

凌思微电子（厦门）有限公司

LINKEDSEMI



LE5030 User Manual

2021/12/01

Confidential

修订记录

版本	修订日期	修订说明	作者
V1.0	2021-12-01	初始版本	gbzhang
V1.1	2021-03-01	勘误：封装命名	gbzhang

confidential

目录

第 1 章	系统概述	20
1.1	主要特性	20
1.2	系统架构框图	22
1.3	系统端口定义	23
1.3.1	封装图	23
1.3.2	端口定义	23
1.4	系统中断定义	24
1.4.1	中断优先级	24
1.4.2	中断向量分配	24
1.5	系统存储管理	25
1.5.1	存储器组织	25
1.5.2	外设地址映射	26
1.6	系统配置管理	27
1.7	系统电源管理	27
1.7.1	芯片电源	28
1.7.1.1	主系统电源域	28
1.7.1.2	备份域电源	28
1.7.2	电源监视	29
1.7.2.1	上电复位 POR 和掉电复位 (PDR)	29
1.7.2.2	可编程电压监测器 (PVD)	29
1.7.3	低功耗模式	30
1.7.3.1	低功耗模式转换	30
1.7.3.2	系统时钟速度	31
1.7.3.3	外设时钟控制	31
1.7.3.4	运行模式	31
1.7.3.5	Standby 模式	31
1.7.3.6	睡眠模式 0	32
1.7.3.7	睡眠模式 1	32
1.7.3.8	睡眠模式 2	32
1.7.3.9	ShutDown 模式	32
1.8	复位时钟管理	32
1.8.1	系统时钟	32
1.8.1.1	外部高速晶体振荡器时钟 HSE	33
1.8.1.2	内部高速 RC 振荡器时钟 HSI	34
1.8.1.3	外部低速晶体振荡器时钟 LSE	34
1.8.1.4	内部低速 RC 振荡器时钟 LSI	34
1.8.1.5	PLL 时钟	34
1.8.1.6	系统时钟选择	34
1.8.2	系统复位	34
1.8.2.1	结构框图	35
1.8.2.2	上电复位/掉电复位	35
1.8.2.3	端口复位	36

	1. 8. 2. 4 看门狗复位	36
	1. 8. 2. 5 MCU 软件系统复位(MCURST)	36
	1. 8. 2. 6 MCU 软件内核复位(CORERST)	36
	1. 8. 2. 7 外设软件复位	36
第 2 章	直接存储访问控制器 (DMA)	37
1. 1	DMA 简介	37
1. 2	DMA 主要特性	37
1. 3	结构框图	37
1. 4	功能描述	38
1. 4. 1	DMA 控制	38
1. 4. 1. 1	握手规则	38
1. 4. 1. 2	DMA 信号	40
1. 4. 2	通道控制数据结构	56
1. 4. 3	DMA 通道配置	64
1. 5	特殊功能寄存器	68
1. 5. 1	DMA 状态寄存器 (DMA_STATUS)	68
1. 5. 2	DMA 配置寄存器 (DMA_CFG)	68
1. 5. 3	DMA 通道控制数据库指针寄存器(CTRL_BASE_PTR)	69
1. 5. 4	DMA 通道交替控制数据库指针寄存器(ALT_CTRL_BASE_PTR)	69
1. 5. 5	DMA 通道等待请求状态寄存器(DMA_WAITONREQ_STATUS)	70
1. 5. 6	DMA 通道软件请求寄存器(CHNL_SW_REQUEST)	70
1. 5. 7	DMA 通道使用冲突设置寄存器(CHNL_USEBURST_SET)	70
1. 5. 8	DMA 通道使用冲突清除寄存器(CHNL_USEBURST_CLR)	71
1. 5. 9	DMA 通道请求屏蔽寄存器(CHNL_REQ_MASK_SET)	71
1. 5. 10	DMA 通道请求屏蔽清除寄存器(CHNL_REQ_MASK_CLR)	72
1. 5. 11	DMA 通道使能配置寄存器(CHNL_ENABLE_SET)	72
1. 5. 12	DMA 通道使能清除寄存器(CHNL_ENABLE_CLR)	73
1. 5. 13	DMA 通道主要交替配置寄存器(CHNL_PRI_ALT_SET)	73
1. 5. 14	DMA 通道主要交替清除寄存器(CHNL_PRI_ALT_CLR)	74
1. 5. 15	DMA 通道优先级配置寄存器(CHNL_PRIORITY_SET)	74
1. 5. 16	DMA 通道优先级清除寄存器(CHNL_PRIORITY_CLR)	75
1. 5. 17	DMA 总线错误清除寄存器(ERR_CLR)	75
1. 5. 18	DMA 中断状态寄存器 (DMA_IFLAG0)	76
1. 5. 19	DMA 中断状态清零寄存器 (DMA_ICFR0)	77
1. 5. 20	DMA 中断使能控制寄存器 (DMA_IEFR0)	77
1. 5. 21	DMA 通道 0 复用选择寄存器 (DMA_CFG0)	78
1. 5. 22	DMA 通道 1 复用选择寄存器 (DMA_CFG1)	78
第 3 章	独立看门狗 (IWDG)	80
1. 1	IWDG 简介	80
1. 2	IWDG 主要特性	80
1. 3	IWDG 功能描述	80
1. 4	IWDG 寄存器描述	81
1. 4. 1	IWDG 重装载寄存器 (IWDG_RLR)	81
1. 4. 2	IWDG 计数器值寄存器 (IWDG_VAL)	81

1. 4. 3	IWDG 控制寄存器 (IWDG_CR)	81
1. 4. 4	IWDG 中断清除寄存器 (IWDG_ICR)	82
1. 4. 5	IWDG 中断标志寄存器 (IWDG_RIF)	82
1. 4. 6	IWDG 锁定寄存器 (IWDG_LOCK)	83
第 4 章	窗口看门狗 (WWDG)	84
1. 1	WWDG 简介	84
1. 2	WWDG 主要特性	84
1. 3	WWDG 功能描述	84
1. 4	WWDG 寄存器描述	86
1. 4. 1	WWDG 重装载寄存器 (WWDG_RLR)	86
1. 4. 2	WWDG 计数值寄存器 (WWDG_VAL)	86
1. 4. 3	WWDG 控制寄存器 (WWDG_CR)	86
1. 4. 4	WWDG 中断清除寄存器 (WWDG_ICR)	87
1. 4. 5	WWDG 中断标志寄存器 (WWDG_RIF)	87
1. 4. 6	WWDG 锁定寄存器 (WWDG_LOCK)	88
第 5 章	通用和复用功能 IO (GPIO 和 AFIO)	89
1. 1	GPIO 简介	89
1. 2	GPIO 主要特性	89
1. 3	GPIO 功能描述	90
1. 3. 1	GPIO 端口控制	90
1. 3. 2	GPIO 端口数据	90
1. 3. 3	GPIO 外部端口中断	91
1. 3. 4	GPIO 端口功能复用	91
1. 4	GPIO 寄存器描述	93
1. 4. 1	GPIO 端口输入数据寄存器 (GPIO_IDR)	93
1. 4. 2	GPIO 端口输出数据寄存器 (GPIO_ODR)	93
1. 4. 3	GPIO 端口置位和复位寄存器 (GPIO_BSRR)	93
1. 4. 4	GPIO 端口锁定寄存器 (GPIO_LCKR)	94
1. 4. 5	GPIO 端口模式寄存器 (GPIO_MDR)	94
1. 4. 6	GPIO 端口输入使能寄存器 (GPIO_IER)	95
1. 4. 7	GPIO 端口输出使能寄存器 (GPIO_OER)	95
1. 4. 8	GPIO 端口上下拉寄存器 (GPIO_PUDR)	96
1. 4. 9	GPIO 端口输出类型寄存器 (GPIO_OTR)	96
1. 4. 10	GPIO 端口驱动寄存器 (GPIO_DSR)	97
1. 4. 11	GPIO 端口滤波寄存器 (GPIO_FLTR)	97
1. 4. 12	GPIO 端口模拟使能寄存器 (GPIO_AER)	97
1. 4. 13	GPIO 端口复用功能寄存器 0 (GPIO_FUNC0)	98
1. 4. 14	GPIO 端口复用功能寄存器 1 (GPIO_FUNC1)	98
1. 4. 15	GPIO 端口复用功能寄存器 2 (GPIO_FUNC2)	99
1. 4. 16	GPIO 端口复用功能寄存器 3 (GPIO_FUNC3)	99
1. 4. 17	GPIO 中断使能寄存器 (GPIO_EIER)	100
1. 4. 18	GPIO 中断禁止寄存器 (GPIO_EIDR)	100
1. 4. 19	GPIO 中断有效状态寄存器 (GPIO_EIVS)	101
1. 4. 20	GPIO 原始中断标志寄存器 (GPIO_ERIF)	101

1. 4. 21	GPIO 屏蔽中断标志寄存器 (GPIO_EIFM)	101
1. 4. 22	GPIO 中断清零寄存器 (GPIO_EICR)	102
1. 4. 23	GPIO 中断上升沿触发使能寄存器 (GPIO_ERTS)	102
1. 4. 24	GPIO 中断下降沿触发使能寄存器 (GPIO_EFTS)	102
1. 4. 25	GPIO 中断软件触发寄存器 (GPIO_ESWI)	103
1. 4. 26	GPIO 中断触发 AD 使能寄存器 (GPIO_EDTE)	103
1. 4. 27	GPIO 中断滤波使能寄存器 (GPIO_EDB)	104
1. 4. 28	GPIO 中断滤波控制寄存器 (GPIO_EDBC)	104
1. 4. 29	GPIO 中断端口选择寄存器 0 (GPIO_EICFG0)	104
1. 4. 30	GPIO 中断端口选择寄存器 1 (GPIO_EICFG1)	106
第 6 章	外设互联 (PIS)	108
1. 1	PIS 简介	108
1. 2	PIS 主要特性	108
1. 3	PIS 功能描述	109
1. 3. 1	PIS 通道选择	109
1. 3. 2	外设之间互连	109
1. 4	PIS 寄存器描述	109
1. 4. 1	PIS 通道 0 控制寄存器 (PIS_CH0CR)	109
1. 4. 2	PIS 通道 1 控制寄存器 (PIS_CH1CR)	111
1. 4. 3	PIS 通道 2 控制寄存器 (PIS_CH2CR)	111
1. 4. 4	PIS 通道 3 控制寄存器 (PIS_CH3CR)	112
1. 4. 5	PIS 通道 4 控制寄存器 (PIS_CH4CR)	112
1. 4. 6	PIS 通道 5 控制寄存器 (PIS_CH5CR)	113
1. 4. 7	PIS 通道 6 控制寄存器 (PIS_CH6CR)	114
1. 4. 8	PIS 通道 7 控制寄存器 (PIS_CH7CR)	114
1. 4. 9	PIS 通道端口输出使能寄存器 (PIS_CHOER)	115
第 7 章	椭圆曲线加密 (ECC)	116
1. 1	ECC 简介	116
1. 2	ECC 功能描述	116
1. 2. 1	有限域上的椭圆曲线运算	118
1. 2. 1. 1	计算 xG	119
1. 2. 2	椭圆曲线加解密算法原理	120
1. 2. 3	椭圆曲线签名算法原理	120
1. 2. 3. 1	签名过程	120
第 8 章	加密处理 (AES/DES)	121
1. 1	AES/DES 简介	121
1. 2	AES/DES 主要特性	121
1. 3	DES 功能描述	121
1. 3. 1	DES/TDES-ECB 模式加密	121
1. 3. 2	DES/TDES-ECB 模式解密	122
1. 3. 3	DES/TDES-CBC 模式加密	122
1. 3. 4	DES/TDES-CBC 模式解密	123
1. 4	AES 功能描述	124
1. 4. 1	AES-ECB 模式加密	124

1.4.2	AES-ECB 模式解密	124
1.4.3	AES-CBC 模式加密	125
1.4.4	AES-CBC 模式解密	126
第 9 章	真随机发生器 (TRNG)	128
1.1	TRNG 简介	128
1.2	TRNG 主要特性	128
1.3	TRNG 功能描述	128
1.3.1	操作流程	128
1.3.2	初始化时间	129
1.3.3	错误管理	129
1.3.4	随机数生成时间	129
1.3.5	随机数种子	129
第 10 章	运算加速器 (CALC)	131
1.1	CALC 简介	131
1.2	CALC 主要特性	131
1.3	CALC 功能描述	131
1.3.1	平方根运算	131
1.3.1.1	算法概述	131
1.3.1.2	完成时间	131
1.3.2	除法运算	132
1.3.2.1	算法概述	132
1.3.2.2	特例说明	132
1.3.2.3	使用说明	133
1.3.2.4	完成时间	133
第 11 章	高级定时器 (ADTIM)	135
1.1	ADTIM 简介	135
1.2	ADTIM 主要特性	135
1.3	ADTIM 功能描述	136
1.3.1	时基单位	136
1.3.2	计数器模式	138
1.3.3	重复计数器	147
1.3.4	时钟源	149
1.3.4.1	内部时钟源(CK_INT)	149
1.3.4.2	外部时钟源模式 1	150
1.3.4.3	外部时钟源模式 2	151
1.3.4.4	内部触发输入(ITRx)	152
1.3.5	捕获/比较信道	153
1.3.6	输入捕获模式	154
1.3.7	PWM 输入模式	155
1.3.8	强制输出模式	156
1.3.9	输出比较模式	156
1.3.10	PWM 模式	157
1.3.11	单脉冲模式	160
1.3.12	互补输出与死区内嵌	162

1. 3. 13	使用暂停功能.....	163
1. 3. 14	清除外部事件 OCxREF 信号	165
1. 3. 15	产生六步 PWM 输出	166
1. 3. 16	编码器接口模式	167
1. 3. 17	定时器输入 XOR 功能	169
1. 3. 18	霍尔传感器接口	169
1. 3. 19	定时器和外部触发的同步	170
1. 3. 19. 1	从模式: 复位模式	170
1. 3. 19. 2	从模式: 门控模式	171
1. 3. 19. 3	从模式: 触发模式	172
1. 3. 19. 4	从模式: 外部时钟模式 2+触发模式	173
1. 3. 20	定时器同步	173
1. 3. 20. 1	使用一个定时器使能另一个定时器	174
1. 3. 20. 2	使用一个定时器去启动另一个定时器	175
1. 3. 20. 3	使用一个外部触发同步地启动 2 个定时器	176
1. 3. 21	除错模式	177
1. 4	ADTIM 寄存器描述	177
1. 4. 1	控制寄存器 1 (ADTIM_CR1)	177
1. 4. 2	控制寄存器 2 (ADTIM_CR2)	179
1. 4. 3	从模式控制寄存器 (ADTIM_SMCR)	180
1. 4. 4	中断使能寄存器 (ADTIM_IER)	182
1. 4. 5	中断禁止寄存器 (ADTIM_IDR)	183
1. 4. 6	中断有效状态寄存器 (ADTIM_IVS)	184
1. 4. 7	原始中断标志寄存器 (ADTIM_RIF)	185
1. 4. 8	屏蔽中断标志寄存器 (ADTIM_IFM)	187
1. 4. 9	中断清除寄存器 (ADTIM_ICR)	188
1. 4. 10	事件产生寄存器 (ADTIM_EGR)	189
1. 4. 11	捕捉/比较模式寄存器 1 (ADTIM_CCMR1)	190
1. 4. 12	捕捉/比较模式寄存器 2 (ADTIM_CCMR2)	194
1. 4. 13	捕捉/比较使能寄存器 (ADTIM_CCER)	195
1. 4. 14	计数寄存器 (ADTIM_CNT)	198
1. 4. 15	预分频寄存器 (ADTIM_PSC)	198
1. 4. 16	自动重装载寄存器 (ADTIM_ARR)	199
1. 4. 17	重复计数寄存器 (ADTIM_RCR)	199
1. 4. 18	捕捉/比较寄存器 1 (ADTIM_CCR1)	200
1. 4. 19	捕捉/比较寄存器 2 (ADTIM_CCR2)	200
1. 4. 20	捕捉/比较寄存器 3 (ADTIM_CCR3)	201
1. 4. 21	捕捉/比较寄存器 4 (ADTIM_CCR4)	201
1. 4. 22	刹车和死区寄存器 (ADTIM_BDTR)	201
1. 4. 23	DMA 事件使能寄存器 (ADTIM_DMAEN)	203
第 12 章	通用 32/16 位定时器 (GPTIMA/B)	204
1. 1	GPTIMA/B 简介	204
1. 2	GPTIMA/B 主要特性	204
1. 3	GPTIMA/B 功能描述	205

1.3.1	定时器时基单位	205
1.3.2	计数器模式	206
1.3.3	时钟源	216
1.3.3.1	内部时钟源(CK_INT)	217
1.3.3.2	外部时钟源模式 1	217
1.3.3.3	外部时钟源模式 2	218
1.3.3.4	内部触发输入(ITRx)	219
1.3.4	捕获/比较信道	220
1.3.5	输入捕获模式	221
1.3.6	PWM 输入模式	222
1.3.7	强制输出模式	223
1.3.8	输出比较模式	223
1.3.9	PWM 模式	224
1.3.10	单脉冲模式	227
1.3.11	清除外部事件 OCxREF 信号	229
1.3.12	编码器接口模式	230
1.3.13	定时器输入 XOR 功能	232
1.3.14	定时器和外部触发的同步	232
1.3.14.1	从模式: 复位模式	232
1.3.14.2	从模式: 门控模式	233
1.3.14.3	从模式: 触发模式	234
1.3.14.4	从模式: 外部时钟模式 2+ 触发模式	234
1.3.15	定时器同步	235
1.3.15.1	使用一个定时器使能另一个定时器	235
1.3.15.2	使用一个定时器去启动另一个定时器	237
1.3.15.3	使用一个外部触发同步地启动 2 个定时器	238
1.3.16	除错模式	239
1.4	GPTIMA/B 寄存器描述	239
1.4.1	控制寄存器 1 (GPTIMA/B_CR1)	239
1.4.2	控制寄存器 2 (GPTIMA/B_CR2)	240
1.4.3	从模式控制寄存器(GPTIMA/B_SMCR)	241
1.4.4	中断使能寄存器 (GPTIMA/B_IER)	243
1.4.5	中断禁止寄存器 (GPTIMA/B_IDR)	244
1.4.6	中断有效状态寄存器 (GPTIMA/B_IVS)	245
1.4.7	原始中断标志寄存器 (GPTIMA/B_RIF)	246
1.4.8	屏蔽中断标志寄存器(GPTIMA/B_IFM)	247
1.4.9	中断清除寄存器(GPTIMA/B_ICR)	248
1.4.10	事件产生寄存器(GPTIMA/B_EGR)	249
1.4.11	捕捉/比较模式寄存器 1 (GPTIMA/B_CCMR1)	250
1.4.12	捕捉/比较模式寄存器 2 (GPTIMA/B_CCMR2)	253
1.4.13	捕捉/比较使能寄存器(GPTIMA/B_CCER)	254
1.4.14	计数寄存器(GPTIMA/B_CNT)	256
1.4.15	预分频寄存器(GPTIMA/B_PSC)	256
1.4.16	自动重装载寄存器(GPTIMA/B_ARR)	257

1. 4. 17	捕捉/比较寄存器 1 (GPTIMA/B_CCR1)	257
1. 4. 18	捕捉/比较寄存器 2 (GPTIMA/B_CCR2)	257
1. 4. 19	捕捉/比较寄存器 3 (GPTIMA/B_CCR3)	258
1. 4. 20	捕捉/比较寄存器 4 (GPTIMA/B_CCR4)	258
1. 4. 21	DMA 事件使能寄存器(GPTIMA/B_DMAEN)	258
第 13 章	通用定时器 (GPTIMC)	260
1. 1	GPTIMC 简介	260
1. 2	GPTIMC 主要特性	260
1. 3	GPTIMC 功能描述	261
1. 3. 1	定时器时基单位	261
1. 3. 2	计数器模式	262
1. 3. 3	重复计数器	266
1. 3. 4	时钟源	267
1. 3. 4. 1	内部时钟源(CK_INT)	267
1. 3. 4. 2	外部时钟源模式 1	267
1. 3. 4. 3	内部触发输入(ITRx)	268
1. 3. 5	捕获/比较信道	269
1. 3. 6	输入捕获模式	271
1. 3. 7	PWM 输入模式	272
1. 3. 8	强制输出模式	273
1. 3. 9	输出比较模式	273
1. 3. 10	PWM 模式	274
1. 3. 11	单脉冲模式	275
1. 3. 12	互补输出与死区内嵌	277
1. 3. 13	使用暂停功能	278
1. 3. 14	产生六步 PWM 输出	280
1. 3. 15	定时器和外部触发的同步	281
1. 3. 15. 1	从模式: 复位模式	281
1. 3. 15. 2	从模式: 门控模式	282
1. 3. 15. 3	从模式: 触发模式	283
1. 3. 15. 4	从模式: 外部时钟模式 2+触发模式	283
1. 3. 16	定时器同步	284
1. 3. 16. 1	使用一个定时器使能另一个定时器	285
1. 3. 16. 2	使用一个定时器去启动另一个定时器	286
1. 3. 16. 3	使用一个外部触发同步地启动 2 个定时器	287
1. 3. 17	除错模式	288
1. 4	GPTIMC 寄存器描述	288
1. 4. 1	控制寄存器 1 (GPTIMC_CR1)	288
1. 4. 2	控制寄存器 2 (GPTIMC_CR2)	289
1. 4. 3	从模式控制寄存器(GPTIMC_SMCR)	291
1. 4. 4	中断使能寄存器(GPTIMC_IER)	292
1. 4. 5	中断禁止寄存器(GPTIMC_IDR)	292
1. 4. 6	中断有效状态寄存器(GPTIMC_IVS)	293
1. 4. 7	原始中断标志寄存器(GPTIMC_RIF)	294

1.4.8	屏蔽中断标志寄存器(GPTIMC_IFM).....	295
1.4.9	中断清除寄存器 (GPTIMC_ICR).....	297
1.4.10	事件产生寄存器(GPTIMC_EGR)	298
1.4.11	捕捉/比较模式寄存器 1 (GPTIMC_CCMR1)	299
1.4.12	捕捉/比较使能寄存器(GPTIMC_CCER).....	302
1.4.13	计数寄存器(GPTIMC_CNT)	304
1.4.14	预分频寄存器 (GPTIMC_PSC)	305
1.4.15	自动重装载寄存器(GPTIMC_ARR).....	305
1.4.16	重复计数寄存器(GPTIMC_RCR)	305
1.4.17	捕捉/比较寄存器 1 (GPTIMC_CCR1)	306
1.4.18	捕捉/比较寄存器 2 (GPTIMC_CCR2)	306
1.4.19	刹车和死区寄存器(GPTIMC_BDTR).....	307
1.4.20	DMA 事件使能寄存器(GPTIMC_DMAEN)	308
第 14 章	基本定时器 (BSTIM)	310
1.1	BSTIM 简介	310
1.2	BSTIM 主要特性	310
1.3	BSTIM 功能描述	310
1.3.1	定时器时基单位	310
1.3.2	计数器模式	312
1.3.3	时钟源.....	316
1.3.4	除错模式	316
1.4	特殊寄存器.....	317
1.4.1	控制寄存器 1 (BSTIM_CR1).....	317
1.4.2	控制寄存器 2 (BSTIM_CR2)	318
1.4.3	中断使能寄存器(BSTIM_IER)	318
1.4.4	中断禁止寄存器(BSTIM_IDR)	319
1.4.5	中断有效状态寄存器(BSTIM_IVS)	319
1.4.6	原始中断标志寄存器(BSTIM_RIF)	319
1.4.7	屏蔽中断标志寄存器(BSTIM_IFM).....	320
1.4.8	中断清除寄存器(BSTIM_ICR)	320
1.4.9	事件产生寄存器(BSTIM_EGR)	321
1.4.10	计数寄存器(BSTIM_CNT)	321
1.4.11	预分频寄存器(BSTIM_PSC).....	322
1.4.12	自动重装载寄存器(BSTIM_ARR).....	322
1.4.13	DMA 事件使能寄存器(BSTIM_DMAEN)	322
第 15 章	低功耗定时器 (LPTIM)	324
1.1	LPTIM 简介	324
1.2	LPTIM 主要特性.....	324
1.3	结构框图	325
1.4	LPTIM 功能描述.....	325
1.4.1	数字滤波器	325
1.4.2	预分频器	326
1.4.3	触发源选择	326
1.4.4	计数模式	327

1.4.5	输出波形	328
1.4.6	寄存器更新	328
1.4.7	LPTIM 中断	329
1.5	LPTIM 寄存器描述	329
1.5.1	LPTIM 控制寄存器 0 (LPTIM_CR0)	329
1.5.2	LPTIM 控制寄存器 1 (LPTIM_CR1)	330
1.5.3	LPTIM 自动加载寄存器 (LPTIM_ARR)	331
1.5.4	LPTIM 计数寄存器 (LPTIM_CNT)	332
1.5.5	LPTIM 比较值寄存器 (LPTIM_CMP)	332
1.5.6	LPTIM 中断使能寄存器 (LPTIM_IER)	333
1.5.7	LPTIM 中断状态寄存器 (LPTIM_ISR)	333
1.5.8	LPTIM 中断状态清零寄存器 (LPTIM_ICR)	334
1.5.9	LPTIM 更新控制寄存器 (LPTIM_UPDATE)	334
1.5.10	LPTIM 写同步状态寄存器 (LPTIM_SYNCSTAT)	334
第 16 章	模拟/数字转换 (ADC)	336
1.1	ADC 简介	336
1.2	ADC 主要特性	336
1.3	ADC 结构框图	337
1.4	ADC 功能描述	337
1.4.1	ADC 控制	337
1.4.2	ADC 时钟	337
1.4.3	通道选择	338
1.4.4	单次转换模式	338
1.4.5	连续转换模式	338
1.4.6	时序图	339
1.4.7	模拟看门狗	339
1.4.8	扫描模式	340
1.4.9	注入通道管理	340
1.4.10	不连续采样模式	340
1.5	数据对齐	341
1.6	可独立设置各通道采样时间	342
1.7	数据管理	342
1.7.1	使用 DMA	342
1.7.2	在不使用 DMA 的情况下管理转换序列	343
1.7.3	在不使用 DMA 和溢出检测情况下进行转换	343
1.8	ADC 中断	343
1.9	ADC 寄存器描述	343
1.9.1	ADC 状态寄存器 (ADC_SR)	343
1.9.2	ADC 状态清零寄存器 (ADC_SFCR)	344
1.9.3	ADC 控制寄存器 1 (ADC_CR1)	345
1.9.4	ADC 控制寄存器 2 (ADC_CR2)	346
1.9.5	ADC 采样时间寄存器 1 (ADC_SMPR1)	347
1.9.6	ADC 注入通道数据偏移寄存器 1 (ADC_JOFR1)	349
1.9.7	ADC 注入通道数据偏移寄存器 2 (ADC_JOFR2)	349

1. 9. 8	ADC 注入通道数据偏移寄存器 3 (ADC_JOFR3)	350
1. 9. 9	ADC 注入通道数据偏移寄存器 4 (ADC_JOFR4)	350
1. 9. 10	ADC 看门狗高阈值寄存器 (ADC_HTR)	350
1. 9. 11	ADC 看门狗低阈值寄存器 (ADC_LTR)	351
1. 9. 12	ADC 规则转换序列寄存器 1 (ADC_RSQR1)	351
1. 9. 13	ADC 规则转换序列寄存器 2 (ADC_RSQR2)	352
1. 9. 14	ADC 注入转换序列寄存器 (ADC_JSQR)	352
1. 9. 15	ADC 转换序列长度寄存器 (ADC_SQR)	353
1. 9. 16	ADC 注入通道数据寄存器 1 (ADC_JDR1)	354
1. 9. 17	ADC 注入通道数据寄存器 2 (ADC_JDR2)	354
1. 9. 18	ADC 注入通道数据寄存器 3 (ADC_JDR3)	354
1. 9. 19	ADC 注入通道数据寄存器 4 (ADC_JDR4)	355
1. 9. 20	ADC 规则通道数据寄存器 (ADC_RDR)	355
1. 9. 21	ADC 通用控制寄存器 (ADC_CCR)	355
第 17 章	实时时钟 (RTC)	357
1. 1	RTC 简介	357
1. 2	RTC 主要特性	357
1. 3	RTC 功能描述	358
1. 3. 1	设定并开启 RTC	358
1. 3. 2	读取 RTC 日期与时间	358
1. 3. 3	RTC 校正	358
1. 3. 3. 1	RTC 校正范例 1(实际频率高于预期频率)	358
1. 3. 3. 2	RTC 校正范例 2(实际频率低于预期频率)	359
1. 3. 4	RTC 睡眠计数器	359
1. 3. 5	RTC 闹铃	359
1. 3. 6	RTC 触发 ADC 或 DAC	360
1. 4	特殊寄存器	360
1. 4. 1	RTC 控制寄存器(RTC_CTRL)	360
1. 4. 2	RTC 唤醒计数器寄存器(RTC_WKUP)	361
1. 4. 3	RTC 日历时间寄存器(RTC_TIME)	361
1. 4. 4	RTC 日历日期寄存器(RTC_CAL)	362
1. 4. 5	RTC 闹钟时间寄存器(RTC_ALTIME)	362
1. 4. 6	RTC 闹钟日期寄存器(RTC_ALCAL)	363
1. 4. 7	RTC 闹钟使能寄存器(RTC_ALEN)	363
1. 4. 8	RTC 触发源选择寄存器(RTC_TRIG)	364
1. 4. 9	RTC 中断使能寄存器(RTC_IER)	365
1. 4. 10	RTC 中断禁止寄存器(RTC_IDR)	367
1. 4. 11	RTC 中断有效状态寄存器(RTC_IVS)	368
1. 4. 12	RTC 原始中断标志寄存器(RTC_RIF)	369
1. 4. 13	RTC 屏蔽中断标志寄存器(RTC_IFM)	370
1. 4. 14	ADC 中断清除寄存器(ADC_ICR)	371
1. 4. 15	RTC 状态寄存器(RTC_STA)	373
1. 4. 16	RTC 读备份域寄存器(RTC_BKEN)	373
第 18 章	集成内部总线接口 (I2C)	374

1.1	I2C 简介	374
1.2	I2C 主要特性	374
1.3	I2C 功能描述	375
1.3.1	I2C 总线协议	375
1.3.1.1	I2C 启动和停止条件	375
1.3.1.2	I2C 应答响应	376
1.3.1.3	I2C 寻址协议	377
1.3.1.4	I2C 发送/接收协议	377
1.3.1.5	I2C 启动字节传输协议	378
1.3.2	I2C 时钟要求	379
1.3.3	I2C 数据传输	379
1.3.4	I2C 从机模式	380
1.3.5	I2C 主机模式	383
1.3.6	I2C_TIMINGR 寄存器配置举例	385
1.3.7	SMBus 特定功能	385
1.3.8	SMBus 初始化	387
1.3.9	SMBus: I2C_TIMEOUTR 寄存器配置举例	389
1.3.10	I2C DMA 请求	389
1.3.11	I2C 错误条件	390
1.3.12	I2C 中断	391
1.4	I2C 寄存器描述	391
1.4.1	I2C 控制寄存器 1 (I2C_CR1)	391
1.4.2	I2C 控制寄存器 2 (I2C_CR2)	393
1.4.3	I2C 本机地址 1 寄存器 (I2C_OAR1)	395
1.4.4	I2C 本机地址 2 寄存器 (I2C_OAR2)	396
1.4.5	I2C 时序寄存器 (I2C_TIMINGR)	396
1.4.6	I2C 超时寄存器 (I2C_TIMEOUTR)	397
1.4.7	I2C 状态寄存器 (I2C_SR)	398
1.4.8	I2C 清标志寄存器 (I2C_CFR)	400
1.4.9	I2C PEC 寄存器 (I2C_PECR)	401
1.4.10	I2C 接收数据寄存器 (I2C_RXDR)	401
1.4.11	I2C 发送数据寄存器 (I2C_TXDR)	402
1.4.12	I2C 中断使能寄存器 (I2C_IER)	402
1.4.13	I2C 中断禁止寄存器 (I2C_IDR)	403
1.4.14	I2C 中断有效状态寄存器 (I2C_IVS)	404
1.4.15	I2C 原始中断标志寄存器 (I2C_RIF)	405
1.4.16	I2C 屏蔽中断标志寄存器 (I2C_IFM)	406
1.4.17	I2C 中断清除标志 (I2C_ICR)	407
第 19 章	串行外设接口 1 (SSI)	409
1.1	SSI 简介	409
1.2	SSI 功能描述	410
1.2.1	时钟比率	411
1.2.2	发送/接收 FIFO 缓冲	412
1.2.3	SSI 中断	413

1.2.4	传输模式	413
1.2.4.1	发送和接收	413
1.2.4.2	仅发送	414
1.2.4.3	仅接收	414
1.2.4.4	读 EEPROM	414
1.2.5	操作模式	414
1.2.5.1	串行主设备模式	414
1.2.5.2	串行从设备模式	420
1.2.6	接口协议	423
1.2.6.1	摩托罗拉串行外围接口协议(SPI)	424
1.2.6.2	德州仪器同步串行协议(SSP)	427
1.2.6.3	国家半导体微线协议	428
1.2.6.4	SPI 增强模式	437
1.3	SSI 寄存器描述	442
1.3.1	SSI 控制寄存器 0(SSI_CR0)	442
1.3.2	SSI 控制寄存器 1(SSI_CR1)	445
1.3.3	LSSP 使能寄存器(SSI_ENR)	445
1.3.4	SSI 微线控制寄存器(SSI_NWCR)	446
1.3.5	SSI 从设备使能寄存器(SSI_SER)	446
1.3.6	SSI 波特率寄存器(SSI_BAUD)	447
1.3.7	SSI 发送 FIFO 阈值寄存器(SSI_TXFTL)	447
1.3.8	SSI 接收 FIFO 阈值寄存器(SSI_RXFTL)	448
1.3.9	SSI 发送 FIFO 指针寄存器(SSI_TXFL)	448
1.3.10	SSI 接收 FIFO 指针寄存器(SSI_RXFL)	448
1.3.11	SSI 状态寄存器(SSI_SR)	449
1.3.12	SSI 中断屏蔽寄存器(SSI_IMR)	450
1.3.13	SSI 中断状态寄存器(SSI_ISR)	451
1.3.14	SSI 中断标志寄存器(SSI_RISR)	451
1.3.15	SSI 发送上溢清除寄存器(SSI_TXOICR)	452
1.3.16	SSI 接收上溢清除寄存器(SSI_RXOICR)	452
1.3.17	SSI 接收下溢清除寄存器(SSI_RXUICR)	453
1.3.18	SSI 多主竞争清除寄存器(SSI_MSTICR)	453
1.3.19	SSI 中断清除寄存器(SSI_ICR)	453
1.3.20	SSI DMA 控制寄存器(SSI_DMACR)	454
1.3.21	SSI DMA 发送数据量寄存器(SSI_DMATDLR)	454
1.3.22	SSI DMA 接收数据量寄存器(SSI_DMARDLR)	455
1.3.23	SSI 数据寄存器(SSI_DR)	455
1.3.24	SSI 接收采样延时寄存器(SSI_RSD)	455
1.3.25	SSI SPI 控制 0 寄存器(SSI_SPICR0)	456
第 20 章	串行外设接口 2(SPI)	458
1.1	SPI 简介	458
1.2	SPI 主要特性	458
1.3	SPI 功能描述	460
1.3.1	基本描述	460

1.3.2	一个主设备和一个从设备之间的通信	461
1.3.2.1	全双工通信	461
1.3.2.2	半双工通信	461
1.3.2.3	简单通信	462
1.3.3	标准多从机通信	462
1.3.4	多主机通信	463
1.3.5	从机选择(NSS)引脚管理	464
1.3.6	通信格式	465
1.3.6.1	时钟相位和极性控制	465
1.3.6.2	数据帧格式	466
1.3.7	SPI 状态标志	466
1.3.7.1	Tx 缓冲器空标志 (TXE)	466
1.3.7.2	Rx 缓冲非空 (RXNE)	466
1.3.7.3	忙标志 (BSY)	467
1.3.8	SPI 错误标志	467
1.3.8.1	溢出标志 (OVR)	467
1.3.8.2	模式故障标志 (MODF)	467
1.3.8.3	CRC 错误 (CRCERR)	467
1.3.8.4	TI 模式帧格式错误(FRE)	468
1.4	SPI 特殊功能	468
1.4.1	TI 模式	468
1.4.2	CRC 计算	468
1.4.2.1	CRC 原则	468
1.4.2.2	CRC 传输管理	468
1.4.2.3	重置 SPIx_TXCRC 和 SPIx_RXCRC 值	469
1.5	I2S 基本描述	469
1.5.1	I2S 全双工	471
1.5.2	支持的音频协议	472
1.5.2.1	I2S 飞利浦标准	472
1.5.2.2	MSB 对齐标准	474
1.5.2.3	LSB 对齐标准	475
1.5.2.4	PCM 标准	477
1.5.3	时钟发生器	477
1.5.4	I2S 主模式	479
1.5.4.1	流程	479
1.5.4.2	发送流程	480
1.5.4.3	接收流程	480
1.5.5	I2S 从模式	480
1.5.5.1	发送流程	481
1.5.5.2	接收流程	481
1.5.6	I2S 状态标志位	482
1.5.6.1	忙标志位(BSY)	482
1.5.6.2	Tx 缓冲空标志 (TXE)	482
1.5.6.3	RX 缓冲非空 (RXNE)	482

1. 5. 6. 4	声道标志位(CHSIDE).....	483
1. 5. 7	I2S 错误标志.....	483
1. 5. 7. 1	下溢标志 (UDR)	483
1. 5. 7. 2	上溢标志(OVR)	483
1. 5. 7. 3	帧错误标志 (FRE)	483
1. 6	SPI 寄存器描述	484
1. 6. 1	SPI 控制寄存器 1 (SPI_CR1)	484
1. 6. 2	SPI 控制寄存器 2 (SPI_CR2)	485
1. 6. 3	SPI 状态寄存器 (SPI_SR)	487
1. 6. 4	SPI 数据寄存器 (SPI_DR)	488
1. 6. 5	SPI CRC 多项式寄存器 (SPI_CRCPR)	488
1. 6. 6	SPI 接收 CRC 寄存器 (SPI_RXCRCR)	489
1. 6. 7	SPI 发送 CRC 寄存器 (SPI_TXCRCR)	489
1. 6. 8	SPI_I2S 配置寄存器 (SPI_I2SCFGR)	490
1. 6. 9	SPI_I2S 预分频寄存器 (SPI_I2SPR)	491
1. 6. 10	SPI 中断使能寄存器 (SPI_IER)	492
1. 6. 11	SPI 中断禁止寄存器 (SPI_IDR)	492
1. 6. 12	SPI 中断有效状态寄存器(SPI_IVS)	493
1. 6. 13	SPI 原始中断标志寄存器(SPI_RIF)	494
1. 6. 14	SPI 屏蔽中断标志寄存器 (SPI_IFM)	495
1. 6. 15	SPI 中断清除寄存器(SPI_ICR)	495
第 21 章	通用异步收发器(UART).....	497
1. 1	UART 简介	497
1. 2	UART 主要特性.....	497
1. 3	UART 功能描述.....	498
1. 3. 1	UART 特性描述.....	499
1. 3. 2	UART 发送.....	500
1. 3. 3	UART 接收.....	502
1. 3. 3. 1	防抖电路.....	502
1. 3. 3. 2	起始位检测	503
1. 3. 4	UART 状态寄存器.....	506
1. 3. 5	UART 波特率生成.....	507
1. 3. 5. 1	如何设置 UART_BRR 寄存器值.....	507
1. 3. 6	UART 自动波特率检测	508
1. 3. 7	UART 自动流控制.....	510
1. 3. 7. 1	Auto-RTSn	510
1. 3. 7. 2	Auto-CTSn	511
1. 3. 7. 3	RS485 驱动使能(DE)	511
1. 3. 8	UART Modbus 通信.....	512
1. 3. 9	UART 奇偶校验控制	512
1. 3. 10	UART 多处理器通信	513
1. 3. 11	UART LIN (本地互连)模式.....	514
1. 3. 12	UART 单线半双工通信	516
1. 3. 13	UART 智能卡模式.....	516

1. 3. 14	UART 红外编解码器模块	520
1. 3. 15	UART 连续通信使用 DMA 模式	522
1. 3. 16	UART 中断	524
1. 4	UART 寄存器描述	525
1. 4. 1	接收数据寄存器(UART_RDR)	525
1. 4. 2	发送数据寄存器(UART_TDR)	525
1. 4. 3	波特率寄存器 (UART_BRR)	526
1. 4. 4	线路控制寄存器(UART_LCR)	526
1. 4. 5	模式控制寄存器(UART_MCR)	528
1. 4. 6	控制寄存器(UART_CR)	530
1. 4. 7	智能卡控制寄存器(UART_RTOR)	531
1. 4. 8	FIFO 控制寄存器(UART_FCR)	532
1. 4. 9	状态寄存器(UART_SR)	533
1. 4. 10	中断使能寄存器(UART_IER)	534
1. 4. 11	中断禁止寄存器(UART_IDR)	535
1. 4. 12	中断有效状态寄存器(UART_IVS)	536
1. 4. 13	原始中断标志寄存器(UART_RIF)	537
1. 4. 14	屏蔽中断标志寄存器(UART_IFM)	538
1. 4. 15	中断清除寄存器(UART_ICR)	539
第 22 章	音频接口 (PDM)	541
1. 1	PDM 简介	541
1. 2	PDM 功能描述	541
1. 3	PDM 寄存器描述	543
1. 3. 1	PDM 控制寄存器(PDM_CR)	543
1. 3. 2	PDM 配置寄存器(PDM_CFG)	543
1. 3. 3	PCM 数据寄存器 0(PCM_DATA0)	544
1. 3. 4	PCM 数据寄存器 1(PCM_DATA1)	544
1. 3. 5	PDM 中断使能寄存器(PDM_IER)	544
1. 3. 6	PDM 中断禁止寄存器(PDM_IDR)	545
1. 3. 7	PDM 中断有效寄存器(PDM_IVS)	545
1. 3. 8	PDM 原始中断标志寄存器(PDM_RIF)	545
1. 3. 9	PDM 屏蔽中断寄存器(PDM_IFM)	546
1. 3. 10	PDM 中断清除寄存器(PDM_ICR)	546
1. 3. 11	PDM 滤波器参数配置(PDM_COEFx)	547
第 23 章	触摸按键 (TK)	548
1. 1	TK 简介	548
1. 2	TK 主要特性	548
1. 3	TK 功能描述	548
1. 4	TK 用户参数调试与设置流程	549
1. 5	TK 中断操作波形示意图	549
1. 6	工作模式	550
1. 7	TK 寄存器描述	550
1. 7. 1	TK 滑条通道号寄存器(TK_SSRX)	550
1. 7. 2	TK 滑条通道数目寄存器(TK_SLR)	550

1. 7. 3	TK 滑条类型寄存器(TK_STR)	551
1. 7. 4	TK 配置寄存器(TK_SCFR)	551
1. 7. 5	TK 防抖参数寄存器(TK_DBR)	551
1. 7. 6	TK 通道使能寄存器 1(TK_CHELRL)	551
1. 7. 7	TK 通道使能寄存器 2(TK_CHEHRL)	552
1. 7. 8	TK 全局灵敏度寄存器(TK_GSR)	552
1. 7. 9	TK 控制寄存器(TK_SCR)	552
1. 7. 10	TK 滑条位置寄存器(TK_SPR)	553
1. 7. 11	TK 通道检测寄存器 1(TK_CNTRLR)	553
1. 7. 12	TK 通道检测寄存器 2(TK_CNTHRL)	553
1. 7. 13	TK 校验和寄存器(TK_CSR)	553
1. 7. 14	TK 补偿电流寄存器(TK_ICGSR)	554
1. 7. 15	TKCDC 寄存器(TK_CFDR)	554
1. 7. 16	TK 低功耗扫描寄存器(TK_LPSPR)	554
1. 7. 17	TK 消拍寄存器(TK_KDCR)	554
1. 7. 18	TK LD 寄存器(TK_LDR)	555
1. 7. 19	TK 扫描周期寄存器(TK_SCTR)	555
1. 7. 20	TK 唤醒阈值寄存器(TK_WTR)	555
1. 7. 21	TK 通道信号寄存器 1(TK_SIGLRx)	556
1. 7. 22	TK 通道信号寄存器 2(TK_SIGHRLx)	556
1. 7. 23	TK 写保护寄存器(TK_WPR)	556
1. 7. 24	TK 使能通道寄存器 0(TK_GMR0)	556
1. 7. 25	TK 使能通道寄存器 1(TK_GMR1)	557
1. 7. 26	TK 单灵敏度寄存器(TK_LSRx)	557
1. 7. 27	TK 群灵敏度寄存器(TK_LSGR)	557

第1章 系统概述

1.1 主要特性

- ◆ 蓝牙 BLE5.0 / BLE5.1
 - 支持 125Kbps/500Kbps/1Mbps/2Mbps
 - 接收灵敏度: -99.7dBm @1Mbps
-97dBm @2Mbps
-105dBm @125kbps
 - 发送功率: +12dBm (最大)
 - 链路增益: 117dB @125kbps (最大)
 - 支持 Single-ended antenna Output
- ◆ 蓝牙 MESH
 - 支持 Bluetooth SIG Mesh
 - 支持私有 MESH
- ◆ MCU 内核
 - 32 位 CPU 内核
 - 最高频率可达 64MHz
 - 最大 64kB SRAM 存储
 - 最大 512kB Flash 存储
 - 支持双线调试协议
- ◆ 系统功耗
 - RX 模式: 4.5mA @3.3V
 - TX 模式: 4.3mA @3.3V 0dBm
 - 深度睡眠模式: 1.1uA (GPIO 唤醒+RTC 唤醒)
 - ShutDown 模式: 600nA (GPIO 唤醒)
- ◆ 电源、复位
 - 主电源域(VDD33)
 - 工作电压范围: $1.8V \leq VDD33 \leq 3.6V$
 - POR,BOR,LVD
- ◆ 时钟
 - 外部高速晶体振荡器: 16MHz
 - 内部高速 RC 振荡器: 24MHz
 - 外部低速晶体振荡器: 32.768KHz
 - 内部低速 RC 振荡器: 32.768KHz
- ◆ 系统外设

- DMA: 支持 8 个多路复用通道
- 看门狗定时器: IWDG 和 WWDG
- 外设互联 PIS
- 系统节拍定时器
- ◆ 安全及运算加速单元
 - ECC 椭圆曲线加密 (256)
 - AES 高级加密 (256/192/128)
 - T/DES 高级加密 (192/128/64)
 - 真随机数发生器 (TRNG)
 - 运算加速器 (CALC)
- ◆ 定时器
 - 1 个高级 16 位定时器 (ADTIM), 支持 4 组 PWM, 其中 3 组支持死区互补
 - 1 个通用 32 位定时器 A (GPTIMA), 支持 4 路 PWM
 - 1 个通用 16 位定时器 B (GPTIMB), 支持 4 路 PWM
 - 1 个通用 16 位定时器 C (GPTIMC), 支持 2 组 PWM, 其中 1 组支持死区互补
 - 1 个基本定时器 (BSTIM)
 - 1 个低功耗定时器 (LPTIM)
- ◆ 实时时钟器 RTC
 - 支持高精度硬件温补
- ◆ 模数转换 ADC
 - 12 位高精度 SAR ADC
 - 最大支持 9 路外部通道
 - 支持 3 路内部通道
 - 内置温度传感器
 - 1/8, 1/4, 1/2, 3/8 VDD
 - 内部 1.4V 参考
- ◆ 通信接口
 - 2 个 I2C 接口 (支持总线仲裁)
 - 2 个 SPI 接口
 - 3 个 UART (可支持 ISO7816, LIN, IrDA 等)
- ◆ 音频接口
 - 2 路 PDM 接口, 支持数字 MIC
 - 1 路 I2S 接口
- ◆ 触控按键 TK
 - 最多支持 14 通道 TK 按键

◆ 通用 IO

- 最大支持 20 个通用 IO

1.2 系统架构框图

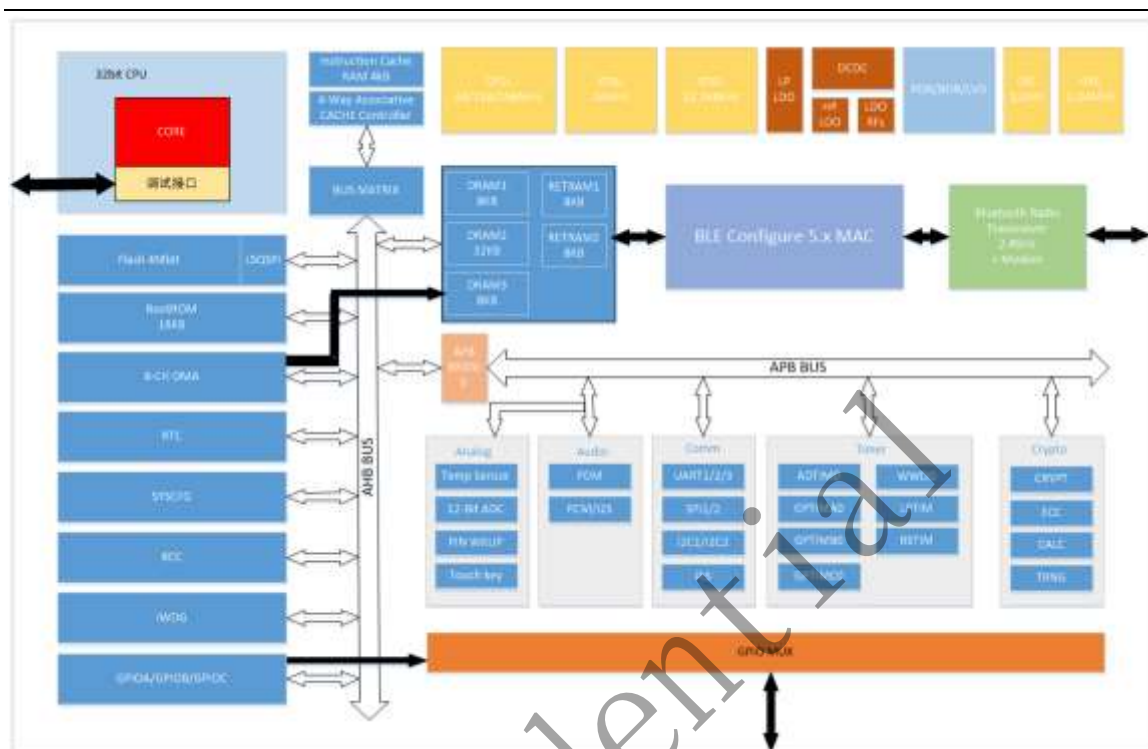


图 1-1 系统架构框图

1.3 系统端口定义

1.3.1 封装图

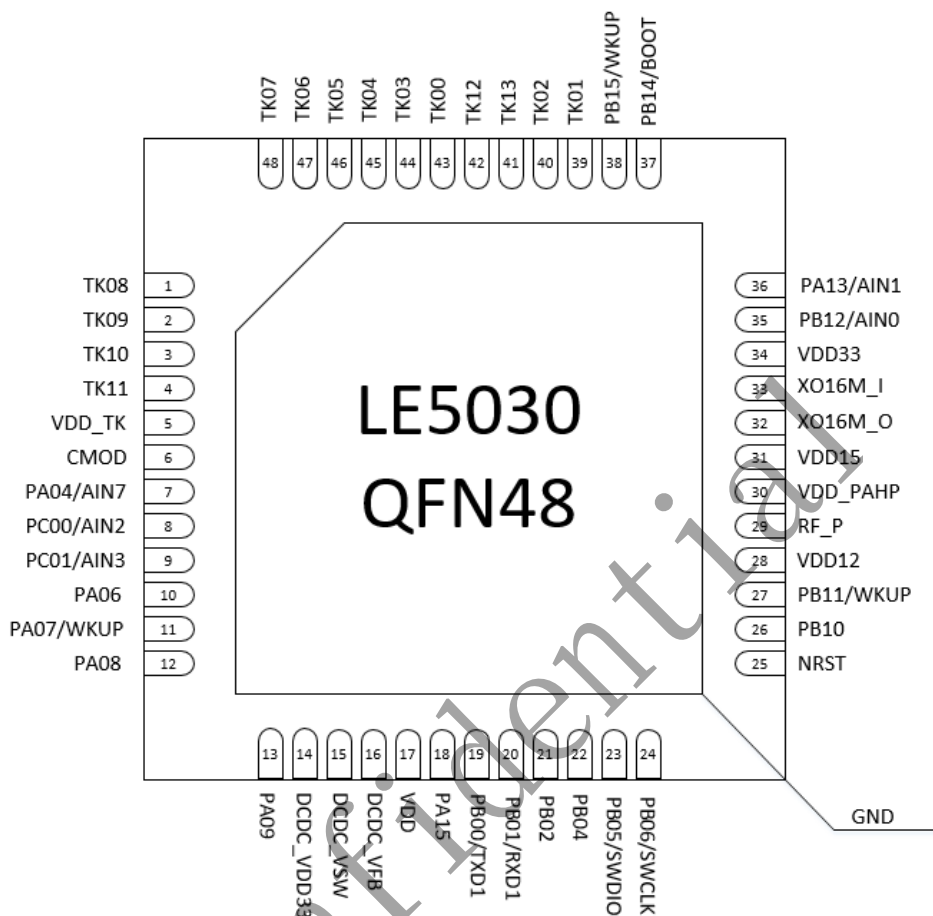


图 1-2 QFN48 封装图

1.3.2 端口定义

序号	信号名	描述
1	TK08	Touch key 通道 08
2	TK09	Touch key 通道 09
3	TK10	Touch key 通道 10
4	TK11	Touch key 通道 11
5	VDD_Tk	Touch key 供电电压
6	CMOD	Touch 稳压电容, 接 1nF~5.6nF
7	PA04	GPIO/ADC 通道 7
8	PC00	GPIO/ADC 通道 2
9	PC01	GPIO/ADC 通道 3
10	PA06	GPIO
11	PA07	GPIO/睡眠唤醒
12	PA08	GPIO

13	PA09	GPIO
14	DCDC_V33	Buck 3.3V 输入
15	DCDC_VSW	Buck SW 调制输出
16	DCDC_VFB	Buck 反馈电压
17	VDD33	芯片 3.3V 电源
18	PA15	GPIO
19	PB00	GPIO/UART1_TXD
20	PB01	GPIO/UART1_RXD
21	PB02	GPIO
22	PB04	GPIO
23	PB05	GPIO (调试数据接口)
24	PB06	GPIO (调试时钟接口)
25	NRST	芯片复位引脚
26	PB10	GPIO
27	PB11	GPIO/睡眠唤醒
28	VDD12	芯片 1.2V 电源
29	RF_P	射频引脚
30	VDD_PAHP	高功率 PA 电源引脚
31	VDD15	芯片 1.5V 电源
32	XO16M_O	16M 晶振输出
33	XO16M_I	16M 晶振输入
34	VDD33	芯片 3.3V 电源
35	PB12	GPIO/ADC 通道 0
36	PB13	GPIO/ADC 通道 1
37	PB14	GPIO/BOOT 控制
38	PB15	GPIO/睡眠唤醒
39	TK01	Touch key 通道 01
40	TK02	Touch key 通道 02
41	TK13	Touch key 通道 13
42	TK12	Touch key 通道 12
43	TK00	Touch key 通道 00
44	TK03	Touch key 通道 03
45	TK04	Touch key 通道 04
46	TK05	Touch key 通道 05
47	TK06	Touch key 通道 06
48	TK07	Touch key 通道 07

表 1-1 QFN48 封装引脚定义

1.4 系统中断定义

1.4.1 中断优先级

中断优先级寄存器(Interrupt Priority Register)每个 byte 的高 2 位为有效位, 支持 4 个中断优先级设置。

1.4.2 中断向量分配

编号	优先级	名称	说明
0			保留
1	-3	Reset	复位
2	-2	NMI	不可屏蔽中断。

3	-1	HardFault	所有类型的错误
4~10			
11		SVCall	通过 SWI 指令调用的系统服务
12		DebugMonitor	调试监视器
13	-	-	保留
14		PendSV	可挂起的系统服务
15	可设置	Systick	系统定时器
16	可设置	EXTI_INTR	外部端口中断
17	可设置	WWDG_INTR	WWDG 中断
18	可设置	LPWKUP_INTR	LPWKUP 中断
19	可设置	BLE_INTR	BLE 中断
20	可设置	RTC_INTR	RTC 中断
21	可设置	DMA_INTR	DMA 中断
22	可设置	QSPI_INTR	QSPI 中断
23	可设置	ECC_INTR	ECC 中断
24	可设置	CACHE_INTR	CACHE 中断
25	可设置	TRNG_INTR	TRNG 中断
26	可设置	IWDG_INTR	IWDG 中断
27	可设置	CRYPT_INTR	CRYPT 中断
28	可设置	PDM_INTR	PDM 中断
29	可设置	BLE_WKUP_INTR	BLE_WKUP 中断
30	可设置	ADC_INTR	ADC 中断
31	可设置	ADTIM1_INTR	ADTIM1 中断
32	可设置	BSTIM1_INTR	BSTIM1 中断
33	可设置	GPTIMA1_INTR	GPTIMA1 中断
34	可设置	GPTIMB1_INTR	GPTIMB1 中断
35	可设置	BLEERR_INTR	BLE_ERR 中断
36	可设置	LVD33_INTR	LVD33 中断
37	可设置	GPTIMC1_INTR	GPTIMC1 中断
38	可设置	LPTIM_INTR	LPTIM 中断
39	可设置	I2C1_INTR	I2C1 中断
40	可设置	I2C2_INTR	I2C2 中断
41	可设置	SSI_INTR	SSI 中断
42	可设置	SPI2_INTR	SPI2 中断
43	可设置	UART1_INTR	UART1 中断
44	可设置	UART2_INTR	UART2 中断
45	可设置	UART3_INTR	UART3 中断
46	可设置	BLE_FIFO_INTR	BLEFIFO 中断
47	可设置	BLE_CRYPT_INTR	BLECRYPT 中断

表 1-2 中断向量分配

1.5 系统存储管理

1.5.1 存储器组织

地址范围	存储	备注
0x1000_0000~0x1000_3FFF	Boot ROM 存储器	
0x1800_0000~0x1FFF_FFFF	Flash 存储器	

0x2000_0000~0x2000_1FFF	DATA SRAM0 存储器
0x2000_2000~0x2000_9FFF	DATA SRAM1 存储器
0x2000_A000~0x2000_BFFF	DATA SRAM2 存储器
0x4000_0000~0x4000_7FFF	APB 外设 1
0x4001_0000~0x4001_7FFF	APB 外设 2
0x4002_0000~0x4002_3FFF	AHB 外设 1
0x4800_0000~0x4800_3FFF	AHB 外设 2
0x5000_0000~0x5000_3FFF	BLE BB
0x5000_4000~0x5000_5FFF	扩展 SRAM
0xE000_0000~0xE00F_FFFF	私有外设
0xF000_0000~0xFFFF_FFFF	ROM ID

表 1-3 存储器组织

1.5.2 外设地址映射

总线	边界地址	外设
APB1	0x4000_0000 - 0x4000_03FF	GPTIMA1
	0x4000_0400 - 0x4000_07FF	GPTIMB1
	0x4000_0800 - 0x4000_0FFF	—
	0x4000_1000 - 0x4000_13FF	BSTIM1
	0x4000_1400 - 0x4000_1BFF	—
	0x4000_1C00 - 0x4000_1FFF	TRNG
	0x4000_2000 - 0x4000_23FF	ECC
	0x4000_2400 - 0x4000_27FF	CALC
	0x4000_2800 - 0x4000_2BFF	CRYPT
	0x4000_2C00 - 0x4000_2FFF	WWDG
	0x4000_3000 - 0x4000_37FF	—
	0x4000_3800 - 0x4000_3BFF	SPI2
	0x4000_3C00 - 0x4000_43FF	—
	0x4000_4400 - 0x4000_47FF	UART2
	0x4000_4800 - 0x4000_4BFF	UART3
	0x4000_4C00 - 0x4000_53FF	—
	0x4000_5400 - 0x4000_57FF	I2C1
	0x4000_5800 - 0x4000_5BFF	I2C2
	0x4000_5C00 - 0x4000_67FF	—
	0x4000_6800 - 0x4000_6BFF	PDM
	0x4000_6C00 - 0x4000_73FF	—
	0x4000_7400 - 0x4000_77FF	BLE_RF
	0x4000_7800 - 0x4000_7BFF	BLE_MDM
	0x4000_7C00 - 0x4000_7FFF	—
APB2	0x4001_0000 - 0x4001_03FF	LPTIM
	0x4001_0400 - 0x4001_07FF	EXTI
	0x4001_0800 - 0x4001_0FFF	—

	0x4001 1000 - 0x4001 13FF	DMA
	0x4001 1400 - 0x4001 17FF	
	0x4001 1800 - 0x4001 1BFF	
	0x4001 1C00 - 0x4001 1FFF	
	0x4001 2000 - 0x4001 23FF	
	0x4001 2400 - 0x4001 27FF	
	0x4001 2800 - 0x4001 2BFF	ADC
	0x4001 2C00 - 0x4001 2FFF	—
	0x4001 3000 - 0x4001 33FF	ADTIM1
	0x4001 3400 - 0x4001 37FF	SSI
	0x4001 3800 - 0x4001 3BFF	—
	0x4001 3C00 - 0x4001 3FFF	UART1
	0x4001 4000 - 0x4001 43FF	—
	0x4001 4400 - 0x4001 47FF	GPTIMC1
	0x4001 4800 - 0x4001 4BFF	—
	0x4001 4C00 - 0x4001 4FFF	PIS
	0x4001 5000 - 0x4001 53FF	—
	0x4001 5400 - 0x4001 57FF	CACHE
	0x4001 5800 - 0x4001 5FFF	—
	0x4002 0000 - 0x4002 03FF	LSQSPI
AHB1	0x4002 0400 - 0x4002 07FF	—
	0x4002 0800 - 0x4002 0BFF	RCC
	0x4002 0C00 - 0x4002 0FFF	SYSCFG
	0x4002 1000 - 0x4002 13FF	RTC
	0x4002 1400 - 0x4002 17FF	IWDG
	0x4002 1800 - 0x4002 1BFF	—
	0x4002 1C00 - 0x4002 1FFF	—
AHB2	0x4002 2000 - 0x4002 23FF	GPIOA
	0x4002 2400 - 0x4002 27FF	GPIOB
	0x4002 2800 - 0x4002 2BFF	GPIOC
	0x4002 2C00 - 0x4002 2FFF	—

表 1-4 外设地址映射

1.6 系统配置管理

- ◆ 支持 Boot Rom 重映射
- ◆ 支持 M0 内核复位
- ◆ 支持备份域读/写

1.7 系统电源管理

LE5030 工作电压（VDD33）为 1.7~3.6V，通过内置的电压调节器提供所需的 1.2V 电源。电源管理单元（PMU）管理芯片的电源以及低功耗模式。在每个低功耗模式下，都有其对应的单元模块状态（使能、中止或掉电）。芯片共分为 6 种功耗模式（Run 模式，Standby 模式，睡眠模式 0，睡眠模式 1，睡眠模式 2，ShutDown 模式）。其中 Run 为芯片正常运行模式，所有的外设模块均可被使能。其余为低功耗模式。Standby 模式为 CPU 最低可恢复模式，在睡眠模式 0 和睡眠模式 1 时，CPU 掉电，大部分外设被中止，DRAM 中数据保持，唤醒之后外

设继续运行，CPU 重新运行。睡眠模式 2 和 ShutDown 模式 1.2V 电源域会掉电，3.3V 电源域寄存器数据保持，GPIO 状态保持。

低功耗模式通过软件操作使能。Standby 模式可通过一系列中断或事件唤醒回到 Run 模式，睡眠模式 0 可以通过 BLE LowPower Timer，RTC，IWDG 和外部 WKUP 引脚唤醒，睡眠模式 2 可以通过 RTC，IWDG 和外部 WKUP 引脚唤醒，睡眠模式 1 和 ShutDown 模式仅可通过外部 WKUP 引脚唤醒回到 Run 模式。

PMU 也可将不需要使用的 RAM 模块关闭以降低芯片总功耗。

- ◆ 支持多种低功耗模式配置
- ◆ 支持多种唤醒源灵活配置
- ◆ 快速的唤醒时间

1.7.1 芯片电源

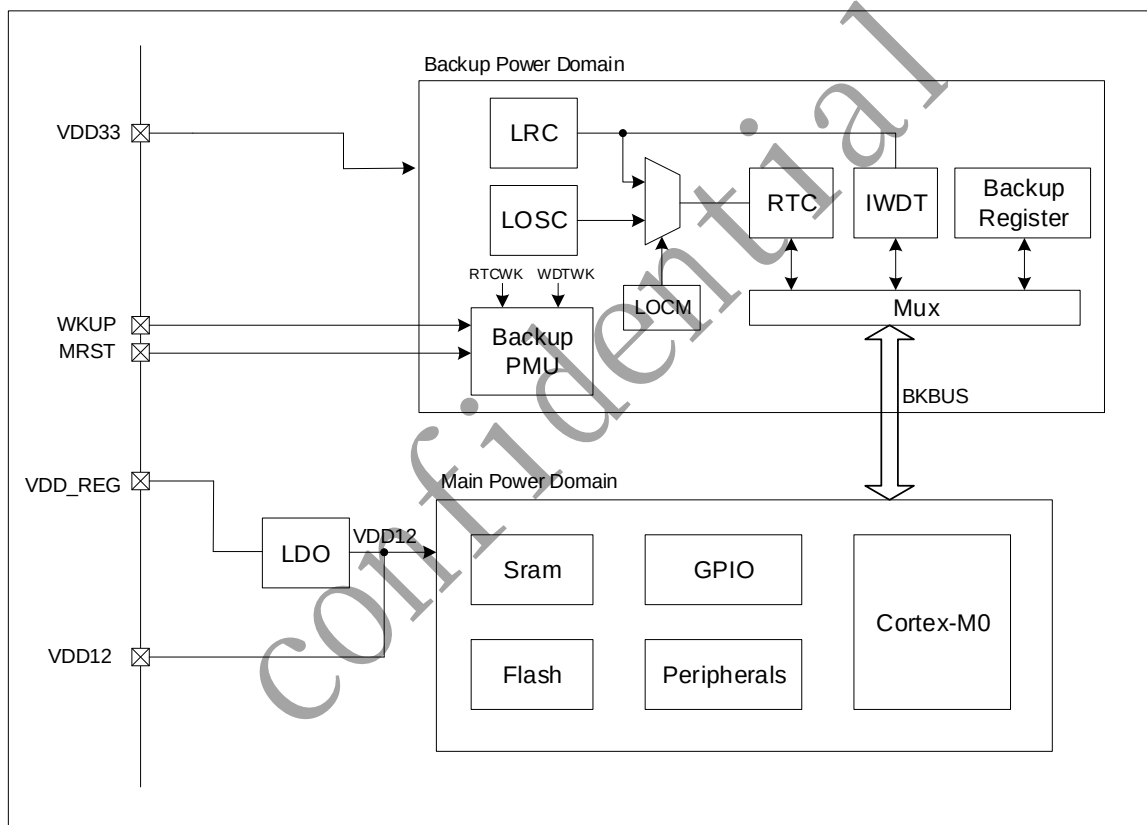


图 1-1 电源示意图

1.7.1.1 主系统电源域

芯片工作电压 VDD33 要求介于 1.7 V 到 3.6V 之间。嵌入式 LDO 用于提供内部 1.2V 数字电源。

芯片备份域电源 VDD33 单独为备份域 RTC、IWDG 和电源管理模块供电。

1.7.1.2 备份域电源

要在 VDD12 关闭后保留备份寄存器的内容并为 RTC 和 IWDG 供电，可以将 VDD33 引脚连接到通过电池或其它电源供电的可选备用电压。

要使 RTC 即使在主数字电源（VDD12）关闭后仍然工作，VDD33 引脚需为以下各模块供电：

- RTC
- IWDG
- LSI
- 备份寄存器

1.7.2 电源监视

1.7.2.1 上电复位 POR 和掉电复位（PDR）

芯片内部集成完整的上电复位（POR）和掉电复位（PDR）电路，当供电电压达到 1.7V 时系统即能正常工作。

当 VDD33 低于指定限位电压 V_{POR}/V_{PDR} 阈值时，系统保持复位状态，而无需外部复位电路。

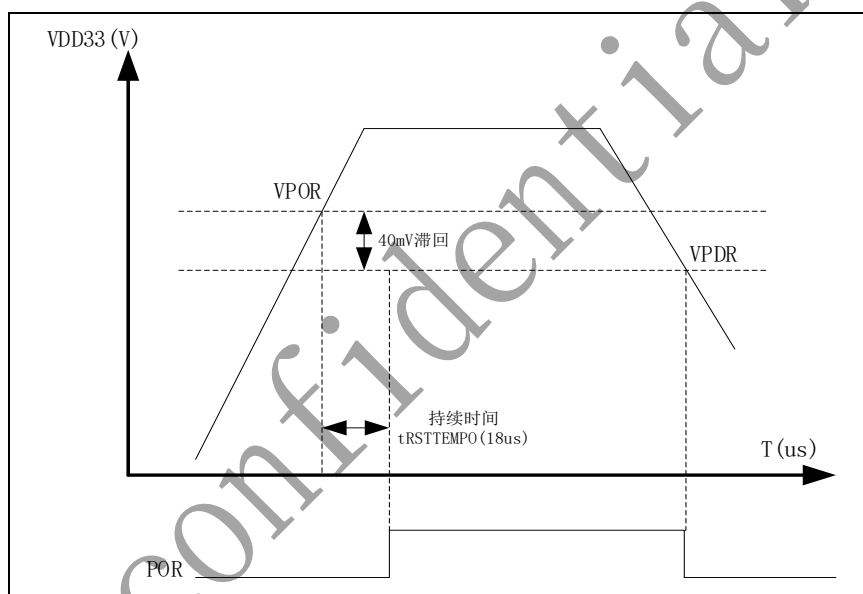


图 1-2 POR/PDR 示意图

1.7.2.2 可编程电压监测器（PVD）

用户可以利用 PVD 电路对 VDD33 电压进行检测，当通过设置 PVD 电路使能位使能 PVD 电路，将 VDD33 电压和设置的监控电压阈值进行比较，当 VDD33 电压低于设置的监控电压阈值时，将产生一个低压监控中断。

PVD 提供了一个状态标志位 PVD_flag，用于指示 VDD33 是大于还是小于 PVD 阈值。该功能的用处之一就是可以在 VDD33 发生跌落时，立即进入中断服务程序中执行紧急关闭系统的任务，若外部有电池供电，则可进入低功耗模式并切换至电池供电。

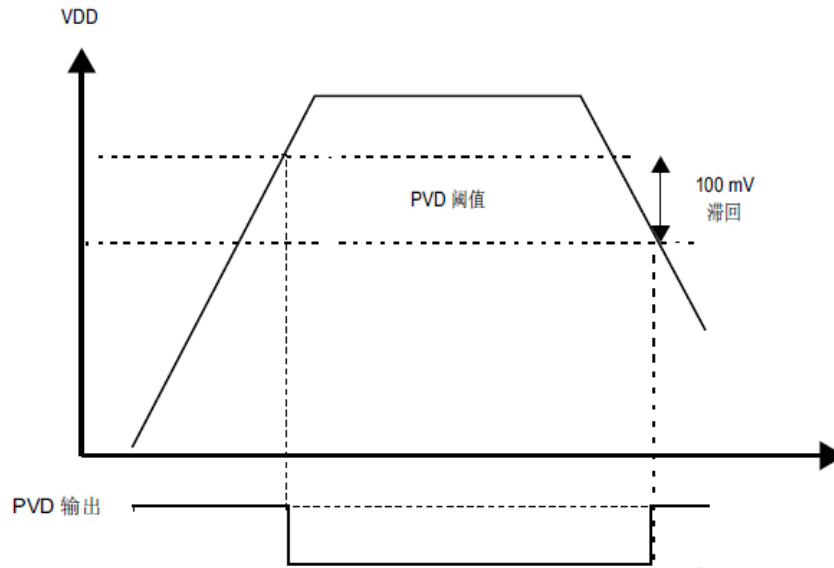


图 1-3 PVD 示意图

1.7.3 低功耗模式

1.7.3.1 低功耗模式转换

默认情况下，系统复位或上电复位后，微控制器进入运行模式。在运行模式下，CPU 通过 HCLK 提供时钟，并执行程序代码。系统提供了多个低功耗模式，可在 CPU 不需要运行时（例如等待外部事件时）节省功耗。由用户根据应用选择具体的低功耗模式，以在低功耗、短启动时间和可用唤醒源之间寻求最佳平衡。

芯片有 5 个低功耗模式：

- Standby 模式
- 睡眠模式 0
- 睡眠模式 1
- 睡眠模式 2
- ShutDown 模式

关于低功耗模式下各模块的详细动作情况请参考。

此外，可通过下列方法之一降低运行模式的功耗：

- 降低系统时钟速度
- 不使用外设时，将对应的外设时钟关闭

进入低功耗模式的转换关系如下图所示：

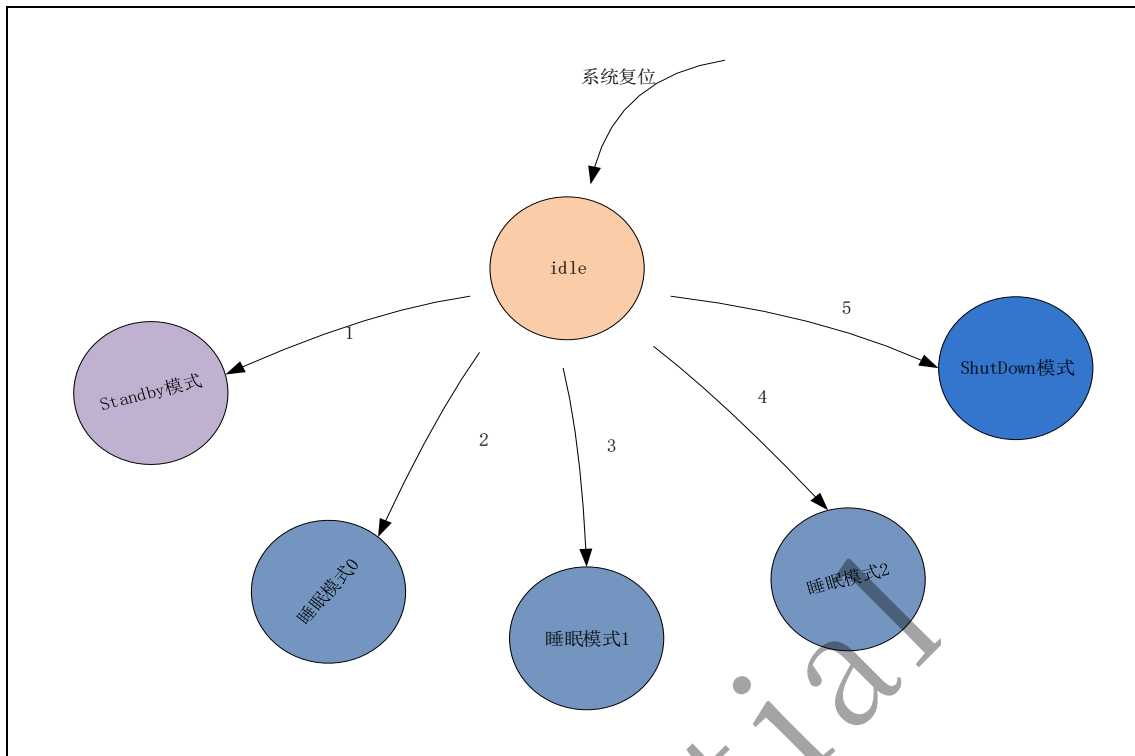


图 1-3 低功耗模式转换图

1. 7. 3. 2 系统时钟速度

在运行模式下，可通过对预分频寄存器编程来降低系统时钟频率。进入睡眠模式之前，也可以使用这些预分频器降低外设频率。也可将系统时钟切换至低速时钟源并关闭高速时钟源来降低功耗。

系统时钟速度的有关详细信息，请参见时钟管理。

1. 7. 3. 3 外设时钟控制

在运行模式下，可通过设置时钟门控来停止各外设和存储器的总线时钟或模块工作时钟以降低功耗。

要进一步降低低功耗模式的功耗，可在执行 WFI 或 WFE 指令之前可通过门控关闭外设时钟。

外设时钟门控配置的有关详细信息，请参见时钟管理。

1. 7. 3. 4 运行模式

- 所有高速时钟源可使能
- 所有外设可使能

1. 7. 3. 5 Standby 模式

- 所有高速时钟源可使能
- CPU 时钟被关断
- 所有外设可使能

1.7.3.6 睡眠模式 0

- 所有高速时钟源禁止
- CPU 内核电源掉电
- 低速时钟源可使能
- SRAM 数据可保持
- 可通过备份域 RTC，IWDG，BLE Timer，复位引脚及唤醒 IO 唤醒 CPU

1.7.3.7 睡眠模式 1

- 所有高速时钟源禁止
- 所有低速时钟源禁止
- SRAM 数据可保持
- 可通过复位引脚和唤醒 IO 唤醒 CPU

1.7.3.8 睡眠模式 2

- 所有高速时钟源禁止
- 低速晶振时钟源禁止
- 低速 RC 时钟源使能
- 主系统电源域掉电
- 备份域寄存器数据保持
- 可通过备份域 RTC，IWDG，复位引脚和唤醒 IO 唤醒 CPU

1.7.3.9 ShutDown 模式

- 主系统电源域掉电
- 备份域寄存器数据保持
- 可通过复位引脚和唤醒 IO 唤醒 CPU

1.8 复位时钟管理

时钟复位管理单元（RCC）主要作用是控制时钟和振荡器。CPU 可对内部各模块的时钟进行使能或关闭。软件可灵活地通过打开或关闭不必要的时钟，来降低系统的总功耗。

1.8.1 系统时钟

- ◆ 支持 5 种不同时钟源用于驱动系统时钟
 - 16MHz 外部高速晶体振荡器（HSE）
 - 24MHz 内部高速 RC 振荡器（HSI）
 - 32.768KHz 外部低速晶体振荡器（LSE）
 - 32.768KHz 内部低速振 RC 荡器（LSI）
 - 内部锁相环倍频时钟（PLL）
- ◆ AHB 外设和 APB 外设可独立预分频

LE5030 还有以下附加的时钟源：

- ◆ 32.768kHz 内部低速 RC 振荡器(LSI)，用于驱动独立看门狗和用于自动从停机或待机模式下唤醒的 RTC 时钟。
- ◆ 用于驱动实时时钟 (RTC)的 32.768 kHz 低速外部晶振振荡器(LSE)。
- ◆ 每种时钟源都可以单独的打开或关断，当它们不用时，可以关断它们来降低功耗。
- ◆ 有多个分频器可用于配置 AHB 和 APB 时钟域，AHB 和 APB 域的最大时钟频率为 64MHz。
- ◆ 所有的外设时钟由其所在的总线时钟 (HCLK 或 PCLK) 驱动，以下几个除外：
 - LSQSPI 控制器由 PLL256M 时钟驱动。
 - BLE MDM 电路由 PLL128M 时钟驱动。
 - PDM 电路由 PLL128M 时钟驱动
 - RTC 时钟来自于 LSE、LSI。
 - IWDG 时钟来自于 LSE、LSI。

1. 8. 1. 1 外部高速晶体振荡器时钟 HSE

高速外部时钟信号 (HSE) 由以下两种时钟源产生：HSE 外部晶体/陶瓷谐振器

为了减少时钟输出的失真和缩短启动稳定时间，晶体/陶瓷谐振器和负载电容器必须尽可能地靠近振荡器引脚。

外部晶体/陶瓷谐振(HSE 晶体)16 MHz 外部振荡器可为系统提供非常精确的主时钟。在时钟控制寄存器(RCC_CK)中的 HSE_RDY 位用来指示高速外部振荡器是否稳定。在启动时，直到这一位被硬件置 1，该时钟才可使用。HSE 时钟可以通过设置时钟控制寄存器(RCC_CK) 中的 HSE_ON 位被启动和关闭。

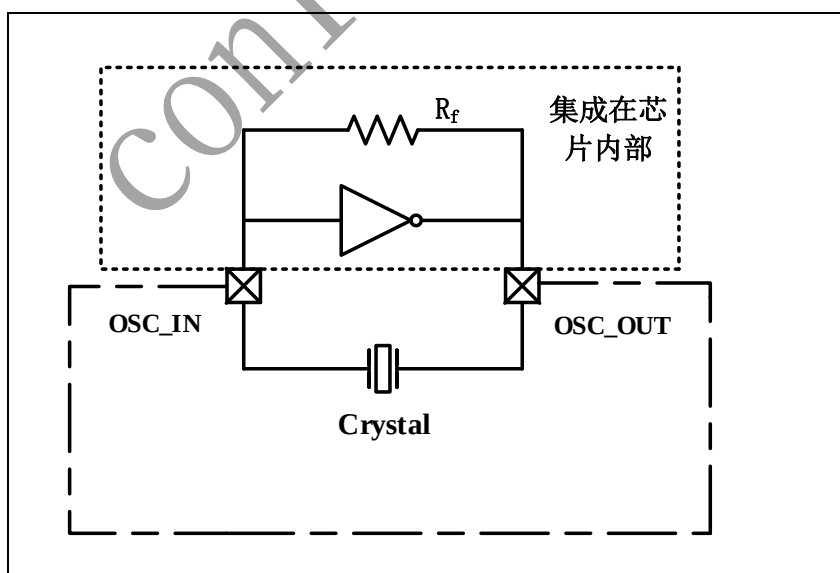


图 1-4 HSE 结构图

1. 8. 1. 2 内部高速 RC 振荡器时钟 HSI

HSI 时钟信号由内部 24MHz/2MHz 的 RC 振荡器产生，可直接作为系统时钟。HSI RC 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。它的启动时间比 HS 晶体振荡器短。然而，即使在校准之后它的时钟频率精度仍较差（相比于晶体振荡器或陶瓷谐振器）。

校准

制造工艺决定了不同芯片的 RC 振荡器频率会不同，这就是为什么每个芯片的 HSI 时钟频率在出厂前已经被校准到 1%(25° C) 的原因。系统复位时，工厂校准值被装载到 HSI 时钟控制寄存器中。

时钟控制寄存器 (RCC_CR) 中的 HIS_RDY 位用来指示 HSI RC 振荡器是否稳定。在时钟启动过程中，直到这一位被硬件置 1，HSI RC 输出时钟才可以被使用。

HSI RC 可由时钟控制寄存器 (RCC_CR) 中的 HIS_ON 位来启动和关闭。

1. 8. 1. 3 外部低速晶体振荡器时钟 LSE

LSE 晶体是一个 32.768kHz 的低速外部晶体或陶瓷谐振器。它为实时时钟或者其他定时功能 提供一个低功耗且精确的时钟源。

晶体振荡器的开和关可用 RCC 控制寄存器(RCC_CR) 中的 LSE_ON 位来控制。RCC 寄存器(RCC_CR)中的 LSE_RDY 位指示 LSE 晶体振荡是否稳定。在该位被硬件置为 1 之前，LSE 的时钟信号都不被使用。

1. 8. 1. 4 内部低速 RC 振荡器时钟 LSI

LSI RC 做为一个低功耗时钟源，它可以在停机和待机模式下保持运行，为独立看门狗和 RTC 提供时钟。LSI 时钟频率大约为 32.768kHz。

LSI RC 振荡器可由时钟控制寄存器(RCC_CR)中的 LSI_ON 位来控制开或关。

在时钟控制/状态寄存器 (RCC_CR) 里的 LSI_RDY 位指示低速内部振荡器是否稳定。在这个位被硬件设置为 1 之前，LSI 时钟都不能被使用。

1. 8. 1. 5 PLL 时钟

内部 PLL 可用 HSE 倍频得到，RCC 时钟控制寄存器 (RCC_CR)。PLL 时钟的开和关有 RCC 时钟控制寄存器（RCC_CR）的 PLL_ON 位来控制。

PLL 的可输出 64MHz 系统时钟，128MHz BLE 时钟，128MHz PDM 时钟和 256MHz LSQSPI 时钟。

1. 8. 1. 6 系统时钟选择

系统时钟可从 HSE,HSI,LSE,LSI 以及 PLL 时钟源中进行选择和切换。可对 RCC 配置寄存器（RCC_CFG）的 SYSCLOCK_SW 位进行配置选择和切换时钟。

1. 8. 2 系统复位

系统复位包括上电复位，掉电复位，外部端口复位，长按键复位，看门狗复位和软件复位。

◆ 备份域支持 POR/PDR

- ◆ 主电源域支持 POR/PDR
- ◆ 支持外部端口复位 MRSTN
- ◆ 支持端口长按键复位 (PB15)
- ◆ 支持看门狗溢出复位 (WWDG/IWDG)
- ◆ 支持 2 种软件复位
 - 复位整个主电源域
 - 复位内核
- ◆ 支持各外设模块的独立软件复位

1. 8. 2. 1 结构框图

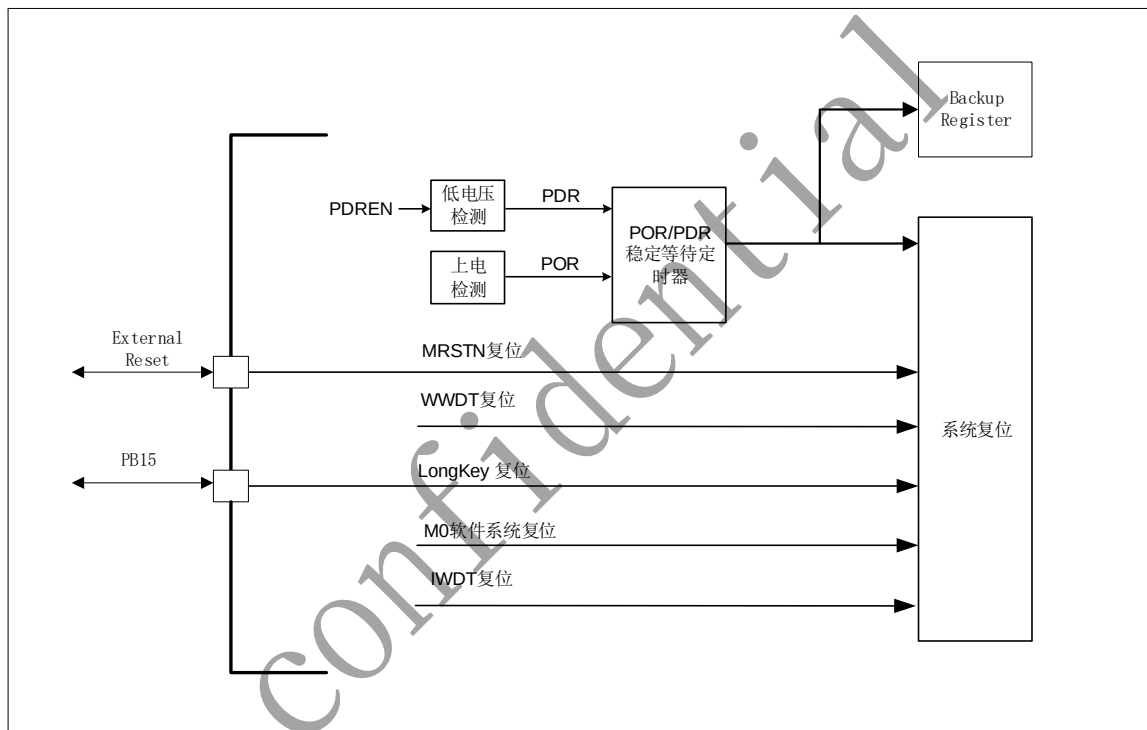


图 1-4 复位结构图

1. 8. 2. 2 上电复位/掉电复位

当 VDD33 低于指定限位电压 V_{POR}/V_{PDR} 阈值时，系统保持复位状态，而无需外部复位电路。

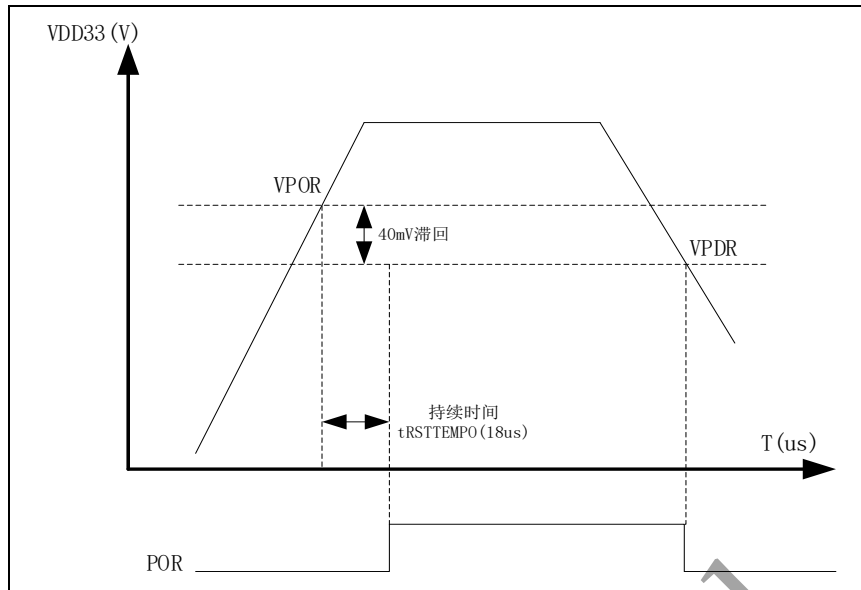


图 1-5 上电复位示意图

1.8.2.3 端口复位

可复位除备份域寄存器外的芯片整体，复位解除后，芯片从 BOOT ROM 启动。

1.8.2.4 看门狗复位

详细描述请参照独立看门狗和窗口看门狗的说明。

可复位除备份域寄存器外的芯片整体，复位解除后，芯片从 BOOT ROM 启动。

1.8.2.5 MCU 软件系统复位(MCURST)

MCU 软件系统复位从内核产生。由应用中断和复位控制寄存器(Application Interrupt and Reset Control Register)的 SYSRESETREQ 位控制，将该位置 1 可对系统复位。

1.8.2.6 MCU 软件内核复位(CORERST)

MCU 软件内核复位由 MCU 内核复位寄存器 (SYSCFG_CORERST) 进行控制，可以复位 MCU 内核（不复位除 MCU 内核外的其他电路）。

1.8.2.7 外设软件复位

对应每个外设分别分配了一个软件复位。

可以通过 AHB 外设复位寄存器 (RCC_AHBRSRST)，APB1 外设复位寄存器 (RCC_APB1RST) 和 APB2 外设复位寄存器 (RCC_APB2RSTR) 对 LE5030 的外设进行独立软件复位。

第2章 直接存储访问控制器（DMA）

1.1 DMA 简介

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

1.2 DMA 主要特性

- ◆ 支持 8 个独立 DMA 通道
- ◆ 每个 DMA 通道都有独立握手信号
- ◆ 每个 DMA 通道的优先级可编程
- ◆ 每个优先级仲裁使用由 DMA 通道号决定的固定优先级
- ◆ 支持多种传输类型：
 - 内存到内存
 - 内存到外设
 - 外设到内存
- ◆ 支持多种 DMA 循环类型
- ◆ 支持多种 DMA 传输数据位宽
- ◆ 每个 DMA 通道都可以访问 primary 和 alternate 通道控制数据结构
- ◆ 所有通道控制数据以小端格式存储在系统内存中
- ◆ 单个 DMA 周期内传输数量可以编程（从 1 到 1024）
- ◆ 传输地址增量可以大于数据宽度
- ◆ 可以指示 AHB 总线上发生的错误

1.3 结构框图

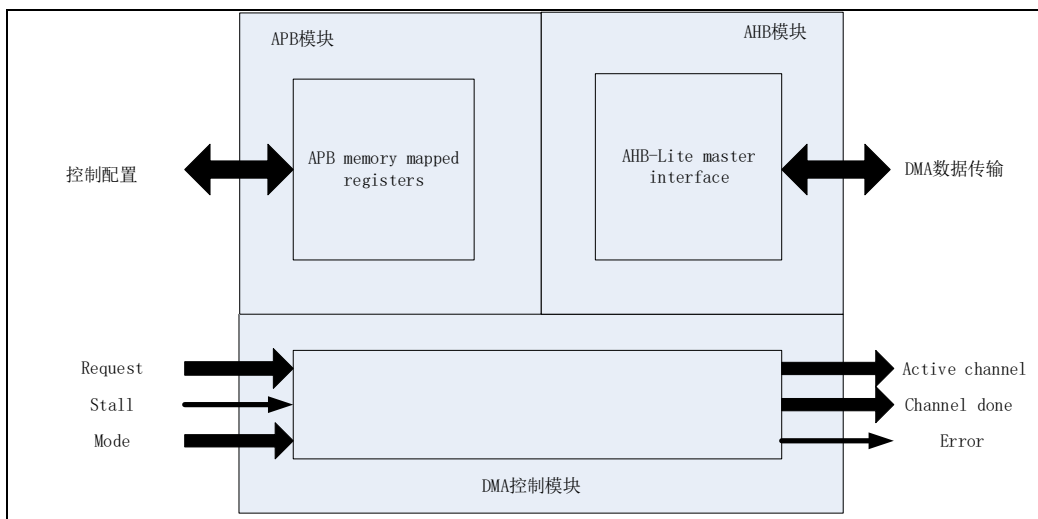


图 2-1 DMA 结构框图

DMA 控制器包含以下主要功能块:

- ◆ APB 控制块, 允许软件配置 DMA 控制器。
- ◆ AHB 控制块, 允许 DMA 读/写访问可用于 DMA 传输的源及目标数据。
- ◆ DMA 控制块, 控制 DMA 的操作, 包括连接外围设备的请求/确认信号。

1.4 功能描述

1.4.1 DMA 控制

- 握手规则
- DMA 信号
- DMA 仲裁率
- 优先级
- DMA 周期类型
- 错误信号

1.4.1.1 握手规则

DMA 握手规则适用于以下条件:

- ◇ 使能通道, 即 CHNL_ENABLE_SET[C]和 MASTER_ENABLE 置 1
- ◇ 不屏蔽 dma_req[C]和 dma_sreq[C]请求输入, 即 CHNL_REQ_MASK_SET[C]清 0

规则	功能描述
1	当 dma_active[C]为低电平, dma_req[C] or dma_sreq[C]维持 1 个或多个 hclk 周期高电平(连续或非连续)可在通道 C 上启动数据传输
2	控制器仅允许单个 dma_active[]为高电平
3	当通道 C 启动传输时, dma_active[C]置为高电平
4	对于不同 DMA 周期类型, 除了外设分散-聚集(scatter-gather)模式外, 在控制器完成 2^R 或 $n\_minus\_1^a$ (两者中, 选择较小值)的传输前, dma_active[C]保持高电平。在外设分散-聚集模式下, 控制器先使用主要数据结构(primary data structure)进行 2^R 次传输, 不进行仲裁, 接着使用交替数据结构(alternate data structure)进行 2^R 或 $n\_minus\_1^a$ 次传输(两者中, 选择较小值)。当完成交替数据结构的传输后, dma_active[C]将变成低电平。
5	在设置 dma_active[C]或 dma_active[]为高电平前, dma_active[C]必须设置为低电平至少 1 个 hclk 周期。
6	对于使能的通道, 仅单个 dma_done[]可置为高电平
7	当 dma_active[C]或 dma_stall 为高电平时, 若 dma_req[C]为高电平, 则 dma_req[C]在上一个周期为低电平的情况下, 控制器才能检测到请求信号。
8	若某一通道的 cycle_ctrl 设置为 3' b100, 3' b101, 3' b110, 或 3' b111, 则 dma_done[C] 将不会被置为高电平。
9	当某一通道上的数据传输已全部完成, 并且 cycle_ctrl 将 dma_done[]置高, 则在 dma_active[]的下降沿会产生以下动作:

	- 若 dma_stall 为低电平, dma_done[]将置高 1 个 hclk 周期。 - 若 dma_stall 为高电平, 将延迟控制器。当 dma_stall 变为低电平, 则控制器将置高 dma_done[]1 个 hclk 周期。
10	dma_waitonreq[C]的状态仅能在 C 通道禁止的情况下改变。
11	当 dma_waitonreq[C]为高电平, 则在下列条件发生前, dma_active[C]会一直保持高电平 - 控制器完成了 2^R 或 $n\_minus\_1^a$ 次传输 - dma_req[C]为低电平 - dma_sreq[C]为低电平
12	若在 hclk 周期上, 设置 dma_active[C]为低电平前, 设置 dma_stall 为高电平, 则: - 在下一个 hclk 周期, 控制器设置 dma_active 为低电平 - 在 dma_stall 设置为低电平前, 通道 C 才会完成传输
13	当 dma_waitonreq[C]为低电平, 控制器将无视 dma_sreq[C]。
14	当 CHNL_USEBURST_SET[C]为 1^b , 控制器将无视 dma_sreq[C]。
15	除了外设分散-聚集模式, 对于其他 DMA 周期类型, 当完成 2^R 次传输时, 若剩余的传输次数小于 2^R , CHNL_USEBURST_SET[C]的读取值将被置为 0。 在外设分散-聚集模式下, 若剩余的使用交替数据结构的传输次数小于 2^R , 则 CHNL_USEBURST_SET[C]的读取值将被置为 0
16	除了外设分散-聚集模式, 对于其他 DMA 周期类型, 当 dma_sreq[C] 和 dma_waitonreq[C]为高电平, 且 dma_req[C]为低电平, 在 dma_active[C]在 hclk 周期上变为高电平之前, 控制器将会进行一次 DMA 传输, 在外设分散-聚集模式下, 当 dma_sreq[C] 和 dma_waitonreq[C] 为高电平, 且 dma_req[C] 为低电平, 在 dma_active[C]在 hclk 周期上变为高电平之前, 控制器会先完成 2^R 次主要数据结构的传输, 且不进行仲裁, 接着再完成一次交替数据结构的 DMA 传输。
17	除了外设分散-聚集模式, 在 DMA 其他周期模式下, 当 dma_req[C]和 dma_sreq[C]为高电平, 在 dma_active[C]在 hclk 周期上变为高电平前, 先发出 dma_req[C]请求, 接着控制器进行 2^R 或 $n\_minus\_1^a$ 次传输。 在外设分散-聚集模式下, 当 dma_sreq[C] 和 dma_sreq [C] 为高电平, 在 dma_active[C]在 hclk 周期上变为高电平之前, 先发出 dma_req[C]请求, 控制器进行 2^R 次主要数据结构的传输, 且不进行仲裁, 接着再完成 2^R 或 $n\_minus\_1^a$ 次(两者中, 选择较小值)交替数据结构的 DMA 传输。
18	当 CHNL_REQ_MASK_SET[C]为 1, 控制器将无视 dma_req[C]和 dma_sreq[C]的请求。

表 2-1 通道使能且未屏蔽请求时的规则

- a、 n_minus_1 在 channel_cfg 的地址中。
- b、 需谨慎设置该比特位。当 n_minus_1 小于 2^R , 控制器不会将 CHNL_USEBURST_SET 清零, 因此 dma_req[C]请求将被屏蔽。若外设不将 dma_req[C]置高, 控制器将不会进行剩余的传输。请参考 Channel useburst set。

当通道禁止时, DMA 控制器的握手规则如下表

规则	功能描述
19	当 dma_req[C]为高电平, 则 dma_done[C]被置为高电平。在这种情况下, 即便通道在禁止 的状态, 控制器也会向主机的处理器发送请求。
20	当 dma_sreq[C]为高电平, 若 dma_waitonreq[C]为高电平且 CHNL_USEBURST_SET[C]为 0, 则 dma_done[C]置为高电平。在这种情况下, 即便通道在禁止的状态, 控制器也会向 主机的处理器发送请求。
21	dma_active[C]始终为低电平。

表 2-2 通道禁止时的规则

1. 4. 1. 2 DMA 信号

- 脉冲请求
- 电平请求
- 完成信号
- 等待请求信号

脉冲请求

下图为当外设使用脉冲信号时，DMA 的请求时序图

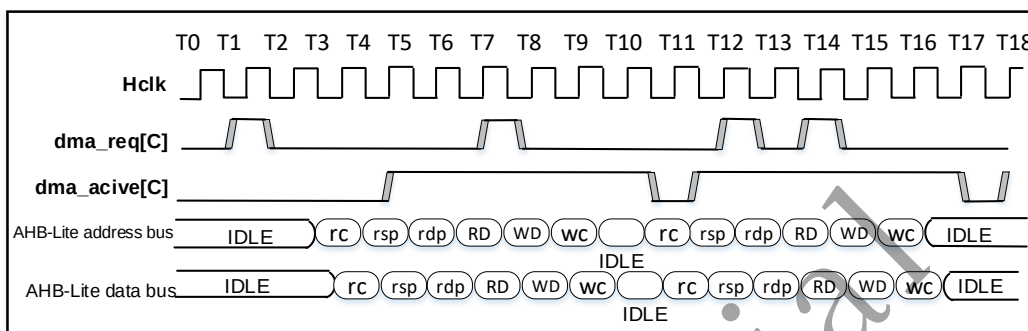


图 2-1 DMA 信号（当外设使用脉冲请求）

T1 若 CHNL_REQ_MASK_SET[C]为 0 (参考规则 18), 控制器会在通道 C 上检测到一个请求(参考规则 1)。

T4 控制器将 dma_active[C]置为有效(参考规则 2 和规则 3)并在通道 C 上启动 DMA 传输。

T4-T7 读取数据结构:

- ◇ rc: 读取通道配置, channel_cfg
- ◇ rsp: 读取源数据的结束指针, src_data_end_ptr
- ◇ rdp: 读取目标数据结束指针, dst_data_end_ptr

T7 若 dma_active[C]为高电平且 CHNL_REQ_MASK_SET[C]为 0(参考规则 18), 通道 C 上会检测到一个请求信号(参考规则 7)。该请求将存在于下一个仲裁过程中。

T7-T9 在通道 C 上进行 DMA 传输

- ◇ RD: 读取数据
- ◇ WD: 写入数据

T9-T10 写通道配置寄存器 channel_cfg

- ◇ wc: 写通道配置, channel_cfg

T10 置 dma_active[C]为无效, 表明 DMA 传输已完成(参考规则 4)

T10-T11 dma_active[C]需维持低电平至少一个 hclk 周期(参考规则 5)

T11 若通道 C 上的请求为最高优先级, 则由于 T7 请求, dma_active[C]将置为有效(参考规则 2 和规则 3)。

T12 若 dma_active[C]为高电平且 CHNL_REQ_MASK_SET[C]为 0(参考规则

18), 通道 C 上将检测到一个请求信号(参考规则 7)。该请求将存在于下一个仲裁过程中。

T14 因 T12 请求挂起, 控制器将无视通道 C 上的请求。

T17 置 dma_active[C]为无效, 表明 DMA 传输已完成(参考规则 4)

T17-T18 dma_active[C]需维持低电平至少一个 hclk 周期(参考规则 5)。

T18 若通道 C 上的请求为最高优先级, 则由于 T12 请求, dma_active[C]将置为有效(参考规则 2 和规则 3)。

电平请求

下图为当外设使用电平信号时, DMA 的请求时序图

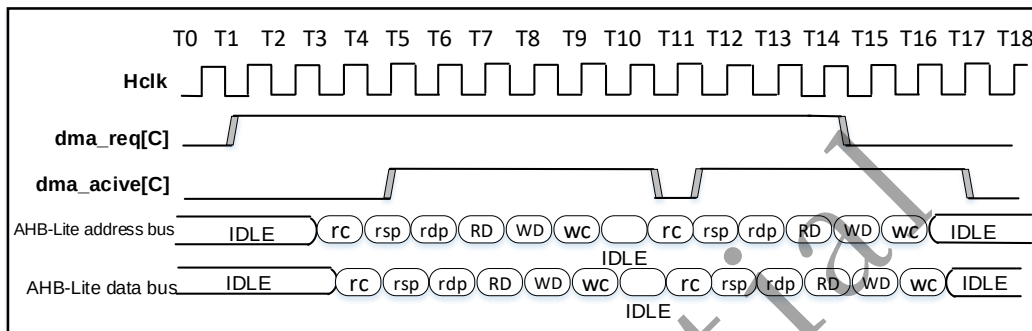


图 2-2 DMA 信号 (当外设使用电平请求)

T1 若 CHNL_REQ_MASK_SET[C]为 0(参考规则 18), 控制器会在通道 C 上检测到一个请求(参考规则 1)。

T4 控制器将 dma_active[C]置为有效(参考规则 2 和规则 3)并在通道 C 上启动 DMA 传输。

T4-T7 读取数据结构:

- ◇ rc: 读取通道配置, channel_cfg
- ◇ rsp: 读取源数据的结束指针, src_data_end_ptr
- ◇ rdp: 读取目标数据结束指针, dst_data_end_ptr

T7-T9 在通道 C 上进行 DMA 传输

- ◇ RD: 读取数据
- ◇ WD: 写入数据

T9-T10 写通道配置寄存器 channel_cfg

- ◇ wc: 写通道配置, channel_cfg

T10 置 dma_active[C]为无效, 表明 DMA 传输已完成(参考规则 4)

如果 chnl_req_mask_set [C]为 0(参见规则 18), 则控制器检测到通道 C 上的一个请求(参见规则 1)。

T10-T11 dma_active[C]需维持低电平至少一个 hclk 周期(参考规则 5)

T11 若通道 C 上的请求为最高优先级, 则由于 T7 请求, dma_active[C]将置为有效(参考规则 2 和规则 3)。

T11-T14 控制器读取数据结构

T14-T16 控制器在通道 C 上进行 DMA 传输。

T15-T16 外设确认 DMA 传输已开始并且置 dma_req[C]为无效。

T16-T17 控制器写通道配置 channel_cfg

T17 控制器置 dma_active[C]为无效, 表明 DMA 传输已完成(参考规则 4)。

使用电平请求信号时, 如果外设不要求额外的 DMA 传输, 但因时间太短不能取消请求, 则 dma_stall 必须置为有效, 可防止控制器结束当前传输(参考规则 7)。

完成信号

下图为当以下条件满足的情况下, dma_done[]信号图。

- dma_stall 和 dma_waitonreq[]为低电平
- dma_stall 为高电平
- dma_waitonreq[]为高电平

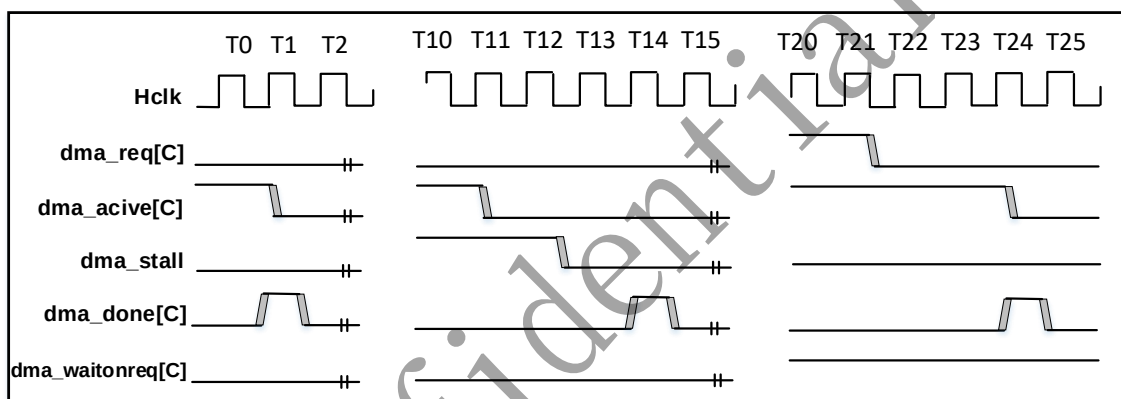


图 2-3 DMA 完成信号

T1 置 dma_active[C]为无效, 表明 DMA 传输已完成(参考规则 4)。

T1-T2 完成 DMA 周期, 且若 cycle_ctrl[2]为 0, dma_done[C]将保持有效一个 hclk 周期(参考规则 8 和规则 9)。对于其他使能的通道, dma_done[]为低电平(参考规则 6)。

T11 置 dma_active[C]为无效, 表明 DMA 传输已完成(参考规则 4)。

注意:

由于 dma_stall 在上一个 hclk 周期上为高电平, 因此 dma_done[C]不能置为有效(参考规则 9 和规则 12)。

T12-T13 外设将 dma_stal 置为无效。

T14-T15 完成 DMA 周期, 且若 cycle_ctrl[2]为 0, dma_done[C]将保持有效一个 hclk 周期(参考规则 8 和规则 9)。对于其他使能的通道, dma_done[]为低电平(参考规则 6)。

T20 当控制器已完成 DMA 传输, 但由于 dma_waitonreq[C]为高电平, 因此只有等 dma_req[C]变为低电平后, 才能将 dma_active[C] (参考规则 11)和 dma_done[C] 分别置为无效和有效(参考规则 9)。

T21-T22 外设将 dma_req 置为无效。

T24 置 dma_active[C]为无效, 表明 DMA 传输已完成(参考规则 4)。

T24-T25 完成 DMA 周期, 且若 cycle_ctrl[2]为 0, dma_done[C]将保持有效一个 hclk 周期 (参考规则 8 和规则 9)。对于其他使能的通道, dma_done[]为低电平(参考规则 6)。

等待请求信号

下图分别举例说明如何利用等待请求信号进行 2^R 次和单次传输。

- 仅使用 dma_waitonreq 时的 DMA 信号
- 使用 dma_waitonreq 和 dma_sreq 时的 DMA 信号

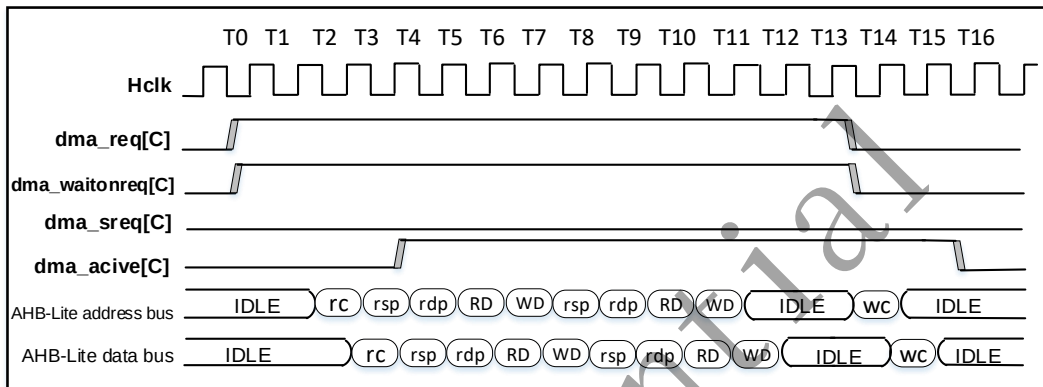


图 2-4 DMA 使用 dma_waitonreq 时序图

T0-T16 外设必须保持 dma_waitonreq[C]的状态不变 (参考规则 10)。

T0-T1 若 CHNL_REQ_MASK_SET[C]为 0(参考规则 18), 通道 C 上会检测到请求信号(参考规则 1)。

T3-T4 dma_req[]和 dma_sreq[]保持为高电平。控制器会回应 dma_req[]请求, 但忽略 dma_sreq[]请求(参考规则 16 和规则 17)。

T4 dma_active[C]置为有效(参考规则 2 和规则 3)且启动通道 C 上的 DMA 传输。

T4-T7 读取数据结构:

- ◇ rc: 读取通道配置, channel_cfg
- ◇ rsp: 读取源数据的结束指针, src_data_end_ptr
- ◇ rdp: 读取目标数据结束指针, dst_data_end_ptr

T7-T9 在通道 C 上进行 DMA 传输

- ◇ RD: 读取数据
- ◇ WD: 写入数据

T9-T11 控制器读取两个结束指针地址, rsp 和 rdp。

T11-T13 控制器通道 C 上进行 DMA 传输。在该例子中, $R = 1$, 因此进行 $2^1 = 2$ 次 DMA 传输。

T13-T14 dma_req[C]和 dma_sreq[C]置为无效。

T15-T16 控制器写通道配置

- ◇ wc: 写通道配置, channel_cfg

T16 置 dma_active[C]为无效, 表明 DMA 传输已完成(参考规则 11)。如果剩余的传输次数小于 2^R (参考规则 15), 将 CHNL_USEBURST_SET[C]的读取值写 0。

下图为当 dma_waitonreq[]为高电平, 进行单次 DMA 传输时的 DMA 信号图。

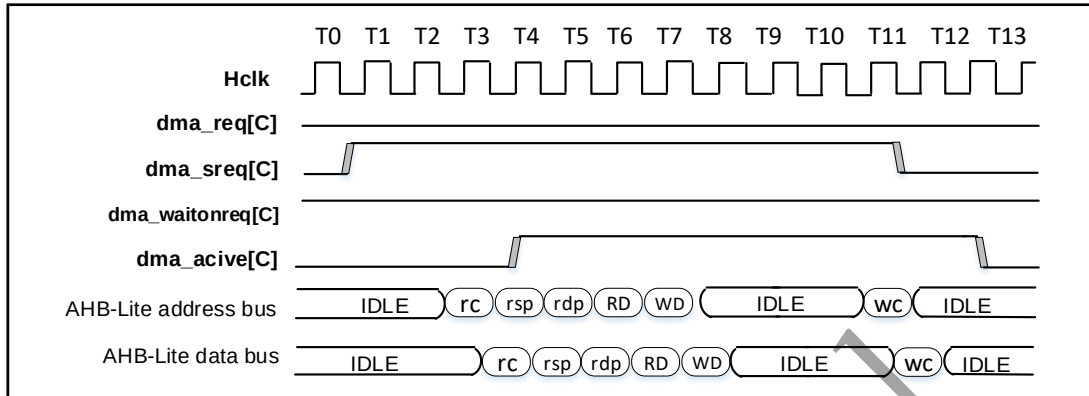


图 2-5 DMA 信号 (当外设使用 dma_waitonreq 和 dma_sreq 请求)

T0-T13 外设必须保持 dma_waitonreq[C]的状态不变(参考规则 10)。

T0-T1 若 chnl_useburst_set[C]为 0(参考规则 13 和规则 14), 通道 C 上会检测到请求信号(参考规则 1)。

T3-T4 控制器对 dma_sreq[]请求作出回应(参考规则 16)

T4 dma_active[C]置为有效(参考规则 2 和规则 3)且启动通道 C 上的 DMA 传输。

T4-T7 读取数据结构:

- ◇ rc: 读取通道配置, channel_cfg
- ◇ rsp: 读取源数据的结束指针, src_data_end_ptr
- ◇ rdp: 读取目标数据结束指针, dst_data_end_ptr

T7-T9 在通道 C 上进行 DMA 传输

- ◇ RD: 读取数据
- ◇ WD: 写入数据

$R = 0$, 只进行 $2^0 = 1$ 次 DMA 传输。

T10-T11 外设置 dma_sreq[C]为无效。

T12-T13 控制器写通道配置

- ◇ wc: 写通道配置, channel_cfg。

T13 控制器置 dma_active[C]为无效, 表明 DMA 传输已完成。

DMA 仲裁率

在 DMA 传输过程中, 用户可配置控制器何时进行仲裁。该动作可缩短响应高优先通道的延时。

控制器提供 4 个比特位，R_power，用来配置在重新进行仲裁前，AHB 总线传输的次数。其中 R 为 2 的次方， 2^R ，决定了仲裁率。当 R = 4，则仲裁率为 $2^4 = 16$ ，即每进行 16 次 DMA 传输就进行一次仲裁。

R_power	x 次 DMA 传输后进行仲裁
b0000	x=1
b0001	x=2
b0010	x=4
b0011	x=8
b0100	x=16
b0101	x=32
b0110	x=64
b0111	x=128
b1000	x=256
b1001	x=512
b1010-b1111	x=1024

表 2-3 仲裁设置

注意：

不要给优先级低的通道赋一个较大的 R_power 值，这样会导致在重新进行仲裁前，控制器不会响应优先级高的请求。

当 $N > 2^R$ 且 N 不是 2^R 的整数倍，控制器会依次先完成 2^R 次传输直到剩余的次数 N 小于 2^R 。剩余的次数 N 会在 DMA 周期的最后完成。

用户可将 R_power 的值保存在通道控制数据结构中。关于 R_power 在数据结构中的具体位置，请参考控制数据配置。

优先级

当控制器进行仲裁的时候，下一个进行仲裁的通道可由以下信息决定：

- 通道数
- 通道优先级

每个通道都可以通过 CHNL_PRIORITY_SET 寄存器设置为默认优先级或者高优先级。

通道 0 的优先级最高。随着通道数增高，优先级随之降低。

通道数	优先级设置	优先级（由高到低）
0	高	DMA 通道最高优先级
1	高	-
2	高	-
-	高	-
-	高	-
-	高	-
30	高	-

31	高	-
0	默认	-
1	默认	-
2	默认	-
-	默认	-
-	默认	-
-	默认	-
30	默认	-
31	默认	DMA 通道最低优先级

表 2-4 DMA 通道优先级

当完成一个 DMA 传输，控制器将轮询所有的 DMA 通道。下方为轮询流程图。

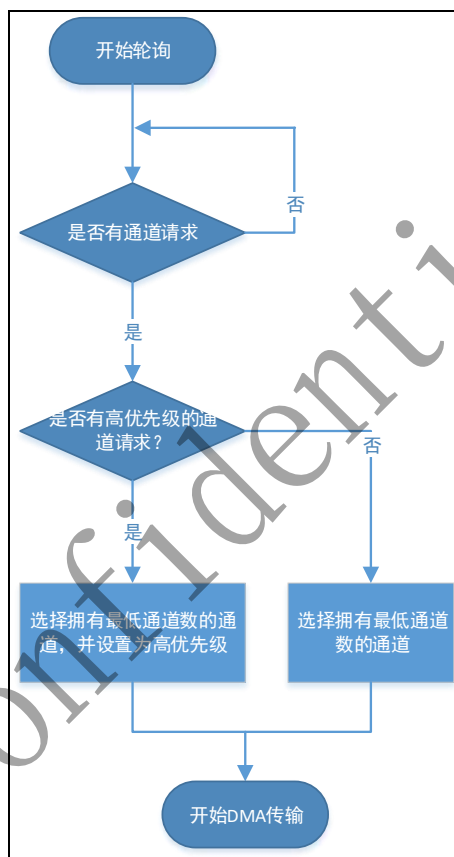


图 2-6 轮询流程图

DMA 周期类型

控制器可通过 cycle_ctrl 来选择 DMA 周期的类型，请参考下表。

cycle_ctrl	功能描述
b000	通道控制数据结构无效
b001	基础 DMA 传输
b010	自动请求
b011	乒乓
b100	存储器分散-聚集(主要数据结构)
b101	存储器分散聚集(交替数据结构)
b110	外设分散-聚集(主要数据结构)

b111	外设分散聚集(交替数据结构)

表 2-5 DMA 周期类型

注意：

cycle_ctrl 位于 channel_cfg 存储地址。

在所有的周期模式下，控制器在完成 2^R 次传输后进行仲裁。若一个低优先级通道被赋予了一个较大的 2^R 值，则到该通道完成 DMA 传输之前，所有其它的通道都不会进行 DMA 传输。因此，用户需谨慎设置 R_power 的值，避免增加高优先级通道的延时。

无效模式

当完成一个 DMA 传输后，控制器会将周期类型设置为无效，避免控制器发送同一个 DMA 周期。

基础模式

1. 在此模式下，用户可以设置控制器使用主要数据结构或者交替数据结构。当通道使能后且控制器接收到请求信号，接下来的流程如下：
2. 控制器展开 2^R 次传输。如果剩余的传输次数为 0，则继续步骤 3。

控制器仲裁如下：

- 若高优先级通道有请求信号，则控制器先响应该通道
 - 若外设或者软件发出请求信号，则继续进行步骤 1
3. 控制器将 dma_done[C]置高一个 hclk 周期，向主控处理器表明该 DMA 周期已完成。

自动请求

在自动请求模式下，仅需要收到一次请求信号就可以完成整个 DMA 周期。无须大幅度增加回应高优先级请求的延时，也无须多次向处理器或者外设发出请求，就可以完成大数据的传输。

用户可以配置控制器使用主要数据结构或者交替数据结构。当通道使能后且控制器接收到该通道上的请求，则流程如下：

1. 控制器展开 2^R 次传输。如果剩余的传输次数为 0，则继续步骤 3。
2. 控制器进行仲裁。当通道 C 的优先级最高，则继续进行步骤 1。
3. 控制器将 dma_done[C]置高一个 hclk 周期，向主控处理器表明该 DMA 周期已完成。

乒乓模式

在乒乓模式下，控制器先使用其中一种数据结构完成一个 DMA 周期，接着再使用另外一种数据结构完成一次 DMA 周期。控制器将会继续转换这两种数据结构直到读到无效的数据结构或者通道被主控处理器禁止。

下图显示为在乒乓模式下的 DMA 传输。

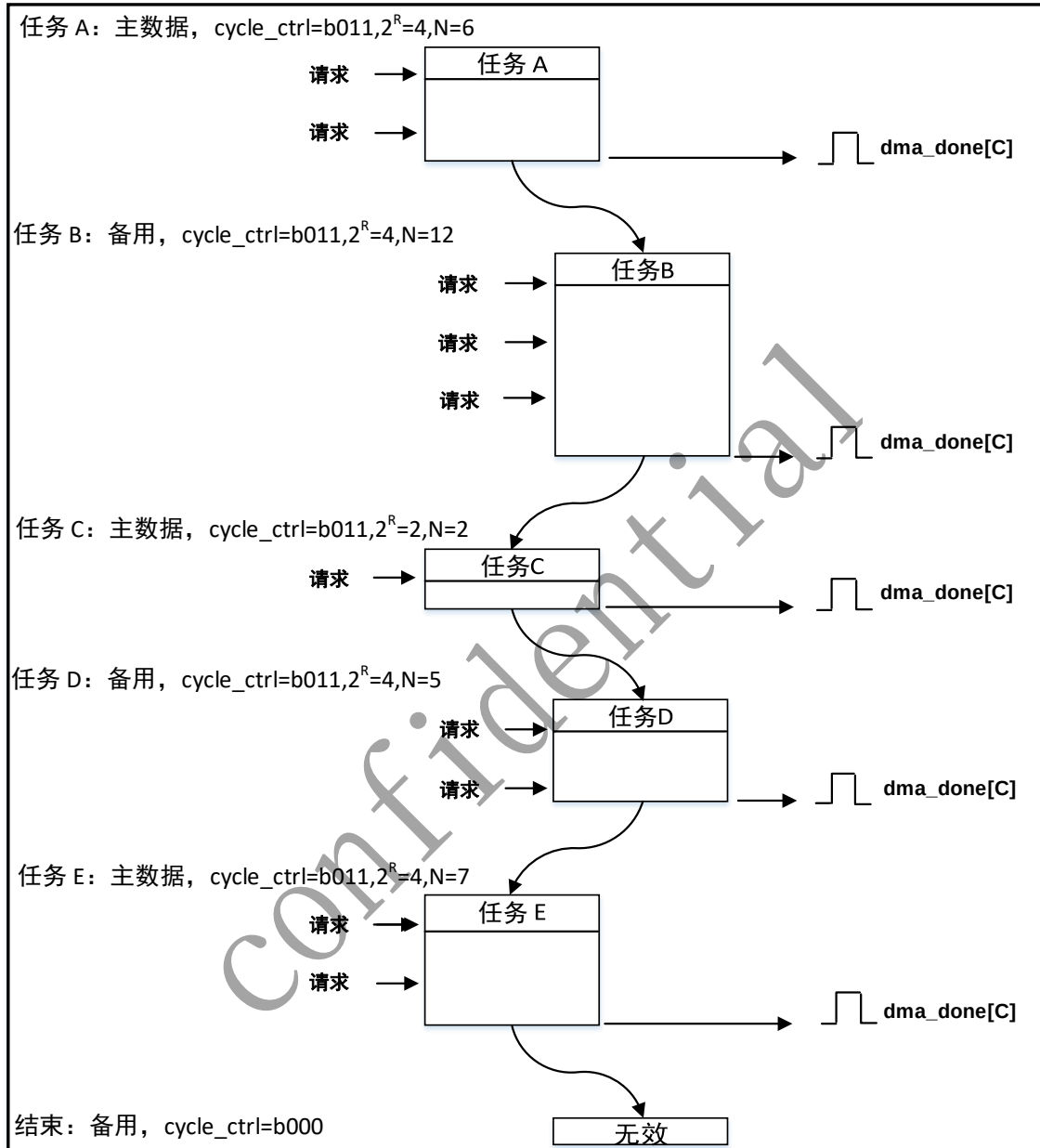


图 2-7 乒乓示例

任务 A

1. 主控处理器配置任务 A 为主要数据结构。
2. 主控处理器配置任务 B 为交替数据结构。当任务 A 完成时，控制器会立刻转换去任务 B，前提是没有高优先级通道提出请求。
3. 控制器接收到一个请求并完成 4 次 DMA 传输。

4. 控制器进行仲裁。当控制器接收到该通道上的请求，如果该通道具有最高优先级，则继续流程。
5. 控制器进行剩余的 2 次 DMA 传输。
6. 控制器将 dma_done[C]置高一个 hclk 周期，并进入仲裁流程。

当任务 A 完成后，主控处理器可配置任务 C 为主要数据结构。在任务 B 完成之后，控制器可立即转换去任务 C，前提是没有高优先级通道提出请求。

当控制器接收到该通道上新的请求且拥有最高优先级，则任务 B 开始执行：

任务 B

7. 控制器进行 4 次 DMA 传输。
8. 控制器进行仲裁。当控制器接收到该通道上的请求，如果该通道具有最高优先级，则继续流程。
9. 控制器进行 4 次 DMA 传输。
10. 控制器进行仲裁。当控制器接收到该通道上的请求，如果该通道具有最高优先级，则继续流程。
11. 控制器进行剩余的 4 次 DMA 传输。
12. 控制器将 dma_done[C]置高一个 hclk 周期，并进入仲裁流程。

当任务 B 完成后，主控处理器可将任务 D 配置为交替数据结构。

当控制器接收到该通道上新的请求且拥有最高优先级，则任务 C 开始执行：

任务 C

13. 控制器进行 2 次 DMA 传输。
14. 控制器将 dma_done[C]置高一个 hclk 周期，并进入仲裁流程。

当任务 C 完成后，主控处理器可将任务 E 配置为主要数据结构。

当控制器接收到该通道上新的请求且拥有最高优先级，则任务 D 开始执行：

任务 D

15. 控制器进行 4 次 DMA 传输。
16. 控制器进行仲裁。当控制器接收到该通道上的请求，如果该通道具有最高优先级，则继续流程。
17. 控制器展开剩余的 DMA 传输。
18. 控制器将 dma_done[C]置高一个 hclk 周期，并进入仲裁流程。

当控制器接收到该通道上新的请求且拥有最高优先级，则任务 E 开始执行：

任务 E

19. 控制器进行 4 次 DMA 传输。
20. 控制器进行仲裁。当控制器接收到该通道上的请求，如果该通道具有最高优先级，则继续流程。

21. 控制器进行剩余的 3 次 DMA 传输。

22. 控制器将 dma_done[C]置高一个 hclk 周期，并进入仲裁流程。

若控制器接收到该通道上的新请求且拥有最高优先级，则控制器会企图开始下一个任务。但由于主控处理器尚未配置交替数据结构，且当任务 D 完成的时候 cycle_ctrl 被设置为 b000，则会终止乒乓 DMA 传输。

注意：

用户可通过设置 cycle_ctrl 为 b001 将任务 E 配置为基础 DMA 周期，以用来终止乒乓 DMA 周期。

存储器分散-聚集

在存储器分散-聚集模式下，控制器先接收到一个初始请求，接着采用主要数据结构进行 4 次 DMA 传输。当传输完成时，会以交替数据结构开始一个新的 DMA 周期。当该周期完成时，控制器将会采用主要数据结构开始新一轮 4 次 DMA 传输。若发生以下任意一种情况，则控制器将停止主要数据结构和交替数据结构的转换：

- 主控处理器将交替数据结构配置为一个基础周期
- 读取到一个无效数据结构

注意：

当完成 N 次主要数据结构的传输后，可设置 cycle_ctrl 为 b000 使其变为无效的数据结构。

当完成一个基础周期的分散-聚集模式传输，控制器仅将 dma_done[C]置为有效。

在分散-聚集模式下，控制器利用主要数据结构来编程交替数据结构。下表主要数据结构的 channel_cfg 配置，分为固定值配置和用户配置。

Bit	Field	Value	功能描述
固定值：			
[31:30]	dst_inc	b10	配置控制器使用字作为地址增量
[29:28]	dst_size	b10	配置控制器使用字传输
[27:26]	src_inc	b10	配置控制器使用字作为地址增量
[25:24]	src_size	b10	配置控制器使用字传输
[17:14]	R_power	b0010	配置控制器进行 4 次 DMA 传输
[3]	next_useburst	0	当配置为存储器分散-聚集模式时，该位必须设置为 0
[2:0]	cycle_ctrl	b100	配置控制器进行分散-聚集 DMA 周期
用户定义：			
[23:21]	dst_prot_ctrl	-	当写入目标数据后，配置 HPROT 的状态
[20:18]	src_prot_ctrl	-	当写入源数据后，配置 HPROT 的状态
[13:4]	n_minus_1	N ^a	配置控制器进行 N 次 DMA 传输，其中 N 为 4 的倍数

表 2-6 存储器分散-聚集数据结构

- a. 由于 R_power 设计为 4，因此 N 必须设置为 4 的倍数。N/4 的值为配置交替数据结构的次

数。

下图为存储器分散-聚集模式的示例：

初始化：

1. 配置主要数据结构来使能复制 A, B, C 的操作：cycle_ctrl = b100, $2^R = 4, N = 16$
2. 利用下表中的结构，将主要源数据写入存储器。

	src_data_end_ptr	dst_data_end_ptr	channel_cfg	Unused
任务 A 数据	0x0A000000	0x0AE00000	Cycle_ctrl = b101, $2^R = 4, N = 3$	0xFFFFFFFF
任务 B 数据	0x0B000000	0x0BE00000	Cycle_ctrl = b101, $2^R = 2, N = 8$	0xFFFFFFFF
任务 C 数据	0x0C000000	0x0CE00000	Cycle_ctrl = b101, $2^R = 8, N = 5$	0xFFFFFFFF
任务 D 数据	0x0D000000	0x0DE00000	Cycle_ctrl = b001, $2^R = 4, N = 4$	0xFFFFFFFF

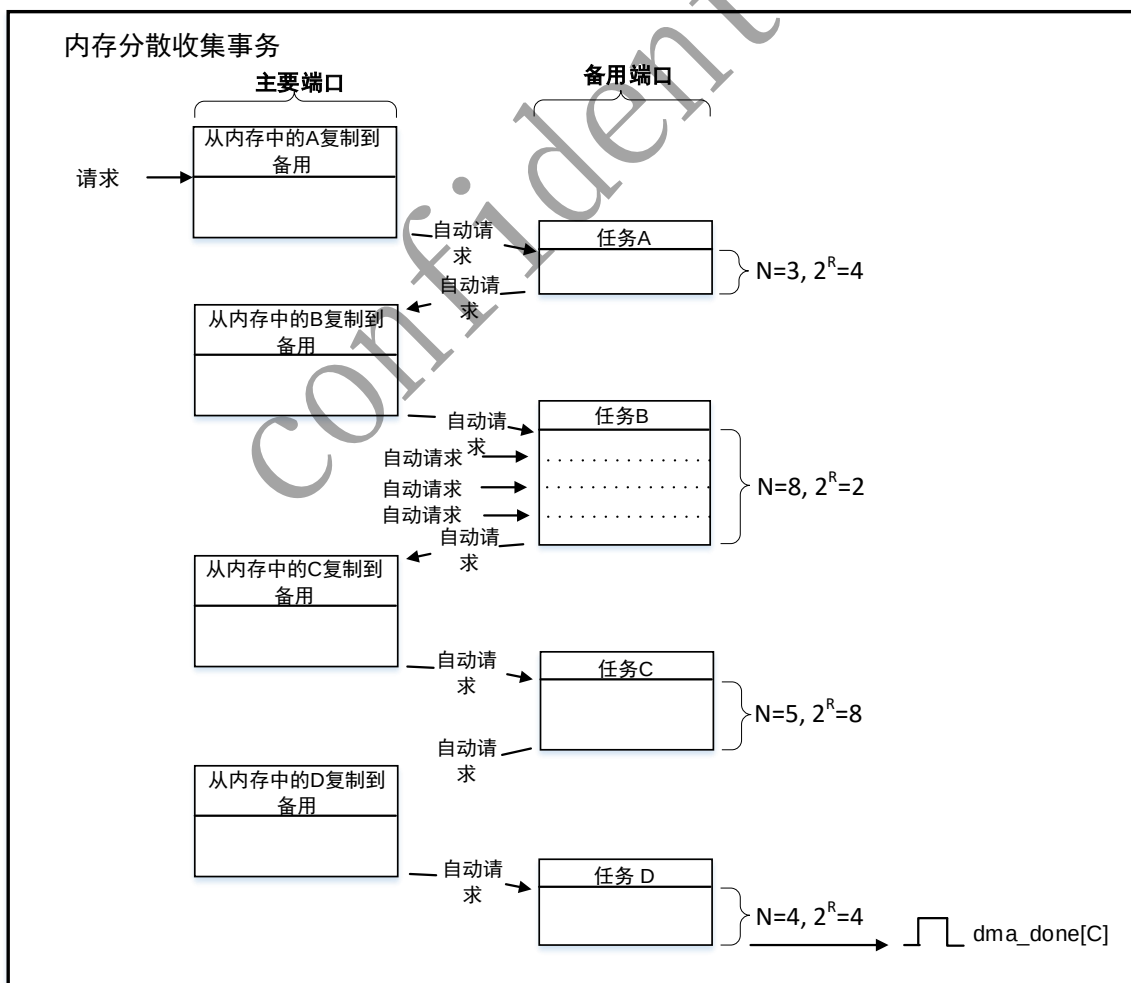


图 2-8 存储器分散-聚集示例

初始化

1. 主控处理器通过设置 `cycle_ctrl` 为 `b100`，使主要数据结构运行于存储器分散-聚集模式。由于单个通道的数据结构包含 4 个字，所以 2^R 必须设置为 4。在该示例中，有 4 个任务，因此 `N` 设为 16。
2. 主控处理器将任务 A, B, C 和 D 的数据结构写入由 `src_data_end_ptr` 指定的存储器地址中。
3. 主控处理器使能该通道。

当控制器接收到 `dma_req[]` 请求或者来自主控处理器的手动请求，则存储器分散-聚集传输开始执行，流程如下：

Primary, 复制 A

1. 在接收到请求后，控制器进行 4 次 DMA 传输，并且任务 A 写为交替数据结构。
2. 控制器生成一个自动请求接着进行仲裁。

任务 A

3. 控制器进行任务 A。当任务 A 完成，控制器生成一个自动请求，接着进行仲裁。

Primary, 复制 B

4. 控制器进行 4 次 DMA 传输，且任务 B 写为交替数据结构。
5. 控制器生成一个自动请求接着进行仲裁。

任务 B

6. 控制器进行任务 B。当任务 B 完成，控制器生成一个自动请求，接着进行仲裁。

Primary, 复制 C

7. 控制器进行 4 次 DMA 传输，且任务 C 写为交替数据结构。
8. 控制器生成一个自动请求接着进行仲裁。

任务 C

9. 控制器进行任务 C。当任务 C 完成，控制器生成一个自动请求，接着进行仲裁。

Primary, 复制 D

10. 控制器进行 4 次 DMA 传输，且任务 D 写为交替数据结构。
11. 控制器设置主要数据结构的 `cycle_ctrl` 为 `b000`，表明该数据结构为无效。
12. 控制器生成一个自动请求接着进行仲裁。

任务 D

13. 控制器使用基础 DMA 周期执行任务 D。
14. 控制器将 `dma_done[C]` 置高一个 `hclk` 周期，并进入仲裁流程。

外设分散-聚集

在外设分散-聚集模式下，控制器接收到一个来自外设的初始请求，接着使用主要数据结构执行 4 次 DMA 传输，然后立即使用交替数据结构启动新的 DMA 周期，无须重新仲裁也无须将 dma_active[C]置为低电平。

注意：

仅在该状况下，当完成主要数据结构的传输后，控制器无须进入仲裁流程。

在该 DMA 周期完成后，控制器重新仲裁。如果控制器接收到外设请求且拥有最高优先级，则执行 4 次主要数据结构的 DMA 传输。接着立即启动交替数据结构的 DMA 周期，无须重新仲裁也无须将 dma_active[C]置为低电平。若发生以下任意一种情况，则控制器将停止主要数据结构和交替数据结构的转换：

- 主控处理器将交替数据结构配置为一个基础周期
- 读取到一个无效数据结构

注意：

当完成 N 次主要数据结构的传输后，可设置 cycle_ctrl 为 b000 使其变为无效的数据结构。

当完成一个基础周期的分散-聚集模式传输，控制器将 dma_done[C]置为有效。

在分散-聚集模式下，控制器利用主要数据结构来编程交替数据结构。下表主要数据结构的 channel_cfg 配置，分为固定值配置和用户配置。

Bit	Field	Value	功能描述
固定值：			
[31:30]	dst_inc	b10	配置控制器使用字作为地址增量
[29:28]	dst_size	b10	配置控制器使用字传输
[27:26]	src_inc	b10	配置控制器使用字作为地址增量
[25:24]	src_size	b10	配置控制器使用字传输
[17:14]	R_power	b0010	配置控制器进行 4 次 DMA 传输
[2:0]	cycle_ctrl	b110	配置控制器进行外设分散-聚集 DMA 周期
用户定义：			
[23:21]	dst_prot_ctrl	-	当写入目标数据后，配置 HPROT 的状态
[20:18]	src_prot_ctrl	-	当写入源数据后，配置 HPROT 的状态
[13:4]	n_minus_1	Na	配置控制器进行 N 次 DMA 传输，其中 N 为 4 的倍数
[3]	next_useburst	-	当设置为 1，在交替传输完成后，控制器会将 CHNL_USEBURST_SET[C]置 1。

a. 由于 R_power 设计为 4，因此 N 必须设置为 4 的倍数。N/4 的值为配置交替数据结构的次数。

下图为外设分散-聚集模式的示例：

初始化：

1. 配置主要数据结构来使能复制 A, B, C 的操作：cycle_ctrl = b110, $2^R = 4, N = 16$
2. 利用下表中的结构，将主要源数据写入存储器。

	src_data_end_ptr	dst_data_end_ptr	channel_cfg	Unused

任务 A 数据	0x0A000000	0x0AE00000	Cycle_ctrl = b101, $2^R = 4, N = 3$	0xFFFFFFFF
任务 B 数据	0x0B000000	0x0BE00000	Cycle_ctrl = b101, $2^R = 2, N = 8$	0xFFFFFFFF
任务 C 数据	0x0C000000	0x0CE00000	Cycle_ctrl = b101, $2^R = 8, N = 5$	0xFFFFFFFF
任务 D 数据	0x0D000000	0x0DE00000	Cycle_ctrl = b001, $2^R = 4, N = 4$	0xFFFFFFFF

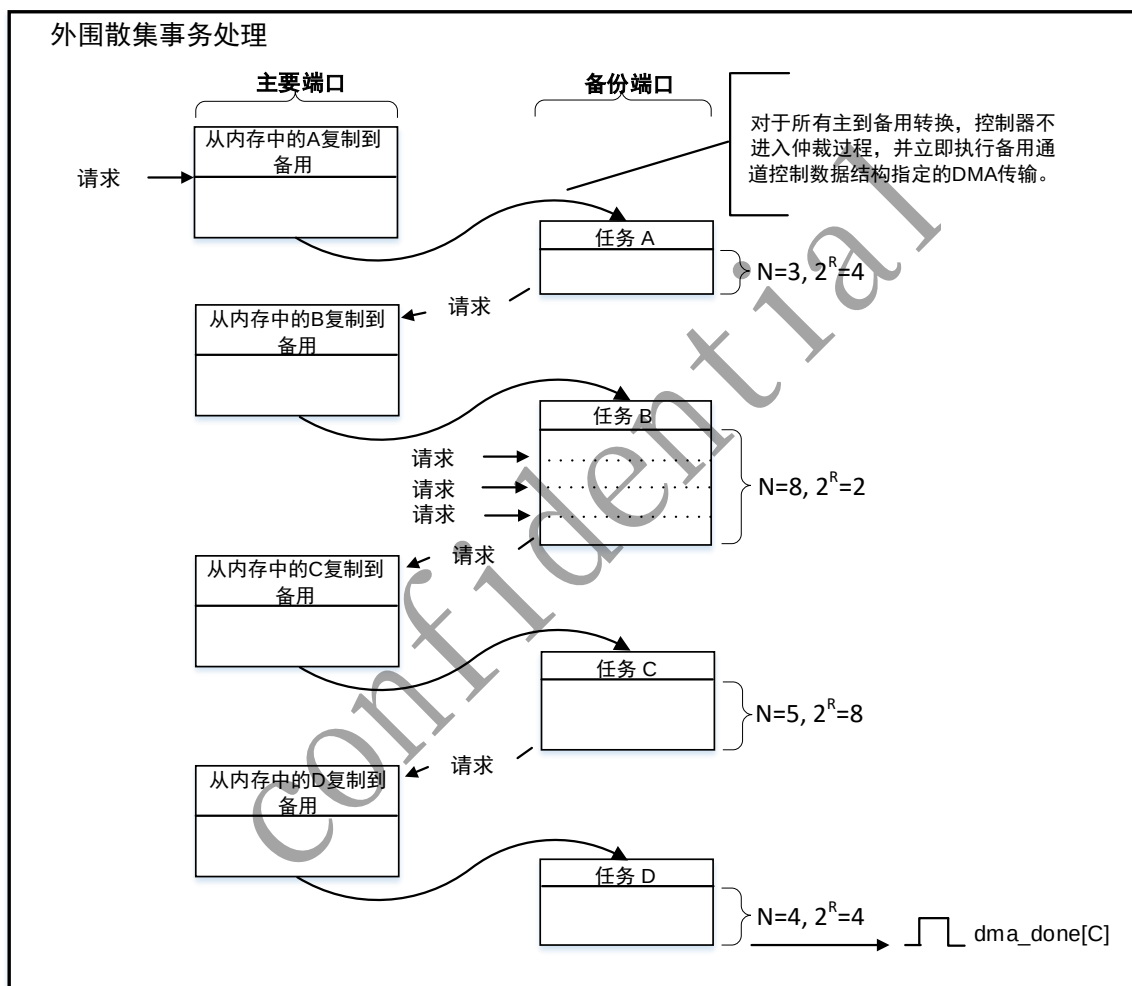


图 2-9 外设交替-聚集示例

初始化

1. 主控处理器通过设置 cycle_ctrl 为 b110, 使主要数据结构运行于外设分散-聚集模式。由于单个通道的数据结构包含 4 个字, 所以 2^R 必须设置为 4。在该示例中, 有 4 个任务, 因此 N 设为 16。
2. 主控处理器将任务 A, B, C 和 D 的数据结构写入由 src_data_end_ptr 指定的存储器地址中。

3. 主控处理器使能该通道。

当控制器接收到 dma_req[] 请求，则外设分散-聚集传输开始执行，流程如下：

Primary, copy A

1. 在接收到请求后，控制器进行 4 次 DMA 传输，并且任务 A 写为交替数据结构。

任务 A

2. 控制器进行任务 A。

3. 当完成任务 A 后，控制器进入仲裁流程。

在外设发出一个新的请求且拥有最高优先级，则继续以下流程：

Primary, copy B

4. 控制器进行 4 次 DMA 传输，且任务 B 写为交替数据结构。

任务 B

5. 控制器执行 4 次 DMA 传输。为使控制器完成该任务，外设还必须再发出 3 次请求。

6. 当任务 B 完成，进入仲裁流程。

在外设发出一个新的请求且拥有最高优先级，则继续以下流程：

Primary, copy C

7. 控制器进行 4 次 DMA 传输，且任务 C 写为交替数据结构。

任务 C

8. 控制器执行任务 C。

9. 当任务 C 完成，进入仲裁流程。

在外设发出一个新的请求且拥有最高优先级，则继续以下流程：

Primary, copy D

10. 控制器进行 4 次 DMA 传输，且任务 D 写为交替数据结构。

11. 控制器设置主要数据结构的 cycle_ctrl 为 b000，表明该数据结构为无效。

任务 D

12. 控制器使用基础 DMA 周期执行任务 D。

13. 控制器将 dma_done[C]置高一个 hclk 周期，并进入仲裁流程。

错误信号

如果控制器在 AHB-Lite 主机接口上检测到一个错误回应，则：

- 禁止该错误信号对应的通道
- 置 dma_err 为高电平

当主机处理器检测到 dma_err 很高时，它必须检查错误发生时哪个通道处于活动状态。它可以做到这一点：

1. 读取 CHNL_ENABLE_SET 寄存器，获取一张禁止通道的列表。

当某通道将 dma_done[] 置为有效，则控制器将禁止该通道。主控处理器上的程序必须记录哪些通道曾置高 dma_done[] 输出。

2. 将步骤 1 中得到的列表和主程序上的记录作比较。若某通道没有 dma_done[C] 被置高的记录，则就是该通道发生了错误信号。

1.4.2 通道控制数据结构

用户必须提供系统存储器空间用来包含通道控制数据结构。该系统存储器必须：

- 提供一个连续的存储空间，以便控制器和主控处理器可以访问
- 其基地址为通道控制数据结构总容的整数倍

下图为当使用 8 个通道和交替数据结构时，控制器所需要的通道控制数据的存储器映射。

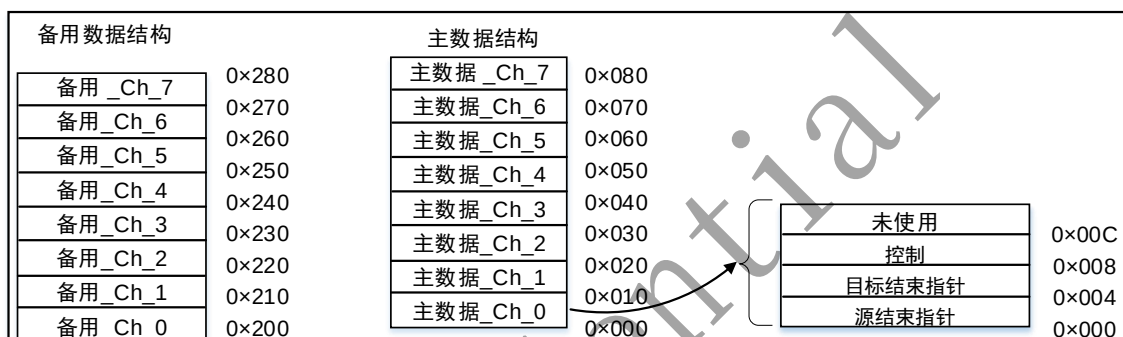


图 2-10 8 通道存储器映射（包括交替数据结构）

该示例中，控制器用地址低 10 位来访问数据结构中的所有地址，因此基地址必须为 0xXXXXXX000, 0xXXXXXX400, 0xXXXXXX800 或者 0xXXXXXXC00。

通过准确设置 CTRL_BASE_PTR 寄存器，用户可配置主要数据结构的基地址。

所需的系统存储器大小取决于以下几个条件：

- 控制器所需要用的 DMA 通道个数
- 若 DMA 通道需要配置使用交替数据结构，请参考通道主要-交替设置。

下表列出了控制器在访问通道控制数据结构的元素时使用的地址位，这取决于控制器包含的通道数。

Address bits							
Number of DMA channels implemented	[9]	[8]	[7]	[6]	[5]	[4]	[3:0]
1						A	
2					A	C[0]	
3-4				A	C[1]	C[0]	0x0, 0x4,
5-8			A	C[2]	C[1]	C[0]	or 0x8
9-16		A	C[3]	C[2]	C[1]	C[0]	
17-32	A	C[4]	C[3]	C[2]	C[1]	C[0]	

表 2-7 通道控制数据结构的地址位设置

A 选择通道控制数据结构：

A = 0 选择主要数据结构

A = 1 选择交替数据结构

C[x:0] 选择 DMA 通道

Address[3:0] 选择以下任一：

0x0 选择源数据结束指针

0x4 选择目标数据结束指针

0x8 选择控制数据配置

0xC 控制器不访问该地址。若有需求，可使主控处理器将该地址用作系统存储器

注意：

用户无需计算交替数据结构的基地址，该信息可由 alt_ctrl_ptr 寄存器提供。

下图显示的存储器映射图为控制器使用 3 个 DMA 通道和交替数据结构。

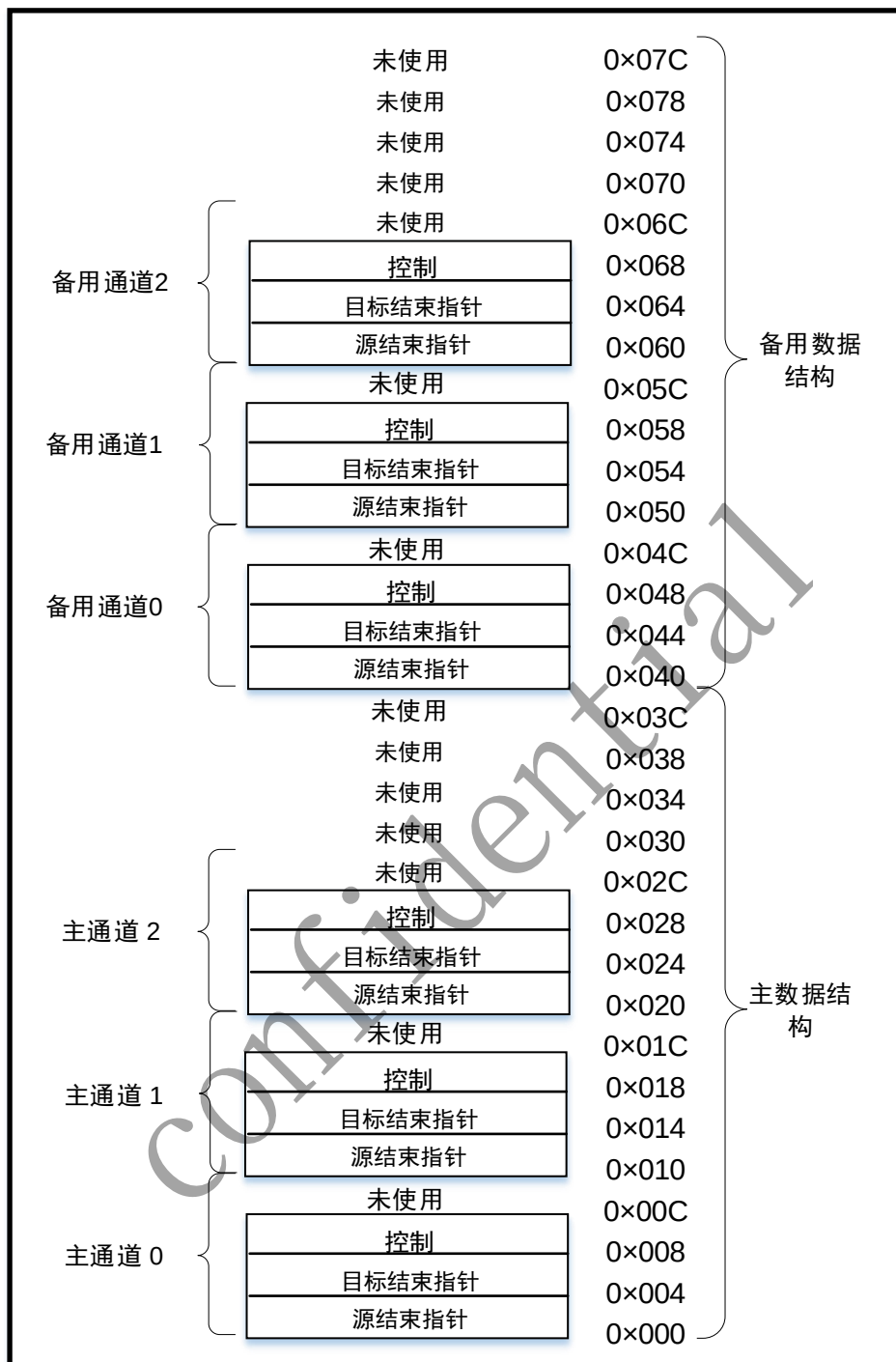


图 2-11 3 DMA 通道存储器映射（包括交替数据结构）

上图显示的系统存储器有 128 个字节。在该示例中，控制器使用地址低 6 位来访问数据结构中的所有地址，因此基地址必须为 0xFFFFFF00 或者 0xFFFFFF80。

下表列出了针对主要数据结构可用的基地址。

DMA 通道数	可用基地址(主要数据结构)
1	0xFFFFFF00, 0xFFFFFF20, 0xFFFFFF40, 0xFFFFFF60, 0xFFFFFF80,

	0xFFFFFFFFA0, 0xFFFFFFFFC0, 0xFFFFFFFFE0
2	0xFFFFFFFF00, 0xFFFFFFFF40, 0xFFFFFFFF80, 0xFFFFFFFFC0
3-4	0xFFFFFFFF00, 0xFFFFFFFF80
5-8	0xFFFFFFFF000, 0xFFFFFFFF100, 0xFFFFFFFF200, 0xFFFFFFFF300, 0xFFFFFFFF400, 0xFFFFFFFF500, 0xFFFFFFFF600, 0xFFFFFFFF700, 0xFFFFFFFF800, 0xFFFFFFFF900, 0xFFFFFFFFA00, 0xFFFFFFFFB00, 0xFFFFFFFFC00, 0xFFFFFFFFD00, 0xFFFFFFFFE00, 0xFFFFFFFFF00
9-16	0xFFFFFFFF000, 0xFFFFFFFF200, 0xFFFFFFFF400, 0xFFFFFFFF600, 0xFFFFFFFF800, 0xFFFFFFFFA00, 0xFFFFFFFFC00, 0xFFFFFFFFE00
17-32	0xFFFFFFFF000, 0xFFFFFFFF400, 0xFFFFFFFF800, 0xFFFFFFFFC00

表 2-8 允许基地址

控制器利用系统存储器来访问两个指针及每个通道控制信息。以下内容详细描述了 32 位存储地址及 DMA 传输地址的计算方法：

- 源数据结束指针
- 目标数据结束指针
- 控制数据配置
- 地址计算

源数据结束指针

src_data_end_ptr 存储地址包含一个指针，指向源数据的最后一个地址。

位	信号名	功能描述
[31:0]	src_data_end_ptr	指向源数据结束地址的指针

表 2-9 src_data_end_ptr 配置说明

在执行 DMA 传输前，该存储地址写入源数据的结束地址。当启动 2^R 次 DMA 传输时，控制器读取 src_data_end_ptr。

注意：

控制器不能写该存储器地址。

目标数据结束指针

dst_data_end_ptr 存储地址包含一个指针，指向目标数据的最后一个地址。

位	信号名	功能描述
[31:0]	dst_data_end_ptr	指向目标数据的结束地址的指针

表 2-10 dst_data_end_ptr 配置说明

在控制器可以执行 DMA 传输之前，该存储地址写入目标数据的结束地址。当控制器开始 2^R 次 dma 传输时，它读取 dst_data_end_ptr。

注意：

控制器不能写该存储器地址。

控制数据配置

channel_cfg 会向控制器提供每一个次 DMA 传输的控制信息。

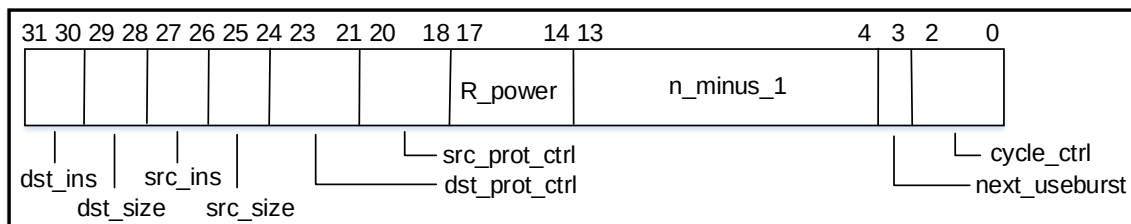


图 2-12 channel_cfg 位配置

位	信号名	功能描述
[31:30]	dst_inc	目标地址增量 地址增量取决于源数据的宽度: 目标数据宽度 = byte b00 = byte. b01 = halfword. b10 = word. b11 = 无增量。地址仍然是 dst_data_end_ptr 中包含的地址。 目标数据宽度 = halfword b00 = reserved. b01 = halfword. b10 = word. b11 = 无增量。地址仍然是 dst_data_end_ptr 中包含的地址。 目标数据宽度 = word b00 = reserved. b01 = reserved. b10 = word. b11 = 无增量。地址仍然是 dst_data_end_ptr 中包含的地址。
[29:28]	dst_size	目标数据大小 注意: dst_size 和 src_size 的值必须一致。
[27:26]	src_inc	设置该位段用来匹配源数据: 源数据宽度 = byte b00 = byte. b01 = halfword. b10 = word. b11 = 无增量。地址仍然是 src_data_end_ptr 中包含的地址。 源数据宽度 = halfword b00 = reserved. b01 = halfword. b10 = word. b11 = 无增量。地址仍然是 src_data_end_ptr 中包含的地址。 源数据宽度 = word b00 = reserved. b01 = reserved. b10 = word. b11 = 无增量。地址仍然是 src_data_end_ptr 中包含的地址。

		址。
[25:24]	src_size	设置该位段用来匹配源数据: b00 = byte b01 = halfword b10 = word b11 = 保留
[23:21]	dst_prot_ctrl	当控制器写目标数据的时候, HRPOT[3:1]状态控制位 Bit [23] HRPOT[3]状态控制位: 0 = HRPOT[3]为低电平, 访问不带高速缓存。 1 = HRPOT[3]为高电平, 访问带高速缓存。 Bit [22] HRPOT[2]状态控制位: 0 = HRPOT[2]为低电平, 访问不带缓冲 1 = HRPOT[2]为高电平, 访问带缓冲。 Bit [21] HRPOT[1]状态控制位: 0 = HRPOT[1]为低电平, 访问为用户模式。 1 = HRPOT[1]为高电平, 访问为特权模式。
[20:18]	src_prot_ctrl	当控制器读取源数据的时候, HRPOT[3:1]状态控制位 Bit [20] HRPOT[3]状态控制位: 0 = HRPOT[3]为低电平, 访问不带高速缓存。 1 = HRPOT[3]为高电平, 访问带高速缓存。 Bit [19] HRPOT[2]状态控制位: 0 = HRPOT[2]为低电平, 访问不带缓冲。 1 = HRPOT[2]为高电平, 访问带缓冲。 Bit [18] HRPOT[1]状态控制位: 0 = HRPOT[1]为低电平, 访问为用户模式。 1 = HRPOT[1]为高电平, 访问为特权模式。
[17:14]	R_power	在控制器重新仲裁前, 该位段决定了 dma 传输可以发生的次数可能的仲裁费率设置如下: b0000 发生 1 次 DMA 传输后仲裁 b0001 发生 2 次 DMA 传输后仲裁 b0010 发生 4 次 DMA 传输后仲裁 b0011 发生 8 次 DMA 传输后仲裁 b0100 发生 16 次 DMA 传输后仲裁 b0101 发生 32 次 DMA 传输后仲裁 b0110 发生 64 次 DMA 传输后仲裁 b0111 发生 128 次 DMA 传输后仲裁 b1000 发生 256 次 DMA 传输后仲裁 b1001 发生 512 次 DMA 传输后仲裁 b1010-b1111 发生 1024 次 DMA 传输后仲裁。由于最大的传输次数为 1024, 由此表明 DMA 传输中无仲裁发生。
[13:4]	n_minus_1	在 DMA 周期开始前, 该位段代表的是 DMA 周期中包含的 DMA 传输的总次数。用户须根据所需要的 DMA 周期的来设置该位段。该 10 位值为 DMA 传输次数减 1。 b0000000000 = 1 次 DMA 传输 b0000000001 = 2 次 DMA 传输 b0000000010 = 3 次 DMA 传输 b0000000011 = 4 次 DMA 传输 b0000000100 = 5 次 DMA 传输 . . .

		b11111111 = 1024 次 DMA 传输 在进入仲裁流程前，控制器会立即更新该位段，可使控制器保存该 DMA 周期中还需完成的剩下的 DMA 传输次数。
[3]	next_useburst	当控制器在外设分散-聚集模式下，且使用交替数据结构，该位段控制 CHNL_USEBURST_SET[C]是否置 1。 需要注意的是，在完成由交替数据结构指定的 DMA 周期前，如果剩余的传输次数小于 2^R，控制器会将 CHNL_USEBURST_SET[C]设置为 0。Next_useburst 的设定控制了是否需要再次修改 CHNL_USEBURST_SET[C]。 在外设分散-聚集 DMA 周期模式下，当使用交替数据结构的 DMA 周期完成后，会发生以下任一情况： 0：控制器不改变 CHNL_USEBURST_SET[C]的值。 如果 CHNL_USEBURST_SET[C]为 0，则对于所有在外设分散-聚集模式下的剩余的 DMA 周期，控制器将回应 dma_req[]和 dma_sreq[]的请求。 1：控制器将 CHNL_USEBURST_SET[C]置 1。因此，对于所有在外设分散-聚集模式下的剩余的 DMA 周期，控制器仅回应 dma_req[]的请求。
[2:0]	cycle_ctrl	DMA 周期的工作模式： b000 停止。表明该数据结构无效。 b001 基础。在进入仲裁流程前，控制器必须接收到一个新的请求才能完成 DMA 周期。 b010 自动请求。在仲裁过程中，控制器在合适的通道上自动插入一个请求。这就意味着最初请求已足够完成 DMA 周期。 b011 乒乓。控制器使用任一数据结构完成一个 DMA 周期。在该周期完成后，且如果主控处理器已更新了原始数据结构，则控制器使用原始数据结构执行一次 DMA 周期。直到读取到无效数据结构或者主控处理器将 cycle_ctrl 改为 b001 或 b010，控制器将会继续进行 DMA 周期。 b100 存储器分散-聚集。当控制器运行在存储器分散-聚集模式下，用户仅可在主要数据结构下使用该值。 b101 存储器分散-聚集。当控制器运行在存储器分散-聚集模式下，用户仅可在交替数据结构下使用该值。 b110 外设分散-聚集。当控制器运行在外设分散-聚集模式下，用户仅可在主要数据结构下使用该值。 b111 外设分散-聚集。当控制器运行在外设分散-聚集模式下，用户仅可在主要交替结构下使用该值。

表 2-11 channel_cfg 配置说明

在 DMA 周期或者 2^R DMA 传输开始的时候，控制器会从系统存储器中获取 channel_cfg 的值。当完成 2^R 或 N 次传输后，新的 channel_cfg 值会被存储到系统存储器中。

控制器不支持 dst_size 和 src_size 拥有两个不同的值。如果检测到两个值不匹配，src_size 的值会被作为源数据和目标数据的大小。当下一次 n_minus_1 更新时，dst_size 将被设置为 src_size 的值。

当完成 N 次传输后，控制器会将 cycle_ctrl 设置为 b000，以此表明 channel_cfg 数据为无效，用来防止控制器重复相同的 DMA 传输。

地址计算

为了计算 DMA 传输的源地址，控制器必须将 `n_minus_1` 的值向左移，移动值由 `src_inc` 定义。接着从源数据结束指针中减去移动后的 `n_minus_1` 的值。同样地，为了计算 DMA 传输的目标地址，控制器也必须将 `n_minus_1` 的值向左移，移动量由 `dst_inc` 定义，接着从目标数据结束指针中减去移动后的 `n_minus_1` 的值。

根据 `src_inc` 和 `dst_inc` 的值，源地址和目标地址的计算可使用同一个等式。：

src_inc=b00 and dst_inc=b00

- 源地址= `src_data_end_ptr - n_minus_1`
- 目标地址= `dst_data_end_ptr - n_minus_1`.

src_inc=b01 and dst_inc=b01

- 源地址= `src_data_end_ptr - (n_minus_1 << 1)`
- 目标地址= `dst_data_end_ptr - (n_minus_1 << 1)`.

src_inc=b10 and dst_inc=b10

- 源地址= `src_data_end_ptr - (n_minus_1 << 2)`
- 目标地址= `dst_data_end_ptr - (n_minus_1 << 2)`.

src_inc=b11 and dst_inc=b11

- 源地址= `src_data_end_ptr`
- 目标地址= `dst_data_end_ptr`.

Initial values of channel_cfg, prior to the DMA cycle				
src_size=b10, dst_inc=b10, n_minus_1=b101, cycle_ctrl=1				
	End Pointer	Count	Difference ^a	Address
DMA transfers	0x2AC	5	0x14	0x298
	0x2AC	4	0x10	0x29C
	0x2AC	3	0xC	0x2A0
	0x2AC	2	0x8	0x2A4
	0x2AC	1	0x4	0x2A8
	0x2AC	0	0x0	0x2AC
Final values of channel_cfg, after the DMA cycle				
src_size=b10, dst_inc=b10, n_minus_1=0, cycle_ctrl=0				

表 2-12 6 个字的 DMA 周期，地址增量为 1 个字

a. 上表中的差值为次数向左移动 `dst_inc` 后的结果。

Initial values of channel_cfg, prior to the DMA cycle				
src_size=b00, dst_inc=b01, n_minus_1=b1011, cycle_ctrl=1, R_power=b11				
	End Pointer	Count	Difference ^a	Address
DMA transfers	0x5E7	11	0x16	0x5D1
	0x5E7	10	0x14	0x5D3
	0x5E7	9	0x12	0x5D5
	0x5E7	8	0x10	0x5D7
	0x5E7	7	0xE	0x5D9
	0x5E7	6	0xC	0x5DB
	0x5E7	5	0xA	0x5DD
	0x5E7	4	0x8	0x5DF
Values of channel_cfg after 2 ^R DMA transfers				
src_size=b00, dst_inc=b01, n_minus_1=b011, cycle_ctrl=1, R_power=b11				
	End Pointer	Count	Difference	Address
DMA transfers	0x5E7	3	0x6	0x5E1
	0x5E7	2	0x4	0x5E3
	0x5E7	1	0x2	0x5E5
	0x5E7	0	0x0	0x5E7
Final values of channel_cfg, after the DMA cycle				
src_size=b00, dst_inc=b01, n_minus_1=0, cycle_ctrl=0 ^b , R_power=b11				

表 2-13 DMA 周期 12 bytes 通过半字增量

- a. 上表中的差值为次数向左移动 dst_inc 后的结果
b. 当控制器完成 DMA 周期后，通过将 cycle_ctrl 清零使 channel_cfg 无效。

1.4.3 DMA 通道配置

DMA 控制器包含 8 个通道，每个 DMA 通道对应一个 DMA 多路复用器，每个多路复用器包含了微控制器所有的 DMA 申请源，由 DMA_CHx_SEL(x=0,1...7)配置选择。多路复用器和 DMA 之间连接图如下：

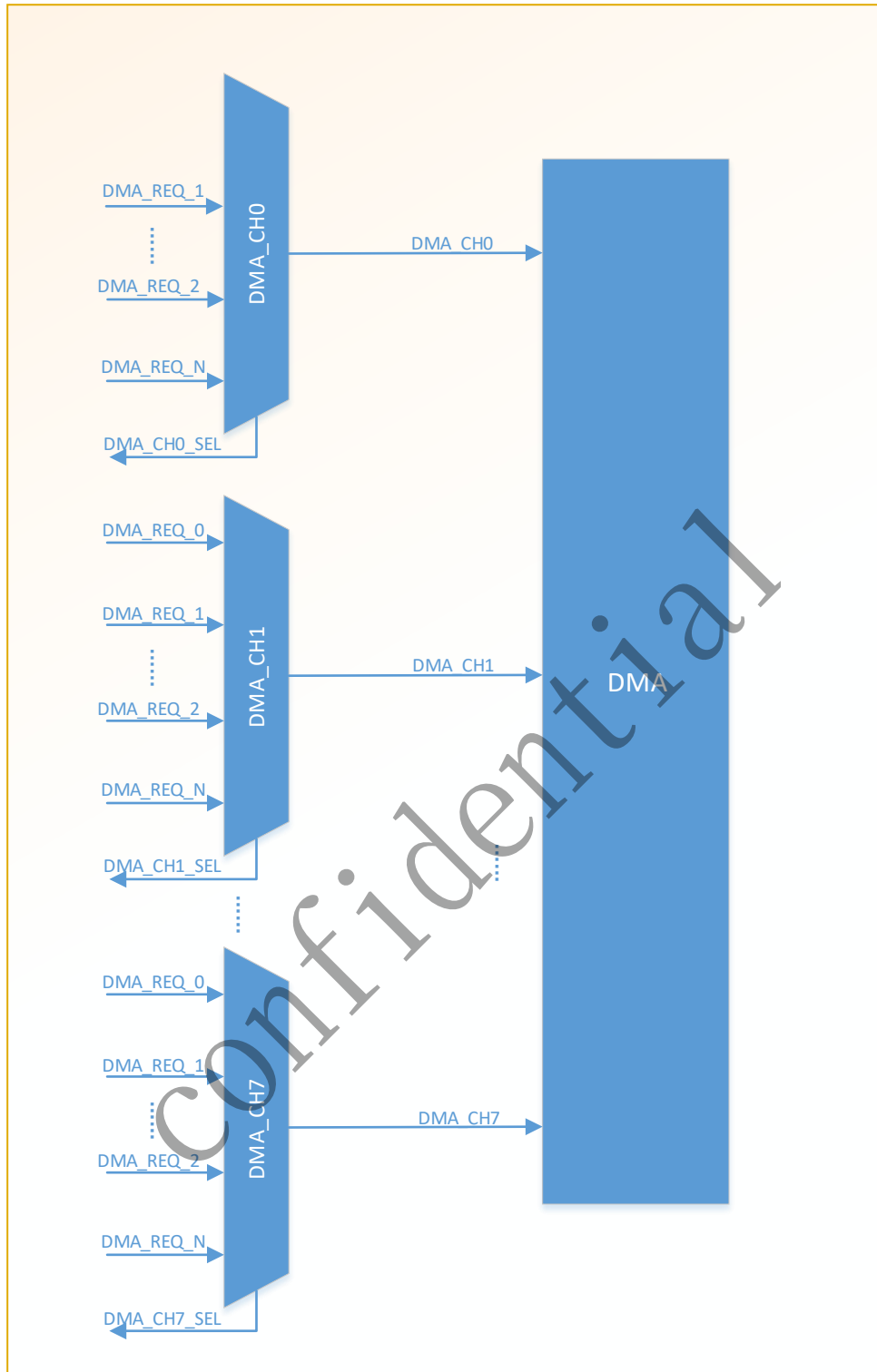


图 2-2 DMA 多路复用器与 DMA 连接图

每个 DMA 多路复用器可选择的 DMA 请求如下表所示：

序号	模块名	DMA 请求源
0	UART1	UART1_RX
1		UART1_TX
2	UART2	UART2_RX

3		UART2_TX
4	UART3	UART3_RX
5		UART3_TX
6	-	保留
7		保留
8	SSI	SSI_RX
9		SSI_TX
10	SPI2	SPI2_RX
11		SPI2_TX
12	I2C1	I2C1_RX
13		I2C1_TX
14	I2C2	I2C2_RX
15		I2C2_TX
16	-	保留
17		保留
18		保留
19	CRYPT	CRYPT_WR
20		CRYPT_RD
21	TRNG	TRNG
22	ECC	ECC
23	ADC	ADC
24	ADTIM1	ADTIM1_CH1
25		ADTIM1_CH2
26		ADTIM1_CH3
27		ADTIM1_CH4
28		ADTIM1_UP
29		ADTIM1_TRIG
30		ADTIM1_COM
31	-	保留
32	GPTIMA1	GPTIMA1_CH1
33		GPTIMA1_CH2
34		GPTIMA1_CH3
35		GPTIMA1_CH4
36		GPTIMA1_UP
37		GPTIMA1_TRIG
38	--	保留
39		保留
40	GPTIMB1	GPTIMB1_CH1
41		GPTIMB1_CH2
42		GPTIMB1_CH3
43		GPTIMB1_CH4
44		GPTIMB1_UP
45		GPTIMB1_TRIG
46	--	保留
47		保留
48	GPTIMC1	GPTIMC1_CH1
49		GPTIMC1_CH2
50		GPTIMC1_UP
51		GPTIMC1_TRIG
52		GPTIMC1_COM

53	-	保留
54		保留
55		保留
56	BSTIM1	BSTIM1_UP
57	-	保留
58		保留
59		保留
60		保留
61		保留
62		保留
63		保留

表 2-14 DMA 请求列表

confidential

1.5 特殊功能寄存器

1.5.1 DMA 状态寄存器 (DMA_STATUS)

		Address offset:		00 _H											
		Reset value:		00010000_00001011_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								STATE				Reserved			MAS TER_ ENAB LE

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留, 保持复位值
STATE	7:4	R	0x0	控制状态机的当前状态: 0000 = 空闲 0001 = 读取通道控制器数据 0010 = 读取源数据结束指针 0011 = 读取目标数据结束指针 0100 = 读取源数据 0101 = 写目标数据 0110 = 等待 DMA 请求清 0 0111 = 写通道控制器数据 1000 = 延迟 1001 = 完成 1010 = 外设分散-聚集转换 1011-1111 = 未定义.
Reserved	3:1	N/A	0x00	保留, 保持复位值
MASTER_ENABLE	0	R	0x0	使能控制器 0 = 使能控制器 1 = 禁止控制器

1.5.2 DMA 配置寄存器 (DMA_CFG)

		Address offset:		04 _H											
		Reset value:		-											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CHNL_PROT_CTRL				Reserved			MAS TER_ ENAB LE

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留, 保持复位值

CHNL_PROT_CTRL	7:5	W	0x0	通过控制 HPROT[3:1]的信号电平对 AHB-Lite 做保护设置: Bit [7] 控制 HPROT[3]来表明是否正在发生可缓存访问 Bit [6] 控制 HPROT[2]来表明是否正在发生可缓冲访问 Bit [5] 控制 HPROT[1]来表明是否正在发生特殊权限访问 注意: 当 bit [n] = 1 相应 HPROT 为高电平。 当 bit [n] = 0 相应 HPROT 为低电平。
Reserved	4:1	N/A	0x00	保留, 保持复位值
MASTER_ENABLE	0	W	0x0	使能控制器 0 =禁止控制器 1 =使能控制器

1. 5. 3 DMA 通道控制数据库指针寄存器(CTRL_BASE_PTR)

		Address offset:	08 _H												
		Reset value:	00000000_00000000_00000000_00000000 _B												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CTRL_BASE_PTR															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTRL_BASE_PTR								Reserved							

Name	Bit	Type	Reset value	Description
CTRL_BASE_PTR	31:9	R/W	0x0	● 主要数据结构基地址指针
Reserved	8:0	N/A	0x00	保留, 保持复位值

1. 5. 4 DMA 通道交替控制数据库指针寄存器(ALT_CTRL_BASE_PTR)

		Address offset:	0C _H												
		Reset value:	00000000_00000000_00000000_00000000 _B												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ALT_CTRL_BASE_PTR															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALT_CTRL_BASE_PTR								Reserved							

Name	Bit	Type	Reset value	Description
ALT_CTRL_BASE_PTR	31:0	R	0x0	交替数据结构的基地址

1. 5. 5 DMA 通道等待请求状态寄存器(DMA_WAITONREQ_STATUS)

		Address offset:		10 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DMA_WAITONREQ_STATUS							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留，保持复位值
DMA_WAITONREQ_STATUS	7:0	R	0x0	通道等待状态： Bit [C] = 0 dma_waitonreq[C] 为低电平 Bit [C] = 1 dma_waitonreq[C] 为高电平

1. 5. 6 DMA 通道软件请求寄存器(CHNL_SW_REQUEST)

		Address offset:		14 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CHNL_SW_REQUEST							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留，保持复位值
CHNL_SW_REQUEST	7:0	W	0x0	通道软件请求： Bit [C] = 0: 通道 n 上不生成 DMA 请求。 Bit [C] = 1: 通道 n 上生成一个 DMA 请求。 该寄存器为只写寄存器，每个对应的 bit 都可在相应的通道上生产软件 DMA 请求

1. 5. 7 DMA 通道使用冲突设置寄存器(CHNL_USEBURST_SET)

		Address offset:		18 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CHNL_USEBURST_SET							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留，保持复位值
CHNL_USEBURST_SET	7:0	R/W	0x0	返回冲突的使用状态，或禁止 dma_sreq[C] 产生 DMA 请求读取： Bit [C] = 0: 通道 C 对 dma_req[C] 或 dma_sreq[C] 请求做出回应。控制器执行 2^R 或单次总线传输。 Bit [C] = 1: 通道 C 不回应 dma_sreq[C] 请求。控制器只回应 dma_req[C] 请求，执行 2^R 次传输。 Write as: Bit [C] = 0 : 无效。使用 CHNL_USEBURST_CLR 寄存器将 bit[C] 写 0。 Bit [C] = 1 : 禁止 dma_sreq[C] 产生 DMA 请求。控制器执行 2^R 次传输。

1. 5. 8 DMA 通道使用冲突清除寄存器(CHNL_USEBURST_CLR)

		Address offset:			1C _H										
		Reset value:			—										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CHNL_USEBURST_CLR							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留，保持复位值
CHNL_USEBURST_CLR	7:0	W	0x0	使能 dma_sreq[C] 产生请求写入： Bit [C] = 0: 无效。使用 CHNL_USEBURST_SET 寄存器禁止 dma_sreq[] 产生请求。 Bit [C] = 1: 使能 dma_sreq[C] 产生 DMA 请求。

1. 5. 9 DMA 通道请求屏蔽寄存器(CHNL_REQ_MASK_SET)

		Address offset:			20 _H										
		Reset value:			00000000_00000000_00000000_00000000 _B										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CHNL_REQ_MASK_SET							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留，保持复位值

CHNL_REQ_MASK_SET	7:0 R/W 0x0	返回 dma_req[]和 dma_sreq[]的屏蔽状态， 或禁止相应通道产生 DMA 请求 读取： Bit [C] = 0: 通道 C 上的外部请求已使能。 Bit [C] = 1: 通道 C 上的外部请求已禁止。 写入： Bit [C] = 0: 无效。使用 CHNL_REQ_MASK_CLR 寄存器来使能 DMA 请求。 Bit [C] = 1: 禁止 dma_req[C]和 dma_sreq[C]产生 DMA 请求。

1. 5. 10 DMA 通道请求屏蔽清除寄存器(CHNL_REQ_MASK_CLR)

		Address offset:		24 _H											
		Reset value:		-											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CHNL_REQ_MASK_CLR							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留，保持复位值
CHNL_REQ_MASK_CLR	7:0	W	0x0	使能通道上的 dma_req[]和 dma_sreq[]请求 写入： Bit [C] = 0: 无效。使用 chnl_req_mask_set 寄存器禁止 dma_req[]和 dma_sreq[]产生请求。 Bit [C] = 1: 使能 dma_sreq[C]或 dma_sreq[]产生 DMA 请求。

1. 5. 11 DMA 通道使能配置寄存器(CHNL_ENABLE_SET)

		Address offset:		28 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CHNL_ENABLE_SET							

Name	Bit	Type	Reset value	Description
Reserved	31:12	N/A	0x0	保留，保持复位值
CHNL_ENABLE_SET	11:0	R/W	0x0	返回通道的状态，或使能相应的通道 读取： Bit [C] = 0 通道 C 禁止。 Bit [C] = 1 通道 C 使能。

写入:

Bit [C] = 0 无效。设置 CHNL_ENABLE_CLR 用来禁止某通道。

Bit [C] = 1 使能相应通道

1. 5. 12 DMA 通道使能清除寄存器(CHNL_ENABLE_CLR)

		Address offset:		2C _H											
		Reset value:		-											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CHNL_ENABLE_CLR							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留, 保持复位值
CHNL_ENABLE_CLR	7:0	W	0x0	禁止相应 DMA 通道 写入: Bit [C] = 0 无效。设置 CHNL_ENABLE_SET 寄存器来使 能 DMA 通道。 Bit [C] = 1 禁止相应通道。 注: 当发生以下任意一种情况时, 控制器可通过设置相应 CHNL_ENABLE_CLR 来禁止通道: 1. 控制器完成 DMA 周期 2. 控制器读取 channel_cfg, 其 cycle_ctrl 位段为 b000 3. AHB-Lite 总线上发生错误

1. 5. 13 DMA 通道主要交替配置寄存器(CHNL_PRI_ALT_SET)

		Address offset:		30 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CHNL_PRI_ALT_SET							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留, 保持复位值
CHNL_PRI_ALT_SET	7:0	R/W	0x0	返回通道的控制数据结构的状态, 或为相应的 DMA 通道选择交替数据结构. 读取: Bit [C] = 0 相应 DMA 通道使用的是主要数据结构。 Bit [C] = 1 相应 DMA 通道使用的是交替数据结构。 写入: Bit [C] = 0 无效。设置 CHNL_PRI_ALT_CLR 中相应的比特位

为 0。

Bit [C] = 1 为相应 DMA 通道选择交替数据结构。

注：

当发生以下情况时，控制器将翻转 CHNL_PRI_ALT_SET 中相应的比特位的值：

1. 在存储器/外设分散-聚集 DMA 周期模式下，完成由主要数据结构指定的 4 次传输。
2. 在乒乓 DMA 周期模式下，完成由主要数据结构指定的所有传输。
3. 在以下 DMA 周期模式下，完成由交替数据结构指定的所有传输。

- 乒乓模式
- 存储器分散-聚集模式
- 外设分散-聚集模式

1. 5. 14 DMA 通道主要交替清除寄存器(CHNL_PRI_ALT_CLR)

		Address offset:		34 _H											
		Reset value:		—											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CHNL_PRI_ALT_CLR							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留，保持复位值
CHNL_PRI_ALT_CLR	7:0	W	0x0	设置相应的比特位，为相应的 DMA 通道选择主要数据结构写入： Bit [C] = 0 无效。设置 CHNL_PRI_ALT_SET 寄存器用来选择交替数据结构。 Bit [C] = 1 为相应 DMA 通道选择主要数据结构。 注： 当发生以下情况时，控制器将翻转 CHNL_PRI_ALT_CLR 中相应的比特位的值： 1. 在存储器/外设分散-聚集 DMA 周期模式下，完成由主要数据结构指定的 4 次传输。 2. 在乒乓 DMA 周期模式下，完成由主要数据结构指定的所有传输。 3. 在以下 DMA 周期模式下，完成由交替数据结构指定的所有传输。 ● 乒乓模式 ● 存储器分散-聚集模式 ● 外设分散-聚集模式

1. 5. 15 DMA 通道优先级配置寄存器(CHNL_PRIORITY_SET)

		Address offset:		38 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											

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

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留，保持复位值
CHNL_PRIORITY_SET	7:0	R/W	0x0	返回通道的优先级屏蔽状态，或设置通道优先级为高 读取： Bit [C] = 0 相应 DMA 通道为默认优先级。 Bit [C] = 1 相应 DMA 通道为高优先级。 写入： Bit [C] = 0 无效。设置 CHNL_PRIORITY_CLR 寄存器将相应通道设置为默认优先级。 Bit [C] = 1 设置相应 DMA 通道为高优先级。

1. 5. 16 DMA 通道优先级清除寄存器(CHNL_PRIORITY_CLR)

		Address offset:		3C _H											
		Reset value:													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CHNL_PRIORITY_CLR							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留，保持复位值
CHNL_PRIORITY_CLR	7:0	R/W	0x0	设置相应的比特位，为相应的 DMA 通道选择默认优先级 写入： Bit [C] = 0 无效。设置 CHNL_PRIORITY_SET 寄存器，将相应 DMA 通道设置为高优先级。 Bit [C] = 1 将相应 DMA 通道设置为默认优先级。

1. 5. 17 DMA 总线错误清除寄存器(ERR_CLR)

		Address offset:		4C _H											
		Reset value:		00000000_00000000_00000000_00000000 _s											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ERR_CLR

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x0	保留，保持复位值
ERR_CLR	0	R/W	0x0	返回 dma_err 的状态，或设置该信号为低电平 读取： 0 = dma_err 为低电平 1 = dma_err 为高电平 写入： 0 = 无效。dma_err 状态不变。 1 = 设置 dma_err 为低电平。 注： 当 AHB-Lite 总线上发生错误的同时，若 dma_err 被置为无效，则错误条件先发生，dma_err 保持有效。

1. 5. 18 DMA 中断状态寄存器（DMA_IFLAG0）

		Address offset:			1000 _H										
		Reset value:			00000000_00000000_00000000_00000000 _B										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA ERRI F	Reserved														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CH7D ONEI F	CH6D ONEI F	CH5D ONEI F	CH4D ONEI F	CH3D ONEI F	CH2D ONEI F	CH1D ONEI F	CH0D ONEI F

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留，保持复位值
CH7DONEIF	7	R	0x0	DMA 通道 7 结束中断标志 0: DMA 未结束 1: DMA 结束
CH6DONEIF	6	R	0x0	DMA 通道 6 结束中断标志 0: DMA 未结束 1: DMA 结束
CH5DONEIF	5	R	0x0	DMA 通道 5 结束中断标志 0: DMA 未结束 1: DMA 结束
CH4DONEIF	4	R	0x0	DMA 通道 4 结束中断标志 0: DMA 未结束 1: DMA 结束
CH3DONEIF	3	R	0x0	DMA 通道 3 结束中断标志 0: DMA 未结束 1: DMA 结束
CH2DONEIF	2	R	0x0	DMA 通道 2 结束中断标志 0: DMA 未结束 1: DMA 结束
CH1DONEIF	1	R	0x0	DMA 通道 1 结束中断标志 0: DMA 未结束 1: DMA 结束
CH0DONEIF	0	R	0x0	DMA 通道 0 结束中断标志 0: DMA 未结束 1: DMA 结束

1. 5. 19 DMA 中断状态清零寄存器 (DMA_ICFR0)

		Address offset:		1008 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA ERRIF		Reserved													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CH7D ONEIF	CH6D ONEIF	CH5D ONEIF	CH4D ONEIF	CH3D ONEIF	CH2D ONEIF	CH1D ONEIF	CH0D ONEIF

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留，保持复位值
CH7DONEC	7	W1	0x0	DMA 通道 7 结束中断标志清零 对该位写 1 有效。
CH6DONEC	6	W1	0x0	DMA 通道 6 结束中断标志清零 对该位写 1 有效。
CH5DONEC	5	W1	0x0	DMA 通道 5 结束中断标志清零 对该位写 1 有效。
CH4DONEC	4	W1	0x0	DMA 通道 4 结束中断标志清零 对该位写 1 有效。
CH3DONEC	3	W1	0x0	DMA 通道 3 结束中断标志清零 对该位写 1 有效。
CH2DONEC	2	W1	0x0	DMA 通道 2 结束中断标志清零 对该位写 1 有效。
CH1DONEC	1	W1	0x0	DMA 通道 1 结束中断标志清零 对该位写 1 有效。
CH0DONEC	0	W1	0x0	DMA 通道 0 结束中断标志清零 对该位写 1 有效。

1. 5. 20 DMA 中断使能控制寄存器 (DMA_IEFR0)

		Address offset:		100C _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DMA ERRIF		Reserved													
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CH7D ONEIF	CH6D ONEIF	CH5D ONEIF	CH4D ONEIF	CH3D ONEIF	CH2D ONEIF	CH1D ONEIF	CH0D ONEIF

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留，保持复位值
CH7DONEIE	7	R/W	0x0	DMA 通道 7 结束中断使能 0: 中断禁止 1: 中断使能

CH6DONEIE	6	R/W	0x0	DMA 通道 6 结束中断使能 0: 中断禁止 1: 中断使能
CH5DONEIE	5	R/W	0x0	DMA 通道 5 结束中断使能 0: 中断禁止 1: 中断使能
CH4DONEIE	4	R/W	0x0	DMA 通道 4 结束中断使能 0: 中断禁止 1: 中断使能
CH3DONEIE	3	R/W	0x0	DMA 通道 3 结束中断使能 0: 中断禁止 1: 中断使能
CH2DONEIE	2	R/W	0x0	DMA 通道 2 结束中断使能 0: 中断禁止 1: 中断使能
CH1DONEIE	1	R/W	0x0	DMA 通道 1 结束中断使能 0: 中断禁止 1: 中断使能
CH0DONEIE	0	R/W	0x0	DMA 通道 0 结束中断使能 0: 中断禁止 1: 中断使能

1. 5. 21 DMA 通道 0 复用选择寄存器 (DMA_CFG0)

		Address offset:		1100 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		DMA_CH3_SEL						Reserved			DMA_CH2_SEL				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DMA_CH1_SEL						Reserved			DMA_CH0_SEL				

Name	Bit	Type	Reset value	Description
Reserved	31:30	N/A	0x00	保留, 保持复位值
DMA_CH3_SEL	29:24	R/W	0x0	同 DMA_CH0_SEL
Reserved	23:22	N/A	0x00	保留, 保持复位值
DMA_CH2_SEL	21:16	R/W	0x0	同 DMA_CH0_SEL
Reserved	15:14	N/A	0x00	保留, 保持复位值
DMA_CH1_SEL	13:8	R/W	0x0	同 DMA_CH0_SEL
Reserved	7:6	N/A	0x00	保留, 保持复位值
DMA_CH0_SEL	5:0	R/W	0x0	详见 DMA 请求列表

1. 5. 22 DMA 通道 1 复用选择寄存器 (DMA_CFG1)

		Address offset:		1104 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		DMA_CH7_SEL						Reserved		DMA_CH6_SEL					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DMA_CH5_SEL						Reserved		DMA_CH4_SEL					

Name	Bit	Type	Reset value	Description
Reserved	31:30	N/A	0x00	保留，保持复位值
DMA_CH7_SEL	29:24	R/W	0x0	同 DMA_CH0_SEL
Reserved	23:22	N/A	0x00	保留，保持复位值
DMA_CH6_SEL	21:16	R/W	0x0	同 DMA_CH0_SEL
Reserved	15:14	N/A	0x00	保留，保持复位值
DMA_CH5_SEL	13:8	R/W	0x0	同 DMA_CH0_SEL
Reserved	7:6	N/A	0x00	保留，保持复位值
DMA_CH4_SEL	5:0	R/W	0x0	同 DMA_CH0_SEL

confidential

第3章 独立看门狗（IWDG）

1.1 IWDG 简介

独立看门狗 IWDG，可用于检测软件和硬件异常，如主时钟停振，程序跑飞不再喂狗等。

1.2 IWDG 主要特性

- ◆ 自由运行的递减计数器
 - 写入 IWDG_RLR 寄存器将重新加载看门狗
 - 看门狗被激活后，则在计数器计数至 0 时产生复位
- ◆ IWDG 中断可唤醒睡眠模式 0 和睡眠模式 2

1.3 IWDG 功能描述

IWDG 看门狗可用于检测软件和硬件异常。

软件配置 EN=1，IWDG 计数器载入 IWDG_RLR 寄存器值，开始递减计数，当计数到 0 时，IWDG 产生中断标志，并在下一个计数时钟到来时，计数器再次载入 IWDG_RLR 寄存器值，并继续递减计数。当计数器再次计数到 0 时，如果 IWDG 中断标志仍没有被软件清零，则 IWDG 模块将产生复位信号。配置 CLKS，可以选择计数时钟源；配置 IWDG_RLR 寄存器，可设置计数初值；读取 IWDG_VAL，可得到 IWDG 当前计数值。写入装载值寄存器 IWDG_RLR 时，计数器当前值寄存器将被载入 IWDG_RLR 寄存器值。

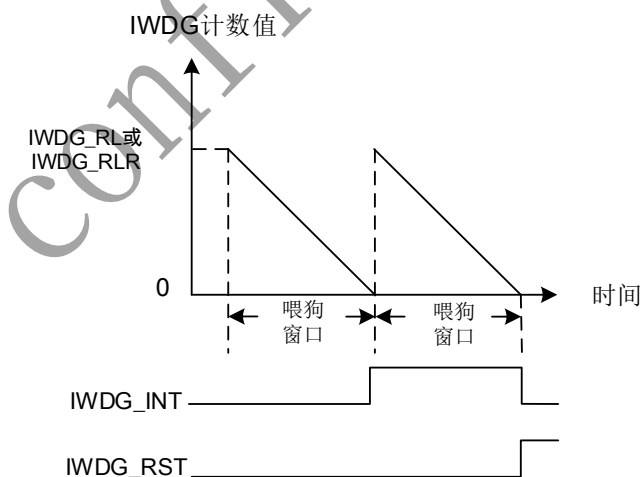


图 3-1 独立看门狗时序图

写 IWDG_ICR，则 IWDG 重新载入计数初值，并重新进行递减计数。

IWDG 模块支持寄存器写保护，配置寄存器 IWDG_LOCK=0x1ACCE551，可去除 IWDG 寄存器的写保护状态，此后可进行被保护的 IWDG 寄存器的写操作，否则无法写 IWDG 寄存器。

1.4 IWDG 寄存器描述

1.4.1 IWDG 重载寄存器 (IWDG_RLR)

		Address offset:		00 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RLR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RLR[15:0]															

Name	Bit	Type	Reset value	Description
RLR	31:0	W	0x0	IWDG 重载值 计数范围 0x0000_0001~0xFFFF_FFFF。如果为 0，IWDG 不计数。

1.4.2 IWDG 计数器值寄存器 (IWDG_VAL)

		Address offset:		04 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VAL[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL[15:0]															

Name	Bit	Type	Reset value	Description
VAL	31:0	R	0x0	IWDG 计数值 读取时返回 IWDG 计数器的当前计数值

1.4.3 IWDG 控制寄存器 (IWDG_CR)

		Address offset:		08 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												CLKS	RSTEN	IE	EN

Name	Bit	Type	Reset value	Description

Reserved	31:6	N/A	0x00	保留，保持复位值
CLKS	5:4	R/W	0x0	IWDG 计数时钟选择位 0: 计数时钟源 HSI 1: 计数时钟源 HSE 2: 计数时钟源 LSI 3: 计数时钟源 LSE
Reserved	3	N/A	0x0	保留，保持复位值
RSTEN	2	R/W	0x0	IWDG 复位使能位 0: 禁止 1: 使能，IWDG 计数到 0 时，产生复位信号，将芯片复位
IE	1	R/W	0x0	IWDG 中断使能位 0: 禁止 1: 使能，IWDG 计数到 0 时，产生中断标志
EN	0	R/W	0x0	IWDG 模块使能位 0: 禁止 1: 使能

1. 4. 4 IWDG 中断清除寄存器 (IWDG_ICR)

		Address offset:		0C _H											
		Reset value:		00000000_00000000_00000000_00000000 ₈											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ICR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ICR[15:0]															

Name	Bit	Type	Reset value	Description
ICR<31:0>	31:0	W	0x0	IWDG 中断标志清 0 位 对 IWDG_ICR 寄存器进行任意写操作，IWDG 中断标志位均被清零，计数器重载 IWDG_RLR 寄存器值，继续递减计数

1. 4. 5 IWDG 中断标志寄存器 (IWDG_RIF)

		Address offset:		10 _H											
		Reset value:		00000000_00000000_00000000_00000000 ₈											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															RIF

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x0	保留，保持复位值
RIF	0	R	0x0	IWDG 中断标志位 0: 未产生中断

1: IWDG 计数器计数到 0, 产生中断
写寄存器 IWDG_ICR, 可清除 IWDG 中断标志位

1. 4. 6 IWDG 锁定寄存器 (IWDG_LOCK)

		Address offset:		100 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															LOCK

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x0	保留, 保持复位值
LOCK	0	R	0x1	IWDG 寄存器保护状态位 0: IWDG 寄存器处于未保护状态 1: IWDG 寄存器处于保护状态 对 IWDG_LOCK 寄存器写入 0x1ACCE551, 被保护的寄存器处于未保护状态; 写入其它值, 处于保护状态

第4章 窗口看门狗（WWDG）

1.1 WWDG 简介

窗口看门狗 WWDG，对于过早或过晚喂狗都将产生 WWDG 复位，可用于检测软件没有喂狗或在禁止喂狗区内喂狗行为，防止程序跑至不可控状态。

1.2 WWDG 主要特性

窗口看门狗 WWDG 功能概述：

- ◆ 支持设定喂狗禁止区
 - 通过 WIN 设置喂狗禁止区
 - WIN 寄存器设定为 11 时，任何区域喂狗不产生复位
 - 喂狗禁止区喂狗产生 WWDG 复位
 - 喂狗禁止区后产生 WWDG 中断
 - WWDG 中断可用作喂狗请求
- ◆ WWDG 溢出长度可设定
 - 可通过 WWDG_RLR 寄存器设定
 - 溢出时产生 WWDG 复位
- ◆ WWDG 中断可用作喂狗请求

1.3 WWDG 功能描述

对于 WWDG 看门狗，过早或过晚喂狗都将产生 WWDG 复位，可用于检测软件的过晚或过早行为，防止程序跑至不可控状态。如中断异常，程序不断进入一个带喂狗指令的子程序的情况，可通过 WWDG 复位消除。

用户可根据程序正常执行的时间设定喂狗窗口，可检测程序未按正常次序执行，跳过某些程序的异常情况。当选用 PCLK 作为 WWDG 时钟时，可精确地检测异常情况。

上电复位后，窗口看门狗不启动，需通过软件配置使能窗口看门狗。软件配置 WWDG_CR 寄存器的 EN 为 1 使能窗口看门狗后，WWDG 计数器载入 WWDG_RLR 寄存器值的 1/4，开始递减计数，当计数到 0 时，窗口计数器加“1”，并在下一个计数时钟到来时，计数器再次载入 WWDG_RLR 寄存器值的 1/4，并继续递减计数。若 WIN 设置为 25%，则窗口计数器为 1 时，WWDG 产生中断标志；若 WIN 设置为 50%，则窗口计数器为 2 时，WWDG 产生中断标志；若 WIN 设置为 75%，则窗口计数器为 3 时，WWDG 产生中断标志。WWDG 产生中断后，直至窗口计数器计数到 4（即 WWDG 累计计数等于 WWDG_RLR）之前，没有在相应的喂狗窗口期进行喂狗动作，则 WWDG 模块将产生复位信号，如下图所示。配置 CLKS，可以选择计数时钟源；配置 WWDG_RLR 寄存器，可设置计数初值；读取 WWDG_VAL，可得到 WWDG 当前计数值。写入装载值寄存器 WWDG_RLR 时，计数器当前值寄存器将被载入 WWDG_RLR 寄存器值的 1/4（前提条件是看门狗要使能）。

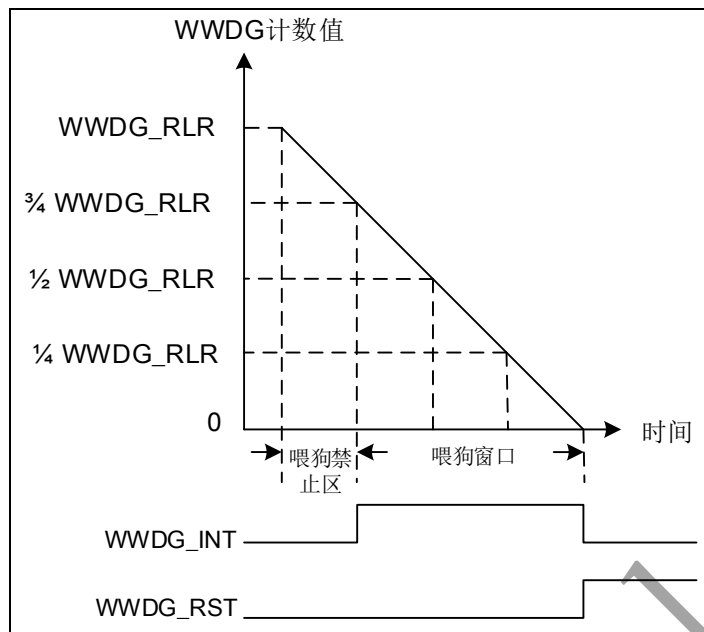


图 4-1 窗口看门狗中断和溢出复位产生时序图 (WIN 设定为 25%)

在喂狗窗口期写 WWDG_ICR, WWDG 将重新载入计数初值, 并重新进行递减计数。在喂狗窗口期外 (如时序图中的喂狗禁止区), 写 WWDG_ICR 会导致产生 WWDG 复位, 如下图所示。喂狗窗口期的定义参考 WWDG_CR 的 WIN 控制位。

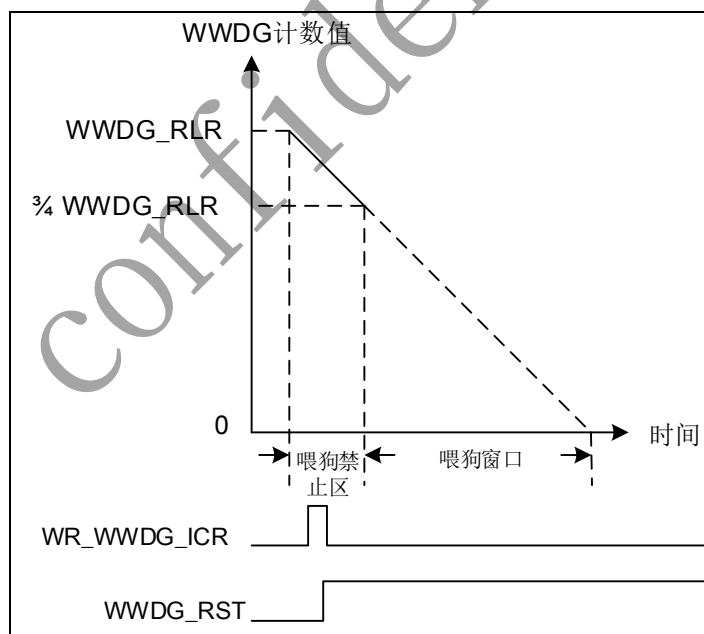


图 4-2 错误的喂狗时序图 (WWDGWIN 设定为 25%)

WWDG 模块支持寄存器写保护, 配置寄存器 WWDG_LOCK=0x1ACCE551, 可去除 WWDG 寄存器的写保护状态, 进行被保护寄存器的写操作, 否则无法对被保护寄存器写入。

1.4 WWDG 寄存器描述

1.4.1 WWDG 重装载寄存器 (WWDG_RLR)

		Address offset:		00 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RLR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RLR[15:0]															

Name	Bit	Type	Reset value	Description
RLR	31:0	W	0x0	WWDG 重装载值 计数范围 0x0000_0001~0xFFFF_FFFF。如果为 0，WWDG 不计数。

1.4.2 WWDG 计数值寄存器 (WWDG_VAL)

		Address offset:		04 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VAL[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VAL[15:0]															

Name	Bit	Type	Reset value	Description
VAL	31:0	R	0x0	WWDG 计数值 读取时返回 WWDG 计数器的当前计数值，其中高两位为窗口计数器当前值

1.4.3 WWDG 控制寄存器 (WWDG_CR)

		Address offset:		08 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										WIN		CLKS	RSTEN	IE	EN

Name	Bit	Type	Reset value	Description

Reserved	31:6	N/A	0x00	保留，保持复位值
WIN	5:4	R/W	0x0	WWDG 禁止喂狗窗口选择位 00: 25%窗口内禁止喂狗，窗口内喂狗产生复位 01: 50%窗口内禁止喂狗，窗口内喂狗产生复位 10: 75%窗口内禁止喂狗，窗口内喂狗产生复位 11: 不禁止喂狗，喂狗将使看门狗计数器重载
CLKS	3	R/W	0x0	WWDG 计数时钟选择位 0: PCLK 1: LSI 时钟（约 32KHz）
RSTEN	2	R/W	0x0	WWDG 复位使能位 0: 禁止 1: 使能，WWDG 计数到 0 时，产生复位信号，将芯片复位
IE	1	R/W	0x0	WWDG 中断使能位 0: 禁止 1: 使能，WWDG 计数到 0 时，产生中断标志
EN	0	R/W	0x0	WWDG 模块使能位 0: 禁止 1: 使能

1.4.4 WWDG 中断清除寄存器 (WWDG_ICR)

Address offset:		0C _H													
Reset value:		00000000_00000000_00000000_00000000 _B													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ICR[31:16]															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ICR[15:0]															

Name	Bit	Type	Reset value	Description
ICR	31:0	W	0x0	WWDG 中断标志清 0 位 对 WWDG_ICR 寄存器进行任意写操作，WWDG 中断标志位均被清零，计数器重载 WWDG_LOAD 寄存器值，继续递减计数

1.4.5 WWDG 中断标志寄存器 (WWDG_RIF)

Address offset:		10 _H													
Reset value:		00000000_00000000_00000000_00000000 _B													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															RIF

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x0	保留，保持复位值
RIF	0	R	0x0	WWDG 中断标志位 0: 未产生中断

1: WWDG 计数器计数到 0, 产生中断
写寄存器 WWDG_ICR, 可清除 WWDG 中断标志位

1. 4. 6 WWDG 锁定寄存器 (WWDG_LOCK)

		Address offset:		100 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															LOCK

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x0	保留, 保持复位值
LOCK	0	R	0x1	WWDG 寄存器保护状态位 0: WWDG 寄存器处于未保护状态 1: WWDG 寄存器处于保护状态 对 WWDG_LOCK 寄存器写入 0x1ACCE551, 被保护的寄存器处于未保护状态; 写入其它值, 处于保护状态

第5章 通用和复用功能 IO（GPIO 和 AFIO）

1.1 GPIO 简介

每个通用 GPIO 端口包含 16 个独立的引脚。这些引脚可单独配置为输入或输出。每个引脚可额外地可配置为开漏输出或带滤波输入模式，配置为输出时可选择每个引脚的驱动强度。

GPIO 引脚可复用为外设功能端口，例如 PWM 输出口或 UART 通信口，每个外设均支持复用到多个引脚上。GPIO 端口支持最多 16 个异步外部中断，可被配置到任何一个 IO 引脚上。并且，GPIO 端口支持通过 PIS 触发其他外设。

1.2 GPIO 主要特性

- ◆ 可配置为输入或输出
- ◆ 输出模式可配置
 - 推挽/开漏
 - 上拉/下拉
- ◆ 输入模式
 - 端口浮空
 - 上拉/下拉
 - 模拟端口
- ◆ 支持端口输出数据的复位/置位，可按位操作
- ◆ 支持复用为外设功能端口
- ◆ 输出驱动能力可配置：四种驱动能力选择
- ◆ 支持 16 个外部输入中断
- ◆ 支持端口配置写保护功能

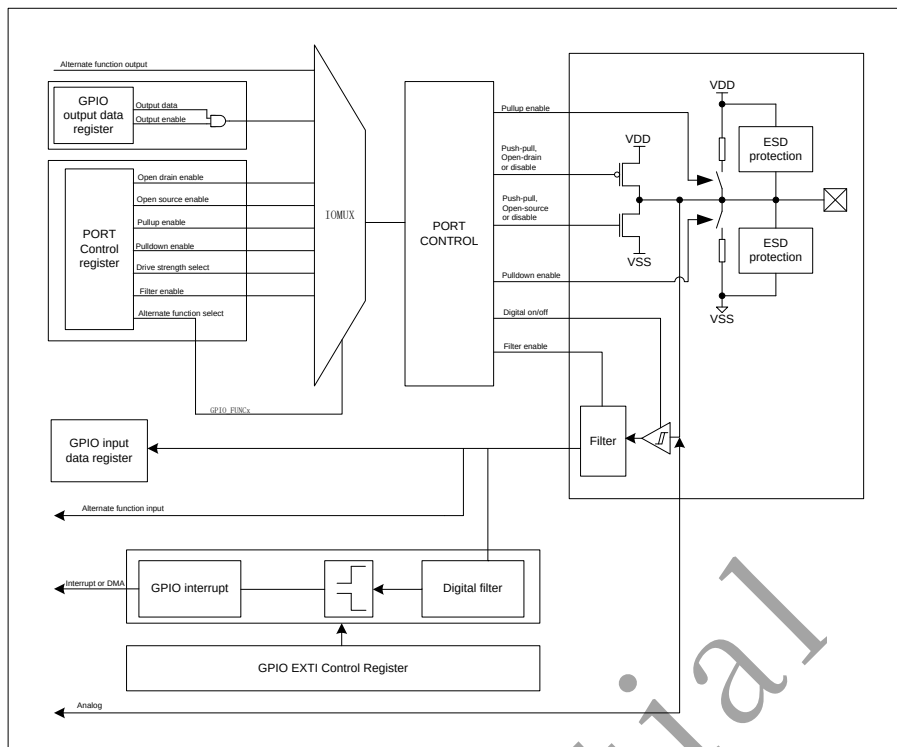


图 5-1 GPIO 结构框图

1.3 GPIO 功能描述

1.3.1 GPIO 端口控制

通过配置 GPIO_MDR，可选择相应端口的模式，可配置为 GPIO 工作模式，模拟工作模式，功能复用模式和测试工作模式。

通过配置 GPIO_PUDR，可使能相应端口的上拉或/和下拉电阻。

通过配置 GPIO_OTR，可将相应端口输出方式配置为推挽和开漏两种模式。在配置为开漏模式时，可使能相应端口的上拉或下拉，以保证正常输出，也可在端口外部连接上拉或下拉电阻。

通过配置 GPIO_DSR，可选择相应端口的输出驱动能力，已满足不同的负载要求。

通过配置 GPIO_FLTR，可使能相应端口的输入滤波功能，可滤除外部引线上高频信号干扰或毛刺。若输入需要较高的实时性，建议关闭输入滤波功能。

通过配置 GPIO_LCKR，可锁定相应端口的控制寄存器数值。直到下一次 CPU 复位锁定才可被解除。端口数据寄存器不受锁定的控制。

1.3.2 GPIO 端口数据

软件可通过读取 GPIO_IDR 才获知端口的电平状态，若相应端口输入滤波被使能，则读到的是端口滤波滤波之后的状态。

通过配置 GPIO_ODR，可选择端口输出电平值，若端口模式已配置为输出，则该值所对应的电平会在管脚上立即生效。

通过配置 GPIO_BSRR，可按位改写端口输出电平值。若同时将某位置位和复位，则置位的优先级更高。

通过配置 GPIO_BSRR，可按位改写端口输出电平值。对置位寄存器某些位进行写入 1 可置位相应端口，写入 0 的位不会影响相应端口的输出电平。对复位寄存器某些位进行写入 1 可复位相应端口，写入 0 的位不会影响相应端口的输出电平。若同时将某位置位和复位，则置位的优先级更高。

1.3.3 GPIO 外部端口中断

通过配置 GPIO_ERTS，可使能外部输入端口上升沿触发。通过配置 GPIO_EFTS 来使能外部输入端口下降沿触发。

通过配置 GPIO_ESWI，可将寄存器 GPIO_ERIF 的中断标志位置位。通过配置 GPIO_EICR，可将 GPIO_ERIF 的中断标志位清除。

外部中断的使能控制由寄存器 GPIO_EIER 和 GPIO_EIDR 控制。

通过配置 GPIO_EICFG0 和 GPIO_EICFG1 可选择外部端口中断的端口，每个外部中断可选择 2~3 个不同的 GPIO 口。

通过配置 GPIO_EDB 可使能相应外部端口中断滤波功能，配置 GPIO_EDBC 中 PRE 和 CNT 位可选滤波时间。

1.3.4 GPIO 端口功能复用

ALT 序号	16 进制	ALT_FUNC	ALT 序号	16 进制	ALT_FUNC
0	0	Debug_IO	32	20	adtim1_ch1n
1	1	Debug_CK	33	21	adtim1_ch2
2	2	uart1_ck	34	22	adtim1_ch2n
3	3	uart1_txd	35	23	adtim1_ch3
4	4	uart1_rxd	36	24	adtim1_ch3n
5	5	uart1_ctsn	37	25	adtim1_ch4
6	6	uart1_rtsn	38	26	adtim1_etr
7	7	ant_sw0	39	27	adtim1_bk
8	8	uart2_txd	40	28	gptima1_ch1
9	9	uart2_rxd	41	29	gptima1_ch2
10	A	ant_sw1	42	2A	gptima1_ch3
11	B	ant_sw2	43	2B	gptima1_ch4
12	C	pis_ch0_out	44	2C	gptima1_etr
13	D	pis_ch1_out	45	2D	gptimb1_ch1
14	E	ant_sw3	46	2E	gptimb1_ch2
15	F	ant_sw4	47	2F	gptimb1_ch3
16	10	i2c1_scl	48	30	gptimb1_ch4
17	11	i2c1_sda	49	31	gptimb1_etr
18	12	i2c1_smba	50	32	gptimc1_ch1
19	13	i2c2_scl	51	33	gptimc1_ch1n
20	14	i2c2_sda	52	34	gptimc1_ch2
21	15	ssi_nss0	53	35	gptimc1_bk

22	16	ssi_sck	54	36	lptim_out
23	17	ssi_dq0	55	37	ant_sw5
24	18	ssi_dq1	56	38	pdm_clk
25	19	ssi_dq2	57	39	uart3_txd
26	1A	ssi_dq3	58	3A	uart3_rxd
27	1B	spi2_sck	59	3B	pdm_data0
28	1C	spi2_nss	60	3C	pdm_data1
29	1D	spi2_mosi	61	3D	ant_sw6
30	1E	spi2_miso	62	3E	ssi_nss1
31	1F	adtim1_ch1	63	3F	spi2_i2s

confidential

1.4 GPIO 寄存器描述

1.4.1 GPIO 端口输入数据寄存器 (GPIO_IDR)

		Address offset:		00 _H											
		Reset valueA:		00000000_00000000_00000000_00000000 _B											
		Reset valueB:		00000000_00000000_00000000_00000000 _B											
		Reset valueC:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDR1 5	IDR1 4	IDR1 3	IDR1 2	IDR1 1	IDR1 0	IDR9	IDR8	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
IDR<y>	15:0	R	0x0	GPIO 输入数据 这些位为只读，读出的值为对应 I/O 的端口的状态

1.4.2 GPIO 端口输出数据寄存器 (GPIO_ODR)

		Address offset:		04 _H											
		Reset valueA:		00000000_00000000_00000000_00000000 _B											
		Reset value B:		00000000_00000000_00000000_00000000 _B											
		Reset value C:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODR 15	ODR 14	ODR 13	ODR 12	ODR 11	ODR 10	ODR 9	ODR 8	ODR 7	ODR 6	ODR 5	ODR 4	ODR 3	ODR 2	ODR 1	ODR 0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
ODR<y>	15:0	R/W	0x0	GPIO 输出数据

1.4.3 GPIO 端口置位和复位寄存器 (GPIO_BSRR)

		Address offset:		08 _H											
		Reset value A:		00000000_00000000_00000000_00000000 _B											
		Reset value B:		00000000_00000000_00000000_00000000 _B											
		Reset value C:		00000000_00000000_00000000_00000000 _B											

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

Name	Bit	Type	Reset value	Description
BRR<y>	31:16	W	0x0	GPIO 复位控制位 0: 无操作 1: 相应位复位 注: 如果同时对 GPIO 置位和复位, 置位的优先级更高
BSR<y>	15:0	W	0x0	GPIO 置位控制位 0: 无操作 1: 相应位置位

1. 4. 4 GPIO 端口锁定寄存器 (GPIO_LCKR)

		Address offset:			0C _H										
		Reset value A:			00000000_00000000_00000000_00000000 _B										
		Reset value B:			00000000_00000000_00000000_00000000 _B										
		Reset value C:			00000000_00000000_00000000_00000000 _B										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															LCKK
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LCK15	LCK14	LCK13	LCK12	LCK11	LCK10	LCK9	LCK8	LCK7	LCK6	LCK5	LCK4	LCK3	LCK2	LCK1	LCK0

Name	Bit	Type	Reset value	Description
Reserved	31:17	N/A	0x0	保留, 保持复位值
LCKK	16	R/W	0x0	LCKK: 锁键 注: 检测到写入关键码高 16 位和低 16 位取反, 且 LCKK 值为 0 才可对 LCK<y>进行写操作, LCKK 一旦被置位, 必须等到下一次 CPU 复位才被清除
LCK<y>	15:0	R/W	0x0	LCK<y>: 端口 y 锁定控制位 0: 相应端口未锁定 1: 相应端口锁定 注: 被锁定相应端口只允许改变输出数据。

1. 4. 5 GPIO 端口模式寄存器 (GPIO_MDR)

		Address offset:	10 _H
		Reset value A:	00000000_00000000_00000000_00000000 _B
		Reset value B:	00000000_00000000_00101000_00001010 _B
		Reset value C:	00000000_00000000_00000000_00000000 _B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MDR15		MDR14		MDR13		MDR12		MDR11		MDR10		MDR9		MDR8	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MDR7		MDR6		MDR5		MDR4		MDR3		MDR2		MDR1		MDR0	

Name	Bit	Type	Reset value	Description
MDR<y>	31:0	R/W	0x0	GPIO 模式控制位 00: GPIO 功能模式 01: 模拟工作模式 10: 功能复用模式 11: 测试工作模式

1. 4. 6 GPIO 端口输入使能寄存器 (GPIO_IER)

		Address offset:		14 _H											
		Reset value A:		00000000_00000000_00000000_00000000 _B											
		Reset value B:		00000000_00000000_00000000_00000000 _B											
		Reset value C:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IER15	IER14	IER13	IER12	IER11	IER10	IER9	IER8	IER7	IER6	IER5	IER4	IER3	IER2	IER1	IER0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留, 保持复位值
IER<y>	15:0	R/W	0x0	GPIO 端口输入使能位 0: GPIO 输入关闭 1: GPIO 输入使能

1. 4. 7 GPIO 端口输出使能寄存器 (GPIO_OER)

		Address offset:		18 _H											
		Reset value A:		00000000_00000000_00000000_00000000 _B											
		Reset value B:		00000000_00000000_00000000_00000000 _B											
		Reset value C:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OER15	OER14	OER13	OER12	OER11	OER10	OER9	OER8	OER7	OER6	OER5	OER4	OER3	OER2	OER1	OER0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
OE <y>	15:0	R/W	0x0	GPIO 端口输出使能位 0: GPIO 输出关闭 1: GPIO 输出使能

1. 4. 8 GPIO 端口上下拉寄存器（GPIO_PUDR）

		Address offset:		1C _H											
		Reset value A:		00000000_00000000_00000000_00000000 _B											
		Reset value B:		00000000_00000000_00000000_00000000 _B											
		Reset value C:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PUDR15		PUDR14		PUDR13		PUDR12		PUDR11		PUDR10		PUDR9		PUDR8	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUDR7		PUDR6		PUDR5		PUDR4		PUDR3		PUDR2		PUDR1		PUDR0	

Name	Bit	Type	Reset value	Description
PUDR<y>	31:0	R/W	0x0	GPIO 上拉和下拉控制位 00: 无上拉和下拉 01: 上拉 10: 下拉 11: 无上拉和下拉

1. 4. 9 GPIO 端口输出类型寄存器（GPIO_OTR）

		Address offset:		20 _H											
		Reset value A:		00000000_00000000_00000000_00000000 _B											
		Reset value B:		00000000_00000000_00000000_00000000 _B											
		Reset value C:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OTR15	OTR14	OTR13	OTR12	OTR11	OTR10	OTR9	OTR8	OTR7	OTR6	OTR5	OTR4	OTR3	OTR2	OTR1	OTR0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
OTR<y>	15:0	R/W	0x0	GPIO 输出类型控制位 0: 推挽输出 1: 开漏输出

1. 4. 10 GPIO 端口驱动寄存器 (GPIO_DSR)

		Address offset:		24 _H											
		Reset value A:		00000000_00000000_00000000_00000000 _B											
		Reset value B:		00000000_00000000_00000000_00000000 _B											
		Reset value C:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DSR15		DSR14		DSR13		DSR12		DSR11		DSR10		DSR9		DSR8	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DSR7		DSR6		DSR5		DSR4		DSR3		DSR2		DSR1		DSR0	

Name	Bit	Type	Reset value	Description
DSR<y>	31:0	R/W	0x0	GPIO 输出驱动控制位 00: 1/4 Max Driver 01: 1/4 Max Driver 10: 1/2 Max Driver 11: Max Driver

1. 4. 11 GPIO 端口滤波寄存器 (GPIO_FLTR)

		Address offset:		28 _H											
		Reset value A:		00000000_00000000_00000000_00000000 _B											
		Reset value B:		00000000_00000000_00000000_00000000 _B											
		Reset value C:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLTR 15	FLTR 14	FLTR 13	FLTR 12	FLTR 11	FLTR 10	FLTR 9	FLTR 8	FLTR 7	FLTR 6	FLTR 5	FLTR 4	FLTR 3	FLTR 2	FLTR 1	FLTR 0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留, 保持复位值
FLTR<y>	15:0	R/W	0x0	GPIO 滤波控制位 0: 20ns 输入滤波禁止 1: 20ns 输入滤波使能

1. 4. 12 GPIO 端口模拟使能寄存器 (GPIO_AER)

		Address offset:		2C _H											
		Reset value A:		00000000_00000000_00000000_00000000 _B											
		Reset value B:		00000000_00000000_00000000_00000000 _B											

		Reset value C:		00000000_00000000_00000000_00000000 _h											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IO15_AER		IO14_AER		IO13_AER		IO12_AER		IO11_AER		IO10_AER		IO9_AER		IO8_AER	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IO7_AER		IO6_AER		IO5_AER		IO4_AER		IO3_AER		IO2_AER		IO1_AER		IO0_AER	

Name	Bit	Type	Reset value	Description
IO_AER<y>	31:0	R/W	0x0	GPIO 类型选择位 00: GPIO 模拟引脚禁止 01: GPIO 模拟引脚 1 使能 10: GPIO 模拟引脚 2 使能

1. 4. 13 GPIO 端口复用功能寄存器 0 (GPIO_FUNC0)

		Address offset:		30 _h											
		Reset value A:		11111111_11111111_11111111_11111111 _b											
		Reset value B:		11111111_11111111_00000100_00000011 _b											
		Reset value C:		00000000_00000000_11111111_11111111 _b											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FUNC3								FUNC2							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FUNC1								FUNC0							

Name	Bit	Type	Reset value	Description
FUNC3	31:24	R/W	0x0	GPIO<3>功能复用选择位 请查看 GPIO 功能复用表
FUNC2	23:16	R/W	0x0	GPIO<2>功能复用选择位 请查看 GPIO 功能复用表
FUNC1	15:8	R/W	0x0	GPIO<1>功能复用选择位 请查看 GPIO 功能复用表
FUNC0	7:0	R/W	0x0	GPIO<0>功能复用选择位 请查看 GPIO 功能复用表

1. 4. 14 GPIO 端口复用功能寄存器 1 (GPIO_FUNC1)

		Address offset:		34 _h											
		Reset value A:		11111111_11111111_11111111_11111111 _b											
		Reset value B:		11111111_00000001_00000000_11111111 _b											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FUNC7								FUNC6							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

FUNC5	FUNC4

Name	Bit	Type	Reset value	Description
FUNC7	31:24	R/W	0x0	GPIO<7>功能复用选择位 请查看 GPIO 功能复用表
FUNC6	23:16	R/W	0x0	GPIO<6>功能复用选择位 请查看 GPIO 功能复用表
FUNC5	15:8	R/W	0x0	GPIO<5>功能复用选择位 请查看 GPIO 功能复用表
FUNC4	7:0	R/W	0x0	GPIO<4>功能复用选择位 请查看 GPIO 功能复用表

1. 4. 15 GPIO 端口复用功能寄存器 2（GPIO_FUNC2）

		Address offset:		38_H											
		Reset value A:		11111111_11111111_11111111_11111111 _B											
		Reset value B:		11111111_11111111_11111111_11111111 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FUNC11								FUNC10							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FUNC9								FUNC8							

Name	Bit	Type	Reset value	Description
FUNC11	31:24	R/W	0x0	GPIO<11>功能复用选择位 请查看 GPIO 功能复用表
FUNC10	23:16	R/W	0x0	GPIO<10>功能复用选择位 请查看 GPIO 功能复用表
FUNC9	15:8	R/W	0x0	GPIO<9>功能复用选择位 请查看 GPIO 功能复用表
FUNC8	7:0	R/W	0x0	GPIO<8>功能复用选择位 请查看 GPIO 功能复用表

1. 4. 16 GPIO 端口复用功能寄存器 3（GPIO_FUNC3）

		Address offset:		3C_H											
		Reset value A:		11111111_11111111_11111111_11111111 _B											
		Reset value B:		11111111_11111111_11111111_11111111 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
FUNC15								FUNC14							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FUNC13								FUNC12							

Name	Bit	Type	Reset value	Description
FUNC15	31:24	R/W	0x0	GPIO<15>功能复用选择位 请查看 GPIO 功能复用表
FUNC14	23:16	R/W	0x0	GPIO<14>功能复用选择位 请查看 GPIO 功能复用表
FUNC13	15:8	R/W	0x0	GPIO<13>功能复用选择位 请查看 GPIO 功能复用表
FUNC12	7:0	R/W	0x0	GPIO<12>功能复用选择位 请查看 GPIO 功能复用表

1. 4. 17 GPIO 中断使能寄存器 (GPIO_EIER)

		Address offset:		00 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIER1 5	EIER1 4	EIER1 3	EIER1 2	EIER1 1	EIER1 0	EIER9	EIER8	EIER7	EIER6	EIER5	EIER4	EIER3	EIER2	EIER1	EIER0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
EIER<y>	15:0	W	0x0	中断使能位 0: 禁止 1: 使能

1. 4. 18 GPIO 中断禁止寄存器 (GPIO_EIDR)

		Address offset:		04 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIDR 15	EIDR 14	EIDR 13	EIDR 12	EIDR 11	EIDR 10	EIDR 9	EIDR 8	EIDR 7	EIDR 6	EIDR 5	EIDR 4	EIDR 3	EIDR 2	EIDR 1	EIDR 0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
EIDR<y>	15:0	W	0x0	中断禁止位 0: 禁止 1: 使能

1. 4. 19 GPIO 中断有效状态寄存器 (GPIO_EIVS)

		Address offset:		08 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIVS1 5	EIVS1 4	EIVS1 3	EIVS1 2	EIVS1 1	EIVS1 0	EIVS9	EIVS8	EIVS7	EIVS6	EIVS5	EIVS4	EIVS3	EIVS2	EIVS1	EIVS0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留, 保持复位值
EIVS<y>	15:0	R	0x0	中断有效状态位 0: 中断无效 1: 中断有效

1. 4. 20 GPIO 原始中断标志寄存器 (GPIO_ERIF)

		Address offset:		0C _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERIF1 5	ERIF1 4	ERIF1 3	ERIF1 2	ERIF1 1	ERIF1 0	ERIF9	ERIF8	ERIF7	ERIF6	ERIF5	ERIF4	ERIF3	ERIF2	ERIF1	ERIF0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留, 保持复位值
ERIF<y>	15:0	R	0x0	原始中断标志位 0: 未检测到有效中断 1: 检测到有效中断

1. 4. 21 GPIO 屏蔽中断标志寄存器 (GPIO_EIFM)

		Address offset:		10 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIFM 15	EIFM 14	EIFM 13	EIFM 12	EIFM 11	EIFM 10	EIFM 9	EIFM 8	EIFM 7	EIFM 6	EIFM 5	EIFM 4	EIFM 3	EIFM 2	EIFM 1	EIFM 0

Name	Bit	Type	Reset value	Description
Reserved	31:20	N/A	0x0	保留，保持复位值
EIFM<y>	19:0	R	0x0	屏蔽中断标志位 0: 未屏蔽中断 1: 屏蔽中断

1. 4. 22 GPIO 中断清零寄存器（GPIO_EICR）

		Address offset:		14 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EICR	EICR	EICR	EICR	EICR1	EICR1	EICR	EICR8	EICR	EICR	EICR5	EICR	EICR	EICR2	EICR	EICR0
15	14	13	12	1	0	9		7	6		4	3		1	

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
EICR<y>	15:0	R	0x0	中断标志清零位 0: 无效 1: 中断标志清零

1. 4. 23 GPIO 中断上升沿触发使能寄存器（GPIO_ERTS）

		Address offset:		18 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERTS	ERTS	ERTS	ERTS	ERTS	ERTS	ERTS	ERTS	ERTS	ERTS	ERTS	ERTS	ERTS	ERTS	ERTS	ERTS
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
ERTS<y>	15:0	R/W	0x0	上升沿触发使能位 0: 禁止 1: 使能

1. 4. 24 GPIO 中断下降沿触发使能寄存器（GPIO_EFTS）

		Address offset:		1C _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											

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

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
EFTS<y>	15:0	R/W	0x0	下降沿触发使能位 0: 禁止 1: 使能

1. 4. 25 GPIO 中断软件触发寄存器（GPIO_ESWI）

		Address offset:		20 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ESWI	ESWI	ESWI	ESWI	ESWI	ESWI	ESWI	ESWI	ESWI	ESWI	ESWI	ESWI	ESWI	ESWI	ESWI	ESWI
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
ESWI<y>	15:0	W1	0x0	中断软件触发位 0: 无操作 1: 软件触发

1. 4. 26 GPIO 中断触发 AD 使能寄存器（GPIO_EDTE）

		Address offset:		24 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EDTE	EDTE	EDTE	EDTE	EDTE	EDTE	EDTE	EDTE	EDTE	EDTE	EDTE	EDTE	EDTE	EDTE	EDTE	EDTE
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
EDTE<y>	15:0	W1	0x0	中断触发 AD 使能位 0: 禁止

1. 4. 27 GPIO 中断滤波使能寄存器 (GPIO_EDB)

		Address offset:		28 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EDB1 5	EDB1 4	EDB1 3	EDB1 2	EDB1 1	EDB1 0	EDB9	EDB8	EDB7	EDB6	EDB5	EDB4	EDB3	EDB2	EDB1	EDB0

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留, 保持复位值
EDB<y>	15:0	W1	0x0	中断滤波使能位 0: 禁止 1: 使能

1. 4. 28 GPIO 中断滤波控制寄存器 (GPIO_EDBC)

		Address offset:		2C _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								PRE<15:8>							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRE<7:0>								Reserved		CNT					

Name	Bit	Type	Reset value	Description
Reserved	31:24	N/A	0x00	保留, 保持复位值
PRE	23:8	R/W	0x0	GPIO 端口中断前置滤波选择位
Reserved	7:6	N/A	0x00	保留, 保持复位值
CNT	5:0	R/W	0x0	GPIO 端口中断去抖滤波选择位

1. 4. 29 GPIO 中断端口选择寄存器 0 (GPIO_EICFG0)

		Address offset:		30 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	EICFGS7			Reserved	EICFGS6			Reserved	EICFGS5			Reserved	EICFGS4		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reser ved	EICFGS3	Reser ved	EICFGS2	Reser ved	EICFGS1	Reser ved	EICFGS0

Name	Bit	Type	Reset value	Description
Reserved	31	N/A	0x00	保留，保持复位值
EICFGS7	30:28	R/W	0x0	EXTI<7>端口选择位 000: PA7 001: PB7 other: 无效
Reserved	27	N/A	0x00	保留，保持复位值
EICFGS6	26:24	R/W	0x0	EXTI<6>端口选择位 000: PA6 001: PB6 other: 无效
Reserved	23	N/A	0x00	保留，保持复位值
EICFGS5	22:20	R/W	0x0	EXTI<5>端口选择位 000: PA5 001: PB5 other: 无效
Reserved	19	N/A	0x00	保留，保持复位值
EICFGS4	18:16	R/W	0x0	EXTI<4>端口选择位 000: PA4 001: PB4 other: 无效
Reserved	15	N/A	0x00	保留，保持复位值
EICFGS3	14:12	R/W	0x0	EXTI<3>端口选择位 000: PA3 001: PB3 other: 无效
Reserved	11	N/A	0x00	保留，保持复位值
EICFGS2	10:8	R/W	0x0	EXTI<2>端口选择位 000: PA2 001: PB2 other: 无效
Reserved	7	N/A	0x00	保留，保持复位值
EICFGS1	6:4	R/W	0x0	EXTI<1>端口选择位 000: PA1 001: PB1 010: PC1 other: 无效
Reserved	3	N/A	0x00	保留，保持复位值
EICFGS0	2:0	R/W	0x0	EXTI<0>端口选择位 000: PA0 001: PB0 010: PC0 other: 无效

1. 4. 30 GPIO 中断端口选择寄存器 1 (GPIO_EICFG1)

		Address offset:		34 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	EICFGS15			Reserved	EICFGS14			Reserved	EICFGS13			Reserved	EICFGS12		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	EICFGS11			Reserved	EICFGS10			Reserved	EICFGS9			Reserved	EICFGS8		

Name	Bit	Type	Reset value	Description
Reserved	31	N/A	0x00	保留，保持复位值
EICFGS15	30:28	R/W	0x0	EXTI<15>端口选择位 000: PA15 001: PB15 other: 无效
Reserved	27	N/A	0x00	保留，保持复位值
EICFGS14	26:24	R/W	0x0	EXTI<14>端口选择位 000: PA14 001: PB14 other: 无效
Reserved	23	N/A	0x00	保留，保持复位值
EICFGS13	22:20	R/W	0x0	EXTI<13>端口选择位 000: PA13 001: PB13 other: 无效
Reserved	19	N/A	0x00	保留，保持复位值
EICFGS12	18:16	R/W	0x0	EXTI<12>端口选择位 000: PA12 001: PB12 other: 无效
Reserved	15	N/A	0x00	保留，保持复位值
EICFGS11	14:12	R/W	0x0	EXTI<11>端口选择位 000: PA11 001: PB11 other: 无效
Reserved	11	N/A	0x00	保留，保持复位值
EICFGS10	10:8	R/W	0x0	EXTI<10>端口选择位 000: PA10 001: PB10 other: 无效
Reserved	7	N/A	0x00	保留，保持复位值
EICFGS9	6:4	R/W	0x0	EXTI<9>端口选择位 000: PA9 001: PB9 other: 无效
Reserved	3	N/A	0x00	保留，保持复位值

EICFGS8	2:0	R/W	0x0	EXTI<8>端口选择位 000: PA8 001: PB8 other: 无效

confidential

第6章 外设互联（PIS）

1.1 PIS 简介

PIS 在微控制器中作为外设互联的桥接口使用，利用 PIS 可实现外设之间的相互触发，控制及自动化工作，提高系统的实时性和快速响应能力，可避免占用过多的 CPU 资源并简化软件工作，为各种应用提供便捷。

1.2 PIS 主要特性

- ◆ 最多支持 8 个 PIS 通道选择
- ◆ 支持同步和异步通道选择
- ◆ 支持信号有效边缘选择
- ◆ 支持通道输出到管脚

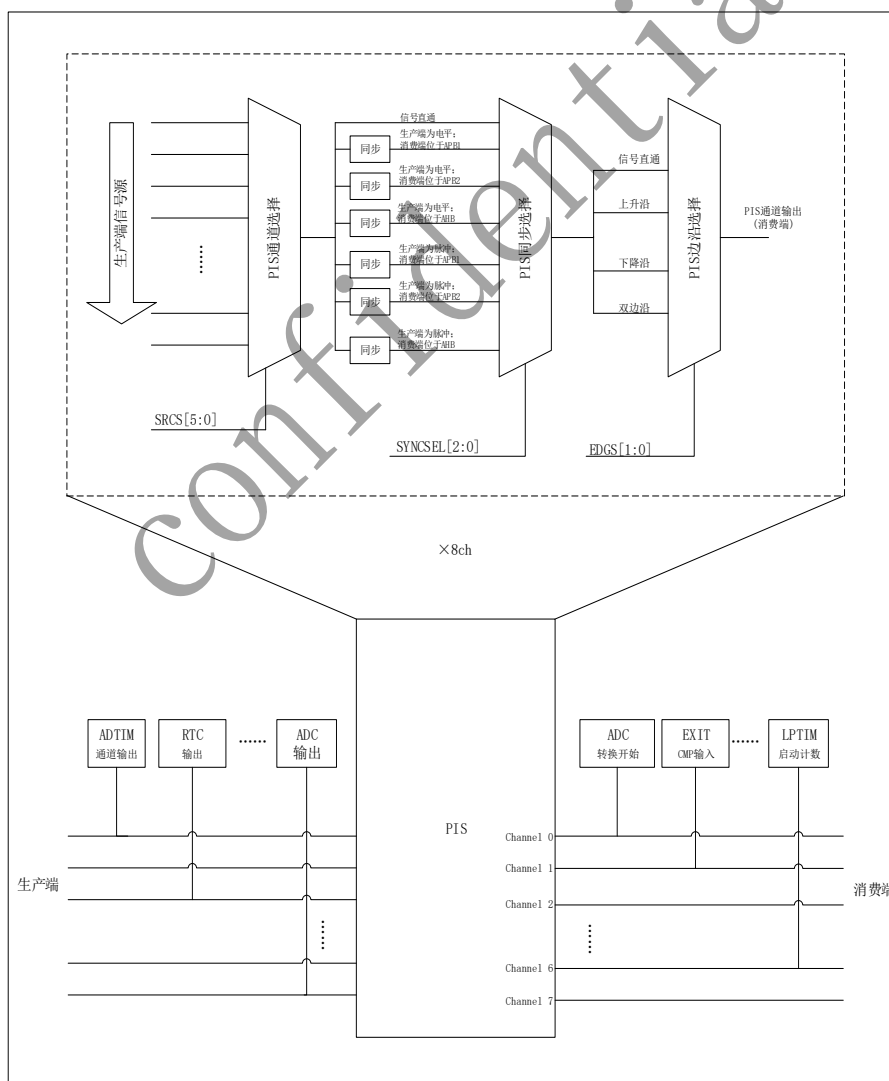


图 6-1 PIS 结构框图

1.3 PIS 功能描述

外设互联可支持 8 个通道资源，每个通道均可对生产端信号进行多路复用。针对不同应用可灵活配置。

1.3.1 PIS 通道选择

PIS 的源端定义为生产端信号，PIS 的输出信号用于消费端。消费端可根据应用需要，来选择合适的生产端信号，并通过选择同步路径以保证正确采样到生产端信号。PIS 的生产端信号选择由 PIS 通道控制寄存器（PIS_CHxCR, x=0~7）配置。PIS 通道控制寄存器的 SRCS 位用来选择生产端信号，配置 DESC 位则选择消费端信号。

1.3.2 外设之间互连

以 LPTIM 和 ADC 为例，在 LPTIM 使用 PCLK 作为计数时钟的情况下，产生与 PCLK 同步的更新事件去触发 ADC 转换动作。可参照如下配置：

1. 设定寄存器 PIS_CH6CR 的 SRCS 为 100000 选择 LPTIM 为生产端模块。
2. 设定寄存器 PIS_CH6CR 的 DESC 为 000001，选择 ADC 规则转换做为消费端信号。
3. 设定寄存器 PIS_CH6CR 的 SYNCSEL 为 00（信号直通），并将 EDGS 设为 00（不输出边沿），此时 PLCK 可任意设定。
4. 配置 ADC 选择外部触发方式进行规则组转换。

配置 LPTIM 进行计数，产生更新事件后将触发 ADC 开始 AD 转换。

1.4 PIS 寄存器描述

1.4.1 PIS 通道 0 控制寄存器（PIS_CH0CR）

		Address offset:		00h											
		Reset value:		00010000_00000000_00000000_00000000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						SYNCSEL		Reserved						EDGS	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		SRCS						Reserved		DESC					

Name	Bit	Type	Reset value	Description
Reserved	31:26	N/A	0x00	保留，保持复位值
SYNCSEL	25:24	R/W	0x00	信号同步选择： 00:信号直通。 在以下场合可设置信号直通： 在一些低功耗应用的场合需采用异步的电平或脉冲； 生产端信号源与消费端处于同一时钟域，无需同步。 01:生产端为电平信号，消费端位于 APB 时钟域 10:生产端为脉冲信号，消费端位于 APB 时钟域 11:预留
Reserved	24:18	N/A	0x00	保留，保持复位值

EDGS	17:16	R/W	0x00	边沿选择: 在输入信号为电平时使用，低功耗应用场合下的异步通道请设置为 00。 00: 不输出边沿 01: 上升沿 10: 下降沿 11: 双边沿
	15:14	N/A	0x00	保留，保持复位值
Reserved				
				输入源选择 00_0001: 保留 00_0010: 保留 00_0100: ADC_AWD_UP ADC 模拟看门狗上限触发 00_0101: ADC_AWD_DN ADC 模拟看门狗下限触发 00_0110: ADC_READY ADC 就绪触发 00_1000: UART1_RXD UART1 RX 接收完成触发 00_1001: UART2_RXD UART2 RX 接收完成触发 00_1010: UART3_RXD UART3 RX 接收完成触发 00_1100: ADTIM1_TRGO ADTIM1 定时器触发 00_1101: ADTIM1_ADCTRIG 等同 ADTIM1_TRGO 01_0000: GPTIMA1_TRGO GPTIMA1 定时器触发 01_0001: GPTIMA1_ADCTRIG 等同 GPTIMA1_TRGO 01_0100: GPTIMB1_TRGO GPTIMA1 定时器触发 01_0101: GPTIMB1_ADCTRIG 等同 GPTIMB1_TRGO 01_1000: GPTIMC1_TRGO GPTIMC1 定时器触发 01_1001: GPTIMC1_ADCTRIG 等同 GPTIMC1_TRGO 01_1100: BSTIM1_TRIG BSTIM1 定时器触发 01_1101: BSTIM1_ADCTRIG 等同 BSTIM1_TRGO 10_0000: LPTIM_TRGO LPTIM 定时器触发 10_1100: EXTI_GPIO_ADCTRIG GPIO 中断触发 11_0000: LVD_CMP LVD 比较器触发 11_0100: RTC_TRIG RTC 触发 Other: 无输入
SRCS	13:8	R/W	0x00	
Reserved				
	7:6	N/A	0x00	保留，保持复位值
DESC				消费端选择 00_0001: ADC_REG_TRIG ADC 规则组消费 00_0010: ADC_INJ_TRIG ADC 注入组消费 00_0100: ADTIM1_ITR0 ADTIM1 计数器内部触发 0 消费 00_0101: ADTIM1_ITR1 ADTIM1 计数器内部触发 1 消费 00_0110: ADTIM1_ITR2 ADTIM1 计数器内部触发 2 消费 00_0111: ADTIM1_ITR3 ADTIM1 计数器内部触发 3 消费 00_1000: ADTIM1_CMP_IN[0] ADTIM1 比较器输入 0 消费 00_1001: ADTIM1_CMP_IN[1] ADTIM1 比较器输入 1 消费 00_1010: ADTIM1_CMP_IN[2] ADTIM1 比较器输入 2 消费 00_1011: ADTIM1_CMP_IN[3] ADTIM1 比较器输入 3 消费 00_1100: GPTIMA1_ITR0 GPTIMA1 计数器内部触发 0 消费 00_1101: GPTIMA1_ITR1 GPTIMA1 计数器内部触发 1 消费 00_1110: GPTIMA1_ITR2 GPTIMA1 计数器内部触发 2 消费 00_1111: GPTIMA1_ITR3 GPTIMA1 计数器内部触发 3 消费 01_0000: GPTIMB1_ITR0 GPTIMB1 计数器内部触发 0 消费 01_0001: GPTIMB1_ITR1 GPTIMB1 计数器内部触发 1 消费 01_0010: GPTIMB1_ITR2 GPTIMB1 计数器内部触发 2 消费 01_0011: GPTIMB1_ITR3 GPTIMB1 计数器内部触发 3 消费 01_0100: GPTIMC1_ITR0 GPTIMC1 计数器内部触发 0 消费 01_0101: GPTIMC1_ITR1 GPTIMC1 计数器内部触发 1 消费 01_0110: GPTIMC1_ITR2 GPTIMC1 计数器内部触发 2 消费 01_0111: GPTIMC1_ITR3 GPTIMC1 计数器内部触发 3 消费 01_1000: GPTIMC1_CMP_IN[0] GPTIMC1 比较器输入 0 消费
	5:0	R/W	0x0	

01_1001:	GPTIMC1_CMP_IN[1]	GPTIMC1 比较器输入 1 消费
01_1010:	GPTIMC1_CMP_IN[2]	GPTIMC1 比较器输入 2 消费
01_1011:	GPTIMC1_CMP_IN[3]	GPTIMC1 比较器输入 3 消费
01_1100:	LPTIM1_ITR0	LPTIM1 计数器内部触发 0 消费
01_1101:	LPTIM1_ITR1	LPTIM1 计数器内部触发 1 消费
01_1110:	LPTIM1_ITR2	LPTIM1 计数器内部触发 2 消费
01_1111:	LPTIM1_ITR3	LPTIM1 计数器内部触发 3 消费
10_0000:	LPTIM1_ITR4	LPTIM1 计数器内部触发 4 消费
10_0001:	LPTIM1_ITR5	LPTIM1 计数器内部触发 5 消费
10_0010:	LPTIM1_ITR6	LPTIM1 计数器内部触发 6 消费
10_0011:	LPTIM1_ITR7	LPTIM1 计数器内部触发 7 消费
Other:	无效	

1. 4. 2 PIS 通道 1 控制寄存器 (PIS_CH1CR)

		Address offset:		04 _H											
		Reset value:		00010000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						SYNCSEL		Reserved						EDGS	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		SRCS						Reserved		DESC					

Name	Bit	Type	Reset value	Description
Reserved	31:26	N/A	0x00	保留, 保持复位值
SYNCSEL	25:24	R/W	0x00	信号同步选择: 参考 PIS_CH0
Reserved	24:18	N/A	0x00	保留, 保持复位值
EDGS	17:16	R/W	0x00	边沿选择: 参考 PIS_CH0
Reserved	15:14	N/A	0x00	保留, 保持复位值
SRCS	13:8	R/W	0x00	输入源选择 参考 PIS_CH0
Reserved	7:6	N/A	0x00	保留, 保持复位值
DESC	5:0	R/W	0x0	消费端选择 参考 PIS_CH0

1. 4. 3 PIS 通道 2 控制寄存器 (PIS_CH2CR)

		Address offset:		08 _H											
		Reset value:		00010000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						SYNCSEL		Reserved						EDGS	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	SRCS	Reserved	DESC

Name	Bit	Type	Reset value	Description
Reserved	31:26	N/A	0x00	保留，保持复位值
SYNCSSEL	25:24	R/W	0x00	信号同步选择： 参考 PIS_CH0
Reserved	24:18	N/A	0x00	保留，保持复位值
EDGS	17:16	R/W	0x00	边沿选择： 参考 PIS_CH0
Reserved	15:14	N/A	0x00	保留，保持复位值
SRCS	13:8	R/W	0x00	输入源选择 参考 PIS_CH0
Reserved	7:6	N/A	0x00	保留，保持复位值
DESC	5:0	R/W	0x0	消费端选择 参考 PIS_CH0

1. 4. 4 PIS 通道 3 控制寄存器（PIS_CH3CR）

		Address offset:		0C _H											
		Reset value:		00010000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						SYNCSSEL			Reserved						EDGS
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		SRCS						Reserved		DESC					

Name	Bit	Type	Reset value	Description
Reserved	31:26	N/A	0x00	保留，保持复位值
SYNCSSEL	25:24	R/W	0x00	信号同步选择： 参考 PIS_CH0
Reserved	24:18	N/A	0x00	保留，保持复位值
EDGS	17:16	R/W	0x00	边沿选择： 参考 PIS_CH0
Reserved	15:14	N/A	0x00	保留，保持复位值
SRCS	13:8	R/W	0x00	输入源选择 参考 PIS_CH0
Reserved	7:6	N/A	0x00	保留，保持复位值
DESC	5:0	R/W	0x0	消费端选择 参考 PIS_CH0

1. 4. 5 PIS 通道 4 控制寄存器（PIS_CH4CR）

		Address offset:		10 _H											
		Reset value:		00010000_00000000_00000000_00000000 _B											

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

Name	Bit	Type	Reset value	Description
Reserved	31:26	N/A	0x00	保留，保持复位值
SYNCSEL	25:24	R/W	0x00	信号同步选择： 参考 PIS_CH0
Reserved	24:18	N/A	0x00	保留，保持复位值
EDGS	17:16	R/W	0x00	边沿选择： 参考 PIS_CH0
Reserved	15:14	N/A	0x00	保留，保持复位值
SRCS	13:8	R/W	0x00	输入源选择 参考 PIS_CH0
Reserved	7:6	N/A	0x00	保留，保持复位值
DESC	5:0	R/W	0x0	消费端选择 参考 PIS_CH0

1. 4. 6 PIS 通道 5 控制寄存器（PIS_CH5CR）

		Address offset:		14h											
		Reset value:		00010000_00000000_00000000_00000000 _h											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						SYNCSEL		Reserved						EDGS	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		SRCS						Reserved		DESC					

Name	Bit	Type	Reset value	Description
Reserved	31:26	N/A	0x00	保留，保持复位值
SYNCSEL	25:24	R/W	0x00	信号同步选择： 参考 PIS_CH0
Reserved	24:18	N/A	0x00	保留，保持复位值
EDGS	17:16	R/W	0x00	边沿选择： 参考 PIS_CH0
Reserved	15:14	N/A	0x00	保留，保持复位值
SRCS	13:8	R/W	0x00	输入源选择 参考 PIS_CH0
Reserved	7:6	N/A	0x00	保留，保持复位值
DESC	5:0	R/W	0x0	消费端选择 参考 PIS_CH0

1.4.7 PIS 通道 6 控制寄存器 (PIS_CH6CR)

		Address offset:		18 _H											
		Reset value:		00010000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						SYNCSEL		Reserved						EDGS	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		SRCS						Reserved		DESC					

Name	Bit	Type	Reset value	Description
Reserved	31:26	N/A	0x00	保留, 保持复位值
SYNCSEL	25:24	R/W	0x00	信号同步选择: 参考 PIS_CH0
Reserved	24:18	N/A	0x00	保留, 保持复位值
EDGS	17:16	R/W	0x00	边沿选择: 参考 PIS_CH0
Reserved	15:14	N/A	0x00	保留, 保持复位值
SRCS	13:8	R/W	0x00	输入源选择 参考 PIS_CH0
Reserved	7:6	N/A	0x00	保留, 保持复位值
DESC	5:0	R/W	0x0	消费端选择 参考 PIS_CH0

1.4.8 PIS 通道 7 控制寄存器 (PIS_CH7CR)

		Address offset:		1C _H											
		Reset value:		00010000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						SYNCSEL		Reserved						EDGS	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		SRCS						Reserved		DESC					

Name	Bit	Type	Reset value	Description
Reserved	31:26	N/A	0x00	保留, 保持复位值
SYNCSEL	25:24	R/W	0x00	信号同步选择: 参考 PIS_CH0
Reserved	24:18	N/A	0x00	保留, 保持复位值
EDGS	17:16	R/W	0x00	边沿选择: 参考 PIS_CH0

Reserved	15:14	N/A	0x00	保留，保持复位值
SRCS	13:8	R/W	0x00	输入源选择 参考 PIS_CH0
Reserved	7:6	N/A	0x00	保留，保持复位值
DESC	5:0	R/W	0x0	消费端选择 参考 PIS_CH0

1.4.9 PIS 通道端口输出使能寄存器（PIS_CHOER）

		Address offset:		40 _H											
		Reset value:		00010000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														CH1 OE	CH0 OE

Name	Bit	Type	Reset value	Description
Reserved	31:2	N/A	0x00	保留，保持复位值
CH1OE	1	R/W	0x00	PIS 通道 1 输出至端口使能 0: 输出到端口禁止 1: 输出到端口使能
CH0OE	0	R/W	0x0	PIS 通道 0 输出至端口使能 0: 输出到端口禁止 1: 输出到端口使能

第7章 椭圆曲线加密（ECC）

1.1 ECC 简介

椭圆曲线加密算法（Elliptic Curve Cryptography），简称 ECC，是基于椭圆曲线数学理论实现的一种非对称加密算法。相比 RSA，ECC 优势是可以使用更短的密钥，来实现与 RSA 相当或更高的安全。

1.2 ECC 功能描述

一般情况下，椭圆曲线可用下列方程式来表示，其中 a, b, c, d 为系数。

$$E: y^2 = ax^3 + bx^2 + cx + d$$

例如，当 $a=1, b=0, c=-2, d=4$ 时，所得到的椭圆曲线为：

$$E: y^2 = x^3 - 2x + 4$$

该椭圆曲线 E 的图像如图 X-1 所示，可以看出根本就不是椭圆形。

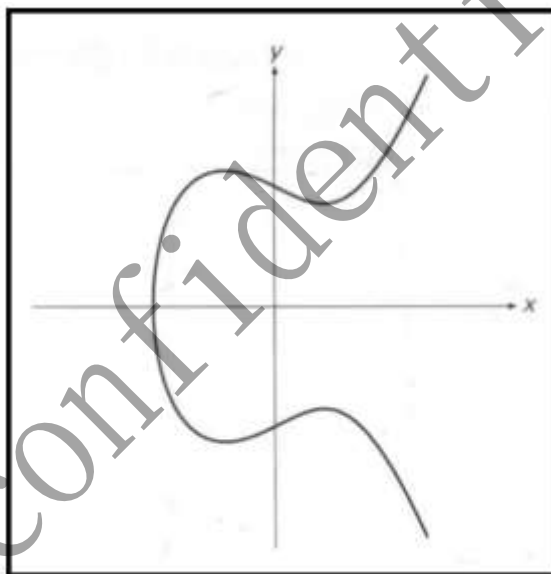
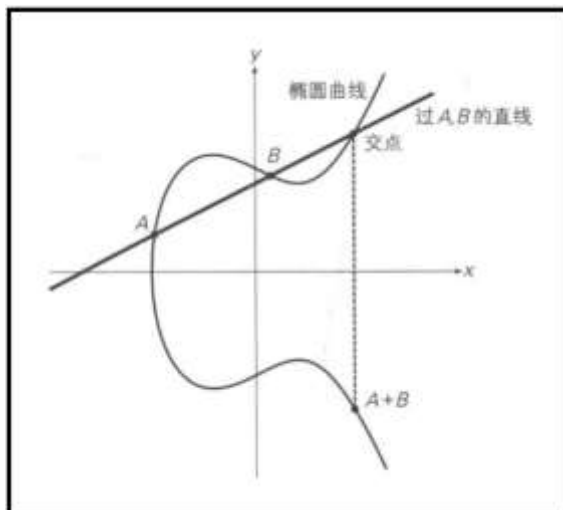


图 7-1 椭圆曲线示例（ $E: y^2 = x^3 - 2x + 4$ ）

定义椭圆曲线的运算规则

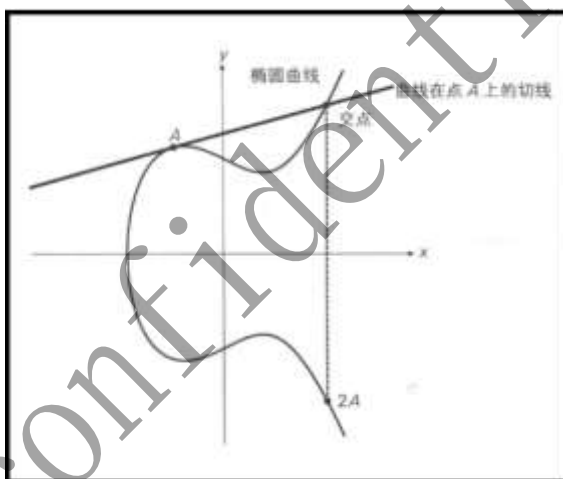
加法

过曲线上的两点 A、B 画一条直线，找到直线与椭圆曲线的交点，交点关于 x 轴对称位置的点，定义为 $A+B$ ，即为加法。如下图所示： $A + B = C$



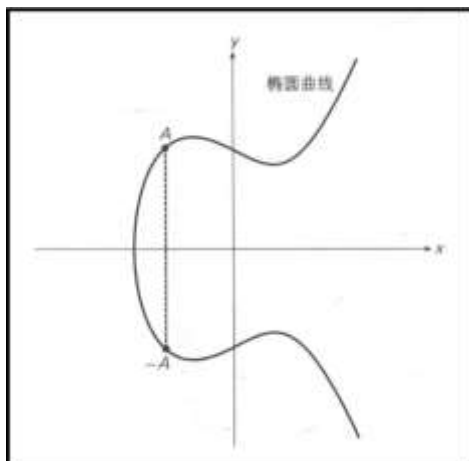
二倍运算

上述方法无法解释 $A + A$ ，即两点重合的情况。因此在这种情况下，将椭圆曲线在 A 点的切线，与椭圆曲线的交点，交点关于 x 轴对称位置的点，定义为 $A + A$ ，即 $2A$ ，即为二倍运算。



正负取反

将 A 关于 x 轴对称位置的点定义为 $-A$ ，即椭圆曲线的正负取反运算。如下图所示：



无穷远点

如果将 A 与 $-A$ 相加，过 A 与 $-A$ 的直线平行于 y 轴，可以认为直线与椭圆曲线相交于无穷远点。

综上，定义了 $A+B$ 、 $2A$ 运算，因此给定椭圆曲线的某一点 G ，可以求出 $2G$ 、 $3G$ （即 $G + 2G$ ）、 $4G$。即：当给定 G 点时，已知 x ，求 xG 点并不困难。反之，已知 xG 点，求 x 则非常困难。此即为椭圆曲线加密算法背后的数学原理。

1.2.1 有限域上的椭圆曲线运算

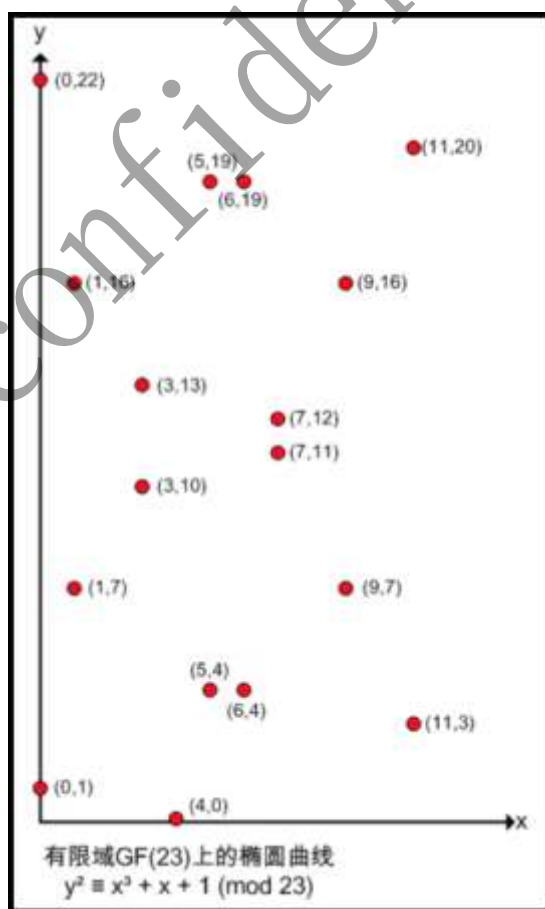
椭圆曲线要形成一条光滑的曲线，要求 x, y 取值均为实数，即实数域上的椭圆曲线。但椭圆曲线加密算法，并非使用实数域，而是使用有限域。按数论定义，有限域 $GF(p)$ 指给定某个质数 p ，由 $0, 1, 2, \dots, p-1$ 共 p 个元素组成的整数集中定义的加减乘除运算。

假设椭圆曲线为 $y^2 = x^3 + x + 1$ ，其在有限域 $GF(23)$ 上时，写作： $y^2 \equiv x^3 + x + 1 \pmod{23}$

此时，椭圆曲线不再是一条光滑曲线，而是一些不连续的点，如下图所示。以点 $(1,7)$ 为例， $7^2 \equiv 1^3 + 1 + 1 \pmod{23}$ 。如此还有如下点：

$(0,1)$ $(0,22)$ $(1,7)$ $(1,16)$ $(3,10)$ $(3,13)$ $(4,0)$ $(5,4)$ $(5,19)$ $(6,4)$ $(6,19)$
 $(7,11)$ $(7,12)$ $(9,7)$ $(9,16)$ $(11,3)$ $(11,20)$ 等等。

另外，如果 $P(x,y)$ 为椭圆曲线上的点，则 $-P$ 即 $(x,-y)$ 也为椭圆曲线上的点。如点 $P(0,1)$ ， $-P=(0,-1)=(0,22)$ 也为椭圆曲线上的点。



1. 2. 1. 1 计算 xG

相关公式如下：有限域 $GF(p)$ 上的椭圆曲线 $y^2 = x^3 + ax + b$ ，若 $P(X_p, Y_p), Q(X_q, Y_q)$ ，且 $P \neq Q$ ，则 $R(X_r, Y_r) = P + Q$ 由如下规则确定：

$$X_r = (\lambda^2 - X_p - X_q) \bmod p$$

$$Y_r = (\lambda(X_p - X_r) - Y_p) \bmod p$$

其中 $\lambda = (Y_q - Y_p)/(X_q - X_p) \bmod p$ (若 $P \neq Q$)， $\lambda = (3X_p^2 + a)/2Y_p \bmod p$ (若 $P = Q$)

因此，有限域 $GF(23)$ 上的椭圆曲线 $y^2 \equiv x^3 + x + 1 \pmod{23}$ ，假设以 $(0,1)$ 为 G 点，计算 $2G$ 、 $3G$ 、 $4G \dots xG$ 等等，方法如下：

计算 $2G$ ：

$$\lambda = (3 \times 0^2 + 1)/2 \times 1 \bmod 23 = (1/2) \bmod 23 = 12$$

$$X_r = (12^2 - 0 - 0) \bmod 23 = 6$$

$$Y_r = (12(0 - 6) - 1) \bmod 23 = 19$$

即 $2G$ 为点 $(6,19)$

计算 $3G$ ： $3G = G + 2G$ ，即 $(0,1) + (6,19)$

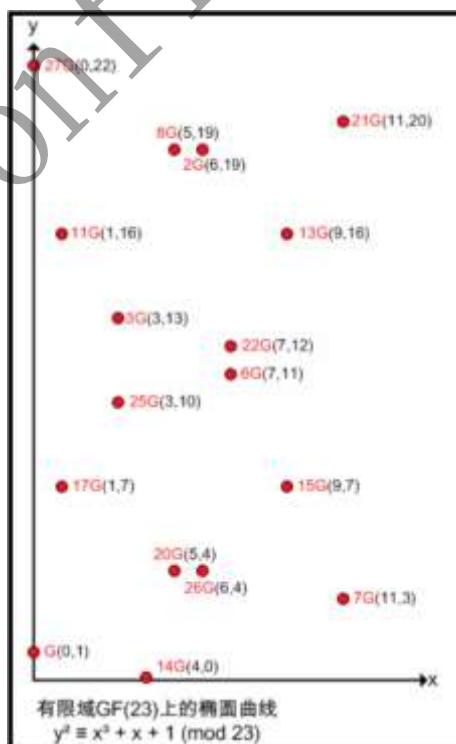
$$\lambda = (19 - 1)/(6 - 0) \bmod 23 = 3$$

$$X_r = (3^2 - 0 - 6) \bmod 23 = 3$$

$$Y_r = (3(0 - 3) - 1) \bmod 23 = 13$$

即 $3G$ 为点 $(3, 13)$

同理计算 $4G$ 、 $5G \dots xG$ ，分布如下图：



1.2.2 椭圆曲线加解密算法原理

建立基于椭圆曲线的加密机制，需要找到类似 RSA 质因子分解或其他求离散对数这样的难题。而椭圆曲线上的已知 G 和 xG 求 x ，是非常困难的，此即为椭圆曲线上的离散对数问题。此处 x 即为私钥， xG 即为公钥。

椭圆曲线加密算法原理如下：

设私钥、公钥分别为 k 、 K ，即 $K = kG$ ，其中 G 为 G 点。

公钥加密：选择随机数 r ，将消息 M 生成密文 C ，该密文是一个点对，即：

$$C = \{rG, M+rK\}, \text{ 其中 } K \text{ 为公钥}$$

私钥解密：

$$M + rK - k(rG) = M + r(kG) - k(rG) = M \quad \text{其中 } k、K \text{ 分别为私钥、公钥。}$$

1.2.3 椭圆曲线签名算法原理

椭圆曲线签名算法，即 ECDSA。设私钥、公钥分别为 k 、 K ，即 $K = kG$ ，其中 G 为 G 点。

私钥签名：

- 1、选择随机数 r ，计算点 $rG(x, y)$ 。
- 2、根据随机数 r 、消息 M 的哈希 h 、私钥 k ，计算 $s = (h + kx)/r$ 。
- 3、将消息 M 、和签名 $\{rG, s\}$ 发给接收方。

公钥验证签名：

- 1、接收方收到消息 M ，以及签名 $\{rG=(x,y), s\}$ 。
- 2、根据消息求哈希 h 。
- 3、使用发送方公钥 K 计算： $hG/s + xK/s$ ，并与 rG 比较，如相等即验签成功。

原理如下：

$$hG/s + xK/s = hG/s + x(kG)/s = (h+xk)G/s = r(h+xk)G / (h+kx) = rG$$

1.2.3.1 签名过程

假设要签名的消息是一个字符串：“Hello World!”。DSA 签名的第一个步骤是待签名的消息生成一个消息摘要。不同的签名算法使用不同的消息摘要算法。而 ECDSA256 使用 SHA256 生成 256 比特的摘要。

摘要生成结束后，应用签名算法对摘要进行签名：

产生一个随机数 k

利用随机数 k ，计算出两个大数 r 和 s 。将 r 和 s 拼在一起就构成了对消息摘要的签名。

这里需要注意的是，因为随机数 k 的存在，对于同一条消息，使用同一个算法，产生的签名是不一样的。从函数的角度来理解，签名函数对同样的输入会产生不同的输出。因为函数内部会将随机值混入签名的过程。

第8章 加密处理（AES/DES）

1.1 AES/DES 简介

硬件加密模块主要用于由硬件对数据进行加密或解密操作，支持的标准有 AES、DES。

AES（Advanced Encryption Standard）是最新的分组对称密码算法，兼容联邦信息处理标准出版物（FIPSPUB197，2001 年 11 月 26 日）规定的高级加密标准（AES）。

DES（Data Encryption Standard）是应用非常广泛的对称密码算法，兼容联邦信息处理标准出版物（FIPS PUB 46-3，1999 年 10 月 25 日）规定的加密标准（DES）和三重 DES（TDES），遵循美国国家标准协会（ANSI）X9.52 标准。

1.2 AES/DES 主要特性

- ◆ 适用于 AES、DES 和 TDES 加密和解密操作
- ◆ AES
 - 支持 128、192 和 256 位的密钥
 - 支持 ECB、CBC 模式
 - 支持 1、8、16 或 32 位数据交换
- ◆ DES/TDES
 - 支持 ECB、CBC 模式
 - 支持 64、128 和 192 位密钥
 - 支持在 CBC 模式下使用的 2×32 位初始化向量（IV）
 - 支持 1、8、16 或 32 位数据交换
- ◆ 支持直接存储器访问（DMA）（用于传入数据和读出已处理数据）
- ◆ 支持产生 CPU 中断请求

1.3 DES 功能描述

1.3.1 DES/TDES-ECB 模式加密

下图介绍 DES 和 TDES 电子密码本（DES/TDES-ECB）模式中的加密。64 位明文数据块（P）经过位/字节/半字交换后作为输入块（I）。输入块通过 DEA 在加密状态下使用 K1 进行加密处理。上述处理过程的输出会直接反馈到 DEA 的输入，在解密状态下使用 K2 执行 DES。上述处理过程的输出会直接反馈到 DEA 的输入，在加密状态下使用 K3 执行 DES。生成的 64 位输出块（O）在执行位/字节/半字交换之后，以密文（C）形式输出到数据寄存器。

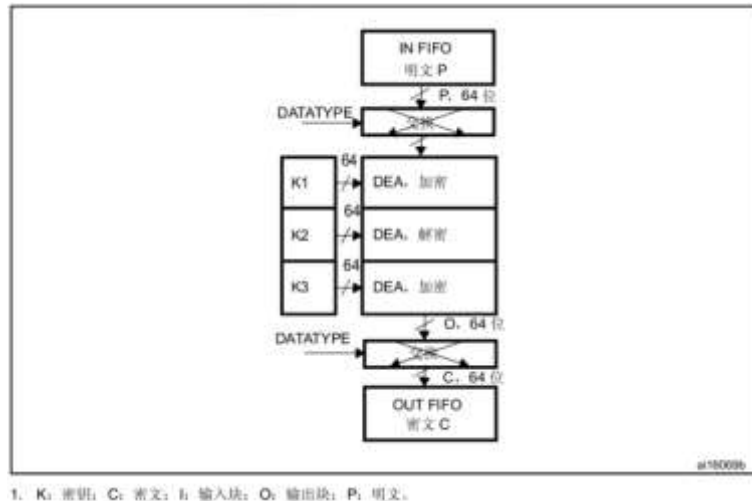


图 8-1 DES/TDES-ECB 模式加密

1.3.2 DES/TDES-ECB 模式解密

下图介绍了 DES/TDES-ECB 解密。64 位方块 (C) 经过位/字节/半字交换后，作为输入块 (I)。该密钥序列与加密过程中使用的密钥序列相反。输入块通过 DEA 在解密状态下使用 K3 进行解密处理。上述处理过程的输出会直接反馈到 DEA 的输入，在加密状态下使用 K2 执行 DES。新结果会直接反馈到 DEA 的输入，在解密状态下使用 K1 执行 DES。生成的 64 位输出块 (O) 在进行位/字节/半字交换后，产生明文 (P)。

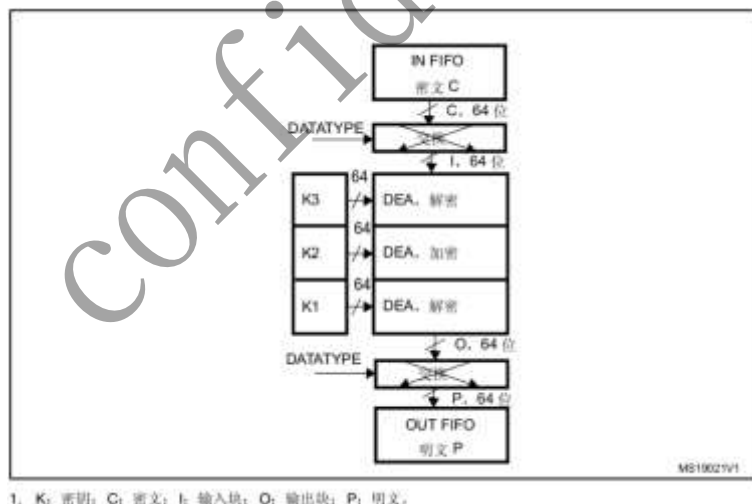


图 8-2 DES/TDES-ECB 模式解密

1.3.3 DES/TDES-CBC 模式加密

下图介绍了 DES 和三重 TDES 密码分组链接 (DES/TDES-CBC) 模式加密。该模式首先将明文消息分成多个 64 位数据块。在 TCBC 加密中，执行位/字节/半字交换后获得的第一个输入块 (I1) 是通过一个 64 位初始化向量 IV 与第一个明文数据块 (P1) 进行异或运算形成的 ($I1 = IV \oplus P1$)。输入块通过 DEA 在加密状态下使用 K1 进行加密处理。上述处理过程的输出会直接反馈到 DEA 的输入，在解密状态下使用

K2 执行 DES。上述处理过程的输出会直接反馈到 DEA 的输入，在加密状态下使用 K3 执行 DES。生成的 64 位输出块（O1）将直接用作密文（C1），即 $C1 = O1$ 。然后，第一个密文块与第二个明文数据块进行异或运算，从而生成第二个输入块（I2） $= (C1 \wedge P2)$ 。注意，此时的 I2 和 P2 指的是第二个块。第二个输入块通过 TDEA 处理而生成第二个密文块。此加密处理会不断将后续密文块和明文块链接到一起，直到消息中最后一个明文块得到加密为止。如果消息中包含的数据块数不是整数，则应按照应用程序指定的方式对最后的不完整数据块进行加密。

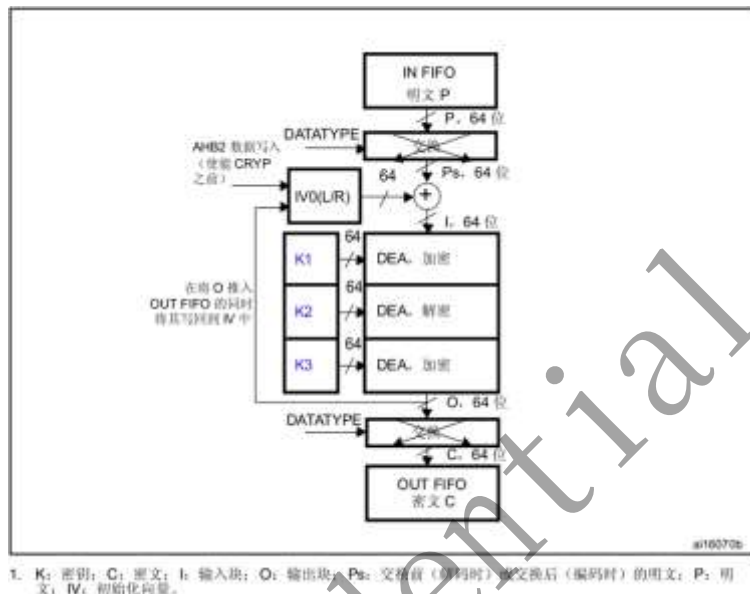


图 8-3 DES/TDES-CBC 模式加密

1. 3. 4 DES/TDES-CBC 模式解密

在 DES/TDES-CBC 解密（如下图）中，第一个密文块（C1）将直接用作输入块（I1）。该密钥序列与加密过程中使用的密钥序列相反。输入块通过 DEA 在解密状态下使用 K3 进行解密处理。上述处理过程的输出会直接反馈到 DEA 的输入，在解密状态下使用 K2 执行 DES。生成值会直接反馈到 DEA 的输入，在解密状态下使用 K1 处理 DES。生成的输出块与 IV（必须与加密期间使用的相同）进行异或运算，从而生成第一个明文块（ $P1 = O1 \wedge IV$ ）。然后，第二个密文块将用作下一个输入块，并由 TDEA 进行处理。生成的输出块与第一个密文块进行异或运算，从而生成第二个明文数据块（ $P2 = O2 \wedge C1$ ）。（注意，P2 和 O2 指的是第二个数据块）。TCBC 解密过程将以此方式继续进行，直到最后一个完整密文块得到解密为止。必须按照应用程序指定的方式对不完整数据块密文进行解密。

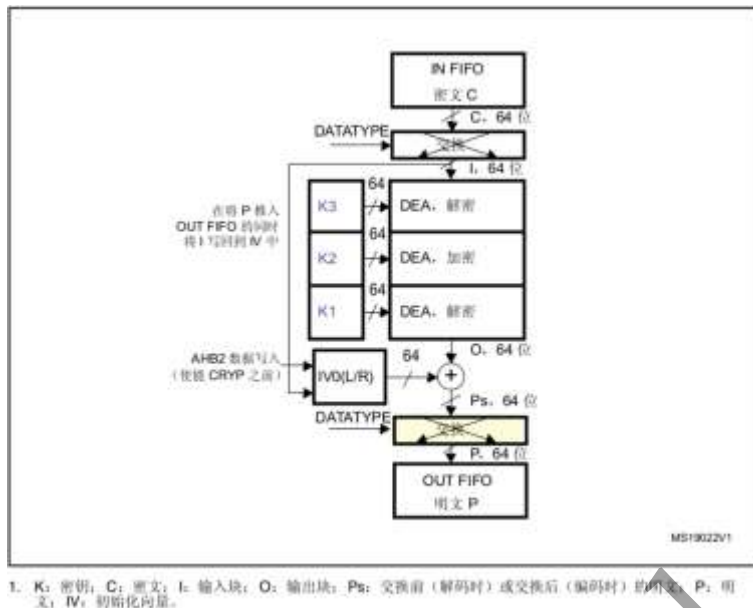


图 8-4 DES/TDES-CBC 模式解密

1.4 AES 功能描述

1.4.1 AES-ECB 模式加密

下图介绍了 AES 电子密码本 (AES-ECB) 模式加密。

在 AES-ECB 加密中, 执行位/字节/半字交换后, 128 位明文数据块 (P) 将用作输入块 (I)。输入块通过 AEA 在加密状态下使用 128 位、192 位或 256 位密钥进行处理。执行位/字节/半字交换后, 生成的 128 位输出块 (O) 将用作密文 (C)。该密文随后给到结果寄存器。

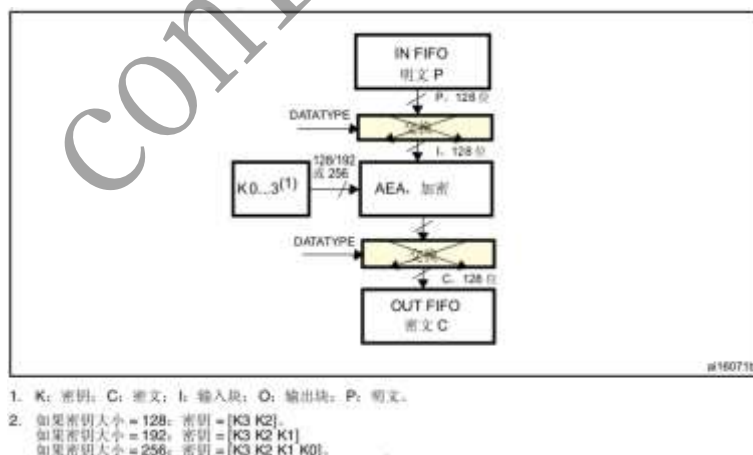


图 8-5 AES-ECB 模式加密

1.4.2 AES-ECB 模式解密

下图介绍了 AES 电子密码本 (AES-ECB) 模式加密。

要在 ECB 模式下执行 AES 解密, 需要准备密钥 (需要针对加密执行完整的密钥计划), 方法为: 收集最后一个轮密钥并将其用作解密密文的第一个轮密钥。该准备过

程通过 AES 内核计算完成。

在 AES-ECB 解密中，执行位/字节/半字交换后，128 位密文块（C）将用作输入块（I）。该密钥序列与加密处理中的密钥序列相反。执行位/字节/半字交换，生成的 128 位输出块（O）将产生明文（P）。

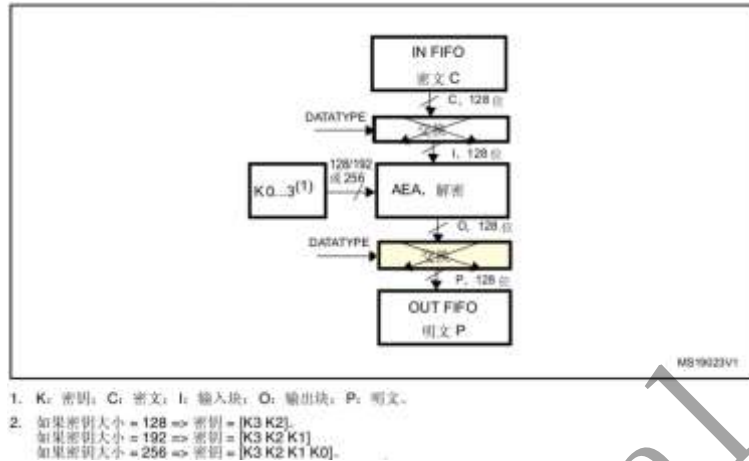


图 8-6 AES-ECB 模式解密

1.4.3 AES-CBC 模式解密

AES 加密分组链接（AES-CBC）模式解密如下图所示。

在 AES-CBC 加密中，执行位/字节/半字交换后所获得的第一个输入块（I1）是通过一个 128 位初始化向量 IV 与第一个明文数据块（P1）进行异或运算形成的（ $I1 = IV \oplus P1$ ）。输入块通过 AEA 在加密状态下使用 128 位、192 位或 256 位密钥进行处理。生成的 128 位输出块（O1）将直接用作密文（C1），即 $C1 = O1$ 。然后，第一个密文块与第二个明文数据块进行异或运算，从而生成第二个输入块（I2）=（ $C1 \oplus P2$ ）。注意，此时的 I2 和 P2 指的是第二个块。第二个输入块通过 AEA 处理而生成第二个密文块。此加密处理会不断将后续密文块和明文块链接到一起，直到消息中最后一个明文块得到加密为止。如果消息中包含的数据块数不是整数，则应按照应用程序指定的方式对最后的不完整数据块进行加密。

在 CBC 模式下（如 ECB 模式），必须准备密钥才可以执行 AES 解密。

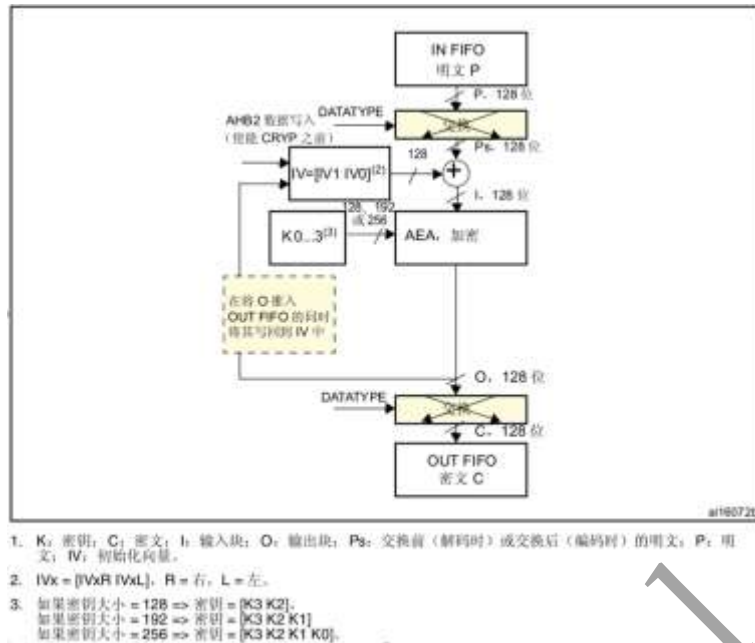


图 8-7 AES-CBC 模式加密

1. 4. 4 AES-CBC 模式解密

在 AES-CBC 解密 (请参见图 210) 中, 第一个 128 位密文块 (C_1) 将直接用作输入块 (I_1)。输入块通过 AEA 在解密状态下使用 128 位、192 位或 256 位密钥进行处理。生成的输出块与 128 位初始化向量 IV (必须与加密期间使用的相同) 进行异或运算, 从而生成第一个明文块 ($P_1 = O_1 \oplus IV$)。然后, 第二个密文块将用作下一个输入块, 并由 AEA 进行处理。生成的输出块与第一个密文块进行异或运算, 从而生成第二个明文数据块 ($P_2 = O_2 \oplus C_1$)。(注意, P_2 和 O_2 指的是第二个数据块。) AES-CBC 解密过程将以此方式继续进行, 直到最后一个完整密文块得到解密为止。必须按照应用程序指定的方式对不完整数据块密文进行解密。

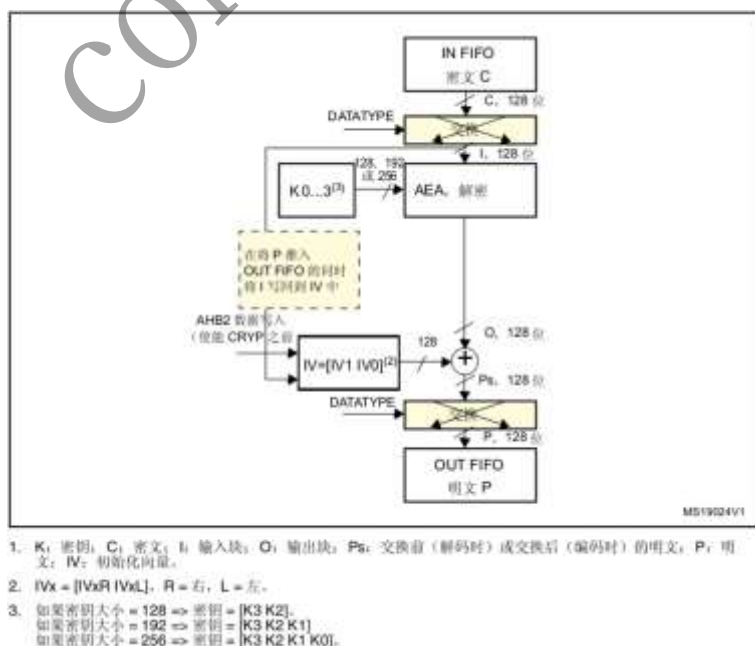


图 8-8 AES-CBC 模式解密

confidential

第9章 真随机发生器（TRNG）

1.1 TRNG 简介

真随机数发生器（TRNG）可生产 1 位串行真随机数或 8/16/32 位并行真随机数

1.2 TRNG 主要特性

- ◆ 支持可编程的随机数位宽
- ◆ 支持可编程的种子值
- ◆ 支持随机性修正模式
- ◆ 支持随机序列错误检测

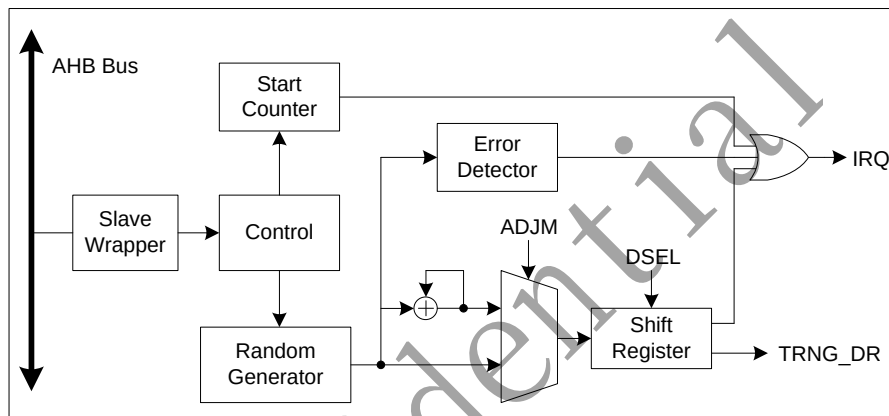


图 9-1 TRNG 结构框图

1.3 TRNG 功能描述

1.3.1 操作流程

要运行 TRNG，请按以下步骤操作：

1. 如果需要，使能中断（TRNG_IE 相应位置 1）。准备好随机数时或出现错误时生成中断。
2. 通过将 TRNG_CR 寄存器中的 TRNGEN 位置 1 使能随机数产生。这会激活随机数生成器、移位寄存器和错误检测器。若果需要随机性更高的数据，可以将 ADJM 位置 1 使能修正模式，但带来随机数生成时间会变长。配置 ADJC 可改变修正模式系数，系数越大，随机性越高，但随机数生成时间也越长。
3. 每次中断时，检查确认未出现错误（TRNG_SR 寄存器中的 SERR 位应为 0，TRNG_IF 寄存器中的 SERR 也为 0），并且随机数已准备就绪（TRNG_SR 寄存器中的 DAVLD 位为 1）。然后即可读取 TRNG_DR 寄存器中的内容。

按照 FIPS PUB（联邦信息处理标准出版物）140-2 的要求，将 TRNGEN 位置 1 后产生的第一个随机数不应使用，但应保存起来，与产生的下一个随机数进行比较。随

后产生的每个随机数都需要与产生的上一个随机数进行比较。如果任何一对进行比较的数字相等，则测试失败（连续随机数发生器测试）。

操作流程示意图如下所示。

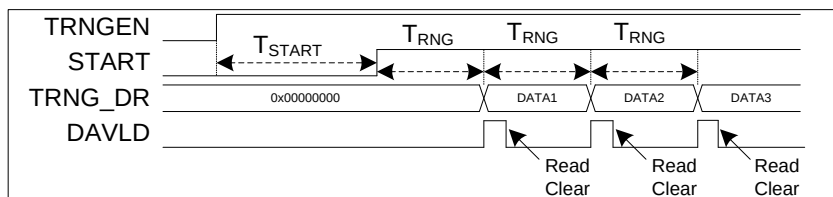


图 9-2 操作流程示意图

1.3.2 初始化时间

TRNG 使能后需要经过一段时间初始化才可产生新的随机数据，初始化时间在 TRNG_START 寄存器配置。根据当前 HCLK 时钟频率，计算出 TRNG_START 的值，使初始化时间不低于 1ms。

计算公式为:

$$T_{START} = T_{HCLK1} \times 2^{TSTART+1}$$

1.3.3 错误管理

出现随机序列错误时，SERR 被硬件置 1，随机数产生被中断。若使能了序列错误中断，则产生该中断。如果 TRNG_DR 寄存器中有新产生的随机数，不能使用该随机数。

应执行以下操作：置位 TRNG_IFCR 寄存器的 SERR 位，将 TRNG_EN 位清零并置 1，以便重新初始化和重新启动 TRNG。

1.3.4 随机数生成时间

随机数生成时间由三者共同决定：TRNG_CR 寄存器中的 DSEL 位、ADJM 位和 ADJC 位。

随机数生成时间公式为:

$$T_{\text{RNG}} \approx 5 \times D \times C^M \text{ (单位 us) (误差范围+66.66\%/-33.33\%)}$$

DSEL	系数 D	ADJM	系数 M	ADJC	系数 C
00	1	0	0	00	2
01	8			01	4
10	16	1	1	10	6
11	32			11	8

表 9-1 随机数生成时间系数表

1.3.5 随机数种子

因为随机序列发生器来自模拟信号，电源和地上的干扰或环境温度的变化等都会导致随机性变差、输出序列的相邻位的自相关系数升高。通过设置一种随机的种子值，可

能使随机序列的随机性能更加优化，降低自相关性。

种子的类型可选择为使用上一次产生的随机数或使用 TRNG_SEED 寄存器值，若使用 TRNG_SEED 寄存器值，则在每次读取新的随机数时同时更新 TRNG_SEED 寄存器。种子的类型也可选择固定为 0 或 1。

Confidential

第10章 运算加速器（CALC）

1.1 CALC 简介

运算加速器（CALC）可以执行平方根和除法的运算加速。

1.2 CALC 主要特性

- ◆ 支持最大 32 位无符号数平方根运算
- ◆ 支持最大 32 位有符号数或无符号数除法运算
- ◆ 支持使用 DMA 写数据执行运算操作

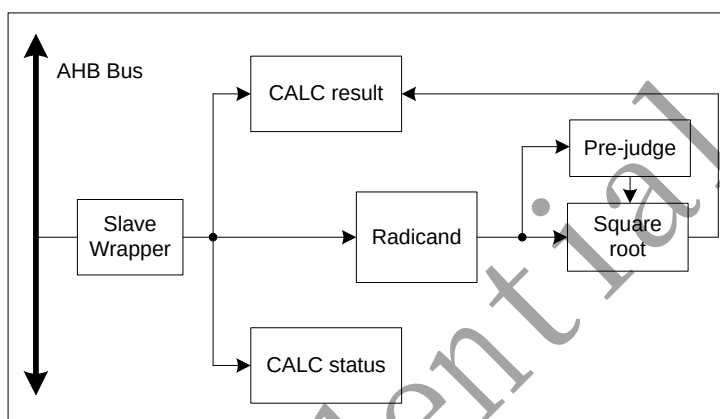


图 10-1 CALC 结构框图

1.3 CALC 功能描述

1.3.1 平方根运算

1.3.1.1 算法概述

无符号数平方根算法可对最大 32 位的无符号数进行平方根运算，运算结果最大为 16 位无符号数。若理论平方根值中含有小数部分，则硬件在计算时会舍去小数，即向 0 的方向取最大整数。

硬件运算电路中带有预判决功能，可根据被开方数 RCND 的数量级，自动选取最小的计算时间。

1.3.1.2 完成时间

平方根算法可根据被开方数 RDCND 中输入数值的进行预判决，硬件针对不同量级自动决定运算时间，不同量级的运算时间可根据下表所示。

RDCND[31:0]	运算时间（BUSY=1 的时钟个数）
1xxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	17
01xx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	16
001x_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	16

0001_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	
0000_1xxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx 0000_01xx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	15
0000_001x_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx 0000_0001_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	14
0000_0000_1xxx_xxxx_xxxx_xxxx_xxxx_xxxx 0000_0000_01xx_xxxx_xxxx_xxxx_xxxx_xxxx	13
0000_0000_001x_xxxx_xxxx_xxxx_xxxx_xxxx 0000_0000_0001_xxxx_xxxx_xxxx_xxxx_xxxx	12
0000_0000_0000_1xxx_xxxx_xxxx_xxxx_xxxx 0000_0000_0000_01xx_xxxx_xxxx_xxxx_xxxx	11
0000_0000_0000_001x_xxxx_xxxx_xxxx_xxxx 0000_0000_0000_0001_xxxx_xxxx_xxxx_xxxx	10
0000_0000_0000_0000_1xxx_xxxx_xxxx_xxxx 0000_0000_0000_0000_01xx_xxxx_xxxx_xxxx	9
0000_0000_0000_0000_001x_xxxx_xxxx_xxxx 0000_0000_0000_0000_0001_xxxx_xxxx_xxxx	8
0000_0000_0000_0000_0000_1xxx_xxxx_xxxx 0000_0000_0000_0000_0000_01xx_xxxx_xxxx	7
0000_0000_0000_0000_0000_001x_xxxx_xxxx 0000_0000_0000_0000_0000_0001_xxxx_xxxx	6
0000_0000_0000_0000_0000_0000_1xxx_xxxx 0000_0000_0000_0000_0000_0000_01xx_xxxx	5
0000_0000_0000_0000_0000_0000_001x_xxxx 0000_0000_0000_0000_0000_0000_0001_xxxx	4
0000_0000_0000_0000_0000_0000_0000_1xxx 0000_0000_0000_0000_0000_0000_0000_01xx	3
0000_0000_0000_0000_0000_0000_0000_00xx	2

表 10-1 平方根运算时间表

1.3.2 除法运算

1.3.2.1 算法概述

除法算法可对最大 32 位的有符号数或无符号数进行平方根运算，运算结果最大为 32 位有符号数或无符号数。有符号运算中 Bit31 位为符号位，负数使用二进制补码的方式表示。

商的符号由被除数和除数共同决定。当被除数和除数符号相同时，商为正数；当被除数和除数符号不同时，商为负数。余数的符号由被除数的符号决定。

硬件运算电路中带有预判决功能，可根据被除数的数量级，自动选取最小的计算时间。

1.3.2.2 特例说明

溢出

在 32 有符号除法运算中，当被除数为 0x8000_0000，除数为 0xFFFF_FFFF 时，即 $-2^{31}/-1$ ，得到的结果应为 2^{31} ，但 32 位有符号数最大可表示的正整数为 $2^{31}-1$ ，该次运算结果将溢出。此时硬件计算所得的商为 0x8000_0000，余数为 0x0000_0000。硬件并无标识位指示运算结果是否为溢出。

除数零

在 32 位有符号数或无符号数除法中，若除数设置为 0，则硬件计算所得的商为 0x0000_0000，余数为 0x0000_0000。该次计算硬件将标志位 DZ 置 1。

1.3.2.3 使用说明

通过设置 SIGN 位来选择运算的 32 数是有符号数还是无符号数。

通过设置 TRM 位来选择触发源。根据操作习惯，一般选择最后一个操作数的写入操作作为除法运算的触发源。

当被除数寄存器 DIVDR 和除数寄存器 DIVSR 写入值后时，除法运算即开始，开始时除法运算标志位 BUSY 被置位。当软件检测到 BUSY 被硬件清零时，表示除法运算已经完成。通过读取商寄存器 DIVQR 和余数寄存器 DIVRR 可获得此次除法运算的结果。

若当运算还未完成，TRM 位所选择的触发动作发生时，硬件会重新开始新的运算，原先的运算结果将被丢弃。

1.3.2.4 完成时间

除法算法可根据被除数 DIVDR 中输入数值的进行预判决，硬件针对不同量级自动决定运算时间，不同量级的运算时间可根据下表所示。

被除数的绝对值 (abs(DIVD))	运算时间 (BUSY=1 的时钟个数)
1xxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	33
01xx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	32
001x_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	31
0001_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	30
0000_1xxx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	29
0000_01xx_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	28
0000_001x_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	27
0000_0001_xxxx_xxxx_xxxx_xxxx_xxxx_xxxx	26
0000_0000_1xxx_xxxx_xxxx_xxxx_xxxx_xxxx	25
0000_0000_01xx_xxxx_xxxx_xxxx_xxxx_xxxx	24
0000_0000_001x_xxxx_xxxx_xxxx_xxxx_xxxx	23
0000_0000_0001_xxxx_xxxx_xxxx_xxxx_xxxx	22
0000_0000_0000_1xxx_xxxx_xxxx_xxxx_xxxx	21
0000_0000_0000_01xx_xxxx_xxxx_xxxx_xxxx	20
0000_0000_0000_001x_xxxx_xxxx_xxxx_xxxx	19
0000_0000_0000_0001_xxxx_xxxx_xxxx_xxxx	18
0000_0000_0000_0000_1xxx_xxxx_xxxx_xxxx	17
0000_0000_0000_0000_01xx_xxxx_xxxx_xxxx	16
0000_0000_0000_0000_001x_xxxx_xxxx_xxxx	15
0000_0000_0000_0000_0001_xxxx_xxxx_xxxx	14
0000_0000_0000_0000_0000_1xxx_xxxx_xxxx	13
0000_0000_0000_0000_0000_01xx_xxxx_xxxx	12
0000_0000_0000_0000_0000_001x_xxxx_xxxx	11
0000_0000_0000_0000_0000_0001_xxxx_xxxx	10
0000_0000_0000_0000_0000_0000_1xxx_xxxx	9

0000_0000_0000_0000_0000_0000_01xx_xxxx	8
0000_0000_0000_0000_0000_0000_001x_xxxx	7
0000_0000_0000_0000_0000_0000_0001_xxxx	6
0000_0000_0000_0000_0000_0000_0000_1xxx	5
0000_0000_0000_0000_0000_0000_0000_01xx	4
0000_0000_0000_0000_0000_0000_0000_001x	3
0000_0000_0000_0000_0000_0000_0000_000x	2

表 10-2 除法运算时间表

confidential

第11章 高级定时器（ADTIM）

1.1 ADTIM 简介

高级控制定时器(ADTIM)由一个 16 位的自动装载计数器组成，它由一个可编程的预分频器驱动。它适合多种用途，包含测量输入信号的脉冲宽度（输入捕获），或者产生输出波形（输出比较、PWM、嵌入死区时间的互补 PWM 等）。

使用定时器预分频和 RCC 时钟控制预分频器，可以实现脉冲宽度和波形周期从几个微秒到几个毫秒的调节。

高级控制定时器(ADTIM)、通用定时器(GPTIMA,GPTIMB,GPTIMC)与基础(BSTIM)定时器是完全独立的，它们不共享任何资源。它们可以同步操作。

1.2 ADTIM 主要特性

- ◆ 16 位递增、递减、递增/递减自动装载计数器。
- ◆ 16 位可编程（可以实时修改）预分频器，分频系数介于 1~65535 之间。
- ◆ 多达 4 个独立通道：
 - 输入捕获
 - 输出比较
 - PWM 生成（边缘和中间对齐模式）
 - 单脉冲模式输出
- ◆ 死区时间可编程的互补输出。
- ◆ 使用外部信号控制定时器和定时器互联的同步电路。
- ◆ 允许在指定数目的技术器周期之后更新定时器寄存器的重复计数器。
- ◆ 刹车输入信号可以将定时器输出信号置于复位状态或者一个已知状态。
- ◆ 发生以下事件时生成中断或 DMA 请求：
 - 更新：计数器上溢/下溢、计数器初始化（通过软件或内部/外部触发）
 - 触发事件（计数器启动、停止、初始化或通过内部/外部触发计数）
 - 输入捕获
 - 输出比较
 - 刹车信号输入
- ◆ 支持用于定位用的增量（正交）编码器和霍尔传感器电路。
- ◆ 触发输入作为外部时钟或者按周期的电流管理。

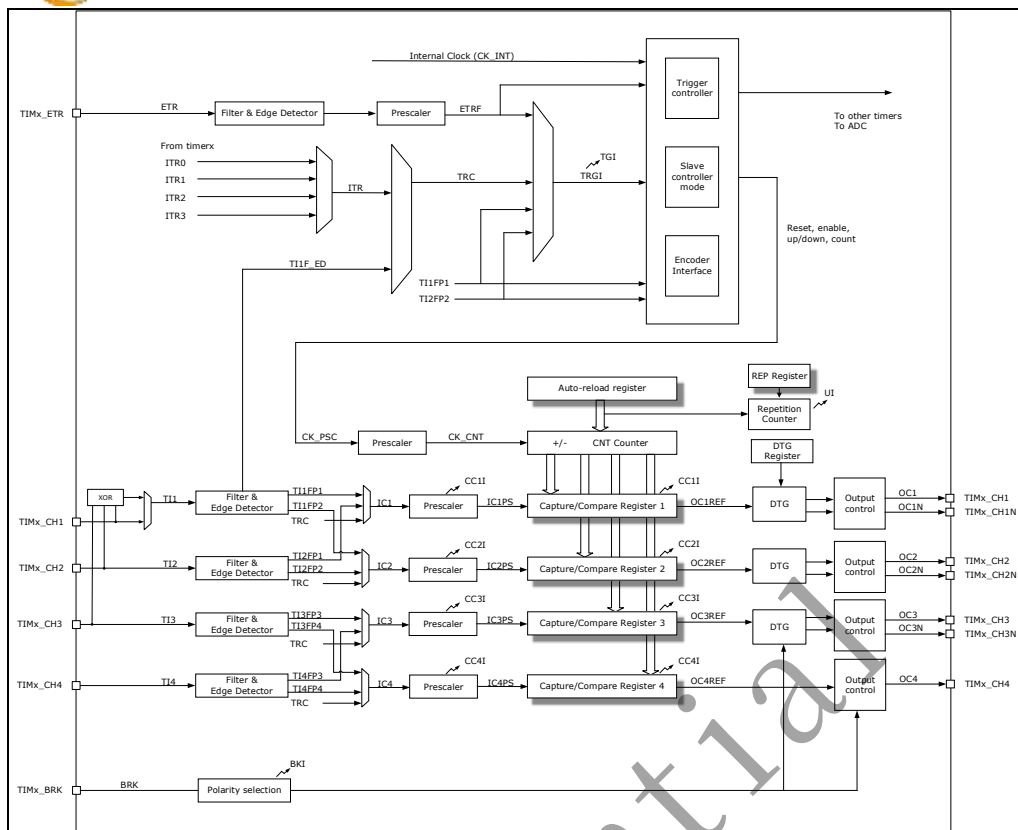


图 11-1 ADTIM 结构框图

1.3 ADTIM 功能描述

1.3.1 时基单位

可编程高级控制定时器的主要部分是一个 16 位计数器和与其相关的自动装载寄存器。这个计数器可以向上计数、向下计数或者向上向下双向计数。此计数器时钟由预分频器分频得到。计数器、自动装载寄存器和预分频器寄存器可以由软件读写，即使计数器还在运行读写仍然有效。

时基单位包含:

- ◆ 计数器寄存器(ADTIM_CNT)
- ◆ 预分频寄存器(ADTIM_PSC)
- ◆ 自动重载寄存器(ADTIM_ARR)
- ◆ 重复计数寄存器(ADTIM_RCR)

自动装载寄存器是预先装载的，写或读自动重载寄存器将访问预装载寄存器。根据在 ADTIM_CR1 寄存器中的自动装载预装载使能位(ARPE)的设置，预装载寄存器的内容被立即或在每次的更新事件 UEV 时传送到影子寄存器。当计数器达到溢出条件(或向下计数时的下溢条件)并当 ADTIM_CR1 寄存器中的 UDIS 位等于 0 时，产生更新事件。更新事件也可以由软件产生。随后会详细描述每一种配置下更新事件的产生。

计数器由预分频器的时钟输出 CK_CNT 驱动，仅当设置了计数器 ADTIM_CR1 寄存器中的计数器使能位(CEN)时，CK_CNT 才有效。

注意，在设置了 ADTIM_CR1 寄存器的 CEN 位的一个时钟周期后，计数器开始计数。

预分频器描述

预分频器可以将计数器的时钟频率按 1 到 65536 之间的任意值分频。它是基于一个(在 ADTIM_PSC 寄存器中的)16 位寄存器控制的 16 位计数器。因为这个控制寄存器带有缓冲器，它能够在运行时被改变。新的预分频器的参数在下次更新事件到来时被采用。预分频计数器仅能向下计数。预分频器被设定后，一旦预分频计数器到达 0，计数器则会向上或向下计数，此外计数器会随着时钟一同更新。图 11-2 与图 11-3 给出了在运行时更改预分频器因子，计数器的动作的例子。

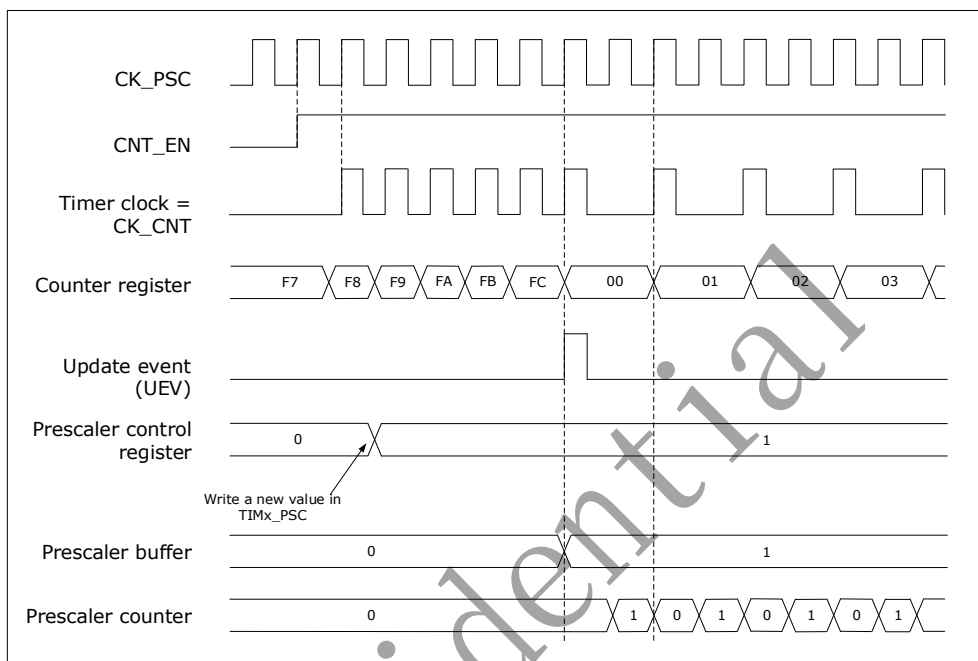


图 11-2 预分频器的参数由 1 变为 2 之计数器时序图

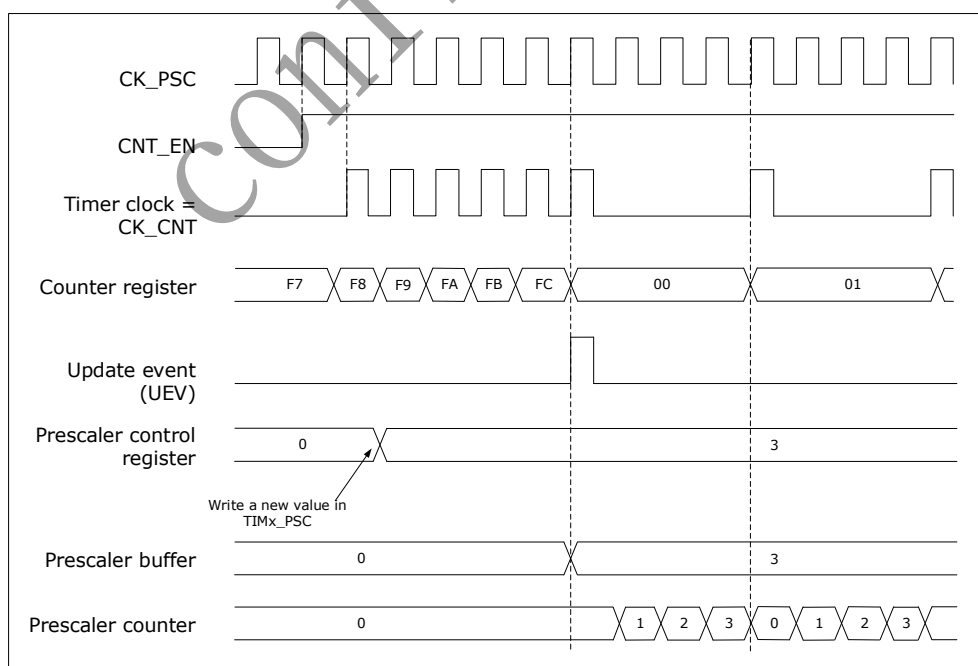


图 11-3 预分频器的参数由 1 变为 4 之计数器时序图

1.3.2 计数器模式

递增计数模式

在递增计数模式，计数器从 0 计数到自动加载值(ADTIM_ARR 寄存器内容)，然后重新从 0 开始计数并且产生一个计数器上溢事件。

如果使用了重复计数器功能，在递增计数达到设置的重复计数次数(ADTIM_RCR)时，才产生更新事件（UEV）；否则每次计数器溢出时都会产生更新事件。

在 ADTIM_EGR 寄存器中(通过软件方式或者使用从模式控制器)设置 UG 位也同样可以产生一个更新事件。

通过软件设置 ADTIM_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清’0’之前，将不产生更新事件。但是在应该产生更新事件时，计数器会被清’0’，同时预分频器的计数也被清 0(但预分频器的数值不变)。此外，如果设置了 ADTIM_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，同时(依据 URS 位)设置更新标志位 (ADTIM_RIF 寄存器中的 UIF 位)。

- ◆ 重复计数器被重新加载为 ADTIM_RCR 寄存器内容。
- ◆ 自动装载影子寄存器被重新设为预装载寄存器的值(ADTIM_ARR)
- ◆ 预分频器的缓冲区被写入预装载寄存器的值(ADTIM_PSC 寄存器的内容)

下列图表为计数器在 ADTIM_ARR=0x36 时计数器在不同时钟频率下的行为范例：

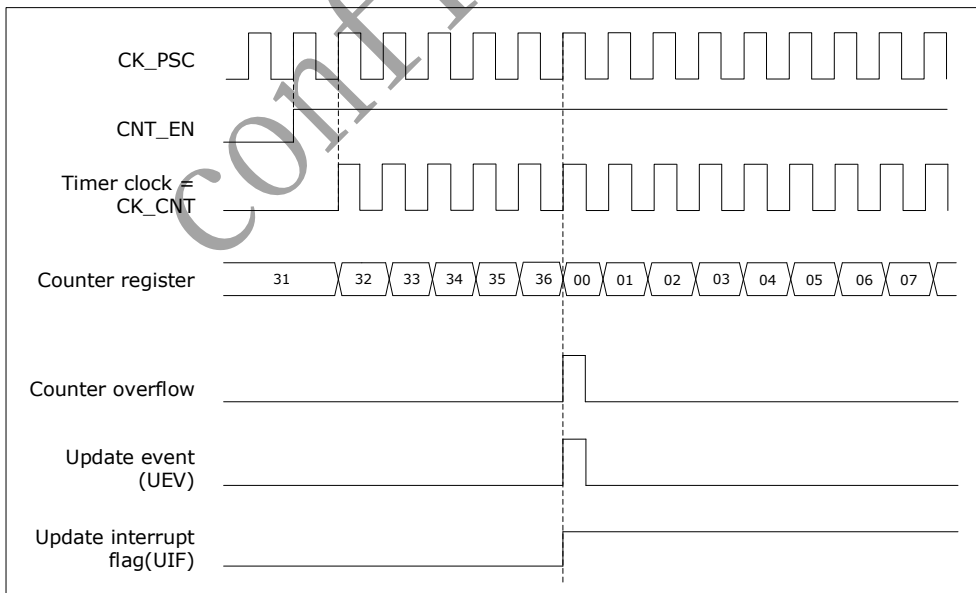


图 11-4 计数器时序图，内部时钟分频因子为 1

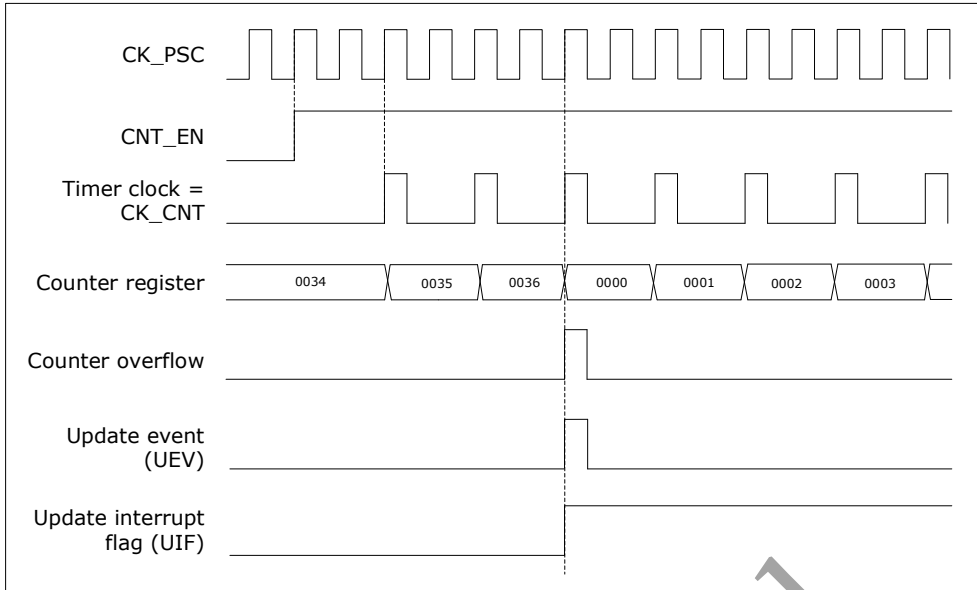


图 11-5 计数器时序图，内部时钟分频因子为 2

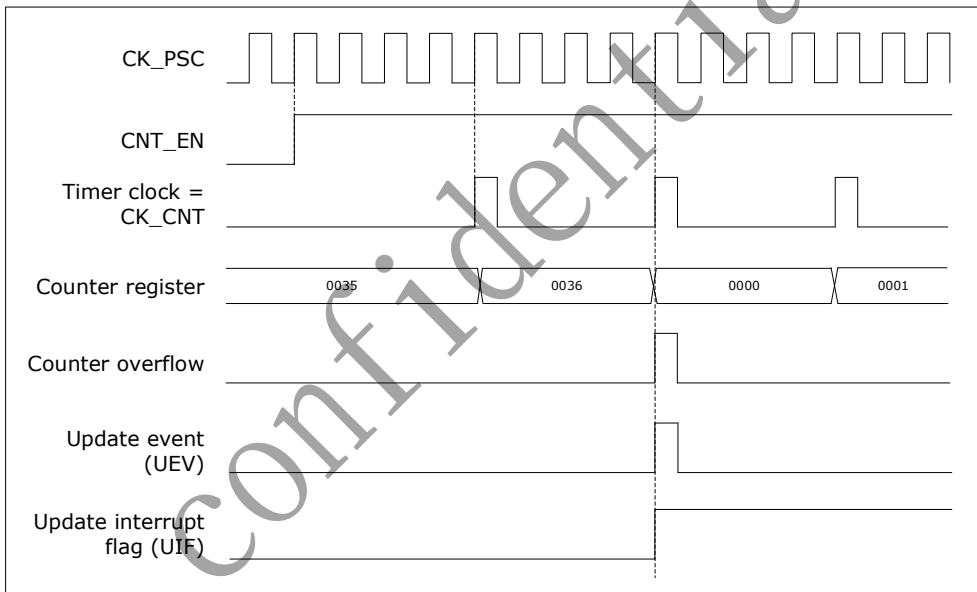


图 11-6 计数器时序图，内部时钟分频因子为 4

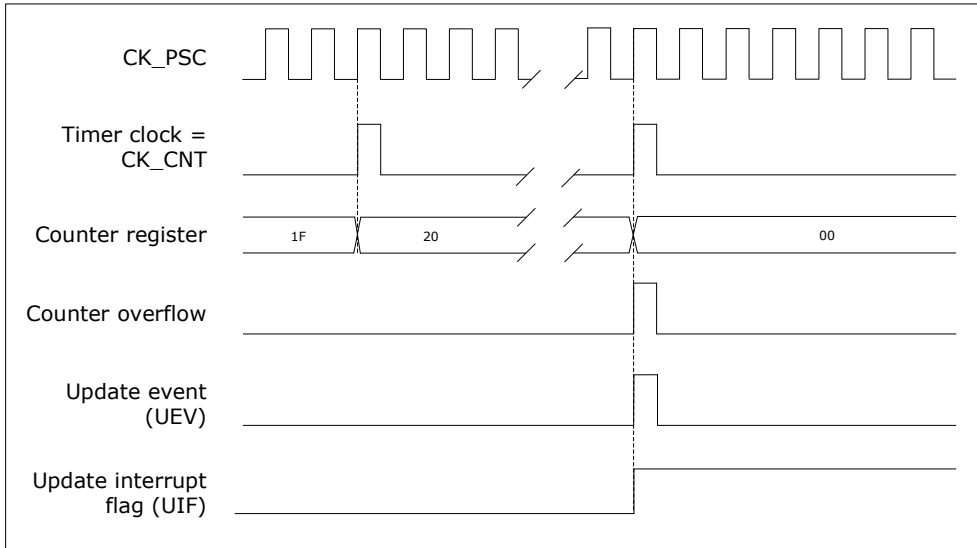


图 11-7 计数器时序图，内部时钟分频因子为 N

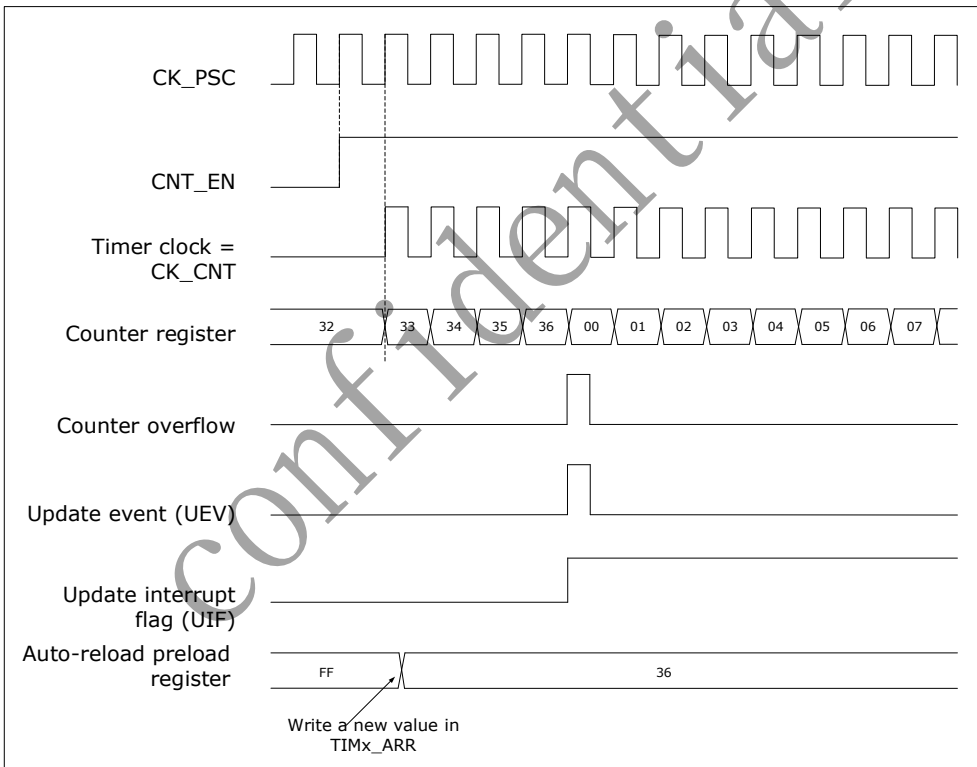


图 11-8 计数器时序图，当 ARPE=0 时的更新事件 (ADTIM_ARR 没有预装载)

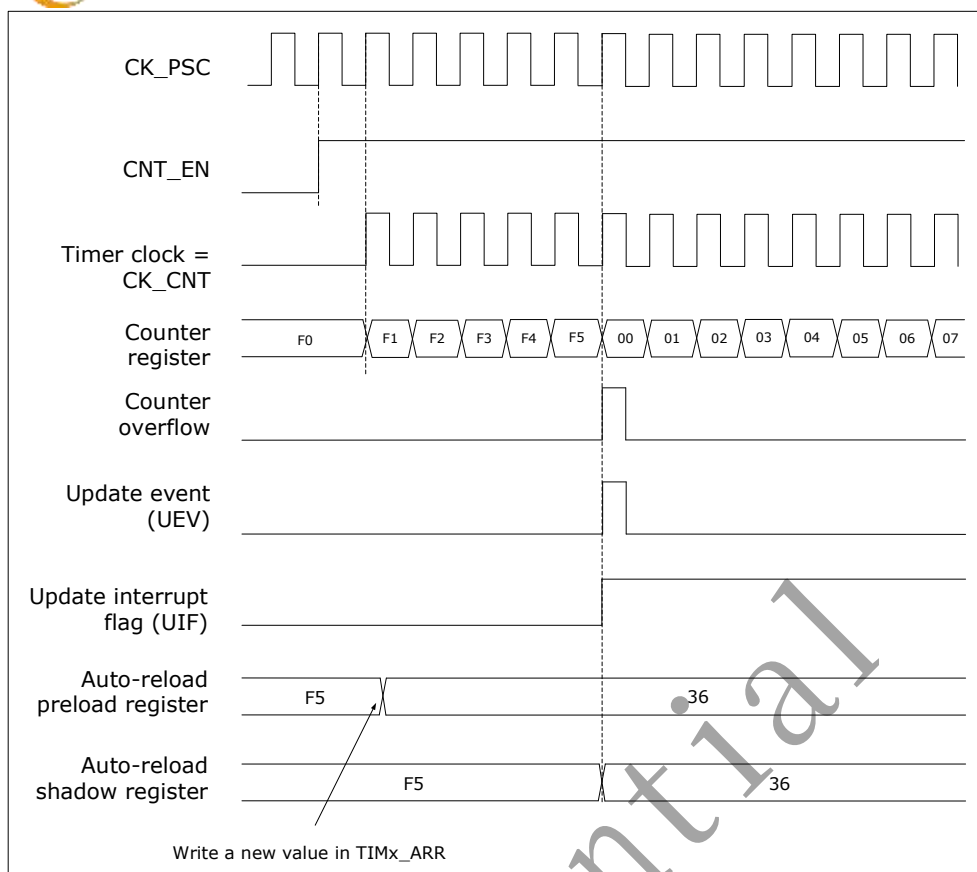


图 11-9 计数器时序图，当 ARPE=1 时的更新事件 (ADTIM_ARR 已预装载)

递减计数模式

在递减计数模式，计数器从自动加载值(ADTIM_ARR 寄存器内容)开始向下计数至 0，然后从自动加载值重新开始并产生一个计数器下溢事件。

如果使用了重复计数器，当向下计数重复了重复计数寄存器(ADTIM_RCR)中设定的次数后，才产生更新事件(UEV)，否则每次计数器下溢时都会产生更新事件。

设置 ADTIM_EGR 寄存器的(通过软件方式或者使用从模式控制器)UG 位，也同样可以产生一个更新事件。

通过软件设置 ADTIM_CR1 寄存器中的 UDIS 位，可以禁止更新事件；这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。在 UDIS 位被清'0'之前，将不产生更新事件。但是在应该产生更新事件时，计数器会被清'0'，同时预分频器的计数也被清 0(但预分频器的数值不变)。

此外，如果设置了 ADTIM_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但硬件不设置 UIF 标志(即不产生中断或 DMA 请求)。这是为了避免在捕获模式下清除计数器时，同时产生更新和捕获中断。

当发生一个更新事件时，所有的寄存器都被更新，同时(依据 URS 位)设置更新标志位(ADTIM_SR 寄存器中的 UIF 位)。

- ◆ 重复计数器被重新加载为 ADTIM_RCR 寄存器的内容

- ◆ 预分频器的缓冲区被写入预装载寄存器的值(ADTIM_PSC 寄存器内容)
- ◆ 自动装载影子寄存器被重新为预装载寄存器的值(ADTIM_ARR 寄存器内容)。

注:如果在计数器重载之前自动重装被更新,此时在下一周期时才是预期的值。

下列图表为计数器在 ADTIM_ARR=0x36 时对应不同时钟频点的行为范例

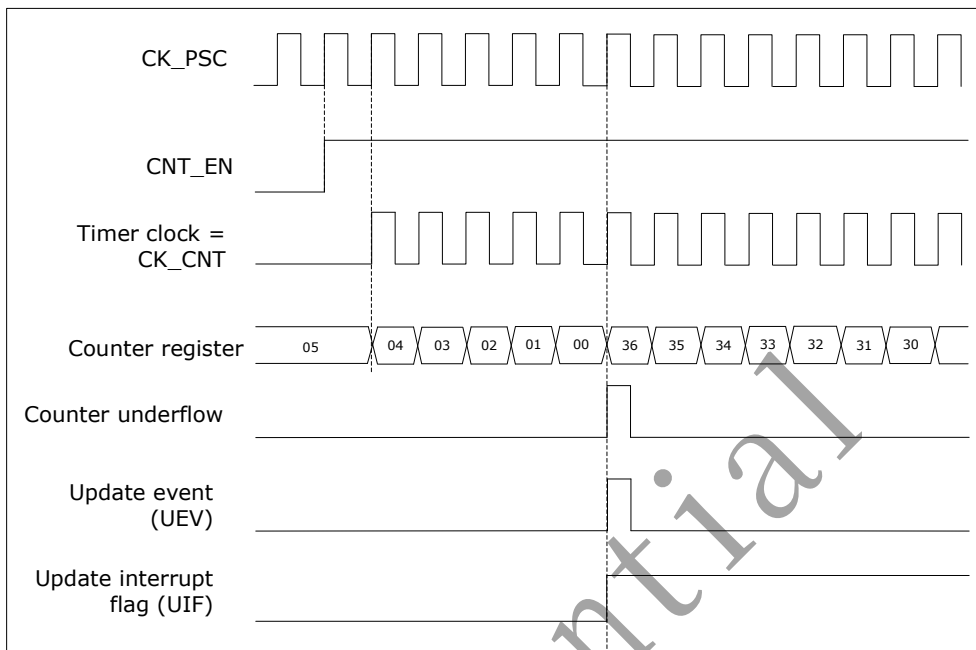


图 11-10 计数器时序图, 内部时钟分频因子为 1

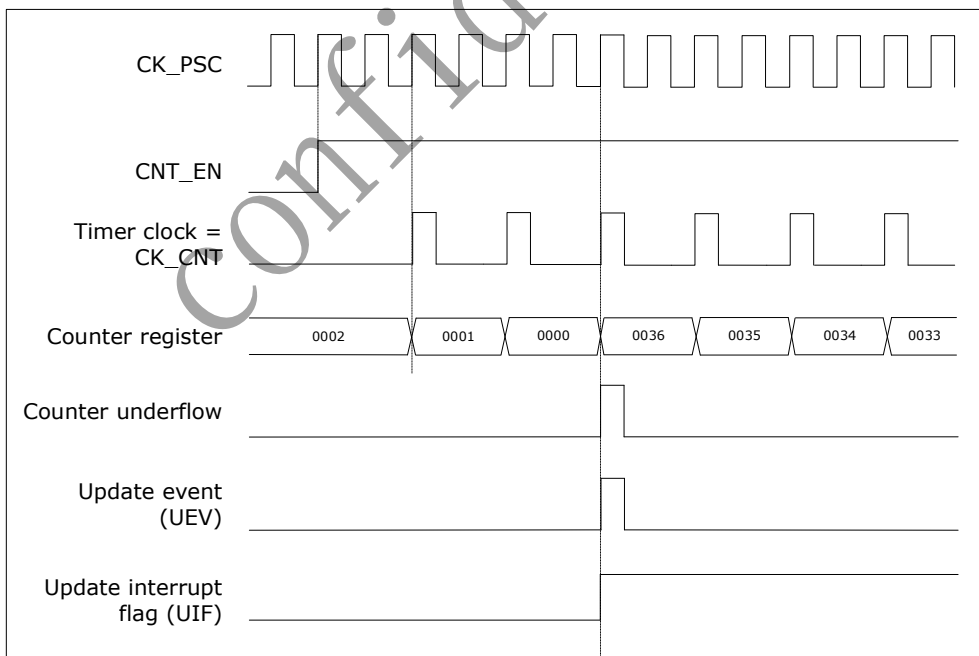


图 11-11 计数器时序图, 内部时钟分频因子为 2

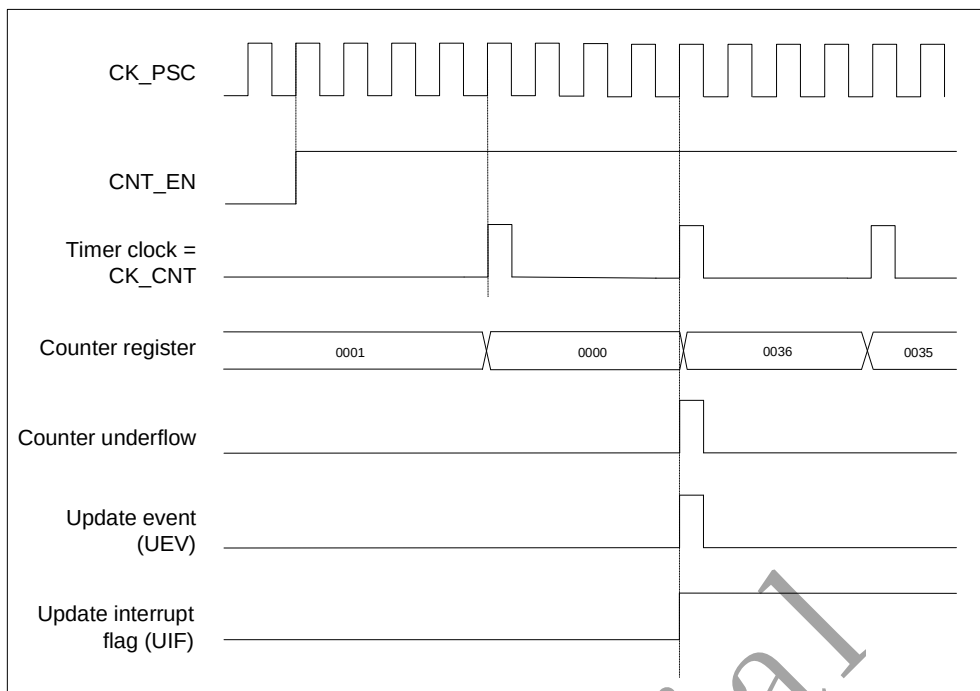


图 11-12 计数器时序图，内部时钟分频因子为 4

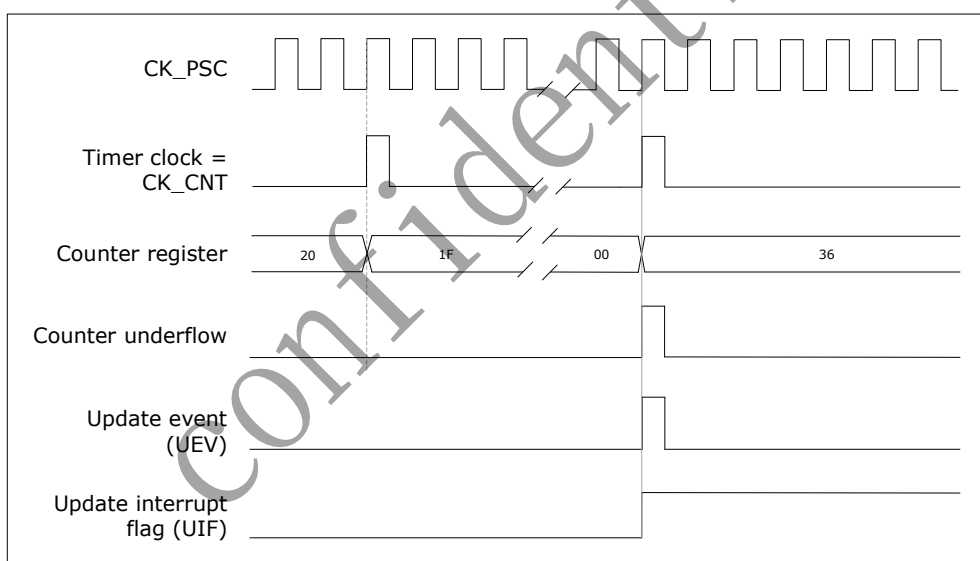


图 11-13 计数器时序图，内部时钟分频因子 N

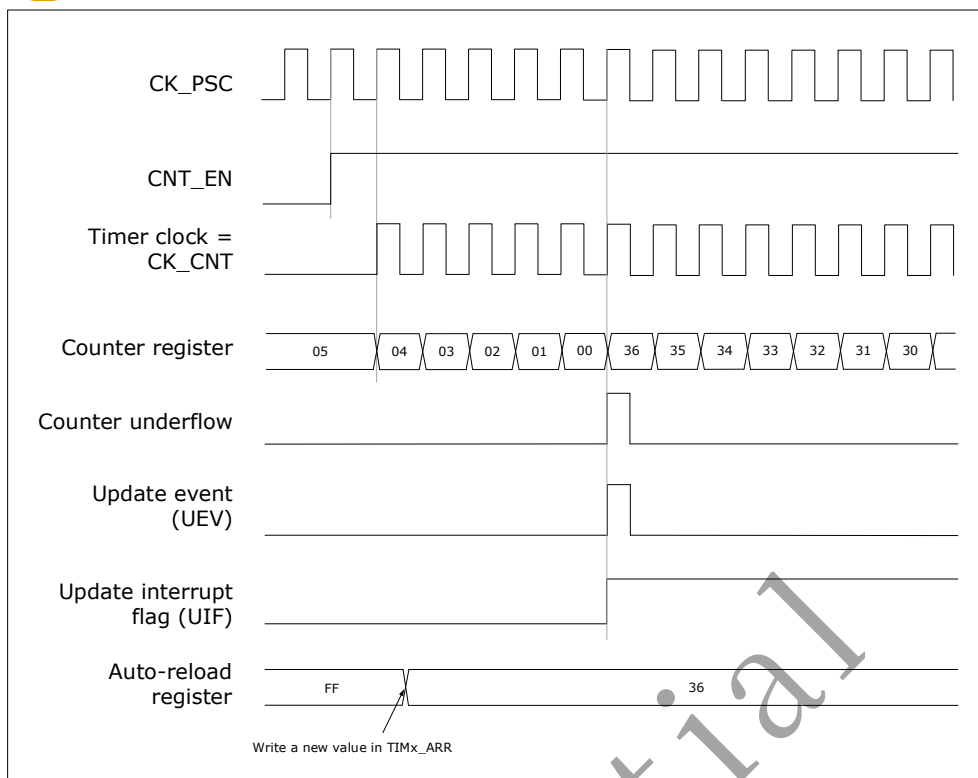


图 11-14 计数器时序图,重复计数器未使用时之更新事件

中心对齐模式(递增/递减计数)

在中央对齐模式，计数器从 0 开始计数到自动加载的值(ADTIM_ARR 寄存器)-1，产生一个计数器溢出事件，然后向下计数到 1 并且产生一个计数器下溢事件；然后再从 0 开始重新计数。当 ADTIM_CR1 的 CMS 位不为"00"时，使能中央对齐模式。已配置为输出的通道的输出比较中断标志在以下情况下被置位：计数器向下计数（中央对齐模式 1, CMS="01"），计数器向上计数（中央对齐模式 2, CMS="10"），计数器向下和向上计数（中央对齐模式 3, CMS="11"）。

在此模式下，不能写入 ADTIM_CR1 中的 DIR 方向位。它由硬件更新并指示当前的计数方向。可以在每次计数上溢和每次计数下溢时产生更新事件；也可以通过(软件或者使用从模式控制器)设置 ADTIM_EGR 寄存器中的 UG 位产生更新事件。此时，计数器重新从 0 开始计数，预分频器也重新从 0 开始计数。

设置 ADTIM_CR1 寄存器中的 UDIS 位可以禁止 UEV 事件。这样可以避免在向预装载寄存器中写入新值时更新影子寄存器。因此 UDIS 位被清为 0 之前不会产生更新事件。然而，计数器仍会根据当前自动重加载的值，继续向上或向下计数。

此外，如果设置了 ADTIM_CR1 寄存器中的 URS 位(选择更新请求)，设置 UG 位将产生一个更新事件 UEV，但不设置 UIF 标志(因此不产生中断和 DMA 请求)，这是为了避免在发生捕获事件并清除计数器时，同时产生更新和捕获中断。

当发生更新事件时，所有的寄存器都被更新，并且(根据 URS 位)更新标志位(ADTIM_RIF 寄存器中的 UIF 位)也被设置。

- ◆ 重复计数器被重置为 ADTIM_RCR 寄存器中的内容。

- ◆ 预分频器的缓存器被加载为预装载(ADTIM_PSC 寄存器)的值。
- ◆ 当前的自动加载寄存器被更新为预装载值(ADTIM_ARR 寄存器中的内容)。

注：如果因为计数器溢出而产生更新，自动重装载将在计数器重载入之前被更新，因此下一个周期才是预期的值(计数器被装载为新的值)。

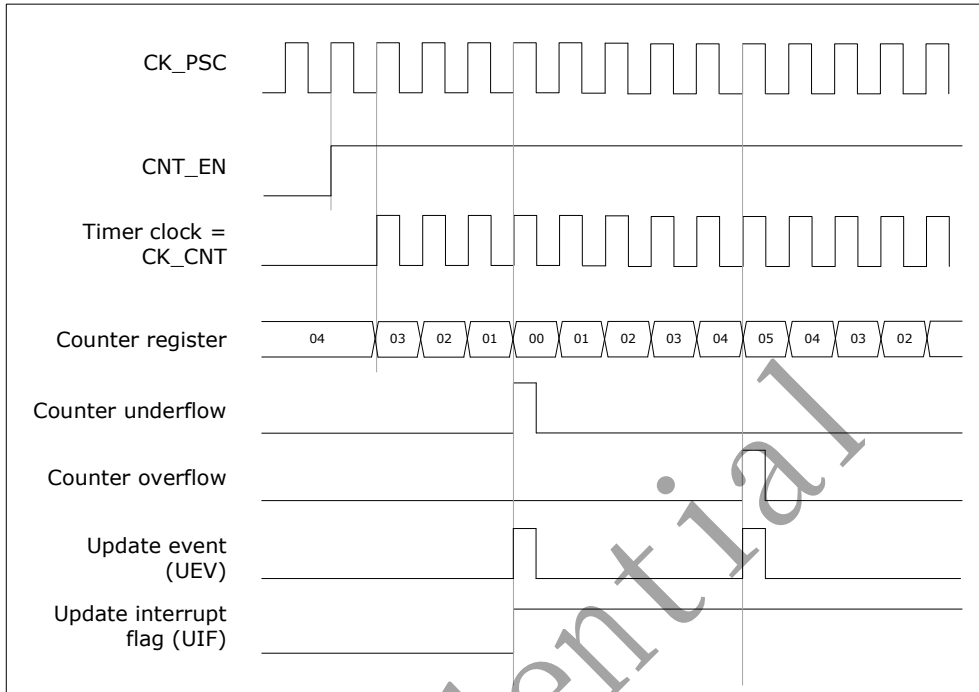


图 11-15 计数器时序图，内部时钟分频因子为 1,ADTIM_ARR=0x5

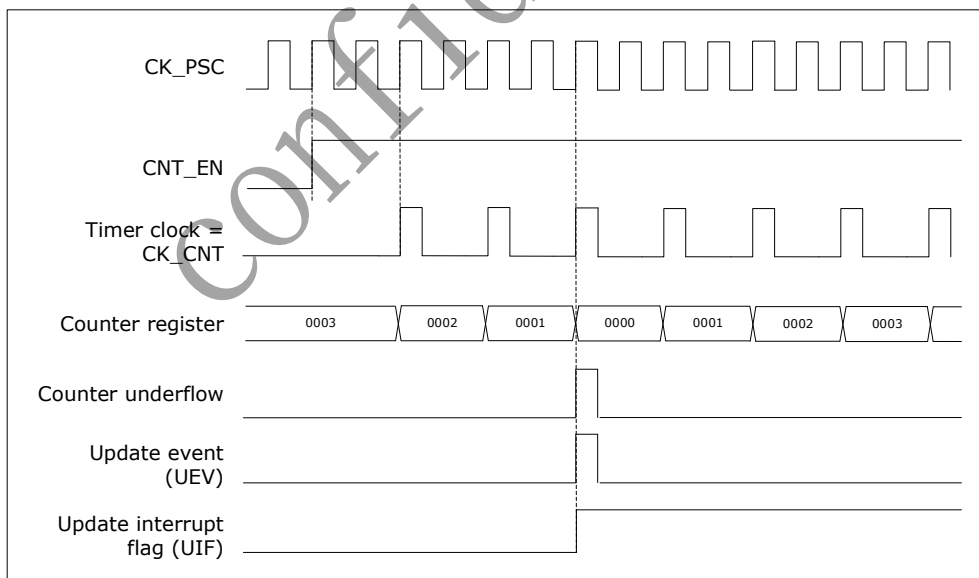


图 11-16 计数器时序图，内部时钟分频因子为 2

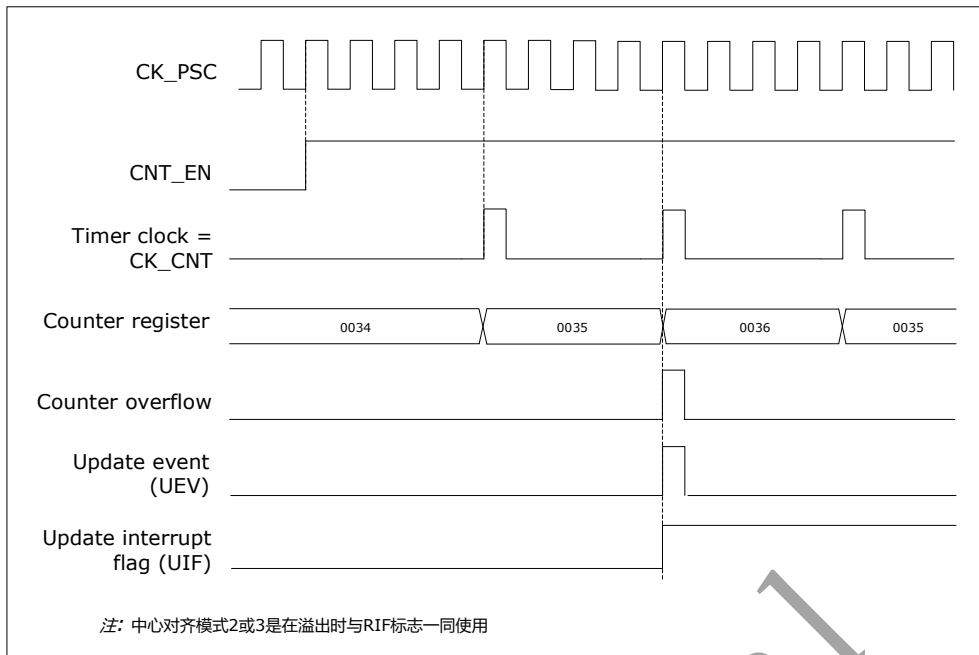


图 11-17 计数器时序图，内部时钟分频因子为 4, ADTIM_ARR=0x36

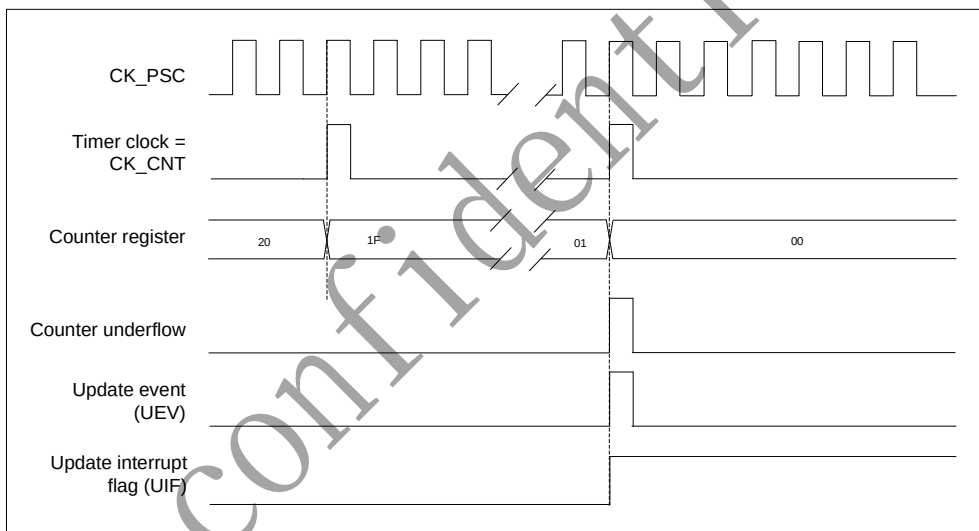


图 11-18 计数器时序图，内部时钟分频因子为 N

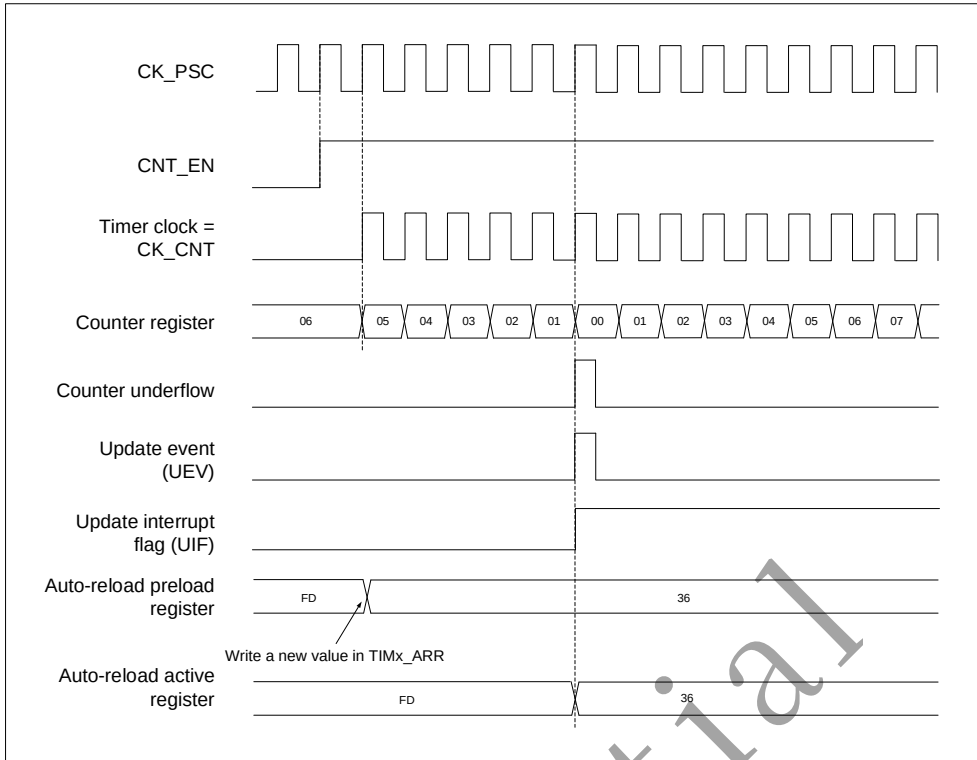


图 11-19 计数器时序图，ARPE=1 时之更新事件(计数器下溢)

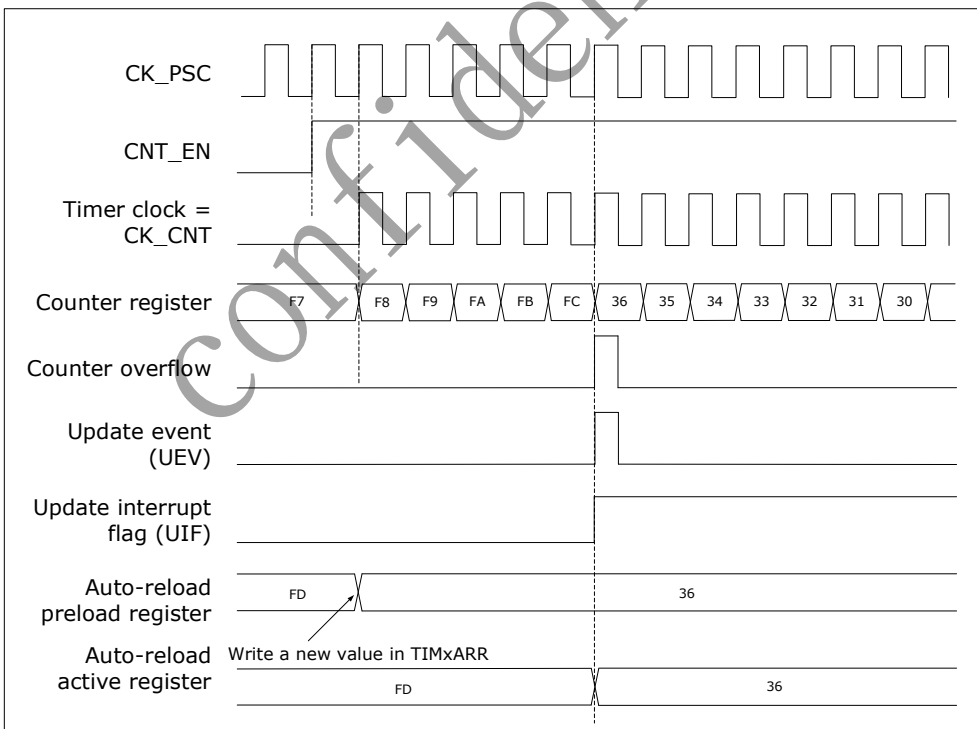


图 11-20 计数器时序图，ARPE=1 时之更新事件(计数器上溢)

1.3.3 重复计数器

11.4.1 节“时基单元”解释了计数器上溢/下溢时是如何产生更新事件(UEV)的，然而事实上它只能在重复计数达到 0 的时候产生。这个特性对产生 PWM 信号非常有用。

这意味着在每 N 次计数上溢或下溢时，数据从预装载寄存器传输到影子寄存器(ADTIM_ARR 自动重载入寄存器，ADTIM_PSC 预装载寄存器，还有在比较模式下的捕获/比较寄存器 ADTIM_CCRx)，N 是 ADTIM_RCR 重复计数寄存器中的值。

重复计数器在下述任一条件成立时递减：

- ◆ 向上计数模式下每次计数器上溢时
- ◆ 向下计数模式下每次计数器下溢时
- ◆ 中央对齐模式下每次上溢和每次下溢时。虽然这样限制了 PWM 的最大循环周期为 128，但它能够在每个 PWM 周期 2 次更新占空比。在中央对齐模式下，因为波形是对称的，如果每个 PWM 周期中仅刷新一次比较寄存器，则最大的分辨率为 $2 \times T_{ck}$ 。

重复计数器是自动加载的，重复速率是由 ADTIM_RCR 寄存器的值定义。当更新事件由软件产生(通过设置 ADTIM_EGR 中的 UG 位)或者通过硬件的从模式控制器产生，则无论重复计数器的值是多少，立即发生更新事件，并且 ADTIM_RCR 寄存器中的内容被重载入到重复计数器。

在中央对齐模式下，如果 RCR 是奇数，一旦 RCR 寄存器被写入并且计数器已经启动，每次溢出和下溢时都发生更新事件。如果 RCR 在计数器开始之前被写入，在溢出时发生 UEV 事件。如果 RCR 在计数器开始之后被写入，在下溢时发生 UEV 事件。例如 RCR=3，则在 RCR 被写入后的每 4 个溢出和下溢产生 UEV 事件。

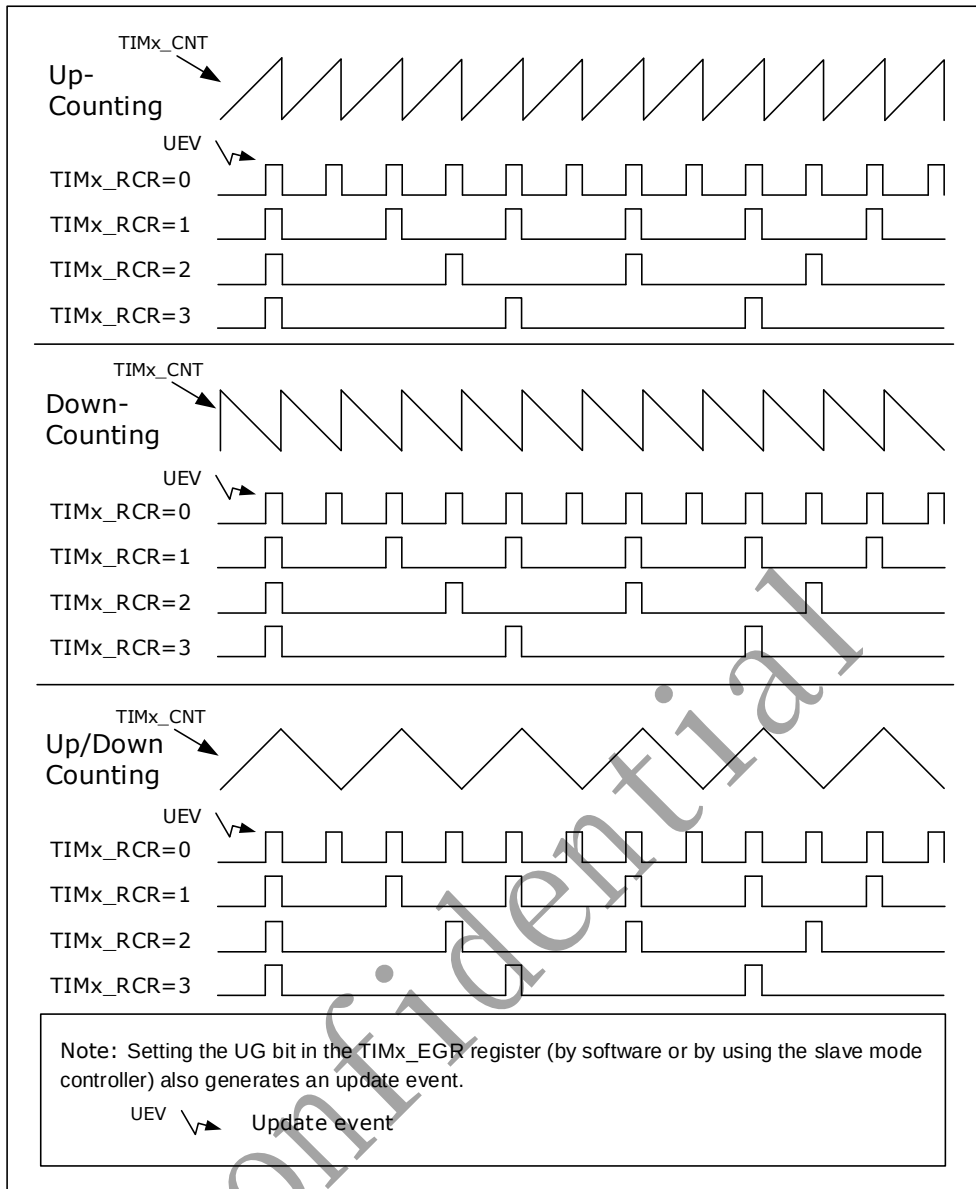


图 11-21 以模式与 ADTIM_RCR 寄存器设置为依据之更新比率范例

1.3.4 时钟源

计数器时钟可由下列时钟源提供:

- ◆ 内部时钟(CK_INT)
- ◆ 外部时钟模式 1:外部输入引脚
- ◆ 外部时钟模式 2:外部触发输入 ETR
- ◆ 内部触发输入(ITRx):使用一定时器作为另一个定时器的预分频器,例如:可以配置定时器 1 作为定时器 2 的预分频器。

1.3.4.1 内部时钟源(CK_INT)

如果禁止了从模式控制器(SMS=000),则 CEN、DIR(ADTIM_CR1 寄存器)和 UG 位(ADTIM_EGR 寄存器)是事实上的控制位,并且只能被软件修改(除非 UG 位自动被清除)。一旦 CEN 位被写成'1',预分频器的时钟就由内部时钟 CK_INT 提供。

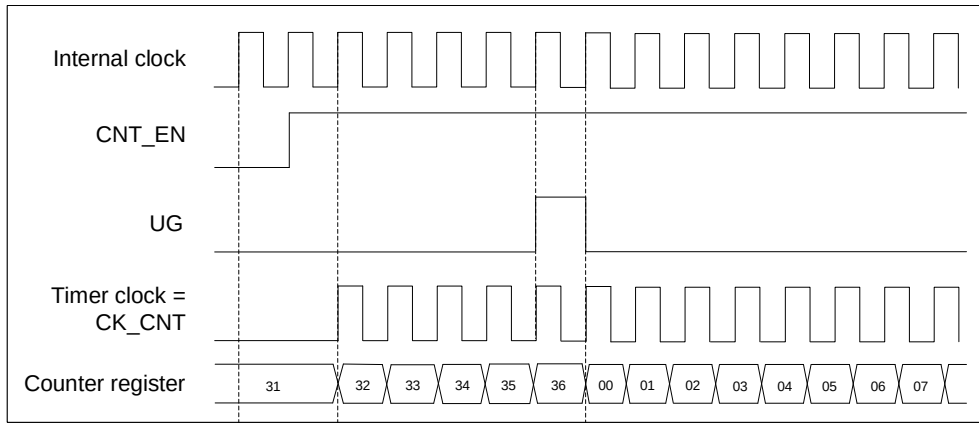


图 11-22 一般模式下之控制电路，内部时钟分频因子为 1

1. 3. 4. 2 外部时钟源模式 1

当 ADTIM_SMCR 寄存器的 SMS=111 时，此模式被选中。计数器在选定输入端的每个上升沿或下降沿计数。

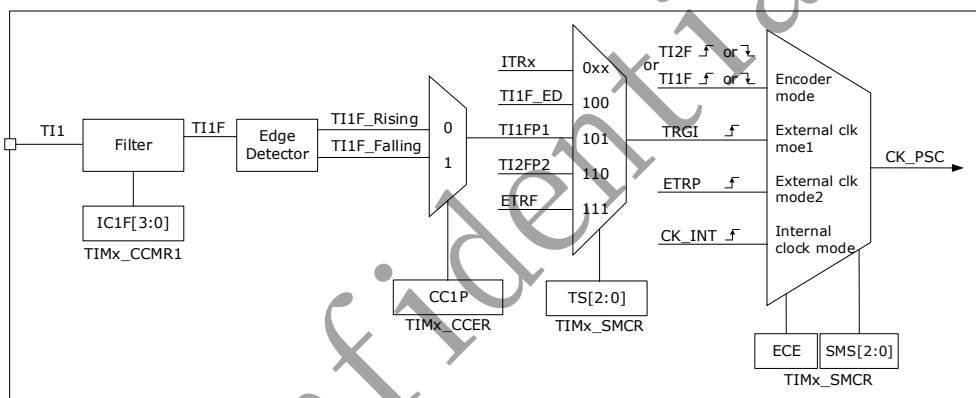


图 11-23 外部时钟连接范例

例:要配置向上计数器在 TI1 输入端的上升沿计数，如以下步骤：

- ◆ 配置 ADTIM_CCMR 寄存器 CC1S=01，配置信道 1 以检测 TI1 输入的上升沿。
- ◆ 写 ADTIM_CCMR 寄存器之 IC1F[3:0]位以配置输入滤波器期间(若没有滤波器需求，维持 IC1F=0000)
- ◆ 写 CC1P=0 至寄存器 ADTIM_CCER 以选择上升缘极性
- ◆ 透过写 ADTIM_SMCR 寄存器的 SMS=111，以配置定时器外部时钟模式 1
- ◆ 写 ADTIM_SMCR 寄存器中的 TS=101，以选定 TI1 作为触发输入源
- ◆ 写 ADTIM_CR1 寄存器之 CEN=1 以使能计数器

当上升缘发生于 TI1，计数器计数一次且 TIF 标志被设置

TI1 上升缘与实际时钟间之延时，取决于 TI1 输入之再同步电路

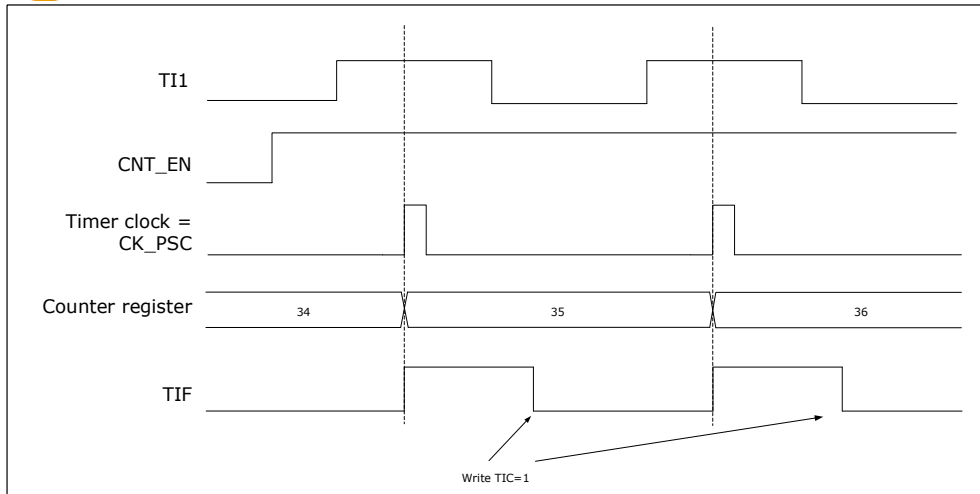


图 11-24 外部时钟模式 1 之控制电路

1.3.4.3 外部时钟源模式 2

此模式透过写 ADTIM_SMCR 寄存器之 ECE=1 被选定

计数器可于每一外部触发输入 ETR 上升或下降缘计数，并可搭配预除频器。

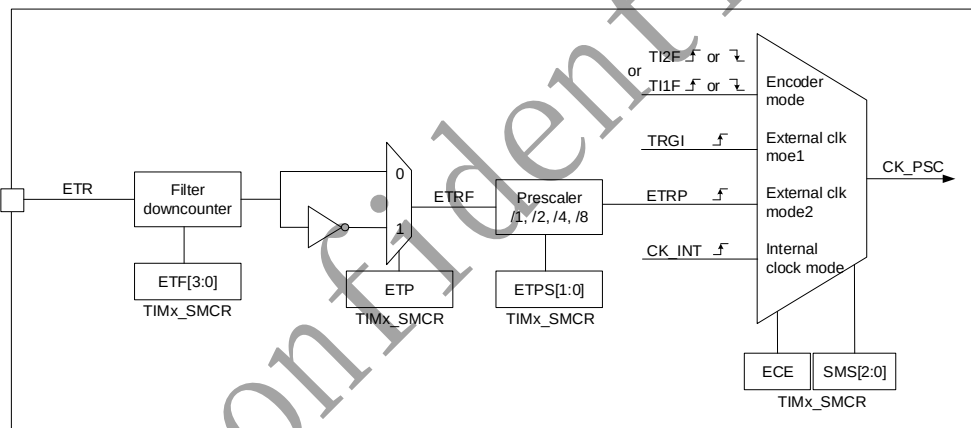


图 11-25 ETR 给予一外部触发输入区块概述

例:欲设置上数计数器于每两个 ETR 上升缘计数，则使用下列方式:

- ◆ 此例中无须使用滤波器，写 ADTIM_SMCR 寄存器之 ETF[3:0]=0000
- ◆ 写 ADTIM_SMCR 寄存器之 ETPS[1:0]=01 以设置预除器
- ◆ 写 ADTIM_SMCR 寄存器之 ETP=0 以选择 ETR 引脚之上升缘侦测
- ◆ 写 ADTIM_SMCR 寄存器之 ECE=1 以使能外部时钟模式 2
- ◆ 写 ADTIM_CR1 寄存器之 CEN=1 以使能计数器

计数器于每两个上升缘计数

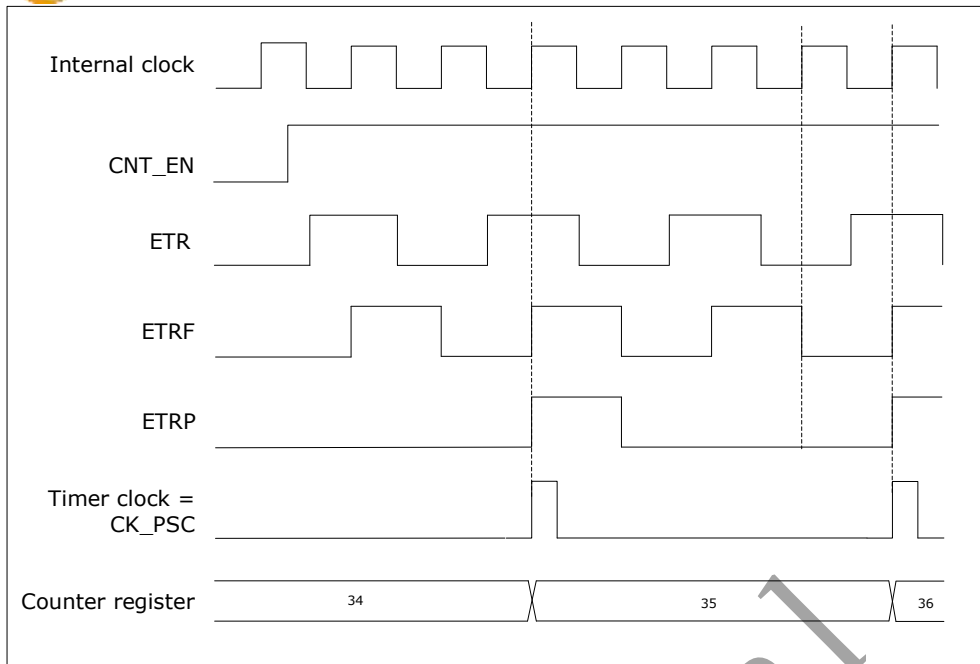


图 11-26 外部时钟模式 2 之控制电路

1. 3. 4. 4 内部触发输入(ITRx)

当 ADTIM_SMCR 寄存器之 SMS=111 时此模式会被选定。计数器于被选定输入端之每一上升或下降缘计数。

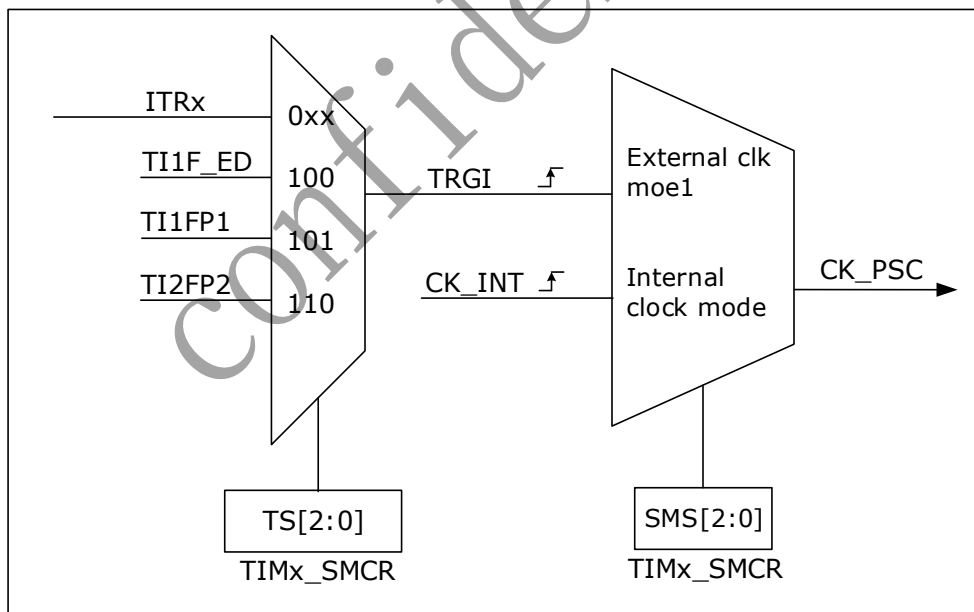


图 11-27ITRx 外部时钟连接范例

例:要配置向上计数器在 ITRx 输入端的上升沿计数, 如以下步骤:

- ◆ 透过写 ADTIM_SMCR 寄存器的 SMS=111, 以配置定时器外部时钟模式 1
- ◆ 写 ADTIM_SMCR 寄存器中的 TS=0xxx, 以选定 ITRx 作为触发输入源
- ◆ 写 ADTIM_CR1 寄存器之 CEN=1 以使能计数器

当上升缘发生于 ITRx，计数器计数一次

ITRx 上升缘与实际时钟间之延时，取决于 ITRx 输入之再同步电路

1.3.5 捕获/比较信道

每一捕获/比较信道，皆是围绕着一捕获/比较寄存器(包含一 shadow 寄存器)，包含一捕获输入部分(含数字滤波器，多路复用与预除器)以及一输出阶段(含比较器与输出控制)。

下图 14-28 为单一捕获/比较信道之概述

输入阶段范例对应之 Tix 输入以产生一滤波后之 TixF 信号，接着一带极性边缘侦测器产生一信号(TixFPx)，可作为属模式控制器之触发输入或是做为捕获控制。该信号于捕获寄存器(ICxPS)之前即预先分频。

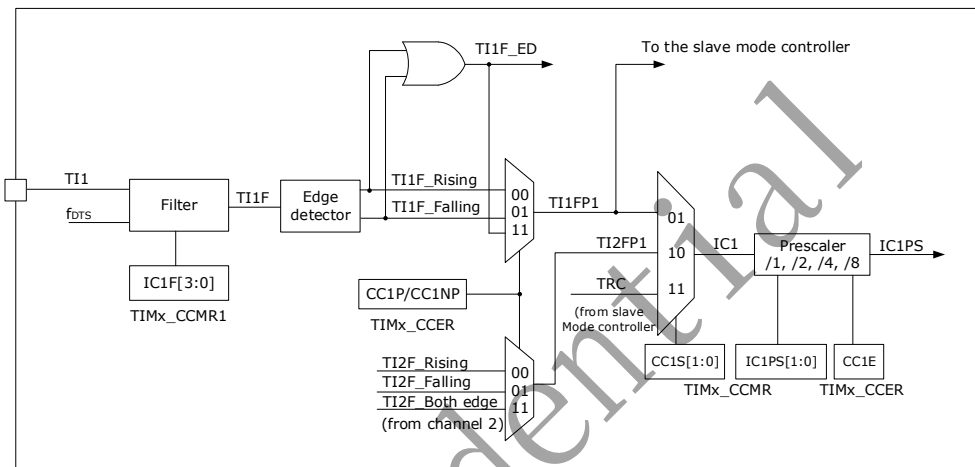


图 11-28 捕获/比较信道(例:信道 1 输入阶段)

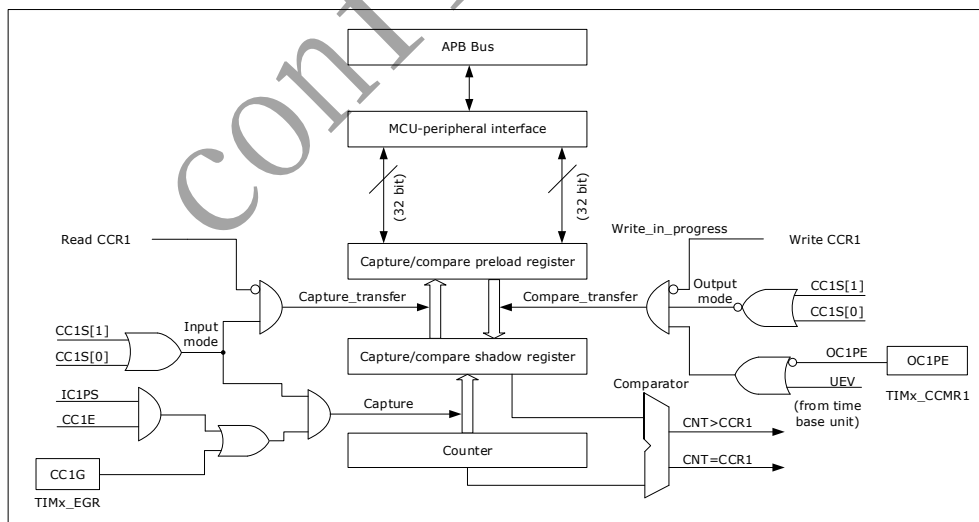


图 11-29 捕捉/比较信道 1 主电路

输出部分产生一个中间波形 OCxRef(高有效)作为基准，链结的末端决定最终输出信号之极性。

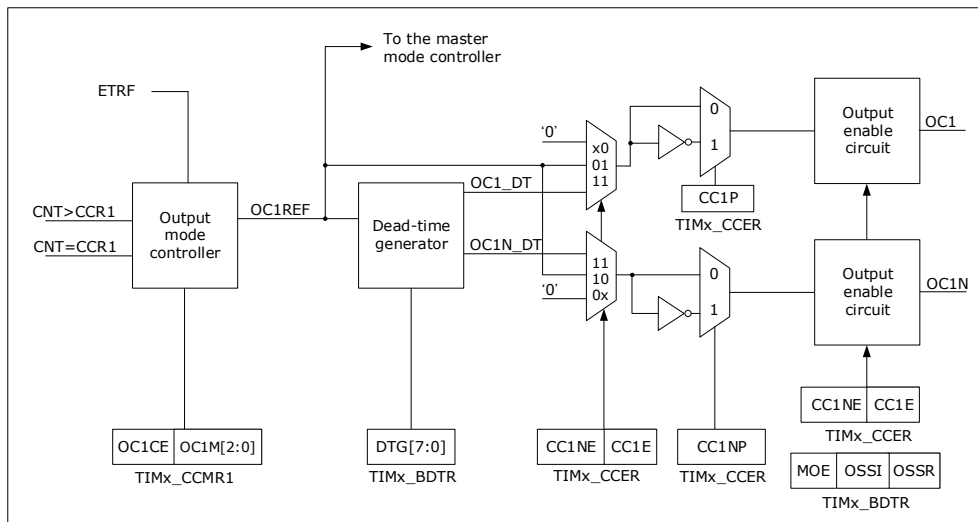


图 11-30 捕捉/比较信道之输出阶段(信道 1 至 3)

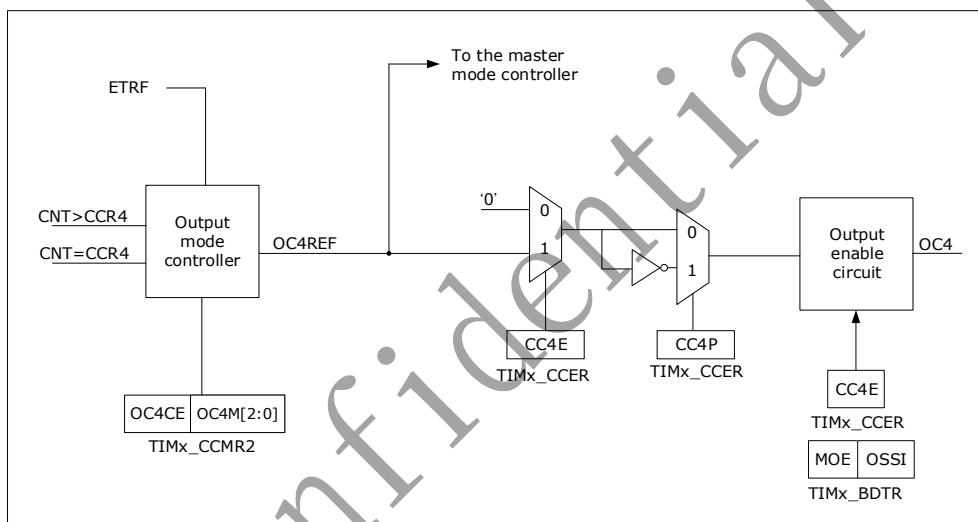


图 11-31 捕捉/比较信道之输出阶段(信道 4)

捕捉/比较区块是由一预载寄存器与影子寄存器组成。读写皆可存取预载寄存器

于捕捉模式下，捕捉实际上皆于影子寄存器完成并复制到预载寄存器。

于比较模式下，预载寄存器内容被复制到影子寄存器与计数器做比较

1.3.6 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(ADTIM_CCRx)中。当发生捕获时，若皆已使能则相应的中断可被送出。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 ADTIM_CCR1 寄存器中，步骤如下：

- ◆ 选择有效输入端：ADTIM_CCR1 必须连接到 TI1 输入，所以写入 ADTIM_CCMR1 寄存器中的 CC1S 为 01。只要 CC1S 不为 '00'，信道被配置为输入，并且

ADTIM_CCR1 寄存器变为只读。

- ◆ 参照连结至定时器之信号，编程你所需要的输入滤波器期间(当输入为 ADTIM_CCMR1 寄存器之 TIx(ICxF 位其中一位).当触发时输入信号于五个必要内部时钟周期内并不稳定。用户需编制比五个时钟周期还长之滤波器期间。可于 8 个新的电平接续采样被侦测到(于 fDTS 频率采样)时开始 TI1 传输。接着写 ADTIM_CCMR1 寄存器之 IC1F 位为 0011。
- ◆ 选择 TI1 信道的有效转换边沿，在 ADTIM_CCER 寄存器中写入 CC1P 与 CC1NP 至 0(上升沿)
- ◆ 对输入预除器编程。于例中希望捕获可在每一传输中作用，因此预除器禁止使用(写 ADTIM_CCMR1 寄存器之 IC1PS 位为 '00').
- ◆ 设定 ADTIM_CCER 寄存器 CC1E 位以使能捕捉计数器至捕捉寄存器
- ◆ 如果需要，透过设定 ADTIM_IER 寄存器之 CC1 位以开启相关中断需求
- ◆ 当输入捕获中断发生时：
- ◆ ADTIM_CCR1 寄存器得到计数器之作用传输值
- ◆ CC1IF 状态被设置(中断状态).CC1OF 也被设定，若至少有 2 个接续捕获发生，然而标志未被清除
- ◆ 如设置 ADTIM_IER 寄存器之 CC1 位，则会产生一个中断。
- ◆ 如设置 ADTIM_DMAEN 寄存器之 CC1 位，则还会产生一个 DMA 请求。

为了处理捕获溢出，建议在读出捕获溢出标志之前读取数据，这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注:ICx 中断需求可由软件设置 ADTIM_EGR 寄存器内对应 CCxG 位产生

1.3.7 PWM 输入模式

该模式是输入捕获模式的一个特例，除下列区别外，操作与输入捕获模式相同：

- ◆ 两个 ICx 信号被映射至同一个 TIx 输入
- ◆ 这 2 个 ICx 信号为边沿有效，但是极性相反
- ◆ 其中一个 TIxFP 信号被作为触发输入信号，而从模式控制器被配置成复位模式

例:可测量输入到 TI1 上的 PWM 信号的长度(ADTIM_CCR1 寄存器)和占空比(ADTIM_CCR2 寄存器)，具体步骤如下(取决于 CK_INT 的频率和预分频器的值)

- ◆ 选择 ADTIM_CCR1 的有效输入：置 ADTIM_CCMR1 寄存器的 CC1S=01(选中 TI1)
- ◆ 选择 TI1FP1 的有效极性(用来捕获数据到 ADTIM_CCR1 中和清除计数器)：置 CC1P 为 0(上升沿有效)
- ◆ 选择 ADTIM_CCR2 有效输入:置 ADTIM_CCMR1 寄存器之 CC2S 位至 10(TI1 被选中)
- ◆ 选择 TI1FP2 有效极性(用于 ADTIM_CCR2 捕捉):置 CC2P 位至为 '1'(于下降缘作用)
- ◆ 选择可作用触发输入:置 ADTIM_SMCR 寄存器之 TS 位为 101(选择 TI1FP1)
- ◆ 设置属模式控制器于重置模式:置 ADTIM_SMCR 寄存器之 SMS 位为 100

- ◆ 使能捕获：置 ADTIM_CCER 寄存器中 CC1E=1 且 CC2E=1。

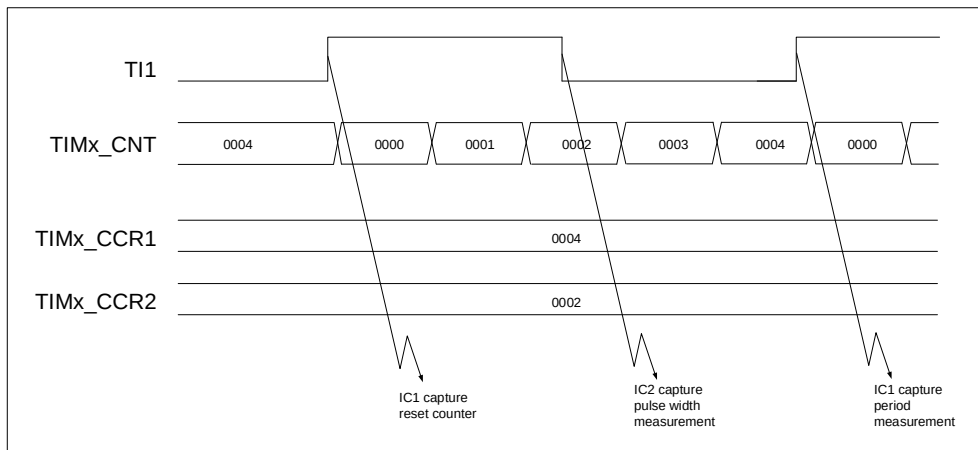


图 11-32PWM 输入模式时序

1.3.8 强制输出模式

在输出模式(ADTIM_CCMRx 寄存器中 CCxS=00)下，输出比较信号(OCxREF 和相应的 OCx/OCxN)能够直接由软件强置为有效或无效状态，而不依赖于输出比较寄存器和计数器间的比较结果。

置 ADTIM_CCMRx 寄存器中相应的 OCxM=101，即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效)，同时 OCx 得到 CCxP 极性相反的信号。

例如：CCxP=0(OCx 高电平有效)，则 OCx 被强置为高电平。

置 ADTIM_CCMRx 寄存器中的 OCxM=100，可强置 OCxREF 信号为低。

该模式下，在 ADTIM_CCRx 影子寄存器和计数器之间的比较仍然在进行，相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。这将会在下面的输出比较模式一节中介绍。

1.3.9 输出比较模式

此功能用于控制输出波形或是当时间消逝时之周期标示。

当于捕捉/比较寄存器与计数器发现匹配时，输出比较功能：

- ◆ 将输出比较模式(ADTIM_CCMRx 寄存器中的 OCxM 位)和输出极性(ADTIM_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时，输出引脚可以保持它的电平(OCxM=000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。
- ◆ 设置中断状态寄存器中的标志位(ADTIM_RIF 寄存器中的 CCxIF 位)。
- ◆ 若设置了相应的中断屏蔽(ADTIM_IER 寄存器中的 CCxIE 位)，则产生一个中断。
- ◆ 若设置了相应的使能位(ADTIM_DMAEN 寄存器中的 CCxDE 位，ADTIM_CR2 寄存器中的 CCDS 位选择 DMA 请求功能)，则产生一个 DMA 请求。

ADTIM_CCRx 寄存器可由或不由预除寄存器之 OCxPE 位编程

于输出比较模式下，更新事件 UEV 对于 OCxREF 与 OCx 输出没有效果。时程解析亦为计数

器之单一计数。输出比较模式也可被用来输出单一脉冲(单脉冲模式)。于输出比较模式，更新事件 UEV 于 OCxREF 与 OCx 无效果

过程:

选定计数器时钟(内部,外部,预除).

- ◆ 写入需要之数据至 ADTIM_ARR 与 ADTIM_CCRx 寄存器.
- ◆ 若中断需求产生则设置 ADTIM_IER 之 CCxIE 位
- ◆ 选择输出模式，例:
 - 当 CNT 符合 CCRx 时写 OCxM=011 以触发 OCx 输出引脚
 - 写 OCxPE =0 禁用预载寄存器
 - 写 CCxP =0 选择高作用极性
 - 写 CCxE =1 使能输出
- ◆ 设置 ADTIM_CR1 寄存器之 CEN 位以使能计数器

ADTIM_CCRx 寄存器可于任何时间被软件更新以控制输出波形,预载寄存器提供预除寄存器并未使能(OCxPE='0',此外 ADTIM_CCRx 影子寄存器只有在下一个更新事件 UEV 被更新).图 11-31 提供一范例:

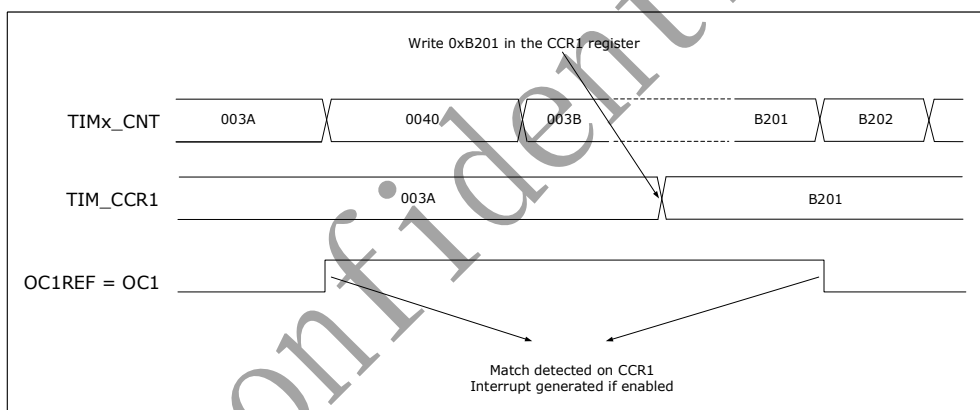


图 11-33 输出比较模式,触发 OC1

1. 3. 10 PWM 模式

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

在 ADTIM_CCMRx 寄存器中的 OCxM 位写入 '110' (PWM 模式 1)或 '111' (PWM 模式 2),能够独立地设置每个 OCx 输出信道产生一路 PWM。必须设置 ADTIM_CCMRx 寄存器 OCxPE 位以使能相应的预装载寄存器，最后还要设置 ADTIM_CR1 寄存器的 ARPE 位, (在向上计数或中心对称模式中)使能自动重载的预装载寄存器。

仅当发生一个更新事件的时候，预装载寄存器才能被传送到影子寄存器，因此在计数器开始计数之前，必须通过设置 ADTIM_EGR 寄存器中的 UG 位以初始化所有寄存器。

OCx 的极性可以通过软件在 ADTIM_CCER 寄存器中的 CCxP 位设置，它可以设置为高电平有效或低电平有效。ADTIM_CCER 寄存器中的 CCxE,CCxNE,MOE,OSSI 和 OSSR 位联合控

制 OCx 输出使能(ADTIM_CCER 与 ADTIM_BDTR 寄存器)。详见 ADTIM_CCER 寄存器的描述。

在 PWM 模式(模式 1 或模式 2)下, ADTIM_CNT 和 ADTIM_CCRx 始终在进行比较, (依据计数器的计数方向)以确定是否符合 $ADTIM\_CCRx \leq ADTIM\_CNT$ 或者 $ADTIM\_CNT \leq ADTIM\_CCRx$ 。

根据 ADTIM_CR1 寄存器中 CMS 位的状态, 定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

PWM 边沿对齐模式

◆ 向上计数配置

当 ADTIM_CR1 寄存器中的 DIR 位为低的时候执行向上计数。参考以下为一 PWM 模式 1 之例子。当 $ADTIM\_CNT < ADTIM\_CCRx$ 时, PWM 参考信号 OCxREF 为高, 否则为低。如果 ADTIM_CCRx 中的比较值大于自动重载值(ADTIM_ARR), 则 OCxREF 保持为 '1'。如果比较值为 0, 则 OCxREF 保持为 '0'。下图为 ADTIM_ARR=8 时边沿对齐的 PWM 波形实例。

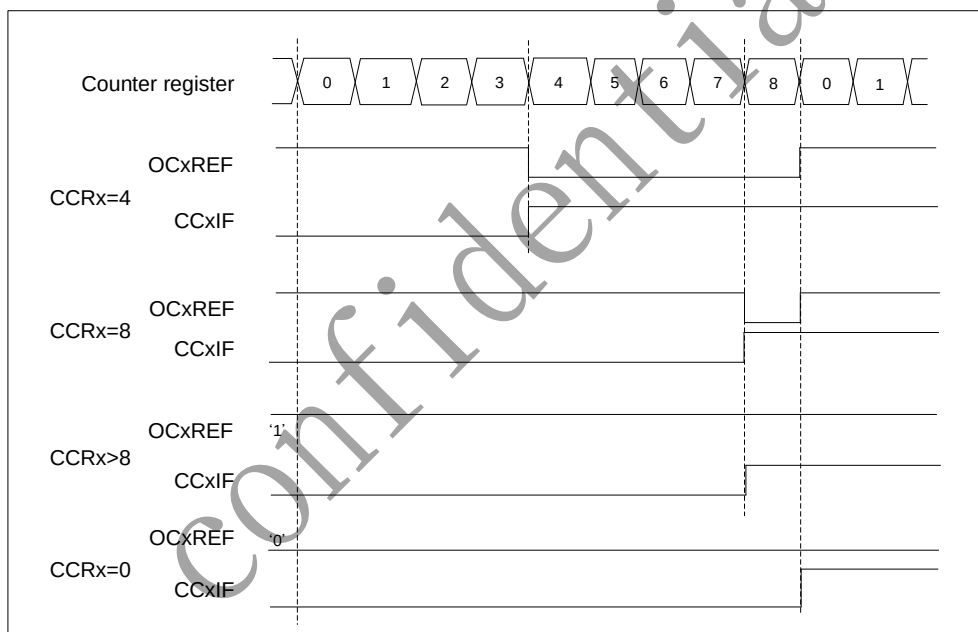


图 11-34 边缘对齐 PWM 波形, 向上计数(ARR=8)

◆ 向下计数配置

当 ADTIM_CR1 寄存器的 DIR 位为高时执行向下计数。在 PWM 模式 1, 当 $ADTIM\_CNT > ADTIM\_CCRx$ 时参考信号 OCxREF 为低, 否则为高。如果 ADTIM_CCRx 中的比较值大于 ADTIM_ARR 中的自动重载值, 则 OCxREF 保持为 '1'。该模式下不能产生 0% 的 PWM 波形。

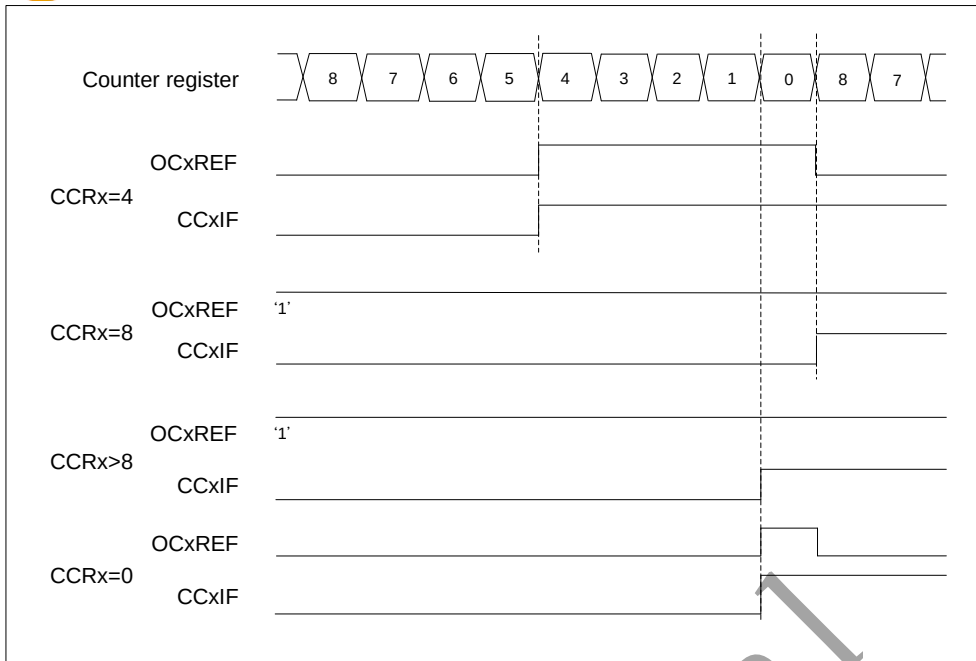


图 11-35 边缘对齐 PWM 波形，向下计数(ARR=8)

PWM 中央对齐模式

当 ADTIM_CR1 寄存器中的 CMS 位不为 '00' 时为中央对齐模式(所有其他的配置对 OCxREF/OCx 信号都有相同的作用)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。

ADTIM_CR1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它

下图给出了一些中央对齐的 PWM 波形的例子:

- ◆ ADTIM_ARR=8,
- ◆ PWM 模式 1,

ADTIM_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

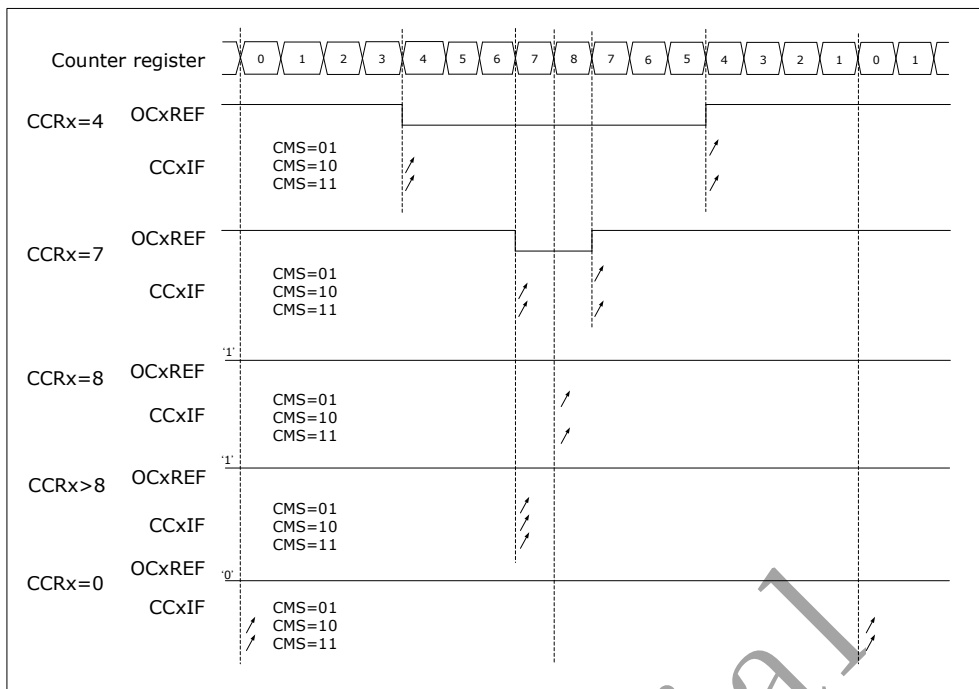


图 11-36 中央对齐 PWM 波形(ARR=8)

使用中央对齐模式的提示:

- ◆ 进入中央对齐模式时, 使用当前的向上/向下计数配置; 这就意味着计数器向上还是向下计数取决于 **ADTIM_CR1** 寄存器中 **DIR** 位的当前值。此外, 软件不能同时修改 **DIR** 和 **CMS** 位。
- ◆ 不推荐当运行在中央对齐模式时改写计数器, 因为这会产生不可预知的结果。特别地:
 - 如果写入计数器的值大于自动重加载的值(**ADTIM_CNT**>**ADTIM_ARR**), 则方向不会被更新。例如, 如果计数器正在向上计数, 它就会继续向上计数。
 - 如果将 **0** 或者 **ADTIM_ARR** 的值写入计数器, 方向被更新, 但不产生更新事件 **UEV**。
- ◆ 使用中央对齐模式最保险的方法, 就是在启动计数器之前产生一个软件更新(设置 **ADTIM_EGR** 位中的 **UG** 位), 并且不要在计数进行过程中修改计数器的值。

1. 3. 11 单脉冲模式

单脉冲模式(OPM)为前一模式之特例。其允许计数器对应刺激起始并于可编程延迟后产生可编程长度之脉冲。

起始计数器可由属模式控制器控制。由输出比较模式或 **PWM** 模式产生波形。可透过设置 **ADTIM_CR1** 寄存器之 **OPM** 位以选择单脉冲模式。如此计数器会在下一个更新事件 **UEV** 自动停止。

单一脉冲仅可于比较值不同于计数器初始值时被正确产生。在开始之前(当定时器于等待触发器时), 必须依照下列方式配置

- ◆ 向上计数: $CNT < CCRx \leq ARR$ (特例, $0 < CCRx$)

◆ 向下计数: $CNT > CCRx$

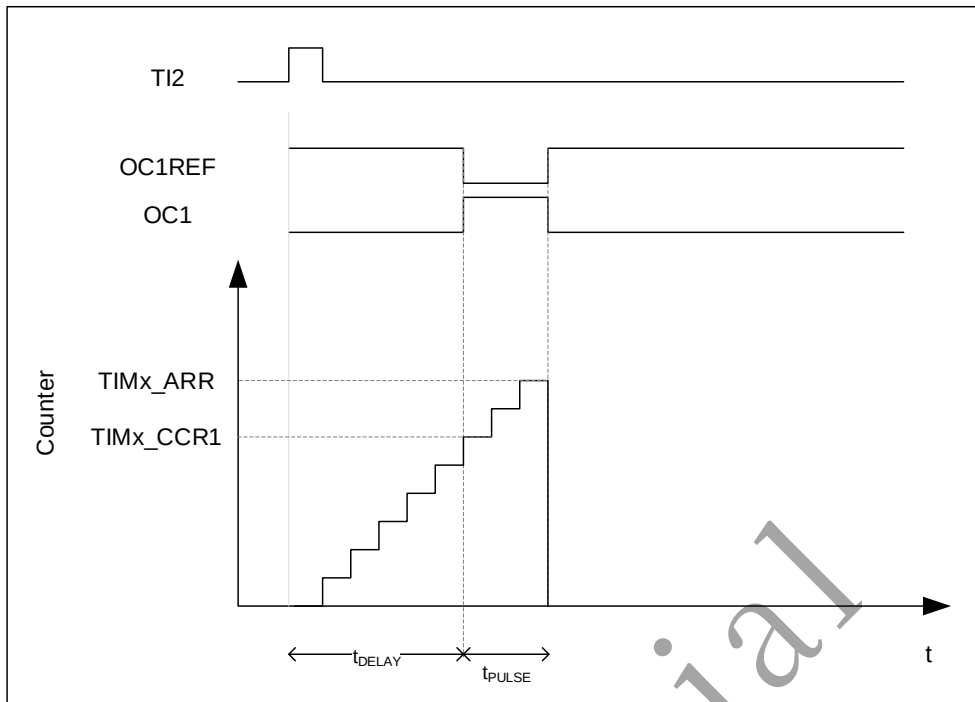


图 11-37 单脉冲模式范例

例如，你需要在在 OC1 上产生一个长度为 tPULSE 的正脉冲，从 TI2 输入脚上检测到一个上升沿开始，延迟 tDELAY 之后。

假定 TI2FP2 作为触发 1:

- ◆ 置 ADTIM_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。
- ◆ 置 ADTIM_CCER 寄存器中的 CC2P=0 和 CC2NP=0，使 TI2FP2 能够检测上升沿。
- ◆ 置 ADTIM_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。
- ◆ 置 ADTIM_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器决定(要考虑时钟频率和计数器预分频器)

- ◆ tDELAY 由 ADTIM_CCR1 寄存器中的值定义。
- ◆ tPULSE 由自动装载值和比较值之间的差值定义(ADTIM_ARR-ADTIM_CCR1)。
- ◆ 假定当发生比较匹配时要产生从 0 到 1 的变换，当计数器达到预装载值时要产生一个从 1 到 0 的变换：首先要置 ADTIM_CCMR1 寄存器的 OC1M=111，选择 PWM 模式 2；根据需要要有选择地使能预装载寄存器：置 ADTIM_CCMR1 中的 OC1PE=1 和 ADTIM_CR1 寄存器中的 ARPE；此时必须向 ADTIM_CCR1 寄存器中填写比较值，在 ADTIM_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，ADTIM_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 ADTIM_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。当 ADTIM_CR1 寄存器中的 OPM=0 时，进入

重复模式。

特殊情况：OCx 快速使能

在单脉冲模式下，在 TIX 输入脚的边沿检测逻辑设置 CEN 位来启动计数器。然后计数器和比较值间的比较结果驱动输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 tDELAY。

如果需要以最小延时输出波形，可以设置 ADTIM_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF(和 OCx)直接响应激励而不再依赖比较的结果，输出的波形与比较匹配时的波形一样。OCxFE 只在信道配置为 PWM 模式 1 和 PWM 模式 2 时起作用。

1.3.12 互补输出与死区内嵌

高级控制定时器可输出两个互补信号并管理开关切换

该时间通常被认作 dead-time 且须依链接之装置输出与其特性调整(因电平切换、电源切换造成之本质延迟...)

可单独选择输出极性(主要输出 OCx 或互补性 OCxN)。透过写 ADTIM_CCER 寄存器之 CCxP 与 CCxNP 位达到前述效果

互补性信号 OCx 与 OCxN 皆由数个控制位结合进行激活:ADTIM_CCER 寄存器之 CCxE 与 CCxNE 位，ADTIM_BDTR 与 ADTIM_CR2 寄存器之 MOE,OISx,OISxN,OSSI 与 OSSR 位。特别是当切换至 IDLE(MOE 下降至 0)状态时，Dead-time 会被作用。

Dead-time 插入藉由设定 CCxE 与 CCxNE 位使能，若有暂停电路则还有 MOE 位。配有一适用于各个信道之 8 位 dead-time 产生器(用户可藉由 ADTIM_BDTR 之 DTSEL 决定上升或下降延迟为 8/4-bit)参考波形 OCxREF，产生 2 个输出 OCx 及 OCxN，若 OCx 与 OCxN 被拉高

- ◆ OCx 输出信号与参考信号相同，上升缘除外，其延迟与参考上升缘相关。
- ◆ OCxN 输出信号为一除了上升缘以外之相对参考信号

相对于参考下降缘，其被延迟

若延迟大于作用输出带宽(OCx 或 OCxN)那么对应脉冲不会产生。

下图显示 dead-time 产生器输出信号与参考信号 OCxREF 间之关系(假设 CCxP=1,CCxNP=0,MOE=1,CCxE=1 且于这些范例中 CCxNE=1)

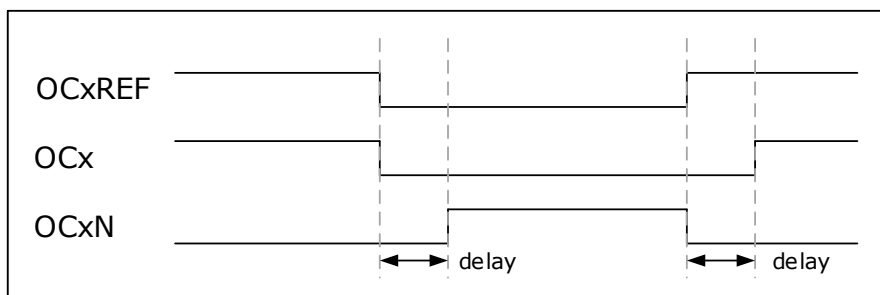


图 11-38 互补输出含 dead-time 插入

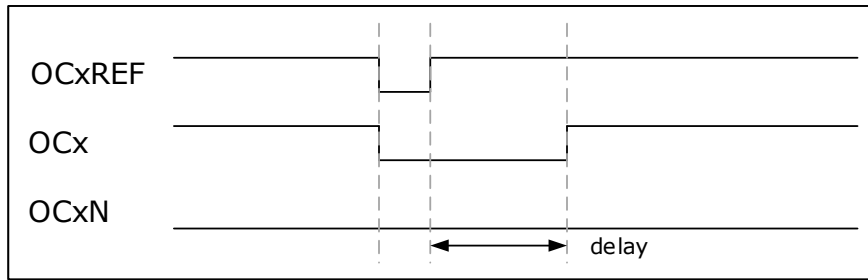


图 11-39 Dead-time 波形含大于负脉冲之延迟

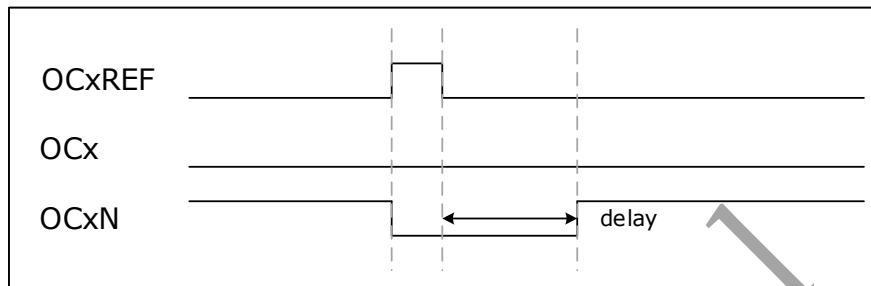


图 11-40 波形大于正脉冲之延迟

重新导向 OCxREF 至 OCx 或 OCxN

于输出模式(强制, 输出比较或 PWM), OCxREF 可透过设置 ADTIM_CCER 寄存器之 CCxE 与 CCxNE 位被重新导向至 OCx 输出或至 OCxN 输出。

可发送特定波形(如 PWM 或静止作用电平)于单一输出, 而互补维持其无作用电平。其他可能为无作用电平同时有输出或是输出作用与互补 dead-time。

注:当只有 OCxN 使能时(CCxE=0,CCxNE=1),其不互补且 OCxREF 拉高时即会作用。举例说明, 若 CCxNP=1 则 OCxN=OCxRef。另一方面, 当 OCx 与 OCxN 皆为使能(CCxE=CCxNE=1)OCx 开始作用, 当 OCxREF 为高然而 OCxN 为互补且 OCxREF 为低时开始作用。

1. 3. 13 使用暂停功能

当使用刹车功能时, 依据相应的控制位(ADTIM_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位, ADTIM_CR2 寄存器中的 OISx 和 OISxN 位), 输出使能信号和无效电平都会被修改。但无论何时, OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。详见带刹车功能的互补输出通道 OCx 和 OCxN 的控制位。

刹车源既可以是刹车输入引脚又可以是一个时钟失败事件。时钟失败事件由复位时钟控制器中的时钟安全系统(CSS)产生, 详见节时钟安全系统。

系统复位后, 刹车电路被禁止, MOE 位为低。设置 ADTIM_BDTR 寄存器中的 BKE 位可以使能刹车功能, 刹车输入信号的极性可以通过配置同一个寄存器中的 BKP 位选择。BKE 和 BKP 可以同时被修改。当写入 BKE 和 BKP 位时, 在真正写入之前会有 1 个 APB 时钟周期的延迟, 因此需要等待一个 APB 时钟周期之后, 才能正确地读回写入的位。

因为 MOE 下降沿可以是异步的，在实际信号(作用在输出端)和同步控制位(在 ADTIM_BDTR 寄存器中)之间设置了一个再同步电路。这个再同步电路会在异步信号和同步信号之间产生延迟。特别的，如果当它为低时写 MOE=1，则读出它之前必须先插入一个延时(空指令)才能读到正确的值。这是因为写入的是异步信号而读的是同步信号。

当发生刹车时(在刹车输入端出现选定的电平)，有下述动作：

- ◆ MOE 位被异步地清除，将输出置于无效状态、空闲状态或者复位状态(由 OSSI 位选择)。这个特性在 MCU 的振荡器关闭时依然有效。
- ◆ 一旦 MOE=0，每一个输出通道输出由 ADTIM_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0，则定时器释放使能输出，否则使能输出始终为高。
- ◆ 当使用互补输出时：
 - 输出首先被置于复位状态即无效的状态(取决于极性)。这是异步操作，即使定时器没有时钟时，此功能也有效。
 - 如果定时器的时钟依然存在，死区生成器将会重新生效，在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下，OCx 和 OCxN 也不能被同时驱动到有效的电平。注，因为重新同步 MOE，死区时间比通常情况下长一些(大约 2 个 ck_tim 的时钟周期)。
 - 如果 OSSI=0，定时器释放使能输出，否则保持使能输出；或一旦 CCxE 与 CCxNE 之一变高时，使能输出变为高。
- ◆ 如果设置了 ADTIM_IER 寄存器中的 BIE 位，当刹车状态标志(ADTIM_RIF 寄存器中的 BIF 位)为'1'时，则产生一个中断。如果设置了 ADTIM_DMAEN 寄存器中的 BDE 位，则产生一个 DMA 请求。
- ◆ 如果设置了 ADTIM_BDTR 寄存器中的 AOE 位，在下一个更新事件 UEV 时 MOE 位被自动置位；例如，这可以用来进行整形。否则，MOE 始终保持低直到被再次置'1'；此时，这个特性可以被用在安全方面，你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

注：刹车输入为电平有效。所以，当刹车输入有效时，不能同时(自动地或者通过软件)设置 MOE。同时，状态标志 BIF 不能被清除。

刹车由 BRK 输入产生，它的有效极性是可编程的，且由 ADTIM_BDTR 寄存器中的 BKE 位开启。

除了刹车输入和输出管理，刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数(死区长度，OCx/OCxN 极性和被禁止的状态，OCxM 配置，刹车使能和极性)。用户可以通过 ADTIM_BDTR 寄存器中的 LOCK 位，从三级保护中选择一种，参看刹车和死区寄存器(ADTIM_BDTR)。在 MCU 复位后 LOCK 位只能被修改一次。

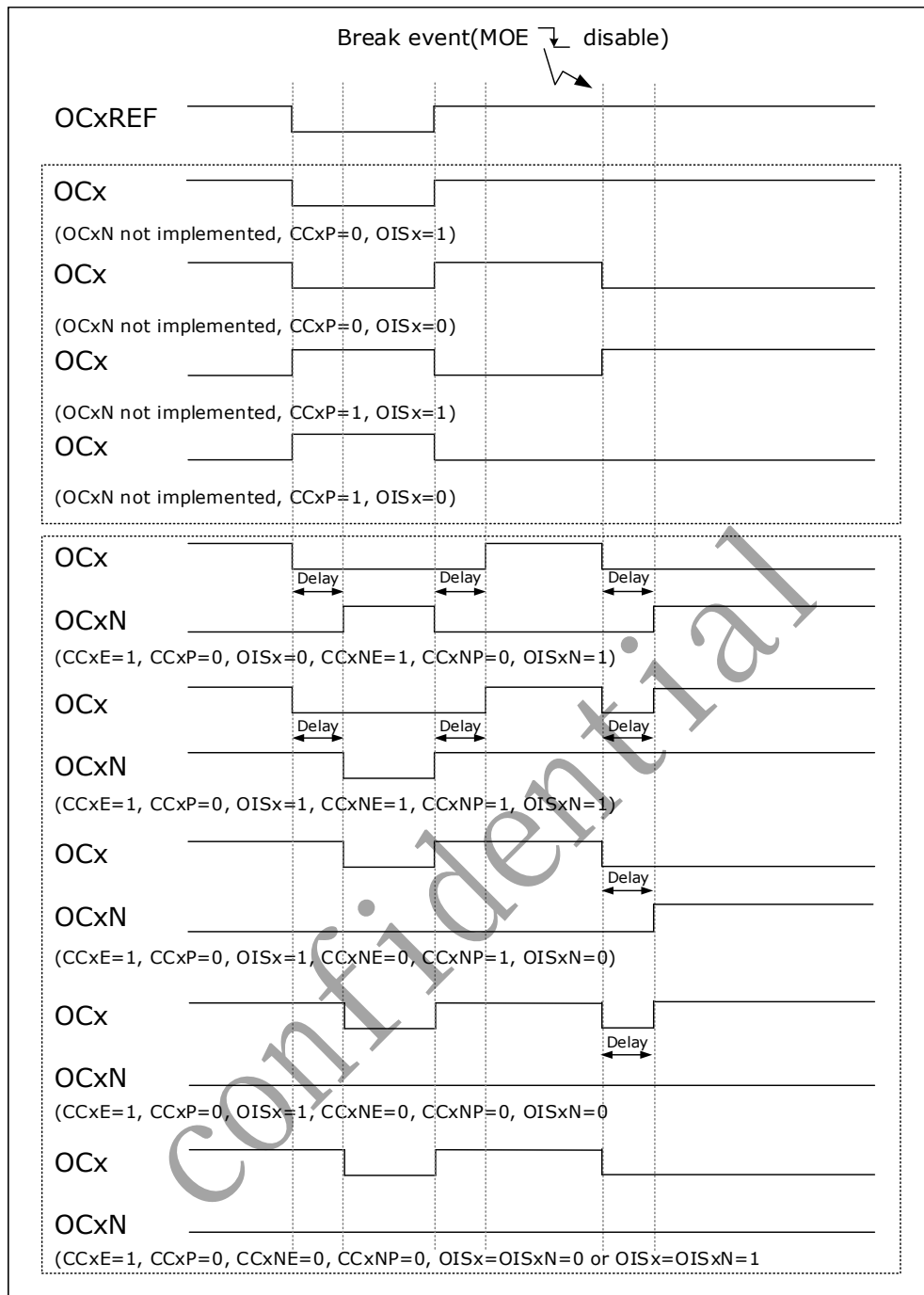


图 11-41 响应暂停之输出行为

1. 3. 14 清除外部事件 OCxREF 信号

给定信道之 OCxREF 信号，于高电平被应用于 OCREF_CLR_INPUT 时可被清除(对应 ADTIM_CCMR 寄存器之 OCxCE 使能位设置为 1)。OCxREF 保持为低，直到下一更新事件(UEV)发生。此功能仅可被用于 PWM 模式。于强制模式下则无作用。

当 ETRF 被选定，ETR 必须如下设置：

给定信道之 OCxREF 信号可透过使用高电平于 ETRF 输入而被驱使为低(对应 ADTIM_CCMR 寄存器之 OCxCE 使能位设置为 1)OCxREF 信号维持低，直到下一更新事件(UEV)发生。

此功能仅可于 PWM 模式下使用，于强制模式下无法使用。

举例说明，OCxREF 信号可被连接至用于电流控制比较器输出。于此例中，ETR 必须设置如下：

- ◆ 外部触发预除器必须保持禁用:ADTIM_SMC 寄存器之 ETPS[1:0]位设置为 ‘00’
- ◆ 外部时钟模式 2 必须禁用:ADTIM_SMCR 寄存器之 ECE 位设置为 ‘0’.
- ◆ 外部触发极性(ETP)与外部触发滤波器(ETF)可根据用户需求设置

下图标示 OCxCE 使能位值，当 ETRF 输入变成高时，OCxREF 位之信号行为。此例中定时器 ADTIM 于 PWM 模式下被编程。

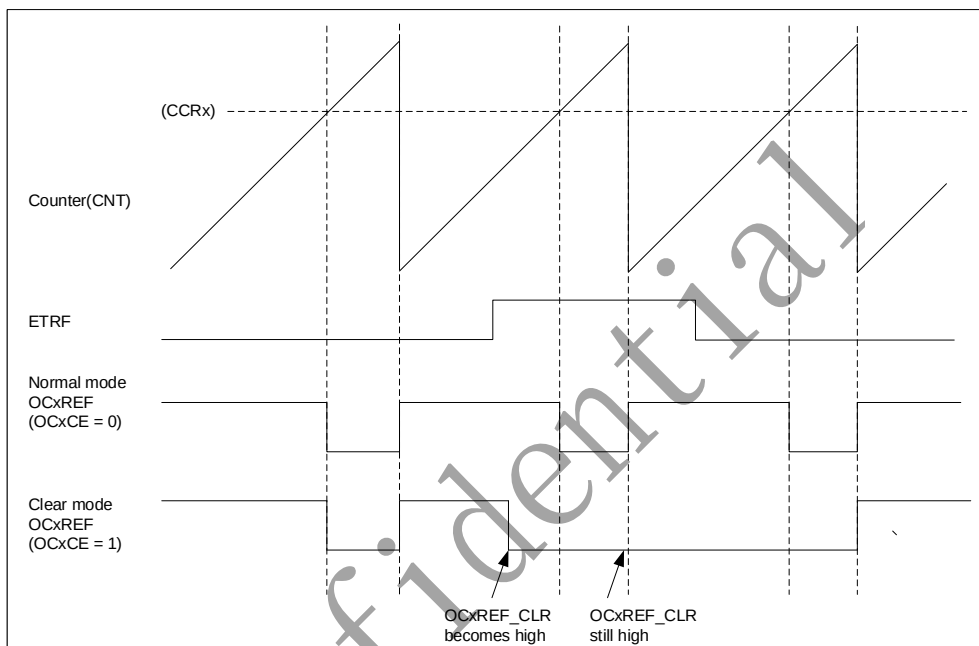


图 11-42 清除 ADTIM 的 OCxREF

注:若为占空比 100%之 PWM(若 $CCR_x > ARR$),OCxREF 会于下一个计数器上溢再度使能

1. 3. 15 产生六步 PWM 输出

当在一个通道上需要互补输出时，预装载位有 OCxM、CCxE 和 CCxNE。在发生 COM 换相事件时，这些预装载位被传送到影子寄存器位。这样你就可以预先设置好下一步骤配置，并在同一个时刻同时修更改所有信道的配置。COM 可以通过设置 ADTIM_EGR 寄存器的 COM 位由软件产生，或在 TRGI 上升沿由硬件产生。

当发生 COM 事件时会设置一个标志位(ADTIM_RIF 寄存器中的 COMIF 位)，这时如果已设置了 ADTIM_IER 寄存器的 COMIE 位，则产生一个中断；如果已设置了 ADTIM_DMAEN 寄存器的 COMDE 位，则产生一个 DMA 请求。

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

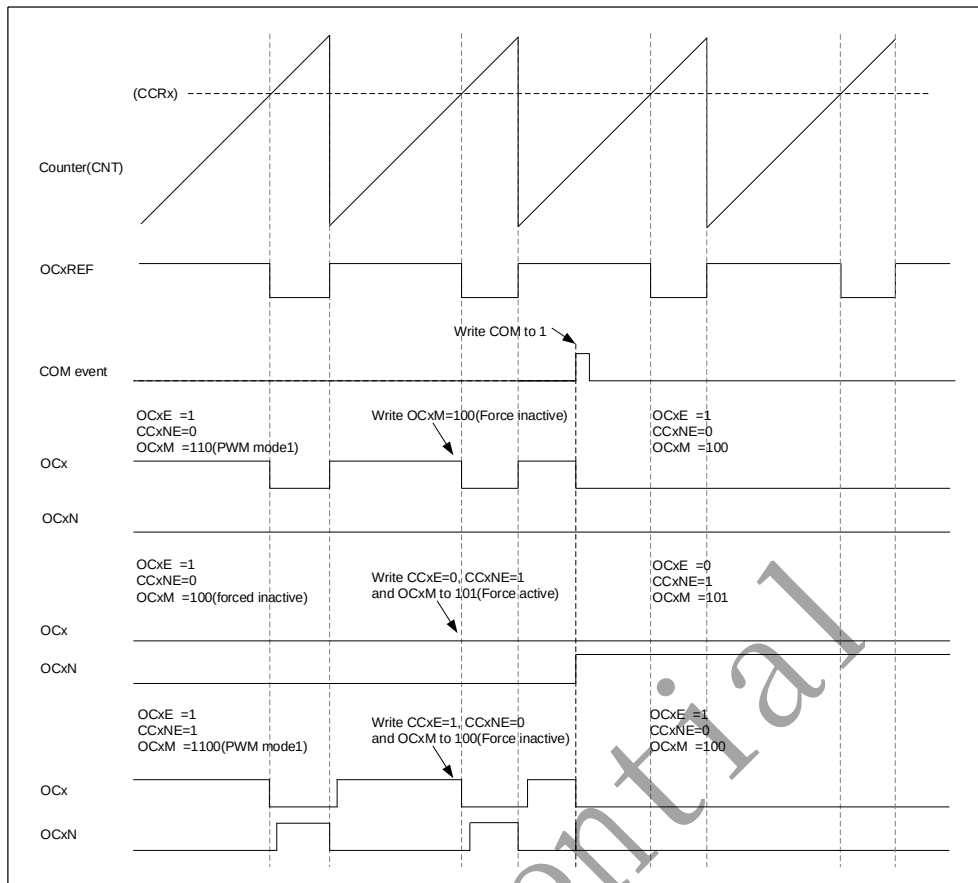


图 11-436 阶产生,COM 范例

1. 3. 16 编码器接口模式

选择编码器接口模式的方法是：如果计数器只在 TI2 的边沿计数，则置 ADTIM_SMCR 寄存器中的 SMS=001；如果只在 TI1 边沿计数，则置 SMS=010；如果计数器同时在 TI1 和 TI2 边沿计数，则置 SMS=011。

通过设置 ADTIM_CCER 寄存器中的 CC1P 和 CC2P 位，可以选择 TI1 和 TI2 极性；如果需要，还可以对输入滤波器编程。CC1P 和 CC2P 位必须保持为低。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。假定计数器已经启动(ADTIM_CR1 寄存器中的 CEN=1)，则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号；如果没有滤波和变相，则 TI1FP1=TI1，TI2FP2=TI2。根据两个输入信号的跳变顺序，产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序，计数器向上或向下计数，同时硬件对 ADTIM_CR1 寄存器的 DIR 位修改。不管计数器是依靠 TI1 还是 TI2 或者同时依靠 TI1 和 TI2 计数，在任一输入端(TI1 或者 TI2)的跳变都会重新计算 DIR 位。

编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 ADTIM_ARR 寄存器的自动装载值之间连续计数(根据方向，或是 0 到 ARR 计数，或是 ARR 到 0 计数)。所以在开始计数之前必须配置 ADTIM_ARR；同样，捕获器、比较器、预分频器、重复计数器、触发输出特性等仍工作如常。编码器模式和外部时钟模式 2 不兼容，因此不能同时操作。

在这个模式下，计数器依照增量编码器的速度和方向被自动的修改，因此计数器的内容始终指着编码器的位置。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。

有效边沿	有效边沿相对信号的电平 (TI1FP1 对 TI2,TI2FP2 对应 TI1)	TI1FP1 信号		TI2FP2 信号	
		上升沿	下降沿	上升沿	下降沿
仅在 TI1 计数	高	下降	上升	不计数	不计数
	低	上升	下降	不计数	不计数
仅在 TI2 计数	高	不计数	不计数	上升	下降
	低	不计数	不计数	下降	上升
在 TI1 和 TI2 上计数	高	下降	上升	上升	下降
	低	上升	下降	下降	上升

计数方向与编码器信号的关系

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

下图提供一计数器运作范例，显示计数信号产生与方向控制。亦可显示如何补偿输入抖动当上下缘皆被选定时。该状况可能发生于若传感器被设置于接近其中一切换点。我们假设此例设置如下：

- ◆ CC1S=01 (ADTIM_CCMR1 寄存器, IC1FP1 映射到 TI1)
- ◆ CC2S=01 (ADTIM_CCMR2 寄存器, IC2FP2 映射到 TI2)
- ◆ CC1P=0 (ADTIM_CCER 寄存器, IC1FP1 不反相, IC1FP1=TI1)
- ◆ CC2P=0 (ADTIM_CCER 寄存器, IC2FP2 不反相, IC2FP2=TI2)
- ◆ SMS=011 (ADTIM_SMCR 寄存器, 所有的输入均在上升沿和下降沿有效).
- ◆ CEN=1 (ADTIM_CR1 寄存器, 计数器使能)

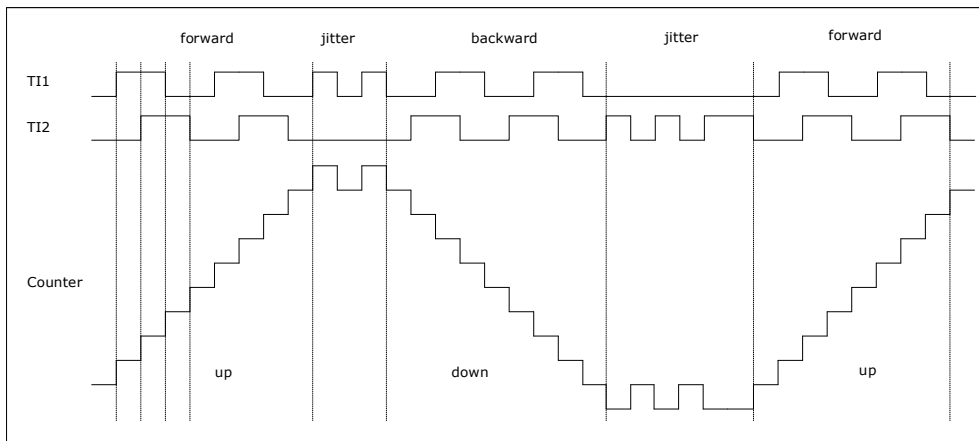


图 11-44 编码器模式下的计数器操作实例

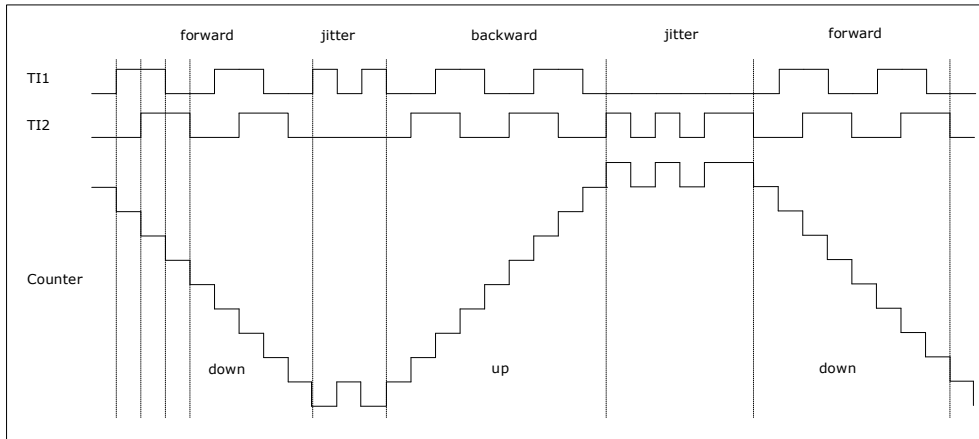


图 11-45IC1FP1 反相的编码器接口模式实例(CC1P=' 1', 其他配置与上例相同)

当设置编码器接口模式时，定时器提供传感器目前位置信息。可透过量测两个第二定时器设定为捕捉模式之编码器事件周期得到动态信息(速度，加速度与减速度)。编码器输出标示机械零位可做为此功能使用。根据两个事件间之时间可知，计数器亦能于通常时间被读取。若可行亦可透过锁存计数器值转换为第三输入捕捉寄存器(捕捉信号必须具周期性且可被另一定时器产生)。

1. 3. 17 定时器输入 XOR 功能

ADTIM_CR2 寄存器之 TI1S 位允许信道 1 输入滤波器被连接至 XOR 门输出，结合三个输入引脚 ADTIM_CH1,ADTIM_CH2 与 ADTIM_CH3。

XOR 输入可被当作所有定时器输入功能使用，如触发器或输入捕捉。于下列章节说明作为连接霍尔传感器之范例

1. 3. 18 霍尔传感器接口

使用高级控制定时器(TIM1)产生 PWM 信号驱动马达，用另一个 ADTIM(TIM2 或 TIM3)定时器作为“接口定时器”来连接霍尔传感器，请参见下图。3 个定时器输入脚(CC1、CC2、CC3)通过一个异或门连接到 TI1 输入通道(通过设置 ADTIM_CR2 寄存器中的 TI1S 位来选择)，“接口定时器”捕获这个信号。

从模式控制器被配置于复位模式，从输入是 TI1F_ED。这样每当 3 个输入之一变化时，计数器从 0 重新开始计数。由此产生一个由霍尔输入端的任何变化而触发的时间基准。

“接口定时器”模式下，捕获/比较信道 1 被配置为捕获模式，捕获信号为 TRC(捕获/比较通道(如：通道 1 输入部分))。捕获值反映了输入端两次变化之间的时间间隔，指示出了马达速度的信息。

“接口定时器”可以用来在输出模式产生一个脉冲，这个脉冲可以(通过触发一个 COM 事件)用于改变高级定时器 (TIM1) 各个通道的属性，TIM1 产生 PWM 信号驱动马达。因此，“接口定时器”通道必须编程为在一个指定的延时(输出比较或 PWM 模式)之后产生一个正脉冲，这个脉冲通过 TRGO 输出被送到高级控制定时器 (TIM1)。

举例：霍尔输入连接到 ADTIM 定时器，要求在每次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 ADTIM 的 PWM 配置。

- ◆ 置 ADTIM_CR2 寄存器的 TI1S 位为'1'，配置三个定时器输入逻辑异或到 TI1 输入，
- ◆ 时基编程：置 ADTIM_ARR 为其最大值(计数器必须通过 TI1 的变化清零)。设置预分频器得到一个最大的计数器周期，它长于传感器上的两次变化的时间间隔。
- ◆ 设置信道 1 为捕获模式(选中 TRC)：置 ADTIM_CCMR1 寄存器中 CC1S=01，如果需要，还可以设置数字滤波器。
- ◆ 设置信道 2 为 PWM 模式 2，带指定的延时：置 ADTIM_CCMR1 寄存器中的 OC2M=111 和 CC2S=00。
- ◆ 选择 OC2REF 作为 TRGO 上的触发输出：置 ADTIM_CR2 寄存器中的 MMS=101。

在高级控制寄存器 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(ADTIM_CR2 寄存器中 CCPC=1)，由触发输入控制 COM 事件(ADTIM_CR2 寄存器中 CCUS=1)。在一次 COM 事件后，下一步再写入 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。

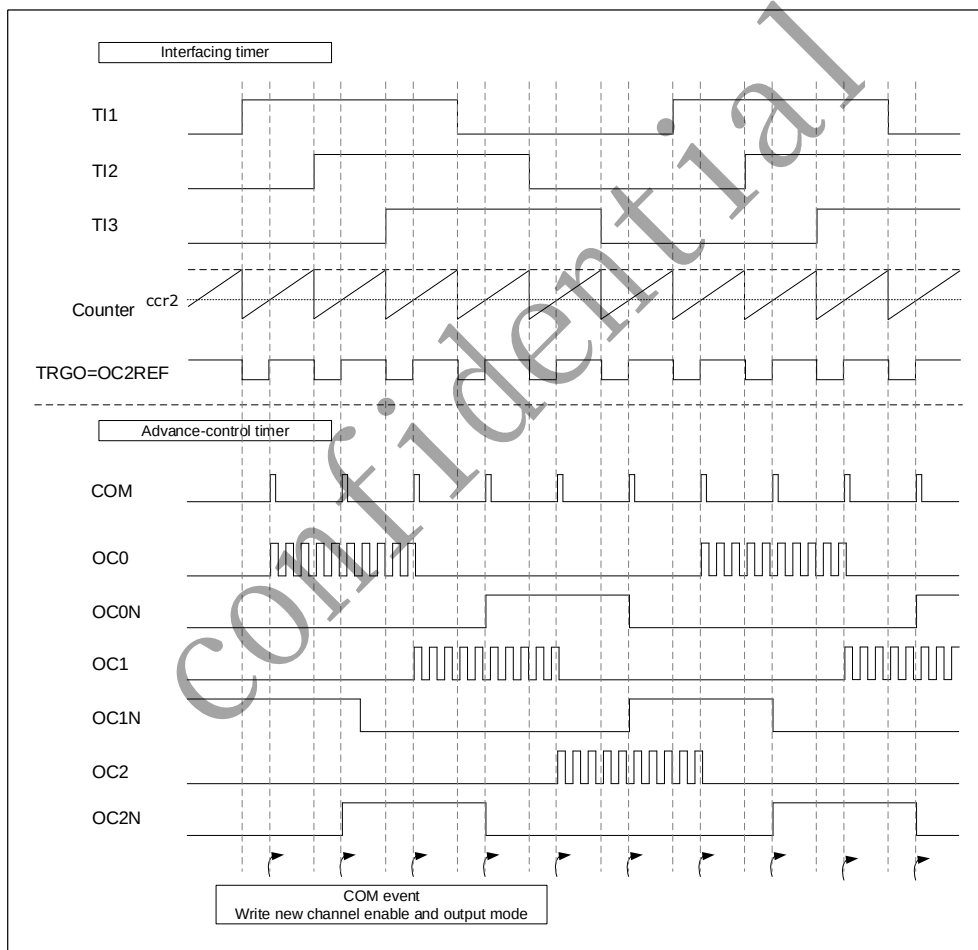


图 11-46 霍尔传感器接口范例

1. 3. 19 定时器和外部触发的同步

ADTIM 定时器能够在多种模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

1. 3. 19. 1 从模式：复位模式

在一个触发输入发生事件时，计数器和它的预分频器能够重新被初始化；同时，如果

ADTIM_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器 (ADTIM_ARR, ADTIM_CCRx) 都被更新。

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

- ◆ 配置信道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 ADTIM_CCMR1 寄存器中 CC1S=01。置 ADTIM_CCER 寄存器中 CC1P=0 和 CC1NP=0 以确定极性(只检测上升沿)。
- ◆ 置 ADTIM_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 ADTIM_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- ◆ 置 ADTIM_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(ADTIM_RIF 寄存器中的 TIF 位)被设置，并根据 ADTIM_IER 寄存器中 TIE(中断使能)位和 ADTIM_DMAEN 寄存器中 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。

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

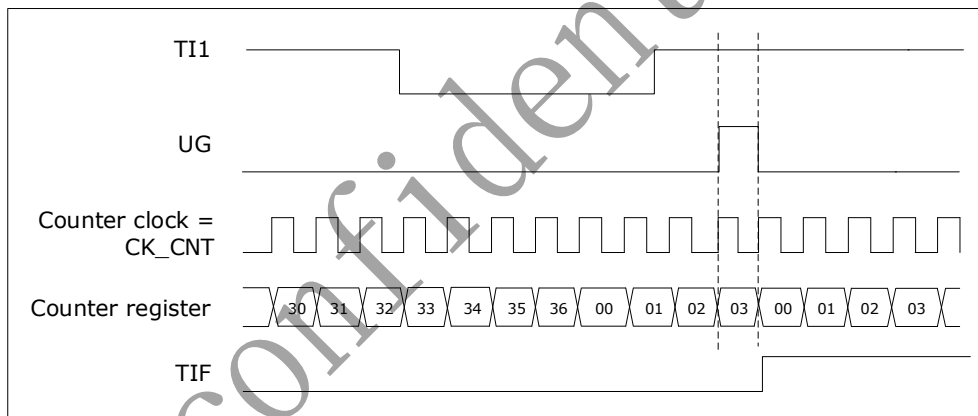


图 11-47 重置模式控制电路

1. 3. 19. 2 从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

- ◆ 配置信道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 ADTIM_CCMR1 寄存器中 CC1S=01。置 ADTIM_CCER 寄存器中 CC1P=1 和 CC1NP=0 以确定极性(只检测低电平)。
- ◆ 置 ADTIM_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 ADTIM_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- ◆ 置 ADTIM_CR1 寄存器中 CEN=1，启动计数器。(在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何)

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

TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的重同步电路。

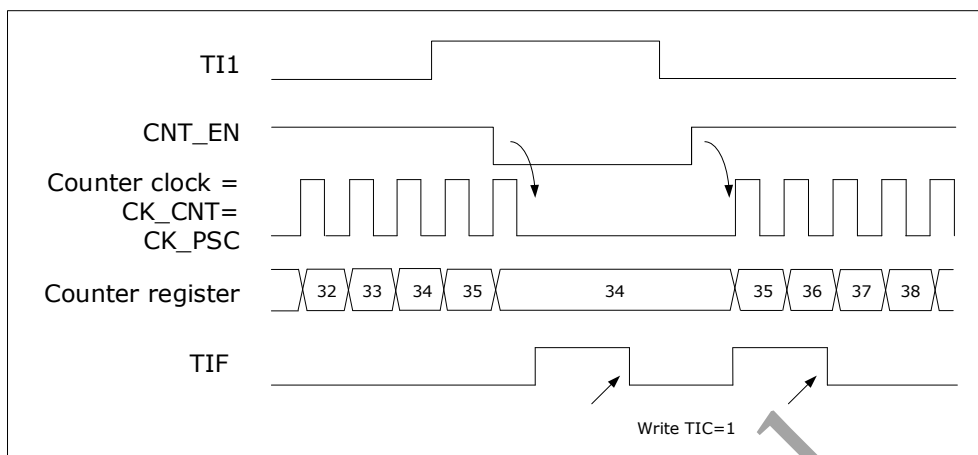


图 11-48 门控模式控制电路

1. 3. 19. 3 从模式：触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- ◆ 配置信道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 ADTIM_CCMR1 寄存器中 CC2S=01。置 ADTIM_CCER 寄存器中 CC2P=1 和 CC1NP=0 以确定极性(只检测低电平)。
- ◆ 置 ADTIM_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 ADTIM_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器按内部时钟开始计数，同时设置 TIF 标志。

TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

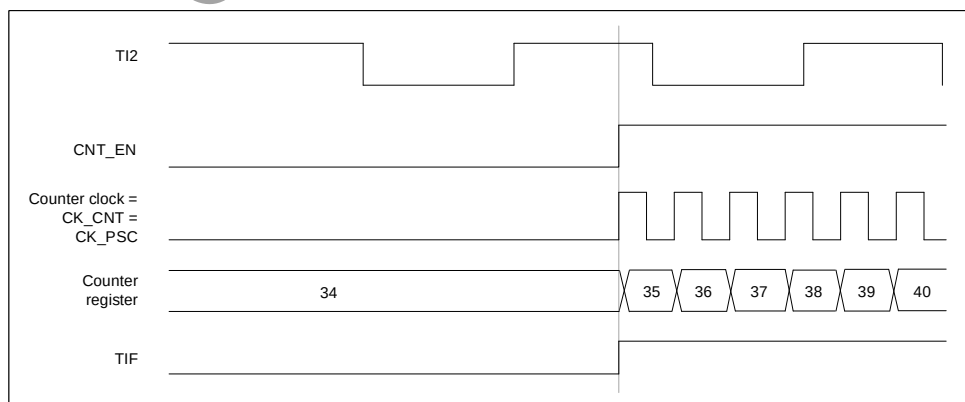


图 11-49 触发模式控制电路

1. 3. 19. 4 从模式：外部时钟模式 2+触发模式

外部时钟模式 2 可以与另一种从模式(外部时钟模式 1 和编码器模式除外)一起使用。这时，ETR 信号被用作外部时钟的输入，可以选择另一个输入作为触发输入（在复位模式、门控模式或触发模式）。不建议使用 ADTIM_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。

在下面的例子中，在 TI1 上出现一个上升沿后，向上计数器在 ETR 的每一个上升沿加一：

- ◆ 通过 ADTIM_SMCR 寄存器配置外部触发输入电路：ETF=0000：没有滤波
 - ETPS=00：不用预分频器
 - ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式 2。
- ◆ 按如下配置信道 1，检测 TI 的上升沿：
 - IC1F=0000：没有滤波
 - 触发操作中不使用捕获预分频器，不需要配置
 - 置 ADTIM_CCMR1 寄存器中 CC1S=01，选择输入捕获源
 - 置 ADTIM_CCER 寄存器中 CC1P=0 和 CC1NP=0 以确定极性(只检测上升沿)
- ◆ 置 ADTIM_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 ADTIM_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。

ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

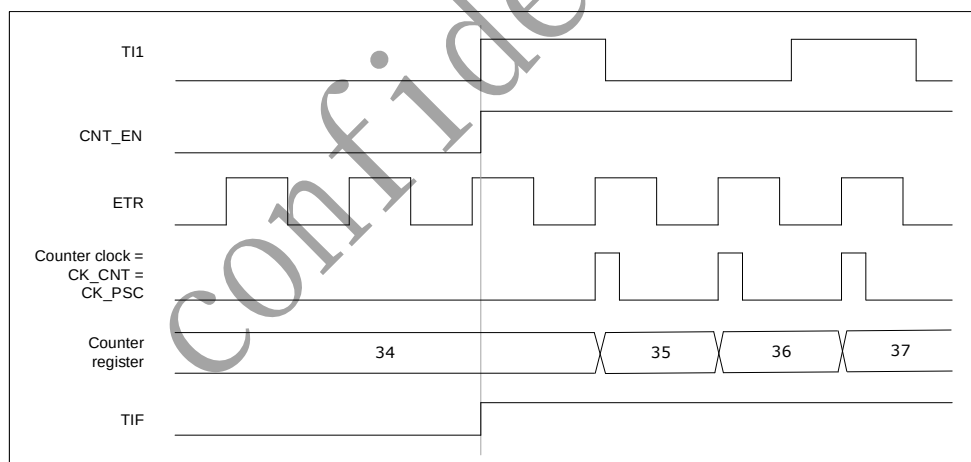


图 11-50 外部时钟模式 2+触发模式下的控制电路

1. 3. 20 定时器同步

所有 ADTIM 定时器在内部相连，用于定时器同步或链接。当一个定时器处于主模式时，它可以对另一个处于从模式的定时器的计数器进行复位、启动、停止或提供时钟等操作。

下图显示了触发选择和主模式选择模块的概况

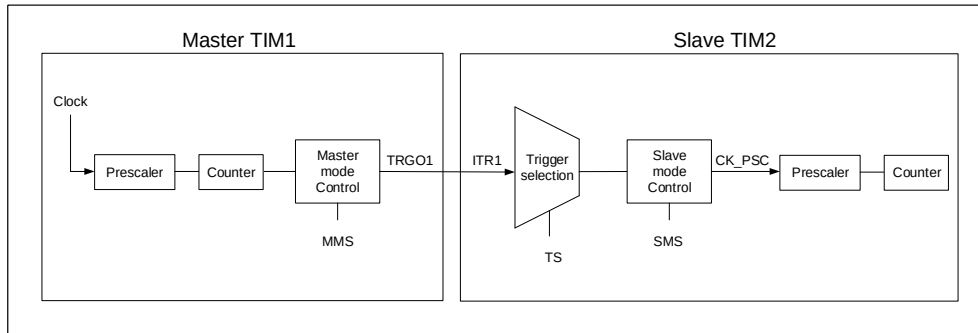


图 11-51 主/从定时器范例

1. 3. 20. 1 使用一个定时器使能另一个定时器

在这个例子中，定时器 2 的使能由定时器 1 的输出比较控制。参考图 130 的连接。只当定时器 1 的 OC1REF 为高时，定时器 2 才对分频后的内部时钟计数。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3 ($f_{CK\_CNT}=f_{CK\_INT}/3$) 得到。

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

- ◆ 配置定时器 1 的 OC1REF 波形(TIM1_CCMR1 寄存器)
- ◆ 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- ◆ 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
- ◆ 置 TIM2_CR1 寄存器的 CEN=1 以使能定时器 2
- ◆ 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1

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

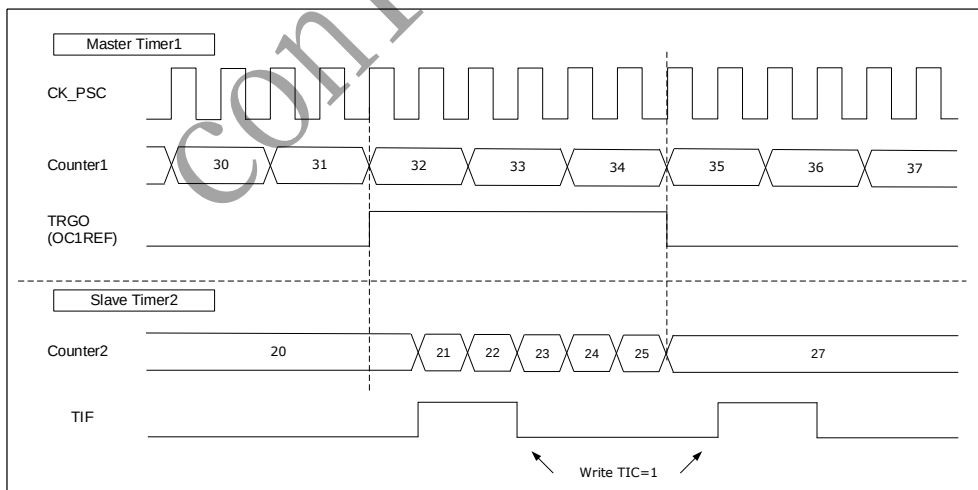


图 11-52 门控从定时器使用主定时器 OC1REF

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

在下一个例子中，需要同步定时器 1 和定时器 2。定时器 1 是主模式并从 0 开始，定时器 2 是

从模式并从 0xE7 开始；2 个定时器的预分频器系数相同。写' 0' 到 TIM1_CR1 的 CEN 位将禁止定时器 1，定时器 2 随即停止。

- ◆ 配置定时器 1 为主模式，送出输出比较 1 参考信号(OC1REF)做为触发输出(TIM1_CR2 寄存器的 MMS=100)。
- ◆ 配置定时器 1 的 OC1REF 波形(TIM1_CCMR1 寄存器)。
- ◆ 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- ◆ 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
- ◆ 置 TIM1_EGR 寄存器的 UG=1，复位定时器 1。
- ◆ 置 TIM2_EGR 寄存器的 UG=1，复位定时器 2。
- ◆ 写' 0xE7'至定时器 2 的计数器(TIM2_CNT)，初始化它为 0xE7。
- ◆ 置 TIM2_CR1 寄存器的 CEN=1 以使能定时器 2。
- ◆ 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1。
- ◆ 置 TIM1_CR1 寄存器的 CEN=0 以停止定时器 1。

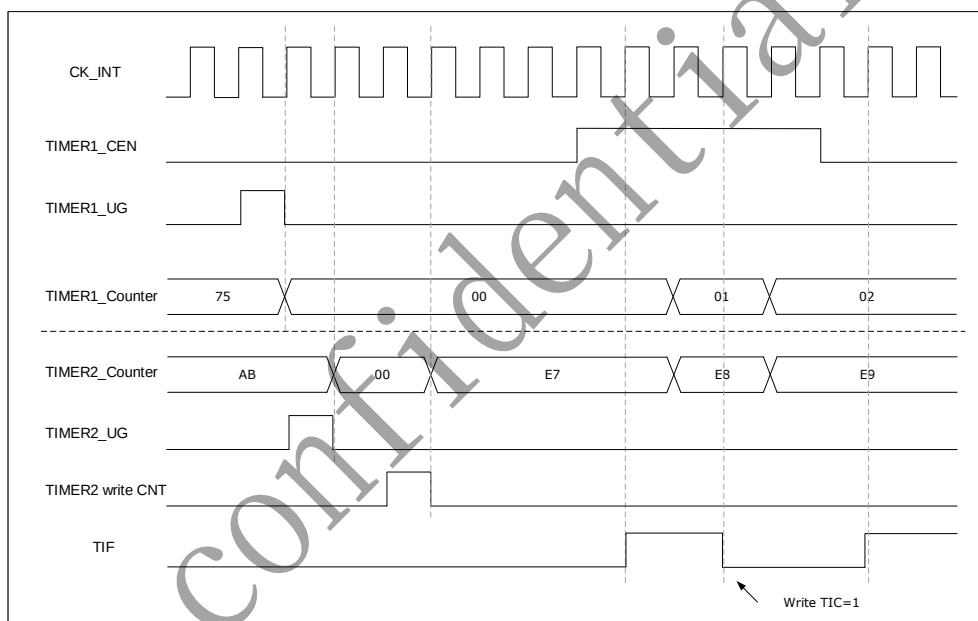


图 11-53 通过使能定时器 1 可以控制定时器 2

1. 3. 20. 2 使用一个定时器去启动另一个定时器

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

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

- ◆ 配置定时器 2 为触发模式(TIM2_SMCR 寄存器的 SMS=110)
- ◆ 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1。

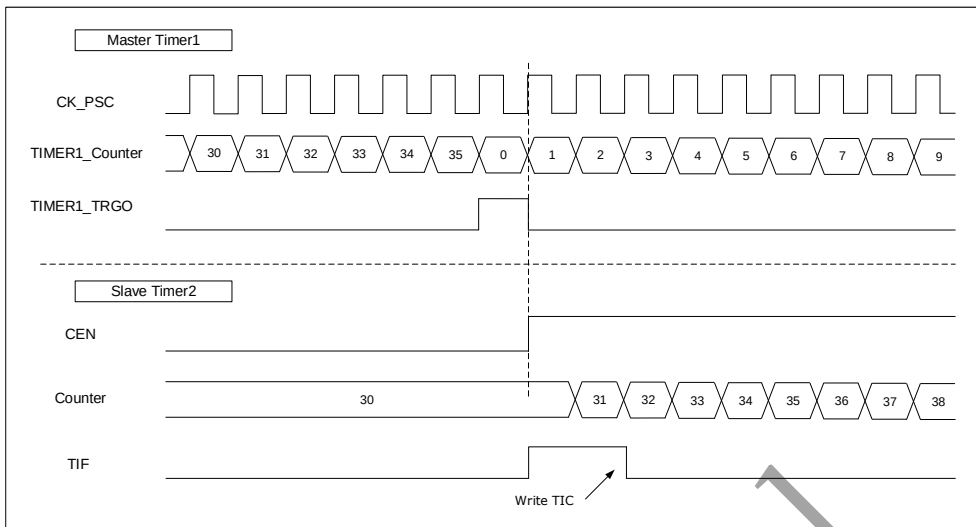


图 11-54 触发中从定时器使用主定时器更新事件

在上图例子中，可以在启动计数之前初始化两个计数器。上图显示在上上图相同配置情况下，使用触发模式而不是门控模式(TIM2_SMCR 寄存器的 SMS=110)的动作。

1. 3. 20. 3 使用一个外部触发同步地启动 2 个定时器

这个例子中当定时器 1 的 TI1 输入上升时使能定时器 1，使能定时器 1 的同时使能定时器 2，参见图 130。为保证计数器的对齐，定时器 1 必须配置为主/从模式(对应 TI1 为从，对应定时器 2 为主)：

- ◆ 配置定时器 1 为主模式，送出它的使能做为触发输出(TIM1_CR2 寄存器的 MMS=001)。
- ◆ 配置定时器 1 为从模式，从 TI1 获得输入触发(TIM1_SMCR 寄存器的 TS='101')。
- ◆ 配置定时器 1 为触发模式(TIM1_SMCR 寄存器的 SMS='110')。
- ◆ 配置定时器 1 为主/从模式，TIM1_SMCR 寄存器的 MSM='1'。
- ◆ 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- ◆ 配置定时器 2 为触发模式(TIM2_SMCR 寄存器的 SMS='110')。

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

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

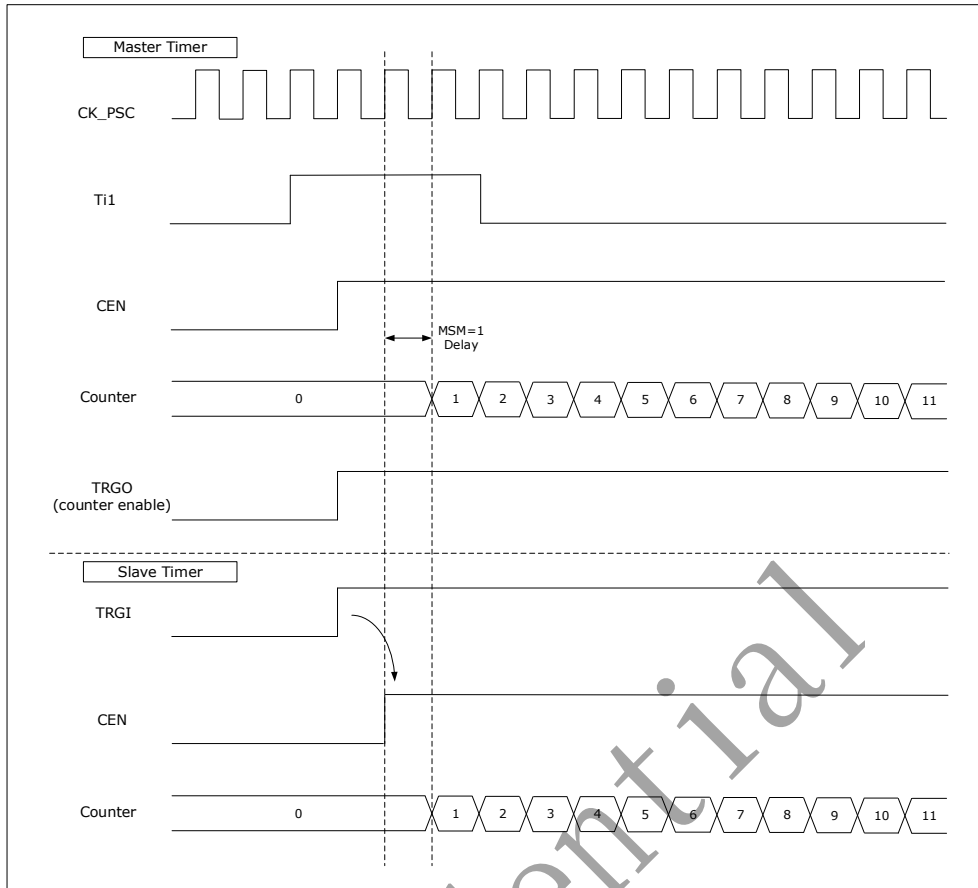


图 11-55 使用定时器 1 的 Ti1 输入触发定时器 1 和定时器 2

1. 3. 21 除错模式

当微控制器进入除错模式(MCU 中心终止),ADTIM 计数器停止计数

1. 4 ADTIM 寄存器描述

1. 4. 1 控制寄存器 1 (ADTIM_CR1)

		Address offset:		0x00											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				CMPSEL[1:0]		CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN

Name	Bit	Type	Reset value	Description
Reserved	31:12	N/A	0x00	保留, 必须始终为复位值。

CMPSEL	11:10	R/W	0x0	CMPSEL: 比较输入选择 00: 比较输入 0 10: 比较输入 1 11: 比较输入 2 11: 比较输入 3
				CKD: 时钟分频因子 这 2 位定义在定时器时钟 (CK_INT) 频率、死区时间和由死区发生器与数字滤波器 (ETR, TIx) 所用的采样时钟之间的分频比例。 00: $tDTS=tCK\_INT$ 01: $tDTS=2*tCK\_INT$ 10: $tDTS=4*tCK\_INT$ 11: 保留, 不要使用这个配置
ARPE	7	R/W	0x0	ARPE: 自动重载预装使能位 0: ADTIM_ARR 寄存器没有缓存 1: ADTIM_ARR 寄存器有缓存
CMS	6:5	R/W	0x0	CMS: 选择中央对齐模式 00: 边沿对齐模式. 计数器依据方向位 (DIR) 向上或向下计数。 01: 中央对齐模式 1. 计数器交替地向上和向下计数。配置为输出的通道 (ADTIM_CCMRx 寄存器中 CCxS=00) 的输出比较中断标志位, 只在计数器向下计数时被设置。 10: 中央对齐模式 2. 计数器交替地向上和向下计数。配置为输出的通道 (ADTIM_CCMRx 寄存器中 CCxS=00) 的输出比较中断标志位, 只在计数器向上计数时被设置。 11: 中央对齐模式 3. 计数器交替地向上和向下计数。配置为输出的通道 (ADTIM_CCMRx 寄存器中 CCxS=00) 的输出比较中断标志位, 在计数器向上和向下计数时均被设置。 注: 在计数器开启时 (CEN=1), 不允许从边沿对齐模式转换到中央对齐模式
				DIR: 计数方向 0: 计数器向上计数 1: 计数器向下计数 注: 当计数器配置为中央对齐模式或编码器模式时, 该位为只读
OPM	3	R/W	0x0	OPM: 单脉冲模式 0: 在发生更新事件时, 计数器不停止 1: 在发生下一次更新事件 (清除 CEN 位) 时, 计数器停止
URS	2	R/W	0x0	URS: 更新请求源 软件通过该位选择 UEV 事件的源 0: 如果使能了更新中断或 DMA 请求, 则下述任一事件产生更新中断或 DMA 请求: - 计数器溢出 / 下溢 - 设置 UG 位 - 从模式控制器产生的更新 1: 如果使能了更新中断或 DMA 请求, 则只有计数器溢出 / 下溢才产生更新中断或 DMA 请求。
				UDIS: 禁止更新 软件通过该位允许 / 禁止 UEV 事件的产生 0: 允许 UEV。更新 (UEV) 事件由下述任一事件产生: - 计数器溢出 / 下溢 - 设置 UG 位 - 从模式控制器产生的更新 1: 禁止 UEV。不产生更新事件, 影子寄存器 (ARR、PSC、CCRx) 保持它们的值。如果设置 UG 位或从模式控制器发出了一个硬件复位, 则计数器和预分频器被重新初始化
UDIS	1	R/W	0x0	

CEN: 使 能 计 数 器			
CEN	0	R/W	0x0
0:计数器禁止 1:计数器使能 注:外接时钟、门控模式和编码器模式只能在软件设置 CEN 位之后进行工作。而触发模式下硬件可以自动设置 CEN 位			

1. 4. 2 控制寄存器 2 (ADTIM_CR2)

		Address offset:		0x04											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	OIS4	OIS3N	OIS3	OIS2N	OIS2	OIS1N	OIS1	TI1S	MMS[2:0]			CCDS	CCUS	Reserved.	CCPC

Name	Bit	Type	Reset value	Description
Reserved	31:15	N/A	0x00	保留，必须始终为复位值。
OIS4	14	R/W	0x0	OIS4: 输出空闲状态 4 (OC4 输出) 参照 OIS1 描述
OIS3N	13	R/W	0x0	OIS3N: 输出空闲状态 3 (OC3N 输出) 参照 OIS1N 描述
OIS3	12	R/W	0x0	OIS3: 输出空闲状态 3 (OC3 输出) 参照 OIS1 描述
OIS2N	11	R/W	0x0	OIS2N: 输出空闲状态 2 (OC2N 输出) 参照 OIS1N 描述
OIS2	10	R/W	0x0	OIS2: 输出空闲状态 2 (OC2 输出) 参照 OIS1 描述
OIS1N	9	R/W	0x0	OIS1N: 输出空闲状态 1 (OC1N 输出) 0: 当 MOE=0 时，在死区时间之后 OC1N=0 1: 当 MOE=0 时，在死区时间之后 OC1N=1 注: 已经设置了 LOCK(ADTIM_BKR 寄存器 LOCK 位)级别 1、2 或 3 后，该位不能被修改
OIS1	8	R/W	0x0	OIS1: 输出空闲状态 1 (OC1 输出) 0: 当 MOE=0 时，如果完成 OC1N，则死区后 OC1=0 1: 当 MOE=0 时，如果完成 OC1N，则死区后 OC1=1 注: 已经设置了 LOCK(ADTIM_BKR 寄存器 LOCK 位)级别 1、2 或 3 后，该位不能被修改
TI1S	7	R/W	0x0	TI1S: TI1 选择 0: ADTIM_CH1 引脚连接到 TI1 输入 1: ADTIM_CH1,ADTIM_CH2 和 ADTIM_CH3 引脚经异或后连接到 TI1 输入

MMS[2:0]	6:4			R/W	0x0	MMS: 主 模 式 选 择 这 3 位用于选择在主模式下送到从定时器的同步信息 (TRGO)。可能的组合如下： 000: 复位 - ADTIM_EGR 寄存器的 UG 位被用于作为触发输出 (TRGO)。如果是触发输入产生的复位 (从模式控制器处于复位模式)，则 TRGO 上的信号相对实际的复位会有一个延迟 001: 使能 - 计数器使能信号 CNT_EN 被用于作为触发输出 (TRGO)。可用于同时启动多个定时器或控制在一段时间内使能从定时器。在门控模式下，计数器使能信号是 CEN 控制位和的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时，TRGO 上会有一个延迟，除非选择了主/从模式 (见 ADTIM_SMCR 寄存器中 MSM 位的描述)。 010: 更新 - 更新事件被选为触发输入 (TRGO) 例如，一个主定时器的时钟可以被用作一个从定时器的预分频器。 011: 比较脉冲 - 在发生一次捕获或一次比较成功时，当要设置 CC1IF 标志时 (即使它已经为高)，触发输出送出一个正脉冲 (TRGO)。 100: 比较 - OC1REF 信号被用于作为触发输出 (TRGO) 101: 比较 - OC2REF 信号被用于作为触发输出 (TRGO) 110: 比较 - OC3REF 信号被用于作为触发输出 (TRGO) 111: 比较 - OC4REF 信号被用于作为触发输出 (TRGO)
	3			R/W	0x0	CCDS: 捕 获 / 比 较 的 DMA 选 择 0: 当发生 CCx 事件时，送出 CCx 的 DMA 请求 1: 当发生更新事件时，送出 CCx 的 DMA 请求
	2			R/W	0x0	CCUS: 捕获/比较控制更新选择 0: 如果捕获/比较控制位是预装载的 (CCPC=1)，只能通过设置 COMG 位更新它们 1: 如果捕获/比较控制位是预装载的 (CCPC=1)，可以通过设置 COMG 位或 TRGI 上的一个上升沿更新它们 注：该位只对具有互补输出的通道起作用
	1			N/A	0x0	保留，必须始终为复位值
	0			R/W	0x0	CCPC: 捕 获 / 比 较 预 装 载 控 制 位 0: CCxE, CCxNE 和 OCxM 位不是预装载的 1: CCxE, CCxNE 和 OCxM 位是预装载的，设置该位后，只能在发生通信 (COM) 事件 (COMG 设置或 TRGI 是检测到上升沿，取决于 CCUS 位) 位时被更新。 注：该位只对具有互补输出的通道起作用。

1. 4. 3 从模式控制寄存器 (ADTIM_SMCR)

		Address offset:			0x08										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS[1:0]		ETF[3:0]				MSM	TS[2:0]			Reser ved	SMS[2:0]		

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，必须始终为复位值

ETP	15	R/W	0x0	ETP: 外部触发极性 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0: ETR 不反相, 高电平或上升沿有效; 1: ETR 被反相, 低电平或下降沿有效。
				ECE: 外部时钟使能位 该位启用外部时钟模式 2 0: 禁止外部时钟模式 2; 1: 使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。 注 1: 设置 ECE 位与选择外部时钟模式 1 并将 TRGI 连到 ETRF(SMS=111 和 TS=111)具有相同功效。 注 2: 下述从模式可以与外部时钟模式 2 同时使用: 复位模式, 门控模式和触发模式; 但是, 这时 TRGI 不能连到 ETRF(TS 位不能是'111')。 注 3: 外部时钟模式 1 和外部时钟模式 2 同时被使能时, 外部时钟的输入是 ETRF。
ETPS	14	R/W	0x0	ETPS: 外部触发预分频 外部触发信号 ETRP 的频率最多是 ADTIMCLK 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。
				ETF: 外部触发滤波 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。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。 0000: 无滤波器, 以 $f_{\text{SAMPLING}}=f_{\text{DTS}}$ 采样 0001: 采样频率 $f_{\text{SAMPLING}}=f_{\text{CK\_INT}}$, N=2 0010: 采样频率 $f_{\text{SAMPLING}}=f_{\text{CK\_INT}}$, N=4 0011: 采样频率 $f_{\text{SAMPLING}}=f_{\text{CK\_INT}}$, N=8 0100: 采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/2$, N=6 0110: 采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/4$, N=6 0111: 采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/4$, N=8 1001: 采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/2$, N=8 1000: 采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/8$, N=6 1001: 采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/8$, N=8 1010: 采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/16$, N=5 1011: 采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/16$, N=6 1100: 采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/16$, N=8 1101: 采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/32$, N=5 1110: 采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/32$, N=6 1111: 采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/32$, N=8 Note: Care must be taken that f_{DTS} is replaced in the formula by CK_INT when ETF[3:0] = 1, 2 or 3.
ETP	13:12	R/W	0x0	
ETF	11:8	R/W	0x0	
MSM	7	R/W	0x0	MSM: 主 / 从模式 0: 无作用; 1: 触发输入(TRGI)上的事件被延迟了, 以允许在当前定时器与它的从定时器间的完美同步(通过 TRGO)。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。

TS	6:4	R/W	0x0	TS[2:0]: 触 发 选 择 这 3 位选择用于同步计数器的触发输入。 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(TI2FP2) 111: 外部触发输入(ETRF) 更多有关 ITRx 的细节, 参见 266 页表 74, ADTIM 内部触发的连接。 注: 这些位只能在未用到(如 SMS=000)时被改变, 以避免在改变时产生错误的边沿检测。
				保留, 必须始终为复位值
Reserved	3	N/A	0x0	SMS: 从 模 式 选 择 当选择了外部信号, 触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 000: 关闭从模式 - 如果 CEN=1, 则预分频器直接由内部时钟驱动。 001: 编码器模式 1 - 根据 TI1FP1 的电平, 计数器在 TI2FP2 的边沿向上/下计数。 010: 编码器模式 2 - 根据 TI2FP2 的电平, 计数器在 TI1FP1 的边沿向上/下计数。 011: 编码器模式 3 - 根据另一个信号的输入电平, 计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100: 复位模式 - 选中的触发输入(TRGI)的上升沿重新初始化计数器, 并且产生一次更新寄存器。 101: 门控模式 - 当触发输入(TRGI)为高时, 计数器的时钟开启。一旦触发输入变为低, 则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110: 触发模式 - 计数器在触发输入 TRGI 的上升沿启动(但不复位), 只有计数器的启动是受控的。 111: 外部时钟模式 1 - 选中的触发输入(TRGI)的上升沿驱动计数器。 注: 如果 TI1F_ED 被选为触发输入(TS=100)时, 不要使用门控模式。这是因为, TI1F_ED 在每次 TI1F 变化时输出一个脉冲, 然而门控模式是要检查触发输入的电平。
SMS	2:0	R/W	0x0	

1. 4. 4 中断使能寄存器 (ADTIM_IER)

		Address offset:		0x0C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4O IE	CC3O IE	CC2O IE	CC1O IE	Reser ved	BIE	TIE	COMI E	CC4I E	CC3I E	CC2I E	CC1I E	UIE

Name	Bit	Type	Reset value	Description
Reserved	31:13	N/A	0x00	保留, 必须始终为复位值

CC4OIE	12	WO	0x0	CC4OIE: 捕捉 / 比较 4 重复捕捉中断使能 0: 无 1: CC4 重复捕捉中断使能.
CC3OIE	11	WO	0x0	CC3OIE: 捕捉 / 比较 3 重复捕捉中断使能 0: 无 1: CC3 重复捕捉中断使能.
CC2OIE	10	WO	0x0	CC2OIE: 捕捉 / 比较 2 重复捕捉中断使能 0: 无 1: CC2 重复捕捉中断使能.
CC1OIE	9	WO	0x0	CC1OIE: 捕捉 / 比较 1 重复捕捉中断使能 0: 无 1: CC1 重复捕捉中断使能.
Reserved	8	N/A	0x0	保留, 必须始终为复位值
BIE	7	WO	0x0	BIE: 刹车中断使能 0: 无 1: 刹车中断使能.
TIE	6	WO	0x0	TIE: 触发中断使能 0: 无 1: 触发中断使能.
COMIE	5	WO	0x0	COMIE: COM 中断使能 0: 无 1: COM 中断使能.
CC4IE	4	WO	0x0	CC4IE: 捕捉 / 比较 4 中断使能 0: 无 1: CC4 中断使能.
CC3IE	3	WO	0x0	CC3IE: 捕捉 / 比较 3 中断使能 0: 无 1: CC3 中断使能.
CC2IE	2	WO	0x0	CC2IE: 捕捉 / 比较 2 中断使能 0: 无 1: CC2 中断使能.
CC1IE	1	WO	0x0	CC1IE: 捕捉 / 比较 1 中断使能 0: 无 1: CC1 中断使能.
UIE	0	WO	0x0	UIE: 更新中断使能 0: 无 1: 更新中断使能.

1. 4. 5 中断禁止寄存器(ADTIM_IDR)

Address offset:		0x10													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4O ID	CC3O ID	CC2O ID	CC1O ID	Reserved	BID	TID	COMI D	CC4I D	CC3I D	CC2I D	CC1I D	UID

Name	Bit	Type	Reset value	Description
Reserved	31:13	N/A	0x00	保留, 必须始终为复位值

CC4OID	12	WO	0x0	CC4OID: 捕捉 / 比较 4 重复捕捉中断禁止 0: 无效 1: CC4 重复捕捉中断禁止.
CC3OID	11	WO	0x0	CC3OID: 捕捉 / 比较 3 重复捕捉中断禁止 0: 无效 1: CC3 重复捕捉中断禁止.
CC2OID	10	WO	0x0	CC2OID: 捕捉 / 比较 2 重复捕捉中断禁止 0: 无效 1: CC2 重复捕捉中断禁止.
CC1OID	9	WO	0x0	CC1OID: 捕捉 / 比较 1 重复捕捉中断禁止 0: 无效 1: CC1 重复捕捉中断禁止.
Reserved	8	N/A	0x0	保留, 必须始终为复位值
BID	7	WO	0x0	BID: 刹车中断禁止 0: 无效 1: 刹车中断禁止.
TID	6	WO	0x0	TID: 触发中断禁止 0: 无效 1: 触发中断禁止.
COMID	5	WO	0x0	COMID: COM 中断禁止 0: 无效 1: COM 中断禁止.
CC4ID	4	WO	0x0	CC4ID: 捕捉 / 比较 4 中断禁止 0: 无效 1: CC4 中断禁止.
CC3ID	3	WO	0x0	CC3ID: 捕捉 / 比较 3 中断禁止 0: 无效 1: CC3 中断禁止.
CC2ID	2	WO	0x0	CC2ID: 捕捉 / 比较 2 中断禁止 0: 无效 1: CC2 中断禁止.
CC1ID	1	WO	0x0	CC1ID: 捕捉 / 比较 1 中断禁止 0: 无效 1: CC1 中断禁止.
UID	0	WO	0x0	UID: 更新中断禁止 0: 无效 1: 更新中断禁止.

1. 4. 6 中断有效状态寄存器(ADTIM_IVS)

		Address offset:		0x14											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4OIS	CC3OIS	CC2OIS	CC1OIS	Reserved	BIS	TIS	COMIS	CC4IS	CC3IS	CC2IS	CC1IS	UIS

Name	Bit	Type	Reset value	Description
Reserved	31:15	N/A	0x00	保留, 必须始终为复位值

CC4OIS	12	RO	0x0	CC4OIS: 捕捉/比较 4 重复捕捉中断有效状态 0: CC4 重复捕捉中断禁止状态。 1: CC4 重复捕捉中断使能状态。 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
				CC3OIS: 捕捉/比较 3 重复捕捉中断有效状态 0: CC3 重复捕捉中断禁止状态。 1: CC3 重复捕捉中断使能状态。 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
CC3OIS	11	RO	0x0	CC2OIS: 捕捉/比较 2 重复捕捉中断有效状态 0: CC2 重复捕捉中断禁止状态。 1: CC2 重复捕捉中断使能状态。 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
				CC1OIS: 捕捉/比较 1 重复捕捉中断有效状态 0: CC1 重复捕捉中断禁止状态。 1: CC1 重复捕捉中断使能状态。 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
CC2OIS	10	RO	0x0	Reserved
				Reserved
CC1OIS	9	RO	0x0	BIS: 刹车中断有效状态 0: 刹车中断禁止状态。 1: 刹车中断使能状态。 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
				TIS: 触发中断有效状态 0: 触发中断禁止状态。 1: 触发中断使能状态。 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
Reserved	8	N/A	0x0	COMIS: COM 中断有效状态 0: COM 中断禁止状态。 1: COM 中断使能状态。 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
				CC4IS: 捕捉/比较 4 中断有效状态 0: CC4 中断禁止状态。 1: CC4 中断使能状态。 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
BIS	7	RO	0x0	CC3IS: 捕捉/比较 3 中断有效状态 0: CC3 中断禁止状态。 1: CC3 中断使能状态。 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
				CC2IS: 捕捉/比较 2 中断有效状态 0: CC2 中断禁止状态。 1: CC2 中断使能状态。 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
TIS	6	RO	0x0	CC1IS: 捕捉/比较 1 中断有效状态 0: CC1 中断禁止状态。 1: CC1 中断使能状态。 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
				UIS: 更新中断有效状态 0: 更新中断禁止状态。 1: 更新中断使能状态。 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
COMIS	5	RO	0x0	
CC4IS	4	RO	0x0	
CC3IS	3	RO	0x0	
CC2IS	2	RO	0x0	
CC1IS	1	RO	0x0	
UIS	0	RO	0x0	

1. 4. 7 原始中断标志寄存器 (ADTIM_RIF)

Address offset:		0x18											
Reset value:		0x0000 0000											

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4OIF	CC3OIF	CC2OIF	CC1OIF	Reserved	BIF	TIF	COMIF	CC4IF	CC3IF	CC2IF	CC1IF	UIF

Name	Bit	Type	Reset value	Description
Reserved	31:13	N/A	0x00	保留，必须始终为复位值
CC4OIF	12	RO	0x0	CC4OIF: 捕捉 / 比较 4 重复捕捉原始中断标志 参考 CC1OIF 描述
CC3OIF	11	RO	0x0	CC3OIF: 捕捉 / 比较 3 重复捕捉原始中断标志 参考 CC1OIF 描述
CC2OIF	10	RO	0x0	CC2OIF: 捕捉 / 比较 2 重复捕捉原始中断标志 参考 CC1OIF 描述
CC1OIF	9	RO	0x0	CC1OIF: 捕捉 / 比较 1 重复捕捉原始中断标志 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。 ADTIM_ICR 寄存器相应位软件写“1”可清除该标志 0：无重复中断产生 1：当 CC1IF 的状态已经为‘1’，计数器的值被捕获到 ADTIM_CCR1 寄存器。
Reserved	8	N/A	0x0	保留，必须始终为复位值
BIF	7	RO	0x0	BIF: 刹车原始中断标志 一旦刹车输入有效，由硬件对该位置‘1’。如果刹车输入无效，ADTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0：无刹车事件产生； 1：刹车输入上检测到有效电平。
TIF	6	RO	0x0	TIF: 触发原始中断标志 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’。ADTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0：无触发器事件产生； 1：触发中断等待响应。
COMIF	5	RO	0x0	COMIF: COM 原始中断标志 一旦产生 COM 事件(当捕获/比较控制位：CCxE、CCxNE、OCxM 已被更新)该位由硬件置‘1’。ADTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0：无 COM 事件产生； 1：COM 中断等待响应。
CC4IF	4	RO	0x0	CC4IF: 捕捉 / 比较 4 原始中断标志 参考 CC1IF 描述
CC3IF	3	RO	0x0	CC3IF: 捕捉 / 比较 3 原始中断标志 参考 CC1IF 描述
CC2IF	2	RO	0x0	CC2IF: 捕捉 / 比较 2 原始中断标志 参考 CC1IF 描述

CC1IF	1	RO	0x0	捕捉 / 比较 1 原始中断标志 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外(参考 ADTIM_CR1 寄存器的 CMS 位)。它由软件清 0'。 0：无匹配发生； 1：ADTIM_CNT 的值与 ADTIM_CCR1 的值匹配。当 ADTIM_CCR1 的内容大于 ADTIM_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置 1'，ADTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0：无输入捕获产生； 1：计数器值被捕获至 ADTIM_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
	0	RO	0x0	UIF：更新原始中断标志 当产生更新事件时该位由硬件置 1'。它由软件清 0'。 0：无更新事件产生； 1：更新中断等待响应。当寄存器被更新时该位由硬件置 1'： 一若 ADTIM_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 一若 ADTIM_CR1 寄存器的 URS=0、UDIS=0，当设置 ADTIM_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 一若 ADTIM_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

1. 4. 8 屏蔽中断标志寄存器 (ADTIM_IFM)

		Address offset:		0x1C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4OIM	CC3OIM	CC2OIM	CC1OIM	Reserved	BIM	TIM	COMIM	CC4IM	CC3IM	CC2IM	CC1IM	UIM

Name	Bit	Type	Reset value	Description
Reserved	31:13	N/A	0x00	保留，必须始终为复位值
CC4OIM	12	RO	0x0	CC4OIM: 捕捉 / 比较 4 重复捕捉屏蔽中断标志 参考 CC1OIM 描述
CC3OIM	11	RO	0x0	CC3OIM: 捕捉 / 比较 3 重复捕捉屏蔽中断标志 参考 CC1OIM 描述
CC2OIM	10	RO	0x0	CC2OIM: 捕捉 / 比较 2 重复捕捉屏蔽中断标志 参考 CC1OIM 描述
CC1OIM	9	RO	0x0	CC1OIM: 捕捉/比较 1 重复捕捉屏蔽中断标志 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。 ADTIM_ICR 寄存器相应位软件写“1”可清除该标志 0：无重复中断产生 1：当 CC1IF 的状态已经为 1'，计数器的值被捕获到 ADTIM_CCR1 寄存器。

Reserved	8	N/A	0x0	保留，必须始终为复位值
BIM	7	RO	0x0	BIM: 刹车屏蔽中断标志 一旦刹车输入有效，由硬件对该位置'1'。如果刹车输入无效，ADTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0: 无刹车事件产生； 1: 刹车输入上检测到有效电平。
TIM	6	RO	0x0	TIM: 触发屏蔽中断标志 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'。ADTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0: 无触发器事件产生； 1: 触发中断等待响应。
COMIM	5	RO	0x0	COMIM: COM 屏蔽中断标志 一旦产生 COM 事件(当捕获/比较控制位: CCxE、CCxNE、OCxM 已被更新)该位由硬件置'1'。ADTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0: 无 COM 事件产生； 1: COM 中断等待响应。
CC4IM	4	RO	0x0	CC4IM: 捕捉 / 比较 4 屏蔽中断标志 参考 CC1IM 描述
CC3IM	3	RO	0x0	CC3IM: 捕捉 / 比较 3 屏蔽中断标志 参考 CC1IM 描述
CC2IM	2	RO	0x0	CC2IM: 捕捉 / 比较 2 屏蔽中断标志 参考 CC1IM 描述
CC1IM	1	RO	0x0	CC1IM : 捕捉 / 比较 1 屏蔽中断标志 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外(参考 ADTIM_CR1 寄存器的 CMS 位)。它由软件清'0'。 0: 无匹配发生； 1: ADTIM_CNT 的值与 ADTIM_CCR1 的值匹配。当 ADTIM_CCR1 的内容大于 ADTIM_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置'1'，ADTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0: 无输入捕获产生； 1: 计数器值被捕获至 ADTIM_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
UIM	0	RO	0x0	UIM: 更新屏蔽中断标志 当产生更新事件时该位由硬件置'1'。它由软件清'0'。 0: 无更新事件产生； 1: 更新中断等待响应。当寄存器被更新时该位由硬件置'1': 一若 ADTIM_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 一若 ADTIM_CR1 寄存器的 URS=0、UDIS=0，当设置 ADTIM_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 一若 ADTIM_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

1. 4. 9 中断清除寄存器 (ADTIM_ICR)

	Address offset:	0x20

Reset value:					0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4OIC	CC3OIC	CC2OIC	CC1OIC	Reserved	BIC	TIC	COMIC	CC4IC	CC3IC	CC2IC	CC1IC	UIC

Name	Bit	Type	Reset value	Description
Reserved	31:13	N/A	0x00	保留，必须始终为复位值.
CC4OIC	12	W1C	0x0	CC4OIC: 捕捉 / 比较 4 重复捕捉中断清除寄存器 参考 CC1OIC 描述
CC3OIC	11	W1C	0x0	CC3OIC: 捕捉 / 比较 3 重复捕捉中断清除寄存器 参考 CC1OIC 描述
CC2OIC	10	W1C	0x0	CC2OIC: 捕捉 / 比较 2 重复捕捉中断清除寄存器 参考 CC1OIC 描述
CC1OIC	9	W1C	0x0	CC1OIC: 捕捉 / 比较 1 重复捕捉中断清除寄存器 0: 无效果 1: CC1OI 中断清除.
Reserved	8	N/A	0x0	保留，必须始终为复位值
BIC	7	W1C	0x0	BIC: 刹车中断清除寄存器 0: 无效果 1: 刹车中断清除.
TIC	6	W1C	0x0	TIC: 触发中断清除寄存器 0: 无效果 1: 触发中断清除.
COMIC	5	W1C	0x0	COMIC: COM 中断清除寄存器 0: 无效果 1: COM 中断清除.
CC4IC	4	W1C	0x0	CC4IC: 捕捉 / 比较 4 中断清除寄存器 参考 CC1IC 描述
CC3IC	3	W1C	0x0	CC3IC: 捕捉 / 比较 3 中断清除寄存器 参考 CC1IC 描述
CC2IC	2	W1C	0x0	CC2IC: 捕捉 / 比较 2 中断清除寄存器 参考 CC1IC 描述
CC1IC	1	W1C	0x0	CC1IC: 捕捉 / 比较 1 中断清除寄存器 0: 无效果 1: 捕捉/比较 1 中断清除.
UIC	0	W1C	0x0	UIC: 更新中断清除寄存器 0: 无效果 1: 更新中断清除.

1. 4. 10 事件产生寄存器(ADTIM_EGR)

Address offset:					0x24										
Reset value:					0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BG	TG	COMG	CC4G	CC3G	CC2G	CC1G	UG

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留，必须始终为复位值
BG	7	WO	0x0	BG: 产生刹车事件 该位由软件置'1'，用于产生一个刹车事件，由硬件自动清'0'。 0: 无动作; 1: 产生一个刹车事件。此时 MOE=0、BIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
TG	6	WO	0x0	TG: 触发产生 该位由软件置'1'，用于产生一个事件，由硬件自动清'0'。 0: 无动作; 1: ADTIM_SR 中 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
COMG	5	WO	0x0	COMG: 捕捉 / 比较控制更新产生 该位由软件置'1'，由硬件自动清'0'。 0: 无动作; 1: 当 CCPC=1，允许更新 CCxE、CCxNE、OCxM 位。 注：该位只对拥有互补输出的通道有效。
CC4G	4	WO	0x0	CC4G: 捕捉 / 比较 4 产生 参考 CC1G 描述
CC3G	3	WO	0x0	CC3G: 捕捉 / 比较 3 产生 参考 CC1G 描述
CC2G	2	WO	0x0	CC2G: 捕捉 / 比较 2 产生 参考 CC1G 描述
CC1G	1	WO	0x0	CC1G: 捕捉 / 比较 1 产生 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 1 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 ADTIM_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。
UG	0	WO	0x0	UG: 产生更新事件 该位由软件置'1'，由硬件自动清'0'。 0: 无动作; 1: 重新初始化计数器，并产生一个(寄存器)更新事件。注意预分频器的计数器也被清'0' (但是预分频系数不变)。若在中心对称模式下或 DIR=0(向上计数)则计数器被清'0'；若 DIR=1(向下计数)则计数器取 ADTIM_ARR 的值。

1. 4. 11 捕捉/比较模式寄存器 1 (ADTIM_CCMR1)

Address offset:		0x28													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2CE	OC2M[2:0]			OC2PE	OC2FE	CC2S[1:0]		OC1CE	OC1M[2:0]			OC1PE	OC1FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				IC1F[3:0]			IC1PSC[1:0]				

输出比较模式

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，必须始终为复位值
OC2CE	15	R/W	0x0	OC2CE: 输出比较 2 清除 0 使能 参考 OC1CE 描述
OC2M	14:12	R/W	0x0	OC2M: 输出比较模式 2 参考 OC1M 描述
OC2PE	11	R/W	0x0	OC2PE: 输出比较 2 预装使能 参考 OC1PE 描述
OC2FE	10	R/W	0x0	OC2FE: 输出比较 2 快速使能 参考 OC1FE 描述
CC2S	9:8	R/W	0x0	CC2S: 捕捉 / 比较 2 选择 该位定义通道的方向(输入/输出)，及输入信号的选择： 00: CC2 通道被配置为输出； 01: CC2 通道被配置为输入，IC2 映射在 TI2 上； 10: CC2 通道被配置为输入，IC2 映射在 TI1 上； 11: CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 ADTIM_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(ADTIM_CCER 寄存器的 CC2E=0)才是可写的。
OC1CE	7	R/W	0x0	OC1CE: 输出比较 1 清除 0 使能 0: OC1REF 不受 ETRF 输入的影响； 1: 一旦检测到 ETRF 输入高电平，清除 OC1REF=0。
OC1M	6:4	R/W	0x0	OC1M: 输出比较模式 1 该 3 位定义了输出参考信号 OC1REF 的动作，而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效，而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000: 冻结。输出比较寄存器 ADTIM_CCR1 与计数器 ADTIM_CNT 间的比较对 OC1REF 不起作用； 001: 匹配时设置通道 1 为有效电平。当计数器 ADTIM_CNT 的值与捕获/比较寄存器 1(ADTIM_CCR1)相同时，强制 OC1REF 为高。 010: 匹配时设置通道 1 为无效电平。当计数器 ADTIM_CNT 的值与捕获/比较寄存器 1(ADTIM_CCR1)相同时，强制 OC1REF 为低。 011: 翻转。当 ADTIM_CCR1=ADTIM_CNT 时，翻转 OC1REF 的电平。 100: 强制为无效电平。强制 OC1REF 为低。 101: 强制为有效电平。强制 OC1REF 为高。 110: PWM 模式 1 — 在向上计数时，一旦 ADTIM_CNT<ADTIM_CCR1 时通道 1 为有效电平，否则为无效电平；在向下计数时，一旦 ADTIM_CNT>ADTIM_CCR1 时通道 1 为无效电平(OC1REF=0)，否则为有效电平(OC1REF=1)。 111: PWM 模式 2 — 在向上计数时，一旦 ADTIM_CNT<ADTIM_CCR1 时通道 1 为无效电平，否则为有效电平；在向下计数时，一旦 ADTIM_CNT>ADTIM_CCR1 时通道 1 为有效电平，否则为无效电平。 注 1: 一旦 LOCK 级别设为 3(ADTIM_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2: 在 PWM 模式 1 或 PWM 模式 2 中，只有当比较结果改变了或在输出比较模式从中冻结模式切换到 PWM 模式时，OC1REF 电平才改变。

注 3：在通道有互补输出时，此时被预装载。如果设置了 ADTIM_CR2 中的 CCPC 位，OC1M 的有效位只能在发生 COM 事件时才能从预装位得到新的值。

OC1PE	3	R/W	0x0	OC1PE: 输出比较 1 预装使能 0: 禁止 ADTIM_CCR1 寄存器的预装载功能，可随时写入 ADTIM_CCR1 寄存器，并且新写入的数值立即起作用。 1: 开启 ADTIM_CCR1 寄存器的预装载功能，读写操作仅对预装载寄存器操作，ADTIM_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1: 一旦 LOCK 级别设为 3 (ADTIM_BDTR 寄存器中的 LOCK 位) 并且 CC1S=00 (该通道配置成输出) 则该位不能被修改。 注 2: 仅在单脉冲模式下 (ADTIM_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
OC1FE	2	R/W	0x0	OC1FE: 输出比较 1 快速使能 该位用于加快 CC 输出对触发输入事件的响应。 0: CC1 的正常操作依赖于计数器与 CCR1 的值，即使工作于触发器状态。当触发器的输入有一个有效沿时，激活 CC1 输出的最小延时为 5 个时钟周期。 1: 输入到触发器的有效沿的作用就象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。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
CC1S	1:0	R/W	0x0	CC1S: 捕捉 / 比较 1 选择 这 2 位定义通道的方向 (输入/输出)，及输入脚的选择： 00: CC1 通道被配置为输出； 01: CC1 通道被配置为输入，IC1 映射在 TI1 上； 10: CC1 通道被配置为输入，IC1 映射在 TI2 上； 11: CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时 (由 ADTIM_SMCR 寄存器的 TS 位选择)。 注: CC1S 仅在通道关闭时 (ADTIM_CCER 寄存器的 CC1E=0) 才是可写的。

输入捕捉模式

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，必须始终为复位值
IC2F	15:12	R/W	0x0	IC2F: 输入捕捉 2 滤波器 参考 IC1F 描述

IC2PSC	11:10	R/W	0x0	IC2PSC: 输入捕捉 2 预分频器 参考 IC1PSC 描述
CC2S	9:8	R/W	0x0	CC2S: 捕捉 / 比较 2 选择 这 2 位定义通道的方向(输入/输出), 及输入脚的选择: 00: CC2 通道被配置为输出; 01: CC2 通道被配置为输入, IC2 映射在 TI2 上; 10: CC2 通道被配置为输入, IC2 映射在 TI1 上; 11: CC2 通道被配置为输入, IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 ADTIM_SMCr 寄存器的 TS 位选择)。 注: CC2S 仅在通道关闭时(ADTIM_CCER 寄存器的 CC2E=0)才是可写的。
IC1F	7:4	R/W	0x0	IC1F: 输入捕捉 1 滤波器 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成, 它记录到 N 个事件后会产生一个输出的跳变: 0000: 没有过滤器, $f_{\text{SAMPLING}} = f_{\text{DTS}}$ 0001: $f_{\text{SAMPLING}} = f_{\text{CK\_INT}}, N = 2$ 0010: $f_{\text{SAMPLING}} = f_{\text{CK\_INT}}, N = 4$ 0011: $f_{\text{SAMPLING}} = f_{\text{CK\_INT}}, N = 8$ 0100: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 2, N = 6$ 0101: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 2, N = 8$ 0110: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 4, N = 6$ 0111: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 4, N = 8$ 1000: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 8, N = 6$ 1001: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 8, N = 8$ 1010: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 16, N = 5$ 1011: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 16, N = 6$ 1100: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 16, N = 8$ 1101: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 32, N = 5$ 1110: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 32, N = 6$ 1111: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 32, N = 8$ 注意: 当 ICxF[3:0] = 1、2 或 3 时, 必须注意公式中 f_{DTS} 用 $f_{\text{CK\_INT}}$ 代替。
IC1PSC	3:2	R/W	0x0	IC1PSC: 输入捕捉 1 预分频器 这 2 位定义了 CC1 输入(IC1)的预分频系数。一旦 CC1E=0(ADTIM_CCER 寄存器中), 则预分频器复位。 00: 无预分频器, 捕获输入口上检测到的每一个边沿都触发一次捕获; 01: 每 2 个事件触发一次捕获; 10: 每 4 个事件触发一次捕获; 11: 每 8 个事件触发一次捕获。
CC1S	1:0	R/W	0x0	CC1S: 捕捉/比较 1 选择 这 2 位定义通道的方向(输入/输出), 及输入脚的选择: 00: CC1 通道被配置为输出; 01: CC1 通道被配置为输入, IC1 映射在 TI1 上; 10: CC1 通道被配置为输入, IC1 映射在 TI2 上; 11: CC1 通道被配置为输入, IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 ADTIM_SMCr 寄存器的 TS 位选择)。 注: CC1S 仅在通道关闭时(ADTIM_CCER 寄存器的 CC1E=0)才是可写的。

通道可用于输入(捕获模式)或输出(比较模式), 通道的方向由相应的 CCxS 位定义。该寄存器其它位的作用在输入和输出模式下不同。OCxx 描述了通道在输出模式下的功能, ICxx 描述了通道在输入模式下的功能。因此必须注意, 同一个位在输出模式和输入模式下的功能是不同的。

1. 4. 12 捕捉/比较模式寄存器 2 (ADTIM_CCMR2)

		Address offset:			0x2C										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4 CE	OC4M[2:0]			OC4 PE	OC4 FE	CC2S[1:0]		OC3 CE	OC3M[2:0]			OC3 PE	OC3 FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]				IC3F[3:0]			IC3PSC[1:0]				

输出比较模式

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，必须始终为复位值
OC4CE	15	R/W	0x0	OC4CE: 输出比较 4 清除 0 使能 参考 OC1CE 描述
OC4M	14:12	R/W	0x0	OC4M: 输出比较 4 模式 参考 OC1M 描述
OC4PE	11	R/W	0x0	OC4PE: 输出比较 4 预分频使能 参考 OC1PE 描述
OC4FE	10	R/W	0x0	OC4FE: 输出比较 4 快速使能 参考 OC1FE 描述
CC4S	9:8	R/W	0x0	CC4S: 捕捉 / 比较 4 选择 该 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00: CC4 通道被配置为输出； 01: CC4 通道被配置为输入，IC4 映射在 TI4 上； 10: CC4 通道被配置为输入，IC4 映射在 TI3 上； 11: CC4 通道被配置为输入，IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 ADTIM_SMCR 寄存器的 TS 位选择)。 注：CC4S 仅在通道关闭时(ADTIM_CCER 寄存器的 CC4E=0)才是可写的。
OC3CE	7	R/W	0x0	OC3CE: 输出比较 3 清除 0 使能 参考 OC1CE 描述
OC3M	6:4	R/W	0x0	OC3M: 输出比较 3 模式 参考 OC1M 描述
OC3PE	3	R/W	0x0	OC3PE: 输出比较 3 预分频使能 参考 OC1PE 描述
OC3FE	2	R/W	0x0	OC3FE: 输出比较 3 快速使能 参考 OC1FE 描述
CC3S	1:0	R/W	0x0	CC3S: 捕捉 / 比较 3 选择 该 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00: CC3 通道被配置为输出； 01: CC3 通道被配置为输入，IC4 映射在 TI3 上； 10: CC3 通道被配置为输入，IC4 映射在 TI4 上； 11: CC3 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 ADTIM_SMCR 寄存器的 TS 位选择)。 注：CC3S 仅在通道关闭时(ADTIM_CCER 寄存器的 CC3E=0)才是可写的。

输入捕捉模式

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，必须始终为复位值

IC4F	15:12	R/W	0x0	IC4F: 输入捕捉 4 滤波器 参考 IC1F 描述
IC4PSC	11:10	R/W	0x0	IC4PSC: 输入捕捉 4 预分频器 参考 IC1PSC 描述
CC4S	9:8	R/W	0x0	CC4S: 捕捉 / 比较 4 选择 这 2 位定义通道的方向(输入/输出), 及输入脚的选择: 00: CC4 通道被配置为输出; 01: CC4 通道被配置为输入, IC3 映射在 TI4 上; 10: CC4 通道被配置为输入, IC3 映射在 TI3 上; 11: CC4 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 ADTIM_SMC 寄存器的 TS 位选择)。 注: CC4S 仅在通道关闭时(ADTIM_CCER 寄存器的 CC4E=0)才是可写的。
IC3F	7:4	R/W	0x0	IC3F: 输入捕捉 3 滤波器 参考 IC1F 描述
IC3PSC	3:2	R/W	0x0	IC3PSC: 输入捕捉 3 预分频器 参考 IC1PSC 描述
CC3S	1:0	R/W	0x0	CC3S: 捕捉 / 比较 3 选择 这 2 位定义通道的方向(输入/输出), 及输入脚的选择: 00: CC3 通道被配置为输出; 01: CC3 通道被配置为输入, IC3 映射在 TI3 上; 10: CC3 通道被配置为输入, IC3 映射在 TI4 上; 11: CC3 通道被配置为输入, IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 ADTIM_SMC 寄存器的 TS 位选择)。 注: CC3S 仅在通道关闭时(ADTIM_CCER 寄存器的 CC3E=0)才是可写的。

1. 4. 13 捕捉/比较使能寄存器 (ADTIM_CCER)

Address offset:		0x30													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	Reserved	CC4P	CC4E	CC3NP	CC3NE	CC3P	CC3E	CC2NP	CC2NE	CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E

Name	Bit	Type	Reset value	Description
Reserved	31:14	N/A	0x00	保留, 必须始终为复位值
CC4P	13	R/W	0x0	CC4P: 捕捉 / 比较 4 输出极性 参考 CC1P 描述
CC4E	12	R/W	0x0	CC4E: 捕捉 / 比较 4 输出使能 参考 CC1E 描述
CC3NP	11	R/W	0x0	CC3NP: 捕捉 / 比较 3 互补输出极性 参考 CC1NP 描述
CC3NE	10	R/W	0x0	CC3NE: 捕捉 / 比较 3 互补输出使能 参考 CC1NE 描述
CC3P	9	R/W	0x0	CC3P: 捕捉 / 比较 3 输出极性 参考 CC1P 描述

CC3E	8	R/W	0x0	CC3E: 捕捉 / 比较 3 输出使能 参考 CC1E 描述
CC2NP	7	R/W	0x0	CC2NP: 捕捉 / 比较 2 互补输出极性 参考 CC1NP 描述
CC2NE	6	R/W	0x0	CC2NE: 捕捉 / 比较 2 互补输出使能 参考 CC1NE 描述
CC2P	5	R/W	0x0	CC2P: 捕捉 / 比较 2 输出极性 参考 CC1P 描述
CC2E	4	R/W	0x0	CC2E: 捕捉 / 比较 2 输出使能 参考 CC1E 描述
CC1NP	3	R/W	0x0	CC1NP: 捕捉 / 比较 1 互补输出极性 CC1 通道配置为输出 0: OC1N 高电平有效; 1: OC1N 低电平有效。 CC1 通道配置为输入 本位用于和 CC1P 联合定义 TI1FP1 和 TI2FP1 的极性, 详见 CC1P 注: 在通道有互补输出时, 本位被预装载。如果寄存器 ADTIM_CR2 的 CCPC 被置 1, 则 CC1NP 的有效位从预装载复制的新值只有在通信事件发生时才起作用。 注: 一旦 LOCK 级别 (ADTIM_BDTR 寄存器中的 LOCK 位) 设为 3 或 2 且 CC1S=00 (通道配置为输出) 则该位不能被修改。
CC1NE	2	R/W	0x0	CC1NE: 捕捉 / 比较 1 互补输出使能 0: 关闭— OC1N 禁止输出, 因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 1: 开启— OC1N 信号输出到对应的输出引脚, 其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 注: 在通道有互补输出时, 本位被预装载。如果寄存器 ADTIM_CR2 的 CCPC 被置 1, 则 CC1NP 的有效位从预装载复制的新值只有在通信事件发生时才起作用。
CC1P	1	R/W	0x0	CC1P: 捕捉 / 比较 1 输出极性 CC1 通道配置为输出: 0: OC1 高电平有效; 1: OC1 低电平有效。 CC1 通道配置为输入: CC1NP/CC1P 位选择在触发或捕捉模式下 TI1FP1 和 TI2FP1 的有效极性。 00: 非反相/上升沿 电路作用于 TIxFP1 的上升沿 (在复位、外部时钟或触发模式下的捕捉或触发操作), TIxFP1 非反相 (在门控模式或编码模式)。 01: 反相/下降沿电路作用于 TIxFP1 的下降沿 (在复位、外部时钟或触发模式下的捕捉或触发操作), TIxFP1 反相 (在门控模式或编码模式)。 00: 保留不用 11: 非反相/上升或下降沿电路作用于 TIxFP1 的上升沿和下降沿 (在复位、外部时钟或触发模式下的捕捉或触发操作), TIxFP1 非反相 (在门控模式)。在编码模式下不能使用此配置。 注: 在通道有互补输出时, 本位被预装载。如果寄存器 ADTIM_CR2 的 CCPC 被置 1, 则 CC1NP 的有效位从预装载复制的新值只有在通信事件发生时才起作用。 注: 一旦 LOCK 级别 (ADTIM_BDTR 寄存器中的 LOCK 位) 设为 3 或 2, 则该位不能被修改。

CC1E	0	R/W	0x0	CC1E: 捕捉 / 比较 1 输出使能 CC1 通道配置为输出: 0: 关闭— OC1N 禁止输出, 因此 OC1N 的电平依赖于 MOE、OSS1、OSSR、OIS1、OIS1N 和 CC1E 位的值。 1: 开启— OC1N 信号输出到对应的输出引脚, 其输出电平依赖于 MOE、OSS1、OSSR、OIS1、OIS1N 和 CC1E 位的值。 CC1 通道配置为输入: 本位用于决定是否一个定时器值的捕捉要装载到捕捉/比较寄存器 1 (ADTIM_CCR1)。 0: 捕捉禁止。 1: 捕捉允许。 注: 在通道有互补输出时, 本位被预装载。如果寄存器 ADTIM_CR2 的 CCPC 被置 1, 则 CC1NP 的有效位从预装载复制的新值只有在通信事件发生时才起作用。

Control bits					Output States ⁽¹⁾	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出禁止(不由定时器驱动) OCx=0, OCx_EN=1	输出禁止 (不由定时器驱动) OCxN=0, OCxN_EN=1
		0	0	1	输出禁止 (不由定时器驱动) OCx=0, OCx_EN=1	OCxREF + 极性 OCxN=OCxREFxorCCxNP, OCxN_EN=0
		0	1	0	OCxREF + 极性 OCx=OCxREFxorCCxP, OCx_EN=0	输出禁止 (不由定时器驱动) OCxN=0, OCxN_EN=1
		0	1	1	OCREF + 极性 + 死区 OCx_EN=0	OCREF 互补输出 (not OCREF) + 极性 + 死区 OCxN_EN=0
		1	0	0	输出禁止 (不由定时器驱动) OCx=CCxP, OCx_EN=1	输出禁止 (不由定时器驱动) OCxN=CCxNP, OCxN_EN=1
		1	0	1	Off-State (输出使能但无效状态) OCx=CCxP, OCx_EN=0	OCxREF + 极性 OCxN=OCxREFxorCCxNP, OCxN_EN=0
		1	1	0	OCxREF + 极性 OCx=OCxREFxorCCxP, OCx_EN=0	Off-State (输出使能但无效状态) OCxN=CCxNP, OCxN_EN=0
		1	1	1	OCREF + 极性 + 死区 OCx_EN=0	OCREF 互补输出 (not OCREF) + 极性 + 死区 OCxN_EN=0
0	0	X	0	0	输出禁止 (不由定时器驱动) OCx=CCxP, OCx_EN=1	输出禁止 (不由定时器驱动) OCxN=CCxNP, OCxN_EN=1

Control bits					Output States ⁽¹⁾	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
	0		0	1	输出禁止 (不由定时器驱动)	
	0		1	0	异步: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1. 若时钟存在: 经过一个死区时间后 OCx=OISx, OCxN=OISxN, 假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平	
	0		1	1		
	1		0	0	输出禁止 (不由定时器驱动) OCx=CCxP, OCx_EN=1	输出禁止 (不由定时器驱动) OCxN=CCxNP, OCxN_EN=1
	1		0	1	Off-State (输出使能但无效状态) 异步: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0 若时钟存在: 经过一个死区时间后 OCx=OISx, OCxN=OISxN, 假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平.	
	1		1	0		
	1		1	1		

Table 11-1 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

1. 4. 14 计数寄存器(ADTIM_CNT)

		Address offset:		0x34											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留, 必须始终为复位值
CNT	15:0	R/W	0x0	CNT: 计数器值 (counter)

1. 4. 15 预分频寄存器 (ADTIM_PSC)

		Address offset:		0x38											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，必须始终为复位值
PSC	15:0	R/W	0x0	PSC: 预分频值 预分频器的值 (Prescaler value) 计数器的时钟频率 (CK_CNT) 等于 $f_{CK\_PSC} / (PSC[15:0] + 1)$ 。每次当更新事件产生时，PSC 的值被装入当前预分频器寄存器；更新事件包括计数器被 TIM_EGR 的 UG 位清 '0' 或被工作在复位模式的从控制器清 '0'。

1. 4. 16 自动重载寄存器 (ADTIM_ARR)

		Address offset:		0x3C											
		Reset value:		0x0000 FFFF											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，必须始终为复位值
ARR	15:0	R/W	0xFFFF	ARR: 自动重载值 ARR 包含了将要装载入实际的自动重载寄存器的值。当自动重载的值为空时，计数器不工作。

1. 4. 17 重复计数寄存器 (ADTIM_RCR)

		Address offset:		0x40											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								REP[7:0]							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留，必须始终为复位值

REP	REP: 重 复 计 数 器 值													
	预装载寄存器被使能后，这些位允许用户设置比较寄存器的更新速率（即周期性地从预装载寄存器传输到当前寄存器）；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器 REP_CNT 达到 0，会产生一个更新事件并且计数器 REP_CNT 重新从 REP 值开始计数。由于 REP_CNT 只有在周期更新事件 U_RC 发生时才重载 REP 值，因此对 ADTIM_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(REP+1) 对应着： — 在边沿对齐模式下，PWM 周期的数目； — 在中心对称模式下，PWM 半周期的数目；													
	7:0	R/W	0x0											

1. 4. 18 捕捉/比较寄存器 1 (ADTIM_CCR1)

		Address offset:			0x44										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，必须始终为复位值
CCR1	15:0	R/W	0x0	CCR1: 捕 捉 / 比 较 通 道 1 的 值 若 CCR1 通道配置为输出： CCR1 决定了装入当前捕获/比较 1 寄存器的值（预装载值）。 如果在 ADTIM_CCMR1 寄存器 (OC1PE 位) 中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。当前捕获/比较寄存器参与同计数器 ADTIM_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CCR1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件 (IC1) 传输的计数器值。

1. 4. 19 捕捉/比较寄存器 2 (ADTIM_CCR2)

		Address offset:			0x48										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，必须始终为复位值

CCR2	15:0	R/W	0x0	CCR2: 捕捉 / 比较通道 2 的值 参考 CCR1 描述

1. 4. 20 捕捉/比较寄存器 3 (ADTIM_CCR3)

		Address offset:			0x4C										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7		5	4	3	2	1	0
CCR3[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留, 必须始终为复位值
CCR3	15:0	R/W	0x0	CCR3: 捕捉 / 比较通道 3 的值 参考 CCR1 描述

1. 4. 21 捕捉/比较寄存器 4 (ADTIM_CCR4)

		Address offset:			0x50										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留, 必须始终为复位值
CCR4	15:0	R/W	0x0	CCR4: 捕捉 / 比较通道 4 的值 参考 CCR1 描述

1. 4. 22 刹车和死区寄存器 (ADTIM_BDTR)

		Address offset:			0x54										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]		DTG[7:0]							

Name	Bit	Type	Reset value	Description

Reserved	31:16	N/A	0x00	保留，必须始终为复位值
MOE	15	R/W	0x0	MOE: 主输出使能 一旦刹车输入有效，该位被硬件异步清'0'。根据 AOE 位的设置值，该位可以由软件清'0'或被自动置1。它仅对配置为输出的通道有效。 0: 禁止 OC 和 OCN 输出或强制为空闲状态 1: 如果设置了相应的使能位(ADTIM_CCER 寄存器的 CCxE、CCxNE 位)，则开启 OC 和 OCN 输出
AOE	14	R/W	0x0	AOE: 自动输出使能 0: MOE 只能被软件置'1' 1: MOE 能被软件置'1'或在下一个更新事件被自动置'1' (如果刹车输入无效) 注: 一旦 LOCK 级别(ADTIM_BDTR 寄存器中的 LOCK 位)设为'1'，则该位不能被修改。
BKP	13	R/W	0x0	BKP: 刹车输入极性 0: 刹车输入低电平有效 1: 刹车输入高电平有效 注: 一旦 LOCK 级别(ADTIM_BDTR 寄存器中的 LOCK 位)设为'1'，则该位不能被修改。 注: 任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用
BKE	12	R/W	0x0	BKE: 刹车使能 0: 刹车输入禁止 (BRK 和 CCS 时钟失效事件) 1: 刹车输入允许 (BRK 和 CCS 时钟失效事件) 注: 一旦 LOCK 级别(ADTIM_BDTR 寄存器中的 LOCK 位)设为'1'，则该位不能被修改。 注: 任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。
OSSR	11	R/W	0x0	OSSR: 运行模式下“关闭状态”选择 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 0: 当定时器不工作时，禁止 OC/OCN 输出 (OC/OCN 使能输出信号=0)。 1: 当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 使能并输出无效电平，然后置 OC/OCN 使能输出信号=1 注: 一旦 LOCK 级别(ADTIM_BDTR 寄存器中的 LOCK 位)设为'2'，则该位不能被修改。
OSSI	10	R/W	0x0	OSSI: 运行模式下“空闲状态”选择 该位用于当 MOE=0 时通道为输出。 0: 当定时器不工作时，禁止 OC/OCN 输出 (OC/OCN 使能输出信号=0)。 1: 当定时器不工作时，一旦 CCxE=1 或 CCxNE=1，OC/OCN 被强制输出空闲电平，置 OC/OCN 使能输出信号=1 注: 一旦 LOCK 级别(ADTIM_BDTR 寄存器中的 LOCK 位)设为'2'，则该位不能被修改。
LOCK[1:0]	9:8	R/W	0x0	LOCK[1:0]: 锁定设置 该位为防止软件错误而提供写保护。 00: 锁定关闭，寄存器无写保护。 01: 锁定级别 1，不能写入 ADTIM_BDTR 寄存器的 DTG、BKE、BKP、AOE 位和 ADTIM_CR2 寄存器的 OISx/OISxN 位 10: 锁定级别 2，不能写入锁定级别 1 中的各位，也不能写入 CC 极性位 (一旦相关通道通过 CCxS 位设为输出，CC 极性位是 ADTIM_CCER 寄存器的 CCxP/CCNxP 位) 以及 OSSR/OSSI 位。 11: 锁定级别 3，不能写入锁定级别 2 中的各位，也不能写入 CC 控制位 (一旦相关通道通过 CCxS 位设为输出，CC 控制位是 ADTIM_CCMRx 寄存器的 OCxM/OCxPE 位)。 注: 在系统复位后，只能写一次 LOCK 位，一旦写入 ADTIM_BDTR 寄存器，则其内容冻结直至复位

DTG[7:0]	7:0 R/W 0x0	DTG[7:0]: 死区发生器设置 这些位定义了插入互补输出之间的死区持续时间。假设 DT 表示其持续时间。 DTG[7:5]=0xx => DT=DTG[7:0]x tdtg with tdtg=tdts. DTG[7:5]=10x => DT=(64+DTG[5:0])xtdtg with Tdtg=2xtDTS. DTG[7:5]=110 => DT=(32+DTG[4:0])xtdtg with Tdtg=8xtDTS. DTG[7:5]=111 => DT=(32+DTG[4:0])xtdtg with Tdtg=16xtDTS. 例: 若 TDTS=125ns(8MHZ), 可能的死区时间为: 0 到 15875ns, 若步长时间为 125ns; 16us 到 31750ns, 若步长时间为 250ns; 32us 到 63us, 若步长时间为 1us; 64us 到 126us, 若步长时间为 2us; 注: 一旦 LOCK 级别(ADTIM_BDTR 寄存器中的 LOCK 位) 设为 1、2 或 3, 则不能修改这些位。

1. 4. 23 DMA 事件使能寄存器 (ADTIM_DMAEN)

		Address offset:			0x58										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TDE	COM DE	CC4D E	CC3D E	CC2D E	CC1D E	UDE

Name	Bit	Type	Reset value	Description
Reserved	31:7	N/A	0x00	保留, 必须始终为复位值
TDE	6	R/W	0x0	TDE: 触发中断 DMA 请求使能 0: 触发中断 DMA 请求禁止. 1: 触发中断 DMA 请求使能.
COMDE	5	R/W	0x0	COMDE: COM 中断 DMA 请求使能 0: COM 中断 DMA 请求禁止. 1: COM 中断 DMA 请求使能.
CC4DE	4	R/W	0x0	CC4DE: 捕捉 / 比较 4 中断 DMA 请求使能 0: CC4 中断 DMA 请求禁止. 1: CC4 中断 DMA 请求使能.
CC3DE	3	R/W	0x0	CC3DE: 捕捉 / 比较 3 中断 DMA 请求使能 0: CC3 中断 DMA 请求禁止. 1: CC3 中断 DMA 请求使能.
CC2DE	2	R/W	0x0	CC2DE: 捕捉 / 比较 2 中断 DMA 请求使能 0: CC2 中断 DMA 请求禁止. 1: CC2 中断 DMA 请求使能.
CC1DE	1	R/W	0x0	CC1DE: 捕捉 / 比较 1 中断 DMA 请求使能 0: CC1 中断 DMA 请求禁止. 1: CC1 中断 DMA 请求使能.
UDE	0	R/W	0x0	UDE: 更新中断 DMA 请求使能 0: 更新中断 DMA 请求禁止. 1: 更新中断 DMA 请求使能.

第12章 通用 32/16 位定时器（GPTIMA/B）

1.1 GPTIMA/B 简介

通用定时器(GPTIMA/B)由一可编程预除器驱动之 32/16 位自动重载计数器构成。

其亦可用于多种用途，包含测量输入信号脉冲长度(输入捕获)或产生输出波形(输出比较,PWM)

可使用定时预除器与 APB 时钟控制预除器来调校脉冲长度与波形周期，由数微秒至数毫秒。

高级控制(ADTIM1),通用(GPTIMA1,GPTIMB1,GPTIMC1)与基本(BSTIM1)定时器皆为完全独立不共享任何时钟源

1.2 GPTIMA/B 主要特性

- ◆ 32/16 位递增、递减、递增/递减自动重载计数器。
- ◆ 16 位可编程预分频器，用于对计数器时钟频率进行分频（即运行时修改），分频系数介于 1 到 65536 之间。
- ◆ 多达 4 个独立通道，可用于：
 - 输入捕获
 - 输出比较
 - PWM 生成（边沿和中心对齐模式）
 - 单脉冲模式输出
- ◆ 同步化电路可控制定时器含外部信号与链接数个定时器。
- ◆ 发生如下事件时生成中断/DMA 请求：
 - 更新：计数器上溢/下溢、计数器初始化（通过软件或内部/外部触发）
 - 触发事件（计数器启动、停止、初始化或通过内部/外部触发计数）
 - 输入捕获(捕获寄存器)
 - 输出比较(计数寄存器配对比较寄存器)
- ◆ 支持定位用增量（正交）编码器和霍尔传感器电路。
- ◆ 外部时钟触发输入或 cycle-by-cycle 电流管理

预除器可将计数器时钟频率依照 1 至 65536 间之任意因子分频。基于一 16 位寄存器控制之 16 位计数器(GPTIMA/B_PSC 寄存器内)。因其具有缓冲功能，可于运行时被变更。新的预除器参数于下一次更新事件到达前被采用。预除计数器仅能向下计数。预除器被设定后，一旦预除计数器到达 0，计数器则会向上或向下计数，此外计数器会随着 APB 时钟一同更新。图 2 与图 3 标示当预除器于运作时变更计数器之行为范例。.

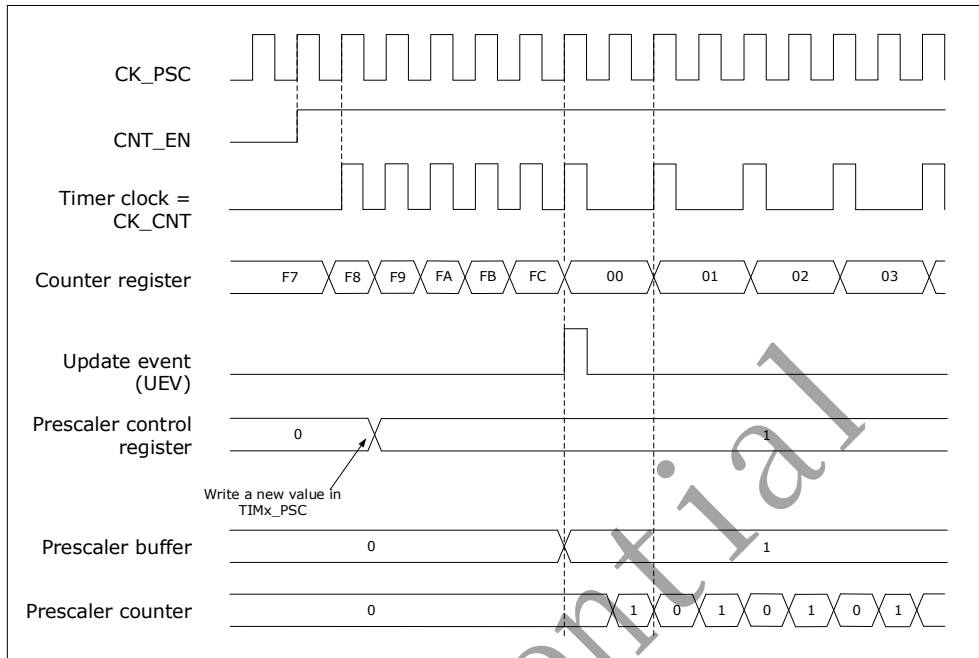


图 12-2 预除器区隔由 1 变更为 2 之计数器时序图

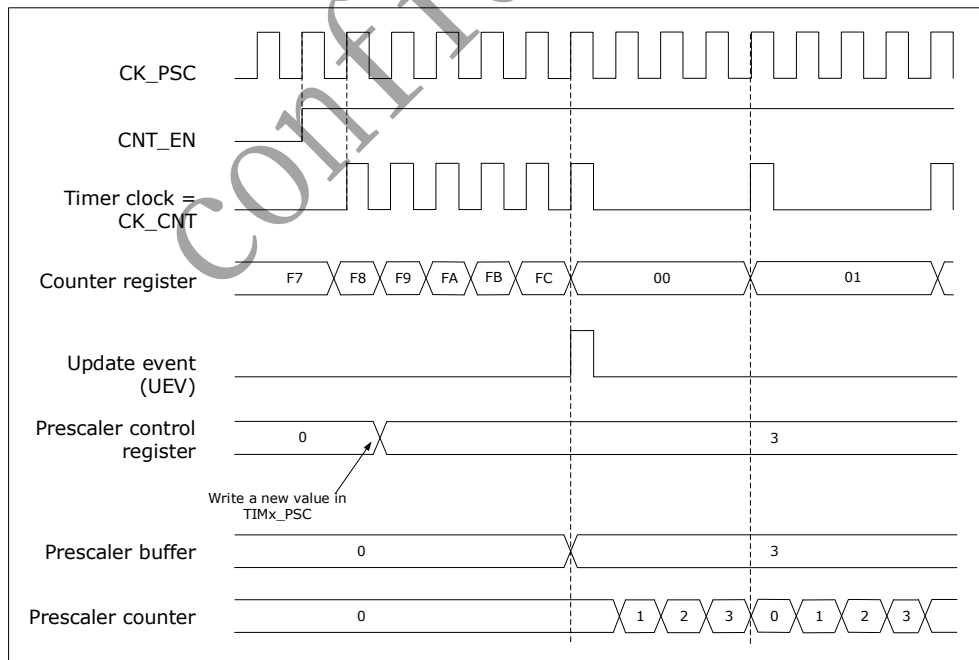


图 12-3 预除器区隔由 1 变更为 4 之计数器时序图

1.3.2 计数器模式

递增计数模式

在递增计数模式，计数器由 0 计算至自动加载值(GPTIMA/B_ARR 寄存器内容)由 0 重新启动并产生一计数器上溢事件。

若重复计数器被使用，更新事件(UEV)会在向上计数依重复计数器(GPTIMA/B_RCR)编程次数重复后产生。此外更新事件会于每一计数器上溢产生。

更新事件可于每一计数器下溢或设置 GPTIMA/B_EGR 寄存器 UG 位产生(透过软件或从模式控制器)。

UEV 事件可由软件透过设置 GPTIMA/B_CR1 寄存器之 UDIS 位以关闭。这是为了避免更新写入新值至预载寄存器时，影子寄存器也一并更新。接着直到 UDIS 位被写 0 皆无更新事件产生。但计数器会由 0 重启，预除器之计数器亦同(但除频率不会变更)。

当更新事件发生时，所有寄存器皆被更新且更新标志(GPTIMA/B_RIF 寄存器 UIF 位)皆被设定：

- ◆ 自动重载影子寄存器被更新为预载值(GPTIMA/B_ARR)
- ◆ 预除器之缓冲器随着预载值重新加载(GPTIMA/B_PSC 寄存器之内容)

下列图表为计数器在 GPTIMA/B_ARR=0x36 时对应不同时钟频点之行为范例：

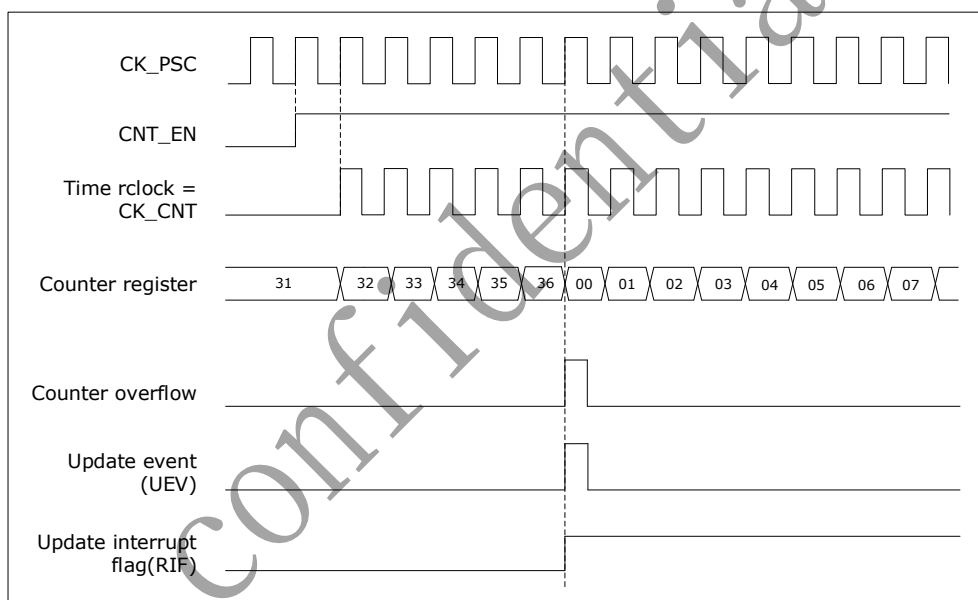


图 12-4 计数器时序图，内部时钟除以 1

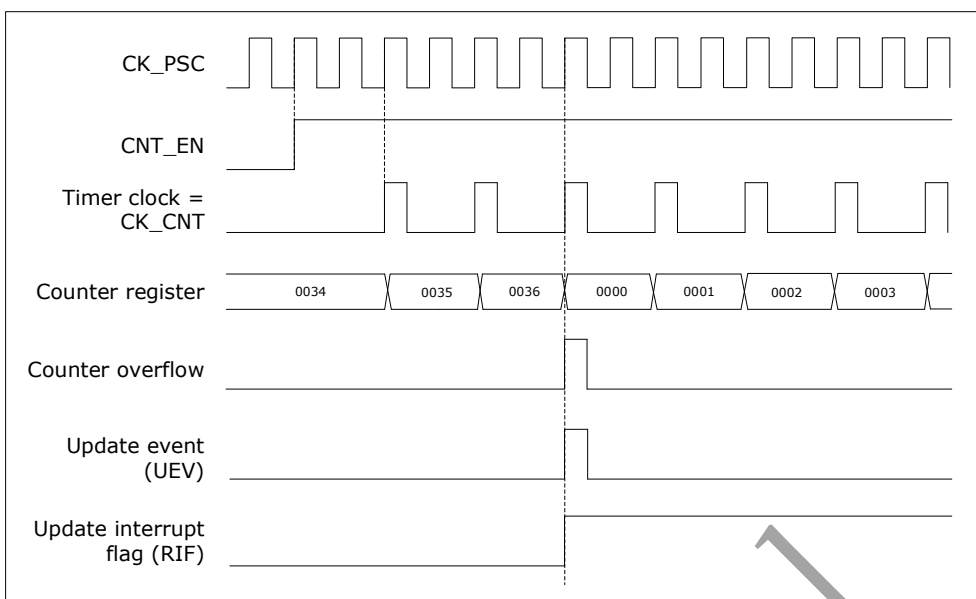


图 12-5 计数器时序图，内部时钟除以 2

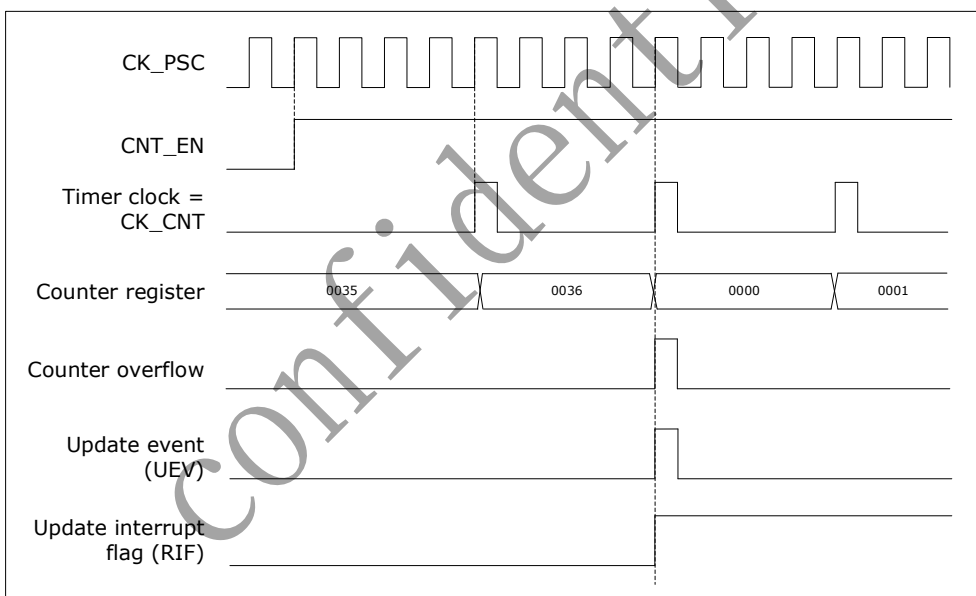


图 12-6 计数器时序图，内部时钟除以 4

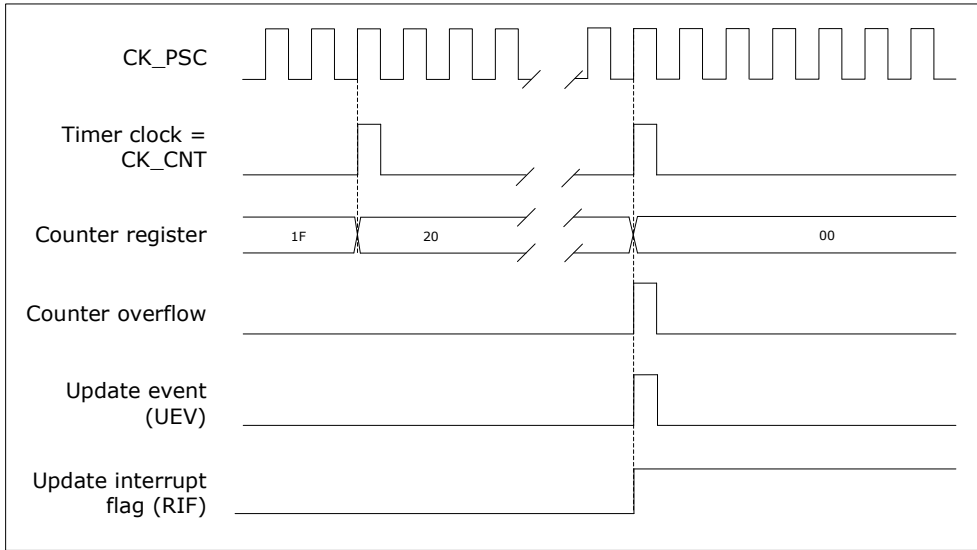


图 12-7 计数器时序图，内部时钟除以 N

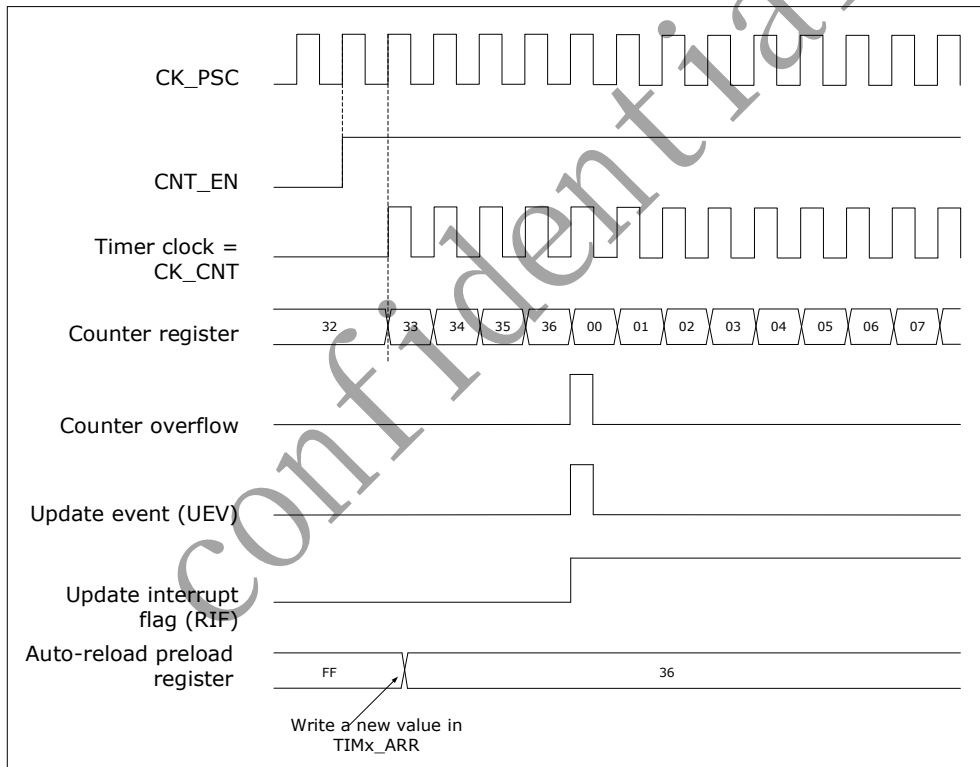


图 12-8 计数器时序图，当 ARPE=0 更新事件 (GPTIMA/B_ARR 无预载)

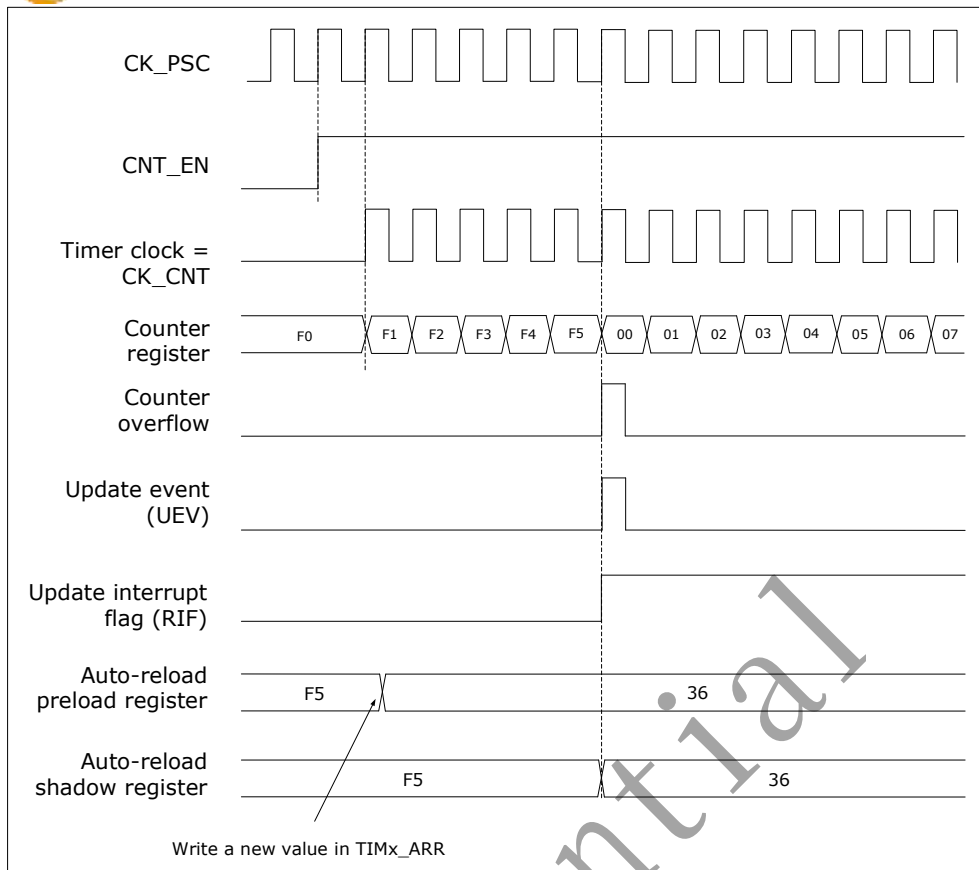


图 12-9 计数器时序图，当 ARPE=1 更新事件 (GPTIMA/B_ARR 预载)

递减计数模式

在递减计数模式，计数器由自动加载值(GPTIMA/B_ARR 寄存器内容)计算至 0，由自动加载值重新启动后产生计数下溢事件。

若重复计数器被使用，更新事件(UEV)会在向下计数依重复计数器(GPTIMA/B_RCR)编程次数重复后产生。此外更新事件会于每一计数器上溢产生。

设定 GPTIMA/B_EGR 寄存器之 UG 位(透过软件或从模式控制器)亦可产生更新事件。

UEV 事件可由软件透过设置 GPTIMA/B_CR1 寄存器之 UDIS 位以关闭。这是为了避免更新写入新值至预载寄存器时，影子寄存器也一并更新。接着直到 UDIS 位被写 0 皆无更新事件产生。但计数器会由 0 重启，预除器之计数器亦同(但除频率不会变更)。

当更新事件发生时，所有寄存器被更新且跟更新状态(GPTIMA/B_RIF 寄存器之 UIF 位)也被设置

- ◆ 寄存器重复计数器重载 GPTIMA/B_RCR 寄存器内容
- ◆ 预除器之缓冲重载预除值(GPTIMA/B_PSC 寄存器内容)
- ◆ 自动重载行为寄存器依预载值更新(GPTIMA/B_ARR 寄存器内容)。注:自动重载会在计数器重载前更新，因此下一周期将会是预期值

下列图表为计数器在 GPTIMA/B_ARR=0x36 时对应不同时钟频点之行为范例

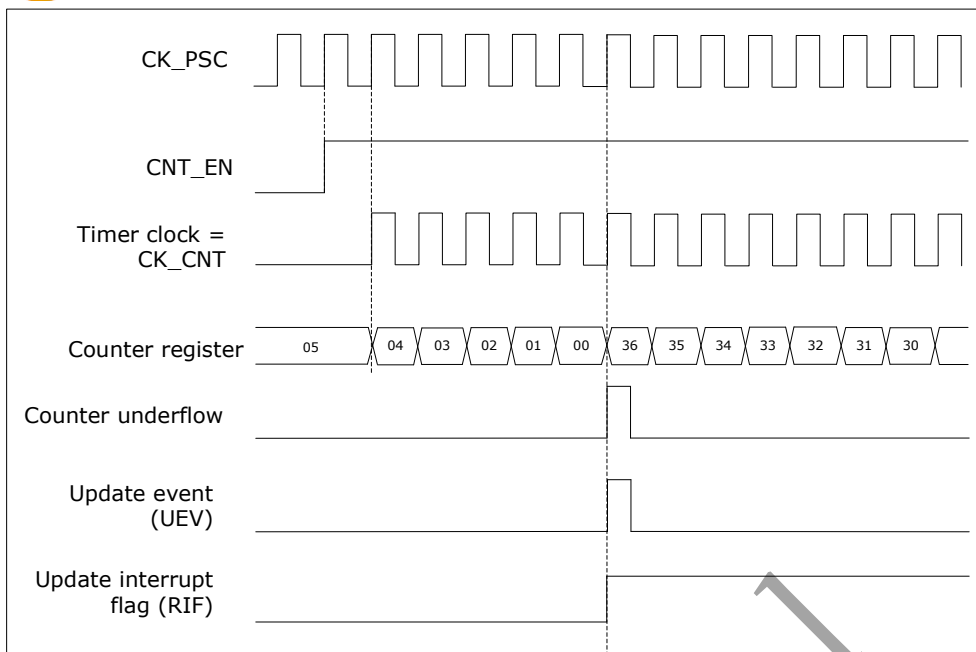


图 12-10 计数器时序图，内部时钟分频因子为 1

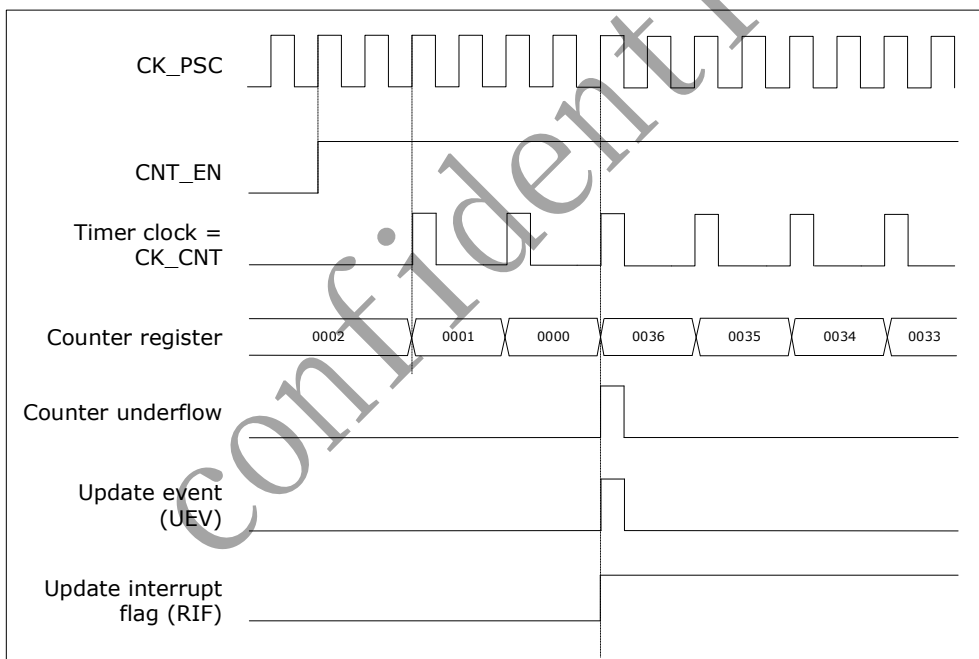


图 12-11 计数器时序图，内部时钟分频因子为 2

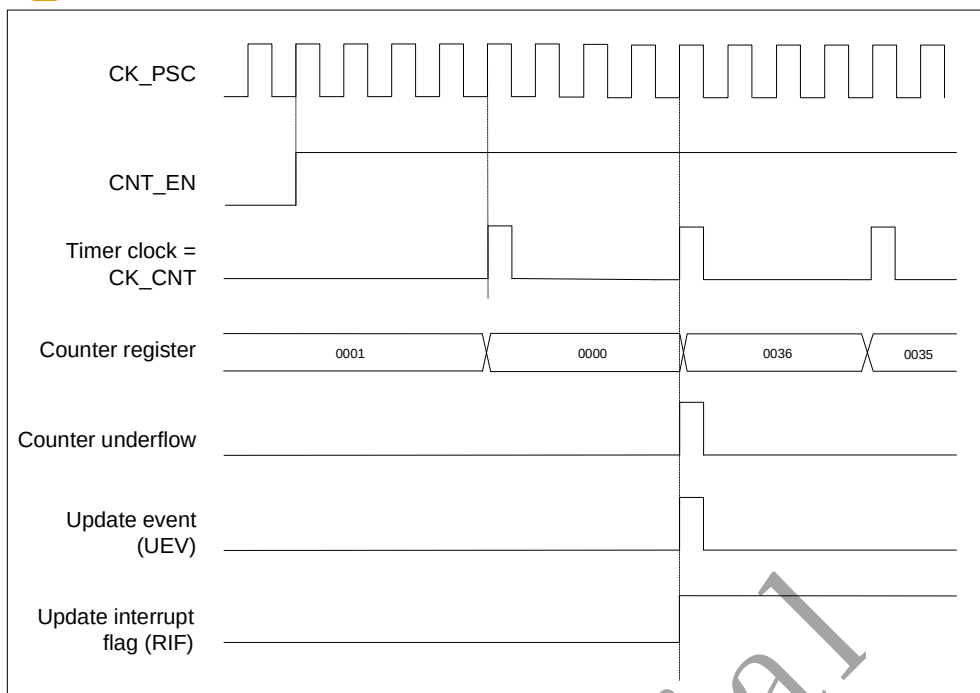


图 12-12 计数器时序图，内部时钟分频因子为 4

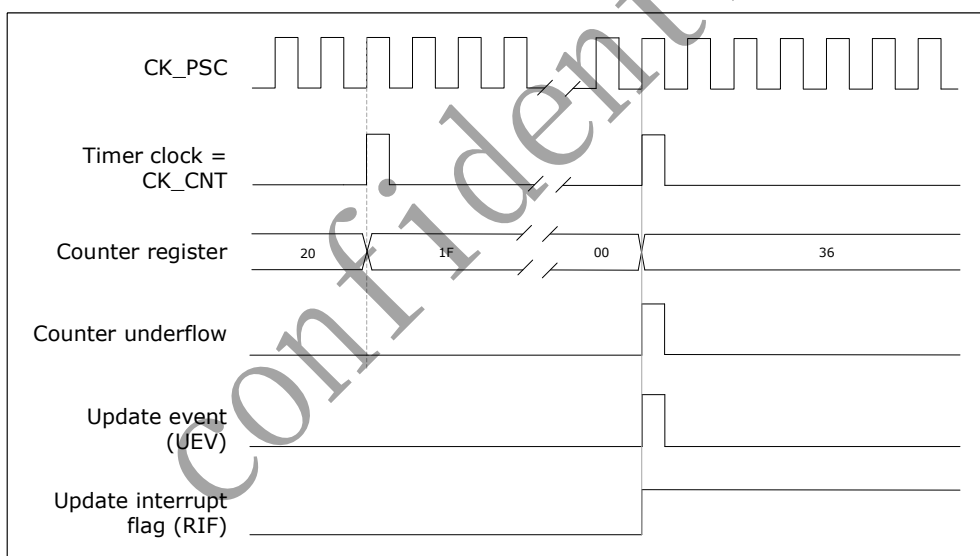


图 12-13 计数器时序图，内部时钟分频因子 N

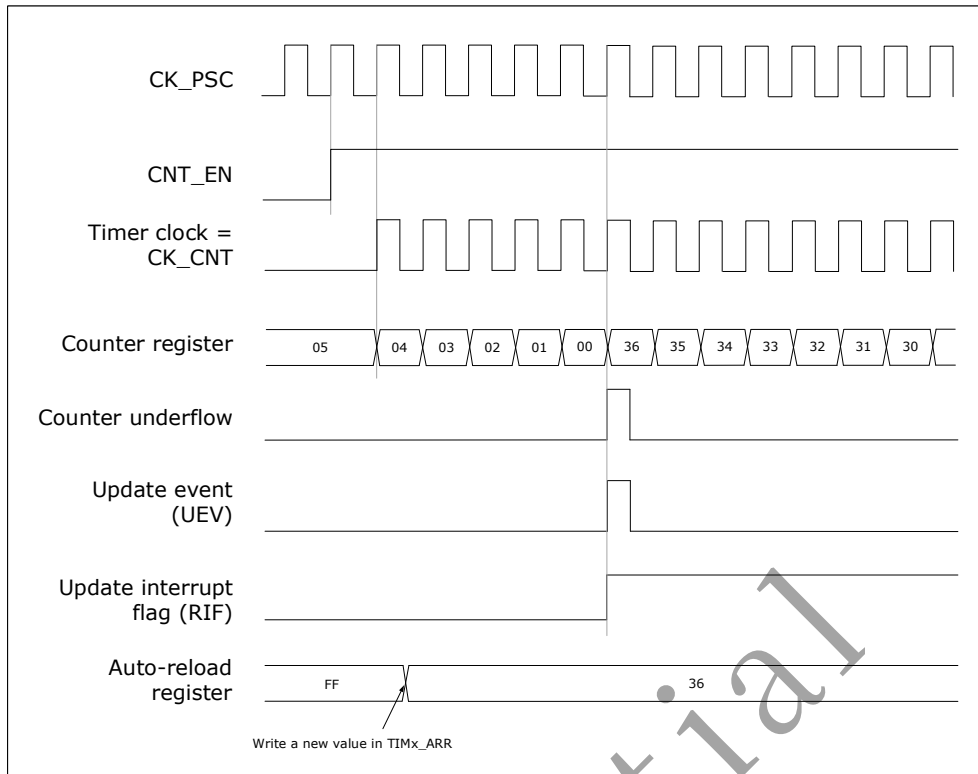


图 12-14 计数器时序图,重复计数器未使用时之更新事件

中心对齐模式(增减计数)

在中心对齐模式时，计数器由 0 开始计算至自动重载值(GPTIMA/B_ARR 寄存器之内容)-1，产生计数器上溢事件，接着由计数自动重载值计算至 1，并产生计数器下溢事件。然后重新由 0 开始计算。

GPTIMA/B_CR1 寄存器之 CMS 位不等于'00'时，中心对齐模式方可作用。各信道之输出比较中断标志输出，当:计数器递减计算(中心对齐模式 1,CMS="01")，计数器递增计算(中心对齐模式 2,CMS="10")，计数器递增/递减计算(中心对齐模式 3,CMS="11")

于此模式下，GPTIMA/B_CR1 寄存器之 DIR 方向位将无法被写。将由硬件对其进行更新并给予计数器之电流方向。

更新事件可藉由每一计数器之上溢产生，且每一计数器下溢或透过设置 GPTIMA/B_EGR 寄存器之 UG 位(透过设置软件或使用属模式控制器)亦可产生一更新事件。如此计数器将重新由 0 计数，预除器之计数器亦同。

UEV 更新事件可透过设置软件或 GPTIMA/B_CR1 寄存器之 UDIS 位关闭。这是为了避免写入新值至预载寄存器时影子寄存器也一并更新。直到 UDIS 位被写入 0 前都不会有更新事件发生。但计数器仍由现有自动加载值向上或向下计数。这是为了避免当发生捕获事件并清除计数器的同时产生更新与捕获中断。

当更新事件发生时，所有寄存器皆更新且状态(GPTIMA/B_RIF 寄存器之 UIF 位)更新为:

- ◆ 重复计数器重载 GPTIMA/B_RCR 寄存器内容
- ◆ 预除缓冲器重载预载值(GPTIMA/B_PSC 寄存器内容)

- ◆ 自动重载作用寄存器藉由预载值更新(GPTIMA/B_ARR 寄存器内容)
- ◆ 注:若更新源为计数器上溢, 自动重载会在计数器重载前更新, 因此下一周期为预期值(计数器载入新值)

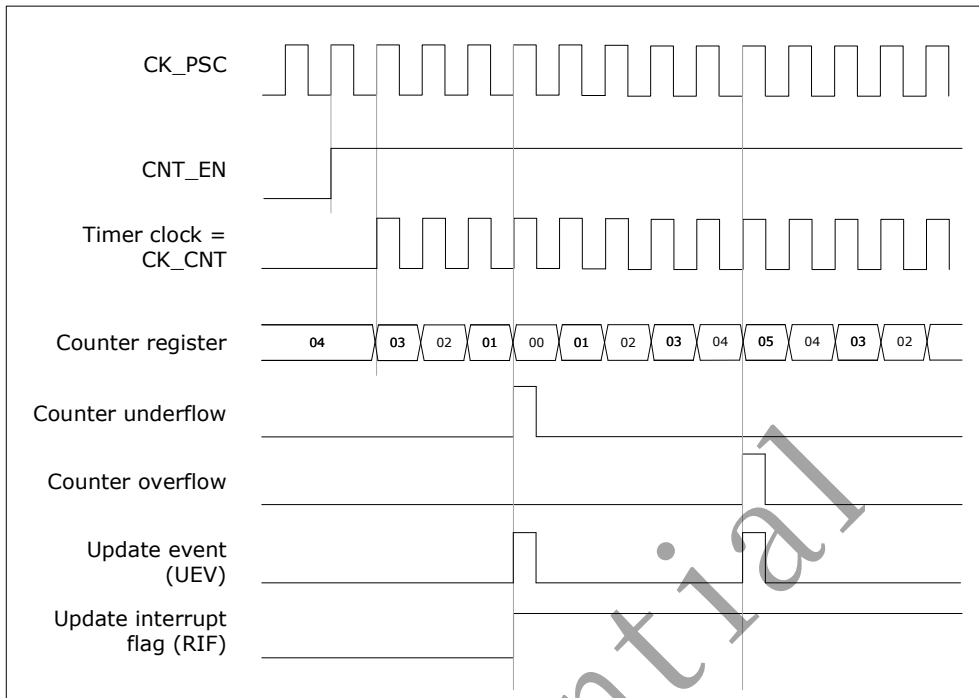


图 12-15 计数器时序图, 内部时钟分频因子为 1,GPTIMA/B_ARR=0x5

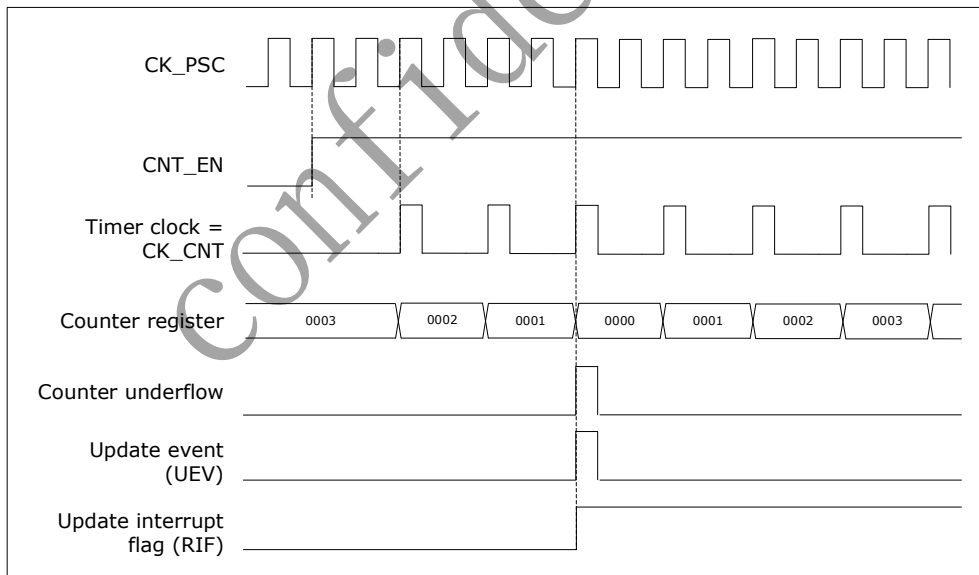


图 12-16 计数器时序图, 内部时钟分频因子为 2

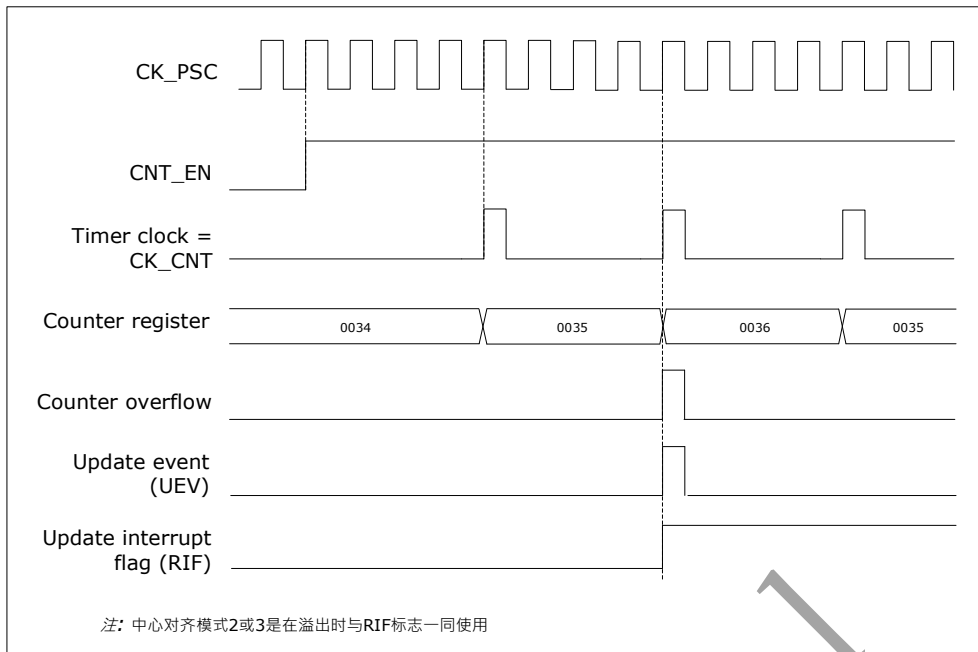


图 12-17 计数器时序图，内部时钟分频因子为 4,GPTIMA/B_ARR=0x36

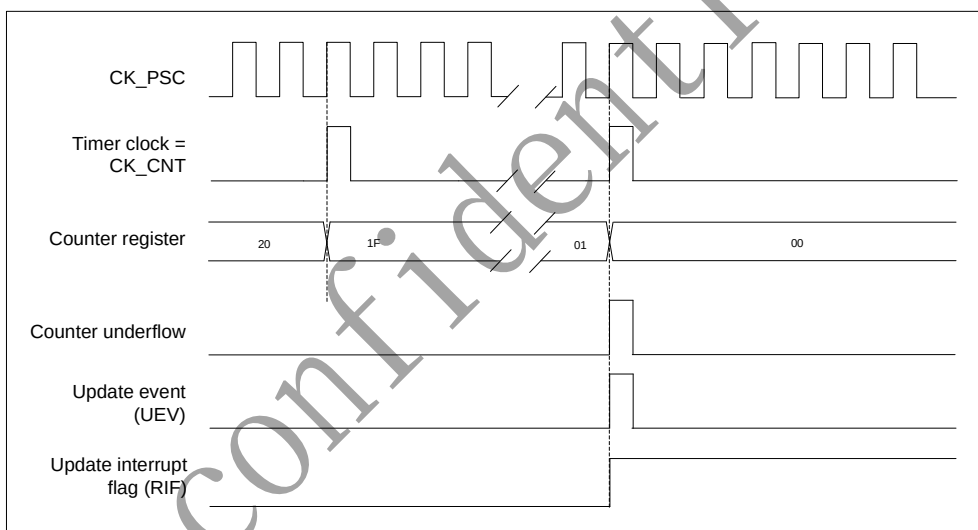


图 12-18 计数器时序图，内部时钟分频因子为 N

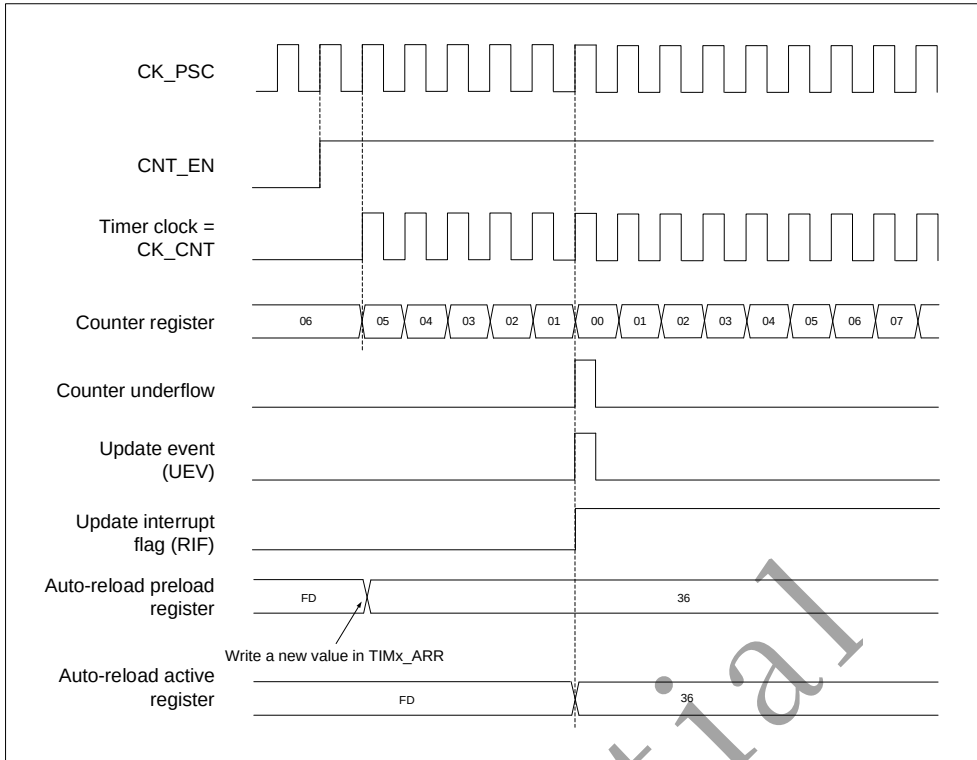


图 12-19 计数器时序图，ARPE=1 时之更新事件(计数器下溢)

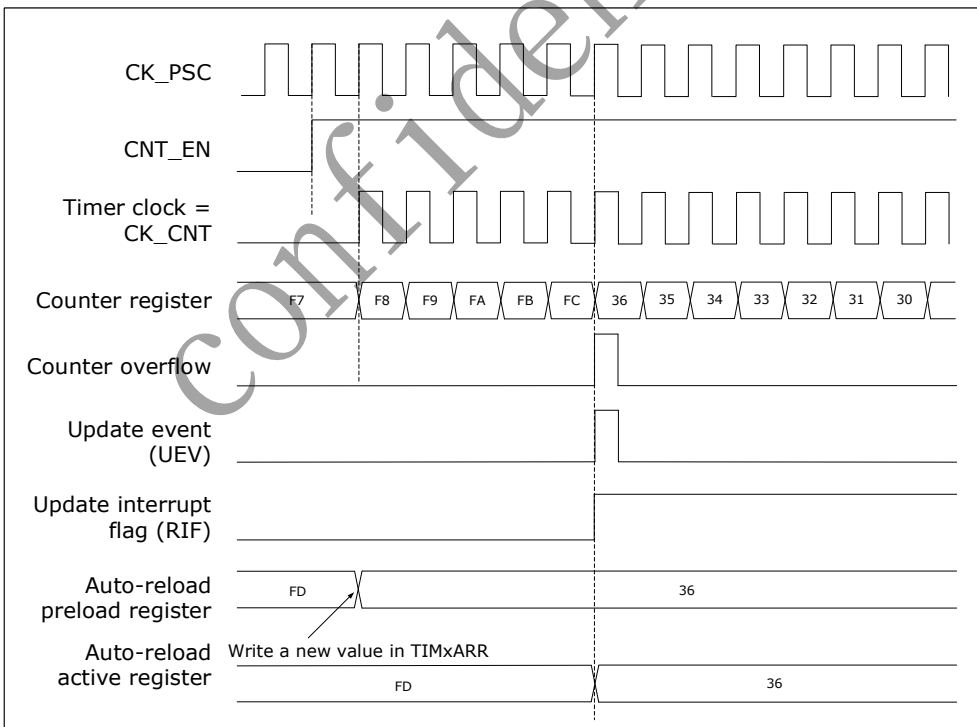


图 12-20 计数器时序图，ARPE=1 时之更新事件(计数器上溢)

1.3.3 时钟源

计数器时钟可由下列时钟源提供:

- ◆ 内部时钟(CK_INT)

- ◆ 外部时钟模式 1:外部输入引脚
- ◆ 外部时钟模式 2:外部触发输入 ETR
- ◆ 内部触发输入(ITRx):使用一定时器作为另一个定时器之预除器, 例:可定时器 1 配置为定时器 2 的预除器。

1. 3. 3. 1 内部时钟源(CK_INT)

若属模式控制器被关闭(GPTIMA/B_SMCR 寄存器内, SMS=000), CEN,DIR 位(GPTIMA/B_CR1 寄存器)与 UG 位(GPTIMA/B_EGR 寄存器)皆为实际控制位且仅可被软件变更(UG 位仍被自动清除)只要 CEN 位被写为'1', 预除器之时钟即由内部时钟 CK_INT 提供。

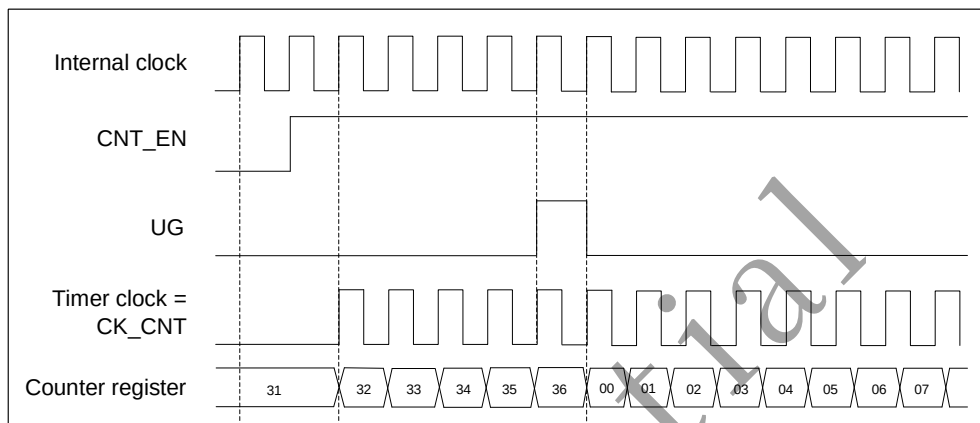


图 12-21 一般模式下之控制电路, 内部时钟分频因子为 1

1. 3. 3. 2 外部时钟源模式 1

当 GPTIMA/B_SMCR 寄存器之 SMS=111 时此模式会被选定。计数器于被选定输入端之每一上升或下降缘计数。

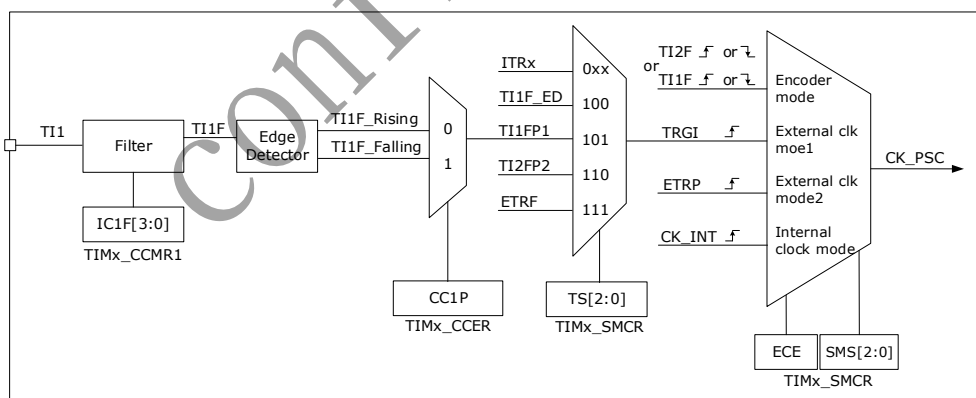


图 12-22 TI1 外部时钟连接范例

例:要配置向上计数器在 T1 输入端的上升沿计数, 如以下步骤:

- ◆ 配置 GPTIMA/B_CCMR 寄存器 CC1S=01, 配置信道 1 以检测 TI1 输入的上升沿。
- ◆ 写 GPTIMA/B_CCMR 寄存器之 IC1F[3:0]位以配置输入滤波器期间(若没有滤波器需求, 维持 IC1F=0000)
- ◆ 写 CC1P=0 至寄存器 GPTIMA/B_CCER 以选择上升缘极性

- ◆ 透过写 GPTIMA/B_SMCR 寄存器的 SMS=111，以配置定时器外部时钟模式 1
- ◆ 写 GPTIMA/B_SMCR 寄存器中的 TS=101，以选定 TI1 作为触发输入源
- ◆ 写 GPTIMA/B_CR1 寄存器之 CEN=1 以使能计数器

当上升缘发生于 TI1，计数器计数一次且 TIF 标志被设置

TI1 上升缘与实际时钟间之延时，取决于 TI1 输入之再同步电路

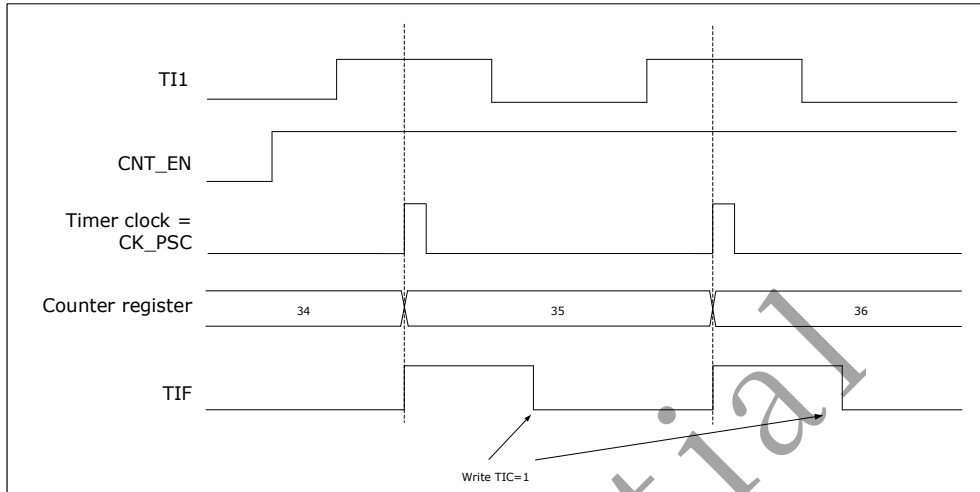


图 12-23 外部时钟模式 1 之控制电路

1. 3. 3. 3 外部时钟源模式 2

此模式透过写 GPTIMA/B_SMCR 寄存器之 ECE=1 被选定

计数器可于每一外部触发输入 ETR 上升或下降缘计数，并可搭配预除频器。

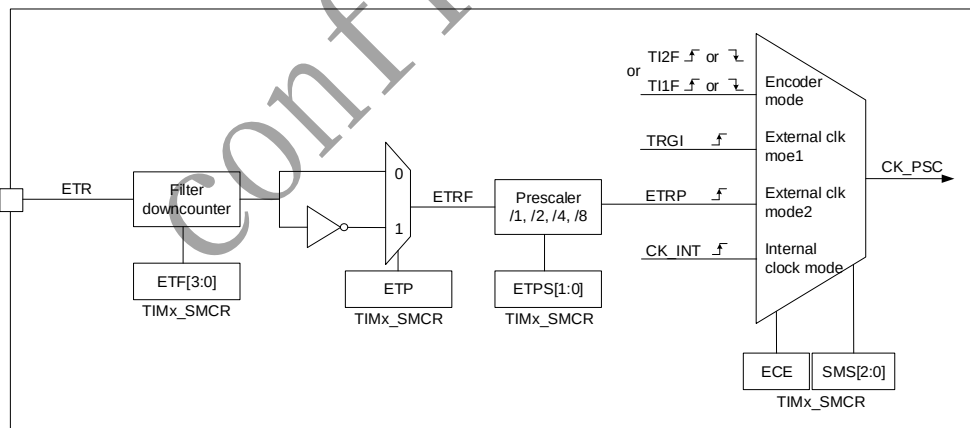


图 12-24 ETR 给予一外部触发输入区块概述

例:欲设置上数计数器于每两个 ETR 上升缘计数，则使用下列方式:

- ◆ 此例中无须使用滤波器，写 GPTIMA/B_SMCR 寄存器之 ETF[3:0]=0000
- ◆ 写 GPTIMA/B_SMCR 寄存器之 ETPS[1:0]=01 以设置预除器
- ◆ 写 GPTIMA/B_SMCR 寄存器之 ETP=0 以选择 ETR 引脚之上升缘侦测
- ◆ 写 GPTIMA/B_SMCR 寄存器之 ECE=1 以使能外部时钟模式 2

◆ 写 GPTIMA/B_CR1 寄存器之 CEN=1 以使能计数器

计数器于每两个上升缘计数

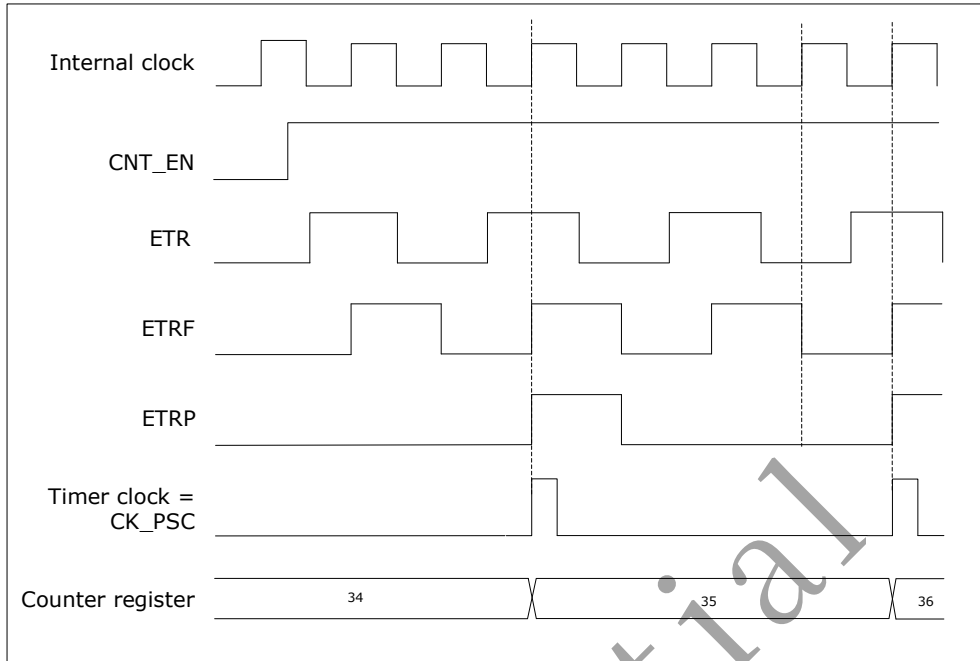


图 12-25 外部时钟模式 2 之控制电路

1. 3. 3. 4 内部触发输入(ITRx)

当 GPTIMA/B_SMCR 寄存器之 SMS=111 时此模式会被选定。计数器于被选定输入端之每一上升或下降缘计数。

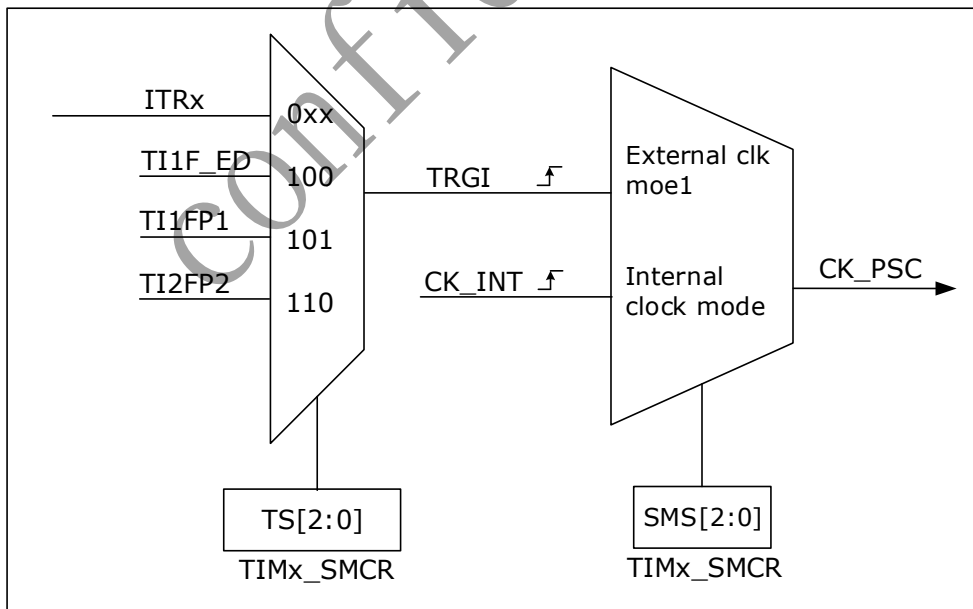


图 12-26 ITRx 外部时钟连接范例

例:要配置向上计数器在 ITRx 输入端的上升沿计数，如以下步骤：

◆ 透过写 GPTIMA/B_SMCR 寄存器的 SMS=111，以配置定时器外部时钟模式 1

- ◆ 写 GPTIMA/B_SMCR 寄存器中的 TS=0xxx，以选定 ITRx 作为触发输入源
- ◆ 写 GPTIMA/B_CR1 寄存器之 CEN=1 以使能计数器

当上升缘发生于 ITRx，计数器计数一次

ITRx 上升缘与实际时钟间之延时，取决于 ITRx 输入之再同步电路

1.3.4 捕获/比较信道

每一捕获/比较信道，皆是围绕着一捕获/比较寄存器(包含一影子寄存器)，包含一捕获输入部分(含数字滤波器，多路复用与预除器)以及一输出阶段(含比较器与输出控制)。

下图 28 为单一捕获/比较信道之概述

输入阶段范例对应之 Tix 输入以产生一滤波后之 TIxF 信号，接着一带极性边缘侦测器产生一信号(TIxFPx)，可作为属模式控制器之触发输入或是做为捕获控制。该信号于捕获寄存器(ICxPS)之前即预先分频。

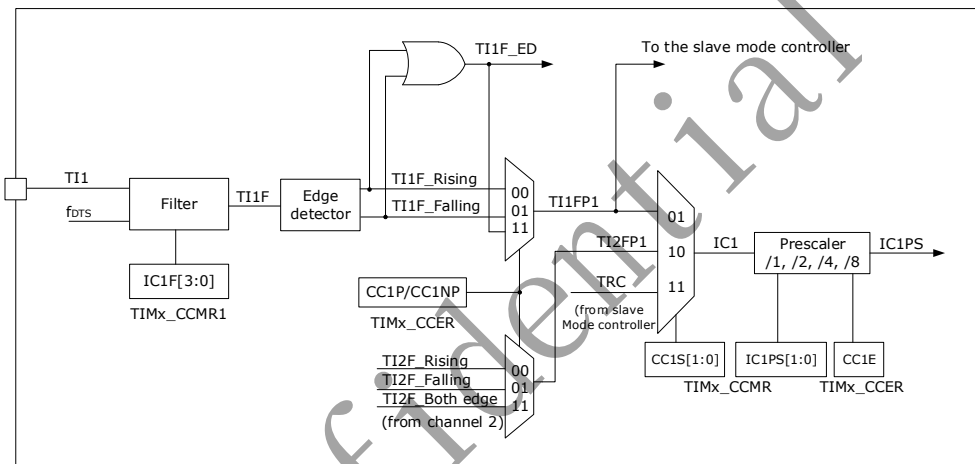


图 12-27 捕获/比较信道(例:信道 1 输入阶段)

输出部分产生一个中间波形 OCxRef(高有效)作为基准，链结的末端决定最终输出信号之极性。

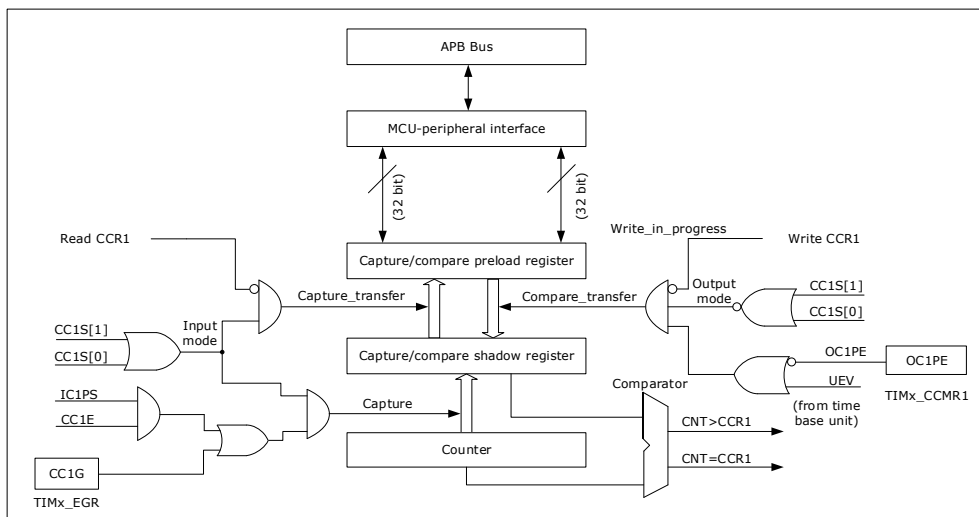


图 12-28 捕捉/比较信道 1 主电路

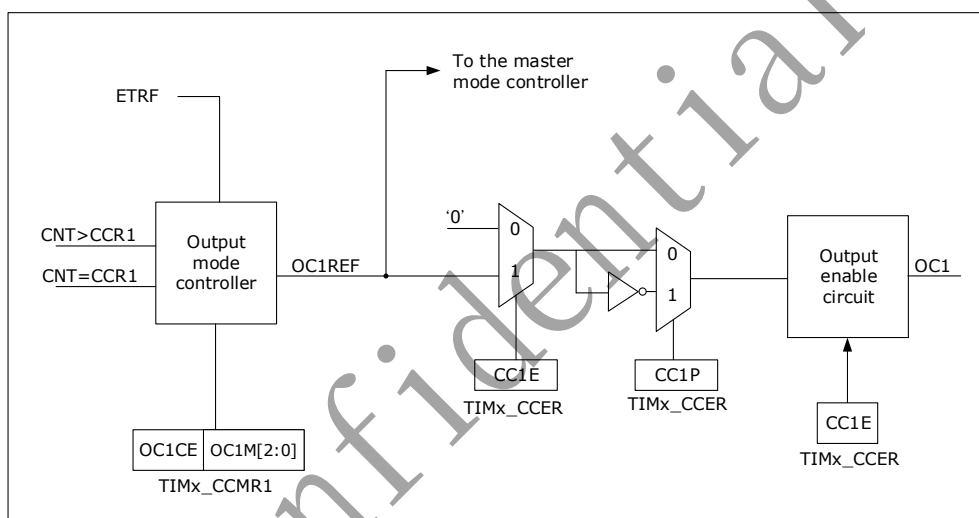


图 12-29 捕捉/比较信道之输出阶段(信道 1 至 4)

捕捉/比较区块是由一预载寄存器与影子寄存器组成。读写皆可存取预载计存器

于捕捉模式下，捕捉实际上皆于影子寄存器完成并复制到预载寄存器。

于比较模式下，预载寄存器内容被复制到影子寄存器与计数器做比较

1.3.5 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(GPTIMA/B CCRx)中。当发生捕获时，若皆已使能则相应的中断可被送出。

以下例子说明如何在 T11 输入的上升沿时捕获计数器的值到 GPTIMA/B_CCR1 寄存器中，步骤如下：

- ◆ 选择有效输入端：GPTIMA/B_CCR1 必须连接到 TI1 输入，所以写入 GPTIMA/B_CCMR1 寄存器中的 CC1S 为 01。只要 CC1S 不为'00'，信道被配置为输入，并且 GPTIMA/B_CCR1 寄存器变为只读。
- ◆ 参照连结至定时器之信号，编程你所需要的输入滤波器期间(当输入为

GPTIMA/B_CCMR1 寄存器之 TIx(ICxF 位其中一位).当触发时输入信号于五个必要内部时钟周期内并不稳定。用户需编制比五个时钟周期还长之滤波器期间。可于 8 个新的电平接续采样被侦测到(于 fDTS 频率采样)时开始 TI1 传输。接着写 GPTIMA/B_CCMR1 寄存器之 IC1F 位为 0011。

- ◆ 选择 TI1 信道的有效转换边沿, 在 GPTIMA/B_CCER 寄存器中写入 CC1P 与 CC1NP 至 0(上升沿)
- ◆ 对输入预除器编程。于例中希望捕获可在每一传输中作用, 因此预除器禁止使用(写 GPTIMA/B_CCMR1 寄存器之 IC1PS 位为'00').
- ◆ 设定 GPTIMA/B_CCER 寄存器 CC1E 位以使能捕捉计数器至捕捉寄存器
- ◆ 如果需要, 透过设定 GPTIMA/B_IER 寄存器之 CC1 位以开启相关中断需求
- ◆ 当输入捕获中断发生时:
- ◆ GPTIMA/B_CCR1 寄存器得到计数器之作用传输值
- ◆ CC1IF 状态被设置(中断状态).CC1OF 也被设定, 若至少有 2 个接续捕获发生, 然而标志未被清除
- ◆ 如设置 GPTIMA/B_IER 寄存器之 CC1 位, 则会产生一个中断。
- ◆ 如设置 GPTIMA/B_DMAEN 寄存器之 CC1 位, 则还会产生一个 DMA 请求。

为了处理捕获溢出, 建议在读出捕获溢出标志之前读取数据, 这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注:ICx 中断需求可由软件设置 GPTIMA/B_EGR 寄存器内对应 CCxG 位产生

1.3.6 PWM 输入模式

该模式是输入捕获模式的一个特例, 除下列区别外, 操作与输入捕获模式相同:

- ◆ 两个 ICx 信号被映射至同一个 TIx 输入
- ◆ 这 2 个 ICx 信号为边沿有效, 但是极性相反
- ◆ 其中一个 TIxFP 信号被作为触发输入信号, 而从模式控制器被配置成复位模式

例:可测量输入到 TI1 上的 PWM 信号的长度(GPTIMA/B_CCR1 寄存器)和占空比(GPTIMA/B_CCR2 寄存器), 具体步骤如下(取决于 CK_INT 的频率和预分频器的值)

- ◆ 选择 GPTIMA/B_CCR1 的有效输入: 置 GPTIMA/B_CCMR1 寄存器的 CC1S=01(选中 TI1)
- ◆ 选择 TI1FP1 的有效极性(用来捕获数据到 GPTIMA/B_CCR1 中和清除计数器): 置 CC1P 为 0(上升沿有效)
- ◆ 选择 GPTIMA/B_CCR2 有效输入:置 GPTIMA/B_CCMR1 寄存器之 CC2S 位至 10(TI1 被选中)
- ◆ 选择 TI1FP2 有效极性(用于 GPTIMA/B_CCR2 捕捉):置 CC2P 位至为'1'(于下降缘作用)
- ◆ 选择可作用触发输入:置 GPTIMA/B_SMCR 寄存器之 TS 位为 101(选择 TI1FP1)
- ◆ 设置属模式控制器于重置模式:置 GPTIMA/B_SMCR 寄存器之 SMS 位为 100
- ◆ 使能捕获: 置 GPTIMA/B_CCER 寄存器中 CC1E=1 且 CC2E=1。

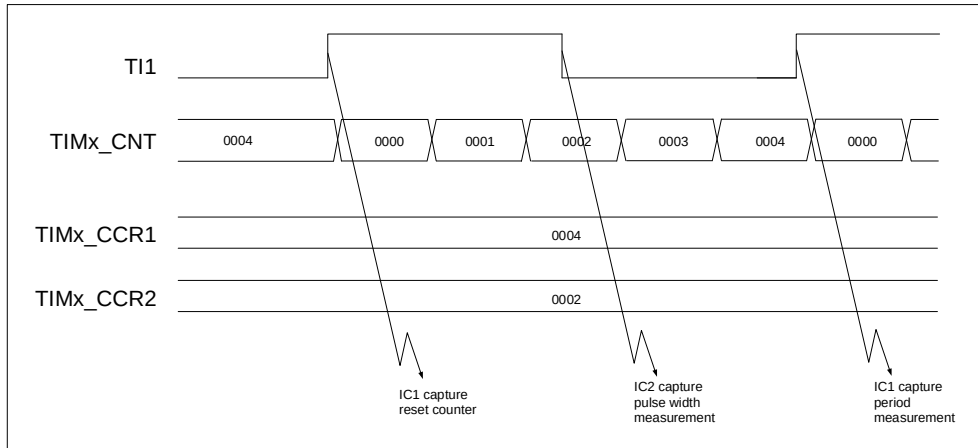


图 12-30 PWM 输入模式时序

1. 3. 7 强制输出模式

在输出模式(GPTIMA/B_CCMRx 寄存器中 CCxS=00)下, 输出比较信号(OCxREF 和相应的 OCx/OCxN)能够直接由软件强置为有效或无效状态, 而不依赖于输出比较寄存器和计数器间的比较结果。

置 GPTIMA/B_CCMRx 寄存器中相应的 OCxM=101, 即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效), 同时 OCx 得到 CCxP 极性相反的信号。

例如: CCxP=0(OCx 高电平有效), 则 OCx 被强置为高电平。

置 GPTIMA/B_CCMRx 寄存器中的 OCxM=100, 可强置 OCxREF 信号为低。

该模式下, 在 GPTIMA/B_CCRx 影子寄存器和计数器之间的比较仍然在进行, 相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。这将会在下节的输出比较模式一节中介绍。

1. 3. 8 输出比较模式

此功能用于控制输出波形或是当时间消逝时之周期标示。

当于捕捉/比较寄存器与计数器发现匹配时, 输出比较功能:

- ◆ 将输出比较模式(GPTIMA/B_CCMRx 寄存器中的 OCxM 位)和输出极性(GPTIMA/B_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时, 输出引脚可以保持它的电平(OCxM=000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。
- ◆ 设置中断状态寄存器中的标志位(GPTIMA/B_RIF 寄存器中的 CCxIF 位)。
- ◆ 若设置了相应的中断屏蔽(GPTIMA/B_IER 寄存器中的 CCxIE 位), 则产生一个中断。
- ◆ 若设置了相应的使能位(GPTIMA/B_DMAEN 寄存器中的 CCxDE 位, GPTIMA/B_CR2 寄存器中的 CCDS 位选择 DMA 请求功能), 则产生一个 DMA 请求。

GPTIMA/B_CCRx 寄存器可由或不由预除寄存器之 OCxPE 位编程

于输出比较模式下, 更新事件 UEV 对于 OCxREF 与 OCx 输出没有效果。时程解析亦为计数器之单一计数。输出比较模式也可被用来输出单一脉冲(单脉冲模式)。于输出比较模式, 更新

事件 UEV 于 OCxREF 与 OCx 无效果

过程:

- ◆ 选定计数器时钟(内部,外部,预除).
- ◆ 写入需要之数据至 GPTIMA/B_ARR 与 GPTIMA/B_CCRx 寄存器.
- ◆ 若中断需求产生则设置 GPTIMA/B_IER 之 CCxIE 位
- ◆ 选择输出模式, 例:
 - 当 CNT 符合 CCRx 时写 OCxM=011 以触发 OCx 输出引脚
 - 写 OCxPE =0 禁用预载寄存器
 - 写 CCxP =0 选择高作用极性
 - 写 CCxE =1 使能输出
- ◆ 设置 GPTIMA/B_CR1 寄存器之 CEN 位以使能计数器

GPTIMA/B_CCRx 寄存器可于任何时间被软件更新以控制输出波形,预载寄存器提供预除寄存器并未使能(OCxPE='0',此外 GPTIMA/B_CCRx 影子寄存器只有在下一个更新事件 UEV 被更新).图 11-31 提供一范例:

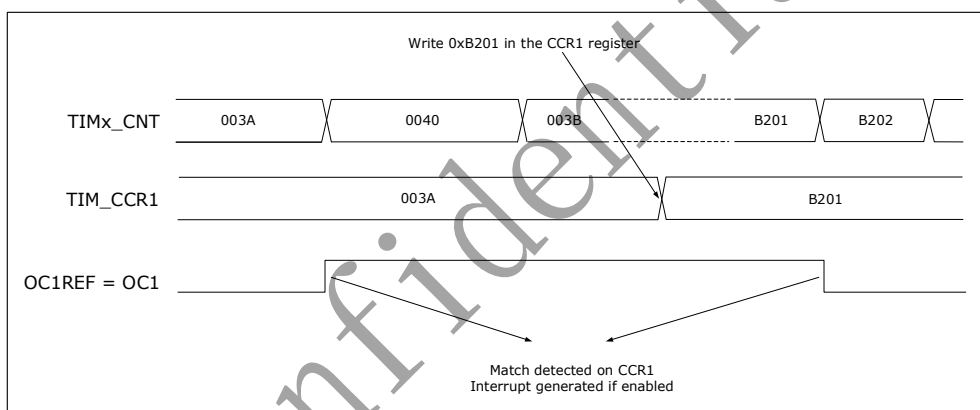


图 12-31 输出比较模式,触发 OC1

1.3.9 PWM 模式

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

在 GPTIMA/B_CCMRx 寄存器中的 OCxM 位写入'110'(PWM 模式 1)或'111'(PWM 模式 2),能够独立地设置每个 OCx 输出信道产生一路 PWM。必须设置 GPTIMA/B_CCMRx 寄存器 OCxPE 位以使能相应的预装载寄存器,最后还要设置 GPTIMA/B_CR1 寄存器的 ARPE 位,(在向上计数或中心对称模式中)使能自动重载的预装载寄存器。

仅当发生一个更新事件的时候,预装载寄存器才能被传送到影子寄存器,因此在计数器开始计数之前,必须通过设置 GPTIMA/B_EGR 寄存器中的 UG 位以初始化所有寄存器。

OCx 的极性可以通过软件在 GPTIMA/B_CCER 寄存器中的 CCxP 位设置,它可以设置为高电平有效或低电平有效。GPTIMA/B_CCER 寄存器中的 CCxE,CCxNE,MOE,OSSI 和 OSSR 位联合控制 OCx 输出使能(GPTIMA/B_CCER 与 GPTIMA/B_BDTR 寄存器)。详见 GPTIMA/B_CCER 寄存器的描述。

在 PWM 模式(模式 1 或模式 2)下, GPTIMA/B_CNT 和 GPTIMA/B_CCRx 始终在进行比较,(依据计数器的计数方向)以确定是否符合 $GPTIMA/B\_CCRx \leq GPTIMA/B\_CNT$ 或者 $GPTIMA/B\_CNT \leq GPTIMA/B\_CCRx$ 。

根据 GPTIMA/B_CR1 寄存器中 CMS 位的状态, 定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

PWM 边沿对齐模式

◆ 向上计数配置

当 GPTIMA/B_CR1 寄存器中的 DIR 位为低的时候执行向上计数。参考以下为一 PWM 模式 1 之例子。当 $GPTIMA/B\_CNT < GPTIMA/B\_CCRx$ 时, PWM 参考信号 OCxREF 为高, 否则为低。如果 GPTIMA/B_CCRx 中的比较值大于自动重装载值 (GPTIMA/B_ARR), 则 OCxREF 保持为‘1’。如果比较值为 0, 则 OCxREF 保持为‘0’。下图为 GPTIMA/B_ARR=8 时边沿对齐的 PWM 波形实例。

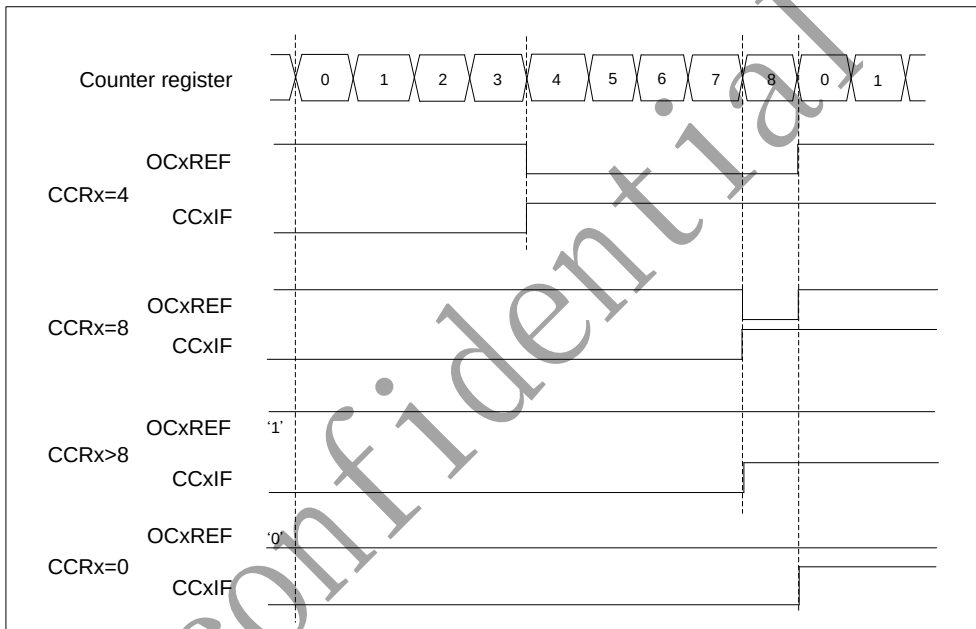


图 12-32 边缘对齐 PWM 波形, 向上计数(ARR=8)

◆ 向下计数配置

当 GPTIMA/B_CR1 寄存器的 DIR 位为高时执行向下计数。在 PWM 模式 1, 当 $GPTIMA/B\_CNT > GPTIMA/B\_CCRx$ 时参考信号 OCxREF 为低, 否则为高。如果 GPTIMA/B_CCRx 中的比较值大于 GPTIMA/B_ARR 中的自动重装载值, 则 OCxREF 保持为‘1’。该模式下不能产生 0% 的 PWM 波形。

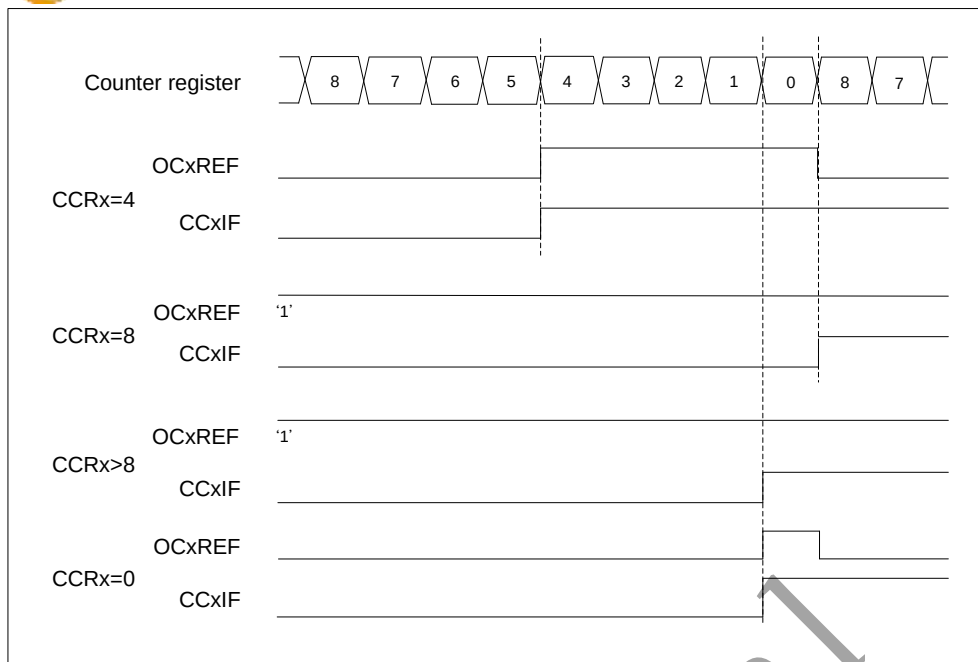


图 12-33 边缘对齐 PWM 波形，向下计数(ARR=8)

PWM 中央对齐模式

当 GPTIMA/B_CR1 寄存器中的 CMS 位不为'00'时为中央对齐模式(所有其他的配置对 OCxREF/OCx 信号都有相同的作用)。根据不同的 CMS 位设置，比较标志可以在计数器向上计数时被置 1、在计数器向下计数时被置 1、或在计数器向上和向下计数时被置 1。

GPTIMA/B_CR1 寄存器中的计数方向位(DIR)由硬件更新，不要用软件修改它

下图给出了一些中央对齐的 PWM 波形的例子:

- ◆ GPTIMA/B_ARR=8,
- ◆ PWM 模式 1,

GPTIMA/B_CR1 寄存器的 CMS=01，在中央对齐模式 1 下，当计数器向下计数时设置比较标志。

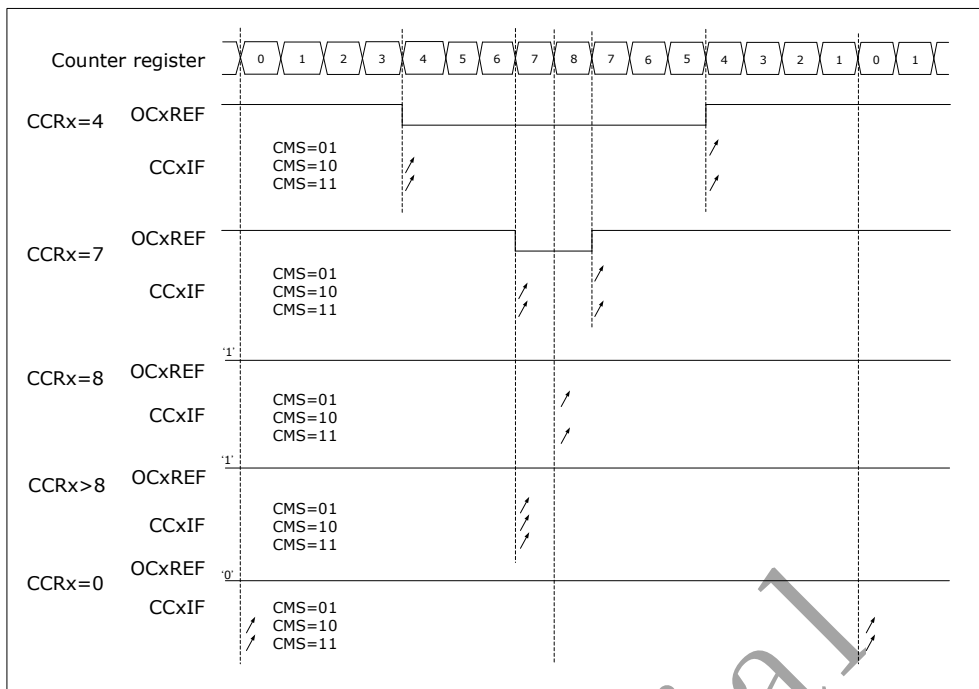


图 12-34 中央对齐 PWM 波形(ARR=8)

使用中央对齐模式的提示：

- ◆ 进入中央对齐模式时，使用当前的向上/向下计数配置；这就意味着计数器向上还是向下计数取决于 GPTIMA/B_CR1 寄存器中 DIR 位的当前值。此外，软件不能同时修改 DIR 和 CMS 位。
- ◆ 不推荐当运行在中央对齐模式时改写计数器，因为这会产生不可预知的结果。特别地：
 - 如果写入计数器的值大于自动重加载的值 (GPTIMA/B_CNT > GPTIMA/B_ARR)，则方向不会被更新。例如，如果计数器正在向上计数，它就会继续向上计数。
 - 如果将 0 或者 GPTIMA/B_ARR 的值写入计数器，方向被更新，但不产生更新事件 UEV。
- ◆ 使用中央对齐模式最保险的方法，就是在启动计数器之前产生一个软件更新(设置 GPTIMA/B_EGR 位中的 UG 位)，并且不要在计数进行过程中修改计数器的值。

1. 3. 10 单脉冲模式

单脉冲模式(OPM)为前一模式之特例。其允许计数器对应刺激起始并于可编程延迟后产生可编程长度之脉冲。

起始计数器可由属模式控制器控制。由输出比较模式或 PWM 模式产生波形。可透过设置 GPTIMA/B_CR1 寄存器之 OPM 位以选择单脉冲模式。如此计数器会在下一个更新事件 UEV 自动停止。

单一脉冲仅可于比较值不同于计数器初始值时被正确产生。在开始之前(当定时器于等待触发器时)，必须依照下列方式配置

- ◆ 向上计数: $CNT < CCRx \leq ARR$ (特例, $0 < CCRx$)
- ◆ 向下计数: $CNT > CCRx$

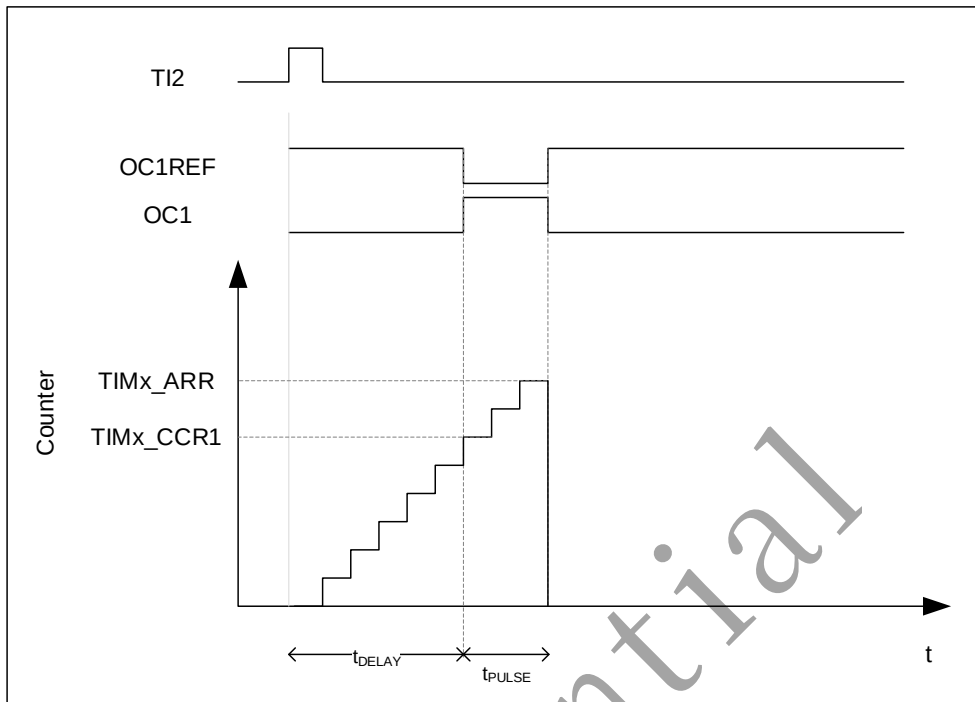


图 12-35 单脉冲模式范例

例如，你需要在在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲，从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后。

假定 TI2FP2 作为触发 1:

- ◆ 置 GPTIMA/B_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。
- ◆ 置 GPTIMA/B_CCER 寄存器中的 CC2P=0 和 CC2NP=0，使 TI2FP2 能够检测上升沿。
- ◆ 置 GPTIMA/B_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发 (TRGI)。
- ◆ 置 GPTIMA/B_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器决定(要考虑时钟频率和计数器预分频器)

- ◆ t_{DELAY} 由 GPTIMA/B_CCR1 寄存器中的值定义。
- ◆ t_{PULSE} 由自动装载值和比较值之间的差值定义($GPTIMA/B\_ARR - GPTIMA/B\_CCR1$)。
- ◆ 假定当发生比较匹配时要产生从 0 到 1 的变换，当计数器达到预装载值时要产生一个从 1 到 0 的变换；首先要置 GPTIMA/B_CCMR1 寄存器的 OC1M=111，选择 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 GPTIMA/B_CCMR1 中的 OC1PE=1 和 GPTIMA/B_CR1 寄存器中的 ARPE；此时必须向 GPTIMA/B_CCR1 寄存器中填写比较值，在 GPTIMA/B_ARR 寄存器中填写自动装载值，设置 UG 位

来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，GPTIMA/B_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 GPTIMA/B_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。当 GPTIMA/B_CR1 寄存器中的 OPM=0 时，进入重复模式。

特殊情况：OCx 快速使能

在单脉冲模式下，在 Tix 输入脚的边沿检测逻辑设置 CEN 位来启动计数器。然后计数器和比较值间的比较结果驱动输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 tDELAY。

如果需要以最小延时输出波形，可以设置 GPTIMA/B_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF(和 OCx)直接响应激励而不再依赖比较的结果，输出的波形与比较匹配时的波形一样。OCxFE 只在信道配置为 PWM 模式 1 和 PWM 模式 2 时起作用。

1. 3. 11 清除外部事件 OCxREF 信号

给定信道之 OCxREF 信号，于高电平被应用于 OCREF_CLR_INPUT 时可被清除(对应 GPTIMA/B_CCMR 寄存器之 OCxCE 使能位设置为 1)。OCxREF 保持为低，直到下一更新事件(UEV)发生。此功能仅可被用于 PWM 模式。于强制模式下则无作用。

OCREF_CLR_INPUT 在 COMP_TRIG 输入与 ETRF(滤波后之 ETR)间可被选定，透过设置 GPTIMA/B_SMCR 寄存器之 OCCS 位。

当 ETRF 被选定，ETR 必须如下设置：

给定信道之 OCxREF 信号可透过使用高电平于 ETRF 输入而被驱使为低(对应 GPTIMA/B_CCMR 寄存器之 OCxCE 使能位设置为 1)OCxREF 信号维持低，直到下一更新事件(UEV)发生。

此功能仅可于 PWM 模式下使用，于强制模式下无法使用。

举例说明，OCxREF 信号可被连接至用于电流控制比较器输出。于此例中，ETR 必须设置如下：

- ◆ 外部触发预除器必须保持禁用:GPTIMA/B_SMC 寄存器之 ETPS[1:0]位设置为'00'
- ◆ 外部时钟模式 2 必须禁用:GPTIMA/B_SMCR 寄存器之 ECE 位设置为'0'.
- ◆ 外部触发极性(ETP)与外部触发滤波器(ETF)可根据用户需求设置
- ◆ 下图标示 OCxCE 使能位值，当 ETRF 输入变成高时，OCxREF 位之信号行为。此例中定时器 GPTIMA/B 于 PWM 模式下被编程。

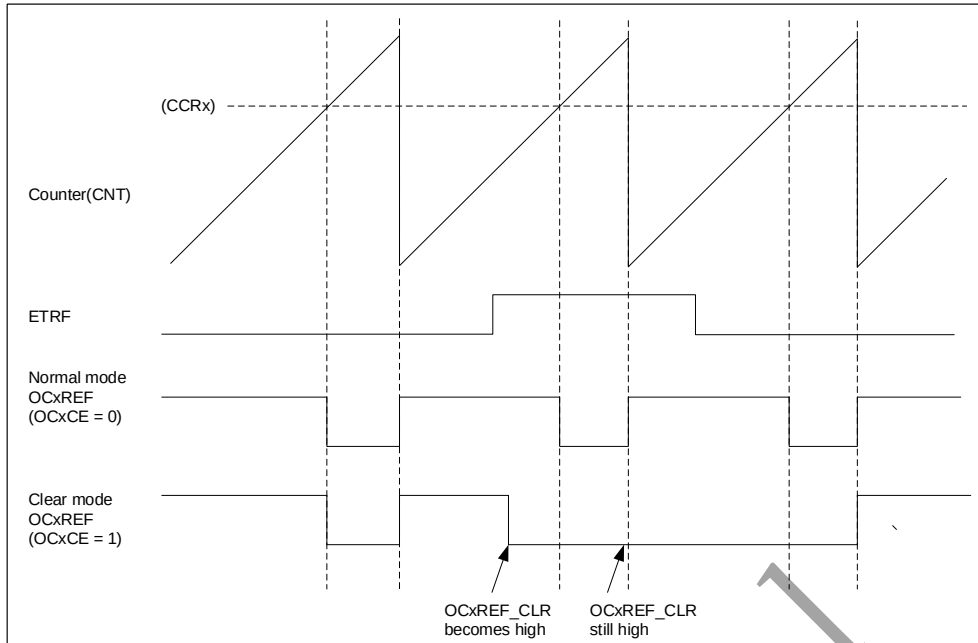


图 12-36 清除 GPTIMA/B 的 OCxREF

注:若为占空比 100%之 PWM(若 $CCR_x > ARR$),OCxREF 会于下一个计数器上溢再度使能

1. 3. 12 编码器接口模式

选择编码器接口模式的方法是: 如果计数器只在 TI2 的边沿计数, 则置 GPTIMA/B_SMCR 寄存器中的 SMS=001; 如果只在 TI1 边沿计数, 则置 SMS=010; 如果计数器同时在 TI1 和 TI2 边沿计数, 则置 SMS=011。

通过设置 GPTIMA/B_CCER 寄存器中的 CC1P 和 CC2P 位, 可以选择 TI1 和 TI2 极性; 如果需要, 还可以对输入滤波器编程。CC1P 和 CC2P 位必须保持为低。

两个输入 TI1 和 TI2 被用来作为增量编码器的接口。假定计数器已经启动(GPTIMA/B_CR1 寄存器中的 CEN=1), 则计数器由每次在 TI1FP1 或 TI2FP2 上的有效跳变驱动。TI1FP1 和 TI2FP2 是 TI1 和 TI2 在通过输入滤波器和极性控制后的信号; 如果没有滤波和变相, 则 TI1FP1=TI1, TI2FP2=TI2。根据两个输入信号的跳变顺序, 产生了计数脉冲和方向信号。依据两个输入信号的跳变顺序, 计数器向上或向下计数, 同时硬件对 GPTIMA/B_CR1 寄存器的 DIR 位修改。不管计数器是依靠 TI1 还是 TI2 或者同时依靠 TI1 和 TI2 计数, 在任一输入端 (TI1 或者 TI2) 的跳变都会重新计算 DIR 位。

编码器接口模式基本上相当于使用了一个带有方向选择的外部时钟。这意味着计数器只在 0 到 GPTIMA/B_ARR 寄存器的自动装载值之间连续计数(根据方向, 或是 0 到 ARR 计数, 或是 ARR 到 0 计数)。所以在开始计数之前必须配置 GPTIMA/B_ARR; 同样, 捕获器、比较器、预分频器、重复计数器、触发输出特性等仍工作如常。编码器模式和外部时钟模式 2 不兼容, 因此不能同时操作。

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

有效边沿	有效边沿相对 信号的电平 (TI1FP1 对 TI2, TI2FP2 对应 TI1)	TI1FP1 信号		TI2FP2 信号	
		上升沿	下降沿	上升沿	下降沿
仅在 TI1 计 数	高	下降	上升	不计数	不计数
	低	上升	下降	不计数	不计数
仅在 TI2 计 数	高	不计数	不计数	上升	下降
	低	不计数	不计数	下降	上升
在 TI1 和 TI2 上计数	高	下降	上升	上升	下降
	低	上升	下降	下降	上升

Table12-1 计数方向与编码器信号的关系

一个外部的增量编码器可以直接与 MCU 连接而不需要外部接口逻辑。但是，一般会使用比较器将编码器的差动输出转换到数字信号，这大大增加了抗噪声干扰能力。编码器输出的第三个信号表示机械零点，可以把它连接到一个外部中断输入并触发一个计数器复位。

下图提供一计数器运作范例，显示计数信号产生与方向控制。亦可显示如何补偿输入抖动当上下缘皆被选定时。该状况可能发生于若传感器被设置于接近其中一切换点。我们假设此例设置如下：

- ◆ CC1S=01 (GPTIMA/B_CCMR1 寄存器, IC1FP1 映射到 TI1)
- ◆ CC2S=01 (GPTIMA/B_CCMR2 寄存器, IC2FP2 映射到 TI2)
- ◆ CC1P=0 (GPTIMA/B_CCER 寄存器, IC1FP1 不反相, IC1FP1=TI1)
- ◆ CC2P=0 (GPTIMA/B_CCER 寄存器, IC2FP2 不反相, IC2FP2=TI2)
- ◆ SMS=011 (GPTIMA/B_SMCR 寄存器, 所有的输入均在上升沿和下降沿有效).
- ◆ CEN=1 (GPTIMA/B_CR1 寄存器, 计数器使能)

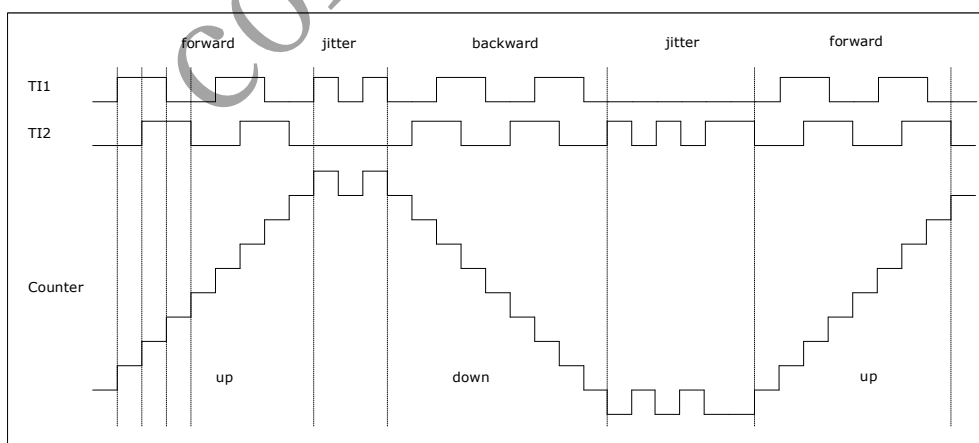


图 12-1 编码器模式下的计数器操作实例

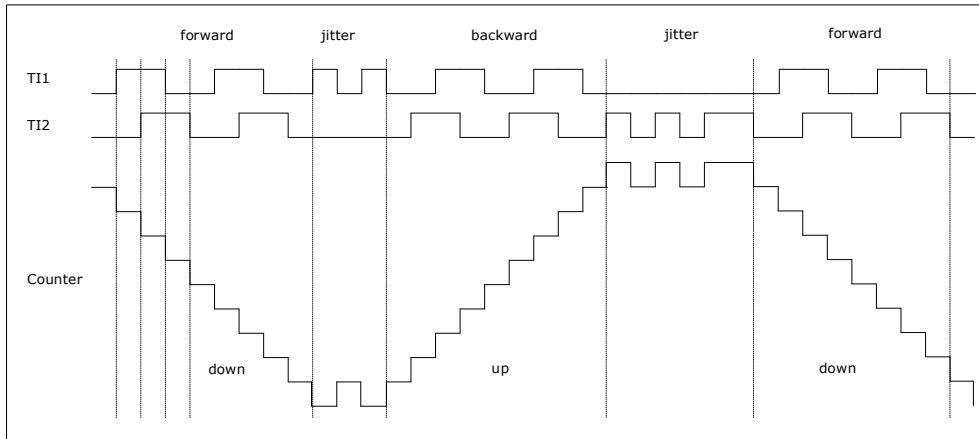


图 12-37IC1FP1 反相的编码器接口模式实例(CC1P='1', 其他配置与上例相同)

当设置编码器接口模式时, 定时器提供传感器当前位置信息。可透过量测两个第二定时器设定为捕捉模式之编码器事件周期得到动态信息(速度, 加速度与减速度)。编码器输出标示机械零位可做为此功能使用。根据两个事件间之时间可知, 计数器亦能于通常时间被读取。若可行亦可透过锁存计数器值转换为第三输入捕捉寄存器(捕捉信号必须具周期性且可被另一定时器产生)。

1. 3. 13 定时器输入 XOR 功能

GPTIMA/B_CR2 寄存器之 TI1S 位允许信道 1 输入滤波器被连接至 XOR 门输出, 结合三个输入引脚 GPTIMA/B_CH1,GPTIMA/B_CH2 与 GPTIMA/B_CH3。

XOR 输入可被当作所有定时器输入功能使用, 如触发器或输入捕捉。于下列章节说明作为连接霍尔传感器之范例

1. 3. 14 定时器和外部触发的同步

GPTIMA/B 定时器能够在多种模式下和一个外部的触发同步: 复位模式、门控模式和触发模式。

1. 3. 14. 1 从模式: 复位模式

在一个触发输入发生事件时, 计数器和它的预分频器能够重新被初始化; 同时, 如果 GPTIMA/B_CR1 寄存器的 URS 位为低, 还产生一个更新事件 UEV; 然后所有的预装载寄存器(GPTIMA/B_ARR, GPTIMA/B_CCRx)都被更新。

在以下的例子中, TI1 输入端的上升沿导致向上计数器被清零:

- ◆ 配置信道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中, 不需要任何滤波器, 因此保持 IC1F=0000)。触发操作中不使用捕获预分频器, 所以不需要配置。CC1S 位只选择输入捕获源, 即 GPTIMA/B_CCMR1 寄存器中 CC1S=01。置 GPTIMA/B_CCER 寄存器中 CC1P=0 和 CC1NP=0 以确定极性(只检测上升沿)。
- ◆ 置 GPTIMA/B_SMCR 寄存器中 SMS=100, 配置定时器为复位模式; 置 GPTIMA/B_SMCR 寄存器中 TS=101, 选择 TI1 作为输入源。
- ◆ 置 GPTIMA/B_CR1 寄存器中 CEN=1, 启动计数器。

计数器开始依据内部时钟计数, 然后正常运转直到 TI1 出现一个上升沿; 此时, 计数器被清零然后从 0 重新开始计数。同时, 触发标志(GPTIMA/B_RIF 寄存器中的 TIF 位)被设置, 并根据 GPTIMA/B_IER 寄存器中 TIE(中断使能)位和 GPTIMA/B_DMAEN 寄存器中 TDE(DMA 使能)位的设置, 产生一个中断请求或一个 DMA 请求。

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

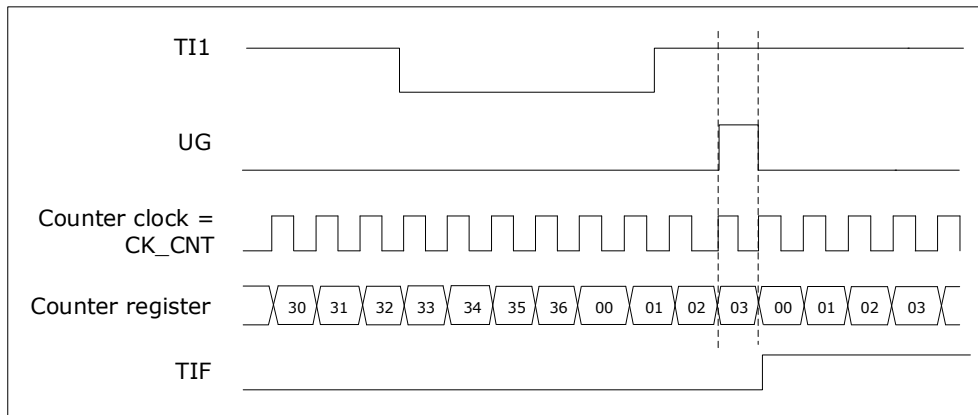


图 12-38 重置模式控制电路

1. 3. 14. 2 从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

- ◆ 配置信道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 GPTIMA/B_CCMR1 寄存器中 CC1S=01。置 GPTIMA/B_CCER 寄存器中 CC1P=1 和 CC1NP=0 以确定极性(只检测低电平)。
- ◆ 置 GPTIMA/B_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 GPTIMA/B_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- ◆ 置 GPTIMA/B_CR1 寄存器中 CEN=1，启动计数器。(在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何)

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

TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的重同步电路。

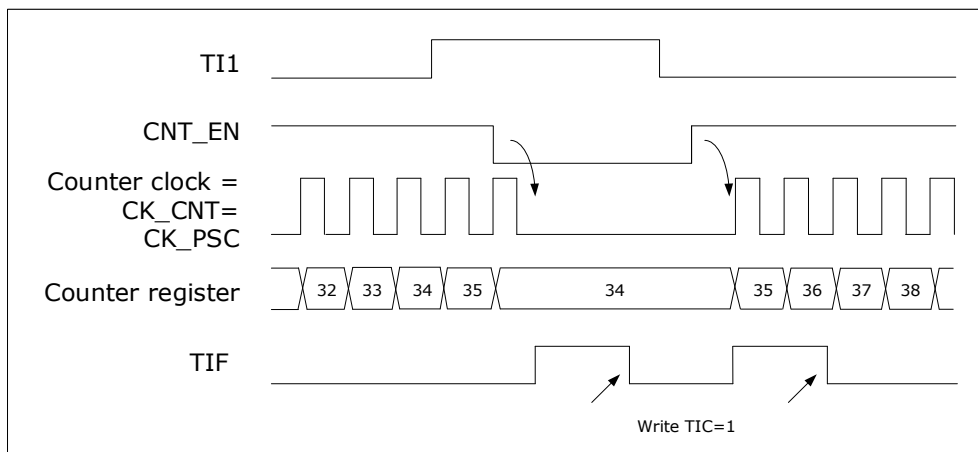


图 12-39 门控模式控制电路

1. 3. 14. 3 从模式：触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- ◆ 配置信道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 GPTIMA/B_CCMR1 寄存器中 CC2S=01。置 GPTIMA/B_CCER 寄存器中 CC2P=1 和 CC1NP=0 以确定极性(只检测低电平)。
- ◆ 置 GPTIMA/B_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 GPTIMA/B_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器按内部时钟开始计数，同时设置 TIF 标志。

TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

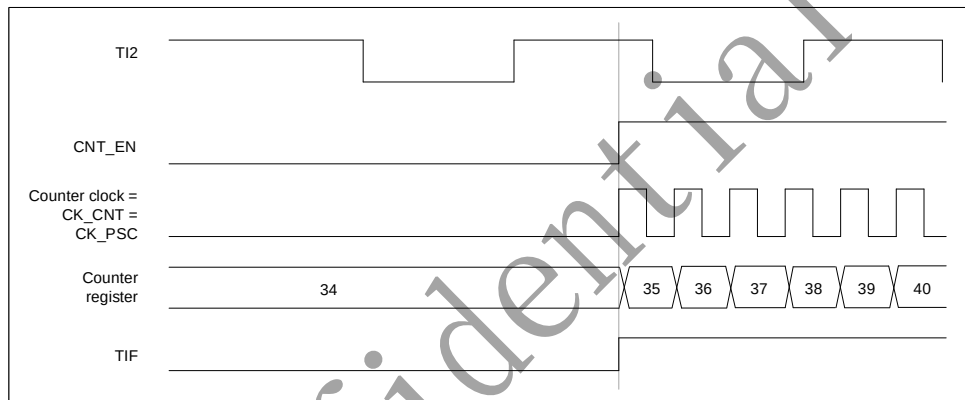


图 12-40 触发模式控制电路

1. 3. 14. 4 从模式：外部时钟模式 2+触发模式

外部时钟模式 2 可以与另一种从模式(外部时钟模式 1 和编码器模式除外)一起使用。这时，ETR 信号被用作外部时钟的输入，可以选择另一个输入作为触发输入（在复位模式、门控模式或触发模式）。不建议使用 GPTIMA/B_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。

在下面的例子中，在 TI1 上出现一个上升沿后，向上计数器在 ETR 的每一个上升沿加一：

- ◆ 通过 GPTIMA/B_SMCR 寄存器配置外部触发输入电路：ETF=0000：没有滤波
 - ETPS=00：不用预分频器
 - ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式 2。
- ◆ 按如下配置信道 1，检测 TI 的上升沿：
 - IC1F=0000：没有滤波
 - 触发操作中不使用捕获预分频器，不需要配置
 - 置 GPTIMA/B_CCMR1 寄存器中 CC1S=01，选择输入捕获源
 - 置 GPTIMA/B_CCER 寄存器中 CC1P=0 和 CC1NP=0 以确定极性(只检测上升沿)

- ◆ 置 GPTIMA/B_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 GPTIMA/B_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。

ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

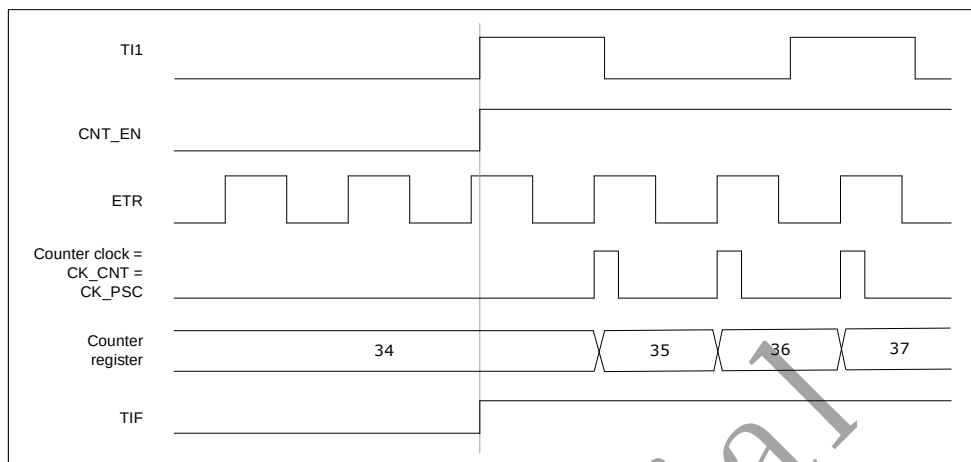


图 12-41 外部时钟模式 2+触发模式下的控制电路

1. 3. 15 定时器同步

所有 GPTIMA/B 定时器在内部相连，用于定时器同步或链接。当一个定时器处于主模式时，它可以对另一个处于从模式的定时器的计数器进行复位、启动、停止或提供时钟等操作。

下图显示了触发选择和主模式选择模块的概况

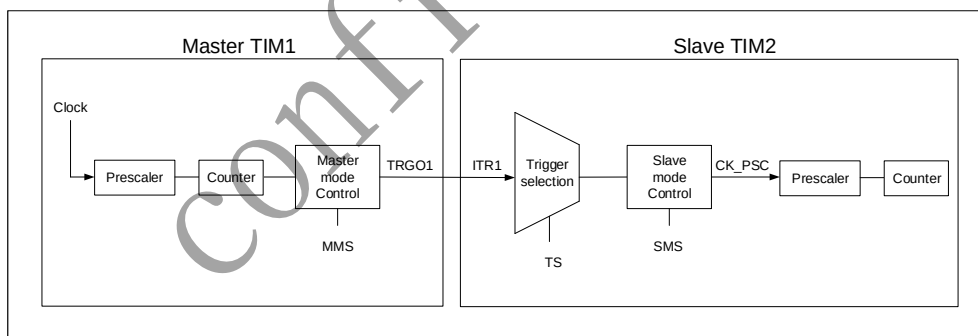


图 12-42 主/从定时器范例

1. 3. 15. 1 使用一个定时器使能另一个定时器

在这个例子中，定时器 2 的使能由定时器 1 的输出比较控制。参考图 130 的连接。只当定时器 1 的 OC1REF 为高时，定时器 2 才对分频后的内部时钟计数。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3 ($f_{CK\_CNT} = f_{CK\_INT}/3$) 得到。

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

- ◆ 配置定时器 1 的 OC1REF 波形(TIM1_CCMR1 寄存器)
- ◆ 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)

- ◆ 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
- ◆ 置 TIM2_CR1 寄存器的 CEN=1 以使能定时器 2
- ◆ 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1

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

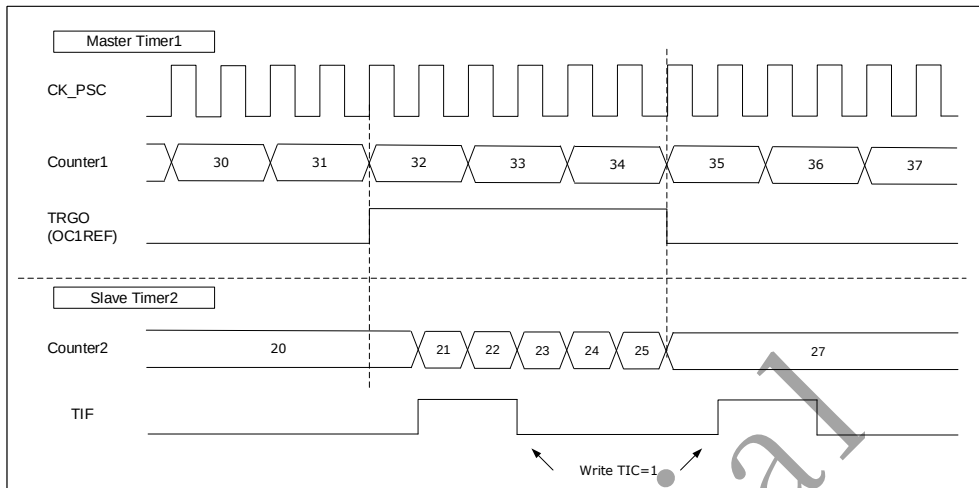


图 12-43 门控从定时器使用主定时器 OC1REF

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

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

- ◆ 配置定时器 1 为主模式，送出输出比较 1 参考信号(OC1REF)做为触发输出(TIM1_CR2 寄存器的 MMS=100)。
- ◆ 配置定时器 1 的 OC1REF 波形(TIM1_CCMR1 寄存器)。
- ◆ 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- ◆ 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
- ◆ 置 TIM1_EGR 寄存器的 UG=1，复位定时器 1。
- ◆ 置 TIM2_EGR 寄存器的 UG=1，复位定时器 2。
- ◆ 写'0xE7'至定时器 2 的计数器(TIM2_CNT)，初始化它为 0xE7。
- ◆ 置 TIM2_CR1 寄存器的 CEN=1 以使能定时器 2。
- ◆ 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1。
- ◆ 置 TIM1_CR1 寄存器的 CEN=0 以停止定时器 1。

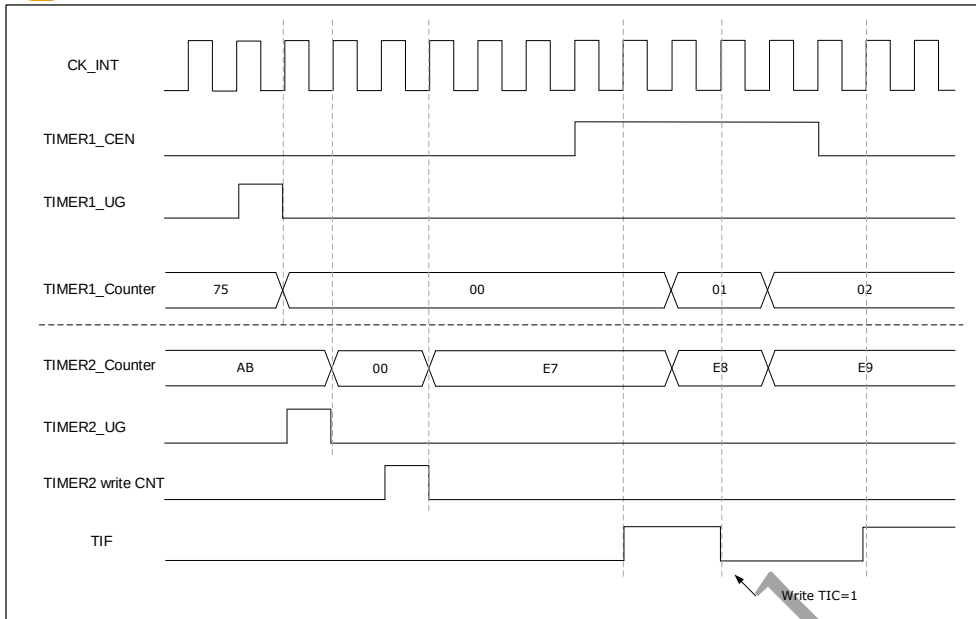


图 12-44 通过使能定时器 1 可以控制定时器 2

1. 3. 15. 2 使用一个定时器去启动另一个定时器

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

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

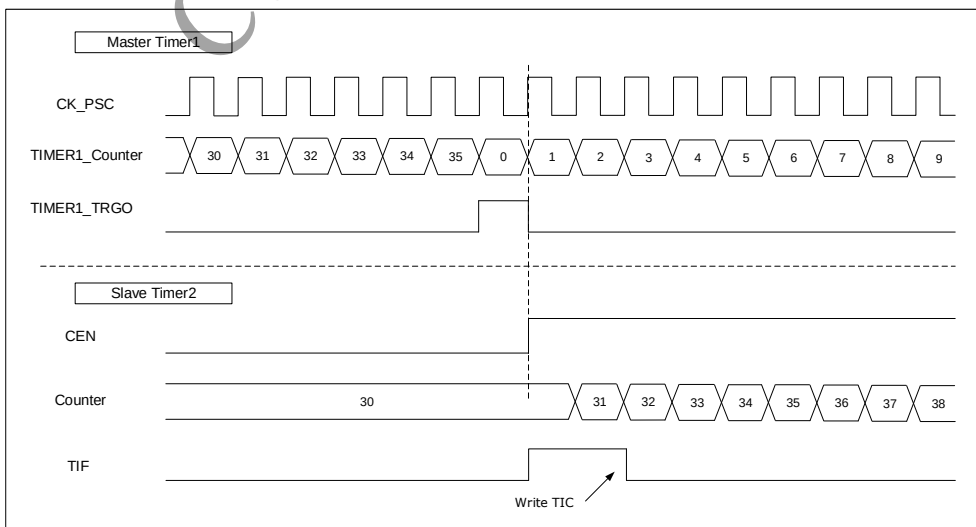


图 12-45 触发中从定时器使用主定时器更新事件

在上图例子中，可以在启动计数之前初始化两个计数器。上图显示在上上图相同配置情况下，使用触发模式而不是门控模式(TIM2_SMCR 寄存器的 SMS=110)的动作。

1. 3. 15. 3 使用一个外部触发同步地启动 2 个定时器

这个例子中当定时器 1 的 TI1 输入上升时使能定时器 1，使能定时器 1 的同时使能定时器 2，参见图 130。为保证计数器的对齐，定时器 1 必须配置为主/从模式(对应 TI1 为从，对应定时器 2 为主)：

- ◆ 配置定时器 1 为主模式，送出它的使能做为触发输出(TIM1_CR2 寄存器的 MMS=001)。
- ◆ 配置定时器 1 为从模式，从 TI1 获得输入触发(TIM1_SMCR 寄存器的 TS='101')。
- ◆ 配置定时器 1 为触发模式(TIM1_SMCR 寄存器的 SMS='110')。
- ◆ 配置定时器 1 为主/从模式，TIM1_SMCR 寄存器的 MSM='1'。
- ◆ 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- ◆ 配置定时器 2 为触发模式(TIM2_SMCR 寄存器的 SMS='110')。

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

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

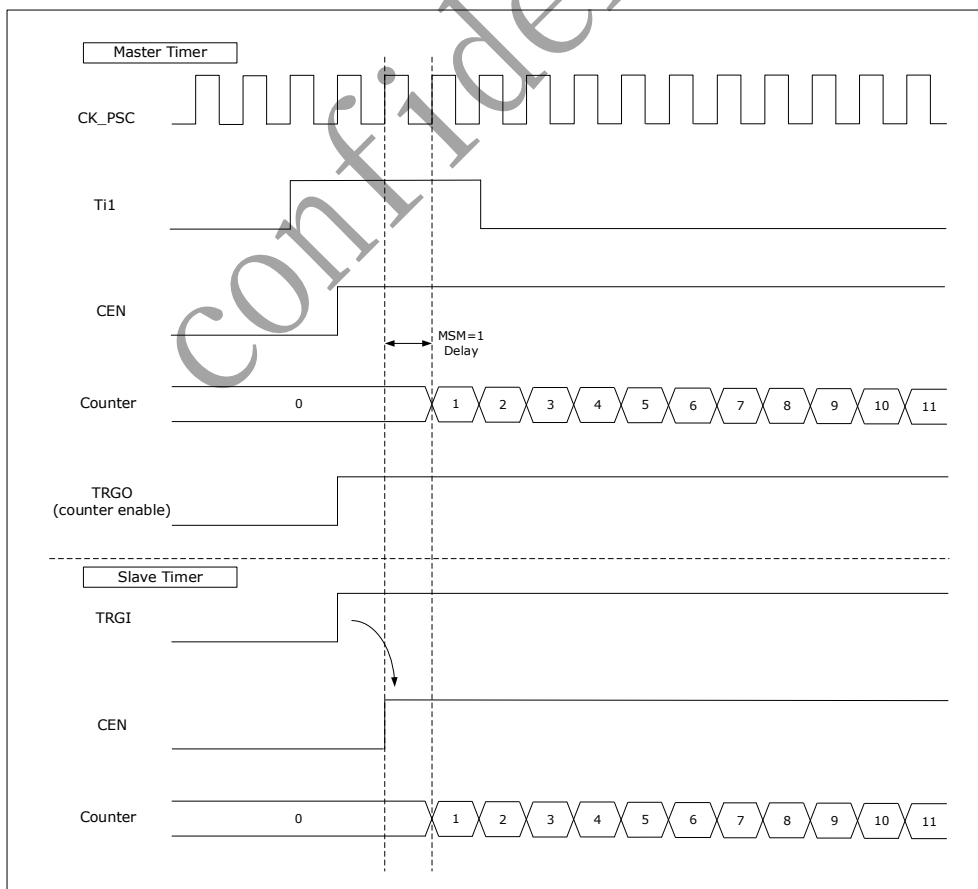


图 12-46 使用定时器 1 的 TI1 输入触发定时器 1 和定时器 2

1. 3. 16 除错模式

当微控制器进入除错模式(MCU 中心终止),GPTIMA/B 计数器停止计数

1. 4 GPTIMA/B 寄存器描述

1. 4. 1 控制寄存器 1 (GPTIMA/B_CR1)

		Address offset:		0x00											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CKD[1:0]		ARPE	CMS[1:0]		DIR	OPM	URS	UDIS	CEN

Name	Bit	Type	Reset value	Description
Reserved	31:10	N/A	0x00	保留, 保持复位值
CKD	9:8	R/W	0x0	CKD: 时钟分频因子 定义在定时器时钟(CK_INT)频率与数字滤波器(ETR, TIx)使用的采样频率之间的分频比例。 00: tDTS = tCK_INT 01: tDTS = 2 x tCK_INT 10: tDTS = 4 x tCK_INT 11: 保留
ARPE	7	R/W	0x0	ARPE: 自动重载预装载允许位 0: GPTIM_ARR 寄存器没有缓冲; 1: GPTIM_ARR 寄存器缓冲器有效。
CMS	6:5	R/W	0x0	CMS: 选择中央对齐模式 00: 边沿对齐模式。计数器依据方向位(DIR)向上或向下计数。 01: 中央对齐模式 1。计数器交替地向上和向下计数。配置为输出的通道 (GPTIM_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位, 只在计数器向下计数时被设置。 10: 中央对齐模式 2。计数器交替地向上和向下计数。配置为输出的通道 (GPTIM_CCMRx 寄存器中 CCxS=00)的输出比较中断标志位, 只在计数器向上计数时被设置。 11: 中央对齐模式 3。计数器交替地向上和向下计数。配置为输出的通道 (GPTIM_CCMRx 寄存器中 CCxS=00) 的输出比较中断标志位, 在计数器向上和向下计数时均 被设置。 注: 在计数器开启时(CEN = 1), 不允许从边沿对齐模式 转换到中央对齐模式 。
DIR	4	R/W	0x0	DIR: 方向 0: 计数器向上计数; 1: 计数器向下计数。 注: 当计数器配置为中央对齐模式或编码器模式时, 该位为只读。
OPM	3	R/W	0x0	OPM: 单脉冲模式 0: 在发生更新事件时, 计数器不停止; 1: 在发生下一次更新事件 (清除 CEN 位) 时, 计数器停止

URS	2	R/W	0x0	URS: 更新请求源 软件通过该位选择 UEV 事件的源 0: 如果使能了更新中断或 DMA 请求, 则下述任一事件产生更新中断或 DMA 请求: — 计数器溢出 / 下溢 — 设置 UG 位 — 从模式控制器产生的更新 1: 如果使能了更新中断或 DMA 请求, 则只有计数器溢出 / 下溢才产生更新中断或 DMA 请求。
	1	R/W	0x0	UDIS: 禁止更新 软件通过该位允许/禁止 UEV 事件的产生 0: 允许 UEV。更新(UEV)事件由下述任一事件产生: — 计数器溢出/下溢 — 设置 UG 位 — 从模式控制器产生的更新具有缓存的寄存器被装入它们的预装载值。 1: 禁止 UEV。不产生更新事件, 影子寄存器 (ARR、PSC、CCR_x) 保持它们的值。如果设置了 UG 位或从模式控制器发出了一个硬件复位, 则计数器和预分频器被重新初始化。
	0	R/W	0x0	CEN: 使能计数器 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。 在单脉冲模式下, 当发生更新事件时, CEN 被自动清除。

1. 4. 2 控制寄存器 2 (GPTIMA/B_CR2)

		Address offset:		0x04											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									MMS[2:0]		CCDS	Res.	Res.	Res.	

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留, 保持复位值
TI1S	7	R/W	0x0	TI1S: TI1 选择 0: GPTIM_CH1 引脚连到 TI1 输入; 1: GPTIM_CH1、GPTIM_CH2 和 GPTIM_CH3 引脚经异或后连到 TI1 输入。

MMS[2:0]	6:4	R/W	0x0	MMS: 主模式选择 这 3 位用于选择在主模式下送到从定时器的同步信息 (TRGO)。可能的组合如下: 000: 复位 - GPTIM_EGR 寄存器的 UG 位被用于作为触发输出 (TRGO)。如果是触发输入产生的复位 (从模式控制器处于复位模式), 则 TRGO 上的信号相对实际的复位会有一个延迟。 001: 使能 - 计数器使能信号 CNT_EN 被用于作为触发输出 (TRGO)。有时需要在同一时间启动多个定时器或控制在一段时间内使能从定时器。计数器使能信号是通过 CEN 控制位和门控模式下的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时, TRGO 上会有一个延迟, 除非选择了主 / 从模式 (见 GPTIM_SMCR 寄存器中 MSM 位的描述)。 010: 更新 - 更新事件被选为触发输入 (TRGO)。例如, 一个主定时器的时钟可以被用作一个从定时器的预分频器。 011: 比较脉冲 - 在发生一次捕获或一次比较成功时, 当要设置 CC1IF 标志时 (即使它已经为高), 触发输出送出一个正脉冲 (TRGO)。 100: 比较 - OC1REF 信号被用于作为触发输出 (TRGO)。 101: 比较 - OC2REF 信号被用于作为触发输出 (TRGO)。 110: 比较 - OC3REF 信号被用于作为触发输出 (TRGO)。 111: 比较 - OC4REF 信号被用于作为触发输出 (TRGO)。
				CCDS: 捕获 / 比较的 DMA 选择 0: 当发生 CCx 事件时, 送出 CCx 的 DMA 请求; 1: 当发生更新事件时, 送出 CCx 的 DMA 请求。
CCDS	3	R/W	0x0	
Reserved	2:0	N/A	0x0	保留, 保持复位值

1. 4. 3 从模式控制寄存器(GPTIMA/B_SMCR)

Address offset:		0x08													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETP	ECE	ETPS[1:0]		ETRF[3:0]				MSM	TS[2:0]			Reserved	SMS[2:0]		

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留, 保持复位值
ETP	15	R/W	0x0	ETP: 外部触发极性 该位选择是用 ETR 还是 ETR 的反相来作为触发操作 0: ETR 不反相, 高电平或上升沿有效; 1: ETR 被反相, 低电平或下降沿有效。
ECE	14	R/W	0x0	ECE: 外部时钟使能位 该位启用外部时钟模式 2 0: 禁止外部时钟模式 2; 1: 使能外部时钟模式 2。计数器由 ETRF 信号上的任意有效边沿驱动。 注 1: 设置 ECE 位与选择外部时钟模式 1 并将 TRGI 连到 ETRF(SMS=111 和 TS=111)具有相同功效。 注 2: 下述从模式可以与外部时钟模式 2 同时使用: 复位模式, 门控模式和触发模式; 但是, 这时 TRGI 不能连到 ETRF(TS 位不能是'111')。

ETPS				注 3：外部时钟模式 1 和外部时钟模式 2 同时被使能时，外部时钟的输入是 ETRF。
				ETPS: 外部触发预分频 外部触发信号 ETRP 的频率最多是 GPTIMA/B 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。
	13:12	R/W	0x0	
ETF				ETF: 外部触发滤波 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。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。 0000：无滤波器，以 $f_{\text{SAMPLING}}=f_{\text{DTS}}$ 采样 0001：采样频率 $f_{\text{SAMPLING}}=f_{\text{CK\_INT}}$ ，N=2 0010：采样频率 $f_{\text{SAMPLING}}=f_{\text{CK\_INT}}$ ，N=4 0011：采样频率 $f_{\text{SAMPLING}}=f_{\text{CK\_INT}}$ ，N=8 0100：采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/2$ ，N=6 0110：采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/4$ ，N=6 0111：采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/4$ ，N=8 1010：采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/2$ ，N=8 1000：采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/8$ ，N=6 1001：采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/8$ ，N=8 1010：采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/16$ ，N=5 1011：采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/16$ ，N=6 1100：采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/16$ ，N=8 1101：采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/32$ ，N=5 1110：采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/32$ ，N=6 1111：采样频率 $f_{\text{SAMPLING}}=f_{\text{DTS}}/32$ ，N=8 Note: Care must be taken that f_{DTS} is replaced in the formula by CK_INT when ETF[3:0] = 1, 2 or 3.
	11:8	R/W	0x0	
MSM				MSM: 主/从模式 0：无作用； 1：触发输入 (TRGI) 上的事件被延迟了，以允许在当前定时器 (通过 TRGO) 与它的从定时器间的完美同步。这对要求把几个定时器同步到一个单一的外部事件时是非常有用的。
	7	R/W	0x0	
				TS: 触发选择 这 3 位选择用于同步计数器的触发输入。 000：内部触发 0 (ITR0) 100：TI1 的边沿检测器 (TI1F_ED) 001：内部触发 1 (ITR1) 101：滤波后的定时器输入 1 (TI1FP1) 010：内部触发 2 (ITR2) 110：滤波后的定时器输入 2 (TI2FP2) 011：内部触发 3 (ITR3) 111：外部触发输入 (ETRF) 注：这些位只能在未用到 (如 SMS = 000) 时被改变，以避免在改变时产生错误的边沿检测。
	6:4	R/W	0x0	
Reserved	3	N/A	0x0	保留，保持复位值

SMS	2:0	R/W	0x0	SMS: 从模式选择 当选择了外部信号, 触发信号 (TRGI) 的有效边沿与选中的外部输入极性相关 (见输入控制寄存器和控制寄存器的说明) 000: 关闭从模式 - 如果 CEN=1, 则预分频器直接由内部时钟驱动。 001: 编码器模式 1 - 根据 TI1FP1 的电平, 计数器在 TI2FP2 的边沿向上/下计数。 010: 编码器模式 2 - 根据 TI2FP2 的电平, 计数器在 TI1FP1 的边沿向上/下计数。 011: 编码器模式 3 - 根据另一个信号的输入电平, 计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100: 复位模式 - 选中的触发输入 (TRGI) 的上升沿重新初始化计数器, 并且产生一个更新寄存器的信号。 101: 门控模式 - 当触发输入 (TRGI) 为高时, 计数器的时钟开启。一旦触发输入变为低, 则计数器停止 (但不复位)。计数器的启动和停止都是受控的。 110: 触发模式 - 计数器在触发输入 TRGI 的上升沿启动 (但不复位), 只有计数器的启动是受控的。 111: 外部时钟模式 1 - 选中的触发输入 (TRGI) 的上升沿驱动计数器。 注: 如果 TI1FEN 被选为触发输入 (TS = 100) 时, 不要使用门控模式。这是因为, TI1FED 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。

1.4.4 中断使能寄存器 (GPTIMA/B_IER)

		Address offset:			0x0C										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4O IE	CC3O IE	CC2O IE	CC1O IE	Reserved		TIE	Reser ved	CC4I E	CC3I E	CC2I E	CC1I E	UIE

Name	Bit	Type	Reset value	Description
Reserved	31:13	N/A	0x00	保留, 保持复位值
CC4OIE	12	WO	0x0	CC4OIE: 捕捉 / 比较 4 重复捕捉中断使能 0: 无效. 1: CC4 重复捕捉中断使能.
CC3OIE	11	WO	0x0	CC3OIE: 捕捉 / 比较 3 重复捕捉中断使能 0: 无 效 1: CC3 重复捕捉中断使能.
CC2OIE	10	WO	0x0	CC2OIE: 捕捉 / 比较 2 重复捕捉中断使能 0: 无 效 1: CC2 重复捕捉中断使能.
CC1OIE	9	WO	0x0	CC1OIE: 捕捉 / 比较 1 重复捕捉中断使能 0: 无 效 1: CC1 重复捕捉中断使能.
Reserved	8:7	N/A	0x0	保留, 保持复位值
TIE	6	WO	0x0	TIE: 触发中断使能 0: 无效. 1: 触发中断允许
Reserved	5	N/A	0x0	保留, 保持复位值

CC4IE	4	WO	0x0	CC4IE: 捕捉/比较 4 中断使能 0: 无效. 1: CC4 中断允许
CC3IE	3	WO	0x0	CC4IE: 捕捉/比较 3 中断使能 0: 无效. 1: CC3 中断允许
CC2IE	2	WO	0x0	CC4IE: 捕捉/比较 2 中断使能 0: 无效. 1: CC2 中断允许
CC1IE	1	WO	0x0	CC4IE: 捕捉/比较 1 中断使能 0: 无效. 1: CC1 中断允许
UIE	0	WO	0x0	UIE: 更新中断使能 0: 无效 1: 更新中断允许

1. 4. 5 中断禁止寄存器 (GPTIMA/B_IDR)

		Address offset:		0x10											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4O ID	CC3O ID	CC2O ID	CC1O ID	Reserved		TID	Reserved	CC4I D	CC3I D	CC2I D	CC1I D	UID

Name	Bit	Type	Reset value	Description
Reserved	31:13	N/A	0x00	保留, 保持复位值
CC4OID	12	WO	0x0	CC4OID: 捕捉 / 比较 4 重复捕捉中断禁止. 0: 无效. 1: CC4 重复捕捉中断禁止.
CC3OID	11	WO	0x0	CC3OID: 捕捉 / 比较 3 重复捕捉中断禁止. 0: 无效. 1: CC3 重复捕捉中断禁止.
CC2OID	10	WO	0x0	CC2OID: 捕捉 / 比较 2 重复捕捉中断禁止. 0: 无效. 1: CC2 重复捕捉中断禁止.
CC1OID	9	WO	0x0	CC1OID: 捕捉 / 比较 1 重复捕捉中断禁止. 0: 无效. 1: CC1 重复捕捉中断禁止.
Reserved	8:7	N/A	0x0	保留, 保持复位值
TID	6	RO	0x0	TID: 触发中断禁止. 0: 无效. 1: 触发中断禁止.
Reserved	5	N/A	0x0	保留, 保持复位值
CC4ID	4	WO	0x0	CC4ID: 捕捉 / 比较 4 中断禁止. 0: 无效. 1: CC4 中断禁止.
CC3ID	3	WO	0x0	CC3ID: 捕捉 / 比较 3 中断禁止. 0: 无效. 1: CC3 中断禁止.

CC2ID	2	WO	0x0	CC2ID: 捕捉 / 比较 2 中断禁止. 0: 无效果. 1: CC2 中断禁止.
CC1ID	1	WO	0x0	CC1ID: 捕捉 / 比较 1 中断禁止. 0: 无效果. 1: CC1 中断禁止.
UID	0	WO	0x0	UID: 更新中断禁止. 0: 无效果. 1: 更新中断禁止.

1. 4. 6 中断有效状态寄存器 (GPTIMA/B_IVS)

		Address offset:		0x14											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4OIS	CC3OIS	CC2OIS	CC1OIS	Reserved		TIS	Reserved	CC4IS	CC3IS	CC2IS	CC1IS	UIS

Name	Bit	Type	Reset value	Description
Reserved	31:15	N/A	0x00	保留, 保持复位值
CC4OIS	12	RO	0x0	CC4OIS: 捕捉/比较 4 重复捕捉中断有效状态 0: CC4 重复捕捉中断禁止状态. 1: CC4 重复捕捉中断使能状态. 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
CC3OIS	11	RO	0x0	CC3OIS: 捕捉 / 比较 3 重复捕捉中断有效状态 0: CC3 重复捕捉中断禁止状态. 1: CC3 重复捕捉中断使能状态. 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
CC2OIS	10	RO	0x0	CC2OIS: 捕捉 / 比较 2 重复捕捉中断有效状态 0: CC2 重复捕捉中断禁止状态. 1: CC2 重复捕捉中断使能状态. 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
CC1OIS	9	RO	0x0	CC1OIS: 捕捉 / 比较 1 重复捕捉中断有效状态 0: CC1 重复捕捉中断禁止状态. 1: CC1 重复捕捉中断使能状态. 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
Reserved	8:7	N/A	0x0	保留, 保持复位值
TIS	6	RO	0x0	TIS: 触发中断有效状态 0: 触发中断禁止状态. 1: 触发中断使能状态. 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
Reserved	5	N/A	0x0	保留, 保持复位值
CC4IS	4	RO	0x0	CC4IS: 捕捉 / 比较 4 中断有效状态 0: CC4 中断禁止状态. 1: CC4 中断使能状态. 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位

CC3IS	3	RO	0x0	CC3IS: 捕捉 / 比较 3 中断有效状态 0: CC3 中断禁止状态 1: CC3 中断使能状态 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
CC2IS	2	RO	0x0	CC2IS: 捕捉 / 比较 2 中断有效状态 0: CC2 中断禁止状态 1: CC2 中断使能状态 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
CC1IS	1	RO	0x0	CC1IS: 捕捉 / 比较 1 中断有效状态 0: CC1 中断禁止状态 1: CC1 中断使能状态 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
UIS	0	RO	0x0	UIS: 更新中断有效状态 0: 更新中断禁止状态 1: 更新中断使能状态 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位

1. 4. 7 原始中断标志寄存器 (GPTIMA/B_RIF)

		Address offset:			0x18										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4O F	CC3O F	CC2O F	CC1O F	Reserved		TIF	Reser ved	CC4IF	CC3IF	CC2IF	CC1IF	UIF

Name	Bit	Type	Reset value	Description
Reserved	31:13	N/A	0x00	保留, 保持复位值
CC4OF	12	RO	0x0	CC4OIF: 捕捉 / 比较 4 重复捕捉原始中断标志 参考 CC1OIF 描述
CC3OF	11	RO	0x0	CC3OIF: 捕捉 / 比较 3 重复捕捉原始中断标志 参考 CC1OIF 描述
CC2OF	10	RO	0x0	CC2OIF: 捕捉 / 比较 2 重复捕捉原始中断标志 参考 CC1OIF 描述
CC1OF	9	RO	0x0	CC1OIF: 捕捉 / 比较 1 重复捕捉原始中断标志 仅当相应的通道被配置为输入捕获时, 该标记可由硬件置 1。 GPTIM_ICR 寄存器相应位软件写“1”可清除该标志 0: 无重复中断产生 1: 当 CC1IF 的状态已经为‘1’, 计数器的值被捕获到 GPTIM_CCR1 寄存器。
Reserved	8:7	N/A	0x0	保留, 保持复位值
TIF	6	RO	0x0	TIF: 触发原始中断标志 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’。GPTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0: 无触发器事件产生; 1: 触发中断等待响应。
Reserved	5	N/A	0x0	保留, 保持复位值
CC4IF	4	RO	0x0	CC4IF: 捕捉 / 比较 4 原始中断标志 参考 CC1IF 描述

CC3IF	3	RO	0x0	CC3IF: 捕捉 / 比较 3 原始中断标志 参考 CC1IF 描述
CC2IF	2	RO	0x0	CC2IF: 捕捉 / 比较 2 原始中断标志 参考 CC1IF 描述
CC1IF	1	RO	0x0	CC1IF: 捕捉 / 比较 1 原始中断标志 如果通道 CC1 配置为输出模式: 当计数器值与比较值匹配时该位由硬件置 1, 但在中心对称模式下除外(参考 GPTIM_CR1 寄存器的 CMS 位)。它由软件清' 0'。 0: 无匹配发生; 1: GPTIM_CNT 的值与 GPTIM_CCR1 的值匹配。当 GPTIM_CCR1 的内容大于 GPTIM_APR 的内容时, 在向上或向上/下计数模式时计数器溢出, 或向下计数模式时的计数器下溢条件下, CC1IF 位变高 如果通道 CC1 配置为输入模式: 当捕获事件发生时该位由硬件置' 1', GPTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0: 无输入捕获产生; 1: 计数器值被捕获至 GPTIM_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
UIF	0	RO	0x0	UIF: 更新原始中断标志 当产生更新事件时该位由硬件置' 1'。它由软件清' 0'。 0: 无更新事件产生; 1: 更新中断等待响应。当寄存器被更新时该位由硬件置' 1': —若 GPTIM_CR1 寄存器的 UDIS=0, 当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 —若 GPTIM_CR1 寄存器的 URS=0、UDIS=0, 当设置 GPTIM_EGR 寄存器的 UG=1 时产生更新事件, 通过软件对计数器 CNT 重新初始化时。 —若 GPTIM_CR1 寄存器的 URS=0、UDIS=0, 当计数器 CNT 被触发事件重新初始化时。

1. 4. 8 屏蔽中断标志寄存器(GPTIMA/B_IFM)

		Address offset:		0x1C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4OIM	CC3OIM	CC2OIM	CC1OIM	Reserved		TIM	Reserved	CC4IM	CC3IM	CC2IM	CC1IM	UIM

Name	Bit	Type	Reset value	Description
Reserved	31:13	N/A	0x00	保留, 保持复位值
CC4OIM	12	RO	0x0	CC4OIM: 捕捉 / 比较 4 重复捕捉屏蔽中断标志 参考 CC1OIM 描述
CC3OIM	11	RO	0x0	CC3OIM: 捕捉 / 比较 3 重复捕捉屏蔽中断标志 参考 CC1OIM 描述
CC2OIM	10	RO	0x0	CC2OIM: 捕捉 / 比较 2 重复捕捉屏蔽中断标志 参考 CC1OIM 描述

CC1OIM	9	RO	0x0	CC1OIM: 捕捉 / 比较 1 重复捕捉屏蔽中断标志 仅当相应的通道被配置为输入捕获时, 该标记可由硬件置 1。 GPTIM_ICR 寄存器相应位软件写“1”可清除该标志 0: 无重复中断产生 1: 当 CC1IF 的状态已经为‘1’, 计数器的值被捕获到 GPTIM_CCR1 寄存器。
Reserved	8:7	N/A	0x0	保留, 保持复位值
TIM	6	RO	0x0	TIM: 触发屏蔽中断标志 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’。GPTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0: 无触发器事件产生; 1: 触发中断等待响应。
Reserved	5	N/A	0x0	保留, 保持复位值
CC4IM	4	RO	0x0	CC4IM: 捕捉 / 比较 4 屏蔽中断标志 参考 CC1IM 描述
CC3IM	3	RO	0x0	CC3IM: 捕捉 / 比较 3 屏蔽中断标志 参考 CC1IM 描述
CC2IM	2	RO	0x0	CC2IM: 捕捉 / 比较 2 屏蔽中断标志 参考 CC1IM 描述
CC1IM	1	RO	0x0	CC1IM : 捕捉 / 比较 1 屏蔽中断标志 如果通道 CC1 配置为输出模式: 当计数器值与比较值匹配时该位由硬件置 1, 但在中心对称模式下除外(参考 GPTIM_CR1 寄存器的 CMS 位)。它由软件清‘0’。 0: 无匹配发生; 1: GPTIM_CNT 的值与 GPTIM_CCR1 的值匹配。当 GPTIM_CCR1 的内容大于 GPTIM_APR 的内容时, 在向上或向上/下计数模式时计数器溢出, 或向下计数模式时的计数器下溢条件下, CC1IF 位变高。 如果通道 CC1 配置为输入模式: 当捕获事件发生时该位由硬件置‘1’, GPTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0: 无输入捕获产生; 1: 计数器值被捕获至 GPTIM_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
UIM	0	RO	0x0	UIM: 更新屏蔽中断标志 当产生更新事件时该位由硬件置‘1’。它由软件清‘0’。 0: 无更新事件产生; 1: 更新中断等待响应。当寄存器被更新时该位由硬件置‘1’: —若 GPTIM_CR1 寄存器的 UDIS=0, 当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 —若 GPTIM_CR1 寄存器的 URS=0、UDIS=0, 当设置 GPTIM_EGR 寄存器的 UG=1 时产生更新事件, 通过软件对计数器 CNT 重新初始化时。 —若 GPTIM_CR1 寄存器的 URS=0、UDIS=0, 当计数器 CNT 被触发事件重新初始化时。

1. 4. 9 中断清除寄存器(GPTIMA/B_ICR)

		Address offset:			0x20										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CC4OIC	CC3OIC	CC2OIC	CC1OIC	Reserved		TIC	Reserved	CC4IC	CC3IC	CC2IC	CC1IC	UIC

Name	Bit	Type	Reset value	Description
Reserved	31:13	N/A	0x00	保留, 保持复位值
CC4OIC	12	W1C	0x0	CC4OIC: 捕捉 / 比较 4 重复捕捉中断清除寄存器 参考 CC1OIC 描述
CC3OIC	11	W1C	0x0	CC3OIC: 捕捉 / 比较 3 重复捕捉中断清除寄存器 参考 CC1OIC 描述
CC2OIC	10	W1C	0x0	CC2OIC: 捕捉 / 比较 2 重复捕捉中断清除寄存器 参考 CC1OIC 描述
CC1OIC	9	W1C	0x0	CC1OIC: 捕捉 / 比较 1 重复捕捉中断清除寄存器 0: 无效 1: CC1OI 中断清除(GPTIM_RIF).
Reserved	8:7	N/A	0x0	保留, 保持复位值
TIC	6	W1C	0x0	TIC: 触发中断清除寄存器 0: 无效 1: 触发中断清除.
Reserved	5	N/A	0x0	保留, 保持复位值
CC4IC	4	W1C	0x0	CC4IC: 捕捉 / 比较 4 中断清除寄存器 参考 CC1IC 描述
CC3IC	3	W1C	0x0	CC3IC: 捕捉 / 比较 3 中断清除寄存器 参考 CC1IC 描述
CC2IC	2	W1C	0x0	CC2IC: 捕捉 / 比较 2 中断清除寄存器 参考 CC1IC 描述
CC1IC	1	W1C	0x0	CC1IC: 捕捉 / 比较 1 中断清除寄存器 0: 无效 1: 捕捉/比较 1 中断清除.
UIC	0	W1C	0x0	UIC: 更新中断清除寄存器 0: 无效 1: 更新中断清除.

1. 4. 10 事件产生寄存器(GPTIMA/B_EGR)

		Address offset:		0x24											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TG	Reserved	CC4G	CC3G	CC2G	CC1G	UG

Name	Bit	Type	Reset value	Description
Reserved	31:7	N/A	0x00	保留, 保持复位值
TG	6	WO	0x0	TG: 产生触发事件 该位由软件置'1', 用于产生一个触发事件, 由硬件自动清'

Reserved	5	N/A	0x0	0'。 0: 无动作; 1: GPTIM_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。
	4	WO	0x0	保留, 保持复位值
	3	WO	0x0	CC4G: 捕捉 / 比较 4 产生参考 CC1G 描述
	2	WO	0x0	CC3G: 捕捉 / 比较 3 产生参考 CC1G 描述
	1	WO	0x0	CC2G: 捕捉 / 比较 2 产生参考 CC1G 描述
CC1G	1	WO	0x0	CC1G: 捕捉 / 比较 1 产生该位由软件置'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 1 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 GPTIM_CCRI 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	WO	0x0	UG: 产生更新事件 该位由软件置'1', 由硬件自动清'0'。 0: 无动作; 1: 重新初始化计数器, 并产生一个(寄存器)更新事件。注意预分频器的计数器也被清'0' (但是预分频系数不变)。若在中心对称模式下或 DIR=0(向上计数)则计数器被清'0'; 若 DIR=1(向下计数)则计数器取 GPTIM_ARR 的值。

1. 4. 11 捕捉/比较模式寄存器 1 (GPTIMA/B_CCMR1)

		Address offset:		0x28											
		Reset value:		0x0000_0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC2 CE	OC2M[2:0]			OC2 PE	OC2 FE	CC2S[1:0]		OC1 CE	OC1M[2:0]			OC1 PE	OC1 FE	CC1S[1:0]	
IC2F[3:0]				IC2PSC[1:0]				IC1F[3:0]							

输出比较模式

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留, 保持复位值
OC2CE	15	R/W	0x0	OC2CE: 输出比较 2 清 0 使能 参考 OC1CE 描述
OC2M	14:12	R/W	0x0	OC2M: 输出比较模式 2 参考 OC1M 描述
OC2PE	11	R/W	0x0	OC2PE: 输出比较 2 预装使能 参考 OC1PE 描述

OC2FE	10	R/W	0x0	OC2FE: 输出比较2快速使能 参考 OC1FE 描述
CC2S	9:8	R/W	0x0	CC2S: 捕捉 / 比较 2 选择 该位定义通道的方向(输入/输出), 及输入信号的选择: 00: CC2 通道被配置为输出; 01: CC2 通道被配置为输入, IC2 映射在 TI2 上; 10: CC2 通道被配置为输入, IC2 映射在 TI1 上; 11: CC2 通道被配置为输入, IC2 映射在 TRC 上。此模式仅工作在内部触发器输入 被选中时(由 GPTIM_SMCR 寄存器的 TS 位选择)。 注: CC2S 仅在通道关闭时(GPTIM_CCER 寄存器的 CC2E=0)才是可写的。
OC1CE	7	R/W	0x0	OC1CE: 输出比较1清除使能 0: OC1REF 不受 ETRF 输入的影响; 1: 一旦检测到 ETRF 输入高电平, 清除 OC1REF=0。
OC1M	6:4	R/W	0x0	OC1M: 输出比较模式1 该 3 位定义了输出参考信号 OC1REF 的动作, 而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效, 而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000: 冻结。输出比较寄存器 GPTIM_CCR1 与计数器 GPTIM_CNT 间的比较对 OC1REF 不起作用; 001: 匹配时设置通道 1 为有效电平。当计数器 GPTIM_CNT 的值与捕获/比较寄存器 1(GPTIM_CCR1)相同时, 强制 OC1REF 为高。 010: 匹配时设置通道 1 为无效电平。当计数器 GPTIM_CNT 的值与捕获/比较寄存器 1(GPTIM_CCR1)相同时, 强制 OC1REF 为低。 011: 翻转。当 GPTIM_CCR1=GPTIM_CNT 时, 翻转 OC1REF 的电平。 100: 强制为无效电平。强制 OC1REF 为低。 101: 强制为有效电平。强制 OC1REF 为高。 110: PWM 模式 1 — 在向上计数时, 一旦 GPTIM_CNT<GPTIM_CCR1 时通道 1 为有效电平, 否则为无效电平; 在向下计数时, 一旦 GPTIM_CNT>GPTIM_CCR1 时通道 1 为无效电平(OC1REF=0), 否则为有效电平(OC1REF=1)。 111: PWM 模式 2 — 在向上计数时, 一旦 GPTIM_CNT<GPTIM_CCR1 时通道 1 为无效电平, 否则为有效电平; 在向下计数时, 一旦 GPTIM_CNT>GPTIM_CCR1 时通道 1 为有效电平, 否则为无效电平。 注 1: 一旦 LOCK 级别设为 3(GPTIM_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2: 在 PWM 模式 1 或 PWM 模式 2 中, 只有当比较结果改变了或在输出比较模式中从冻结模式切换到 PWM 模式时, OC1REF 电平才改变。 注 3: 在通道有互补输出时, 此时被预装载。如果设置了 GPTIM_CR2 中的 CCPC 位, OC1M 的有效位只能在发生 COM 事件时才能从预装载位得到新的值。
OC1PE	3	R/W	0x0	OC1PE: 输出比较1预装使能 0: 禁止 GPTIM_CCR1 寄存器的预装载功能, 可随时写入 GPTIM_CCR1 寄存器, 并且新写入的数值立即起作用。 1: 开启 GPTIM_CCR1 寄存器的预装载功能, 读写操作仅对预装载寄存器操作, GPTIM_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1: 一旦 LOCK 级别设为 3(GPTIM_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2: 仅在单脉冲模式下(GPTIM_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
OC1FE	2	R/W	0x0	OC1FE: 输出比较1快速使能 该位用于加快 CC 输出对触发输入事件的响应。 0: CC1 的正常操作依赖于计数器与 CCR1 的值, 即使工作于触发器状态。当触发器的输入有一个有效沿时, 激活 CC1 输出的最小

CC1S	1:0	R/W	0x0	延时为 5 个时钟周期。 1: 输入到触发器的有效沿的作用就象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。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
				CC1S: 捕捉 / 比较 1 选择 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00: CC1 通道被配置为输出； 01: CC1 通道被配置为输入，IC1 映射在 TI1 上； 10: CC1 通道被配置为输入，IC1 映射在 TI2 上； 11: CC1 通道被配置为输入，IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 GPTIM_SMC 寄存器的 TS 位选择)。 注：CC1S 仅在通道关闭时(GPTIM_CCER 寄存器的 CC1E=0)才是可写的。

输入捕捉模式

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
IC2F	15:12	R/W	0x0	IC2F: 输入捕捉 2 滤波器 参考 IC1F 描述
IC2PSC	11:10	R/W	0x0	IC2PSC: 输入捕捉 2 预分频器 参考 IC1PSC 描述
CC2S	9:8	R/W	0x0	CC2S: 捕捉 / 比较 2 选择 这 2 位定义通道的方向(输入/输出)，及输入脚的选择： 00: CC2 通道被配置为输出； 01: CC2 通道被配置为输入，IC2 映射在 TI2 上； 10: CC2 通道被配置为输入，IC2 映射在 TI1 上； 11: CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 GPTIM_SMC 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(GPTIM_CCER 寄存器的 CC2E=0)才是可写的。
IC1F	7:4	R/W	0x0	IC1F: 输入捕捉 1 滤波器 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成，它记录到 N 个事件后会产生一个输出的跳变： 0000: 没有过滤器，$f_{\text{SAMPLING}} = f_{\text{DTS}}$ 0001: $f_{\text{SAMPLING}} = f_{\text{CK\_INT}}, N = 2$ 0010: $f_{\text{SAMPLING}} = f_{\text{CK\_INT}}, N = 4$ 0011: $f_{\text{SAMPLING}} = f_{\text{CK\_INT}}, N = 8$ 0100: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 2, N = 6$ 0101: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 2, N = 8$ 0110: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 4, N = 6$ 0111: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 4, N = 8$ 1000: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 8, N = 6$ 1001: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 8, N = 8$ 1010: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 16, N = 5$ 1011: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 16, N = 6$ 1100: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 16, N = 8$ 1101: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 32, N = 5$ 1110: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 32, N = 6$ 1111: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 32, N = 8$ 注意: 当 ICxF[3:0] = 1、2 或 3 时，必须注意公式中 fDTS 用 CK_INT 代替。

IC1PSC	3:2	R/W	0x0	IC1PSC: 输入捕捉 1 预分频器 这 2 位定义了 CC1 输入 (IC1) 的预分频系数。一旦 CC1E=0(GPTIM_CCER 寄存器中), 则预分频器复位。 00: 无预分频器, 捕获输入口上检测到的每一个边沿都触发一次捕获; 01: 每 2 个事件触发一次捕获; 10: 每 4 个事件触发一次捕获; 11: 每 8 个事件触发一次捕获。
				CC1S: 捕捉/比较 1 选择 这 2 位定义通道的方向(输入/输出), 及输入脚的选择: 00: CC1 通道被配置为输出; 01: CC1 通道被配置为输入, IC1 映射在 TI1 上; 10: CC1 通道被配置为输入, IC1 映射在 TI2 上; 11: CC1 通道被配置为输入, IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 GPTIM_SMCR 寄存器的 TS 位选择)。 注: CC1S 仅在通道关闭时(GPTIM_CCER 寄存器的 CC1E=0)才是可写的。

1. 4. 12 捕捉/比较模式寄存器 2 (GPTIMA/B_CCMR2)

		Address offset:			0x2C										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OC4 CE	OC4M[2:0]			OC4 PE	OC4 FE	CC2S[1:0]		OC3 CE	OC3M[2:0]			OC3 PE	OC3 FE	CC3S[1:0]	
IC4F[3:0]				IC4PSC[1:0]		IC3F[3:0]					IC3PSC[1:0]				

输出比较模式

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留, 保持复位值
OC4CE	15	R/W	0x0	OC4CE: 输出比较 4 清除 0 使能 参考 OC1CE 描述
OC4M	14:12	R/W	0x0	OC4M: 输出比较 4 模式 参考 OC1M 描述
OC4PE	11	R/W	0x0	OC4PE: 输出比较 4 预分频使能 参考 OC1PE 描述
OC4FE	10	R/W	0x0	OC4FE: 输出比较 4 快速使能 参考 OC1FE 描述
CC4S	9:8	R/W	0x0	CC4S: 捕捉 / 比较 4 选择 该 2 位定义通道的方向(输入/输出), 及输入脚的选择: 00: CC4 通道被配置为输出; 01: CC4 通道被配置为输入, IC4 映射在 TI4 上; 10: CC4 通道被配置为输入, IC4 映射在 TI3 上; 11: CC4 通道被配置为输入, IC4 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 GPTIM_SMCR 寄存器的 TS 位选择)。 注: CC4S 仅在通道关闭时(GPTIM_CCER 寄存器的 CC4E=0)才是可写的。
OC3CE	7	R/W	0x0	OC3CE: 输出比较 3 清除 0 使能 参考 OC1CE 描述

OC3M	6:4	R/W	0x0	OC3M: 输出比较3模式 参考 OC1M 描述
OC3PE	3	R/W	0x0	OC3PE: 输出比较3预分频使能 参考 OC1PE 描述
OC3FE	2	R/W	0x0	OC3FE: 输出比较3快速使能 参考 OC1FE 描述
CC3S	1:0	R/W	0x0	CC3S: 捕捉/比较3选择 该2位定义通道的方向(输入/输出), 及输入脚的选择: 00: CC3 通道被配置为输出; 01: CC3 通道被配置为输入, IC4 映射在 TI3 上; 10: CC3 通道被配置为输入, IC4 映射在 TI4 上; 11: CC3 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 GPTIM_SMCR 寄存器的 TS 位选择)。 注: CC3S 仅在通道关闭时(GPTIM_CCER 寄存器的 CC3E=0)才是可写的。

输入捕捉模式

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留, 保持复位值
IC4F	15:12	R/W	0x0	IC4F: 输入捕捉4滤波器 参考 IC1F 描述
IC4PSC	11:10	R/W	0x0	IC4PSC: 输入捕捉4预分频器 参考 IC1PSC 描述
CC4S	9:8	R/W	0x0	CC4S: 捕捉/比较4选择 这2位定义通道的方向(输入/输出), 及输入脚的选择: 00: CC4 通道被配置为输出; 01: CC4 通道被配置为输入, IC3 映射在 TI4 上; 10: CC4 通道被配置为输入, IC3 映射在 TI3 上; 11: CC4 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 GPTIM_SMCR 寄存器的 TS 位选择)。 注: CC4S 仅在通道关闭时(GPTIM_CCER 寄存器的 CC4E=0)才是可写的。
IC3F	7:4	R/W	0x0	IC3F: 输入捕捉3滤波器 参考 IC1F 描述
IC3PSC	3:2	R/W	0x0	IC3PSC: 输入捕捉3预分频器 参考 IC1PSC 描述
CC3S	1:0	R/W	0x0	CC3S: 捕捉/比较3选择 这2位定义通道的方向(输入/输出), 及输入脚的选择: 00: CC3 通道被配置为输出; 01: CC3 通道被配置为输入, IC3 映射在 TI3 上; 10: CC3 通道被配置为输入, IC3 映射在 TI4 上; 11: CC3 通道被配置为输入, IC3 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 GPTIM_SMCR 寄存器的 TS 位选择)。 注: CC3S 仅在通道关闭时(GPTIM_CCER 寄存器的 CC3E=0)才是可写的。

1. 4. 13 捕捉/比较使能寄存器(GPTIMA/B_CCER)

Address offset:		0x30													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC4NP	Reserved	CC4P	CC4E	CC3NP	Reserved	CC3P	CC3E	CC2NP	Reserved	CC2P	CC2E	CC1NP	Reserved	CC1P	CC1E

Name	Bit	Type	Reset value	Description
Reserved	31:14	N/A	0x00	保留，保持复位值
CC4P	13	R/W	0x0	CC4P: 捕捉 / 比较 4 输出极性 参考 CC1P 描述
CC4E	12	R/W	0x0	CC4E: 捕捉 / 比较 4 输出使能 参考 CC1E 描述
Reserved	11:10	N/A	0x0	保留，保持复位值
CC3P	9	R/W	0x0	CC3P: 捕捉 / 比较 3 输出极性 参考 CC1P 描述
CC3E	8	R/W	0x0	CC3E: 捕捉 / 比较 3 输出使能 参考 CC1E 描述
Reserved	7:6	N/A	0x0	保留，保持复位值
CC2P	5	R/W	0x0	CC2P: 捕捉 / 比较 2 输出极性 参考 CC1P 描述
CC2E	4	R/W	0x0	CC2E: 捕捉 / 比较 2 输出使能 参考 CC1E 描述
Reserved	3:2	N/A	0x0	保留，保持复位值
CC1P	1	R/W	0x0	CC1P: 捕捉 / 比较 1 输出极性 CC1 通道配置为输出: 0: OC1 高电平有效; 1: OC1 低电平有效。 CC1 通道配置为输入: CC1NP/CC1P 位选择在触发或捕捉模式下 TI1FP1 和 TI2FP1 的有效极性。 00: 非反相/上升沿 电路作用于 TIxFP1 的上升沿(在复位、外部时钟或触发模式下的捕捉或触发操作), TIxFP1 非反相(在门控模式或编码模式)。 01: 反相/下降沿电路作用于 TIxFP1 的下降沿(在复位、外部时钟或触发模式下的捕捉或触发操作), TIxFP1 反相(在门控模式或编码模式)。 00: 保留不用 11: 非反相/上升或下降沿电路作用于 TIxFP1 的上升沿和下降沿(在复位、外部时钟或触发模式下的捕捉或触发操作), TIxFP1 非反相(在门控模式)。在编码模式下不能使用此配置。 注: 在通道有互补输出时, 本位被预装载。如果寄存器 GPTIM_CR2 的 CCPC 被置 1, 则 CC1NP 的有效位从预装载复制的新值只有在通信事件发生时才起作用。 注: 一旦 LOCK 级别(GPTIM_BDTR 寄存器中的 LOCK 位)设为 3 或 2, 则该位不能被修改。

CC1E	0	R/W	0x0	CC1E: 捕 捉 / 比 较 1 输 出 使 能 CC1 通 道 配 置 为 输 出 : 0: 关闭— OC1N 禁止输出, 因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 1: 开启— OC1N 信号输出到对应的输出引脚, 其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 CC1 通 道 配 置 为 输 入 : 本位用于决定是否一个定时器值的捕捉要装载到捕捉/比较寄存器 1(GPTIM_CCR1)。 0: 捕捉禁止。 1: 捕捉允许。 注: 在通道有互补输出时, 本位被预装载。如果寄存器 GPTIM_CR2 的 CCPC 被置 1, 则 CC1NP 的有效位从预装载复制的新值只有在通信事件发生时才起作用。

1. 4. 14 计数寄存器(GPTIMA/B_CNT)

		Address offset:			0x34										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT[31:16] (Only GPTIMA)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															

Name	Bit	Type	Reset value	Description
CNT[31:16]	31:16	N/A	0x0	CNT[31:16]: 高 16 位值 (仅 GPTIMA1).
CNT[15:0]	15:0	R/W	0x0	CNT[15:0]: 低 16 位值

1. 4. 15 预分频寄存器(GPTIMA/B_PSC)

		Address offset:		0x38											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留, 保持复位值
PSC	15:0	R/W	0x0	PSC: 预 分 频 值 预分频器的值(Prescaler value)计数器的时钟频率(CK_CNT)等于 $f_{CK\_PSC}/(PSC[15:0]+1)$。每次当更新事件产生时, PSC 的值被装入当前预分频器寄存器; 更新事件包括计数器被 TIM_EGR 的 UG 位清'0'或被工作在复位模式的从控制器清'0'。

1. 4. 16 自动重载寄存器(GPTIMA/B_ARR)

		Address offset:			0x3C										
		Reset value:			0xFFFF FFFF(Only GPTIMA) or 0x0000 FFFF										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT[31:16] (Only GPTIMA)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															

Name	Bit	Type	Reset value	Description
ARR[31:16]	31:16	R/W	0xFFFF	ARR[31:16]: 自动重载高 16 位值 (仅 GPTIMA1).
ARR[15:0]	15:0	R/W	0xFFFF	ARR[15:0]: 自动重载值 ARR 包含了将要装载入实际的自动重载寄存器的值。 当自动重载的值为空时, 计数器不工作。

1. 4. 17 捕捉/比较寄存器 1 (GPTIMA/B_CCR1)

		Address offset:			0x44										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR1[31:16] (Only GPTIMA)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															

Name	Bit	Type	Reset value	Description
CCR1[31:16]	31:16	R/W	0x0	CCR1[31:16]: 捕捉/比较通道 1 高 16 位值 (仅 GPTIMA1).
CCR1[15:0]	15:0	R/W	0x0	CCR1[15:0]: 捕捉 / 比较通道 1 低 16 位值 若 CC1 通道配置为输出: CCR1 决定了装入当前捕捉/比较 1 寄存器的值(预装载值)。 如果在 GPTIM_CCMR1 寄存器(OC1PE 位)中未选择预装载功能, 写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时, 此预装载值才传输至当前捕捉/比较 1 寄存器中。当前捕捉/比较寄存器参与同计数器 GPTIM_CNT 的比较, 并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入: CCR1 包含了由上一次输入捕捉 1 事件(IC1)传输的计数器值。

1. 4. 18 捕捉/比较寄存器 2 (GPTIMA/B_CCR2)

		Address offset:			0x48										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

CCR2[31:16] (Only GPTIMA)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															

Name	Bit	Type	Reset value	Description
CCR2[31:16]	31:16	R/W	0x0	CCR2[31:16]: 捕捉/比较通道 2 高 16 位值 (仅 GPTIMA1).
CCR2[15:0]	15:0	R/W	0x0	CCR2[15:0]: 捕捉 / 比较通道 2 低 16 位值 参考 CCR1 描述

1. 4. 19 捕捉/比较寄存器 3 (GPTIMA/B_CCR3)

		Address offset:		0x4C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR3[31:16] (Only GPTIMA)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR3[15:0]															

Name	Bit	Type	Reset value	Description
CCR3[31:16]	31:16	R/W	0x0	CCR3[31:16]: 捕捉/比较通道 3 高 16 位值 (仅 GPTIMA1).
CCR3[15:0]	15:0	R/W	0x0	CCR3[15:0]: 捕捉 / 比较通道 3 低 16 位值 参考 CCR1 描述

1. 4. 20 捕捉/比较寄存器 4 (GPTIMA/B_CCR4)

		Address offset:		0x50											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CCR4[31:16] (Only GPTIMA)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR4[15:0]															

Name	Bit	Type	Reset value	Description
CCR4[31:16]	31:16	R/W	0x0	CCR4[31:16]: 捕捉/比较通道 4 高 16 位值 (仅 GPTIMA1).
CCR4[15:0]	15:0	R/W	0x0	CCR4[15:0]: 捕捉 / 比较通道 4 低 16 位值 参考 CCR1 描述

1. 4. 21 DMA 事件使能寄存器(GPTIMA/B_DMAEN)

		Address offset:		0x58											
		Reset value:		0x0000 0000											

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TDE	Reserved	CC4DE	CC3DE	CC2DE	CC1DE	UDE

Name	Bit	Type	Reset value	Description
Reserved	31:7	N/A	0x00	保留，保持复位值
TDE	6	R/W	0x0	TDE: 触发中断 DMA 请求使能 0: 触发中断 DMA 请求禁止. 1: 触发中断 DMA 请求使能.
Reserved	5	N/A	0x0	保留，保持复位值
CC4DE	4	R/W	0x0	CC4DE: 捕捉 / 比较 4 中断 DMA 请求使能 0: CC4 中断 DMA 请求禁止. 1: CC4 中断 DMA 请求使能.
CC3DE	3	R/W	0x0	CC3DE: 捕捉 / 比较 3 中断 DMA 请求使能 0: CC3 中断 DMA 请求禁止. 1: CC3 中断 DMA 请求使能.
CC2DE	2	R/W	0x0	CC2DE: 捕捉 / 比较 2 中断 DMA 请求使能 0: CC2 中断 DMA 请求禁止. 1: CC2 中断 DMA 请求使能.
CC1DE	1	R/W	0x0	CC1DE: 捕捉 / 比较 1 中断 DMA 请求使能 0: CC1 中断 DMA 请求禁止. 1: CC1 中断 DMA 请求使能.
UDE	0	R/W	0x0	UDE: 更新中断 DMA 请求使能 0: 更新中断 DMA 请求禁止. 1: 更新中断 DMA 请求使能.

第13章 通用定时器（GPTIMC）

1.1 GPTIMC 简介

通用定时器(GPTIMC)由一可编程预除器驱动之 16 位自动重载计数器构成。

其亦可用于多种用途，包含测量输入信号脉冲长度(输入捕获)或产生输出波形(输出比较,PWM, 互补 PWM 含 dead-time 插入)

可使用定时预除器与 APB 时钟控制预除器来调校脉冲长度与波形周期，由数微秒至数毫秒。

高级控制(ADTIM1),通用(GPTIMA1,GPTIMB1,GPTIMC1)与基本(BSTIM1)定时器皆为完全独立不共享任何时钟源

1.2 GPTIMC 主要特性

- ◆ 16 位上数自动加载计数器
- ◆ 16 位可编程预除器其计数器时钟频率允许被 1 与 65536 间之任何因子除(于运行中)
- ◆ 高达两个独立信道
 - 输入捕获
 - 输出比较
 - PWM 产生(边缘与中央对齐模式)
 - 单触发模式输出
- ◆ 互补输出可编程 dead-time 寄存器
- ◆ 同步化电路可控制定时器含外部信号与链接数个定时器
- ◆ 重复计数器用于更新定时寄存器，仅于计数器周期之已定义编号后可作用
- ◆ 暂停输入以将定时器输出信号置于重置状态或是已知状态
- ◆ 中断产生以下事件/DMA:
 - 更新:计数器上溢/下溢，计数器初始化(透过软件或内部/外部触发)
 - 触发事件(计数器起始，停止，初始化或由内部/外部触发计数)
 - 输入捕获(捕获寄存器)
 - 输出比较(计数寄存器配对比较寄存器)
 - 刹车信号输入
- ◆ 外部时钟触发输入或 cycle-by-cycle 电流管理

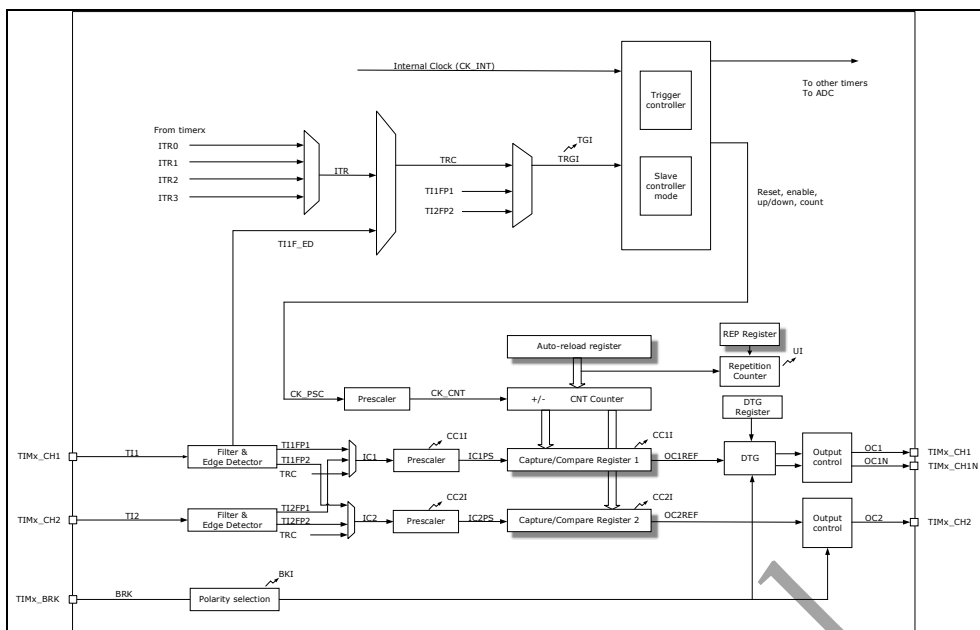


图 13-1 通用定时器块图

1.3 GPTIMC 功能描述

1.3.1 定时器时基单位

编程通用定时器主区块为一带自动加载寄存器之 16 位计数器。计数器可递增计算。计数时钟可被预除器区分。

计数器、自动重载寄存器与预除寄存器可由软件读写。此为计数器运作时之真正对等。

时基单位包含:

- ◆ 计数寄存器(GPTIMC_CNT)
- ◆ 预除寄存器(GPTIMC_PSC)
- ◆ 自动重载寄存器(GPTIMC_ARR)
- ◆ 重复计数寄存器(GPTIMC_RCR)

自动重载寄存器已被预载。由自动重载寄存器写入或读出以存取预载寄存器。预载寄存器的内容皆被永久传输至影子寄存器或是每个更新事件(UEV)，取决于 GPTIMC_CR1 寄存器之自动预载使能位(ARPE)。当计数器到达上溢(或向下计数时的下溢)以及若 UDIS 位同等于 GPTIMC_CR1 寄存器的 0，更新事件即被送出。更新事件其亦可由软件产生。更新事件的产生在各自设置中被描述。

计数器是由预除器输出 CK_CNT 计时，仅当 GPTIMC_CR1 寄存器之计数器使能位(CEN)被设置时被使能(参照属模式控制器描述以取得更多计数器使能信息)。

注:实际计数器使能信号 CNT_EN 被设置为 GPTIMC_CR1 寄存器 CEN 位后的 1 个时钟周期

预除器描述

预除器可将计数器时钟频率依照 1 至 65536 间之任意因子分频。基于一 16 位寄存器控制之 16 位计数器(GPTIMC_PSC 寄存器内)。因其具有缓冲功能，可于运行时被变更。新的预除器参数于下一次更新事件到达前被采用。预除计数器仅能向下计数。预除器被设定后，一旦预除计数器到达 0，计数器则会向上或向下计数，此外计数器会随着 APB 时钟一同更新。图 2 与图 3 标示当预除器于运作时变更计数器之行为范例。

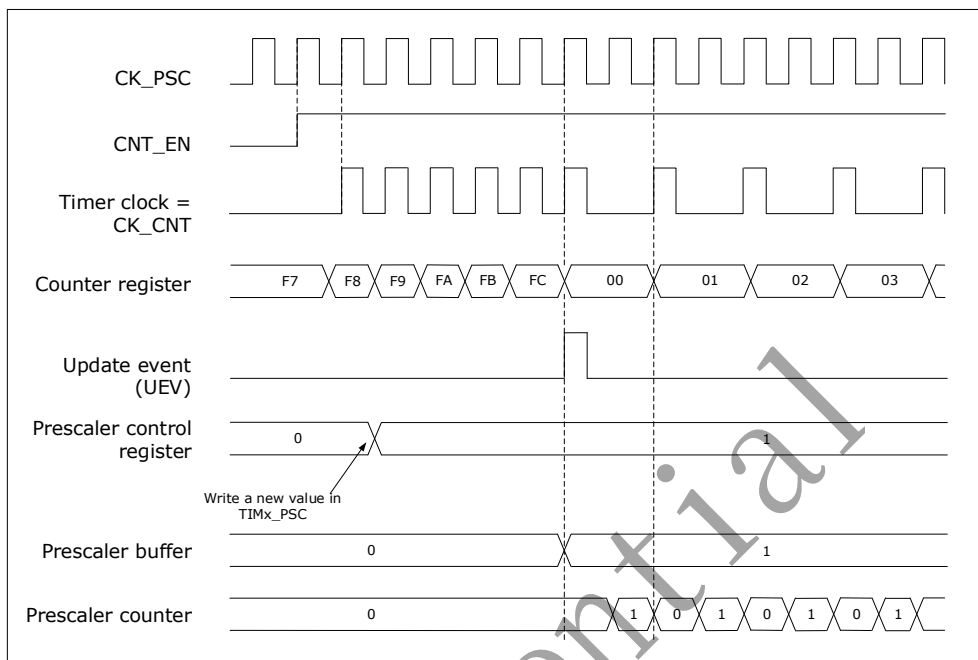


图 13-2 预除器区隔由 1 变更为 2 之计数器时序图

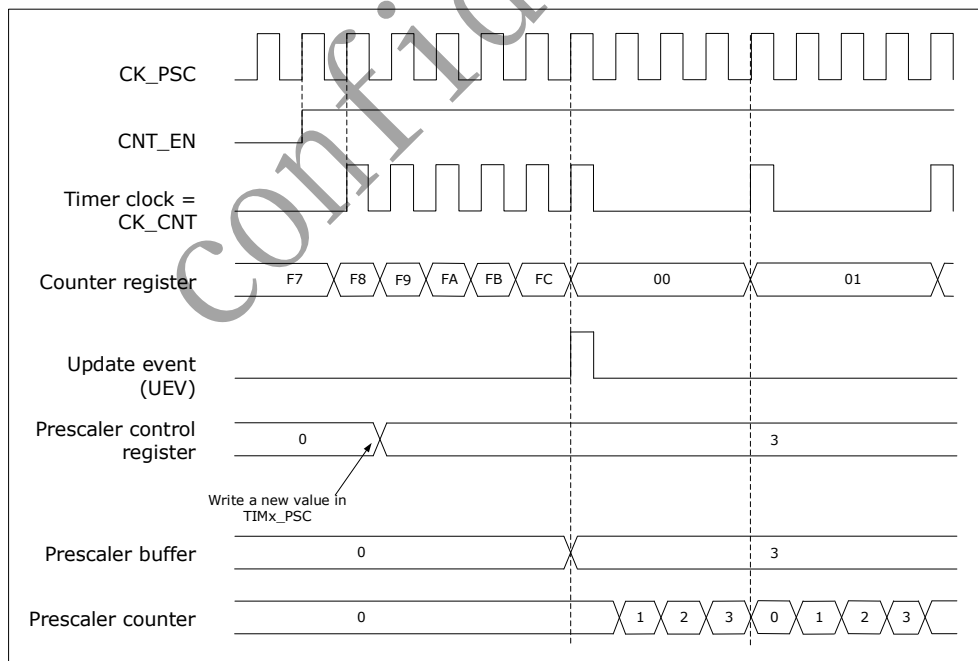


图 13-3 预除器区隔由 1 变更为 4 之计数器时序图

1.3.2 计数器模式

递增计数模式

在递增计数模式，计数器由 0 计算至自动加载值(GPTIMC_ARR 寄存器内容)由 0 重新启动并产生一计数器上溢事件。

若重复计数器被使用，更新事件(UEV)会在向上计数依重复计数器(GPTIMC_RCR)编程次数重复后产生。此外更新事件会于每一计数器上溢产生。

更新事件可于每一计数器下溢或设置 GPTIMC_EGR 寄存器 UG 位产生(透过软件或从模式控制器)。

UEV 事件可由软件透过设置 GPTIMC_CR1 寄存器之 UDIS 位以关闭。这是为了避免更新写入新值至预载寄存器时，影子寄存器也一并更新。接着直到 UDIS 位被写 0 皆无更新事件产生。但计数器会由 0 重启，预除器之计数器亦同(但除频率不会变更)。

当更新事件发生时，所有寄存器皆被更新且更新标志(GPTIMC_RIF 寄存器 UIF 位)皆被设定：

- ◆ 重复计数器随 GPTIMC_RCR 寄存器内容重新加载
- ◆ 自动重载影子寄存器被更新为预载值(GPTIMC_ARR)
- ◆ 预除器之缓冲器随着预载值重新加载(GPTIMC_PSC 寄存器之内容)

下列图表为计数器在 GPTIMC_ARR=0x36 时对应不同时钟频点之行为范例：

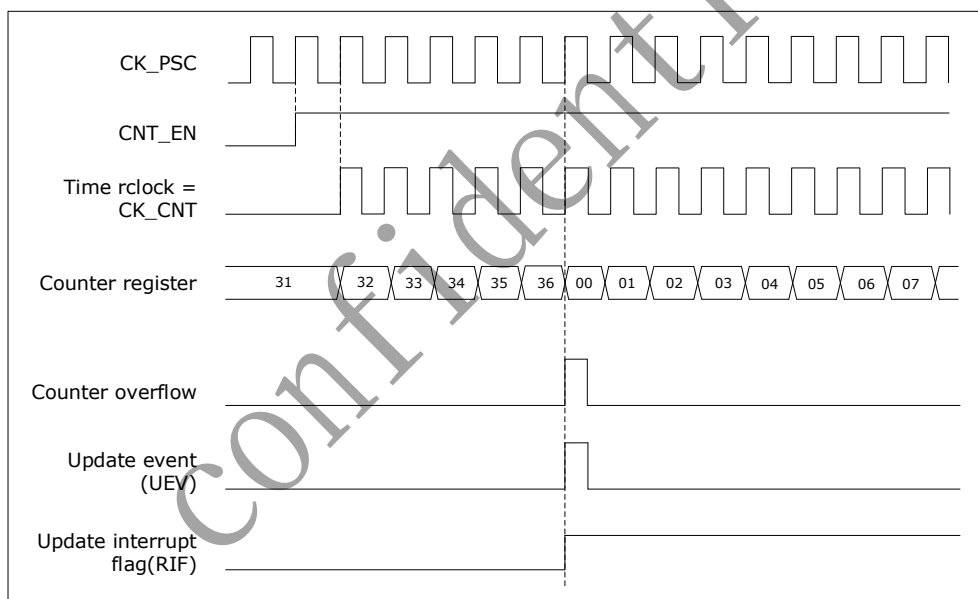


图 13-4 计数器时序图，内部时钟除以 1

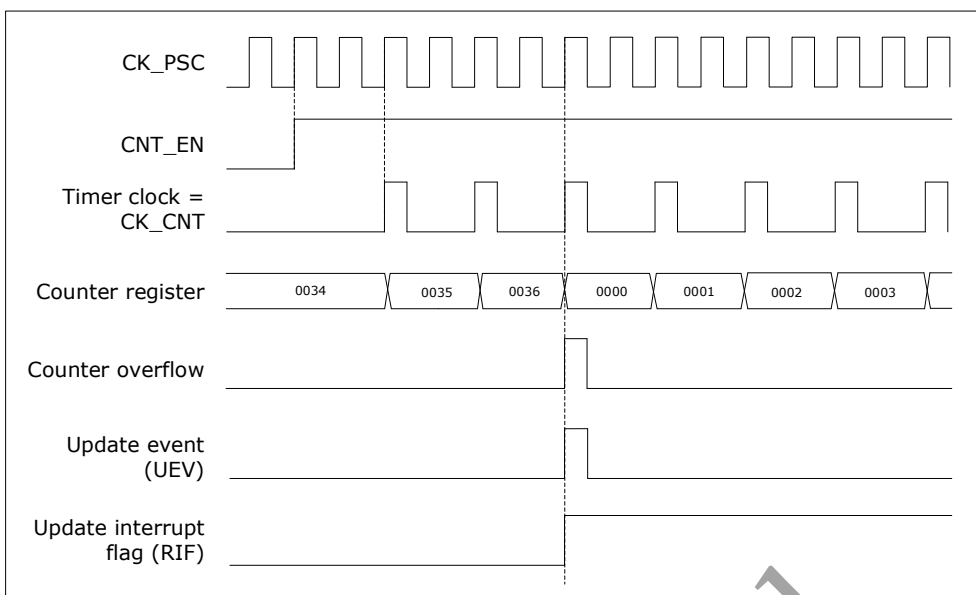


图 13-5 计数器时序图，内部时钟除以 2

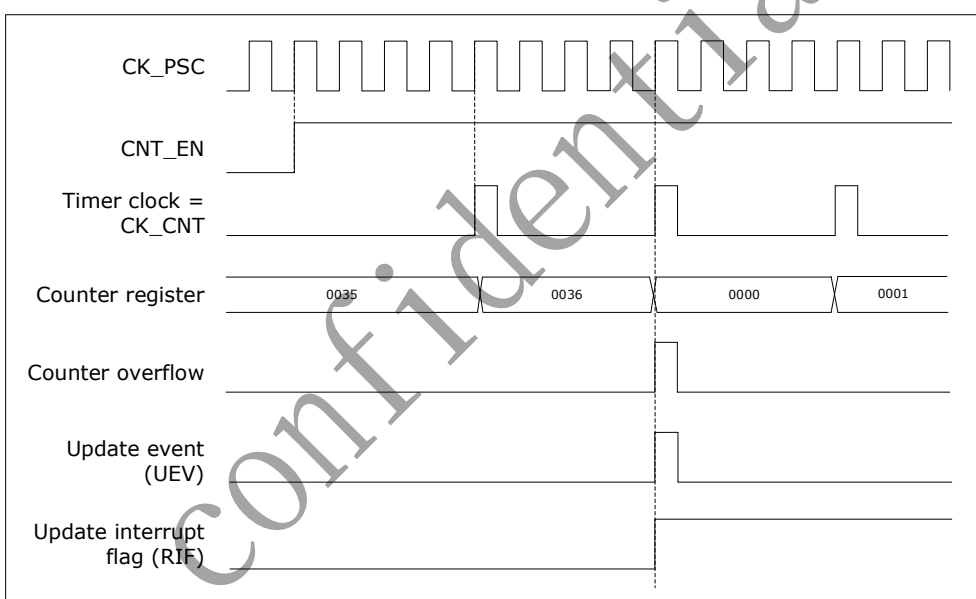


图 13-6 计数器时序图，内部时钟除以 4

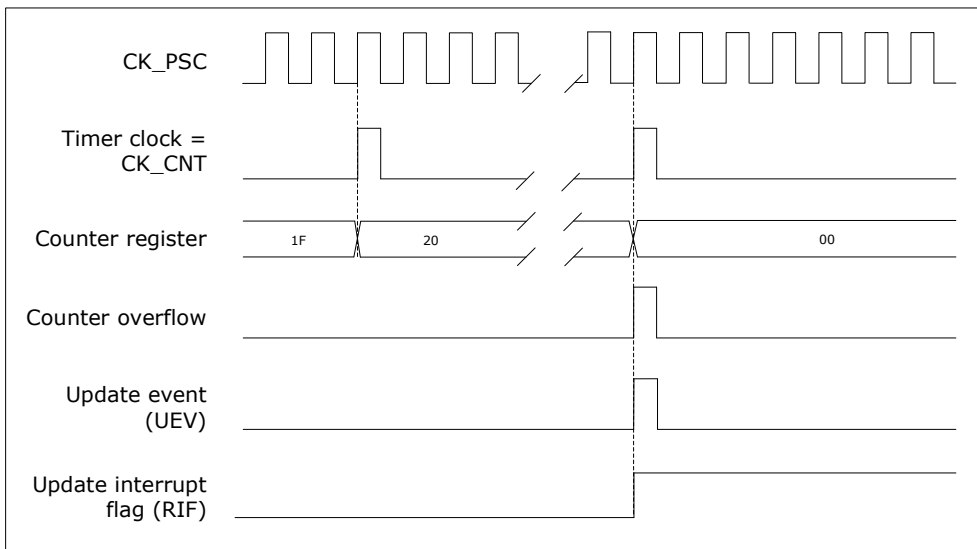


图 13-7 计数器时序图，内部时钟除以 N

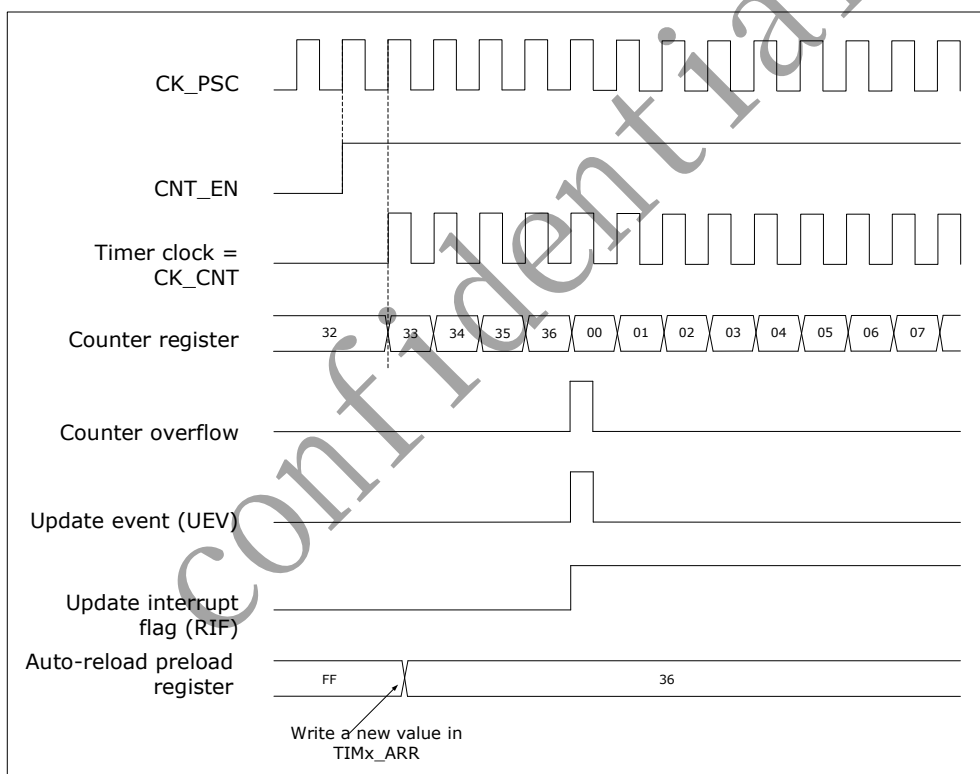


图 13-8 计数器时序图，当 ARPE=0 更新事件 (GPTIMC_ARR 无预载)

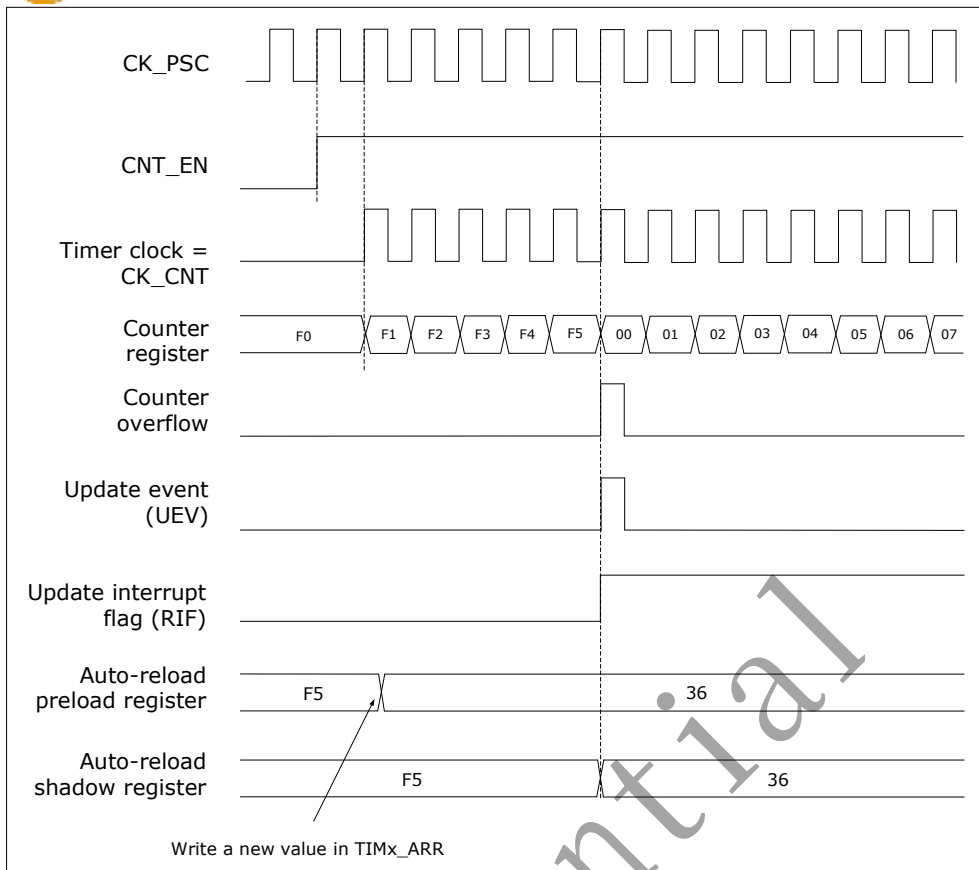


图 13-9 计数器时序图，当 ARPE=1 更新事件 (GPTIMC_ARR 预载)

1.3.3 重复计数器

实际仅于当重复计数器到达零时产生。可被用于产生 PWM 信号。时基单位描述更新事件 (UEV) 如何透过探访计数器上溢/下溢被产生。

这代表数据皆由预载寄存器传输至影子寄存器 (GPTIMC_ARR 预载寄存器，GPTIMC_PSC 预除寄存器，但于比较模式下亦为 GPTIMC_CCRx 捕捉/比较寄存器) 每一 N 计数器上溢或下溢，此时 N 为 GPTIMC_RCR 重复计数寄存器之值。

重复计数器会因下列原因减少：

- ◆ 于每次计数器向上计数模式之上溢

重复计数器为自动加载形式，重复比率维持为 GPTIMC_RCR 寄存器定义之值 (参考图 1-8)，当软件产生更新事件时 (设定 GPTIMC_EGR 寄存器 UG 位) 或是由硬件透过属模式控制器产生，不论重复计数器值以及重新加载寄存器之内容为何，都会突然发生。

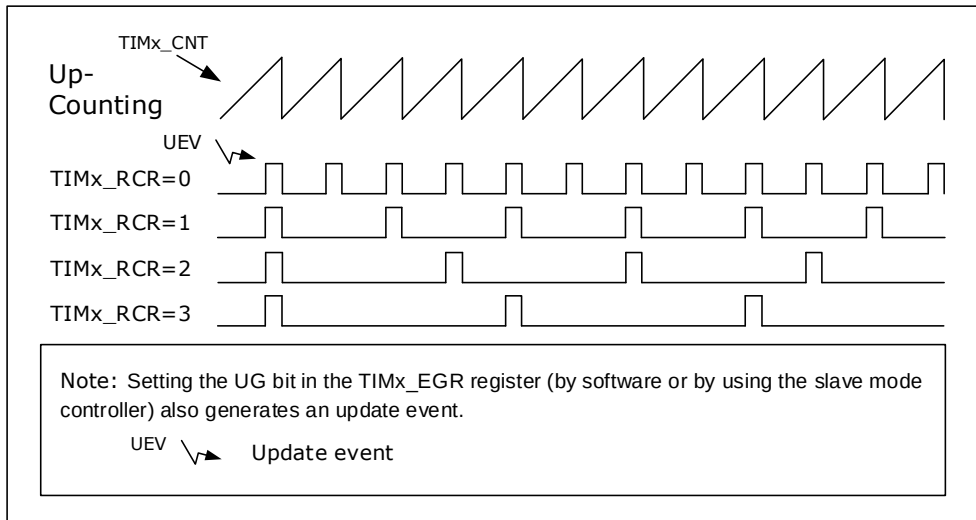


图 13-10 以模式与 GPTIMC_RCR 寄存器设置为依据之更新比率范例

1. 3. 4 时钟源

计数器时钟可由下列时钟源提供:

- ◆ 内部时钟(CK_INT)
- ◆ 外部时钟模式 1:外部输入引脚
- ◆ 内部触发输入(ITRx):使用一定时器作为另一个定时器等预除器, 例:可定时器 1 配置为定时器 2 的预除器。

1. 3. 4. 1 内部时钟源(CK_INT)

若属模式控制器被关闭(GPTIMC_SMCR 寄存器内, SMS=000), CEN,DIR 位(GPTIMC_CR1 寄存器)与 UG 位(GPTIMC_EGR 寄存器)皆为实际控制位且仅可被软件变更(UG 位仍被自动清除)只要 CEN 位被写为'1', 预除器之时钟即由内部时钟 CK_INT 提供。

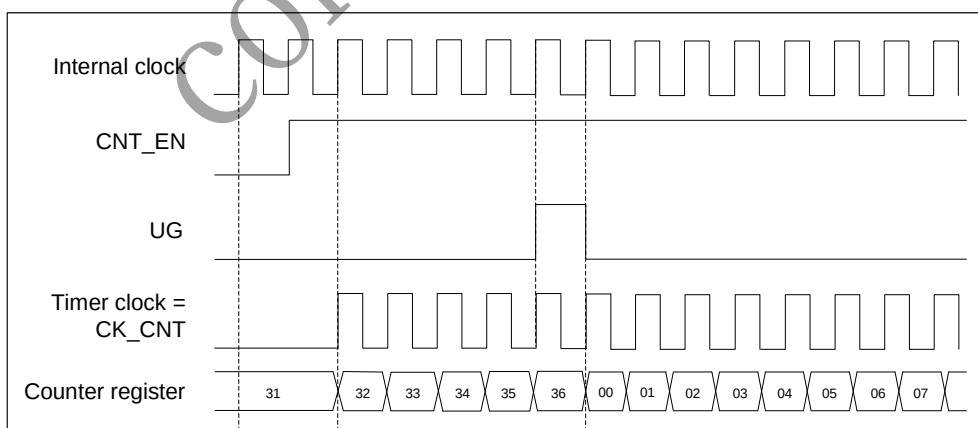


图 13-11 一般模式下之控制电路, 内部时钟分频因子为 1

1. 3. 4. 2 外部时钟源模式 1

当 GPTIMC_SMCR 寄存器之 SMS=111 时此模式会被选定。计数器于被选定输入端之每一上升或下降缘计数。

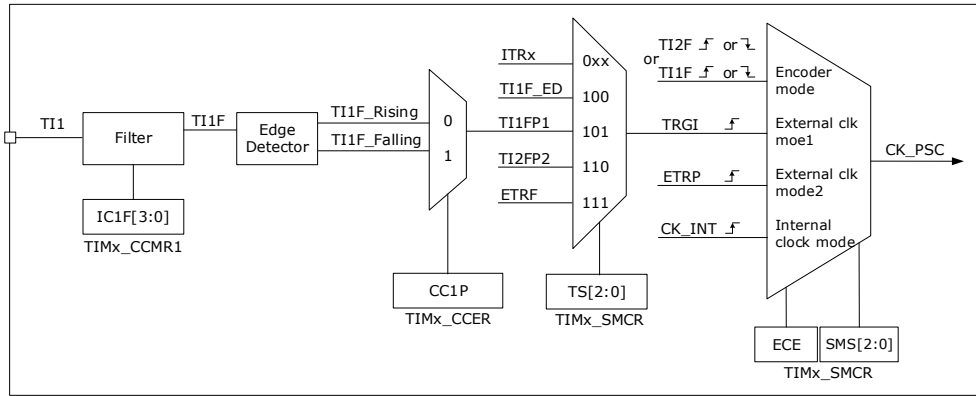


图 13-12 TI1 外部时钟连接范例

例:要配置向上计数器在 T1 输入端的上升沿计数，如以下步骤:

- ◆ 配置 GPTIMC_CCMR 寄存器 CC1S=01，配置信道 1 以检测 TI1 输入的上升沿。
- ◆ 写 GPTIMC_CCMR 寄存器之 IC1F[3:0]位以配置输入滤波器期间(若没有滤波器需求，维持 IC1F=0000)
- ◆ 写 CC1P=0 至寄存器 GPTIMC_CCER 以选择上升缘极性
- ◆ 透过写 GPTIMC_SMCR 寄存器的 SMS=111，以配置定时器外部时钟模式 1
- ◆ 写 GPTIMC_SMCR 寄存器中的 TS=0111，以选定 TI1 作为触发输入源
- ◆ 写 GPTIMC_CR1 寄存器之 CEN=1 以使能计数器

当上升缘发生于 TI1，计数器计数一次且 TIS 标志被设置

TI1 上升缘与实际时钟间之延时，取决于 TI1 输入之再同步电路

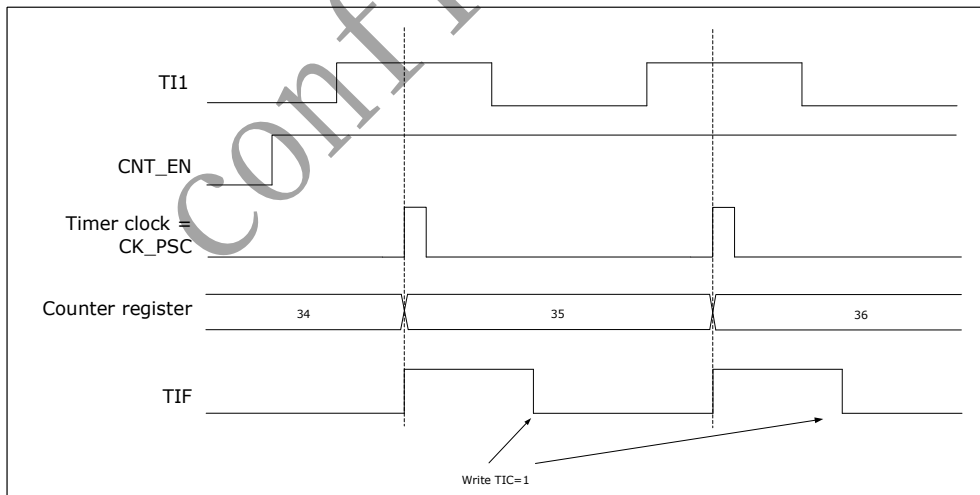


图 13-13 外部时钟模式 1 之控制电路

1. 3. 4. 3 内部触发输入(ITRx)

当 GPTIMC_SMCR 寄存器之 SMS=111 时此模式会被选定。计数器于被选定输入端之每一上升或下降缘计数。

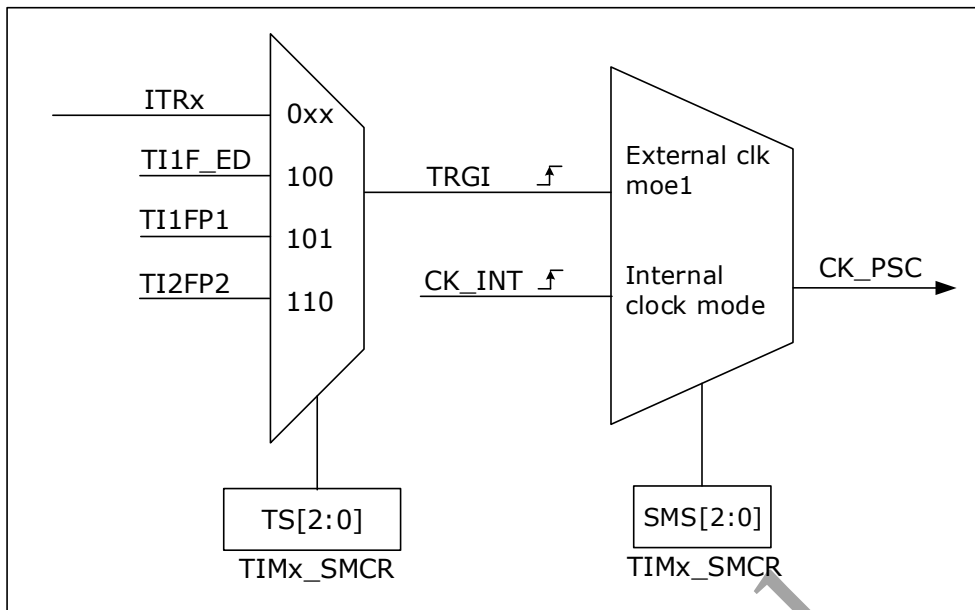


图 13-14 ITRx 外部时钟连接范例

例:要配置向上计数器在 ITRx 输入端的上升沿计数，如以下步骤:

- ◆ 透过写 GPTIMC_SMCR 寄存器的 SMS=111，以配置定时器外部时钟模式 1
- ◆ 写 GPTIMC_SMCR 寄存器中的 TS=0xxx，以选定 ITRx 作为触发输入源
- ◆ 写 GPTIMC_CR1 寄存器之 CEN=1 以使能计数器

当上升缘发生于 ITRx，计数器计数一次

ITRx 上升缘与实际时钟间之延时，取决于 ITRx 输入之再同步电路

1.3.5 捕获/比较信道

每一捕获/比较信道，皆是围绕着一捕获/比较寄存器(包含一影子寄存器)，包含一捕获输入部分(含数字滤波器，多路复用与预除器)以及一输出阶段(含比较器与输出控制)。

下图 28 为单一捕获/比较信道之概述

输入阶段范例对应之 Ti_x 输入以产生一滤波后之 Ti_xF 信号，接着一带极性边缘侦测器产生一信号(Ti_xFP_x)，可作为属模式控制器之触发输入或是做为捕获控制。该信号于捕获寄存器 (IC_xPS)之前即预先分频。

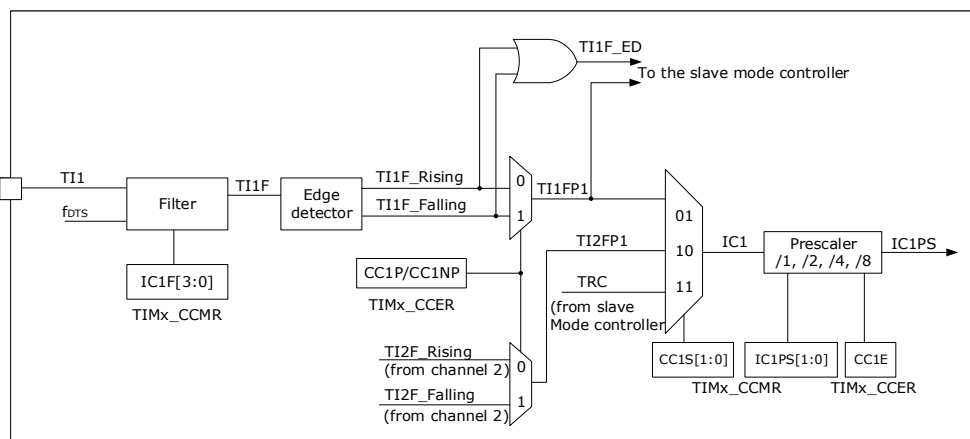


图 13-15 捕获/比较信道(例:信道 1 输入阶段)

输出部分产生一个中间波形 OCxRef(高有效)作为基准，链结的末端决定最终输出信号之极性。

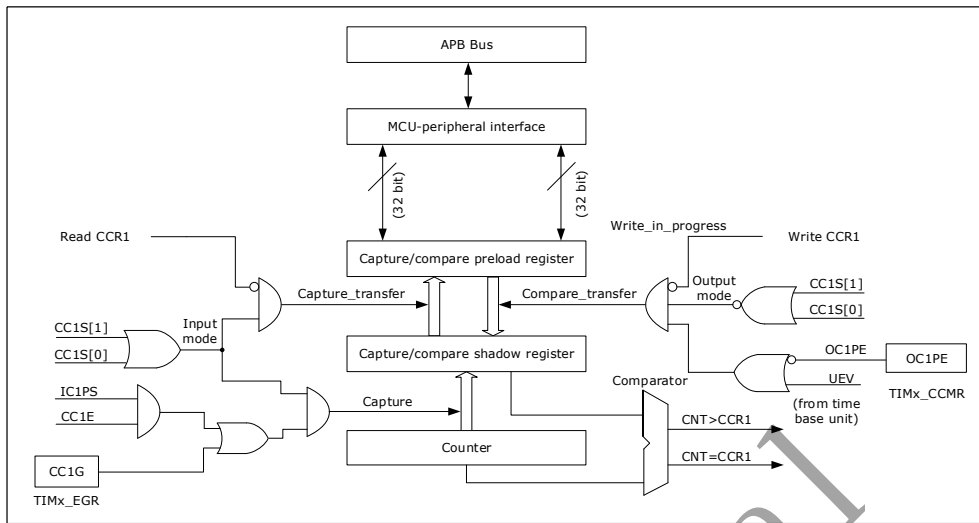


图 13-16 捕捉/比较信道 1 主电路

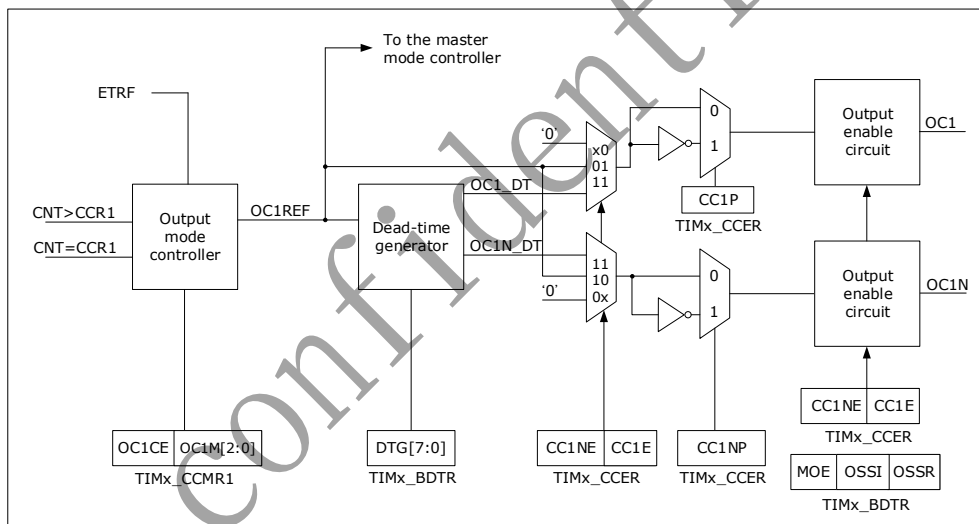


图 13-17 捕捉/比较信道之输出阶段(信道 1)

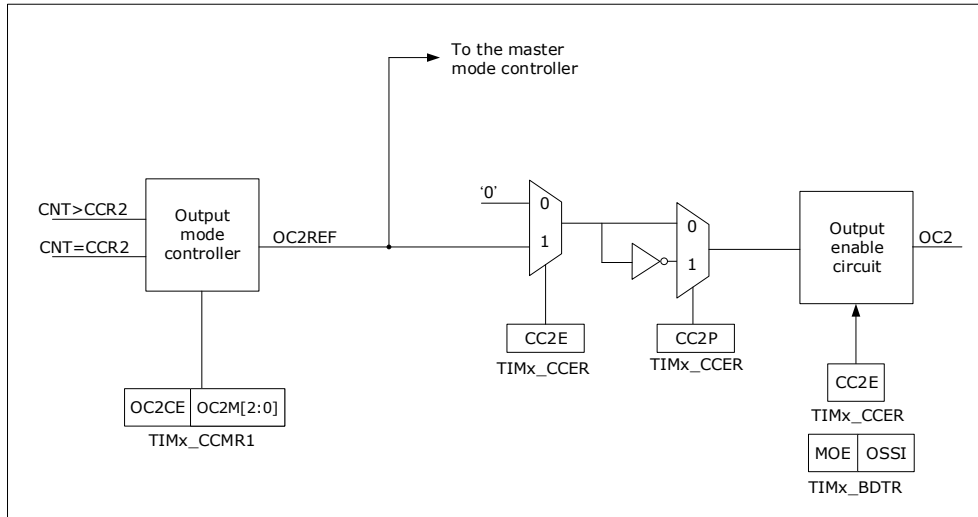


图 13-18 捕捉/比较信道之输出阶段(信道 2)

捕捉/比较区块是由一预载寄存器与影子寄存器组成。读写皆可存取预载寄存器。于捕捉模式下，捕捉实际上皆于影子寄存器完成并复制到预载寄存器。于比较模式下，预载寄存器内容被复制到影子寄存器与计数器做比较。

1.3.6 输入捕获模式

在输入捕获模式下，当检测到 ICx 信号上相应的边沿后，计数器的当前值被锁存到捕获/比较寄存器(GPTIMC_CCRx)中。当发生捕获时，若皆已使能则相应的中断可被送出。

以下例子说明如何在 TI1 输入的上升沿时捕获计数器的值到 GPTIMC_CCR1 寄存器中，步骤如下：

- ◆ 选择有效输入端：GPTIMC_CCR1 必须连接到 TI1 输入，所以写入 GPTIMC_CCMR1 寄存器中的 CC1S 为 01。只要 CC1S 不为'00'，信道被配置为输入，并且 GPTIMC_CCR1 寄存器变为只读。
- ◆ 参照连结至定时器之信号，编程你所需要的输入滤波器期间(当输入为 GPTIMC_CCMR1 寄存器之 TIX(ICxF 位其中一位)。当触发时输入信号于五个必要内部时钟周期内并不稳定。用户需编制比五个时钟周期还长之滤波器期间。可于 8 个新的电平接续采样被侦测到(于 fDTS 频率采样)时开始 TI1 传输。接着写 GPTIMC_CCMR1 寄存器之 IC1F 位为 0011。
- ◆ 选择 TI1 信道的有效转换边沿，在 GPTIMC_CCER 寄存器中写入 CC1P 与 CC1NP 至 0(上升沿)
- ◆ 对输入预除器编程。于例中希望捕获可在每一传输中作用，因此预除器禁止使用(写 GPTIMC_CCMR1 寄存器之 IC1PS 位为'00')。
- ◆ 设定 GPTIMC_CCER 寄存器 CC1E 位以使能捕捉计数器至捕捉寄存器
- ◆ 如果需要，透过设定 GPTIMC_IER 寄存器之 CC1 位以开启相关中断需求
- ◆ 当输入捕获中断发生时：

- ◆ GPTIMC_CCR1 寄存器得到计数器之作用传输值
- ◆ CC1IF 状态被设置(中断状态).CC1OF 也被设定, 若至少有 2 个接续捕获发生, 然而标志未被清除
- ◆ 如设置 GPTIMC_IER 寄存器之 CC1 位, 则会产生一个中断。
- ◆ 如设置 GPTIMC_DMAEN 寄存器之 CC1 位, 则还会产生一个 DMA 请求。

为了处理捕获溢出, 建议在读出捕获溢出标志之前读取数据, 这是为了避免丢失在读出捕获溢出标志之后和读取数据之前可能产生的捕获溢出信息。

注:ICx 中断需求可由软件设置 GPTIMC_EGR 寄存器内对应 CCxG 位产生

1. 3. 7 PWM 输入模式

该模式是输入捕获模式的一个特例, 除下列区别外, 操作与输入捕获模式相同:

- ◆ 两个 ICx 信号被映射至同一个 TIx 输入
- ◆ 这 2 个 ICx 信号为边沿有效, 但是极性相反
- ◆ 其中一个 TIxFP 信号被作为触发输入信号, 而从模式控制器被配置成复位模式

例:可测量输入到 TI1 上的 PWM 信号的长度(GPTIMC_CCR1 寄存器)和占空比(GPTIMC_CCR2 寄存器), 具体步骤如下(取决于 CK_INT 的频率和预分频器的值)

- ◆ 选择 GPTIMC_CCR1 的有效输入: 置 GPTIMC_CCMR1 寄存器的 CC1S=01(选中 TI1)
- ◆ 选择 TI1FP1 的有效极性(用来捕获数据到 GPTIMC_CCR1 中和清除计数器): 置 CC1P 为 0(上升沿有效)
- ◆ 选择 GPTIMC_CCR2 有效输入:置 GPTIMC_CCMR1 寄存器之 CC2S 位至 10(TI1 被选中)
- ◆ 选择 TI1FP2 有效极性(用于 GPTIMC_CCR2 捕捉):置 CC2P 位至为'1'(于下降缘作用)
- ◆ 选择可作用触发输入:置 GPTIMC_SMCR 寄存器之 TS 位为 101(选择 TI1FP1)
- ◆ 设置属模式控制器于重置模式:置 GPTIMC_SMCR 寄存器之 SMS 位为 100
- ◆ 使能捕获: 置 GPTIMC_CCER 寄存器中 CC1E=1 且 CC2E=1。

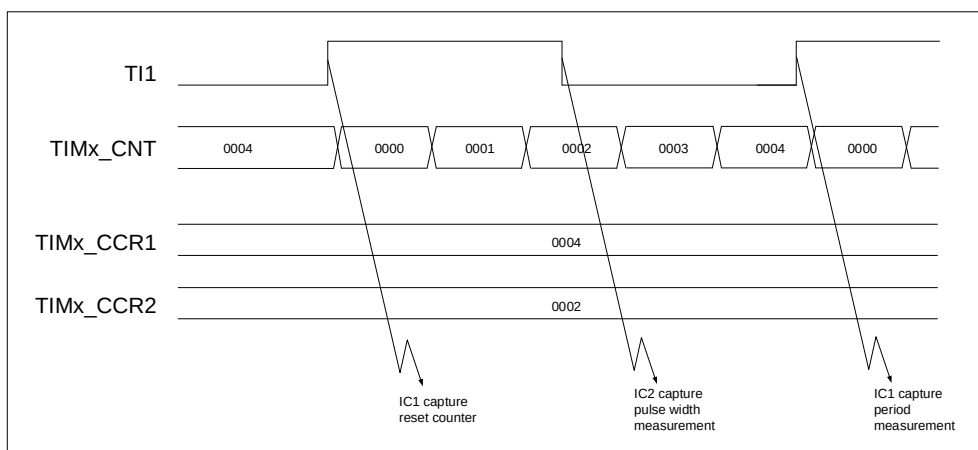


图 13-19PWM 输入模式时序

1.3.8 强制输出模式

在输出模式(GPTIMC_CCMRx 寄存器中 CCxS=00)下, 输出比较信号(OCxREF 和相应的 OCx/OCxN)能够直接由软件强置为有效或无效状态, 而不依赖于输出比较寄存器和计数器间的比较结果。

置 GPTIMC_CCMRx 寄存器中相应的 OCxM=101, 即可强置输出比较信号(OCxREF/OCx)为有效状态。这样 OCxREF 被强置为高电平(OCxREF 始终为高电平有效), 同时 OCx 得到 CCxP 极性相反的信号。

例如: CCxP=0(OCx 高电平有效), 则 OCx 被强置为高电平。

置 GPTIMC_CCMRx 寄存器中的 OCxM=100, 可强置 OCxREF 信号为低。

该模式下, 在 GPTIMC_CCRx 影子寄存器和计数器之间的比较仍然在进行, 相应的标志也会被修改。因此仍然会产生相应的中断和 DMA 请求。这将会在下面的输出比较模式一节中介绍。

1.3.9 输出比较模式

此功能用于控制输出波形或是当时间消逝时之周期标示。

当于捕捉/比较寄存器与计数器发现匹配时, 输出比较功能:

- ◆ 将输出比较模式(GPTIMC_CCMRx 寄存器中的 OCxM 位)和输出极性(GPTIMC_CCER 寄存器中的 CCxP 位)定义的值输出到对应的引脚上。在比较匹配时, 输出引脚可以保持它的电平(OCxM=000)、被设置成有效电平(OCxM=001)、被设置成无效电平(OCxM=010)或进行翻转(OCxM=011)。
- ◆ 设置中断状态寄存器中的标志位(GPTIMC_RIF 寄存器中的 CCxIF 位)。
- ◆ 若设置了相应的中断屏蔽(GPTIMC_IER 寄存器中的 CCxIE 位), 则产生一个中断。
- ◆ 若设置了相应的使能位(GPTIMC_DMAEN 寄存器中的 CCxDE 位, GPTIMC_CR2 寄存器中的 CCDS 位选择 DMA 请求功能), 则产生一个 DMA 请求。

GPTIMC_CCRx 寄存器可由或不由预除寄存器之 OCxPE 位编程

于输出比较模式下, 更新事件 UEV 对于 OCxREF 与 OCx 输出没有效果。时程解析亦为计数器之单一计数。输出比较模式也可被用来输出单一脉冲(单脉冲模式)。于输出比较模式, 更新事件 UEV 于 OCxREF 与 OCx 无效果

过程:

- ◆ 选定计数器时钟(内部,外部,预除)。
- ◆ 写入需要之数据至 GPTIMC_ARR 与 GPTIMC_CCRx 寄存器。
- ◆ 若中断需求产生则设置 GPTIMC_IER 之 CCxIE 位
- ◆ 选择输出模式, 例:
 - 当 CNT 符合 CCRx 时写 OCxM=011 以触发 OCx 输出引脚

- 写 OCxPE =0 禁用预载寄存器
- 写 CCxP =0 选择高作用极性
- 写 CCxE =1 使能输出

◆ 设置 GPTIMC_CR1 寄存器之 CEN 位以使能计数器

GPTIMC_CCRx 寄存器可于任何时间被软件更新以控制输出波形,预载寄存器提供预除寄存器并未使能(OCxPE='0',此外 GPTIMC_CCRx 影子寄存器只有在下一个更新事件 UEV 被更新).

图 11-31 提供一范例:

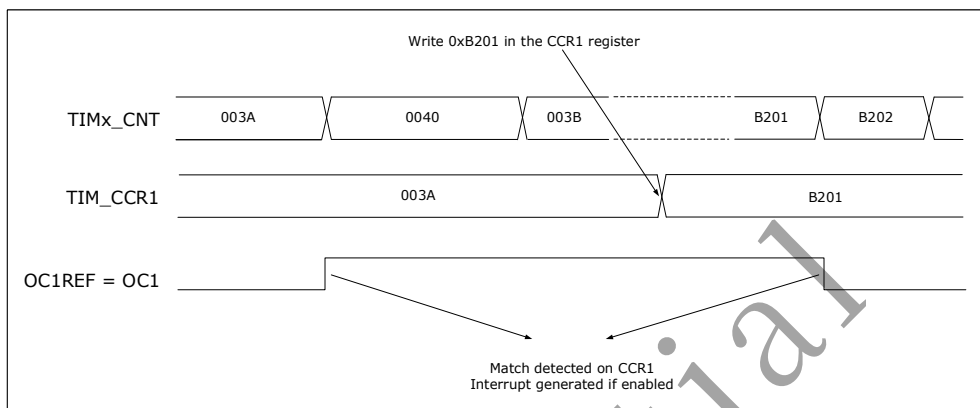


图 13-20 输出比较模式,触发 OC1

1. 3. 10 PWM 模式

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

在 GPTIMC_CCMRx 寄存器中的 OCxM 位写入 '110'(PWM 模式 1)或 '111'(PWM 模式 2), 能够独立地设置每个 OCx 输出信道产生一路 PWM。必须设置 GPTIMC_CCMRx 寄存器 OCxPE 位以使能相应的预装载寄存器, 最后还要设置 GPTIMC_CR1 寄存器的 ARPE 位, (在向上计数或中心对称模式中)使能自动重载的预装载寄存器。

仅当发生一个更新事件的时候, 预装载寄存器才能被传送到影子寄存器, 因此在计数器开始计数之前, 必须通过设置 GPTIMC_EGR 寄存器中的 UG 位以初始化所有寄存器。

OCx 的极性可以通过软件在 GPTIMC_CCER 寄存器中的 CCxP 位设置, 它可以设置为高电平有效或低电平有效。GPTIMC_CCER 寄存器中的 CCxE, CCxNE, MOE, OSS1 和 OSSR 位联合控制 OCx 输出使能(GPTIMC_CCER 与 GPTIMC_BDTR 寄存器)。详见 GPTIMC_CCER 寄存器的描述。

在 PWM 模式(模式 1 或模式 2)下, GPTIMC_CNT 和 GPTIMC_CCRx 始终在进行比较, (依据计数器的计数方向)以确定是否符合 $GPTIMC\_CCRx \leq GPTIMC\_CNT$ 或者 $GPTIMC\_CNT \leq GPTIMC\_CCRx$ 。

根据 GPTIMC_CR1 寄存器中 CMS 位的状态, 定时器能够产生边沿对齐的 PWM 信号或中央对齐的 PWM 信号。

PWM 边沿对齐模式

◆ 向上计数配置

当 GPTIMC_CR1 寄存器中的 DIR 位为低的时候执行向上计数。参考以下为一 PWM 模式 1 之例子。当 $GPTIMC\_CNT < GPTIMC\_CCRx$ 时, PWM 参考信号 OCxREF 为高, 否则为低。如果 $GPTIMC\_CCRx$ 中的比较值大于自动重载值($GPTIMC\_ARR$), 则 OCxREF 保持为‘1’。如果比较值为 0, 则 OCxREF 保持为‘0’。下图为 $GPTIMC\_ARR=8$ 时边沿对齐的 PWM 波形实例。

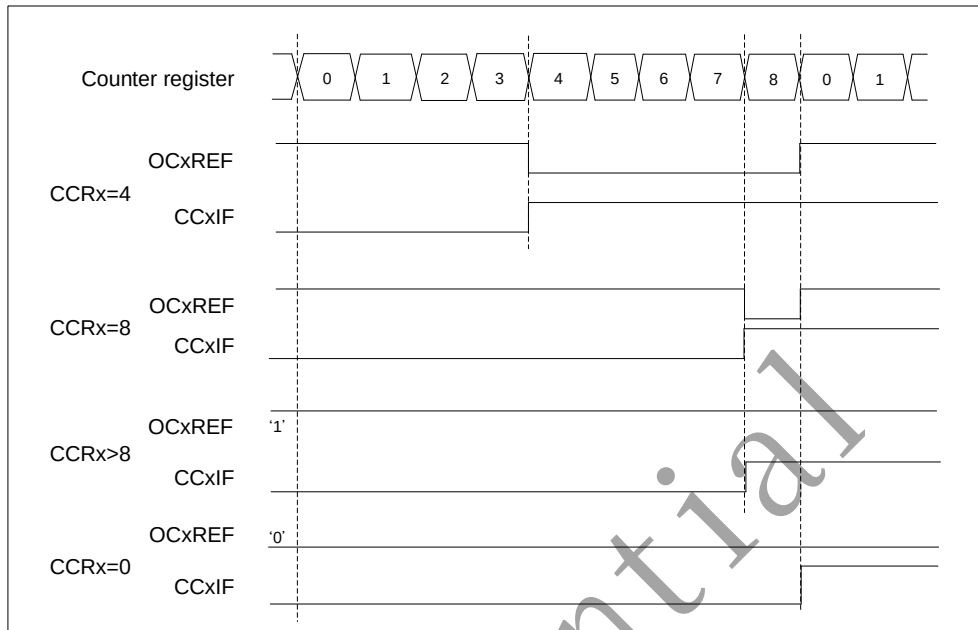


图 13-21 边缘对齐 PWM 波形, 向上计数($ARR=8$)

1. 3. 11 单脉冲模式

单脉冲模式(OPM)为前一模式之特例。其允许计数器对应刺激起始并于可编程延迟后产生可编程长度之脉冲。

起始计数器可由属模式控制器控制。由输出比较模式或 PWM 模式产生波形。可透过设置 GPTIMC_CR1 寄存器之 OPM 位以选择单脉冲模式。如此计数器会在下一个更新事件 UEV 自动停止。

单一脉冲仅可于比较值不同于计数器初始值时被正确产生。在开始之前(当定时器于等待触发器时), 必须依照下列方式配置

- ◆ 向上计数: $CNT < CCRx \leq ARR$ (特例, $0 < CCRx$)
- ◆ 向下计数: $CNT > CCRx$

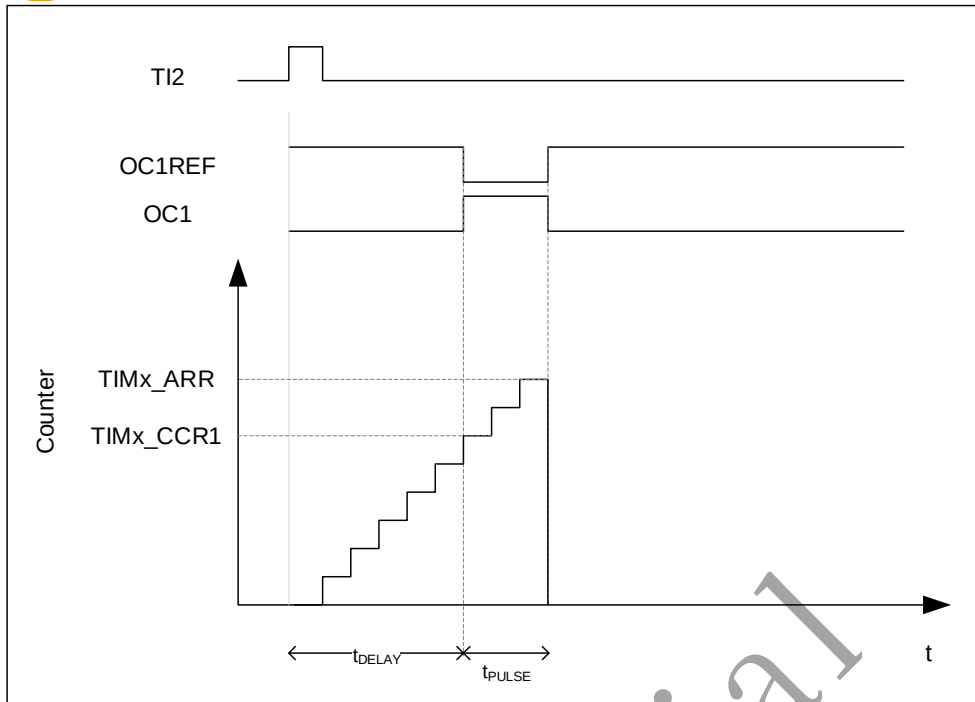


图 13-22 单脉冲模式范例

例如，你需要在在 OC1 上产生一个长度为 t_{PULSE} 的正脉冲，从 TI2 输入脚上检测到一个上升沿开始，延迟 t_{DELAY} 之后。

假定 TI2FP2 作为触发 1:

- ◆ 置 GPTIMC_CCMR1 寄存器中的 CC2S=01，把 TI2FP2 映像到 TI2。
- ◆ 置 GPTIMC_CCER 寄存器中的 CC2P=0 和 CC2NP=0，使 TI2FP2 能够检测上升沿。
- ◆ 置 GPTIMC_SMCR 寄存器中的 TS=110，TI2FP2 作为从模式控制器的触发(TRGI)。
- ◆ 置 GPTIMC_SMCR 寄存器中的 SMS=110(触发模式)，TI2FP2 被用来启动计数器。

OPM 的波形由写入比较寄存器决定(要考虑时钟频率和计数器预分频器)

- ◆ t_{DELAY} 由 GPTIMC_CCR1 寄存器中的值定义。
- ◆ t_{PULSE} 由自动装载值和比较值之间的差值定义($GPTIMC\_ARR - GPTIMC\_CCR1$)。
- ◆ 假定当发生比较匹配时要产生从 0 到 1 的变换，当计数器达到预装载值时要产生一个从 1 到 0 的变换：首先要置 GPTIMC_CCMR1 寄存器的 OC1M=111，选择 PWM 模式 2；根据需要有选择地使能预装载寄存器：置 GPTIMC_CCMR1 中的 OC1PE=1 和 GPTIMC_CR1 寄存器中的 ARPE；此时必须向 GPTIMC_CCR1 寄存器中填写比较值，在 GPTIMC_ARR 寄存器中填写自动装载值，设置 UG 位来产生一个更新事件，然后等待在 TI2 上的一个外部触发事件。本例中，CC1P=0。

在这个例子中，GPTIMC_CR1 寄存器中的 DIR 和 CMS 位应该置低。

因为只需要一个脉冲，所以必须设置 GPTIMC_CR1 寄存器中的 OPM=1，在下一个更新事件(当计数器从自动装载值翻转到 0)时停止计数。当 GPTIMC_CR1 寄存器中的 OPM=0 时，进入重复模式。

特殊情况：OCx 快速使能

在单脉冲模式下，在 T1x 输入脚的边沿检测逻辑设置 CEN 位来启动计数器。然后计数器和比较值间的比较结果驱动输出的转换。但是这些操作需要一定的时钟周期，因此它限制了可得到的最小延时 tDELAY。

如果需要以最小延时输出波形，可以设置 GPTIMC_CCMRx 寄存器中的 OCxFE 位；此时 OCxREF(和 OCx)直接响应激励而不再依赖比较的结果，输出的波形与比较匹配时的波形一样。OCxFE 只在信道配置为 PWM 模式 1 和 PWM 模式 2 时起作用。

1. 3. 12 互补输出与死区内嵌

高级控制定时器可输出两个互补信号并管理开关切换

该时间通常被认作 dead-time 且须依链接之装置输出与其特性调整(因电平切换、电源切换造成之本质延迟...)

可单独选择输出极性(主要输出 OCx 或互补性 OCxN)。透过写 GPTIMC_CCER 寄存器之 CCxP 与 CCxNP 位达到前述效果

互补性信号 OCx 与 OCxN 皆由数个控制位结合进行激活:GPTIMC_CCER 寄存器之 CCxE 与 CCxNE 位，GPTIMC_BDTR 与 GPTIMC_CR2 寄存器之 MOE,OISx,OISxN,OSSI 与 OSSR 位。特别是当切换至 IDLE(MOE 下降至 0)状态时，Dead-time 会被作用。

Dead-time 插入藉由设定 CCxE 与 CCxNE 位使能，若有暂停电路则还有 MOE 位。配有一适用于各个信道之 8 位 dead-time 产生器(用户可藉由 GPTIMC_BDTR 之 DTSEL 决定上升或下降延迟为 8/4-bit)参考波形 OCxREF，产生 2 个输出 OCx 及 OCxN，若 OCx 与 OCxN 被拉高

- ◆ OCx 输出信号与参考信号相同，上升缘除外，其延迟与参考上升缘相关。
- ◆ OCxN 输出信号为一除了上升缘以外之相对参考信号

相对于参考下降缘，其被延迟

若延迟大于作用输出带宽(OCx 或 OCxN)那么对应脉冲不会产生。

下图显示 dead-time 产生器输出信号与参考信号 OCxREF 间之关系(假设 CCxP=1,CCxNP=0,MOE=1,CCxE=1 且于这些范例中 CCxNE=1)

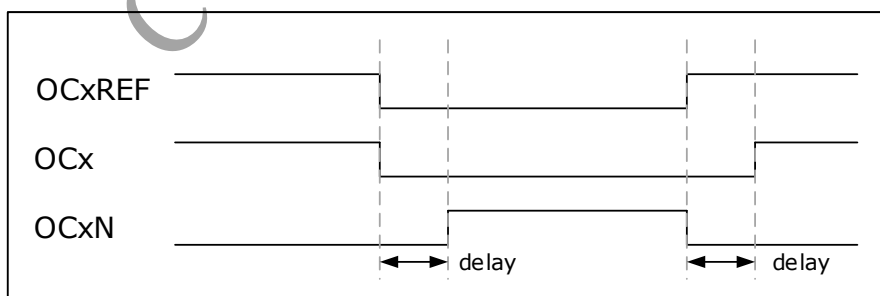


图 13-23 互补输出含 dead-time 插入

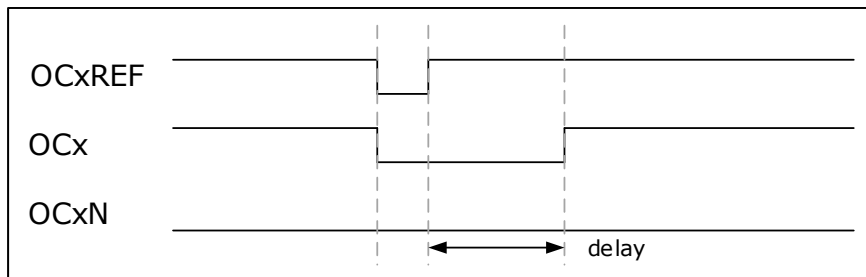


图 13-24Dead-time 波形含大于负脉冲之延迟

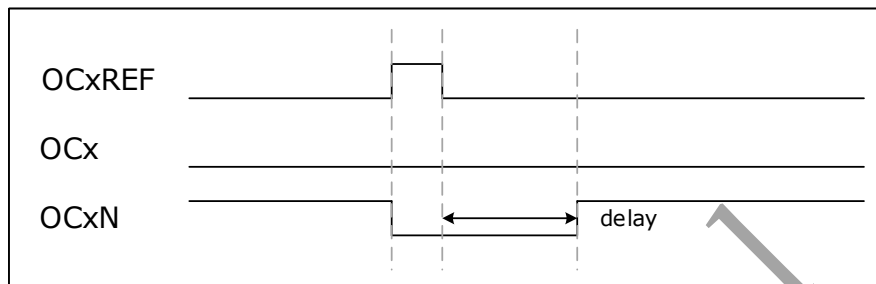


图 13-25 波形大于正脉冲之延迟

重新导向 OCxREF 至 OCx 或 OCxN

于输出模式(强制, 输出比较或 PWM), OCxREF 可透过设置 GPTIMC_CCER 寄存器之 CCxE 与 CCxNE 位被重新导向至 OCx 输出或至 OCxN 输出。

可发送特定波形(如 PWM 或静止作用电平)于单一输出, 而互补维持其无作用电平。其他可能为无作用电平同时有输出或是输出作用与互补 dead-time。

注:当只有 OCxN 使能时(CCxE=0, CCxNE=1), 其不互补且 OCxREF 拉高时即会作用。举例说明, 若 CCxNP=1 则 OCxN=OCxRef。另一方面, 当 OCx 与 OCxN 皆为使能(CCxE=CCxNE=1)OCx 开始作用, 当 OCxREF 为高然而 OCxN 为互补且 OCxREF 为低时开始作用。

1. 3. 13 使用暂停功能

当使用刹车功能时, 依据相应的控制位(GPTIMC_BDTR 寄存器中的 MOE、OSSI 和 OSSR 位, GPTIMC_CR2 寄存器中的 OISx 和 OISxN 位), 输出使能信号和无效电平都会被修改。但无论何时, OCx 和 OCxN 输出不能在同一时间同时处于有效电平上。详见带刹车功能的互补输出通道 OCx 和 OCxN 的控制位。

刹车源既可以是刹车输入引脚又可以是一个时钟失败事件。时钟失败事件由复位时钟控制器中的时钟安全系统(CSS)产生, 详见节时钟安全系统。

系统复位后, 刹车电路被禁止, MOE 位为低。设置 GPTIMC_BDTR 寄存器中的 BKE 位可以使能刹车功能, 刹车输入信号的极性可以通过配置同一个寄存器中的 BKP 位选择。BKE 和 BKP 可以同时被修改。当写入 BKE 和 BKP 位时, 在真正写入之前会有 1 个 APB 时钟周期的延迟, 因此需要等待一个 APB 时钟周期之后, 才能正确地读回写入的位。

因为 MOE 下降沿可以是异步的, 在实际信号(作用在输出端)和同步控制位(在

GPTIMC_BDTR 寄存器中)之间设置了一个再同步电路。这个再同步电路会在异步信号和同步信号之间产生延迟。特别的, 如果当它为低时写 MOE=1, 则读出它之前必须先插入一个延时(空指令)才能读到正确的值。这是因为写入的是异步信号而读的是同步信号。

当发生刹车时(在刹车输入端出现选定的电平), 有下述动作:

- ◆ MOE 位被异步地清除, 将输出置于无效状态、空闲状态或者复位状态(由 OSSI 位选择)。这个特性在 MCU 的振荡器关闭时依然有效。
- ◆ 一旦 MOE=0, 每一个输出通道输出由 GPTIMC_CR2 寄存器中的 OISx 位设定的电平。如果 OSSI=0, 则定时器释放使能输出, 否则使能输出始终为高。
- ◆ 当使用互补输出时:
 - 输出首先被置于复位状态即无效的状态(取决于极性)。这是异步操作, 即使定时器没有时钟时, 此功能也有效。
 - 如果定时器的时钟依然存在, 死区生成器将会重新生效, 在死区之后根据 OISx 和 OISxN 位指示的电平驱动输出端口。即使在这种情况下, OCx 和 OCxN 也不能被同时驱动到有效的电平。注, 因为重新同步 MOE, 死区时间比通常情况下长一些(大约 2 个 ck_tim 的时钟周期)。
 - 如果 OSSI=0, 定时器释放使能输出, 否则保持使能输出; 或一旦 CCxE 与 CCxNE 之一变高时, 使能输出变为高。
- ◆ 如果设置了 GPTIMC_IER 寄存器中的 BIE 位, 当刹车状态标志(GPTIMC_RIF 寄存器中的 BIF 位)为'1'时, 则产生一个中断。如果设置了 GPTIMC_DMAEN 寄存器中的 BDE 位, 则产生一个 DMA 请求。
- ◆ 如果设置了 GPTIMC_BDTR 寄存器中的 AOE 位, 在下一个更新事件 UEV 时 MOE 位被自动置位; 例如, 这可以用来进行整形。否则, MOE 始终保持低直到被再次置'1'; 此时, 这个特性可以被用在安全方面, 你可以把刹车输入连到电源驱动的报警输出、热敏传感器或者其他安全器件上。

注: 刹车输入为电平有效。所以, 当刹车输入有效时, 不能同时(自动地或者通过软件)设置 MOE。同时, 状态标志 BIF 不能被清除。

刹车由 BRK 输入产生, 它的有效极性是可编程的, 且由 GPTIMC_BDTR 寄存器中的 BKE 位开启。

除了刹车输入和输出管理, 刹车电路中还实现了写保护以保证应用程序的安全。它允许用户冻结几个配置参数(死区长度, OCx/OCxN 极性和被禁止的状态, OCxM 配置, 刹车使能和极性)。用户可以通过 GPTIMC_BDTR 寄存器中的 LOCK 位, 从三级保护中选择一种, 参看刹车和死区寄存器(GPTIMC_BDTR)。在 MCU 复位后 LOCK 位只能被修改一次。

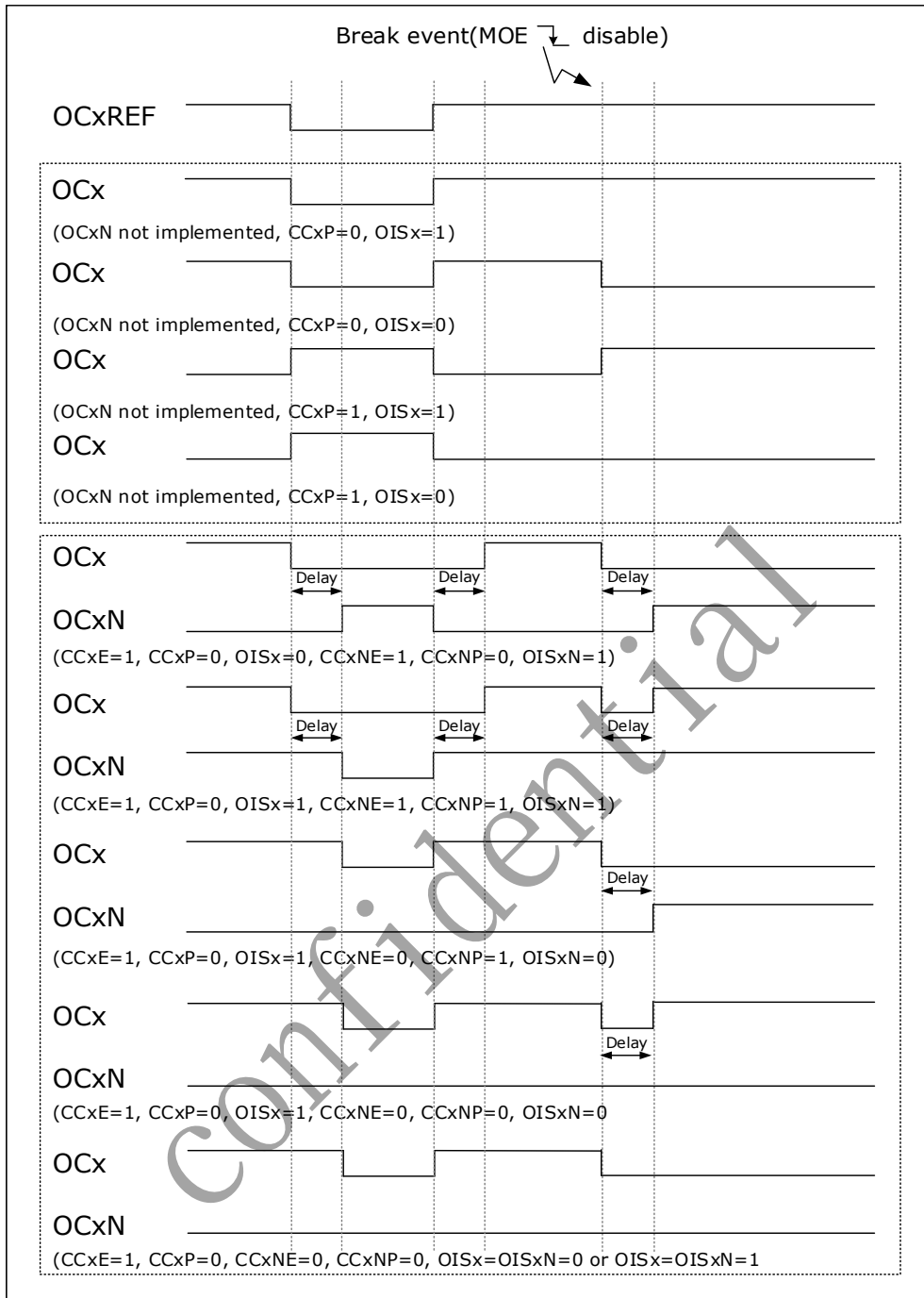


图 13-26 响应暂停之输出行为

1. 3. 14 产生六步 PWM 输出

当在一个通道上需要互补输出时，预装载位有 OCxM、CCxE 和 CCxNE。在发生 COM 换相事件时，这些预装载位被传送到影子寄存器位。这样你就可以预先设置好下一步配置，并在同一个时刻同时修更改所有信道的配置。COM 可以通过设置 GPTIMC_EGR 寄存器的 COM 位由软件产生，或在 TRGI 上升沿由硬件产生。

当发生 COM 事件时会设置一个标志位(GPTIMC_SR 寄存器中的 COMIF 位)，这时如果已设置了 GPTIMC_IER 寄存器的 COMIE 位，则产生一个中断；如果已设置了 GPTIMC_DMAEN

寄存器的 COMDE 位，则产生一个 DMA 请求。

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

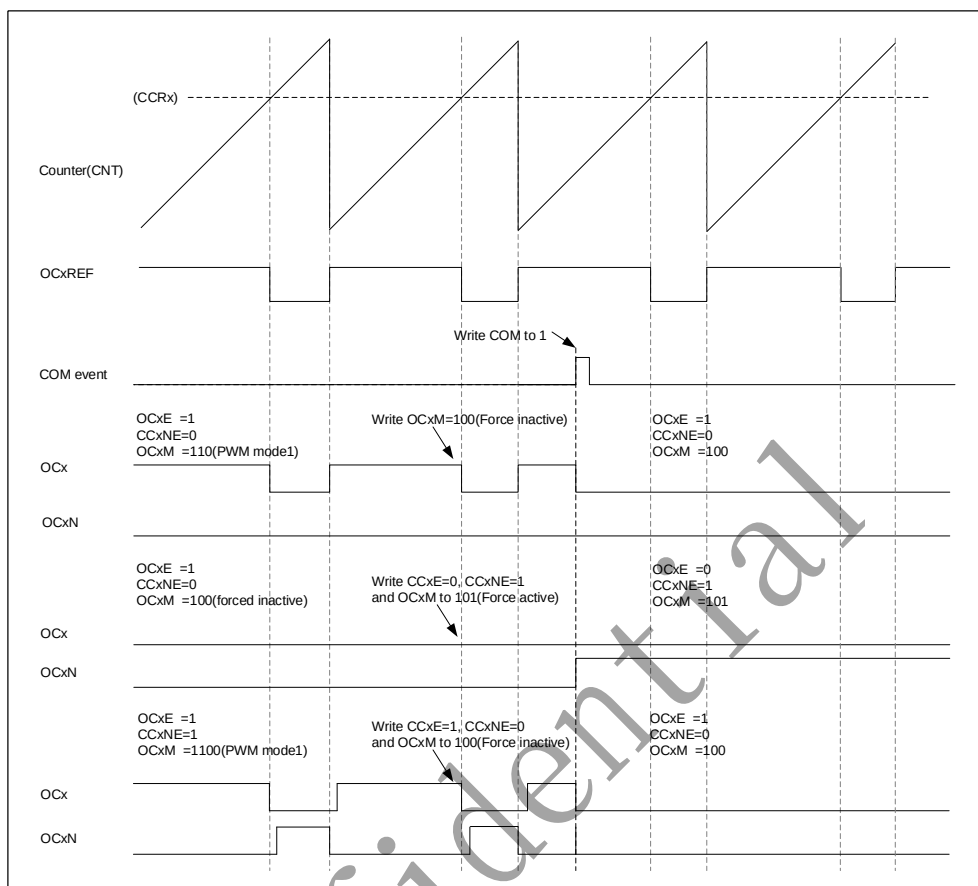


图 13-276 阶产生,COM 范例

1. 3. 15 定时器和外部触发的同步

GPTIMC 定时器能够在多种模式下和一个外部的触发同步：复位模式、门控模式和触发模式。

1. 3. 15. 1 从模式：复位模式

在一个触发输入发生事件时，计数器和它的预分频器能够重新被初始化；同时，如果 GPTIMC_CR1 寄存器的 URS 位为低，还产生一个更新事件 UEV；然后所有的预装载寄存器 (GPTIMC_ARR, GPTIMC_CCRx) 都被更新。

在以下的例子中，TI1 输入端的上升沿导致向上计数器被清零：

- ◆ 配置信道 1 以检测 TI1 的上升沿。配置输入滤波器的带宽(在本例中，不需要任何滤波器，因此保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位只选择输入捕获源，即 GPTIMC_CCMR1 寄存器中 CC1S=01。置 GPTIMC_CCER 寄存器中 CC1P=0 和 CC1NP=0 以确定极性(只检测上升沿)。
- ◆ 置 GPTIMC_SMCR 寄存器中 SMS=100，配置定时器为复位模式；置 GPTIMC_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- ◆ 置 GPTIMC_CR1 寄存器中 CEN=1，启动计数器。

计数器开始依据内部时钟计数，然后正常运转直到 TI1 出现一个上升沿；此时，计数器被清零然后从 0 重新开始计数。同时，触发标志(GPTIMC_RIF 寄存器中的 TIF 位)被设置，并根据 GPTIMC_IER 寄存器中 TIE(中断使能)位和 GPTIMC_DMAEN 寄存器中 TDE(DMA 使能)位的设置，产生一个中断请求或一个 DMA 请求。

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

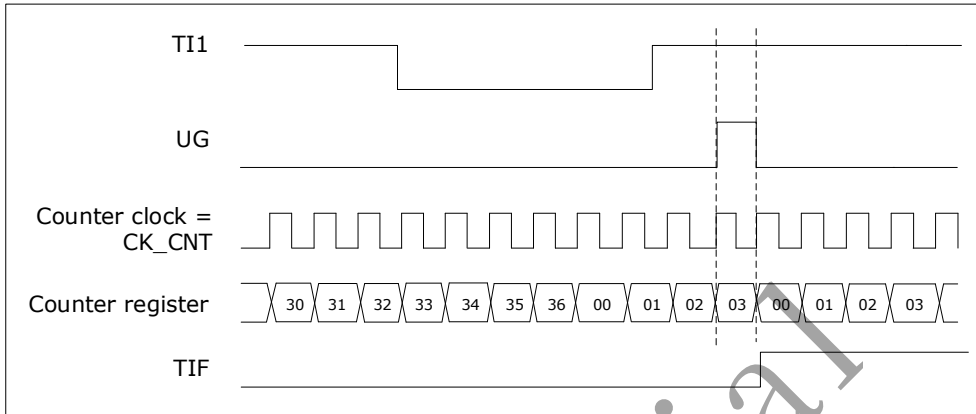


图 13-28 重置模式控制电路

1. 3. 15. 2 从模式：门控模式

按照选中的输入端电平使能计数器。

在如下的例子中，计数器只在 TI1 为低时向上计数：

- ◆ 配置信道 1 以检测 TI1 上的低电平。配置输入滤波器带宽(本例中，不需要滤波，所以保持 IC1F=0000)。触发操作中不使用捕获预分频器，所以不需要配置。CC1S 位用于选择输入捕获源，置 GPTIMC_CCMR1 寄存器中 CC1S=01。置 GPTIMC_CCER 寄存器中 CC1P=1 和 CC1NP=0 以确定极性(只检测低电平)。
- ◆ 置 GPTIMC_SMCR 寄存器中 SMS=101，配置定时器为门控模式；置 GPTIMC_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。
- ◆ 置 GPTIMC_CR1 寄存器中 CEN=1，启动计数器。(在门控模式下，如果 CEN=0，则计数器不能启动，不论触发输入电平如何)

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

TI1 上升沿和计数器实际停止之间的延时取决于 TI1 输入端的重同步电路。

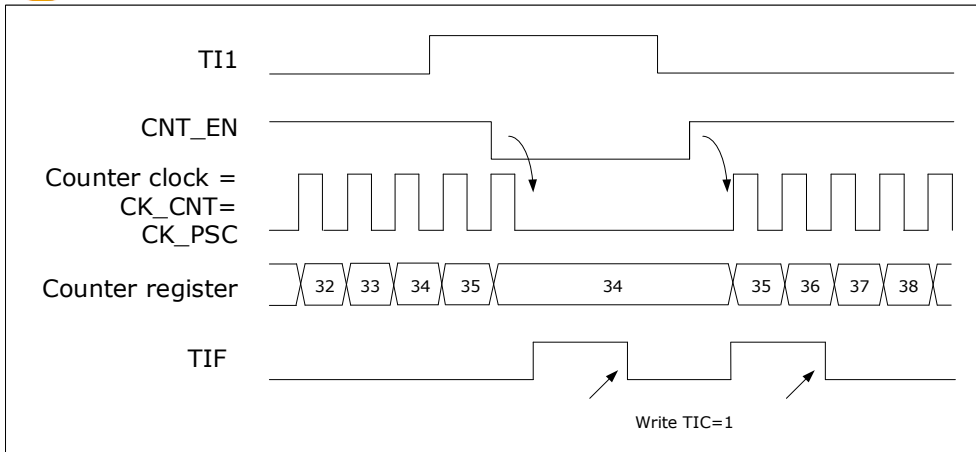


图 13-29 门控模式控制电路

1. 3. 15. 3 从模式：触发模式

输入端上选中的事件使能计数器。

在下面的例子中，计数器在 TI2 输入的上升沿开始向上计数：

- ◆ 配置信道 2 检测 TI2 的上升沿。配置输入滤波器带宽(本例中，不需要任何滤波器，保持 IC2F=0000)。触发操作中不使用捕获预分频器，不需要配置。CC2S 位只用于选择输入捕获源，置 GPTIMC_CCMR1 寄存器中 CC2S=01。置 GPTIMC_CCER 寄存器中 CC2P=1 和 CC1NP=0 以确定极性(只检测低电平)。
- ◆ 置 GPTIMC_SMCR 寄存器中 SMS=110，配置定时器为触发模式；置 GPTIMC_SMCR 寄存器中 TS=110，选择 TI2 作为输入源。

当 TI2 出现一个上升沿时，计数器按内部时钟开始计数，同时设置 TIF 标志。

TI2 上升沿和计数器启动计数之间的延时，取决于 TI2 输入端的重同步电路。

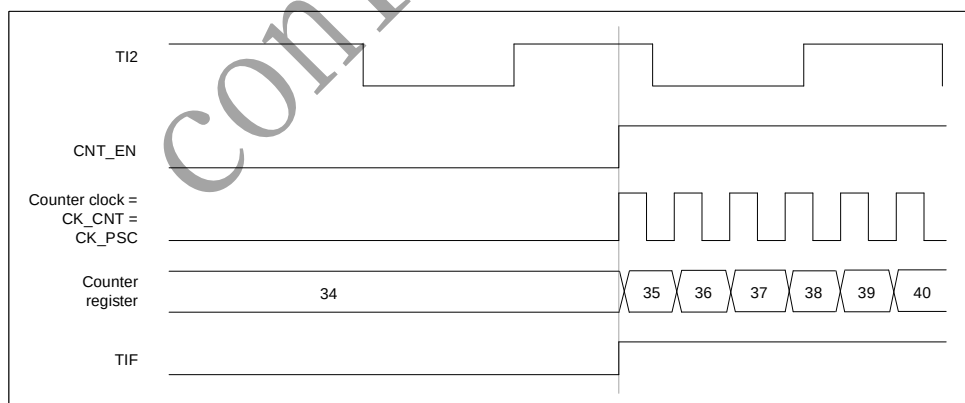


图 13-30 触发模式控制电路

1. 3. 15. 4 从模式：外部时钟模式 2+触发模式

外部时钟模式 2 可以与另一种从模式(外部时钟模式 1 和编码器模式除外)一起使用。这时，ETR 信号被用作外部时钟的输入，可以选择另一个输入作为触发输入（在复位模式、门控模式或触发模式）。不建议使用 GPTIMC_SMCR 寄存器的 TS 位选择 ETR 作为 TRGI。

在下面的例子中，在 TI1 上出现一个上升沿后，向上计数器在 ETR 的每一个上升沿加一：

- ◆ 通过 GPTIMC_SMCR 寄存器配置外部触发输入电路：ETF=0000：没有滤波
 - ETPS=00：不用预分频器
 - ETP=0：检测 ETR 的上升沿，置 ECE=1 使能外部时钟模式 2。
- ◆ 按如下配置信道 1，检测 TI 的上升沿：
 - IC1F=0000：没有滤波
 - 触发操作中不使用捕获预分频器，不需要配置
 - 置 GPTIMC_CCMR1 寄存器中 CC1S=01，选择输入捕获源
 - 置 GPTIMC_CCER 寄存器中 CC1P=0 和 CC1NP=0 以确定极性(只检测上升沿)
- ◆ 置 GPTIMC_SMCR 寄存器中 SMS=110，配置定时器为触发模式。置 GPTIMC_SMCR 寄存器中 TS=101，选择 TI1 作为输入源。

当 TI1 上出现一个上升沿时，TIF 标志被设置，计数器开始在 ETR 的上升沿计数。

ETR 信号的上升沿和计数器实际复位间的延时，取决于 ETRP 输入端的重同步电路。

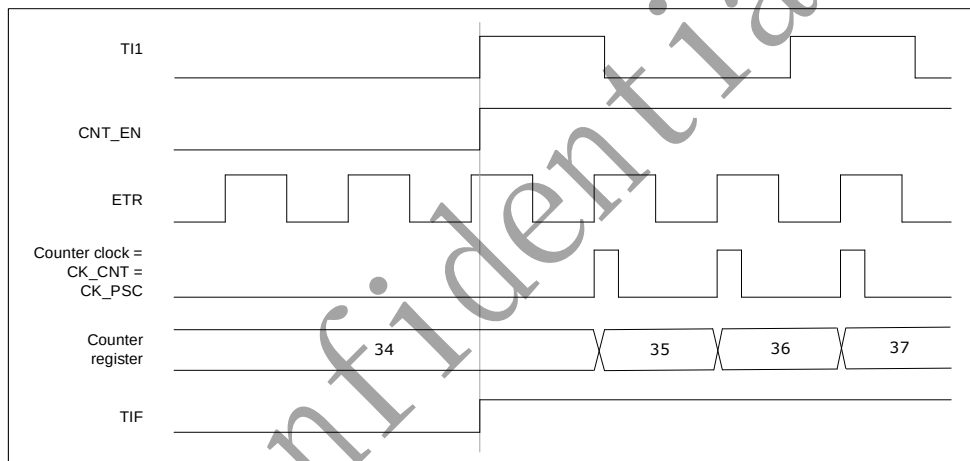


图 13-31 外部时钟模式 2+触发模式下的控制电路

1. 3. 16 定时器同步

所有 GPTIMC 定时器在内部相连，用于定时器同步或链接。当一个定时器处于主模式时，它可以对另一个处于从模式的定时器的计数器进行复位、启动、停止或提供时钟等操作。

下图显示了触发选择和主模式选择模块的概况

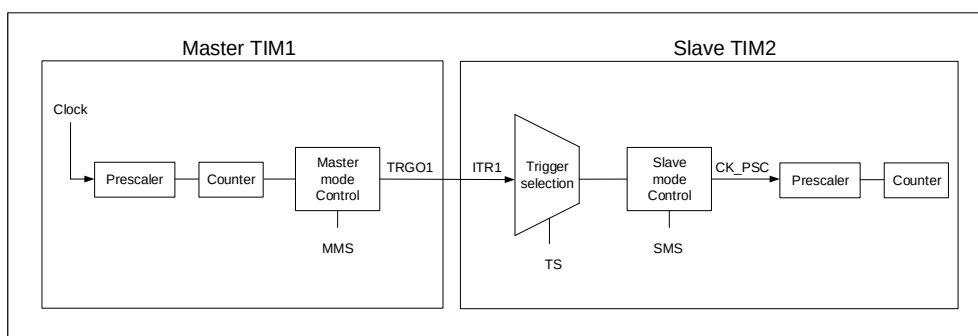


图 13-32 主/从定时器范例

1. 3. 16. 1 使用一个定时器使能另一个定时器

在这个例子中，定时器 2 的使能由定时器 1 的输出比较控制。参考图 130 的连接。只当定时器 1 的 OC1REF 为高时，定时器 2 才对分频后的内部时钟计数。两个定时器的时钟频率都是由预分频器对 CK_INT 除以 3($f_{CK\_CNT}=f_{CK\_INT}/3$)得到。

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

- ◆ 配置定时器 1 的 OC1REF 波形(TIM1_CCMR1 寄存器)
- ◆ 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- ◆ 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)
- ◆ 置 TIM2_CR1 寄存器的 CEN=1 以使能定时器 2
- ◆ 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1

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

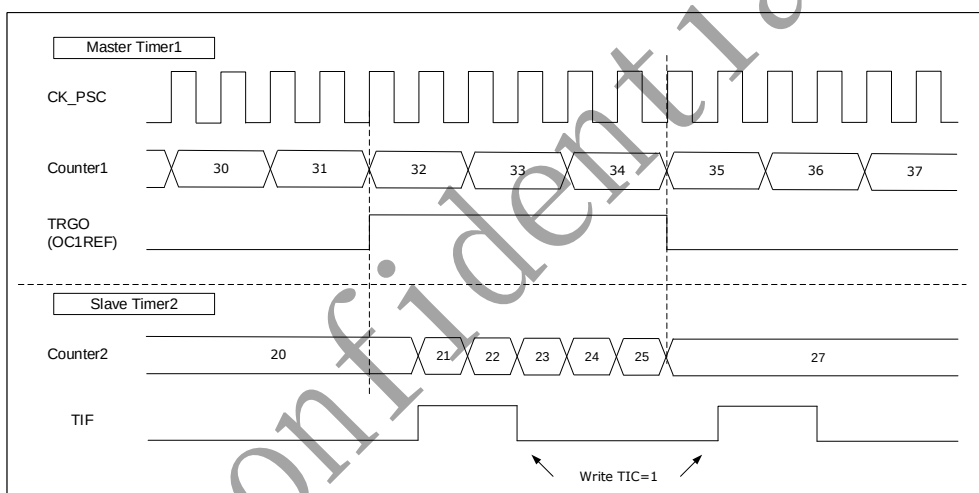


图 13-33 门控从定时器使用主定时器 OC1REF

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

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

- ◆ 配置定时器 1 为主模式，送出输出比较 1 参考信号(OC1REF)做为触发输出(TIM1_CR2 寄存器的 MMS=100)。
- ◆ 配置定时器 1 的 OC1REF 波形(TIM1_CCMR1 寄存器)。
- ◆ 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- ◆ 配置定时器 2 为门控模式(TIM2_SMCR 寄存器的 SMS=101)

- ◆ 置 TIM1_EGR 寄存器的 UG=1, 复位定时器 1。
- ◆ 置 TIM2_EGR 寄存器的 UG=1, 复位定时器 2。
- ◆ 写'0xE7'至定时器 2 的计数器(TIM2_CNT), 初始化它为 0xE7。
- ◆ 置 TIM2_CR1 寄存器的 CEN=1 以使能定时器 2。
- ◆ 置 TIM1_CR1 寄存器的 CEN=1 以启动定时器 1。
- ◆ 置 TIM1_CR1 寄存器的 CEN=0 以停止定时器 1。

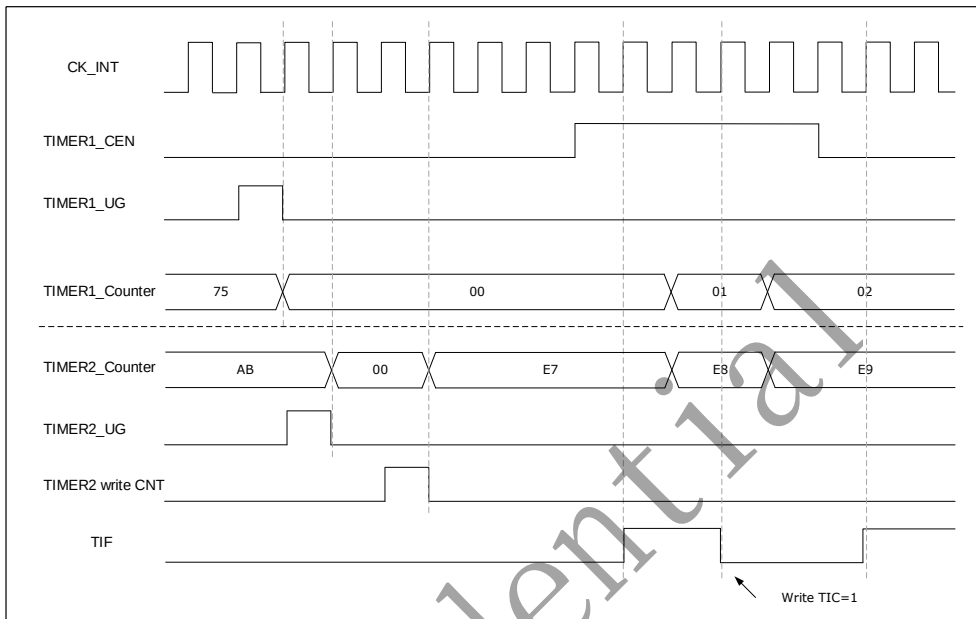


图 13-34 通过使能定时器 1 可以控制定时器 2

1. 3. 16. 2 使用一个定时器去启动另一个定时器

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

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

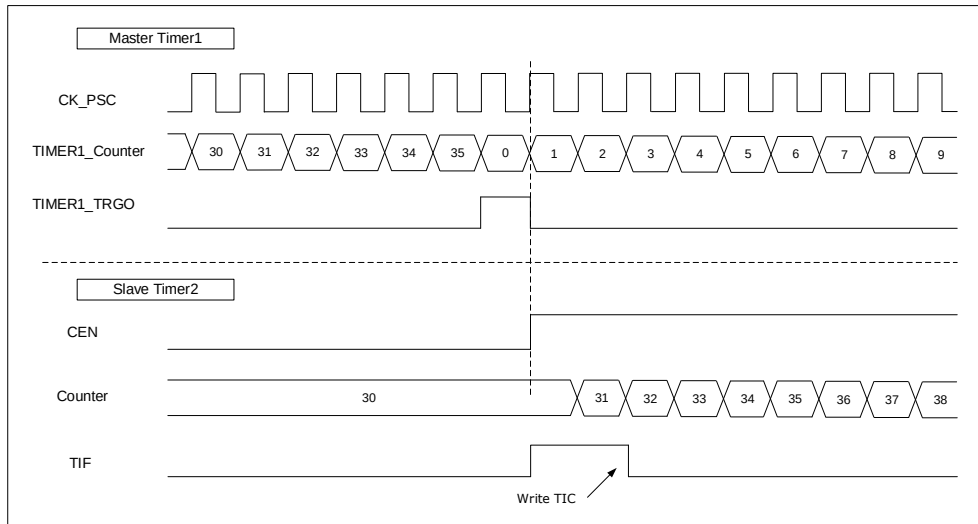


图 13-35 触发中从定时器使用主定时器更新事件

在上图例子中，可以在启动计数之前初始化两个计数器。上图显示在上上图相同配置情况下，使用触发模式而不是门控模式(TIM2_SMCR 寄存器的 SMS=110)的动作。

1. 3. 16. 3 使用一个外部触发同步地启动 2 个定时器

这个例子中当定时器 1 的 TI1 输入上升时使能定时器 1，使能定时器 1 的同时使能定时器 2，参见图 130。为保证计数器的对齐，定时器 1 必须配置为主/从模式(对应 TI1 为从，对应定时器 2 为主)：

- ◆ 配置定时器 1 为主模式，送出它的使能做为触发输出(TIM1_CR2 寄存器的 MMS=001)。
- ◆ 配置定时器 1 为从模式，从 TI1 获得输入触发(TIM1_SMCR 寄存器的 TS='101')。
- ◆ 配置定时器 1 为触发模式(TIM1_SMCR 寄存器的 SMS='110')。
- ◆ 配置定时器 1 为主/从模式，TIM1_SMCR 寄存器的 MSM='1'。
- ◆ 配置定时器 2 从定时器 1 获得输入触发(TIM2_SMCR 寄存器的 TS=000)
- ◆ 配置定时器 2 为触发模式(TIM2_SMCR 寄存器的 SMS='110')。

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

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

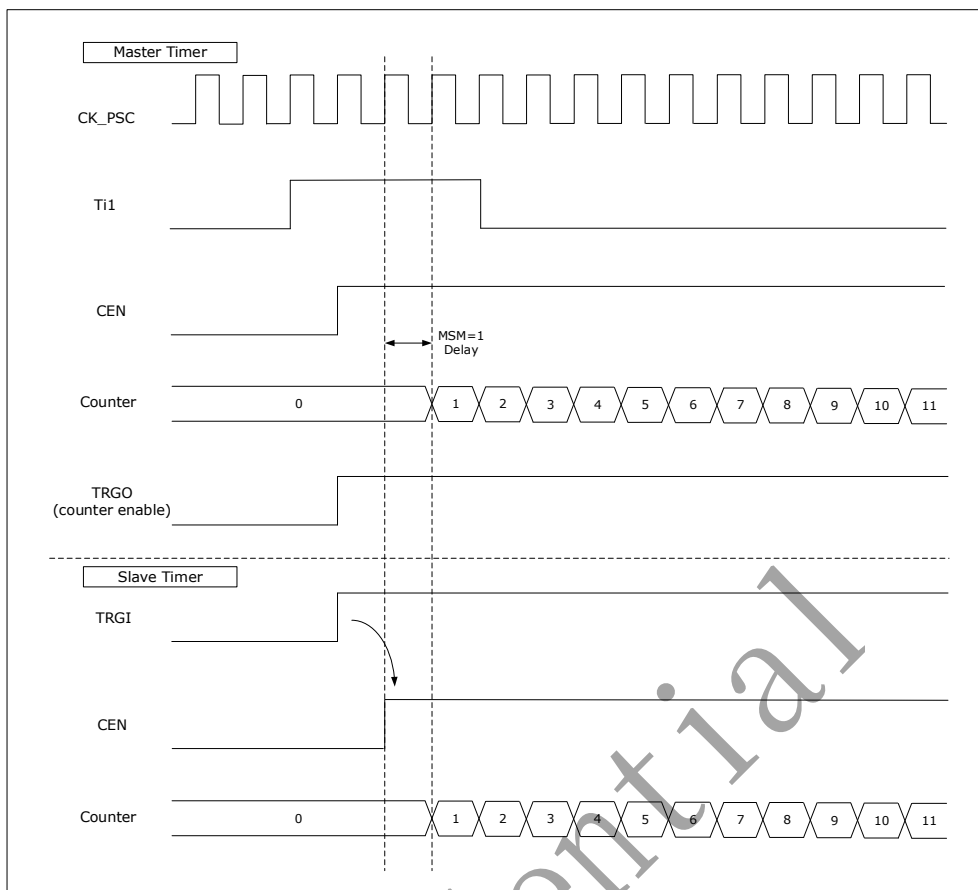


图 13-36 使用定时器 1 的 Ti1 输入触发定时器 1 和定时器 2

1. 3. 17 除错模式

当微控制器进入除错模式(MCU 中心终止),GPTIMC 计数器停止计数

1. 4 GPTIMC 寄存器描述

1. 4. 1 控制寄存器 1 (GPTIMC_CR1)

		Address offset:		0x00											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		CMPSEL[2:0]			CMPSELP		CKD[1:0]		ARPE		Reserved		OPM	URS	CEN

Name	Bit	Type	Reset value	Description
Reserved	31:12	N/A	0x00	保留, 保持复位值
CMPSEL	11:10	R/W	0x0	CMPSEL: 比较输入选择
				00: 比较输入 0
				10: 比较输入 1

				11: 比较输入 2
				11: 比较输入 3
CKD	9:8	R/W	0x0	CKD: 时钟分频因子 这 2 位定义在定时器时钟 (CK_INT) 频率、死区时间和由死区发生器与数字滤波器 (ETR, TIx) 所用的采样时钟之间的分频比例。 00: $tDTS=tCK\_INT$ 01: $tDTS=2*tCK\_INT$ 10: $tDTS=4*tCK\_INT$ 11: 保留, 不要使用这个配置
ARPE	7	R/W	0x0	ARPE: 自动重载预装使能位 0: GPTIM_ARR 寄存器没有缓存 1: GPTIM_ARR 寄存器有缓存
Reserved	6:4	N/A	0x0	保留, 保持复位值
OPM	3	R/W	0x0	OPM: 单脉冲模式 0: 在发生更新事件时, 计数器不停止 1: 在发生下一次更新事件 (清除 CEN 位) 时, 计数器停止
URS	2	R/W	0x0	URS: 更新请求源 软件通过该位选择 UEV 事件的源 0: 如果使能了更新中断或 DMA 请求, 则下述任一事件产生更新中断或 DMA 请求: - 计数器溢出 / 下溢 - 设置 UG 位 - 从模式控制器产生的更新 1: 如果使能了更新中断或 DMA 请求, 则只有计数器溢出 / 下溢才产生更新中断或 DMA 请求.
UDIS	1	R/W	0x0	UDIS: 禁止更新 软件通过该位允许 / 禁止 UEV 事件的产生 0: 允许 UEV。更新 (UEV) 事件由下述任一事件产生: - 计数器溢出 / 下溢 - 设置 UG 位 - 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值 1: 禁 UEV。不产生更新事件, 影子寄存器 (ARR、PSC、CCRx) 保持它们的值。如果设置 UG 位或从模式控制器发出了一个硬件复位, 则计数器和预分频器被重新初始化
CEN	0	R/W	0x0	CEN: 使能计数器 0: 计数器禁止 1: 计数器使能 注: 外接时钟、门控模式和编码器模式只能在软件设置 CEN 位之后进行工作。而触发模式下硬件可以自动设置 CEN 位

1. 4. 2 控制寄存器 2 (GPTIMC_CR2)

		Address offset:		0x04											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					OIS2	OIS1N	OIS1	Reserved	MMS[2:0]			CCDS	CCUS	Reserved	CCPC

Name	Bit	Type	Reset value	Description
Reserved	31:11	N/A	0x00	保留, 保持复位值
OIS2	10	R/W	0x0	OIS2: 输出空闲状态 2 (OC2 输出) 参照 OIS1 描述
OIS1N	9	R/W	0x0	OIS1N: 输出空闲状态 1 (OC1N 输出) 0: 当 MOE=0 时, 在死区时间之后 OC1N=0 1: 当 MOE=0 时, 在死区时间之后 OC1N=1 注: 已经设置了 LOCK(GPTIM_BKR 寄存器 LOCK 位)级别 1、2 或 3 后, 该位不能被修改
OIS1	8	R/W	0x0	OIS1: 输出空闲状态 1 (OC1 输出) 0: 当 MOE=0 时, 如果完成 OC1N, 则死区后 OC1=0 1: 当 MOE=0 时, 如果完成 OC1N, 则死区后 OC1=1 注: 已经设置了 LOCK(GPTIM_BKR 寄存器 LOCK 位)级别 1、2 或 3 后, 该位不能被修改
Reserved	7	N/A	0x0	保留, 保持复位值
MMS[2:0]	6:4	R/W	0x0	MMS: 主模式选择 这 3 位用于选择在主模式下送到从定时器的同步信息 (TRGO)。可能的组合如下: 000: 复位 - GPTIM_EGR 寄存器的 UG 位被用于作为触发输出 (TRGO)。如果是触发输入产生的复位(从模式控制器处于复位模式), 则 TRGO 上的信号相对实际的复位会有一个延迟 001: 使能 - 计数器使能信号 CNT_EN 被用于作为触发输出 (TRGO)。可用于同时启动多个定时器或控制在一段时间内使能从定时器。在门控模式下, 计数器使能信号是 CEN 控制位和的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时, TRGO 上会有一个延迟, 除非选择了主/从模式(见 GPTIM_SMCR 寄存器中 MSM 位的描述)。 010: 更新-更新事件被选为触发输入 (TRGO) 例如, 一个主定时器的时钟可以被用作一个从定时器的预分频器。 011: 比较脉冲-在发生一次捕获或一次比较成功时, 当要设置 CC1IF 标志时 (即使它已经为高), 触发输出送出一个正脉冲 (TRGO)。 100: 比较 - OC1REF 信号被用于作为触发输出 (TRGO) 101: 比较 - OC2REF 信号被用于作为触发输出 (TRGO) 110: 比较 - OC3REF 信号被用于作为触发输出 (TRGO) 111: 比较 - OC4REF 信号被用于作为触发输出 (TRGO)
CCDS	3	R/W	0x0	CCDS: 捕获 / 比较的 DMA 选择 0: 当发生 CCx 事件时, 送出 CCx 的 DMA 请求 1: 当发生更新事件时, 送出 CCx 的 DMA 请求
CCUS	2	R/W	0x0	CCUS: 捕获 / 比较控制更新选择 0: 如果捕获/比较控制位是预装载的(CCPC=1), 只能通过设置 COMG 位更新它们 1: 如果捕获/比较控制位是预装载的(CCPC=1), 可以通过设置 COMG 位或 TRGI 上的一个上升沿更新它们 注: 该位只对具有互补输出的通道起作用
Reserved	1	N/A	0x0	保留, 保持复位值
CCPC	0	R/W	0x0	CCPC: 捕获 / 比较预装载控制位 0: CCxE, CCxNE 和 OCxM 位不是预装载的 1: CCxE, CCxNE 和 OCxM 位是预装载的, 设置该位后, 只能在发生通信(COM)事件 (COMG 设置或 TRGI 是检测到上升沿, 取决于 CCUS 位) 位时被更新。 注: 该位只对具有互补输出的通道起作用。

1. 4. 3 从模式控制寄存器(GPTIMC_SMCR)

		Address offset:		0x08											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								MSM	TS[2:0]			Reserved	SMS[2:0]		

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留, 保持复位值
MSM	7	R/W	0x0	MSM: 主 / 从 模 式 0: 无作用; 1: 触发输入(TRGI)上的事件被延迟了, 以允许在当前定时器与它的从定时器间的完美同步(通过 TRGO)。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。
TS	6:4	R/W	0x0	TS[2:0]: 触 发 选 择 这 3 位选择用于同步计数器的触发输入。 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(TI2FP2) 111: 外部触发输入(ETRF) 更多有关 ITRx 的细节, 参见 266 页表 74, GPTIM 内部触发的连接。 注: 这些位只能在未用到(如 SMS=000)时被改变, 以避免在改变时产生错误的边沿检测。
Reserved	3	N/A	0x0	保留, 必须始终为复位值
SMS	2:0	R/W	0x0	SMS: 从 模 式 选 择 当选择了外部信号, 触发信号(TRGI)的有效边沿与选中的外部输入极性相关(见输入控制寄存器和控制寄存器的说明) 000: 关闭从模式 - 如果 CEN=1, 则预分频器直接由内部时钟驱动。 001: 编码器模式 1 - 根据 TI1FP1 的电平, 计数器在 TI2FP2 的边沿向上/下计数。 010: 编码器模式 2 - 根据 TI2FP2 的电平, 计数器在 TI1FP1 的边沿向上/下计数。 011: 编码器模式 3 - 根据另一个信号的输入电平, 计数器在 TI1FP1 和 TI2FP2 的边沿向上/下计数。 100: 复位模式 - 选中的触发输入(TRGI)的上升沿重新初始化计数器, 并且产生一次更新寄存器。 101: 门控模式 - 当触发输入(TRGI)为高时, 计数器的时钟开启。一旦触发输入变为低, 则计数器停止(但不复位)。计数器的启动和停止都是受控的。 110: 触发模式 - 计数器在触发输入 TRGI 的上升沿启动(但不复位), 只有计数器的启动是受控的。 111: 外部时钟模式 1 - 选中的触发输入(TRGI)的上升沿驱动计数器。 注: 如果 TI1F_ED 被选为触发输入(TS=100)时, 不要使用门控模式。这是因为, TI1F_ED 在每次 TI1F 变化时输出一个脉冲, 然而门控模式是要检查触发输入的电平。

1.4.4 中断使能寄存器(GPTIMC_IER)

		Address offset:				0x0C									
		Reset value:				0x0000 0000									
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					CC2O IE	CC1O IE	Reserved	BIE	TIE	COMI E	Reserved		CC2I E	CC1I E	UIE

Name	Bit	Type	Reset value	Description
Reserved	31:11	N/A	0x00	保留, 保持复位值
CC2OIE	10	WO	0x0	CC2OIE: 捕捉 / 比较 2 重复捕捉中断使能 0: 无 1: CC2 重复捕捉中断使能.
CC1OIE	9	WO	0x0	CC1OIE: 捕捉 / 比较 1 重复捕捉中断使能 0: 无 1: CC1 重复捕捉中断使能.
Reserved	8	N/A	0x0	保留, 保持复位值
BIE	7	WO	0x0	BIE: 刹车中断使能 0: 无 1: 刹车中断使能.
TIE	6	WO	0x0	TIE: 触发中断使能 0: 无 1: 触发中断使能.
COMIE	5	WO	0x0	COMIE: COM 中断使能 0: 无 1: COM 中断使能.
Reserved	4:3	N/A	0x0	保留, 保持复位值
CC2IE	2	WO	0x0	CC2IE: 捕捉 / 比较 2 中断使能 0: 无 1: CC2 中断使能.
CC1IE	1	WO	0x0	CC1IE: 捕捉 / 比较 1 中断使能 0: 无 1: CC1 中断使能.
UIE	0	WO	0x0	UIE: 更新中断使能 0: 无 1: 更新中断使能.

1.4.5 中断禁止寄存器(GPTIMC_IDR)

		Address offset:				0x10									
		Reset value:				0x0000 0000									
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					CC2O ID	CC1O ID	Reserved	BID	TID	COMI D	Reserved		CC2I D	CC1I D	UID

Name	Bit	Type	Reset value	Description
Reserved	31:11	N/A	0x00	保留，保持复位值
CC2OID	10	WO	0x0	CC2OID: 捕捉 / 比较 2 重复捕捉中断禁止. 0: 无效果. 1: CC2 重复捕捉中断禁止.
CC1OID	9	WO	0x0	CC1OID: 捕捉 / 比较 1 重复捕捉中断禁止. 0: 无效果. 1: CC1 重复捕捉中断禁止.
Reserved	8	N/A	0x0	保留，保持复位值
BID	7	WO	0x0	BID: 刹车中断禁止. 0: 无效果. 1: 刹车中断禁止.
TID	6	WO	0x0	TID: 触发中断禁止. 0: 无效果. 1: 触发中断禁止.
COMID	5	WO	0x0	COMID: COM 中断禁止. 0: 无效果. 1: COM 中断禁止.
Reserved	4:3	N/A	0x0	保留，保持复位值
CC2ID	2	WO	0x0	CC2ID: 捕捉 / 比较 2 中断禁止. 0: 无效果. 1: CC2 中断禁止.
CC1ID	1	WO	0x0	CC1ID: 捕捉 / 比较 1 中断禁止. 0: 无效果. 1: CC1 中断禁止.
UID	0	WO	0x0	UID: 更新中断禁止. 0: 无效果. 1: 更新中断禁止.

1. 4. 6 中断有效状态寄存器(GPTIMC_IVS)

		Address offset:		0x14											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					CC2OIS	CC1OIS	Reserved	BIS	TIS	COMIS	Reserved	CC2IS	CC1IS	UIS	

Name	Bit	Type	Reset value	Description
Reserved	31:11	N/A	0x00	保留，保持复位值
CC2OIS	10	RO	0x0	CC2OIS: 捕捉 / 比较 2 重复捕捉中断有效状态. 0: CC2 重复捕捉中断禁止状态. 1: CC2 重复捕捉中断使能状态. 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
CC1OIS	9	RO	0x0	CC1OIS: 捕捉 / 比较 1 重复捕捉中断有效状态. 0: CC1 重复捕捉中断禁止状态. 1: CC1 重复捕捉中断使能状态. 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位

Reserved	8	N/A	0x0	保留，保持复位值
BIS	7	RO	0x0	BIS: 刹车中断有效状态 0: 刹车中断禁止状态 1: 刹车中断使能状态 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
TIS	6	RO	0x0	TIS: 触发中断有效状态 0: 触发中断禁止状态 1: 触发中断使能状态 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
COMIS	5	RO	0x0	COMIS: COM 中断有效状态 0: COM 中断禁止状态 1: COM 中断使能状态 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
Reserved	4:3	N/A	0x0	保留，保持复位值
CC2IS	2	RO	0x0	CC2IS: 捕捉 / 比较 2 中断有效状态 0: CC2 中断禁止状态 1: CC2 中断使能状态 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
CC1IS	1	RO	0x0	CC1IS: 捕捉 / 比较 1 中断有效状态 0: CC1 中断禁止状态 1: CC1 中断使能状态 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位
UIS	0	RO	0x0	UIS: 更新中断有效状态 0: 更新中断禁止状态 1: 更新中断使能状态 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位

1. 4. 7 原始中断标志寄存器(GPTIMC_RIF)

Address offset:		0x18													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					CC2OIF	CC1OIF	Reserved	BIF	TIF	COMIF	Reserved	CC2IF	CC1IF	UIF	

Name	Bit	Type	Reset value	Description
Reserved	31:11	N/A	0x00	保留，保持复位值
CC2OIF	10	RO	0x0	CC2OIF: 捕捉 / 比较 2 重复捕捉原始中断标志 参考 CC1OIF 描述
CC1OIF	9	RO	0x0	CC1OIF: 捕捉 / 比较 1 重复捕捉原始中断标志 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。 GPTIM_ICR 寄存器相应位软件写“1”可清除该标志 0：无重复中断产生 1：当 CC1IF 的状态已经为‘1’，计数器的值被捕获到 GPTIM_CCR1 寄存器。
Reserved	8	N/A	0x0	保留，保持复位值

BIF	7	RO	0x0	BIF: 刹车原始中断标志 一旦刹车输入有效, 由硬件对该位置'1'。如果刹车输入无效, GPTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0: 无刹车事件产生; 1: 刹车输入上检测到有效电平。
	6	RO	0x0	TIF: 触发原始中断标志 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'。GPTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0: 无触发器事件产生; 1: 触发中断等待响应。
COMIF	5	RO	0x0	COMIF: COM 原始中断标志 一旦产生 COM 事件(当捕获/比较控制位: CCxE、CCxNE、OCxM 已被更新)该位由硬件置'1'。GPTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0: 无 COM 事件产生; 1: COM 中断等待响应。
Reserved	4:3	N/A	0x0	保留, 保持复位值
CC2IF	2	RO	0x0	CC2IF: 捕捉 / 比较 2 原始中断标志 参考 CC1IF 描述
CC1IF	1	RO	0x0	CC1IF: 捕捉 / 比较 1 原始中断标志 如果通道 CC1 配置为输出模式: 当计数器值与比较值匹配时该位由硬件置 1, 但在中心对称模式下除外(参考 GPTIM_CR1 寄存器的 CMS 位)。它由软件清'0'。 0: 无匹配发生; 1: GPTIM_CNT 的值与 GPTIM_CCR1 的值匹配。当 GPTIM_CCR1 的内容大于 GPTIM_APR 的内容时, 在向上或向上/下计数模式时计数器溢出, 或向下计数模式时的计数器下溢条件下, CC1IF 位变高。 如果通道 CC1 配置为输入模式: 当捕获事件发生时该位由硬件置'1', GPTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0: 无输入捕获产生; 1: 计数器值被捕获至 GPTIM_CCR1(在 IC1 上检测到与所选极性相同的边沿)。
UIF	0	RO	0x0	UIF: 更新 CC2IF: 中断标志 当产生更新事件时该位由硬件置'1'。它由软件清'0'。 0: 无更新事件产生; 1: 更新中断等待响应。当寄存器被更新时该位由硬件置'1': —若 GPTIM_CR1 寄存器的 UDIS=0, 当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 —若 GPTIM_CR1 寄存器的 URS=0、UDIS=0, 当设置 GPTIM_EGR 寄存器的 UG=1 时产生更新事件, 通过软件对计数器 CNT 重新初始化时。 —若 GPTIM_CR1 寄存器的 URS=0、UDIS=0, 当计数器 CNT 被触发事件重新初始化时。

1. 4. 8 屏蔽中断标志寄存器(GPTIMC_IFM)

Address offset:		0x1C													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	CC2OIM	CC1OIM	Reserved	BIM	TIM	COMIM	Reserved	CC2IM	CC1IM	UIM

Name	Bit	Type	Reset value	Description
Reserved	31:11	N/A	0x00	保留，保持复位值
CC2OIM	10	RO	0x0	CC2OIM: 捕捉 / 比较 2 重复捕捉屏蔽中断标志 参考 CC1OIM 描述
CC1OIM	9	RO	0x0	CC1OIM: 捕捉 / 比较 1 重复捕捉屏蔽中断标志 仅当相应的通道被配置为输入捕获时，该标记可由硬件置 1。 GPTIM_ICR 寄存器相应位软件写“1”可清除该标志 0：无重复中断产生 1：当 CC1IF 的状态已经为‘1’，计数器的值被捕获到 GPTIM_CCR1 寄存器。
Reserved	8	N/A	0x0	保留，保持复位值
BIM	7	RO	0x0	BIM: 刹车屏蔽中断标志 一旦刹车输入有效，由硬件对该位置‘1’。如果刹车输入无效，GPTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0：无刹车事件产生； 1：刹车输入上检测到有效电平。
TIM	6	RO	0x0	TIM: 触发屏蔽中断标志 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’。GPTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0：无触发器事件产生； 1：触发中断等待响应。
COMIM	5	RO	0x0	COMIM: COM 屏蔽中断标志 一旦产生 COM 事件(当捕获/比较控制位：CCxE、CCxNE、OCxM 已被更新)该位由硬件置‘1’。GPTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0：无 COM 事件产生； 1：COM 中断等待响应。
Reserved	4:3	N/A	0x0	保留，保持复位值
CC2IM	2	RO	0x0	CC2IM: 捕捉 / 比较 2 屏蔽中断标志 参考 CC1IM 描述
CC1IM	1	RO	0x0	CC1IM : 捕捉 / 比较 1 屏蔽中断标志 如果通道 CC1 配置为输出模式： 当计数器值与比较值匹配时该位由硬件置 1，但在中心对称模式下除外(参考 GPTIM_CR1 寄存器的 CMS 位)。它由软件清‘0’。 0：无匹配发生； 1：GPTIM_CNT 的值与 GPTIM_CCR1 的值匹配。当 GPTIM_CCR1 的内容大于 GPTIM_APR 的内容时，在向上或向上/下计数模式时计数器溢出，或向下计数模式时的计数器下溢条件下，CC1IF 位变高 如果通道 CC1 配置为输入模式： 当捕获事件发生时该位由硬件置‘1’，GPTIM_ICR 寄存器相应位软件写“1”可清除该标志。 0：无输入捕获产生； 1：计数器值被捕获至 GPTIM_CCR1(在 IC1 上检测到与所选极性相同的边沿)。

UIM	0	RO	0x0	UIM: 更新屏蔽中断标志 当产生更新事件时该位由硬件置'1'。它由软件清'0'。
				0: 无更新事件产生; 1: 更新中断等待响应。当寄存器被更新时该位由硬件置'1': 一若 GPTIM_CR1 寄存器的 UDIS=0, 当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 一若 GPTIM_CR1 寄存器的 URS=0、UDIS=0, 当设置 GPTIM_EGR 寄存器的 UG=1 时产生更新事件, 通过软件对计数器 CNT 重新初始化时。 一若 GPTIM_CR1 寄存器的 URS=0、UDIS=0, 当计数器 CNT 被触发事件重新初始化时。

1. 4. 9 中断清除寄存器 (GPTIMC_ICR)

		Address offset:		0x20											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					CC2O C	CC1O C	Reser ved	BIC	TIC	COMI C	Reserved		CC2I C	CC1I C	UIC

Name	Bit	Type	Reset value	Description
Reserved	31:11	N/A	0x00	保留, 保持复位值
CC2OIC	10	W1C	0x0	CC2OIC: 捕捉 / 比较 2 重复捕捉中断清除寄存器 参考 CC1OIC 描述
CC1OIC	9	W1C	0x0	CC1OIC: 捕捉 / 比较 1 重复捕捉中断清除寄存器 0: 无 效 1: CC1OI 中断清除.
Reserved	8	N/A	0x0	保留, 保持复位值
BIC	7	W1C	0x0	BIC: 刹车 中 断 清 除 寄 存 器 0: 无 效 1: 刹车中断清除.
TIC	6	W1C	0x0	TIC: 触 发 中 断 清 除 寄 存 器 0: 无 效 1: 触发中断清除.
COMIC	5	W1C	0x0	COMIC: COM 中 断 清 除 寄 存 器 0: 无 效 1: COM 中断清除.
Reserved	4:3	N/A	0x0	保留, 保持复位值
CC2IC	2	W1C	0x0	CC2IC: 捕捉 / 比较 2 中 断 清 除 寄 存 器 参考 CC1IC 描述
CC1IC	1	W1C	0x0	CC1IC: 捕捉 / 比较 1 中 断 清 除 寄 存 器 0: 无 效 1: 捕捉/比较 1 中断清除.
UIC	0	W1C	0x0	UIC: 更 新 中 断 清 除 寄 存 器 0: 无 效 1: 更新中断清除(GPTIM_RIF).

1. 4. 10 事件产生寄存器(GPTIMC_EGR)

		Address offset:		0x24											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								BG	TG	COMG	Reserved		CC2G	CC1G	UG

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留，保持复位值
BG	7	WO	0x0	BG: 产生刹车事件 该位由软件置'1'，用于产生一个刹车事件，由硬件自动清'0'。 0: 无动作; 1: 产生一个刹车事件。此时 MOE=0、BIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
TG	6	WO	0x0	TG: 触发产生 该位由软件置'1'，用于产生一个事件，由硬件自动清'0'。 0: 无动作; 1: GPTIM_SR 中 TIF=1，若开启对应的中断和 DMA，则产生相应的中断和 DMA。
COMG	5	WO	0x0	COMG: 捕捉 / 比较控制更新产生 该位由软件置'1'，由硬件自动清'0'。 0: 无动作; 1: 当 CCPC=1，允许更新 CCxE、CCxNE、OCxM 位。 注：该位只对拥有互补输出的通道有效。
Reserved	4:3	N/A	0x0	保留，保持复位值
CC2G	2	WO	0x0	CC2G: 捕捉 / 比较 2 产生 参考 CC1G 描述
CC1G	1	WO	0x0	CC1G: 捕捉 / 比较 1 产生 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 1 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 GPTIM_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。
UG	0	WO	0x0	UG: 产生更新事件 该位由软件置'1'，由硬件自动清'0'。 0: 无动作; 1: 重新初始化计数器，并产生一个(寄存器)更新事件。注意预分频器的计数器也被清'0' (但是预分频系数不变)。若在中心对称模式下或 DIR=0(向上计数)则计数器被清'0'；若 DIR=1(向下计数)则计数器取 GPTIM_ARR 的值。

1. 4. 11 捕捉/比较模式寄存器 1 (GPTIMC_CCMR1)

		Address offset:			0x28										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	OC2M[2:0]			OC2 PE	OC2 FE	CC2S[1:0]		Reserved	OC1M[2:0]			OC1 PE	OC1 FE	CC1S[1:0]	
IC2F[3:0]			IC2PSC[1:0]		IC1F[3:0]			IC1PSC[1:0]							

输出比较模式

Name	Bit	Type	Reset value	Description
Reserved	31:15	N/A	0x00	保留，保持复位值
OC2CE	15	R/W	0x0	OC2CE: 输出比较 2 清除 0 使能 参考 OC1CE 描述
OC2M	14:12	R/W	0x0	OC2M: 输出比较 模式 2 参考 OC1M 描述
OC2PE	11	R/W	0x0	OC2PE: 输出比较 2 预装使能 参考 OC1PE 描述
OC2FE	10	R/W	0x0	OC2FE: 输出比较 2 快速使能 参考 OC1FE 描述
CC2S	9:8	R/W	0x0	CC2S: 捕捉 / 比较 2 选择 该位定义通道的方向(输入/输出)，及输入信号的选择： 00: CC2 通道被配置为输出； 01: CC2 通道被配置为输入，IC2 映射在 TI2 上； 10: CC2 通道被配置为输入，IC2 映射在 TI1 上； 11: CC2 通道被配置为输入，IC2 映射在 TRC 上。此模式仅工作在内部触发器输入 被选中时(由 GPTIM_SMCR 寄存器的 TS 位选择)。 注：CC2S 仅在通道关闭时(GPTIM_CCER 寄存器的 CC2E=0)才是可写的。
OC1CE	7	R/W	0x0	OC1CE: 输出比较 1 清除 0 使能 0: OC1REF 不受 ETRF 输入的影响； 1: 一旦检测到 ETRF 输入高电平，清除 OC1REF=0。

OC1M	6:4	R/W	0x0	OC1M: 输出比较模式 1 该 3 位定义了输出参考信号 OC1REF 的动作, 而 OC1REF 决定了 OC1、OC1N 的值。OC1REF 是高电平有效, 而 OC1、OC1N 的有效电平取决于 CC1P、CC1NP 位。 000: 冻结。输出比较寄存器 GPTIM_CCR1 与计数器 GPTIM_CNT 间的比较对 OC1REF 不起作用; 001: 匹配时设置通道 1 为有效电平。当计数器 GPTIM_CNT 的值与捕获/比较寄存器 1(GPTIM_CCR1)相同时, 强制 OC1REF 为高。 010: 匹配时设置通道 1 为无效电平。当计数器 GPTIM_CNT 的值与捕获/比较寄存器 1(GPTIM_CCR1)相同时, 强制 OC1REF 为低。 011: 翻转。当 GPTIM_CCR1=GPTIM_CNT 时, 翻转 OC1REF 的电平。 100: 强制为无效电平。强制 OC1REF 为低。 101: 强制为有效电平。强制 OC1REF 为高。 110: PWM 模式 1 — 在向上计数时, 一旦 GPTIM_CNT<GPTIM_CCR1 时通道 1 为有效电平, 否则为无效电平; 在向下计数时, 一旦 GPTIM_CNT>GPTIM_CCR1 时通道 1 为无效电平(OC1REF=0), 否则为有效电平(OC1REF=1)。 111: PWM 模式 2 — 在向上计数时, 一旦 GPTIM_CNT<GPTIM_CCR1 时通道 1 为无效电平, 否则为有效电平; 在向下计数时, 一旦 GPTIM_CNT>GPTIM_CCR1 时通道 1 为有效电平, 否则为无效电平。 注 1: 一旦 LOCK 级别设为 3(GPTIM_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2: 在 PWM 模式 1 或 PWM 模式 2 中, 只有当比较结果改变了或在输出比较模式从中冻结模式切换到 PWM 模式时, OC1REF 电平才改变。 注 3: 在通道有互补输出时, 此时被预装载。如果设置了 GPTIM_CR2 中的 CCPC 位, OC1M 的有效位只能在发生 COM 事件时才能从预装位得到新的值。
OC1PE	3	R/W	0x0	OC1PE: 输出比较 1 预装使能 0: 禁止 GPTIM_CCR1 寄存器的预装载功能, 可随时写入 GPTIM_CCR1 寄存器, 并且新写入的数值立即起作用。 1: 开启 GPTIM_CCR1 寄存器的预装载功能, 读写操作仅对预装载寄存器操作, GPTIM_CCR1 的预装载值在更新事件到来时被加载至当前寄存器中。 注 1: 一旦 LOCK 级别设为 3(GPTIM_BDTR 寄存器中的 LOCK 位)并且 CC1S=00(该通道配置成输出)则该位不能被修改。 注 2: 仅在单脉冲模式下(GPTIM_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
OC1FE	2	R/W	0x0	OC1FE: 输出比较 1 快速使能 该位用于加快 CC 输出对触发输入事件的响应。 0: CC1 的正常操作依赖于计数器与 CCR1 的值, 即使工作于触发器状态。当触发器的输入有一个有效沿时, 激活 CC1 输出的最小延时为 5 个时钟周期。 1: 输入到触发器的有效沿的作用就象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。采样触发器的有效沿和 CC1 输出间的延时被缩短为 3 个时钟周期。OCFE 只在通道被配置成 PWM1 或 PWM2 模式时起作用。
CC1S	1:0	R/W	0x0	CC1S: 捕捉 / 比较 1 选择 这 2 位定义通道的方向(输入/输出), 及输入脚的选择: 00: CC1 通道被配置为输出; 01: CC1 通道被配置为输入, IC1 映射在 TI1 上; 10: CC1 通道被配置为输入, IC1 映射在 TI2 上; 11: CC1 通道被配置为输入, IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 GPTIM_SMCR 寄存器的 TS 位选择)。 注: CC1S 仅在通道关闭时(GPTIM_CCER 寄存器的 CC1E=0)才是可写的。

输入捕捉模式

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留, 保持复位值
IC2F	15:12	R/W	0x0	IC2F: 输入捕捉 2 滤波器 参考 IC1F 描述
IC2PSC	11:10	R/W	0x0	IC2PSC: 输入捕捉 2 预分频器 参考 IC1PSC 描述
CC2S	9:8	R/W	0x0	CC2S: 捕捉 / 比较 2 选择 这 2 位定义通道的方向(输入/输出), 及输入脚的选择: 00: CC2 通道被配置为输出; 01: CC2 通道被配置为输入, IC2 映射在 TI2 上; 10: CC2 通道被配置为输入, IC2 映射在 TI1 上; 11: CC2 通道被配置为输入, IC2 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 GPTIM_SMCR 寄存器的 TS 位选择)。 注: CC2S 仅在通道关闭时(GPTIM_CCER 寄存器的 CC2E=0)才是可写的。
IC1F	7:4	R/W	0x0	IC1F: 输入捕捉 1 滤波器 这几位定义了 TI1 输入的采样频率及数字滤波器长度。数字滤波器由一个事件计数器组成, 它记录到 N 个事件后会产生一个输出的跳变: 0000: 没有过滤器, $f_{\text{SAMPLING}} = f_{\text{DTS}}$ 0001: $f_{\text{SAMPLING}} = f_{\text{CK\_INT}}, N = 2$ 0010: $f_{\text{SAMPLING}} = f_{\text{CK\_INT}}, N = 4$ 0011: $f_{\text{SAMPLING}} = f_{\text{CK\_INT}}, N = 8$ 0100: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 2, N = 6$ 0101: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 2, N = 8$ 0110: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 4, N = 6$ 0111: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 4, N = 8$ 1000: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 8, N = 6$ 1001: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 8, N = 8$ 1010: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 16, N = 5$ 1011: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 16, N = 6$ 1100: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 16, N = 8$ 1101: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 32, N = 5$ 1110: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 32, N = 6$ 1111: $f_{\text{SAMPLING}} = f_{\text{DTS}} / 32, N = 8$ 注意: 当 ICxF[3:0] = 1、2 或 3 时, 必须注意公式中 fDTS 用 CK_INT 代替。
IC1PSC	3:2	R/W	0x0	IC1PSC: 输入捕捉 1 预分频器 这 2 位定义了 CC1 输入(IC1)的预分频系数。一旦 CC1E=0(GPTIM_CCER 寄存器中), 则预分频器复位。 00: 无预分频器, 捕获输入口上检测到的每一个边沿都触发一次捕获; 01: 每 2 个事件触发一次捕获; 10: 每 4 个事件触发一次捕获; 11: 每 8 个事件触发一次捕获。
CC1S	1:0	R/W	0x0	CC1S: 捕捉/比较 1 选择 这 2 位定义通道的方向(输入/输出), 及输入脚的选择: 00: CC1 通道被配置为输出; 01: CC1 通道被配置为输入, IC1 映射在 TI1 上; 10: CC1 通道被配置为输入, IC1 映射在 TI2 上; 11: CC1 通道被配置为输入, IC1 映射在 TRC 上。此模式仅工作在内部触发器输入被选中时(由 GPTIM_SMCR 寄存器的 TS 位选择)。 注: CC1S 仅在通道关闭时(GPTIM_CCER 寄存器的 CC1E=0)才是可写的。

1. 4. 12 捕捉/比较使能寄存器(GPTIMC_CCER)

		Address offset:		0x30											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										CC2P	CC2E	CC1NP	CC1NE	CC1P	CC1E

Name	Bit	Type	Reset value	Description
Reserved	31:6	N/A	0x00	保留，保持复位值
CC2P	5	R/W	0x0	CC2P: 捕捉 / 比较 2 输出 极性 参考 CC1P 描述
CC2E	4	R/W	0x0	CC2E: 捕捉 / 比较 2 输出 使能 参考 CC1E 描述
CC1NP	3	R/W	0x0	CC1NP: 捕捉 / 比较 1 互补 输出 极性 CC1 通道配置为输出 0: OC1N 高电平有效; 1: OC1N 低电平有效。 CC1 通道配置为输入 本位用于和 CC1P 联合定义 TI1FP1 和 TI2FP1 的极性, 详见 CC1P 注: 在通道有互补输出时, 本位被预装载。如果寄存器 GPTIM_CR2 的 CCPC 被置 1, 则 CC1NP 的有效位从预装载复制的新值只有在通信事件发生时才起作用。 注: 一旦 LOCK 级别(GPTIM_BDTR 寄存器中的 LOCK 位)设为 3 或 2 且 CC1S=00(通道配置为输出)则该位不能被修改。
CC1NE	2	R/W	0x0	CC1NE: 捕捉 / 比较 1 互补 输出 使能 0: 关闭— OC1N 禁止输出, 因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 1: 开启— OC1N 信号输出到对应的输出引脚, 其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 注: 在通道有互补输出时, 本位被预装载。如果寄存器 GPTIM_CR2 的 CCPC 被置 1, 则 CC1NP 的有效位从预装载复制的新值只有在通信事件发生时才起作用。

CC1P	1	R/W	0x0	CC1P: 捕捉 / 比较 1 输出 极性 CC1 通道配置为输出: 0: OC1 高电平有效; 1: OC1 低电平有效。 CC1 通道配置为输入: CC1NP/CC1P 位选择在触发或捕捉模式下 TI1FP1 和 TI2FP1 的有效极性。 00: 非反相/上升沿 电路作用于 TIxFP1 的上升沿(在复位、外部时钟或触发模式下的捕捉或触发操作), TIxFP1 非反相(在门控模式或编码模式)。 01: 反相/下降沿电路作用于 TIxFP1 的下降沿(在复位、外部时钟或触发模式下的捕捉或触发操作), TIxFP1 反相(在门控模式或编码模式)。 00: 保留不用 11: 非反相/上升或下降沿电路作用于 TIxFP1 的上升沿和下降沿(在复位、外部时钟或触发模式下的捕捉或触发操作), TIxFP1 非反相(在门控模式)。 在编码模式下不能使用此配置。 注: 在通道有互补输出时, 本位被预装载。如果寄存器 GPTIM_CR2 的 CCPC 被置 1, 则 CC1NP 的有效位从预装载复制的新值只有在通信事件发生时才起作用。 注: 一旦 LOCK 级别(GPTIM_BDTR 寄存器中的 LOCK 位)设为 3 或 2, 则该位不能被修改。
CC1E	0	R/W	0x0	CC1E: 捕捉 / 比较 1 输出 使能 CC1 通道配置为输出: 0: 关闭— OC1N 禁止输出, 因此 OC1N 的电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 1: 开启— OC1N 信号输出到对应的输出引脚, 其输出电平依赖于 MOE、OSSI、OSSR、OIS1、OIS1N 和 CC1E 位的值。 CC1 通道配置为输入: 本位用于决定是否一个定时器值的捕捉要装载到捕捉/比较寄存器 1(GPTIM_CCR1)。 0: 捕捉禁止。 1: 捕捉允许。 注: 在通道有互补输出时, 本位被预装载。如果寄存器 GPTIM_CR2 的 CCPC 被置 1, 则 CC1NP 的有效位从预装载复制的新值只有在通信事件发生时才起作用。

Control bits					Output States ⁽¹⁾	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
1	X	0	0	0	输出禁止(不由定时器驱动) OCx=0, OCx_EN=1	输出禁止(不由定时器驱动) OCxN=0, OCxN_EN=1
		0	0	1	输出禁止(不由定时器驱动) OCx=0, OCx_EN=1	OCxREF + 极性 OCxN=OCxREFxorCCxNP, OCxN_EN=0
		0	1	0	OCxREF + 极性 OCx=OCxREFxorCCxP, OCx_EN=0	输出禁止(不由定时器驱动) OCxN=0, OCxN_EN=1
		0	1	1	OCREF + 极性 + 死区 OCx_EN=0	OCREF 互补输出 (not OCREF) + 极性 + 死区 OCxN_EN=0

Control bits					Output States ⁽¹⁾	
MOE bit	OSSI bit	OSSR bit	CCxE bit	CCxNE bit	OCx 输出状态	OCxN 输出状态
		1	0	0	输出禁止 (不由定时器驱动) OCx=CCxP, OCx_EN=1	输出禁止 (不由定时器驱动) OCxN=CCxNP, OCxN_EN=1
		1	0	1	Off-State (输出使能但无效状态) OCx=CCxP, OCx_EN=0	OCxREF + 极性 OCxN=OCxREFxorCCxNP, OCxN_EN=0
		1	1	0	OCxREF + 极性 OCx=OCxREFxorCCxP, OCx_EN=0	Off-State (输出使能但无效状态) OCxN=CCxNP, OCxN_EN=0
		1	1	1	OCREF + 极性 + 死区 OCx_EN=0	OCREF 互补输出 (not OCREF) + 极性 + 死区 OCxN_EN=0
0	0	X	0	0	输出禁止 (不由定时器驱动) OCx=CCxP, OCx_EN=1	输出禁止 (不由定时器驱动) OCxN=CCxNP, OCxN_EN=1
	0		0	1	输出禁止 (不由定时器驱动) 异步: OCx=CCxP, OCx_EN=1, OCxN=CCxNP, OCxN_EN=1. 若时钟存在: 经过一个死区时间后 OCx=OISx, OCxN=OISxN, 假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平	
	0		1	0		
	0		1	1	输出禁止 (不由定时器驱动) OCx=CCxP, OCx_EN=1	
	1		0	0		
	1		0	1	Off-State (输出使能但无效状态) 异步: OCx=CCxP, OCx_EN=0, OCxN=CCxNP, OCxN_EN=0	
	1		1	0		
	1		1	1	若时钟存在: 经过一个死区时间后 OCx=OISx, OCxN=OISxN, 假设 OISx 和 OISxN 并不都对应 OCx 和 OCxN 的有效电平。	

表 13-1 带刹车功能的互补输出通道 OCx 和 OCxN 的控制位

1. 4. 13 计数寄存器(GPTIMC_CNT)

		Address offset:		0x34											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
CNT	15:0	R/W	0x0	CNT: 计数器值 (counter)

1. 4. 14 预分频寄存器 (GPTIMC_PSC)

		Address offset:		0x38											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
PSC	15:0	R/W	0x0	PSC: 预分频值 预分频器的值 (Prescaler value) 计数器的时钟频率 (CK_CNT) 等于 $f_{CK\_PSC} / (PSC[15:0] + 1)$。每次当更新事件产生时，PSC 的值被装入当前预分频器寄存器；更新事件包括计数器被 TIM_EGR 的 UG 位清 '0' 或被工作在复位模式的从控制器清 '0'。

1. 4. 15 自动重装载寄存器 (GPTIMC_ARR)

		Address offset:		0x3C											
		Reset value:		0x0000 FFFF											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
ARR	15:0	R/W	0xFFFF	ARR: 自动重装载值 ARR 包含了将要装载入实际的自动重装载寄存器的值。当自动重装载的值为空时，计数器不工作。

1. 4. 16 重复计数寄存器 (GPTIMC_RCR)

		Address offset:		0x40											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								REP[7:0]							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留，保持复位值
REP	7:0	R/W	0x0	REP: 重 复 计 数 器 值 预装载寄存器被使能后，这些位允许用户设置比较寄存器的更新速率(即周期性地从预装载寄存器传输到当前寄存器)；如果允许产生更新中断，则会同时影响产生更新中断的速率。 每次向下计数器 REP_CNT 达到 0，会产生一个更新事件并且计数器 REP_CNT 重新从 REP 值开始计数。由于 REP_CNT 只有在周期更新事件 U_RC 发生时才重载 REP 值，因此对 GPTIM_RCR 寄存器写入的新值只在下次周期更新事件发生时才起作用。 这意味着在 PWM 模式中，(REP+1) 对应着： - 在边沿对齐模式下，PWM 周期的数目； - 在中心对称模式下，PWM 半周期的数目；

1. 4. 17 捕捉/比较寄存器 1 (GPTIMC_CCR1)

		Address offset:		0x44											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR1[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
CCR1	15:0	R/W	0x0	CCR1: 捕 捉 / 比 较 通 道 1 的 值 若 CC1 通道配置为输出： CCR1 决定了装入当前捕获/比较 1 寄存器的值(预装载值)。 如果在 GPTIM_CCMR1 寄存器(OC1PE 位)中未选择预装载功能，写入的数值会立即传输至当前寄存器中。否则只有当更新事件发生时，此预装载值才传输至当前捕获/比较 1 寄存器中。当前捕获/比较寄存器参与同计数器 GPTIM_CNT 的比较，并在 OC1 端口上产生输出信号。 若 CC1 通道配置为输入： CCR1 包含了由上一次输入捕获 1 事件(IC1)传输的计数器值。

1. 4. 18 捕捉/比较寄存器 2 (GPTIMC_CCR2)

		Address offset:		0x48											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCR2[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
CCR2	15:0	R/W	0x0	CCR2: 捕捉 / 比较通道 2 的值 参考 CCR1 描述

1. 4. 19 刹车和死区寄存器(GPTIMC_BDTR)

		Address offset:		0x54											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOE	AOE	BKP	BKE	OSSR	OSSI	LOCK[1:0]			DTG[7:0]						

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
MOE	15	R/W	0x0	MOE: 主输出使能 一旦刹车输入有效，该位被硬件异步清'0'。根据 AOE 位的设置值，该位可以由软件清'0'或被自动置1。它仅对配置为输出的通道有效。 0: 禁止 OC 和 OCN 输出或强制为空闲状态 1: 如果设置了相应的使能位(GPTIM_CCER 寄存器的 CCxE、CCxNE 位)，则开启 OC 和 OCN 输出
AOE	14	R/W	0x0	AOE: 自动输出使能 0: MOE 只能被软件置'1' 1: MOE 能被软件置'1'，或在下一个更新事件被自动置'1' (如果刹车输入无效) 注：一旦 LOCK 级别(GPTIM_BDTR 寄存器中的 LOCK 位)设为'1'，则该位不能被修改。
BKP	13	R/W	0x0	BKP: 刹车输入极性 0: 刹车输入低电平有效 1: 刹车输入高电平有效 注：一旦 LOCK 级别(GPTIM_BDTR 寄存器中的 LOCK 位)设为'1'，则该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用
BKE	12	R/W	0x0	BKE: 刹车使能 0: 刹车输入禁止 (BRK 和 CCS 时钟失效事件) 1: 刹车输入允许 (BRK 和 CCS 时钟失效事件) 注：一旦 LOCK 级别(GPTIM_BDTR 寄存器中的 LOCK 位)设为'1'，则该位不能被修改。 注：任何对该位的写操作都需要一个 APB 时钟的延迟以后才能起作用。

OSSR	11	R/W	0x0	OSSR: 运行模式下“关闭状态”选择 该位用于当 MOE=1 且通道为互补输出时。没有互补输出的定时器中不存在 OSSR 位。 0: 当定时器不工作时, 禁止 OC/OCN 输出 (OC/OCN 使能输出信号=0)。 1: 当定时器不工作时, 一旦 CCxE=1 或 CCxNE=1, OC/OCN 使能并输出无效电平, 然后置 OC/OCN 使能输出信号=1 注: 一旦 LOCK 级别(GPTIM_BDTR 寄存器中的 LOCK 位)设为'2', 则该位不能被修改。
	10	R/W	0x0	OSSI: 运行模式下“空闲状态”选择 该位用于当 MOE=0 时通道为输出。 0: 当定时器不工作时, 禁止 OC/OCN 输出 (OC/OCN 使能输出信号=0)。 1: 当定时器不工作时, 一旦 CCxE=1 或 CCxNE=1, OC/OCN 被强制输出空闲电平, 置 OC/OCN 使能输出信号=1 注: 一旦 LOCK 级别(GPTIM_BDTR 寄存器中的 LOCK 位)设为'2', 则该位不能被修改。
	9:8	R/W	0x0	LOCK[1:0]: 锁定设置 该位为防止软件错误而提供写保护。 00: 锁定关闭, 寄存器无写保护。 01: 锁定级别 1, 不能写入 GPTIM_BDTR 寄存器的 DTG、BKE、BKP、AOE 位和 GPTIM_CR2 寄存器的 OISx/OISxN 位 10: 锁定级别 2, 不能写入锁定级别 1 中的各位, 也不能写入 CC 极性位 (一旦相关通道通过 CCxS 位设为输出, CC 极性位是 GPTIM_CCER 寄存器的 CCxP/CCxN 位) 以及 OSSR/OSSI 位。 11: 锁定级别 3, 不能写入锁定级别 2 中的各位, 也不能写入 CC 控制位 (一旦相关通道通过 CCxS 位设为输出, CC 控制位是 GPTIM_CCMRx 寄存器的 OCxM/OCxPE 位)。 注: 在系统复位后, 只能写一次 LOCK 位, 一旦写入 GPTIM_BDTR 寄存器, 则其内容冻结直至复位
DTG[7:0]	7:0	R/W	0x0	DTG[7:0]: 死区发生器设置 这些位定义了插入互补输出之间的死区持续时间。假设 DT 表示其持续时间。 DTG[7:5]=0xx => DT=DTG[7:0]x tdtg with tdtg=tDTS. DTG[7:5]=10x => DT=(64+DTG[5:0])xtdtg with Tdtg=2xtDTS. DTG[7:5]=110 => DT=(32+DTG[4:0])xtdtg with Tdtg=8xtDTS. DTG[7:5]=111 => DT=(32+DTG[4:0])xtdtg with Tdtg=16xtDTS. 例: 若 Tdts=125ns(8MHZ), 可能的死区时间为: 0 到 15875ns, 若步长时间为 125ns; 16us 到 31750ns, 若步长时间为 250ns; 32us 到 63us, 若步长时间为 1us; 64us 到 126us, 若步长时间为 2us; 注: 一旦 LOCK 级别(GPTIM_BDTR 寄存器中的 LOCK 位) 设为 1、2 或 3, 则不能修改这些位。

1. 4. 20 DMA 事件使能寄存器(GPTIMC_DMAEN)

Address offset:		0x58													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	TDE	COMDE	Reserved	CC2DE	CC1DE	UDE

Name	Bit	Type	Reset value	Description
Reserved	31:7	N/A	0x00	保留，保持复位值
TDE	6	R/W	0x0	TDE: 触发中断 DMA 请求使能 0: 触发中断 DMA 请求禁止。 1: 触发中断 DMA 请求使能。
COMDE	5	R/W	0x0	COMDE: COM 中断 DMA 请求使能 0: COM 中断 DMA 请求禁止。 1: COM 中断 DMA 请求使能。
Reserved	4:3	N/A	0x0	保留，保持复位值
CC2DE	2	R/W	0x0	CC2DE: 捕捉 / 比较 2 中断 DMA 请求使能 0: CC2 中断 DMA 请求禁止。 1: CC2 中断 DMA 请求使能。
CC1DE	1	R/W	0x0	CC1DE: 捕捉 / 比较 1 中断 DMA 请求使能 0: CC1 中断 DMA 请求禁止。 1: CC1 中断 DMA 请求使能。
UDE	0	R/W	0x0	UDE: 更新中断 DMA 请求使能 0: 更新中断 DMA 请求禁止。 1: 更新中断 DMA 请求使能。

第14章 基本定时器（BSTIM）

1.1 BSTIM 简介

基本定时器(BSTIM)由一可编程预除器驱动之 16 位自动重载计数器构成.其亦可用于多种用途,包含测量输入信号脉冲长度(输入捕获)或产生输出波形(输出比较,PWM).可使用定时预除器与 APB 时钟控制预除器来调校脉冲长度与波形周期,由数微秒至数毫秒。

高级控制(ADTIM),通用(GPTIMA,GPTIMB,GPTIMC)与基本(BASTIM)定时器皆为完全独立不共享任何时钟源

1.2 BSTIM 主要特性

- ◆ 16 位递增自动重载计数器。
- ◆ 16 位可编程预分频器,用于对计数器时钟频率进行分频(即运行时修改),分频系数介于 1 到 65536 之间。
- ◆ 同步化电路可控制定时器含外部信号与链接数个定时器
- ◆ 发生如下事件时生成中断/DMA 请求:
 - 更新: 计数器上溢/下溢,计数器初始化(透过软件或内部/外部触发)
- ◆ 外部时钟触发输入或 cycle-by-cycle 电流管理

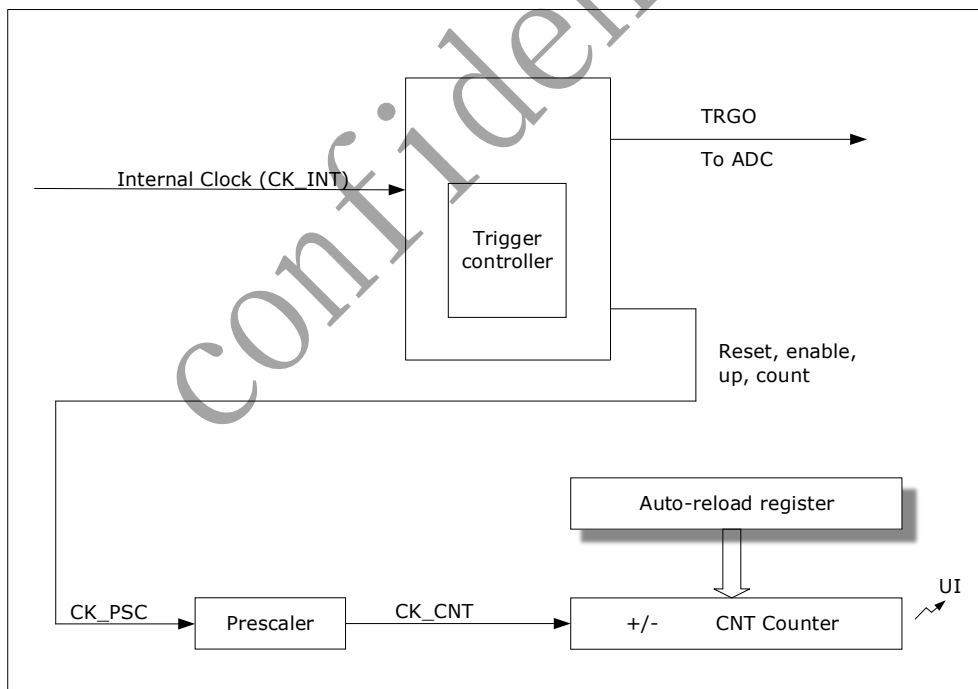


图 14-1 基本定时器块图

1.3 BSTIM 功能描述

1.3.1 定时器时基单位

编程基本定时器主区块为一带自动加载寄存器之 16 位计数器。计数器可递增计算。计数时钟

可被预除器区分。

计数器、自动重载寄存器与预除寄存器可由软件读写。此为计数器运作时之真正对等。

时基单位包含：

- ◆ 计数寄存器 (BSTIM_CNT)
- ◆ 预除寄存器 (BSTIM_PSC)
- ◆ 自动重载寄存器 (BSTIM_ARR)

自动重载寄存器已被预载。由自动重载寄存器写入或读出以存取预载寄存器。预载寄存器的内容皆被永久传输至影子寄存器或是每个更新事件(UEV)，取决于 BSTIM_CR1 寄存器之自动预载使能位(ARPE)。当计数器到达上溢(或向下计数时的下溢)以及若 UDIS 位同等于 BSTIM_CR1 寄存器的 0，更新事件即被送出。更新事件其亦可由软件产生。更新事件的产生在各自设置中被描述。

计数器是由预除器输出 CK_CNT 计时，仅当 BSTIM_CR1 寄存器之计数器使能位(CEN)被设置时被使能(参照属模式控制器描述以取得更多计数器使能信息)。

注:实际计数器使能信号 CNT_EN 被设置为 BSTIM_CR1 寄存器 CEN 位后的 1 个时钟周期

预除器描述

预除器可将计数器时钟频率依照 1 至 65536 间之任意因子分频。基于一 16 位寄存器控制之 16 位计数器(BSTIM_PSC 寄存器内)。因其具有缓冲功能，可于运行时被变更。新的预除器参数于下一次更新事件到达前被采用。预除计数器仅能向下计数。预除器被设定后，一旦预除计数器到达 0，计数器则会向上或向下计数，此外计数器会随着 APB 时钟一同更新。图 2 与图 3 标示当预除器于运作时变更计数器之行为范例。

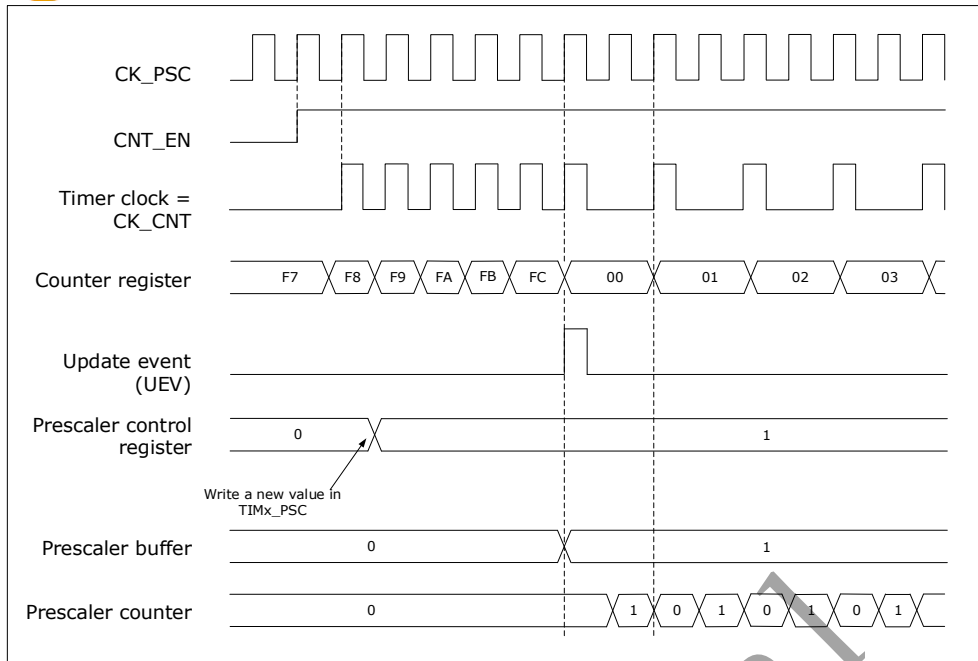


图 14-2 预除器区隔由 1 变更为 2 之计数器时序图

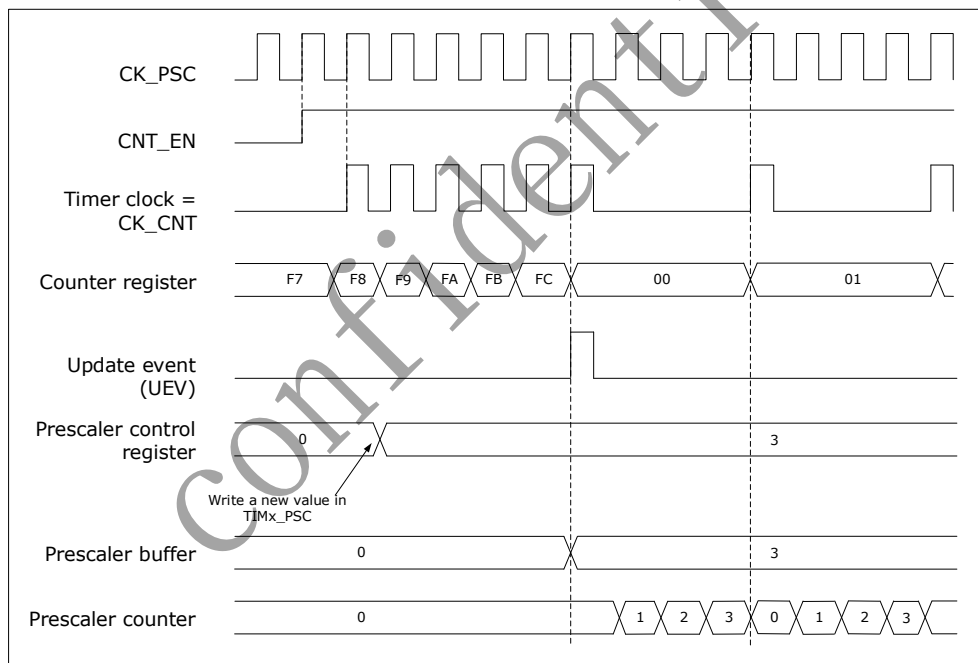


图 14-3 预除器区隔由 1 变更为 4 之计数器时序图

1.3.2 计数器模式

递增计数模式

在递增计数模式，计数器由 0 计算至自动加载值(BSTIM_ARR 寄存器内容)由 0 重新启动并产生一计数器上溢事件。

若重复计数器被使用，更新事件(UEV)会在向上计数依重复计数器(BSTIM_RCR)编程次数重复后产生。此外更新事件会于每一计数器上溢产生。

更新事件可于每一计数器下溢或设置 BSTIM_EGR 寄存器 UG 位产生(透过软件或从模式控制

器)。

UEV 事件可由软件透过设置 BSTIM_CR1 寄存器之 UDIS 位以关闭。这是为了避免更新写入新值至预载寄存器时，影子寄存器也一并更新。接着直到 UDIS 位被写 0 皆无更新事件产生。但计数器会由 0 重启，预除器之计数器亦同(但除频率不会变更)。

当更新事件发生时，所有寄存器皆被更新且更新标志(BSTIM_RIF 寄存器 UIF 位)皆被设定：

- ◆ 自动重载影子寄存器被更新为预载值(BSTIM_ARR)
- ◆ 预除器之缓冲器随着预载值重新加载(BSTIM_PSC 寄存器之内容)

下列图表为计数器在 BSTIM_ARR=0x36 时对应不同时钟频点之行为范例：

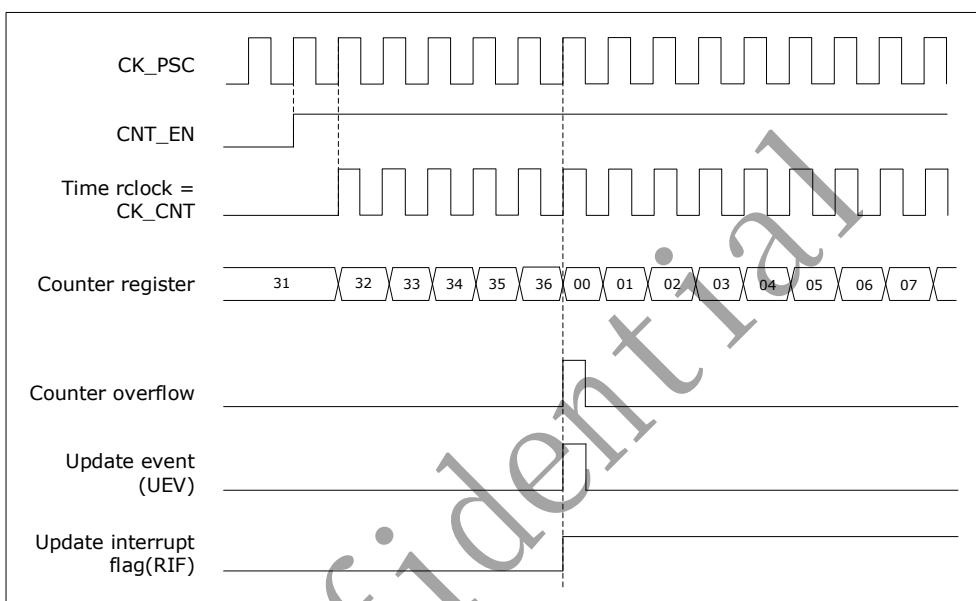


图 14-4 计数器时序图，内部时钟除以 1

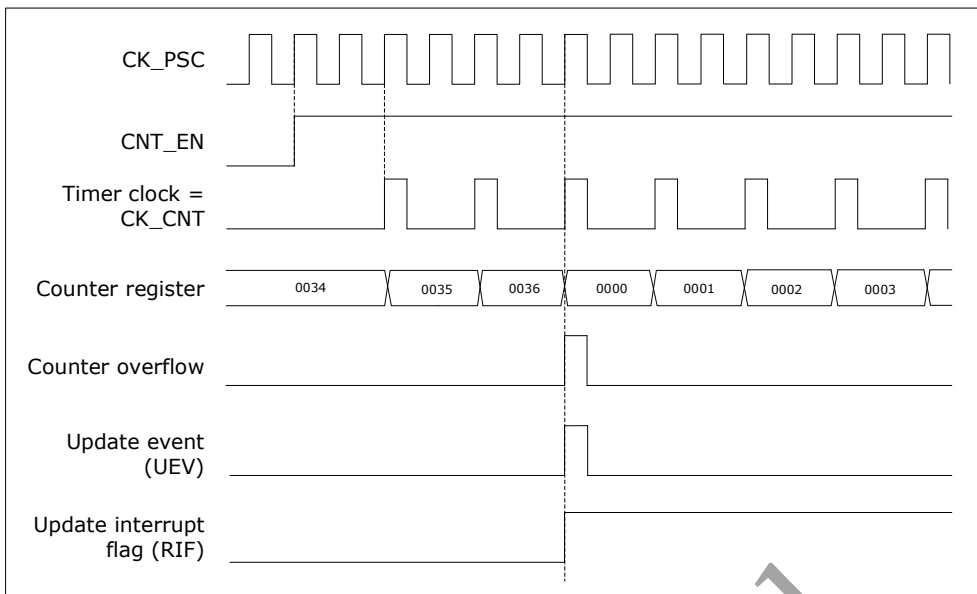


图 14-5 计数器时序图，内部时钟除以 2

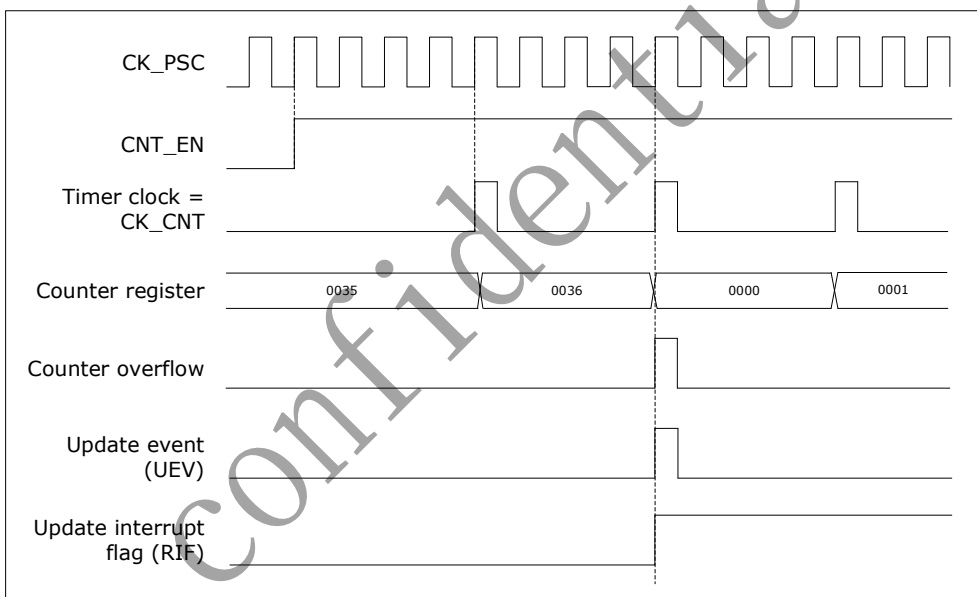


图 14-6 计数器时序图，内部时钟除以 4

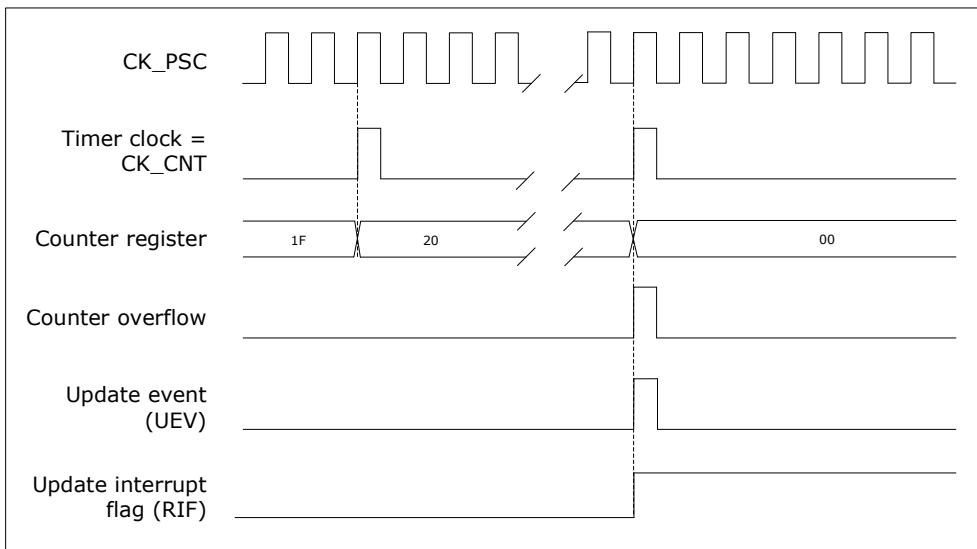


图 14-7 计数器时序图，内部时钟除以 N

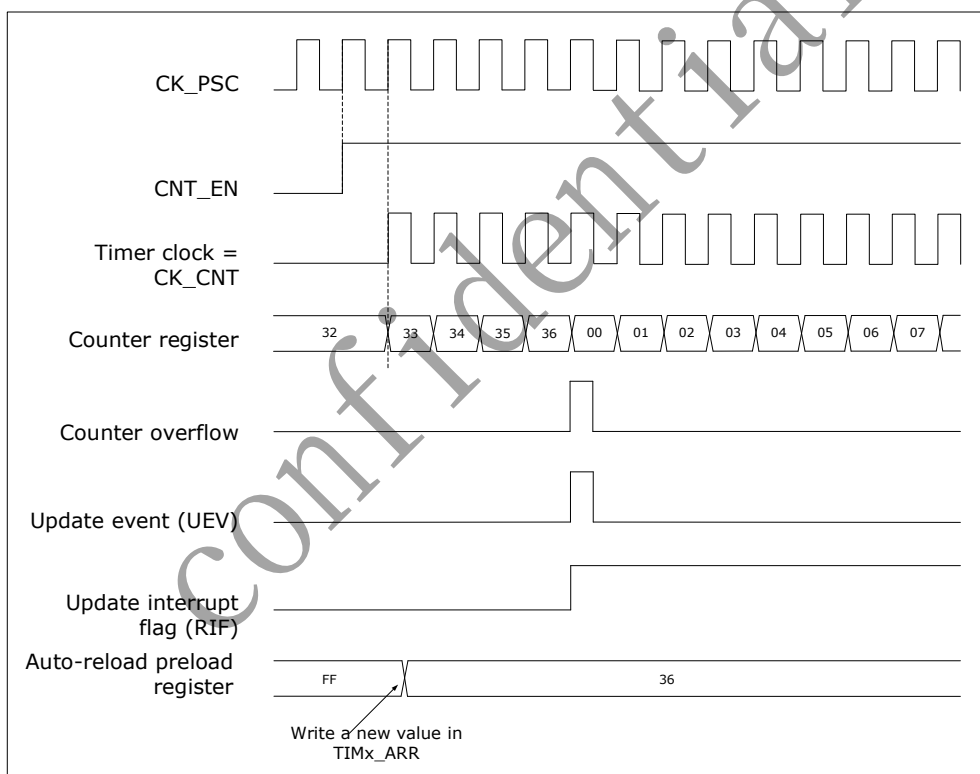


图 14-8 计数器时序图，当 ARPE=0 更新事件 (BSTIM_ARR 无预载)

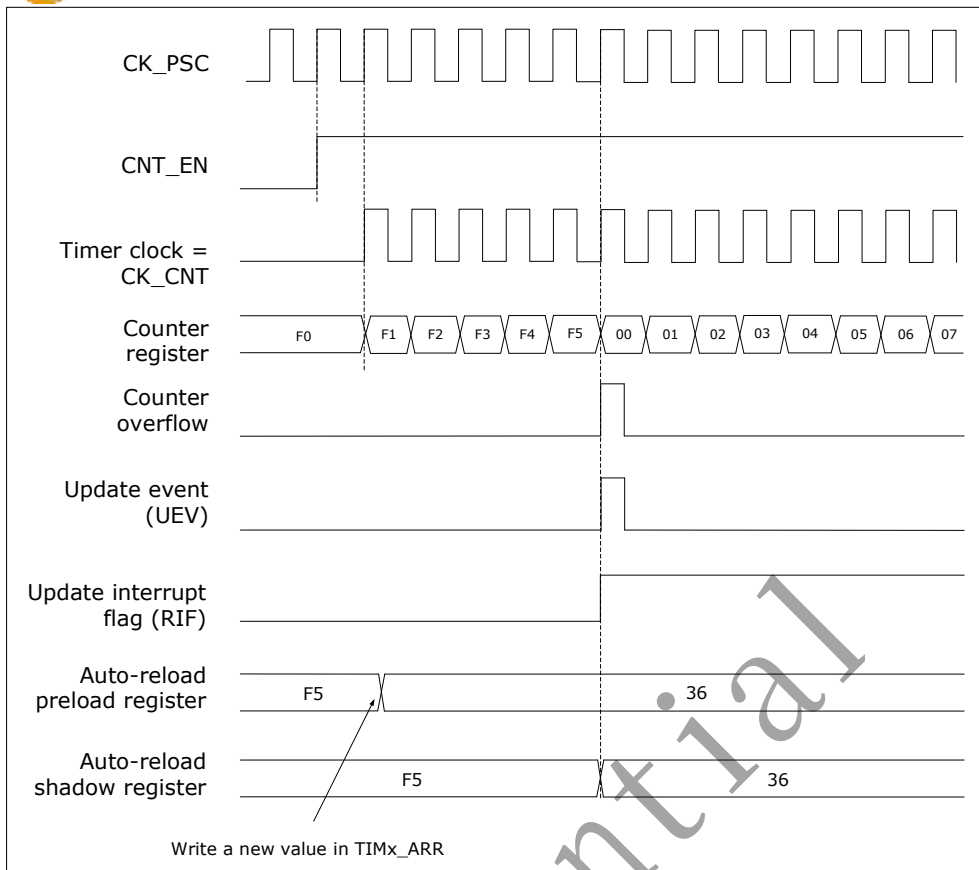


图 14-9 计数器时序图，当 ARPE=1 更新事件 (BSTIM_ARR 预载)

1.3.3 时钟源

CEN, DIR 位 (BSTIM_CR1 寄存器) 与 UG 位 (BSTIM_EGR 寄存器) 皆为实际控制位且仅可被软件变更 (UG 位仍被自动清除) 只要 CEN 位被写为 '1'，预除器之时钟即由内部时钟 CK_INT 提供。

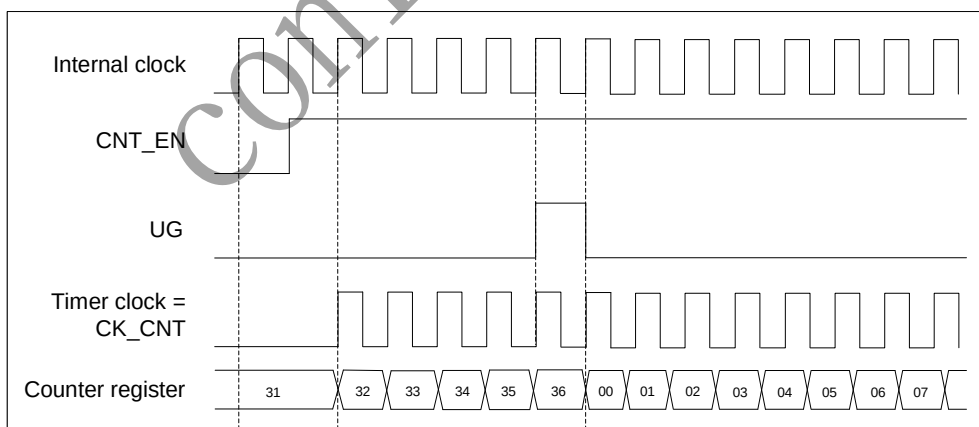


图 14-10 一般模式下之控制电路，内部时钟分频因子为 1

1.3.4 除错模式

当微控制器进入除错模式 (MCU 中心终止), BSTIM 计数器停止计数

1.4 特殊寄存器

1.4.1 控制寄存器 1 (BSTIM_CR1)

		Address offset:		0x00											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								ARPE	Reserved			OPM	URS	UDIS	CEN

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留，保持复位值
ARPE	7	R/W	0x0	ARPE: 自动重载预装使能位 0: BSTIM_ARR 寄存器没有缓存 1: BSTIM_ARR 寄存器有缓存
Reserved	6:4	N/A	0x0	保留，保持复位值
OPM	3	R/W	0x0	OPM:单脉冲模式 0: 在发生更新事件时，计数器不停止 1: 在发生下一次更新事件（清除 CEN 位）时，计数器停止
URS	2	R/W	0x0	URS: 更新请求源 软件通过该位选择 UEV 事件的源 0: 如果使能了更新中断或 DMA 请求，则下述任一事件产生更新中断或 DMA 请求： - 计数器溢出 / 下溢位 - 设置 UG 位 - 从模式控制器产生的更新 1: 如果使能了更新中断或 DMA 请求，则只有计数器溢出 / 下溢才产生更新中断或 DMA 请求。
UDIS	1	R/W	0x0	UDIS: 禁止更新 软件通过该位允许 / 禁止 UEV 事件的产生 0: 允许 UEV。更新 (UEV) 事件由下述任一事件产生： - 计数器溢出 / 下溢位 - 设置 UG 位 - 从模式控制器产生的更新 具有缓存的寄存器被装入它们的预装载值 1: 禁 UEV。不产生更新事件，影子寄存器 (ARR、PSC、CCRx) 保持它们的值。如果设置 UG 位或从模式控制器发出了一个硬件复位，则计数器和预分频器被重新初始化
CEN	0	R/W	0x0	CEN: 使能计数器 0: 计数器禁止 1: 计数器使能 注: 外接时钟、门控模式和编码器模式只能在软件设置 CEN 位之后进行工作。而触发模式下硬件可以自动设置 CEN 位

1. 4. 2 控制寄存器 2 (BSTIM_CR2)

		Address offset:		0x04											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										MMS[2:0]		Reserved			

Name	Bit	Type	Reset value	Description
Reserved	31:7	N/A	0x00	保留, 保持复位值
MMS[2:0]	6:4	R/W	0x0	MMS: 主 模 式 选 择 这 3 位用于选择在主模式下送到从定时器的同步信息 (TRGO)。可能的组合如下: 000: 复位 - BSTIM_EGR 寄存器的 UG 位被用于作为触发输出 (TRGO)。如果是触发输入产生的复位 (从模式控制器处于复位模式), 则 TRGO 上的信号相对实际的复位会有一个延迟 001: 使能 - 计数器使能信号 CNT_EN 被用于作为触发输出 (TRGO)。可用于同时启动多个定时器或控制在一段时间内使能从定时器。在门控模式下, 计数器使能信号是 CEN 控制位和的触发输入信号的逻辑或产生。当计数器使能信号受控于触发输入时, TRGO 上会有一个延迟, 除非选择了主/从模式 (见 BSTIM_SMCR 寄存器中 MSM 位的描述)。 010: 更新 - 更新事件被选为触发输入 (TRGO) 例如, 一个主定时器的时钟可以被用作一个从定时器的预分频器。 011: 比较脉冲 - 在发生一次捕获或一次比较成功时, 当要设置 CC1IF 标志时 (即使它已经为高), 触发输出送出一个正脉冲 (TRGO)。 100: 比较 - OC1REF 信号被用于作为触发输出 (TRGO) 101: 比较 - OC2REF 信号被用于作为触发输出 (TRGO) 110: 比较 - OC3REF 信号被用于作为触发输出 (TRGO) 111: 比较 - OC4REF 信号被用于作为触发输出 (TRGO)
Reserved	3:0	N/A	0x0	保留, 保持复位值

1. 4. 3 中断使能寄存器 (BSTIM_IER)

		Address offset:		0x0C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															UIE

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留, 保持复位值

UIE	0	WO	0x0	UIE: 更新中断使能. 0: 无中断使能. 1: 更新中断使能.

1.4.4 中断禁止寄存器(BSTIM_IDR)

		Address offset:		0x10											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															UID

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留, 保持复位值
UID	0	WO	0x0	UID: 更新中断禁止. 0: 无中断禁止. 1: 更新中断禁止.

1.4.5 中断有效状态寄存器(BSTIM_IVS)

		Address offset:		0x14											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															UIS

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留, 保持复位值
UIS	0	RO	0x0	UIS: 更新中断有效状态. 0: 更新中断禁止状态. 1: 更新中断使能状态. 在 IER/IDR 中任一寄存器中写“1”将使能或禁止该位

1.4.6 原始中断标志寄存器(BSTIM_RIF)

		Address offset:		0x18											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

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

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值
UIF	0	RO	0x0	UIF: 更新原始中断标志 当产生更新事件时该位由硬件置'1'。它由软件清'0'。 0: 无更新事件产生； 1: 更新中断等待响应。当寄存器被更新时该位由硬件置'1'： 一若 BSTIM_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 一若 BSTIM_CR1 寄存器的 URS=0、UDIS=0，当设置 BSTIM_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 一若 BSTIM_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

1. 4. 7 屏蔽中断标志寄存器(BSTIM_IFM)

		Address offset:		0x1C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															UIM

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值
UIM	0	RO	0x0	UIM: 更新屏蔽中断标志 当产生更新事件时该位由硬件置'1'。它由软件清'0'。 0: 无更新事件产生； 1: 更新中断等待响应。当寄存器被更新时该位由硬件置'1'： 一若 BSTIM_CR1 寄存器的 UDIS=0，当重复计数器数值上溢或下溢时(重复计数器=0 时产生更新事件)。 一若 BSTIM_CR1 寄存器的 URS=0、UDIS=0，当设置 BSTIM_EGR 寄存器的 UG=1 时产生更新事件，通过软件对计数器 CNT 重新初始化时。 一若 BSTIM_CR1 寄存器的 URS=0、UDIS=0，当计数器 CNT 被触发事件重新初始化时。

1. 4. 8 中断清除寄存器(BSTIM_ICR)

		Address offset:		0x20											
		Reset value:		0x0000 0000											

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

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值
UIC	0	W1C	0x0	UIC: 更新中断清除寄存器。 0: 无。 1: 更新中断清除。

1. 4. 9 事件产生寄存器(BSTIM_EGR)

		Address offset:		0x24											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															UG

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值
UG	0	WO	0x0	UG: 产生更新事件。 该位由软件置'1'，由硬件自动清'0'。 0: 无动作； 1: 重新初始化计数器，并产生一个(寄存器)更新事件。注意预分频器的计数器也被清'0' (但是预分频系数不变)。若在中心对称模式下或 DIR=0(向上计数)则计数器被清'0'；若 DIR=1(向下计数)则计数器取 BSTIM_ARR 的值。

1. 4. 10 计数寄存器(BSTIM_CNT)

		Address offset:		0x34											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值

CNT	15:0	R/W	0x0	CNT: 计数器值

1. 4. 11 预分频寄存器(BSTIM_PSC)

		Address offset:		0x38											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PSC[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留, 保持复位值
PSC	15:0	R/W	0x0	PSC: 预分频值 预分频器的值 (Prescaler value) 计数器的时钟频率 (CK_CNT) 等于 $f_{CK\_PSC} / (PSC[15:0] + 1)$ 。每次当更新事件产生时, PSC 的值被装入当前预分频器寄存器; 更新事件包括计数器被 TIM_EGR 的 UG 位清 '0' 或被工作在复位模式的从控制器清 '0'。

1. 4. 12 自动重装载寄存器(BSTIM_ARR)

		Address offset:		0x3C											
		Reset value:		0x0000 FFFF											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留, 保持复位值
ARR	15:0	R/W	0xFFFF	ARR: 自动重装载值 ARR 包含了将要装载入实际的自动重装载寄存器的值。当自动重装载的值为空时, 计数器不工作。

1. 4. 13 DMA 事件使能寄存器(BSTIM_DMAEN)

		Address offset:		0x58											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

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

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值
UDE	0	R/W	0x0	UDE: 更新中断 DMA 请求使能 0: 更新中断 DMA 请求禁止。 1: 更新中断 DMA 请求使能。

confidential

第15章 低功耗定时器（LPTIM）

1.1 LPTIM 简介

LPTIM 是为低功耗应用而设计的 16 位计数器。得益于多种时钟源选择，LPTIM 支持使用外部输入作为时钟源，可以在内部时钟停止的情况下计数。

1.2 LPTIM 主要特性

- ◆ 16 位向上计数
- ◆ 3 位预分频器支持 8 种分频系数 (1,2,4,8,16,32,64,128)
- ◆ 时钟源
 - 内部时钟源：HSE,HSI,LSI,LSE,HSE,PLL
 - 外部时钟源：外部端口输入
- ◆ 16 位 ARR 自动加载寄存器
- ◆ 16 位比较寄存器
- ◆ 连续或单发模式可选
- ◆ 软件或硬件触发可选
- ◆ 可编程滤波器
- ◆ 可配置输出：脉冲，PWM，翻转
- ◆ 输出极性可配

注意：数字滤波器的滤波灵敏度控制是按组分别控制的，但不每一个滤波器进行单独控制。

滤波器的灵敏度是对所选择通道信号进行连续采样，由采样值的次数来确定信号的变化是否为合法。

下图表示 2 次连续采样时的滤波器行为

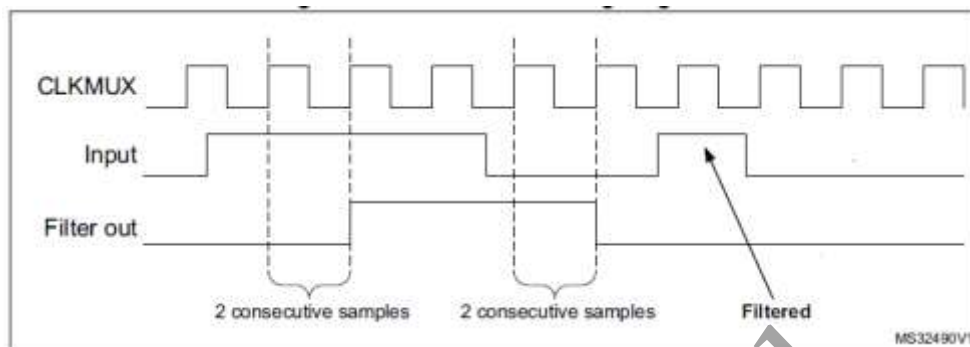


图 15-2 数字滤波器

1. 4. 2 预分频器

LPTIM 的 16 位计数器前置一个可配置 2 的 N 次方分频的预分频器。预分频器的分频系数由 PRESC[2:0]控制。下表列出了所有可设置系数

programming	dividing factor
000	/1
001	/2
010	/4
011	/8
100	/16
101	/32
110	/64
111	/128

图 15-3 预分频器分频系数

1. 4. 3 触发源选择

LPTIM 计数器可由软件启动，也可选择 8 个触发通道中的一个作为触发源，在检测到该通道信号的有效边缘之后，启动计数器。

TRIGEN[1: 0]用来配置选择 LPTIM 的触发源。

当 TRIGEN=00 时，软件置位 CNTSTRT 或者 SNGSTRT 可马上启动 LPTIM 计数器。

余下各种情况分别配置触发信号的有效边缘。LPTIM 计数器将在检测到有效触发边缘后开始计数。

当 TRIGEN 不为 00 时，可设置 TRIGSEL[2:0]来选择触发通道作为 LPTIM 的启动

信号。

外部触发信号对于 LPTIM 来说是异步的。所以在检测到触发后，至少需要 2 个计数时钟的迟延时间来保证同步。待同步完成，计数器被启动。

如果计数器已经启动，此时再次检测到新的触发事件，该事件将被忽略。

注意：在置位 SNGSTRT/CNTSTRT 之前，必须使能计数器。在计数器禁止期间，对 SNGSTRT/CNTSTRT 操作时无效的。

1.4.4 计数模式

LPTIM 有两种工作模式：

连续模式：计数器自由计数，计数器由触发事件启动后便一直计数，除非设定关闭计数器。

单发模式：计数器由触发事件启动计数后，计数到 ARR 值后停止。检测到新的触发事件会重启计数器。在计数器启动之后，直到计数值达到 ARR 这段期间产生的触发事件都将被忽略。

若要进行单发计数，应先置位 SNGSTRT。

如果选择了外部触发，每一个在 SNGSTRT 置位之后和计数器停止之后到达的触发事件，将启动新的单发计数。如下图所示：

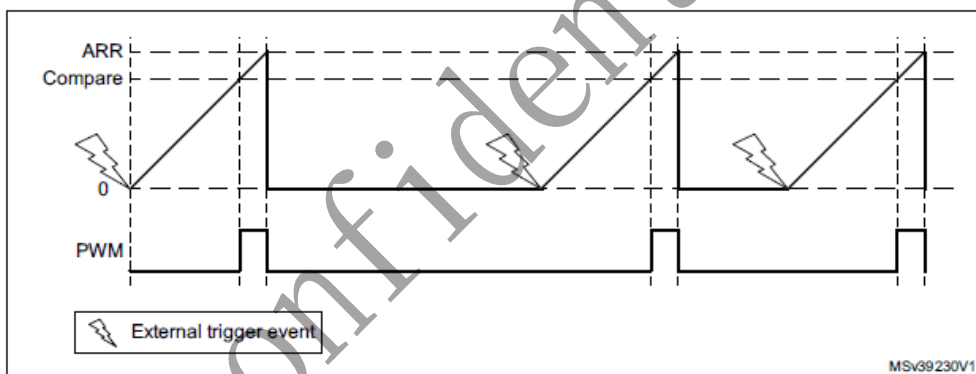


图 15-4 单发计数

若要进行连续计数，需软件置位 CNTSTRT。

选择外部触发的情况下，置位 CNTSTRT 后，触发事件将启动计数器开始连续计数。之后的触发事件将被忽略。如下图所示：

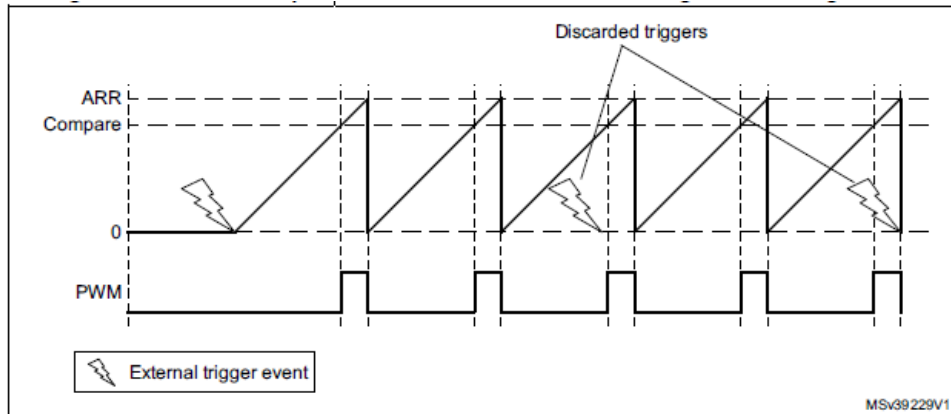


图 15-5 连续计数

如果采用软件启动的方式，置位 CTNSTRT 将启动计数器进行连续计数。

SNTSTRT 和 CNTSTRT 位只能在计数器被使能后（ENABLE 位为 1）才能置位。单发模式和连续模式之间可以相互切换。

在连续模式计数期间，置位 SNGSTRT 可将 LPTIM 切换至单发模式。计数器在计数至 ARR 后将停止。

在单发模式计时期间，置位 CNTSTRT 将 LPTIM 切换至连续计数模式。计数器计数至 ARR 后将重启计数。

1.4.5 输出波形

2 个 16 位寄存器，LPTIM_ARR（自动加载寄存器）和 LPTIM_CMP（比较寄存器），用于生成各种不同的波形，并输出到端口。

PWM: LPTIM 输出在 LPTIM_CNT 和 LPTIM_CMP 匹配时置位，在 LPTIM_CNT 和 LPTIM_ARR 匹配时清零。

脉冲: LPTIM 输出在 LPTIM_CNT 和 LPTIM_ARR 匹配时将置位一个计数时钟周期，在下次匹配之前一直处于清零状态。

翻转: LPTIM 输出在 LPTIM_CNT 和 LPTIM_ARR 匹配时将当前输出翻转。

1.4.6 寄存器更新

LPTIM_ARR 和 LPTIM_CMP 寄存器可在 APB 总线写操作之后马上更新或者在计数器计完当前周期后更新。PRELOAD 可控制 LPTIM_ARR 和 LPTIM_CMP 的更新方式。

当 PRELOAD 为 0 时，LPTIM_ARR 和 LPTIM_CMP 在写操作之后马上更新。

当 PRELOAD 为 1 时，若计数器已经启动，LPTIM_ARR 和 LPTIM_CMP 将在当前计数周期结束后更新。

APB 总线和 LPTIM 使用不同的时钟，所以在 APB 总线的写操作和设定值真正写到计数器的比较寄存器之间是有一些延迟的。在这期间，须避免对同一个寄存器的写操作。

LPTIM_SYNCSTAT 的 ARRWBSY 和 CMPWBSY 标志位分别表示对 LPTIM_ARR 和 LPTIM_CMP 的写操作是否完成。

对 LPTIM_ARR 和 LPTIM_CMP 写之后，新的写命令必须在前一次写操作结束之后才能进行。连续的写操作可导致不可预期的结果。

1.4.7 LPTIM 中断

如果 LPTIM_IER 使能，以下事件可产生中断或唤醒事件

- 计数比较匹配
- 自动加载匹配
- 外部触发事件

1.5 LPTIM 寄存器描述

1.5.1 LPTIM 控制寄存器 0 (LPTIM_CR0)

		Address offset:		0x00											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									PRELOAD	WAVEPOL	WAVE		TRIGEN		Reserved
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIGSEL			Reserved	PRESC			Reserved	TRGFLT		Reserved	CKFLT		Reserved	CKPOL	CKSEL

Name	Bit	Type	Reset value	Description
Reserved	31:23	N/A	0x00	保留，保持复位值
PRELOAD	22	R/W	0x0	寄存器更新模式 控制LPTIM_ARR和LPTIM_CMP寄存器的更新 0: 寄存器在APB总线写操作后更新 1: 寄存器在 LPTIM 计数周期结束后更新
WAVEPOL	21	R/W	0x0	输出波形极性 0: 输出不取反 1: 输出取反
WAVE	20:19	R/W	0x0	输出波形格式选择 00: 输出禁止 01: LPTIM计数溢出时输出翻转 10: LPTIM计数溢出时输出脉冲 11: LPTIM输出PWM波
TRIGEN	18:17	R/W	0x0	触发使能及极性 TRIGEN控制LPTIM是否由外部触发启动。如果选择外部触发，有3中极性可供选择 00: 软件触发 01: 上升沿有效 10: 下降沿有效 11: 双边沿有效
Reserved	16	N/A	0x0	保留，保持复位值
TRIGSEL	15:13	R/W	0x0	触发选择 TRIGSEL在如下8个通道中选择触发源 000: ext_trig0 001: ext_trig1 010: ext_trig2 011: ext_trig3 100: ext_trig4 101: ext_trig5

Reserved				110: ext_trig6 111: ext_trig7
	12	N/A	0x0	保留，保持复位值
PRESC				时钟预分频 PRESC配置预分频系数。 000: /1 001: /2 010: /4 011: /8 100: /16 101: /32 110: /64 111: /128
	11:9	R/W	0x0	
Reserved				保留，保持复位值
	8	N/A	0x0	
TRGFLT				配置触发信号的数字滤波器 TRGFLT的值设定了滤波宽度（连续采样1或0的次数超过TRGFLT设定的个数才能被认为是合法的信号电平变化） 00: 外部触发信号任何变化都是合法变化 01: 外部触发信号电平须维持2个时钟周期以上 10: 外部触发信号电平须维持4个时钟周期以上 11: 外部触发信号电平须维持8个时钟周期以上
	7:6	R/W	0x0	
Reserved				保留，保持复位值
	5	N/A	0x0	
CKFLT				配置外部时钟的数字滤波器 CKFLT的值设定了滤波宽度（连续采样1或0的次数超过CKFLT设定的个数才能被认为是合法的信号电平变化） 00: 外部时钟任何变化都是合法变化 01: 外部时钟信号电平须维持2个时钟周期以上 10: 外部时钟信号电平须维持4个时钟周期以上 11: 外部时钟信号电平须维持8个时钟周期以上
	4:3	R/W	0x0	
Reserved				保留，保持复位值
	2	N/A	0x0	
CKPOL				时钟极性 如果LPTIM采用外部时钟源，CKPOL用于配置计数的有效边沿 0: 上升沿为计数有效边沿 1: 下降沿为计数有效边沿
	1	R/W	0x0	
CKSEL				时钟选择 CKSEL用于选择哪个时钟源作为LPTIM的计数时钟 0: LPTIM采用内部时钟源（APB时钟及其他片上时钟） 1: LPTIM采用外部时钟源（从input1输入）
	0	R/W	0x0	

1.5.2 LPTIM 控制寄存器 1 (LPTIM_CR1)

		Address offset:		0x04											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													CNTS TRT	SNGS TRT	ENAB LE

Name	Bit	Type	Reset value	Description
Reserved	31:3	N/A	0x00	保留，保持复位值
CNTSTRT	2	R/W	0x0	LPTIM启动连续模式 该位由软件置位，由硬件清零 在软件启动方式（TRIGEN[1:0]='00'）下，置位CNTSTRT将启动LPTIM进入连续模式。 在软件启动方式禁止时（TRIGEN[1:0]不为00），置位CNTSTRT后，LPTIM将在检测到外部触发后启动连续计数。 如果LPTIM在单发计数模式下，置位CNTSTRT将使LPTIM计数到LPTIM_ARR后也会继续计数。 该位只能在LPTIM使能后进行设定。读取该位始终为0。
SNGSTRT	1	R/W	0x0	LPTIM启动单发模式 该位由软件置位，由硬件清零 在软件启动方式（TRIGEN[1:0]='00'）下，置位SNGSTRT将启动LPTIM进入单发模式。 在软件启动方式禁止时（TRIGEN[1:0]不为00），置位SNGSTRT后，LPTIM将在检测到外部触发后启动单发计数。 如果LPTIM在连续计数模式下，置位SNGSTRT将使LPTIM计数到LPTIM_ARR后停止计数。 该位只能在LPTIM使能后进行设定。读取该位始终为0。
ENABLE	0	R/W	0x0	LPTIM使能 ENABLE位由软件置位或清零 0:LPTIM禁止 1:LPTIM使能

1.5.3 LPTIM 自动加载寄存器（LPTIM_ARR）

		Address offset:		0x08											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARR[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值

ARR	15:0	R/W	0x0	自动加载值 ARR是LPTIM的自动加载值。 ARR必须严格大于CMP[15: 0]的值。 ARR 的值必须在 LPTIM 使能后改写（ENABLE 设为‘1’）

1.5.4 LPTIM 计数寄存器（LPTIM_CNT）

		Address offset:		0x0C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
CNT	15:0	R	0x0	计数值 当LPTIM正在以异步时钟计数，读取LPTIM_CNT可能会返回不可信赖值。所以，在这种情况下有必要进行2次连续的读操作，并且如果两次读取的值相同，说明当前读到的计数值是可靠的。

1.5.5 LPTIM 比较值寄存器（LPTIM_CMP）

		Address offset:		0x10											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMP[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
CMP	15:0	R/W	0x0	比较值 CMP作为LPTIM的比较值 LPTIM_CMP 的值只能在 LPTIM 使能后被改写（ENABLE 设为‘1’）

1. 5. 6 LPTIM 中断使能寄存器 (LPTIM_IER)

		Address offset:		0x18											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													EXTT RIGIE	ARR MIE	CMP MIE

Name	Bit	Type	Reset value	Description
Reserved	31:3	N/A	0x00	保留, 保持复位值
EXTTRIGIE	2	R/W	0x0	外部触发有效边沿中断 0: EXTTRIG中断禁止 1: EXTTRIG 中断使能
ARRMIE	1	R/W	0x0	自动加载中断使能 0: ARRM中断禁止 1: ARRM 中断使能
CMPMIE	0	R/W	0x0	比较匹配中断使能 0: CMPM中断禁止 1: CMPM 中断使能

1. 5. 7 LPTIM 中断状态寄存器 (LPTIM_ISR)

		Address offset:		0x1C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													EXTT RIG	ARR M	CMP M

Name	Bit	Type	Reset value	Description
Reserved	31:3	N/A	0x00	保留, 保持复位值
EXTTRIG	2	R	0x0	外部触发有效边沿事件 EXTTRIG 由硬件置位, 表示在所选择的外部触发通路上检测到有效边沿。该位通过软件操作 LPTIM_IFC 清零。
ARRM	1	R	0x0	自动加载值匹配 ARRM 由硬件置位, 表示 LPTIM_CNT 计数值达到 LPTIM_ARR 寄存器的设定值。该位通过软件操作 LPTIM_IFC 清零。
CMPM	0	R	0x0	比较匹配 CMPM 由硬件置位, 表示 LPTIM_CNT 计数值达到 LPTIM_CMP 寄存器的设定值。该位通过软件操作 LPTIM_IFC 清零。

1.5.8 LPTIM 中断状态清零寄存器 (LPTIM_ICR)

		Address offset:		0x20											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													EXTT RIG	ARR M	CMP M

Name	Bit	Type	Reset value	Description
Reserved	31:3	N/A	0x00	保留，保持复位值
EXTTRIG	2	W	0x0	外部触发有效边沿事件标志清零 对该位写 1 有效。
ARRM	1	W	0x0	自动加载值匹配标志清零 对该位写 1 有效。
CMPM	0	W	0x0	比较匹配标志清零 对该位写 1 有效。

1.5.9 LPTIM 更新控制寄存器 (LPTIM_UPDATE)

		Address offset:		0x30											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															UDIS

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值
UDIS	0	R/W	0x0	寄存器更新 0: 寄存器写入后新值将被更新到低速时钟域 1: 寄存器写入后新值不被更新 置 1 后，寄存器新写值将不被更新直到该位被清零。用该位可控制多个寄存器同时更新。

1.5.10 LPTIM 写同步状态寄存器 (LPTIM_SYNCSTAT)

		Address offset:		0x34											
		Reset value:		0x0000 0000											

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												CMP WBS Y	ARR WBS Y	CON 1WB SY	Reser ved

Name	Bit	Type	Reset value	Description
Reserved	31:4	N/A	0x00	保留，保持复位值
CMPWBSY	3	R/W	0x0	寄存器CMP写同步状态 0: 同步未开始或同步已结束 1: 同步中
ARRWBSY	2	R/W	0x0	寄存器ARR写同步状态 0: 同步未开始或同步已结束 1: 同步中
CON1WBSY	1	R/W	0x0	寄存器CON1写同步状态 0: 同步未开始或同步已结束 1: 同步中
Reserved	0	N/A	0x0	保留，保持复位值

第16章 模拟/数字转换（ADC）

1.1 ADC 简介

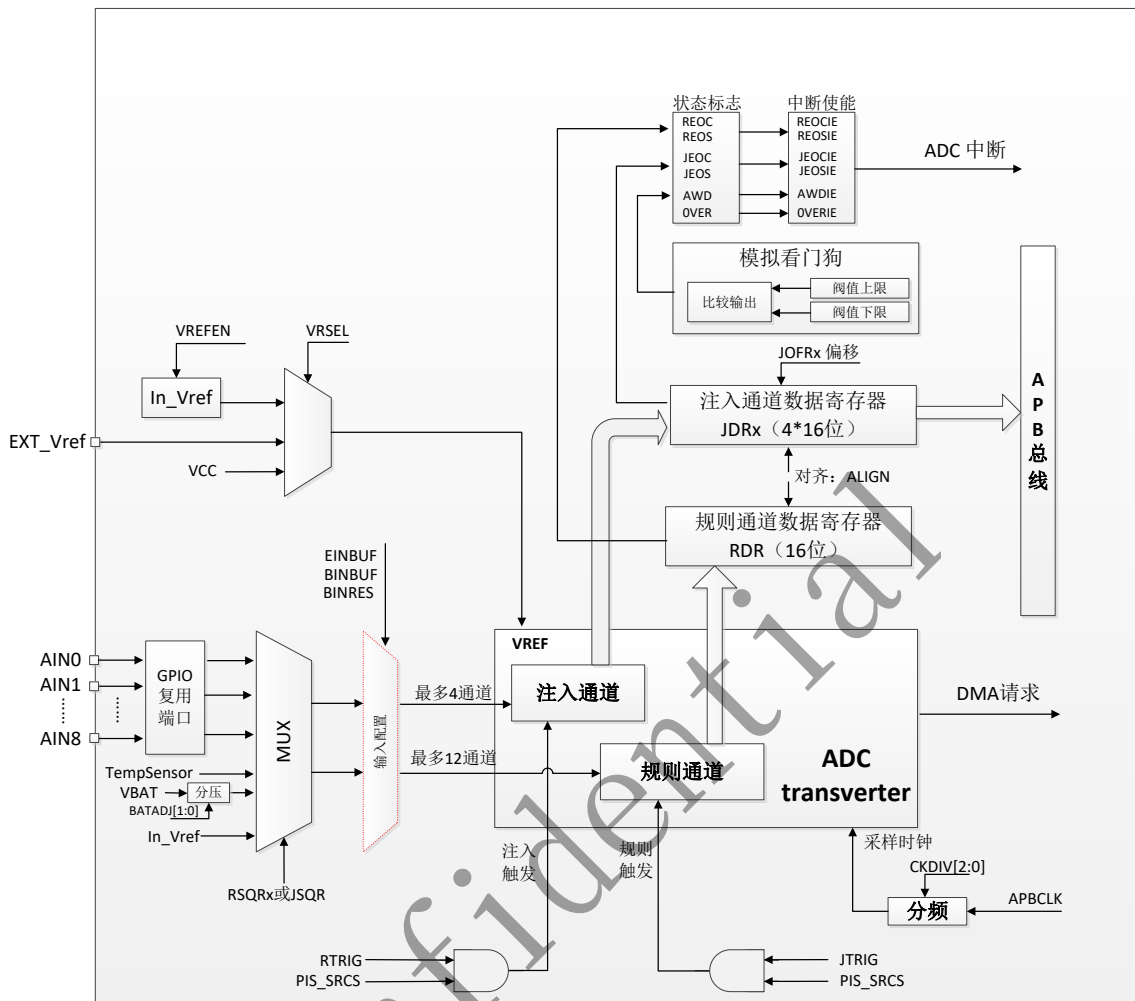
ADC 模块是 12 位精度的逐次逼近型模数转换器。它具有多达 12 个复用通道，可测量来自 9 个外部信号、两个内部信号和 VBAT 电压信号。这些通道的 A/D 转换可在单次、连续、扫描或不连续采样模式下进行。ADC 的结果存储在一个左对齐或右对齐的 16 位数据寄存器中。

ADC 模块具有模拟看门狗特性，允许应用检测输入电压是否超过了用户自定义的阈值上限或下限。

1.2 ADC 主要特性

- ◆ 可配置的转换精度（6/8/10/12 位）
- ◆ 在规则转换、注入转换结束后以及发生模拟看门狗或溢出事件时产生中断
- ◆ 支持单次或连续转换模式
- ◆ 用于自动将通道 0 转换为通道“n”的扫描模式
- ◆ 可配置的数据对齐方式
- ◆ 可独立设置各通道采样时间
- ◆ 可配置外部触发器选项，可为规则转换和注入转换配置极性
- ◆ 支持不连续采样模式
- ◆ 可配置的参考源选择
- ◆ 可配置的转换时钟分频
- ◆ 规则通道转换期间可产生 DMA 请求

1.3 ADC 结构框图



ADC 结构框图

1.4 ADC 功能描述

1.4.1 ADC 控制

可通过将 ADC_CR2 寄存器中的 ADEN 位置 1 来为使能 ADC。

RTRIG 或 JTRIG 位置 1 时，启动 AD 转换。

1.4.2 ADC 时钟

ADC 具有两个时钟方案：

- 用于模拟电路的时钟：ADCCLK

此时钟来自于经可编程预分频器分频的 APB 时钟，该预分频器可将 APB 时钟产生 1~128 分频时钟供 ADC 模拟电路使用。

- 用于数字接口的时钟（用于寄存器读/写访问）

此时钟为 APB 时钟。

1.4.3 通道选择

有 12 条复用通道。可以将转换分为两组：规则转换和注入转换。每个组包含一个转换序列，该序列可按任意顺序在任意通道上完成。例如，可按以下顺序对序列进行转换：ADC_IN3、ADC_IN8、ADC_IN2、ADC_IN2、ADC_IN0、ADC_IN2、ADC_IN2、ADC_IN7。

- 一个规则转换组最多由 12 个转换构成。必须在 ADC_RSQRx 寄存器中选择转换序列的规则通道及其顺序。规则转换组中的转换总数必须写入 ADC_SQR 寄存器中的 RSQL 位。
- 一个注入转换组最多由 4 个转换构成。必须在 ADC_JSQR 寄存器中选择转换序列的注入通道及其顺序。注入转换组中的转换总数必须写入 ADC_SQR 寄存器中的 JSQR 位。

温度传感器、VBAT 和 VREF_1P4 内部通道：

- 温度传感器内部连接到通道 ADC_IN9。
- VBAT 通过分压网路(BATADJ)连接通道 ADC_IN10。
- 内部参考电压 VREF_1P4 连接内部通道 ADC_IN11。

1.4.4 单次转换模式

在单次转换模式下，ADC 执行一次转换。CONT 位为 0 时，可通过以下方式启动此模式：

- 将 ADC_CR2 寄存器中的 RTRIG 位置 1（仅适用于规则通道）
- 将 ADC_CR2 寄存器中的 JTRIG 位置 1（仅适用于注入通道）
- 外部触发（适用于规则通道或注入通道）

完成所选通道的转换之后：

- 如果转换了规则通道：
 - 转换数据存储在 16 位 ADC_RDR 寄存器中
 - REOC（转换结束）标志置 1
 - REOCIE 位置 1 时将产生中断
- 如果转换了注入通道：
 - 转换数据存储在 16 位 ADC_JDR1 寄存器中
 - JEOC（注入转换结束）标志置 1
 - JEOCIE 位置 1 时将产生中断

然后，ADC 停止。

1.4.5 连续转换模式

在连续转换模式下，ADC 结束一个转换后立即启动一个新的转换。CONT 位为 1 时，可通过外部触发或将 ADC_CR2 寄存器中的 RTRIG 位置 1 来启动此模式（仅适用于规则通道）。

每次转换之后：

- 如果转换了规则通道组：
 - 上次转换的数据存储在 16 位 ADC_RDR 寄存器中

- REOC（转换结束）标志置 1
- REOCIE 位置 1 时将产生中断
- 规则组完成全部设置通道数(RSQL)转换后，REOS（转换序列结束）标志置 1
- 规则组完成全波设置通道数(RSQL)转换后，REOSIE 位置 1 时将产生中断

无法连续转换注入通道。连续模式下唯一的例外情况是，在使能 JAUTO 位时，注入通道配置为在规则通道之后自动转换。

1.4.6 时序图

如图所示，ADC 在开始精确转换之前需要一段稳定时间 t_{STAB} 。ADC 开始转换并经过 15 个时钟周期后，REOC 标志置 1，转换结果存放在 16 位 ADC 数据寄存器中。

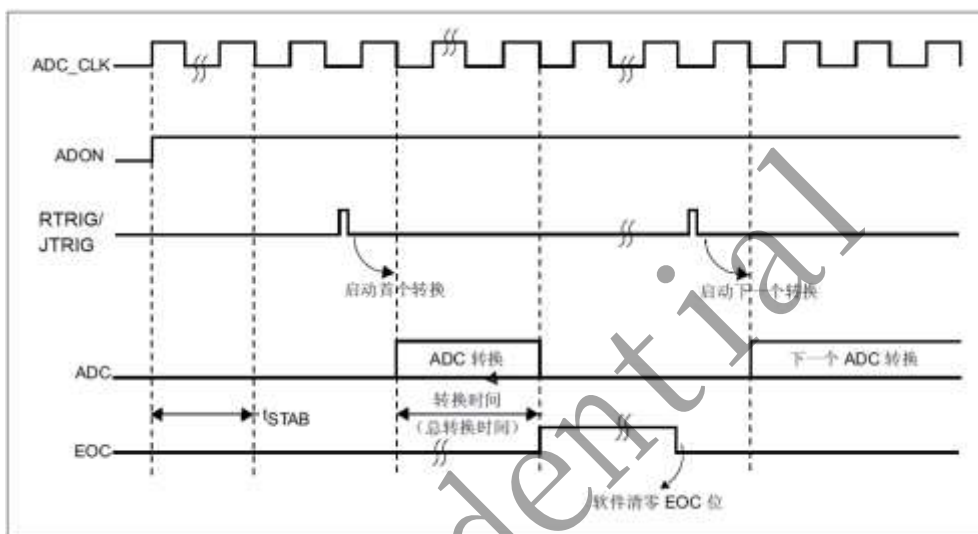


图 16-1 ADC 转换时序图

1.4.7 模拟看门狗

如果 ADC 转换的模拟电压低于阈值下限或高于阈值上限，则 AWD 模拟看门狗状态位会置 1。这些阈值在 ADC_HTR 和 ADC_LTR 寄存器的 12 个最低有效位中进行编程。可以使用 ADC_CR1 寄存器中的 AWDIE 位使能中断。

阈值与 ADC_CR2 寄存器中的 ALIGN 位的所选对齐方式无关。在对齐之前，会将模拟电压与阈值上限和下限进行比较。

下表介绍了应如何配置 ADC_CR1 寄存器才能在一个或多个通道上使能模拟看门狗。

模拟看门狗保护通道	AWDSGL	RAWDEN	JAWDEN
无	X	0	0
所有注入通道	0	0	1
所有规则通道	0	1	0
所有规则通道和注入通道	0	1	1
单个注入通道 (AWDCH 选择)	1	0	1
单个规则通道 (AWDCH 选择)	1	1	0
单个规则通道或注入通道 (AWDCH 选择)	1	1	1

表 16-1 模拟看门狗通道选择

1. 4. 8 扫描模式

此模式用于扫描一组模拟通道。

通过将 ADC_CR1 寄存器中的 SCAN 位置 1 来选择扫描模式。将此位置 1 后，ADC 会扫描在 ADC_RSQRx 寄存器（对于规则通道）或 ADC_JSQR 寄存器（对于注入通道）中选择的所有通道。为组中的每个通道都执行一次转换。每次转换结束后，会自动转换该组中的下一个通道。如果将 CONT 位置 1，规则通道转换不会在组中最后一个所选通道处停止，而是再次从第一个所选通道继续转换。

如果将 DMAEN 位置 1，则在每次规则通道转换之后，均使用直接存储器访问 (DMA) 控制器将转换自规则通道组的数据（存储在 ADC_DR 寄存器中）传输到 SRAM。

在以下情况下，ADC_SR 寄存器中的 REOC 位置 1:

- 如果 REOCS 位清零，在每个规则组序列转换结束时
- 如果 REOCS 位置 1，在每个规则通道转换结束时

从注入通道转换的数据始终存储在 ADC_JDRx 寄存器中。

1. 4. 9 注入通道管理

触发注入

要使用触发注入，必须将 ADC_CR1 寄存器中的 JAUTO 位清零。

1. 通过外部触发或将 ADC_CR2 寄存器中的 RTRIG 位置 1 来启动规则通道组转换。
2. 如果在规则通道组转换期间出现外部注入触发或者 JTRIG 位置 1，则当前的转换会复位，并且注入通道序列会切换为单次扫描模式。
3. 然后，规则通道组的规则转换会从上上次中断的规则转换处恢复。

如果在注入转换期间出现规则事件，注入转换不会中断，但在注入序列结束时执行规则序列。

使用触发注入时，必须确保触发事件之间的间隔长于注入序列。

自动注入

如果将 JAUTO 位置 1，则注入组中的通道会在规则组通道之后自动转换。这可用于转换最多由 16 个转换构成的序列，这些转换在 ADC_RSQRx 和 ADC_JSQR 寄存器中编程。

在此模式下，必须禁止注入通道上的外部触发。

如果 CONT 位和 JAUTO 位均已置 1，则在转换规则通道之后会继续转换注入通道。

不能同时使用自动注入和不连续采样模式。

1. 4. 10 不连续采样模式

规则组

可将 ADC_CR1 寄存器中的 DISCEN 位置 1 来使能此模式。该模式可用于转换含有 n ($n \leq 8$) 个转换的短序列，该短序列是在 ADC_RSQRx 寄存器中选择的转换序列的一部分。可通过写入 ADC_CR1 寄存器中的 DISCNUM 位来指定 n 的值。

出现外部触发时，将启动在 ADC_RSQRx 寄存器中选择的接下来 n 个转换，直到序列中的所有转换均完成为止。通过 ADC_SQR 寄存器中的 RSQL 位定义总序列长度。

示例：

$n = 3$ ，要转换的通道 = 0、1、2、3、6、7、9、10

第 1 次触发：转换序列 0、1、2

第 2 次触发：转换序列 3、6、7

第 3 次触发：转换序列 9、10 并生成 REOC 事件

第 4 次触发：转换序列 0、1、2

在不连续采样模式下转换规则组时，不会出现翻转。

转换完所有子组后，下一个触发信号将启动第一个子组的转换。在上述示例中，第 4 次触发重新转换了第 1 个子组中的通道 0、1 和 2。

注入组

可将 ADC_CR1 寄存器中的 JDISCEN 位置 1 来使能此模式。在出现外部触发事件之后，可使用该模式逐通道转换在 ADC_JSQR 寄存器中选择的序列。

出现外部触发时，将启动在 ADC_JSQR 寄存器中选择的下一个通道转换，直到序列中的所有转换均完成为止。通过 ADC_SQR 寄存器中的 JSQL 位定义总序列长度。

示例：

$n = 1$ ，要转换的通道 = 1、2、3

第 1 次触发：转换通道 1

第 2 次触发：转换通道 2

第 3 次触发：转换通道 3 并生成 EOC 和 JEEOC 事件

第 4 次触发：通道 1

转换完所有注入通道后，下一个触发信号将启动第一个注入通道的转换。在上述示例中，第 4 次触发重新转换了第 1 个注入通道。

不能同时使用自动注入和不连续采样模式。

不得同时为规则组和注入组设置不连续采样模式。只能针对一个组使能不连续采样模式。

1.5 数据对齐

ADC_CR2 寄存器中的 ALIGN 位用于选择转换后存储的数据的对齐方式。可选择左对齐和右对齐两种方式，如下图所示。

注入通道组的转换数据将加上 ADC_JOFRx 寄存器中写入的用户自定义偏移量，因

此结果可以是一个负值。

对于规则组中的通道，不会减去任何偏移量，因此只有十二个位有效。

1.6 可独立设置各通道采样时间

ADC 会在数个 ADCCLK 周期内对输入电压进行采样，可使用 ADC_SMPR1 寄存器中的 SMPx 位修改周期数。每个通道均可以使用不同的采样时间进行采样。

总转换时间的计算公式如下：

$T_{conv} = \text{ADCCLK 时钟周期} \times \text{时钟周期数}$ （周期数=采样周期+转换周期）

其中，转换周期固定 15 个时钟周期，采样周期通过 SMPR 寄存器配置。

示例：

ADCCLK = 200KHz 且采样周期 = 15+15 个周期时：

$T_{conv} = 5\mu s \times 30 = 150\mu s$ （APB 为 16MHz 时）

1.7 数据管理

1.7.1 使用 DMA

由于规则通道组只有一个数据寄存器，因此，对于多个规则通道的转换，使用 DMA 非常有帮助。这样可以避免丢失在下次写入之前还未被读出的 ADC_DR 寄存器中的数据。

在使能 DMA 模式的情况下（ADC_CR2 寄存器中的 DMA 位置 1），每完成规则通道组中的一个通道转换后，都会生成一个 DMA 请求。这样便可将转换的数据从 ADC_RDR 寄存器传输到用软件选择的目标位置。

尽管如此，如果数据丢失（溢出），则会将 ADC_SR 寄存器中的 OVR 位置 1 并生成一个中断（如果 OVR1E 使能位已置 1）。随后会禁止 DMA 传输并且不再接受 DMA 请求。在这种情况下，如果生成 DMA 请求，则会中止正在进行的规则转换并忽略之后的规则触发。随后需要将所使用的 DMA 流中的 OVR 标志和 DMAEN 位清零，并重新初始化 DMA 和 ADC，以将需要的转换通道数据传输到正确的存储器单元。只有这样，才能恢复转换并再次使能数据传输。注入通道转换不会受到溢出错误的影响。

在 DMA 模式下，当 OVR = 1 时，传送完最后一个有效数据后会阻止 DMA 请求，这意味着传输到 RAM 的所有数据均被视为有效。

在最后一次 DMA 传输（DMA 控制器的 ctrl_data 寄存器中配置的传输次数）结束时：

- 如果将 ADC_CR2 寄存器中的 DMAEN 位清零，则不会向 DMA 控制器发出新的 DMA 请求（这可避免产生溢出错误）。不过，硬件不会将 DMA 位清零。必须将该位写入 0 然后写入 1 才能启动新的传输。
- 如果将 DMAEN 位置 1，则可继续生成请求。从而允许在双缓冲区循环模式下配置 DMA。要在使用 DMA 时将 ADC 从 OVR 状态中恢复，请按以下步骤操作：

1. 重新初始化 DMA
2. 将 ADC_SR 寄存器中的 ADC OVR 位清零
3. 触发 ADC 以开始转换。

1.7.2 在不使用 DMA 的情况下管理转换序列

如果转换过程足够慢，则可使用软件来处理转换序列。在这种情况下，每当一个通道转换结束时，REOC 都会置 1，并且可以读取 ADC_RDR 寄存器。溢出管理与使用 DMA 时的管理相同。

要在 REOC 位置 1 时将 ADC 从 OVR 状态中恢复，请按以下步骤操作：

1. 将 ADC_SFCR 寄存器中的 OVRC 位置 1
2. 触发 ADC 以开始转换。

1.7.3 在不使用 DMA 和溢出检测情况下进行转换

ADC 在转换一个或多个通道时不是每次都读取数据的情况下，这可能会很有用（例如，存在模拟看门狗时）。为此，必须禁止 DMA (DMA = 0) 并且忽略 REOC 标志，仅检测序列结束时 REOS 标志状态。在此配置中，溢出检测已禁止。

1.8 ADC 中断

当模拟看门狗状态位和溢出状态位分别置 1 时，规则组和注入组在转换结束时可能会产生中断。可以使用单独的中断使能位以实现灵活性。

ADC_SR 寄存器中存在另外两个标志，但这两个标志不存在中断相关性：

- JTRIG（开始转换注入组的通道）
- RTRIG（开始转换规则组的通道）

中断事件	事件标志位	使能控制位
规则组通道转换结束	REOC	REOCIE
规则组所有通道转换序列接收	REOS	REOSIE
注入组通道转换结束	JEOC	JEOCIE
注入组所有通道转换序列接收	JEOS	JEOSIE
发生模拟看门狗事件	AWD	AWDIE
溢出	OVR	OVRIE

表 16-2 ADC 中断

1.9 ADC 寄存器描述

1.9.1 ADC 状态寄存器（ADC_SR）

		Address offset:		00h											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						JSTRT	RSTR T	Reserved		JEOS	REOS	OVR	JEOC	REOC	AWD

Name	Bit	Type	Reset value	Description
Reserved	31:10	N/A	0x00	保留。
JSTRTC	9	R	0x0	注入转换开始标志位 0: 无操作 1: 产生注入转换开始标志
RSTRTC	8	R	0x0	规则转换开始标志位 0: 无操作 1: 产生规则转换开始标志
Reserved	7:6	N/A	0x00	保留。
JEOS	5	R	0x0	注入转换序列结束标志位 0: 无操作 1: 注入转换序列转换结束标志
REOS	4	R	0x0	规则转换序列结束标志位 0: 无操作 1: 产生规则转换序列转换结束标志
OVR	3	R	0x0	转换溢出标志位 0: 无操作 1: 产生转换溢出标志
JEOC	2	R	0x0	注入转换结束标志位 0: 无操作 1: 产生注入转换结束标志
REOC	1	R	0x0	规则转换结束标志位 0: 无操作 1: 产生规则转换结束标志
AWD	0	R	0x0	模拟看门狗标志位 0: 无操作 1: 产生模拟看门狗标志

1. 9. 2 ADC 状态清零寄存器 (ADC_SF CR)

		Address offset:		04 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						JSTRT	RSTR T	Reserved		JEOS C	REOS C	OVRC	JEOC C	REOC C	AWD

Name	Bit	Type	Reset value	Description
Reserved	31:10	N/A	0x00	保留
JSTRTC	9	W1	0x0	注入转换开始标志位清零 0: 无操作 1: 注入转换开始标志位清零
RSTRTC	8	W1	0x0	规则转换开始标志位清零 0: 无操作 1: 规则转换开始标志位清零

Reserved	7:6	N/A	0x00	保留
JEOSC	5	W	0x0	注入转换序列转换结束标志位清零 0: 无操作 1: 注入转换序列转换结束标志位清零
REOSC	4	W	0x0	规则转换序列转换结束标志位清零 0: 无操作 1: 规则转换序列转换结束标志位清零
OVRC	3	W1	0x0	转换溢出标志位清零 0: 无操作 1: 转换溢出标志位清零
JEOCC	2	W1	0x0	注入转换结束标志位清零 0: 无操作 1: 注入转换结束标志位清零
REOCC	1	W1	0x0	规则转换结束标志位清零 0: 无操作 1: 规则转换结束标志位清零
AWDC	0	W1	0x0	模拟看门狗标志位清零 0: 无操作 1: 模拟看门狗标志位清零

1.9.3 ADC 控制寄存器 1 (ADC_CR1)

		Address offset:		08 _H											
		Reset value:		00000011_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res					OVRI E	RES		RAW DEN	JAW DEN	Res		DISCNUM			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res	JEOSI E	REOS IE	JDISC EN	RDIS CEN	JAUT O	AWD SGL	SCAN	JEOCI E	AWDI E	REOC IE	AWDCH				

Name	Bit	Type	Reset value	Description
Reserved	31:27	N/A	0x00	保留.
OVRIE	26	R/W	0x0	溢出中断使能位 0: 禁止 1: 使能
RES	25:24	R/W	0x3	ADC 转换精度选择位 00: 6 位 01: 8 位 10: 10 位 11: 12 位
RAWDEN	23	R/W	0x0	规则通道模拟看门狗使能位 0: 禁止 1: 使能
JAWDEN	22	R/W	0x0	注入通道模拟看门狗使能位 0: 禁止 1: 使能
Reserved	21:20	N/A	0x00	保留.
DISCNUM	19:16	R/W	0x0	外部触发不连续转换通道数选择位 000: 1 个通道 001: 2 个通道

Reserved				
				111: 8 个通道
	15	N/A	0x00	保留.
				注入转换序列完成中断使能位
	14	R/W	0x0	0: 禁止 1: 使能
				规则转换序列完成中断使能位
	13	R/W	0x0	0: 禁止 1: 使能
				注入通道不连续转换使能位
	12	R/W	0x0	0: 禁止 1: 使能
				规则通道不连续转换使能位
	11	R/W	0x0	0: 禁止 1: 使能
				注入组自动转换使能位
	10	R/W	0x0	0: 禁止 1: 使能
				扫描模式单一通道模拟看门狗使能位
	9	R/W	0x0	0: 禁止 1: 使能
				扫描模式使能位
	8	R/W	0x0	0: 禁止 1: 使能
				注入转换完成中断使能位
	7	R/W	0x0	0: 禁止 1: 使能
				模拟看门狗中断使能位
	6	R/W	0x0	0: 禁止 1: 使能
				规则转换完成中断使能位
	5	R/W	0x0	0: 禁止 1: 使能
				模拟开门狗通道选择位
				00000: ADC 输入通道 0
				00001: ADC 输入通道 1
				
				01011: ADC 输入通道 11
				其他: 保留

1. 9. 4 ADC 控制寄存器 2 (ADC_CR2)

		Address offset:				0C _H									
		Reset value:				00000000_00000000_00000000_00000000 _B									
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RDY_ PH	RTRI G	BATADJ		Res	BINB UF	BINR ES	EINB UF	Res	JTRIG	Res					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				ALIG N	Reserved		DMA EN	Reserved				TEST	DIFF	CONT	ADE N

Name	Bit	Type	Reset value	Description

RDY_PH	31	R/W	0x00	ADC Ready 信号检测相位选择 0: 检测 Ready 信号上升沿 1: 检测 Ready 信号下降沿
RTRIG	30	W	0x0	规则通道转换软件触发位 0: 无操作 1: 触发开始规则通道转换
BATADJ	29:28	R/W	0x0	电池电压检测调整 00: 1/8 VBAT 01: 2/8 VBAT 10: 3/8 VBAT 11: 4/8 VBAT
Reserved	27	N/A	0x00	保留
BINBUF	26	R/W	0x0	Bypass 输入驱动运放 0: 无效 1: 有效
BINRES	25	R/W	0x0	Bypass 输入电阻 1/3 分压网络控制 0: 无效 1: 有效
EINBUF	24	R/W	0x0	使能输入驱动运放 0: 禁止 1: 使能
Reserved	23	N/A	0x0	保留
JTRIG	22	W1	0x0	注入通道转换软件触发位 0: 无操作 1: 触发开始注入通道转换
Reserved	21:12	N/A	0x00	保留
ALIGN	11	R/W	0x0	数据对齐方式位 0: 右对齐 1: 左对齐
Reserved	10:9	N/A	0x00	保留
DMAEN	8	R/W	0x0	DMA 访问使能位 0: 禁止 1: 使能
Reserved	7:4	N/A	0x00	保留
TEST	3	R/W	0x0	ADC 测试控制位 0: 禁止 1: 使能
DIFF	2	R/W	0x0	差分模式使能位 0: 禁止 1: 使能
CONT	1	R/W	0x0	连续转换选择位 0: 单次转换 1: 连续转换
ADEN	0	R/W	0x0	ADC 使能位 0: 禁止 1: 使能

1. 9. 5 ADC 采样时间寄存器 1 (ADC_SMPR1)

	Address offset:	10H

		Reset value:				00000000_00000000_00000000_00000000 _B									
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								SMP11		SMP10		SMP9		SMP8	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP7		SMP6		SMP5		SMP4		SMP3		SMP2		SMP1		SMP0	

Name	Bit	Type	Reset value	Description
Reserved	31:24	N/A	0x00	保留
SMP11	23:22	R/W	0x0	通道 11 采样时间选择位 00: 1 个周期 01: 2 个周期 10: 4 个周期 11: 15 个周期
SMP10	21:20	R/W	0x0	通道 10 采样时间选择位 00: 1 个周期 01: 2 个周期 10: 4 个周期 11: 15 个周期
SMP9	19:18	R/W	0x0	通道 9 采样时间选择位 00: 1 个周期 01: 2 个周期 10: 4 个周期 11: 15 个周期
SMP8	17:16	R/W	0x0	通道 8 采样时间选择位 00: 1 个周期 01: 2 个周期 10: 4 个周期 11: 15 个周期
SMP7	15:14	R/W	0x0	通道 7 采样时间选择位 00: 1 个周期 01: 2 个周期 10: 4 个周期 11: 15 个周期
SMP6	13:12	R/W	0x0	通道 6 采样时间选择位 00: 1 个周期 01: 2 个周期 10: 4 个周期 11: 15 个周期
SMP5	11:10	R/W	0x0	通道 5 采样时间选择位 00: 1 个周期 01: 2 个周期 10: 4 个周期 11: 15 个周期
SMP4	9:8	R/W	0x0	通道 4 采样时间选择位 00: 1 个周期 01: 2 个周期 10: 4 个周期 11: 15 个周期
SMP3	7:6	R/W	0x0	通道 3 采样时间选择位 00: 1 个周期 01: 2 个周期 10: 4 个周期 11: 15 个周期
SMP2	5:4	R/W	0x0	通道 2 采样时间选择位

SMP1	3:2	R/W	0x0	00: 1 个周期 01: 2 个周期 10: 4 个周期 11: 15 个周期
				通道 1 采样时间选择位 00: 1 个周期 01: 2 个周期 10: 4 个周期 11: 15 个周期
SMP0	1:0	R/W	0x0	通道 0 采样时间选择位 00: 1 个周期 01: 2 个周期 10: 4 个周期 11: 15 个周期

1. 9. 6 ADC 注入通道数据偏移寄存器 1 (ADC_JOFR1)

Address offset:		18 _H													
Reset value:		00000000_00000000_00000000_00000000 _B													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				JOFF1											

Name	Bit	Type	Reset value	Description
Reserved	31:12	N/A	0x00	保留
JOFF1	11:0	R/W	0x0	注入通道 1 数据偏移量 ADC_JDR1 中的数据为原始转换数据加上偏移量

1. 9. 7 ADC 注入通道数据偏移寄存器 2 (ADC_JOFR2)

Address offset:		1C _H													
Reset value:		00000000_00000000_00000000_00000000 _B													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				JOFF2											

Name	Bit	Type	Reset value	Description
Reserved	31:12	N/A	0x00	保留
JOFF2	11:0	R/W	0x0	注入通道 2 数据偏移量 ADC_JDR2 中的数据为原始转换数据加上偏移量

1. 9. 8 ADC 注入通道数据偏移寄存器 3 (ADC_JOFR3)

		Address offset:		20 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				JOFF3											

Name	Bit	Type	Reset value	Description
Reserved	31:12	N/A	0x00	保留
JOFF3	11:0	R/W	0x0	注入通道 3 数据偏移量 ADC_JDR3 中的数据为原始转换数据加上偏移量

1. 9. 9 ADC 注入通道数据偏移寄存器 4 (ADC_JOFR4)

		Address offset:		24 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				JOFF4											

Name	Bit	Type	Reset value	Description
Reserved	31:12	N/A	0x00	保留
JOFF4	11:0	R/W	0x0	注入通道 4 数据偏移量 ADC_JDR4 中的数据为原始转换数据加上偏移量

1. 9. 10 ADC 看门狗高阈值寄存器 (ADC_HTR)

		Address offset:		28 _H											
		Reset value:		00000000_00000000_00001111_11111111 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				HT											

Name	Bit	Type	Reset value	Description

Reserved	31:12	N/A	0x00	保留
HT	11:0	R/W	0x0	模拟看门狗高阈值 当原始数据大于高阈值时产生看门狗事件

1. 9. 11 ADC 看门狗低阈值寄存器 (ADC_LTR)

Address offset:		2C _H													
Reset value:		00000000_00000000_00000000_00000000 _B													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				LT											

Name	Bit	Type	Reset value	Description
Reserved	31:12	N/A	0x00	保留
LT	11:0	R/W	0x0	模拟看门狗低阈值 当原始数据小于低阈值时产生看门狗事件

1. 9. 12 ADC 规则转换序列寄存器 1 (ADC_RSQR1)

Address offset:		30 _H													
Reset value:		00000000_00000000_00000000_00000000 _B													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RSQ8				RSQ7				RSQ6				RSQ5			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSQ4				RSQ3				RSQ2				RSQ1			

Name	Bit	Type	Reset value	Description
RSQ8	31:28	R/W	0x0	规则序列第 8 次转换通道编号 0000~1011: 通道 0~11 其他: 预留
RSQ7	27:24	R/W	0x0	规则序列第 7 次转换通道编号 0000~1011: 通道 0~11 其他: 预留
RSQ6	23:20	R/W	0x0	规则序列第 6 次转换通道编号 0000~1011: 通道 0~11 其他: 预留
RSQ5	19:16	R/W	0x0	规则序列第 5 次转换通道编号 0000~1011: 通道 0~11 其他: 预留
RSQ4	15:12	R/W	0x0	规则序列第 4 次转换通道编号 0000~1011: 通道 0~11 其他: 预留

RSQ3	11:8	R/W	0x0	规则序列第 3 次转换通道编号 0000~1011: 通道 0~11 其他: 预留
RSQ2	7:4	R/W	0x0	规则序列第 2 次转换通道编号 0000~1011: 通道 0~11 其他: 预留
RSQ1	3:0	R/W	0x0	规则序列第 1 次转换通道编号 0000~1011: 通道 0~11 其他: 预留

1. 9. 13 ADC 规则转换序列寄存器 2 (ADC_RSQR2)

		Address offset:		34 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RSQ12				RSQ11				RSQ10				RSQ9			

Name	Bit	Type	Reset value	Description
Reserved	31:16	R/W	0x0	保留
RSQ12	15:12	R/W	0x0	规则序列第 12 次转换通道编号 0000~1011: 通道 0~11 其他: 预留
RSQ11	11:8	R/W	0x0	规则序列第 11 次转换通道编号 0000~1011: 通道 0~11 其他: 预留
RSQ10	7:4	R/W	0x0	规则序列第 10 次转换通道编号 0000~1011: 通道 0~11 其他: 预留
RSQ9	3:0	R/W	0x0	规则序列第 9 次转换通道编号 0000~1011: 通道 0~11 其他: 预留

1. 9. 14 ADC 注入转换序列寄存器 (ADC_JSQR)

		Address offset:		40 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				JSQ4				Reserved				JSQ3			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				JSQ2				Reserved				JSQ1			

Name	Bit	Type	Reset value	Description
Reserved	31:28	N/A	0x00	保留
JSQ4	27:24	R/W	0x0	注入序列第 4 次转换通道编号 0000~1011: 通道 0~11 其他: 预留
Reserved	23:20	N/A	0x00	保留
JSQ3	19:16	R/W	0x0	注入序列第 3 次转换通道编号 0000~1011: 通道 0~11 其他: 预留
Reserved	15:12	N/A	0x00	保留
JSQ2	11:8	R/W	0x0	注入序列第 2 次转换通道编号 0000~1011: 通道 0~11 其他: 预留
Reserved	7:4	N/A	0x00	保留
JSQ1	3:0	R/W	0x0	注入序列第 1 次转换通道编号 0000~1011: 通道 0~11 其他: 预留

1. 9. 15 ADC 转换序列长度寄存器 (ADC_SQR)

		Address offset:			44 _H										
		Reset value:			00000000_00000000_00000000_00000000 _B										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						JSQ1		Reserved				RSQL			

Name	Bit	Type	Reset value	Description
Reserved	31:10	N/A	0x00	保留
JSQ1	9:8	R/W	0x0	注入转换序列长度选择位 00: 1 次转换 01: 2 次转换 10: 3 次转换 11: 4 次转换
Reserved	7:4	N/A	0x00	保留
RSQL	3:0	R/W	0x0	规则转换序列长度选择位 0000: 1 次转换 0001: 2 次转换 0111: 8 次转换 1000: 9 次转换 1001: 10 次转换 1010: 11 次转换 1011: 12 次转换 其他: 预留

1. 9. 16 ADC 注入通道数据寄存器 1 (ADC_JDR1)

		Address offset:			48 _H										
		Reset value:			00000000_00000000_00000000_00000000 _B										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JDATA															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留
JDATA	15:0	R	0x0	注入通道 1 转换数据 该数据为对齐之后的数据

1. 9. 17 ADC 注入通道数据寄存器 2 (ADC_JDR2)

		Address offset:			4C _H										
		Reset value:			00000000_00000000_00000000_00000000 _B										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JDATA															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留
JDATA	15:0	R	0x0	注入通道 2 转换数据 该数据为对齐之后的数据

1. 9. 18 ADC 注入通道数据寄存器 3 (ADC_JDR3)

		Address offset:			50 _H										
		Reset value:			00000000_00000000_00000000_00000000 _B										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JDATA															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留
JDATA	15:0	R	0x0	注入通道 3 转换数据

该数据为对齐之后的数据

1. 9. 19 ADC 注入通道数据寄存器 4 (ADC_JDR4)

		Address offset:		54 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JDATA															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留
JDATA	15:0	R	0x0	注入通道 4 转换数据 该数据为对齐之后的数据

1. 9. 20 ADC 规则通道数据寄存器 (ADC_RDR)

		Address offset:		64 _H											
		Reset value:		00000000_00000000_00000000_00000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RDATA															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留
RDATA	15:0	R	0x0	规则通道转换数据 该数据为对齐之后的数据

1. 9. 21 ADC 通用控制寄存器 (ADC_CCR)

		Address offset:		68 _H											
		Reset value:		10000000_00000000_00100000_01000000 _B											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
OCALV								GCALV							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VBUF EN	ADBP	MSBCAL		GCAL	LPCT L	OSCA L	VCM EN	VREF EN	VRSEL			TRIM _EN	CKDIV		

Name	Bit	Type	Reset value	Description
OCALV	31:24	R/W	0x80	offset 校准控制位 0: 断开 1: 负参考 VREFN 输入 ADC 转换
GCALV	23:16	R/W	0x0	增益校准控制位 0: 断开 1: 参考电压输入 ADC 转换
VBUFEN	15	R/W	0x0	内部参考电压 Buffer 使能位 0: 禁止 1: 使能
ADBP	14	R/W	0x0	内部参考电压 Bypass 模式控制 0: 使能内部 Buffer 1: 禁用内部 Buffer
MSBCAL	13:12	R/W	0x2	ADC 最高位电容校准 10: default
GCAL	11	R/W	0x0	增益校准使能位 0: 禁止 1: 使能
LPCTL	10	R/W	0x0	低功耗控制位 0: 禁止 1: 使能
OSCAL	9	R/W	0x0	偏置校准使能位 0: 禁止 1: 使能
VCMEN	8	R/W	0x0	共模电压使能位 0: 禁止 1: 使能
VREFEN	7	R/W	0x0	内部参考电压使能位 0: 禁止 1: 使能
VRSEL	6:4	R/W	0x4	参考电压选择位 001: 电源做位参考电压 010: 外部输入作为参考电压 100: 内部参考电压
TRIM_EN	3	R/W	0x0	ADC 数据修调使能位 0: 禁止 1: 使能
CKDIV	2:0	R/W	0x0	ADC 时钟分频选择位 000: 1 分频 001: 2 分频 111: 128 分频

第17章 实时时钟（RTC）

1.1 RTC 简介

Real Time Clock (RTC)可提供使用者准确的时间以及日期资讯，这些资讯皆以 BCD 的格式储存在 RTC 控制暂存器内，同时提供使用者自行设定闹铃功能。RTC 的 CLK 来源也分为 2 种供使用者选择，这 2 种 CLK 分别为外部 32.768K Hz 的晶体震荡器、内部约 32.768K Hz 的 RC 震荡器。当 MCU 处在低功耗模式时，RTC 仍可继续提供准确的时间资讯。

1.2 RTC 主要特性

- ◆ 支持日历显示：年、月、日。
- ◆ 支持时间显示：星期、小时、分钟、秒。
- ◆ 日历与时间皆以 BCD 格式储存。
- ◆ 支持闰年侦测功能。
- ◆ 支持用户自行设定 RTC 计数器校正功能。
- ◆ 支持用户自行设定闹铃中断功能。
- ◆ 支持年、月、日、星期、小时、分钟、秒的翻转(Rollover)中断。
- ◆ 支持 MCU 在睡眠模式下保持计数。
- ◆ 支持用户自行配置睡眠计数器。
- ◆ 支持最长 16777216 秒(194 天)的睡眠计数。
- ◆ 支持最短 1/16 秒的睡眠计数。
- ◆ 支持侦测日历、时间是否已被清除。

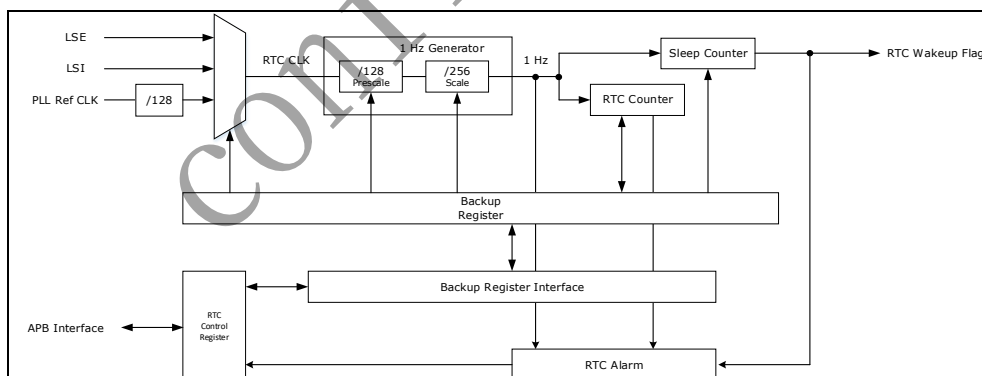


图 17-1 RTC 结构框图

RTC 主要分为 2 个部分，第一个部分为 RTC 计数器，第二个部分为 RTC 时间储存与闹铃。RTC 计数器的部份主要负责计算日期以及时间，并于 MCU 处在睡眠模式时提供唤醒 CPU 的功能。而另一个部分则是提供日期以及时间的读取，同时依据使用者所设定的闹铃发出中断。

1.3 RTC 功能描述

1.3.1 设定并开启 RTC

在开启 RTC 计数之前，必须先对 RTC_CTRL 寄存器进行设定，并于 RTC_CAL 填入日期以及在 RTC_TIME 填入时间，日期以及时间皆须以 BCD 格式填入。使用者可设定 CKSEL 来选定 RTC CLK 的来源为 LSE、LSI 或是 PLL 参考 CLK，须注意的是当 MCU 在睡眠模式时，是无法选择 PLL 参考 CLK 当作 RTC CLK 的来源。紧接着必须依据所选的 RTC CLK 来源设定 PSCALE 与 SCALE 来产生不同的除频倍率，藉此产生准确的 1 秒(1 Hz)讯号，在预设情况下这 2 个寄存器是将 RTC CLK 除上 32768 来产生 1 秒讯号。

当设定完成后，即可设定 RTC_EN 来开启 RTC 的功能。在 RTC 计数期间，为了确保日期以及时间不能不随意更改，因此会暂时锁定 RTC_CAL 以及 RTC_TIME 的写入功能，若需要更改 RTC 的日期以及时间，则必须关闭 RTC 计数功能以后才能够进行修改。

1.3.2 读取 RTC 日期与时间

由于 RTC 日期与时间的计数是位在 Backup Register 区，因此每一秒皆须将日期以及时间同步回 APB 的暂存器，为了确保使用者不会读取到错误的时间资讯，在读取 RTC 日期以及时间之前，可以先去检查 RTC_STA 暂存器内的 SYNDONE 讯号，当此讯号被设定为 1 时代表已同步完成，若此讯号为 0 则代表 RTC 正在同步 Backup Register 区的时间资讯。

此外，使用者也可以设定 RTC_BKEN 来开启直接读取 Backup Register 内所保存的日期以及时间的功能。若藉由 Backup Register 接口直接读取日期与时间的话，需等待 19 个 APB CLK 的读取时间。读取完毕以后，需自行手动关闭 RTC_BKEN。

当 MCU 进入睡眠模式，并关闭 APB 区的电源时，会让 RTC 控制寄存器内的时间资讯被清除，因此当 MCU 被唤醒时若要读取日期以及时间资讯的话，需要额外等待 1 秒的时间让 RTC 同步 Backup Register 区的时间资讯，或是直接透过 Backup Register 接口读取日期与时间。

1.3.3 RTC 校正

由于 RTC CLK 的来源并非准确的 32.768K Hz，因此需要针对所使用的 CLK 进行频率补偿，以确保能计算出准确的 1 秒。校正数值的计算方式如下：

1. 每秒校正数值 = 实际频率整数 - 预期频率整数。若此数值为正值代表实际频率高于预期频率，需要增加计数的次数来算出准确的 1 秒；若此数值为负值代表实际频率低于预期频率，需要减少计数的次数来算出准确的 1 秒。
2. 每分钟校正数值 = (实际频率小数 - 预期频率小数) x 60。若此数值为正值，则需要增加计数的次数，反之则需要减少计数的次数。

当计算完校正数值以后，则依据数值的正负号决定要增加校正数值或是减少校正数值。校正的周期可以选择每秒校正或是每分钟校正。下列为 RTC 校正范例，校正周期接设定为每秒校正。

1.3.3.1 RTC 校正范例 1(实际频率高于预期频率)

预期频率：32768Hz

实际频率：32770.5Hz

每秒校正数值：32770 - 32768 = +2

每分钟校正值: $0.5 \times 60 = +30$

校正步骤如下:

1. 设定 RTC_CTRL(0x0)内的 CYCLE 为 1, 开启每秒校正功能, 并设定 CALVAL 的数值为 2。
2. 设定 MODE 为 0, 开启每秒增加校正数值。
3. 设定 CALIB 开启 RTC 校正功能。
4. 当第 60 秒发生时, 更改 RTC 校正数值 CALVAL 为 $(2+30) = 32$, 并于下一秒将校正数值更改回 2。
5. 重复第 4 步校正流程。

1.3.3.2 RTC 校正范例 2(实际频率低于预期频率)

预期频率: 32768Hz

实际频率: 32766.5Hz

每秒校正数值: $32766 - 32768 = -2$

每分钟校正值: $0.5 \times 60 = +30$

校正步骤如下:

6. 设定 RTC_CTRL(0x0)内的 CYCLE 为 1, 并设定 CALVAL 的数值为 2。
7. 设定 MODE 为 1, 开启每秒减少校正数值。
8. 设定 CALIB 开启 RTC 每秒校正功能。
9. 当第 60 秒发生时, 更改 RTC 校正数值 CALVAL 为 $(-2+30) = 28$ 同时更改 MODE 为 0 开启每秒增加校正数值, 并于下一秒将校正数值更改回 2, 同时更改 MODE 为 1, 回复每秒减少校正数值。
10. 重複第 4 步校正流程。

1.3.4 RTC 睡眠计数器

RTC 的睡眠计数器可由使用者自行设定 RTC_WKUP(0x04)内的 WKSEL 来决定开启计数的时间点, 当使用者设定 WKSEL 为 0x0 时代表不使用 RTC 的睡眠计数器; 当设定 WKSEL 为 0x1 时, 此时睡眠计数器可当成一般计数器使用, 使用者可藉由设定 WKCAL 来决定要计数多少秒, 当计数完成时可发出中断讯号告知。当设定 WKSEL 为 0x2 时, 睡眠计数器会在收到 CPU 发出的 SLEEP 讯号以后开始计数, 并于计数完成时唤醒 MCU。计数器的最大值可设定至 16777216 秒, 即 194 天。此外, RTC 的睡眠计数器也可以设定计数週期。使用者可以藉由设定 WKSCAL 来决定睡眠计数器的计数週期, 分别为 1/2 秒、1/4 秒、1/8 秒或是 1/16 秒计数一次。

1.3.5 RTC 闹铃

RTC 的闹铃会在 RTC 的日期或时间符合设定时发出中断告知使用者。使用者可设定 RTC_ALEN(0x18)开启 RTC 闹铃的种类, 并于 RTC_ALTIME(0x10)以及 RTC_ALCAL(0x14)

设定产生闹铃的时间。闹铃的种类分为秒闹铃、分钟闹铃、小时闹铃、星期闹铃、日闹铃、月闹铃、年闹铃以及日期与时间完全相符的闹铃。

1.3.6 RTC 触发 ADC 或 DAC

RTC 允许在中断条件满足时送出 ADC 以及 DAC 的触发讯号，使用者可设定 RTC_TRIG(0x1C)暂存器来决定要送出触发讯号的条件为何，此触发讯号仅会维持一个 APB CLK 的宽度。

1.4 特殊寄存器

1.4.1 RTC 控制寄存器(RTC_CTRL)

		Address offset:		0x00											
		Reset value:		0x00FF 7F00											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	MODE	CALVAL[5:0]						SCALE[7:0]							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	PSCALE[6:0]						Reserved		CYCLE	CALIB	CKSEL		Reserved	RTCE N	

Name	Bit	Type	Reset value	Description
Reserved	31	N/A	0x0	保留，保持复位值
MODE	30	R/W	0x0	RTC Clock Calibration Mode 1: 当匹配校准周期时，降低校准值。 0: 当匹配校准周期时，增加校准值。
CALVAL	29:24	R/W	0x0	RTC Clock Calibration Value 最大校准值为 CALV.
SCALE	23:16	R/W	0xFF	RTC Clock Counter Scale 计数器的最大除数是(SCALE+1).
Reserved	15	N/A	0x0	保留，保持复位值
PSCALE	14:8	R/W	0x7F	RTC Clock Counter Prescale 计数器的最大除数是(PSCALE+1).
Reserved	7:6	N/A	0x0	保留，保持复位值
CYCLE	5	R/W	0x0	RTC Calibration Cycle Select 0: 启用校准功能时，校准周期为 60s。 1: 启用校准功能时，校准周期为 1s
CALIB	4	R/W	0x0	RTC Calibration Function Enable 0: 校准功能禁止 1: 校准功能使能
CKSEL	3:2	R/W	0x0	RTC Clock Source Selection 01: LSE 时钟作为 RTC 时钟 10: LSI 时钟作为 RTC 时钟 00, 11: 保留
Reserved	1	N/A	0x0	保留，保持复位值
RTCE N	0	R/W	0x0	RTC Counter Enable 0: 禁止 RTC 1: 使能 RTC

1. 4. 2 RTC 唤醒计数器寄存器(RTC_WKUP)

		Address offset:		0x04											
		Reset value:		0x00FF FFFF											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		WKSCAL[3:0]				WKSEL[1:0]		WKCAL[23:16]							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WKCAL[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:30	N/A	0x0	保留, 保持复位值
WKSCAL	29:26	R/W	0x0	RTC Wakeup Counter Scale Value 0x0: 除 1, 唤醒计数器每 1 秒计数一次。 0x1: 除 2, 唤醒计数器每 1/2 秒计数一次。 0x2: 除 4, 唤醒计数器每 1/4 秒计数一次。 0x4: 除 8, 唤醒计数器每 1/8 秒计数一次。 0x8: 除 16, 唤醒计数器每 1/16 秒计数一次。
WKSEL	25:24	R/W	0x0	RTC Wakeup Counter Events Selet 00: 关 01: RTC 启动时启动计数器。 10: 接收休眠标志产生时启动计数器。 11: 保留
WKCAL	23:0	R/W	0xFFFFFFFF	RTC Wakeup Counter seting value 最大值为 WKCAL 秒。

1. 4. 3 RTC 日历时间寄存器(RTC_TIME)

		Address offset:		0x08											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				WEEK[2:0]			Reserved			HOUR_T[1:0]		HOUR_U[3:0]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	MIN_T[2:0]			MIN_U[3:0]			Reserved	SEC_T[2:0]			SEC_U[3:0]				

Name	Bit	Type	Reset value	Description
Reserved	31:27	N/A	0x0	保留, 保持复位值
WEEK	26:24	R/W	0x0	Week Value 周, 以 BCD 格式存储
Reserved	23:22	N/A	0x0	保留, 保持复位值
HOUR_T	21:20	R/W	0x0	Hour Tens Value 时, 十位值以 BCD 格式存储
HOUR_U	19:16	R/W	0x0	Hour Units Value 时, 个位值以 BCD 格式存储
Reserved	15	N/A	0x0	保留, 保持复位值
MIN_T	14:12	R/W	0x0	Minute Tens Value 分, 十位值以 BCD 格式存储。

MIN_U	11:8	R/W	0x0	Minute 分, 个位值以 BCD 格式存储。	Units	Value
Reserved	7	N/A	0x0	保留, 保持复位值		
SEC_T	6:4	R/W	0x0	Second 秒, 十位值以 BCD 格式存储。	Tens	Value
SEC_U	3:0	R/W	0x0	Second 秒, 个位值以 BCD 格式存储。	Units	Value

1. 4. 4 RTC 日历日期寄存器(RTC_CAL)

		Address offset:		0x0C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								YEAR_T[3:0]				YEAR_U[3:0]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			MON_T		MON_U[3:0]			Reserved		DATE_T[1:0]		DATE_U[3:0]			

Name	Bit	Type	Reset value	Description
Reserved	31:24	N/A	0x0	保留, 保持复位值
YEAR_T	23:20	R/W	0x0	Year 年, 十位值以 BCD 格式存储。
YEAR_U	19:16	R/W	0x0	Year 年, 个位值以 BCD 格式存储。
Reserved	15:13	N/A	0x0	保留, 保持复位值
MON_T	12	R/W	0x0	Month 月, 十位值以 BCD 格式存储。
MON_U	11:8	R/W	0x0	Month 月, 个位值以 BCD 格式存储。
Reserved	7:6	N/A	0x0	保留, 保持复位值
DATE_T	5:4	R/W	0x0	Date 日, 十位值以 BCD 格式存储。
DATE_U	3:0	R/W	0x0	Date 日, 个位值以 BCD 格式存储。

1. 4. 5 RTC 闹钟时间寄存器(RTC_ALTIME)

		Address offset:		0x10											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					WEEK[2:0]			Reserved		HOUR_T[1:0]		HOUR_U[3:0]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	MIN_T[2:0]			MIN_U[3:0]			Reserved	SEC_T[2:0]			SEC_U[3:0]				

Name	Bit	Type	Reset value	Description

Reserved	31:27	N/A	0x0	保留, 保持复位值			
WEEK	26:24	R/W	0x0	Alarm 周警报, 以 BCD 格式存储。	Week		Value
Reserved	23:22	N/A	0x0	保留, 保持复位值			
HOUR_T	21:20	R/W	0x0	Alarm 小时警报, 十位值。以 BCD 格式存储。	Hour	Tens	Value
HOUR_U	19:16	R/W	0x0	Alarm 小时警报, 个位值。以 BCD 格式存储。	Hour	Units	Value
Reserved	15	N/A	0x0	保留, 保持复位值			
MIN_T	14:12	R/W	0x0	Alarm 分警报, 十位值。以 BCD 格式存储。	Minute	Tens	Value
MIN_U	11:8	R/W	0x0	Alarm 分警报, 个位值。以 BCD 格式存储。	Minute	Units	Value
Reserved	7	N/A	0x0	保留, 保持复位值			
SEC_T	6:4	R/W	0x0	Alarm 秒警报, 十位值。以 BCD 格式存储。	Second	Tens	Value
SEC_U	3:0	R/W	0x0	Alarm 秒警报, 个位值。以 BCD 格式存储。	Second	Units	Value

1. 4. 6 RTC 闹钟日期寄存器(RTC_ALCAL)

		Address offset:		0x14											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								YEAR_T[3:0]				YEAR_U[3:0]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			MON_T	MON_U[3:0]				Reserved		DATE_T[1:0]		DATE_U[3:0]			

Name	Bit	Type	Reset value	Description
Reserved	31:24	N/A	0x0	保留, 保持复位值
YEAR_T	23:20	R/W	0x0	Alarm Year Tens Value 年警报, 十位值。以 BCD 格式存储。
YEAR_U	19:16	R/W	0x0	Alarm Year 年警报, 个位值。以 BCD 格式存储。
Reserved	15:13	N/A	0x0	保留, 保持复位值
MON_T	12	R/W	0x0	Alarm Month 月警报, 十位值。以 BCD 格式存储。
MON_U	11:8	R/W	0x0	Alarm Month 月警报, 个位值。以 BCD 格式存储。
Reserved	7:6	N/A	0x0	保留, 保持复位值
DATE_T	5:4	R/W	0x0	Alarm Date 日警报, 十位值。以 BCD 格式存储。
DATE_U	3:0	R/W	0x0	Alarm Date 日警报, 个位值。以 BCD 格式存储。

1. 4. 7 RTC 闹钟使能寄存器(RTC_ALEN)

		Address offset:		0x18											

		Reset value:				0x0000 0000									
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									YEAR	MON TH	DATE	WEE K	HOU R	MIN	SEC

Name	Bit	Type	Reset value	Description
Reserved	31:7	N/A	0x0	保留，保持复位值
YEAR	6	R/W	0x0	Year 0 1: 使能
MONTH	5	R/W	0x0	Month 0 1: 使能
DATE	4	R/W	0x0	Date 0 1: 使能
WEEK	3	R/W	0x0	Week 0 1: 使能
HOUR	2	R/W	0x0	Hour 0 1: 使能
MIN	1	R/W	0x0	Minute 0 1: 使能
SEC	0	R/W	0x0	Second 0 1: 使能

1. 4. 8 RTC 触发源选择寄存器(RTC_TRIG)

		Address offset:				0x1C									
		Reset value:				0x0000 0000									
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															WKTM
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F1HZ	RYEAR	RMON	RDATE	RWEEK	RHOUR	RMIN	RSEC	AMALL	AYEAR	AMON	ADATE	AWEEK	AHOUR	AMIN	ASEC

Name	Bit	Type	Reset value	Description
Reserved	31:17	N/A	0x0	保留，保持复位值
WKTM	16	R/W	0x0	RTC Wakeup Counter Match Trigger Enable 0 : 禁 止 1: 使能
F1HZ	15	R/W	0x0	RTC 1 Hz Trigger Enable 0 : 禁 止 1: 使能

RYEAR	14	R/W	0x0	RTC 0 1: 使能	Year	Roll :	Alarm	Trigger 禁	Enable 止
RMON	13	R/W	0x0	RTC 0 1: 使能	Month	Roll :	Alarm	Trigger 禁	Enable 止
RDATE	12	R/W	0x0	RTC 0 1: 使能	Date	Roll :	Alarm	Trigger 禁	Enable 止
RWEEK	11	R/W	0x0	RTC 0 1: 使能	Week	Roll :	Alarm	Trigger 禁	Enable 止
RHOUR	10	R/W	0x0	RTC 0 1: 使能	Hour	Roll :	Alarm	Trigger 禁	Enable 止
RMIN	9	R/W	0x0	RTC 0 1: 使能	Minute	Roll :	Alarm	Trigger 禁	Enable 止
RSEC	8	R/W	0x0	RTC 0 1: 使能	Second	Roll :	Alarm	Trigger 禁	Enable 止
AMALL	7	R/W	0x0	RTC 0 1: 使能	Match User should enable all alarm function enable in RTC_ALEN when selecting this trigger	All :	Alarm	Trigger 禁	Enable 止
AYEAR	6	R/W	0x0	RTC 0 1: 使能	Match Year	Roll :	Alarm	Trigger 禁	Enable 止
AMON	5	R/W	0x0	RTC 0 1: 使能	Match Month	Roll :	Alarm	Trigger 禁	Enable 止
ADTAE	4	R/W	0x0	RTC 0 1: 使能	Match Date	Roll :	Alarm	Trigger 禁	Enable 止
AWEEK	3	R/W	0x0	RTC 0 1: 使能	Match Week	Roll :	Alarm	Trigger 禁	Enable 止
AHOUR	2	R/W	0x0	RTC 0 1: 使能	Match Hour	Roll :	Alarm	Trigger 禁	Enable 止
AMIN	1	R/W	0x0	RTC 0 1: 使能	Match Minute	Roll :	Alarm	Trigger 禁	Enable 止
ASEC	0	R/W	0x0	RTC 0 1: 使能	Match Second	Roll :	Alarm	Trigger 禁	Enable 止

1. 4. 9 RTC 中断使能寄存器(RTC_IER)

		Address offset:		0x20											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															WKT M
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

F1HZ	RYEAR	RMONTH	RDAT E	RWE EK	RHO UR	RMIN	RSEC	AMALL	AYEAR	AMON	ADAT E	AWE EK	AHO UR	AMIN	ASEC

Name	Bit	Type	Reset value	Description
Reserved	31:17	N/A	0x0	保留，保持复位值
WKT M	16	WO	0x0	RTC Wakeup Counter Match Interrupt Enable 0: 无效 1: 使能中断
F1HZ	15	WO	0x0	RTC 1 Hz Interrupt Enable 0: 无效 1: 使能中断
RYEAR	14	WO	0x0	RTC Year Roll Alarm Interrupt Enable 0: 无效 1: 使能中断
RMON	13	WO	0x0	RTC Month Roll Alarm Interrupt Enable 0: 无效 1: 使能中断
RDATE	12	WO	0x0	RTC Date Roll Alarm Interrupt Enable 0: 无效 1: 使能中断
RWEEK	11	WO	0x0	RTC Week Roll Alarm Interrupt Enable 0: 无效 1: 使能中断
RHO UR	10	WO	0x0	RTC Hour Roll Alarm Interrupt Enable 0: 无效 1: 使能中断
RMIN	9	WO	0x0	RTC Minute Roll Alarm Interrupt Enable 0: 无效 1: 使能中断
RSEC	8	WO	0x0	RTC Second Roll Alarm Interrupt Enable 0: 无效 1: 使能中断
AMALL	7	WO	0x0	RTC Match All Alarm Interrupt Enable 0: 无效 1: 使能中断
AYEAR	6	WO	0x0	RTC Match Year Alarm Interrupt Enable 0: 无效 1: 使能中断
AMON	5	WO	0x0	RTC Match Month Alarm Interrupt Enable 0: 无效 1: 使能中断
ADTAE	4	WO	0x0	RTC Match Date Alarm Interrupt Enable 0: 无效 1: 使能中断
AWE EK	3	WO	0x0	RTC Match Week Alarm Interrupt Enable 0: 无效 1: 使能中断
AHO UR	2	WO	0x0	RTC Match Hour Alarm Interrupt Enable 0: 无效 1: 使能中断
AMIN	1	WO	0x0	RTC Match Minute Alarm Interrupt Enable 0: 无效 1: 使能中断
ASEC	0	WO	0x0	RTC Match Second Alarm Interrupt Enable 0: 无效 1: 使能中断

1. 4. 10 RTC 中断禁止寄存器(RTC_IDR)

		Address offset:		0x24											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															WKT M
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F1HZ	RYEAR	RMON	RDATE	RWEEK	RHOUR	RMIN	RSEC	AMALL	AYEAR	AMON	ADTAE	AWEEK	AHOUR	AMIN	ASEC

Name	Bit	Type	Reset value	Description
Reserved	31:17	N/A	0x0	保留，保持复位值
WKT M	16	WO	0x0	RTC Wakeup Counter Match Interrupt Disable 0: 无效 1: 禁止中断
F1HZ	15	WO	0x0	RTC 1 Hz Interrupt Disable 0: 无效 1: 禁止中断
RYEAR	14	WO	0x0	RTC Year Roll Alarm Interrupt Disable 0: 无效 1: 禁止中断
RMON	13	WO	0x0	RTC Month Roll Alarm Interrupt Disable 0: 无效 1: 禁止中断
RDATE	12	WO	0x0	RTC Date Roll Alarm Interrupt Disable 0: 无效 1: 禁止中断
RWEEK	11	WO	0x0	RTC Week Roll Alarm Interrupt Disable 0: 无效 1: 禁止中断
RHOUR	10	WO	0x0	RTC Hour Roll Alarm Interrupt Disable 0: 无效 1: 禁止中断
RMIN	9	WO	0x0	RTC Minute Roll Alarm Interrupt Disable 0: 无效 1: 禁止中断
RSEC	8	WO	0x0	RTC Second Roll Alarm Interrupt Disable 0: 无效 1: 禁止中断
AMALL	7	WO	0x0	RTC Match All Alarm Interrupt Disable 0: 无效 1: 禁止中断
AYEAR	6	WO	0x0	RTC Match Year Alarm Interrupt Disable 0: 无效 1: 禁止中断
AMON	5	WO	0x0	RTC Match Month Alarm Interrupt Disable 0: 无效 1: 禁止中断
ADTAE	4	WO	0x0	RTC Match Date Alarm Interrupt Disable 0: 无效 1: 禁止中断
AWEEK	3	WO	0x0	RTC Match Week Alarm Interrupt Disable 0: 无效 1: 禁止中断

AHOUR	2	WO	0x0	0: 无效 1: 禁止中断	RTC Match	Hour	Alarm	Interrupt	Disable
				0: 无效 1: 禁止中断					
AMIN	1	WO	0x0	0: 无效 1: 禁止中断	RTC Match	Minute	Alarm	Interrupt	Disable
				0: 无效 1: 禁止中断					
ASEC	0	WO	0x0	0: 无效 1: 禁止中断	RTC Match	Second	Alarm	Interrupt	Disable
				0: 无效 1: 禁止中断					

1. 4. 11 RTC 中断有效状态寄存器(RTC_IVS)

Address offset:		0x28													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															WKT M
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F1HZ	RYEAR	RMON	RDATE	RWEEK	RHOUR	RMIN	RSEC	AMALL	AYEAR	AMON	ADATE	AWEEK	AHOUR	AMIN	ASEC

Name	Bit	Type	Reset value	Description
Reserved	31:17	N/A	0x0	保留，保持复位值
WKT M	16	RO	0x0	RTC Wakeup Counter Match Interrupt Valid Status 0: 中 断 禁 止 1: 中断使能
F1HZ	15	RO	0x0	RTC 1 Hz Interrupt Valid Status 0: 中 断 禁 止 1: 中断使能
RYEAR	14	RO	0x0	RTC Year Roll Alarm Interrupt Valid Status 0: 中 断 禁 止 1: 中断使能
RMON	13	RO	0x0	RTC Month Roll Alarm Interrupt Valid Status 0: 中 断 禁 止 1: 中断使能
RDATE	12	RO	0x0	RTC Date Roll Alarm Interrupt Valid Status 0: 中 断 禁 止 1: 中断使能
RWEEK	11	RO	0x0	RTC Week Roll Alarm Interrupt Valid Status 0: 中 断 禁 止 1: 中断使能
RHOUR	10	RO	0x0	RTC Hour Roll Alarm Interrupt Valid Status 0: 中 断 禁 止 1: 中断使能
RMIN	9	RO	0x0	RTC Minute Roll Alarm Interrupt Valid Status 0: 中 断 禁 止 1: 中断使能
RSEC	8	RO	0x0	RTC Second Roll Alarm Interrupt Valid Status 0: 中 断 禁 止 1: 中断使能

AMALL	7	RO	0x0	RTC 0: 中断使能 1: 中断使能	Match 中	All	Alarm 断	Interrupt 禁	Valid 禁	Status 止
AYEAR	6	RO	0x0	RTC 0: 中断使能 1: 中断使能	Match 中	Year	Alarm 断	Interrupt 禁	Valid 禁	Status 止
AMON	5	RO	0x0	RTC 0: 中断使能 1: 中断使能	Match 中	Month	Alarm 断	Interrupt 禁	Valid 禁	Status 止
ADTAE	4	RO	0x0	RTC 0: 中断使能 1: 中断使能	Match 中	Date	Alarm 断	Interrupt 禁	Valid 禁	Status 止
AWEEK	3	RO	0x0	RTC 0: 中断使能 1: 中断使能	Match 中	Week	Alarm 断	Interrupt 禁	Valid 禁	Status 止
AHOUR	2	RO	0x0	RTC 0: 中断使能 1: 中断使能	Match 中	Hour	Alarm 断	Interrupt 禁	Valid 禁	Status 止
AMIN	1	RO	0x0	RTC 0: 中断使能 1: 中断使能	Match 中	Minute	Alarm 断	Interrupt 禁	Valid 禁	Status 止
ASEC	0	RO	0x0	RTC 0: 中断使能 1: 中断使能	Match 中	Second	Alarm 断	Interrupt 禁	Valid 禁	Status 止

1. 4. 12 RTC 原始中断标志寄存器(RTC_RIF)

		Address offset:				0x2C										
		Reset value:				0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
Reserved															WKT M	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
F1HZ	RYEAR	RMON	RDATE	RWEK	RHOUR	RMIN	RSEC	AMALL	AYEAR	AMON	ADATE	AWEK	AHOUR	AMIN	ASEC	

Name	Bit	Type	Reset value	Description
Reserved	31:17	N/A	0x0	保留，保持复位值
WKT M	16	RO	0x0	RTC Wakeup Counter Match Raw Interrupt Flag Status 0: 没 有 中 断 1: 中断产生
F1HZ	15	RO	0x0	RTC 1 Hz Raw Interrupt Flag Status 0: 没 有 中 断 1: 中断产生
RYEAR	14	RO	0x0	RTC Year Roll Alarm Raw Interrupt Flag Status 0: 没 有 中 断 1: 中断产生
RMON	13	RO	0x0	RTC Month Roll Alarm Raw Interrupt Flag Status 0: 没 有 中 断 1: 中断产生
RDATE	12	RO	0x0	RTC Date Roll Alarm Raw Interrupt Flag Status 0: 没 有 中 断 1: 中断产生

RWEEK	11	RO	0x0	RTC Week 0: 没 1: 中断产生	Roll	Alarm	Raw	Interrupt	Flag	Status
RHOUR	10	RO	0x0	RTC Hour 0: 没 1: 中断产生	Roll	Alarm	Raw	Interrupt	Flag	Status
RMIN	9	RO	0x0	RTC Minute 0: 没 1: 中断产生	Roll	Alarm	Raw	Interrupt	Flag	Status
RSEC	8	RO	0x0	RTC Second 0: 没 1: 中断产生	Roll	Alarm	Raw	Interrupt	Flag	Status
AMALL	7	RO	0x0	RTC Match 0: 没 1: 中断产生	All	Alarm	Raw	Interrupt	Flag	Status
AYEAR	6	RO	0x0	RTC Match 0: 没 1: 中断产生	Year	Alarm	Raw	Interrupt	Flag	Status
AMON	5	RO	0x0	RTC Match 0: 没 1: 中断产生	Month	Alarm	Raw	Interrupt	Flag	Status
ADTAE	4	RO	0x0	RTC Match 0: 没 1: 中断产生	Date	Alarm	Raw	Interrupt	Flag	Status
AWEEK	3	RO	0x0	RTC Match 0: 没 1: 中断产生	Week	Alarm	Raw	Interrupt	Flag	Status
AHOUR	2	RO	0x0	RTC Match 0: 没 1: 中断产生	Hour	Alarm	Raw	Interrupt	Flag	Status
AMIN	1	RO	0x0	RTC Match 0: 没 1: 中断产生	Minute	Alarm	Raw	Interrupt	Flag	Status
ASEC	0	RO	0x0	RTC Match 0: 没 1: 中断产生	Second	Alarm	Raw	Interrupt	Flag	Status

1. 4. 13 RTC 屏蔽中断标志寄存器(RTC_IFM)

Address offset:		0x30													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															WKT M
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F1HZ	RYEAR	RMON	RDAT E	RWE EK	RHO UR	RMIN	RSEC	AMALL	AYEAR	AMON	ADAT E	AWE EK	AHO UR	AMI N	ASEC

Name	Bit	Type	Reset value	Description
Reserved	31:17	N/A	0x0	保留，保持复位值
WKT M	16	RO	0x0	RTC Wakeup Counter Match Interrupt Flag Masked Status 0: 没 有 中 断 1: 中断产生

F1HZ	15	RO	0x0	RTC 1 Hz 0: 没 1: 产生中断	Interrupt 有	Flag 中	Masked 中	Status 断
RYEAR	14	RO	0x0	RTC Year 0: 没 1: 中断产生	Alarm 有	Interrupt 有	Flag 中	Status 断
RMON	13	RO	0x0	RTC Month 0: 没 1: 屏蔽中断	Alarm 有	Interrupt 屏	Flag 蔽	Status 断
RDATE	12	RO	0x0	RTC Date 0: 没 1: 中断产生	Alarm 有	Interrupt 有	Flag 中	Status 断
RWEEK	11	RO	0x0	RTC Week 0: 没 1: 屏蔽中断	Alarm 有	Interrupt 屏	Flag 蔽	Status 断
RHOUR	10	RO	0x0	RTC Hour 0: 没 1: 中断产生	Alarm 有	Interrupt 有	Flag 中	Status 断
RMIN	9	RO	0x0	RTC Minute 0: 没 1: 产生中断	Alarm 有	Interrupt 有	Flag 中	Status 断
RSEC	8	RO	0x0	RTC Second 0: 没 1: 中断产生	Alarm 有	Interrupt 有	Flag 中	Status 断
AMALL	7	RO	0x0	RTC Match 0: 没 1: 中断产生	Alarm 有	Interrupt 有	Flag 中	Status 断
AYEAR	6	RO	0x0	RTC Match 0: 没 1: 中断产生	Alarm 有	Interrupt 有	Flag 中	Status 断
AMON	5	RO	0x0	RTC Match 0: 没 1: 产生中断	Alarm 有	Interrupt 有	Flag 中	Status 断
ADTAE	4	RO	0x0	RTC Match 0: 没 1: 产生中断	Alarm 有	Interrupt 有	Flag 中	Status 断
AWEEK	3	RO	0x0	RTC Match 0: 没 1: 屏蔽中断	Alarm 有	Interrupt 屏	Flag 蔽	Status 断
AHOUR	2	RO	0x0	RTC Match 0: 没 1: 中断产生	Alarm 有	Interrupt 有	Flag 中	Status 断
AMIN	1	RO	0x0	RTC Match 0: 没 1: 中断产生	Alarm 有	Interrupt 有	Flag 中	Status 断
ASEC	0	RO	0x0	RTC Match 0: 没 1: 中断产生	Alarm 有	Interrupt 有	Flag 中	Status 断

1. 4. 14 RTC 中断清除寄存器(RTC_ICR)

		Address offset:				0x34									
		Reset value:				0x0000 0000									
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														WKTM	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
F1HZ	RYEAR	RMON	RDATE	RWEEK	RHOUR	RMIN	RSEC	AMALL	AYEAR	AMON	ADAT	AWEK	AHOUR	AMIN	ASEC

Name	Bit	Type	Reset value	Description
Reserved	31:17	N/A	0x0	保留，保持复位值
WKTMM	16	W1C	0x0	RTC Wakeup Counter Match Interrupt Clear 0: 无效 1: 清除中断
F1HZ	15	W1C	0x0	RTC 1 Hz Interrupt Clear 0: 无效 1: 清除中断
RYEAR	14	W1C	0x0	RTC Year Roll Alarm Interrupt Clear 0: 无效 1: 清除中断
RMON	13	W1C	0x0	RTC Month Roll Alarm Interrupt Clear 0: 无效 1: 清除中断
RDATE	12	W1C	0x0	RTC Date Roll Alarm Interrupt Clear 0: 无效 1: 清除中断
RWEEK	11	W1C	0x0	RTC Week Roll Alarm Interrupt Clear 0: 无效 1: 清除中断
RHOUR	10	W1C	0x0	RTC Hour Roll Alarm Interrupt Clear 0: 无效 1: 清除中断
RMIN	9	W1C	0x0	RTC Minute Roll Alarm Interrupt Clear 0: 无效 1: 清除中断
RSEC	8	W1C	0x0	RTC Second Roll Alarm Interrupt Clear 0: 无效 1: 清除中断
AMALL	7	W1C	0x0	RTC Match All Alarm Interrupt Clear 0: 无效 1: 清除中断
AYEAR	6	W1C	0x0	RTC Match Year Alarm Interrupt Clear 0: 无效 1: 清除中断
AMON	5	W1C	0x0	RTC Match Month Alarm Interrupt Clear 0: 无效 1: 清除中断
ADTAE	4	W1C	0x0	RTC Match Date Alarm Interrupt Clear 0: 无效 1: 清除中断
AWEK	3	W1C	0x0	RTC Match Week Alarm Interrupt Clear 0: 无效 1: 清除中断
AHOUR	2	W1C	0x0	RTC Match Hour Alarm Interrupt Clear 0: 无效 1: 清除中断
AMIN	1	W1C	0x0	RTC Match Minute Alarm Interrupt Clear 0: 无效 1: 清除中断
ASEC	0	W1C	0x0	RTC Match Second Alarm Interrupt Clear 0: 无效 1: 清除中断

1. 4. 15 RTC 状态寄存器(RTC_STA)

		Address offset:		0x40											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														SYN DONE	EMPTY

Name	Bit	Type	Reset value	Description
Reserved	31:2	N/A	0x0	保留，保持复位值
SYNDONE	1	RO	0x0	RTC Calendar Synchronization status 0 : RTC 在同步过程中 1 : RTC 同步过程结束
EMPTY	0	RO	0x0	RTC Calendar Status 0 = RTC 日历不为空 1 = RTC 日历为空

1. 4. 16 RTC 读备份域寄存器(RTC_BKEN)

		Address offset:		0x44											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															BKEN

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x0	保留，保持复位值
BKEN	0	R/W	0x0	Read RTC Backup Register Enable 如果设置 RTC_BKCALEN，读取 rtctime&rtccal 寄存器需要更长的时间 (19 APB CLK) 0 : 从控制寄存器中读取 RTCTIME 和 RTCCAL 1 : 从备份寄存器中读取 RTCTIME 和 RTCCAL。

第18章 集成内部总线接口（I2C）

1.1 I2C 简介

I2C (芯片间) 总线接口连接微控制器和串行 I2C 总线。它提供多主机功能，控制所有 I2C 总线特定的时序、协议、仲裁和定时。它支持标准模式，快速模式和超快速模式。同时与 SMBus（系统管理总线）和 PMBus（电源管理总线）保持兼容。可以使用 DMA 以减轻 CPU 的负担。

1.2 I2C 主要特性

- ◆ 从机模式和主机模式
- ◆ 多主机功能
- ◆ 标准模式(高达 100 kHz)
- ◆ 快速模式(高达 400 kHz)
- ◆ 超快速模式(高达 1 MHz)
- ◆ 7 位和 10 位地址模式
- ◆ 多个 7 位从地址(2 个地址，其中一个可屏蔽)
- ◆ 所有 7 位地址应答模式
- ◆ 广播呼叫
- ◆ 可编程建立和保持时间
- ◆ 可选的时钟延长
- ◆ 可编程数字噪声滤波器
- ◆ 深度为 8 的发送/接收 FIFOs
- ◆ DMA 功能
- ◆ SMBus 标准
- ◆ 硬件 PEC(包错误检查)的生成与验证，带 ACK 控制
- ◆ 命令与数据应答控制
- ◆ 支持地址解析协议（ARP）
- ◆ 主机和设备支持
- ◆ SMBus 报警
- ◆ 超时和空闲状态检测
- ◆ 与 PMBus 版本 1.1 标准兼容

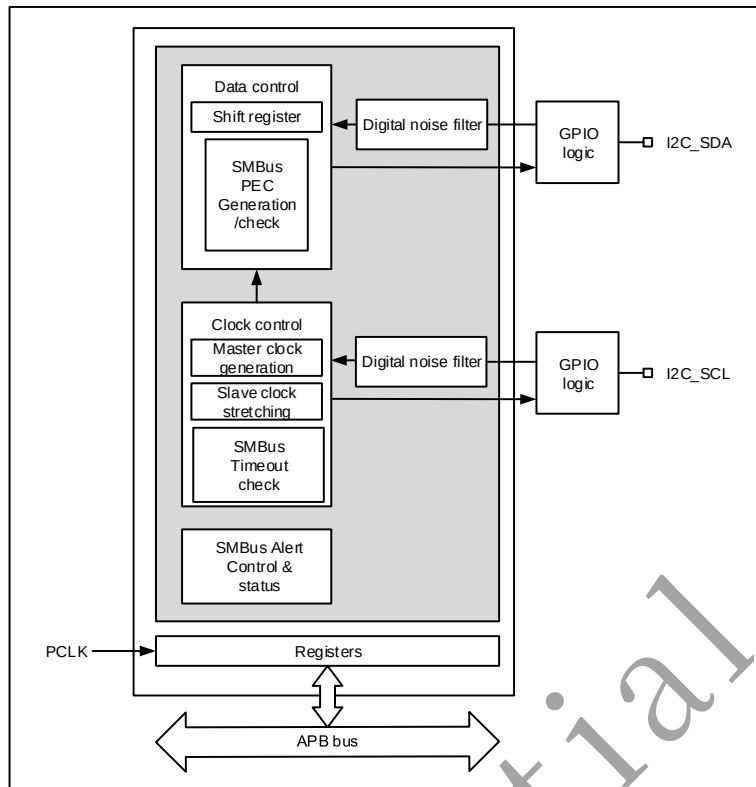


图 18-1 I2C 结构框图

1.3 I2C 功能描述

除了接收和发送数据，这个接口从串行转换成并行格式，以及其逆过程。通过软件启用或禁用中断。该接口通过数据引脚（SDA）和时钟引脚（SCL）连接到 I2C 总线。它可以连接标准（100 kHz），快速模式（400 kHz）或快速模式（1 MHz）的 I2C 总线。

此接口也可以通过数据引脚（SDA）和时钟引脚（SCL）连接到的 SMBus。

如果有 SMBus 功能支持：附加的 SMBus ALERT 引脚（SMBA）也是可用的。

1.3.1 I2C 总线协议

I2C 是一种双线双向串行总线，它提供了一种简单而有效的设备间数据交换方法。I2C 标准是一个真正的多主总线，包括冲突检测和仲裁，如果两个或多个主总线试图同时控制总线，则可以防止数据损坏。串行数据(SDA)经过串行时钟（SCL）同步，数据在主机和从机之间逐字节地传输。每个数据字节 8 位长。

1.3.1.1 I2C 启动和停止条件

I2C 规范将启动条件定义为 SDA 线从高状态到低状态的转换，而 SCL 线则是高状态。启动条件总是由主机生成，表示总线从空闲状态转换到活动状态。每个数据位都有一个 SCL 时钟脉冲，首先传输 MSB。每个传输的字节后面都有一个确认位。当 SCL 高时，SDA 行上的转换被解释为命令(启动或停止)。在 SCL 的高周期内对每个比特进行采样;因此，SDA 线只能在 SCL 的低周期内改变，在 SCL 的高周期内必须保持稳定。

当总线空闲时，SCL 和 SDA 信号都通过总线外部上拉电阻被拉高。当主机要在总线上启动传输时，主机发出一个 START 条件。启动条件被定义为当 SCL 为 1 时，SDA 信号从高到低的跃迁。当主机想要终止传输时，主机发出一个 STOP 条件。STOP 条件被定义为当 SCL 为 1 时，SDA 线从低到高的跃迁。下图显示了启动和停止条件的时间。当数据在总线上传输时，当 SCL 为 1 时，SDA 线必须是稳定的。

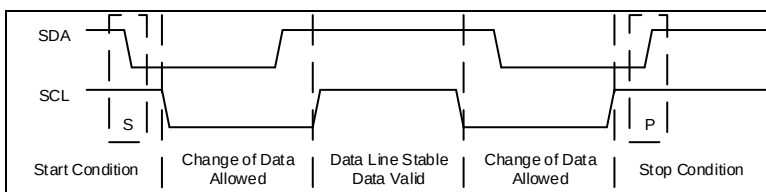


图 18-2 START 和 STOP 条件

1. 3. 1. 2I2C 应答响应

数据传输带应答是必须的。应答时钟脉冲由主机产生。发送机释放 SDA 线(高)在应答时钟脉冲期间。

接收机必须在应答时钟脉冲期间拉低 SDA 线，以便在此时钟脉冲的高周期内保持稳定的低电平。当然，建立和保持时间也必须考虑在内。

通常，被寻址的接收机必须在接收到每个字节后生成一个应答，除非消息以 CBUS 地址开始。

当从机没有应答从机地址时(例如，它无法接收或传输，因为它正在执行一些实时功能)，数据行必须由从机保持高电平。主机可以生成终止传输的 STOP 条件，或者生成启动新传输的重复 START 条件。

如果从机接收方应答了从机地址，但在传输一段时间后无法接收更多的数据字节，则主机必须再次中止传输。由从机在随后第一个字节后生成 NACK 进行提示。从机保持数据行高电平，由主机生成一个停止或重复启动条件。

如果主接收机参与传输，它必须在最后一个字节后的时钟脉冲周期产生 NACK 来向从机发送机表明数据结束信号。从机发送机必须释放数据线，以允许主机产生一个停止或重复启动的条件

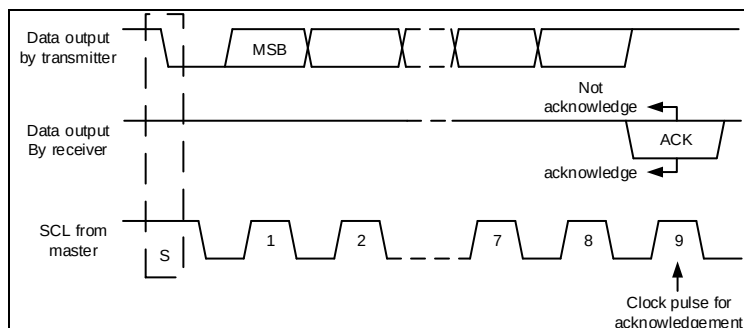


图 18-3 I2C 总线应答

1. 3. 1. 3I2C 寻址协议

两种寻址方式:7 位寻址方式和 10 位寻址方式。7 位寻址方式,第一字节的前七位(Bit7:1)设置从机地址, LSB (Bit0) 是读/写 (R/W)。当 Bit8 被设置为 0 时, 主机对从机进行写操作。当 Bit8(R/W)设置为 1 时,主机对从机进行读操作。数据传输必须 MSB 优先。10 位寻址方式, 传输两个字节来设置 10 位地址。第一个字节的传输包含以下定义。前五位(Bit7:3)通知从机,这是 10 位传输方式,接下来的两位(Bit2:1)是设置从机地址 Bit9:8,LSB (Bit0) 是读/写位。传输的第二个字节设置从机地址的 7:0 位。

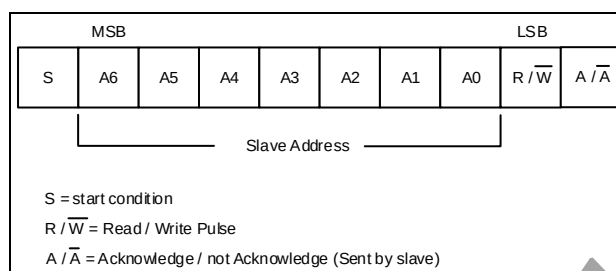


图 18-4 7 位地址格式

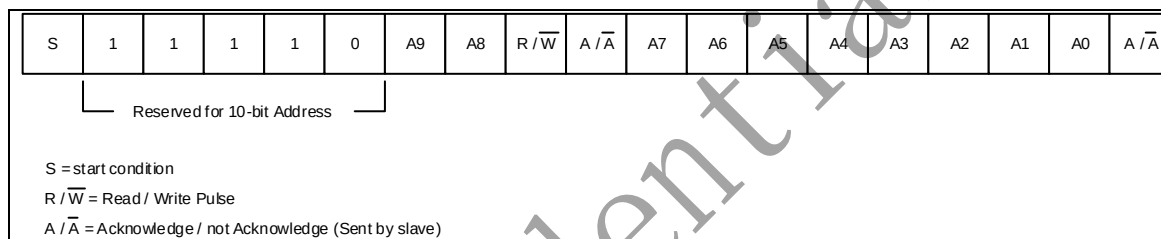


图 18-5 10 位地址格式

Slave Address	R/W Bit	Description
0000 000	0	General Call Address.
0000 000	1	START byte
0000 001	X	CBUS address
1111 0XX	X	10-bit slave addressing

表 18-1 第一个字节定义

1. 3. 1. 4I2C 发送/接收协议

所有数据都以字节格式传输, 每次数据传输不受传输字节数限制。主机发送地址和读/写位或传送一个字节数据到从机,从接收机必须响应一个应答信号。当从接收器没有应答响应时, 主机通过发出 STOP 条件终止传输。从机应保持 SDA 线高, 以便主机可以终止传输。

如果主发射机正在传输数据, 然后从接收机在接收到每个字节的数据后都应响应主发射机。传输格式如下:

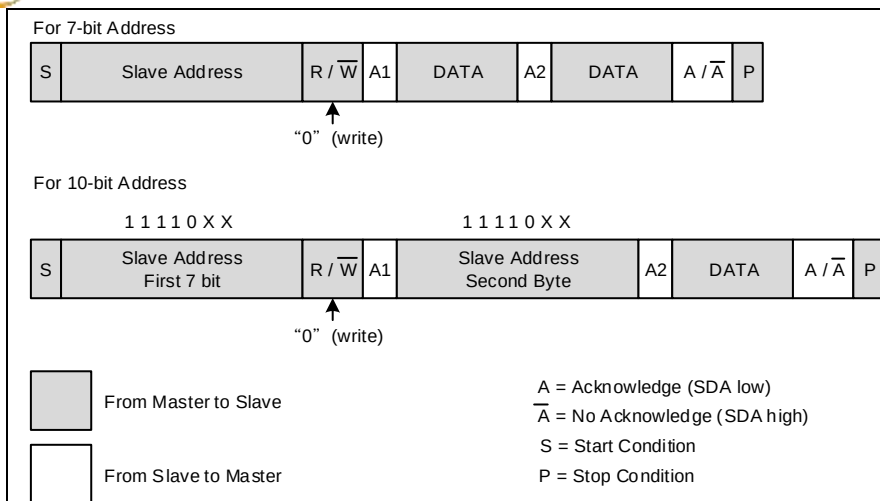


图 18-6 主模式 – 发送协议

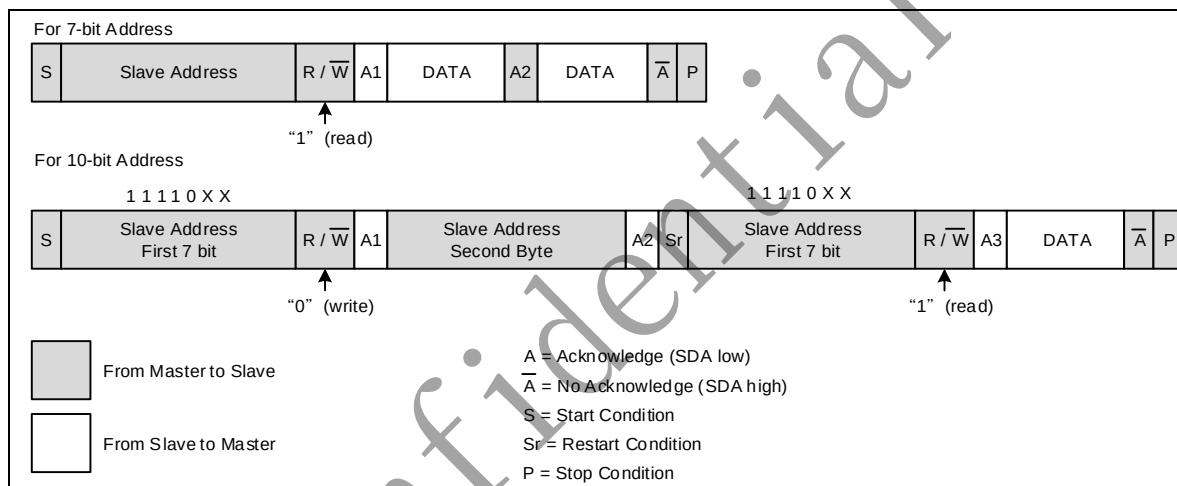


图 18-7 主模式—接收协议

1. 3. 1. 5I2C 启动字节传输协议

启动字节传输协议是为没有专用 I2C 硬件模块的系统设置的。当 I2C 做为从设备时，它总是以所支持的最高速度对 I2C 总线进行采样，因此它从不需要传输启动字节。但是，当 I2C 是一个主设备时，它支持在每次传输开始之前生成并传输启动字节，以满足从设备的需要。启动字节协议由 7 个 0 和一个 1 组成，如下所示。这允许正在轮询总线的处理器对地址阶段进行抽样，直到检测到 0 为止。一旦微控制器检测到 0，它就从欠采样率切换到正确的主采样率。

启动字节的步骤如下：

1. 主设备生成一个启动条件。
2. 主设备传输启动字节(0000 0001)。
3. 主设备发送 ACK 时钟脉冲。
4. 没有从设备设置 ACK 信号为 0。

5. 主设备生成一个重复启动(Sr)条件。

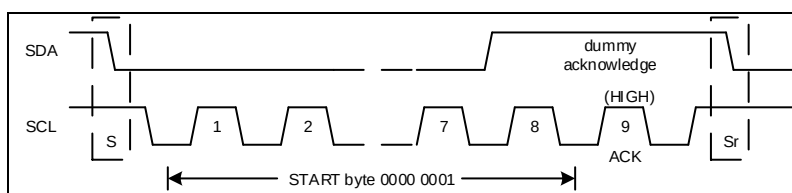


图 18-8 启动字节过程

1.3.2 I2C 时钟要求

I2C 内核由 I2C_CLK 提供时钟。

I2C_CLK 的时钟周期 t_{I2C_CLK} 必须遵守以下条件:

$$t_{I2C\_CLK} < (t_{LOW} - t_{filters}) / 4 \text{ and } t_{I2C\_CLK} < t_{HIGH}$$

其中:

t_{LOW} : SCL 低电平时间

t_{HIGH} : SCL 高电平时间

$t_{filters}$: 当使能时, 数字滤波器带来的延迟之和。数字滤波器延迟是 $DNF \times t_{I2C\_CLK}$ 。

PCLK 时钟周期 t_{PCLK} 必须满足以下条件: $t_{PCLK} < 4/3 t_{SCL}$

其中 t_{SCL} : SCL 周期

1.3.3 I2C 数据传输

在 SDA 行的每个字节必须是 8 位字长。每次传输可以传输的字节数是没限制的。每个字节后面必须跟一个确认位。首先用最高有效位(MSB)传输数据。如果一个从设备不能接收或传输另一个完整字节的数据, 直到它执行一些其他功能, 例如服务一个内部中断, 它可以保持时钟线 SCL 低, 迫使主设备进入等待状态。当从设备准备好接收另一个字节的数据并释放时钟线 SCL 时, 数据传输将继续。

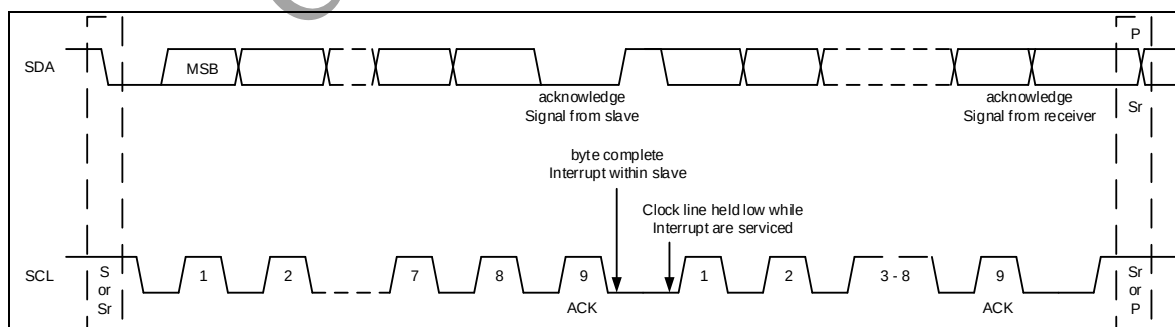


图 18-9 在 I2C 总线上进行数据传输

接收

SDA 输入会填充移位寄存器。在第 8 个 SCL 脉冲（当收到完整的数据字节）之后，如果 I2Cx_RXDR 寄存器是空的（RXNE = 0）移位寄存器的内容会被复制到 I2Cx_RXDR 寄存器。如果 RXNE = 1，也就是说，尚未读取之前收到的数据字节，这时 SCL 线被拉低直到 I2Cx_RXDR 被读取。这个拉伸被插入到第 8 和第 9 个 SCL 脉冲之间（应答脉冲之前）。

发送

如果 I2Cx_TXDR 寄存器不为空（的 TXE = 0），其内容将在第 9 个 SCL 脉冲（应答脉冲）之后被复制到移位寄存器。然后移位寄存器的内容被移到 SDA 线上。如果 TXE = 1，也就是说，I2Cx_TXDR 中没有数据被写入，SCL 线会被拉长为低直到 I2Cx_TXDR 有新数据被写入。在第 9 个 SCL 脉冲之后完成拉伸。

硬件传输管理

I2C 有一个内嵌的字节计数器，可以管理发送的字节数，以便在各种模式中关闭通讯：

- ◆ 主模式下生成 NACK、STOP 和 ReSTART
- ◆ 从机接收模式下的 ACK 控制
- ◆ SMBus 的功能支持时的 PEC 的生成 / 检查

字节计数器总是在主模式下使用。默认情况下，它在从模式下被禁用，但它可以通过软件设置 I2Cx_CR2 寄存器的 SBC 位（从字节控制）来启用。

要传输的字节数编程在 I2Cx_CR2 寄存器的 NBYTES[7:0] 位域。如果要传输的字节数（nBytes 个）大于 255，或一个接收器要控制收到的数据字节的应答值，就必须设置 I2Cx_CR2 寄存器的 RELOAD 位来选择重载模式。在这种模式下，当 NBYTES 中编程的字节个数被发送完毕后，TCR 标志被置 1，如果 TCIE 为 1，则会产生一个中断。SCL 在 TCR 标志为 1 期间被拉伸。向 NBYTES 写入是一个非零的值时，TCR 标志由软件清除。

当 NBYTES 被重载为上次的字节数时，RELOAD 位必须被清零。

当主模式下 RELOAD = 0，计数器可以按下列 2 种模式工作：

- ◆ **自动结束模式：**自动结束模式（I2Cx_CR2 寄存器的 AUTOEND = 1）。在这种模式下，一旦发送字节数达到了 NBYTES[7:0] 位域中设置的字节数，主机自动发送一个停止条件。
- ◆ **软件结束模式：**软件结束模式（I2Cx_CR2 寄存器的 AUTOEND = 0）。在这种模式下，发送字节数达到了 NBYTES[7:0] 位域中设置的字节数后需要软件的干预，这时 TC 标志会被置 1，如果 TCIE 位为 1，还会产生中断。在 TC 标志为 1 期间，SCL 信号被拉伸。在 I2Cx_CR2 寄存器的 START 或 STOP 位被置 1 时，TC 标志由软件清除。主机要发送一个 RESTART 条件时，必须使用此模式。

1.3.4 I2C 从机模式

I2C 从机初始化

要工作在从机模式下，您必须启用至少一个从机地址。两个寄存器 I2Cx_OAR1 和 I2Cx_OAR2 都可以用来写入从机的本机地址 OA1 和 OA2。

- ◆ 通过设置 I2Cx_OAR1 寄存器的 OA1MODE 位，可将 OA1 配置在 7 位模式（默认）

或 10 位地址模式。通过设置 I2Cx_OAR1 寄存器的 OA1EN 位来启用 OA1。

- ◆ 如果需要额外的从机地址，你可以配置第二个从机地址 OA2。配置 I2Cx_OAR2 寄存器的 OA2MSK [2:0] 位域，可以最多屏蔽 OA2 的低 7 位。因此从 1 到 6 为 OA2MSK 配置，只有 OA2 [7:2]，OA2 [7:3]，OA2 [7:4]，OA2 [7:5]，OA2 [7:6] 或 OA2 [7] 参与与接收到的地址的比较操作。只要 OA2MSK 不等于 0，被 OA2 地址比较排除的 I2C 地址（0000 XXX 和 1111 xxx）不会被应答。如果 OA2MSK = 7，收到的所有 7 位地址（保留地址除外）都会被应答。OA2 始终是一个 7 位地址，不可以 10 位。如果他们启用特定的使能位，这些保留地址也可以应答，就是如果他们被写在 I2Cx_OAR1 或 I2Cx_OAR2 寄存器中并且 OA2MSK = 0。通过设置 I2Cx_OAR2 寄存器的 OA2EN 位来启用 OA2。
- ◆ I2Cx_CR1 寄存器的 GCEN 位被置 1 时，呼叫地址被启用。

当 I2C 由启用了的地址选中时，ADDR 中断状态标志被置 1，如果 ADDR1E 位为 1，就会产生一个中断。

默认情况下，从机使用它的时钟延长的功能，这意味着，它可根据需要在 SCL 信号拉低，以便执行软件动作。如果主机不支持时钟延长，I2C 的 I2Cx_CR1 寄存器的 NOSTRETCH 必须配置为 1。

收到地址中断后，如果启用了多个地址，你必须读 I2Cx_ISR 寄存器中的 ADDCODE[6:0] 位以检查是哪一个地址匹配上了。还要检查 DIR 标志，以了解数据传输的方向。

从时钟延长(NOSTRETCH = 0)

在默认模式下，I2C 从机于下列情况下拉低 SCL 时钟：

- ◆ 当地址标志被置位：接收到的地址和启用了的从机地址之一匹配上。这种拖延在 ADDR 标志被软件清零后被释放。清零该位的办法是将 ADDR1CF 位置 1。
- ◆ 在发送时，如果以前的数据传输完毕后，并没有新的数据被写到 I2Cx_TXDR 寄存器，或者如果在 ADDR 标志被清除（TXE = 1）后还没有写一个字节。只要数据被写入 I2Cx_TXDR 寄存器，就会释放这个拖延。
- ◆ 在接收时如果 I2Cx_RXDR 寄存器的内容还没有被读走，又有一个新的数据被收进来。这时延长在读取 I2Cx_RXDR 时被释放。
- ◆ 在从机字节控制模式，重载模式（SBC = 1 和 RELOAD = 1）时，如果 TCR = 1，意味着最后一个数据字节已发送完毕。在向 NBYTES[7:0] 位域写入一个非零值会清除 TCR 标志，这时延长也会释放。
- ◆ 在检测到 SCL 下降边缘后，I2C 延伸 SCL 低电平。
- ◆ $[(SDADEL+SCLDEL+1) \times (PRESC+1) + 1] \times t_{I2CCLK}$.

从机不带时钟延长(NOSTRETCH = 1)

当 I2Cx_CR1 寄存器的 NOSTRETCH = 1 时，I2C 从机不会延长 SCL 信号。

- ◆ 在 ADDR 标志置位的时候 SCL 时钟不会延长。
- ◆ 发送中，数据必须在对于发送的第一个 SCL 脉冲之前写入到 I2Cx_TXDR 寄存器。如果没有，会发生欠载，I2Cx_ISR 寄存器中的 UDR 标志被置位，如果 I2Cx_CR1

寄存器的 **ERRIE** 位为 **1**，会产生一个中断。如果首个数据发送已经开始而 **STOPF** 位还是 **1**（未被清掉）时，**OVR** 标志也会被设置。因此，如果你在写下次发送的首个发送数据之后才清除前一次传输的 **STOPF** 标志，必须先确认 **OVR** 状态，甚至于首个数据已经发送。

- ◆ 在接收时，必须在下一个字节的第 9 个 **SCL** 脉冲（**ACK** 脉冲）发生前，从 **I2Cx_RXDR** 寄存器将数据读走。如果没有，会发生溢出，**I2Cx_ISR** 寄存器中的 **OVR** 标志被置位，如果 **I2Cx_CR1** 寄存器的 **ERRIE** 位为 **1**，会产生一个中断。

从机字节控制模式

为了允许从机接收模式的字节 **ACK** 控制，从机字节控制模式必须通过设置 **I2Cx_CR1** 寄存器的 **SBC** 位来启用。这是与 **SMBus** 标准兼容所必需的。

必须选择重载模式，以便允许在从机接收模式下的字节 **ACK** 控制（**RELOAD** = **1**）。要获得每个字节的控制，必须在 **ADDR** 中断服务程序中将 **NBYTES** 初始化为 **0x1**，并在每接收一个字节后重新加载为 **0x1**。每当收到一个字节，**TCR** 位置 **1**，在第 8 和第 9 个 **SCL** 脉冲间，延长 **SCL** 信号为低。可以从 **I2Cx_RXDR** 寄存器读数据，然后决定要不要通过配置 **I2Cx_CR2** 寄存器的 **ACK** 位来发 **ACK**。对 **NBYTES** 写入一个非零值会释放 **SCL** 延长： **ACK** 或 **NAK** 被发送，然后可以接收下一字节。

可以写入一个比 **0x1** 更大的值到 **NBYTES**，在这种情况下，会连续接收 **NBYTES** 个数据。

注： 1： **SBC** 位必须在 **I2C** 被禁用时配置，或当从机没有被寻址到时，或者当 **ADDR**= **1** 时。

2： **RELOAD** 位的值可以在 **ADDR**=**1** 或者当 **TCR**=**1** 的时候改变。

警告： 从机字节控制模式不兼容 **NOSTRETCH** 模式。不允许在 **NOSTRETCH**=**1** 的时候置位 **SBC**。

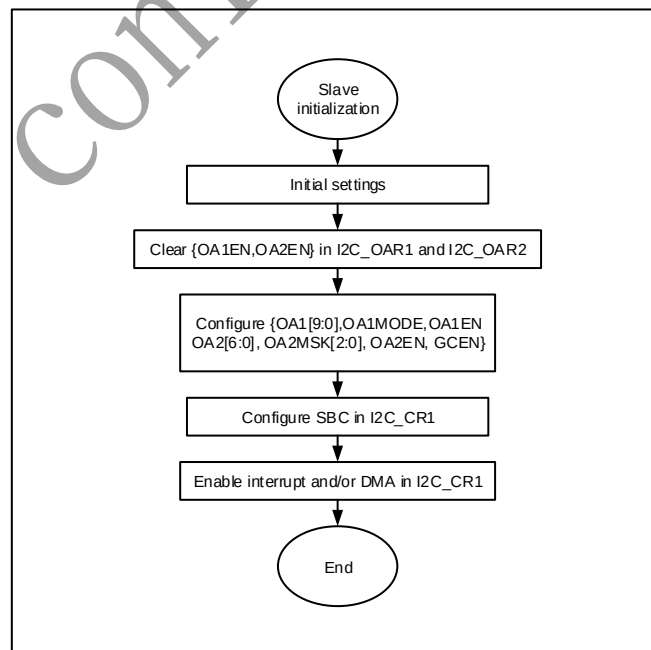


图 18-10 从模式初始化流程图

1. 3. 5 I2C 主机模式

I2C 主机初始化

在启动外设之前，必须设置 I2Cx_TIMINGR 寄存器的 SCLH 和 SCLL 的位来配置 I2C 主时钟。

这会实施一个时钟同步机制，以支持多主机环境和从机时钟延长。

为了让时钟同步：

- ◆ 从 SCL 低电平的内部检测开始用 SCLL 计数器来计数时钟低电平个数。
- ◆ 从 SCL 高电平的内部检测开始用 SCLH 计数器来计数时钟高电平个数。

I2C 依靠 SCL 下降沿的一个 tSYNC1 延迟和 SCL 输入噪声滤波器（模拟+ 数字）来检测自己的 SCL 低电平，并将 SCL 同步到 I2Cx_CLK 时钟。一旦 SCLL 计数器达到了 I2Cx_TIMINGr 寄存器的 SCLL[7:0] 位域中的编程值，I2C 释放 SCL 到高电平。

I2C 依靠 SCL 上升沿的一个 tSYNC2 延迟和 SCL 输入噪声滤波器（模拟+ 数字）来检测自己的 SCL 高电平，并将 SCL 同步到 I2Cx_CLK 时钟。一旦 SCLH 计数器达到了 I2Cx_TIMINGR 寄存器的 SCLH[7:0] 位域中的编程值，I2C 将 SCL 拉低到低电平。

因此，主机时钟周期是：

$$t_{SCL} = t_{SYNC1} + t_{SYNC2} + \{[(SCLH+1) + (SCLL+1)] \times (PRESC+1) \times t_{I2C\_CLK}\}$$

tSYNC1 时间取决于以下参数：

- ◆ SCL 下降斜率
- ◆ 启用时，数字滤波器所带来的输入延迟: DNF x t_{I2C_CLK}
- ◆ 由于 SCL 的同步 I2C_CLK 时钟（2 至 3 I2C_CLK 周期）造成的延迟

tSYNC2 时间取决于以下参数：

- ◆ SCL 的上升斜率
- ◆ 启用时，数字滤波器所带来的输入延迟: DNF x t_{I2C_CLK}
- ◆ 由于 SCL 的同步 I2C_CLK 时钟（2 至 3 I2C_CLK 周期）造成的延迟

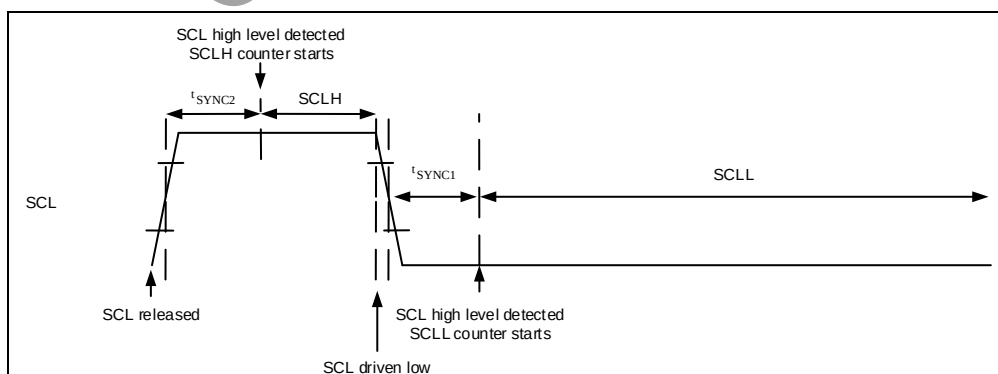


图 18-11 主时钟生成

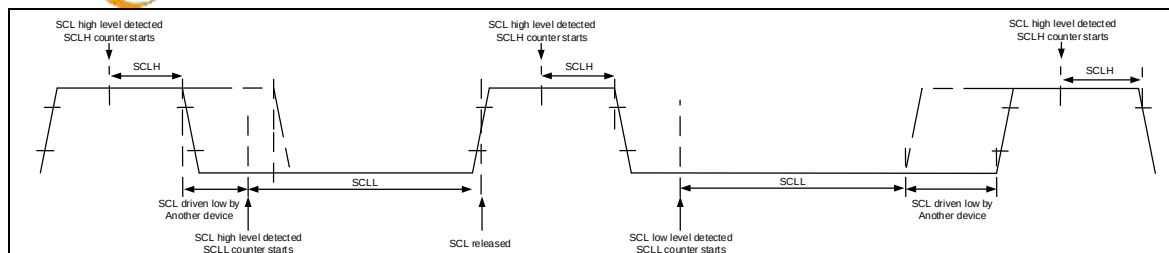


图 18-12 SCL 主时钟同步

主机通信初始化(地址阶段)

为了启动通讯，必须在 I2Cx_CR2 寄存器中设置从机地址的下列参数：

- ◆ 地址模式（7 位或 10 位）：ADD10
- ◆ 要发送的从机地址：SADD [9:0]
- ◆ 传输方向：RD_WRN
- ◆ 在 10 位地址的情况下读取：HEAD10R 位。在必须发送完整的地址顺序时，HEAD10R 必须被先置位，以示在方向改变时仅需要帧头的区别。
- ◆ 要传输的字节数：NBYTES[7:0]. 如果字节数大于等于 255 个字节，NBYTES[7:0] 最初必须使用 0xff 填充。

随后必须设置 I2Cx_CR2 寄存器的 START 位。START 位一旦被置 1，就不再允许更改上述所有位。

只要检测到总线是空闲（BUSY = 0）的并插入一个延时后，主机自动发送 START 条件，随后就是发送从机地址。

主在仲裁丢失的情况下，主机自动切换回从模式，如果从机地址被选中，还将会自动发送 ACK 应答。

注：在从机地址在总线上面发出后，无论收到 ACK 值的内容是什么，START 位都由硬件复位。如果发生了仲裁丢失。START 位也由硬件清零。在 START 位为 1 期间，如果 I2C 作为一个从机（ADDR=1）被选中，I2C 会切换到从机模式，并且在 ADDRCONF 位被置 1 时 START 位被清零。

注：相同的程序适用于重复开始条件。在这种情况下，BUSY = 1。

主机接收器寻址一个 10 位地址的从机的初始化

- ◆ 如果从机地址是 10 位格式，你可以选择清除 I2Cx_CR2 寄存器的 HEAD10R 位以发送一个完整的读序列。在这种情况下，主机在 START 位被置 1 后，自动发送下面的完整序列：(Re)Start + 从机地址 10 位头的写操作 + 从机地址的第二字节 + RStart + 从机地址 10 位头的读操作
- ◆ 如果主机要寻址到一个 10 位地址的从机，将数据发送给这个从机，然后从同一个从机读取数据，主机发送流程必须首先完成。然后以 HEAD10R = 1 的状态重复开始条件和 10 位从机地址。在这种情况下，主机发送序列如下：ReStart+ 从机地址 10 位头的读操作

1.3.6 I2C_TIMINGR 寄存器配置举例

下表提供了如何编写 I2C_TIMINGR 以获得符合 I2C 规范的计时的示例。

Parameter	Standard-mode (Sm)		Fast-mode (Fm)	Fast-mode Plus (Fm+)
	10kHz	100kHz	400kHz	500kHz
PRESC	3	3	1	0
SCLL	0xC7	0x13	0x9	0x4
t_{SCLL}	200x250 ns = 50 μ s	20x250 ns = 5.0 μ s	10x125 ns = 1250 ns	6x62.5 ns = 312.5 ns
SCLH	0xC3	0xF	0x3	0x2
t_{SCLH}	196x250 ns = 49 μ s	16x250 ns = 4.0 μ s	4x125ns = 500ns	3x62.5ns = 187.5ns
$t_{SCL}^{(1)}$	$\sim 100 \mu$ S ⁽²⁾	$\sim 10 \mu$ S ⁽²⁾	~ 2500 n S ⁽³⁾	~ 1000 n S ⁽⁴⁾
SDADEL	0x2	0x2	0x2	0x0
t_{SDADEL}	2x250 ns = 500 ns	2x250 ns = 500 ns	2x125 ns = 250 ns	0 ns
SCLDEL	0x4	0x4	0x3	0x2
t_{SCLDEL}	5x250 ns = 1250 ns	5x250 ns = 1250 ns	4x125 ns = 500 ns	3x62.5 ns = 187.5 ns

表 18-2 $f_{I2CCLK} = 16\text{MHz}$ 的时序设置例子

1. 由于 SCL 内部检测带来的延迟, SCL 周期 t_{SCL} 要大于 $t_{SCLL} + t_{SCLH}$ 。提供的 t_{SCL} 值只是举例。
2. $t_{SYNC1} + t_{SYNC2}$ 最小值是 $4xt_{I2CCLK} = 250$ ns. 基于例子 $t_{SYNC1} + t_{SYNC2} = 1000$ ns
3. $t_{SYNC1} + t_{SYNC2}$ 最小值是 $4xt_{I2CCLK} = 250$ ns. 基于例子 $t_{SYNC1} + t_{SYNC2} = 750$ ns
4. $t_{SYNC1} + t_{SYNC2}$ 最小值是 $4xt_{I2CCLK} = 250$ ns. 基于例子 $t_{SYNC1} + t_{SYNC2} = 500$ ns

1.3.7 SMBus 特定功能简介

系统管理总线 (SMBus) 是一个种两线接口, 通过它可以与各种设备和系统的其余部分互相通讯。它基于 I2C 操作原理。SMBus 提供了一种针对系统和电源管理相关任务的控制总线。

该外设兼容的 SMBus 规范 2.0 版(<http://smbus.org>).

系统管理总线规范是指三种类型的设备。

- ◆ 从机是指接收或相应命令的设备。
- ◆ 主机是下达命令, 产生时钟和终止传输的设备。
- ◆ HOST 是一个特殊的主机, 它向系统 CPU 提供主接口。HOST 必须具备主机和从机的双重功能, 并且必须支持 SMBus HOST 通知协议。一个系统中只可以有一个 HOST。

本外设可以配置为主机或从机, 当然也可以作为 HOST。

SMBus 是基于 I2C 规范修订版 2.1。

总线协议

对于任何给定的设备有 11 个可能的命令协议。一个设备可以使用任何或全部的 11 个协议进行通信。协议是快速命令，发送字节，接收字节，写字节，写字，读字节，读字，过程调用，块读，块写和块写块读过程调用。这些协议是由用户软件实现的。对于这些协议的更多细节，请参考 SMBus 规范版本 2.0 (<http://smbus.org>)。

地址解析协议 (ARP)

SMBus 从机地址冲突的问题可以通过给每个从设备动态标定一个新的独特的地址的方式解决。为了分配地址，需要一种区分每个设备的机制，每个设备必须拥有一个唯一的设备标识符。这个 128 位的数字是由软件实现的。

本外设支持地址解析协议 (ARP)。将 I2Cx_CR1 寄存器的 SMBDEN 位置 1，会启用 SMBus 设备的默认地址 (0b1100 001)。ARP 命令由用户软件实现。

ARP 支持的仲裁动作也是在从机模式下完成。

SMBus 地址解析协议的更多细节，请参考 SMBus 规范版本 2.0 (<http://smbus.org>)。

命令的接收和数据应答的控制

一个 SMBus 的接收器必须能够对每个收到的命令或数据回应 NACK。为了允许从机模式下的 ACK 控制，必须将 I2Cx_CR1 寄存器的 SBC 位置 1，以启用从机字节控制模式。

HOST 通知协议

设置 I2Cx_CR1 寄存器的 SMBHEN 位，使得本外设支持 HOST 通知协议。在这种情况下，HOST 会应答 SMBus 主机地址 (0b0001000)。

使用此协议时，本设备会作为主机，而 HOST 则会作为一个从机

SMBus 报警

本外设可选 SMBus 提醒信号支持。一个仅作为从机的设备在想要发起通讯的时候，可以通过 SMBALERT 引脚通知 HOST。HOST 会处理这个中断，并且随即通过提醒响应地址 (0b0001100) 来访问全部的 SMBALERT 设备。只有将 SMBALERT 引脚拉低的设备会回应提醒响应地址。

当被配置为从机设备 (SMBHEN = 0) 时，将 I2Cx_CR1 寄存器的 ALERTEN 位置 1，会导致 SMBA 引脚被拉低。提醒响应地址则会同时启用。

当被配置为 HOST 设备 (SMBHEN = 1) 时，只要发现 ALERTEN=1，并且在 SMBA 引脚上检测到一个下降沿，I2Cx_ISR 寄存器中的提醒标志就会被置位。如果 I2Cx_CR1 寄存器中的 ERRIE 位为 1，则会产生中断。当 ALERTEN = 0 时，即使是 SMBA 外部引脚为低，提醒线也会被认为是高。

如果不需要使用 SMBus 提醒引脚，只要是果 ALERTEN = 0，SMBA 脚就可以被用来作为一个标准的 GPIO。

包错误检查

SMBus 规范中介绍到的包错误检查机制可以提高通信可靠性。包错误检查是通过在每个消息后面附带一个包错误检查码来实现的。PEC 码通过对全部的消息字节（包括地址和读写位）使用多项式 $C(x) = x^8 + x^2 + x + 1$ （CRC-8）计算得来。

本外设内嵌硬件 PEC 计算单元，在接收数据的时候如果发现 PEC 结果不匹配，会允许自动发送一个 NACK。

超时

本外设内嵌一个硬件定时器，以便和 SMBus 规范 V2.0 中定义的 3 个超时相符。

Symbol	Parameter	Limits		Unit
		Min	Max	
t_{TIMEOUT}	Detect clock low timeout	25	35	ms
$t_{\text{LOW:SEXT}}^{(1)}$	Cumulative clock low extend time (slave device)	-	25	ms
$t_{\text{LOW:MEXT}}^{(2)}$	Cumulative clock low extend time (master device)	-	10	ms

表 18-3 SMBus 超时说明

1. $t_{\text{LOW:SEXT}}$ 是一个给定的从机设备从 START 到 STOP 之间可以延长的时钟周期的累计。另一个从机设备或主机也可能对时钟进行占用从而导致总的时钟低的占用时间大于 $t_{\text{LOW:SEXT}}$ 。因此，这个参数的测量条件是从机作为一个全速的主机的唯一通讯目标。
2. $t_{\text{LOW:MEXT}}$ 是一个主机设备按照 START 到 ACK，ACK 到 ACK 或者 ACK 到 STOP 的方式发送一个字节所允许的时钟周期的累计。另一个从机设备或主机也可能对时钟进行占用从而导致总的时钟低的占用时间大于 $t_{\text{LOW:MEXT}}$ 。因此，这个参数的测量条件是有只有一个全速的从机作为唯一的通讯目标。

总线空闲检测

如果主机发现时钟和数据信号保持为高的时间 t_{IDLE} 大于 $t_{\text{HIGH,MAX}}$ ，就可以认为目前总线处于空闲状态。

这个时序参数覆盖了那种主机被动态的加入到总线上，不一定检测到了 SMBCLK 或者 SMBDAT 上的状态转换的条件。在这种情况下，主机必须等待足够长的时间，以确保传输是否是正在进行中。本外设支持硬件总线空闲检测。

1.3.8 SMBus 初始化

为了正确执行 SMBus 通讯，在 I2C 初始化的基础上还有一些其它的初始化动作要做：

命令的接收和数据应答控制 (从机模式)

一个 SMBus 的接收器必须能够对每个收到的命令或数据回应 NACK。为了允许从机模式下的 ACK 控制，必须将 I2Cx_CR1 寄存器的 SBC 位置 1，以启用从机字节控制模式。

特定地址 (从机模式)

如有必要，需要启用特定的 SMBus 地址。

- ◆ 将 I2Cx_CR1 寄存器的 SMBDEN 位置 1，会启用 SMBus 设备的默认地址 (0b1100001)。
- ◆ 将 I2Cx_CR1 寄存器的 SMBHEN 位置 1 会启用 SMBus 主机地址 (0b0001 000)。
- ◆ I2Cx_CR1 寄存器的 ALERTEN 位被置 1 时，会启用通知响应地址。

包错误检查

将 I2Cx_CR1 寄存器的 PECEN 位置 1，会启用的 PEC 计算这时的 PEC 传送由硬件计数器位 (I2Cx_CR2 寄存器中的 nBytes 个[7:0]) 来配合管理。PECEN 必须在使能 I2C 之前设置好。

既然 PEC 传输是有硬件字节计数器管理的，所以在 SMBus 工作于从机模式时，SBC 位必须先置 1。PEC 数据会在传送了 NBYTES-1 个数据被传送完毕之后发送，当然这时 PECBYTE 位要求是 1，并且 RELOAD 位要求是 0。如果 RELOAD 位是 1，那 PECBYTE 位就没有作用了。

警告：当 I2C 启用时不允许更改 PECEN 的配置。

超时检测

启用超时检测功能，须将 I2Cx_TIMEOCTR 寄存器的 TIMOUTEN 位和 TEXTEN 位置 1。定时器须用某种方式编程，以实现在 SMBus 规范 2.0 给出的最大时间之前检测到超时。

◆ t_{TIMEOUT} 检查

要启用 t_{TIMEOUT} 检查，先要将要检查的 t_{TIMEOUT} 参数对应的定时器重载值写入到 12 位的 TIMEOUTA[11:0]。要检查 SCL 低电平超时，还需要保证 TIDLE 位为 0。在 I2Cx_TIMEOCTR 寄存器的 TIMOUTEN 被置 1 使能了定时器后。如果 SCL 被拉低的时间大于 $(\text{TIMEOUTA}+1) \times 2048 \times t_{\text{I2CCLK}}$ ，I2Cx_ISR 寄存器中的 TIMEOUT 标志会被置 1。

警告：在 TIMOUTEN 位为 1 的时候不允许改变 TIMEOUTA [11:0] 位域和 TIDLE 位的配置。

◆ $t_{\text{LOW:SEXT}}$ 和 $t_{\text{LOW:MEXT}}$ 检查

必须配置 12 位定时器 TIMEOUTB，从机模式会检查 $t_{\text{LOW:SEXT}}$ ，而主机模式会检查 $t_{\text{LOW:MEXT}}$ 。标准只规定了最大值，可选择与之相同的值。在 I2Cx_TIMEOCTR 寄存器的 TIMOUTEN 被置 1 使能了定时器后。如果 SMBus 外设将 SCL 拉低的时间达到了 $(\text{TIMEOUTB}+1) \times 2048 \times t_{\text{I2C\_CLK}}$ ，并且又没有达到 532 页总线空闲检查中描述的时间，I2Cx_ISR 寄存器中的 TIMEOUT 标志会被置 1。

警告：当 TEXTEN 位为 1 时，不允许改变 TIMEOUTB 的配置。

总线空闲检测

要使能 t_{IDLE} 检查, 必须根据要实现的 t_{IDLE} 参数向 TIMEOUTA[11:0] 位域写入预装载值。要同时检测 SCL 和 SDA 上的高电平超时, $TIDLE$ 须写为 1。

将 I2Cx_TIMEOUTR 寄存器中的 TIMOUTEN 位置 1, 使能了定时器后。

如果 SCL 和 SDA 线同时保持高的时间大于 $(TIMEOUTA+1) \times 4 \times t_{I2CCLK}$, I2Cx_ISR 寄存器中的 TIMEOUT 标志会被置 1。

警告: 在 TIMOUTEN 位为 1 的时候不允许改变 TIMEOUTA [11:0] 位域和 $TIDLE$ 位的配置。

1. 3. 9 SMBus: I2C_TIMEOUTR 寄存器配置举例

◆ 配置 TIMEOUT 最长到 25 毫秒:

f_{I2CCLK}	TIMEOUT[11:0] bits	$TIDLE$ bit	TIMOUTEN bit	$t_{TIMEOUT}$
8MHz	0x61	0	1	$98 \times 2048 \times 125 \text{ ns} = 25 \text{ ms}$
16MHz	0xC3	0	1	$196 \times 2048 \times 62.5 \text{ ns} = 25 \text{ ms}$
32MHz	0x186	0	1	$391 \times 2048 \times 31.25 \text{ ns} = 25 \text{ ms}$

表 18-4 各种 I2C_CLK 频率下 TIMEOUT 设置举例

◆ $t_{LOW:SEXT}$ 和 $t_{LOW:MEXT}$ 最长为 8 ms:

f_{I2CCLK}	TIMEOUT[11:0] bits	TXMOUTEN bit	$t_{LOW:EXT}$
8MHz	0x1F	1	$32 \times 2048 \times 125 \text{ ns} = 8 \text{ ms}$
16MHz	0x3F	1	$64 \times 2048 \times 62.5 \text{ ns} = 8 \text{ ms}$
32MHz	0x7C	1	$125 \times 2048 \times 31.25 \text{ ns} = 8 \text{ ms}$

表 18-5 各种 I2C_CLK 频率下 TIMEOUTB 设置举例

◆ t_{IDLE} 最长为 50 μs

f_{I2CCLK}	TIMEOUT[11:0] bits	$TIDLE$ bit	TIMOUTEN bit	t_{IDLE}
8MHz	0x63	1	1	$100 \times 4 \times 125 \text{ ns} = 50 \mu\text{s}$
16MHz	0xC7	1	1	$200 \times 4 \times 62.5 \text{ ns} = 50 \mu\text{s}$
32MHz	0x18F	1	1	$400 \times 4 \times 31.25 \text{ ns} = 50 \mu\text{s}$

表 18-6 各种 I2C_CLK 频率下 TIMEOUTA 设置举例

1. 3. 10 I2C DMA 请求

利用 DMA 发送

通过设置在 I2Cx_CR1 寄存器 TXDMAEN 位, 可以启用 DMA (直接存储器存取) 发送。数据被预先放到 DMA 外设所设定的 SRAM 区域发往 I2Cx_TXDR 寄存器, 而不管 TXE 位是不是 1。

只使用 DMA 传输数据。

利用 DMA 接收

通过设置在 I2Cx_CR1 寄存器 RXDMAEN 位, 可以启用 DMA (直接存储器存取) 接收。数据会从 I2Cx_RXDR 寄存器取出来, 转移到 DMA 外设中指向的 SRAM 区域, 而不管 RXNE 位是不是 1。只有数据 (包括 PEC) 被 DMA 传输。

1. 3. 11 I2C 错误条件

以下是可能导致通信失败的错误条件。

总线错误 (BERR)

如果在 9 倍 SCL 时钟脉冲以外检测到一个 START 或者 STOP 条件, 会发生一个总线错误。当 SCL 为高的时候在 SDA 上出现上升沿或者下降沿, 会检测到 START 或 STOP 条件。

只有在 I2C 处于数据传输状态下 (作为主机在发送或作为从机被寻址到) 才会有总线错误标志置位, 例如在从机模式下的寻址阶段就不会。

对于从机模式下, 一个错位的 START 或者 RESTART 条件会被从机当成正常的 START 条件来处理。

当检测到总线错误, I2Cx_ISR 寄存器中的 BERR 标志会被硬件置 1, 如果 I2Cx_CR1 寄存器中的 ERRIE 位为 1, 则会产生中断。

仲裁丢失 (ARLO)

在往 SDA 线上发高电平, 但在 SCL 的上升沿却从 SDA 采样得到低电平时, 就会检测到一次仲裁丢失错误。

- ◆ 在主机模式下, 仲裁丢失在地址阶段、数据阶段以及数据确认阶段进行检测。在这种情况下, SDA 和 SCL 线被释放, START 控制位由硬件清零, 主机自动切换到从机模式。
- ◆ 在从机模式下, 仲裁丢失在数据阶段和数据确认阶段进行检测。在这种情况下, 传输被终止, SCL 和 SDA 线被释放。

当检测到仲裁丢失错误, I2Cx_ISR 寄存器中的 ARLO 标志会被硬件置 1, 如果 I2Cx_CR1 寄存器中的 ERRIE 位为 1, 则会产生中断。

溢出/欠载错误 (OVR)

如果 NOSTRETCH=1, 在从机模式下只要下列条件发生, 就会检测到欠载或溢出错误:

- ◆ 在接收时, 尚未读取 RXDR 寄存器的时候又有一个新的字节接收到。新收到的字节会丢失, 并且会有一个 NACK 自动发送出去, 当作是对这个新的字节的响应。
- ◆ 在发送时:
 - 当应该发送第一个数据字节时却有 STOPF=1。如果 TXE=0, 那么 I2Cx_TXDR 寄存器的值会发送出去, 如果不是 0, 那么就会发送 0xFF。
 - 在一个新的字节应该被写入到 I2Cx_TXDR 寄存器, 但却没有写, 那就会将 0xFF 发送出去。

当检测到欠载/溢出错误，I2Cx_ISR 寄存器中的 OVR 标志会被硬件置 1，如果 I2Cx_CR1 寄存器中的 ERRIE 位为 1，则会产生中断。

1.3.12 I2C 中断

I2C 中的中断由一组 6 个寄存器控制。

◆ 中断控制 (IER, IDR, IVS)

I2C 中断使能寄存器(I2C_IER)通过写入“1”来启用中断请求。类似地，I2C 中断禁止寄存器(I2C_IDR)通过写入'1'来禁用中断请求。IER 和 IDR 是只写寄存器，以上寄存器的整体结果可以通过中断状态寄存器(I2C_IVS)来显示。IVS 是一个只读寄存器，使用“1”或“0”来表示中断请求是否有效。

◆ 中断标志状态(RIF)

I2C 中断标志寄存器(I2C_RIF)是一个只读寄存器，用于从模块读取中断状态。这个寄存器中的位显示 I2C 的真实状态。当观察到以下情况时，I2C 可以产生中断:

- SMBus 报警
- 遇到超时
- PEC 错误
- 溢出/欠载
- 仲裁丢失
- 总线错误
- 传输完成并重新加载
- 发送完成
- 检测到停止位
- 接收无应答
- 地址匹配
- 接收缓存非空
- 发送缓存空

◆ 中断屏蔽 (IFM)

I2C 中断标志屏蔽状态寄存器(I2C_IFM)用于读取模块的屏蔽中断状态，显示哪个中断被屏蔽。IFM 中的每个位是 IVS 和 RIF 中的各自的位进行逻辑与操作。

◆ 中断清除 (ICR)

向寄存器中的位写入一个“1”可以清除相应的中断。

1.4 I2C 寄存器描述

1.4.1 I2C 控制寄存器 1 (I2C_CR1)

		Address offset:	0x00
		Reset value:	0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved	RXFTH[2:0]			Reserved	TXFTH[2:0]			PECEN	ALERTEN	SMBDEN	SMBHEN	GCEN	Reserved	NOSTRETCH	SBC
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDMAEN	TXDMAEN	Reserved			DNF[3:0]			Reserved							PE

Name	Bit	Type	Reset value	Description
Reserved	31	N/A	0	保留并保持复位值。
RXFTH[2:0]	30:28	R/W	0x0	RXFTH[2:0]: Rx FIFO Threshold 此位提供了 Rx FIFO 触发 RXNE 阈值。 注意:如果启用了 Rx DMA, 此位也提供了 Rx DMA 请求的条件。
Reserved	27	N/A	0	保留并保持复位值。
TXFTH[2:0]	26:24	R/W	0x0	TXFTH[2:0]: Tx FIFO Threshold 此位提供 Tx FIFO 触发 TXE 阈值。 注意:如果启用 Tx DMA, 此位也提供 Tx DMA 请求的条件。
PECEN	23	R/W	0	PECEN: PEC enable 0: PEC 计算关闭 1: PEC 计算使能 注意:如果不支持 SMBus 特性, 此位将被保留, 并由硬件强制为“0”。
ALERTEN	22	R/W	0	ALERTEN: SMBus alert enable (SMBHEN=0): 0:释放 SMBA pin 为高并禁止警报响应地址头:跟在 NACK 后的 0001100x. 1: 驱动 SMBA pin 为低并使能警报响应地址头:跟在 ACK 后的 0001100x. Host mode (SMBHEN=1): 0: 不支持 SMBus 警告 pin (SMBA). 1: 支持 SMBus 警告 pin (SMBA). 注意:ALERTEN=0 时, SMBA 引脚可以用作标准 GPIO。如果不支持 SMBus 特性, 则保留此位, 并由硬件强制为“0”。
SMBDEN	21	R/W	0	SMBDEN: SMBus Device Default address enable 0: 设备默认地址禁用。地址 0b1100001x 无应答。 1: 设备默认地址使能。地址 0b1100001x 有应答。 注意:如果不支持 SMBus 特性, 此位将被保留, 并由硬件强制为“0”。
SMBHEN	20	R/W	0	SMBHEN: SMBus Host address enable 0: 主机地址禁用。地址 0b0001000x 无应答。 1: 主机地址使能。地址 0b0001000x 有应答。 注意:如果不支持 SMBus 特性, 此位将被保留, 并由硬件强制为“0”。
GCEN	19	R/W	0	GCEN: General call enable 0: 常规呼叫禁止。地址 0b00000000 无应答。 1: 常规呼叫使能。地址 0b00000000 有应答。
Reserved	18	N/A	0	保留并保持复位值。
NOSTRETCH	17	R/W	0	NOSTRETCH: Clock stretching disable 此位用于禁用从模式时钟扩展。在主模式下必须保持为 0。 0: 时钟扩展使能 1: 时钟扩展禁止 注:此位只能在 I2C 被禁用时进行编程(PE = 0)。

SBC	16	R/W	0	SBC: Slave byte control 此位用于从模式下使能硬件字节控制。 0: 从模式硬件字节控制禁止 1: 从模式硬件字节控制使能
RXDMAEN	15	R/W	0	RXDMAEN:DMAreception requests enable 0: DMA 接收模式禁止 1: DMA 接收模式使能
TXDMAEN	14	R/W	0	TXDMAEN:DMAtransmission requests enable 0: DMA 发送模式禁止 1: DMA 发送模式使能
Reserved	13;12	N/A	0	保留并保持复位值。
DNF[3:0]	11:8	R/W	0x0	DNF[3:0]: Digital noise filter 此位域用于配置 SDA 和 SCL 输入端数字噪声滤波器。数字滤波器可过滤长度达 DNF[3:0] * tI2CCLK 的噪声 0000: 数字滤波器关闭 0001: 数字滤波器使能且过滤能力高达 1 tI2CCLK ... 1111: 数字滤波器使能且过滤能力高达 15 tI2CCLK 注意:如果同时启用模拟滤波器,则将数字滤波器添加到模拟滤波器中。数字滤波器只能在 I2C 被禁用时进行编程(PE = 0)。
Reserved	7:1	N/A	0x0	保留,保持复位值
PE	0	R/W	0	PE: Peripheral enable 0: 外设禁止 1: 外设使能 注意:当 PE=0 时,释放 I2C SCL 和 SDA 线。内部状态机和状态位被恢复到它们的复位值。当清除时, PE 必须保持低电平至少 3 个 APB 时钟周期。

1. 4. 2 I2C 控制寄存器 2 (I2C_CR2)

Address offset:		0x04													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					PECBYTE	AUTOEND	RELOAD	NBYTES[7:0]							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NACK	STOP	START	HEAD10R	ADD10	RDWRN	SADD[9:0]									

Name	Bit	Type	Reset value	Description
Reserved	31:27	N/A	0x0	保留,保持复位值
PECBYTE	26	R/S	0	PECBYTE: Packet error checking byte 此位由软件设置,当传输 PEC、接收到 STOP、接收地址匹配或 PE=0 时,由硬件清除。 0: 没有 PEC 传输。 1: 请求 PEC 传输 / 接收 注意:此位写“0”无效。 当 RELOAD 设置为“1”时,此位无效。 当从模式下 SBC 为“0”时,此位无效。 如果不支持 SMBus 特性,则保留此位,并由硬件强制为“0”。

AUTOEND	25	R/W	0	AUTOEND: Automatic end mode (master mode) 此位由软件设置并清除。 0: 软件结束模式:在传输完成 NBYTES 数据后设置 TC 标志, 并将 SCL 拉低。 1: 自动结束模式:在传输完成 NBYTES 数据后自动发送 STOP 请求。 注意:此位在从模式或 RELOAD 设置为“1”时, 无效。
	24	R/W	0	RELOAD: NBYTES reload mode 此位由软件设置并清除。 0: 在 NBYTES 数据传输完成之后, 通信完成(跟 STOP 或 RESTART 信号)。 1: 在 NBYTES 数据传输完成之后, 通信没有完成(NBYTES 将被重新加载)。在传输 NBYTES 数据后 TCR 标志将被置“1”, 并将 SCL 拉低。
	23:16	R/W	0x0	NBYTES[7:0]: Number of bytes 设置要传输/接收的字节数。从模式且 SBC=0 时, 该位域无效。 注意:在设置 Start 后, 此位域不予许被修改。
	15	R/S	0	NACK: NACK generation (slave mode) 此位是由软件设置, 当发送 NACK、接收到 STOP、接收地址匹配或 PE=0 时, 由硬件清除。 0: 当前接收字节之后发送 ACK 信号。 1: 当前接收字节之后发送 NACK 信号。 注意:此位写“0”无效。 此位仅在从模式下使用:在主接收模式下, 无论 NACK 位值是多少, 都会在 STOP 或 RESTART 之前自动生成 NACK。当从接收 NOSTRETCH 模式下发生溢出时, 无论 NACK 位值是多少, 都会自动生成 NACK。当硬件 PEC 检查使能时(PECBYTE=1), PEC 确认值不依赖于 NACK 值。
	14	R/S	0	STOP: Stop generation (master mode) 此位由软件设置, 当检测到 STOP 信号或者 PE = 0 时, 硬件清“0”。 In Master Mode: 0: 没有 STOP 信号产生。 1: 当前字节传输后生成 STOP 信号。 注意:此位写“0”无效。
	13	R/S	0	START: Start generation 此位由软件设置, 在启动后发送地址序列、仲裁丢失、检测到超时错误或 PE = 0 时由硬件清“0”。也可以通过 I2C_ICR 寄存器的 ADDRCF 位写“1”来清除。 0: 没有 Start 信号产生。 1: Restart/Start 信号产生。 -如果 I2C 处于 AUTOEND=0 的主模式, 那么在 NBYTES 传输结束后, 当 RELOAD=0 时, 设置此位将生成一个重复 Start 信号。 -否则, 一旦总线空闲, 设置此位将生成一个 START 信号。 注意:此位写“0”无效。 即使总线 BUSY 或 I2C 处于从模式, START 位也可以设置。 当 RELOAD 设置为“1”时, 此位没有作用。 在 10 位寻址模式下, 如果在地址的第一部分后接收到一个 NACK, START 位不会被硬件清除且主机将重新发送地址序列, 除非 START 位被软件清除
	12	R/W	0	HEAD10R: 10-bit address header only read direction (master receiver mode) 0: 主机发送完整 10 位从机地址读序列: START+2 字节 10 位地址和 Write direction+RESTART+10 位地址的前 7 位和 Read direction。 注意:当 START 位被置“1”后, 此位不允许被更改。
	11	R/W	0	ADD10: 10-bit addressing mode (master mode) 0:主机在 7 位寻址模式下运行 1: 主机在 10 位寻址模式下运行 注意:不允许在起始条件置起后更改此位。

RD_WRN	10	R/W	0	RD_WRN: Transfer direction (master mode) 0: 主机请求写传输。 1: 主机请求读传输。 注意: 不允许在起始条件置起后更改此位。
SADD[9:8]	9:8	R/W	0x0	SADD[9:8]: Slave address bit 9:8 (master mode) 7 位寻址模式(ADD10 = 0): 这些位无关紧要 10 位寻址模式(ADD10 = 1): 这些位写入要发送的从地址的 Bit9:8 注意: 不允许在起始条件置起后更改此位。
SADD[7:1]	7:1	R/W	0x0	SADD[7:1]: Slave address bit 7:1 (master mode) 7 位寻址模式(ADD10 = 0): 这些位写入要发送的 7 位从地址 10 位寻址模式(ADD10 = 1): 这些位写入要发送的从地址的 Bit7:1。 注意: 不允许在起始条件置起后更改此位。
SADD[0]	0	R/W	0x0	SADD[0]: Slave address bit 0 (master mode) 7 位寻址模式(ADD10 = 0): 此位无关紧要 10 位寻址模式(ADD10 = 1): 此位写入要发送的从地址的 Bit0 注意: 不允许在起始条件置起后更改此位。

1. 4. 3 I2C 本机地址 1 寄存器(I2C_OAR1)

Address offset:		0x08													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OA1EN	Reserved				OA1MODE	OA1[9:8]			OA1[7:1]						OA1[0]

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留, 保持复位值
OA1EN	15	R/W	0	OA1EN: 从机接口地址 1 使能位 0: 从机地址 1 禁止, 从机接收地址 OA1 无应答 1: 从机地址 1 使能, 从机接收地址 OA1 应答
Reserved	14:11	N/A	0x0	保留, 保持复位值
OA1MODE	10	R/W	0	OA1MODE: 从机接口地址 1 寻址模式 0: 7 位从机接口地址 1: 10 位从机接口地址 注: 此 Bit 位只有在 OA1EN=0 时可写
OA1[9:8]	9:8	R/W	0x0	OA1[9:8]: 10 位从机接口地址 1 的 8-9 位 7 位从机寻址模式: 无意义 10 位从机寻址模式: 10 位接口地址中的 9-8 位 注: 这些 Bit 位只有在 OA1EN=0 可写。

OA1[7:1]	7:1	R/W	0x0	OA1[7:1]: 10 位从机接口地址 1 的 1-7 位或 7 位从机接口地址 7 位从机寻址模式: 7 位接口地址 10 位从机寻址模式: 10 位接口地址中的 7-1 位 注: 这些 Bit 位只有在 OA1EN=0 可写
OA1[0]	0	R/W	0	OA1[0]: 10 位从机接口地址 1 的第 0 位 7 位从机寻址模式: 无意义 10 位从机寻址模式: 10 位从机接口地址的第 0 位 注: 此 Bit 位只有在 OA1EN=0 时可写

1.4.4 I2C 本机地址 2 寄存器 (I2C_OAR2)

		Address offset:		0x0C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OA2EN	Reserved				OA2MSK[2:0]				OA2[7:1]				Reserved		

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留, 保持复位值
OA2EN	15	R/W	0	OA2EN: 从机接口地址 2 使能位 0: 本机地址 2 禁止, 从机接收地址 OA2 无应答 1: 本机地址 2 使能, 从机接收地址 OA2 有应答
Reserved	14:11	N/A	0x0	保留, 保持复位值
OA2MSK[2:0]	10:8	R/W	0x0	OA2MSK[2:0]: 从机接口地址 2 屏蔽位 000: 无屏蔽 001: OA2[1] 被屏蔽。仅对 OA2[7:2] 进行比较 010: OA2[2:1] 被屏蔽。仅对 OA2[7:3] 进行比较 011: OA2[3:1] 被屏蔽。仅对 OA2[7:4] 进行比较 100: OA2[4:1] 被屏蔽。仅对 OA2[7:5] 进行比较 101: OA2[5:1] 被屏蔽。仅对 OA2[7:6] 进行比较 110: OA2[6:1] 被屏蔽。仅对 OA2[7] 进行比较 111: OA2[7:1] 被屏蔽。不进行比较, 且对所有 7 位地址都应答 注: 这些 Bit 位只有在 OA2EN=0 可写 一旦 OA2MSK 不等于 0, I2C 接收地址即使匹配, 也不会应答保留地址 (0b0000xxx 和 0b1111xxx)
OA2[7:1]	7:1	R/W	0x0	OA2[7:1]: 7 位从机接口地址 2 7 位寻址模式: 7 位地址 注: 这些 Bit 位只有在 OA2EN=0 可写
Reserved	0	N/A	0	保留, 保持复位值

1.4.5 I2C 时序寄存器(I2C_TIMINGR)

		Address offset:		0x10											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

PRESC[3:0]				Reserved				SCLDEL[3:0]				SDADEL[3:0]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCLH[7:0]								SCLL[7:0]							

Name	Bit	Type	Reset value	Description
PRESC[3:0]	31:28	R/W	0x0	PRESC[3:0]: Timing prescaler 此位用于预分频 I2CCLK，以生成用于数据建立（setup）和保持（hold）计数器以及用于 SCL 高电平和低电平计数器的时钟周期。 $t_{PRESC} = (PRESC+1) \times t_{I2CCLK}$ 此寄存器必须在 I2C 关闭时 (PE = 0) 进行配置。
Reserved	27:24	N/A	0x0	保留，保持复位值
SCLDEL[3:0]	23:20	R/W	0x0	SCLDEL[3:0]: Data setup time 此位用于生成 SDA 边缘和 SCL 上升沿之间的延迟 tSCLDEL。在 NOSTRETCH = 0 的主模式和从模式下，SCL 在 tSCLDEL 期间低电平拉伸。 $t_{SCLDEL} = (SCLDEL+1) \times t_{PRESC}$ 注意:tSCLDEL 用于生成 tSU:DAT 时序。 此寄存器必须在 I2C 关闭时 (PE = 0) 进行配置。
SDADEL[3:0]	19:16	R/W	0x0	SDADEL[3:0]: Data hold time 此位用于生成 SCL 下降沿和 SDA 边缘之间的延迟 tSDADEL。在 NOSTRETCH = 0 的主模式和从模式下，SCL 在 tSDADEL 期间低电平拉伸。 $t_{SDADEL} = SDADEL \times t_{PRESC}$ 注:SDADEL 用于生成 tHD:DAT 时序。 此寄存器必须在 I2C 关闭时 (PE = 0) 进行配置。
SCLH[7:0]	15:8	R/W	0x0	SCLH[7:0]: SCL high period (master mode) 此位在主模式下用于生成 SCL 高电平周期。 $t_{SCLH} = (SCLH+1) \times t_{PRESC}$ 注 :SCLH 也用于生成 tSU:STO 和 tHD:STA 时序。 此寄存器必须在 I2C 关闭时 (PE = 0) 进行配置。
SCLL[7:0]	7:0	R/W	0x0	SCLL[7:0]: SCL low period (master mode) 此位在主模式下用于生成 SCL 低电平周期。 $t_{SCLL} = (SCLL+1) \times t_{PRESC}$ 注:SCLL 还用于生成 tBUF 和 tSU:STA 时序。 此寄存器必须在 I2C 关闭时 (PE = 0) 进行配置。

1. 4. 6 I2C 超时寄存器(I2C _TIMEOUTR)

		Address offset:		0x14											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TEXTEN	Reserved			TIMEOUTB[11:0]											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMEOUTEN	Reserved		TIDLE	TIMEOUTA[11:0]											

Name	Bit	Type	Reset value	Description

TEXTEN	31	R/W	0	TEXTEN: 累积时钟延长超时使能 0: 禁用累积时钟延长超时检测 1: 启用累积时钟延长超时检测。 当 I2C 接口完成累积 SCL 延长超过 TLOW: EXT 时, 会检测到超时错误 (TIMEOUT = 1)。
Reserved	30:28	N/A	0x0	保留, 保持复位值
TIMEOUTB[11:0]	27:16	R/W	0x0	TIMEOUTB[11:0]: 总线超时 B 该字段用于配置累积时钟延长超时: 在主机模式下, 检测到主机累积时钟低延长时间 (TLOW:MEXT)。在从机模式下, 检测到从机累积时钟低延长时间 (TLOW:SEXT)。 $TLOW:EXT = (TIMEOUTB + 1) \times 2048 \times TI2CCLK$ 注: 仅当 TEXTEN=0 时才能写入这些位。
TIMEOUTEN	15	R/W	0	TIMEOUTEN: 时钟超时启用 0: 禁用 SCL 超时检测 1: 启用 SCL 超时检测: 当 SCL 为低电平超过 TTIMEOUT (TIDLE = 0) 或高电平超过 TIDLE (TIDLE = 1) 时, 检测到超时错误 (TIMEOUT = 1)
Reserved	14:13	N/A	0x0	保留, 保持复位值
TIDLE	12	R/W	0	TIDLE: 空闲时钟超时检测 0: TIMEOUTA 用于检测 SCL 低超时 1: TIMEOUTA 用于检测 SCL 和 SDA 高超时 (总线空闲状态) 注: 仅当 TIMEOUTEN = 0 时才能写入该位。
TIMEOUTA[11:0]	11:0	R/W	0x0	TIMEOUTA[11:0]: 总线超时 A 该字段用于配置: - 当 TIDLE=0 时, SCL 低超时条件 TTIMEOUT $T_{TIMEOUT} = (TIMEOUTA + 1) \times 2048 \times TI2CCLK$ - 当 TIDLE=1 时, 总线空闲状态 (SCL 和 SDA 均为高电平) $T_{IDLE} = (TIMEOUTA + 1) \times 4 \times TI2CCLK$ 注: 仅当 TIMEOUTEN=0 时才能写入这些位。

1. 4. 7 I2C 状态寄存器 (I2C SR)

Address offset:		0x18													
Reset value:		0x0000 0001													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXFLV[3:0]				TXFLV[3:0]				ADDCODE[6:0]						DIR	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUSY	Reserved	ALERT	TIMEOUT	PECE RR	OVR	ARLO	BERR	TCR	TC	STOP F	NACK F	ADDR	RXNE	Reserved	TXE

Name	Bit	Type	Reset value	Description
RXFLV[3:0]	31:28	R	0x0	RXFLV[3:0]: Rx FIFO Level 值。
TXFLV[3:0]	27:24	R	0x0	TXFLV[3:0]: Tx FIFO Level 值。
ADDCODE[6:0]	23:17	R	0x0	ADDCODE[6:0]: Address match code (Slave mode) 当地址匹配发生时 (ADDR=1), 此位用于更新接收到的地址在 10 位地址的情况下, ADDCODE 指示地址中除去 2 个 MSB 的包头

DIR	16	R	0	DIR: Transfer direction (Slave mode) 当地址匹配事件 (ADDR=1) 发生时, 将更新此标志。 0: 写传输, 从机进入接收模式。 1: 读取传输, 从机进入发射模式。
BUSY	15	R	0	BUSY: Bus busy 此标志指示总线正在进行通信。当检测到启动条件时, 硬件将此标志置起。当检测到停止条件或 PE=0 时, 硬件清除。
Reserved	14	N/A	0x0	保留, 保持复位值
ALERT	13	R	0	ALERT: SMBus alert 当 SMBHEN=1 (SMBus 主机配置)、ALERTEN=1 和 SMBA 引脚上检测到 SMBALERT 事件(下降沿)时, 硬件将此标志置起。软件设置 ALERTCF 位对其进行清除。 注意:当 PE=0 时, 此位被硬件清除。 如果不支持 SMBus 特性, 则保留此位, 并由硬件强制'0'。
TIMEOUT	12	R	0	TIMEOUT: Timeout or tLOW detection flag 当超时或扩展时钟超时发生时, 硬件将此标志置起。软件设置 TIMEOUTCF 位对其进行清除。 注意:当 PE=0 时, 此位被硬件清除。 如果不支持 SMBus 特性, 则保留此位, 并由硬件强制'0'。
PECERR	11	R	0	PECERR: PEC Error in reception 当接收到的 PEC 与 PEC 寄存器内容不匹配时, 硬件将此标志置起。在错误的 PEC 接收之后会自动发送一个 NACK。软件设置 PECCF 位对其进行清除。 注意:当 PE=0 时, 此位被硬件清除。 如果不支持 SMBus 特性, 则保留此位并由硬件强制'0'。
OVR	10	R	0	OVR: Overrun/Underrun (slave mode) 当发生上溢/下溢错误时, 硬件在从模式且 NOSTRETCH=1 时, 硬件将此标志置起。软件设置 OVRCF 位对其进行清除。 注意:当 PE=0 时, 此位被硬件清除。
ARLO	9	R	0	ARLO: Arbitration lost 当仲裁丢失, 此标志由硬件设置。软件设置 ARLOCF 位对其进行清除。 注意:当 PE=0 时, 此位被硬件清除。
BERR	8	R	0	BERR: Bus error 当外设进行传输时, 硬件检测到错误的启动或停止条件, 硬件将此标志置起。从模式下的地址阶段不设置该标志。软件设置 BERRCF 位对其进行清除。 注意:当 PE=0 时, 此位被硬件清除。
TCR	7	R	0	TCR: Transfer Complete Reload 当 RELOAD=1 和 NBYTES 数据已被传输时, 硬件将此标志置起。当 NBYTES 被软件写入非零值时被清除。 注意:当 PE=0 时, 此位被硬件清除。 此标志仅适用于主模式或从模式设置了 SBC 位。
TC	6	R	0	TC: Transfer Complete (master mode) 当 RELOAD=0、AUTOEND=0 和 NBYTES 数据已被传输时, 硬件将此标志置起。当软件设置开始条件或停止条件时被清除。 注意:当 PE=0 时, 此位被硬件清除。
STOPF	5	R	0	STOPF: Stop detection flag 当和外设进行通信时, 总线上检测到一个停止条件时, 硬件将此标志置起: -如果停止条件是由外围设备生成的, 则可以作为主设备。 -或作为一个从设备, 前提是外围设备已在此转移之前被寻址。 软件通过设置 STOPCF 位对其进行清除。 注意:当 PE=0 时, 此位被硬件清除

NACKF	4	R	0	NACKF: Not Acknowledge received flag 当字节传输之后接收到 NACK 时，硬件将此标志置起。软件通过设置 NACKCF 位对其进行清除。 注意:当 PE=0 时，此位被硬件清除。
ADDR	3	R	0	ADDR: Address matched (slave mode) 当接收到的从地址与一个有效从地址匹配，硬件将此标志置起。软件通过设置 ADDR CF 位对其进行清除。 注意:当 PE=0 时，此位被硬件清除。
RXNE	2	R	0	RXNE: Receive data register not empty (receivers) 当接收到的数据被复制到 I2C_RXDR 寄存器，并可被读取时，硬件将此标志置起。软件读取 I2C_RXDR 时被清除。 注意:当 PE=0 时，此位被硬件清除。
Reserved	1	N/A	0	保留，保持复位值
TXE	0	R	1	TXE: Transmit data register empty (transmitters) 当 I2C_TXDR 寄存器为空时，硬件将此标志置起。当软件将要发送的下一个数据写入 I2C_TXDR 寄存器时被清除。 软件可以对该位写入“1”来刷新传输数据寄存器 I2C_TXDR。 注意:当 PE=0 时，此位由硬件设置

1. 4. 8 I2C 清标志寄存器(I2C_CFR)

		Address offset:		0x1C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ALER TCF	TIMO UTCF	PECC F	OVR C F	ARLO CF	BERR CF	Reserved		STOP CF	NACK CF	ADD RCF	Reserved		

Name	Bit	Type	Reset value	Description
Reserved	31:14	N/A	0x0	保留，保持复位值
ALERTCF	13	W	0	ALERTCF: Alert flag clear 此位写 1 清除 I2C_SR 寄存器中的 ALERT 标志位。 注意:如果不支持 SMBus 特性，这个位将被保留，并由硬件强制为'0'。
TIMOUTCF	12	W	0	TIMOUTCF: Timeout detection flag clear 此位写 1 清除 I2C_SR 寄存器中的 TIMEOUT 标志位。 注意:如果不支持 SMBus 特性，这个位将被保留，并由硬件强制为'0'。
PECCF	11	W	0	PECCF: PEC Error flag clear 此位写 1 清除 I2C_SR 寄存器中的 PECERR 标志位。 注意:如果不支持 SMBus 特性，这个位将被保留，并由硬件强制为'0'。
OVR CF	10	W	0	OVR CF: Overrun/Underrun flag clear 此位写 1 清除 I2C_SR 寄存器中的 OVR 标志位。
ARLOCF	9	W	0	ARLOCF: Arbitration Lost flag clear 此位写 1 清除 I2C_SR 寄存器中的 ARLO 标志位。

BERRCF	8	W	0	BERRCF: Bus error flag clear 此位写 1 清除 I2C_SR 寄存器中的 BERRF 标志位。
Reserved	7:6	N/A	0	保留, 保持复位值
STOPCF	5	W	0	STOPCF: Stop detection flag clear 此位写 1 清除 I2C_SR 寄存器中的 STOPF 标志位。
NACKCF	4	W	0	NACKCF: Not Acknowledge flag clear 此位写 1 清除 I2C_SR 寄存器中的 ACKF 标志位。
ADDRCF	3	W	0	ADDRCF: Address matched flag clear 此位写 1 清除 I2C_SR 寄存器中的 ADDR 标志位。W 此位写 1 同样可以清除 I2C_CR2 寄存器中的 START 位。
Reserved	2:0	N/A	0	保留, 保持复位值

1. 4. 9 I2C PEC 寄存器(I2C_PECR)

		Address offset:		0x20											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								PEC[7:0]							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留, 保持复位值
PEC[7:0]	7:0	R	0x0	PEC[7:0]: Packet error checking register 当 PECEN=1 时, 该字段包含内部 PEC。 当 PE=0 时, 硬件将清除 PEC。

1. 4. 10 I2C 接收数据寄存器 (I2C_RXDR)

		Address offset:		0x24											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RXDATA[7:0]							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留, 保持复位值
RXDATA[7:0]	7:0	R	0x0	RXDATA[7:0]: 8-bit receive data 从 I2C 总线接收的数据字节。

1. 4. 11 I2C 发送数据寄存器 (I2C_TXDR)

		Address offset:		0x28											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TXDATA[7:0]							

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x0	保留，保持复位值
TXDATA[7:0]	7:0	W	0x0	TXDATA[7:0]: 8-bit transmit data 要传输到 I2C 总线的数据字节。 注意:这些位只能在 TXE=1 时编写。

1. 4. 12 I2C 中断使能寄存器 (I2C_IER)

		Address offset:		0x2C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ALER TIE	TOUT IE	PECEI E	OVR I E	ARLO IE	BERR IE	TCRIE	TCIE	STOP IE	NACK IE	ADD RIE	RXNE IE	Reser ved	TXEIE

Name	Bit	Type	Reset value	Description
Reserved	31:14	N/A	0x0	保留，保持复位值
ALERTIE	13	W	0	ALERTIE: SMBus Alert interrupt enable 0: 无作 1: 使能中断
TOUTIE	12	W	0	TOUTIE: Timeout interrupt enable 0: 无作 1: 使能中断
PECEIE	11	W	0	PECEIE: PEC error interrupt enable 0: 无作 1: 使能中断
OVRIE	10	W	0	OVRIE: Overrun/Underrun interrupt enable 0: 无作 1: 使能中断
ARLOIE	9	W	0	ARLOIE: Arbitration loss interrupt enable 0: 无作 1: 使能中断
BERRIE	8	W	0	BERRIE: Bus error interrupt enable 0: 无作 1: 使能中断

TCRIE	7	W	0	TCRIE: Transfer complete and Reload interrupt enable 0: 无作 1: 使能中断
TCIE	6	W	0	TCIE: Transfer complete interrupt enable 0: 无作 1: 使能中断
STOPIE	5	W	0	STOPIE: Stop detection interrupt enable 0: 无作 1: 使能中断
NACKIE	4	W	0	NACKIE: NACK reception interrupt enable 0: 无作 1: 使能中断
ADDRIE	3	W	0	ADDRIE: Address matched interrupt enable 0: 无作 1: 使能中断
RXNEIE	2	W	0	RXNEIE: Receive buffer not empty interrupt enable 0: 无作 1: 使能中断
Reserved	1	N/A	0	保留, 保持复位值
TXEIE	0	W	0	TXEIE: Transmit buffer empty interrupt enable 0: 无作 1: 使能中断

1. 4. 13 I2C 中断禁止寄存器(I2C_IDR)

		Address offset:		0x30											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ALERTID	TOUTID	PECEID	OVRIID	ARLOID	BERRID	TCRIID	TCID	STOPID	NACKID	ADDRID	RXNEID	Reserved	TXEIID

Name	Bit	Type	Reset value	Description
Reserved	31:14	N/A	0x0	保留, 保持复位值
ALERTID	13	W	0	ALERTID: SMBus Alert interrupt disable 0: 无作 1: 禁止中断
TOUTID	12	W	0	TOUTID: Timeout interrupt disable 0: 无作 1: 禁止中断
PECEID	11	W	0	PECEID: PEC error interrupt disable 0: 无 1: 禁止中断
OVRIID	10	W	0	OVRIID: Overrun/Underrun interrupt disable 0: 无 1: 禁止中断
ARLOID	9	W	0	ARLOID: Arbitration loss interrupt disable 0: 无 1: 禁止中断
BERRID	8	W	0	BERRID: Bus error interrupt disable 0: 无 1: 禁止中断

TCRID	7	W	0	TCRID: Transfer complete and Reload interrupt disable 0: 无作 1: 禁止中断
TCID	6	W	0	TCID: Transfer complete interrupt disable 0: 无作 1: 禁止中断
STOPID	5	W	0	STOPID: Stop detection interrupt disable 0: 无作 1: 禁止中断
NACKID	4	W	0	NACKID: NACK reception interrupt disable 0: 无作 1: 禁止中断
ADDRID	3	W	0	ADDRID: Address matched interrupt disable 0: 无作 1: 禁止中断
RXNEID	2	W	0	RXNEID: Receive buffer not empty interrupt disable 0: 无作 1: 禁止中断
Reserved	1	N/A	0	保留，保持复位值
TXEID	0	W	0	TXEID: Transmit buffer empty interrupt disable 0: 无作 1: 禁止中断

1. 4. 14I2C 中断有效状态寄存器(I2C_IVS)

		Address offset:		0x34											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ALERTIV	TOUTIV	PECEIV	OVRIV	ARLOIV	BERRIV	TCRIV	TCIV	STOPIV	NACKIV	ADDRIV	RXNEIV	Reserved	TXEIV

Name	Bit	Type	Reset value	Description
Reserved	31:14	N/A	0x0	保留，保持复位值
ALERTIV	13	R	0	ALERTIV: SMBusAlert interrupt valid 0: 无作 1: 有效中断
TOUTIV	12	R	0	TOUTIV: Timeout interrupt valid 0: 无作 1: 有效中断
PECEIV	11	R	0	PECEIV: PEC error interrupt valid 0: 无作 1: 有效中断
OVRIV	10	R	0	OVRIV: Overrun/Underrun interrupt valid 0: 无作 1: 有效中断
ARLOIV	9	R	0	ARLOIV: Arbitration loss interrupt valid 0: 无作 1: 有效中断
BERRIV	8	R	0	BERRIV: Bus error interrupt valid 0: 无作 1: 有效中断

TCRIV	7	R	0	TCRIV: Transfer complete and Reload interrupt valid 0: 无作 1: 有效中断
TCIV	6	R	0	TCIV: Transfer complete interrupt valid 0: 无作 1: 有效中断
STOPIV	5	R	0	STOPIV: Stop detection interrupt valid 0: 无作 1: 有效中断
NACKIV	4	R	0	NACKIV: NACK reception interrupt valid 0: 无作 1: 有效中断
ADDRIV	3	R	0	ADDRIV: Address matched interrupt valid 0: 无作 1: 有效中断
RXNEIV	2	R	0	RXNEIV: Receive buffer not empty interrupt valid 0: 无作 1: 有效中断
Reserved	1	N/A	0	保留，保持复位值
TXEIV	0	R	0	TXEIV: Transmit buffer empty interrupt valid 0: 无作 1: 有效中断

1. 4. 15 I2C 原始中断标志寄存器 (I2C_RIF)

		Address offset:		0x38											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ALER TRI	TOUT RI	PECE RI	OVR RI	ARLO RI	BERR RI	TCRR I	TCRI	STOP RI	NACK RI	ADD RRI	RXNE RI	Reser ved	TXERI

Name	Bit	Type	Reset value	Description
Reserved	31:14	N/A	0x0	保留，保持复位值
ALERTRI	13	R	0	ALERTRI: SMBus Alert raw interrupt flag status 0: 没 有 中 断 1: 中断产生
TOUTRI	12	R	0	TOUTRI: Timeout raw interrupt flag status 0: 没 有 中 断 1: 中断产生
PECERI	11	R	0	PECERI: PEC error raw interrupt flag status 0: 没 有 中 断 1: 中断产生
OVRRI	10	R	0	OVRRI: Overrun/Underrun raw interrupt flag status 0: 没 有 中 断 1: 中断产生
ARLORI	9	R	0	ARLORI: Arbitration loss raw interrupt flag status 0: 没 有 中 断 1: 中断产生
BERRRI	8	R	0	BERRRI: Bus error raw interrupt flag status 0: 没 有 中 断 1: 中断产生

TCRRI	7	R	0	TCRRI: Transfer complete and Reload raw interrupt flag status 0: 没 有 中 断 1: 中断产生
TCRI	6	R	0	TCRI: Transfer complete raw interrupt flag status 0: 无 作 用 1: 原始中断标志状态
STOPRI	5	R	0	STOPRI: Stop detection raw interrupt flag status 0: 没 有 中 断 1: 中断产生
NACKRI	4	R	0	NACKRI: NACK reception raw interrupt flag status 0: 没 有 中 断 1: 中断产生
ADDRRI	3	R	0	ADDRRI: Address matched raw interrupt flag status 0: 没 有 中 断 1: 中断产生
RXNERI	2	R	0	RXNERI: Receive buffer not empty raw interrupt flag status 0: 没 有 中 断 1: 中断产生
Reserved	1	N/A	0	保留, 保持复位值
TXERI	0	R	0	TXERI: Transmit buffer empty raw interrupt flag status 0: 没 有 中 断 1: 中断产生

1. 4. 16 I2C 屏蔽中断标志寄存器(I2C_IFM)

		Address offset:		0x3C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ALER TFM	TOUT FM	PECE FM	OVRF M	ARLO FM	BERR FM	TCRF M	TCF M	STOP FM	NACK FM	ADD RFM	RXNE FM	Reser ved	TXEF M

Name	Bit	Type	Reset value	Description
Reserved	31:14	N/A	0x0	保留, 保持复位值
ALERTFM	13	R	0	ALERTFM: SMBus Alert interrupt flag masked status 0: 没 有 中 断 1: 中断产生
TOUTFM	12	R	0	TOUTFM: Timeout interrupt flag masked status 0: 没 有 中 断 1: 中断产生
PECEFM	11	R	0	PECEFM: PEC error interrupt flag masked status 0: 没 有 中 断 1: 中断产生
OVRFM	10	R	0	OVRFM: Overrun/Underrun interrupt flag masked status 0: 无 作 用 1: 中断标志屏蔽状态
ARLOFM	9	R	0	ARLOFM: Arbitration loss interrupt flag masked status 0: 没 有 中 断 1: 中断产生
BERRFM	8	R	0	BERRFM: Bus error interrupt flag masked status 0: 没 有 中 断 1: 中断产生

TCRFM	7	R	0	TCRFM: Transfer complete and Reload interrupt flag masked status 0: 没 有 中 断 1: 中断产生
TCFM	6	R	0	TCFM: Transfer complete interrupt flag masked status 0: 没 有 中 断 1: 中断产生
STOPFM	5	R	0	STOPFM: Stop detection interrupt flag masked status 0: 没 有 中 断 1: 中断产生
NACKFM	4	R	0	NACKFM: NACK reception interrupt flag masked status 0: 无 作 用 1: 中断标志屏蔽状态
ADDRFM	3	R	0	ADDRFM: Address matched interrupt flag masked status 0: 没 有 中 断 1: 中断产生
RXNEFM	2	R	0	RXNEFM: Receive buffer not empty interrupt flag masked status 0: 没 有 中 断 1: 中断产生
Reserved	1	N/A	0	保留, 保持复位值
TXEFM	0	R	0	TXEFM: Transmit buffer empty interrupt flag masked status 0: 没 有 中 断 1: 中断产生

1. 4. 17 I2C 中断清除标志(I2C_ICR)

		Address offset:		0x3C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ALERTIC	TOUTIC	PECEIC	OVRIC	ARLOIC	BERRIC	TCRIC	TCIC	STOPIC	NACKIC	ADDRIC	RXNEIC	Reserved	TXEIC

Name	Bit	Type	Reset value	Description
Reserved	31:14	N/A	0x0	保留, 保持复位值
ALERTIC	13	W1C	0	ALERTIC: SMBus Alert interrupt clear 0: 清除中断 1: 清除中断
TOUTIC	12	W1C	0	TOUTIC: Timeout interrupt clear 0: 清除中断 1: 清除中断
PECEIC	11	W1C	0	PECEIC: PEC error interrupt clear 0: 清除中断 1: 清除中断
OVRIC	10	W1C	0	OVRIC: Overrun/Underrun interrupt clear 0: 清除中断 1: 清除中断
ARLOIC	9	W1C	0	ARLOIC: Arbitration loss interrupt clear 0: 清除中断 1: 清除中断

BERRIC	8	W1C	0	BERRIC: 0: 无 1: 清除中断	Bus error	interrupt 作	clear 用
TCRIC	7	W1C	0	TCRIC: Transfer 0: 无 1: 清除中断	complete and	Reload interrupt 作	clear 用
TCIC	6	W1C	0	TCIC: Transfer 0: 无 1: 清除中断	complete	interrupt 作	clear 用
STOPIC	5	W1C	0	STOPIC: 0: 无 1: 清除中断	Stop detection	interrupt 作	clear 用
NACKIC	4	W1C	0	NACKIC: 0: 无 1: 清除中断	NACK reception	interrupt 作	clear 用
ADDRIC	3	W1C	0	ADDRIC: 0: 无 1: 清除中断	Address matched	interrupt 作	clear 用
RXNEIC	2	W1C	0	RXNEIC: 0: 无 1: 清除中断	Receive buffer not empty	interrupt 作	clear 用
Reserved	1	N/A	0	保留，保持复位值			
TXEIC	0	W1C	0	TXEIC: 0: 无 1: 清除中断	Transmit buffer empty	interrupt 作	clear 用

第19章 串行外设接口 1(SSI)

1.1 SSI 简介

SSI 是一个全双工主/从同步串行接口。主机处理器通过 APB 接口访问 SSI 上的数据、控制和状态信息。SSI 通过一组 dma 信号与 DMA 控制器进行连接，SSI 可配置为两种操作模式之一：串行主模式或串行从模式。SSI 可以使用以下接口之一连接到任何串行主或串行从外围设备：

- ◆ 摩托罗拉串行外围接口(SPI)：来自摩托罗拉的四线全双工串行协议。串行时钟相位和极性有四种可能的组合。时钟相位(SCPH)决定串行传输是从从选择信号的下降沿开始，还是从串行时钟的第一个边缘开始。当 SSI 空闲或禁用时，从机选择线保持高位。有关更多信息，请参阅“摩托罗拉串行外围接口(SPI)”
- ◆ 德州仪器串行协议(SSP)：一种四线全双工串行协议。被用于 SPI 和 Microwire 协议的从机选择线同时作为 SSP 协议的帧指示。有关更多信息，请参阅“德州仪器同步串行协议(SSP)”。
- ◆ 国家半导体微线：半双工串行协议，它使用一个控制字从串行主机传输到目标串行从机。有关更多信息，请参阅“国家半导体微线”。

可以对控制寄存器 0 (SSI_CR0)中的 FRF(帧格式)位字段进行编程，以选择使用哪个协议。



图 19-1 SSI 结构框图

1.2 SSI 功能描述

SSI 支持的串行协议允许通过硬件或软件的选择或寻址到串行从机。在硬件上实现时，串行从机是在专用硬件控制选择线下选择的。从串行主机生成的选择行数等于总线上的串行从机的数量。串行主设备在开始数据传输之前判断目标串行从机的选择线。

当在软件中实现时，所有串行从设备的输入选择应起源于串行主机上的单个从机选择输出。在这种模式下，假定串行主机只有一个从机选择输出。如果系统中有多个串行主机，所有主机的从机选择输出可以逻辑地生成一个从机选择输入，用于所有串行从设备。

软件域中的主程序控制目标从设备的选择;软件将使用所有从设备的 SSIENR 寄存器，以控制哪个从设备来响应串行主设备发出的传输请求。

下面的示例是伪代码，演示了如何使用软件来选择目标从设备。

软件选择目标从设备的例子

```
int main() {  
  
    // This function sets the SSI_EN bit to logic '0' in the SSIENR register  
    // of each device on the serial bus  
    disable_all_serial_devices();  
  
    // This function initializes the master device for the serial transfer  
    // 1. Write SSI_CR0 to match the required transfer  
    // 2. If transfer is receive only write number of frames into CTRLR1  
    // 3. Write BAUDR to set the transfer baud rate.  
    // 4. Write TXFTLR and RXFTLR to set FIFO threshold levels  
    // 5. Write IMR register to set interrupt masks  
    // 6. Write SER register bit[0] to logic '1'  
    // 7. Write SSIENR register bit[0] to logic '1' to enable the master.  
    initialize_mst(ssi_mst_1);  
  
    // This function initializes the target slave device (slave 1 in this example)  
    // for the serial transfer.  
    // 1. Write SSI_CR0 to match the required transfer  
    // 2. Write TXFTLR and RXFTLR to set FIFO threshold levels  
    // 3. Write IMR register to set interrupt masks  
    // 4. Write SSIENR register bit[0] to logic '1' to enable the slave.  
    // 5. If the slave is to transmit data, write data into TX FIFO  
    // Now the slave is enabled and awaiting an active level on its  
    // ss_in_n input port. Note all other serial slaves are disabled (SSI_EN=0)  
    // and therefore will not respond to an active level on their ss_in_n port.  
    initialize_slv(ssi_slv_1);  
  
    // This function begins the serial transfer by writing transmit data into  
    // the master's TX FIFO.  
    start_serial_xfer(ssi_mst_1);  
}
```

```
// User can poll the busy status with a function or use an ISR to determine
// when the serial transfer has completed. }
```

The SSI does not enforce hardware or software control for serial-slave device selection. You can configure the SSI for either implementation, illustrated in Figure 3-1.

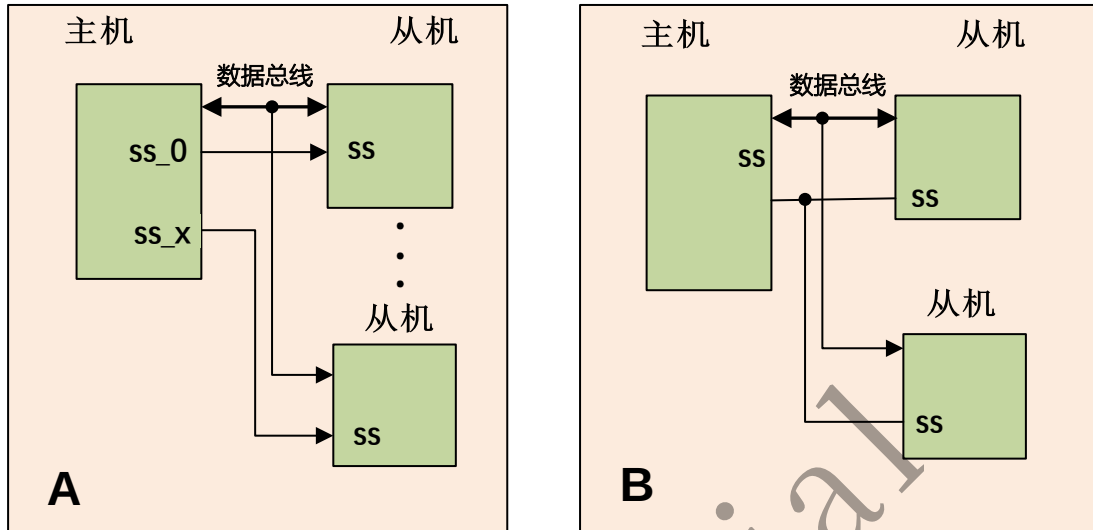


图 19-2 硬件/软件从设备选择

1.2.1 时钟比率

当 SSI 被配置为一个主设备时，时钟速率(sclk_out)的最大频率是 ssi_clk 频率的一半。这允许移位控制逻辑捕获 sclk_out 的一个时钟边缘上的数据，并传播相反边缘上的数据。

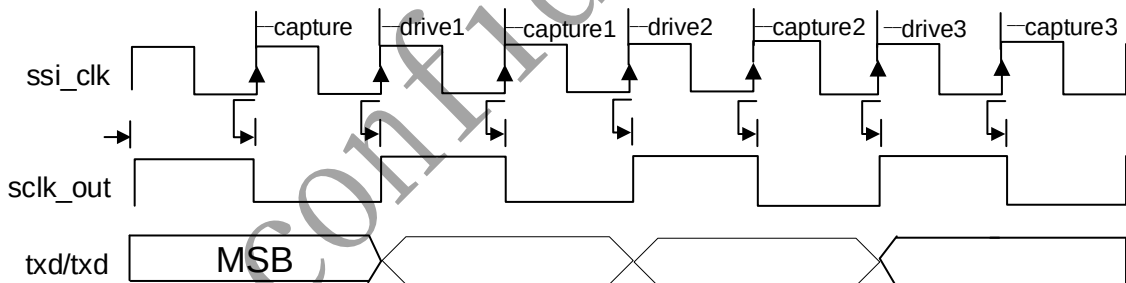


图 19-3 最大 sclk_out/ssi_clk 比率

sclk_out 线仅在进行传输过程中才切换。在其他任何时候，它都处于非活动状态，这是由它在其中操作的串行协议定义的。

sclk_out 的频率可由以下公式得到:

$$F_{sclk\_out} = \frac{F_{ssi\_clk}}{SCKDV}$$

SCKDV 是可编程寄存器 BAUDR 中的一个位字段，可以保存 0 到 65,534 之间的任何偶值。如果 SCKDV 为 0，则禁用 sclk_out。

当 SSI 被配置为从设备时，ssi_clk 的最小频率取决于从设备的操作。

1.2.2 发送/接收 FIFO 缓冲

SSI 发送和接收 FIFO 缓冲区深度是 4，FIFO 缓冲区的宽度是 32 位，串行规范规定串行传输(数据帧)的长度可以是 4 到 16/32 位。小于 32 位的数据帧在写入传输 FIFO 缓冲区时必须是对齐的。移位控制逻辑自动右对齐接收 FIFO 缓冲区中的接收数据。

FIFO 缓冲区中的每个数据条目都包含一个数据帧。不能在一个 FIFO 位置存储多个数据帧;例如，您不能在一个 FIFO 位置存储两个 8 位数据帧。如果需要 8 位数据帧，则串行移位器在传输数据时将忽略或不使用 FIFO 条目的上 8 位。

注意：当禁用 SSI (SSI_EN = 0)或重置它(presetn)时，发送和接收 FIFO 缓冲区将被清除。

发送的 FIFO 是由 APB 总线向 SSI 数据寄存器(DR)写入命令加载的。通过移位控制逻辑将数据从发射 FIFO 弹出(删除)到发射移位寄存器。当 FIFO 中的条目数小于或等于 FIFO 阈值时，传输 FIFO 将生成一个 FIFO 空中断请求(ssi_txe_intr)。通过可编程寄存器 TXFTLR 设置的阈值确定了产生中断的 FIFO 项的级别。阈值允许您向处理器提供发送 FIFO 几乎为空的早期指示。如果您试图将数据写入已经写满的 FIFO，则会生成一个传输 FIFO 溢出中断(ssi_txo_intr)。

APB 向 SSI 数据寄存器(DR)发送读命令，从接收 FIFO 中弹出数据。接收 FIFO 由移位控制逻辑从接收移位寄存器加载。当 FIFO 中的条目数大于或等于 FIFO 阈值加 1 时，接收 FIFO 将生成一个 FIFO-full 中断请求(ssi_rxf_intr)。通过可编程寄存器 RXFTLR 设置的阈值确定了产生中断的 FIFO 项的级别。

阈值允许您向处理器提供接收 FIFO 接近满的早期指示。接收 FIFO 溢出中断(ssi_rxo_intr)是在接收移位逻辑试图将数据加载到完全满的接收 FIFO 中时生成的。但是，新接收的数据丢失了。如果您试图从一个空的接收 FIFO 读取数据，则会生成一个接收 FIFO 下溢中断(ssi_rxu_intr)。这将警告处理器所读取的数据无效。

TFT Value	Description
00	当在发送 FIFO 中出现 0 个数据条目时，将产生 ssi_txe_intr
01	当在发送 FIFO 中出现 1 个或更少的数据条目时，将产生 ssi_txe_intr
10	当在发送 FIFO 中出现 2 个或更少的数据条目时，将产生 ssi_txe_intr
11	当在发送 FIFO 中出现 3 个或更少的数据条目时，将产生 ssi_txe_intr

表 19-1 发送 FIFO 阈值(TFT)解码值

RFT Value	Description
00	当在接收 FIFO 中出现 1 个或多个数据条目时，将产生 ssi_rxf_intr
01	当在接收 FIFO 中出现 2 个或多个数据条目时，将产生 ssi_rxf_intr
10	当在接收 FIFO 中出现 3 个或多个数据条目时，将产生 ssi_rxf_intr
11	当在接收 FIFO 中出现 4 个或多个数据条目时，将产生 ssi_rxf_intr

表 19-2 接收 FIFO 阈值(RFT)解码值

1.2.3 SSI 中断

SSI 支持组合的和单独的中断请求，每个中断请求都可以被屏蔽。组合中断请求是屏蔽后所有其他 SSI 中断的请求结果。所有的 SSI 中断都是电平中断，高电平有效。

SSI 中断描述如下：

- ◆ 发送 FIFO 空中断(ssi_txe_intr)——在发送 FIFO 等于或低于其阈值时被置起，需要中断服务来防止运行不足。通过软件可编程寄存器设置的阈值决定了产生中断的传输 FIFO 项的级别。当数据被写入发送 FIFO 缓冲区时，硬件会清除这个中断，使其超过阈值水平。
- ◆ 发送 FIFO 溢出中断(ssi_txo_intr) -当一个 APB 访问试图写入到已经填满的发送 FIFO 时，中断被置起。且从 APB 写入的数据被丢弃。这个中断将一直保持置起，直到您读取发送 FIFO 溢出中断清除寄存器(TXOICR)。
- ◆ 接收 FIFO 完全中断(ssi_rxf_intr)——在接收 FIFO 等于或大于其阈值加 1 时设置，需要中断服务来防止溢出。通过软件可编程寄存器设置的阈值决定了接收 FIFO 项的级别，在这个级别上产生中断。当从接收 FIFO 缓冲区读取数据时，硬件将清除这个中断，使其低于阈值水平。
- ◆ 接收 FIFO 溢出中断(ssi_rxo_intr) -当接收逻辑试图将数据完全填满后放入接收 FIFO 时设置。设置时，将丢弃新接收的数据。这个中断保持设置，直到你读取接收 FIFO 溢出中断清除寄存器(RXOICR)。
- ◆ 接收 FIFO 下溢中断(ssi_rxu_intr)——当 APB 尝试从空的接收 FIFO 中读取时，中断置起。而从接收 FIFO 中读取值为 0。这个中断保持设置，直到你读取接收 FIFO 下溢中断清除寄存器(RXUICR)。
- ◆ 多主争用中断(ssi_mst_intr)——只有在将 SSI 组件配置为串行主设备时才会出现。当串行总线上的另一个串行主机选择 SSI 主机作为串行从设备并正在主动传输数据时，中断将置起。这将通知处理器串行总线上可能出现的争用。在读取多主中断清除寄存器(MSTICR)之前，此中断将保持设置。
- ◆ 组合中断请求(ssi_intr)——或屏蔽后所有上述中断请求的结果。要屏蔽这个中断信号，必须屏蔽所有其他 SSI 中断请求。

1.2.4 传输模式

在串行总线上传输数据时，传输模式(TMOD)是通过写入控制寄存器 0 (SSI_CR0)来设置的。

注意:传输模式设置不影响串行双工传输。对于由 MWCR 寄存器控制的微波传输，忽略 TMOD。

1.2.4.1 发送和接收

当 TMOD = 2'b00 时，发送和接收逻辑都是有效的。数据传输按照选定的帧格式(串行协议)正常进行。发送数据从发送 FIFO 中弹出，并通过 txd 数据线发送到目标设备，目标设备使用 rxd 数据线上的数据进行应答。来自目标设备的接收数据在每个数据帧结束时从接收移位寄存器移动到接收 FIFO。

1. 2. 4. 2 仅发送

当 $TMOD = 2'b01$ 时, 接收数据无效, 不应存储在接收 FIFO 中。数据传输按照选定的帧格式(串行协议)正常进行。发送数据从发送 FIFO 中弹出, 并通过 txd 数据线发送到目标设备, 目标设备使用 rxd 数据线上的数据进行应答。在数据帧结束时, 接收移位寄存器不会将其新接收的数据加载到接收 FIFO 中。接收移位寄存器中的数据被下一次传输覆盖。在输入此模式时, 应该屏蔽来自接收逻辑的中断。

1. 2. 4. 3 仅接收

当 $TMOD = 2'b10$ 时, 发送的数据无效。当配置为从站时, 发送 FIFO 不会在只接收模式中弹出。在传输期间, txd 输出保持在恒定的逻辑级别。数据传输按照选定的帧格式(串行协议)正常进行。来自目标设备的接收数据在每个数据帧结束时从接收移位寄存器移动到接收 FIFO。在输入此模式时, 应该屏蔽来自传输逻辑的中断。

1. 2. 4. 4 读 EEPROM

注意:此传输模式仅对主配置有效。

当 $TMOD = 2'b11$ 时, 传输数据用于向 EEPROM 设备传输操作码和/或地址。通常这需要三个数据帧(8 位操作码, 然后是 8 位上地址和 8 位下地址)。在操作码和地址的传输过程中, 接收逻辑不会捕获任何数据(只要 SSI 主机在它的 txd 行上传输数据, rxd 数据线上的数据就会被忽略)。SSI 主进程继续传输数据, 直到传输 FIFO 为空。因此, 在传输 FIFO 时, 您应该只有足够的数据帧来为 EEPROM 提供操作码和地址。如果传输的 FIFO 中的数据帧比需要的多, 那么读数据就会丢失。

当发送的 FIFO 变为空(所有控制信息都已发送)时, 接收数据线(rxd)上的数据有效, 并存储在接收 FIFO 中;txd 输出保持在恒定的逻辑级别。串行传输一直持续到 SSI 主接收的数据帧数与 CTRLR1 寄存器+ 1 中的 NDF 字段值匹配为止。

注意:当将 SSI 配置为 SSP 模式时, 不支持 EEPROM 读模式。

1. 2. 5 操作模式

1. 2. 5. 1 串行主设备模式

当配置为串行主设备时, SSI 启动并控制所有串行传输。

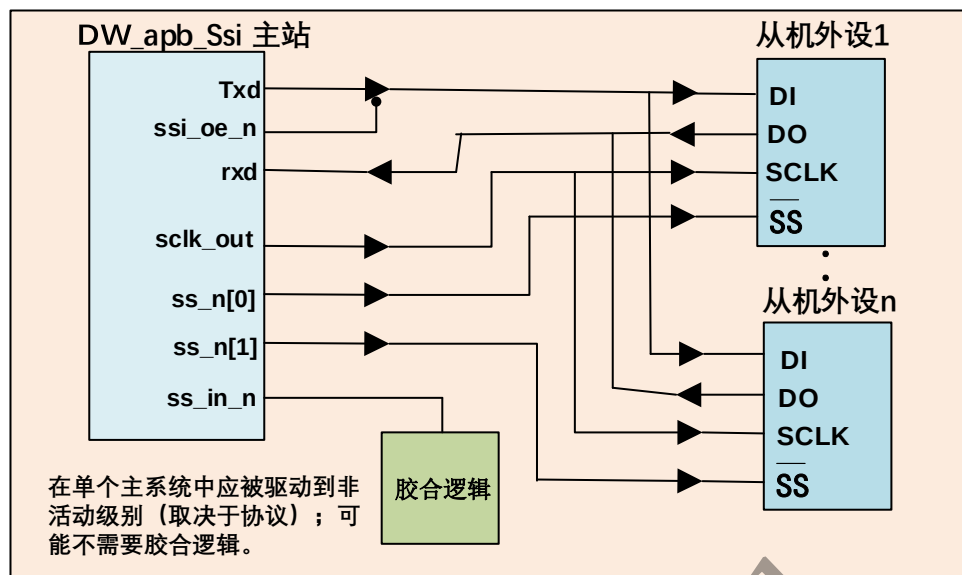


图 19-4 SSI 配置为主设备

由 SSI 生成并控制的串行比特率时钟在 `sclk_out` 行上被推出。当禁用 SSI (`SSI_EN = 0`) 时，不会发生串行传输，并且 `sclk_out` 处于“非活动”状态，这是由串行协议定义的，它在串行协议下运行。

RXD 采样延时

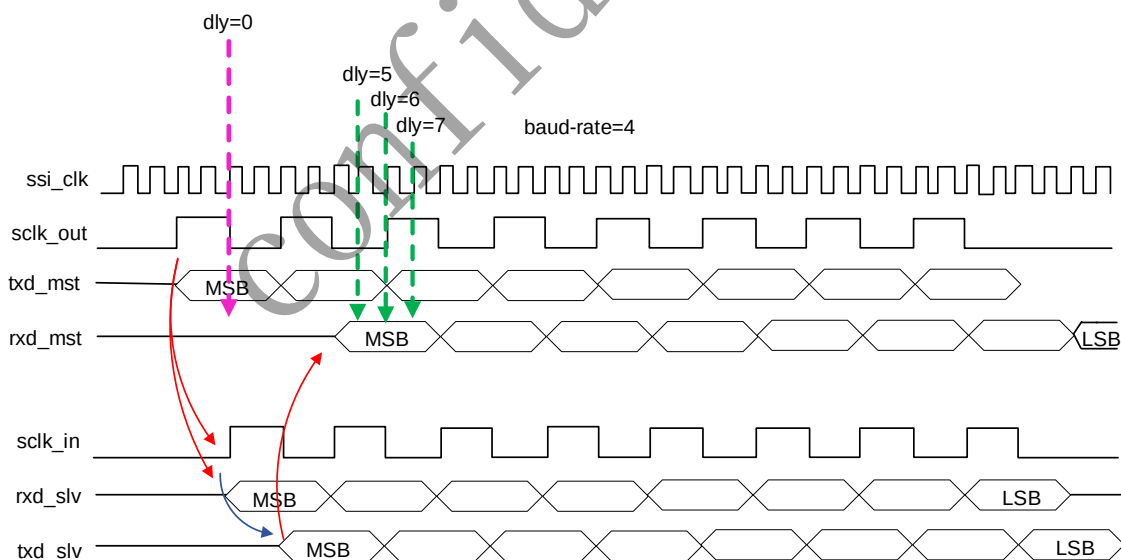


图 19-5 `sclk_out` 信号绕线延迟的影响

红色箭头表示主从设备之间的绕线延迟

蓝色箭头表示采样延时从接收 `sclk_in` 到驱动 `txd` 输出

从机使用来自主机的 `sclk_out` 信号作为时钟，以便将 `rxd` 信号数据驱动到串行总线上。从设备

对 `sclk_out` 信号进行路由和采样延迟意味着在主设备对 `rx` 信号进行采样之前, `rx` 位没有稳定到正确的值。

数据传输

数据传输由串行主设备启动。当启用 SSI 时(`SSI_EN=1`), 在传输 FIFO 中至少有一个有效的数据条目, 并且选择了一个串行从设备。在主动传输数据时, 将在状态寄存器(SR)中设置 `busy` 标志(`busy`)。在尝试新的串行传输之前, 必须等待 `busy` 标志被清除。

注意:当数据写入传输 FIFO 时, 不设置忙状态。这个位只有在目标从设备被选择并且传送正在进行时才会被设定。将数据写入传输 FIFO 后, 移位逻辑直到 `sclk_out` 信号的正边缘出现时才开始串行传输。等待这个正边缘的延迟取决于串行传输的波特率。在轮询繁忙状态之前, 您应该首先轮询 TFE 状态(等待 1)或等待 $\text{BAUDR} * \text{ssi\_clk}$ 时钟周期。

主设备的 SPI 和 SSP 串行传输

当传输模式为“发送和接收”或“仅传输”(TMOD = 2'b00 或 TMOD = 2'b01, 分别)时, 当传输 FIFO 为空时, 由移位控制逻辑终止传输。对于连续的数据传输, 必须确保传输 FIFO 缓冲区在所有数据传输之前不会变为空。发送 FIFO 阈值水平(TXFTLR)可用于提前中断(`ssi_txe_intr`)处理器, 指示发送 FIFO 缓冲区几乎为空。当 DMA 用于 APB 访问时, 传输数据层(DMATDLR)可用于早期请求(`dma_tx_req`) DMA 控制器, 这表明传输 FIFO 几乎是空的。然后可以用数据来填充 FIFO 以继续串行传输。在启用串行从设备之前, 用户还可以将数据块(至少两个 FIFO 条目)写入传输 FIFO。这确保串行传输直到组成连续传输的数据帧数在传输 FIFO 中出现时才开始。

当传输模式为“仅接收”(TMOD = 2'b10)时, 当选择一个串行从设备时, 通过向传输 FIFO 写入一个“虚拟”数据字开始串行传输。在串行传输期间, SSI 的 `tx` 输出保持在恒定的逻辑级别。发送 FIFO 在开始时只弹出一次, 并且在串行传输期间可能一直为空。串行传输的结束由控制寄存器 1 (CTRLR1)中的“数据帧数”(NDF)字段控制。

例如, 如果您希望从一个串行从设备接收 24 个数据帧, 您应该用值 23 来编写 NDF 字段;当接收的帧数等于 NDF 值+ 1 时, 接收逻辑终止串行传输。这种传输模式增加了 APB 总线的带宽, 因为传输 FIFO 在传输期间不需要处理。每次接收 FIFO 产生一个 FIFO 完全中断请求时, 都应该读取接收 FIFO 缓冲区, 以防止溢出。

当传输模式为“读 eeprom”(TMOD = 2'b11)时, 当选择一个串行从设备(EEPROM)时, 通过将操作码和/或地址写入传输 FIFO 来启动串行传输。操作码和地址被传输到 EEPROM 设备, 然后从 EEPROM 设备接收读数据并存储在接收 FIFO 中。串行传输的结束由控制寄存器 1 (CTRLR1)中的 NDF 字段控制。

注意:当将 SSI 配置为 SSP 模式时, 不支持 EEPROM 读模式。

接收 FIFO 阈值水平(RXFTLR)可用于早期指示接收 FIFO 已接近满。当 aDMAis 用于 APB 访问时, 接收数据层(DMARDLR)可用于 DMA 控制器的早期请求(`dma_rx_req`), 这表示接收 FIFO 已接近满。

从 SSI 串行主机完成 SPI 或 SSP 串行传输的典型软件流程如下:

1. 如果启用了 SSI, 则通过向 SSI Enable 寄存器(SSIENR)写入 0 来禁用它。

2. 设置 SSI 控制寄存器进行传输;这些寄存器可以按任何顺序设置。

- ◆ 写控制寄存器 0 (SSI_CR0)。对于 SPI 传输，串行时钟极性和串行时钟相位参数必须设置与目标从设备相同。
- ◆ 如果传输模式仅接收，则将 CTRLR1(控制寄存器 1)写入传输中的帧数减 1;例如，如果您想接收 4 个数据帧，那么用 3 来写这个寄存器。
- ◆ 写入波特率选择寄存器(BAUDR)来设置传输的波特率。
- ◆ 编写发送和接收 FIFO 门限电平寄存器(分别为 TXFTLR 和 RXFTLR)来设置 FIFO 门限电平
- ◆ 写 IMR 寄存器来设置中断屏蔽。
- ◆ 设置从设备使能寄存器(SER)寄存器选择的目标从设备。如果在这里启用了从设备，那么在发送 FIFO 中出现一个有效的数据条目时，传输就开始了。如果在写入数据寄存器(DR)之前没有启用从设备，则在启用从设备之前不会开始发送。

3. 通过向 SSIENR 寄存器写入 1 来启用 SSI。

4. 将传输到目标从设备的数据写入到发送 FIFO 中(写入 DR)。如果此时在 SER 寄存器中没有启用从设备，那么现在就可以开始传输了

5. 轮询忙状态以等待传输完成。如果发送一个 FIFO 空中断请求，写发送 FIFO(写 DR)。如果发出了接收 FIFO 完全中断请求，则读取接收 FIFO(读 DR)。

6. 当发送 FIFO 为空时，移位控制逻辑将停止传输。如果传输模式仅接收(TMOD = 2'b10)，则当接收到指定数目的帧时，移位控制逻辑将停止传输。当传输完成时，繁忙状态将重置为 0。

7. 如果传输模式不是只传输(TMOD != 01)，则读取接收 FIFO，直到它为空。

8. 通过将 0 写入 SSIENR 来禁用 SSI。

下图显示了启动 SSI 主 SPI/SSP 串行传输的典型软件流程。该图还显示了串行主组件内部的硬件流。

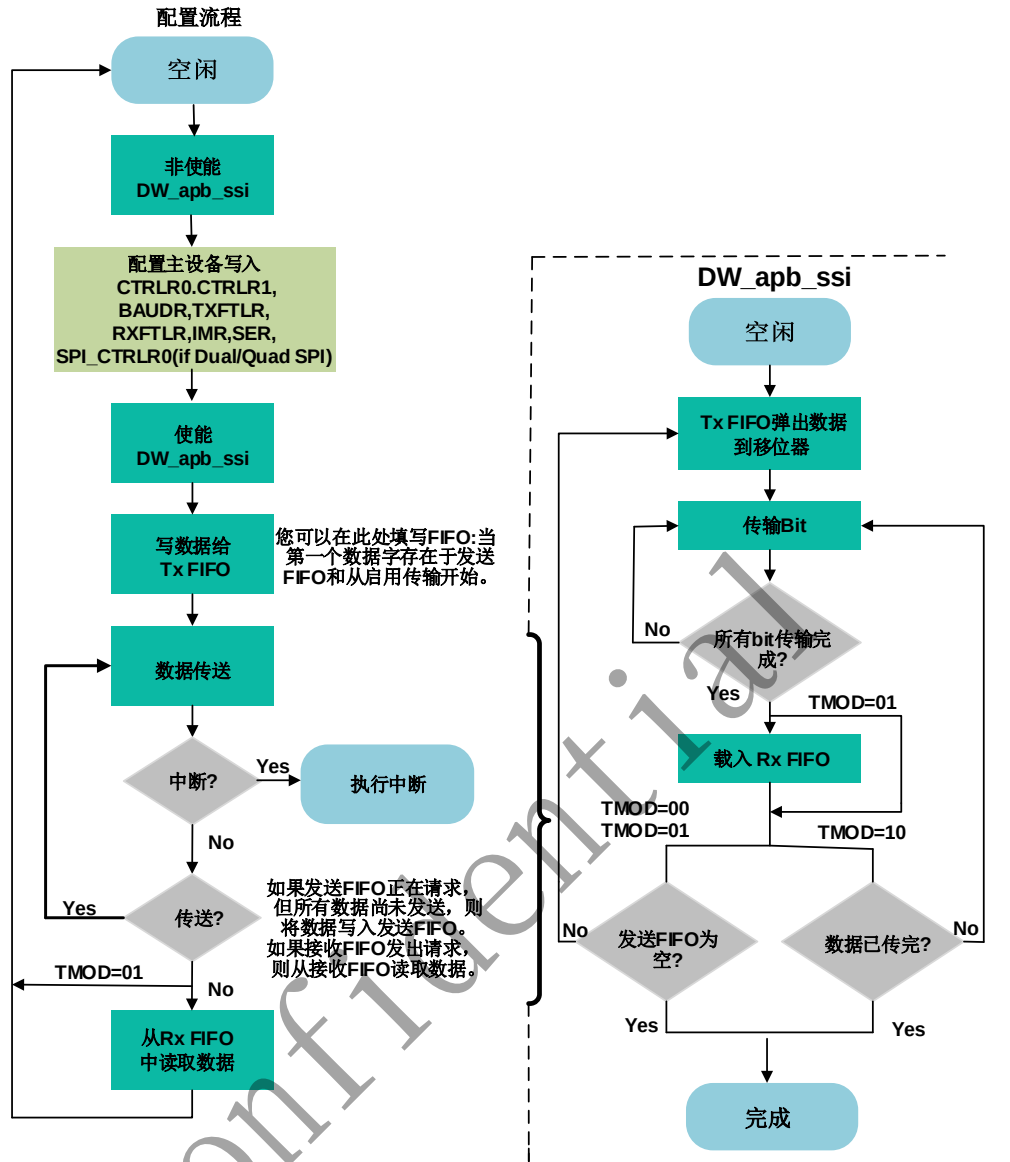


图 19-6 SSI 主设备 SPI/SSP 传输流程

主设备微线串行传输

来自 SSI 串行主机的微线串行传输由微线控制寄存器(MWCR)控制。MWHS Bit 位启用和禁用微线握手接口。MDD Bit 位控制数据帧的方向(控制帧总是由主发送, 从接收)。MWMOD Bit 位定义传输是顺序的还是非顺序的。

当传送 FIFO 中至少有一个控制字并且启用了—个从设备时, SSI 串行主设备将启动所有微线传输。当 SSI 主设备传输数据帧(MDD = 1)时, 当发送 FIFO 为空时, 移位逻辑终止传输。当 SSI 主设备接收到数据帧(MDD = 1)时, 传输的终止取决于 MWMOD 位字段的设置。如果传输是不连续的(MWMOD = 0), 则当发送 FIFO 为空时, 从从属节点转移数据帧后, 传输将终止。当传输是连续的(MWMOD = 1)时, 当接收到的数据帧数等于 CTRLR1 寄存器+ 1 中的值时, 通过移位逻辑终止传输。

当启用 SSI 主服务器上的握手接口(MWHS =1)时, 将在传输之后轮询目标从设备的状态。只

有当从属设备报告就绪状态时，SSI 主设备才能完成传输并清除其忙碌状态。如果传输是连续的，直到从设备返回就绪状态才会发送下一个控制/数据帧。

从 SSI 串行主设备完成微波串行传输的典型软件流程如下：

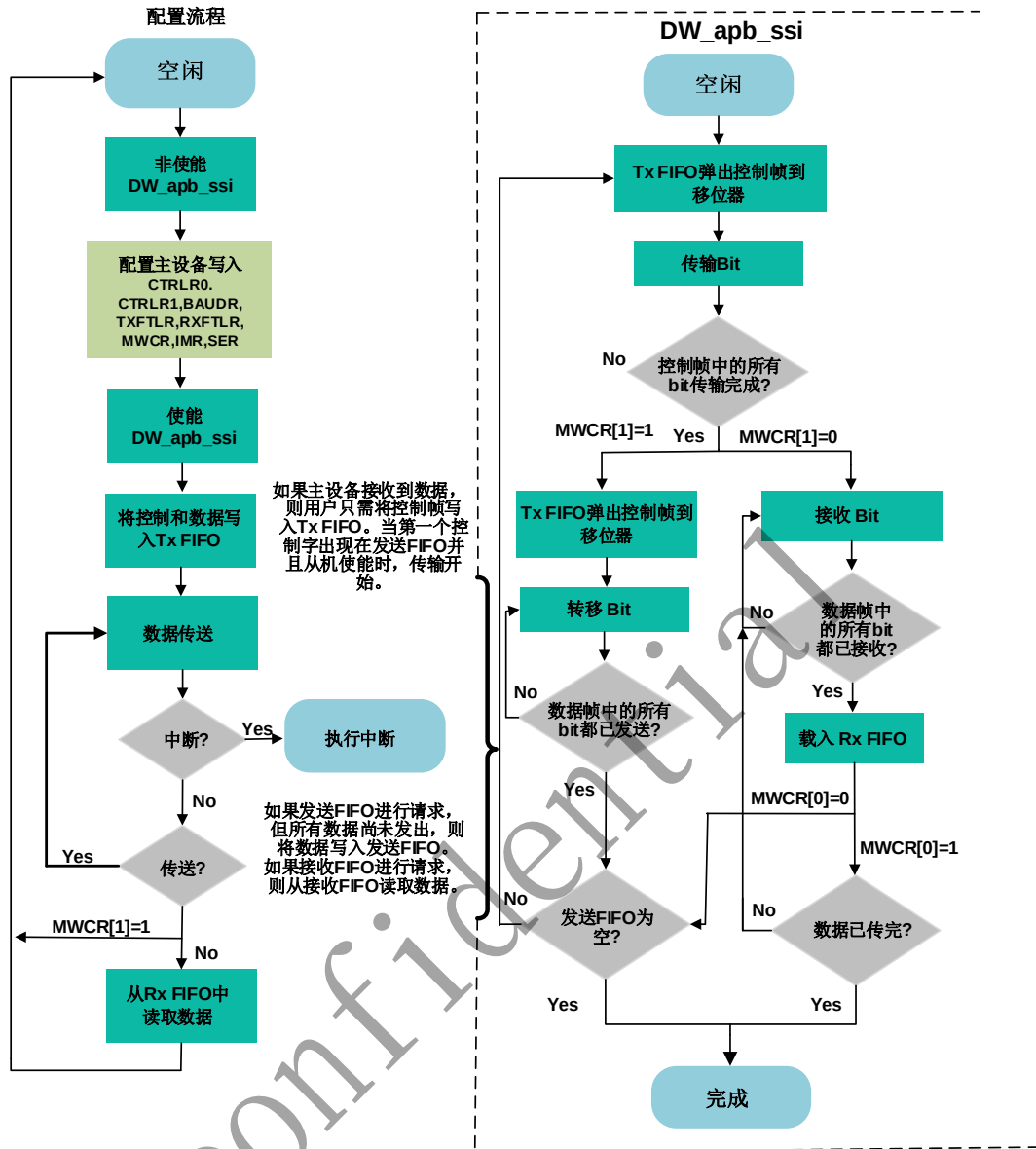
1. 如果启用了 SSI，则通过将 0 写入 SSIENR 来禁用它。
2. 设置 SSI 控制寄存器用于传输。这些寄存器可以按任何顺序设置。写入 SSI_CR0 设置传输参数。
 - ◆ 如果传输是连续的，并且 SSI 主机接收到数据，则使用传输中的帧数减去 1 来编写 CTRLR1；例如，如果您想要接收 4 个数据帧，那么用 3 来编写这个寄存器。
 - ◆ 编写 BAUDR 来设置传输的波特率。
 - ◆ 编写 TXFTLR 和 RXFTLR 来设置 FIFO 阈值水平。
 - ◆ 写 IMR 寄存器来设置中断屏蔽。

您可以编写 SER 寄存器来启用目标从设备进行选择。如果在这里启用了从设备，那么在发送 FIFO 中出现一个有效的数据条目时，传输就开始了。如果在写入 DR 寄存器之前没有启用从设备，则在启用从设备之前不会开始传输。

3. 通过向 SSIENR 寄存器写入 1 来启用 SSI。
4. 如果 SSI 主设备传输数据，则将控制和数据字写入发送 FIFO(写 DR)。如果 SSI 主设备接收到数据，将控制字写入到发送 FIFO 中。如果此时在 SER 寄存器中没有启用从设备，那么现在就可以开始传输了。
5. 轮询忙状态以等待传输完成。

如果发送一个 FIFO 空中断请求，写发送 FIFO(写 DR)。如果发出了接收 FIFO 完全中断请求，则读取接收 FIFO(读 DR)。
6. 当发送 FIFO 为空时，移位控制逻辑将停止传输。如果传输模式是连续的，并且 SSI 主设备接收数据，那么当接收到指定数量的数据帧时，移位控制逻辑将停止传输。当传输完成时，繁忙状态将重置为 0。
7. 如果 SSI 主设备接收数据，则读取接收 FIFO，直到它为空。
8. 通过将 0 写入 SSIENR 来禁用 SSI。

下图显示了启动 SSI 主设备微线串行传输的典型软件流程。该图还显示了串行主组件内部的硬件流。



1.2.5.2 串行从设备模式

此模式允许与主外围设备进行串行通信。当将 SSI 配置为从设备时，所有串行传输都由串行总线主设备发起和控制。下图显示了在单主设备总线系统中作为串行从设备配置的 SSI 示例。

当在配置过程中选择 SSI 串行从设备时，它将其 txd 数据启用到串行总线上。所有与串行从设备之间的数据传输都在串行时钟线(sclk_in)上进行调节，由串行主设备驱动。数据从串行时钟线的一端串行从设备传输到另一端进行采样。

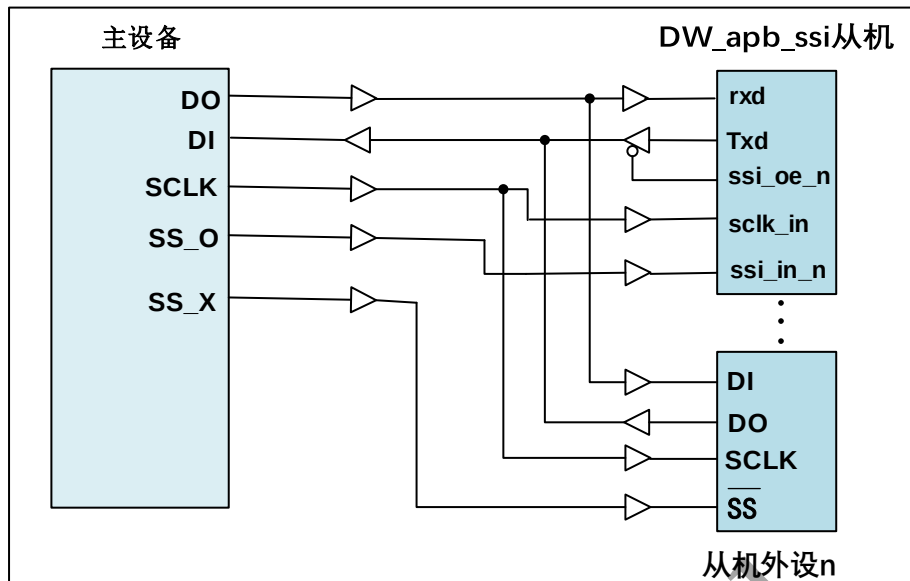


图 19-8 SSI 配置为从设备

当 SSI 没有被选择位串行从设备时，它不能干扰串行主设备和其他串行从设备之间的数据传输。当串行从设备没有被选择时，它的 txd 输出被缓冲，导致高阻抗驱动到串行主设备 rxd 线路。

控制数据传输的串行时钟由串行主设备生成，并输入到 SSI 从设备的 sclk_in 上。在被总线主人选择之前，从设备保持在空闲状态。当不主动传输数据时，从设备必须将它的 txd 线保持在高阻抗状态，以避免对从设备的串行传输产生干扰。可以使用 ssi_oe_n 行来控制 txd 输出缓冲。只要选择了主设备，从设备就会继续在主设备之间来回传输数据。如果主设备传输给所有的串行从设备，一个控制位(SLV_OE)在 SSI 控制寄存器 0 (SSI_CR0)可以编程通知从设备，如果它应该响应来自它的 txd 线路数据。

从设备的 SPI 和 SSP 串行传输

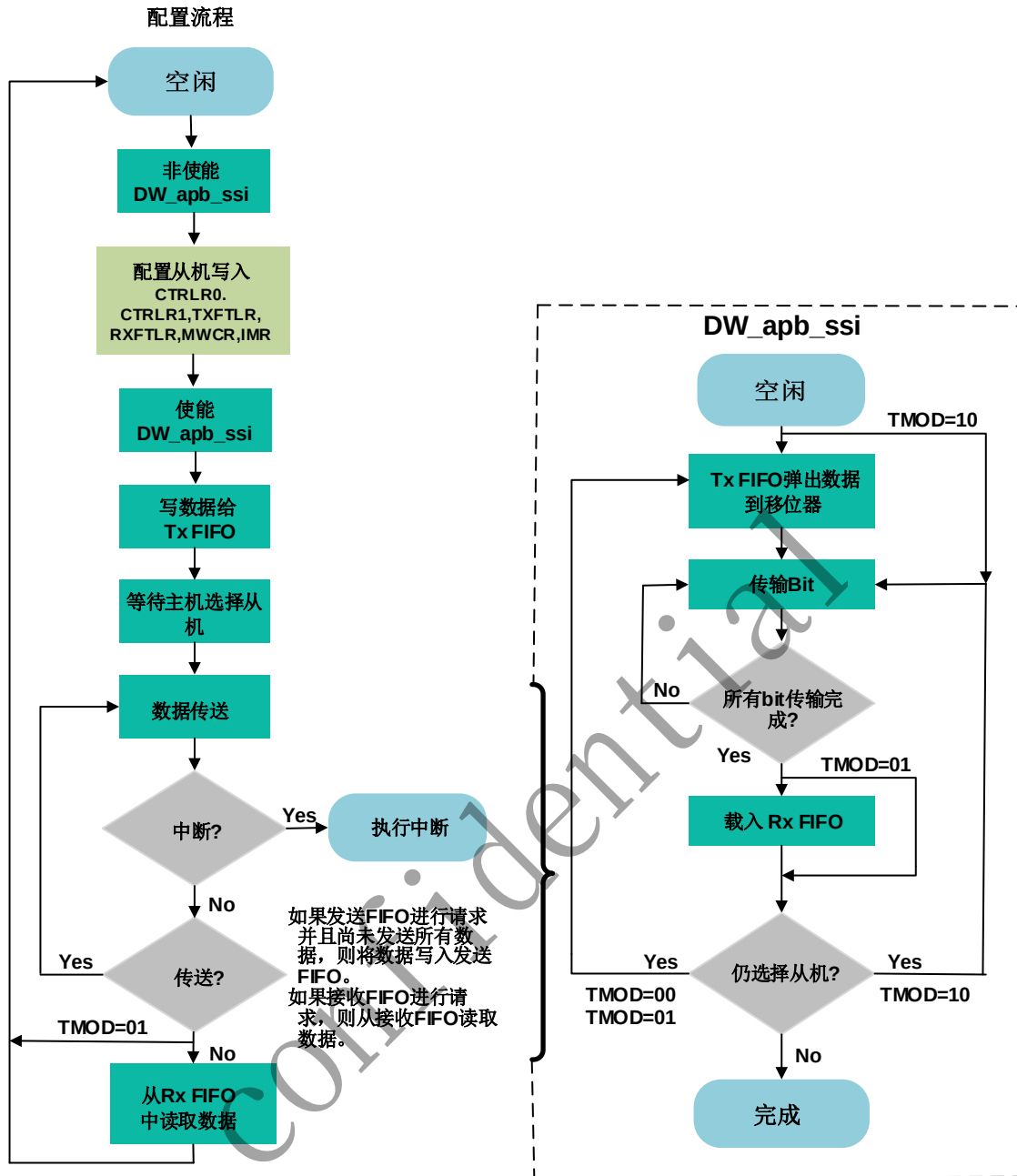
如果 SSI 从设备只接收(TMOD=10)，则发送 FIFO 不需要包含有效数据，因为每次选择从设备时，发送移位寄存器中当前的数据都被重新发送。当 TMOD=01 时，状态寄存器(SR)中的 TXE 错误标志没有设置。当使用此模式时，您应该屏蔽传输 FIFO 空中断。

如果 SSI 从设备向主设备传输数据，在串行主设备启动传输之前，必须确保数据存在于发送 FIFO 中。如果在发送 FIFO 中没有数据存在时，主设备向 SSI 从设备发起数据传输，则在 SSI 状态寄存器中设置错误标志(TXE)，并在 txd 上重新发送之前传输的数据帧。对于连续的数据传输，必须确保发送 FIFO 缓冲区在所有数据传输之前不会变为空。发送 FIFO 门限电平寄存器(TXFTLR)可用于早期中断(ssi_txe_intr)处理，表明发送 FIFO 缓冲区几乎是空的。当 DMA 控制器用于 APB 访问时，DMA 传输数据级寄存器(DMATDLR)可用于 DMA 控制器的早期请求(dma_tx_req)，这表明发送 FIFO 几乎是空的。然后可以用数据来填充 FIFO 以继续串行传输。每次接收 FIFO 产生一个 FIFO 完全中断请求时，都应该读取接收 FIFO 缓冲区，以防止溢出。接收 FIFO 阈值水平寄存器(RXFTLR)可用于早期指示接收 FIFO 已接近满。当 DMA 控制器用于 APB 访问时，DMA 接收数据级寄存器(DMARDLR)可用于 DMA 控制器的早期请求(dma_rx_req)，这表示接收 FIFO 已接近满。

完成从串行主设备到 SSI 从设备的连续串行传输的典型软件流程如下:

1. 如果启用了 SSI, 则通过将 0 写入 SSIENR 来禁用它。
2. 设置 SSI 控制寄存器用于传输。这些寄存器可以按任何顺序设置。
 - a. 写入 SSI_CR0(对于 SPI 传输, SCPH 和 SCPOL 必须设置为与主设备相同)。
 - b. 编写 TXFTLR 和 RXFTLR 来设置 FIFO 阈值水平。
 - c. 写 IMR 寄存器来设置中断屏蔽。
3. 通过向 SSIENR 寄存器写入 1 来启用 SSI。
4. 如果传输模式是发送和接收(TMOD=2'b00)或只传输(TMOD=2'b01), 则将要传输给主设备的数据写入到发送 FIFO (write DR)中。如果只接收传输模式(TMOD=2'b10), 则不需要将数据写入发送 FIFO;传输移位寄存器中的当前值被重新传输。
5. SSI 从设备现在可以进行串行传输了。当串行主设备选择 SSI 从设备时, 传输就开始了。
6. 当传输正在进行时, 可以轮询忙碌状态以返回传输状态。如果发送一个 FIFO 空中断请求, 写发送 FIFO(写 DR)。如果发出了接收 FIFO 完全中断请求, 则读取接收 FIFO(读 DR)。
7. 当串行主设备将 SSI 从设备移除时, 传输结束。当传输完成时, 繁忙状态将重置为 0。
8. 如果传输模式不是只传输(TMOD != 01), 读取接收 FIFO 直到为空。
9. 通过将 0 写入 SSIENR 来禁用 SSI。

下图显示了 SSI 从设备的 SPI 或 SSP 串行传输的典型软件流。该图还显示了从串行组件内部的硬件流。



从设备微线串行传输

当将 SSI 配置为从设备时，微线协议的操作方式与 SPI 协议非常相似。SSI 从设备对控制帧没有解码

1.2.6 接口协议

1. 2. 6. 1 摩托罗拉串行外围接口协议(SPI)

在 SPI 中, 时钟极性(SCPOL)配置参数决定串行时钟的非活动状态是高还是低。要传输数据, 两个 SPI 外围设备必须具有相同的串行时钟相位(SCPH)和时钟极性(SCPOL)值。

当配置参数 SCPH = 0 时, 从选择信号下降沿开始数据传输。所述第一数据位由所述串行时钟的第一边缘上的主从外围设备捕获; 因此, 有效的数据必须在第一个串行时钟边缘之前出现在 txd 和 rxd 行上。

下图显示了单个 SPI 数据传输的时间图, SCPH = 0。配置参数 SCPOL = 0 和 SCPOL = 1 显示的是串行时钟。

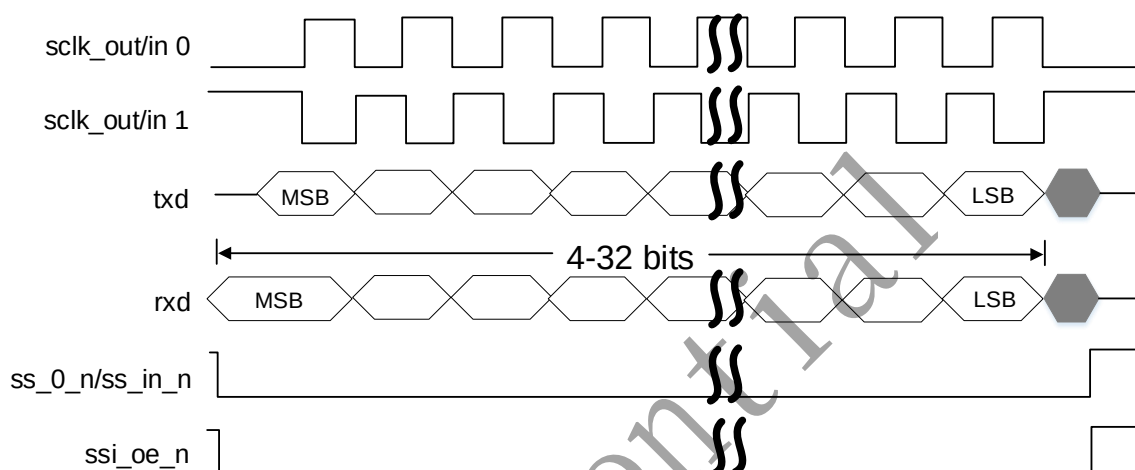


图 19-10 SPI 串行格式(SCPH = 0)

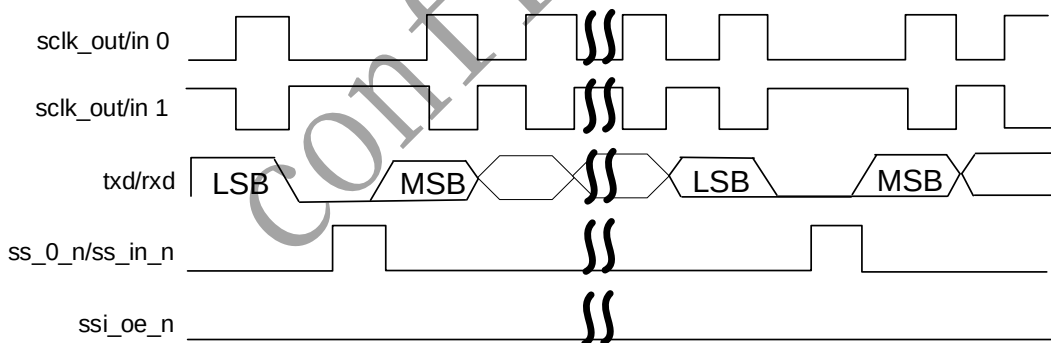


图 19-11 串行格式连续传输 (SCPH = 0)

当配置参数 SCPH = 1 时, 主从外设都在从选择行激活后的第一个串行时钟边缘上开始传输数据。第一个数据位捕获在第二个(拖尾)串行时钟边缘上。数据由串行时钟前缘上的主外设和从外设传播。在连续的数据帧传输期间, 从选择行可能保持低活动状态, 直到捕获最后一帧的最后一位。

下图显示了配置参数 SCPH = 1 时 SPI 格式的时序图。

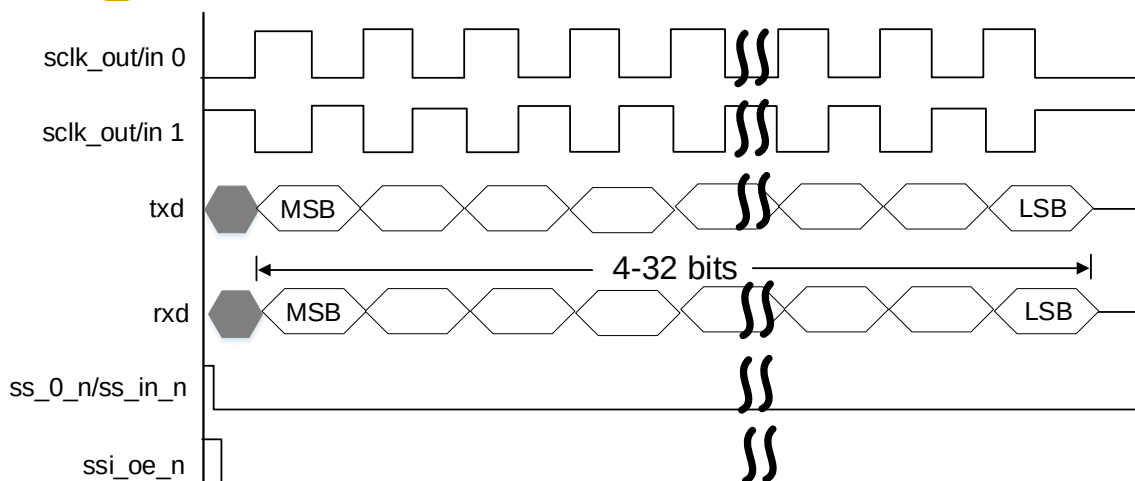


图 19-12 SPI 串行格式 (SCPH = 1)

连续数据帧以与单帧相同的方式传输，下一帧的 MSB 紧随当前帧的 LSB 之后。从选择信号在传输期间保持活动状态。

下图显示了配置参数 SCPH = 1 时连续 SPI 传输的时序图。

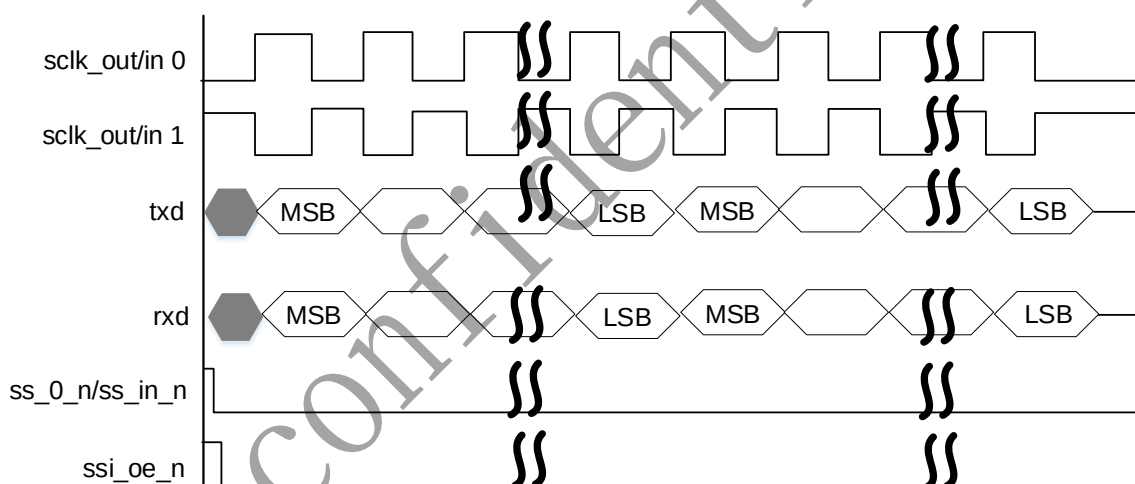


图 19-13 SPI 连续传输串行格式(SCPH = 1)

在 SSI 上执行 SPI 串行通信有四种可能的传输模式。对于发送和接收传输(控制寄存器 0 = 2'b00 的传输模式字段(9:8))，从 SSI 传输到外部串行设备的数据被写入到发送 FIFO 中。从外部串行设备接收到的数据被推送到 SSI 的接收 FIFO 中。

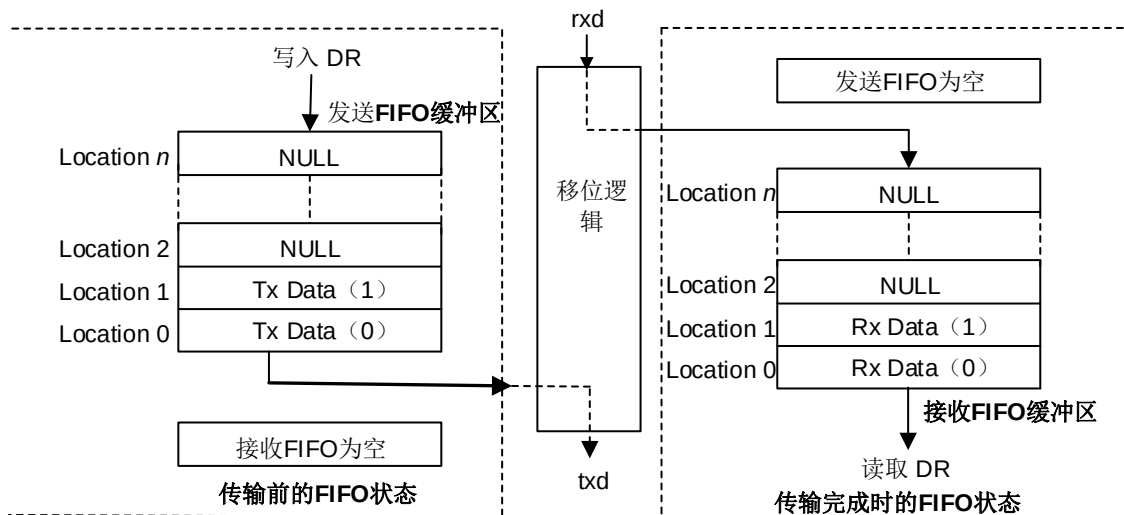


图 19-14 发送和接收的 SPI 和 SSP 传输的 FIFO 状态

对于仅用于传输的传输(控制寄存器 0 = 2'b01 的传输模式字段(9:8)), 从 SSI 传输到外部串行设备的数据被写入到发送 FIFO 中。由于从外部串行设备接收到的数据被认为是无效的, 所以它不存储在 SSI 接收 FIFO 中。

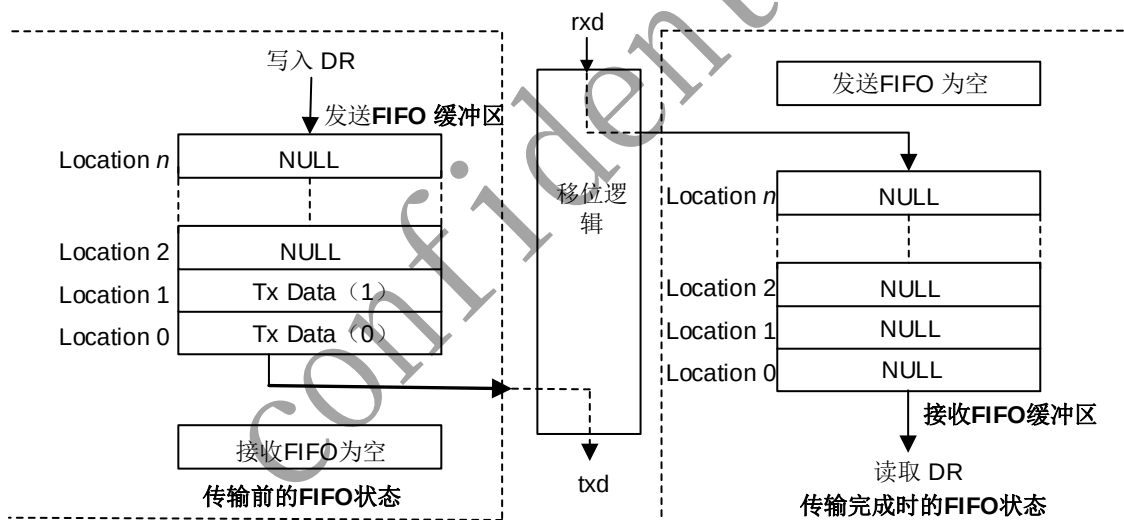


图 19-15 只发送的 SPI 和 SSP 传输的 FIFO 状态

对于只接收传输(控制寄存器 0 = 2'b10 的传输模式字段(9:8)), 从 SSI 传输到外部串行设备的数据无效, 因此在发送 FIFO 中写入一个虚拟字开始串行传输。在串行传输期间, SSI 的 txd 输出保持在恒定的逻辑级别。从外部串行设备接收到的数据被推送到 SSI 的接收 FIFO 中。

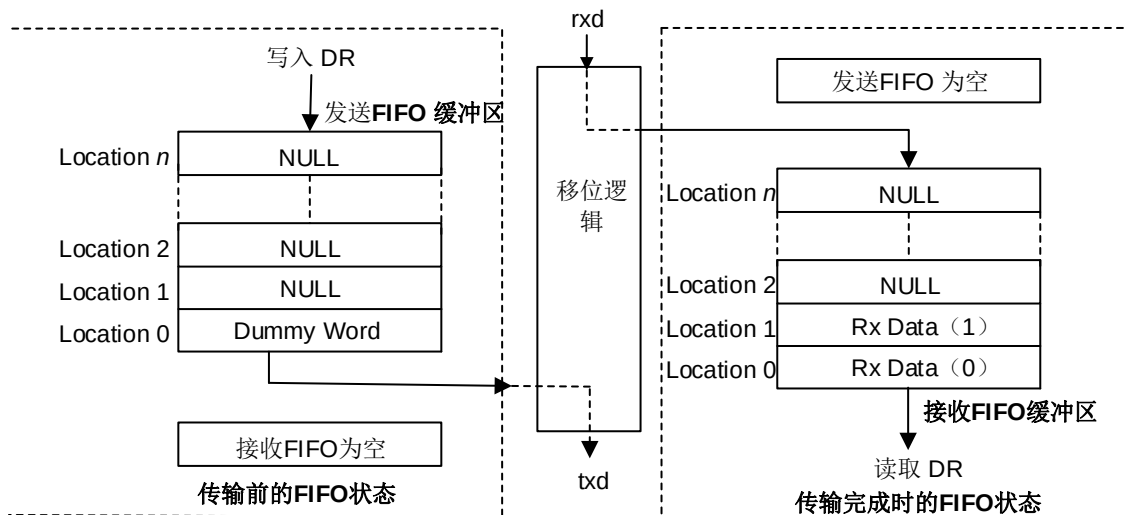


图 19-16 只接收的 SPI 和 SSP 传输的 FIFO 状态

对于 eeprom_read 传输(控制寄存器 0 = 2'b11 的传输模式字段[9:8]), 将操作码、EEPROM 地址写入发送 FIFO。在传输这些控制帧的过程中, SSI 主机不会捕获接收到的数据。在控制帧被传输之后, 来自 EEPROM 的接收数据被存储在接收 FIFO 中。

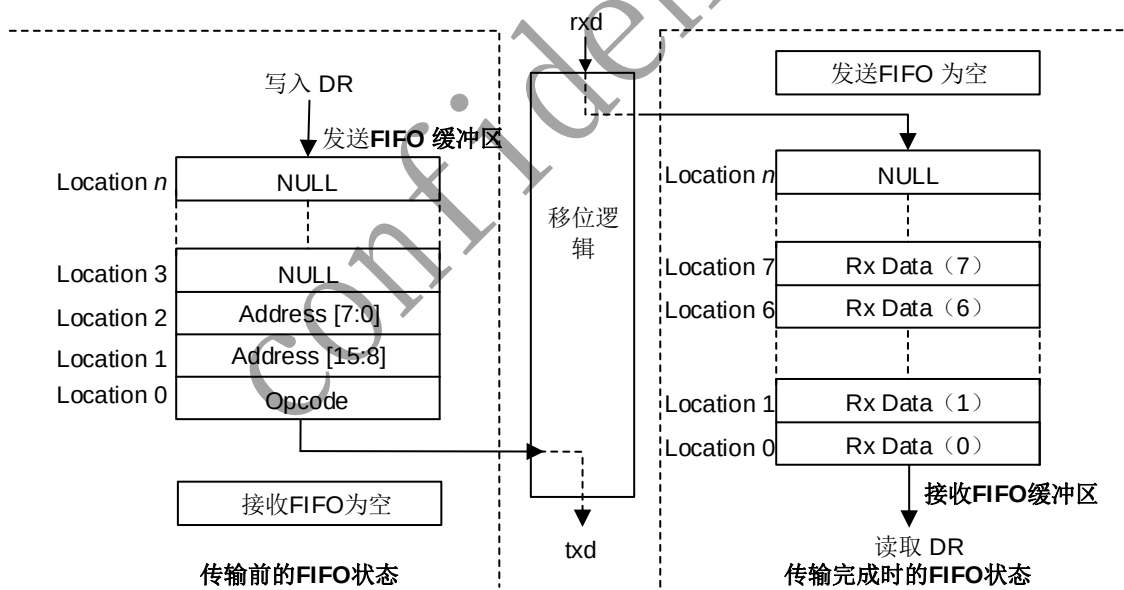


图 19-17 EEPROM 读传输模式的 FIFO 状态

1. 2. 6. 2 德州仪器同步串行协议(SSP)

数据传输首先对一个串行时钟周期判断帧指示行(ss_0_n/ss_in_n)。待传输的数据在一个串行时钟周期后驱动到 tx 行;类似地, 来自从设备的数据被驱动到 rx 行。数据在串行时钟的上升沿(sclk_out/sclk_in)传播, 并在下降沿捕获。

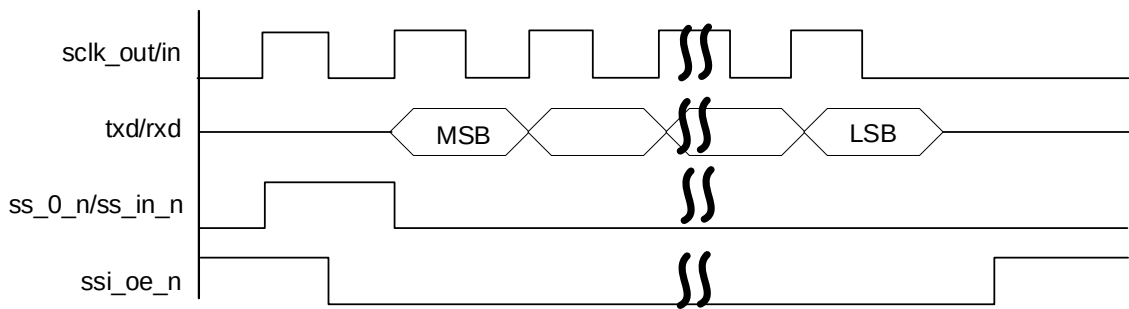


图 19-18 SSP 串行格式

连续数据帧以与单个数据帧相同的方式传输。帧指示器在与当前传输的 LSB 相同的周期内对一个时钟周期进行判断，表示随后将执行另一个数据帧。

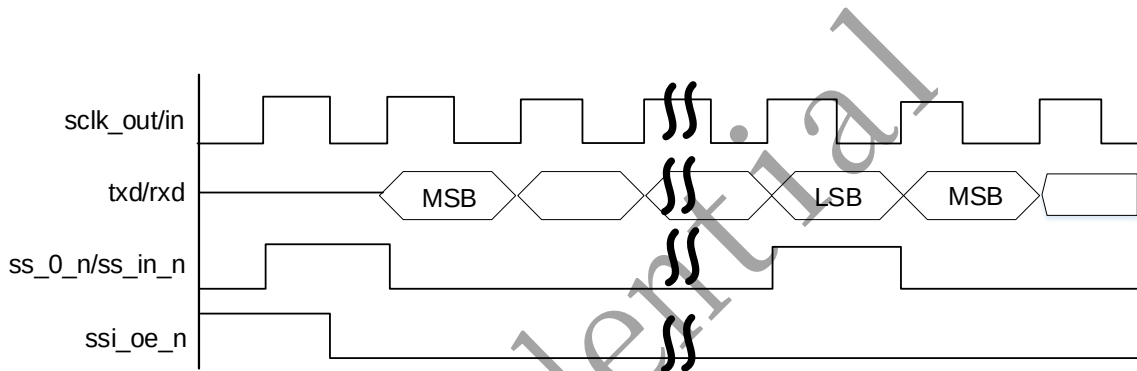


图 19-19 连续 SSP 传输串行格式

1. 2. 6. 3 国家半导体微线协议

当 SSI 被配置为串行主设备时，数据传输从从设备选择信号(ss_0_n)的下降沿开始。半串行时钟(sclk_out)周期过后，控制的第一个位在 txd 行上发送。控制字的长度可以在 1 到 16 位之间，通过在 SSI_CR0 中写入位字段 CFS (bits 15:12)来设置。控制字的其余部分由 SSI 串行主设备传输(在 sclk_out 的下降沿传播)。在这个传输过程中，串行主机的 rxd 线路上没有数据(高阻抗)。

数据字的方向由微线控制寄存器(MWCR)中的 MDD Bit 位(第 1 位)控制。当 MDD=0 时，这表示 SSI 串行主设备从外部串行从设备接收数据。在控制字的 LSB 传输后的一个时钟周期内，从外设用一个虚拟的 0 位响应，然后是数据帧，长度是 32 位。数据在串行时钟的下降沿传播，并在上升沿捕获。

从设备选择信号在传输期间保持低激活状态，并在数据传输后的一个半时钟周期内取消判断。下图显示了 SSI 主机微线单次从外部串行从设备传输的时序图。

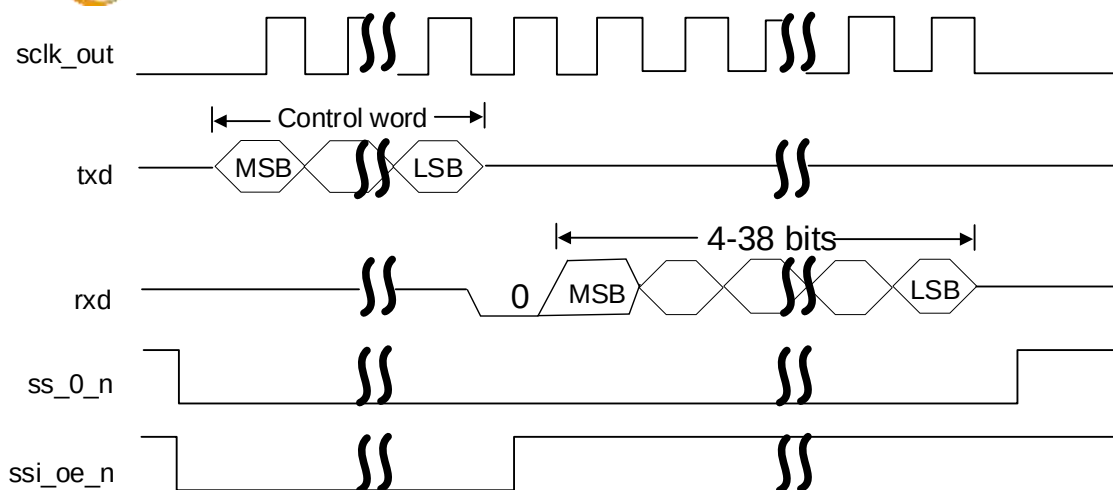


图 19-20 SSI 主机微线单词串行传输(MDD=0)

下图展示了数据和控制帧在传输之前的 FIFO 中的结构;数值编程到 MWCR 寄存器也显示。

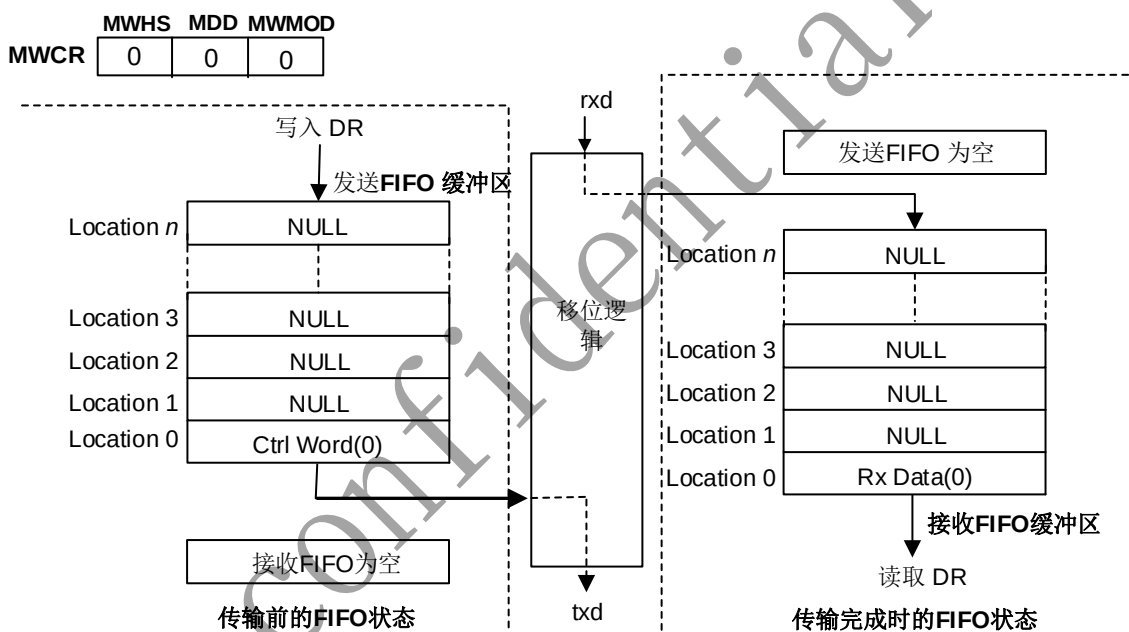


图 19-21 单次微线传输 FIFO 状态(接收数据帧)

微线协议的连续传输可以是顺序的,也可以是非顺序的,由 MWCR 寄存器中的 MWMOD Bit 位(0 位)控制。

非顺序的连续传输如下图所示,下一个传输的控制字紧跟在当前数据字的 LSB 之后。

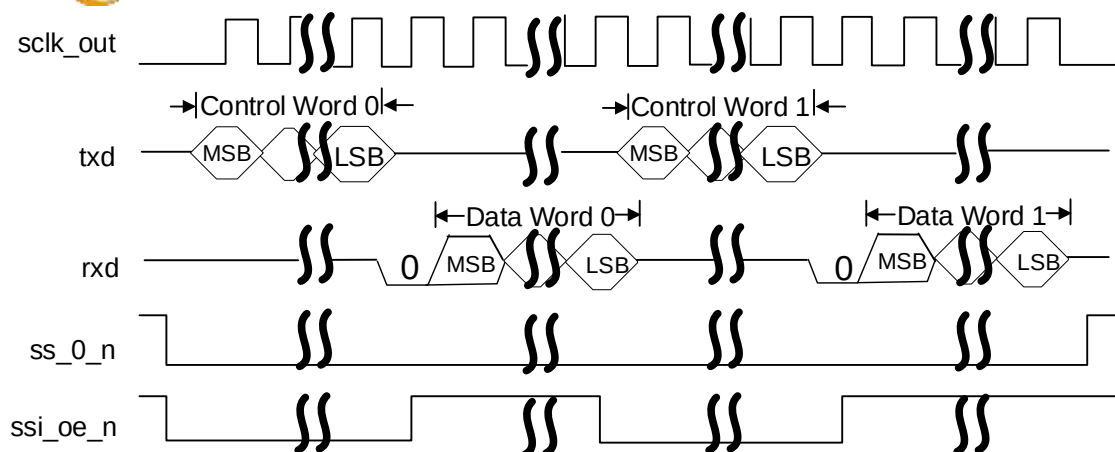


图 19-22 非顺序的微线连续传输(接收数据帧)

执行连续的非顺序传输所需要的惟一修改是在发送 FIFO 缓冲区中写入更多的控制字;如下图所示。在本例中,从外部串行从设备读取两个数据字。

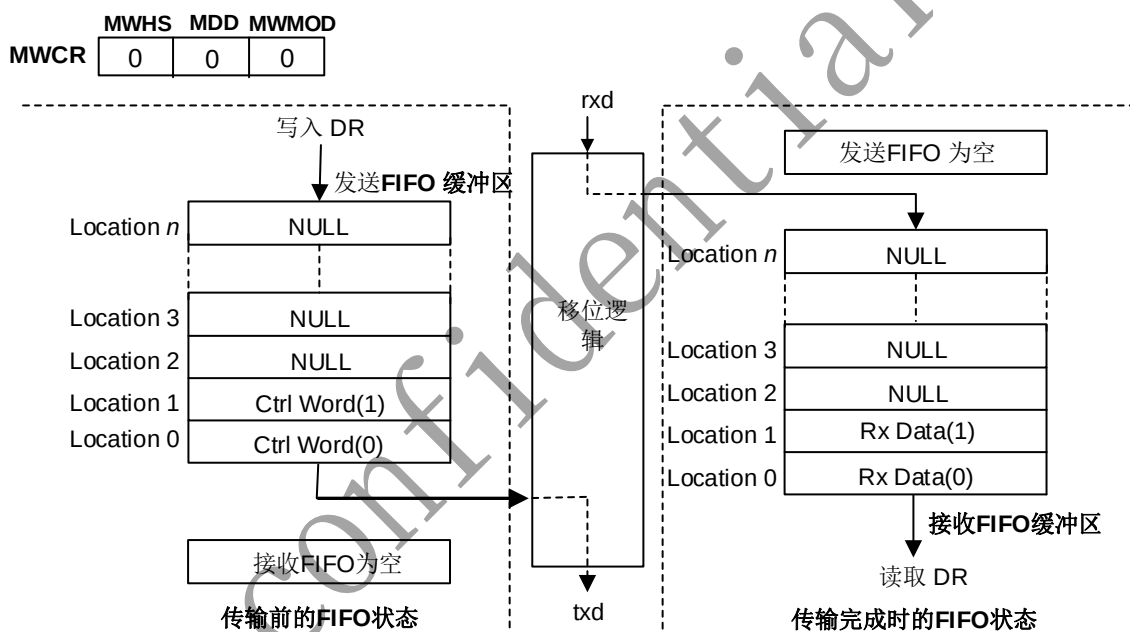


图 19-23 非顺序微波传输的 FIFO 状态(接收数据帧)

在顺序连续传输期间,只有一个控制字从 SSI 主机传输。传输是以与非顺序读取操作相同的方式开始的,但是这个循环将继续读取更多的数据。从设备自动地增加它的地址指针到下一个位置并且继续从那个位置提供数据。可以用这种方式读取任意数量的位置;当接收到的单词数等于 CTRLR1 寄存器中的值加 1 时,SSI 主设备终止传输。

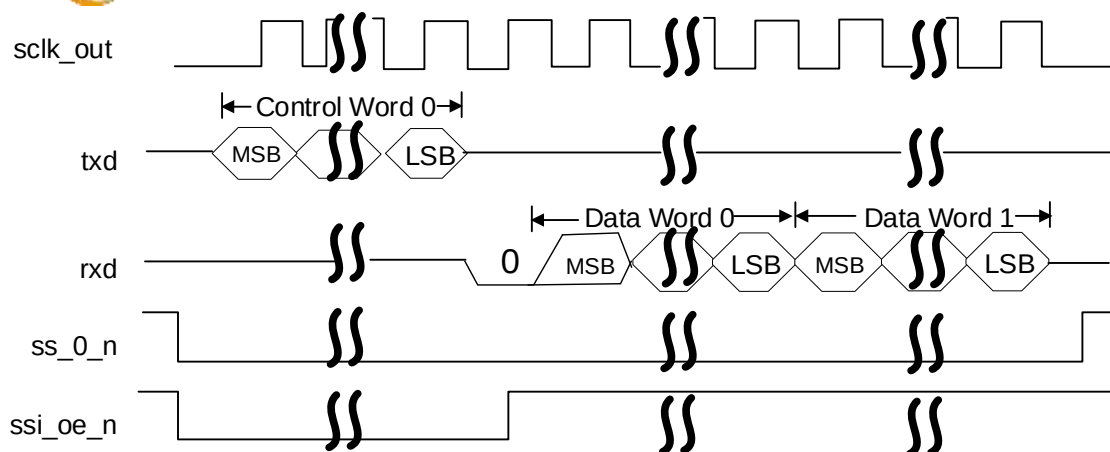


图 19-24 顺序的微波连续传输(接收数据帧)

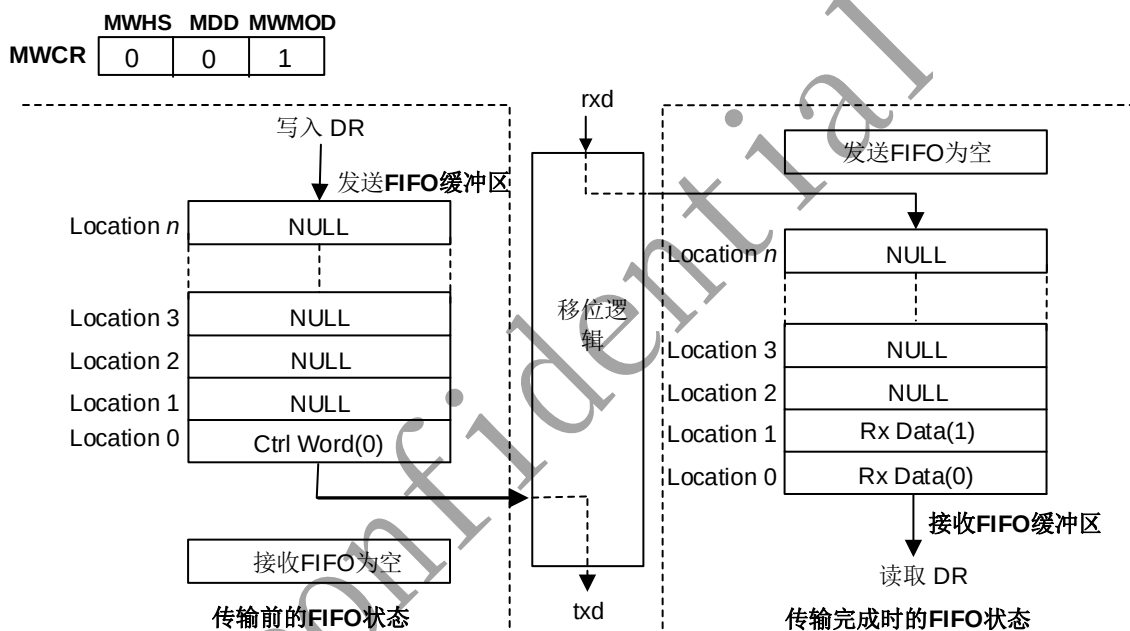


图 19-25 顺序的微波线传输的 FIFO 状态(接收数据帧)

当 $MDD = 1$ 时，这表示 SSI 串行主设备将数据传输给外部串行从设备。在控制字的 LSB 被传送之后，SSI 主设备立即开始传送数据帧到从外设。

下图显示了单个 SSI 串行主设备写到外部串行从设备的时序图。

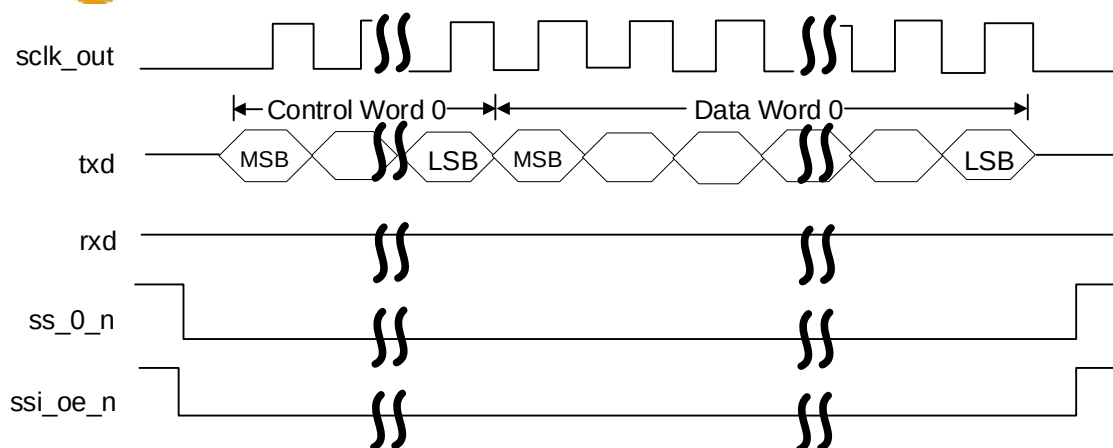


图 19-26 单次微线传输(传输数据帧)

注意:SSI 不支持连续连续的微线写入, 其中 $MDD = 1$, $MWMOD = 1$ 。

下图显示了数据和控制帧在传输之前的 FIFO 中的结构, 也显示了被编程到 MWCR 寄存器中的值

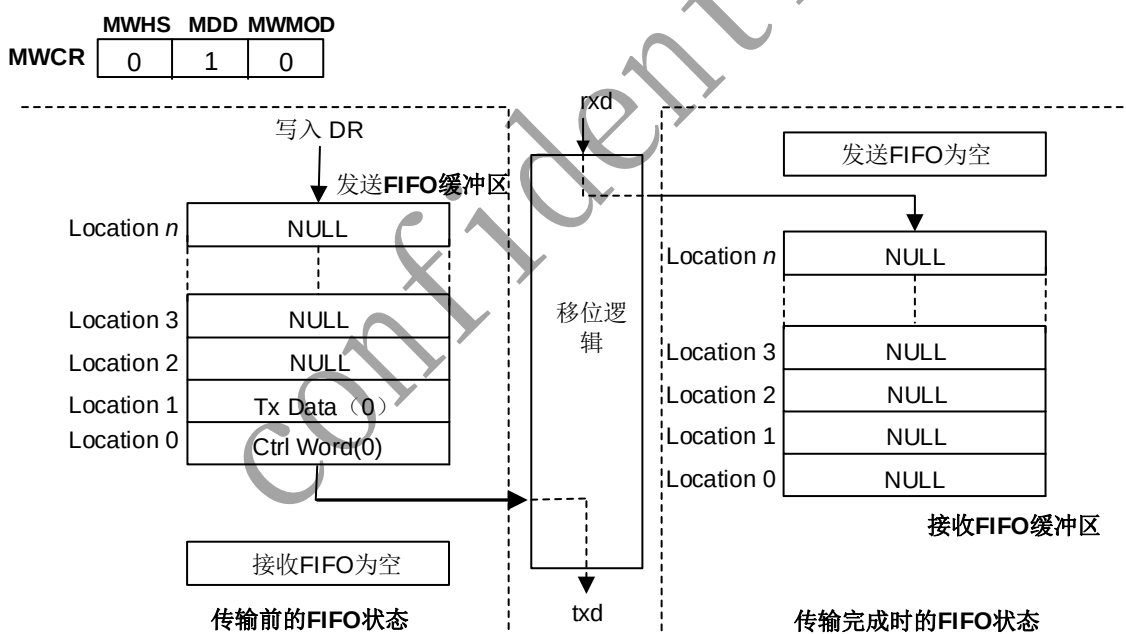


图 19-27 单次微线传输 FIFO 状态(传输数据帧)

连续传输如下图所示, 下一个传输的控制字紧跟在当前数据字的 LSB 之后。

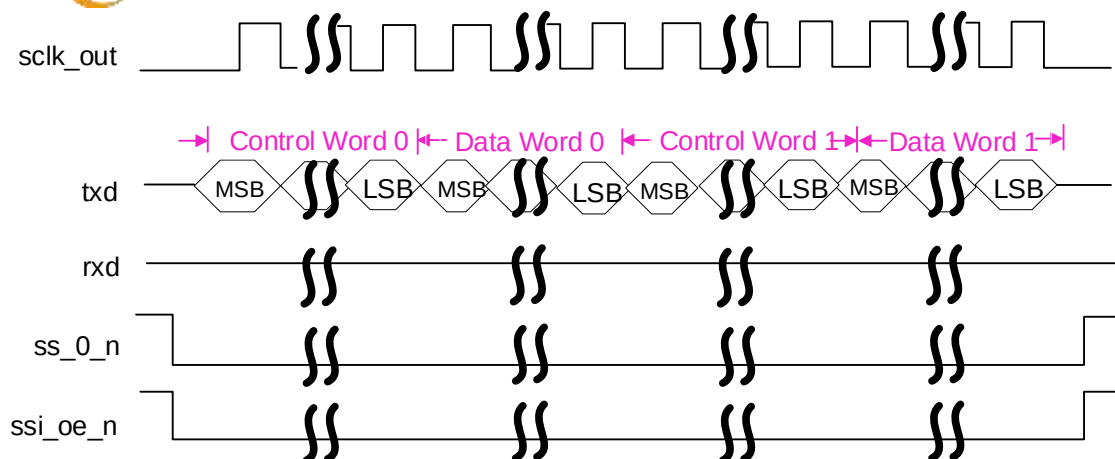


图 19-28 连续微线传输(传输数据帧)

要执行连续的传输，惟一需要做的修改是将更多的控制和数据字写到传输 FIFO 缓冲区中，如下图所示。此示例显示两个数据字被写入外部串行从设备。

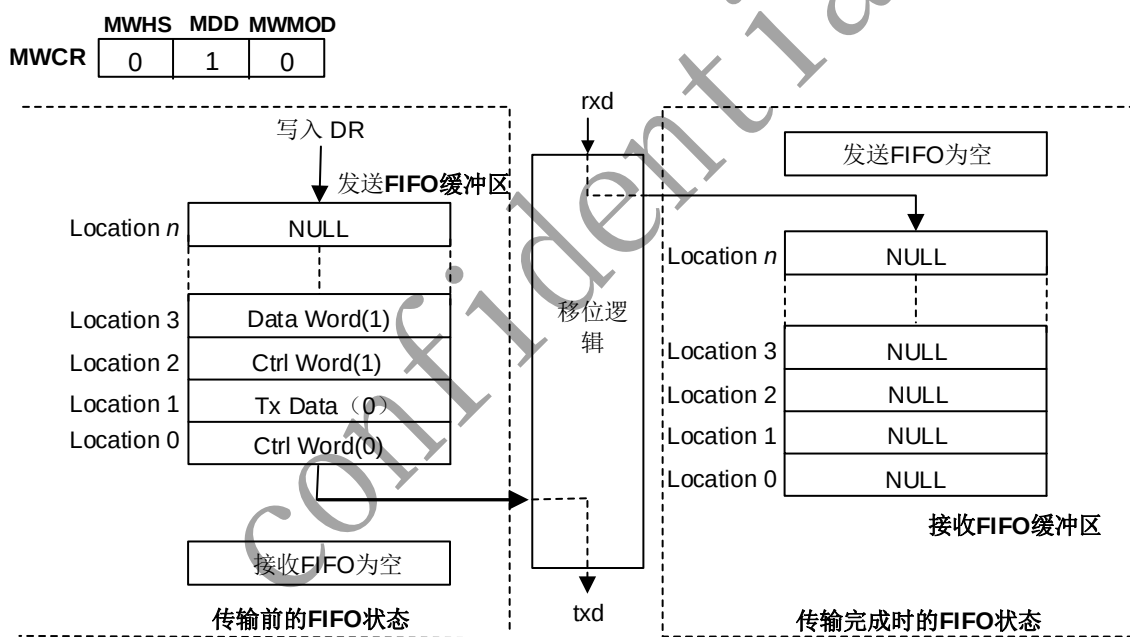


图 19-29 连续微线传输 FIFO 状态(传输数据帧)

微线握手接口也可以用于 SSI 主设备对外部串行从设备进行写操作。要启用握手接口，必须将 1 写入 MWCR 寄存器上的 MHS Bit 位(第 2 位)。当 MHS 设置为 1 时，SSI 串行主设备在完成传输或传输下一个连续传输的控制字之前，会检查从设备的就绪状态。

下图显示了一个启用了握手接口的连续微线传输示例。

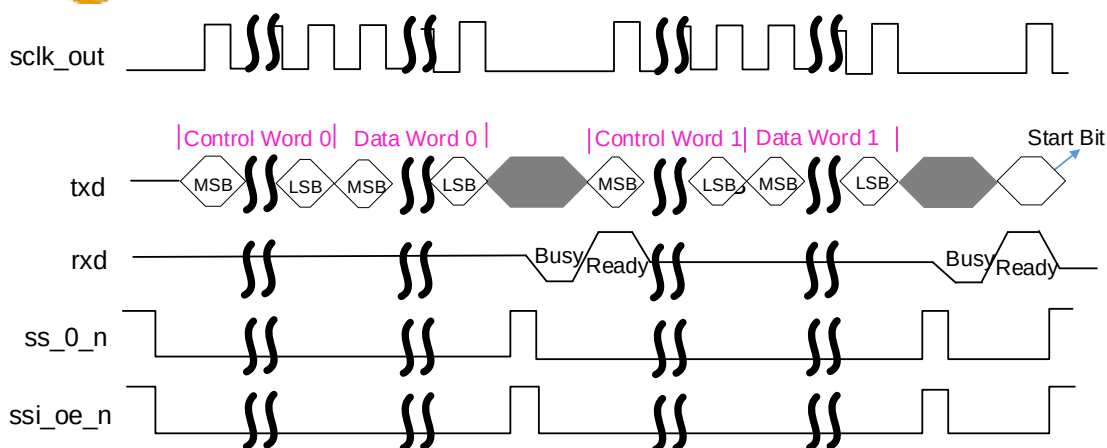


图 19-30 连续微线握手传输(传输数据帧)

在将第一个数据字传输到串行从设备之后，SSI 主设备轮询 rxd 输入，等待从设备的就绪状态。接收到就绪状态后，SSI 主设备开始传输下一个控制字。在传输完最后一帧数据后，SSI 主设备在完成传输之前传输一个开始位来清除从设备的就绪状态。

要从 SSI 主设备向串行从设备传输控制字(后面不跟数据)，必须在发送 FIFO 缓冲区中只有一个条目。连续传输两个控制字是不可能的，因为 SSI 中的移位逻辑将第二个控制字视为数据字。当 SSI 主机只传输一个控制字时，必须设置 MDD Bit 位(MWCR 寄存器的第 1 位)。

在以下 2 图所示的时序图中，启用了握手接口。如果握手接口被禁用(MHS=0)，则在控制字的 LSB 被从设备捕获之后，SSI 主设备在一个 sclk_out 周期终止传输。

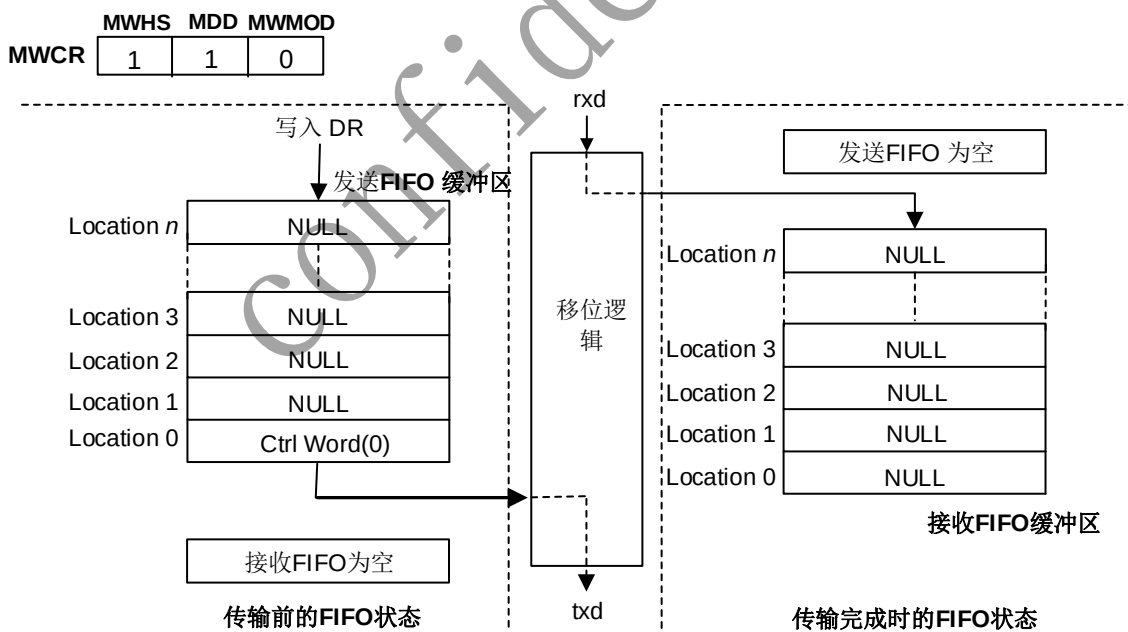


图 19-31 微线控制字传输 FIFO 状态

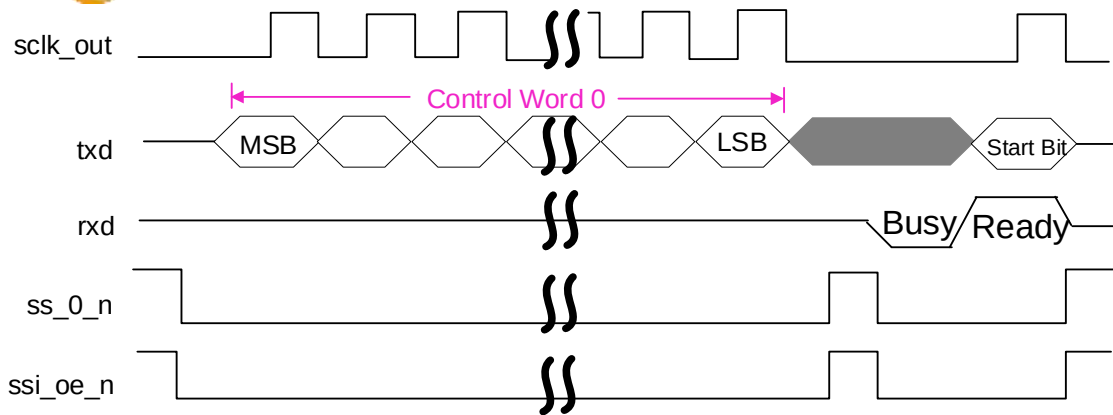


图 19-32 微线控制字

当 SSI 配置为串行从设备时，数据传输从从设备选择信号(ss_in_n)的下降沿开始。半串行时钟(sclk_in)周期过后，控制的第一个位出现在 rxd 行上。控制字的长度可以是 1 到 16 位，通过在 SSI_CR0 寄存器中写入 CFS Bit 位来设置。CFS Bit 位必须设置为来自串行主设备的预期控制字的大小。控制字的其余部分由 SSI 串行从设备接收(在 sclk_in 的上升边缘捕获)。在此接收期间，串行从机的 txd 线上没有数据被驱动(高阻抗)。

数据字的方向由 MDD Bit 位(第 1 位)MWCR 寄存器控制。当 MDD=0 时，这表示 SSI 从外部串行主设备接收数据。在传输控制字之后，串行主机立即开始将数据帧驱动到 SSI 从设备 rxd 线路上。数据在串行时钟的下降沿传播，并在上升沿捕获。从设备选择信号在传输过程中保持低电平激活状态，在数据传输后的一个半时钟周期中取消判断。SSI 从设备输出使能信号(ssi_oe_n)在传输期间保持非活动状态。

下图显示了 SSI 从设备从外部串行主设备读取单个传输的时序图。

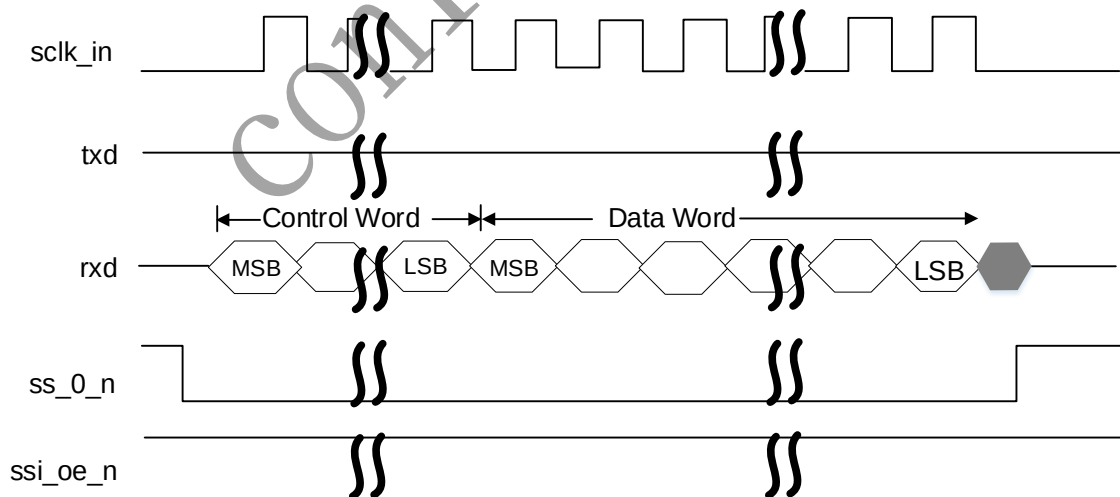


图 19-33 SSI 从设备单次微线串行传输 (MDD = 0)

下图显示了数据和控制帧是如何在接收 FIFO 中完成传输的;还显示的是编程到 MWCR 寄存器的值。

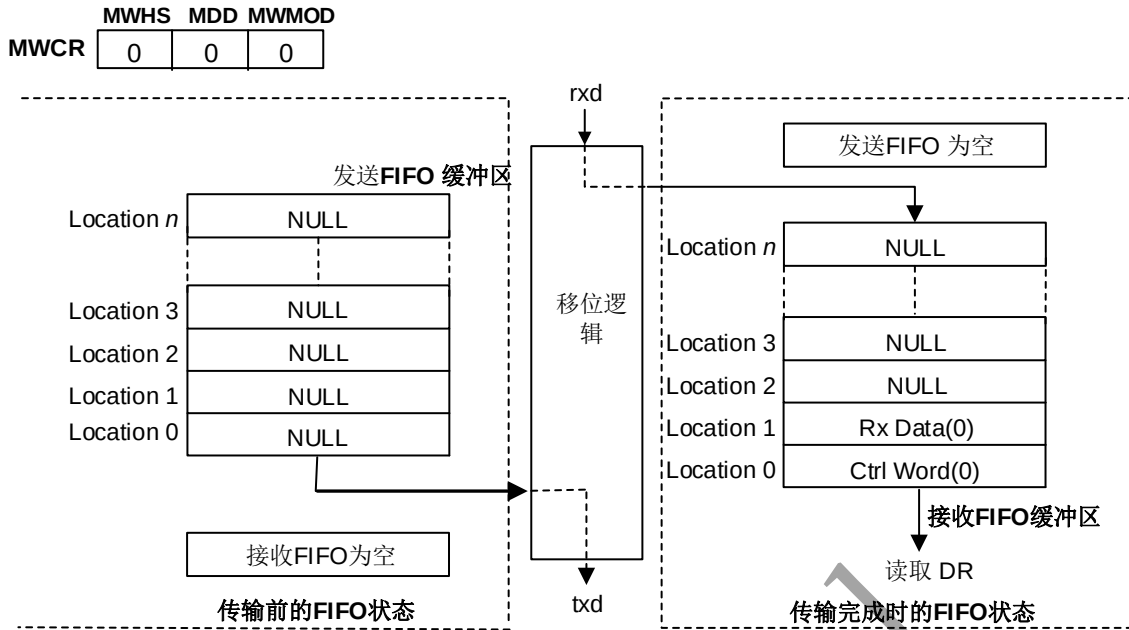


图 19-34 单次微线传输 FIFO 状态(接收数据帧)

当 MDD=1 时，这表明 SSI 从串行从设备将数据传输给外部串行主设备。在传输控制字的 LSB 之后，SSI 从传输一个虚拟的 0 位，然后在 txd 线路上传输 32 位数据帧。

下图显示了 SSI 作为从设备单次写入外部串行主设备的时序图。

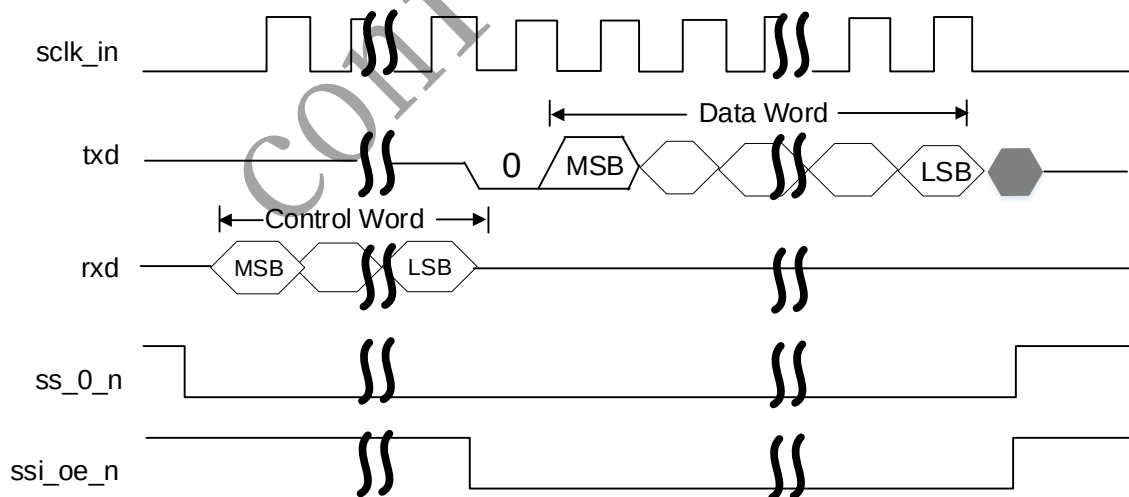


图 19-35 SSI 从设备单次微线串行传输(MDD=1)

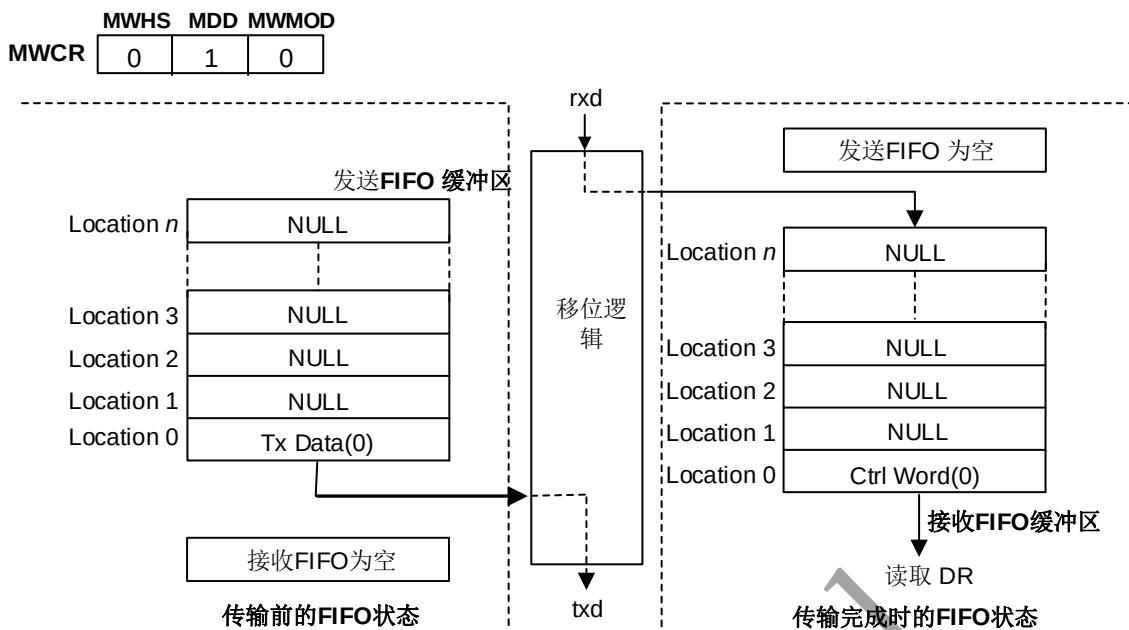


图 19-36 单次微线传输 FIFO 状态(传输数据帧)

1. 2. 6. 4 SPI 增强模式

在 SPI 增强模式下执行写操作

Quad SPI 写操作可分为三部分:

- ◆ 指令阶段
- ◆ 地址阶段
- ◆ 数据阶段

以下寄存器用于写操作:

- ◆ SSI_CR0.SPI_FRF -指定帧的传输格式。
- ◆ SSI_SPICR0 - 指定指令长度、地址和数据。
- ◆ SSI_SPICR0.INST_L - 指定一条指令的长度(一条指令可能的值是 0、4、8 或 16 位)。
- ◆ SSI_SPICR0.ADDR_L - 指定地址长度
- ◆ SSI_CR0.DFS_32 - 指定长度的数据。

一个指令采用一个 FIFO 位置, 地址可以采用多个 FIFO 位置。指令和地址都必须在数据寄存器(DR)中编程。SSI 将一直等待, 直到两者都被编程来启动写操作。

指令、地址和数据可以通过 Quad 模式发送, 该模式可以从 SSI_SPICR0.TRANS_TYPE 和 SSI_CR0.SPI_FRF 中进行选择。

Note :

- ◆ 如果 SSI_CR0.SPI_FRF 被选择为“标准 SPI 格式”, 所有内容都以标准 SPI 模式发

送，SSI_SPICR0.TRANS_TYPE 被忽略。

- ◆ SSI_CR0.SPI_FRF 只适用于 SSI_CR0.FRF 设定为 00。

下图显示了 Quad SPI 模式下的典型写操作。N 的值为 3。对于一个写操作，指令和地址只发送一次，然后在 DR 中编程数据帧，直到发送 FIFO 为空。

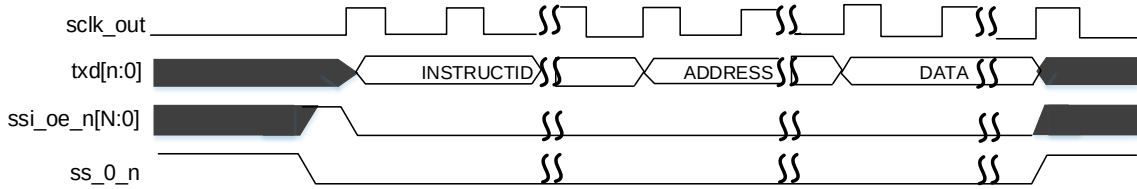


图 19-37 Quad SPI 模式典型的写操作

Quad 写操作初始化，SSI_CR0.SPI_FRF 必须分别设置为 01/10/11。这将设置传输类型，对于每个写命令，数据将以 SSI_CR0.SPI_FRF 中指定的格式传输。

下面是在 SPI 增强模式中进行写操作的可能情况

- ◆ Case A: 指令和地址都以标准 SPI 格式传输，即 SSI_SPICR0.TRANS_TYPE 必须设置为 00。下图显示了以标准 SPI 格式传输指令和地址时的时序图。如果 SSI_CR0.SPI_FRF 设置为 2'b11，则 N 的值为 7。如果 SSI_CR0.SPI_FRF 值设置为 2'b10，则 N 值为 3。如果 SSI_CR0.SPI_FRF 设置为 2'b01，则 N 值为 1。

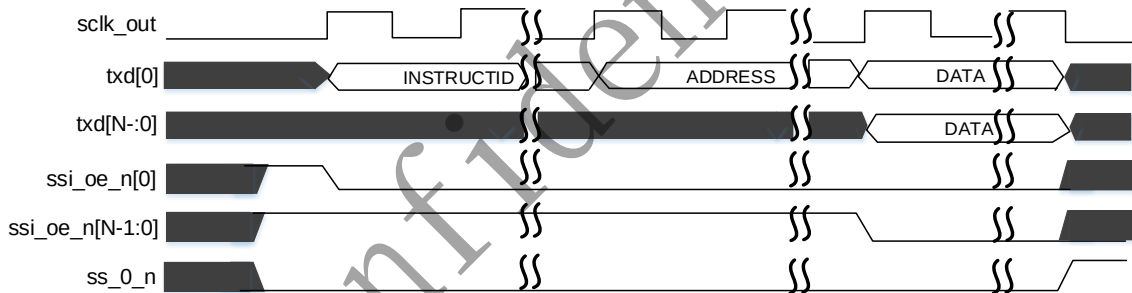


图 19-38 以标准 SPI 格式传送的指令和地址

- ◆ Case B: 指令和地址都以 SPI 增强模式传输，即 SSI_SPICR0.TRANS_TYPE 必须设置为 10。下图显示了指令和地址以 SSI_CR0.SPI_FRF 中指定 SPI 格式传输的时序图。如果 SSI_CR0.SPI_FRF 设置为 2'b11，则 N 的值为 7。如果 SSI_CR0.SPI_FRF 设置为 2'b10，则 N 的值为 3，如果 SSI_CR0.SPI_FRF 设置为 2'b01，则 N 的值为 1。

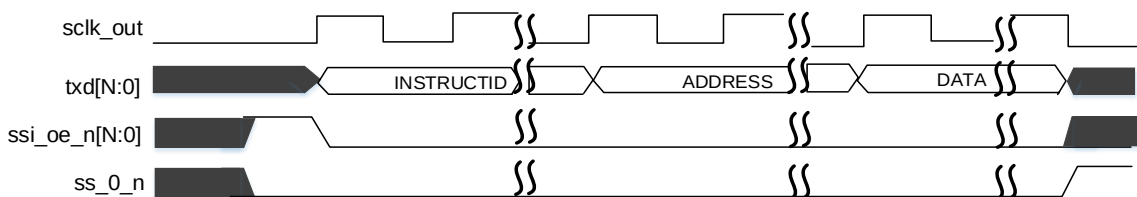


图 19-39 指令和地址都以 SPI 增强模式传输

- ◆ Case C: 指令只在 SPI 增强模式下传输，SSI_SPICR0.TRANS_TYPE 必须设置为 2'b10。下图显示了这种传输的时序图。如果 SSI_CR0.SPI_FRF 设置为 2'b11，则 N 的值为 7。如果 SSI_CR0.SPI_FRF 设置为 2'b10，则 N 的值为 3。如果 SSI_CR0.SPI_FRF 设置为 2'b01，则 N 的值为 1。

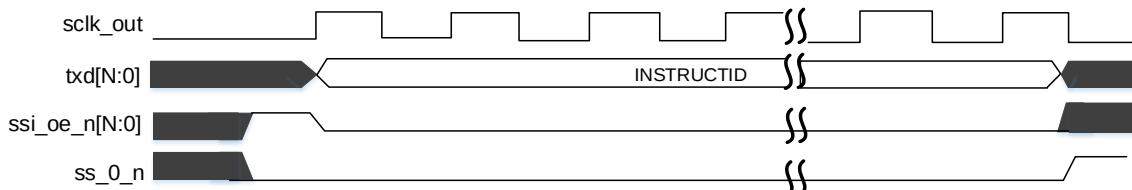


图 19-40 指令只能在 SPI 增强模式中传输

SPI 增强模式下的读操作

Quad SPI 读操作可分为四个阶段:

- ◆ 指令阶段
- ◆ 地址阶段
- ◆ 等待周期
- ◆ 数据阶段

可以使用 SSI_SPICR0.WAIT_CYCLES 对等待周期进行编程。被编程到 SSI_SPICR0.WAIT_CYCLES 的值将直接映射到 sclk_out 时间。例如，WAIT_CYCLES=0 表示没有等待，WAIT_CYCLES=1 表示一个等待周期，依此类推。等待周期被引入到目标从设备中，从输入到输出的不同模式及不同设备，等待周期可能都不同。

对于读操作，SSI 只发送一次指令和控制数据，并一直等待到它接收到 NDF (CTRLR1 寄存器) 数量的数据帧，然后取消从选择信号的断言。

下图显示了 Quad SPI 模式下的典型读操作。N 的值为 3。

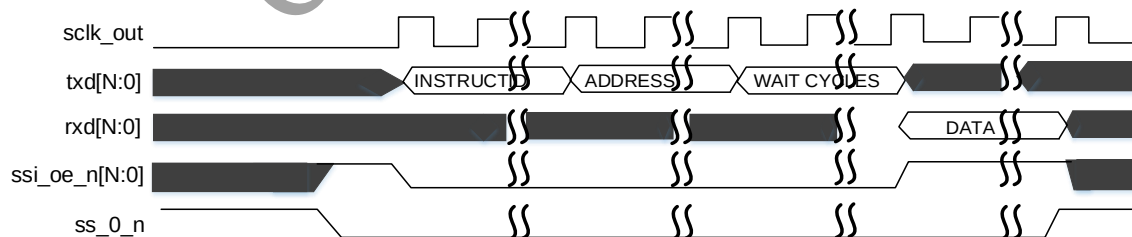


图 19-41 SPI 增强模式下的典型读操作

Quad 模式读操作初始 haul，SSI_CR0.SPI_FRF 必须分别设置为 01/10/11。它将设置传输类型，现在每个读命令的数据都将以 CTRLR0.SPI_FRF 中指定的格式进行传输。

以下是在 SPI 增强模式下可能发生的写操作:

- ◆ Case A: 指令和地址都以标准的 SPI 格式传输。为此 SSI_SPICR0.TRANS_TYPE

应该设置为 2'b00。下图显示了以标准 SPI 格式传输指令和地址时的时序图。该图还显示了 address 之后的等待周期，可以在 SSI_SPICR0.WAIT_CYCLES 中对其进行编程。如果 SSI_CR0.SPI_FRF 设置为 2'b11，则 N 的值为 7。如果 SSI_CR0.SPI_FRF 设置为 2'b10，则 N 的值为 3。如果 SSI_CR0.SPI_FRF 设置为 2'b01，则 N 的值为 1。

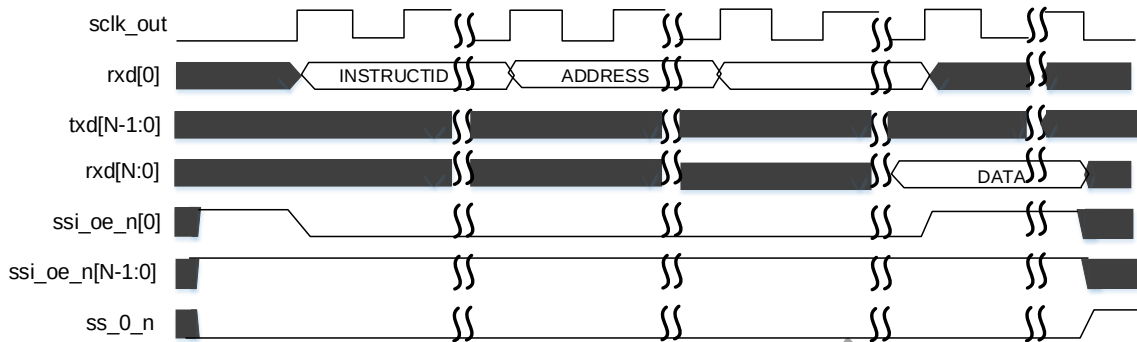


图 19-42 以标准 SPI 格式传送的指令和地址

- ◆ Case B: 以标准方式传输的指令和以 Dual SPI 模式传输的地址。为此 SSI_SPICR0.TRANS_TYPE 应该设置为 2'b01。下图显示了指令以标准格式传输和地址以双 SPI 格式传输的时序图。如果 SSI_CR0.SPI_FRF 设置为 2'b11，则 N 的值为 7。如果 SSI_CR0.SPI_FRF 设置为 2'b10，则 N 的值为 3。如果 SSI_CR0.SPI_FRF 设置为 2'b01，则 N 的值为 1。

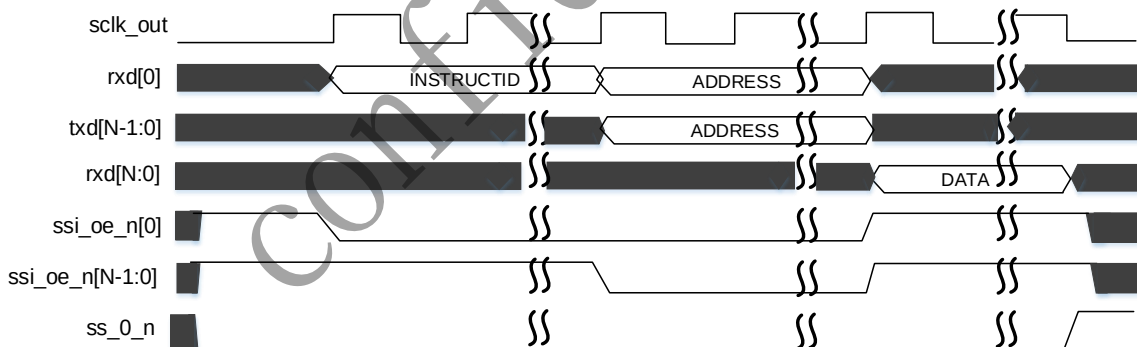


图 19-43 以 SPI 标准模式传输的指令和以增强模式传输的地址

- ◆ Case C: 指令和地址都以双 SPI 格式传输。为此 SSI_SPICR0.TRANS_TYPE 应该设置为 2'b10。下图显示了指令和地址以双 SPI 格式传输的时序图。如果 SSI_CR0.SPI_FRF 设置为 2'b11，则 N 的值为 7。如果 SSI_CR0.SPI_FRF 设置为 2'b10，则 N 的值为 3。如果 SSI_CR0.SPI_FRF 设置为 2'b01，则 N 的值为 1。

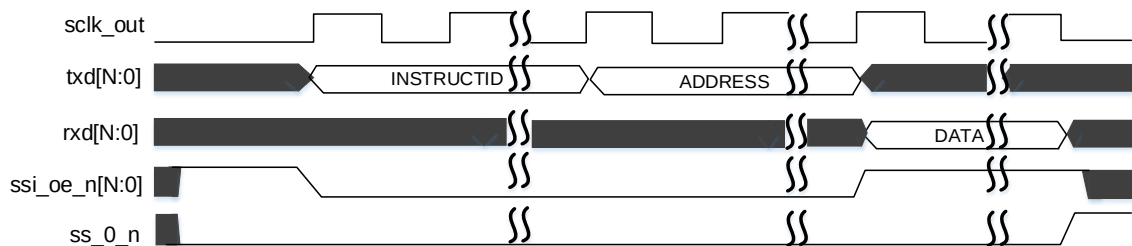


图 19-44 以 SPI 增强模式传输的指令和地址

- ◆ Case D: 无指令，无地址读取传输。为此 `SSI_SPICR0.ADDR_L` and `SSI_SPICR0.INST_L` 应该设置为 0，且 `SSI_SPICR0.WAIT_CYCLES` 设置为一个非 0 值。表列出了 `ADDR_L` 解码值和 SPI 增强模式的描述。

ADDR_L 解码值	功能描述
0000	0-bit 地址宽度
0001	4-bit 地址宽度
0010	8-bit 地址宽度
0011	12-bit 地址宽度
0100	16-bit 地址宽度
0101	20-bit 地址宽度
0110	24-bit 地址宽度
0111	28-bit 地址宽度
1000	32-bit 地址宽度
1001	36-bit 地址宽度
1010	40-bit 地址宽度
1011	44-bit 地址宽度
1100	48-bit 地址宽度
1101	52-bit 地址宽度
1110	56-bit 地址宽度
1111	60-bit 地址宽度

表 19-3 SPI 增强模式中 ADDR_L 解码值

下图显示了这种类型的转移的时序图。如果 `SSI_CR0.SPI_FRF` 设置为 `2'b11`，则 `N` 的值为 7。如果 `SSI_CR0.SPI_FRF` 设置为 `2'b10`，则 `N` 的值为 3。如果 `SSI_CR0.SPI_FRF` 设置为 `2'b01`，则 `N` 的值为 1。为了启动这种传输，软件必须在数据寄存器(DR)中执行一个虚拟的写操作，SSI 将等待已编程的等待周期，然后获取 `NDF` 字段中指定的数据量。

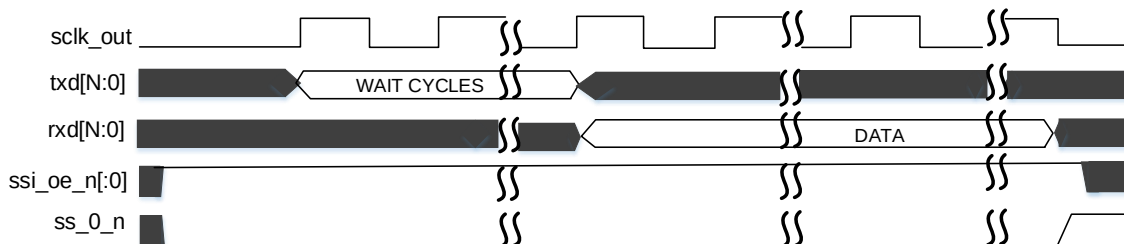


图 19-45 无指令，无地址读取传输

SPI 增强模式的高级 I/O 映射

下图显示了 SSI 在 Quad 模式下与另一个支持 Quad 模式的 SPI 设备之间的 I/O 映射。如图所示，IO[1]引脚与标准 SPI 模式下的 DO 一样使用，并与 rxd[1]引脚连接，rxd[1]引脚将对标准模式下的输入进行采样。

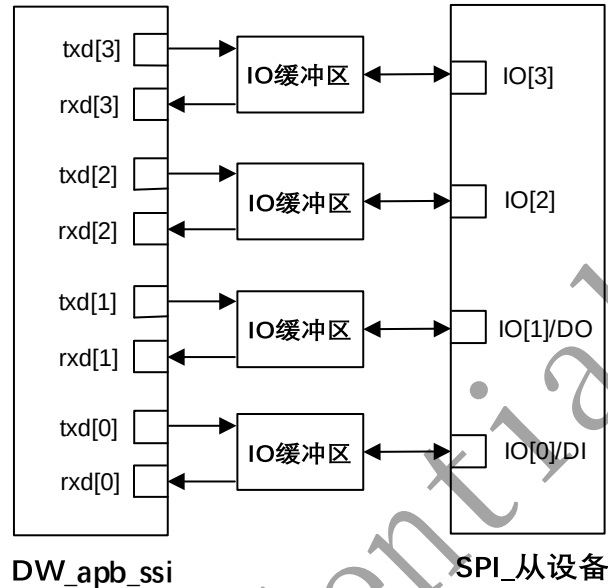


图 19-46 Quad SPI 模式的高级 I/O 映射

1.3 SSI 寄存器描述

1.3.1 SSI 控制寄存器 0(SSI_CR0)

		Address offset:		0x00											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								SSTE	Reserved	SPI_FRF		DFS_32			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CFS				SRL	SLV_OE	TMOD		SCPOL	SCPH	FRF		Reserved			

Name	Bit	Type	Reset value	Description
Reserved	31:25	N/A	0x00	保留，保持复位值
SSTE	24	R/W	0x0	Slave select toggle enable 当在时钟相位(SCPH)设置为 0 的 SPI 模式下运行时，该寄存器控制从选择行(ss*_n)在数据帧之间的行为。如果将该寄存器字段设置为 1，则 ss*_n 行将在连续数据帧之间进行切换，串行时钟

				(sclk)将保持默认值, 而 ss*_n 为高;如果这个寄存器字段设置为 0,ss*_n 将保持较低的水平, 而 sclk 将在传输期间连续运行。
Reserved	23	N/A	0x0	保留, 保持复位值
SSI_FRF	22:21	R/W	0x0	SSI Frame format 选择发送/接收数据的数据帧格式 Values: 0x0 (STD_SSI_FRF): 标准 SPI 帧格式 0x1 (DUAL_SSI_FRF): Dual SPI 帧格式 0x2 (QUAD_SSI_FRF): Quad SPI 帧格式 0x3 (OCTAL_SSI_FRF): 保留 Data Frame size. 用于选择 32 位传输模式下的数据帧大小。当数据帧的大小被编程为小于 32 位时, 接收数据被接收逻辑自动右对齐, 接收的 FIFO 的上位为零。在写入到传输 FIFO 之前, 您需要确保传输数据是右对齐的。传输逻辑在传输数据时忽略上面未使用的位。 注意: -如果 SSI_FRF = 0x01, DFS 值应该是 2 的倍数, -如果 SSI_FRF = 0x10, DFS 值应该是 4 的倍数, Values: 0x3 (FRAME_04BITS): 4-bit 串行数据传输 0x4 (FRAME_05BITS): 5-bit 串行数据传输 0x5 (FRAME_06BITS): 6-bit 串行数据传输 0x6 (FRAME_07BITS): 7-bit 串行数据传输 0x7 (FRAME_08BITS): 8-bit 串行数据传输 0x8 (FRAME_09BITS): 9-bit 串行数据传输 0x9 (FRAME_10BITS): 10-bit 串行数据传输 0xa (FRAME_11BITS): 11-bit 串行数据传输 0xb (FRAME_12BITS): 12-bit 串行数据传输 0xc (FRAME_13BITS): 13-bit 串行数据传输 0xd (FRAME_14BITS): 14-bit 串行数据传输 0xe (FRAME_15BITS): 15-bit 串行数据传输 0xf (FRAME_16BITS): 16-bit 串行数据传输 0x10 (FRAME_17BITS): 17-bit 串行数据传输 0x11 (FRAME_18BITS): 18-bit 串行数据传输 0x12 (FRAME_19BITS): 19-bit 串行数据传输 0x13 (FRAME_20BITS): 20-bit 串行数据传输 0x14 (FRAME_21BITS): 21-bit 串行数据传输 0x15 (FRAME_22BITS): 22-bit 串行数据传输 0x16 (FRAME_23BITS): 23-bit 串行数据传输 0x17 (FRAME_24BITS): 24-bit 串行数据传输 0x18 (FRAME_25BITS): 25-bit 串行数据传输 0x19 (FRAME_26BITS): 26-bit 串行数据传输 0x1a (FRAME_27BITS): 27-bit 串行数据传输 0x1b (FRAME_28BITS): 28-bit 串行数据传输 0x1c (FRAME_29BITS): 29-bit 串行数据传输 0x1d (FRAME_30BITS): 30-bit 串行数据传输 0x1e (FRAME_31BITS): 31-bit 串行数据传输 0x1f (FRAME_32BITS): 32-bit 串行数据传输
DFS_32	20:16	R/W	0x0	
CFS	15:12	R/W	0x0	Control frame size 选择微先帧格式的控制字的长度。 Values: 0x0 (SIZE_01_BIT): 1-bit 控制字 0x1 (SIZE_02_BIT): 2-bit 控制字 0x2 (SIZE_03_BIT): 3-bit 控制字 0x3 (SIZE_04_BIT): 4-bit 控制字 0x4 (SIZE_05_BIT): 5-bit 控制字 0x5 (SIZE_06_BIT): 6-bit 控制字 0x6 (SIZE_07_BIT): 7-bit 控制字

				0x7 (SIZE_08_BIT): 8-bit 控制字 0x8 (SIZE_09_BIT): 9-bit 控制字 0x9 (SIZE_10_BIT): 10-bit 控制字 0xa (SIZE_11_BIT): 11-bit 控制字 0xb (SIZE_12_BIT): 12-bit 控制字 0xc (SIZE_13_BIT): 13-bit 控制字 0xd (SIZE_14_BIT): 14-bit 控制字 0xe (SIZE_15_BIT): 15-bit 控制字 0xf (SIZE_16_BIT): 16-bit 控制字
SRL	11	R/W	0x0	Shift register loop 仅用于测试目的。当内部激活时，将发送移位寄存器输出连接到接收移位寄存器输入。 Values: 0x1 (TESTING_MODE): 测试模式:Tx 和 Rx shift reg 连接 0x0 (NORMAL_MODE): 正常模式
SLV_OE	10	R/W	0x0	Slave Output Enable. 仅在将 SSI 配置为串行从设备时才相关。当配置为串行主设备时，此 Bit 位没有任何功能。这个位启用或禁用来自 SSI 串行从设备的 ssi_oe_n 输出设置。当 SLV_OE = 1 时，ssi_oe_n 输出永远不可能是活动的。当 ssi_oe_n 输出控制从 txd 输出上的三态缓冲区分时，当 SLV_OE = 1 时，从 txd 输出上始终存在高阻抗状态。这在主设备以广播模式传输时非常有用(主设备将数据传输到所有从设备)。只有一个从设备可以响应主设备 rxd 线路上的数据。此位在重置后启用，如果不希望此设备以数据响应，则必须通过软件禁用(使用广播模式时)。 Values: 0x1 (DISABLED): 从设备输出禁止 0x0 (ENABLED): 从设备输出使能
TMOD	9:8	R/W	0x0	Transfer mode 选择串行通信的传输模式。此字段不影响传输的口是心非。仅指示接收或传输数据是否有效。 在只发送模式下，从外部设备接收到的数据无效，且未存储在接收 FIFO 存储器中;它将在下一次转移时被覆盖。 在仅接收模式下，传输的数据无效。在对发送 FIFO 进行第一次写操作之后，在传输期间将重新发送相同的单词。 在发送和接收模式中，发送和接收数据都是有效的。发送继续进行，直到发送 FIFO 为空。从外部设备接收到的数据存储在接收 FIFO 内存中，主机处理器可以访问它。 在 eeprom-read 模式下，在传输控制数据时，接收数据无效。当所有控制数据发送到 EEPROM 时，接收数据有效，传输数据无效。在这种模式下，发送 FIFO 中的所有数据都被认为是控制数据。此传输模式仅在将 SSI 配置为主设备时有效。 00 –发送和接收 01 – 仅发送 10 – 仅接收 11 - EEPROM Read 当 SSI 设置为 Quad SPI 模式时 10 – 读操作 01 – 写操作
SCPOL	7	R/W	0x0	Serial clock polarity 当帧格式(FRF)设置为摩托罗拉外围接口协议时有效。用于选择串行时钟的失效极性，当 SSI 主时钟不在串行总线上主动传输数据时，该串行时钟处于非活动状态。 Values: 0x0 (SCLK_LOW): 串行时钟低电平为非活动状态 0x1 (SCLK_HIGH): 串行时钟高电平为非活动状态
SCPH	6	R/W	0x0	Serial clock phase 当帧格式(FRF)设置为摩托罗拉外围接口协议时有效。串行时钟相位选择串行时钟与从时钟选择信号的关系。

FRF				当 SCPH = 0 时，在串行时钟的第一个边缘捕获数据。 当 SCPH = 1 时，从设备选择被激活后，串行时钟开始切换一个周期，并在串行时钟的第二个边缘捕获数据。 Values: 0x0 (SCPH_MIDDLE): 串行时钟在第一个数据位的中间进行切换 0x1 (SCPH_START): 串行时钟在第一个数据位开始时切换
	5:4	R/W	0x0	frame format: 选择哪个串行协议传输数据。 Values: 0x0 (MOTOROLA_SPI): 摩托罗拉外围接口协议 (SPI) 帧格式 0x1 (TEXAS_SSP): 德州仪器同步串行写哟 (SSP) 帧格式 0x2 (NS_MICROWIRE): 国家半导体微线协议 0x3 (RESERVED): 保留
Reserved	3:0	R/W	0x0	保留，保持复位值

1. 3. 2 SSI 控制寄存器 1(SSI_CR1)

		Address offset:		0x04											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NDF															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
NDF	15:0	R/W	0x0	Number of data frames 当 TMOD = 10 或 TMOD = 11 时，这个寄存器字段设置 SSI 连续接收的数据帧数。SSI 继续接收串行数据，直到接收到的数据帧数等于这个寄存器值加上 1，这使您能够在连续传输中接收最多 64 KB 的数据。当 SSI 被配置为一个串行从设备时，只要选择了从设备，传输就会继续。因此，这个寄存器没有任何作用，当 SSI 被配置为一个串行从设备时，它就不存在了。

1. 3. 3 LSSP 使能寄存器(SSI_ENR)

		Address offset:		0x08											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															EN

Name	Bit	Type	Reset value	Description
Reserved	31:4	N/A	0x00	保留，保持复位值

EN	0	R/W	0x0	Enable and disable the SSI 启用和禁用所有 SSI 操作。当禁用时，所有串行传输将立即停止。当设备被禁用时，发送和接收 FIFO 缓冲区被清除。在 SSI 设备启用时不能对某些控制寄存器进行编程。当禁用时，ssi_sleep 输出被置起(在延迟之后)，并通知系统关闭 ssi_clk 是安全的，从而节省了系统的功耗。 Values: 0x0 (DISABLE): 禁止串行传输 0x1 (ENABLED): 使能串行传输

1. 3. 4 SSI 微线控制寄存器(SSI_NWCR)

		Address offset:			0x0C										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													MHS	MOD	NWM OD

Name	Bit	Type	Reset value	Description
Reserved	31:3	N/A	0x00	保留，保持复位值
MHS	2	R/W	0x0	Handshaking 仅在将 SSI 配置为串行主设备时才相关。当配置为串行从设备时，这个 Bit 位没有任何功能。用于启用和禁用微线协议的繁忙/就绪握手接口。当启用时，SSI 在传输最后一个数据/控制位之后，在清除 SR 寄存器中的繁忙状态之前，从目标从设备中检查就绪状态。 Values: 0x0 (DISABLE): 握手接口禁止 0x1 (ENABLED): 握手接口使能
MOD	1	R/W	0x0	Direction control 定义使用微线串行协议时数据字的方向。当这个位设置为 0 时，SSI 从外部串行设备接收数据字。当该位设置为 1 时，数据字从 SSI 传输到外部串行设备。 Values: 0x0 (RECEIVE): SSI 接收数据 0x1 (TRANSMIT): SSI 发送数据
NWMOD	0	R/W	0x0	Microwire Transfer mode 定义微线传输是顺序的还是非顺序的。当使用顺序模式时，只需要一个控制字来发送或接收数据字块。当使用非顺序模式时，每个传输或接收的数据字必须有一个控制字。 Values: 0x0 (NON_SEQUENTIAL): 非顺序微线传输 0x1 (SEQUENTIAL): 顺序微线传输

1. 3. 5 SSI 从设备使能寄存器(SSI_SER)

		Address offset:		0x10										
		Reset value:		0x0000 0000										

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

Name	Bit	Type	Reset value	Description
Reserved	31:2	N/A	0x00	保留，保持复位值
SER	1:0	R/W	0x0	Slave Select Enable Flag. 该寄存器中的每个位对应于来自 SSI 主设备的从设备选择线路 (ss_x_n)。当寄存器中的一个位被设置(1)时，当串行传输开始时，来自主设备寄存器的相应从设备选择线路被激活。需要注意的是，在开始传输之前，在这个寄存器中设置或清除位对相应的从设备选择线路输出没有影响。在开始传输之前，您应该启用这个寄存器中与主设备想要通信的从设备相对应的位。当不在广播模式下运行时，只需要设置该字段中的一个位。 Values: 0x0 (NOT_SELECTED): 没有选择从设备 0x1 (SELECTED): 选择从设备

1. 3. 6 SSI 波特率寄存器(SSI_BAUD)

Address offset:		0x14													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SCKDV															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
SCKDV	15:0	R/W	0x0	SSI Clock Divider. 这个字段的 LSB 总是设置为 0，不受写操作的影响，因为写操作可以确保在这个寄存器中保存一个偶数值。如果值为 0，则禁用串行输出时钟(sclk_out)。sclk_out 的频率由下式得到： $F_{sclk\_out} = F_{ssi\_clk} / SCKDV$ 其中 SCKDV 是 2 到 65534 之间的任意偶值。

1. 3. 7 SSI 发送 FIFO 阈值寄存器(SSI_TXFTL)

Address offset:		0x18													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														TFT	

Name	Bit	Type	Reset value	Description
Reserved	31:2	N/A	0x00	保留，保持复位值
TFT	1:0	R/W	0x0	TX FIFO Threshold 控制发送 FIFO 控制器触发中断的阈值(或以下)。当传输 FIFO 条目的数量小于或等于这个值时，将触发传输 FIFO 空中断。 当传输 FIFO 中出现 TFT 或更少的数据项时，将产生 ssi_txe_intr

1. 3. 8 SSI 接收 FIFO 阈值寄存器(SSI_RXFTL)

		Address offset:		0x1C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														RFT	

Name	Bit	Type	Reset value	Description
Reserved	31:2	N/A	0x00	保留，保持复位值
RFT	1:0	R/W	0x0	RX FIFO Threshold 控制接收 FIFO 控制器触发中断的阈值(或以上)。当接收 FIFO 条目的数量大于或等于这个值+ 1 时，接收 FIFO 完全中断将被触发。 当接收 FIFO 中出现 RFT 或更多数据条目时，将产生 ssi_rxf_intr

1. 3. 9 SSI 发送 FIFO 指针寄存器(SSI_TXFL)

		Address offset:		0x20											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														TXTFL	

Name	Bit	Type	Reset value	Description
Reserved	31:3	N/A	0x00	保留，保持复位值
TXTFL	2:0	R	0x0	TX FIFO Level 指示发送 FIFO 中有效数据项的数目。

1. 3. 10 SSI 接收 FIFO 指针寄存器(SSI_RXFL)

		Address offset:		0x24											
		Reset value:		0x0000 0000											

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

Name	Bit	Type	Reset value	Description
Reserved	31:3	N/A	0x00	保留，保持复位值
RXTFL	2:0	R/W	0x0	RX FIFO Level 指示接收 FIFO 中有效数据项的数目。

1. 3. 11 SSI 状态寄存器(SSI_SR)

		Address offset:		0x28											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									DCOL	TXE	RFF	RFNE	TFE	TFNF	BUSY

Name	Bit	Type	Reset value	Description
Reserved	31:7	N/A	0x00	保留，保持复位值
DCOL	6	R/W	0x0	Data Collision Error 仅当 SSI 被配置为主设备时才相关。当 SSI 主设备正在传输过程中，其他主设备产生 ss_in_n 输入时，将设置这个位。这将通知处理器，最后一次数据传输在完成之前停止。读取时清除此位。 0x0 (NO_ERROR_CONDITION): 无错误 0x1 (TX_COLLISION_ERROR): 发送数据冲突错误
TXE	5	R	0x0	Transmission Error 如果发送 FIFO 在发送开始时为空，则置起。只有将 SSI 配置为从设备时才能设置这个位。来自先前传输的数据在 txd 线路上重新发送。读取时清除此位。 Values: 0x0 (NO_ERROR): 无错误 0x1 (TX_ERROR): 传输错误
RFF	4	R	0x0	Receive FIFO Full. 当接收 FIFO 完全满时，设置这个位。当接收 FIFO 包含一个或多个空位置时，清除这个位。 0x0 (NOT_FULL): 接收 FIFO 不满 0x1 (FULL): 接收 FIFO 满
RFNE	3	R	0x0	Receive FIFO Not Empty. 当接收 FIFO 包含一个或多个条目时置起，当接收 FIFO 为空时清除。这个位可以通过软件轮询来完全清空接收 FIFO。 Values: 0x0 (EMPTY): 接收 FIFO 为空 0x1 (NOT_EMPTY): 接收 FIFO 不为空
TFE	2	R	0x0	Transmit FIFO Empty. 当传送 FIFO 完全为空时置起。当发送 FIFO 包含一个或多个有效条目时，清除这个位。此 Bit 位不请求中断。

TFNF	1	R	0x0	0x0 (NOT_EMPTY):发送 FIFO 不为空 0x1 (EMPTY): 发送 FIFO 为空 Transmit FIFO Not Full. 当传送 FIFO 包含一个或多个空位置时设置, 当 FIFO 满时清除。 0x0 (FULL): 发送 FIFO 为满 0x1 (NOT_FULL): 发送 FIFO 不为满
				SSI Busy Flag. 置起时, 表示串行传输正在进行中;当清除时, 表示 SSI 处于空闲或禁用状态。 0x0 (INACTIVE): SSI 空闲或禁止 0x1 (ACTIVE): SSI 正在传输数据
BUSY	0	R	0x0	

1. 3. 12 SSI 中断屏蔽寄存器(SSIMR)

		Address offset:			0x2C										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										MSTI M	RXFI M	RXOI M	RXUI M	TXOI M	TXEI M

Name	Bit	Type	Reset value	Description
Reserved	31:6	N/A	0x00	保留, 保持复位值
MSTIM	5	R	0x0	Multi-Master Contention Interrupt Mask. 如果将 SSI 配置为串行从设备, 则不存在此位字段。 Values: 0x0 (MASKED): ssi_mst_intr 中断屏蔽 0x1 (UNMASKED): ssi_mst_intr 中断不屏蔽
RXFIM	4	R	0x0	Receive FIFO Full Interrupt Mask Values: 0x0 (MASKED): ssi_rxf_intr 中断屏蔽 0x1 (UNMASKED): ssi_rxf_intr 中断不屏蔽
RXOIM	3	R	0x0	Receive FIFO Overflow Interrupt Mask Values: 0x0 (MASKED): ssi_rxo_intr 中断屏蔽 0x1 (UNMASKED): ssi_rxo_intr 中断不屏蔽
RXUIM	2	R	0x0	RX FIFO underflow int mask Values: 0x0 (MASKED): ssi_rxu_intr 中断屏蔽 0x1 (UNMASKED): ssi_rxu_intr 中断不屏蔽
TXOIM	1	R	0x0	TX FIFO OVERFLOW INT MASK Values: 0x0 (MASKED): ssi_txo_intr 中断屏蔽 0x1 (UNMASKED): ssi_txo_intr 中断不屏蔽
TXEIM	0	R	0x0	TX FIFO EMPTY INT MASK Values: 0x0 (MASKED): ssi_txe_intr 中断屏蔽 0x1 (UNMASKED): ssi_txe_intr 中断不屏蔽

1. 3. 13 SSI 中断状态寄存器(SSI_ISR)

		Address offset:		0x30											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										MSTI S	RXFIS	RXOI S	RXUI S	TXOI S	TXEIS

Name	Bit	Type	Reset value	Description
Reserved	31:6	N/A	0x00	保留，保持复位值
MSTIS	5	R	0x0	Multi-Master Contention Interrupt Status. 如果将 SSI 配置为串行从设备，则不存在此位字段。 0x0 (INACTIVE): ssi_mst_intr 中断无效 0x1 (ACTIVE): ssi_mst_intr 中断有效
RXFIS	4	R	0x0	Receive FIFO Full Interrupt Status 0x0 (INACTIVE): ssi_rxf_intr 中断无效 0x1 (ACTIVE): ssi_rxf_intr 中断有效
RXOIS	3	R	0x0	Receive FIFO Overflow Interrupt Status 0x0 (INACTIVE): ssi_rxo_intr 中断无效 0x1 (ACTIVE): ssi_rxo_intr 中断有效
RXUIS	2	R	0x0	Receive FIFO Underflow Interrupt Status Values: 0x0 (INACTIVE): ssi_rxu_intr 中断无效 0x1 (ACTIVE): ssi_rxu_intr 中断有效
TXOIS	1	R	0x0	Transmit FIFO Overflow Interrupt Status 0x0 (INACTIVE): ssi_txo_intr 中断无效 0x1 (ACTIVE): ssi_txo_intr 中断有效
TXEIS	0	R	0x0	Transmit FIFO Empty Interrupt Status 0x0 (INACTIVE): ssi_txe_intr 中断无效 0x1 (ACTIVE): ssi_txe_intr 中断有效

1. 3. 14 SSI 中断标志寄存器(SSI_RISR)

		Address offset:		0x34											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										MSTI R	RXFI R	RXOI R	RXUI R	TXOI R	TXEI R

Name	Bit	Type	Reset value	Description
Reserved	31:6	N/A	0x00	保留，保持复位值
MSTIR	5	R	0x0	Multi-Master Contention Raw Interrupt Status. 如果将 SSI 配置为串行从设备，则不存在此位字段。 Values: 0x0 (INACTIVE): ssi_mst_intr 未产生中断（屏蔽前） 0x1 (ACTIVE): ssi_mst_intr 产生中断（屏蔽前）

RXFIR	4	R	0x0	Receive FIFO Full Raw Interrupt Status Values: 0x0 (INACTIVE): ssi_rxf_intr 未产生中断（屏蔽前） 0x1 (ACTIVE): ssi_rxf_intr 产生中断（屏蔽前）
	3	R	0x0	Receive FIFO Overflow Raw Interrupt Status Values: 0x1 (ACTIVE): ssi_rxo_intr 未产生中断（屏蔽前） 0x0 (INACTIVE): ssi_rxo_intr 产生中断（屏蔽前）
	2	R	0x0	Receive FIFO Underflow Raw Interrupt Status Values: 0x0 (INACTIVE): ssi_rxu_intr 未产生中断（屏蔽前） 0x1 (ACTIVE): ssi_rxu_intr 产生中断（屏蔽前）
	1	R	0x0	Transmit FIFO Overflow Raw Interrupt Status Values: 0x0 (INACTIVE): ssi_txo_intr 未产生中断（屏蔽前） 0x1 (ACTIVE): ssi_txo_intr 产生中断（屏蔽前）
	0	R	0x0	Transmit FIFO Empty Raw Interrupt Status Values: 0x0 (INACTIVE): ssi_txe_intr 未产生中断（屏蔽前） 0x1 (ACTIVE): ssi_txe_intr 产生中断（屏蔽前）

1. 3. 15 SSI 发送上溢清除寄存器(SSI_TXOICR)

		Address offset:		0x38											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															TXOI CR

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值
TXOICR	0	R	0x0	Clear Transmit FIFO Overflow Interrupt. 这个寄存器反映中断的状态。从这个寄存器读取一个数据，清除 ssi_txo_intr 中断;写作没有效果。

1. 3. 16 SSI 接收上溢清除寄存器(SSI_RXOICR)

		Address offset:		0x3C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															RXOI CR

Name	Bit	Type	Reset value	Description

Reserved	31:1	N/A	0x00	保留，保持复位值
RXOICR	0	R	0x0	Clear Receive FIFO Overflow Interrupt. 这个寄存器反映中断的状态。从这个寄存器读取一个数据，清除 ssi_rxo_intr 中断;写作没有效果。

1. 3. 17 SSI 接收下溢清除寄存器(SSI_RXUICR)

		Address offset:		0x40											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															RXUI CR

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值
RXUICR	0	R	0x0	Clear Receive FIFO Underflow Interrupt. 这个寄存器反映中断的状态。从这个寄存器读取一个数据，清除 ssi_rxu_intr 中断;写作没有效果。

1. 3. 18 SSI 多主竞争清除寄存器(SSI_MSTICR)

		Address offset:		0x44											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															MSTI CR

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值
MSTICR	0	R	0x0	Clear Multi-Master Contention Interrupt. 这个寄存器反映中断的状态。从这个寄存器读取一个数据，清除 ssi_mst_intr 中断;写作没有效果。

1. 3. 19 SSI 中断清除寄存器(SSI_ICR)

		Address offset:		0x48											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

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

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值
ICR	0	R	0x0	Clear Interrupts. 如果下面的任何中断是活动的，则设置此寄存器。读操作清除 ssi_txo_intr、ssi_rxu_intr、ssi_rxo_intr 和 ssi_mst_intr 中断。写入此寄存器无效。

1. 3. 20 SSI DMA 控制寄存器(SSI_DMCCR)

		Address offset:		0x4C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														TDM AE	RDM AE

Name	Bit	Type	Reset value	Description
Reserved	31:2	N/A	0x00	保留，保持复位值
TDMAE	1	R/W	0x0	TransmitDMAEnable. 这个位启用/禁用发送 FIFO DMA 通道。 Values: 0x0 (DISABLE): 发送 DMA 禁止 0x1 (ENABLED): 发送 DMA 使能
RDMAE	0	R/W	0x0	ReceiveDMAEnable. 这个位启用/禁用接收 FIFO DMA 通道。 Values: 0x0 (DISABLE): 接收 DMA 禁止 0x1 (ENABLED): 接收 DMA 使能

1. 3. 21 SSI DMA 发送数据量寄存器(SSI_DMATDLR)

		Address offset:		0x50											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														DMATDL	

Name	Bit	Type	Reset value	Description

Reserved	31:2	N/A	0x00	保留，保持复位值
DMATDL	1:0	R/W	0x0	Transmit Data Level. 这个 Bit 位控制发送逻辑发出 DMA 请求的级别。它等于阈值时；也就是说，dma_tx_req 信号是在发送 FIFO 中的有效数据项数等于或小于该字段值，且 TDMAE = 1 时生成的。 dma_tx_req 在发送 FIFO 中出现 DMATDL 或更少的数据项时产生

1. 3. 22 SSI DMA 接收数据量寄存器(SSI_DMARDLR)

		Address offset:		0x54											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														DMARDL	

Name	Bit	Type	Reset value	Description
Reserved	31:2	N/A	0x00	保留，保持复位值
DMARDL	1:0	R/W	0x0	Receive Data Level. 这个 Bit 位控制接收逻辑发出 DMA 请求的级别。阈值 = DMARDL+1;也就是说，dma_rx_req 是在接收 FIFO 中有效数据条目的数量等于或大于此字段值+ 1 且 RDMAE=1 时生成的。 dma_rx_req 是在接收 FIFO 中出现 DMARDL 或更多有效数据条目时断言的。

1. 3. 23 SSI 数据寄存器(SSI_DR)

		Address offset:		0x60~0xEC											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DR<31:16>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DR<15:0>															

Name	Bit	Type	Reset value	Description
DR	31:0	R/W	0x0	Data Register. 当写入到此寄存器时，必须对数据进行右对齐。读取的数据自动右对齐。

1. 3. 24 SSI 接收采样延时寄存器(SSI_RSD)

		Address offset:		0xF0											
		Reset value:		0x0000 0000											

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

Name	Bit	Type	Reset value	Description
Reserved	31:8	N/A	0x00	保留，保持复位值
RSD	7:0	R/W	0x0	Rxd Sample Delay. 这个寄存器用于延迟 rxd 输入端口的采样。每个值表示 rxd 的样本上的单个 ssi_clk 延迟。 注意:如果这个寄存器的编程值超过了内部移位寄存器的深度，则对 rxd 示例应用零延迟。

1. 3. 25 SSI SPI 控制 0 寄存器(SSI_SPICR0)

Address offset:		0xF4													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WAIT_CYCLE					Reserved	INST_L		Reserved			ADDR_L			TRANS_TYPE	

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
WAIT_CYCLE	15:11	R/W	0x0	Wait cycles 在控制帧传输和数据接收之间的 Dual/Quad SPI 模式的等待周期数。此值指定为 SPI 时钟周期数。
Reserved	10	N/A	0x0	保留，保持复位值
INST_L	9:8	R/W	0x0	Instruction Length Dual/Quad mode instruction length in bits. Values: 0x0 (INST_L_0): 0-bit (无指令) 0x1 (INST_L_1): 4-bit 指令 0x2 (INST_L_2): 8-bit 指令 0x3 (INST_L_3): 16-bit 指令
Reserved	7:6	R/W	0x0	保留，保持复位值
ADDR_L	5:2	R/W	0x0	Address Length. 这个位定义要传输的地址长度。只有在将这么多的比特编入 FIFO 之后，传输才可以开始。 Values: 0x0 (ADDR_L_0): 0-bit 地址位宽 0x1 (ADDR_L_1): 4-bit 地址位宽 0x2 (ADDR_L_2): 8-bit 地址位宽 0x3 (ADDR_L_3): 12-bit 地址位宽 0x4 (ADDR_L_4): 16-bit 地址位宽 0x5 (ADDR_L_5): 20-bit 地址位宽 0x6 (ADDR_L_6): 24-bit 地址位宽 0x7 (ADDR_L_7): 28-bit 地址位宽

TRANS_TYPE	1:0	R/W	0x0	0x8 (ADDR_L_8): 32-bit 地址位宽 0x9 (ADDR_L_9): 36-bit 地址位宽 0xa (ADDR_L_10): 40-bit 地址位宽 0xb (ADDR_L_11): 44-bit 地址位宽 0xc (ADDR_L_12): 48-bit 地址位宽 0xd (ADDR_L_13): 52-bit 地址位宽 0xe (ADDR_L_14): 56-bit 地址位宽 0xf (ADDR_L_15): 60-bit 地址位宽
				Address and instruction transfer format. 选择 SSI 是否以标准 SPI 模式或 SSI_CR0 中选择的 SPI 模式传输指令/地址。SPI_FRF 字段。 00 -指令和地址将以标准的 SPI 模式发送。 01 -指令将以标准 SPI 模式发送，地址将以 SSI_CR0.SPI_FRF 指定的模式发送。 10 -指令和地址都将按照 SSI_CR0.SPI_FRF 指定的模式发送。 11 -保留。

confidential

第20章 串行外设接口 2(SPI)

1.1 SPI 简介

可以使用的 SPI / I2S 接口和外部设备进行基于 SPI 协议和 I2S 音频协议的通信。SPI 或 I2S 模式可通过软件选择。器件复位后默认选中 SPI 模式。

串行外设接口 (SPI) 协议，支持半双工，全双工和简单同步等方式与外部设备的串行通信。该接口可配置为主机模式，在这种情况下，它向外部的从属设备提供通信时钟 (SCK)。界面也能够以多重配置的方式操作。

I2S 音频协议，也是一个同步串行通信接口，使用 3 个外部信号。它可以解决四个不同的音频标准，包括飞利浦的 I2S 标准的 MSB 和 LSB 对齐的标准和 PCM 标准。它可以实现主从模式的半双工通信。I2S 作为主设备的时候可以向外部从设备的提供通信时钟。

1.2 SPI 主要特性

◆ SPI 主要特性

- 主设备或从设备模式
- 三线全双工同步传输
- 两条线的半双工同步传输 (双向数据线)
- 两条线的简单同步传输 (单向数据线)
- 8 位到 16 位数据的大小选择
- 多重模式的能力
- 8 个主模式波特率分频器。
- 从模式频率高达 $f_{PCLK}/2$ 。
- 主机和从机模式下都可以由硬件或软件管理 NSS：主/从模式操作的动态变化
- 可编程时钟极性和相位
- 高位在前或低位在前可设置
- 专用的发送和接收状态标志，全部支持中断触发
- SPI 总线忙状态标志
- 支持 SPI 摩托罗拉方式
- 硬件 CRC 功能实现可靠的通信：
 - ◇ CRC 值可以作为 TX 模式的最后一个字节传送
 - ◇ 自动 CRC 错误检查上次接收到的字节
- 支持 Rx 和 Tx FIFO，深度是 16
- 支持 DMA 传输
- 支持 SPI TI 模式

◆ I2S 特性

- 单工通信 (仅发送或接收)

- 主或从操作
- 8 位线性可编程分频器，以达到精确的音频采样频率（从 8 kHz 到 96kHz）
- 数据格式可以是 16 位，24 位或 32 位
- 音频通道固定数据帧为 16 位（16 位数据帧）或 32 位（16 位，24 位，32 位数据帧）
- 可编程的时钟极性（稳态）
- 从发送模式的下溢标志和接收功能（主或从）的溢出标志
- 16 位的发送和接收寄存器，在通道两端各有一个寄存器
- 支持的 I2S 协议：
 - ◇ I2S 飞利浦标准
 - ◇ MSB 对齐标准（左对齐）
 - ◇ LSB 对齐标准（右对齐）
 - ◇ PCM 标准（16 位通道帧上带长的或短的帧同步或者 16 位数据帧扩展为 32 位通道帧）
- 数据方向始终是 MSB 在先
- 支持发送和接收的 DMA 支持（16 位宽）
- 主时钟可以向一个外部的音频组件输出。速率固定为 256 倍的 FS（其中 FS 是音频采样频率）
- 支持 TX 和 RX FIFO，深度是 16
- 提供 2 路外部时钟输入

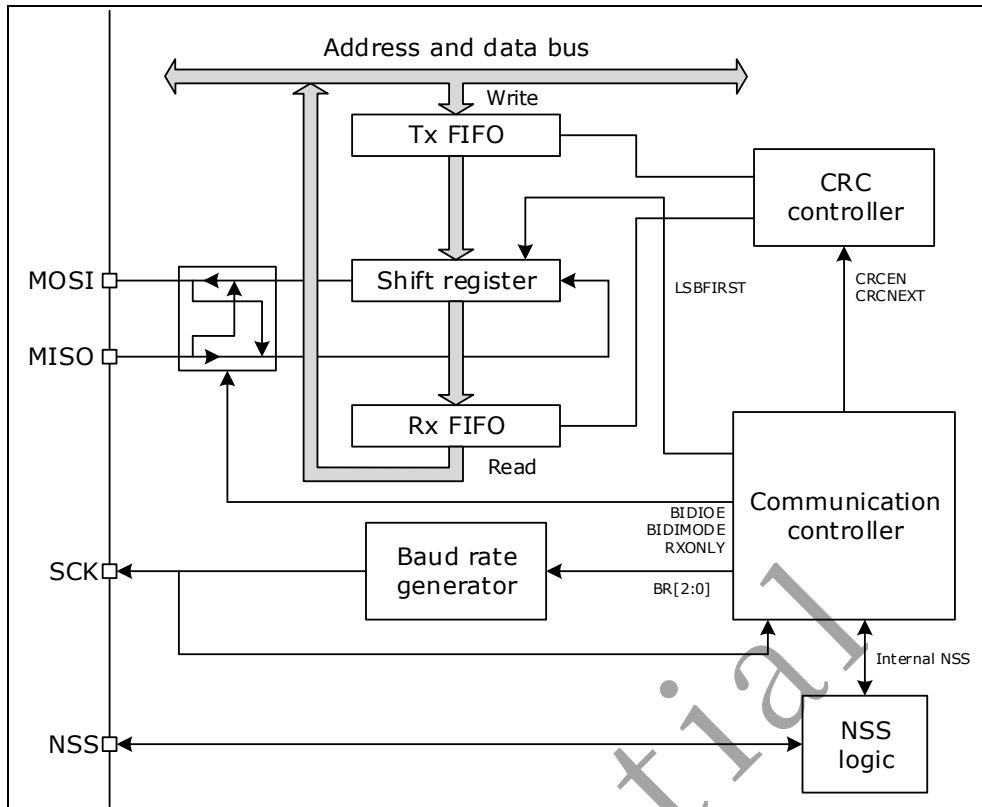


图 20-1 SPI 框图

1.3 SPI 功能描述

1.3.1 基本描述

SPI 允许 MCU 和外部设备之间的同步串行通信。应用软件可以通过状态标志，或使用专用的 SPI 中断来管理通信过程。SPI 和它们之间的相互作用的主要内容如图 22-1 所示。

通常 SPI 通过 4 个引脚与外部设备相连。

- ◆ **MISO:** 主设备输入/从设备输出引脚。一般情况下，此引脚用于在从模式下的数据发送和主模式下的数据接收。
- ◆ **MOSI:** 主设备输出/从设备输入引脚。一般情况下，此引脚用于在从模式下的数据接收和主模式下的数据发送。
- ◆ **SCK:** SPI 串行时钟输出引脚，作为主设备输入，从设备的输出。
- ◆ **NSS:** 从机片选脚。根据 SPI 和 NSS 设置，此引脚可用于：
 - 选择一个从机来通信
 - 同步数据帧或者
 - 检测多个主机之间的冲突

SPI 总线允许一个主设备和一个或多个从设备之间的通信。总线由至少有两条线-时钟信号和同步数据传输。其他信号基于 SPI 节点和主机间的数据交换目的来添加。

1.3.2 一个主设备和一个从设备之间的通信

SPI 允许 MCU 根据目标的设备和应用程序的要求，使用不同的配置进行通信。这些配置使用 2 或 3 线（软件 NSS 管理）或 3 或 4 线（硬件 NSS 管理）。通信总是由主机发起的。

1.3.2.1 全双工通信

默认情况下，SPI 被配置为全双工通信。在此配置中，主机和从机的移位寄存器通过两个单向线，MOSI 和 MISO 引脚进行连接。在 SPI 通信时，按照主机提供的 SCK 时钟沿进行同步数据传输。主机的数据经由 MOSI 发送到从机，从机的数据经由 MISO 线发送到主机。当数据帧传输完成（所有位转移）时，主机和从机间的信息交换就完成了。

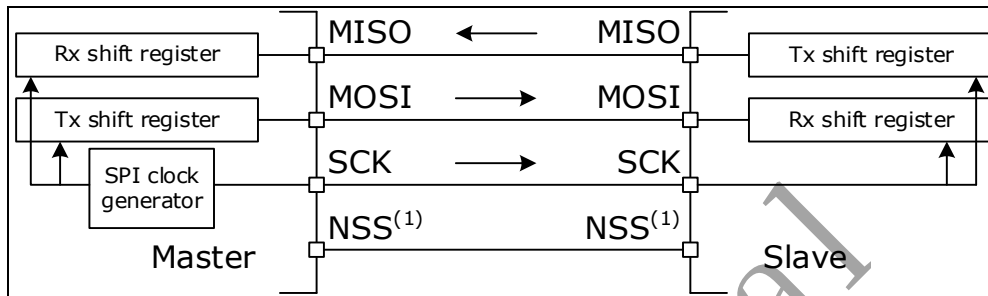


图 20-2 全双工通信

1. NSS 引脚可用于提供主从之间的硬件控制流。用于主机选择需要通信的外围从机设备。

1.3.2.2 半双工通信

通过设置在 SPIx_CR1 寄存器 BIDIMODE 位，可以令的 SPI 工作在 half-duplex 模式下。在此配置中性由一个单一的跨接线连接处主机寄存器和从机。在通信期间，数据按照 SPIx_CR1 寄存器的 BDIOE 位的设定，由主机移位寄存器同步于 SCK 时钟沿传输到从机。在此配置中，主机的 MISO 引脚和从机的 MOSI 引脚都是自由的，可作为 GPIO 供其他应用程序使用。

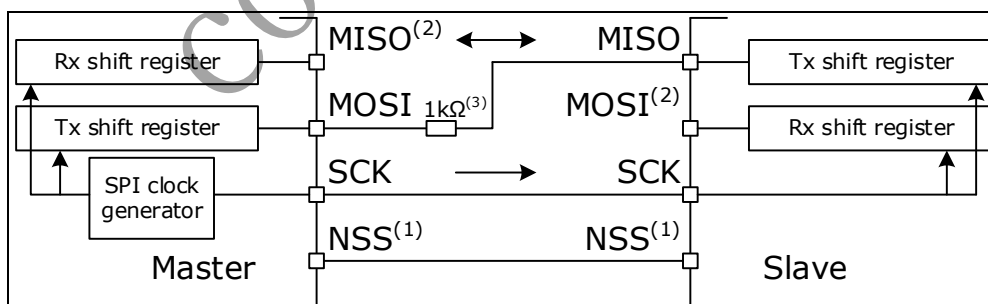


图 20-3 半双工通信

1. NSS 引脚可用于提供主从之间的硬件控制流。用于主机选择需要通信的外围从机设备。
2. 在此配置中，主机的 MISO 引脚和从机的 MOSI 引脚可以作为 GPIO 用。
3. 当工作在双向模式下的两个节点之间的通信方向不同步改变，新的发送器访问公共数据线，而原来的发送器仍然在电路上保持相反的值(该值取决于 SPI 配置和通

信数据)时,就会出现临界情况。然后,两个节点在公用线路上临时提供相反的输出级别,直到下一个节点也相应地更改其方向设置。建议在此模式下,在 MISO 和 MOSI 引脚之间插入串联电阻,以保护输出,并限制输出之间的电流。

1.3.2.3 简单通信

可以通过 SPIx_CR2 寄存器 RXONLY 位选择 SPI 工作在单工模式下,实现单发送或单接收通讯。在此配置中,由一个单一的跨接线连接主机寄存器和从机。其余 MISO 和 MOSI 引脚不用于通信,并可以作为标准的 GPIO 使用。

- ◆ 单发送模式 (RXONLY=0): 配置设置和全双工相同。应用程序必须忽略在未使用的输入引脚上捕获的信息。这个脚可以作为一个标准的 GPIO 引脚。
- ◆ 单接收模式 (RXONLY=1): 应用程序可以设置 RXONLY 位以禁用 SPI 输出功能。

在配置为从机时, MISO 输出被禁用,并且可以当作一个 GPIO 引脚来使用。在从机选择信号处于激活状态时,从机将从 MOSI 引脚连续接收数据。根据数据缓冲区的配置,会出现数据接收事件。

在配置为主机时, MOSI 输出被禁用,并且可以当作一个 GPIO 引脚来使用。SPI 使能时,时钟信号会不断产生。要停止时钟输出,只有清除 RXONLY 位或 SPE 位,并等待直到从 MISO 引脚的输入样式的完成并填充数据缓冲区结构,这取决于其具体配置。

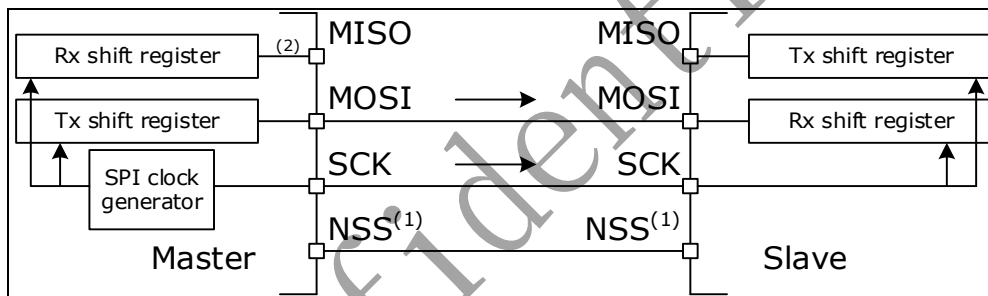


图 20-4 简单通信

1. NSS 引脚可用于提供主从之间的硬件控制流。可选地,大头针可以不使用的外围。然后必须在内部处理主服务器和从服务器的流。
2. 在移位寄存器中输入的信息被捕获,但在标准单发送模式下必须丢弃。(例如,OVF 格式标志)
3. 在此配置中,只要是 MISO 引脚都可用作 GPIO。

注: 简单通信模式可以被一个固定传输方向的半双工通信模式替代 (当 BDIO 位不变时,双向模式启用)。

1.3.3 标准多从机通信

在有两个或两个以上独立的从机的配置时,主机用 GPIO 引脚来管理每个从机的片选线。主机必须通过 GPIO 拉低其中一个从机的 NSS 引脚。一旦做到这一步,就会建立一个标准的主机和专用的通信渠道。

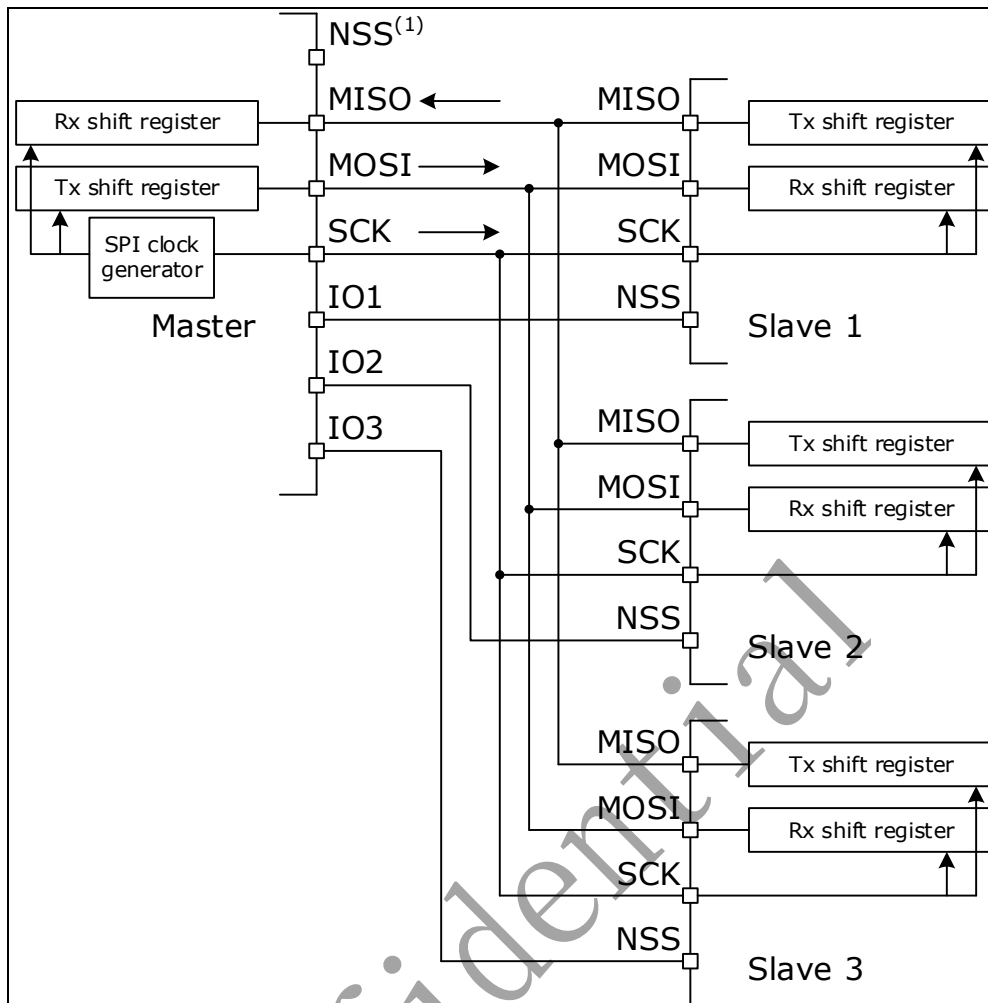


图 20-5 多从机通信

1. NSS pin is not used on master side at this configuration. It has to be managed internally (SSM=1, SSI=1) to prevent any MODF error.
2. 由于从机的 MISO 引脚连接在一起，所有的从机都必须将他们的 MISO 引脚设置为备用功能漏极开路状态。

1.3.4 多主机通信

除非 SPI 总线主要不是为多主功能设计的，否则用户可以使用 build in 特性来检测两个节点之间的潜在冲突，同时尝试控制总线。对于这种检测，NSS 引脚是在硬件输入模式下配置的。

以这种模式工作的两个以上 SPI 节点的连接是不可能的，因为在同一时间只有一个节点可以将其输出应用到公共数据线上。

当节点不活动时，默认情况下两者都处于从属模式。一旦一个节点想要超越总线上的控制，它将自己切换到主模式，并通过专用的 GPIO 引脚对另一个节点的从选择输入应用活动电平。会话完成后，释放主动从选择信号，控制总线临时的节点返回到被动从模式，等待下一个会话开始。

如果两个节点可能同时发出母版请求，则会出现总线冲突事件(请参阅模式错误 MODF 事件)。然后，用户可以应用一些简单的仲裁过程(例如，通过在两个节点上应用预定义的不同超时来

推迟下一次尝试)。

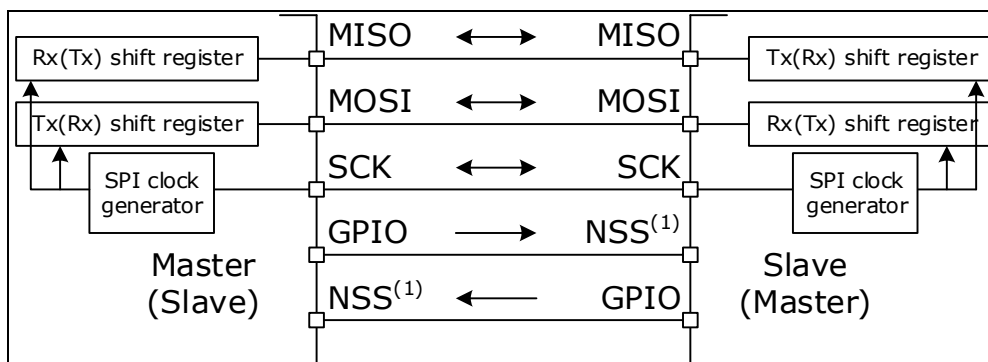


图 20-6 多主机通信

1. NSS 引脚在两个节点的硬件输入模式下配置。它的活动级支持 MISO 行输出控制，因为被动节点被配置为从节点。

1.3.5 从机选择(NSS)引脚管理

在从机模式下时，NSS 作为一个标准的“片选”输入工作，并允许主从通信。在主机模式下，NSS 可以用来作为输入或输出。作为输入，它可以防止多主总线冲突，作为输出，它可以驱动一个单一的从机片选信号。

可以设置 SPIx_CR1 寄存器中的 SSM 位，来选择使用硬件或软件的从机选择管理：

- ◆ **软件 NSS 管理 (SSM=1)：**在此配置中，从机选择信息由内部寄存器 SPIx_CR1 的 SSI 位值来驱动。外部 NSS 引脚可以由其他应用程序自由支配。
- ◆ **硬件 NSS 管理 (SSM=0)：**在这种情况下，有两种可能的配置。这种设置由（寄存器 SPIx_CR1 SSOE 位）来决定 NSS 的输出。
 - **NSS 输出使能 (SSM=0, SSOE=1)：**此配置仅用于当 MCU 作为主机时。NSS 引脚由硬件管理。在 SPI 作为主模式 (SPE 的=1) 的同时 NSS 信号被拉低，并保持低直到 SPI 被禁止 (SPE=0)。如果 NSS 脉冲模式被打开 (NSSP=1)，在连续通信间隔期间可以产生一个 NSS 脉冲。在这种 NSS 设置中，SPI 不能支持多重主机工作状态。
 - **NSS 输出禁止 (SSM=0, SSOE=0)：**如果在总线上 MCU 作为主机，那么此配置就允许多主机通讯。如果在这种模式下 NSS 引脚被拉低，SPI 进入主模式故障状态，并自动重新配置为从机模式。在从机模式下，NSS 引脚作为标准的片选输入来工作，当 NSS 线被拉低时，从机被选中。

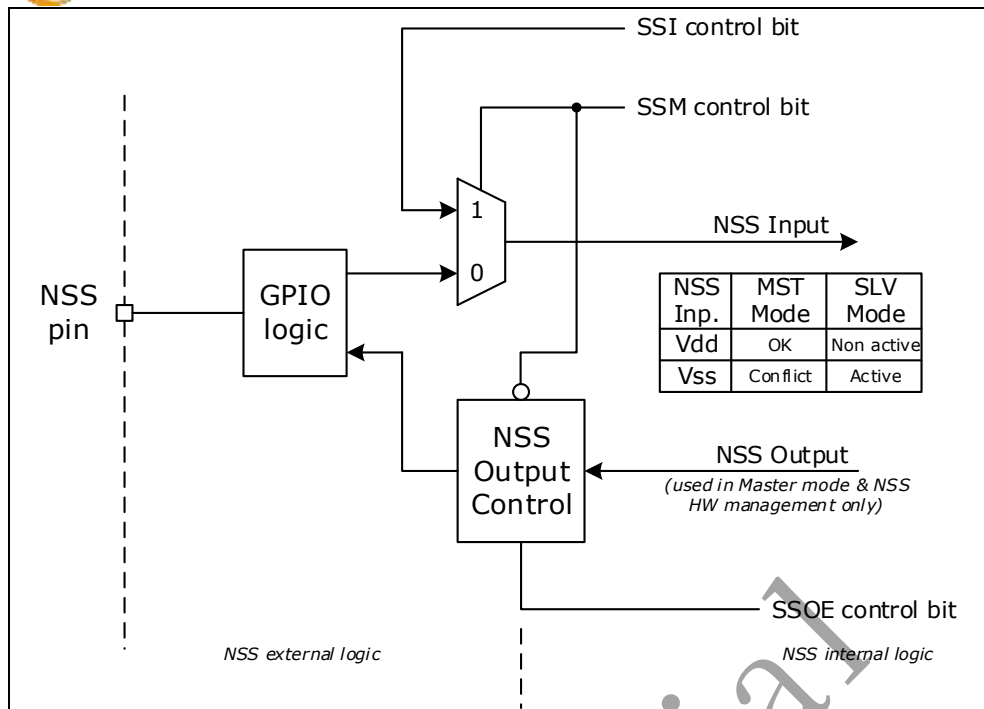


图 20-7NSS 引脚管理

1. 3. 6 通信格式

在 SPI 通信中，接收和发送操作同时进行。串行时钟（SCK）同步发送并同时采样数据线上的信息。通讯格式取决于时钟相位，时钟极性和数据帧格式。为了能够正常通讯，主机和从机必须遵循相同的通信格式。

1. 3. 6. 1 时钟相位和极性控制

可以通过 SPIx_CR1 的 CPOL 和 CPHA 位用软件选择四种可能的时序关系 CPOL（时钟极性）位控制着在没有数据在发送时的空闲状态的时钟输出电平。该位既针对主机模式也针对从机模式。如果 CPOL 被清零，SCK 引脚在闲置状态输出低电平。如果 CPOL 置 1，SCK 引脚在闲置状态输出高电平。

如果 CPHA 位被置 1，SCK 引脚上的第二沿对准的是第一位的捕获时机（如果 CPOL 位是 0 则为下降沿，如果 CPOL 位为 1 则为上升沿）。每次发生这个时钟切换时，数据被锁存。如果 CPHA 位为 0，SCK 引脚上的第个沿对准第一位的捕获时机（如果 CPOL 位为 1，则为下降沿，如果 CPOL 位为 0，则为上升沿）。每次发生这个时钟切换时，数据被锁存。

CPOL（时钟极性）和 CPHA（时钟相位）的选择共同决定数据采集时的时钟边沿。

下图显示 CPHA 和 CPOL 的四种组合的 SPI 全双工传输。

注：

- 1: 改变的 CPOL/CPHA 位之前，SPI 必须通过清零 SPE 位先关闭。
- 2: 在 SCK 的空闲状态，必须符合在 SPIx_CR1 寄存器中对极性的选择（如果 CPOL=1 上拉 SCK，如果 CPOL=0，下拉 SCK）。

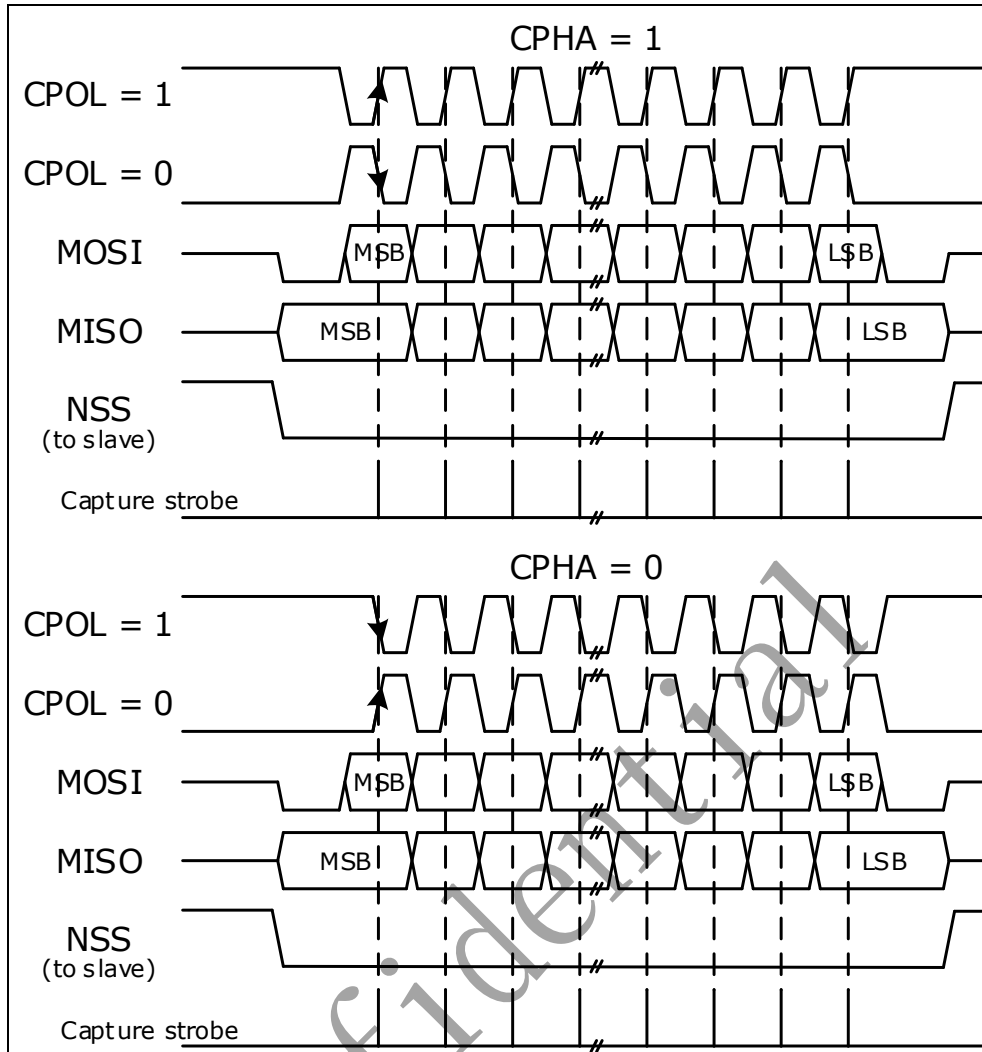


图 20-8SPI 格式

注: 数据位的顺序取决于 LSBFIRST 位的设置。

1. 3. 6. 2 数据帧格式

SPI 移位寄存器可以设置 MSB 在前或 LSB 在前, 取决于的 LSBFIRST 位的值。数据字长使用 SPI_CR1 寄存器的 DFF 位进行选择。它可以设定从 8 位或 16 位的长度, 同时适用于发送和接收。

1. 3. 7 SPI 状态标志

应用程序可通过 3 个状态标志来了解 SPI 总线的全部状态。

1. 3. 7. 1Tx 缓冲器空标志 (TXE)

当它被设置时, TXE 标志表明 Tx 缓冲区是空的, 下一个要传输的数据可以被加载到缓冲区中。TXE 标志通过写入 SPI_DR 寄存器来清除。

1. 3. 7. 2Rx 缓冲非空 (RXNE)

设置时, RXNE 标志指示在 Rx 缓冲区中有有效的接收数据。通过从 SPI_DR 寄存器中读取来清除它。

1.3.7.3 忙标志 (BSY)

BSY 标志由硬件设置和清零（软件改写这个标志是无效的）。

当 BSY 为 1 时，它表示 SPI 正处于数据传输过程中（SPI 总线忙）。在主双向接收模式（MSTR=1 和 BDM=1 和 BDOE=0）中有一个例外，在接收期间 BSY 标志保持较低。

BSY 标志可以在某些模式下用于检测传输的结束，从而防止 SPI 外围时钟在进入低功耗模式或软件处理 NSS 脉冲结束之前被禁用时最后一次传输的损坏。

BSY 标志在防止多主机系统中的写碰撞也是很有用的。BSY 标志在下列条件之一达成时被清除：

- ◆ 当 SPI 被正确禁用时
- ◆ 当在主模式（MODF 位设置为 1）中检测到故障时
- ◆ 在主模式下，当完成了数据发送，并且没有新的数据要发送时
- ◆ 在从模式下，当两次数据传输之间间隔达到至少一个 SPI 时钟周期时

注:建议使用 TXE 和 RXNE 标志（而不是 BSY 标志）来处理数据发送或接收操作

1.3.8 SPI 错误标志

如果下列错误标志被置 1 并且相应中断使能位被置 1，就会产生 SPI 中断。

1.3.8.1 溢出标志 (OVR)

当主机或从机完成下一帧数据接收后，而之前 Rx 缓冲区的前一帧数据未完成读取，就会发生溢出情况(设置了 RXNE 标志)。

溢出的情况发生时，新收到的值不会覆盖 RX 缓冲器的前一个数据。读取 SPI_DR 寄存器得到之前收到的数据，随后传输的所有数据都将丢失。

在读 SPIx_SR 寄存器后，再去读一下 SPIx_DR 寄存器，就会清除 OVR 位。

1.3.8.2 模式故障标志 (MODF)

主机得知其内部 NSS 信号（NSS 引脚为硬件模式，或在 NSS 软件模式 SSI 位）被拉低时，发生模式故障位。这将自动设置 MODF 位。主机模式故障对 SPI 接口的影响包括以下方面：

- ◆ MODF 位被置 1，另外如果 MODFIE 位被置 1，会产生 SPI 中断。
- ◆ SPE 位被清零。这将阻止主机的数据输出，同时禁用 SPI 接口。
- ◆ MSTR 位被清除，从而迫使设备转到从机模式。

使用下面的软件序列，以清除 MODF 位：

1. 在 MODF 位被设置后，对 SPIx_SR 寄存器做一次读或写访问。
2. 然后写一下 SPIx_CR1 寄存器。

为了避免系统中的多个从机发生冲突，NSS 引脚必须在 MODF 位清零过程中拉高。SPE 和 MSTR 位可以在这个清零过程后恢复到原来的状态。作为一种安全措施，硬件不允许在 MODF 位为 1 期间将 SPE 的 MSTR 位置 1。在从模式下，MODF 位永远不可能被置 1，除非是以前的多主机冲突的结果。

1.3.8.3 CRC 错误 (CRCERR)

在 SPIx_CR1 寄存器中的 CRCEN 位为 1 时，这个标志用来确认收到的数据的有效性。如果

接收移位寄存器中的值和接收 SPIx_RXCRCR 值不匹配，SPIx_SR 寄存器中的 CRCERR 标志会被置 1。该标志由软件清除。

1.3.8.4 TI 模式帧格式错误(FRE)

当 SPI 工作在从模式并且配置方式符合 TI 模式协议的时，如果数据通讯期间，在 NSS 上出现一个脉冲时，会引起一次 TI 模式帧格式错误。发生此错误时，SPIx_SR 寄存器中的 FRE 的标志会被置 1。在发生错误时 SPI 不会被禁止，NSS 脉冲会被忽略，SPI 会在开始新的传输前等待下一个 NSS 脉冲。因为错误检测可能会导致两个字节的丢失，数据可能会损坏。

读 SPIx_SR 寄存器，FRE 的标志会被清除。如果 FREIE 位为 1，会在 NSS 错误发生后产生一个中断。在这种情况下，SPI 应该被禁用，因为数据的完整性将不能保证了，通信应该重新开始。先重新使能从机，再重新初始化主机就行了。

1.4 SPI 特殊功能

1.4.1 TI 模式

SPI 接口兼容 TI 协议。SPIx_CR2 寄存器中的 FRF 位可以用来令 SPI 兼容这个协议。

无论 SPIx_CR1 寄存器中怎么设置，时钟极性和相位都会强制为符合 TI 协议要求。NSS 管理也是按照 TI 协议，这使得在这种情况下通过 SPIx_CR1 和 SPIx_CR2 寄存器（SSM 的 SSOE，SSI），不可能配置 NSS 管理。

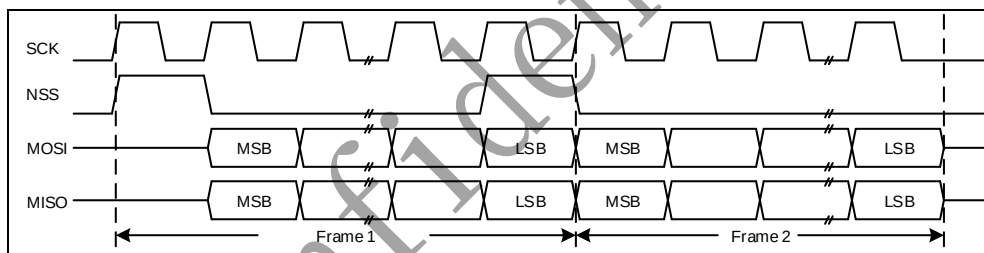


图 20-9 TI SSP format

1.4.2 CRC 计算

有两个独立的 CRC 计算器（在发送和接收数据流程），分别用以检查发送和接收的数据的可靠性。SPI 根据 DFF 位选择的数据格式提供 CRC8 或 CRC16 计算。CRC 是使用 SPI_CRCPR 寄存器中编程的多项式串行计算的。

1.4.2.1 CRC 原则

在使能 SPI 之前（SPE=0）通过设置 SPIx_CR1 寄存器中的 CRCEN 位，使能 CRC 计算。CRC 值由一个奇次可编程多项式按每位计算。计算处理从采样时钟源开始执行，这个时钟沿由 SPIx_CR1 寄存器的 CPHA 和 CPOL 位定义。在 CPU 控制数据块传输结束的同时，计算出的 CRC 结果自动参与检查。当发现 CRC 不匹配现象的时候，CRCERR 标志被置 1，表示 CRC 错误。对 CRC 计算的正确处理过程取决于 SPI 配置和所选择的传输管理。

注:多项式的值应该是奇数。不支持偶数值。

1.4.2.2 CRC 传输管理

在最后一个数据发送或接收完之前，SPI 的启动和传输都没什么特别的。要在当前传输的数据之后跟上 CRC 数据，必须将 SPIx_CR1 中的 CRCNEXT 位设置为 1。设置 CRCNEXT 位的

操作必须在最后一个数据帧交易结束之前完成。在 CRC 数据传输期间，CRC 计算会被冻结。

接收到的 CRC 数据像正常的数据字节或字存储在 RX 缓冲器中。

CRC 数据交换，通常需要在数据序列结束后，再占用一个或多个数据通信的时间。

当收到最后的 CRC 数据后，会自动将收到的值和 `SPIx_RXCRC` 寄存器的值进行比较。软件必须坚持 `SPIx_SR` 寄存器中的 `CRCERR` 标志以判断传输过程中数据是否被破坏。软件对 `CRCERR` 标志写 0 就可清除它。

CRC 数据收到后，被存储在 RX 缓冲器中，必须读 `SPIx_DR` 寄存器以清除 `RXNE` 标志。

1. 4. 2. 3 重置 `SPIx_TXCRC` 和 `SPIx_RXCRC` 值

启用 CRC 计算时，将自动清除 `SPIx_TXCRC` 和 `SPIx_RXCRC` 值。

当 SPI 配置为从模式下启用 CRC 特性时，即使在 NSS 引脚上应用了高级别的 CRC 计算，也会执行 CRC 计算。这种情况可能会发生在例如多从机的环境中，主机交替寻址从机的通信过程。

在禁用从机(NSS 为高)和启用新从机(NSS 为低)之间，应该在主机和从机端清除 CRC 值，以便重新同步主机和从机各自的 CRC 计算。

为了清除 CRC，遵循以下顺序：

1. 禁止 SPI
2. 清除 `CRCEN` 位
3. 使能 `CRCEN` 位
4. 使能 SPI

1. 5 I2S 基本描述

I²S 的结构框图如下图所示：

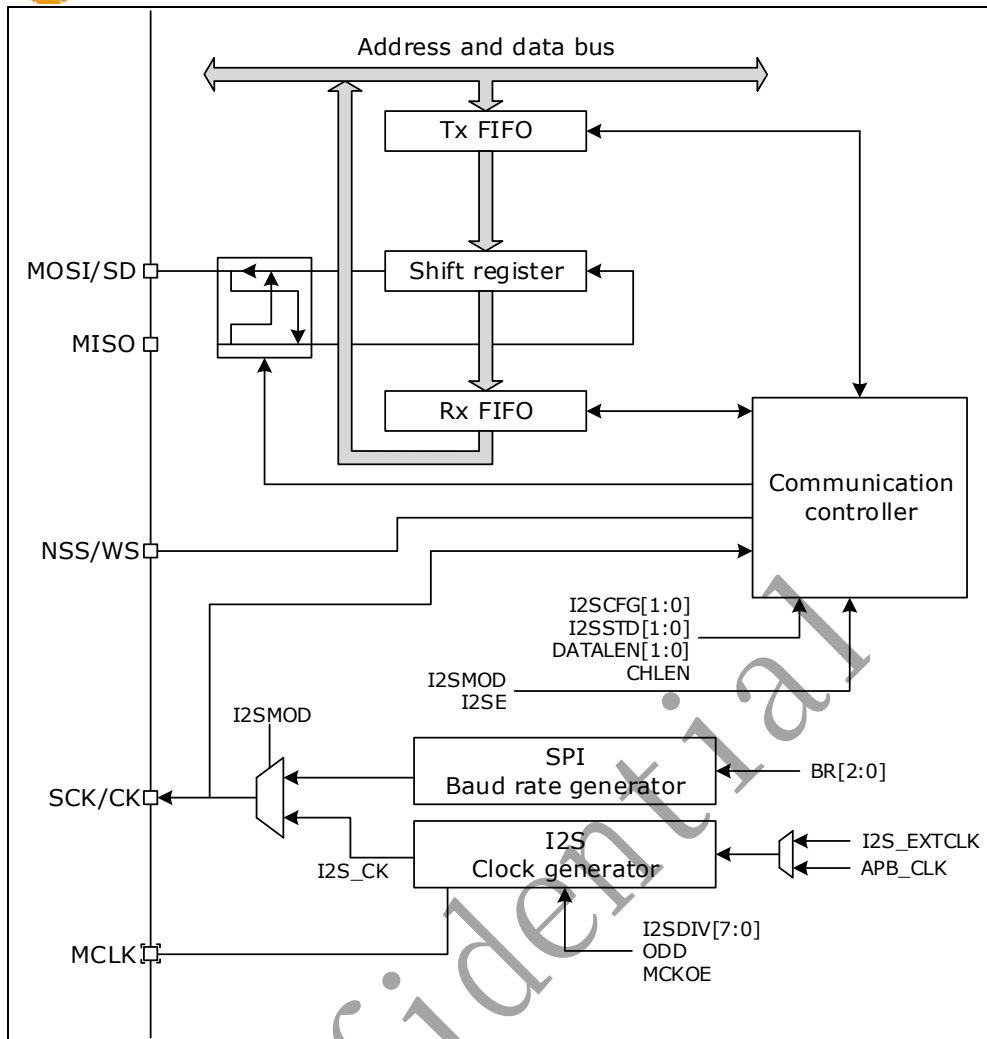


图 20-10 I2S 模块结构

1. MCK 映射在 MISO 引脚。

当启用 I2S 功能时，SPI 可以作为音频 I2S 接口(通过在 SPIx_I2SCFGR 寄存器中设置 I2SMOD 位)。这个接口主要使用与 SPI 相同的引脚、标志和中断。

I2S 和 SPI 共用三个引脚:

- ◆ **SD:** 串行数据 (MOSI 引脚) 发送或接收的两个时分复用数据通道 (仅在单工模式下)。
- ◆ **WS:** 字选择 (NSS 引脚) 主模式下作为数据控制信号输出，从动模式下作为输入。
- ◆ **CK:** 串行时钟 (SCK 引脚) 主模式下作为串行时钟输出，从模式下为输入。

在某些外部音频设备需要主时钟时，可以另有一个附加引脚输出时钟:

- ◆ **MCK:** 主时钟(独立映射)，在 I2S 配置为主模式，寄存器 SPI_I2SPR 的 MCKOE 位为 '1' 时，作为输出额外的时钟信号引脚使用。输出时钟信号的频率预先设置为 $256 \times F_s$ ，其中 F_s 是音频信号的采样频率

设置成主模式时，I2S 使用自身的时钟发生器来产生通信用的时钟信号这个时钟发生器也是主时钟输出的时钟源。I2S 模式下有 2 个额外的寄存器，一个是与时钟发生器配置相关的寄存器

SPI_I2SPR, 另一个是 I2S 通用配置寄存器 SPI_I2SCFGR(可设置音频标准、从/主模式、数据格式、数据包帧、时钟极性参数)。

在 I2S 模式下不使用寄存器 SPI_CR1 和所有的 CRC 寄存器。同样, I2S 模式下也不使用寄存器 SPI_CR2 的 SSOE 位, 和寄存器 SPI_SR 的 MODF 位和 CRCERR 位。

I2S 使用与 SPI 相同的寄存器 SPI_DR 用作 16 位宽模式数据传输。

1. 5. 1 I2S 全双工

下图显示了如何使用两个 SPI2S 实例执行全双工通信。在这种情况下, 两个 spi2 的 WS 和 CK IOs 必须连接在一起。

对于主全双工模式, 一个 SPI2S 块必须在主模式下编程(I2SCFG = '10'或'11'), 另一个 SPI2S 块必须在从模式下编程(I2SCFG = '00'或'01')。可以生成 MCK, 也可以不生成 MCK, 这取决于应用程序的需要。

对于从机的全双工模式, 两个 SPI2S 块必须在从机中编程。其中一个在从属接收器(I2SCFG = '01'), 另一个在从属发射机(I2SCFG = '00')。主外部设备然后提供位时钟(CK)和帧同步(WS)。

请注意, 全双工模式可以用于所有支持的标准:I2S 飞利浦, MSB 认证, LSB 认证和 PCM。

对于全双工模式, 两个 SPI2S 实例必须使用相同的标准, 具有相同的参数:I2SMOD、I2SSTD、CKPOL、PCMSYNC、DATLEN 和 CHLEN 必须在两个实例上包含相同的值。

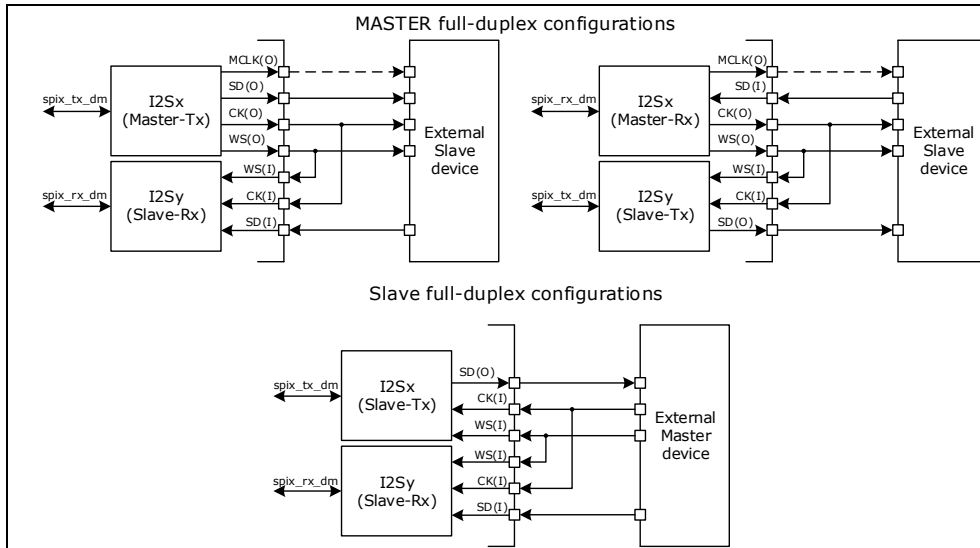


图 20-11 I2S 全双工通信

1. 5. 2 支持的音频协议

三线总线支持 2 个声道上音频数据的时分复用：左声道和右声道。但是只有一个 16 位寄存器用作发送或接收。因此，软件必须在对数据寄存器写入数据时，根据当前传输中的声道写入相应的数据；同样，在读取寄存器数据时，通过检查寄存器 SPI_SR 的 CHSIDE 位来判明接收到的数据属于哪个声道。左声道总是先于右声道发送数据(CHSIDE 位在 PCM 协议下无意义)。

有四种可用的数据和包帧组合。可以通过以下四种数据格式发送数据：

- ◆ 16 位数据打包进 16 位帧
- ◆ 16 位数据打包进 32 位帧
- ◆ 24 位数据打包进 32 位帧
- ◆ 32 位数据打包进 32 位帧

在使用 16 位数据扩展到 32 位帧时，前 16 位(MSB)是有意义的数字，后 16 位(LSB)被强制为 0，该操作不需要软件干预，也不需要 DMA 请求(仅需要一次读/写操作)。

24 位和 32 位数据帧需要 CPU 对寄存器 SPI_DR 进行 2 次读或写操作，在使用 DMA 时，需要 2 次 DMA 传输。对于 24 位数据，扩展到 32 位后，最低 8 位由硬件置 0。

对于所有的数据格式和通讯标准，总是先发送最高位(MSB)。

I2S 接口支持四种音频标准，可以通过设置寄存器 SPI_I2SCFGR 的 I2SSTD[1:0]位和 PCMSY 位来选择。

1. 5. 2. 1 I2S 飞利浦标准

在此标准下，引脚 WS 用来指示正在发送的数据属于哪个声道。在发送第一位数据(MSB)前 1 个时钟周期，该引脚即为有效。

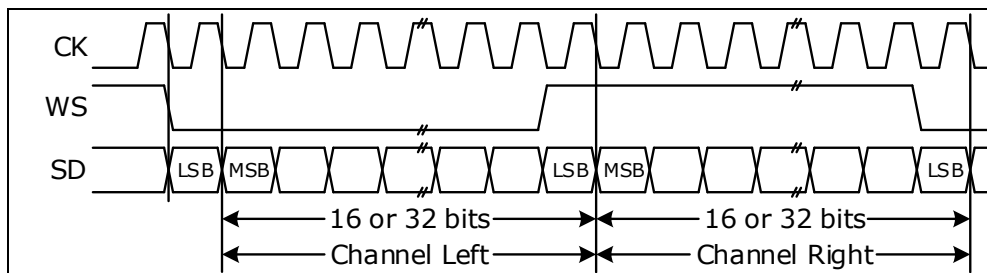


图 20-12 I²S 飞利浦协议波形(16/32 位全精度, CPOL = 0)

发送方在时钟信号(CK)的下降沿改变数据, 接收方在上升沿读取数据。WS 信号也在时钟信号的下降沿锁存。

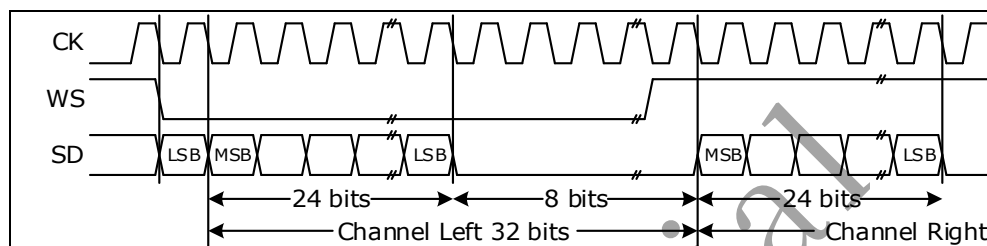


图 20-13 I²S 飞利浦协议标准波形(24 位帧, CPOL = 0)

次模式需要对寄存器 SPI_DR 进行 2 次读或写操作

◆ 在发送模式下:

如果要发送 0x8EAA33(24 位):

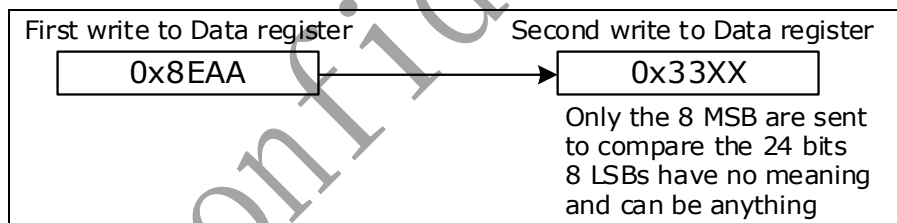


图 20-14 发送 0x8EAA33

◆ 在接收模式下:

如果接收 0x8EAA33:

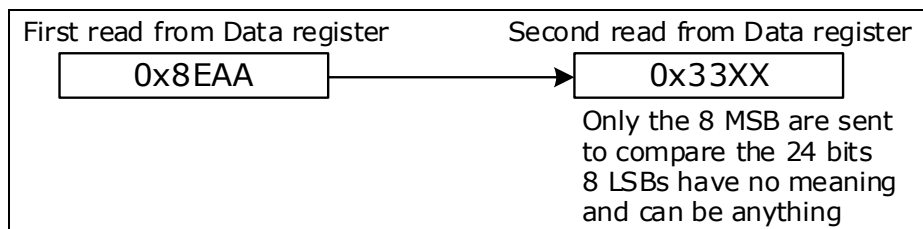


图 20-15 接收 0x8EAA33

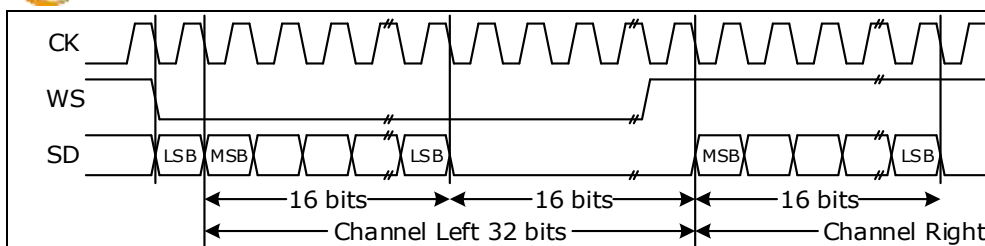


图 20-16 I2S 飞利浦协议标准波形(16 位扩展至 32 位包帧, CPOL = 0)

在 I2S 配置阶段, 如果选择将 16 位数据扩展到 32 位声道帧, 只需要访问一次寄存器 SPI_DR。用来扩展到 32 位的低 16 位被硬件置为 0x0000。

如果待传输或者接收的数据是 0x76A3(扩展到 32 位是 0x76A30000), 需要的操作如下图所示。

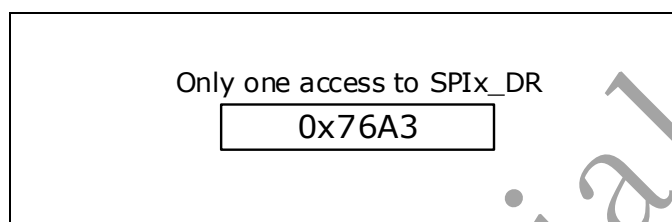


图 20-17 16 位扩展至 32 位包帧的例子

在发送时需要将 MSB 写入寄存器 SPI_DR; 标志位 TXE 为 '1' 表示可以写入新的数据, 如果允许了相应的中断, 则可以产生中断。发送是由硬件完成的, 即使还未发送出后 16 位的 0x0000, 也会设置 TXE 并产生相应的中断。

接收时, 每次收到高 16 位半字(MSB)后, 标志位 RXNE 置 '1', 如果允许了相应的中断, 则可以产生中断。

这样, 在 2 次读和写之间有更多的时间, 可以防止下溢或者上溢的情况发生。

1. 5. 2. 2MSB 对齐标准

在此标准下, WS 信号和第一个数据位, 即最高位(MSB)同时产生。

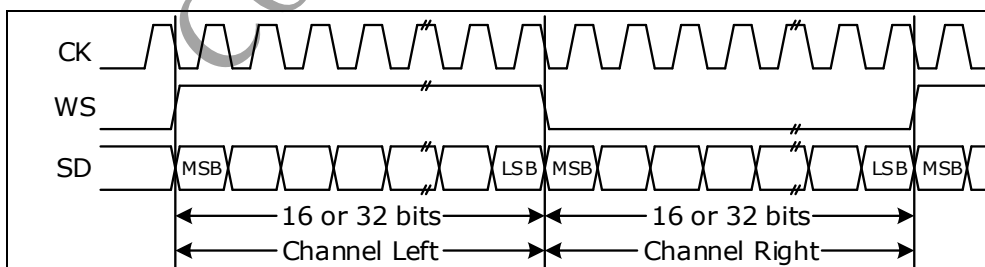


图 20-18MSB 对齐 16 位或 32 位全精度, CPOL = 0

发送方在时钟信号的下降沿改变数据; 接收方是在上升沿读取数据。

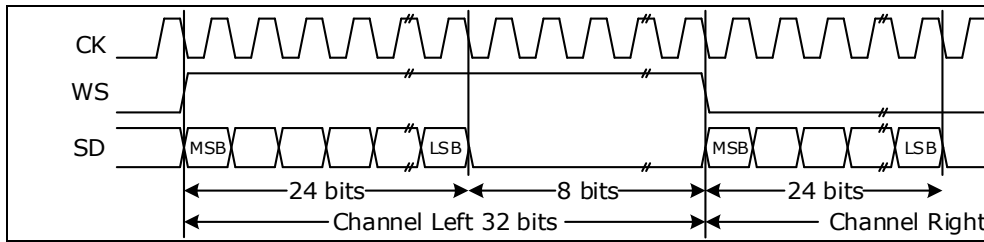


图 20-19MSB 对齐 24 位数据，CPOL = 0

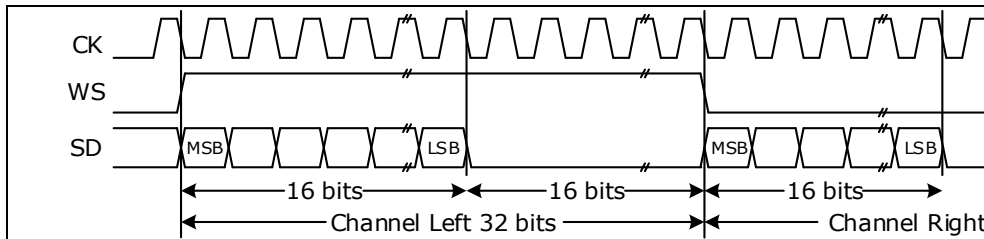


图 20-20MSB 对齐 16 位数据扩展到 32 位包帧，CPOL = 0

1. 5. 2. 3LSB 对齐标准

此标准与 MSB 对齐标准类似(在 16 位或 32 位全精度帧格式下无区别)。

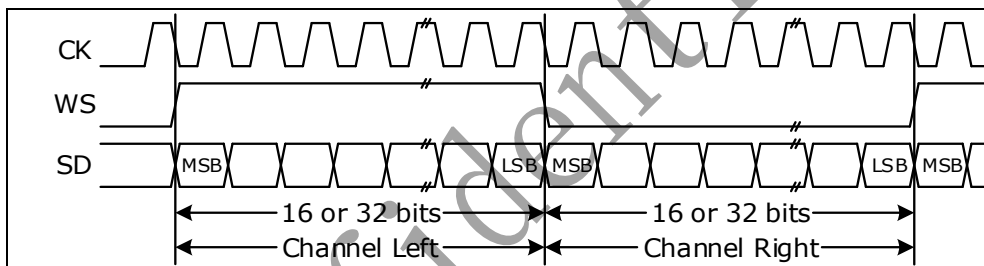


图 20-21LSB 对齐 16 位或 32 位全精度，CPOL = 0

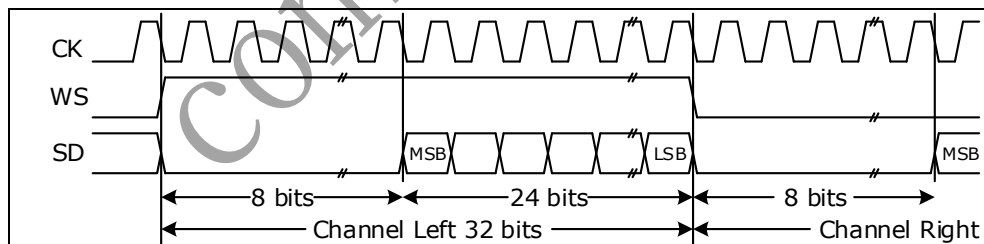


图 20-22LSB 对齐 24 位数据，CPOL = 0

◆ 在发送模式下:

如果要发送数据 0x3478AE，需要通过软件或者 DMA 对寄存器 SPI_DR 进行 2 次写操作。操作流程如下图所示。

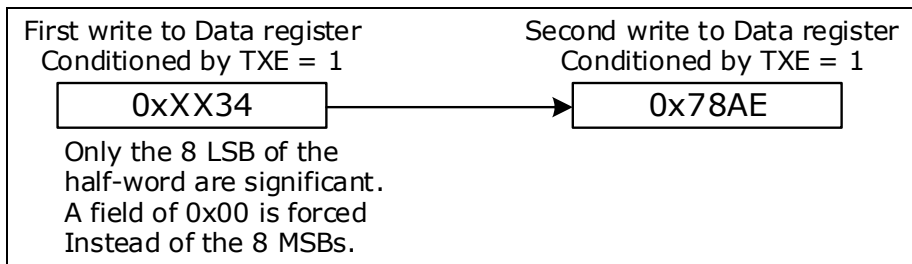


图 20-23 发送 0x3478AE 要求的操作

◆ 在接收模式下:

如果要接收数据 0x3478AE，需要在 2 个连续的 RXNE 事件发生时，分别对寄存器 SPI_DR 进行 2 次读操作。

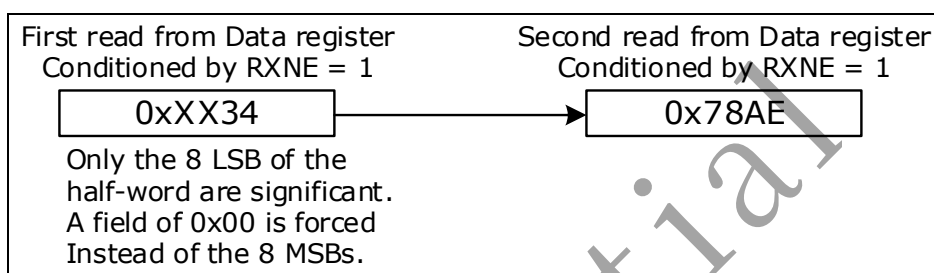


图 20-24 接收 0x3478AE 要求的操作

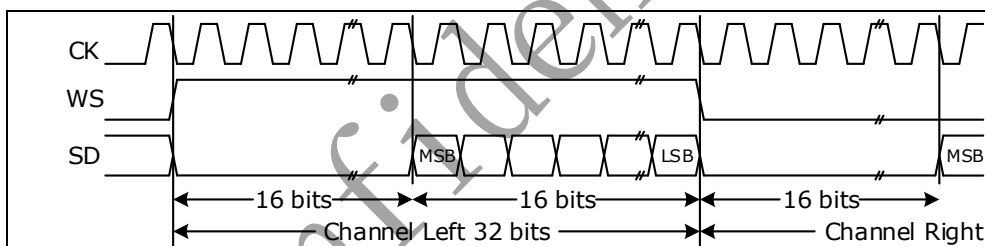


图 20-25 LSB 对齐 16 位数据扩展到 32 位包帧，CPOL = 0

在 I2S 配置阶段，如果选择将 16 位数据扩展到 32 位声道帧，只需要访问一次寄存器 SPI_DR。用来扩展到 32 位的低 16 位被硬件置为 0x0000。在这种情况下，它对应的半字的 MSB。

如果待传输或者接收的数据是 0x76A3(扩展到 32 位是 0x000076A3)，需要的操作如下图所示。

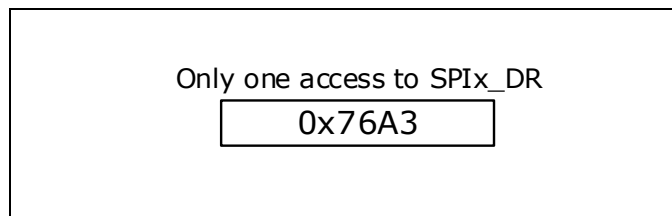


图 20-26 16 位扩展至 32 位包帧的例子

在发送时，如果 TXE 为 ‘1’，用户需要写入待发送的数据(即 0x76A3)。用来扩展到 32 位的 0x0000 部分由硬件首先发送出去，一旦有效数据开始从 SD 引脚送出，即发生下一次 TXE 事

件。

在接收时，一旦接收到有效数据(而不是 0x0000 部分)，即发生 RXNE 事件。

这样，在 2 次读和写之间有更多的时间，可以防止下溢或者上溢的情况发生。

1.5.2.4 PCM 标准

在 PCM 标准下，不存在声道选择的信息。PCM 标准有 2 种可用的帧结构，短帧或者长帧，可以通过设置寄存器 SPI_I2SCFGR 的 PCMSYNC 位来选择。

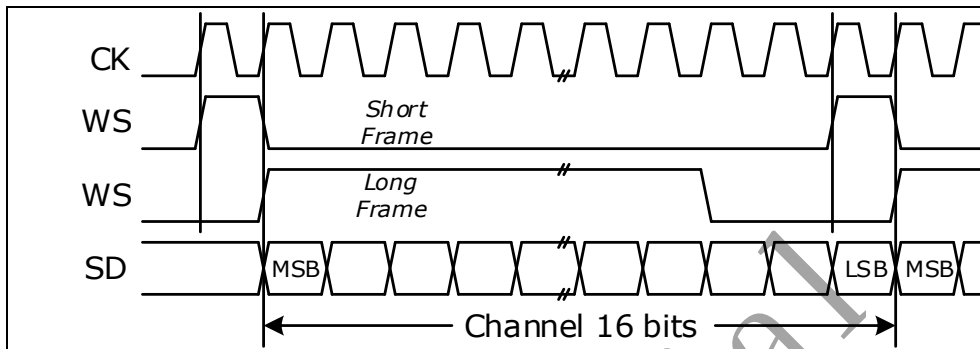


图 20-27 PCM 标准波形(16 位)

对于长帧，主模式下，用来同步的 WS 信号有效的时间固定为 13 位。

对于短帧，用来同步的 WS 信号长度只有 1 位。

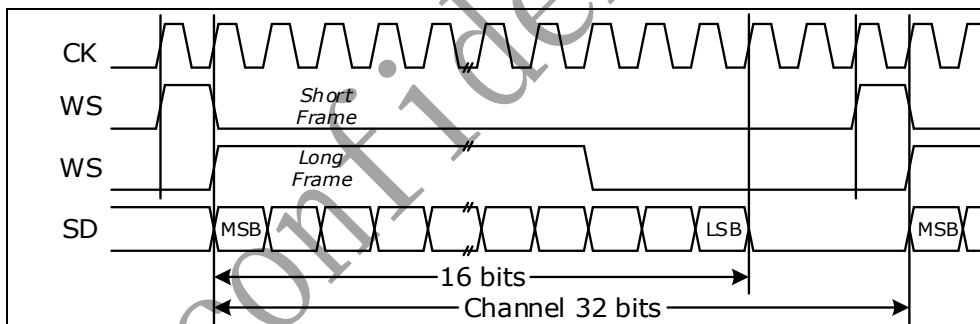


图 20-28 PCM 标准波形(16 位扩展到 32 位包帧)

注：无论哪种模式(主或从)、哪种同步方式(短帧或长帧)，连续的 2 帧数据之间和 2 个同步信号之间的时间差，(即使是从模式)需要通过设置 SPI_I2SCFGR 寄存器的 DATLEN 位和 CHLEN 位来确定。

1.5.3 时钟发生器

I2S 的比特率即确定了在 I2S 数据线上的数据流和 I2S 的时钟信号频率。

I2S 比特率=每个声道的比特数×声道数目×音频采样频率

对于一个具有左右声道和 16 位音频信号，I2S 比特率计算如下：

I2S 比特率= $16 \times 2 \times f_s$

如果包长为 32 位，则有：I2S 比特率= $32 \times 2 \times f_s$

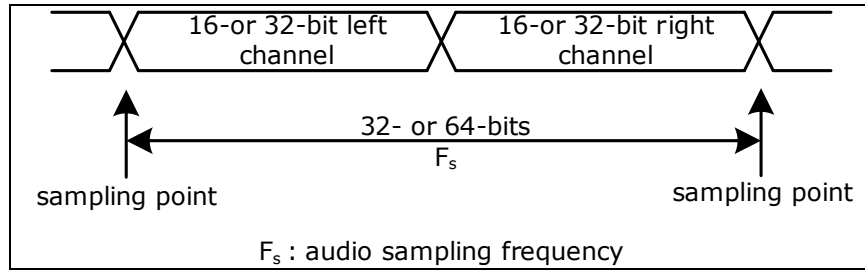


图 20-29 音频采样频率定义

在主模式下，为了获得需要的音频频率，需要正确地对线性分频器进行设置。

下图展示了通信时钟架构。I2Sx 时钟是系统时钟。

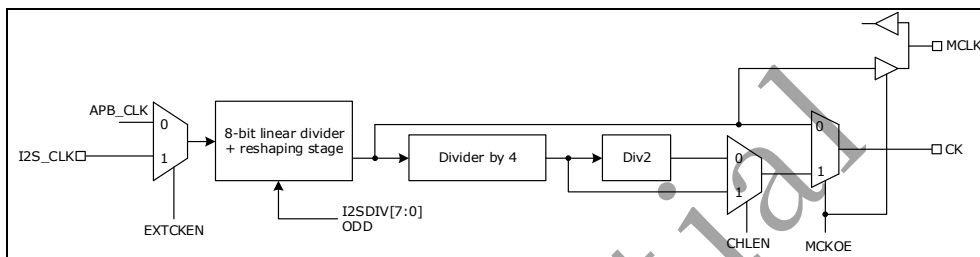


图 20-30 I²S clock generator architecture

音频的采样频率可以是 192 KHz, 96 kHz, 48 kHz, 44.1 kHz, 32 kHz,

22.05 kHz, 16 kHz, 11.025 kHz or 8 kHz (或任何此范围内的数值). 为了获得需要的频率，需按照以下公式设置线性分频器：

当需要生成主时钟时(寄存器 SPI_I2SPR 的 MCKOE 位为 ' 1 ')：

$$\text{声道的帧长为 16 位时, } f_s = \text{I2SxCLK} / [(16 \times 2) \times ((2 \times \text{I2SDIV}) + \text{ODD}) \times 8]$$

$$\text{声道的帧长为 32 位时, } f_s = \text{I2SxCLK} / [(32 \times 2) \times ((2 \times \text{I2SDIV}) + \text{ODD}) \times 4]$$

当关闭主时钟时(MCKOE 位为 ' 0 ')：

$$\text{声道的帧长为 16 位时, } f_s = \text{I2SxCLK} / [(16 \times 2) \times ((2 \times \text{I2SDIV}) + \text{ODD})]$$

$$\text{声道的帧长为 32 位时, } f_s = \text{I2SxCLK} / [(32 \times 2) \times ((2 \times \text{I2SDIV}) + \text{ODD})]$$

下表给出了不同时钟配置时，精确参数的例子。

注： 可以使用其它配置以达到优化时钟精确度的目的。

SYSClk (MHz)	Data length	I2SDIV	I2SODD	MCLK	Target fs(Hz)	Real fs (Hz)	Error
32	16	5	0	No	96000	100000	4.1667%
32	32	2	0	No	96000	100000	4.1667%
32	16	10	1	No	48000	47619	0.7937%
32	32	5	0	No	48000	50000	4.1667%

32	16	11	1	No	44100	43478	1.4098%
32	32	5	1	No	44100	45454	3.0715%
32	16	15	1	No	32000	32258	0.8065%
32	32	8	0	No	32000	31250	2.3430%
32	16	22	1	No	22050	22222	0.7811%
32	32	11	1	No	22050	21739	1.4098%
32	16	31	1	No	16000	15873	0.7937%
32	32	15	1	No	16000	16129	0.8065%
32	16	45	1	No	11025	10989	0.3264%
32	32	22	1	No	11025	11111	0.7811%
32	16	62	1	No	8000	8000	0.0000%
32	32	31	1	No	8000	7936	0.7937%
32	16	2	0	Yes	32000	31250	2.3430%
32	32	2	0	Yes	32000	31250	2.3430%
32	16	3	0	Yes	22050	20833	5.5170%
32	32	3	0	Yes	22050	20833	5.5170%
32	16	4	0	Yes	16000	15625	2.3428%
32	32	4	0	Yes	16000	15625	2.3428%
32	16	5	1	Yes	11025	11363	3.0715%
32	32	5	1	Yes	11025	11363	3.0715%
32	16	8	0	Yes	8000	7812	2.3428%
32	32	8	0	Yes	8000	7812	2.3428%

表 20-1 使用标准的 8 MHz HSE 时钟得到精确的音频频率

1. 5. 4 I2S 主模式

设置 I2S 工作在主模式，串行时钟由引脚 CK 输出，字选信号由引脚 WS 产生。可以通过设置寄存器 SPI_I2SPR 的 MCKOE 位来选择输出或者不输出主时钟(MCK)。

1. 5. 4. 1 流程

1. 设置寄存器 SPI_I2SPR 的 I2SDIV[7:0] 定义与音频采样频率相符的串行时钟波特率。同时也要定义寄存器 SPI_I2SPR 的 ODD 位。
2. 设置 CKPOL 位定义通信用时钟在空闲时的电平状态。如果需要向外部的 DAC/ADC 音频器件提供主时钟 MCK，将寄存器 SPI_I2SPR 的 MCKOE 位置为' 1'。(按照不同的 MCK 输出状态，计算 I2SDIV 和 ODD 的值)。
3. 设置寄存器 SPI_I2SCFGR 的 I2SMOD 位为' 1' 激活 I2S 功能，设置 I2SSTD[1:0] 和 PCMSYNC 位选择所用的 I2S 标准，设置 CHLEN 选择每个声道的数据位数。还要设置寄存器 SPI_I2SCFGR 的 I2SCFG[1:0] 选择 I2S 主模式和方向(发送端还是接收端)。
4. 如果需要，可以通过设置寄存器 SPI_CR2 来打开所需的 interrupt 功能和 DMA 功能。
5. 必须将寄存器 SPI_I2SCFGR 的 I2SE 位置为' 1'。引脚 WS 和 CK 需要配置为输出模式。

如果寄存器 SPI_I2SPR 的 MCKOE 位为' 1'，引脚 MCK 也要配置成输出模式。

1. 5. 4. 2 发送流程

当写入 1 个半字(16 位) 的数据至发送缓存，发送流程开始。

假设第一个写入发送缓存的数据对应的是左声道数据。当数据从发送缓存移到移位寄存器时，标志位 TXE 置' 1'，这时，要把对应右声道的数据写入发送缓存。标志位 CHSIDE 提示了目前待传输的数据对应哪个声道。标志位 CHSIDE 的值在 TXE 为' 1' 时更新，因此它在 TXE 为' 1' 时有意义。

在先左声道后右声道的数据都传输完成后，才能被认为是一个完整的数据帧。不可以只传输部分数据帧，如仅有左声道的数据。

当发出第一位数据的同时，半字数据被并行地传送至 16 位移位寄存器，然后后面的位依次按高位在前的顺序从引脚 MOSI/SD 发出。每次数据从发送缓存移至移位寄存器时，标志位 TXE 置为' 1'，如果寄存器 SPI_CR2 的 TXEIE 位为' 1'，则产生中断。写入数据的操作取决于所选择的 I2S 标准，详见 25.6.2 节。

为了保证连续的音频数据传输，建议在当前传输完成之前，对寄存器 SPI_DR 写入下一个要传输的数据。

建议在要关闭 I2S 功能时，等待标志位 TXE=1 及 BSY=0，再将 I2SE 位清' 0'。

1. 5. 4. 3 接收流程

接收流程的配置步骤除了第 3 点外，与发送流程的一致(参见前述的" 发送流程")，需要通过配置 I2SCFG[1:0] 来选择主接收模式。

无论何种数据和声道长度，音频数据总是以 16 位包的形式接收。即每次填满接收缓存后，标志位 RXNE 置' 1'，如果寄存器 SPI_CR2 的 RXNEIE 位为' 1'，则产生中断。根据配置的数据和声道长度，收到左声道或右声道的数据会需要 1 次或者 2 次把数据传送到接收缓存的过程。

对寄存器 SPI_DR 进行读操作即可清除 RXNE 标志位。

每次接收以后即更新 CHSIDE。它的值取决于 I2S 单元产生的 WS 信号。

读取数据的操作取决于所选择的 I2S 标准，详见 26.6.2 节。

如果前一个接收到的数据还没有被读取，又接收到新数据，即发生上溢，标志位 OVR 被置为' 1'，如果寄存器 SPI_CR2 的 ERRIE 位为' 1'，则产生中断，表示发生了错误。

1. 5. 5 I2S 从模式

在从模式下，I2S 可以设置成发送和接收模式。从模式的配置方式基本遵循和配置主模式一样的流程。在从模式下，不需要 I2S 接口提供时钟。时钟信号和 WS 信号都由外部主 I2S 设备提供，连接到相应的引脚上。因此用户无需配置时钟。

配置步骤列举如下：

1. 设置寄存器 SPI_I2SCFGR 的 I2SMOD 位激活 I2S 功能；设置 I2SSTD[1:0] 来选择所用的 I2S 标准；设置 DATLEN[1:0] 选择数据的比特数；设置 CHLEN 选择每个声道的数据位数。设置寄存器 SPI_I2SCFGR 的 I2SCFG[1:0] 选择 I2S 从模式的数据方向(发送端还是接收端)。
2. 根据需要，设置寄存器 SPI_CR2 打开所需的中断功能和 DMA 功能。
3. 必须设置寄存器 SPI_I2SCFGR 的 I2SE 位为 '1'。

1. 5. 5. 1 发送流程

当外部主设备发送时钟信号，并且当 NSS_WS 信号请求传输数据时，发送流程开始。必须先使能从设备，并且写入 I2S 数据寄存器之后，外部主设备才能开始通信。

对于 I2S 的 MSB 对齐和 LSB 对齐模式，第一个写入数据寄存器的数据项对应左声道的数据。当开始通信时，数据从发送缓冲器传送到移位寄存器，然后标志位 TXE 置为 '1'；这时，要把对应右声道的数据项写入 I2S 数据寄存器。

标志位 CHSIDE 提示了目前待传输的数据对应哪个声道。与主模式的发送流程相比，在从模式中，CHSIDE 取决于来自外部主 I2S 的 WS 信号。这意味着从 I2S 在接收到主端生成时钟信号之前，就要准备好第一个要发送的数据。WS 信号为 '1' 表示先发送左声道。

注：设置 I2SE 位为 '1' 的时间，应当比 CK 引脚上的主 I2S 时钟信号早至少 2 个 PCLK 时钟周期。

当发出第一位数据的时候，半字数据并行地通过 I2S 内部总线传输至 16 位移位寄存器，然后其它位依次按高位在前的顺序从引脚 MOSI/SD 发出。每次数据从发送缓冲器传送到移位寄存器时，标志位 TXE 置 '1'，如果寄存器 SPI_CR2 的 TXEIE 位为 '1'，则产生中断。

注意，在对发送缓冲器写入数据前，要确认标志位 TXE 为 '1'。写入数据的操作取决于所选中的 I2S 标准，详见 25.6.2 节。

为了保证连续的音频数据传输，建议在当前传输完成之前，对寄存器 SPI_DR 写入下一个要传输的数据。如果在代表下一个数据传输的第一个时钟边沿到达之前，新的数据仍然没有写入寄存器 SPI_DR，下溢标志位会置 '1'，并可能产生中断；它指示软件发送数据错误。

如果寄存器 SPI_CR2 的 ERRIE 位为 '1'，在寄存器 SPI_SR 的标志位 UDR 为高时，就会产生中断。建议在这时关闭 I2S，然后重新从左声道开始发送数据。建议在清除 I2SE 位关闭 I2S 之前，先等待 TXE=1 并且 BSY=0。.

1. 5. 5. 2 接收流程

配置步骤除了第 1 点（参见 20.6.5.1 小结）外，与发送流程一致。需要通过配置 SPIx_I2SCFGR 寄存器中的 I2SCFG[1:0] 来选择主接收模式。

无论何种数据和声道长度，音频数据总是以 16 位包的形式接收，即每次填满接收缓存，标志位 RXNE 置 '1'，如果寄存器 SPI_CR2 的 RXNEIE 位为 '1'，则产生中断。按照不同的数据和声道长度设置，收到左声道或者右声道数据会需要 1 次或者 2 次传输数据至接收缓冲器

的过程。

每次接收到数据(将要从 SPI_DR 读出) 以后即更新 CHSIDE, 它对应 I2S 单元产生的 WS 信号。读取 SPI_DR 寄存器, 将清除 RXNE 位。

读取数据的操作取决于所选中的 I2S 标准, 详见 25.6.2 节。

在还没有读出前一个接收到的数据, 又接收到新数据时, 即产生上溢, 并设置标志位 OVR 为' 1'; 如果寄存器 SPI_CR2 的 ERRIE 位为' 1', 则产生中断, 指示发生了错误。

要关闭 I2S 功能时, 需要在接收到最后一次 RXNE=1 时将 I2SE 位清' 0'。

注: 外部主 I2S 器件需要有通过音频声道发送/ 接收 16 位或 32 位数据包的功能。

1. 5. 6 I2S 状态标志位

应用程序可通过 3 个状态标志来了解 I2S 总线的全部状态。

1. 5. 6. 1 忙标志位(BSY)

BSY 标志由硬件设置和清零 (软件改写这个标志是没有用的)。它指示 I2S 通信层的状态。

该位为' 1' 时表明 I2S 通讯正在进行中, 但有一个例外: 主接收模式(I2SCFG=11) 下, 在接收期间 BSY 标志始终为低。

在软件要关闭 I2S 模块之前, 可以使用 BSY 标志检测传输是否结束, 这样可以避免破坏最后一次传输, 因此需要严格按照下述过程执行。

当传输开始时, BSY 标志被置为' 1', 除非 I2S 模块处于主接收模式。

下述情况时, 该标志位被清除:

- ◆ 当传输结束时 (除了主发送模式, 这种模式下通信是连续的)
- ◆ 当关闭 I2S 模块时。

当通信是连续的时候:

- ◆ 在主发送模式时, 整个传输期间, BSY 标志始终为高;
- ◆ 在从模式时, 每个数据项传输之间, BSY 标志在 1 个 I2S 时钟周期内变低。

注: 不要使用 BSY 标志处理每一个数据项的发送和接收, 最好使用 TXE 和 RXNE 标志。

1. 5. 6. 2 Tx 缓冲空标志 (TXE)

该标志位为' 1' 表示发送缓冲器为空, 可以对发送缓冲器写入新的待发送数据。在发送缓冲器中已有数据时, 标志位清' 0'。在 I2S 被关闭时(I2SE 位为' 0'), 该标志位也为' 0'。

1. 5. 6. 3 Rx 缓冲非空 (RXNE)

该标志位置' 1' 表示在接收缓存里有接收到的有效数据。在读取寄存器 SPI_DR 时, 该位清' 0'。

1. 5. 6. 4 声道标志位(CHSIDE)

在发送模式下, 该标志位在 TXE 为高时刷新, 指示从 SD 引脚上发送的数据所在的声道。
如果在从发送模式下发生了下溢错误, 该标志位的值无效, 在重新开始通讯前需要把 I2S 关闭再打开。

在接收模式下, 该标志位在寄存器 SPI_DR 接收到数据时刷新, 这指示接收到的数据所在的声道。注意, 如果发生错误(如上溢 OVR), 该标志位无意义, 需要将 I2S 关闭再打开(同时, 如果必要修改 I2S 的配置)。

在 PCM 标准下, 无论短帧格式还是长帧格式, 这个标志位都没有意义。

如果寄存器 SPI_SR 的标志位 OVR 或 UDR 为' 1', 且寄存器 SPI_CR2 的 ERRIE 位为' 1', 则会产生中断。(中断源已经被清除后)可以通过读寄存器 SPI_SR 来清除中断标志。

1. 5. 7 I2S 错误标志

I2S 单元有 3 个错误标志位。

1. 5. 7. 1 下溢标志 (UDR)

在从发送模式下, 如果数据传输的第一个时钟边沿到达时, 新的数据仍然没有写入 SPI_DR 寄存器, 该标志位会被置' 1'。在寄存器 SPI_I2SCFGR 的 I2SMOD 位置' 1' 后, 该标志位才有效。如果寄存器 SPI_CR2 的 ERRIE 位为' 1', 就会产生中断。

通过对寄存器 SPI_SR 进行读操作来清除该标志位。

1. 5. 7. 2 上溢标志(OVR)

如果还没有读出前一个接收到的数据时, 又接收到新的数据, 即产生上溢, 该标志位置' 1', 因此, 输入的数据都将丢失。如果寄存器 SPI_CR2 的 ERRIE 位为' 1', 则产生中断指示发生了错误。这时, 接收缓存的内容, 不会刷新为从发送设备送来的新数据。对寄存器 SPI_DR 的读操作返回最后一个正确接收到的数据。其他所有在上溢发生后由发送设备发出的 16 位数据都会丢失。

通过先读寄存器 SPI_SR 再读寄存器 SPI_DR, 来清除该标志位。

1. 5. 7. 3 帧错误标志 (FRE)

仅当 I2S 配置为从模式时, 硬件才能设置此标志。如果外部主服务器正在更改 WS 行, 而从服务器不希望进行此更改, 则设置此属性。如果同步丢失, 则需要执行以下步骤才能从这种状态中恢复, 并将外部主设备与 I2S 从设备重新同步:

1. 禁用 I2S
2. 当在 WS 线上检测到正确的电平时再次启用它(WS 线在 I2Smode 中是高电平, 而在 MSB 或 lsb -或 PCM 模式中则是低电平。

主设备和从设备之间的不同步可能是由于 SCK 通信时钟或 WS 帧同步线上的噪声环境造成的。如果设置 ERRIE 位, 则会产生错误中断。读取状态寄存器时, 软件会清除去同步标志(FRE)。

1.6 SPI 寄存器描述

可以用半字（16 位）或字（32 位）的方式操作这些外设寄存器。此外 SPI_DR 可以用 8 位的方式访问。

1.6.1 SPI 控制寄存器 1 (SPI_CR1)

		Address offset:		0x00											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BIDI MODE	BIDI OE	CRCE N	CRCN EXT	CRCL	RXO NLY	SSM	SSI	LSBFI RST	SPE	BR[2:0]			MST R	CPOL	CPHA

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
BIDIMODE	15	R/W	0	BIDIMODE: Bidirectional data mode enable. 0: 选择 2 线的单向数据模式 1: 选择单线双向数据模式 注：该位在 I2S 模式下没有用
BIDIOE	14	R/W	0	BIDIOE: Output enable in bidirectional mode 和 BIDIMODE 位一起决定在“单线双向”模式下数据的输出方向 0: 输出禁止(只收模式); 1: 输出使能(只发模式)。 注：1. 在主模式下，用 MOSI 引脚。在从模式下，用 MISO 引脚。2. 该位在 I2S 模式下没有用
CRCEEN	13	R/W	0	CRCEEN: Hardware CRC calculation enable 0: CRC 计算禁用 1: CRC 计算使能 注：1. 只有当 SPI 被禁止，该位才可以被写入。
CRCNEXT	12	R/W	0	CRCNEXT: Transmit CRC next 0: 下一个发送的值来自发送缓冲区。 1: 下一个发送的值来自发送 CRC 寄存器。 注：1. 在 SPI_DR 寄存器写入最后一个数据后应立即设置该位。 2. 该位在 I2S 模式下没有用
CRCL	11	R/W	0	CRCL: CRC length 由软件设置和清零，用于选择 CRC 长度。 0: 8 位 CRC 长度 1: 16 位 CRC 长度 注：1. 只有当 SPI 被禁止，该位才可以被写入。2. 该位在 I2S 模式下没有用
RXONLY	10	R/W	0	RXONLY: Receive only mode enabled. 和 BIDIMODE 位一起决定在“2 线单向”模式下数据的输出方向。在多个从设备的配置中，在未被访问的从设备上该位被置 1，使得只有被访问的从设备有输出，从而不会造成数据线上数据冲突。 0: 全双工（发送和接收） 1: 输出禁止(只收模式); 注：该位在 I2S 模式下没有用

SSM	9	R/W	0	SSM: Software slave management 当SSM被置位时，NSS引脚上的电平由SSI位的值决定。 0: 禁止软件从设备管理; 1: 启用软件从设备管理 注：在I2S模式和SPI TI模式下，这个位不可用。
SSI	8	R/W	0	SSI: Internal slave select 此位只有当SSM位为1的时候才有效果。它决定了NSS上的电平，在NSS引脚上的I/O操作无效。 注：在I2S模式和SPI TI模式下，这个位不可用。
LSBFIRST	7	R/W	0	LSBFIRST: Frame format 0: 先发送MSB; 1: 先发送LSB。 注：1. 当通信在进行时不能改变该位的值。2. 在I2S模式和SPI TI模式下，这个位不可用。
SPE	6	R/W	0	SPE: SPI enable 0: 禁止SPI设备; 1: 开启SPI设备。 注：1. 该位在I2S模式下没有用 2. 当关闭SPI设备时，请按照第642页的过程操作。
BR[2:0]	5:3	R/W	0x0	BR[2:0]: Baud rate control 000: 无效 001: 无效 010: fPCLK/8 011: fPCLK/16 100: fPCLK/32 101: fPCLK/64 110: fPCLK/128 111: fPCLK/256 注：当通信在进行时不能改变该位的值。该位在I2S模式下没有用
MSTR	2	R/W	0	MSTR: Master selection 0: 配置为从设备 1: 配置为主设备 注：1. 当通信在进行时不能改变该位的值。2. 该位在I2S模式下没有用
CPOL	1	R/W	0	CPOL: Clock polarity 0: 空闲状态时，SCK保持低电平; 1: 空闲状态时，SCK保持高电平。 注：1. 当通信在进行时不能改变该位的值。2. 在I2S模式和SPI TI模式下，这个位不可用。
CPHA	0	R/W	0	CPHA: Clock phase 0: 第一个时钟沿对准第一位数据 1: 第二和时钟沿对准第一位数据 注：1. 当通信在进行时不能改变该位的值。 2. 在I2S模式和SPI TI模式下，这个位不可用。

1. 6. 2 SPI 控制寄存器 2 (SPI_CR2)

		Address offset:		0x04											
		Reset value:		0x0000 0700											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				RXFTH[3:0]				Reserved				TXFTH[3:0]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	LDM A_TX	LDM A_RX	Reserved	DS[3:0]	Reserved	FRF	NSSP	SSOE	TXD MAE N	RXD MAE N

Name	Bit	Type	Reset value	Description
Reserved	31:28	N/A	0x0	保留，保持复位值
RXFTH[3:0]	27:24	R/W	0x0	Rx FIFO Threshold RXFTH[3:0]: 用于表示接收 FIO 触发 RXNE 状态阈值 注：当接收 DMA 使能时，这个 Bit 同时表示 DMA 接收请求条件
Reserved	23:20	N/A	0x0	保留，保持复位值
TXFTH[3:0]	19:16	R/W	0x0	Tx FIFO Threshold TXFTH[3:0]: 用于表示发送 FIO 触发 TXE 状态阈值 注：当发送 DMA 使能时，这个 Bit 同时表示 DMA 发送请求条件
Reserved	15:12	N/A	0	保留，保持复位值
DS[3:0]	11:8	R/W	0x7	DS [3:0]: Data size 这些位配置 SPI 传输数据的位宽： 0000~0010: 不使用 0011: 4 位 0100: 5 位 0101: 6 位 0110: 7 位 0111: 8 位 1000: 9 位 1001: 10 位 1010: 11 位 1011: 12 位 1100: 13 位 1101: 14 位 1110: 15 位 1111: 16 位 如果软件试图写一个“不使用”的值，他们会被迫赋值“0111”（8 位）。注：这些位在 I2S 模式下没有用
Reserved	7:5	N/A	0x0	保留，保持复位值
FRF	4	R/W	0	FRF: Frame format 0: SPI Motorola 模式 1: SPI TI 模式 注： 1. 该位必须在 SPI 被禁止（SPE = 0）的时候写入 2. 该位在 I2S 模式下没有用
NSSP	3	R/W	0	NSSP: NSS pulse management 该位仅在主模式下使用。它允许 SPI 在连续传输时，两个数据传输之间产生一个 NSS 脉冲。在单个数据传输的情况下，它会在传输结束后将 NSS 脚强制为高电平。 在 CPHA=1 或 FRF=1 的时候，这个位没有什么意义。 0: 没有 NSS 脉冲 1: 产生 NSS 脉冲 注： 1. 该位必须在 SPI 被禁止（SPE = 0）的时候写入 2. 在 I2S 模式和 SPI TI 模式下，这个位不可用。
SSOE	2	R/W	0	SSOE: SS output enable 0: 在主模式下 SS 输出被禁用，SPI 接口可以工作在多主机的配置下。 1: SPI 接口启用的同时主模式下启用 SS 输出。SPI 接口不能在多主环境下工作。注：在 I2S 模式和 SPI TI 模式下，这个位不可用。

TXDMAEN	1	R/W	0	TXDMAEN: Tx bufferDMAenable 当此位被设置, TXE 标志被设置时, 产生一个 DMA 请求。 0: Tx 缓冲 DMA 禁止 1: Tx 缓冲 DMA 使能
RXDMAEN	0	R/W	0	RXDMAEN: Rx bufferDMAenable 当此位被设置, RXNE 标志被设置时, 产生一个 DMA 请求。 0: Rx 缓冲 DMA 禁止 1: Rx 缓冲 DMA 使能

1. 6. 3 SPI 状态寄存器 (SPI_SR)

		Address offset:		0x08											
		Reset value:		0x0000 0002											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				RXFLV[4:0]				Reserved				TXFLV[4:0]			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FRE	BSY	OVR	MOD F	CRC ERR	UDR	CHSI DE	RXNE

Name	Bit	Type	Reset value	Description
Reserved	31:29	N/A	0x0	保留, 保持复位值
RXFLV[4:0]	28:24	R	0x0	RXFLV[4:0]: Rx FIFO level 指示接收 FIFO 指针.
Reserved	23:21	N/A	0x0	保留, 保持复位值
TXFLV[4:0]	20:16	R	0x0	TXFLV[4:0]: Tx FIFO level 指示发送 FIFO 指针.
Reserved	15:9	N/A	0x0	保留, 保持复位值
FRE	8	R	0	FRE: Frame format error 0: 没发生帧格式错误 1: 发生了一个帧格式错误
BSY	7	R	0	BSY: Busy flag 0: SPI (或 I2S) 不忙 1: (SPI 或 I2S) 通信忙或发送缓冲区不为空 由硬件设置和清除。
OVR	6	R	0	OVR: Overrun flag 0: 没有发生溢出 1: 发生溢出 此标志由硬件置位, 由软件序列复位。 软件序列, 请参阅第 669 页 关于错误标志。
MODF	5	R	0	MODF: Mode fault 0: 无模式故障发生 1: 模式故障发生 此标志由硬件置位, 由软件序列复位。 软件序列, 请参阅第 647 页 25.4 章。注: I2S 模式不使用
CRCERR	4	RC/WO	0	CRCERR: CRC error flag 0: 收到的 CRC 值和 SPIx_RXCRCR 的值是匹配的 1: 收到的 CRC 值和 SPIx_RXCRCR 值不匹配 此标志由硬件置位, 由软件清零。 注: I2S 模式下不使用

UDR	3	R	0	UDR: Underrun flag 0: 没有欠载发生 1: 欠载发生 此标志由硬件置位，由软件序列复位。 软件序列。注： 该位在 SPI 模式下没有用
CHSIDE	2	R	0	CHSIDE: Channel side 0: 需要传输或者接收左声道； 1: 需要传输或者接收右声道； 注： 该位在 SPI 模式下没有用 在 PCM 模式下无意义
TXE	1	R	1	TXE: Transmit buffer empty 0: Tx 缓冲区非空 1: TX 缓冲器空
RXNE	0	R	0	RXNE: Receive buffer not empty 0: RX 缓冲区空 1: Rx 缓冲非空

1. 6. 4 SPI 数据寄存器 (SPI_DR)

		Address offset:		0x0C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DR[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
DR[15:0]	15:0	R/W	0x0	DR[15:0]: Data register 待发送或者已经收到的数据 数据寄存器作为 Rx 和 Tx FIFOs 的接口。 当读取数据寄存器时，RXFIFO 会被访问，而写入数据寄存器则会访问 TXFIFO（见 25.3.7 小结：数据发送和接收流程）。 注： 数据始终是右对齐。 当写寄存器的时候，没有用到的位会被忽略，读的时候，没有用到的位会是 0. 接收门限的设置必须始终符合目前使用的读访问方式。

1. 6. 5 SPI CRC 多项式寄存器 (SPI_CRCPR)

		Address offset:		0x10											
		Reset value:		0x0000 0007											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRCPOLY[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
CRCPOLY[15:0]	15:0	R/W	0x0	CRCPOLY[15:0]: CRC polynomial register 该寄存器包含 CRC 计算多项式。 该寄存器的复位值是 (0007H)。根据需要，可以配置成另一个多项式 注：多项式的值应该是奇数。不支持偶数值。

1. 6. 6 SPI 接收 CRC 寄存器 (SPI_RXCRCR)

		Address offset:	0x14												
		Reset value:	0x0000 0000												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RxCRC[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值
RxCRC[15:0]	15:0	R	0x0	RxCRC[15:0]: Rx CRC register 当启用 CRC 计算功能，RxCRC [15:0] 位包含根据收到的字节计算出来的 CRC 值。 当 SPIx_CR1 寄存器的 CRCEN 位被写为 1 的时候，这个寄存器被复位。CRC 计算使用 SPI_CRCPR 中的多项式。 当数据帧格式被设置为 8 位 (SPIx_CR1 的 CRCL 位为 0) 时，仅低 8 位参与计算，并且按照 CRC8 的方法进行； 当数据帧格式为 16 位 (SPIx_CR1 的 CRCL 位为 1) 时，寄存器中的所有 16 位都参与计算，并且按照 CRC16 的方法进行； 注：在 BSY 标志为 1 的期间读这个寄存器的值，可能得到一个不正确结果。I2S 模式下不使用。

1. 6. 7 SPI 发送 CRC 寄存器 (SPI_TXCRCR)

		Address offset:	0x18												
		Reset value:	0x0000 0000												
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TxCRC[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x0	保留，保持复位值

TxCRC[15:0]	15:0	R	0x0	TxCRC[15:0]: Tx CRC register 当启用 CRC 计算功能，TxCRC [7:0] 位包含根据发送的字节计算出来的 CRC 值。 当 SPIx_CR1 寄存器的 CRCEN 位被写为 1 的时候，这个寄存器被复位。CRC 计算使用 SPI_CRCPR 中的多项式。 当数据帧格式被设置为 8 位（SPIx_CR1 中的 CRCL 位为 0）时，仅低 8 位参与计算，并且按照 CRC8 的方法进行； 当数据帧格式为 16 位（SPIx_CR1 的 CRCL 位为 1）时，寄存器中的所有 16 位都参与计算，并且按照 CRC16 的方法进行； 注：在 BSY 标志为 1 的期间读这个寄存器的值，可能得到一个不正确结果。注：I2S 模式下不使用。

1. 6. 8 SPI_I2S 配置寄存器 (SPI_I2SCFGR)

Address offset:		0x1C													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				I2SMOD	I2SE	I2SCFG		PCMSYNC	Reserved	I2SSTD		CKPOL	DATLEN		CHLEN

Name	Bit	Type	Reset value	Description
Reserved	31:12	N/A	0x0	保留，保持复位值
I2SMOD	11	R/W	0	I2SMOD: I2S mode selection 0: 选中 SPI 模式 1: 选中 I2S 模式 注：应该在 SPI 或 I2S 被禁用时，改变这个位。
I2SE	10	R/W	0	I2SE: I2S enable 0: I2S 的外设被禁用 1: I2S 外设被使能 注：SPI 模式不使用
I2SCFG	9:8	R/W	0x0	I2SCFG: I2S configuration mode 00: 从机 - 发送 01: 从机 - 接收 10: 主机 - 发送 11: 主机 - 接收 注：应该在 SPI 或 I2S 被禁用时，改变这个位。SPI 模式下不使用。
PCMSYNC	7	R/W	0	PCMSYNC: PCM frame synchronization 0: 短帧同步 1: 长帧同步 注：此位有只有在在 I2SSTD = 11（PCM 标准）的时候有意义，SPI 模式下不使用。
Reserved	6	N/A	0	保留，保持复位值
I2SSTD	5:4	R/W	0x0	I2SSTD: I2S standard selection 00: I2S 飞利浦标准 01: 高字节对齐标准（左对齐） 10: 低字节对齐标准（右对齐） 11: PCM 标准 注：为了正确的操作，这些位应该在 I2S 被禁用的时候配置。SPI 模式不使用

CKPOL	3	R/W	0	CKPOL: Inactive state clock polarity 0: I2S 时钟静止态为低电平; 1: I2S 时钟静止态为高电平; 注: 为了正确的操作, 这些位应该在 I2S 被禁用的时候配置。SPI 模式不使用
DATLEN	2:1	R/W	0x0	DATLEN: Data length to be transferred 00: 16 位数据长度 01: 24 位数据长度 10: 32 位数据长度 11: 不允许 注: 为了正确的操作, 这些位应该在 I2S 被禁用的时候配置。SPI 模式不使用
CHLEN	0	R/W	0	CHLEN: Channel length (number of bits per audio channel) 0: 16 位宽 1: 32 位宽 只有在 DATLEN = 00 时该位的写操作才有意义, 否则声道长度都由硬件固定为 32 位。 注: 为了正确的操作, 这些位应该在 I2S 被禁用的时候配置。SPI 模式不使用

1. 6. 9 SPI_I2S 预分频寄存器 (SPI_I2SPR)

		Address offset:		0x20											
		Reset value:		0x0000 0002											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					EXTCKEN	MCKOE	ODD	I2SDIV[7:0]							

Name	Bit	Type	Reset value	Description
Reserved	31:11	N/A	0x0	保留, 保持复位值
EXTCKEN	10	R/W	0	EXTCKEN: External I2S Clock Enable 0: 选 择 APB 时钟 1: 选 择 外 部 时钟 注: 应该在 I2S 被禁用时, 改变这个位。它仅用于 I2S 为主模式的状态。SPI 模式不使用
MCKOE	9	R/W	0	MCKOE: Master clock output enable 0: 主时钟输出被禁用 1: 主时钟输出启用 注: 应该在 I2S 被禁用时, 改变这个位。它仅用于 I2S 为主模式的状态。SPI 模式不使用
ODD	8	R/W	0	ODD: Odd factor for the prescaler 0: 实际分频系数 = I2SDIV * 2; 1: 实际分频系数 = (I2SDIV * 2) + 1; 注: 应该在 I2S 被禁用时, 改变这个位。它仅用于 I2S 为主模式的状态。SPI 模式不使用

I2SDIV[7:0]	7:0	R/W	0x2	I2SDIV[7:0]: I2S linear prescaler 不允许设置 I2SDIV [7:0] =0 或者 I2SDIV [7:0] =1 的值。注：应该在 I2S 被禁用时，改变这个位。它仅用于 I2S 为主模式的状态。SPI 模式不使用

1. 6. 10 SPI 中断使能寄存器 (SPI_IER)

		Address offset:		0x24											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							FREIE	Reser ved	OVRI E	MOD FIE	CRCE RRIE	UDRI E	Reser ved	TXEIE	RXNE IE

Name	Bit	Type	Reset value	Description
Reserved	31:9	N/A	0x0	保留，保持复位值
FREIE	8	W	0	FREIE: Frame format error interrupt enable 0: 无 1: 中断使能
Reserved	7	N/A	0	保留，保持复位值
OVRIE	6	W	0	OVRIE: Overrun flag interrupt enable 0: 无 1: 中断使能
MODFIE	5	W	0	MODFIE: Mode fault interrupt enable 0: 无 1: 中断使能
CRCE RRIE	4	W	0	CRCE RRIE: CRC error interrupt enable 0: 无 1: 中断使能
UDRIE	3	W	0	UDRIE: Underrun flag interrupt enable 0: 无 1: 中断使能
Reserved	2	N/A	0	保留，保持复位值
TXEIE	1	W	0	TXEIE: Transmit buffer empty interrupt enable 0: 无 1: 中断使能
RXNEIE	0	W	0	RXNEIE: Receive buffer not empty interrupt enable 0: 无 1: 中断使能

1. 6. 11 SPI 中断禁止寄存器 (SPI_IDR)

		Address offset:		0x28											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Reserved	FREID	Reserved	OVRI	MODFID	CRCE	UDRI	Reserved	TXEID	RXNEID

Name	Bit	Type	Reset value	Description
Reserved	31:9	N/A	0x0	保留，保持复位值
FREID	8	W	0	FREID: Frame format error interrupt disable 0: 无效 1: 中断禁止
Reserved	7	N/A	0	保留，保持复位值
OVRI	6	W	0	OVRI: Overrun flag interrupt disable 0: 无效 1: 中断禁止
MODFID	5	W	0	MODFID: Mode fault interrupt disable 0: 无效 1: 中断禁止
CRCE	4	W	0	CRCE: CRC error interrupt disable 0: 无效 1: 中断禁止
UDRI	3	W	0	UDRI: Underrun flag interrupt disable 0: 无效 1: 中断禁止
Reserved	2	N/A	0	保留，保持复位值
TXEID	1	W	0	TXEID: Transmit buffer empty interrupt disable 0: 无效 1: 中断禁止
RXNEID	0	W	0	RXNEID: Receive buffer not empty interrupt disable 0: 无效 1: 中断禁止

1. 6. 12 SPI 中断有效状态寄存器(SPI_IVS)

		Address offset:		0x2C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FREIV	Reserved	OVRI	MODFIV	CRCE	UDRI	Reserved	RXNEIV

Name	Bit	Type	Reset value	Description
Reserved	31:9	N/A	0x0	保留，保持复位值
FREIV	8	R	0	FREIV: Frame format error interrupt valid state 0: 中断禁止状态 1: 中断使能状态
Reserved	7	N/A	0	保留，保持复位值
OVRI	6	R	0	OVRI: Overrun flag interrupt valid state 0: 中断禁止状态 1: 中断使能状态
MODFIV	5	R	0	MODFIV: Mode fault interrupt valid state 0: 中断禁止状态 1: 中断使能状态

CRCERRIV	4	R	0	CRCERRIV: CRC error interrupt status 0: 中断禁止 1: 中断使能状态
UDRIV	3	R	0	UDRIV: Underrun flag interrupt status 0: 中断禁止 1: 中断使能状态
Reserved	2	N/A	0	保留, 保持复位值
TXEIV	1	R	0	TXEIV: Transmit buffer empty interrupt status 0: 中断禁止 1: 中断使能状态
RXNEIV	0	R	0	RXNEIV: Receive buffer not empty interrupt status 0: 中断禁止 1: 中断使能状态

1. 6. 13 SPI 原始中断标志寄存器(SPI_RIF)

Address offset:		0x30													
Reset value:		0x0000 0000													
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							FRERI	Reserved	OVRRRI	MODFRI	CRCERRRI	UDRRI	Reserved	TXERI	RXNEIRI

Name	Bit	Type	Reset value	Description
Reserved	31:9	N/A	0x0	保留, 保持复位值
FRERI	8	R	0	FRERI: Frame format error raw interrupt flag status 0: 没有中断标志 1: 产生中断标志位
Reserved	7	N/A	0	保留, 保持复位值
OVRRRI	6	R	0	OVRRRI: Overrun flag raw interrupt flag status 0: 没有中断标志 1: 产生中断标志位
MODFRI	5	R	0	MODFRI: Mode fault raw interrupt flag status 0: 没有中断标志 1: 产生中断标志位
CRCERRRI	4	R	0	CRCERRRI: CRC error raw interrupt flag status 0: 没有中断标志 1: 产生中断标志位
UDRRI	3	R	0	UDRRI: Underrun flag raw interrupt flag status 0: 没有中断标志 1: 产生中断标志位
Reserved	2	N/A	0	保留, 保持复位值
TXERI	1	R	0	TXERI: Transmit buffer empty raw interrupt flag status 0: 没有中断标志 1: 产生中断标志位
RXNEIRI	0	R	0	RXNEIRI: Receive buffer not empty raw interrupt flag status 0: 没有中断标志 1: 产生中断标志位

1. 6. 14 SPI 屏蔽中断标志寄存器 (SPI_IFM)

		Address offset:		0x34											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							FREFM	Reserved	OVRFM	MODFFM	CRCE RRFM	UDRFM	Reserved	TXEFM	RXNEFM

Name	Bit	Type	Reset value	Description
Reserved	31:9	N/A	0x0	保留，保持复位值
FREFM	8	R	0	FREFM: Frame format error interrupt flag masked status 0: 没有中断标志位 1: 产生中断标志位
Reserved	7	N/A	0	保留，保持复位值
OVRFM	6	R	0	OVRFM: Overrun flag interrupt flag masked status 0: 没有中断标志位 1: 产生中断标志位
MODFFM	5	R	0	MODFFM: Mode fault interrupt flag masked status 0: 无 1: 屏蔽中断
CRCE RRFM	4	R	0	CRCE RRFM: CRC error interrupt flag masked status 0: 没有中断标志位 1: 产生中断标志位
UDRFM	3	R	0	UDRFM: Underrun flag interrupt flag masked status 0: 没有中断标志位 1: 产生中断标志位
Reserved	2	N/A	0	保留，保持复位值
TXEFM	1	R	0	TXEFM: Transmit buffer empty interrupt flag masked status 0: 没有中断标志位 1: 产生中断标志位
RXNEFM	0	R	0	RXNEFM: Receive buffer not empty interrupt flag masked status 0: 没有中断标志位 1: 产生中断标志位

1. 6. 15 SPI 中断清除寄存器(SPI_ICR)

		Address offset:		0x38											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							FREIC	Reserved	OVRI C	MOD FIC	CRCE RRIC	UDRI C	Reserved	TXEIC	RXNEIC

Name	Bit	Type	Reset value	Description
Reserved	31:9	N/A	0x0	保留，保持复位值

FREIC	8	W1C	0	FREIC: Frame format error interrupt clear 0: 无 1: 清除中断标志位
Reserved	7	N/A	0	保留，保持复位值
OVRIC	6	W1C	0	OVRIC: Overrun flag interrupt clear 0: 无 1: 清除中断标志位
MODFIC	5	W1C	0	MODFIC: Mode fault interrupt clear 0: 无 1: 清除中断标志位
CRCERRIC	4	W1C	0	CRCERRIC: CRC error interrupt clear 0: 无 1: 清除中断标志位
UDRIC	3	W1C	0	UDRIC: Underrun flag interrupt clear 0: 无 1: 清除中断标志位
Reserved	2	N/A	0	保留，保持复位值
TXEIC	1	W1C	0	TXEIC: Transmit buffer empty interrupt clear 0: 无 1: 清除中断标志位
RXNEIC	0	W1C	0	RXNEIC: Receive buffer not empty interrupt clear 0: 无 1: 清除中断标志位

第21章 通用异步收发器(UART)

1.1 UART 简介

通用同步异步收发器 (UART) 提供了一个灵活的方式, 使 MCU 可以与外部设备通过工业标准 NRZ 的形式实现全双工异步串行数据通讯。UART 可以使用分数波特率发生器, 提供了超宽的波特率设置范围。

UART 支持非同步通讯模式和半双工单线通讯。也支持 LIN (本地互联网络), 智能卡协议和 IrDA (红外数据协会) SIR ENDEC 规范和 modem 流控操作 (CTS_n/RTS_n) 同时还支持多机通讯方式。

可以使用 DMA 实现多缓冲区设置, 从而能够支持高速数据通讯。

1.2 UART 主要特性

- ◆ 全双工异步通信
- ◆ 16-byte 接收和发送 FIFO
- ◆ 兼容 16C550 标准
- ◆ 可程序设计接收缓冲触发点
- ◆ 支持各信道独立波特率可程序设计
- ◆ 支持自动波特率检测
- ◆ 十二个中断源
- ◆ 可与 DMA 使用
 - 利用 DMA 功能将收/发字节缓冲到保留的 SRAM 空间
- ◆ 内置小数波特率发生器, 覆盖范围广的波特率不需要特定值的外部晶体
 - 可编程收发波特率高达 3MHz, 最低可达 732Hz(时钟频率为 48MHz)
- ◆ 支持硬件自动流控制/流控制功能 (CTS_n、RTS_n), RTS_n 控制流触发点可程序设计
 - Modem 硬件自动控制
 - RS485 发送始能控制
- ◆ 支持 CTS 唤醒功能
- ◆ 支持 IrDA SIR 模式
 - 支持 3/16 位周期调制
- ◆ 支持 RS-485
 - 支持 9-位模式
 - 多处理器通信
- ◆ 完全可程序设计的串行接口特性
 - 可程序设计数据位个数, 即 5-,6-,7-,8-,9-位
 - 校验位, 奇、偶、无校验或者固定校验位生成和检测可程序设计

- 停止位长度可程序设计：1，2 位，在智能卡模式中支持 0.5，1.5 位
- 可设置高位在前或低位在前
- ◆ 单线半双工通讯
- ◆ 交换 Tx/Rx pin 配置
- ◆ LIN 主机的断开信号发送能力和 LIN 从机的断开信号检测能力
 - 将 UART 设置为 LIN 模式时，有 13 位的断开信号发生器与断开信号检测功能
- ◆ 智能卡模式
 - 支持 ISO/IEC7816-3 标准定义的 T=0 和 T=1 智能卡异步协议
 - 智能卡使用的 1.5 停止位长度
- ◆ 支持 ModBus 通讯
 - 超时检测功能
 - CR/LF 字符识别
- ◆ 杂讯侦测

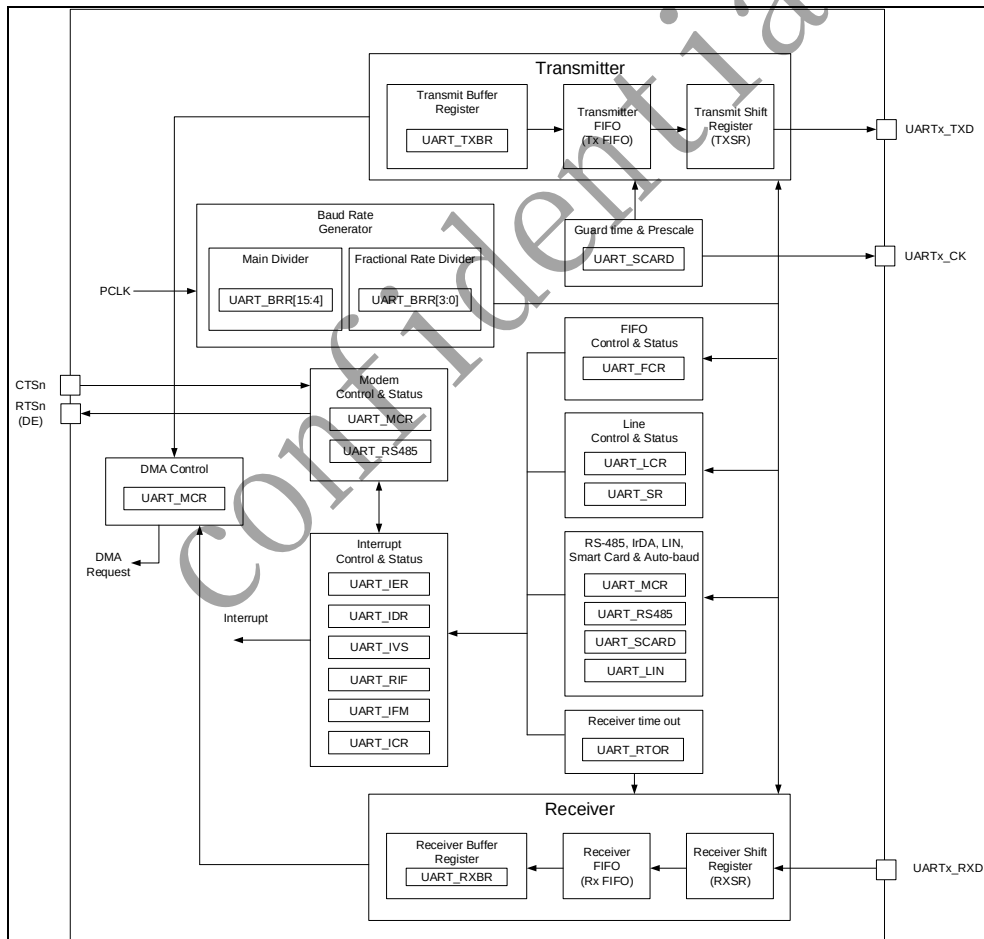


图 21-1UART 框图

1.3 UART 功能描述

接口与外部设备通过三个引脚相连，如下图。任何 UART 双向通讯要求最少有两个引脚：接

收数据输入 (RX) 和发送数据输出 (TX)

RXD: 接收数据输入是串行数据的输入口。使用过采样技术来完成数据恢复，以区别输入数据和噪声。

TXD: 数据发送输出。当发送器被禁止，输出脚回到其 I/O 口配置状态。当发送器被使能，但不发送数据时，TXD 脚为高电平输出。在单线和智能卡模式中，这个口线既用于发送数据也用于接收数据。

通过这些引脚，串行数据用数据帧的形式发送和接收：

- ◆ 在发送和接收之前为空闲状态
- ◆ 起始位
- ◆ 数据字 (5,6,7 或 8 位) 可透过 LCR 寄存器的 MSBFIRST 位设定最低有效位在前或后
- ◆ 1, 2 个停止位表明帧的结束(0.5, 1.5 个停止位用于智能卡模式)
- ◆ 采用小数波特率发生器，整数 12 位小数 4 位
- ◆ 一个状态寄存器 (UART_SR)
- ◆ 分开的接收和发送数据寄存器 (UART_RDR, UART_TDR)
- ◆ 一个波特率寄存器 (UART_BRR) -12 位整数和 4 位小数。
- ◆ 一个智能卡寄存器 (UART_GTPR) 用于智能卡模式。
- ◆ 一个接收时间寄存器 (UART_RTOR) 侦测输入讯号时间并产生中断

下面的引脚在智能卡模式中会用到：

SCLK:时钟输出。智能卡模式中，SCLK 引脚会向智能卡提供时钟。

下列引脚用于支持硬件流控制模式：

CTS_n:低发送，当高电平时作为发送阻塞信号。

RTS_n:请求发送，表明 UART 已经准备好接收数据（低的时候）。

下列引脚在 RS485 驱动使能控制的时候会用到：

DE:驱动使能将外部收发器的发送模式激活。

注：DE 和 RTS_n 共享同一个外部引脚。

1.3.1 UART 特性描述

配置 LCR 寄存器中的 DLS 位可选择 8-5 位字长。

默认设置中，发送和接收的起始位都是低电平。而停止位都是高电平。

这个逻辑可以在极性控制中单独的设置为反向。

- ◆ 8 位字符宽度:DLS[1:0]=00
- ◆ 7 位字符宽度:DLS[1:0]=01
- ◆ 6 位字符宽度:DLS[1:0]=10

◆ 5 位字符宽度:DLS[1:0]=11

注：第 9 位使用于 RS485 multiprocessor 模式

空闲符号被视为完全由'1'组成的完整的数据帧，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)。

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，发送器会再插入 2 个停止位。

发送和接收由一个共享的波特率发生器驱动，当发送器和接收器的使能位分别置 1 时，分别为其产生时钟。

下面是每个模块的详细说明。

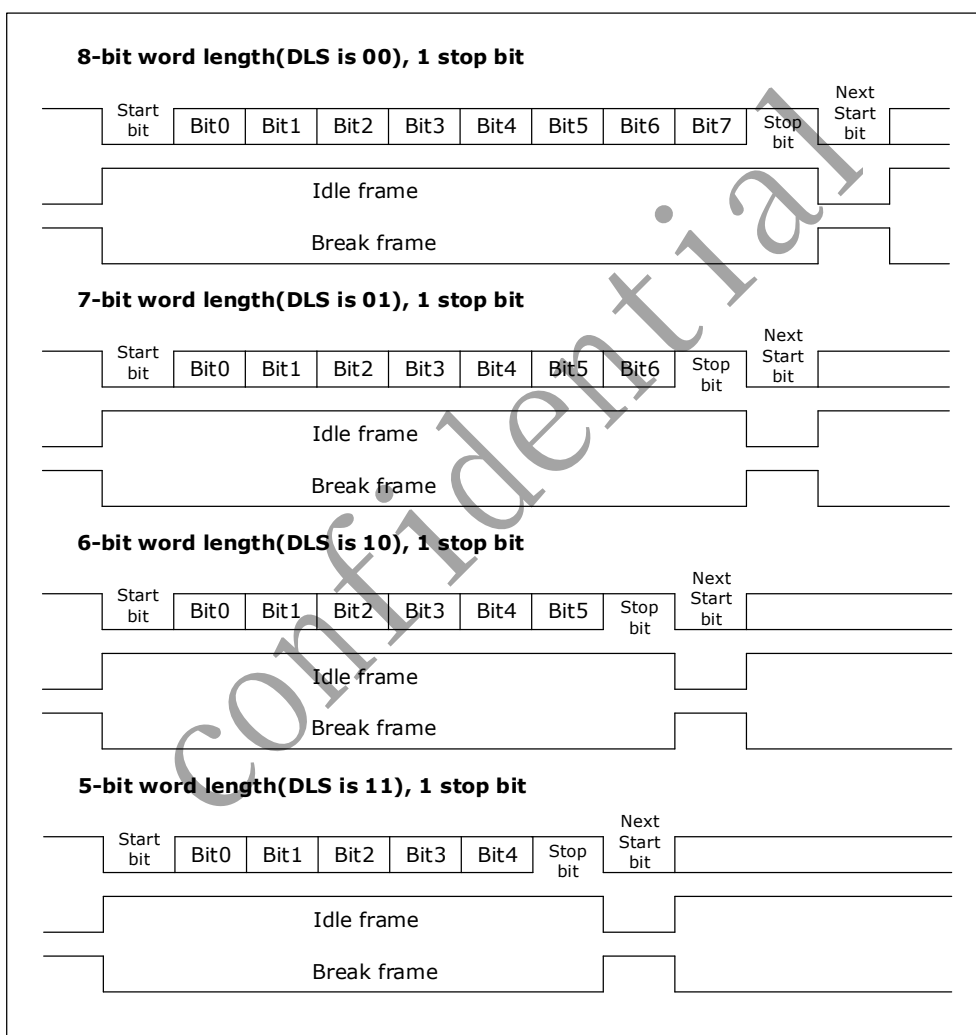


图 21-2 数字通信长度配置

1.3.2 UART 发送

发送器根据 DLS 位的状态发送 8-5 位的数据字。

当写入 TDR 寄存器，由发送移位寄存器中的数据在 TXD 脚上输出。

字符发送

在 UART 发送期间，在 TX 引脚上首先移出数据的最低有效位。在此模式里，TDR 寄存器充当了一个内部总线和发送移位寄存器之间的缓冲器(TXSR)，如 Figute1-1。

每个字符之前都有一个低电平的起始位.之后跟着停止位，停止位的数目是可选的。

UART 支持多种停止位的选择：0.5,1,1.5 和 2 个停止位。

注: 1. 在写入 TDR 寄存器数据必须先令 SR 寄存器中的 TFFULL 位为 0

2. 打开 TX 开关后，数据才能在 TX 脚上输出

可配置的停止位

随每个字符发送的停止位的位数可以通过 LCR 寄存器中的 STOP 位进行编程。

- ◆ 0.5 个停止位：在智能卡模式下发送和接收数据时使用。
- ◆ 1 个停止位：停止位的位数的默认值。
- ◆ 1.5 个停止位：在智能卡模式下发送和接收数据时使用。
- ◆ 2 个停止位：可用于常规 UART 模式、以及调制解调器模式。

空闲帧包括了停止位。

断开帧可透过 MCR 寄存器的 BKREQ 位产生 5 位低电平(当 DLS=11 时)，6 位低电平(当 DLS=10 时)，7 位低电平(当 DLS=01 时)或者 8 位低电平(当 DLS=00 时)，后跟 2 个停止位。

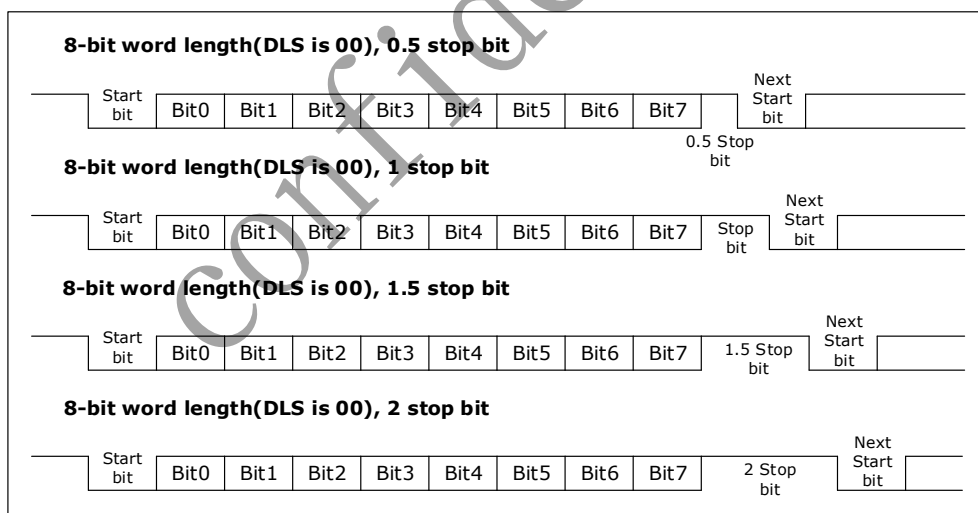


图 21-3 停止位配置

配置步骤：

1. 设置 LCR 寄存器中的 DLS 位来定义字长。
2. 利用 BRR 寄存器选择希望的波特率。
3. 在 LCR 寄存器中的 STOP 位设置停止位的位数。
4. 如果采用多缓冲器通信，配置 MCR 寄存器中的 DMA 使能位(TXDMAEN)。按多缓冲器通

信中的描述配置 DMA 寄存器。

5. 把要发送的数据写进 TDR 寄存器(此动作将清除 SR 寄存器中的 TFEMPTY 位)。

6. 在 TDR 寄存器中写入数据字時，要等待 SR 寄存器中的 TFFULL 位為 0，它表示 FIFO 中未滿足 16byte。当需要关闭 UART，需要确认传输结束 SR 寄存器中的 TSBUSY 位為 0，避免破坏最后一次传输。

1.3.3 UART 接收

1.3.3.1 防抖电路

在 UART_RXD 引脚上配置了一个防抖电路，输入讯号须维持至少 8 个高位或低位，才能使得讯号反应至 UART 中，反之则被忽略，如下叙述。

在下图中，SYNC0 是指输入信号由系统时钟一次采样;SYNC1 是指 SYNC0 被系统时钟一次采样; SYNC2 表示 SYNC1 由系统时钟一次采样。SYNC0、SYNC1 和 SYNC2 可以表示成 $x[n] \times T_s$ ， $x[n+1] \times T_s$ 和 $x[n+2] \times T_s$ 。 T_s 是系统时钟周期， n 是采样时间。如果关闭防抖模块，时间就可以表示为 $x[n+1] \times T_s$ 。

如果打开防抖模块，SYNC2 信号将进入去抖电路，那么它将被采样与计数(用户设定样本频率和计数时间值)。时间可表示为 $[(SPT+1) \times (FILTCNT+1)] \times T_s$ 。SPT 是采样频率值，FILTCNT 是计数次数。当 SYNC1 和 SYNC2 不相等时，计数值将被清零。如果计数值溢满 FILTCNT 寄存器，防抖模块将输出信号。

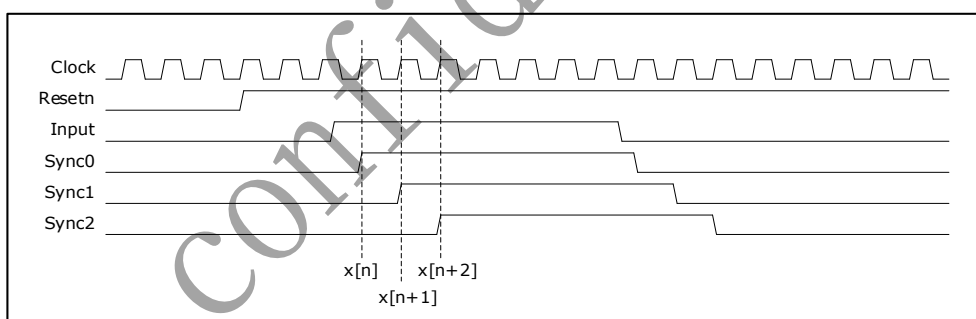


图 21-4 防抖波形

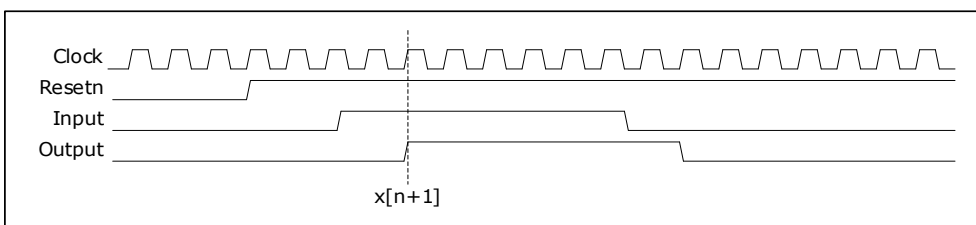


图 21-5 输出去抖动

1.3.3.2 起始位检测

接收器根据 LCR 寄存器中 DLS 位的状态接收 8-5 位的数据字。

起始位侦测

在 UART 中，如果辨认出一个特殊的采样序列，那么就认为侦测到一个起始位。

该序列为：1 1 1 0 X 0 X 0 X 0 0 0 0 X XXXXXX

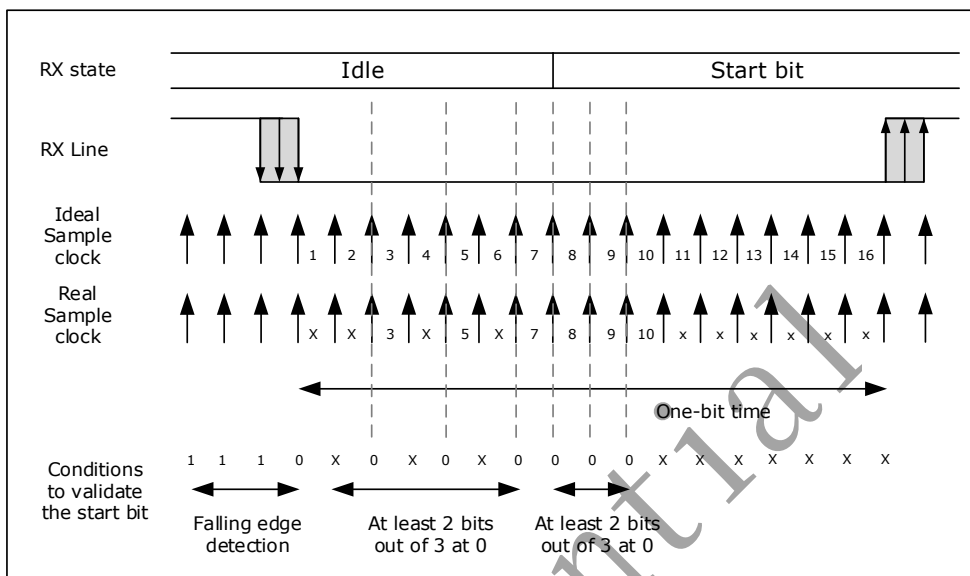


图 21-6 起始位检测

注：如果该序列不完整，那么接收端将退出起始位侦测并回到空闲状态(不设置标志位)开始等待下降沿。

如果 3 个采样点都为'0'(在第 3、5、7 位的第一次采样，和在第 8、9、10 的第二次采样都为'0')，则确认收到起始位。

如果两次 3 个采样点上仅有 2 个是'0'(第 3、5、7 位的采样点和第 8、9、10 位的采样点)，那么起始位仍然是有效的。如果不能满足这个条件，则中止起始位的侦测过程，接收器会回到空闲状态。

如果两次 3 个采样点上仅有 2 个是'1'(第 3、5、7 位的采样点和第 8、9、10 位的采样点)，那么起始位是无效的，将退出起始位侦测并回到空闲状态，并会设置 NOISE 噪声标志位。

配置步骤：

1. 设置 LCR 寄存器中的 DLS 位来定义字长。
2. 利用波特率寄存器 BRR 寄存器选择希望的波特率。
3. 在 LCR 寄存器中的 STOP 位设置停止位的位数。
4. 如果采用多缓冲器通信，配置 MCR 寄存器中的 DMA 使能位(RXDMAEN)。按多缓冲器通信中的描述配置 DMA 寄存器。
5. 设置 LCR 寄存器中的 RXEN 位，这将激活接收器，使它开始寻找起始位。

当一个字符被接收到时

SR 寄存器的 RFEMPTY 位被置 0。它表明移位寄存器的内容被转移到 RDR。换句话说，数据已经被接收并且可以被读出(包括与之有关的错误标志)。

这时如果 IER 寄存器的 RFTH 位是 1，且 Rx FIFO threshold 为 1，将会引起中断请求。

在接收期间如果检测到帧错误，噪音或溢出错误，错误标志将被置起。RXBERR 标志也会和 RFTH 一起被置 1。

在多缓冲器通信时，RFEMPTY 在每个字节接收后被清零，并由 DMA 对数据寄存器的读操作而置起。

RFFULL 位必须在下一字符接收结束前被清零，以避免溢出错误。

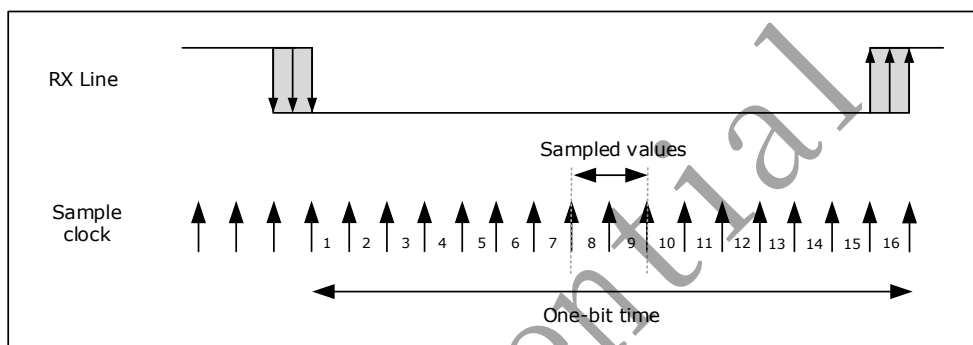


图 21-7 数据采样

帧错误

当以下情况发生时检测到帧错误：

由于没有同步上或大量噪音的原因，停止位没有在预期的时间上接收和识别出来。

当帧错误被检测到时：

1. FERR 位被硬件置 1
2. 此时读取的 RDR 寄存器数据可能有错。
3. 寄存器 SR 中的 FERR 位显示当前从 FIFO 读取的 RDR 寄存器是否为帧错误。
4. 寄存器 RIF 中的 RXBERR 位则会在接收的过程中被置起，若 IER 中的 RXBERR 位为 1 则会产生中断。

奇偶位错误

当以下情况发生时检测到奇偶性错误：

由于没有同步上或大量噪音的原因，奇偶位没有在预期的时间上接收和识别出来。

当奇偶位错误被检测到时：

1. PERR 位被硬件置 1
2. 此时读取的 UART_RDR 寄存器数据可能有错。
3. 寄存器 SR 中的 PERR 位显示当前从 FIFO 读取的 RDR 寄存器是否为帧错误。
4. 寄存器 RIF 中的 RXBERR 位则会在接收的过程中被置起，若 IER 中的 RXBERR 位为 1 则会产生中断。

断开错误

当以下情况发生时检测到断开错误：

由于没有同步上或大量噪音的原因，资料与停止位为 0 且没有在预期的时间上接收和识别出来。

当断开错误被检测到时：

1. BKERR 位被硬件置 1
2. 此时读取的 RDR 寄存器数据可能有错。
3. 寄存器 SR 中的 BKERR 位显示当前从 FIFO 读取的 RDR 寄存器是否为帧错误。
4. 这个错误并不会产生中断。

溢出错误

当以下情况发生时检测到溢出错误：

由于 FIFO 已满 16byte 还没有被读取，而又接收到一个字符，则发生溢出错误。

当溢出错误被检测到时：

1. RFOERR 位被硬件置 1
2. RDR 内容将不会丢失，读取 RDR 寄存器仍能得到先前的数据。
3. 移位寄存器中以前的内容将被覆盖，随后接收的数据将丢失。
4. 寄存器 RIF 中的 RFOERR 位则会在接收的过程中被置起，若 IER 中的 RFOERR 位为 1 则会产生中断。

Sampled value	Received bit value
000	0

001	0
010	0
011	1
100	0
101	1
110	1
111	1

表 21-1 采样数据噪声检测

1.3.4 UART 状态寄存器

在 UART 中配置了 13 种 status 提供使用者坐使用，分别为

PERR, FERR, BKERR, CTSSTA, RSBUSY, RFEMPTY, RFFULL, RFOERR, RFUERR, TSBUSY, TFEMPTY, TFFULL, TFOERR

◆ PERR(Parity error):

当所接收的字符没有正确的校验字节，将生成一个奇偶校验错误。该错误与 FIFO 顶部的字符有关，即读取寄存器 RDR 的字符。

◆ FERR(Frame error):

当接收到的字符停止位为 0 时，产生帧错误。该错误与 FIFO 顶部的字符有关，即读取寄存器 RDR 的字符。

◆ BKERR(Break error):

当接收到的字符与字符停止位为 0 时，产生断开错误。该错误与 FIFO 顶部的字符有关，即读取寄存器 RDR 的字符。

◆ CTSSTA(CTSnstatuserror):

清除发送。此位为 CTSn pin 上的准位。当 CTSSTA 为 1 这是一个迹象表明调制解调器和数据设备已准备好与 UART 进行数据交换。

◆ RSBUSY(Rx shifter register busy):

当此位为 1 表示接收器正在接收字符，为 0 则为接收完成。

◆ RFEMPTY(Rx FIFO empty):

当此位为 1 表示 Rx FIFO 内无任何字符，为 0 则为已接收 1 笔以上的字符。

◆ RFFULL(Rx FIFO full):

当此位为 1 表示 Rx FIFO 内已有 16 笔字符，需要被读取。

◆ RFOERR(Rx FIFO overrun):

当此位为 1 表示 Rx FIFO 内已有 16 笔字符，且又再接收 1 笔字符，此时 FIFO 内字符不会丢失，接收的字符则会被丢失。

◆ RFUERR(Rx FIFO underrun):

当此位为 1 表示 Rx FIFO 内无任何字符，且又被读取。

◆ TSBUSY(Tx shifter register busy):

当此位为 1 表示发送器正在传送字符，为 0 则为传送完成。

◆ TFEMPTY(Tx FIFO empty):

当此位为 1 表示 Tx FIFO 内无任何字符，为 0 则为准备发送 1 笔以上的字符。

◆ TFFULL(Tx FIFO full):

当此位为 1 表示 Tx FIFO 内已有 16 笔字符，准备发送。

◆ TFOERR(Tx FIFO overrun):

当此位为 1 表示 Tx FIFO 内已有 16 笔字符，且又在写入 1 笔字符，此时 FIFO 内字符不会丢失，写入的字符则会被丢失。

1. 3. 5 UART 波特率生成

接收器和发送器的波特率在 UARTDIV 的整数和小数寄存器中的值应设置成相同。

UARTDIV=UART.BRR.

$$\text{Tx/Rx baud} = \frac{\text{Fck}}{16 * \text{UARTDIV}}$$

UARTDIV 是一个无符号的定点数。这 12 位的值设置在 BRR 寄存器中的 BRR[15:4]中。小数部分按 4 位计算，设置在 BRR 寄存器中的 BRR[3:0]中。

注: 当 LCR 寄存中的 RXEN 为 1 时，BRR 寄存器无法被写入。

1. 3. 5. 1 如何设置 UART_BRR 寄存器值

◆ 在 16MHz 下，为了得到 1200baud

- $\text{UARTDIV} = 16000000 / 1200 / 16 = 13333.33 / 16 = 833.33\text{d}$
- $\text{BRR}[15:4] = 833\text{d} = 341\text{h}$
- $\text{BRR}[3:0] = 13333.33 - 13328 = 5.33\text{d} = 5\text{h}$ (四捨五入)
- $\text{BRR}[15:0] = \text{UARTDIV} = 833\text{d} = 3415\text{h}$

◆ 在 32MHz 下，为了得到 460.8Kbaud

- $\text{UARTDIV} = 32000000 / 460800 / 16 = 69.94 / 16 = 4.37\text{d}$
- $\text{BRR}[15:4] = 4\text{d} = 4\text{h}$
- $\text{BRR}[3:0] = 69.94 - 64 = 5.94\text{d} = 6\text{h}$ (四捨五入)
- $\text{BRR}[15:0] = \text{UARTDIV} = 52\text{d} = 46\text{h}$

Baud rate		Oversampling by 16		
No.	Desired	Actual	BRR	%Error=(Calculated-Desired) B.Rate/Desired B.Rate
1	734Bps	733Bps	0x5526	0
2	1.2KBps	1.2KBps	0x3415	0

3	2.4KBps	2.4KBps	0x1A0B	0
4	4.8KBps	4.8KBps	0xD05	0
5	9.6KBps	9.6KBps	0x683	0
6	19.2KBps	19.2KBps	0x341	0
7	38.4KBps	38.4KBps	0x1A1	0
8	57.6KBps	57.6KBps	0x116	0.03
9	115.2KBps	115.11KBps	0x8B	0.08
10	230.4KBps	230.77KBps	0x45	0.16
11	460.8KBps	461.54KBps	0x23	0.16
12	921.6KBps	923.07KBps	0x11	0.16

表 21-2 f_{CK} = 16MHz 时，设置波特率的误差计算

1.3.6 UART 自动波特率检测

UART 可以根据接收到的一个字符来检测和自动设置 BRR 寄存器的值。自动波特率检测在两种情况下有用：

- ◆ 通讯速度不可知的情况下
- ◆ 使用低精度时钟源，需要在不测量时钟偏差的条件下纠正波特率的时候。

时钟源的频率必须和预期的波特率保持相对的稳定

（过采样率为 16，并且波特率处于 $f_{CK}/65535$ 和 $f_{CK}/16$ 之间）。

在打开自动波特率检测之前，字符的内容必须先确认。有两个可能的字符内容，能够通过 MCR 寄存器中的 ABRMOD 位进行选择。具体是：

- ◆ 模式 0(0x00):波特率在 UART 的 RXD 引脚的两个连续的下降沿上测量（起始位的下降沿和最低显著位的下降沿（LSB））
- ◆ 模式 1(0x01):波特率下降沿 UART 的 RXD 引脚（起始位的长度）的后续的上升沿之间测量。
- ◆ 模式 2(0x10):波特率下降沿 UART 的 RXD 引脚（起始位的长度）的后续的上升沿之间测量(e.g. 0xFE)。

将 MCR 寄存器中的 ABREN 位置 1，就打开自动波特率检测功能了。然后 UART 就在 RXD 线上等待第一个字符过来。自动波特率操作结束后，RIF 寄存器中的 ABEND 标志会被硬件置 1，若 IER 寄存器中的 ABEND 位为 1 则会产生中断。

如果线路噪声严重，不能保证得到的波特率是准确的。这时 BRR 值可能是错的或者 ABEND 错误标志会被置 1。在通讯速度超出自动波特率检测范围（位长度不在 0x10 到 0xFFFF 个时

钟周期之间) 时也会发生这种情况。

在此模式中配置了两个中断, 分别为 time out 与 end block。

在软件并未清除 MCR 寄存器中的 ABREN 下, UART 计数器数会持续计数直到 0xFFFFF 后, 硬件会自动将 ABREN 清除, 并将 RIF 寄存器中的 ABTO 置起, 若 IER 寄存器中的 ABTO 位为 1 则会产生中断。

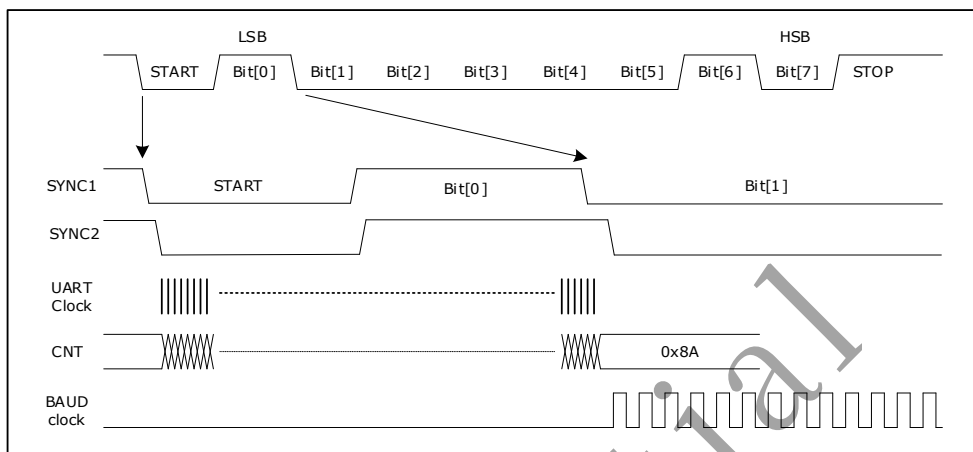


图 21-8 自动波特率检测 (模式 0)

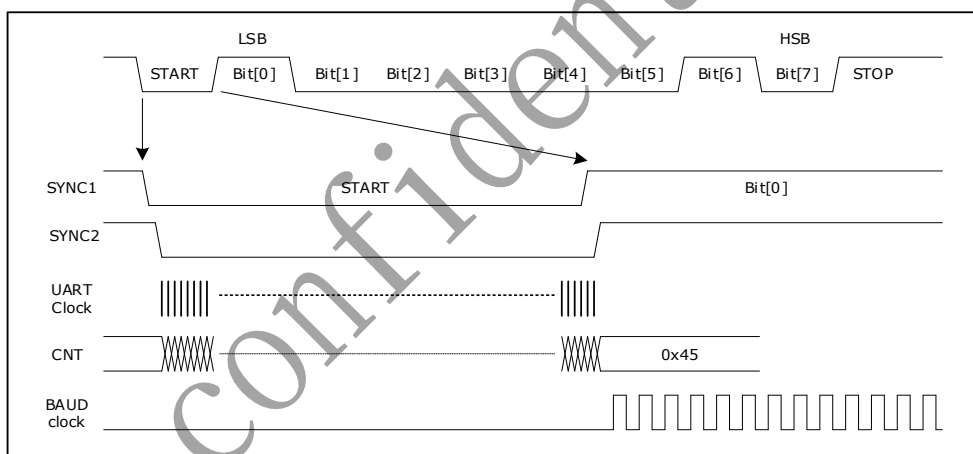


图 21-9 自动波特率检测 (模式 1)

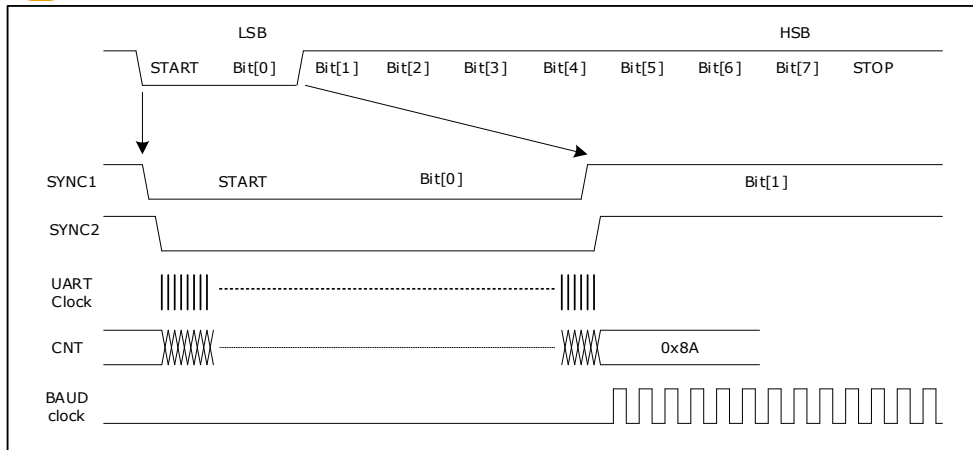


图 21-10 自动波特率检测（模式 2）

1. 3. 7 UART 自动流控制

如果使能自动流控制，接收 FIFO 和发送 FIFO 会通过 UARTx_RTSn 和 UARTx_CTSn 引脚去控制 UART 的接收(RXD)和发送(TXD)。

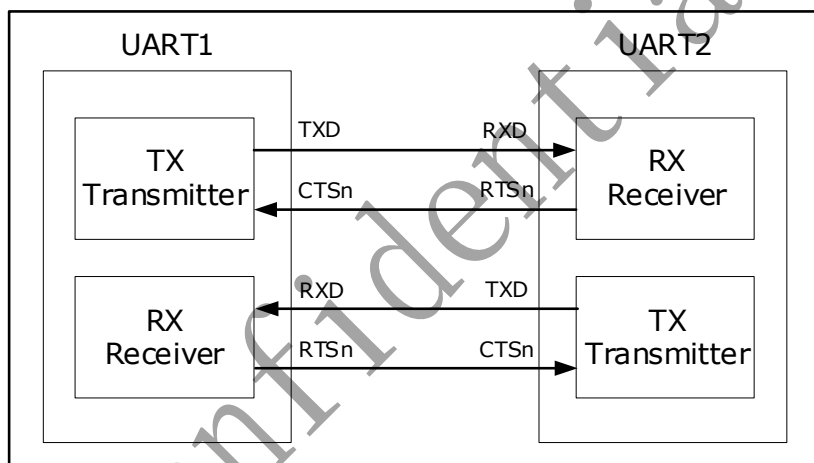


图 21-11 自动流控制连接框图

1. 3. 7. 1 Auto-RTSn

当 Auto-RTS 被使能时(MCR 寄存器中的 AFCEN 为 1)，接收 FIFO 达到由 FCR 寄存器设置的阈值，UART_RTSn 输出会拉为高电平。当 UART_RTSn 连接到另一个 UART 设备的 UART_CTSn 输入，另一个 UART 会停止发送串行数据，直到接收 FIFO 完全是空的

可选择接收 FIFO 阈值的值是：1,4,8, 14 byte。一个额外的字符有可能会在 UART_RTSn 成为无效后传送到 UART（由于已经在其他 UART 进入发送器的数据块），阈值设置为 14 byte 能用一个字符的安全带使用 FIFO 最大。

通过读取接收缓冲 RDR 寄存器，一旦得知接收 FIFO 完全是空的时候，UART_RTSn 变低电平，发信通知其他 UART 继续发送数据。

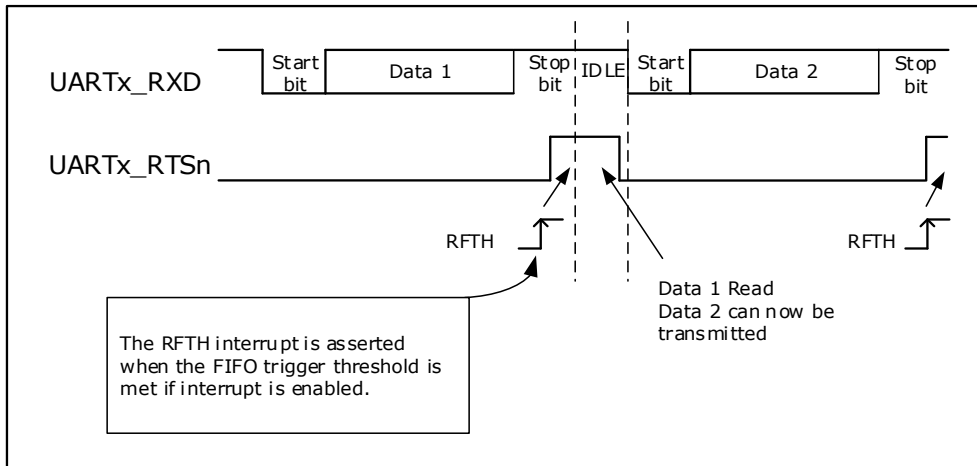


图 21-12 自动 RTS 流程控制

1. 3. 7. 2 Auto-CTS

当 Auto-CTS 被使能时(MCR 寄存器中的 AFCEN 为 1)，发送器在发送下一帧前检查 UART_CTSn 输入。如果 UART_CTSn 有效(被拉成低电平)，则发送 FIFO 内的数据，否则停止发送。若 UART_CTSn 在传输期间被变成无效，当前的传输完成后才停止发送。

只要 UART_CTSn 输入一变换状态，硬件就自动设置 RIF 寄存器中的 DCTS 位。它表明接收器是否准备好进行通信。如果设置了 IER 寄存器的 DCTS 位，则产生中断。

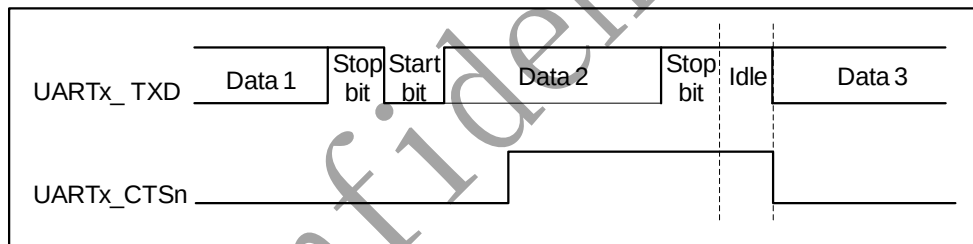


图 21-13 自动 CTS 功能

自动流控制可以减少系统中断，当自动流控制使能，CTS_n 状态不会触发系统中断。因为是设备自动地控制其收发器。当没启用自动流控制，收发器将发送的信息存储到 Tx FIFO，产生接收溢出。

1. 3. 7. 3 RS485 驱动使能(DE)

驱动使能功能使用 RS485 寄存器的 AADACEN 位为 1 使能。它允许用户通过 DE（驱动使能）信号来激活外部收发器的控制端。滞后时间是一个发送消息的最后一个字节的停止位和释放 DE 信号之间的时间间隔，这个时间可以在 RS485 寄存器中的 DLY 位设置。DE 信号的极性则可以通过 RS485 寄存器中的 AADINV 位中设置并进行选择。

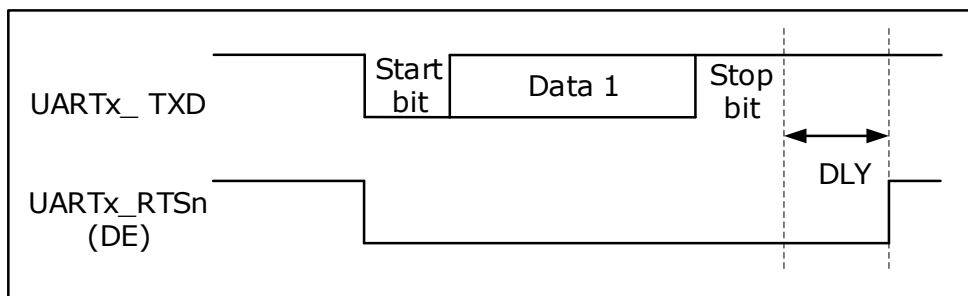


图 21-14 当 AADINV=0 时，驱动使能

1. 3. 8 UART Modbus 通信

UART 提供对 Modbus/RTU 和 Modbus/ASCII 协议实现的基本支持。Modbus/RTU 是一个半双工，块式传输协议。协议的控制部分（地址识别，块完整性控制和通信解释）必须由软件来完成。

UART 提供对块尾检测的基本支持，无需软件的经常性介入。

◆ Modbus/RTU

这个模式下，块尾一般公认为一个超过 2 个字符长度的静默阶段。通过一个可设置的超时长度功能来实现。

超时功能和相应的中断必须通过 RTOR 寄存器中的 RTOEN 位和 IER 寄存器中的 RXTO 位来打开。RTOR 寄存器中的 RTO 位要填入一个与超时长度相当的数字（例如 2 个字符长度为 22 个位长）。当接收线路保持空闲阶段达到这个长度时，在最后一个停止位被收到之后，会产生一个中断，表示当前的块接收已经完毕。

◆ Modbus/ASCII

在这个模式，块尾被公认为回车字符（CR/LF）串。UART 用字符匹配功能实现这个机制。

将 LF 的 ASCII 码写到 ADD[7:0]区域，IER 寄存器中的 CMIE 位来打开，那么软件就会在收到 LF 字符后或者能够在 DMA 缓冲区中找到 CR/LF 字符时得到提示。

1. 3. 9 UART 奇偶校验控制

设置 LCR 寄存器中的 PCE 位，可以使能校验控制(发送时生成一个校验位，接收时进行校验检查)。根据 DLS 位定义的帧长度，可能的 UART 帧格式栏在下表中。

DLS[1:0]	PCE bit	UART frame
00	0	Start bit 8bit data Stop bit
00	1	Start bit 8bit data Parity bit Stop bit
01	0	Start bit 7bit data Stop bit
01	1	Start bit 7bit data Parity bit Stop bit
10	0	Start bit 6bit data Stop bit
10	1	Start bit 6bit data Parity bit Stop bit
11	0	Start bit 5bit data Stop bit

11	1	Start bit 5bit data Parity bit Stop bit

表 21-3 帧格式

◆ 奇校验

校验位的内容使得一帧中的 8,7,6 或 5 个 LSB 数据以及校验位中'1'的个数为奇数。

例如：数据=00110101，有 4 个'1'，如果选择奇校验(在 LCR 寄存器中的 PS=0)，校验位将是'1'。

◆ 偶校验

校验位的内容使得一帧中的 8,7,6 或 5 个 LSB 数据以及校验位中'1'的个数为偶数。

例如：数据=00110101，有 4 个'1'，如果选择偶校验(在 LCR 寄存器中的 PS=1)，校验位将是'0'。

◆ 接收时的校验检查

如果校验检查失败，SR 寄存器中的 PERR 标志会被置 1，如果 IER 寄存器中的 RXBERR 为 1，将引发相应中断。

◆ 发送时的校验生成

如果 LCR 寄存器的 PCE 位被置 1，写进数据寄存器的数据的 MSB 位被校验位替换后发送出去(如果选择奇校验奇数个'1'，如果选择偶校验偶数个'1')。

1. 3. 10 UART 多处理器通信

1. 设置 DLS 位为 8 位字长(第 9bit 为判断地址或数据)
2. 设置 RS485 寄存器的 AADEN 位为 1 以进入模式。
3. 设置 RS485 寄存器的 ADDR 位配置匹配地址

可以将多个 UART 连接成一个网络来实现多机通讯。例如某个 UART 设备可以是主，它的 TX 输出和其他 UART 从设备的 RXD 输入相连接；UART 从设备各自的 TXD 输出逻辑地与在一起，并且和主设备的 RX 输入相连接。

在多处理器配置中，我们通常希望只有被寻址的接收者才被激活，来接收随后的数据，这样就可以减少由未被寻址的接收器的参与带来的多余的 UART 服务开销。

未被寻址的设备可启用其静默功能进入静默模式。要使用静默模式功能，RS485 寄存器的 AADEN 位必须被置 1。

在这个模式里，如果 MSB 是 1，该字节被认为是地址，否则被认为是数据。在一个地址字节中，目标接收器的地址被放在 RS485 寄存器的 ADDR 位。

如果接收到的字节与它的编程地址不匹配时，UART 进入静默模式。当 UART 进到静默模式后，接收字节时既不会动 RIF 寄存器的 RFTH 位标志也不会产生中断或发出 DMA 请求。

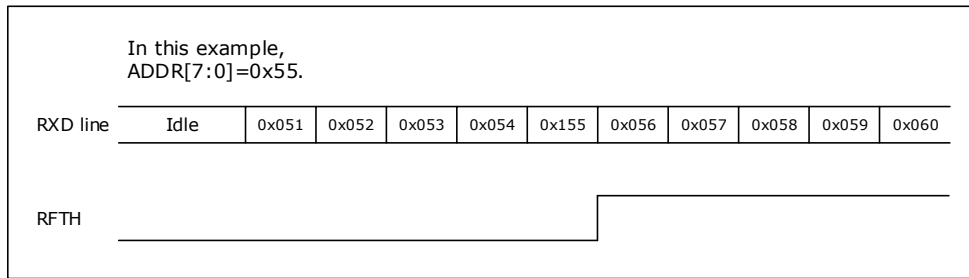


图 21-15 Mute mode using address mark detection

1. 3. 11 UART LIN (本地互连)模式

◆ LIN 发送

和常规的 UART 发送相同，但包含下列区别：

1. 设置 DLS 位为 8 位字长
2. 设置 LIN 寄存器的 LINEN 位为 1 以进入 LIN 模式。这时，置 LIN 寄存器的 LINBKRQ 位为 1 将发送 13 位'0'作为断开符号。然后发两位'1'，以允许对下一个开始位的检测。

◆ LIN 接收

当 LIN 模式被使能时(LIN 寄存器的 LINEN 位为 1)，断开符号检测电路被激活。该检测完全独立于 UART 接收器。不管是在总线空闲时还是在发送某数据帧期间，断开符号只要一出现就能检测到。

一旦接收器被激活(LCR 寄存器的 RXEN 位为 1)，电路就开始监测 RXD 上的起始信号。监测起始位的方法同检测断开符号或数据是一样的。当起始位被检测到后，电路对每个接下来的位，在每个位的第 8, 9, 10 个过采样时钟点上进行采样，就像针对数据一样。如果 10 个(当 MCR 寄存器中的 LINBDL=0)或 11 个(当 MCR 寄存器中的 LINBDL=1)连续位都是'0'，并且又跟着一个定界符，RIF 寄存器的 LINBK 位标志就会被置 1。如果 IER 寄存器的 LINBK 位为 1，还会产生中断。在确认断开符号前，要检查定界符，因为它表示 RXD 线已经回到高电平。

如果在第 10 或 11 个采样点之前采样到了'1'，检测电路取消当前检测并重新寻找起始位。

如果 LIN 模式被禁止，接收器继续如正常 UART 那样工作，不再考虑检测断开符号。

如果 LIN 模式被激活(LINEN=1)，只要一发生帧错误(例如：停止位检测到'0'，这种情况出现在断开符号被接收到的时候)，接收器就停止，直到断开符号检测电路接收到一个'1'(这种情况发生于断开符号没有完整的发出来)，或一个定界符(这种情况发生于已经检测到一个完整的断开符号)。

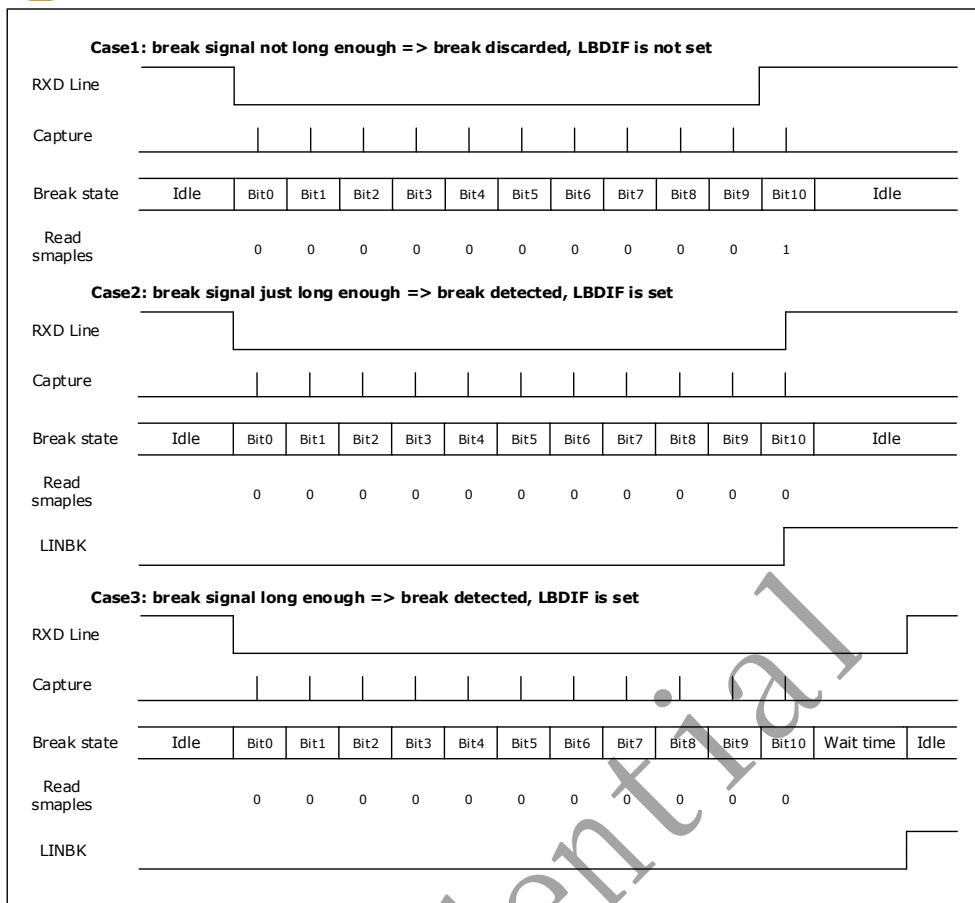


图 21-16 LIN 模式下的断开检测(11 位断开长度 - 设置了 LBDL 位)

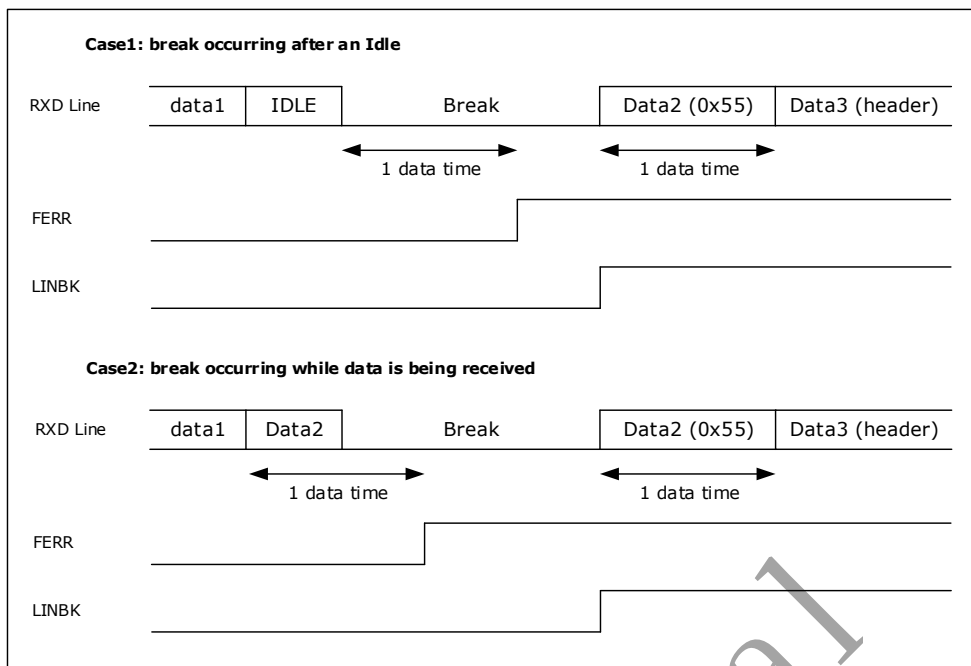


图 21-17 LIN 模式下的断开检测与帧错误的检测

1. 3. 12 UART 单线半双工通信

UART 可以配置成遵循单线半双工协议。在单线半双工模式下，TXD 和 RXD 引脚在芯片在内部是连在一起的。使用控制位“HALF DUPLEX ENABLE”(MCR 寄存器中的 HDEN 位)选择半双工或全双工通信。

◆ 当 HDEN 为‘1’时：

- TX 和 RX 引脚在芯片在内部是连在一起的
- RX 不再被使用
- 当没有数据传输时，TX 总是被释放。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。这就意味该 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)。

除此以外，通信与正常 UART 模式类似。由软件来管理线上的冲突(例如通过使用一个中央仲裁器)。特别注意，发送从不会被硬件所阻碍。只要数据一写到数据寄存器中，发送就会开始。

1. 3. 13 UART 智能卡模式

1. 8 位数据位加校验位：即 LCR 寄存器中 DLS=11，PCE=1

2. 0.5/1.5 个停止位：即 LCR 寄存器的 STOP=0、1

3. 设置 SCARD 寄存器的 SCEN 为 1 以进入 Smart card 模式

在 T=0（字符）模式中，保护时间内，校验错误在字符发送完毕后被提出。

所示为在数据线上有校验错误和没有校验错误时的情形。

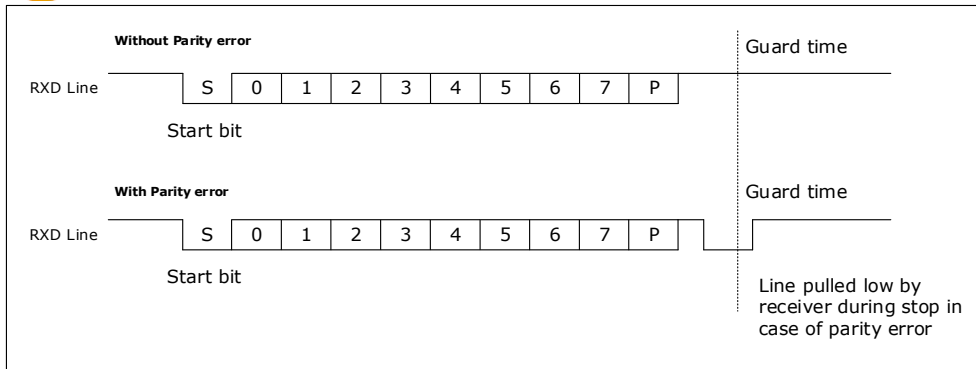


图 21-18 ISO 7816-3 异步通信协议

当连接到智能卡时，UART 的 TXD 输出脚和智能卡通过同一根双向数据线进行通讯。所以 TXD 引脚必须配置成开漏状态。

智能卡模式实现一套单线半双工通讯协议。

- ◆ 发送数据在确认经过最少 1/2 个波特率时钟周期后从发送移位寄存器中移出。正常操作模式下下一个满的发送移位寄存器的内容在下一个波特时钟沿开始移出。在智能卡模式下则会准确的加入一个 1/2 波特时钟的延迟。
- ◆ 发送时，如果智能卡检测到一个校验错误，它会用拉低数据线的方式（NACK）将这个情况提示给 UART。这个 NACK 信号（拉低传输线 1 个波特时钟）将在发送方（配置为 1.5 个停止位）引起一个帧错误。UART 能够捕获这个情况并依照协议自动进行重发。重试的次数可以在 SCCNT 区域进行设置。如果 UART 连续收到 NACK 信号，并重试达到设定的次数，UART 会停止发送并提示帧错误。
- ◆ 智能卡发送自动重试：UART 在收到 NACK 信号后，会等待 2.5 个波特周期，然后再给出重发字符的起始位。最后一个重发字符被顺利送达后，RIF 寄存器的 TSEMPY 位就立即置 1 了，这里没有保护时间的。如果软件想要再重发，这里必须确保最少 2 个波特周期的延时，这是标准规定的。
- ◆ 如果使用 1.5 个停止位，并且在一帧数据的接收过程中检出校验错误，传输线会在在一帧传输完后被拉低 1 个波特周期。这是在向智能卡表示，传给 UART 的数据不完全正确。如果 NACK 控制位为 1，校验错误会导致接收方（UART）发一个 NACK 出来，否则就不发（在 T=1 的模式下）。如果接收字符不正确，也就不会引起 RFNE 的变化以及接收 DMA 请求。按照协议规定，智能卡必须再发一个相同的字符。如果按照 SCCNT 域的设定重试次数达到了而收到的字符还是错的，UART 就不会再发 NACK 了，而是提示一个校验错误。
- ◆ 智能卡接收的自动重试：如果 UART 向智能卡发了 NACK 但卡却没有重发字符回应，就会使 SR 寄存器的 RSBUSY 标志残留下来。
- ◆ 在发送过程中，UART 会在两次成功的发送字符之间插入一个保护时间（具体长度在保护时间寄存器中设定）。保护时间长度是从前一个字符的停止位开始计算的，由 SCARD 寄存器的 GT[7:0]位设置，区域必须按照所须的 CGT（7816-3 标准要求的保护时间长度）设置一个值，最少是 12（相当于 1 个字符的时间）。
- ◆ 可以调整保护时间的设置来延迟 RIF 寄存器的 TSEMPY 标志的置 1。
- ◆ 正常操作中，TSEMPY 标志是在发送移位寄存器为空，并且也没有进一步的发送

要求的是后被置 1 的。而在智能卡模式下，发送移位寄存器为空的事件只会触发保护时间计数器，令其计数直到所设定的保护时间过完。在这个过程中，TSEMPY 会强制为 0。当保护时间计数器达到所设定的上限，TSEMPY 才会被置 1。

- ◆ 对 TSEMPY 标志的撤销不受智能卡模式的影响。
- ◆ 如果在发送过程的最后检出帧错误（就是从接收方传来一个 NACK），这个 NACK 是不会被当作起始位来检测的。根据 ISO 协议，接收到的 NACK 的持续时间可以是 1 或 2 个波特时钟周期。
- ◆ 在接收器这边，如果检测到一个校验错误，并且已经发送 NACK，接收器也不对把 NACK 当作起始位。

注： 断开字符在智能卡模式中没有意义。一个带帧错误的 0x00 数据将被当成数据而不是断开符号。

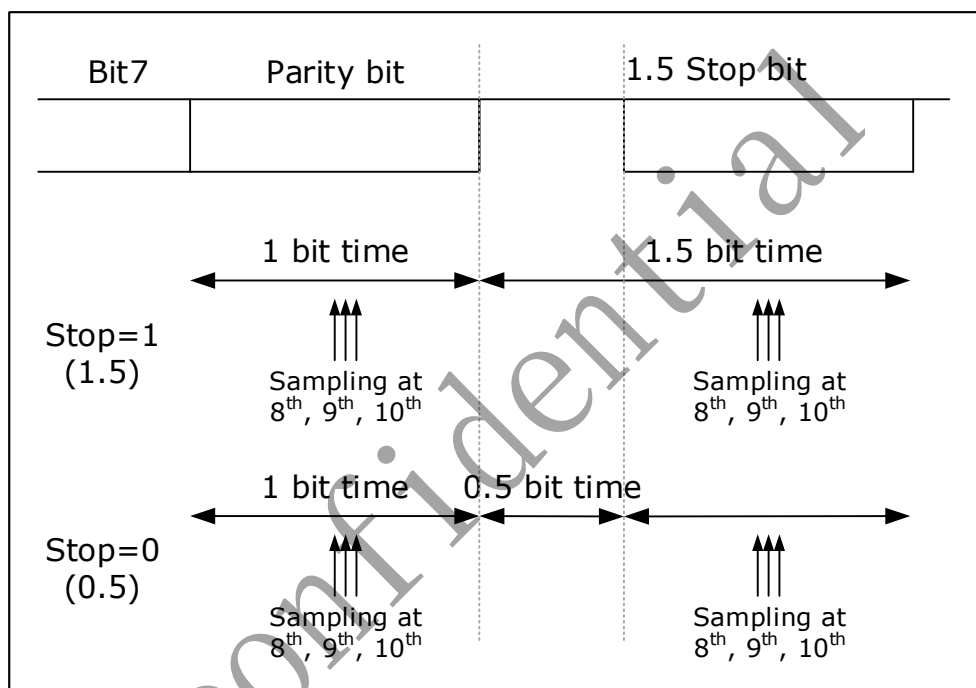


图 21-19 校验错误检测

UART 可以通过 SCLK 脚向智能卡提供时钟。智能卡模式中，SCLK 和通讯没有关系，只是通过一个 5 位的预分频器从内部外设时钟源得到时钟信号。这个分频系数在 SCARD 寄存器的 PSC 位设置。SCLK 频率可以设置在 $f_{ck}/2$ 到 $f_{ck}/64$ 之间, f_{ck} 指外设输入时钟。

块模式(T=1)

在块模式（T=1）中，校验错误是要通过将 SCARD 寄存器的 SCNACK 位清零来关闭的。

当要从智能卡读取数据时，在块模式下，软件必须将 RTOR 寄存器的 RTO 位设为 BWT（块等待时间）-11 的值。如果这个时间到期了，还没有收到卡的回应，将会引起 RIF 寄存器中的 RXTO 位为 1，若 IER 寄存器中的 RXTO 为 1，则会引起中断。如果超时之前收到了第一个字节，若 FCR 寄存器的 RXTH 位为 0，则会引起 RIF 寄存器中的 RFTH 位为 1，若 IER 寄存器中的 RFTH 为 1，则会引起中断。

在收到第一个字节之后，必须将 RTO 寄存器设置成 CWT（字符等待时间）-11 的值，这是为了允许在两个连续的字符之间自动检测最大等待时间。这个时间以波特时间作为单位。如果智能卡在前一个字符发送结束后到设定的 CWT 周期之间没有发送新的字符，UART 会通过 RXTO 标志提示给软件，若 IER 寄存器中的 RXTO 为 1，则会引起中断。

注：在 STOP=0,1 的情况下，RTO 计数器从最后一个字符的第一个停止位开始计数。按照智能卡协议的定义，BWT/CWT 值是从最后一个字符的起始位开始计算的。所以 RTO 寄存器必须为 BWT-11 或者 CWT-11，从而将各自的字符长度也算在内。

有一个块长度计数器可以统计 UART 收到的全部字符的个数。这个计数器在 UART 开始发送的时候（RXEN=0）自动清零。块长度信息位于智能卡发出数据块的第三字节（序言部分）。这个值必须写入到 SCARD 寄存器的 BLEN 位。当使用 DMA 模式，从块的开始前，这个寄存器必须设定为最小值（0x0）。为了得到这个值，在收到第 4 个字节的时候会引发一个中断。软件必须从接收缓冲区中读取到这个长度域（第三字节）。

在中断驱动接收模式，块的长度可以由软件提取出来并作检查。在块开始前，BLEN 可以设为最大值（0xFF）。实际值则要在收到第三字节之后被写到寄存器中。

如果块格式为使用 LSI 纵向冗余校验（1 个校验自己），BLEN 就等于 LEN。如果块格式使用 CRC 循环冗余校验（2 个校验字节），BLEN 就等于 LEN+1。整个块长度（包括序言部分，校验和信息区域）等于 BLEN+4。块尾将通过 EOBIF 标志以及相应中断（EOBIE 位被置 1 时）提示给软件。

在块长度出错的情况下，块尾则会引发 RXTO 中断（字符等待时间溢出）。

注：（LSI/CRC）校验码的验算需要由软件来完成。

◆ 直接和反向转换

智能卡协议定义两种转换方式：直接方式和反向方式。

直接转换方式定义如下：低位在前，逻辑位为 1 相当于传输线高电平，采用偶校验。要使用这个转换方式，下列控制位必须设置：MSBFIRST=0，DATAINV=0(默认值)。

反向转换方式定义如下：高位在前，逻辑位为 1 相当于传输线低电平，采用奇校验。要使用这个转换方式，下列控制位必须设置：MSBFIRST=1，DATAINV=1。

注：当逻辑数据为反转状态时（0=H，1=L），校验位也是采用同样的逻辑（0 是高，1 是低）。

为了确认卡的转换模式，卡会将初始化字符 TS 作为对 ATR（应答复位）帧的回应。

TS 的两种可能的模式为：LHHL LLL LLH 和 LHHL HHH LLH。

(H) LHHL LLL LLH 将启用反向转换模式：状态低对应数据 1，第二个时刻包含的信息为数据

的最高位。用反向方式解码时，传达的字节等于'3F'。

(H) LHHL HHH LLH 将启用直接转换模式：状态高对应数据 1，第二个时刻包含的信息为数据的最低位。用直接方式解码时，传达的字节等于'3B'。

当在第 2 到第 10 个时刻之间的 9 个时间段中有偶数个 1 时，字符校验正确。

由于 UART 并不知道用哪种转换模式，所以它要能够确认转换的样式并且照着执行。这个确认工作不是由硬件完成，而是由软件去判断。此外，假设 UART 被配置为直接转换模式（默认），而卡的答复是按反向转换的，TS=LHHL LLL LLH => UART 收到的字符会是'03'并且校验规则是奇校验。

因此，有两个方法可以用来确认 TS 的转换样式：

方法 1: UART 被设置为标准智能卡模式，直接转换。这时，TS 样式确认产生出一个校验错误中断并发送一个错误信号给卡。

校验错误中断提示软件说卡在直接转换模式下没有给出正确的应答。软件来重新设置 UART 为反向转换模式。

收到错误信号后，卡会再次发送一个同样 TS 字符，由于 UART 已经被重新设置过，这次就会被正确的接收了。

这样，在对校验错误中断的回复中，软件可以对 UART 重新设置，从而向卡发出新的复位命令，然后重新等待 TS。

方法 2: UART 被设置为 9 位字长/无校验的模式，无反向转换。在这个情况下能够收到的两种 TS 样式如下：

(H) LHHL LLL LLH = 0x103 -> 选为反向转换模式

(H) LHHL HHH LLH = 0x13B -> 选为直接转换模式

软件检查接收到的字符，来判断该用哪一种转换模式，然后将 UART 设置成正确的模式来适应下一个字符的接收。

如果两个都不符合，那就向卡发送一个复位命令以重启新一轮会话。

1. 3. 14 UART 红外编解码器模块

设置 MCR 寄存器的 IREN 为 1 以进入 IrDA 模式。

IrDA SIR 物理层规定使用反向归零调制方案（RZI），该方案用一个红外光脉冲代表逻辑'0'

SIR 发送编码器对 UART 给出的 NRZ 比特流进行调制。输出脉冲流用来驱动一个外部的输出驱动器和红外 LED。UART 的 SIR ENDEC 最高只支持到 115.2Kbps 速率。在正常模式下，

脉冲宽度规定为一个位周期的 3/16。

SIR 接收解码器对来自红外接收器的归零位比特流进行解调，并将接收到的 NRZ 串行比特流输出到 UART。在空闲状态里，解码器输入通常是高(标记状态)。发送编码器输出的极性和解码器的输入相反。当解码器输入低时，检测到一个起始位。

- ◆ IrDA 是一个半双工通信协议。如果发送器忙(也就是 UART 正在送数据给 IrDA 编码器)，IrDA 接收线上的任何数据将被 IrDA 解码器忽视。如果接收器忙(也就是 UART 正在接收从 IrDA 解码器来的解码数据)，从 UART 到 IrDA 的 TX 上的数据将不会被 IrDA 编码。当接收数据时，应该避免发送，因为将被发送的数据可能被破坏。
- ◆ SIR 发送逻辑把'0'作为高脉冲发送，把'1'作为低电平发送。脉冲的宽度规定为正常模式时位周期的 3/16
- ◆ SIR 解码器把 IrDA 兼容的接收信号转变成给 UART 的比特流。
- ◆ SIR 接收逻辑把高电平状态解释为'1'，把低脉冲解释为'0'。
- ◆ 发送编码器输出与解码器输入有着相反的极性。当空闲时，SIR 输出处于低状态。
- ◆ IrDA 规范要求脉冲要宽于 1.41us。脉冲宽度是可编程的。接收器端的尖峰脉冲检测逻辑滤除宽度小于 2 个 PSC 周期的脉冲(PSC 是在 IrDA 低功耗波特率寄存器 SCARD 寄存器)。宽度小于 1 个 PSC 周期的脉冲一定被滤除掉，但是那些宽度大于 1 个而小于 2 个 PSC 周期的脉冲可能被接收或滤除，那些宽度大于 2 个周期的将被视为一个有效的脉冲。
- ◆ 接收器可以与一个低功耗发送器通信。
- ◆ 在 IrDA 模式里，LCR 寄存器中的 STOP 位必须配置成 1 个停止位。

IrDA 低功耗模式

发送器:

在低功耗模式，脉冲宽度不再持续 3/16 个位周期。而是原来的三分之一，他的频率最小可以是 1.42MHz。通常这个值是 1.8432MHz(1.42MHz<PSC<2.12MHz)。一个低功耗模式可编程分频器把系统时钟进行分频以达到这个值。

接收器:

低功耗模式的接收类似于正常模式的接收。为了滤除尖峰干扰脉冲，UART 应该滤除宽度短于 1 个 PSC 的脉冲。只有持续时间大于 2 个周期的 IrDA 低功耗波特率时钟(SCARD 寄存器中的 PSC)的低电平信号才被接受为有效的信号。

注:

- 1.宽度小于 2 个大于 1 个 PSC 周期的脉冲可能会也可能不会被滤除。
- 2.接收器的建立时间应该由软件管理。IrDA 物理层技术规范规定了在发送和接收之间最小要有 10ms 的延时(IrDA 是一个半双工协议)。

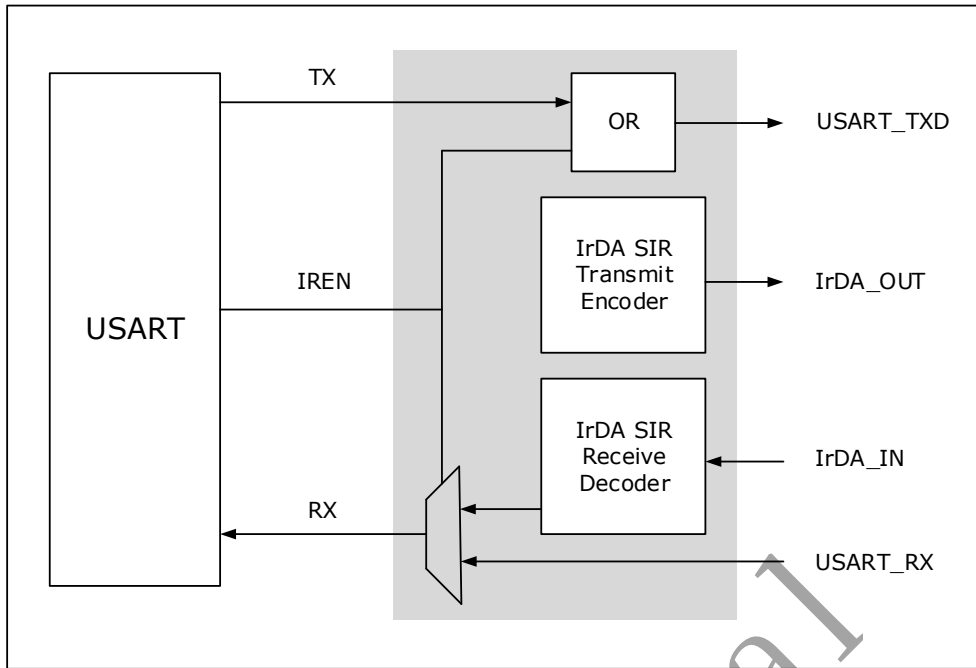


图 21-20 IrDA SIR ENDEC-block diagram

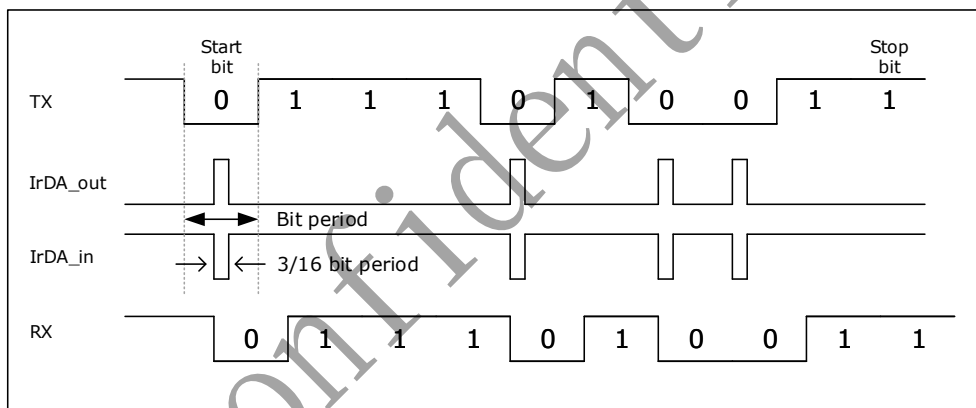


图 21-21 IrDA data modulation(3/16)-Normal Mode

1. 3. 15 UART 连续通信使用 DMA 模式

设置 MCR 的 RXDMAEN 为 1 始能 RX DMA 或 TXDMAEN 为 1 始能 TX DMA。

UART 可以利用 DMA 连续通信。Rx 缓冲器和 Tx 缓冲器的 DMA 请求是分别产生的。

利用 DMA 发送

使用 DMA 进行发送，可以通过设置 MCR 寄存器中的 TXDMAEN 位始能。在数据被预先放到 DMA 外设所设定的 SRAM 区域设置一个 DMA 通道给 UART 发送，要使用下列步骤（x 指通道号）：

- ◆ 在 DMA 控制寄存器上将 TDR 寄存器地址配置成 DMA 传输的目的地址。在每个发送事件后，数据将被传送到这个地址。
- ◆ 在 DMA 控制寄存器上将内存地址配置成 DMA 传输的源地址。在每个 TFTH 事件后，将从此存储器区读出数据并传送到 TDR 寄存器中。

- ◆ 在 DMA 控制寄存器中配置要传输的总的字节数。
- ◆ 在 DMA 寄存器上配置信道优先级。
- ◆ 根据应用程序的要求，配置在传输完成一半还是全部完成时产生 DMA 中断。
- ◆ 将 ICR 寄存器的 TFTH 位置 1 以清除 RIF 寄存器的 TFTH 标志。
- ◆ 在 DMA 寄存器上激活该通道。

注：若使用 FIFO，可一次写入多笔传输字节数，并设定 FCR 寄存器中的 TXTL 位，可设定 FIFO 深度为 0,2,4,8 个字符时，将新的数据将被传送到 TDR 寄存器中。

当传输完成 DMA 控制器指定的数据量时，DMA 控制器在该 DMA 通道的中断向量上产生一中断。

在发送模式下，当 DMA 传输完所有要发送的数据时，DMA 控制器设置 IFM 寄存器的 TFEMPTY 标志；监视 RIF 寄存器的 TFEMPTY 标志可以确认 UART 通信是否结束。这样可以在关闭 UART 或进入停机模式之前避免破坏最后一次传输的数据。软件必须等待 TSEEMPTY 被置 1。TSEEMPTY 标志在全部数据发送期间会是零，并且在最后一帧数据发送出去之后会由硬件置 1。

利用 DMA 接收

使用 DMA 进行接收，可以通过设置 MCR 寄存器中的 RXDMAEN 始能。当收到一个字节数据时，从 RDR 寄存器取出来的数据会被转移到 DMA 外设中指向的 SRAM 区域将一个 DMA 通道设置给 UART 接收，要按照下列步骤：

- ◆ 在 DMA 控制寄存器上将 RDR 寄存器地址配置成 DMA 传输的源地址。在每个 RFTH 事件后，数据将从这个地址取走。
- ◆ 在 DMA 控制寄存器上将内存地址配置成 DMA 传输的目标地址。在每个 RFTH 事件后，数据将从 RDR 寄存器取往这个目标地址。
- ◆ 在 DMA 控制寄存器中配置要传输的总的字节数。
- ◆ 在 DMA 寄存器上配置信道优先级。
- ◆ 根据应用程序的要求，配置在传输完成一半还是全部完成时产生 DMA 中断。
- ◆ 在 DMA 寄存器上激活该通道。

注：若使用 FIFO，可一次接收多笔传输字节数，并设定 FCR 寄存器中的 RXTL，可设定 FIFO 深度为 1,4,8,14 个字符时，将新的数据将被传送到 RDR 寄存器中。

当传输完成 DMA 控制器指定的数据量时，DMA 控制器在该 DMA 通道的中断向量上产生一中断。

多缓冲器通信中的错误标志和中断产生

在多缓冲器通信的情况下，通信期间如果发生任何错误，会在当前字节传输后将错误标志置 1。如果中断使能位被置 1，将产生中断。在单个字节接收的情况下，和 IFM 寄存器中的 RXBERR 和 RFOERR 一起被置起的帧错误和溢出错误，有单独的的错误标志中断使能位；如果被置 1 了，会在当前字节传输结束后，产生中断。

1. 3. 16 UART 中断

◆ UART_IER

中断致能寄存器，此位设定 1 时，表示开启中断功能，并且同时反映在 IVS 寄存器。此寄存器属于 Write Only，并且只允许写入 1，无法写 0 取消致能中断设定。

◆ UART_IDR

中断禁能寄存器，此位设定 1 时，表示关闭中断功能，并且同时反映在 IVS 寄存器。此寄存器属于 Write Only，并且只允许写入 1，无法写 0 取消禁能中断设定。

◆ UART_IVS

中断有效位状态寄存器，反映 IER 与 IDR 寄存器所设定的结果。0:中断禁能
1:中断致能

◆ UART_RIF

无屏蔽中断旗标寄存器，反映所有发生中断事件的状态，无论 IVS 是否有致能中断，皆会反映在此寄存器中，主要提供使用者监控无屏蔽的中断位，使否有错误事件发生。

◆ UART_IFM

屏蔽中断旗标寄存器，记录中断致能位所发生中断事件。0:无中断事件 1:发生中断事件

◆ UART_ICR

中断清除寄存器，此位设定 1 时，清除中断旗标 RIF 与 IFM，此寄存器属于 Write One Clear，并且只允许写入 1 清除，无法写 0 清除。

在 UART 中，配置了 17 种中断，分别为下表。

Interrupt event	Interrupt flag
Receiver Byte error	RXBERR
Auto-Baud rate detection End	ABEND
Auto-Baud rate detection Timeout	ABTO
Delta CTS status	DCTS
Receiver Timeout	RXTO
Address match	ADDRM
LIN break detection	LINBK
End of block	EOB
START bit Noise detection	NOISE
Receive FIFO trigger threshold	RFTH
Receive FIFO full	RFFULL
Receive FIFO overrun	RFOERR

Receive FIFO underrun	RFUERR
Transmit shift register empty	TSEMPY
Transmit FIFO trigger threshold	TFTH
Transmit FIFO empty	TFEMPTY
Transmit FIFO overrun	TFOVER

表 21-4 中断表

1.4 UART 寄存器描述

1.4.1 接收数据寄存器(UART_RDR)

		Address offset:			0x00										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								RDR[8:0]							

Name	Bit	Type	Reset value	Description
Reserved	31:9	N/A	0x00	保留，保持复位值
RDR	8:0	RO	0x0	RDR: 接收数据寄存器，包含所收到的字节 RDR 寄存器提供输入移位寄存器和内部总线间的并行接口。当接收数据时打开了校验位，读这个寄存器得到的最高位是校验位。 Bit 8 仅用在 RS485 模式

1.4.2 发送数据寄存器(UART_TDR)

		Address offset:				0x04									
		Reset value:				0x0000 0000									
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								TDR[8:0]							

Name	Bit	Type	Reset value	Description
Reserved	31:9	N/A	0x00	保留，保持复位值
TDR	8:0	R/W	0x0	TDR: 发送数据寄存器，用于写入要发送的数据字节 TDR 寄存器提供发送移位寄存器和内部总线间的并行接口。当发送的时候设置了校验功能（PCE=1），向最高位（位 7 还是位 8 取决于设置的字长）写入的信息是无效的，因为它总是要被校验位代替了之后再发送的。Bit 8 仅用在 RS485 模式

1. 4. 3 波特率寄存器 (UART_BRR)

		Address offset:		0x08											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BRR[15:0]															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
BRR	15:4	R/W	0x0	BRR[15:4]: UARTDIV 的整数部分 这 12 位定义 UART 分频器除法因子的整数部分 此寄存器只能在 UART 未被使能时改写 (RXEN=0)。 注：使用自动波特率功能时则可自动写入
BRR	3:0	R/W	0x0	BRR[3:0]: UARTDIV 的小数部分 这 4 位定义 UART 分频器除法因子的小数部分 此寄存器只能在 UART 未被使能时改写 (RXEN=0)。 注：使用自动波特率功能时则可自动写入

1. 4. 4 线路控制寄存器(UART_LCR)

		Address offset:		0x0C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	SWAP	TXINV	RXINV	DATAINV	MSB	RTOEN	BRWEN	BC	RXEN	PS	PCE	STOP	DLS[1:0]		

Name	Bit	Type	Reset value	Description
Reserved	31:14	N/A	0x00	保留，保持复位值
SWAP	13	R/W	0x0	SWAP: 交换 TX/RX 引脚功能使能 该 Bit 位由软件置 1 和清零 0: 交换禁止，TX/RX 引脚功能按照标准引脚分配使用 1: 交换使能，TX/RX 引脚功能交换使用。 这用于和其它 UART 口进行交叉互联的时候。 此寄存器只能在 UART 未被使能时改写 (RXEN=0)
TXINV	12	R/W	0x0	TXINV: TX 引脚电平反向使能 该 Bit 位由软件置 1 和清零 0: TX 脚信号工作于标准逻辑电平 1: TX 脚信号被反向。(这可以用于 TX 线上带有外部反相器的时候) 此寄存器只能在 UART 未被使能时改写

RXINV	11	R/W	0x0	RXINV : RX 引脚电平反向使能 该 Bit 位由软件置 1 和清零 0: RX 脚信号工作于标准逻辑电平 1: RX 脚信号被反向。(这可以用于 RX 线上带有外部反相器的时候) 此寄存器只能在 UART 未被使能时改写 (RXEN=0)
	10	R/W	0x0	DATAINV: 二进制数据反向使能 该 Bit 位由软件置 1 和清零 0: 数据寄存器中的逻辑数据在发送和接收的时候, 采用正/直接逻辑。(1=H, 0=L) 1: 数据寄存器中的逻辑数据在发送和接收的时候, 采用负/反向逻辑。(1=L, 0=H). 校验位也一样反向。 此寄存器只能在 UART 未被使能时改写 (RXEN=0)
	9	R/W	0x0	MSB: 高位在前使能 该 Bit 位由软件置 1 和清零 0: 数据在发送和接收的时候, 采用起始位在前, 后面跟着第 0 位的顺序。 1: 数据在发送和接收的时候, 采用起始位在前, 后面跟着最高位 (位 8/7/6/5) 的顺序。 此寄存器只能在 UART 未被使能时改写 (RXEN=0)
	8	R/W	0x0	RTOEN: 接收器超时使能 该 Bit 位由软件置 1 和清零 0: 接收器超时检测功能关闭 1: 接收器超时检测功能开启 当这个功能开启, 只要 RX 线发现空闲 (不是接收) 达到 RTOR 寄存器 (接收超时寄存器) 中设置的时间长度后, UART_RIF 寄存器中的 RTOIF 标志会被硬件置 1。
	7	R/W	0x0	BRWEN: 波特率配置使能 该 Bit 位由软件置 1 和清零 0: 波特率配置禁止 1: 波特率配置使能 此寄存器只能在 UART 未被使能时改写 (RXEN=0)
	6	R/W	0x0	BC: 断 开 控 制 该 Bit 位由软件置 1 和清零 0: 断开禁止 1: 断开使能, 会使得 TX 输出为 0 此寄存器只能在 UART 未被使能时改写 (RXEN=0)
	5	R/W	0x0	RXEN: 接收器使能 该 Bit 位由软件置 1 和清零 0: 接收器被关闭 1: 接收器被打开并开始等待起始位
	4	R/W	0x0	PS: 校 验 选 择 该 Bit 位由软件置 1 和清零 这个 Bit 位选择在校验生成和检测功能被打开的时候 (PCE=1) 使用奇校验还是使用偶校验 0: 偶校验 1: 奇 校 验 此寄存器只能在 UART 未被使能时改写 (RXEN=0)
	3	R/W	0x0	PCE: 校验控制使能位 该 Bit 位由软件置 1 和清零 使能校验功能, 计算好的校验位被插入到最高位, 并检测接收数据的校验位 (接收与发送功能) 0: 校验控制禁止 1: 校验控制使能 此寄存器只能在 UART 未被使能时改写 (RXEN=0)

STOP	2	R/W	0x0	STOP: 停止位选择 该 Bit 位由软件置 1 和清零 普通模式： 0: 1 个停止位 1: 2 个停止位 (如果 DLS[1:0]=11, 则 1.5 个停止位) 智能卡模式： 0: 0.5 个停止位 1: 1.5 个停止位 此寄存器只能在 UART 未被使能时改写 (RXEN=0)
	1:0	R/W	0x0	DLS: 数据长度选择 该 Bit 位由软件置 1 和清零 00: 5-bit 数据格式 01: 6-bit 数据格式 10: 7-bit 数据格式 11: 8-bit 数据格式 注: 9-bit 数据格式用于 RS485 地址模式 此寄存器只能在 UART 未被使能时改写 (RXEN=0)

1. 4. 5 模式控制寄存器(UART_MCR)

		Address offset:		0x10											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										HDSEL	CLKEN	SCCNT		SCNACK	SCEN
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABR REPT	ABRMOD		ABREN	DMAEN	LINBDL	BKREQ	LINEIN	AADINV	AADDIR	AADNOR	AADEN	RTSCTL	AFCEIN	LBEN	IREN

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留, 保持复位值
HDSEL	22	R/W	0x0	HDSEL: 半双工选择 该 Bit 位由软件置 1 和清零 0: 不选择半双工模式 1: 选择半双工模式
CLKEN	21	R/W	0x0	CLKEN: 智能卡时钟使能 该 Bit 位由软件置 1 和清零 0: 禁止 CK 引脚 1: 使能 CK 引脚
SCCNT	20:18	R/W	0x0	SCCNT: 智能卡重试计数 指定智能卡模式中接收和发送的重试次数。 在发送模式下, 指定在产生发送错误 (FE=1) 前自动发送重试次数。 在接收模式下, 指定在产生接收错误前 (RXNE 和 PCE=1) 所作的错误接收的尝试的次数。 0x0: 重发功能关闭 - 在发送和接收模式下不进行自动重发。 0x1 to 0x7: 自动重传的尝试次数 (在提示错误之前)
SCNACK	17	R/W	0x0	SCNACK: 智能卡 NACK 使能 此位由软件置 1 和清零 0: 在奇偶校验错误情况下, 不发送 NACK 1: 在奇偶校验错误情况下, 发送 NACK

SCEN	16	R/W	0x0	SCEN: 智能卡模式使能 该 Bit 位由软件置 1 和清零.该位用于打开智能卡模式。 0: 禁止智能卡模式 1: 使能智能卡模式
ABRREPT	15	R/W	0x0	ABRREPT: 重复自动波特率检测 此位由软件置 1 和清零. 此位用于使能自动波特率检测第一次失败后重复一次 0: 重复自动波特率检测禁止 1: 重复自动波特率检测使能
ABRMOD	14:13	R/W	0x0	ABRMOD: 自动波特率模式选择 该 Bit 位由软件置 1 和清零 00: 检测第一个下降沿到第二个下降沿 (检测 2 位) 01: 检测第一个下降沿到第一个上升沿 (检测 1 位) 10: 检测第一个下降沿到第一个上升沿 (检测 2 位) 11: 保留
ABREN	12	R/W	0x0	ABREN: 自动波特率检测使能 此位由软件置 1, 由硬件在完成波特率检测后清 0 0: 自动波特率检测被禁止。 1: 自动波特率检测被打开
DMAEN	11	R/W	0x0	DMAEN: DMA 使能 此 Bit 位由软件置 1 和清零. 此位使能 DMA 功能 0: 禁止 DMA 1: 使能 DMA
LBDL	10	R/W	0x0	LBDL: LIN 断开侦测字长设置 此 Bit 位由软件置 1 和清零 这个位用来选择使用 11 位还是 10 位断开检测。 0: 10 位断开检测 1: 11 位断开检测
BKREQ	9	WO	0x0	BKREQ: 断开请求 此 Bit 位由软件置 1, 然后由硬件自动清零在下一个时钟周期 此位写 1 将发送一个断开请求, 一旦传输状态机有效, 它将产生 N Bit 的低脉冲信号依据 LCR 寄存器中 DLS 控制位 0: LIN 模式断开请求禁止 1: LIN 模式断开请求使能
LINEN	8	R/W	0x0	LINEN: LIN 模式使能 此 Bit 位由软件置 1 和清零 0: LIN 模式禁止 1: LIN 模式使能
AADINV	7	R/W	0x0	AADINV: 自动地址侦测反向使能 此 Bit 位由软件置 1 和清零, 设置驱动使能引脚 (RTSn/DE)的输出电平。 0: 当发送器开始发送数据时, 驱动使能引脚输出 0, 发送完成且 FIFO 内无数据时, 驱动使能引脚输出 1 1: 当发送器开始发送数据时, 驱动使能引脚输出 1, 发送完成且 FIFO 内无数据时, 驱动使能引脚输出 0
AADDR	6	R/W	0x0	AADDR: 自动地址侦测自动控制使能 此 Bit 位由软件置 1 和清零 0: 自动地址侦测自动控制禁止 1: 自动地址侦测自动控制使能
AADNOR	5	R/W	0x0	AADNOR: 自动地址侦测普通模式使能 此 Bit 位由软件置 1 和清零 0: 自动地址侦测普通模式禁止 1: 自动地址侦测普通模式使能 注: 此位不能和 AADEN 同时有效

AADEN	4	R/W	0x0	AADEN: 自动地址侦测模式使能 此 Bit 位由软件置 1 和清零 0: 自动地址侦测模式禁止 1: 自动地址侦测模式使能
RTSCTRL	3	R/W	0x0	RTSCTRL: RTSn 设置控制 此 Bit 位由软件置 1 和清零 0: 自动流量控制禁止时, RTSn 电平为高 1: 自动流量控制使能时, RTSn 电平为低
AFCEN	2	R/W	0x0	AFCEN: 自动流量控制使能 此 Bit 位由软件置 1 和清零 0: 自动流控制禁止 1: 自动流控制使能
LBEN	1	R/W	0x0	LBEN: 环回模式使能 此 Bit 位由软件置 1 和清零。 此模式用于 UART 测试项目的诊断模式。在 UART 普通模式下运行, TX 引脚输出为高电平, 串行数据在内部回送至 RX。在此模式下, 所有终端都是正常运行的。 0: 环回模式禁止 1: 环回模式使能
IREN	0	R/W	0x0	IREN: 红外模式使能 此 Bit 位由软件置 1 和清零 0: 红外线模式禁止 1: 红外线模式使能

1.4.6 控制寄存器(UART_CR)

		Address offset:		0x14											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GT[7:0]								PSC[7:0]							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DLY[7:0]								ADDR[7:0]							

Name	Bit	Type	Reset value	Description
GT	31:24	R/W	0x0	GT: 保护时间设置 该 Bit 位由软件置 1 和清零 设置保护时间长度, 以使用波特时钟为单位。 在智能卡模式中使用。完成标志 (RIF 寄存器 TBC 位) 在保护时间过后置起。

PSC	23:16	R/W	0x0	PSC: 分频器数值 该 Bit 位由软件置 1 和清零 在红外低功耗和正常模式下: PSC[7:0] = IrDA 正常和低功耗模式波特率对 UART 时钟源进行分频以获得低功耗模式下的频率: 时钟源按寄存器给定的值来分频 (8 位有效): 00000000: 1 分 频 00000001: 2 分 频 00000010: 3 分 频 00000011: 4 分 频 ... 智能卡模式: PSC[4:0]: 预分频器值用于设定对 UART 时钟源的分频系数, 得到智能卡时钟。按寄存器中的值 (低 5 位有效) 乘以 2 得到的值作为分频系数来分频: 00000: 2 分频 00001: 4 分频 00010: 6 分 频 00011: 8 分频 ...
				DLY: 延 迟 数 值 该 Bit 位由软件置 1 和清零 包含一个方向控制 RTSn 延迟值。这个寄存器协同一个 8 位计数器一起工作, 根据除数位持续时间的数量
				ADDR: 地 址 匹 配 数 值 该 Bit 位由软件置 1 和清零 该位域给出本设备 UART 节点地址 这是多处理器通信下的静默模式中使用的, 使用地址标记来唤醒某个 UART 设备。

1. 4. 7 智能卡控制寄存器(UART_RTOR)

		Address offset:		0x18											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BLEN[7:0]								RTO[23:16]							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RTO[15:0]															

Name	Bit	Type	Reset value	Description
BLEN	31:24	R/W	0x0	BLEN: 块长度 该 Bit 位由软件置 1 和清零 设置了智能卡模式 T=1 的接收时的块长度。 例如 BLEN = 0 -> 0 个信号字符 BLEN = 1 -> 1 个信号字符 BLEN = 255 -> 255 个信号字符 这个功能也可以在其他模式中使用, 当 LCR 寄存器 RXEN 位清除时, 块长度计数器会重新计数

RTO	23:0	R/W	0xFF	RTO: 接收器超时数值
				该 Bit 位由软件置 1 和清零 设置接收超时时间, 使用波特率时钟的字长为单 位。在标准模式下, 接收最后一个字节后, 在超时时间内未检测到新的起始位, 将置起 RIF 寄存器的 RXT0 位, 此位由软件设置和清除。在智能卡模式下, 这个数值是用来实现 CWT 和 BWT

1. 4. 8 FIFO 控制寄存器(UART_FCR)

		Address offset:		0x1c											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													TXFL[4:3]		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXFL[2:0]				RXFL[4:0]				TXTL[1:0]		RXTL[1:0]		Reserved	TFRST	RFRST	FIFOEN

Name	Bit	Type	Reset value	Description
Reserved	31:18	N/A	0x00	保留, 保持复位值
TXFL	17:13	RO	0x0	TXFL: 发送 FIFO 中数据个数 用于指示发送 FIFO 中的数据量。从 0~16。
RXFL	12:8	RO	0x0	RXFL: 接收 FIFO 中数据个数 用于指示接收 FIFO 中的数据量。从 0~16。
TXTL	7:6	R/W	0x0	TXTL: 发送 FIFO 触发阈值 用于选择发送 FIFO 的触发阈值, 发送 FIFO 达到阈值则触发标志位生成。 00: 空 FIFO 01: 2 个字符在 FIFO 中 10: 4 个字符在 FIFO 中 11: 8 个字符在 FIFO 中
RXTL	5:4	R/W	0x0	RXTL: 接收 FIFO 触发阈值 用于选择接收 FIFO 的触发阈值, 接收 FIFO 达到阈值则触发标志位生成。 00: 1 个字符在 FIFO 中 01: 4 个字符在 FIFO 中 10: 8 个字符在 FIFO 中 11: 14 个字符在 FIFO 中
Reserved	3	N/A	0x00	保留, 保持复位值
TFRST	2	WO	0x0	TFRST: 发送 FIFO 重置使能 当设置该位为 1 时, 将清除发送 FIFO 中的数据, 并将 FIFO 视为空 FIFO。注意, 此位将在下一个时钟周期中自动清 0。 0: 发送 FIFO 重置禁止 1: 发送 FIFO 重置使能
RFRST	1	WO	0x0	RFRST: 接收 FIFO 重置使能 当设置该位为 1 时, 将清除接收 FIFO 中的数据, 并将 FIFO 视为空 FIFO。注意, 此位将在下一个时钟周期中自动清 0。 0: 接收 FIFO 重置禁止 1: 接收 FIFO 重置使能
FIFOEN	0	R/W	0x0	FIFOEN: FIFO 功能使能 0: 禁止 FIFO 功能

1: 使能 FIFO 功能

1.4.9 状态寄存器(UART_SR)

		Address offset:		0x20											
		Reset value:		0x0000 0660											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	CTS	DCTS	RFF	RFNE	TFEM	TFNF	BUSY	RFE	TEMT	TBEM	BF	FE	PE	ORE	DR

Name	Bit	Type	Reset value	Description
Reserved	31:15	N/A	0x00	保留，保持复位值
CTS	14	RO	0x0	CTS: Clear to Send 这个位是 CTSn 的补码。当 CTSn 被断言时，它表示调制解调器或数据已经准备好与 UART 进行交换数据。 0: CTSn 输入没有被断言 1: CTSn 输入被断言
DCTS	13	RO	0x0	DCTS: Delta Clear To Send
RFF	12	RO	0x0	RFF: 接收 FIFO 满状态 此 Bit 位由硬件进行置位和清零，当发送 FIFO 不再处于满状态 0: 接收 FIFO 处于非满状态 1: 接收 FIFO 处于满状态
RFNE	11	RO	0x0	RFNE: 接收 FIFO 非空状态 用于指示接收 FIFO 非空状态 0: 接收 FIFO 处于空状态 1: 接收 FIFO 处于非空状态
TFEM	10	RO	0x0	TFEM: 发送 FIFO 空状态 此 Bit 位由硬件进行置位和清零，当发送 FIFO 不再处于空状态 0: 发送 FIFO 处于不空状态 1: 发送 FIFO 处于空状态
TFNF	9	RO	0x0	TFNF: 发送 FIFO 非满状态 用于指示发送 FIFO 不满状态 0: 发送 FIFO 处于满状态 1: 发送 FIFO 处于不满状态
BUSY	8	RO	0x0	BUSY: 忙状态 该 Bit 位由硬件置 1 和清零 0: 发射机处于空闲状态 1: 发射机处于工作状态
RFE	7	RO	0x0	RFE: 接收 FIFO 数据错误 用于指示接收 FIFO 数据错误 0: 接收 FIFO 数据正确 1: 接收 FIFO 数据错误
TEMT	6	RO	0x0	TEMT: 发送机空状态 用于指示发送为空 0: 发送不为空 1: 发送为空
TBEM	5	RO	0x0	TBEM: 发送 Buffer 空状态 用于指示发送 Buffer 寄存器为空 0: 发送 Buffer 不为空 1: 发送 Buffer 为空

BI	4	RO	0x0	BI: 断开中断状态 用于指示检测到一个中断在串行输入数据序列中。此错误与 FIFO 顶部的数据相关。 0: 没有中断错误 1: 中断错误
	3	RO	0x0	FE: 帧错误 当接收到的 STOP 位字符为逻辑 0 时(即接收端没有有效的停止位), 出现帧错误。 此错误与 FIFO 顶部的数据相关。 0: 没有检测到帧错误 1: 检测到帧错误
	2	RO	0x0	PE: 校验错误 当在接收数据的时候发现校验错误, 这个位会由硬件置 1。 此错误与 FIFO 顶部的数据相关。 0: 没有校验错误 1: 校验错误
	1	RO	0x0	ORE: 溢出错误 当上一数据未读取就接收到新数据, 发生接收溢出错误, 读 UART_SR 清零该位。FIFO 模式下, 当 FIFO 已满且在传输时写入新数据时, 将发生溢出错误。FIFO 中的数据被保留, UART_RDR 寄存器中的数据丢失。 在非 FIFO 模式下, 如果上一数据未被读取就接收到新的字符。则发生溢出错误, 此时 UART_RDR 寄存器数据被覆盖 0: 没有溢出错误 1: 发生溢出错误
	0	RO	0x0	DR: 数据接收完成 用于指示已完成数据接收 0: 数据接收未完成 1: 数据接收完成

1. 4. 10 中断使能寄存器(UART_IER)

		Address offset:		0x24											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				CM	EOB	TC	LINBK	ABTO	ABE	BUSY	RTO	MDS	RXS	TXS	RXRD

Name	Bit	Type	Reset value	
Reserved	31:12	N/A	0x00	保留, 保持复位值
CM	11	WO	0x0	Character Match Interrupt Enable 0: 无 1: 中断使能.
EOB	10	WO	0x0	End of Block Interrupt Enable 0: 无 1: 中断使能.
TC	9	WO	0x0	Transmission Complete Interrupt Enable 0: 无 1: 中断使能.

LINBK	8	WO	0x0	LIN Break Detection Interrupt Enable 0: 无 1: 中断使能.	效
ABTO	7	WO	0x0	Auto Baud Rate Timeout Interrupt Enable 0: 无 1: 中断使能.	效
ABE	6	WO	0x0	Auto Baud Rate End Interrupt Enable 0: 无 1: 中断使能.	效
BUSY	5	WO	0x0	Busy Detect Interrupt Enable 0: 无 1: 中断使能.	效
RTO	4	WO	0x0	Character Timeout Interrupt Enable 0: 无 1: 中断使能.	效
MDS	3	WO	0x0	Modem Status Interrupt Enable 0: 无 1: 中断使能.	效
RXS	2	WO	0x0	Receiver Line Status Interrupt Enable 0: 无 1: 中断使能.	效
TXS	1	WO	0x0	THR Empty Interrupt Enable 0: 无 1: 中断使能.	效
RXRD	0	WO	0x0	Receive Data Available Interrupt Enable 0: 无 1: 中断使能.	效

1. 4. 11 中断禁止寄存器(UART_IDR)

		Address offset:		0x28											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				CM	EOB	TC	LINBK	ABTO	ABE	BUSY	RTO	MDS	RXS	TXS	RXRD

Name	Bit	Type	Reset value	
Reserved	31:12	N/A	0x00	保留, 保持复位值
CM	11	WO	0x0	Character Match Interrupt Disable 0: 无 1: 中断禁止
EOB	10	WO	0x0	End of Block Interrupt Disable 0: 无 1: 中断禁止
TC	9	WO	0x0	Transmission Complete Interrupt Disable 0: 无 1: 中断禁止
LINBK	8	WO	0x0	LIN Break Detection Interrupt Disable 0: 无 1: 中断禁止

ABTO	7	WO	0x0	Auto Baud Rate Timeout Interrupt Disable 0: 无 1: 中断禁止	效
ABE	6	WO	0x0	Auto Baud Rate End Interrupt Disable 0: 无 1: 中断禁止	效
BUSY	5	WO	0x0	Busy Detect Interrupt Disable 0: 无 1: 中断禁止	效
RTO	4	WO	0x0	Character Timeout Interrupt Disable 0: 无 1: 中断禁止	效
MDS	3	WO	0x0	Modem Status Interrupt Disable 0: 无 1: 中断禁止	效
RXS	2	WO	0x0	Receiver Line Status Interrupt Disable 0: 无 1: 中断禁止	效
TXS	1	WO	0x0	THR Empty Interrupt Disable 0: 无 1: 中断禁止	效
RXRD	0	WO	0x0	Receive Data Available Interrupt Disable 0: 无 1: 中断禁止	效

1. 4. 12 中断有效状态寄存器(UART_IVS)

		Address offset:		0x2C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				CM	EOB	TC	LINBK	ABTO	ABE	BUSY	RTO	MDS	RXS	TXS	RXRD

Name	Bit	Type	Reset value	
Reserved	31:12	N/A	0x00	保留，保持复位值
CM	11	RO	0x0	Character Match Interruptenable status 0: 中 断 禁 止 1: 中断使能
EOB	10	RO	0x0	End of Block Interruptenable status 0: 中 断 禁 止 1: 中断使能
TC	9	RO	0x0	Transmission Complete Interruptenable status 0: 中 断 禁 止 1: 中断使能
LINBK	8	RO	0x0	LIN Break Detection Interruptenable status 0: 中 断 禁 止 1: 中断使能
ABTO	7	RO	0x0	Auto Baud Rate Timeout Interruptenable status 0: 中 断 禁 止 1: 中断使能

ABE	6	RO	0x0	Auto Baud Rate End Interruptenable status 0: 中 断 禁 止 1: 中断使能
BZD	5	RO	0x0	Busy Detect Interruptenable status 0: 中 断 禁 止 1: 中断使能
RTO	4	RO	0x0	Character Timeout Interruptenable status 0: 中 断 禁 止 1: 中断使能
MDS	3	RO	0x0	Modem Status Interruptenable status 0: 中 断 禁 止 1: 中断使能
RXS	2	RO	0x0	Reveiver Line Status Interrupt enablestatus 0: 中 断 禁 止 1: 中断使能
TXS	1	RO	0x0	THR Empty Interruptenable status 0: 中 断 禁 止 1: 中断使能
RXRD	0	RO	0x0	Reveive Data Available Interruptenable status 0: 中 断 禁 止 1: 中断使能

1. 4. 13 原始中断标志寄存器(UART_RIF)

		Address offset:		0x30											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				CM	EOB	TC	LINBK	ABTO	ABE	BUSY	RTO	MDS	RXS	TXS	RXRD

Name	Bit	Type	Reset value	
Reserved	31:12	N/A	0x00	保留，保持复位值
CM	11	RO	0x0	Character Match Interrupt Disable 0: 没 有 中 断 标 志 位 1: 产生中断标志位
EOB	10	RO	0x0	End of Block Interrupt Disable 0: 没 有 中 断 标 志 位 1: 产生中断标志位
TC	9	RO	0x0	Transmission Complete Interrupt Disable 0: 没 有 中 断 标 志 位 1: 产生中断标志位
LINBK	8	RO	0x0	LIN Break Detection Interrupt Disable 0: 没 有 中 断 标 志 位 1: 产生中断标志位
ABTO	7	RO	0x0	Auto Baud Rate Timeout Interrupt Disable 0: 没 有 中 断 标 志 位 1: 产生中断标志位
ABE	6	RO	0x0	Auto Baud Rate End Interrupt Disable 0: 没 有 中 断 标 志 位 1: 产生中断标志位

BUSY	5	RO	0x0	Busy Detect Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
RTO	4	RO	0x0	Character Timeout Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
MDS	3	RO	0x0	Modem Status Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
RXS	2	RO	0x0	Reveiver Line Status Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
TXS	1	RO	0x0	THR Empty Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
RXRD	0	RO	0x0	Reveive Data Available Interrupt Flag 0: 没有中断标志位 1: 产生中断标志位

1. 4. 14 屏蔽中断标志寄存器(UART_IFM)

		Address offset:		0x34											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				CM	EOB	TC	LINBK	ABTO	ABE	BUSY	RTO	MDS	RXS	TXS	RXRD

Name	Bit	Type	Reset value	
Reserved	31:12	N/A	0x00	保留, 保持复位值
CM	11	RO	0x0	Character Match Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
EOB	10	RO	0x0	End of Block Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
TC	9	RO	0x0	Transmission Complete Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
LINBK	8	RO	0x0	LIN Break Detection Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
ABTO	7	RO	0x0	Auto Baud Rate Timeout Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
ABE	6	RO	0x0	Auto Baud Rate End Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
BUSY	5	RO	0x0	Busy Detect Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位

RTO	4	RO	0x0	Character Timeout Interrupt Disable 0: 无屏蔽中断 1: 屏蔽中断
MDS	3	RO	0x0	Modem Status Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
RXS	2	RO	0x0	Reveiver Line Status Interrupt Disable 0: 无屏蔽中断 1: 屏蔽中断
TXS	1	RO	0x0	THR Empty Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位
RXRD	0	RO	0x0	Reveive Data Available Interrupt Disable 0: 没有中断标志位 1: 产生中断标志位

1. 4. 15 中断清除寄存器(UART_ICR)

		Address offset:		0x38											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				CM	EOB	TC	LINBK	ABTO	ABE	BUSY	RTO	MDS	RXS	TXS	RXRD

Name	Bit	Type	Reset value	
Reserved	31:12	N/A	0x00	保留，保持复位值
CM	11	RO	0x0	Character Match Interrupt Disable 0: 无 1: 清除中断标志位
EOB	10	RO	0x0	End of Block Interrupt Disable 0: 无 1: 清除中断标志位
TC	9	RO	0x0	Transmission Complete Interrupt Disable 0: 无 1: 清除中断标志位
LINBK	8	RO	0x0	LIN Break Detection Interrupt Disable 0: 无 1: 清除中断标志位
ABTO	7	RO	0x0	Auto Baud Rate Timeout Interrupt Disable 0: 无 1: 清除中断标志位
ABE	6	RO	0x0	Auto Baud Rate End Interrupt Disable 0: 无 1: 清除中断标志位
BUSY	5	RO	0x0	Busy Detect Interrupt Disable 0: 无 1: 清除中断标志位

RTO	4	RO	0x0	Character Timeout Interrupt Disable 0: 无 1: 清除中断标志位	效
MDS	3	RO	0x0	Modem Status Interrupt Disable 0: 无 1: 清除中断标志位	效
RXS	2	RO	0x0	Reveiver Line Status Interrupt Disable 0: 无 1: 清除中断标志位	效
TXS	1	RO	0x0	THR Empty Interrupt Disable 0: 无 1: 清除中断标志位	效
RXRD	0	RO	0x0	Reveive Data Available Interrupt Disable 0: 无 1: 清除中断标志位	效

Confidential

第22章 音频接口（PDM）

1.1 PDM 简介

PDM（Pulse Density Modulation）是一种用数字信号表示模拟信号的调制方法。同为将模拟量转换为数字量的方法，PCM 使用等间隔采样方法，将每次采样的模拟分量幅度表示为 N 位的数字分量（ $N = \text{量化深度}$ ），因此 PCM 方式每次采样的结果都是 $N \text{ bit}$ 字长的数据。PDM 则使用远高于 PCM 采样率的时钟采样调制模拟分量，只有 1 位输出，要么为 0，要么为 1。因此通过 PDM 方式表示的数字音频也被称为 Oversampled 1-bit Audio。相比 PDM 一连串的 0 和 1，PCM 的量化结果更为直观简单。

1.2 PDM 功能描述

在以 PDM 方式作为模数转换方法的应用接收端，需要用到抽取滤波器（Decimation Filter）将密密麻麻的 0 和 1 代表的密度分量转换为幅值分量，而 PCM 方式得到的就已经是幅值相关的数字分量。图 20 示意为通过 PDM 方式数字化的正弦波。

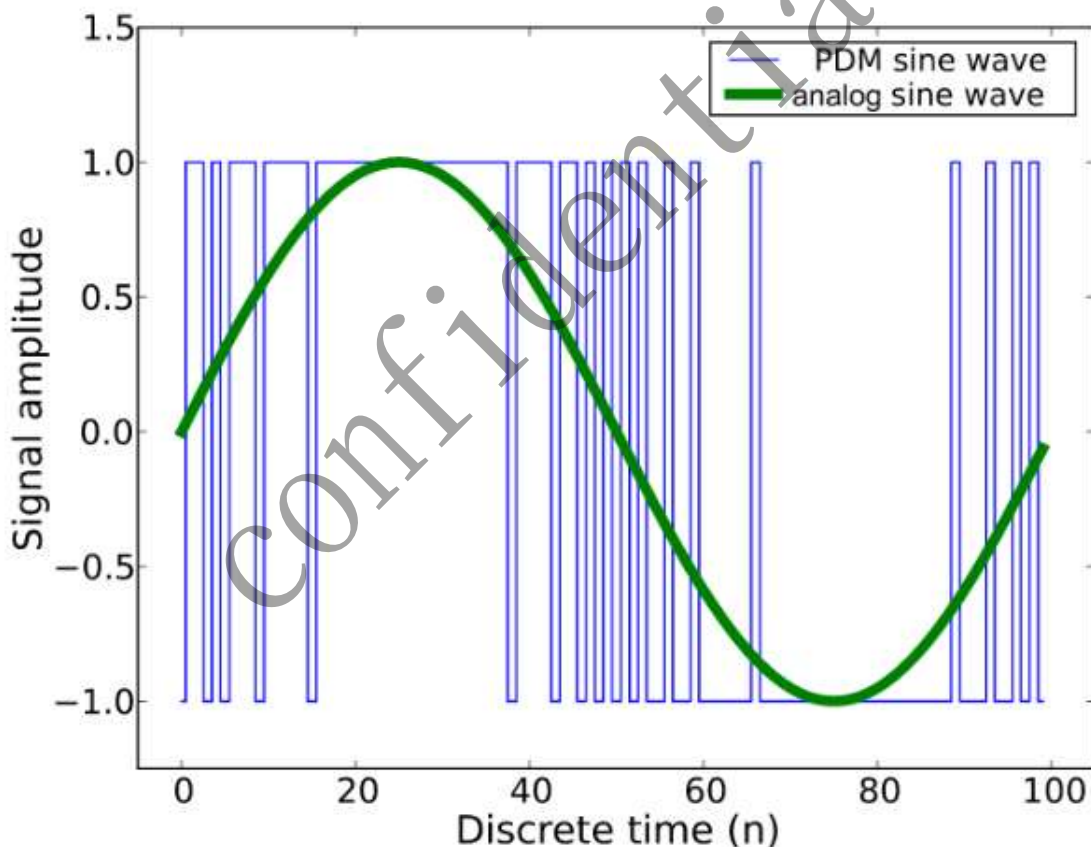


图 22-1 PDM 方式表示的正弦波

PCM 方式的逻辑更加简单，但需要用到数据时钟，采样时钟和数据信号三根信号线；PDM 方式的逻辑相对复杂，但它只需要两根信号线，即时钟和数据。PDM 在诸如手机和平板等对于空间限制严格的场合有着广泛的应用前景。在数字麦克风领域，应用最广的就是 PDM 接口，其次为 I2S 接口。PDM 格式的音频信号可以在比如 LCD 屏这样 Noise 干扰强的电路附近走线（等于没说，这里指数字信号抗干扰能力相比于模拟信号更强，同样 PCM 也具有此优势）。

通过 PDM 接口方式，传输双声道数据只要用到两根信号线。如图 21 示意两个 PDM 接口的发送设备与同一个接收设备的连接情况，比如 **Source 1/2** 分别作为左右声道的麦克风，通过这种方式可以将采集到的双声道数据传送到接收设备。主设备（此例中作为接收设备）为两个从设备提供时钟，分别在时钟的上升沿和下降沿触发选择 **Source 1/2** 作为数据输入。图 22 为 Maxim 的 Class-D 类型功放 **MAX98358** 对 PDM 接口时序的要求，可以看到它在 **PDM_CLK** 的上升沿采样左声道数据，在 **PDM_CLK** 下降沿采样右声道数据。

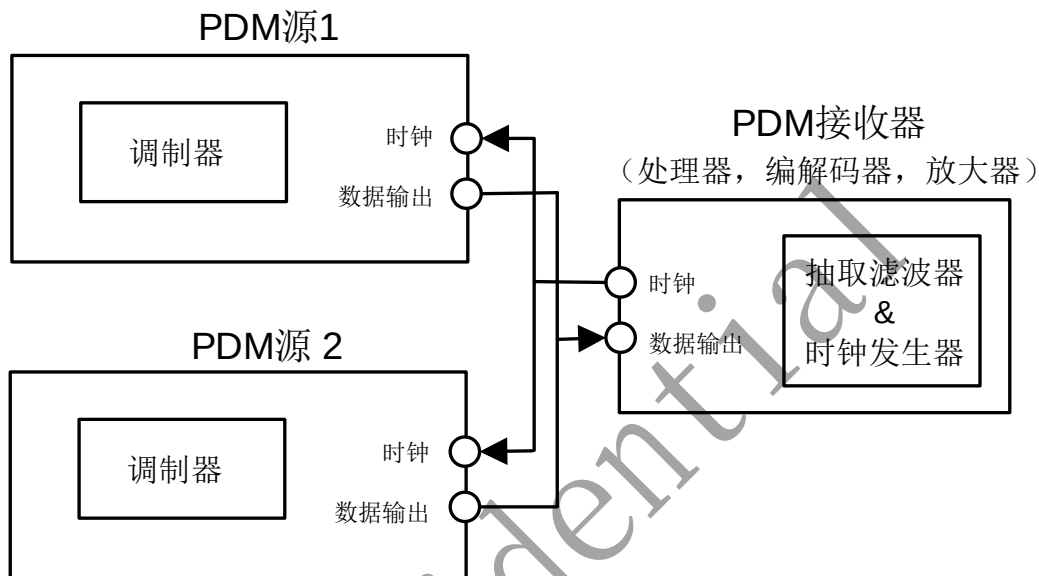


图 22-2 PDM 连接示意图 (2 发送设备 + 1 接收设备)

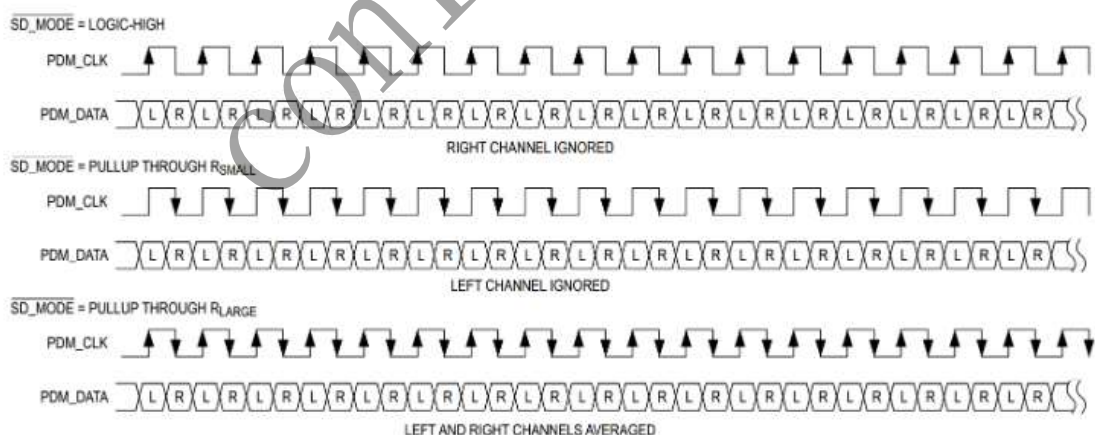


图 22-3 PDM 时序框图

基于 PDM 的架构不同于 I2S 和 TDM 之处在于，抽取滤波器（Decimation Filter）不在发送设备，而在接收设备内部。源端输出是原始的高采样率（oversample）调制数据，如 Sigma-Delta 调制器的输出，而不是像 I2S 中那样的抽取数据（An I2S output digital microphone includes the decimation filter, so its output is already at a standard audio sample rate that's

easy to interface to and process.)。基于 PDM 接口的应用降低了发送设备的复杂性，由于作为接收设备的 CODEC 内部集成抽取滤波器，因此系统整体复杂度大大降低。对于数字麦克风而言，通过使用面向 CODEC 或处理器制造的更精细硅工艺，而非传统麦克风使用的工艺，可以实现更高效率的抽取滤波器。

1.3 PDM 寄存器描述

1.3.1 PDM 控制寄存器(PDM_CR)

		Address offset:		0x00											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												PDM_CHN	PDM_DMA_EN	PDM_EN	

Name	Bit	Type	Reset value	Description
Reserved	31:4	N/A	0x00	保留，保持复位值
PDM_CHN	3	R/W	0x0	PDM 通道设置
PDM_DMA_EN	2:1	R/W	0x0	PDM DMA 通道使能
PDM_EN	0	R/W	0x0	PDM 使能

1.3.2 PDM 配置寄存器(PDM_CFG)

		Address offset:		0x04											
		Reset value:		0x033F 0A7C											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				DATA_GAIN				SAMPLE_RATE							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP_DELAY								PDMCLK_RATIO							

Name	Bit	Type	Reset value	Description
Reserved	31:28	N/A	0x00	保留，保持复位值
DATA_GAIN	27:24	R/W	0x3	数据增益
SAMPLE_RATE	23:16	R/W	0x3F	采样率设置
CAP_DELAY	15:8	R/W	0x0A	捕捉延时设置
PDMCLK_RATIO	7:0	R/W	0x7C	PDM 时钟速率设置

1. 3. 3 PCM 数据寄存器 0(PCM_DATA0)

		Address offset:			0x08										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCM_DATA0															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
PCM_DATA0	15:0	RO	0x0	PCM 数据 0

1. 3. 4 PCM 数据寄存器 1(PCM_DATA1)

		Address offset:			0x0C										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PCM_DATA1															

Name	Bit	Type	Reset value	Description
Reserved	31:16	N/A	0x00	保留，保持复位值
PCM_DATA1	15:0	RO	0x0	PCM 数据 1

1. 3. 5 PDM 中断使能寄存器(PDM_IER)

		Address offset:			0x10										
		Reset value:			0x0000 0000										
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															IER

Name	Bit	Type	Reset value	Description

Reserved	31:1	N/A	0x00	保留，保持复位值
IER	0	WO	0x0	中断使能 0: 无效 1: 使能

1.3.6 PDM 中断禁止寄存器(PDM_IDR)

		Address offset:		0x14											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															IDR

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值
IDR	0	WO	0x0	中断禁止 0: 无效 1: 禁止

1.3.7 PDM 中断有效寄存器(PDM_IVS)

		Address offset:		0x18											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															IVS

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值
IVS	0	RO	0x0	中断状态 0: 中断禁止 1: 中断使能

1.3.8 PDM 原始中断标志寄存器(PDM_RIF)

		Address offset:		0x1C											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

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

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值 原始中断标志位
RIF	0	RO	0x0	0: 没有中断标志位 1: 产生中断标志位

1. 3. 9 PDM 屏蔽中断寄存器(PDM_IFM)

		Address offset:		0x20											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															IFM

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值 屏蔽中断标志位
IFM	0	RO	0x0	0: 没有中断标志位 1: 产生中断标志位

1. 3. 10 PDM 中断清除寄存器(PDM_ICR)

		Address offset:		0x24											
		Reset value:		0x0000 0000											
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															ICR

Name	Bit	Type	Reset value	Description
Reserved	31:1	N/A	0x00	保留，保持复位值 中断清除
ICR	0	WO	0x0	0: 无效 1: 清除中断标志

1. 3. 11 PDM 滤波器参数配置(PDM_COEFx)

		Address offset:				0x40~0x7F									
		Reset value:				0x0000 0000									
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
COEF_DATAx<31:16>															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COEF_DATAx<15:0>															

Name	Bit	Type	Reset value	Description
COEF_DATAx	31:0	WO	0x0	PDM 参数

confidential

第23章 触摸按键（TK）

1.1 TK 简介

LE5030 集成了高性能的电容式感应触摸模块，按键可以穿透绝缘材料外壳 8mm（玻璃、塑料等等）以上，准确无误地侦测到手指的有效触摸。并保证了产品的灵敏度、稳定性、可靠性等不会因环境条件的改变或长期使用而发生变化，并具有防水和强抗干扰能力，超强防护，超强适应温度范围。

1.2 TK 主要特性

- ◆ 电容按键支持
 - 最多支持 14 个感应通道
 - 参数自动调校
- ◆ 性能指标
 - 动态模式下典型功耗：2.3mA
 - 监控模式下典型功耗：5uA@250ms
 - 触摸电容检测范围：5pF~50pF

1.3 TK 功能描述

（1）TK 电容通道

TK 模块具有 14 个电容检测通道，每个检测通道都可以作为电容按键或者滑条中的一个通道使用，即用户设置参数、回读参数和设置完参数后的扫描结果。

用户可以将任一电容检测通道设置为按钮，或设置为滑条组中的一个通道，同时每一电容检测通道可以被独立的打开或关闭，通道的开关由寄存器 TK_CHELR 或 TK_CHEHR 控制。芯片上电后所有检测通道默认开启，建议用户将未使用的检测通道设置为禁用，没有被用到的电容检测通道建议在 PCB 上应当接地。

（2）滑条功能

用户可以选择一些检测通道并把它们定义为一个滑条组，滑条最多由 12 个检测通道组成。滑条的类型由寄存器 TK_STR 决定，它可以是线性或圆形的。寄存器 TK_SLR 决定滑条总共由多少个检测通道组成。寄存器组 TK_SSRX 保存了滑条组中检测通道的顺序，用户需将组成滑条的检测通道依照他们在滑条中的顺序依次写入 00h-0Bh 地址中。滑条组的位置幅值为 1 到 200，0 表示滑条没有被触发。当手指放到 TK_SSRX 中保存的第一个通道时识别的位置为 1，当手指放到 TK_SSRX 中保存的最后一个通道时识别的位置为 200。

（3）中断功能

当任一电容检测通道（不论此时是滑条还是按钮）检测到有触摸或触摸离开时，TK 模块将会产生 IRQ 中断，500ms 内如果有按键状态变化将不响应中断，用户可以手动清除中断，以达到更快的响应速度。如果用户软件轮询来查询按键状态，则可以忽略这个中断。

（4）按键状态读取

主控制器可以读出电容按钮有无触摸的信息(它储存在寄存器 TK_CNTLRL 和 TK_CNTHRL 中，此开关信息只针对被设置为按钮的检测通道)，和手指在滑条上滑动的位置信息(储存在寄存器 TK_SPR 中，此位置信息只针对被设置为滑条的检测通道)。

当寄存器 ScanCR 的 ScanEN 位被写 1 时，则开始扫描所有被使能的电容按钮。

(5) 低功耗扫描模式

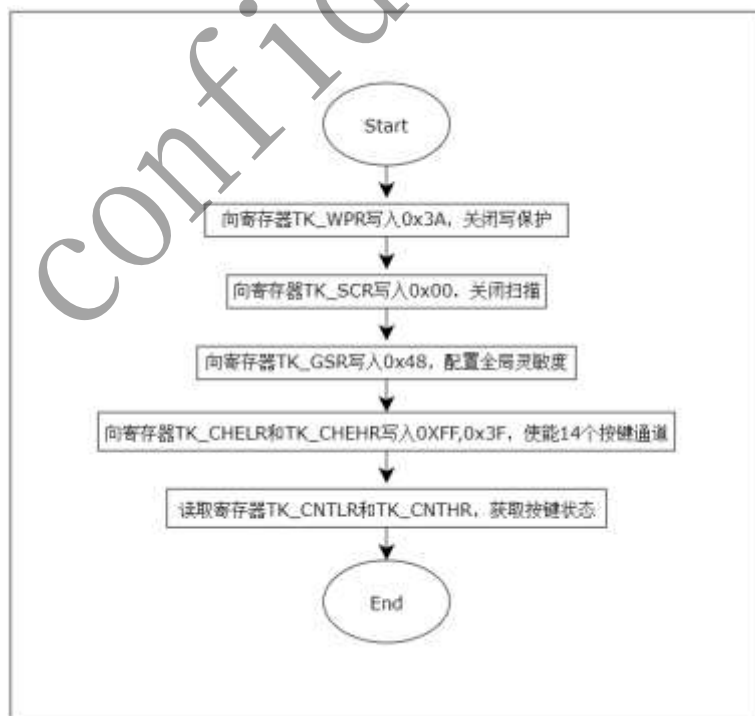
低功耗模式，为了得到最低的功耗，用户可以往 ScanCR 寄存器 LP 位中写 1 使能低功耗扫描模式。在一定时间内没有检测到触摸时(通过 TK_LDR 寄存器配置，默认 4s)，TK 模块会自动进入低功耗扫描模式以降低功耗，注意 TK 模块在进入低功耗扫描模式后寄存器无法读写，当有触摸信号(IRQ)产生后，将退出低功耗模式，进入到正常扫描模式，如果用户主控是使用软件轮询方式查询扫描结果则不建议设置 LP 位。

(6) 睡眠模式

睡眠模式，为了得到最低的功耗，用户可以往 ScanCR 寄存器 PD 位中写 1 使按键进入深睡眠模式。在深睡眠模式下所有活动的功能都被禁止。如果用户需要唤醒 TK 模块需要重新初始化。注意在唤醒后需要重新使能扫描并配置扫描模式和参数。深睡眠比低功耗扫描时的功耗更低。

1.4 TK 用户参数调试与设置流程

用户可以将所有设置参数存储于 ROM 中，在上电后将参数传到 TK 模块的寄存器里。当芯片上电后 TK 模块默认使能，在 15 秒内必须完成参数设置，否则 TK 模块会自动进入深睡眠状态，此时电容按键将无法操作。下面是一个简单的例子：配置全局灵敏度，使能扫描并选用标准模式，然后回读按键状态。

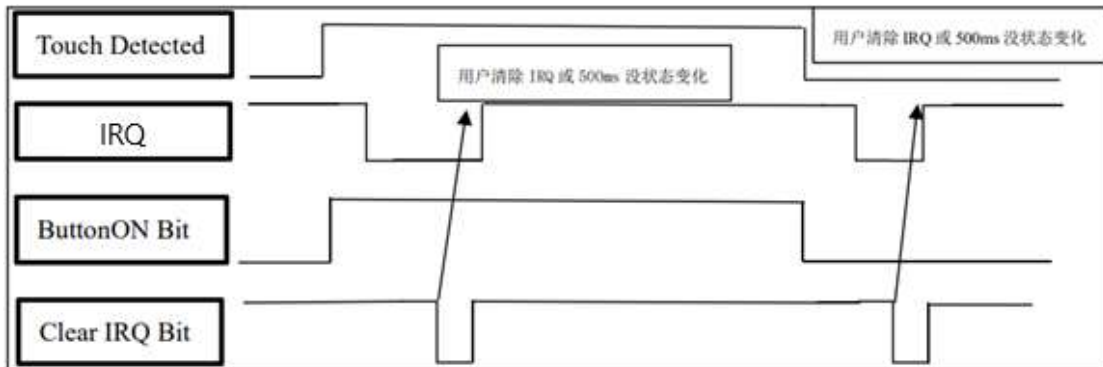


1.5 TK 中断操作波形示意图

当检测到一个感应通道的状态和前一状态不同时，例如触摸按下和离开时，TK 模块将会

拉低 IRQ，500ms 内没有按键状态变化（按下和抬起）则会自动拉高。用户也可以检测到中断后，写 0 到 ScanCR 寄存器的 IRQ 位，用来清除这个中断。用户也可以忽略这一中断而采用主控软件轮询方式查询感应通道的检测状态。下面是一个清除中断的例子：

向寄存器 TK_SCR 写入命令 0x01，以清除 IRQ 同时继续扫描



1.6 工作模式

(1) 监控模式

当接收到进入低功耗模式指令时，芯片切换到监控模式；在此模式下，TK 模块以较低频率，通过自容扫描检测可能到来的触摸动作，并迅速切换到动态模式。

(2) 动态模式

当频繁有触摸操作时，处于此模式；在此模式下，芯片快速对按键进行自电容扫描，以及及时检测。

1.7 TK 寄存器描述

1.7.1 TK 滑条通道号寄存器(TK_SSRX)

Address offset:		0x00~0x0B						
Reset value:		0x00						
7	6	5	4	3	2	1	0	
SlideSeq[x]								

Name	Bit	Type	Reset value	Description
SlideSeq[x]	7:0	N/A	0x00	滑条中第 x 个检测通道的通道号，例如 0A 表示检测通道 S10R

1.7.2 TK 滑条通道数目寄存器(TK_SLR)

Address offset:		0x10						
Reset value:		0x03						
7	6	5	4	3	2	1	0	
SlideSize								

Name	Bit	Type	Reset value	Description
SlideSize	7:0	N/A	0x03	组成滑条的通道数目，最多 12 个，最少三个

1.7.3 TK 滑条类型寄存器(TK_STR)

Address offset:		0x11					
Reset value:		0x00					
7	6	5	4	3	2	1	0
Reserved							SlideType

Name	Bit	Type	Reset value	Description
Reserved	7:1	N/A	0x0	保留，保持复位值
SlideSize	0	R/W	0	滑条的类型：0：线性滑条 1：圆型滑条

1.7.4 TK 配置寄存器(TK_SCFR)

Address offset:		0x12					
Reset value:		0x10					
7	6	5	4	3	2	1	0
Reserved			MultiOn	DNPR	NPRR[2:0]		

Name	Bit	Type	Reset value	Description
Reserved	7:5	N/A	0x00	必须为 0
MultiOn	4	R/W	1	使能多个感应通道同时输出触摸结果。 1：使能；0：禁用，缺省为 1
DNPR	3	R/W	0	禁用噪声保护，如禁用噪声保护，则任何超出触摸信号阈值的检测结果都会被采用。 1：禁止 0：使能缺省为 0
NPRR	2:0	R/W	0x0	噪声门限比例倍数，如果检测信号超出触摸信号阈值的 NPRR 倍，则忽略此次检测结果。0：4 倍阈值 1：6 倍阈值... 7：18 倍阈值缺省为 0

1.7.5 TK 防抖参数寄存器(TK_DBR)

Address offset:		0x13					
Reset value:		0x03					
7	6	5	4	3	2	1	0
Debounce							

Name	Bit	Type	Reset value	Description
Debounce	7:0	R/W	0x3	防抖动参数，缺省值为 3，提高按键按下后的稳定性

1.7.6 TK 通道使能寄存器 1(TK_CHEL1R)

Address offset:		0x14					
Reset value:		0xFF					
7	6	5	4	3	2	1	0
ChannelEN[7:0]							

Name	Bit	Type	Reset value	Description
ChannelEN	7:0	R/W	0xFF	ChannelEN[7:0]通道使能控制，每个 bit 位对应每个通道号，缺省为 1；建议关闭不使用的通道。 1：使能电容检测通道。0：禁用。

1.7.7 TK 通道使能寄存器 2(TK_CHEHR)

		Address offset:	0x15						
		Reset value:	0x3F						
7		6		5	4	3	2	1	0
Reserved			ChannelEN[13:8]						

Name	Bit	Type	Reset value	Description
Reserved	7:6	N/A	0x0	保留，保持复位值
ChannelEN	5:0	R/W	0x3F	ChannelEN[13:8]使能控制，缺省为 1，同 ChannelEN[7:0]

1.7.8 TK 全局灵敏度寄存器(TK_GSR)

		Address offset:	0x1E					
		Reset value:	0x00					
7	6	5	4	3	2	1	0	
GbSen[7:0]								

Name	Bit	Type	Reset value	Description
GbSen	7:0	R/W	0x00	全局灵敏度设置，共 256 级，0 最高，缺省为 0； 过高或过低的灵敏度都可能导致触摸检测失败。灵敏度过低时用户触摸无响应或需大力增大触摸面积才有响应，灵敏度过高时用户轻触或接近时有响应，但正常触摸时反而没响应，这是因为误触发了噪声保护功能。 触摸灵敏度的更改必须先关闭扫描后再设置，新的灵敏度数据将在重新使能扫描 100ms 后生效。

1.7.9 TK 控制寄存器(TK_SCR)

		Address offset:	0x1F												
		Reset value:	0x00												
7		6		5		4		3		2		1		0	
PD		LP		Reserved								IRQ		ScanEN	

Name	Bit	Type	Reset value	Description
PD	7	R/W	0	写 1 进入睡眠模式，进入睡眠模式之前必须禁用芯片扫描，即置 ScanEN 为 0
LP	6	R/W	0	芯片是否自动进入低功耗扫描模式控制，1：一段时间没有触摸自动进入低功耗扫描 0：禁用；缺省为 0

Reserved	5:2	N/A	0x00	保留，保持复位值
IRQ	1	N/A	0	写 0 将清除 IRQ 标志
ScanEN	0	N/A	0	芯片扫描使能。1：使能电容检测扫描；0：禁用

1. 7. 10 TK 滑条位置寄存器(TK_SPR)

		Address offset:	0x21					
		Reset value:	0x00					
7	6	5	4	3	2	1	0	
SlidePos[7:0]								

Name	Bit	Type	Reset value	Description
SlidePos	7:0	R	0x00	滑条的触摸位置，1-200，0 表示没有滑条触摸

1. 7. 11 TK 通道检测寄存器 1(TK_CNTLR)

		Address offset:	0x22					
		Reset value:	0x00					
7	6	5	4	3	2	1	0	
ButtonON [7:0]								

Name	Bit	Type	Reset value	Description
ButtonON	7:0	R	0x00	Channel[7:0]的检测状态，如检测通道被用于滑条则该检测 通道不会产生 ButtonON 触摸信息，每位 BIT 表示各通道的状态

1. 7. 12 TK 通道检测寄存器 2(TK_CNTHR)

		Address offset:	0x23					
		Reset value:	0x00					
7	6	5	4	3	2	1	0	
Reserved		ButtonON [13:8]						

Name	Bit	Type	Reset value	Description
Reserved	7:6	N/A	0x00	保留，保持复位值
ButtonON	5:0	R	0x00	Channel[13:8]的检测状态，如检测通道被用于滑条则该检测 通道不会产生 ButtonON 触摸信息

1. 7. 13 TK 校验和寄存器(TK_CSR)

		Address offset:	0x25					
		Reset value:	0x00					
7	6	5	4	3	2	1	0	
CHKSUM[7:0]								

Name	Bit	Type	Reset value	Description
CHKSUM	7:0	R	0x00	ButtonON 的校验和。 如果 Reg@21H+Reg@22H+Reg@23H+Reg@25H+0xC5=0x00 则校验和正确

1. 7. 14 TK 补偿电流寄存器(TK_ICGSR)

		Address offset:	0x29												
		Reset value:	0x28												
7		6		5		4		3		2		1		0	
icom_GS [7:0]															

Name	Bit	Type	Reset value	Description
icom_GS	7:0	R/W	0x28	取值范围为 0~127，默认为 40。该值配置越大，群扫描信号越大，越容易低功耗唤醒；但要避免因为过大而无法进入低功耗

1. 7. 15 TKCDC 寄存器(TK_CFDR)

		Address offset:	0x2A					
		Reset value:	0x07					
7	6	5	4	3	2	1	0	
cdc_fsw_div[7:0]								

Name	Bit	Type	Reset value	Description
cdc_fsw_div	7:0	R/W	0x07	定义 CDC 扫描频率 默认为 0x07。

1. 7. 16 TK 低功耗扫描寄存器(TK_LPSPR)

		Address offset:	0x2B					
		Reset value:	0x00					
7	6	5	4	3	2	1	0	
low_power_scan_period [7:0]								

Name	Bit	Type	Reset value	Description
LPSP	7:0	R/W	0x00	定义软件低功耗下的扫描周期，三种模式可选 2: 100ms 1: 250ms 0: 500ms 缺省为 0，即 500ms 扫描一次

1. 7. 17 TK 消拍寄存器(TK_KDCR)

		Address offset:	0x2C					
		Reset value:	0x01					
7	6	5	4	3	2	1	0	

key_debounce_count [7:0]

Name	Bit	Type	Reset value	Description
KDC	7:0	R/W	0x01	定义按键按下和抬起时的消抖拍数，缺省为 1，即连续两拍按下才会识别为按键按下，连续两拍没有触摸才会识别为按键抬起，参数范围：1-3。

1. 7. 18 TK LD 寄存器(TK_LDR)

		Address offset:	0x30					
		Reset value:	0x04					
7	6	5	4	3	2	1	0	
lp_delay [7:0]								

Name	Bit	Type	Reset value	Description
lp_delay	7:0	R/W	0x04	定义 lp_delay*1 秒没有按键状态变化后进入低功耗扫描模式，默认为 4，即 4 秒

1. 7. 19 TK 扫描周期寄存器(TK_SCTR)

		Address offset:	0x31					
		Reset value:	0x01					
7	6	5	4	3	2	1	0	
scan_period [7:0]								

Name	Bit	Type	Reset value	Description
scan_period	7:0	R/W	0x01	定义扫描周期，scan_period*16ms 更新一次按键或滑条状态，默认为 1，即 16ms 更新一次，参数设置范围：1-3

1. 7. 20 TK 唤醒阈值寄存器(TK_WTR)

		Address offset:	0x32					
		Reset value:	0xFF					
7	6	5	4	3	2	1	0	
wake_th [7:0]								

Name	Bit	Type	Reset value	Description
wake_th	7:0	R/W	0xFF	唤醒阈值设置，缺省值为 0xff，越小越不灵敏 芯片内部有两种唤醒机制： 1. 超低功耗唤醒，当 wake_th 不为 0xff 时进入此模式，此时功耗在 10uA 以下，阈值范围 从 30 到 60，值越小越灵敏。（即，硬件低功耗） 2. 硬件轮询唤醒，wake_th 为 0xff 时进入此模式，即软件低功耗。 当 LPSP=0 时，10min 平均功耗约 2.7uA。 当 LPSP=1 时，10min 平均功耗约 5uA。 注：具体功耗大小与硬件设计、参数配置等因素相关。

1. 7. 21 TK 通道信号寄存器 1(TK_SIGLRx)

Address offset:		0x40					
Reset value:		0x00					
7	6	5	4	3	2	1	0
Signal[x][7:0]							

Name	Bit	Type	Reset value	Description
Signal[x]	7:0	R	0x00	第 x 个电容通道信号的高 8 位，CSK14S 的则为低 8 位

1. 7. 22 TK 通道信号寄存器 2(TK_SIGHRx)

Address offset:		0x41~0x5B					
Reset value:		0x00					
7	6	5	4	3	2	1	0
Signal[x][7:0]							

Name	Bit	Type	Reset value	Description
Signal[x]	7:0	R	0x00	第 x 个电容通道信号的的低 8 位，CSK14S 的则为高 8 位

1. 7. 23 TK 写保护寄存器(TK_WPR)

Address offset:		0xAF					
Reset value:		0x00					
7	6	5	4	3	2	1	0
Write Protection[7:0]							

Name	Bit	Type	Reset value	Description
WPR	7:0	R/W	0x00	写入 3A 将会放开写保护，允许对寄存器进行写操作。如任何两次写操作间隔超过 0.15S，写保护将会自动生效

1. 7. 24 TK 使能通道寄存器 0(TK_GMR0)

Address offset:		0XB0					
Reset value:		0x00					
7	6	5	4	3	2	1	0
GsMask[0]							

Name	Bit	Type	Reset value	Description
GsMask[0]	7:0	R/W	0x00	指定通道 7——通道 0 是否参与低功耗扫描，1: 参与 0: 不参与，每位 BIT 对应控制不同通道，默认都不参与，建议把不使用的通道关闭

1. 7. 25 TK 使能通道寄存器 1(TK_GMR1)

Address offset:		0XB1					
Reset value:		0x00					
7	6	5	4	3	2	1	0
Reserved		GsMask[1]					

Name	Bit	Type	Reset value	Description
Reserved	7:6	N/A	0x00	保留，保持复位值
GsMask[1]	5:0	R/W	0x00	指定通道 13-通道 8 是否参与低功耗扫描，1：参与 0 不参与

1. 7. 26 TK 单灵敏度寄存器(TK_LSRx)

Address offset:		0XE0~0xED					
Reset value:		0x00					
7	6	5	4	3	2	1	0
Lsense [x]							

Name	Bit	Type	Reset value	Description
Lsense [x]	7:0	R/W	0x00	检测通道灵敏度设置，1-255,1 最高，缺省为 0，灵敏度设置触摸灵敏度的更改必须先关闭扫描后再设置，新的灵敏度数据将在重新使能扫描后生效，其余检测通道的灵敏度设置依次存储于 E1H-EDH 中，EEH 为软件低功耗灵敏度寄存器。

1. 7. 27 TK 群灵敏度寄存器(TK_LSGR)

Address offset:		0XEE					
Reset value:		0x00					
7	6	5	4	3	2	1	0
LSense_GS							

Name	Bit	Type	Reset value	Description
LSense_GS	7:0	R/W	0x00	群灵敏度设置，1-255,1 最高，默认为 0（不配置），直接使用全局灵敏度寄存器 GbSen 所配置值；当配置为非 0 时，则使用所配置值。具体含义参照 GbSen 含义对照表。通过配置该寄存器，可优化低功耗唤醒。

附录1 电气特性

Parameter	Description	Min.	Typ.	Max.	Unit
Operating Temperature		-40		85/105	°C
Supply Voltage	Regulator supply input	1.8	3.3	3.6	V
t_{R_SUP}	Power supply rise time (0V to 3.3V)			330	ms
Current Consumption	TX Mode @ 0dBm Output (DCDC ON)		4.3		mA
	RX Mode (DCDC ON)		4.5		mA
	Normal Mode		2		mA
	Low Power Mode2		1.1		uA
	Low Power Mode3		0.7		uA
TK Module Current	Normal Mode		2.3		mA
	Low Power Scan @500ms		2.7		uA
	Low Power Scan @250ms		5		uA
	Low Power Scan @100ms		12.5		uA
V_{IH}	High Level Input Voltage	0.8*VDD33		VDD33	V
V_{IL}	Low Level Input Voltage	0		0.2*VDD33	V
V_{OH}	High Level Output Voltage	VDD33-0.4		VDD33	V
V_{OL}	Low Level Output Voltage			0.4	V
Source Current			30		mA
Sink Current			60		mA
V_{ESD}	Electrostatic discharge voltage		± 8000		V

附录1.1 RX spec

Symbol	Description	Min	Typ	Max	Unit
Receiver operation					
PRX_{MAX}	Maximum received signal strength at < 0.1% BER.		-1.5		dBm
$PRX_{SENS,2M}$	Sensitivity (0.1% BER) at 2 Mbps	-97	-96		dBm
$PRX_{SENS,1M}$	Sensitivity (0.1% BER) at 1 Mbps.	-100.7	-99.7		dBm
$PRX_{SENS,500k}$	Sensitivity (0.1% BER) at 500 kbps.	-99	-98		dBm
$PRX_{SENS,125k}$	Sensitivity (0.1% BER) at 125 kbps.	-105	-104		dBm
$P_{SENS\ IT}$ 1 Mbps BLE	Receiver sensitivity: Ideal transmitter.	-100	-99		dBm
$P_{SENS\ DT}$ 1 Mbps BLE	Receiver sensitivity: Dirty transmitter.	-99.5	-98.5		dBm
RX selectivity					
C/I_{CO}	C/I co-channel.		-2		dB
C/I_{1ST}	1st ACS, C/I 1 MHz.		-17		dB
C/I_{2ND}	2nd ACS, C/I 2 MHz.		-41		dB
C/I_{3+N}	ACS, C/I (3+n) MHz offset [n = 0, 1, 2, ...].		-50		dB
C/I_{Image}	Image blocking level.		-42.5		dB
$C/I_{Image\pm 1MHz}$	Adjacent channel to image blocking level (± 1 MHz).		-45		dB

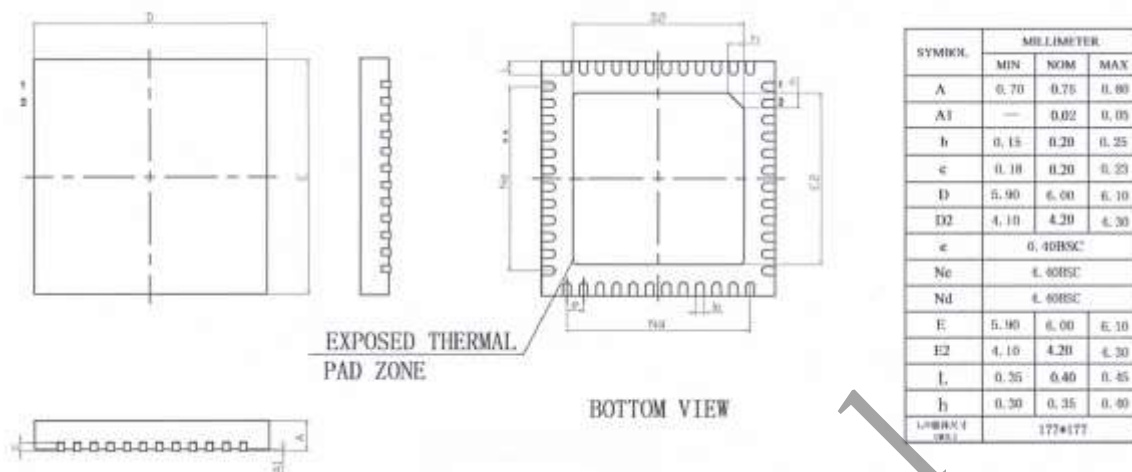
附录1.2 TX spec

Symbol	Description	Min	Typ	Max	Unit
Low Power PA					
P_{RF_PA}	Maximum output power.		8		dBm
P_{RFC_PA}	RF power control range.		40		dB
P_{BW1_PA}	20 dB bandwidth for modulated carrier (1 Mbps).		1000		KHz
High Power PA					
P_{RF_PAHP}	Maximum output power.		12		dBm
P_{RFC_PAHP}	RF power control range.		12		dB
P_{BW1_PAHP}	20 dB bandwidth for modulated carrier (1 Mbps).		1000		KHz

confidential

附录2 封装信息

附录2.1 QFN48L(0606X0.75-0.40)



附录3 订货代码

型号	协议	包装	最小包数量	封装
LE5030BI	BLE5.0	卷带	2500	QFN48

confidential

附录4 产品命令规则

