

ARM Cortex®-M0

32-位 微控制器

NuMicro™ M058S 系列 技术参考手册

本文档中所描述的信息是芯唐科技的独家知识产权，未经芯唐许可，不得转载。

芯唐提供本文档仅供以基于NuMicro™ 微控制器的系统设计为目的参考。芯唐不为错误或遗漏承担任何责任。

所有数据和规格如有变更恕不另行通知。

获取更多信息或解决疑问，请与芯唐科技联系。

www.nuvoton.com

目录

1 概述	10
2 特性	11
3 缩写	14
3.1 缩写列表	14
4 选型表和管脚图	15
4.1 NuMicro™ M058S 系列选型指南	15
4.2 管脚图	16
4.2.1 TSSOP20 pin	16
4.2.2 QFN 33-pin	17
4.2.3 LQFP 48-pin	18
4.2.4 LQFP 64-pin	19
4.3 管脚描述	20
5 方块图	24
5.1 NuMicro™ M058S 系列方块图	24
6 功能描述	25
6.1 ARM® Cortex®-M0 内核	25
6.2 系统管理器	27
6.2.1 概述	27
6.2.2 系统复位	27
6.2.3 系统电源架构	28
6.2.4 系统存储器映射	29
6.2.5 系统存储器映射表	30
6.2.6 系统定时器 (SysTick)	63
6.2.7 系统定时器控制寄存器映射	63
6.2.8 嵌套向量中断控制器 (NVIC)	67
6.2.9 系统控制块 (SCB)	109
6.3 时钟控制器	117
6.3.1 概述	117
6.3.2 系统时钟和SysTick时钟	120
6.3.3 掉电模式时钟	120
6.3.4 分频器输出	121
6.3.5 寄存器映射	123
6.3.6 寄存器描述	124
6.4 Flash存储控制器(FMC)	142
6.4.1 概述	142
6.4.2 特性	142

6.4.3 框图	142
6.4.4 FMC 存储器组织	144
6.4.5 数据 Flash	146
6.4.6 用户配置	147
6.4.7 启动选择	149
6.4.8 在应用编程(IAP)	150
6.4.9 在系统编程(ISP)	151
6.4.10 ISP控制流程	151
6.4.11 通过向量重映射多启动	154
6.4.12 寄存器映射	155
6.4.13 寄存器描述	157
6.5 通用 I/O (GPIO)	166
6.5.1 概述	166
6.5.2 特性	166
6.5.3 基本配置	166
6.5.4 功能描述	167
6.5.5 GPIO 中断和唤醒功能	168
6.5.6 寄存器映射	169
6.5.7 寄存器描述	174
6.6 定时器控制器(TMR)	188
6.6.1 概述	188
6.6.2 特性	188
6.6.3 框图	189
6.6.4 基本配置	190
6.6.5 功能描述	190
6.6.6 寄存器映射	193
6.6.7 寄存器描述	195
6.7 PWM发生器和捕捉定时器	205
6.7.1 概述	205
6.7.2 特性	206
6.7.3 框图	207
6.7.4 基本设定	209
6.7.5 功能描述	209
6.7.6 寄存器映射	221
6.7.7 寄存器描述	222
6.8 看门狗定时器(WDT)	251
6.8.1 概述	251

6.8.2 特性	251
6.8.3 框图	251
6.8.4 基本配置	252
6.8.5 功能描述	252
6.8.6 寄存器映射	254
6.8.7 寄存器描述	255
6.9 窗口看门狗定时器 (WWDT)	258
6.9.1 简介	258
6.9.2 特性	258
6.9.3 框图	258
6.9.4 基本配置	259
6.9.5 功能描述	259
6.9.6 寄存器映射	261
6.9.7 寄存器描述	262
6.10 串口控制器 (UART)	267
6.10.1 概述	267
6.10.2 特性	267
6.10.3 框图	267
6.10.4 基本配置	270
6.10.5 功能描述	270
6.10.6 寄存器映射	283
6.10.7 寄存器描述	284
6.11 I ² C 总线控制器 (I ² C)	306
6.11.1 概述	306
6.11.2 特性	306
6.11.3 基本配置	307
6.11.4 功能描述	307
6.11.5 I ² C 协议	308
6.11.6 I ² C 协议寄存器	310
6.11.7 工作模式	314
6.11.8 寄存器映射	322
6.11.9 寄存器描述	324
6.12 串行外围设备接口 (SPI)	334
6.12.1 概述	334
6.12.2 特性	334
6.12.3 框图	335
6.12.4 基本配置	335

6.12.5 功能描述	336
6.12.6 时序图	343
6.12.7 编程例子	346
6.12.8 寄存器映射	350
6.12.9 寄存器描述	351
6.13 模数转换 (ADC)	365
6.13.1 概述	365
6.13.2 特性	365
6.13.3 框图	366
6.13.4 基本配置	367
6.13.5 功能描述	367
6.13.6 寄存器映射	374
6.13.7 寄存器描述	375
7 电器特性	385
8 封装尺寸	386
8.1 LQFP-64 (7x7mm, Thickness 1.4 mm, Footprint 2.0mm)	386
8.2 LQFP-48 (7x7x1.4mm ² Footprint 2.0mm)	387
8.3 QFN-33 (5X5 mm ² , Thickness 0.8mm, Pitch 0.5 mm)	388
8.4 TSSOP-20 (4.4x6.5 mm)	389
9 修订历史	390

插图目录

图 4.1-1 NuMicro™ M058S 系列命名规则	15
图 4.2-1 NuMicro™ M058S TSSOP20 管脚图.....	16
图 4.2-2 NuMicro™ M058S 系列 QFN-33 管脚图.....	17
图 4.2-3 NuMicro™ M058S 系列 LQFP-48 管脚图	18
图 4.2-4 NuMicro™ M058S 系列 LQFP-64 管脚图	19
图 5.1-1 NuMicro™ M058S 系列方块图	24
图 6.1-1 功能框图.....	25
图 6.2-1 NuMicro M058S 系列电源架构图.....	28
图 6.3-1 时钟发生器框图.....	117
图 6.3-2 时钟源控制器概览(1/2).....	118
图 6.3-3 时钟源控制器概览(2/2).....	119
图 6.3-4 系统时钟框图	120
图 6.3-5 SysTick 时钟控制框图.....	120
图 6.3-6 分频器的时钟源.....	121
图 6.3-7 分频器框图	122
图 6.4-1 Flash 存储器控制框图.....	143
图 6.4-2 Flash 存储器组织结构	145
图 6.4-3 Flash 存储器结构	146
图 6.4-4 启动选择 (BS) 用于上电动作.....	149
图 6.4-5 APROM 和 LDRROM 启动的程序执行范围	149
图 6.4-6 IAP 使能时代码的执行范围	151
图 6.4-7 当 CBS[0] = 1 BS 位启动选择流程示例	152
图 6.4-8 ISP 流程示例.....	153
图 6.4-9 通过向量页重映射多启动	155
图 6.5-1 推挽输出.....	167
图 6.5-2 开漏输出.....	167
图 6.5-3 准双向 I/O 模式	168
图 6.6-1 定时器控制器框图	189
图 6.6-2 定时器控制器时钟源	189
图 6.6-3 Continuous Counting 连续计数模式	191
图 6.6-4 内部-定时器触发捕捉时序	192
图 6.7-1 PWM 发生器 0 时钟源控制	207
图 6.7-2 PWM 发生器 0 结构框图.....	207

图 6.7-3 PWM 生成器 2 时钟源控制	208
图 6.7-4 PWM 发生器 2 结构框图	208
图 6.7-5 PWM-定时器内部比较器输出图例	209
图 6.7-6 PWM定时器操作时序	210
图 6.7-7 PWM周期中断发生时序波形图	210
图 6.7-8中心对齐模式输出波形	211
图 6.7-9 PWM中心对齐中断发生时序图	212
图 6.7-10 PWM双缓存图解	213
图 6.7-11 PWM控制器输出占空比	213
图 6.7-12 对-PWM带死区发生器操作输出	214
图 6.7-13中心对齐模式下PWM触发ADC Flag (PWMxTF)时序图	215
图 6.7-14边缘对齐模式下PWM触发ADC Flag (PWMxTF)时序图	216
图 6.7-15中心对齐模式下PWM触发ADC转换时序图	217
图 6.7-16捕捉操作时序	218
图 6.7-17 A 组PWM-定时器中断结构图	219
图6.8-1 看门狗定时器时钟控制	251
图6.8-2 看门狗定时器框图	252
图6.8-3 看门狗定时器超时溢出间隔和复位周期时序图	253
图6.9-1窗口看门狗定时器时钟控制	258
图6.9-2 窗口看门狗定时器时钟框图	258
图 6.9-3 窗口看门狗定时器复位和重载过程	260
图 6.10-1 串口时钟控制框图	268
图 6.10-2 串口控制器框图	268
图 6.10-3 自动流控模块框图	270
图 6.10-4 UART CTS 自动流控使能	275
图 6.10-5 UART RTS 自动流控使能	276
图 6.10-6 UART RTS 软件控制的流控	276
图 6.10-7 IrDA 控制模块框图	277
图 6.10-8 IrDA TX/RX 时序图	278
图 6.10-9 LIN 的帧结构	278
图 6.10-10 RS-485 自动方向模式下RTS 驱动电平	280
图6.10-11 RS-485 RTS 软件控制下的驱动电平	281
图 6.10-12 RS-485 帧结构	282
图 6.11-1 I ² C 总线时序	307

图 6.11-2 I ² C 协议	308
图 6.11-3 起始 (START) 和停止 (STOP) 信号条件	308
图 6.11-4 I ² C 总线上的位传输	309
图 6.11-5 I ² C 总线上的应答信号	309
图 6.11-6 主机向从机传输数据	310
图 6.11-7 主机向从机读取数据	310
图 6.11-8 I ² C 数据移位方向	311
图 6.11-9 I ² C 超时计数器框图	313
图 6.11-10 根据 I ² C 当前状态控制 I ² C 总线	314
图 6.11-11 主机发送模式控制流程	315
图 6.11-12 从机模式控制流程	317
图 6.11-13 广播呼叫模式	319
图 6.11-14 EEPROM 随机读取	320
图 6.11-15 EEPROM 随机读协议	321
图 6.12-1 SPI 框图	335
图 6.12-2 SPI 主机模式应用框图	336
图 6.12-3 SPI 从机模式应用框图	336
图 6.12-4 一次传输 32-位长度 (主模式下)	337
图 6.12-5 字节重排序	339
图 6.12-6 字节休眠时序波形	339
图 6.12-7 FIFO 模式框图	340
图 6.12-8 SPI 主机模式下的时序	343
图 6.12-9 SPI 主机模式下的时序 (交替 SPI 时钟相位)	344
图 6.12-10 SPI 从机模式下的时序	345
图 6.12-11 SPI 从机模式下的时序 (交替 SPI 时钟相位)	346
图 6.13-1 AD 控制器框图	366
图 6.13-2 ADC 时钟控制	367
图 6.13-3 单一模式转换时序图	369
图 6.13-4 单周期扫描模式下使能信道的时序图	371
图 6.13-5 使能信道的连续扫描模式时序图	372
图 6.13-6 A/D 转换结果监控逻辑框图	373
图 6.13-7 A/D 控制器中断	374
图 6.13-8 ADC 单端输入电压和转换结果图	376
图 6.13-9 ADC 差分输入和转换结果图	376

列表

表 3.1-1 缩写列表.....	14
表 4.1-1 NuMicro™ M058S 系列选型指南	15
表 6.2-1 片上模块的地址空间分配	30
表 6.2-2 异常模型.....	68
表 6.2-3 系统中断映射向量表	69
表 6.2-4 向量表格式.....	69
表 6.3-1 掉电模式控制	126
表 6.4-1 M058S 系列不同功能列表 (FMC)	142
表 6.4-2 支持的启动选项.....	150
表 6.4-3 ISP 命令表	153
表 6.8-1 看门狗定时器超时溢出间隔周期选择	253
表 6.9-1 窗口看门狗定时器预分频值选择	259
表 6.10-1 UART 控制器波特率公式表.....	271
表 6.10-2 UART 控制器波特率参数设置表	271
表 6.10-3 UART 控制器波特率寄存器设置表.....	272
表 6.10-4 UART 控制器中断源和标志列表	273
表 6.10-5 UART 线控制的数据位和停止位长度设置.....	274
表 6.10-6 UART 线控制的奇偶校验位设置	274
表 6.11-1 I ² C 状态码描述	312

1 概述

NuMicro™ M058S 系列32-位微控制器内嵌ARM® Cortex®-M0内核，适用于工业控制和需要多种通讯接口的应用领域。M058S系列包含如下的型号：M058SSAN，M058SLAN和M058SZAN。

NuMicro™ M058S 系列最高可运行至50MHz之高，可在2.5V~5.5V，-40℃ ~ 85℃范围内工作，因此可应用于各种各样需要高性能CPU的工业控制和应用的场合。M058S系列提供32KB闪存，4KB数据闪存，4KB ISP闪存和4KB SRAM。

有许多系统级外设功能，如I/O端口，Timer，UART，SPI，I²C，PWM，ADC，看门狗定时器和窗口看门狗。这些有用的功能使M058S系列在广泛的应用中更加强大。

此外，NuMicro™ M058S 系列带有 ISP（在系统编程）功能， ICP（在电路编程）功能和IAP（在应用编程）功能。这些功能允许用户更新程序存储器而无需将芯片从实际最终产品中拿出。

2 特性

- 内核
 - ARM® Cortex®-M0 内核，最高可达 50 MHz
 - 一个 24-位系统定时器
 - 支持低功耗睡眠模式
 - 一个单周期32-位硬件乘法器
 - 嵌套向量中断控制器NVIC支持32个中断输入，每个中断有4个优先级
 - 支持串行线调试（SWD）接口，2个观察点/4个断点
- 宽操作电压范围：2.5V ~ 5.5V
- 存储器
 - 32KB Flash 用于存储用户程序 (APROM)
 - 4 KB Flash 用于数据存储 (Data Flash)
 - 4 KB Flash 用于存储ISP升级时的引导代码 (LDROM)
 - 4 KB SRAM 用于内部高速暂存存储器 (SRAM)
- 时钟控制
 - 可编程的系统时钟源
 - 22.1184 MHz 内部振荡器
 - 4~24 MHz 外部晶振输入
 - 10 kHz 低功耗振荡器，用于看门狗定时器和睡眠模式下的唤醒
 - PLL 支持 CPU 最高运行在 50 MHz
- I/O 端口
 - 在LQFP-64封装中，最多支持55个通用 I/O 端口(GPIO)
 - 四种 I/O 工作模式：
 - ◆ 准双向模式
 - ◆ 推挽输出模式
 - ◆ 开漏输出模式
 - ◆ 高阻抗输入模式
 - 可选TTL输入或者施密特触发输入
 - I/O管脚可被配置为边沿/电平触发模式的中断源
 - 可在上电复位（POR）后配置I/O模式
- 定时器
 - 提供四通道32-位定时器；每个定时器都带有一个8-位预分频计数器和24-位向上计数器
 - 每组定时器有独立的时钟源
 - 24-位定时器数据可通过TDR（定时器数据寄存器）读出
 - 提供单次，周期和翻转操作模式
 - 提供事件计数功能
 - 提供外部捕获/复位计数器功能
 - 额外的功能：
 - ◆ 两个来自外部触发和内部10KHz的定时器时钟源
 - ◆ TIMER 唤醒功能
 - ◆ 外部捕获输入源可以由TxEX进行选择
 - ◆ 翻转模式输出引脚可以由TxEN或TMx进行选择

◆ Inter-Timer 触发模式

- 看门狗定时器 (WDT)
 - 多路时钟源选择
 - 支持在掉电模式和休眠模式下唤醒
 - 看门狗定时器溢出时可选择产生中断或复位
 - 可选择的溢出复位延时周期时间
- 窗口看门狗定时器(WWDT)
 - 6-位向下计数器, 带有11-bit预分频, 用于宽泛的窗口选择
- PWM
 - 两个内置的16-bit PWM 发生器, 提供四路PWM输出或两对互补的PWM输出
 - 每个PWM发生器可以单独选择时钟源, 时钟分频器, 8位时钟预分频器, 和死区发生器
 - PWM中断与PWM周期同步
 - 16位捕捉定时器(与PWM定时器共享)支持捕获输入信号的上升沿/下降沿
 - 支持捕获中断
 - 额外功能
 - ◆ 内部 10 kHz PWM 时钟源
 - ◆ 极性翻转功能
 - ◆ 中心对齐功能
 - ◆ 使能定时器 duty 中断功能
 - ◆ 两种 PWM 中断周期/ duty类型选择
 - ◆ 周期/ duty触发ADC功能
- UART
 - 波特率发生器可编程
 - 接收器和发送器支持缓冲, 均带有15字节的FIFO缓冲
 - 可选的流控功能(CTS 和 RTS)
 - 支持 IrDA(SIR) 功能
 - 支持 RS485 功能
 - 支持 LIN 功能
- SPI
 - 支持 SPI主机/从机模式
 - 全双工同步串行数据传输
 - 提供3线模式
 - 传输数据长度从8位到32位可变
 - 可设置MSB或LSB优先的传输模式
 - 32位传输模式下支持字节挂起模式
 - 额外功能
 - ◆ PLL时钟源
 - ◆ 4级深度FIFO缓冲, 以为更好的性能和灵活的 SPI 传输模式
- I²C
 - 多达两组I²C模块
 - 支持主/从模式
 - 主从机之间双向数据传输

- 多主机总线支持（无中心主机）
- 多主机同时发送数据时进行仲裁，总线上串行数据不会被损坏
- 串行时钟同步使得不同比特率的设备可以通过一条串行总线传输数据
- 串行时钟同步可用作握手机制来暂停和恢复串行传输
- 可编程配置的时钟可适应多样化的传输速率控制
- 支持多地址识别 (4组从机地址带屏蔽选项)
- ADC
 - 12位逐次逼近式模数转换器
 - 最多8信道单端输入或者4通道差分输入
 - 支持单次转换模式/突发模式/单周期扫描模式/连续扫描模式
 - 差分模式下，支持2的补码/无符号格式的转换结果
 - 每个通道有独立的存放转换结果的寄存器
 - 支持转换结果监测（或比较），用于阈值电压检测
 - 转换开始可由软件或外部引脚触发
 - 额外功能
 - ◆ A/D转换由PWM中心对齐触发或者边沿对齐触发
 - ◆ PWM 触发延迟功能
- ISP (在系统编程) 和 ICP (在电路编程)
- IAP (在应用编程)
- 内嵌一个温度传感器，1℃分辨率
- 欠压检测(Brown-out Detector)
 - 支持4级检测电压: 4.4V/3.7V/2.7V/2.2V
 - 支持欠压中断和复位选择
- 96-bit唯一ID号
- 低电压复位 (Low Voltage Reset)
 - 阈值电压: 2.0V
- 工作温度
 - -40℃~85℃
- 封装
 - 无铅封装 (RoHS)
 - 64-pin LQFP, 48-pin LQFP, 33-pin QFN, 20-pin TSSOP

3 缩写

3.1 缩写列表

缩写	描述
ADC	Analog-to-Digital Converter
APB	Advanced Peripheral Bus
AHB	Advanced High-Performance Bus
BOD	Brown-out Detection
FIFO	First In, First Out
FMC	Flash Memory Controller
GPIO	General-Purpose Input/Output
HCLK	The Clock of Advanced High-Performance Bus
HIRC	22.1184 MHz Internal High Speed RC Oscillator
HXT	4~24 MHz External High Speed Crystal Oscillator
IAP	In Application Programming
ICP	In Circuit Programming
ISP	In System Programming
LDO	Low Dropout Regulator
LIRC	10 kHz internal low speed RC oscillator (LIRC)
NVIC	Nested Vectored Interrupt Controller
PCLK	The Clock of Advanced Peripheral Bus
PLL	Phase-Locked Loop
PWM	Pulse Width Modulation
SPI	Serial Peripheral Interface
SPS	Samples per Second
TMR	Timer Controller
UART	Universal Asynchronous Receiver/Transmitter
UCID	Unique Customer ID
WDT	Watchdog Timer
WWDT	Window Watchdog Timer

表 3.1-1 缩写列表

4 选型表和管脚图

4.1 NuMicro™ M058S 系列选型指南

Part Number	APROM (KB)	RAM (KB)	Data Flash (KB)	ISP ROM (KB)	I/O	Timer (32-Bit)	Connectivity			PWM (16-bit)	ADC (12-bit)	WDT	WWDT	ISP/ICP/IAP	Package	Operating Temperature Range(°C)
							UART	SPI	I ² C							
M058SFAN	32	4	4	4	14	4	1	1	1	1	2	✓	✓	✓	TSSOP20	-40 to +85
M058SZAN	32	4	4	4	26	4	1	1	1	2	5	✓	✓	✓	QFN33	-40 to +85
M058SLAN	32	4	4	4	42	4	1	1	2	4	8	✓	✓	✓	LQFP48	-40 to +85
M058SSAN	32	4	4	4	55	4	1	1	2	4	8	✓	✓	✓	LQFP64	-40 to +85

表 4.1-1 NuMicro™ M058S 系列选型指南

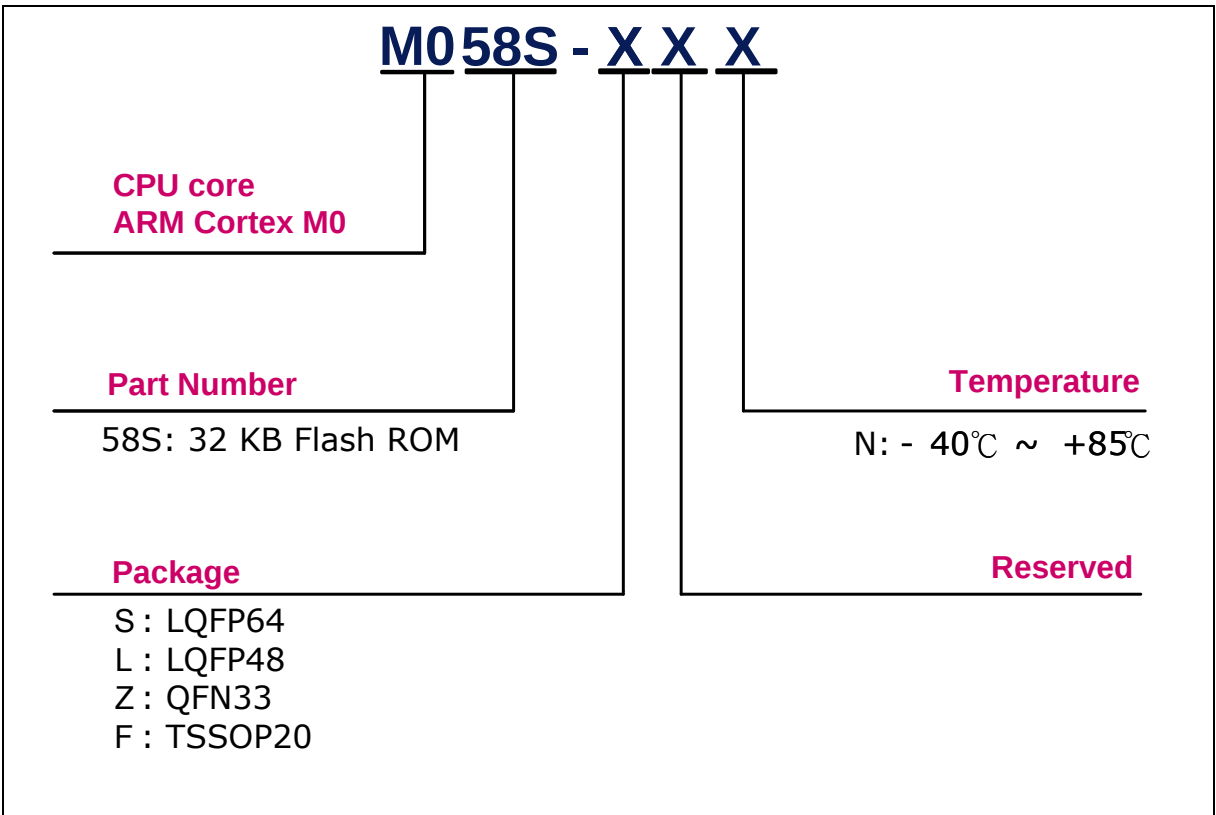


图 4.1-1 NuMicro™ M058S 系列命名规则

4.2 管脚图

4.2.1 TSSOP20 pin

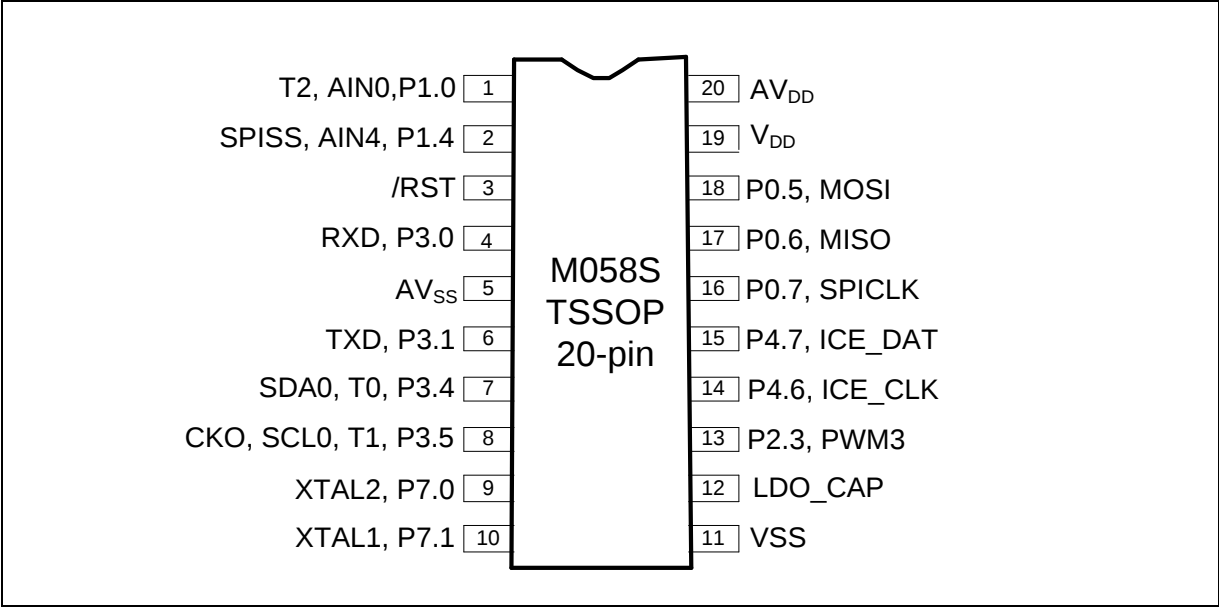


图 4.2-1 NuMicro™ M058S TSSOP20 管脚图

4.2.2 QFN 33-pin

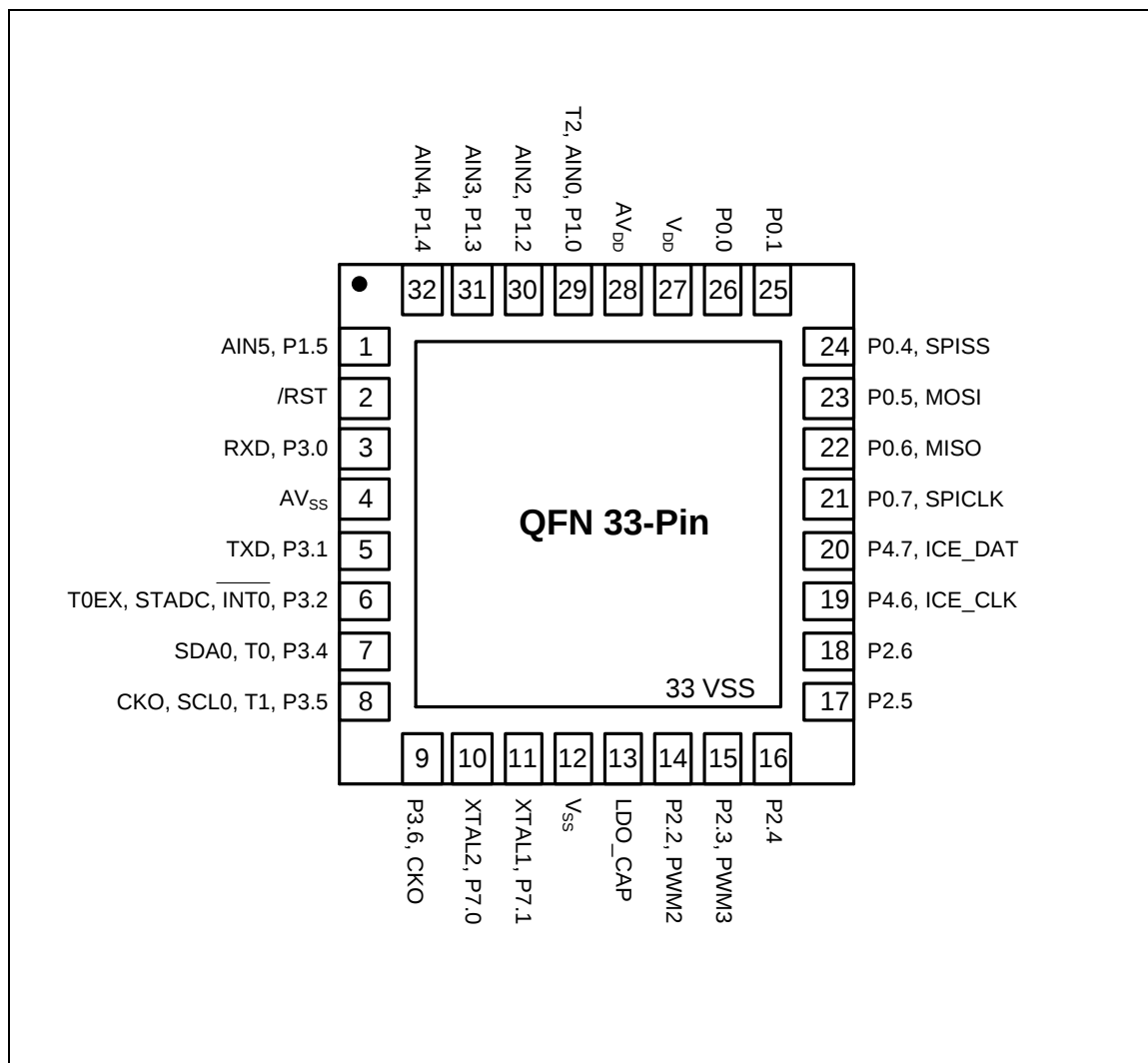


图 4.2-2 NuMicro™ M058S 系列 QFN-33 管脚图

4.2.3 LQFP 48-pin

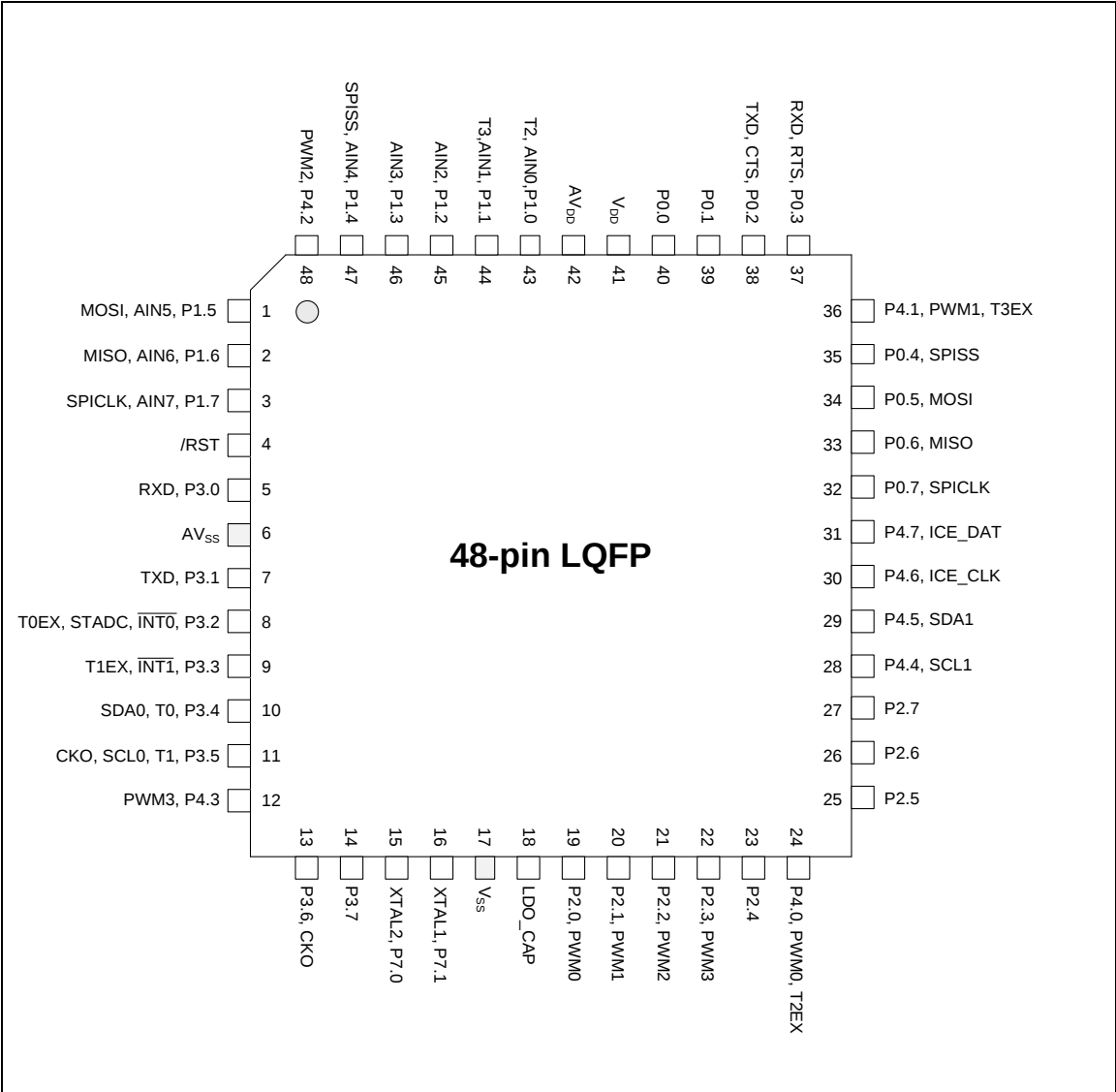


图 4.2-3 NuMicro™ M058S 系列 LQFP-48 管脚图

4.2.4 LQFP 64-pin

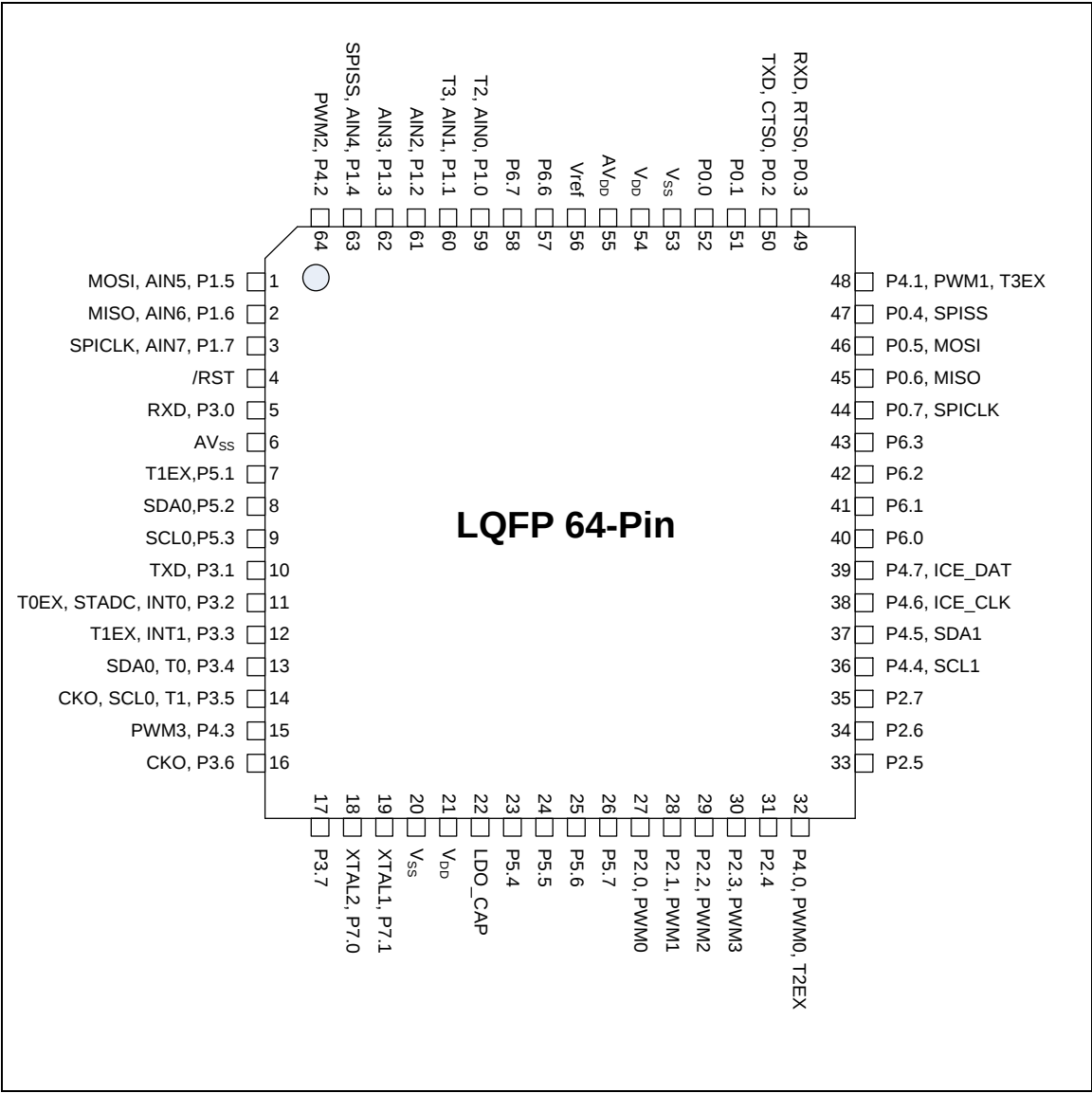


图 4.2-4 NuMicro™ M058S 系列 LQFP-64 管脚图

4.3 管脚描述

管脚编号				符号	复用功能			类型 ^[1]	描述
TSSOP 20	QFN 33	LQFP 48	LQFP 64		1	2	3		
19	27	41	21 54	V _{DD}				P	电源供应管脚，为IO端口、内部PLL电路LDO源和数字电路提供电源
11	12 33	17	20 53	V _{SS}				P	数字电源地
20	28	42	55	AV _{DD}				P	内部模拟电路电源输入管脚
5	4	6	6	AV _{SS}				P	模拟电路地管脚
	NC	NC	56	V _{ref}				P	ADC 参考电压输入管脚
12	13	18	22	LDO _CAP				P	LDO 输出管脚 备注: 必须外接1uF电容
3	2	4	4	/RST				I (ST)	/RST管脚为施密特触发输入管脚，用于硬件设备复位。当该管脚上接入“低”电位，并保持768个内部22 MHz RC 高速晶振时钟周期后，芯片复位。 /RST管脚具有内部上拉电阻，对该管脚通过外部电容接地，就可以完成上电复位。 注意: 建议在/RST引脚上使用10kΩ上拉电阻和10 uF电容器
	26	40	52	P0.0				I/O	PORT0: 通用 I/O 口，可由软件配置成四种模式。Port0为多路复用引脚，包括 CTS1, RTS1, CTS0, RTS0, SPISS, MOSI, MISO 和 SPICLK。 引脚 SPISS, MOSI, MISO 和 SPICLK 用作SPI功能。 CTS: 用于UART 清除发送输入引脚。 RTS: 用于UART 请求发送输出引脚。 RXD/TXD 引脚用于UART 功能。
	25	39	51	P0.1				I/O	
	NC	38	50	P0.2	CTS		TXD ^[2]	I/O	
	NC	37	49	P0.3	RTS		RXD ^[2]	I/O	
	24	35	47	P0.4	SPISS ^[2]			I/O	
18	23	34	46	P0.5	MOSI ^[2]			I/O	
17	22	33	45	P0.6	MISO ^[2]			I/O	
16	21	32	44	P0.7	SPICLK ^[2]			I/O	

管脚编号				符号	复用功能			类型 ^[1]	描述
TSSOP 20	QFN 33	LQFP 48	LQFP 64		1	2	3		
1	29	43	59	P1.0	T2	AIN0		I/O	PORT1: 通用 I/O 口，可由软件配置成四种模式。Port1 为多路复用引脚，包括 T2, T3, SPISS0, MOSI, MISO 和 SPICLK。
	NC	44	60	P1.1	T3	AIN1		I/O	
	30	45	61	P1.2		AIN2		I/O	
	31	46	62	P1.3		AIN3		I/O	
2	32	47	63	P1.4	SPISS ^[2]	AIN4		I/O	引脚 SPISS0, MOSI, MISO, 和 SCLK 用于 SPI 功能。
	1	1	1	P1.5	MOSI ^[2]	AIN5		I/O	引脚 AIN0~AIN7 用于 12 位 ADC 功能。
	NC	2	2	P1.6	MISO ^[2]	AIN6		I/O	引脚 T2/T3 用于 Timer2/3 的外部事件计数器输入功能。
	NC	3	3	P1.7	SPICLK ^[2]	AIN7		I/O	
	NC	19	27	P2.0	PWM0 ^[2]			I/O	
	NC	20	28	P2.1	PWM1 ^[2]			I/O	
	14	21	29	P2.2	PWM2 ^[2]			I/O	PORT2: 通用 I/O 口，可由软件配置成四种模式。它还有另一种功能。 引脚 PWM0~PWM3 用于 PWM 功能。
13	15	22	30	P2.3	PWM3 ^[2]			I/O	
	16	23	31	P2.4				I/O	
	17	25	33	P2.5				I/O	
	18	26	34	P2.6				I/O	PORT3: 通用 I/O 口，可由软件配置成四种模式。Port3 为多路复用引脚，包括 RXD, TXD, /INT0, /INT1, T0 和 T1。
	NC	27	35	P2.7				I/O	
4	3	5	5	P3.0	RXD ^[2]			I/O	
6	5	7	10	P3.1	TXD ^[2]			I/O	
	6	8	11	P3.2	/INT0	STADC	T0EX	I/O	引脚 RXD/TXD 用于 UART 功能。
	NC	9	12	P3.3	/INT1		T1EX	I/O	引脚 SDA0/SCL0 用于 I ² C0 功能。
7	7	10	13	P3.4	T0	SDA0		I/O	CKO: HCLK 时钟输出。
8	8	11	14	P3.5	T1	SCL0	CKO ^[2] ₁	I/O	引脚 STADC 用于 ADC 外部触发输入。
	9	13	16	P3.6		CKO		I/O	引脚 T0/T1 用于 Timer0/1 外部

管脚编号				符号	复用功能			类型 ^[1]	描述
TSSOP 20	QFN 33	LQFP 48	LQFP 64		1	2	3		
	NC	14	17	P3.7				I/O	事件计数器输入功能。 引脚 T0EX/T1EX 用于 Timer0/1外部捕获/复位触发输入功能。
	NC	24	32	P4.0	PWM0 ^[2]		T2EX	I/O	PORT4: 通用 I/O 口，可由软件配置成四种模式。Port4 为多路复用引脚，包括PWM0-3, SCL1, SDA1, ICE_CLK 和 ICE_DAT。 引脚 ICE_CLK/ICE_DAT 用于 JTAG-ICE 功能。 PWM0-3 功能可以使用P2.0-P2.3 或 P4.0-P4.3。
	NC	36	48	P4.1	PWM1 ^[2]		T3EX	I/O	
	NC	48	64	P4.2	PWM2 ^[2]			I/O	
	NC	12	15	P4.3	PWM3 ^[2]			I/O	
	NC	28	36	P4.4		SCL1		I/O	引脚 T2EX/T3EX 用于 Timer2/3外部捕获/复位触发输入功能。 注意：建议在ICE_CLK和 ICE_DAT引脚上使用100kΩ上拉电阻
	NC	29	37	P4.5		SDA1		I/O	
14	19	30	38	P4.6	ICE_CLK			I/O	
15	20	31	39	P4.7	ICE_DAT			I/O	
	NC	NC	7	P5.1	T1EX			I/O	PORT5: 通用 I/O 口，可由软件配置成四种模式。Port5 为多路复用引脚，包括 T0EX, T1EX, SDA0 和 SCL0。 引脚 T0EX/T1EX Timer0/1外部捕获/复位触发输入功能。 引脚 SDA0/SCL0 用于 I ² C0 功能。
	NC	NC	8	P5.2	SDA0			I/O	
	NC	NC	9	P5.3	SCL0			I/O	
	NC	NC	23	P5.4				I/O	
	NC	NC	24	P5.5				I/O	
	NC	NC	25	P5.6				I/O	
	NC	NC	26	P5.7				I/O	
	NC	NC	40	P6.0				I/O	PORT6: 通用 I/O 口，可由软件配置成四种模式。
	NC	NC	41	P6.1				I/O	
	NC	NC	42	P6.2				I/O	
	NC	NC	43	P6.3				I/O	
	NC	NC	57	P6.6				I/O	

管脚编号				符号	复用功能			类型 ^[1]	描述
TSSOP 20	QFN 33	LQFP 48	LQFP 64		1	2	3		
	NC	NC	58	P6.7				I/O	
9	10	15	18	P7.0	XTAL2			I/O, O	PORT7: 通用 I/O 口，可由软件配置成四种模式。Port7 为多路复用引脚，包括 XTAL。
10	11	16	19	P7.1	XTAL1			I/O, I(ST)	XTAL:外部4~24 MHz (高速) 晶振引脚

注 1: I/O 类型描述。I: 输入，O: 输出，I/O: 准双向，D: 开漏，P: 电源管脚，ST: Schmitt 触发器。

注 2: PWM0 ~ PWM3, RXD, TXD, RXD1, TXD1, SCL1, SDA1 和 CKO 可以被配置为不同的引脚，然而某一时刻只有一个引脚可以配置为该功能，例如：软件不能同时分配 RXD 到 P0.3 和 P3.0。

5 方块图

5.1 NuMicro™ M058S 系列方块图

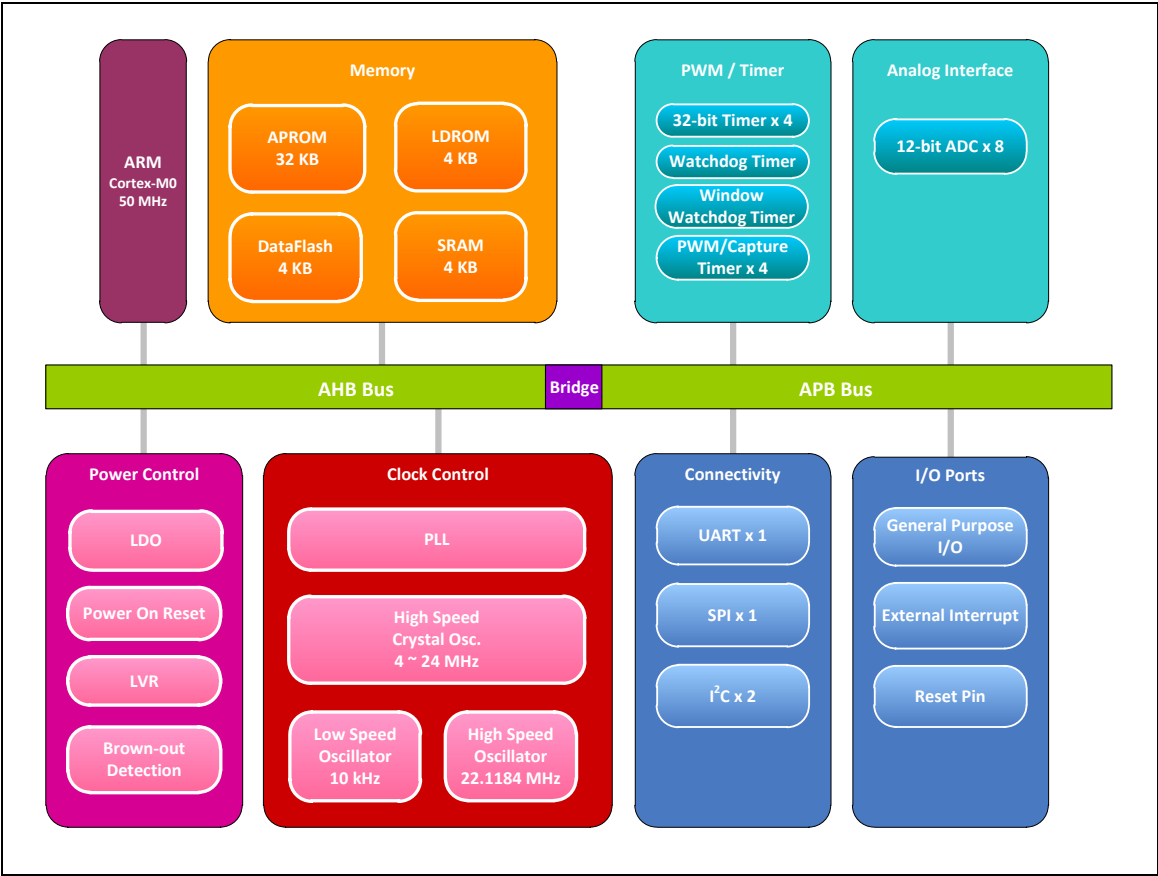


图 5.1-1 NuMicro™ M058S 系列方块图

6 功能描述

6.1 ARM® Cortex®-M0 内核

Cortex®-M0处理器是32位多级可配置的RISC处理器。它有AMBA AHB-Lite接口和嵌套向量中断控制器（NVIC），具有可选的硬件调试功能，可以执行Thumb指令，并与其它Cortex-M系列兼容。该系列处理器支持两种操作模式Thread模式和Handler模式。当有异常发生时，处理器进入Handler模式。异常返回只能在Handler模式下发生。当复位时，处理器会进入Thread模式，处理器也可在异常返回时进入到Thread模式。下图显示了处理器内核的各个功能模块。

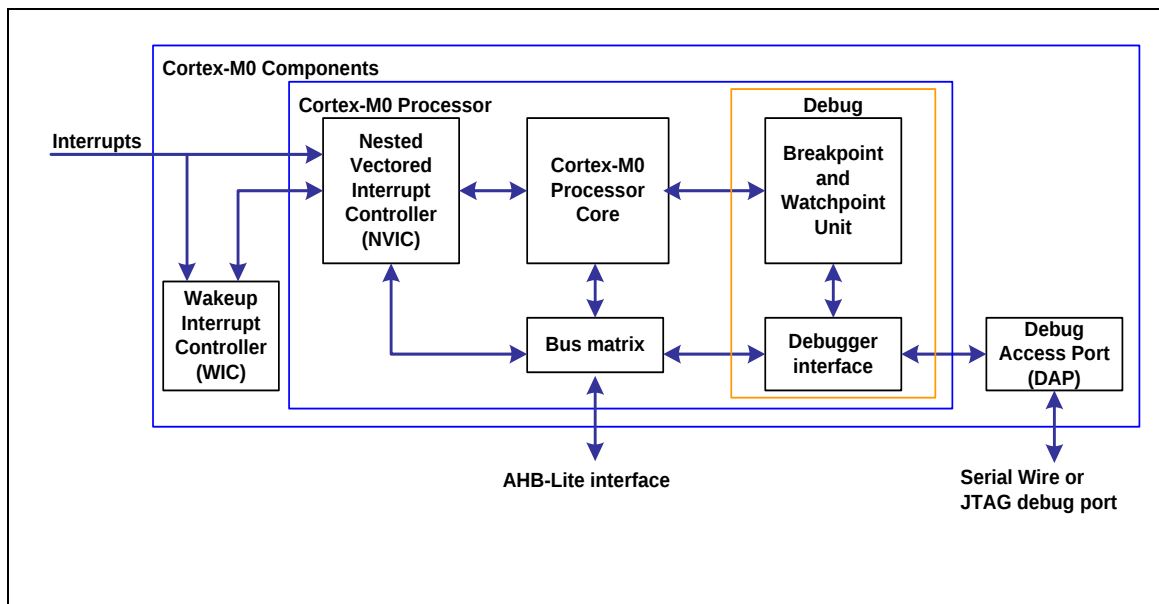


图 6.1-1 功能框图

设备提供：

- 低门数处理器：
 - ARMv6-M Thumb® 指令集
 - Thumb-2 技术
 - ARMv6-M 兼容 24-bit SysTick 定时器
 - 32-bit 硬件乘法器
 - 系统接口支持小端（little-endian）数据访问
 - 具有确定性，固定延迟，中断处理的能力
 - 加载/存储多个数据和多周期乘法指令可被终止然后重新开始，从而实现快速中断处理
 - C应用程序二进制接口的异常兼容模式（C-ABI）。这个 ARMv6-M 的模式允许用户使用纯C函数实现中断处理。
 - 使用等待中断（WFI），等待事件（WFE）指令，或者从中断返回时直接进入睡眠的 sleep-on-exit 特性可以进入低功耗的休眠模式。
- NVIC:
 - 32 个外部中断，每个中断有4个优先级
 - 专用的不可屏蔽中断（NMI）

- 同时支持电平和脉冲中断触发
- 中断唤醒控制器（WIC），支持低功耗睡眠模式
- 调试
 - 四个硬件断点
 - 两个观察点
 - 用于非侵入式代码分析的程序计数采样寄存器（PCSR）
 - 单步和向量捕获能力
- 总线接口
 - 单一 32位的 AMBA-3 AHB-Lite系统接口，为所有的系统外设和存储器提供方便的集成。
 - 支持 DAP(Debug Access Port)的单一 32位的从机端口。

6.2 系统管理器

6.2.1 概述

系统管理包括如下章节

- 系统复位
- 系统电源架构
- 系统存储器映射
- 用于产品ID，芯片复位及片上模块复位，多功能管脚控制的系统管理寄存器
- 系统定时器 (SysTick)
- 嵌套向量中断控制器(NVIC)
- 系统控制寄存器

6.2.2 系统复位

系统复位可以由如下事件发起，这些复位事件标志可以由寄存器RSTSRC读出。

- 硬件复位
 - 上电复位 (POR)
 - 复位脚 (nRST) 上有低电平
 - 看门狗定时溢出复位(WDT)
 - 低电压复位(LVR)
 - 欠压检测复位(BOD)
- 软件复位
 - MCU 复位 - SYSRESETREQ(AIRCR[2])
 - Cortex-M0 内核一次性复位 - CPU_RST(IPRSTC1[1])
 - 芯片一次性复位- CHIP_RST(IPRSTC1[0])

注: MCU 复位和CPU复位之后，ISPCON.BS 的值保持不变。

6.2.3 系统电源架构

该芯片的电源架构分为2个部分：

- 来自 AV_{DD} 和 AV_{SS} 的模拟电源，为模拟部分提供工作电压。 AV_{DD} 必须等于 V_{DD} 以避免泄漏电流。
- 来自 V_{DD} 和 V_{SS} 的数字电源，为内部稳压器和I/O引脚提供电压，内部稳压器向数字操作提供固定的1.8V电压。

内部稳压器（LDO_CAP）的输出，需要在相应管脚附近接一颗电容。模拟电源（ AV_{DD} ）应该和数字电源（ V_{DD} ）电压相同。下图显示了M058S系列的电源分布：

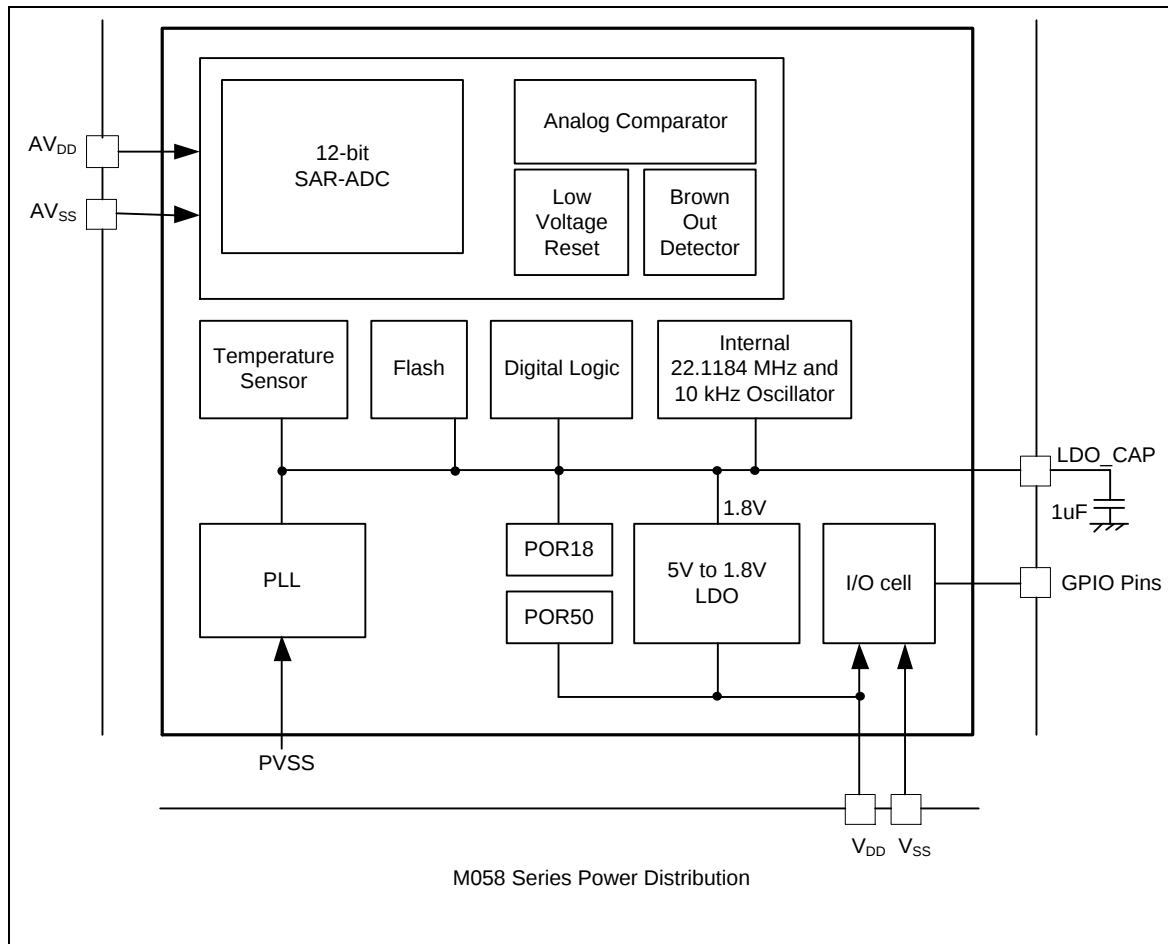


图 6.2-1 NuMicro M058S 系列电源架构图

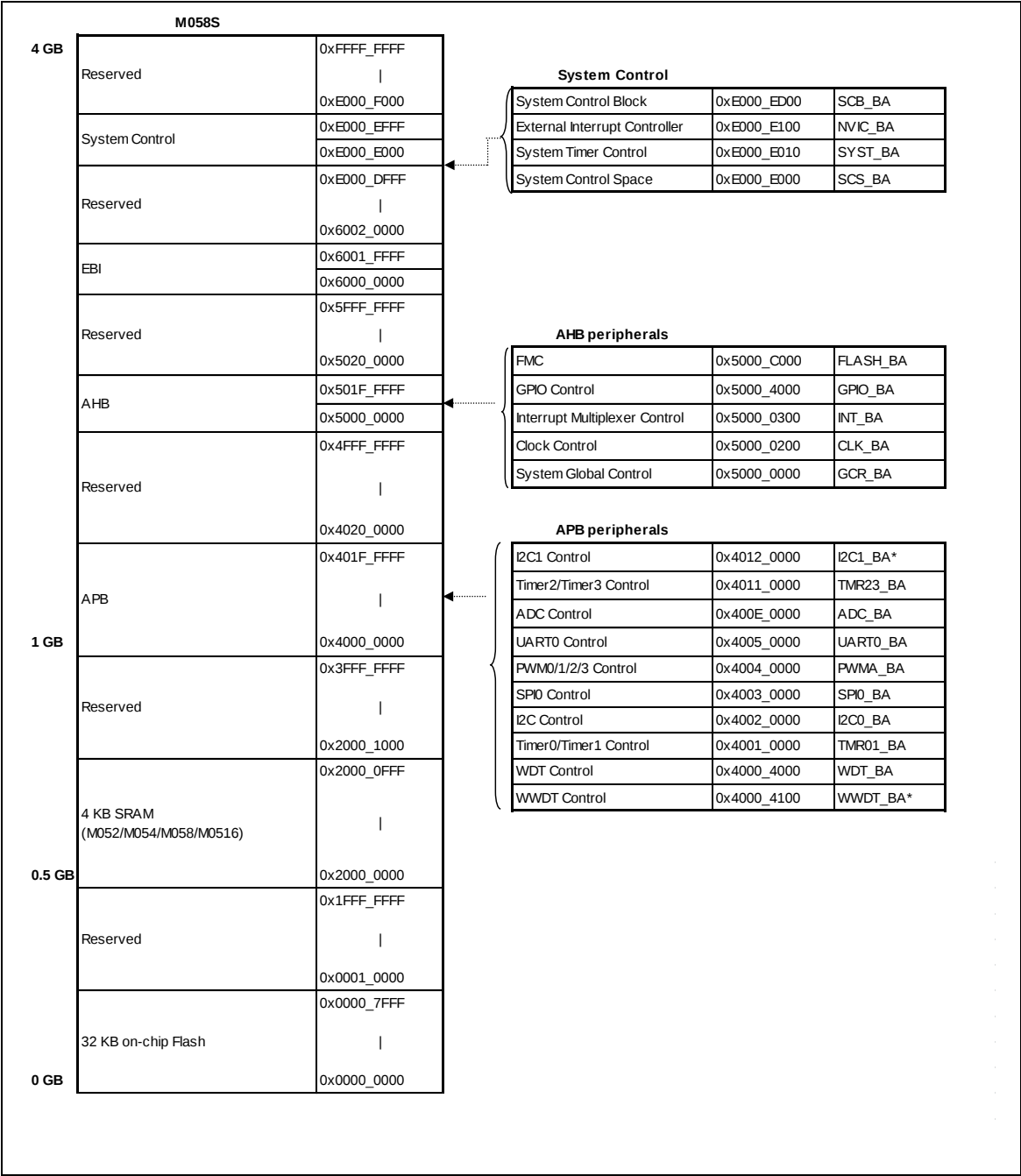
6.2.4 系统存储器映射

NuMicro™ M058S系列提供4G字节的寻址空间。每个片上模块存储器地址分配情况如下表所示。详细的寄存器定义和寻址空间以及编程细节将在后续的各个片上外设描述章节里描述。NuMicro™ M058S 系列仅支持小端数据格式。

地址空间	标志	模块
Flash & SRAM 内存空间		
0x0000_0000 – 0x0000_7FFF	FLASH_BA	FLASH 内存空间 (32 KB)
0x2000_0000 – 0x2000_0FFF	SRAM_BA	SRAM 内存空间 (4 KB)
AHB 模块空间 (0x5000_0000 – 0x501F_FFFF)		
0x5000_0000 – 0x5000_01FF	GCR_BA	系统全局控制寄存器
0x5000_0200 – 0x5000_02FF	CLK_BA	时钟控制寄存器
0x5000_0300 – 0x5000_03FF	INT_BA	多路中断控制寄存器
0x5000_4000 – 0x5000_7FFF	GPIO_BA	GPIO (P0~P7) 控制寄存器
0x5000_C000 – 0x5000_FFFF	FMC_BA	Flash 存储器控制寄存器
APB 模块空间(0x4000_0000 ~ 0x400F_FFFF)		
0x4000_4000 – 0x4000_00FF	WDT_BA	看门狗控制寄存器
0x4000_4100 – 0x4000_7FFF	WWDT_BA	窗口看门狗控制寄存器
0x4001_0000 – 0x4001_3FFF	TMR01_BA	Timer0/Timer1 控制寄存器
0x4002_0000 – 0x4002_3FFF	I2C0_BA	I2C0 接口控制寄存器
0x4003_0000 – 0x4003_3FFF	SPI0_BA	SPI0 带有主/从功能的控制寄存器
0x4004_0000 – 0x4004_3FFF	PWMA_BA	PWM0/1/2/3 控制寄存器
0x4005_0000 – 0x4005_3FFF	UART0_BA	UART0 控制寄存器
0x400E_0000 – 0x400E_FFFF	ADC_BA	数字模拟转换器 (ADC) 控制寄存器
0x4011_0000 – 0x4011_3FFF	TMR23_BA	Timer2/Timer3 控制寄存器
0x4012_0000 – 0x4012_3FFF	I2C1_BA	I2C1接口控制寄存器
系统控制空间 (0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SYST_BA	系统定时器控制寄存器
0xE000_E100 – 0xE000_ECFF	NVIC_BA	外部中断控制器控制寄存器
0xE000_ED00 – 0xE000_ED8F	SCB_BA	系统控制块寄存器

表 6.2-1 片上模块的地址空间分配

6.2.5 系统存储器映射表



系统管理器控制寄存器映射

R: 只读, W: 只写, R/W: 可读写。

寄存器	偏移	R/W	描述	复位值
GCR 基地址: GCR_BA = 0x5000_0000				
PDID	GCR_BA+0x00	R	设备ID 寄存器	0x0000_5810
RSTSRC	GCR_BA+0x04	R/W	系统复位源寄存器	0x0000_00XX
IPRSTC1	GCR_BA+0x08	R/W	外围复位控制寄存器1	0x0000_0000
IPRSTC2	GCR_BA+0x0C	R/W	外围复位控制寄存器2	0x0000_0000
BODCR	GCR_BA+0x18	R/W	欠压检测控制寄存器	0x0000_008X
TEMPCR	GCR_BA+0x1C	R/W	温度传感器控制寄存器	0x0000_0000
PORCR	GCR_BA+0x24	R/W	上电复位控制寄存器	0x0000_00XX
P0_MFP	GCR_BA+0x30	R/W	P0复用功能和输入类型控制寄存器	0x0000_0000
P1_MFP	GCR_BA+0x34	R/W	P1复用功能和输入类型控制寄存器	0x0000_0000
P2_MFP	GCR_BA+0x38	R/W	P2复用功能和输入类型控制寄存器	0x0000_0000
P3_MFP	GCR_BA+0x3C	R/W	P3复用功能和输入类型控制寄存器	0x0000_0000
P4_MFP	GCR_BA+0x40	R/W	P4 输入类型控制寄存器	0x0000_00C0
P5_MFP	GCR_BA+0x44	R/W	P5 输入类型控制寄存器	0x0000_0000
P6_MFP	GCR_BA+0x48	R/W	P6 输入类型控制寄存器	0x0000_0000
P7_MFP	GCR_BA+0x4C	R/W	P7 输入类型控制寄存器	0x0000_0003
REGWRPROT	GCR_BA+0x100	R/W	寄存器写保护控制寄存器	0x0000_0000

设备 ID 寄存器 (PDID)

寄存器	偏移	R/W	描述	复位值
PDID	GCR_BA+0x00	R	设备 ID 寄存器	0x0000_5810 ^[1]

[1] 每个器件具有一个独一无二的默认复位值。

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

位	描述	
[31:0]	PDID	产品设备识别码 该寄存器反映器件的识别码。软件可以读该寄存器识别所使用的器件。 例如, M058SLAN PDID 识别码是 0x0000_5810.

NuMicro™ M058S 系列	产品设备识别码
M058SSAN	0x0000_5816
M058SFAN	0x0000_5818
M058SLAN	0x0000_5810
M058SZAN	0x0000_5813

系统复位源寄存器 (RSTSRC)

该寄存器提供特殊的信息用于软件识别引起芯片上次复位操作的复位源。

寄存器	偏移	R/W	描述	复位值
RSTSRC	GCR_BA+0x04	R/W	系统复位源寄存器	0x0000_00XX

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

位	描述	
[31:8]	Reserved	保留
[7]	RSTS_CPU	CPU 复位标志 如果软件写CPU_RST (IPRSTC1[1])为1，将导致Cortex®-M0 内核和 Flash memory controller (FMC)复位。RSTS_CPU标志将由硬件置位 0 = CPU 无复位 1 = 软件置 CPU_RST为1时，Cortex®-M0 内核和FMC复位。 注: 向该位写 1 清零。
[6]	Reserved	保留
[5]	RSTS_MCU	MCU 复位标志 RSTS_MCU 标志位由来自Cortex®-M0内核的“复位信号”来置位，用于表示之前的复位源。 0 = Cortex®-M0无复位 1 = Cortex®-M0 通过写1到Cortex®-M0 内核的系统控制寄存器 SYSRESETREQ (AIRCRC[2], 应用中断和复位控制寄存器, address = 0xE000ED0C)来发出复位信号复位系统。 注: 向该位写 1 清零。
[4]	RSTS_BOD	欠压检测复位标志 RSTS_BOD标志位由欠压检测器的“复位信号”来置位，用于表示之前的复位源。 0 = BOD 无复位 1 = BOD 已经发出复位信号来复位系统 注: 向该位写 1 清零。

[3]	RSTS_LVR	低电压复位标志 RSTS_LVR 标志位由低电压复位控制器的“复位信号”来置位，用于表示之前的复位源。 0 = LVR无复位 1 = LVR 控制器已经发出复位信号来复位系统 注: 向该位写 1 清零。
[2]	RSTS_WDT	看门狗复位标志 RSTS_WDT 标志位由看门狗定时器的“复位信号”来置位，用于表示之前的复位源。 0 = 看门狗定时器无复位 1 = 看门狗定时器已经发出复位信号来复位系统 注: 向该位写 1 清零。
[1]	RSTS_RESET	复位引脚复位标志 RSTS_RESET 标志位由复位引脚的“复位信号”来置位，用于表示之前的复位源。 0 = nRST 引脚无复位 1 = nRST 引脚已经发出复位信号来复位系统 注: 向该位写1清零。
[0]	RSTS_POR	上电复位标志 RSTS_POR 标志位由上电复位(POR)控制器或CHIP_RST位(IPRSTC1[0])的“复位信号”来置位，用于表示之前的复位源。 0 = POR 或 CHIP_RST无复位 1 = 上电复位(POR) 或 CHIP_RST已经发出复位信号来复位系统 注: 向该位写1清零。

外设复位控制寄存器1 (IPRSTC1)

寄存器	偏移	R/W	描述	复位值
IPRSTC1	GCR_BA+0x08	R/W	外设复位控制寄存器1	0x0000_0000

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

位	描述	
[31:2]	Reserved	保留
[1]	CPU_RST	Cortex-M0 内核一次性复位 (写保护) 设置该位仅复位Cortex-M0内核及Flash存储控制器(FMC)，该位将在2个时钟周期后自动为零。 0 = Cortex-M0 内核正常工作 1 = Cortex-M0 内核一次性复位 注: 该位是写保护的, 程序需要依次向地址0x5000_0100写入下列数据 “59h”, “16h”, 和 “88h” 来关闭保护功能。参考寄存器REGWRPROT, 地址 GCR_BA+0x100。
[0]	CHIP_RST	芯片一次性复位 (写保护) 设置该位将会复位整个芯片, 包括Cortex-M0内核及所有外设, 该位将在2个时钟周期后自动为零。 CHIP_RST 与 POR 复位一样。所有芯片控制器都会复位并且芯片设置将从CONFIG0里重新加载。 0 = Chip 正常工作 1 = Chip 一次性复位 注: 该位是写保护的, 程序需要依次向地址0x5000_0100写入下列数据 “59h”, “16h”, 和 “88h” 来关闭保护功能。参考寄存器REGWRPROT, 地址 GCR_BA+0x100。

外设复位控制寄存器2 (IPRSTC2)

设置这些位为“1”将会产生一个异步的复位信号给相应的模块。用户需要设置这些位为“0”来将模块从复位状态恢复。

寄存器	偏移	R/W	描述	复位值
IPRSTC2	GCR_BA+0x0C	R/W	外设复位控制寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved			ADC_RST	Reserved			
23	22	21	20	19	18	17	16
Reserved			PWM03_RST	Reserved			UART0_RST
15	14	13	12	11	10	9	8
Reserved			SPI0_RST	Reserved		I2C1_RST	I2C0_RST
7	6	5	4	3	2	1	0
Reserved		TMR3_RST	TMR2_RST	TMR1_RST	TMR0_RST	GPIO_RST	Reserved

位	描述	
[31:29]	Reserved	保留。
[28]	ADC_RST	ADC 控制器复位 0 = ADC 控制器正常工作 1 = ADC 控制器复位
[27:21]	Reserved	保留
[20]	PWM03_RST	PWM03 控制器复位 0 = PWM03 控制器正常工作 1 = PWM03 控制器复位。
[19:17]	Reserved	保留
[16]	UART0_RST	UART0 控制器复位 0 = UART0 控制器正常工作 1 = UART0 控制器复位。
[15:13]	Reserved	保留
[12]	SPI0_RST	SPI0 控制器复位 0 = SPI0 控制器正常工作 1 = SPI0 控制器复位。
[11:10]	Reserved	保留

[9]	I2C1_RST	I²C1 控制器复位 0= I ² C1 控制器正常工作 1= I ² C1 控制器复位.
[8]	I2C0_RST	I²C0 控制器复位 0= I ² C0 控制器正常工作 1= I ² C0 控制器复位.
[7:6]	Reserved	保留
[5]	TMR3_RST	Timer3 控制器复位 0 = Timer3 控制器正常工作 1 = Timer3 控制器复位.
[4]	TMR2_RST	Timer2 控制器复位 0 = Timer2 控制器正常工作 1 = Timer2 控制器复位.
[3]	TMR1_RST	Timer1 控制器复位 0 = Timer1 控制器正常工作 1 = Timer1 控制器复位.
[2]	TMR0_RST	Timer0 控制器复位 0 = Timer0 控制器正常工作 1 = Timer0 控制器复位.
[1]	GPIO_RST	GPIO (P0~P4) 控制器复位 0 = GPIO 控制器正常工作 1 = GPIO 控制器复位.
[0]	Reserved	保留

欠压检测控制寄存器(BODCR)

BODCR 控制寄存器的部分值由flash配置区域（ CONFIG0/CONFIG1 ） 初始化并且是写保护的，编程这些被保护的位需要依次向地址0x5000_0100写入“ 59h”，“ 16h”，“ 88h”，禁用寄存器保护。参考寄存器REGWRPROT，其地址为GCR_BA+0x100。

寄存器	偏移	R/W	描述	复位值
BODCR	GCR_BA+0x18	R/W	欠压检测控制寄存器	0x0000_008X

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

位	描述	
[31:8]	Reserved	保留
[7]	LVR_EN	低压复位使能（写保护位） 输入电源电压低于LVR电路设置时， LVR复位整个芯片。默认配置下LVR复位功能是使能的。 0 =禁用低电压复位功能 1 =使能低电压复位功能,使能该位后,LVR功能将在100us后生效,以使LVR输出稳定(默认) 注: 该位是写保护的, 程序需要依次向地址0x5000_0100写入下列数据 “59h”, “16h”, 和 “88h” 来关闭保护功能。参考寄存器REGWRPROT， 地址 GCR_BA+0x100。
[6]	BOD_OUT	欠压检测器输出状态位 0 =欠压检测器输出状态为0。表示检测到电压高于BOD_VL的设置或BOD_EN为0。 1 =欠压检测器输出状态为 1。表示检测到电压低于 BOD_VL 的设置。如果 BOD_EN 是 0, BOD 功能禁用, 该位通常响应为 0。
[5]	BOD_LPM	欠压检测器低功耗模式（写保护位） 0 = BOD 工作在正常模式（默认） 1 = BOD 工作在低功耗模式 注1: BOD 在正常模式下消耗电流约为100 uA，低功耗模式下能减小到当前的约1/10, 但BOD响应速度变慢。

		注2: 该位是写保护的, 程序需要依次向地址0x5000_0100写入下列数据 “59h”, “16h”, 和 “88h” 来关闭保护功能。参考寄存器REGWRPROT, 地址 GCR_BA+0x100。		
[4]	BOD_INTF	欠压检测中断标志 0 = 欠压检测器没有检测到任何电压由VDD 下降或上升至BOD_VL 的设定值。 1 = 当欠压检测到VDD下降或者上升到BOD_VL设定电压, 该位置为1, 如果欠压电压中断被使能, 则发生欠压中断。 注: 向该位写 1 清零。		
[3]	BOD_RSTEN	欠压复位使能 (写保护位) 0 = 使能欠压“中断”功能 当使能BOD功能 (BOD_EN为高) 和BOD中断功能 (BOD_RSTEN为低) 时, 如果 BOD_OUT为高, 则BOD将产生一个中断。BOD中断将会一直保持直到BOD_EN设置为0。通过禁用NVIC BOD中断或者禁用BOD功能 (BOD_EN 为低) 可以将BOD中断停用。 1 = 使能欠压“复位”功能 注1: 当同时使能欠压检测功能 (BOD_EN 为高) 和 BOD 复位功能 (BOD_RSTEN 为高), 如果检测到电压低于阈电压(BOD_OUT 为高), BOD将发送信号来复位芯片。 注2: 默认值由用户配置flash 控制寄存器CONFIG0[20] 时设置。 注3: 该位是写保护的, 程序需要依次向地址0x5000_0100写入下列数据 “59h”, “16h”, 和 “88h” 来关闭保护功能。参考寄存器REGWRPROT, 地址 GCR_BA+0x100。		
[2:1]	BOD_VL	欠压检测阈电压选择 (写保护位) 默认值 由用户配置flash 控制寄存器CONFIG0 bit[22:21]决定。		
		BOV_VL[1]	BOV_VL[0]	欠压值
		1	1	4.4V
		1	0	3.7V
		0	1	2.7V
		0	0	2.2V
		注: 该位是写保护的, 程序需要依次向地址0x5000_0100写入下列数据 “59h”, “16h”, 和 “88h” 来关闭保护功能。参考寄存器REGWRPROT, 地址 GCR_BA+0x100。		
[0]	BOD_EN	欠压检测使能 (写保护位) 默认值由FLASH中用户配置寄存器CONFIG0 bit[23]决定 0 = 禁用欠压检测功能 1 = 使能欠压检测功能 注: 该位是写保护的, 程序需要依次向地址0x5000_0100写入下列数据 “59h”, “16h”, 和 “88h” 来关闭保护功能。参考寄存器REGWRPROT, 地址 GCR_BA+0x100。		

温度传感器控制寄存器 (TEMPCR)

寄存器	偏移	R/W	描述	复位值
TEMPCR	GCR_BA+0x1C	R/W	温度传感器控制寄存器	0x0000_0000

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

位	描述	
[31:1]	Reserved	保留
[0]	VTEMP_EN	使能温度传感器 该位用来使能/禁用温度传感器功能。 0 = 禁用温度传感器功能（默认） 1 = 始能温度传感器功能 注:在该位设置为 1 后，温度传感器的输出值可以由 ADC 的转换结果获得。ADC 转换功能的细节请参考 ADC 章节。

上电复位控制寄存器 (PORCR)

寄存器	偏移	R/W	描述	复位值
PORCR	GCR_BA+0x24	R/W	上电复位控制寄存器	0x0000_00XX

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

位	描述	
[31:16]	Reserved	保留
[15:0]	POR_DIS_CODE	上电复位使能位（写保护） 上电时，POR电路产生复位信号使整个芯片复位，但是电源部分的干扰可能引起POR再次有效。用户可以通过给该区域写0x5AA5，来禁用POR内部电路，避免不可预知的干扰引起芯片复位。 当该区域设置为其它值或者芯片由其它复位源复位时，POR功能将会再次有效，其它复位源包括： nRST引脚复位、看门狗复位、LVR复位、BOD复位、ICE复位命令和软件复位。 注：该位是写保护的，程序需要依次向地址 0x5000_0100 写入下列数据 “59h”，“16h”，和 “88h” 来关闭保护功能。参考寄存器 REGWRPROT，地址 GCR_BA+0x100。

Port0 多功能控制寄存器 (P0_MFP)

寄存器	偏移	R/W	描述	复位值
P0_MFP	GCR_BA+0x30	R/W	P0 多功能复用与输入类型控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
P0_TYPE[7:0]							
15	14	13	12	11	10	9	8
P0_ALT[7:0]							
7	6	5	4	3	2	1	0
P0_MFP[7:0]							

位	描述		
[31:24]	Reserved	保留	
[23:16]	P0_TYPEn	使能P0[7:0] 施密特输入触发功能 0 = 禁用 P0[7:0] I/O施密特输入触发功能 1 = 使能 P0[7:0] I/O施密特输入触发功能	
[15]	P0_ALT[7]	P0.7 复用功能选择 P0.7 的功能取决于P0_MFP[7] 和 P0_ALT[7]	
		P0_ALT[7]	P0_MFP[7] P0.7 功能
		0	0 P0.7
		0	1 保留
		1	0 SPICLK0(SPI0)
		1	1 保留
[14]	P0_ALT[6]	P0.6 复用功能选择 P0.6 的功能取决于 P0_MFP[6] 和 P0_ALT[6].	
		P0_ALT[6]	P0_MFP[6] P0.6 功能
		0	0 P0.6

		<table> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>MISO0(SPI0)</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	0	1	保留	1	0	MISO0(SPI0)	1	1	保留						
0	1	保留															
1	0	MISO0(SPI0)															
1	1	保留															
[13]	P0_ALT[5]	P0.5 复用功能选择 P0.5 的功能取决于 P0_MFP[5] 和 P0_ALT[5]. <table> <tr> <th>P0_ALT[5]</th><th>P0_MFP[5]</th><th>P0.5 功能</th></tr> <tr> <td>0</td><td>0</td><td>P0.5</td></tr> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>MOSI0(SPI0)</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P0_ALT[5]	P0_MFP[5]	P0.5 功能	0	0	P0.5	0	1	保留	1	0	MOSI0(SPI0)	1	1	保留
P0_ALT[5]	P0_MFP[5]	P0.5 功能															
0	0	P0.5															
0	1	保留															
1	0	MOSI0(SPI0)															
1	1	保留															
[12]	P0_ALT[4]	P0.4 复用功能选择 P0.4的功能取决于P0_MFP[4] 和 P0_ALT[4]. <table> <tr> <th>P0_ALT[4]</th><th>P0_MFP[4]</th><th>P0.4 功能</th></tr> <tr> <td>0</td><td>0</td><td>P0.4</td></tr> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>SPISS0(SPI0)</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P0_ALT[4]	P0_MFP[4]	P0.4 功能	0	0	P0.4	0	1	保留	1	0	SPISS0(SPI0)	1	1	保留
P0_ALT[4]	P0_MFP[4]	P0.4 功能															
0	0	P0.4															
0	1	保留															
1	0	SPISS0(SPI0)															
1	1	保留															
[11]	P0_ALT[3]	P0.3 复用功能选择 P0.3 的功能取决于 P0_MFP[3] 和 P0_ALT[3]. <table> <tr> <th>P0_ALT[3]</th><th>P0_MFP[3]</th><th>P0.3 功能</th></tr> <tr> <td>0</td><td>0</td><td>P0.3</td></tr> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>RTS0(UART0)</td></tr> <tr> <td>1</td><td>1</td><td>RXD0(UART0)</td></tr> </table>	P0_ALT[3]	P0_MFP[3]	P0.3 功能	0	0	P0.3	0	1	保留	1	0	RTS0(UART0)	1	1	RXD0(UART0)
P0_ALT[3]	P0_MFP[3]	P0.3 功能															
0	0	P0.3															
0	1	保留															
1	0	RTS0(UART0)															
1	1	RXD0(UART0)															
[10]	P0_ALT[2]	P0.2 复用功能选择 P0.2 的功能取决于 P0_MFP[2] 和 P0_ALT[2]. <table> <tr> <th>P0_ALT[2]</th><th>P0_MFP[2]</th><th>P0.2 功能</th></tr> <tr> <td>0</td><td>0</td><td>P0.2</td></tr> <tr> <td>0</td><td>1</td><td>保留</td></tr> </table>	P0_ALT[2]	P0_MFP[2]	P0.2 功能	0	0	P0.2	0	1	保留						
P0_ALT[2]	P0_MFP[2]	P0.2 功能															
0	0	P0.2															
0	1	保留															

		<table> <tr> <td>1</td><td>0</td><td>CTS0(UART0)</td></tr> <tr> <td>1</td><td>1</td><td>TXD0(UART0)</td></tr> </table>	1	0	CTS0(UART0)	1	1	TXD0(UART0)									
1	0	CTS0(UART0)															
1	1	TXD0(UART0)															
[9]	P0_ALT[1]	P0.1 复用功能选择 P0.1 的功能取决于 P0_MFP[1], P0_ALT[1] 和 P0_ALT1[1]. <table> <tr> <th>P0_ALT[1]</th><th>P0_MFP[1]</th><th>P0.1功能</th></tr> <tr> <td>0</td><td>0</td><td>P0.1</td></tr> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>保留</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P0_ALT[1]	P0_MFP[1]	P0.1功能	0	0	P0.1	0	1	保留	1	0	保留	1	1	保留
P0_ALT[1]	P0_MFP[1]	P0.1功能															
0	0	P0.1															
0	1	保留															
1	0	保留															
1	1	保留															
[8]	P0_ALT[0]	P0.0 复用功能选择 P0.0 的功能取决于 P0_MFP[0], P0_ALT[0] 和 P0_ALT1[0]. <table> <tr> <th>P0_ALT[0]</th><th>P0_MFP[0]</th><th>P0.0功能</th></tr> <tr> <td>0</td><td>0</td><td>P0.0</td></tr> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>保留</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P0_ALT[0]	P0_MFP[0]	P0.0功能	0	0	P0.0	0	1	保留	1	0	保留	1	1	保留
P0_ALT[0]	P0_MFP[0]	P0.0功能															
0	0	P0.0															
0	1	保留															
1	0	保留															
1	1	保留															
[7:0]	P0_MFP[7:0]	P0 复用功能选择 P0 的功能取决于 P0_MFP 和 P0_ALT. 参考 P0_ALT 来获取详细描述。															

Port1 多功能控制寄存器 (P1_MFP)

寄存器	偏移	R/W	描述	复位值
P1_MFP	GCR_BA+0x34	R/W	P1 多功能复用与输入类型控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
P1_TYPE[7:0]							
15	14	13	12	11	10	9	8
P1_ALT[7:0]							
7	6	5	4	3	2	1	0
P1_MFP[7:0]							

位	描述		
[31:24]	Reserved	保留	
[23:16]	P1_TYPEn	使能 P1[7:0] 施密特输入触发功能 0 = 禁用 P1[7:0] I/O 施密特输入触发功能 1 = 使能 P1[7:0] I/O 施密特输入触发功能	
[15]	P1_ALT[7]	P1.7 复用功能选择 P1.7 的功能取决于 P1_MFP[7] 和 P1_ALT[7].	
		P1_ALT[7]	P1.7 功能
		0	P1.7
		0	AIN7(ADC)
		1	SPICLK0(SPI0)
[14]	P1_ALT[6]	1	保留
		P1.6 复用功能选择 P1.6 的功能取决于 P1_MFP[6] 和 P1_ALT[6].	
		P1_ALT[6]	P1.6 功能
		0	P1.6

		<table> <tr> <td>0</td><td>1</td><td>AIN6(ADC)</td></tr> <tr> <td>1</td><td>0</td><td>MISO0(SPI0)</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	0	1	AIN6(ADC)	1	0	MISO0(SPI0)	1	1	保留						
0	1	AIN6(ADC)															
1	0	MISO0(SPI0)															
1	1	保留															
[13]	P1_ALT[5]	P1.5 复用功能选择 P1.5 的功能取决于 P1_MFP[5] 和 P1_ALT[5]. <table> <tr> <th>P1_ALT[5]</th><th>P1_MFP[5]</th><th>P1.5 功能</th></tr> <tr> <td>0</td><td>0</td><td>P1.5</td></tr> <tr> <td>0</td><td>1</td><td>AIN5(ADC)</td></tr> <tr> <td>1</td><td>0</td><td>MOSI0(SPI0)</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P1_ALT[5]	P1_MFP[5]	P1.5 功能	0	0	P1.5	0	1	AIN5(ADC)	1	0	MOSI0(SPI0)	1	1	保留
P1_ALT[5]	P1_MFP[5]	P1.5 功能															
0	0	P1.5															
0	1	AIN5(ADC)															
1	0	MOSI0(SPI0)															
1	1	保留															
[12]	P1_ALT[4]	P1.4 复用功能选择 P1.4 的功能取决于 P1_MFP[4] 和 P1_ALT[4]. <table> <tr> <th>P1_ALT[4]</th><th>P1_MFP[4]</th><th>P1.4 功能</th></tr> <tr> <td>0</td><td>0</td><td>P1.4</td></tr> <tr> <td>0</td><td>1</td><td>AIN4(ADC)</td></tr> <tr> <td>1</td><td>0</td><td>SPISS0(SPI0)</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P1_ALT[4]	P1_MFP[4]	P1.4 功能	0	0	P1.4	0	1	AIN4(ADC)	1	0	SPISS0(SPI0)	1	1	保留
P1_ALT[4]	P1_MFP[4]	P1.4 功能															
0	0	P1.4															
0	1	AIN4(ADC)															
1	0	SPISS0(SPI0)															
1	1	保留															
[11]	P1_ALT[3]	P1.3 复用功能选择 P1.3 的功能取决于 P1_MFP[3] 和 P1_ALT[3]. <table> <tr> <th>P1_ALT[3]</th><th>P1_MFP[3]</th><th>P1.3 功能</th></tr> <tr> <td>0</td><td>0</td><td>P1.3</td></tr> <tr> <td>0</td><td>1</td><td>AIN3(ADC)</td></tr> <tr> <td>1</td><td>0</td><td>保留</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P1_ALT[3]	P1_MFP[3]	P1.3 功能	0	0	P1.3	0	1	AIN3(ADC)	1	0	保留	1	1	保留
P1_ALT[3]	P1_MFP[3]	P1.3 功能															
0	0	P1.3															
0	1	AIN3(ADC)															
1	0	保留															
1	1	保留															
[10]	P1_ALT[2]	P1.2 复用功能选择 P1.2 的功能取决于 P1_MFP[2] 和 P1_ALT[2]. <table> <tr> <th>P1_ALT[2]</th><th>P1_MFP[2]</th><th>P1.2 功能</th></tr> <tr> <td>0</td><td>0</td><td>P1.2</td></tr> <tr> <td>0</td><td>1</td><td>AIN2(ADC)</td></tr> </table>	P1_ALT[2]	P1_MFP[2]	P1.2 功能	0	0	P1.2	0	1	AIN2(ADC)						
P1_ALT[2]	P1_MFP[2]	P1.2 功能															
0	0	P1.2															
0	1	AIN2(ADC)															

		1	0	保留
		1	1	保留
[9]	P1_ALT[1]	P1.1 复用功能选择 P1.1 的功能取决于 P1_MFP[1] 和 P1_ALT[1].		
		P1_ALT[1]	P1_MFP[1]	P1.1 功能
		0	0	P1.1
		0	1	AIN1(ADC)
		1	0	T3(Timer3)
		1	1	保留
[8]	P1_ALT[0]	P1.0 复用功能选择 P1.0 的功能取决于 P1_MFP[0] 和 P1_ALT[0].		
		P1_ALT[0]	P1_MFP[0]	P1.0 功能
		0	0	P1.0
		0	1	AIN0(ADC)
		1	0	T2(Timer2)
		1	1	保留
[7:0]	P1_MFP[7:0]	P1 复用功能选择 P1 的功能取决于 P1_MFP 和 P1_ALT. 参考 P1_ALT 来获取详细描述。		

Port2 多功能控制寄存器 (P2_MFP)

寄存器	偏移	R/W	描述	复位值
P2_MFP	GCR_BA+0x38	R/W	P2 多功能复用与输入类型控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
P2_TYPE[7:0]							
15	14	13	12	11	10	9	8
P2_ALT[7:0]							
7	6	5	4	3	2	1	0
P2_MFP[7:0]							

位	描述																
[31:24]	Reserved	保留															
[23:16]	P2_TYPEn	使能 P2[7:0] 施密特输入触发功能 0 = 禁用 P2[7:0] I/O 施密特输入触发功能 1 = 使能 P2[7:0] I/O 施密特输入触发功能															
[15]	P2_ALT[7]	P2.7 复用功能选择 P2.7 的功能取决于 P2_MFP[7] 和 P2_ALT[7]。 <table> <tr> <th>P2_ALT[7]</th><th>P2_MFP[7]</th><th>P2.7 功能</th></tr> <tr> <td>0</td><td>0</td><td>P2.7</td></tr> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>保留</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P2_ALT[7]	P2_MFP[7]	P2.7 功能	0	0	P2.7	0	1	保留	1	0	保留	1	1	保留
P2_ALT[7]	P2_MFP[7]	P2.7 功能															
0	0	P2.7															
0	1	保留															
1	0	保留															
1	1	保留															
[14]	P2_ALT[6]	P2.6 复用功能选择 P2.6 的功能取决于 P2_MFP[6] 和 P2_ALT[6]。 <table> <tr> <th>P2_ALT[6]</th><th>P2_MFP[6]</th><th>P2.6 功能</th></tr> <tr> <td>0</td><td>0</td><td>P2.6</td></tr> </table>	P2_ALT[6]	P2_MFP[6]	P2.6 功能	0	0	P2.6									
P2_ALT[6]	P2_MFP[6]	P2.6 功能															
0	0	P2.6															

		<table> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>保留</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	0	1	保留	1	0	保留	1	1	保留						
0	1	保留															
1	0	保留															
1	1	保留															
[13]	P2_ ALT[5]	P2.5 复用功能选择 P2.5 的功能取决于 P2_MFP[5] 和 P2_ALT[5]. <table> <tr> <th>P2_ ALT[5]</th><th>P2_MFP[5]</th><th>P2.5 功能</th></tr> <tr> <td>0</td><td>0</td><td>P2.5</td></tr> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>保留</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P2_ ALT[5]	P2_MFP[5]	P2.5 功能	0	0	P2.5	0	1	保留	1	0	保留	1	1	保留
P2_ ALT[5]	P2_MFP[5]	P2.5 功能															
0	0	P2.5															
0	1	保留															
1	0	保留															
1	1	保留															
[12]	P2_ ALT[4]	P2.4 复用功能选择 P2.4 的功能取决于 P2_MFP[4] 和 P2_ALT[4]. <table> <tr> <th>P2_ ALT[4]</th><th>P2_MFP[4]</th><th>P2.4 功能</th></tr> <tr> <td>0</td><td>0</td><td>P2.4</td></tr> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>保留</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P2_ ALT[4]	P2_MFP[4]	P2.4 功能	0	0	P2.4	0	1	保留	1	0	保留	1	1	保留
P2_ ALT[4]	P2_MFP[4]	P2.4 功能															
0	0	P2.4															
0	1	保留															
1	0	保留															
1	1	保留															
[11]	P2_ ALT[3]	P2.3 复用功能选择 P2.3 的功能取决于 P2_MFP[3] 和 P2_ALT[3]. <table> <tr> <th>P2_ ALT[3]</th><th>P2_MFP[3]</th><th>P2.3 功能</th></tr> <tr> <td>0</td><td>0</td><td>P2.3</td></tr> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>PWM3(PWMA 通道 3)</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P2_ ALT[3]	P2_MFP[3]	P2.3 功能	0	0	P2.3	0	1	保留	1	0	PWM3(PWMA 通道 3)	1	1	保留
P2_ ALT[3]	P2_MFP[3]	P2.3 功能															
0	0	P2.3															
0	1	保留															
1	0	PWM3(PWMA 通道 3)															
1	1	保留															
[10]	P2_ ALT[2]	P2.2 复用功能选择 P2.2 的功能取决于 P2_MFP[2] 和 P2_ALT[2]. <table> <tr> <th>P2_ ALT[2]</th><th>P2_MFP[2]</th><th>P2.2 功能</th></tr> <tr> <td>0</td><td>0</td><td>P2.2</td></tr> </table>	P2_ ALT[2]	P2_MFP[2]	P2.2 功能	0	0	P2.2									
P2_ ALT[2]	P2_MFP[2]	P2.2 功能															
0	0	P2.2															

		<table> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>PWM2(PWMA 通道 2)</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	0	1	保留	1	0	PWM2(PWMA 通道 2)	1	1	保留						
0	1	保留															
1	0	PWM2(PWMA 通道 2)															
1	1	保留															
[9]	P2_ALT[1]	P2.1 复用功能选择 P2.1 的功能取决于 P2_MFP[1] 和 P2_ALT[1]. <table> <tr> <th>P2_ALT[1]</th><th>P2_MFP[1]</th><th>P2.1 功能</th></tr> <tr> <td>0</td><td>0</td><td>P2.1</td></tr> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>PWM1(PWMA 通道 1)</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P2_ALT[1]	P2_MFP[1]	P2.1 功能	0	0	P2.1	0	1	保留	1	0	PWM1(PWMA 通道 1)	1	1	保留
P2_ALT[1]	P2_MFP[1]	P2.1 功能															
0	0	P2.1															
0	1	保留															
1	0	PWM1(PWMA 通道 1)															
1	1	保留															
[8]	P2_ALT[0]	P2.0 复用功能选择 P2.0 的功能取决于 P2_MFP[0] 和 P2_ALT[0]. <table> <tr> <th>P2_ALT[0]</th><th>P2_MFP[0]</th><th>P2.0 功能</th></tr> <tr> <td>0</td><td>0</td><td>P2.0</td></tr> <tr> <td>0</td><td>1</td><td>保留</td></tr> <tr> <td>1</td><td>0</td><td>PWM0(PWMA 通道 0)</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P2_ALT[0]	P2_MFP[0]	P2.0 功能	0	0	P2.0	0	1	保留	1	0	PWM0(PWMA 通道 0)	1	1	保留
P2_ALT[0]	P2_MFP[0]	P2.0 功能															
0	0	P2.0															
0	1	保留															
1	0	PWM0(PWMA 通道 0)															
1	1	保留															
[7:0]	P2_MFP[7:0]	P2 复用功能选择 P2 的功能取决于 P2_MFP 和 P2_ALT. 参照 P2_ALT 来获取详细描述。															

Port3 多功能控制寄存器 (P3_MFP)

寄存器	偏移	R/W	描述	复位值
P3_MFP	GCR_BA+0x3C	R/W	P3 多功能复用与输入类型控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
P3_TYPE[7:0]							
15	14	13	12	11	10	9	8
P3_ALT[7:0]							
7	6	5	4	3	2	1	0
P3_MFP[7:0]							

位	描述			
[31:24]	Reserved	保留		
[23:16]	P3_TYPEn	使能 P3[7:0] 施密特输入触发功能 0 = 禁用 P3[7:0] I/O 施密特输入触发功能 1 = 使能 P3[7:0] I/O 施密特输入触发功能		
[15]	P3_ALT[7]	P3.7 复用功能选择 P3.7 的功能取决于 P3_MFP[7] 和 P3_ALT[7].		
		P3_ALT[7]	P3_MFP[7]	P3.7 功能
		0	0	P3.7
		0	1	保留
		1	0	保留
[14]	P3_ALT[6]	P3.6 复用功能选择 P3.6 的功能取决于 P3_MFP[6] 和 P3_ALT[6].		
		P3_ALT[6]	P3_MFP[6]	P3.6 功能
		0	0	P3.6
		0	1	保留

		1	0	CKO(时钟输出)	
		1	1	保留	
[13]	P3_ALT[5]	P3.5 复用功能选择 P3.5 的功能取决于 P3_MFP[5] 和 P3_ALT[5].			
		P3_ALT[5]	P3_MFP[5]	P3.5 功能	
		0	0	P3.5	
		0	1	T1(Timer1)	
		1	0	SCL0(I ² C0)	
		1	1	CKO(时钟输出)	
[12]	P3_ALT[4]	P3.4 复用功能选择 P3.4 的功能取决于 P3_MFP[4] 和 P3_ALT[4].			
		P3_ALT[4]	P3_MFP[4]	P3.4 功能	
		0	0	P3.4	
		0	1	T0(Timer0)	
		1	0	SDA0(I ² C0)	
		1	1	保留	
[11]	P3_ALT[3]	P3.3 复用功能选择 P3.3 的功能取决于 P3_MFP[3] 和 P3_ALT[3].			
		P3_ALT[3]	P3_MFP[3]	P3.3 功能	
		0	0	P3.3	
		0	1	/INT1	
		1	0	保留	
		1	1	T1EX	
[10]	P3_ALT[2]	P3.2 复用功能选择 P3.2 的功能取决于 P3_MFP[2] 和 P3_ALT[2].			
		P3_ALT[2]	P3_MFP[2]	P3.2 时钟	
		0	0	P3.2	
		0	1	/INT0	
		1	0	T0EX	
		1	1	保留	

[9]	P3_ALT[1]	P3.1 复用功能选择 P3.1 的功能取决于 P3_MFP[1] 和 P3_ALT[1].		
		P3_ALT[1]	P3_MFP[1]	P3.1 功能
		0	0	P3.1
		0	1	TXD(UART0)
		1	0	保留
		1	1	保留
[8]	P3_ALT[0]	P3.0 复用功能选择 P3.0 的功能取决于 P3_MFP[0] 和 P3_ALT[0].		
		P3_ALT[0]	P3_MFP[0]	P3.0 功能
		0	0	P3.0
		0	1	RXD(UART0)
		1	0	保留
		1	1	保留
[7:0]	P3_MFP[7:0]	P3 复用功能选择 P3 的功能取决于 P3_MFP 和 P3_ALT。 参照 P3_ALT 来获取详细描述。		

Port4 多功能控制寄存器 (P4_MFP)

寄存器	偏移	R/W	描述	复位值
P4_MFP	GCR_BA+0x40	R/W	P4 多功能复用与输入类型控制寄存器	0x0000_00C0

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
P4_TYPE[7:0]							
15	14	13	12	11	10	9	8
P4_ALT[7:0]							
7	6	5	4	3	2	1	0
P4_MFP[7:0]							

位	描述		
[31:24]	Reserved	保留	
[23:16]	P4_TYPEn	使能 P4[7:0] 施密特输入触发功能 0= 禁用 P4[7:0] I/O 施密特输入触发功能 1= 使能 P4[7:0] I/O 施密特输入触发功能	
[15]	P4_ALT[7]	P4.7 复用功能选择 P4.7 的功能取决于 P4_MFP[7] 和 P4_ALT[7].	
		P4_ALT[7]	P4_MFP[7]
		0	0
		0	1
		1	x
[14]	P4_ALT[6]	P4.6 复用功能选择 P4.6 的功能取决于 P4_MFP[6] 和 P4_ALT[6].	
		P4_ALT[6]	P4_MFP[6]
		0	0
		0	1
		1	x

		1	x	保留															
[13]	P4_ ALT[5]	P4.5 复用功能选择 P4.5 的功能取决于 P4_MFP[5] 和 P4_ALT[5]. <table><tr><th>P4_ ALT[5]</th><th>P4_MFP[5]</th><th>P4.5 功能</th></tr><tr><td>0</td><td>0</td><td>P4.5</td></tr><tr><td>0</td><td>1</td><td>保留</td></tr><tr><td>1</td><td>0</td><td>SDA1(I²C1)</td></tr><tr><td>1</td><td>1</td><td>保留</td></tr></table>			P4_ ALT[5]	P4_MFP[5]	P4.5 功能	0	0	P4.5	0	1	保留	1	0	SDA1(I ² C1)	1	1	保留
P4_ ALT[5]	P4_MFP[5]	P4.5 功能																	
0	0	P4.5																	
0	1	保留																	
1	0	SDA1(I ² C1)																	
1	1	保留																	
[12]	P4_ ALT[4]	P4.4 复用功能选择 P4.4 的功能取决于 P4_MFP[4] 和 P4_ALT[4]. <table><tr><th>P4_ ALT[4]</th><th>P4_MFP[4]</th><th>P4.4 功能</th></tr><tr><td>0</td><td>0</td><td>P4.4</td></tr><tr><td>0</td><td>1</td><td>保留</td></tr><tr><td>1</td><td>0</td><td>SCL1(I²C1)</td></tr><tr><td>1</td><td>1</td><td>保留</td></tr></table>			P4_ ALT[4]	P4_MFP[4]	P4.4 功能	0	0	P4.4	0	1	保留	1	0	SCL1(I ² C1)	1	1	保留
P4_ ALT[4]	P4_MFP[4]	P4.4 功能																	
0	0	P4.4																	
0	1	保留																	
1	0	SCL1(I ² C1)																	
1	1	保留																	
[11]	P4_ ALT[3]	P4.3 复用功能选择 P4.3 的功能取决于 P4_MFP[3] 和 P4_ALT[3]. <table><tr><th>P4_ ALT[3]</th><th>P4_MFP[3]</th><th>P4.3 功能</th></tr><tr><td>0</td><td>0</td><td>P4.3</td></tr><tr><td>0</td><td>1</td><td>PWM3(PWM 发生器 2)</td></tr><tr><td>1</td><td>x</td><td>保留</td></tr></table>			P4_ ALT[3]	P4_MFP[3]	P4.3 功能	0	0	P4.3	0	1	PWM3(PWM 发生器 2)	1	x	保留			
P4_ ALT[3]	P4_MFP[3]	P4.3 功能																	
0	0	P4.3																	
0	1	PWM3(PWM 发生器 2)																	
1	x	保留																	
[10]	P4_ ALT[2]	P4.2 复用功能选择 P4.2 的功能取决于 P4_MFP[2] 和 P4_ALT[2]. <table><tr><th>P4_ ALT[2]</th><th>P4_MFP[2]</th><th>P4.2 功能</th></tr><tr><td>0</td><td>0</td><td>P4.2</td></tr><tr><td>0</td><td>1</td><td>PWM2(PWM 发生器 2)</td></tr><tr><td>1</td><td>x</td><td>保留</td></tr></table>			P4_ ALT[2]	P4_MFP[2]	P4.2 功能	0	0	P4.2	0	1	PWM2(PWM 发生器 2)	1	x	保留			
P4_ ALT[2]	P4_MFP[2]	P4.2 功能																	
0	0	P4.2																	
0	1	PWM2(PWM 发生器 2)																	
1	x	保留																	
[9]	P4_ ALT[1]	P4.1 复用功能选择 P4.1 的功能取决于 P4_MFP[1] 和 P4_ALT[1].																	

		<table> <tr> <th>P4_ALT[1]</th><th>P4_MFP[1]</th><th>P4.1 功能</th></tr> <tr> <td>0</td><td>0</td><td>P4.1</td></tr> <tr> <td>0</td><td>1</td><td>PWM1(PWM 发生器0)</td></tr> <tr> <td>1</td><td>0</td><td>T3EX</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P4_ALT[1]	P4_MFP[1]	P4.1 功能	0	0	P4.1	0	1	PWM1(PWM 发生器0)	1	0	T3EX	1	1	保留
P4_ALT[1]	P4_MFP[1]	P4.1 功能															
0	0	P4.1															
0	1	PWM1(PWM 发生器0)															
1	0	T3EX															
1	1	保留															
[8]	P4_ALT[0]	P4.0 复用功能选择 P4.0 的功能取决于 P4_MFP[0] 和 P4_ALT[0]. <table> <tr> <th>P4_ALT[0]</th><th>P4_MFP[0]</th><th>P4.0 功能</th></tr> <tr> <td>0</td><td>0</td><td>P4.0</td></tr> <tr> <td>0</td><td>1</td><td>PWM0(PWM 发生器0)</td></tr> <tr> <td>1</td><td>0</td><td>T2EX</td></tr> <tr> <td>1</td><td>1</td><td>保留</td></tr> </table>	P4_ALT[0]	P4_MFP[0]	P4.0 功能	0	0	P4.0	0	1	PWM0(PWM 发生器0)	1	0	T2EX	1	1	保留
P4_ALT[0]	P4_MFP[0]	P4.0 功能															
0	0	P4.0															
0	1	PWM0(PWM 发生器0)															
1	0	T2EX															
1	1	保留															
[7:0]	P4_MFP[7:0]	P4复用功能选择 P4 的功能取决于 P4_MFP 和 P4_ALT。 参照 P4_ALT来获取详细描述。															

Port5 多功能控制寄存器 (P5 MFP)

寄存器	偏移	R/W	描述	复位值
P5_MFP	GCR_BA+0x44	R/W	P5 多功能复用与输入类型控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
P5_TYPE[7:0]							
15	14	13	12	11	10	9	8
Reserved				P5_ALT[3:0]			
7	6	5	4	3	2	1	0
Reserved				P5_MFP[3:0]			

位	描述		
[31:24]	Reserved	保留	
[23:16]	P5_TYPEn	使能 P5[7:0] 施密特输入触发功能 0= 禁用 P5[7:0] I/O 施密特输入触发功能 1= 使能 P5[7:0] I/O 施密特输入触发功能	
[15:12]	Reserved	保留	
[11]	P5_ALT[3]	P5.3 复用功能选择 P5.3 的功能取决于 P5_MFP[3] 和 P5_ALT[3].	
		P5_ALT[3]	P5.3 功能
		0	P5.3
		0	SCL0(I ² C0)
		1	保留
[10]	P5_ALT[2]	P5.2 复用功能选择 P5.2 的功能取决于 P5_MFP[2] 和 P5_ALT[2].	
		P4_ALT[2]	P4.2 功能
		0	P5.2

		<table><tr><td>0</td><td>1</td><td>SDA0(I²C0)</td></tr><tr><td>1</td><td>x</td><td>保留</td></tr></table>	0	1	SDA0(I ² C0)	1	x	保留									
0	1	SDA0(I ² C0)															
1	x	保留															
[9]	P5_ALT[1]	P5.1 复用功能选择 P5.1 的功能取决于 P5_MFP[1] 和 P5_ALT[1]. <table><tr><th>P5_ALT[1]</th><th>P5_MFP[1]</th><th>P5.1 功能</th></tr><tr><td>0</td><td>0</td><td>P5.1</td></tr><tr><td>0</td><td>1</td><td>T1EX</td></tr><tr><td>1</td><td>0</td><td>保留</td></tr><tr><td>1</td><td>1</td><td>保留</td></tr></table>	P5_ALT[1]	P5_MFP[1]	P5.1 功能	0	0	P5.1	0	1	T1EX	1	0	保留	1	1	保留
P5_ALT[1]	P5_MFP[1]	P5.1 功能															
0	0	P5.1															
0	1	T1EX															
1	0	保留															
1	1	保留															
[8]	P5_ALT[0]	P5.0 复用功能选择 P5.0 的功能取决于 P5_MFP[0] 和 P5_ALT[0]. <table><tr><th>P5_ALT[0]</th><th>P5_MFP[0]</th><th>P5.0 功能</th></tr><tr><td>0</td><td>0</td><td>P5.0</td></tr><tr><td>0</td><td>1</td><td>T0EX</td></tr><tr><td>1</td><td>0</td><td>保留</td></tr><tr><td>1</td><td>1</td><td>保留</td></tr></table>	P5_ALT[0]	P5_MFP[0]	P5.0 功能	0	0	P5.0	0	1	T0EX	1	0	保留	1	1	保留
P5_ALT[0]	P5_MFP[0]	P5.0 功能															
0	0	P5.0															
0	1	T0EX															
1	0	保留															
1	1	保留															
[7:4]	Reserved	保留															
[3:0]	P5_MFP[3:0]	P5 复用功能选择 P5 的功能取决于 P5_MFP 和 P5_ALT。 参照 P5_ALT 来获取详细描述。															

Port6 多功能控制寄存器 (P6 MFP)

寄存器	偏移	R/W	描述	复位值
P6_MFP	GCR_BA+0x48	R/W	P6 多功能复用与输入类型控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
P6_TYPE[7:0]							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:24]	Reserved	保留
[23:16]	P6_TYPEn	使能 P6[7:0] 施密特输入触发功能 0= 禁用 P6[7:0] I/O 施密特输入触发功能 1= 使能 P6[7:0] I/O 施密特输入触发功能
[15:0]	Reserved	保留

Port7 多功能控制寄存器 (P7 MFP)

寄存器	偏移	R/W	描述	复位值
P7_MFP	GCR_BA+0x4C	R/W	P7 多功能复用与输入类型控制寄存器	0x0000_0003

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						P7_TYPE[1:0]	
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved						P7_MFP[1:0]	

位	描述	
[31:18]	Reserved	保留
[17:16]	P7_TYPEn	使能 P7[1:0] 施密特输入触发功能 0= 禁用 P7.0 和 P7.1施密特输入触发功能 1= 使能 P7.0 和 P7.1 施密特输入触发功能
[15:2]	Reserved	保留
[1:0]	P7_MFP[1:0]	P7 复用功能选择 P7 的功能取决于 CONFIG0[27] 0 = P7.0 和 P7.1 配置为 GPIO 引脚 1 = P7.0 和 P7.1 配置为外部 4~24MHz (高速) 晶振引脚

寄存器写保护控制寄存器 (REGWRPROT)

一些系统控制寄存器需要被保护起来，以防止误操作而影响芯片运行。这些系统控制寄存器在上电复位之后到用户解锁之前都是锁定的。为了使用户能够编写这些受保护的寄存器，需要下述的特殊编程来实现停用寄存器保护的时序。这个停用寄存器保护的时序就是连续依次写入“59h”，“16h”“88h”到寄存器REGWRPROT，地址为0x5000_0100。在这三个数据之间写入任何其他数据，不同时序或写入其他地址都会终止整个时序，导致无法解锁。

在解锁后，用户可以检测解锁指示位：地址0x5000_0100的bit0，1表示已经解锁定，0表示锁定。用户可以更新目标寄存器的值，向地址0x5000_0100写入任意值，就可以重新使能寄存器保护。

写这个寄存器用于禁用/使能寄存器保护功能，读这个寄存器用于得到REGPROTDIS的状态。

寄存器	偏移	R/W	描述	复位值
REGWRPROT	GCR_BA+0x100	R/W	寄存器写保护控制寄存器	0x0000_0000

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

位	描述	
[31:16]	Reserved	保留
[7:0]	REGWRPROT	寄存器写保护控制寄存器（只写） 一些寄存器具有写保护功能。写这些寄存器必须通过依次写入“59h”，“16h”，“88h”到REGWRPROT来关闭保护功能。这个时序完成后，REGPROTDIS位将被置为1，写保护寄存器可以正常写入数据。
[0]	REGPROTDIS	寄存器写保护禁用标志（只读） 0 = 写保护已使能，任何向被保护的寄存器的写入操作都将被忽略。 1 = 写保护已禁用。 受保护的寄存器有：

寄存器	地址	备注
IPRSTC1	0x5000_0008	
BODCR	0x5000_0018	
PORCR	0x5000_0024	
PWRCON	0x5000_0200	在电源唤醒中断被清时，bit[6]不受保护
APBCLK bit[0]	0x5000_0208	Bit[0] 是看门狗时钟使能
CLKSEL0	0x5000_0210	选择HCLK 和 CPU STCLK 的时钟源
CLKSEL1 bit[1:0]	0x5000_0214	选择看门狗的时钟源
NMI_SEL bit[8]	0x5000_0380	NMI 中断使能
ISPCON	0x5000_C000	Flash ISP 控制
ISPTRG	0x5000_C010	ISP 触发控制
WTCR	0x4000_4000	看门狗定时器控制
FATCON	0x5000_C018	Flash 访问时间控制

注: 这些受保护的位在寄存器描述旁注释为“（写保护）”。

6.2.6 系统定时器 (SysTick)

Cortex-M0 包含一个集成的系统定时器 — SysTick。它提供一种简单的，24位写清零、递减、自装载同时具有可灵活控制机制的计数器。该计数器可用作实时系统(RTOS) 的滴答定时器或一个简单的计数器。

当系统定时器使能后，将从 SysTick 的当前值寄存器 (SYST_CVR) 的值向下计数到0，并在下一个时钟周期，重新加载在 SysTick 重新加载值寄存器 (SYST_RVR) 的值，然后随接下来的时钟递减。当计数器减到0时，标志位COUNTFLAG置位，读 COUNTFLAG 位使其清零。

复位后，SYST_CVR 的值未知。在使能前，软件应该写该寄存器使其清零。这样确保定时器在使能后以SYST_RVR中的值计数，而非任意值。

若SYST_RVR是0，在重新加载这个值后，定时器将保持当前值0，这种机制可以用来在不使用系统定时器的使能位的情形下禁用系统定时器。

详情请参考“ARM® Cortex®-M0 Technical Reference Manual”和“ARM® v6-M Architecture Reference Manual”。

6.2.7 系统定时器控制寄存器映射

R: 只读, W: 只写 R/W: 可读写

寄存器	偏移	R/W	描述	复位值
SYST 基地址: SYST_BA = 0xE000_E010				
SYST_CSR	SYST_BA+0x00	R/W	SysTick 控制与状态寄存器	0x0000_0000
SYST_RVR	SYST_BA+0x04	R/W	SysTick 重新加载值寄存器	0xFFFF_FFFF
SYST_CVR	SYST_BA+0x08	R/W	SysTick 当前值寄存器	0xFFFF_FFFF

SysTick 控制和状态寄存器 (SYST_CSR)

寄存器	偏移	R/W	描述	复位值
SYST_CSR	SYST_BA+0x00	R/W	SysTick 控制和状态寄存器	0x0000_0000

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

位	描述	
[31:17]	Reserved	保留
[16]	COUNTFLAG	系统节拍器计数标志 从上次该寄存器被读，如果定时器计数到0，则返回1。 计数由1到0时，COUNTFLAG 被置位。 读该位或向当前值寄存器写时，COUNTFLAG 被清零。
[15:3]	Reserved	保留
[2]	CLKSRC	系统节拍器时钟源选择 0 = 时钟源可选，参照 STCLK_S。 1 = 内核时钟用于 SysTick timer。
[1]	TICKINT	系统节拍器中断使能 0 = 向下计数到 0 不会引起 SysTick 异常而挂起。软件根据 COUNTFLAG 来确定是否已经发生计数到 0。 1 = 向下计数到 0 将引起 SysTick 异常而挂起。软件清 SysTick 当前值寄存器将不会导致 SysTick 挂起。
[0]	ENABLE	系统节拍器计数器使能 0 = 禁用计数器 1 = 计数器使能并运行于连拍方式 (multi-shot manner)

SysTick 重新加载值寄存器 (SYST_RVR)

寄存器	偏移	R/W	描述	复位值
SYST_RVR	SYST_BA+0x04	R/W	SysTick 重新加载值寄存器	0xFFFF_XXXX

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

位	描述	
[31:24]	Reserved	保留
[23:0]	RELOAD	系统节拍器重装载值 当计数器达到 0 时，该值加载到当前值寄存器。

SysTick 当前值寄存器 (SYST_CVR)

寄存器	偏移	R/W	描述	复位值
SYST_CVR	SYST_BA+0x08	R/W	SysTick 当前值寄存器	0XXXXX_XXXX

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

位	描述	
[31:24]	Reserved	保留
[23:0]	CURRENT	系统节拍器当前值 当前计数值，为采样时刻的计数器的值，计数器不提供读-修改-写保护功能，该寄存器为写清零的，软件写入任何值将清寄存器为 0。这些位不支持读为零(read as zero)，参见系统重装载值寄存器 (SYST_RVR)。

6.2.8 嵌套向量中断控制器 (NVIC)

Cortex-M0 提供中断控制器，作为异常模式的组成部分，称之为“嵌套向量中断控制器 (NVIC)”。它与处理器内核紧密联系，并具有以下特性：

- 支持嵌套和向量中断
- 自动保存和恢复进程
- 简化的精确的中断延迟

NVIC 对所有支持的异常按优先级排序并处理。所有异常在“处理模式”处理。NVIC 结构支持具有四级优先级的 32 个 (IRQ[31:0]) 离散中断。所有的中断和大多数系统异常可以配置为不同的优先级。当中断发生时，NVIC 将比较新中断与当前中断的优先级，如果新中断优先级高于当前中断，则新中断将代替当前中断被处理。

当任何中断被响应时，中断服务程序 (ISR) 的起始地址从内存的向量表中取得。不需要由软件决定响应哪个中断，也不要软件跳转到相关 ISP 的起始地址。当取得起始地址时，NVIC 将自动保存处理器状态，包括以下寄存器 “PC, PSR, LR, R0~R3, R12” 的值到栈中。在 ISR 结束时，NVIC 将从栈中恢复提到的寄存器的值，恢复正常操作，因此处理器将花费更少的并且确定的时间去处理中断请求。

NVIC 支持末尾连锁 (Tail Chaining)，有效地处理尾对尾中断 (back-to-back interrupts)，无需重复保存和恢复当前状态，从而减少从当前 ISR 结束切换到等待处理的 ISR 的延迟时间。NVIC 还支持晚到 (Late Arrival)，可以提升同时发生的 ISR 的效率。在当前 ISR 开始执行 (保存处理器状态并获取起始地址阶段) 之前如果较高优先级中断请求发生，NVIC 将立即选择处理更高优先级的中断，从而提高了实时性。

详情请参考 “ARM® Cortex®-M0 Technical Reference Manual” 和 “ARM® v6-M Architecture Reference Manual”。

6.2.8.1 异常模式和系统中断映射

下表列出了 NuMicro M058S™ 系列支持的异常模型。软件可以对其中一些异常以及所有中断设置 4 级优先级。最高用户可配置优先级记为“0”，最低优先级记为“3”，所有用户可配置的优先级的默认值为“0”。注意：优先级“0”在整个系统中为第 4 优先级，排在“Reset”，“NMI”与“Hard Fault”之后。

异常名称	向量号	优先级
Reset	1	-3
NMI	2	-2
Hard Fault	3	-1
Reserved	4 ~ 10	保留
SVCall	11	可配置
Reserved	12 ~ 13	保留
PendSV	14	可配置

SysTick	15	可配置
Interrupt (IRQ0 ~ IRQ31)	16 ~ 47	可配置

表 6.2-2 异常模型

异常号	向量地址	中断号 (中断寄存器中的位)	中断名称	源模块	中断描述	掉电唤醒
1-15					系统异常	
16	0x40	0	BOD_INT	Brown-out	欠压检测中断	Yes
17	0x44	1	WDT_INT	WDT	看门狗定时器中断	Yes
18	0x48	2	EINT0	GPIO	P3.2管脚上的外部信号中断	Yes
19	0x4C	3	EINT1	GPIO	P3.3管脚上的外部信号中断	Yes
20	0x50	4	GP01_INT	GPIO	P0[7:0] / P1[7:0] 管脚上的外部信号中断	Yes
21	0x54	5	GP234_INT	GPIO	P2[7:0]/P3[7:0]/P4[7:0] 管脚上的外部信号中断, 除了 P32 和 P33	Yes
22	0x58	6	PWMA_INT	PWM0~3	PWM0, PWM1, PWM2 和 PWM3 中断	No
23	0x5C	7	保留	-	-	-
24	0x60	8	TMR0_INT	TMR0	Timer 0 中断	No
25	0x64	9	TMR1_INT	TMR1	Timer 1 中断	No
26	0x68	10	TMR2_INT	TMR2	Timer 2 中断	No
27	0x6C	11	TMR3_INT	TMR3	Timer 3 中断	No
28	0x70	12	UART0_INT	UART0	UART0 中断	Yes
29	0x74	13	保留	-	-	-
30	0x78	14	SPI0_INT	SPI0	SPI0 中断	No
31	0x7C	15	保留	-	-	-
32	0x80	16	GP5_INT	GPIO	P5[7:0] 管脚上的外部信号中断	Yes
33	0x84	17	GP67_INT	GPIO	P6[7:0] / P7[1:0] 管脚上的外部信号中断	Yes
34	0x88	18	I2C0_INT	I ² C0	I ² C0 中断	Yes

35	0x8C	19	I2C1_INT	I ² C1	I ² C1 中断	Yes
36	0x90	20	CAP0_INT	PWM	PWM0捕获中断	No
37	0x94	21	CAP1_INT	PWM	PWM1捕获中断	No
38	0x98	22	CAP2_INT	PWM	PWM2捕获中断	No
39	0x9C	23	CAP3_INT	PWM	PWM3捕获中断	No
40-43	0x90-0xAC	20-27	Reserved	-	-	-
44	0xB0	28	PWRWU_INT	CLKC	在掉电状态唤醒芯片的时钟控制器中断	Yes
45	0xB4	29	ADC_INT	ADC	ADC 中断	No
46-47	0xB8-0xBC	30-31	Reserved	-	-	

表 6.2-3 系统中断映射向量表

6.2.8.2 向量表

当响应中断时，处理器自动从内存的向量表中取出中断服务例程（ISR）的起始地址。对于 ARMv6-M，向量表的基地址固定为 0x00000000。向量表包括复位后堆栈的初始值以及所有异常处理器的入口地址。前一页的向量号定义了进入向量表与如上章节所示的异常处理入口有关的顺序。

向量表字偏移	描述
0	SP_main – 主栈指针
向量号	异常入口指针，用向量号表示

表 6.2-4 向量表格式

6.2.8.3 操作说明

通过写相应中断使能设置寄存器或清使能寄存器位域，可以使能NVIC中断或禁用NVIC中断，这些寄存器通过写1使能和写1清除为零，读取这两个寄存器均返回当前相应中断的使能状态。当某一个中断被禁用时，中断声明将使该中断处于等待处理状态，然而，该中断不会被激活。如果某一个中断在被禁用时处于激活状态，该中断就保持在激活状态，直到通过复位或异常返回来清除。清使能位可以阻止新的相应中断被激活。

NVIC中断可以使用互补的寄存器对来挂起/解除挂起以使能/禁用这些中断，这些寄存器分别为 Set-Pending寄存器与Clear-Pending寄存器，这些寄存器使用写1使能和写1清除的方式，读取这两种寄存器都返回相应中断的当前挂起状态。Clear-Pending寄存器不会对处于激活状态的中断的执行状态产生任何影响。

NVIC中断通过更新32位寄存器中的各个8位字段(每个寄存器支持4个中断)来分配中断的优先级。

与NVIC相关的通用寄存器都可以通过系统控制空间的内存区域访问，下一节将作出描述。

6.2.8.4 NVIC控制寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移	R/W	描述	复位值
NVIC 基地址: NVIC_BA = 0xE000_E100				
NVIC_ISER	NVIC_BA+0x000	R/W	IRQ0 ~ IRQ31 设置使能控制寄存器	0x0000_0000
NVIC_ICER	NVIC_BA+0x080	R/W	IRQ0 ~ IRQ31 清使能控制寄存器	0x0000_0000
NVIC_ISPR	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ31 设置挂起控制寄存器	0x0000_0000
NVIC_ICPR	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ31 清挂起控制寄存器	0x0000_0000
NVIC_IPR0	NVIC_BA+0x300	R/W	IRQ0 ~ IRQ3 优先级控制寄存器	0x0000_0000
NVIC_IPR1	NVIC_BA+0x304	R/W	IRQ4 ~ IRQ7 优先级控制寄存器	0x0000_0000
NVIC_IPR2	NVIC_BA+0x308	R/W	IRQ8 ~ IRQ11 优先级控制寄存器	0x0000_0000
NVIC_IPR3	NVIC_BA+0x30C	R/W	IRQ12 ~ IRQ15 优先级控制寄存器	0x0000_0000
NVIC_IPR4	NVIC_BA+0x310	R/W	IRQ16 ~ IRQ19 优先级控制寄存器	0x0000_0000
NVIC_IPR5	NVIC_BA+0x314	R/W	IRQ20 ~ IRQ23 优先级控制寄存器	0x0000_0000
NVIC_IPR6	NVIC_BA+0x318	R/W	IRQ24 ~ IRQ27 优先级控制寄存器	0x0000_0000
NVIC_IPR7	NVIC_BA+0x31C	R/W	IRQ28 ~ IRQ31 优先级控制寄存器	0x0000_0000

IRQ0 ~ IRQ31设置使能控制寄存器 (NVIC_ISER)

寄存器	偏移	R/W	描述	复位值
NVIC_ISER	NVIC_BA+0x000	R/W	IRQ0 ~ IRQ31设置使能控制寄存器	0x0000_0000

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

位	描述	
[31:0]	SETENA	中断使能寄存器 使能一个或者多个中断，每位代表 IRQ0 ~ IRQ31 的中断号（向量号：从16到47） 写： 0 = 无效果 1 = 写 1 使能相关中断 读： 0 = 相关中断状态被禁止 1 = 相关中断状态已使能 读取该寄存器返回当前使能状态

IRQ0 ~ IRQ31清使能控制寄存器(NVIC_ICER)

寄存器	偏移	R/W	描述	复位值
NVIC_ICER	NVIC_BA+0x080	R/W	IRQ0 ~ IRQ31清使能控制寄存器	0x0000_0000

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

位	描述	
[31:0]	CLRENA	中断禁用位 禁用一个或者多个中断，每位代表 IRQ0 ~ IRQ31 的中断号（向量号：从16到 47） 写： 0 = 无效 1 = 写1禁用相关中断 读： 0 = 相关中断状态被禁止 1 = 相关中断状态已使能 读取该寄存器返回当前使能状态

IRQ0 ~ IRQ31设置挂起控制寄存器 (NVIC ISPR)

寄存器	偏移	R/W	描述	复位值
NVIC_ISPR	NVIC_BA+0x100	R/W	IRQ0 ~ IRQ31设置挂起控制寄存器	0x0000_0000

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

位	描述	
[31:0]	SETPEND	设置中断挂起寄存器 写: 0 = 无效果 1 = 写 1, 挂起相应中断。每位代表 IRQ0 ~ IRQ31 的中断号 (向量号: 从16到47) 读: 0 = 相关中断不在挂起状态 1 = 相关中断处于挂起状态 读取该寄存器返回当前等待处理的中断状态

IRQ0 ~ IRQ31清挂起控制寄存器 (NVIC_ICPR)

寄存器	偏移	R/W	描述	复位值
NVIC_ICPR	NVIC_BA+0x180	R/W	IRQ0 ~ IRQ31清挂起控制寄存器	0x0000_0000

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

位	描述	
[31:0]	CLRPEND	清中断挂起寄存器 写操作: 0 = 无效 1 = 写1清除挂起状态。每位代表 IRQ0 ~ IRQ31 的中断号 (向量号: 从16到47) 读操作: 0 = 相关寄存器不在挂起状态 1 = 相关寄存器处于挂起状态 读取该寄存器返回当前等待处理的中断状态

IRQ0 ~ IRQ3 中断优先级寄存器 (NVIC_IPR0)

寄存器	偏移	R/W	描述	复位值
NVIC_IPR0	NVIC_BA+0x300	R/W	IRQ0 ~ IRQ3 优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_3		Reserved					
23	22	21	20	19	18	17	16
PRI_2		Reserved					
15	14	13	12	11	10	9	8
PRI_1		Reserved					
7	6	5	4	3	2	1	0
PRI_0		Reserved					

位	描述	
[31:30]	PRI_3	IRQ3 优先级 “0”表示最高优先级,“3”表示最低优先级
[23:22]	PRI_2	IRQ2优先级 “0”表示最高优先级,“3”表示最低优先级
[15:14]	PRI_1	IRQ1优先级 “0”表示最高优先级,“3”表示最低优先级
[7:6]	PRI_0	IRQ0优先级 “0”表示最高优先级,“3”表示最低优先级

IRQ4 ~ IRQ7 中断优先级寄存器 (NVIC_IPR1)

寄存器	偏移	R/W	描述	复位值
NVIC_IPR1	NVIC_BA+0x304	R/W	IRQ4 ~ IRQ7 优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_7		Reserved					
23	22	21	20	19	18	17	16
PRI_6		Reserved					
15	14	13	12	11	10	9	8
PRI_5		Reserved					
7	6	5	4	3	2	1	0
PRI_4		Reserved					

位	描述	
[31:30]	PRI_7	IRQ7优先级 “0”表示最高优先级,“3”表示最低优先级
[23:22]	PRI_6	IRQ6优先级 “0”表示最高优先级,“3”表示最低优先级
[15:14]	PRI_5	IRQ5优先级 “0”表示最高优先级,“3”表示最低优先级
[7:6]	PRI_4	IRQ4优先级 “0”表示最高优先级,“3”表示最低优先级

IRQ8 ~ IRQ11 中断优先级寄存器 (NVIC IPR2)

寄存器	偏移	R/W	描述	复位值
NVIC_IPR2	NVIC_BA+0x308	R/W	IRQ8 ~ IRQ11 优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_11		Reserved					
23	22	21	20	19	18	17	16
PRI_10		Reserved					
15	14	13	12	11	10	9	8
PRI_9		Reserved					
7	6	5	4	3	2	1	0
PRI_8		Reserved					

位	描述	
[31:30]	PRI_11	IRQ11优先级 “0”表示最高优先级,“3”表示最低优先级
[23:22]	PRI_10	IRQ10优先级 “0”表示最高优先级,“3”表示最低优先级
[15:14]	PRI_9	IRQ9优先级 “0”表示最高优先级,“3”表示最低优先级
[7:6]	PRI_8	IRQ8优先级 “0”表示最高优先级,“3”表示最低优先级

IRQ12 ~ IRQ15 中断优先级寄存器 (NVIC IPR3)

寄存器	偏移	R/W	描述	复位值
NVIC_IPR3	NVIC_BA+0x30C	R/W	IRQ12 ~ IRQ15 优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_15		Reserved					
23	22	21	20	19	18	17	16
PRI_14		Reserved					
15	14	13	12	11	10	9	8
PRI_13		Reserved					
7	6	5	4	3	2	1	0
PRI_12		Reserved					

位	描述	
[31:30]	PRI_15	IRQ15优先级 “0”表示最高优先级,“3”表示最低优先级
[23:22]	PRI_14	IRQ14优先级 “0”表示最高优先级,“3”表示最低优先级
[15:14]	PRI_13	IRQ13优先级 “0”表示最高优先级,“3”表示最低优先级
[7:6]	PRI_12	IRQ12优先级 “0”表示最高优先级,“3”表示最低优先级

IRQ16 ~ IRQ19 中断优先级寄存器 (NVIC IPR4)

寄存器	偏移	R/W	描述	复位值
NVIC_IPR4	NVIC_BA+0x310	R/W	IRQ16 ~ IRQ19 优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_19		Reserved					
23	22	21	20	19	18	17	16
PRI_18		Reserved					
15	14	13	12	11	10	9	8
PRI_17		Reserved					
7	6	5	4	3	2	1	0
PRI_16		Reserved					

位	描述	
[31:30]	PRI_19	IRQ19优先级 “0”表示最高优先级,“3”表示最低优先级
[23:22]	PRI_18	IRQ18优先级 “0”表示最高优先级,“3”表示最低优先级
[15:14]	PRI_17	IRQ17优先级 “0”表示最高优先级,“3”表示最低优先级
[7:6]	PRI_16	IRQ16优先级 “0”表示最高优先级,“3”表示最低优先级

IRQ20 ~ IRQ23 中断优先级寄存器 (NVIC IPR5)

寄存器	偏移	R/W	描述	复位值
NVIC_IPR5	NVIC_BA+0x314	R/W	IRQ20 ~ IRQ23 优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_23		Reserved					
23	22	21	20	19	18	17	16
PRI_22		Reserved					
15	14	13	12	11	10	9	8
PRI_21		Reserved					
7	6	5	4	3	2	1	0
PRI_20		Reserved					

位	描述	
[31:30]	PRI_23	IRQ23优先级 “0”表示最高优先级,“3”表示最低优先级
[23:22]	PRI_22	IRQ22优先级 “0”表示最高优先级,“3”表示最低优先级
[15:14]	PRI_21	IRQ21优先级 “0”表示最高优先级,“3”表示最低优先级
[7:6]	PRI_20	IRQ20优先级 “0”表示最高优先级,“3”表示最低优先级

IRQ24 ~ IRQ27 中断优先级寄存器 (NVIC IPR6)

寄存器	偏移	R/W	描述	复位值
NVIC_IPR6	NVIC_BA+0x318	R/W	IRQ24 ~ IRQ27 优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_27		Reserved					
23	22	21	20	19	18	17	16
PRI_26		Reserved					
15	14	13	12	11	10	9	8
PRI_25		Reserved					
7	6	5	4	3	2	1	0
PRI_24		Reserved					

位	描述	
[31:30]	PRI_27	IRQ27优先级 “0”表示最高优先级,“3”表示最低优先级
[23:22]	PRI_26	IRQ26优先级 “0”表示最高优先级,“3”表示最低优先级
[15:14]	PRI_25	IRQ25优先级 “0”表示最高优先级,“3”表示最低优先级
[7:6]	PRI_24	IRQ24优先级 “0”表示最高优先级,“3”表示最低优先级

IRQ28 ~ IRQ31 中断优先级寄存器 (NVIC IPR7)

寄存器	偏移	R/W	描述	复位值
NVIC_IPR7	NVIC_BA+0x31C	R/W	IRQ28 ~ IRQ31 优先级控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PRI_31		Reserved					
23	22	21	20	19	18	17	16
PRI_30		Reserved					
15	14	13	12	11	10	9	8
PRI_29		Reserved					
7	6	5	4	3	2	1	0
PRI_28		Reserved					

位	描述	
[31:30]	PRI_31	IRQ31优先级 “0”表示最高优先级,“3”表示最低优先级
[23:22]	PRI_30	IRQ30优先级 “0”表示最高优先级,“3”表示最低优先级
[15:14]	PRI_29	IRQ29优先级 “0”表示最高优先级,“3”表示最低优先级
[7:6]	PRI_28	IRQ28优先级 “0”表示最高优先级,“3”表示最低优先级

中断源控制寄存器

除了与NVIC相关的中断控制寄存器外，NuMicro M058S™系列还有一些特殊控制寄存器以利于中断功能，包括“中断源识别”，”NMI 源选择”与“中断测试模式”。描述如下：

R: 只读, **W:** 只写, **R/W:** 可读写

寄存器	偏移	R/W	描述	复位值
INT 基地址: INT_BA = 0x5000_0300				
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) 中断源识别	0xxxxx_xxxx
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) 中断源识别	0xxxxx_xxxx
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) 中断源识别	0xxxxx_xxxx
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) 中断源识别	0xxxxx_xxxx
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (P0/1) 中断源识别	0xxxxx_xxxx
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (P2/3/4) 中断源识别	0xxxxx_xxxx
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWMA) 中断源识别	0xxxxx_xxxx
IRQ7_SRC	INT_BA+0x1C	R	保留	0xxxxx_xxxx
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) 中断源识别	0xxxxx_xxxx
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) 中断源识别	0xxxxx_xxxx
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (TMR2) 中断源识别	0xxxxx_xxxx
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (TMR3) 中断源识别	0xxxxx_xxxx
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (UART0) 中断源识别	0xxxxx_xxxx
IRQ13_SRC	INT_BA+0x34	R	保留	0xxxxx_xxxx
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI0) 中断源识别	0xxxxx_xxxx
IRQ15_SRC	INT_BA+0x3C	R	保留	0xxxxx_xxxx
IRQ16_SRC	INT_BA+0x40	R	IRQ16 (P5) 中断源识别	0xxxxx_xxxx
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (P6/7) 中断源识别	0xxxxx_xxxx
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I ² C0) 中断源识别	0xxxxx_xxxx
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (I2C1) 中断源识别	0xxxxx_xxxx
IRQ20_SRC	INT_BA+0x50	R	IRQ20 (CAP0) 中断源识别	0xxxxx_xxxx
IRQ21_SRC	INT_BA+0x54	R	IRQ21 (CAP1) 中断源识别	0xxxxx_xxxx

IRQ22_SRC	INT_BA+0x58	R	IRQ22 (CAP2) 中断源识别	0XXXXX_XXXX
IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (CAP3) 中断源识别	0XXXXX_XXXX
IRQ24_SRC	INT_BA+0x60	R	保留	0XXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	保留	0XXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	保留	0XXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	R	保留	0XXXXX_XXXX
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU) 中断源识别	0XXXXX_XXXX
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) 中断源识别	0XXXXX_XXXX
IRQ30_SRC	INT_BA+0x78	R	保留	0XXXXX_XXXX
IRQ31_SRC	INT_BA+0x7C	R	保留	0XXXXX_XXXX
NMI_SEL	INT_BA+0x80	R/W	NMI Source Interrupt Select Control Register	0x0000_0000

IRQ0 (BOD) 中断源识别寄存器 (IRQ0_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) 中断源识别	0xxxxx_xxxx

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

位	描述	
[2:0]	INT_SRC	IRQ0 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ0 源不是来自BOD中断 (BOD_INT). 1 = IRQ0 源来自 BOD中断 (BOD_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ1 (WDT) 中断源识别寄存器 (IRQ1_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) 中断源识别	0XXXXX_XXXX

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

位	描述	
[2:0]	INT_SRC	IRQ1 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ1 源不是来自看门狗中断 (WDT_INT). 1 = IRQ1 源来自看门狗中断 (WDT_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ2 (EINT0) 中断源识别寄存器 (IRQ2_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) 中断源识别	0xxxxx_xxxx

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

位	描述
[2:0]	IRQ2 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ2源不是来自外部中断 0 – P3.2 (EINT0). 1 = IRQ2源来自外部中断 0 – P3.2 (EINT0). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ3 (EINT1) 中断源识别寄存器 (IRQ3_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) 中断源识别	0xxxxx_xxxx

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

位	描述	
[2:0]	INT_SRC	IRQ3 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ3 源不是来自外部中断1 – P3.3 (EINT1). 1 = IRQ3 源来自外部中断1 – P3.3 (EINT1). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ4 (P0/I) 中断源识别寄存器 (IRQ4_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (P0/I) 中断源识别	0XXXXX_XXXX

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

位	描述	
[2:0]	INT_SRC	IRQ4 源识别 INT_SRC[2] = 保留 INT_SRC[1]: 0 = IRQ4 源不是来自 P1 中断 (P1_INT). 1 = IRQ4 源来自 P1 中断 (P1_INT). INT_SRC[0]: 0 = IRQ4 源不是来自 P0 中断 (P0_INT). 1 = IRQ4 源来自 P0 中断 (P0_INT). 注1: 同一时刻, IRQ4 中断可以来自多个中断源. 注2: 当中断标志被清除时, 相应的位将被自动清除。

IRQ5 (P2/3/4) 中断源识别寄存器 (IRQ5_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (P2/3/4) 中断源识别	0xxxxx_xxxx

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

位	描述
[2:0]	IRQ5 源识别 INT_SRC[2]: 0 = IRQ5 源不是来自 P4 中断 (P4_INT). 1 = IRQ5 源来自 P4 中断 (P4_INT). INT_SRC[1]: 0 = IRQ5 源不是来自 P3 中断 (P3_INT). 1 = IRQ5 源来自 P3 中断 (P3_INT). INT_SRC[0]: 0 = IRQ5 源不是来自 P2 中断 (P2_INT). 1 = IRQ5 源来自 P2 中断 (P2_INT). 注1: 同一时刻, IRQ5 中断可以来自多个中断源. 注2: 当中断标志被清除时, 相应的位将被自动清除。

IRQ6 (PWMA) 中断源识别寄存器 (IRQ6_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWMA) 中断源识别	0xxxxx_xxxx

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

位	描述
[3:0]	IRQ6 源识别 INT_SRC[3]: 0 = IRQ6 源不是来自PWM3(PWMA 通道 3) 中断 (PWM3_INT). 1 = IRQ6 源来自PWM3(PWMA 通道 3) 中断 (PWM3_INT). INT_SRC[2]: 0 = IRQ6 源不是来自PWM2(PWMA 通道 2) 中断 (PWM2_INT). 1 = IRQ6 源来自PWM2(PWMA 通道 2) 中断 (PWM2_INT). INT_SRC[1]: 0 = IRQ6 源不是来自PWM1(PWMA 通道 1) 中断 (PWM1_INT). 1 = IRQ6 源来自PWM1(PWMA 通道 1) 中断 (PWM1_INT). INT_SRC[0]: 0 = IRQ6 源不是来自PWM0(PWMA 通道 0) 中断 (PWM0_INT). 1 = IRQ6 源来自PWM0(PWMA 通道 0) 中断 (PWM0_INT). 注1: 同一时刻, IRQ6 中断可以来自多个中断源. 注2: 当中断标志被清除时, 相应的位将被自动清除.

IRQ8 (TMR0) 中断源识别寄存器 (IRQ8_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) 中断源识别	0xxxxx_xxxx

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

位	描述
[2:0]	IRQ8 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ8 源不是来自Timer0中断 (TMR0_INT). 1 = IRQ8 源来自Timer0中断 (TMR0_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ9 (TMR1) 中断源识别寄存器 (IRQ9_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) 中断源识别	0xxxxx_xxxx

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

位	描述
[2:0]	IRQ9 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ9 源不是来自Timer1中断 (TMR1_INT). 1 = IRQ9 源来自Timer1中断 (TMR1_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ10 (TMR2) 中断源识别寄存器 (IRQ10_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (TMR2) 中断源识别	0xxxxx_xxxx

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

位	描述	
[2:0]	INT_SRC	IRQ10 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ10 源不是来自Timer2中断 (TMR2_INT). 1 = IRQ10 源来自Timer2中断 (TMR2_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ11 (TMR3) 中断源识别寄存器 (IRQ11_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (TMR3) 中断源识别	0xxxxx_xxxx

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

位	描述	
[2:0]	INT_SRC	IRQ11 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ11 源不是来自Timer3中断 (TMR3_INT). 1 = IRQ11 源来自Timer3中断 (TMR3_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ12 (UART0) 中断源识别寄存器 (IRQ12_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (UART0) 中断源识别	0xxxxx_xxxx

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

位	描述	
[2:0]	INT_SRC	IRQ12 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ12 源不是来自UART0中断 (UART0_INT). 1 = IRQ12 源来自UART0中断 (UART0_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ14 (SPI0) 中断源识别寄存器 (IRQ14_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI0) 中断源识别	0xxxxx_xxxx

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

位	描述	
[2:0]	INT_SRC	IRQ14 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ14 源不是来自SPI0中断 (SPI0_INT). 1 = IRQ14 源来自SPI0中断 (SPI0_INT). 注:当中断标志被清除时，相应的位将被自动清除。

IRQ16 (P5) 中断源识别寄存器 (IRQ16_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ16_SRC	INT_BA+0x10	R	IRQ16 (P5) 中断源识别	0XXXXX_XXXX

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

位	描述	
[2:0]	INT_SRC	IRQ16 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ16 源不是来自P5中断 (P5_INT). 1 = IRQ16 源来自P5中断 (P5_INT). 注1: 同一时刻, IRQ16 中断可以来自多个中断源. 注2: 当中断标志被清除时, 相应的位将被自动清除。

IRQ17 (P6/7) 中断源识别寄存器 (IRQ17_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (P6/7) 中断源识别	0xFFFF_FFFF

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

位	描述
[2:0]	IRQ5 源识别 INT_SRC[2] = 保留 INT_SRC[1]: 0 = IRQ17 源不是来自P7中断 (P7_INT). 1 = IRQ17 源来自P7中断 (P7_INT). INT_SRC[0]: 0 = IRQ17 源不是来自P6中断 (P6_INT). 1 = IRQ17 源来自P6中断 (P6_INT). 注1: 同一时刻, IRQ17 中断可以来自多个中断源. 注2: 当中断标志被清除时, 相应的位将被自动清除。

IRQ18 (I²C0) 中断源识别寄存器 (IRQ18_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I ² C0) 中断源识别	0xxxxx_xxxx

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

位	描述	
[2:0]	INT_SRC	IRQ18 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ18 源不是来自I²C0中断 (I2C0_INT). 1 = IRQ18 源来自I²C0中断 (I2C0_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ19 (I²C1) 中断源识别寄存器 (IRQ19_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (I ² C1) 中断源识别	0XXXXX_XXXX

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

位	描述	
[2:0]	INT_SRC	IRQ19 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ19 源不是来自I²C1中断 (I2C1_INT). 1 = IRQ19 源来自I²C1中断 (I2C1_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ20(CAP0) 中断源识别寄存器 (IRQ20_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ20_SRC	INT_BA+0x50	R	IRQ20 (CAP0) 中断源识别	0xxxxx_xxxx

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

位	描述
[2:0]	INT_SRC IRQ20 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ20 源不是来自PWM0 捕获中断 (CAP0_INT). 1 = IRQ20 源来自PWM0 捕获中断 (CAP0_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ21(CAP1) 中断源识别寄存器 (IRQ21_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ21_SRC	INT_BA+0x54	R	IRQ20 (CAP1) 中断源识别	0xxxxx_xxxx

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

位	描述	
[2:0]	INT_SRC	IRQ21 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ21 源不是来自PWM1 捕获中断 (CAP1_INT). 1 = IRQ21 源来自PWM1 捕获中断 (CAP1_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ22(CAP2) 中断源识别寄存器 (IRQ22_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (CAP2) 中断源识别	0xxxxx_xxxx

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

位	描述	
[2:0]	INT_SRC	IRQ20 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ22 源不是来自PWM2 捕获中断 (CAP2_INT). 1 = IRQ22 源来自PWM2 捕获中断 (CAP2_INT). 注:当中断标志被清除时，相应的位将被自动清除。

IRQ23(CAP3) 中断源识别寄存器 (IRQ23_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (CAP0) 中断源识别	0xxxxx_xxxx

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

位	描述	
[2:0]	INT_SRC	IRQ23 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ23 源不是来自PWM3 捕获中断 (CAP3_INT). 1 = IRQ23 源来自PWM3 捕获中断 (CAP3_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ28 (PWRWU) 中断源识别寄存器 (IRQ28_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU) 中断源识别	0xxxxx_xxxx

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

位	描述	
[2:0]	INT_SRC	IRQ28 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ28 源不是来自睡眠模式下的唤醒中断 (PWRWU_INT). 1 = IRQ28 源来自睡眠模式下的唤醒中断 (PWRWU_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

IRQ29 (ADC) 中断源识别寄存器 (IRQ29_SRC)

寄存器	偏移	R/W	描述	复位值
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (ADC) 中断源识别	0xxxxx_xxxx

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

位	描述	
[2:0]	INT_SRC	IRQ29 源识别 INT_SRC[2] = 保留 INT_SRC[1] = 保留 INT_SRC[0]: 0 = IRQ29 源不是来自ADC中断 (ADC_INT). 1 = IRQ29 源来自ADC中断 (ADC_INT). 注:当中断标志被清除时, 相应的位将被自动清除。

NMI中断源选择控制寄存器(NMI_SEL)

寄存器	偏移	R/W	描述	复位值
NMI_SEL	INT_BA+0x80	R/W	NMI中断源选择控制寄存器	0x0000_0000

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

位	描述	
[31:5]	Reserved	保留
[8]	NMI_EN	NMI中断使能 (写保护) 1 = 使能 NMI 中断 0 = 禁止 NMI 注: 该位是写保护的, 程序需要依次向地址0x5000_0100写入下列数据 “59h”, “16h”, 和 “88h” 来关闭保护功能。参考寄存器REGWRPROT, 地址 GCR_BA+0x100。
[4:0]	NMI_SEL	NMI 中断源选择 通过设置 NMI_SEL 可以在外围设备中断中选择Cortex®-M0 的 NMI 中断。

6.2.9 系统控制块 (SCB)

Cortex®-M0 的状态和操作模式都由系统控制寄存器来控制，包括 CPUID，Cortex®-M0 中断优先级和 Cortex®-M0 电源管理都可以通过系统控制寄存器来控制。

更多详情请参考 “ARM® Cortex®-M0 Technical Reference Manual” 和 “ARM® v6-M Architecture Reference Manual”。

6.2.9.1 系统控制块寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移	R/W	描述	复位值
SCB 基地址: SCB_BA = 0xE000_ED00				
CPUID	SCB_BA+0x000	R	CPUID 寄存器	0x410CC200
ICSR	SCB_BA+0x004	R/W	中断控制状态寄存器	0x0000_0000
AIRCR	SCB_BA+0x00C	R/W	中断应用和复位控制寄存器	0xFA05_0000
SCR	SCB_BA+0x010	R/W	系统控制寄存器	0x0000_0000
SHPR2	SCB_BA+0x01C	R/W	系统处理优先级寄存器 2	0x0000_0000
SHPR3	SCB_BA+0x020	R/W	系统处理优先级寄存器 3	0x0000_0000

CPUID 基础寄存器 (CPUID)

寄存器	偏移	R/W	描述	复位值
CPUID	SCB_BA+0x000	R	CPUID 寄存器	0x410CC200

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

位	描述	
[31:24]	IMPLEMENTER	执行码 执行码由 ARM 分配 (ARM = 0x41)
[23:20]	Reserved	保留
[19:16]	PART	处理器架构 ARMv6-M 读出的值为 0xC
[15:4]	PARTNO	处理器编号 读出的值为 0xC20.
[3:0]	REVISION	修订版本号 读出的值为 0x0

中断控制状态寄存器 (ICSR)

寄存器	偏移	R/W	描述	复位值
ICSR	SCB_BA+0x004	R/W	中断控制状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
NMIPENDSET	Reserved		PENDSVSET	PENDSVCLR	PENDSTSET	PENDSTCLR	Reserved
23	22	21	20	19	18	17	16
ISRPREEMPT	ISRPENDING	Reserved				VECTPENDING[5:4]	
15	14	13	12	11	10	9	8
VECTPENDING[3:0]				Reserved			
7	6	5	4	3	2	1	0
Reserved		VECTACTIVE[5:0]					

位	描述	
[31]	NMIPENDSET	NMI 设置挂起位 写: 0 = 无效果 1 = 更改 NMI 异常状态为挂起 读: 0 = NMI 异常没有挂起 1 = NMI 异常挂起 注: 因为 NMI 是最高优先级的异常, 正常情况下处理器一旦检测到这一位被写1, 就会立刻进入 NMI 异常处理程序。进入处理程序后处理器会将该位清为零。这意味着通过 NMI异常处理来读取该位返回1时, 只有当处理器正在执行处理时, 再次产生NMI 信号这一种情形。
[30:29]	Reserved	保留
[28]	PENDSVSET	PendSV 设置挂起位 写: 0 = 无效果 1 = 更改 PendSV异常状态为挂起 读: 0 = PendSV异常没有挂起 1 = PendSV异常挂起 注: 向该位写1是将 PendSV 异常状态设为挂起的唯一方法。

[27]	PENDSVCLR	PendSV清除挂起位 写: 0 = 无效果 1 = 移除 PendSV 异常的挂起状态 只写位, 当想清除 PENDSV 位时, 必须同时“向 PENDSVSET 写 0 和向 PENDSVCLR 写 1”。
[26]	PENDSTSET	SysTick 异常设置挂起位 写: 0 = 无效果 1 = 更改 SysTick异常状态为挂起 读: 0 = SysTick异常没有挂起 1 = SysTick异常挂起
[25]	PENDSTCLR	SysTick 异常清除挂起位 写: 0 = 无效果 1 = 移除 SysTick 异常的挂起状态 注: 只写位。当想清除 PENDST 位时, 必须同时“向 PENDSTSET 写0 和 向 PENDSTCLR 写 1”。
[24]	Reserved	保留
[23]	ISRPREEMPT	中断抢占位 若置位, 则挂起的异常将从调试停止状态退出, 进入活动状态。 该位为只读
[22]	ISRPENDING	中断挂起标志, 不包括 NMI 和 Faults (只读) 0 = 中断没有挂起 1 =中断挂起
[21:18]	Reserved	保留
[17:12]	VECTPENDING	最高优先级挂起已使能异常的异常号 0 =没有异常挂起 非零 =最高优先级挂起已使能异常的异常号
[11:6]	Reserved	保留
[5:0]	VECTACTIVE	包含处于活动状态的异常号 0 = 线程模式 非零 = 当前处于活动状态的异常的异常号

应用中断和复位控制寄存器(AIRCR)

寄存器	偏移	R/W	描述	复位值
AIRCR	SCB_BA+0x00C	R/W	应用中断和复位控制寄存器	0xFA05_0000

31	30	29	28	27	26	25	24
VECTORKEY[15:8]							
23	22	21	20	19	18	17	16
VECTORKEY[7:0]							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved					SYSRESET REQ	VECTCLKA CTIVE	Reserved

位	描述	
[31:16]	VECTORKEY	寄存器访问密钥 写： 作时，VECTORKEY 区域必须写入 0x05FA，否则，写操作会被忽略。VECTORKEY 区域被用来防止误写该寄存器，以免重设系统或清除异常状态。 读： 读出的值为 0xFA05
[15:3]	Reserved	保留
[2]	SYSRESETREQ	系统复位请求 该位写1，将产生一个复位信号给芯片表示有复位请求。 该位是只写的，在复位时自动清零。
[1]	VECTCLRACTIVE	异常活动状态清除位 保留为调试模式使用。写该寄存器时，用户必须向该位写0，否则将产生不可预测的结果。
[0]	Reserved	保留

系统控制寄存器(SCR)

寄存器	偏移	R/W	描述	复位值
SCR	SCB_BA+0x010	R/W	系统控制寄存器	0x0000_0000

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

位	描述	
[31:5]	Reserved	保留
[4]	SEVONPEND	在挂起状态时的发送事件位 0 = 只有使能中断或者事件可以唤醒处理器，不包括禁用中断在内。 1 = 使能事件和所有中断，包括禁用中断，都可以唤醒处理器。 当一个事件或中断进入挂起状态，事件信号从 WFE 唤醒处理器。如果处理器不是在等待该事件，该事件将会被注册，并在下一个WFE中有效。 处理器也会在执行一个 SEV 指令或者一个外部事件时被唤醒。
[3]	Reserved	保留
[2]	SLEEPDEEP	系统深度休眠和休眠模式选择 控制处理器在低功耗模式下是使用休眠还是深度休眠： 0 = 休眠模式 1 = 深度休眠模式
[1]	SLEEPONEXIT	退出时睡眠（Sleep-on-exit）使能控制 表示当从 Handler 模式切换到 Thread 模式时，是否使用 sleep-on-exit 0 = 当切换到 Thread 模式时不睡眠。 1 = 当从中断处理函数返回Thread模式时进入睡眠或者深度睡眠模式 设置该位为1 使能一个中断驱动应用，避免返回到一个空的主函数应用。
[0]	Reserved	保留

系统处理优先级寄存器2 (SHPR2)

寄存器	偏移	R/W	描述	复位值
SHPR2	SCB_BA+0x01C	R/W	系统处理优先级寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
PRI_11		Reserved					
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:30]	PRI_11	系统处理 11 – SVCall “0” 表示最高优先级,“3” 表示最低优先级
[29:0]	Reserved	保留

系统处理优先级寄存器 3 (SHPR3)

寄存器	偏移	R/W	描述	复位值
SHPR3	SCB_BA+0x020	R/W	系统处理优先级寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
PRI_15		Reserved					
23	22	21	20	19	18	17	16
PRI_14		Reserved					
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved							

位	描述	
[31:30]	PRI_15	系统处理 15 – SysTick “0”表示最高优先级,“3”表示最低优先级
[29:24]	Reserved	保留
[23:22]	PRI_14	系统处理14 – PendSV “0”表示最高优先级,“3”表示最低优先级
[21:0]	Reserved	保留

6.3 时钟控制器

6.3.1 概述

时钟控制器为整个芯片提供时钟源，包括系统时钟和所有外围设备时钟。该控制器还通过单独时钟的开或关，时钟源选择和分频器来进行功耗控制。PWR_DOWN_EN (PWRCON[7]) 位和 PD_WAIT_CPU(PWRCON[8]) 位同时设置为1, 并且CPU Cortex®-M0内核执行WFI指令，芯片将进入掉电模式。直到唤醒中断发生，芯片才会退出掉电模式。在掉电模式下，时钟控制器关闭外部4~24MHz晶振和内部22.1184MHz高速RC振荡器，以降低整个系统功耗。下图展示了时钟发生器和时钟源控制的概述。

时钟发生器由如下4个时钟源组成：

- 外部4~24 MHz高速晶振(HXT)
- 可编程的PLL输出时钟频率(PLL 由外部 4~24 MHz 晶振或内部 22.1184 MHz 振荡器提供时钟源)
- 内部22.1184 MHz高速振荡器(HIRC)
- 内部10 kHz低速RC振荡器(LIRC)

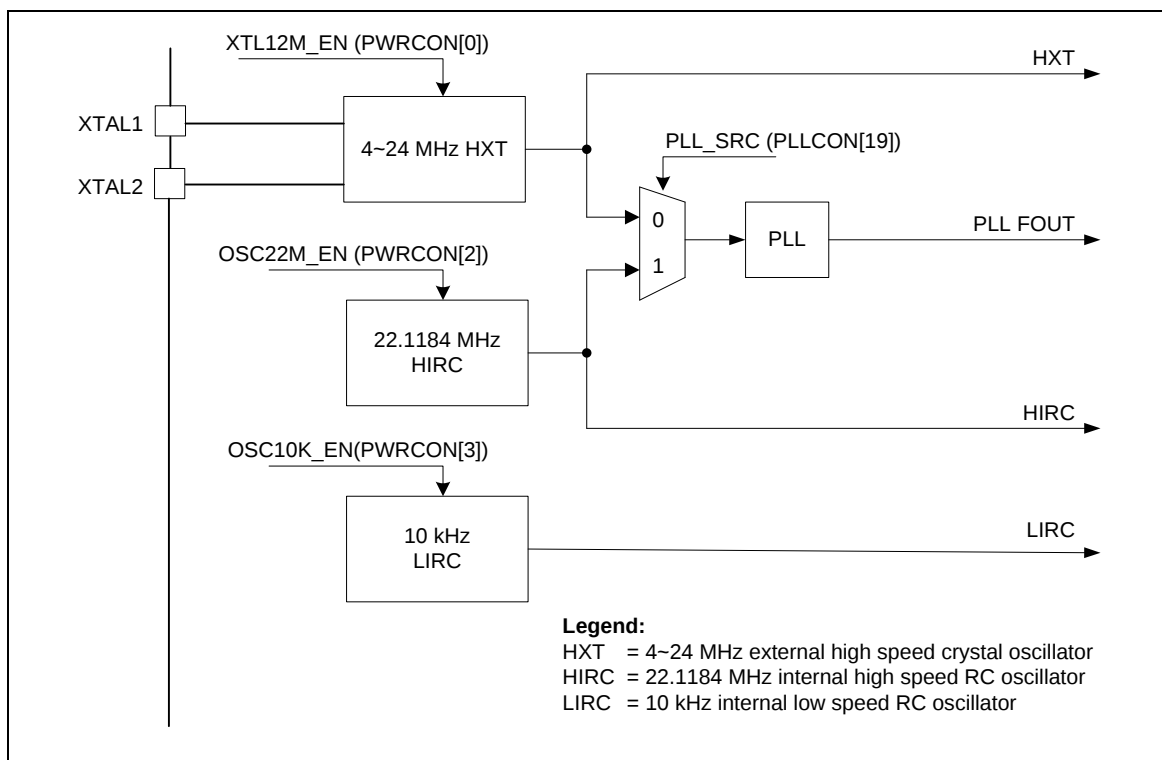


图 6.3-1 时钟发生器框图

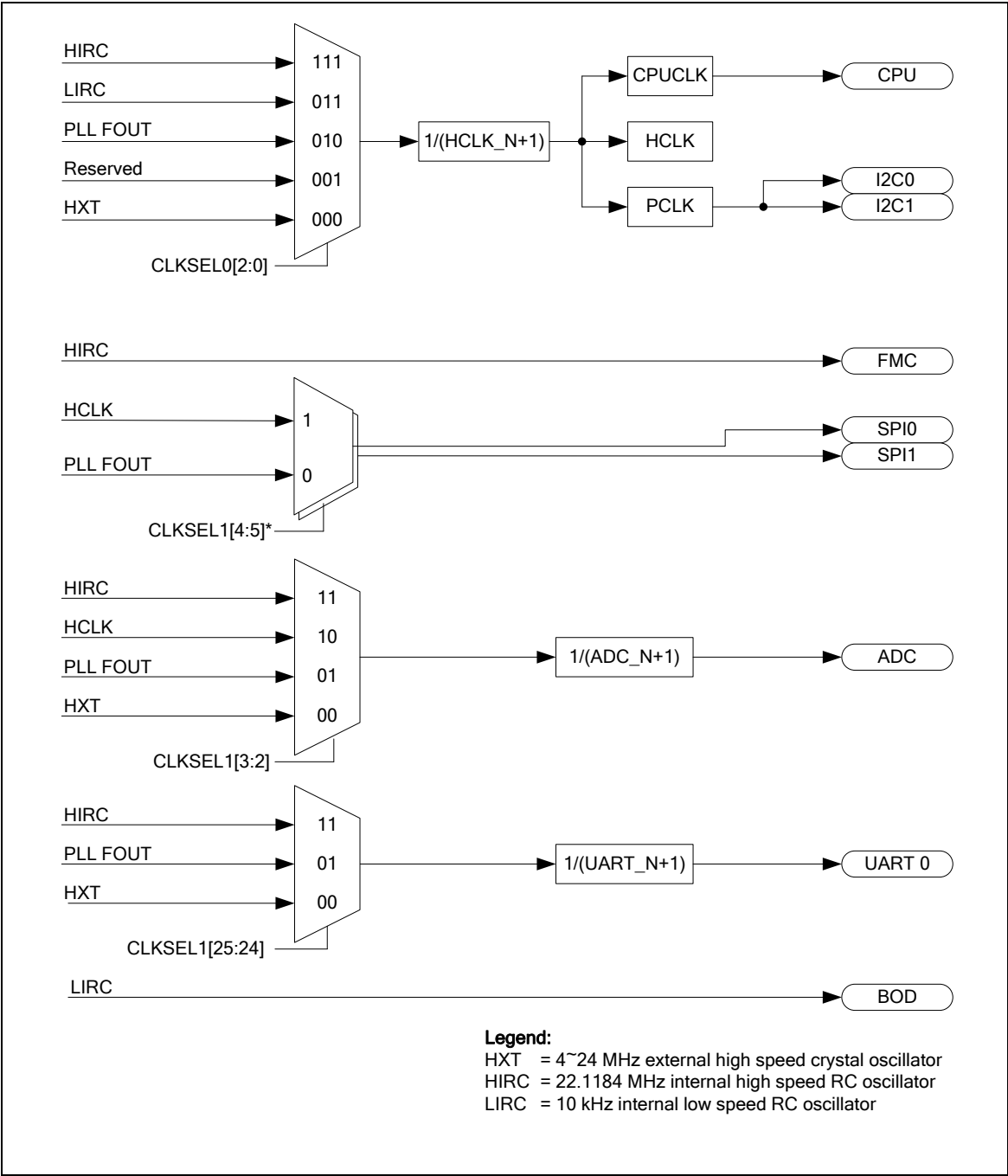


图 6.3-2 时钟源控制器概览(1/2)

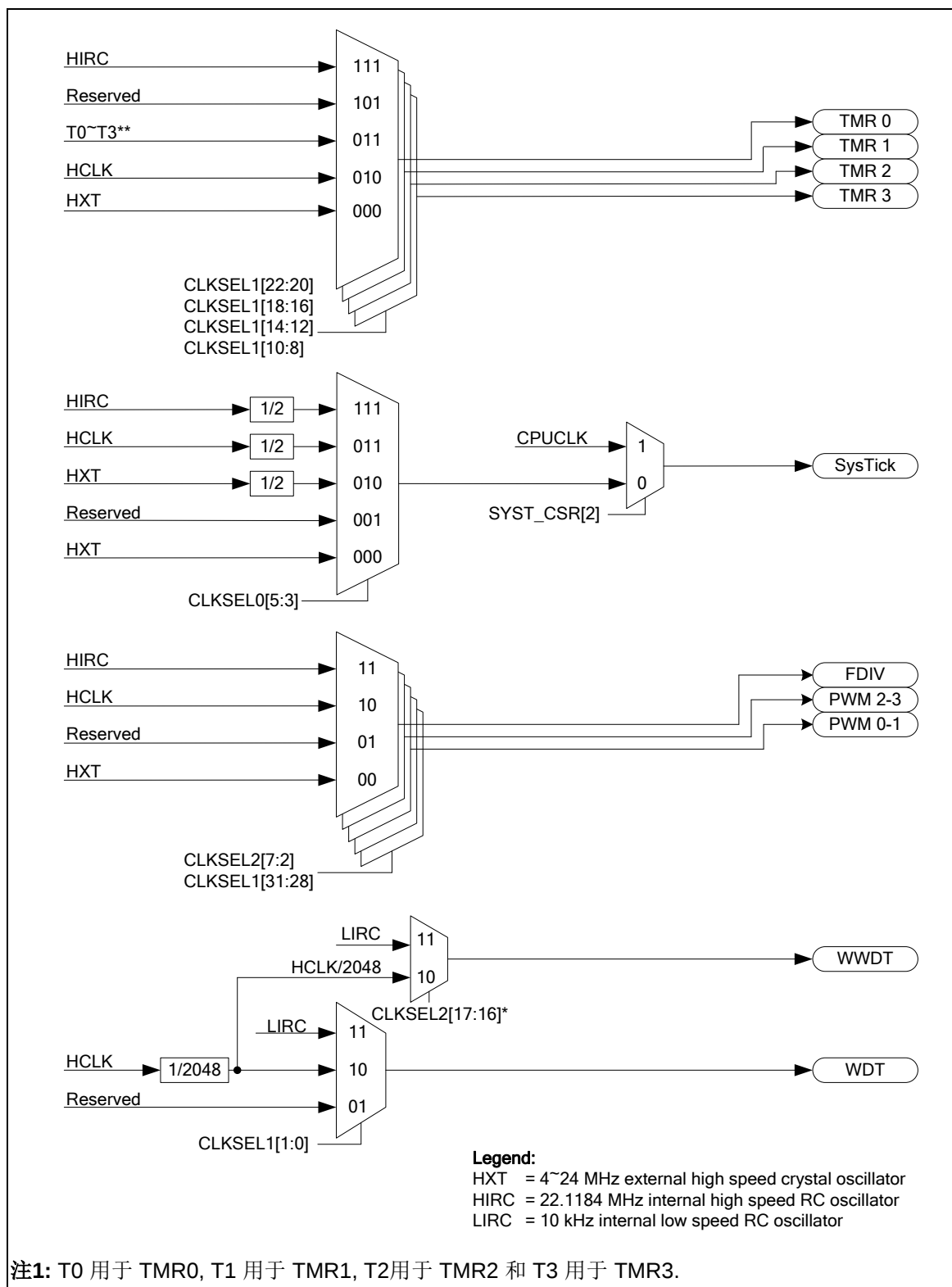


图 6.3-3 时钟源控制器概览(2/2)

6.3.2 系统时钟和 SysTick 时钟

系统时钟有 4 个时钟源，由时钟发生器发生。时钟源切换取决于寄存器 HCLK_S (CLKSEL0[2:0])。框图如下所示：

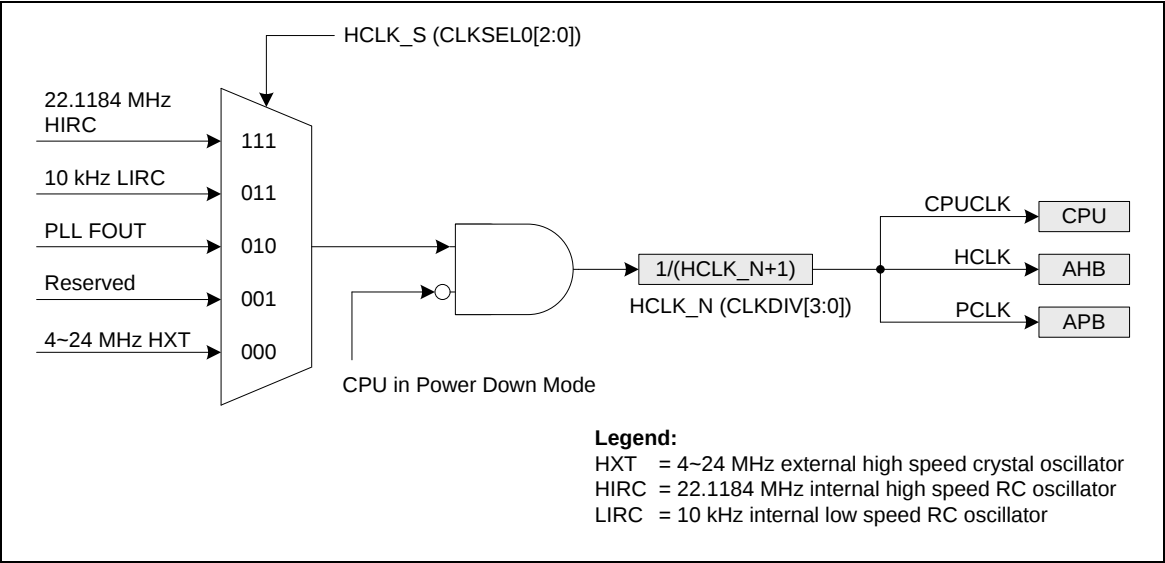


图 6.3-4 系统时钟框图

Cortex™-M0内核的SysTick 时钟源可以选择CPU时钟或外部时钟(SYST_CSR[2])。如果使用外部时钟， SysTick 时钟 (STCLK) 有 4 个时钟源。时钟源切换取决于寄存器 STCLK_S (CLKSEL0[5:3])。框图如下所示：

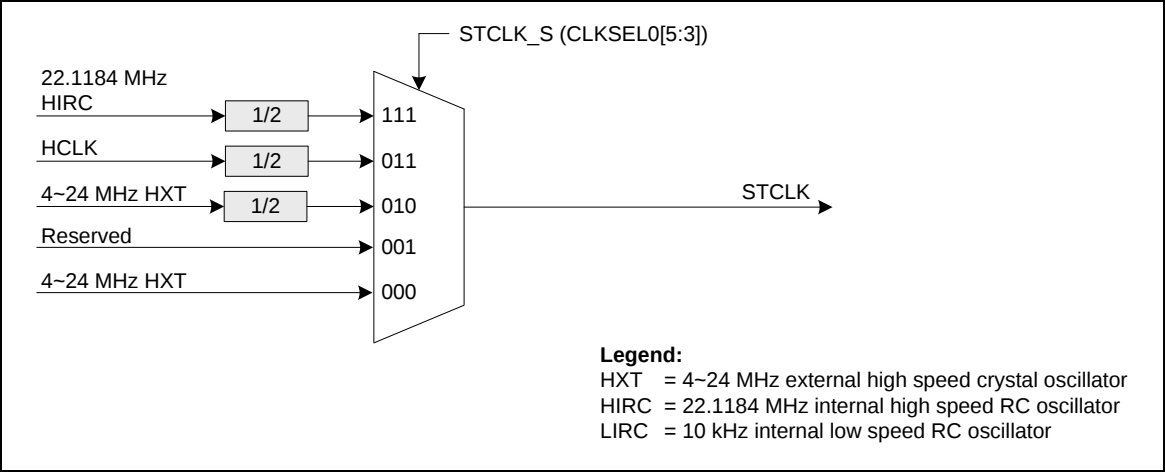


图 6.3-5 SysTick 时钟控制框图

6.3.3 掉电模式时钟

当芯片进入掉电模式，系统时钟，一些时钟源和一些外设时钟将被关闭。也有一些时钟源和外设时钟仍在工作。

下列时钟仍在工作：

- 时钟发生器
- 10 kHz内部低速振荡器
- 外设时钟（当时钟源来自10 kHz内部低速振荡器）

6.3.4 分频器输出

该设备带有一个2的幂次分频器，该分频器由16个链式二分频器的移位寄存器组成。由16选1的多路转换器选择16个移位寄存器其中一个输出并反映到CKO管脚上。因此共有2的16次幂的时钟分频选择，分频范围从 $F_{in}/2^1$ 到 $F_{in}/2^{16}$ ，此处 F_{in} 是到时钟分频器的时钟输入频率。

输出公式： $F_{out} = F_{in}/2^{(N+1)}$ ，其中 F_{in} 为输入时钟频率， F_{out} 为时钟分频器输出频率，N 为 FSEL(FRQDIV[3:0])中的4位值。

往DIVIDER_EN (FRQDIV[4])写1，链式计数器开始计数。往DIVIDER_EN (FRQDIV[4])写0，链式计数器直到分频时钟到达低态并保持低态时停止。

如果DIVIDER1(FRQDIV[5])设置为1，分频器时钟(FRQDIV_CLK)将忽略2的幂次分频器。分频器时钟将直接输出到CKO。

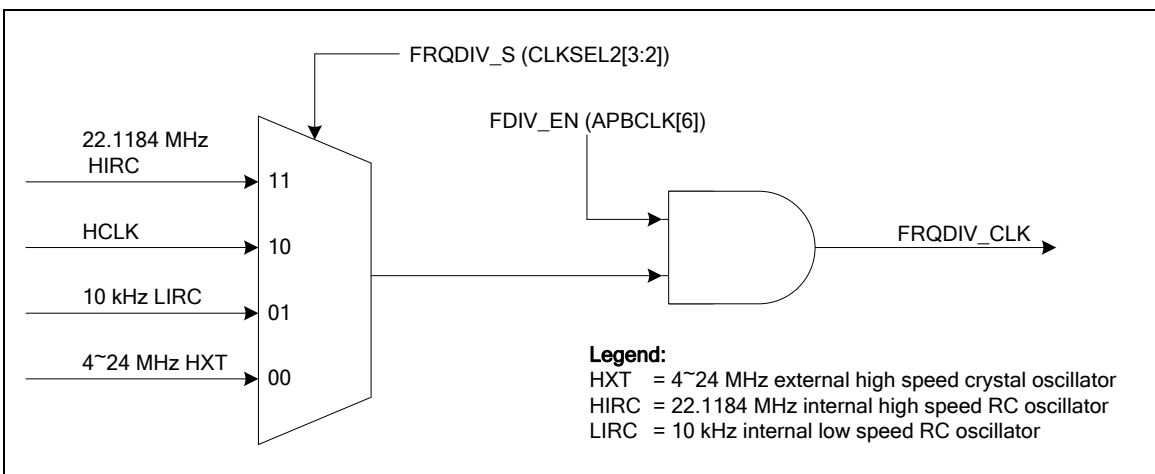


图 6.3-6 分频器的时钟源

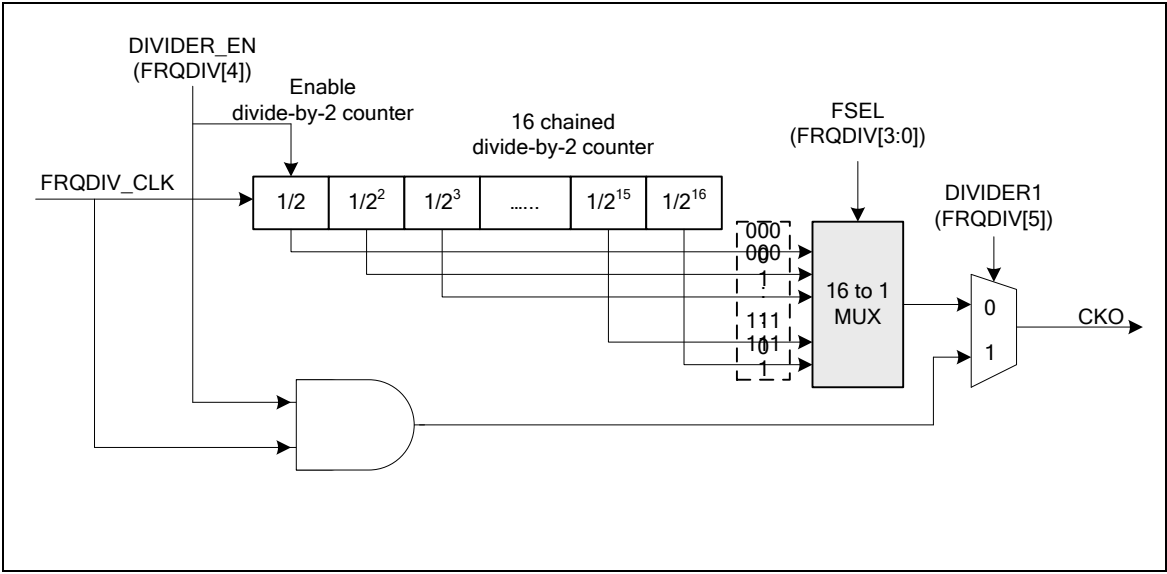


图 6.3-7 分频器框图

6.3.5 寄存器映射

R:只读, W: 只写, R/W:读/写

寄存器	偏移量	R/W	描述	复位值
CLK 基地址: CLK_BA = 0x5000_0200				
PWRCON	CLK_BA+0x00	R/W	系统掉电控制寄存器	0x0000_001X
AHBCLK	CLK_BA+0x04	R/W	AHB设备时钟使能控制寄存器	0x0000_000D
APBCLK	CLK_BA+0x08	R/W	APB设备时钟使能控制寄存器	0x0000_000X
CLKSTATUS	CLK_BA+0x0C	R/W	时钟状态监控寄存器	0x0000_00XX
CLKSEL0	CLK_BA+0x10	R/W	时钟源选择控制寄存器 0	0x0000_003X
CLKSEL1	CLK_BA+0x14	R/W	时钟源选择控制寄存器 1	0xFFFF_FFFF
CLKDIV	CLK_BA+0x18	R/W	时钟分频数目寄存器	0x0000_0000
CLKSEL2	CLK_BA+0x1C	R/W	时钟源选择控制寄存器2	0x0000_00FF
PLLCON	CLK_BA+0x20	R/W	PLL 控制寄存器	0x0005_C22E
FRQDIV	CLK_BA+0x24	R/W	分频器控制寄存器	0x0000_0000

6.3.6 寄存器描述

掉电控制寄存器 (PWRCON)

除 BIT[6]，PWRCON 的其他位都受保护，需要向地址 0x5000_0100 依次写入 “59h”，“16h”，“88h” 来解锁这些位。请参考寄存器 REGWRPROT （地址为 GCR_BA+0x100）

寄存器	偏移量	R/W	描述	复位值
PWRCON	CLK_BA+0x00	R/W	系统掉电控制寄存器	0x0000_001X

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							PD_WAIT_CPU
7	6	5	4	3	2	1	0
PWR_DOWN_EN	PD_WU_STS	PD_WU_INT_EN	PD_WU_DLY	OSC10K_EN	OSC22M_EN	Reserved	XTL12M_EN

Bit	描述	
[31:9]	Reserved	保留
[8]	PD_WAIT_CPU	进入掉电条件控制 (写保护) 0 = 在 PWR_DOWN_EN 位置 1 时，芯片进入掉电模式。 1 = 在 PD_WAIT_CPU 和 PWR_DOWN_EN 位都置 1 而且 CPU 执行 WFI 指令时，芯片进入掉电模式
[7]	PWR_DOWN_EN	系统掉电模式使能位(写保护) 当该位被置 1，掉电模式使能，掉电操作由PD_WAIT_CPU位来控制。 (a) 如果 PD_WAIT_CPU 为 0，则芯片会在置位 PWR_DOWN_EN 后，立即进入掉电模式。 (b) 如果 PD_WAIT_CPU 为 1，则芯片会一直运行直到 CPU运行WFI指令。 当芯片从掉电模式中被唤醒，该位由硬件清零。用户需要重新设置该位才能使能下一次的掉电。 在掉电模式下，外部 4~24 MHz 高速晶振(HXT)与内部 22.1184 MHz高速振荡器(HIRC)将被禁用，但是内部 10 kHz低速振荡器不受掉电模式的控制。 在掉电模式下，PLL 与系统时钟将被禁用，忽略时钟源选择。如果外设时钟源为内部 10 kHz 低速振荡器，则外设的时钟不受掉电模式的控制。 0 = 执行 WFI 命令，芯片工作于正常模式或者芯片进入 idle 模式 1 = 芯片立即进入掉电模式或者等待 CPU 休眠命令 WFI
[6]	PD_WU_STS	芯片掉电模式唤醒中断状态

		该位由掉电唤醒事件置位，表示从掉电模式恢复。 如果GPIO, UART, WDT或BOD 唤醒发生，该标志置位。 注意：只有在 PD_WU_INT_EN (PWRCON[5]) 被置 1 的时候，该位才工作。写 1 该位清零。
[5]	PD_WU_INT_EN	掉电模式唤醒中断使能位（写保护） 0 = 禁止. 1 = 使能. 注意：当PD_WU_STS 和 PD_WU_INT_EN 都为1时产生中断。
[4]	PD_WU_DLY	使能唤醒延时计数器（写保护） 当芯片从掉电模式中被唤醒时，时钟控制将会延迟一定的时钟周期以等待系统时钟稳定。 当芯片需要工作在外部 4~24 MHz高速晶振(HXT)条件下时，延迟时钟周期为 4096 个时钟周期；当芯片需要工作在内部 22.1184 MHz 高速振荡器(HIRC)条件下时，延迟时钟周期为 256 个时钟周期。 0 = 禁用时钟周期的延迟 1 = 使能时钟周期的延迟
[3]	OSC10K_EN	内部10 KHz低速RC振荡器(LIRC)使能位（写保护） 0 = 禁用10 kHz 内部低速 RC 振荡器(LIRC) 1 = 使能10 kHz 内部低速 RC 振荡器(LIRC)
[2]	OSC22M_EN	22.1184 MHz 内部高速RC 振荡器 (HIRC)使能位 (写保护) 0 = 禁用22.1184 MHz 内部高速RC振荡器(HIRC) 1 = 使能22.1184 MHz 内部高速RC振荡器(HIRC)
[1]	Reserved	保留.
[0]	XTL12M_EN	4~24 MHz 外部高速晶振(HXT)使能位（写保护） 该位的默认值由 flash 控制器的用户配置寄存器CONFIG0 [26:24] 设置。当默认的时钟源为外部 4~24 MHz 高速晶振时，该位自动置 1。 0 = 禁用4 ~ 24 MHz 外部高速晶振(HXT) 1 = 使能4~24 MHz 外部高速晶振(HXT)

寄存器或指令 模式	SLEEPDEEP (SCR[2])	PD_WAIT_CPU (PWRCON[8])	PWR_DOWN_EN (PWRCON[7])	CPU 运行 WFI 指令	禁用时钟
正常运行模式	0	0	0	NO	通过控制寄存器禁用所有时钟
Idle 模式 (CPU进入休眠模式)	0	x	0	YES	仅禁用 CPU 时钟
掉电模式 (CPU 进 入深度休眠模式)	1	1	1	YES	大部分时钟被禁用，仅10 kHz 及选它为时钟源的WDT外设时 钟可运行。

表 6.3-1 掉电模式控制

当芯片进入掉电模式时，用户可以通过一些中断源唤醒芯片。用户必须在设置 PWR_DOWN_EN 位 (PWRCON[7]) 之前，使能相关的中断源及相应的NVIC IRQ 使能位 (NVIC_ISER) 从而保证芯片能进入掉电模式以及能够成功被唤醒。

AHB设备时钟使能控制寄存器(AHBCLK)

该寄存器各个位用于使能/禁用系统时钟，AHB总线设备时钟。

寄存器	偏移量	R/W	描述	复位值
AHBCLK	CLK_BA+0x04	R/W	AHB设备时钟使能控制寄存器	0x0000_000D

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

Bit	描述	
[31:3]	Reserved	保留
[2]	ISP_EN	Flash ISP 控制器时钟使能位 0 = 禁用Flash ISP外围时钟 1 = 使能Flash ISP外围时钟
[1:0]	Reserved	保留。

APB设备时钟使能控制寄存器(APBCLK)

该寄存器各个位用于使能/禁用外设控制器时钟。

寄存器	偏移量	R/W	描述	复位值
APBCLK	CLK_BA+0x08	R/W	APB设备时钟使能控制寄存器	0x0000_000X

31	30	29	28	27	26	25	24
Reserved			ADC_EN	Reserved			
23	22	21	20	19	18	17	16
Reserved		PWM23_EN	PWM01_EN	Reserved			UART0_EN
15	14	13	12	11	10	9	8
Reserved			SPI0_EN	Reserved		I2C1_EN	I2C0_EN
7	6	5	4	3	2	1	0
Reserved	FDIV_EN	TMR3_EN	TMR2_EN	TMR1_EN	TMR0_EN	Reserved	WDT_EN

Bits	描述	
[31:29]	Reserved	保留。
[28]	ADC_EN	ADC 时钟使能控制 0 = 禁用ADC 时钟 1 = 使能ADC 时钟
[27:23]	Reserved	保留。
[21]	PWM23_EN	PWM_23时钟使能控制 0 = 禁止PWM23 时钟 1 = 使能PWM23 时钟
[20]	PWM01_EN	PWM_01时钟使能控制 0 = 禁止PWM01 时钟。 1 = 使能PWM01时钟。
[19:17]	Reserved	保留。
[16]	UART0_EN	UART0时钟使能控制 0 =禁用 UART0 时钟 1 =使能 UART0 时钟
[15:13]	Reserved	保留。
[12]	SPI0_EN	SPI0时钟使能控制

		0 = 禁用SPI0 时钟 1 = 使能SPI0 时钟
[11:10]	Reserved	保留.
[9]	I2C1_EN	I ² C1时钟使能控制 0 = 禁用I ² C1时钟 1 = 使能I ² C1时钟
[8]	I2C0_EN	I ² C0时钟使能控制 0 = 禁用I ² C0时钟 1 = 使能I ² C0时钟
[7]	Reserved	保留.
[6]	FDIV_EN	分频器输出时钟使能控制 0 =禁用 FDIV时钟 1 =使能 FDIV时钟
[5]	TMR3_EN	Timer3 时钟使能控制 0 = 禁用Timer3时钟 1 = 使能Timer3时钟
[4]	TMR2_EN	Timer2 时钟使能控制 0 = 禁用Timer2时钟 1 = 使能Timer2时钟
[3]	TMR1_EN	Timer1 时钟使能控制 0 = 禁用Timer1时钟 1 = 使能Timer1时钟
[2]	TMR0_EN	Timer0 时钟使能控制 0 = 禁用Timer0时钟 1 = 使能Timer0时钟
[1]	Reserved	保留.
[0]	WDT_EN	Watchdog 定时器时钟使能位(写保护) 0 = 禁用Watchdog 时钟 1 = 使能Watchdog 时钟 注意: 该位受保护, 编程该位时, 需要依次向地址0x5000_0100写入“59h”, “16h”, “88h” 该操作参考寄存器REGWRPROT(地址GCR_BA+0x100)

时钟状态寄存器(CLKSTATUS)

该寄存器各个位用于监控芯片时钟源是否稳定，时钟切换是否失败

寄存器	偏移量	R/W	描述	复位值
CLKSTATUS	CLK_BA+0x0C	R/W	时钟状态监控寄存器	0x0000_00XX

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CLK_SW_FAIL	Reserved		OSC22M_STB	OSC10K_STB	PLL_STB	Reserved	XTL12M_STB

Bit	描述	
[31:8]	Reserved	保留.
[7]	CLK_SW_FAIL	时钟切换失败标志 0 = 时钟切换成功 1 = 时钟切换失败 注意1: 当软件切换系统时钟源，该位更新。如果切换的目标时钟源稳定该位置0，如果切换的目标时钟源不稳定该位置1 注意2: 该位只读。选择的时钟源稳定后硬件自动将系统时钟切换为所选的时钟源，并由硬件自动清除CLK_SE_FAIL。
[6:5]	Reserved	保留.
[4]	OSC22M_STB	内部22.1184 MHz高速RC振荡器(HIRC)时钟源稳定标志（只读） 0 = 22.1184 MHz 内部高速RC 振荡器(HIRC) 时钟不稳定或者禁用 1 = 22.1184 MHz 内部高速RC 振荡器(HIRC) 时钟使能并稳定
[3]	OSC10K_STB	内部10 KHz 低速振荡器 (LIRC) 时钟源稳定标志（只读） 0 = 内部10 kHz低速RC振荡器 (LIRC) 时钟不稳定或者禁用 1 = 内部10 kHz低速RC振荡器(LIRC) 时钟使能并稳定
[2]	PLL_STB	内部 PLL 时钟源稳定标志(只读) 0 = 内部 PLL时钟不稳定或者禁用。 1 = 内部PLL 时钟稳定。

[1]	Reserved	保留.
[0]	XTL12M_STB	外部4~24 MHz 高速晶振 (HXT) 时钟源稳定标志 (只读) 0 = 外部4~24 MHz高速晶振(HXT) 时钟不稳定或者禁用 1 = 外部4~24 MHz高速晶振(HXT) 时钟使能并稳定

时钟源选择控制寄存器0 (CLKSEL0)

寄存器	偏移量	R/W	描述	复位值
CLKSEL0	CLK_BA+0x10	R/W	时钟源选择控制寄存器0	0x0000_003X

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

Bit	描述	
[31:6]	Reserved	保留.
[5:3]	STCLK_S	Cortex®-M0 SysTick时钟源选择(写保护) 如果SYST_CSR[2] = 1, SysTick 时钟源来自 HCLK 如果SYST_CSR[2] = 0, SysTick 时钟源由STCLK_S(CLKSEL0[5:3])定义. 000 = 时钟源为外部 4~24 MHz 高速晶振时钟 001 = 保留 010 = 时钟源为外部 4~24 MHz高速 晶振时钟/2分频 011 = 时钟源为 HCLK/2分频 111 = 时钟源为内部 22.1184 MHz 高速振荡器时钟/2分频 其他=保留 注意1: 该位是受保护位。编程时, 需要向地址 0x5000_0100 依次写入 “59h”, “16h”, “88h” 来解锁寄存器保护。请参考寄存器 REGWRPROT (地址: GCR_BA+0x100)。 注意2: 如果 SysTick 时钟源不是来自 HCLK (例如 SYST_CSR[2] = 0), SysTick 时钟源必须小于或等于HCLK/2
[2:0]	HCLK_S	HCLK时钟源选择(写保护) 000 = 时钟源为HXT. 001 = 保留. 010 = 时钟源为PLL. 011 = 时钟源为 LIRC. 111 = 时钟源为 HIRC.

		其他 = 保留。 注意1: 在时钟切换前，相关时钟源（预选和新选）必须打开并稳定。 注意2: 在任何复位后，这 3 位的默认值将从Flash 控制器的用户配置寄存器 CFOSC (CONFIG0 [26:24]) 加载。因此默认值可能为 000b 或者 111b。 注意3: 该位是受保护位。编程时，需要向地址 0x5000_0100 依次写入 “59h”, “16h”, “88h” 来解锁寄存器保护。请参考寄存器 REGWRPROT （地址：GCR_BA+0x100）。
--	--	--

时钟源选择控制寄存器 1 (CLKSEL1)

在时钟切换之前，必须打开相关的时钟源（预选和新选）。

寄存器	偏移量	R/W	描述	复位值
CLKSEL1	CLK_BA+0x14	R/W	时钟源选择控制寄存器1	0xFFFF_FFFF

31	30	29	28	27	26	25	24
PWM23_S		PWM01_S		Reserved		UART_S	
23	22	21	20	19	18	17	16
Reserved		TMR3_S		Reserved		TMR2_S	
15	14	13	12	11	10	9	8
Reserved		TMR1_S		Reserved		TMR0_S	
7	6	5	4	3	2	1	0
Reserved			SPI0_S	ADC_S		WDT_S	

Bit	描述	
[31:30]	PWM23_S	PWM2 和PWM3时钟源选择 PWM2和PWM3 共享同一个引擎时钟源，并共享同一个预分频器。 00 = 时钟源为HXT. 01 = 时钟源为LIRC. 10 = 时钟源为HCLK. 11 = 时钟源为HIRC.
[29:28]	PWM01_S	PWM0 和PWM1时钟源选择 PWM0和PWM1 共享同一个引擎时钟源，并共享同一个预分频器。 00 = 时钟源为HXT. 01 = 时钟源为LIRC. 10 = 时钟源为HCLK. 11 = 时钟源为HIRC.
[27:26]	Reserved	保留.
[25:24]	UART_S	UART 时钟源选择 00 = 时钟源为HXT. 01 = 时钟源为PLL.

		10 = 保留. 11 = 时钟源为HIRC.
[23]	Reserved	保留
[22:20]	TMR3_S	TIMER3 时钟源选择 000 = 时钟源为HXT. 010 = 时钟源为HCLK. 011 = 时钟源为外部触发器 T3. 101 = 时钟源为LIRC. 111 = 时钟源为HIRC. 其他= 保留.
[19]	Reserved	保留.
[18:16]	TMR2_S	TIMER2时钟源选择 000 = 时钟源为HXT. 010 = 时钟源为HCLK. 011 = 时钟源为外部触发器T2. 101 = 时钟源为LIRC. 111 = 时钟源为HIRC. 其他 = 保留.
[15]	Reserved	保留.
[14:12]	TMR1_S	TIMER1 时钟源选择 000 = 时钟源为HXT. 010 = 时钟源为HCLK. 011 = 时钟源为外部触发器 T1. 101 = 时钟源为LIRC. 111 = 时钟源为HIRC. 其他 = 保留.
[11]	Reserved	保留.
[10:8]	TMR0_S	TIMER0 时钟源选择 000 = 时钟源为HXT. 010 = 时钟源为HCLK. 011 = 时钟源为外部触发器 T0. 101 = 时钟源为LIRC. 111 = 时钟源为HIRC. 其他 = 保留.
[7:5]	Reserved	保留.

[4]	SPI0_S	SPI0时钟源选择 0 = 时钟源为PLL. 1 = 时钟源为HCLK.
[3:2]	ADC_S	ADC外围时钟源选择 00 = 时钟源为HXT. 01 = 时钟源为PLL. 10 = 时钟源为HCLK. 11 = 时钟源为HIRC.
[1:0]	WDT_S	Watchdog 时钟源选择 (写保护) 00 = 保留. 01 = 保留. 10 = 时钟源为HCLK/2048 时钟. 11 = 时钟源为LIRC. 注意: 该位是受保护位。编程时，需要向地址 0x5000_0100 依次写入 “59h”, “16h”, “88h” 来解锁寄存器保护。请参考寄存器 REGWRPROT （地址：GCR_BA+0x100）。

时钟源选择控制寄存器2 (CLKSEL2)

在时钟切换之前，必须打开相关的时钟源（预选和新选）。

寄存器	偏移量	R/W	描述	复位值
CLKSEL2	CLK_BA+0x1C	R/W	时钟源选择控制寄存器 2	0x0000_00FF

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

Bit	描述	
[31:18]	Reserved	保留。
[17:16]	WWDT_S	窗口看门狗时钟源选择 10 = 时钟源为HCLK/2048 时钟。 11 = 时钟源为LIRC。
[15:4]	Reserved	保留。
[3:2]	FRQDIV_S	时钟分频器时钟源选择 00 = 时钟源为HXT。 01 = 保留 10 = 时钟源为HCLK。 11 = 时钟源为HIRC。
[1:0]	Reserved	保留。

时钟分频寄存器(CLKDIV)

寄存器	偏移量	R/W	描述	复位值
CLKDIV	CLK_BA+0x18	R/W	时钟分频数寄存器	0x0000_0000

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

Bit	描述	
[31:24]	Reserved	保留
[23:16]	ADC_N	ADC 外围时钟源的时钟除频数 $\text{ADC 外围时钟频率} = (\text{ADC 时钟源频率}) / (\text{ADC_N} + 1)$
[15:12]	Reserved	保留
[11:8]	UART_N	UART 时钟源的时钟除频数 $\text{UART 时钟频率} = (\text{UART 时钟源频率}) / (\text{UART_N} + 1)$
[7:4]	Reserved	保留
[3:0]	HCLK_N	HCLK 时钟源的时钟除频数 $\text{HCLK 时钟频率} = (\text{HCLK 时钟源频率}) / (\text{HCLK_N} + 1)$

PLL控制寄存器 (PLLCON)

PLL 的参考时钟源来自外部 4~24 MHz 高速晶振时钟(HXT)或者内部 22.1184 MHz 高速振荡器(HIRC)。该寄存器用于控制 PLL 的输出频率和 PLL 的工作模式。

寄存器	偏移量	R/W	描述	复位值
PLLCON	CLK_BA+0x20	R/W	PLL 控制寄存器	0x0005_C22E

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

Bit	描述	
[19]	PLL_SRC	PLL时钟源选择 0 = PLL 时钟源为 HXT. 1 = PLL时钟源为HIRC.
[18]	OE	PLL OE (FOUT 使能) 引脚控制 0 = PLL FOUT 使能. 1 = PLL FOUT 固定为低.
[17]	BP	PLL旁路控制 0 = PLL 正常模式 (默认) 1 = PLL 时钟输出与PLL源时钟输入相同
[16]	PD	掉电模式 如果设置寄存器 PWRCON 的 PWR_DOWN_EN 位为 1, PLL 也将进入掉电模式。 0 = PLL 正常模式 1 = PLL 进入掉电模式 (默认)
[15:14]	OUT_DV	PLL 输出分频控制位 参考下表公式

[13:9]	IN_DV	PLL 输入分频控制位 参考下表公式
[8:0]	FB_DV	PLL 反馈分频控制位 参考下表公式

PLL输出时钟频率设置:

$$F_{OUT} = F_{IN} \times \frac{NF}{NR} \times \frac{1}{NO}$$

约束条件:

1. 4MHz < F_{IN} < 24MHz
2. 800KHz < $\frac{F_{IN}}{2 * NR}$ < 7.5MHz
3. $100MHz < F_{CO} = F_{IN} \times \frac{NF}{NR} < 200MHz$
120MHz < F_{CO} is preferred

符号	描述
F_{OUT}	输出时钟频率
F_{IN}	输入（参考）时钟频率
NR	输入分频 ($IN_DV + 2$)
NF	反馈分频 ($FB_DV + 2$)
NO	OUT_DV = "00" : NO = 1 OUT_DV = "01" : NO = 2 OUT_DV = "10" : NO = 2 OUT_DV = "11" : NO = 4

默认频率设置

默认值: 0xC22E
 F_{IN} = 12 MHz
 $NR = (1+2) = 3$
 $NF = (46+2) = 48$
 $NO = 4$
 $F_{OUT} = 12/4 \times 48 \times 1/3 = 48$ MHz

频率分频器控制寄存器(FRQDIV)

寄存器	偏移量	R/W	描述	复位值
FRQDIV	CLK_BA+0x24	R/W	频率分频控制寄存器	0x0000_0000

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

Bit	描述	
[31:5]	Reserved	保留.
[4]	DIVIDER_EN	频率分频器使能控制 0 = 禁用频率分频器 1 = 使能频率分频器
[3:0]	FSEL	分频器输出频率选择位 输出频率的公式: $F_{out} = F_{in}/2^{(N+1)}$. F_{in} 为输入时钟频率 F_{out} 为分频器输出时钟频率 N为FSEL[3:0] 的4位值

6.4 Flash 存储控制器(FMC)

6.4.1 概述

M058S系列 具有64/32/16/8 KB 字节的片上FLASH，用于存储应用程序（APROM），用户可以通过ISP更新这些FLASH。在系统编程（ISP）和在应用编程(IAP) 可以让用户可以直接更新已经焊接在PCB板上芯片的程序。上电后，Config0的启动选择(CBS)决定Cortex-M0 CPU从APROM或LDROM读取代码。M0M058S 额外提供了4KB的数据闪存来为用户在关机前存储64/32/16/8 KB APROM里的应用程序所依赖的数据。

M058S系列的CONFIG0提供更多的高级功能，包括带三态I/O启动，启动后默认使能WDT,在掉电模式使能WDT,和IAP功能。M058S增加的功能如下表所示：

	M058S 系列
CONFIG[6]	支持IAP功能和多启动功能
CONFIG[10]	选择启动后I/O状态
CONFIG[30]	当启动后默认开启WDT，支持掉电模式WDT
CONFIG[31]	支持启动后默认开启WDT.

表 6.4-1 M058S 系列不同功能列表 (FMC)

6.4.2 特性

- 连续地址读访问零等待状态时，最高可达50 MHz
- 32 KB应用程序存储空间(APROM)
- 4KB在系统编程 (ISP) 加载程序空间(LDROM)
- 固定 4 KB 数据 Flash
- 所有嵌入Flash支持512字节页擦除
- 支持在系统编程(ISP)和在应用编程(IAP)来更新片上Flash EPROM

6.4.3 框图

FLASH存储器控制器包括AHB从接口，ISP 控制逻辑，烧写器接口和FLASH宏接口时序控制逻辑。FLASH存储器控制器框图如下图所示：

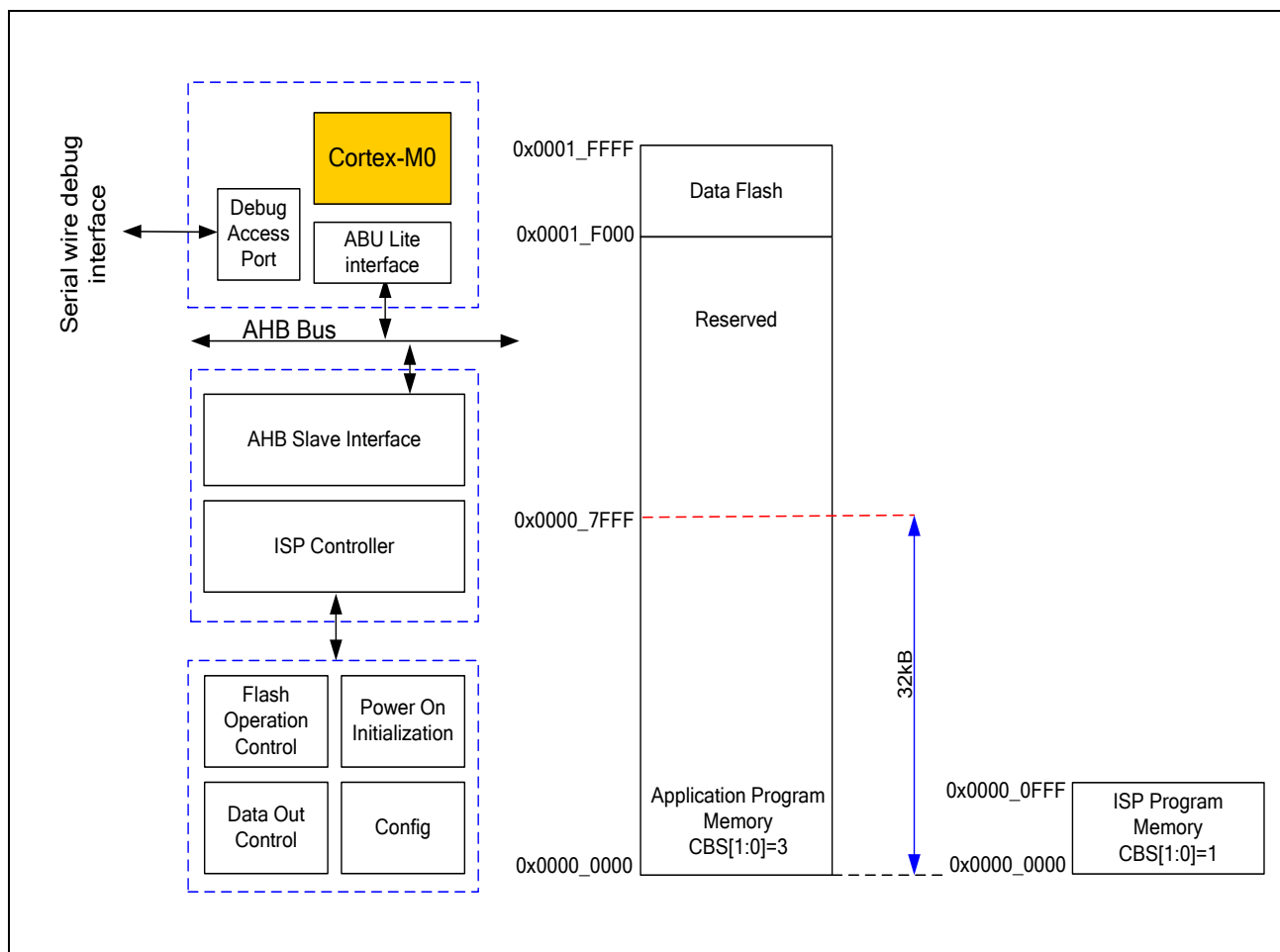


图 6.4-1 Flash 存储器控制框图

6.4.4 FMC 存储器组织

M058S系列flash存储器由程序存储器(APROM)，数据FLASH，ISP载入程序存储器(LDROM)，和用户配置区组成。

程序存储器是用户应用的主要存储器，叫做APROM，用户可以将他们的应用程序写到APROM并设置系统从APROM启动。

设计ISP 装载程序存储器是用来实现在系统编程（ISP）功能。LDROM独立于APROM，也能设置从LDROM启动，因此当APROM的代码损坏时用户可以用LDROM启动系统，避免系统启动失败。

数据FLASH是给用户存储数据的，它可以通过ISP读取或存储器读取，并且通过ISP编程。每个擦除单元是512字节。数据flash的大小固定4K，起始地址为：0x0001_F000。

用户配置提供了几个字节来控制系统逻辑，如FLASH安全锁，启动选择，欠压电平，数据FLASH基地址等。用户配置如同是上电设置的保险，上电时用户配置从FLASH存储器装载到它相应的控制寄存器。

在NuMicro™家族，flash存储器组织跟系统存储映射不同。当用户用ISP命令去读、编程、或者擦除flash，存储器组织被用到。在CPU访问FLASH存储器获取代码或数据的时候，系统存储器组织被用到。例如，当系统设置为从LDROM启动（CBS = 01b），CPU将从LDROM的0x0 ~ 0x0FFF取代码。但是，如果用户想用ISP读LDROM，仍然要从LDROM的地址0x0010_0000 ~ 0x0010_0FFF去读。

下列表和图显示了APROM, LDROM，数据 Flash和用户配置的地址映像信息。

模块名称	规格	起始地址	结束地址
AP-ROM	32 KB	0x0000_0000	0x0000_7FFF (32 KB)
数据 Flash	4 KB	0x0001_F000	0x0001_FFFF
LD-ROM	4 KB	0x0010_0000	0x0010_0FFF
用户配置	1 Words	0x0030_0000	0x0030_0000

表 6-18 Flash 存储器地址映像

Flash存储器组织结构如下所示:

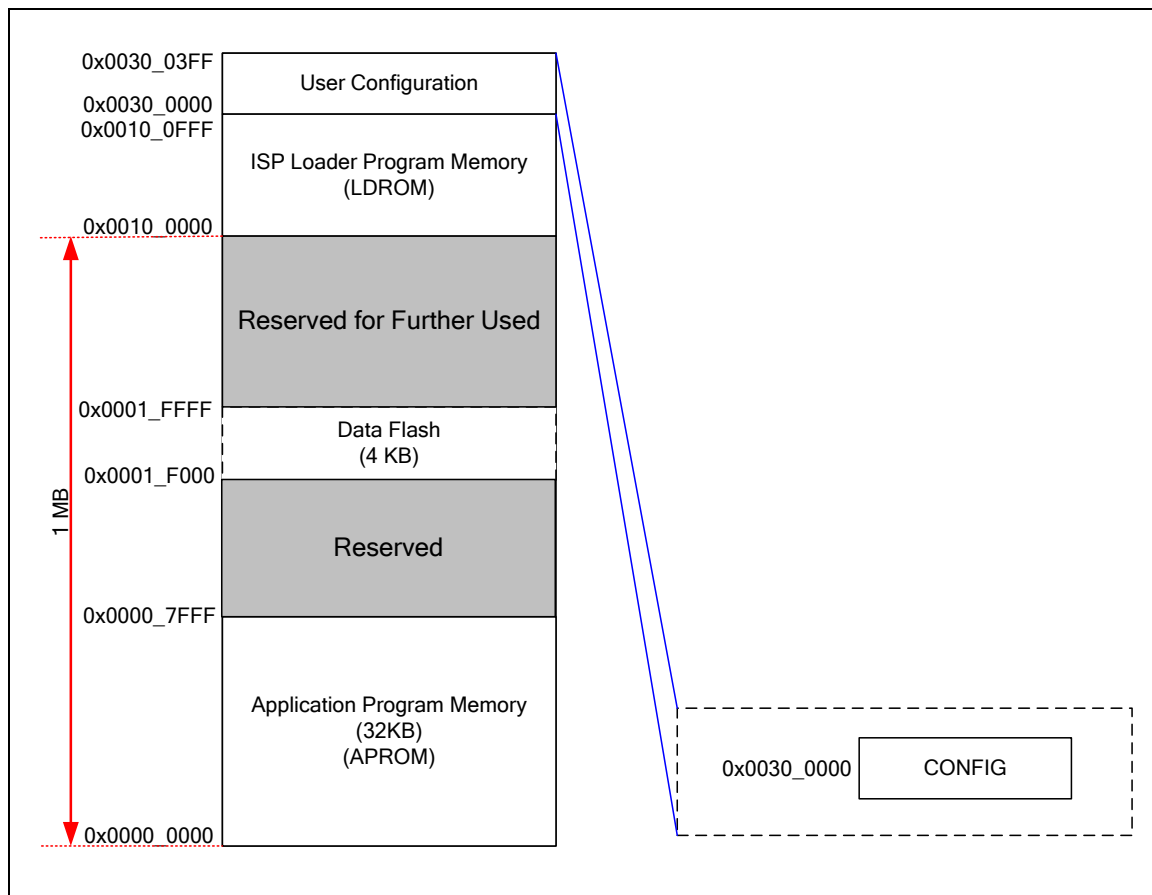


图 6.4-2 Flash 存储器组织结构

6.4.5 数据 Flash

M058S系列提供了数据Flash用于用户存储数据。它通过ISP进行读/写。擦除单元是512 bytes。当一个字要更改，所有128字都需要预先拷贝到另一页或者到SRAM。对于8/16/32/64 KB Flash 存储设备，数据Flash大小是4KB，并且起始地址固定在0x0001_F000。

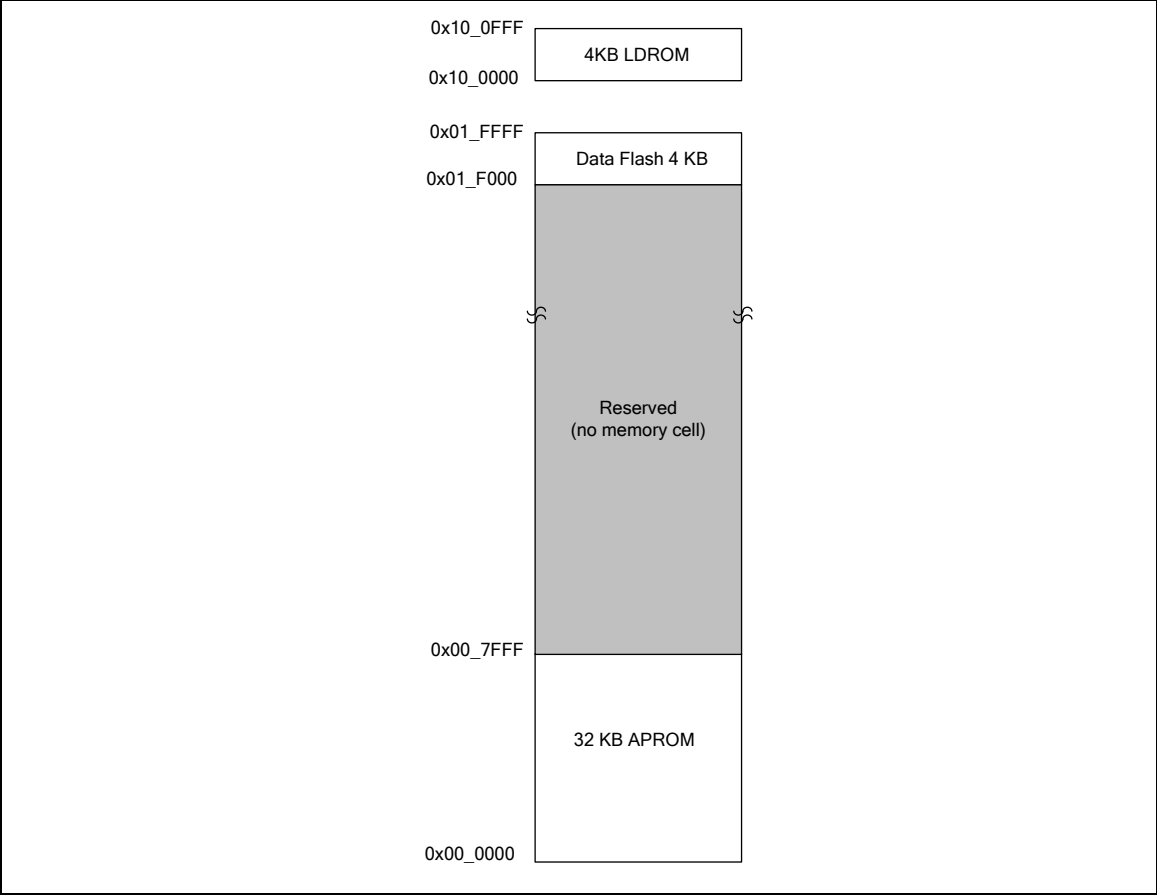


图 6.4-3 Flash 存储器结构

6.4.6 用户配置

6.4.6.1 启动选择

用户配置是内部可编程的配置区域，用于启动选项。用户配置位于Flash存储器组织的0x300000，并有两个32位字节。用户配置所有的更改将在系统重启后生效。

CONFIG0 (地址 = 0x0030_0000)

31	30	29	28	27	26	25	24
CWDTEN	CWDTPDEN	Reserved		CFGXT1	CFOSC		
23	22	21	20	19	18	17	16
CBODEN	CBOV		CBORST	Reserved			
15	14	13	12	11	10	9	8
Reserved					CIOINI	Reserved	
7	6	5	4	3	2	1	0
CBS		Reserved				LOCK	Reserved

BIT	描述									
[31]	CWDTEN	看门狗使能控制 0 =看门狗时钟使能并且芯片上电后看门狗时钟源强制为 OSC10K 1 =芯片上电后禁止看门狗时钟								
[30]	CWDTPDEN	看门狗时钟掉电使能控制 0 = OSC10K 看门狗定时器时钟源一直强制使能 1 = OSC10K 看门狗定时器时钟源在掉电模式下由OSC10K_EN (PWRCON[3]) 控制 注: 此位只有在CWDTEN 为0时有效.								
[29:28]	Reserved	保留.								
[27]	CFGXT1	P7[1:0] 复用功能选择 0 = P7.0 和 P7.1 配置为 GPIO 引脚 1 = P7.0 和 P7.1 配置为外部 4~24MHz (高速) 晶振引脚								
[26:24]	CFOSC	复位后CPU时钟源选择 <table><tr><th>CFOSC[2:0]</th><th>时钟源</th></tr><tr><td>000</td><td>外部高速时钟 (4 ~ 24 MHz)</td></tr><tr><td>111</td><td>内部 22.1184 MHz 高速时钟</td></tr><tr><td>其他</td><td>保留</td></tr></table> 复位后, CFOSC的值将被加载到系统寄存器CLKSEL0.HCLK_S[2:0].	CFOSC[2:0]	时钟源	000	外部高速时钟 (4 ~ 24 MHz)	111	内部 22.1184 MHz 高速时钟	其他	保留
CFOSC[2:0]	时钟源									
000	外部高速时钟 (4 ~ 24 MHz)									
111	内部 22.1184 MHz 高速时钟									
其他	保留									
[23]	CBODEN	欠压检测使能 0= 上电后, 使能欠压检测 1= 上电后, 禁用欠压检测								

[22:21]	CBOV	欠压电压选择	
		CBOV	欠压电压
		11	4.4V
		10	3.7V
		01	2.7V
		00	2.2V
[20]	CBORST	欠压复位使能 0 = 上电后, 使能欠压复位 1 = 上电后, 禁用欠压复位	
[19:11]	Reserved	保留.	
[10]	CIOINI	I/O初始状态选择 0= 上电后所有GPIO默认为三态模式 1=上电后所有GPIO默认为准双向模式	
[9:8]	Reserved	保留.	
[7:6]	CBS	芯片启动选择 00=由LDROM启动带IAP功能 01=由LDROM启动不带IAP功能 10=由APROM启动带IAP功能 11=由APROM启动不带IAP功能 用户可以设 CBS[0] = 0 来支持 IAP 功能。当 CBS[0] = 0, LDROM 映像到地址 0x100000, APROM 映像到地址 0x0。用户可以通过他们的地址直接访问不需要启动切换。换言之, 在支持 IAP 的情况下, 在 LDROM 和 APROM 里面的代码可以相互调用。 注 1: 当 CBS[0] = 1, ISPCON 的 BS 位只能用于控制启动切换。 注2: 当CBS[0] = 0, VECMAP只能用于重映射到0x0~0x1ff。	
[5:2]	Reserved	保留.	
[1]	LOCK	安全加密控制 0 = 加密Flash数据 1 = 解除Flash 数据加密 当FLASH数据被加密, 仅有设备ID, Config0 和Config1 可被烧写器和ICP通过串口调试接口读取. 其他数据锁定为0xFFFFFFFF. 无论数据是否锁定, ISP 都可以读取 对于加密系统, 用户可以用 ISP 命令来关闭 LOCK 位, 或者用 ICP 芯片擦除命令来擦除整个芯片	
[0]	Reserved	保留.	

注:用户配置的保留位应当一直保持为‘1’。

6.4.6.2 欠压检测

M058S系列包含欠压检测功能用来监控 V_{DD} 管脚上的电压。当BOD使能时，如果 V_{DD} 电压下降到低于CBOV设置的电压，BOD事件将被触发。当BOD被检测到时用户可以决定通过使能CBORST来BOD复位或者只是由NVIC使能BOD中断。因为无论什么时候 V_{DD} 电压低于设置的CBOV，BOD复位就会被启用，所以用户必须确定CBOV设置来避免BOD重复复位。例如， $V_{DD}=3.3V$ ，CBOV只能设置00'b 或 01'b。否则，当BOD被使能CBOV是10'b 或11'b时系统将停留在BOD复位状态。

6.4.7 启动选择

M058S系列提供在系统编程(ISP)功能，支持芯片在PCB板上直接更新程序存储器。提供4KB加载程序内存用于存储ISP固件。用户可以通过设置Config0中的CBS位来选择从APROM还是从LDROM取指令来运行。

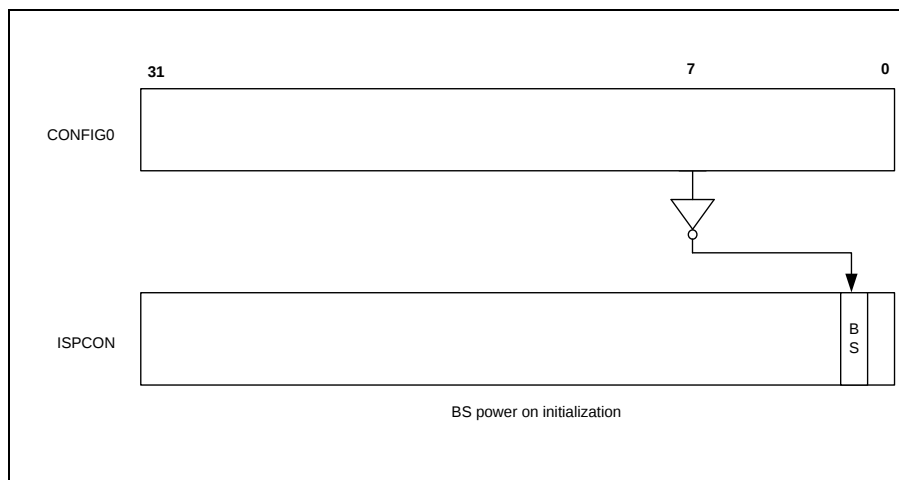


图 6.4-4 启动选择 (BS)用于上电动作

Config0中的CBS除了设置从APROM或者LDROM启动之外还用于启动后控制系统存储器映射。当CBS[0] = 1 和 CBS[1] = 1为从APROM启动，APROM的应用程序不能通过读储存器的方式来访问LDROM.同样当CBS[0] = 1 和 CBS[1] = 0为LDROM启动，LDROM的程序也不能通过读储存器的方式来访问APROM。下图表示当系统从APROM和LDROM启动时的储存器映射。

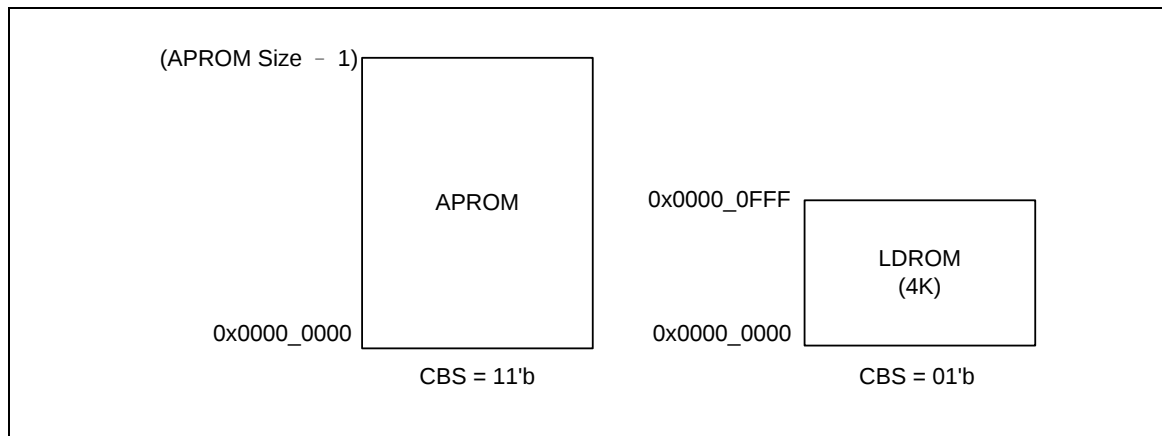


图 6.4-5 APROM 和 LDROM 启动的程序执行范围

对于应用程序，不改变启动模式情况下 软件需要在APROM执行代码并调用LDROM的功能或者在LDROM执行代码并调用APROM功能，CBS[0]需要设置为0，这叫做在应用编程(IAP)。

CBS[1:0]	启动 选择
00	LDROM 带 IAP 功能 芯片从LDROM启动，程序执行范围包括LDROM和大部分APROM（除了APROM的前512 bytes作为LDROM的映射） LDROM地址映像到0x0010_0000 ~ 0x0010_0FFF，同样LDROM的前512 bytes映像到地址0x0000_0000 ~ 0x0000_01FF。 地址0x0000_0000 ~ 0x0000_01FF可以被重映射到ISP执行范围的内的任何其他页。
01	LDROM 不带 IAP 功能 芯片从LDROM启动，程序只在LDROM内执行,APROM不能被程序直接访问除非用ISP。 这个模式下LDROM被写保护。
10	APROM 带 IAP 功能 芯片从APROM启动，程序执行范围包括LDROM和APROM，LDROM地址映像到0x0010_0000~0x0010_0FFF。 地址0x0000_0000 ~ 0x0000_01FF可以被重映射到ISP执行范围的内的任何其他页。
11	APROM 不带 IAP 功能 芯片从APROM启动，程序执行范围只有APROM。LDROM不能被程序直接访问除非用ISP。 这个模式下APROM被写保护。

表 6.4-2 支持的启动选项

6.4.8 在应用编程(IAP)

M058S系列提供在应用编程功能，用户无需复位系统就可以在APROM和LDROM之间切换执行代码。用户可以通过设置CONFIG0 (CBS[1:0]) 为2 或 0并重启来使能IAP功能。

在芯片从APROM启动使能IAP功能(CBS[1:0] = 2)的情况下，代码的执行地址范围包括所有的APROM和LDROM。APROM 地址空间保持不变，但4 KB LDROM的地址空间被映像到0x0010_0000~ 0x0010_1FFF。

在芯片从LDROM启动并使能IAP功能(CBS[1:0] = 0)的情况下，代码的执行地址范围包括所有的LDROM 和 APROM(除了第一页)。执行代码不能访问APROM的第一页，因为第一页的执行地址默认变成LDROM第一页的镜像。同时4 KB LDROM也被映射到0x0010_0000~ 0x0010_1FFF。

当IAP使能时，请地址映像参考下图：

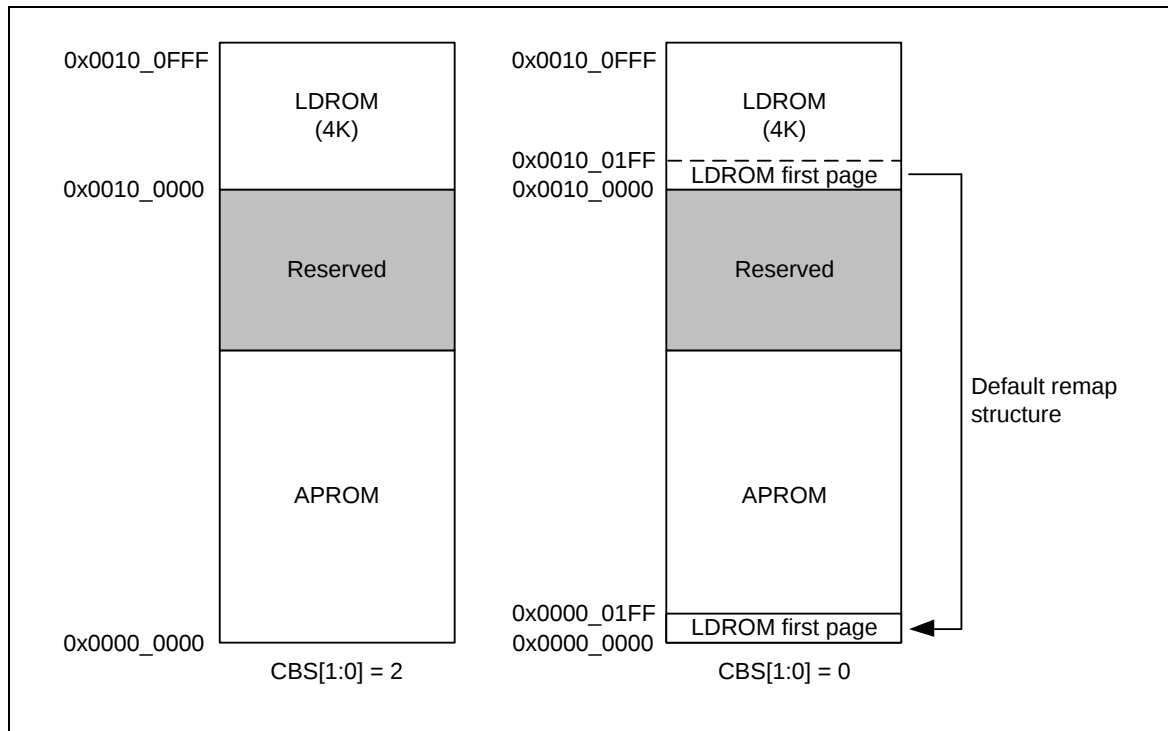


图 6.4-6 IAP 使能时代码的执行范围

当芯片使能了IAP功能，任何时候，代码可执行范围内的任何其他页都可被镜像到执行代码的第一页(0x0000_0000~0x0000_01FF)。用户可以通过ISP向量页命令填重映像的目标地址到ISPADR来改变执行代码的第一页的重新映像地址。改变重映像地址后，通过读ISPSTA 寄存器中VECMAP域，用户可以检查改变是否成功。

6.4.9 在系统编程(ISP)

M058S系列支持ISP模式，当烧写失败时设备可以在软件控制重新烧写，避免系统崩溃的风险。因此，更新应用固件的功能,可以让应用更为广泛。

为了支持ISP，M058S系列包含了LDROM和ISP控制器。用户可以执行在LDROM的ISP装载程序，装载程序可以通过ISP寄存器编写用户应用代码（APROM）。换言之，装载程序可以直接在板上更新系统固件。多种硬件外设接口让ISP更新固件变得更为简单。最常用的方法是通过UART连接LDROM的ISP装载程序来执行ISP。PC一般通过串口传输新的APROM代码，ISP下载程序接收后并通过ISP命令对APROM重新编程。

6.4.10 ISP 控制流程

M058S系列支持通过用户配置初始化定义选择从APROM或LDROM启动。更改用户配置后重启系统生效。如果用户想不更改用户配置来切换APROM或LDROM模式（CBS[0] = 1），那么需要控制ISPCON控制寄存器的BS位然后通过IPRSTC1控制寄存器复位CPU。通过BS位来切换启动流程如下图。只当CBS[0] = 1时通过BS位启动切换功能有效。

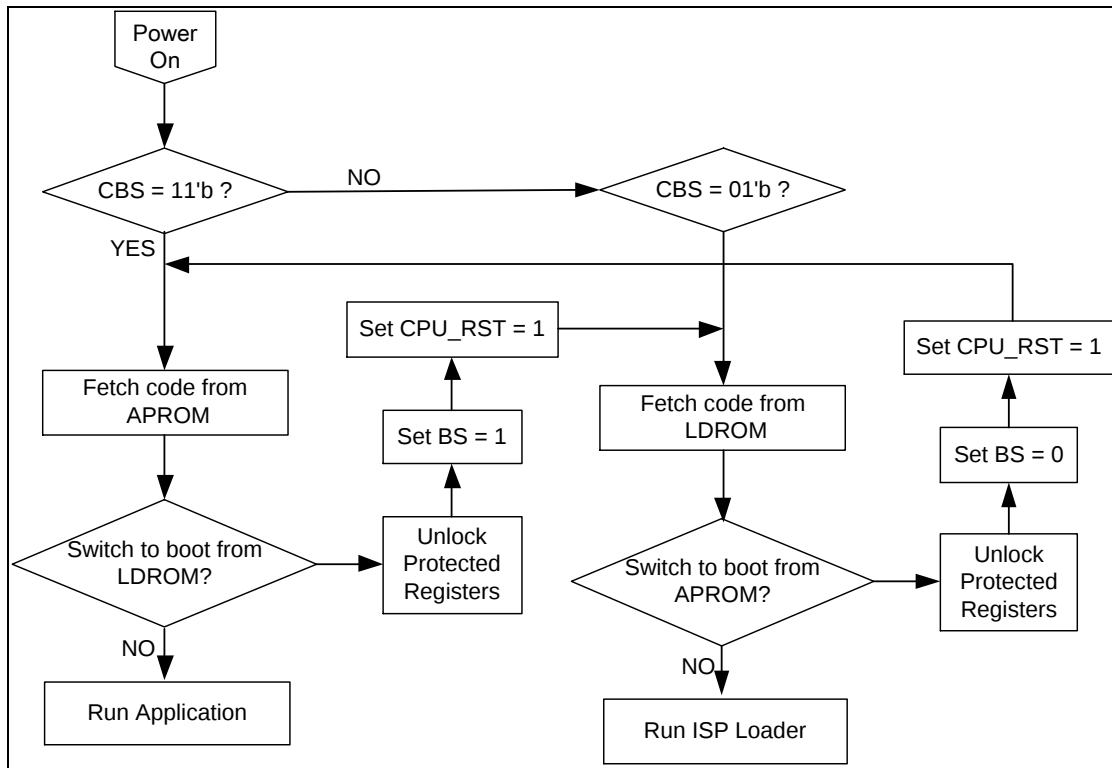


图 6.4-7 当 CBS[0] = 1 BS 位启动选择流程示例

通过LDROM软件更新APROM或者通过APROM软件更新LDROM可以避免在更新失败时系统崩溃。

ISP控制器支持读、写、擦除内部FLASH存储器。ISP控制器的几个控制位是被写保护的，因此设置前需要解锁。解锁保护寄存器位软件需要依次写0x59, 0x16 和 0x88到REGWRPROT。如果解锁成功REGWRPROT将被置1.解锁过程不能被其他访问中断，否则会解锁失败。

解锁保护寄存器位后，用户需要设置ISPCON控制寄存器来决定更新LDROM、用户配置区、APROM和使能ISP控制器。

一旦ISPCON寄存器被设置成功，用户可以通过设置ISPCMD来擦除、读或者写。根据目标FLASH存储器源地址来设置ISPADR。ISPDAT可以用于设置数据写入或者返回 ISPCMD读的数据。

最后设置ISPTRG控制寄存器的ISPGO位来执行相应的ISP功能。当ISP功能完成后ISPGO位将自动清0。为了确保在CPU继续运行之前ISP功能已经完成，在ISPGO设置之后，应该使用ISB指令。

ISP功能完成后几个错误条件需要检查。如果出现错误，ISP操作就没有开始并且ISP失败标志ISPFF被置位。ISPFF标志只能由软件清除。当ISPFF位保持1下一个ISP程序仍然可以开始，因此，建议在每个ISP寄存器操作完成后检查ISPFF位，如果被置1要清0。

当ISPGO位被置1，CPU 将一直等到ISP操作到完成，这个时期内外围设备照常保持运行。任何中断请求都不会响应直到CPU完成ISP操作。当ISP操作完成ISPGO位将被硬件自动清0。用户可以通过ISPGO位来检查ISP操作是否完成。用户应该在ISPGO置1之后加ISB指令，确保ISP操作之后的指令正确执行。

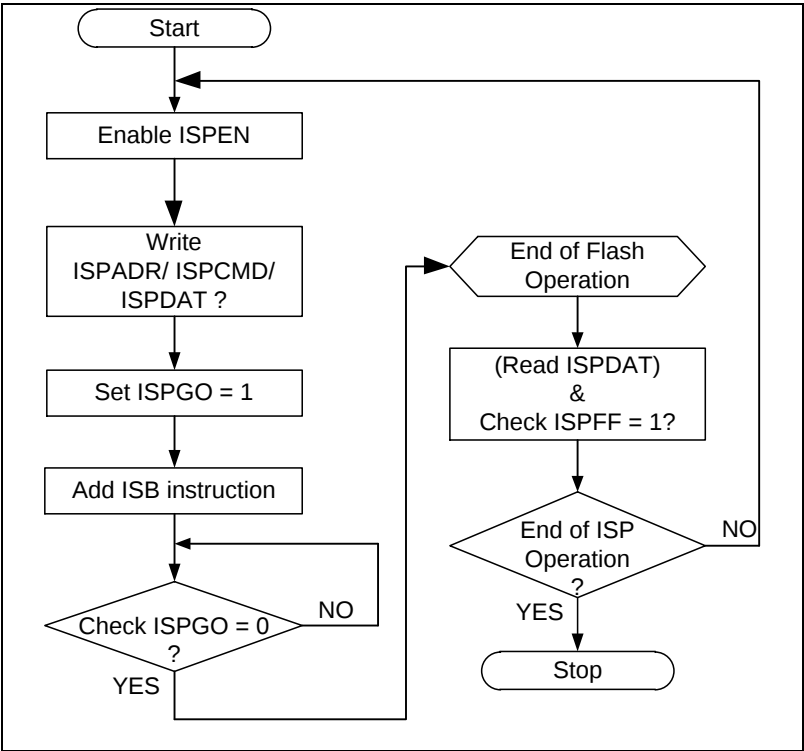


图 6.4-8 ISP 流程示例

ISP 命令	ISPCMD	ISPADR	ISPDAT
FLASH 页擦除	0x22	FLASH存储器结构的有效地址 它必须由512字节/页排列	无效
FLASH Program	0x21	FLASH存储器结构的有效地址	写数据
FLASH Read	0x00	FLASH存储器结构的有效地址	返回数据
读 UID	0x04	0x0000_0000	UID字0
		0x0000_0004	UID字1
		0x0000_0008	UID字2
读 公司 ID	0x0B	无效	公司 ID (0xDA)
向量页重映射	0x2E	APROM 或 LDROM 的页 它必须由512字节/页排列	无效

表 6.4-3 ISP 命令表

6.4.11 通过向量重映射多启动

M058S 系列可以支持通过向量页重映像功能来从不同地址启动。当CBS[0] = 0, LDROM 和 APROM的所有页都可以被重映像到向量地址0x0。重映像地址可以通过寄存器ISPSTA的VECMAP得到。当CBS[1:0] = 2, 上电时重映像地址默认为0x0。意思是上电时向量页从APROM的第一页所映像的APROM地址来启动。当CBS[1:0] = 0, 重映像地址是0x100000。意思是上电时向量页从LDROM的第一页所映像LDROM地址来启动。

重映像地址可以通过向量页Re-Map命令来更改。用户可以通过向量页Re-Map命令重映像指定页到向量页, 而不用CPU_RST 或 SYSRESETREQ来复位系统。CPU将获取新的堆栈并从新的向量页复位处理器指针, 不需要启动到指定应用程序区。

例如, 假如用户在APROM中有两个独立应用叫做App0 和 App1。App0位于0x0, App1位于0x8000。CBS[1:0]置0从LDROM启动。上电时系统将执行LDROM的代码。LDROM的代码将决定从App0 还是 App1启动。为了从App0启动, 在LDROM的代码将使能ISP并设置向量页重映射到0x0, 不用通过CPU_RST (不复位I/O和外设) 或SYSRESETREQ (复位I/O和外设) 来启动到App0。为了从App1启动, 在LDROM的代码将使能ISP并设置向量页重映射到0x8000, 也不用通过CPU_RST (不复位I/O和外设) 或SYSRESETREQ (复位I/O和外设) 来启动到App1。下图显示如何使用向量页重映射启动到不同应用。

为设置向量页重映像, 用户需要设置新的页地址到ISPADR, 设置重映像命令代码0x2E到ISPCMD。然后通过ISPGO置1来触发ISP命令。用户可以通过ISPSTA控制寄存器来确认新的向量页映像地址。

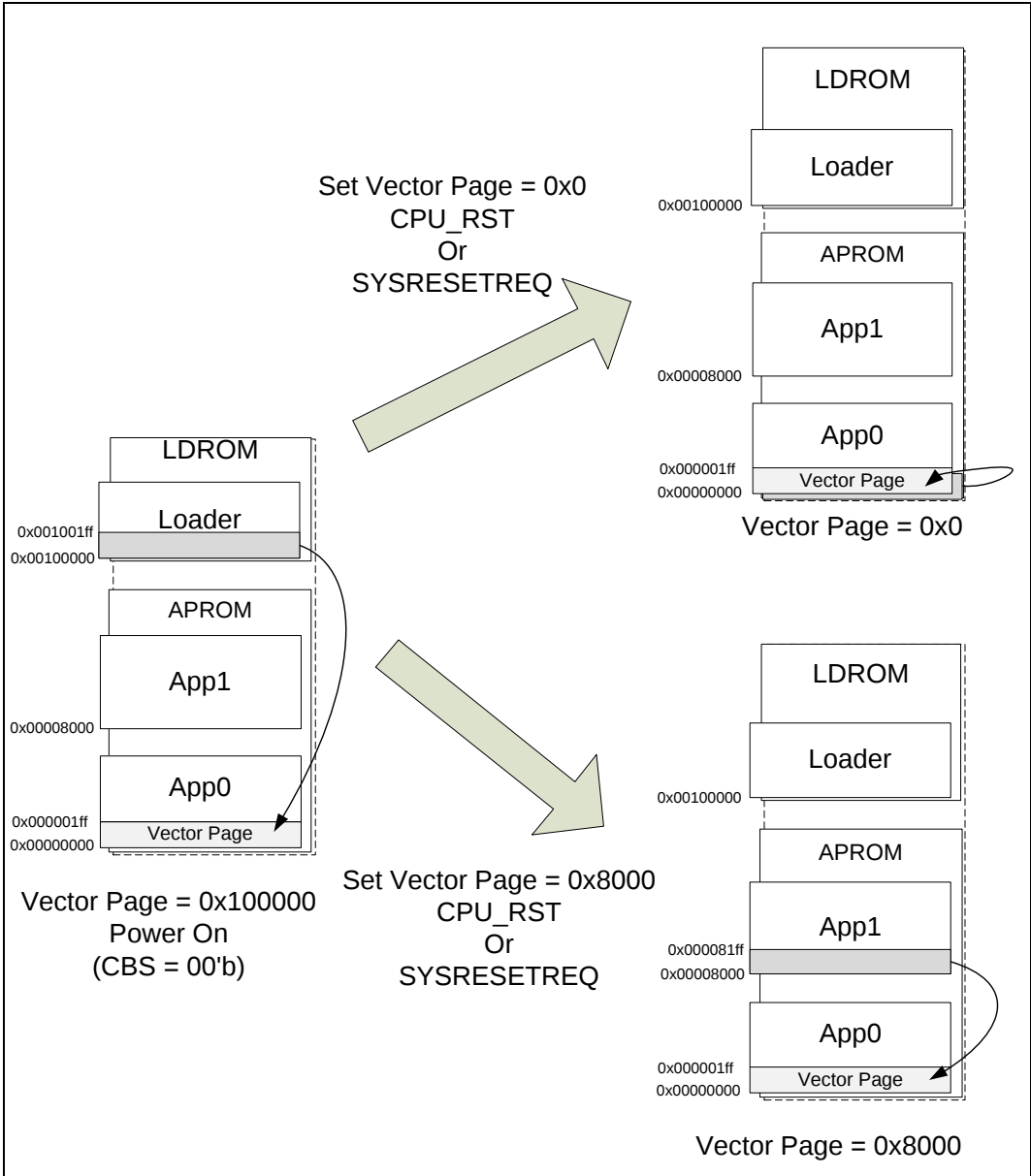


图 6.4-9通过向量页重映射多启动

6.4.12 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
FMC 基地址: FMC_BA = 0x5000_C000				
ISPCON	FMC_BA+0x00	R/W	ISP 控制寄存器	0x0000_0000
ISPADR	FMC_BA+0x04	R/W	ISP 地址寄存器	0x0000_0000
ISPDAT	FMC_BA+0x08	R/W	ISP 数据寄存器	0x0000_0000

ISPCMD	FMC_BA+0x0C	R/W	ISP 命令寄存器	0x0000_0000
ISPTRG	FMC_BA+0x10	R/W	ISP 触发寄存器	0x0000_0000
DFBADR	FMC_BA+0x14	R	数据Flash起始地址	0x0001_F000
FATCON	FMC_BA+0x18	R/W	Flash访问时间控制寄存器	0x0000_0000
ISPSTA	FMC_BA+0x40	R/W	ISP 状态寄存器	0x0000_0000

6.4.13 寄存器描述

ISP控制寄存器(ISPCON)

寄存器	偏移量	R/W	描述	复位值
ISPCON	FMC_BA+0x00	R/W	ISP控制寄存器	0x0000_0000

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

Bits	描述	
[31:7]	Reserved	保留
[6]	ISPPF	ISP 失败标志 (写保护) 当ISP遇到下列条件时, 该位由硬件置位: (1) APUEN等于0时, APROM 写APROM. (2) LDROM 写LDROM. (3)定义无效地址, 如超过正常范围. 注: 该位写 1 清零
[5]	LDUEN	LDROM更新使能 位(写保护位) 0 = 禁止LDROM更新 1 =当芯片在 APROM 中运行时, LDROM 可以更新。
[4]	CFGUEN	CONFIG 更新使能控制(写保护) 写1到该位, 软件通过ISP更新CONFIG值, 不管程序在APROM 还是 LDROM运行。 0 = 禁止ISP更新用户配置 1 =使能ISP更新用户配置
[3]	APUEN	APROM 更新使能 (写保护位) 0 = 当芯片在APROM中运行时APROM 不能被更新。 1 = 当芯片在APROM中运行时APROM 可以被更新。
[2]	Reserved	保留。
[1]	BS	启动选择 (写保护位)

		置位/清零该位选择下次是由LDROM启动还是由APROM启动, 该位也可作为MCU启动状态的标志, 用于检查芯片是由LDROM还是APROM启动的.复位时该位被初始化为 Config0 的 CBS位的反转值, 除非CPU复位(RSTS_CPU 时 1)或者系统复位(RSTS_SYS)发生。 1 = 由LDROM启动 0 = 由APROM启动
[0]	ISPEN	ISP 使能(写保护位) 设置该位使能ISP功能. 1 = 使能 ISP 功能 0 = 禁用 ISP 功能

ISP地址寄存器(ISPADR)

寄存器	偏移量	R/W	描述	复位值
ISPADR	FMC_BA+0x04	R/W	ISP地址寄存器	0x0000_0000

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

Bits	描述	
[31:0]	ISPADR	ISP 地址 M058S 系列有最大 16K x 32-bit (32 KB)内嵌 Flash,只支持字编程。执行 ISP 功能时, ISPARD[1:0] 必须为 00'b。

ISP数据寄存器 (ISPDAT)

寄存器	偏移量	R/W	描述	复位值
ISPDAT	FMC_BA+0x08	R/W	ISP 数据寄存器	0x0000_0000

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

Bits	描述	
[31:0]	ISPDAT	ISP 数据 ISP写操作之前，写数据到该寄存器 ISP读操作后，可从该寄存器读数据

ISP命令寄存器 (ISPCMD)

寄存器	偏移量	R/W	描述	复位值
ISPCMD	FMC_BA+0x0C	R/W	ISP 命令 寄存器	0x0000_0000

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

Bits	描述	
[31:6]	Reserved	保留.
[5:0]	ISPCMD	ISP命令 ISP 命令表如下: 0x00 = 读 0x04 =读UID 0x0B = 读公司ID (0xDA). 0x21 = 写. 0x22 = 页擦除. 0x2E =向量页重映射.

ISP触发控制寄存器 (ISPTRG)

寄存器	偏移量	R/W	描述	复位值
ISPTRG	FMC_BA+0x10	R/W	ISP触发控制寄存器	0x0000_0000

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

Bits	描述	
[31:1]	Reserved	保留.
[0]	ISPGO	ISP开始触发（写保护） 写 1 开始ISP操作，当ISP操作结束后，该位由硬件自动清零. 0 = ISP 操作结束 1 = ISP 正在执行

数据FLASH基地址寄存器(DFBADR)

寄存器	偏移量	R/W	描述	复位值
DFBADR	FMC_BA+0x14	R	数据Flash基地址	0x0001_F000

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

Bits	描述	
[31:0]	DFBADR	数据FLASH基地址 该寄存器为数据FLASH起始地址寄存器，只读。 32 KB flash的存储器设备中数据FLASH空间为4 KB，起始地址由内部硬件固定在0x0001_F000。

Flash访问时间控制寄存器(FATCON)

寄存器	偏移量	R/W	描述	复位值
FATCON	FMC_BA+0x18	R/W	Flash访问时间控制寄存器	0x0000_0000

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

Bits	描述	
[31:8]	Reserved	保留.
[7]	Reserved	保持为 0.
[6:5]	Reserved	保留.
[4]	LFOM	低频优化模式控制（写保） 当运作频率低于25 MHz，该位置1芯片可以运作更高效。 0 = 禁止低频优化模式 1 = 使能低频优化模式
[0]	Reserved	保留.

ISP状态寄存器(ISPSTA)

寄存器	偏移量	R/W	描述	复位值
ISPSTA	FMC_BA+0x40	R/W	ISP 状态寄存器	0x0000_0000

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

Bits	描述	
[31:21]	Reserved	保留。
[20:9]	VECMAP	向量页映像地址（写保护） 当前 flash 地址空间 0x0000_0000~0x0000_01FF 映像到地址 {VECMAP[11:0], 9'h000} ~ {VECMAP[11:0], 9'h1FF}
[8:7]	Reserved	保留。
[6]	ISPFF	ISP失败标志（写保护） 当 ISP 遇到下列条件时，该位由硬件置位： (1) APROM 写APROM (2) LDROM 写LDROM (3) 如果 CFGUEN 设置为 0，CONFIG 被擦除/编程 (4) 目的地址无效，比如超过正常范围 写 1 清除该位。 注意：该位的功能同ISPCON bit6.
[5:3]	Reserved	保留。
[2:1]	CBS	芯片启动选择（只读） 这是CBS在CONFIG0中的镜像
[0]	ISPGO	ISP 开始触发（只读） 写 1 开始 ISP 操作，当 ISP 操作结束后，该位由硬件自动清零。 0 = ISP 操作结束 1 = ISP 正在执行 注意：这位同ISPTRG bit0

6.5 通用 I/O (GPIO)

6.5.1 概述

本MCU有58个通用I/O管脚并复用指定功能。58个管脚排成8个端口，名为P0, P1... 到 P7。除了P7[1:0]，其他每个端口多至8个管脚。每个管脚都独立并且有相应寄存器位来控制该管脚模式功能和数据。

每个I/O管脚的I/O类型都可以通过软件单独配置成为输入、输出、开漏或者转双向模式。所有I/O设置成准双向模式时，端口数据寄存器Px_DOUT[7:0]复位到0x000_00FF。每个I/O管脚都配备一个独立阻值为110KΩ~300KΩ的弱上拉电阻连接到V_{DD}，V_{DD}范围从5.0 V 到2.5 V。

6.5.2 特性

- 四种 I/O 模式：
 - 高阻态输入
 - 推挽输出
 - 开漏输出
 - 准双向 通过Px_MFP[23:16]选择TTL/Schmitt触发输入模式。
- I/O可以配置为边沿/电平触发的中断源
- I/O管脚内部上拉电阻只在准双向I/O模式中使用。
- 使能管脚中断功能也将使能管脚唤醒功能。
- 通过CIOINI(CONFIG[10]) 可配置所有I/O复位之后的默认模式。
 - CIOINI = 0, 复位后所有的GPIO管脚是三态（高阻）模式
 - CIOINI = 1, 复位后所有的GPIO管脚是准双向模式

6.5.3 基本配置

GPIO管脚通过功能寄存器P0_MFP, P1_MFP, P2_MFP, P3_MFP, P4_MFP, P5_MFP 和 P7_MFP 来配置。

6.5.4 功能描述

6.5.4.1 输入模式

设置Px_PMD (PMDn[1:0]) 为 00, Px.n管脚为输入模式, I/O管脚为三态 (高阻), 没有输出驱动能力。Px_PIN的值反映相应端口的状态。

6.5.4.2 推挽输出模式

设置Px_PMD (PMDn[1:0]) 为 01, Px.n为推挽输出模式, I/O支持数字输出功能, 有拉/灌电流能力。相应位Px_DOUT[n]相应的值被送到相应管脚上

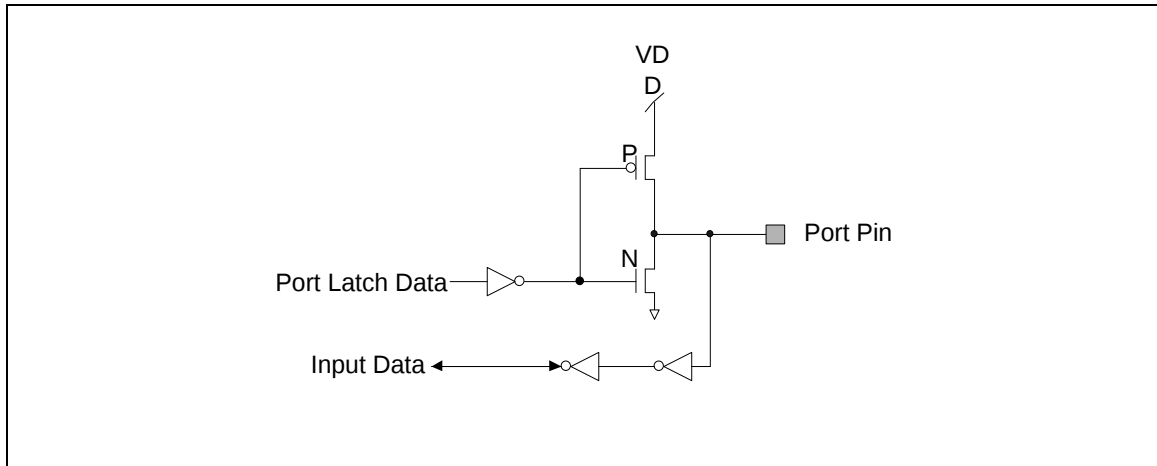


图 6.5-1 推挽输出

6.5.4.3 开漏输出模式

设置Px_PMD (PMDn[1:0])为10, Px.n管脚为开漏模式且I/O管脚数字输出功能仅支持灌电流, 驱动到高电平需要一个外加上拉电阻。如果相应位Px_DOUT[n]的值为0, 管脚上输出低。如果相应位Px_DOUT[n]的值为1, 该管脚输出为高, 由外部上拉电阻控制。

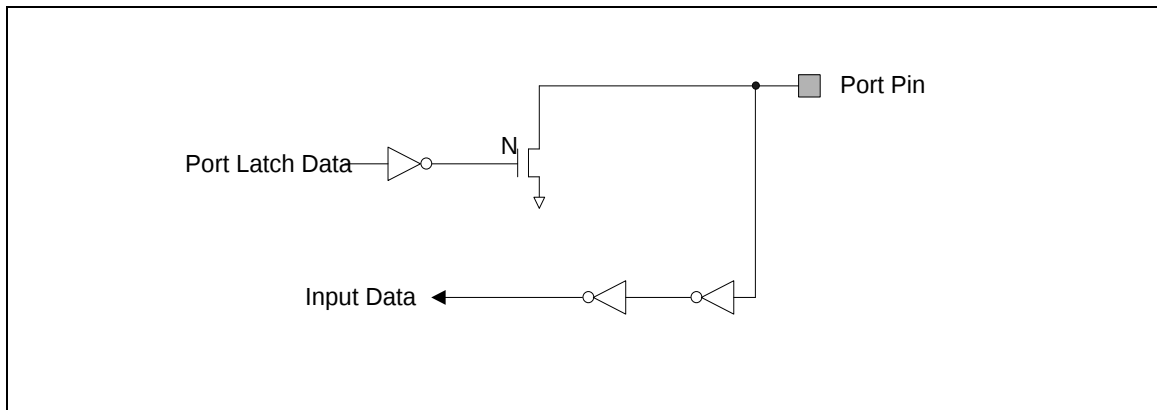


图 6.5-2 开漏输出

6.5.4.4 准双向模式

设置Px_PMD (PMDn[1:0])为11, Px.n管脚为准双向模式, I/O同时支持数字输出和输入功能, 但拉电流能力只有数百uA。要实现数字输入, 需要先将相应Px_DOUT[n]位置1。准双向输出是80C51

及其派生产品常见的模式。若相应位Px_DOUT[n]为0，管脚上输出为“低”。若相应位Px_DOUT[n]为‘1’，该管脚将检测管脚值。若管脚值为高，没有任何动作，若管脚值为低，该管脚上将驱动2个时钟周期的强上拉，然后禁止强输出驱动，同时管脚状态由内部上拉电阻控制。

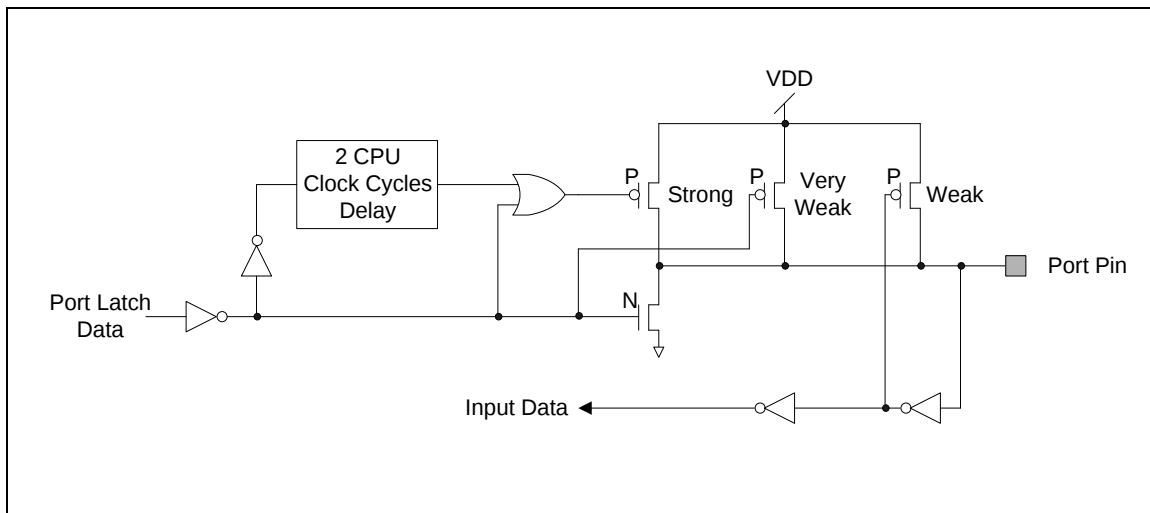


图 6.5-3 准双向 I/O 模式

6.5.5 GPIO 中断和唤醒功能

每个GPIO管脚都可以通过相应Px_IEN位和Px_IMD设置成芯片的中断源。有五种中断条件可以设置：低电平触发、高电平触发、下降沿触发、上升沿触发和上下边缘都触发。在边沿触发中用户可以通过使能输入信号去抖功能来阻止由噪声引起的意外中断。去抖时钟源和采样周期可以通过DBNCECON寄存器来设置。

当芯片进入空闲模式或掉电模式后，GPIO也可以用来当作唤醒源。唤醒触发条件设置跟GPIO中断触发设置相同。

6.5.6 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
GP_BA = 0x5000_4000				
P0_PMD	GP_BA+0x000	R/W	P0 管脚I/O模式控制	0x0000_FFFF
P0_OFFD	GP_BA+0x004	R/W	P0 数字通路关闭控制	0x0000_0000
P0_DOUT	GP_BA+0x008	R/W	P0 数据输出值	0x0000_00FF
P0_DMASK	GP_BA+0x00C	R/W	P0 数据输出写屏蔽	0x0000_0000
P0_PIN	GP_BA+0x010	R	P0 管脚值	0x0000_00XX
P0_DBEN	GP_BA+0x014	R/W	P0 去抖使能	0x0000_0000
P0_IMD	GP_BA+0x018	R/W	P0 中断模式控制	0x0000_0000
P0_IEN	GP_BA+0x01C	R/W	P0 中断使能	0x0000_0000
P0_ISRC	GP_BA+0x020	R/WC	P0 中断源标志	0xFFFF_FFFF
P1_PMD	GP_BA+0x040	R/W	P1管脚I/O模式控制	0x0000_FFFF
P1_OFFD	GP_BA+0x044	R/W	P1数字通路关闭控制	0x0000_0000
P1_DOUT	GP_BA+0x048	R/W	P1数据输出值	0x0000_00FF
P1_DMASK	GP_BA+0x04C	R/W	P1 数据输出写屏蔽	0x0000_0000
P1_PIN	GP_BA+0x050	R	P1管脚值	0x0000_00XX
P1_DBEN	GP_BA+0x054	R/W	P1去抖使能	0x0000_0000
P1_IMD	GP_BA+0x058	R/W	P1中断模式控制	0x0000_0000
P1_IEN	GP_BA+0x05C	R/W	P1中断使能	0x0000_0000
P1_ISRC	GP_BA+0x060	R/WC	P1 中断源标志	0xFFFF_FFFF
P2_PMD	GP_BA+0x080	R/W	P2 管脚I/O模式控制	0x0000_FFFF
P2_OFFD	GP_BA+0x084	R/W	P2 数字通路关闭控制	0x0000_0000
P2_DOUT	GP_BA+0x088	R/W	P2数据输出值	0x0000_00FF
P2_DMASK	GP_BA+0x08C	R/W	P2数据输出写屏蔽	0x0000_0000
P2_PIN	GP_BA+0x090	R	P2管脚值	0x0000_00XX
P2_DBEN	GP_BA+0x094	R/W	P2去抖使能	0x0000_0000

P2_IMD	GP_BA+0x098	R/W	P2中断模式控制	0x0000_0000
P2_IEN	GP_BA+0x09C	R/W	P2中断使能	0x0000_0000
P2_ISRC	GP_BA+0x0A0	R/WC	P2中断源标志	0xXXXX_XXXX
P3_PMD	GP_BA+0x0C0	R/W	P3管脚I/O模式控制	0x0000_FFFF
P3_OFFD	GP_BA+0x0C4	R/W	P3数字通路关闭控制	0x0000_0000
P3_DOUT	GP_BA+0x0C8	R/W	P3数据输出值	0x0000_00FF
P3_DMASK	GP_BA+0x0CC	R/W	P3数据输出写屏蔽	0x0000_0000
P3_PIN	GP_BA+0x0D0	R	P3管脚值	0x0000_00XX
P3_DBEN	GP_BA+0x0D4	R/W	P3去抖使能	0x0000_0000
P3_IMD	GP_BA+0x0D8	R/W	P3中断模式控制	0x0000_0000
P3_IEN	GP_BA+0x0DC	R/W	P3中断使能	0x0000_0000
P3_ISRC	GP_BA+0x0E0	R/WC	P3中断源标志	0xXXXX_XXXX
P4_PMD	GP_BA+0x100	R/W	P4管脚I/O模式控制	0x0000_FFFF
P4_OFFD	GP_BA+0x104	R/W	P4数字通路关闭控制	0x0000_0000
P4_DOUT	GP_BA+0x108	R/W	P4数据输出值	0x0000_00FF
P4_DMASK	GP_BA+0x10C	R/W	P4数据输出写屏蔽	0x0000_0000
P4_PIN	GP_BA+0x110	R	P4管脚值	0x0000_00XX
P4_DBEN	GP_BA+0x114	R/W	P4去抖使能	0x0000_0000
P4_IMD	GP_BA+0x118	R/W	P4中断模式控制	0x0000_0000
P4_IEN	GP_BA+0x11C	R/W	P4中断使能	0x0000_0000
P4_ISRC	GP_BA+0x120	R/WC	P4中断源标志	0xXXXX_XXXX
DBNCECON	GP_BA+0x180	R/W	去抖循环控制	0x0000_0020
P00_DOUT	GP_BA+0x200	R/W	P0.0 数据输出值	0x0000_0001
P01_DOUT	GP_BA+0x204	R/W	P0.1 数据输出值	0x0000_0001
P02_DOUT	GP_BA+0x208	R/W	P0.2 数据输出值	0x0000_0001
P03_DOUT	GP_BA+0x20C	R/W	P0.3 数据输出值	0x0000_0001
P04_DOUT	GP_BA+0x210	R/W	P0.4 数据输出值	0x0000_0001
P05_DOUT	GP_BA+0x214	R/W	P0.5 数据输出值	0x0000_0001

P06_DOUT	GP_BA+0x218	R/W	P0.6 数据输出值	0x0000_0001
P07_DOUT	GP_BA+0x21C	R/W	P0.7 数据输出值	0x0000_0001
P10_DOUT	GP_BA+0x220	R/W	P1.0 数据输出值	0x0000_0001
P11_DOUT	GP_BA+0x224	R/W	P1.1 数据输出值	0x0000_0001
P12_DOUT	GP_BA+0x228	R/W	P1.2 数据输出值	0x0000_0001
P13_DOUT	GP_BA+0x22C	R/W	P1.3 数据输出值	0x0000_0001
P14_DOUT	GP_BA+0x230	R/W	P1.4 数据输出值	0x0000_0001
P15_DOUT	GP_BA+0x234	R/W	P1.5 数据输出值	0x0000_0001
P16_DOUT	GP_BA+0x238	R/W	P1.6 数据输出值	0x0000_0001
P17_DOUT	GP_BA+0x23C	R/W	P1.7 数据输出值	0x0000_0001
P20_DOUT	GP_BA+0x240	R/W	P2.0 数据输出值	0x0000_0001
P21_DOUT	GP_BA+0x244	R/W	P2.1 数据输出值	0x0000_0001
P22_DOUT	GP_BA+0x248	R/W	P2.2 数据输出值	0x0000_0001
P23_DOUT	GP_BA+0x24C	R/W	P2.3 数据输出值	0x0000_0001
P24_DOUT	GP_BA+0x250	R/W	P2.4 数据输出值	0x0000_0001
P25_DOUT	GP_BA+0x254	R/W	P2.5 数据输出值	0x0000_0001
P26_DOUT	GP_BA+0x258	R/W	P2.6 数据输出值	0x0000_0001
P27_DOUT	GP_BA+0x25C	R/W	P2.7 数据输出值	0x0000_0001
P30_DOUT	GP_BA+0x260	R/W	P3.0 数据输出值	0x0000_0001
P31_DOUT	GP_BA+0x264	R/W	P3.1 数据输出值	0x0000_0001
P32_DOUT	GP_BA+0x268	R/W	P3.2 数据输出值	0x0000_0001
P33_DOUT	GP_BA+0x26C	R/W	P3.3 数据输出值	0x0000_0001
P34_DOUT	GP_BA+0x270	R/W	P3.4 数据输出值	0x0000_0001
P35_DOUT	GP_BA+0x274	R/W	P3.5 数据输出值	0x0000_0001
P36_DOUT	GP_BA+0x278	R/W	P3.6 数据输出值	0x0000_0001
P37_DOUT	GP_BA+0x27C	R/W	P3.7 数据输出值	0x0000_0001
P40_DOUT	GP_BA+0x280	R/W	P4.0 数据输出值	0x0000_0001
P41_DOUT	GP_BA+0x284	R/W	P4.1 数据输出值	0x0000_0001

P42_DOUT	GP_BA+0x288	R/W	P4.2 数据输出值	0x0000_0001
P43_DOUT	GP_BA+0x28C	R/W	P4.3 数据输出值	0x0000_0001
P44_DOUT	GP_BA+0x290	R/W	P4.4 数据输出值	0x0000_0001
P45_DOUT	GP_BA+0x294	R/W	P4.5 数据输出值	0x0000_0001
P46_DOUT	GP_BA+0x298	R/W	P4.6 数据输出值	0x0000_0001
P47_DOUT	GP_BA+0x29C	R/W	P4.7 数据输出值	0x0000_0001
P5_PMD	GP_BA+0x2C0	R/W	P5 管脚I/O模式控制	0x0000_FFFF
P5_OFFD	GP_BA+0x2C4	R/W	P5 数字通路关闭控制	0x0000_0000
P5_DOUT	GP_BA+0x2C8	R/W	P5 数据输出值	0x0000_00FF
P5_DMASK	GP_BA+0x2CC	R/W	P5 数据输出写屏蔽	0x0000_0000
P5_PIN	GP_BA+0x2D0	R	P5 管脚值	0x0000_00XX
P5_DBEN	GP_BA+0x2D4	R/W	P5 去抖使能	0x0000_0000
P5_IMD	GP_BA+0x2D8	R/W	P5 中断模式控制	0x0000_0000
P5_IEN	GP_BA+0x2DC	R/W	P5 中断使能	0x0000_0000
P5_ISRC	GP_BA+0x2E0	R/WC	P5 中断源标志	0XXXXX_XXXX
P6_PMD	GP_BA+0x300	R/W	P6 管脚I/O模式控制	0x0000_FFFF
P6_OFFD	GP_BA+0x304	R/W	P6 数字通路关闭控制	0x0000_0000
P6_DOUT	GP_BA+0x308	R/W	P6 数据输出值	0x0000_00FF
P6_DMASK	GP_BA+0x30C	R/W	P6 数据输出写屏蔽	0x0000_0000
P6_PIN	GP_BA+0x310	R	P6 管脚值	0x0000_00XX
P6_DBEN	GP_BA+0x314	R/W	P6 去抖使能	0x0000_0000
P6_IMD	GP_BA+0x318	R/W	P6 中断模式控制	0x0000_0000
P6_IEN	GP_BA+0x31C	R/W	P6 中断使能	0x0000_0000
P6_ISRC	GP_BA+0x320	R/WC	P6 中断源标志	0XXXXX_XXXX
P7_PMD	GP_BA+0x340	R/W	P7 管脚I/O模式控制	0x0000_FFFF
P7_OFFD	GP_BA+0x344	R/W	P7 数字通路关闭控制	0x0000_0000
P7_DOUT	GP_BA+0x348	R/W	P7 数据输出值	0x0000_00FF
P7_DMASK	GP_BA+0x34C	R/W	P7 数据输出写屏蔽	0x0000_0000

P7_PIN	GP_BA+0x350	R	P7 管脚值	0x0000_00XX
P7_DBEN	GP_BA+0x354	R/W	P7 去抖使能	0x0000_0000
P7_IMD	GP_BA+0x358	R/W	P7 中断模式控制	0x0000_0000
P7_IEN	GP_BA+0x35C	R/W	P7 中断使能	0x0000_0000
P7_ISRC	GP_BA+0x360	R/WC	P7 中断源标志	0XXXXX_XXXX
P50_DOUT	GP_BA+0x380	R/W	P5.0 数据输出值	0x0000_0001
P51_DOUT	GP_BA+0x384	R/W	P5.1 数据输出值	0x0000_0001
P52_DOUT	GP_BA+0x388	R/W	P5.2 数据输出值	0x0000_0001
P53_DOUT	GP_BA+0x38C	R/W	P5.3 数据输出值	0x0000_0001
P54_DOUT	GP_BA+0x390	R/W	P5.4 数据输出值	0x0000_0001
P55_DOUT	GP_BA+0x394	R/W	P5.5 数据输出值	0x0000_0001
P56_DOUT	GP_BA+0x398	R/W	P5.6 数据输出值	0x0000_0001
P57_DOUT	GP_BA+0x39C	R/W	P5.7 数据输出值	0x0000_0001
P60_DOUT	GP_BA+0x3A0	R/W	P6.0 数据输出值	0x0000_0001
P61_DOUT	GP_BA+0x3A4	R/W	P6.1 数据输出值	0x0000_0001
P62_DOUT	GP_BA+0x3A8	R/W	P6.2 数据输出值	0x0000_0001
P63_DOUT	GP_BA+0x3AC	R/W	P6.3 数据输出值	0x0000_0001
P64_DOUT	GP_BA+0x3B0	R/W	P6.4 数据输出值	0x0000_0001
P65_DOUT	GP_BA+0x3B4	R/W	P6.5 数据输出值	0x0000_0001
P66_DOUT	GP_BA+0x3B8	R/W	P6.6 数据输出值	0x0000_0001
P67_DOUT	GP_BA+0x3BC	R/W	P6.7 数据输出值	0x0000_0001
P70_DOUT	GP_BA+0x3C0	R/W	P7.0 数据输出值	0x0000_0001
P71_DOUT	GP_BA+0x3C4	R/W	P7.1 数据输出值	0x0000_0001

6.5.7 寄存器描述

端口 0-7 I/O 模式控制 (Px_PMD)

寄存器	偏移量	R/W	描述	复位值
P0_PMD	GP_BA+0x000	R/W	P0 管脚I/O模式控制	0x0000_FFFF
P1_PMD	GP_BA+0x040	R/W	P1 管脚I/O模式控制	0x0000_FFFF
P2_PMD	GP_BA+0x080	R/W	P2 管脚I/O模式控制	0x0000_FFFF
P3_PMD	GP_BA+0x0C0	R/W	P3 管脚I/O模式控制	0x0000_FFFF
P4_PMD	GP_BA+0x100	R/W	P4 管脚I/O模式控制	0x0000_FFFF
P5_PMD	GP_BA+0x2C0	R/W	P5 管脚I/O模式控制	0x0000_FFFF
P6_PMD	GP_BA+0x300	R/W	P6 管脚I/O模式控制	0x0000_FFFF
P7_PMD	GP_BA+0x340	R/W	P7 管脚I/O模式控制	0x0000_FFFF

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
PMD7		PMD6		PMD5		PMD4	
7	6	5	4	3	2	1	0
PMD3		PMD2		PMD1		PMD0	

Bits	描述	
[31:16]	Reserved	保留
[2n+1 :2n]	PMDn	Px I/O Pin[n] 模式控制 决定每个Px管脚的I/O类型 00 = Px [n] 管脚 输入模式 01 = Px [n] 管脚 输出模式 10 = Px [n] 管脚 开漏模式 11 = Px [n] 管脚 准双向模式 x=0~7, n = 0~7

端口 0-7 数字通路关闭控制 (Px_OFFD)

寄存器	偏移量	R/W	描述	复位值
P0_OFFD	GP_BA+0x004	R/W	P0 数字通路关闭控制	0x0000_0000
P1_OFFD	GP_BA+0x044	R/W	P1 数字通路关闭控制	0x0000_0000
P2_OFFD	GP_BA+0x084	R/W	P2 数字通路关闭控制	0x0000_0000
P3_OFFD	GP_BA+0x0C4	R/W	P3 数字通路关闭控制	0x0000_0000
P4_OFFD	GP_BA+0x104	R/W	P4 数字通路关闭控制	0x0000_0000
P5_OFFD	GP_BA+0x2C4	R/W	P5 数字通路关闭控制	0x0000_0000
P6_OFFD	GP_BA+0x304	R/W	P6 数字通路关闭控制	0x0000_0000
P7_OFFD	GP_BA+0x344	R/W	P7 数字通路关闭控制	0x0000_0000

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

Bits	描述	
[31:24]	Reserved	保留。
[23:16]	OFFD	端口 0-7 Pin [n] 数字通路关闭控制 这些位用于控制Px.n管脚相应的这些数字输入通路是否关闭。 如果输入为模拟信号，用户可以关闭Px.n管脚输入通路防止漏电。 0 = 使能Px.n 管脚数字通路 1 = 关闭 Px.n管脚数字通路(数字输入拉低). 注: x = 0~7, n = 0~7.
[15:0]	Reserved	保留。

端口 0-7 数据输出值 (Px DOUT)

寄存器	偏移量	R/W	描述	复位值
P0_DOUT	GP_BA+0x008	R/W	P0 数据输出值	0x0000_00FF
P1_DOUT	GP_BA+0x048	R/W	P1 数据输出值	0x0000_00FF
P2_DOUT	GP_BA+0x088	R/W	P2 数据输出值	0x0000_00FF
P3_DOUT	GP_BA+0x0C8	R/W	P3 数据输出值	0x0000_00FF
P4_DOUT	GP_BA+0x108	R/W	P4 数据输出值	0x0000_00FF
P5_DOUT	GP_BA+0x2C8	R/W	P5 数据输出值	0x0000_00FF
P6_DOUT	GP_BA+0x308	R/W	P6 数据输出值	0x0000_00FF
P7_DOUT	GP_BA+0x348	R/W	P7 数据输出值	0x0000_00FF

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

Bits	描述	
[31:8]	Reserved	保留.
[15:0]	DOUT[n]	端口 0-7 Pin [n] 输出值 在Px.n配置成输出，开漏和准双向模式时，这些位控制Px.n管脚的状态。 0 = Px.n配置成输出，开漏和准双向模式时，Px.n状态为低 1 = Px.n配置成输出，开漏和准双向模式时，Px.n状态为高 注: x = 0~7, n = 0~7.

端口0-7 数据输出写屏蔽 (Px_DMASK)

寄存器	偏移量	R/W	描述	复位值
P0_DMASK	GP_BA+0x00C	R/W	P0 数据输出写屏蔽	0x0000_0000
P1_DMASK	GP_BA+0x04C	R/W	P1 数据输出写屏蔽	0x0000_0000
P2_DMASK	GP_BA+0x08C	R/W	P2 数据输出写屏蔽	0x0000_0000
P3_DMASK	GP_BA+0x0CC	R/W	P3 数据输出写屏蔽	0x0000_0000
P4_DMASK	GP_BA+0x10C	R/W	P4 数据输出写屏蔽	0x0000_0000
P5_DMASK	GP_BA+0x2CC	R/W	P5 数据输出写屏蔽	0x0000_0000
P6_DMASK	GP_BA+0x30C	R/W	P6 数据输出写屏蔽	0x0000_0000
P7_DMASK	GP_BA+0x34C	R/W	P7 数据输出写屏蔽	0x0000_0000

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

Bits	描述	
[31:8]	Reserved	保留。
[7:0]	DMASK[n]	端口 0-7 Pin [n] 数据输出写屏蔽 这些位用于保护相应的Px_DOUT[n]位。当DMASK[n]位置1，相应的Px_DOUT[n]位被保护。写信号被屏蔽时，写到保护位的数据被忽略。 0 = Px_DOUT[n]相应位可以更新。 1 = Px_DOUT[n] 相应位被保护。 注 1： x = 0~7, n = 0~7。 注 2： 这些功能只保护 Px_DOUT[n]相应位，不保护 Pxn_PDIO 相应位。

端口 0-7 管脚值 (Px_PIN)

寄存器	偏移量	R/W	描述	复位值
P0_PIN	GP_BA+0x010	R	P0 管脚值	0x0000_00XX
P1_PIN	GP_BA+0x050	R	P1 管脚值	0x0000_00XX
P2_PIN	GP_BA+0x090	R	P2 管脚值	0x0000_00XX
P3_PIN	GP_BA+0x0D0	R	P3 管脚值	0x0000_00XX
P4_PIN	GP_BA+0x110	R	P4 管脚值	0x0000_00XX
P5_PIN	GP_BA+0x2D0	R	P5 管脚值	0x0000_00XX
P6_PIN	GP_BA+0x310	R	P6 管脚值	0x0000_00XX
P7_PIN	GP_BA+0x350	R	P7 管脚值	0x0000_00XX

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

Bits	描述	
[31:8]	Reserved	保留
[15:0]	PIN[n]	端口 0-7 Pin [n] 管脚值 寄存器的每个位真实反映了Px.n每个管脚状态。如果该位是1，表明相应管脚状态是高，相反则为低。 注: x = 0~7, n = 0~7.

端口 0-7 去抖使能 (Px DBEN)

寄存器	偏移量	R/W	描述	复位值
P0_DBEN	GP_BA+0x014	R/W	P0 去抖使能 控制	0x0000_0000
P1_DBEN	GP_BA+0x054	R/W	P1 去抖使能 控制	0x0000_0000
P2_DBEN	GP_BA+0x094	R/W	P2 去抖使能 控制	0x0000_0000
P3_DBEN	GP_BA+0x0D4	R/W	P3 去抖使能 控制	0x0000_0000
P4_DBEN	GP_BA+0x114	R/W	P4 去抖使能 控制	0x0000_0000
P5_DBEN	GP_BA+0x2D4	R/W	P5 去抖使能 控制	0x0000_0000
P6_DBEN	GP_BA+0x314	R/W	P6 去抖使能 控制	0x0000_0000
P7_DBEN	GP_BA+0x354	R/W	P7 去抖使能 控制	0x0000_0000

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

Bits	描述	
[31:8]	Reserved	保留.
[15:0]	DBEN[n]	端口 0-7 Pin [n]输入信号去抖使能 控制 DBEN[n]用于使能相应位的去抖动功能. 如果输入信号脉冲宽度不能被两个连续的去抖动采样周期所采样, 则被视为抖动信号, 从而不触发中断. 去抖动时钟源由DBNCECON[4]控制, 一个去抖动周期由DBNCECON[3:0]控制. 0 = 禁止Px.n去抖功能 1 =使能 Px.n 去抖功能 去抖功能只在边缘触发中断有效, 如果是电平触发中断, 该位被忽略。 注1: x = 0~7, n = 0~7. 注 2: 如果 Px.n 管脚选作掉电唤醒源, 用户应当在进入掉电模式前关闭去抖功能。这样可以避免唤醒后由 Px.n 管脚去抖功能导致的再次中断事件发生。

端口 0-7 中断模式控制 (Px_IMD)

寄存器	偏移量	R/W	描述	复位值
P0_IMD	GP_BA+0x018	R/W	P0 中断模式控制	0x0000_0000
P1_IMD	GP_BA+0x058	R/W	P1 中断模式控制	0x0000_0000
P2_IMD	GP_BA+0x098	R/W	P2 中断模式控制	0x0000_0000
P3_IMD	GP_BA+0x0D8	R/W	P3 中断模式控制	0x0000_0000
P4_IMD	GP_BA+0x118	R/W	P4 中断模式控制	0x0000_0000
P5_IMD	GP_BA+0x2D8	R/W	P5 中断模式控制	0xFFFF_XX00
P6_IMD	GP_BA+0x318	R/W	P6 中断模式控制	0xFFFF_XX00
P7_IMD	GP_BA+0x358	R/W	P7 中断模式控制	0xFFFF_XX00

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

Bits	描述	
[31:8]	Reserved	保留
[15:0]	IMD[n]	端口 0-7 Pin [n] 边沿或电平检测中断模式控制 IMD[n] 用于控制电平触发或边沿触发中断。若为边沿触发中断，触发源可由去抖动控制，如果是电平触发中断，输入源由一个HCLK时钟采样并产生中断。 0 = 边缘触发中断 1 = 电平触发中断 如果设置管脚为电平触发模式，则在寄存器Px_IEN中，只能设置一个电平；若设置了两个电平都触发中断，则设置被忽略，不会产生中断 去抖功能只对边缘触发有效，如果是电平触发，去抖功能被忽略。 注: x = 0~7, n = 0~7.

端口 0-7 中断使能控制 (Px_IEN)

寄存器	偏移量	R/W	描述	复位值
P0_IEN	GP_BA+0x01C	R/W	P0 中断使能控制	0x0000_0000
P1_IEN	GP_BA+0x05C	R/W	P1 中断使能控制	0x0000_0000
P2_IEN	GP_BA+0x09C	R/W	P2 中断使能控制	0x0000_0000
P3_IEN	GP_BA+0x0DC	R/W	P3 中断使能控制	0x0000_0000
P4_IEN	GP_BA+0x11C	R/W	P4 中断使能控制	0x0000_0000
P5_IEN	GP_BA+0x2DC	R/W	P5 中断使能控制	0x0000_0000
P6_IEN	GP_BA+0x31C	R/W	P6 中断使能控制	0x0000_0000
P7_IEN	GP_BA+0x35C	R/W	P7 中断使能控制	0x0000_0000

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

Bits	描述	
[31:24]	Reserved	保留.
[23:16]	IR_EN[n]	端口 0-7 Pin [n]输入上升沿或输入高电平中断使能 IR_EN[n] 用于使能相应Px.n管脚输入的中断. 置1也可以使能管脚唤醒功能. 当IR_EN[n] 位置 1 : 如果中断是电平触发(IMD[n] 为 1), Px.n管脚状态为高电平时将产生中断. 如果中断是边缘触发(IMD[n] 为 0), Px.n管脚状态由低到高跳转时将产生中断. 0 = 禁止Px.n高电平或由低电平到高电平变化的中断 1 = 使能Px.n高电平或由低电平到高电平变化的中断 注: x = 0~7, n = 0~7.
[15:8]	Reserved	保留.

[7:0]	IF_EN[n]	端口 0-7 Pin [n] 输入下降沿或输入低电平中断使能 IF_EN[n] 用于使能相应Px.n管脚输入的中断。置1也可以使能管脚唤醒功能 当 IF_EN[n] 位置 1 : 如果中断是电平触发(IMD[n] 为 1), Px.n管脚状态为低电平时将产生中断。 如果中断是边缘触发(IMD[n] 为 0), Px.n管脚状态由高到低跳转时将产生中断。 0 = 禁止Px.n低电平或由高电平到低电平变化的中断 1 = 使能Px.n低电平或由高电平到低电平变化的中断 注: x = 0~7, n = 0~7.
-------	----------	--

端口 0-7 中断源标志 (Px_ISRC)

寄存器	偏移量	R/W	描述	复位值
P0_ISRC	GP_BA+0x020	R/W	P0 中断源标志	0x0000_0000
P1_ISRC	GP_BA+0x060	R/W	P1 中断源标志	0x0000_0000
P2_ISRC	GP_BA+0x0A0	R/W	P2 中断源标志	0x0000_0000
P3_ISRC	GP_BA+0x0E0	R/W	P3 中断源标志	0x0000_0000
P4_ISRC	GP_BA+0x120	R/W	P4 中断源标志	0x0000_0000
P5_ISRC	GP_BA+0x2E0	R/WC	P5 中断源标志	0x0000_0000
P6_ISRC	GP_BA+0x320	R/WC	P6 中断源标志	0x0000_0000
P7_ISRC	GP_BA+0x360	R/WC	P7 中断源标志	0x0000_0000

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

Bits	描述	
[31:8]	Reserved	保留.
[7:0]	ISRC[n]	端口 0-7 Pin [n] 中断源标志 写： 0 = 无动作。 1 = 清相应挂起中断。 读： 0 = Px.n 没有中断 1 = Px.n 产生一个中断 注：x = 0~7, n = 0~7.

中断去抖动周期控制(DBNCECON)

寄存器	偏移量	R/W	描述	复位值
DBNCECON	GP_BA+0x180	R/W	中断去抖控制	0x0000_0020

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

Bits	描述	
[31:6]	Reserved	保留
[5]	ICLK_ON	中断时钟On 模式 0 = 边沿检测电路仅在 I/O管脚对应的Px_IEN位置 1有效。 1 =复位之后所有 I/O 边沿检测电路使能。 注: 如果没有相关的特定应用, 建议关掉时钟以减少耗电
[4]	DBCLKSRC	去抖动计数器时钟源选择 0 =去抖动计数器时钟源为 HCLK。 1 =去抖动计数器时钟源为内部 10 KHz 时钟。
[3:0]	DBCLKSEL	去抖动采样周期选择 0000 = 采样中断输入信号, 每 1 clocks一次. 0001 = 采样中断输入信号, 每 2 clocks一次. 0010 = 采样中断输入信号, 每 4 clocks一次. 0011 = 采样中断输入信号, 每 8 clocks一次. 0100 = 采样中断输入信号, 每 16 clocks一次. 0101 = 采样中断输入信号, 每 32 clocks一次. 0110 = 采样中断输入信号, 每 64 clocks一次. 0111 = 采样中断输入信号, 每 128 clocks一次. 1000 = 采样中断输入信号, 每 256 clocks一次. 1001 = 采样中断输入信号, 每 2*256 clocks一次. 1010 = 采样中断输入信号, 每 4*256 clocks一次.

		1011 = 采样中断输入信号, 每 8*256 clocks一次. 1100 = 采样中断输入信号, 每 16*256 clocks一次. 1101 = 采样中断输入信号, 每 32*256 clocks一次. 1110 = 采样中断输入信号, 每 64*256 clocks一次. 1111 = 采样中断输入信号, 每 128*256 clocks一次.
--	--	--

GPIO Px.n 管脚数据输入/输出 (Pxn_PDIO)

寄存器	偏移量	R/W	描述	复位值
P0n_PDIO n = 0,1...7	GP_BA+0x200 + 0x04 * n	R/W	GPIO P0.n Pin 数据输入/输出	0x0000_000X
P1n_PDIO n = 0,1...7	GP_BA+0x220 + 0x04 * n	R/W	GPIO P1.n Pin 数据输入/输出	0x0000_000X
P2n_PDIO n = 0,1...7	GP_BA+0x240 + 0x04 * n	R/W	GPIO P2.n Pin 数据输入/输出	0x0000_000X
P3n_PDIO n = 0,1...7	GP_BA+0x260 + 0x04 * n	R/W	GPIO P3.n Pin 数据输入/输出	0x0000_000X
P4n_PDIO n = 0,1...7	GP_BA+0x280 + 0x04 * n	R/W	GPIO P4.n Pin 数据输入/输出	0x0000_000X
P5n_PDIO	GP_BA+0x380 + 0x04 * n	R/W	GPIO P5.n Pin 数据输入/输出	0x0000_000X
P6n_PDIO	GP_BA+0x3A0 + 0x04 * n	R/W	GPIO P6.n Pin 数据输入/输出	0x0000_000X
P7n_PDIO	GP_BA+0x3C0 + 0x04 * n	R/W	GPIO P7.n Pin 数据输入/输出	0x0000_000X

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

Bits	描述	
[31:1]	Reserved	保留
[0]	Pxn_PDIO	GPIO Px.n Pin 数据输入/输出 写该位可以控制一个GPIO管脚的输出值 0 = 设置相应GPIO管脚为低 1 = 设置相应GPIO管脚为高

		读该寄存器得到GPIO管脚状态 例如: 写P00_PDIO即把值反映P0_DOUT[0]位上, 读P00_PDIO即读取P0_PIN[0]的值. 注1: x = 0~7, n = 0~7。 注 2: 写操作不受 Px_DMASK[n]影响。
--	--	--

6.6 定时器控制器(TMR)

6.6.1 概述

定时器控制器包含 4 组 32-位定时器，TIMER0~TIMER3，为用户提供便捷的计数定时功能。定时器可执行很多功能，如频率测量，时间延迟，时钟生成，外部输入管脚事件计数和外部捕捉管脚脉宽测量等。

6.6.2 特性

- 4 组 32-位定时器，带24位向上计数器和一个8位的预分频计数器
- 每个定时器都有独立的时钟源
- 提供 one-shot, periodic, toggle 和 continuous 共四种计数操作模式
- 超时周期 = (输入的定时器时钟周期) * (8-位预分频计数器 + 1) * (24-位 TCMP)
- 最大计数周期 = $(1 / T \text{ MHz}) * (2^8) * (2^{24})$ ，T 是定时器周期
- 通过 TDR（定时器数据寄存器）可读取内部 24 位向上计数器的值
- 支持事件计数功能，可用于计数外部管脚的事件(T0~T3)
- 通过TCAP（定时器捕捉数据寄存器）可读取 24 位捕捉计数器的值
- 支持外部管脚捕捉(T0EX~T3EX)，可用于脉宽测量
- 支持外部引脚捕捉(T0EX~T3EX)，可用于复位24位向上定时器
- 如果定时器中断信号产生，支持芯片从空闲/掉电模式唤醒
- 支持内部定时器触发模式

6.6.4 基本配置

寄存器APBCLK[5:2]用于使能定时器0~定时器3的外围时钟源，寄存器CLKSEL1[10:8](定时器0)，CLKSEL1[14:12](定时器1)，CLKSEL1[18:16](定时器2)，CLKSEL1[22:20](定时器3)用于选择不同的时钟源。

6.6.5 功能描述

6.6.5.1 定时器中断标志

定时器控制器支持两个中断标志：一个是TIF标志，该标志在定时器计数器值(TDR)与定时器比较值(TCMP)相匹配时置位，另一个是TEXIF标志，该标志在TxEX管脚的变化与TEX_EDGE的设置一致时置位。

6.6.5.2 定时器计数工作模式

定时器控制器提供四种定时计数模式：one-shot, periodic, toggle-output和continuous counting模式。

One-shot 模式

如果定时器工作在单周期 (one-shot) 模式(TCSR[28:27]为00)且 CEN (TCSR[30] 定时器使能位)置1，则定时器的计数器开始计数。一旦定时器计数器TDR的值达到定时器比较器寄存器 (TCMP) 的值时，TIF标志将变为1，TDR的值和CEN位将由定时器控制器清零，然后定时器计数操作停止。与此同时，如果 IE (TCSR[29] 中断使能位) 使能，定时器中断信号产生并送到 NVIC 通知 CPU。

Periodic 模式

如果定时器工作在周期 (periodic) 模式(TCSR[28:27]为01)且 CEN (TCSR[30] 定时器使能位)置1，则定时器的计数器开始计数。一旦定时器计数器TDR的值达到定时器比较器寄存器 (TCMP) 的值时，TIF标志将变为1，TDR的值将由定时器控制器清零，然后定时器重新计数。与此同时，如果 IE (TCSR[29] 中断使能位) 使能，则定时器中断信号产生并送到 NVIC 通知 CPU。在该模式，定时器控制器周期性计数并与TCMPR的值比较，直到 CEN位由软件清0。

Toggle-output 模式

如果定时器工作在触发输出 (toggle-out) 模式(TCSR[28:27]为10)且 CEN (TCSR[30] 定时器使能位)置1，则定时器的计数器开始计数。toggle-out 模式的计数操作大部分与周期模式是一样的，除了在TIF位置位时，toggle-out 模式会配有相关的T0~T3管脚来输出信号。即：管脚T0~T3上的触发输出信号为前后翻转输出，占空比为50%。

Continuous Counting 模式

如果定时器工作在连续计数 (continuous counting) 模式(TCSR[28:27]为11)且 CEN (TCSR[30] 定时器使能位)置1，则定时器的计数器开始计数。一旦定时器计数器(TDR)的值达到定时器比较器寄存器 (TCMP) 的值时，TIF标志将变为1，但TDR的值继续保持向上计数。与此同时，如果 IE (TCSR[29] 中断使能位) 使能，则定时器中断信号产生并送到 NVIC 通知 CPU。在该模式，用户可以立刻改变TCMP值，而不需要停止定时器计数和重新启动定时器计数。

例如，先把定时器比较寄存器 (TCMP)的值设置为 80。当定时器计数器的值 (TDR 的值) 达到 80

时, 定时器计数器继续计数, TDR的值也不会回到0, 而是继续计数, 从81, 82, 83, ... 到 $(2^{24}-1)$, 然后再从0, 1, 2, 3, ... 到 $2^{24}-1$, 如此往复。接着, 如果软件改变TCMPR的值为200并且清除TIF标志位, 当TDR的值达到200时, TIF标志将再次变为1。最后, 软件改变TCMPR的值为500并且清除TIF标志, 当TDR的值达到500时, TIF标志将再次变为1

在该模式, 计数器计数为连续的。所以该操作模式叫做连续计数模式。

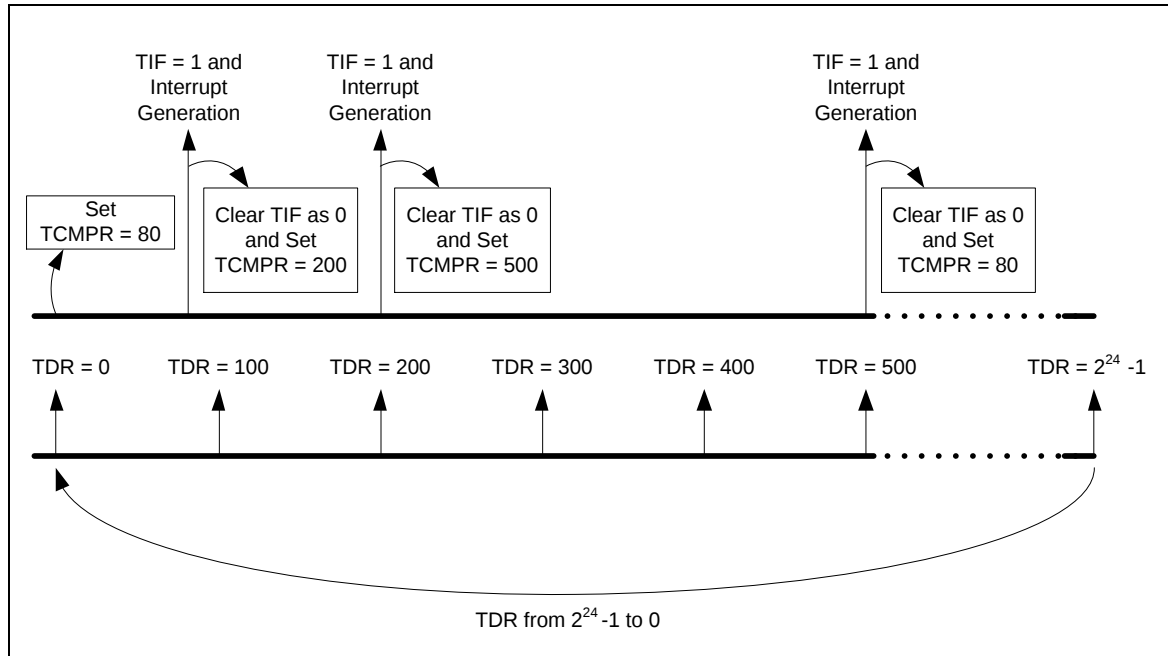


图 6.6-3 Continuous Counting 连续计数模式

6.6.5.3 事件计数模式

定时器控制器还提供这样的应用, 能对输入事件(来自管脚 $TMx\ x=0\sim3$)计数并将事件的次数反映到TDR值, 这可以称为事件计数功能。要使用该功能, CTB(TCSR[24])位需置位并且定时器外设时钟源必须设为HCLK。

软件可以通过TCDB(TEXCON[7])位来使能或关闭Tx管脚消抖电路。如果Tx管脚消抖电路关闭, 输入事件频率必须少于 $1/3HCLK$, 如果消抖电路打开, 输入事件的频率须小于 $1/8HCLK$, 以保证TDR的值是正确的。软件也可以通过设置TX_PHASE(TEXCON[0])来选择边沿检测Tx管脚的相位。

在事件计数模式, 定时器计数操作模式可以设置为单次, 周期, 和连续计数模式对来自Tx管脚的输入事件进行计数的TDR值。

6.6.5.4 事件捕捉模式

当检测到TxEX管脚 ($x=0\sim3$) 上有边沿变化时, 事件捕捉功能可将TDR值捕捉到TCAP中。在这种模式下, RSTCAPSEL (TEXCON[4])应设置为0用于将TxEX管脚选择用于事件捕捉功能, 定时器外围时钟源也应该设置为HCLK。

软件可以通过TEXDB (TEXCON[6])位来使能或关闭TxEX管脚消抖电路。如果TxEX管脚的消抖电路关闭, 输入事件频率必须少于 $1/3HCLK$, 如果消抖电路打开, 输入事件的频率须小于 $1/8HCLK$,

以保证捕捉功能可以工作正常。软件也可以通过设置TEX_EDGE (TEXCON[2:1])来选择边沿检测TxEX管脚的相位。

6.6.5.5 Event Reset Counter Mode 事件复位计数模式

当检测到TxEX管脚(x=0~3)有边沿转变时，定时器同样提供事件复位计数器功能来复位TDR的值。在该模式，大部分设置与事件捕捉功能相同，除了RSTCAPSEL(TEXCON[4])位必须设置为1来选择TxEX转变时用作事件复位计数器。

6.6.5.6 内部定时器触发捕捉模式

在这种模式下，Timer0/2会被强制设置在计数模式，用于对外部事件计数，并产生内部信号(INTR_TMR_TRG)用于触发Timer1/3计数启动或停止。同样，Timer1/3会被强制设置在捕捉模式并根据Timer0/2计数器状态来启动/停止触发计数

如果使能Timer0内部定时器触发捕捉，Timer1将被强制在触发计数捕捉功能。如果使能Timer2内部定时器触发捕捉，Timer3将被强制在触发计数捕捉功能。

启动触发

当置位Timer0/2的INTR_TRG_EN位时，如果Timer0/2 24-位计数器(TDR)计数值从0x0 计数到0x1，INTR_TMR_TRG将发生一个上升沿改变，Timer1/3计数器将立刻自动启动计数。

停止触发

当Timer0/2 TDR到达Timer0/2 TCMR中值时，Timer0/2将使INTR_TMR_TRG发生下降沿改变。然后Timer0/2计数器模式功能将被禁止，INTR_TRG_EN 位会由硬件清零，而Timer1/3也会停止计数。同时，Timer1/3 TDR会被保存到Timer1/3 TCAP寄存器中。

用户可以使用内部定时器触发模式更精准地测量外部事件(Tx)周期。下图为设置Timer0工作在事件计数模式，Timer1工作在触发计数捕捉模式时内部定时器触发捕捉模式的范例流程。

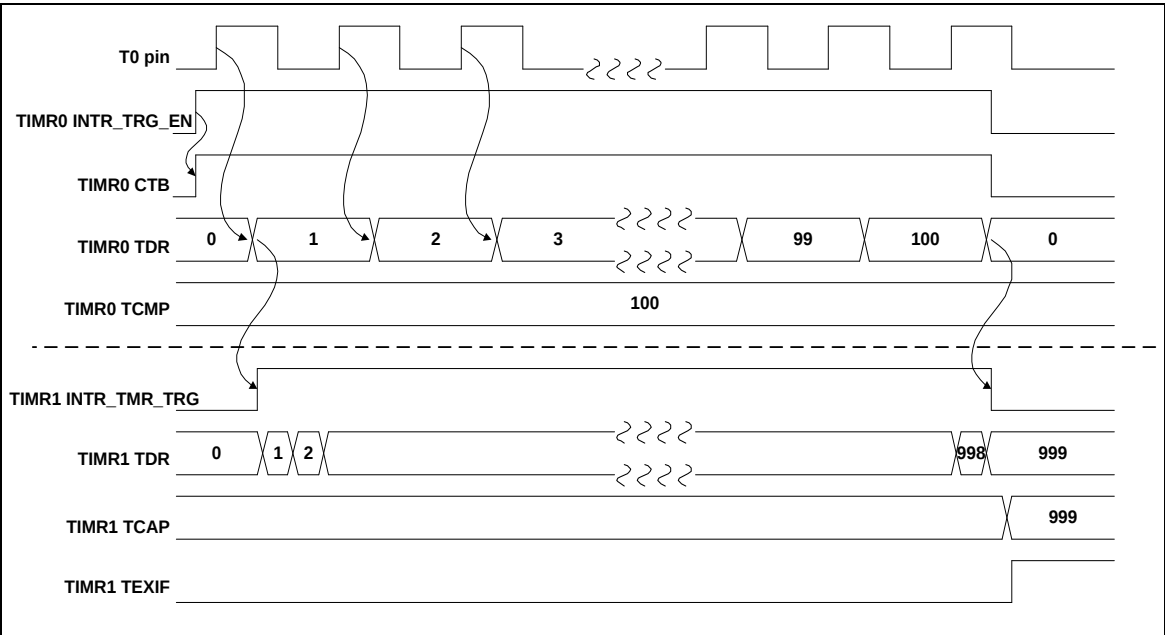


图 6.6-4 内部-定时器触发捕捉时序

6.6.6 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移地址	R/W	描述	复位值
TIMER 基地址:				
TMR01_BA = 0x4001_0000				
TMR23_BA = 0x4011_0000				
TCSR0	TMR01_BA+0x00	R/W	Timer0 控制和状态寄存器	0x0000_0005
TCMPR0	TMR01_BA+0x04	R/W	Timer0 比较寄存器	0x0000_0000
TISR0	TMR01_BA+0x08	R/W	Timer0 中断状态寄存器	0x0000_0000
TDR0	TMR01_BA+0x0C	R	Timer0 数据寄存器	0x0000_0000
TCAP0	TMR01_BA+0x10	R	Timer0 捕捉数据寄存器	0x0000_0000
TEXCON0	TMR01_BA+0x14	R/W	Timer0 外部控制寄存器	0x0000_0000
TEXISR0	TMR01_BA+0x18	R/W	Timer0 外部中断状态寄存器	0x0000_0000
TCSR1	TMR01_BA+0x20	R/W	Timer1 控制和状态寄存器	0x0000_0005
TCMPR1	TMR01_BA+0x24	R/W	Timer1 比较寄存器	0x0000_0000
TISR1	TMR01_BA+0x28	R/W	Timer1 中断状态寄存器	0x0000_0000
TDR1	TMR01_BA+0x2C	R	Timer1 数据寄存器	0x0000_0000
TCAP1	TMR01_BA+0x30	R	Timer1 捕捉数据寄存器	0x0000_0000
TEXCON1	TMR01_BA+0x34	R/W	Timer1 外部控制寄存器	0x0000_0000
TEXISR1	TMR01_BA+0x38	R/W	Timer1 外部中断状态寄存器	0x0000_0000
TCSR2	TMR23_BA+0x00	R/W	Timer2 控制和状态寄存器	0x0000_0005
TCMPR2	TMR23_BA+0x04	R/W	Timer2 比较寄存器	0x0000_0000
TISR2	TMR23_BA+0x08	R/W	Timer2 中断和状态寄存器	0x0000_0000
TDR2	TMR23_BA+0x0C	R	Timer2 数据寄存器	0x0000_0000
TCAP2	TMR23_BA+0x10	R	Timer2 捕捉数据寄存器	0x0000_0000
TEXCON2	TMR23_BA+0x14	R/W	Timer2 外部控制寄存器	0x0000_0000
TEXISR2	TMR23_BA+0x18	R/W	Timer2 外部中断状态寄存器	0x0000_0000
TCSR3	TMR23_BA+0x20	R/W	Timer3 控制和状态寄存器	0x0000_0005
TCMPR3	TMR23_BA+0x24	R/W	Timer3 比较寄存器	0x0000_0000
TISR3	TMR23_BA+0x28	R/W	Timer3 中断状态寄存器	0x0000_0000
TDR3	TMR23_BA+0x2C	R	Timer3 数据寄存器	0x0000_0000
TCAP3	TMR23_BA+0x30	R	Timer3 捕捉数据寄存器	0x0000_0000

TEXCON3	TMR23_BA+0x34	R/W	Timer3 外部控制寄存器	0x0000_0000
TEXISR3	TMR23_BA+0x38	R/W	Timer3 外部中断状态寄存器	0x0000_0000

6.6.7 寄存器描述

定时器控制和状态寄存器(TCSR)

寄存器	偏移地址	R/W	描述	复位值
TCSR0	TMR01_BA+0x00	R/W	Timer0 控制和状态寄存器	0x0000_0005
TCSR1	TMR01_BA+0x20	R/W	Timer1 控制和状态寄存器	0x0000_0005
TCSR2	TMR23_BA+0x00	R/W	Timer2 控制和状态寄存器	0x0000_0005
TCSR3	TMR23_BA+0x20	R/W	Timer3 控制和状态寄存器	0x0000_0005

31	30	29	28	27	26	25	24
DBGACK_TMR	CEN	IE	MODE[1:0]		CRST	CACT	CTB
23	22	21	20	19	18	17	16
WAKE_EN	Reserved	TOUT_SEL	PERIODIC_SEL	INTR_TRG_EN	Reserved		TDR_EN
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PRESCALE[7:0]							

位	描述	
[31]	DBGACK_TMR	仿真器 (ICE) 调试模式响应禁止位 (写保护位) 0 = 仿真器调试模式响应影响定时器计数。 当调试器调试模式响应时, 定时器计数器将被固定住。 1 = 仿真器调试模式响应禁用。 无论调试器调试模式响应与否, 定时器计数器将持续计数下去。
[30]	CEN	定时器使能位 0 = 停止/暂停计数 1 = 开始计数 注意1: 在停止状态, 设置 CEN 为 1 将使能 24-位向上计数器从上次停止的计数值继续计数。 注意2: 在 one-shot 模式下 (TCSR [28:27] =00), 当相应的定时中断标志(TIF)产生时, 该位由硬件自动清零。
[29]	IE	中断使能位 0 = 禁用定时器中断 1 = 使能定时器中断 如果该位使能, 当定时器中断标志 (TIF) 置 1, 定时器中断信号产生并通知CPU。
[28:27]	MODE	定时器操作模式

		00 = 定时器工作在单触发模式 (one-shot) 01 = 定时器工作在周期模式 (period) 10 = 定时器工作在Toggle-output 模式 11 = 定时器工作在连续计数模式(Continuous Counting)
[26]	CRST	定时器复位 0 = 该位写 0 无效 1 = 如果CACT为1，复位定时器的8-位预分频计数器，内部24-位向上计数器的值和CEN位
[25]	CACT	定时器激活状态位（只读） 该位表示当前定时器计数器的状态。 0 = 24-位定时器计数器未激活 1 = 24-位定时器计数器激活
[24]	CTB	计数模式使能位 该位用来使能外部计数管脚功能。当定时器用作事件计数，该位需要设置成1，并选择HCLK 作为定时器时钟源。详情请参看6.6.5.3 节 0 = 禁用事件计数器模式 1 = 使能事件计数器模式
[23]	WAKE_EN	唤醒功能使能 0 = 唤醒触发事件禁止 1 = 唤醒触发事件使能
[22]	Reserved	Reserved.
[21]	TOUT_SEL	翻转输出管脚选择 0 = 翻转输出管脚为Tx管脚 1 = 翻转输出管脚为TxEx管脚
[20]	PERIODIC_SEL	周期模式行为选择使能 0=周期模式行为选择禁止。 定时器工作在周期模式时，如果用户更新TCMP，TDR会被复位为默认值。 1=周期模式行为选择使能。 定时器工作在周期模式时，如果用户更新TCMP，限制如下： 如果更新的TCMP > TDR，TCMP将被更新，TDR保持连续计数 如果更新的TCMP=TDR，定时器时间溢出中断将立即发生 如果更新的TCMP<TDR，TDR会被复位为默认值
[19]	INTR_TRG_EN	内部-定时器触发模式使能 设置该位将会使能内部-定时器捕捉功能 Timer0/2将工作在计数模式，用于对外部时钟和事件计数 同样，Timer1/3将工作在触发计数模式用于捕捉。 0 = 内部-定时器触发捕捉模式禁止 1 = 内部-定时器触发捕捉模式使能

		注：对于Timer1/3，该位可以忽略，读回值一直为0
[18:17]	Reserved	Reserved.
[16]	TDR_EN	数据加载使能 当该位置1，计数器正在计数时，定时器计数器值(TDR) 会被内部24位向上计数器持续更新。 0 = 定时器数据寄存器更新禁止 1 = 当定时器计数器激活， 定时器数据寄存器更新使能
[15:8]	Reserved	Reserved.
[7:0]	PRESCALE	预分频计数器 时钟输入根据 PRESCALE +1 进行预分频。如果 PRESCALE 数值为0，不进行预分频。

定时器比较寄存器 (TCMPR)

寄存器	偏移地址	R/W	描述	复位值
TCMPR0	TMR01_BA+0x04	R/W	Timer0 比较寄存器	0x0000_0000
TCMPR1	TMR01_BA+0x24	R/W	Timer1 比较寄存器	0x0000_0000
TCMPR2	TMR23_BA+0x04	R/W	Timer2 比较寄存器	0x0000_0000
TCMPR3	TMR23_BA+0x24	R/W	Timer3 比较寄存器	0x0000_0000

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

位	描述	
[31:24]	Reserved	保留.
[23:0]	TCMP	定时器比较值 TCMP 是一个24-位比较寄存器。当内部 24-位向上计数器的值与 TCMP 的值相等时，TIF标志将置1。 超时周期 = (定时器时钟输入周期) * (8-bit PRESCALE + 1) * (24-bit TCMP) 注意1: 不能向 TCMP 里写 0x0 或 0x1，否则内核将运行到未知状态。 注意 2: 当定时器工作在 continuous counting 模式时，如果软件写一个新的值到 TCMP，24-位向上计数定时器将继续计数。如果定时器工作在其他模式，如果软件写一个新的值到 TCMP，定时器将使用新比较值并退出当前计数，开始重新计数。

定时器中断状态寄存器 (TISR)

寄存器	偏移地址	R/W	描述	复位值
TISR0	TMR01_BA+0x08	R/W	Timer0 中断状态寄存器	0x0000_0000
TISR1	TMR01_BA+0x28	R/W	Timer1 中断状态寄存器	0x0000_0000
TISR2	TMR23_BA+0x08	R/W	Timer2 中断状态寄存器	0x0000_0000
TISR3	TMR23_BA+0x28	R/W	Timer3 中断状态寄存器	0x0000_0000

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

位	描述	
[31:2]	Reserved	Reserved.
[1]	TWF	定时器唤醒标志 该位表示定时器中断唤醒标志状态。 0 = 定时器不会引起CPU 唤醒 1 =如果定时器中断信号产生， CPU 从空闲或掉电模式唤醒 注:该位必须通过软件写1清0.
[0]	TIF	定时器中断标志 该位提示当TDR值达到TCMP值时, 定时器中断标志状态。 0 = 无影响 1 =TDR值达到TCMP值 注:该位必须通过软件写1清0.

定时器数据寄存器 (TDR)

寄存器	偏移地址	R/W	描述	复位值
TDR0	TMR01_BA+0x0C	R	Timer0 数据寄存器	0x0000_0000
TDR1	TMR01_BA+0x2C	R	Timer1 数据寄存器	0x0000_0000
TDR2	TMR23_BA+0x0C	R	Timer2 数据寄存器	0x0000_0000
TDR3	TMR23_BA+0x2C	R	Timer3 数据寄存器	0x0000_0000

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

位	描述	
[31:24]	Reserved	Reserved.
[23:0]	TDR	定时器数据寄存器 如果 TDR_EN (TCSR[16]) 置 1, 定时器计数器值(TDR) 会不断被更新, 用来监视内部24位向上计数器值。

定时器捕捉数据寄存器 (TCAP)

寄存器	偏移地址	R/W	描述	复位值
TCAP0	TMR01_BA+0x10	R	Timer0 捕捉数据寄存器	0x0000_0000
TCAP1	TMR01_BA+0x30	R	Timer1 捕捉数据寄存器	0x0000_0000
TCAP2	TMR23_BA+0x10	R	Timer2 捕捉数据寄存器	0x0000_0000
TCAP3	TMR23_BA+0x30	R	Timer3 捕捉数据寄存器	0x0000_0000

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

位	描述	
[31:24]	Reserved	Reserved.
[23:0]	TCAP	定时器捕捉数据寄存器 当 TEXIF置1时，当前TDR的值将立刻自动被载入到 TCAP中。

定时器外部控制寄存器 (TEXCON)

寄存器	偏移地址	R/W	描述	复位值
TEXCON0	TMR01_BA+0x14	R/W	Timer0 外部控制寄存器	0x0000_0000
TEXCON1	TMR01_BA+0x34	R/W	Timer1 外部控制寄存器	0x0000_0000
TEXCON2	TMR23_BA+0x14	R/W	Timer2 外部控制寄存器	0x0000_0000
TEXCON3	TMR23_BA+0x34	R/W	Timer3 外部控制寄存器	0x0000_0000

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

位	描述	
[31:8]	Reserved	Reserved.
[7]	TCDB	定时器外部计数器输入管脚防抖动使能位 0 = Tx管脚禁用防抖动 1 = Tx管脚使能防抖动 如果该位使能, Tx管脚边沿检测带防抖动电路。
[6]	TEXDB	定时器外部捕捉输入管脚防抖动使能位 0 = TxEX 管脚禁用防抖动 1 = TxEX 管脚使能防抖动 如果该位使能, TxEX 管脚边沿检测带防抖动电路。
[5]	TEXIEN	定时器外部捕捉中断使能位 0 = TxEX 管脚检测中断禁止. 1 = TxEX 管脚检测中断使能 如果TEXIEN 使能, 当TEXIF 标志设为1时, 定时器会产生一个外部捕捉中断信号并通知CPU发生了中断。.
[4]	RSTCAPSEL	定时器外部复位计数器/ 定时器外部捕捉模式选择 0 = 当TEXIF 被置为1时, 如果TxEX 管脚上电平发生变化, TDR值将会保存到TCAP中 1 = TxE管脚上电平发生变化将复位24位向上计数器
[3]	TEXEN	定时器外部管脚功能使能位 该位使能TxEX管脚上的RSTCAPSEL功能。

		0 = TxEX 管脚上的RSTCAPSEL功能忽略。 1 = TxEX 管脚上的RSTCAPSEL功能激活。
[2:1]	TEX_EDGE	定时器外部捕捉管脚边沿检测选择 00 = TxEX管脚上 发生1 到 0的转变将被检测。 01 = TxEX管脚上 发生0 到 1的转变将被检测。 10 = TxEX管脚上 发生1 到 0 或 0 到 1的转变都将被检测。 11 = 保留。
[0]	TX_PHASE	定时器外部计数管脚相位检测选择 该位表示Tx管脚上的相位检测。 0 = TxEX管脚上的下降沿将被统计。 1 = TxEX管脚上的上升沿将被统计。

定时器外部中断状态寄存器(TEXISR)

寄存器	偏移地址	R/W	描述	复位值
TEXISR0	TMR01_BA+0x18	R/W	Timer0 外部中断状态寄存器	0x0000_0000
TEXISR1	TMR01_BA+0x38	R/W	Timer1 外部中断状态寄存器	0x0000_0000
TEXISR2	TMR23_BA+0x18	R/W	Timer2 外部中断状态寄存器	0x0000_0000
TEXISR3	TMR23_BA+0x38	R/W	Timer3 外部中断状态寄存器	0x0000_0000

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

位	描述	
[31:1]	Reserved	Reserved.
[0]	TEXIF	定时器外部中断标志位 该位表示外部捕捉中断标志状态。. 当TEXEN使能，且TxEX管脚选择为外部捕捉功能，在TxEX管脚上的边沿转变与TEX_EDGE 设定相匹配时，该标志将由硬件置位。 0 = 没有TxEX管脚中断发生 1 = 有TxEX管脚中断发生 注意：该位写 1 清零。

6.7 PWM 发生器和捕捉定时器

6.7.1 概述

NuMicro M058S有1套PWM，共支持2个PWM生成器，可配置为4个独立的PWM输出：

PWM0~PWM3，或配置成2对互补PWM:(PWM0, PWM1)和(PWM2, PWM3)，具有可编程的死区设定。

每个PWM发生器包括一个8位预分频器，一个时钟除频器（提供5级时钟除频(1, 1/2, 1/4, 1/8, 1/16)），两个PWM定时器（有两个时钟选择），两个16位PWM计数器（用于PWM周期控制），两个16位比较器（用于PWM占空比控制）以及一个死区发生器。2对PWM生成器共提供4个独立的PWM中断标志，这些中断标志在对应PWM向下计数达到零时置位（由硬件来完成），每个PWM中断源都有独立的使能位。PWM发生器可以配置为单触发模式用于产生单个PWM周期信号或自动重载模式用于产生连续PWM波形。

当DZEN01 (PCR[4]) 被置位，PWM0与PWM1执行互补PWM功能；这一对PWM的周期，占空比和死区时间由PWM0 定时器和死区发生器0决定。同样，PWM互补对(PWM2, PWM3)由PWM2、定时器和死区发生器2控制。关于PWM定时器内部结构，参考下图

为防止PWM 输出不稳定波形，16位向下计数计数器和16位比较器采用双缓存。当用户向计数器/比较器寄存器写入新的值时，只有当计数器下数到0后，新写入的值才会被重新加载到计数器/比较器。双缓存特性可以避免PWM输出毛刺。

当16位向下计数计数器达到0时，中断请求产生。如果PWM定时器被定义为自动装载模式，当向下计数器达到0时，自动重新加载PWM计数寄存器(CNRx)的值，然后开始递减，之后会不断重复进行。如果定时器设为单次触发模式，向下计数器达到0时将停止计数，并产生一个中断请求。

PWM 计数比较器的值用于调制高脉冲的宽度，在下数计数器的值等于比较寄存器的值时，计数器控制逻辑将改变，使得PWM输出变成高电平。

PWM定时器的另一个功能是数字输入捕捉功能。当捕捉功能使能时，PWM输出引脚被切为输入捕捉模式。捕捉器0和PWM0 共享PWM0 定时器，捕捉器1和PWM1 共享PWM1 定时器，以此类推。因此，在使能捕捉功能之前，用户必须先设置PWM定时器。捕捉功能使能后，当输入端口有上升沿时转变时，PWM 计数器的值会被锁存入CRLR，输入端口有下降沿转变时，PWM 计数器的值会被锁存入CFLR。通过软件设定CRL_IE0(CCR0[1])(使能上升沿锁存中断)和CFL_IE0(CCR0[2])(使能下降沿锁存中断)，可以设置捕捉器通道0产生中断的条件。同样设定CRL_IE1(CCR0[17])和CFL_IE1(CCR0[18])，可以设定捕捉器通道1产生中断的条件。通过设定CCR2的相应的数据位，捕捉通道2和3也有同样的功能。对每组而言，每当捕捉器触发中断0/1/2/3时，PWM计数器0/1/2/3的值也会同时被重新加载。

PWM可以支持的最大的捕捉频率由捕捉中断延迟决定。捕捉中断发生时，软件至少执行以下三步：第一步，读PIIR 获取中断源，第二步，读CRLRx/CFLRx(x=0~3) 获取捕捉的值，第三步写1清PIIR 为0。如果中断到完成的延时时间为T0，捕捉信号在 (T0) 间隔内一定不能改变。此条件下，

最大捕捉频率为1/T0.

6.7.2 特性

PWM功能:

PWM模块共有2个PWM生成器。每个PWM生成器都有一个8-位预分频器，一个时钟除频器，2个PWM-定时器（向下计数），一个死区发生器和2个PWM输出

- PWMA (PWM group A) 为一组PWM，共支持4个PWM通道或2对互补PWM通道
- PWM 组有两个PWM发生器。每个PWM支持8-位预分频器，两个时钟除频器，两个 PWM 定时器（向下计数），一个死区发生器和两个PWM 输出。
- 最高16位分辨率
- 单次或自动装载模式
- 边沿对齐或中心对齐两种选择
- PWM触发ADC转换功能

捕捉功能:

- 与PWM发生器共享时序逻辑控制
- 支持4个捕捉输入通道，与4个PWM输出通道共享
- 每个通道支持1个上升沿锁存寄存器(CRLRx)，一个下降沿锁存寄存器(CFLRx) 和捕捉中断标志(CAPIFx)

6.7.3 框图

下图举例说明PWM成对工作时的结构(例如：PWM-Timer 0&1 为一对，PWM-Timer 2&3 为一对)。

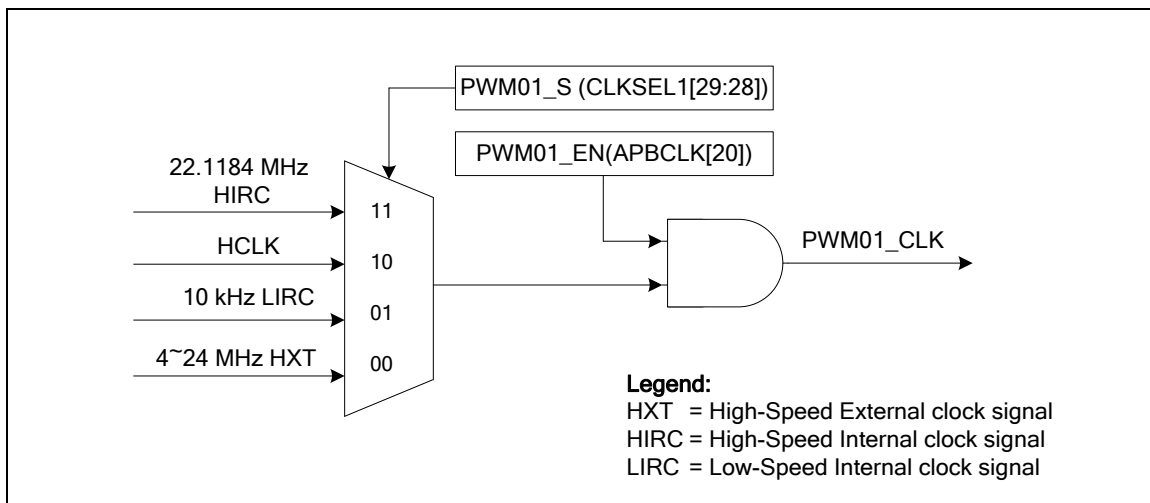


图 6.7-1 PWM 发生器 0 时钟源控制

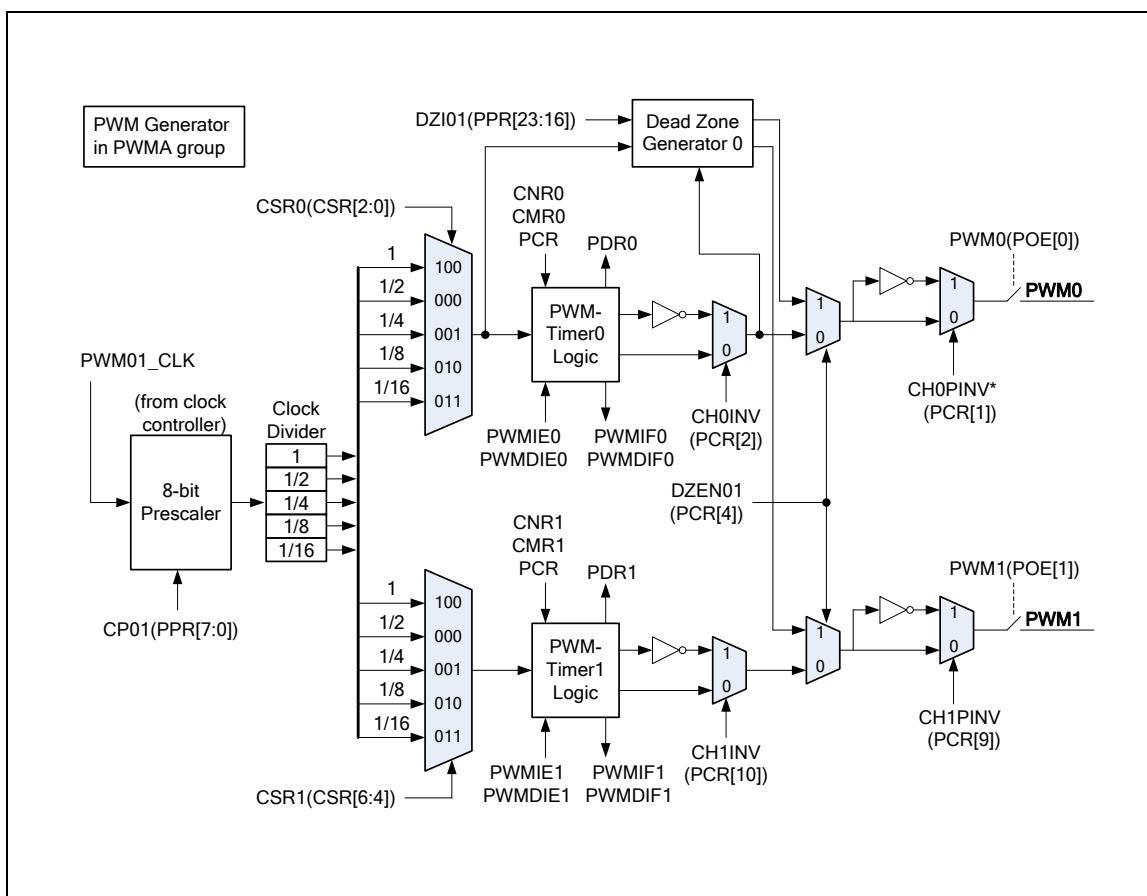


图 6.7-2 PWM 发生器 0 结构框图

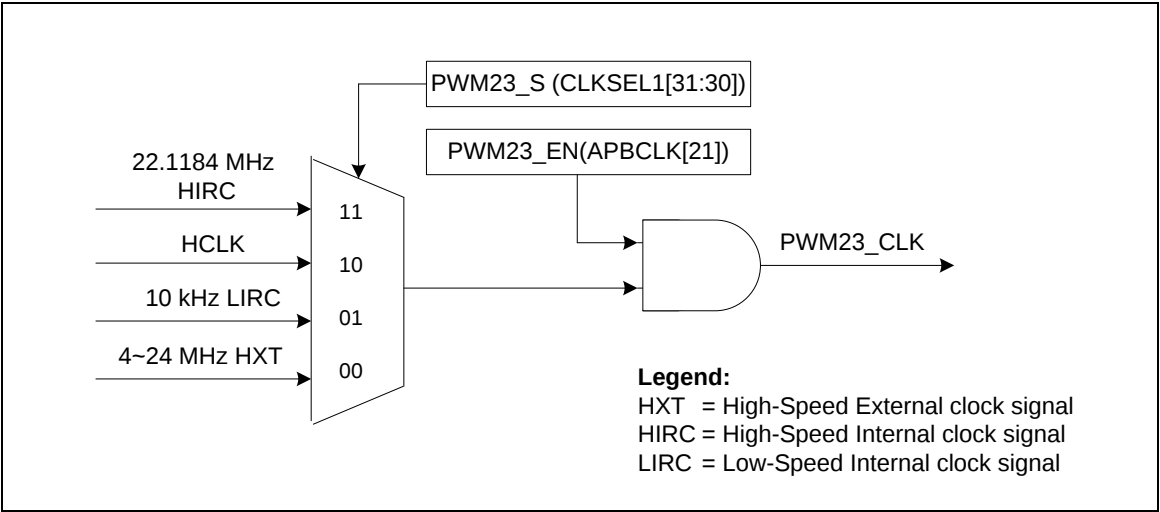


图 6.7-3 PWM 生成器 2 时钟源控制

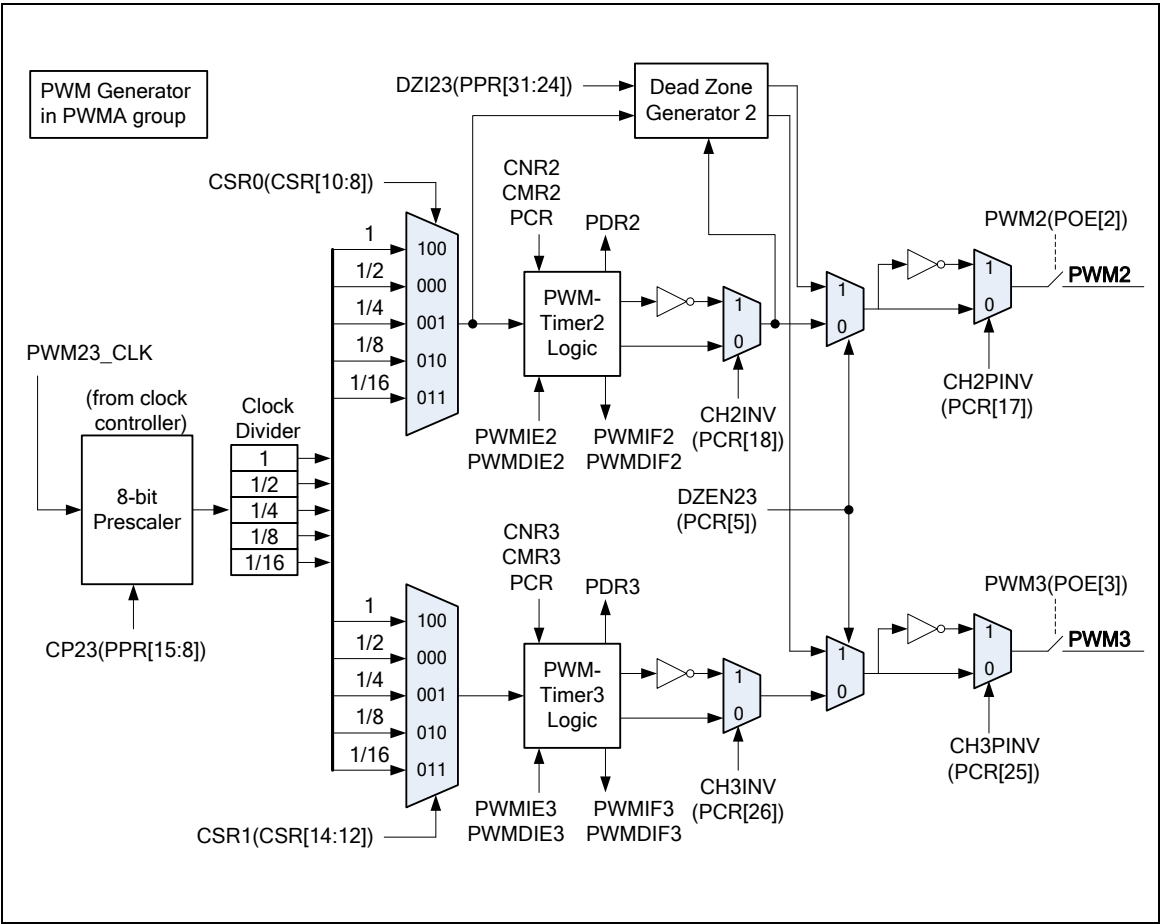


图 6.7-4 PWM 发生器 2 结构框图

6.7.4 基本设定

PWM引脚功能在寄存器P2_MFP 和 P4_MFP中设定。

PWM 时钟使能由APBCLK[21:20]设定。PWM时钟源由CLKSEL1[31:28]选择。

6.7.5 功能描述

6.7.5.1 PWM-定时器操作

PWM 控制器支持两种操作模式: 边沿对齐和中心对齐

6.7.5.2 边沿对齐 PWM (向下计数)

边沿对齐PWM 输出模式下, 16位PWM计数器从CNRx开始向下计数, 当等于CMRx(旧)时, PWM 发生器将输出低电平。计数器继续下数直到0, 同时PWM发生器反转输出高电平。如果 CHxMODE=1, CMRx(新)和CNRx(新)将被更新, 如果PWM中断被使能(PIERx = 1), PWM将产生中断

PWM 周期和占空比由向下计数的PWM寄存器(CNR)以及PWM比较寄存器(CMR) 控制。PWM定时器工作时序如下图所示. 脉宽宽度调制的公式如下, PWM定时器比较器图如图所示

注意: 在PWM 功能使能前, PWM通道相应的GPIO管脚应配置成PWM功能(使能POE并禁止CAPENR).

- PWM 频率 = $\text{PWM}_{xy_CLK} / [(\text{prescale} + 1) * (\text{clock divider}) * (\text{CNR} + 1)]$; XY代表01, 23, 取决于所选择的PWM通道
- 占空比 = $(\text{CMR} + 1) / (\text{CNR} + 1)$
- $\text{CMR} \geq \text{CNR}$: PWM 输出高
- $\text{CMR} < \text{CNR}$: PWM 低脉宽 = $(\text{CNR} - \text{CMR}) \text{ unit}^{[1]}$; PWM高脉宽 = $(\text{CMR} + 1) \text{ unit}$
- $\text{CMR} = 0$: PWM 低脉宽 = $(\text{CNR}) \text{ unit}$; PWM 高脉宽 = 1 unit

注意[1]: unit = 一个 PWM 时钟周期

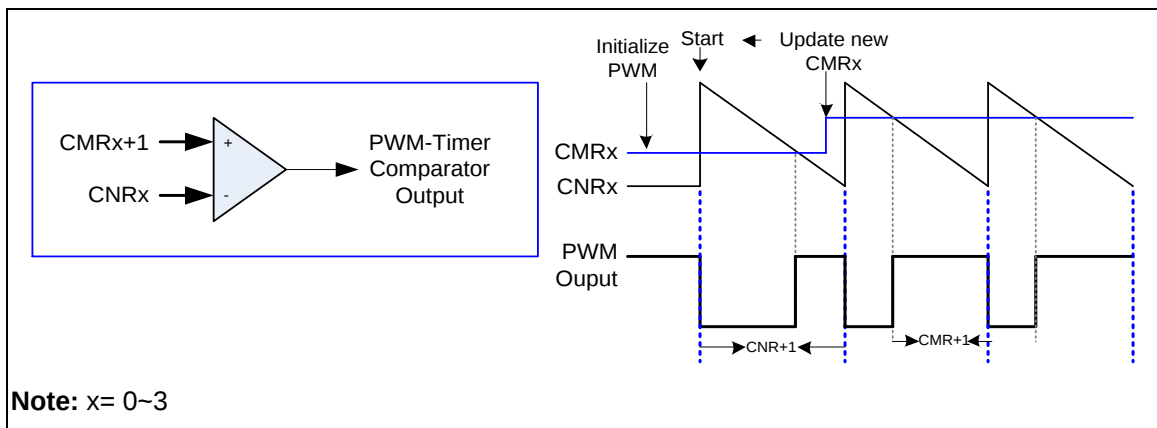


图 6.7-5 PWM-定时器内部比较器输出图例

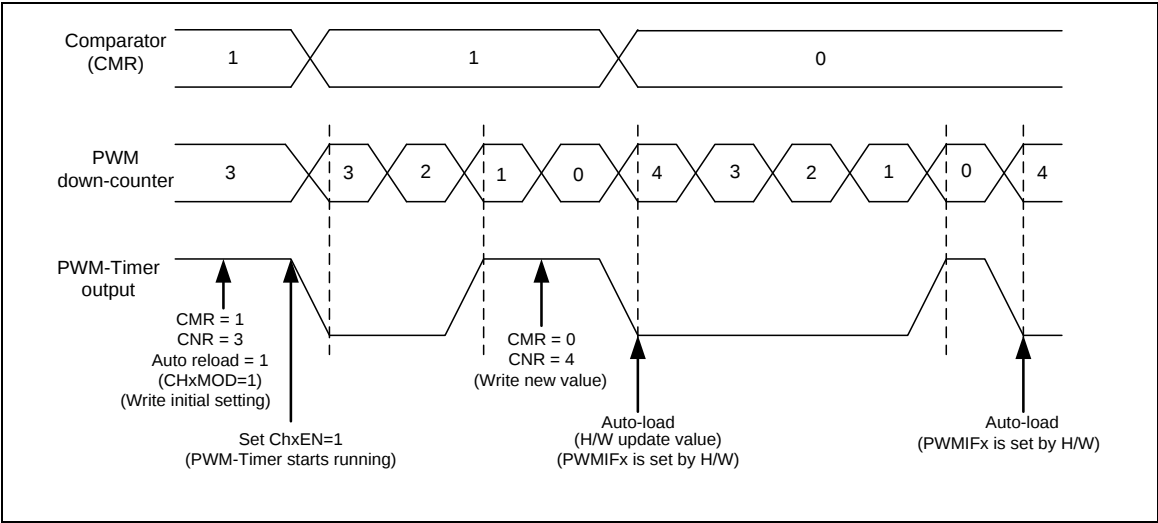


图 6.7-6 PWM 定时器操作时序

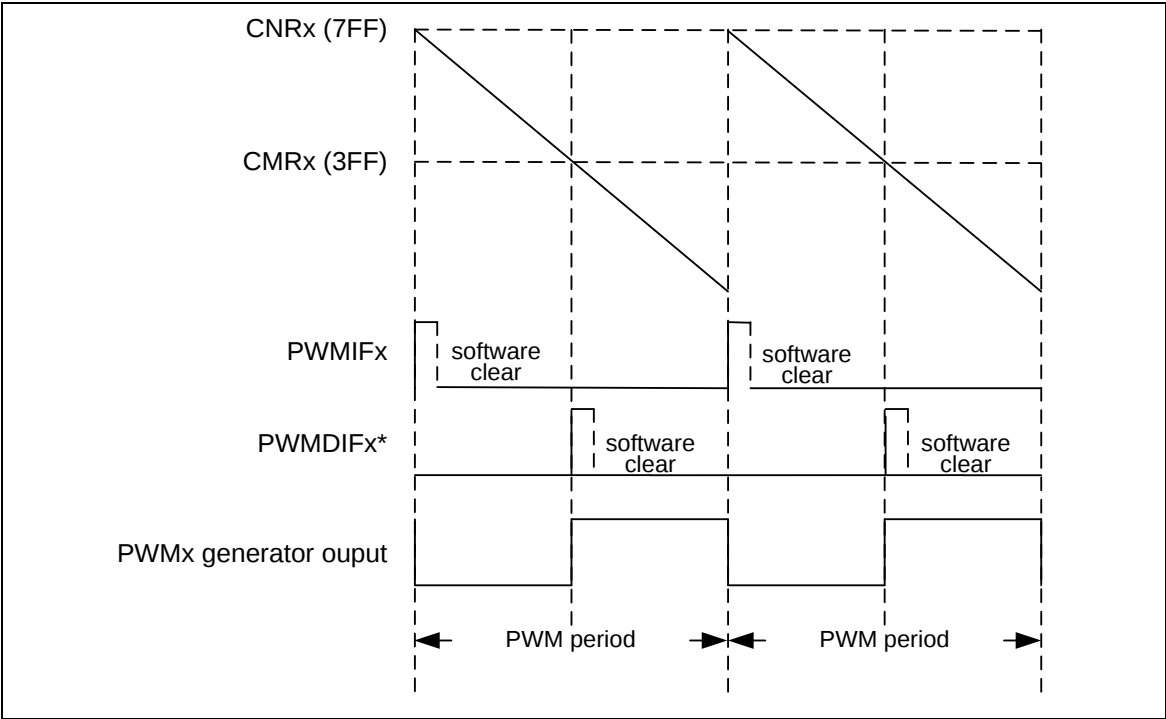


图 6.7-7 PWM 周期中断发生时序波形图

6.7.5.3 中心对齐PWM (向上向下-计数)

当PWM 定时器配置为向上计数/向下计数的计数器模式时，模块将产生中心对齐PWM信号。PWM 计数器从0开始向上计数，直到等于CMRx (旧)的值，此时触发PWMx发生器使其输出低电平。计数器继续计数直到CNRx (旧)，之后计数器开始自动向下计数，当再次等于CMRx (旧)时,PWMx 发生器输出为高电平。一旦PWM计数器下溢，如果CHxMODE = 1; PWMx 周期寄存器CNRx (新)和占空比寄存器CMRx (新) 将被更新。

- PWM 频率= $PWM_{xy_CLK} / [(prescale+1) * (clock\ divider) * (CNR+1)]$; xy,代表 01或23, 取决于所选的通道
- 占空比 = $[(2 \times CMR) + 1] / [2 \times (CNR+1)]$
- $CMR > CNR$: PWM 输出高
- $CMR \leq CNR$: PWM 低脉宽= $2 \times (CNR-CMR) + 1 \text{ unit}^{[1]}$; PWM 高脉宽 = $(2 \times CMR) + 1 \text{ unit}$
- $CMR = 0$: PWM 低脉宽 = $2 \times CNR + 1 \text{ unit}$; PWM 高脉宽 = 1 unit

注意[1]: unit = 一个PWM时钟周期

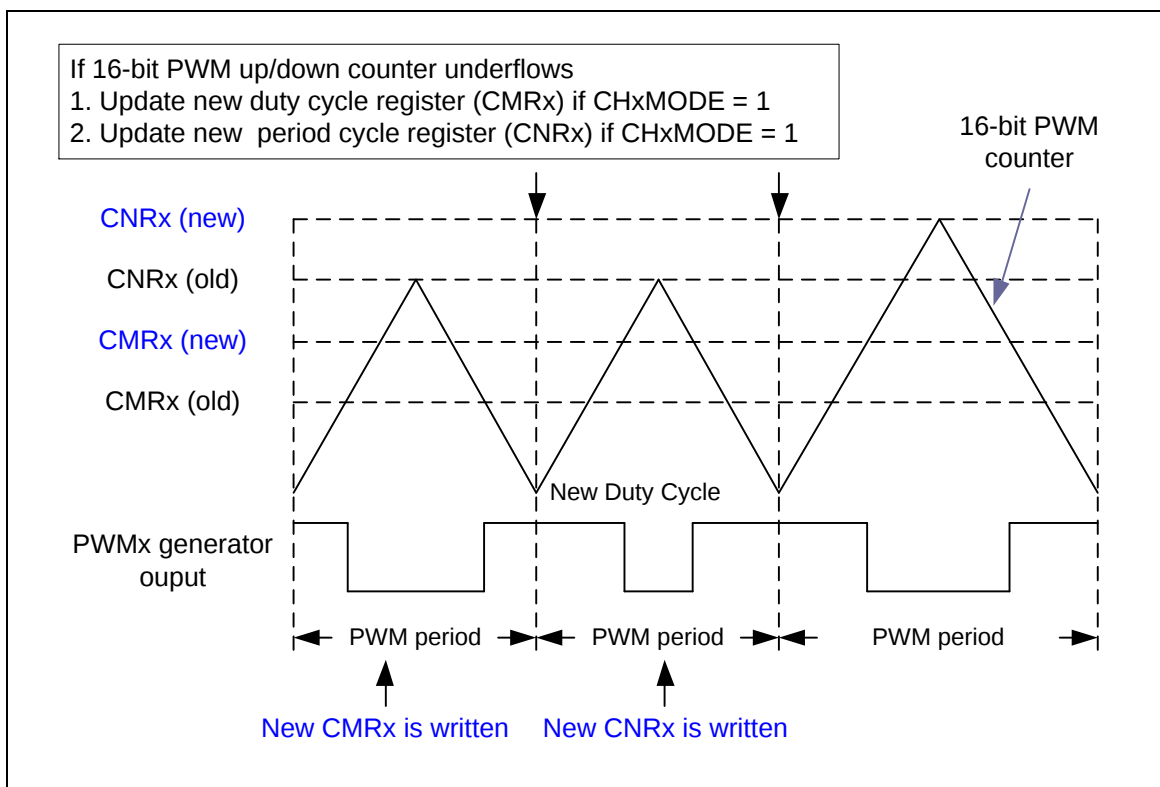


图 6.7-8 中心对齐模式输出波形

在中心对齐模式下，四种指定时序状况下系统可产生两种中断：周期中断和占空比中断。如果 $INT_{xx}TYPE$ ($PIER[17:16]$) = 0、向下计数的计数器等于0时或 $INT_{xx}TYPE$ ($PIER[17:16]$) = 1、向上计数的计数器等于 $CNRx$ 的值时PWM周期中断将产生，例如，PWM中心点。PWM占空比中断发生在 $INT_{xx}DTYPE$ ($PIER[25:24]$ = 0)、向下计数的计数器等于 CMR 的值时，或是 $INT_{xx}DTYPE$ ($PIER[25:24]$ = 1)、向上计数的计数器等于 CMR 的值时。

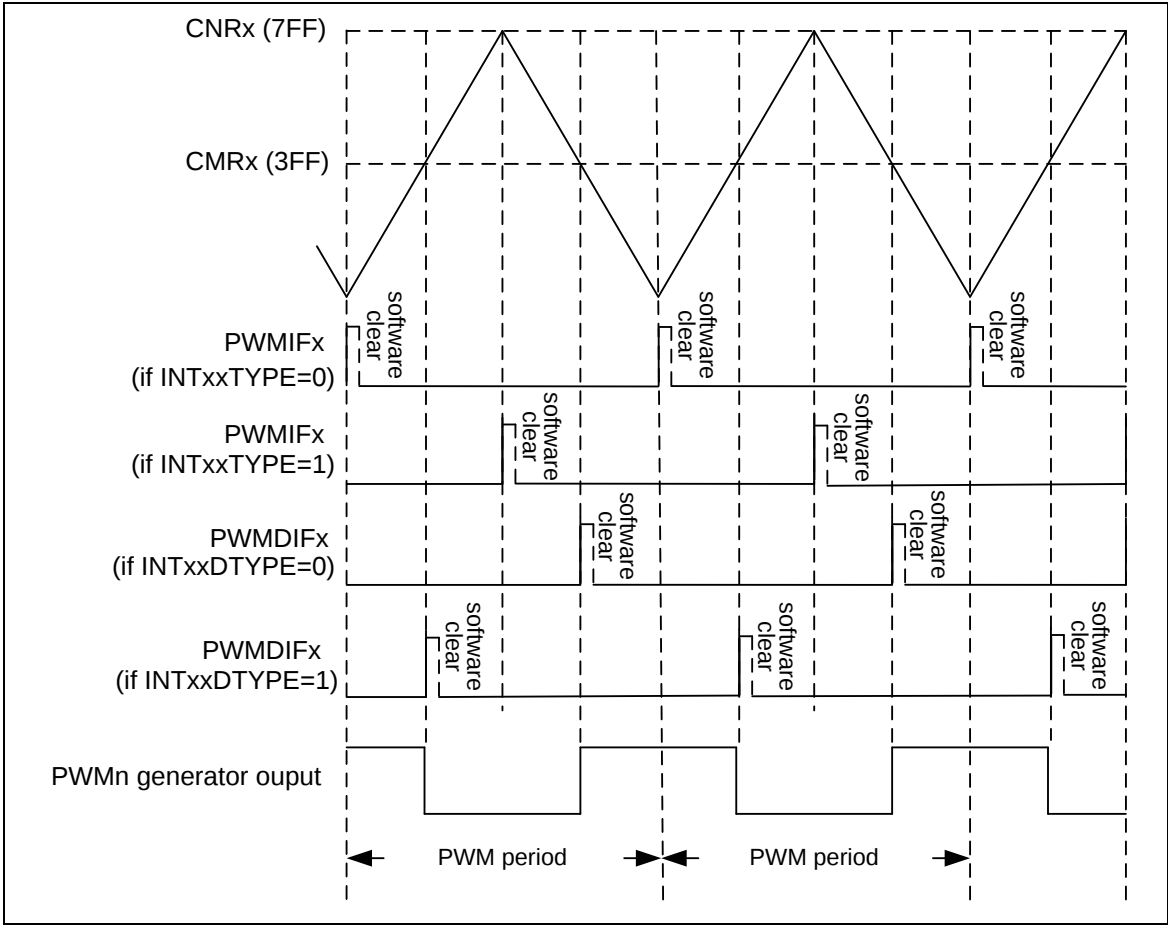


图 6.7-9 PWM 中心对齐中断发生时序图

6.7.5.4 PWM 双缓存, 自动重载以及单触发模式

PWM 定时器具有双缓存功能。寄存器新写入的值，会在下一个周期加载，并不会影响当前的定时器操作。PWM 计数器的值可以写入CNRx，当前PWM 计数器的值可从PDRx 读出。

如果CH0MOD位被设定为0，PWM0工作模式为单触发模式；如果CH0MOD位被设定为1时，PWM0工作模式为自动重载模式。推荐大家在使能PWM0计数器计数（即设定CH0EN为1）之前先切换PWM0的工作模式，因为在PWM0的工作模式更改时，CNR0和CMR0的内容将会被清0，而PWM0的周期和占空比也会被重置。当PWM0工作在单触发模式下，首先写值到CMR0和CNR0，然后通过设定CH0EN 位为1来使能PWM0的计数器，使其开始计数。在PWM0计数器的值从CNR0到0（向下计数）之后，CNR0和CMR0的内容将通过硬件清0，PWM的计数器将停止计数。软件需要写新的内容到CMR0和CNR0寄存器，来设定下一次的周期和占空比。当再一次开启单触发操作，CMR0的内容应该被先写，其原因为当CNR0被写入一个非0的值后，PWM0计数器将会自动地开始计数。当PWM0工作在自动重载模式下，CMR0和CNR0应该被先写值，然后通过设定CH0EN (PCR[0]) 位为1来使能PWM0计数器，使其开始计数。当PWM0计数器向下计数到0时，计数器将会重新装载CNR0的内容，如果CNR0的值被设定为0，计数器将停止计数。PWM1~ PWM3具有和PWM0一样的功能。

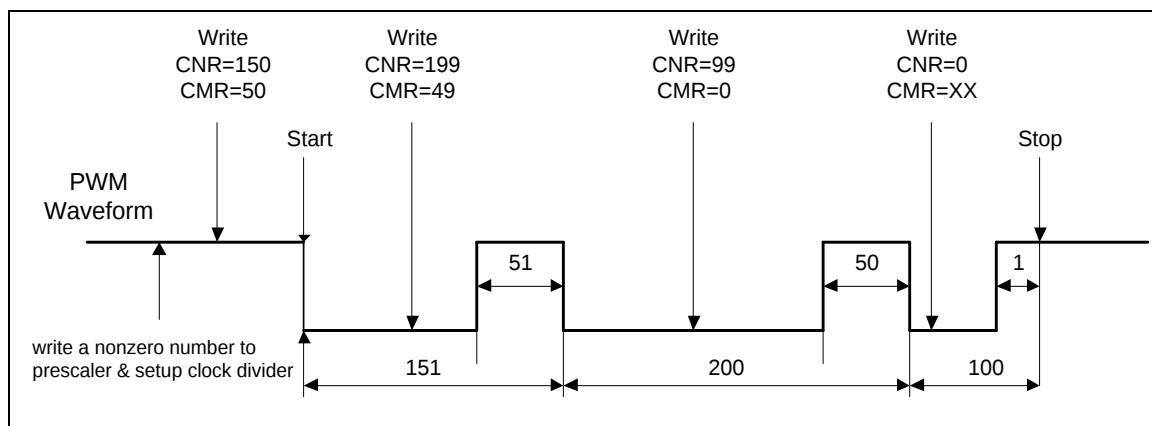


图 6.7-10 PWM 双缓存图解

6.7.5.5 占空比调制

双缓存允许CMRx寄存器的内容可以在任何时间被改写。写入的值在下一个周期才起作用

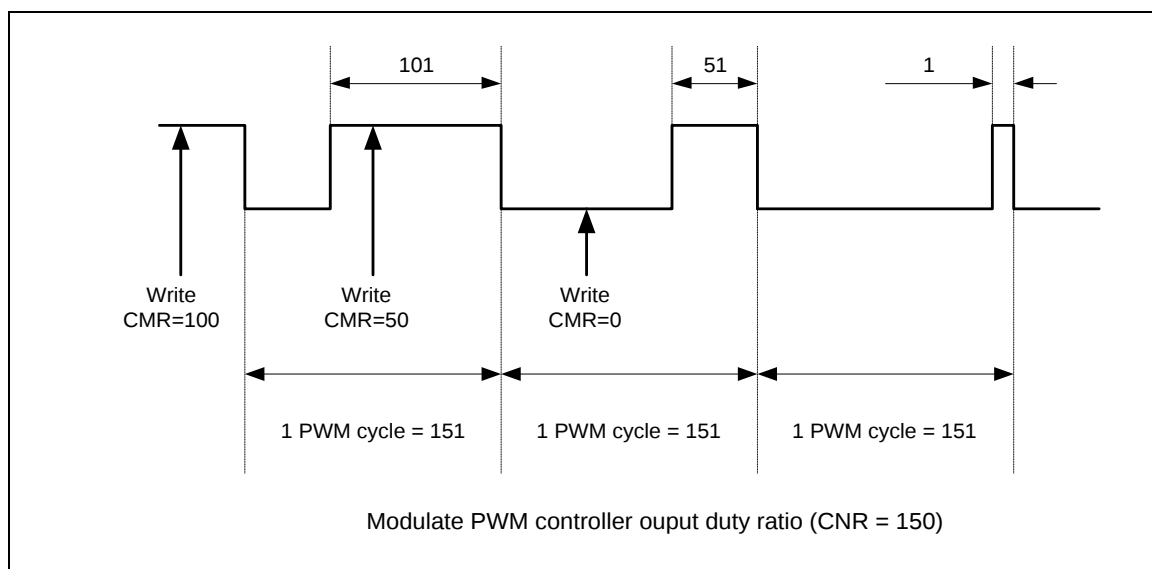


图 6.7-11 PWM 控制器输出占空比

6.7.5.6 死区发生器

PWM控制器提供死区发生器. 用于保护电源器件. 该功能在PWM上升沿产生可编程的延迟时间，用户通过编程PPRx.DZI来确定死区间隔.

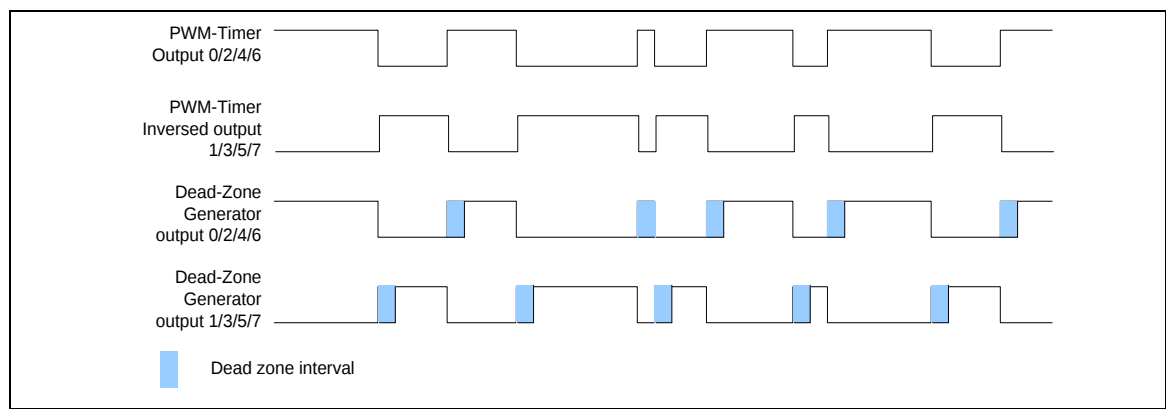


图 6.7-12 对-PWM 带死区发生器操作输出

6.7.5.7 PWM- 定时器触发ADC启动转换

PWM可以成为ADC转换的触发源。当PWM工作在中心对齐模式时，有4个PWM触发ADC转换的时间点可以选择。当PWM工作在边缘对齐模式时，有2个PWM触发ADC转换的时间点可以选择。

中心对齐模式时，通过设定INTxxDTYPE/ INTxxTYPE 和 PWMxDTEN/ PWMxTEN，PWM可在不同的时间点触发ADC转换开始。触发ADC的时序和设定如下所示。

INTxTEN	INTxxTYPE	描述
1	0	PWM计数器向下计数到0
1	1	PWM计数器向上计数到CNRx + 1

INTxDTEN	INTxxDTYPE	描述
1	0	PWM计数器向下计数到CMRx
1	1	PWM计数器向上计数到CMRx

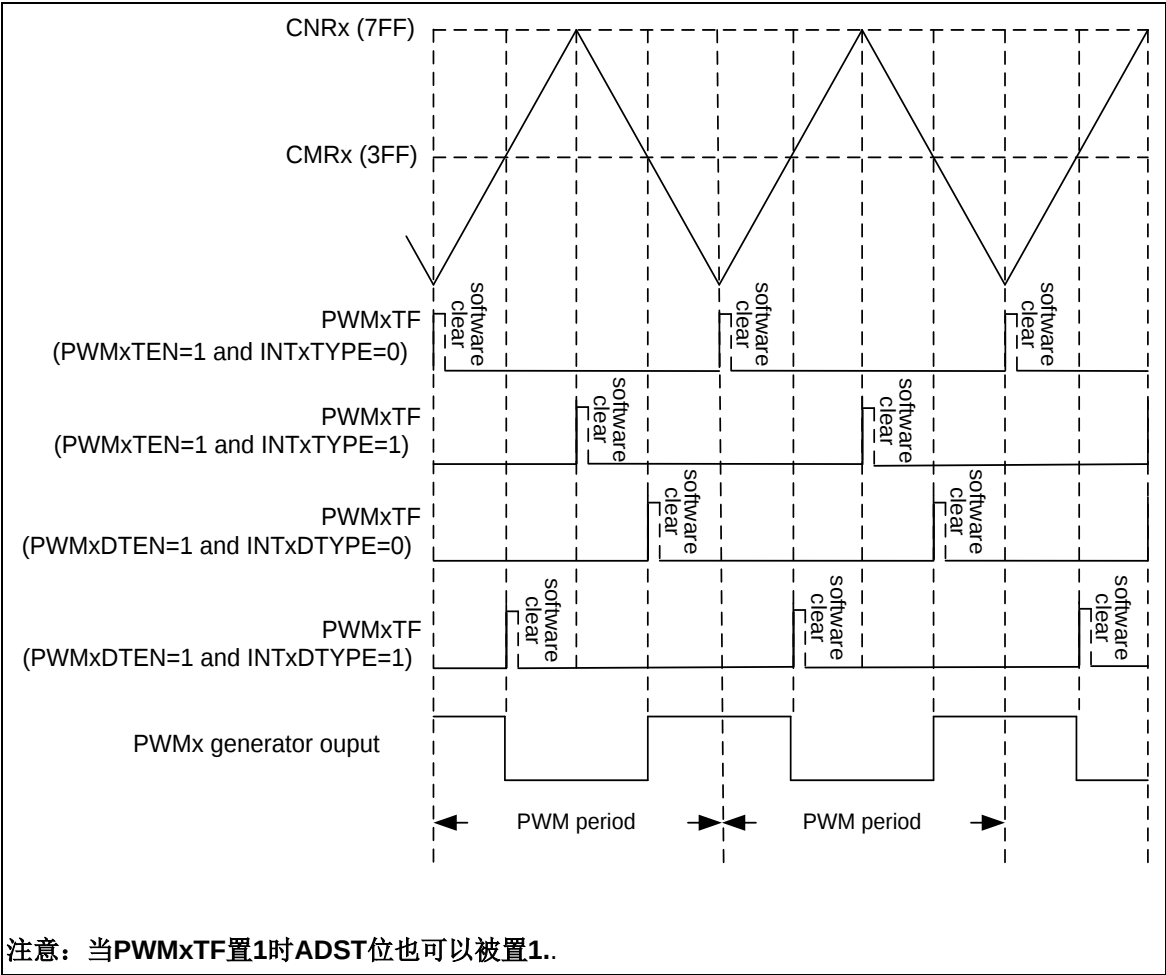


图 6.7-13 中心对齐模式下 PWM 触发 ADC Flag (PWMxTF)时序图

边缘对齐模式时，通过设定PWMxDTEN/ PWMxTEN，PWM可在不同的时间点触发ADC转换开始。触发ADC时序和设定如下所示

INTxDTEN	描述
1	PWM计数器向下计数到CMRx
INTxTEN	描述
1	PWM计数器向下计数到0

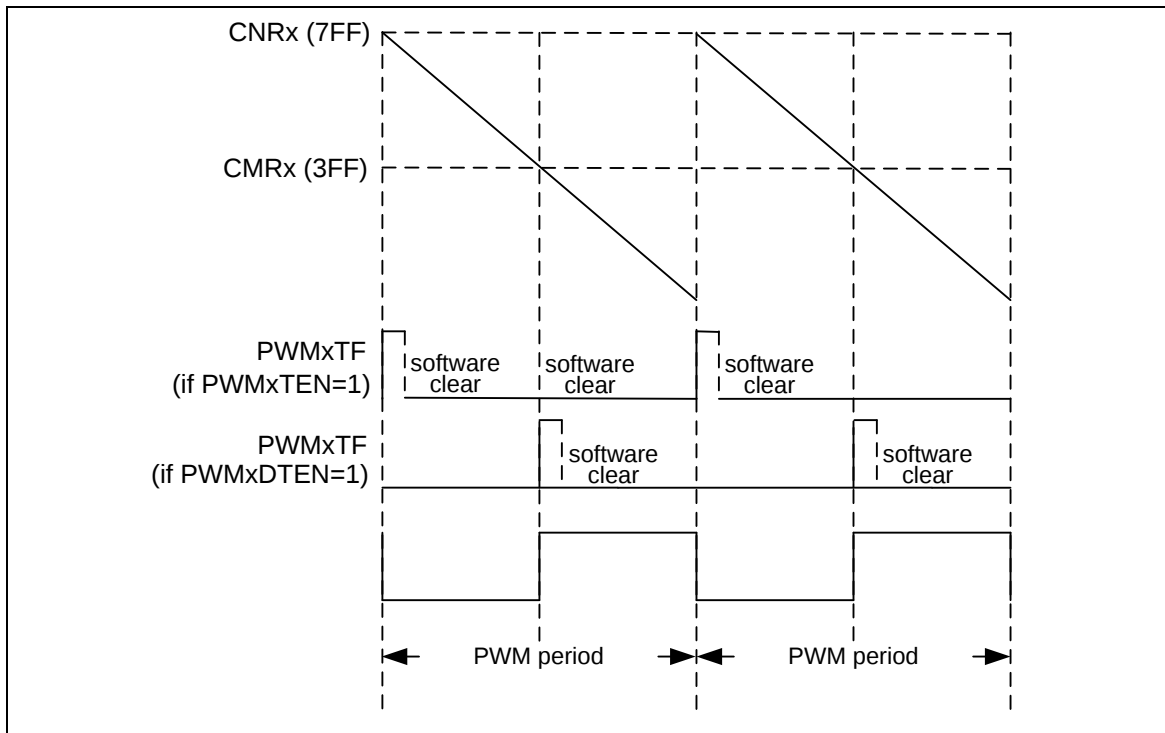


图 6.7-14 边缘对齐模式下 PWM 触发 ADC Flag (PWMxTF)时序图

6.7.5.8 PWM 触发ADC转换开始流程

1. 配置预分频寄存器(PPRx)，设置时钟预分频数(CPxx)
2. 配置时钟源选择寄存器(CSRx)，设置时钟源除频数。
3. 配置PWM控制寄存器(PCRx)，设定自动重载模式(CHxMOD = 1)，PWM对齐模式(PWMxxTYPE)和禁止PWM-Timer (CHxEN = 0)
4. 配置PWM控制寄存器(PCRx)，设定翻转on/off (CHxINV)，极性on/off (CHxPINV)和死区发生器on/off (DZENxx)。(可选)
5. 配置PWM中断使能寄存器(PIER)，设定PWM周期中断类型(INTxxTYPE)和PWM占空比中断类型(INTxxDTYPE)
6. 配置PWM触发控制寄存器(TCON)，使能/禁止PWM周期触发ADC(PWMxTEN)和PWM占空比触发ADC (PWMxDTEN)
7. 配置比较寄存器(CMRx)，设定PWM占空比
8. 配置PWM计数器寄存器(CNRx)，设定PWM-定时器载入值
9. 配置PWM输出使能寄存器(POE)，使能PWM输出通道(PWMx = 1)
10. 在ADC控制寄存器中配置ADC触发延迟控制寄存器(ADTDCR)，设置PWM触发延迟时间(PTDT) (可选)。
11. 在ADC控制寄存器中配置ADC信道使能寄存器(ADCHER)，使能模拟输入通道(CHENx = 1)
12. 在ADC控制寄存器中配置ADC控制寄存器(ADCR)，设置硬件触发源来自PWM触发(TRGS = 3)，外部触发使能(TRGEN = 1)，A/D转换器工作模式(ADMD = 0,2,3)和使能A/D转换器(ADEN

- = 1)。
13. 配置PWM控制寄存器(PCR)，使能PWM定时器开始运行(CHxEN = 1)
 14. 当PWM触发标志被硬件置为1时，ADST (ADCR[11])位将会被硬件置为1
 15. 软件可通过查询A/D转换结束标志– ADF (ADSR[0])来检查转换是否完成
 16. 在单一模式和单次循环扫描模式下，ADST (ADCR[11])位将会被自动清除为0。在连续扫描模式下，ADST将一直保持为1直到软件清除该位为止

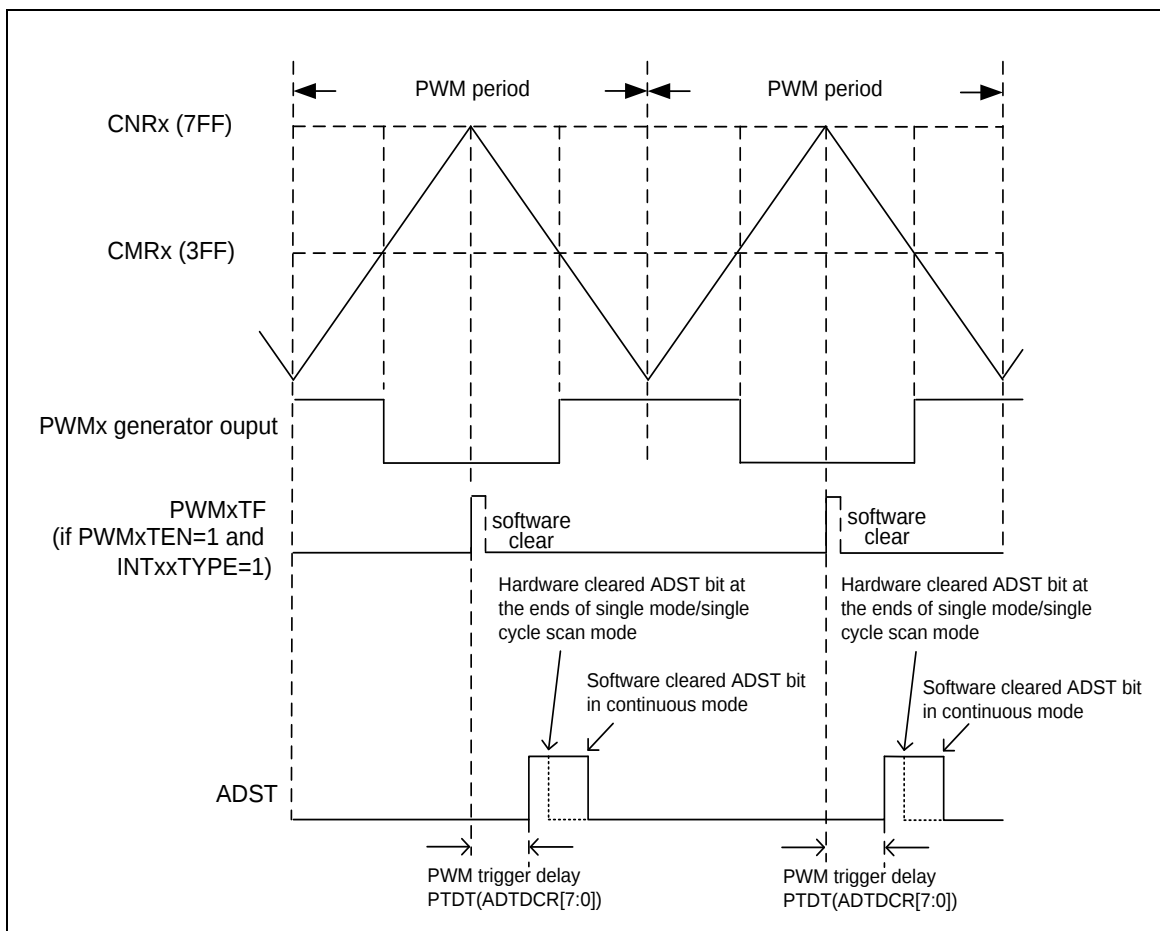


图 6.7-15 中心对齐模式下 PWM 触发 ADC 转换时序图

6.7.5.9 捕捉模式的操作

捕捉器0和PWM0 使用同一个定时器，捕捉器1和PWM1共同使用另一个定时器，以此类推。当输入信号有上升沿转变时，捕捉器将把PWM 计数器的值存入CFLRx寄存器；当输入信号有下降沿转变时，捕捉器将把PWM 计数器的值存入CFLRx寄存器。捕捉器通道0通过软件设定CCR0[1](使能上升沿锁存中断)和CCR0[2](使能下降沿锁存中断)可以判定其中断是什么中断源导致。同样，捕捉器通道1通过设定CCR0[17]和CCR0[18]具有同样的功能。依此类推。不管捕捉控制器执行何种捕捉中断，相应的PWM 计数器都会在同一时间重新装载CNRx 的值。**注意:** 在捕捉功能使能前，相应的管脚必须配置为捕获功能(禁止POE并使能CAPENR)。

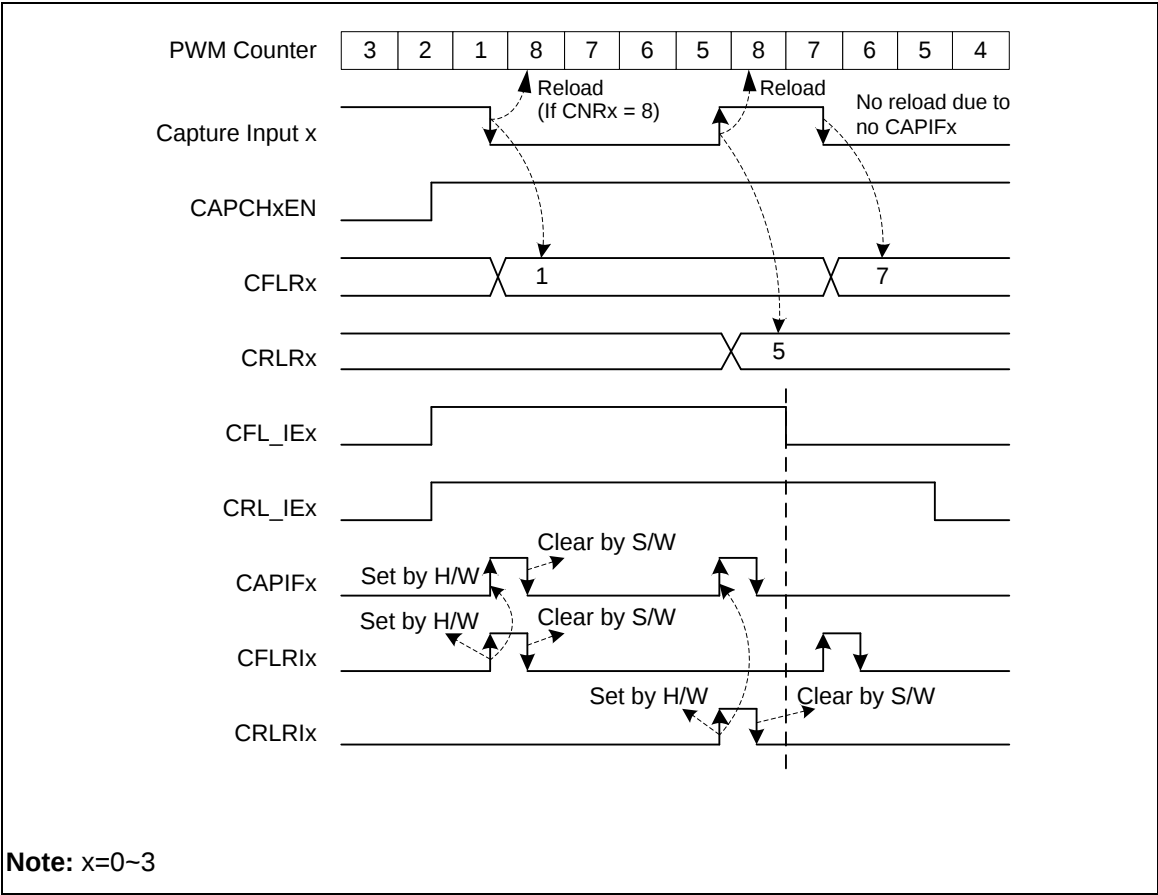


图 6.7-16 捕捉操作时序

在上述范例中，CNR 为8:

1. 当捕捉中断标志(CAPIFx)置位时，PWM计数器将重载CNRx.
2. 通道低脉冲宽度为(CNR + 1 - CRLR).
3. 通道高脉冲宽度为(CNR + 1 - CFLR).

6.7.5.10 PWM-定时器中断结构

共有4个PWM 中断，PWM0_INT~PWM3_INT，对高级中断控制器（AIC）来说，这些中断属于PWMA_INT。PWM 0 与捕捉器0共享同一个中断。PWM1 与捕捉器1 共享同一个中断，以此类

推。因此，在同一个信道PWM功能和捕捉功能不能同时使用。下图描述了PWM 定时器中断的结构

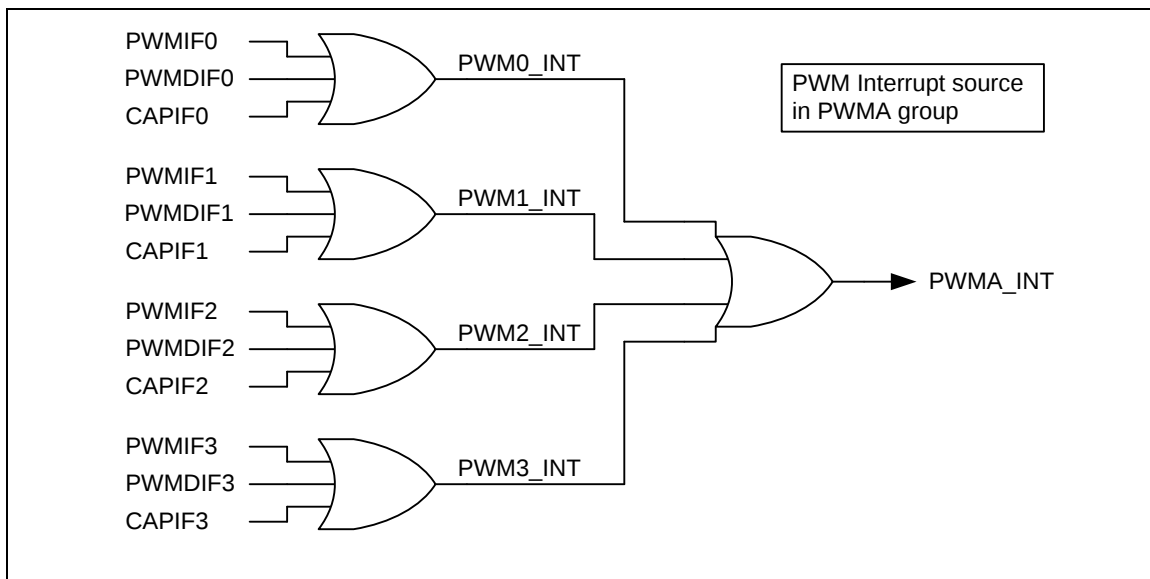


图 6.7-17 A 组 PWM-定时器中断结构图

6.7.5.11 PWM-定时器开启步骤

启动PWM 推荐下列步骤：

1. 配置预分频(PPR)寄存器，设置时钟预分频数(CPx)
2. 配置时钟选择寄存器(CSR)，选择时钟源(CSRx)
3. 配置PWM控制寄存器(PCR)，设置自动重载模式(CHxMOD = 1)并禁止PWM-定时器(CHxEN = 0)。
4. 配置PWM控制寄存器(PCR)，设置翻转开/关(CHxINV)和死区发生器开/关(DZENxx)。(可选)
5. 配置比较寄存器(CMRx)，设置PWM占空比(CMRx)
6. 配置PWM计数器寄存器(CNRx)，设置PWM-定时器载入值(CNRx)
7. 配置PWM中断使能寄存器(PIER)，设置PWM周期中断使能位(PWMIEx)。(可选)
8. 配置PWM控制输出使能寄存器(POE)，使能PWM输出通道(PWMx = 1)
9. 配置PWM控制寄存器(PCR)，使能PWM定时器开始运行(CHxEN = 1)

6.7.5.12 单次触发模式下PWM-定时器重新启动的步骤

在PWM单次触发模式下，当PWM波形产生一次后，PWM定时器将自动停止且CNR和CMR的值将会被硬件清0，软件必须重新填写CMR和CNR的值才能重新启动下一个PWM单次波形，关于再次启动产生PWM单次波形的步骤，建议按如下步骤进行：

1. 配置比较器寄存器(CMRx)，设定PWM占空比。

2. 配置PWM计数器寄存器(CNRx)，设定PWM周期.在设定CNRx之后，PWM 波形将会产生

6.7.5.13 PWM-定时器停止步骤

方法1:

设定16-位计数器(CNRx) 为0，并监测数据寄存器（PDRx，16-位向下计数器的当前值）。当PDRx 达到0，关闭PWM 定时器(PCR 的CHxEN 位). (推荐)

方法2:

设定16位计数器(CNRx) 为0，当中断发生时，关闭PWM定时器(PCR 的CHxEN 位). (推荐)

方法 3:

直接关闭PWM 定时器(PCR 的CHxEN 位). (不推荐)

不推荐方式3是因为禁止CHxEN 将立即停止PWM 输出信号，会导致PWM 输出的占空比改变, 可能引起电机控制电路的损坏

6.7.5.14 捕捉开始步骤

1. 配置预分频(PPR)寄存器，设置时钟预分频数(CPxx)
2. 配置时钟选择寄存器(CSR)，选择时钟源(CSRx)
3. 配置PWM控制寄存器(PCR)，设置自动重载模式(CHxMOD = 1)并禁止PWM-定时器(CHxEN = 0)。
4. 配置PWM捕捉控制寄存器(CCRx)，使能上升沿中断(CRL_IEx)和下降沿中断(CFL_IEx)，输入信号翻转开/关(INVx)
5. 配置PWM计数器寄存器(CNRx)，设置PWM-定时器载入值(CNRx)
6. 配置PWM控制寄存器(PCR)，使能PWM定时器开始运行(CHxEN = 1)
7. 配置捕捉输入信道使能寄存器（CAPENR），使能对应的GPIO管脚为捕捉功能

6.7.6 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移地址	R/W	描述	复位值
PWM 基地址: ● PWM group A PWMA_BA = 0x4004_0000				
PPR	PWMA_BA+0x00	R/W	PWM 预分频寄存器	0x0000_0000
CSR	PWMA_BA+0x04	R/W	PWM 除频时钟选择寄存器	0x0000_0000
PCR	PWMA_BA+0x08	R/W	PWM 控制寄存器	0x0000_0000
CNR0	PWMA_BA+0x0C	R/W	PWM 计数寄存器0	0x0000_0000
CMR0	PWMA_BA+0x10	R/W	PWM 比较寄存器0	0x0000_0000
PDR0	PWMA_BA+0x14	R	PWM 数据寄存器 0	0x0000_0000
CNR1	PWMA_BA+0x18	R/W	PWM 计数寄存器1	0x0000_0000
CMR1	PWMA_BA+0x1C	R/W	PWM 比较寄存器1	0x0000_0000
PDR1	PWMA_BA+0x20	R	PWM 数据寄存器1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWM 计数寄存器 2	0x0000_0000
CMR2	PWMA_BA+0x28	R/W	PWM 比较寄存器2	0x0000_0000
PDR2	PWMA_BA+0x2C	R	PWM 数据寄存器2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWM 计数寄存器r 3	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWM 比较寄存器 3	0x0000_0000
PDR3	PWMA_BA+0x38	R	PWM 数据寄存器 3	0x0000_0000
PIER	PWMA_BA+0x40	R/W	PWM 中断使能寄存器	0x0000_0000
PIIR	PWMA_BA+0x44	R/W	PWM 中断标志寄存器	0x0000_0000
CCR0	PWMA_BA+0x50	R/W	PWM 捕捉控制寄存器 0	0x0000_0000
CCR2	PWMA_BA+0x54	R/W	PWM 捕捉控制寄存器2	0x0000_0000
CRLR0	PWMA_BA+0x58	R	PWM 捕捉上升沿锁存寄存器(通道 0)	0x0000_0000
CFLR0	PWMA_BA+0x5C	R	PWM 捕捉下降沿锁存寄存器 (通道0)	0x0000_0000
CRLR1	PWMA_BA+0x60	R	PWM捕捉上升沿锁存寄存器(通道1)	0x0000_0000
CFLR1	PWMA_BA+0x64	R	PWM捕捉下降沿锁存寄存器(通道1)	0x0000_0000
CRLR2	PWMA_BA+0x68	R	PWM 捕捉上升沿锁存寄存器(通道2)	0x0000_0000

CFLR2	PWMA_BA+0x6C	R	PWM 捕捉下降沿锁存寄存器 (通道2)	0x0000_0000
CRLR3	PWMA_BA+0x70	R	PWM 捕捉上升沿锁存寄存器(通道3)	0x0000_0000
CFLR3	PWMA_BA+0x74	R	PWM捕捉下降沿锁存寄存器(通道3)	0x0000_0000
CAPENR	PWMA_BA+0x78	R/W	PWM 捕捉输入0~3 使能寄存器	0x0000_0000
POE	PWMA_BA+0x7C	R/W	PWM 使能通道0~3 输出	0x0000_0000
TCON	PWMA_BA+0x80	R/W	PWM 关于通道0~3的 触发控制	0x0000_0000
TSTATUS	PWMA_BA+0x84	R/W	PWM 触发状态寄存器	0x0000_0000

6.7.7 寄存器描述

PWM预分频寄存器 (PPR)

寄存器	偏移地址	R/W	描述	复位值
PPR	PWMA_BA+0x00	R/W	PWM预分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DZI23							
23	22	21	20	19	18	17	16
DZI01							
15	14	13	12	11	10	9	8
CP23							
7	6	5	4	3	2	1	0
CP01							

位	描述	
[31:24]	DZI23	信道2与信道3的死区间隔 该8位决定死区长度。 死区长度的单位时间= $[(\text{prescale}+1) \times (\text{clock source divider})] / \text{PWMxy_CLK}$ (xy 是 23).
[23:16]	DZI01	信道0与信道1的死区间隔 该8位决定死区长度。 死区长度的单位时间= $[(\text{prescale}+1) \times (\text{clock source divider})] / \text{PWMxy_CLK}$ (xy是 01)
[15:8]	CP23	时钟预分频 器2 时钟在输入到相应的PWM 定时器之前被(CP23 + 1) 除频

		如果CP23=0, 预分频器2输出时钟停止。相应PWM 定时器也停止.
[7:0]	CP01	时钟预分频 器0 时钟在输入到相应的PWM 定时器之前被(CP01 + 1) 除频 如果CP01=0, 预分频器0输出时钟停止。相应PWM 定时器也停止.

PWM时钟除频选择寄存器 (CSR)

寄存器	偏移地址	R/W	描述	复位值
CSR	PWMA_BA+0x04	R/W	PWM 时钟除频选择寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	CSR3			Reserved	CSR2		
7	6	5	4	3	2	1	0
Reserved	CSR1			Reserved	CSR0		

位	描述													
[31:15]	Reserved	Reserved.												
[14:12]	CSR3	PWM 定时器3 时钟除频选择 PWM 定时器3的时钟除频选择												
		<table><tr><th>CSR3 [14:12]</th><th>输入时钟除频数</th></tr><tr><td>000</td><td>2</td></tr><tr><td>001</td><td>4</td></tr><tr><td>010</td><td>8</td></tr><tr><td>011</td><td>16</td></tr><tr><td>100</td><td>1</td></tr></table>	CSR3 [14:12]	输入时钟除频数	000	2	001	4	010	8	011	16	100	1
		CSR3 [14:12]	输入时钟除频数											
		000	2											
		001	4											
		010	8											
		011	16											
100	1													
[11]	Reserved	Reserved.												
[10:8]	CSR2	PWM 定时器 2时钟除频选择 PWM 定时器2的时钟除频选择 (表和 CSR3一样)												
[7]	Reserved	Reserved												
[6:4]	CSR1	PWM 定时器 1时钟除频选择 PWM 定时器1的时钟除频选择 (表和 CSR3一样)												

[3]	Reserved	Reserved
[2:0]	CSR0	PWM 定时器0时钟除频选择 PWM 定时器0的时钟除频选择 (表和 CSR3一样)

PWM 控制寄存器(PCR)

寄存器	偏移地址	R/W	描述	复位值
PCR	PWMA_BA+0x08	R/W	PWM 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
PWM23TYPE	PWM01TYPE	Reserved		CH3MOD	CH3INV	CH3PINV	CH3EN
23	22	21	20	19	18	17	16
Reserved				CH2MOD	CH2INV	CH2PINV	CH2EN
15	14	13	12	11	10	9	8
Reserved				CH1MOD	CH1INV	CH1PINV	CH1EN
7	6	5	4	3	2	1	0
Reserved		DZEN23	DZEN01	CH0MOD	CH0INV	CH0PINV	CH0EN

位	描述	
[31]	PWM23TYPE	PWM23对齐类型选择 0 = 边沿对齐 1 = 中心对齐
[30]	PWM01TYPE	PWM01对齐类型选择 0 =边沿对齐 1 =中心对齐
[29:28]	Reserved	Reserved.
[27]	CH3MOD	PWM-定时器 3 自动重载/单次模式控制 0 = 单次模式 1 = 自动重载 注意： 如果该位的值变化，会使CNR3 和CMR3 清0.
[26]	CH3INV	PWM-定时器 3输出反转使能 0 = 反转关闭 1 = 反转打开
[25]	CH3PINV	PWM-定时器 3 输出极性反转使能 0 = PWM3输出极性不反转 1 = PWM3输出极性反转使能
[24]	CH3EN	PWM-定时器3 使能 0 =相应的PWM 定时器停止运行 1 =相应的PWM 定时器开始运行

[23:20]	Reserved	Reserved.
[19]	CH2MOD	PWM-定时器2自动重载/单次模式 0 = 单次模式 1 = 自动重载模式 注意: 如果该位的值变化, 会使CNR2和CMR2 清0.
[18]	CH2INV	PWM-定时器2 输出反转使能 0 = 反转关闭 1 = 反转打开
[17]	CH2PINV	PWM-定时器 2输出极性反转使能 0 = PWM2输出极性不反转 1 = PWM2输出极性反转使能.
[16]	CH2EN	PWM-定时器 2 使能 0 = 相应的PWM 定时器停止运行 1 = 相应的PWM 定时器开始运行
[15:12]	Reserved	Reserved.
[11]	CH1MOD	PWM-定时器1 自动重载/单次模式 0 = 单次模式 1 = 自动重载 注意: 如果该位的值变化, 会使CNR1和CMR1清0.
[10]	CH1INV	PWM-定时器 1 输出反转使能 0 = 反转关闭 1 = 反转打开
[9]	CH1PINV	PWM-定时器 1输出极性反转使能 0 = PWM1输出极性不反转. 1 = PWM1输出极性反转使能
[8]	CH1EN	PWM-定时器1 使能 0 = 相应的PWM 定时器停止运行 1 = 相应的PWM 定时器开始运行
[7:6]	Reserved	Reserved.
[5]	DZEN23	死区 2 发生器使能 0 = 禁用. 1 = 使能 注意: 当死区发生器使能时, PWM2 和PWM3 将成为PWM A 组的一对互补信号。
[4]	DZEN01	死区0 发生器使能 0 = 禁用. 1 = 使能 注意: 当死区发生器使能, PWM0 和 PWM1将成为PWM A 组的一对互补信号。
[3]	CH0MOD	PWM-定时器 0 自动重载/单次模式 0 = 单次模式. 1 = 自动重载 注意: 如果该位的值变化, 会使CNR0和CMR0 清0.

[2]	CH0INV	PWM-定时器 0 输出反转使能 0 =反转关闭 1 =反转打开
[1]	CH0PINV	PWM-定时器 0 输出极性反转使能 0 = PWM0输出极性不反转. 1 = PWM0输出极性反转使能
[0]	CH0EN	PWM-定时器 0 使能 0 =相应的PWM 定时器停止运行. 1 =相应的PWM 定时器开始运行

PWM计数寄存器3-0 (CNR3-0)

寄存器	偏移地址	R/W	描述	复位值
CNR0	PWMA_BA+0x0C	R/W	PWM 计数寄存器0	0x0000_0000
CNR1	PWMA_BA+0x18	R/W	PWM计数寄存器 1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWM计数寄存器 2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWM计数寄存器3	0x0000_0000

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

位	描述	
[31:16]	Reserved	Reserved
[15:0]	CNRx	PWM 定时器载入值 CNR决定了PWM周期。 PWM频率= $PWM_{xy_CLK} / [(prescale+1) * (clock\ divider) * (CNR+1)]$; xy可能为01, 23, 取决于所选择的PWM 通道 边沿对齐模式 ● 占空比= $(CMR+1)/(CNR+1)$. ● $CMR \geq CNR$: PWM 输出高 ● $CMR < CNR$: PWM 低脉宽= $(CNR-CMR)$ unit; PWM 高脉宽= $(CMR+1)$ unit. ● $CMR = 0$: PWM低脉宽= (CNR) unit; PWM高脉宽= 1 unit. 中心对齐模式: ● 占空比= $[(2 \times CMR) + 1] / [2 \times (CNR+1)]$. ● $CMR > CNR$: PWM输出高. ● $CMR \leq CNR$: PWM低脉宽= $2 \times (CNR-CMR) + 1$ unit; PWM高脉宽= $(2 \times CMR) + 1$ unit.

		CMR = 0: PWM 低脉宽 = 2 x CNR + 1 unit; PWM 高脉宽 = 1 unit. (Unit = 一个PWM时钟周期) . 注意1: CNR写入数据后将在下一个PWM 周期生效 注意2: 当CNR 等于0时， PWM 输出总是高电平 注意3:当PWM 工作在中心对齐模式时， CNR 的值应该被设在0x0001 到0xFFFE 之间。 如果CNR 等于0x0000 或 0xFFFF， PWM 工作将不正常
--	--	---

PWM比较寄存器3-0 (CMR3-0)

寄存器	偏移地址	R/W	描述	复位值
CMR0	PWMA_BA+0x10	R/W	PWM 比较器0	0x0000_0000
CMR1	PWMA_BA+0x1C	R/W	PWM比较器1	0x0000_0000
CMR2	PWMA_BA+0x28	R/W	PWM比较器2	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWM比较器3	0x0000_0000

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

位	描述	
[31:16]	Reserved	Reserved
[15:0]	CMRx	PWM 比较寄存器 CMR 决定PWM占空比 PWM 频率 = PWMxy_CLK/[(prescale+1)*(clock divider)*(CNR+1)]; xy, 可能 01或 23, 取决于所选择的PWM 通道. 边沿对齐模式 - 占空比 = (CMR+1)/(CNR+1). - CMR >= CNR: PWM 输出高. - CMR < CNR: PWM低脉宽= (CNR-CMR) unit; PWM 高脉宽 = (CMR+1) unit. - CMR = 0: PWM 低脉宽= (CNR) unit; PWM高脉宽 = 1 unit. 中心对齐模式: - 占空比= [(2 x CMR) + 1]/[2 x (CNR+1)]. - CMR > CNR: PWM 输出高 - CMR <= CNR: PWM低脉宽= 2 x (CNR-CMR) + 1 unit; PWM高脉宽= (2 x CMR)

		+ 1 unit. ● CMR = 0: PWM低脉宽= 2 x CNR + 1 unit; PWM高脉宽= 1 unit. (Unit = 一个PWM 周期). 注意: CMR写入数据后将在下一个PWM 周期生效
--	--	---

PWM数据寄存器3-0 (PDR 3-0)

寄存器	偏移地址	R/W	描述	复位值
PDR0	PWMA_BA+0x14	R	PWM 数据寄存器 0	0x0000_0000
PDR1	PWMA_BA+0x20	R	PWM 数据寄存器1	0x0000_0000
PDR2	PWMA_BA+0x2C	R	PWM 数据寄存器 2	0x0000_0000
PDR3	PWMA_BA+0x38	R	PWM 数据寄存器3	0x0000_0000

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

位	描述	
[31:16]	Reserved	Reserved.
[15:0]	PDRx	PWM 数据寄存器 用户可以监控PDR来查询16位计数器的当前值.

PWM中断使能寄存器(PIER)

寄存器	偏移地址	R/W	描述	复位值
PIER	PWMA_BA+0x40	R/W	PWM中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved						INT23DTYP E	INT01DTYP E
23	22	21	20	19	18	17	16
Reserved						INT23TYPE	INT01TYPE
15	14	13	12	11	10	9	8
Reserved				PWMDIE3	PWMDIE2	PWMDIE1	PWMDIE0
7	6	5	4	3	2	1	0
Reserved				PWMIE3	PWMIE2	PWMIE1	PWMIE0

位	描述	
[31:26]	Reserved	保留
[25]	INT23DTYPE	PWM23占空比中断类型选择 0 = 当PWM计数器往下计数到CMRx时， PWMDIFx将被置位。当PWM计数器往下计数到CMRx时， 如果对应的PWM触发使能位(PWMxDTEN)为1， PWM将触发ADC转换。 1 = 当PWM计数器往上计数到CMRx时， PWMDIFx将被置位。当PWM计数器往上计数到CMRx时， 如果对应的PWM触发使能位(PWMxDTEN)为1， PWM将触发ADC转换。 注意： 设置INT23DTYPE为1只在PWM工作在中心对齐方式时有效
[24]	INT01DTYPE	PWM01占空比中断类型选择 0 = 当PWM计数器往下计数到CMRx时， PWMDIFx将被置位。当PWM计数器往下计数到CMRx时， 如果对应的PWM触发使能位(PWMxDTEN)为1， PWM将触发ADC转换。 1 = 当PWM计数器往上计数到CMRx时， PWMDIFx将被置位。当PWM计数器往上计数到CMRx时， 如果对应的PWM触发使能位(PWMxDTEN)为1， PWM将触发ADC转换。 注意： 设置INT01DTYPE为1只在PWM工作在中心对齐方式时有效
[23:18]	Reserved	Reserved.
[17]	INT23TYPE	PWM23周期中断类型选择 0 = 当PWM计数器下溢出时， PWMDIFx将被置位。当PWM计数器下溢出时， 如果对应的PWM触发使能位(PWMxDTEN)为1， PWM将触发ADC转换。 1 = 当PWM计数器计数到CNRx时， PWMDIFx将被置位。当PWM计数器计数到CNRx时， 如果对应的PWM触发使能位(PWMxDTEN)为1， PWM将触发ADC转换。

		注意: 设置INT23TYPE为1只在PWM工作在中心对齐方式时有效
[16]	INT01TYPE	PWM01周期中断类型选择 0 =当PWM计数器下溢出时, PWMDIFx将被置位。当PWM计数器下溢出时, 如果对应的PWM触发使能位(PWMxDTEN)为1, PWM将触发ADC转换。 1 =当PWM计数器计数到CNRx时, PWMDIFx将被置位。当PWM计数器计数到CNRx时, 如果对应的PWM触发使能位(PWMxDTEN)为1, PWM将触发ADC转换。 注意: 设置 INT01TYPE 为 1 只在 PWM 工作在中心对齐方式时有效
[15:12]	Reserved	Reserved.
[11]	PWMDIE3	PWM 通道3占空比中断使能位 0 =禁止 1 =使能.
[10]	PWMDIE2	PWM 通道2占空比中断使能位 0 =禁止 1 =使能.
[9]	PWMDIE1	PWM 通道1占空比中断使能位 0 =禁止 1 =使能.
[8]	PWMDIE0	PWM 通道0占空比中断使能位 0 =禁止 1 =使能.
[3]	PWMIE3	PWM 通道 3 周期中断使能位 0 =禁止 1 =使能. .
[2]	PWMIE2	PWM 通道2周期中断使能位 0 =禁止 1 =使能. .
[1]	PWMIE1	PWM 通道 1 周期中断使能位 0 =禁止 1 =使能. .
[0]	PWMIE0	PWM 通道 0 周期中断使能位 0 =禁止 1 =使能. .

PWM中断标志寄存器(PIIR)

寄存器	偏移地址	R/W	描述	复位值
PIIR	PWMA_BA+0x44	R/W	PWM中断标志寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				PWMDIF3	PWMDIF2	PWMDIF1	PWMDIF0
7	6	5	4	3	2	1	0
Reserved				PWMIF3	PWMIF2	PWMIF1	PWMIF0

位	描述	
[31:12]	Reserved	Reserved.
[11]	PWMDIF3	PWM通道3占空比中断标志 当INTDTYPE23 = 0, PWM 通道3计数器向下计数, 其值等于CMR3 寄存器的值时, 这个标志将由硬件置位 当INTDTYPE23 = 1, PWM 通道3计数器向上计数, 其值等于CMR3 寄存器的值时, 这个标志将由硬件置位 注意: 软件可以写1清除
[10]	PWMDIF2	PWM通道2占空比中断标志 当INTDTYPE23 = 0, PWM 通道2计数器向下计数, 其值等于CMR2 寄存器的值时, 这个标志将由硬件置位 当INTDTYPE23 = 1, PWM 通道2计数器向上计数, 其值等于CMR2 寄存器的值时, 这个标志将由硬件置位 注意: 软件可以写1清除
[9]	PWMDIF1	PWM通道1占空比中断标志 当INTDTYPE01 = 0, PWM 通道1计数器向下计数, 其值等于CMR1 寄存器的值时, 这个标志将由硬件置位 当INTDTYPE01 = 1, PWM 通道1计数器向上计数, 其值等于CMR1 寄存器的值时, 这个标志将由硬件置位 注意: 软件可以写1清除
[8]	PWMDIF0	PWM通道0占空比中断标志 当INTDTYPE01 = 0, PWM 通道0计数器向下计数, 其值等于CMR0 寄存器的值时, 这个标志将由硬件置位 当INTDTYPE01 = 1, PWM 通道0计数器向上计数, 其值等于CMR0 寄存器的值时, 这个标志将由硬件置位 注意: 软件可以写1清除

[7:4]	Reserved	Reserved.
[3]	PWMIF3	PWM通道3周期中断标志 当INTDTYPE23 = 0, PWM 通道3计数器等于0时, 这个标志将由硬件置位 当INTDTYPE23 = 1, PWM 通道3计数器等于CNR3 寄存器的值时, 这个标志将由硬件置位 注意: 软件可以写1清除
[2]	PWMIF2	PWM通道2周期中断标志 当INTDTYPE22 = 0, PWM 通道2计数器等于0时, 这个标志将由硬件置位 当INTDTYPE22 = 1, PWM 通道2计数器等于CNR2 寄存器的值时, 这个标志将由硬件置位 注意: 软件可以写 1 清除
[1]	PWMIF1	PWM通道1周期中断标志 当INTDTYPE01 = 0, PWM 通道1计数器等于0时, 这个标志将由硬件置位 当INTDTYPE01 = 1, PWM 通道1计数器等于CNR1 寄存器的值时, 这个标志将由硬件置位 注意: 软件可以写 1 清除
[0]	PWMIF0	PWM通道0周期中断标志 当INTDTYPE00 = 0, PWM 通道0计数器等于0时, 这个标志将由硬件置位 当INTDTYPE00 = 1, PWM 通道0计数器等于CNR0 寄存器的值时, 这个标志将由硬件置位 注意: 软件可以写 1 清除

捕捉控制寄存器0(CCR0)

寄存器	偏移地址	R/W	描述	复位值
CCR0	PWMA_BA+0x50	R/W	PWM捕捉控制寄存器0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CFLRI1	CRLRI1	Reserved	CAPIF1	CAPCH1EN	CFL_IE1	CRL_IE1	INV1
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CFLRI0	CRLRI0	Reserved	CAPIF0	CAPCH0EN	CFL_IE0	CRL_IE0	INV0

位	描述	
[31:24]	Reserved	Reserved.
[23]	CFLRI1	CFLR1锁定标志位 当PWM 组输入通道1发生下降沿变化时, CFLR1 锁存PWM 向下计数器的值, 该位由硬件置位. 注意: 软件写 1 清该位.
[22]	CRLRI1	CRLR1锁定标志位 当PWM 组输入通道1发生上升沿变化时, CRLR1 锁存PWM 向下计数器的值, 该位由硬件置位. 注意: 软件写1清该位..
[21]	Reserved	保留.
[20]	CAPIF1	通道 1捕捉中断指示标志 如果PWM 组通道1上升锁存中断使能(CRL_IE1=1), PWM 组通道1发生上升沿转变会使CAPIF1 为高; 同样, 如果PWM 组通道1下降锁定中断使能(CFL_IE1=1), 下降沿会使CAPIF1 置高. 注意: 写1 清该位为0
[19]	CAPCH1EN	信道1捕捉功能使能位 0 = 禁用PWM 组信道1的捕捉功能 1 = 使能PWM 组信道1的捕捉功能. 注意1: 当使能时, 捕捉功能将锁存PWM 计数器的值, 并存储到CRLR (上升锁存) 或CFLR (下降锁存). 注意2: 当禁用时, 捕捉器不更新CRLR 或CFLR, 并关闭通道1中断.
[18]	CFL_IE1	通道1下降沿锁存中断使能 0 = 禁用下降沿锁存中断

		1 = 使能下降沿锁存中断 注意： 使能时，如果检测到PWM 组通道1有下降沿转变，捕捉中断产生。
[17]	CRL_IE1	通道1上升沿锁存中断使能 0 = 禁用上升沿锁存中断 1 = 使能上升沿锁存中断 注意： 使能时，如果检测到PWM 组通道1有上升沿转变，捕捉中断产生。
[16]	INV1	通道1反转使能 0 = 反转关闭 1 = 反转打开 来自 GPIO 引脚的输入信号反转之后再输入到捕捉定时器
[15:8]	Reserved	Reserved.
[7]	CFLR0	CFLR0锁定标志位 当PWM 组输入通道0发生下降沿变化时, CFLR0 锁存PWM 向下计数器的值, 该位由硬件置位。 注意： 软件写1清该位。
[6]	CRLR0	CRLR0锁定标志位 当PWM 组输入通道0发生上升沿变化时, CRLR0 锁存PWM 向下计数器的值, 该位由硬件置位。 注意： 软件写1清该位..
[5]	Reserved	Reserved.
[4]	CAPIF0	通道0捕捉中断指示标志 如果PWM 组通道0上升锁存中断使能(CRL_IE0=1), PWM 组通道0发生上升沿转变会使CAPIF0 为高; 同样, 如果PWM 组通道0下降锁定中断使能(CFL_IE0=1), 下降沿会使CAPIF0 置高。 注意： 写1 清该位为0
[3]	CAPCH0EN	信道0捕捉功能使能位 0 = 禁用PWM 组信道0的捕捉功能 1 = 使能PWM 组信道0的捕捉功能。 注意1： 当使能时，捕捉功能将锁存PWM 计数器的值，并存储到CRLR (上升锁存) 或CFLR (下降锁存)。 注意2： 当禁用时，捕捉器不更新CRLR 或CFLR, 并关闭通道0中断。
[2]	CFL_IE0	通道0下降沿锁存中断使能 0 = 禁用下降沿锁存中断 1 = 使能下降沿锁存中断 注意： 使能时，如果检测到PWM 组通道0有下降沿转变，捕捉中断产生。
[1]	CRL_IE0	通道0上升沿锁存中断使能 0 = 禁用上升沿锁存中断 1 = 使能上升沿锁存中断

		注意：使能时，如果检测到PWM 组通道0有上升沿转变，捕捉中断产生.
[0]	INV0	通道0反转使能 0 = 反转关闭 1 = 反转打开 来自 GPIO 引脚的输入信号反转之后再输入到捕捉定时器

捕捉控制寄存器2 (CCR2)

寄存器	偏移地址	R/W	描述	复位值
CCR2	PWMA_BA+0x54	R/W	PWM捕捉控制寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CFLRI3	CRLRI3	Reserved	CAPIF3	CAPCH3EN	CFL_IE3	CRL_IE3	INV3
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CFLRI2	CRLRI2	Reserved	CAPIF2	CAPCH2EN	CFL_IE2	CRL_IE2	INV2

位	描述	
[31:24]	Reserved	Reserved.
[23]	CFLRI3	CFLR3锁定标志位 当PWM 组输入通道3发生下降沿变化时, CFLR3 锁存PWM 向下计数器的值, 该位由硬件置位. 注意: 软件写 1 清该位.
[22]	CRLRI3	CRLR3锁定标志位 当PWM 组输入通道3发生上升沿变化时, CRLR3 锁存PWM 向下计数器的值, 该位由硬件置位. 注意: 软件写1清该位..
[21]	Reserved	保留.
[20]	CAPIF3	通道 3捕捉中断指示标志 如果PWM 组通道3上升锁存中断使能(CRL_IE3=1), PWM 组通道3发生上升沿转变会使CAPIF3为高; 同样, 如果PWM 组通道3下降锁定中断使能(CFL_IE3=1), 下降沿会使CAPIF3 置高. 注意: 写1 清该位为0
[19]	CAPCH3EN	信道3捕捉功能使能位 0 = 禁用PWM 组信道3的捕捉功能 1 = 使能PWM 组信道3的捕捉功能. 注意1: 当使能时, 捕捉功能将锁存PWM 计数器的值, 并存储到CRLR (上升锁存) 或CFLR (下降锁存).

		注意2: 当禁用时, 捕捉器不更新CRLR 或CFLR, 并关闭通道3中断.
[18]	CFL_IE3	通道3下降沿锁存中断使能 0 = 禁用下降沿锁存中断 1 = 使能下降沿锁存中断 注意: 使能时, 如果检测到PWM 组通道3有下降沿转变, 捕捉中断产生.
[17]	CRL_IE3	通道3上升沿锁存中断使能 0 = 禁用上升沿锁存中断 1 = 使能上升沿锁存中断 注意: 使能时, 如果检测到PWM 组通道3有上升沿转变, 捕捉中断产生.
[16]	INV3	通道3反转使能 0 = 反转关闭 1 = 反转打开 来自 GPIO 引脚的输入信号反转之后再输入到捕捉定时器
[15:8]	Reserved	Reserved.
[7]	CFLRI2	CFLR2锁定标志位 当PWM 组输入通道2发生下降沿变化时, CFLR2 锁存PWM 向下计数器的值, 该位由硬件置位. 注意: 软件写1清该位.
[6]	CRLRI2	CRLR2锁定标志位 当PWM 组输入通道2发生上升沿变化时, CRLR2 锁存PWM 向下计数器的值, 该位由硬件置位. 注意: 软件写1清该位..
[5]	Reserved	Reserved.
[4]	CAPIF2	通道2捕捉中断指示标志 如果PWM 组通道2上升锁存中断使能(CRL_IE2=1), PWM 组通道2发生上升沿转变会使CAPIF2 为高; 同样, 如果PWM 组通道2下降锁定中断使能(CFL_IE=1), 下降沿会使CAPIF2 置高. 注意: 写1 清该位为0
[3]	CAPCH2EN	信道2捕捉功能使能位 0 = 禁用PWM 组信道2的捕捉功能 1 = 使能PWM 组信道2的捕捉功能. 注意1: 当使能时, 捕捉功能将锁存PWM 计数器的值, 并存储到CRLR (上升锁存) 或CFLR (下降锁存). 注意2: 当禁用时, 捕捉器不更新CRLR 或CFLR, 并关闭通道2中断.
[2]	CFL_IE2	通道2下降沿锁存中断使能 0 = 禁用下降沿锁存中断 1 = 使能下降沿锁存中断

		注意：使能时，如果检测到PWM 组通道2有下降沿转变，捕捉中断产生.
[1]	CRL_IE2	通道2上升沿锁存中断使能 0 = 禁用上升沿锁存中断 1 = 使能上升沿锁存中断 注意：使能时，如果检测到PWM 组通道2有上升沿转变，捕捉中断产生.
[0]	INV2	通道2反转使能 0 = 反转关闭 1 = 反转打开 来自 GPIO 引脚的输入信号反转之后再输入到捕捉定时器

捕捉上升沿锁存寄存器3-0 (CRLR3-0)

寄存器	偏移地址	R/W	描述	复位值
CRLR0	PWMA_BA+0x58	R	PWM 捕捉上升沿锁存寄存器 (通道 0)	0x0000_0000
CRLR1	PWMA_BA+0x60	R	PWM 捕捉上升沿锁存寄存器 (通道1)	0x0000_0000
CRLR2	PWMA_BA+0x68	R	PWM 捕捉上升沿锁存寄存器 (通道2)	0x0000_0000
CRLR3	PWMA_BA+0x70	R	PWM 捕捉上升沿锁存寄存器 (通道3)	0x0000_0000

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

位	描述	
[31:16]	Reserved	Reserved.
[15:0]	CRLRx	捕捉上升沿锁存寄存器 当通道 0/1/2/3 有上升沿转变时，锁存 PWM 计数。

捕捉下降沿锁存寄存器3-0 (CFLR3-0)

寄存器	偏移地址	R/W	描述	复位值
CFLR0	PWMA_BA+0x5C	R	PWM捕捉下降沿锁存寄存器（通道 0）	0x0000_0000
CFLR1	PWMA_BA+0x64	R	PWM捕捉下降沿锁存寄存器（通道 1）	0x0000_0000
CFLR2	PWMA_BA+0x6C	R	PWM捕捉下降沿锁存寄存器（通道 2）	0x0000_0000
CFLR3	PWMA_BA+0x74	R	PWM捕捉下降沿锁存寄存器（通道 3）	0x0000_0000

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

位	描述	
[31:16]	Reserved	Reserved.
[15:0]	CFLRx	捕捉下降沿锁存寄存器 当通道 0/1/2/3 有下降沿转变时，锁存 PWM 计数

捕捉输入使能寄存器(CAPENR)

寄存器	偏移地址	R/W	描述	复位值
CAPENR	PWMA_BA+0x78	R/W	PWM捕捉输入 0~3 使能寄存器	0x0000_0000

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

位	描述	
[31:4]	Reserved	Reserved.
[3:0]	CAPENR	捕捉输入使能位 0 =捕捉输入禁止。(PWMx多功能管脚输入不影响输入捕捉功能) 1 =捕捉输入使能。(PWMx多功能管脚输入影响输入捕捉功能) 位3210对应PWM A组各通道 位 xxx1 →捕捉通道0使能。捕捉输入通道可选择P2. 0或P4. 0, 用户可通过设置多功能管脚寄存器来选择。 Bit xx1x →捕捉通道1使能。捕捉输入通道可选择P2. 1或P4. 1, 用户可通过设置多功能管脚寄存器来选择。 Bit x1xx →捕捉通道2使能。捕捉输入通道可选择P2. 2或P4. 2, 用户可通过设置多功能管脚寄存器来选择。 Bit 1xxx →捕捉通道3使能。捕捉输入通道可选择P2. 3或P4. 3, 用户可通过设置多功能管脚寄存器来选择。

PWM 输出使能寄存器(POE)

寄存器	偏移地址	R/W	描述	复位值
POE	PWMA_BA+0x7C	R/W	PWM 通道0~3输出使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				PWM3	PWM2	PWM1	PWM0

位	描述	
[31:4]	Reserved	Reserved.
[3]	PWM3	通道 3 输出使能位 0 =禁止 PWM 通道 3 输出 1 =使能 PWM 通道 3 输出 注意： 相应的 GPIO 管脚必须切换到 PWM 功能
[2]	PWM2	通道 2 输出使能位 0 =禁止 PWM 通道 2 输出 1 =使能 PWM 通道 2 输出 注意： 相应的 GPIO 管脚必须切换到 PWM 功能
[1]	PWM1	通道 1 输出使能位 0 =禁止 PWM 通道 1 输出 1 =使能 PWM 通道 1 输出 注意： 相应的 GPIO 管脚必须切换到 PWM 功能
[0]	PWM0	通道0 输出使能位 0 =禁止 PWM 通道 0 输出 1 =使能 PWM 通道 0 输出 注意： 相应的 GPIO 管脚必须切换到 PWM 功能

PWM触发控制寄存器(TCON)

寄存器	偏移地址	R/W	描述	复位值
TCON	PWMA_BA+0x80	R/W	PWM通道0~3触发控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved				PWM3DTEN	PWM2DTEN	PWM1DTEN	PWM0DTEN
7	6	5	4	3	2	1	0
Reserved				PWM3TEN	PWM2TEN	PWM1TEN	PWM0TEN

位	描述	
[31:12]	Reserved	Reserved.
[11]	PWM3DTEN	通道 3 DUTY触发ADC使能位 0 = PWM信道 3触发 ADC功能禁止 1 = PWM信道 3触发 ADC功能使能 当PWM工作在边沿对齐方式时, 当PWM信道3计数器下数到CMR3时, 使能该位可以使能PWM触发ADC开始转换。 当PWM工作在中心对齐方式时, 当PWM通道3计数器根据INT23DTYPE中设定向上或向下数到CMR3时, 使能该位可以使能PWM触发ADC开始转换。
[10]	PWM2DTEN	通道 2 DUTY触发ADC使能位 0 = PWM信道 2触发 ADC功能禁止 1 = PWM信道2触发 ADC功能使能 当PWM工作在边沿对齐方式时, 当PWM信道2计数器下数到CMR2时, 使能该位可以使能PWM触发ADC开始转换。 当 PWM 工作在中心对齐方式时, 当 PWM 通道 2 计数器根据 INT23DTYPE 中设定向上或向下数到 CMR2 时, 使能该位可以使能 PWM 触发 ADC 开始转换。
[9]	PWM1DTEN	通道 1 DUTY触发ADC使能位 0 = PWM信道 1触发 ADC功能禁止 1 = PWM信道1触发 ADC功能使能 当PWM工作在边沿对齐方式时, 当PWM信道1计数器下数到CMR1时, 使能该位可以使能PWM触发ADC开始转换。 当 PWM 工作在中心对齐方式时, 当 PWM 通道 1 计数器根据 INT01DTYPE 中设定向上或向下数到 CMR1 时, 使能该位可以使能 PWM 触发 ADC 开始转换。

[8]	PWM0DTEN	通道 0 DUTY触发ADC使能位 0 = PWM信道 0触发 ADC功能禁止 1 = PWM信道0触发 ADC功能使能 当PWM工作在边沿对齐方式时，当PWM信道0计数器下数到CMR0时，使能该位可以使能PWM触发ADC开始转换。 当 PWM 工作在中心对齐方式时，当 PWM 通道 0 计数器根据 INT01DTYPE 中设定向上或向下数到 CMR0 时，使能该位可以使能 PWM 触发 ADC 开始转换。
[7:4]	Reserved	Reserved.
[3]	PWM3TEN	通道 3 周期触发使能位 0 = PWM信道 3触发 ADC功能禁止 1 = PWM信道 3触发 ADC功能使能 当PWM工作在边沿对齐方式时，当PWM信道3计数器下数溢出时，使能该位可以使能PWM触发ADC开始转换。 当PWM工作在中心对齐方式时，当PWM通道3计数器根据INT23DTYPE中设定向上数到(CNR3 + 1)或下数溢出时，使能该位可以使能PWM触发ADC开始转换。
[2]	PWM2TEN	通道 2 周期触发使能位 0 = PWM信道 2触发 ADC功能禁止 1 = PWM信道2触发 ADC功能使能 当PWM工作在边沿对齐方式时，当PWM信道2计数器下数溢出时，使能该位可以使能PWM触发ADC开始转换。 当 PWM 工作在中心对齐方式时，当 PWM 通道 2 计数器根据 INT23DTYPE 中设定向上数到(CNR2+ 1)或下数溢出时，使能该位可以使能 PWM 触发 ADC 开始转换。
[1]	PWM1TEN	通道1 周期触发使能位 0 = PWM信道 1触发 ADC功能禁止 1 = PWM信道1触发 ADC功能使能 当PWM工作在边沿对齐方式时，当PWM信道1计数器下数溢出时，使能该位可以使能PWM触发ADC开始转换。 当 PWM 工作在中心对齐方式时，当 PWM 通道 1 计数器根据 INT01DTYPE 中设定向上数到(CNR1 + 1)或下数溢出时，使能该位可以使能 PWM 触发 ADC 开始转换。
[0]	PWM0TEN	通道0 周期触发使能位 0 = PWM信道0触发 ADC功能禁止 1 = PWM信道0触发 ADC功能使能 当PWM工作在边沿对齐方式时，当PWM信道0计数器下数溢出时，使能该位可以使能PWM触发ADC开始转换。 当 PWM 工作在中心对齐方式时，当 PWM 通道 0 计数器根据 INT01DTYPE 中设定向上数到(CNR0 + 1)或下数溢出时，使能该位可以使能 PWM 触发 ADC 开始转换。

PWM 触发状态寄存器(TSTATUS)

寄存器	偏移地址	R/W	描述	复位值
TSTATUS	PWMA_BA+0x84	R/W	PWM 触发状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
Reserved				PWM3TF	PWM2TF	PWM1TF	PWM0TF

位	描述	
[31:4]	Reserved	Reserved.
[3]	PWM3TF	通道 3触发ADC标志 当PWM通道3触发ADC条件满足时，该位由硬件置1。如果ADC的触发源选择PWM，在该位置1后，ADC开始转换。 软件写1清0该位
[2]	PWM2TF	通道 2触发ADC标志 当PWM通道2触发ADC条件满足时，该位由硬件置1。如果ADC的触发源选择PWM，在该位置1后，ADC开始转换。 软件写1清0该位
[1]	PWM1TF	通道1触发ADC标志 当PWM通道1触发ADC条件满足时，该位由硬件置1。如果ADC的触发源选择PWM，在该位置1后，ADC开始转换。 软件写1清0该位
[0]	PWM0TF	通道 0触发ADC标志 当PWM通道0触发ADC条件满足时，该位由硬件置1。如果ADC的触发源选择PWM，在该位置1后，ADC开始转换。 软件写1清0该位

6.8 看门狗定时器(WDT)

6.8.1 概述

设计看门狗定时器的目的是，当系统运行到一个未知状态时，通过它来使系统复位。这种做法可以预防系统进入到无限期的死循环。此外，该看门狗定时器支持系统从Idle/Power-down模式中唤醒功能。

6.8.2 特性

- 18位的自由向上计数器,可满足各种看门狗定时器的超时间隔要求
- 超时间隔有($2^4 \sim 2^{18}$)个WDT_CLK时钟周期可选，如果WDT_CLK = 10 kHz，那么超时间隔是104 ms ~ 26.3168 s
- 系统复位保持时间为 $(1 / \text{WDT_CLK}) * 63$
- 支持看门狗定时器复位延时周期
 - ◆ 可选的复位延时周期包括 $(1026、130、18 \text{ 或 } 3) * \text{WDT_CLK}$ 个复位延时周期
- 当CWDTEN (CONFIG[31] 看门狗使能)位被置为0，支持芯片上电或芯片复位后看门狗强制打开.
- 支持看门狗定时器超时唤醒功能，此时时钟源必须选择内部低速10k时钟源

6.8.3 框图

看门狗定时器时钟控制和框图如下：

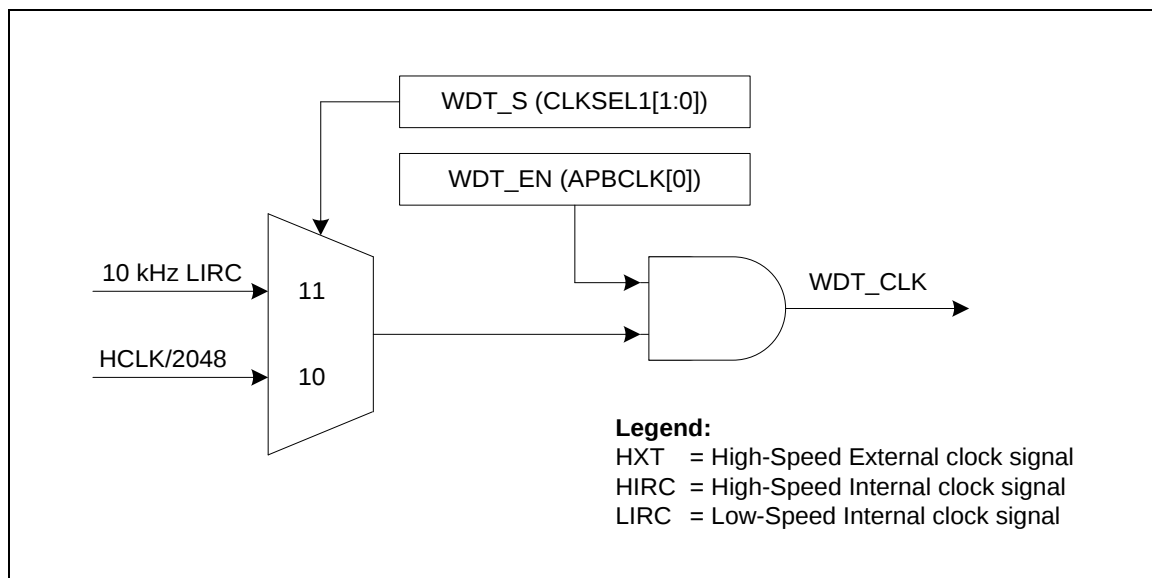


图 6.8-1 看门狗定时器时钟控制

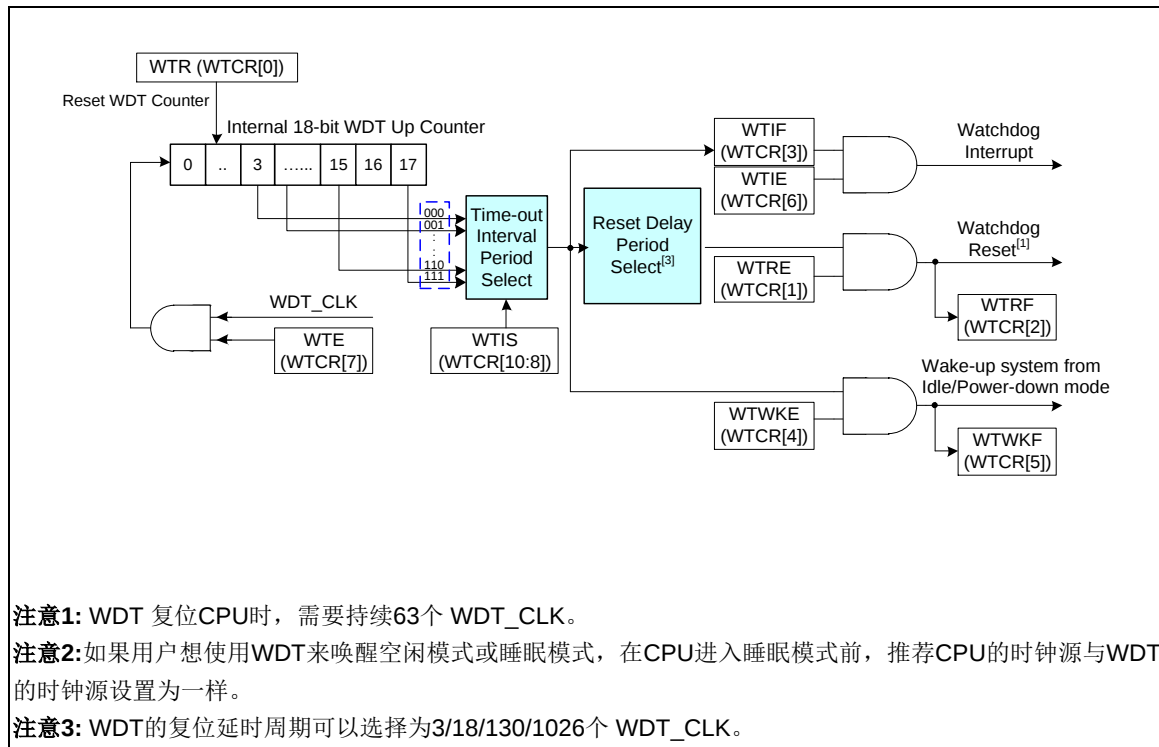


图 6.8-2 看门狗定时器框图

6.8.4 基本配置

看门狗定时器的时钟通过APBCLK[0]位使能，时钟源选择通过CLKSEL1[1:0]设置

或者用户可以在烧录时把CONFIG[31]配置为0，从而在芯片上电或RESET后，会强制使能看门狗定时器和10k时钟源。

6.8.5 功能描述

看门狗定时器(WDT)包含一个18位的，可设置超时间隔的向上计数器。表6-21显示了WDT超时间隔的选择，图6.8-3(看门狗定时器超时间隔和复位周期时序)显示了WDT超时间隔和复位周期时序。

WDT 超时溢出中断

对WTE置位可以使能WDT功能，然后WDT计数器开始向上计数。通过设置WTIS可以选择8个不同的超时间隔。当WDT计数器计到WTIS设定值时，WDT产生溢出中断，然后WTIF标志立即被置1。

WDT 复位延时周期和复位系统

WTIF标志被置位后，还会有一个固定的延时周期TRSTD，用户应在TRSTD 延时周期计完之前，对WTR置1来复位18位的WDT向上计数器的值，以防止产生WDT超时溢出复位信号。如果TRSTD时间计满以后，WDT向上计数器的值仍没有被清除，此时若WTRE位为1，则WTRF标志会被置1，然后芯片立即进入复位状态。请参考图6.8-3，T_{RST}复位周期将保持至少63个WDT时钟周期，然后芯片从复位向量地址(0x0000_0000)重新开始执行程序。芯片由WDT超时溢出复位后，WTRF标志将会保持1。用户可以通过检查该标志来确认系统是否因WDT超时溢出复位

WDT 唤醒

如果看门狗时钟源选择为内部10k, 且WTWKE位被使能, 看门狗超时溢出中断产生后, 系统能从 Power-down 模式中被唤醒。与此同时, WTWKF 标志会被自动置1。用户可以通过查询 WTWKF 标志, 知道系统是否由看门狗超时溢出中断唤醒。

超时溢出 周期选择 WTIS	超时溢出间隔周期 T_{TIS}	复位延时周期 T_{RSTD}
000	$2^4 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
001	$2^6 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
010	$2^8 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
011	$2^{10} * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
100	$2^{12} * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
101	$2^{14} * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
110	$2^{16} * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
111	$2^{18} * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$

表 6.8-1 看门狗定时器超时溢出间隔周期选择

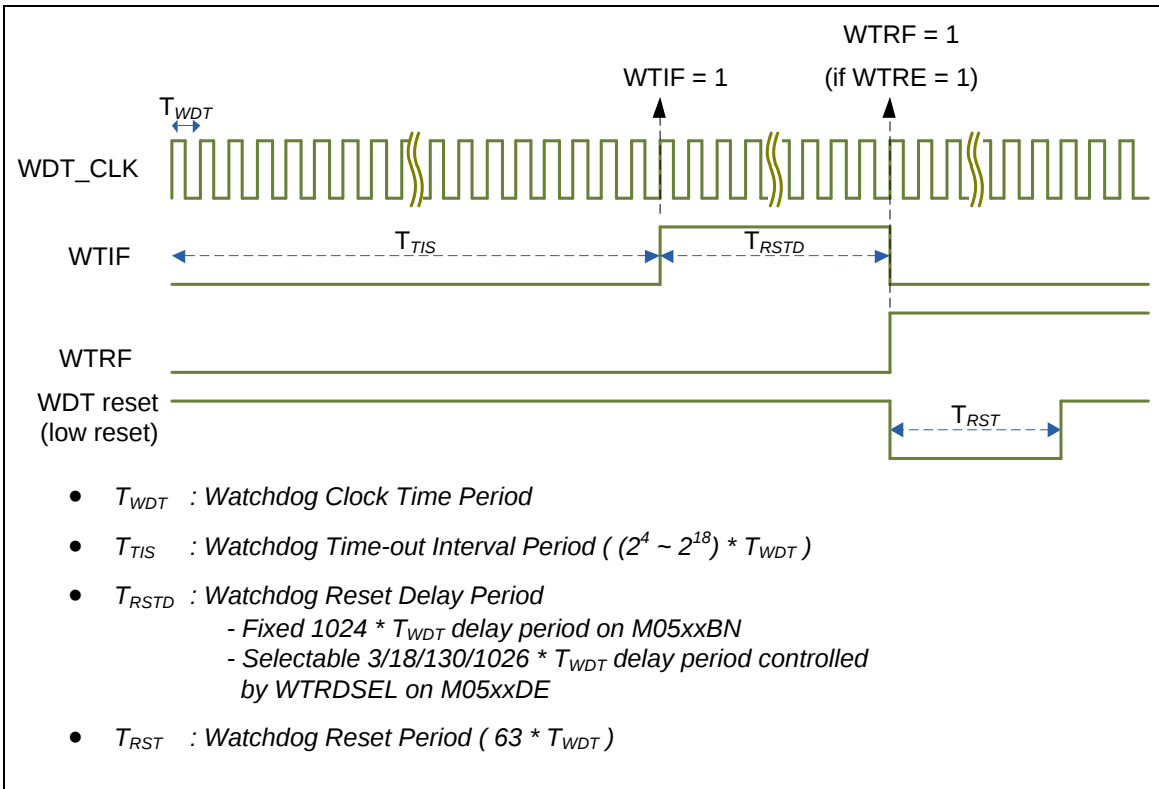


图 6.8-3 看门狗定时器超时溢出间隔和复位周期时序图

6.8.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移地址	R/W	描述	复位值
WDT 基地址: WDT_BA = 0x4000_4000				
WTCR	WDT_BA+0x00	R/W	看门狗定时器控制寄存器	0x0000_0700
WTCRALT	WDT_BA+0x04	R/W	看门狗定时器选择控制寄存器	0x0000_0000

6.8.7 寄存器描述

看门狗定时器控制寄存器(WTCR)

寄存器	偏移地址	R/W	描述	复位值
WTCR	WDT_BA+0x00	R/W	看门狗定时器控制寄存器	0x0000_0700

31	30	29	28	27	26	25	24
DBGACK_WDT	保留						
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留					WTIS		
7	6	5	4	3	2	1	0
WTE	WTIE	WTWKF	WTWKE	WTIF	WTRF	WTRE	WTR

位	描述	
[31]	DBGACK_WDT	ICE 调试模式下看门狗计数器控制位(写保护) 0 = ICE 调试模式影响看门狗计数。 当ICE使CPU运行停止时, 看门狗计数器也停止。 1 = ICE 调试模式不影响看门狗计数。 不论CPU是否由ICE停止还是运行, 看门狗计数器都继续计数
[30:11]	保留	保留
[10:8]	WTIS	看门狗定时器超时溢出周期选择 (写保护) 以下三个位用于选择看门狗超时溢出周期 $000 = 2^4 * T_{WDT}$ $001 = 2^6 * T_{WDT}$ $010 = 2^8 * T_{WDT}$ $011 = 2^{10} * T_{WDT}$ $100 = 2^{12} * T_{WDT}$ $101 = 2^{14} * T_{WDT}$ $110 = 2^{16} * T_{WDT}$ $111 = 2^{18} * T_{WDT}$
[7]	WTE	看门狗定时器使能位 (写保护) 0 =看门狗定时器禁止. (该操作会复位内部计数器的值.) 1 =看门狗定时器使能.
[6]	WTIE	看门狗定时器超时溢出中断使能(写保护) 如果该位使能,看门狗超时溢出后会产生中断信号, 并告知CPU

		0 = 看门狗超时溢出中断禁止. 1 = 看门狗超时溢出中断使能.
[5]	WTWKF	看门狗定时器溢出唤醒标志 该位表明看门狗定时器中断唤醒标志的状态 0 = 看门狗定时器没有引起芯片唤醒 1 = 如果看门狗超时溢出中断信号产生, 芯片从Idle或Power-down 模式中被唤醒 注意: 该位为写1清零.
[4]	WTWKE	看门狗定时器超时溢出唤醒功能控制位 (写保护) 如果此位被置1, 当WTIF被置1且WTIE被使能时, 看门狗超时溢出中断信号将会触发唤醒MCU. 0 = 唤醒触发事件禁止, 即便看门狗定时器超时溢出中断发生, 也不唤醒MCU 1 = 唤醒触发事件使能, 如果看门狗定时溢出中断产生, 将唤醒MCU 注意: MCU能被看门狗定时器溢出中断信号唤醒的条件是看门狗定时器时钟源必须为内部10k时钟
[3]	WTIF	看门狗定时器超时溢出标志 当看门狗定时器计数器的值达到溢出时间间隔时, 该位将被置1 0 = 没有发生看门狗超时溢出中断 1 = 有看门狗超时溢出中断发生. 注意: 该位写1清零
[2]	WTRF	看门狗定时器超时复位标志 该位用来指示系统的复位是否因看门狗定时器超时溢出发生 0 = 没有发生看门狗超时溢出复位 1 = 发生了看门狗超时溢出复位. 注意: 该位写1清零.
[1]	WTRE	看门狗定时器超时复位使能位 (写保护) 如果看门狗计数器的值达到设定值, 且固定的WDT复位时间计完前还没有被清零, 如果该位被设置, 会引发看门狗超时溢出复位 0 = 看门狗定时溢出复位功能禁止. 1 = 看门狗定时溢出复位功能使能 注意: 通过设置WTRDSEL位来选择$3/18/130/1026 * T_{WDT}$复位延时周期
[0]	WTR	复位看门狗计数器 (写保护) 0 = 不影响. 1 = 18位看门狗定时器计数值复位清零. 注意: 该位由硬件自动清零.

看门狗定时器选择控制寄存器 (WTCRALT)

寄存器	偏移地址	R/W	描述	复位值
WTCRALT	WDT_BA+0x04	R/W	看门狗定时器选择控制寄存器	0x0000_0000

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

位	描述	
[31:2]	保留	保留
[1:0]	WTRDSEL	看门狗定时器复位延时选择 (写保护) 当看门狗定时器发生溢出, 在看门狗复位延时的时段内可以将看门狗计数器清零, 以防止看门狗溢出复位. 对不同的看门狗溢出时间, 用户也可以选择适当的看门狗复位延时时间 00 =看门狗定时器复位延时时间是1026 * WDT_CLK. 01 =看门狗定时器复位延时时间是130 * WDT_CLK. 10 =看门狗定时器复位延时时间是18 * WDT_CLK. 11 =看门狗定时器复位延时时间是3 * WDT_CLK. 注意: 如果看门狗定时器超时复位发生, 该寄存器的值将被清零

6.9 窗口看门狗定时器 (WWDT)

6.9.1 简介

窗口看门狗定时器用于在一个窗口时间内执行系统复位，以防止程序在不可预知条件下跑到一个不可控的状态。

6.9.2 特性

- 6位向下计数值(WWDT_CVAL) 和6位比较窗口值(WIN_CMP)，使得窗口周期更加灵活
- 支持由4位值来选择16种看门狗预分频值，预分频器最大由11位组成，因此最大可2048分频

6.9.3 框图

窗口看门狗定时器的时钟控制和框图如下

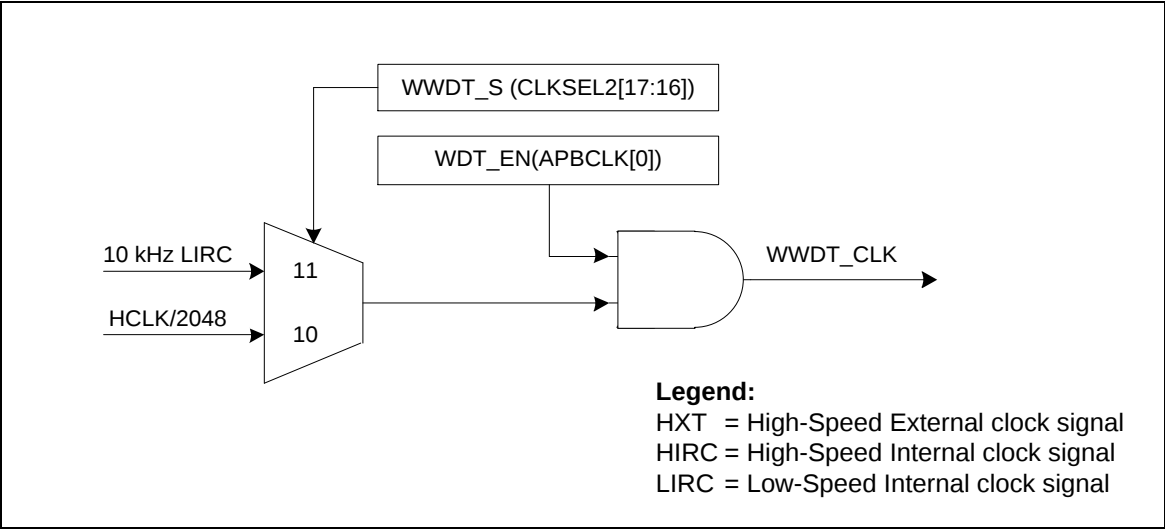


图 6.9-1 窗口看门狗定时器时钟控制

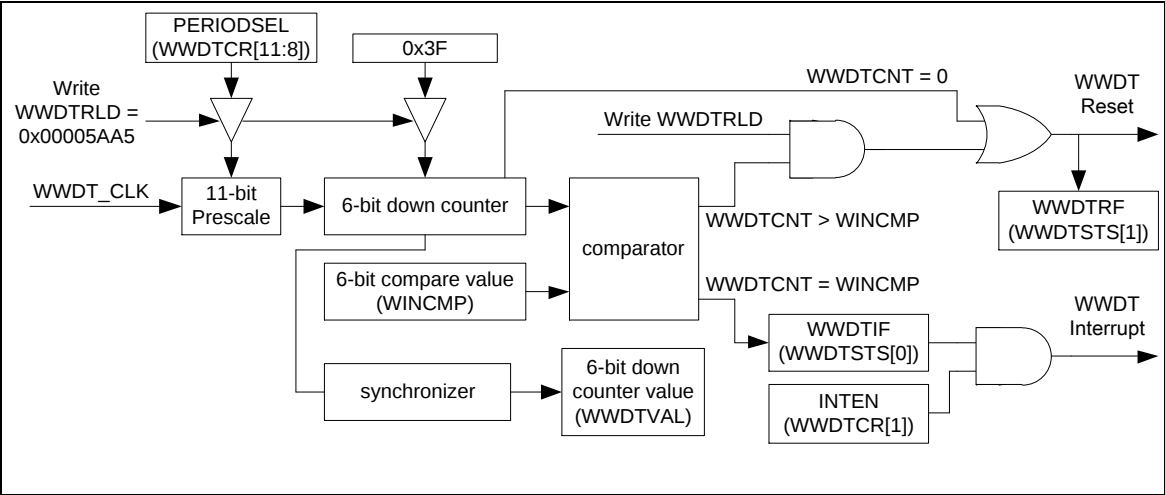


图 6.9-2 窗口看门狗定时器时钟框图

6.9.4 基本配置

窗口看门狗定时器外设时钟源通过APBCLK[0]来使能，通过CLKSEL2[17:16]来选择

6.9.5 功能描述

窗口看门狗定时器(WWDT)包含一个6位向下计数器，该计数器带一个可选择预分频值，不同的预分频值对应不同的看门狗溢出时间间隔。

6位窗口看门狗定时器的时钟源可以是系统时钟经2048 (HCLK/2048)分频的时钟，也可以是内部10k低速RC振荡时钟源，看门狗的时钟源带一个可选择的11位预分频值，该值可通过PERIODSEL来设置选择，对应预分频值如下表

周期选择	预分频值	最大溢出间隔周期	最大溢出时间间隔 (WWDT_CLK=10 kHz)
0000	1	$1 * 64 * T_{WWDT}$	6.4 ms
0001	2	$2 * 64 * T_{WWDT}$	12.8 ms
0010	4	$4 * 64 * T_{WWDT}$	25.6 ms
0011	8	$8 * 64 * T_{WWDT}$	51.2 ms
0100	16	$16 * 64 * T_{WWDT}$	102.4 ms
0101	32	$32 * 64 * T_{WWDT}$	204.8 ms
0110	64	$64 * 64 * T_{WWDT}$	409.6 ms
0111	128	$128 * 64 * T_{WWDT}$	819.2 ms
1000	192	$192 * 64 * T_{WWDT}$	1.2288 s
1001	256	$256 * 64 * T_{WWDT}$	1.6384 s
1010	384	$384 * 64 * T_{WWDT}$	2.4576 s
1011	512	$512 * 64 * T_{WWDT}$	3.2768 s
1100	768	$768 * 64 * T_{WWDT}$	4.9152 s
1101	1024	$1024 * 64 * T_{WWDT}$	6.5536 s
1110	1536	$1536 * 64 * T_{WWDT}$	9.8304 s
1111	2048	$2048 * 64 * T_{WWDT}$	13.1072 s

表 6.9-1 窗口看门狗定时器预分频值选择

窗口看门狗定时器的计数

当WWDTEN位被使能，窗口看门狗向下计数器将会从0x3F向下递减计数到0。为了防止程序在非用户指定位置关闭窗口看门狗定时器，窗口看门狗定时器控制寄存器WWDTCCR在芯片上电或复位后仅

可写一次。当WWDTEN位被软件使能以后，用户不能禁止窗口看门狗定时器 (WWDTEN)，不能修改计数器预分频周期(PERIODSEL)，也不能修改窗口比较值 (WINCMP)，除非芯片复位。

窗口看门狗定时器比较匹配中断

窗口看门狗定时器向下计数过程中，当窗口看门狗定时器计数值(WWDTCVAL) 等于比较值 WINCMP时， WWDTIF会被置1，并且WWDTIF可以被软件清零;如果WWDTIE位被使能，当 WWDTIF 位被硬件置1时，就会产生窗口看门狗比较匹配中断。

窗口看门狗定时器复位系统

当WWDTIF产生后，用户必须通过对WWDTRLD写0x00005AA5，进行重载计数器，使它的值回到初始值0x3F，也可以防止窗口看门狗定时器的值继续向下计数到0，从而产生窗口看门狗定时器复位系统信号通知系统复位。

如果寄存器WWDTCVAL的当前值大于比较寄存器WINCMP的值，此时如果用户对WWDTRLD寄存器写0x00005AA5，窗口看门狗定时器复位系统信号将会立即产生，并导致芯片复位

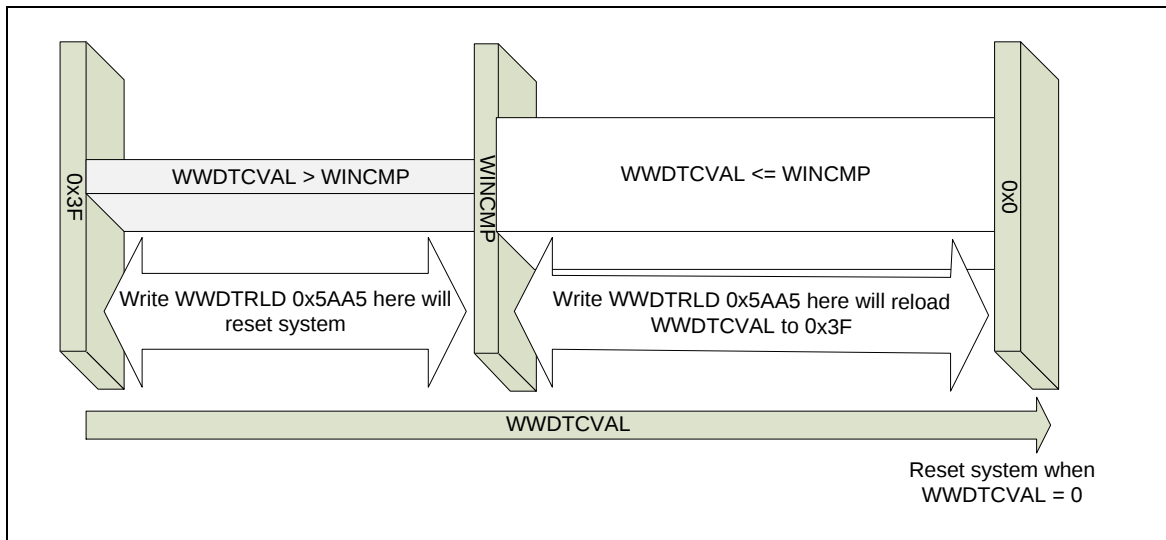


图 6.9-3 窗口看门狗定时器复位和重载过程

窗口看门狗定时器的窗口设置限制

当用户对WWDTRLD寄存器写0x00005AA5来重载WWDTRLD的值到0x3F的时候，大概需要3个窗口看门狗定时器时钟来同步重载命令到实际执行重载操作。这意味着如果用户设置PERIODSEL的值为0000，即计数器预分频值设置为1，WINCMP寄存器的值必须大于2；否则，当WWDTRIF标志产生后，写WWDTRLD来重载窗口看门狗定时器的值为0x3F操作无效，将会一直出现窗口看门狗定时器引起的系统复位。

周期选择	预分频值	有效的WINCMP比较值
0000	1	0x3 ~ 0x3F
0001	2	0x2 ~ 0x3F
其它	其它	0x0 ~ 0x3F

表 6-9 WINCMP 寄存器值设置限制

6.9.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移地址	R/W	描述	复位值
WWDTR 基地址: WWDTR_BA = 0x4000_4100				
WWDTRLD	WWDTR_BA+0x00	W	窗口看门狗定时器重载计数器寄存器	0x0000_0000
WWDTCR	WWDTR_BA+0x04	R/W	窗口看门狗定时器控制寄存器	0x003F_0800
WWDTSR	WWDTR_BA+0x08	R/W	窗口看门狗定时器状态寄存器	0x0000_0000
WWDTCVR	WWDTR_BA+0x0C	R	窗口看门狗定时器计数器值寄存器	0x0000_003F

6.9.7 寄存器描述

窗口看门狗定时器重载计数器寄存器 (WWDTRLD)

寄存器	偏移地址	R/W	描述	复位值
WWDTRLD	WWDT_BA+0x00	W	窗口看门狗定时器重载计数器寄存器	0x0000_0000

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

位	描述	
[31:0]	WWDTRLD	窗口看门狗定时器重载计数器寄存器 对此寄存器写0x00005AA5 将会重载窗口看门狗定时器计数器的值到0x3F. 注意:用户只能是当窗口看门狗计数器的值在0到WINCMP之间时, 才可以写WWDTRLD寄存器来重载窗口看门狗计数器。如果窗口看门狗计数器的值大于WINCMP时写WWDTRLD寄存器, 窗口看门狗定时器复位信号会立即产生

窗口看门狗定时器控制寄存器 (WWDTCR)

寄存器	偏移地址	R/W	寄存器	复位值
WWDTCR	WWDT_BA+0x04	R/W	窗口看门狗定时器控制寄存器	0x003F_0800

注意:此寄存器当芯片上电或复位后, 只能写一次

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

位	描述	
[31]	DBGACK_WWDT	ICE 调试模式下窗口看门狗计数控制位 0 =ICE 调试模式下影响窗口看门狗定时器计数, 当CPU由ICE停止后, 窗口看门狗向下计数器计数值将保持 1 = ICE 调试模式下不影响窗口看门狗定时器计数, 窗口看门狗定时器向下计数器将会保持继续计数, 不管CPU是否受ICE保持。
[30:22]	保留	保留
[21:16]	WINCMP	窗口看门狗定时器窗口比较寄存器 设置此寄存器来调整有效重载窗口 注意: 当当前窗口看门狗定时器计数器的值在0到WINCMP之间时, 用户才可以写WWDTRLD来重载窗口看门狗定时器计数器的值。如果当前窗口看门狗定时器计数器的值大于WINCMP时, 用户去写WWDTRLD寄存器, 窗口看门狗定时器复位信号会立即产生。
[15:12]	保留	保留
[11:8]	PERIODSEL	窗口看门狗定时器计数器预分频周期选择 0000 = 预分频为1; 最大定时溢出周期是 $1 * 64 * TWWDT$. 0001 = 预分频为2; 最大定时溢出周期是 $2 * 64 * TWWDT$. 0010 = 预分频为4; 最大定时溢出周期是 $4 * 64 * TWWDT$. 0011 = 预分频为8; 最大定时溢出周期是 $8 * 64 * TWWDT$. 0100 = 预分频为16; 最大定时溢出周期是 $16 * 64 * TWWDT$. 0101 = 预分频为32; 最大定时溢出周期是 $32 * 64 * TWWDT$. 0110 = 预分频为64; 最大定时溢出周期是 $64 * 64 * TWWDT$. 0111 = 预分频为128; 最大定时溢出周期是 $128 * 64 * TWWDT$. 1000 = 预分频为192; 最大定时溢出周期是 $192 * 64 * TWWDT$.

		1001 = 预分频为256; 最大定时溢出周期是$256 * 64 * TWWDT$. 1010 = 预分频为384; 最大定时溢出周期是$384 * 64 * TWWDT$. 1011 = 预分频为512; 最大定时溢出周期是$512 * 64 * TWWDT$. 1100 = 预分频为768; 最大定时溢出周期是$768 * 64 * TWWDT$. 1101 = 预分频为1024; 最大定时溢出周期是$1024 * 64 * TWWDT$. 1110 = 预分频为1536; 最大定时溢出周期是$1536 * 64 * TWWDT$. 1111 = 预分频为2048; 最大定时溢出周期是$2048 * 64 * TWWDT$.
[7:2]	保留	保留
[1]	WWDTIE	窗口看门狗定时器中断使能位 如果该位被使能, 当有窗口看门狗定时器计数器比较匹配中断信号产生时, 就会通知CPU 0 = 窗口看门狗定时器计数器比较匹配中断禁止 1 = 窗口看门狗定时器计数器比较匹配中断使能
[0]	WWDTEN	窗口看门狗定时器使能位 设置该位来使能窗口看门狗定时器计数器计数 0 = 窗口看门狗定时器计数器停止. 1 = 窗口看门狗定时器计数器计数开始

窗口看门狗定时器状态寄存器 (WWDTSR)

寄存器	偏移地址	R/W	描述	复位值
WWDTSR	WWDT_BA+0x08	R/W	窗口看门狗定时器状态寄存器	0x0000_0000

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

位	描述	
[31:2]	保留	保留
[1]	WWDTRF	窗口看门狗定时器溢出复位标志 该位指示系统是否已被窗口看门狗定时器溢出复位 0 = 窗口看门狗定时器没有发生复位 1 = 窗口看门狗定时器发生了复位 注意：该位是写1清零
[0]	WWDTIF	窗口看门狗定时器比较匹配中断标志 该位指示当窗口看门狗定时器计数器的值与比较值匹配时，窗口看门狗的中断状态标志 0 = 窗口看门狗定时器计数器的值与WINCMP寄存器的值不匹配 1 = 窗口看门狗定时器计数器的值与WINCMP寄存器的值匹配 注意：该位是写1清零

窗口看门狗定时器计数器值寄存器 (WWDTTCVR)

寄存器	偏移地址	R/W	描述	复位值
WWDTTCVR	WWDT_BA+0x0C	R	窗口看门狗定时器计数器值寄存器	0x0000_003F

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

位	描述	
[31:6]	保留	保留
[5:0]	WWDTTCVAL	窗口看门狗定时器计数器值 WWDTTCVAL将会被持续更新，以指示6位向下计数器的值

6.10 串口控制器 (UART)

6.10.1 概述

NuMicro M058S系列提供最多2信道异步串行接口。UART控制器为普通串口，支持硬件流控功能。UART控制器的接收过程是把外设的串行数据转为并行数据，发送过程是把CPU的并行数据转成串行数据发送出去。UART控制器也支持IrDA串行功能、LIN主/从功能，和RS-485功能模式。每个UART通道支持七种类型的中断。

6.10.2 特性

- 全双工，异步通信口
- 收和发各16个字节的FIFO缓冲区
- 支持硬件自动流控功能(CTS, RTS)，RTS自动流控触发电平可设
- 接收缓存的触发等级的数据长度可设
- 每个通道的波特率可单独设置
- 支持CTS引脚触发唤醒功能
- 支持8位的超时溢出，用于接收缓存的检测功能
- 可通过设置DLY (UA_TOR [15:8])寄存器的相应位，来设置两个数据间（从上一个stop 位到下一个start位之间）的时间间隔
- 支持break错误，帧错误，校验错误和收发缓冲区溢出检测等功能
- 可编程串行接口特性
 - 数据位长度可设为5~8位
 - 校验位可设为，奇、偶校验、无校验或 固定校验位的产生和检测
 - 可设置停止位的长度为，1位,1.5位或2位
- 支持 IrDA SIR 功能模式
 - 支持正常模式下3/16位宽功能
- 支持 LIN 功能模式
 - 支持LIN 主/从模式
 - 支持传输中产生break功能可设
 - 支持接收器break检测功能可设
- 支持 RS-485 功能模式
 - 支持 RS-485 9位模式
 - 支持软硬件控制使能RTS管脚来直接控制RS-485传输方向

6.10.3 框图

串口时钟控制和模块框图如下

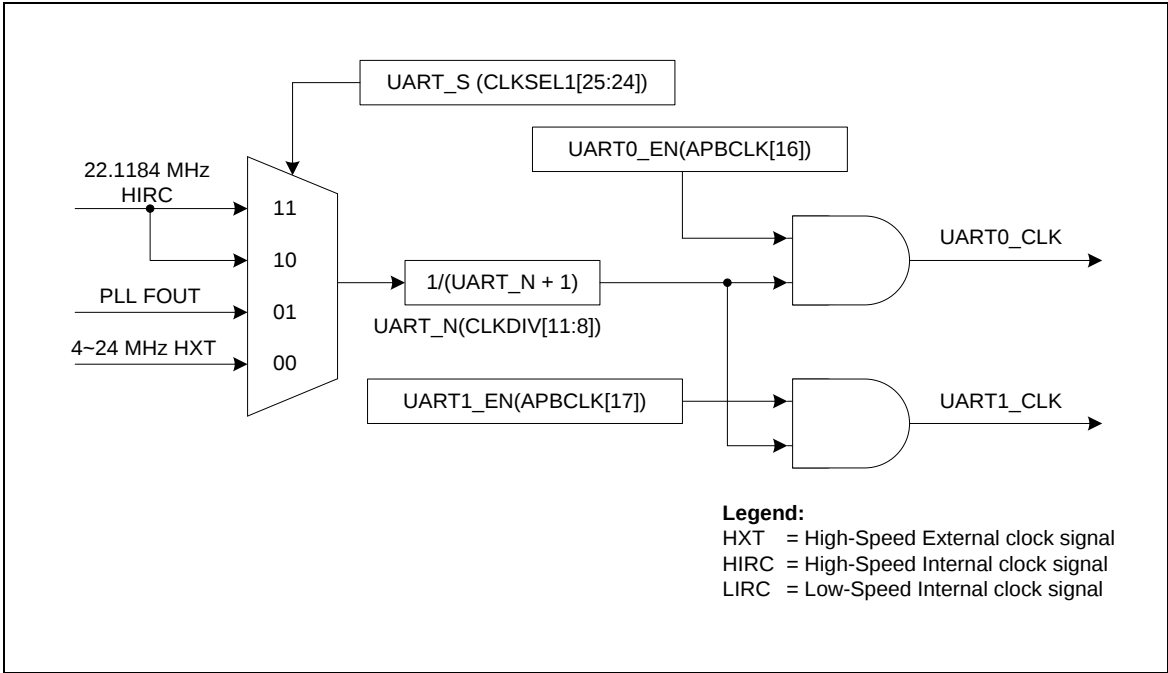


图 6.10-1 串口时钟控制框图

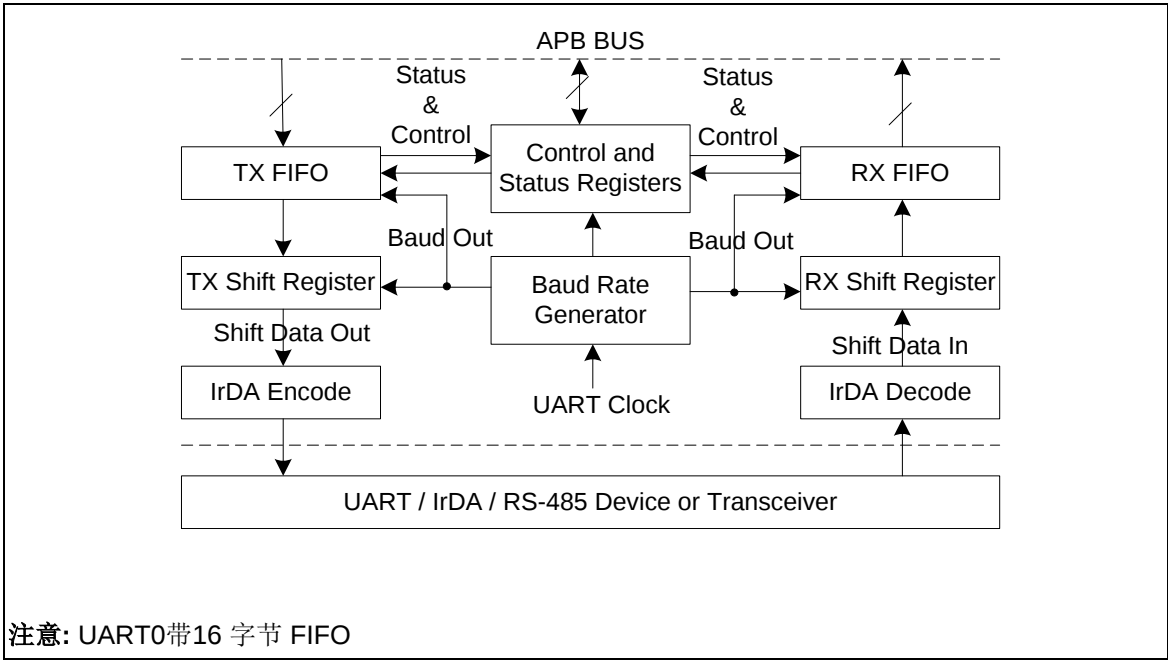


图 6.10-2 串口控制器框图

每个模块功能详细描述如下:

TX_FIFO

发送器带有一个16字节的FIFO缓冲区可以减少CPU中断的频率

RX_FIFO

接收器带有一个16字节的FIFO缓冲区（每个字节加三个错误位）可以减少CPU的中断频率

发送移位寄存器

该模块用于控制把并行数据串行输出

接收移位寄存器

该模块用于控制把串行数据并行输入

Modem控制寄存器

该寄存器用于控制到Modem或数传机（类似于Modem的外设）的接口

波特率发生器

通过把输入的外部时钟除频得到期望的波特率。详情请参考波特率公式

IrDA 编码

该模块是IrDA编码控制模块

IrDA 解码

该模块是IrDA解码控制模块

控制和状态寄存器

该区域是用于发送器和接收器的寄存器组。包含FIFO控制寄存器(UA_FCR)，FIFO状态寄存器(UA_FSR),和线控寄存器(UA_LCR)。超时控制寄存器(UA_TOR)用于标识超时中断产生的条件。该寄存器组还包括中断使能寄存器(UA_IER)和中断状态寄存器(UA_ISR)，以使能或禁止相应中断,中断源查询。一共有7种类型的中断，包括发送FIFO空中断(THRE_INT),接收达到阈值(RDA_INT)中断，线状态中断(校验错误或帧错误或break中断) (RLS_INT)，超时中断(TOUT_INT)，MODEM/唤醒状态中断 (MODEM_INT)，缓存错误中断 (BUF_ERR_INT) 和 LIN接收器break域检测中断 (LIN_INT)

下面为自动流控功能模块框图

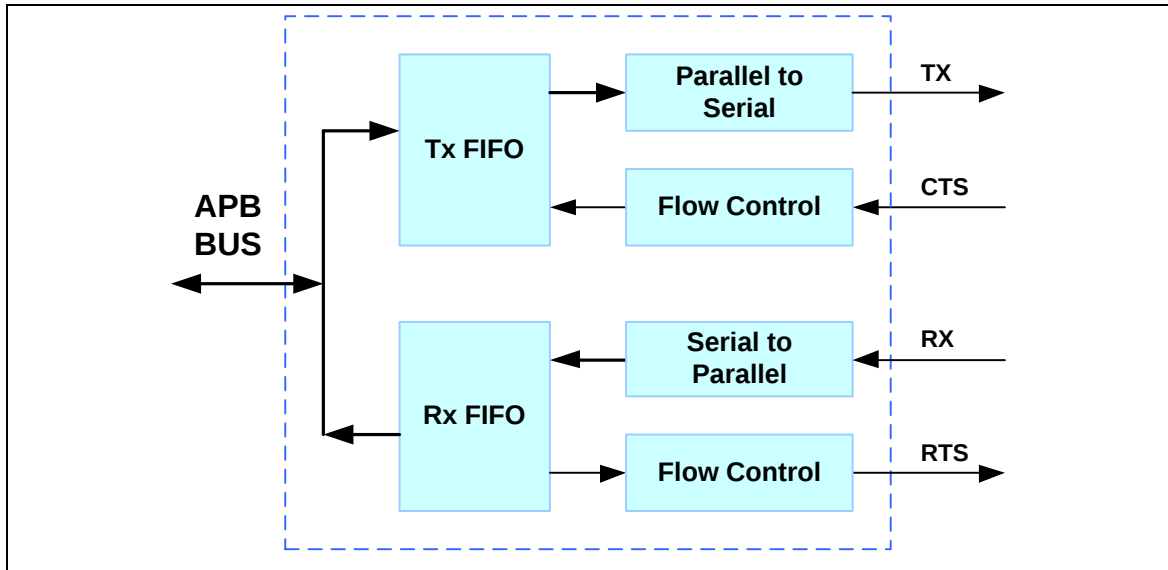


图 6.10-3 自动流控模块框图

6.10.4 基本配置

UART控制器功能脚在P0_MFP寄存器中配置

UART控制器时钟，由UART0_EN(APBCLK[16])中使能UART0。

UART控制器时钟源通过UART_S(CLKSEL1[25:24])位来选择。

UART控制器时钟预分频通过UART_N(CLKDIV[11:8])位来设置

6.10.5 功能描述

UART控制器支持四个功能模式，包括UART，IrDA，LIN和RS-485模式。用户可以通过设置寄存器UA_FUN_SEL来选择功能。四种功能模式的详细描述在6.10.5.1到6.10.5.4章节。同样更多信息请参看6.10.5.5到6.10.5.8章节。

6.10.5.1 串口控制器波特率发生器

UART控制器包含一个可编程波特率发生器，其通过分频器对输入时钟源分频，得到收发数据所需的串行时钟。波特率计算公式：波特率= $UART_CLK / M * [BRD + 2]$ ，这里M和BRD在波特率分频器寄存器(UA_BAUD)中定义。下表列出了各种条件下的UART波特率的计算公式和参数设置。通过设置相应的波特率参数和寄存器可以得到零误差的波特率。IrDA功能模式，波特率发生器必须是模式0。

模式	DIV_X_EN	DIV_X_ONE	除数 X	BRD	M	波特率公式
模式 0	0	0	B	A	16	$UART_CLK / [16 * (A+2)]$
模式 1	1	0	B	A	B+1	$UART_CLK / [(B+1) * (A+2)]$, B 必须 ≥ 8
模式 2	1	1	无关	A	1	$UART_CLK / (A+2)$, A 必须 ≥ 8

表 6.10-1 UART 控制器波特率公式表

UART 外围时钟= 22.1184 MHz			
波特率	模式 0	模式 1	模式 2
921600	不支持	A=0, B=11	A=22
460800	A=1	A=1, B=15 A=2, B=11	A=46
230400	A=4	A=4, B=15 A=6, B=11	A=94
115200	A=10	A=10, B=15 A=14, B=11	A=190
57600	A=22	A=22, B=15 A=30, B=11	A=382
38400	A=34	A=62, B=8 A=46, B=11 A=34, B=15	A=574
19200	A=70	A=126, B=8 A=94, B=11 A=70, B=15	A=1150
9600	A=142	A=254, B=8 A=190, B=11 A=142, B=15	A=2302
4800	A=286	A=510, B=8 A=382, B=11 A=286, B=15	A=4606

表 6.10-2 UART 控制器波特率参数设置表

UART 外围时钟= 22.1184 MHz			
波特率	模式 0	模式 1	模式 2
921600	不支持	0x2B00_0000	0x3000_0016
460800	0x0000_0001	0x2F00_0001 0x2B00_0002	0x3000_002E
230400	0x0000_0004	0x2F00_0004 0x2B00_0006	0x3000_005E
115200	0x0000_000A	0x2F00_000A 0x2B00_000E	0x3000_00BE
57600	0x0000_0016	0x2F00_0016 0x2B00_001E	0x3000_017E
38400	0x0000_0022	0x2800_003E 0x2B00_002E 0x2F00_0022	0x3000_023E
19200	0x0000_0046	0x2800_007E 0x2B00_005E 0x2F00_0046	0x3000_047E
9600	0x0000_008E	0x2800_00FE 0x2B00_00BE 0x2F00_008E	0x3000_08FE
4800	0x0000_011E	0x2800_01FE 0x2B00_017E 0x2F00_011E	0x3000_11FE

表 6.10-3 UART 控制器波特率寄存器设置表

6.10.5.2 UART 控制器FIFO控制和状态

UART控制器内置一个16字节的发送FIFO (TX_FIFO)和一个16字节的接收FIFO (RX_FIFO), 通信中,使用这些收发FIFO缓冲寄存器可以减少对CPU的中断次数。CPU在任何时候都可以读取UART的状态。状态信息包括UART传输操作的类型和条件, 以及接收过程有可能发生的3个错误状态(奇偶校验错误, 帧错误, Break中断)。FIFO控制和状态也支持所有的功能模式, 包括UART, IrDA, LIN和RS-485模式。

6.10.5.3 UART 控制器唤醒功能

当芯片在Power-down模式时, 外部CTS脚上电平变化可以唤醒芯片。该功能在所有功能模式下都有效。用户如果要使用唤醒功能还必须使能MODEN_INT中断。

6.10.5.4 UART 控制器中断和状态

每个UART控制器支持7种类型的中断，它包括：

- 接收阈值水平达到后的中断(RDA_INT)
- 发送FIFO空中断(THRE_INT)
- Line状态中断（奇偶校验错误，帧错误，Break中断）(RLS_INT)
- MODEM/唤醒状态中断(MODEM_INT)
- 接收缓冲区超时溢出中断(TOUT_INT)
- 缓冲区错误中断(BUF_ERR_INT)
- LIN 接收器Break域检测中断(LIN_RX_BREAK_INT)

下表描述了中断源和中断标志。当中断使能且有中断标志时就会产生中断。用户必须在中断后清中断标志

中断源	中断指示	中断使能位	中断标志	清中断标志方法
接收数据有效中断	RDA_INT	RDA_IEN	RDA_IF	读 UA_RBR
发送保持寄存器空中断	THRE_INT	THRE_IEN	THRE_IF	写 UA_THR
接收线状态中断	RLS_INT	RLS_IEN	RLS_IF = (BIF or FEF or PEF)	写 '1' 到 BIF/FEF/ PEF
			RLS_IF = RS485_ADD_DETF	写 '1' 到 RS485_ADD_DETF
Modem 状态中断	MODEM_INT	MODEM_IEN	MODEM_IF = DCTSIF	写 '1' 到 DCTSIF
RX 超时溢出中断	TOUT_INT	RTO_IEN	TOUT_IF	读 UA_RBR
缓存错误中断	BUF_ERR_INT	BUF_ERR_IEN	BUF_ERR_IF = (TX_OVER_IF 或 RX_OVER_IF)	写 '1' 到 TX_OVER_IF / RX_OVER_IF
LIN 接收Break 域检测中断	LIN_RX_BREAK_INT	LIN_RX_BRK_IEN	LIN_RX_BREAK_IF	写 '1' 到 LIN_RX_BREAK_IF

表 6.10-4 UART 控制器中断源和标志列表

6.10.5.5 UART 功能模式

UART控制器提供了UART功能（用户须设置UA_FUN_SEL [1:0] 为“00”设置为UART功能）。
UART 波特率最高速度是1 Mbps.

UART为全双工异步通讯接口。收发器各包含一个16字节的FIFO缓冲区。用户可以设置接收时的FIFO触发阈值以及超时溢出检测时间。发送数据帧间（即从上一帧停止位到下一帧起始位）的时间间隔可通过DLY (UA_TOR [15:8])位可设。UART支持硬件自动流控功能(CTS, RTS),且RTS流控触发电平可设，全双工串行接口通信参数可设。

UART 线控制功能

UART控制器通过设置UA_LCR寄存器支持串行接口全部特性。可以通过对UA_LCR寄存器设置数据位和停止位长度以及奇偶校验。下表列出了UART数据位和停止位长度的设置以及UART奇偶校验位的设置

NSB (UA_LCR[2])	WLS (UA_LCR[1:0])	数据位长度(bit)	停止位长度 (bit)
0	00	5	1
0	01	6	1
0	10	7	1
0	11	8	1
1	00	5	1.5
1	01	6	2
1	10	7	2
1	11	8	2

表 6.10-5 UART 线控制的数据位和停止位长度设置

奇偶校验类型	SPE (UA_LCR[5])	EPE (UA_LCR[4])	PBE (UA_LCR[3])	描述
无校验	x	x	0	无奇偶校验位输出
奇校验	0	0	1	奇校验位的计算方法是把数据流中的所有的1相加，使得包括校验位在内1的总数为奇数个
偶校验	0	1	1	偶校验位的计算方法是把数据流中的所有的1相加，使得包括校验位在内1的总数为偶数个。
奇偶校验位强制置1	1	0	1	奇偶校验位总是逻辑1 不管数据位中1的个数是多少（奇偶计数），奇偶校验位永远都是逻辑1
奇偶校验位强制为0	1	1	1	奇偶校验位总是逻辑0 不管数据位中1的个数是多少（奇偶计数），奇偶校验位永远都是逻辑0

表 6.10-6 UART 线控制的奇偶校验位设置

UART 自动流控功能

UART支持自动流控功能，该功能用到两根信号线CTS (清零发送) 和RTS (请求发送)来控制UART与外部设备（如Modem）间的数据传输。当自动流控使能后，只有等到UART对外部设备发出有效的RTS信号后才允许接收数据，否则不接收。当RX FIFO接收到数据的数量达等于RTS_TRI_LEV (UA_FCR [19:16])位的值后,RTS信号会被取消。当UART检测到外部设备给CTS信号脚有效信号后，UART开始发送数据。否则UART不会发送数据。

以下框图展示了UART CTS自动流控功能模式。用户必须要先设置AUTO_CTS_EN (UA_IER [13])以使能CTS自动流控功能。LEV_CTS (UA_MCR [8])位可以设置CTS脚输入的有效状态。当CTS脚上任何电平变化将导致DCTSIF (UA_MSR [0])位被置1，然后TX脚将从TX FIFO自动发出数据。

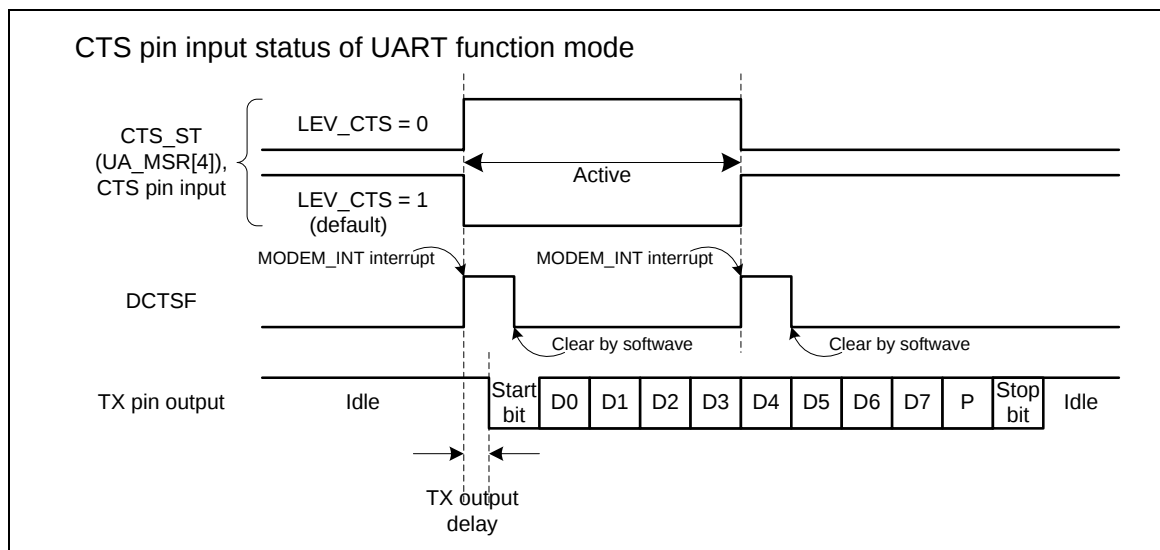


图 6.10-4 UART CTS 自动流控使能

如下图所示，UART RTS自动流控模式(AUTO_RTS_EN (UA_IER[12])=1)中，nRTS的触发阈值由UART FIFO控制寄存器的RTS_RTI_LEV(UA_FCR[19:16])位来控制。

设置LEV_RTS(UA_MCR[9])位可以控制RTS脚的反向或非反向。用户可以读RTS_ST(UA_MCR[13])位来知道真实RTS脚输出电压的逻辑状态。

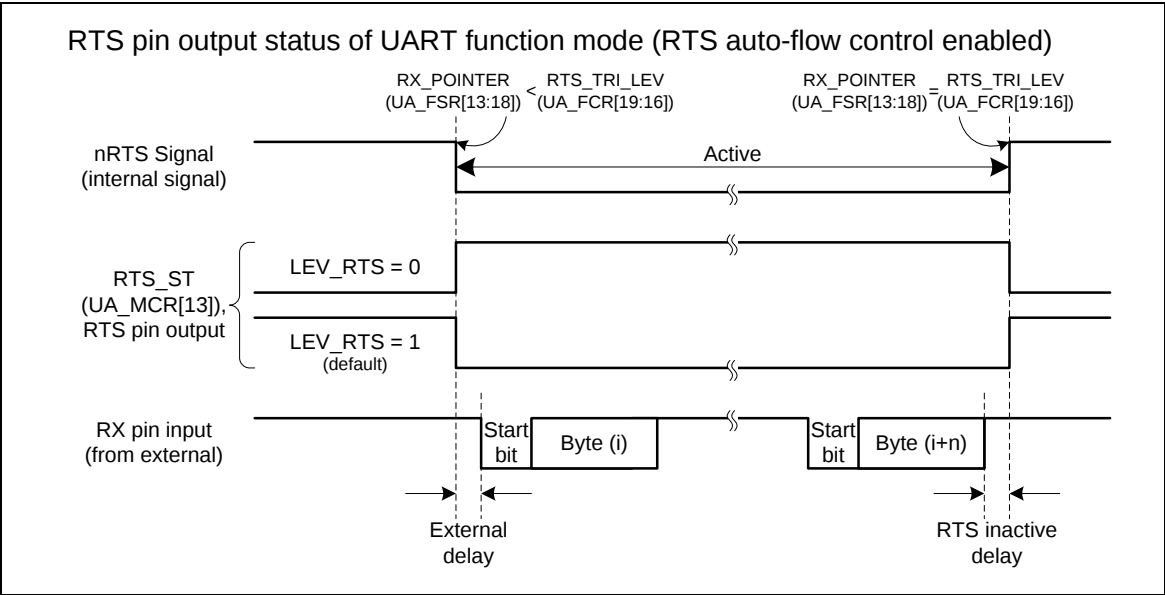


图 6.10-5 UART RTS 自动流控使能

如下图所示，在软件模式下(AUTO_RTS_EN(UA_IER[12])=0)，RTS流控直接由RTS(UA_MCR[1])位软件来控制。

设置LEV_RTS(UA_MCR[9])位可以控制RTS输出脚状态与RTS(UA_MCR[1])控制状态是同向还是反向。用户可以读RTS_ST(UA_MCR[13])位来获取RTS脚真实输出电平状态。

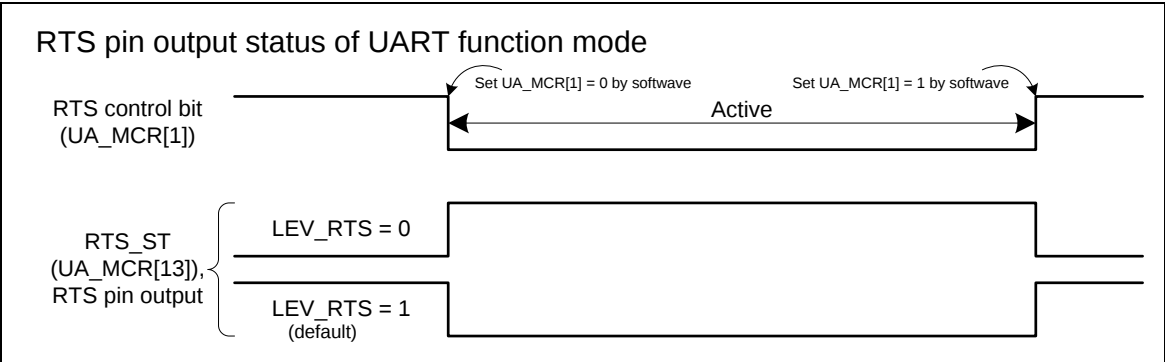


图 6.10-6 UART RTS 软件控制的流控

6.10.5.6 IrDA 功能模式

UART控制器也提供Serial IrDA (SIR, 串行红外)功能(用户必须设置UA_FUN_SEL [1:0] 为'10')来使用IrDA功能。SIR规范定义了一个短距离红外异步串行传输模式, 包括一个起始位, 8个数据位, 一个停止位。最大速率115.2kbps。IrDA SIR模块包含一个IrDA SIR协议编/解码器。IrDA SIR协议是半双工工作模式。所以它不能同时收发数据。IrDA SIR物理层规定了发送与接收数据的时间上至少10ms的时间间隔, 该延迟特性需通过软件来完成。

IrDA 模式下, DIV_X_EN (UA_BAUD [29])位需被禁止。

波特率= Clock / (16 * BRD), 这里BRD是UA_BAUD寄存器中定义的波特率分频器。

以下框图展示了IrDA控制模块框图

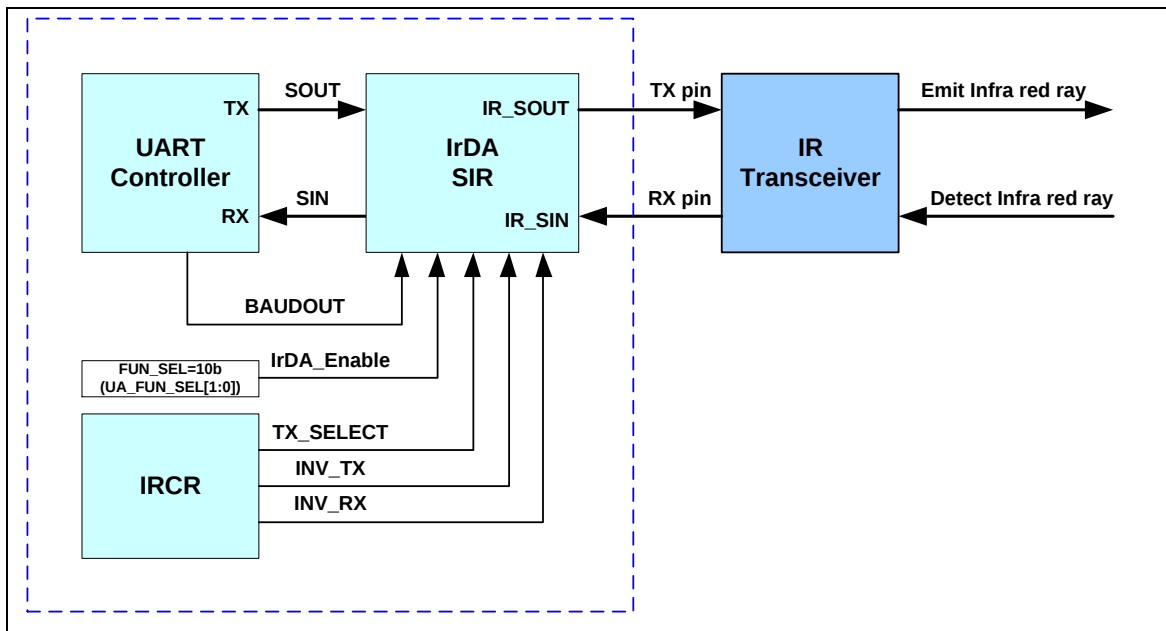


图 6.10-7 IrDA 控制模块框图

IrDA SIR 发送编码

IrDA SIR 传送编码调制采用 Non-Return-to Zero (NRZ)编码, 将数据流编码后从UART输出。IrDA SIR 物理层指定使用归零反向调制编码 (Return-to-Zero, Inverted (RZI)), 用一个红外光脉冲代表逻辑 0, 被调整的脉冲输出到外部输出驱动器和红外线发光二极管。

在正常模式下, 传输脉冲的宽度为 3/16 波特率周期。

IrDA SIR 接收编码

IrDA SIR 接收解码器对输入管脚的(Return-to-Zero, Inverted (RZI))串行位流进行解调, 并输出NRZ 串行位流到 UART接收数据输入端。在空闲状态里, 解码器输入端通常为高。(因此, ICR (INV_RX [6])位默认为1)。

当解码器输入端为低时, 表明接收到一个起始位。

IrDA SIR 操作

IrDA SIR 编码/解码提供 UART 数据流和半双工串行 SIR 之间的转换功能。IrDA编码/解码波形图如下：

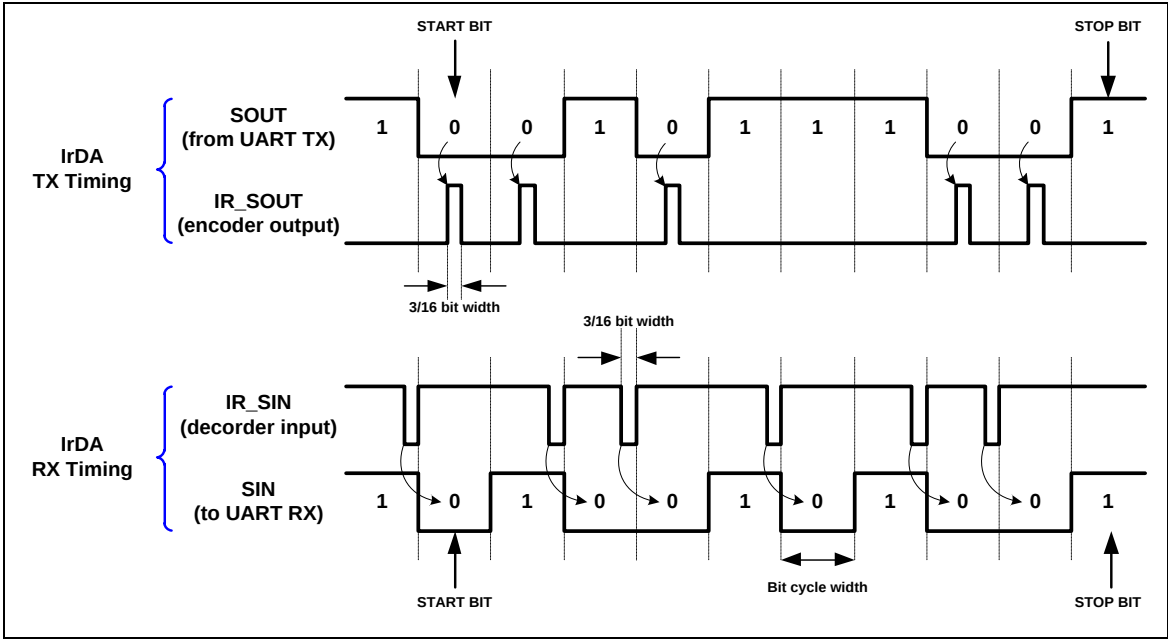


图 6.10-8 IrDA TX/RX 时序图

6.10.5.7 LIN (本地互连网络) 功能模式

UART控制器支持LIN功能，用户需通过设置FUN_SEL (UA_FUN_SEL[1:0])为 '01'来将UART使能为LIN模式。在LIN功能模式，根据LIN的标准，每个字节由值为0（显性）的START位开始，接着是8位数据位，没有奇偶校验位，LSB优先，由一个值为1（隐性）的STOP位结束。下图是LIN功能模式帧的结构。

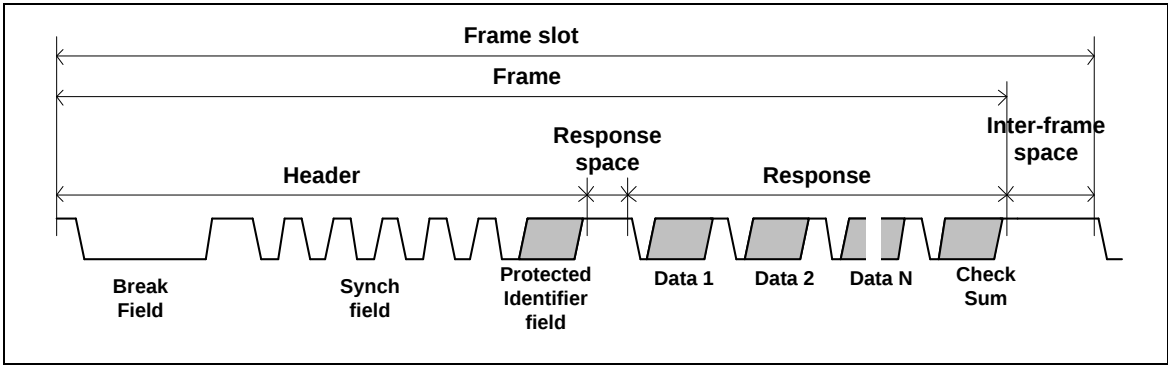


图 6.10-9 LIN 的帧结构

总线发送 (TX)编程流程如下:

1. 设置UA_FUN_SEL[1:0] 为 '01' 使能 LIN 总线模式
2. 填UA_LIN_BCNT寄存器中的UA_LIN_BKFL来选择break域的长度 (break 域的长度为 UA_LIN_BKFL + 2)
3. 填0x55 到UA_THR寄存器请求发送synch域
4. 将protected identifier 的值填到UA_THR寄存器, 来请求identifier域被发送
5. 设置 UA_LIN_BCNT 寄存器的 LIN_TX_EN 位开始发送 (当 break 域操作结束时, LIN_TX_EN将被自动清除为0)
6. 当THR 最后字节的STOP位被发送到总线上时, 硬件将设定UA_FSR中的TE_FLAG为1
7. 填N 个字节数据和checksum到 UA_THR , 然后重复步骤5 和 6发送数据

LIN总线接收(RX)编程流程如下:

1. 设置UA_FUN_SEL[1:0] 为 '01' 来使能LIN总线模式
2. 设置UA_LIN_BCNT寄存器的LIN_RX_EN位来使能LIN RX 模式
3. 等待UA_ISR中的LIN_RX_BREAK_IF标志位, 来检测是否收到break域
4. 等待UA_ISR的RDA_IF标志被置, 以从UR_RBR 寄存器读回收到的数据

6.10.5.8 RS-485 功能模式

UART控制器另一个可选择的功能是RS-485功能 (用户必须设置UA_FUN_SEL [1:0]为 “11”来使能RS-485功能), 方向控制则是由异步串口的RTS脚来控制或者通过软件编程GPIO(P0.3为RTS0 和 P0.1 为 RTS1)来执行。RS-485收发器的驱动控制是通过RTS控制信号来驱动的。RS-485模式下的RX和TX大多数特性与UART相同。

RS-485模式, 控制器可以配置成 RS-485 可寻址的从机模式, RS-485 主机发送可通过设置奇偶检验位 (9th bit) 为 1标识地址特性。对于数据特性, 奇偶检验位设置为 0。设置寄存器 UA_LCR 控制第9位 (PBE, EPE和SPE被置位时, 第9位发送 0; PBE 和 SPE 置位, EPE清零时, 第 9 位发送 1)。

该控制器支持三种操作模式: RS-485 普通多点操作模式 (NMM), RS-485 自动地址识别模式 (AAD) 和 RS-485 自动方向控制模式 (AUD), 可通过UA_ALT_CSR寄存器的设置选择其中一种工作模式, 通过设置DLY (UA_TOR [15:8])可以设置上一个停止位与下一个开始位之间的延迟时间。

RS-485 普通多点操作模式 (NMM)

RS-485 普通多点操作模式, 首先, 软件必须决定在检测到地址字节之前的数据是否存储到RX-FIFO。如果软件想忽略在检测到地址之前的任何数据, 流程是先设置RX_DIS (UA_FCR [8]), 然后使能RS485_NMM (UA_ALT_CSR [8]), 接收器将会忽略数据, 直到检测到地址字节 (bit9 =1) 并且地址字节数据存储在RX-FIFO。如果软件想接收在检测到地址字节之前的任何数据, 流程是先禁用RX_DIS (UA_FCR [8]), 然后使能RS485_NMM (UA_ALT_CSR [8]), 接收器将接收任何数据。

如果检测到地址字节 (bit9 =1), 会产生一个中断到 CPU, 软件可以通过设置RX_DIS (UA_FCR [8]), 决定是否使能或禁用接收器接收数据。如果使能接收器, 就会接收所有字节数据并存储到RX-FIFO。如果设置RX_DIS (UA_FCR [8])为禁用接收器, 会忽略所有接收到的字节数据, 直到检测到下一个地址字节。当检测到地址字节后, 控制器将清除RX_DIS (UA_FCR [8])位且地址字节数

据将存储到 RX-FIFO

RS-485 自动地址识别工作模式 (AAD)

RS-485 自动地址识别模式，接收器在检测到地址字节 (bit9=1)，并且地址字节数据与 ADDR_MATCH (UA_ALT_CSR[31:24])的值相匹配之前，将忽略所有数据。地址字节数据将存储在 RX FIFO。所有接收字节数据将被接收并存储于 RX-FIFO 直到地址字节数据与ADDR_MATCH (UA_ALT_CSR[31:24])的值不匹配时。

RS-485 自动方向模式 (AUD)

RS-485 控制器的另一个功能是 RS-485 自动方向控制功能，用户必须设置 RS485_AUD(UA_ALT_CSR[10]) = 1来使能RS-485自动方向模式。RS-485 通过 RTS 来驱动控制异步串口的控制信号，使能RS-485 驱动器。RTS 连接到RS-485 驱动器，设置RTS线为高（逻辑1）使能RS-485 驱动器。设置 RTS 为低（逻辑0），使驱动器进入 tri-state 状态。用户通过设置寄存器 UA_MCR 中的 LEV_RTS 位改变 RTS 驱动电平

以下框图展示了AUD模式下RS-485 RTS驱动电平。RTS脚在TX数据发送阶段自动驱动收发器。

设置LEV_RTS(UA_MCR[9])位可以控制RTS脚的输出电平。用户可以通过读 RTS_ST(UA_MCR[13])位来知道RTS脚上实际的输出逻辑电平。

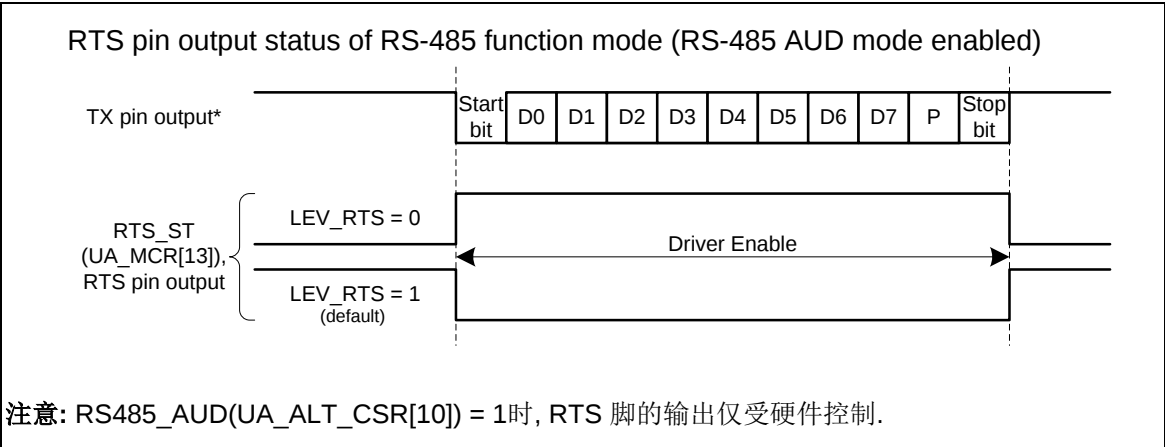


图 6.10-10 RS-485 自动方向模式下 RTS 驱动电平

下图展示了通过软件控制(RS485_AUD(UA_ALT_CSR[10])=0)RS-485 RTS脚的驱动电平。RTS驱动电平通过RTS(UA_MCR[1])位来控制。

设置LEV_RTS(UA_MCR[9])位可以控制RTS脚的输出与RTS(UA_MCR[1])控制位是否反向。用户可以读RTS_ST(UA_MCR[13])位来知道RTS脚上实际的逻辑电平。

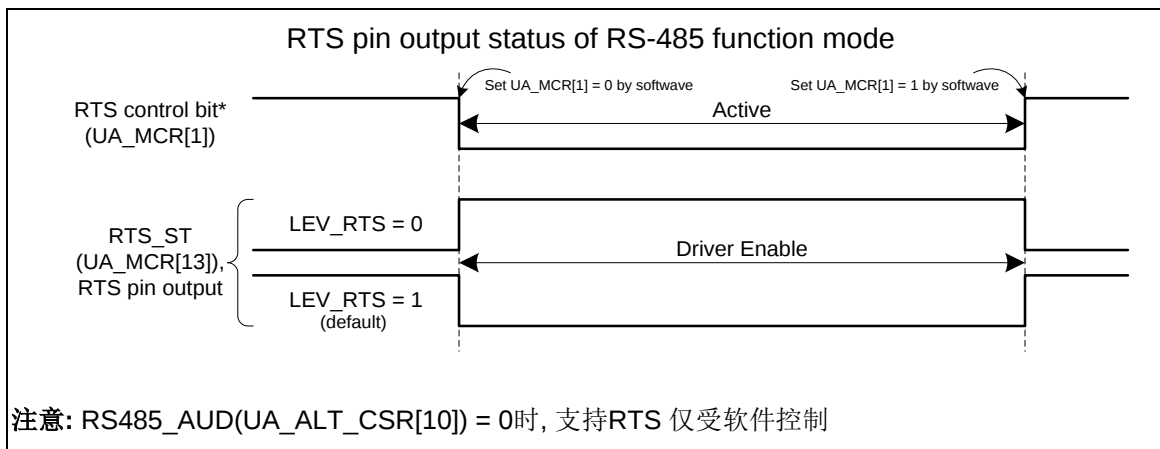


图 6.10-11 RS-485 RTS 软件控制下的驱动电平

编程流程示例:

1. 设置 UA_FUN_SEL 中的 FUN_SEL 选择 RS-485 功能
2. 设置RX_DIS (UA_FCR[8])位使能或禁用 RS-485 接收器
3. 设置RS485_NMM (UA_ALT_CSR[8])或 RS485_AAD (UA_ALT_CSR[9])模式
4. 如果选择RS485_AAD (UA_ALT_CSR[9])模式, ADDR_MATCH (UA_ALT_CSR[31:24])需设置成自动地址匹配值
5. 设置RS485_AUD (UA_ALT_CSR[10])来决定是否为自动方向控制

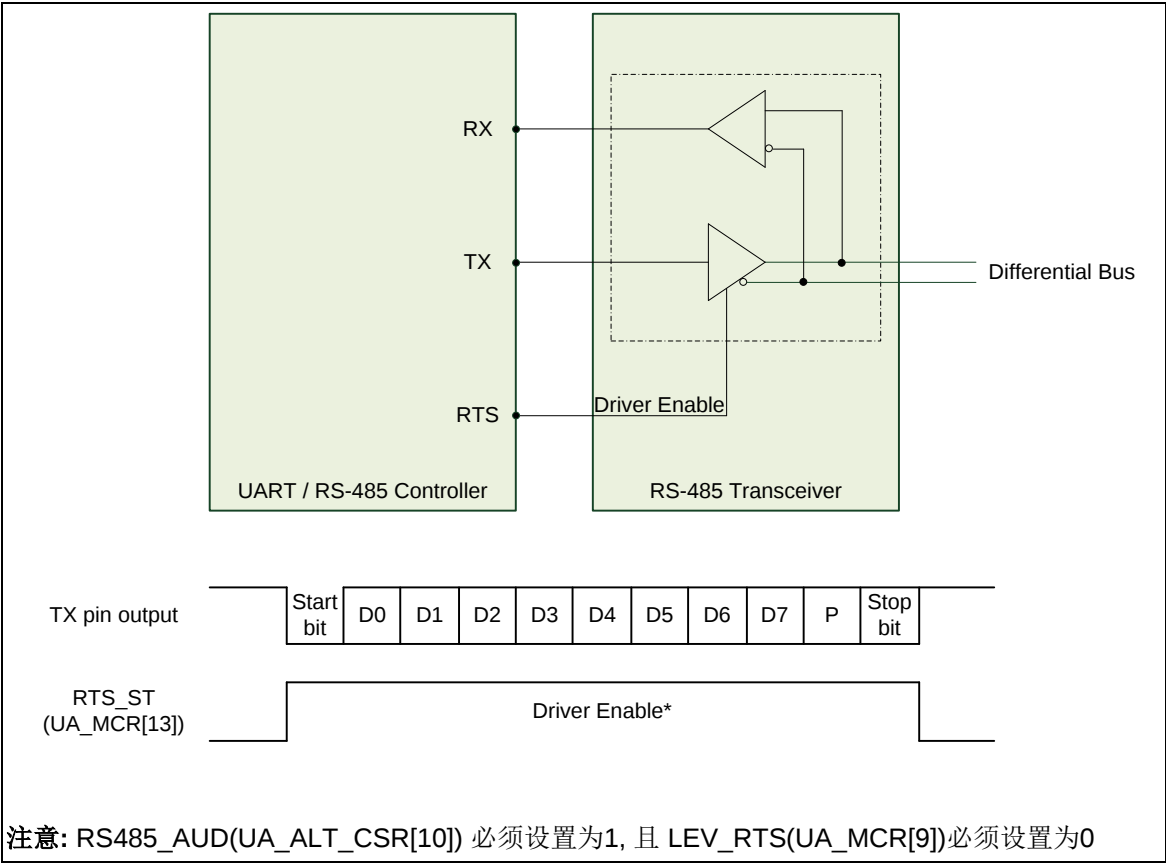


图 6.10-12 RS-485 帧结构

6.10.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移地址	R/W	描述	复位值
UART 基地址:				
UART_BA = 0x4005_0000				
UA_RBR	UARTx_BA+0x00	R	UART 接收缓存寄存器	未定义
UA_THR	UARTx_BA+0x00	W	UART 发送保持寄存器	未定义
UA_IER	UARTx_BA+0x04	R/W	UART 中断使能寄存器	0x0000_0000
UA_FCR	UARTx_BA+0x08	R/W	UART FIFO 控制寄存器	0x0000_0101
UA_LCR	UARTx_BA+0x0C	R/W	UART 线控制寄存器	0x0000_0000
UA_MCR	UARTx_BA+0x10	R/W	UART Modem 控制寄存器	0x0000_0200
UA_MSR	UARTx_BA+0x14	R/W	UART Modem 状态寄存器	0x0000_0110
UA_FSR	UARTx_BA+0x18	R/W	UART FIFO 状态寄存器	0x1040_4000
UA_ISR	UARTx_BA+0x1C	R/W	UART 中断状态寄存器	0x0000_0002
UA_TOR	UARTx_BA+0x20	R/W	UART 超时溢出寄存器	0x0000_0000
UA_BAUD	UARTx_BA+0x24	R/W	UART 波特率分频寄存器	0x0F00_0000
UA_IRCR	UARTx_BA+0x28	R/W	UART IrDA 控制寄存器	0x0000_0040
UA_ALT_CSR	UARTx_BA+0x2C	R/W	UART 选择控制/状态寄存器	0x0000_0000
UA_FUN_SEL	UARTx_BA+0x30	R/W	UART 功能选择寄存器	0x0000_0000

6.10.7 寄存器描述

UART 接收缓存寄存器 (UA_RBR)

寄存器	偏移地址	R/W	描述	复位值
UA_RBR	UARTx_BA+0x00	R	UART 接收缓存寄存器	未定义

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

位	描述	
[31:8]	保留	保留
[7:0]	RBR	接收缓存寄存器(只读) 读该寄存器, UART 将返回从UART_RX脚接收到的8位数据(LSB 优先).

UART 发送保持寄存器 (UA_THR)

寄存器	偏移地址	R/W	描述	复位值
UA_THR	UARTx_BA+0x00	W	UART 发送保持寄存器	未定义

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

位	描述	
[31:8]	保留	保留
[7:0]	THR	发送保持寄存器 写数据到该寄存器, 数据将会保存到发送FIFO. UART控制器将会通过UART_TX脚把存放在FIFO里的最前面的数据发送出去(LSB 优先)

UART 中断使能寄存器 (UA_IER)

寄存器	偏移地址	R/W	描述	复位值
UA_IER	UARTx_BA+0x04	R/W	UART 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留		AUTO_CTS_EN	AUTO_RTS_EN	TIME_OUT_EN	保留		LIN_RX_BRK_IEN
7	6	5	4	3	2	1	0
保留	WAKE_EN	BUF_ERR_IEN	RTO_IEN	MODEM_IEN	RLS_IEN	THRE_IEN	RDA_IEN

位	描述	
[31:14]	保留	保留
[13]	AUTO_CTS_EN	CTS自动流控使能位 0 = CTS自动流控禁止. 1 = CTS自动流控使能. 注意: 当 CTS 自动流控使能后, 当CTS输入有效时, UART 会发送数据到外部设备 (UART将不发送数据到外设直到CTS有效时)。
[12]	AUTO_RTS_EN	RTS 自动流控使能位 0 = RTS 自动流控禁止 1 = RTS 自动流控使能 当 RTS 自动流控使能后, 如果 RX FIFO 中字节的数量等于 RTS_TRI_LEV (UA_FCR [19:16]), UART 会自动禁止 RTS 信号
[11]	TIME_OUT_EN	超时计数器使能位 0 = 超时计数器禁止 1 = 超时计数器使能
[10:9]	保留	保留
[8]	LIN_RX_BRK_IEN	LIN 总线RX Break域检测中断使能位 0 = LIN 总线RX Break域检测中断禁止 1 = LIN 总线RX Break域检测中断使能 注: 该位用于LIN 功能模式
[7]	保留	保留
[6]	WAKE_EN	UART 唤醒功能使能 0 = UART 唤醒功能禁止. 1 =UART 唤醒功能使能

		注意： 当芯片进入掉电模式时，一个外部 CTS 信号的改变会使芯片从掉电模式中唤醒
[5]	BUF_ERR_IEN	缓存错误中断使能 0 = INT_BUF_ERR 禁止 1 = INT_BUF_ERR 使能
[4]	RTO_IEN	RX 超时中断使能位 0 = TOUT_INT 中断关闭. 1 = TOUT_INT 中断使能.
[3]	MODEM_IEN	Modem 状态中断使能位 0 = MODEM_INT 状态中断关闭 1 = MODEM_INT 状态中断使能
[2]	RLS_IEN	接收线状态中断使能位 0 = RLS_INT 关闭. 1 = RLS_INT 使能.
[1]	THRE_IEN	发送保持寄存器空中断使能位 0 = THRE_INT 关闭 1 = THRE_INT 使能.
[0]	RDA_IEN	接收数据可用中断使能位 0 = RDA_INT 关闭 1 = RDA_INT 使能.

UART FIFO 控制寄存器 (UA_FCR)

寄存器	偏移地址	R/W	描述	复位值
UA_FCR	UARTx_BA+0x08	R/W	UART FIFO 控制寄存器	0x0000_0101

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

位	描述	
[31:20]	保留	保留
[19:16]	RTS_TRI_LEV	RTS 自动流控触发阈值 0000 = RTS 触发阈值为1 字节. 0001 = RTS触发阈值为4 字节. 0010 = RTS触发阈值为8 字节. 0011 = RTS触发阈值为14 字节. 其它 = 保留. 注: 该区域用于自动 RTS 流控制
[15:9]	保留	保留
[8]	RX_DIS	接收器禁止设置位 是否禁用接收器 (置 1 禁用接收器) 0 =接收器使能 1 =接收器禁止 注意: 该位用于 RS-485 普通多点模式。需要在 RS-485_NMM (UA_ALT_CSR [8])之前设置
[7:4]	RFITL	RX FIFO 中断 (RDA_INT) 触发级别 当FIFO 接收字节数等于 RFITL 后, RDA_IF 将被置位 (如果UA_IER寄存器的 RDA_IEN位使能, 将产生中断)。 0000 = RX FIFO 触发中断阈值为 1 byte. 0001 = RX FIFO 触发中断阈值为4 bytes. 0010 = RX FIFO 触发中断阈值为8 bytes. 0011 = RX FIFO 触发中断阈值为14 bytes. 其它 = 保留.
[3]	保留	保留

[2]	TFR	TX 软件复位域 当TX_RST置位，发送 FIFO 内的所有数据和 TX 内部状态机将被清除。 0 = 不影响。 1 = 该位写 1 将复位 TX 内部状态机和指针 注意：该位至少 3 个 UART 时钟周期后才会自动清零
[1]	RFR	RX 软件复位域 当RX_RST被置位，接收 FIFO 内的所有数据和 RX 内部状态机将被清除。 0 = 不影响。 1 = 该位写 1 将复位 RX 内部状态机和指针。 注意：该位至少 3 个 UART 时钟周期后才会自动清零。
[0]	保留	保留

UART 线控制寄存器 (UA_LCR)

寄存器	偏移地址	R/W	描述	复位值
UA_LCR	UARTx_BA+0x0C	R/W	UART 线控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留	BCB	SPE	EPE	PBE	NSB	WLS	

位	描述	
[31:7]	保留	保留。
[6]	BCB	Break 控制位 当该位被置逻辑 1，串行数据输出 (TX) 将强制到 Spacing 状态 (逻辑 0)。该位仅作用于 TX，对传输逻辑不起作用。 0 = Break 控制禁止 1 = Break 控制使能
[5]	SPE	Stick 奇偶校验使能 0 = 禁用 Stick 奇偶校验 1 = 如果 PBE (UA_LCR[3]) 和 EBE (UA_LCR[4]) 都为逻辑 1，奇偶校验位发送和检验值为逻辑 0。如果 PBE (UA_LCR[3]) 是 1，EBE (UA_LCR[4]) 是 0，则奇偶校验位发送和检验值为 1。
[4]	EPE	偶校验使能 0 = 逻辑 1 的奇数数目在每个字节中被发送和检验 1 = 逻辑 1 的偶数数目在每个字中被发送和检验 注意： 该位只在 PBE (UA_LCR[3]) 置位时有效。
[3]	PBE	校验位使能控制 0 = 无校验位 1 = 每一个发送字符中都产生校验位，对每一个传进来的数据进行奇偶校验位检测
[2]	NSB	“停止位”个数 0 = 当发送数据时，产生 1 “STOP bit” 1 = 当发送数据，选择 5-位 字长度时，产生 1.5 “STOP bit”，当选择 6-、7- 和 8-位字长度时，产生 2 “STOP bit”
[1:0]	WLS	字长度选择 00 = 字长是 5-bit.

		01 =字长是6-bit. 10 =字长是7-bit. 11 =字长是8-bit.
--	--	---

UART Modem 控制寄存器 (UA_MCR)

寄存器	偏移地址	R/W	描述	复位值
UA_MCR	UARTx_BA+0x10	R/W	UART Modem 控制寄存器	0x0000_0200

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

位	描述	
[31:14]	保留	保留
[13]	RTS_ST	RTS 脚状态(只读) 该位的值对应于RTS脚的输出电平 0 = RTS脚为低电平逻辑状态 1 = RTS脚为高电平逻辑状态.
[12:10]	保留	保留
[9]	LEV_RTS	RTS 脚的有效电平 该位定义了RTS输出脚的有效电平 0 = RTS 脚输出为高电平有效. 1 = RTS 脚输出为低电平有效. 注意1: UART功能模式请参考图图 6.10-5 和 图 6.10-6 注意2: RS-485 功能模式请参考图图 6.10-10 and 图6.10-11
[8:2]	保留	保留
[1]	RTS	RTS (请求发送) 信号控制 该位直接控制内部RTS脚信号是否有效, 然后使用LEV_RTS位的配置驱动RTS脚输出 0 = RTS 信号有效. 1 = RTS 信号无效. 注意1: UART 功能模式下, 当RTS自动流控被使能后, RTS信号控制位无效 注意 2: RS-485 模式下, 当 RS-485 自动方向模式 (AUD) 被使能后, RTS 信号控制位无效
[0]	保留	保留

UART Modem 状态寄存器 (UA MSR)

寄存器	偏移地址	R/W	描述	复位值
UA_MSR	UARTx_BA+0x14	R/W	UART Modem 状态寄存器	0x0000_0110

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

位	描述	
[31:9]	保留	保留
[8]	LEV_CTS	CTS 脚有效电平 该位定义了CTS输入脚的有效电平 0 = CTS输入脚高电平有效 1 = CTS输入脚低电平有效 注意: 请参考 图 6.10-4
[7:5]	保留	保留
[4]	CTS_ST	CTS 脚状态(只读) 该位对应于CTS脚输入逻辑状态 0 = CTS脚输入状态为低电平 1 = CTS脚输入状态为高电平 注意: 当UART控制器外围时钟被使能, 且CTS功能被使能, 该位才有效。
[3:1]	保留	保留
[0]	DCTS_F	检测到 CTS 引脚电平改变标志 如果CTS输入脚上有电平变化, 该位将被置位, 如果MODEM_IEN (UA_IER [3])位被置位, 将会产生Modem中断。 0 = CTS 输入脚没有电平变化。 1 = CTS输入脚有电平变化。 注意: 该位写“1”清零。

UART FIFO 状态寄存器 (UA_FSR)

寄存器	偏移地址	R/W	描述	复位值
UA_FSR	UARTx_BA+0x18	R/W	UART FIFO 状态寄存器	0x1040_4000

31	30	29	28	27	26	25	24
保留			TE_FLAG	保留			TX_OVER_IF
23	22	21	20	19	18	17	16
TX_FULL	TX_EMPTY	TX_POINTER					
15	14	13	12	11	10	9	8
RX_FULL	RX_EMPTY	RX_POINTER					
7	6	5	4	3	2	1	0
保留	BIF	FEF	PEF	RS485_ADD_DETF	保留		RX_OVER_IF

位	描述	
[31:29]	保留	保留.
[28]	TE_FLAG	发送空标志(只读) 当TX FIFO (UA_THR)为空, 并且最后一个字节的STOP位也已经被发送完毕, 那么TE_FLAG被硬件置1 0 = TX FIFO 不为空. 1 = TX FIFO 为空. 注意: 当 TX FIFO 不为空或最后一个字节没有被全部发送完毕, 此位被自动清零
[27:25]	保留	保留
[24]	TX_OVER_IF	TX溢出错误中断标志 如果TX FIFO (UA_THR)满, 此时如果再向UA_THR写入数据, 将会导致此位被置1. 0 = TX FIFO 未溢出 1 = TX FIFO 已溢出. 注意: 此位可以写 1 清零。
[23]	TX_FULL	发送FIFO满(只读) 此位用于指示TX FIFO是否已满 0 = TX FIFO 未满 1 = TX FIFO 已满. 注意: 当TX FIFO 缓存中的字节数等于16时,此位将被置1. 否则被硬件清零
[22]	TX_EMPTY	发送FIFO 空(只读) 此位指示TX FIFO是否为空 0 = TX FIFO不为空 1 = TX FIFO为空 注意: 当TX FIFO中的最后一个字节被发送到发送移位寄存器, 硬件将把此位置1.当有数据写入THR (TX FIFO不为空)时, 此位被清零

[21:16]	TX_POINTER	TX FIFO 指针(只读) 此域指示TX FIFO缓冲区指针位置。当CPU写一个字节到UA_THR寄存器时，TX_POINTER将累加1.当TX FIFO发送一个字节到发送移位寄存器中，TX_POINTER指针将减1. TX_POINTER显示最大值是15。当TX FIFO缓冲区所填充数据数量达到16时，TX_FULL将被置1，TX_POINTER被清零。如果TX FIFO中发送一个字节到发送移位寄存器，TX_FULL位将被清零，TX_POINTER显示15
[15]	RX_FULL	接收 FIFO 满(只读) 该位指示RX FIFO 是否已满 0 = RX FIFO 未满 1 = RX FIFO 已满 注意：当RX FIFO缓冲区中的数据数量等于16后，此位将被置1，否则被硬件清零。
[14]	RX_EMPTY	接收 FIFO空(只读) 此位指示RX FIFO是否为空 0 = RX FIFO不为空. 1 = RX FIFO为空. 注意：当RX FIFO中最后一个字节被CPU读取后，硬件将对此位置1，UART接收到新数据后此位将被清零。
[13:8]	RX_POINTER	RX FIFO指针(只读) 此位指示RX FIFO缓冲区指针。当UART从外部设备接收到一个字节，RX_POINTER将累加1.当RX FIFO的数据被CPU读取一个字节，RX_POINTER将递减1. RX_POINTER显示的最大值是15.当RX FIFO的数据达到16时，RX_FULL位将被置1，RX_POINTER显示0，RX FIFO 当中的数据被CPU读取一个后，RX_FULL 将被清零，RX_POINTER显示15
[7]	保留	保留
[6]	BIF	Break中断标志位(只读) 每当接收到数据输入 (RX) 维持在“spacing state” (logic 0) 的时间长于一个全字的传输时间（即“start bit” + data bits + parity + stop bits 的总时间），该位置 1。 0 =没有Break中断产生. 1 =有Break中断产生. 注意1：此位只读，但是可以写1清零 注意2：此位只读，但是可以向RFR (UA_FCR [1])写1清零
[5]	FEF	帧错误标志(只读) 每当接收到的字符没有合法的“停止位”（即跟着最后的数据位后或奇偶校验位检测为0）该位置 1。 0 =没有帧错误产生 1 =有帧错误产生. 注意1：此位只读，但是可以写1清零 注意2：此位只读，但是可以向RFR (UA_FCR [1])写1清零
[4]	PEF	校验错误标志 (只读) 每当接收到的字符没有合法的奇偶校验位，该位置 1。 0 = 没有奇偶校验错误产生 1 = 有奇偶校验错误产生 注意1：此位只读，但是可以写1清零

		注意 2: 此位只读，但是可以向 RFR (UA_FCR [1])写 1 清零
[3]	RS485_ADD_DETF	RS-485 地址数据检测标志(只读) 当RS485_ADD_EN (UA_ALT_CSR[15])位置1使能地址检测模式，且接收检测到带地址位 (bit 9 = 1)的数据时，该位置1. 0 = 接收到的数据没有地址位标记 (bit 9 = '1'). 1 = 接收到的数据有地址位标记 (bit 9 = '1'). 注意1: 此位用于RS-485功能模式 注意 2: 该位写 1 清零。
[2:1]	保留	保留
[0]	RX_OVER_IF	RX 溢出错误标志(只读) 当RX FIFO溢出时该位被置1 如果接收到的数据数量大于 RX_FIFO (UA_RBR)的大小16 字节，该位置位 0 = RX FIFO未溢出. 1 = RX FIFO溢出 注意: 该位写 1 清零

UART 中断状态控制寄存器 (UA_ISR)

寄存器	偏移地址	R/W	描述	复位值
UA_ISR	UARTx_BA+0x1C	R/W	UART 中断状态寄存器	0x0000_0002

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
LIN_RX_BREAK_INT	保留	BUF_ERR_INT	TOUT_INT	MODEM_INT	RLS_INT	THRE_INT	RDA_INT
7	6	5	4	3	2	1	0
LIN_RX_BREAK_IF	保留	BUF_ERR_IF	TOUT_IF	MODEM_IF	RLS_IF	THRE_IF	RDA_IF

位	描述	
[31:14]	保留	保留
[15]	LIN_RX_BREAK_INT	LIN总线RX Break域中断检测标志(只读) 如果LIN_RX_BRK_IEN 和 LIN_RX_BREAK_IF都被置1, 该位置 1 0 =没有LIN总线RX Break中断产生 1 =有LIN总线RX Break中断产生
[14]	保留	保留
[13]	BUF_ERR_INT	Buffer错误中断标志(只读) 如果BUF_ERR_IEN 和 BUF_ERR_IF都被置1, 该位置 1 0 =没有buffer错误中断产生. 1 =有buffer错误中断产生
[12]	TOUT_INT	超时溢出中断标志(只读) 如果RTO_IEN 和 TOUT_IF都被置1, 该位置 1 0 =没有定时溢出中断产生 1 =有定时溢出中断产生.
[11]	MODEM_INT	MODEM状态中断标志(只读) 如果MODEM_IEN 和 MODEM_IF都被置1, 该位置 1 0 =没有Modem 中断产生. 1 =有Modem 中断产生..
[10]	RLS_INT	接收线状态中断标志(只读) 如果RLS_IEN 和 RLS_IF都被置1, 该位置 1 0 =没有RLS中断产生

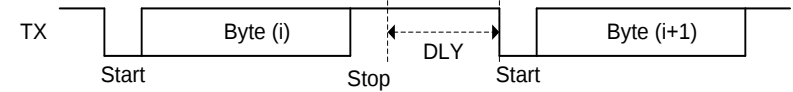
		1 =有RLS中断产生.
[9]	THRE_INT	发送保持寄存器空中断标志(只读) 如果THRE_IEN 和 THRE_IF都被置1, 该位置 1 0 =没有THRE中断产生 1 =有THRE中断产生
[8]	RDA_INT	接收可用数据中断标志(只读) 如果RDA_IEN 和 RDA_IF都被置1, 该位置 1 0 =没有RDA中断产生. 1 =有RDA中断产生.
[7]	LIN_RX_BREAK_IF	LIN总线RX Break域中断检测标志 当RX接收到LIN总线Break域时, 该位被置1, 如果LIN_RX_BRK_IEN 有使能, 将产生LIN总线Break中断。 0 =没有LIN总线RX Break中断产生 1 =有LIN总线RX Break中断产生 注意: 该位通过写 1 清零
[6]	保留	保留
[5]	BUF_ERR_IF	Buffer 错误中断标志 (只读) 当TX FIFO或RX FIFO溢出 (TX_OVER_IF 或 RX_OVER_IF)时, 该位置1. 当BUF_ERR_IF被置位, 传输出错.如果BUF_ERR_IEN (UA_IER [5])被使能, buffer 错误中断将产生 0 = 没有buffer 错误中断标志产生 1 = 有buffer 错误中断标志产生 注意: 此位只读。当 TX_OVER_IF 和 RX_OVER_IF 都被清零, 该位被清零.
[4]	TOUT_IF	超时溢出中断标志 (只读) 当 RX FIFO 非空且 RX FIFO无活动发生, 超时溢出计数器等于TOIC 时, 该位置位。如果TOUT_IEN (UA_IER [4])使能, 超时溢出中断将产生。 0 =没有超时溢出中断标志产生. 1 =有超时溢出中断标志产生 注: 该位只读, 用户可以读 UA_RBR (RX处于活动状态) 清除该位
[3]	MODEM_IF	MODEM 中断标志(只读) 当CTS管脚有状态改变 (DCTSIF= 1)时, 该位置位。如果 (UA_IER [MODEM_IEN])被使能, Modem 中断将产生。 0 =没有Modem 中断标志产生. 1 =有Modem 中断标志产生. 注: 该位只读, 当向DCTSIF写1清除后, 该位清除
[2]	RLS_IF	接收线中断标志(只读) 当 RX 接收数据有校验错误,帧错误或 break 错误 (BIF,FEF和PEF 3位中至少有1位被置), 该位被置位。如果RLS_IEN (UA_IER [2])使能, 将产生RLS 中断。 0 =没有RLS中断标志产生 1 =有RLS中断标志产生. 注意1: 在 RS-485功能模式, 该域被置位包括“接收检测和接收到的地址字节字符 (bit9 = '1') 位”, 同时, RS485_ADD_DET (UA_FSR[3])位也被置1。 注意2: 该位只读, 当BIF, FEF 和 PEF三位都被清除后, 该位重置为 0
[1]	THRE_IF	发送保持寄存器空中断标志(只读)

		当TX FIFO的最后数据传输到发送移位寄存器时，该位置位。如果THRE_IEN (UA_IER[1])被使能，将产生THRE 中断。 0 =没有THRE中断标志产生 1 =有THRE中断标志产生。 注意: 该位只读，当写数据到 THR (TX FIFO 非空) 时，该位将被清除
[0]	RDA_IF	接收可用数据中断标志(只读) 当 RX FIFO的数据数量等于 RFITL时，RDA_IF将被置位。如果RDA_IEN (UA_IER [0]) 被使能，将产生RDA 中断。 0 =没有RDA中断标志产生。 1 =有RDA中断标志产生。 注意:该位只读，当 RX FIFO 的未读数据低于阈值(RFITL)时，该位将被清除

UART 超时溢出寄存器 (UA_TOR)

寄存器	偏移地址	R/W	描述	复位值
UA_TOR	UARTx_BA+0x20	R/W	UART 超时溢出寄存器	0x0000_0000

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

位	描述	
[31:16]	保留	保留
[15:8]	DLY	TX 延迟时间值 该域用于配置上一次停止位和下一次开始位之间的传输延迟时间 
[7:0]	TOIC	超时溢出中断比较器 当 RX FIFO 接收到一个新的数据字，超时溢出计数器将重置并开始计数 (计数时钟 = 波特率)。一旦超时计数器的内容(TOUT_CNT)等于超时中断比较器(TOIC)时，如果此时 RTO_IEN (UA_IER [4])为使能，则接收超时中断(INT_TOUT)产生。新来的数据字或 RX FIFO 为空时清除 TOUT_INT。为了避免接收超时中断，一接收到一个字符就立即产生，TOIC 的值必须设置在 40 到 255 之间。例如，如果 TOIC 设置为 40，当 UART 传输设置为 1 停止位和无奇偶校验位时，超时中断将在 4 个字符没有收到之后产生。

UART 波特率分频寄存器 (UA_BAUD)

寄存器	偏移地址	R/W	描述	复位值
UA_BAUD	UARTx_BA+0x24	R/W	UART 波特率分频寄存器	0x0F00_0000

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

位	描述	
[31:30]	保留	保留
[29]	DIV_X_EN	分频 X 使能控制位 BRD = 波特率分频值, 波特率的公式为 波特率= Clock / [M * (BRD + 2)]; M的默认值是16. 0 = 分频值X禁用 (M = 16). 1 = 分频值X使能 (M = X+1, 但是DIVIDER_X [27:24] 必须>= 8). 注意1: 参考章节6.10.5.1查看详细信息 注意2: IrDA模式下, 该位须禁止.
[28]	DIV_X_ONE	分频X 等于1 0 = 分频值M = X (M = X+1, 但是DIVIDER_X[27:24] 必须>= 8). 1 = 分频值M = 1 (M = 1, 但是BRD [15:0] 必须>= 8). 注意: 参考章节6.10.5.1查看详细信息
[27:24]	DIVIDER_X	分频X 波特率除频 M = X+1.
[23:16]	保留	保留
[15:0]	BRD	波特率分频器 该域用于波特率分频

UART IrDA 控制寄存器 (UART_IRCR)

寄存器	偏移地址	R/W	描述	复位值
UA_IRCR	UARTx_BA+0x28	R/W	UART IrDA 控制寄存器	0x0000_0040

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

位	描述	
[31:7]	保留	保留
[6]	INV_RX	IrDA 反向接收输入信号控制 0 = 不反向接收输入信号. 1 = 反向接收输入信号.
[5]	INV_TX	IrDA 反向发送信号控制 0 = 信号不反向发送. 1 = 信号反向发送
[4:2]	保留	保留
[1]	TX_SELECT	IrDA 接收使能控制 0 = IrDA发送禁止接收使能. 1 = IrDA发送使能接收禁止
[0]	保留	保留

UART 复用控制/状态寄存器 (UA_ALT_CSR)

寄存器	偏移地址	R/W	描述	复位值
UA_ALT_CSR	UARTx_BA+0x2C	R/W	UART 复用控制/状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
ADDR_MATCH							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
RS485_ADD_EN	保留				RS485_AUD	RS485_AAD	RS485_NMM
7	6	5	4	3	2	1	0
LIN_TX_EN	LIN_RX_EN	保留		UA_LIN_BKFL			

位	描述	
[31:24]	ADDR_MATCH	地址匹配值寄存器 该位包括 RS-485 地址匹配值。 注意： 该域用于 RS-485 自动地址检测模式
[23:16]	保留	保留
[15]	RS485_ADD_EN	RS-485 地址检测使能位 该位用于使能 RS-485 地址检测使能模式。 0 = 地址检测模式禁止。 1 = 地址检测模式使能。 注意： 该位用于 RS-485 任意操作模式。
[14:11]	保留	保留
[10]	RS485_AUD	RS-485 自动方向模式 (AUD)控制位 0 = RS-485自动方向操作模式(AUD)禁止。 1 = RS-485自动方向操作模式(AUD)使能 注意： 在RS-485_AAD 或 RS-485_NMM 操作模式有效
[9]	RS485_AAD	RS-485 自动地址检测操作模式 (AAD)控制位 0 = RS-485自动地址方向操作模式 (AAD)禁止。 1 = RS-485自动地址方向操作模式 (AAD)使能。 注意： 在 RS-485_NMM 操作模式下无效
[8]	RS485_NMM	RS-485 普通多点操作模式 (NMM) 控制位 0 = RS-485 普通多点操作模式 (NMM)禁止。 1 = RS-485 普通多点操作模式 (NMM)使能 注： 在 RS-485_AAD 操作模式下无效。

[7]	LIN_TX_EN	LIN TX Break 模式控制使能 0 = LIN TX Break 模式禁止 1 = LIN TX Break 模式使能. 注意: 当 TX break 域传输操作完成后, 该位将被自动清除.
[6]	LIN_RX_EN	LIN RX 控制使能 0 =LIN RX 模式禁止. 1 = LIN RX 模式使能
[5:4]	保留	保留
[3:0]	UA_LIN_BKFL	UART LIN Break 域长度 该域表示一个 4-位 LIN TX break 域计数 注意: 该break 域长度为 UA_LIN_BKFL + 2

UART 功能选择寄存器 (UA_FUN_SEL)

寄存器	偏移地址	R/W	描述	复位值
UA_FUN_SEL	UARTx_BA+0x30	R/W	UART 功能选择寄存器	0x0000_0000

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

位	描述	
[31:2]	保留	保留
[1:0]	FUN_SEL	功能选择使能位 串口控制器的功能模式选择 00 = UART功能使能. 01 = LIN 功能使能 10 = IrDA 功能使能 11 = RS-485 功能使能.

6.11 I²C 总线控制器(I²C)

6.11.1 概述

I²C为2线，双向串行总线，为设备之间的数据通讯提供了简单有效的方法。I²C标准是多主机总线，包括冲突检测和仲裁，以防止在两个或多个主机试图同时控制总线时发生数据冲突。M058S系列有2组I²C，提供睡眠唤醒功能。

6.11.2 特性

I²C总线通过两根线(SDA和SCL)在连接到总线上的设备间传输数据，总线的主要特征包括：

- 最多支持两组I²C 接口
- 支持主/从模式
- 主从间可双向数据传输
- 多主机总线(无中心主机)
- 多主机同时发送数据支持仲裁功能，而不会破坏总线上的数据
- 总线采用串行同步时钟，可实现设备之间实现各种速率传输
- 串行时钟同步使用了一种握手机制来挂起和恢复数据传输
- 内建14位溢出定时器，当I²C总线中止且定时器溢出，产生I²C中断
- 可配置不同时钟以适用于可变速率控制
- 支持7位从地址模式
- I²C 总线控制器支持多地址识别（4组从机地址带mask 选项）
- 支持唤醒功能

6.11.3 基本配置

I²C0的基本配置如下：

- I2C0管脚从寄存器P3_MFP [13:12]配置
- 通过设置I2C0_EN(APBCLK [8])使能I2C0时钟
- 通过设置I2C0_RST (IPRSTC2 [8]) 复位I2C0控制器

I²C1的基本配置如下：

- I2C1管脚从寄存器P2_MFP [5:4] 或 P4_MFP[5:4]配置
- 通过设置I2C1_EN (APBCLK [9])使能I2C1时钟
- 通过设置I2C1_RST(IPRSTC2 [9])复位I2C1控制器

6.11.4 功能描述

I2C总线中，数据通过I2C和I2C两根线在主从机间逐字节同步传送。每个字节数据长度是8位。一个SCL 时钟脉冲传输一个数据位，数据由最高位MSB 开始传输，每个传输字节后跟随一个应答位，每个位在SCL 为高时被采样；因此，SDA 线只有在SCL 为低时才可以改变，在SCL 为高时SDA 保持稳定。当SCL 为高时，SDA 线上的跳变视为一个命令(START或STOP)。更多关于I2C总线时序的细节请参考下图。

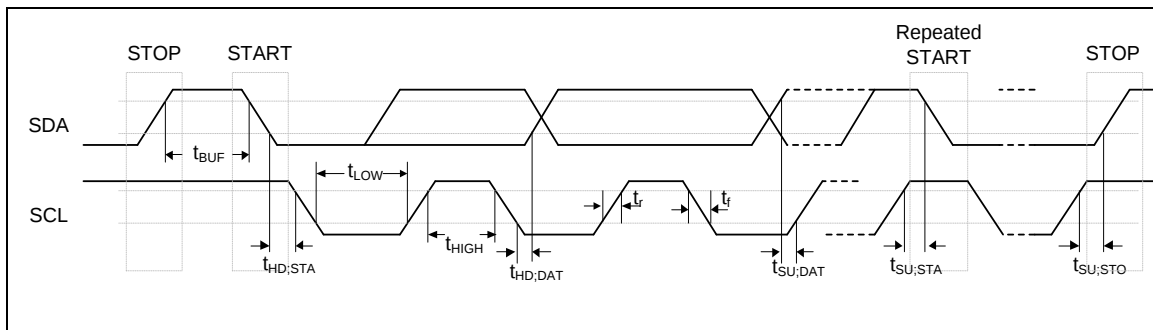


图 6.11-1 I²C 总线时序

片上I2C外设提供了一个符合I2C总线规范的串行接口。I2C端口自动处理字节传输。将I2CON寄存器中的ENS1位置为'1'，即可使能该端口。I2C 硬件接口通过SDA 与SCL两个管脚连到I2C总线。当I/O 管脚作为I2C 端口使用时，用户必须事先设定I/O 管脚功能为I2C功能。

注意： SDA 和SCL两个管脚需要上拉电阻，因为这个两个管脚为开漏脚。

6.11.5 I²C 协议

标准I²C 协议如下图，通常标准通讯有以下4部分：

- 1) 起始信号(START)或者重复起始信号(Repeated START)
- 2) 从机地址传输和R/W 位传输
- 3) 数据传输
- 4) 停止信号(STOP)

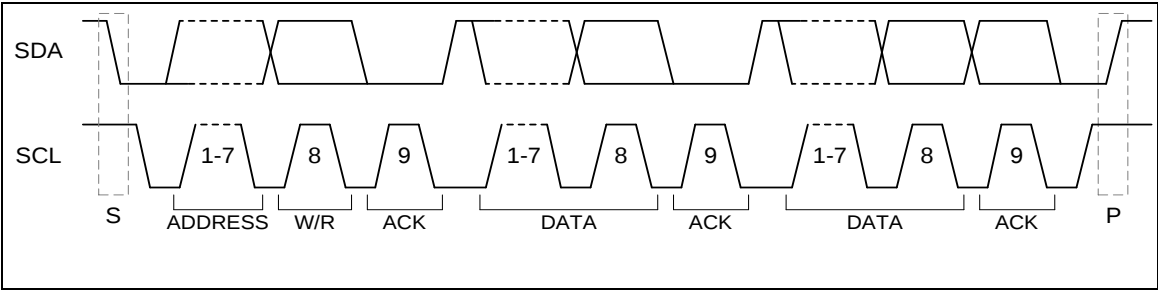


图 6.11-2 I²C 协议

6.11.5.1 起始或重复起始信号

当总线处于释放/空闲状态下，说明没有主机设备占用总线（SDA 与 SCL线均为高电平），主机可以通过发送起始(START)信号来发起传输过程。起始信号，通常表示为S位，当SCL 线为高时，SDA 线上信号由高至低变化，就被定义为起始信号。起始信号表示一个新的数据传输的开始。

重复起始信号前不一定要有停止信号，重复起始信号通常用“Sr”位表示。使用这个方法，主机跟其它从机或同一个从机变更不同的传输方向（例如，写设备变为读设备）而不必释放总线。

6.11.5.2 停止(STOP) 信号

主机可以通过产生一个停止信号来终止数据通讯。停止信号，通常用“P”来表示，当SCL 线为高时，SDA线上信号由低至高变化，就被定义为停止信号。

下图为起始、重复起始和停止信号的波形。

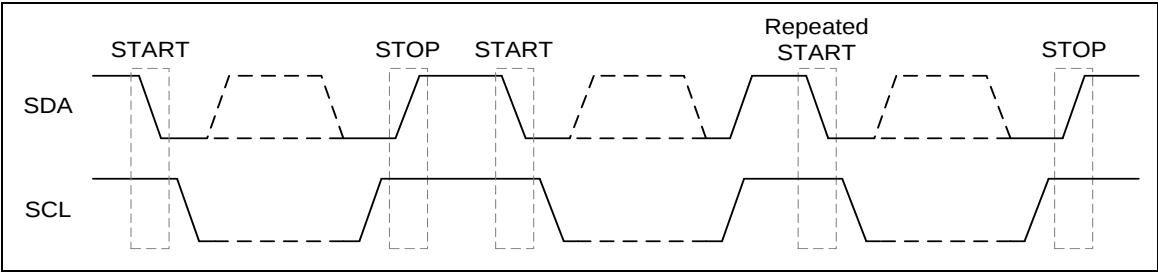


图 6.11-3 起始（START）和停止（STOP）信号条件

6.11.5.3 从机地址传输

起始信号后，主机立即传输的第一个字节就是从机地址(SLA)。这是一个7位呼叫地址跟随一个读/写(R/W) 位。R/W 位标志从机信号的数据传输方向。总线上不能有两个相同的从机地址，只有被主机寻址的从机设备，才会在第9个SCL 时钟周期时通过将SDA 拉低作为应答

6.11.5.4 数据传输

当从机设备接收到带R/W位的正确的地址后，就可以根据R/W 位所决定的方向进行数据传输。每个字节传输完后，紧接着在第9个SCL时钟周期会有一个应答信号位。如果从机上产生无应答信号(NACK)，主机可以产生(STOP)停止信号来中止数据传输，或者产生重复起始信号来开始新一轮的数据传输。

当主机作为接收设备时，发生无应答信号(NACK)，则从机释放SDA 线，让主机产生停止信号或重复起始信号。

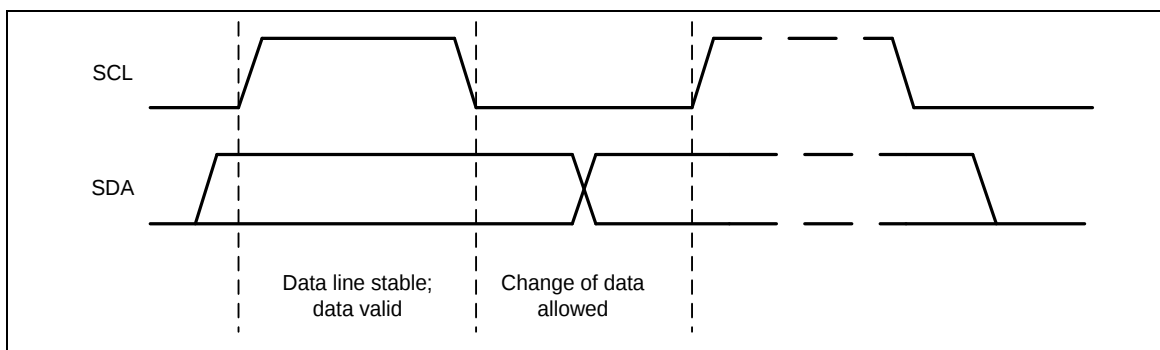


图 6.11-4 I²C 总线上的位传输

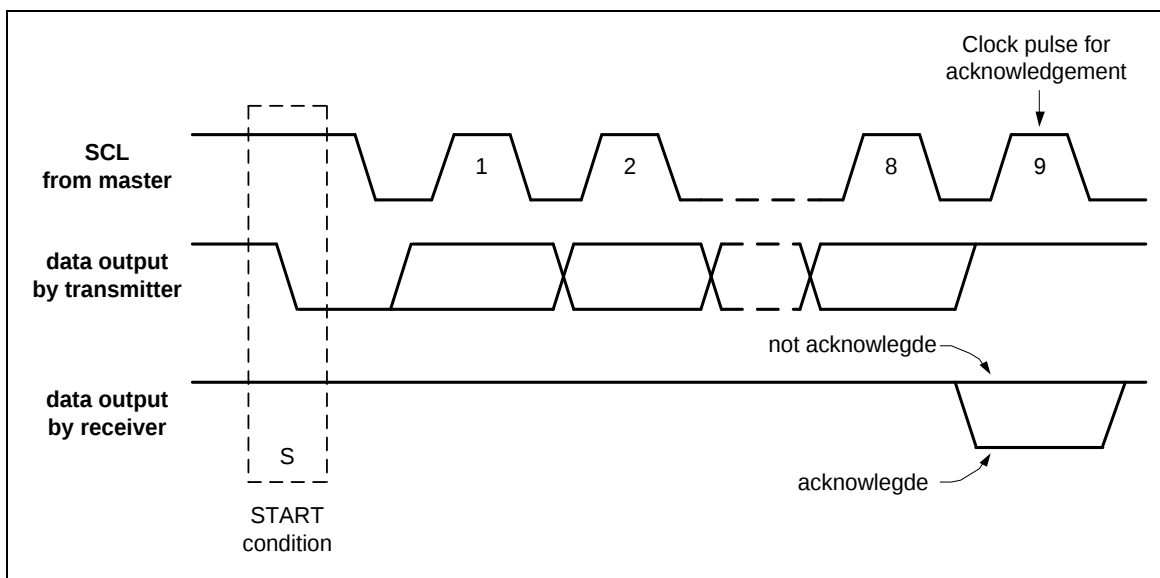


图 6.11-5 I²C 总线上的应答信号

6.11.5.5 I²C总线上的数据传输

下图表示主机向从机传输数据。主机发出一个7-bit地址和1-bit写指示,表示主机想要传送数据给从机。从机回复应答给主机之后，主机继续传输数据

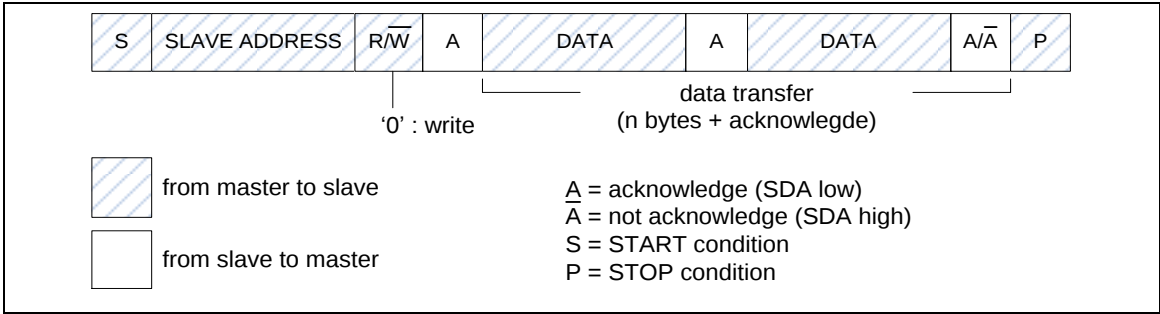


图 6.11-6 主机向从机传输数据

下图表示主机向从机读取数据。主机发7位地址寻址和1-bit读指示，表示主机要向从机读取数据，从机返回应答给主机后就开始传输数据。

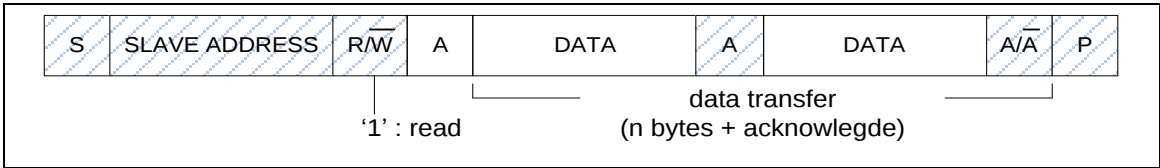


图 6.11-7 主机向从机读取数据

6.11.6 I²C 协议寄存器

通过下列15个特殊功能寄存器来控制I²C端口：I2CON（控制寄存器）I2CSTATUS（状态寄存器），I2CDAT（数据寄存器），I2CADDRn（地址寄存器，n=0~3），I2CADMn（地址掩码寄存器，n=0~3），I2CLK（时钟速率寄存器），I2CTOC（超时寄存器），I2CWKUPCON（唤醒控制寄存器），I2CWKUPSTS（唤醒状态寄存器）。

地址寄存器(I2CADDR)

I²C端口内建4个从机地址寄存器I2CADDRn (n=0~3)。当I2C作为主机时，这四个寄存器的内容没意义。在从机模式下，位字段I2CADDRn[7:1] 必须装载芯片自己的从机地址。当I2CADDRn 地址与接收到的从机地址符合时I²C硬件会作出反应。

I²C端口支持“广播呼叫”功能。当GC位(I2CADDRn [0]) 被置1，I²C端口硬件将应答广播呼叫的地址(00H)。清GC 位可禁用“广播呼叫”功能。

当GC 位被置1且I²C处于从机模式时，主机发出广播呼叫地址到I²C总线后，从机可以通过地址00H接收广播呼叫地址，然后它将跟随GC 模式的状态

从机地址掩码寄存器(I2CADM)

I²C总线控制器带有四个地址掩码寄存器I2CADMn (n=0~3) 支持多地址识别。当地址掩码寄存器被置1，意味着收到的相应地址位是“不考虑”。如果置0，则收到相应地址位应跟地址寄存器完全一样

数据寄存器(I2CDAT)

该寄存器包含一个准备发送或刚接收到的一个字节的串行数据。当不在字节移位处理过程时，CPU 可以直接读写访问 I2CDAT[7:0]。当 I2C 处于一个确定的状态并且串行中断标志(SI) 被置1，I2CDAT[7:0]中的数据保持稳定。当数据被移出，总线上的数据同时被移入。I2CDAT [7:0]的内容总是总线上传递的最后一个字节的数据。

应答位由 I²C 硬件控制，不能被 CPU 操作。串行数据在 SCL 线上，串行时钟脉冲上升沿时移入 I2CDAT [7:0]。当一个字节被移入 I2CDAT [7:0]时，I2CDAT [7:0] 中的串行数据是可用的，应答位 (ACK 或 NACK) 在第9个时钟脉冲由控制逻辑返回。为了在发送数据时监控总线的状态，当发送 I2CDAT [7:0]到总线时，总线数据将同时被移入 I2CDAT [7:0]。在发送数据过程中，串行数据在 SCL 时钟脉冲的下降沿从 I2CDAT [7:0] 移出，在 SCL 时钟脉冲的上升沿数据移入 I2CDAT [7:0]

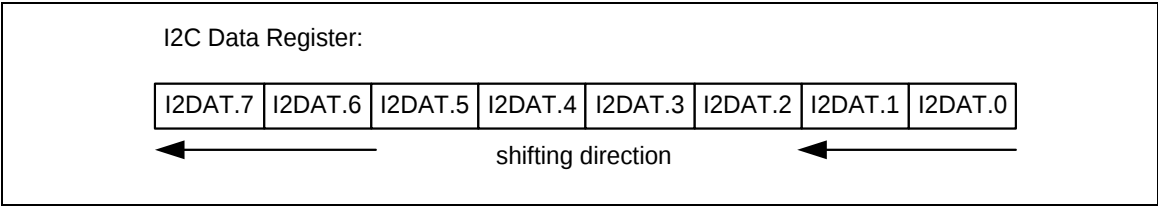


图 6.11-8 I²C 数据移位方向

控制寄存器(I2CON)

CPU 可以直接读写 I2CON [7:0]，当 I²C 端口通过设置 ENS1(I2CON[6]) 为高使能时，内部状态由 I2CON 和 I²C 硬件逻辑控制。

有两个位会受硬件影响：当 I²C 硬件产生中断时 SI 被置位，当总线上产生停止信号时，STO 位被清零。当 ENS1=0 时 STO 也会被清零。

一旦有新的状态码产生并存储在 I2CSTATUS，I²C 中断标志位 SI(I2CON[3]) 将被自动置位。若 EI (I2CON [7]) 位被使能，I²C 中断将产生。I2CSTATUS[7:0] 用来存储内部状态码，SI 被软件清除前内容一直保持。

状态寄存器(I2CSTATUS)

I2CSTATUS [7:0] 是一个 8-位只读寄存器。I2CSTATUS [7:0] 共有 26 个可能的状态码。当 I2CSTATUS [7:0] 的值为 0xF8 时，没有串行中断请求。所有其它的 I2CSTATUS [7:0] 的值对应 I²C 的某一状态。当进入其中任一状态时，就会产生状态中断请求(SI= 1)。在 SI 被硬件置位一个周期后，有效状态码出现在 I2CSTATUS [7:0] 中，且在 SI 被软件复位后仍存在一个机器周期。

此外，状态 0x00 表示总线错误，这会出现起始或停止条件处在不正确的 I²C 格式帧。总线错误会发生在地址字节、一个数据字节或一个应答位的串行传输。为了把 I²C 从总线错误中恢复，STO 应被置位，SI 应被清除，以进入无地址从机模式。然后 STO 被清除来释放总线，并等待下一个传输。出现总线错误操作期间 I²C 总线不能识别停止条件。

主机模式	从机模式
------	------

状态码	描述	状态码	描述
0x08	起始	0xA0	从机发送重新起始或停止
0x10	主机重新起始	0xA8	从机发送地址ACK
0x18	主机发送地址ACK	0xB0	从机发送仲裁丢失
0x20	主机发送地址NACK	0xB8	从机发送数据ACK
0x28	主机发送数据ACK	0xC0	从机发送数据NACK
0x30	主机发送数据NACK	0xC8	从机发送最后数据ACK
0x38	主机仲裁丢失	0x60	从机接收地址 ACK
0x40	主机地址接收ACK	0x68	从机接收仲裁丢失
0x48	主机地址接收NACK	0x80	从机接收数据 ACK
0x50	主机数据地址接收 ACK	0x88	从机接收数据NACK
0x58	主机数据接收 NACK	0x70	广播模式地址ACK
0x00	总线错误	0x78	广播模式仲裁丢失
		0x90	广播模式数据 ACK
		0x98	广播模式数据 NACK
0xF8	总线释放 注意： 主/从模式的“0xF8”状态，都不会产生中断		

表 6.11-1 I²C 状态码描述

时钟波特率位(I2CLK)

当I²C 在主机模式下，I²C数据的波特率由I2CLK[7:0] 寄存器设定。在从机模式下不需要设置。在从机模式下，I²C 将自动与主机I²C设备时钟频率同步。

I²C 数据波特率的设定：I²C数据波特率= (系统时钟) / (4x (I2CLK [7:0] +1))。如果系统时钟 = 16 MHz, I2CLK [7:0] = 40 (28H), 那么I²C数据波特率= 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec。

超时计数器寄存器(I2CTOC)

MCU 提供一个14 位超时计数器来处理当I2C总线挂起的情况。当超时计数器功能使能后，计数器开始计数直至溢出(TIF=1) 并向CPU 产生I2C中断或者清除ENTI为0来停止计数。当超时计数器使能，SI标志置高会使计数器复位， SI清零后重新开始计数。如果I2C总线挂起，会导致I2CSTATUS 及SI 标志在一段时间内没有被更新，该14-位超时计数器可能会溢出，从而产生I2C中断通知CPU。参考下图14-位超时计数器的框图。用户可以写1清TIF为0。

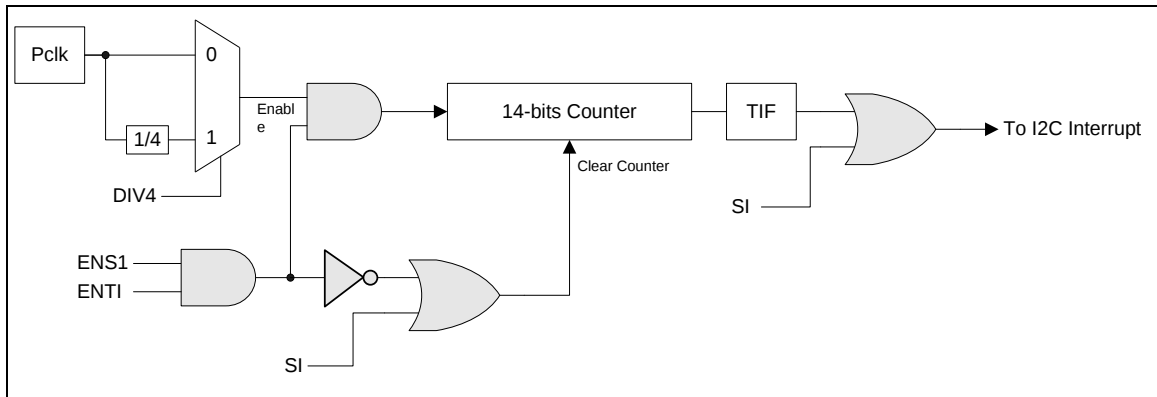


图 6.11-9 I²C 超时计数器框图

唤醒控制寄存器(I2CWKUPCON)

当芯片进入掉电模式，其他的I²C主机可以通过寻址I²C设备唤醒我们的芯片。用户必须在进入掉电模式之前做相关的设置。当I²C设备的四个地址寄存器的其中一个发生地址匹配唤醒芯片时，此时下一个数据将被作废。

唤醒状态寄存器(I2CWKUPSTS)

当系统被其他的I²C主机设备唤醒时，WKUPIF被置位表示该事件发生。用户需要写“1”来清除此位。

6.11.7 工作模式

片上I²C端口支持三种操作模式：主模式，从模式和广播呼叫模式。

应用中，I²C 端口可以作为主机和从机。在从机模式，I²C端口硬件会查找自身从机地址和广播呼叫地址，如果这两个地址的任一个被检测到，并且从机打算从主机接收或向主机发送数据(通过设置AA位)，应答脉冲将会在第9个时钟被发出，此时，如果中断被使能，则在主机和从机设备上都会发生一次中断请求。在主控制器想要成为总线主机时，在进入主机模式之前，硬件需等待到总线空闲使合理的从机动作不会被打断。在主机模式下，如果总线仲裁丢失，I²C立即切换到从机模式，并可以在同一次串行传输过程中检测自身的从机地址

为控制I²C总线的各种模式传输，用户需要按照I2CSTATUS寄存器的当前状态码来设置I2CON, I2CDAT。换句话说就是，每个I²C动作都要查询I2CSTATUS寄存器的当前状态，并设置I2CON, I2CDAT寄存器来让总线动作。最后，通过I2CSTATUS来检查响应状态。

在SI (I2CON[3])标志清除后，I2CON 中的STA, STO和 AA位用来控制I²C硬件的下一个状态。当完成一个新的动作，I2CSTATUS的状态代码将被更新，SI标志将被设置。如果I²C中断控制位EI (I2CON [7])被置位，新状态代码对应的动作或者软件将在中断服务程序中被执行。

下图显示当前I²C状态码是0x08，然后设置I2CDATA=SLA+W和(STA,STO,SI,AA) = (0,0,1,x)来发送地址到I²C总线。如果在总线上的从机地址相匹配且返回ACK，I2CSTATUS状态码更新为0x18。

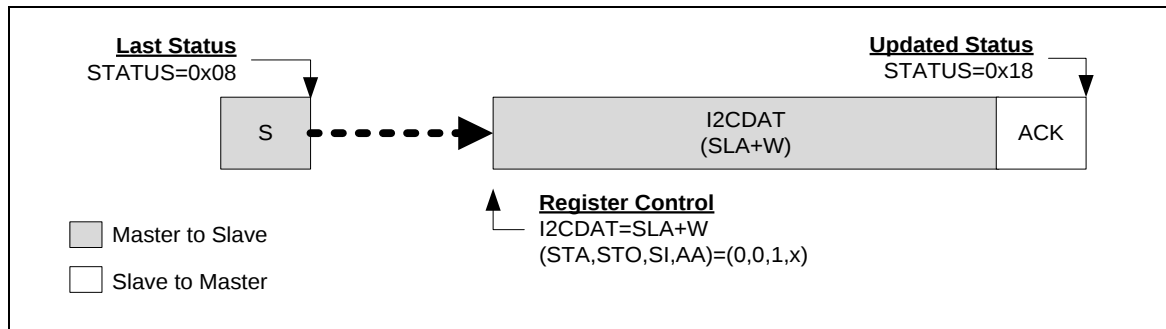


图 6.11-10根据I²C当前状态控制I²C总线

6.11.7.1 主机模式

下图显示了I²C主机模式中所有可能的协议。用户需要遵循恰当的流程来执行I²C协议。

换句话说，用户可以根据(图 6.11-11)发送一个起始信号到总线，I²C总线将设置成主机发送模式，或根据(图 6.11-12图 6.11-12)设置成主机接收模式。起始信号设置成功后新的状态码将是0x08。起始信号后，用户可以发送从机地址，读/写位，数据和重复起始，停止来执行I²C协议。

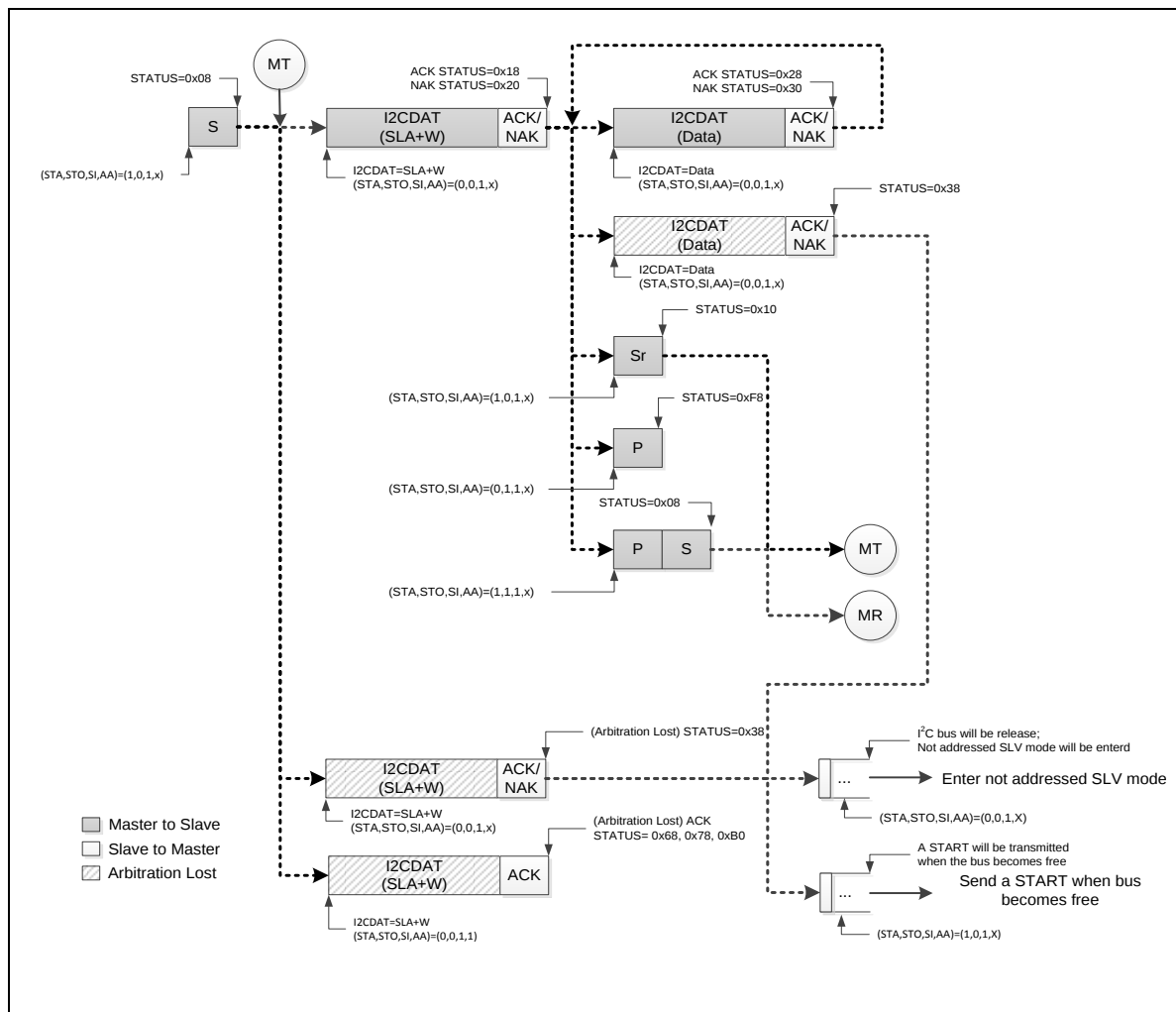


图 6.11-11 主机发送模式控制流程

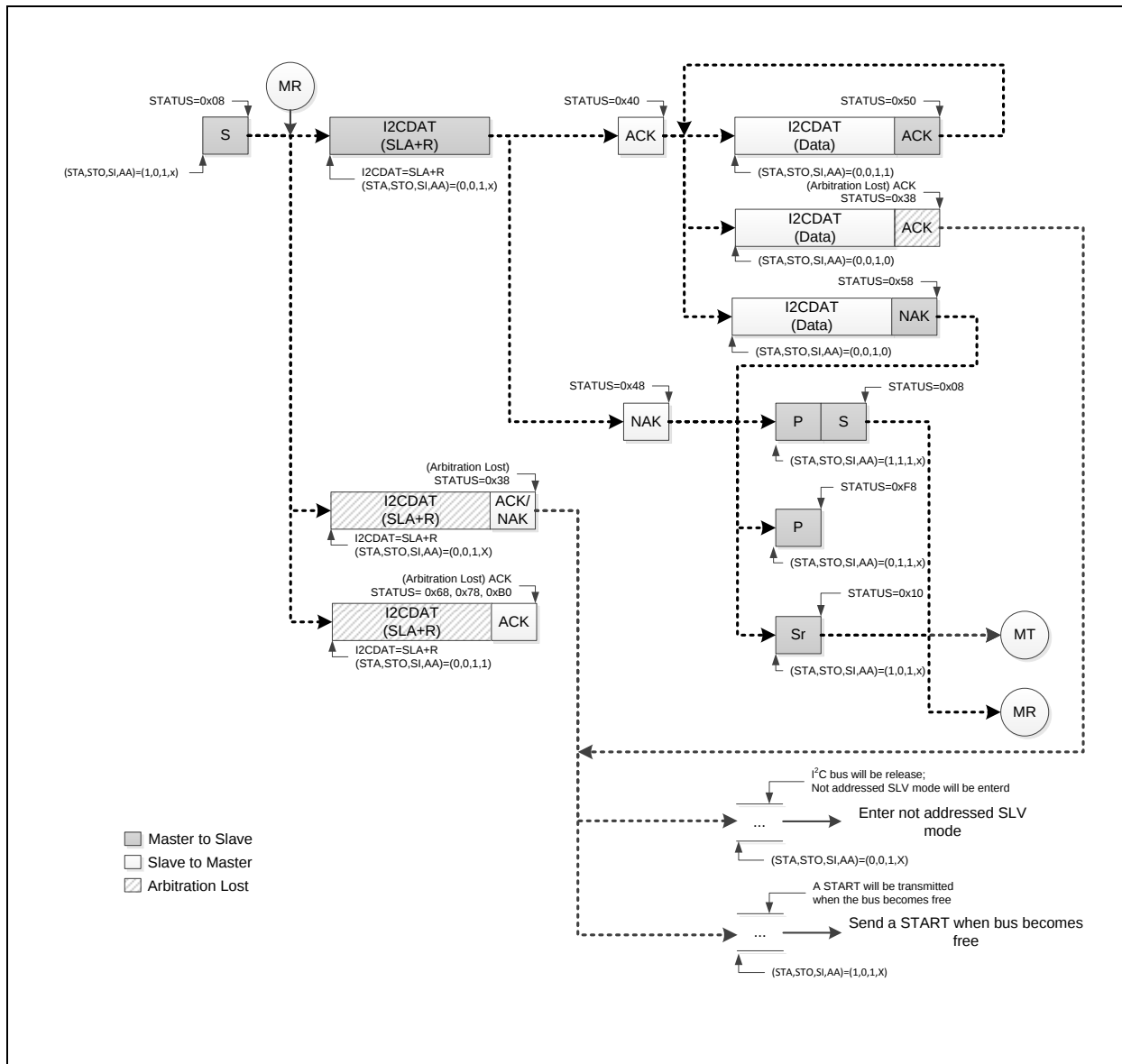


图 6-23 主机接收模式控制流程

如果I²C在主机模式并且仲裁丢失，状态码将置为0x38，在状态0x38时，用户可以设置(STA, STO, SI, AA) = (1, 0, 1, X)在总线空闲时，发送起始信号来重新开始主机操作。另外，用户可以设置(STA, STO, SI, AA) = (0, 0, 1, X)来释放总线，并进入无地址从机模式。

6.11.7.2 从机模式

复位后默认情况下，I²C不会被寻址，并且不会识别I²C总线上的地址。用户可以通过设置从机地址I2CADDRx 和(STA, STO, SI, AA) = (0, 0, 1, 1)来让I²C识别主机发送的地址。图 6.11-12所示为从机模式可能的所有流程。用户需要遵循（图 6.11-12）的流程来实现他们的I²C协议。

如果在主机模式仲裁丢失，I²C端口立即切换到从机模式并且能够在同一串行传输中识别自有的从机地址。如果在仲裁丢失后，识别到地址是SLA+W（主机想写数据到从机），状态码是0X68。如果在仲裁丢失后识别到地址是SLA+R（主机想向从机读数据），状态码是0XB0。

注意：I²C通讯期间，当从机模式的SI标志被写‘1’， SCL将被释放。

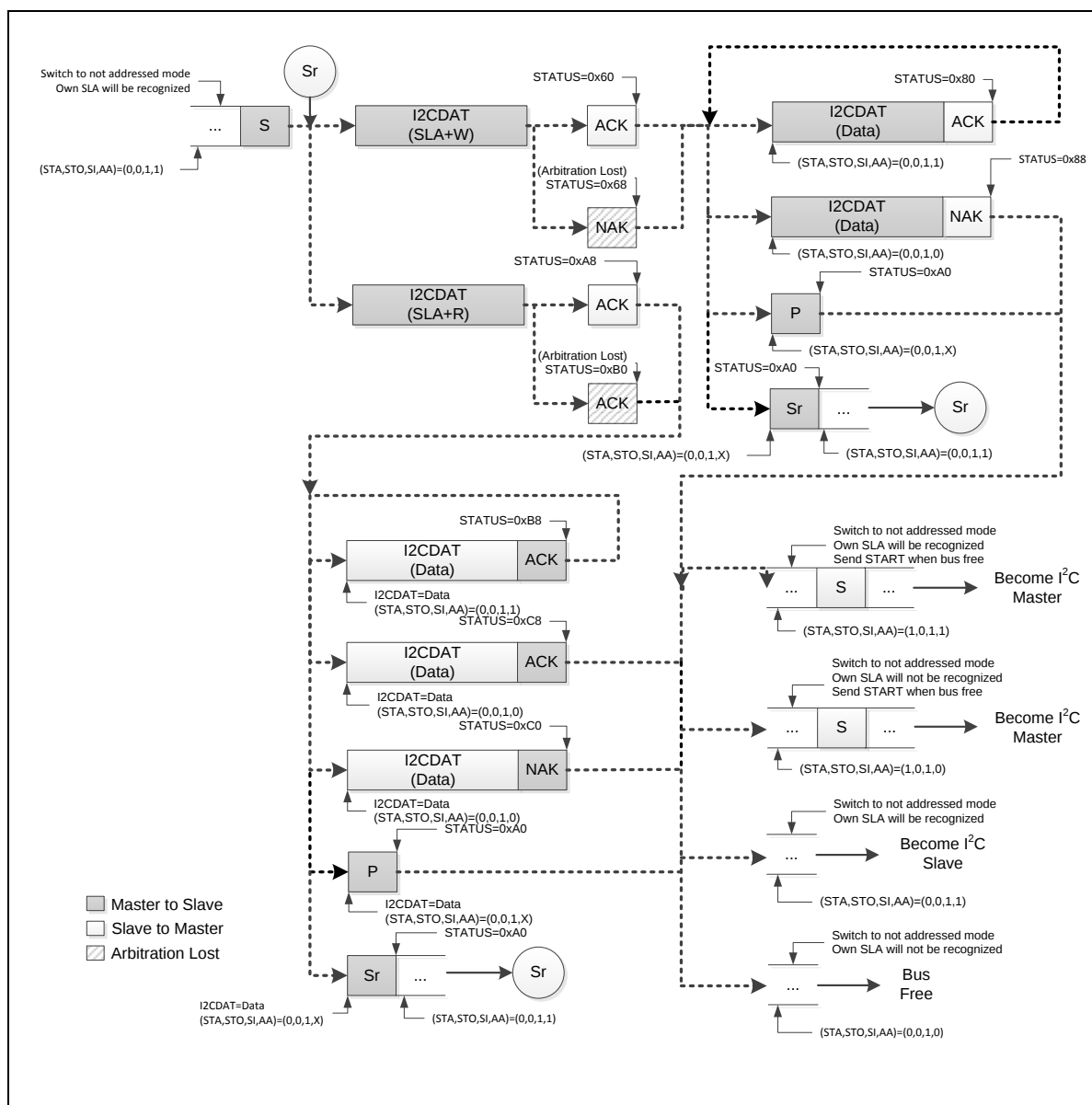


图 6.11-12 从机模式控制流程

如果I²C处在被寻址的从机接收数据模式却收到停止或重复起始信号，状态码是0xA0。当状态码是0xA0时，用户可以遵循如上图状态码是0x88的操作。

如果I²C处在被寻址的从机传输数据模式却收到停止或重复起始信号，状态码是0xA0。当状态码是0xA0时，用户可以遵循如上图状态码是0xC8的操作。

注意：从机获得0x88, 0xC8, 0xC0 和0xA0状态后,从机会切换到无地址模式，自身SLA不会被辨识。如果进入这种状态，从机不再从主机接收任何信号或地址。需要复位才能离开这种状态。

6.11.7.3 广播呼叫模式(GC)

如果 GC位(I2CADDRn [0]) 被置位，I²C端口硬件将响应广播呼叫地址(0x00).用户可以通过清GC位来禁止呼叫功能。当I²C在从机模式时GC 位被置位，可以接收主机地址(0x00)的广播呼叫，将遵循广播模式状态。

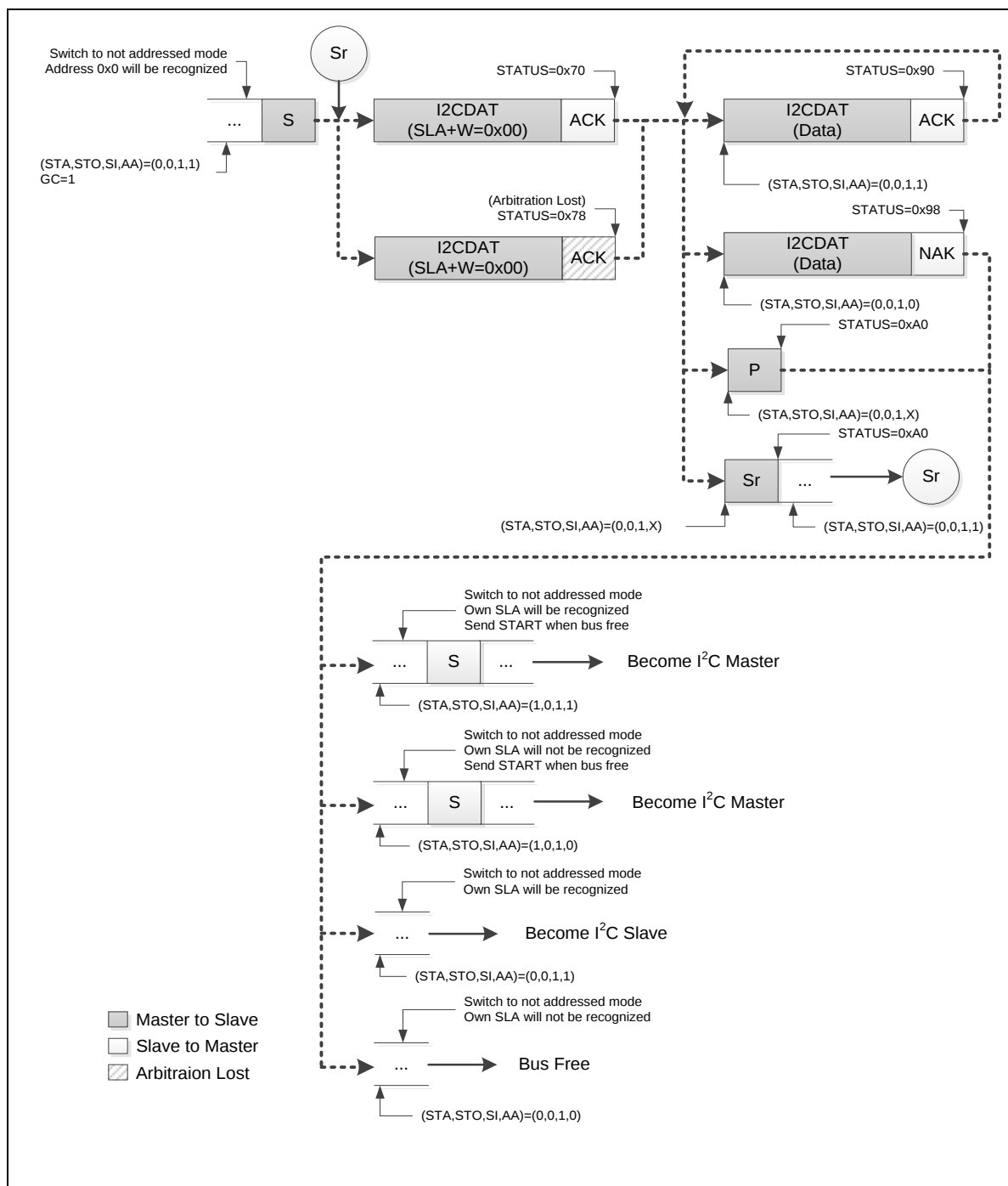


图 6.11-13 广播呼叫模式

如果I²C处在接收数据的广播呼叫模式，收到停止或重复起始信号，状态码是0xA0。当状态码是0xA0时，用户可以遵循如上图状态码是0x98的流程处理

注意：从机获得0x98 和0xA0 状态后，从机会切换到无地址模式并且自身SLA将不被辨识。如果进入这种状态，从机不再从主机接收任何信号或地址。此时，需要复位才能离开这种状态。

6.11.7.4 多主机模式

在一些应用中，在一个I²C总线上有多个主机访问从机，并有可能同时在传送数据。M058S 的I²C是支持多主机模式，并包含有冲突检测和仲裁，防止数据损坏。

- 当I2CSTATUS = 0x38，表示接收到一个仲裁丢失。仲裁丢失事件可能发生在发送起始位、数据位或者停止位期间。用户可以在总线空闲时设置(STA, STO, SI, AA) = (1, 0, 1, X)来再次发送起始位，或者设置(STA, STO, SI, AA) = (0, 0, 1, X)发送停止位，以返回到无地址从机模式。
- 当I2CSTATUS = 0x00，接收到一个“总线错误”，为从I²C总线错误中恢复过来，STO应被设置且SI应被清0，然后再把STO清0来释放总线
 - 设置(STA, STO, SI, AA) = (0, 1, 1, X)停止当前传输。
 - 设置(STA, STO, SI, AA) = (0, 0, 1, X)释放总线。

6.11.7.5 EEPROM随机读取例子

下面是I2C从EEPROM读取数据，配置I2C相关寄存器的流程

1. 在“GP3_MFP”和“ALT_MFP”寄存器中将多功能管脚设置成SCL 和SDA管脚。
2. 通过使能“APBCLK”寄存器的I2C_EN=1使能I2C APB时钟。
3. 通过设置“IPRSTC2”寄存器I2C_RST= 1来复位I²C控制器，然后设置I2C_RST= 0将I²C控制器设置成普通操作。
4. 通过设置“I2CON”寄存器ENS1=1使能I²C控制器。
5. 通过在I2CLK寄存器写I²C时钟分频值来设置I²C时钟速度。
6. 在“NVIC_ISER”寄存器中设置SETENA=0x00040000来设置I²C IRQ。
7. 通过设置“I2CON”寄存器EI=1使能I²C中断
8. 设置I²C地址寄存器“I2CADDR0~I2CADDR3”。

随机读操作是访问EEPROM的其中一种方法。这个方法是允许主机访问EEPROM的任何一个地址。下图显示EEPROM随机读取操作过程

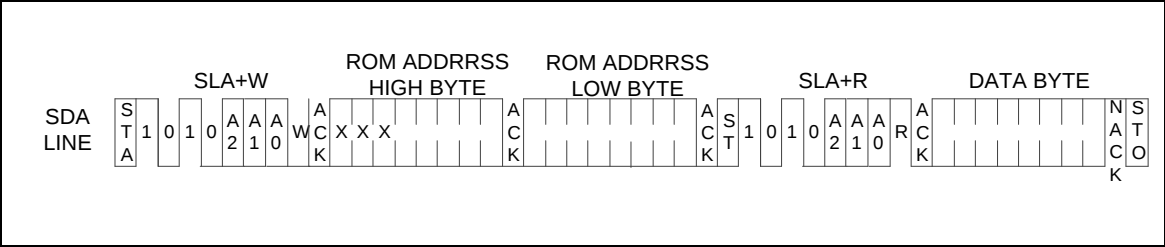


图 6.11-14 EEPROM 随机读取

下图显示如何使用I²C控制器来实现对EEPROM的随机读取协议

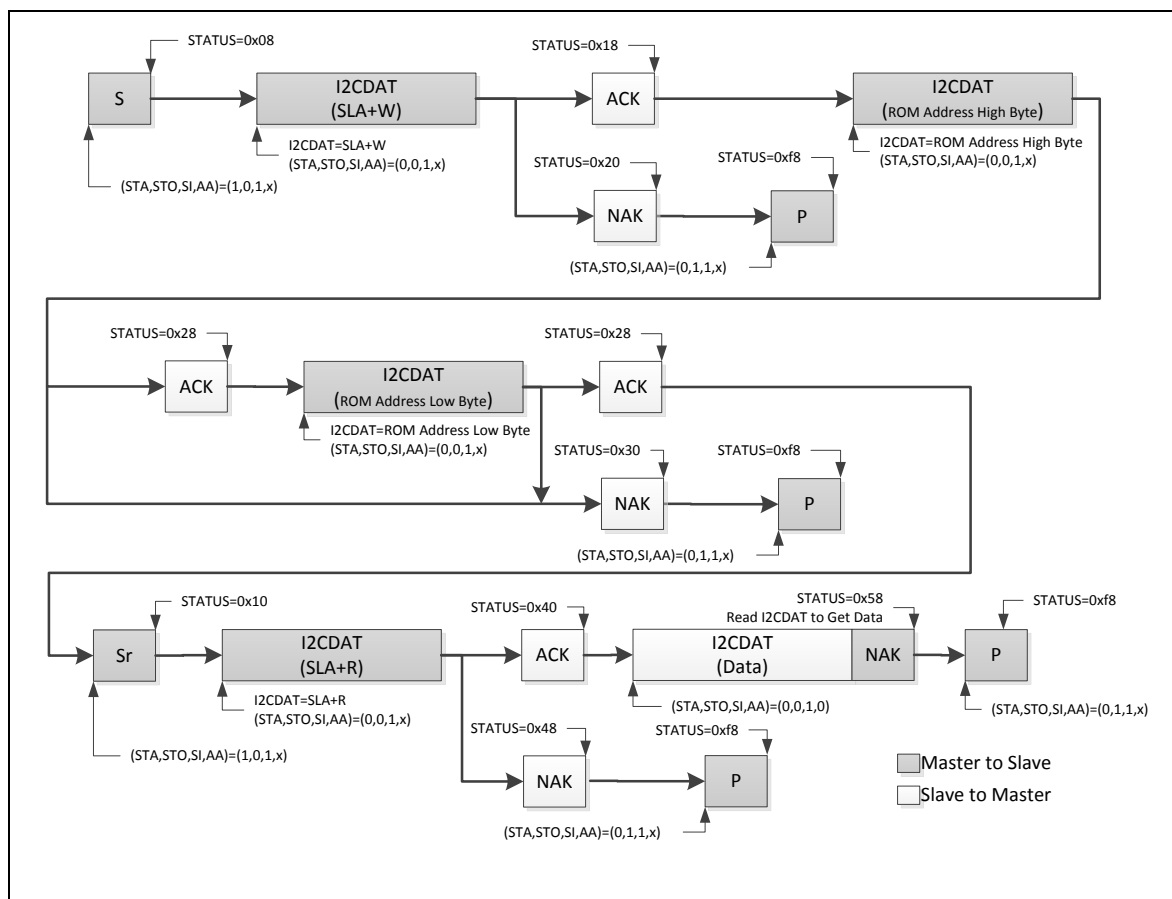


图 6.11-15 EEPROM 随机读协议

I²C控制器作为主机发送起始信号到总线，然后发送SLA+W (从机地址 + 写位)到EEPROM，跟着由两个字节数据地址来设置EEPROM被读的地址。然后是重复起始信号跟着SLA+R被发送来读EEPROM数据

6.11.8 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移量	R/W	描述	复位值
I²C基地址: I2C0_BA = 0x4002_0000 I2C1_BA = 0x4012_0000				
I2CON x=0,1	I2Cx_BA+0x00	R/W	I ² C 控制寄存器	0x0000_0000
I2CADDR0 x=0,1	I2Cx_BA+0x04	R/W	I ² C 从机地址寄存器0	0x0000_0000
I2CDAT x=0,1	I2Cx_BA+0x08	R/W	I ² C 数据寄存器	0x0000_0000
I2CSTATUS x=0,1	I2Cx_BA+0x0C	R	I ² C 状态寄存器	0x0000_00F8
I2CLK x=0,1	I2Cx_BA+0x10	R/W	I ² C时钟分频寄存器	0x0000_0000
I2CTOC x=0,1	I2Cx_BA+0x14	R/W	I ² C超时控制寄存器	0x0000_0000
I2CADDR1 x=0,1	I2Cx_BA+0x18	R/W	I ² C从机地址寄存器1	0x0000_0000
I2CADDR2 x=0,1	I2Cx_BA+0x1C	R/W	I ² C从机地址寄存器2	0x0000_0000
I2CADDR3 x=0,1	I2Cx_BA+0x20	R/W	I ² C从机地址寄存器3	0x0000_0000
I2CADM0 x=0,1	I2Cx_BA+0x24	R/W	I ² C 从机地址掩码寄存器0	0x0000_0000
I2CADM1 x=0,1	I2Cx_BA+0x28	R/W	I ² C从机地址掩码寄存器1	0x0000_0000
I2CADM2 x=0,1	I2Cx_BA+0x2C	R/W	I ² C从机地址掩码寄存器2	0x0000_0000
I2CADM3 x=0,1	I2Cx_BA+0x30	R/W	I ² C从机地址掩码寄存器3	0x0000_0000
I2CWKUPCON x=0,1	I2Cx_BA+0x3C	R/W	I ² C唤醒控制寄存器	0x0000_0000

I2C WKUPSTS x=0,1	I2Cx_BA+0x40	R/W	I ² C 唤醒状态寄存器	0x0000_0000
----------------------	--------------	-----	--------------------------	-------------

6.11.9 寄存器描述

I²C 控制寄存器 (I2CON)

寄存器	偏移量	R/W	描述	复位值
I2CON	I2Cx_BA+0x00	R/W	I ² C控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
EI	ENS1	STA	STO	SI	AA	保留	

位	描述	
[31:8]	保留	保留.
[7]	EI	使能中断 0 = 禁用 I ² C 中断 1 = 使能 I ² C 中断
[6]	ENS1	I ² C 控制器使能位 0 = 禁用 1 = 使能 设置使能 I ² C 串行控制器功能。当 ENS1=1, I ² C 串行功能使能。功能管脚功能必须先设置为 I ² C 功能。
[5]	STA	I ² C 起始控制位 设置 STA 为 1, 进入主机模式, 如果总线处于空闲状态, I ² C 硬件会送出 START 或 重复 START 条件。
[4]	STO	I ² C 停止控制位 在主机模式, 设置 STO 来传送一个 STOP 条件到总线, 然后 I ² C 硬件将会检查总线状况, 如果检测到一个 STOP 状况, 这个标志会被硬件自动清除。在 I ² C 从机模式, 设置 STO 复位 I ² C 硬件来定义“无地址”从机模式, 这表示在从机接收模式不再接收从主机设备发送的数据。
[3]	SI	I ² C 中断标志 当一个新的 I ² C 状态出现在寄存器 I2CSTATUS 时, SI 标志由硬件置位, 并且如果 EI (I2CON [7]) 位被置位, 则产生 I ² C 中断请求。SI 必须由软件通过向该位写 '1' 清零。
[2]	AA	应答控制位 当 AA=1 先于地址或数据接收, 在以下两种情况: 1.) 从机正在应答主机发送的地址。

		2.) 接收设备正在应答发送设备发送的数据，在SCL线上的应答时钟脉冲期间将返回一个应答（SDA上的低电平）。当 AA = 0 先于地址或数据接收，则在ISCL线上的应答时钟脉冲期间将返回一个非应答（SDA上的高电平）。
[1:0]	保留	保留。

I²C数据寄存器(I2CDAT)

寄存器	偏移量	R/W	描述	复位值
I2CDAT	I2Cx_BA+0x08	R/W	I ² C数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
I2CDAT[7:0]							

位	描述	
[31:8]	保留	保留.
[7:0]	I2CDAT	I ² C 数据寄存器 该区域 为8 位 I ² C 串行端口的传输数据

I²C状态寄存器(I2CSTATUS)

寄存器	偏移量	R/W	描述	复位值
I2CSTATUS	I2Cx_BA+0x0C	R	I ² C状态寄存器	0x0000_00F8

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
I2CSTATUS[7:0]							

位	描述	
[31:8]	保留	保留.
[7:0]	I2CSTATUS	I²C 状态寄存器 最后三个位总是0，高5位包含状态代码。共有26个可能状态码。当I2CSTATUS值是0xF8没有串行中断请求。 所有其他的 I2CSTATUS的值对应 I²C 的状态。当进入其中任一状态时，就会产生状态中断请求 (SI= 1)。在 SI 被硬件置位或 SI 被软件复位后一个机器周期，有效状态码出现在 I2CSTATUS中。 此外，0x00状态表示总线错误。总线错误发生在 START 或 STOP 条件出现在帧结构不正确的位置。不正确的位置比如是在串行传输地址字节、数据字节或应答位期间。

I²C时钟分频寄存器(I2CLK)

寄存器	偏移量	R/W	描述	复位值
I2CLK	I2Cx_BA+0x10	R/W	I ² C时钟分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
I2CLK[7:0]							

位	描述	
[31:8]	保留	保留.
[7:0]	I2CLK	I²C 时钟分频寄存器 I²C 数据波特率 = PCLK / (4x (I2CLK+1)). 注意: I2CLK 的最小值是 4.

I²C 超时计数器寄存器(I2CTOC)

寄存器	偏移量	R/W	描述	复位值
I2CTOC	I2Cx_BA+0x14	R/W	I ² C 超时计数器寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留					ENTI	DIV4	TIF

位	描述	
[31:3]	保留	保留。
[2]	ENTI	超时计数器使能控制 0 = 禁用 1 = 使能 注意：当使能该位，则14-位溢出定时器将会在 SI 清零后开始计数。SI 置1将会复位计数器，SI 位清零之后，计数器会重新开始计数。
[1]	DIV4	超时定时器输入时钟除以 4 0 = 时钟除以 4 禁用 1 = 时钟除以 4 使能 注意：该位使能后，超时时间扩大 4 倍。
[0]	TIF	超时溢出标志 I²C 当超时发生时，该位由硬件置位，如果此时 I²C 的中断使能位 EI 置1，则可引发 CPU 的中断。 注意：写 1 清除该位。

I²C从机地址寄存器(I2CADDRx)

寄存器	偏移量	R/W	描述	复位值
I2CADDR0	I2Cx_BA+0x04	R/W	I ² C从机地址寄存器0	0x0000_0000
I2CADDR1	I2Cx_BA+0x18	R/W	I ² C从机地址寄存器1	0x0000_0000
I2CADDR2	I2Cx_BA+0x1C	R/W	I ² C从机地址寄存器2	0x0000_0000
I2CADDR3	I2Cx_BA+0x20	R/W	I ² C从机地址寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
I2CADDR[7:1]							GC

位	描述	
[31:8]	保留	保留.
[7:1]	I2CADDR	I ² C 地址寄存器 主机模式下, 该寄存器内容没有意义。从机模式下, 高七位作为芯片本身的地址。如果地址符合, I ² C 硬件将会自动应答。
[0]	GC	广播呼叫功能 0 = 禁用广播呼叫功能 1 = 使能广播呼叫功能

I²C从机地址掩码寄存器(I2CADMx)

寄存器	偏移量	R/W	描述	复位值
I2CADM0	I2Cx_BA+0x24	R/W	I ² C从机地址掩码寄存器0	0x0000_0000
I2CADM1	I2Cx_BA+0x28	R/W	I ² C从机地址掩码寄存器1	0x0000_0000
I2CADM2	I2Cx_BA+0x2C	R/W	I ² C从机地址掩码寄存器2	0x0000_0000
I2CADM3	I2Cx_BA+0x30	R/W	I ² C从机地址掩码寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
I2CADM[7:1]							保留

位	描述	
[31:8]	保留	保留.
[7:1]	I2CADM	I ² C 地址掩码寄存器 0 = 禁用地址掩码（接收到的相应地址必须完全符合地址寄存器） 1 = 使能地址掩码（接收到的相应地址位不予辨识）
[0]	保留	保留.

I²C唤醒控制寄存器(I2CWKUPCON)

寄存器	偏移量	R/W	描述	复位值
I2CWKUPCON	I2Cx_BA+0x3C	R/W	I ² C唤醒控制寄存器	0x0000_0000

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

位	描述	
[31:1]	保留	保留.
[0]	WKUPEN	I ² C唤醒功能使能 1 = I ² C 唤醒功能使能. 0 = I ² C 唤醒功能禁止.

I²C唤醒状态寄存器(I2CWKUPSTS)

寄存器	偏移量	R/W	描述	复位值
I2CWKUPSTS	I2Cx_BA+0x40	R/W	I ² C唤醒状态寄存器	0x0000_0000

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

位	描述	
[31:1]	保留	保留.
[0]	WKUPIF	I ² C唤醒中断标志 当芯片在掉电模式下由I ² C中断唤醒后，该位被置1。该位软件写1清零

6.12 串行外围设备接口(SPI)

6.12.1 概述

串行外围设备接口(SPI) 是一个工作于全双工模式的同步串行数据通讯协议。设备可工作在主/从模式，使用4线双向接口相互通讯。NuMicro™ M058S系列带一组SPI控制器，当从一个外围设备接收数据时，SPI执行串-并的转换，而在数据向外围设备发送时执行并-串的转换。该SPI控制器可以配置为主设备或从设备。

6.12.2 特性

- 一组 SPI 控制器
- 支持主机和从机工作模式
- 传输位长度可配置
- 一次传输可收发两个字数据
- 带 FIFO 缓存
- 支持MSB或LSB优先传输
- 支持字节重排序功能
- 支持字节或者字休眠模式
- 支持丛机3线模式
- SPI总线时钟速率可配置成与系统时钟相等

6.12.3 框图

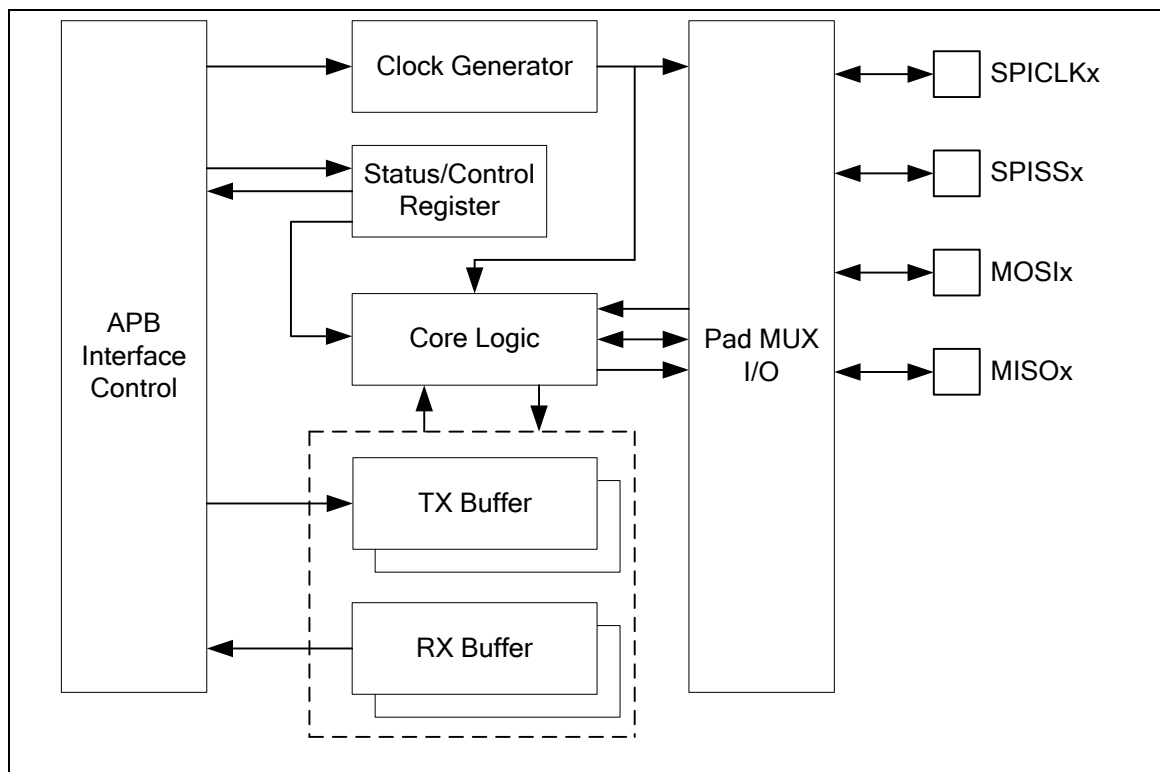


图 6.12-1 SPI 框图

6.12.4 基本配置

SPI管脚功能由P1_MFP寄存器来配置。SPI的时钟通过寄存器APBCLK[12]位来使能。通过CLKSEL1[4] and CLKSEL1[5]位来选择SPI的时钟源

6.12.5 功能描述

6.12.5.1 术语

SPI外设时钟和SPI总线时钟

SPI控制器需要SPI外设时钟来驱动SPI逻辑单元执行数据传输。SPI总线时钟就是SPICLKx管脚上的时钟。

M058S SPI外设主/从模式下，时钟速率决定于时钟源、BCn (SPI_CNTRL2[31])选项和时钟分频器(SPI_DIVIDER[7:0])的设置。寄存器CLKSEL1的SPIx_S位决定SPI外设时钟的时钟源。时钟源可以是HCLK或是PLL输出时钟。设置BCn位为0，可兼容早先产品的SPI时钟速率计算。寄存器SPI_DIVIDER的DIVIDER位的设定值决定时钟速率计算时的分频值。

M058S SPI 主模式下，SPI 外设时钟等于SPI总线时钟。

M058S SPI 从模式下，SPI 总线时钟由片外主模式芯片提供。从机设备的SPI外设时钟速率必须快于连接在一起的主机设备的SPI总线时钟速率。不论工作在主机还是从机模式，SPI外设的时钟频率不能快于APB时钟速率。

主/从机模式

SPI控制器可通过设置寄存器SPI_CNTRL的第18位SLAVE配置成主机或从机模式，来与片外SPI从机或者主机通信。主机模式与从机模式的应用框图如下：

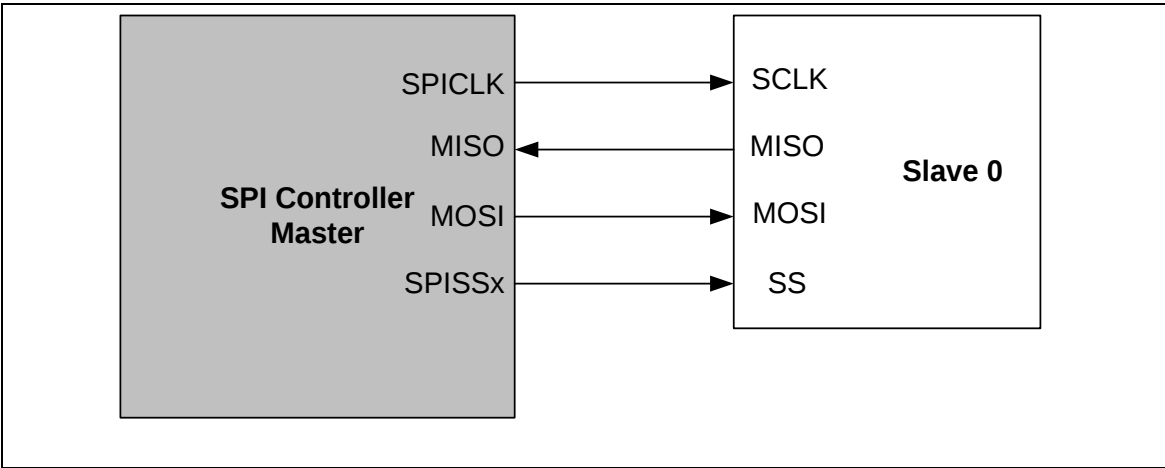


图 6.12-2 SPI 主机模式应用框图

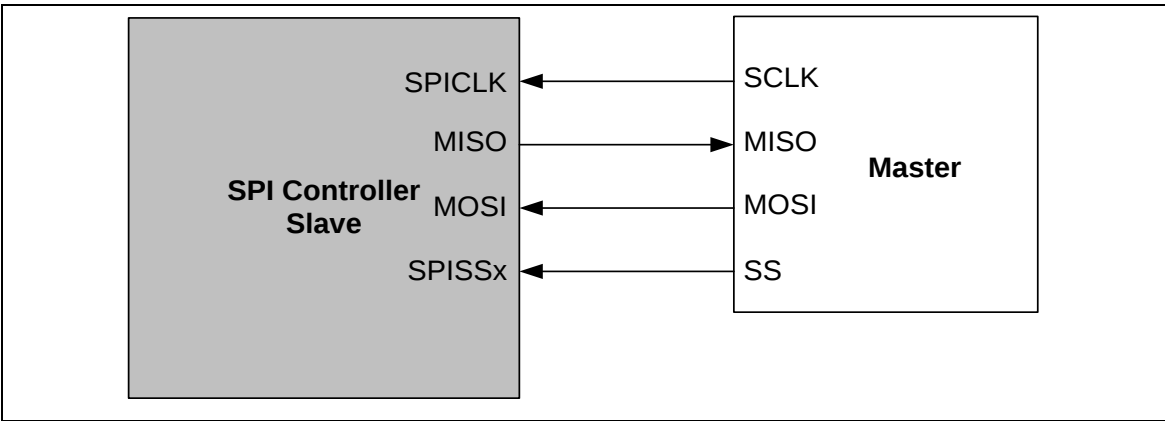


图 6.12-3 SPI 从机模式应用框图

时钟极性

CLKP位(SPI_CTL[11])定义总线时钟的空闲状态。如果CLKP=1，空闲时SPICLK输出为高电平状态，否则如果CLKP=0，空闲时SPICLK输出为低电平状态。

发送/接收位长

一个事务的位长由TX_BIT_LEN位域(SPI_CNTRL[7:3])来定义。对于发送和接收，在一个事务中，位长可配置为多达32位。

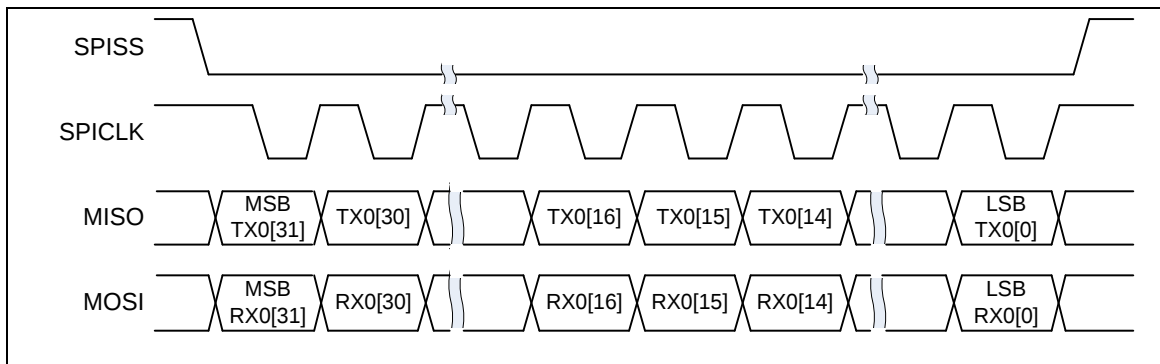


图 6.12-4 一次传输 32-位长度 (主模式下)

LSB优先

LSB位(SPI_CNTRL[10])定义了一个事务中位的传输序列是从LSB还是MSB开始。如果设定LSB位(SPI_CNTRL[10])为1，传输序列为LSB优先。否则如果LSB位为0，传输序列是MSB优先。

发送边沿

TX_NEG 位 (SPI_CNTRL[2])定义了数据在SPI总线时钟SPICLK的下降沿还是上升沿被发送出去。

接收边沿

Rx_NEG 位 (SPI_CNTRL[1])定义了数据在SPI总线时钟SPICLK的下降沿还是上升沿被接收。

注意：TX_NEG和RX_NEG的设置是相互排斥的，换句话说就是不要在相同的时钟沿发送和接收数据。

字休眠

在主机模式下，SP_CYCLE (SPI_CNTRL[15:12])提供一个在两个连续事务之间可配置的休眠时间间隔。休眠间隔指的是从前一个事务的最后一个时钟沿到下一个传输事务的第一时钟沿之间的时间间隔。

SP_CYCLE的默认值是0x3(3.5 SPI总线时钟周期)。如果软件禁止FIFO模式，该SP_CYCLE设定值对字休眠间隔不起作用。

从机选择

在主机模式下，SPI控制器能通过从机选择输出脚SPISSx来驱动片外从机设备。在从机模式，片外主机设备通过SPISSx输入端口驱动从机选择信号到SPI控制器。在主/从机模式下，从机选择信号的有效状态可以通过设置SS_LVL位(SPI_SSR[2])来设定为低有效或高有效，SS_LTRIG位(SPI_SSR[4])来设定从机选择信号SPISSx为电平触发还是边沿触发。触发条件的选择取决于所连设备的从机/主机的类型。

在从机模式下，如果SS_LTRIG位被配置成电平触发，则LTRIG_FLAG位(SPI_SSR[5])用来表示在一个事务中，接收到的数据个数和接收到的数据位数是否满足TX_NUM和TX_BIT_LEN的设定值（传输完成的意思是，当从机选择信号无效或SPI控制器完成了一个数据传输引起单位传输中断标志被置1）。

电平触发/边沿触发

在从机模式下，从机选择信号可以被配置为电平触发或边沿触发。边沿触发模式，数据从一个有效的从机选择信号边沿开始，到一个无效的从机选择信号边沿结束。当检测到一个无效边沿，(SPI_CNTRL[16])单元传输中断标志将设置为1。如果主机没有发送一个无效边沿给从机，传输过程不会完成，从机的单元传输中断标志不会被置位。电平触发模式，当下面的两个条件中的一个发生，从机的单元传输中断标志将会被置位。第一个条件是传输的数据位个数与TX_BIT_LEN位的设定值相匹配时，从机的单元传输中断标志将会被置位。同样第二个条件是，如果主机正在传输期间的时候设置从机选择管脚为无效电平，将会强制从机设备终结当前传输，而不管已经传输了多少位数据，从机的单元传输中断标志将会被置位。用户可以读取LTRIG_FLAG位状态位来检查数据是否已经全部传完。

6.12.5.2 自动从机选择

在主机模式下，如果AUTOSS位(SPI_SSR[3])设置为1，从机选择信号将会自动产生，并根据SSR位(SPI_SSR[0])是否使能，将从机选择信号输出到SPISSx管脚上。这意味着当通过设置GO_BUSY位(SPI_CNTRL[0])来开始数据传输时，从机选择信号自动设置为有效状态，在数据传输结束后自动被设置为无效状态。如果AUTOSS被清零时，从机选择输出信号需要通过手工置位/清除SSR位，从而使从机进入激活或非激活状态。从机选择输出信号的激活电平状态由SS_LVL位(SPI_SSR[2])来定义。

6.12.5.3 字节重排序功能

当传输设置为MSB 优先(LSB = 0)且字节重排序功能被使能，当位长度被配置为32位(TX_BIT_LEN=0)，存储在TX 缓存与RX 缓存的数据将按[Byte0, Byte1, Byte2, Byte3]的顺序重新排列。数据将以Byte0, Byte1, Byte2,和Byte3的顺序进行发送/接收。如果TX_BIT_LEN位被置位设为24-位模式，存储在TX 缓存与RX 缓存的数据将重新按[未知byte, Byte0, Byte1, Byte2]的顺序排列。SPI 控制器将按照BYTE0, BYTE1, BYTE2 的顺序发送/接收数据，每个字节MSB 优先发送/接收。16-位模式的规则与上面相同。字节重排序功能只在TX_BIT_LEN为16, 24, 和32 位时适用。

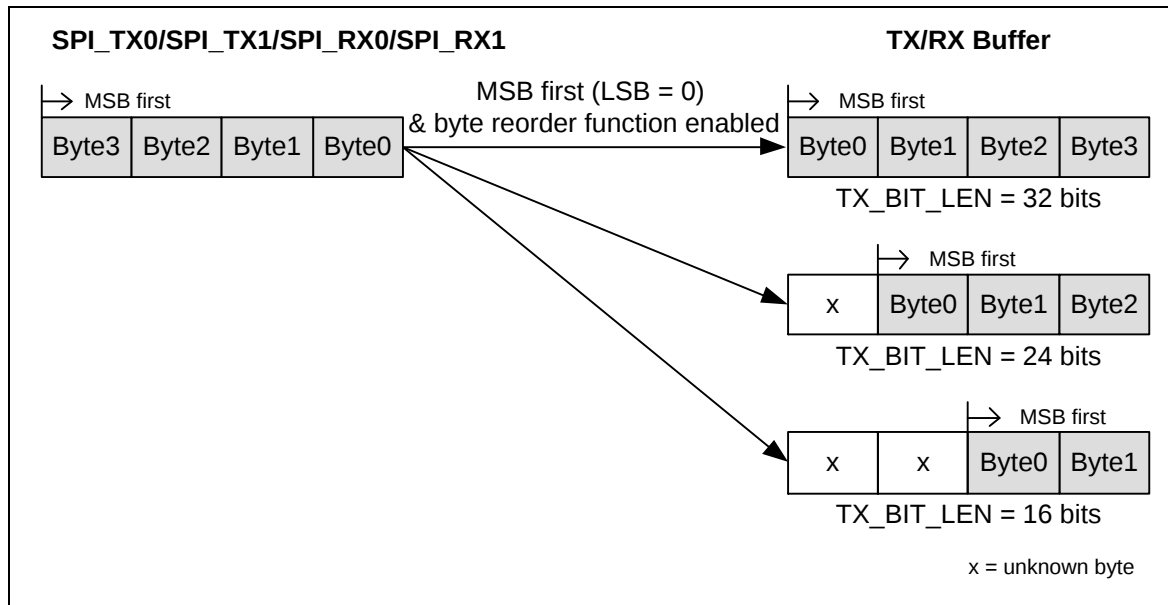


图 6.12-5 字节重排序

6.12.5.4 字节休眠功能

字节休眠间隔和字休眠间隔都在SP_CYCLE寄存器中配置。

主机模式下，如果把SPI_CNTRL[19]置1使能字节重排序功能，硬件将会在两个连续传输字节之间插入一个0.5 ~ 15.5个串行时钟周期的休眠间隔。TX_BIT_LEN可以配置为16,24或32位长度。

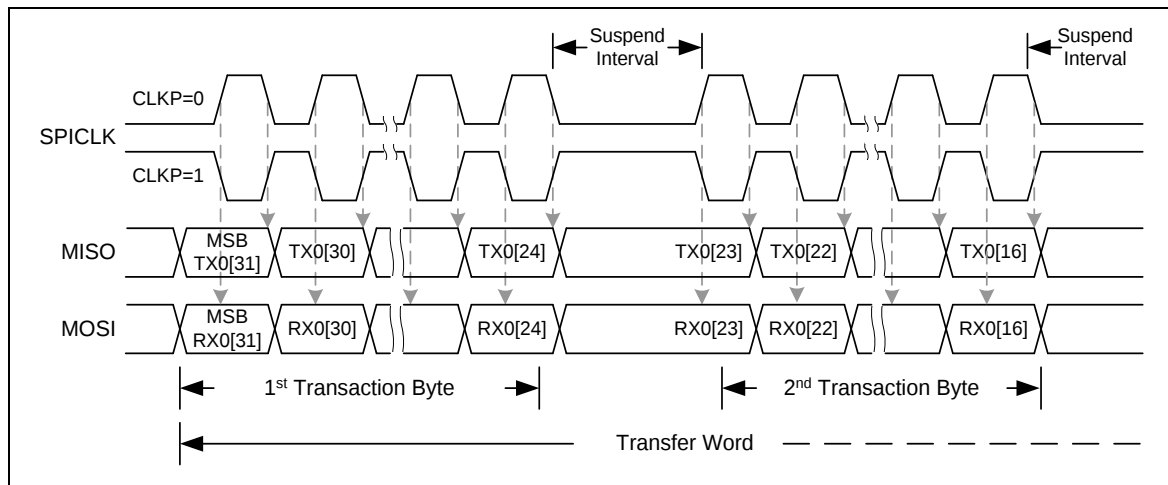


图 6.12-6 字节休眠时序波形

6.12.5.5 从机3-线模式

当NOSLVSEL位(SPI_CNTRL2[8])由软件置1使能从机3线模式时，在从机模式下，SPI控制器可以在没有片选信号下工作。NOSLVSEL位仅在从机模式下有效。在与一个主机通讯时，仅需要三个管脚，SPICLK, MISO和MOSI。SPISS脚可以配置成一个GPIO。当NOSLVSEL位被设为1时，在GO_BUSY位被置为1后，SPI从机将开始准备发送和接收数据。当接收数据位的个数满足TX_BIT_LEN和TX_NUM定义时，单元传输中断标志IF(SPI_CNTRL[16])将会置位。

注意:在从机3-线模式下，SS_LTRIG(SPI_SSR[4])必须设为1。

6.12.5.6 FIFO 模式

当 FIFO (SPI_CNTRL[21]) 设为1时，SPI控制器支持FIFO模式。SPI控制器配备了4个32-bit发送和接收FIFO缓存。

发送FIFO缓存是一个4层深度，32位宽，先进先出的寄存器缓存。数据可以通过软件写SPI_TX0寄存器写入发送FIFO缓存。存储在发送FIFO缓存的数据会被传输控制逻辑读取和发送。如果4层发送FIFO缓存满了，TX_FULL位会被置1。当SPI传输逻辑单元抽出发送FIFO缓存的最后一个数据，那么4层发送FIFO缓存为空，TX_EMPTY位被置1。注意最后一笔传输还在进行时，TX_EMPTY标志已被置1。

接收FIFO缓存也是一个4层深度，32位宽，先进先出的寄存器缓存。接收控制逻辑存储接收到的数据到该缓存。FIFO缓存数据可以通过软件从SPI_RX0寄存器读取。FIFO还带有一些相关的状态位，如RX_EMPTY和RX_FULL，用来指示当前FIFO缓存的状态。

FIFO模式下，发送和接收阈值可以通过软件设置TX_THRESHOLD和RX_THRESHOLD来设定。当存储在发送FIFO缓存的有效数据计数小于或等于TX_THRESHOLD设定，TX_INTSTS位会被置1。当存储在接收FIFO缓存的有效数据计数大于RX_THRESHOLD设定，RX_INTSTS位会被置1。

在FIFO模式，4个数据可以事先通过软件写入SPI发送FIFO缓存。当SPI控制器工作在FIFO模式，SPI_CNTRL寄存器中的GO_BUSY位由硬件控制，SPI_CNTRL寄存器的内容不能被软件修改，除非FIFO位被清0禁止FIFO模式。

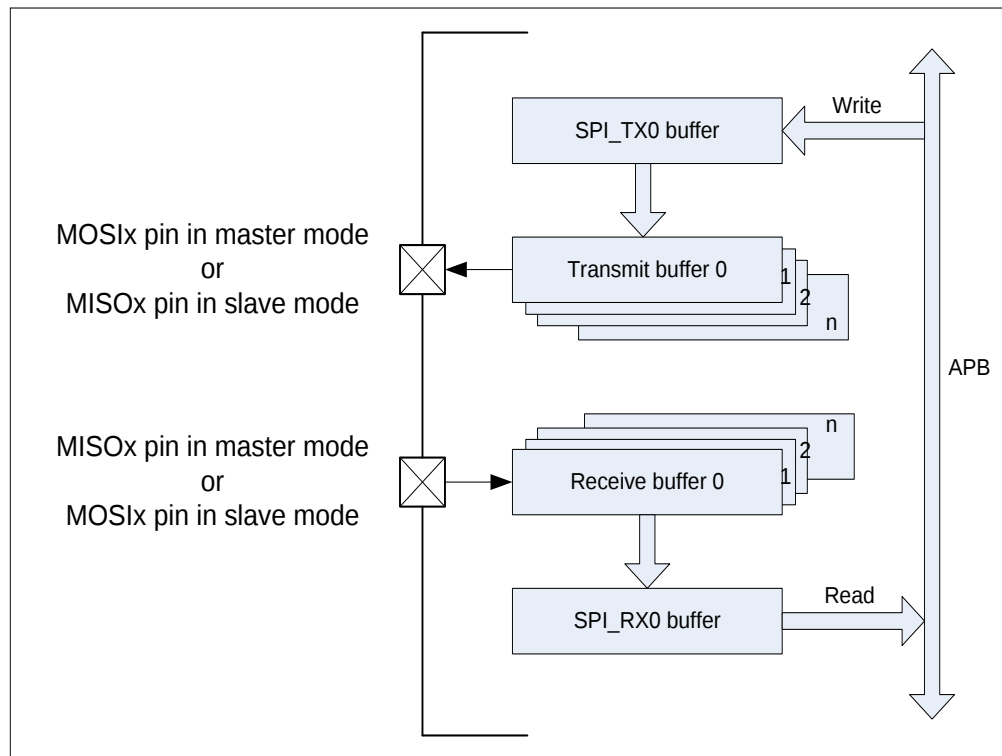


图 6.12-7 FIFO 模式框图

在主机发送操作中，当FIFO位设置为1，且软件写第一个数据到SPI_TX0寄存器时，TX_EMPTY标志将会被清0。只要发送FIFO缓存非空，发送操作立即开始。用户可以立即写下一个要发送的数据。在FIFO模式下，SPI控制器将会在两个连续的事务之间插入一个休眠间隔，休眠间隔的长度由SP_CYCLE (SPI_CNTRL [15:12]) 的设定值决定。只要TX_FULL标志为0，用户可以一直写数据到SPI_TX0寄存器。

如果要发送的数据更新及时，接下来的事务将会被自动触发。如果在所有数据传输完成之后，SPI_TX0 寄存器没有被更新，则传输停止。

在主机接收操作中，串行数据从MISOx管脚接收并被存储在接收 FIFO 缓存。当接收 FIFO 缓存中包含未读的数据时，RX_EMPTY将会被清除为 0。只要RX_EMPTY标志为 0，软件就可以从SPI_RX0 寄存器读取接收到的数据。如果接收 FIFO 缓存包含 4 个未读数据，RX_FULL标志将会被设置1。此时，SPI控制器将会停止接收数据，直到软件读取SPI_RX0寄存器。

从机模式下，当 FIFO 位被设置为 1，GO_BUSY 位将会被硬件自动设置为 1。如果用户想停止从机模式SPI数据传输，FIFO位和GO_BUSY位必须通过软件清零。

在从机发送操作中，当软件写数据到 SPI_TX0 寄存器，数据将会被加载到发送 FIFO 缓存，且TX_EMPTY标志将会被清0。当从设备从主机接收到时钟信号，发送操作将会开始。只要TX_FULL标志为 0，软件就可以写数据到 SPI_TX0 寄存器。在所有数据都被 SPI 发送逻辑单元发送出去，且软件没有更新 SPI_TX0 寄存器，TX_EMPTY标志将会被设置为 1。

在从机接收操作中，串行数据从MOSIx管脚接收，并被存储到接收SPI_RX0缓存寄存器。接收机制与主机模式接收操作类似。

6.12.5.7 中断

SPI单位传输中断

当SPI控制器完成一个单位传输，单位传输中断标志IF (SPI_CNTRL[16]) 将会被置1。如果中断使能位 IE (SPI_CNTRL[17]) 置位，则单位传输中断事件将会给CPU产生中断。单位传输中断标志位只能写 1 清零。

SPI从机 3-线模式开始中断

在 3 线模式下，当从机检测到 SPI 时钟信号时，3 线模式会产生开始中断标志，SLV_START_INTSTS (SPI_CNTRL2[11])将会被置为 1。如果SSTA_INTEN位 (SPI_CNTRL2[10])被设置1，SPI控制器将会触发一个中断。如果接收到的数据位小于 TX_BIT_LEN 的设定要求，在由用户定义的期望的时间内再没有串行时钟输入，用户可以设置 SLV_ABORT(SPI_CNTRL2[9])位来中止当前传输。如果软件设置 SLV_ABORT位为1，单元传输中断标志IF将会被置位。

接收 FIFO 超时中断

FIFO 模式下，有超时功能通知用户。如果超时中断使能位FIFO_CTL[21]置1，在FIFO里有一个接收到的数据，并且在主机模式下超过64个SPI引擎时钟周期，或在从机模式下超过576个SPI引擎时钟周期，没有被软件读取，则将会发出一个超时中断。

发送 FIFO 中断

FIFO模式下，如果发送FIFO缓存的有效数据计数小于或等于TX_THRESHOLD的设定值，发送FIFO中断标志会被置1。如果SPI_FIFO_CTL[3]置1，发送FIFO中断使能，SPI控制器会产生一个发送FIFO中断到系统。

接收 FIFO 中断

FIFO 模式下，如果接收FIFO缓存的有效数据计数大于RX_THRESHOLD的设定值，接收FIFO中断标志会被置1。如果SPI_FIFO_CTL[2]置1，接收FIFO中断使能，SPI控制器会产生一个接收FIFO中

断到系统。

6.12.6 时序图

从机选择信号的有效状态可以由 SS_LVL (SPI_SSR[2]) 位和 SS_LTRIG (SPI_SSR[4]) 位来定义。串行时钟 (SPICLK) 的空闲状态可以通过 CLKP 位 (SPI_CNTRL[11]) 配置为高电平或低电平。传输数据位长度在 TX_BIT_LEN (SPI_CNTRL[7:3])中定义,发送/接收数据的顺序是以 MSB 或 LSB 优先, 由 LSB 位 (SPI_CNTRL[10]) 来定义。用户可以通过设置 TX_NEG/RX_NEG (SPI_CNTRL[2] / SPI_CNTRL[1])位来选择发送/接收数据时串行时钟的边沿。四种主机/从机的SPI操作时序及相关设置如下。

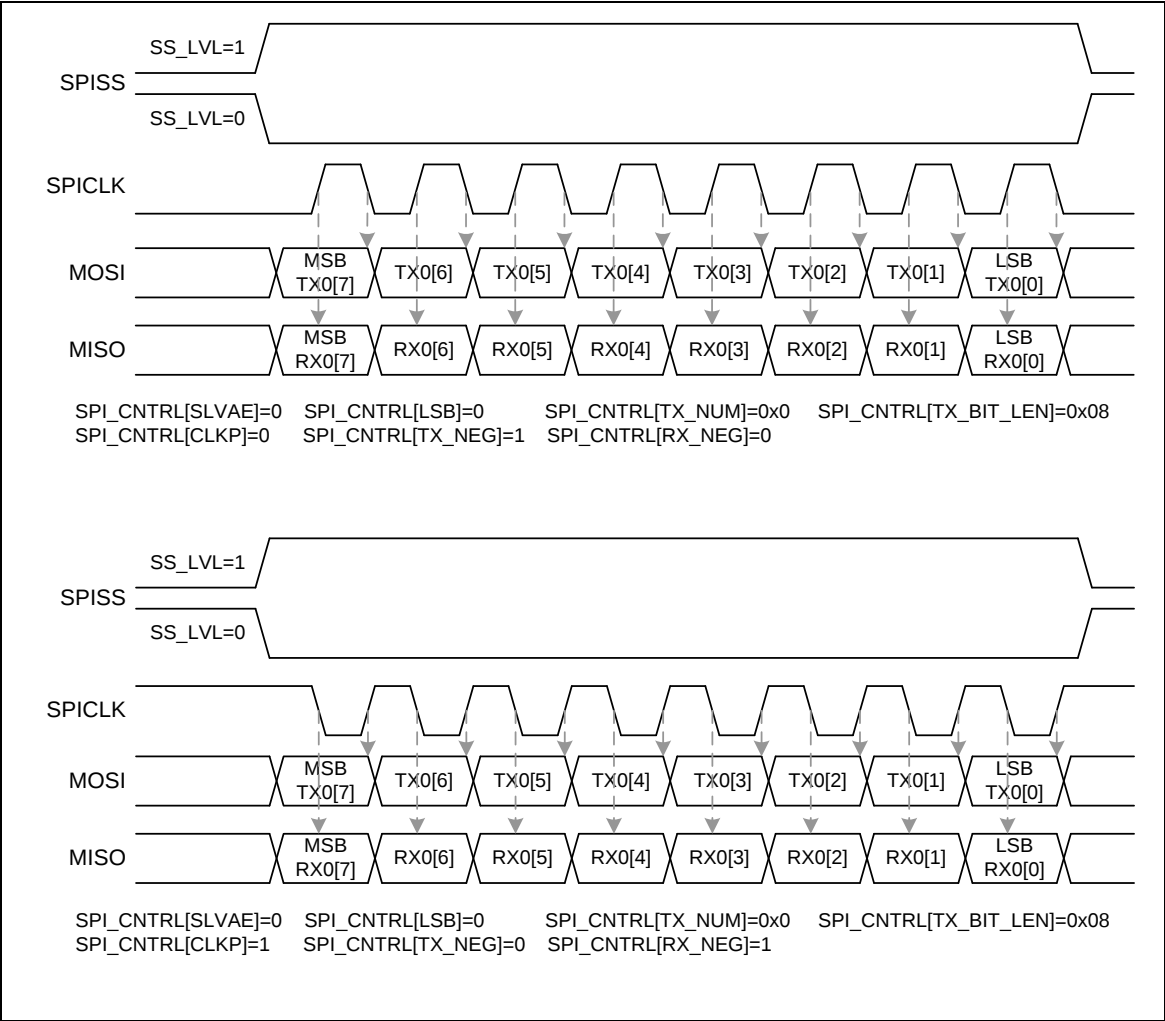


图 6.12-8 SPI 主机模式下的时序

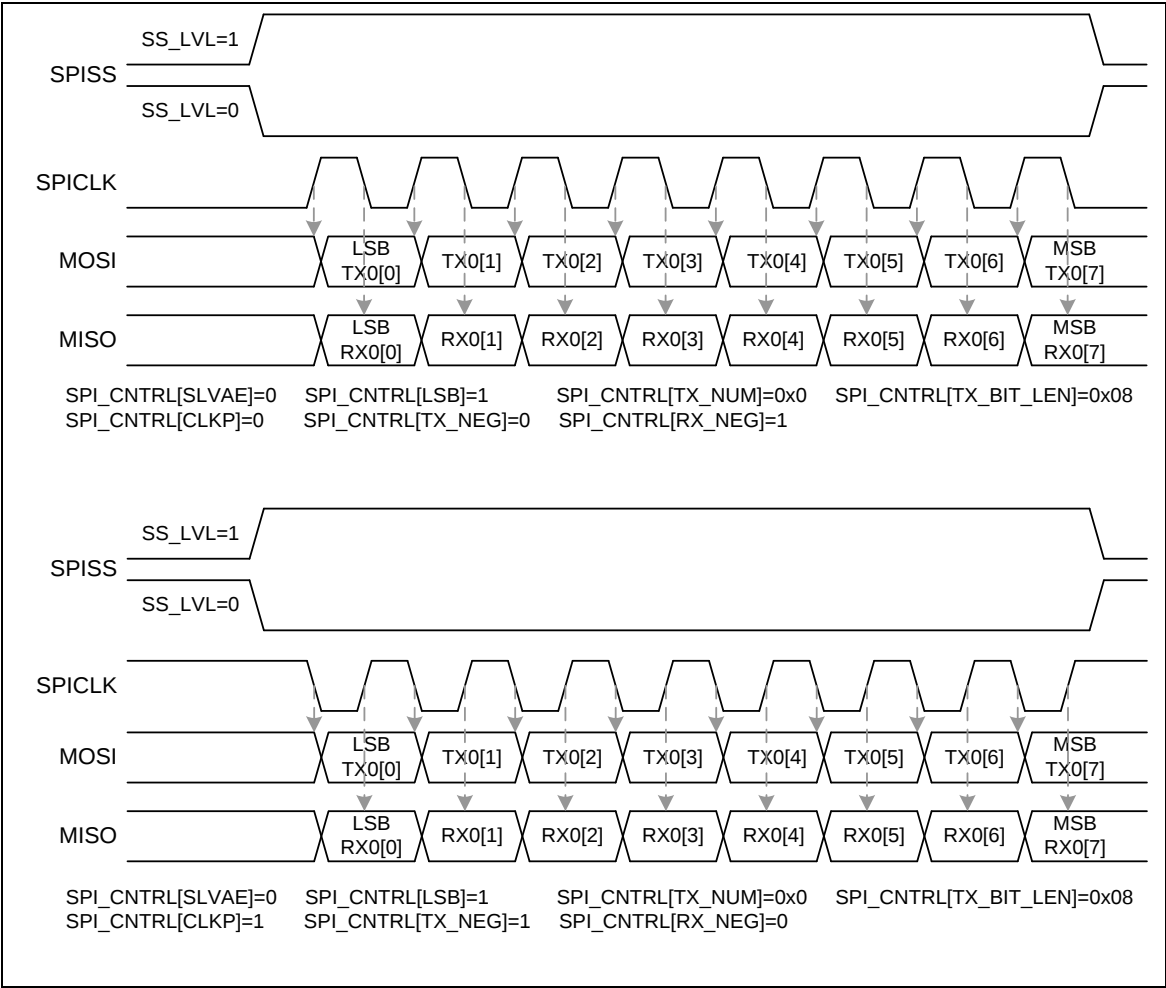


图 6.12-9 SPI 主机模式下的时序(交替 SPI 时钟相位)

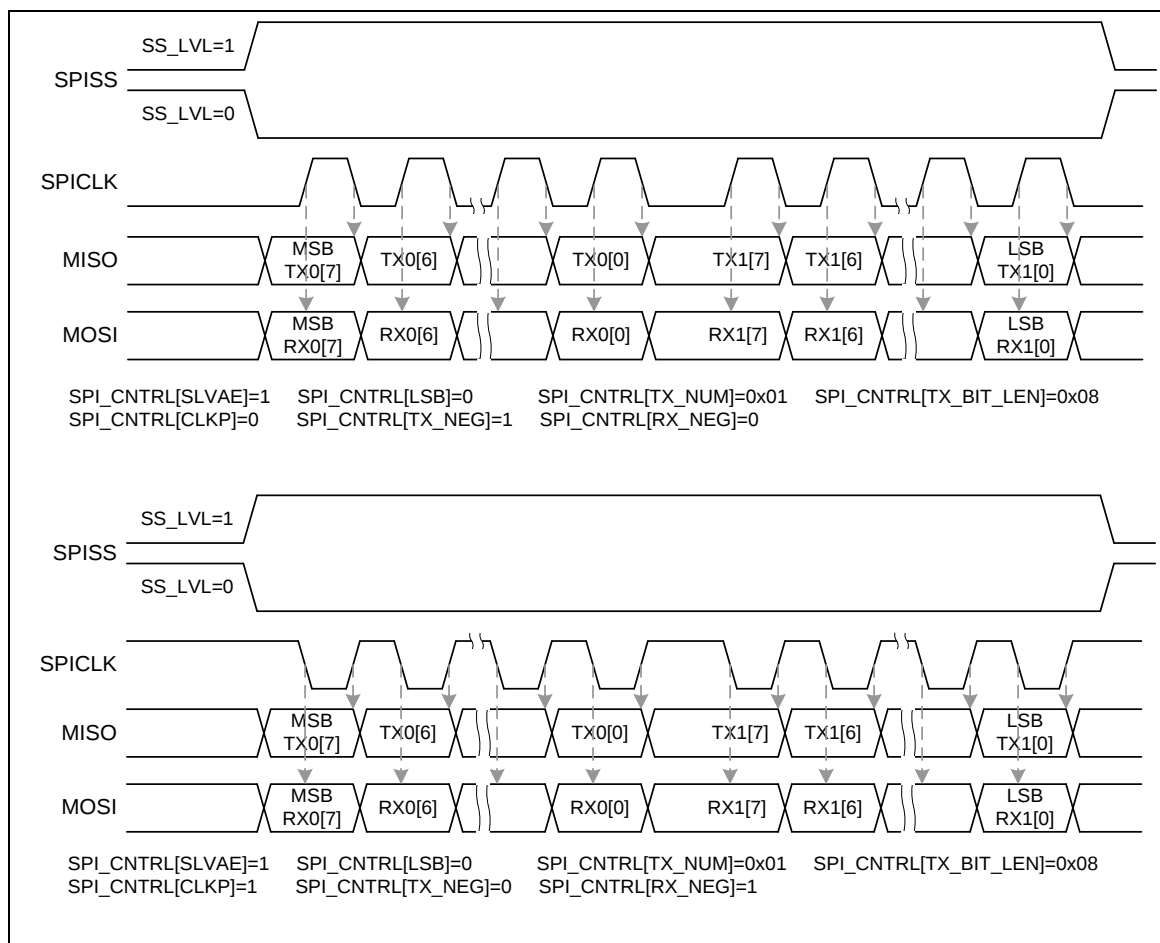


图 6.12-10 SPI 从机模式下的时序

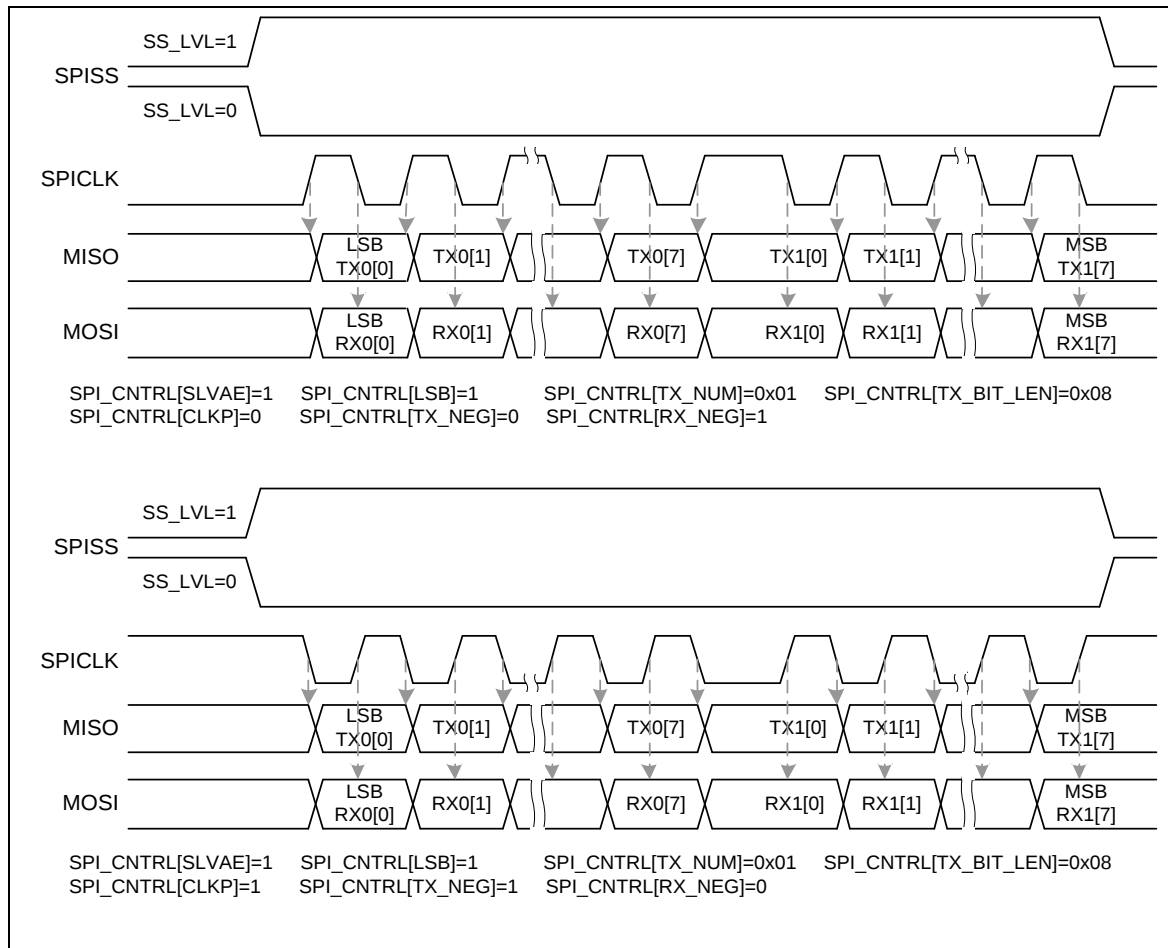


图 6.12-11 SPI 从机模式下的时序(交替 SPI 时钟相位)

6.12.7 编程例子

例1: SPI 控制器作为主机去访问一个片外从机设备，过程如下：

- 数据在串行时钟上升沿锁存
- 数据在串行时钟下降沿传输
- MSB 先传输
- 8位数据宽度
- SPICLK 空闲模式为低电平状态
- 每次只发送/接收一个字节
- 一个片外从机设备相连。从机选择信号为低电平有效

操作流程如下

- 1) 设置 DIVIDER (SPI_DIVIDER [7:0]) 寄存器来决定串行时钟输出频率。
- 2) 写一个合适的值到SPI_CNTRL寄存器来对主模式相关配置进行设定

1. SLAVE(SPI_CNTRL[18] = 0)设置SPI控制器为主机
 2. 设置CLKP位(SPI_CNTRL[11] = 0)使得总线时钟空闲位低状态。
 3. 通过设置TX_NEG (SPI_CNTRL[2] = 1)位选择数据在串行时钟的下降沿传输。
 4. 通过设置 RX_NEG (SPI_CNTRL[1] = 0) 位选择数据锁存在串行时钟的上升沿。
 5. 通过设置 TX_BIT_LEN 位域 (SPI_CNTRL[7:3] = 0x08) 设定一次数据传输的长度为8-位。
 6. 通过设置 MSB 位 (SPI_CNTRL[10] = 0) 设定为 MSB 优先传输。不必考虑 SP_CYCLE(SPI_CNTRL[15:12])位，因为此情况下非FIFO模式。
- 3) 写一个合适的值到 SPI_SSR 寄存器来对主模式相关配置进行设定
1. 清除自动从选择位 AUTOSS (SPI_SSR[3] = 0)
在从机选择有效电平位 SS_LVL (SPI_SSR[2])=0设定从机选择为低电平触发输出。
 2. 通过设定从机选择寄存器位SSR (SPI_SSR[0]) 设置从机选择信号有效，以激活片外从设备。
- 4) 如果 SPI 主机要发送（写）一个字节的的数据到片外从机设备，则将所要发送到数据写入 SPI_TX0 寄存器。
- 5) 如果 SPI 主机只是要从片外从机设备接收（读）一个字节的的数据，不必管被传输出去的数据是什么，寄存器SPI_TX0 不需要通过软件更新。
- 6) 使能 GO_BUSY 位 (SPI_CNTRL [0] = 1) 开始 SPI 接口的数据传输。
- 7) 等待 SPI 中断发生（如果中断使能位 IE 被使能）或轮询检测 GO_BUSY 位直到被硬件自动清零。
- 8) 从寄存器 RX0 [7:0]中读取接收到的一个字节数据。
- 9) 重复步骤 4) 继续其他数据传输或设置 SSR [0] 为 0 来停止片外从机设备。

以下是基于M058S BSP的SPI数据传输如上面描述的例子

```
/* Set module clock divider. SPI clock rate = HCLK / ((11+1)*2) = 2MHz */
SPI0->DIVIDER = SPI0->DIVIDER & (~SPI_DIVIDER_DIVIDER_Msk) | SPI_DIVIDER_DIV(11);

/* Configure SPI0 as a master, MSB first, clock idle low, falling clock edge Tx, rising edge Rx and 8-bit transaction */
SPI0->CNTRL = SPI_CNTRL_MASTER_MODE | SPI_CNTRL_MSB_FIRST | SPI_CNTRL_CLK_IDLE_LOW |
              SPI_CNTRL_TX_FALLING | SPI_CNTRL_RX_RISING | SPI_CNTRL_TX_BIT_LEN(8);

/* Disable the automatic hardware slave select function. Select the SS pin and configure as low-active. */
SPI0->SSR = SPI_SSR_SW_SS_PIN_LOW;

/* write the first data of source buffer to Tx register of SPI0. */
_SPI_WRITE_TX_BUFFER0(SPI0, SourceData[0]);

/* Start the data transfer */
_SPI_SET_GO(SPI0);

/* Check the GO_BUSY bit */
While(SPI0->CNTRL & SPI_CNTRL_GO_BUSY);

/* Read the received data */
DestData[0] = _SPI_GET_RX0_DATA(SPI0)&0xFF;
...
```

例 2: SPI 控制器作为从机设备，和一片外主机设备相连。外设主设备通过 SPI 接口与 SPI 从机通信。过程如下：

- 数据在串行时钟上升沿锁存
- 数据在串行时钟下降沿传输
- LSB 先传输
- 8位数据长度
- SPICLK 空闲状态为高电平
- 每次发送/接收一个字节
- 从机选择信号为高电平触发

操作流程如下：

- 1) 设置DIVIDER (SPI_DIVIDER[7:0])来设定从机外设时钟频率。该时钟频率必须大于SPI总线时钟频率。
- 2) 给从模式下的SPI_SSR寄存器写一个适当值。通过SS_LVL (SPI_SSR[2] = 1)设置从机片选有效电平，SS_LTRIG (SPI_SSR[4] = 1)选择电平触发方式。
- 3) 通过设置 SPI_CNTRL 寄存器来控制 SPI 从机的行为。
 1. 通过设置 SLAVE 位 (SPI_CNTRL[18] = 1) 将 SPI 控制器设为从机设备
 2. 通过设置 CLKP 位 (SPI_CNTRL[11] = 1) 将串行时钟的空闲状态设为高电平
 3. 通过设置 TX_NEG 位 (SPI_CNTRL[2] = 1) 选择数据在串行时钟的下降沿传输
 4. 通过设置 RX_NEG 位 (SPI_CNTRL[1] = 0) 选择数据在串行时钟的上升沿锁存。
 5. 通过设置 TX_BIT_LEN 位域 (SPI_CNTRL[7:3] = 0x08) 设定一次字传输的长度为8-位。
 6. 设置发送数据个数为一个字节
 7. 通过设置 LSB 位 (SPI_CNTRL[10] = 1) 设定为 LSB 传输优先。不考虑SP_CYCLE (SPI_CNTRL[15:12])位，因此情况下非burst模式
- 4) 如果 SPI 从机要发送(被读)一个字节的的数据到片外主机，则将所要发送的数据写入到SPI_TX0 寄存器。
- 5) 如果 SPI 从机只是要从片外主机接收(被写)一字节数据，用户不必关心什么数据将被传输，不需要软件更新SPI_TX0 寄存器。
- 6) 使能 GO_BUSY 位 (SPI_CNTRL[0] = 1) 来等待片外主机设备的从机选择触发输入和串行时钟输入，以便开始在 SPI 接口的数据传输。
- 7) 等待 SPI 中断发生（如果中断使能位 IE被使能）或轮询检测 GO_BUSY 位直到被硬件自动清零。
- 8) 从寄存器SPI_RX0[7:0] 中读取接收到的一个字节数据。
- 9) 重复步骤 4) 继续其他数据传输或清GO_BUSY位停止数据传输。

以下是基于M058S BSP的SPI数据传输如上面描述的例子

```

/* Configure SPI0 as a high level active device. */
SPI0->SSR = SPI_SSR_SLAVE_HIGH_LEVEL_ACTIVE;

/* Configure SPI0 as a slave, LSB first, clock idle high, falling clock edge Tx, rising edge Rx and 8-bit transaction */
SPI0->CNTRL = SPI_CNTRL_SLAVE_MODE | SPI_CNTRL_LSB_FIRST | SPI_CNTRL_CLK_IDLE_HIGH |
              SPI_CNTRL_TX_FALLING | SPI_CNTRL_RX_RISING | SPI_CNTRL_TX_BIT_LEN(8);
/* write the first data of source buffer to Tx register of SPI0. */
_SPI_WRITE_TX_BUFFER0(SPI0, SourceData[0]);

/* set the GO_BUSY bit of SPI0 */
_SPI_SET_GO(SPI0);

/* Check the GO_BUSY bit */
While(SPI0->CNTRL & SPI_CNTRL_GO_BUSY);

/* Read the received data */
DestData[0] = _SPI_GET_RX0_DATA(SPI0)&0xFF;
...

```

6.12.8 寄存器映射

R: 只读, W: 只写, R/W: 读/写

寄存器	偏移地址	R/W	描述	复位值
SPI 基地址				
SPI0_BA = 0x4003_0000				
SPI_CNTRL	SPI0_BA+0x00	R/W	控制和状态寄存器	0x0500_0004
SPI_DIVIDER	SPI0_BA+0x04	R/W	时钟分频寄存器	0x0000_0000
SPI_SSR	SPI0_BA+0x08	R/W	从机选择寄存器	0x0000_0000
SPI_RX0	SPI0_BA+0x10	R	数据接收寄存器0	0x0000_0000
SPI_TX0	SPI0_BA+0x20	W	数据发送寄存器0	0x0000_0000
SPI_CNTRL2	SPI0_BA+0x3C	R/W	控制和状态寄存器 2	0x0000_0000
SPI_FIFO_CTL	SPI0_BA+0x40	R/W	SPI FIFO控制寄存器	0x2200_0000
SPI_STATUS	SPI0_BA+0x44	R/W	SPI状态寄存器	0x0500_0000

6.12.9 寄存器描述

SPI控制与状态寄存器(SPI_CNTRL)

寄存器	偏移地址	R/W	描述	复位值
SPI_CNTRL	SPIx_BA+0x00	R/W	控制与状态寄存器	0x0500_0004

31	30	29	28	27	26	25	24
保留				TX_FULL	TX_EMPTY	RX_FULL	RX_EMPTY
23	22	21	20	19	18	17	16
保留		FIFO	REORDER		SLAVE	IE	IF
15	14	13	12	11	10	9	8
SP_CYCLE				CLKP	LSB	保留	
7	6	5	4	3	2	1	0
TX_BIT_LEN					TX_NEG	RX_NEG	GO_BUSY

位	描述	
[31:28]	保留	保留.
[27]	TX_FULL	发送 FIFO缓存满标志 (只读) 该位是SPI_STATUS[27]的相互镜像位. 0 = 表示发送数据缓存没满 1 = 表示 FIFO 上的发送数据缓存满了
[26]	TX_EMPTY	发送 FIFO缓存空 标志 (只读) 该位是SPI_STATUS[26]的相互镜像位 0 = 表示发送数据缓存非空 1 = 表示发送数据缓存为空
[25]	RX_FULL	接收 FIFO 缓存满标志 (只读) 该位是SPI_STATUS[25]的相互镜像位 0 = 表示接收数据缓存没满 1 = 表示 FIFO 上的接收数据缓存满了
[24]	RX_EMPTY	接收 FIFO 缓存空标志 (只读) 该位是SPI_STATUS[24]的相互镜像位 0 = 表示接收数据缓存非空 1 = 表示接收数据缓存为空
[23:22]	保留	保留.
[21]	FIFO	FIFO 模式使能位 在主机模式, 如果 FIFO 模式被使能, 在数据被写入发送 FIFO 中之后, GO_BUSY 将会自动被硬件设置为 1. 发送完后, GO_BUSY 将会自动清0. 0 = 禁用 FIFO 模式

		1 = 使能 FIFO 模式 注意:在使能 FIFO 模式前, 其他相关的设定必须先设定。
[20:19]	REORDER	字节重排序功能使能 和字节休眠间隔功能选择 00 = 禁止字节重排序功能。 01 = 使能字节重排序功能, 在每两个字节之间插入一个字节休眠间隔。字节休眠间隔时间取决于 SP_CYCLE 的设置。 10 = 保留。 11 = 保留。 注意:字节重排序仅在 TX_BIT_LEN 被定义为 16 位, 24 位和 32 位时有效。
[18]	SLAVE	从机模式使能位 0 = SPI 控制器被设置为主机模式。 1 = SPI 控制器被设置为从机模式
[17]	IE	中断使能位 0 = 禁用 SPI 中断 1 = 使能 SPI 中断
[16]	IF	中断标志位 0 = 表示没有传输还完成 1 = 表示传输完成 注意: 该位写 1 清零。
[15:12]	SP_CYCLE	休眠间隔(仅主模式) 该四位提供一个在传输过程中连续两个发送/接收事务之间可配置的休眠间隔。休眠间隔是指前一个事务的最后一个时钟边缘和后一个事务的第一个时钟边缘之间的间隔。 默认值是 0x3。休眠间隔长度根据如下公式获得: $(SP_CYCLE[3:0] + 0.5) * SPICLK \text{时钟周期}$ 例如: SP_CYCLE = 0x0 ... 0.5 SPICLK 时钟周期 SP_CYCLE = 0x1 ... 1.5 SPICLK 时钟周期 SP_CYCLE = 0xE ... 14.5 SPICLK 时钟周期 SP_CYCLE = 0xF ... 15.5 SPICLK 时钟周期
[11]	CLKP	时钟极性 0 = SPICLK 空闲状态的电平默认为低。 1 = SPICLK 空闲状态的电平默认为高
[10]	LSB	LSB 优先 0 = MSB 收发优先 1 = LSB 收发优先。
[9:8]	保留	保留。
[7:3]	TX_BIT_LEN	传输位长 该位域指定在一个发送/接收事务中, 多少个数据位将会被传输。最小位长是 8, 最多可以达到 32 位。

		TX_BIT_LEN = 0x08 ... 8 位 TX_BIT_LEN = 0x09 ... 9位 TX_BIT_LEN = 0x1F ... 31位 TX_BIT_LEN = 0x00 ... 32位
[2]	TX_NEG	在下降沿发送 0 = 在 SPICLK 的上升沿发送数据信号改变 1 = 在 SPICLK 的下降沿发送数据信号改变
[1]	RX_NEG	在下降沿接收 0 = 在 SPICLK 的上升沿锁存数据输入信号 1 = 在 SPICLK 的下降沿锁存数据输入信号.
[0]	GO_BUSY	SPI 传输触发位和忙状态 在 FIFO 模式下，该位由硬件控制。软件不能修改该位的值。 非FIFO模式下，数据传输过程中该位保持1。当传输完成，该位被自动清零。 0 = 如果SPI正在传输数据，写零到该位停止数据传输。 1 = 在主机模式下，写 1 到该位开始 SPI 数据传输。在从机模式下，写 1 到该位表示从机已经准备好与主机进行通信 注意 1: 当 FIFO 模式被禁止，在 GO_BUSY 位写 1前，所有配置必须事先被设定好。 2: 从机模式下，如果 FIFO 模式被禁止，在整个数据传输过程中，如果SPI时钟保持空闲状态，从机片选信号无效后，GO_BUSY仍不会被清零。

SPI分频寄存器(SPI DIVIDER)

寄存器	偏移地址	R/W	描述	复位值
SPI_DIVIDER	SPIx_BA+0x04	R/W	时钟分频寄存器	0x0000_0000

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

位	描述	
[31:16]	保留	保留.
[15:0]	DIVIDER	时钟分频器 仅DIVIDER[7:0]有效。该域的值是决定SPI外设时钟频率，f_{spi}和SPICLK输出管脚上的总线时钟频率。该频率可根据下列公式获得： 如果BCn, SPI_CNTRL2[31], 为 0 $f_{spi} = \frac{f_{SPI_clock_src}}{(DIVIDER + 1) * 2}$ 否则如果BCn置1, $f_{spi} = \frac{f_{SPI_clock_src}}{(DIVIDER + 1)}$ 这里 $f_{SPI_clock_src}$ 是 SPI 引擎时钟源，在 CLKSEL1 中定义

SPI从机选择寄存器(SPI SSR)

寄存器	偏移地址	R/W	描述	复位值
SPI_SSR	SPIx_BA+0x08	R/W	从机选择寄存器	0x0000_0000

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

位	描述	
[31:6]	保留	保留。
[5]	LTRIG_FLAG	电平触发标志 当 SS_LTRIG 位在从机模式下置位，读该位的值可用来表示一次传输完成后，接收到的数据位数是否达到要求。 0 = 一次传输的数据长度或位长度不满足指定的要求 1 = 传输的字节个数和位长度满足TX_NUM 和 TX_BIT_LEN.定义的要求。TX_NUM的设置M058S中无效，仅TX_BIT_LEN有效。 注意: 该位只读。仅在从机模式有效。
[4]	SS_LTRIG	从机选择电平触发使能 (仅从模式) 0 = 输入从机选择信号是边沿触发，该值为默认值。根据 SS_LVL 来决定是下降沿/上升沿触发。 1 = 从机选择信号将是电平触发。根据 SS_LVL 来决定是高电平/低电平触发。
[3]	AUTOSS	自动从机选择使能 (仅主模式) 0 = 如果该位清零，从机选择信号将由SSR位的设定值来决定激活或失效。 1 = 如果该位被置位，SPISSx信号自动产生。这表示当通过设置 GO_BUSY 位开始发送/接收时，从机选择信号将由 SPI 控制器设为有效状态，而当每次发送/接收结束时，从机选择信号又会设为无效状态。
[2]	SS_LVL	从机选择触发电平 定义从机选择信号 (SPISSx) 的有效状态。 0 = 从机选择信号SPISSx在低电平/下降沿有效 1 = 从机选择信号SPISSx在高电平/上升沿有效
[1]	保留	保留。
[0]	SSR	从机选择控制位 (仅主模式) 如果 AUTOSS 位被清零 0 = 设置SPISSx为非激活状态

		1= 设置SPISSx为激活状态 如果AUTOSS位被置1， 0 = 保持SPISSx为非激活状态。 1 = 将会使SPISSx线在发送/接收的时间内被自动驱动到激活状态，而其他时间为非激活状态。SPISSx的激活状态类型由 SS_LVL 指定。
--	--	---

SPI数据接收寄存器(SPI_RX)

寄存器	偏移地址	R/W	描述	复位值
SPI_RX0	SPIx_BA+0x10	R	数据接收寄存器0	0x0000_0000

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

位	描述
[31:0]	数据接收寄存器 数据接收寄存器保存从SPI数据输入管脚上次接收到的数据。有效位长度依据SPI_CNTRL寄存器的设置。 例如。如果TX_BIT_LEN是0x08，RX0[7:0]就是接收到的数据。其它位中的数据为未知数。数据接收寄存器为只读寄存器。

SPI数据发送寄存器(SPI_TX)

寄存器	偏移地址	R/W	描述	复位值
SPI_TX0	SPIx_BA+0x20	W	数据发送寄存器0	0x0000_0000

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

位	描述	
[31:0]	TX	数据发送寄存器 数据发送寄存器内存储下一次被发送的数据。数据的有效长度依据 SPI_CNTRL 寄存器内定义的长度决定。 例如，如果 TX_BIT_LEN 为 0x08，则 TX0[7:0] 位将会被发送。

SPI控制和状态寄存器2 (SPI_CNTRL2)

寄存器	偏移地址	R/W	描述	复位值
SPI_CNTRL2	SPIx_BA+0x3C	R/W	控制和状态寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
BCn	保留						
23	22	21	20	19	18	17	16
保留							SS_INT_OPT
15	14	13	12	11	10	9	8
保留				SLV_START_INTSTS	SSTA_INTEN	SLV_ABORT	NOSLVSEL
7	6	5	4	3	2	1	0
保留							

位	描述	
[31]	BCn	SPI 引擎时钟向后兼容选项 0 = 时钟配置与M05xxBN向后兼容。 1 = 时钟配置与M05xxBN不向后兼容。 详情请参考SPI_DIVIDER 寄存器的描述
[30:17]	保留	保留。
[16]	SS_INT_OPT	从机选择无效中断选项 该设置仅在SPI控制器配置为电平触发从机设备时有效。 0 = 当从机选择信号变为无效电平时，IF 位不会被置 1 1 = 当从机选择信号变为无效电平时，IF 位会被置 1
[15:12]	Reserved	保留。
[11]	SLV_START_INTSTS	从机 3-线模式开始中断状态 该位用来表示在3线模式下，传输是否已经开始。它是SPI_STATUS[11]的相互镜像位。 0 = 自从 SSTA_INTEN 置 1，从机没有检测到任何 SPI 时钟。 1 = 在3线模式下，传输已经开始。该位在传输完成后自动清零或写 1 清位零。
[10]	SSTA_INTEN	从机3-线模式开始中断使能位 在3线从机模式下当传输开始时，该位用于使能中断。在传输开始后，用户定义的时间过后，如果没有传输完成的中断，则用户可以置位 SLV_ABORT 位强制完成传输。 0 =禁用传输开始中断。 1 =使能传输开始中断。该位在传输完成后清0或 SLV_START_INTSTS 位被清除后清0。

[9]	SLV_ABORT	从机 3-线模式终止控制 在正常操作中，当接收到数据符合 TX_BIT_LEN 和TX_NUM的要求位的值时，会有中断事件。在M058S中TX_NUM的设置是无效的，只需考虑对TX_BIT_LEN的设置。 在3线从机模式下，如果接收到的位数少于要求的值，而且在一次传输的时间后没有更多的串口时钟输入时，用户可以设定该位来强制完成当前的传输，然后用户就可以收到一个传输完成的中断事件。 注意: 软件设置该位为 1 后，该位会被硬件自动清 0。
[8]	NOSLVSEL	从机 3-线模式使能位 该位用于忽略从机模式下的从机选择信号。当该位置1时，SPI控制器可以工作于3-线接口，包括SPICLK, SPI_MISO, 和SPI_MOSI。 0 = 4-线双向接口。 1 = SPI控制器在从模式下为3-线双向接口。当该位被置1，GO_BUSY置1后，控制器可以收/发数据。 注意: 在从机 3-线模式，SS_LTRIG 位 (SPI_SSR[4])会被自动置 1。
[7:0]	保留	保留。

SPI FIFO 控制寄存器 (SPI FIFO CTL)

寄存器	偏移地址	R/W	描述	复位值
SPI_FIFO_CTL	SPiX_BA+0x40	R/W	SPI FIFO 控制寄存器	0x2200_0000

31	30	29	28	27	26	25	24
保留		TX_THRESHOLD		保留		RX_THRESHOLD	
23	22	21	20	19	18	17	16
保留		TIMEOUT_INTEN	保留				
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留	RXOV_INTEN	保留		TX_INTEN	RX_INTEN	TX_CLR	RX_CLR

位	描述	
[31:30]	保留	保留.
[29:28]	TX_THRESHOLD	发送 FIFO 阈值 如果发送FIFO缓存的有效数据个数小于或者等于TX_THRESHOLD, TX_INTSTS 将会被设置为 1, 否则TX_INTSTS 将会被设置为 0。
[27:26]	保留	保留.
[25:24]	RX_THRESHOLD	接收 FIFO 阈值 如果接收FIFO缓存有效数据个数大于RX_THRESHOLD, RX_INTSTS 将会被设置为 1, 否则RX_INTSTS 将会被设置为 0。
[23:22]	保留	保留.
[21]	TIMEOUT_INTEN	接收 FIFO 超时中断使能 0 = 超时中断禁止 1 = 超时中断使能
[20:7]	保留	保留.
[6]	RXOV_INTEN	接收 FIFO 溢出中断使能位 0 = 接收 FIFO溢出中断禁止 1 = 接收 FIFO 溢出中断使能
[5:4]	保留	保留.
[3]	TX_INTEN	发送阈值中断使能位 0 = TX阈值中断禁止 1 = TX阈值中断使能

[2]	RX_INTEN	接收阈值中断使能 0 = RX 阈值中断禁止 1 = RX 阈值中断使能
[1]	TX_CLR	清发送FIFO 缓存 0 = 无效 1 = 清发送FIFO 缓存。TX_FULL标志会被清0并且TX_EMPTY 标志会被置1。通过软件写1到该位且发送FIFO被清除后，硬件会自动将它清0。
[0]	RX_CLR	清 FIFO 缓存 0 = 无效 1 = 清接收 FIFO 缓存。RX_FULL标志会被清0并且TX_EMPTY 标志会被置1。通过软件写1到该位且接收FIFO被清除后，硬件会自动将它清0。

SPI状态寄存器(SPI STATUS)

寄存器	偏移地址	R/W	描述	复位值
SPI_STATUS	SPIx_BA+0x44	R/W	SPI 状态寄存器	0x0500_0000

31	30	29	28	27	26	25	24
TX_FIFO_COUNT				TX_FULL	TX_EMPTY	RX_FULL	RX_EMPTY
23	22	21	20	19	18	17	16
保留			TIMEOUT	保留			IF
15	14	13	12	11	10	9	8
RX_FIFO_COUNT				SLV_START_INTSTS	保留		
7	6	5	4	3	2	1	0
保留			TX_INTSTS	保留	RX_OVERRUN	保留	RX_INTSTS

位	描述	
[31:28]	TX_FIFO_COUNT	发送 FIFO 数据计数 (只读) 该位域表明发送FIFO缓存的有效数据计数。
[27]	TX_FULL	发送 FIFO 缓存满标志 (只读) SPI_CNTRL[27]的相互镜像位。 0 = 发送 FIFO 缓存没满 1 = 发送 FIFO 缓存满
[26]	TX_EMPTY	发送 FIFO 缓存空标志 (只读) SPI_CNTRL[26] 的相互镜像位。 0 = 发送 FIFO 缓存非空 1 = 发送 FIFO 缓存空
[25]	RX_FULL	接收 FIFO 缓存满标志(只读) SPI_CNTRL[25] 的相互镜像位。 0 = 接收 FIFO 缓存不满 1 = 接收 FIFO 缓存满
[24]	RX_EMPTY	接收 FIFO 缓存空标志 (只读) SPI_CNTRL[24] 的相互镜像位。 0 = 接收 FIFO 缓存非空 1 = 接收 FIFO 缓存为空。
[23:21]	保留	保留。
[20]	TIMEOUT	超时中断标志 0 = 没有收到 FIFO 超时事件。 1 = 接收 FIFO 缓存非空且主机模式下超过64个SPI时钟周期或从机模式下超过

		576个SPI引擎时钟周期没有读操作。当接收到的FIFO缓存被软件读取时，超时状态会自动清0。 注意： 向该位写1清0
[19:17]	保留	保留。
[16]	IF	SPI 单位传输中断标志 SPI_CNTRL[16]的相互镜像位。 0 = 没有传输完成 1 = SPI 控制器已经完成一个单元传输。 注意： 向该位写1清0。
[15:12]	RX_FIFO_COUNT	接收 FIFO 数据计数 (只读) 该域表明接收 FIFO 缓存有效数据计数。
[11]	SLV_START_INTSTS	从机开始中断状态 该位用来表明在从机3线模式下，是否已经开始了一次传输。是 SPI_CNTRL2[11]的相互镜像位。 0 = 传输未开始。 1 = 在从机3线模式下，已经开始一次传输。当一次传输结束，或者写1到该位会清除该位。
[10:5]	保留	保留。
[4]	TX_INTSTS	发送 FIFO 阈值中断状态 (只读) 0 =发送FIFO缓存的有效数据计数大于TX_THRESHOLD的设定值。 1 =发送FIFO缓存的有效数据计数少于或等于TX_THRESHOLD的设定值。 注意： 如果 TX_INTEN = 1 且 TX_INTSTS = 1，SPI 控制器会产生一个SPI中断请求。
[3]	保留	保留。
[2]	RX_OVERRUN	接收 FIFO 溢出 状态 当接收 FIFO缓存满，接下来的数据将会丢掉，该位会置1。 注意： 向该位写1清0
[1]	保留	保留。
[0]	RX_INTSTS	接收 FIFO 阈值中断状态(只读) 0 = 接收FIFO缓存的有效数据计数小于或等于RX_THRESHOLD的设定值。 1 = 接收FIFO缓存的有效数据计数大于RX_THRESHOLD的设定值。 注意： 如果 RX_INTEN = 1 且 RX_INTSTS = 1，SPI 控制器会产生一个SPI中断请求

6.13 模数转换 (ADC)

6.13.1 概述

NuMicro™ M058S系列包含一个12位8通道逐次逼近型模数转换（SAR A/D）。A/D转换支持四种操作模式：单次，突发，单周期扫描及连续扫描模式。A/D转换可由软件，外部引脚(STADC/P3.2)或PWM触发

6.13.2 特性

- 模拟输入电压范围：0 ~ AV_{DD}
- 12位分辨率10位有效保证
- 多达8路单端模拟输入通道或者4路差分模拟输入通道
- ADC外设频率最高可达16MHz
- 高达760k SPS的采样率
- 四种操作模式：
 - ◆ 单次模式：在特定的通道上执行一次A/D转换
 - ◆ 突发模式：A/D转换器在特定的一个通道上连续采样转换，并将结果保存在FIFO中
 - ◆ 单周期扫描模式：A/D转换器在所有使能的信道上，按照信道号码从小到大，执行一次转换
 - ◆ 连续扫描模式：A/D转换器持续执行单周期扫描直到软件停止A/D转换.
- A/D转换触发条件：
 - ◆ 软件写1到ADST位
 - ◆ 外部管脚(STADC)
 - ◆ PWM触发，可选择延时周期
- 每个通道的转换结果都存储在对应的数据寄存器中，并且带有valid/overrun标志
- 转换结果可与指定的值进行比较。当转换值和指定比较寄存器中的设定值相同时，用户还可以选择是否产生一个中断请求
- 通道7支持3路输入源：外部模拟电压，内部带隙电压及内部温度传感器

6.13.3 框图

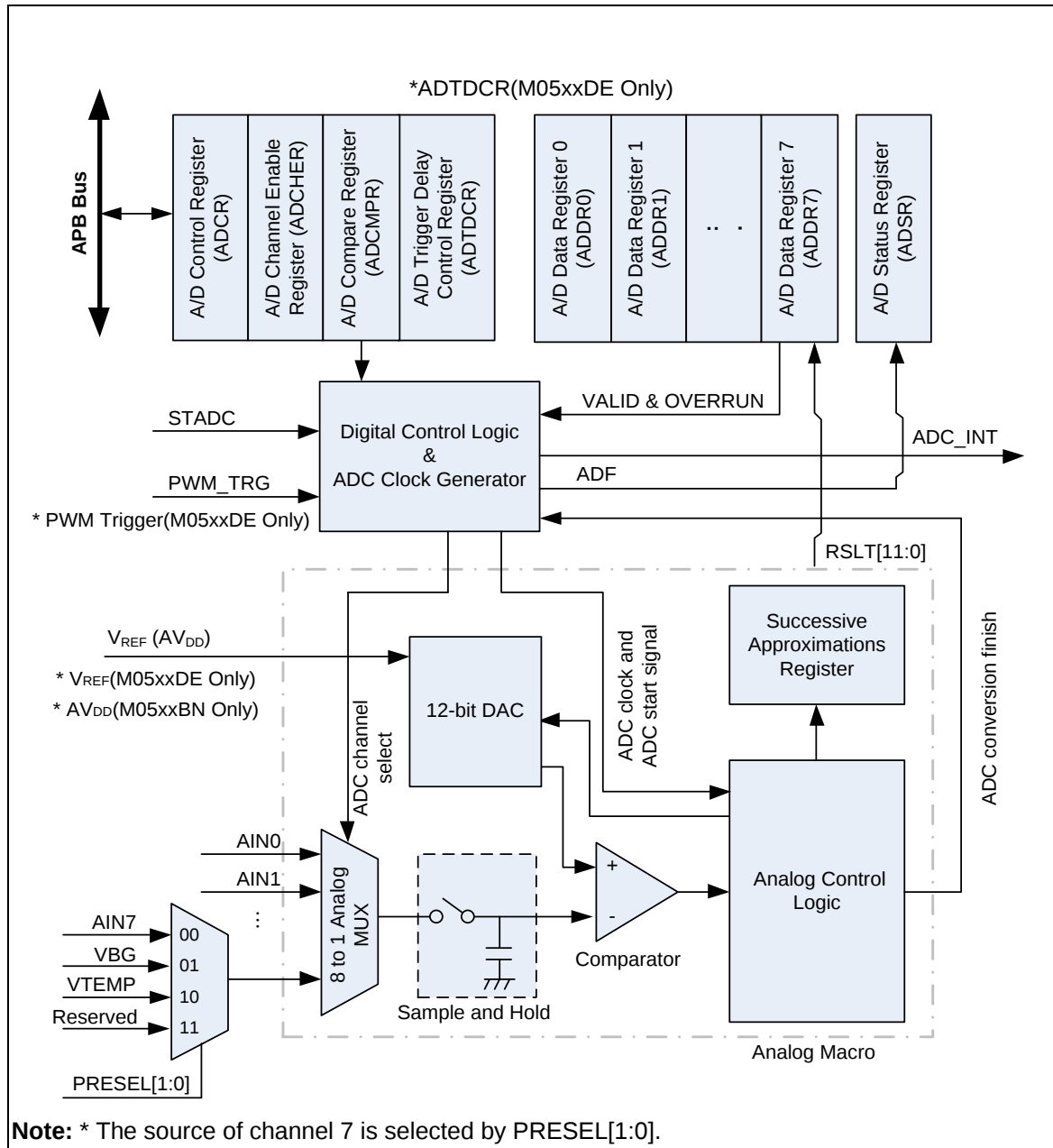


图 6.13-1 AD 控制器框图

6.13.4 基本配置

ADC引脚功能是在P1_MFP寄存器中配置的，建议将模拟信号输入管脚禁止数字输入以防止漏电流，可以通过设置P1_OFFD寄存器禁止数字输入

ADC 模块控制器的时钟源是由ADC_EN(CLK_APBCLK[28])控制使能的。ADC外设时钟源通过CLKSEL1[3:2]选择，CLKDIV[23:16]设置预分频

6.13.5 功能描述

A/D转换器采用逐次逼近转换方式，转换结果为12位数据。ADC有四种工作模式：单一模式，突发模式，单周期扫描模式和连续扫描模式。当需要改变转换模式或模拟输入通道时，为防止错误操作，软件必须先清ADST (ADCR[11]) 为0

6.13.5.1 ADC时钟发生器

最大采样率可达760k SPS。ADC有4个时钟源，通过ADC_S(CLKSEL1[3:2]) 进行选择，ADC时钟频率按如下公式进行8位预分频：

ADC时钟频率 = (ADC 时钟源频率) / (ADC_N+1);8位的ADC_N位于寄存器CLKDIV[23:16]

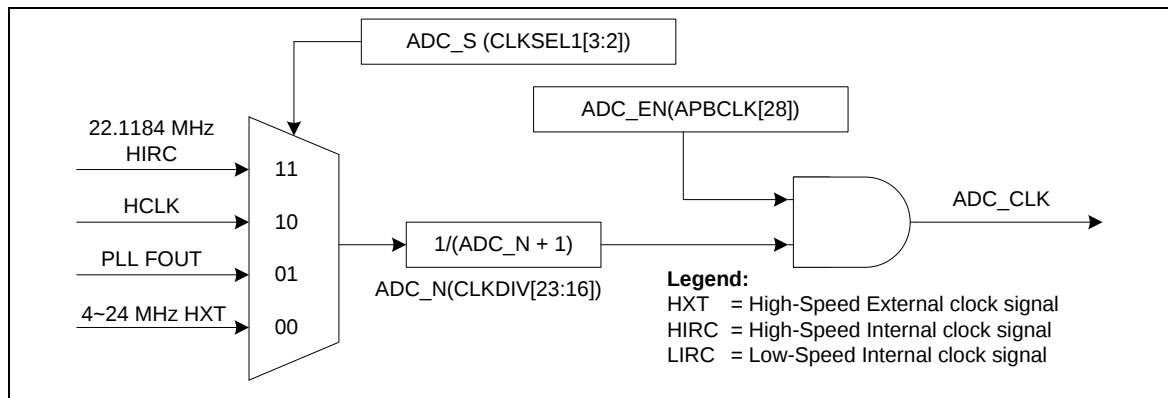


图 6.13-2 ADC 时钟控制

6.13.5.2 单一模式

在单一模式下，A/D转换器仅对指定的一个通道进行一次转换。操作流程为：

1. 当ADCR.ADST位被软件或外部触发置1时，A/D转换开始。
2. 当A/D转换完成，转换结果将存储到与通道对应的A/D数据寄存器中
3. A/D转换完成后，ADSR.ADF位会被置1，如果此时ADCR.ADIE=1,则产生ADC中断。
4. 在A/D转换期间，ADST位一直保持为1。当A/D转换结束，ADST位会自动清0，A/D转换器进入IDLE状态。

注1：如果在单一模式下，如果软件使能了多个通道，则编号最小的通道将会被选中进行转换，其他通道将被忽略

注2：当ADC正在转换时ADST位被清零，BUSY位将被立即清零，但是ADC不会完成当前的转换，直接进入IDLE状态

通过ADC驱动，用户可以调用下面程序完成单周期ADC操作

```

/* Set the ADC operation mode as Single, input mode as single-end and enable the ADC converter */
ADC->ADCR = (ADC_ADCR_ADMD_SINGLE
              | ADC_ADCR_DIFFEN_SINGLE_END
              | ADC_ADCR_ADEN_CONVERTER_ENABLE);

/* Enable analog input channel 2 */
_ADC_SET_CHANNEL(1<<2);

/* Clear the A/D interrupt flag for safe */
ADC->ADSR = ADC_ADSR_ADF_Msk;

/* Enable the ADC interrupt */
_ADC_ENABLE_ADC_INT();
NVIC_EnableIRQ(ADC_IRQn);

/* Start A/D conversion */
_ADC_START_CONVERT();
    
```

下图是ADC单一模式的时序图

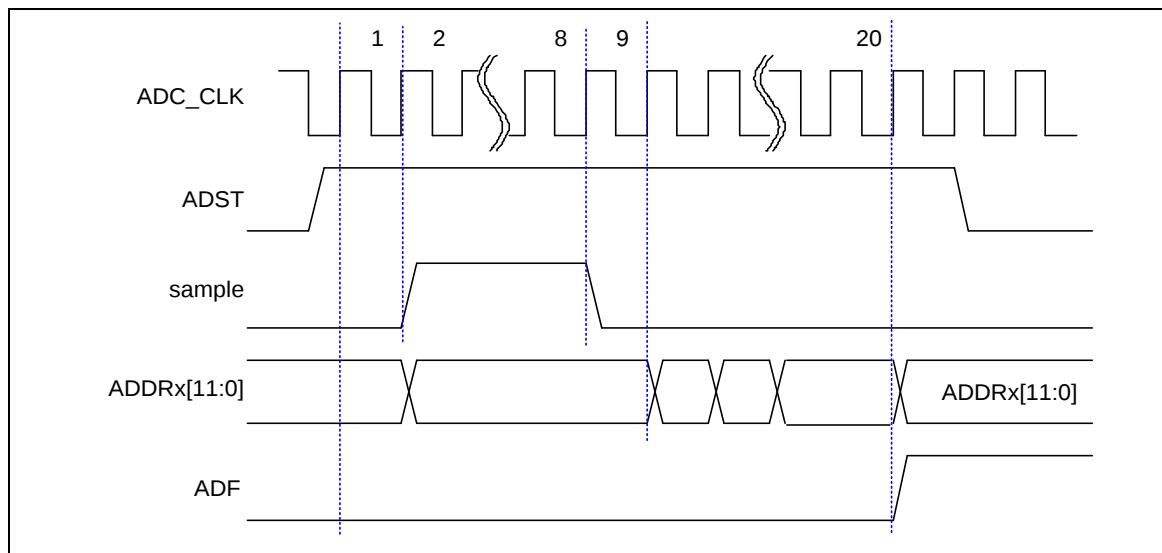


图 6.13-3 单一模式转换时序图

6.13.5.3 突发模式

在突发模式下，A/D转换特定的单个通道数据，并将结果顺序存储到FIFO（最多8个采样数据）中，操作流程如下：

1. 当ADCR.ADST位被软件或外部触发置1，A/D在最低通道开始转换
2. 特定通道转换结束后，结果被顺序传送到FIFO中，只能通过A/D数据寄存器0读取
3. FIFO中采样数据超过4个时，ADSR.ADF被置1，此时如果ADIE位也为1，则会在完成当前A/D转换后产生中断
4. 重复步骤2，3直到ADST位被清零，然后ADC不会等待当前转换结束而马上进入IDLE状态

注1：在突发模式下如果软件使能了不止一个通道，只有最低编号的通道被转换，其他通道被忽略

通过Nuvoton驱动，用户可以调用下面程序完成ADC突发模式操作

```
/* Set the ADC operation mode as Burst, input mode as single-end and enable the ADC converter */
ADC->ADCR = (ADC_ADCR_ADMD_BURST
              | ADC_ADCR_DIFFEN_SINGLE_END
              | ADC_ADCR_ADEN_CONVERTER_ENABLE);
/* Enable analog input channel 2 */
_ADC_SET_CHANNEL(1<<2);

/* Clear the A/D interrupt flag for safe */
ADC->ADSR = ADC_ADSR_ADF_Msk;

/* Enable the ADC interrupt */
```

```

_ADC_ENABLE_ADC_INT();
NVIC_EnableIRQ(ADC_IRQn);

/* start A/D conversion */
_ADC_START_CONVERT();

```

6.13.5.4 单周期扫描模式

在单周期扫描模式下，A/D转换器会对所有指定的通道进行一次采样和转换，通道转换顺序为编号最小的通道到编号最大的通道依次开始，操作流程如下：

1. 当 ADCR.ADST 位被软件或外部触发输入置为 1 时，A/D 将从最小编号的通道开始转换。
2. 每路A/D转换完成后，转换结果将会传输到相应通道的A/D数据寄存器中。
3. 当所有选中的通道都完成转换后，ADSR.ADF位会被置1。若此时ADC中断功能使能，则产生ADC中断。
4. A/D转换结束后，ADST位自动清零，A/D转换器进入IDLE状态。如果在所有使能ADC通道转换完成之前，ADST位被清除为0，则ADC控制器将完成当前转换，并将转换结果存储到当前转换通道的ADDRx中。

通过ADC驱动客户可以调用下面程序完成ADC单周期扫描模式的操作

```

/* Set the ADC operation mode as Single-cycle, input mode as single-end and enable the ADC converter */
ADC->ADCR = (ADC_ADCR_ADMD_SINGLE_CYCLE
              | ADC_ADCR_DIFFEN_SINGLE_END
              | ADC_ADCR_ADEN_CONVERTER_ENABLE);
/* Enable analog input channel 0, 1, 2 and 3 */
_ADC_SET_CHANNEL(0xF);

/* clear the A/D interrupt flag for safe */
ADC->ADSR = ADC_ADSR_ADF_Msk;

/* start A/D conversion */
_ADC_START_CONVERT();

/* Wait conversion done */
_ADC_WAIT_CONVERSION_DONE();

```

下图为使能多个通道（0，2，3，7）的单周期扫描模式时序图

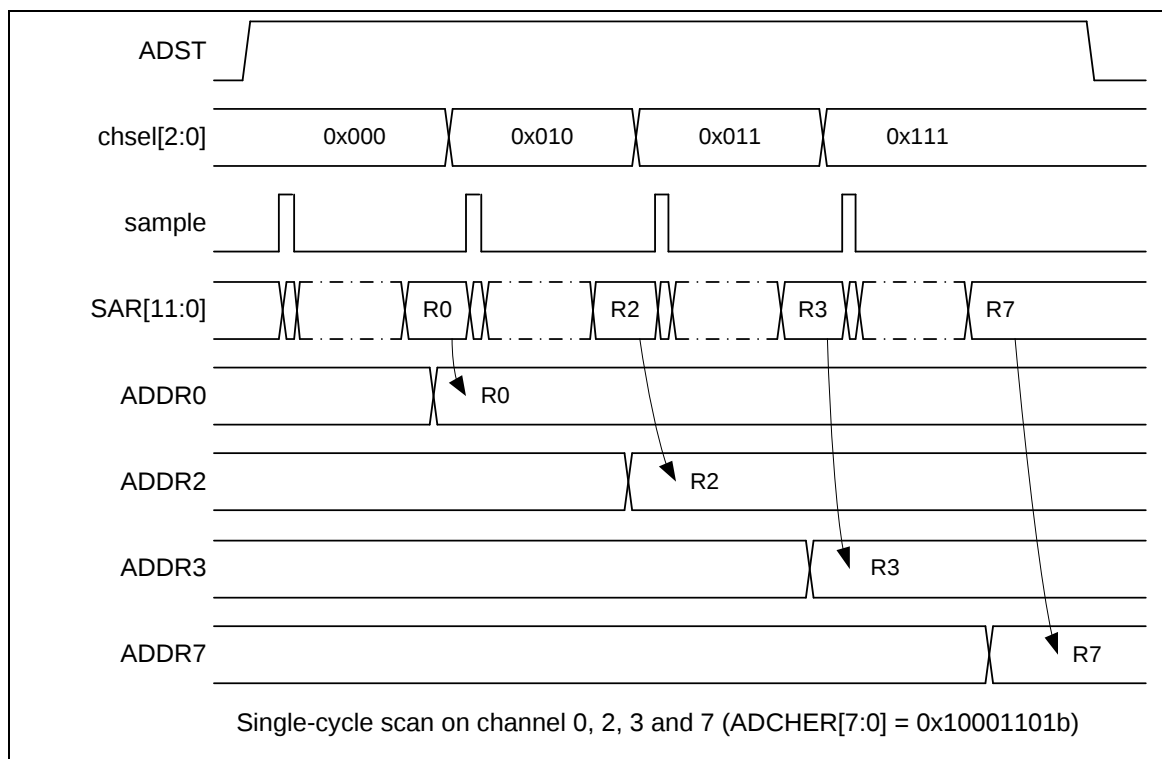


图 6.13-4 单周期扫描模式下使能信道的时序图

6.13.5.5 连续扫描模式

在连续扫描模式下，A/D转换器会对所有由ADCHER.CHEN使能的通道(最多8个)进行连续的循环扫描。操作如下：

1. 当ADCR.ADST被软件或者外部触发输入设置为1时，A/D转换将会从编号最小的通道开始
2. 所有使能通道的A/D转换完成后，A/D转换结果将被装载到相应通道的A/D数据寄存器中。
3. 当所有被使能的通道依次完成一次A/D转换后，ADSR.ADF位将会被置1。如果此时ADC中断功能已经使能，则产生中断。如果软件没有清除ADST位，则又会从最小编号的使能通道开始新的转换。
4. 只要ADST保持为1，就会不断重复步骤2到步骤3。当ADST被清为0时，ADC转换器不会完成当前转换而直接停止

通过ADC驱动，客户调用下面程序可以完成连续扫描模式的ADC转换

```
/* Set the ADC operation mode as Continuous Scan, input mode as single-end and enable the ADC converter */
ADC->ADCR = (ADC_ADCR_ADMD_CONTINUOUS
              | ADC_ADCR_DIFFEN_SINGLE_END
              | ADC_ADCR_ADEN_CONVERTER_ENABLE);

/* Enable analog input channel 0, 1, 2 and 3 */
_ADC_SET_CHANNEL(0xF);
```

```
/* Clear the A/D interrupt flag for safe */
ADC->ADSR = ADC_ADSR_ADF_Msk;

/* Start A/D conversion */
_ADC_START_CONVERT();

/* Wait conversion done */
_ADC_WAIT_CONVERSION_DONE();

/* Stop A/D conversion */
_ADC_STOP_CONVERT();
```

下图为使能多个通道（0，2，3，7）的连续扫描模式时序图：

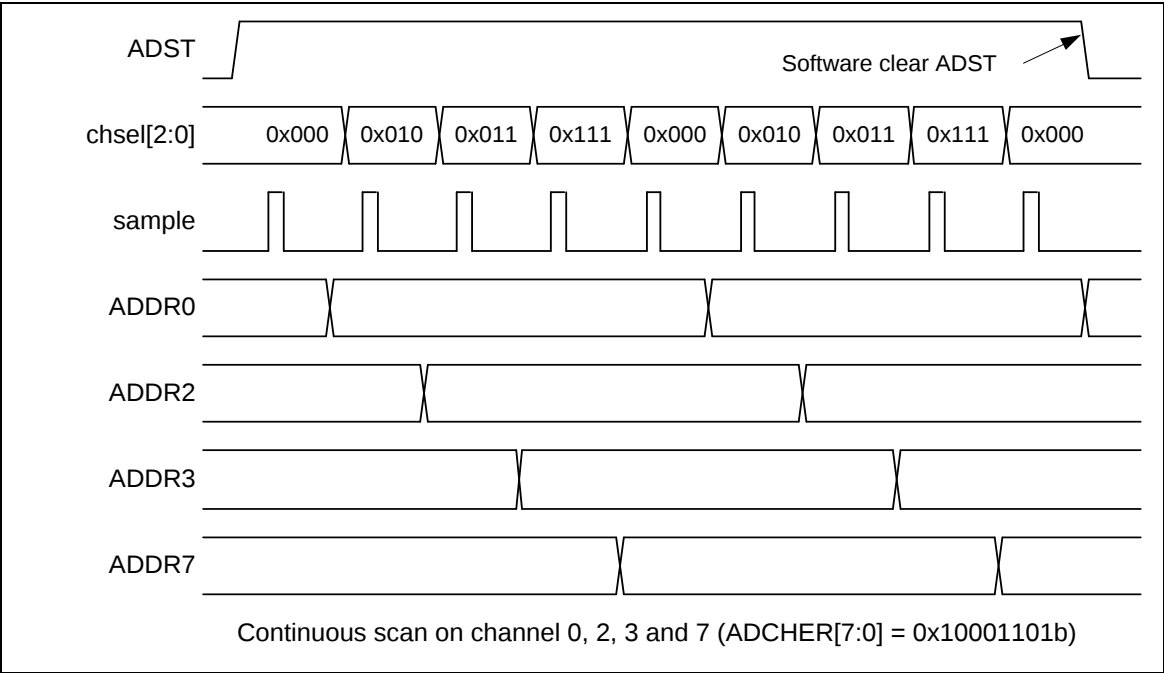


图 6.13-5 使能信道的连续扫描模式时序图

6.13.5.6 外部触发输入采样和A/D转换时间

在单周期扫描模式下，A/D转换可由外部管脚触发转换。当设置ADCR.TRGEN为1用来使能ADC外部触发功能时，设定ADCR.TRGS为00选择STADC管脚做为外部触发输入，再设定ADCR.TRGCOND来设置触发条件是上/下边沿触发还是高/低电平触发。如果选择电平触发，STADC管脚需要保持设定状态至少8个PCLK。ADST将在第9个PCLK时被置为1并且开始进行转换。在电平触发模式状态下，如果外部触发输入维持在有效状态，转换将会持续进行。仅当外部触发条件消失才停止。若选择边沿触发模式，高或低的状态至少需要保持4个PCLK，低于该值的脉冲将被忽略。

注：用户必须在使能ADC时钟至少4个PCLKs后再使能ADC或外部触发功能

6.13.5.7 PWM 触发

在单周期扫描模式下，如果设置ADCR.TRGEN为1并且将ADCR.TRGS设置为11b， PWM就可以作为ADC的触发源。当使能PWM触发ADC功能时，设置PTDT[7:0]可以在PWM触发和ADC开始转换之间加入一个延时

6.13.5.8 比较器监测转换结果功能模式

NuMicro™ M058S 系列的ADC控制器提供两组比较寄存器ADCMPR0和ADCMPR1，用于监控A/D转换控制器（最多支持）2路通道的转换结果值。软件可通过设定CMPCH(ADCMPR0/1[5:3])来选择监控哪路通道，而CMPCOND位 (ADCMPR0/1[2])则用于设置比较条件，如果CMPCOND为0当转换结果小于CMPD(ADCMPR0/1 [27:16])设定值时，内部匹配计数器将会加1；如果CMPCOND为1当转换结果大于等于CMPD(ADCMPR0/1 [27:16])设定值时，内部匹配计数器将会加1。当CMPCH指定的通道转换完成时，比较行为会被自动触发一次。当比较结果和设定值相匹配，比较匹配计数器将加1，否则匹配计数器将会被清0。当计数器的值和设定值(CMPMATCNT (ADCMPR0/1 [11:8])+1) 匹配时， CMPF0/1 位 (ADSR[1]/[2]) 将会置1。如果 CMPIE 位 (ADCMPR0/1 [1])为1，将产生ADC_INT中断请求。使用这个功能可以在扫描模式下，无需软件介入就可以监控外部模拟输入管脚电压变化。具体逻辑框图如下：

注：如果用户使能了差分输入模式，比较结果可以被表示为标准2进制格式（无符号格式）或2's补码格式（有符号格式），软件可以设置DMOF(ADCR[31])选择输出格式

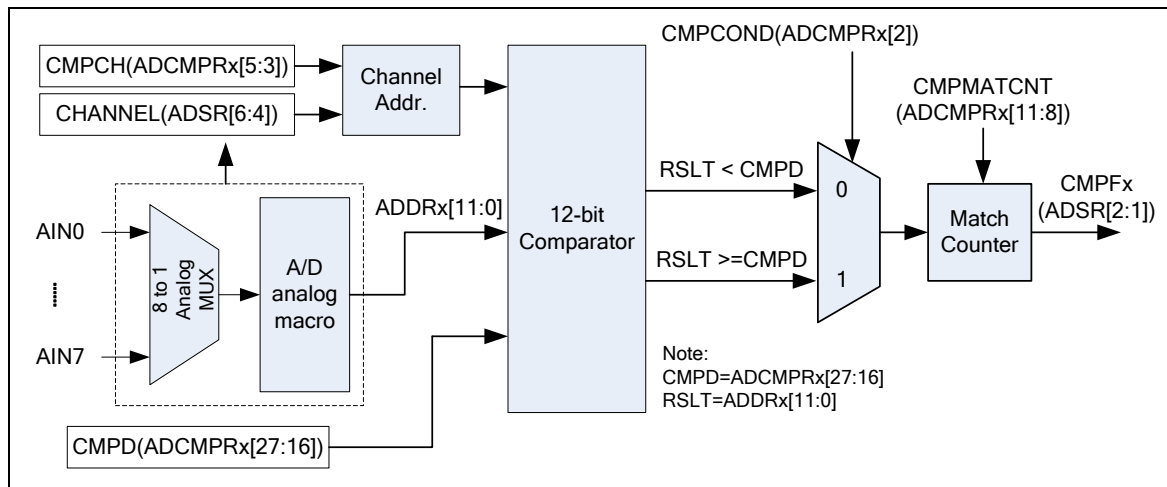


图 6.13-6 A/D 转换结果监控逻辑框图

6.13.5.9 中断源

ADC中断有三个中断源。当某个ADC操作模式结束转换时，A/D转换的结束标志ADF位将被置1。CMPF0 (ADSR[1]) 和CMPF1 (ADSR[2])是比较功能的比较标志。当A/D转换结果同ADCMPR0/1寄存器设定值匹配时，相应的比较标志会被置1。当ADF(ADSR[0])、CMPF0和CMPF1中任意一个标志被置1，且其相应中断使能位ADIE(ADCR[1]) 及CMPIE(ADCMPR0/1[1])置1时，将产生ADC中断。软件可通过清除标志位以撤销该中断请求

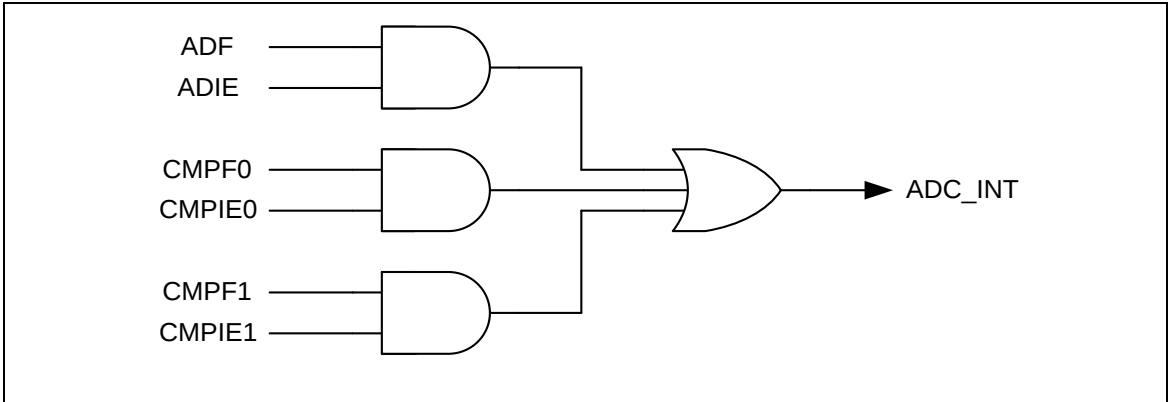


图 6.13-7 A/D 控制器中断

6.13.6 寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位值
ADC 基地址: ADC_BA = 0x400E_0000				
ADDR0	ADC_BA+0x00	R	ADC数据寄存器0	0x0000_0000
ADDR1	ADC_BA+0x04	R	ADC 数据寄存器 1	0x0000_0000
ADDR2	ADC_BA+0x08	R	ADC 数据寄存器 2	0x0000_0000
ADDR3	ADC_BA+0x0C	R	ADC 数据寄存器3	0x0000_0000
ADDR4	ADC_BA+0x10	R	ADC 数据寄存器4	0x0000_0000
ADDR5	ADC_BA+0x14	R	ADC 数据寄存器 5	0x0000_0000
ADDR6	ADC_BA+0x18	R	ADC 数据寄存器 6	0x0000_0000
ADDR7	ADC_BA+0x1C	R	ADC 数据寄存器7	0x0000_0000
ADCR	ADC_BA+0x20	R/W	ADC 控制寄存器	0x0000_0000
ADCHER	ADC_BA+0x24	R/W	ADC通道使能寄存器	0x0000_0000
ADCMPR0	ADC_BA+0x28	R/W	ADC比较寄存器0	0x0000_0000
ADCMPR1	ADC_BA+0x2C	R/W	ADC 比较寄存器1	0x0000_0000
ADSR	ADC_BA+0x30	R/W	ADC 状态寄存器	0x0000_0000
ADTDCCR	ADC_BA+0x44	R/W	ADC触发延时控制寄存器	0x0000_0000

6.13.7 寄存器描述

ADC 数据寄存器 (ADDR0 ~ ADDR7)

寄存器	偏移量	R/W	描述	复位值
ADDR0	ADC_BA+0x00	R	ADC数据寄存器0	0x0000_0000
ADDR1	ADC_BA+0x04	R	ADC数据寄存器1	0x0000_0000
ADDR2	ADC_BA+0x08	R	ADC数据寄存器2	0x0000_0000
ADDR3	ADC_BA+0x0C	R	ADC数据寄存器3	0x0000_0000
ADDR4	ADC_BA+0x10	R	ADC数据寄存器4	0x0000_0000
ADDR5	ADC_BA+0x14	R	ADC数据寄存器5	0x0000_0000
ADDR6	ADC_BA+0x18	R	ADC数据寄存器6	0x0000_0000
ADDR7	ADC_BA+0x1C	R	ADC数据寄存器7	0x0000_0000

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

位	描述	
[31:18]	保留	保留。
[17]	VALID	有效标志 0 = RSLT(ADDRx[15:0], x=0~7)中的数据无效 1 = RSLT(ADDRx[15:0], x=0~7)中的数据有效 当相应的模拟通道转换完成后, 该位置1。读ADDR寄存器后, 该位由硬件自动清除。 该位只读 注: 当ADC正在转换时, 如果用户希望了解这个通道转换数据是否结束, 应该去查询ADSR寄存器的VALID位而不是此位
[16]	OVERRUN	Overrun 标志 (只读) 0 = RSLT (ADDRx[15:0], x=0~7)中的数据是最新的转换结果 1 = RSLT (ADDRx[15:0], x=0~7)中的数据被覆盖过 如果新的转换结果载入到RSLT寄存器之前, 上一次的转换结果还没有被读取, 则OVERRUN 标志将被置1, 之前的转换结果也会丢失。在读ADDR寄存器后, 该

		位由硬件自动清零。 该位只读
[15:0]	RSLT	A/D 转换结果 该域存放ADC的转换结果

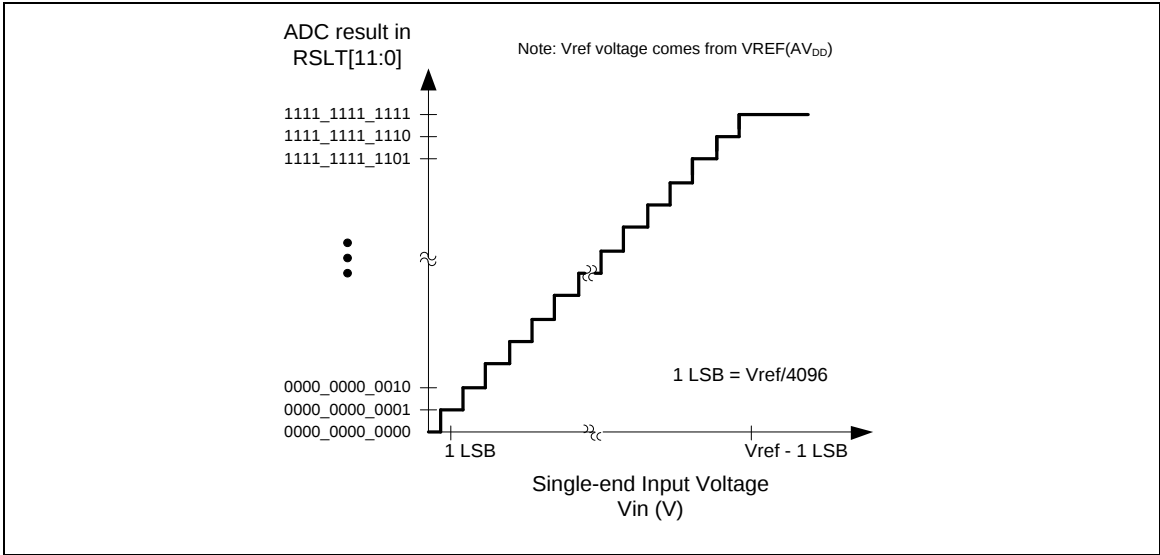


图 6.13-8 ADC 单端输入电压和转换结果图

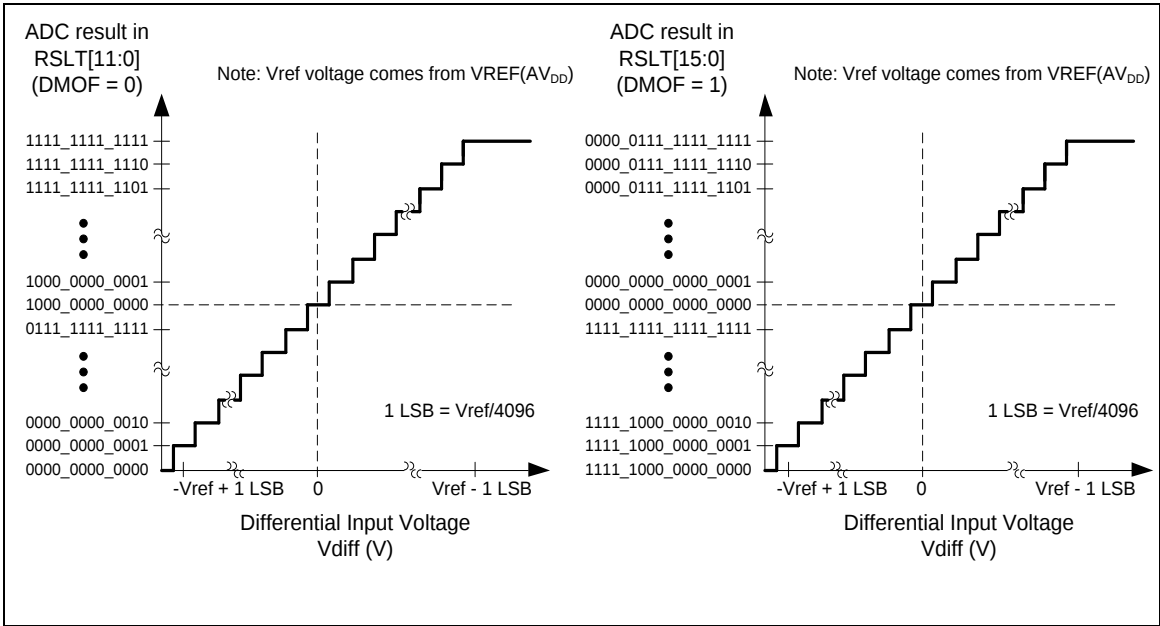


图 6.13-9 ADC 差分输入和转换结果图

ADC 控制寄存器 (ADCR)

寄存器	偏移量	R/W	描述	复位值
ADCR	ADC_BA+0x20	R/W	ADC 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DMOF	保留						
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留				ADST	DIFFEN	保留	TRGEN
7	6	5	4	3	2	1	0
TRGCOND		TRGS		ADMD		ADIE	ADEN

位	描述																		
[31]	DMOF	A/D差分输入模式的输出格式 0 = A/D转换结果存储在ADDRX寄存器的RSLT，格式为无符号格式 1 = A/D转换结果存储在ADDRX寄存器的RSLT，格式为二进制补码格式																	
[30:12]	保留	保留.																	
[11]	ADST	A/D转换开始 0 =转换停止，A/D转换器进入空闲状态 1 =转换开始 . ADST位可以通过三种方式设置为1，分别为：软件设定，PWM触发和外部STADC管脚触发。单一模式和单周期模式下，在转换结束后，ADST将被硬件自动清除。在连续扫描模式和突发模式下，A/D转换将一直进行直到软件向该位写0或芯片复位																	
[10]	DIFFEN	差分输入模式控制 0 =单端模拟输入模式 1 =差分模拟输入模式																	
		<table><tr><th rowspan="2">差分输入对通道</th><th colspan="2">ADC 模拟输入</th></tr><tr><th>V_{plus}</th><th>V_{minus}</th></tr><tr><td>0</td><td>AIN0</td><td>AIN1</td></tr><tr><td>1</td><td>AIN2</td><td>AIN 3</td></tr><tr><td>2</td><td>AIN 4</td><td>AIN 5</td></tr><tr><td>3</td><td>AIN 6</td><td>AIN 7</td></tr></table>	差分输入对通道	ADC 模拟输入		V _{plus}	V _{minus}	0	AIN0	AIN1	1	AIN2	AIN 3	2	AIN 4	AIN 5	3	AIN 6	AIN 7
		差分输入对通道		ADC 模拟输入															
			V _{plus}	V _{minus}															
		0	AIN0	AIN1															
		1	AIN2	AIN 3															
		2	AIN 4	AIN 5															
		3	AIN 6	AIN 7															
差分输入电压(V _{diff}) = V _{plus} - V _{minus} . 其中V _{plus} 是模拟正向输入，V _{minus} 是模拟负向输入。																			

		注： 在差分输入模式下，只需要在ADCHER使能两个相应信道的偶数信道即可。转换结果将放置于相应的使能通道的寄存器里
[9]	保留	保留。
[8]	TRGEN	外部触发使能位 使能或禁止通过外部STADC引脚方式（外部STADC管脚或PWM触发）触发A/D转换 0 = 禁止 1 = 使能 ADC硬件触发功能 只能在单周期扫描模式下支持 在硬件触发模式下，ADST 位 (ADCR[11])可被所选择的硬件触发源置1
[7:6]	TRGCOND	外部触发条件 这2位决定使用外部管脚STADC触发条件为电平触发还是边沿触发。电平触发信号必须保持至少8 PCLKs的稳定状态，而边沿触发信号必须保持至少4 PCLKs的高或低状态 00 = 低电平 01 = 高电平 10 = 下降沿 11 = 上升沿
[5:4]	TRGS	硬件触发源 00 = A/D转换由外部STADC管脚触发开始 11 = A/D转换由PWM触发开始 其它 = 保留 改变TRGS前，软件必须先禁用TRGEN (ADCR[8])和ADST (ADCR[11])
[3:2]	ADMD	A/D转换器工作模式 00 = 单一转换 01 = 突发模式 10 = 单周期扫描 11 = 连续扫描 注1: 当改变工作模式时，软件须先禁止ADST位(ADCR[11])。 注2: 在突发模式时，A/D结果数据总是存储在数据寄存器0
[1]	ADIE	A/D中断使能位 0 = 禁止A/D中断 1 = 使能A/D中断 如果ADIE 位 (ADCR[1])被置1，则A/D转换结束后将产生中断请求
[0]	ADEN	A/D转换器使能位 0 = 禁止 1 = 使能 在开始 A/D 转换前，该位需设置为 1。清除该位为 0 将禁用 A/D 转换器模拟电路从而节省功耗

ADC 通道使能寄存器 (ADCHER)

寄存器	偏移量	R/W	描述	复位值
ADCHER	ADC_BA+0x24	R/W	ADC通道使能寄存器	0x0000_0000

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

位	描述	
[31:10]	保留	保留.
[9:8]	PRESEL[1:0]	模拟输入通道7选择 00 = 外部模拟输入 01 = 内部带隙电压. 10 = 内部温度传感器 11 = 保留 注: 当选择内部带隙电压作为通道7模拟输入时, ADC时钟源要限制在300Khz以下.
[7:0]	CHEN	模拟输入通道使能位 设置CHEN[7:0]用于使能相应的模拟输入通道7~0。如果DIFFEN位(ADCR[10])设置为1, 只有偶数位通道需要使能 0 = 禁止ADC输入通道 1 = 使能ADC输入通道

A/D 比较寄存器 0/1 (ADCMPR0/1)

寄存器	偏移量	R/W	描述	复位值
ADCMPR0	ADC_BA+0x28	R/W	ADC 比较寄存器 0	0x0000_0000
ADCMPR1	ADC_BA+0x2C	R/W	ADC 比较寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
保留				CMPD[11:8]			
23	22	21	20	19	18	17	16
CMPD[7:0]							
15	14	13	12	11	10	9	8
保留				CMPMATCNT			
7	6	5	4	3	2	1	0
保留		CMPCH			CMPCOND	CMPIE	CMPEN

位	描述	
[31:28]	保留	保留。
[27:16]	CMPD	比较数值 此12位数据用于和指定通道的转换结果进行比较。 注：CMPD 中应该写入无符号格式数据
[15:12]	保留	保留。
[11:8]	CMPMATCNT	比较匹配计数 当指定A/D通道的模拟转换值和比较条件CMPCOND (ADCMPR0/1[2])相匹配时，内部计数器将加1，在比较过程中，当内部计数器达到设定值(CMPMATCNT (ADCMPR0/1[11:8]) +1)时，将置CMPF0/1 位(ADSR[1]/[2])为1。
[7:6]	保留	保留。
[5:3]	CMPCH	比较通道选择 000 =选择通道0转换结果进行比较 001 =选择通道1转换结果进行比较 010 =选择通道2转换结果进行比较 011 =选择通道3转换结果进行比较。 100 =选择通道4转换结果进行比较 101 =选择通道5转换结果进行比较。 110 =选择通道6转换结果进行比较 111 =选择通道7转换结果进行比较
[2]	CMPCOND	比较条件 0 = 设置比较条件为：当12位A/D转换结果小于 12位CMPD (ADCMPR0/1[27:16])，内部匹配计数器将加1 1 =设置比较条件为：当12位A/D转换结果大于或等于 12位

		CMPD(ADCMR0/1[27:16]), 内部匹配计数器将加1 注意: 当内部计数器达到(CMPMATCNT (ADCMR0/1[11:8])+1), CMPF0/1 (ADSR[1]/[2])将会被置1
[1]	CMPIE	比较中断使能位 0 = 禁止比较中断 1 =使能比较中断 如果使能了比较功能, 而且比较条件满足CMPCOND(ADCMR0/1[2])和CMPMATCNT (ADCMR0/1[11:8])中的设定, 则CMPF0/1位(ADSR[1]/[2])将会被置1, 同时, 如果CMPIE(ADCMR0/1[1])为1, 将产生比较中断请求
[0]	CMPEN	比较使能位 0 =禁止比较功能 1 =使能比较功能 设置此位为1可使能ADC控制器 将指定通道转换结果载入到ADDR寄存器后与CMPD (ADCMR0/1[27:16])中数据进行比较

A/D 状态寄存器 (ADSR)

寄存器	偏移量	R/W	描述	复位值
ADSR	ADC_BA+0x30	R/W	ADC状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
OVERRUN[7:0]							
15	14	13	12	11	10	9	8
VALID[7:0]							
7	6	5	4	3	2	1	0
保留	CHANNEL			BUSY	CMPF1	CMPF0	ADF

位	描述	
[31:24]	保留	保留.
[23:16]	OVERRUN	Overrun 标志 (只读) 该位为OVERRUN 位 (ADDR0~7[16])的镜像 当ADC在突发模式并且FIFO被overrun了, OVERRUN[7:0]所有位都被置1
[15:8]	VALID	数据有效标志(只读) 该位为VALID 位 (ADDR0~7[17])的镜像 当ADC在突发模式如果有任意转换结果有效, VALID[7:0]所有位置1
[7]	保留	保留.
[6:4]	CHANNEL	当前转换通道 当BUSY(ADSR[3])=1时, 该域反应当前转换通道号。当BUSY=0时, 它表示下一个要转换的通道号 该位只读
[3]	BUSY	忙/空闲 0 = A/D转换器处于空闲状态 1 = A/D转换器处于忙状态 该位为ADST位(ADCR[11])的镜像位。 该位只读.
[2]	CMPF1	比较标志1 当所选择的通道的A/D转换结果和ADCMPR1的设定条件匹配时, 该位置1。该位写1清除 0 = ADDR中的转换结果和ADCMPR1 的设定值不匹配 1 = ADDR中的转换结果和ADCMPR1的设定值匹配
[1]	CMPF0	比较标志 0 当所选择的通道的A/D转换结果和ADCMPR0的设定条件匹配时, 该位置1。该位

		写1清除 0 = ADDR中的转换结果和ADCMPR0 的设定值不匹配 1 = ADDR中的转换结果和ADCMPR0的设定值匹配
[0]	ADF	A/D转换结束标志 该状态标志表明A/D转换结束，软件写1清0 ADF在以下三个条件下被置1： 1. 当在单次模式下 A/D 转换结束 2. 在单周期扫描及连续扫描模式下，所有指定的通道 A/D 转换结束 3. 在突发模式下FIFO中有大于4个采样数据

ADC 触发延时控制寄存器(ADTDCR)

寄存器	偏移量	R/W	描述	复位值
ADTDCR	ADC_BA+0x44	R/W	ADC触发延时控制寄存器	0x0000_0000

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

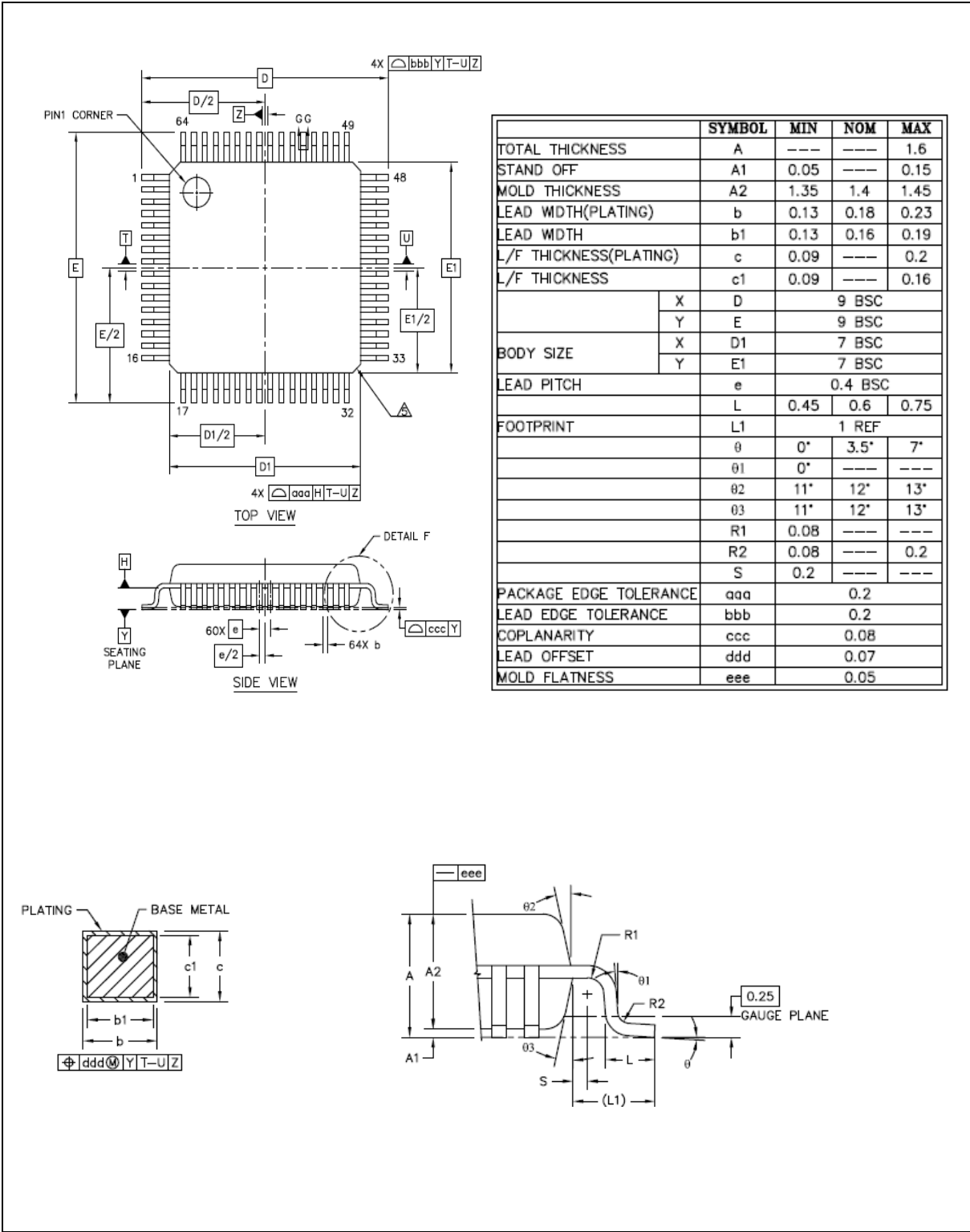
位	描述	
[31:8]	保留	保留.
[7:0]	PTDT	PWM触发延时 PWM触发ADC后此域用来延时采样开始时间，延时为: $(4 * PTDT) * \text{系统时钟}$

7 电器特性

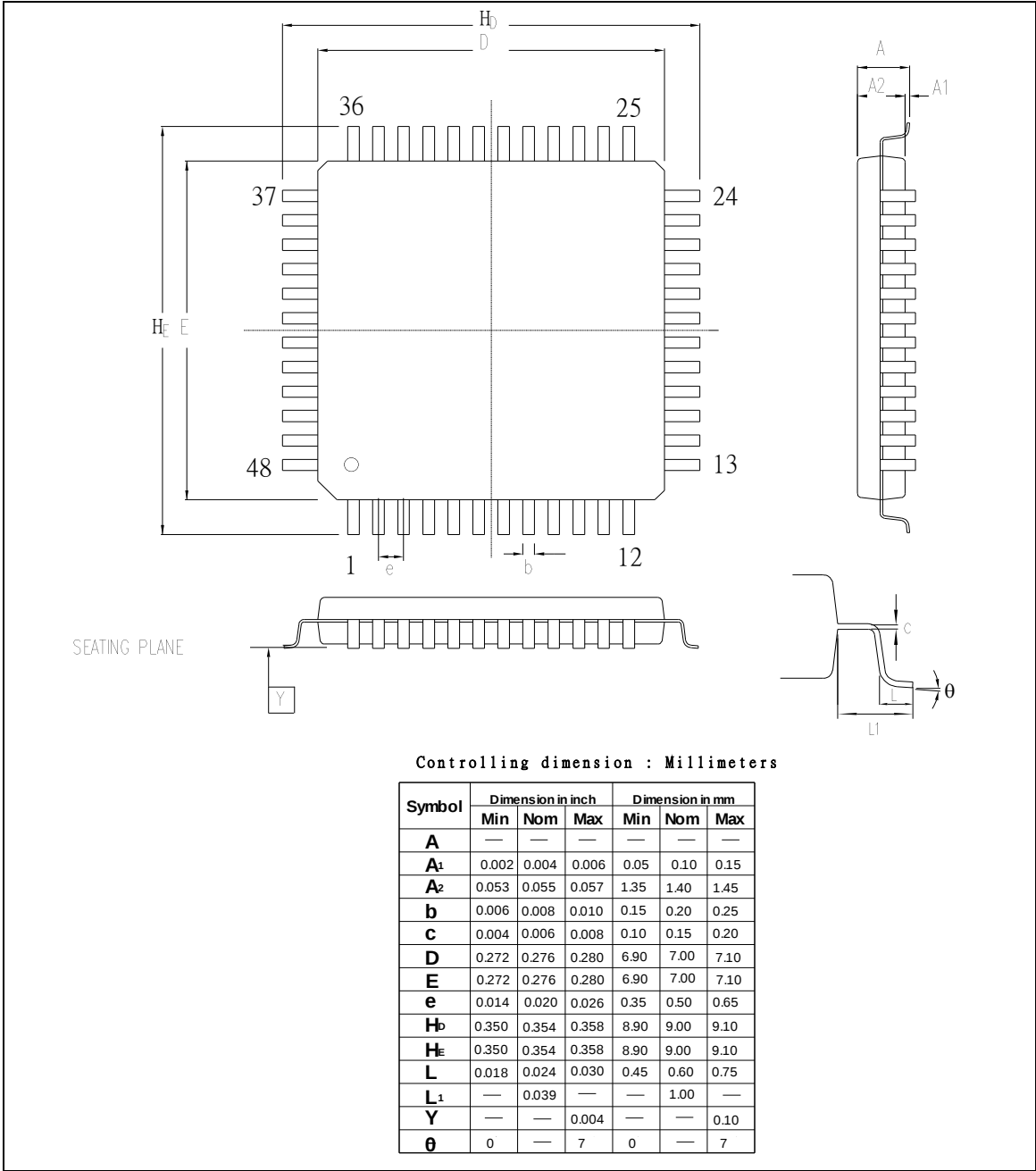
M058S系列的电器特性请参考相关数据手册

8 封装尺寸

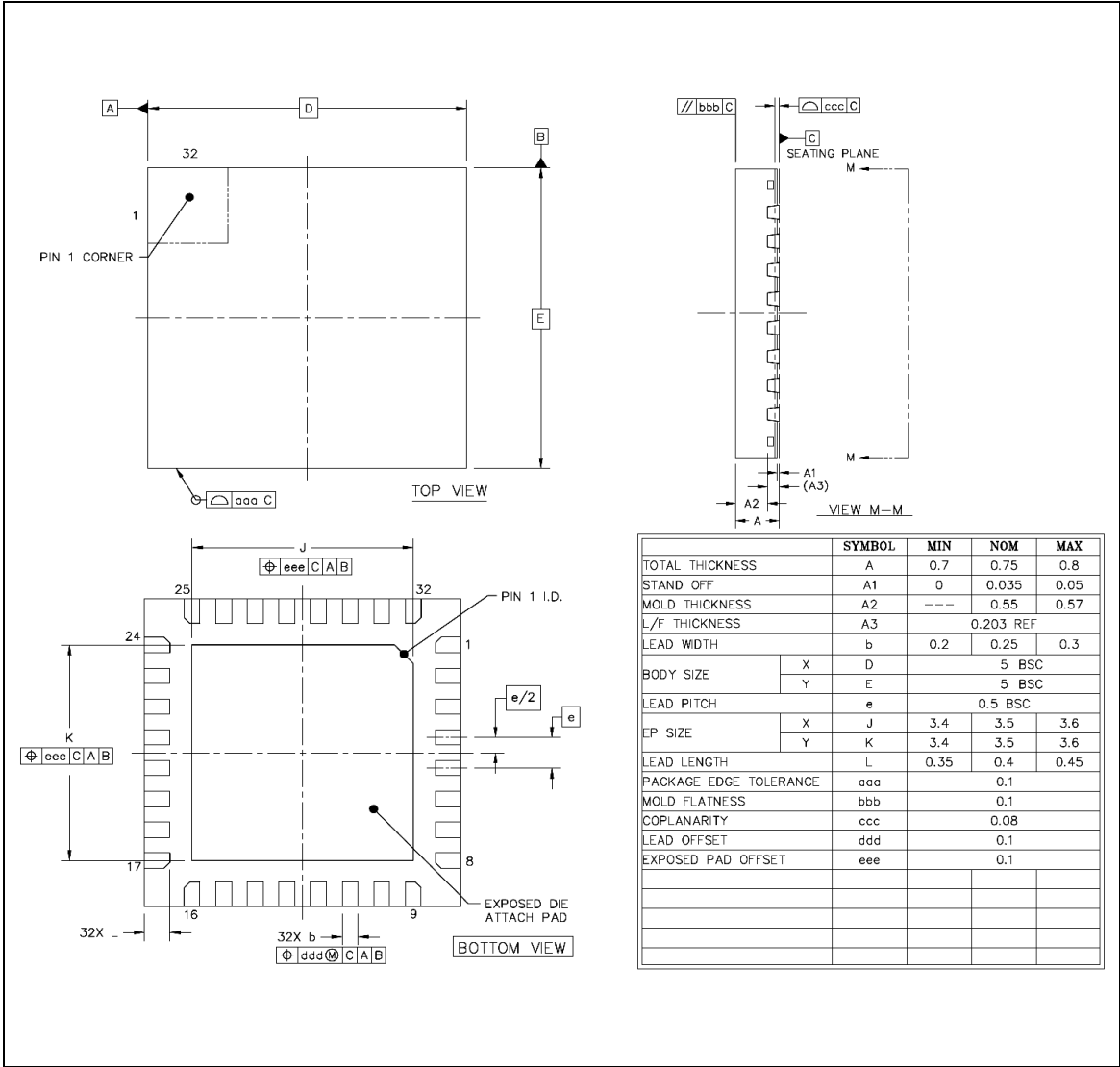
8.1 LQFP-64 (7x7mm, Thickness 1.4 mm, Footprint 2.0mm)



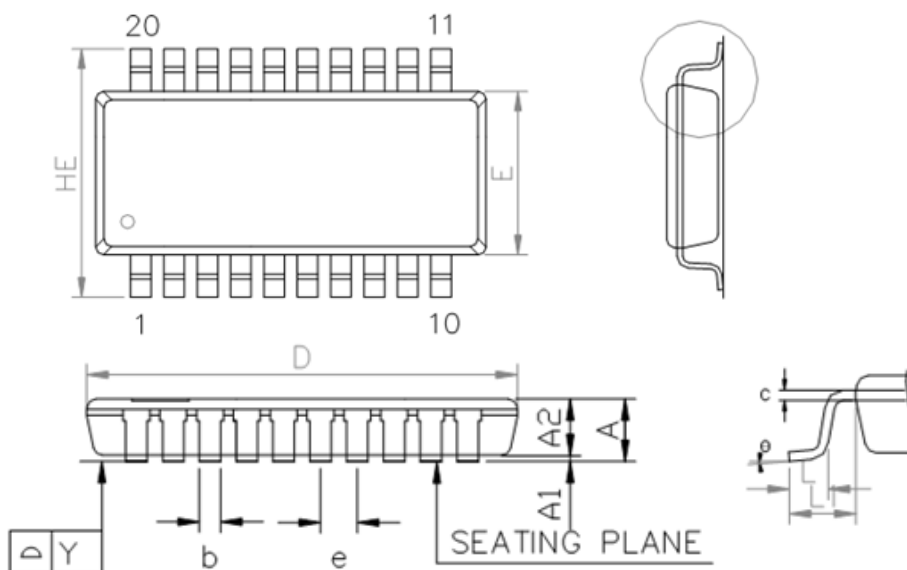
8.2 LQFP-48 (7x7x1.4mm² Footprint 2.0mm)



8.3 QFN-33 (5X5 mm², Thickness 0.8mm, Pitch 0.5 mm)



8.4 TSSOP-20 (4.4x6.5 mm)



SYMBOL	DIMENSION (MM)			DIMENSION (INCH)		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	-	-	1.20	-	-	0.047
A1	0.05	-	0.15	0.002	-	0.006
A2	0.80	0.90	1.05	0.031	0.035	0.041
E	4.30	4.40	4.50	0.169	0.173	0.177
HE	6.40 BSC			0.252 BSC		
D	6.40	6.50	6.60	0.252	0.256	0.260
L	0.50	0.60	0.75	0.020	0.024	0.030
L1	1.00 REF			0.039 REF		
b	0.19	-	0.30	0.007	-	0.012
e	0.65 BSC			0.026 BSC		
c	0.09	-	0.20	0.004	-	0.008
θ	0°	-	8°	0°	-	8°
Y	0.10 BASIC			0.004 BASIC		

9 修订历史

版本	日期	描述
1.00	Apr. 24, 2014	第一版
1.01	Sep. 12, 2014	1. 调整 表 4.1-1 NuMicro™ 的格式。 2. 更新 图 4.1-1 NuMicro™ 3. 改变 Chapter 5 方块图与 Chapter 6 功能描述的顺序。 4. 修正错字与描述不清的语句。
1.02	Nov. 27, 2014	1. 修正 表 4.1-1 NuMicro™.
1.03	Nov. 29, 2017	1. 在 6.4.6 小节增加 CONFIG0[27] "CFGXT1" 与其说明.
1.04	Apr. 30, 2020	1. 在第 4.3 节中添加了有关 ICE_DAT、ICE_CLK 和/RST 引脚的硬件参考设计的说明。

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, “Insecure Usage”.

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer’s risk, and in the event that third parties lay claims to Nuvoton as a result of customer’s Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

*Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.*