围 0~63, 对应读操作周期 1~64 个时钟周期

### 26.3.5 存储器控制寄存器(SMCTLR)

地址偏移: 0xA4

复位值: 0x81

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Sm_data_width_set2			Sm_data_width_set1			Sm_data_width_set0			保留						flash_rp
Type	rw			rw			rw									rw

Bit	Field	Type	Reset	Description
31:16	Reserved			保留, 始终读为 0
15:13	Sm_data_width_set2	rw	0x0	存储器数据总线位宽设置 000 - 16 bits 001 - 32 bits 100 - 8 bits 其余值保留
12:10	Sm_data_width_set1	rw	0x0	存储器数据总线位宽设置 000 - 16 bits 001 - 32 bits 100 - 8 bits 其余值保留
9:7	Sm_data_width_set0	rw	0x1	存储器数据总线位宽设置 000 - 16 bits 001 - 32 bits 100 - 8 bits 其余值保留
6:1	Reserved			保留, 始终读为 0
0	Flash_rp	rw	0x1	FLASH 复位低功耗模式, NORFLASH 模式时需配置为 1

# 27 模拟/数字转换(ADC)

## 27.1 简介

ADC 是 12 位的逐次逼近型 (SAR) 模拟数字转换器，可以将模拟信号转换成数字信号。

转换器支持多种工作模式:单次转换和连续转换模式，并且可以选择通道自动扫描及扫描方向。转换的启动方式有软件设定、外部引脚触发以及各个定时器启动。

## 27.2 功能框图

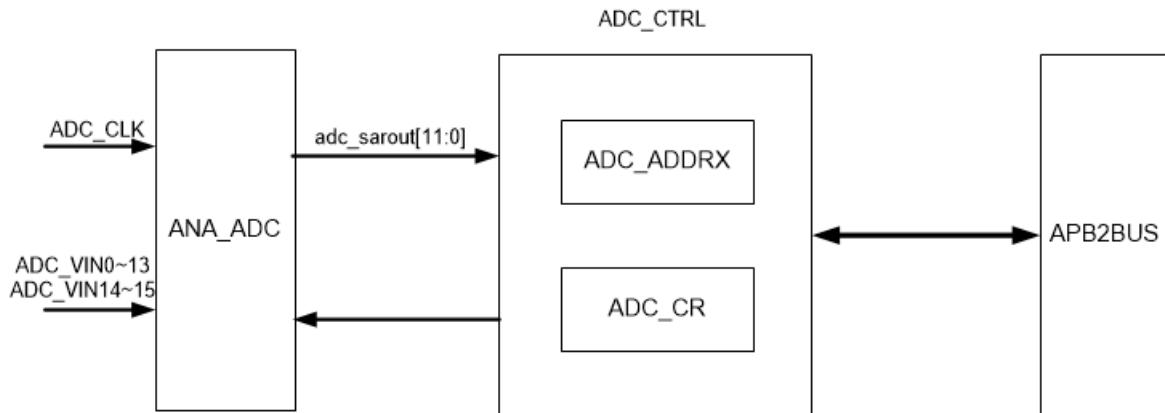
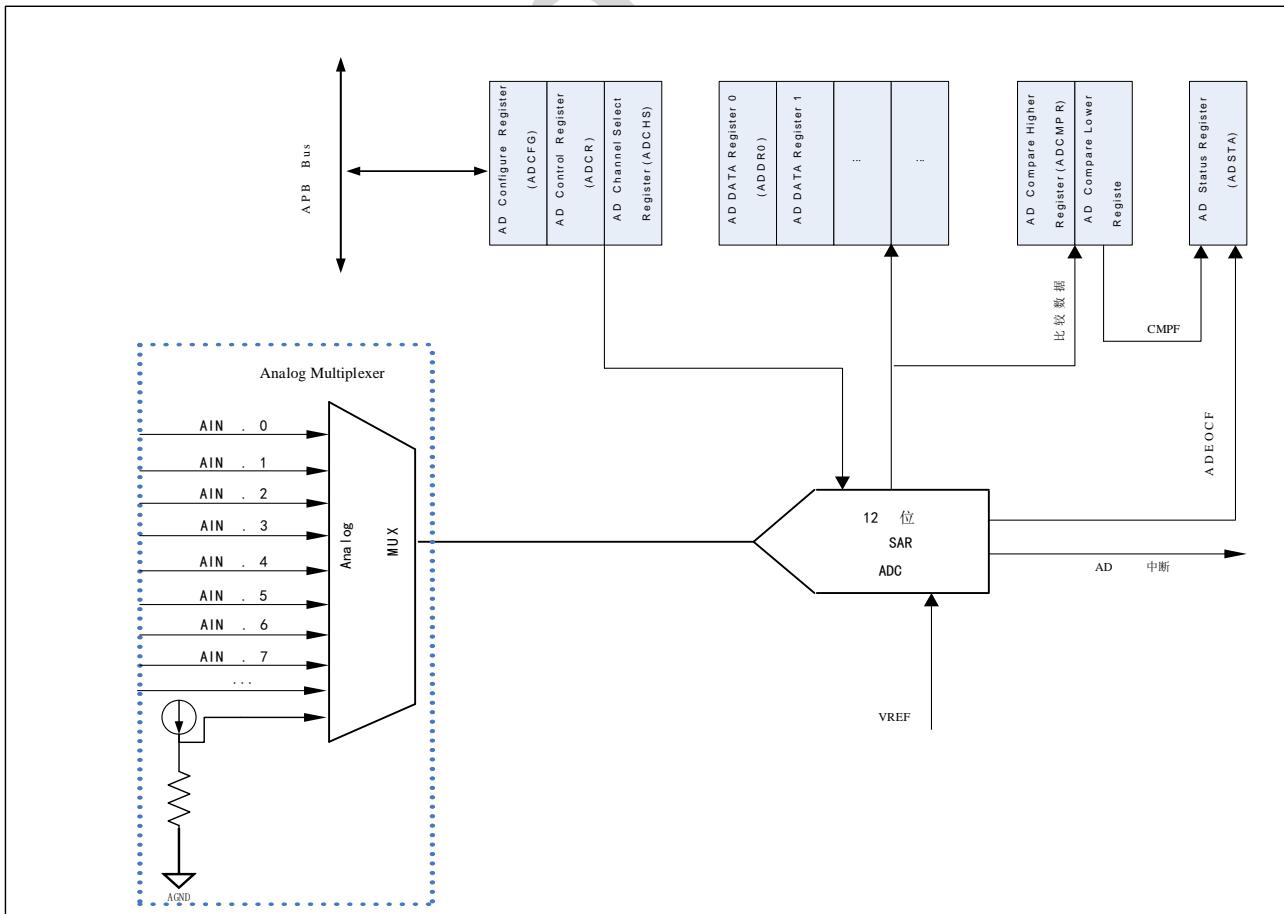


图 27-1 ADC 系统框图



---

图 27-2 ADC 框图

注:T\_SENSOR (温度传感器) 通道在 ADC1 的 AIN14 通道, V\_SENSOR (内部参考电压) 通道在 ADC1 的 AIN15 通道。

### 27.3 主要特征

- 4) 最高 12 位可编程分辨率的 SAR ADC, ADC1 有 14 路外部输入通道和 2 路内部通道, ADC2 和 ADC3 有 16 路外部输入通道;
- 5) 支持最大输入时钟为 15MHz, 由 PCLK2 分频产生 ;
- 6) 高达 1Msps 转换速率;
- 7) 支持普通工作模式:
  - c) 单次转换模式:A/D 转换在指定通道完成一次转换;
  - d) 单周期扫描模式:A/D 转换在所有指定通道 (从低序号通道到高序号通道, 或从高序号通道到低序号通道) 完成一个周期转换;
  - e) 连续扫描模式:A/D 转换连续执行单周期扫描模式直到软件停止 A/D 转换。若需中途修改转换通道只能停止 A/D 转换, 配置完相关寄存器再重新开启转换;
- 8) 支持任意通道工作模式:
  - f) 单次转换模式:A/D 转换在指定通道完成一次转换;
  - g) 单周期扫描模式:A/D 转换在所有指定通道 (可按照任意顺序) 完成一个周期转换;
  - h) 连续扫描模式:A/D 转换连续执行单周期扫描模式直到软件停止 A/D 转换。若想修改转换通道, 不必停止转换, 可配置相应寄存器, 在下一个扫描周期开始将进行新的通道转换;
- 9) 支持注入通道工作模式:
  - a) 自动注入: 任意通道工作完成后自动开始注入通道工作;
  - b) 事件注入: 注入事件到来, A/D 转换完成当前转换后开始注入通道转换, 完成注入通道转换后继续执行任意通道转换;
- 10) 通道采样时间可为每个通道分别配置、分辨率可通过软件按通道分别配置
- 11) 支持 DMA 传输;
- 12) A/D 转换开始条件:
  - i) 软件启动;
  - j) 外部触发启动, 且软件可配置外部触发延时;
  - k) Timer1/2/3/8 匹配或 TRGO 信号, 外部 EXTI 信号源;
- 13) 模拟看门狗功能, 转换结果可和指定的阈值区间相比较, 当转换值超出设定的阈值区间时, 可以根据用户设定判断是否产生中断请求

### 27.4 中断

ADC 包括窗口比较器中断 ADWIE 和转换完成中断 ADIE, 单通道转换完成中断 EOCIE, 采样结束中断 EOSMPIE, 注入通道组转换完成中断 JEOSIE, 注入单通道转换完成中断 JELOCIE, 注入采样结束中断 JEOSMPIE。当配置中断使能后, 发生相应的事件后, 产生相应的中断。

## 27.5 DMA

单周期扫描和连续扫描时通道转换的值存储在各自通道的数据寄存（ADDRn）中，最近一次转换的结果也会同步保存在 ADC\_ADDATA 寄存器中。DMA 传输时可以选择传输某个特定通道的数据，或者传输所有扫描通道的结果。

## 27.6 功能描述

### 27.6.1 时钟

ADC 的输入时钟与 APB2\_CLK 同步。在使用 ADC 之前，要先使能 RCC 控制器中的时钟使能控制位。

### 27.6.2 数据偏差校正

注入通道转换数据进行偏差校正后进入注入通道数据寄存器 ADC\_JADDATA 和 ADC\_JDR0~3:ADC\_JADDATA=SAR\_DATA- JOFR0~3.

### 27.6.3 数据对齐

通过配置 ADCR 寄存器中的 ALIGN 位，可以选择转换后数据储存为左对齐或右对齐。

非注入通道数据没有偏移值；注入组通道转换的数据值减去了在 ADC\_JOFRx 寄存器中定义的偏移量，因此结果可以是负值，SEXT 位是扩展的符号值。

如下图所示：

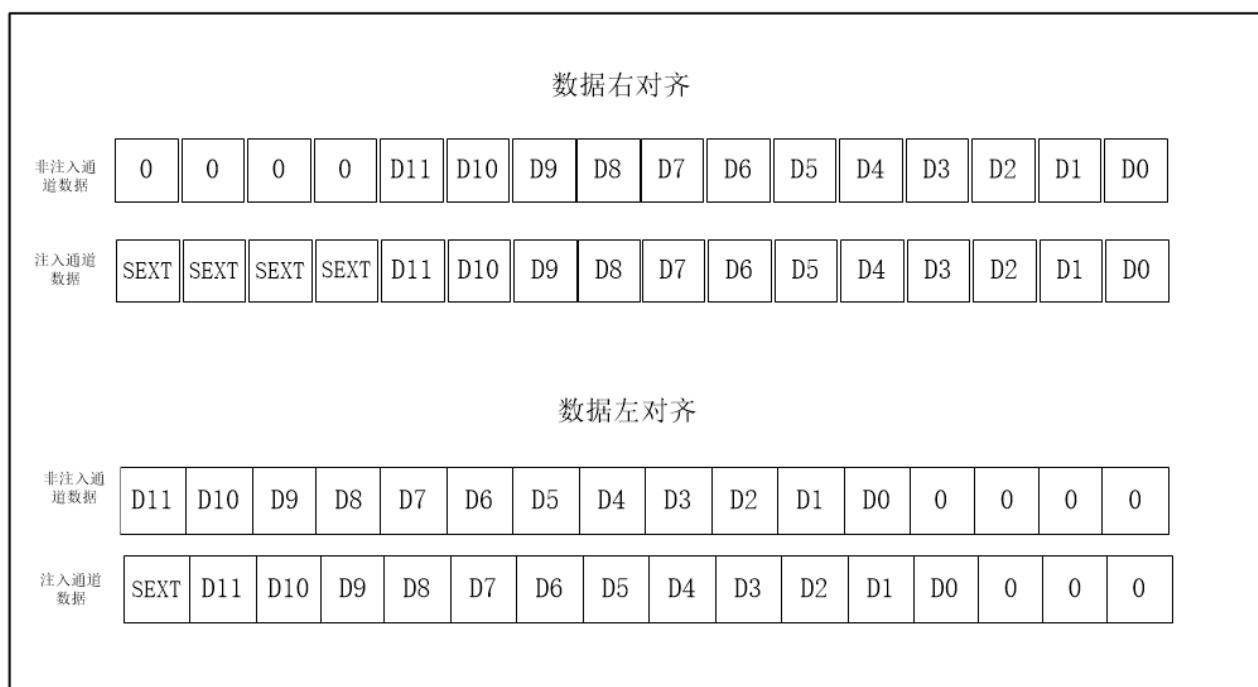


图 27-3 数据对齐方式

### 27.6.4 可编程分辨率

ADC 转换有效位数可通过 ADC\_ADCFG 寄存器中的 RSLTCTL[2:0]位更改，以便加快数据转换速率，有效数据位是在 12 位数据高位对齐。

## 27.6.5 可编程采样时间

ADC 转换采样时间可通过 ADC\_SMPR1~ADC\_SMPR2 寄存器中的 SAMPCTL0~SAMPCTL15 位配置每个通道的采样时间。

ADC 的时钟 ADC\_CLK 由 PCLK2 分频得到，分频系数可通过设置 ADC\_ADCFG 寄存器的 ADCPRE 位来确定，即  $PCLK2/(N+2)$  分频后作为 ADC 时钟。ADC 使用若干个 ADC\_CLK 周期对输入电压采样，采样周期数目可以通过 ADC\_ADCFG 寄存器中的 SAMCTL[3:0]位更改。设置 ADC 分辨率为 n 位 ( $n=8, 9, 10, 11, 12$ )，每个通道采样周期为 m。

采样频率采样时间计算如下：

$$F_{\text{sample}} = F_{\text{ADCLK}} / (m+n+0.5)。$$

假设分辨率配置为 12Bit，每个通道采样周期为 2.5T，则  $F_{\text{sample}}=F_{\text{ADC\_CLK}}/15$ 。总转换时间如下计算：

$T_{\text{CONV}}=\text{采样时间}+12.5$  个转换周期。

例如当 ADC\_CLK=15MHz，采样时间为 2.5 周期， $T_{\text{CONV}}=2.5+12.5=15$ ，总转换时间为  $1\mu\text{s}$ 。

## 27.6.6 数据通道寄存器

ADC 的转换完成后，非注入转换结果存储在寄存器 ADC\_ADDATA 中，CHANNELSEL 表示当前数据对应的通道号。

注入转换结果存储在寄存器 ADC\_JADDATA 中，JCHANNELSEL 表示当前数据对应的注入通道号。

## 27.6.7 通道选择

ADC1 有 14 路外部输入通道 0~13、内部温度传感器通道 14 和内部 1.2V 参考电压通道 15，ADC2 和 ADC3 有 16 路外部输入通道 0~15，每个外部输入通道都有独立的使能位，可通过设置 ADC\_ADCHS 寄存器（普通工作模式），设置 ADC\_ANY\_CFG、ADC\_CHANY0、ADC\_CHANY1（任意通道工作模式），设置 JSQR（注入通道工作模式）。

## 27.7 ADC 开关

ADC\_ADCFG 寄存器的 ADEN 位为 ADC 的控制开关，当 ADEN 位为 0 时，模拟转换单元进入掉电模式。

一个典型的工作流程如下：

通过设置 ADEN 位为 1 上电 ADC 模拟转换单元，延迟一段时间后设置 ADC\_ADCR 寄存器的 ADST 位，然后 ADC 就可以被有效的触发源启动转换。通过清除 ADST 位可以停止转换。注入通道设置 ADC\_ANY\_CR 寄存器的 JADST 位启动注入转换，清除 JADST 也可以停止转换。转换过程中，窗口看门狗开始工作，根据检测发生的事件产生相应中断。

### 27.7.1 普通工作模式

#### 27.7.1.1 单次采样

配置的通道执行一次转换操作后进入空闲模式，具体流程如下：

- 1) 通过软件、外部触发输入及定时器溢出置位 ADC\_ADCR 寄存器的 ADST，开始 A/D 转换；

- 2) A/D 转换完成时, 转换的数据值将存储于数据寄存器 ADC\_ADDATA 和 ADDRn 中;
- 3) A/D 转换完成时, 状态寄存器 ADC\_ADSTA 的 ADIF 位置 1, 若此时控制寄存器 ADC\_ADCR 的 ADIE 位置 1, 将产生 AD 转换结束中断请求;
- 4) A/D 转换期间, ADST 位保持为 1, A/D 通道转换结束以后, ADST 位自动清 0, A/D 转换器进入空闲模式。

注:在单次转换模式下, 如果软件使能多于一个通道, 序号最小的通道被转换, 其他通道被忽略。

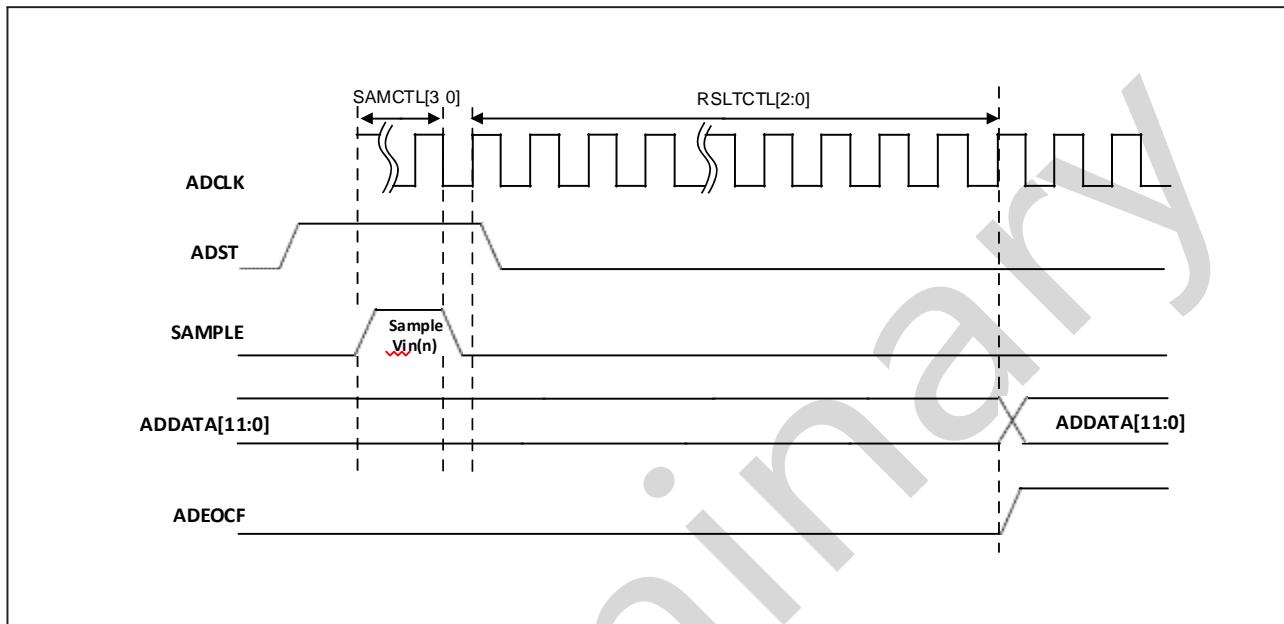


图 27-4 单次转换模式时序图

### 27.7.1.2 单周期扫描模式

在单周期扫描模式下, 将按使能的通道按顺序 (可通过配置 ADC\_ADCR 寄存器位 SCANDIR 选择扫描通道方向) 进行一次 A/D 转换, 操作步骤如下:

- 1) 软件或外部触发置位 ADST 开始, 外部触发可软件配置触发延时, 方向设置默认从最小序号通道到最大序号通道的 A/D 转换, 也可按照程序设置, 从最大序号通道到最小序号通道的 A/D 转换。
- 2) 每路 A/D 转换完成后, A/D 转换数值将有序装载到相应通道的数据寄存器中, ADIF 转换结束标志被设置, 如果设置了转换结束中断, 则在所有通道转换都完成后产生中断请求。
- 3) 最后一个 A/D 通道采样结束后, ADST 位自动清 0, A/D 转换器进入空闲状态。

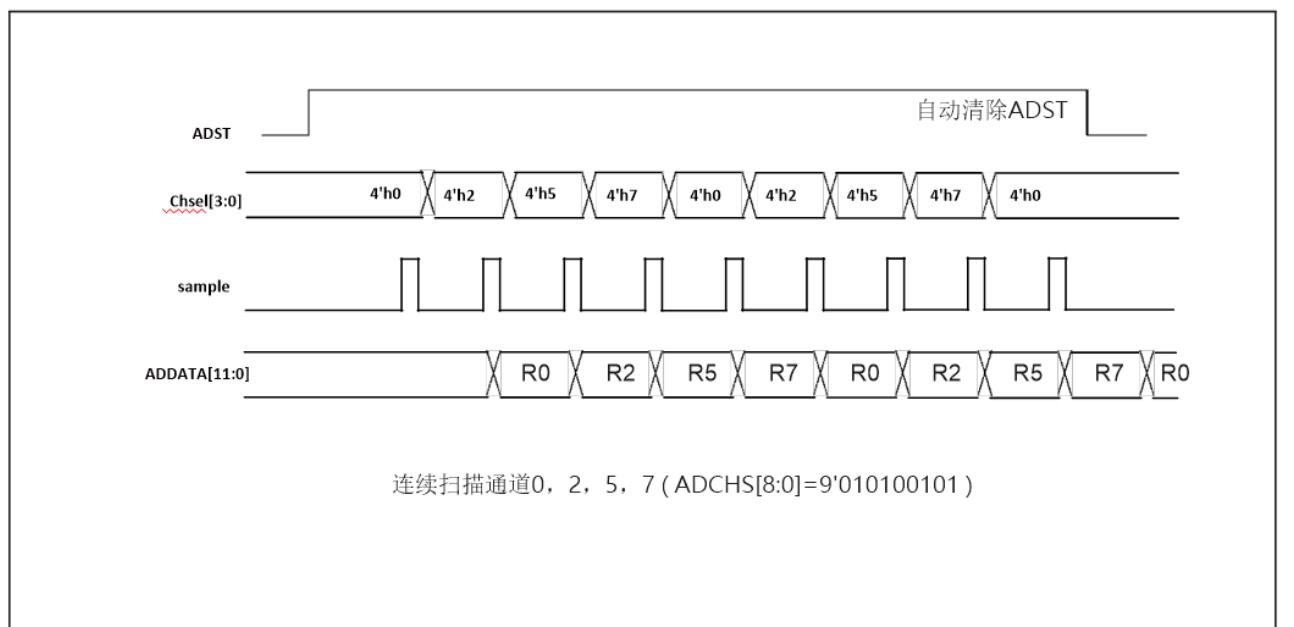


图 27-5 单周期扫描下使能通道转换时序图（通道方向从低到高）

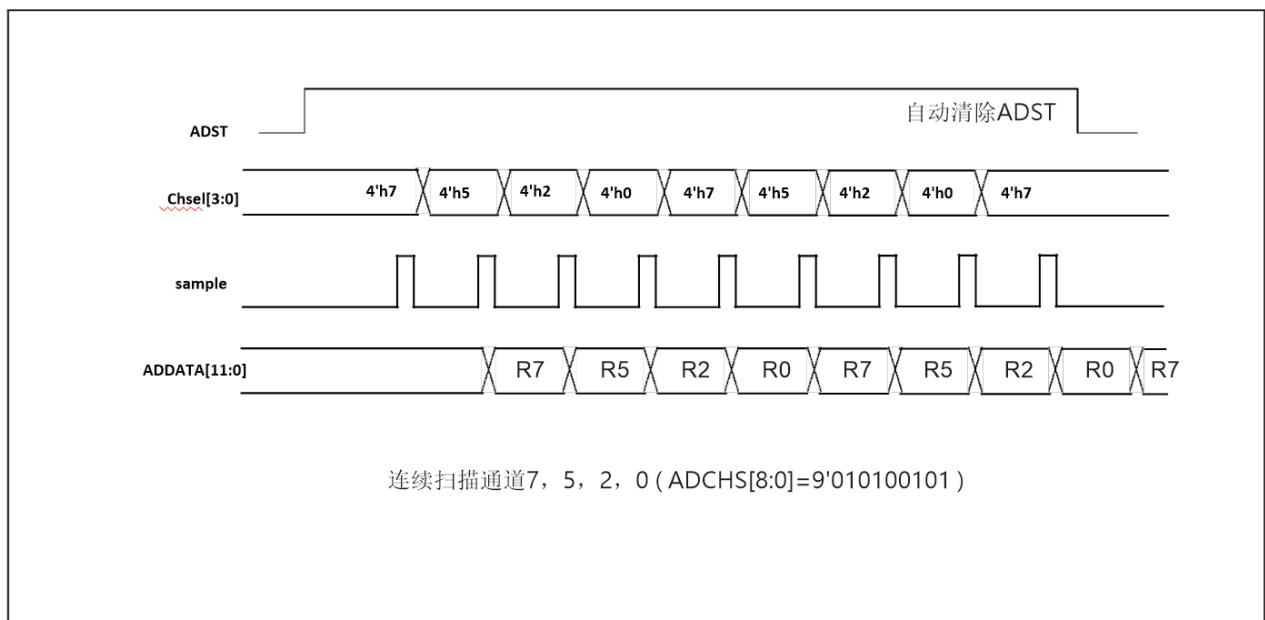


图 27-6 单周期扫描下使能通道转换时序图（通道方向从高到低）

### 27.7.1.3 连续扫描模式

在连续扫描模式下，A/D 转换在 ADC\_ADCHS 寄存器中的 CHENn 位被使能的通道上顺序进行（可通过配置寄存器 ADC\_ADCR 位 SCANDIR 选择扫描通道方向），操作步骤如下：

- 1) 软件或外部触发置位 ADST 开始，外部触发可软件配置触发延时，方向设置默认从最小序号通道到最大序号通道的 A/D 转换，也可按照程序设置，从最大序号通道到最小序号通道的 A/D 转换。
- 2) 当所有通道的 A/D 转换完成一遍后，A/D 转换数值将有序装载到相应的数据寄存器中，ADIF 转换结束标志被设置，如果设置了转换结束中断，则在所有通道转换都完成后产生中断请求。
- 3) 只要 ADST 位保持为 1，持续进行 A/D 转换。当 ADST 位被清 0，转换停止，进入空闲状态。当 ADST 清 0，A/D 转换将完成当前转换。

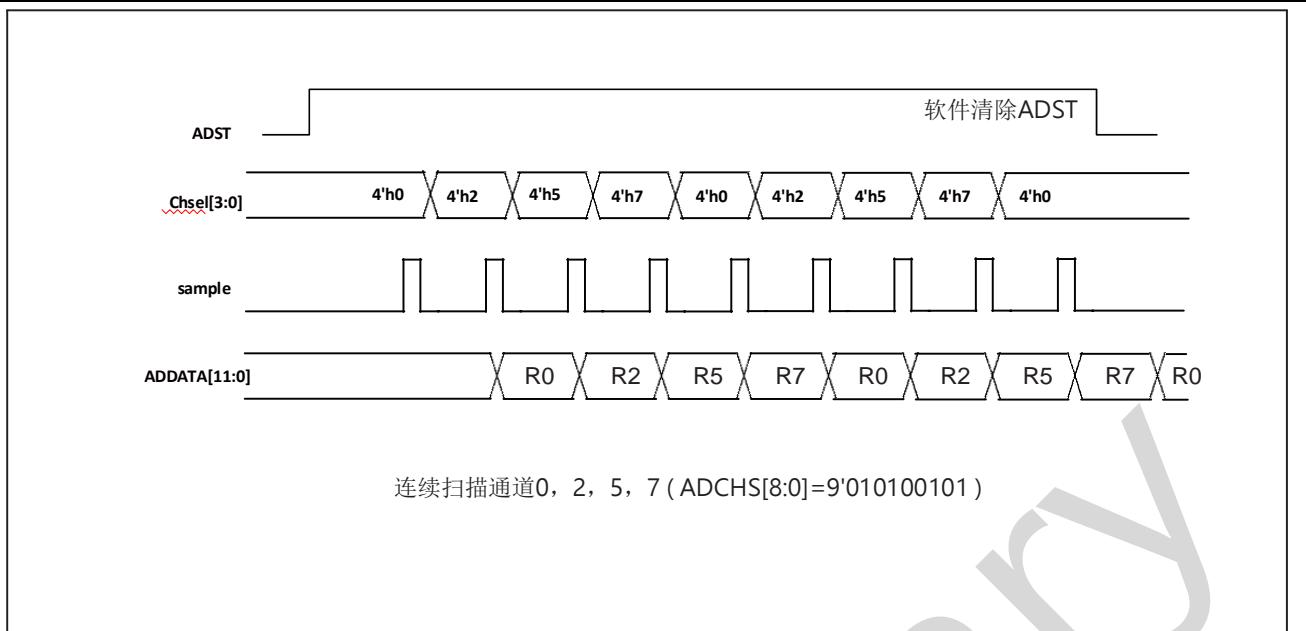


图 27-7 连续扫描模式使能通道转换时序图（通道方向由低到高）

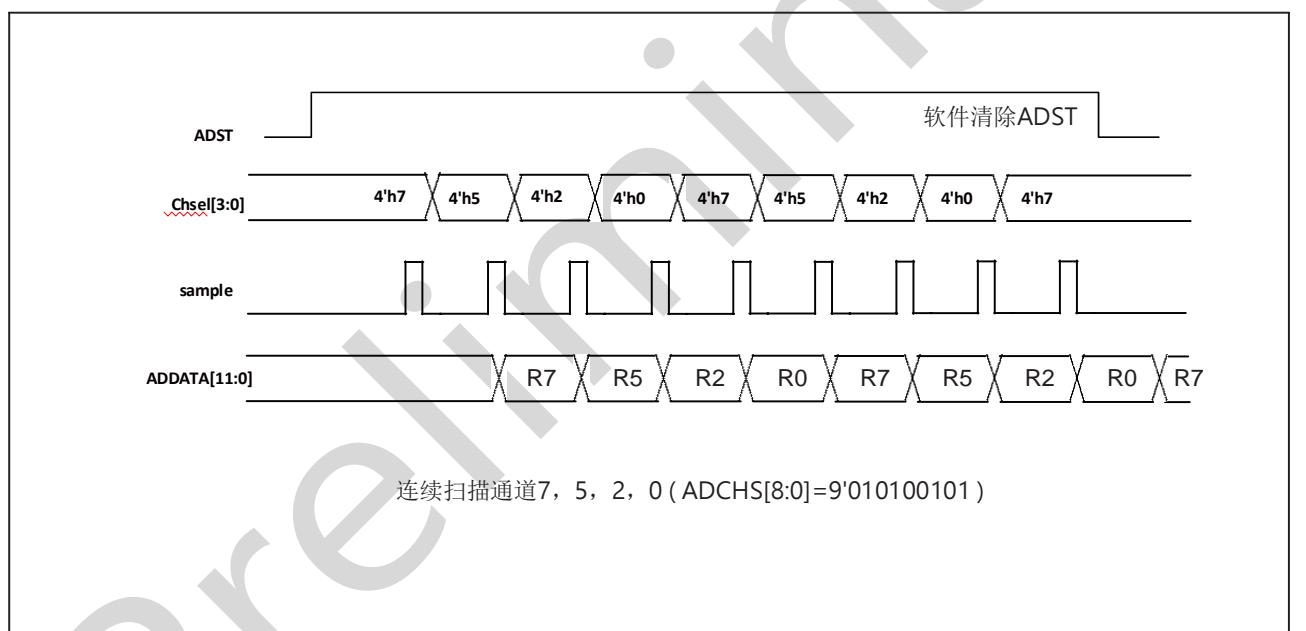


图 27-8 连续扫描模式使能通道转换时序图（通道方向由高到低）

## 27.7.2 任意通道工作模式

### 27.7.2.1 单次转换模式

单次转换模式的定义为，A/D 转换相应通道上只执行一次。

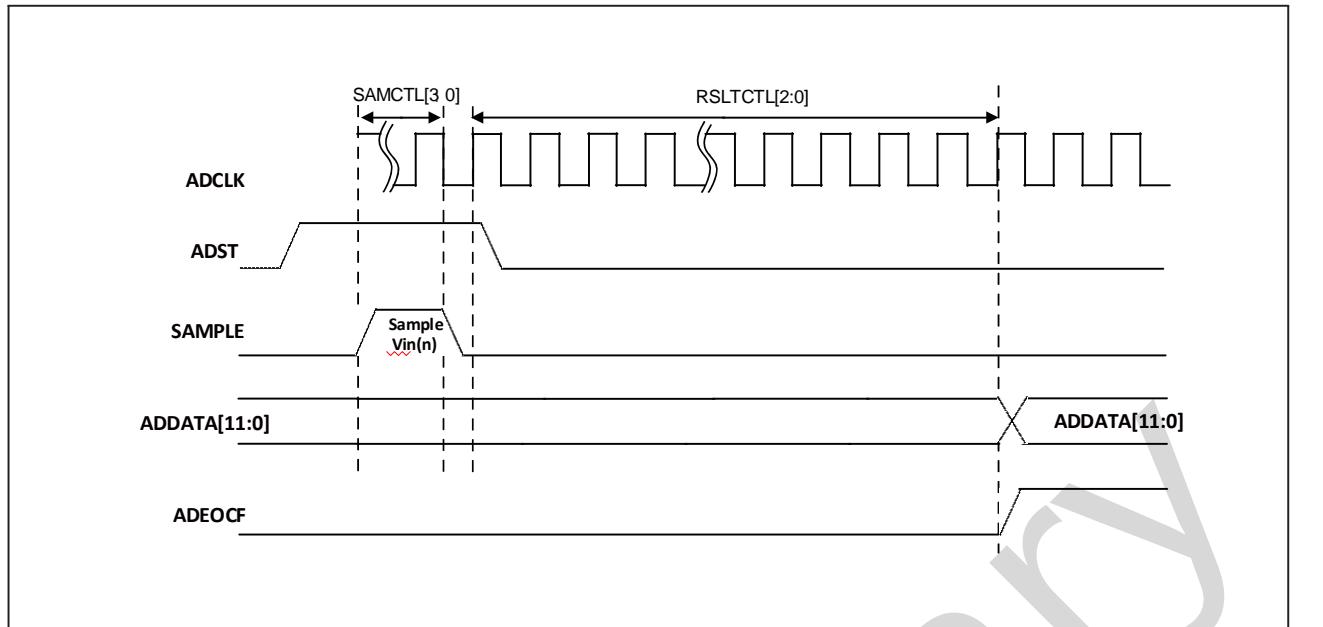


图 27-9 单次转换模式，通道转换时序图

### 27.7.2.2 单周期扫描模式

在单周期扫描模式下，A/D 转换通道按照软件配置执行一遍，具体流程如下：

- 1) 软件设置寄存器 `ADC_ANY_CFG`, `ADC_CHANY0`, `ADC_CHANY1`, 将需要转换的通道、数量设置好, 然后置位 `CHANY_MDEN`。
- 2) 通过软件、外部触发置位 `ADC_ADCR` 寄存器的 `ADST`, 外部触发可软件配置触发延时, A/D 转换方向从 `CHANY_SEL0` 到 `CHANY_SEL15`, 转换通道数量由 `CHANY_NUM` 配置, 且 `CHANY_SEL0` 到 `CHANY_SEL15` 是任意配置的, 可以完全相同, 或完全不相同。
- 3) 每路 A/D 转换完成时, A/D 转换的数据值将有序装载到相应通道的数据寄存器中, `ADIF` 转换结束标志被设置, 若此时控制寄存器 `ADC_ADCR` 的 `ADIE` 位置 1, 将产生 AD 转换结束中断请求。
- 4) A/D 最后一个通道采样结束后, `ADST` 位自动清 0, A/D 转换器进入空闲模式。
- 5) 若在 A/D 转换过程中, 软件更新 `ADC_ANY_CFG`, `ADC_CHANY0`, `ADC_CHANY1`, 硬件不会立即更新这些配置, 只会在当前设置的通道都转换结束时更新, 然后等待下一次软件置位 `ADST`。

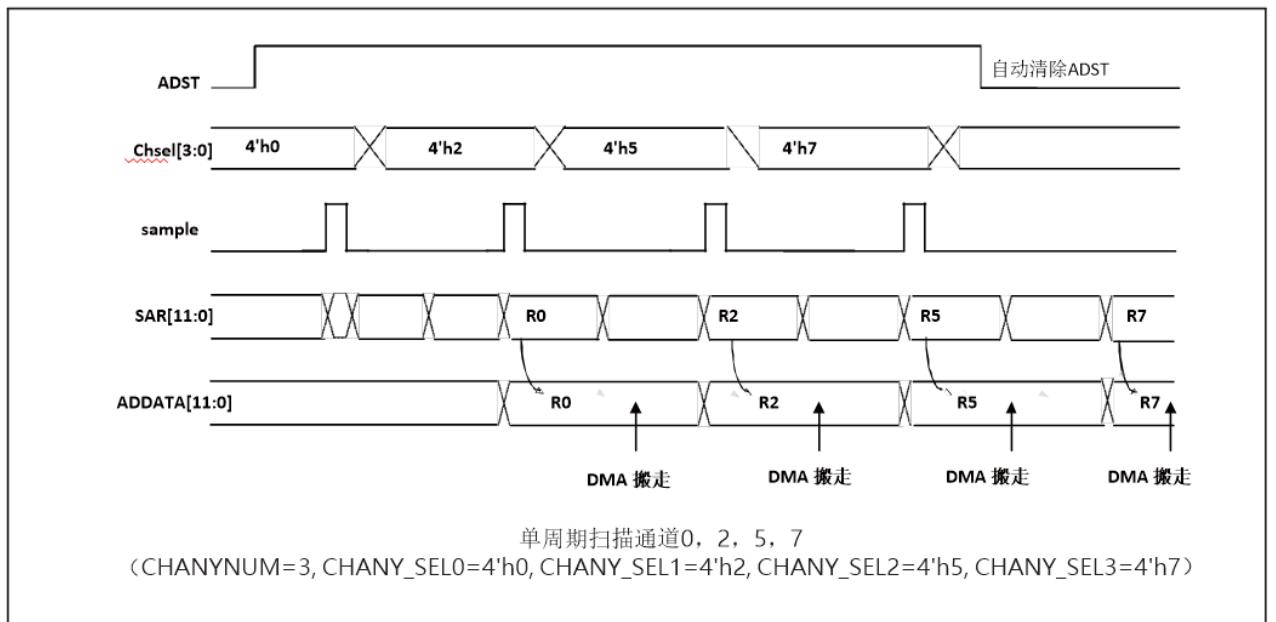


图 27-10 单周期扫描下通道转换时序图

### 27.7.2.3 连续扫描模式

在连续扫描模式下，A/D 转换通道按照软件配置一直执行，直到软件禁止。具体流程如下：

- 1) 软件设置寄存器 ADC\_ANY\_CFG, ADC\_CHANY0, ADC\_CHANY1, 将需要转换的通道、数量设置好，然后置位 CHANY\_MDEN。
- 2) 通过软件、外部触发置位 ADC\_ADCR 寄存器的 ADST，外部触发可软件配置触发延时，A/D 转换方向从 CHANY\_SEL0 到 CHANY\_SEL15，转换通道数量由 CHANY\_NUM 配置，且 CHANY\_SEL0 到 CHANY\_SEL15 是任意配置的，可以完全相同，或完全不相同。
- 3) 每路 A/D 转换完成时，A/D 转换的数据值将有序装载到相应通道的数据寄存器中，ADIF 转换结束标志被设置，若此时控制寄存器 ADC\_ADCR 的 ADIE 位置 1，将产生 AD 转换结束中断请求。
- 4) 只要 ADST 位保持为 1，持续进行 A/D 转换。当 ADST 位被清 0，当前 A/D 转换完成后停止，A/D 转换器进入空闲状态。
- 5) 若在 A/D 转换过程中，软件更新 ADC\_ANY\_CFG, ADC\_CHANY0, ADC\_CHANY1，硬件不会立即更新这些配置，只会在当前设置的通道都转换结束时更新，即下一个扫描周期开始新的通道转换。

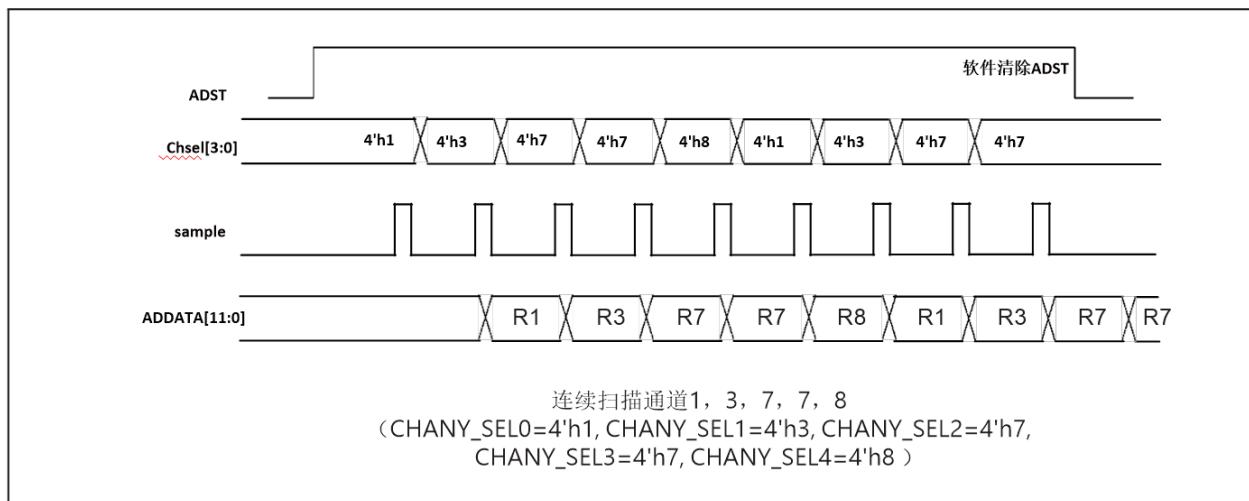


图 27-11 连续扫描模式，通道转换时序图

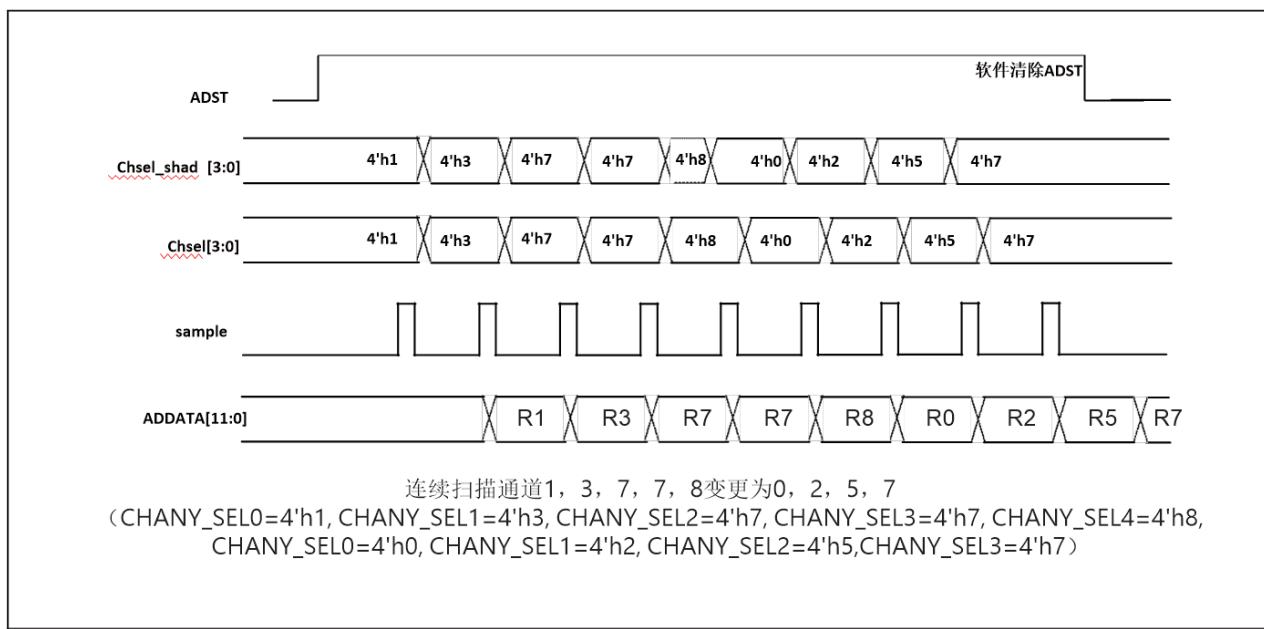


图 27-12 连续扫描模式，动态更新配置时序图

### 27.7.3 注入通道工作模式

#### 27.7.3.1 自动注入转换模式

任意通道配置通道转换完成后，自动开始注入通道转换。如果任意通道是连续扫描模式，需要清除ADST才能停止转换。

配置自动注入转换模式应禁止注入事件发生。

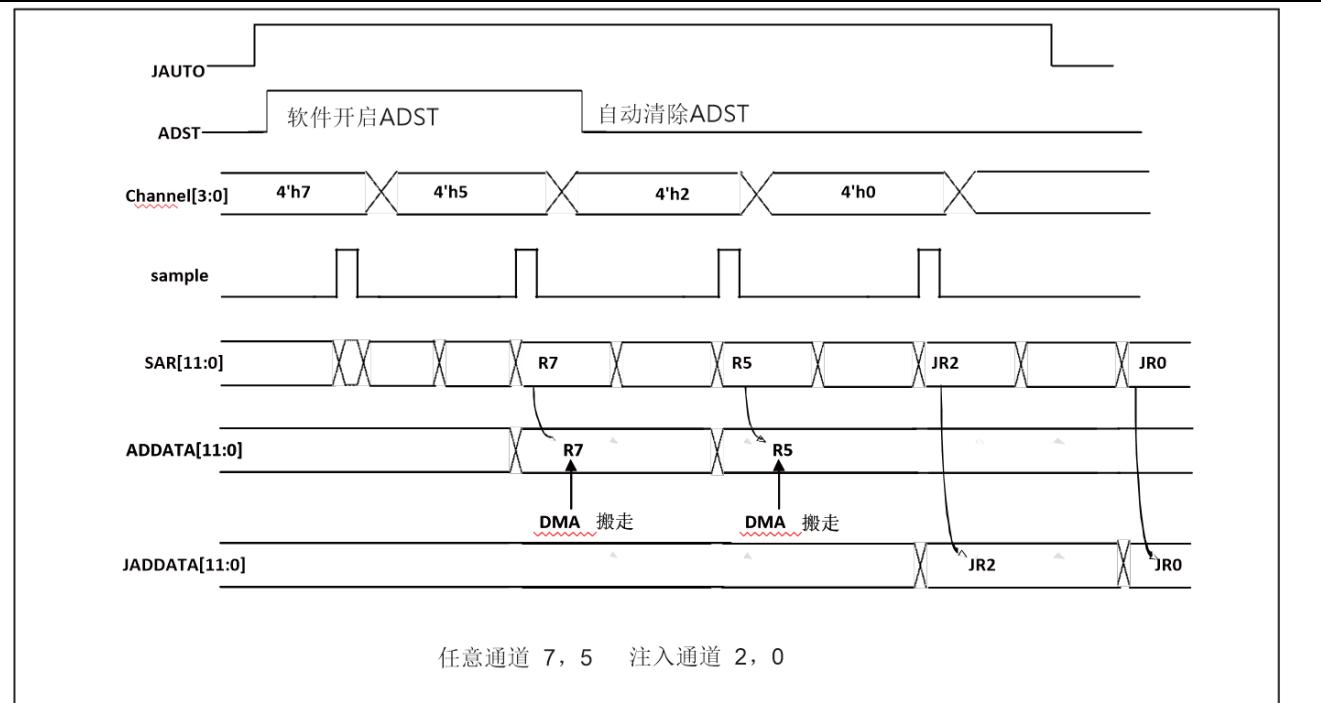


图 27-13 自动注入模式，通道转换时序图

### 27.7.3.2 事件注入工作模式

注入事件下禁止自动注入模式。注入事件发生后（包括软件和触发），如果当前转换正在进行，则完成

当前转换后开始注入通道转换。

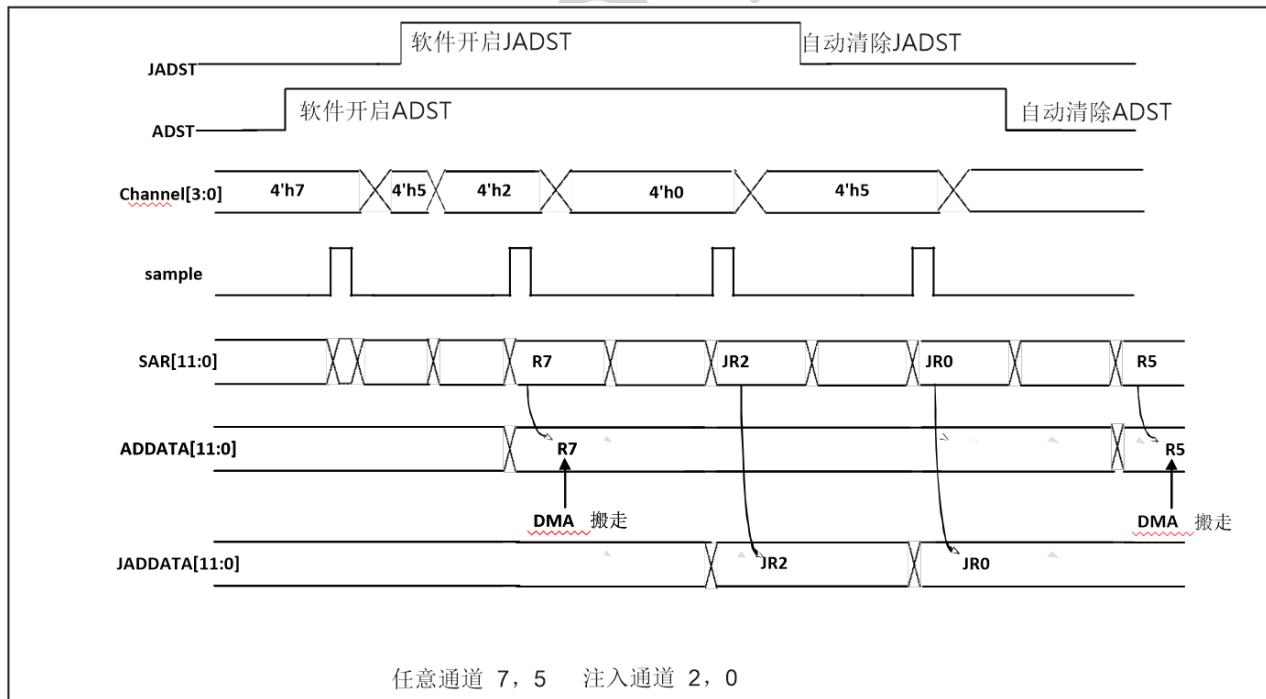


图 27-14 任意通道转换时注入事件转换通道时序图

在注入通道转换期间有任意通道事件产生，注入转换不会结束，但会保存任意通道事件。完成本次转换

序列后，开始任意通道事件的转换。

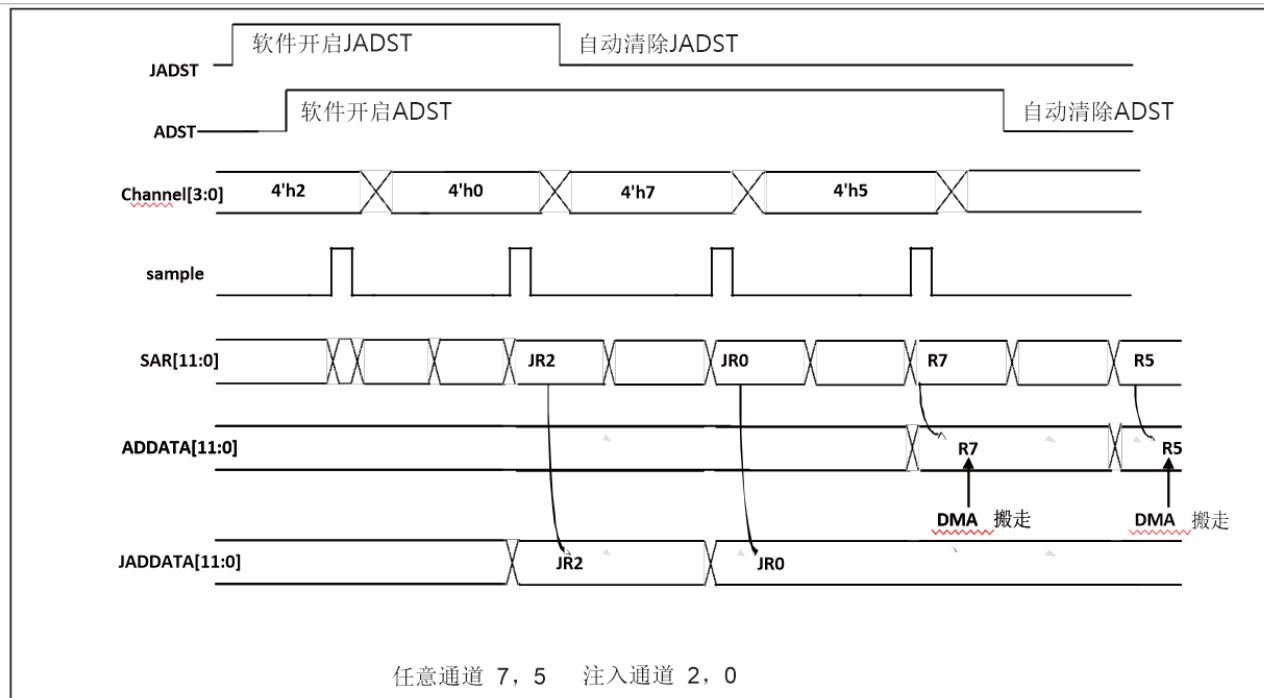


图 27-15 注入通道转换时任意事件转换通道发生时序图

使用触发开始注入通道转换，必须保证触发事件的间隔长与注入序列，注入通道转换期间发生注入事件将被忽略。

#### 27.7.4 ADC 触发信号

除了软件触发，ADC 转换的触发源还包括定时器和外部事件。

在触发信号产生后，延时 N 个 PCLK2 的时钟周期再开始采样。如果是触发扫描模式，只有第一个通道采样被延时，其余通道是在上一个采样结束后立即开始。

通过设置 ADC\_ADCR 寄存器的 TRGEN 位可以使用外部事件触发非注入通道转换；设置 ADC\_ANY\_CR 寄存器的 JTRGEN 位可以使用外部事件触发注入通道转换。

通过设置 ADC\_ADCR 寄存器的 TRGSEL 位可以选择非注入通道外部触发源；设置 ADC\_ANY\_CR 寄存器的 JTRGSEL 位可以选择注入通道外部触发源。

具体的外部触发源选择情况，可以参考 AD 控制寄存器（ADC\_ADCR.TRGSEL 或 ADC\_ANY\_CR.JTRGSEL）相关位的描述。外部触发可设置延时控制，具体参考 AD 控制寄存器（ADC\_ADCR.TRGSHIFT 或 ADC\_ANY\_CR.J TRGSHIFT）相关位的描述。

#### 27.7.5 模拟看门狗

比较模式下提供了上限和下限两个比较寄存器。可通过软件设定 CMPCH 位选择监控通道。当 CPMHDATA 大于或等于 CPMLDATA 时，比较结果大于或等于 ADC\_ADCMPR 寄存器的 CMPHDATA 指定

值或者小于 CMPLDATA 指定值，状态寄存器 ADC\_ADSTA 的 ADWIF 位置 1。

当 CPMHDATA 小于 CPMLDATA 时，比较结果如果等于 CPMHDATA 指定值或者处于两个指定值之间，则状态寄存器 ADC\_ADSTA 的 ADWIF 位置 1。

如果控制寄存器 ADC\_ADCR 的 ADWIE 置位，将产生中断请求。

## 27.7.6 内部温度传感器

内置的温度传感器仅用来检测器件内部的温度变化 (TA)。如果需要测量精确的温度，需要使用外置的温度传感器。

通过设置 ADC\_ADCFG 寄存器的 TSEN 位可以打开温度传感器，复位 TSEN 位可以单独关闭温度传感器。

通过设置 ADC\_ADCHS 寄存器的 CHENTS 位选择温度传感器通道。

温度数值计算如下：

$$T(\text{ }^{\circ}\text{C}) = (V_{\text{SENSE}} - V_{25}) / \text{Avg\_Slope} + 25$$

$V_{25}$ : 25 $\text{ }^{\circ}\text{C}$  时的  $V_{\text{SENSE}}$  值；

$V_{\text{SENSE}}$ : 温度传感器当前的输出电压；

$V_{\text{SENSE}} = \text{Value} * V_{\text{dd}} / 4096$  (Value 是 ADC 的转换结果数据)；

Avg\_Slope: 温度与  $V_{\text{SENSE}}$  曲线的平均斜率 (以 mV/ $^{\circ}\text{C}$  或  $\mu\text{V}/^{\circ}\text{C}$  表示)；

$V_{25}$  和 Avg\_Slope 的典型值请参考数据手册温度传感器章节。

## 27.7.7 内部电压传感器

ADC 的内部信号源通道连接了一个内部基准参考电压，大小为 1.2V，此通道把 1.2V 的参考电压输出转换为数字值。

内部参考电压有单独的使能位，可通过设置寄存器的相应位开启或关闭。

## 27.8 寄存器描述

表 27-1 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	ADC_ADDATA	A/D 数据寄存器	0x00000000
0x04	ADC_ADCFG	A/D 配置寄存器	0x00000000
0x08	ADC_ADCR	A/D 控制寄存器	0x00000000
0x0C	ADC_ADCHS	A/D 通道选择寄存器	0x00000000
0x10	ADC_ADCMPR	A/D 窗口比较寄存器	0x00000000
0x14	ADC_ADSTA	A/D 状态寄存器	0x00000000
0x18~0x3C	ADC_ADDR 0 ~ 9	A/D 数据寄存器	0x00000000

Offset	Acronym	Register Name	Reset
0x50~0x54	ADC_ADDR	A/D 数据寄存器	0x00000000
0x58	ADC_ADSTA_EXT	A/D 扩展状态寄存器	0x00000000
0x5C	ADC_CHANY0	A/D 任意通道通道选择寄存器 0	0x00000000
0x60	ADC_CHANY1	A/D 任意通道通道选择寄存器 1	0x00000000
0x64	ADC_ANY_CFG	A/D 任意通道配置寄存器	0x00000000
0x68	ADC_ANY_CR	A/D 任意通道控制寄存器	0x00000000
0x70	ADC_SMPR1	A/D 采样配置寄存器 1	0x00000000
0x74	ADC_SMPR2	A/D 采样配置寄存器 2	0x00000000
0x7C~0x88	ADC_JOFR0~3	A/D 注入通道数据补偿寄存器	0x00000000
0x8C	ADC_JSQR	A/D 注入通道连续寄存器	0x00000000
0x90	ADC_JADDATA	A/D 注入通道数据寄存器	0x00000000
0xB0~0xBC	ADC_JDR0~3	A/D 注入通道数据寄存器	0x00000000

## 27.8.1 A/D 数据寄存器 (ADC\_ADDATA)

偏移地址:0x00

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.												VAILD	OVER RUN	CHANNELSEL	
Type													r	r		
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	DATA															
Type	r															

Bit	Field	Type	Reset	Description
31:22	Reserved		0x0	保留, 始终读为 0
21	VALID	r	0x00	有效标志位 (只读) (Valid flag) 1= DATA[11:0] 位数据有效 0= DATA[11:0] 位数据无效 相应模拟通道转换完成后, 将该位置位, 读 ADC_ADDATA 寄存器后, 该位由硬件清除。
20	OVERRUN	r	0x00	数据覆盖标志位 (只读) (Overrun flag) 1= DATA[11:0] 数据被覆盖 0= DATA[11:0] 数据最近一次转换结果。

Bit	Field	Type	Reset	Description
				新的转换结果装载至寄存器之前，若 DATA[11:0] 的数据没有被读取，OVERRUN 将置 1。读 ADC_ADDATA 寄存器后，该位由硬件清除。
19:16	CHANNELSEL	r	0x00	<p>该 4 位显示当前数据所对应的通道 (Channel selection)</p> <p>0000 = 通道 0 的转换数据      0001 = 通道 1 的转换数据      0010 = 通道 2 的转换数据      0011 = 通道 3 的转换数据      0100 = 通道 4 的转换数据      0101 = 通道 5 的转换数据      0110 = 通道 6 的转换数据      0111 = 通道 7 的转换数据      1000 = 通道 8 的转换数据      1001 = 通道 9 的转换数据      1010 = 通道 10 的转换数据      1011 = 通道 11 的转换数据      1100 = 通道 12 的转换数据      1101 = 通道 13 的转换数据      1110 = 温度传感器的转换数据      1111 = 内部参考电压的转换数据      其他:无效   </p>
15:0	DATA	r	0x00	12 位 A/D 转换结果 (Transfer data) 根据设置左对齐或者右对齐.

## 27.8.2 A/D 配置寄存器 (ADC\_ADCFG)

偏移地址:0x04

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Res.																JADWEN
Type																	rw
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	Res.	ADCPRE	Res.			RSLTCTL			ADCPRE			VSEN	TSEN	ADWEN	ADEN		

Type		rw		rw	rw	rw	rw	rw	rw
Bit	Field	Type	Reset	Description					
31:17	Reserved		0x0	保留, 始终读为 0					
16	JADWEN	rw	0x00	A/D 注入通道窗口比较器使能 (ADC Inject Channel window comparison enable) 1=注入通道 A/D 窗口比较器使能 0=注入通道 A/D 窗口比较器禁用					
15	Reserved		0x0	保留, 始终读为 0					
14	ADCPRE	rw	0x00	ADC 预分频 (ADC prescaler) 作为 ADCPRE[3:0] 的最低位, 与 Bit[6:4] 合用					
13 : 10	Reserved		0x00	保留, 始终读为 0					
9: 7	RSLTCTL	rw	0x00	选择 ADCx 转换数据分辨率 (resolution) 000:12 位有效 001:11 位有效 010:10 位有效 011:9 位有效 100:8 位有效					
6: 4	ADCPRE	rw	0x00	ADC 预分频 (ADC prescaler) 由软件置'1'或清'0'来确定 ADC 时钟频率。 PCLK2 的([6:4], [14])+2)分频作为 ADC 时钟					
3	VSEN	rw	0x00	内部参考电压使能 (Voltage Sensor enable) 1:内部电压传感器使能 0:内部电压传感器禁用					
2	TSEN	rw	0x00	温度传感器使能控制位 (Temperature sensor enable) 1=温度传感器使能 0=温度传感器禁止					

Bit	Field	Type	Reset	Description
1	ADWEN	rw	0x00	<p>非注入通道 A/D 窗口比较器使能(ADC window comparison enable)</p> <p>1=非注入通道 A/D 窗口比较器使能</p> <p>0=非注入通道 A/D 窗口比较器禁用</p>
0	ADEN	rw	0x00	<p>A/D 转换使能 (ADC enable)</p> <p>1= 使能</p> <p>0= 禁用</p>

### 27.8.3 A/D 控制寄存器 (ADC\_ADCR)

偏移地址:0x08

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.				EO CIE	EO SM	TRG_ED GE	Res.		TRGSHIFT			TRGSEL		SCAN DIR	
Type					rw	rw	rw		rw		rw			rw		
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CMPCH				ALI GN	ADMD		AD ST	Re s.	TRGSEL			DMA EN	TRG EN	ADW IE	ADIE
Type	rw				rw	rw		rw		rw			rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:28	Reserved		0x0	保留, 始终读为 0
27	EOCIE	rw	0x00	<p>A/D 非注入通道转换完成中断使能 (ADC end of convert interrupt enable)</p> <p>1=使能 A/D 单次转换完成中断</p> <p>0=禁用 A/D 单次转换完成中断</p> <p>如果 EOCIF 置位, A/D 转换结束后产生中断请求。</p>
26	EOSMPIE	rw	0x00	<p>A/D 非注入通道采样完成中断使能 (ADC end of sample interrupt enable)</p> <p>1=使能 A/D 采样完成中断</p> <p>0=禁用 A/D 采样完成中断</p> <p>如果 EOSMPIF 置位, A/D 采样结束后产生中断请求。</p>

Bit	Field	Type	Reset	Description
25 : 24	TRG_EDGE	rw	0x00	<p>触发边沿选择</p> <p>00:双沿触发 01:下沿触发 10:上沿触发 11:屏蔽触发</p>
23:22	Reserved		0x0	保留, 始终读为 0
21 : 19	TRGSHIFT	rw	0x00	<p>外部触发延时采样 (External trigger shift sample)</p> <p>在触发信号产生后, 延时 N 个 PCLK2 的时钟周期再开始采样。</p> <p>如果是触发扫描模式, 其他通道是在上一个采样结束后立即开始。</p> <p>0:不延时 1:4 个周期 2:16 个周期 3:32 个周期 4:64 个周期 5:128 个周期 6:256 个周期 7:512 个周期</p>
18 :17	TRGSEL	rw	0x00	<p>外部触发源选择 (External trigger selection)</p> <p>与 Bit[6:4]合用</p>
16	SCANDIR	rw	0x00	<p>ADC 扫描通道顺序 (ADC scan direction)</p> <p>在单周期扫描或者连续扫描方式时, 设置扫描通道的顺序</p> <p>0:ADC 通道选择寄存器按从低到高的顺序扫描 1:ADC 通道选择寄存器按从高到低的顺序扫描</p>
15 : 12	CMPCH	rw	0x00	<p>窗口比较通道选择 (Window comparison channel selection)</p> <p>0000 = 选择比较通道 0 转换结果 0001 = 选择比较通道 1 转换结果 0010 = 选择比较通道 2 转换结果 0011 = 选择比较通道 3 转换结果</p>

Bit	Field	Type	Reset	Description
				<p>0100 = 选择比较通道 4 转换结果      0101 = 选择比较通道 5 转换结果      0110 = 选择比较通道 6 转换结果      0111 = 选择比较通道 7 转换结果      1000 = 选择比较通道 8 转换结果      1001 = 选择比较通道 9 转换结果      1010 = 选择比较通道 10 转换结果      1011 = 选择比较通道 11 转换结果      1100 = 选择比较通道 12 转换结果      1101 = 选择比较通道 13 转换结果      1110 = 选择比较温度传感器转换结果      1111 = 所有扫描通道其他:无效</p>
11	ALIGN	rw	0x00	<p>数据对齐 (Data alignment)      0:右对齐      1:左对齐</p>
10 : 9	ADMD	rw	0x00	<p>A/D 转换模式 (ADC mode)      00:单次转换      01:单周期扫描      10:连续扫描      当改变转换模式时，软件要先禁用 ADST 位。</p>
8	ADST	rw	0x00	<p>A/D 转换开始 (ADC start)      1=转换开始      0=转换结束或进入空闲状态      ADST 清除有下列两种方式:      在单次模式或者单周期模式下，转换完成后，ADST 将被硬件自动清除。      在连续扫描模式下，A/D 转换将一直进行，直到软件写'0'到该位或系统复位</p>
7	Reserved			保留，始终读为 0。

Bit	Field	Type	Reset	Description
6: 4	TRGSEL	rw	0x00	<p>外部触发源选择 (External trigger selection), 位 [18:17, 6:4]</p> <p>ADC1、ADC2、ADC3 选择外部触发源</p> <ul style="list-style-type: none"> <li>00000:TIM1_CC1</li> <li>00001:TIM1_CC2</li> <li>00010:TIM1_CC3</li> <li>00011:TIM2_CC2</li> <li>00100:TIM3_TRGO</li> <li>00101: TIM1_CC4 或 TIM1_CC5</li> <li>00110:TIM3_CC1</li> <li>00111:EXTI 线 11</li> <li>01000:TIM1_TRGO</li> <li>01001:TIM8_CC4</li> <li>01010:TIM8_CC4 或 TIM8_CC5</li> <li>01011:TIM2_CC1</li> <li>01100:TIM3_CC4</li> <li>01101:TIM2_TRGO</li> <li>01111:EXTI 线 15</li> <li>10000:TIM1_CC4</li> <li>10001:TIM1_CC5</li> <li>其他:无效</li> </ul>
3	DMAEN	rw	0x00	<p>DMA 使能 (Direct memory access enable)</p> <p>1= DMA 请求使能</p> <p>0= DMA 禁止</p>
2	TRGEN	rw	0x00	<p>外部硬件触发源 (External trigger enable) 1=使用外部触发信号启动 A/D 转换</p> <p>0=不用外部触发信号启动 A/D 转换</p>
1	ADWIE	rw	0x00	<p>A/D 窗口比较器中断使能 ( ADC window comparator interrupt enable)</p> <p>1=使能 A/D 窗口比较器中断</p> <p>0=禁用 A/D 窗口比较器中断</p>

Bit	Field	Type	Reset	Description
0	ADIE	rw	0x00	A/D 非注入通道组转换中断使能 (ADC interrupt enable) 1=使能 A/D 中断 0=禁用 A/D 中断 如果 ADIF 置位，A/D 转换结束后产生中断请求。

## 27.8.4 A/D 通道选择寄存器 (ADC\_ADCHS)

偏移地址:0x0C

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CH ENV S	CH EN TS	CH EN 13	CH EN 12	CH EN 11	CH EN 10	CH EN 9	CH EN 8	CH EN 7	CH EN 6	CH EN 5	CH EN 4	CH EN 3	CH EN 2	CH EN 1	CH EN 0
Type	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:16	Reserved		0x0	保留, 始终读为 0
15	CHENVS	rw	0x00	ADC2/3: 模拟输入通道 15 使能 (Analog input channel 15 enable) ADC1: 内部参考电压使能 (Voltage Sensor enable) 1=使能 0=禁用
14	CHENTS	rw	0x00	ADC2/3: 模拟输入通道 14 使能 (Analog input channel 14 enable) ADC1: 温度传感器使能 (Temperature Sensor enable) 1=使能 0=禁用
13	CHEN13	rw	0x00	模拟输入通道 13 使能 (Analog input channel 13 enable) 1=使能 0=禁用

Bit	Field	Type	Reset	Description
12	CHEN12	rw	0x00	模拟输入通道 12 使能 (Analog input channel 12 enable) 1=使能 0=禁用
11	CHEN11	rw	0x00	模拟输入通道 11 使能 (Analog input channel 11 enable) 1=使能 0=禁用
10	CHEN10	rw	0x00	模拟输入通道 10 使能 (Analog input channel 10 enable) 1=使能 0=禁用
9	CHEN9	rw	0x00	模拟输入通道 9 使能 (Analog input channel 9 enable) 1=使能 0=禁用
8	CHEN8	rw	0x00	模拟输入通道 8 使能 (Analog input channel 8 enable) 1=使能 0=禁用
7	CHEN7	rw	0x00	模拟输入通道 7 使能 (Analog input channel 7 enable) 1=使能 0=禁用
6	CHEN6	rw	0x00	模拟输入通道 6 使能 (Analog input channel 6 enable) 1=使能 0=禁用
5	CHEN5	rw	0x00	模拟输入通道 5 使能 (Analog input channel 5 enable) 1=使能 0=禁用
4	CHEN4	rw	0x00	模拟输入通道 4 使能 (Analog input channel 4 enable) 1=使能 0=禁用
3	CHEN3	rw	0x00	模拟输入通道 3 使能 (Analog input channel 3 enable) 1=使能 0=禁用
2	CHEN2	rw	0x00	模拟输入通道 2 使能 (Analog input channel 2 enable) 1=使能 0=禁用

Bit	Field	Type	Reset	Description
1	CHEN1	rw	0x00	模拟输入通道 1 使能 (Analog input channel 1 enable) 1=使能 0=禁用
0	CHEN0	rw	0x00	模拟输入通道 0 使能 (Analog input channel 0 enable) 1=使能 0=禁用

注: 如果通道使能都为 0, 则通道 0 使能。ADC\_ADSTA 的 BUSY 有效时无法操作此寄存器。

### 27.8.5 A/D 窗口比较寄存器 (ADC\_ADCMPR)

偏移地址:0x10

复位值:0x00000000

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

Bit	Field	Type	Reset	Description
31:28	Reserved		0x0	保留, 始终读为 0
27 : 16	CMPHDATA	rw	0x00	比较数值上限 (Compare data high limit) 该 12 位数值将和指定通道的转换结果相比较。
15:12	Reserved		0x0	保留, 始终读为 0
11 : 0	CMPLDATA	rw	0x00	比较数值下限 (Compare data low limit) 该 12 位数值将和指定通道的转换结果相比较。

### 27.8.6 A/D 状态寄存器 (ADC\_ADSTA)

偏移地址:0x14

复位值:0x00000000

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

Field	OVERRUN												VALID				
Type	r												r				
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	VALID								CHANNEL				Res.	BUSY	ADWIF	ADIF	
Type	r								r				r	rc_w1	rc_w1		

Bit	Field	Type	Reset	Description
31:20	OVERRUN	r	0x00	通道 0 ~ 11 的数据覆盖标志位 (Overrun flag)
19:8	VALID	r	0x00	通道 0 ~ 11 的有效标志位 (Valid flag)
7:4	CHANNEL	r	0x00	当前转换通道 (Current conversion channel) 该 4 位在 BUSY=1 时表示进行转换中的通道。 BUSY=0 时表示可进行下次转换的通道。
3	Reserved		0x00	保留, 始终读为 0
2	BUSY	r	0x00	非注入通道忙/空闲 (Busy) 1= A/D 转换器忙碌 0= A/D 转换器空
1	ADWIF	rc_w1	0x00	比较标志位 (ADC window comparator interrupt flag) 当 CPMHDATA 大于等于 CPMLDATA 时, 选择的 A/D 转换通道比较结果大于或等于 ADCMPR 寄存器的 CMPHDATA 指定值或者小于 CPMLDATA 指定值, 状态寄存器 ADSTA 的 ADWIF 位置 1。 当 CPMHDATA 小于 CPMLDATA 时, 选择的 A/D 转换通道比较结果如果等于 CPMHDATA 指定值或者处于两个指定值之间, 则状态寄存器 ADSTA 的 ADWIF 位置 1。 该标志位写'1'清零。

Bit	Field	Type	Reset	Description
0	ADIF	rc_w1	0x00	A/D 通道组转换结束标志位 (ADC convert complete flag) 该位由硬件在通道组转换结束时设置，由软件清除。 1= A/D 转换完成 0= A/D 转换未完成 该标志位写'1'清零。

## 27.8.7 A/D 数据寄存器 (ADC\_ADDR0~15)

偏移地址:0x18 ~ 0x54

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.												VAILD	OVERRUN	Res.	
Type													r	r		
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	DATA															
Type	r															

Bit	Field	Type	Reset	Description
31:22	Reserved		0x0	保留, 始终读为 0
21	VALID	r	0x00	有效标志位 (只读) (Valid flag) 1= DATA[11:0]位数据有效 0= DATA[11:0]位数据无效 相应模拟通道转换完成后, 将该位置位, 读 ADC_ADDR 寄存器后, 该位由硬件清除。
20	OVERRUN	r	0x00	数据覆盖标志位 (只读) (Overrun flag) 1= DATA [11:0]数据被覆盖 0= DATA [11:0]数据最近一次转换结果 新的转换结果装载至寄存器之前, 若 DATA[11:0]的数据没有被读取, OVERRUN 将置'1', 读 ADC_ADDR 寄存器后, 该位由硬件清除。
19 : 16	Reserved			保留, 始终读为 0。

Bit	Field	Type	Reset	Description
15 : 0	DATA	r	0x00	通道的 12 位 A/D 转换结果 (Transfer data) 根据设置左对齐或者右对齐。

## 27.8.8 A/D 扩展状态寄存器 (ADC\_ADSTA\_EXT)

偏移地址:0x58

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
Field	Res.												JBU	JEOS	JEO	JEOS	EO	EOS
Type													SY	IF	CIF	MPIF	CIF	MPIF
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
Field	Res.												OVERRUN	VALID				
Type									r					r				

Bit	Field	Type	Reset	Description
31:8	Reserved		0x0	保留, 始终读为 0
21	JBUSY	r	0x00	注入通道忙/空闲 (Inject Channel Busy) 1= A/D 转换器忙碌 0= A/D 转换器空
20	JEOSIF	r	0x00	A/D 注入通道连续转换结束标志位 (ADC Inject Channel End of Sequential Convert flag) 该位由硬件在通道组转换结束时设置, 由软件清除。 1= A/D 转换完成 0= A/D 转换未完成 该标志位写'1'清零。
19	JEOCIF	r	0x00	A/D 注入通道转换结束标志位 (ADC Inject Channel End of Convert flag) 该位由硬件在通道转换完成时设置, 由软件清除。 1= A/D 转换完成 0= A/D 转换未完成 该标志位写'1'清零。

Bit	Field	Type	Reset	Description
18	JEOSMPIF	r	0x00	<p>A/D 注入通道采样结束标志位 (Inject Channel ADC End of Sample flag)</p> <p>该位由硬件在通道采样结束时设置，由软件清除。</p> <p>1= A/D 采样转换完成 0= A/D 采样未完成</p> <p>该标志位写'1'清零。</p>
17	EOCIF	r	0x00	<p>A/D 转换结束标志位 (ADC End of convert flag)</p> <p>该位由硬件在通道转换结束时设置，由软件清除。</p> <p>1= A/D 转换完成 0= A/D 转换未完成</p> <p>该标志位写'1'清零。</p>
16	EOSMPIF	r	0x00	<p>A/D 采样结束标志位 (ADC End of Sample flag)</p> <p>该位由硬件在通道组转换结束时设置，由软件清除。</p> <p>1= A/D 采样转换完成 0= A/D 采样未完成</p> <p>该标志位写'1'清零。</p>
15:8	Reserved		0x00	保留，始终读为 0。
7: 4	OVERRUN	r	0x00	<p>通道的数据覆盖标志位 (Overrun flag)</p> <p>1000: ADC2/3 通道 15 ADC1 内部参考电压 (V_SENSOR)</p> <p>0100: ADC2/3 通道 14 ADC1 温度传感器 (T_SENSOR)</p> <p>0010: 通道 13</p> <p>0001: 通道 12</p>

Bit	Field	Type	Reset	Description
3:0	VALID	r	0x00	通道的有效标志位 (Valid flag) 1000: ADC2/3 通道 15 ADC1 内部参考电压 (V_SENSOR) 0100: ADC2/3 通道 14 ADC1 温度传感器 (T_SENSOR) 0010: 通道 13 0001: 通道 12

### 27.8.9 A/D 任意通道通道选择寄存器 0 (ADC\_CHANY0)

偏移地址:0x5C

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	CHANY_SEL7				CHANY_SEL6				CHANY_SEL5				CHANY_SEL4			
Type	rw															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CHANY_SEL3				CHANY_SEL2				CHANY_SEL1				CHANY_SEL0			
Type	rw															

Bit	Field	Type	Reset	Description
31:28	CHANY_SEL7	rw	0x00	可配置为通道 0~通道 15 中的任意通道。
27:24	CHANY_SEL6	rw	0x00	可配置为通道 0~通道 15 中的任意通道。
23:20	CHANY_SEL5	rw	0x00	可配置为通道 0~~通道 15 中的任意通道。
19:16	CHANY_SEL4	rw	0x00	可配置为通道 0~通道 15 中的任意通道。
15:12	CHANY_SEL3	rw	0x00	可配置为通道 0~通道 15 中的任意通道。
11:8	CHANY_SEL2	rw	0x00	可配置为通道 0~通道 15 中的任意通道。
7:4	CHANY_SEL1	rw	0x00	可配置为通道 0~~通道 15 中的任意通道。
3:0	CHANY_SEL0	rw	0x00	可配置为通道 0~~通道 15 中的任意通道。

注: 单周期扫描或连续扫描模式下, 硬件会启动 ADC\_CHANY0 影子寄存器, 在 ADC 未开始工作

时， 软件写 ADC\_CHANY0 的话，也会写到其影子寄存器；在 ADC 工作期间，若更改 ADC\_CHANY0 的值，只会更新其影子寄存器，且当 ADC 开始转换最后一个通道时，影子寄存器的值会更新至 ADC\_CHANY0， 这样即可完成动态切换通道。

### 27.8.10 A/D 任意通道通道选择寄存器 1 (ADC\_CHANY1)

偏移地址:0x60

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	CHANY_SEL15				CHANY_SEL14				CHANY_SEL13				CHANY_SEL12			
Type	rw															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	CHANY_SEL11				CHANY_SEL10				CHANY_SEL9				CHANY_SEL8			
Type	rw															

Bit	Field	Type	Reset	Description
31:28	CHANY_SEL15	rw	0x0	可配置为通道 0~通道 15 中的任意通道。
27:24	CHANY_SEL14	rw	0x0	可配置为通道 0~通道 15 中的任意通道。
23:20	CHANY_SEL13	rw	0x0	可配置为通道 0~通道 15 中的任意通道。
19:16	CHANY_SEL12	rw	0x0	可配置为通道 0~通道 15 中的任意通道。
15:12	CHANY_SEL11	rw	0x0	可配置为通道 0~通道 15 中的任意通道。
11:8	CHANY_SEL10	rw	0x0	可配置为通道 0~通道 15 中的任意通道。
7:4	CHANY_SEL9	rw	0x0	可配置为通道 0~通道 15 中的任意通道。
3:0	CHANY_SEL8	rw	0x0	可配置为通道 0~通道 15 中的任意通道。

注：单周期扫描或连续扫描模式下，硬件会启动 ADC\_CHANY1 影子寄存器，在 ADC 未开始工作时， 软件写 ADC\_CHANY1 的话，也会写到其影子寄存器；在 ADC 工作期间，若更改 ADC\_CHANY1 的值，只会更新其影子寄存器，且当 ADC 开始转换最后一个通道时，影子寄存器的值会更新至 ADC\_CHANY1， 这样即可完成动态切换通道。

## 27.8.11 A/D 任意通道配置寄存器 (ADC\_ANY\_CFG)

偏移地址:0x64

复位值:0x00000000

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

Bit	Field	Type	Reset	Description
31:4	Reserved		0x0	保留, 始终读为 0
3:0	CHANY_NUM	rw	0x00	通道数配置: 0: 0 通道 1: 0 ~ 1 通道 2: 0 ~ 2 通道 ... 14: 0 ~ 14 通道 15: 0 ~ 15 通道

注: 单周期扫描或连续扫描模式下, 硬件会启动 ADC\_NUM 影子寄存器, 在 ADC 未开始工作时, 软件写 ADC\_NUM 的话, 也会写到其影子寄存器; 在 ADC 工作期间, 若更改 ADC\_NUM 的值, 只会更新其影子寄存器, 且当 ADC 开始转换最后一个通道时, 影子寄存器的值会更新至 ADC\_NUM, 这样即可完成动态切换通道。

## 27.8.12 A/D 任意通道控制寄存器 (ADC\_ANY\_CR)

偏移地址:0x68

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Field	Res.																
Type																	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Field	JTRGSHIFT				Res.		JTRGSEL			JTRG EN	JAD ST	JAU TO	JEO SIE	JEO CIE	JEOS MPIE	JCE N	CHANY_ MDEN
Type	rw						rw			rw	rw	rw	rw	rw	rw	rw	

Bit	Field	Type	Reset	Description
31:18	Reserved		0x0	保留, 始终读为 0
17:16	JTRG_EDGE	rw	0x00	<p>注入通道触发边沿选择</p> <p>00:双沿触发 01:下沿触发 10:上沿触发 11:屏蔽触发</p>
15:13	JTRGSHIFT	rw	0x00	<p>注入通道外部触发延时采样 (Injected Channel External trigger shift sample)</p> <p>在触发信号产生后, 延时 N 个 PCLK2 的时钟周期再开始采样。</p> <p>如果是触发扫描模式, 其他通道是在上一个采样结束后立即开始。</p> <p>0:不延时 1:4 个周期 2:16 个周期 3:32 个周期 4:64 个周期 5:128 个周期 6:256 个周期 7:512 个周期</p>
12:11	Reserved			保留, 始终读为 0

Bit	Field	Type	Reset	Description
10:8	JTRGSEL	rw	0x00	<p>注入通道外部触发源选择 (Injected Channel External trigger selection)</p> <p>选择外部触发源 ADC1/ADC2</p> <p>00000:TIM1_TRGO 00001:TIM1_CC4 00010:TIM1_CC4 和 TIM1_CC5 0011:TIM2_CC1 0100:TIM3_CC4 0101:TIM8_CC4 0110:TIM8_CC4 和 TIM8_CC5 0111:EXTI 线 12</p> <p>其他:无效</p> <p>选择外部触发源 ADC3</p> <p>00000:TIM1_TRGO 00001:TIM1_CC4 00010:TIM1_CC4 和 TIM1_CC5 0011:TIM4_CC3 0100:TIM5_CC4 0101:TIM8_CC4 0110:TIM8_CC4 和 TIM8_CC5 0111:EXTI 线 12</p> <p>其他:无效</p>
7	JTRGEN	rw	0x00	<p>注入通道外部硬件触发源 (Injected Channel External trigger enable)</p> <p>1=使用外部触发信号启动 A/D 转换 0=不用外部触发信号启动 A/D 转换</p>

Bit	Field	Type	Reset	Description
6	JADST	rw	0x00	<p>注入通道 A/D 转换开始 (Injected Channel ADC start) :先开启 JCEN</p> <p>1=注入通道转换开始 0=注入通道转换结束或进入空闲状态</p> <p>JADST 清除有下列两种方式:</p> <p>在注入通道转换完成后, JADST 将被硬件自动清除。</p> <p>在清除 JCEN, JADST 将被硬件自动清除。</p>
5	JAUTO	rw	0x00	<p>自动注入转换 (Automatic Injected Group Convert)</p> <p>1=开启自动注入转换 0=关闭自动注入转换</p>
4	JEOSIE	rw	0x00	<p>A/D 注入通道连续转换完成中断使能 (ADC Injected Channel End of Sequential Convert Interrupt enable)</p> <p>1=使能 A/D 采样完成中断 0=禁用 A/D 采样完成中断</p> <p>如果 JEOCIF 置位, 通道组 A/D 转换结束后产生中断请求。</p>
3	JEOCIE	rw	0x00	<p>A/D 注入通道转换完成中断使能 (ADC Injected Channel end of convert interrupt enable)</p> <p>1=使能 A/D 采样完成中断 0=禁用 A/D 采样完成中断</p> <p>如果 JEOCIF 置位, A/D 转换结束后产生中断请求。</p>
2	JEOSMPIE	rw	0x00	<p>A/D 注入通道采样完成中断使能 (ADC Injected Channel end of sample interrupt enable)</p> <p>1=使能 A/D 采样完成中断 0=禁用 A/D 采样完成中断</p> <p>如果 JEOSMPIF 置位, A/D 转换结束后产生中断请求。</p>
1	JCEN	rw	0x00	<p>A/D 注入通道转换使能 (Injected Channel ADC enable)</p> <p>1= 使能 0= 禁用</p>

Bit	Field	Type	Reset	Description
0	CHANY_MDEN	rw	0x00	<p>任意通道配置模式使能位: 1:使能 0:禁止</p> <p>在使能后, 配置 ADC 通道功能发生变化。原先通道只由寄存器 ADC_ADCHS 的 CHENx 控制选择, 现在由两部分共同控制, CHANY_NUM 是配置通道 0~通道 15 中的通道数, 然后通道 0~通道 15 分别由 CHANY_SEL0 ~CHANY_SEL15 配置为任意 ADC 通道。</p>

注: 在任意通道模式, 且单周期 I 连续扫描模式下, 关闭 ADC 时, 必须先禁止 ADC\_ADCR 的 ADST 位, 然后判断 ADC\_ADSTA 的 BUSY 位是否为 0, 即等到 ADC 转换完成, 再禁止 ADC\_ANY\_CR 的 CHANY\_MDEN 位。

### 27.8.13 A/D 采样配置寄存器 (ADC\_SMPR1)

偏移地址:0x70

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	SAMCTL7				SAMCTL6				SAMCTL5				SAMCTL4			
Type	rw															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	SAMCTL3				SAMCTL2				SAMCTL1				SAMCTL0			
Type	rw															

Bit	Field	Type	Reset	Description
31 : 0	SAMCTLx	rw	0x00	<p>选择通道 x0~7 的采样时间 (Channel x Sample time selection)</p> <p>这些位用于独立地选择每个通道的采样时间。在采样周期中通道选择位必须保持不变。</p> <p>0000:2.5 周期 0100:42.5 周期 0001:8.5 周期 0101:56.5 周期 0010:14.5 周期 0110:72.5 周期 0011:29.5 周期 0111:240.5 周期 1000:3.5 周期 1001:4.5 周期 1010:5.5 周期 1011:6.5 周期 1100:7.5 周期 其他:保留 保留, 始终读为 0</p>

### 27.8.14 A/D 采样配置寄存器 (ADC\_SMPR2)

偏移地址:0x74

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	SAMCTL15				SAMCTL14				SAMCTL13				SAMCTL12			
Type	rw															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	SAMCTL11				SAMCTL10				SAMCTL9				SAMCTL8			
Type	rw															

Bit	Field	Type	Reset	Description
31 : 0	SAMCTLx	rw	0x00	<p>选择通道 8~15 的采样时间 (Channel x Sample time selection)</p> <p>这些位用于独立地选择每个通道的采样时间。在采样周期中通道选择位必须保持不变。</p>

					0000:2.5 周期
					0100:42.5 周期
					0001:8.5 周期
					0101:56.5 周期
					0010:14.5 周期
					0110:72.5 周期
					0011:29.5 周期
					0111:240.5 周期
					1000:3.5 周期
					1001:4.5 周期
					1010:5.5 周期
					1011:6.5 周期
					1100:7.5 周期
					其他:保留
					保留, 始终读为 0

### 27.8.15 A/D 注入通道数据补偿 (ADC\_JOFR0~3)

偏移地址:0x7C~0x88

复位值:0x00000000

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

Bit	Field	Type	Reset	Description
31:12	Reserved		0x0	保留, 始终读为 0
11 : 0	JOFR	rw	0x00	12 位 A/D 转换结果补偿寄存器

### 27.8.16 A/D 注入通道连续寄存器 (ADC\_JSQR)

偏移地址:0x8C

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res										JNUM		JSQ3			
Type											rw		rw			
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	JSQ3.	JSQ2				JSQ1				JSQ0						
Type	rw	rw				rw				rw						

Bit	Field	Type	Reset	Description
23:22	Reserved			保留, 始终读为 0
21:20	JNUM	rw	0x0	注入模式通道数配置 0: 0 通道 1: 0 ~ 1 通道 2: 0 ~ 2 通道 3: 0 ~ 3 通道
19:15	JSQ3	rw	0x0	00000~01111:可配置为通道 0~通道 15 中的任意通道为注入通道。 其他: 保留
14:10	JSQ2	rw	0x0	00000~01111:可配置为通道 0~通道 15 中的任意通道为注入通道。 其他: 保留
9:5	JSQ1	rw	0x0	00000~01111:可配置为通道 0~通道 15 中的任意通道为注入通道。 其他: 保留
4:0	JSQ0	rw	0x0	00000~01111:可配置为通道 0~通道 15 中的任意通道为注入通道。 其他: 保留

注: 单周期扫描或连续扫描模式下, 硬件会启动 ADC\_JSQR 影子寄存器, 在 ADC 未开始工作时, 软件写 ADC\_JSQR 的话, 也会写到其影子寄存器; 在 ADC 工作期间, 若更改 ADC\_JSQR 的值, 只会更新其影子寄存器, 且当 ADC 开始转换最后一个通道时, 影子寄存器的值会更新至 ADC\_JSQR, 这样即

可完成动态切换通道。

### 27.8.17 A/D 注入通道数据寄存器 (ADC\_JADDATA)

偏移地址:0x90

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.										JVALID	JOVER RUN	Res	JCHANNELSEL		
Type											r	r		r		
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	JDATA															
Type	r															

Bit	Field	Type	Reset	Description
31:23	Reserved		0x0	保留, 始终读为 0
22	JVALID	r	0x00	<p>注入通道有效标志位 (只读) (Injected Channel Valid flag) 1= JDATA[11:0] 位数据有效 0= JDATA[11:0] 位数据无效 相应模拟通道转换完成后, 将该位置位, 读 ADC_JADDATA 寄存器后, 该位由硬件清除。</p>
21	JOVERRUN	r	0x00	<p>注入通道数据覆盖标志位 (只读) (Injected Channel Overrun flag) 1= JDATA[11:0] 数据被覆盖 0= JDATA[11:0] 数据最近一次转换结果。 新的转换结果装载至寄存器之前, 若 JDATA[11:0] 的数据没有被读取, JOVERRUN 将置 1。读 ADC_JADDATA 寄存器后, 该位由硬件清除。</p>
20	Reserved		0x0	保留, 始终读为 0

Bit	Field	Type	Reset	Description
19:16	JCHANNELSEL	r	0x00	<p>该 4 位显示当前数据所对应的注入通道 ( Injected Channel Channel selection)</p> <p>0000 = 通道 0 的转换数据 0001 = 通道 1 的转换数据 0010 = 通道 2 的转换数据 0011 = 通道 3 的转换数据 0100 = 通道 4 的转换数据 0101 = 通道 5 的转换数据 0110 = 通道 6 的转换数据 0111 = 通道 7 的转换数据 1000 = 通道 8 的转换数据 1001 = 通道 9 的转换数据 1010 = 通道 10 的转换数据 1011 = 通道 11 的转换数据 1100 = 通道 12 的转换数据 1101 = 通道 13 的转换数据 1110 = 温度传感器的转换数据 1111 = 内部参考电压的转换数据 其他:无效</p>
15:0	JDATA	r	0x00	12 位 A/D 注入通道转换结果 (Injected Channel Transfer data) 根据设置左对齐或者右对齐.

### 27.8.18 A/D 注入通道数据寄存器 (ADC\_JDR0~3)

偏移地址:0xB0 ~ 0xBC

复位值:0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.								JVALID	JOVER RUN	Res.					
Type									r	r						
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	JDATA															
Type	r															

Bit	Field	Type	Reset	Description
31:23	Reserved		0x0	保留, 始终读为 0
22	JVALID	r	0x00	<p>注入通道有效标志位 (只读) (Injected Channel Valid flag)</p> <p>1= JDATA[11:0]位数据有效</p> <p>0= JDATA[11:0]位数据无效</p> <p>相应模拟通道转换完成后, 将该位置位, 读 JDR 寄存器后, 该位由硬件清除。</p>
21	JOVERRUN	r	0x00	<p>注入通道数据覆盖标志位 (只读) (Injected Channel Overrun flag)</p> <p>1= JDATA [11:0]数据被覆盖</p> <p>0= JDATA [11:0]数据最近一次转换结果</p> <p>新的转换结果装载至寄存器之前, 若 JDATA[11:0]的数据没有被读取, JOVERRUN 将置'1', 读 JDR 寄存器后, 该位由硬件清除。</p>
20 : 16	Reserved			保留, 始终读为 0。
15 : 0	JDATA	r	0x00	<p>12 位 A/D 注入通道的转换结果 (Injected Channel Transfer data)</p> <p>根据设置左对齐或者右对齐。</p>

## 28 数字/模拟转换器 (DAC)

本章介绍数字/模拟转换器的单/双通道工作模式及其相关内容。

### 28.1 简要介绍

数字/模拟转换器 (DAC) 包含了两个独立的 12 位数字输入, 模拟电压输出的数字/模拟转换器。每个通道都有单独的转换器, 既可以工作在单通道模式, 也可以工作在双通道模式。在双通道模式下, 可以同步地更新 2 个通道的输出, 2 个通道的转换既能各自分别进行, 也可以同时进行。

数字/模拟转换数据有 8 位或者 12 位模式, 也能配置成与 DMA 控制器协同工作。DAC 工作在 12 位数据模式时, 数据可以配置成左对齐或者右对齐。

为了保证 DAC 能够正常工作, DAC 的输入时钟频率不能超过 1MHz。

### 28.2 主要特征

- 1) 具有 2 个 DAC 转换通道: 每个通道各自对应 1 个自己的转换器
- 2) 具有单通道和双通道模式

- 3) 双 DAC 通道同时或者分别转换
- 4) 每个通道都有 DMA 功能
- 5) 转换数据 8 位或者 12 位单调输出
- 6) 在 12 位模式下数据左对齐或者右对齐
- 7) 具有同步更新功能
- 8) 具有噪声波形生成功能
- 9) 具有三角波形生成功能
- 10) 既可软件触发转换也可以由外部信号触发转换

DAC 单个通道的功能框图如下：

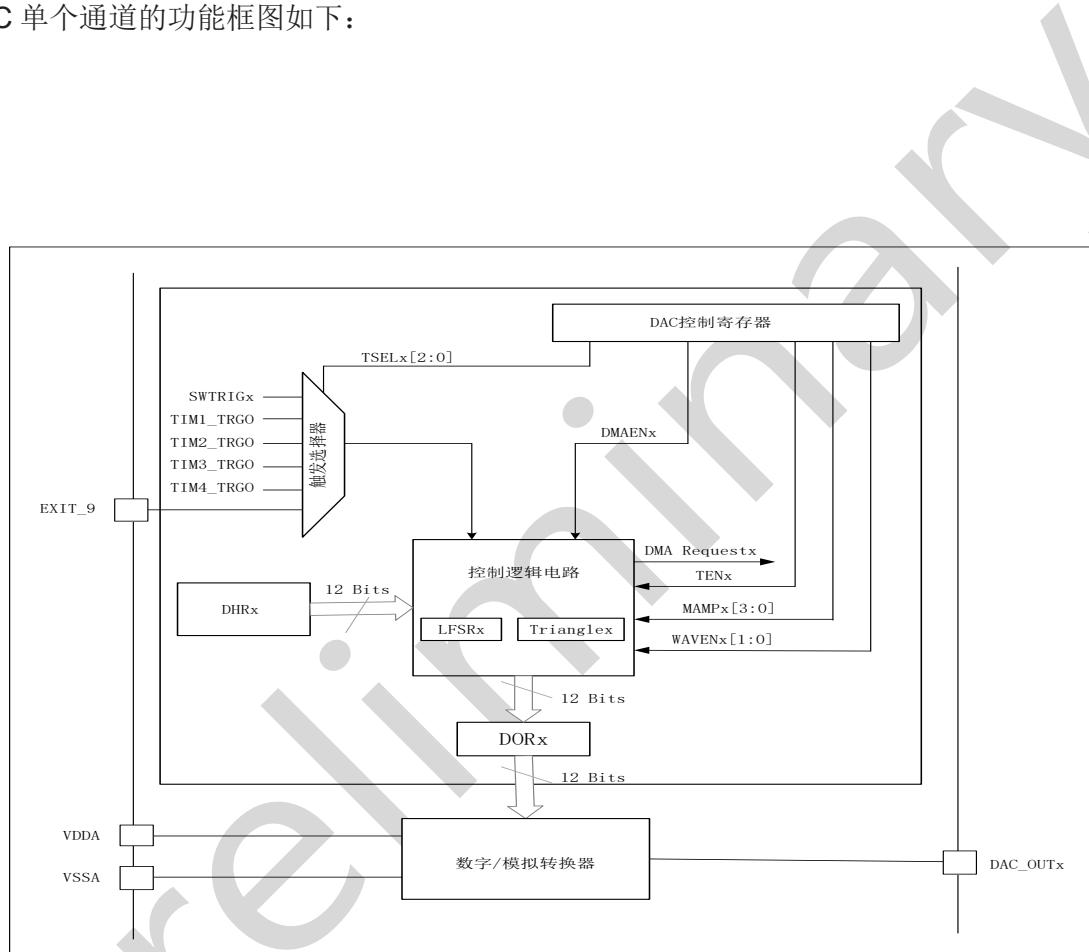


图 28-1 DAC 通道功能框图

相关管脚的描述如下。

表 28-1 DAC 管脚

名称	型号类型	注释
VDDA	输入, 模拟电源	模拟电源
VSSA	输入, 模拟电源地	模拟电源的地线
DAC_OUTx	模拟输出信号	DAC 通道x 的模拟输出

注：当 DAC 通道使能时，相应的 GPIO 管脚（PA4 或者 PA5）会自动连接到 DAC 的模拟输出

（DAC\_OUTx）。为了防止额外的功耗和避免寄生的干扰，应当提前设置管脚PA4 或者PA5 为模拟输

入 (AIN)。

## 28.3 功能描述

### 28.3.1 DAC 通道的使能

置位 DAC\_CR 控制寄存器的 ENx 位即可开始对 DAC 通道 x(x 代表 1 或 2, 下同)模拟部分电路的供电。经过一段启动时间  $t_{wakeup}$  后, DAC 通道 x 即被使能。

注: DAC 通道 x 的数字部分电路的正常工作不受该位控制 (ENx 位只是使能通道 x 的模拟部分电路)。

### 28.3.2 使能 DAC 输出缓存

DAC 集成了 2 个输出缓存, 可以用来减少输出阻抗, 无需外部运放即可直接驱动外部负载。每个 DAC 通道输出缓存可以通过设置 DAC\_CR 寄存器的相应位 BOFFx 来使能或者关闭。

### 28.3.3 DAC 输出电压

数字输入经过 DAC 转换成模拟电压输出, 其范围为 0 到  $V_{REF}$ , 满足下面的公式:

$$\text{DAC 输出} = V_{REF} \times (\text{DOR} / 4095)$$

### 28.3.4 DAC 触发源的选择

置位 DAC\_CR 寄存器的 TENx , 则 DAC 转换可以由某外部事件触发 ( 定时器计数器, 外部中断线)。共有 6 个可能的外部事件, 通过配置寄存器 DAC\_CR 的 TSELx[2: 0] 来选择其中之一触发转换。

表 28-2 外部触发

触发源	类型	TSEL[2:0]
定时器 1 TRGO 事件	来自片上定时器的内部信号	000
定时器 3 TRGO 事件		001
无效		010
无效		011
定时器 2 TRGO 事件		100
定时器 4 TRGO 事件		101
EXTI 线路 9	外部管脚	110
SWTRIG (软件触发)	软件控制位	111

如果选择硬件触发 (TSEL[2:0]==111) , 每次 DAC 接口检测到来自选中定时器 TRGO 输出, 或者外部中断线 9 的上升沿, 数据保持寄存器 DAC\_DHRx 中的数据会被传送到数据输出寄存器 DAC\_DORx 中。在 3 个 APB 时钟周期后, 寄存器 DAC\_DORx 更新为新值。

如果选择软件触发 (TSEL[2:0]==111) , 当 SWTRIG 位置' 1', 转换即开始。当寄存器 DAC\_DORx 从寄存器 DAC\_DHRx 取得数据后, SWTRIG 位由硬件自动清 0。

注:

- TSELx[2: 0] 位在 ENx 位被置位时不能改变。
- 如果选中软件触发，则数据从寄存器 DAC\_DHRx 装入寄存器 DAC\_DORx 只需要一个 APB 时钟周期。

### 28.3.5 DMA 请求

任一 DAC 通道都具有 DMA 功能，可分别用于各自通道的 DMA 请求。当寄存器 DAC\_DHRx 的数据被传到寄存器 DAC\_DORx 时，如果 DMA\_ENx 位置位，则产生一个 DMA 请求。

在双 DAC 通道模式下，可以置位 2 个通道的 DMA\_ENx 位，产生 2 个 DMA 请求。也可以选择置位 1 个 DMA\_ENx 位，只进行 1 个 DMA 传输。

注意：

DAC 的 DMA 请求不会累计，因此如果第 2 个外部触发发生在响应第 1 个外部触发之前，则第 2 个 DMA 请求无效，也不会报错。

### 28.3.6 DAC 数据格式

数据写入数据保持寄存器的格式根据单双通道模式各有不同，分别说明如下：

1) DAC 单通道x，分3种情况：

- 8 位数据右对齐：须将数据写入寄存器 DAC\_DHR8Rx 的 [7: 0] 位（实际是存入数据保持寄存器 DHRx[11: 4] 位）
- 12 位数据左对齐：须将数据写入寄存器 DAC\_DHR12Lx 的 [15: 4] 位（实际是存入数据保持寄存器 DHRx[11: 0] 位）
- 12 位数据右对齐：须将数据写入寄存器 DAC\_DHR12Rx 的 [11: 0] 位（实际是存入数据保持寄存器 DHRx[11: 0] 位）

经过相应的移位。DHRx 的内容由硬件自动完成，一是通过硬件，二是通过软件

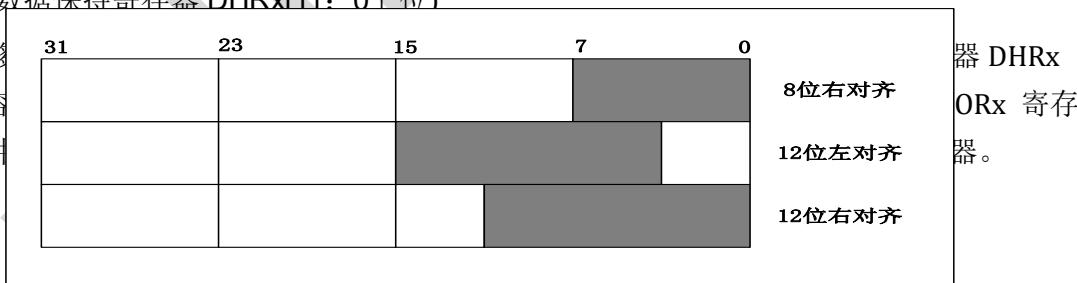


图 28-2 DAC 单通道模式的数据保持寄存器

2) DAC 双通道，分3种情况：

- 8 位数据右对齐：须将 DAC 通道 1 数据写入寄存器 DAC\_DHR8RD 的 [7: 0] 位（实际是存入数据保持寄存器 DHR1[11: 4] 位），将 DAC 通道 2 数据写入寄存器 DAC\_DHR8RD 的 [15: 8] 位（实际是存入数据保持寄存器 DHR2[11: 4] 位）
- 12 位数据左对齐：须将 DAC 通道 1 数据写入寄存器 DAC\_DHR12LD 的 [15: 4] 位（实际是存入数据保持寄存器 DHR1[11: 0] 位），将 DAC 通道 2 数据写入寄存器 DAC\_DHR12LD 的 [31: 20] 位（实际是存入数据保持寄存器 DHR2[11: 0] 位）

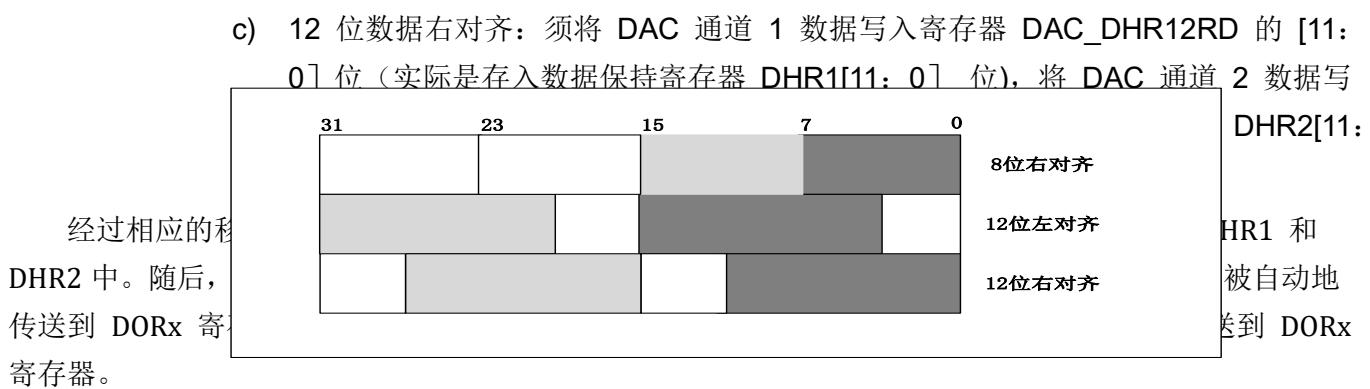


图 28-3 DAC 双通道模式的数据保持寄存器

### 28.3.7 带三角波生成的 DAC 转换

通过配置 DAC\_CR 寄存器的 WAVE<sub>x</sub>[1:0] 位为‘10’可以选中 DAC 的三角波生成功能，使得在 DC 或者缓慢变化的信号上叠加一个幅度的三角波。三角波的幅度可以通过设置 DAC\_CR 寄存器的 MAMP<sub>x</sub>[3:0] 位来选择。内部的三角波计数器在每次触发事件之后再经过 3 个 APB 时钟周期会累加 1。计数器的值与寄存器 DAC\_DHR<sub>x</sub> 的数值相加并丢弃溢出位后写入寄存器 DAC\_DOR<sub>x</sub>。当三角波计数器的值小于 MAMP[3:0] 位定义的最大幅度时，三角波计数器才会累加。当累加到设置的最大幅度时，计数器开始翻转基值 然后再开始累加，如此循环往复，周而复始。将 DAC\_CR 寄存器的 WAVE<sub>x</sub>[1:0] 位清零可以取消三角波的生成。

图 28-4 DAC 三角波生成

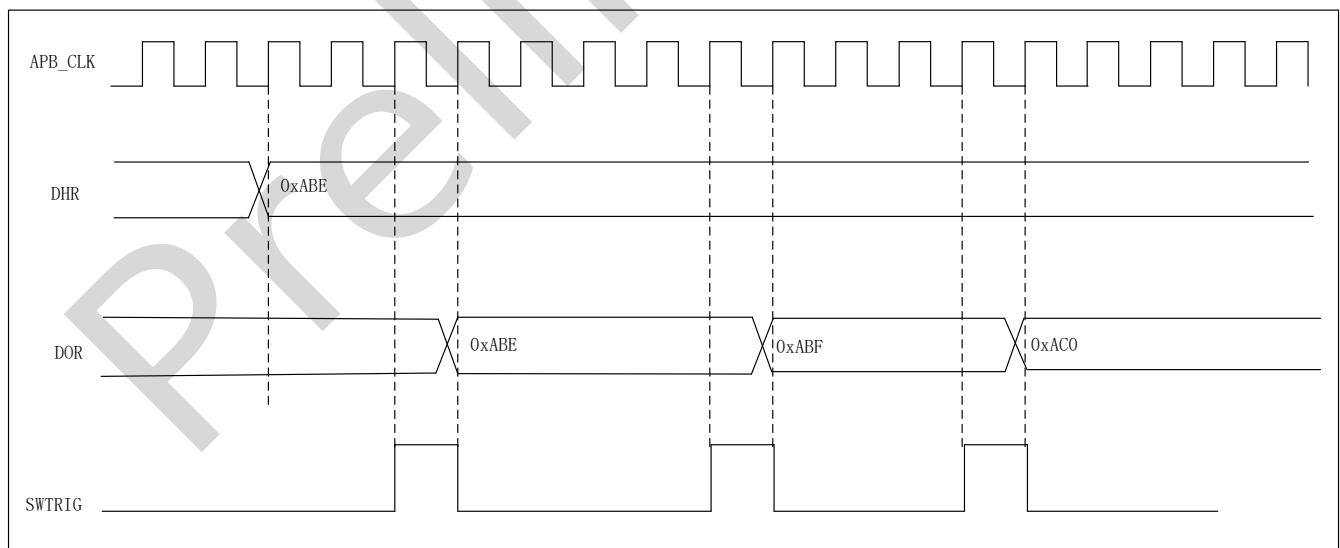


图 28-5 带三角波生成的 DAC 转换（使能软件触发）

注：

- 必须使能 DAC 触发才能产生三角波，即置位 DAC\_CR 寄存器的 TEN<sub>x</sub>。
- 必须在使能 DAC 之前设置 MAMP[3:0]，否则其值不能修改。

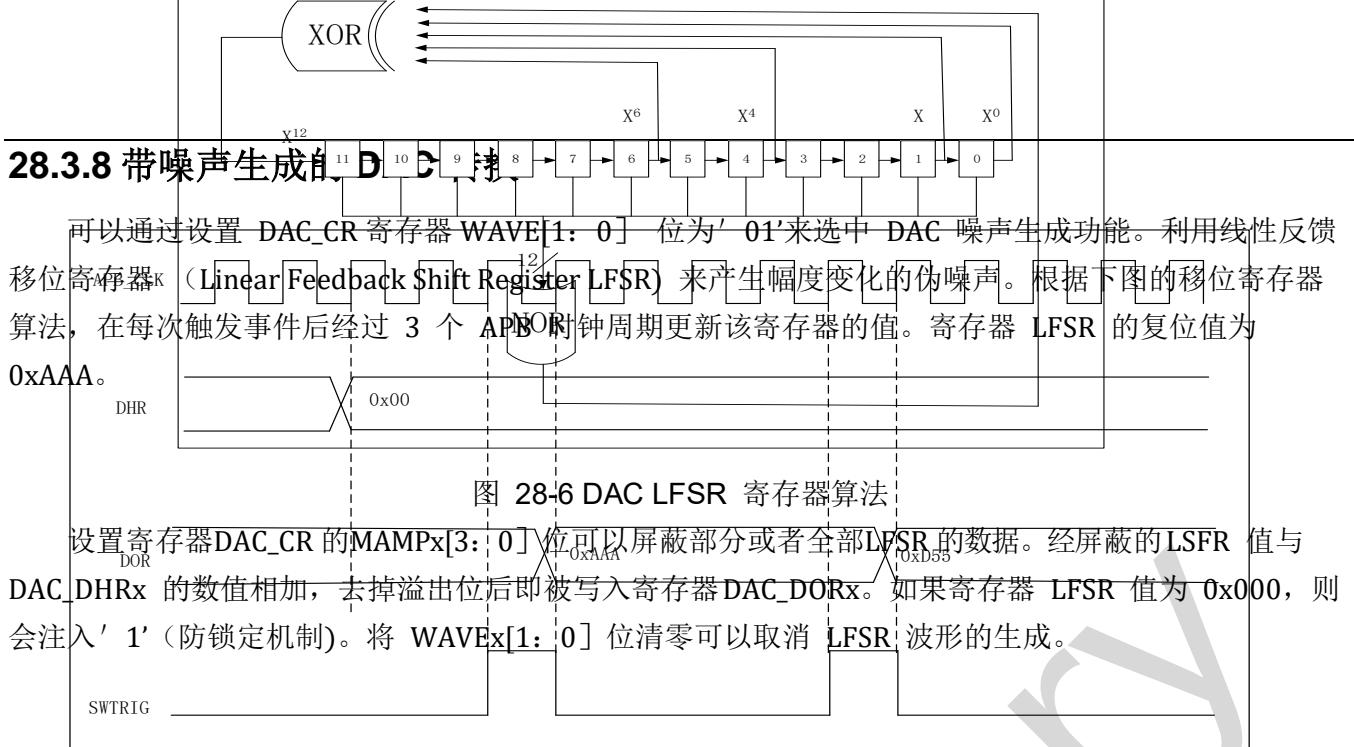


图 28-6 DAC LFSR 寄存器算法

注：必须使能 DAC 触发才能产生噪声，即置位  $DAC\_CR$  寄存器的  $TENx$ 。

### 28.3.9 DAC 转换

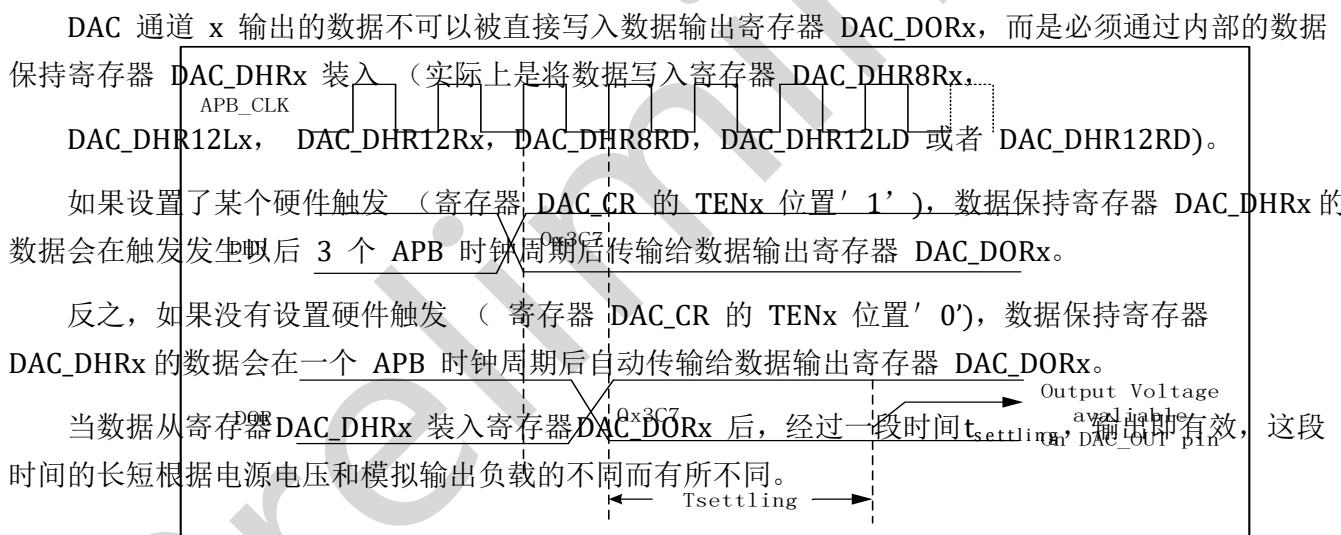


图 28-8  $TEN=0$  触发关闭时转换的时间示意图

### 28.3.10 双 DAC 通道转换

DAC 集成了 3 个供双 DAC 模式使用的寄存器：DHR8RD, DHR12RD 和 DHR12LD，能够更高效地利用总线带宽，在需要 2 个 DAC 同时工作的情况下，只需要访问一个寄存器即可完成同时驱动 2 个 DAC 通道的任务。

对于双DAC 通道转换和这些专用寄存器，共有 11 种不同配置组合的转换模式可供选择使用（每种模式配置完成后均需将双 DAC 通道转换数据装入所需的 DHR 寄存器（DHR12RD, DHR12LD 或者 DHR8RD）中）。

1) 统一触发的不带波形生成的转换。（ $TENx$  置位， $TSELx$  为相同值）

当触发事件发生，寄存器  $DHR1$  和  $DHR2$  的值在 3 个 APB 时钟周期后分别传入寄存器

---

## DAC DOR1 和 DAC DOR2。

- 2) 统一触发的带相同三角波生成的转换。 (TENx 置位, TSELx 为相同值, WAVE<sub>x</sub>='1x', MAMP<sub>x</sub> 为相同值)

当触发事件发生, 寄存器 DHR1 的值加上相同最大幅值的三角波计数器值, 其和在 3 个 APB 时钟周期后传入寄存器 DAC DOR1, 然后更新通道 1 三角波计数器。同时, 寄存器 DHR2 的值加上相同最大幅值的三角波计数器值, 其和在 3 个 APB 时钟周期后传入寄存器 DAC DOR2, 然后更新通道 2 三角波计数器。

- 3) 统一触发的带不同三角波生成的转换。 (TENx 置位, TSELx 为相同值, WAVE<sub>x</sub>='1x', MAMP<sub>x</sub> 为不同值)

当触发事件发生, 寄存器 DHR1 的值加上 MAMP1[3: 0] 所设最大幅值的三角波计数器值, 其和在 3 个 APB 时钟周期后传入寄存器 DAC DOR1, 然后更新通道 1 三角波计数器。同时, 寄存器 DHR2 的值加上 MAMP2[3: 0] 所设最大幅值的三角波计数器值, 其和在 3 个 APB 时钟周期后传入寄存器 DAC DOR2, 然后更新通道 2 三角波计数器。

- 4) 统一触发的带相同LFSR生成的转换。 (TENx置位, TSELx为相同值, WAVE<sub>x</sub>='01', MAMP<sub>x</sub>为相同值)

当触发事件发生, 寄存器DHR1的值加上带相同屏蔽的 LFSR1 计数器值, 其和在3 个 APB 时钟周期后传入寄存器 DAC DOR1, 然后更新 LFSR1 计数器。同时, 寄存器DHR2的值加上带相同屏蔽的 LFSR2 计数器值, 其和在3 个 APB 时钟周期后传入寄存器 DAC DOR2, 然后更新 LFSR2 计数器。

- 5) 统一触发的带不同 LFSR 生成的转换。 (TENx 置位, TSELx 为相同值, WAVE<sub>x</sub>='01', MAMP<sub>x</sub> 为不同值)

当触发事件发生, 寄存器 DHR1 的值加上带 MAMP1[3: 0] 所设屏蔽的 LFSR1 计数器值, 其和在 3 个 APB 时钟周期后传入寄存器 DAC DOR1, 然后更新 LFSR1 计数器。

同时, 寄存器 DHR2 的值加上带 MAMP1[3: 0] 所设屏蔽的 LFSR2 计数器值, 其和在 3 个 APB 时钟周期后传入寄存器 DAC DOR2, 然后更新 LFSR2 计数器。

- 6) 统一软件启动的转换。

该配置下, 一个 APB 时钟周期后, 寄存器 DHR1 和 DHR2 的值即被分别传入寄存器 DAC DOR1 和 DAC DOR2。

- 7) 分别触发的不带波形生成的转换。 (TENx 置位, TSELx 为不同值以选择不同触发源)。

当 DAC 通道 1 触发事件发生, 寄存器 DHR1 的值在 3 个 APB 时钟周期后传入寄存器 DAC DOR1。当 DAC 通道 2 触发事件发生, 寄存器 DHR2 的值在 3 个 APB 时钟周期后传入寄存器 DAC DOR2。

- 8) 分别触发的带相同三角波生成的转换。 (TENx置位, TSELx为不同值, WAVE<sub>x</sub>='1x', MAMP<sub>x</sub>为相同值)

当 DAC 通道 1 触发事件发生, 寄存器DHR1的值加上相同最大幅值的三角波计数器值, 其和在3 个 APB 时钟周期后传入寄存器 DAC DOR1 , 然后更新 DAC 通道 1 三角波计数器。当 DAC 通道 2 触发事件发生, 寄存器DHR2的值加上相同最大幅值的三角波计数器值,

其和在3个APB时钟周期后传入寄存器 DAC DOR2，然后更新 DAC 通道2三角波计数器。

- 9) 分别触发的带不同三角波生成的转换。（TENx置位，TSELx为不同值，WAVE<sub>x</sub>='1x'，MAMP<sub>x</sub>为不同值）

当 DAC 通道 1 触发事件发生，寄存器DHR1的值加上MAMP1[3: 0] 所设最大幅值的三角波计数器值，其和在3个APB时钟周期后传入寄存器 DAC DOR1，然后更新 DAC 通道 1 三角波计数器。当 DAC 通道 2 触发事件发生，寄存器DHR2的值加上MAMP2[3: 0] 所设最大幅值的三角波计数器值，其和在3个APB时钟周期后传入寄存器 DAC DOR2，然后更新 DAC 通道 2 三角波计数器。

- 10) 分别触发的带相同 LFSR 生成的转换。（TENx 置位，TSELx 为不同值，WAVE<sub>x</sub>='01'，MAMP<sub>x</sub> 为相同的 LFSR 屏蔽值。）。

当 DAC 通道 1 触发事件发生，寄存器 DHR1 的值加上带相同屏蔽的 LFSR1 计数器值，其和在 3 个 APB 时钟周期后传入寄存器 DAC DOR1，然后更新 LFSR1 计数器。当 DAC 通道 2 触发事件发生，寄存器 DHR2 的值加上带相同屏蔽的 LFSR2 计数器值，其和在 3 个 APB 时钟周期后传入寄存器 DAC DOR2，然后更新 LFSR2 计数器。

- 11) 分别触发的带不同LFSR 生成的转换。（TENx置位，TSELx为不同值，WAVE<sub>x</sub>='01'，MAMP<sub>x</sub>为不同值）

当 DAC 通道 1 触发事件发生，寄存器DHR1的值加上带 MAMP1[3: 0] 所设屏蔽的 LFSR1 计数器值，其和在3个APB时钟周期后传入寄存器 DAC DOR1，然后更新 LFSR1 计数器。当 DAC 通道 2 触发事件发生，寄存器DHR2的值加上带 MAMP2[3: 0] 所设屏蔽的 LFSR2 计数器值，其和在3个APB时钟周期后传入寄存器 DAC DOR2，然后更新 LFSR2 计数器。

这些转换模式在只使用一个DAC 通道的情况下仍然可用。

## 28.4 DAC 寄存器

表 28-3 DAC 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	DAC_CR	DAC 控制寄存器	0x00000000
0x04	DAC_SWTRIGR	DAC 软件触发寄存器	0x00000000
0x08	DAC_DHR12R1	DAC 通道 1 的 12 位右对齐数据保持寄存器	0x00000000
0x0C	DAC_DHR12L1	DAC 通道 1 的 12 位左对齐数据保持寄存器	0x00000000
0x10	DAC_DHR8R1	DAC 通道 1 的 8 位右对齐数据保持寄存器	0x00000000
0x14	DAC_DHR12R2	DAC 通道 2 的 12 位右对齐数据保持寄存器	0x00000000
0x18	DAC_DHR12L2	DAC 通道 2 的 12 位左对齐数据保持寄存器	0x00000000
0x1C	DAC_DHR8R2	DAC 通道 2 的 8 位右对齐数据保持寄存器	0x00000000

Offset	Acronym	Register Name	Reset
0x20	DAC_DHR12RD	双 DAC 的 12 位右对齐数据保持寄存器	0x00000000
0x24	DAC_DHR12LD	双 DAC 的 12 位左对齐数据保持寄存器	0x00000000
0x28	DAC_DHR8RD	双 DAC 的 8 位右对齐数据保持寄存器	0x00000000
0x2C	DAC_DOR1	DAC 通道 1 数据输出寄存器	0x00000000
0x30	DAC_DOR2	DAC 通道 2 数据输出寄存器	0x00000000

### 28.4.1 DAC 控制寄存器 (DAC\_CR)

偏移地址: 0x00

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved			DMA_EN2	MAMP2				WAVE2		TSEL2			TEN2	BOFF2	EN2
Type				rw	rw				rw		rw			rw	rw	rw
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved			DMA_EN1	MAMP1				WAVE1		TSEL1			TEN1	BOFF1	EN1
Type				rw	rw				rw		rw			rw	rw	rw

Bit	Field	Type	Reset	Description
31 : 29	Reserved			始终读为0。
28	DMA_EN2	rw	0x00	<p>DAC 通道2 DMA 使能 (DAC channel2 DMA enable) 该位由软件设置和清除。 0: 关闭 DAC 通道2 DMA 功能; 1: 使能 DAC 通道2 DMA 功能。</p>
27:24	MAMP2	rw	0x00	<p>DAC 通道 2 屏蔽 / 幅值选择器 (DAC channel2 mask / amplitude selector) 由软件设置该位, 用来在产生噪声时选择屏蔽位, 产生三角波时选择波形的幅值。 0000: 不屏蔽 LFSR 位 0 / 三角波幅值等于 1; 0001: 不屏蔽 LFSR 位 [1: 0] / 三角波幅值等于 3; 0010: 不屏蔽 LFSR 位 [2: 0] / 三角波幅值等于 7; 0011: 不屏蔽 LFSR 位 [3: 0] / 三角波幅值等于 15; 0100: 不屏蔽 LFSR 位 [4: 0] / 三角波幅值等于 31;</p>

Bit	Field	Type	Reset	Description
				<p>0101: 不屏蔽 LFSR 位 [5: 0] / 三角波幅值等于 63;</p> <p>0110: 不屏蔽 LFSR 位 [6: 0] / 三角波幅值等于 127;</p> <p>0111: 不屏蔽 LFSR 位 [7: 0] / 三角波幅值等于 255;</p> <p>1000: 不屏蔽 LFSR 位 [8: 0] / 三角波幅值等于 511;</p> <p>1001: 不屏蔽 LFSR 位 [9: 0] / 三角波幅值等于 1023;</p> <p>1010: 不屏蔽 LFSR 位 [10: 0] / 三角波幅值等于 2047;</p> <p>≥1011: 不屏蔽 LFSR 位 [11: 0] / 三角波幅值等于 4095。</p>
23:22	WAVE2	rw	0x00	<p>DAC 通道 2 噪声 / 三角波生成使能 (DAC channel2 noise / triangle wave generation enable)</p> <p>该位由软件设置和清除。</p> <p>00: 关闭波形生成功能;</p> <p>01: 生成噪声波形;</p> <p>1x: 生成三角波形。</p>
21:19	TSEL2	rw	0x00	<p>DAC 通道2 触发选择 (DACchannel2 triggerselection)</p> <p>该位用于选择 DAC 通道 2 的外部触发源。</p> <p>000: TIM1 TRGO 事件;</p> <p>001: TIM3 TRGO 事件;</p> <p>010: 无效;</p> <p>011: 无效;</p> <p>100: TIM2 TRGO 事件;</p> <p>101: TIM4 TRGO 事件;</p> <p>110: 外部中断线9;</p> <p>111: 软件触发。</p> <p>注意: 该位只能在TEN2=1 (DAC 通道2 触发使能) 时设置。</p>
18	TEN2	rw	0x00	<p>DAC 通道 2 触发使能 (DAC channel2 trigger enable)</p> <p>该位由软件设置和清除, 用来使能 / 关闭 DAC 通道 2 的触发。</p> <p>0: 关闭 DAC 通道 2 触发, 写入寄存器 DAC DHRx 的数据在 1 个 APB 时钟周期后传入寄存器 DAC DORx;</p> <p>1: 使能 DAC 通道 2 触发, 写入寄存器 DAC DHRx 的数据在 3 个 APB 时钟周期后传入寄存器 DAC DORx。</p> <p>注意: 如果选择软件触发, 写入寄存器 DAC DHRx 的数据只要 1 个 APB 时钟周期就可以传入寄存器 DAC DORx。</p>

Bit	Field	Type	Reset	Description
17	BOFF2	rw	0x00	<p>DAC 通道 2 输出缓存关闭 (DAC channel2 output buffer disable)</p> <p>该位由软件设置和清除，用来使能 / 关闭 DAC 通道 2 的输出缓存。</p> <p>0: 使能 DAC 通道 2 输出缓存； 1: 关闭 DAC 通道 2 输出缓存。</p>
16	EN2	rw	0x00	<p>DAC 通道 2 使能 (DAC channel2 enable)</p> <p>该位由软件设置和清除，用来使能 / 关闭 DAC 通道 2。</p> <p>0: 关闭 DAC 通道 2 ； 1: 使能 DAC 通道 2 。</p>
15 : 13	Reserved			始终读为 0。
12	DMA_EN1	rw	0x00	<p>DAC 通道 1 DMA 使能 (DAC channel1 DMA enable)</p> <p>该位由软件设置和清除。</p> <p>0: 关闭 DAC 通道 1 DMA 功能； 1: 使能 DAC 通道 1 DMA 功能。</p>
11: 8	MAMP1	rw	0x00	<p>DAC 通道 1 屏蔽 / 幅值选择器 (DAC channel1 mask / amplitude selector)</p> <p>由软件设置该位，用来在产生噪声时选择屏蔽位，产生三角波时选择波形的幅值。</p> <p>0000: 不屏蔽 LFSR 位 0 / 三角波幅值等于 1; 0001: 不屏蔽 LFSR 位 [1: 0] / 三角波幅值等于 3; 0010: 不屏蔽 LFSR 位 [2: 0] / 三角波幅值等于 7; 0011: 不屏蔽 LFSR 位 [3: 0] / 三角波幅值等于 15; 0100: 不屏蔽 LFSR 位 [4: 0] / 三角波幅值等于 31; 0101: 不屏蔽 LFSR 位 [5: 0] / 三角波幅值等于 63; 0110: 不屏蔽 LFSR 位 [6: 0] / 三角波幅值等于 127; 0111: 不屏蔽 LFSR 位 [7: 0] / 三角波幅值等于 255; 1000: 不屏蔽 LFSR 位 [8: 0] / 三角波幅值等于 511; 1001: 不屏蔽 LFSR 位 [9: 0] / 三角波幅值等于 1023; 1010: 不屏蔽 LFSR 位 [10: 0] / 三角波幅值等于 2047; ≥ 1011: 不屏蔽 LFSR 位 [11: 0] / 三角波幅值等于 4095;</p>

Bit	Field	Type	Reset	Description
7: 6	WAVE1	rw	0x00	<p>DAC 通道 1 噪声 / 三角波生成使能 (DAC channel1 noise / triangle wave generation enable)</p> <p>该位由软件设置和清除。</p> <p>00: 关闭波形生成功能;</p> <p>01: 产生噪声波形;</p> <p>1x: 产生三角波形。</p>
5: 3	TSEL1	rw	0x00	<p>DAC 通道 1 触发选择 (DAC channel1 trigger selection)</p> <p>该位用于选择 DAC 通道 1 的外部触发源。</p> <p>000: TIM1 TRGO 事件;</p> <p>001: TIM3 TRGO 事件;</p> <p>010: 无效;</p> <p>011: 无效;</p> <p>100: TIM2 TRGO 事件;</p> <p>101: TIM4 TRGO 事件;</p> <p>110: 外部中断线 9;</p> <p>111: 软件触发。</p> <p>注意: 该位只能在 TEN1 = 1 (DAC 通道 1 触发使能) 时设置。</p>
2	TEN1	rw	0x00	<p>DAC 通道 1 触发使能 (DAC channel1 trigger enable)</p> <p>该位由软件设置和清除, 用来使能 / 关闭 DAC 通道 1 的触 发。</p> <p>0: 关闭 DAC 通道 1 触发, 写入寄存器 DAC DHRx 的数据 在 1 个 APB 时钟周期后传入寄存器 DAC DORx;</p> <p>1: 使能 DAC 通道 1 触发, 写入寄存器 DAC DHRx 的数据 在 3 个 APB 时钟周期后传入寄存器 DAC DORx。</p> <p>注意: 如果选择软件触发, 写入寄存器 DAC DHRx 的数据只 要 1 个 APB 时钟周期就可以传入寄存器 DAC DORx。</p>
1	BOFF1	rw	0x00	<p>DAC 通道 1 输出缓存关闭 (DAC channel1 output buffer disable)</p> <p>该位由软件设置和清除, 用来使能 / 关闭 DAC 通道 1 的输出</p> <p>0: 使能 DAC 通道 1 输出缓存;</p> <p>1: 关闭 DAC 通道 1 输出缓存。</p>

Bit	Field	Type	Reset	Description
0	EN1	rw	0x00	<p>DAC 通道 1 使能 (DAC channel1 enable)</p> <p>该位由软件设置和清除, 用来使能 / 关闭 DAC 通道 1。</p> <p>0: 关闭 DAC 通道 1 ;</p> <p>1: 使能 DAC 通道 1 。</p>

## 28.4.2 DAC 软件触发寄存器 (DAC\_SWTRIGR)

偏移地址: 0x04

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved	DACPREG										Reserved				SW_TRIG2
Type		w													w	w

Bit	Field	Bit	Reset	Description
31:15	Reserved			始终读为 0。
14 : 8	DACPREG	w	0x00	<p>DAC 预分频 (DAC prescale)</p> <p>PCLK1 2<sup>n+1</sup> 分频后作为 DAC 时钟。</p>
7 : 2	Reserved			始终读为 0。
1	SW_TRIG2	w	0x00	<p>DAC 通道 2 软件触发 (DAC channel2 software trigger)</p> <p>该位由软件设置和清除, 用来使能 / 关闭软件触发。</p> <p>0: 关闭 DAC 通道 2 软件触发;</p> <p>1: 使能 DAC 通道 2 软件触发.</p> <p>注意: 当寄存器 DAC DHR2 的数据传入寄存器 DAC DOR2 时, 该位由硬件置' 0' (1 个 APB 时钟周期后)。</p>
0	SW_TRIG1	w	0x00	<p>DAC 通道 1 软件触发 (DAC channel1 software trigger)</p> <p>该位由软件设置和清除, 用来使能 / 关闭软件触发。</p> <p>0: 关闭 DAC 通道 1 软件触发;</p> <p>1: 使能 DAC 通道 1 软件触发.</p> <p>注意: 当寄存器 DAC DHR1 的数据传入寄存器 DAC DOR1 时, 该位由硬件置' 0' (1 个 APB 时钟周期后)。</p>

### 28.4.3 DAC 通道 1 的 12 位右对齐数据保持寄存器 (DAC\_DHR12R1)

偏移地址: 0x08

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved				DACC1DHR											
Type					rw											

Bit	Field	Type	Reset	Description
31:12	Reserved			始终读为 0。
11 : 0	DACC1DHR	rw	0x0000	DAC 通道 1 的 12 位右对齐数据 (DAC channel1 12-bit right-aligned data) 该位由软件写入，表示 DAC 通道 1 的 12 位数据。

#### 28.4.4 DAC 通道 1 的 12 位左对齐数据保持寄存器 (DAC\_DHR12L1)

偏移地址: 0x0C

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	DACC1DHR															
Type	rw															

Bit	Field	Type	Reset	Description
31 : 16	Reserved			始终读为0。
15 : 4	DACC1DHR	rw	0x0000	DAC 通道1的12位左对齐数据 (DACchannel1 12-bit left aligned data) 该位由软件写入，表示DAC 通道1 的12 位数据。
3: 0	Reserved			始终读为0。

## 28.4.5 DAC 通道 1 的 8 位右对齐数据保持寄存器 (DAC\_DHR8R1)

偏移地址: 0x10

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved								DACC1DHR							
Type									rw							

Bit	Field	Type	Reset	Description
31: 8	Reserved			始终读为 0。
7: 0	DACC1DHR	rw	0x00	DAC 通道 1 的 8 位右对齐数据 (DAC channel1 8-bit right-aligned data) 该位由软件写入, 表示 DAC 通道 1 的 8 位数据。

## 28.4.6 DAC 通道 2 的 12 位右对齐数据保持寄存器 (DAC\_DHR12R2)

偏移地址: 0x14

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved				DACC2DHR											
Type					rw											

Bit	Field	Type	Reset	Description
31:12	Reserved			始终读为 0。

Bit	Field	Type	Reset	Description
11 : 0	DACC2DHR	rw	0x0000	<p>DAC 通道 2 的 12 位右对齐数据 (DAC channel2 12-bit right-aligned data)</p> <p>该位由软件写入，表示 DAC 通道 2 的 12 位数据。</p>

#### 28.4.7 DAC 通道 2 的 12 位左对齐数据保持寄存器 (DAC\_DHR12L2)

偏移地址: 0x18

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	DACC2DHR															
Type	rw															

Bit	Field	Type	Reset	Description
31 : 16	Reserved			始终读为0。
15 : 4	DACC2DHR	rw	0x0000	<p>DAC 通道2的12位左对齐数据 (DAC channel2 12-bitleft-aligned data)</p> <p>该位由软件写入，表示DAC 通道2 的12 位数据。</p>
3: 0	Reserved			始终读为0。

#### 28.4.8 DAC 通道 2 的 8 位右对齐数据保持寄存器 (DAC\_DHR8R2)

偏移地址: 0x1C

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit	Reserved								DACC2DHR							
Type									rw							

Bit	Field	Type	Reset	Description
31 : 8	Reserved			始终读为 0。
7 : 0	DACC2DHR	rw	0x00	DAC 通道 2 的 8 位右对齐数据 (DAC channel2 8-bit right-aligned data) 该位由软件写入，表示 DAC 通道 2 的 8 位数据。

#### 28.4.9 双 DAC 的 12 位右对齐数据保持寄存器 (DAC\_DHR12RD)

偏移地址: 0x20

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved			DACC2DHR												
Type				rw												
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved			DACC1DHR												
Type				rw												

Bit	Field	Type	Reset	Description
31 : 28	Reserved			始终读为 0。
27 : 16	DACC2DHR	rw	0x0000	DAC 通道 2 的 12 位右对齐数据 (DAC channel2 12-bit right-aligned data) 该位由软件写入，表示 DAC 通道 2 的 12 位数据。
15 : 12	Reserved			始终读为 0。
11 : 0	DACC1DHR	rw	0x0000	DAC 通道 1 的 12 位右对齐数据 (DAC channel1 12-bit right-aligned data) 该位由软件写入，表示 DAC 通道 1 的 12 位数据。

#### 28.4.10 双 DAC 的 12 位左对齐数据保持寄存器 (DAC\_DHR12LD)

偏移地址: 0x24

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	DACC2DHR													Reserved		
Type	rw															
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	DACC1DHR													Reserved		
Type	rw															

Bit	Field	Type	Reset	Description
31 : 20	DACC2DHR	rw	0x0000	DAC 通道2的12位左对齐数据 (DAC channel2 12-bit left-aligned data) 该位由软件写入，表示DAC 通道2 的12 位数据。
19 : 16	Reserved			始终读为0。
15 : 4	DACC1DHR	rw	0x0000	DAC 通道1的12位左对齐数据 (DAC channel1 12-bit left-aligned data) 该位由软件写入，表示DAC 通道1 的12 位数据。
3: 0	Reserved			始终读为0。

#### 28.4.11 双 DAC 的 8 位右对齐数据保持寄存器 (DAC\_DHR8RD)

偏移地址: 0x28

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	DACC2DHR															
Type	rw															

Bit	Field	Type	Reset	Description
31 : 16	Reserved			始终读为0。
15 : 8	DACC2DHR	rw	0x00	DAC 通道2的8位右对齐数据 (DAC channel2 8-bit right-aligned data) 该位由软件写入，表示DAC 通道2 的8 位数据。

7: 0	DACC1DHR	rw	0x00	DAC 通道1 的8位右对齐数据 (DAC channel1 8-bit right-aligned data) 该位由软件写入，表示DAC 通道1 的8位数据。
------	----------	----	------	--

#### 28.4.12 DAC 通道 1 数据输出寄存器 (DAC\_DOR1)

偏移地址: 0x2C

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved				DACC1DOR											
Type					rw											

Bit	Field	Type	Reset	Description
31 : 12	Reserved			始终读为0。
11 : 0	DACC1DOR	rw	0x0000	DAC 通道1 输出数据 (DAC channel1 data output) 该位由软件写入，表示DAC 通道1 的输出数据。

#### 28.4.13 DAC 通道 2 数据输出寄存器 (DAC\_DOR2)

偏移地址: 0x30

复位值: 0x0000\_0000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Reserved															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Reserved				DACC2DOR											
Type					rw											

Bit	Field	Type	Reset	Description
31 : 12	Reserved			始终读为0。
11 : 0	DACC2DOR	rw	0x0000	DAC 通道2 输出数据 (DAC channel2 data output) 该位由软件写入，表示DAC 通道2 的输出数据。

## 29 比较器(COMP)

### 29.1 简介

芯片提供两组用于模拟输入电压比较的解决方案：COMP1 和 COMP2，并集成数字滤波器，可输出至定时器。

### 29.2 功能框图

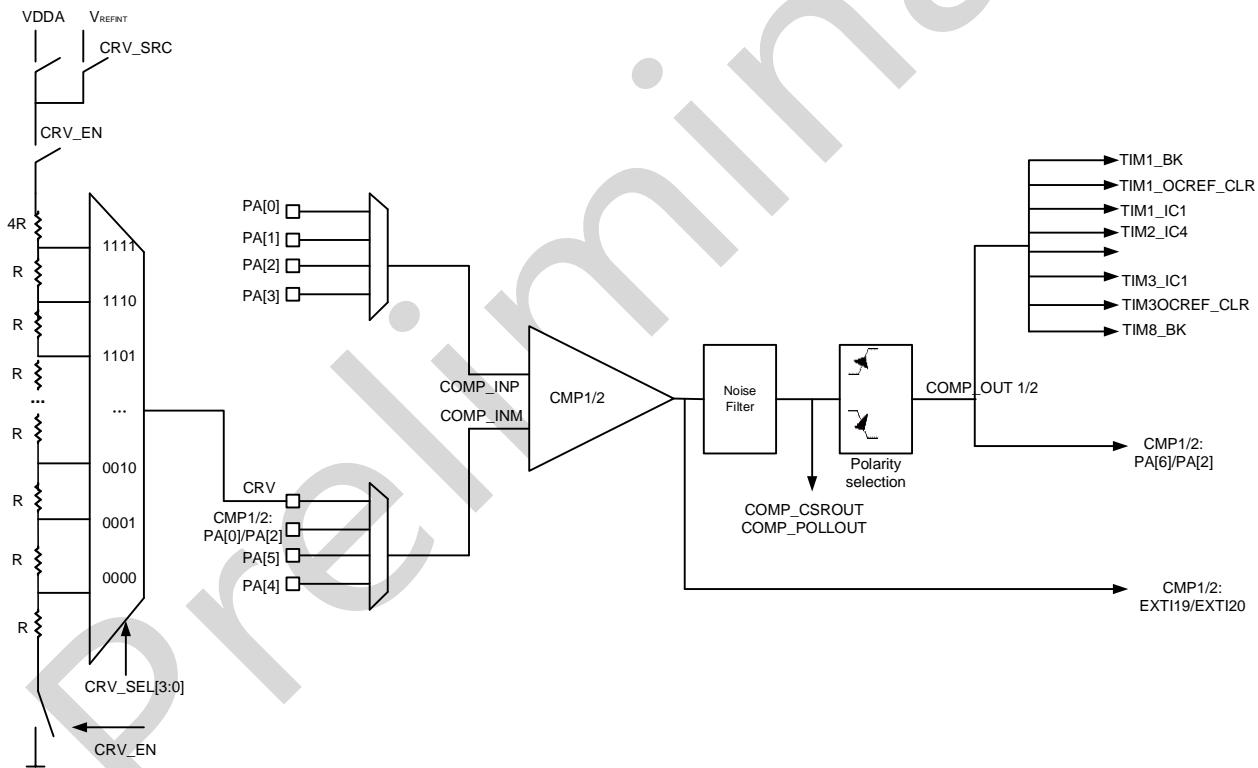


图 29-1 比较器框图

### 29.3 主要特征

- 1) 模拟输入为可复用的 I/O 引脚；
- 2) 可编程迟滞电压；
- 3) 支持多种速率和功耗；
- 4) 支持比较结果的滤波功能，滤波周期可配置；

- 5) 输出至 I/O 引脚或定时器;
- 6) 支持通过外部事件把 CPU 从睡眠和停机模式唤醒 ;
- 7) 支持 4 个正相输入和 4 个反相输入，带有轮询功能：
  - a) 可以实现定周期切换的轮询功能;
  - b) 可控制轮询通道 1/2/3 或 1/2;
  - c) 可选固定反向输入端。

## 29.4 功能描述

### 29.4.1 时钟复位

COMP 的输入时钟与 APB2\_CLK 同步。在使用比较器之前，要先使能 RCC 控制器中的时钟使能控制位。配置 RCC 控制器中的软件复位请求可进行软件复位操作。

### 29.4.2 比较器开关控制

在使用比较器之前，需要设置 COMPx\_CSR 寄存器的 EN 位给 COMP 上电。设置 EN 位时，它将 COMP 从断电状态唤醒，清除 EN 位可停止比较器工作。

### 29.4.3 比较器输入和输出

当 I/O 用作比较器输入时，必须在 GPIO 寄存器中将其设置为模拟模式。比较器的输出可选择滤波功能，可以输出到 I/O，也可以作为内部各种定时器的输入，详细参考 COMPx\_CSR 寄存器的 OUT\_SEL。

### 29.4.4 比较器通道选择

COMP 有四个正相输入和四个反相输入通道，正相输入可从四个外部引脚之间选取，反相输入可从三个外部引脚或者 CRV 电压分压值。CRV 的电压可选择 VDDA 或者内部基准 1.2V 电压( $V_{REFINT}$ )的分压。

COMP 的输入通道可以在普通工作模式下通过软件选择，也可以在轮询工作模式下通过硬件轮询的方式分时监测多个通道的比较结果，从逻辑上类似于多个比较器同时工作。

在普通工作模式下，比较器比较所选择的 INP 和 INM 端口上的信号，具体流程如下：

- 1) 配置 COMPx\_CSR 寄存器的 INP\_SEL 位和 INM\_SEL 位，选择所要比较的信号;
- 2) 配置 COMPx\_CSR 寄存器的 EN 位，比较器开始上电工作;
- 3) 比较的结果存放于 COMPx\_CSR 寄存器的 OUT 位。

另外，当 COMP 的 INM\_SEL 选择 CRV 时，需要配置 COMP\_CRV 寄存器的 CRV\_SEL 位，然后将 CRV\_EN 置位。

在轮询工作模式下，COMP 的 INP 端口上的信号将会周期性的轮询变化，而 INM 端口的信号可以配置 COMPx\_POLL 寄存器的 F1XN 位选择跟随 INP 端口变化或者由 COMPx\_CSR 的 INM\_SEL 位来

配置。需要注意的是，当启动轮询功能以后，COMPx\_CSR 的 INP\_SEL 位将失去作用，同样的，如果 COMPx\_POLL 寄存器的 F1XN 位选择 INM 端口跟随 INP 轮询变化，COMPx\_CSR 的 INN\_SEL 位也将失去作用。具体流程如下：

- 1) 配置 COMPx\_POLL 寄存器的 PER1OD 位来选择所需要的轮询等待周期；
- 2) 配置 COMPx\_POLL 寄存器的 F1XN 位来决定 INM 端口的信号是否跟随 INP 端口轮询变化；
- 3) 配置 COMPx\_POLL 寄存器的 POLL\_CH 位决定所需要轮询的通道是 1/2/3 或者 1/2；
- 4) 配置 COMPx\_POLL 寄存器的 POLL\_EN 位，启动轮询功能；
- 5) 配置 COMPx\_CSR 寄存器的 EN 位，比较器开始上电工作；
- 6) 轮询比较的结果存放于 COMPx\_POLL 寄存器的 POUT 位，其中 POUT[2]、POUT[1]、POUT[0] 位分别存放轮询通道 3/2/1 的比较结果。

#### 29.4.5 中断和唤醒

比较器的输出可以内部连接到事件控制器。每个比较器有自己的 EXTI 信号，能产生事件来退出低功耗模式。详细内容可以参考手册的中断和事件部分。

#### 29.4.6 功耗模式

在具体应用中可以通过调整比较器功耗和响应时间得到最优的结果。

COMPx\_CSR 寄存器的 MODE 位有下面几种设置：

- 1) 00: 高速/高功耗；
- 2) 01: 中速/中等功耗；
- 3) 10: 低速/低功耗；
- 4) 11: 极低速/极低功耗。

#### 29.4.7 比较器锁定机制

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

为了这个目的，比较器控制和状态寄存器可以设为写保护（只读）。

一旦设置完成，LOCK 位被设置为 1，这导致整个 COMPx\_CSR 寄存器变成只读，包括 LOCK 位在内。写保护只能被 MCU 复位所清除。

#### 29.4.8 迟滞电压

为了避免噪声信号导致的无效输入，比较器支持可配置的迟滞电压。

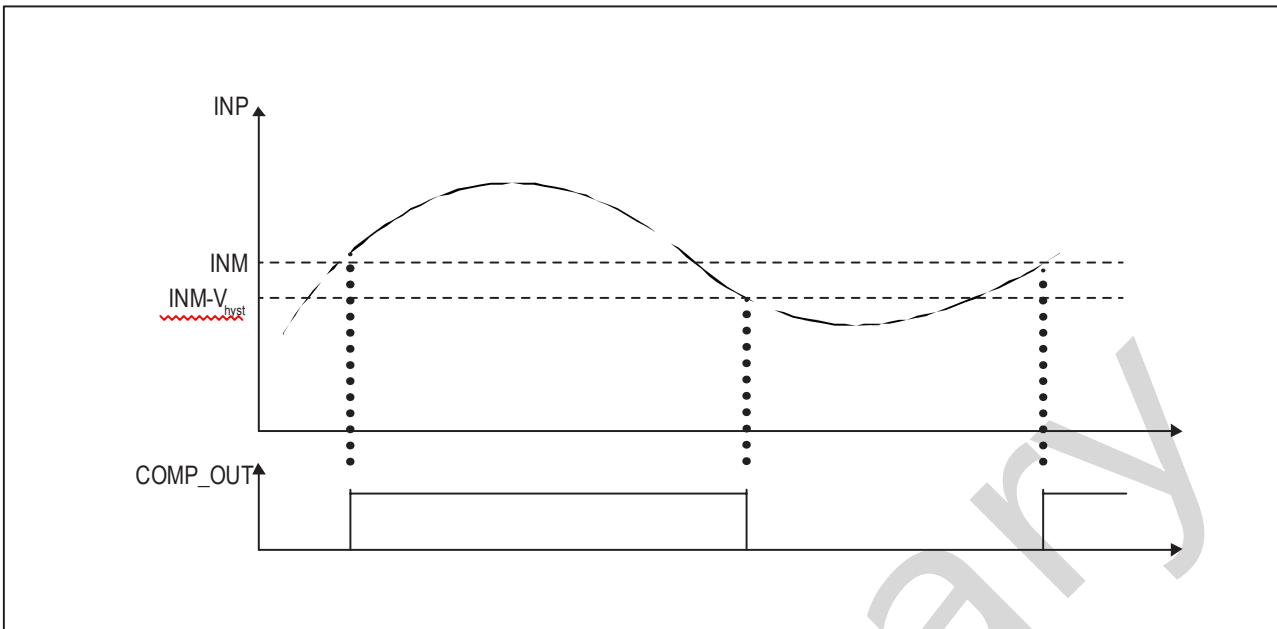


图 29-2 比较器的迟滞

## 29.5 比较器寄存器描述

表 29-1 COMP 寄存器概览

Offset	Acronym	Register Name	Reset
0x00, 0x04	COMPx_CSR (x=1, 2)	比较器 x (x=1, 2) 控制和状态寄存器	0x00000000
0x18	COMP_CRV	比较器外部参考电压寄存器	0x00000000
0x1C, 0x20	COMPx_POLL (x=1, 2)	比较器 x (x=1, 2) 轮询寄存器	0x00000000

### 29.5.1 比较器控制状态寄存器(COMPx\_CSR)(x=1,2)

偏移地址: 0x00, 0x04

复位值: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	LOCK	OUT	Res.										OFLT		HYST	
Type	rw	rw											rw		rw	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	POL	Res.	OUT_SEL				Res.	INP_SEL		Res.	INM_SEL		MODE		Res.	EN
Type	rw	rw	rw				rw	rw		rw	rw		rw		rw	rw

Bit	Field	Type	Reset	Description
31	LOCK	rw	0x00	<p>比较器锁 (Comparator lock)</p> <p>该位只能写一次，由软件置'1'，由系统复位清零。它令比较器 x 的所有控制位为只读。</p> <p>1: COMPx_CSR 只读 0: COMPx_CSR 可读可写</p>
30	OUT	r	0x00	<p>比较器 x 输出 (Comparator x lock)</p> <p>反映比较器 x 输出状态。</p> <p>1: 高输出 (正相输入高于反相输入) 0: 低输出 (正相输入低于反相输入)</p>
29 : 21	Reserved			始终读为 0。
20 : 18	OFLT	rw	0x00	<p>比较器 x 输出滤波 (Comparator output filter)</p> <p>这些位控制比较器 x 的输出滤波，连续的 PCLK2 时钟比较器输出不变则认为是有效的结果，否则保持不变。</p> <p>111: 128 个时钟周期 110: 64 个时钟周期 101: 32 个时钟周期 100: 16 个时钟周期 011: 8 个时钟周期 010: 4 个时钟周期 001: 2 个时钟周期 000: 无滤波</p>
17 : 16	HYST	rw	0x00	<p>比较器 x 迟滞电压 (Comparator x hysteresis)</p> <p>这些位控制比较器 x 的迟滞电压。</p> <p>11: 90mV 10: 30mV 01: 15mV 00: 0mV</p>
15	POL	rw	0x00	<p>比较器 x 输出极性 (Comparator x output polarity)</p> <p>该位用于切换比较器 x 输出极性。</p> <p>1: 反相输出 0: 同相输出</p>
14	Reserved			始终读为 0。

Bit	Field	Type	Reset	Description
13 : 10	OUT_SEL	rw	0x00	<p>比较器 x 输出选择 (Comparator x output selection)  这些位用来选择比较 x 输出方向。</p> <p>0010: 定时器 1 刹车输入  0100: 定时器 8 刹车输入  0110: 定时器 1 Ocrefclear 输入  0111: 定时器 1 输入捕捉 1  1000: 定时器 2 输入捕捉 4  1001: 定时器 2OCrefclear 输入  1010: 定时器 3 输入捕捉 1  1011: 定时器 3Ocrefclear 输入  其他: 无选择</p>
9	Reserved			始终读为 0。
8: 7	INP_SEL	rw	0x00	<p>比较器 x 正相输入选择 (Comparator x normal phase input selection)  这些位用于选择连接到比较器 x 的正相输入的信号源。</p> <p>比较器 1:</p> <p>00: COMP1_INP0 (PA0)  01: COMP1_INP1 (PA1)  10: COMP1_INP2 (PA2)  11: COMP1_INP3 (PA3)</p> <p>比较器 2:</p> <p>00: COMP2_INP0 (PA0)  01: COMP2_INP1 (PA1)  10: COMP2_INP2 (PA2)  11: COMP2_INP3 (PA3)</p>
6	Reserved			始终读为 0。
5: 4	INM_SEL	rw	0x00	<p>比较器 x 反相输入选择 (Comparator x inverting input selection)  这些位用于选择连接到比较器 x 的反相输入的信号源。</p> <p>比较器 1:</p> <p>00: COMP1_INM0 (PA4)  01: COMP1_INM1 (PA5)</p>

Bit	Field	Type	Reset	Description
				10: COMP1_INM2 (PA0) 11: COMP1_INM3 (CRV) 比较器 2: 00: COMP2_INM0 (PA4) 01: COMP2_INM1 (PA5) 10: COMP2_INM2 (PA2) 11: COMP2_INM3 (CRV)
3: 2	MODE	rw	0x00	比较器 x 模式 (Comparator x mode) 比较器 x 的工作模式控制位，允许调整速率和功耗。 11: 极低功率/极低速 10: 低功率/低速 01: 中等功率/中速 00: 高功率/高速
1	Reserved			始终读为 0。
0	EN	rw	0x00	比较器 x 使能 (Comparator x enable) 该位是比较器开关控制位。 1: 比较器 x 打开 0: 比较器 x 关闭

## 29.5.2 比较器外部参考电压寄存器(COMP\_CRV)

偏移地址: 0x18

复位值: 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	Res.										CRV_SRC	CRV_EN	CRV_SEL			
Type											rw	rw	rw			

Bit	Field	Type	Reset	Description
31:6	Reserved		0x0	保留, 始终读为 0

Bit	Field	Type	Reset	Description
5	CRV_SRC	rw	0x00	<p>比较器参考电压源选择 (Comparator external reference voltage source select)</p> <p>0: V<sub>REFINT</sub></p> <p>1: VDDA</p>
4	CRV_EN	rw	0x00	<p>比较器参考电压使能 (Comparator external reference voltage enable)</p> <p>1: 比较器参考电压使能</p> <p>0: 比较器参考电压禁止</p>
3: 0	CRV_SEL	rw	0x00	<p>比较器外部参考电压选择 (Comparator external reference voltage select)</p> <p>选择比较器外部参考电压</p> <p>0000: 1/20VDDA / V<sub>REFINT</sub></p> <p>0001: 2/20VDDA / V<sub>REFINT</sub></p> <p>0010: 3/20VDDA / V<sub>REFINT</sub></p> <p>0011: 4/20VDDA / V<sub>REFINT</sub></p> <p>0100: 5/20VDDA / V<sub>REFINT</sub></p> <p>0101: 6/20VDDA / V<sub>REFINT</sub></p> <p>0110: 7/20VDDA / V<sub>REFINT</sub></p> <p>0111: 8/20VDDA / V<sub>REFINT</sub></p> <p>1000: 9/20VDDA / V<sub>REFINT</sub></p> <p>1001: 10/20VDDA / V<sub>REFINT</sub></p> <p>1010: 11/20VDDA / V<sub>REFINT</sub></p> <p>1011: 12/20VDDA / V<sub>REFINT</sub></p> <p>1100: 13/20VDDA / V<sub>REFINT</sub></p> <p>1101: 14/20VDDA / V<sub>REFINT</sub></p> <p>1110: 15/20VDDA / V<sub>REFINT</sub></p> <p>1111: 16/20VDDA / V<sub>REFINT</sub></p>

### 29.5.3 比较器轮询寄存器(COMPx\_POLL)(x=1,2)

偏移地址: 0x1C, 0x20

复位值: 0x00000000

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

Field	Res.															
Type																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field				POUT			Res.	PERIOD			Res.	FIXN	POLL_CH	POLL_EN		
Type				r				rw				rw	rw	rw		

Bit	Field	Type	Reset	Description
31:11	Reserved		0x0	保留, 始终读为 0
10 : 8	POUT	r	0x00	轮询通道输出 (Polling output) 只读, 反映轮询通道输出状态, POUT[0]对应通道 1, POUT[1]对应通道 2, POUT[2]对应通道 3。 1: 高输出 (正相输入高于反相输入) 0: 低输出 (正相输入低于反相输入)
7	Reserved			始终读为 0。
6: 4	PERIOD	rw	0x00	轮询等待周期 (polling wait cycle) 每 n 个 PCLK2 周期切换到下一个轮询通道。 111: 128 个时钟周期 110: 64 个时钟周期 101: 32 个时钟周期 100: 16 个时钟周期 011: 8 个时钟周期 010: 4 个时钟周期 001: 2 个时钟周期 000: 1 个时钟周期
3	Reserved			始终读为 0。
2	FIXN	rw	0x00	轮询通道反相输入端固定 (Polling inverting input fix) 1: 轮询通道反相输入固定。由 CSR 寄存器 1NM_SEL 决定。 0: 轮询通道反相输入不固定。与 1NP 通道同时变化, 此时 1NM_SEL 无效。

Bit	Field	Type	Reset	Description
1	POLL_CH	rw	0x00	<p>比较器轮询通道 (Comparator polling channel)</p> <p>1: 轮询通道 1/2/3 0: 轮询通道 1/2</p> <p>注: 此时 1NP_SEL 无效。</p>
0	POLL_EN	rw	0x00	<p>比较器轮询模式使能 (Comparator polling enable)</p> <p>1: 比较器轮询模式使能 0: 比较器轮询模式禁止</p>

## 30 循环冗余校验计算单元(CRC)

### 30.1 CRC 简介

CRC 计算单元利用固定的多项式计算 32 位的 CRC 校验码。主要用于检测数据传输过程中的差错。除此之外，CRC 计算单元还能够计算出软件的标识，将该标识与连接时生成的参考标识比较，比较结果存放在指定的位置。

### 30.2 CRC 功能框图

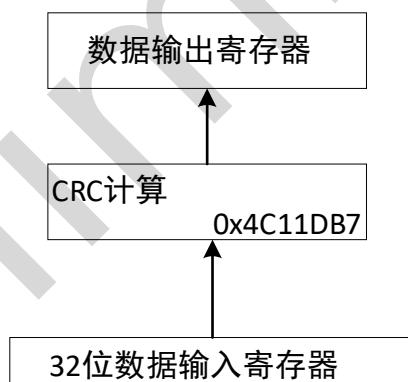


图 30-1 CRC 计算单元框图

### 30.3 主要特征

- 1) 统一计算多项式；
- 2)  $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$ , 该多项式同以太网计算多项式相同；
- 3) 32 位宽的数据输入输出寄存器，从输入数据到输出结果，CRC 计算周期为 3 个 HCLK；
- 4) 通用 32 位中间数据寄存器(可用于存放中间结果)。

## 30.4 功能描述

对数据寄存器进行写操作，进行 CRC 运算，前一次 CRC 计算结果和新计算结果的组合作为新的计算结果（每次按 32 位进行计算）。

执行 CRC 计算时，CPU 无法操作寄存器，故可以对寄存器 CRC\_DR 进行背靠背写入或者连续地写-读操作。

控制 CRC\_CTRL 寄存器的 RESET 位重置 CRC\_DR 寄存器的值为 0xFFFF FFFF。但不改变 CRC\_IDR 寄存器内的数据。

提供 8 位独立的数据寄存器（CRC\_IDR），与 CRC 计算无关，任何时候可以独立的读写操作，读 32 位数据寄存器返回上一次的计算结果。

可以通过设置寄存器 CRC\_CTRL 的 poly32\_mgn 位来设置算法为 crc32/MPEG-2 还是 crc32。

可以通过设置寄存器 CRC\_CTRL 的 crc\_bitsel 位来设置输入为 8 位/16 位/32 位。

可以通过设置寄存器 CRC\_CTRL 的 big\_ei 位来设置输入为大端还是小端。

可以通过设置寄存器 CRC\_CTRL 的 big\_eo 位来设置输出为大端还是小端。

可以通过软件，对寄存器 CRC\_MIR 来保留数据的中间结果。

当在第一串数据未计算完插入第二串数据的情况的步骤：

- 1) 将第一次计算的中间结果由软件从 CRC 中间数据寄存器(CRC\_MIR)中读出并进行存储；
- 2) 再计算第二串数据前，复位 CRC\_DR 寄存器(将 CRC 控制寄存器(CRC\_CTRL)的第 0 位置 1)；
- 3) 计算第二串数据，等第二串数据计算完成，读出第二串数据的结果(CRC\_DR 寄存器)；
- 4) 复位 CRC\_DR 寄存器(将 CRC 控制寄存器(CRC\_CTRL)的第 0 位置 1)；
- 5) 将第一步中存储的中间结果写入 CRC\_MIR 寄存器中，并继续进行计算第一串未完成的数据，待计算完成，从 CRC\_DR 寄存器读出第一串数据的结果。

## 30.5 寄存器描述

表 30-1 CRC 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	CRC_DR	CRC 数据寄存器	0xFFFFFFFF
0x04	CRC_IDR	CRC 独立数据寄存器	0x00000000
0x08	CRC_CTRL	CRC 控制寄存器	0x00000000
0x0C	CRC_MIR	CRC 中间数据寄存器	0xFFFFFFFF

### 30.5.1 CRC 数据寄存器(CRC\_DR)

偏移地址： 0x00

复位值: 0xFFFF FFFF

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

Bit	Field	Type	Reset	Description
31:0	DR	rw	0xFFFF_FFFF	DR: 数据寄存器位 (data register bits) 当数据写入 CRC 单元时, 用作输入寄存器, 读取该寄存器返回 CRC 计算结果

### 30.5.2 CRC 独立数据寄存器(CRC\_IDR)

偏移地址: 0x04

复位值: 0x0000 0000

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

Bit	Field	Type	Reset	Description
31:8	Reserve		0x0	保留, 始终读为 0
7:0	IDR	rw	0x00	IDR: 8 位通用数据寄存器位 (General-purpose 8-bit data register bits) 临时存放 1 字节的数据空间。 不受控制寄存器 CRC_CTRL 的 RESET 位控制

注: 此寄存器不参与 CRC 计算, 用于存放数据。

### 30.5.3 CRC 控制寄存器(CRC\_CTRL)

偏移地址: 0x08

复位值: 0x0000 0000

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

Field	Res.				BIG_EO	BIG_EI	CRC_BITSEL	POLY32_MGN	RES ET
Type					rw	rw	rw	rw	w

Bit	Field	Type	Reset	Description
31:6	Reserve		0x0	保留, 始终读为 0
5	BIG_EO	rw	0x0	BIG_EO: 输出大小端选择 1: 大端 0: 小端
4	BIG_EI	rw	0x0	BIG_EI: 输入大小端选择 1: 大端 0: 小端
3:2	CRC_BITSEL	rw	0x0	CRC_BITSEL: 输入位宽选择 10: 输入为 8 位 01: 输入为 16 位 00: 输入为 32 位
1	POLY32_MGN	rw	0x0	POLY32_MGN: 算法选择 1: 算法为 crc32 0: 算法为 crc32/MPRG-2
0	RESET	w	0x0	RESET: 复位 CRC 计算单元 (CRC reset) 配置数据寄存器 (CRC_DR) 为 0xFFFF FFFF。 该位只能写'1', 同时硬件自动实现清'0'。

### 30.5.4 CRC 中间数据寄存器(CRC\_MIR)

偏移地址: 0x0C

复位值: 0xFFFF FFFF

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

Bit	Field	Type	Reset	Description
31:0	CRC_MIR	rw	0xFFFF_FFFF	CRC_MIR[31:0]:中间数据寄存器 middle data register 用于读取计算的中间结果, 读写都是 32 位操作

注: 增加此寄存器的目的是为了实现当第一组数据还没计算结束时, 想要插入第二组数据进行计算之前, 软件可以将第一组数据当前计算的中间结果 CRC\_MIR 进行存储, 当第二组数据计算结束时可以取回存储的数据并且继续进行第一组还没结束的计算。

## 31 调试支持 (DEBUG)

### 31.1 DEBUG 简介

芯片内核包含调试模块，主要用于功能的调试。当内核在取指（指令断点）或访问数据（数据断点）时停止，硬件调试模块允许内核停止，此时，用户可以查询内核的内部状态和系统的外部状态。查询完成后，复原内核与外设，继续执行程序。

当连接芯片与调试器开始调试时，调试器自动调用内核的调试模块进行调试操作。

### 31.2 功能框图

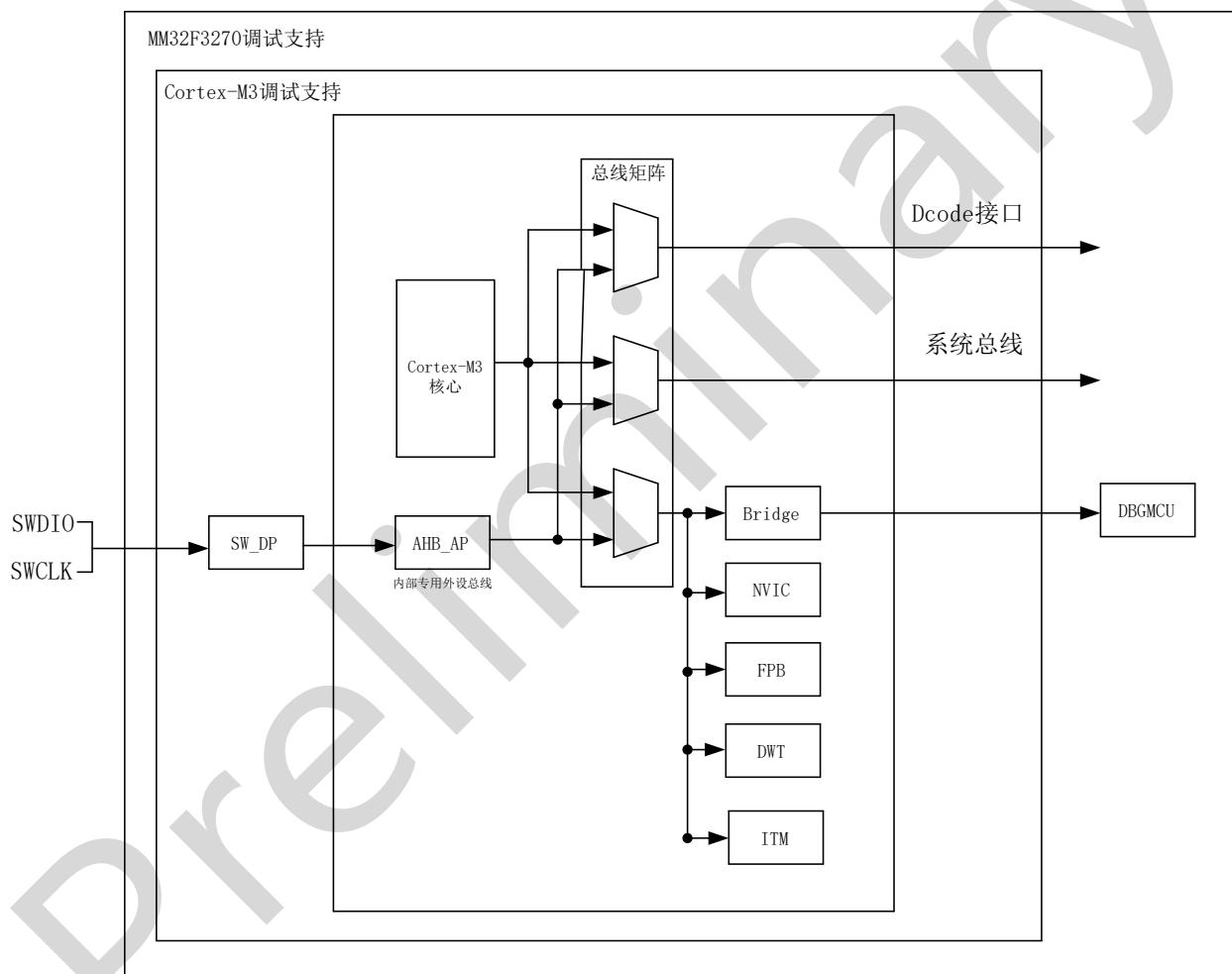


图 31-1 MM32 系列级别和 CPU 级别的调试框

CPU 内核含有调试单元，该单元由以下部分组成：

SW\_DP:串行数据线

BPU:断电调试单元

DWT:系统调试及跟踪

### 31.2.1 SWD 内部上拉与下拉

SWD 输入直接连接到 D 触发器控制着调试模式，不能悬空，同时对于 SWCLK 引脚，直接连接到一些 D 触发器的时钟端。

为了保证 I/O 电平可控，SWD 输入脚上内嵌入了上拉和下拉电平选择。

1) SWDIO：内部上拉

SWCLK：带下拉的输入

软件可以把这些 I/O 口作为普通的 I/O 口使用。

### 31.2.2 SWD 调试端口

该芯片的 2 个普通 I/O 口可用作 SW-DP 接口引脚。这些引脚在所有的封装里都存在。

表 31-1 SWJ 调试端口管脚

SWJ-DP 端口引脚名称	SW 调试接口		引脚分配
	类型	调试功能	
SWDIO 调试接口	输入/输出	串行数据输入/输出	PA13
SWCLK	输入	串行时钟	PA14

## 31.3 ID 代码和锁定机制

在芯片内部有多个 ID 编码。

### 31.3.1 微控制器设备 ID 编码

微控制器内部包含 MCU ID 编码。这个 ID 定义了 MCU 的硅片版本，并且映射到外部 APB 总线上。作为 DEBUG\_MCU 的组成部分，通过用户代码与 SW 调试接口均能够获取此编码。

DEBUGMCU\_IDCODE

地址：0x40007080 只支持 32 位访问，只读

表 31-2 设备 ID

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

Bit	Field	Type	Reset	Description
31:8	DEV_ID	r		设备识别编码 (Device identifier)

### 31.3.2 Cortex JEDEC-106 ID 代码

CPU 有一个 JEDEC-106ID 编码。它位于映射到内部 APB 总线地址为 0xE00FF000\_0xE00FFFFF 的 4KB ROM 表中。

下表是该系列的各个 ID 编码：

表 31-3 ID 编码

ID 名	芯片
DEV_ID	0xCC9AA0E7
CPU TAP SW ID	0xBB11477

## 31.4 功能描述

### 31.4.1 SW 协议介绍

此同步串行协议使用 2 个引脚：

主机到目标的时钟信号 (SWCLK) 与双向数据信号 (SWDIO)

SWDIO 作为双向数据线，需连接上拉电阻 (ARM 建议值 100K)

数据从低位开始传输，允许读写寄存器组 DPACC 与 APACC

根据协议，当 SWDIO 改变方向时，同时需要插入一个转换时间（默认一个 bit 时间，具体可以通过 SWCLK 调整），这段间内，任何设备不能驱动此信号线。

### 31.4.2 SW 协议序列

一次序列包含三个阶段：

- 1) 主机发送 8 位包请求；
- 2) 目标发送 3 位确认应答；
- 3) 根据配置方向，主机或目标发送 32 位数据；

表 31-4 8bit 请求位包

比特位	名称	描述
0	起始	必须为 1
1	APnDP	0: 访问 DP 1: 访问 AP
2	RnW	0: 写请求 1: 读请求
4: 3	A[3 : 2]	DP 或 AP 寄存器的地址

比特位	名称	描述
5	Parity	前面比特位的校验位
6	Stop	0
7	Park	不能由主机驱动，由于有上拉，目标永远读为 1

更多关于 DAPCC 与 APACC 寄存器的信息，查看 ARM 相关的 CPU 技术参考手册。

每一个包请求紧跟一个 bit 转换时间（默认为 1 位）

表 31-5 3bit 应答包

比特位	名称	描述
0 .. 2	ACK	001: 失败 010: 等待 100: 成功

注：当应答信号（ACK）位上表的情况之一时，应答位后有一个转换时间。

表 31-6 33bit 请求位包

比特位	名称	描述
0 .. 31	WDATA/RDATA	写或读的数据
32	Parity	32 位数据的奇偶校验位

注：读数据位结束后等待一个转换时间。

### 31.4.3 SW-DP 状态机 (Reset,idle states, ID code)

SW-DP 状态机通过内部的 ID 码识别 SW\_DP，遵守 JEP-106 标准，具体信息请参考 ARM 相关手册（Cortex M3）

SW-DP 的状态机不会工作（当调试读取 ID 时候）。

当出现了上电复位，或 DP 从 JTAG 切换到 SWD 后，或超过 50 个周期的高电平时，SW-DP 状态机将处于复位状态；

状态机会从 RESET 状态切换到 IDLE 状态（出现了至少 2 个周期的低电平）；

状态机开始处于复位态，必须先切换到 IDLE 态，执行读 DP-SW ID 寄存器的操作。否则，调试器无法进行其它正常的传输，会出现 ACK 失败；

### 31.4.4 DP 和 AP 读 / 写访问

对 DP 的读操作没有传递性：调试器将直接获得数据（如果 ACK=成功），或者等待（如果 ACK=等待）；

对 AP 的读操作具有传递性。这意味着前一次读操作的结果只能在下一次操作时获得。如果下一次的操作不是对 AP 的访问，则必须读 DP-RDBUFF 寄存器来获得上一次读操作的结果；

DP-CTRL/STAT 寄存器的 READ OK 标志位会在每次 AP 读操作和 RDBUFF 读操作后更新，以通知调试器 AP 的读操作是否成功；

SW-DP 具有写缓冲区（DP 和 AP 都有写缓冲），这使得其他传输在进行时，仍然可以接受写操作。如果写缓冲区满，调试器将获得一个等待的 ACK 响应。读 IDCODE 寄存器，读 CTRL/STAT 寄存器和写 ABORT 寄存器操作在写缓冲区满时仍被接受；

由于 SWCLK 和 HCLK 的异步性，需要在写操作后（在奇偶校验位后）插入 2 个额外的 SWCLK 周期，以确保内部写操作正确完成。这两个额外的时钟周期需要在线路为低时插入（IDLE 状态下）。这个操作步骤在写 CTRL/STAT 寄存器以提出一个上电请求时尤其重要，否则下一个操作（在内核上电后才有效的操作）会立即执行，这将导致失败；

### 31.4.5 SW-DP 寄存器

当 APnDP=0 时，可以访问以下这些寄存器。

表 31-7 SW\_DP 寄存器

A[3:2]	读 / 写	SELECT 寄存器 的 CTRLSEL 位	寄存器	描述
00	读		IDCODE	固定为 0x1BA0 1477 (用于识别 SW-DP)。
00	写		ABORT	
01	读/写	0	DP-CTRL /STAT	请求一个系统或调试的上电操作；配置 AP 访问的操作式； 控制比较，校验操作； 读取一些状态位 (溢出，上电响应)。
01	读/写	1	WIRE CONTROL	配置串行通信物理层协议 (如转换时间长度等)。
10	读		READ RESEND	允许从一个错误的调试传输中恢复数据而不用重复最初的 AP 传输。
10	写		SELECT	选择当前的访问端口和有效的 4 字长寄存器窗口。
11	读/写		READ BUFFER	由于 AP 的访问具有传递性 (当前 AP 读操作的结果会在下次 AP 传输时传出)，因此这个寄存器非常必要。这个寄存器会从 AP 捕获上一次读操作的数据结果，因此可以获得数据而不必再启动一个新的 AP 传输。

### 31.4.6 SW-AP 寄存器

当 APnDP=1 时，可以访问

AP 寄存器的访问地址由以下两部分组成:

- 1) A[3: 2]的值  
DP SELECT 寄存器的当前值

## 31.5 MCU 调试模块 (MCUDBG)

MCU 调试模块协助调试器提供以下功能:

- 1) 断点时定时器与看门狗的时钟控制
- 2) 分配控制跟踪脚
- 3) 控制低功耗模式

### 31.5.1 低功耗模式的调试支持

MCU 具有多种低功耗模式，能够关闭 CPU 时钟，降低 CPU 的功耗，通过执行 WFE 或 WFI 指令进入低功耗模式。FCLK 与 HCLK 对于调试操作时必须的，不能关闭，同时 MCU 通过配置一些寄存器来改变低功耗模式特性，从而支持在低功耗模式下调试代码，具体的配置如下。

- 1) 当进入睡眠模式时，为了能够将 HCLK 提供同 FCLK（系统配置）相同时钟，调试器必须先置位 DEBUGMCU\_CR 寄存器的 DBG\_SLEEP 位。
- 2) 当进入停机模式时，必须配置 DBG\_STOP 位，该操作会激活调试器内部振荡器，从而为 FCLK 与 HCLK 提供时钟。

### 31.5.2 支持定时器、看门狗

当产生断点时，根据定时器和看门狗的应用不同来选择计数器的工作模式，有如下的 2 种：

- 1) 计数器继续计数。应用在输出 PWM 波控制电机。
- 2) 计数器停止计数。应用在看门狗计数器。

### 31.5.3 调试 MCU 配置寄存器

该寄存器支持调试模式下配置 MCU，主要包括:

- 1) 支持定时器和看门狗的计数器；支持低功耗模式；
- 2) 支持分配跟踪引脚 DEBUGMCU\_CR 寄存器映射到外部 APB 总线的基址为 0x40013404。此寄存器只能由 PORE\_SET 异步复位。内核在复位状态下时，调试器可写该寄存器。如果调试器不支持这些特性，也可通过软件写这些寄存器。

## 31.6 DEBUG 寄存器描述

表 31-8 DBG 寄存器概览

Offset	Acronym	Register Name	Reset
0x00	DEBUG	DEBUG 控制寄存器	0x00000000

### 31.6.1 DEBUG 控制寄存器 (DEBUG\_CR)

地址: 0x40007084 只支持 32 位访问

POR 复位: 0x0000 0000(不被系统复位所复位)

Bit	31	30	29	28	27	26	25	24
Field	Reserved							
Type								
Bit	23	22	21	20	19	18	17	16
Field	Reserved			debug_tim 7_stop	debug_tim 6_stop	debug_tim 5_stop	debug_tim 8_stop	Reserved
Type				rw	rw	rw	rw	
Bit	15	14	13	12	11	10	9	8
Field	Reserved	Reserved	debug_tim 4_stop	debug_tim 3_stop	debug_tim 2_stop	debug_tim 1_stop	debug_wwdg_stop	debug_iwdg_stop
Type			rw	rw	rw	rw	rw	
Bit	7:6		5	4	3	2	1	0
Field	Reserved		Reserved	Reserved	debug_stop_for_Id o	debug_sta dby	debug_sto p	debug_slee p
Type					rw	rw	rw	rw

Bit	Field	Type	Reset	Description
31:21	Reserved		0x0	保留, 始终读为 0
20:17	debug_timx_stop	rw	0x0	当内核进入调试状态时计数器停止工作 (x = 8,5,6,7,) (TIMx counter stopped when core is halted) 0: 选中定时器的计数器仍然正常工作 1: 选中定时器的计数器停止工作
16:15	Reserved		0x0	保留, 始终读为 0
14	Reserved		0x0	保留, 始终读为 0

Bit	Field	Type	Reset	Description
13:10	debug_timx_stop	rw	0x0	<p>当内核进入调试状态时计数器停止工作 x = 1,2,3,4 (TIMx counter stopped when core is halted)</p> <p>0: 选中定时器的计数器仍然正常工作 1: 选中定时器的计数器停止工作</p>
9	debug_wwdg_stop	rw	0x0	<p>当内核进入调试状态时调试窗口看门狗停止工作 ( Debug window watchdog stopped when core is halted)</p> <p>0: 窗口看门狗计数器仍然正常工作 1: 窗口看门狗计数器停止工作</p>
8	debug_iwdg_stop	rw	0x0	<p>当内核进入调试状态时看门狗停止工作 (Debug independent watchdog stopped when core is halted)</p> <p>0: 看门狗计数器仍然正常工作 1: 看门狗计数器停止工作</p>
7:6	Reserved		0x0	保留, 始终读为 0
5	Reserved		0x0	保留, 始终读为 0
4	Reserved		0x0	保留, 始终读为 0
3	debug_stop_for_ldo	rw	0x0	<p>调试停机模式 (Debug Stop mode)</p> <p>0: 进入 stop 模式时, 关掉 LDO 1: 不能进入 stop 模式, 不会关掉 LDO</p>

Bit	Field	Type	Reset	Description
2	debug_standby	rw	0x0	<p>调试待机模式 (Debug Standby mode)</p> <p>0: 数字电路部分完全断电, HCLK 与 FCLK 关闭, 退出 STANDBY 模式 (除开指示退出待机模式的标志位), 与复位是一样的</p> <p>1: 时钟 FCLK 与 HCLK 关闭, FCLK 与 HCLK 时钟由内部 RL 振荡器提供时钟。微控制器通过系统复位方式退出 STANDBY 模式和复位是一样的。</p>
1	debug_stop	rw	0x0	<p>调试停机模式 (Debug Stop mode)</p> <p>0: 在停机模式时, 时钟控制器禁止一切时钟 (包括 HCLK 和 FCLK)。如果 HSI 时钟 4 分频后作为 PLL 输入时钟, 当从 STOP 模式退出时, 时钟配置与复位之后的配置一致, 软件不需要重新配置时钟控制系统。</p> <p>当 HSE 时钟直接为 PLL 输入时钟时, 从 STOP 模式退出时, 软件必须重新配置时钟控制系统。</p> <p>1: 在停机模式时, FCLK 与 HCLK 关闭, FCLK 和 HCLK 时钟都由内部振动器提供。当退出停机模式时, 软件必须重新配置时钟控制系统。</p>
0	debug_sleep	rw	0x0	<p>调试睡眠模式 (Debug Sleep mode)</p> <p>0: 在睡眠模式时, 时钟 FCLK 开启, FCLK 保持默认配置的系统时钟, HCLK 则关闭。睡眠模式不会复位配置好的时钟系统, 因此退出睡眠模式时, 软件不需重新配置系统时钟</p> <p>1: 在睡眠模式时, FCLK 和 HCLK 时钟开启都保持由原先配置好的系统时钟提供。</p>

## 32 器件电子签名(DEVICE)

### 32.1 DEVICE 简介

器件电子签名实质就是在出厂时就已经编写好, 用来识别一个系列微控制器信息的身份标识码。通过 96 位的唯一身份标识寄存器存放在闪存存储器模块的系统存储区域。

用户可以通过外设和固件分别以字节 (8 位), 半字 (16 位) 和全字 (32 位) 读取器件电子签名, 从而自动匹配同一系列不同配置的微处理器。在任何情况下, 用户都不可以修改器件电子签名。

器件电子签名还可以用来:

- 1) 作为密码使用，在烧写闪存时，可以通过器件电子签名和软件加解密算法混合使用，从而提高代码在闪存中的安全性能
- 2) 作为序列号，用作终端应用中的序列号
- 3) 激活带安全机制的引导程序

## 32.2 寄存器描述

表 32-1 存储器容量寄存器概览

Offset	Acronym	Register Name	Reset
0x00	UID1	唯一标识码	0XXXXXXXXX
0x02	UID2	唯一标识码	0XXXXXXXXX
0x04	UID3	唯一标识码	0XXXXXXXXX
0x08	UID4	唯一标识码	0XXXXXXXXX

### 32.2.1 唯一标识码 (UID1)

基地址: 0xFFFF F7E8

地址偏移: 0x00

只读，其值在出厂时编写

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	U_ID															
Type	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
15:0	U_ID	r		U_ID: 唯一身份标志 15: 0 位 (15:0 unique ID bits) 这个域的数值也预留作为未来的其他功能。

### 32.2.2 唯一标识码 (UID2)

偏移地址: 0x02

只读，其值在出厂时编写

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	U_ID															
Type	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
15:0	U_ID	r		U_ID: 唯一身份标志 31: 16 位 (31: 16 unique ID bits) 这个域的数值也预留作为未来的其他功能。

### 32.2.3 唯一标识码 (UID3)

偏移地址: 0x04

只读, 其值在出厂时编写

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Field	U_ID															
Type	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Field	U_ID															
Type	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Field	Type	Reset	Description
31:0	U_ID	r		U_ID: 唯一身份标志 63: 32 位 (63: 32 unique ID bits) 这个域的数值也预留作为未来的其他功能。

### 32.2.4 唯一标识码 (UID4)

偏移地址: 0x08

只读, 其值在出厂时编写

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

Bit	Field	Type	Reset	Description
31:0	U_ID	r		U_ID: 唯一身份标志 95: 64 位 (95: 64 unique ID bits) 这个域的数值也预留作为未来的其他功能。