

PT32L031 系列
用户手册

目录

1 概述	11
1.1 ARM M0 内核	11
1.2 低功耗模式	11
1.3 存储器	11
1.4 时钟	11
1.5 高精度 ADC	11
1.6 比较器	12
1.7 蜂鸣器	12
1.8 串口	12
1.9 定时器 (TIMER)	12
1.10 安全机制	13
1.11 GPIO	13
1.12 调试接口	13
2 封装信息	14
2.1 LQFP32/QFN32(4*4)/QFN32(5*5)	14
2.2 QFN24(4*4)	14
2.3 TSSOP28	15
2.4 TSSOP20	15
3 管脚定义	16
4 框图	17
5 地址空间分配	18
5.1 AHB 地址分配	18
5.2 APB 地址映射	18
5.3 FLASH NVR 配置	19
5.4 UID	20
5.5 系统控制寄存器列表	21
5.6 系统控制寄存器描述	22
5.6.1 Boot 模式寄存器 REMAP	22
5.6.2 复位信息寄存器 RSTINFO	22
5.6.3 重启控制寄存器 RETRIMING_CTRL	23
5.6.4 复位使能控制寄存器 RESET_CTRL	24
5.6.5 CUSTOMER_ID 信息寄存器 CUSTOMER_ID	25
5.6.6 FLASH 保护状态寄存器 FLASH_PROTECT	25
5.6.7 UID 信息[31:0] 寄存器 UID_INFO1	26
5.6.8 UID 信息[63:32] 寄存器 UID_INFO2	26
5.6.9 UID 信息[95:64] 寄存器 UID_INFO3	26
5.6.10 系统控制寄存器 SCR	27
6 矢量中断控制器	28

6.1 矢量中断控制寄存器列表	29
6.2 矢量中断控制寄存器描述	29
6.2.1 系统中断使能寄存器 <i>ISER</i>	29
6.2.2 系统中断禁止寄存器 <i>ICER</i>	33
6.2.3 系统中断挂起设定寄存器 <i>ISPR</i>	37
6.2.4 系统中断挂起清除寄存器 <i>ICPR</i>	42
7 时钟系统	47
7.1 高速晶体振荡时钟	47
7.2 内部 24MHz RC 振荡时钟	48
7.3 内部 32kHz RC 振荡时钟	48
7.4 外部输入高频时钟	48
7.5 时钟切换的安全性	48
7.6 高速时钟监测	48
7.7 寄存器说明	49
7.7.1 系统时钟选择寄存器 <i>SYSCLK_SEL</i>	49
8 低功耗模式	51
8.1 低功耗模式的进入	51
8.2 休眠模式	51
8.3 深度睡眠模式	51
8.4 低功耗模式的唤醒	51
9 GPIO	52
9.1 特征	52
9.2 输入输出方向控制	52
9.3 端口输入中断	52
9.4 端口映射操作	53
9.5 端口输入内部上拉或下拉	53
9.6 端口输出驱动能力	54
9.7 端口开漏输出功能	54
9.8 端口施密特功能	54
9.9 端口复用功能	54
9.10 寄存器列表	55
9.11 寄存器描述	58
9.11.1 端口数据寄存器 <i>Pn_DAT</i>	58
9.11.2 端口数据输出锁存寄存器 <i>Pn_LAT</i>	59
9.11.3 端口输出使能置位寄存器 <i>Pn_OES</i>	60
9.11.4 端口输出使能清除寄存器 <i>Pn_OEC</i>	61
9.11.5 外设复用置位寄存器 0 <i>Pn_AFSO</i>	62
9.11.6 外设复用清除寄存器 0 <i>Pn_AFC</i>	63
9.11.7 中断使能置位寄存器 <i>Pn_IES</i>	64
9.11.8 中断使能清除寄存器 <i>Pn_IEC</i>	65
9.11.9 中断类型置位寄存器 0 <i>Pn_ITS0</i>	66
9.11.10 中断类型清除寄存器 0 <i>Pn_ITC0</i>	67

9.11.11	中断类型置位寄存器 1 Pn_ITS1	68
9.11.12	中断类型清除寄存器 1 Pn_ITC1	69
9.11.13	中断极性设置寄存器 Pn_PLS	70
9.11.14	中断极性清除寄存器 Pn_PLC	71
9.11.15	中断标志寄存器 Pn_IF	72
9.11.16	内部上拉使能置位寄存器 Pn_PUS	73
9.11.17	内部上拉使能清除寄存器 Pn_PUC	74
9.11.18	内部下拉使能置位寄存器 Pn_PDS	75
9.11.19	内部下拉使能清除寄存器 Pn_PDC	76
9.11.20	输出开漏使能置位寄存器 Pn_ODS	77
9.11.21	输出开漏使能清除寄存器 Pn_ODC	78
9.11.22	模拟功能使能置位寄存器 Pn_ANAS	79
9.11.23	模拟功能使能清除寄存器 Pn_ANAC	80
9.11.24	外设复用置位寄存器 1 PA_AFS1	81
9.11.25	端口低 8 位映射操作区域 Pn_MASKL	82
9.11.26	端口高 8 位映射操作区域 Pn_MASKH	82
10	系统定时器	83
10.1	综述	83
10.2	寄存器列表	83
10.3	寄存器描述	84
10.3.1	控制状态寄存器 CORET_CSR	84
10.3.2	回填值寄存器 CORET_RVR	85
10.3.3	当前值寄存器 CORET_CVR	85
11	高级定时器	86
11.1	综述	86
11.2	功能框图	86
11.3	定时器匹配值更新	89
11.4	定时器 TMR0, TMR1 通道捕捉	92
11.4.1	捕捉通道信号同步滤波	92
11.4.2	捕捉功能描述	92
11.5	PWM 输出	93
11.5.1	PWM 模式	93
11.5.2	死区时间插入	96
11.5.3	刹车插入	97
11.6	寄存器列表	100
11.7	寄存器描述	102
11.7.1	定时器 TMRn 中断寄存器 TMRn_IR (n = 0, 1)	102
11.7.2	定时器 TMRn 中断累计次数控制寄存器 TMRn_INT_RPT (n = 0, 1)	103
11.7.3	定时器 TMRn 中断累计次数计数寄存器 TMRn_INT_RPTC (n = 0, 1)	104
11.7.4	定时器 TMRn 控制寄存器 TMRn_TCR (n = 0, 1)	104
11.7.5	定时器 TMRn 预分频系数寄存器 TMRn_PR (n = 0, 1)	105
11.7.6	定时器 TMRn 预分频当前计数值寄存器 TMRn_PC (n = 0, 1)	106
11.7.7	定时器 TMRn 匹配值寄存器 TMRn_MRx (n = 0, 1; x = 0, 1, 2, 3, 4)	106

11.7.8 定时器 TMRn 当前计数值寄存器 TMRn_TC (n = 0, 1)	107
11.7.9 定时器 TMRn 匹配控制寄存器 TMRn_MCR (n = 0, 1)	107
11.7.10 定时器 TMRn 捕获控制寄存器 TMRn_CCR (n = 0, 1)	109
11.7.11 定时器 TMRn 捕捉值寄存器 TMRn_CRx (n = 0, 1; x = 1, 2, 3, 4)	111
11.7.12 定时器 TMRn 比较输出控制寄存器 TMRn_CMR (n = 0, 1)	112
11.7.13 定时器 TMRn 死区时间控制寄存器 TMRn_DT (n = 0, 1)	114
12 普通定时器	115
12.1 普通定时器综述	115
12.2 定时器 TMR2, TMR3 框图	115
12.3 寄存器列表	117
12.4 寄存器描述	118
12.4.1 定时器 TMRn 中断寄存器 TMRn_IR (n = 2, 3)	118
12.4.2 定时器 TMRn 控制寄存器 TMRn_TCR (n = 2, 3)	118
12.4.3 定时器 TMRn 预分频系数寄存器 TMRn_PR (n = 2, 3)	119
12.4.4 定时器 TMRn 预分频当前计数值寄存器 TMRn_PC (n = 2, 3)	120
12.4.5 定时器 TMRn 匹配值寄存器 TMRn_MR0 (n = 2, 3)	120
12.4.6 定时器 TMRn 当前计数值寄存器 TMRn_TC (n = 2, 3)	121
12.4.7 定时器 TMRn 匹配控制寄存器 TMRn_MCR (n = 2, 3)	121
13 低功耗定时器 TMR4	123
13.1 定时器综述	123
13.2 定时器 TMR4 框图	123
13.3 寄存器列表	124
13.4 寄存器描述	125
13.4.1 定时器 TMR4 中断寄存器 TMR4_IR	125
13.4.2 定时器 TMR4 控制寄存器 TMR4_TCR	125
13.4.3 定时器 TMR4 预分频系数寄存器 TMR4_PR	126
13.4.4 定时器 TMR4 预分频当前计数值寄存器 TMR4_PC	127
13.4.5 定时器 TMR4 匹配值寄存器 TMR4_MR0	127
13.4.6 定时器 TMR4 当前计数值寄存器 TMR4_TC	128
13.4.7 定时器 TMR4 匹配控制寄存器 TMR4_MCR	128
14 蜂鸣器	129
14.1 蜂鸣器综述	129
14.2 蜂鸣器框图	129
14.3 蜂鸣器软件操作说明	129
14.4 寄存器列表	129
14.5 寄存器描述	130
14.5.1 蜂鸣器控制寄存器 BEEP_CON	130
15 系统看门狗	131
15.1 系统看门狗综述	131
15.2 系统看门狗功能描述	131
15.3 系统看门狗软件操作说明	131

15.3.1	启动系统看门狗.....	131
15.3.2	系统看门狗.....	131
15.4	寄存器列表.....	132
15.5	寄存器描述.....	132
15.5.1	SYSWDG 计数重载寄存器 SYSWDGLOAD.....	132
15.5.2	SYSWDG 计数值寄存器 SYSWDGVALUE.....	133
15.5.3	SYSWDG 控制寄存器 SYSWDGCTRL.....	133
15.5.4	SYSWDG 中断清除寄存器 SYSWDGINTCLR.....	134
15.5.5	SYSWDG 原始中断标志寄存器 SYSWDGRIS.....	134
15.5.6	SYSWDG 掩蔽中断标志寄存器 SYSWDGMIS.....	135
15.5.7	SYSWDG 锁定控制寄存器 SYSWDGLOCK.....	135
16	独立看门狗.....	137
16.1	独立看门狗综述.....	137
16.2	独立看门狗功能描述.....	137
16.3	独立看门狗软件操作说明.....	137
16.3.1	启动独立看门狗.....	137
16.3.2	独立看门狗中断.....	137
16.4	寄存器列表.....	138
16.5	寄存器描述.....	138
16.5.1	IWDG 计数重载寄存器 IWDGLOAD.....	138
16.5.2	IWDG 计数值寄存器 IWDGVALUE.....	139
16.5.3	IWDG 控制寄存器 IWDGCTRL.....	139
16.5.4	IWDG 中断清除寄存器 IWDGINTCLR.....	140
16.5.5	IWDG 原始中断标志寄存器 IWDGRIS.....	140
16.5.6	IWDG 掩蔽中断标志寄存器 IWDGMIS.....	141
16.5.7	IWDG 锁定控制寄存器 IWDGLOCK.....	141
17	UART.....	143
17.1	UART 模块综述.....	143
17.2	UART 主要特性.....	143
17.3	UART 功能概述.....	144
17.4	UART 数据格式.....	144
17.5	UART 波特率计算和设定.....	145
17.6	UART 数据发送.....	147
17.7	UART 数据接收.....	148
17.8	UART 错误检测机制.....	148
17.9	UART 中断响应.....	148
17.10	UART 红外通讯功能.....	149
17.11	UART ISO7816 功能.....	150
17.12	UART 单线半双工通讯模式.....	151
17.13	寄存器列表.....	152
17.14	寄存器描述.....	153
17.14.1	收发数据 FIFO 缓冲寄存器 UARTn_DAT.....	153
17.14.2	模块控制寄存器 UARTn_CTL.....	154

17.14.3	波特率控制寄存器 UARTn_BR.....	156
17.14.4	中断控制寄存器 UARTn_IE.....	157
17.14.5	状态寄存器 UARTn_ST.....	158
17.14.1	状态清除寄存器 UARTn_STCLR.....	159
17.14.2	帧间隔时间寄存器 UARTn_GT.....	161
17.14.3	超时控制寄存器 UARTn_TO.....	161
17.14.4	发送队列复位寄存器 UARTn_TXFR.....	162
17.14.5	接收队列复位寄存器 UARTn_RXFR.....	162
17.14.6	ISO7816 收发控制寄存器 UARTn_ISO7816.....	163
17.14.7	红外发送解调控制寄存器 UARTn_IR_CTL.....	164
17.14.8	红外发送占空比控制寄存器 UARTn_IR_PWMC.....	165
18	SPI.....	166
18.1	概述.....	166
18.2	SPI 通讯信号.....	166
18.3	SPI 工作模式.....	166
18.3.1	TI 同步串行模式.....	166
18.3.2	National Microwire 模式.....	167
18.3.3	Motorola 模式.....	167
18.4	SPI 通讯波特率计算和设定.....	169
18.5	SPI 中断.....	170
18.6	寄存器列表.....	170
18.7	寄存器描述.....	171
18.7.1	SPIn 控制寄存器 0 SPIn_CRO.....	171
18.7.2	SPIn 控制寄存器 1 SPIn_CR1.....	172
18.7.3	SPIn 数据寄存器 SPIn_DR.....	173
18.7.4	SPIn 状态寄存器 SPIn_SR.....	173
18.7.5	SPIn 时钟预分频寄存器 SPIn_CPSR.....	173
18.7.6	SPIn 中断使能寄存器 SPIn_IE.....	175
18.7.7	SPIn 原始中断标志寄存器 SPIn_RIS.....	176
18.7.8	SPIn 使能中断标志寄存器 SPIn_MIS.....	177
18.7.9	SPIn 中断标志清除寄存器 SPIn_ICR.....	178
18.7.10	SPIn 片选信号控制寄存器 SPIn_CSCR.....	178
19	I2C.....	179
19.1	I2C 模块综述.....	179
19.2	I2C 模块综述.....	179
19.3	I2C 协议简述.....	181
19.3.1	起始位.....	181
19.3.2	从机寻址.....	181
19.3.3	数据传输.....	182
19.3.4	停止位.....	182
19.3.5	重复起始位.....	182
19.3.6	总线仲裁.....	182
19.3.7	时钟同步.....	183

19.3.8 通讯握手.....	183
19.3.9 时钟延展.....	183
19.3.10 广播呼叫寻址.....	184
19.4 I2C 主模式（主机）.....	184
19.4.1 I2C 主模式寻址方式.....	184
19.4.2 I2C 主模式数据发送.....	184
19.4.3 I2C 主模式数据接收.....	184
19.4.4 I2C 主模式错误信息.....	185
19.5 I2C 从模式（从机）.....	185
19.5.1 I2C 从模式地址匹配.....	185
19.5.2 I2C 从模式数据接收.....	185
19.5.3 I2C 从模式数据发送.....	186
19.5.4 I2C 从模式通讯终止.....	186
19.5.5 I2C 从模式错误信息.....	186
19.6 I2C 时钟速度计算和设定.....	186
19.7 I2C 状态信息和中断响应.....	188
19.8 寄存器列表.....	190
19.9 寄存器描述.....	191
19.9.1 I2C 控制设定寄存器 I2C_CTRLSET.....	191
19.9.2 I2C 控制设定寄存器 I2C_STAT.....	193
19.9.3 I2C 数据寄存器 I2C_DATA.....	193
19.9.4 I2C 地址寄存器 I2C_ADDR.....	194
19.9.5 I2C 控制清除寄存器 I2C_CTRLCLR.....	194
20 ADC.....	196
20.1 概述.....	196
20.2 特点.....	196
20.3 ADC 控制时序.....	196
20.3.1 ADC 启动时序.....	196
20.3.2 ADC 转换时序.....	196
20.4 用户操作.....	198
20.4.1 ADC 单次转换模式.....	198
20.4.2 连续转换模式.....	198
20.5 寄存器列表.....	199
20.6 寄存器描述.....	199
20.6.1 ADC 控制寄存器 ADC_CON.....	199
20.6.2 ADC 复位释放时间寄存器 ADC_TRSTN.....	200
20.6.3 ADC 状态寄存器 ADC_STAT.....	201
20.6.4 ADC 状态寄存器 ADC_DATA.....	201
21 比较器.....	202
21.1 概述.....	202
21.2 结构框图.....	202
21.3 功能描述.....	202
21.3.1 比较器控制.....	202

21.3.2	数字滤波.....	204
21.3.3	中断生成.....	204
21.3.4	睡眠唤醒.....	205
21.3.5	OPA 控制.....	206
21.4	寄存器列表.....	206
21.5	寄存器描述.....	207
21.5.1	比较器控制寄存器 COMP_CTRL.....	207
21.5.2	比较器控制寄存器 COMP_AMPCTRL.....	208
21.5.3	比较器 DAC 输入数据寄存器 COMP_DAC.....	208
21.5.4	比较器中断控制寄存器 COMP_IE.....	209
21.5.5	比较器中断标志寄存器 COMP_IF.....	210
21.5.6	比较器初始化延时配置寄存器 COMP_INITCNT.....	211
22	CRC 模块.....	212
22.1	概述.....	212
22.2	用户操作.....	212
22.3	寄存器列表.....	212
22.4	寄存器描述.....	213
22.4.1	CRC 控制寄存器 (CRC_CTRL).....	213
22.4.2	CRC 种子寄存器 (CRC_SEED).....	214
22.4.3	CRC 多项式寄存器 (CRC_POLY).....	215
22.4.4	CRC 数据输入寄存器 (CRC_DIN).....	215
22.4.5	CRC 数据输出寄存器 (CRC_DOUT).....	216
23	模拟功能杂项控制.....	217
23.1	概述.....	217
23.2	寄存器列表.....	217
23.3	寄存器描述.....	218
23.3.1	LVD 控制寄存器 LVD_CTRL.....	218
23.3.2	高频 crystal 振荡控制寄存器 1 OSC_CON1.....	219
23.3.3	高频 crystal 振荡控制寄存器 2 OSC_CON2.....	220
23.3.4	内部 24M RC 振荡控制寄存器 IOSC_24M_CON.....	221
23.3.5	内部 32K RC 振荡控制寄存器 IOSC_32K_CON.....	221
23.3.6	倍频模块控制寄存器 FD_CON.....	222
23.3.7	Deglitch 延时控制寄存器 DEGLITCH_CON.....	222
23.3.8	HighSink GPIO 控制寄存器 GPIO_HS_CON.....	223
23.3.9	主时钟输出分频控制寄存器 MCO_DIV_CON.....	224
24	FLASH 控制器.....	225
24.1	FLASH 特性.....	225
24.2	FLASH 区域划分.....	225
24.3	FLASH 操作控制.....	226
24.4	FLASH 擦写.....	226
24.5	FLASH 内容的安全保护.....	226
24.6	FLASH 启动顺序.....	226

24.6.1 正常启动.....	227
24.6.2 BootLoader 启动.....	227
24.7 FLASH 擦写时钟选择.....	227
24.8 寄存器列表.....	228
24.9 寄存器描述.....	228
24.9.1 FLASH 命令寄存器 FLCMD.....	228
24.9.2 FLASH 中断状态寄存器 FLISR.....	230
24.9.3 FLASH 中断使能寄存器 FLIER.....	231
24.9.4 FLASH 地址寄存器 FLAR.....	232
24.9.5 FLASH 编程数据寄存器 FLDR.....	233
24.9.6 FLASH 编程数据寄存器 FLDIV.....	233
25 修改历史.....	234

1 概述

PT32L031 芯片是一颗兼容 1.8V 至 3.6V 的 MCU 主控芯片，具有很高的集成度和较高的性能。芯片使用 M0 内核，有比较齐全的数字、模拟外设。

1.1 ARM M0 内核

- 最高 CPU 运行时钟频率 48MHz
- 外设总线与内核时钟频率互相独立
- 优先级控制的中断嵌套响应机制

1.2 低功耗模式

- 工作电压：1.8V~3.6V
- 支持上电复位
- 深度睡眠时功耗 0.9uA
- 提供 Sleep 和 Deep Sleep 两种低功耗模式
- 在 Sleep 模式下，所有中断源可以唤醒
- 在 Deep Sleep 模式下，提供多种唤醒源（IO 唤醒，外部复位唤醒，独立看门狗唤醒，比较器唤醒，低功耗 Timer 唤醒，仿真器调试唤醒等）

1.3 存储器

- 指令存储器：16K-Byte/32K-Byte FLASH
- 数据存储器：2K-Byte/4K-Byte SRAM
- BootLoader 区域会占用 2Kbyte，用户信息区会占用 0.5Kbyte

1.4 时钟

- 高速晶体振荡器时钟：4~25MHz，外接 4~25MHz 晶体
- 内置高速 RC 振荡器时钟：25MHz，25°C 时精度±1%，-40~85°C 时精度为±3%
- 内置低速 RC 振荡器时钟：32KHz，精度±10%，模块功耗 3 微安，可供看门狗和时钟监测模块使用
- 2 倍频时钟：高速晶体振荡时钟的 2 倍频
- 分频器时钟：高速时钟的 2/4/8/16/32 分频时钟，外设时钟和 CPU 时钟独立

1.5 高精度 ADC

- 12 位高精度
- 变化速率为 500Ksps/1Msps
- 至多 13 路通道输入，最后一路通道固定接 bandgap 电压

- 支持单次转换 (single mode) /连续转换 (continuous scan mode)
- 支持外部 I/O 触发一次转换 (上升沿, 下降沿, 任意电平切换)
- 支持内部定时器 timer0/timer1/timer2/timer3 定时触发一次转换

1.6 比较器

- 2 路模拟比较器
- 比较器模式下,比较结果可触发中断,可用于唤醒系统. 比较器结果可通过寄存器读取, 也可通过指定管脚输出 (推挽输出)

1.7 蜂鸣器

- 单独外设模块可输出不同频率方波驱动蜂鸣器

1.8 串口

- 2 个 IIC 串口
 - 支持主/从模式
 - 支持标准速率 (100Kb/s)
- 2 个 SPI 串口
 - 支持主/从模式
 - 支持标准 SPI 接口
- 2 个 UART 串口
 - 支持波特率可编程
 - 所有串口支持无极性 RX 接收模式
 - 其中 1 个固定支持 38KHz 低功耗红外发送调制
 - 所有串口支持单线通讯
 - 可支持 ISO7816, 数据单线半双工收发, 提供 2/4/8/16/32/64/128 外设分频时钟输出

1.9 定时器 (Timer)

- 1 个 24 位系统定时器 (M0 内核自带)
- 2 个 16 位高级定时器, timer0 & timer1
 - 支持 8 位预分频
 - 支持 16 位递增计数, 递减计数, 递减递增交替计数
 - 每个 timer 支持 4 路输入捕捉, 4 路比较输出
 - 输入捕捉支持上升沿捕捉, 下降沿捕捉, 上升沿至下降沿捕捉, 下降沿至上升沿捕捉
 - 比较输出支持死区时间可调, 支持互补 pwm 输出
 - 支持输入中断, 输出中断和溢出中断以及 break input

- 2 个 16 位普通定时器，timer2 & timer3
 - 支持 8 位预分频可用于触发 ADC 转换
 - 支持 16 位递增计数，递减计数
 - 可用于触发 ADC 转换

- 1 个 16 位低功耗定时器，timer4
 - 支持 16 位递增计数，使用内部低速振荡 32kHz 时钟作为计数时钟，可唤醒系统

1.10 安全机制

- 片上看门狗
 - 系统看门狗，32 位递减计数，使用系统时钟作为计数时钟
 - 独立看门狗，32 位递减计数，使用内部低速振荡 32kHz 时钟作为计数时钟
- 低电压监控，当电压低于安全值时，输出中断或复位
 - 触发阈值支持：4V，3.5V，3V，2.75V，2.5V，2.2V，2.0V，1.7V
- 晶体振荡器/倍频时钟监控，当晶体振荡器/倍频时钟作为系统主时钟时，一旦晶体振荡/倍频时钟失效，芯片可以自动切换到内部高速振荡器 24MHz 时钟

1.11 GPIO

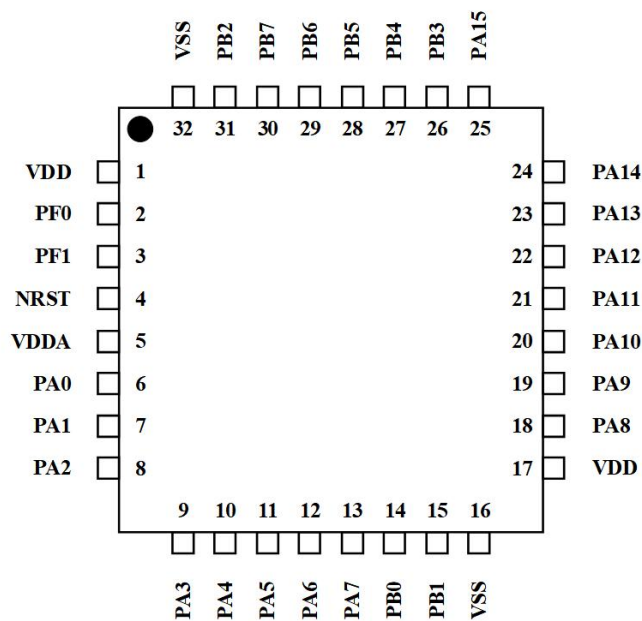
- 提供至多 26 个 GPIO 口
- 每个 GPIO 口均可提供外部中断并用于唤醒系统
- 支持 4 个 GPIO 承受较大灌电流，电流强度可达 70mA
- 支持弱上拉（下拉）功能，上拉（下拉）电阻为 50kΩ
- 支持输出强驱动，普通驱动电流为 8 mA，强输出驱动电流为 20mA
- 支持开漏功能
- 支持模拟模式（作为 ADC 输入或比较器(OPA)输入 pad 时）

1.12 调试接口

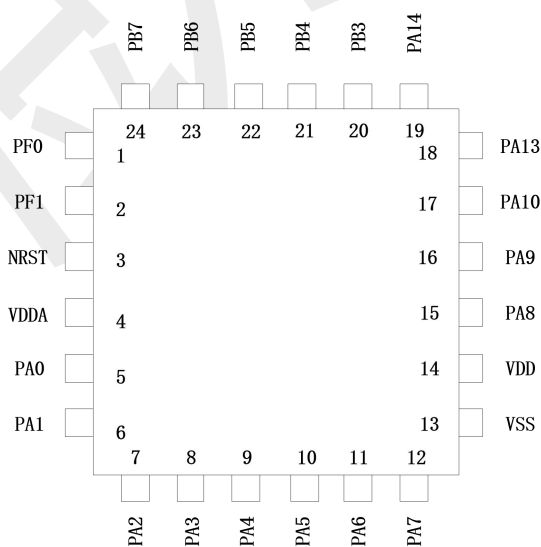
- 使用 swd 标准两线制调试接口

2 封装信息

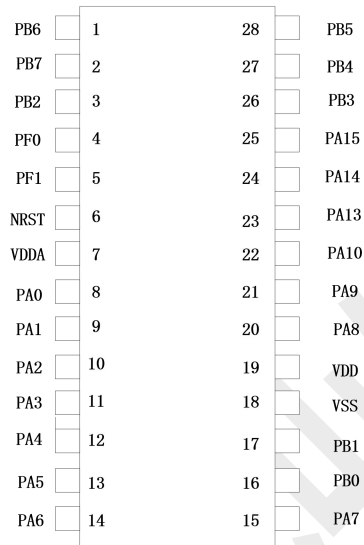
2.1 LQFP32/QFN32(4*4)/QFN32(5*5)



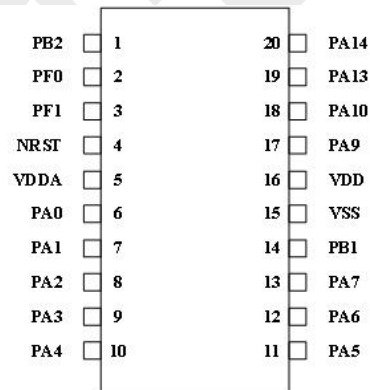
2.2 QFN24(4*4)



2.3 TSSOP28



2.4 TSSOP20



3 管脚定义

32 脚 Q/L	28 脚 T	24 脚 Q	20 脚 T	引脚名	数字功能							模拟功能	
					主功能	替换功能 0	替换功能 1	替换功能 2	替换功能 3	替换功能 4	替换功能 5		
1	-	-	-	VDD	VDD								
2	4	1	2	PF0	PF0	OSC_IN	I2C0_SDA						
3	5	2	3	PF1	PF1	OSC_OUT	I2C0_SCL						
4	6	3	4	NRST	NRST								
5	7	4	5	VDDA	VDDA								
6	8	5	6	PA0	PA0		TIM0_CH1N						ADC_IN0 CMP_OPA_1_P
7	9	6	7	PA1	PA1		TIM0_CH2N						ADC_IN1 CMP_OPA_1_N
8	10	7	8	PA2	PA2	UART0_TX	TIM0_CH3N		UART1_TX				ADC_IN2 OPA1_OUT
9	11	8	9	PA3	PA3	UART0_RX	TIM0_CH4N		UART1_RX				ADC_IN3 CMP_OPA_0_P
10	12	9	10	PA4	PA4	SPI0_CS	UART1_TX		UART0_TX				ADC_IN4 CMP_OPA_0_N
11	13	10	11	PA5	PA5	SPI0_SCK	UART1_RX		UART0_RX				ADC_IN5 OPA0_OUT
12	14	11	12	PA6	PA6	SPI0_MISO	TIM1_BKIN						ADC_IN6
13	15	12	13	PA7	PA7	SPI0_MOSI	TIM1_CH1N	ADC_ETR					ADC_IN7
14	16	-	-	PB0	PB0		TIM1_CH2N						
15	17	-	14	PB1	PB1	IR_TXD	TIM1_CH3N						ADC_IN8
16	18	13	15	VSS	VSS								
17	19	14	16	VDD	VDD								
18	20	15	-	PA8	PA8	MCO	TIM1_CH1						
19	21	16	17	PA9	PA9	CMP0_OUT	TIM1_CH2		UART1_TX	I2C1_SCL	MCO		
20	22	17	18	PA10	PA10	CMP1_OUT	TIM1_CH3		UART1_RX	I2C1_SDA			
21	-	-	-	PA11	PA11	I2C1_SCL	TIM1_CH4	I2C0_SCL					
22	-	-	-	PA12	PA12	I2C1_SDA	TIM1_CH4N	I2C0_SDA					
23	23	18	19	PA13	PA13	SWDIO	IR_TXD						
24	24	19	20	PA14	PA14	SWCLK		UART1_TX	UART0_TX				
25	25	-	-	PA15	PA15		SPI1_CS	UART1_RX	UART0_RX				
26	26	20	-	PB3	PB3	TIM0_BKIN	SPI1_SCK						ADC_IN10
27	27	21	-	PB4	PB4	TIM0_CH1	SPI1_MISO						ADC_IN11
28	28	22	-	PB5	PB5	TIM0_CH2	SPI1_MOSI						
29	1	23	-	PB6	PB6	TIM0_CH3	I2C1_SCL	UART1_TX	UART0_TX	I2C0_SCL			
30	2	24	-	PB7	PB7	TIM0_CH4	I2C1_SDA	UART1_RX	UART0_RX	I2C0_SDA			
31	3	-	1	PB2	PB2	BEEP	MCO						ADC_IN9
32	-	-	-	VSS	VSS								

Note: 32 脚, QFN32/LQFP32; 28 脚, TSSOP28; 20 脚, TSSOP20

4 框图

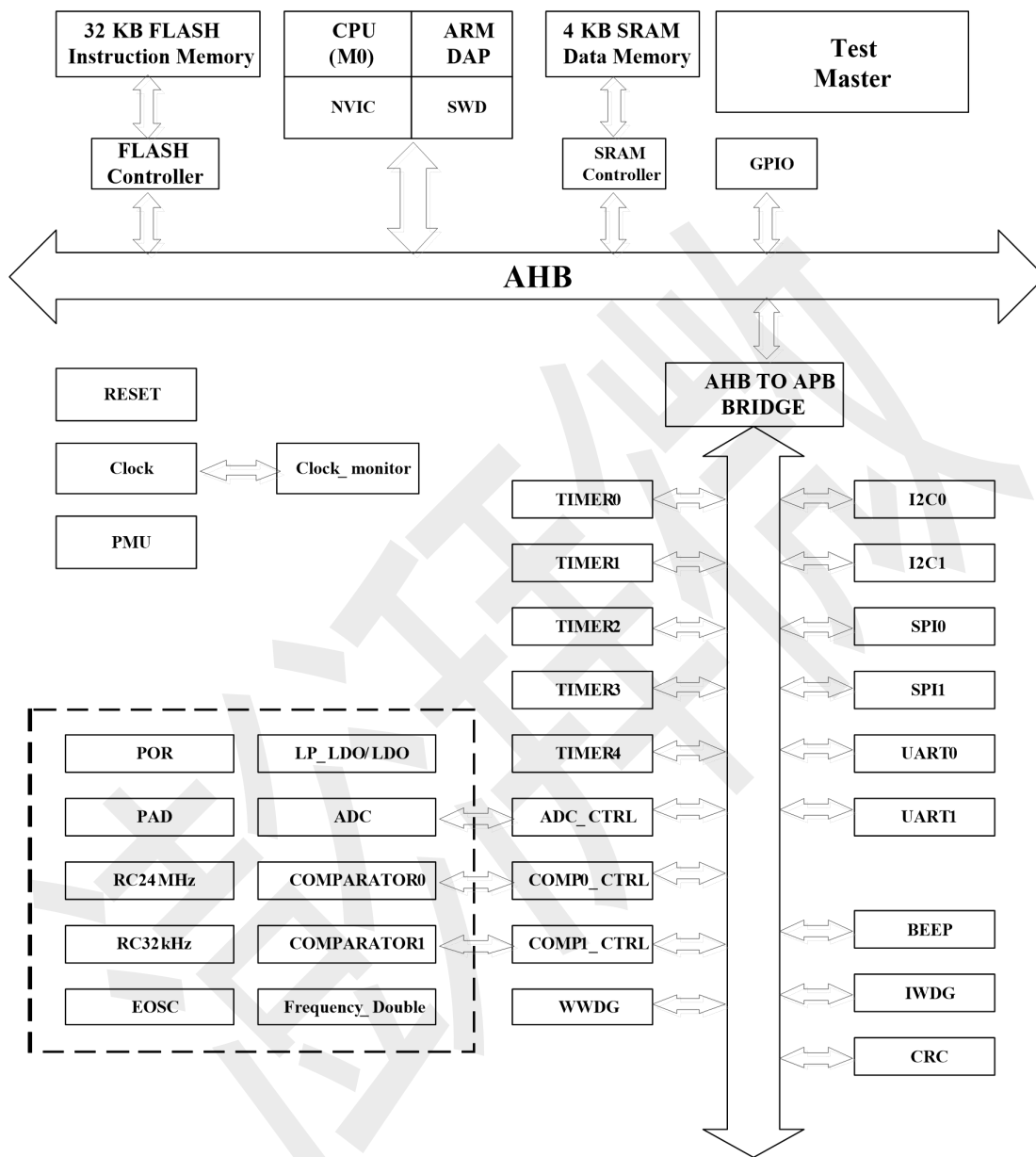


图 4-1 系统框架图

5 地址空间分配

PT32L031 使用统一的物理编址，下面介绍芯片的具体地址分配。

5.1 AHB 地址分配

AHB Memory Address	Size	Space
0x0000_0000 – 0x0000_7FFF	32KB	指令 FLASH 空间
0x0000_8000 – 0x0000_F5FF	32KB	FLASH 主代码空间映射 (仅在执行 bootloader 时有效)
0x0000_F600 – 0x1FFF_FFFF	-	保留
0x2000_0000 – 0x2000_0FFF	4KB	SRAM 数据区
0x2000_1000 – 0x3FFF_FFFF	-	保留
0x4000_0000 – 0x4001_7FFF	96KB	APB
0x4001_8000 – 0x4001_EFFF	-	保留
0x4001_F000 – 0x4001_FFFF	4KB	系统控制寄存器区
0x4002_0000 – 0x47FF_FFFF	-	保留
0x4800_0000 – 0x4800_0FFF	4KB	PA 寄存器区
0x4800_1000 – 0x4800_1FFF	4KB	PB 寄存器区
0x4800_2000 – 0x4800_2FFF	4KB	PF 寄存器区
0x4800_3000 – 0x4800_3FFF	-	保留
0x4800_4000 – 0xE000_DFFF	-	保留
0xE000_E000 – 0xE000_EFFF	4KB	内核控制寄存器区 (补)
0xE000_F000 – 0xFFFF_FFFF	-	保留
0xF000_0000 – 0xF000_0FFF	-	保留
0xF000_1000 – 0xFFFF_FFFF	-	保留 (访问异常区)

5.2 APB 地址映射

APB Memory Address	Size	Peripherals
0x4000_0000 – 0x4000_07FF	2KB	FLASH Controller
0x4000_0800 – 0x4000_0BFF	1KB	Comparator0
0x4000_0C00 – 0x4000_0FFF	1KB	Comparator1
0x4000_1000 – 0x4000_13FF	1KB	TIMER2
0x4000_1400 – 0x4000_17FF	1KB	TIMER3
0x4000_1800 – 0x4000_1BFF	1KB	Analog MISC controller
0x4000_1C00 – 0x4000_1FFF	1KB	TIMER4 (LPTMR)
0x4000_1C00 – 0x4000_1FFF	1KB	-
0x4000_2000 – 0x4000_27FF	2KB	SYSWDG
0x4000_2800 – 0x4000_2BFF	1KB	BEEP

0x4000_2C00 – 0x4000_2FFF	1KB	LED
0x4000_3000 – 0x4000_37FF	2KB	IWDG
0x4000_3800 – 0x4000_3BFF	1KB	SPI1
0x4000_3C00 – 0x4000_3FFF	1KB	CRC
0x4000_4000 – 0x4000_43FF	1KB	保留
0x4000_4400 – 0x4000_47FF	1KB	UART0
0x4000_4800 – 0x4000_53FF	3KB	保留
0x4000_5400 – 0x4000_57FF	1KB	I2C0
0x4000_5800 – 0x4000_5BFF	1KB	I2C1
0x4000_5C00 – 0x4001_23FF	50KB	保留
0x4001_2400 – 0x4001_27FF	1KB	ADC
0x4001_2800 – 0x4001_2BFF	1KB	TIMER0
0x4001_2C00 – 0x4001_2FFF	1KB	TIMER1
0x4001_3000 – 0x4001_33FF	1KB	SPI0
0x4001_3400 – 0x4001_37FF	1KB	保留
0x4001_3800 – 0x4001_3BFF	1KB	UART1
0x4001_3C00 – 0x4001_7FFF	17KB	保留

5.3 FLASH NVR 配置

在 FLASH 的最后一个 sector（地址范围 0x0000_7E00 ~ 0x0000_0x7FFF）保存着用户可配置参数信息，这些参数可由用户擦写，芯片上电后自动读取这些参数并映射到对应的寄存器，用户不可访问该区域，可以通过对应的映射寄存器查看参数。该区域内不可运行程序（即 PC 指针不能往此区域跳转）

Address	Name	Bits	Description
0x0000_7E00	CUSTOMER_ID	31~0	CUSTOMER_ID_INFO[31:0] 用户可以通过 CUSTOMER_ID 寄存器查看该值
0x0000_7E04	FLASH_BOOT	7~0	若bit[7:0] = 8'hC4，Flash将选择从bootloader启动，否则将直接启动去执行主程序 用户可以通过REMAP寄存器查看当前执行的程序状态
0x0000_7E08	FLASH_PROTECT	31~0	若bit[31:0] = 32'h15EC1CCA，Flash将处于保护状态，否则Flash不处于保护状态 用户可以通过FLASH_PROTECT寄存器查看上电后的Flash保护状态（仅在每次上电时更新）
0x0000_7E0C	IWDG_EN	7~0	若bit[7:0] = 8'hB5，独立看门狗将默认启动，否则独立看门狗默认关闭

5.4 UID

为方便产品追踪和维护管理，芯片在制造阶段可将制造过程的特有信息（UID）写入 NVR 区域，作为唯一标签识别号。芯片在上电 BOOT 过程中将 NVR 区域内的数据读出并复制到系统控制相应的寄存器组中，供应用随时读取。

上述 UID 共计 96bit（3 个 Word），用户可从系统控制寄存器单元对应的寄存器组读取 UID 信息，具体的寄存器定义请参见 UID_INFO1，UID_INFO2，UID_INFO3 寄存器。

5.5 系统控制寄存器列表

地址	寄存器	描述	备注
0x4001_F000	REMAP	Boot 模式寄存器	REMAP 说明
0x4001_F00C	SYSCLK_SEL	系统时钟控制寄存器	SYSCLK_SEL 说明
0x4001_F010	RSTINFO	复位信息寄存器	RSTINFO 说明
0x4001_F014	RETRIMING_CTRL	重启控制寄存器	RETRIMING_CTRL 说明
0x4001_F018	RESET_CTRL	复位使能控制寄存器	RESET_CTRL 说明
0x4001_F020	CUSTOMER_ID	客户 ID 信息	CUSTOMER_ID 说明
0x4001_F024	FLASH_PROTECT	FLASH 保护状态寄存器	FLASH_PROTECT 说明
0x4001_F030	UID_INFO1	UID 信息[31:0]	UID_INFO1 说明
0x4001_F034	UID_INFO2	UID 信息[63:32]	UID_INFO2 说明
0x4001_F038	UID_INFO3	UID 信息[95:64]	UID_INFO3 说明
0xE000_ED10	SCR	系统控制寄存器	SCR 说明

5.6 系统控制寄存器描述
5.6.1 Boot 模式寄存器 REMAP

(地址: 0x4001_F000)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	BOOT
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] BOOT:** **BOOT 模式**
 该位仅写 0 有效
 在执行 bootloader 程序时写 0 之后需系统复位才能转去用户主程序
 0b: 当前所执行程序为用户程序
 1b: 当前所执行程序为 bootloader 程序
- **位[31:1] :** **保留**

5.6.2 复位信息寄存器 RSTINFO

(地址: 0x4001_F010)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	LDO_1p5V	LVD_5V	EXRST	POR	FD_FAIL	EOSC_FAIL	SYSWDG	-	IWDG	SYSRSTREQ
R/W	R	R	R	R	R	R	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	R	RW1c	RW1c
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] SYSRSTREQ:** **系统软复位导致复位**
 0b: 未产生复位
 1b: 产生过复位, 软件可对该位写 1 将其清除
- **位[1] IWDG:** **独立看门狗复位导致复位**
 0b: 未产生复位
 1b: 产生过复位, 软件可对该位写 1 将其清除

- 位[2] : 保留
- 位[3] SYSWDG: 系统看门狗复位导致复位
 - 0b: 未产生复位
 - 1b: 产生过复位, 软件可对该位写 1 将其清除
- 位[4] EOSC_FAIL: 晶振时钟失效导致复位
 - 0b: 未产生复位
 - 1b: 产生过复位, 软件可对该位写 1 将其清除
- 位[5] FD_FAIL: 倍频时钟失效导致复位
 - 0b: 未产生复位
 - 1b: 产生过复位, 软件可对该位写 1 将其清除
- 位[6] POR: POR 复位导致复位
 - 0b: 未产生复位
 - 1b: 产生过复位, 软件可对该位写 1 将其清除
- 位[7] EXRST: 外部管脚复位导致复位
 - 0b: 未产生复位
 - 1b: 产生过复位, 软件可对该位写 1 将其清除
- 位[8] LVD_5V: LVD_5V 复位导致复位
 - 0b: 未产生复位
 - 1b: 产生过复位, 软件可对该位写 1 将其清除
- 位[9] LDO_1p5V: LDO_1p5V 复位导致复位
 - 0b: 未产生复位
 - 1b: 产生过复位, 软件可对该位写 1 将其清除
- 位[31:10] : 保留

5.6.3 重启控制寄存器 RETRIMING_CTRL

(地址: 0x4001_F014)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	RETRIMING															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[15:0] RETRIMING: 重启密码
当软件写 RESET_CTRL 寄存器的 RETRIMING_EN 为 1, 并且写 RETRIMING 为 0xAB56 时, 芯片将重启复位
- 位[31:16] : 保留

5.6.4 复位使能控制寄存器 RESET_CTRL

(地址: 0x4001_F018)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	RETRIMING_EN	LDO_1p5V_RST_EN	LDO_5V_RST_EN	-	BG_RST_EN	FD_FAIL_RST_EN	EOSC_FAIL_RST_EN	-	-	-	-
R/W	R	R	R	-	R	RW	RW	RW	R	RW	RW	RW	R	R	R	R
复位	0	0	0	-	0	0	1	1	0	1	0	0	0	0	0	0

- 位[3:0] : 保留
- 位[4] EOSC_FAIL_RST_EN: 晶振时钟失效复位使能
0b: 复位禁止
1b: 复位使能
- 位[5] FD_FAIL_RST_EN: 倍频时钟失效复位使能
0b: 复位禁止
1b: 复位使能
- 位[6] BG_RST_EN: BandGap 复位使能
0b: 复位禁止
1b: 复位使能
- 位[7] : 保留
- 位[8] LVD_5V_RST_EN: LVD_5V 复位使能
0b: 复位禁止
1b: 复位使能
- 位[9] LDO_1p5V_RST_EN: LDO_1p5V 复位使能
0b: 复位禁止
1b: 复位使能
- 位[10] RETRIMING_EN: 重启使能
0b: 重启禁止
1b: 重启使能
- 位[31:10] : 保留

5.6.5 CUSTOMER_ID 信息寄存器 CUSTOMER_ID

(地址: 0x4001_F020)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	CUSTOMER_ID															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	CUSTOMER_ID															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- 位[31:0] CUSTOMER_ID: CUSTOMER_ID 信息

5.6.6 FLASH 保护状态寄存器 FLASH_PROTECT

(地址: 0x4001_F024)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	PROTECT
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] PROTECT: FLASH 保护状态

0b: FLASH 数据未保护

1b: FLASH 数据处于保护状态

- 位[31:1] : 保留

5.6.7 UID 信息[31:0]寄存器 UID_INFO1

(地址: 0x4001_F030)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	UID_INFO1															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	UID_INFO1															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- 位[31:0] UID_INFO1: UID 信息 1

5.6.8 UID 信息[63:32]寄存器 UID_INFO2

(地址: 0x4001_F034)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	UID_INFO2															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	UID_INFO2															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- 位[31:0] UID_INFO2: UID 信息 2

5.6.9 UID 信息[95:64]寄存器 UID_INFO3

(地址: 0x4001_F038)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	UID_INFO3															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	UID_INFO3															

R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- 位[31:0] UID_INFO3: **UID 信息 3**

5.6.10 系统控制寄存器 SCR

(地址: 0x E000_ED10)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	SEVON PEND	-	SLEEP DEEP	SLEEP ON EXIT	-
R/W	R	R	R	R	R	R	R	R	R	R	R	RW	R	RW	RW	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] : **保留**
- 位[1] SLEEPONEXIT:
 - 0b: 系统从最低优先级的中断处理程序中退出时, 不进入睡眠模式
 - 1b: 系统从最低优先级的中断处理程序中退出时, 立即进入睡眠模式
- 位[2] SLEEPDEEP: **睡眠模式选择**
 - 0b: 休眠模式
 - 1b: 深度睡眠模式
- 位[3] : **保留**
- 位[4] SEVONPEND: **事件唤醒控制**
 - 0b: 只有使能的中断或者事件可以唤醒 CPU, 未使能的中断不可唤醒 CPU
 - 1b: 未使能的中断或者时间均可唤醒 CPU
- 位[31:5] : **保留**

6 矢量中断控制器

NVIC用于中断的高效处理。矢量中断控制器最大可支持32个中断源。每个中断源拥有独立软件可编程的中断优先级。矢量中断控制器收集来自不同中断源的中断请求，依据中断优先级对中断请求进行仲裁。最高优先级的中断将获得控制权并请求处理器。

矢量中断控制器支持中断嵌套。当处理器正在处理一个中断请求的同时，来了一个更高优先级的中断请求，处理器将中断当前中断服务程序的处理，响应该更高优先级的中断请求。在更高优先级的中断请求处理结束时，CPU 返回被打断的中断服务程序继续执行。矢量中断控制器允许高优先级的中断请求抢占低优先级的中断请求，但不允许同级别或者低优先级的中断抢占，保证了中断响应的实时性。

Exception Number	IRQ Number	Priority	Exception or Interrupt	Vector
-	-	-	Reserved	0x00
1	-	-3	Reset	0x04
2	-14	-2	NMI(IWDG)	0x08
3	-13	-1	HardFault	0x0C
4~10	-	-	Reserved	-
11	-5	configurable	SVCALL	0x2c
12 ~ 13	-	-	Reserved	-
14	-2	configurable	PendSV	0x38
15	-1	configurable	SysTick	0x3C
16	0	configurable	SYSWDG	0x40
17	1	configurable	CLK_FAIL	0x44
18	2	configurable	-	0x48
19	3	configurable	FLASH	0x4C
20	4	configurable	-	0x50
21	5	configurable	PA	0x54
22	6	configurable	PB	0x58
23	7	configurable	PF	0x5c
24	8	configurable	-	0x60
25	9	configurable	Comparator0	0x64
26	10	configurable	Comparator1	0x68
27	11	configurable	-	0x6c
28	12	configurable	ADC	0x70
29	13	configurable	TIM1	0x74
30	14	configurable	TIM0	0x78
31	15	configurable	TIM4	0x7c
32	16	configurable	TIM3	0x80
33	17	configurable	TIM2	0x84
34	18	configurable	LED	0x88
35	19	configurable	-	0x8c

36	20	configurable	LVD_5V	0x90
37	21	configurable	LDO_1p5V	0x94
38	22	configurable	-	0x98
39	23	configurable	I2C0	0x9C
40	24	configurable	I2C1	0xA0
41	25	configurable	SPI0	0xA4
42	26	configurable	SPI1	0xA8
43	27	configurable	UART0	0xAC
44	28	configurable	UART1	0xB0

6.1 矢量中断控制寄存器列表

地址	寄存器	描述	备注
0xE000_E100	ISER	系统中断使能寄存器	ISER 说明
0xE000_E180	ICER	系统中断禁止寄存器	ICER 说明
0xE000_E200	ISPR	系统中断挂起设定寄存器	ISPR 说明
0xE000_E280	ICPR	系统中断挂起清除寄存器	ICPR 说明

6.2 矢量中断控制寄存器描述

6.2.1 系统中断使能寄存器 ISER

(地址: 0xE000_E100)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	UART1	UART0	SPI1	SPI0	I2C1	I2C0	-	LDO_1p5V	LVD_5V	-	-	TIM2	TIM3
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	TIM4	TIM0	TIM1	ADC	-	COMP1	COMP0	-	PF	PB	PA	-	FLASH	-	CLOCK_FAIL	SYSWDG
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] SYSWDG: 系统看门狗中断使能

读:

0b: 系统看门狗中断禁止

1b: 系统看门狗中断使能

写:

0b: 无效

1b: 系统看门狗中断使能

- **位[1] CLOCK_FAIL:** **晶振时钟/倍频时钟失效中断使能**
 读:
 0b: 晶振时钟/倍频时钟中断禁止
 1b: 晶振时钟/倍频时钟中断使能
 写:
 0b: 无效
 1b: 晶振时钟/倍频时钟中断使能
- **位[2] :** **保留**
- **位[3] FLASH:** **FLASH 控制中断使能**
 读:
 0b: FLASH 中断禁止
 1b: FLASH 中断使能
 写:
 0b: 无效
 1b: FLASH 中断使能
- **位[4] :** **保留**
- **位[5] PA:** **PA 端口中断使能**
 读:
 0b: PA 端口中断禁止
 1b: PA 端口中断使能
 写:
 0b: 无效
 1b: PA 端口中断使能
- **位[6] PB:** **PB 端口中断使能**
 读:
 0b: PB 端口中断禁止
 1b: PB 端口中断使能
 写:
 0b: 无效
 1b: PB 端口中断使能
- **位[7] PF:** **PF 端口中断使能**
 读:
 0b: PF 端口中断禁止
 1b: PF 端口中断使能
 写:
 0b: 无效
 1b: PF 端口中断使能
- **位[8] :** **保留**
- **位[9] COMP0:** **比较器 0 中断使能**
 读:
 0b: 比较器 0 中断禁止
 1b: 比较器 0 中断使能
 写:
 0b: 无效

- 1b: 比较器 0 中断使能
- **位[10] COMP1:** 比较器 1 中断使能
- 读:
- 0b: 比较器 1 中断禁止
- 1b: 比较器 1 中断使能
- 写:
- 0b: 无效
- 1b: 比较器 1 中断使能
- **位[11] :** 保留
- **位[12] ADC:** ADC 中断使能
- 读:
- 0b: ADC 中断禁止
- 1b: ADC 中断使能
- 写:
- 0b: 无效
- 1b: ADC 中断使能
- **位[13] TIM1:** **TIMER1 中断使能**
- 读:
- 0b: TIMER1 中断禁止
- 1b: TIMER1 中断使能
- 写:
- 0b: 无效
- 1b: TIMER1 中断使能
- **位[14] TIM0:** **TIMER0 中断使能**
- 读:
- 0b: TIMER0 中断禁止
- 1b: TIMER0 中断使能
- 写:
- 0b: 无效
- 1b: TIMER0 中断使能
- **位[15] TIM4:** **TIMER4 中断使能**
- 读:
- 0b: TIMER4 中断禁止
- 1b: TIMER4 中断使能
- 写:
- 0b: 无效
- 1b: TIMER4 中断使能
- **位[16] TIM3:** **TIMER3 中断使能**
- 读:
- 0b: TIMER3 中断禁止
- 1b: TIMER3 中断使能
- 写:
- 0b: 无效
- 1b: TIMER3 中断使能

- 1b: SPI0 中断使能
- **位[26] SPI1:** SPI1 中断使能
- 读:
- 0b: SPI1 中断禁止
- 1b: SPI1 中断使能
- 写:
- 0b: 无效
- 1b: SPI1 中断使能
- **位[27] UART0:** UART0 中断使能
- 读:
- 0b: UART0 中断禁止
- 1b: UART0 中断使能
- 写:
- 0b: 无效
- 1b: UART0 中断使能
- **位[28] UART1:** UART1 中断使能
- 读:
- 0b: UART1 中断禁止
- 1b: UART1 中断使能
- 写:
- 0b: 无效
- 1b: UART1 中断使能
- **位[31:29] :** 保留

6.2.2 系统中断禁止寄存器 ICER

(地址: 0xE000_E180)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	UART1	UART0	SPI1	SPI0	I2C1	I2C0	-	LDO_1p5V	LVD_5V	-	-	TIM2	TIM3
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	TIM4	TIM0	TIM1	ADC	-	COMP1	COMP0	-	PF	PB	PA	-	FLASH	-	CLOCK_FAIL	SYSWDG
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] SYSWDG:** 系统看门狗中断禁止
- 读:
- 0b: 系统看门狗中断禁止
- 1b: 系统看门狗中断使能
- 写:

- 0b: 无效
- 1b: 系统看门狗中断禁止
- **位[1] CLOCK_FAIL: 晶振时钟/倍频时钟失效中断禁止**
- 读:
- 0b: 晶振时钟/倍频时钟中断禁止
- 1b: 晶振时钟/倍频时钟中断使能
- 写:
- 0b: 无效
- 1b: 晶振时钟/倍频时钟中断禁止
- **位[2] : 保留**
- **位[3] FLASH: FLASH 控制中断禁止**
- 读:
- 0b: FLASH 中断禁止
- 1b: FLASH 中断使能
- 写:
- 0b: 无效
- 1b: FLASH 中断禁止
- **位[4] : 保留**
- **位[5] PA: PA 端口中断禁止**
- 读:
- 0b: PA 端口中断禁止
- 1b: PA 端口中断使能
- 写:
- 0b: 无效
- 1b: PA 端口中断禁止
- **位[6] PB: PB 端口中断禁止**
- 读:
- 0b: PB 端口中断禁止
- 1b: PB 端口中断使能
- 写:
- 0b: 无效
- 1b: PB 端口中断禁止
- **位[7] PF: PF 端口中断禁止**
- 读:
- 0b: PF 端口中断禁止
- 1b: PF 端口中断使能
- 写:
- 0b: 无效
- 1b: PF 端口中断禁止
- **位[8] : 保留**
- **位[9] COMP0: 比较器 0 中断禁止**
- 读:
- 0b: 比较器 0 中断禁止
- 1b: 比较器 0 中断使能

- 写:
- 0b: 无效
- 1b: 比较器 0 中断禁止
- **位[10] COMP1:** 比较器 1 中断禁止
- 读:
- 0b: 比较器 1 中断禁止
- 1b: 比较器 1 中断使能
- 写:
- 0b: 无效
- 1b: 比较器 1 中断禁止
- **位[11] :** 保留
- **位[12] ADC:** ADC 中断禁止
- 读:
- 0b: ADC 中断禁止
- 1b: ADC 中断使能
- 写:
- 0b: 无效
- 1b: ADC 中断禁止
- **位[13] TIM1:** TIMER1 中断禁止
- 读:
- 0b: TIMER1 中断禁止
- 1b: TIMER1 中断使能
- 写:
- 0b: 无效
- 1b: TIMER1 中断禁止
- **位[14] TIM0:** TIMER0 中断禁止
- 读:
- 0b: TIMER0 中断禁止
- 1b: TIMER0 中断使能
- 写:
- 0b: 无效
- 1b: TIMER0 中断禁止
- **位[15] TIM4:** TIMER4 中断禁止
- 读:
- 0b: TIMER4 中断禁止
- 1b: TIMER4 中断使能
- 写:
- 0b: 无效
- 1b: TIMER4 中断禁止
- **位[16] TIM3:** TIMER3 中断禁止
- 读:
- 0b: TIMER3 中断禁止
- 1b: TIMER3 中断使能
- 写:

- 0b: 无效
- 1b: TIMER3 中断禁止
- **位[17] TIM2:** **TIMER2 中断禁止**
- 读:
- 0b: TIMER2 中断禁止
- 1b: TIMER2 中断使能
- 写:
- 0b: 无效
- 1b: TIMER2 中断禁止
- **位[18] :** **保留**
- **位[19] :** **保留**
- **位[20] LVD_5V:** **电源低电压中断禁止**
- 读:
- 0b: 电源低电压中断禁止
- 1b: 电源低电压中断使能
- 写:
- 0b: 无效
- 1b: 电源低电压中断禁止
- **位[21] LDO_1P5V:** **LDO_1P5V 低电压中断禁止**
- 读:
- 0b: LDO 低电压中断禁止
- 1b: LDO 低电压中断使能
- 写:
- 0b: 无效
- 1b: LDO 低电压中断禁止
- **位[22] :** **保留**
- **位[23] I2C0:** **I2C0 中断禁止**
- 读:
- 0b: I2C0 中断禁止
- 1b: I2C0 中断使能
- 写:
- 0b: 无效
- 1b: I2C0 中断禁止
- **位[24] I2C1:** **I2C1 中断禁止**
- 读:
- 0b: I2C1 中断禁止
- 1b: I2C1 中断使能
- 写:
- 0b: 无效
- 1b: I2C1 中断禁止
- **位[25] SPI0:** **SPI0 中断禁止**
- 读:
- 0b: SPI0 中断禁止
- 1b: SPI0 中断使能

- 写:
 - 0b: 无效
 - 1b: SPI0 中断禁止
- **位[26] SPI1: SPI1 中断禁止**
 - 读:
 - 0b: SPI1 中断禁止
 - 1b: SPI1 中断使能
 - 写:
 - 0b: 无效
 - 1b: SPI1 中断禁止
- **位[27] UART0: UART0 中断禁止**
 - 读:
 - 0b: UART0 中断禁止
 - 1b: UART0 中断使能
 - 写:
 - 0b: 无效
 - 1b: UART0 中断禁止
- **位[28] UART1: UART1 中断禁止**
 - 读:
 - 0b: UART1 中断禁止
 - 1b: UART1 中断使能
 - 写:
 - 0b: 无效
 - 1b: UART1 中断禁止
- **位[31:29] : 保留**

6.2.3 系统中断挂起设定寄存器 ISPR

(地址: 0xE000_E200)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	UART1	UART0	SPI1	SPI0	I2C1	I2C0	-	LDO_1p5V	LVD_5V	-	-	TIM2	TIM3
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	TIM4	TIM0	TIM1	ADC	-	COMP1	COMP0	-	PF	PB	PA	-	FLASH	-	CLOCK_FAIL	SYSDWG
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] SYSDWG: 系统看门狗中断挂起等待设定**
 - 读:

- 0b: 系统看门狗中断无请求
- 1b: 系统看门狗中断请求挂起等待
- 写:
- 0b: 无效
- 1b: 系统看门狗中断挂起等待
- **位[1] CLOCK_FAIL: 晶振时钟/倍频时钟失效中断挂起等待设定**
- 读:
- 0b: 晶振时钟/倍频时钟中断无请求
- 1b: 晶振时钟/倍频时钟中断请求挂起等待
- 写:
- 0b: 无效
- 1b: 晶振时钟/倍频时钟中断挂起等待
- **位[2] :** **保留**
- **位[3] FLASH: FLASH 控制中断挂起等待设定**
- 读:
- 0b: FLASH 中断无请求
- 1b: FLASH 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: FLASH 中断挂起等待
- **位[4] :** **保留**
- **位[5] PA: PA 端口中断挂起等待设定**
- 读:
- 0b: PA 端口中断无请求
- 1b: PA 端口中断请求挂起等待
- 写:
- 0b: 无效
- 1b: PA 端口中断挂起等待
- **位[6] PB: PB 端口中断挂起等待设定**
- 读:
- 0b: PB 端口中断无请求
- 1b: PB 端口中断请求挂起等待
- 写:
- 0b: 无效
- 1b: PB 端口中断挂起等待
- **位[7] PF: PF 端口中断挂起等待设定**
- 读:
- 0b: PF 端口中断无请求
- 1b: PF 端口中断请求挂起等待
- 写:
- 0b: 无效
- 1b: PF 端口中断挂起等待
- **位[8] :** **保留**
- **位[9] COMP0: 比较器 0 中断挂起等待设定**

- 读:
- 0b: 比较器 0 中断无请求
- 1b: 比较器 0 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: 比较器 0 中断挂起等待
- **位[10] COMP1: 比较器 1 中断挂起等待设定**
- 读:
- 0b: 比较器 1 中断无请求
- 1b: 比较器 1 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: 比较器 1 中断挂起等待
- **位[11] :** 保留
- **位[12] ADC: ADC 中断挂起等待设定**
- 读:
- 0b: ADC 中断无请求
- 1b: ADC 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: ADC 中断挂起等待
- **位[13] TIM1: TIMER1 中断挂起等待设定**
- 读:
- 0b: TIMER1 中断无请求
- 1b: TIMER1 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: TIMER1 中断挂起等待
- **位[14] TIM0: TIMER0 中断挂起等待设定**
- 读:
- 0b: TIMER0 中断无请求
- 1b: TIMER0 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: TIMER0 中断挂起等待
- **位[15] TIM4: TIMER4 中断挂起等待设定**
- 读:
- 0b: TIMER4 中断无请求
- 1b: TIMER4 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: TIMER4 中断挂起等待
- **位[16] TIM3: TIMER3 中断挂起等待设定**
- 读:

- 0b: TIMER3 中断无请求
- 1b: TIMER3 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: TIMER3 中断挂起等待
- **位[17] TIM2: TIMER2 中断挂起等待设定**
- 读:
- 0b: TIMER2 中断无请求
- 1b: TIMER2 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: TIMER2 中断挂起等待
- **位[18] : 保留**
- **位[19] : 保留**
- **位[20] LVD_5V: 电源低电压中断挂起等待设定**
- 读:
- 0b: 电源低电压中断无请求
- 1b: 电源低电压中断请求挂起等待
- 写:
- 0b: 无效
- 1b: 电源低电压中断挂起等待
- **位[21] LDO_1P5V: LDO_1P5V 低电压中断挂起等待设定**
- 读:
- 0b: LDO 低电压中断无请求
- 1b: LDO 低电压中断请求挂起等待
- 写:
- 0b: 无效
- 1b: LDO 低电压中断挂起等待
- **位[22] : 保留**
- **位[23] I2C0: I2C0 中断挂起等待设定**
- 读:
- 0b: I2C0 中断无请求
- 1b: I2C0 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: I2C0 中断挂起等待
- **位[24] I2C1: I2C1 中断挂起等待设定**
- 读:
- 0b: I2C1 中断无请求
- 1b: I2C1 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: I2C1 中断挂起等待
- **位[25] SPI0: SPI0 中断挂起等待设定**

- 读:
- 0b: SPI0 中断无请求
- 1b: SPI0 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: SPI0 中断挂起等待
- **位[26] SPI1: SPI1 中断挂起等待设定**
- 读:
- 0b: SPI1 中断无请求
- 1b: SPI1 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: SPI1 中断挂起等待
- **位[27] UART0: UART0 中断挂起等待设定**
- 读:
- 0b: UART0 中断无请求
- 1b: UART0 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: UART0 中断挂起等待
- **位[28] UART1: UART1 中断挂起等待设定**
- 读:
- 0b: UART1 中断无请求
- 1b: UART1 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: UART1 中断挂起等待
- **位[31:29] : 保留**

6.2.4 系统中断挂起清除寄存器 ICPR

(地址: 0xE000_E280)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	UART1	UART0	SPI1	SPI0	I2C1	I2C0	-	LDO_1p5v	LVD_5V	-	-	TIM2	TIM3
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	TIM4	TIM0	TIM1	ADC	-	COMP1	COMP0	-	PF	PB	PA	-	FLASH	-	CLOCK_FAIL	SYSWDG
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] SYSWDG: 系统看门狗中断挂起等待清除**
 读:
 0b: 系统看门狗中断无请求
 1b: 系统看门狗中断请求挂起等待
 写:
 0b: 无效
 1b: 系统看门狗中断挂起等待清除
- **位[1] CLOCK_FAIL: 晶振时钟/倍频时钟失效中断挂起等待清除**
 读:
 0b: 晶振时钟/倍频时钟中断无请求
 1b: 晶振时钟/倍频时钟中断请求挂起等待
 写:
 0b: 无效
 1b: 晶振时钟/倍频时钟中断挂起等待清除
- **位[2] : 保留**
- **位[3] FLASH: FLASH 控制中断挂起等待清除**
 读:
 0b: FLASH 中断无请求
 1b: FLASH 中断请求挂起等待
 写:
 0b: 无效
 1b: FLASH 中断挂起等待清除
- **位[4] : 保留**
- **位[5] PA: PA 端口中断挂起等待清除**
 读:
 0b: PA 端口中断无请求
 1b: PA 端口中断请求挂起等待
 写:
 0b: 无效

- 1b: PA 端口中断挂起等待清除
- **位[6] PB:** **PB 端口中断挂起等待清除**
- 读:
- 0b: PB 端口中断无请求
- 1b: PB 端口中断请求挂起等待
- 写:
- 0b: 无效
- 1b: PB 端口中断挂起等待清除
- **位[7] PF:** **PF 端口中断挂起等待清除**
- 读:
- 0b: PF 端口中断无请求
- 1b: PF 端口中断请求挂起等待
- 写:
- 0b: 无效
- 1b: PF 端口中断挂起等待清除
- **位[8] :** **保留**
- **位[9] COMP0:** **比较器 0 中断挂起等待清除**
- 读:
- 0b: 比较器 0 中断无请求
- 1b: 比较器 0 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: 比较器 0 中断挂起等待清除
- **位[10] COMP1:** **比较器 1 中断挂起等待清除**
- 读:
- 0b: 比较器 1 中断无请求
- 1b: 比较器 1 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: 比较器 1 中断挂起等待清除
- **位[11] :** **保留**
- **位[12] ADC:** **ADC 中断挂起等待清除**
- 读:
- 0b: ADC 中断无请求
- 1b: ADC 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: ADC 中断挂起等待清除
- **位[13] TIM1:** **TIMER1 中断挂起等待清除**
- 读:
- 0b: TIMER1 中断无请求
- 1b: TIMER1 中断请求挂起等待
- 写:
- 0b: 无效

- 1b: TIMER1 中断挂起等待清除
- **位[14] TIM0:** **TIMER0 中断挂起等待清除**
- 读:
- 0b: TIMER0 中断无请求
- 1b: TIMER0 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: TIMER0 中断挂起等待清除
- **位[15] TIM4:** **TIMER4 中断挂起等待清除**
- 读:
- 0b: TIMER4 中断无请求
- 1b: TIMER4 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: TIMER4 中断挂起等待清除
- **位[16] TIM3:** **TIMER3 中断挂起等待清除**
- 读:
- 0b: TIMER3 中断无请求
- 1b: TIMER3 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: TIMER3 中断挂起等待清除
- **位[17] TIM2:** **TIMER2 中断挂起等待清除**
- 读:
- 0b: TIMER2 中断无请求
- 1b: TIMER2 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: TIMER2 中断挂起等待清除
- **位[18] :** **保留**
- **位[19] :** **保留**
- **位[20] LVD_5V:** **电源低电压中断挂起等待清除**
- 读:
- 0b: 电源低电压中断无请求
- 1b: 电源低电压中断请求挂起等待
- 写:
- 0b: 无效
- 1b: 电源低电压中断挂起等待清除
- **位[21] LDO_1P5V:** **LDO_1P5V 低电压中断挂起等待清除**
- 读:
- 0b: LDO 低电压中断无请求
- 1b: LDO 低电压中断请求挂起等待
- 写:
- 0b: 无效

- 1b: LDO 低电压中断挂起等待清除
- 位[22] : 保留
- 位[23] I2C0: I2C0 中断挂起等待清除
- 读:
- 0b: I2C0 中断无请求
- 1b: I2C0 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: I2C0 中断挂起等待清除
- 位[24] I2C1: I2C1 中断挂起等待清除
- 读:
- 0b: I2C1 中断无请求
- 1b: I2C1 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: I2C1 中断挂起等待清除
- 位[25] SPI0: SPI0 中断挂起等待清除
- 读:
- 0b: SPI0 中断无请求
- 1b: SPI0 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: SPI0 中断挂起等待清除
- 位[26] SPI1: SPI1 中断挂起等待清除
- 读:
- 0b: SPI1 中断无请求
- 1b: SPI1 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: SPI1 中断挂起等待清除
- 位[27] UART0: UART0 中断挂起等待清除
- 读:
- 0b: UART0 中断无请求
- 1b: UART0 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: UART0 中断挂起等待清除
- 位[28] UART1: UART1 中断挂起等待清除
- 读:
- 0b: UART1 中断无请求
- 1b: UART1 中断请求挂起等待
- 写:
- 0b: 无效
- 1b: UART1 中断挂起等待清除

- 位[31:29] :

保留

澎湃微

7 时钟系统

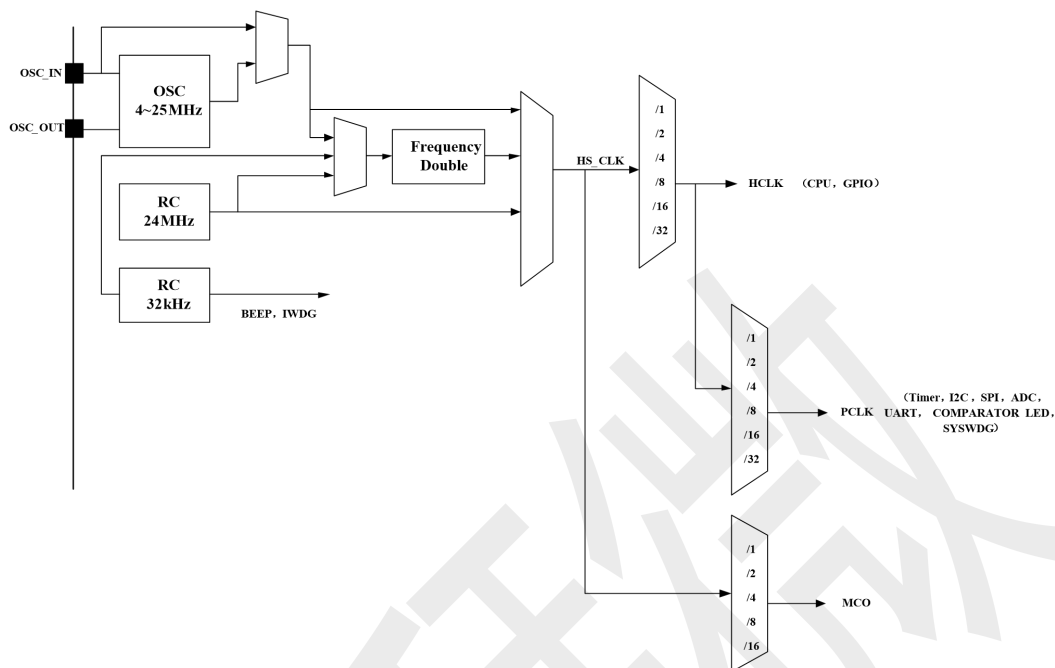


图 7-1 时钟结构图

PT32L031 有下面这些时钟：

1. 高速晶体振荡时钟，时钟频率 4MHz~25MHz。
2. 高速两倍频时钟，时钟频率 64kHz~50MHz。时钟源可以为内部低速 RC 振荡时钟（时钟频率为 32.768kHz），内部高速 RC 振荡时钟（时钟频率为 24MHz）或者高速晶体振荡时钟。
3. 内部低速 RC 振荡时钟，时钟频率 32.768kHz。
4. 内部高速 RC 振荡时钟，时钟频率 24MHz。
5. 外部输入高频时钟，时钟频率 4MHz~25MHz。

7.1 高速晶体振荡时钟

高速晶体振荡器，支持外接 4~25MHz 晶体，可以输出 4~25MHz 时钟，该时钟可以用作 MCU 系统主时钟。

注意：在启动高速晶体振荡器后，需要等待 50us(或者查看 `OSC_CON1` 的 RDY 位)，振荡器才稳定工作。也就是说，在启动高速晶体振荡器之后，需要等待 50us 之后(`OSC_CON1` 的 RDY 位为 1)，才能把主时钟切换到高速晶体振荡时钟。

高速晶体振荡时钟可以作为倍频时钟源，倍频时钟模块输出 4MHz~25MHz 的 2 倍频时钟。这个倍频时钟也可以作为 MCU 系统主时钟。

当 MCU 系统主时钟为高速晶体振荡时钟，高速晶体振荡时钟监测模块将被自动开启，实时检测是否发生晶体振荡时钟停止振荡的情况。

晶体引脚与 GPIO 复用，当不接外部高频晶体时，引脚可作为普通 GPIO 使用。

7.2 内部 24MHz RC 振荡时钟

内部 24MHz RC 振荡时钟，可以作为 MCU 系统主时钟。当外部晶体检测失效后，可以自动切换到该时钟工作。

当芯片进入深度睡眠模式时，MCU 的系统主时钟自动切换到内部 24MHz 时钟工作，芯片从深度睡眠模式唤醒后，系统主时钟仍由内部 24MHz 时钟驱动。当需要使用其他时钟作为系统主时钟时，用户需要重新修改时钟选择配置(配置 `SYSCLK_SEL` 的 `SYSCLK_SEL`)，选择相应的系统时钟，或者使能 `SYSCLK_SEL` 的 `SYSCLK_RECOVER` 为 1，系统将从深度睡眠模式唤醒之后，自动切换到进入睡眠之前的系统时钟工作。

7.3 内部 32kHz RC 振荡时钟

内部 32kHz RC 振荡时钟精度较低，可以作为 IWDG 的计数时钟。

7.4 外部输入高频时钟

当高速晶体振荡引脚不接晶体时，PF0 管脚可配置为 GPIO 直接输入外部高频时钟，时钟频率范围控制在 4MHz~25MHz，PF0 管脚可作 GPIO 脚使用。当使用外部高频时钟为系统主时钟时，需要配置 `OSC_CON1` 的 `ex_20m_clk_en` 为 1，然后再选择外部高频时钟为系统主时钟(配置 `SYSCLK_SEL` 的 `SYSCLK_SEL` 为 01)。

7.5 时钟切换的安全性

对主时钟进行切换时，需要注意切换的安全性。当主时钟切换到一个不工作的时钟时，很可能会导致系统死机。例如当高速振荡器不接晶体，而系统时钟又选择为高速晶体振荡时钟时，系统时钟不工作，系统会死机。此时如果时钟检测模块开启对高速晶体振荡时钟的检测，则系统时钟会自动切换到内部 24M RC 时钟，同时时钟检测模块会发出中断或者复位信号。

7.6 高速时钟监测

芯片内部有时钟监测模块检测高速晶振时钟是否停止振荡或者倍频时钟是否失效。时钟监测模块启动后，如果发生高速晶振时钟或者倍频时钟失效，可以产生中断或者复位，并且主时钟会自动切换到内部 24M RC 时钟。

高速晶振时钟监测模块将在系统主时钟设置为高速晶振时钟时自动启动，倍频时钟监测模块将在系统主时钟设置为倍频时钟时自动启动，软件可以配置当主时钟失效时，系统是否需要复位 (`RESET_CTRL` 寄存器的 `EOSC_FAIL_RST_EN` 位和 `FD_FAIL_RST_EN` 位)。在系统控制寄存器中的 `SYSCLK_SEL` 寄存器，可以查询时钟出现失效的状态信息。当时钟失效复位配置为无效，时钟停振中断配置为有效后，监测到高速晶振时钟或倍频时钟失效后，可以输出系统中断。当主时钟配置为高速晶振时钟或者倍频时钟，一旦发生时钟失效，主时钟会自动切换到内部 24M RC 时钟，确保系统仍能正常运行。

7.7 寄存器说明
7.7.1 系统时钟选择寄存器 SYSCLK_SEL

(地址: 0x4001_F00C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	SYSCLK_ RECOVER	-	-	WAKE_DELAY	
R/W	R	R	R	R	R	R	R	R	R	R	R	RW	R	R	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	FD_ FAIL	FD_ DETECT _EN	EOSC_ FAIL	EOSC_ DETECT _EN	-	PCLK_DIV			-	HCLK_DIV			FDCLK_SEL	SYSCLK_SEL		
R/W	RW	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[1:0] SYSCLK_SEL: 系统时钟选择**
 00b: 内部 RC 24MHz 时钟
 01b: 外部高速时钟 (高速晶振时钟/外部输入时钟)
 10b: 倍频时钟
- **位[3:2] FDCLK_SEL: 倍频时钟源选择**
 该位在倍频模块使能时不可写
 00b: 内部 RC 24MHz 时钟
 01b: 外部高速时钟 (高速晶振时钟/外部输入时钟)
 1xb: 内部 RC 32kHz 时钟
- **位[6:4] HCLK_DIV: HCLK 分频选择**
 000b: 不分频
 001b: 2 分频
 010b: 4 分频
 011b: 8 分频
 100b: 16 分频
 101b: 32 分频
 其他: 不分频
- **位[7] : 保留**

- **位[10:8] PCLK_DIV: PCLK 分频选择**
 000b: 不分频
 001b: 2分频
 010b: 4分频
 011b: 8分频
 100b: 16分频
 101b: 32分频
 其他: 不分频

- **位[11] :** 保留
- **位[12] EOSC_DETECT_EN: 外部高速时钟失效监测使能**
 该位在 SYSCLK_SEL 位选择外部高速时钟时被强制开启, 深度睡眠时将被关闭
 0b: 不使能
 1b: 使能
- **位[13] EOSC_FAIL: 外部高速时钟失效标志位**
 该位在 EOSC_DETECT_EN 为零时或时钟正常时自动清零
 0b: 外部高速时钟未发生失效
 1b: 外部高速时钟发生失效
- **位[14] FD_DETECT_EN: 倍频时钟失效监测使能**
 该位在 SYSCLK_SEL 位选择倍频时钟时被强制开启, 深度睡眠时将被关闭
 0b: 不使能
 1b: 使能
- **位[15] FD_FAIL: 倍频时钟失效标志位**
 该位在 FD_DETECT_EN 为零时或时钟正常时自动清零
 0b: 倍频时钟未发生失效
 1b: 倍频时钟发生失效
- **位[17:16] WAKE_DELAY: 唤醒延时选择**
 当 SYSCLK_RECOVER 为 1 时, 若进入深度睡眠之前的系统时钟为晶振时钟, 则该位只能配成 00。
 00b: 最大延时, 320 个 24MHz 时钟周期
 01b: 次大延时, 160 个 24MHz 时钟周期
 10b: 次小延时, 132 个 24MHz 时钟周期
 11b: 最小延时, 96 个 24MHz 时钟周期
- **位[19:18] :** 保留
- **位[20] SYSCLK_RECOVER: 系统时钟唤醒后自动恢复选择**
 0b: 系统时钟在深度睡眠唤醒后不自动切换成深度睡眠之前的时钟
 1b: 系统时钟在深度睡眠唤醒后自动切换成深度睡眠之前的时钟
- **位[31:21] :** 保留

8 低功耗模式

PT32L031 有如下 3 种工作模式，正常工作模式，休眠模式和深度休眠模式。其中休眠模式和深度休眠模式为低功耗模式。

8.1 低功耗模式的进入

使用 ARM Cortex-M0 的 Wait for Interrupt (WFI)和 wait for Event (WFE)两条指令可以使芯片进入休眠模式或深度睡眠模式。当执行 WFI 或 WFE 指令后，芯片进入哪种低功耗模式，由系统控制寄存器（SCR）的 SLEEPDEEP 位决定。请参考系统控制寄存器（SCR）的说明。

8.2 休眠模式

当芯片进入休眠模式后，CPU 时钟停止，CPU 模块停止工作，外设（系统）时钟仍维持工作，FLASH 进入 standby mode。进入休眠模式前，软件可以主动关闭任何不需要工作的数字外设和模拟模块，这样芯片的功耗会大幅降低。

8.3 深度睡眠模式

当芯片进入深度睡眠模式前，需要手动关闭部分模拟模块（如 ADC）和不需要的数字外设模块；当芯片进入深度睡眠模式后，内部模拟模块如 BandGap, Main LDO, RC_24MHz, Frequency_Double, LVD, Crystal 被强制关闭，CPU 时钟和系统主时钟停止工作，芯片的功耗非常微小。

除了被强制关闭的模拟模块，剩下的模拟模块由软件主动关闭，所有的 GPIO 不能浮空。注意不关闭的模拟模块会导致芯片在 SLEEP 模式下功耗变大。

8.4 低功耗模式的唤醒

芯片进入休眠模式后，可以通过以下方式唤醒：

1. 外部复位
2. 调试模式请求
3. 所有使能的中断源

芯片进入深度睡眠模式后，可以通过以下方式唤醒：

1. 外部复位
2. 外部中断（GPIO 的电平中断）
3. 内部中断（Timer4 中断，IWDG 中断，比较器中断）
4. 调试模式请求

当芯片从深度睡眠模式被唤醒后，内部 24MHz RC 振荡首先恢复工作，此时主时钟为 24MHz RC 振荡器时钟，之后等到电源稳定后（BandGap 和 Main LDO 电压稳定），CPU 和 FLASH 也开始恢复工作。

9 GPIO

PT32L031 有 3 组 GPIO，PA，PB，PF。其中：

- PA: PA0~PA15
- PB: PB0~PB7
- PF: PF0~PF1

总共有 26 个 GPIO 口。

9.1 特征

PT32L031 的 IO 口配置为 GPIO 时，具有如下特征：

- GPIO 口均可配置为输入或输出
- 所有的 GPIO 均可作为外部 IO 中断源，每组 GPIO 各有一个中断，软件可通过中断状态寄存器查看具体的引起中断的 IO 引脚
- 所有的 GPIO 在输出模式下上下拉自动禁止
- GPIO 在输入模式下引脚呈高阻抗
- GPIO 在输入模式下可选用内部上拉或下拉
- GPIO 提供开漏输出能力
- 通过位掩模提供 GPIO 位操作
- 通过分开的置位和清除操作控制寄存器，提供线程安全的端口操作
- 通过外设复用功能寄存器配置 IO 口为 GPIO 或者其他功能（如外设功能）
- 提供端口映射操作，可以只对端口某一位、或某几位进行操作。

9.2 输入输出方向控制

GPIO 的输入/输出方向控制由 Pn_OES, Pn_OEC 寄存器控制，Pn_OES 寄存器可以使能 IO 口输出使能，Pn_OEC 寄存器可以清除 IO 的输出使能，设置相应 IO 口为输入。具体控制方法见寄存器说明。

9.3 端口输入中断

所有 GPIO 端口均支持输入中断，软件通过四组寄存器，实现不同模式的输入中断响应。需要注意的是所有 GPIO 引脚在任何时候（包括设为端口输出模式或外设使能状态下）都具备引脚中断触发和响应功能。应用软件必须鉴别所需引脚的中断需求，对 Pn_IEC 寄存器相关位写 1（中断使能清零），以禁止无关的引脚产生中断。

控制外部输入端口中断的四对寄存器分别是 Pn_IES, Pn_IEC, Pn_ITS0, Pn_ITC0, Pn_ITS1, Pn_ITC1 和 Pn_PLS, Pn_PLC。其中 Pn_IES, Pn_IEC 控制端口中断的使能和禁止；Pn_ITS0, Pn_ITC0, Pn_ITS1, Pn_ITC1 控制端口中断的类型；Pn_PLS, Pn_PLC 控制端口中断的极性。

端口中断使能位	端口中断极性位	端口中断类型位0	端口中断类型位1	中断响应模式
0	-	-	-	中断禁止
1	0	0	0	低电平
1	0	1	0	下降沿
1	1	0	0	高电平
1	1	1	0	上升沿
1	X	X	1	电平变化

PORT1任一引脚所选择的中断被触发后，状态寄存器Pn_IF中的相应位被置1。所有引脚的中断请求都由同一个中断服务程序得到响应，用户软件在中断服务程序中需要查询Pn_IF和引脚状态确定具体的引脚中断源，并对Pn_IF的对应位写1来清除中断标志。

9.4 端口映射操作

通过映射操作可以实现针对同一端口的一位或多位进行置1或清0，避免了传统逐位进行端口设定时，顺序运行的程序代码被其它进程打断而引发的状态设置冲突。对于16位宽的一个端口，映射操作分低8位和高8位两组进行，分别映射至Pn_MASKL和Pn_MASKH。

实现原理如下：

例如，需要设端口Pn的位[1:0]为1，清位[7:6]为0，均位于低8位组，软件只需对Pn_MASKL数组区的某个单元直接进行一次16位的写操作即可完成所有工作（仅低8位有效，高8位可为任意值）。具体数组单元下标值由需要操作的数据位位置决定，在此例中需要操作的是位[1:0]和位[7:6]，所处位置按二进制描述为11000011，故数组地址单元确定为Pn_MASKL[0xC3]；赋值内容按需要设定的0或1决定，此例中需要操作的是位[1:0]=11，位[7:6]=00，按二进制描述为00xxxx11(x代表0/1任意)，最终，代码Pn_MASKL[0xC3]=0x03（高8位数据可任意）即一次实现了四个数据位同步置1和清零的操作。

针对端口高8位的操作也可以用完全相同的方式实现，映射数组区为Pn_MASKH，写入的数据必须在高8位[15:8]，低8位可为任意值：

例如，需要设端口Pn的位[8]为1，清位[13:12]为0，均位于高8位组，软件只需对Pn_MASKH数组区的某个单元直接进行一次16位的写操作即可完成所有工作（仅高8位有效，低8位可为任意值）。具体数组单元下标值由需要操作的数据位位置决定，在此例中需要操作的是位[8]和位[13:12]，只考虑高8位时按二进制描述为00110001，故数组地址单元确定为Pn_MASKH[0x31]；赋值内容按需要设定的0或1决定，此例中需要操作的是位[8]=1，位[13:12]=00，所处高8位的位置按二进制描述为xx00xxx1(x代表0/1任意)，最终，代码Pn_MASKH[0x31]=0x0100（低8位数据可任意）即一次实现了三个数据位同步置1和清零的操作。

9.5 端口输入内部上拉或下拉

当引脚被配置成端口输入模式时，每个引脚都可以独立控制启用片内的弱上拉或下拉电阻（等效阻值约为50K）。在引脚设为端口输出模式时，该上拉或下拉将被自动禁止。PnPUS可以使能端口上拉，PnPUC可以清除端口上拉，关闭上拉；PnPDS可以使能端口下拉，PnPDC可以清除端口下拉，关闭下拉。

9.6 端口输出驱动能力

在5V供电条件下，所有端口引脚均能支持20mA灌入、8mA拉出的电流驱动能力，但芯片整体的电流驱动不能超过最大极限标称值。

9.7 端口开漏输出功能

任一端口引脚在输出时均可设为开漏模式。此模式下芯片在低电平 0 输出时为低电平驱动；在高电平 1 输出时，芯片引脚为“漏极开路”状态，必须在片外通过电阻上拉到一个特定的电平电压后才能在引脚上得到一个真实的电压值。外部上拉电压不能超过芯片 $V_{dd}+0.6V$ 。通过 Pn_ODS, Pn_ODC 寄存器可以设置使能或清除 IO 口开漏功能。

9.8 端口施密特功能

CMOS 特性的端口引脚均具备施密特功能。

9.9 端口复用功能

IO 口通常默认功能为 GPIO (Debug 口除外)。通过对寄存器 Pn_AFS0/1, Pn_AFC 寄存器操作，可以开启或关闭 IO 口的外设复用功能，具体 IO 端口的外设功能复用详见管脚定义

每个 IO 口的复用功能由 2bit 或 3bit (PF 管脚的复用功能控制位为 2bit, PA 和 PB 管脚的复用功能控制位为 3bit, 详见 Pn_AFS0/1 寄存器说明) 控制，复用功能位的每个值分别对应不同的复用功能，如下：

- 3'b000 / 2'b00: 主功能
- 3'b001 / 2'b01: 复用功能 0
- 3'b010 / 2'b10: 复用功能 1
- 3'b011 / 2'b11: 复用功能 2
- 3'b100: 复用功能 3
- 3'b101: 复用功能 4
- 3'b110: 复用功能 5
- 3'b111: 复用功能 6

9.10 寄存器列表

地址	寄存器	描述	备注
0x4800_0000	PA_DAT	PA 端口数据寄存器	PA_DAT说明
0x4800_0004	PA_LAT	PA 端口数据输出锁存寄存器	PA_LAT说明
0x4800_0008	PA_OES	PA 端口输出使能置位寄存器	PA_OES说明
0x4800_000C	PA_OEC	PA 端口输出使能清除寄存器	PA_OEC说明
0x4800_0010	PA_AFS0	PA 外设复用置位寄存器 0	PA_AFS0说明
0x4800_0014	PA_AFC	PA 外设复用清除寄存器	PA_AFC说明
0x4800_0018	PA_IES	PA 中断使能置位寄存器	PA_IES说明
0x4800_001C	PA_IEC	PA 中断使能清除寄存器	PA_IEC说明
0x4800_0020	PA_ITS0	PA 中断类型置位寄存器 0	PA_ITS0说明
0x4800_0024	PA_ITC0	PA 中断类型清除寄存器 0	PA_ITC0说明
0x4800_0028	PA_ITS1	PA 中断类型置位寄存器 1	PA_ITS1说明
0x4800_002C	PA_ITC1	PA 中断类型清除寄存器 1	PA_ITC1说明
0x4800_0030	PA_PLS	PA 中断极性设置寄存器	PA_PLS说明
0x4800_0034	PA_PLC	PA 中断极性清除寄存器	PA_PLC说明
0x4800_0038	PA_IF	PA 中断标志寄存器	PA_IF说明
0x4800_003C	PA_PUS	PA 内部上拉使能置位寄存器	PA_PUS说明
0x4800_0040	PA_PUC	PA 内部上拉使能清除寄存器	PA_PUC说明
0x4800_0044	PA_PDS	PA 内部下拉使能置位寄存器	PA_PDS说明
0x4800_0048	PA_PDC	PA 内部下拉使能清除寄存器	PA_PDC说明
0x4800_004C	PA_ODS	PA 输出开漏使能置位寄存器	PA_ODS说明
0x4800_0050	PA_ODC	PA 输出开漏使能清除寄存器	PA_ODC说明
0x4800_0054	PA_ANAS	PA 模拟功能使能置位寄存器	PA_ANAS说明
0x4800_0058	PA_ANAC	PA 模拟功能使能清除寄存器	PA_ANAC说明
0x4800_005C	PA_AFS1	PA 外设复用置位寄存器 1	PA_AFS1说明
0x4800_0400 ~ 0x4800_07FC	PA_MASKL	PA 端口低 8 位映射操作区	PA_MASKL说明
0x4800_0800 ~ 0x4800_0BFC	PA_MASKH	PA 端口高 8 位映射操作区	PA_MASKH说明

地址	寄存器	描述	备注
0x4800_1000	PB_DAT	PB 端口数据寄存器	PB_DAT 说明
0x4800_1004	PB_LAT	PB 端口数据输出锁存寄存器	PB_LAT 说明
0x4800_1008	PB_OES	PB 端口输出使能置位寄存器	PB_OES 说明
0x4800_100C	PB_OEC	PB 端口输出使能清除寄存器	PB_OEC 说明
0x4800_1010	PB_AFS0	PB 外设复用置位寄存器 0	PB_AFS0 说明
0x4800_1014	PB_AFC0	PB 外设复用清除寄存器 0	PB_AFC 说明
0x4800_1018	PB_IES	PB 中断使能置位寄存器	PB_IES 说明
0x4800_101C	PB_IEC	PB 中断使能清除寄存器	PB_IEC 说明
0x4800_1020	PB_ITS0	PB 中断类型置位寄存器 0	PB_ITS0 说明
0x4800_1024	PB_ITC0	PB 中断类型清除寄存器 0	PB_ITC0 说明
0x4800_1028	PB_ITS1	PB 中断类型置位寄存器 1	PB_ITS1 说明
0x4800_102C	PB_ITC1	PB 中断类型清除寄存器 1	PB_ITC1 说明
0x4800_1030	PB_PLS	PB 中断极性设置寄存器	PB_PLS 说明
0x4800_1034	PB_PLC	PB 中断极性清除寄存器	PB_PLC 说明
0x4800_1038	PB_IF	PB 中断标志寄存器	PB_IF 说明
0x4800_103C	PB_PUS	PB 内部上拉使能置位寄存器	PB_PUS 说明
0x4800_1040	PB_PUC	PB 内部上拉使能清除寄存器	PB_PUC 说明
0x4800_1044	PB_PDS	PB 内部下拉使能置位寄存器	PB_PDS 说明
0x4800_1048	PB_PDC	PB 内部下拉使能清除寄存器	PB_PDC 说明
0x4800_104C	PB_ODS	PB 输出开漏使能置位寄存器	PB_ODS 说明
0x4800_1050	PB_ODC	PB 输出开漏使能清除寄存器	PB_ODC 说明
0x4800_1054	PB_ANAS	PB 模拟功能使能置位寄存器	PB_ANAS 说明
0x4800_1058	PB_ANAC	PB 模拟功能使能清除寄存器	PB_ANAC 说明
0x4800_1400 ~ 0x4800_17FC	PB_MASKL	PB 端口低 8 位映射操作区	PB_MASKL 说明
0x4800_1800 ~ 0x4800_1BFC	PB_MASKH	PB 端口高 8 位映射操作区	PB_MASKH 说明

地址	寄存器	描述	备注
0x4800_2000	PF_DAT	PF 端口数据寄存器	PF_DAT说明
0x4800_2004	PF_LAT	PF 端口数据输出锁存寄存器	PF_LAT说明
0x4800_2008	PF_OES	PF 端口输出使能置位寄存器	PF_OES说明
0x4800_200C	PF_OEC	PF 端口输出使能清除寄存器	PF_OEC说明
0x4800_2010	PF_AFS0	PF 外设复用置位寄存器 0	PF_AFS0说明
0x4800_2014	PF_AFC0	PF 外设复用清除寄存器 0	PF_AFC说明
0x4800_2018	PF_IES	PF 中断使能置位寄存器	PF_IES说明
0x4800_201C	PF_IEC	PF 中断使能清除寄存器	PF_IEC说明
0x4800_2020	PF_ITS0	PF 中断类型置位寄存器 0	PF_ITS0说明
0x4800_2024	PF_ITC0	PF 中断类型清除寄存器 0	PF_ITC0说明
0x4800_2028	PF_ITS1	PF 中断类型置位寄存器 1	PF_ITS1说明
0x4800_202C	PF_ITC1	PF 中断类型清除寄存器 1	PF_ITC1说明
0x4800_2030	PF_PLS	PF 中断极性设置寄存器	PF_PLS说明
0x4800_2034	PF_PLC	PF 中断极性清除寄存器	PF_PLC说明
0x4800_2038	PF_IF	PF 中断标志寄存器	PF_IF说明
0x4800_203C	PF_PUS	PF 内部上拉使能置位寄存器	PF_PUS说明
0x4800_2040	PF_PUC	PF 内部上拉使能清除寄存器	PF_PUC说明
0x4800_2044	PF_PDS	PF 内部下拉使能置位寄存器	PF_PDS说明
0x4800_2048	PF_PDC	PF 内部下拉使能清除寄存器	PF_PDC说明
0x4800_204C	PF_ODS	PF 输出开漏使能置位寄存器	PF_ODS说明
0x4800_2050	PF_ODC	PF 输出开漏使能清除寄存器	PF_ODC说明
0x4800_2400 ~ 0x4800_27FC	PF_MASKL	PF 端口低 8 位映射操作区	PF_MASKL说明
0x4800_2800 ~ 0x4800_2BFC	PF_MASKH	PF 端口高 8 位映射操作区	PF_MASKH说明

9.11 寄存器描述
9.11.1 端口数据寄存器 Pn_DAT

(地址: PA: 0x4800_0000; PB: 0x4800_1000; PF: 0x4800_2000)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **端口数据位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 任何时候读 PTn_DAT 时, 读回的是端口引脚的实际电平逻辑状态;
 任何时候写 PTn_DAT 时, 实际写入的是端口的数据输出锁存寄存器 PTn_LAT, 而不是 PTn_DAT 寄存器
 读寄存器时:
 0b: 端口引脚为逻辑低电平
 1b: 端口引脚为逻辑高电平

 写寄存器且对应引脚输出使能时:
 0b: 端口引脚输出逻辑低电平
 1b: 端口引脚输出逻辑高电平
- **位[31:16] :** **保留**

9.11.2 端口数据输出锁存寄存器 Pn_LAT

(地址: PA: 0x4800_0004; PB: 0x4800_1004; PF: 0x4800_2004)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **端口数据位**

PA 的有效位为 bit[15:0]

PB 的有效位为 bit[7:0]

PF 的有效位为 bit[1:0]

当需要在端口引脚上输出逻辑电平时，输出值将锁存在此寄存器中并保持，但引脚上是否出现对应的高低逻辑电平，则还取决于对应引脚的输出使能设定，以及相关外围硬件电路设计

读寄存器时，读回此寄存器的锁定值，而不是引脚逻辑状态：

0b: 端口输出锁定为逻辑低电平

1b: 端口输出锁定为逻辑高电平

写寄存器时，将数据锁存在寄存器中：

0b: 端口输出锁存为逻辑低电平

1b: 端口输出锁存为逻辑高电平

- **位[31:16] :** **保留**

9.11.3 端口输出使能置位寄存器 Pn_OES

(地址: PA: 0x4800_0008; PB: 0x4800_1008; PF: 0x4800_2008)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **端口输出使能设定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 若引脚未开启任何复用功能时, 要使引脚能够输出高低逻辑电平, 此寄存器中的对应位必须写 1

 读寄存器时, 读回引脚的输入输出方向设置:
 0b: 端口引脚为输入状态
 1b: 端口引脚为输出状态
 写寄存器时, 引脚输出使能设定:
 0b: 无效操作
 1b: 端口引脚输出使能设定, 引脚为输出模式
- **位[31:16] :** **保留**

9.11.4 端口输出使能清除寄存器 Pn_OEC

(地址: PA: 0x4800_000C; PB: 0x4800_100C; PF: 0x4800_200C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚输出禁止设定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 若引脚未开启任何复用功能时, 要使引脚能够为输入模式, 此寄存器中的对应位必须写 1

 读寄存器时, 读回值始终为 0
 写寄存器时, 引脚输出禁止设定:
 0b: 无效操作
 1b: 端口引脚输出使能禁止, 引脚为输入模式
- **位[31:16] :** **保留**

9.11.5 外设复用置位寄存器 0 Pn_AFS0

(地址: PA: 0x4800_0010; PB: 0x4800_1010; PF: 0x4800_2010)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	Pn7			-	Pn6			-	Pn5			-	Pn4		
R/W	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	Pn3			-	Pn2			-	Pn1			-	Pn0		
R/W	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[31:0] DATA:** 引脚外设复用设定位
 - PA 的有效位为 bit[31:0]**
 - PB 的有效位为 bit[31:0]**
 - PF 的有效位为 bit[7:0]且每个引脚仅两位复用功能位可写, 例如 PF0 的复用功能设置位为 bit[1:0], PF1 的复用功能设置位为 bit[5:4]**
 - 若引脚需要复用外设功能时, 此寄存器中的对应位必须写 1, 具体外设复用列表详见管脚定义
 - 读寄存器时, 读回引脚的外设功能设置:
 - 0b: 端口引脚对应外设功能位设置值为 1
 - 1b: 端口引脚对应外设功能位设置值为 0
 - 写寄存器时, 引脚对应外设功能位使能设定:
 - 0b: 端口引脚对应外设功能位设置为 0
 - 1b: 端口引脚对应外设功能位设置为 1

9.11.6 外设复用清除寄存器 0 Pn_AFC

(地址: PA: 0x4800_0014; PB: 0x4800_1014; PF: 0x4800_2014)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚外设复用清除位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 该寄存器中的一位对应一个 IO 口，如对于 PA 端口，D0 对应控制 PA0 的复用功能清除，D1 对应控制 PA1 的复用功能清除。当要清除 PA0 的引脚复用功能时，往 D0 写 1 即可，使其恢复 GPIO 功能
 读寄存器时，读回值始终为 0
 写寄存器时，引脚外设功能设置位清除，恢复为 GPIO 功能：
 0b: 无效操作
 1b: 端口引脚外设复用设定清除为 0
- **位[31:16] :** **保留**

9.11.7 中断使能置位寄存器 Pn_IES

(地址: PA: 0x4800_0018; PB: 0x4800_1018; PF: 0x4800_2018)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚输入中断使能设定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 若使引脚对输入状态产生中断响应, 此寄存器中的对应位必须写 1

读寄存器时, 读回引脚中断设定状态:

0b: 端口引脚输入中断禁止

1b: 端口引脚输入中断使能

写寄存器时, 使能引脚的输入中断功能:

0b: 无效操作

1b: 端口引脚输入中断功能使能

- **位[31:16] :** **保留**

9.11.8 中断使能清除寄存器 Pn_IEC

(地址： PA: 0x4800_001C; PB: 0x4800_101C; PF: 0x4800_201C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** 引脚输入中断禁止设定位
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 若使引脚对输入状态不产生中断响应，， 设定 Pn_IEC 寄存器相关位以禁止无关的引脚中断

 读寄存器时， 读回值始终为 0
 写寄存器时， 使能引脚的输入中断功能：
 0b: 无效操作
 1b: 端口引脚输入中断功能禁止
- **位[31:16] :** 保留

9.11.9 中断类型置位寄存器 0 Pn_ITS0

(地址: PA: 0x4800_0020; PB: 0x4800_1020; PF: 0x4800_2020)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚中断类别 0 设置定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 读寄存器时, 读回引脚中断类别状态:
 0b: 端口引脚输入电平中断
 1b: 端口引脚输入沿跳变中断
 写寄存器时, 设定引脚输入的中断类别:
 0b: 无效操作
 1b: 端口引脚输入中断类别为沿跳变中断
- **位[31:16] :** **保留**

9.11.10 中断类型清除寄存器 0 Pn_ITC0

(地址: PA: 0x4800_0024; PB: 0x4800_1024; PF: 0x4800_2024)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚中断类别 0 清除设定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 读寄存器时, 读回值始终为 0
 写寄存器时, 设定引脚输入的中断类别:
 0b: 无效操作
 1b: 端口引脚输入中断类别为电平中断
- **位[31:16] :** **保留**

9.11.11 中断类型置位寄存器 1 Pn_ITS1

(地址: PA: 0x4800_0028; PB: 0x4800_1028; PF: 0x4800_2028)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚中断类别 1 设置定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 读寄存器时, 读回引脚中断类别状态:
 0b: 端口引脚输入电平中断或沿跳变中断
 1b: 端口引脚输入任意电平中断
 写寄存器时, 设定引脚输入的中断类别:
 0b: 无效操作
 1b: 端口引脚输入中断类别为任意电平中断
- **位[31:16] :** **保留**

9.11.12 中断类型清除寄存器 1 Pn_ITC1

(地址: PA: 0x4800_002C; PB: 0x4800_102C; PF: 0x4800_202C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚中断类别 1 清除设定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 读寄存器时, 读回值始终为 0
 写寄存器时, 设定引脚输入的中断类别:
 0b: 无效操作
 1b: 端口引脚输入电平中断或沿跳变中断
- **位[31:16] :** **保留**

9.11.13 中断极性设置寄存器 Pn_PLS

(地址: PA: 0x4800_0030; PB: 0x4800_1030; PF: 0x4800_2030)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚中断极性设置定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 读寄存器时, 读回引脚中断极性状态:
 0b: 端口引脚输入中断为低电平或下降沿有效
 1b: 端口引脚输入中断为高电平或上升沿有效
 写寄存器时, 设定引脚输入的中断极性:
 0b: 无效操作
 1b: 端口引脚输入中断极性为高电平或上升沿有效
- **位[31:16] :** **保留**

9.11.14 中断极性清除寄存器 Pn_PLC

(地址: PA: 0x4800_0034; PB: 0x4800_1034; PF: 0x4800_2034)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚中断极性清除设定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 读寄存器时, 读回值始终为 0
 写寄存器时, 设定引脚输入的中断极性:
 0b: 无效操作
 1b: 端口引脚输入中断极性为低电平或下降沿有效
- **位[31:16] :** **保留**

9.11.15 中断标志寄存器 Pn_IF

(地址: PA: 0x4800_0038; PB: 0x4800_1038; PF: 0x4800_2038)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚中断标志位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 读寄存器时, 读回引脚的输入中断标志:
 0b: 端口引脚输入无中断发生
 1b: 端口引脚输入有中断发生
 写寄存器时, 清除引脚的输入中断标志:
 0b: 无效操作
 1b: 端口引脚的输入中断标志清除
- **位[31:16] :** **保留**

9.11.16 内部上拉使能置位寄存器 Pn_PUS

(地址: PA: 0x4800_003C; PB: 0x4800_103C; PF: 0x4800_203C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚上拉使能设定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 若引脚在输入模式下使能片内上拉电阻，此寄存器中的对应位必须写 1

 读寄存器时，读回引脚输入上拉设置：
 0b: 端口引脚输入上拉功能禁止
 1b: 端口引脚输入上拉功能使能
 写寄存器时，引脚输入上拉使能设定：
 0b: 无效操作
 1b: 端口引脚输入上拉使能
- **位[31:16] :** **保留**

9.11.17 内部上拉使能清除寄存器 Pn_PUC

(地址: PA: 0x4800_0040; PB: 0x4800_1040; PF: 0x4800_2040)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚上拉禁止设定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 若禁止引脚在输入模式下使能片内上拉电阻, 此寄存器中的对应位必须写 1

 读寄存器时, 读回值始终为 0
 写寄存器时, 引脚输入上拉禁止设定:
 0b: 无效操作
 1b: 端口引脚输入上拉禁止
- **位[31:16] :** **保留**

9.11.18 内部下拉使能置位寄存器 Pn_PDS

(地址: PA: 0x4800_0044; PB: 0x4800_1044; PF: 0x4800_2044)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚下拉使能设定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 若引脚在输入模式下使能片内下拉电阻，此寄存器中的对应位必须写 1

 读寄存器时，读回引脚输入下拉设置：
 0b: 端口引脚输入下拉功能禁止
 1b: 端口引脚输入下拉功能使能
 写寄存器时，引脚输入下拉使能设定：
 0b: 无效操作
 1b: 端口引脚输入下拉使能
- **位[31:16] :** **保留**

9.11.19 内部下拉使能清除寄存器 Pn_PDC

(地址: PA: 0x4800_0048; PB: 0x4800_1048; PF: 0x4800_2048)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** 引脚下拉禁止设定位
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 若禁止引脚在输入模式下使能片内下拉电阻, 此寄存器中的对应位必须写 1

 读寄存器时, 读回值始终为 0
 写寄存器时, 引脚输入下拉禁止设定:
 0b: 无效操作
 1b: 端口引脚输入下拉禁止
- **位[31:16] :** 保留

9.11.20 输出开漏使能置位寄存器 Pn_ODS

(地址: PA: 0x4800_004C; PB: 0x4800_104C; PF: 0x4800_204C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **端口输出开漏使能设定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 若使引脚输出开漏被使能，此寄存器中的对应位必须写 1

读寄存器时，读回引脚输出开漏设置：

0b: 端口引脚输出开漏功能禁止

1b: 端口引脚输出开漏功能使能

写寄存器时，引脚输出开漏使能设定：

0b: 无效操作

1b: 端口引脚输出开漏使能

- **位[31:16] :** **保留**

9.11.21 输出开漏使能清除寄存器 Pn_ODC

(地址: PA: 0x4800_0050; PB: 0x4800_1050; PF: 0x4800_2050)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** 引脚输出开漏禁止设定位
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 若使引脚输出开漏被禁止，此寄存器中的对应位必须写 1

 读寄存器时，读回值始终为 0
 写寄存器时，引脚输出开漏禁止设定：
 0b: 无效操作
 1b: 端口引脚输出开漏禁止
- **位[31:16] :** 保留

9.11.22 模拟功能使能置位寄存器 Pn_ANAS

(地址: PA: 0x4800_0054; PB: 0x4800_1054; PF: 0x4800_2054)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **端口模拟功能使能设定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 若使引脚为模拟管脚时，此寄存器中的对应位必须写 1

读寄存器时，读回引脚的模拟功能使能设置：

0b: 端口引脚为数字 IO

1b: 端口引脚为模拟 IO

写寄存器时，引脚模拟功能使能设定：

0b: 无效操作

1b: 端口模拟功能使能设定，引脚为模拟 IO

- **位[31:16] :** **保留**

9.11.23 模拟功能使能清除寄存器 Pn_ANAC

(地址: PA: 0x4800_0058; PB: 0x4800_1058; PF: 0x4800_2058)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] DATA:** **引脚模拟功能禁止设定位**
PA 的有效位为 bit[15:0]
PB 的有效位为 bit[7:0]
PF 的有效位为 bit[1:0]
 若使引脚为数字 IO, 此寄存器中的对应位必须写 1

 读寄存器时, 读回值始终为 0
 写寄存器时, 引脚模拟功能禁止设定:
 0b: 无效操作
 1b: 端口引脚模拟功能使能禁止, 引脚为数字 IO
- **位[31:16] :** **保留**

9.11.24 外设复用置位寄存器 1 PA_AFS1

(地址: PA: 0x4800_005C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	PA15			-	PA14			-	PA13			-	PA12		
R/W	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	PA11			-	PA10			-	PA9			-	PA8		
R/W	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW	R	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[31:0] DATA: 引脚外设复用设定位

若引脚需要复用外设功能时，此寄存器中的对应位必须写 1，具体外设复用列表详见管脚定义

读寄存器时，读回引脚的外设功能设置：

0b: 端口引脚对应外设功能位设置值为 1

1b: 端口引脚对应外设功能位设置值为 0

写寄存器时，引脚对应外设功能位使能设定：

0b: 端口引脚对应外设功能位设置为 0

1b: 端口引脚对应外设功能位设置为 1

9.11.25 端口低 8 位映射操作区域 Pn_MASKL

Pn_MASKL 为一块长度为 256 字的虚拟内存区,用以映射针对端口 Pn 低 8 位中任意多位的一次性置 1 和清 0 组合操作 (线程安全操作)

n = A, 针对 PA 端口模块; n = B, 针对 PB 端口模块; n = F, 针对 PF 端口模块;

该区域只写, 读回数据为无效. 映射操作的法则请参阅[端口映射操作章节](#)

9.11.26 端口高 8 位映射操作区域 Pn_MASKH

Pn_MASKH 为一块长度为 256 字的虚拟内存区,用以映射针对端口 Pn 高 8 位中任意多位的一次性置 1 和清 0 组合操作 (线程安全操作)

n = A, 针对 PA 端口模块; n = B, 针对 PB 端口模块; n = F, 针对 PF 端口模块;

该区域只写, 读回数据为无效. 映射操作的法则请参阅[端口映射操作章节](#)

10 系统定时器

10.1 综述

系统定时器，Systick，是芯片 CPU Cortex-M0 自带的一个模块，它主要用于计时。

系统定时器提供了一个简单易用的 24 位循环递减的计数器，当系统定时器使能时，计数器开始工作。当计数器递减到 0 时，会向矢量中断控制器发起中断请求，申请获得处理器响应并处理系统定时器的事务。

10.2 寄存器列表

地址	寄存器	描述	备注
0xE000_E010	CORET_CSR	控制状态寄存器	CORET_CSR 说明
0xE000_E014	CORET_RVR	回填值寄存器	CORET_RVR 说明
0xE000_E018	CORET_CVR	当前值寄存器	CORET_CVR 说明

10.3 寄存器描述
10.3.1 控制状态寄存器 CORET_CSR

(地址: 0xE000_E010)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	ZF
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	IE	EN
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

- 位[0] EN: **系统定时器使能控制**
 0b: 系统定时器禁止
 1b: 系统定时器使能
- 位[1] IE: **系统定时器中断控制**
写 CORET_CVR 寄存器会使计数归零, 但并不能触发中断
 0b: 禁止系统定时器计数归零中断
 1b: 允许系统定时器计数归零中断
- 位[15:2] **保留**
- 位[16] ZF: **系统定时器计数归零标志**
读 CORET_CSR 寄存器以及写 CORET_CVR 寄存器均会清零 ZF
 0b: 自上一次读取后系统定时器计数尚未归零
 1b: 自上一次读取后系统定时器计数发生归零
- 位[31:17] **保留**

10.3.2 回填值寄存器 CORET_RVR

(地址: 0xE000_E018)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	RELOAD							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	RELOAD															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

- **位[23:0] RELOAD:** **系统定时器计数重载值设定**
 RELOAD 取值范围可为 0x00_0001-0xFF_FFFF 之间的任意值。如果设为 0 则系统定时器将停止工作。
 如果需要系统时标的中断间隔为 N 个时钟周期, 则 RELOAD=N-1。例如需要每 100 个时钟周期产生中断, 则 RELOAD=99
- **位[31:24]** **保留**

10.3.3 当前值寄存器 CORET_CVR

(地址: 0xE000_E018)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	CURRENT							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	CURRENT															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

- **位[23:0] CURRENT:** **系统定时器当前计数值**
 系统定时器当前计数值。
- **位[31:24]** **保留**

11 高级定时器

11.1 综述

PT32L031 内设置有 2 个 16 位宽的高级定时器，TMR0、TMR1，带 8 位宽的预分频，支持递增/递减/递增递减交替计数模式，各支持 4 路捕捉输入和 4 路单独 PWM 输出功能，支持刹车输入。

11.2 功能框图

定时器 TMR0，TMR1 均为 16 位宽通用定时器，支持递增，递减或者递增递减交替计数模式。

定时器 TMR0，TMR1 配有四路外部引脚信号的沿跳变的时间捕捉和一路比较定时的功能，可利用同一个定时器资源，灵活实现外部信号变化时间点的精确捕捉和特定的软件定时控制。

定时器 TMR0，TMR1 可产生比较定时中断，捕捉中断和刹车输入中断，但所有中断信号将使用同一个信号输出端口，应用软件需通过 TMR1_IR（TMR0_IR）寄存器查询标志位以判断产生中断的具体事件。

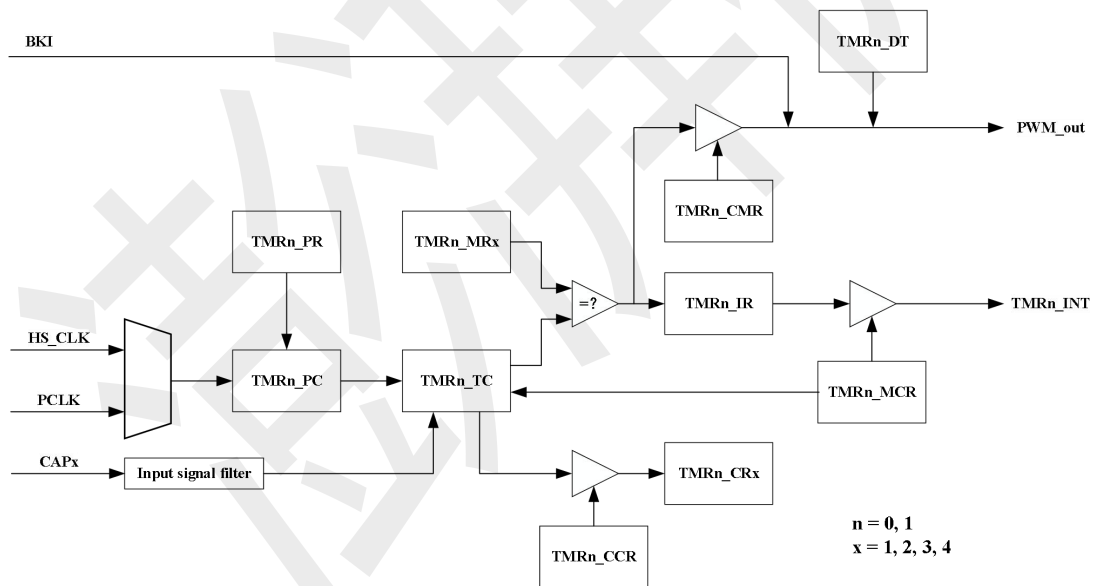


图 11-1. 定时器 TMR0，TMR1 工作原理图

图 11-1 显示了高级定时器的工作原理。定时器所有寄存器的读写均在 PCLK 下可通过软件进行读写操作，定时器时钟可以为 HS_CLK 或者 PCLK，TMRn_PC 寄存器根据 TMRn_PR 寄存器内设置的分频系数对定时器时钟进行计数分频，TMRn_TC 寄存器在通过 TMRn_PC 寄存器分频后产生的时钟下计数。

TMRn_TC 一共有三种计数方式，递增计数，递减计数和递增递减交替计数。

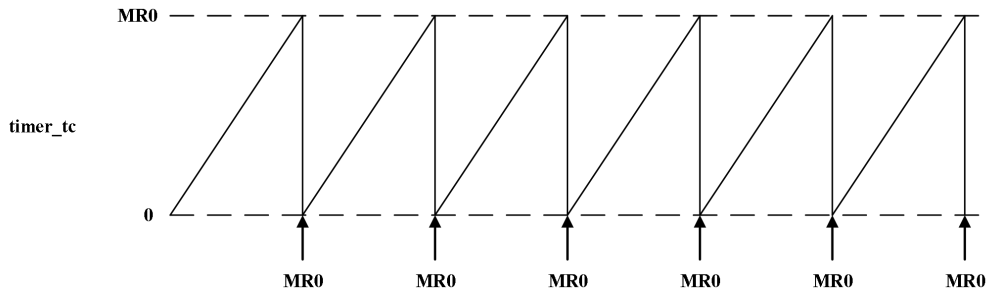


图 11-2 递增计数示意图

递增计数模式下，当 TMRn_TC 与 TMRn_MR0 寄存器内的值相同时，TMRn_IR 会产生相应的比较定时上溢出信号（TMRn_IR 寄存器的 MR0 置 1），同时定时器会复位 TMRn_TC 寄存器为 0 重新开始计数，并且根据 TMRn_MCR 寄存器内的参数设置产生相应的输出信号（输出比较定时中断信号或停止定时器）。递增计数的示意图如图 11-2 所示。

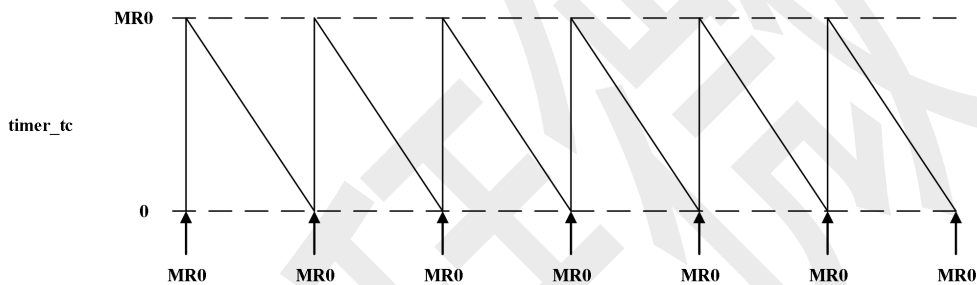


图 11-3 递减计数示意图

递减计数模式下，当 TMRn_TC 为 0 时，TMRn_IR 会产生相应的比较定时下溢出信号（TMRn_IR 寄存器的 MR0 置 1），同时定时器会复位 TMRn_TC 寄存器为 TMRn_MR0 重新开始计数，并且根据 TMRn_MCR 寄存器内的参数设置产生相应的输出信号（输出比较定时中断信号或停止定时器）。递减计数的示意图如图 11-3 所示。

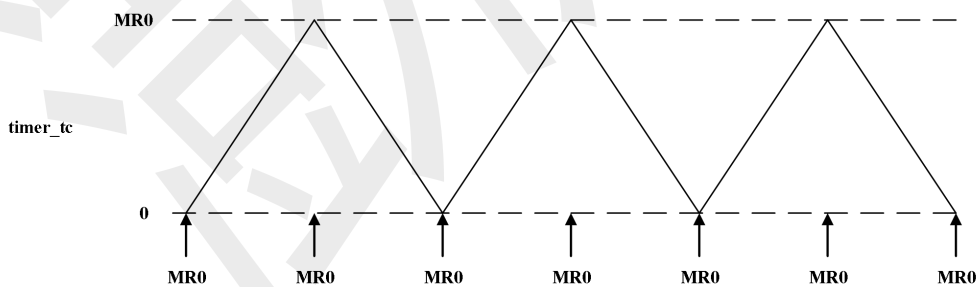


图 11-4 交替计数示意图

递增递减交替计数模式下，当 TMRn_TC 为 0 或者与 TMRn_MR0 寄存器内的值相同时，TMRn_IR 会产生相应的比较定时上溢出或者下溢出信号（TMRn_IR 寄存器的 MR0 置 1），同时定时器会根据当前的计数方向复位 TMRn_TC 寄存器为 0 或者为 TMRn_MR0 重新开始计数，并且根据 TMRn_MCR 寄存器内的参数设置产生相应的输出信号（输出比较定时中断信号，复位 TMRn_TC 寄存器或停止定时器）。交替计数的示意图如图 11-4 所示。

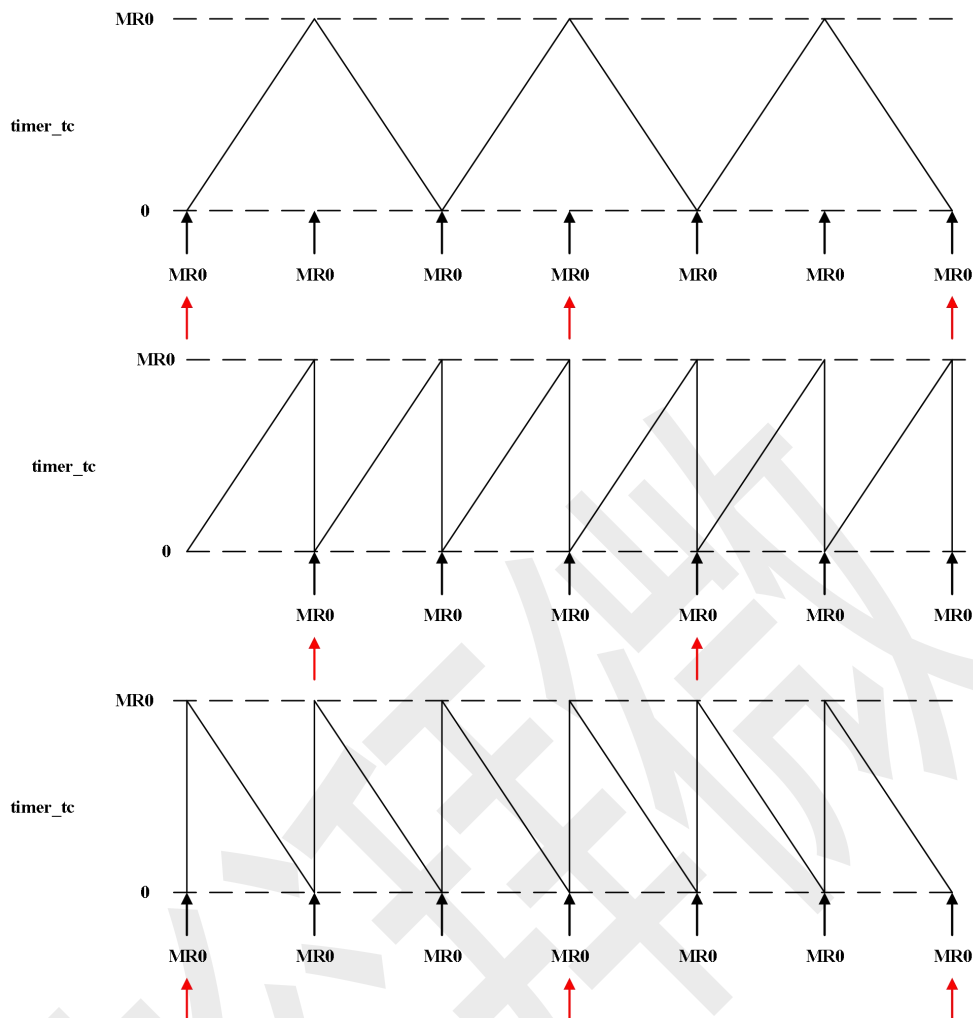


图 11-5 累计溢出输出中断示意图 (INT_RPT = 2)

定时器的上溢出和下溢出的标志位在每次定时器溢出时都会对应的置 1，但是若软件使能了中断输出，则可以通过 TMRn_INT_RPT 来控制累计若干次溢出后才输出中断，如图 11-5 所示。图 11-5 中显示了当 TMRn_INT_RPT 为 2 时，则每隔三次溢出，定时器才会输出一次中断（红色箭头）。

高级定时器同时还支持 4 路捕捉功能和 4 路 PWM 输出功能，同一时刻，管脚只能配置为通道输入或者 PWM 输出中的一种。

当定时器的管脚被配置为捕捉输入时，定时器对捕捉通道信号 CAPx 进行同步滤波处理后，根据 TMRn_CCR 寄存器的设置，对 CAPx 的上升沿，下降沿跳变进行捕获。当任意一个捕捉通道信号出现 TMRn_CCR 寄存器设定的电平沿(上升沿或下降沿)跳变时，TMRn_IR 寄存器相关标志位会被更新 (CRn_R/F 置 1)。同时，出现所设定的电平沿跳变时所对应的 TMRn_TC 寄存器的值会被存入对应通道的 TMRn_CRn 寄存器，同时根据 TMRn_CCR 寄存器内的参数设置产生输出信号（输出捕捉中断信号）并且决定 TMRn_TC 是否复位。

当定时器的管脚被配置为 PWM 输出时，当 TMRn_TC 与任意一个 TMRn_MR_x (x = 1,2,3,4) 寄存器内的值相同时，PWM 输出会根据 TMRn_CMR 寄存器内的 PWM 模式设置，输出对应的 PWM 波形，同时 PWM 输出还支持刹车输入和互补输出以及死区时间插入，详见 PWM 输出章节描述。

11.3 定时器匹配值更新

定时器匹配值在定时器未使能时，是立即更新的，软件无需写 update 位，如图 10-6 所示，

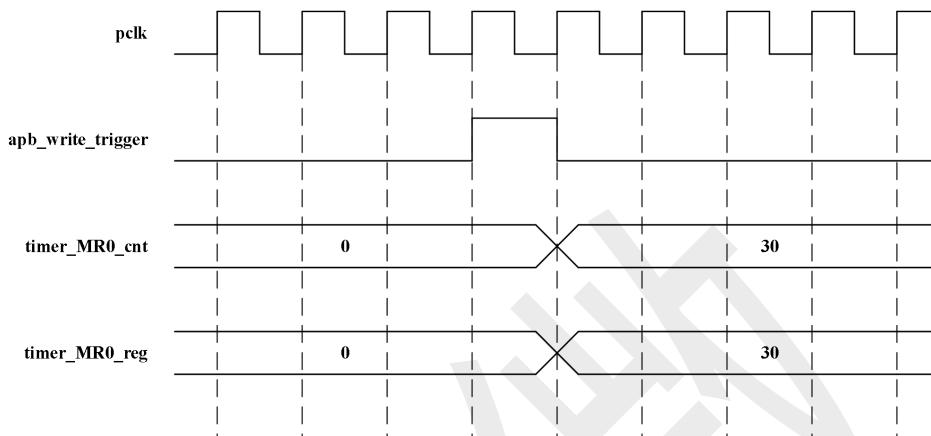


图 11-6 定时器未使能时匹配值更新

timer_MR_x_reg 为软件可写寄存器，timer_MR_x_cnt 为定时器内部对 timer_MR_x_reg 寄存器的映射。

定时器匹配值在定时器使能后，有两种更新方式，一个是等待当前周期结束后才更新，一个是通过写软件 update 位立即更新。

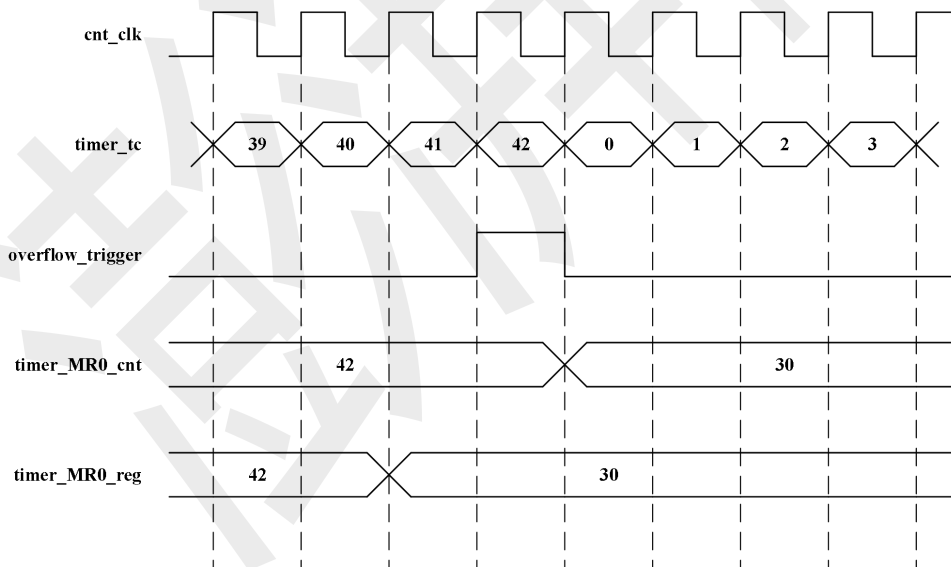


图 11-7 递增计数时自动更新匹配值

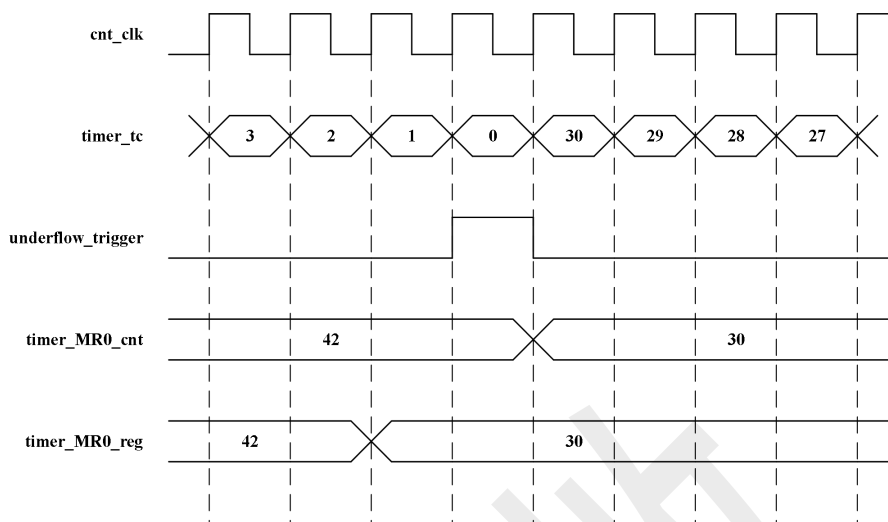


图 11-8 递减计数时自动更新匹配值

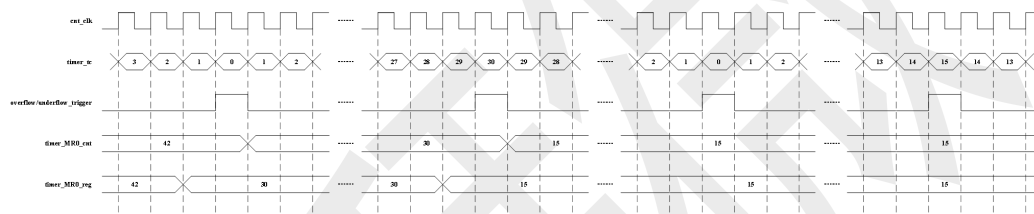


图 11-9 交替计数时自动更新匹配值

图 11-7，图 11-8，图 11-9 显示了不同计数方式下，定时器匹配值自动更新的时序。当软件写 timer_MR_x_reg 后，timer_MR_x_cnt 并不会立刻更新匹配值，而是会等待当前周期结束后才会更新匹配值，以保证周期的完整性。

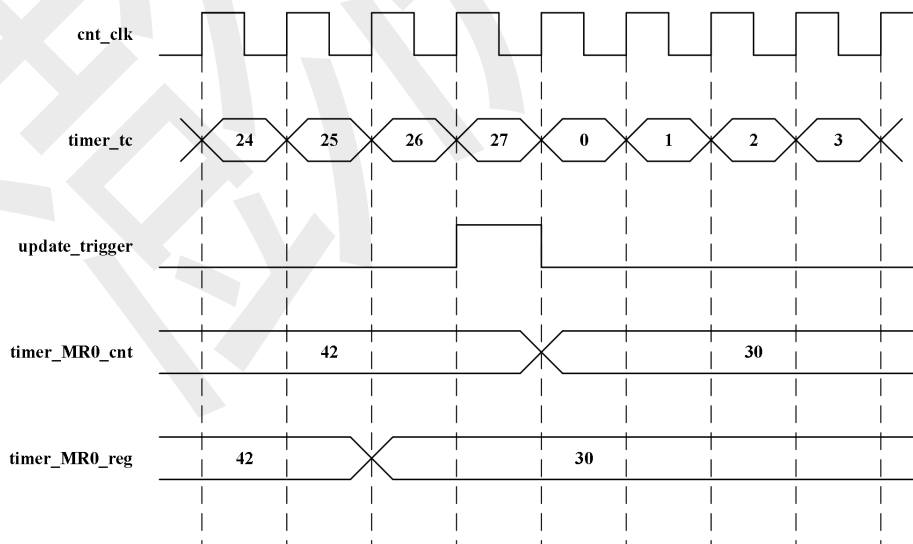


图 11-10 递增计数时手动更新匹配值

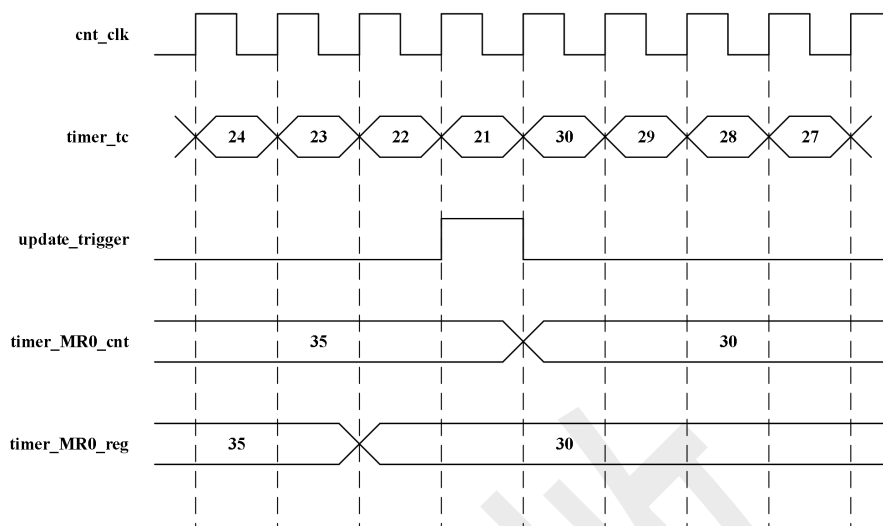


图 11-11 递减计数时手动更新匹配值

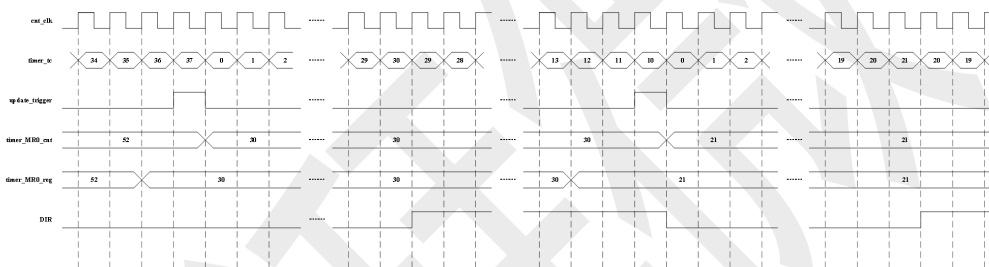


图 11-12 交替计数时手动更新匹配值

图 11-10，图 11-11，图 11-12 显示了不同计数方式下，定时器匹配值手动更新的时序。当软件写 timer_MR_x_reg 后，不仅可以等当前周期结束后 timer_MR_x_cnt 自动更新匹配值，也可以通过写 update 位，强制更新定时器匹配值，当 update 位置 1 时，定时器的计数值 (timer_tc) 以及预分频计数值 (timer_pc) 均会清零，timer_MR_x_cnt 会加载 timer_MR_x_reg 的值，定时器重新开始计数(其中交替计数模式下，定时器重新开始于递增计数，见图 11-12)。

11.4 定时器 TMR0, TMR1 通道捕捉

11.4.1 捕捉通道信号同步滤波

对于捕捉通道的输入信号，硬件会对捕捉通道进行一个同步滤波处理，如图 11-13 所示，

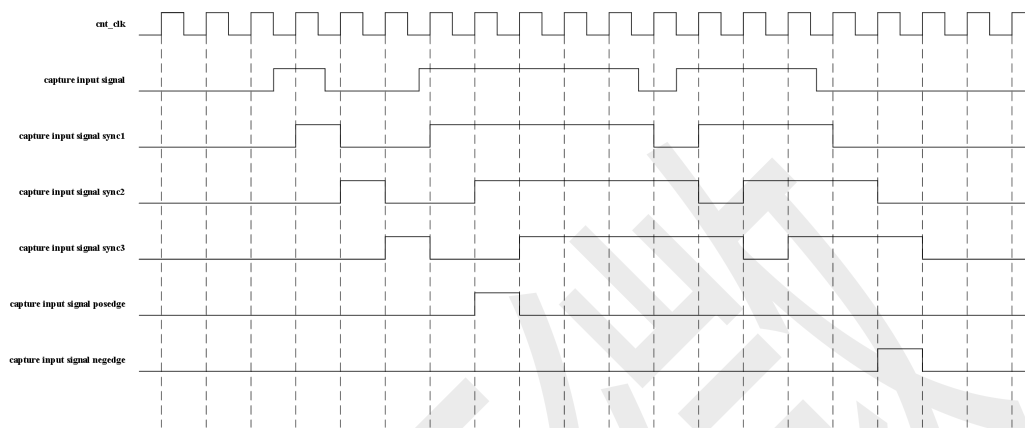


图 11-13 捕捉通道信号同步滤波

对于捕捉通道的输入信号，硬件会用 PWM 时钟对输入信号进行 3 拍的同步，当电平翻转后持续时间不小于两个时钟周期时才认为当前的这次电平翻转为有效的电平变化沿，这样，可滤除宽度小于 1 个采样时钟周期的毛刺。

11.4.2 捕捉功能描述

TMR0, TMR1 各自配有四路捕捉通道。当任意一个外部引脚通道信号出现 TMRn_CCR 寄存器设定的电平沿（上升沿或下降沿）跳变时，TMRn_IR 寄存器相关标志位会被更新（CRn_R/F 置 1）。同时，出现所设定的电平沿跳变时所对应的 TMRn_TC 寄存器的值会被存入对应通道的 TMRn_CRn 寄存器，同时根据 TMRn_CCR 寄存器内的参数设置产生输出信号（输出捕捉中断信号）并且决定 TMRn_TC 是否复位。

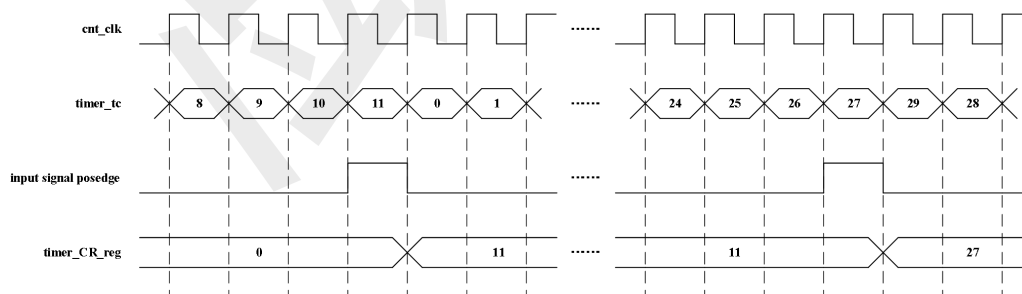


图 11-14 TMRn_TC 值被存入 TMRn_CR 寄存器且 TMRn_TC 复位（仅捕捉上升沿）

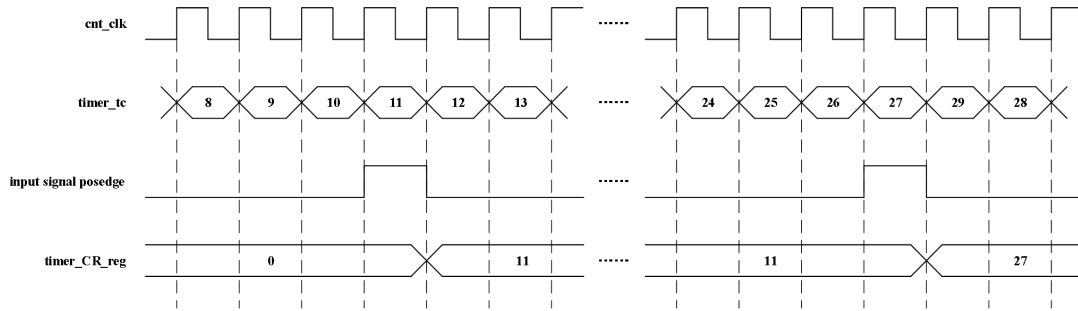


图 11-15 TMRn_TC 值被存入 TMRn_CR 寄存器且 TMRn_TC 不复位（仅捕捉上升沿）
 软件可通过配置 TMRn_CCR 寄存器来选择 TMRn_TC 在捕捉通道信号跳变后 TMRn_TC 是复位重新开始计数（图 11-14）还是继续计数（图 11-15）。

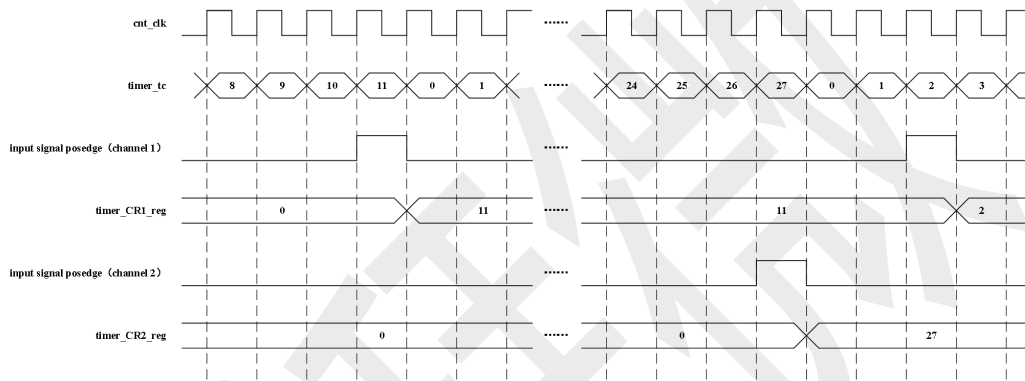


图 11-16 TMRn_TC 值被存入 TMRn_CR 寄存器且 TMRn_TC 双通道复位（仅捕捉上升沿）
 当软件配置 TMRn_TC 在捕捉通道信号跳变后复位计数时，若同时使能了两路捕捉通道发生跳变后复位计数值，则会造成捕捉值错误，如图 11-16 所示。所以建议捕捉模式下的计数值复位仅开启一路，以免出现捕捉值错误。

11.5 PWM 输出

11.5.1 PWM 模式

TMR0 和 TMR1 各自最多可支持四路引脚的比较输出功能。软件可通过配置 TMRn_CMR 寄存器选择 PWM 模式。

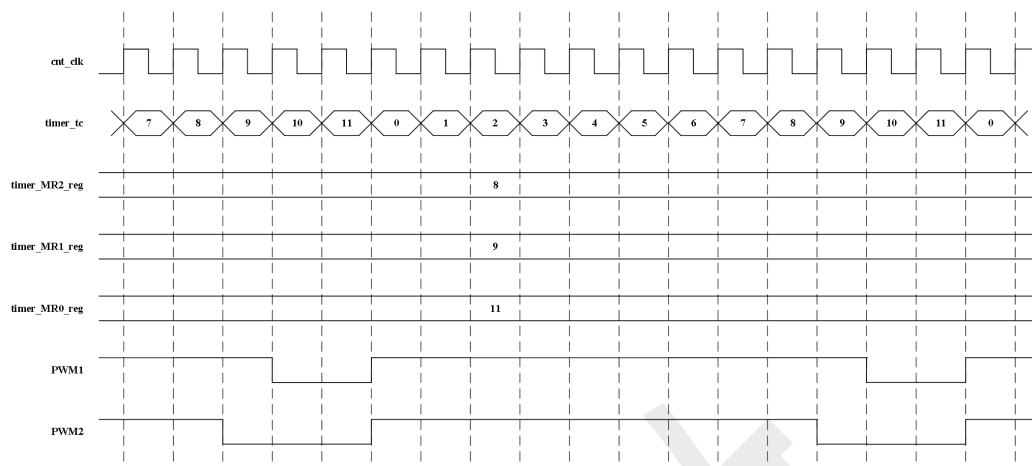


图 11-17 PWM 模式为 001 输出波形（递增）

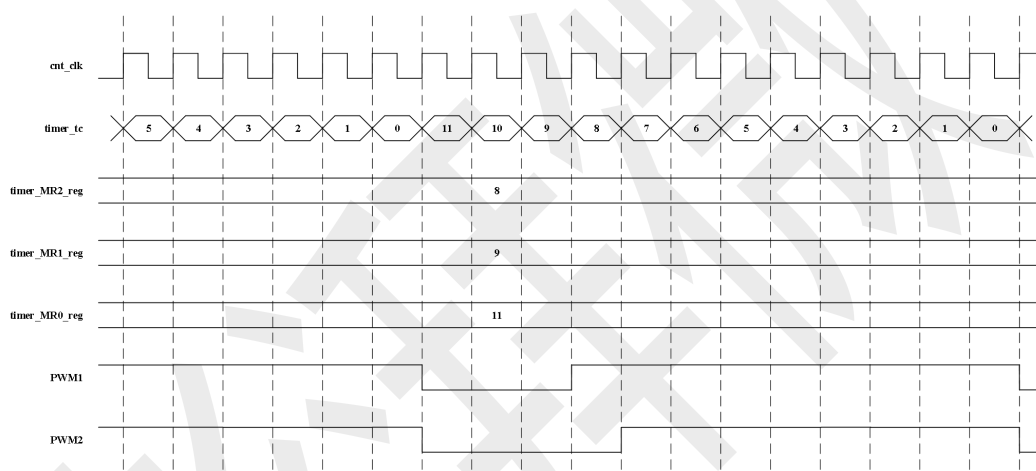


图 11-18 PWM 模式为 001 输出波形（递减）

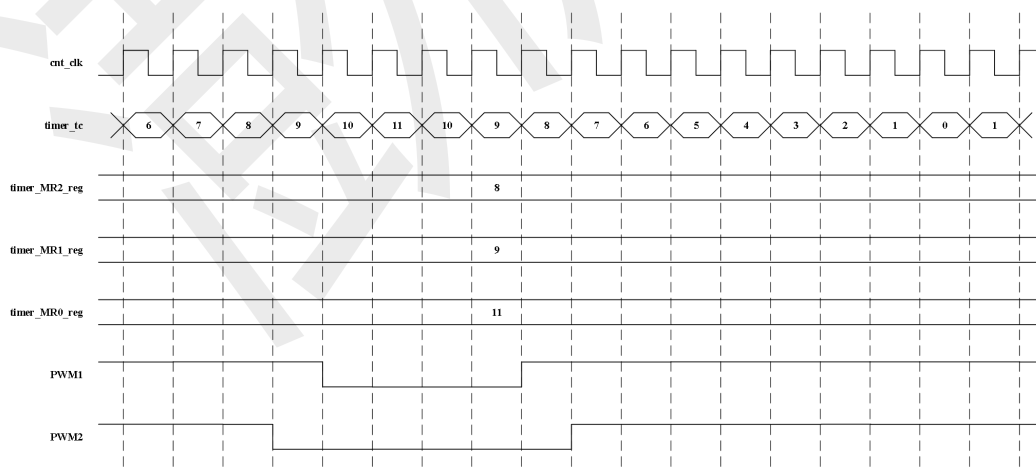


图 11-19 PWM 模式为 001 输出波形（交替）

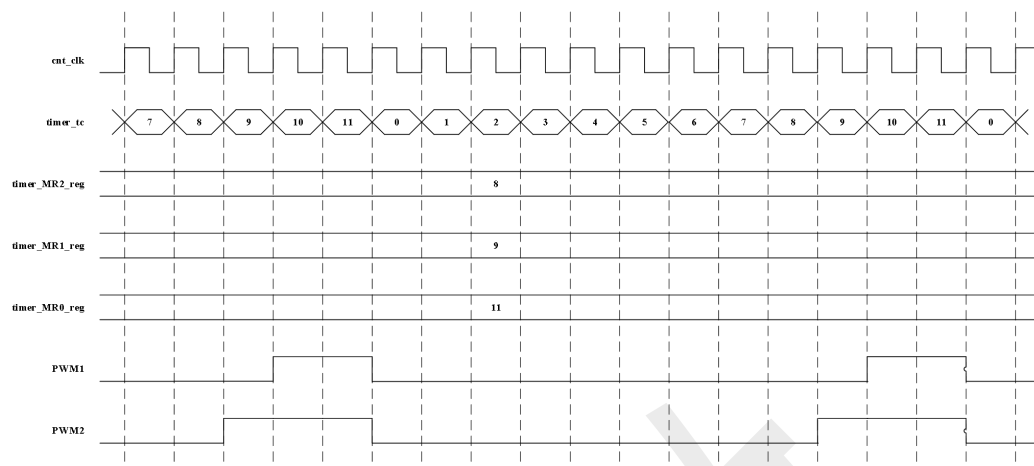


图 11-20 PWM 模式为 010 输出波形（递增）

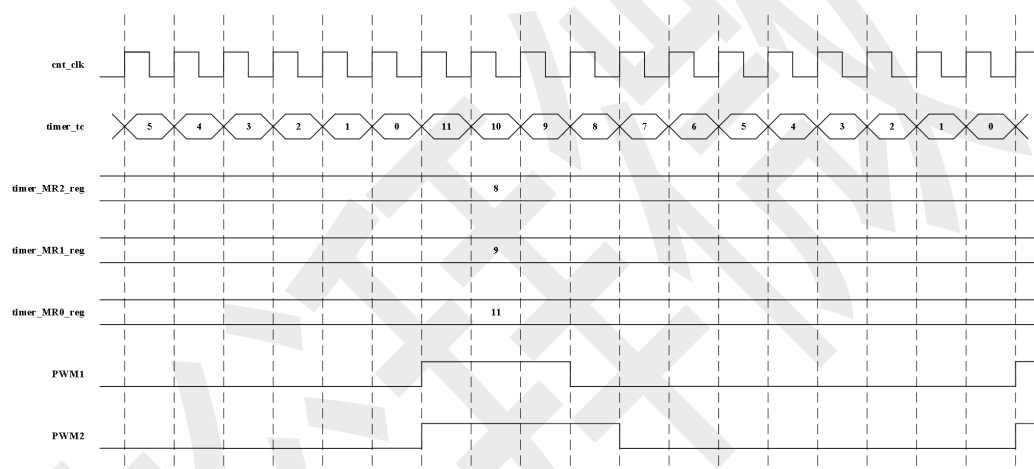


图 11-21 PWM 模式为 010 输出波形（递减）

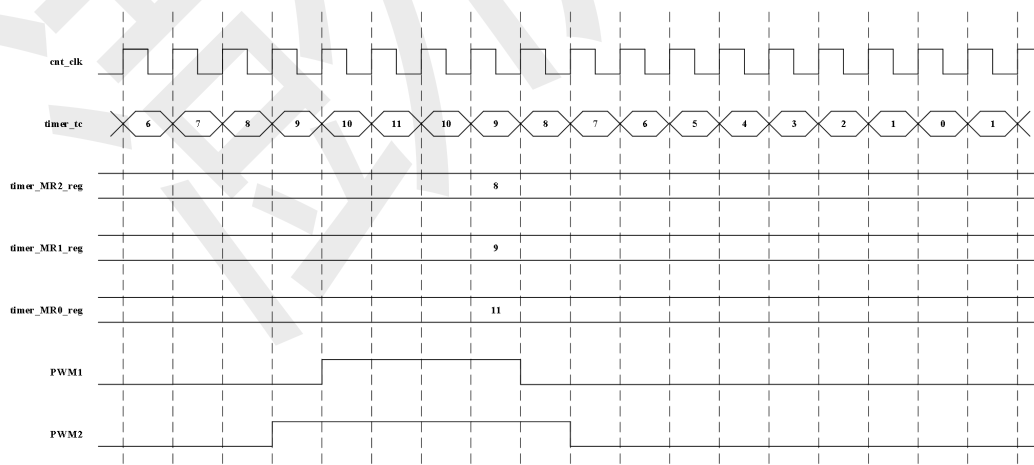


图 11-22 PWM 模式为 010 输出波形（交替）

当 PWM 通道输出波形时，此时的通道输出的周期控制由寄存器 TMRn_MR0 设定，4 路通道波形的有效占空比则分别通过 TMRn_MR1/2/3/4 寄存器设定。软件任何时候都可以设定周期和占空比寄存器，但只有在当前周期结束时，设定值才被自动装入对应的锁定寄存器并生效，开始下一个 PWM 周期的波形输出，以保证 PWM 周期的完整性（自动装载可参考图 10-3，图 10-4，图 10-5）。

TMRn_TC 复位时，将所有通道波形输出置为 0 或 1（根据比较输出控制寄存器设定）；当 TMRn_TC 和 TMRn_MR_x（x=0, 1, 2, 3, 4）比较一致时，对应的输出通道波形变为 0 或变为 1（根据比较输出控制寄存器设定）。若 TMRn_MR_x ≥ TMRn_MR0，则其对应的通道波形输出保持恒定状态（占空比 100%）。

11.5.2 死区时间插入

PWM 通道还可两两配对，互补输出。当软件使能互补输出时，通道 1 和通道 2 组成一组互补输出，通道 3 和通道 4 组成一组互补输出。此时，pwm 波形的占空比由通道 1（TMRn_MR1）和通道 3（TMRn_MR3）控制。

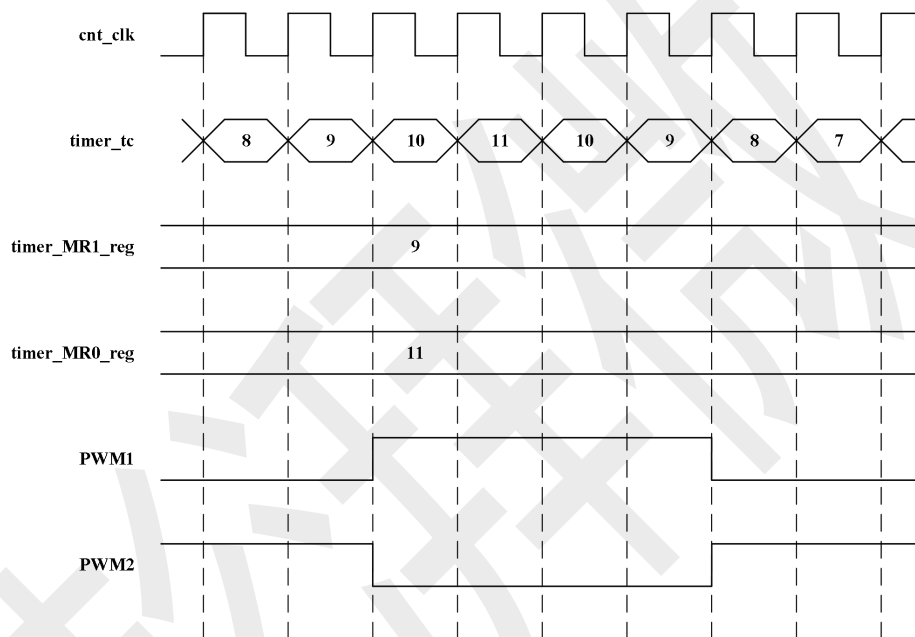


图 11-23 无死区插入的互补 PWM 输出波形

图 11-23 显示了互补输出时的波形，通道 1 和通道 2 的状态互反。同理，通道 3 和通道 4 在互补输出 PWM 波形时也是状态互反。

由于在现实中，电路存在延时，为了避免互补输出时，两通道同时导通，故 PWM 输出支持死区时间配置以避免因电路延时造成的两通道同时导通。

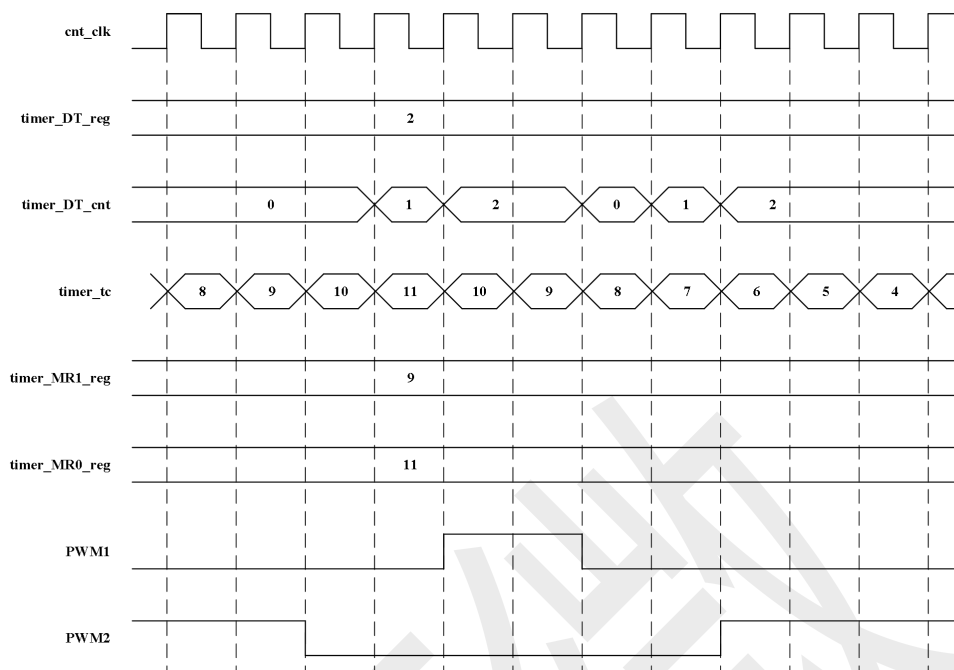


图 11-24 死区插入的互补 PWM 输出波形

图 11-24 显示了死区时间是如何在 PWM 波形中插入的，死区时间仅在互补输出模式下会插入，独立输出时，无死区时间插入。

11.5.3 刹车插入

PWM 输出还支持刹车输入，软件可通过配置 TMRn_CMR 寄存器使能/禁止刹车输入 (BKE)，刹车输入的有效电平 (BKP) 以及刹车输入发生时是否产生中断 (BKI)。

硬件对刹车输入也会进行同步滤波的处理 (参考 10.4.1 章节) 以防止毛刺误触发刹车输入功能。

当刹车输入处于有效电平时，PWM 输出会根据 TMRn_CMR 寄存器的 PWM 通道初始值设置，强制让 PWM 输出处于初始电平，并复位当前计数值 (TMRn_TC) 和当前预分频计数值 (TMRn_PC)。

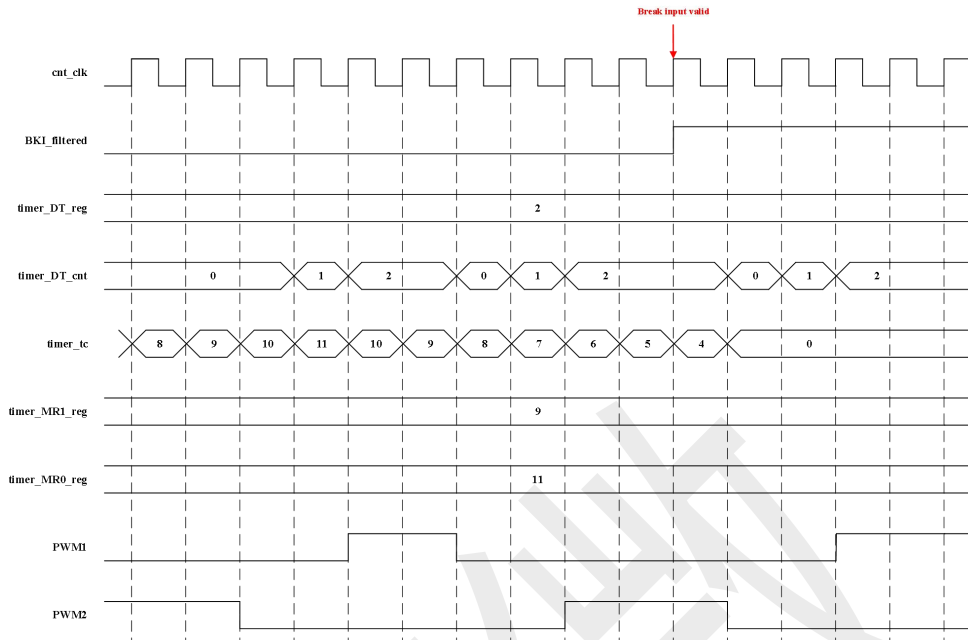


图 11-25 刹车输入有效电平时，死区插入的互补 PWM 输出波形 (PWM1 = 1)

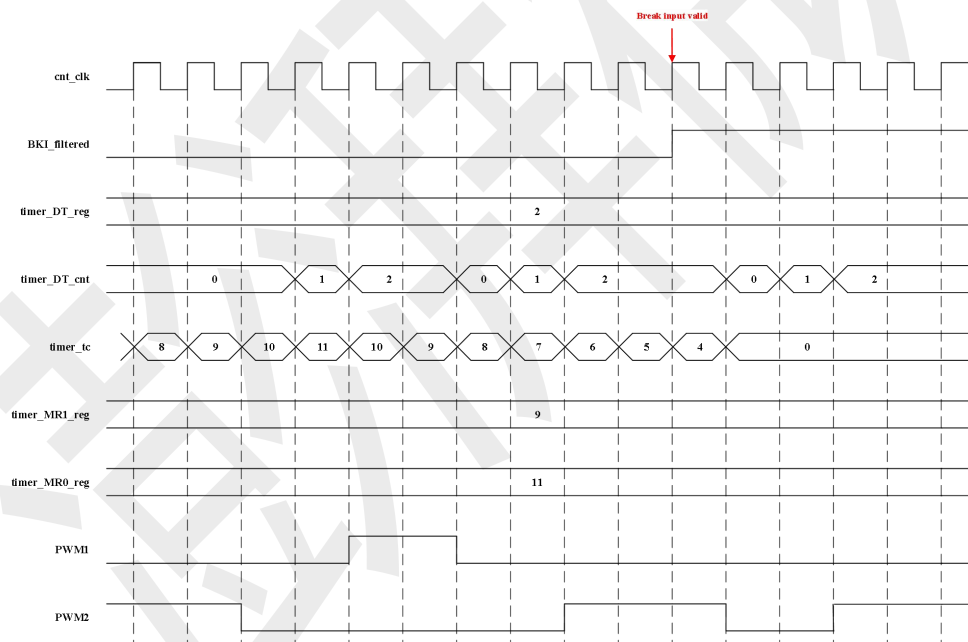


图 11-26 刹车输入有效电平时，死区插入的互补 PWM 输出波形 (PWM1 = 0)

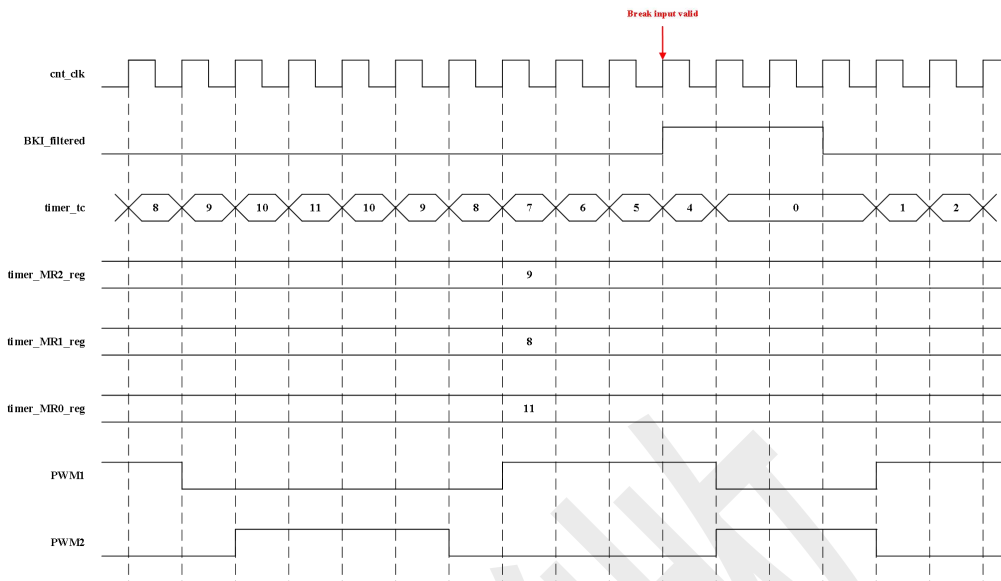


图 11-27 刹车输入有效电平时，独立 PWM 输出波形

图 11-26 显示了 PWM 模式为 010b 在刹车输入时，互补 PWM 输出的波形；图 11-27 显示了 PWM1 通道的 PWM 模式为 010b 且初始态设为 0，PWM2 通道的 PWM 模式为 001b 且初始态设为 1，在刹车输入有效时，作为独立 PWM 输出的波形。

11.6 寄存器列表

地址	寄存器	描述	备注
0x4001_2800	TMR0_IR	TMR0 中断标志寄存器	TMR0_IR 说明
0x4001_2804	TMR0_TCR	TMR0 控制寄存器	TMR0_TCR 说明
0x4001_2808	TMR0_INT_RPT	TMR0 溢出累计次数寄存器	TMR0_INT_RPT 说明
0x4001_280C	TMR0_INT_RPTC	TMR0 溢出累计计数寄存器	TMR0_INT_RPTC 说明
0x4001_2810	TMR0_PR	TMR0 预分频系数寄存器	TMR0_PR 说明
0x4001_2814	TMR0_PC	TMR0 预分频计数值寄存器	TMR0_PC 说明
0x4001_2818	TMR0_TC	TMR0 计数值寄存器	TMR0_TC 说明
0x4001_281C	TMR0_MCR	TMR0 匹配控制寄存器	TMR0_MCR 说明
0x4001_2820	TMR0_MR0	TMR0 匹配值 0 寄存器	TMR0_MR0 说明
0x4001_2824	TMR0_MR1	TMR0 匹配值 1 寄存器	TMR0_MR1 说明
0x4001_2828	TMR0_MR2	TMR0 匹配值 2 寄存器	TMR0_MR2 说明
0x4001_282C	TMR0_MR3	TMR0 匹配值 3 寄存器	TMR0_MR3 说明
0x4001_2830	TMR0_MR4	TMR0 匹配值 4 寄存器	TMR0_MR4 说明
0x4001_2834	TMR0_CCR	TMR0 捕获控制寄存器	TMR0_CCR 说明
0x4001_2838	TMR0_CR1	TMR0 通道 1 捕获值寄存器	TMR0_CR1 说明
0x4001_283C	TMR0_CR2	TMR0 通道 2 捕获值寄存器	TMR0_CR2 说明
0x4001_2840	TMR0_CR3	TMR0 通道 3 捕获值寄存器	TMR0_CR3 说明
0x4001_2844	TMR0_CR4	TMR0 通道 4 捕获值寄存器	TMR0_CR4 说明
0x4001_2848	TMR0_CMR	TMR0 比较输出控制寄存器	TMR0_CMR 说明
0x4001_284C	TMR0_DT	TMR0 死区时间控制寄存器	TMR0_DT 说明
0x4001_2C00	TMR1_IR	TMR1 中断标志寄存器	TMR1_IR 说明
0x4001_2C04	TMR1_INT_RPT	TMR1 溢出累计次数寄存器	TMR1_TCR 说明
0x4001_2C08	TMR1_INT_RPTC	TMR1 溢出累计计数寄存器	TMR1_INT_RPT 说明
0x4001_2C0C	TMR1_TCR	TMR1 控制寄存器	TMR1_INT_RPTC 说明
0x4001_2C10	TMR1_PR	TMR1 预分频系数寄存器	TMR1_PR 说明
0x4001_2C14	TMR1_PC	TMR1 预分频计数值寄存器	TMR1_PC 说明
0x4001_2C18	TMR1_TC	TMR1 计数值寄存器	TMR1_TC 说明
0x4001_2C1C	TMR1_MCR	TMR1 匹配控制寄存器	TMR1_MCR 说明
0x4001_2C20	TMR1_MR0	TMR1 匹配值 0 寄存器	TMR1_MR0 说明
0x4001_2C24	TMR1_MR1	TMR1 匹配值 1 寄存器	TMR1_MR1 说明
0x4001_2C28	TMR1_MR2	TMR1 匹配值 2 寄存器	TMR1_MR2 说明
0x4001_2C2C	TMR1_MR3	TMR1 匹配值 3 寄存器	TMR1_MR3 说明
0x4001_2C30	TMR1_MR4	TMR1 匹配值 3 寄存器	TMR1_MR4 说明
0x4001_2C34	TMR1_CCR	TMR1 捕获控制寄存器	TMR1_CCR 说明
0x4001_2C38	TMR1_CR1	TMR1 通道 1 捕获值寄存器	TMR1_CR1 说明
0x4001_2C3C	TMR1_CR2	TMR1 通道 2 捕获值寄存器	TMR1_CR2 说明

0x4001_2C40	TMR1_CR3	TMR1 通道 3 捕获值寄存器	TMR1_CR3 说明
0x4001_2C44	TMR1_CR4	TMR1 通道 4 捕获值寄存器	TMR1_CR4 说明
0x4001_2C48	TMR1_CMR	TMR1 比较输出控制寄存器	TMR1_CMR 说明
0x4001_2C4C	TMR1_DT	TMR1 死区时间控制寄存器	TMR1_DT 说明

11.7 寄存器描述

11.7.1 定时器 TMRn 中断寄存器 TMRn_IR (n = 0, 1)

(地址: TMR0: 0x4001_2800; TMR1: 0x4001_2C00)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	BK	CR4_F	CR4_R	CR3_F	CR3_R	CR2_F	CR2_R	CR1_F	CR1_R	MR4	MR3	MR2	MR1	MR0
R/W	R	R	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c	RW1c
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] MR0: 定时器匹配 0 标志位**
 0b: TMRn 的 TMRn_TC 寄存器在递增计数时与 TMRn_MR0 寄存器的值未发生过匹配或者在递减计数时未计数到 0x0000
 1b: TMRn 的 TMRn_TC 寄存器在递增计数时与 TMRn_MR0 寄存器的值发生过匹配或者在递减计数时到过 0x0000
 软件可对该位写 1 将其清除
- **位[1] MR1: 定时器匹配 1 标志位**
 0b: TMRn 的 TMRn_TC 寄存器与 TMRn_MR1 寄存器的值未发生过匹配
 1b: TMRn 的 TMRn_TC 寄存器与 TMRn_MR1 寄存器的值发生过匹配,
 软件可对该位写 1 将其清除
- **位[2] MR2: 定时器匹配 2 标志位**
 0b: TMRn 的 TMRn_TC 寄存器与 TMRn_MR2 寄存器的值未发生过匹配
 1b: TMRn 的 TMRn_TC 寄存器与 TMRn_MR2 寄存器的值发生过匹配,
 软件可对该位写 1 将其清除
- **位[3] MR3: 定时器匹配 3 标志位**
 0b: TMRn 的 TMRn_TC 寄存器与 TMRn_MR3 寄存器的值未发生过匹配
 1b: TMRn 的 TMRn_TC 寄存器与 TMRn_MR3 寄存器的值发生过匹配,
 软件可对该位写 1 将其清除
- **位[4] MR4: 定时器匹配 4 标志位**
 0b: TMRn 的 TMRn_TC 寄存器与 TMRn_MR4 寄存器的值未发生过匹配
 1b: TMRn 的 TMRn_TC 寄存器与 TMRn_MR4 寄存器的值发生过匹配,
 软件可对该位写 1 将其清除
- **位[5] CR1_R: 定时器捕获 1 上升沿标志位**
 0b: TMRn 捕获 1 通道上升沿捕获未发生
 1b: TMRn 捕获 1 通道上升沿捕获发生, 软件可对该位写 1 将其清除

- **位[6] CR1_F:** **定时器捕获 1 下降沿标志位**
 0b: TMRn 捕获 1 通道下降沿捕获未发生
 1b: TMRn 捕获 1 通道下降沿捕获发生, 软件可对该位写 1 将其清除
- **位[7] CR2_R:** **定时器捕获 2 上升沿标志位**
 0b: TMRn 捕获 2 通道上升沿捕获未发生
 1b: TMRn 捕获 2 通道上升沿捕获发生, 软件可对该位写 1 将其清除
- **位[8] CR2_F:** **定时器捕获 2 下降沿标志位**
 0b: TMRn 捕获 2 通道下降沿捕获未发生
 1b: TMRn 捕获 2 通道下降沿捕获发生, 软件可对该位写 1 将其清除
- **位[9] CR3_R:** **定时器捕获 3 上升沿标志位**
 0b: TMRn 捕获 3 通道上升沿捕获未发生
 1b: TMRn 捕获 3 通道上升沿捕获发生, 软件可对该位写 1 将其清除
- **位[10] CR3_F:** **定时器捕获 3 下降沿标志位**
 0b: TMRn 捕获 3 通道下降沿捕获未发生
 1b: TMRn 捕获 3 通道下降沿捕获发生, 软件可对该位写 1 将其清除
- **位[11] CR4_R:** **定时器捕获 4 上升沿标志位**
 0b: TMRn 捕获 4 通道上升沿捕获未发生
 1b: TMRn 捕获 4 通道上升沿捕获发生, 软件可对该位写 1 将其清除
- **位[12] CR4_F:** **定时器捕获 4 下降沿标志位**
 0b: TMRn 捕获 4 通道下降沿捕获未发生
 1b: TMRn 捕获 4 通道下降沿捕获发生, 软件可对该位写 1 将其清除
- **位[13] BK:** **定时器刹车输入标志位**
 0b: TMRn 刹车输入未发生
 1b: TMRn 刹车输入发生, 软件可对该位写 1 将其清除
- **位[31:14] :** **保留**

11.7.2 定时器 TMRn 中断累计次数控制寄存器 TMRn_INT_RPT (n = 0, 1)

(地址: TMR0: 0x4001_2808; TMR1: 0x4001_2C08)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	INT_RPT			
R/W	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[3:0] INT_RPT:** **中断累计次数**
 取值范围为 0x0~0xF, 最多可累计 16 次上溢出或者下溢出后输出中断
- **位[31:4] :** **保留**

11.7.3 定时器 TMRn 中断累计次数计数寄存器 TMRn_INT_RPTC (n = 0, 1)

(地址: TMR0: 0x4001_280C; TMR1: 0x4001_2C0C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	INT_RPTC			
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[3:0] INT_RPTC: 中断累计次数当前计数值
当软件更新 INT_RPT 时, 该位会清零并重新开始计数
- 位[31:4] : 保留

11.7.4 定时器 TMRn 控制寄存器 TMRn_TCR (n = 0, 1)

(地址: TMR0: 0x4001_2804; TMR1: 0x4001_2C04)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	CLKS	UPDATE	EN
R/W	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	W	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] EN: 定时器使能控制位
0b: 定时器启动
1b: 定时器禁止
- 位[1] UPDATE: 定时器匹配值更新控制位
该位读恒为 0, 软件对该位写 1 后更新定时器内部匹配值并复位当前计数值(TMRn_TC)和预分频计数值 (TMRn_PC)
- 位[2] CLKS: 定时器时钟选择控制位
0b: PCLK
1b: HS_CLK
- 位[31:3] : 保留

11.7.5 定时器 TMRn 预分频系数寄存器 TMRn_PR (n = 0, 1)

(地址: TMR0: 0x4001_2810; TMR1: 0x4001_2C10)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	PR							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[7:0] PR:** **定时器预分频系数**
 取值范围为 0~255 (十进制)。系统时钟的分频计算公式为:
 $f_{TMR} = f_{SYS} / (PR + 1)$
 其中:
 f_{TMR} 为定时器工作计数的时钟频率
 f_{SYS} 为芯片外设时钟频率
 PR 为预分频系数
 取值范围 0x00~0xFF
- **位[31:8] :** **保留**

11.7.6 定时器 TMRn 预分频当前计数值寄存器 TMRn_PC (n = 0, 1)

(地址: TMR0: 0x4001_2814; TMR1: 0x4001_2C14)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	PC							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[7:0] PC: 定时器预分频计数当前值
取值范围 0x00~0xFF
- 位[31:8] : 保留

11.7.7 定时器 TMRn 匹配值寄存器 TMRn_MRx (n = 0, 1; x = 0, 1, 2, 3, 4)

(地址: TMR0:

MR0: 0x4001_2820; MR1: 0x4001_2824; MR2: 0x4001_2828;

MR3: 0x4001_282C; MR4: 0x4001_2830;

TMR1:

MR0: 0x4001_2C20; MR1: 0x4001_2C24; MR2: 0x4001_2C28;

MR3: 0x4001_2C2C; MR4: 0x4001_2C30)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	MRx															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[15:0] MRx: 定时器匹配值
取值范围 0x0000~0xFFFF
- 位[31:16] : 保留

11.7.8 定时器 TMRn 当前计数值寄存器 TMRn_TC (n = 0, 1)

(地址: TMR0: 0x4001_2818; TMR1: 0x4001_2C18)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	TC															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[15:0] TC: 定时器当前计数值
取值范围 0x0000~0xFFFF
- 位[31:16] : 保留

11.7.9 定时器 TMRn 匹配控制寄存器 TMRn_MCR (n = 0, 1)

(地址: TMR0:

MR0: 0x4001_2820; MR1: 0x4001_2824; MR2: 0x4001_2828;

MR3: 0x4001_282C; MR4: 0x4001_2830;

TMR1:

MR0: 0x4001_2C20; MR1: 0x4001_2C24; MR2: 0x4001_2C28;

MR3: 0x4001_2C2C; MR4: 0x4001_2C30)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	DIR_RV	DIR	MR0S	MR4I	MR3I	MR2I	MR1I	MR0I
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] MR0I: TMRn_TC 和 TMRn_MR0 匹配时产生中断控制位
0b: 不产生中断
1b: 产生中断
- 位[1] MR1I: TMRn_TC 和 TMRn_MR1 匹配时产生中断控制位
0b: 不产生中断

- 1b: 产生中断
- 位[2] MR2I: TMRn_TC 和 TMRn_MR2 匹配时产生中断控制位
- 0b: 不产生中断
- 1b: 产生中断

- **位[3] MR3I:** **TMRn_TC 和 TMRn_MR3 匹配时产生中断控制位**
 0b: 不产生中断
 1b: 产生中断
- **位[4] MR4I:** **TMRn_TC 和 TMRn_MR4 匹配时产生中断控制位**
 0b: 不产生中断
 1b: 产生中断
- **位[5] MR0S:** **TMRn_TC 和 TMRn_MR0 匹配时计数器停止控制位**
 0b: 不停止计数
 1b: 停止计数
- **位[6] DIR:** **TMRn_TC 计数方向控制位**
 0b: 递增计数
 1b: 递减计数
- **位[7] DIR_RV:** **TMRn_TC 计数方向交替控制位**
 0b: 方向轮流交替禁止
 1b: 方向轮流交替使能
- **位[31:8] :** **保留**

11.7.10 定时器 TMRn 捕获控制寄存器 TMRn_CCR (n = 0, 1)

(地址: TMR0: 0x4001_2834; TMR1: 0x4001_2C34)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	CAP4IE	CAP4RST	CAP4FE	CAP4RE	CAP3IE	CAP3RST	CAP3FE	CAP3RE	CAP2IE	CAP2RST	CAP2FE	CAP2RE	CAP1IE	CAP1RST	CAP1FE	CAP1RE
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] CAP1RE:** **通道 1 脉冲上升沿捕捉使能控制**
 0b: 通道 1 脉冲上升沿捕捉禁止
 1b: 通道 1 脉冲上升沿捕捉使能
- **位[1] CAP1FE:** **通道 1 脉冲下降沿捕捉使能控制**
 0b: 通道 1 脉冲下降沿捕捉禁止
 1b: 通道 1 脉冲下降沿捕捉使能
- **位[2] CAP1RST:** **通道 1 脉冲沿捕捉计数器复位使能控制**
 0b: 通道 1 脉冲沿捕捉不复位计时器
 1b: 通道 1 脉冲沿捕捉复位计时器
- **位[3] CAP1IE:** **通道 1 脉冲沿捕捉中断使能控制**
 0b: 通道 1 脉冲沿捕捉中断禁止
 1b: 通道 1 脉冲沿捕捉中断使能

- **位[4] CAP2RE:** **通道 2 脉冲上升沿捕捉使能控制**
 0b: 通道 2 脉冲上升沿捕捉禁止
 1b: 通道 2 脉冲上升沿捕捉使能
- **位[5] CAP2FE:** **通道 2 脉冲下降沿捕捉使能控制**
 0b: 通道 2 脉冲下降沿捕捉禁止
 1b: 通道 2 脉冲下降沿捕捉使能
- **位[6] CAP2RST:** **通道 2 脉冲沿捕捉计数器复位使能控制**
 0b: 通道 2 脉冲沿捕捉不复位计时器
 1b: 通道 2 脉冲沿捕捉复位计时器
- **位[7] CAP2IE:** **通道 2 脉冲沿捕捉中断使能控制**
 0b: 通道 2 脉冲沿捕捉中断禁止
 1b: 通道 2 脉冲沿捕捉中断使能
- **位[8] CAP3RE:** **通道 3 脉冲上升沿捕捉使能控制**
 0b: 通道 3 脉冲上升沿捕捉禁止
 1b: 通道 3 脉冲上升沿捕捉使能
- **位[9] CAP3FE:** **通道 3 脉冲下降沿捕捉使能控制**
 0b: 通道 3 脉冲下降沿捕捉禁止
 1b: 通道 3 脉冲下降沿捕捉使能
- **位[10] CAP3RST:** **通道 3 脉冲沿捕捉计数器复位使能控制**
 0b: 通道 3 脉冲沿捕捉不复位计时器
 1b: 通道 3 脉冲沿捕捉复位计时器
- **位[11] CAP3IE:** **通道 3 脉冲沿捕捉中断使能控制**
 0b: 通道 3 脉冲沿捕捉中断禁止
 1b: 通道 3 脉冲沿捕捉中断使能
- **位[12] CAP4RE:** **通道 4 脉冲上升沿捕捉使能控制**
 0b: 通道 4 脉冲上升沿捕捉禁止
 1b: 通道 4 脉冲上升沿捕捉使能
- **位[13] CAP4FE:** **通道 4 脉冲下降沿捕捉使能控制**
 0b: 通道 4 脉冲下降沿捕捉禁止
 1b: 通道 4 脉冲下降沿捕捉使能
- **位[14] CAP4RST:** **通道 4 脉冲沿捕捉计数器复位使能控制**
 0b: 通道 4 脉冲沿捕捉不复位计时器
 1b: 通道 4 脉冲沿捕捉复位计时器
- **位[15] CAP4IE:** **通道 4 脉冲沿捕捉中断使能控制**
 0b: 通道 4 脉冲沿捕捉中断禁止
 1b: 通道 4 脉冲沿捕捉中断使能
- **位[31:16] :** **保留**

11.7.11 定时器 TMRn 捕捉值寄存器 TMRn_CRx (n = 0, 1; x = 1, 2, 3, 4)

(地址: TMR0:

CR1: 0x4001_2838; CR2: 0x4001_283C; CR3: 0x4001_2840; CR4: 0x4001_2844;

TMR1:

CR1: 0x4001_2C38; CR2: 0x4001_2C3C; CR3: 0x4001_2C40; CR4: 0x4001_2C44)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	CAPEEDGE
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	CAPx															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] CAPx:** **捕捉通道 x 沿变化时间值**
该位只读, 取值范围 0x0000~0xFFFF
- **位[16] CAPEEDGE:** **捕获沿标志位**
0b: 当前捕获值 CAPx 为通道下降沿时间点
1b: 当前捕获值 CAPx 为通道上升沿时间点
- **位[31:17] :** **保留**

11.7.12 定时器 TMRn 比较输出控制寄存器 TMRn_CMCR (n = 0, 1)

(地址: TMR0: 0x4001_2848; TMR1: 0x4001_2848)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	BKI	BKP	BKE	PWM4n_EN	PWM3n_EN	PWM2n_EN	PWM1n_EN	PWM4_IDLE	PWM3_IDLE	PWM2_IDLE	PWM1_IDLE
R/W	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	MC4		-	-	MC3		-	-	MC2		-	-	MC1	
R/W	R	R	RW	RW	R	R	RW	RW	R	R	RW	RW	R	R	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[1:0] MC1:** **输出通道 1 比较输出模式选择**
 - 00b: PWM1 不输出
 - 01b: TMRn_TC 大于 TMRn_MR1 时, 对应 PWM1 输出 0, 否则输出 1
 - 10b: TMRn_TC 大于 TMRn_MR1 时, 对应 PWM1 输出 1, 否则输出 0
 - 11b: 计数到 TMRn_MR1 匹配值时, PWM1 翻转
- **位[3:2] :** **保留**
- **位[5:4] MC2:** **输出通道 2 比较输出模式选择**
 - 00b: PWM2 不输出
 - 01b: TMRn_TC 大于 TMRn_MR2 时, 对应 PWM2 清 0, 否则置 1
 - 10b: TMRn_TC 大于 TMRn_MR2 时, 对应 PWM2 置 1, 否则清 0
 - 11b: 计数到 TMRn_MR2 匹配值时, PWM2 翻转
- **位[7:6] :** **保留**
- **位[9:8] MC3:** **输出通道 3 比较输出模式选择**
 - 00b: PWM3 不输出
 - 01b: TMRn_TC 大于 TMRn_MR3 时, 对应 PWM3 清 0, 否则置 1
 - 10b: TMRn_TC 大于 TMRn_MR3 时, 对应 PWM3 置 1, 否则清 0
 - 11b: 计数到 TMRn_MR3 匹配值时, PWM3 翻转
- **位[11:10] :** **保留**
- **位[13:12] MC4:** **输出通道 4 比较输出模式选择**
 - 00b: PWM4 不输出
 - 01b: TMRn_TC 大于 TMRn_MR4 时, 对应 PWM4 清 0, 否则置 1
 - 10b: TMRn_TC 大于 TMRn_MR4 时, 对应 PWM4 置 1, 否则清 0
 - 11b: 计数到 TMRn_MR4 匹配值时, PWM4 翻转
- **位[15:14] :** **保留**
- **位[16] PWM1_IDLE:** **输出通道 1 初始值**

当通道 1 的 PWM 模式不为 00b 时, 仅在定时器未使能或者刹车输入有效时, 输出通道 1 的电平为该位设置的状态

 - 0b: 输出通道 0 初始值为 0
 - 1b: 输出通道 0 初始值为 1

- **位[17] PWM2_IDLE:** **输出通道 2 初始值**
当通道 2 的 PWM 模式不为 00b 时，仅在定时器未使能或者刹车输入有效时，输出通道 2 的电平为该位设置的状态
0b: 输出通道 1 初始值为 0
1b: 输出通道 1 初始值为 1
- **位[18] PWM3_IDLE:** **输出通道 3 初始值**
当通道 3 的 PWM 模式不为 00b 时，仅在定时器未使能或者刹车输入有效时，输出通道 3 的电平为该位设置的状态
0b: 输出通道 2 初始值为 0
1b: 输出通道 2 初始值为 1
- **位[19] PWM4_IDLE:** **输出通道 4 初始值**
当通道 4 的 PWM 模式不为 00b 时，仅在定时器未使能或者刹车输入有效时，输出通道 4 的电平为该位设置的状态
0b: 输出通道 3 初始值为 0
1b: 输出通道 3 初始值为 1
- **位[20] PWM1n_EN:** **输出通道 1 互补输出使能**
0b: 独立输出
1b: 互补输出
- **位[21] PWM1n_EN:** **输出通道 1 互补输出使能**
0b: 独立输出
1b: 互补输出
- **位[22] PWM1n_EN:** **输出通道 1 互补输出使能**
0b: 独立输出
1b: 互补输出
- **位[23] PWM1n_EN:** **输出通道 1 互补输出使能**
0b: 独立输出
1b: 互补输出
- **位[24] BKE:** **刹车输入使能控制**
0b: 刹车输入禁止
1b: 刹车输入使能
- **位[25] BKP:** **刹车输入有效电平控制**
0b: 刹车输入低电平有效
1b: 刹车输入高电平有效
- **位[26] BKI:** **刹车输入中断使能控制**
0b: 刹车输入中断禁止
1b: 刹车输入中断使能
- **位[31:27] :** **保留**

11.7.13 定时器 TMRn 死区时间控制寄存器 TMRn_DT (n = 0, 1)

(地址: TMR0: 0x4001_284C; TMR1: 0x4001_2C4C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	DT							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[7:0] DT: 死区时间控制
LSB 为一个定时器时钟周期
- 位[31:8] : 保留

12 普通定时器

12.1 普通定时器综述

定时器 TMR2、TMR3 均为 16 位宽通用定时器，工作于外设时钟下，可选择递增或者递减的计数方式。

12.2 定时器 TMR2, TMR3 框图

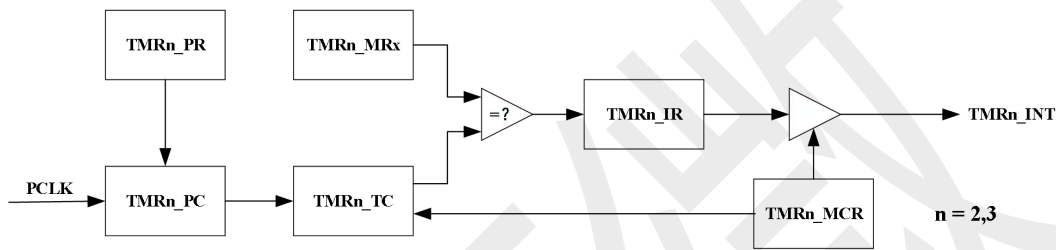


图 12-1 定时器 TMR2、TMR3 工作原理图

图 12-1 显示了定时器 TMR2、TMR3 的工作原理。定时器所有寄存器的读写均在 PCLK 下可通过软件进行读写操作，定时器时钟为 PCLK，TMRn_PC 寄存器根据 TMRn_PR 寄存器内设置的分频系数对定时器时钟进行计数分频，TMRn_TC 寄存器在通过 TMRn_PC 寄存器分频后产生的时钟下计数。

TMRn_TC 一共有两种计数方式，递增计数和递减计数。

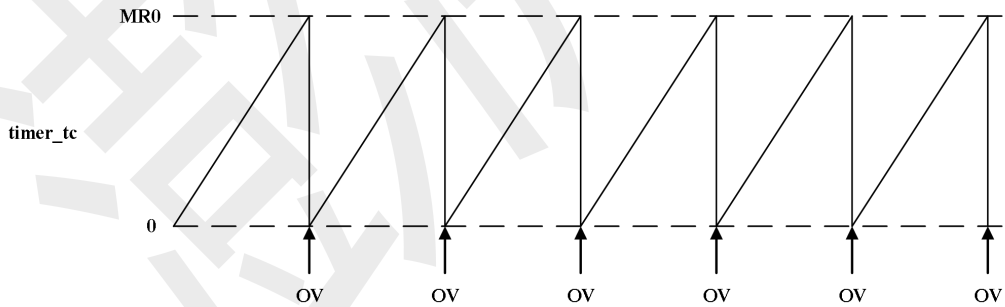


图 12-2 递增计数示意图

递增计数模式下，当 TMRn_TC 与 TMRn_MR0 寄存器内的值相同时，TMRn_IR 会产生相应的比较定时溢出信号（TMRn_IR 寄存器的 MR0 置 1），同时定时器会根据 TMRn_MCR 寄存器内的参数设置产生相应的输出信号（输出比较定时中断信号或停止定时器或复位 TMRn_TC 寄存器为 0 重新开始计数）。递增计数的示意图如图 12-2 所示。

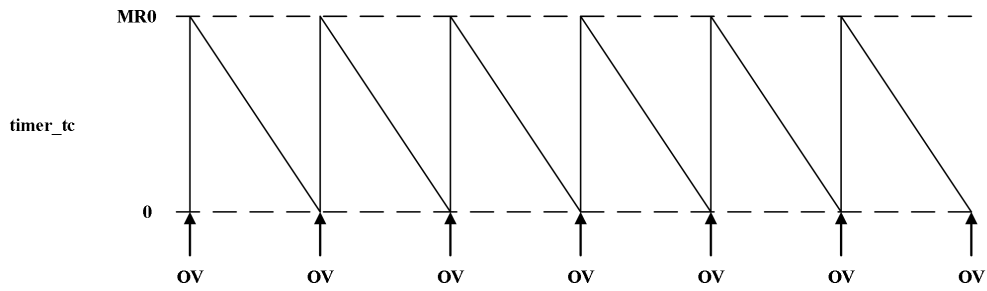


图 12-3 递减计数示意图

递减计数模式下，当 TMRn_TC 为 0 时，TMRn_IR 会产生相应的比较定时下溢出信号（TMRn_IR 寄存器的 UV 置 1），同时定时器会根据 TMRn_MCR 寄存器内的参数设置产生相应的输出信号（输出比较定时中断信号或停止定时器或复位 TMRn_TC 寄存器为 TMRn_MR0 重新开始计数）。递减计数的示意图如图 12-3 所示。

12.3 寄存器列表

地址	寄存器	描述	备注
0x4000_1000	TMR2_IR	TMR2 中断寄存器	TMR2_IR 说明
0x4000_1004	TMR2_TCR	TMR2 控制寄存器	TMR2_TCR 说明
0x4000_1008	TMR2_PR	TMR2 预分频系数寄存器	TMR2_PR 说明
0x4000_100C	TMR2_PC	TMR2 预分频计数值寄存器	TMR2_PC 说明
0x4000_1010	TMR2_TC	TMR2 计数值寄存器	TMR2_TC 说明
0x4000_1014	TMR2_MCR	TMR2 匹配控制寄存器	TMR2_MCR 说明
0x4000_1018	TMR2_MR0	TMR2 匹配值寄存器	TMR2_MR0 说明
0x4000_1400	TMR3_IR	TMR3 中断寄存器	TMR3_IR 说明
0x4000_1404	TMR3_TCR	TMR3 控制寄存器	TMR3_TCR 说明
0x4000_1408	TMR3_PR	TMR3 预分频系数寄存器	TMR3_PR 说明
0x4000_140C	TMR3_PC	TMR3 预分频计数值寄存器	TMR3_PC 说明
0x4000_1410	TMR3_TC	TMR3 计数值寄存器	TMR3_TC 说明
0x4000_1414	TMR3_MCR	TMR3 匹配控制寄存器	TMR3_MCR 说明
0x4000_1418	TMR3_MR0	TMR3 匹配值寄存器	TMR3_MR0 说明

12.4 寄存器描述

12.4.1 定时器 TMRn 中断寄存器 TMRn_IR (n = 2, 3)

(地址: TMR2: 0x4001_1000; TMR3: 0x4001_1400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	OV
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW1c
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] OV: 定时器溢出中断标志位
 - 0b: TMRn 的 TMRn_TC 寄存器未发生过溢出
 - 1b: TMRn 的 TMRn_TC 寄存器发生过溢出
软件可对该位写 1 将其清除
- 位[31:1] : 保留

12.4.2 定时器 TMRn 控制寄存器 TMRn_TCR (n = 2, 3)

(地址: TMR2: 0x4001_1004; TMR3: 0x4001_1404)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RST	EN
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] EN: 定时器使能控制位
 - 0b: 定时器启动
 - 1b: 定时器禁止

- **位[1] RST: 定时器匹复位控制位**
当写此位为 1 时，会对定时器计数值寄存器和预分频计数值寄存器执行一次复位操作
读:
0b: 定时器不在复位过程中
1b: 定时器计数值寄存器、预分频计数值寄存器复位过程中
写:
0b: 无效
1b: 复位定时器计数器寄存器和预分频器寄存器
- **位[31:2] : 保留**

12.4.3 定时器 TMRn 预分频系数寄存器 TMRn_PR (n = 2, 3)

(地址: TMR2: 0x4001_1008; TMR3: 0x4001_1408)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	PR							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[7:0] PR: 定时器预分频系数**
取值范围为 0~255（十进制）。系统时钟的分频计算公式为：
 $f_{TMR} = f_{SYS} / (PR + 1)$
其中：
f_{TMR} 为定时器工作计数的时钟频率
f_{SYS} 为芯片系统时钟频率
PR 为预分频系数
取值范围 0x00~0xFF
- **位[31:8] : 保留**

12.4.4 定时器 TMRn 预分频当前计数值寄存器 TMRn_PC (n = 2, 3)

(地址: TMR2: 0x4001_100C; TMR3: 0x4001_140C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	PC							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[7:0] PC: 定时器预分频计数当前值
取值范围 0x00~0xFF
- 位[31:8] : 保留

12.4.5 定时器 TMRn 匹配值寄存器 TMRn_MR0 (n = 2, 3)

(地址: TMR2: 0x4001_1018; TMR3: 0x4001_1418)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	MR0															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[15:0] MR0: 定时器匹配值
取值范围 0x0000~0xFFFF
- 位[31:16] : 保留

12.4.6 定时器 TMRn 当前计数值寄存器 TMRn_TC (n = 2, 3)

(地址: TMR2: 0x4001_1010; TMR3: 0x4001_1410)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	TC															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[15:0] TC: 定时器当前计数值
取值范围 0x0000~0xFFFF
- 位[31:16] : 保留

12.4.7 定时器 TMRn 匹配控制寄存器 TMRn_MCR (n = 2, 3)

(地址: TMR2: 0x4001_1014; TMR3: 0x4001_1414)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	DIR	MR0S	MR0R	MR0I
R/W	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] MR0I: TMRn_TC 溢出时产生中断控制位
0b: 不产生中断
1b: 产生中断
- 位[1] MR0R: TMRn_TC 溢出时计数复位控制位
0b: 不产生复位
1b: 产生复位
- 位[2] MR0S: TMRn_TC 溢出时计数器停止控制位
0b: 不停止计数
1b: 停止计数

- 位[3] DIR: TMRn_TC 计数方向控制位
 - 0b: 递增计数
 - 1b: 递减计数
- 位[31:4] : 保留

13 低功耗定时器 TMR4

13.1 定时器综述

定时器 TMR4 为 16 位宽通用定时器，工作在内部低速振荡 32kHz 时钟下，递减计数。TMR4 能够在休眠状态下工作，在系统休眠状态下输出信号可唤醒休眠。

13.2 定时器 timer4 框图

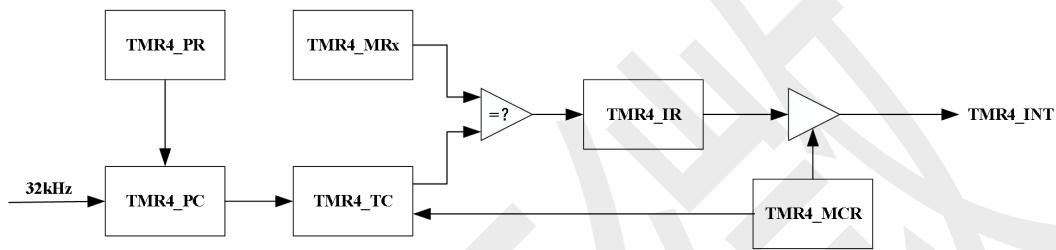


图 13-1 定时器 timer4 工作原理图

图 13-1 显示了定时器 timer4 的工作原理。定时器所有寄存器的读写均在 PCLK 下可通过软件进行读写操作，定时器时钟为内部低速振荡 32kHz 时钟，TMRn_PC 寄存器根据 TMRn_PR 寄存器内设置的分频系数对定时器时钟进行计数分频，TMRn_TC 寄存器在通过 TMRn_PC 寄存器分频后产生的时钟下递增计数。

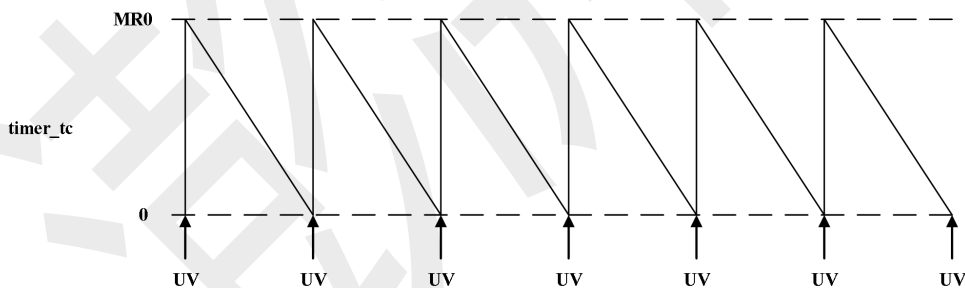


图 13-2 递增计数示意图

递增计数模式下，当 TMRn_TC 与 TMRn_MR0 寄存器内的值相同时，TMRn_IR 会产生相应的比较定时上溢出信号（TMRn_IR 寄存器的 MR0 置 1），同时定时器会根据 TMRn_MCR 寄存器内的参数设置产生相应的输出信号（输出比较定时中断信号或停止定时器或复位 TMRn_TC 寄存器为 0 重新开始计数）。递增计数的示意图如图 13-2 所示。

TMR4 的寄存器配置是在 PCLK 域下，而计数器则是工作在内部低速振荡 32kHz 时钟下，所以两个时钟域间有同步逻辑。当 TMR4 的寄存器由软件在 PCLK 下操作更新时，随即读这些寄存器时，能立即读到更新值，但该更新值需要等定时器在两个计数时钟（内部低速振荡 32kHz 时钟）域周期同步后才会生效。

13.3 寄存器列表

地址	寄存器	描述	备注
0x4000_1C00	TMR4_IR	TMR4 中断寄存器	TMR4_IR 说明
0x4000_1C04	TMR4_TCR	TMR4 控制寄存器	TMR4_TCR 说明
0x4000_1C08	TMR4_PR	TMR4 预分频系数寄存器	TMR4_PR 说明
0x4000_1C0C	TMR4_PC	TMR4 预分频计数值寄存器	TMR4_PC 说明
0x4000_1C10	TMR4_TC	TMR4 计数值寄存器	TMR4_TC 说明
0x4000_1C14	TMR4_MCR	TMR4 匹配控制寄存器	TMR4_MCR 说明
0x4000_1C18	TMR4_MR0	TMR4 匹配值寄存器	TMR4_MR0 说明

13.4 寄存器描述
13.4.1 定时器 TMR4 中断寄存器 TMR4_IR

(地址: 0x4001_1C00)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	UV
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW1c
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] UV: **定时器溢出中断标志位**
 0b: TMR4 的 TMR4_TC 寄存器发生过溢出
 1b: TMR4 的 TMR4_TC 寄存器未发生过溢出
 软件可对该位写 1 将其清除
- 位[31:1]: **保留**

13.4.2 定时器 TMR4 控制寄存器 TMR4_TCR

(地址: 0x4001_1C04)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RST	EN
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] EN: **定时器使能控制位**
 0b: 定时器启动
 1b: 定时器禁止

- **位[1] RST: 定时器复位控制位**
 当写此位为 1 时，会对定时器计数值寄存器和预分频计数值寄存器执行一次复位操作
 读:
 0b: 定时器不在复位过程中
 1b: 定时器计数值寄存器、预分频计数值寄存器复位过程中
 写:
 0b: 无效
 1b: 复位定时器计数器寄存器和预分频器寄存器
- **位[31:2] :** 保留

13.4.3 定时器 TMR4 预分频系数寄存器 TMR4_PR

(地址: 0x4001_1C08)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	PR							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[7:0] PR: 定时器预分频系数**
 取值范围为 0~255（十进制）。系统时钟的分频计算公式为：
 $f_{TMR} = f_{SYS} / (PR + 1)$
 其中：
 f_{TMR} 为定时器工作计数的时钟频率
 f_{SYS} 为芯片系统时钟频率
 PR 为预分频系数
 取值范围 0x00~0xFF
- **位[31:8] :** 保留

13.4.4 定时器 TMR4 预分频当前计数值寄存器 TMR4_PC

(地址: 0x4001_1C0C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	PC							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[7:0] PC: 定时器预分频计数当前值
取值范围 0x00~0xFF
- 位[31:8] : 保留

13.4.5 定时器 TMR4 匹配值寄存器 TMR4_MR0

(地址: 0x4001_1C18)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	MR0															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[15:0] MR0: 定时器匹配值
取值范围 0x0000~0xFFFF
- 位[31:16] : 保留

13.4.6 定时器 TMR4 当前计数值寄存器 TMR4_TC

(地址: 0x4001_1C10)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	TC															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[15:0] TC: 定时器当前计数值
取值范围 0x0000~0xFFFF
- 位[31:16] : 保留

13.4.7 定时器 TMR4 匹配控制寄存器 TMR4_MCR

(地址: 0x4001_1C14)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MR0R	MROI
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] MR0I: TMR4_TC 溢出时产生中断控制位
0b: 不产生中断
1b: 产生中断
- 位[1] MR0R: TMR4_TC 溢出时计数复位控制位
0b: 不产生复位
1b: 产生复位
- 位[31:2] : 保留

14 蜂鸣器

14.1 蜂鸣器综述

蜂鸣器模块可产生1kHz，2kHz或者4kHz的方波信号驱动蜂鸣器，工作在内部低速振荡32kHz时钟下。

14.2 蜂鸣器框图

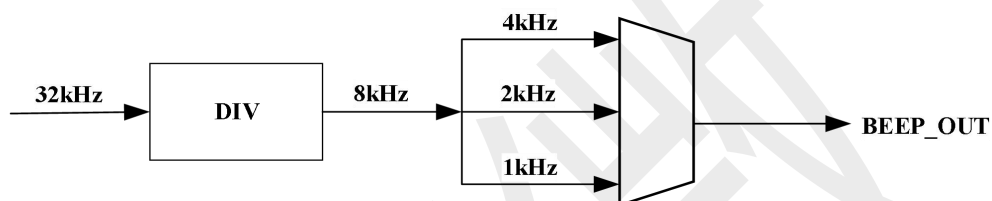


图 14-1 蜂鸣器工作原理图

图 14-1 显示了蜂鸣器的内部框图，蜂鸣器模块通过对内部低速振荡时钟的分频得到频率为 8kHz 的时标信号，再对此时标信号进行 2 分频，4 分频或者 8 分频得到 4kHz，2kHz 或者 1kHz 的方波信号并输出。

14.3 蜂鸣器软件操作说明

以下步骤描述仅供参考：

1. 软件配置 DIV 得到 8kHz 的时标信号（DIV 数值取决于当前的内部低速振荡 32kHz 时钟的真实频率）
2. 软件根据需求配置 SEL 选择输出对应频率的方波
3. 使能蜂鸣器模块

14.4 寄存器列表

地址	寄存器	描述	备注
0x4000_2800	BEEP_CON	蜂鸣器控制寄存器	BEEP_CON 说明

14.5 寄存器描述
14.5.1 蜂鸣器控制寄存器 BEEP_CON

(地址: 0x4000_2800)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	SEL		DIV					EN
R/W	R	R	R	R	R	R	R	R	RW		RW					RW
复位	0	0	0	0	0	0	0	0	00		00000					0

- **位[0] EN:** **蜂鸣器使能控制**
 0b: 蜂鸣器禁止
 1b: 蜂鸣器使能
- **位[5:1] DIV:** **时钟分频系数**
 根据内部低速振荡时钟的频率分频得到 8kHz 的方波
 00000: 2 分频
 00001: 3 分频
 00010: 4 分频

 11110: 31 分频
 11111: 32 分频
- **位[7:6] SEL:** **输出频率选择**
 00: 输出 1kHz
 01: 输出 2kHz
 10: 输出 4kHz
 11: 输出 4kHz
- **位[31:8]:** **保留**

15 系统看门狗

15.1 系统看门狗综述

系统看门狗用于监测由于外部干扰或不可预知的逻辑条件所产生的软件错误，这样的软件错误通常会导致应用程序不按照预期的方式运行，系统看门狗可提供超时复位或者产生中断的功能，提醒软件程序运行错误。适用于那些要求看门狗在精确计时的应用场景。

15.2 系统看门狗功能描述

系统看门狗模块为一个 32 位宽的递减计数器。系统看门狗计数器启动时由重载设定寄存器（SYSWDGLOAD）载入初值，然后在时钟的驱动下每个周期进行递减计数。

看门狗计数器启动后，递减计数值第一次归零时，将触发SYSWDG中断响应。第一次归零后，SYSWDG重新载入初值，继续递减计数，当计数值第二次归零，且SYSWDGCTRL寄存器的RSTE位设为1，且之前的中断未清除的情况下，将立即引发一次系统硬件复位，即看门狗复位。

15.3 系统看门狗软件操作说明

15.3.1 启动系统看门狗

上电复位后系统看门狗模块默认处于禁止状态，应用软件根据需求写SYSWDGLOAD寄存器设定计数器初始值，然后设置SYSWDGCTRL寄存器的INTE位为1启动SYSWDG开始工作。

15.3.2 系统看门狗

应用软件可以编写自己的中断服务程序来应对SYSWDG的中断事件，如果发生SYSWDG中断，一般意味着应用软件已有过长的间隔时间没有执行清除SYSWDG或者过短的时间间隔执行了清除SYSWDG，可用于软件调试阶段的可靠性诊断。正式发布的应用程序不能在自己的中断服务程序中执行清除看门狗中断标志指令，否则看门狗将永远不会引发系统复位，失去了其系统监控的作用。

应用软件在正常运行时，必须在设定的SYSWDG定时时间范围内对SYSWDGINTCLR寄存器写一次任意数，以重载复位SYSWDG计数器。正常情况下不应出现SYSWDG中断，更不能出现SYSWDG复位。

15.4 寄存器列表

地址	寄存器	描述	备注
0x4000_2000	SYSWDGLOAD	SYSWDG 计数重载寄存器	SYSWDGLOAD 说明
0x4000_2004	SYSWDGVALUE	SYSWDG 计数值寄存器	SYSWDGVALUE 说明
0x4000_2008	SYSWDGCTRL	SYSWDG 控制寄存器	SYSWDGCTRL 说明
0x4000_200C	SYSWDGINTCLR	SYSWDG 中断清除寄存器	SYSWDGINTCLR 说明
0x4000_2010	SYSWDGRIS	SYSWDG 原始中断标志寄存器	SYSWDGRIS 说明
0x4000_2014	SYSWDGMIS	SYSWDG 掩蔽中断标志寄存器	SYSWDGMIS 说明
0x4000_2400	SYSWDGLOCK	SYSWDG 锁定控制寄存器	SYSWDGLOCK 说明

15.5 寄存器描述
15.5.1 SYSWDG 计数重载寄存器 SYSWDGLOAD

(地址: 0x4000_2000)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	SYSWDGLOAD															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	SYSWDGLOAD															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- **位[31:0] SYSWDGLOAD:** 系统看门狗重载初始值
 该值决定了 SYSWDG 的定时时间, 计算 SYSWDGLOAD 重载值的公式为:

$$\text{SYSWDGLOAD} = T_{\text{WDT}} * f_{\text{CLK}} - 1$$
 其中:
 T_{WDT} 为期望的 SYSWDG 定时时间, 单位为 ms
 f_{CLK} 为 SYSWDG 的时钟频率, 为外设时钟
 例如, 若应用软件希望设置约 20ms 的看门狗定时时间, 则按上述公式算得 SYSWDGLOAD 应赋值为 639

15.5.2 SYSWDG 计数值寄存器 SYSWDGVALUE

(地址: 0x4000_2004)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	SYSWDGVALUE															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	SYSWDGVALUE															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- **位[31:0] SYSWDGVALUE: 系统看门狗当前计数值**
SYSWDG 的递减计数器的当前计数值, SYSWDG 在启动时或被清除后, SYSWDGLOA 的设定值被载入该寄存器, 然后在外设时钟的驱动下进行递减计数。

15.5.3 SYSWDG 控制寄存器 SYSWDGCTRL

(地址: 0x4000_2008)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	DBGE	RSTE	INTE
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] INTE: 系统看门狗使能控制**
0b: SYSWDG 模块禁止, 无中断响应
1b: SYSWDG 模块使能, 同时 SYSWDG 中断被使能 (NVIC 中断还是需另外开启)
- **位[1] RSTE: 系统看门狗复位控制**
0b: SYSWDG 复位禁止
1b: SYSWDG 复位使能
- **位[2] DBGE: 系统看门狗调试挂起控制**
0b: SYSWDG 在调试暂停时被挂起
1b: SYSWDG 在调试暂停时仍旧工作
- **位[31:3]: 保留**

15.5.4 SYSWDG 中断清除寄存器 SYSWDGINTCLR

(地址: 0x4000_200C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

该寄存器为只写，对其进行任意数据的写操作都会将 SYSWDGLOAD 的值重载入 SYSWDGVALUE 寄存器内重新启动递减计数，同时清零 SYSWDG 中断标志（如果已经被置 1）

应用软件在设定并启动 SYSWDG 模块后，必须在 SYSWDG 计数归零前对该寄存器进行一次写操作以复位 SYSWDG

15.5.5 SYSWDG 原始中断标志寄存器 SYSWDGRIS

(地址: 0x4000_2010)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RIF
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] RIF: **SYSWDG 原始中断标志信息位**
 0b: SYSWDG 未发生归零
 1b: SYSWDG 发生归零，写 SYSWDGCLR 将其清 0
- 位[31:1]: **保留**

15.5.6 SYSWDG 掩蔽中断标志寄存器 SYSWDGMIS

(地址: 0x4000_2014)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MIF
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] MIF: SYSWDG 掩蔽中断标志信息位**

该位只读

该位为 SYSWDG 计数归零中断原始标志位 RIF 经 INTE 掩蔽后的状态, 如果 RIF 和 INTE 同时为 1, 则 MIF 为 1, 引发 SYSWDG 中断; 否则为 0, 不产生 SYSWDG 中断。

对 SYSWDGINTCLR 寄存器进行一次写入后清除 RIF, 亦同时清楚了 MIF

0b: SYSWDG 掩蔽中断标志无效

(可能是实际无中断, 或 SYSWDGCTRL 的 INTE 未使能)

1b: SYSWDG 掩蔽中断标志有效, 写 SYSWDGINTCLR 将其清 0

- **位[31:1]: 保留**

15.5.7 SYSWDG 锁定控制寄存器 SYSWDGLOCK

(地址: 0x4000_2400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	LOCK
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

该寄存器提供了一种 SYSWDG 系统安全锁定的机制, 可避免在系统运行过程中, 因软件设计缺陷或程序跑飞时意外篡改 SYSWDG 模块的配置. 一旦 SYSWDG 模块被锁定, 本模块所有的寄存器都无法被软件改写。

对 SYSWDGLOCK 寄存器写入一个特殊值“0x1ACCE551”后, 将对 SYSWDG 模块解锁, 解锁后软件可以自由配置相关的寄存器; 对该寄存器写入任意其它值后, SYSWDG 模块即

被锁定，模块内任何可配置的寄存器都无法被改写。

任何时候读此寄存器时，仅最低位有意义，指示 SYSWDG 的锁定状态。

- **位[0] LOCK:** **SYSWDG 锁定标志位**
 - 0b: SYSWDG 模块未锁定，模块内相关寄存器可改写
 - 1b: SYSWDG 模块锁定，模块内所有寄存器不可被改写
- **位[31:1]:** **保留**

16 独立看门狗

16.1 独立看门狗综述

独立看门狗模块可以用于解决处理器因为硬件或软件的故障所发生的错误。它工作于片内低速振荡时钟32kHz，因此即使是主时钟失效时它仍然照常工作。适用于需要看门狗作为一个在主程序之外，能够完全独立工作，并且对时间精度要求较低场合。

16.2 独立看门狗功能描述

独立看门狗模块为一个32位宽的递减计数器。独立看门狗计数器启动时由重载设定寄存器（IWDGLOAD）载入初值，然后在时钟的驱动下每个周期进行递减计数。

独立看门狗计数器启动后，递减计数值第一次归零时，将触发IWDG中断响应。第一次归零后，IWDG重新载入初值，继续递减计数，当计数值第二次归零，且IWDGCTRL寄存器的RSTE位设为1，且之前的中断未清除的情况下，将立即引发一次系统硬件复位，即独立看门狗复位。

16.3 独立看门狗软件操作说明

16.3.1 启动独立看门狗

上电复位后独立看门狗模块默认处于禁止状态，应用软件根据需求写IWDGLOAD寄存器设定计数器初始值，然后设置IWDGCTRL寄存器的INTE位为1启动IWDG开始工作。

16.3.2 独立看门狗中断

应用软件可以编写自己的中断服务程序来应对IWDG的中断事件，如果发生IWDG中断，一般意味着应用软件已有过长的间隔时间没有执行清除IWDG工作，可用于软件调试阶段的可靠性诊断。正式发布的应用程序不能在自己的中断服务程序中执行清除独立看门狗中断标志指令，否则独立看门狗将永远不会引发系统复位，失去了其系统监控的作用。

应用软件在正常运行时，必须在设定的IWDG定时时间范围内对IWDGINTCLR寄存器写一次任意数，以重载复位IWDG计数器。正常情况下不应出现IWDG中断，更不能出现IWDG复位

由于IWDG工作在内部低速振荡32kHz时钟下，和系统时钟之间存在同步逻辑，当IWDG计数重载寄存器（IWDGLOAD）和控制寄存器（IWDGCTRL）在apb总线上操作更新时，IWDG将在两个32kHz时钟周期后更新数据。

当IWDG在内部低速32kHz时钟下计数，同步逻辑会先锁住IWDG上计数器的值，然后再同步PCLK，以供CPU能读取独立看门狗值寄存器。独立看门狗清除中断寄存器也需要2个32kHz时钟周期和2个PCLK时钟同步，软件必须小心处理。

16.4 寄存器列表

地址	寄存器	描述	备注
0x4000_3000	IWDGLOAD	IWDG 计数重载寄存器	IWDGLOAD 说明
0x4000_3004	IWDGVALUE	IWDG 计数值寄存器	IWDGVALUE 说明
0x4000_3008	IWDGCTRL	IWDG 控制寄存器	IWDGCTRL 说明
0x4000_300C	IWDGINTCLR	IWDG 中断清除寄存器	IWDGINTCLR 说明
0x4000_3010	IWDGRIS	IWDG 原始中断标志寄存器	IWDGRIS 说明
0x4000_3014	IWDGMIS	IWDG 掩蔽中断标志寄存器	IWDGMIS 说明
0x4000_3400	IWDGLOCK	IWDG 锁定控制寄存器	IWDGLOCK 说明

16.5 寄存器描述
16.5.1 IWDG 计数重载寄存器 IWDGLOAD

(地址: 0x4000_3000)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	IWDGLOAD															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IWDGLOAD															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- 位[31:0] IWDGLOAD: 独立看门狗重载初始值

该值决定了 IWDG 的定时时间，计算 IWDGLOAD 重载值的公式为：

$$IWDGLOAD = T_{WDT} * f_{CLK} - 1$$

其中：

 T_{WDT} 为期望的 IWDG 定时时间，单位为 ms

 f_{CLK} 为 IWDG 的时钟频率，为 32kHz

例如，若应用软件希望设置约 20ms 的独立看门狗定时时间，则按上述公式算得 IWDGLOAD 应赋值为 639，但必须注意 IWDG 所用的片内 32kHz 时钟的误差范围为 ±20%，软件应考虑极限情况下时钟频率最高可达 38.4kHz，所以为保证任何情况下达到至少 20ms 的独立看门狗定时时间，IWDGLOAD 应赋值为 767

16.5.2 IWDG 计数值寄存器 IWDGVALUE

(地址: 0x4000_3004)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	IWDGVALUE															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	IWDGVALUE															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- **位[31:0] IWDGVALUE:** **独立看门狗当前计数值**
IWDG 的递减计数器的当前计数值, IWDG 在启动时或被清除后, IWDGLOAD 的设定值被载入该寄存器, 然后在 IWDG 时钟的驱动下进行递减计数。

16.5.3 IWDG 控制寄存器 IWDGCTRL

(地址: 0x4000_3008)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	DBGE	RSTE	INTE
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] INTE:** **独立看门狗使能控制**
0b: IWDG 模块禁止, 无中断响应
1b: IWDG 模块使能, 同时 IWDG 中断被使能 (NVIC 中断还是需另外开启)
- **位[1] RSTE:** **独立看门狗复位控制**
0b: IWDG 复位禁止
1b: IWDG 复位使能
- **位[2] DBGE:** **独立看门狗调试挂起控制**
0b: IWDG 在调试暂停时被挂起
1b: IWDG 在调试暂停时仍旧工作
- **位[31:3]:** **保留**

16.5.4 IWDG 中断清除寄存器 IWDGINTCLR

(地址: 0x4000_300C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

该寄存器为只写,对其进行任意数据的写操作都会将 IWDGLOAD 的值重载入 IWDGVALUE 寄存器内重新启动递减计数,同时清零 IWDG 中断标志(如果已经被置 1)
 应用软件在设定并启动 IWDG 模块后,必须在 IWDG 计数归零前对该寄存器进行一次写操作以复位 IWDG

16.5.5 IWDG 原始中断标志寄存器 IWDGRIS

(地址: 0x4000_3010)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	RIF
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] RIF: IWDG 原始中断标志信息位
 0b: IWDG 未发生归零
 1b: IWDG 发生归零, 写 IWDGCLR 将其清 0
- 位[31:1]: 保留

16.5.6 IWDG 掩蔽中断标志寄存器 IWDGMIS

(地址: 0x4000_3014)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	MIF
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] MIF: IWDG 掩蔽中断标志信息位**

该位只读

该位为 IWDG 计数归零中断原始标志位 RIF 经 INTE 掩蔽后的状态, 如果 RIF 和 INTE 同时为 1, 则 MIF 为 1, 引发 IWDG 中断; 否则为 0, 不产生 IWDG 中断。

对 IWDGINTCLR 寄存器进行一次写入后清除 RIF, 亦同时清楚了 MIF

0b: IWDG 掩蔽中断标志无效

(可能是实际无中断, 或 IWDGCTRL 的 INTE 未使能)

1b: IWDG 掩蔽中断标志有效, 写 IWDGINTCLR 将其清 0

- **位[31:1]: 保留**

16.5.7 IWDG 锁定控制寄存器 IWDGLOCK

(地址: 0x4000_3400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	LOCK
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	R/W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

该寄存器提供了一种 IWDG 系统安全锁定的机制, 可避免在系统运行过程中, 因软件设计缺陷或程序跑飞时意外篡改 IWDG 模块的配置. 一旦 IWDG 模块被锁定, 本模块所有的寄存器都无法被软件改写。

对 IWDGLOCK 寄存器写入一个特殊值“0x1ACCE551”后, 将对 IWDG 模块解锁, 解锁后软件可以自由配置相关的寄存器; 对该寄存器写入任意其它值后, IWDG 模块即被锁定,

模块内任何可配置的寄存器都无法被改写。

任何时候读此寄存器时，仅最低位有意义，指示 IWDG 的锁定状态。

- **位[0] LOCK:** **IWDG 锁定标志位**
 - 0b: IWDG 模块未锁定，模块内相关寄存器可改写
 - 1b: IWDG 模块锁定，模块内所有寄存器不可被改写
- **位[31:1]:** **保留**

17 UART

17.1 UART 模块综述

PT32L031 芯片内集成了 2 个基本功能完全一致但相互独立的 UART 通讯模块，分别为 UART0、UART1。

下文叙述中将其统一描述成 UARTn (n=0, 1)。应用系统可以利用这 2 个 UART 模块实现和外界的异步数据通讯（例如 RS232、RS485 等）。

UARTn 模块包含一个发送器 FIFO (TX FIFO) 和一个 8 接收器 FIFO (RX FIFO)。支持多种类型的中断。UARTn 有一个可编程的波特率发生器，能对 PCLK 进行分频以产生 UARTn 发送器和接收器所需的波特率时钟。

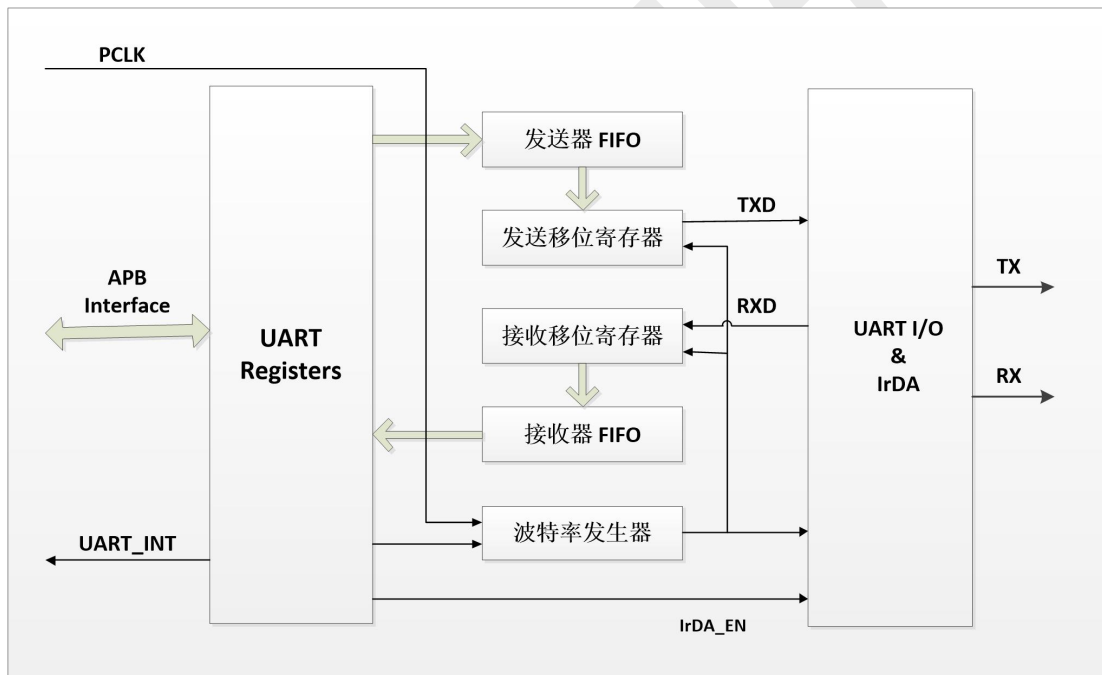


图 17-1: UART 框图

17.2 UART 主要特性

- 全双工通信能力
- 灵活可编程的波特率设定, 支持标准 9600bps、19200bps、28800bps、38400bps、57600bps、115.2Kbps 等, 或其它特殊应用的波特率
- 可编程的数据传输格式:
 - ✓ 8 位数据、7 位数据+1 位奇偶校验、8 位数据+1 位奇偶校验、9 位数据
 - ✓ 可编程奇或偶校验方式
 - ✓ 可编程设定 0.5 位、1 位、1.5 位或 2 位停止位
- 发送器 FIFO: 缓冲深度 8 级;
- 接收器 FIFO: 缓冲深度 8 级;
- 错误检测: 奇偶校验错误、溢出错误和帧错误

- 可选发送数据极性
- 支持单线半双工通信
- 支持 ISO7816 通讯

17.3 UART 功能概述

UART 的数据收发需要使用相同的数据帧结构和传输波特率。通过 TX/RX 引脚实现 UART 的全双工通信。

- TX: 数据发送引脚。在单线和 ISO7816 模式下，该引脚被用于数据的发送和接收。
- RX: 数据接收引脚

UART 模块提供片内数据发送端至接收端的回绕功能，无需外部硬件电路和实际线路连接，即可方便地进行软件自诊断。

使用时需注意：

- 只有当波特率发生器的控制位 (RUN) 被置 1 后，才能进行串行数据的收发。如果 RUN 位被清 0，则数据收发过程将立即终止，TXD 引脚输出始终保持空闲态。除非应用程序确定 UART 通讯处于空闲状态，不然 RUN 位应始终被置为 1。
- 如果将控制模式 (UARTn_CTL.MODE) 配置成某一个未定义的保留模式，UART 模块的工作将不可预测
- 在 9 位数据模式，或 8 位数据加唤醒位模式时，必须屏蔽奇偶校验错误中断以防止假的奇偶校验错误。因为在这两种配置模式下数据帧内没有奇偶校验位信息。
- 软件若在 UART 工作时需要切换模式，需在 UART 为 idle 时切换，并在每次切换完模式之后主动写一次发送队列复位寄存器和接收队列复位寄存器以清空发送队列和接收队列，保证 UART 在新的模式下能够正常收发数据。

17.4 UART 数据格式

串行数据帧格式：

- 总线在发送和接收前处于空闲状态
- 1 个起始位
- LSB 优先的数据位 (7~9)
- 可选的 1 位校验位
- 停止位 (0.5、1、1.5 或 2 位)

UART 工作模式分 8 位、9 位数据收发模式。

8 位数据帧有两种组成形式：

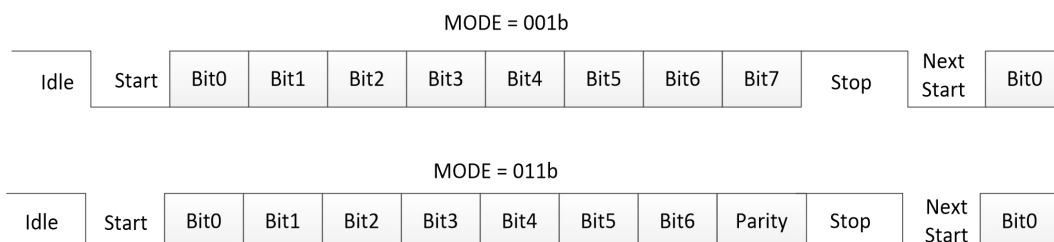
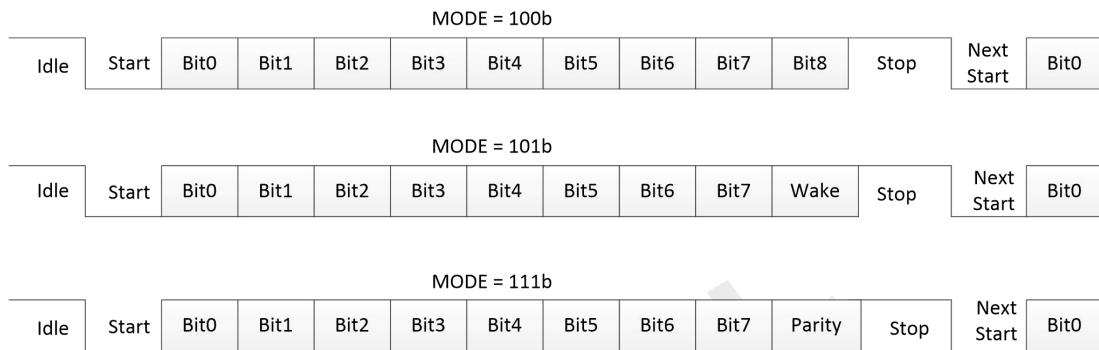


图 17-2: UART 8 位数据帧格式 (普通极性)

9 位数据帧有三种组成形式:


图 17-3: UART 9 位数据帧格式 (普通极性)

串行数据发送的极性是可配置的, 普通极性时, 空闲状态时 TX 引脚为高电平, 起始位电平与数据线空闲状态相反, 停止位与数据线空闲状态相同, 数据位 0 为低电平, 1 为高电平; 反相极性时, 所有电平和普通极性相反。提供反相发送模式可以方便不同电压系统间通讯接口的设计, 亦可方便实现开漏驱动型通讯。

发送校验位的产生和检测可以是奇校验或偶校验, 取决于 PBR 位的设定。设为奇校验时, 在数据字和奇偶校验位中共有奇数个逻辑“1”被发送或被检测到; 设为偶校验时, 在数据字和奇偶校验位中共有偶数个逻辑“1”被发送或被检测到。接收时, 校验位和有效数据位一起被存入接收缓冲 UARTn_RXB 队列中。

当 UART 处于传输唤醒模式, 所有接收到的第 9 位为“0”的数据帧都将被忽略; 直到接收到的数据帧其第 9 位为 1 时, 接收结果才会被存入 RX FIFO 缓冲队列。一旦检测到第一个第 9 位为“1”的数据并存入 RX FIFO 时, UARTn_ST 寄存器中的 RXNE 标志位将被置位, 如果 UARTn_IE 寄存器中的 RXNE_EN 位置 1, 将产生中断。这一特性可应用于主-从结构多点通信。当主机需要与某一指定地址的从机时, 它先发送 8 位有效地址外加第 9 位为 1 的一个 9 位寻址帧, 所有从机都能收到这个的寻址帧, 并和本机设定地址作比对, 如果地址相符, 则立即将本机的 UART 工作模式转为 9 位数据模式, 继续接收后面的数据帧 (数据帧第 9 位为 0), 地址不符的从机则不会响应该数据帧, 可以提高系统运行效率。

17.5 UART 波特率计算和设定

UART 通信过程中, 发送器和接收器的波特率需设置相同值, UART 内嵌的可编程波特率发生器, 通过 RUN 位控制。软件通过配置波特率控制寄存器 UARTn_BR 设置 UART 通信的波特率, 波特率与 UART 时钟 PCLK 关系如下:

$$\text{波特率} = f_{\text{SYS}} / (16 / (2^{\text{SR}}) * \text{BR}) \quad \text{式①}$$

$$\text{BR} = f_{\text{SYS}} / (16 / (2^{\text{SR}}) * \text{波特率}) \quad \text{式②}$$

其中:

- 式①的波特率为实际波特率, 式②的波特率为预期波特率

- f_{SYS} 为 PCLK 频率
- BR 作为系统时钟分频
- SR 用于控制每位码元的采样率
- 该式中 SR 取值为 0/1/2, 分别代表 16/8/4 倍码元采样率

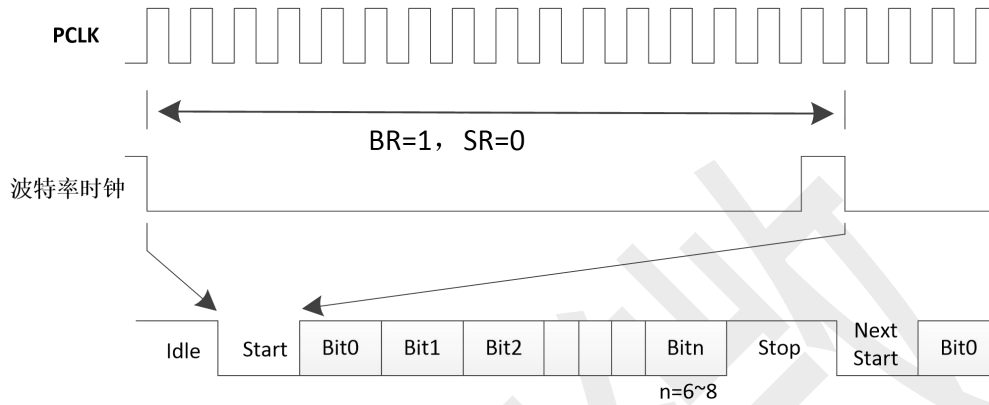


图 17-4: UART 时钟和数据帧时序

以下列出了在不同系统时钟频率下常用的波特率设定值和误差 (SR=00b), 可供应用时快速参考。

表 17-1.常用的波特率设定和误差 (SR=00b)

波特率	$f_{CLK} = 24MHz$		$f_{CLK} = 16MHz$		$f_{CLK} = 8MHz$		$f_{CLK} = 4MHz$		$f_{CLK} = 2MHz$		$f_{CLK} = 1MHz$	
	BR	误差	BR	误差	BR	误差	BR	误差	BR	误差	BR	误差
115200	0x000D	0.16%	0x0009	-3.55%	0x0004	8.51%	0x0002	8.51%	0x0001	8.51%	0x0001	-45.75%
38400	0x0027	0.16%	0x001A	0.16%	0x000D	0.16%	0x0007	-6.99%	0x0003	8.51%	0x0002	-18.62%
28800	0x0034	0.16%	0x0023	-0.79%	0x0011	2.12%	0x0009	-3.55%	0x0004	8.51%	0x0002	8.51%
19200	0x004E	0.16%	0x0034	0.16%	0x001A	0.16%	0x000D	0.16%	0x0007	-6.99%	0x0003	8.51%
9600	0x009C	0.16%	0x0068	0.16%	0x0034	0.16%	0x001A	0.16%	0x000D	0.16%	0x0007	-6.99%
4800	0x0139	-0.16%	0x00D0	0.16%	0x0068	0.16%	0x0034	0.16%	0x001A	0.16%	0x000D	0.16%
2400	0x0271	0%	0x01A1	-0.08%	0x00D0	0.16%	0x0068	0.16%	0x0034	0.16%	0x001A	0.16%
1200	0x04E2	0%	0x0341	0.04%	0x01A1	-0.08%	0x00D0	0.16%	0x0068	0.16%	0x0034	0.16%

表 17-2.常用的波特率设定和误差 (SR=01b)

波特率	$f_{CLK} = 24MHz$		$f_{CLK} = 16MHz$		$f_{CLK} = 8MHz$		$f_{CLK} = 4MHz$		$f_{CLK} = 2MHz$		$f_{CLK} = 1MHz$	
	BR	误差	BR	误差	BR	误差	BR	误差	BR	误差	BR	误差
115200	0x001A	0.16%	0x0011	2.12%	0x0009	-3.55%	0x0004	8.51%	0x0002	8.51%	0x0001	8.51%
38400	0x004E	0.16%	0x0034	0.16%	0x001A	0.16%	0x000D	0.16%	0x0007	-6.99%	0x0003	8.51%
28800	0x0068	0.16%	0x0045	0.64%	0x0023	-0.79%	0x0011	2.12%	0x0009	-3.55%	0x0004	8.51%
19200	0x009C	0.16%	0x0068	0.16%	0x0034	0.16%	0x001A	0.16%	0x000D	0.16%	0x0007	-6.99%

9600	0x00139	-0.16%	0x00D0	0.16%	0x0068	0.16%	0x0034	0.16%	0x001A	0.16%	0x000D	0.16%
4800	0x0271	0%	0x01A1	-0.08%	0x00D0	0.16%	0x0068	0.16%	0x0034	0.16%	0x001A	0.16%
2400	0x04E2	0%	0x0341	0.04%	0x001A1	-0.08%	0x00D0	0.16%	0x0068	0.16%	0x0034	0.16%
1200	0x09C4	0%	0x0683	-0.02%	0x0341	0.04%	0x01A1	-0.08%	0x00D0	0.16%	0x0068	0.16%

表 17-3.常用的波特率设定和误差 (SR=10b)

波特率	f _{CLK} = 24MHz		f _{CLK} = 16MHz		f _{CLK} = 8MHz		f _{CLK} = 4MHz		f _{CLK} = 2MHz		f _{CLK} = 1MHz	
	BR	误差	BR	误差	BR	误差	BR	误差	BR	误差	BR	误差
115200	0x0034	0.16%	0x0023	-0.79%	0x0011	2.12%	0x0009	-3.55%	0x0004	8.51%	0x0002	8.51%
38400	0x009C	0.16%	0x0068	0.16%	0x0034	0.16%	0x001A	0.16%	0x000D	0.16%	0x0007	-6.99%
28800	0x00D0	0.16%	0x008B	-0.08%	0x0045	0.64%	0x0023	-0.79%	0x0011	2.12%	0x0009	-3.55%
19200	0x0139	-0.16%	0x00D0	0.16%	0x0068	0.16%	0x0034	0.16%	0x001A	0.16%	0x000D	0.16%
9600	0x0271	0%	0x01A1	-0.08%	0x00D0	0.16%	0x0068	0.16%	0x0034	0.16%	0x001A	0.16%
4800	0x04E2	0%	0x0341	0.04%	0x01A1	-0.08%	0x00D0	0.16%	0x0068	0.16%	0x0034	0.16%
2400	0x09C4	0%	0x0683	-0.02%	0x00341	0.04%	0x01A1	-0.08%	0x00D0	0.16%	0x0068	0.16%
1200	0x1388	0%	0x0D05	0.01%	0x0683	-0.02%	0x0341	0.04%	0x01A1	-0.08%	0x00D0	0.16%

波特率设定，根据式②计算得到 UARTn_BR 的设定值，并根据式①确认实际波特率，为了数据收发的可靠性，误差不得超过 1%。使用特殊频率的系统时钟，可以得到较高的波特率精度。

17.6 UART 数据发送

在 RUN 位置 1，往发送缓冲寄存器 UARTn_DAT 写入一个数据后，发送器根据设置的数据帧格式，启动数据的串行发送过程。

UARTn 包含一个 8 级发送 FIFO 缓冲器，软件可以连续往 UARTn_DAT 寄存器写入，直至发送 FIFO 队列满（发送缓冲队列满标志 TXF 为 1），这样可实现多个数据帧的连续发送，提高发送效率，降低 CPU 的干预度。当最后一个数据帧的最后 1 位被实际发送出去后，发送器空闲状态标志 TXEND 被置 1，告知所有数据的串行发送已完成。

配置步骤：

1. 配置控制寄存器 UARTn_CTL 配置所需要的数据帧格式（字长/校验/停止位/极性）
2. 在 UARTn_BR 配置要求的波特率
3. UARTn_CTL 的 RUN 位置 1，启动传输
4. 在 UARTn_DR 写入要发送的数据
5. 根据 UARTn_ST 的轮询当前 UART 状态，可根据需要重复步骤 4

17.7 UART 数据接收

在 RUN & RXEN 置 1 时，UART 模块准备接收串行数据。当 RXD 引脚检测到起始位，开始数据帧的接收过程，根据 UARTn_CTL 配置所需要的数据帧格式接收 8 或者 9 位数据位，当接收到最后一位停止位后，即认为一个完整的数据帧接收成功，UART 模块将收到的数据存入 RX FIFO，队列非空标志 RXNE 置 1，但不会更新校验错误标志 PERR 和帧错误标志 FERR（接收数据校验标志将在数据被从接收队列中读出时更新），并可通过应用软件读出。

UARTn 包含一个深度为 8 级的接收 FIFO 缓冲器，UART 可以继续接收后面的数据存入接收 FIFO，直至接收 FIFO 队列满，而不需要立即读回。在接收队列满（接收缓存队列满标志 RXF 为 1）后如果又有新的数据被收到，则将发生队列溢出的错误，并将状态标志位 OVERR 置 1，此时最新接收的数据将被丢弃不会存入接收 FIFO，UARTn_ST 寄存器被相应更新以反映该次新数据的接收无效。多级的接收 FIFO 缓冲，应用软件可以连续读取多个数据（只要队列中有数据），提高了 CPU 的运行效率。

配置步骤：

1. 配置控制寄存器 UARTn_CTL 配置所需要的数据帧格式（字长/校验/停止位/极性）
2. 在 UARTn_BR 配置要求的波特率
3. UARTn_CTL 的 RUN 位置 1，RXEN 位置 1，开始检测起始位
4. 当接收到数据，UARTn_ST 寄存器被相应更新

如果在数据帧接收过程中软件清除 RXEN 位，当前正在接收的数据帧依然可以被完整的接收，并更新各状态标志。但后续的数据将不能被接收到。

17.8 UART 错误检测机制

UART 模块提供了通信错误检测机制，确保了串行数据通信的可靠性，下列事件发生时，若相应的中断使能位被置位，UART 模块可产生一个中断：

- 接收缓冲队列溢出：在接收队列满后，又有新的数据被收到
- 奇偶校验错误：接收到的字符没有正确的奇偶校验位
- 帧接收错误：接收字符没有有效停止位

当应用软件从接收队列中读取暂存的接收数据时，每读取一个数据后，其对应的校验信息将同步反映在 UARTn_ST 状态寄存器中。

17.9 UART 中断响应

UART 的各种中断事件被连接到同一个中断向量(见图 17-5)，有以下各种中断事件：

- 发送期间：发送缓冲队列全空中断、发送缓冲队列半空中断、发送缓冲队列全满中断、发送全部完成中断。
- 接收期间：接收缓冲队列非空中断、接收缓冲队列半满中断、接收缓冲队列全满中断、接收缓冲队列尚未清空超时中断、接收队列全空超时中断、接收缓冲队列溢出中断、奇偶校验错误中断、帧错误中断。

如果对应的使能控制位被设置，这些事件就会产生各自的中断，如下图：



图 17-5: UART 中断映射图

中断标志的清除方式按不同的标志而不同：错误中断标志 PERR、FERR 和 OVERR 则只能通过由软件对中断状态寄存器 UARTn_ST 的对应位写 1 来清 0；其余中断标志由硬件自动清 0。

17.10 UART 红外通讯功能

UART0 支持串行数据红外调制发送，实现了 38KHz 红外载波数据发送调制功能，UART0 模块在发送数据位 0 时，可产生经调制后的 38KHz 波形驱动片外红外发色管。

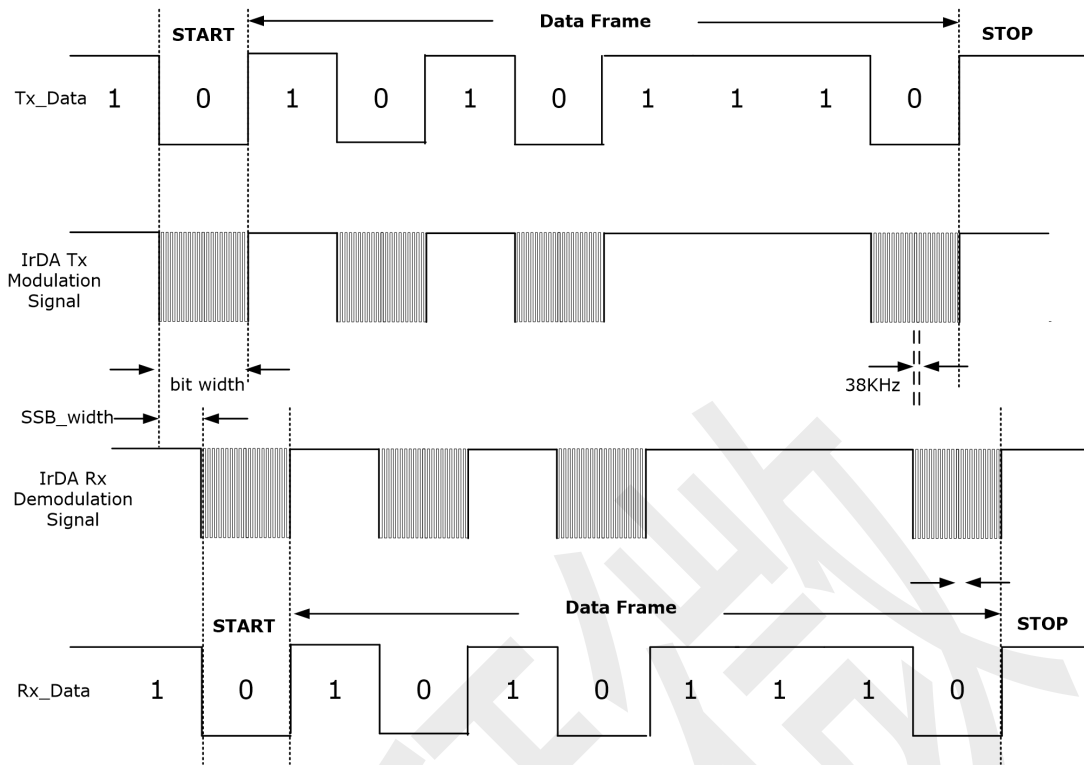


图 17-6: IrDA 调制与解调(普通极性)

为应对片外红外发射驱动电路的不同设计，在 UART 红外调制极性是可以配置的：

- UART0_IRC.TXPOL=0：空闲或发送数据“1”时输出高电平；发送数据0位时输出38KHz调制波形，适用于外部P管驱动。
- UART0_IRC.TXPOL=1：空闲或发送数据“1”时输出低电平；发送数据0位时输出38KHz调制波形，适用于外部N管驱动。

UART0 在红外收发模式下，支持 600bps、1200bps 和 2400bps 三种波特率，38KHz 调制载波的 PWM 脉宽调制，可通过配置 UART_IR_PWMC 寄存器来产生不同占空比的载波输出。PWM 调制精度基于系统时钟周期，具体载波调制方式参见 UART_IR_PWMC 寄存器描述。

17.11 UART ISO7816 功能

UART 模块支持 ISO7816 功能。当 UARTn_ISO7816 EN 使能后，即进入 ISO7816 模式，相关引脚将被重新分配。通过 TxD/RxD 引脚实现与智能卡通信。

- TxD: ISO7816DAT 的发送和接收脚。在接收模式下 TxD 脚是输入端口，可以接收输入，并送入 RX FIFO；在发送模式下，TxD 切换为输出端口，UARTn 模块的 TX FIFO 数据通过该引脚串行输出。
- RxD: 串行时钟信号 ISO7816CLK 输出。

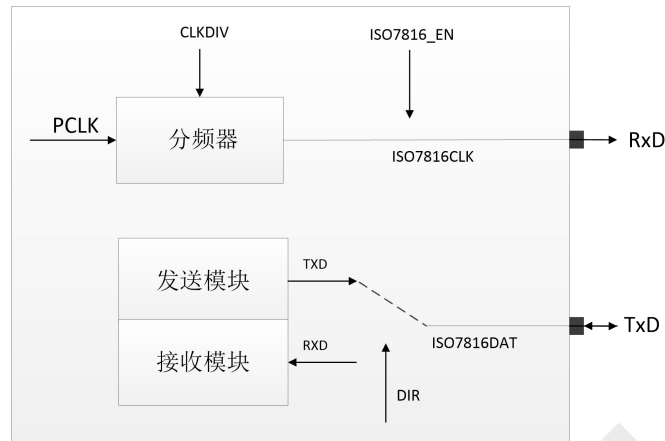


图 17-7: UART ISO7816 模块功能示意图

按照 ISO7816 协议，UARTn 应该被设置为 MODE = 111b、奇偶校验位、停止位，以满足“1 位起始位、8 位数据位、1 位校验位以及 1-2 位停止位”的串行数据收发时序要求。其基本接口时序如下图 17-8 所示。其中：Start 代表帧起始位；Byte_i 代表 8 位数据；Parity 代表奇偶校验位；Pause 代表帧间隙时段；Error Single 为错误应答时段。

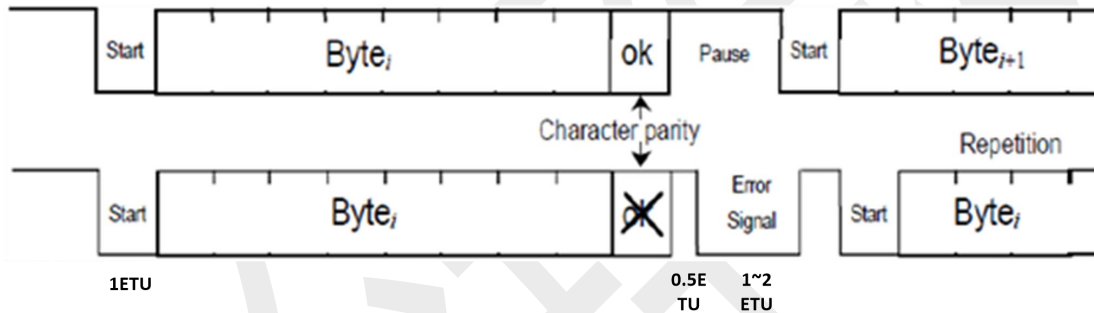


图 17-8: ISO7816 接口时序

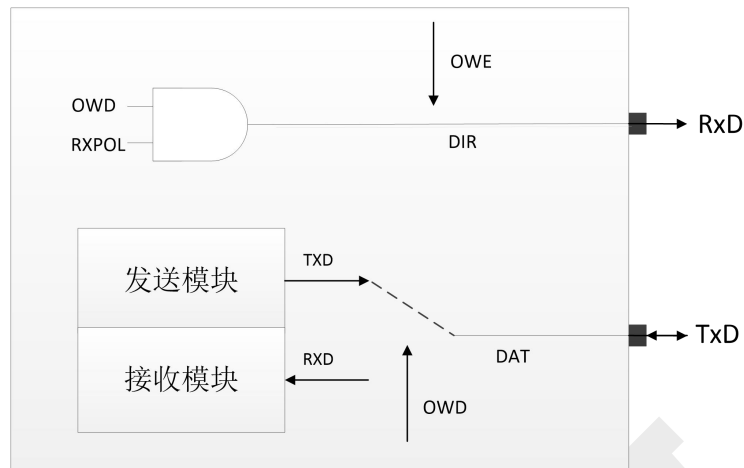
UART 模块在硬件层面上提供了 ISO7816 功能所需的数据单线双向通讯以及 1-5MHz 时钟输出。应用软件必须按所选时钟频率，正确配置 UARTn 的波特率，使串行通讯的码元时间宽度满足 ISO7816 所规定的 ETU 计算标准。

在与智能卡通信过程中，UART 的 TxD 引脚应配置成开漏，在发现校验位错误的情况下，0.5 位停止位结束后，接收器可以将数据线拉低。但 ISO7816 协议所要求的错误应答检测、重复发送以及额外的帧间隙时间（Guard-time）控制，则需要应用软件配合才能实现。

17.12 UART 单线半双工通讯模式

UART 模块可配置为单线半双工的通讯模式。软件可通过配置 UARTn_CTR 寄存器的 OWE 位来选择 UART 的通讯模式，在单线半双工模式下，相关引脚将被重新分配。

- TxD: 用于接收/发送数据。OWD 位为 1 时，TxD 管脚用于发送数据；OWD 位为 0 时，TxD 管脚用于接收数据。
- RxD: 输出数据的传输方向。数据的传输方向由传输数据的极性 RXPOL 位和单线半双工通讯模式下的数据传输方向 OWD 位决定。


图 17-9: UART 单线半双工通讯模式功能示意图
17.13 寄存器列表

地址	寄存器	描述	备注
0x4000_4400	UART0_DAT	UART0 收发数据寄存器	UART0_DAT 说明
0x4000_4404	UART0_CTL	UART0 模块控制寄存器	UART0_CTL 说明
0x4000_4408	UART0_BR	UART0 波特率控制寄存器	UART0_BR 说明
0x4000_440C	UART0_IE	UART0 中断使能控制寄存器	UART0_IE 说明
0x4000_4410	UART0_ST	UART0 接收状态寄存器	UART0_ST 说明
0x4000_4410	UART0_STCLR	UART0 状态清除寄存器	UART0_STCLR 说明
0x4000_4414	UART0_GT	UART0 帧间隔时间寄存器	UART0_GT 说明
0x4000_4418	UART0_TO	UART0 超时控制寄存器	UART0_TO 说明
0x4000_441C	UART0_TXFR	UART0 发送复位寄存器	UART0_TXFR 说明
0x4000_4420	UART0_RXFR	UART0 接收复位寄存器	UART0_RXFR 说明
0x4000_4424	UART0_ISO7816	UART0 ISO7816 收发控制寄存器	UART0_ISO7816 说明
0x4000_4428	UART0_IR_CTL	UART0 红外发送解调控制寄存器	UART0_IR_CTL 说明
0x4000_442C	UART0_IR_PWMC	UART0 红外发送占空比控制寄存器	UART0_IR_PWMC 说明
0x4001_3800	UART1_DAT	UART1 收发数据寄存器	UART1_DAT 说明
0x4001_3804	UART1_CTL	UART1 模块控制寄存器	UART1_CTL 说明
0x4001_3808	UART1_BR	UART1 波特率控制寄存器	UART1_BR 说明
0x4001_380C	UART1_IE	UART1 中断使能控制寄存器	UART1_IE 说明
0x4001_3810	UART1_ST	UART1 接收状态寄存器	UART1_ST 说明
0x4001_3810	UART1_STCLR	UART1 状态清除寄存器	UART1_STCLR 说明
0x4001_3814	UART1_GT	UART1 帧间隔时间寄存器	UART1_GT 说明
0x4001_3818	UART1_TO	UART1 超时控制寄存器	UART1_TO 说明
0x4001_381C	UART1_TXFR	UART1 发送复位寄存器	UART1_TXFR 说明
0x4001_3820	UART1_RXFR	UART1 接收复位寄存器	UART1_RXFR 说明
0x4001_3824	UART1_ISO7816	UART0 ISO7816 收发控制寄存器	UART1_ISO7816 说明

17.14 寄存器描述

17.14.1 收发数据 FIFO 缓冲寄存器 UARTn_DAT

(地址: 0x4000_2400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	DATA								
R/W	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

UARTn_DAT 是接收和发送数据 FIFO 缓冲队列，深度为 8 级。其中 n 为 UART 模块编号，取值为 0 或 1。

软件对 UARTn_DAT 写入数据即为往 TX-FIFO 内写入数据，有效位数定义如下：

9 位数据帧格式下 (UARTn_CTL 寄存器的 mode 为 3'b100 或 3'b101)，写 UARTn_DAT 低 9 位有效

8 位数据帧格式下 (UARTn_CTL 寄存器的 mode 为 3'b001 或 3'b111)，写 UARTn_DAT 低 8 位有效

7 位数据帧格式下 (UARTn_CTL 寄存器的 mode 为 3'b011)，写 UARTn_DAT 低 7 位有效

软件对 UARTn_DAT 读取数据即为从 RX-FIFO 内读出数据，有效位数定义如下：

9 位数据帧格式下 (UARTn_CTL 寄存器的 mode 为 3'b100 或 3'b101)，读 UARTn_DAT 低 9 位为有效数据

8 位数据帧格式下 (UARTn_CTL 寄存器的 mode 为 3'b001 或 3'b111)，读 UARTn_DAT 低 8 位为有效数据

7 位数据帧格式下 (UARTn_CTL 寄存器的 mode 为 3'b011)，读 UARTn_DAT 低 7 位为有效数据

- **位[8] RXEN:** **数据接收使能**
 0b: 禁止数据接收
 1b: 允许数据接收
- **位[10:15]:** **保留**
- **位[16] RXPOL:** **URAT 数据接收极性控制**
 0b: 标准数据极性（空闲为高电平，起始位为低电平；数据 1=高电平，0=低电平）
 1b: 反相数据极性（空闲为低电平，起始位为高电平；数据 1=低电平，0=高电平）
- **位[17] TXPOL:** **URAT 数据发送极性控制**
 0b: 标准数据极性（空闲为高电平，起始位为低电平；数据 1=高电平，0=低电平）
 1b: 反相数据极性（空闲为低电平，起始位为高电平；数据 1=低电平，0=高电平）
- **位[20:18]:** **保留**
- **位[21] OWE:** **URAT 单线通讯模式使能控制**
 0b: 标准两线全双工模式，模块对应的 TX 引脚为数据发送，RX 引脚为数据接收
 1b: 单线半双工模式，模块对应的 TX 引脚为数据接收或发送，RX 引脚则为发送/接收状态指示（见下面 OWD 位说明）
- **位[22] OWD:** **URAT 单线通讯接收/发送方向设置**
 该位只有在 OWE 置为 1 使能单线通讯模式时才有意义
 0b: 单线模式下模块 TX 引脚为串行数据接收
 1b: 单线模式下模块 TX 引脚为串行数据发送
 单线通讯模式下 OWD 位所配置的接收或发送转态将同步反映在模块的 RX 引脚上，可用于外围通讯电路的方向控制。其真值表为：

RX 外设功能	OWE 位	OWD 位	RXPOL 位	RX 引脚
1	1	0	0	0
1	1	1	0	1
1	1	0	1	1
1	1	1	1	0
1	0	x	x	串行数据接收
0	x	x	x	数字 GPIO

- **位[31:30] 保留**

17.14.3 波特率控制寄存器 UARTn_BR

(地址: 0x4000_2400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	SR	
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	BR															
R/W	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

- **位[15:0] BR:** **波特率分频系数**
该值的设定使用方法请参考 [UART 波特率计算和设定](#) 章节描述
- **位[17:16] SR:** **串行通讯码元采样率设定**
 00b: 标准采样, 采样频率 = $f_{PCLK}/16$
 01b: 8 倍速采样, 采样频率 = $f_{PCLK}/8$
 10b: 4 倍速采样, 采样频率 = $f_{PCLK}/4$
 11b: 4 倍速采样, 采样速率 = $f_{PCLK}/4$
- **位[31:18] 保留**

17.14.4 中断控制寄存器 UARTn_IE

(地址: 0x4000_2400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	TXEND _EN	TXF _EN	TXHE _EN	TXE _EN	OVER _EN	FERR _EN	PERR _EN	TOIDLE _EN	TONE _EN	RXF _EN	RXHF _EN	RXNE _EN
R/W	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] RXNE_EN:** 接收缓冲队列非空中断允许控制位
0b: 禁止中断
1b: 允许中断
- **位[1] RXHF_EN:** 接收缓冲队列半满中断允许控制位
0b: 禁止中断
1b: 允许中断
- **位[2] RXF_EN:** 接收缓冲队列全满中断允许控制位
0b: 禁止中断
1b: 允许中断
- **位[3] TONE_EN:** 接收缓冲队列清空超时中断允许控制位
0b: 禁止中断
1b: 允许中断
- **位[4] TOIDLE_EN:** 空闲超时中断允许控制位
0b: 禁止中断
1b: 允许中断
- **位[5] PERR_EN:** 奇偶校验错误中断允许控制位
0b: 禁止中断
1b: 允许中断
- **位[6] FERR_EN:** 帧错误中断允许控制位
0b: 禁止中断
1b: 允许中断
- **位[7] OVERR_EN:** 接收缓冲队列溢出中断允许控制位
0b: 禁止中断
1b: 允许中断
- **位[8] TXE_EN:** 发送缓冲队列全空中断允许控制位
0b: 禁止中断
1b: 允许中断

- **位[9] TXHE_EN:** 发送缓冲队列半空中断允许控制位
 0b: 禁止中断
 1b: 允许中断
- **位[10] TXF_EN:** 发送缓冲队列全满中断允许控制位
 0b: 禁止中断
 1b: 允许中断
- **位[11] TXEND_EN:** 发送全部完成中断允许控制位
 0b: 禁止中断
 1b: 允许中断
- **位[31:12] :** 保留

17.14.5 状态寄存器 UARTn_ST

(地址: 0x4000_2400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	TXEND	TXF	TXHE	TXE	OVER	FERR	PERR	TOIDLE	TONE	RXF	RXHF	RXNE
R/W	R	R	R	R	R	R	R	R	RW1c	RW1c	RW1c	R	R	R	R	R
复位	0	0	0	0	1	0	1	1	0	0	0	1	0	0	0	0

- **位[0] RXNE:** 接收缓冲队列非空中断标志位
 0b: 接收缓冲队列为空, 队列中没有任何数据
 1b: 接收缓冲队列中至少有一个数据
 该标志位由硬件设定, 软件无法改写
- **位[1] RXHF:** 接收缓冲队列半满中断标志位
 0b: 接收缓冲队列中的数据少于或等于队列长度的一半
 1b: 接收缓冲队列中的数据大于队列长度的一半
 该标志位由硬件设定, 软件无法改写; 队列长度为 8
- **位[2] RXF:** 接收缓冲队列全满中断标志位
 0b: 接收缓冲队列中数据尚未全满
 1b: 接收缓冲队列中数据已经全满
 该标志位由硬件设定, 软件无法改写
- **位[3] TONE:** 接收缓冲队列尚未清空超时中断标志位
 0b: 没有超时
 1b: 当 UARTn_TO 寄存器设定的超时时间已到但接收缓冲队列尚未被清空
 该标志位由硬件设定, 软件无法改写

- **位[4] TOIDLE:** **接收队列全空超时中断标志位**
 0b: 没有超时
 1b: 当 UARTn_TO 寄存器设定的超时时间已到但接收缓冲队列全空
 该标志位由硬件设定，软件无法改写
- **位[5] PERR:** **奇偶校验错误中断标志位**
 0b: 最近一次接收无奇偶校验错误
 1b: 最近一次接收检有奇偶校验错误
 该标志位必须由软件对 PERR 位写 1 才能将其清除
- **位[6] FERR:** **帧错误中断标志位**
 0b: 最近一次接收无帧错误
 1b: 最近一次接收有帧错误
 该标志位必须由软件对 FERR 位写 1 才能将其清除
- **位[7] OVERR:** **接收缓冲队列溢出中断标志位**
 0b: 接收缓冲队列无溢出
 1b: 接收缓冲队列发生溢出
 该标志位必须由软件对 OVERR 位写 1 才能将其清除
- **位[8] TXE:** **发送缓冲队列全空中断标志位**
 0b: 发送缓冲队列中至少有一个数据
 1b: 发送缓冲队列为空，队列中没有任何数据
 该标志位由硬件设定，软件无法改写
- **位[9] TXHE:** **发送缓冲队列半空中断标志位**
 0b: 发送缓冲队列中的数据超过队列长度的一半
 1b: 发送缓冲队列中的数据少于或等于队列长度的一半
 该标志位由硬件设定，软件无法改写
- **位[10] TXF:** **发送缓冲队列全满中断标志位**
 0b: 发送缓冲队列中数据尚未全满
 1b: 发送缓冲队列中数据已经全满
 该标志位由硬件设定，软件无法改写
- **位[11] TXEND:** **发送全部完成中断标志位**
 0b: 数据发送正在进行
 1b: 数据发送全部结束
 该标志位由硬件设定，软件无法改写
- **位[31:12] :** **保留**

17.14.1 状态清除寄存器 UARTn_STCLR

(地址: UART0: 0x4000_4410; UART1: 0x4001_3810)

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

位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	OVER	FERR	PERR	-	-	-	-
R/W	-	-	-	-	-	-	-	-	-	W	W	W	-	-	-	-
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[3:0] :** 保留（不可读）
- **位[4] PERR:** 奇偶校验错误中断清除
 - 0b: 无效
 - 1b: 写 1 清除
 - 该位不可读
- **位[5] FERR:** 帧错误中断标志位
 - 0b: 无效
 - 1b: 写 1 清除
 - 该位不可读
- **位[6] OVERR:** 接收缓冲队列溢出中断标志位
 - 0b: 无效
 - 1b: 写 1 清除
 - 该位不可读
- **位[31:7] :** 保留（不可读）

17.14.2 帧间隔时间寄存器 UARTn_GT

(地址: 0x4000_2400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	GT							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[7:0] GT:** **帧间隔时间**
发送两个连续数据帧的间隔时间, LSB 为单个位元的时间宽度
- **位[31:8] :** **保留**

17.14.3 超时控制寄存器 UARTn_TO

(地址: 0x4000_2400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	TO							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[7:0] TO:** **超时时间**
用于数据接收过程中的超时判断, LSB 为单个位元的时间宽度
当设定的超时时间已到但接收缓冲器仍然为非空时, UARTn_ST 寄存器的 TONE 位被置 1; 当设定的超时时间已到但接收缓冲器仍然为全空时, UARTn_ST 寄存器的 TOIDLE 位被置 1。
如果 UARTn 开始接收数据, UARTn_ST 寄存器的 TONE 位和 UARTn_ST 寄存器的 TOIDLE 位的值将被复位成 0, 直到设定的超时时间到了才会被更新。
超时时间的计时在 UARTn 被使能且接收为空闲的时候进行计时, 每次读取 UARTn_DAT 寄存器或重新写 UARTn_TO 寄存器会重新计时。
- **位[31:8] :** **保留**

17.14.4 发送队列复位寄存器 UARTn_TXFR

(地址: 0x4000_2400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[31:0] TXFR: 发送队列复位**
对 UARTn_TXFR 寄存器进行一次写入操作后, 不管写入什么数值, 发送缓冲队列即被复位清空。
该寄存器只写, 读恒为零。

17.14.5 接收队列复位寄存器 UARTn_RXFR

(地址: 0x4000_2400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[31:0] RXFR: 接收队列复位**
对 UARTn_RXFR 寄存器进行一次写入操作后, 不管写入什么数值, 接收缓冲队列即被复位清空。
该寄存器只写, 读恒为零。

17.14.6 ISO7816 收发控制寄存器 UARTn_ISO7816

(地址: 0x4000_2400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	CLKDIV			-	-	DIR	EN
R/W	R	R	R	R	R	R	R	R	R	RW	RW	RW	R	R	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] EN: **ISO7816 使能控制**
 0b: ISO7816 功能禁止
 1b: ISO7816 功能使能
- 位[1] DIR: **ISO7816 数据传输方向控制**
 0b: ISO7816 数据接收状态
 1b: ISO7816 数据发送状态
- 位[3:2]: **保留**
- 位[6:4] CLKDIV: **ISO7816 时钟频率选择**
 ISO7816 时钟频率为 PCLK 频率的分频, 分频系数选择如下:

CLKDIV	分频系数
000	1
001	2
010	4
011	8
100	16
101	32
其它	4

- 位[31:7]: **保留**

17.14.7 红外发送解调控制寄存器 UARTn_IR_CTL

(地址: 0x4000_2400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	SSB_WIDTH			
R/W	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	IR_RST	-	-	-	-	-	IR_SPD		-	-	TXPOL	EN
R/W	R	R	R	R	RW	R	R	R	R	RW	RW	RW	R	R	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] EN:** 红外功能使能控制
 - 0b: 红外功能禁止
 - 1b: 红外功能使能
- **位[1] TXPOL:** 红外发送极性控制
 - 0b: 正常极性
 - 1b: 反极性
- **位[3:2]:** 保留
- **位[5:4] IR_SPD:** 红外速率选择控制
 - 00b: 600bps
 - 01b: 1200bps
 - 10b: 2400bps
 - 11b: 保留
- **位[10:6]:** 保留
- **位[11] IR_RST:** 红外调制软复位
 - 当该位为 1 时, 不调制
 - 1b: 复位调制电路
 - 0b: 正常解调红外
- **位[15:12]:** 保留
- **位[19:16] SSB_WIDTH:** 红外解调起始位宽度缩减
 - 红外接收信号解调时, 根据波形调整解调电路对于起始位的宽度解调
 - 例: SSB_WIDTH = 1, 意味着起始位相较于标准的波特率少一个 38kHz 时钟周期
- **位[31:20]:** 保留

17.14.8 红外发送占空比控制寄存器 UARTn_IR_PWMC

(地址: 0x4000_2400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	PWMC											
R/W	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[11:0] PWMC: 红外调制占空比控制**
 PWM 调制基于系统时钟周期和 38KHz 载波周期的分频比, 假设一个 38KHz 载波周期包含 100 个 PCLK 时钟周期, 则配置 PWMC=1 时, 高电平为一个 PCLK 时钟周期, 低电平为 99 个 PCLK 时钟周期, 以此类推。默认 PWMC=0, 表示 38KHz 载波为 1:1 的占空比。
 注意: 若配置的 PWMC 所表示的脉宽宽度超出了 38KHz 载波周期, 则可能会出现无法预计的异常调制波形。
- **位[31:12]: 保留**

18 SPI

18.1 概述

SPI 是一种同步串行外设接口，对数据进行串并转换，通过主从方式的方式实现 MCU 和外围器件的数据通讯。

SPI 通讯是全双工的，发送或接收数据都是高位在先，在数据发送的同时进行数据接收。按通讯时钟的不同提供方式，SPI 通讯分主机和从机两种。SPI 主机的时钟由本地产生，通讯的发起和结束完全由自己主动控制；SPI 从机的时钟为外部输入（来自 SPI 主机），被动响应主机的通讯。

PT32L031 的 SPI 模块支持主/从模式工作。基本特性如下：

- 一帧数据传输 4~16 位可编程
- 主机最高传输速率为系统时钟（PCLK）二分频
- 最大 8 级接收 FIFO 缓冲队列
- 最大 8 级发送 FIFO 缓冲队列
- 主机模式下可编程传输速率

18.2 SPI 通讯信号

SPI 通讯所用引脚描述如下：

- CS：SPI 串行通讯从机片选控制。每一个从机都有一个独立的片选信号控制。
- SCK：SPI 串行通讯时钟，时钟信号只有 SPI 主机才能发出，所有从机的时钟只能输入来自主机的时钟信号。
- MISO：SPI 串行通讯数据输入。该数据为从机发出，在主机侧为输入。
- MOSI：SPI 串行通讯数据输出。该数据为主机发出，在从机侧为输入。

18.3 SPI 工作模式

SPI 模块支持三种模式，Motorola SPI 格式，TI 同步串行格式以及 National Microwire 模式。

18.3.1 TI 同步串行模式

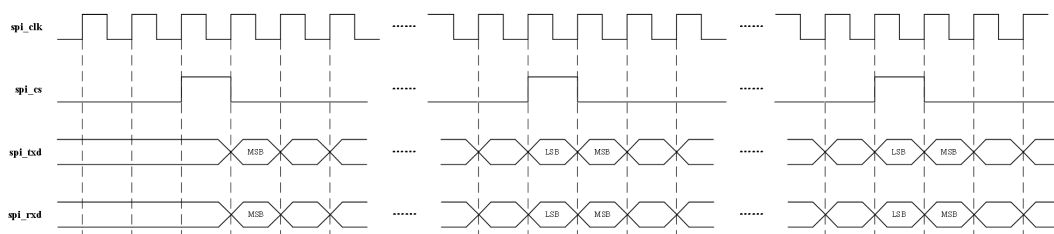


图 18-1 TI 同步串行模式数据传输时序图

当 SPI 处于 TI 同步串行模式时，SPI 处于全双工的工作状态，在空闲状态下 SPI 时钟和片选信号 CS 处于低电平，一旦发送 FIFO 内被写入了数据，则 CS 立即拉高一个 SPI 时钟周期，之后数据开始串行传输，数据帧长短可通过 SPIn_CR0 寄存器的 DSS 位配置。

数据传输时，高位先发，数据在 SPI 时钟上升沿打出，在下降沿保持稳定。

18.3.2 National Microwire 模式

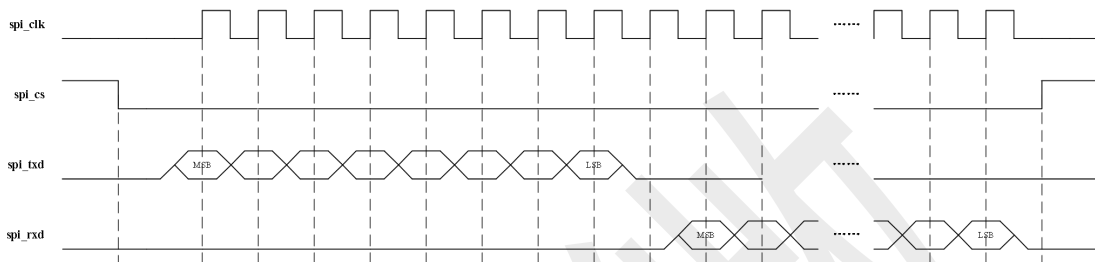


图 18-2 National Microwire 模式单帧数据传输时序图

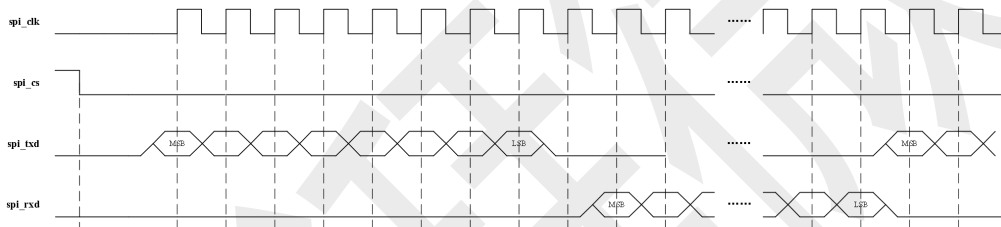


图 18-3 National Microwire 模式连续传输时序图

当 SPI 处于 National Microwire 模式时，SPI 处于半双工的工作状态，在空闲状态下 SPI 时钟处于低电平，片选信号 CS 处于高电平，一旦发送 FIFO 内被写入了数据，则主机拉低 CS 信号，之后数据开始串行传输，先固定发送 8bit 数据，之后隔一个 SPI 时钟周期后开始接收从机发送的数据，接收的数据帧长短可通过 SPIn_CR0 寄存器的 DSS 位配置。

数据传输时，高位先发，数据在 SPI 时钟下降沿打出，在上升沿保持稳定。

18.3.3 Motorola 模式

当 SPI 处于 Motorola 模式时，SPI 处于全双工的工作状态，并且根据 SPIn_CR0 寄存器的 SCLKP 位的设定，分为 4 种不同的工作模式，分别为模式 00、模式 01、模式 10 和模式 11。

18.3.3.1 模式 00

当设为模式 00 时，在空闲状态下 SPI 时钟保持低电平。主机把从机的片选信号 CS 拉低后，从机发往主机的第一位数据即出现在主机的 MISO 引脚上就绪；主机往 SPI 数据发送寄存器内写入数据后，发往从机的第一位数据也即刻出现在 MOSI 引脚上，之后在每一个时钟上升沿主机和从机各自同步采样接收对方发送过来的数据位并送入内部移位寄存器，在其后的时钟下降沿各自打出下一位数据。当一个 SPI 数据帧（4~16 位）传输结束后，接收到的数据从内部移位寄存器转入 SPI 接收 FIFO，SPI 时钟恢复到低电平状态。主机可继续往发送寄存器内写入数据进行连续发送，或者通过软件把从机片选信号拉高，结束 SPI 通讯。

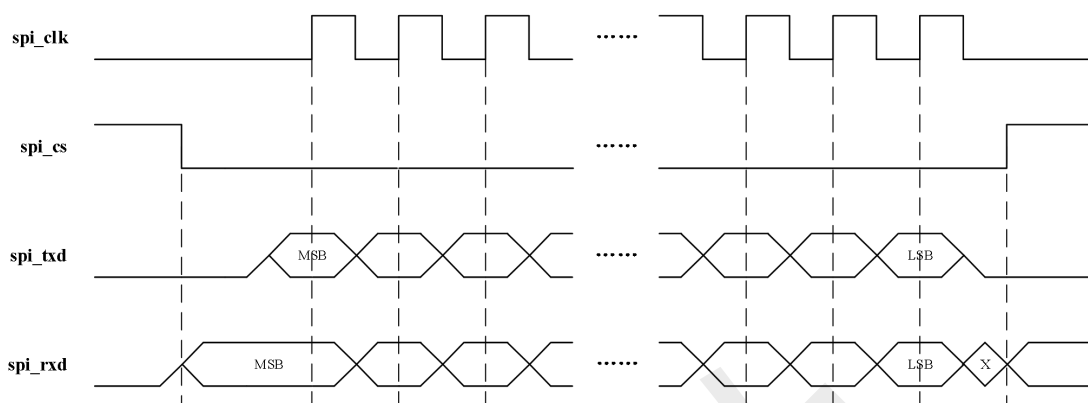


图 18-4 模式 00 时序图 (SCLKP=00)

18.3.3.2 模式 01

当设为模式 01 时，在空闲状态下 SPI 时钟保持高电平。主机把从机的片选信号 CS 拉低后，从机发往主机的第一位数据即出现在主机的 MISO 引脚上就绪；主机往 SPI 数据发送寄存器内写入数据后，发往从机的第一位数据也即刻出现在 MOSI 引脚上，之后在每一个时钟下降沿主机和从机各自同步采样接收对方发送过来的数据位并送入内部移位寄存器，在其后的时钟上升沿各自打出下一位数据。当一个 SPI 数据帧（4~16 位）传输结束后，接收到的数据从内部移位寄存器转存入 SPI 接收 FIFO，SPI 时钟恢复到低电平状态。主机可继续往发送寄存器内写入数据进行连续发送，或者通过软件把从机片选信号拉高，结束 SPI 通讯。

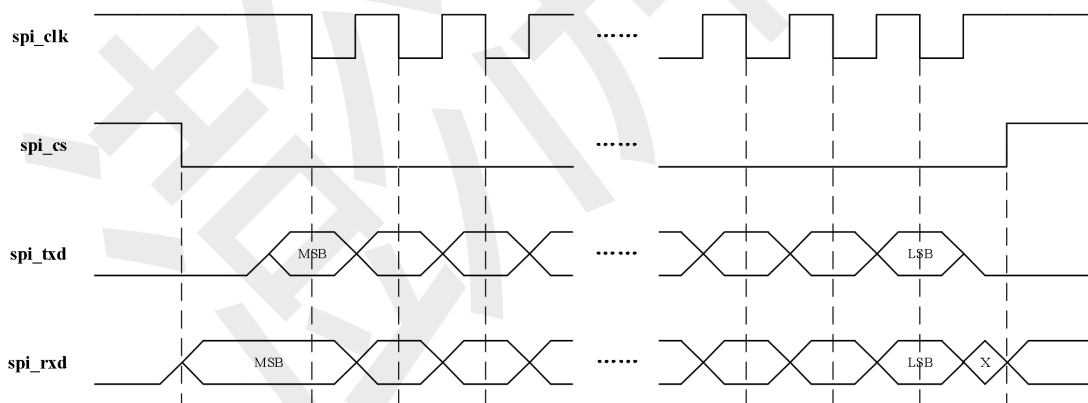


图 18-5 模式 01 时序图 (SCLKP=01)

18.3.3.3 模式 10

当设为模式 10 时，在空闲状态下 SPI 时钟保持低电平。主机把从机的片选信号 CS 拉低后，从机发往主机的第一位数据即出现在主机的 MISO 引脚上就绪；主机往 SPI 数据发送寄存器内写入数据后，发往从机的第一位数据也即刻出现在 MOSI 引脚上，之后在每一个时钟下降沿主机和从机各自同步采样接收对方发送过来的数据位并送入内部移位寄存器，在其后的时钟上升沿各自打出下一位数据。当一个 SPI 数据帧（4~16 位）传输结束后，接收到的数据从内部移位寄存器转存入 SPI 接收 FIFO，SPI 时钟恢复到低电平状态。主机可继续往发

送寄存器内写入数据进行连续发送，或者通过软件把从机片选信号拉高，结束 SPI 通讯。

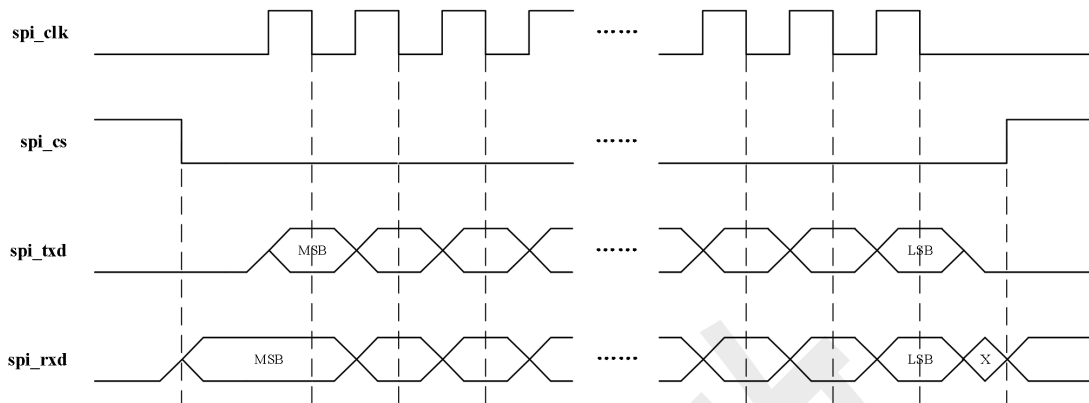


图 18-6: 模式 10 时序图 (SCLKP=10)

18.3.3.4 模式 11

当设为模式 10 时，在空闲状态下 SPI 时钟保持高电平。主机把从机的片选信号 CS 拉低后，从机发往主机的第一位数据即出现在主机的 MISO 引脚上就绪；主机往 SPI 数据发送寄存器内写入数据后，发往从机的第一位数据也即刻出现在 MOSI 引脚上，之后在每一个时钟上升沿主机和从机各自同步采样接收对方发送过来的数据位并送入内部移位寄存器，在其后的时钟下降沿各自打出下一位数据。当一个 SPI 数据帧（4~16 位）传输结束后，接收到的数据从内部移位寄存器转存入 SPI 接收 FIFO，SPI 时钟恢复到低电平状态。主机可继续往发送寄存器内写入数据进行连续发送，或者通过软件把从机片选信号拉高，结束 SPI 通讯。

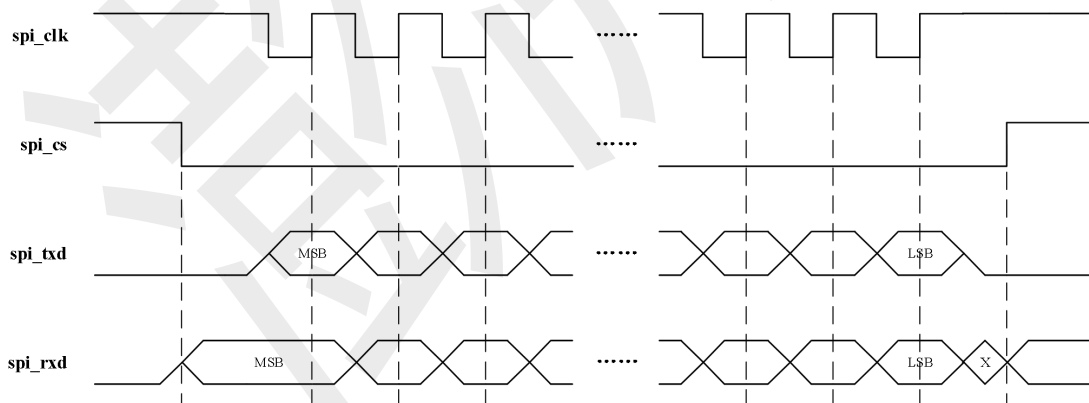


图 18-7: 模式 11 时序图 (SCLKP=11)

18.4 SPI 通讯波特率计算和设定

当 SPI 被设置为主机模式 (SPIn_CR1 寄存器的 MS 位为 0) 时，SPI 通讯时钟由硬件主动产生并发送至从机，其通讯速率计算公式如下：

$$f_{SPI} = f_{SYS} / (CPSDVSr * (1 + SCR))$$

其中：

- f_{SPI} 为期望的 SPI 通讯速率，同时用于接收和发送
- f_{SYS} 为 PCLK 时钟频率
- CPSDVSR 为 PCLK 预分频寄存器的设定值，取值范围为 2~254 间的偶数（其第 0 位必须为 0），软件可通过 SPIIn 时钟预分频寄存器 SPIIn_CPSR 的 CPSDVSR 位配置所需预分频系数
- SCR 为经过预分频后的时钟的分频系数，取值范围为 0~255，分别对应于 1~256 分频，软件可通过 SPIIn_CR0 控制寄存器 1 的 SCR 位配置所需分频系数，
当芯片被设置为 SPI 从机（SPIIn_CR1 寄存器的 MS 位为 1）时，SPI 通讯时钟为外部输入。为能保证从机模式下的数据接收，必须满足芯片系统时钟（PCLK）频率大于等于 SPI 数据时钟频率的 12 倍。

18.5 SPI 中断

SPI 模块的中断源一共有四个，分别对应数据发送 FIFO 和数据接收 FIFO 的不同状态：

接收数据 FIFO 溢出

当软件没有及时读取接收到的数据，导致接收 FIFO 中已经接收到 8 个数据处于 FIFO 为满的状态时，这时又收到第 9 个数据时，接收 FIFO 就会发生溢出，最早收到的数据被顶出队列而丢弃，接收 FIFO 溢出标志位（OV）被置 1

接收 FIFO 非空并且软件读取数据超时

当接收 FIFO 非空（FIFO 中有一个或多个数据）但软件没有在一定时间内（32 个 SPI 通讯时钟）进行任何的读取数据操作，接收 FIFO 非空超时标志位（TO）被置 1

接收 FIFO 数据过半

当接收 FIFO 中的数据过半（数据个数 ≥ 4 ）时，接收 FIFO 数据半满或过半标志位（RX）被置 1

发送 FIFO 数据过半

当发送 FIFO 中的数据过半（数据个数 ≥ 4 ）时，发送 FIFO 数据半满或过半标志位（TX）被置 1

上述 4 个状态标志位软件可通过原始中断标志寄存器 SPIIn_RIS 查看。软件可通过配置中断使能控制寄存器 SPIIn_IE 决定上述 4 个状态标志位中哪些中断源被使能从而发出中断请求，经 SPIIn_IE 使能控制后的中断标志反映在使能中断标志寄存器 SPIIn_MIS 中。

被使能后的中断标志通过或的逻辑，最终向 CPU 发出同一个中断请求，软件可通过查询使能中断标志寄存器 SPIIn_MIS 寄存器查询具体的中断源并做出相应的处理。

18.6 寄存器列表

地址	寄存器	描述	备注
0x4001_3000	SPIO_CR0	SPIO 控制寄存器 0	SPIO_CR0 说明
0x4001_3004	SPIO_CR1	SPIO 控制寄存器 1	SPIO_CR1 说明
0x4001_3008	SPIO_DR	SPIO 数据寄存器	SPIO_DR 说明
0x4001_300C	SPIO_SR	SPIO 状态寄存器	SPIO_SR 说明
0x4001_3010	SPIO_CPSR	SPIO 时钟预分频寄存器	SPIO_CPSR 说明
0x4001_3014	SPIO_IE	SPIO 中断使能寄存器	SPIO_IE 说明

0x4001_3018	SPI0_RIS	SPI0 原始中断标志寄存器	SPI0_RIS 说明
0x4001_301C	SPI0_MIS	SPI0 使能中断标志寄存器	SPI0_MIS 说明
0x4001_3020	SPI0_ICR	SPI0 中断标志清除寄存器	SPI0_ICR 说明
0x4001_3028	SPI0_CSCCR	SPI0 片选信号控制寄存器	SPI0_CSCCR 说明
0x4000_3800	SPI1_CR0	SPI1 控制寄存器 0	SPI1_CR0 说明
0x4000_3804	SPI1_CR1	SPI1 控制寄存器 1	SPI1_CR1 说明
0x4000_3808	SPI1_DR	SPI1 数据寄存器	SPI1_DR 说明
0x4000_380C	SPI1_SR	SPI1 状态寄存器	SPI1_SR 说明
0x4000_3810	SPI1_CPSR	SPI1 时钟预分频寄存器	SPI1_CPSR 说明
0x4000_3814	SPI1_IE	SPI1 中断使能寄存器	SPI1_IE 说明
0x4000_3818	SPI1_RIS	SPI1 原始中断标志寄存器	SPI1_RIS 说明
0x4000_381C	SPI1_MIS	SPI1 使能中断标志寄存器	SPI1_MIS 说明
0x4000_3820	SPI1_ICR	SPI1 中断标志清除寄存器	SPI1_ICR 说明
0x4000_3828	SPI1_CSCCR	SPI1 片选信号控制寄存器	SPI1_CSCCR 说明

18.7 寄存器描述

18.7.1 SPIn 控制寄存器 0 SPI_n_CR0

(地址: SPI0: 0x4001_3000; SPI1: 0x4000_3800)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	SCR						SCLKP		FRF		DSS					
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[3:0] DSS: 帧数据长度选择

- 0000b: 保留
- 0001b: 保留
- 0010b: 保留
- 0011b: 数据帧长度为 4 位
- 0100b: 数据帧长度为 5 位
- 0101b: 数据帧长度为 6 位
- 0110b: 数据帧长度为 7 位
- 0111b: 数据帧长度为 8 位
- 1000b: 数据帧长度为 9 位

- 1001b: 数据帧长度为 10 位
- 1010b: 数据帧长度为 11 位
- 1011b: 数据帧长度为 12 位
- 1100b: 数据帧长度为 13 位
- 1101b: 数据帧长度为 14 位
- 1110b: 数据帧长度为 15 位
- 1111b: 数据帧长度为 16 位
- **位[5:4] FRF: 帧格式选择**
 - 00b: Motorola SPI 格式
 - 01b: TI 同步串行格式
 - 10b: Microwire 格式
 - 11b: 保留
- **位[7:6] SCLKP: SPI 时钟极性相位控制**
 - 00b: 空闲状态下时钟保持低电平, 时钟上升沿采样数据, 时钟下降沿出数据
 - 01b: 空闲状态下时钟保持高电平, 时钟下降沿采样数据, 时钟上升沿出数据
 - 10b: 空闲状态下时钟保持低电平, 时钟下降沿采样数据, 时钟上升沿出数据
 - 11b: 空闲状态下时钟保持高电平, 时钟上升沿采样数据, 时钟下降沿出数据
- **位[15:8] SCR: SPI 时钟分频系数**
详见 18.3 SPI 通讯波特率计算和设定章节介绍
- **位[31:16] :** 保留

18.7.2 SPIIn 控制寄存器 1 SPIIn_CR1

(地址: SPI0: 0x4001_3004; SPI1: 0x4000_3804)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	MS	EN	LBM
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] LBM: 回绕模式控制**
 - 0b: 正常工作模式
 - 1b: 回绕传输模式 (自发自收)
- **位[1] EN: SPI 使能控制**
 - 00b: SPI 使能
 - 01b: SPI 禁止
- **位[2] MS: 主从模式选择**
 - 0b: 主机模式
 - 1b: 从机模式

- 位[31:3] : 保留

18.7.3 SPIIn 数据寄存器 SPIIn_DR

(地址: SPI0: 0x4001_3008; SPI1: 0x4000_3808)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	DATA															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[15:0] DATA: 传输数据
数据的格式均为右对齐, 若数据长度小于 16 位, 向发送 FIFO 写入数据时则需保持右对齐格式, 否则无效的高位数据将被忽略, 不被发送; 接收 FIFO 的数据格式自动为右对齐格式

写: 将数据写入发送 FIFO

读: 从接收 FIFO 读出数据

- 位[31:16] : 保留

18.7.4 SPIIn 状态寄存器 SPIIn_SR

(地址: SPI0: 0x4001_300C; SPI1: 0x4000_380C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	BSY	REF	RNE	TNF	TFE
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] TFE: 发送 FIFO 全空标志位

0b: 发送 FIFO 非空

1b: 发送 FIFO 为空

- 位[1] TNF: 发送 FIFO 未满载标志位

- 0b: 发送 FIFO 全满
- 1b: 发送 FIFO 未滿

- **位[2] RNE:** 接收 FIFO 非空标志位
 - 0b: 接收 FIFO 为空
 - 1b: 接收 FIFO 非空
- **位[3] RFF:** 接收 FIFO 全滿标志位
 - 0b: 接收 FIFO 未滿
 - 1b: 接收 FIFO 全滿
- **位[4] BSY:** 数据传输状态标志位
 - 0b: SPI 处于空闲状态
 - 1b: SPI 处于传输数据状态或者发送 FIFO 非空
- **位[31:5] :** 保留

18.7.5 SPI_{In} 时钟预分频寄存器 SPI_{In}_CPSR

(地址: SPI0: 0x4001_3010; SPI1: 0x4000_3810)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	CPSDVSR							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[7:0] CPSDVSR:** SPI 时钟预分频系数
CPSDVSR 必须是 2~254 之间的偶数 (bit0 只读, 恒为 0)
具体使用详见 18.3 章节描述
- **位[31:8] :** 保留

18.7.8 SPIn 使能中断标志寄存器 SPIn_MIS

(地址: SPI0: 0x4001_301C; SPI1: 0x4000_381C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	TX	RX	TO	OV
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] OV:** **接收 FIFO 溢出中断标志位**
 0b: 接收 FIFO 未发生溢出或中断未使能
 1b: 接收 FIFO 溢出并产生中断
- **位[1] TO:** **接收 FIFO 非空并超时中断标志位**
 0b: 接收 FIFO 为空或中断未使能
 1b: 接收 FIFO 非空并超时并产生中断
- **位[2] RX:** **接收 FIFO 数据半满或过半中断标志位**
 0b: 接收 FIFO 数据未过半或中断未使能
 1b: 接收 FIFO 数据半满或已过半并产生中断
- **位[3] TX:** **发送 FIFO 数据半满或未半满中断标志位**
 0b: 发送 FIFO 数据已过半或中断未使能
 1b: 发送 FIFO 数据半满或未半满并产生中断
- **位[31:4] :** **保留**

18.7.9 SPIn 中断标志清除寄存器 SPIn_ICR

(地址: SPI0: 0x4001_3020; SPI1: 0x4000_3820)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	TOCLR	OVCLR
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] OVCLR:** **接收 FIFO 溢出中断清除**
 0b: 无作用
 1b: 清除接收 FIFO 溢出中断
- **位[1] TOCLR:** **接收 FIFO 非空并超时中断标志位**
 0b: 无作用
 1b: 清除接收 FIFO 非空并超时中断
- **位[31:2] :** **保留**

18.7.10 SPIn 片选信号控制寄存器 SPIn_CSCR

(地址: SPI0: 0x4001_3028; SPI1: 0x4000_3828)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	SWCS	SWSEL	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[1:0] :** **保留**
- **位[2] SWSEL:** **片选信号控制方式选择**
 0b: SPI 模块硬件自动控制
 1b: 软件通过配置 SWCS 位控制
- **位[3] SWCS:** **软件片选信号控制**
 0b: 片选信号为 0
 1b: 片选信号为 1

19 I2C

19.1 I2C 模块综述

19.2 I2C 模块综述

I2C是嵌入式系统设计中经常被用到的一种串行通讯总线。它基于SCL（串行时钟）和SDA（串行数据）双线联机，以主从方式实现多个互联设备之间的双向数据通讯。

PT32L031片内I2C模块支持主模式和从模式通讯方式，其基本特性如下：

- 内含并行数据/串行I2C协议转换器
- 支持多个主机并存，提供总线仲裁机制，使总线串行数据不损坏
- 支持7位从机寻址模式
- 支持广播呼叫
- 提供数据发送和接收状态标识
- 提供字节传输结束标识
- 通讯错误检测
- 支持标准（100Kbps）I2C通讯速率模式

作为I2C主机时，其特性包括：

- 产生总线通讯时钟
- 实现总线仲裁
- 7位地址寻址从机，并控制数据读写方向
- 产生总线通讯起始位、重复起始位和停止位
- 检测通讯错误

作为I2C从机时，其特性包括：

- 检测起始位、重复起始位和停止位
- 可编程I2C7位地址匹配寻址
- 传输过程中检测起始和停止条件
- 检测通讯错误

启用 I2C 模块前，需要将 SDA、SCL 的 IO 复用功能使能，使 SDA、SCL 端口的 GPIO 功能禁止，IO 口作为 SDA、SCL 使用。具体参考 GPIO 模块端口复用的说明。启用 I2C 模块后，SDA 和 SCL 线被配置成高阻开漏状态，使能弱上拉，上拉电阻为 50K 左右。外部硬件也可以用上拉电阻将其拉至高电平，上拉电阻阻值视工作电压、通讯速度和总线负载而定，一般可选 5~10K。SCL 和 SDA 线是双向的，当 I²C 总线处于自由或空闲状态时，两个引脚都处于高电平状态，适用于多个互连设备所要求的线与功能。

I2C模块在开始工作前，首先必须由软件按芯片的系统时钟频率设定I2C_CTLSET中CR2/CR1/CR0分频系数，选择合适的SCL通讯时钟频率，详见I2C时钟速度计算和设定；然后配置I2C_CTLSET寄存器EN位使能I2C模块，通讯过程的状态信息则在I2C_STAT状态寄存器中反映。如果作为I2C从机使用，则还需在I2C_ADDR地址寄存器中设定本机的地址码。

在发送数据前，SCL 线先被发送方拉低，软件写入一个字节到 I2C_DATA 数据寄存器

后，SDA 和 SCL 开始逐位输出数据和时钟信号进行数据发送；在接收到数据后，SCL 线被接收方持续拉低，直到接收方从 I2C_DATA 数据寄存器中读出数据。

19.3 I2C 协议简述

I2C总线协议要求所有挂在总线上的设备，其SDA和SCL线必须是开漏输出状态，换句话说，任何设备可以输出低电平将SDA或SCL线拉低，但不能输出高电平驱动。总线上的高电平只能依靠硬件上拉电阻获得。

一般的I2C总线通讯包含四部分：

- 起始位发送，表示一次通讯过程开始
- 从机地址发送
- 数据传输
- 停止位发送，表示一次通讯过程结束

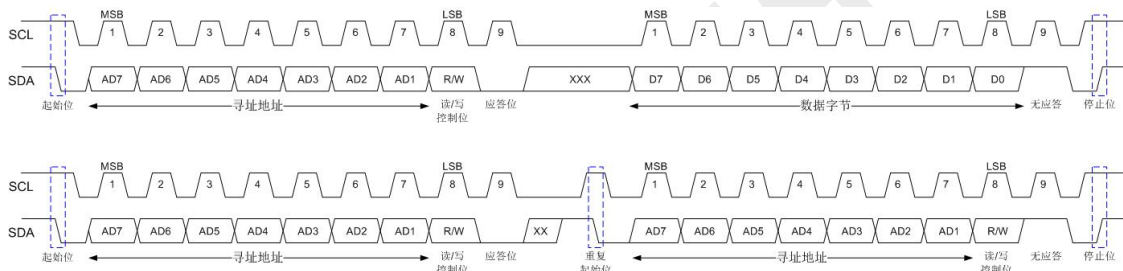


图19-1 I²C总线通讯波形示意图

图19-1显示了一个的I2C总线通讯过程示例。

19.3.1 起始位

总线空闲时，没有任何设备占用总线（SCL和SDA线处于逻辑高电平），主机可以通过发送起始位发起通信。如图19-1所示，起始位定义为，当SCL在高电平时，SDA从高到低的电平跳变。该信号表示开始新的数据传输（每次数据传输可能包含几个字节的数据），并使所有从机退出空闲状态。

19.3.2 从机寻址

起始位发出后传输的第一个字节数据是主机发送的从机寻址地址。其中的高7位是从机的地址码，最低位R/W决定总线上数据传输的方向：

R/W = 1 : 读取传输，从机向主机传输数据

R/W = 0 : 写入传输，主机向从机传输数据

从机地址可以配置I2C_ADDR的ADDR栏位，当主机发送的地址与总线上从机地址匹配时，该从机在第9个时钟周期时将SDA线拉低（见图19-1），回送应答位（ACK）进行响应。

I2C 总线上不能有两个地址相同的从机。如果 I2C 模块是主机，它就不应该发送与其自身从机地址相同的地址。I2C 设备不能同时既是主机又是从机，但是如果在寻址过程中出现仲裁丢失，设备就重新返回到从机模式并正确运行，并可被另一个主机寻址。

19.3.3 数据传输

成功实现从机寻址后，就可以按照主机发送的R/W位指定的方向逐字节地进行数据传输。主机发送地址后的所有传输都被称为数据传输，即使它们包含从机的子地址报文。

每个数据字节的长度均为8位。在传输数据时，在SCL时钟的高电平期间，SDA的数据必须保持稳定。只有当SCL为低时，SDA的数据才可以更改，SCL的一个时钟脉冲传输一个数据位，最高位被首先传输，如图19-1所示。每个数据字节后面都有一个应答位（第9位），该位由从机（发送时）或主机（读取时）回送信号，通过在第9个时钟周期时把SDA线拉低来实现。这样，一个完整数据字节的传输需要9个时钟脉冲。如果对方在第9个时钟周期时无应答（NACK），则其必须保留SDA线在高电平。主机发送数据，从机返回无应答（NACK），主机将接收到的无应答信号解释为不成功的数据传输；如果主机在接收一个数据字节传输后未应答，返回一个无应答（NACK）给从机，从机将其理解为数据传输结束，并释放总线给主机，因此，为了在收到最后一个字节后产生一个NACK脉冲，在读倒数第二个数据字节之后必须清除ACK位。对于以上两种情况，数据传输都被中止，主机会进行以下两种操作之一：

- 发送停止位，终止数据传输，并释放总线
- 发送重复起始位，开始新呼叫

19.3.4 停止位

主机可以通过发送停止位终止通信，以释放总线。然而，主机可以直接发送起始位和呼叫命令，而无需首先发送停止位，这被称为重复启动。停止位的定义是SCL在逻辑1位置时的从低到高的SDA电平跳变（见图19-1）。即便从机发出一个应答，主机也可以发送停止位，此时从机必须释放总线。

19.3.5 重复起始位

如图19-1所示，重复起始位是在无需首先生成停止位以终止通信的情况下，再发送一个开始（START）的信号。该信号由主机用来与另外一个从机进行通信，或者在不同模式（发送/接收模式）中与同一从机进行通信，而不需要释放总线。

19.3.6 总线仲裁

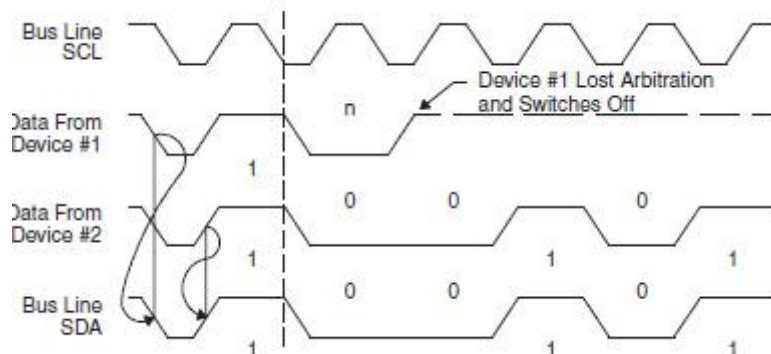


图 19-2 I²C 两个主机仲裁过程示意图

只有在I2C总线处于空闲或者自由的状态时，主机才能开始传输数据。I2C总线是真正的多主控总线，允许一个以上的主机连接到I2C总线上。如果两个甚至多个主机试图同时控制总线，仲裁机制赋予发送二进制低位(逻辑低)串行数据的设备更高的优先级。竞争主机的相对优先级由数据仲裁过程确定，如果一个总线主机发出逻辑1，而另一个发出逻辑0，那么前者就丢失仲裁（见图19-2）。失败的一方立即切换到从机接收模式，停止驱动SDA输出。在这种情况下，从主模式到从模式的转换不会生成停止条件。与此同时，丢失仲裁的设备状态位I2C_STAT=0x38，表示仲裁丢失。

19.3.7 时钟同步

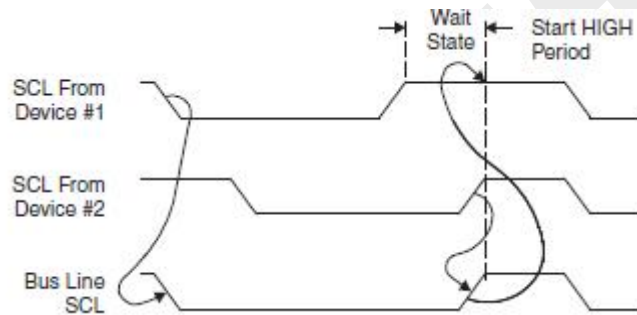


图 19-3 I²C 总线仲裁过程中 SCL 波形示意图

在仲裁过程中，不同主机发送的时钟需要被同步。由于SCL线上的线与逻辑，所以SCL上的电平跳变受连接到总线上的所有设备影响。图19-3显示了时钟同步的仲裁过程中，SCL线上的电平变化。当主机的时钟进入低电平周期后，它将一直保持SCL线的低电平，直到时钟到达高电平。然而，如果另一个主机时钟依旧是低电平，这时主机时钟从低到高的变化就不会改变SCL线的状态。因此，经过同步的时钟SCL拉低的时间由具有最长的低电平周期的主机决定。低电平周期较短的主机在该时段进入高电平等待状态（wait-state）（见图19-3）。当所有主机均结束低电平周期时，同步时钟SCL线才被释放和拉高。这时主机时钟和SCL线的状态一致，所有主机开始计数它们的高电平周期。第一个结束高电平周期的主机再次拉低SCL线。所以，经过同步的时钟SCL的低电平周期由所有主机中具有最长的低电平周期的主机决定，高电平周期由所有主机中具有最短高电平周期的主机决定。

19.3.8 通讯握手

时钟同步机制可以用作数据传输中的通讯握手。在完成一个字节的传输（9个位）后，从机可以保持SCL低位。在这种情况下，从机会暂停总线时钟，强迫主机时钟进入等待状态，直到从机释放SCL线。

19.3.9 时钟延展

时钟同步机制可以被从机用于减缓传输的比特速率。在主机已经拉低SCL后，从机可以继续拉低SCL一定时间，然后再释放它。如果从机SCL低电平周期长于主机SCL低电平周期，

那么就会将SCL总线信号低电平周期延展。

19.3.10 广播呼叫寻址

广播呼叫寻址功能可以用来寻址所有连接到 I²C 总线的设备。在寻址上，主机可通过向地址寄存器 I2C_ADDR 中的 ADDR 写入 0x00 启动广播呼叫寻址功能。

从机设备可通过设置相应的使能 I2C_ADDR 中的从机广播寻址控制位 GC 为 1，以使其支持广播呼叫寻址功能。如果 GC 位已经置 1 以支持广播呼叫寻址功能，I2C_CTLSET 寄存器中的 AA 位也应置 1，当设备接收到值为 00H 的地址时将返回一个确认信号(ACK)。当此条件发生时，从机状态寄存器更新，I2C_STAT=0x0E。

19.4 I2C 主模式（主机）

I2C 模块在初始化后默认处于从模式状态。在总线处于空闲状态下 (I2C_STAT=0xF8)，软件设定 I2C_CTLCLR_STA=1，模块即进入主模式并在总线上产生一个起始位。当 I2C_STAT 更新状态为 0x08，表示产生一个起始位完毕，并在中断允许时产生中断请求。此时主机将 SCL 线持续拉低，直到软件往 I2C_DATA 数据寄存器内写入首个字节（从机地址和传输方向控制位）进行发送。

19.4.1 I2C 主模式寻址方式

I2C 在主模式下在起始位后的首个发送字节一定是从机的地址码，PT32L031 支持 7 位寻址模式。

7 位寻址：

- 主机发送地址码字节，其中 7 位地址加最低位读写控制 R/W，决定主设备是发送器模式或接收器模式
- 查询等待或中断响应字节发送完毕
- 若从机地址匹配，并应答，则状态位 I2C_STAT=0x18；若无应答，则 I2C_STAT=0x20

地址发送完毕且从机正确应答后，主机将 SCL 线持续拉低，下一步主机将进入数据发送或数据接收过程。

19.4.2 I2C 主模式数据发送

主机发送完从机的寻址地址（其中的读写控制位为 0）后，主机为发送器模式，可以继续往 I2C_DATA 寄存器内写入数据进行发送，在数据被写入 I2C_DATA 寄存器前主机将 SCL 线持续拉低。主机每次发送完一个字节且收到从机的应答，则状态位寄存器 I2C_STAT=0x28；若从机无应答，则状态位寄存器 I2C_STAT=0x30，模块可以发送停止位终止 I2C 通讯。当最后一个字节发送完毕后，主机发送一个停止位，模块随即转为从模式。

19.4.3 I2C 主模式数据接收

主机发送完从机的寻址地址（其中的读写控制位为 1）后，主机为接收器设备模式，即

开始从从机处读取数据。每读取一个字节后主机根据ACK位的设定自动产生应答（I2C_CTLSET_AA=1时）或非应答（I2C_CTLSET_AA=0时），主机软件可查询I2C_STAT状态或中断响应，此时主机持续将SCL线拉低，直到主机软件把接收的数据从I2C_DATA寄存器中读走。在读取最新收到的数据前，如果主机确定下一个字节将是最后一个需读取的字节时，需要将ACK位清0（I2C_CTLSET_AA=0），确保在读取下一个字节时主机会向从机发送非应答位，从机收到无应答（NACK）就会认为数据传输结束，终止传输；读取最后一个字节后，主机发送一个停止位，释放总线，模块随即转为从模式。

19.4.4 I2C 主模式错误信息

I2C模块在主模式下如果检测到总线通讯发生错误，I2C_STAT状态将被置为0x00，同时模块将释放对总线的控制。

在一个字节传输过程在，当I2C侦测到总线产生一个停止位或起始位时，认为通讯发生错误。当检测到停止位导致的总线发生错误时，本机会释放总线，状态位I2C_STAT=0x00，产生中断标志位（I2C_CTLSET_SI=1），清除此标志位之后总线恢复空闲状态，I2C_STAT=0xF8；当检测到起始位导致的总线发生错误时，本机状态位I2C_STAT=0x00，产生中断标志位（I2C_CTLSET_SI=1），清除此标志位之后，主机需要发送停止位，使总线恢复空闲状态，I2C_STAT=0xF8。

19.5 I2C 从模式（从机）

I2C模块在初始化后默认处于从模式状态，等待总线上产生起始位，随后接收主机发出的寻址地址码并和本机设定地址进行比对匹配。

19.5.1 I2C 从模式地址匹配

I2C从模式地址寻址为7位地址：

- 从机接收的地址码字节高7位和地址寄存器I2C_ADDR [7:1]设定的本机地址作匹配比对，接收的地址码字节最低位为读写控制位，不参与地址比对。
- 主机发送7位地址与本机地址比对匹配成功，从机回送应答位；否则回送非应答位，从机等待下一个起始位。
- 地址匹配后从机将SCL线持续拉低，直到从机软件清除中断标志位后将其释放（I2C_CTLCLR_SI写1，清除I2C_CTLSET_SI中断标志位）

19.5.2 I2C 从模式数据接收

在地址匹配流程后（其中的读写控制位为0）从机即进入数据接收器模式。从机每接收到一个字节后即回送一个应答信息（应答或非应答，取决于I2C_CTLSET_ACK位的设定），状态位寄存器I2C_STAT更新状态（详见表19-2），并在中断允许时产生中断请求。此时从机将SCL线持续拉低，直到从机软件从I2C_DR寄存器内将数据读出。

19.5.3 I2C 从模式数据发送

在地址匹配流程后（其中的读写控制位为 1）从机即进入数据发送器模式，将 SCL 线持续拉低，从机软件往 I2C_DATA 寄存器内写入一个字节后 SCL 线被释放，随即开始发送。在收到对方的应答信息后更新 I2C_STAT 状态（详见表 19-2），中断允许时产生中断请求。

19.5.4 I2C 从模式通讯终止

在最后一个字节传输完成后如果主机发出停止位，从机也随即终止本次通讯过程，并在中断允许时产生中断请求。

19.5.5 I2C 从模式错误信息

I2C 模块在从模式下如果检测到总线通讯发生错误，I2C_STAT 状态将被置为 0x00，同时模块将释放对总线的控制。

在一个字节传输过程在，当 I2C 侦测到总线产生一个停止位或起始位时，认为通讯发生错误。当检测到停止位导致的总线发生错误时，本机会释放总线，状态位 I2C_STAT=0x00，产生中断标志位（I2C_CTLSET_SI=1），清除此标志位之后总线恢复空闲状态，I2C_STAT=0xF8；当检测到起始位导致的总线发生错误时，本机状态位 I2C_STAT=0x00，产生中断标志位（I2C_CTLSET_SI=1），清除此标志位之后，主机需要发送停止位，使总线恢复空闲状态，I2C_STAT=0xF8。

19.6 I2C 时钟速度计算和设定

I2C 通讯的时钟速度取决于系统时钟频率和分频系数。分频系数通过对 I2C 控制寄存器 I2C_CTLSET 中 CR2/CR1/CR0 三位选择，用户要修改这三位，需要先对这三位的相关位清除（对 I2C_CTLCLR 中 CR2/CR1/CR0 置 1），再对 I2C_CTLSET 中 CR2/CR1/CR0 写值，具体计算方法如下：

$$f_{SCL} = f_{SYS} / DIV$$

其中：

f_{SCL} 为期望的 I2C 通讯 SCL 时钟频率

f_{SYS} 为系统时钟频率

DIV 为系统时钟分频系数，按下表 19-1 选择

应用软件在已知系统时钟频率的前提下，选择合适的分频系数来设定 I2C 通讯的时钟频率。I2C 最高时钟频率不应超过 400Kbps

CR2	CR1	CR0	DIV
0	0	0	256 (266)
0	0	1	224 (234)
0	1	0	192 (202)
0	1	1	160 (170)
1	0	0	960 (970)

1	0	1	120 (130)
1	1	0	60 (70)
1	1	1	保留

表 19-1 产生 I²C 时钟频率时系统时钟分频系数

19.7 I2C 状态信息和中断响应

PT32L031 芯片的 I2C 模块有多个事件可产生同一个中断标识 (I2C_CTLSET_SI)。各事件由 I2C_STAT 寄存器中的位[7:3]状态信息表述, 见表 19-2 说明。

I2C_STAT[7:3]	I ² C总线状态	I2C_CTLSET相关位信息				I ² C模块的下一步应对
		STA	STO	SI	AA	
00000	主模式或从模式被寻址下 I ² C 通讯错误	0	1	0	X	I ² C 总线释放
00001	主模式发送起始位 (START) 完成	X	0	0	X	发送从机地址+写控制位, 接收ACK
00010	主模式发送重复起始位 (Repeated START) 完成	X	0	0	X	同发送起始位 (00001) 发送从机地址+读控制位, 主机转入接收模式
00011	主模式发送从机地址+写控制位完成, 收到ACK响应	0	0	0	X	发送数据字节并接收ACK
		1	0	0	X	发送重复起始位
		0	1	0	X	发送停止位, STO标志置位
		1	1	0	X	停止位后紧接着发送起始位, STO标志清零
00100	主模式发送从机地址+写控制位完成, 无ACK响应	0	0	0	X	发送数据字节并接收ACK
		1	0	0	X	发送重复起始位
		0	1	0	X	发送停止位, STO标志置位
		1	1	0	X	停止位后紧接着发送起始位, STO标志清零
00101	主模式发送数据字节完成, 收到ACK响应	0	0	0	X	发送数据字节并接收ACK
		1	0	0	X	发送重复起始位
		0	1	0	X	发送停止位, STO标志置位
		1	1	0	X	停止位后紧接着发送起始位, STO标志清零
00110	主模式发送数据字节完成, 无ACK响应	0	0	0	X	发送数据字节并接收ACK
		0	0	0	X	发送重复起始位
		1	0	0	X	发送停止位, STO标志置位
		1	0	0	X	停止位后紧接着发送起始位, STO标志清零
00111	主模式在发送地址或数据时总线仲裁丢失	0	0	0	X	I ² C 总线被释放
		1	0	0	X	I ² C 总线空闲时发送起始位
01000	主模式发送从机地址+读控制位完成, 收到ACK响应	0	0	0	0	接收数据, 无ACK响应
		1	0	0	1	接收数据, 有ACK响应
01001	从机地址+读控制位发送完成, 无ACK响应	1	0	0	X	发送重复起始位
		0	1	0	X	发送停止位, STO标志置位
		1	1	0	X	停止位后紧接着发送起始位, STO标志清零
01010	数据字节接收完成, 回送ACK响应	0	0	0	0	接收数据, 无ACK响应
		0	0	0	1	接收数据, 有ACK响应
01011	数据字节接收完成, 回送NACK响应	1	0	0	X	发送重复起始位
		0	1	0	X	发送停止位, STO标志置位
		1	1	0	X	停止位后紧接着发送起始位, STO标志清零
01100	从模式地址+写控制位接	X	0	0	0	接收数据, 回应NACK

	收完成，回送ACK响应	X	0	0	1	接收数据，回应ACK
01101	主模式下总线仲裁丢失， 作为从模式收到地址+写 控制位，并回送ACK响应	X	0	0	0	接收数据，回应NACK
		X	0	0	1	接收数据，回应ACK
01110	收到广播寻址（0x00）， 并回送ACK响应	X	0	0	0	接收数据，回应NACK
		X	0	0	1	接收数据，回应ACK
01111	主模式下总线仲裁丢失， 作为从模式收到广播寻 址，并回送ACK响应	X	0	0	0	接收数据，回应NACK
		X	0	0	1	接收数据，回应ACK
10000	从模式被寻址下收到一个 数据字节，并回送ACK 响应	X	0	0	0	接收数据，回应NACK
		X	0	0	1	接收数据，回应ACK
10001	从模式被寻址下收到一个 数据字节，并回送NACK 响应	0	0	0	0	模块转为非寻址下的从模式，不再识别地址和 广播寻址
		0	0	0	1	模块转为非寻址下的从模式，可识别地址和广 播寻址
		1	0	0	0	模块转为非寻址下的从模式，不再识别地址和 广播寻址；当总线空闲时将发送起始位
		1	0	0	1	模块转为非寻址下的从模式，可识别地址和广 播寻址；当总线空闲时将发送起始位
10010	广播被寻址下收到一个数 据字节，并回送ACK响应	X	0	0	0	接收数据，回应NACK
		X	0	0	1	接收数据，回应ACK
10011	广播被寻址下收到一个数 据字节，并回送NACK响应	0	0	0	0	模块转为非寻址下的从模式，不再识别地址和 广播寻址
		0	0	0	1	模块转为非寻址下的从模式，可识别地址和广 播寻址
		1	0	0	0	模块转为非寻址下的从模式，不再识别地址和 广播寻址；当总线空闲时将发送起始位
		1	0	0	1	模块转为非寻址下的从模式，可识别地址和广 播寻址；当总线空闲时将发送起始位
10100	从模式被寻址下收到停止 位（STOP）或重复起始位 （Repeated START）	0	0	0	0	模块转为非寻址下的从模式，不再识别地址和 广播寻址
		0	0	0	1	模块转为非寻址下的从模式，可识别地址和广 播寻址
		1	0	0	0	模块转为非寻址下的从模式，不再识别地址和 广播寻址；当总线空闲时将发送起始位
		1	0	0	1	模块转为非寻址下的从模式，可识别地址和广 播寻址；当总线空闲时将发送起始位
10101	从模式地址+读控制位接 收完成，回送ACK响应	X	0	0	0	发送最后一个字节，接收ACK应答
		X	0	0	1	发送一个字节，接收ACK应答
10110	主模式下总线仲裁丢失， 作为从模式收到地址+读 控制位，并回送ACK响应	X	0	0	0	发送最后一个字节，接收ACK应答
		X	0	0	1	发送一个字节，接收ACK应答

10111	从模式发送数据字节完成，收到ACK响应	X	0	0	0	发送最后一个字节，接收ACK应答
		X	0	0	1	发送一个字节，接收ACK应答
11000	从模式发送数据字节完成，无ACK响应	0	0	0	0	模块转为非寻址下的从模式，不再识别地址和广播寻址
		0	0	0	1	模块转为非寻址下的从模式，可识别地址和广播寻址
		1	0	0	0	模块转为非寻址下的从模式，不再识别地址和广播寻址；当总线空闲时将发送起始位
		1	0	0	1	模块转为非寻址下的从模式，可识别地址和广播寻址；当总线空闲时将发送起始位
11001	从模式发送最后一个数据字节完成，收到ACK响应	0	0	0	0	模块转为非寻址下的从模式，不再识别地址和广播寻址
		0	0	0	1	模块转为非寻址下的从模式，可识别地址和广播寻址
		1	0	0	0	模块转为非寻址下的从模式，不再识别地址和广播寻址；当总线空闲时将发送起始位
		1	0	0	1	模块转为非寻址下的从模式，可识别地址和广播寻址；当总线空闲时将发送起始位
11111	总线空闲	0	0	0	0	-

表 19-2 I²C 通讯状态信息

当除了总线空闲状态外的任一状态发生时，模块置 I2C_CTLSET_SI 位为 1，应用软件可以设定 I2C 模块的系统中断使能位（NVIC_ISER_I2C=1），响应中断请求并进入中断服务程序。

19.8 寄存器列表

地址	寄存器	描述	备注
0x4000_5400	I2C_CTLSET	I2C控制设定寄存器	I2C_CTLSET说明
0x4000_5404	I2C_STAT	I2C状态寄存器	I2C_STAT说明
0x4000_5408	I2C_DATA	I2C数据寄存器	I2C_DATA说明
0x4000_540C	I2C_ADDR	I2C地址寄存器	I2C_ADDR说明
0x4000_5418	I2C_CTLCLR	I2C控制清除寄存器	I2C_CTLCLR说明

19.9 寄存器描述
19.9.1 I2C 控制设定寄存器 I2C_CTRLSET

(地址: 0x4000_5400)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域		-	-	-	-	-			CR2	EN	STA	STO	SI	AA	CR1	CR0
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] CR0:** **I2C 通讯时钟分频系数 0**
 读
 0b: CR0 为 0
 1b: CR0 为 1
 写
 0b: 无效
 1b: CR0 置 1
- **位[1] CR1:** **I2C 通讯时钟分频系数 1**
 读
 0b: CR1 为 0
 1b: CR1 为 1
 写
 0b: 无效
 1b: CR1 置 1
- **位[2] AA:** **应答位控制**
 读为当前 AA 位值
 0b: 不发送应答位
 1b: 在收到地址字节或数据字节后发送应答位
 写
 0b: 无效
 1b: AA 置 1

- **位[3] SI:** **I²C 中断标志位**
 读
 0b: I²C 模块无中断
 1b: I²C 模块有中断
 写
 0b: 无效
 1b: SI 标志清 0

- **位[4] STO:** **I²C 发送停止位**
 主模式下对此位写 1 后，总线上发出停止位。在停止位完成后，硬件自动将 STO 清 0
 读为当前 STO 位值
 0b: 无停止位
 1b: 发送停止位过程中
 写
 0b: 无效
 1b: STO 置 1

- **位[5] STA:** **I²C 发送起始位**
 该位置 1 后，模块检测总线处于空闲状态时，发送一个起始位，起始位结束后，该位
 被硬件自动清 0
 读为当前 STA 位值
 0b: 无起始位
 1b: 发送起始位过程中
 写
 0b: 无效
 1b: STA 置 1

- **位[6] EN:** **I²C 模块使能控制**
 读为当前 EN 位值
 0b: 禁止 I²C 模块
 1b: 禁止 I²C 模块
 写
 0b: 无效
 1b: EN 置 1

- **位[7] CR2:** **I²C 通讯时钟分频系数 2**
 读
 0b: CR2 为 0
 1b: CR2 为 1
 写
 0b: 无效
 1b: CR2 置 1

- **位[31:8]:** **保留**

19.9.2 I2C 控制设定寄存器 I2C_STAT

(地址: 0x4000_5404)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	STAT				-	-	-	
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0

- 位[2:0] : 保留
- 位[7:3] STAT: I2C 总线状态
共有可能的 27 种状态, 其中的 26 个状态可产生中断请求, 见表 20-2 的说明
- 位[31:8]: 保留

19.9.3 I2C 数据寄存器 I2C_DATA

(地址: 0x4000_5408)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
位域	-	-	-	-	-	-	-	-	DATA								-	-
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW		
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

- 位[2:0] : 保留
- 位[7:0] DATA: 8 位数据
保存总线发送或接收的数据
发送时: 软件写入 8 位数据后随即启动发送流程
接收时: 第一个自动收到的字节为地址的低字节, 软件必须逐个从 I2C_DR 寄存器读取刚接收的数据, 才能顺序接收其它数据字节
- 位[31:8]: 保留

19.9.4 I2C 地址寄存器 I2C_ADDR

(地址: 0x4000_540C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	ADDR							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[2:0] : 保留
- 位[0] GC: 广播寻址使能位
 - 0b: 禁止从机广播寻址
 - 1b: 使能从机广播寻址
- 位[7:1] ADDR: 从机地址
ADDR[7:1]为 7 位寻址匹配地址
- 位[31:8]: 保留

19.9.5 I2C 控制清除寄存器 I2C_CTRLCLR

(地址: 0x4000_5418)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	CR2	EN	STA	STO	SI	AA	CR1	CR0
R/W	R	R	R	R	R	R	R	R	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] CR0: I2C 通讯时钟分频系数 0
 - 写
 - 0b: 无效
 - 1b: CR0 清 0

- **位[1] CR1:** **I2C 通讯时钟分频系数 1**
写
0b: 无效
1b: CR1 清 0
- **位[2] AA:** **应答位控制**
写
0b: 无效
1b: AA 位清 0
- **位[3] SI:** **I2C 中断标志位**
写
0b: 无效
1b: SI 标志清 0
- **位[4] STO:** **I2C 发送停止位**
写
0b: 无效
1b: STO 清 0
- **位[5] STA:** **I2C 发送起始位**
写
0b: 无效
1b: STA 清 0
- **位[6] EN:** **I2C 模块使能控制**
写
0b: 无效
1b: EN 清 0
- **位[7] CR2:** **I2C 通讯时钟分频系数 2**
写
0b: 无效
1b: CR2 清 0
- **位[31:8] :** **保留**

20 ADC

20.1 概述

PT32L031 有一个 12 位的模数转换器 ADC，该 ADC 有 13 个通道，允许 ADC 测量 12 个外部输入引脚电压以及内部 Bandgap 电压。

20.2 特点

- 12 位的模数转换分辨率
- 最多 13 路单端模拟信号输入，6 路差分模拟信号输入
- 片内通道包括：
 - BandGap 电压
- 输入电压范围：0~Vdd
- 参考电压：Vdd
- AD 转换完成可触发中断
- 单次 AD 转换模式、连续 AD 转换模式、定时器溢出信号触发 AD 转换模式、外部触发

20.3 ADC 控制时序

20.3.1 ADC 启动时序

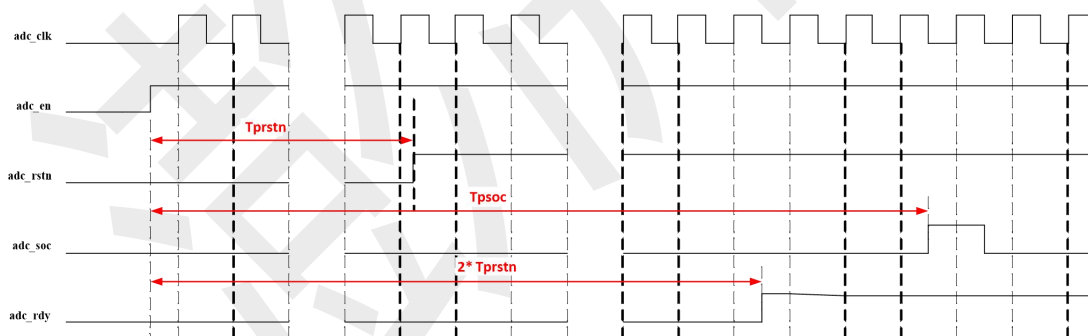


图 20-1 ADC 启动时序

图 20-1 为 ADC 启动时序，其中 T_{prstn} 需要大于 1us， T_{psoc} 需要大于 2us，其中 T_{prstn} 软件可通过配置 ADC_TRSTN 寄存器来配置时间长短，硬件内部会在 $T_{prstn} * 2$ 的时间后置 ADC_STAT 寄存器的 RDY 位为 1，软件读到 RDY 位为 1 后，可启动 ADC 开始转换

20.3.2 ADC 转换时序

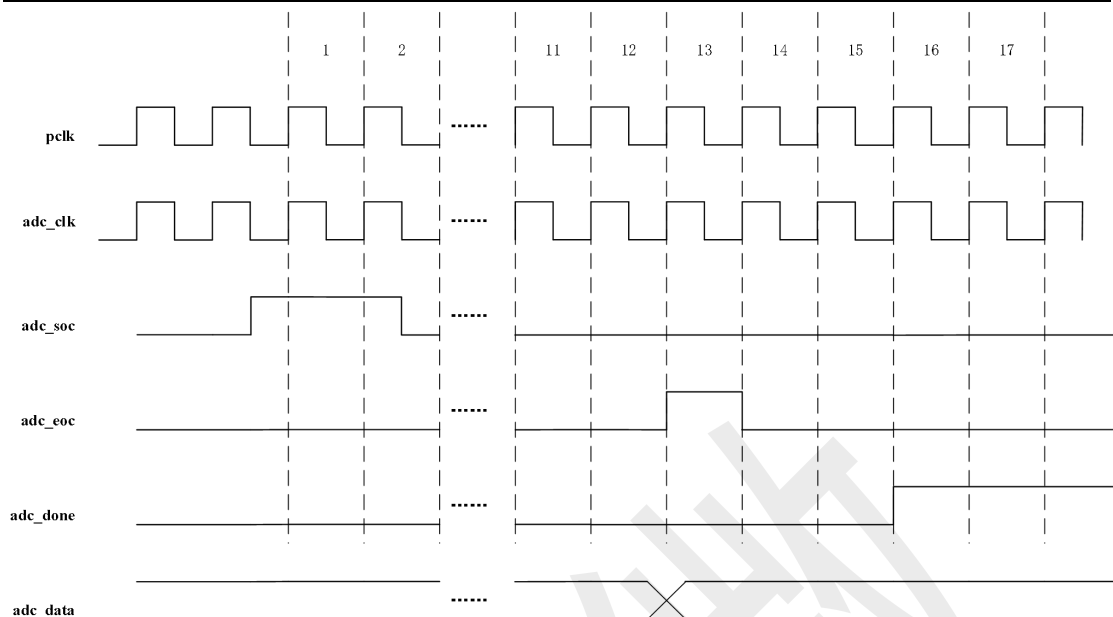


图 20-2 ADC_CLK 不分频时单次转换时序

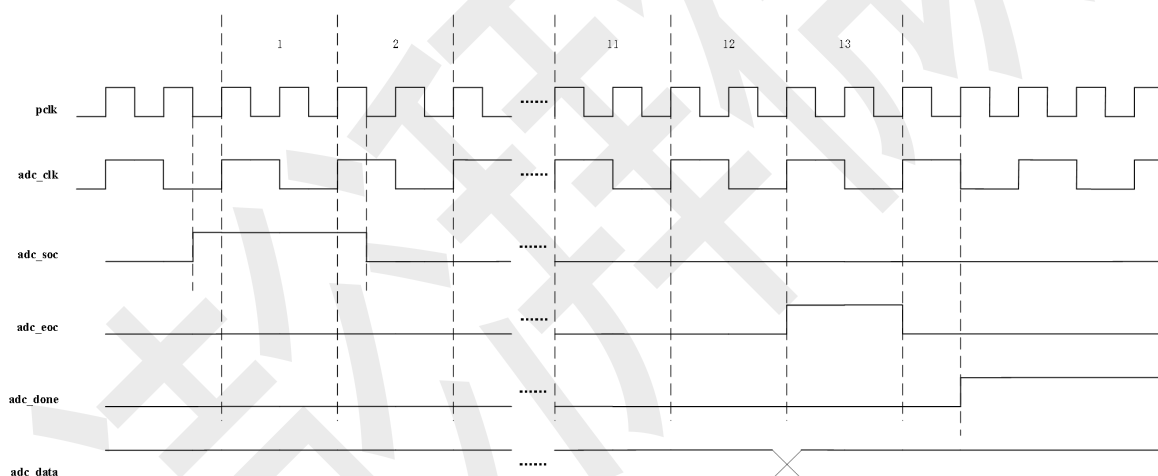


图 20-3 ADC_CLK 二分频时单次转换时序

上图为 ADC 时钟不分频和 ADC 时钟为 pclk 二分频时的单次转换时序图，每次由软件配置 ADC_CON 寄存器的 SOC 位为 1 来启动一次 ADC 转换。

由于 EOC 信号来自于 ADC 模块，需要同步，所以 ADC_DONE 将在 EOC 之后两个 pclk 时钟周期后才置起。

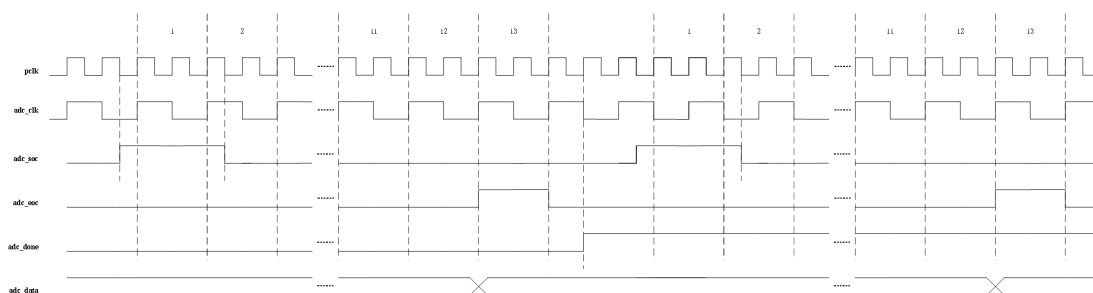


图 20-4 连续转换时序

图 18-4 为 ADC 的连续转换时序图，软件通过 ADC_CON 寄存器配置 ADC 转换模式为连续转换模式后，ADC 模块会自动开始连续转换，软件无需配置 ADC_CON 寄存器的 SOC 位为 1 去启动 ADC 转换，在每次转换完成之后会自动开始下一次的转换。

20.4 用户操作

从以上时序图看出，无论是单次转换或是连续转换，每次使能 ADC 后（ADCON0 寄存器的 AD_EN 置 1），都需要软件延时 2us，以等待 ADC 内部参考源稳定。然后再启动 ADC 转换。

ADC 有三种转换模式：

20.4.1 ADC 单次转换模式

- 1) 软件写 ADC_TRSTN 寄存器配置初始化时间
- 2) 软件写 ADC_CON 寄存器内的 MODE 为选择单次转换模式。
- 3) 软件写 ADC_CON 寄存器内的 SOC_SEL 选择单次转换模式下的启动信号来源
- 4) 软件写 ADC_CON 寄存器内的 EN 使能 ADC 模块，并根据 SOC_SEL 的选择等待 SOC 信号（TIMER 触发或者外部触发）或者配置 SOC 信号（软件写 ADC_CON 寄存器的 SOC 为 1）启动 ADC 转换

20.4.2 连续转换模式

- 1) 软件写 ADC_TRSTN 寄存器配置初始化时间
- 2) 软件写 ADC_CON 寄存器内的 MODE 选择连续转换模式。
- 3) 软件写 ADC_CON 寄存器内的 EN 使能 ADC 模块，ADC 模块将在 ADC_STAT 寄存器的 RDY 位为 1 后自动启动 ADC 转换
2. 若 ADC 中断使能打开（通过中断使能设置寄存器 VIC_IUSER 配置），每次转换完成时还会产生中断信号。

- 位[7] HS: **ADC 内部延时选择**
 0b: 低速采样, 更大延时
 1b: 高速采样, 减少延时
- 位[9:8] CLK_DIV_SEL: **ADC 时钟分频选择**
 00b: 不分频
 01b: 二分频
 10b: 四分频
 11b: 八分频
- 位[10] IE: **ADC 完成转换中断使能**
 0b: 禁止中断
 1b: 允许中断
- 位[11] ALIGN: **ADC 转换结果对齐格式选择**
 0b: 右对齐
 1b: 左对齐
- 位[12] SOC: **ADC 启动转换控制**
 该位只写, 读恒为零, 软件对该位写 1 且 ADC 模式为单次转换和触发源选择为软件寄存器控制时, 触发一次 ADC 转换
- 位[31:13] : **保留**

20.6.2 ADC 复位释放时间寄存器 ADC_TRSTN

(地址: 0x4001_2404)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	TRSTN					
R/W	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

- 位[5:0] TRSTN: **ADC 复位释放时间**
 LSB 为 2 个 PCLK 时钟周期, TRSTN 的时间应大于 1us, 软件需根据当前 PCLK 时钟频率设定
- 位[31:6] : **保留**

21 比较器

21.1 概述

PT32L031 集成了 2 个基本功能完全一致但相互独立的比较器模块，对两路输入信号进行比较，输出相应的比较结果。比较器负端输入固定；正端输入可选。

比较器的输出经数字滤波模块，可以滤除不定状态下产生的毛刺，确保内部中断及输出至芯片管脚的比较结果的可靠性。

21.2 结构框图

下图是比较器模块的内部结构图，其中虚线右侧为数字部分：

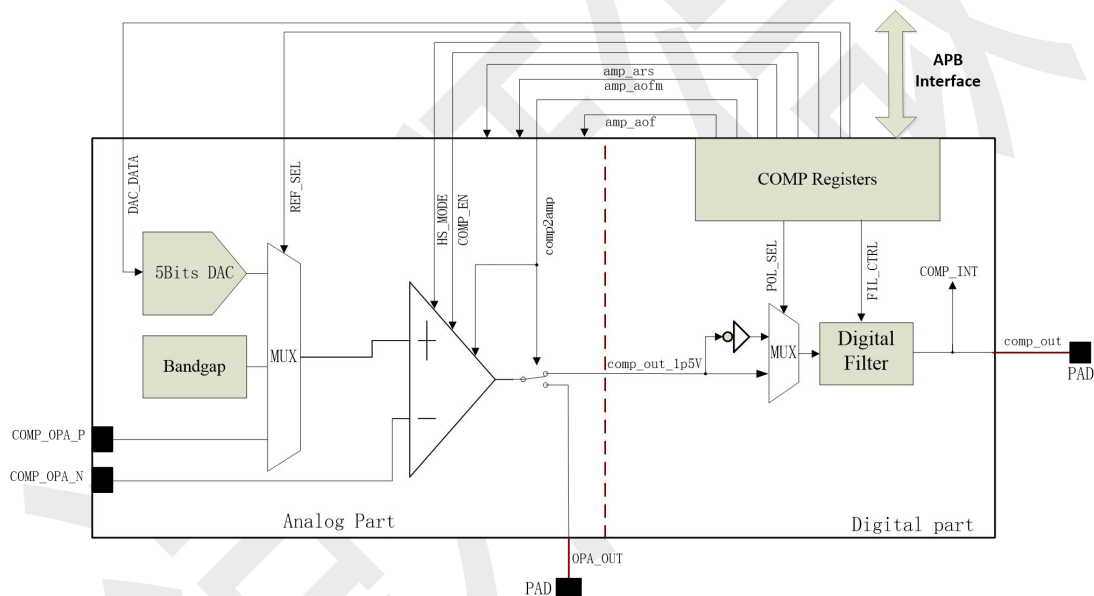


图21-1 比较器结构框图

比较器的输出如上图所示，可以经过数字电路输出到GPIO，也可以直接和GPIO相连，通过GPIO的模拟端口送到PAD。

21.3 功能描述

21.3.1 比较器控制

比较器不使能时，比较器的输出固定为低电平，比较器的正端输入大于负端输入时，比较器输出高电平，否则比较器输出低电平。

比较器负端输入固定为外部引脚输入；正端输入取决于 REF_SEL，可选外部输入引脚、BandGap 电压或者 DAC 的输出。用户在比较器使能后不可以更改正端输入信号源，正端输入信号源只可在比较器未使能时进行修改。

比较器使能后需要经过初始化延时，及比较器传输延时后才能输出有效的比较结果。根据正端输入信号的选择不同，初始化延时也不同。当正端输入信号为外部输入引脚 COMP_OPA_P 或 BandGap 电压时，初始化延时为 15us（取最大值）；当正端输入信号为 DAC 的输出时，初始化延时为 30us（取最大值）。其中，当正端输入信号为 DAC 的输出时，每次用户更改 DAC 输入数据寄存器 COMP_DAC，都需要等待 30us 的初始化延时，确保 DAC 输出电压稳定。初始化延时默认为 60 个 24MHz 时钟周期，若 PCLK 频率发生改变，则需通过配置 COMP_INTCNT 寄存器相应地调整初始化延时周期数，INIT_DELAY 值代表系统时钟的周期数。

比较器使能，经过初始化延时，硬件使能数字滤波模块，可滤除比较器在输入电压穿越时存在的输出毛刺，在经过传输延时后，可以得到有效且稳定的比较结果。初始化阶段，硬件数字滤波模块关闭（filter_en 为低电平），维持数字滤波输出电平值不变。因此确保了在初始化延时过程中，不会误触发中断。

具体的比较器比较控制时序如下所示：

1) 正端输入信号为 com_inn_3v 或 BandGap 电压

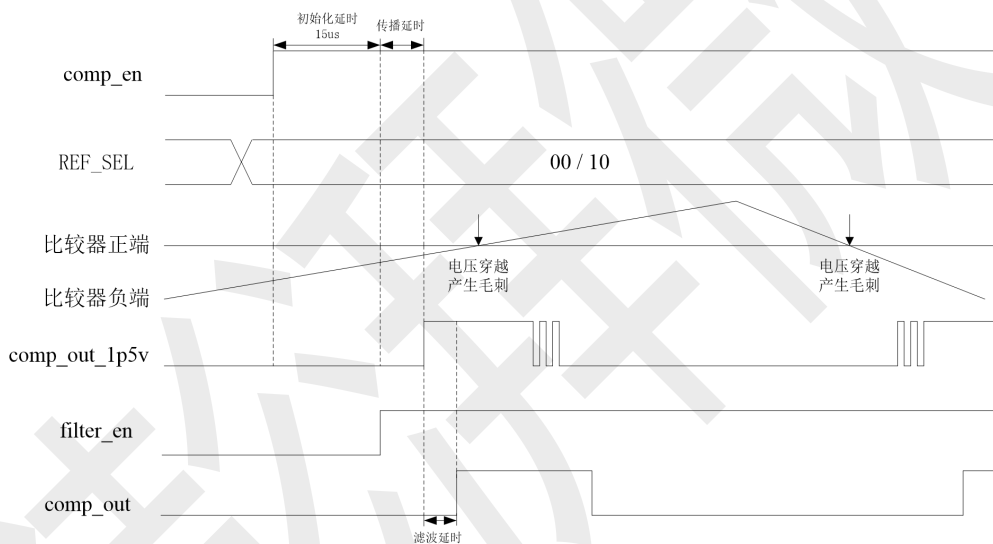


图 21-2 比较器比较控制时序（REF=00/10）

2) 正端输入信号为 DAC 的输出

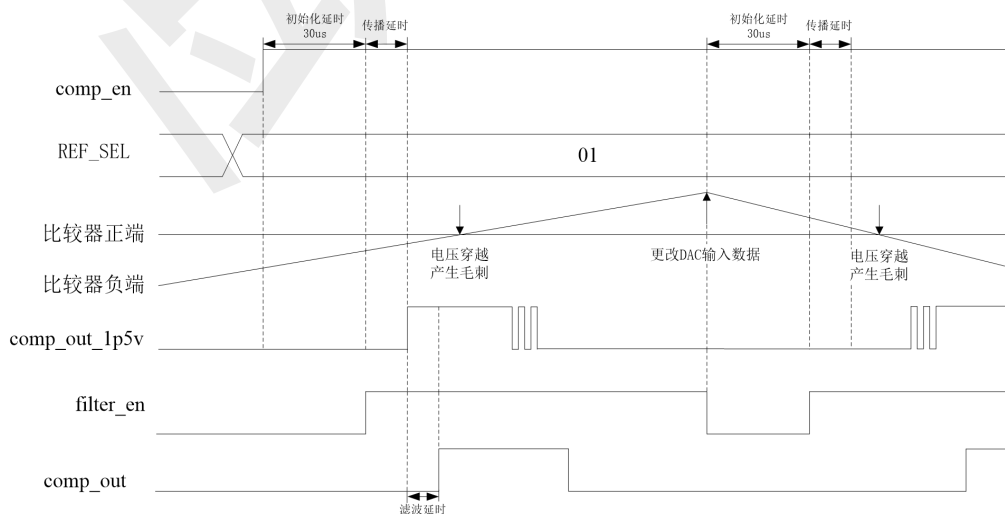


图 21-3 比较器比较控制时序 (REF=01)

可以对比较器的响应时间进行编程，以满足功耗和应用需求之间的权衡。当 HS_MODE 位为 1 时，为高速比较模式，此时传输延时为 100ns~200ns；当 HS_MODE 位为 0 时，为低速和低功耗状态，此时传输延时为 0.8us~2us。

比较器输出电平的极性是可配置的，当 POL_SEL 位为 1 是反相极性时，所有电平和普通极性相反。（注：若没有特别说明，本文描述的是普通极性下的比较器输出）

21.3.2 数字滤波

由于比较器在正负两端输入信号发生穿越时容易产生毛刺，为了滤除穿越过程中产生的不确定状态，故将比较器比较结果输出至数字滤波模块。

数字滤波的原理是在单位时间内，对比较器输出电平进行采样（滤波采样次数可配置），若所有连续采样点的电平值一致，则输出当前电平值，否则采样电平不发生变化，从而滤除不同宽度的毛刺。

滤波输出状态分为高电平和低电平两个状态（默认低电平）。当滤波输出状态为低电平时，判断所有采样点电平值相与的结果，若为 1，则表示采样到高电平，输出状态变为高电平，否则维持当前输出低电平状态不变；当滤波输出状态为高电平时，判断所有采样点电平值相或的结果，若为 0，则表示采样到低电平，滤波输出状态变为低电平，否则维持当前输出高电平状态不变。

FIL_CTRL 位默认为 16 次采样，由于数字滤波存在滤波器延时，若 FIL_CTRL 值设置过大，会导致比较器的响应速度变低，对快速变化的输入信号不响应。所以用户应根据具体使用场景，合理配置 FIL_CTRL 位的值。

21.3.3 中断生成

比较器输出电平经数字滤波后，输出波形产生上升或下降沿时，设置相应的中断使能控制位，比较器可以产生中断。比较器提供三种中断触发模式，分别是电平上升沿触发中断，电平下降沿触发中断，以及电平沿触发中断（无论上升沿或下降沿）。

例如，当比较器的输出上升沿时，比较器上升沿标志位 RIF 置 1，如果比较器输出上升沿中断使控制位 RIE 为 1，将触发一个上升沿中断。对比较器中断标志寄存器的 RIF 位写 1 将清除该标志位。若同时设置 RIE 和 FIE 位为 1，则比较器输出电平无论产生上升沿或下降沿都会触发中断。

下图为比较器反向极性输出波形图，当负端输入大于正端输入时，比较器输出高电平，下降沿检测标志位 FIF 置 1；否则比较器输出低电平，上升沿检测标志位 RIF 置 1。

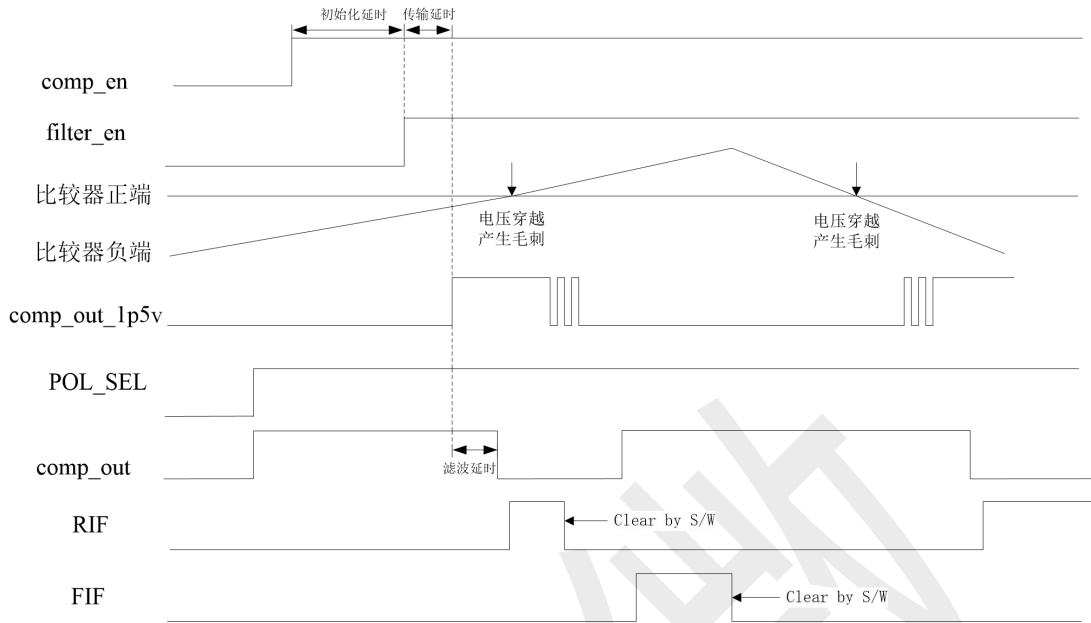


图21-4. 比较器状态更新时序图 (REF=00/10)

状态寄存器内的相关标志位反映发生的是何种中断，中断控制寄存器内的对应位决定了该中断是否被允许响应。多个被允许的中断标志信息通过逻辑“或”后，产生一个统一的串行通讯中断请求送入内核中断控制器，请求系统响应中断服务。下图为比较器的中断映射图，其中 RIE_WAKE 和 FIE_WAKE 控制的中断使能，仅应用于深度睡眠模式。

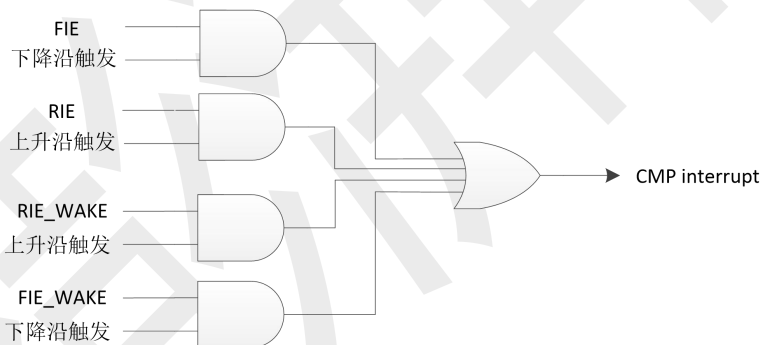


图21-5 比较器中断映射图

21.3.4 睡眠唤醒

在进入睡眠模式前，使能比较器并配置比较器的正端工作模式，可在睡眠模式下，通过改变比较器的输出结果来唤醒系统。

设置 FIE/RIF，比较器输出下降沿/上升沿可以使 MCU 从休眠模式唤醒。如图 23-6 所示

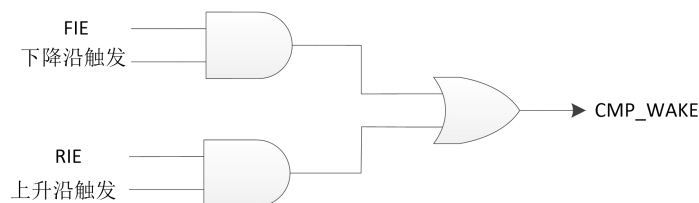
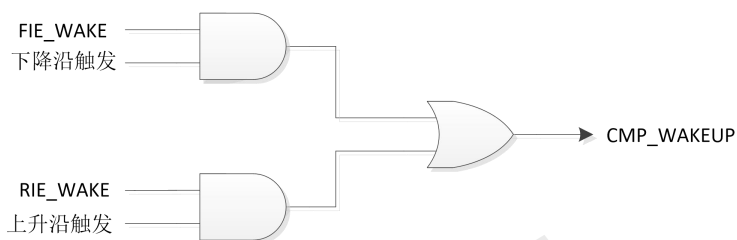


图21-6 比较器休眠模式唤醒信号

在深度睡眠模式下，用户通过设置 FIE_WAKE/RIE_WAKE，比较器输出下降沿/上升沿可以使 MCU 从深度睡眠模式唤醒。


图21-7 比较器深度睡眠模式唤醒信号

用户可在系统唤醒后，在中断程序中，通过中断标志寄存器 COMP_IF 来查询当前触发中断的触发沿类型，并对写 1 清除对应的中断唤醒标志。

21.3.5 OPA 控制

比较器还可以配置为 OPA 模式进行工作，将输入和输出相关的 IO 的模拟功能使能，comp2amp 位置 1，即工作在 OPA 模式。

21.4 寄存器列表

地址	寄存器	描述	备注
0x4000_0800	COMP0_CTRL	比较器 0 控制寄存器	COMP0_CTRL 说明
0x4000_0804	COMP0_AMPCTRL	比较器 0 OPB 控制寄存器	COMP0_AMPCTRL 说明
0x4000_0808	COMP0_DAC	比较器 0 DAC 输入数据寄存器	COMP0_DAC 说明
0x4000_080C	COMP0_IE	比较器 0 中断设置寄存器	COMP0_IE 说明
0x4000_0810	COMP0_IF	比较器 0 中断标志寄存器	COMP0_IF 说明
0x4000_0814	COMP0_INITCNT	比较器 0 初始化延时配置寄存器	COMP0_INITCNT 说明
0x4000_0C00	COMP1_CTRL	比较器 1 控制寄存器	COMP1_CTRL 说明
0x4000_0C04	COMP1_AMPCTRL	比较器 1 OPA 控制寄存器	COMP1_AMPCTRL 说明
0x4000_0C08	COMP1_DAC	比较器 1 DAC 输入数据寄存器	COMP1_DAC 说明
0x4000_0C0C	COMP1_IE	比较器 1 中断设置寄存器	COMP1_IE 说明
0x4000_0C10	COMP1_IF	比较器 1 中断标志寄存器	COMP1_IF 说明
0x4000_0C14	COMP1_INITCNT	比较器 1 初始化延时配置寄存器	COMP1_INITCNT 说明

21.5 寄存器描述
21.5.1 比较器控制寄存器 COMP_CTRL

(地址: COMP0: 0x4800_0800; COMP1: 0x4800_0C00)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	POL_SEL	HS_MODE	FIL_CTRL			REF_SEL		COUT	EN
R/W	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	R	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] EN:** **比较器使能控制**
 0b: 比较器禁止
 1b: 比较器使能
- **位[1] COUT:** **比较器输出结果**
 0b: 正端输入不大于负端输入
 1b: 正端输入大于负端输入
- **位[3:2] REF_SEL:** **比较器正端输入选择**
 00b: 选择 COMP_OPA_P
 01b: 选择 DAC 输出
 10b: 选择 BandGap 电压
 11b: 同默认值
- **位[6:4] FIL_CTRL:** **数字滤波采样控制**
 000b: 无滤波直通
 001b: 2 次采样
 010b: 4 次采样
 011b: 8 次采样
 100b: 16 次采样
 101b: 32 次采样
 110b: 64 次采样
 111b: 同默认值
- **位[7] HS_MODE:** **高速模式使能控制**
 0b: 低速模式
 1b: 高速模式
- **位[8] POL_SEL:** **比较器输出极性选择**
 0b: 正常极性

- 1b: 极性反转
- 位[31:11] : 保留

21.5.2 比较器控制寄存器 COMP_AMPCTRL

(地址: COMP0: 0x4800_0804; COMP1: 0x4800_0C04)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	comp 2amp
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW
复位	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

- 位[0] comp2amp: OPA 模式使能控制
 - 0b: 比较器模式
 - 1b: OPA 模式
- 位[31:1] : 保留

21.5.3 比较器 DAC 输入数据寄存器 COMP_DAC

(地址: COMP0: 0x4800_0808; COMP1: 0x4800_0C08)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	DAC_DATA				
R/W	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[4:0] DAC_DATA: DAC 输入数据
 - 00000b: $1/32 * V_{dd}$
 - 00001b: $2/32 * V_{dd}$
 - 00010b: $3/32 * V_{dd}$
 -
 - 11110b: $31/32 * V_{dd}$

- 11111b: 32/32*Vdd
 - 位[31:5] : 保留

21.5.4 比较器中断控制寄存器 COMP_IE

(地址: COMP0: 0x4800_080C; COMP1: 0x4800_0C0C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	RIE_WAKE	FIE_WAKE	RIE	FIE
R/W	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] FIE: 比较器输出下降沿（正常极性）触发中断使能控制
 0b: 禁止中断
 1b: 允许中断
- 位[1] RIE: 比较器输出上升沿（正常极性）触发中断使能控制
 0b: 禁止中断
 1b: 允许中断
- 位[2] FIE_WAKE: 比较器输出下降沿（正常极性）触发中断使能控制（睡眠模式下）
 0b: 禁止中断
 1b: 允许中断
- 位[3] RIE_WAKE: 比较器输出上升沿（正常极性）触发中断使能控制（睡眠模式下）
 0b: 禁止中断
 1b: 允许中断
- 位[31:4] : 保留

21.5.5 比较器中断标志寄存器 COMP_IF

(地址: COMP0: 0x4800_0810; COMP1: 0x4800_0C10)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	RIF_WAKE	FIF_WAKE	RIF	FIF
R/W	R	R	R	R	R	R	R	R	R	R	R	R	RW1c	RW1c	RW1c	RW1c
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] FIF:** 比较器输出下降沿（正常极性）触发中断标志位
 0b: 未发生下降沿触发中断
 1b: 发生下降沿触发中断，软件可对该位写 1 清零
- **位[1] RIF:** 比较器输出上升沿（正常极性）触发中断标志位
 0b: 未发生上升沿触发中断
 1b: 发生上升沿触发中断，软件可对该位写 1 清零
- **位[2] FIF_WAKE:** 比较器输出下降沿（正常极性）触发中断标志位（睡眠模式下）
 0b: 未发生下降沿触发中断
 1b: 发生下降沿触发中断，软件可对该位写 1 清零
- **位[3] RIF_WAKE:** 比较器输出上升沿（正常极性）触发中断标志位（睡眠模式下）
 0b: 未发生上升沿触发中断
 1b: 发生上升沿触发中断，软件可对该位写 1 清零
- **位[31:4] :** 保留

21.5.6 比较器初始化延时配置寄存器 COMP_INITCNT

(地址: COMP0: 0x4800_0814; COMP1: 0x4800_0C14)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	INIT_DELAY									
R/W	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	1	0	1	1	0	1	0	0	0

- 位[9:0] INIT_DELAY: 初始化延时时间配置
LSB 为一个外设时钟周期，默认为 2.5us (60 个 24MHz 时钟周期) 的延时
- 位[31:10] : 保留

22 CRC 模块

22.1 概述

本模块是 CRC 硬件加速器，省却软件逐位计算所消耗的 CPU 时间。本 CRC 模块的种子和多项式皆可配置，以适应不同通讯标准。16 位 CRC 计算周期为单时钟周期。

22.2 用户操作

- 1) 按实际应用需要配置 CRC_CTRL 寄存器。
- 2) 配置 CRC_POLY 寄存器和 CRC_SEED 寄存器设定多项式和种子。
- 3) 使能 CRC 同时置 CRC_CTRL 寄存器的 CRC_Init 位为 1，装载 CRC_SEED。
- 4) 将需要计算的数据写入 CRC_DIN 寄存器，CRC_DOUT 随即得出目前为止的数据序列 CRC 校验值。

注：每次开始新的 CRC 校验时，需要写一次 CRC_CTRL 寄存器的 CRC_Init 位为 1 装载 CRC_SEED 以开始新的 CRC 校验计算。

22.3 寄存器列表

地址	寄存器	描述	备注
0x4000_3C00	CRC_CTRL	CRC 控制寄存器	CRC_CTRL 说明
0x4000_3C04	CRC_SEED	CRC 种子寄存器	CRC_SEED 说明
0x4000_3C08	CRC_POLY	CRC 多项式寄存器	CRC_POLY 说明
0x4000_3C0C	CRC_DIN	CRC 数据输入	CRC_DIN 说明
0x4000_3C10	CRC_DOUT	CRC 数据输出	CRC_DOUT 说明

22.4 寄存器描述
22.4.1 CRC 控制寄存器 (CRC_CTRL)

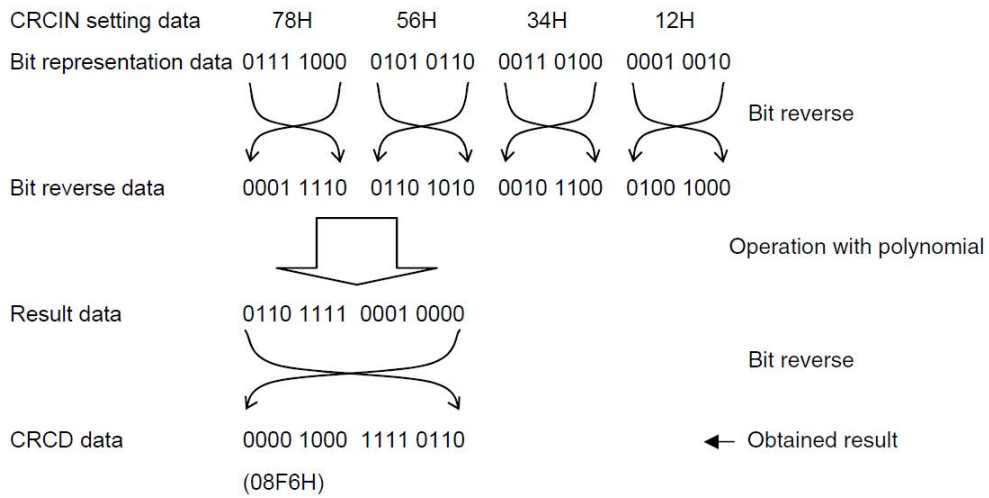
(地址: 0x4000_3C00)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	Bit_ Output Reverse	Byte_ Input_ Reverse	Bit_ Input_ Reverse	CRC_ DIN_ WIDTH	CRC_ Init	CRC_ Enable
R/W	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] CRC_Enable: CRC 模块使能控制**
 0b: CRC 模块禁止
 1b: CRC 模块使能
- **位[1] CRC_Init: CRC 初始化控制**
 该位读恒为 0
 写入 1 即启动初始化, 读返回值始终是 0
- **位[2] CRC_DIN_WIDTH: CRC 输入选择**
 0b: 8 位输入模式
 1b: 16 位输入模式
- **位[3] Bit_Input_Reverse: 输入位序列颠倒**
 无论 16 位和 8 位模式下, 输入位序列颠倒只在字节内进行
 0b: 不颠倒
 1b: 颠倒
- **位[4] Byte_Input_Reverse: 输入字节列颠倒**
 输入字节序列颠倒只对 16-bit 模式有效
 0b: 不颠倒
 1b: 颠倒
- **位[5] Bit_Output_Reverse: 输出位序列颠倒**
 输出位序列颠倒是对整个 16-bit 输出而言, 而不是字节内部颠倒
 0b: 不颠倒
 1b: 颠倒

关于颠倒功能，见下图所示：



22.4.2 CRC 种子寄存器 (CRC_SEED)

(地址: 0x4000_3C04)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	CRC_SEED															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

- 位[15:0] CRC_SEED: CRC 种子

22.4.3 CRC 多项式寄存器 (CRC_POLY)

(地址: 0x4000_3C08)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	CRC_POLY															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

- **位[15:0] CRC_POLY: CRC 多项式寄存器**
 默认值为 16'h8005, 表示 $G(X) = X^{16} + X^{15} + X^2 + 1$;
 如果需要配置多项式为 $G(X) = X^{16} + X^{12} + X^5 + 1$, 则需要写入 16'h1021。
 注: 最高位的因子不需要写入寄存器

22.4.4 CRC 数据输入寄存器 (CRC_DIN)

(地址: 0x4000_3C0C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	CRC_DIN															
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] CRC_DIN: CRC 数据输入寄存器**
 该寄存器只写不可读, 读恒为 0
 注: 8 位模式下, 仅低 8 位有效

22.4.5 CRC 数据输出寄存器 (CRC_DOUT)

(地址: 0x4000_3C10)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	CRC_DOUT															
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[15:0] CRC_DOUT:** **CRC 数据输出寄存器**
 该寄存器只读不可写
 注: 8 位模式下, 仅低 8 位有效

23 模拟功能杂项控制

23.1 概述

PT32L031 有较多的模拟模块，与之关联有一些杂项控制，都汇总到本章节，便于软件管理。这些杂项控制，由寄存器实现，用户通过对寄存器的设置，完成模拟功能的杂项控制。比较独立的 ADC、Comparator 等模拟模块有独立的控制器，不在本章节描述。

23.2 寄存器列表

地址	寄存器	描述	备注
0x4000_1800	LVD_CON	LVD 控制寄存器	LVD_CON 说明
0x4000_1804	LDO_LVD15_CON	1.5V LDO 控制寄存器	LDO_LVD15_CON 说明
0x4000_1808	OSC_CON1	高频 crystal 振荡控制寄存器 1	OSC_CON1 说明
0x4000_180C	OSC_CON2	高频 crystal 振荡控制寄存器 2	OSC_CON2 说明
0x4000_1810	IOSC_24M_CON	内部 24M RC 振荡控制寄存器	IOSC_24M_CON 说明
0x4000_1814	IOSC_32K_CON	内部 32K RC 振荡控制寄存器	IOSC_32K_CON 说明
0x4000_1818	FD_CON	倍频模块控制寄存器	FD_CON 说明
0x4000_181C	DEGLITCH_CON	Deglitch 延时控制寄存器	DEGLITCH_CON 说明
0x4000_1820	GPIO_HS_EN	GPIO 大电流控制寄存器	GPIO_HS_EN 说明
0x4000_1824	MCO_DIV_CON	主时钟输出分频控制寄存器	MCO_DIV_CON 说明

23.3 寄存器描述

23.3.1 LVD 控制寄存器 LVD_CTRL

(地址: 0x4000_1800)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	LVD_TRIP			EN
R/W	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

- **位[0] EN:** **LVD 模块使能控制**
 0b: LVD 模块禁止
 1b: LVD 模块使能
- **位[3:1] LVD_TRIP:** **LVD 阈值选择**
 000b: 1.7V (内部 one-hot 译码: 00000001)
 001b: 2.0V (内部 one-hot 译码: 00000010)
 010b: 2.2V (内部 one-hot 译码: 00000100)
 011b: 2.5V (内部 one-hot 译码: 00001000)
 100b: 2.75V (内部 one-hot 译码: 00010000)
 101b: 3.0V (内部 one-hot 译码: 00100000)
 110b: 3.5V (内部 one-hot 译码: 01000000)
 111b: 4.0V (内部 one-hot 译码: 10000000)
- **位[31:4]:** **保留**

23.3.2 高频 crystal 振荡控制寄存器 1 OSC_CON1

(地址: 0x4000_1808)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	RDY	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	deglitch h_bp_ en	deglitch_ pw_sel		-	-	-	osc_ range	-	ex_20 m_clk _en	osc_en _alc	osc_en
R/W	R	R	R	R	R	RW	RW	RW	R	R	R	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

- 位[0] **osc_en**: **OSC 使能控制**
0b: 不使能
1b: 使能
- 位[1] **osc_en_alc**: **OSC 幅度自动控制**
0b: 不使能
1b: 使能
- 位[2] **ex_20m_clk_en**: **外部输入时钟使能控制**
0b: 不使能
1b: 使能
- 位[3]: **保留**
- 位[4] **osc_range**: **Crystal 20M 时钟频率范围选择**
0b: 4 ~ 10M
1b: 10M ~ 25M
- 位[7:5]: **保留**
- 位[9:8] **deglitch_pw_sel**: **高频 crystal deglitch 延迟选择**
00b: 4ns
01b: 6ns
00b: 8ns
01b: 10ns
- 位[10] **deglitch_bp_en**: **deglitch bypass 控制**
0b: bypass 关闭
1b: bypass 开启
- 位[30:11]: **保留**
- 位[31] **RDY**: **晶振启动就绪标志位**
0b: 启动未就绪
1b: 启动就绪

23.3.3 高频 crystal 振荡控制寄存器 2 OSC_CON2

(地址: 0x4000_180C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
位域	-	-	-	-	-	-	osc_crystal_fb_res_ctrl		-	osc_crystal_c_ctrl				-	osc_drive		
R/W	R	R	R	R	R	R	RW	RW	R	RW	RW	RW	RW	R	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1

- 位[2:0] **osc_drive**: **crystal 驱动强度控制**
 - 000b: x1
 - 001b: x1.5
 - 010b: x2
 - 011b: x2.5
 - 100b: x3
 - 101b: x3.5
 - 110b: x4
 - 111b: x4.5
- 位[3]: **保留**
- 位[6:4] **osc_crystal_c_ctrl**: **crystal 的匹配电容选择控制**
 - 000b: 9pF
 - 001b: 12pF
 - 010b: 15pF
 - 011b: 18pF
 - 100b: 21pF
 - 101b: 24pF
 - 110b: 27pF
 - 111b: 30pF
- 位[7]: **保留**
- 位[9:8] **osc_crystal_fb_res_ctrl**: **crystal 的反馈电阻选择控制**
 - 00b: 1100KΩ
 - 01b: 800KΩ
 - 10b: 500kΩ
 - 11b: 200kΩ
- 位[31:10]: **保留**

23.3.4 内部 24M RC 振荡控制寄存器 IOOSC_24M_CON

(地址: 0x4000_1810)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EN
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

- 位[0] EN: **RC 24M 使能控制**
 0b: 不使能
 1b: 使能
- 位[31:1]: **保留**

23.3.5 内部 32K RC 振荡控制寄存器 IOOSC_32K_CON

(地址: 0x4000_1814)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EN
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	R	R	R	RW
复位	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1

- 位[0] EN: **RC 32K 使能控制**
 0b: 不使能
 1b: 使能
- 位[31:1]: **保留**

23.3.6 倍频模块控制寄存器 FD_CON

(地址: 0x4000_1818)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	EN
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] EN:** 倍频模块使能控制
 0b: 不使能
 1b: 使能
- **位[31:1]:** 保留

23.3.7 Deglitch 延时控制寄存器 DEGLITCH_CON

(地址: 0x4000_181C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	deglitch_PF		deglitch_PB		deglitch_PA		deglitch_comp1		deglitch_comp0	
R/W	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1

- **位[1:0] deglitch_comp0:** 比较器 0 输出延迟控制
 00b: 1us
 01b: 2us
 10b: 3us
 11b: 4us
- **位[3:2] deglitch_comp1:** 比较器 1 输出延迟控制
 00b: 1us
 01b: 2us
 10b: 3us

- 11b: 4us
- 位[5:4] **deglitch_PA:** PA 输出延迟控制
 - 00b: 1us
 - 01b: 2us
 - 10b: 3us
 - 11b: 4us
- 位[7:6] **deglitch_PB:** PB 输出延迟控制
 - 00b: 1us
 - 01b: 2us
 - 10b: 3us
 - 11b: 4us
- 位[9:8] **deglitch_PF:** PF 输出延迟控制
 - 00b: 1us
 - 01b: 2us
 - 10b: 3us
 - 11b: 4us
- 位[31:10]: 保留

23.3.8 HighSink GPIO 控制寄存器 GPIO_HS_CON

(地址: 0x4000_1820)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	PA9_HS	PA10_HS	PB1_HS	PB2_HS
R/W	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] **PB2_HS:** PB2 High Sink 使能控制
该位在 LED 模块使能 (LED_CON 寄存器的 EN 为 1) 时写无效
 - 0b: 不使能
 - 1b: 使能
- 位[1] **PB1_HS:** PB1 High Sink 使能控制
该位在 LED 模块使能 (LED_CON 寄存器的 EN 为 1) 时写无效
 - 0b: 不使能
 - 1b: 使能
- 位[2] **PA10_HS:** PA10 High Sink 使能控制
该位在 LED 模块使能 (LED_CON 寄存器的 EN 为 1) 时写无效
 - 0b: 不使能
 - 1b: 使能

- 位[3] PA9_HS: PA9 High Sink 使能控制
该位在 LED 模块使能 (LED_CON 寄存器的 EN 为 1) 时写无效
0b: 不使能
1b: 使能
- 位[31:4]: 保留

23.3.9 主时钟输出分频控制寄存器 MCO_DIV_CON

(地址: 0x4000_1824)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	MCO_DIV		
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[2:0] MCO_DIV: 主时钟输出分频控制
部分 IO 管脚可以用于输出系统时钟, 具体管脚分配见管脚定义
000b: 不分频
001b: 2 分频
010b: 4 分频
011b: 8 分频
100b: 16 分频
其他: 不分频
- 位[31:3]: 保留

24 FLASH 控制器

24.1 Flash 特性

PT32L031 片上提供 32KB 的 Flash 存储空间用于存放程序代码，Flash 存储空间还有一小块附属的 Flash 信息配置区域，主要用以存储器件出厂时的一些参数设置，用户应用不可见。

Flash 可以按 512 字节为最小单位进行块擦除。通过外部编程工具可将用户代码或数据写入 Flash，程序代码在运行时也可以对 Flash 空间进行自编程。

Flash 数据擦写次数保证 >20,000 次，数据保存 >100 年 @25 度

Flash 基本特性如下：

- Flash 程序区有 32KB 的可用空间，其中 0.5Kbyte 为用户信息区，2Kbyte 为 Bootloader 区域。
- Flash 快速擦除和编程：
 - 512 字节块擦除 4-6ms
 - 单字编程 6-7.5us
- >20,000 擦写次数
- >100 年数据保存（25℃）

24.2 Flash 区域划分

PT32L031 的 Flash 的空间在不同的 Flash 规格下，有着不同的区域划分。

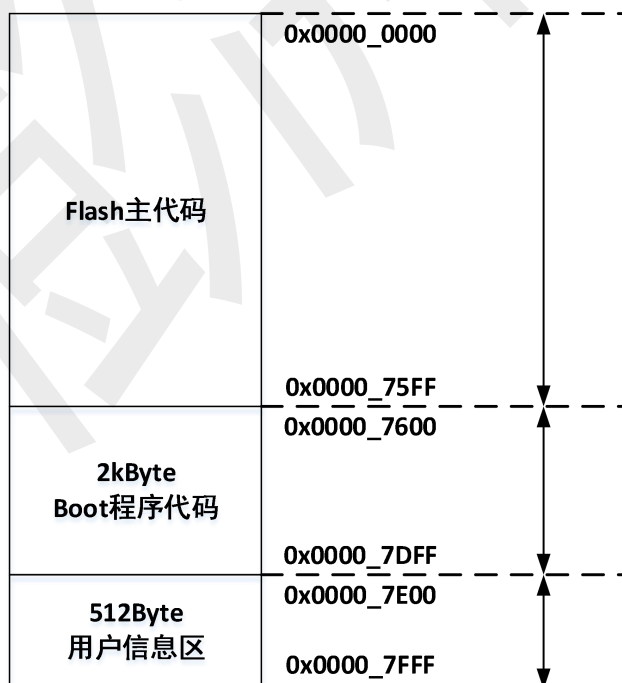


图 24-1 32kByte Flash 规格芯片的 Flash 区域划分

其中 Flash 最后一个 sector（512Byte，地址范围 0x0000_7E00 ~ 0x0000_7FFF）用于存放用户信息配置数据（详见 [NVR 区域说明](#)）

1) 16K-Byte Flash

主程序区域地址范围: 0x0000_0000 ~ 0x0000_3FFF (16kByte)

Boot 程序存放地址范围: 0x0000_7600 ~ 0x0000_7DFF (2kByte)

2) 32K-Byte Flash

主程序区域地址范围: 0x0000_0000 ~ 0x0000_75FF (29.5kByte)

Boot 程序存放地址范围: 0x0000_7600 ~ 0x0000_7DFF (2kByte)

24.3 Flash 操作控制

Flash 控制模块封装了对 Flash 程序区进行擦除和编程的接口, 使得应用程序无需关注 Flash 区在擦除和数据编程时的物理时序和逻辑控制, 只需对相关寄存器进行读写操作, 设定 Flash 操作命令、地址和数据, 即可实现整个 Flash 区的擦除或数据编程工作。

Flash 操作可为:

任意逻辑地址处 Flash 数据读取

任意逻辑地址处 Flash 数据编程写入, 写入前应确保单元数据为 0xFFFFFFFF (擦除状态) 以 512 字节为单位进行块擦除

24.4 FLASH 擦写

Flash 可以通过在电路编程 (in circuit programming, ICP), 在系统编程 (in system programming, ISP) 或者在应用编程 (in application programming, IAP) 进行烧写。

需要注意的是, 当正在进行 Flash 的数据编程写入或块擦除命令时, 无法读取 Flash 内的任何数据。如果是应用程序在运行过程中进行 Flash 的擦除和编程操作, 一旦启动命令, 程序指令即无法从 Flash 处执行。从启动 Flash 编程至 Flash 模块内部编程执行完毕的这段过程中所需执行的任何代码, 必须事先拷贝至 RAM 区, 禁止所有系统中断, 代码跳转至 RAM 区运行, 判断编程结束后再返回, 恢复所需中断响应。

24.5 Flash 内容的安全保护

PT32L031 提供高级别的 Flash 程序区域代码安全保护机制, 通过烧录工具写入 Flash 程序区空间的任何代码数据将被自动保护而无法再次读出。

用户在向 Flash 的用户信息区域写入正确的 FLASH_PROTECT 后, 需要重新上电才能使得 Flash 保护生效。

用户可通过写 Flash 的可配置参数区域的 FLASH_PROTECT 为错误值或者擦除 Flash 最后一个 sector 解除 Flash 保护, 在执行上述两个动作以解除 Flash 保护之前, Flash 会对主代码区域执行一次擦除动作, 以保护 Flash 数据。

24.6 Flash 启动顺序

在用户可配置参数信息区域内有一个 Flash 启动位 (FLASH_BOOT) 选择 (地址: 0x7E04), 用于指示 PC 指针在上电复位后的跳转。

24.6.1 正常启动

若 FLASH_BOOT 不为 32'hC4 时，PC 指针将从 0 地址启动直接去执行主程序。如果 Flash 处于保护状态，不允许通过 swd 接口访问或者写 FLASH（保护状态下可以擦除程序），只有在 Flash 不处于保护状态下，才可以通过 swd 接口对 Flash 进行擦写读取访问。

在运行主程序的过程中，对 Flash 的擦写读取操作均为实际的地址。例如软件需要更新主程序的内容，则需往 FLAR 寄存器内写入的值的范围为 0x0000 ~ 0x75FF，实际就是往 Flash 的 0x0000 ~ 0x75FF 地址写入数据。运行主程序期间，禁止访问和擦写 bootloader 区域的内容。

24.6.2 BootLoader 启动

当 FLASH_BOOT 为 32'hC4 时，上电复位后，PC 将跳转去 bootloader 程序，在运行 bootloader 的程序过程中，如果 Flash 处于保护状态，不允许通过 swd 接口访问或者写 FLASH（保护状态下可以擦除程序），只有在 Flash 不处于保护状态下，才可以通过 swd 接口对 Flash 进行擦写读取访问。

在执行 Bootloader 程序时，对 Flash 的擦写读取操作为映射后的地址。若软件需要更新 bootloader 程序的内容，则需往 FLAR 寄存器内写入的值的范围为 0x0000 ~ 0x0800，实际是往 Flash 的 0x7600 ~ 0x7DFF 地址写入数据；若软件需要更新主程序的内容，则需往 FLAR 寄存器内写入的值的范围为 0x8000 ~ 0xF5FF，实际是往 Flash 的 0x0000 ~ 0x75FF 地址写入数据。若软件需要更新用户信息区的内容，此部分地址未经过映射，软件往 FLAR 寄存器内写入的值的范围为 0x7E00 ~ 0x7FFF。

Bootloader 程序在出厂前固化在 Flash 内。

24.7 Flash 擦写时钟选择

PT32L031 的 Flash 擦写时钟为通过 FLDIV 寄存器的 FLDIV 来对系统时钟进行分频产生 1MHz 的擦写时钟。

24.8 寄存器列表

地址	寄存器	描述	备注
0x4000_0000	FLCMD	FLASH 命令寄存器	FLCMD 说明
0x4000_0004	FLISR	FLASH 中断标志寄存器	FLISR 说明
0x4000_0008	FLIER	FLASH 中断使能寄存器	FLIER 说明
0x4000_000C	FLAR	FLASH 地址寄存器	FLAR 说明
0x4000_0010	FLDR	FLASH 编程数据寄存器	FLDR 说明
0x4000_0028	FLDIV	FLASH 擦写时钟分频寄存器	FLDIV 说明

24.9 寄存器描述
24.9.1 FLASH 命令寄存器 FLCMD

(地址: 0x4000_0000)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	KEYCODE															
R/W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	AINC	-	DPSTB	-	-	MODE	NWS			-	CMD		START	
R/W	R	R	RW	R	RW	R	R	RW	RW	RW	RW	R	R	R	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] START: Flash 操作启动**
 写操作
 0b: 无效
 1b: 启动 Flash 操作命令, 对该位写 1 时, 务必对
 读操作
 0b: Flash 操作命令执行完毕
 1b: Flash 操作命令执行中 (执行完毕后硬件自动清 0)
- **位[3:1] CMD: Flash 操作命令**
 000b: 单字 (32 位) 写入编程
 001b: 块擦除
 其他: 保留
- **位[7:5] NWS: Flash 操作等待周期**
 设定 Flash 命令操作时的等待周期
 000b: 0 周期等待
 001b: 1 周期等待
 其他 保留

- **位[8] MODE:** **Flash 操作模式**
 0b: Flash 读模式
 1b: Flash 写模式
 注: 当该位为 0 时, 任何对 Flash 的写操作或者擦除均无效并且无任何的警告或标志位产生;
- **位[10:9] :** **保留**
- **位[11] DPSTB:** **Flash 深度休眠控制**
 0b: Flash 深度休眠禁止
 1b: Flash 深度休眠使能
 注: 当软件执行的程序不在 Flash 区而是在别的存储区 (比如 SRAM) 时, 才能对该位进行置 1 操作, 不然 Flash 将无法再通过软件唤醒。
- **位[12] :** **保留**
- **位[13] AINC:** **Flash 地址自动递增**
 0b: Flash 读写一次后 FLAR 寄存器值保持不变
 1b: Flash 读写一次后 FLAR 寄存器值自动加 4
- **位[15:14] :** **保留**
- **位[31:16] :** **寄存器写操作安全密码输入**
 针对此寄存器低 16 位区域的任何写入操作, 必须在此高 16 位区域根据 FLAR 寄存器内的地址同步写入操作的安全密码, 对应用户信息配置区域的安全密码为 0xC5AE, 其他区域的安全密码为 0xADEB。

24.9.2 FLASH 中断状态寄存器 FLISR

(地址: 0x4000_0004)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	ADDR_ ERR	-	CMD_ ERR	-	PROTEC T_ERAS E_END	ERASE _END	WRITE_ END
R/W	R	R	R	R	R	R	R	R	R	RW1c	R	RW1c	R	RW1c	RW1c	RW1c
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[0] WRITE_END:** **Flash 写操作命令完成标志位**
 0b: 写命令未完成
 1b: 写命令完成, 软件可对该位写 1 将其清 0
- **位[1] ERASE_END:** **Flash 块擦除操作命令完成标志位**
 0b: 块擦除命令未完成
 1b: 块擦除命令完成, 软件可对该位写 1 将其清 0
- **位[2] PROTECT_ERASE_END:** **Flash 解除保护擦除主程序操作完成标志位**
 0b: Flash 解除保护时擦除主程序未完成
 1b: Flash 解除保护时擦除主程序完成, 软件可对该位写 1 将其清 0
- **位[3]:** **保留**
- **位[4] CMD_ERR:** **Flash 操作命令错误标志位**
 0b: 命令正确
 1b: 命令错误, 软件可对该位写 1 将其清 0
- **位[5] :** **保留**
- **位[6] ADDR_ERR:** **Flash 地址错误标志位**
 0b: 地址正确
 1b: 地址错误 (Flash 操作地址超限), 软件可对该位写 1 将其清 0
- **位[31:7] :** **保留**

24.9.3 FLASH 中断使能寄存器 FLIER

(地址: 0x4000_0008)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	-	ADDR_ ERRE	-	CMD_ ERRE	-	PROTEC T_ERAS E_ENDE	ERASE _ENDE	WRITE_ ENDE
R/W	R	R	R	R	R	R	R	R	R	RW	R	RW	R	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- 位[0] WRITE_ENDE: Flash 写操作命令完成中断使能
设定 FLISR 寄存器的 WRITE_END 标志位是否引发中断请求
0b: 中断禁止
1b: 中断使能
- 位[1] ERASE_ENDE: Flash 擦除操作命令完成中断使能
设定 FLISR 寄存器的 ERASE_END 标志位是否引发中断请求
0b: 中断禁止
1b: 中断使能
- 位[2] PROTECT_ERASE_ENDE: Flash 解除保护擦除主程序操作完成中断使能
设定 FLISR 寄存器的 PROTECT_ERASE_END 标志位是否引发中断请求
0b: 中断禁止
1b: 中断使能
- 位[3]: 保留
- 位[4] CMD_ERRE: Flash 操作命令错误标志位
设定 FLISR 寄存器的 CMD_ERR 标志位是否引发中断请求
0b: 中断禁止
1b: 中断使能
- 位[5] : 保留
- 位[6] ADDR_ERRE: Flash 地址错误标志位
设定 FLISR 寄存器的 ADDR_ERR 标志位是否引发中断请求
0b: 中断禁止
1b: 中断使能
- 位[31:7] : 保留

24.9.4 FLASH 地址寄存器 FLAR

(地址: 0x4000_000C)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	BADDR	
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[1:0] BADDR:** **Flash 程序区字节寻址地址**
Flash 操作总是以 4 字节的字为单位进行, 故此两位无效 (软件写该位时无效)
- **位[15:2] ADDR:** **Flash 程序区字寻址逻辑地址**
总计 32KB 的 Flash 程序区主代码空间等价于 8K 的逻辑字空间 (每字 4 字节), 每个逻辑地址处对应将寻址 4 字节的 Flash 数据
- **位[31:15] :** **保留**

24.9.5 FLASH 编程数据寄存器 FLDR

(地址: 0x4000_0010)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	FLDR															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	FLDR															
R/W	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- **位[31:0] FLDR:** **Flash 编程数据**
 内容为 0 时的数据位将被写入 Flash, 1 的数据位在编程时被忽略
 编程前确保 Flash 单元中的数据已被擦除为 0xFFFF_FFFF

24.9.6 FLASH 编程数据寄存器 FLDIV

(地址: 0x4000_0028)

位	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
位域	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
R/W	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
复位	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

位	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
位域	-	-	-	-	-	-	-	-	FLDIV							
R/W	R	R	R	R	R	R	R	R	RW	RW	RW	RW	RW	RW	RW	RW
复位	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0

- **位[7:0] FLDIV:** **Flash 擦写时钟分频系数**
 DIV 用于对系统时钟进行分频, 产生 1MHz 的 Flash 擦写时钟。DIV 的配置公式

$$freq_{TICK} = \frac{freq_{SYS}}{FLDIV}$$

注: $freq_{TICK}$ 为分频后的 1MHz, $freq_{SYS}$ 为系统时钟频率。

25 修改历史

版本	修改日期	修改内容
V1.0	2020/3/25	初版
V1.1	2021/12/2	1) 修改已知文字错误