ARM Cortex™-M0

32位微控制器

NuMicro M051™ BN 系列
技术参考手册

文件更新日期: 4月10日, 2012
版本 V1.03

目录

1 概述 ····· 11

2 特征 ····· 12

3 方块图 ····· 16

4 选型表 ····· 17

5 管脚配置 ····· 19

5.1 QFN 33 pin ····· 19

5.2 LQFP 48 pin ····· 20

5.3 管脚描述 ····· 21

6 功能描述 ····· 24

6.1 ARM® Cortex™-M0内核 ····· 24

6.2 系统管理器 ····· 26

6.2.1 概述 ····· 26
6.2.2 系统复位 ····· 26
6.2.3 系统电源架构 ····· 26
6.2.4 系统存储器映射 ····· 28
6.2.5 系统存储器映射表 ····· 30
6.2.6 系统管理器控制寄存器映射 ····· 31
6.2.7 系统定时器(SysTick) ····· 62
6.2.8 嵌套向量中断控制器 (NVIC) ····· 67
6.2.9 系统控制器寄存器映射 ····· 91

6.3 时钟控制器 ····· 99

6.3.1 概述 ····· 99
6.3.2 时钟发生器方块图 ····· 99
6.3.3 系统时钟 & SysTick 时钟 ····· 101
6.3.4 AHB 时钟源选择 ····· 102
6.3.5 外设时钟源选择 ····· 103
6.3.6 掉电模式（深度睡眠模式） 时钟 ····· 104
6.3.7 分频器输出 ····· 104
6.3.8 时钟控制寄存器映射 ····· 105
6.3.9 时钟控制寄存器描述 ····· 106

6.4 通用I/O ····· 127

6.4.1 概述 ····· 127
6.4.2 Port 0-4 控制器寄存器映射映射 ····· 129
6.4.3 Port 0-4 控制器寄存器描述 ····· 133

6.5 I2C 总线控制器 (主机/从机) ····· 147

6.5.1 概述 ····· 147
6.5.2 特征 ····· 147
6.5.3 功能描述 ····· 148
6.5.4 I2C 协议寄存器 ····· 152

6.5.5 I2C 控制器寄存器映射 ····· 156
6.5.6 I2C 控制器寄存器描述 ····· 157
6.5.7 操作模式 ····· 165

6.6 PWM发生器和捕捉定时器 ····· 171
6.6.1 概述 ····· 171
6.6.2 特征 ····· 172
6.6.3 PWM 框图 ····· 173
6.6.4 PWM 功能描述 ····· 177
6.6.5 PWM 控制器寄存器映射 ····· 184
6.6.6 PWM 控制器寄存器描述 ····· 187

6.7 串行外设接口(SPI)控制器 ····· 211
6.7.1 概述 ····· 211
6.7.2 特性 ····· 211
6.7.3 SPI 框图 ····· 212
6.7.4 SPI 功能描述 ····· 213
6.7.5 SPI 时序波形图 ····· 221
6.7.6 SPI编程范例 ····· 223
6.7.7 SPI串行总线控制寄存器映射 ····· 225
6.7.8 SPI控制寄存器描述 ····· 226

6.8 定时器控制器 ····· 237
6.8.1 概述 ····· 237
6.8.2 特征 ····· 237
6.8.3 定时器方框图 ····· 238
6.8.4 功能描述 ····· 239
6.8.5 定时器控制器寄存器映射 ····· 242

6.9 看门狗定时器 (WDT) ····· 254
6.9.1 概述 ····· 254
6.9.2 特征 ····· 256
6.9.3 WDT 框图 ····· 256
6.9.4 看门狗定时器控制寄存器映射 ····· 257

6.10 UART接口控制器 ····· 260
6.10.1 概述 ····· 260
6.10.2 特性 ····· 262
6.10.3 UART 框图 ····· 263
6.10.4 IrDA 模式 ····· 266
6.10.5 LIN (Local Interconnection Network) 模式 ····· 268
6.10.6 RS-485 模式 ····· 269
6.10.7 UART 接口控制寄存器映射 ····· 271
6.10.8 UART接口控制寄存器描述 ····· 273

6.11 模拟数字转换(ADC) ····· 298
6.11.1 概述 ····· 298
6.11.2 特征 ····· 298
6.11.3 ADC框图 ····· 299
6.11.4 ADC操作步骤 ····· 300

6.11.5 ADC 寄存器映射 ····· 305
6.11.6 ADC 寄存器描述 ····· 306

6.12 模拟比较器 (ACMP) ····· 318
6.12.1 概述 ····· 318
6.12.2 特性 ····· 318
6.12.3 框图 ····· 318
6.12.4 功能描述 ····· 319
6.12.5 寄存器映射 ····· 320
6.12.6 寄存器描述 ····· 321

6.13 外部总线接口 (EBI) ····· 326
6.13.1 概述 ····· 326
6.13.2 特性 ····· 326
6.13.3 EBI 框图 ····· 327
6.13.4 操作步骤 ····· 328
6.13.5 EBI 控制器寄存器映射 ····· 334
6.13.6 EBI 控制器寄存器描述 ····· 334

6.14 Flash内存控制器(FMC) ····· 337
6.14.1 概述 ····· 337
6.14.2 特性 ····· 337
6.14.3 FMC 框图 ····· 338
6.14.4 FMC组织结构 ····· 339
6.14.5 启动选择 ····· 341
6.14.6 Data Flash ····· 342
6.14.7 在系统编程(ISP) ····· 343
6.14.8 FMC控制寄存器映射 ····· 346
6.14.9 FMC控制器寄存器描述 ····· 347

7 用户 配置区 ····· 355

8 典型应用电路 ····· 357

9 电气特性 ····· 358

9.1 绝对最大额定值 ····· 358

9.2 DC电气特性 ····· 359

9.3 AC 电气特性 ····· 364
9.3.1 外部高速晶振 ····· 364
9.3.2 外部振荡器 ····· 364
9.3.3 外部高速晶振的典型应用电路 ····· 365
9.3.4 内部 22.1184 MHz RC振荡器 ····· 366
9.3.5 内部 10kHz RC 振荡器 ····· 366

9.4 模拟量特性 ····· 367
9.4.1 12-bit SARADC规格 ····· 367
9.4.2 LDO规格 & Power 管理 ····· 368
9.4.3 低压复位规格 ····· 369
9.4.4 欠压检测规格 ····· 369

9.4.5 上电复位规格(5V) ····· 369
9.4.6 温度传感器规格 ····· 370
9.4.7 比较器规格 ····· 370
9.5 Flash DC 电器特性 ····· 371
10 封装尺寸 ····· 372
10.1 LQFP-48 (7x7x1.4mm^2 Footprint 2.0mm) ····· 372
10.2 QFN-33 (5X5 mm^2, Thickness 0.8mm, Pitch 0.5 mm) ····· 373
11 版本历史 ····· 374

LIST OF FIGURES

图6.2.1-1NuMicro™ M051 系列方块图 16
图6.2.1-1 NuMicro™ 命名规则 18
图6.2.1-1 NuMicro™ M051 系列QFN33 引脚图 19
图6.2.1-1 NuMicro™ M051 系列 LQFP-48引脚图 20
图6.2.1-1 功能框图 24
图6.2.3-1 NuMicro M051™ 系列电源架构图 27
图6.3.2-1时钟发生器方块图 100
图6.3.3-1系统时钟框图 101
图6.3.3-2SysTick时钟控制框图 101
图6.3.4-1AHB 时钟源HCLK 102
图6.3.5-1外设时钟源选择PCLK 103
图6.3.7-1分频器的时钟源 104
图6.3.7-2分频器框图 105
图6.4.1-1推挽输出 127
图6.4.1-2开漏输出 128
图6.4.1-3 准双端I/O 模式 128
图6.5.1-1 I2C 总线时序 147
图6.5.3-1 I2C 协议 148
图6.5.3-2主机向从机传输数据 149
图6.5.3-3 主机读取从机的数据 149
图6.5.3-4 START 和 STOP 条件 150
图6.5.3-5 I2C总线上的位传输 151
图6.5.3-6 I2C总线上的应答信号 151
图6.5.4-1 I2C 数据移位 153
图6.5.4-2: I2C 超时计数器框图 155
图 6.5.7-1对如下五图的说明 165
图6.5.7-2 主机发送模式 166
图6.5.7-3主机接收模式 167
图6.5.7-5从机发送模式 168
图6.5.7-4从机接收模式 169
图6.5.7-6广播呼叫模式 170

图6.6.3-1 PWM发生器 0 时钟源控制 173
图6.6.3-2 PWM 发生器0结构框图 173
图6.6.3-3 PWM 发生器 2 时钟源控制 174
图6.6.3-4 PWM 发生器 2结构框图 174
图6.6.3-5PWM 发生器 4 时钟源控制 175
图6.6.3-6 PWM 发生器 4 结构框图 175
图6.6.3-7 PWM 发生器 6 时钟源控制 176
图6.6.3-8 PWM 发生器 6 结构框图 176
图6.6.4-1PWM定时器内部比较器输出 177
图6.6.4-2 PWM定时器操作时序 178
图6.6.4-3 PWM 双缓存图解 178
图6.6.4-4 PWM 控制器输出占空比 179
图6.6.4-5死区发生器操作 179
图6.6.4-6 捕捉操作时序 180
图6.6.4-7 PWM A组 PWM-定时器中断结构图 181
图6.6.4-8 PWM B组 PWM-定时器中断结构图 181
图6.7.3-1SPI 框图 212
图6.7.4-1 SPI主机模式应用框图 213
图6.7.4-2 SPI从机模式应用框图 213
图6.7.4-3 可调串行时钟频率 215
图 6.7.4-4 一次传输32-Bit 216
图6.7.4-5 一次传输两笔 (Burst Mode) 216
图 6.7.4-6 字节重排列 218
图6.7.4-7 字节休眠时序波形 219
图6.7.5-1 主机模式下SPI 时序 221
图6.7.5-2 主机模式下SPI 时序(Alternate Phase of SPICLK) 221
图6.7.5-3 从机模式下SPI 时序 222
图6.7.5-4 从机模式下SPI 时序(Alternate Phase of SPICLK) 222
图6.8.3-1定时器控制器框图 238
图6.8.3-2定时器控制的时钟源 238
图6.8.4-1 连续计数模式 240
图6.9.1-1 中断时序与复位信号时序 255

图6.9.3-1 看门狗定时器时钟控制 256
图6.9.3-2 看门狗定时器框图 256
图6.10.3-1 UART 时钟控制框图 263
图6.10.3-2 UART 框图 263
图6.10.3-3自动流控制框图 265
图6.10.4-1 IrDA 框图 266
图6.10.4-2 IrDA TX/RX 时序框图 267
图 6.10.5-1 LIN 帧结构 268
图6.10.6-1 RS-485 帧结构 270
图6.11.3-1ADC 控制器框图 299
图6.11.4-1 ADC时钟控制 300
图6.11.4-2单次转换模式时序图 301
图6.11.4-3使能通道上单周期扫描时序图 302
图6.11.4-4 使能通道的连续扫描时序图 303
图6.11.4-5 A/D 转换结果监控框图 304
图6.11.4-6 A/D 控制器中断 304
图6.11.6-1 ADC单端输入转换电压和转换结果映射图 307
图6.11.6-2 ADC差分输入转换电压和转换结果映射图 308
图 6.11.9-1 模拟比较器框图 318
图 6.11.10-1 比较器中断源 319
图6.12.3-1 EBI 框图 327
图6.12.4-1 16位EBI数据宽度与16位器件连接 328
图6.12.4-2 8位EBI数据宽度与8位设备连接 329
图6.12.4-3 16位数据宽度的时序控制波形 331
图6.12.4-4 8位数据宽度时序控制波形 332
图6.12.4-5 插入空闲周期的时序控制波形 333
图6.13.3-1 Flash存储器控制器框图 338
图6.13.4-1 Flash存储器组织结构 340
图6.13.5-1上电时启动选择(BS)从CONFIG0加载 341
图6.13.6-1 Flash存储器结构 342
图6.13.7-1 ISPGo 时序图 343
图6.13.7-2 ISP 软件编程流程 344

图9.3.3-1 典型晶振应用电路..... 365

LIST OF TABLES

表 5.1-1 NuMicro™ M051 系列产品选型指南 17
表 5.3-1 NuMicro™ M051 系列引脚描述 23
表 6.2-1片上模块的地址空间分配 28
表 6.2-2 片上模块的地址空间分配 29
表 6.2-3 异常模式 69
表 6.2-4 系统中断映射 69
表 6.2-5 向量表格式 70
表 6.3-1掉电模式控制表 108
表 6.5-1 I^2C 状态码描述表 154
表6.7-1 字节顺序和字节休眠条件 219
表6.9-1看门狗超时间隔选择 254
表 6.10-1 UART波特率方程 260
表 6.10-2 UART波特率设置表 261
表 6.10-3 UART 中断源和标志表（ Software Mode） 290
表 6.10-4 波特率方程表 293
表 6.13-1 Flash存储器地址映射Flash存储器组织结构如下所示: 339
表 6.13-2 ISP模式 345

1 概述

NuMicro M051™ 系列是以ARM® Cortex™-M0为内核的32位微控制器，应用于工业控制和需要丰富通信接口的领域. Cortex™-M0是ARM最新的32位嵌入式处理器，成本仅相当于传统的8位微控制器。NuMicro M051™ 系列包括M052, M054, M058 和 M0516.

NuMicro M051™系列运行频率最高可达50MHz，因此M051系列可应用于各种各样的工业控制和需要高性能CPU的领域. NuMicro M051™ 系列内嵌有 8K/16K/32K/64K-字节的flash存储器, 4K字节数据flash存储器，4K字节在系统编程（ISP）的flash存储器，及4K字节SRAM存储器.

许多系统级外设功能，如I/O端口、 EBI (外部总线接口)、Timer、UART、 SPI、I2C、PWM、ADC，看门狗定时器和欠压检测，都已经被集成在NuMicro M051™ 系列，以减少系统外围元器件数量，节省电路板空间和系统成本. 这些功能使NuMicro M051™ 系列适用于广泛应用。

此外，NuMicro M051™系列带有ISP（在系统编程）和ICP（在电路编程）功能，允许用户无需取下芯片，直接在电路板上对程序存储器进行升级。

2 特征

- 内核
  - ARM® Cortex™-M0内核运行频率可达50MHz.
  - 一个 24位系统定时器。
  - 支持低功耗睡眠模式.
  - 单指令周期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。
  - 支持串行调试（SWD）接口，2 个观察点/4 断点。
- 内建一组 LDO支持宽工作电压范围：2.5V~5.5V
- 存储器
  - 8KB/16KB/32KB/64KB Flash用于存储用户程序(APROM)
  - 4KB Flash用于存储数据(DataFlash)
  - 4KB Flash用于存储ISP引导代码 (LDROM)
  - 4KB字节SRAM用作内部高速暂存存储器
- 时钟控制
  - 可编程的系统时钟源
  - 4~24 MHz外部高速晶振输入
  - 22.1184MHz内部高速振荡器（精度可达3%）
  - 低功耗10KHz 的低速振荡器用于看门狗及睡眠模式唤醒CPU
  - PLL支持CPU最高运行在50MHz
- I/O 端口
  - 在LQFP-48管脚封装中，最多支持40个通用I/O端口（GPIO）
  - 4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
  - 可选择TTL输入或施密特触发输入
  - I/O管脚可被配置为边沿/电平触发模式的中断源
  - 较强的拉电流驱动能力和灌电流承受能力
- 定时器
  - 4组32位定时器，每组定时器均带有24位上数计数器和8位预分频器

- 每个定时器有独立的时钟源
- 24位定时器当前值可由定时器数据寄存器（TDR）读出
- 提供3种工作模式：单脉冲模式，周期模式，开关模式。
- 支持事件计数功能
- 提供外部信号捕获/复位计数器功能，等同于8051 Timer2

- 看门狗定时器
  - 多路时钟源选择
  - 支持在掉电模式和休眠模式下唤醒CPU的功能
  - 看门狗定时器溢出时可选择产生中断/系统复位
- PWM
  - 内建4个16位PWM发生器，提供8路PWM输出或4对互补的PWM输出
  - 每个PWM发生器配有单独的时钟源选择器，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)支持输入信号的上升沿/下降沿的捕捉功能
  - 支持捕捉中断
- UART
  - 最多两组UART设备
  - 可编程波特率发生器
  - 接收器和发送器支持缓冲，均带有15bytes的FIFO缓冲
  - 流控功能供选择(CTS 和 RTS)
  - 支持 IrDA(SIR) 功能
  - 支持RS485功能
  - 支持LIN功能
- SPI
  - 最高支持2组SPI设备
  - 支持 SPI主机/从机模式
  - 全双工同步串行数据传输
  - 从模式下，支持3线模式
  - 数据长度可改变（从1到32位）
  - 可设置MSB或LSB优先的传输模式
  - Rx可在串行时钟的上升/下降沿锁存数据
  - Tx可在串行时钟的上升/下降沿发送数据
  - 32位传输模式下支持字节暂停模式

- I2C
  - 支持主机/从机模式
  - 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ADC，转换速率达 760k SPS
  - 最多8通道单端输入或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个互补/无符号格式的转换结果
  - 每通道有独立的存放转换结果的寄存器
  - 支持转换值监测（或比较），用于门槛电压检测
  - 转换开始可由软件或外部触发
- ACMP
  - 最多两个模拟比较器
  - 负端可以选择内部带隙电压或者外部输入电压
  - 当比较结果发生改变时发生中断
  - 支持掉电模式下唤醒CPU的功能
- EBI (外部总线接口)，用于外部存储器映射设备的访问
  - 可访问的空间: 8位模式下为64KB，16位模式下为128KB
  - 支持8bit/16bit 数据宽度
  - 16bit数据宽度下，支持字节写功能
- 在系统编程（ISP）与在电路编程（ICP）
- 内嵌温度传感器，1℃分辨率
- 欠压检测
  - 支持四级检测电压: 4.3V/3.7V/2.7V/2.2V
  - 支持欠压中断和复位选择
- 96-bit唯一ID号
- LVR （低电压复位）

- 门槛电压: 2.0V
- 工作温度: -40℃~85℃
- 封装:
  - 无铅封装 (RoHS)
  - 48-pin LQFP, 33-pin QFN

3 方块图

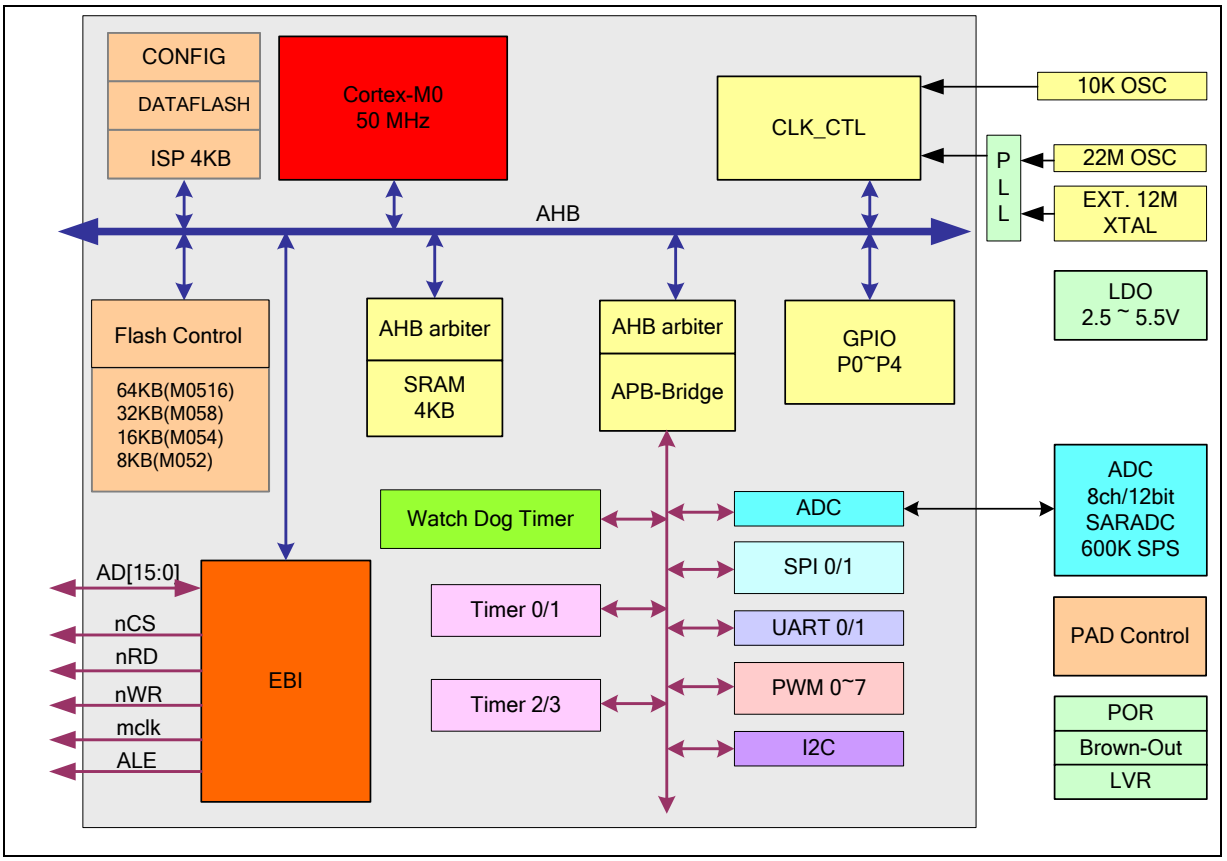


图 6.2.1-1NuMicro™ M051 系列方块图

4 选型表

NuMicro M051™ 系列产品选型指南

Part number	APROM	RAM	Data Flash	LDROM	I/O	Timer	Connectivity			COMP	PWM	ADC	EBI	ISP ICP	Package
							UART	SPI	I2C						
M052LBN	8KB	4KB	4KB	4KB	40	4x32-bit	2	2	1	2	8	8X12-bit	v	v	LQFP48
M052ZBN	8KB	4KB	4KB	4KB	24	4x32-bit	2	1	1	2	5	8X12-bit		v	QFN33
M054LBN	16KB	4KB	4KB	4KB	40	4x32-bit	2	2	1	2	8	8X12-bit	v	v	LQFP48
M054ZBN	16KB	4KB	4KB	4KB	24	4x32-bit	2	1	1	2	5	8X12-bit		v	QFN33
M058LBN	32KB	4KB	4KB	4KB	40	4x32-bit	2	2	1	2	8	8x12-bit	v	v	LQFP48
M058ZBN	32KB	4KB	4KB	4KB	24	4x32-bit	2	1	1	2	5	8x12-bit		v	QFN33
M0516LBN	64KB	4KB	4KB	4KB	40	4x32-bit	2	2	1	2	8	8X12-bit	v	v	LQFP48
M0516ZBN	64KB	4KB	4KB	4KB	24	4x32-bit	2	1	1	2	5	8X12-bit		v	QFN33

表 5.1-1 NuMicro™ M051 系列产品选型指南

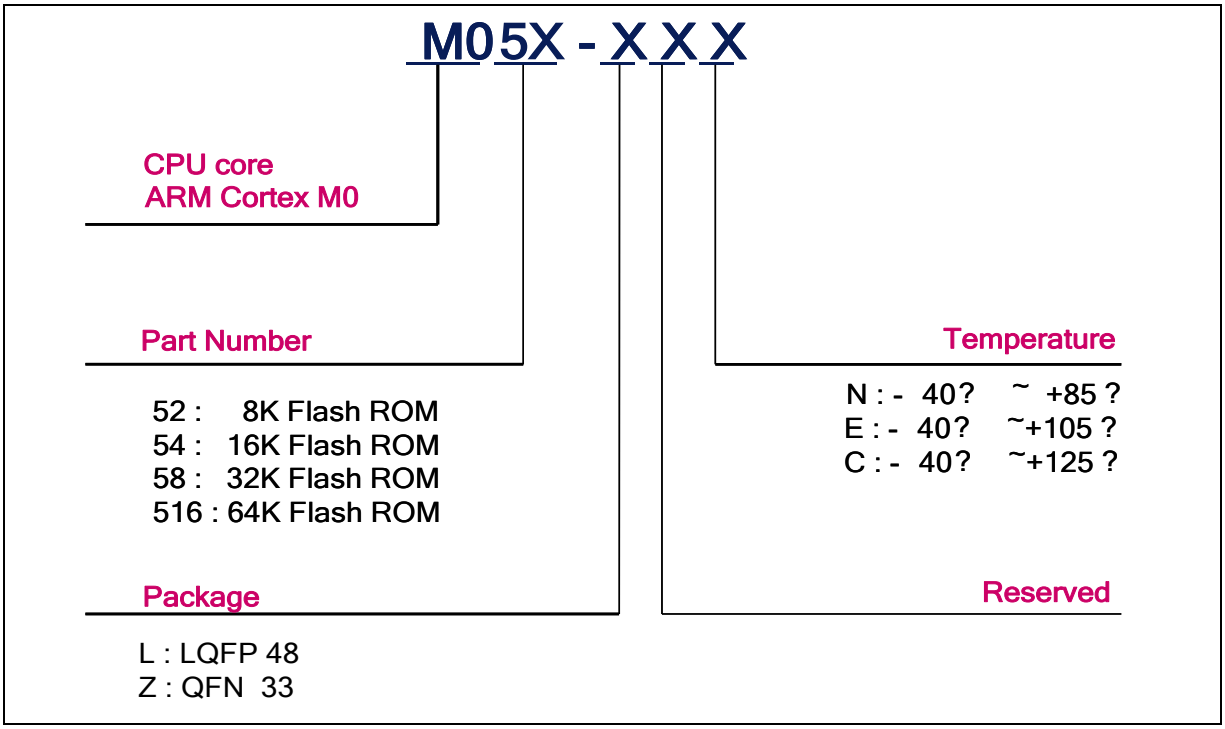


图6.2.1-1 NuMicro™ 命名规则

5 管脚配置

5.1 QFN 33 pin

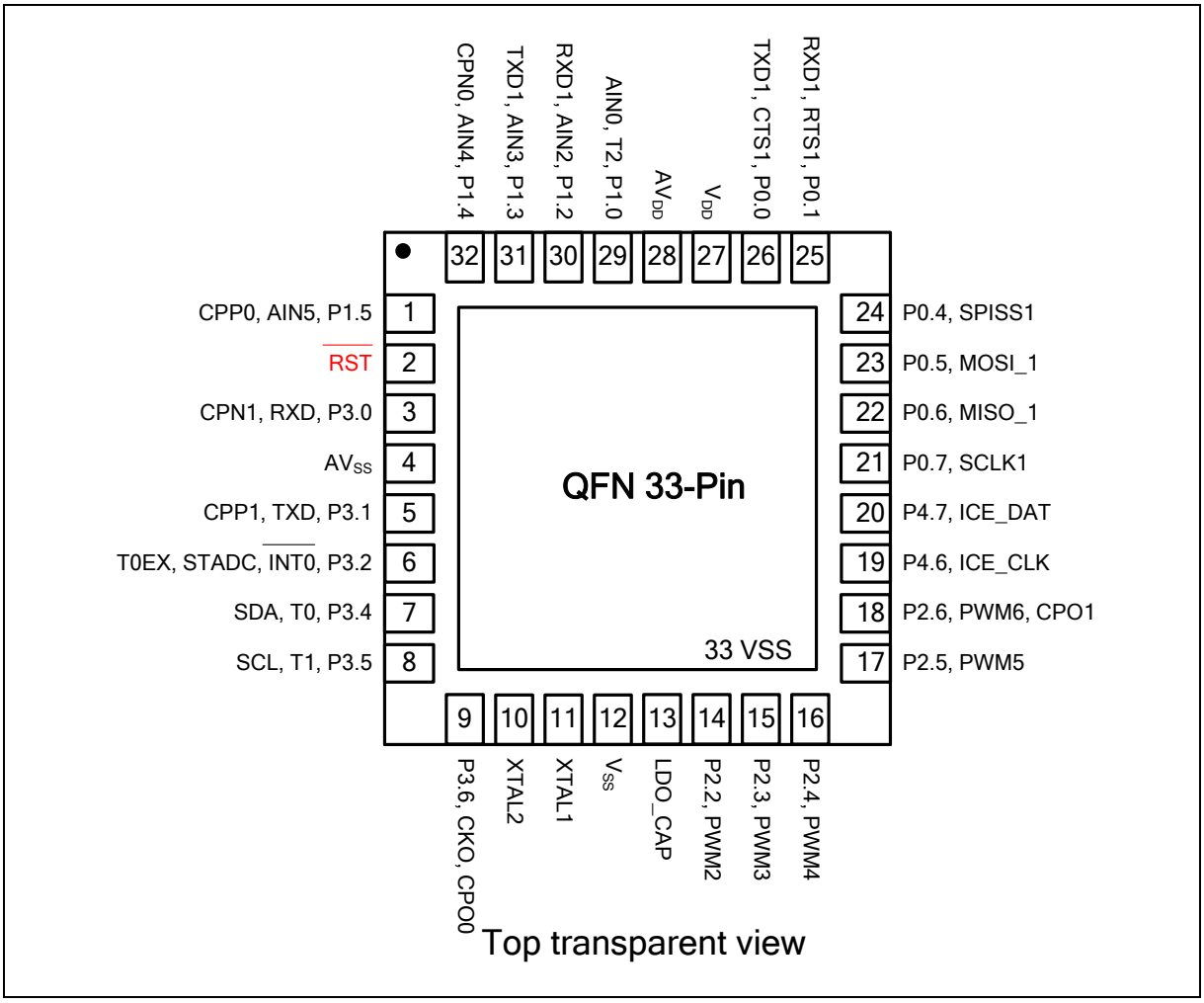


图6.2.1-1 NuMicro™ M051 系列QFN33 引脚图

5.2 LQFP 48 pin

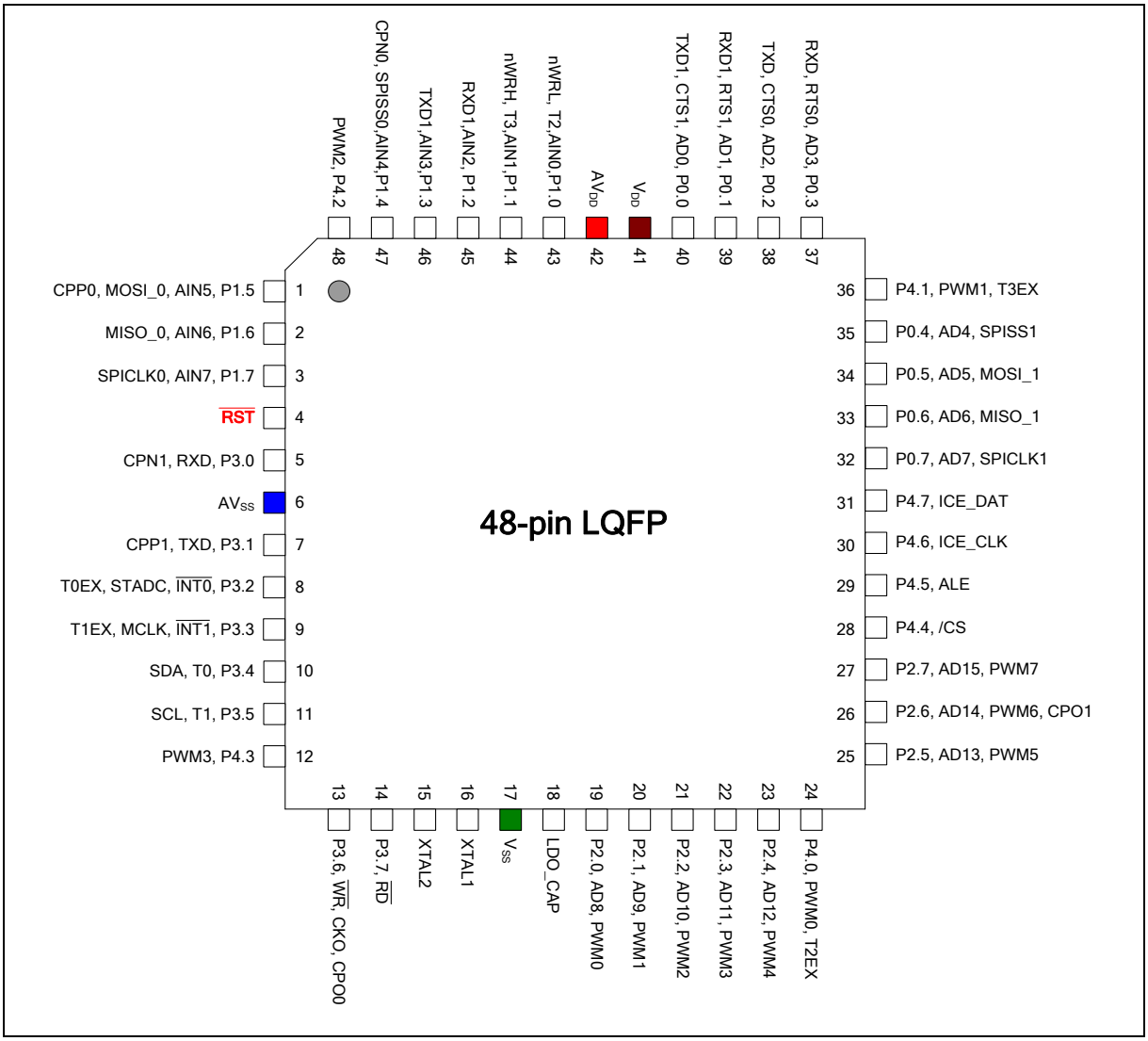


图6.2.1-1 NuMicro™ M051 系列 LQFP-48引脚图

5.3 管脚描述

管脚号		符号	复用功能			类型[1]	描述
QFN33	LQFP48		1	2	3		
11	16	XTAL1				I (ST)	**晶振脚1:** 内部反向放大器输入管脚。当 FOSC[1:0] (CONFIG3[1:0]) 都为1（默认值）时，系统时钟由外部高速晶振或谐振器输入。
10	15	XTAL2				O	**晶振脚2:** 内部反向放大器输出管脚。此管脚输出晶振脚1的反向信号。
27	41	VDD				P	**电源输入脚:** 提供**数字**电源V_{DD}。
12 33	17	VSS				P	**地:** 数字电源地
28	42	AVDD				P	**模拟电源输入脚:** 提供**模拟**电源 AV_{DD}
4	**6**	**AVSS**				**P**	**地:** 模拟 电源地
13	**18**	**LDO_CAP**				**P**	**LDO:** LDO 输出管脚 **注: 必须外接 1uF 电容。**
2	**4**	**/RST**				**I (ST)**	**复位脚:** /RST管脚为施密特触发输入管脚，用于芯片复位。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。
26	**40**	**P0.0**	**CTS1**	**AD0**	TXD1[2]	**D, I/O**	**端口0:** 端口0是8位的，具有四种输出模式，2种输入模式的管脚。并与下列功能复用，包括CTS1, RTS1, CTS0, RTS0, TXD1, RXD1, TXD, RXD, SPISS1, MOSI_1, MISO_1, 及SPCLK1。 当外部总线接口（EBI）被使能时，P0可复用为AD[7:0]。 SPISS1, MOSI_1, MISO_1, 和SPICLK1引脚用于SPI功能。 CTS0/1: UART0/1清除发送输入引脚 RTS0/1: UART0/1请求发送输出引脚
25	39	P0.1	RTS1	AD1	RXD1[2]	D, I/O	
NC	38	P0.2	CTS0	AD2	TXD[2]	D, I/O	
NC	37	P0.3	RTS0	AD3	RXD[2]	D, I/O	
24	35	P0.4	SPISS1	AD4		D,	

管脚号		符号	复用功能			类型[1]	描述
QFN33	LQFP48		1	2	3		
						I/O	RXD/TXD：用于 UART0 功能 RXD1/TXD1：用于UART1功能.
23	34	P0.5	MOSI_1	AD5		D, I/O	
22	33	P0.6	MISO_1	AD6		D, I/O	
21	32	P0.7	SPISCLK 1	AD7		D, I/O	
29	43	P1.0	T2	AIN0	$\overline{WRL}$	I/O	**端口1:** 端口1是8位的，具有四种输出模式，2种输入模式的管脚。并与下列功能复用，包括T2, T3, RXD1, TXD1, $\overline{WRL}$, $\overline{WRH}$, SPISS0, MOSI_0, MISO_0, 及SPICLK0. T2：Timer2的外部事件计数器输入管脚 T3：Timer3的外部事件计数器输入管脚 SPISS0, MOSI_0, MISO_0, 和SPICLK0引脚用于SPI功能。 AIN0~AIN7：用于12位ADC的模拟信号输入脚 RXD1/TXD1：供UART1使用 $\overline{WRL}$ / $\overline{WRH}$：EBI模式下，16-bit数据宽度时，用于输出低/高字节写使能信号· CPN0/CPP0：用于比较器0的正负端输入.
NC	44	P1.1	T3	AIN1	$\overline{WRH}$	I/O	
30	45	P1.2	RXD1	AIN2		I/O	
31	46	P1.3	TXD1	AIN3		I/O	
32	47	P1.4	SPISS0	AIN4	CPN0	I/O	
1	1	P1.5	MOSI_0	AIN5	CPP0	I/O	
NC	2	P1.6	MISO_0	AIN6		I/O	
NC	3	P1.7	SPICLK0	AIN7		I/O	
NC	19	P2.0	PWM0	AD8		D, I/O	**端口2:** 端口2是8位的，具有4种输出模式，2种输入模式的管脚。并与下列功能复用。 当外部总线接口（EBI）被使能时，P2可复用为AD[15:8]。 PWM0~PWM7 用于PWM输出功能 CPO1：比较器1的输出引脚
NC	20	P2.1	PWM1	AD9		D, I/O	
14	21	P2.2	PWM2	AD10		D, I/O	
15	22	P2.3	PWM3	AD11		D, I/O	
16	23	P2.4	PWM4	AD12		D, I/O	
17	25	P2.5	PWM5	AD13		D, I/O	

管脚号		符号	复用功能			类型[1]	描述
QFN33	LQFP48		1	2	3		
18	26	P2.6	PWM6	AD14	CPO1	D, I/O	
NC	27	P2.7	PWM7	AD15		D, I/O	
3	**5**	**P3.0**	**RXD**		CPN1	**I/O**	**端口3:** 端口3是8位的，具有4种输出模式，2种输入模式的管脚。并与下列功能复用。包括 RXD，TXD，$\overline{INT0}$，$\overline{INT1}$，T0，T1，$\overline{WR}$，$\overline{RD}$，STADC，MCLK，SDA，SCL，CKO，CPN1，CPP1，T0EX，T1EX及CPO0。 T0：Timer0的外部事件计数器输入管脚 T1：Timer1的外部事件计数器输入管脚 RXD/TXD：供UART0使用 SDA/SCK：供I2C功能使用 MCLK：EBI 时钟输出脚 CKO：HCLK 时钟输出 STADC：ADC 外部触发信号脚 CPN1/CPP1：比较器1的正/负端输入引脚. CPO0：比较器0的输出引脚 T0EX/T1EX：Timer0/1的外部捕获/复位功能输入引脚.
5	7	P3.1	TXD		CPP1	I/O	
6	8	P3.2	$\overline{INT0}$	STADC	T0EX	I/O	
NC	9	P3.3	$\overline{INT1}$	MCLK	T1EX	I/O	
7	10	P3.4	T0	SDA		I/O	
8	11	P3.5	T1	SCL		I/O	
9	13	P3.6	$\overline{WR}$	CKO	CPO0	I/O	
NC	14	P3.7	$\overline{RD}$			I/O	
NC	24	P4.0	PWM0		T2EX	I/O	**端口4:** 端口4是8位的，具有4种输出模式，2种输入模式的管脚。并与下列功能复用。包括/CS， ALE，ICE_CLK 及ICE_DAT。 /CS EBI 的片选信号脚。 ALE (地址锁存使能脚)：用于使能地址锁存，在端口0和端口2上把地址从数据中分离出来。 ICE_CLK/ICE_DAT：用于JTAG仿真。 当EBI使能，P4.0-P4.3可用作PWM0-3。 T2EX/T3EX：Timer2/3的外部捕获/复位功能输入引脚.
NC	36	P4.1	PWM1		T3EX	I/O	
NC	48	P4.2	PWM2			I/O	
NC	12	P4.3	PWM3			I/O	
NC	28	P4.4	/CS			I/O	
NC	29	P4.5	ALE			I/O	
19	30	P4.6	ICE_CLK			I/O	
20	31	P4.7	ICE_DAT			I/O	

表 5.3-1 NuMicro™ M051 系列引脚描述

[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。

[2] 引脚特性由软件设定，但是某一时刻只有一个功能可以被选择.

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。图6.2.1-1 显示了处理器内核的各个功能模块.

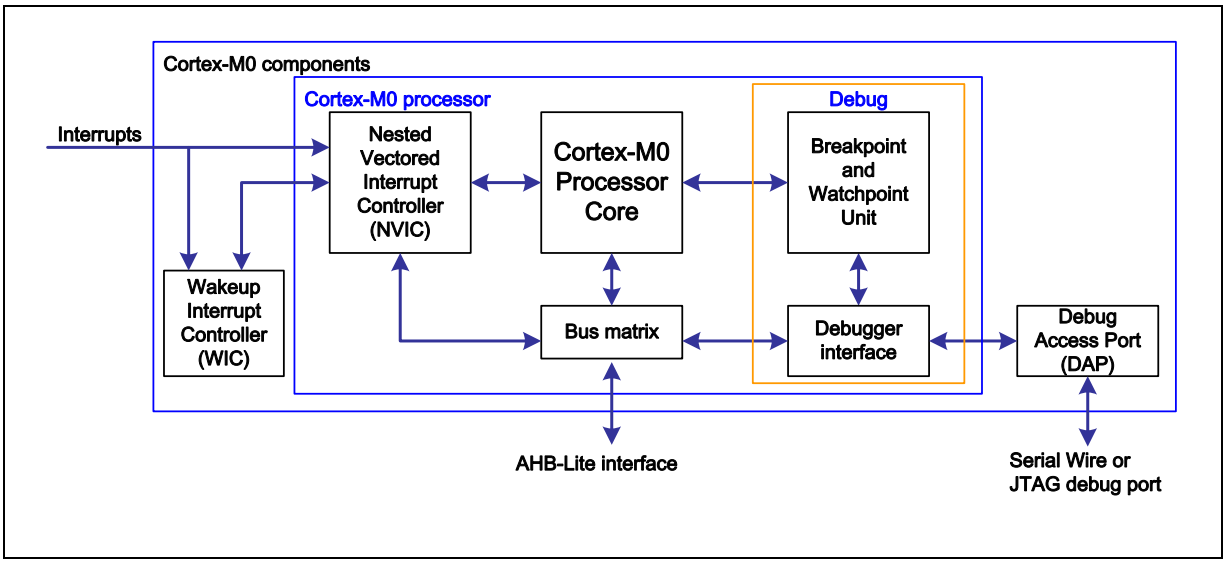


图6.2.1-1 功能框图

设备提供:

低门数处理器特性:

- 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，固定延迟的中断处理能力。
- 可以丢弃和重新开始多次加载/存储和多周期乘法以保证快速中断处理。
- 与 C 应用程序二进制接口兼容的异常模式（C-ABI）。
  ARMv6-M（C-ABI）兼容异常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，或者从中断返回时的 sleep-on-exit 特性可以进入低功耗的休眠模式。

NVIC 特征:

- 32 个外部中断输入，每个中断具有 4 级优先级。
- 不可屏蔽中断输入（NMI）。
- 支持电平和脉冲触发中断。
- 中断唤醒控制器(WIC)，支持极低功耗休眠模式。

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

系统管理器包括如下功能

- 系统复位
- 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

有如下事件之一发生时，系统复位，这些复位事件标志可以由寄存器**RSTRC**读出.

- 上电复位(POR)
- 复位脚（/RESET）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)
- CPU复位
- 系统复位

6.2.3 系统电源架构

该器件的电源架构分为三个部分：

- 由AVDD 和AVSS提供的模拟电源，为模拟部分提供工作电压。
- 由VDD与VSS提供的数字电源，为内部稳压器提供电压，内部稳压器向数字操作和I/O管脚提供固定的1.8V电压。

内部电压稳压器（LDO）的输出，需要在相应管脚附近接一颗电容。图6.2.3-1示出了该设备的电源架构：

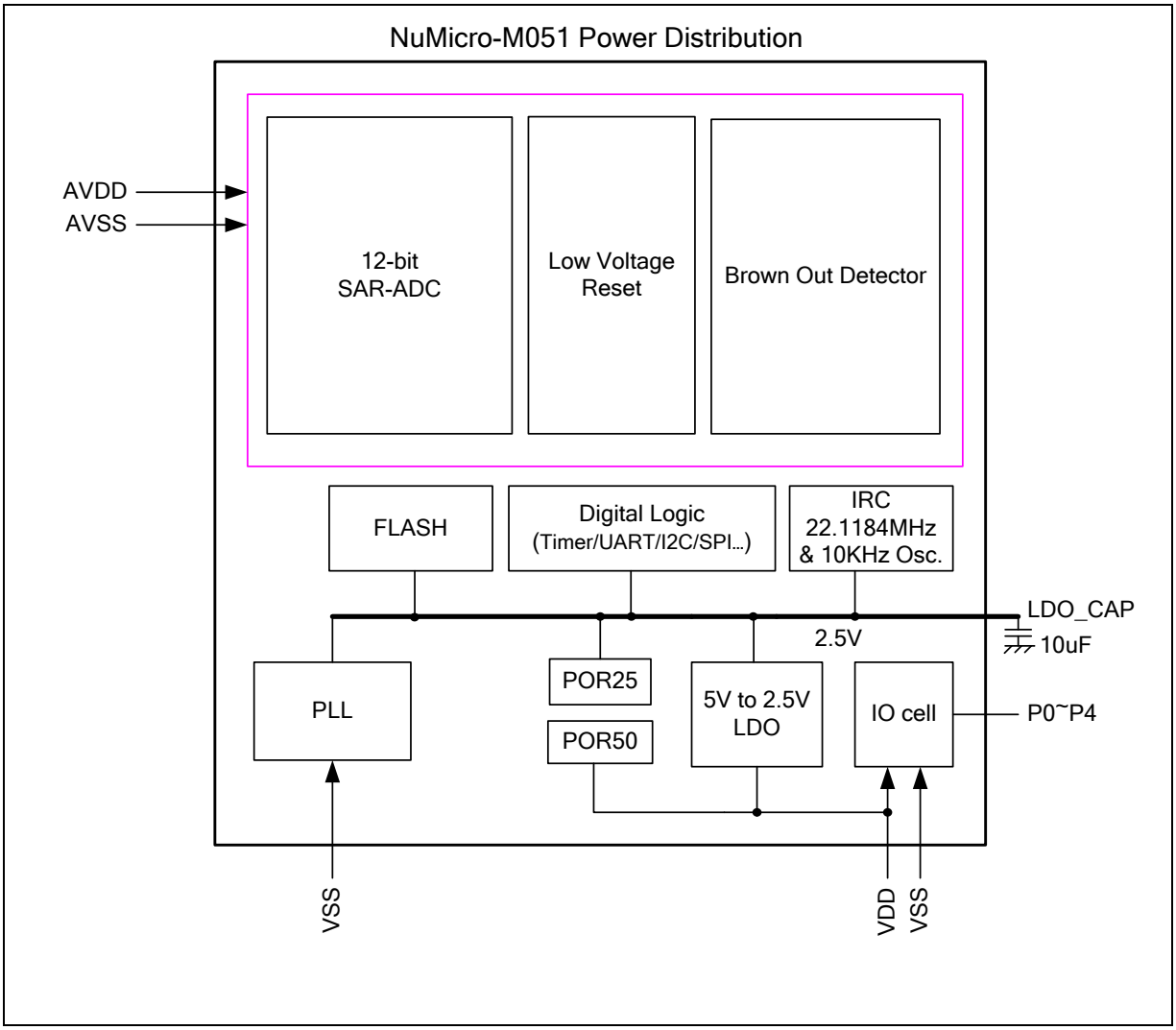


图 6.2.3-1 NuMicro M051™ 系列电源架构图

6.2.4 系统存储器映射

NuMicro M051™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在表6.2-1中示出。详细的寄存器地址分配和编程将在后续的各个片上外设描述章节被描述。NuMicro M051™ 系列仅支持小端数据格式。

表 6.2-1片上模块的地址空间分配

地址空间	标志	模块
Flash & SRAM 内存空间		
0x0000_0000 – 0x0000_FFFF	FLASH_BA	FLASH内存空间(64KB)
0x2000_0000 – 0x2000_0FFF	SRAM_BA	SRAM内存空间(4KB)
EBI 空间 (0x6000_0000 ~ 0x6001_FFFF)		
0x6000_0000 – 0x6001_FFFF	EBI_BA	EBI 空间
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~P4) 控制寄存器
0x5000_C000 – 0x5000_FFFF	FMC_BA	Flash 存储器控制寄存器
0x5001_0000 – 0x5001_3FFF	EBI_CTL_BA	EBI 控制寄存器 (128KB)
APB模块空间(0x4000_0000 ~ 0x400F_FFFF)		
0x4000_4000 – 0x4000_7FFF	WDT_BA	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	I2C_BA	I2C接口控制寄存器
0x4003_0000 – 0x4003_3FFF	SPI0_BA	带主/从功能的SPI0控制寄存器
0x4003_4000 – 0x4003_7FFF	SPI1_BA	带主/从功能的SPI1 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
0x400D_0000 – 0x400D_3FF	ACMP_BA	模拟比较器控制寄存器
0x400E_0000 – 0x400E_FFFF	ADC_BA	模数转换器(ADC)控制寄存器

地址空间	标志	模块
0x4011_0000 – 0x4011_3FFF	TMR23_BA	Timer2/Timer3控制寄存器
0x4014_0000 – 0x4014_3FFF	PWMB_BA	PWM4/5/6/7控制寄存器
0x4015_0000 – 0x4015_3FFF	UART1_BA	UART1 控制寄存器
System Control Space (0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SCS_BA	System 定时器控制寄存器
0xE000_E100 – 0xE000_ECFF	SCS_BA	外部中断控制器控制寄存器
0xE000_ED00 – 0xE000_ED8F	SCS_BA	System控制寄存器

表 6.2-2 片上模块的地址空间分配

6.2.5 系统存储器映射表

M052/54/58/516

4 GB	Reserved	0xFFFF_FFFF | 0xE000_F000
	System Control	0xE000_EFFF 0xE000_E000
	Reserved	0xE000_E00F | 0x6002_0000
	EBI	0x6001_FFFF 0x6000_0000
	Reserved	0x5FFF_FFFF | 0x5020_0000
	AHB	0x501F_FFFF 0x5000_0000
	Reserved	0x4FFF_FFFF | 0x4020_0000
	APB	0x401F_FFFF | 0x4000_0000
1 GB	Reserved	0x3FFF_FFFF | 0x2000_1000
	4 KB SRAM (M052/M054/M058/M0516)	0x2000_0FFF | 0x2000_0000
0.5 GB	Reserved	0x1FFF_FFFF | 0x0001_0000
	64 KB on-chip Flash (M0516)	0x0000_FFFF
	32 KB on-chip Flash (M058)	0x0000_7FFF
	16 KB on-chip Flash (M054)	0x0000_3FFF
	8 KB on-chip Flash (M052)	0x0000_1FFF 0x0000_0000
0 GB		

System Control

System Timer Control	0xE000_E000	SCS_BA

AHB peripherals

EBI Control	0x5001_0000	EBI_CTL_BA
FMC	0x5000_C000	FLASH_BA
GPIO Control	0x5000_4000	GPIO_BA
Interrupt Multiplexer Control	0x5000_0300	INT_BA
Clock Control	0x5000_0200	CLK_BA
System Global Control	0x5000_0000	GCR_BA

APB peripherals

UART1 Control	0x4015_0000	UART1_BA
PWM4/5/6/7 Control	0x4014_0000	PWMB_BA
Timer2/Timer3 Control	0x4011_0000	TMR23_BA
ADC Control	0x400E_0000	ADC_BA
COMP control	0x400D_0000	ACMP_BA
UART0 Control	0x4005_0000	UART0_BA
PWM0/1/2/3 Control	0x4004_0000	PWMA_BA
SPI1 Control	0x4003_4000	SPI1_BA
SPI0 Control	0x4003_0000	SPI0_BA
I2C Control	0x4002_0000	I2C_BA
Timer0/Timer1 Control	0x4001_0000	TMR01_BA
WDT Control	0x4000_4000	WDT_BA

6.2.6 系统管理器控制寄存器映射

寄存器	偏移量	R/W	描述	复位后的值
GCR_BA = 0x5000_0000				
PDID	GCR_BA+0x00	R	设备ID寄存器	0x1000_5200
RSTSRC	GCR_BA+0x04	R/W	系统复位源寄存器	0x0000_00XX
IPRSTC1	GCR_BA+0x08	R/W	外设复位控制寄存器1	0x0000_0000
IPRSTC2	GCR_BA+0x0C	R/W	外设复位控制寄存器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	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
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	偏移量	R/W	描述	复位后的值
PDID	GCR_BA+0x00	R	设备ID寄存器	0x1000_5200[1]

[1]每个型号的设备复位后都有一个唯一的默认ID.

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]							

Bits	描述	
[31:0]	PDID	**产品设备识别码.** 该寄存器反映器件的识别码。S/W可以读该寄存器识别所使用的器件. 例如, M052LBN PDID 的识别码是0x1000_5200.

NuMicro M051™ series	Part Device Identification Number
M052LBN	0x10005200
M054LBN	0x10005400
M058LBN	0x10005800
M0516LBN	0x10005A00
M052ZBN	0x10005203
M054ZBN	0x10005403
M058ZBN	0x10005803
M0516ZBN	0x10005A03

系统复位源寄存器(RSTSRC)

该寄存器提供具体的信息给软件用于识别上次操作引起芯片复位的复位源。

寄存器	偏移量	R/W	描述	复位后的值
RSTSRC	GCR_BA+04	R/W	系统复位源寄存器	0x0000_00XX

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
RSTS_CPU	RSTS_PMU	RSTS_MCU	RSTS_BOD	RSTS_LVR	RSTS_WDT	RSTS_RESET	RSTS_POR

Bits	描述	
[31:8]	保留	保留
[7]	**RSTS_CPU**	当软件向**CPU_RST** (IPRSTCR1[1])写入“1”，复位Cortex-M0 CPU 内核和FLASH控制器（FMC）时，**RSTS_CPU** 标志由硬件置位。. 1= 软件置CPU_RST为1时， Cortex-M0 CPU 内核与FMC复位. 0= CPU无复位 向该位写1清零.
[6]	保留	保留
[5]	**RSTS_MCU**	**RSTS_MCU**由来自MCU Cortex_M0 的“复位信号“置位，以表示当前的复位源. 1= MCU Cortex_M0 在软件向SYSRESTREQ(AIRCR[2]写1时，发出复位信号以复位系统. 0= MCU无复位 向该位写1清零.
[4]	**RSTS_BOD**	RSTS_BOD标志位由欠压检测模块的” 复位信号”置1，用于表示当前复位源。 1: 欠压检测模块发出复位信号使系统复位. 0: BOD无复位 向该位写1清零.

Bits	描述	
[3]	**RSTS_LVR**	RSTS_LVR标志位由低压复位模块的" 复位信号"置1，用于表示当前复位源。 1: 低压 LVR 模块发出复位信号使系统复位. 0: LVR无复位 向该位写1清零.
[2]	**RSTS_WDT**	**RSTS_WDT** 标志位由看门狗模块的" 复位信号"置1，用于说明当前复位源。 1: 看门狗模块发出复位信号使系统复位. 0: 没有看门狗复位信号 向该位写1清零
[1]	**RSTS_RESET**	**RSTS_RESET**标志位由/RESET脚的" 复位信号"置1，用于说明当前复位源。 1： /RESET脚上发出复位信号使系统复位. 0： 没有/RESET复位信号 向该位写1清零.
[0]	**RSTS_POR**	RSTS_POR标志位由POR模块的" 复位信号"置1，用于说明当前的复位源。 1: 上电复位POR发出复位信号使系统复位. 0: 没有POR复位信号 向该位写1清零.

外设复位控制寄存器1 (IPRSTC1)

寄存器	偏移量	R/W	描述	复位后的值
IPRSTC1	GCR_BA+08	R/W	外设复位控制寄存器1	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				EBI_RST	保留	CPU_RST	CHIP_RST

Bits	描述	
[31:4]	保留	保留
[3]	EBI_RST	**EBI 控制器复位（写保护）** 设置该位为 “1” 将产生复位信号到EBI。用户需要再将该位置”0”才能退出复位状态 该位是受保护的位，修改该位时，需要依次向0x5000_0100写入”59h”, “16h”, “88h”解除寄存器保护。 参考寄存器 REGWRPROT，地址GCR_BA + 0x100. 0= EBI 控制器正常工作 1= EBI 控制器复位
[2]	保留	保留
[1]	CPU_RST	**CPU内核复位（写保护）** 该位置1，CPU内核和Flash存储控制器复位。两个时钟周期后，该位自动清零。 该位是受保护的位，修改该位时，需要依次向0x5000_0100写入”59h”, “16h”, “88h” 解除寄存器保护， 参考寄存器 REGWRPROT，地址GCR_BA + 0x100. 0 : 正常 1 : 复位CPU

Bits	描述	
[0]	**CHIP_RST**	芯片复位（写保护） 该位置1，芯片复位，包括CPU内核和所有外设均复位。，两个时钟周期后，该位自动清零. CHIP_RST与上电复位一样，所有的芯片模块都复位，芯片缺省设置从flash重新加载 该位是写保护的，修改该位时，需要依次向0x5000_0100写入”59h”, “16h”, “88h” 解除寄存器保护。 参考寄存器 REGWRPROT，地址GCR_BA + 0x100. 0 : 正常 1 : 复位芯片

外设复位控制寄存器2 (IPRSTC2)

置”1”这些位将会产生异步复位信号给相应的IP。用户需要清零相应位来使IP离开复位状态。

寄存器	偏移量	R/W	描述	复位后的值
IPRST2	GCR_BA+0C	R/W	外设复位控制寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
保留			ADC_RST	保留			
23	22	21	20	19	18	17	16
保留	ACMP_RST	PWM47_RST	PWM03_RST	保留		UART1_RST	UART0_RST
15	14	13	12	11	10	9	8
保留		SPI1_RST	SPI0_RST	保留			I2C_RST
7	6	5	4	3	2	1	0
保留		TMR3_RST	TMR2_RST	TMR1_RST	TMR0_RST	GPIO_RST	保留

Bits	描述	
[31:29]	保留	保留
[28]	ADC_RST	**ADC控制器复位** “0”: ADC 模块正常工作 “1”: ADC 模块复位
[27:23]	保留	保留
[22]	ACMP_RST	**模拟比较器控制器复位** 0 = 模拟比较器控制器正常工作 1 = 模拟比较器控制器复位
[21]	PWM47_RST	**PWM4～7 控制器复位** 0= PWM4～7 模块正常工作 1= PWM4～7 模块复位
[20]	PWM03_RST	**PWM0～3 控制器复位** 0= PWM0～3 模块正常工作 1= PWM0～3 模块复位
[19:18]	保留	保留

Bits	描述	
[17]	**UART1_RST**	**UART1控制器复位** 0= UART1 正常工作 1= UART1 模块复位
[16]	**UART0_RST**	**UART0控制器复位** 0= UART0 正常工作 1= UART0 模块复位
[15:14]	**保留**	保留
[13]	**SPI1_ RST**	**SPI1控制器复位** 0= SPI1 正常工作 1= SPI1 模块复位
[12]	**SPI0_ RST**	**SPI0控制器复位** 0= SPI0 正常工作 1= SPI0 模块复位
[11:9]	**保留**	保留
[8]	**I2C_RST**	**I2C控制器复位** 0= I2C 模块 正常工作 1= I2C 模块复位
[7:6]	**保留**	保留
[5]	**TMR3_RST**	**Timer3控制器复位** 0= Timer3 正常工作 1= Timer3 模块复位
[4]	**TMR2_RST**	**Timer2控制器复位** 0= Timer2 正常工作 1= Timer2 模块复位
[3]	**TMR1_RST**	**Timer1控制器复位** 0= Timer1 正常工作 1= Timer1 模块复位

Bits	描述	
[2]	**TMR0_RST**	**Timer0控制器复位** 0= Timer0 正常工作 1= Timer0 复位
[1]	**GPIO_RST**	GPIO (P0~P4) 控制器复位 0= GPIO 正常工作 1= GPIO 复位
[0]	**保留**	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	偏移量	R/W	描述	复位后的值
BODCR	GCR_BA+18	R/W	欠压检测控制寄存器	0x0000_008X

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**
LVR_EN	**BOD_OUT**	**BOD_LPM**	**BOD_INTF**	**BOD_RSTEN**	**BOD_VL**		**BOD_EN**

Bits	描述	
[31:8]	保留	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. LVR默认配置下LVR复位是使能的，典型的LVR值为2.0V 1= 使能低电压复位功能，使能该位100US后，LVR功能生效**(默认)** 0= 禁用低电压复位功能
[6]	**BOD_OUT**	欠压检测输出的状态位 1 = 欠压检测输出状态为 1，表示检测到的电压低于 BOD_VL 设置。若 BOD_EN 是“0”, 该位保持为 “0” 0 =欠压检测输出状态为0，表示检测到的电压高于BOD_VL设置
[5]	**BOD_LPM**	欠压检测低功耗模式（写保护位） 1 = 使能 BOD 低功耗模式 0 = BOD 工作于正常模式(默认) BOD 在正常模式下消耗电流约为100uA, 低功耗模式下减少到当前的1/10，并且BOD响应速度变慢.
[4]	**BOD_INTF**	欠压检测中断标志

Bits	描述	
		1= 欠压检测到V_{DD} 下降到BOD_VL 的设定电压或V_{DD} 升到BOD_VL 的设定电压，该位设置为1，如果欠压中断被使能，则发生欠压中断. 0= 没有检测到任何电压由V_{DD} 下降或上升至BOD_VL 设定值.
[3]	**BOD_RSTEN**	**欠压复位使能（上电初始化和写保护位）** 1 = 使能欠压复位功能，当欠压检测功能使能后，检测的电压低于门槛电压 ，芯片发生复位 默认值由用户在配置flash控制寄存器时的config0 bit[20]设置 0 = 使能欠压中断功能，当欠压检测功能使能后，检测的电压低于门槛电压 ，就发送中断信号给MCU Cortex-M0 当BOD_EN使能，且中断被声明时，该中断会持续到将BOD_EN设置为"0". 通过禁用CPU中的NVIC以禁用BOD中断或者通过禁用BOD_EN禁用中断源可禁用CPU响应中断，如果需要BOD功能时，可重新使能BOD_EN功能
[2:1]	**BOD_VL**	**欠压检测门槛电压 电压选择 （上电初始化和写保护位）** 默认值 由用户在配置FLASH控制寄存器config0 bit[22:21]时设定

BOV_VL[1]	BOV_VL[0]	欠压值
1	1	4.5V
1	0	.8
0	1	2.7V
0	0	2.2V

[0]	**BOD_EN**	**欠压检测使能（上电初始化和写保护位）** 默认值 由用户在配置FLASH控制寄存器config0 bit[23]时设定 1 = 使能欠压检测功能 0 =禁用欠压检测功能

Temperature Sensor Control Register (TEMPCR)

寄存器	偏移量	R/W	描述	复位后的值
TEMPCR	GCR_BA+0x1C	R/W	Temperature Sensor Control Register	0x0000_0000

31	30	29	28	27	26	25	24
预留							
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							VTEMP_EN

Bits	描述	
[31:1]	保留	保留
[0]	VTEMP_EN	使能温度传感器 这个bit 用来使能/禁止温度传感器功能. 1 = 使能温度传感器功能 0 = 禁止温度传感器功能(默认) 这个比特设为1之后，ADC通道7可以选择温度传感器输入，温度值可以在ADC的转换结果中读到. 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	偏移量	R/W	描述	复位后的值
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
保留							
23	22	21	20	19	18	17	16
保留							
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]							

Bits	描述	
[31:16]	保留	保留
[15:0]	POR_DIS_CODE	上电复位使能控制（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再次发出复位信号。如果将POR_DIS_CODE 设置为0x5AA5，POR复位功能将被禁用，直到电源电压很低，导致POR_DIS_CODE 设置为其他值，或者由芯片的其他复位功能引起复位时POR功能重新有效，这些复位功能包括： /RESET引脚复位，看门狗，LVR复位，BOD复位，ICE复位命令和软件复位. 该寄存器是写保护的寄存器，写该位需要先向地址0x5000_0100依次写入“59h”, “16h”, “88h” 解除寄存器保护。参考寄存器REGWRPROT的设置，其地址为GCR_BA + 0x100.

Port0多功能控制寄存器 (P0_MFP)

寄存器	偏移量	R/W	描述	复位后的值
P0_MFP	GCR_BA+30	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
保留							
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]							

Bits	描述	
[31:24]	保留	保留
[23:16]	**P0_TYPEn**	**使能P0[7:0]史密特触发输入功能** 1= 使能P0[7:0] I/O史密特触发输入功能 0= 禁用P0[7:0] I/O史密特触发功能输入
[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 的功能 0 | 0 | P0.7 0 | 1 | AD7(EBI) 1 | 0 | SPICLK1(SPI1) | 1 | 保留

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td>[14]</td><td>P0_ ALT[6]</td><td>P0.6复用功能选择
P0.6的功能取决于P0_MFP[6] 和 P0_ALT[6].
<table>
<tr><th>P0_ALT[6]</th><th>P0_MFP[6]</th><th>P0.6 的功能</th></tr>
<tr><td>0</td><td>0</td><td>P0.6</td></tr>
<tr><td>0</td><td>1</td><td>AD6(EBI)</td></tr>
<tr><td></td><td>0</td><td>MISO_1(SPI1)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[13]</td><td>P0_ ALT[5]</td><td>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 的功能</th></tr>
<tr><td>0</td><td>0</td><td>P0.5</td></tr>
<tr><td>0</td><td>1</td><td>AD5(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>MOSI_1(SPI1)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[12]</td><td>P0_ ALT[4]</td><td>P0.4复用功能选择
of 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 的功能</th></tr>
<tr><td>0</td><td>0</td><td>P0.4</td></tr>
<tr><td>0</td><td>1</td><td>AD4(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>SPISS1(SPI1)</td></tr>
<tr><td></td><td>1</td><td>保留</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td>[11]</td><td>P0_ ALT[3]</td><td>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的功能</th></tr>
<tr><td>0</td><td>0</td><td>P0.3</td></tr>
<tr><td>0</td><td>1</td><td>AD3(EB)</td></tr>
<tr><td>1</td><td>0</td><td>RTS0(UART0)</td></tr>
<tr><td></td><td>1</td><td>RXD</td></tr>
</table></td></tr>
<tr><td>[10]</td><td>P0_ ALT[2]</td><td>P0.2复用功能选择
T 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的功能</th></tr>
<tr><td>0</td><td>0</td><td>P0.2</td></tr>
<tr><td>0</td><td>1</td><td>D2 EBI)</td></tr>
<tr><td>1</td><td>0</td><td>CTS0(UART0)</td></tr>
<tr><td>1</td><td>1</td><td>TXD</td></tr>
</table></td></tr>
<tr><td>[9]</td><td>P0_ ALT[1]</td><td>P0.1复用功能选择
P0.1 的功能取决于P0_MFP[1] 和 P0_ALT[1].
<table>
<tr><th>P0_ALT[1]</th><th>P0_MFP[1]</th><th>P0.1的功能</th></tr>
<tr><td>0</td><td>0</td><td>P0.1</td></tr>
<tr><td>0</td><td>1</td><td>A 1(BI)</td></tr>
<tr><td>1</td><td>0</td><td>RTS1(UART1)</td></tr>
<tr><td>1</td><td>1</td><td>RXD1</td></tr>
</table></td></tr>
<tr><td>[8]</td><td>P0_ ALT[0]</td><td>P0.0复用功能选择
P0.0的功能取决于 P0_MFP[0] 和 P0_ALT[0].
<table>
<tr><th>P0_ALT[0]</th><th>P0_MF [0</th><th>P0.0的功能</th></tr>
<tr><td>0</td><td>0</td><td>P0.0</td></tr>
<tr><td>0</td><td>1</td><td>AD0(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>CTS1(UART1)</td></tr>
<tr><td>1</td><td>1</td><td>TXD1</td></tr>
</table></td></tr>
</table>

Bits	描述	
[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的详细描述

Port1多功能控制寄存器 (P1_MFP)

寄存器	偏移量	R/W	描述	复位后的值
P1_MFP	GCR_BA+34	R/W	P1复用功能与输入类型控制寄存器	0x0000_0000

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

Bits	描述	
[31:24]	保留	保留
[23:16]	P1_TYPEn	**使能P1[7:0] 输入史密特触发功能** 1= 使能P1[7:0] I/O输入史密特触发功能 0= 禁用P1[7:0] I/O输入史密特触发功能
[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_MFP[7] | P1.7 的功能 0 | 0 | P1. 0 | 1 | AIN7(AD) 1 | 0 | SPICLK0(SPI0) 1 | 1 | 保留

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td>[14]</td><td>P1_ ALT[6]</td><td>P1.6复用功能选择
P1.6 的功能取决于 P1_MFP[6] 和 P1_ALT[6].
<table>
<tr><th>P1_ALT[6]</th><th>P1_MFP[6]</th><th>P1.6 的功能</th></tr>
<tr><td>0</td><td>0</td><td>P1.6</td></tr>
<tr><td>0</td><td>1</td><td>AIN6(ADC)</td></tr>
<tr><td>1</td><td>0</td><td>MISO_0(SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[13]</td><td>P1_ ALT[5]</td><td>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_A T[5]</th><th>P1_MFP[5</th><th>P1.5 的功能</th></tr>
<tr><td>0</td><td>0</td><td>P1.5</td></tr>
<tr><td>0</td><td>1</td><td>AIN5(A C)</td></tr>
<tr><td>1</td><td>0</td><td>MOSI_0(SPI0)</td></tr>
<tr><td>1</td><td>1</td><td>CPP0</td></tr>
</table></td></tr>
<tr><td>[12]</td><td>P1_ ALT[4]</td><td>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的功能</th></tr>
<tr><td>0</td><td>0</td><td>P1.4</td></tr>
<tr><td></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>CPN0</td></tr>
</table></td></tr>
<tr><td>[11]</td><td>P1_ ALT[3]</td><td>P1.3 复用功能选择
P1.3的功能取决于P1_MFP[3] and P1_ALT[3].
<table>
<tr><th>P1_ALT[3]</th><th>P1_MFP[3]</th><th>P1.3的功能</th></tr>
<tr><td>0</td><td>0</td><td>P1 3</td></tr>
<tr><td></td><td>1</td><td>AIN3(ADC)</td></tr>
<tr><td>1</td><td>0</td><td>TXD1(UART1)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td>[10]</td><td>P1_ ALT[2]</td><td>P1.2 复用功能选择
P1.2的功能取决于P1_MFP[2] and P1_ALT[2].
<table>
<tr><th>1_ LT[2]</th><th>P1_MFP[2]</th><th>P1.2的功能</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>
<tr><td>1</td><td>0</td><td>RXD1(UART1)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[9]</td><td>P1_ ALT[1]</td><td>P1.1 复用功能选择
P1.1的功能取决于P1_MFP[1] and P1_ALT[1].
<table>
<tr><th>1_ALT[1]</th><th>P1_MFP[1]</th><th>P1.1的功能</th></tr>
<tr><td>0</td><td>0</td><td>P1.1</td></tr>
<tr><td></td><td>1</td><td>AIN1(ADC)</td></tr>
<tr><td>1</td><td>0</td><td>T (Timer3)</td></tr>
<tr><td>1</td><td>1</td><td>nWRH</td></tr>
</table></td></tr>
<tr><td>[8]</td><td>P1_ ALT[0]</td><td>P1.0 复用功能选择
P1.0的功能取决于P1_MFP[0] and P1_ALT[0].
<table>
<tr><th>P1_ALT[0]</th><th>P1_MFP[0]</th><th>P1.0的功能</th></tr>
<tr><td>0</td><td>0</td><td>P1.0</td></tr>
<tr><td>0</td><td>1</td><td>AIN0(ADC)</td></tr>
<tr><td>1</td><td>0</td><td>T2 (Timer2)</td></tr>
<tr><td>1</td><td>1</td><td>nWRL</td></tr>
</table></td></tr>
<tr><td>[7:0]</td><td>P1_MFP[7:0]</td><td>P1 复用功能选择
P1的功能取决于P1_MFP 和 P1_ALT.
参考P1_ALT的详细描述</td></tr>
</table>

Port2多功能控制寄存器(P2_MFP)

寄存器	偏移量	R/W	描述	复位后的值
P2_MFP	GCR_BA+38	R/W	P2复用功能与输入类型控制寄存器	0x0000_0000

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

Bits	描述	
[31:24]	保留	保留
[23:16]	P2_TYPEn	**P2[7:0] 输入史密特触发功能使能** 1= 使能P2[7:0] I/O输入史密特触发功能 0= 禁用P2[7:0] I/O输入史密特触发功能
[15]	P2_ ALT[7]	P2.7 复用功能选择 P2.7的功能取决于P2_MFP[7] and P2_ALT[7]. P2_ALT[7] | P2_MFP[7] | P2.7 的功能 | 0 | P2.7 0 | 1 | AD15(EBI) 1 | 0 | PWM7(PWM generator 6) 1 | 1 | 保留

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td>[14]</td><td>P2_ ALT[6]</td><td>P2.6 复用功能选择
P2.6的功能取决于P2_MFP[6] and P2_ALT[6].
<table>
<tr><th>P2_ALT[6]</th><th>2_ FP[6]</th><th>P2.6 的功能</th></tr>
<tr><td>0</td><td>0</td><td>P .6</td></tr>
<tr><td>0</td><td>1</td><td>AD14(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>PWM6(PWM generator 6)</td></tr>
<tr><td>1</td><td>1</td><td>CPO1</td></tr>
</table></td></tr>
<tr><td>[13]</td><td>P2_ ALT[5]</td><td>P2.5 复用功能选择
P2.5的功能取决于P2_MFP[5] and P2_ALT[5].
<table>
<tr><th>P2_ALT[5]</th><th>P2_MFP[5]</th><th>P2.5 的功能</th></tr>
<tr><td>0</td><td>0</td><td>P2.5</td></tr>
<tr><td>0</td><td>1</td><td>AD13(EBI)</td></tr>
<tr><td></td><td>0</td><td>PWM5(PWM generator 4)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[12]</td><td>P2_ ALT[4]</td><td>P2.4 复用功能选择
P2.4的功能取决于P2_MFP[4] and P2_ALT[4].
<table>
<tr><th>P2_ALT[4]</th><th>P2_MFP[4]</th><th>P2.4的功能</th></tr>
<tr><td>0</td><td>0</td><td>P2.4</td></tr>
<tr><td>0</td><td>1</td><td>AD12 EBI)</td></tr>
<tr><td>1</td><td>0</td><td>PWM4(PWM generator 4)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[11]</td><td>P2_ ALT[3]</td><td>P2.3 复用功能选择
P2.3的功能取决于P2_MFP[3] and P2_ALT[3].
<table>
<tr><th>P2_ALT[3]</th><th>P2_MFP[3]</th><th>P2.3的功能</th></tr>
<tr><td>0</td><td>0</td><td>P2.3</td></tr>
<tr><td>0</td><td>1</td><td>AD11(EB)</td></tr>
<tr><td>1</td><td>0</td><td>PWM3(PWM generator 2)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td>[10]</td><td>P2_ ALT[2]</td><td>P2.2 复用功能选择
P2.2的功能取决于P2_MFP[2] and P2_ALT[2].
<table>
<tr><th>P2_ALT[2]</th><th>P2_ FP[2]</th><th>P2.2的功能</th></tr>
<tr><td>0</td><td>0</td><td>P2.2</td></tr>
<tr><td></td><td>1</td><td>AD10 EBI)</td></tr>
<tr><td>1</td><td>0</td><td>PWM2(PWM generator 2)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[9]</td><td>P2_ ALT[1]</td><td>P2.1 复用功能选择
P2.1的功能取决于P2_MFP[1] and P2_ALT[1].
<table>
<tr><th>P2_ALT[1]</th><th>P2_MFP[1]</th><th>P2.1的功能</th></tr>
<tr><td>0</td><td>0</td><td>P2.1</td></tr>
<tr><td>0</td><td>1</td><td>AD9(EBI)</td></tr>
<tr><td>1</td><td>0</td><td>PWM1(PWM generator 0)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[8]</td><td>P2_ ALT[0]</td><td>P2.0 复用功能选择
P2.0的功能取决于P2_MFP[0] and P2_ALT[0].
<table>
<tr><th>P2_ALT[0]</th><th>P2_MFP[0]</th><th>P2.0的功能</th></tr>
<tr><td>0</td><td></td><td>P2.0</td></tr>
<tr><td>0</td><td>1</td><td>AD (EBI)</td></tr>
<tr><td>1</td><td>0</td><td>PWM0(PWM generator 0)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[7:0]</td><td>P2_MFP[7:0]</td><td>P2 复用功能选择
P2的功能取决于P2_MFP 和 P2_ALT.
参考P2_ALT的详细描述.</td></tr>
</table>

Port3多功能控制寄存器(P3_MFP)

寄存器	偏移量	R/W	描述	复位后的值
P3_MFP	GCR_BA+3C	R/W	P3复用功能与输入类型控制寄存器	0x0000_0000

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

Bits	描述	
[31:24]	保留	保留
[23:16]	P3_TYPEn	**使能P3[7:0] 输入史密特触发功能** 1= 使能P3[7:0] I/O输入史密特触发功能 0= 禁用P3[7:0] I/O输入史密特触发功能
[15]	P3_ ALT[7]	P3.7 复用功能选择 P3.7的功能取决于P3_MFP[7] and P3_ALT[7]. P3_ALT[7] | P3_MFP[7] | P3.7 的功能 0 | 0 | P3 7 0 | 1 | RD(EBI) 1 | x | 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[ ] | P3_MFP[6] | P3.6 的功能 0 | 0 | P3.6 0 | 1 | WR(EBI) 1 | | CKO(lock Driver output) 1 | 1 | CPO0

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td>[13]</td><td>P3_ ALT[5]</td><td>P3.5 复用功能选择
P3.5的功能取决于P3_MFP[5] 和 P3_ALT[5].
<table>
<tr><th>P3_ALT[5</th><th>P _MFP[5]</th><th>P3.5 的功能</th></tr>
<tr><td>0</td><td>0</td><td>P3.5</td></tr>
<tr><td>0</td><td>1</td><td>T1(Timer1)</td></tr>
<tr><td>1</td><td>0</td><td>SCL(I2C)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[12]</td><td>P3_ ALT[4]</td><td>P3.4 复用功能选择
P3.4的功能取决于P3_MFP[4] 和 P3_ALT[4].
<table>
<tr><th>P3_ALT[4]</th><th>3_MFP[4</th><th>P3.4的功能</th></tr>
<tr><td>0</td><td>0</td><td>P3.4</td></tr>
<tr><td>0</td><td>1</td><td>T0(Timer0)</td></tr>
<tr><td>1</td><td></td><td>SDA(I2C)</td></tr>
<tr><td></td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[11]</td><td>P3_ ALT[3]</td><td>P3.3 复用功能选择
P3.3的功能取决于P3_MFP[3] 和 P3_ALT[3].
<table>
<tr><th>P3_A T[3]</th><th>P3_MFP[3]</th><th>P3.3的功能</th></tr>
<tr><td>0</td><td>0</td><td>P3.3</td></tr>
<tr><td>0</td><td>1</td><td>/INT1</td></tr>
<tr><td>1</td><td>0</td><td>MCLK(EBI)</td></tr>
<tr><td>1</td><td>x</td><td>T1EX</td></tr>
</table></td></tr>
<tr><td>[10]</td><td>P3_ ALT[2]</td><td>P3.2 复用功能选择
P3.2的功能取决于P3_MFP[2] and P3_ALT[2].
<table>
<tr><th>P3_ALT[2]</th><th>P3_MFP[2]</th><th>P3.2的功能</th></tr>
<tr><td>0</td><td>0</td><td>P3.2</td></tr>
<tr><td>0</td><td></td><td>/INT0</td></tr>
<tr><td>1</td><td>0</td><td>T0EX</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td>[9]</td><td>P3_ ALT[1]</td><td>P3.1 复用功能选择
P3.1的功能取决于P3_MFP[1] and P3_ALT[1].
<table>
<tr><th>P3_ALT[1]</th><th>P3_MFP[1]</th><th>P3.1的功能</th></tr>
<tr><td>0</td><td>0</td><td>P3.1</td></tr>
<tr><td>0</td><td>1</td><td>TXD(UART0)</td></tr>
<tr><td>1</td><td>0</td><td>CPP1(CMP)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[8]</td><td>P3_ ALT[0]</td><td>P3.0 复用功能选择
P3.0的功能取决于P3_MFP[0] and P3_ALT[0].
<table>
<tr><th>P3_ALT[0]</th><th>P3_MFP[0</th><th>P3.0的功能</th></tr>
<tr><td>0</td><td>0</td><td>P3.0</td></tr>
<tr><td>0</td><td>1</td><td>RXD(UART0)</td></tr>
<tr><td>1</td><td>0</td><td>CPN1(CMP)</td></tr>
<tr><td>1</td><td>1</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[7:0]</td><td>P3_MFP[7:0]</td><td>P3 复用功能选择
P3的功能取决于P3_MFP 和 P3_ALT.
参考 P3_ALT的详细描述.</td></tr>
</table>

Port4多功能控制寄存器(P4_MFP)

寄存器	偏移量	R/W	描述	复位后的值
P4_MFP	GCR_BA+40	R/W	P4复用功能与输入类型控制寄存器	0x0000_00C0

31	30	29	28	27	26	25	24
保留							
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]							

Bits	描述	
[31:24]	保留	保留
[23:16]	P4_TYPEn	**使能P4[7:0] 输入史密特触发功能** 1= 使能P4[7:0] I/O输入史密特触发功能 0= 禁用P4[7:0] I/O输入史密特触发功能
[15]	P4_ ALT[7]	P4.7 复用功能选择 P4.7的功能取决于P4_MFP[7] and P4_ALT[7]. P4_ALT[7] | P4_MFP 7] | P4.7 的功能 0 | 0 | P4.7 0 | 1 | ICE_DAT(ICE) 1 | x | 保留
[14]	P4_ ALT[6]	P4.6 复用功能选择 P4.6的功能取决于P4_MFP[6] and P4_ALT[6]. P4 ALT[6] | P4_ FP[6] | P4.6 的功能 0 | 0 | P4.6 0 | 1 | ICE_CLK(ICE) 1 | x | 保留

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td>[13]</td><td>P4_ ALT[5]</td><td>P4.5 复用功能选择
P4.5的功能取决于P4_MFP[5] and P4_ALT[5].
<table>
<tr><th>P4_ALT 5]</th><th>P4_MFP[5]</th><th>P4.5 的功能</th></tr>
<tr><td>0</td><td>0</td><td>P4.5</td></tr>
<tr><td>0</td><td>1</td><td>ALE(EBI)</td></tr>
<tr><td>1</td><td>x</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[12]</td><td>P4_ ALT[4]</td><td>P4.4 复用功能选择
P4.4的功能取决于P4_MFP[4] and P4_ALT[4].
<table>
<tr><th>P4_ALT[ ]</th><th>P4_MFP[4]</th><th>P4.4的功能</th></tr>
<tr><td>0</td><td>0</td><td>P4.4</td></tr>
<tr><td>0</td><td>1</td><td>/CS(EBI)</td></tr>
<tr><td>1</td><td>x</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[11]</td><td>P4_ ALT[3]</td><td>P4.3 复用功能选择
P4.3的功能取决于P4_MFP[3] and P4_ALT[3].
<table>
<tr><th>P4_AL [3</th><th>P4_MFP[3]</th><th>P4.3的功能</th></tr>
<tr><td>0</td><td>0</td><td>P4.3</td></tr>
<tr><td>0</td><td>1</td><td>PWM3(PWM generator 2)</td></tr>
<tr><td>1</td><td>x</td><td>保留</td></tr>
</table></td></tr>
<tr><td>[10]</td><td>P4_ ALT[2]</td><td>P4.2 复用功能选择
P4.2的功能取决于P4_MFP[2] and P4_ALT[2].
<table>
<tr><th>P4_ALT[2</th><th>P4_MFP[2]</th><th>P4.2的功能</th></tr>
<tr><td>0</td><td>0</td><td>P4.2</td></tr>
<tr><td>0</td><td>1</td><td>PWM2(PWM generator 2)</td></tr>
<tr><td>1</td><td></td><td>保留</td></tr>
</table></td></tr>
</table>

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td>[9]</td><td>P4_ ALT[1]</td><td>P4.1 复用功能选择
P4.1的功能取决于P4_MFP[1] and P4_ALT[1].
<table>
<tr><th>P4_AL [1]</th><th>P4_MFP[1]</th><th>P4.1的功能</th></tr>
<tr><td>0</td><td>0</td><td>P4.1</td></tr>
<tr><td>0</td><td>1</td><td>PWM1(PWM generator 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></td></tr>
<tr><td>[8]</td><td>P4_ ALT[0]</td><td>P4.0 复用功能选择
P4.0的功能取决于P4_MFP[0] and P4_ALT[0].
<table>
<tr><th>P4_ALT[0]</th><th>P4_MFP[0]</th><th>P4.0的功能</th></tr>
<tr><td>0</td><td>0</td><td>P .</td></tr>
<tr><td>0</td><td>1</td><td>PWM0(PWM generator 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></td></tr>
<tr><td>[7:0]</td><td>P4_MFP[7:0]</td><td>P4 复用功能选择
P4的功能取决于P4_MFP 和 P4_ALT.
参考P4_ALT 的详细描述.</td></tr>
</table>

寄存器写保护控制寄存器(REGWRPROT)

有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后是被保护的，直到用户关闭寄存器保护。用户如果要编程这些被保护的寄存器，需要写一个解锁序列到REGWRPROT寄存器。解锁序列就是连续写“59h”，“16h”，“88h”到REGWRPROT寄存器，地址是0x5000_010。在写这三个数据过程中，任何不同的数据值，不同的顺序或者任何对其他地址的写，都会中止解锁序列。

保护被关闭之后，用户可以检查保护禁止位（地址0x5000_0100的比特0），"1"表示保护关闭，"0"表示保护使能。然后用户可以更新被保护的目标寄存器的值，并随时可以通过写任何数据到地址"0x5000_0100"来使能寄存器保护。

写该寄存器来关闭/使能寄存器保护，读该寄存器来获取REGPROTDIS状态。

寄存器	偏移量	R/W	描述	复位后的值
REGWRPROT	GCR_BA+100	R/W	寄存器写保护控制寄存器	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**
REGWRPROT[7:1]							**REGWRPROT [0]** **REGPROTDIS**

Bits	描述	
[31:16]	保留	保留
[7:0]	**REGWRPROT**	**寄存器写保护码**(只写) 编程写保护寄存器之前必须通过一个解锁序列来移除写保护功能，解锁序列就是依次写“59h”, “16h”, “88h” 到这个域，这个时序完成之后, REGPROTDIS 位将被置1，写保护寄存器可以正常写入数据.
[0]	**REGPROTDIS**	**寄存器写保护**(只读) 1 = 解除写保护，写保护寄存器可以正常编程 0 =使能寄存器的写保护，不能向受保护寄存器写入数据. 受保护的寄存器有:

Bits	描述			
		Registers	**Addr ss**	**Note**
		IPRSTC1	0x5000_0008	
		BODCR	0x5000_0018	
		PORCR	0x5000_0024	
		PWRCON	0x5000_0200	在清除电源唤醒中断时bit[6]不被保护
		APBCLK bit[0]	0x5000_0208	bit[0]：看门狗时钟使能
		CLKSEL0	0x5000_0210	HCLK和CPU STCLK时钟源选择
		CLK_SEL1 bit[1:0]	0x5000_0214	看门狗时钟源选择
		NMI_SEL bit[8]	0x5000_0380	NMI 中断使能
		ISPCON	0x5000_C000	Flash在系统（ISP）编程控制寄存器
		ISPTRG	0x5000_C010	
		WTCR	0x4000_4000	看门狗控制寄存器
		FATCON	0x5000_C018	

6.2.7 系统定时器(SysTick)

Cortex-M0 包含一个集成的系统定时器, SysTick. SysTick 提供一种简单，24位写清零，下数计数，计数至0后自动重装载的计数器，有一个灵活的控制机制。计数器可作为实时操作系统的节拍定时器或者一个简单的计数器。

使能后，系统定时器从SysTick 当前值寄存器(SYST_CVR)的值向下计数到0，并在下一个时钟边沿，重新加载SysTick重装载值寄存器(SYST_RVR)的值到SysTick当前值寄存器(SYST_CVR) ，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，读系统定时器的控制与状态寄存器（SYST_CSR）将清零标志位COUNTFLAG。

复位后，SYST_CVR 的值未知。使能前，软件应该向寄存器写入0。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” 与 “ARM® v6-M Architecture Reference Manual”.

6.2.7.1 *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	偏移量	R/W	描述	复位后的值
SCS_BA = 0xE000_E000				
SYST_CSR	SCS_BA + 010	R/W	SysTick控制与状态寄存器	0x0000_0004
SYST_RVR	SCS_BA + 014	R/W	SysTick重装载值寄存器	0xXXXX_XXXX
SYST_CVR	SCS_BA + 018	R/W	SysTick 当前值寄存器	0xXXXX_XXXX

SysTick 控制与状态（SYST_CSR）

寄存器	偏移量	R/W	描述	复位后的值
SYST_CSR	SCS_BA+0x10	R/W	SysTick控制与状态	0x0000_0004

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

Bits	描述	
[31:17]	保留	保留
[16]	**COUNTFLAG**	从上次该寄存器读取，如果定时器计数到0，则返回1. 计数由1到0时，COUNTFLAG 置位. 在读该位或向系统定时器当前值寄存器（SYST_CVR）写时，COUNTFLAG 被清零。
[15:3]	保留	保留
[2]	**CLKSRC**	1= 内核时钟用于SysTick. 0= 时钟源可选, 参考STCLK_S.
[1]	**TICKINT**	1 : 向下计数到0将引起SysTick 异常而挂起。清SysTick 当前值寄存器的值将不会导致SysTick 挂起. 0 : 向下计数到0不会引起SysTick异常而挂起。软件通过设置COUNTFLAG 来确定是否已经发生计数到0
[0]	**ENABLE**	1 : 计数器开始计数，并运行于周期模式. 0 : 禁用计数器

SysTick重装载值寄存器（SYST_RVR）

寄存器	偏移量	R/W	描述	复位后的值
SYST_RVR	SCS_BA + 014	R/W	SysTick重装载值寄存器	0xXXXX_XXXX

31	30	29	28	27	26	25	24
保留							
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]							

Bits	描述	
[31:24]	保留	保留
[23:0]	**RELOAD**	当计数器达到0时，值加载到当前值寄存器.

SysTick当前值寄存器（SYST_CVR）

寄存器	偏移量	R/W	描述	复位后的值
SYST_CVR	SCS _BA + 018	R/W	SysTick当前值寄存器	0xXXXX_XXXX

31	30	29	28	27	26	25	24
保留							
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]							

Bits	描述	
[31:24]	保留	保留
[23:0]	CURRENT	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
- 自动保存和恢复上下文
- 可动态改变优先级
- 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的起始地址可从内存的向量表中取得。不需要确定哪个中断被响应，也不要软件分配相关中断服务程序（ISR）的起始地址。当起始地址取得时，NVIC将自动保存处理状态，包括以下寄存器“PC, PSR, LR, R0~R3, R12” 的值到栈中。 在ISR结束时, NVIC 将从栈中恢复相关寄存器的值，恢复正常操作， 因此处理器将花费更少的确定的时间去处理中断请求。

NVIC支持末尾连锁“Tail Chaining” ，有效处理背对背中断“back-to-back interrupts”， 即无需保存和恢复当前状态从而减少从当前ISR结束切换到挂起的ISR的延迟时间。NVIC还支持晚到“Late Arrival”，改善同时发生的ISR的效率。当较高优先级中断请求发生在当前ISR开始执行之前（保存处理器状态和获取起始地址阶段），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” 与 “ARM® v6-M Architecture Reference Manual”.

6.2.8.1 *异常模式和系统中断映射*

表6.2.2列出了NuMicro M051™系列支持的异常模式。 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：优先级为“0”在整个系统中为第4优先级，排在“Reset”, “NMI” 与“Hard Fault”之后。

异常号	向量地址	中断号 (Bit in Interrupt Registers)	中断名	源IP	中断描述	掉电唤醒
1-15	0x00-0x3C	-	-	-	系统异常	
16	0x40	0	BOD_OUT	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 脚上的外部信号中断	Yes
19	0x4C	3	EINT1	GPIO	P3.3 脚上的外部信号中断	Yes
20	0x50	4	GP01_INT	GPIO	P0[7:0] / P1[7:0] 外部信号中断	Yes
21	0x54	5	GP234_INT	GPIO	P2[7:0]/P3[7:0]/P4[7:0] 外部信号中断，除 P32 和 P33	Yes
22	0x58	6	PWMA_INT	PWM0~3	PWM0, PWM1, PWM2 和 PWM3 中断	No
23	0x5C	7	PWMB_INT	PWM4~7	PWM4, PWM5, PWM6 和 PWM7 中断	No
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	UART1_INT	UART1	UART1中断	Yes
30	0x78	14	SPI0_INT	SPI0	SPI0中断	No
31	0x7C	15	SPI1_INT	SPI1	SPI1中断	No
32-33	0x80-0x84	16-17	-	-	-	-
34	0x88	18	I2C_INT	I2C	I2C中断	No
35-40	0x8C-0xAC	19-24	-	-	-	-
41	0xA4	25	ACMP_INT	ACMP	模拟比较器0和模拟比较器1的中断	Yes
42-43	0xA8-0xAC	26-27				

异常号	向量地址	中断号 (Bit in Interrupt Registers)	中断名	源IP	中断描述	掉电唤醒
44	0xB0	28	PWRWU_INT	CLKC	从掉电状态唤醒的时钟控制器中断	Yes
45	0xB4	29	ADC_INT	ADC	ADC 中断	No
46-47	0xB8-0xBC	30-31	-	-	-	

表 6.2-3 异常模式

异常名称	*向量号*	*优先级*
Reset	**1**	**-3**
NMI	**2**	**-2**
Hard Fault	**3**	**-1**
保留	**4 ~ 10**	*保留*
SVCall	**11**	可配置
保留	**12 ~ 13**	*保留*
PendSV	**14**	可配置
SysTick	**15**	可配置
Interrupt (IRQ0 ~ IRQ31)	**16 ~ 47**	可配置

表 6.2-4 系统中断映射

6.2.8.2 *向量表*

当任何中断被响应时，处理器会自动从内存的向量表中获取中断服务程序（ISR）的起始地址。对于ARMv6-M, 向量表的基地址固定在0x00000000。向量表包括复位后栈指针的初始值，所有异常处理函数的入口地址。在上一页定义的向量号定义向量表中与上一部分说明的异常处理函数入口相关的入口顺序。

向量表字偏移量	*描述*
0	SP_main -主堆栈指针
Vector Number	异常入口指针，用向量号表示

表 6.2-5 向量表格式

6.2.8.3 *操作描述*

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清零，读取这两种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挂起，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。清使能位可以阻止相关中断被再次激活。

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清楚的方式，读取这两种寄存器返回当前相应中断的挂起状态。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从内存系统控制空间的一块区域访问，下一节将作出描述.

6.2.8.4 *NVIC 控制寄存器*

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位后的值
SCS_BA = 0xE000_E000				
NVIC _ISER	SCS_BA + 100	R/W	IRQ0 ~ IRQ31 设置使能控制寄存器	0x0000_0000
NVIC_ICER	SCS_BA + 180	R/W	IRQ0 ~ IRQ31 清使能控制寄存器	0x0000_0000
NVIC_ISPR	SCS_BA + 200	R/W	IRQ0 ~ IRQ31 设置挂起控制寄存器	0x0000_0000
NVIC_ICPR	SCS_BA + 280	R/W	IRQ0 ~ IRQ31 清挂起控制寄存器	0x0000_0000
NVIC_IPR0	SCS_BA + 400	R/W	IRQ0 ~ IRQ3 优先级 控制寄存器	0x0000_0000
NVIC_IPR1	SCS_BA + 404	R/W	IRQ4 ~ IRQ7优先级控制寄存器	0x0000_0000
NVIC_IPR2	SCS_BA + 408	R/W	IRQ8 ~ IRQ11优先级控制寄存器	0x0000_0000
NVIC_IPR3	SCS_BA + 40C	R/W	IRQ12 ~ IRQ15优先级控制寄存器	0x0000_0000
NVIC_IPR4	SCS_BA + 410	R/W	IRQ16 ~ IRQ19优先级控制寄存器	0x0000_0000
NVIC_IPR5	SCS_BA + 414	R/W	IRQ20 ~ IRQ23优先级控制寄存器	0x0000_0000
NVIC_IPR6	SCS_BA + 418	R/W	IRQ24 ~ IRQ27优先级控制寄存器	0x0000_0000
NVIC_IPR7	SCS_BA + 41C	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	偏移量	R/W	描述	复位的值
NVIC_ISER	SCS _BA + 100	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]							

Bits	描述	
[31:0]	SETENA	使能1个或多个中断，每位代表从IRQ0 ~ IRQ31的中断号(向量号：16 ~ 47). 写1使能相关中断 写0无效 寄存器读取返回当前使能状态.

IRQ0 ~ IRQ31清使能控制寄存器（NVIC_ICER）

寄存器	偏移量	R/W	描述	复位后的值
NVIC_ICER	SCS _BA + 180	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]							

Bits	描述	
[31:0]	**CLRENA**	禁用1个或多个中断，每位代表从IRQ0 ~ IRQ31的中断号 (向量号： 16 ~ 47). 写1禁用相应中断 写0无效 寄存器读取返回当前使能状态.

IRQ0 ~ IRQ31设置挂起控制寄存器（NVIC_ISPR）

寄存器	偏移量	R/W	描述	复位后的值
NVIC_ISPR	SCS _BA + 200	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]							

Bits	描述	
[31:0]	**SETPEND**	软件写1，挂起相应中断。每位代表从IRQ0 ~ IRQ31 的中断号(向量号： 16 ~ 47). 写0无效 寄存器读取返回当前挂起状态

IRQ0 ~ IRQ31清挂起控制寄存器（NVIC_ICPR）

寄存器	偏移量	R/W	描述	复位后的值
NVIC_ICPR	SCS _BA + 280	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]							

Bits	描述	
[31:0]	**CLRPEND**	写1清除相应中断挂起，每位代表从IRQ0 ~ IRQ31的中断号 (向量号： 16 ~ 47). 写0无效. 寄存器读取返回当前挂起状态.

IRQ0 ~ IRQ3中断优先级寄存器（NVIC_IPR0）

寄存器	偏移量	R/W	描述	复位后的值
NVIC_IPR0	SCS _BA + 400	R/W	IRQ0 ~ IRQ3中断优先级寄存器	0x0000_0000

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

Bits	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	偏移量	R/W	描述	复位后的值
NVIC_IPR1	SCS _BA + 404	R/W	IRQ4 ~ IRQ7中断优先级寄存器	0x0000_0000

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

Bits	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	偏移量	R/W	描述	复位后的值
NVIC_IPR2	SCS _BA + 408	R/W	IRQ8 ~ IRQ11中断优先级寄存器	0x0000_0000

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

Bits	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	偏移量	R/W	描述	复位后的值
NVIC_IPR3	SCS _BA + 40C	R/W	IRQ12 ~ IRQ15中断优先级寄存器	0x0000_0000

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

Bits	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	偏移量	R/W	描述	复位后的值
NVIC_IPR4	SCS _BA + 410	R/W	IRQ16 ~ IRQ19中断优先级寄存器	0x0000_0000

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

Bits	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	偏移量	R/W	描述	复位后的值
NVIC_IPR5	SCS _BA + 414	R/W	IRQ20 ~ IRQ23中断优先级寄存器	0x0000_0000

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

Bits	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	偏移量	R/W	描述	复位后的值
NVIC_IPR6	SCS _BA + 418	R/W	IRQ24 ~ IRQ27中断优先级寄存器	0x0000_0000

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

Bits	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	偏移量	R/W	描述	复位后的值
NVIC_IPR7	SCS _BA + 41C	R/W	IRQ28 ~ IRQ31中断优先级寄存器	0x0000_0000

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

Bits	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

6.2.8.5 *中断源控制寄存器*

除了与NVIC相关的中断控制寄存器外，NuMicro M051™系列还有一些特殊控制寄存器执行中断功能，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, W&C: 写1清零

寄存器	偏移量	R/W	描述	复位后的值
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	IRQ7 (PWMB) 中断源识别	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 (URT0) 中断源识别	0xXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (URT1) 中断源识别	0xXXXX_XXXX
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI0) 中断源识别	0xXXXX_XXXX
IRQ15_SRC	INT_BA+0x3C	R	IRQ15 (SPI1) 中断源识别	0xXXXX_XXXX
IRQ16_SRC	INT_BA+0x40	保留	保留	0xXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	保留	保留	0xXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (I2C) 中断源识别	0xXXXX_XXXX
IRQ19_SRC	INT_BA+0x4C	保留	保留	0xXXXX_XXXX
IRQ20_SRC	INT_BA+0x50	保留	保留	0xXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	保留	保留	0xXXXX_XXXX

寄存器	偏移量	R/W	描述	复位后的值
IRQ22_SRC	INT_BA+0x58	保留	保留	0xXXXX_XXXX
IRQ23_SRC	INT_BA+0x5C	保留	保留	0xXXXX_XXXX
IRQ24_SRC	INT_BA+0x60	保留	保留	0xXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	保留	保留	0xXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	保留	保留	0xXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	保留	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	保留	保留	0xXXXX_XXXX
IRQ31_SRC	INT_BA+0x7C	保留	保留	0xXXXX_XXXX
NMI_SEL	INT_BA+0x80	R/W	NMI中断源选择控制寄存器	0x0000_0000
MCU_IRQ	INT_BA+0x84	R/W	MCU IRQ号识别寄存器	0x0000_0000

中断源识别寄存器(IRQn_SRC)

寄存器	偏移量	R/W	描述	复位后的值
IRQn_SRC	INT_BA+0x00 …….. INT_BA+0x7C	R	MCU IRQ0 (BOD) 中断源识别 : MCU IRQ31 (保留) 中断源识别	0xXXXX_XXXX

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				**INT_SRC[3]**	**INT_SRC[2:0]**		

地址	INT-Num	Bits	描述
INT_BA+0x00	0	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: BOD_INT
INT_BA+0x04	1	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: WDT_INT
INT_BA+0x08	2	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: EINT0 –P3.2引脚上的外部中断0
INT_BA+0x0C	3	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: EINT1 - P3.3上的外部中断1
INT_BA+0x10	4	[2:0]	Bit2: 1'b0 Bit1: P1_INT Bit0: P0_INT

地址	INT-Num	Bits	描述
INT_BA+0x14	5	[2:0]	Bit2: P4_INT Bit1: P3_INT Bit0: P2_INT
INT_BA+0x18	6	[3:0]	Bit3: PWM3_INT Bit2: PWM2_INT Bit1: PWM1_INT Bit0: PWM0_INT
INT_BA+0x1C	7	[3:0]	Bit3: PWM7_INT Bit2: PWM6_INT Bit1: PWM5_INT Bit0: PWM4_INT
INT_BA+0x20	8	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: TMR0_INT
INT_BA+0x24	9	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: TMR1_INT
INT_BA+0x28	10	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: TMR2_INT
INT_BA+0x2C	11	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: TMR3_INT
INT_BA+0x30	12	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: URT0_INT
INT_BA+0x34	13	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: URT1_INT
INT_BA+0x38	14	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: SPI0_INT

地址	INT-Num	Bits	描述
INT_BA+0x3C	15	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: SPI1_INT
INT_BA+0x40	16	[2:0]	保留
INT_BA+0x44	17	[2:0]	保留
INT_BA+0x48	18	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: I2C_INT
INT_BA+0x4C	19	[2:0]	保留
INT_BA+0x50	20	[2:0]	保留
INT_BA+0x54	21	[2:0]	保留
INT_BA+0x58	22	[2:0]	保留
INT_BA+0x5C	23	[2:0]	保留
INT_BA+0x60	24	[2:0]	保留
INT_BA+0x64	25	[2:0]	保留
INT_BA+0x68	26	[2:0]	保留
INT_BA+0x6C	27	[2:0]	保留
INT_BA+0x70	28	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: PWRWU_INT
INT_BA+0x74	29	[2:0]	Bit2: 1'b0 Bit1: 1'b0 Bit0: ADC_INT
INT_BA+0x78	30	[2:0]	保留
INT_BA+0x7C	31	[2:0]	保留

NMI中断源选择控制寄存器(NMI_SEL)

寄存器	偏移量	R/W	描述	复位后的值
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
保留							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留							**NMI_EN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留			**NMI_SEL[4:0]**				

Bits	描述	
[31:9]	保留	保留
[8]	**NMI_EN**	NMI 中断使能 (写保护) 1 = 使能 NMI 中断 0 = 禁止 NMI 中断 这个比特是写保护的，意味着在写这个比特之前需要连续写“59h”，“16h”，“88h”到地址 0x5000_0100 来解锁.参考寄存器 REGWRPROT，地址 GCR_BA+0x100
[7:5]	保留	保留
[4:0]	**NMI_SEL**	Cortex-M0的NMI 中断源可以从interrupt[31:0]中选择一个 NMI_SEL bit[4:0] 用于选择NMI 中断源

MCU中断请求源寄存器(MCU_IRQ)

寄存器	偏移量	R/W	描述	复位后的值
MCU_IRQ	INT_BA+0x84	R/W	MCU中断请求源寄存器	0x0000_0000

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

Bits	描述	
[31:0]	MCU_IRQ	**MCU IRQ 源寄存器** MCU_IRQ 从外围设备收集所有中断，并同步对Cortex-M0产生中断。以下两种模式均可中断Cortex-M0，正常模式与测试模式. MCU_IRQ从每一个外设收集中断，同步他们，然后触发Cortex-M0中断。 MCU_IRQ[n] 是 “0”：置 MCU_IRQ[n] 为“1”，将向Cortex_M0 NVIC[n]触发一个中断. MCU_IRQ[n] 是 “1”: (意味着有中断请求等待处理) 此时写”1”到MCU_IRQ[n]将清除中断，写 “0” :无效.

6.2.9 系统控制器寄存器映射

Cortex-M0的状态和操作模式由系统控制寄存器控制，包括CPUID，Cortex-M0中断优先级和Cortex-M0电源管理都可以通过这些系统控制寄存器控制。

更多详情请参考 “ARM® Cortex™-M0 Technical Reference Manual” 与 “ARM® v6-M Architecture Reference Manual”.

R: 只读, W: 只写, R/W: 可读写, W&C: 写1清零

寄存器	偏移量	R/W	描述	复位后的值
SCS_BA = 0xE000_E000				
CPUID	SCS_BA + D00	R	CPUID Base 寄存器	0x0000_0000
ICSR	SCS_BA + D04	R/W	中断控制状态寄存器	0x0000_0000
AIRCR	SCS_BA+ 0xD0C	R/W	中断应用和复位控制寄存器	0xFA05_0000
SCR	SCS_BA + D10	R/W	系统控制寄存器	0x0000_0000
SHPR2	SCS_BA + D1C	R/W	系统处理器优先级寄存器2	0x0000_0000
SHPR3	SCS_BA + D20	R/W	系统处理器优先级寄存器3	0x0000_0000

CPUID Base寄存器(CPUID)

寄存器	偏移量	R/W	描述	复位后的值
CPUID	SCS_BA + D00	R	CPUID Base寄存器	0x 410CC200

31	30	29	28	27	26	25	24
IMPLEMENTER[7:0]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留				**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]**			

Bits	描述	
[31:24]	**IMPLEMENTER**	由ARM分配执行码. (ARM = 0x41)
[23:20]	保留	保留
[19:16]	**PART**	ARMv6-M 读取值为0xC
[15:4]	**PARTNO**	读取值为 0xC20.
[3:0]	**REVISION**	读取值为 0x0

中断控制状态寄存器(ICSR)

寄存器	偏移量	R/W	描述	复位后的值
ICSR	SCS_BA + D04	R/W	中断控制状态寄存器	0x 00000000

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

Bits	R/W	描述	
[31]	R/W	NMIPENDSET	触发NMI中断 写: 0 = 无作用 1 = 改变NMI异常状态为未处理. 读: 0 = 没有NMI 异常等待处理 1 = 有NMI异常等待处理 设置该位将激活NMI，由于NMI是最高优先级的异常，正常情况下，只要处理器检测到写'1'到这个位，将马上进入中断处理函数，然后这个位就会被自动清成'0'。这就意味着如果在中断处理函数中读这个位返回'1，就是NMI中断再次被置。
[30:29]		保留	保留

Bits	R/W	描述	
[28]	R/W	PENDSVSET	触发PendSV中断 写: 0 = 无作用 1 = 改变PendSV异常状态为未处理. 读: 0 = 没有PendSV异常等待处理 1 = 有PendSV异常等待处理 触发PendSV 中断。写”1”到这个位是唯一触发PendSV的方式。
[27]	W	PENDSVCLR	写1 清除PendSV 中断. 这个位是只写的，当要清除PendSV中断的时候，必须同时对PENDSVSET写”0”， PENDSVCLR写”1”
[26]	R/W	PENDSTSET	写”1”可以触发SysTick中断. 读可以得到SysTick中断当前状态 （1为有SysTick中断等待处理，0为没有SysTick中断等待处理）。
[25]	W	PENDSTCLR	写1清除等待处理的SysTick.中断
[23]	R	ISRPREEMPT	如果置位，当从调试状态退出时，未处理的异常将被处理.
[22]	R	ISRPENDING	有中断等待处理标志位，不包括NMI和HardFault中断 0 = 没有中断等待处理 1 = 有中断等待处理 这个位是只读的.
[21:18]		保留	保留
[17:12]	R	VECTPENDING	指示挂起的最高优先级异常号 0 = 没有中断等待处理 非0 = 等待处理的最高优先级异常号
[8:0]	R	VECTACTIVE	包含当前正在处理的异常号 0: 线程模式 非0: 当前正在处理的异常号

中断应用和复位控制寄存器(AIRCR)

寄存器	偏移量	R/W	描述	复位后的值
AIRCR	SCS_BA+0xD0C	R/W	中断应用和复位控制寄存器	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**
保留							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留		**SYSRESETREQ**		**VECTCLKACTIVE**		保留	

Bits	描述	
[31:16]	**VECTORKEY**	写该寄存器时，该域应该是0x05FA, 否则写动作结果不可预测。
[15:3]	保留	保留
[2]	**SYSRESETREQ**	该位写1，产生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**	该位置1，清除所有异常的有效状态. 该位只写，只有在内核挂起时可写. 注:重新初始化堆栈是调试器（debugger）的任务
[0]	保留	保留

系统控制寄存器(SCR)

寄存器	偏移量	R/W	描述	复位后的值
SCR	SCS_BA + D10	R/W	系统控制寄存器	0x 00000000

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			SEVONPEND	保留	SLEEPDEEP	SLEEPONEXIT	保留

Bits	描述	
[4]	SEVONPEND	当事件或者中断进入未处理状态时，将把CPU从WFE指令唤醒。如果处理器没有正在等待一个事件，这个事件将被注册，在下一个WFE指令时起作用.
[2]	SLEEPDEEP	控制处理器进入睡眠还是深度睡眠状态 0：睡眠 1：深度睡眠.
[1]	SLEEPONEXIT	当从Handler模式返回Thread模式时，是否进入睡眠模式 0： 返回Thread模式时，不要进入睡眠 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 设置这个比特为”1”，可以使能一个中断驱动的应用，避免从中断返回时进入一个空的主函数

系统处理器优先级寄存器2 (SHPR2)

寄存器	偏移量	R/W	描述	复位后的值
SHPR2	SCS_BA + D1C	R/W	系统处理器优先级寄存器2	0x 00000000

31	30	29	28	27	26	25	24
PRI_11		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							

Bits	描述	
[31:30]	**PRI_11**	**系统处理器的优先级11 – SVCall** “0” 表示最高优先级& “3” 表示最低优先级

系统处理器优先级寄存器3 (SHPR3)

寄存器	偏移量	R/W	描述	复位后的值
SHPR3	SCS_BA + D20	R/W	系统处理器优先级寄存器3	0x 00000000

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

Bits	描述	
[31:30]	**PRI_15**	**系统处理器优先级15 – SysTick** “0” 表示最高优先级 & “3” 表示最低优先级
[23:22]	**PRI_14**	**系统处理器优先级14 – PendSV** “0” 表示最高优先级 & “3” 表示最低优先级

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，包括系统时钟和所有外设时钟。时钟控制器还利用独立的时钟ON/OFF控制、时钟源选择和时钟分频器，实现电源控制功能。在CPU置位掉电使能位(PWR_DOWN_EN)且Cortex-M0执行WFI指令之后，芯片会进入掉电模式，在那之后，芯片等待唤醒中断源被触发以离开掉电模式。在掉电模式下，时钟控制器关闭外部高速晶振和内部22.1184MHz高速振荡器，以降低整个系统的功耗。

6.3.2 时钟发生器方块图

时钟发生器由如下4个时钟源组成:

- 一个外部 4~24 MHz 高速晶振
- 一个内部 22.1184 MHz RC高速振荡器
- 一个可编程的 PLL FOUT(PLL时钟源可以选择外部4~24MHz高速晶振和内部22.1184MHz高速振荡器)
- 一个内部 10KHz 低速振荡器

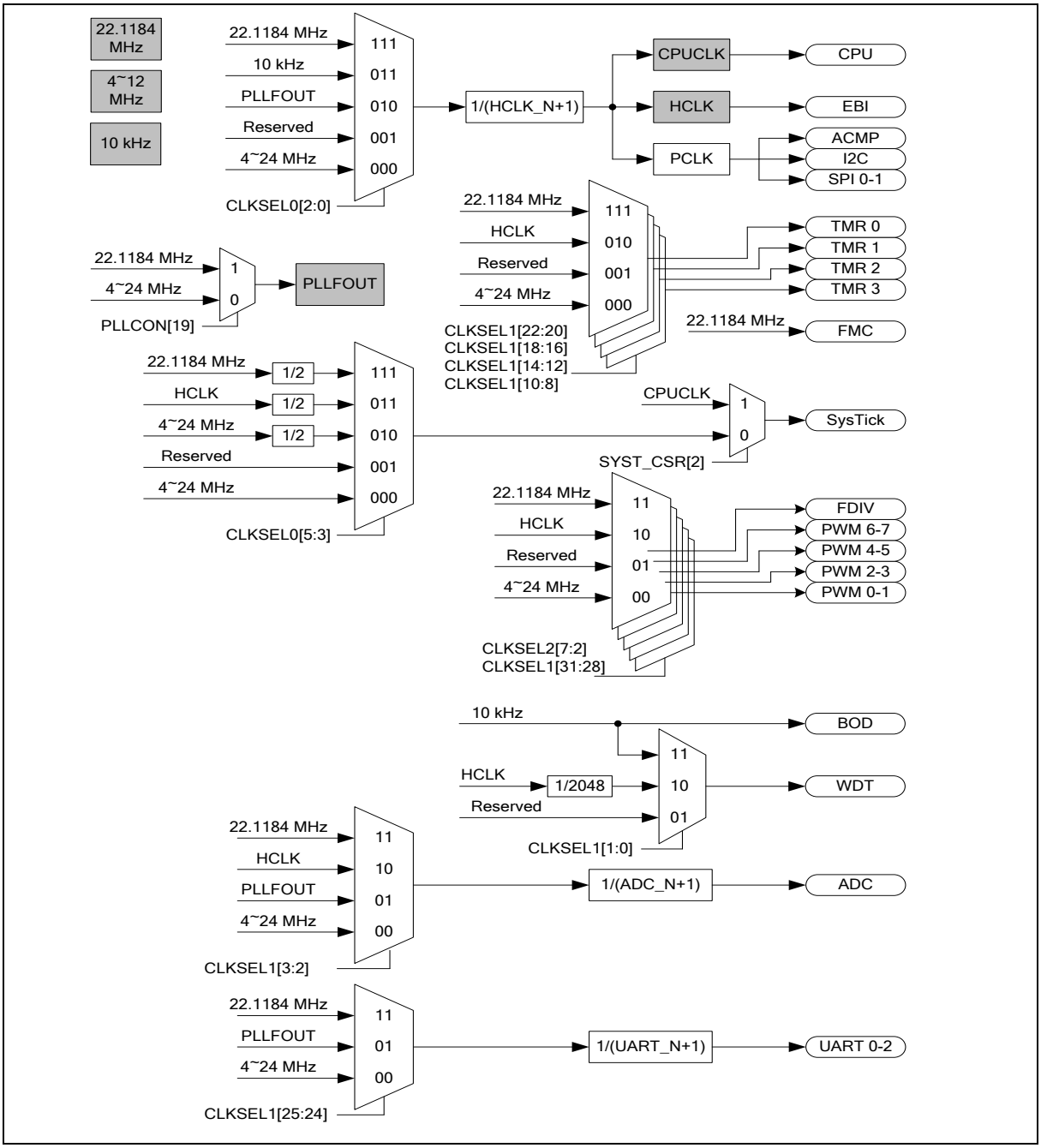


图6.3.2-1时钟发生器方块图

6.3.3 系统时钟 & SysTick 时钟

系统时钟有4个时钟源，由时钟发生器模块产生。使用寄存器HCLK_S(CLKSEL0[2:0])可以切换不同的时钟，系统时钟框图如下所示。

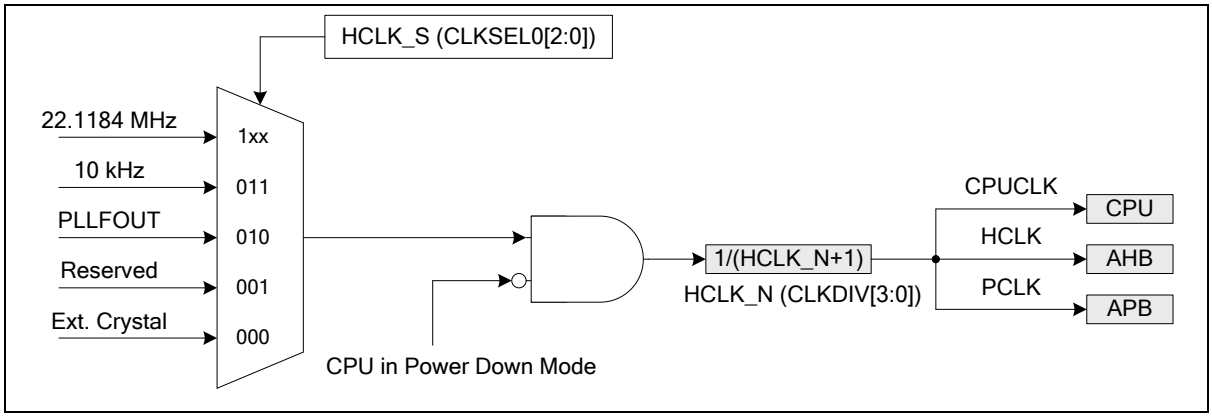


图6.3.3-1系统时钟框图

在Cortex-M0核中的SysTick的时钟源可以使用CPU时钟或者外部时钟(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]的设置。SysTick时钟框图在图6.3.3-2示出。

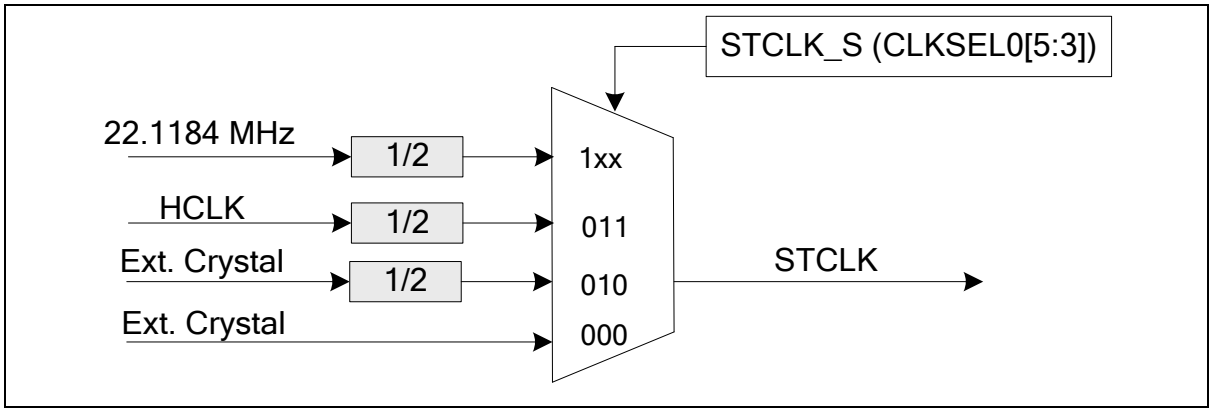


图6.3.3-2SysTick时钟控制框图

6.3.4 AHB 时钟源选择

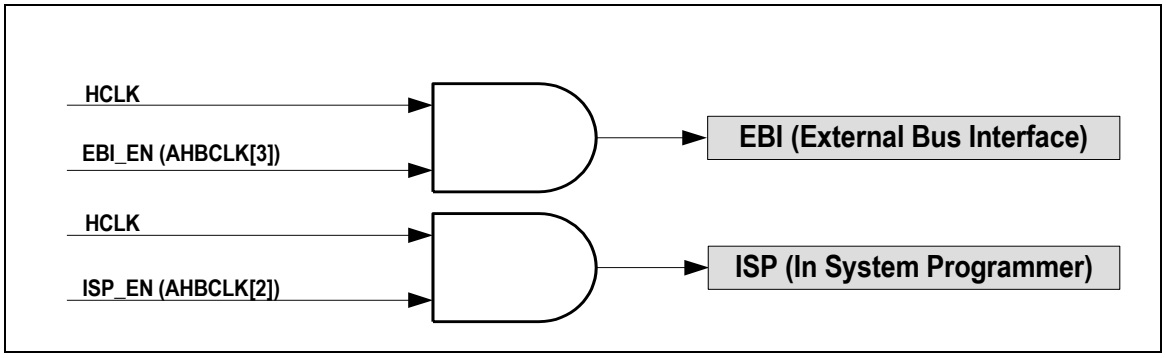


图6.3.4-1AHB 时钟源HCLK

6.3.5 外设时钟源选择

不同的外设，其时钟有不同的时钟源切换设置。请参考CLKSEL1 & APBCLK寄存器的描述。

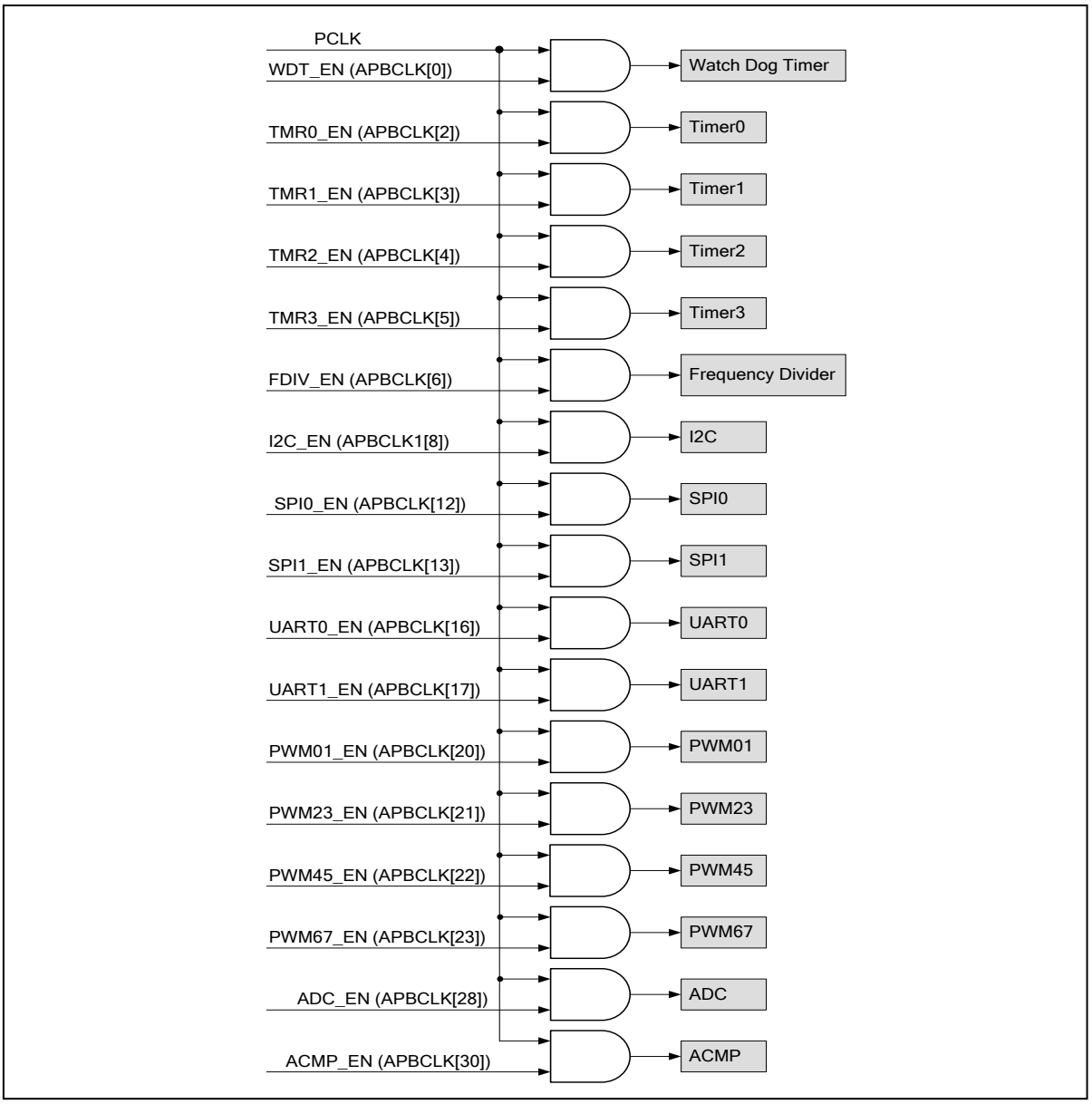


图6.3.5-1外设时钟源选择PCLK

6.3.6 掉电模式（深度睡眠模式） 时钟

当芯片进入掉电模式后，大部分时钟源、外设时钟和系统时钟将会被禁用，如果在CPU进入掉电模式之前没有关闭内部10K，它将保持有效，一些选择10K做时钟源的外设仍可以处于激活状态。

如下外设仍然可以保持激活：

- 外设时钟 (当这些IP采用内部10KHz低速振荡器作时钟源时)
  - 看门狗时钟
  - Timer 0/1/2/3 时钟
  - PWM 时钟

6.3.7 分频器输出

该设备包含一个由16级2分频移位寄存器组成的分频器。其中哪一级的值被输出由一个16选1的多路转换器选择，并被映射到P3.6输出。所以有16种以2为幂的时钟分频选择，频率从 $Fin/2^1$ 到$Fin/2^{17}$，其中Fin 为输入到时钟分频器的时钟频率。

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

当写1到DIVIDER_EN(FRQDIV[4])，链计数器开始计数，当写0到DIVIDER_EN(FRQDIV[4])，链计数器持续计数直到分频时钟达到低状态并停留在低状态。

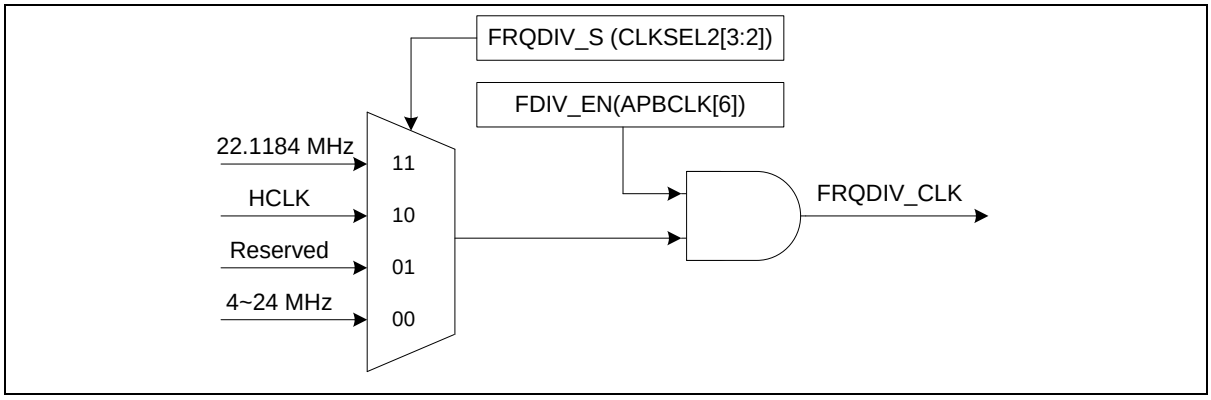


图6.3.7-1分频器的时钟源

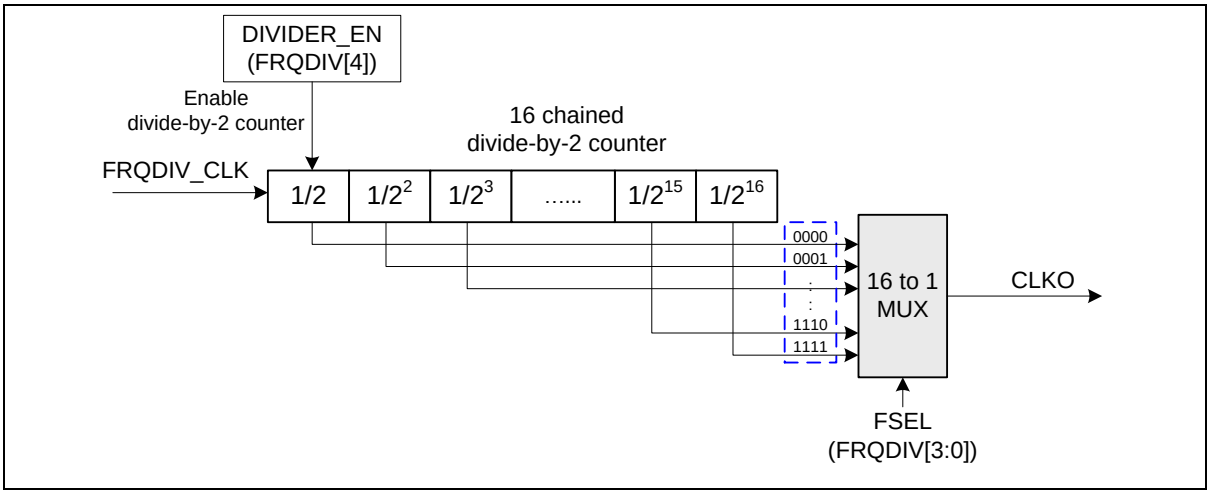


图6.3.7-2分频器框图

6.3.8 时钟控制寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位后的值
PWRCON	CLK_BA + 00	R/W	系统掉电控制寄存器	0x0000_000X
AHBCLK	CLK_BA + 04	R/W	AHB 设备时钟使能控制寄存器	0x0000_0001
APBCLK	CLK_BA + 08	R/W	APB 设备时钟使能控制寄存器	0x0000_000x
CLKSTATUS	CLK_BA + 0C	R/W	时钟状态监控寄存器	0x0000_00XX
CLKSEL0	CLK_BA + 10	R/W	时钟源选择控制寄存器0	0xFFFF_FFFX
CLKSEL1	CLK_BA + 14	R/W	时钟源选择控制寄存器1	0xFFFF_FFFX
CLKDIV	CLK_BA_+ 18	R/W	时钟分频数目寄存器	0x0000_0000
CLKSEL2	CLK_BA + 1C	R/W	时钟源选择控制寄存器2	0xFFFF_FFFX
PLLCON	CLK_BA + 20	R/W	PLL控制寄存器	0x0005_C22E
FRQDIV	CLK_BA + 24	R/W	分频器控制寄存器	0x0000_0000

6.3.9 时钟控制寄存器描述

掉电控制寄存器（**PWRCON**）

除BIT[6]外，PWRCON的其他位都受保护。要编程这些被保护的位需要向写地址0x5000_0100写入"59h", "16h", "88h"去禁用寄存器保护。 参考寄存器**REGWRPROT**，其地址是GCR_BA + 0x100

寄存器	偏移量	R/W	描述	复位后的值
PWRCON	CLK_BA + 00	R/W	系统掉电控制寄存器	0x0000_000X

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							**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**	**保留**	**XTL12M_EN**

Bits	描述	
[31:9]	保留	保留
[8]	**PD_WAIT_CPU**	**控制进入掉电模式的条件** 1 = 在PWR_DOWN_EN置1与CPU执行WFE/WFI指令时，芯片进入掉电模式。 0 = PWR_DOWN_EN 置1时，芯片进入掉电模式
[7]	**PWR_DOWN_EN**	**使能系统掉电模式** 该位置 “1” ，使能芯片的掉电模式，激活芯片的掉电行为取决于PD_WAIT_CPU位 (a) PD_WAIT_CPU 为 “0”,置位PWR_DOWN_EN 后，芯片马上进入掉电模式。 (b) PD_WAIT_CPU 为 “1”, 置位PWR_DOWN_EN 后，直到CPU进入睡眠模式(WFI)，芯片才进入掉电模式 芯片由掉电模式唤醒，该位自动清零，在下次掉电时，用户需要重新置位该位。 掉电 模式下, 外部高速晶振（4~24MHz）与22.1184 MHz高速振荡器被禁用， 10 KHz低速振荡器的使能不受该位控制 掉电时, PLL 与系统时钟也被禁用,时钟源选择被忽略. 如果外设以10KHz低速振荡器作为时钟，则其时钟不受该位控制 1 = 芯片立即进入掉电模式 或等待CPU休眠命令WFI 0 = 芯片工作于正常模式或CPU进入空闲模式 (休眠模式)

Bits	描述	
[6]	**PD_WU_STS**	**芯片掉电唤醒状态标志** 若“掉电唤醒”置位, 表明芯片从掉电模式恢复 如果GPIO, UART , WDT, ACMP或者BOD唤醒，该标志置位 写1清零. 注意: 这个比特只有在 PD_WU_INT_EN (PWRCON[5]) 被置为 1的时候才会被置位
[5]	**PD_WU_INT_EN**	**掉电模式唤醒的中断使能** 0 = 禁用 1 = 使能。从掉电唤醒时，产生中断. 当 PD_WU_STS 和 PD_WU_INT_EN 都为高时中断将发生
[4]	**PD_WU_DLY**	**唤醒延迟计数器使能.** 当芯片从掉电模式唤醒时，该域可以控制延迟一定时钟周期以等待系统时钟稳定. 当芯片工作于外部高速晶振（4~24MHz），延迟时间为4096 个时钟周期，工作于22.1184MHZ时，延迟256 个时钟周期。 1 = 使能时钟周期延迟 0 = 禁用时钟周期延迟
[3]	**OSC10K_EN**	**内部 10KHz 低速振荡器控制（写保护）** 1 = 使能10KHz 低速振荡器 0 =禁用10KHz 低速振荡器
[2]	**OSC22M_EN**	**内部 22.1184MHz高速振荡器控制（写保护）** 1 = 使能22.1184MHz高速振荡器 0 =禁用22.1184MHz高速振荡器
[1]	**保留**	保留
[0]	**XTL12M_EN**	**外部 12MHz晶振控制（写保护）** 该位的默认值由flash控制器用户配置寄存器config0 [26:24] 设置。 当默认时钟源为外部高速晶振（4~24MHz）. 该位自动置1 1 = 使能晶振 0 =禁用晶振

寄存器 指令模式	PWR_DOWN_EN	PD_WAIT_CPU	CPU 运行 WFE/WFI指令	时钟门控
正常运行模式	1'b0	1'b0	NO	通过控制寄存器关闭所有时钟
IDLE 模式 **(CPU 进入休眠模式)**	1'b0	1'b0	YES	仅CPU内部时钟关闭
Power_down 模式	1'b1	1'b0	NO	大部分时钟关闭，仅外部10K与WDT/Timer/PWM/ADC可能仍然处于激活状态.
Power_down Mode **(CPU 进入深度休眠模式)**	1'b1	1'b1	YES	大部分时钟关闭，仅外部10K与WDT/Timer/PWM/ADC可能仍然处于激活状态

表 6.3-1 掉电模式控制表

当芯片进入掉电模式时，用户可以通过一些中断唤醒CPU。在设定PWR_DOWN_EN位（PWRCON[7]）之前用户应该使能相关中断和NVIC IRQ使能位(NVIC_ISER)，以确保进入掉电模式以后可以成功唤醒。

AHB设备时钟使能控制寄存器（AHBCLK）

该寄存器各位用于使能/禁用系统与AHB设备时钟

寄存器	偏移量	R/W	描述	复位后的值
AHBCLK	CLK_BA + 04	R/W	AHB设备时钟使能控制寄存器	0x0000_0001

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				**EBI_EN**	**ISP_EN**	保留	保留

Bits	描述	
[31:4]	保留	保留
[3]	**EBI_EN**	**EBI 控制器时钟使能控制.** 1 = 使能EBI控制器时钟. 0 = 禁用EBI控制器时钟.
[2]	**ISP_EN**	**Flash ISP 控制器时钟使能控制.** 1 = 使能 the Flash ISP控制器时钟. 0 = 禁用the Flash ISP控制器时钟.
[1:0]	**保留**	保留

APB 设备时钟使能控制寄存器（APBCLK）

该寄存器的各位用于使能/禁用APB设备和外设时钟.

寄存器	偏移量	R/W	描述	复位后的值
APBCLK	CLK_BA + 08	R/W	APB设备时钟使能控制寄存器	0x0000_000X

31	30	29	28	27	26	25	24
保留	ACMP_EN	保留	ADC_EN	保留			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PWM67_EN	PWM45_EN	PWM23_EN	PWM01_EN	保留		UART1_EN	UART0_EN
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留		SPI1_EN	SPI0_EN	保留			I2C_EN
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留	FDIV_EN	TMR3_EN	TMR2_EN	TMR1_EN	TMR0_EN	保留	WDT_EN

Bits	描述	
[31]	保留	保留
[30]	**ACMP_EN**	**ACMP时钟使能控制** 1 = 使能 ACMP 时钟 0 = 禁用 ACMP 时钟
[29]	保留	保留
[28]	**ADC_EN**	**ADC时钟使能控制** 1 = 使能 ADC 时钟 0 = 禁用 ADC 时钟
[27:24]	保留	保留
[23]	**PWM67_EN**	**PWM_67 时钟使能** 1 = 使能 PWM67 时钟 0 = 禁用 PWM67 时钟
[22]	**PWM45_EN**	**PWM_45时钟使能** 1 = 使能 PWM45 时钟 0 = 禁用 PWM45 时钟

Bits	描述	
[21]	**PWM23_EN**	**PWM_23时钟使能** 1 = 使能 PWM23 时钟 0 = 禁用 PWM23 时钟
[20]	**PWM01_EN**	**PWM_01时钟使能** 1 = 使能 PWM**01**时钟 0 = 禁用 PWM**01** 时钟
[19:18]	保留	保留
[17]	**UART1_EN**	**UART1 时钟使能** 1 = 使能 UART1 时钟 0 = 禁用 UART1 时钟
[16]	**UART0_EN**	**UART0 时钟使能** 1 = 使能 UART0 时钟 0 = 禁用 UART0 时钟
[15:14]	保留	保留
[13]	**SPI1_EN**	**SPI1 时钟使能** 1 = 使能 SPI1 时钟 0 = 禁用 SPI1 时钟
[12]	**SPI0_EN**	**SPI0 时钟使能** 1 = 使能 SPI0 时钟 0 = 禁用 SPI0 时钟
[11:9]	保留	保留
[8]	**I2C_EN**	**I2C 时钟使能** 1 = 使能 I2C 时钟 0 = 禁用 I2C 时钟
[7]	保留	保留
[6]	**FDIV_EN**	分频器输出时钟使能控制 0 = 禁用 1 = 使能

Bits	描述	
[5]	**TMR3_EN**	**Timer3 时钟使能控制** 0 = 禁用 1 = 使能
[4]	**TMR2_EN**	**Timer2 时钟使能控制** 0 = 禁用 1 = 使能
[3]	**TMR1_EN**	**Timer1 时钟使能控制** 0 = 禁用 1 = 使能
[2]	**TMR0_EN**	**Timer0 时钟使能控制** 0 = 禁用 1 = 使能
[1]	保留	保留
[0]	**WDT_EN**	**Watch Dog 时钟使能 (写保护)** 该位是受保护的位，对该位编程时，需要向0x5000_0100 依次写入“59h”, “16h”, “88h” 来解除寄存器写保护，参考寄存器REGWRPROT 默认值由flash控制器用户可配置寄存器congig0 bit[31]设置 0 = 禁用 1 = 使能

时钟状态寄存器（CLKSTATUS）

该寄存器各位用于监控芯片时钟是否稳定，时钟切换是否失败.

寄存器	偏移量	R/W	描述	复位后的值
CLKSTATUS	CLK_BA + 0C	R/W	时钟状态监控寄存器	0x0000_00XX

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
CLK_SW_FAIL	保留		OSC22M_STB	OSC10K_STB	PLL_STB	保留	XTL12M_STB

Bits	描述	
[31:8]	保留	-
[7]	CLK_SW_FAIL	时钟切换失败标志 1 = 时钟切换失败 0 = 时钟切换成功 当目标切换时钟源不稳定时，该位置位 写1清零.
[6:5]	保留	-
[4]	OSC22M_STB	OSC22M (内部 22.1184 MHz)时钟源稳定标志（只读） 1 = OSC22M 时钟稳定 0 = OSC22M 时钟不稳定或没有使能
[3]	OSC10K_STB	OSC10K 时钟源稳定标志（只读） 1 = OSC10K 时钟稳定 0 = OSC10K 时钟不稳定或没有使能
[2]	PLL_STB	PLL 时钟源稳定标志（只读）

Bits	描述	
		1 = PLL 时钟稳定 0 = PLL 时钟不稳定或没有使能
[1]	保留	-
[0]	**XTL12M_STB**	**XTL12M 时钟源稳定标志（只读）** 1 = XTL12M 时钟稳定 0 = XTL12M 时钟不稳定或没有使能

时钟源选择控制寄存器0 （CLKSEL0）

寄存器	偏移量	R/W	描述	复位后的值
CLKSEL0[1]	CLK_BA + 10	R/W	时钟源选择控制寄存器0	0xFFFF_FFFX[2]

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		**STCLK_S**			**HCLK_S**		

Bits	描述	
[31:6]	保留	保留
[5:3]	**STCLK_S**	**MCU Cortex_M0 SysTick 时钟源选择（写保护）.** 受保护位，对该位编程时，需要向0x5000_0100 依次写入“59h”, “16h”, “88h” 来解除寄存器写保护，参考寄存器REGWRPROT，地址为GCR_BA + 0x100 000 = 外部高速晶振（4~24MHz） 001 = 保留 010 =外部高速晶振（4~24MHz）1/2分频 011 = HCLK**/2** 111 = 内部 22.1184MHz 高速振荡器 1/2分频. 其它 = 保留

Bits	描述	
[2:0]	**HCLK_S**	**HCLK 时钟源选择（写保护）.** **注:** 1. 在时钟切换之前，相关时钟源(当前和新选)都必须打开 2. 任何复位后，默认值将从用户配置寄存器**CFOSC(Config0[26:24])** 加载，所以默认值可为000b 或 111b. 3. 受保护位，对该位编程时，需要向0x5000_0100 依次写入“59h”, “16h”, “88h”来解除寄存器写保护，参考寄存器REGWRPROT，地址为GCR_BA + 0x100 00 =外部高速晶振（4~24MHz） 001 = 保留 010 = PLL 时钟 011 = 内部 10KHz 低速振荡器时钟 111 = 内部 22.1184MHz 高速振荡器时钟 Others = 保留.

时钟源选择控制寄存器1（CLKSEL1）

在时钟切换前，相关的时钟源（当前和新选），必须都打开。

寄存器	偏移量	R/W	描述	复位后的值
CLKSEL1	CLK_BA + 14	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**		保留		**UART_S**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留	**TMR3_S**			保留	**TMR2_S**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留	**TMR1_S**			保留	**TMR0_S**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留				**ADC_S**		**WDT_S**	

Bits	描述	
[31:30]	**PWM23_S**	**PWM3 与 PWM2的时钟源选择.** PWM3 与 PWM2使用相同的时钟源 和相同的分频 00 = 外部高速晶振（4~24MHz） 01 = 保留 10 = HCLK 11 = 内部 22.1184MHz 高速振荡器
[29:28]	**PWM01_S**	**PWM1 与 PWM0的时钟源选择.** PWM1 与 PWM0使用相同的时钟源 和相同的分频 00 = 外部高速晶振（4~24MHz） 01 = 保留 10 = HCLK 11 = 内部22.1184MHz 高速振荡器
[27:26]	保留	保留

Bits	描述	
[25:24]	**UART_S**	**UART时钟源选择.** 00 =外部高速晶振（4~24MHz） 01 = PLL 10 = 保留 11 = 内部22.118422MHz 振荡器
[23]	保留	保留
[22:20]	**TMR3_S**	**TIMER3 时钟源选择.** 000 =外部高速晶振（4~24MHz） 001 =保留 010 = HCLK 111 =内部 22.1184MHz 高速振荡器 Others = 保留
[19]	保留	保留
[18:16]	**TMR2_S**	**TIMER2时钟源选择.** 000 =外部高速晶振（4~24MHz） 001 =保留 010 = HCLK 111 =内部 22.1184MHz 高速振荡器 Others = 保留
[15]	保留	保留
[14:12]	**TMR1_S**	**TIMER1 时钟源选择.** 000 =外部高速晶振（4~24MHz） 001 =保留 010 = HCLK 111 =内部 22.1184MHz 高速振荡器 Others = 保留
[11]	保留	保留
[10:8]	**TMR0_S**	**TIMER0 时钟源选择.** 000 =外部高速晶振（4~24MHz） 001 =保留 010 = HCLK 111 =内部 22.1184MHz 高速振荡器 Others = 保留
[7:4]	保留	保留

Bits	描述	
[3:2]	**ADC_S**	**ADC 时钟源选择.** 00 =外部高速晶振（4~24MHz） 01 = PLL 10 = HCLK 11 =内部 I 22.1184 MHz 振荡器
[1:0]	**WDT_S**	**WDG CLK 时钟源选择.** 受保护位，对该位编程时，需要向0x5000_0100 依次写入“59h”, “16h”, “88h” 来解除寄存器写保护，参考寄存器REGWRPROT，地址为GCR_BA + 0x100 00 =外部高速晶振（4~24MHz） 01 =保留 10 = HCLK/2048时钟 11 = 内部 10KHz低速振荡器时钟

时钟源选择控制寄存器（CLKSEL2）

在时钟切换前，相关的时钟源(当前和新选的)，必须都打开.

寄存器	偏移量	R/W	描述	复位后的值
CLKSEL2	CLK_BA + 1C	R/W	时钟源选择控制寄存器 2	**0x0000_00FF**

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**
PWM67_S		**PWM45_S**		**FRQDIV_S**		保留	

Bits	描述	
[31:8]	保留	保留
[7:6]	**PWM67_S**	**PWM6 与 PWM7的时钟源选择**. PWM6 与 PWM7使用相同的时钟源 和相同的分频器 00 = 外部高速晶振（4~24MHz） 01 = 保留 10 = HCLK 11 = 内部 22.1184MHz 高速振荡器
[5:4]	**PWM45_S**	**PWM4 与 PWM5的时钟源选择**. PWM4 与 PWM5使用相同的时钟源 和相同的分频器 00 = 外部高速晶振（4~24MHz） 01 = 保留 10 = HCLK 11 = 内部22.1184MHz 高速振荡器

Bits	描述	
[3:2]	**FRQDIV_S**	**时钟分频器时钟源选择** 00 = 外部高速晶振（4~24MHz） 01 = 保留 10 = HCLK 11 = 内部 22.1184MHz 高速振荡器
[1:0]	保留	保留

时钟分频寄存器(CLKDIV)

寄存器	偏移量	R/W	描述	复位后的值
CLKDIV	CLK_BA_+ 18	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**
ADC_N							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留				**UART_N**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留				**HCLK_N**			

Bits	描述	
[31:24]	保留	保留
[23:16]	**ADC_N**	ADC时钟频率=ADC时钟源频率/（ADC_N + 1）
[15:12]	保留	保留
[11:8]	**UART_N**	UART时钟频率 = (UART时钟源频率) / (UART_N + 1)
[7:4]	保留	保留
[3:0]	**HCLK_N**	HCLK 时钟频率 = (HCLK时钟源频率) / (HCLK_N + 1)

PLL控制寄存器（PLLCON）

PLL的参考时钟输入来自外部高速晶振时钟（4~24MHz）输入或内部22.1184MHz高速振荡器，该寄存器用于控制PLL的输出频率和PLL的操作模式

寄存器	偏移量	R/W	描述	复位后的值
PLLCON	CLK_BA + 20	R/W	PLL控制寄存器	0x0005_C22E

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

Bits	描述	
[19]	**PLL_SRC**	**PLL时钟源选择** 1 = PLL 时钟源为22.1184 MHz 振荡器 0 = PLL时钟源为外部高速晶振（4~24MHz）
[18]	**OE**	**PLL OE (FOUT enable)引脚控制** 0 = 使能 PLL FOUT 1 = PLL FOUT 固定为低
[17]	**BP**	**PLL 旁路控制** 0 = PLL 正常模式 (默认) 1 = PLL 时钟输出与时钟输入相同(XTALin)
[16]	**PD**	**掉电模式.** 如果设置PWRCON寄存器的IDLE位为”1”，PLL也会进入掉电模式 0 = PLL正常模式 (默认) 1 = PLL 掉电模式
[15:14]	**OUT_DV**	PLL 输出分频控制

Bits	描述	
[13:9]	**IN_DV**	PLL输入分频控制
[8:0]	**FB_DV**	PLL反馈分频控制

PLL输出时钟频率设置

$$FOUT = FIN \times \frac{NF}{NR} \times \frac{1}{NO}$$

约束条件:

1. $4MHz < FIN < 24MHz$
2. $800KHz < \frac{FIN}{2 * NR} < 7.5MHz$
3. $100MHz < FCO = FIN \times \frac{NF}{NR} < 200MHz$
   $120MHz < FCO$ ispreferred

符号	说明
FOUT	输出时钟频率
FIN	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PLL频率设置

PLLCON默认值： 0xC22E
FIN = 12 MHz
NR = (1+2) = 3
NF = (46+2) = 48
NO = 4
FOUT = 12/4 x 48 x 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	复位后的值
FRQDIV	CLK_BA+ 24	R/W	频率分频器控制寄存器	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			**DIVIDER_EN**	**FSEL**			

Bits	描述	
[31:5]	保留	保留
[4]	**DIVIDER_EN**	**频率分频器使能位** 0 = 禁用频率分频 1 = 使能频率分频
[3:0]	**FSEL**	**分频器输出频率选择位** 输出频率的公式是 $\mathbf{F_{out} = F_{in}/2^{(N+1)}}$, 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]的值。

6.4 通用 I/O

6.4.1 概述

这款MCU有40个通用I/O引脚，并和某些功能复用引脚。40个引脚分配在P0, P1, P2, P3 和 P4五个端口上，每个端口最多8个引脚。每个引脚都是独立的，都有相应的寄存器来控制引脚工作模式与数据。

每个I/O引脚上的I/O类型可由软件独立地配置为输入，输出，开漏或准双向模式。默认所有的I/O引脚处于准双向模式，端口数据寄存器Px_DOUT[7:0]的值为0x000_00FF。每个I/O引脚配有一个非常弱的独立的上拉电阻，VDD从5.0V 到 2.5V时，内部弱上拉电阻阻值大约为110KΩ~300KΩ。

6.4.1.1 *输入模式说明*

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

6.4.1.2 *输出模式说明*

设置Px_PMD(PMDn[1:0])为 2'b01，Px[n]为输出模式，I/O引脚支持数字输出功能，有拉电流/灌电流能力。Px_DOUT 相应位的值被送到相应引脚上。

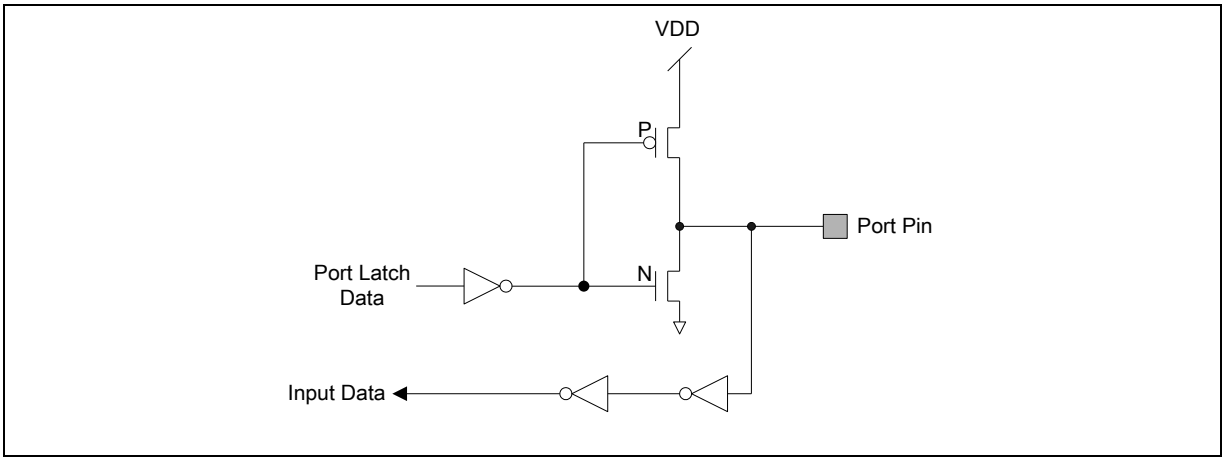


图 6.4.1-1 推挽输出

6.4.1.3 *开漏模式说明*

设置 Px_PMD(PMDn[1:0])为 2'b10，Px[n]为开漏模式，I/O支持数字输出功能，但仅有灌电流能力， 为了把I/O引脚拉到高电平状态，需要外接一颗上拉电阻. 如果Px_DOUT相应位bit [n]的值为"0"，引脚上输出低电平. 如果Px_DOUT 相应位bit [n]的值为"1", 该引脚输出为高电平，由内部上拉电阻或外部上拉电阻控制。

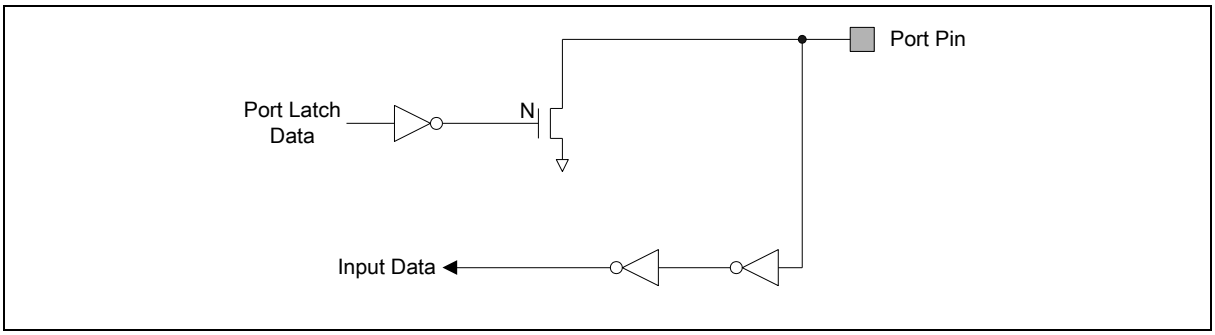


图 6.4.1-2 开漏输出

6.4.1.4 *准双向模式说明*

设置Px_PMD(PMDn[1:0]) 为 2'b11，Px[n]引脚为准双向模式，I/O同时支持数字输出和输入功能，但拉电流仅达数百uA. 要实现数字输入，需要先将Px_DOUT 相应位置1。准双向输出是80C51及其派生产品所共有的模式。若Px_DOUT相应位bit[n]为"0", 引脚上输出为"低电平". 若Px_DOUT相应位bit[n]为"1", 该引脚将核对引脚值. 若引脚值为高，没有任何动作，若引脚值为低，该引脚置为强高2个时钟周期，然后禁用强输出驱动，引脚状态由内部上拉电阻控制。注: 准双向模式的拉电流能力仅有200uA到30uA(相应VDD的电压从5.0V到2.5V)

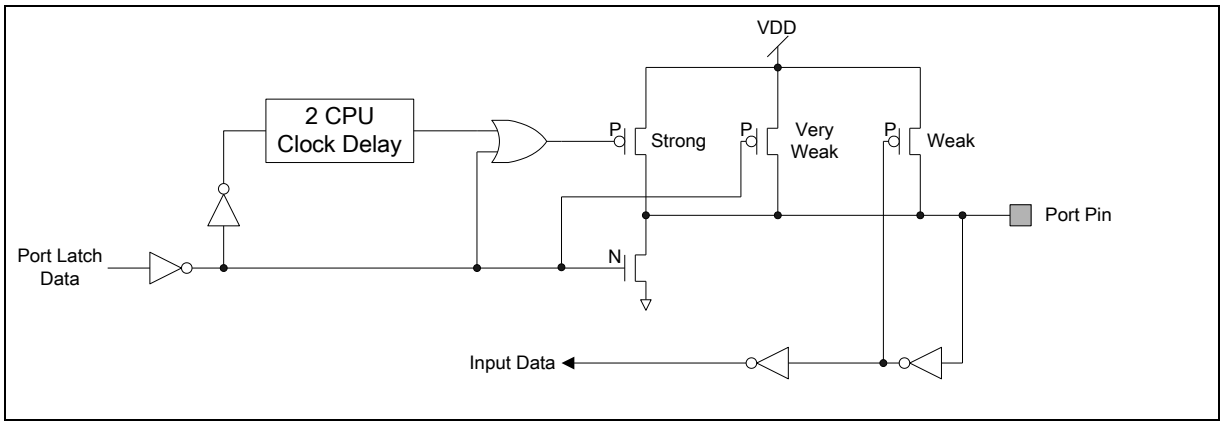


图 6.4.1-3 准双端 I/O 模式

6.4.2 Port 0-4 控制器寄存器映射映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位后的值
GP_BA = 0x5000_4000				
P0_PMD	GP_BA+0x000	R/W	P0 Bit模式控制	0x0000_FFFF
P0_OFFD	GP_BA+0x004	R/W	P0 Bit OFF数字使能	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管脚数值	0x0000_00XX
P0_DBEN	GP_BA+0x014	R/W	P0防反弹使能	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	0xXXXX_XXXX
P1_PMD	GP_BA+0x040	R/W	P1 Bit模式使能	0x0000_FFFF
P1_OFFD	GP_BA+0x044	R/W	P1 Bit OFF数字使能	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管脚数值	0x0000_00XX
P1_DBEN	GP_BA+0x054	R/W	P1防反弹使能	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	0xXXXX_XXXX
P2_PMD	GP_BA+0x080	R/W	P2 Bit 模式使能	0x0000_FFFF
P2_OFFD	GP_BA+0x084	R/W	P2 Bit OFF数字使能	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 管脚数值	0x0000_00XX
P2_DBEN	GP_BA+0x094	R/W	P2防反弹使能	0x0000_0000

寄存器	偏移量	R/W	描述	复位后的值
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 Bit模式使能	0x0000_FFFF
P3_OFFD	GP_BA+0x0C4	R/W	P3 Bit OFF数字使能	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管脚数值	0x0000_00XX
P3_DBEN	GP_BA+0x0D4	R/W	P3防反弹使能	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 Bit模式使能	0x0000_FFFF
P4_OFFD	GP_BA+0x104	R/W	P4 Bit OFF数字使能	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 E管脚数值	0x0000_00XX
P4_DBEN	GP_BA+0x114	R/W	P4 防反弹使能	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	防反弹周期控制	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

寄存器	偏移量	R/W	描述	复位后的值
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

寄存器	偏移量	R/W	描述	复位后的值
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

6.4.3 Port 0-4 控制器寄存器描述

Port 0-4 I/O 模式控制(Px_PMD)

寄存器	偏移量	R/W	描述	复位后的值
P0_PMD	GP_BA+0x000	R/W	P0 Pin I/O 模式控制	0x0000_FFFF
P1_PMD	GP_BA+0x040	R/W	P1 Pin I/O 模式控制	0x0000_FFFF
P2_PMD	GP_BA+0x080	R/W	P2 Pin I/O 模式控制	0x0000_FFFF
P3_PMD	GP_BA+0x0C0	R/W	P3 Pin I/O 模式控制	0x0000_FFFF
P4_PMD	GP_BA+0x100	R/W	P4 Pin I/O 模式控制	0x0000_FFFF

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**
PMD7		**PMD6**		**PMD5**		**PMD4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PMD3		**PMD2**		**PMD1**		**PMD0**	

Bits	描述	
[31:16]	保留	保留
[2n+1 :2n]	**PMDn**	**Px I/O Pin[n]** 模式控制 Px的I/O类型 00 = Px [n] 输入模式. 01 = Px [n] 输出模式. 10 = Px [n] 开漏模式. 11 = Px [n] 准双端模式 x=0~4, n = 0~7

Port 0-4 Bit OFF 数字寄存器使能 (Px_OFFD)

寄存器	偏移量	R/W	描述	复位后的值
P0_OFFD	GP_BA+0x004	R/W	P0 数字输入通道关闭控制	0x0000_0000
P1_OFFD	GP_BA+0x044	R/W	P1 数字输入通道关闭控制	0x0000_0000
P2_OFFD	GP_BA+0x084	R/W	P2 数字输入通道关闭控制	0x0000_0000
P3_OFFD	GP_BA+0x0C4	R/W	P3 数字输入通道关闭控制	0x0000_0000
P4_OFFD	GP_BA+0x104	R/W	P4 数字输入通道关闭控制	0x0000_0000

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

Bits	描述	
[31:24]	保留	保留
[23:16]	**OFFD**	**OFFD: Px Pin[n] OFF 数字输入通道关闭控制** 1 = 关闭 IO数字输入通道（数字输入连接到低电平） 0 = 使能 IO 数字输入通道 x=0~4, n = 0~7
[15:0]	保留	保留

Port 0-4数据输出值(Px_DOUT)

寄存器	偏移量	R/W	描述	复位后的值
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

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**
DOUT[7:0]							

Bits	描述	
[31:8]	保留	保留
[n]	**DOUT[n]**	**Px Pin[n] 输出值** **Px**配置成输出，开漏，和准双向模式时，这些位控制**Px**引脚状态. 1 = 相应的输出模式使能位设置时，**Px** Pin[n] 为高. 0 =相应的输出模式使能位设置时，**Px** Pin[n] 为低. x=0~4, n = 0~7

Port0-4数据输出写屏蔽(Px _DMASK)

寄存器	偏移量	R/W	描述	复位后的值
P0_DMASK	GP_BA+0x00C	R/W	P0数据输出写屏蔽	0xXXXX_XX00
P1_DMASK	GP_BA+0x04C	R/W	P1数据输出写屏蔽	0xXXXX_XX00
P2_DMASK	GP_BA+0x08C	R/W	P2数据输出写屏蔽	0xXXXX_XX00
P3_DMASK	GP_BA+0x0CC	R/W	P3数据输出写屏蔽	0xXXXX_XX00
P4_DMASK	GP_BA+0x10C	R/W	P4数据输出写屏蔽	0xXXXX_XX00

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**
DMASK[7:0]							

Bits	描述	
[31:8]	保留	保留
[n]	**DMASK[n]**	**Px 数据输出写屏蔽（写保护）** 用于保护寄存器Px_DOUT 的相应bit[n]. 当设置DMASK bit[n] 为 “1”时， Px_DOUT 相应bit 被保护，此时不能向被保护位写数据，写Px_DOUT相应bit不起作用 0 = 相应的Px_DOUT [n] 位未被屏蔽 1 = 相应的Px_DOUT [n] 位被屏蔽 x=0~4, n = 0~7

Port 0-4管脚数据(Px _PIN)

寄存器	偏移量	R/W	描述	复位后的值
P0_PIN	GP_BA+0x010	R	P0管脚状态	0x0000_00XX
P1_PIN	GP_BA+0x050	R	P1管脚状态	0x0000_00XX
P2_PIN	GP_BA+0x090	R	P2管脚状态	0x0000_00XX
P3_PIN	GP_BA+0x0D0	R	P3管脚状态	0x0000_00XX
P4_PIN	GP_BA+0x110	R	P4管脚状态	0x0000_00XX

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**
PIN[7:0]							

Bits	描述	
[31:8]	保留	保留
[n]	**PIN[n]**	**Px 管脚状态** 这些位的值反映各个Px 引脚的实际状态为高还是低 x=0~4, n = 0~7

Port 0-4防反弹使能(Px DBEN)

寄存器	偏移量	R/W	描述	复位后的值
P0_DBEN	GP_BA+0x014	R/W	P0防反弹使能	0xXXXX_XX00
P1_DBEN	GP_BA+0x054	R/W	P1防反弹使能	0xXXXX_XX00
P2_DBEN	GP_BA+0x094	R/W	P2防反弹使能	0xXXXX_XX00
P3_DBEN	GP_BA+0x0D4	R/W	P3防反弹使能	0xXXXX_XX00
P4_DBEN	GP_BA+0x114	R/W	P4防反弹使能	0xXXXX_XX00

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**
DBEN[7:0]							

Bits	描述	
[31:8]	保留	保留
[n]	**DBEN[n]**	**Px输入信号防反弹使能** DBEN[n] 用于使能相应位的防反弹功能。如果输入信号脉冲宽度不能被两个连续的防反弹采样周期所采样，则输入信号被视为信号反弹，从而不触发中断. DBEN[n] 仅用于边沿触发 中断，不用于电平触发中断 0 = 禁用 bit[n] 防反弹功能 1 = 使能 bit[n] 防反弹功能 防反弹功能对于边沿触发中断有效，对于电平触发中断模式，防反弹功能使能位不起作用. x=0~4, n = 0~7

Port 0-4中断模式控制(Px _IMD)

寄存器	偏移量	R/W	描述	复位后的值
P0_IMD	GP_BA+0x018	R/W	P0中断模式控制	0xXXXX_XX00
P1_IMD	GP_BA+0x058	R/W	P1中断模式控制	0xXXXX_XX00
P2_IMD	GP_BA+0x098	R/W	P2中断模式控制	0xXXXX_XX00
P3_IMD	GP_BA+0x0D8	R/W	P3中断模式控制	0xXXXX_XX00
P4_IMD	GP_BA+0x118	R/W	P4中断模式控制	0xXXXX_XX00

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**
IMD[7:0]							

Bits	描述	
[31:8]	保留	保留
[n]	**IMD[n]**	**Port 0-4中断模式控制** IMD[n] 用于控制中断是电平触发或边沿触发。若中断是边沿触发，触发源可以由防反弹控制，如果中断是电平触发，触发源由一个HCLK时钟采样并产生中断 0 = 边沿触发中断 1 = 电平触发中断 设置引脚为电平触发中断时，寄存器Px_IEN中只有一种电平可以被设定。若设置两种电平都触发中断，设置将被忽略，不会产生中断 防反弹功能对于边沿触发中断有效，对于电平触发中断无效. x=0~4, n = 0~7

Port 0-4中断使能控制(Px _IEN)

寄存器	偏移量	R/W	描述	复位后的值
P0_IEN	GP_BA+0x01C	R/W	P0中断使能	0x0000_0000
P1_IEN	GP_BA+0x05C	R/W	P1中断使能	0x0000_0000
P2_IEN	GP_BA+0x09C	R/W	P2中断使能	0x0000_0000
P3_IEN	GP_BA+0x0DC	R/W	P3中断使能	0x0000_0000
P4_IEN	GP_BA+0x11C	R/W	P4中断使能	0x0000_0000

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

Bits	描述	
[31:24]	保留	保留
[n+16]	**IR_EN[n]**	**Port 0-4 输入上升沿或输入高电平的中断使能** IR_EN[n] 用于使能相应Px[n]输入的中断。置 “1” 也可以使能引脚唤醒功能 设置 IR_EN[n] 位为 “1”时: 如果中断是电平触发模式，输入Px[n]的状态为高电平时，产生中断。 如果中断是边沿触发模式，输入Px[n]的状态由低电平到高电平变化时，产生中断 1 = 使能Px[n] I高电平或由低电平到高电平变化的中断 0 = 禁用Px[n] I高电平或由低电平到高电平变化的中断. x=0~4, n = 0~7
[15:8]	保留	保留

Bits	描述	
[n]	**IF_EN[n]**	**Port 0-4输入下降沿或输入低电平的中断使能** IF_EN[n] 用于使能相应Px[n]输入的中断。置 “1” 也可以使能引脚唤醒功能 设置 IF_EB[n] 位为 “1”时: 如果中断是电平触发模式，输入Px[n]的状态为低电平时，产生中断。 如果中断是边沿触发模式，输入Px[n]的状态由高电平到低电平变化时，产生中断。 1 =使能Px[n]低电平或由高电平到低电平变化的中断 0 =禁用Px[n]低电平或由高电平到低电平变化的中断 x=0~4, n = 0~7

Port 0-4 中断触发源(Px _ISRC)

寄存器	偏移量	R/W	描述	复位后的值
P0_ISRC	GP_BA+0x020	R/WC	P0中断触发源	0x0000_0000
P1_ISRC	GP_BA+0x060	R/WC	P1中断触发源	0x0000_0000
P2_ISRC	GP_BA+0x0A0	R/WC	P2中断触发源	0x0000_0000
P3_ISRC	GP_BA+0x0E0	R/WC	P3中断触发源	0x0000_0000
P4_ISRC	GP_BA+0x120	R/WC	P4中断触发源	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**
IF_ ISRC[7:0]							

Bits	描述	
[31:8]	保留	保留
[n]	**ISRC[n]**	**Port 0-4** 中断触发源 读 : 1 = Px[n]产生中断 0 = Px[n]没有中断 写 : 1= 清相应的中断标志 0= 无动作 x=0~4, n = 0~7

中断防反弹周期控制(DBNCECON)

寄存器	偏移量	R/W	描述	复位后的值
DBNCECON	GP_BA+0x180	R/W	外部中断防反弹控制	0x0000_0020

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		**ICLK_ON**	**DBCLKSRC**	**DBCLKSEL**			

Bits	描述	
[5]	**ICLK_ON**	**中断时钟On模式** 如果pin[n]中断被禁用，设置该位为“0” 将关闭中断产生电路的时钟 0 = 如果P0/1/2/3/4[n] 的中断被禁用，关闭时钟 1 = 总是使能中断产生电路的时钟 n=0~7
[4]	**DBCLKSRC**	**防反弹计数器时钟源选择** 1 = 防反弹计数器时钟源为内部10KHz 时钟 0 = 防反弹计数器时钟源为 HCLK

Bits	描述	
[3:0]	**DBCLKSEL**	防反弹采样周期选择

DBCLKSEL	描述
0	每 1 个时钟周期采样中断输入一
	每 2 个时钟周期采样中断输入一次
2	每4 个时钟采样中断输入周期一次
3	每8 个时钟周期采样中断输入一次
4	每16 个时钟周期采样中断输入一次
	每32 个时钟周期采样中断输入一次
6	每64 个时钟周期采样中断输入一次
7	每128 个时钟周期采样中断输入一次
8	每256 个时钟周期采样中断输入一次
9	每 2*256 个时钟周期采样中断输入一次
10	每4*256个时钟周期采样中断输入一次
11	每8*256 个时钟周期采样中断输入一次
12	每16*256 个时钟周期采样中断输入一次
13	每32*256 个时钟周期采样中断输入一次
14	每64*256 个时钟周期采样中断输入一次
15	每128*256 个时钟周期采样中断输入一次

GPIO 端口 [P0/P1/P2/P3/P4] I/O 位输出控制 (Pxx_DOUT)

寄存器	偏移量	R/W	描述	复位后的值
P0x_DOUT	GP_BA+0x200 - GP_BA+0x21C	R/W	P0 Pin I/O位输出控制	0x0000_0001
P1x_DOUT	GP_BA+0x220 - GP_BA+0x23C	R/W	P1 Pin I/O位输出控制	0x0000_0001
P2x_DOUT	GP_BA+0x240 - GP_BA+0x25C	R/W	P2 Pin I/O位输出控制	0x0000_0001
P3x_DOUT	GP_BA+0x260 - GP_BA+0x27C	R/W	P3 Pin I/O位输出控制	0x0000_0001
P4x_DOUT	GP_BA+0x280 - GP_BA+0x29C	R/W	P4 Pin I/O位输出控制	0x0000_0001

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							**Pxx_DOUT**

Bits	描述	
[0]	**Pxx_DOUT**	**Pxx I/O**位输出控制 设置该位可以控制一个GPIO管脚的输出值 1 = 设置相应的GPIO位为高 0 = 设置相应的GPIO位为低

Bits	描述	
		读这个寄存器可以得到IO引脚的状态 例如：写P00_DOUT将反应写的值到bit P0_DOUT[0]，读P00_DOUT将得到P0_PIN[0]的值

6.5 I2C 总线控制器 (主机/从机)

6.5.1 概述

I2C为双线，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。标准I2C是多主机总线，包括冲突检测和仲裁机制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。

依靠SCL时钟同步，数据在主机与从机间在SDA数据线上一字节一字节的传输，每个字节为8位长度，一个SCL时钟脉冲传输一个数据位，数据由最高位MSB首先传输，每个传输字节后跟随一个应答位，每个位在SCL为高时采样；因此，SDA线只有在SCL为低时才可以改变，在SCL为高时SDA必须保持稳定。当SCL为高时，SDA线上的跳变视为一个命令（START 或 STOP）， 更多详细的I2C总线时序请参考图6.5-1。

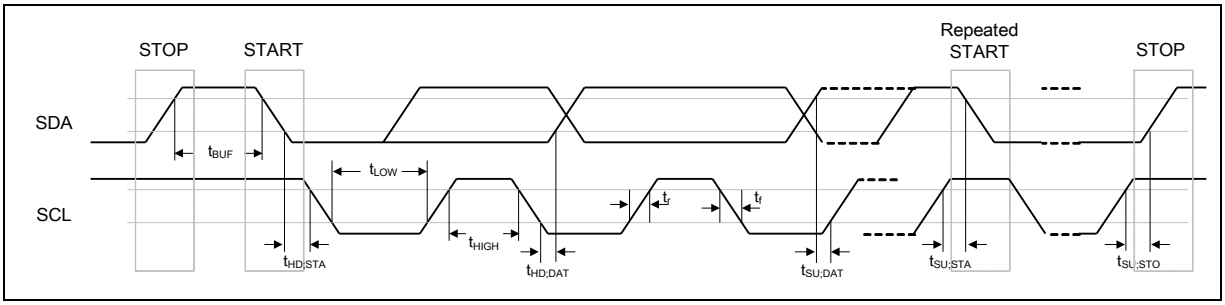


图 6.5.1-1 I2C 总线时序

该设备的片上I2C提供符合I2C总线标准模式规范的串行接口，I2C端口自动处理字节传输，将I2CON的ENS1位设置为1，可以使能该端口。 I2C H/W 接口通过两个引脚连接到I2C总线：SDA (串行数据线) 与 SCL (串行时钟线)。当这两个引脚被配置为开漏模式时，引脚SDA 与 SCL 用于 I2C操作需要上拉电阻。在作为 I2C 端口使用时, 用户必须先将这两个引脚设置为I2C功能。

6.5.2 特征

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

- 支持主机和从机模式
- 主从机之间双向数据传输
- 多主机总线支持 (无中心主机)
- 多主机间同时发送数据仲裁，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器件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方式来暂停和恢复串行传输
- 内建一个14位超时计数器，当I2C总线挂起并且计数器溢出时，该计数器将请求I2C中断
- 需要外部上拉用于高电平输出
- 可编程的时钟适用于不同速率控制

- 支持7位寻址模式
- I2C总线控制器支持多地址识别 (4组从机地址带屏蔽选项)

6.5.3 功能描述

6.5.3.1 *I2C协议*

通常标准I2C传输包含四个部分:

1) 起始信号或重复起始信号的产生
2) 从机地址和R/W位传输
3) 数据传输
4) 停止信号的产生

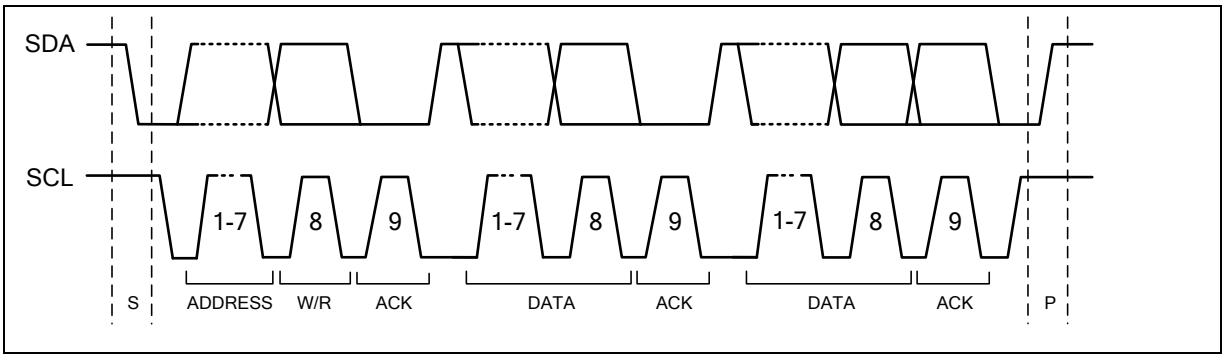


图 6.5.3-1 I2C 协议

6.5.3.2 *I2C总线上的数据传输*

主机发送器用7位地址寻址从机接收器

传输方向未改变

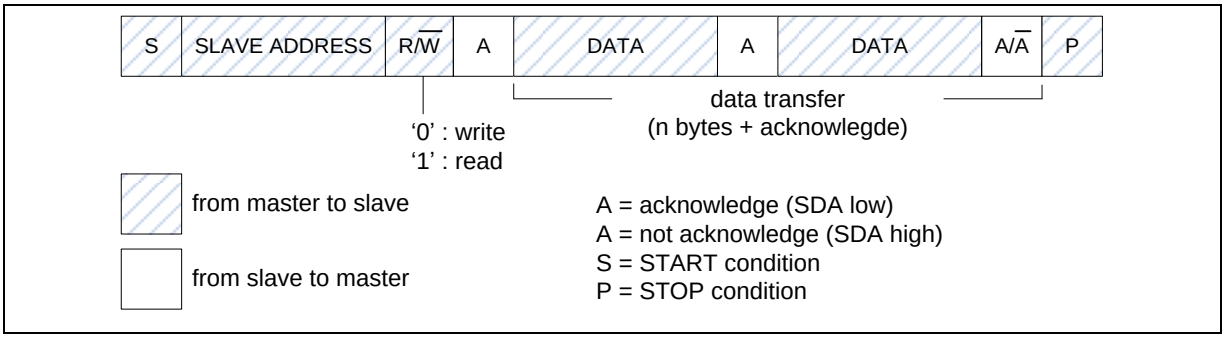


图 6.5.3-2 主机向从机传输数据

第一个字节(地址)传输后立即读取从机数据

传输方向改变

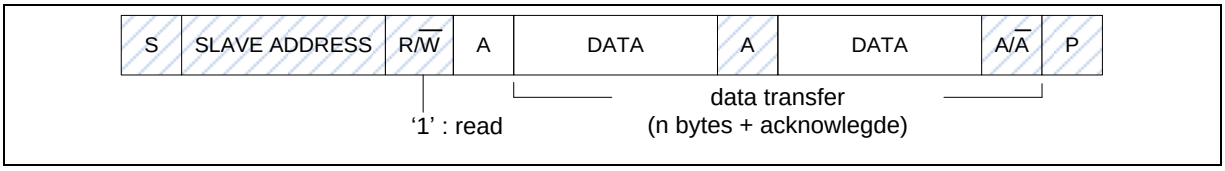


图 6.5.3-3 主机读取从机的数据

6.5.3.3 *起始或重复起始信号*

当总线处于空闲状态下，即没有任何主机设备占有总线(SCL 和SDA线同时为高)时，主机可以通过发送起始信号发起一次数据传输。起始信号，通常表示为**S-bit**, 定义为当SCL线为高电平时，SDA线上产生一个高电平到低电平的跳变。起始信号表示新的数据传输的开始。

重复起始信号 (Sr) 是指在两个START信号间不存在STOP信号 。主机用这种方式来和另外一个或同一个从机在不同的传输方向并且不释放总线的情形下通讯（如从写向一个设备到从该设备读取）。

停止信号

主机可以通过产生一个停止信号来结束通讯。停止信号，通常用**P-bit**表示， 定义为当SCL线为高电平时，SDA线上产生一个低电平到高电平的跳变。

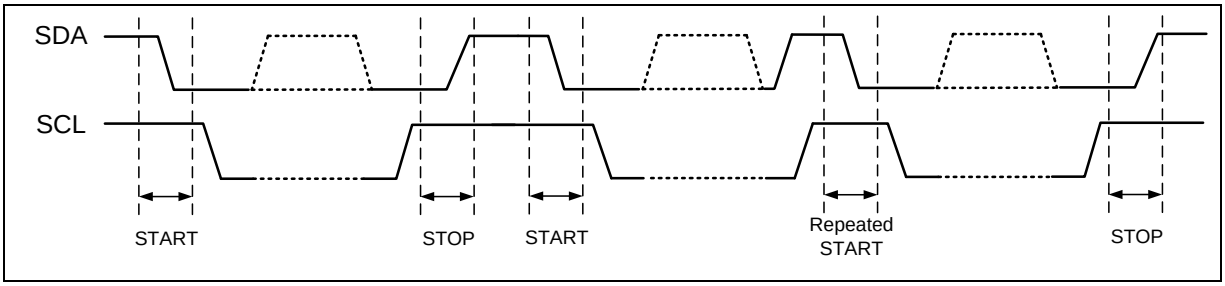


图 6.5.3-4 START 和 STOP 条件

6.5.3.4 *从机地址传输*

起始信号后传输的第一个字节是从机地址，从机地址的头7位是呼叫地址，紧跟7位地址后的是RW位。RW位通知从机数据传输方向。系统当中不会有两台从机有相同的地址。只有地址匹配的从机才会在SCL的第9个时钟周期拉低SDA作为应答信号来响应主机。

6.5.3.5 *数据传输*

当从机寻址成功完成，就可以根据主机发送的RW位所决定的方向，开始一字节一字节的传输数据，每一个传输的字节会在第九个SCL时钟周期跟随一个应答位，如果从机上产生无应答信号**(NACK)**，主机可以产生一个停止信号来中止本次数据传输，或者产生重复起始信号开始新一轮的数据传输。

如果主机作为接收设备，没有应答(NACK)从机，则从机释放SDA线，以便于主机产生一个停止或重复起始信号。

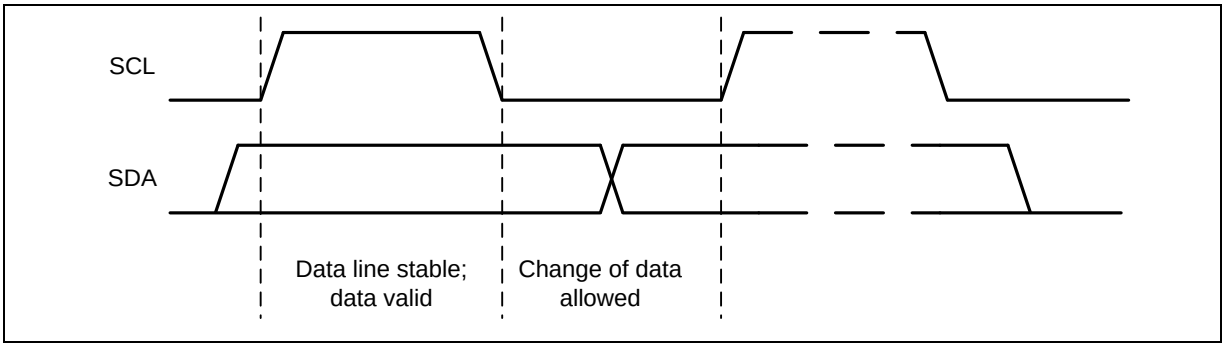


图 6.5.3-5 I2C 总线上的位传输

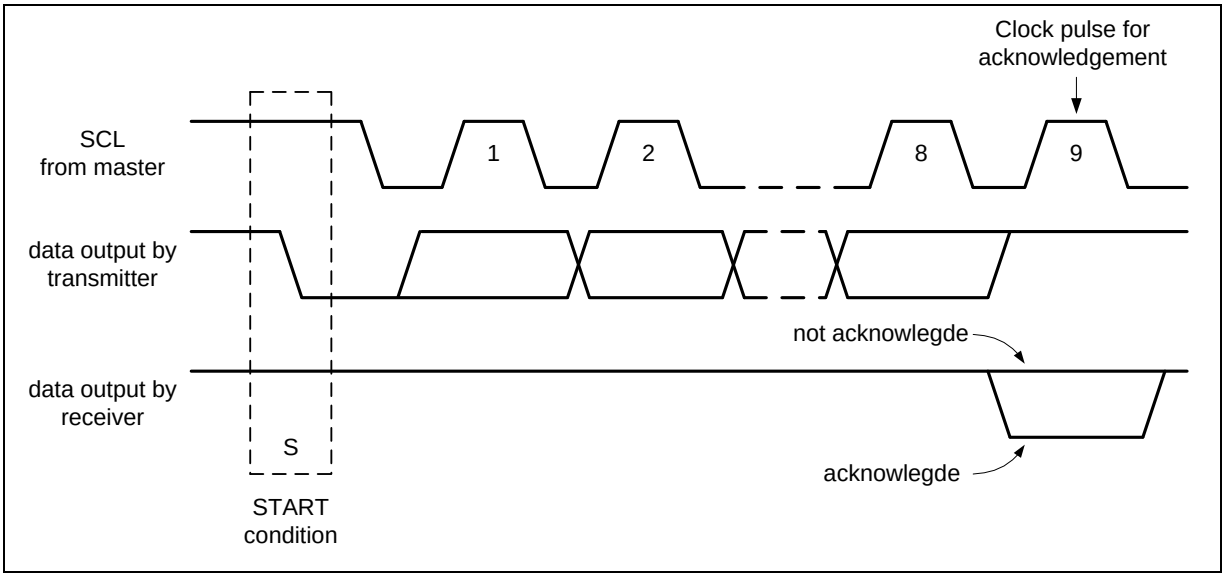


图 6.5.3-6 I2C 总线上的应答信号

6.5.4 I2C 协议寄存器

CPU通过如下13个特殊功能寄存器连接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（地址屏蔽寄存器，n=0~3）I2CLK (时钟速率寄存器) 和 I2CTOC (超时计数器寄存器) 。所有这些I2C特殊功能寄存器的第31位至第8位都是保留的，不具备任何功能，读回值为0.

当ENS1(I2CON[6])置1，I2C端口使能后，内部状态由I2CON和I2C总线逻辑状态控制。当有新的状态码产生，并被存储到I2CSTATUS之后，I2C中断标志位SI(I2CON[3])将自动被置位。若此时EI(I2CON[7])被设定为高，将会产生I2C中断。位域I2CSTATUS[7:3]存储内部状态码，I2CSTATUS寄存器的低三位始终为0，在SI被软件清零之前，I2CSTATUS寄存器的值保持稳定。 I2C的基地址是4002_0000。

6.5.4.1 *地址寄存器(I2CADDR)*

I2C端口带有4个从机地址寄存器I2CADDRn (n=0~3)。当I2C处于主机模式时，这四个寄存器的值是无关的。 在从机模式下，位域I2CADDRn [7:1]必须装入MCU自身从机地址，当I2CADDR地址与接收的从机地址符合时，I2C硬件将应答。

I2C端口支持广播呼叫功能。当GC位(I2CADDRn [0])被置位，I2C端口硬件会响应广播呼叫地址(00H). 清GC位可禁止广播呼叫功能。

当GC位被置位，且I2C处于从机模式时，在主机发送广播呼叫地址到I2C总线上之后，I2C可以通过从机地址00H接收广播呼叫，然后它将跟随GC模式的状态。

I2C总线控制器支持多地址识别，带有4组地址屏蔽寄存器I2CADMn (n=0~3)。当地址屏蔽寄存器某一位置1，表示接收到的地址的相应位将被忽略。如果该位置0，表示接收到的地址的相应位应当与地址寄存器中相应位的值完全一致。

6.5.4.2 *数据寄存器(I2CDAT)*

该寄存器存储的内容是准备发送的或刚接收的串行数据一个字节的数据。只要不在移位处理的过程中，CPU可以直接读写这8位I2CDAT [7:0]。当I2C处于已经定义过的状态下，且串行中断标志(SI)被置位，只要SI位一直处于置位状态，I2CDAT[7:0]中的数据保持稳定。在数据被移出的过程中，总线上的数据同时被移入，I2CDAT[7:0]总是包含出现在总线上的最后一个字节数据，这样，万一仲裁丢失，从主机发送器到从机接收器的传输数据仍然正确放在I2CDAT[7:0]中。

I2CDAT[7:0]和应答位一起组成一个9位的移位寄存器，应答位由I2C的硬件控制，CPU不能访问。串行数据在SCL线上串行时钟脉冲的上升沿被移入I2CDAT[7:0]。当一个字节被移入到I2CDAT [7:0]后，I2CDAT [7:0]中的串行数据是可以使用的，应答位(ACK或NACK) 由控制逻辑在第9个时钟返回。串行数据在SCL时钟脉冲的下降沿从I2CDAT[7:0]被移出，在SCL时钟脉冲的上升沿被移入I2CDAT[7:0]。

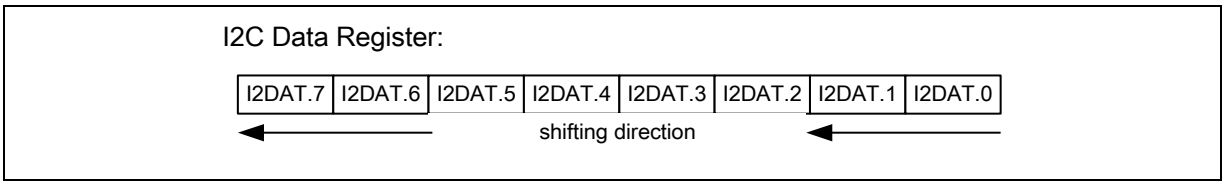


图 6.5.4-1 I2C 数据移位

6.5.4.3 *控制寄存器(I2CON)*

CPU可以直接读写这个8位位域I2CON[7:0]，I2CON有2位会受到硬件的影响：SI位在I2C硬件请求一个串行中断时被置位，STO位在总线上出现停止条件或者ENSI=0时被清除。

EI　　中断使能

ENSI　　设置使能I2C串行功能模块。当ENSI=1，I2C串行功能使能。复用管脚SDA与SCL必须设置为I2C功能。

STA　　I2C起始控制位。 设置STA为逻辑1，I2C进入主机模式，I2C 硬件向总线发送起始信号或者重复起始信号。

STO　　I2C停止控制位，在主机模式下，置位STO发送一个停止条件到总线上，然后I2C硬件将会检测总线条件，如果一个停止条件被检测到，该标志将会被硬件自动清零。在从机模式，STO标志被置位将复位I2C硬件至不可寻址从机模式。这意味着该设备不再处于从机接收模式，不能从主发送设备接收数据。

SI　　I2C中断标志。当有新的SIO状态出现在I2CSTATUS寄存器中时，SI标志由硬件置位，如果EI（I2CON［7］）置位，则产生I2C中断请求。SI 必须由软件清零。向该位写1清零SI。

AA　　发起应答控制位。若在地址或数据接收之前，AA=1，则在下列情况下：1.从机应答主机发送的地址信息 2.接收设备应答发送设备发送的数据 时将在SCL线的应答时钟脉冲期间返回应答信号（SDA为低）；若在地址或数据接收之前，AA=0, 则在SCL线的应答时钟脉冲期间返回不应答信号（SDA为高）。

6.5.4.4 *状态寄存器(I2CSTATUS)*

I2CSTATUS [7:0] 是一个8-位只读寄存器.。低3位一直为0。位域I2CSTATUS[7:3]包含状态码。有26个可能的状态码。当I2CSTATUS [7:0]的内容是F8H时，没有串行中断请求。所有其它的I2CSTATUS [7:3]的值对应于定义的I^2C状态。 所有的状态在表6.5-1列出。当进入这些状态中的任一个，就会产生状态中断请求(SI = 1)。在SI被硬件置位1个机器周期后，有效状态码出现在I2CSTATUS [7:3]中，并保持稳定至SI被软件清除的下一个机器周期。

另外，00H状态表示总线错误。总线错误发生在起始或停止信号出现在帧结构非法的位置。非法位置比如是在串行传输地址字节，数据字节或应答位期间。为了将I2C从总线错误中恢复，需要置位STO，清除SI从而进入不可寻址从机模式，然后清除STO释放总线并等待新的通信。I2C总线在总线错误时不能识别停止信号。

主模式		从模式	
状态	描述	状态	描述
0x08	Start	0xA0	Slave Transmit Repeat Start or Stop
0x10	Master Repeat Start	0xA8	Slave Transmit Address ACK
0x18	Master Transmit Address ACK	0xB0	Slave Transmit Arbitration Lost
0x20	Master Transmit Address NACK	0xB8	Slave Transmit Data ACK
0x28	Master Transmit Data ACK	0xC0	Slave Transmit Data NACK
0x30	Master Transmit Data NACK	0xC8	Slave Transmit Last Data ACK
0x38	Master Arbitration Lost	0x60	Slave Receive Address ACK
0x40	Master Receive ACK	0x68	Slave Receive Arbitration Lost
0x48	Master Receive NACK	0x80	Slave Receive Data ACK
0x50	Master Receive ACK	0x88	Slave Receive Data NACK
0x58	Master Receive NACK	0x70	GC mode Address ACK
0x00	Bus error	0x78	GC mode Arbitration Lost
		0x90	GC mode Data ACK
		0x98	GC mode Data NACK
0xF8	Bus Released Note: Status “0xF8” exists in both master/slave modes, and it won’t raise interrupt.		

表 6.5-1 I^{2}C 状态码描述表

6.5.4.5 *I2C 时钟波特率位 (I2CLK)*

当I^{2}C在主机模式下，I2C数据的波特率由I2CLK[7:0]寄存器决定。在从机模式下时I2CLK[7:0]的值不重要；在从机模式下，I2C将自动与主机I2C设备时钟频率同步，频率可高达1MHz。

I2C数据波特率设定是：I2C的数据波特率 = PCLK /(4x(I2CLK[7:0]+1))，如果PCLK=16MHz, I2CLK[7:0]= 40(28H)，I2C的数据波特率I2C = 16MHz /(4X (40 +1)) = 97.5K比特/秒.

6.5.4.6 *The I2C超时计数寄存器 (I2CTOC)*

有一个14位的超时计数器可以用于处理I2C总线挂起。当计数功能使能后，计数器开始计数直至发生超时，此时TIF置1，并向CPU产生I2C中断或者通过将ENTI清除为0关闭计数功能。当超时计数器使能后，设定SI标志为高会复位计数器，清零SI之后计数器会重新开始计数。如果I2C总线挂起，会使I2STATUS及SI标志在一段时间内不再更新。该14位超时计数器可能溢出并向CPU发出I2C中断请求。关于14位超时计数器参考图6.5-9，用户可通过对TIF位写1清0该标志。

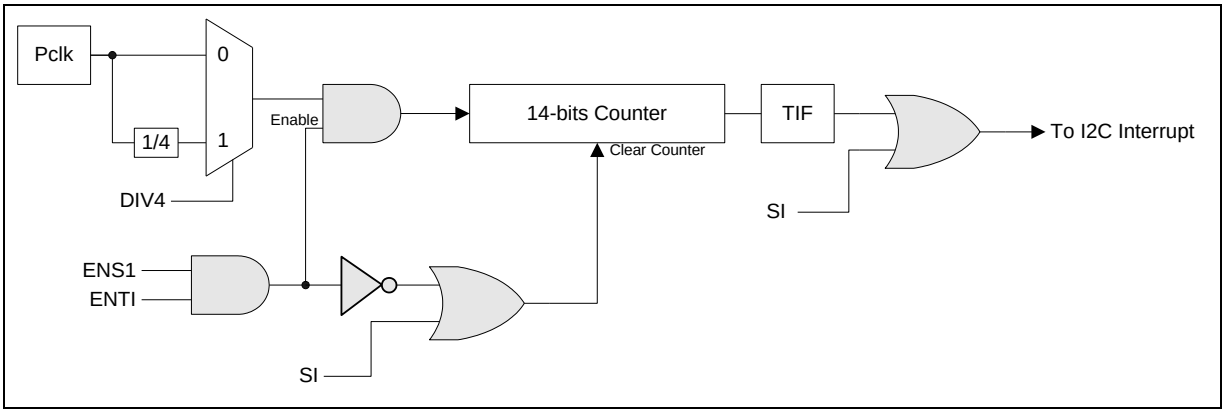


图 6.5.4-2: I2C 超时计数器框图

6.5.5 I2C 控制器寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位后的值
I2CON	I2C_BA+0x00	R/W	I2C控制寄存器	0x0000_0000
I2CADRR0	I2C_BA+0x04	R/W	I2C 从机地址寄存器0	0x0000_0000
I2CDAT	I2C_BA+0x08	R/W	I2C 数据寄存器	0x0000_0000
I2CSTATUS	I2C_BA+0x0C	R	I2C 状态寄存器	0x0000_00F8
I2CLK	I2C_BA+0x10	R/W	I2C 时钟时钟分频寄存器	0x0000_0000
I2CTOC	I2C_BA+0x14	R/W	I2C 超时控制寄存器	0x0000_0000
I2CADDR1	I2C_BA+0x18	R/W	从机地址寄存器1	0x0000_0000
I2CADDR2	I2C_BA+0x1C	R/W	从机地址寄存器2	0x0000_0000
I2CADDR3	I2C_BA+0x20	R/W	从机地址寄存器3	0x0000_0000
I2CADM0	I2C_BA+0x24	R/W	从机隐藏地址寄存器0	0x0000_0000
I2CADM1	I2C_BA+0x28	R/W	从机隐藏地址寄存器1	0x0000_0000
I2CADM2	I2C_BA+0x2C	R/W	从机隐藏地址寄存器2	0x0000_0000
I2CADM3	I2C_BA+0x30	R/W	从机隐藏地址寄存器3	0x0000_0000

6.5.6 I2C 控制器寄存器描述

I2C控制寄存器(I2CON)

寄存器	偏移量	R/W	描述	复位后的值
I2CON	I2C_BA+0x00	R/W	I2C 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	**ENSI**	**STA**	**STO**	**SI**	**AA**	保留	

Bits	描述	
[31:8]	保留	保留
[7]	**EI**	使能中断 1 = 使能I^2C中断功能 0 = 禁用I^2C中断功能
[6]	**ENSI**	**I2C 控制使能位** 1 = 使能 0 = 禁用 当ENSI=1 I2C串行功能使能，SDA和SCL 管脚必须设置为I2C功能
[5]	**STA**	I2C 起始控制位 STA置1 ，进入主机模式，如果总线处于空闲状态，I2C硬件会送出起始信号或重复起始信号.
[4]	**STO**	I2C 停止控制位 在主机模式下，置位STO将向总线传输停止条件，进而I2C硬件会检查总线状态。一旦检测到停止条件，该位将被硬件自动清零。在从机模式下，置位STO会将I2C硬件复位至不可寻址从机模式。这意味着该设备不再处于从机接收模式，不能从主发送设备接收数据。
[3]	**SI**	**I2C 中断标志位**

Bits	描述	
		I2CSTATUS 寄存器有新的SIO状态时，硬件置位SI标志。如果EI (I2CON [7])已经置位，就产生I2C中断请求。SI 必须由软件清零。向该位写1清零。
[2]	**AA**	接收应答控制位 若在地址或数据接收之前，AA=1，则在下列情况下：1.从机应答主机发送的地址信息 2.接收设备应答发送设备发送的数据 时将在SCL时钟的应答时钟脉冲间返回应答信号（SDA为低）；若在地址或数据接收之前，AA=0, 则在SCL的应答时钟脉冲间不会返回应答信号（SDA为高）。
[1:0]	保留	保留

I2C 数据寄存器 (I2CDAT)

寄存器	偏移量	R/W	描述	复位后的值
I2CDAT	I2C_BA+0x08	R/W	I2C 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]							

Bits	描述	
[31:8]	保留	保留
[7:0]	**I2CDAT**	**I2C** 数据寄存器 Bit[7:0] 为8位I2C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	复位后的值
I2CSTATUS	I2C_BA+0x0C	R/W	I2C 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]							

Bits	描述	
[31:8]	保留	保留
[7:0]	**I2CSTATUS**	**I2C状态寄存器** 低三位始终是0；高5位包含状态码。总共有26种状态码；当I2CSTATUS的值是F8H，表示没有串行中断请求；其它的所有的I2CSTATUS值可以反映I2C的状态。当进入这些状态时会产生一个状态中断请求(SI=1)。一个有效的状态码在SI被硬件设为'1'后一个周期内反映到I2CSTATUS中，并保持稳定至SI被软件清零的下一个周期。另外，状态码是00H时表示总线错误；当'起始'或'结束'时出现帧结构的非法位置时会产生总线错误。比如在串行传输地址字节中出现的数据字节或应答位就是非法的。

I2C波特率控制寄存器(I2CLK)

寄存器	偏移量	R/W	描述	复位后的值
I2CLK	I2C_BA+0x10	R/W	I2C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]							

Bits	描述	
[31:8]	保留	保留
[7:0]	**I2CLK**	**I2C** 波特率控制寄存器 I2C波特率 = PCLK /(4x(I2CLK+1)). 注意：I2CLK的最小值是4

I2C超时计数寄存器 (I2CTOC)

寄存器	偏移量	R/W	描述	复位后的值
I2CTOC	I2C_BA+0x14	R/W	I2C超时计数寄存器	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**

Bits	描述	
[31:3]	**保留**	保留
[2]	**ENTI**	**超时计数使能/禁用** 1 = 使能 0 = 禁用 当计数器被使能，SI被清0后，14位超时计数寄存器开始计数。对SI置1会使计数器复位，在SI清零后计数器重新开始计数
[1]	**DIV4**	**超时计数输入时钟除4** 1 = 使能 0 = 禁用 使能后，溢出时间延长4倍。
[0]	**TIF**	**超时标志** 当I^2C超时发生时，这个位由硬件置位，如果I^2C中断同时使能，可引发CPU的中断软件写"1"清零.

I2C 从机地址寄存器(I2CADDRx)

寄存器	偏移量	R/W	描述	复位后的值
I2CADDR0	I2C_BA+0x04	R/W	I2C 从机地址寄存器0	0x0000_0000
I2CADDR1	I2C_BA+0x18	R/W	I2C从机地址寄存器1	0x0000_0000
I2CADDR2	I2C_BA+0x1C	R/W	I2C 从机地址寄存器2	0x0000_0000
I2CADDR3	I2C_BA+0x20	R/W	I2C 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**

Bits	描述	
[31:8]	保留	保留
[7:1]	**I2CADDR**	I2C地址寄存器: 主机模式下，该寄存器的值无效，从机模式下，高七位为MCU自身地址，如果地址匹配，I2C硬件会响应.
[0]	**GC**	广播呼叫功能. 0: 禁用广播呼叫功能. 1: 允许广播呼叫功能

I2C 从机隐藏地址寄存器(I2CADMx)

寄存器	偏移量	R/W	描述	复位后的值
I2CADM0	I2C_BA+0x24	R/W	I2C s从机地址掩码寄存器0	0x0000_0000
I2CADM1	I2C_BA+0x28	R/W	I2C从机地址掩码寄存器1	0x0000_0000
I2CADM2	I2C_BA+0x2C	R/W	I2C从机地址掩码寄存器2	0x0000_0000
I2CADM3	I2C_BA+0x30	R/W	I2C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**
I2CADMx[7:1]							保留

Bits	描述	
[31:8]	保留	保留
[7:1]	**I2CADMx**	I2C 隐藏地址寄存器: 1 = 掩码使能(不关心收到的相应地址位) 0 = 掩码禁止(接收到的地址必须完全符合正确的地址内容) I2C总线支持多掩码地址识别。当设置允许掩码时，接收到的从机地址是否正确不予关心，当选择为禁用掩码时，从机地址必须完全符合其真实的地址才给予响应.
[0]	保留	保留

6.5.7 操作模式

片上I2C端口支持5种操作模式：主机发送，主机接收，从机发送，从机接收和广播呼叫模式。

在实际应用中，I2C端口可以作为主机或从机。在从机模式，I2C端口寻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。在主控芯片要成为总线主机时，在进入主机模式之前，硬件等待总线空闲以使可能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从机地址。

SI标志被清除以后，I2CON寄存器中的比特STA,STO 和AA将决定I2C硬件的下一个状态。新的动作完成时，一个新的状态码将被更新，SI将再次被置。如果I2C中断控制位EI(I2CON [7])被置，状态码的处理以及下一个I2C硬件状态可以在中断处理函数中进行。

下图描述了五种操作模式以及它们的详细数据流程

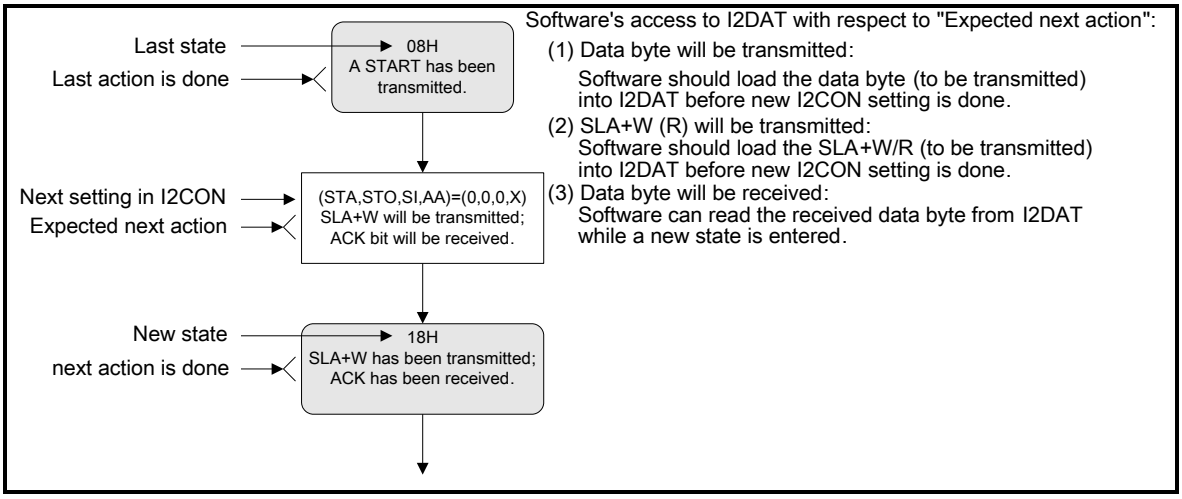


图 6.5.7-1对如下五图的说明

6.5.7.1 *主机发送模式*

当SCL线上输出串行时钟时，数据通过SDA线输出。第一个发送的字节包含从设备的地址（7位）和数据传输方向位（1位）。在该模式下，方向位(R/W)为0，在图6.5.7-2中以“W”表示。这样发送的第一个字节为SLA+W。串行数据一次发送8位。在每个字节发送完成后，将接收到一个应答位。起始和停止条件将被输出以表明串行传输的开始和结束。

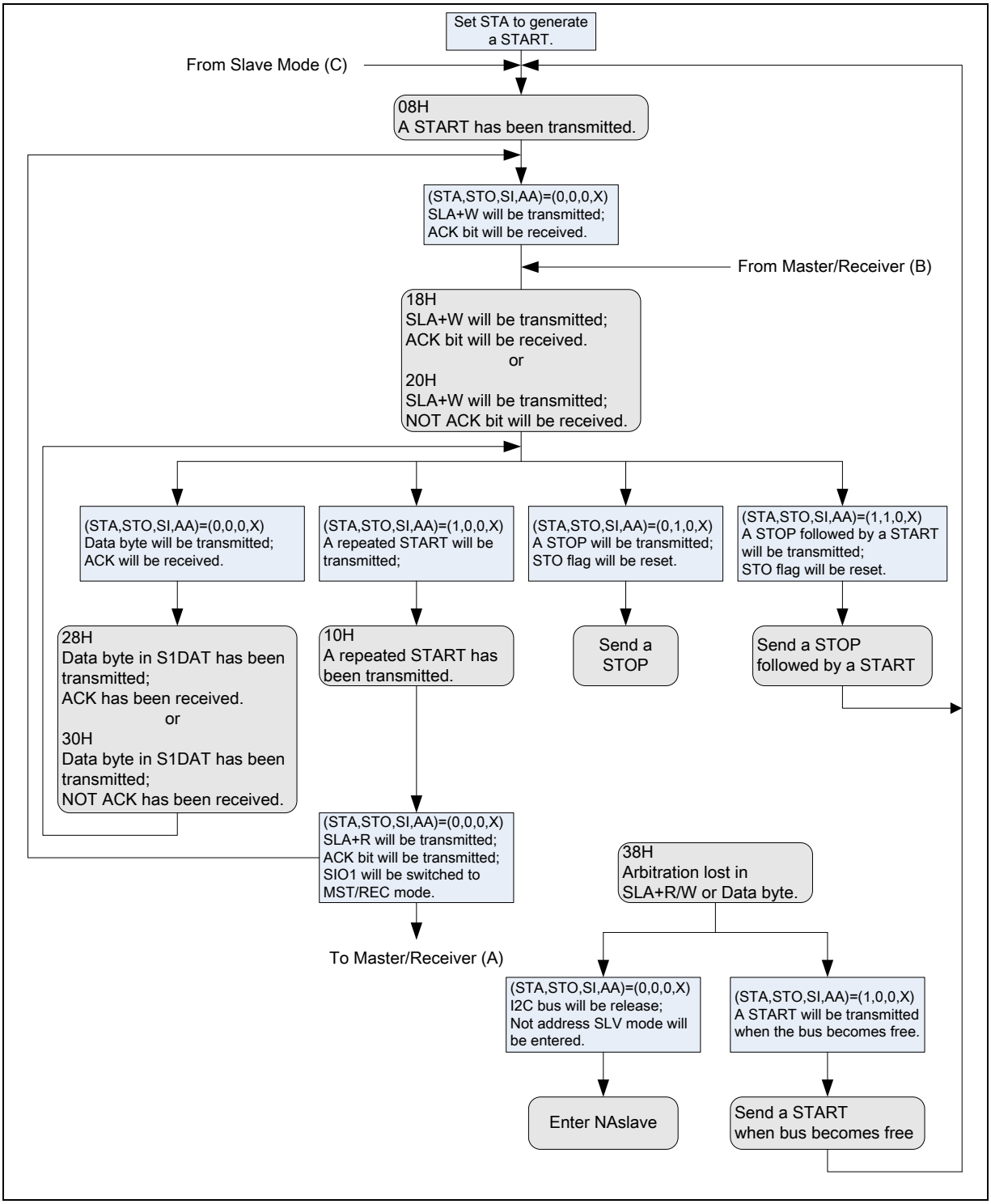


图 6.5.7-2 主机发送模式

6.5.7.2 *主机接收模式*

在该模式下，方向位(R/W)为1，在图6.5.7-3中以“R”表示。这样发送的第一个字节为SLA+R。当SCL

线上输出串行时钟时，数据通过SDA线接收。串行数据一次接收8位。在每个字节接收完成后，一个应答位将被发送。起始和停止条件将被输出以表明串行传输的开始和结束。

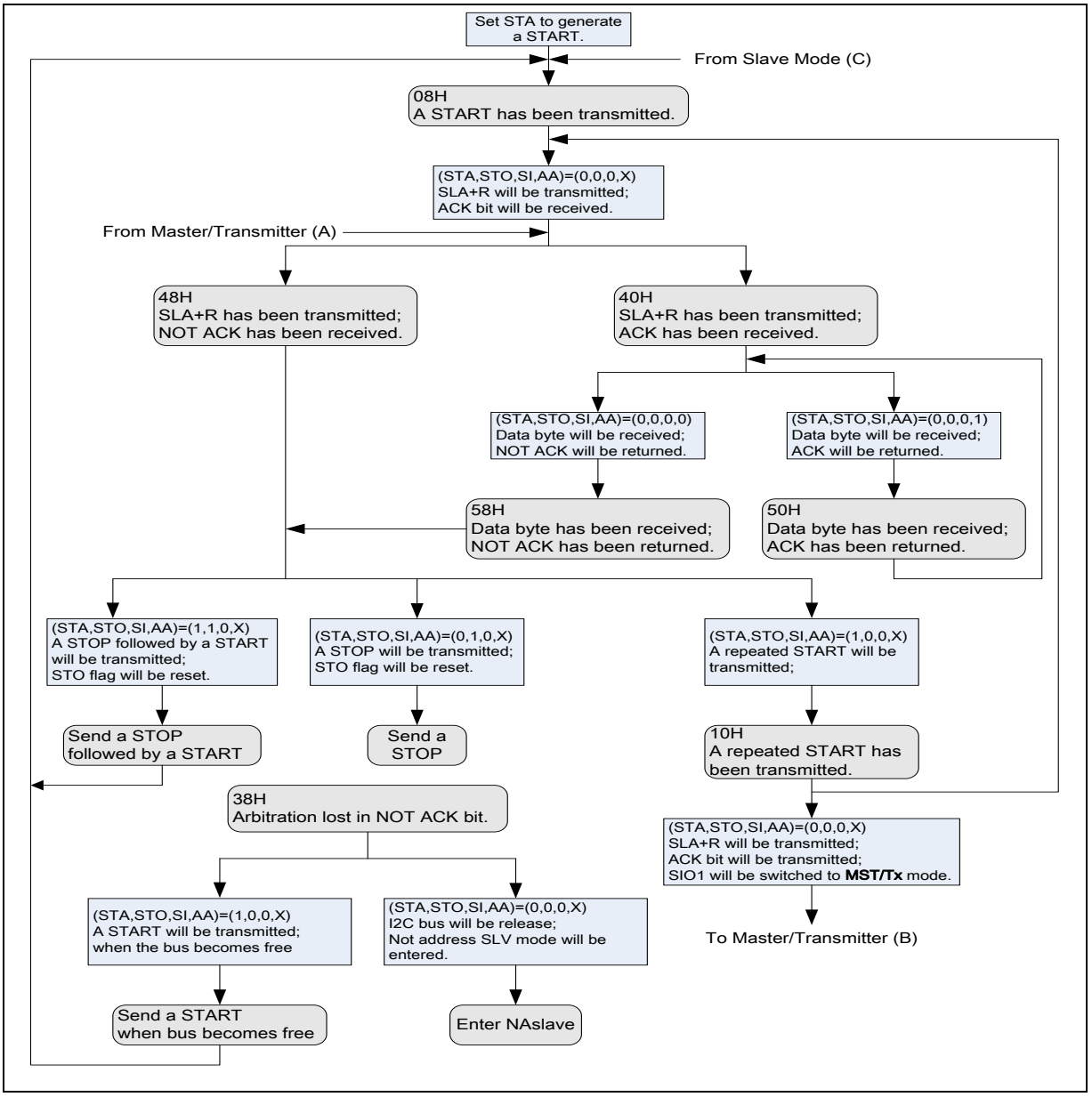


图 6.5.7-3 主机接收模式

6.5.7.3 *从机发送模式*

对第一个字节的接收和处理跟在从机接收模式一样。然而，在该模式下，数据传输方向位用来标示的传输方向是颠倒的。当串行时钟通过SCL输入时，串行数据通过SDA被发出。起始和停止条件将被认为是串行传输的开始和结束

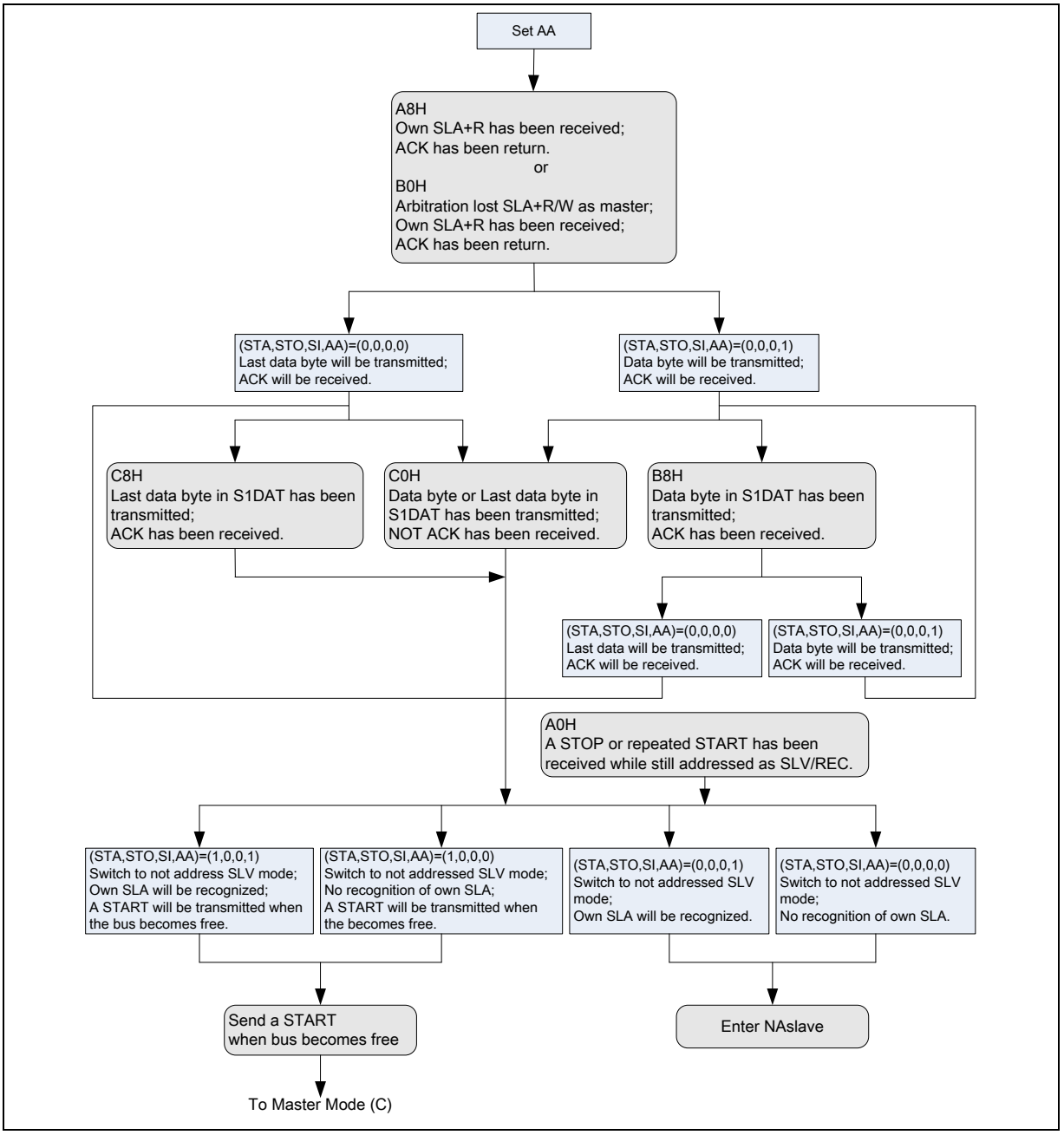


图 6.5.7-4 从机发送模式

6.5.7.4 *从机接收模式*

在该模式下，串行数据和串行时钟通过SDA和SCL接收。在接收到一个字节后，一个应答位将被发送。起始和停止条件将被认为是串行传输的开始和结束。地址识别将在从地址和数据传输方向位接收到时由硬件执行。

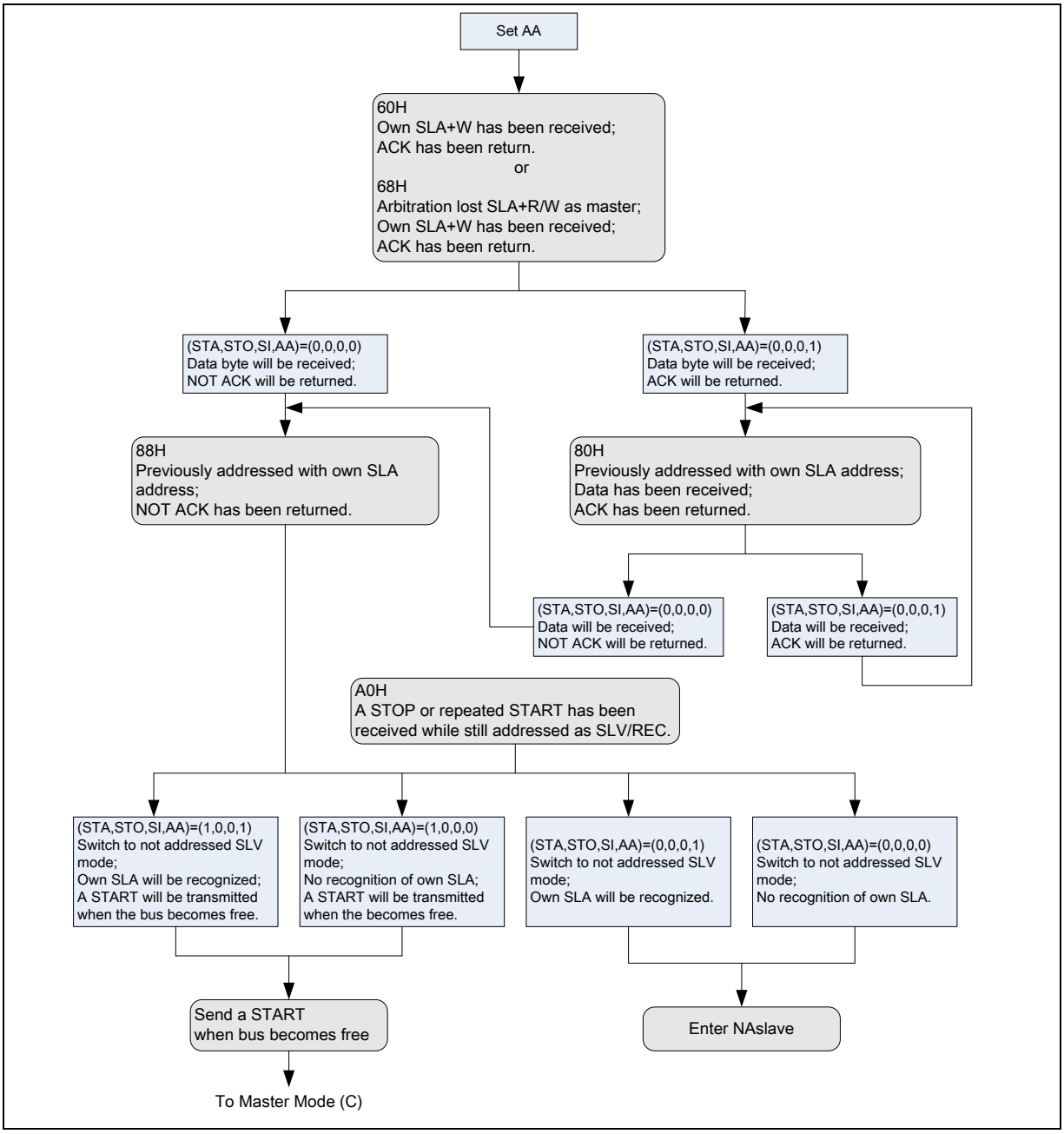


图 6.5.7-5 从机接收模式

6.5.7.5 *广播呼叫 (GC) 模式*

如**錯誤! 找不到參照來源**。所示，如果 GC bit (I2CADDRn [0]) 被设，I^2C 端口硬件将响应广播呼叫地址 (00H).当GC bit被设并且I^2C在从模式时，当主机发送广播呼叫地址到I^2C总线上时，可以接收广播呼叫地址00H. 串行数据和串行时钟从SDA和SCL接收. 每个字节被接收以后，应答bit将被发送. START 和 STOP 条件被识别为传输的开始和结束. 接收到地址和方向位以后，地址识别由硬件完成

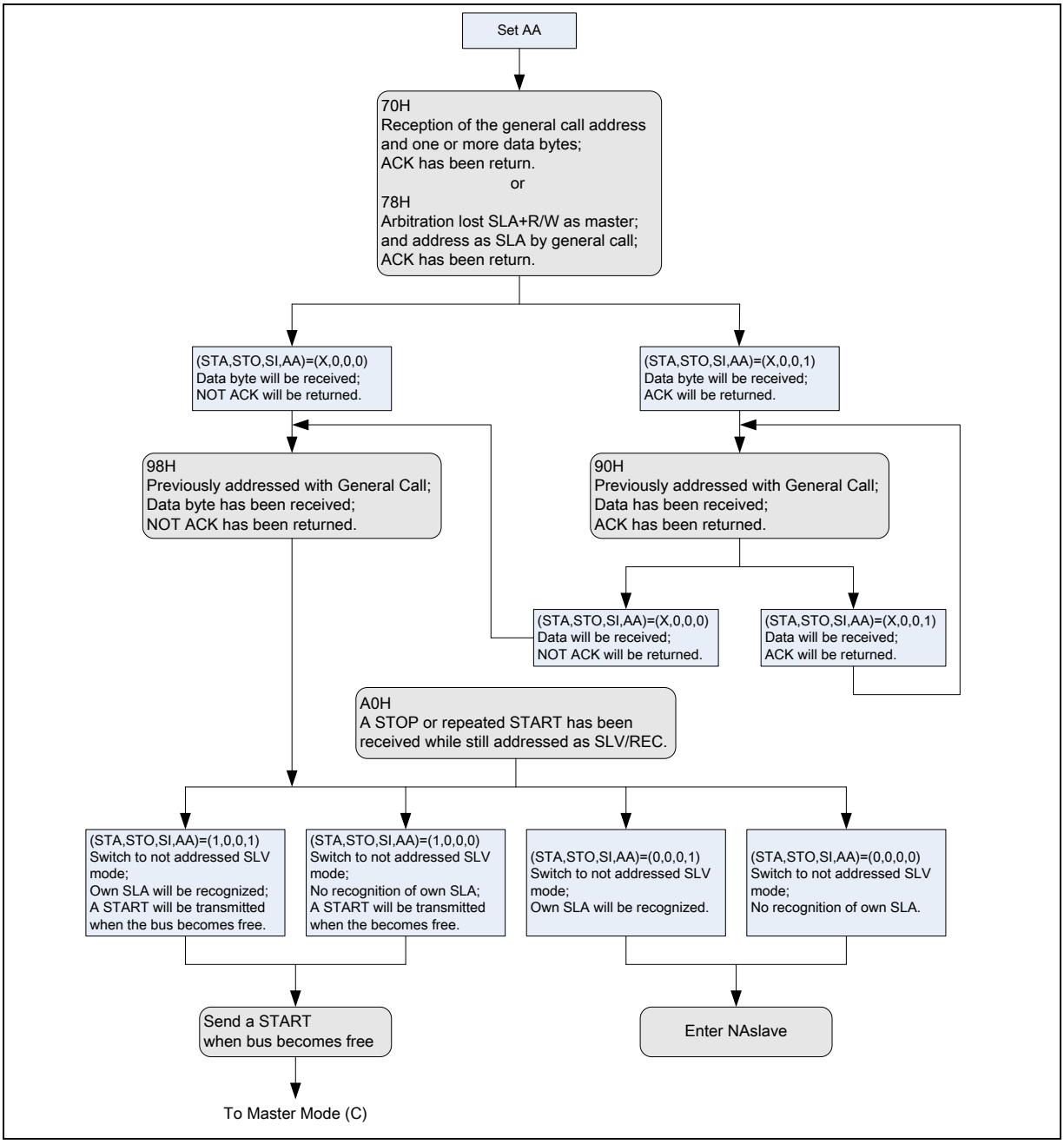


图 6.5.7-6 广播呼叫模式

6.6 PWM 发生器和捕捉定时器

6.6.1 概述

NuMicro M051™ 系列有2个PWM组，共有4组PWM发生器，可配置成8个独立的PWM输出，PWM0~PWM7, 或者4个互补的PWM对，(PWM0, PWM1), (PWM2, PWM3), (PWM4, PWM5) 和 (PWM6, PWM7)，带4个可编程的死区发生器.

每组PWM发生器带有8位预分频器，一个时钟分频器提供5种分频(1, 1/2, 1/4, 1/8, 1/16), 两个PWM定时器包括2个时钟选择器，两个16位PWM向下计数计数器用于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。4组PWM发生器提供8个独立的PWM中断标志，这些中断标志当相应的PWM周期向下计数器达到零时由硬件置位。每个PWM中断源和它相应的中断使能位可以引起CPU请求PWM中断。PWM发生器可以配置为单触发模式产生仅仅一个PWM周期或自动重载模式连续输出PWM波形。

当PCR.DZEN01置位, PWM0 与 PWM1执行互补的PWM对功能，这一对PWM的时序，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)，(PWM4, PWM5) 与 (PWM6, PWM7) 分别由 PWM2，PWM4 与 PWM6 定时器和死区发生器2，4，6控制，参考下图查看PWM定时器架构。

为防止PWM输出不稳定波形，16位向下计数器和16位比较器采用双缓存。当用户向计数器/比较器缓冲寄存器内写入值，只有当向下计数器的值达到0时，被更新的值才会被装载到16位计数器/比较器。该双缓冲特性避免PWM输出波形上产生毛刺。

当16位向下计数器达到0时，中断请求产生。如果PWM定时器被配置为自动重装载模式，当向下计数器达到0时，会自动重新装载PWM计数器寄存器(CNRx)的值，并开始递减计数，如此连续重复。如果定时器设为单触发模式，当向下计数器达到0时，向下计数器停止计数，并产生一个中断请求。

PWM计数器比较器的值用于高电平脉冲宽度调制，当向下计数器的值与比较寄存器的值相同时，计数器控制逻辑反转输出为高电平。

PWM定时器可复用为数字输入捕捉功能。如果捕捉功能使能，PWM的输出引脚将被切换至捕捉输入模式。捕捉器0和PWM0使用同一个定时器，捕捉器1和PWM1使用另一组定时器，以此类推。因此在使用捕捉功能之前，用户必须预先配置PMW定时器。捕捉功能使能后，捕捉器在输入通道的上升沿将PWM计数器值锁存至捕捉上升沿锁存寄存器(CRLR)，在输入通道的下降沿将PWM计数器值锁存至捕捉下降沿锁存寄存器(CFLR)。捕捉通道0中断是可编程的，通过设定CCR0.CRL_IE0[1] (上升沿锁存中断使能)和CCR0.CFL_IE0[2]] (下降沿锁存中断使能) 来决定中断发生的条件。 通过设置CCR0.CRL_IE1[17]和CCR0.CRL_IE1[18]，捕捉通道1有同样的特性。通过设置相应的控制位，每组的通道0到通道3有同样的特性。 对于每一组，不管捕捉何时产生中断0/1/2/3，PWM计数器0/1/2/3都将在该时刻重载。

最大的捕捉频率受捕捉中断延迟限制。捕捉中断发生时，软件至少要执行三个步骤：读PIIRx 以得到中断源，读PWM_CRLx/PWM_CFLx(x=0到3) 以得到捕捉值，写1清PIIRx。 如果中断延迟要花时间T0完成， 在这段时间内(T0)，捕捉信号一定不能翻转。在这种情况下，最大的捕捉频率将是1/T0。例如：

HCLK = 50 MHz, PWM_CLK = 25 MHz, 中断延迟时间 900 ns

因此最大的捕捉频率将是1/900ns ≈ 1000 kHz

6.6.2 特征

6.6.2.1 *PWM功能特性:*

PWM 组有两个PWM发生器。每个PWM发生器支持一个8位的预分频器，一个时钟分频器，两个PWM定时器（向下计数），一个死区发生器和两路PWM输出。

- 最高16位分辨率
- PWM 中断请求与PWM周期同步
- 单触发模式或自动重载模式
- 2个PWM组 (PWMA/PWMB) 支持8个PWM通道

6.6.2.2 *捕捉功能模块特性:*

- 与PWM发生器共享时序控制逻辑
- 8 路捕捉输入通道与8个PWM输出通道复用
- 每个通道支持一个上升沿锁存寄存器(CRLR)，一个下降沿锁存寄存器(CFLR)和捕捉中断标志(CAPIFx)

6.6.3 PWM 框图

图6.6.36.6.3-1按对说明PWM架构(定时器0&1为一对， 定时器2&3为另外一对，诸如此类)。

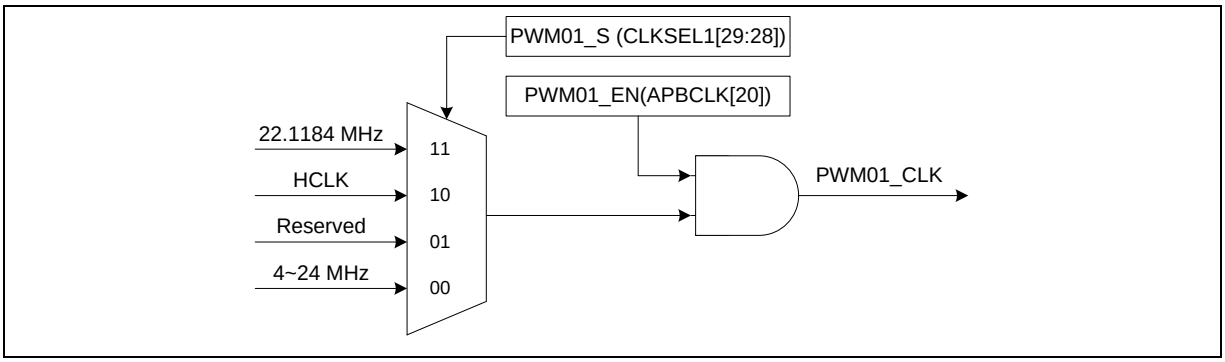


图 6.6.3-1 PWM 发生器 0 时钟源控制

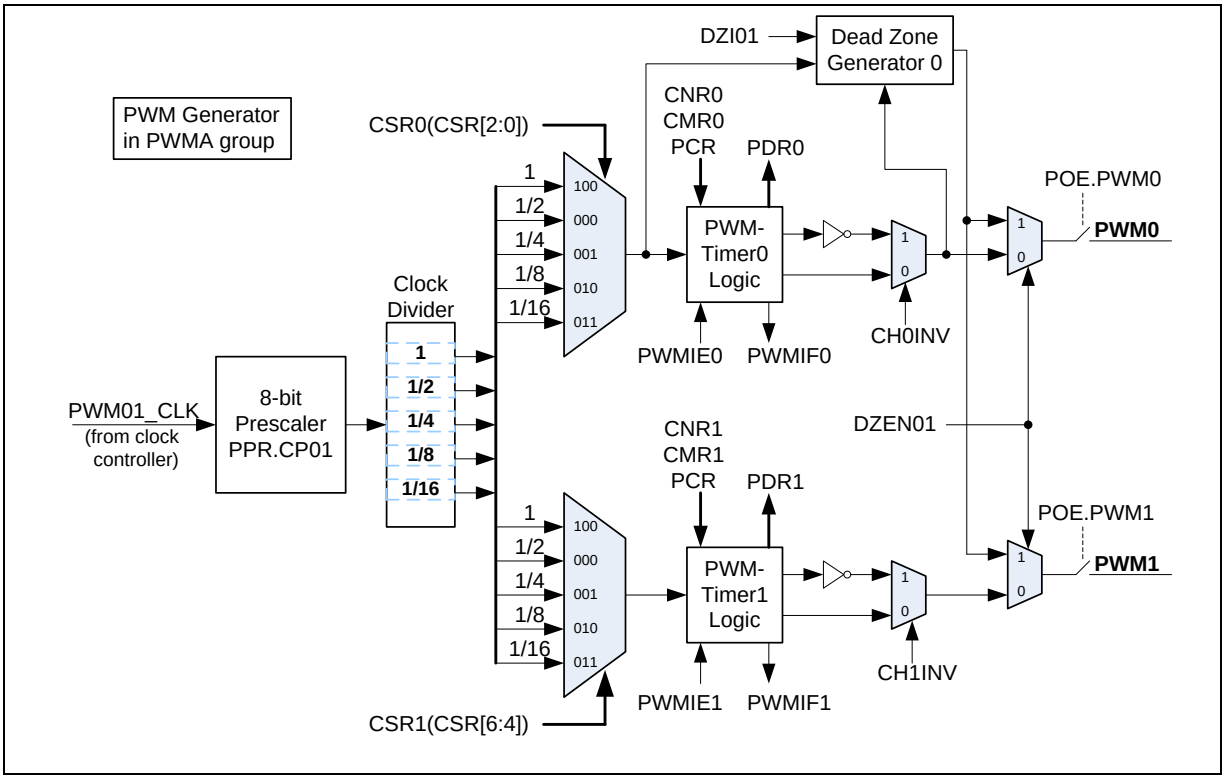


图 6.6.3-2 PWM 发生器 0 结构框图

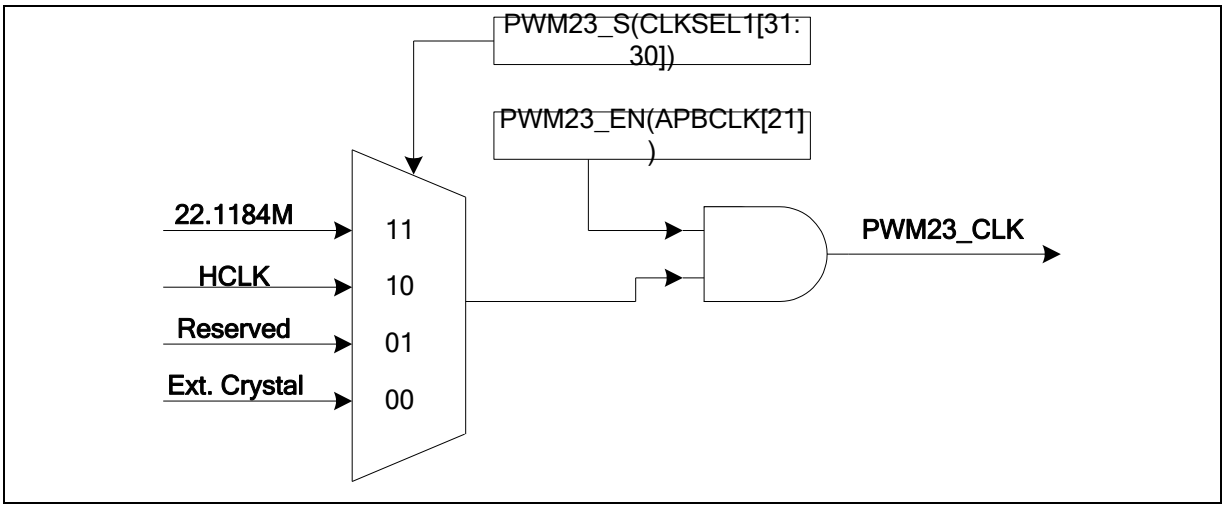


图 6.6.3-3 PWM 发生器 2 时钟源控制

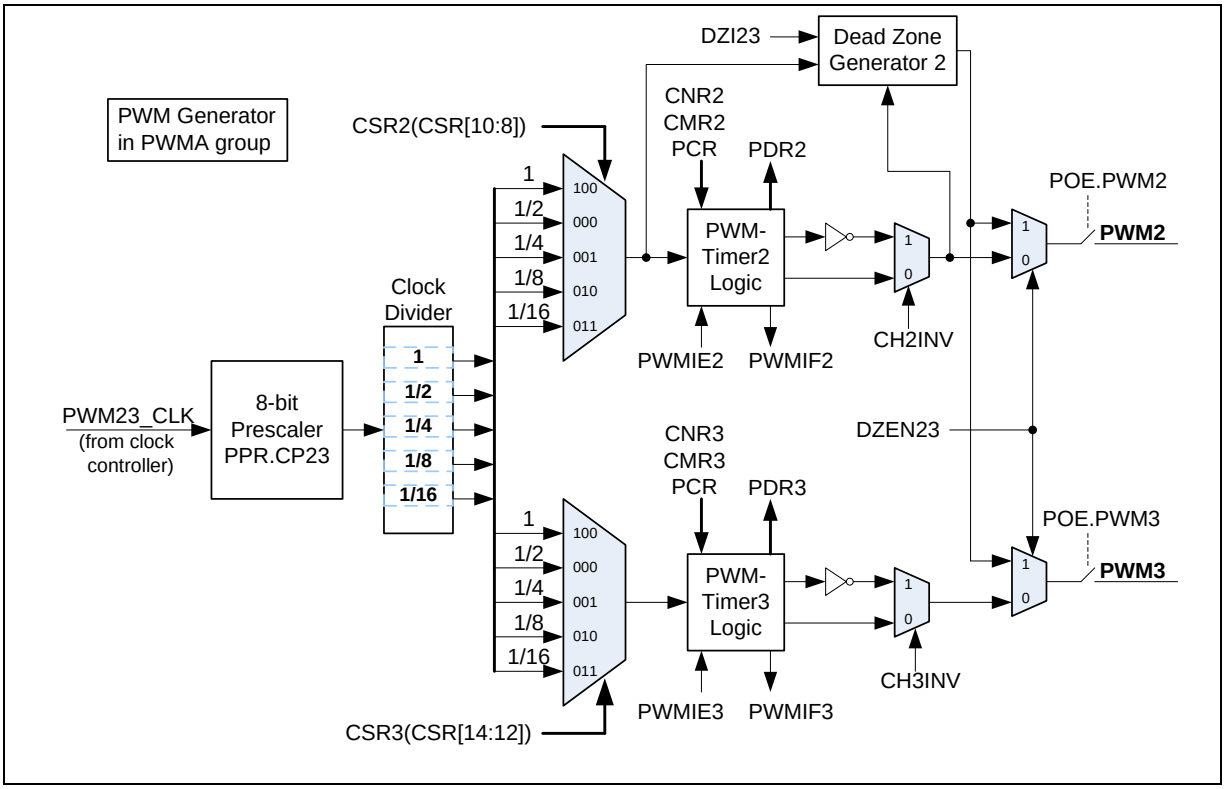


图 6.6.3-4 PWM 发生器 2 结构框图

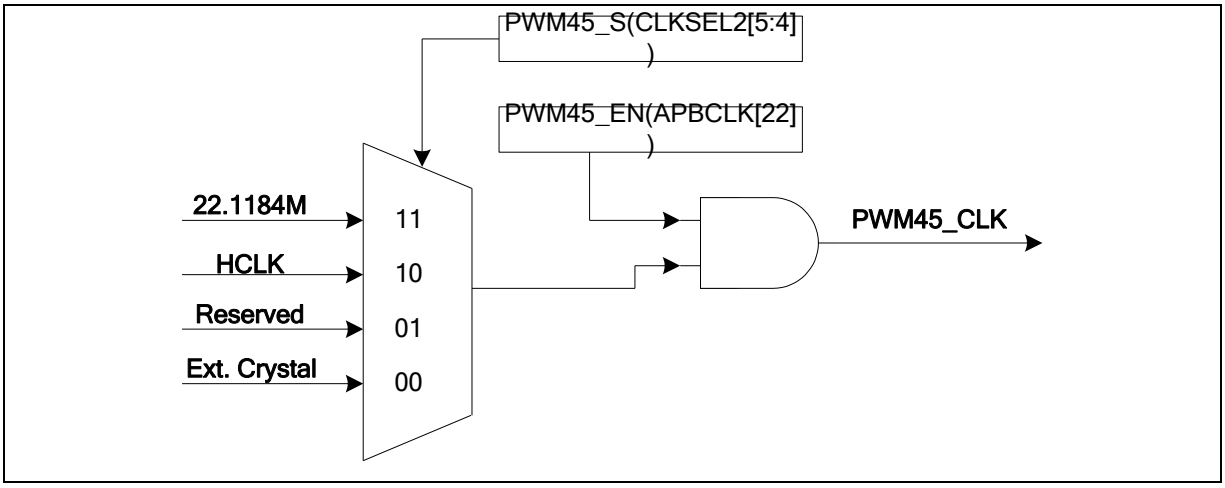


图 6.6.3-5PWM 发生器 4 时钟源控制

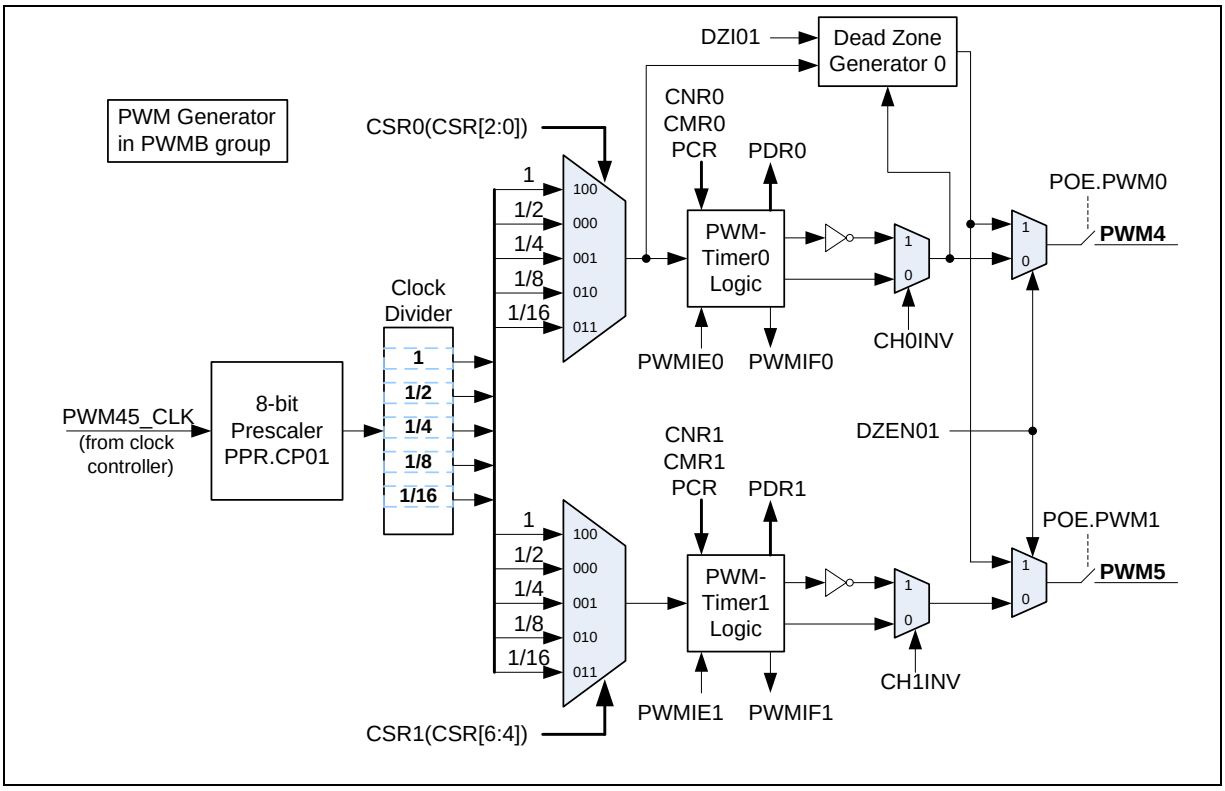


图 6.6.3-6 PWM 发生器 4 结构框图

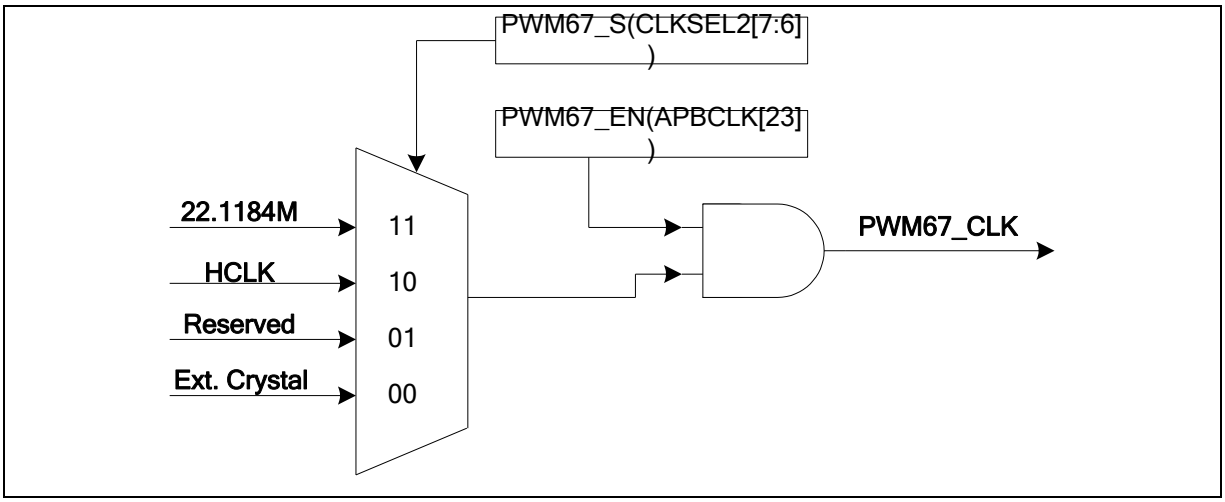


图 6.6.3-7 PWM 发生器 6 时钟源控制

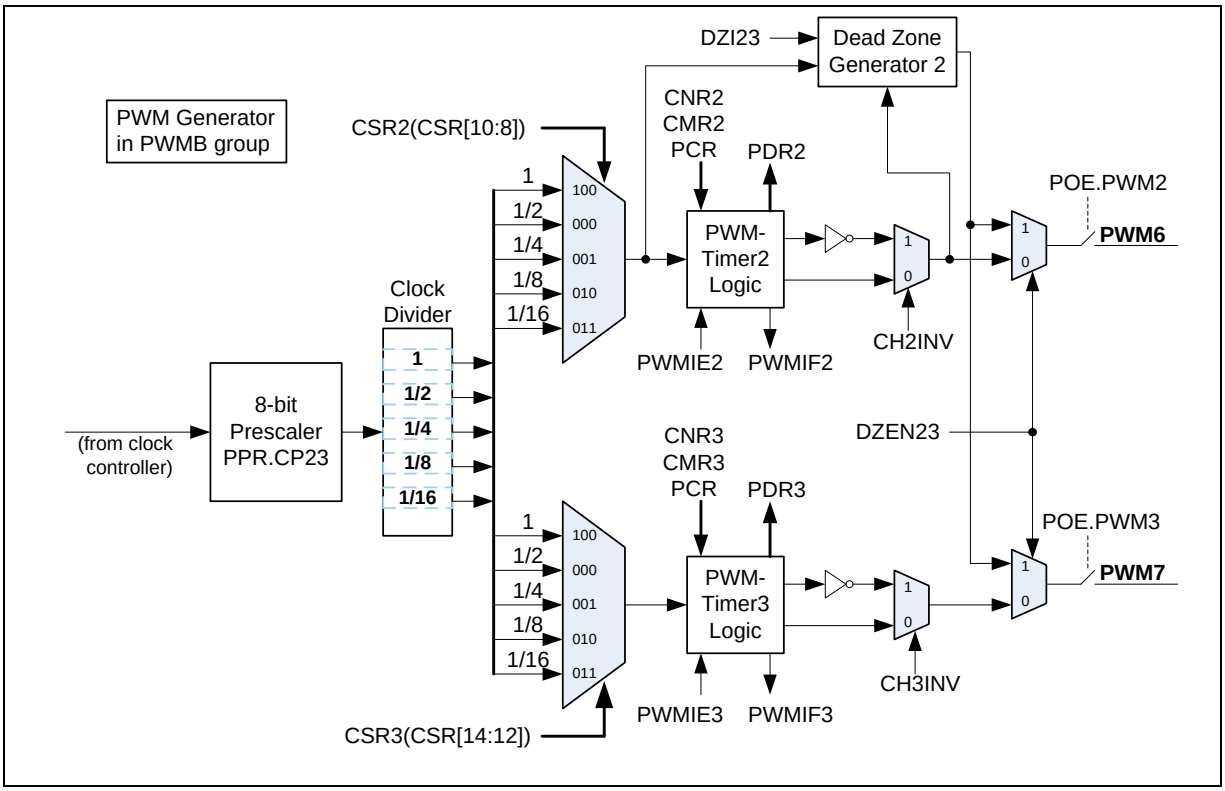


图 6.6.3-8 PWM 发生器 6 结构框图

6.6.4 PWM 功能描述

6.6.4.1 *PWM-定时器操作*

PWM 周期和占空比控制由PWM向下计数器寄存器(CNR)以及PWM比较寄存器(CMR)配置。PWM定时器工作时序如图6.6.4-2所示。. 脉宽调制的公式如下，PWM定时器比较器的说明如图**錯誤! 找不到參照來源**。所示。注意：相应的GPIO管脚必须配置成PWM功能(使能POE 和禁用CAPENR)。

PWM 频率 = PWMxy_CLK/((prescale+1)*(clock divider)*(CNR+1)); xy代表01, 23, 45 或 67, 取决于所选择的PWM通道.

- 占空比 = (CMR+1)/(CNR+1)
- CMR >= CNR：PWM输出为高
- CMR < CNR：PWM低脉冲宽度= (CNR-CMR) unit[1]; PWM高脉冲宽度 = (CMR+1) unit
- CMR = 0：PWM 低脉冲宽度 = (CNR) unit; PWM高脉冲宽度= 1 unit

注: 1. unit = 一个PWM时钟周期

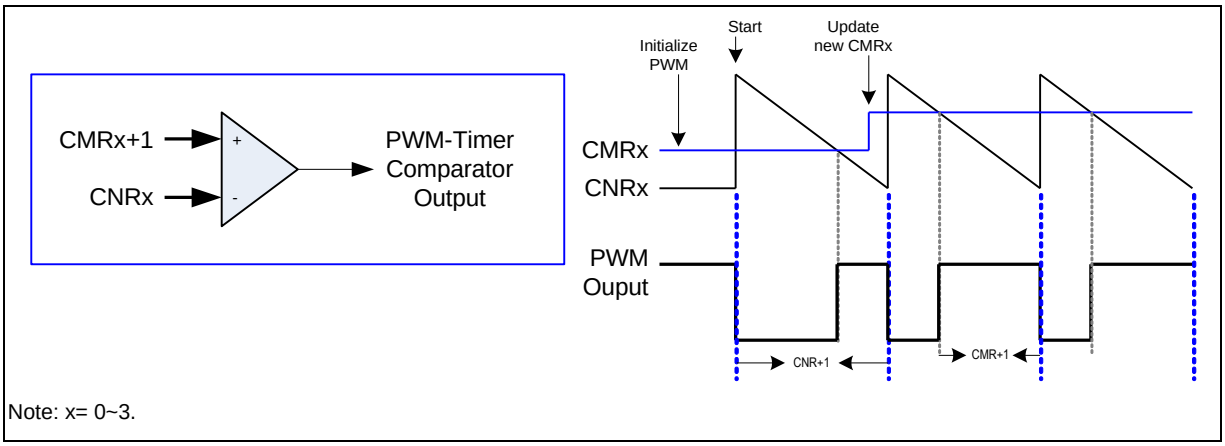


图 6.6.4-1PWM 定时器内部比较器输出

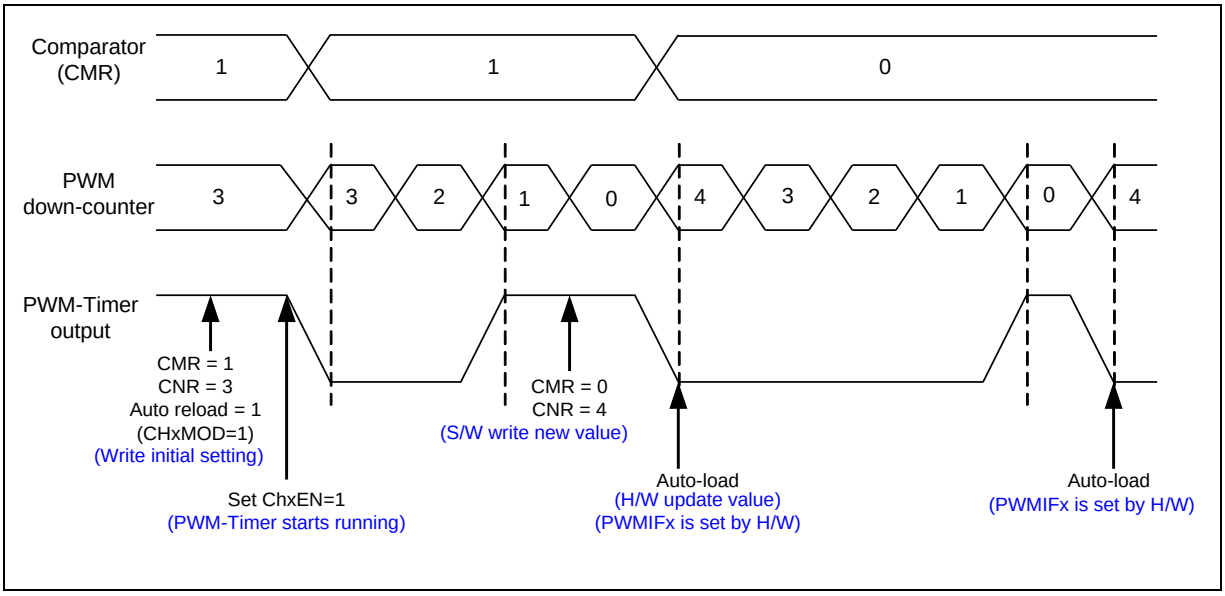


图 6.6.4-2 PWM 定时器操作时序

6.6.4.2 *PWM双缓存，自动重载以及单触发模式*

NuMicro M051™系列PWM定时器具有双缓存功能。新的值将在下一个周期开始时重新加载，不会影响当前定时器的工作。PWM计数器的值可写入CNRx，当前PWM计数器的值可以从PDRx读取。

PWM 控制寄存器(PCR) 的CH0MOD 位定义PWM0是自动重载模式还是单次触发模式。如果CH0MOD被设置为1，当PWM计数器达到0，自动重载CNR0的值到PWM计数器，如果要进行模式切换，推荐在将CH0EN设为1之前进行，因为当操作模式改变时，CNR0和CMR0的值将被清成0，此模式下应该先写CMR0和CNR0再使能CH0EN。如果CH0MOD被设定为0，当PWM计数器计数到0，计数器立即停止计数，这种模式下，应该先写CMR0和CNR0，然后再使能CH0EN，当PWM0从CNR0计数到0时，CNR0和CMR0的值将被硬件清成0，如果要再输出一个周期，软件需要重新写CNR0和CMR0的值，这时CMR0应该先写，再写CNR0，因为此时一旦CNR0为非0，PWM0将自动重新开始计数。自动重载模式下，如果CNR0被设定为0，当PWM计数器计数到0，计数器将停止计数。PWM1~PWM7 运行状态与PWM0 相同.，

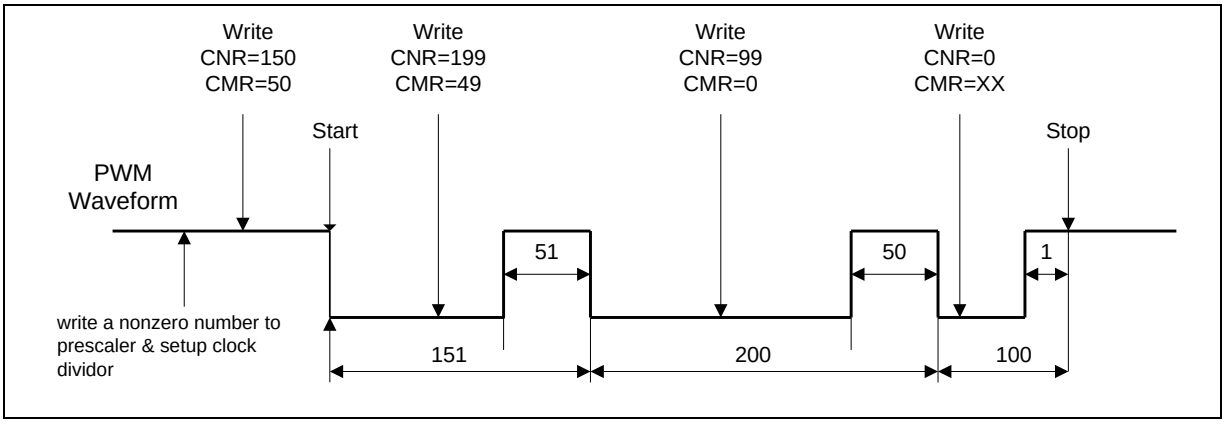


图 6.6.4-3 PWM 双缓存图解

6.6.4.3 *调制占空比*

双缓存功能允许CMRx在当前周期的任意时刻被写入。写入的值将在下个周期生效。

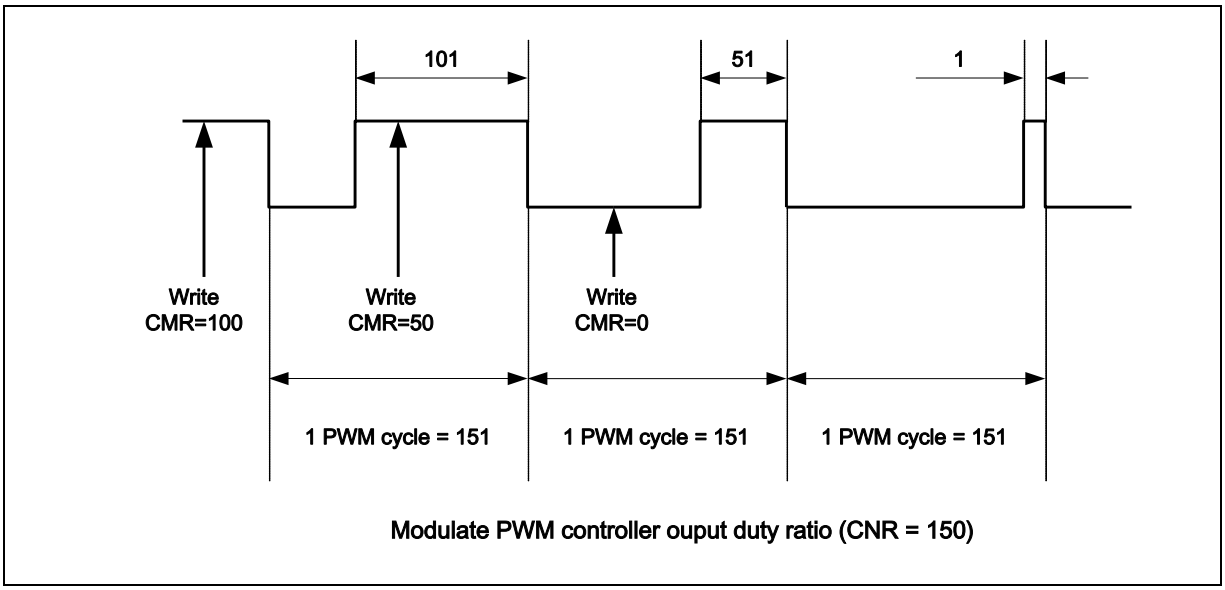


图 6.6.4-4 PWM 控制器输出占空比

6.6.4.4 *死区发生器*

NuMicro M051™系列提供PWM死区发生器，用于保护功率器件。该功能产生可编程的时隙来延迟PWM上升沿输出，用户可通过编程PPRx.DZI确定死区间隔。

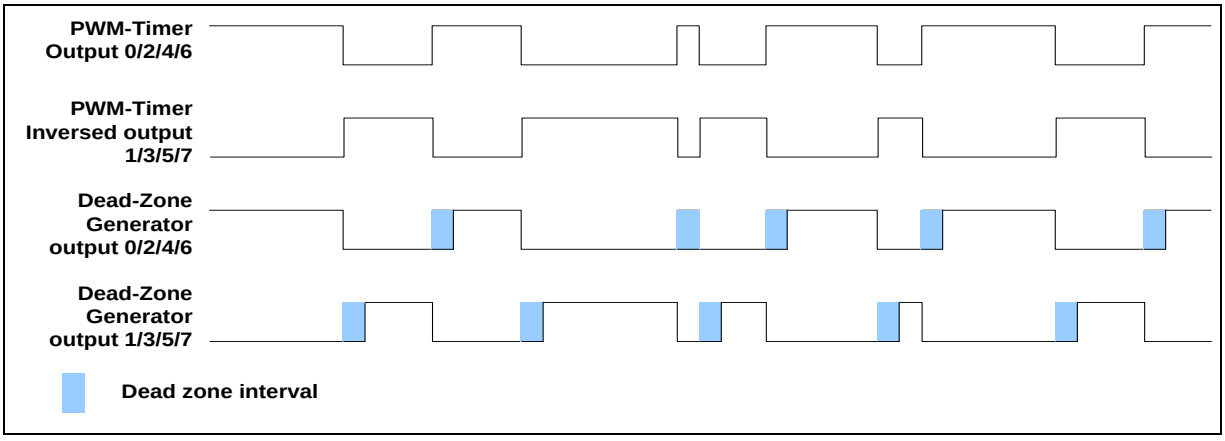


图 6.6.4-5 死区发生器操作

6.6.4.5 *捕捉操作*

捕捉器0和PWM0共用同一个定时器，捕捉器1和PWM1共用另一个定时器，以此类推。捕捉器总是在输

入通道产生一个上升跳变时将PWM计数器的值锁存至CRLRx，在输入通道产生一个下降跳变时将PWM计数器的值锁存至CFLRx。 捕捉通道0中断是可编程的，通过设定CCR0.CRL_IE0[1] (上升沿锁存中断使能) 和 CCR0.CFL_IE0[2]] (下降沿锁存中断使能) 来决定中断发生的条件。通过设置CCR0.CRL_IE1[17]和CCR0.CRL_IE1[18]，捕捉通道1有同样的特性。无论捕捉模块何时触发一个捕捉中断，相应的PWM计数器都将在此刻重载CNRx的值。注：相应的GPIO管脚必须配置成捕捉功能(禁用POE和使能CAPENR).

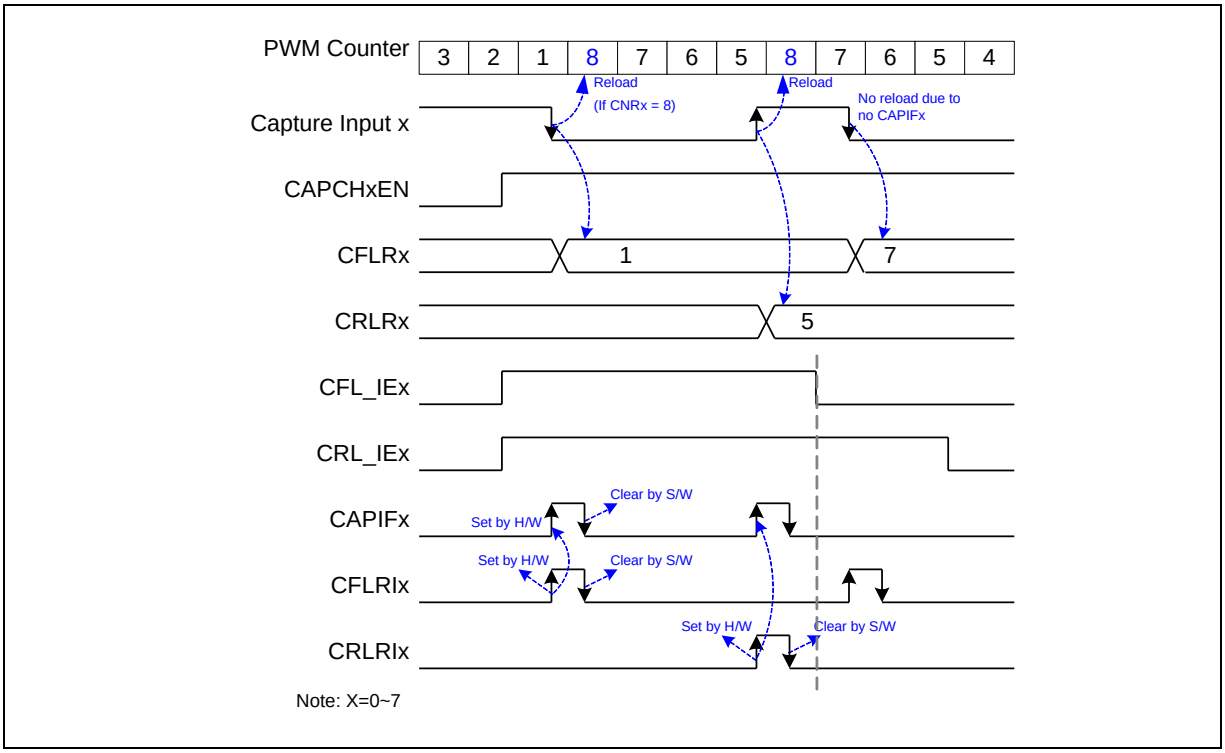


图 6.6.4-6 捕捉操作时序

在上述范例中，CNR 为8:

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

6.6.4.6 *PWM-定时器中断结构*

PWM 0 与捕捉器0共用同一个中断。PWM1 与捕捉器1 共用同一个中断，以此类推。因此，同一通道的PWM功能和捕捉功能不能同时使用。下图说明了 PWM定时器中断结构。提供8个PWM中断，PWM0_INT~PWM7_INT, 对于增强型中断控制寄存器(AIC)可分为PWMA_INT 与PWMB_INT。

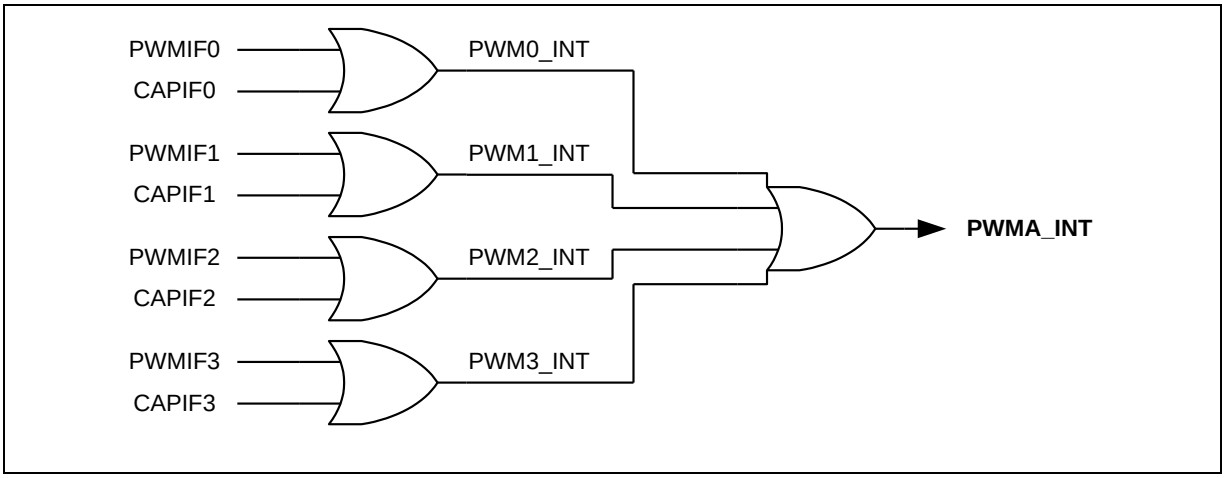


图 6.6.4-7 PWM A 组 PWM-定时器中断结构图

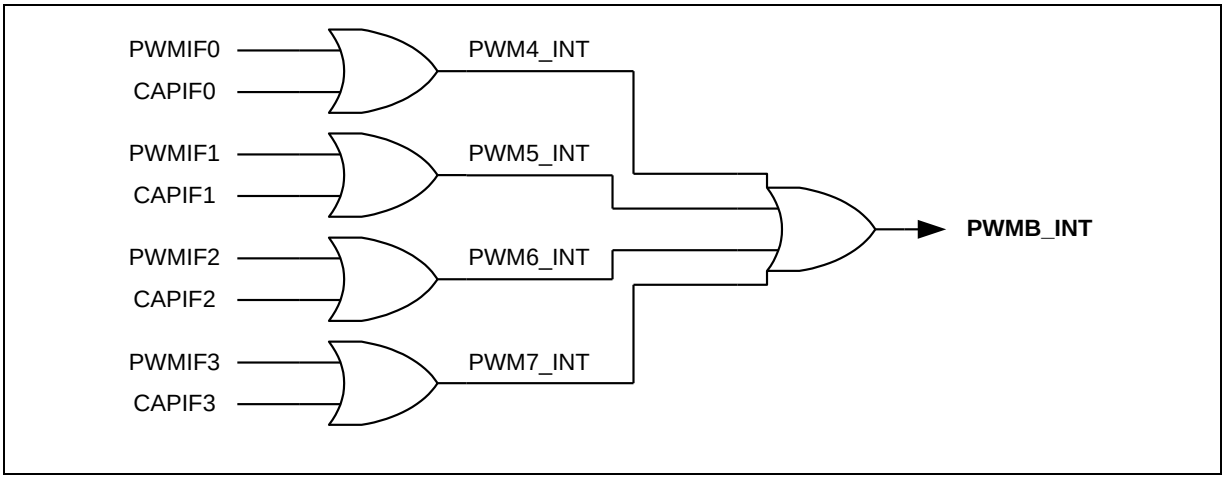


图 6.6.4-8 PWM B 组 PWM-定时器中断结构图

6.6.4.7 *PWM-定时器开启步骤*

推荐使用如下步骤启动PWM驱动器。

1. 配置时钟选择器(CSR)
2. 配置预分频器 (PPR)
3. 配置反向打开/关闭，死区发生器打开/关闭，自动重载/单触发模式，并停止PWM定时器 (PCR)
4. 配置比较器寄存器(CMR) 设定PWM占空比
5. 配置PWM计数器寄存器 (CNR) 设定PWM周期.
6. 配置中断使能寄存器(PIER)
7. 配置相应的GPIO管脚为PWM功能(使能 POE 和禁用CAPENR).
8. 使能PWM定时器开始运行 置PCR中的 CHxEN 为1

6.6.4.8 *PWM-定时器关闭步骤*

方式 1:

设定16位向下计数计数器(CNR)为0 ，并监视PDR（16位向下计数器的当前值）。当PDR达到0 ，关闭PWM定时器 (PCR的CHxEN位)。 ***(推荐)***

方式 2:

设定16位向下计数计数器(CNR)为0，当中断请求发生，关闭PWM定时器(PCR的CHxEN位)。 ***(推荐)***

方式 3:

直接关闭PWM定时器(PCR的CHxEN位)。(不推荐)

不推荐方式3的原因是：禁用CHxEN将立即停止PWM输出信号，引起PWM输出占空比的改变，这可能导致电机控制电路损坏。

6.6.4.9 *捕捉开始步骤*

- 配置时钟选择器(CSR)
- 配置预分频器(PPR)
- 配置通道使能，上升/下降沿中断使能以及输入信号反向打开/关闭 (CCR0, CCR1)
- 配置PWM下数计数器寄存器(CNR)
- 配置相应GPIO管脚为捕捉功能(禁用 POE和使能 CAPENR).
- 使能PWM定时器 开始运行(置PCR中的 CHxEN 为1)

6.6.5 PWM 控制器寄存器映射

R: 只读, **W**: 只写, **R/W**: 可读写

寄存器	偏移量	R/W	描述	复位后的值
PWMA_BA = 0x4004_0000 (PWMA组) **PWMB_BA = 0x4014_0000 (PWMB组)**				
PPR	PWMA_BA+0x00	R/W	PWMA组预分频寄存器	0x0000_0000
	PWMB_BA+0x00	R/W	PWMB组 预分频寄存器	0x0000_0000
CSR	PWMA_BA+0x04	R/W	PWMA组时钟选择寄存器	0x0000_0000
	PWMB_BA+0x04	R/W	PWMB组时钟选择寄存器	0x0000_0000
PCR	PWMA_BA+0x08	R/W	PWMA组控制寄存器	0x0000_0000
	PWMB_BA+0x08	R/W	PWMB组控制寄存器	0x0000_0000
CNR0	PWMA_BA+0x0C	R/W	PWMA组计数寄存器0	0x0000_0000
	PWMB_BA+0x0C	R/W	PWMB组计数寄存器0	0x0000_0000
CMR0	PWMA_BA+0x10	R/W	PWMA组比较寄存器0	0x0000_0000
	PWMB_BA+0x10	R/W	PWMB组比较寄存器0	0x0000_0000
PDR0	PWMA_BA+0x14	R	PWMA组数据寄存器0	0x0000_0000
	PWMB_BA+0x14	R	PWMB组数据寄存器0	0x0000_0000
CNR1	PWMA_BA+0x18	R/W	PWMA组计数寄存器1	0x0000_0000
	PWMB_BA+0x18	R/W	PWMB组计数寄存器1	0x0000_0000
CMR1	PWMA_BA+0x1C	R/W	PWMA组比较寄存器1	0x0000_0000
	PWMB_BA+0x1C	R/W	PWMB组比较寄存器1	0x0000_0000
PDR1	PWMA_BA+0x20	R	PWMA组数据寄存器1	0x0000_0000
	PWMB_BA+0x20	R	PWMB组数据寄存器1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWMA组计数寄存器2	0x0000_0000
	PWMB_BA+0x24	R/W	PWMB组计数寄存器2	0x0000_0000
CMR2	PWMA_BA+0x28	R/W	PWMA组 比较寄存器2	0x0000_0000
	PWMB_BA+0x28	R/W	PWMB组 比较寄存器 2	0x0000_0000
PDR2	PWMA_BA+0x2C	R	PWMA组数据寄存器 2	0x0000_0000

寄存器	偏移量	R/W	描述	复位后的值
	PWMB_BA+0x2C	R	PWMB组数据寄存器 2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWMA组 计数寄存器 3	0x0000_0000
	PWMB_BA+0x30	R/W	PWMB组 计数寄存器3	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWMA组比较寄存器3	0x0000_0000
	PWMB_BA+0x34	R/W	PWMB组比较寄存器3	0x0000_0000
PDR3	PWMA_BA+0x38	R	PWMA组数据寄存器3	0x0000_0000
	PWMB_BA+0x38	R	PWMB组数据寄存器3	0x0000_0000
PIER	PWMA_BA+0x40	R/W	PWMA组中断使能寄存器	0x0000_0000
	PWMB_BA+0x40	R/W	PWMB组中断使能寄存器	0x0000_0000
PIIR	PWMA_BA+0x44	R/C	PWMA组中断标志寄存器	0x0000_0000
	PWMB_BA+0x44	R/C	PWMB组中断标志寄存器	0x0000_0000
CCR0	PWMA_BA+0x50	R/W	PWMA组 捕捉控制寄存器 0	0x0000_0000
	PWMB_BA+0x50	R/W	PWMB组捕捉控制寄存器0	0x0000_0000
CCR2	PWMA_BA+0x54	R/W	PWMA组捕捉控制寄存器2	0x0000_0000
	PWMB_BA+0x54	R/W	PWMB组捕捉控制寄存器2	0x0000_0000
CRLR0	PWMA_BA+0x58	R/W	PWMA组捕捉上升沿锁存寄存器(Channel 0)	0x0000_0000
	PWMB_BA+0x58	R/W	PWMB组捕捉上升沿锁存寄存器(Channel 0)	0x0000_0000
CFLR0	PWMA_BA+0x5C	R/W	PWMA组捕捉下降沿锁存寄存器(Channel 0)	0x0000_0000
	PWMB_BA+0x5C	R/W	PWMB组捕捉下降沿锁存寄存器(Channel 0)	0x0000_0000
CRLR1	PWMA_BA+0x60	R/W	PWMA组捕捉上升沿锁存寄存器(Channel 1)	0x0000_0000
	PWMB_BA+0x60	R/W	PWMB组捕捉上升沿锁存寄存器(Channel 1)	0x0000_0000
CFLR1	PWMA_BA+0x64	R/W	PWMA组捕捉下降沿锁存寄存器(Channel 1)	0x0000_0000
	PWMB_BA+0x64	R/W	PWMB组捕捉下降沿锁存寄存器(Channel 1)	0x0000_0000
CRLR2	PWMA_BA+0x68	R/W	PWMA组捕捉上升沿锁存寄存器(Channel 2)	0x0000_0000
	PWMB_BA+0x68	R/W	PWMB组捕捉上升沿锁存寄存器(Channel 2)	0x0000_0000
CFLR2	PWMA_BA+0x6C	R/W	PWMA组捕捉下降沿锁存寄存器(Channel 2)	0x0000_0000
	PWMB_BA+0x6C	R/W	PWMB组捕捉下降沿锁存寄存器(Channel 2)	0x0000_0000
CRLR3	PWMA_BA+0x70	R/W	PWMA组捕捉上升沿锁存寄存器(Channel 3)	0x0000_0000

寄存器	偏移量	R/W	描述	复位后的值
	PWMB_BA+0x70	R/W	PWMB组捕捉上升沿锁存寄存器(Channel 3)	0x0000_0000
CFLR3	PWMA_BA+0x74	R/W	PWMA组捕捉下降沿锁存寄存器(Channel 3)	0x0000_0000
	PWMB_BA+0x74	R/W	PWMB组捕捉下降沿锁存寄存器(Channel 3)	0x0000_0000
CAPENR	PWMA_BA+0x78	R/W	PWMA组捕捉输入 0~3 使能寄存器	0x0000_0000
	PWMB_BA+0x78	R/W	PWMB组捕捉输入 0~3 使能寄存器	0x0000_0000
POE	PWMA_BA+0x7C	R/W	PWMA组通道0～3输出使能	0x0000_0000
	PWMB_BA+0x7C	R/W	PWMB组通道0～3输出使能	0x0000_0000

6.6.6 PWM 控制器寄存器描述

PWM 预分频寄存器 (PPR)

寄存器	偏移量	R/W	描述	复位后的值
PPR	PWMA_BA+0x00	R/W	PWM A组预分频寄存器	0x0000_0000
	PWMB_BA+0x00	R/W	PWM B组预分频寄存器	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							

Bits	描述	
[31:24]	**DZI23**	**死区间隔设定寄存器**(PWMA组的PWM2与PWM3, PWMB组的PWM6与PWM7) 该8位寄存器决定死区长度. 每单位死区时间长度由相应的CSR位决定.
[23:16]	**DZI01**	**死区间隔设定寄存器**(PWMA组的PWM0与PWM1, PWMB组的PWM4与PWM5) 该8位寄存器决定死区长度. 每单位死区时间长度由相应的CSR位决定.
[15:8]	**CP23**	**时钟预分频器 2** (PWMA组的PWM counter 2 & 3，PWMB组的PWM counter 6 & 7) 时钟输入相应PWM计数器之前，根据(CP23 + 1)分频 **如果CP23=0, 预分频器2输出时钟停止，PWM 计数器2和3也停止**.
[7:0]	**CP01**	**时钟预分频器0**(PWMA组的PWM counter 0 & 1，PWMB组的PWM counter 4 & 5) 时钟输入相应PWM计数器之前，根据(CP01 + 1)分频 **如果CP01=0, 预分频器0输出时钟停止。PWM计数器0和1也停止**.

PWM 时钟选择寄存器(CSR)

寄存器	偏移量	R/W	描述	复位后的值
CSR	PWMA_BA+0x04	R/W	PWMA 组时钟选择寄存器	0x0000_0000
	PWMB_BA+0x04	R/W	PWMB 组时钟选择寄存器	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	CSR3			保留	CSR2		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留	CSR1			保留	CSR0		

Bits	描述	
[31:15]	保留	保留
[14:12]	**CSR3**	**定时器 3 时钟源选择**(PWM定时器3对应于A组， PWM定时器7对应于B组) 为PWM定时器选择时钟输入. CSR3 [14:12] / 输入时钟分频 100 / 1 011 / 16 010 / 8 001 / 4 000 / 2
[11]	保留	保留
[10:8]	**CSR2**	**定时器 2 时钟源选择**(PWM定时器2 对应于A组， PWM定时器6 对应于B组) 为PWM定时器选择时钟输入. (表格同CSR3)
[7]	保留	保留

Bits	描述	
[6:4]	**CSR1**	**定时器 1 时钟源选择**(PWM定时器1 对应于A组， PWM定时器5 对应于B组) 为PWM定时器选择时钟输入. (表格同CSR3)
[3]	**保留**	保留
[2:0]	**CSR0**	**定时器 0 时钟源选择**(PWM定时器0 对应于A组， PWM定时器4 对应于B组) 为PWM定时器选择时钟输入. (表格同CSR3)

PWM 控制寄存器(PCR)

寄存器	偏移量	R/W	描述	复位后的值
PCR	PWMA_BA+0x08	R/W	PWM A组控制寄存器 (PCR)	0x0000_0000
	PWMB_BA+0x08	R/W	PWM B组控制寄存器(PCR)	0x0000_0000

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

Bits	描述	
[31:28]	保留	保留
[27]	**CH3MOD**	**PWM-定时器3 自动重载/单触发模式选择**(PWM定时器3 对应于A组 ， PWM定时器7 对应于B组) 1 = 自动重载模式 0 = 单触发模式 **注:** 如果该位由0置1，会使CNR3 和CMR3 清位
[26]	**CH3INV**	**PWM定时器3反向打开/关闭**(PWM定时器3 对应于A组 ， PWM定时器7 对应于B组) 1 = 反向打开 0 = 反向关闭
[25]	保留	保留
[24]	**CH3EN**	**PWM定时器3 使能/禁用**(PWM定时器3 对应于A组， PWM定时器7 对应于B组) 1 = 使能相应PWM定时器开始运行 0 = 停止相应PWM定时器运行
[23:20]	保留	保留

Bits	描述	
[19]	**CH2MOD**	**PWM-定时器2 自动重载/单触发模式选择** (PWM定时器2 对应于A组 ， PWM定时器6 对应于B组) 1 = 自动重载模式 0 = 单触发模式 **注:** 如果该位由0置1，会使CNR2和CMR2清位.
[18]	**CH2INV**	**PWM-定时器2反向打开/关闭** (PWM定时器2 对应于A组 ， PWM定时器6 对应于B组) 1 = 反向打开 0 = 反向关闭
[17]	保留	保留
[16]	**CH2EN**	**PWM定时器2 使能/禁用** (PWM定时器2 对应于A组， PWM定时器6 对应于B组) 1 = 使能相应PWM定时器开始运行 0 = 停止相应PWM定时器运行
[15:12]	保留	保留
[11]	**CH1MOD**	**PWM定时器1 自动重载/单触发模式选择** (PWM定时器1 对应于A组， PWM定时器5 对应于B组) 1 = 自动重载模式 0 = 单触发模式 **注:** 如果该位由0置1，会使CNR1和CMR1清位.
[10]	**CH1INV**	**PWM-定时器1反向打开/关闭** (PWM定时器1 对应于A组 ， PWM定时器5 对应于B组) 1 = 反向打开 0 = 反向关闭
[9]	保留	保留
[8]	**CH1EN**	**PWM定时器1 使能/禁用** (PWM定时器1 对应于A组， PWM定时器5 对应于B组) 1 = 使能相应PWM定时器开始运行 0 = 停止相应PWM定时器运行
[7:6]	保留	保留
[5]	**DZEN23**	**死区发生器2使能/禁用**(PWM2 and PWM3 pair 对应于 PWMA组, PWM6 and PWM7 pair 对应于 PWMB组) 1 = 使能 0 = 禁用 **注**: 当死区发生器使能，PWM A组的PWM2与PWM3将成为互补对, PWM B组的

Bits	描述	
		PWM6与PWM7将成为互补对.
[4]	**DZEN01**	**死区发生器0使能/禁用**(PWM0 and PWM1 pair 对应于 PWMA组, PWM4 and PWM5 pair 对应于 PWMB组) 1 = 使能 0 = 禁用 **注**: 当死区发生器使能，PWM A组的PWM0与PWM1将成为互补对, PWM B组的PWM4与PWM5将成为互补对.
[3]	**CH0MOD**	**PWM-定时器 0 自动加载/单触发模式选择** (PWM定时器0 对应于A组 and PWM定时器4 对应于B组) 1 = 自动重载模式 0 = 单触发模式 **注:** 如果该位由0置1，会使CNR0和CMR0清位.
[2]	**CH0INV**	**PWM-定时器0反向打开/关闭** (PWM定时器0 对应于A组 and PWM定时器4 对应于B组) 1 = 反向打开 0 = 反向关闭
[1]	**保留**	保留
[0]	**CH0EN**	**PWM-定时器0 使能/禁用** (PWM定时器0 对应于A组 and PWM定时器4 对应于B组) 1 = 使能相应PWM定时器开始运行 0 = 停止相应PWM定时器运行

PWM 计数器寄存器3-0 (CNR3-0)

寄存器	偏移量	R/W	描述	复位后的值
CNR0	PWMA_BA+0x0C	R/W	PWM A组计数器寄存器0	0x0000_0000
	PWMB_BA+0x0C	R/W	PWM B组计数器寄存器0	0x0000_0000
CNR1	PWMA_BA+0x18	R/W	PWM A组计数器寄存器1	0x0000_0000
	PWMB_BA+0x18	R/W	PWM B组计数器寄存器1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWM A组计数器寄存器2	0x0000_0000
	PWMB_BA+0x24	R/W	PWM B组计数器寄存器2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWM A组计数器寄存器3	0x0000_0000
	PWMB_BA+0x30	R/W	PWM B组计数器寄存器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**
CNRx [15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNRx [7:0]							

Bits	描述	
[31:16]	保留	保留
[15:0]	**CNRx**	**PWM 计数器/定时器载入值** CNR 决定PWM的周期. ● PWM 频率 = PWMxy_CLK/ ((prescale+1) * (clock divider) * (CNR+1)); xy代表01, 23, 45或 67, 取决于所选择的PWM通道. ● 占空比 = (CMR+1)/(CNR+1). ● CMR >= CNR: PWM输出高. ● CMR < CNR: PWM低脉冲宽度= (CNR-CMR) unit; PWM高脉冲宽度=

Bits	描述	
		(CMR+1) unit. ● CMR = 0: PWM低脉冲宽度= (CNR) unit; PWM高脉冲宽度= 1 unit (Unit = 一个 PWM 时钟周期) 注: CNR写入数据后将在下一个PWM周期生效.

PWM 比较器寄存器 3-0 (CMR3-0)

寄存器	偏移量	R/W	描述	复位后的值
CMR0	PWMA_BA+0x10	R/W	PWM A组比较器寄存器0	0x0000_0000
	PWMB_BA+0x10	R/W	PWM B组比较器寄存器0	0x0000_0000
CMR1	PWMA_BA+0x1C	R/W	PWM A组比较器寄存器1	0x0000_0000
	PWMB_BA+0x1C	R/W	PWM B组比较器寄存器1	0x0000_0000
CMR2	PWMA_BA+0x28	R/W	PWM A组比较器寄存器2	0x0000_0000
	PWMB_BA+0x28	R/W	PWM B组比较器寄存器2	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWM A组比较器寄存器3	0x0000_0000
	PWMB_BA+0x34	R/W	PWM B组比较器寄存器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**
CMRx [15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMRx [7:0]							

Bits	描述	
[31:16]	保留	保留
[15:0]	**CMRx**	**PWM 比较器寄存器** CMR 决定 PWM 的占空比. ● PWM 频率 = PWMxy_CLK/(prescale+1)*(clock divider)/(CNR+1); xy代表 01, 23, 45 or 67, 取决于所选择的PWM通道. ● 占空比 = (CMR+1)/(CNR+1). ● CMR >= CNR: PWM 输出高. ● CMR < CNR: PWM低脉冲宽度= (CNR-CMR) unit; PWM高脉冲宽度 = (CMR+1) unit.

Bits	描述	
		● CMR = 0: PWM低脉冲宽度= (CNR) unit; PWM 高脉冲宽度 = 1 unit (Unit =一个PWM时钟周期) Note: 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	偏移量	R/W	描述	复位后的值
PDR0	PWMA_BA0+0x14	R	PWM A组数据寄存器0	0x0000_0000
	PWMB_BA0+0x14	R	PWMB组数据寄存器0	0x0000_0000
PDR1	PWMA_BA0+0x20	R	PWMA组数据寄存器1	0x0000_0000
	PWMB_BA0+0x20	R	PWMB组数据寄存器1	0x0000_0000
PDR2	PWMA_BA0+0x2C	R	PWMA组数据寄存器2	0x0000_0000
	PWMB_BA0+0x2C	R	PWMB组数据寄存器2	0x0000_0000
PDR3	PWMA_BA0+0x38	R	PWMA组数据寄存器3	0x0000_0000
	PWMB_BA0+0x38	R	PWMB组数据寄存器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**
PDR[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PDR[7:0]							

Bits	描述	
[31:16]	保留	保留
[15:0]	**PDRx**	**PWM 数据寄存器** 用户可以查询PDR得到16位计数器的当前值.

PWM 中断使能寄存器 (PIER)

寄存器	偏移量	R/W	描述	复位后的值
PIER	PWMA_BA+0x40	R/W	PWM A组中断使能寄存器	0x0000_0000
	PWMB_BA+0x40	R/W	PWM B组中断使能寄存器	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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留				**PIER3**	**PIER2**	**PIER1**	**PIER0**

Bits	描述	
[31:4]	保留	保留
[3]	**PWMIE3**	**PWM 通道3中断使能** 1 = 使能 0 = 禁用
[2]	**PWMIE2**	**PWM 通道2中断使能** 1 = 使能 0 = 禁用
[1]	**PWMIE1**	**PWM 通道1中断使能** 1 = 使能 0 = 禁用
[0]	**PWMIE0**	**PWM 通道0中断使能** 1 = 使能 0 = 禁用

PWM 中断标志寄存器(PIIR)

寄存器	偏移量	R/W	描述	复位后的值
PIIR	PWMA_BA+0x44	R/W	PWM A组中断标志寄存器	0x0000_0000
	PWMB_BA+0x44	R/W	PWM B组中断标志寄存器	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				**PWMIF3**	**PWMIF2**	**PWMIF1**	**PWMIF0**

Bits	描述	
[31:4]	保留	保留
[3]	**PWMIF3**	**PWM 通道3中断状态** 如果PWM3中断使能位(PWMIE3)为"1"时，当PWM3向下计数至0时，硬件将该位置1。软件写1清该位.
[2]	**PWMIF2**	**PWM 通道 2 中断状态** 如果PWM2中断使能位(PWMIE2)为"1"时，当PWM2向下计数至0时，硬件将该位置1。软件写1清该位.
[1]	**PWMIF1**	**PWM 通道 1 中断状态** 如果PWM1中断使能位(PWMIE1为"1"时，当PWM1向下计数至0时，硬件将该位置1。软件写1清该位.
[0]	**PWMIF0**	**PWM 通道 0中断状态** 如果PWM0中断使能位(PWMIE0)为"1"时，当PWM0向下计数至0时，硬件将该位置1。软件写1清该位

注: 用户可通过对PIIR相应的位写1来对中断标志清零.

捕捉控制寄存器(CCR0)

寄存器	偏移量	R/W	描述	复位后的值
CCR0	PWMA_BA+0x50	R/W	PWM A组捕捉控制寄存器	0x0000_0000
	PWMB_BA+0x50	R/W	PWM B组捕捉控制寄存器	0x0000_0000

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

Bits	描述	
[31:24]	保留	保留
[23]	**CFLRI1**	**CFLR1锁存指示位** 在PWM输入通道1的下降沿，CFLR1 锁存PWM向下计数器，该位由硬件置位。写1清该位.
[22]	**CRLRI1**	**CRLR1锁存指示位** 在PWM输入通道1的上升沿，CFLR1 锁存PWM向下计数器，并且该位由硬件置位。写1清该位.
[5]	保留	保留
[20]	**CAPIF1**	**捕捉器1中断标志** 如果PWM组通道1上升沿锁存中断使能(CRL_IE1=1), PWM组通道1的上升沿转变将使CAPIF1为高; 同样, 如果下降沿锁存中断使能(CFL_IE1=1)，下降沿转变将使CAPIF1 为高. 该标志由软件写1清零.
[19]	**CAPCH1EN**	**捕捉器通道1使能/禁用** 1 = 使能PWM组通道1的捕捉功能. 0 = 禁用PWM组通道1的捕捉功能 使能时，捕捉锁存PWM计数器并保存CRLR（上升沿锁定）和CFLR（上升沿锁

Bits	描述	
		定）. 禁用时，捕捉器不更新CRLR和CFLR，并禁用PWM组通道1中断.
[18]	**CFL_IE1**	**PWM 组通道1下降沿锁存中断使能** 1 = 使能向下锁定中断 0 = 禁用向下锁定中断 使能时，如果捕捉器检测到PWM组通道1有下降沿，捕捉器产生中断.
[17]	**CRL_IE1**	**PWM 组通道1上升沿锁存中断使能** 1 = 使能向上锁定中断 0 = 禁用向上锁定中断 使能时，如果捕捉器检测到PWM组通道1有上升沿，捕捉器产生中断.
[16]	**INV1**	**通道1反向打开/关闭** 1 = 反向打开。输入到寄存器的信号与通道上的实际信号电平反向。 0 = 反向关闭
[15:8]	保留	保留
[7]	**CFLRI0**	**CFLR0锁存标志位** 在PWM输入通道0的下降沿，CFLR0 锁存PWM向下计数器，该位由硬件置位。 写1清零
[6]	**CRLRI0**	**CRLR0锁存标志位** 在PWM输入通道0的上升沿，CRLR0 锁存PWM向下计数器，该位由硬件置位。 写1清零.
[5]	保留	保留
[4]	**CAPIF0**	**捕捉器0中断标志** 如果PWM组通道1上升沿锁存中断使能(CRL_IE0=1), PWM组通道1的上升沿将使CAPIF0为高; 同样, 如果使能下降沿锁存中断使能(CRL_IE0=1)，下降沿将使CAPIF0为高. 该标志由软件写1清零.
[3]	**CAPCH0EN**	**捕捉器通道0使能/禁用** 1 = 使能PWM组通道0的捕捉功能. 0 = 禁用PWM组通道0的捕捉功能 使能时，捕捉锁定PWM计数器并保存CRLR（向上锁定）和CFLR（向下锁定）. 禁用时，捕捉器不更新CRLR和CFLR，并禁用PWM组通道0中断.
[2]	**CFL_IE0**	**通道0下降沿锁存中断使能**

Bits	描述	
		1 = 使能下降沿锁存中断 0 = 禁用下降沿锁存中断 使能时，捕捉器检测到PWM组通道0有下降沿转变，捕捉器产生中断.
[1]	**CRL_IE0**	**PWM 组通道0上升沿锁存中断使能** 1 = 使能上升沿锁存中断 0 = 禁用上升沿锁存中断 使能时，如果捕捉器检测到PWM组通道0有上升沿转变，捕捉器产生中断.
[0]	**INV0**	**通道0反向打开/关闭** 1 = 反向打开。输入到寄存器的信号与通道上的实际信号电平反向。 0 = 反向关闭

捕捉控制寄存器(CCR2)

寄存器	偏移量	R/W	描述	复位后的值
CCR2	PWMA_BA+0x54	R/W	PWM A组捕捉控制寄存器	0x0000_0000
	PWMB_BA+0x54	R/W	PWM B组捕捉控制寄存器	0x0000_0000

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

Bits	描述	
[31:24]	保留	保留
[23]	**CFLRI3**	**CFLR3锁存标志位** 当PWM输入通道3有下降沿转变时，CFLR3锁存PWM向下计数器的值，该位由硬件置位.写1清0.
[22]	**CRLRI3**	**CRLR3锁存标志位** 当PWM输入通道3有下降沿转变时，CFLR3锁寸PWM向下计数器的值，该位由硬件置位.写1清0.
[21]	保留	保留
[20]	**CAPIF3**	**捕捉器1中断标志** 如果PWM组通道3上升沿锁存中断使能(CRL_IE3=1), PWM组通道3的上升沿转变将使CAPIF3为高; 同样, 如果下降沿锁存使能(CFL_IE3=1)，下降沿转变将使CAPIF3 为高. 该标志由软件写1清零.
[19]	**CAPCH3EN**	**捕捉器通道3使能/禁用** 1 = 使能PWM组通道3的捕捉功能. 0 = 禁用PWM组通道3的捕捉功能 使能时，捕捉器锁存PWM计数器并保存到CRLR（上升沿锁定）和CFLR（下降沿

Bits	描述	
		锁定）. 禁用时，捕捉器不更新CRLR和CFLR，并禁用PWM组通道3中断.
[18]	**CFL_IE3**	**PWM 组通道3下降沿锁存中断使能** 1 = 使能下降沿锁存中断 0 = 禁用下降沿锁存中断 使能时，捕捉器检测到PWM组通道3有下降沿转变，捕捉器产生中断.
[17]	**CRL_IE3**	**PWM 组通道3上升沿锁存中断使能** 1 = 使能上升沿锁存中断 0 = 禁用上升沿锁存中断 使能时，如果捕捉器检测到PWM组通道3有上升沿转变，捕捉器产生中断.
[16]	**INV3**	**通道3反向打开/关闭** 1 = 反向打开。输入到寄存器的信号与通道上的实际信号电平反向。 0 = 反向关闭
[15:8]	**保留**	保留
[7]	**CFLRI2**	**CFLR2锁存标志位** 当PWM输入通道2有下降沿转变时，CFLR2锁存PWM向下计数器的值，该位由硬件置位. 写1清0.
[6]	**CRLRI2**	**CRLR2锁存标志位** 当PWM输入通道2有下降沿转变时，CRLR2锁存PWM向下计数器的值，该位由硬件置位. 写1清0.
[5]	**保留**	保留
[4]	**CAPIF2**	**捕捉器2中断指示标志** 如果PWM组通道2上升沿锁定中断使能(CRL_IE2=1), PWM组通道2的上升沿将使CAPIF2为高; 同样, 如果使能(CFL_IE2=1)，下降沿将使CAPIF2 为高. 该标志由软件写1清零.
[3]	**CAPCH2EN**	**捕捉器通道2使能/禁用** 1 = 使能PWM组通道2的捕捉功能. 0 = 禁用PWM组通道2的捕捉功能 使能时，捕捉器锁存PWM计数器并保存到CRLR（上升沿锁定）和CFLR（下降沿）. 禁用时，捕捉器不更新CRLR和CFLR，并禁用PWM组通道2中断.

Bits	描述	
[2]	**CFL_IE2**	**PWM 组通道2下降沿锁存中断使能** 1 = 使能下降沿锁存中断 0 = 禁用下降沿锁存中断 使能时，捕捉器检测到PWM组通道2有下降沿转变时，捕捉器产生中断.
[1]	**CRL_IE2**	**PWM 组通道2上升沿锁存中断使能** 1 = 使能上升沿锁定中断 0 = 禁用上升沿锁定中断 使能时，如果捕捉器检测到PWM组通道2有上升沿转变时，捕捉器产生中断.
[0]	**INV2**	**通道2反向打开/关闭** 1 = 反向打开。输入到寄存器的信号与通道上的实际信号电平反向。 0 = 反向关闭

捕捉器上升沿锁存寄存器3-0 (CRLR3-0)

寄存器	偏移量	R/W	描述	复位后的值
CRLR0	PWMA_BA+0x58	R	PWM A组捕捉器上升沿锁存寄存器(channel 0)	0x0000_0000
	PWMB_BA+0x58	R	PWM B组捕捉器上升沿锁存寄存器(channel 0)	0x0000_0000
CRLR1	PWMA_BA+0x60	R	PWM A组捕捉器上升沿锁存寄存器(channel 1)	0x0000_0000
	PWMB_BA+0x60	R	PWM B组捕捉器上升沿锁存寄存器(channel 1)	0x0000_0000
CRLR2	PWMA_BA+0x68	R	PWM A组捕捉器上升沿锁存寄存器(channel 2)	0x0000_0000
	PWMB_BA+0x68	R	PWM B组捕捉器上升沿锁存寄存器(channel 2)	0x0000_0000
CRLR3	PWMA_BA+0x70	R	PWM A组捕捉器上升沿锁存寄存器(channel 3)	0x0000_0000
	PWMB_BA+0x70	R	PWM B组捕捉器上升沿锁存寄存器(channel 3)	0x0000_0000

注: 当CPU时钟低于PWM/Capture时钟时，写CRLRx不保证正确.

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
CRLRx [15:8]							
7	6	5	4	3	2	1	0
CRLRx [7:0]							

Bits	描述	
[31:16]	保留	保留
[15:0]	CRLRx	捕捉上升沿锁存寄存器 通道0/1/2/3 有上升沿时转变时，锁存PWM计数器的值到这个寄存器.

捕捉下降沿锁存寄存器3-0 (CFLR3-0)

寄存器	偏移量	R/W	描述	复位后的值
CFLR0	PWMA_BA+0x5C	R	PWM A组捕捉器下降沿锁存寄存器(channel 0)	0x0000_0000
	PWMB_BA+0x5C	R	PWM B组捕捉器下降沿锁存寄存器(channel 0)	0x0000_0000
CFLR1	PWMA_BA+0x64	R	PWM A组捕捉器下降沿锁存寄存器(channel 1)	0x0000_0000
	PWMB_BA+0x64	R	PWM B组捕捉器下降沿锁存寄存器(channel 1)	0x0000_0000
CFLR2	PWMA_BA+0x6C	R	PWM A组捕捉器下降沿锁存寄存器(channel 2)	0x0000_0000
	PWMB_BA+0x6C	R	PWM B组捕捉器下降沿锁存寄存器(channel 2)	0x0000_0000
CFLR3	PWMA_BA+0x74	R	PWM A组捕捉器下降沿锁存寄存器(channel 3)	0x0000_0000
	PWMB_BA+0x74	R	PWM B组捕捉器下降沿锁存寄存器(channel 3)	0x0000_0000

注: 当CPU时钟低于PWM/Capture时钟时，不保证对 CFLRx 的写正确.

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
CFLRx [15:8]							
7	6	5	4	3	2	1	0
CFLRx [7:0]							

Bits	描述	
[31:16]	保留	保留
[15:0]	CFLRx	捕捉器下降沿锁存寄存器 通道0/1/2/3 有下降沿转变时，锁存PWM计数器的值到这个寄存器.

捕捉输入使能寄存器(CAPENR)

寄存器	偏移量	R/W	描述	复位后的值
CAPENR	PWMA_BA+0x78	R/W	PWM A组捕捉器输入0~3 使能寄存器	0x0000_0000
	PWMB_BA+0x78	R/W	PWM B组捕捉器输入0~3 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							
7	6	5	4	3	2	1	0
保留				CAPENR			

Bits	描述	
[3:0]	CAPENR	捕捉器输入使能寄存器 有4组捕捉输入。Bit0~Bit3 用于控制每个输入的打开 / 关闭. 0 =关闭(PWMx 复用脚输入对捕捉器不产生影响) 1 = 打开 (PWMx 复用脚将影响捕捉器功能.) CAPENR Bit 3210 用于PWM A组 Bit xxx1 ➔ 捕捉通道0 从 P2 [0] 输入 Bit xx1x ➔ 捕捉通道1 从 P2 [1] 输入 Bit x1xx ➔ 捕捉通道2 从 P2 [2] 输入 Bit 1xxx ➔ 捕捉通道3 从 P2 [3] 输入 Bit 3210用于PWM B组 Bit xxx1 ➔ 捕捉通道0 从 P2 [4] 输入 Bit xx1x ➔ 捕捉通道1 从 P2 [5] 输入 Bit x1xx ➔ 捕捉通道2 从 P2 [6] 输入 Bit 1xxx ➔ 捕捉通道3 从 P2 [7] 输入

PWM输出使能寄存器 (POE)

寄存器	偏移量	R/W	描述	复位后的值
POE	PWMA_BA+0x7C	R/W	PWM A组输出使能寄存器（通道0～3）	0x0000_0000
	PWMB_BA+0x7C	R/W	PWM B组输出使能寄存器（通道0～3）	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**
保留				**PWM3**	**PWM2**	**PWM1**	**PWM0**

Bits	描述	
[3]	**PWM3**	**PWM 通道3输出使能寄存器** 1 = 使能PWM通道3输出 0 = 禁用PWM通道3输出 注: GPIO相应管脚必须切换到PWM功能
[2]	**PWM2**	**PWM 通道2输出使能寄存器** 1 = 使能PWM通道2输出 0 = 禁用PWM通道2输出 注: GPIO相应管脚必须切换到PWM功能
[1]	**PWM1**	**PWM 通道1输出使能寄存器** 1 = 使能PWM通道1输出 0 = 禁用PWM通道1输出 注: GPIO相应管脚必须切换到PWM功能
[0]	**PWM0**	**PWM 通道0输出使能寄存器** 1 = 使能PWM通道0输出

Bits	描述	
		0 = 禁用PWM通道0输出 注: GPIO相应管脚必须切换到PWM功能

6.7 串行外设接口(SPI)控制器

6.7.1 概述

串行外围设备接口(SPI)是一个工作于全双工模式下的同步串行数据通讯协议。设备通过4线双向接口工作于主机/从机模式进行通讯。NuMicro M051™系列包括最多2组SPI控制器，将从外设接收到的数据进行串并转换, 或将要发送到外设的数据进行并串转换。每组SPI控制器都可被设置成主机；也可设置为被片外主机设备控制的从机。为了某些特别的应用，控制器支持可变串行时钟。

6.7.2 特性

- 最多两组SPI控制器
- 支持主/从机模式
- 比特长度可配置，一个传输字最多可达32比特；传输次数可配置，一次最多可传输2笔，所以一次数据传输的最大比特长度是64比特
- 支持burst操作模式，在一次传输过程中，发送/接收最多一次可以传输两个字
- 支持MSB 或 LSB 优先传输
- 字节或字休眠模式
- 主机模式下支持两种可编程的串行时钟频率
- 从机模式下支持3线模式，没有从设备片选
- SPI时钟频率可以配置等于系统时钟频率

6.7.3 SPI 框图

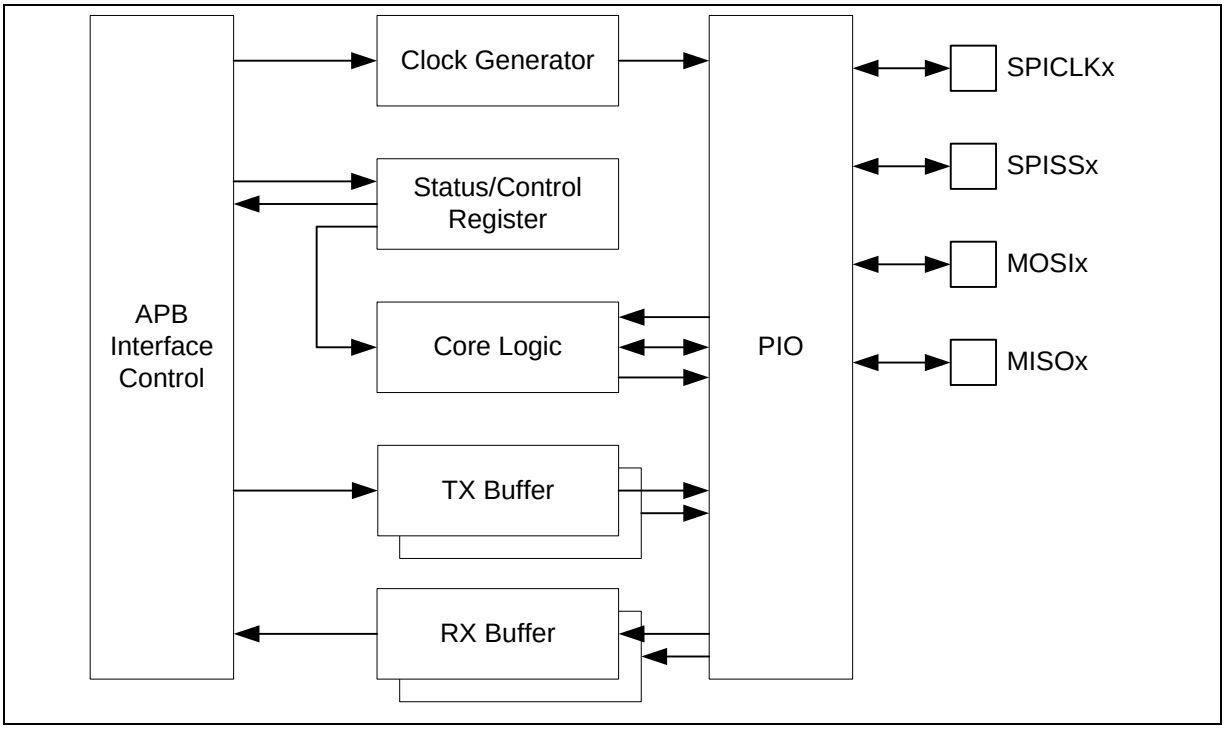


图 6.7.3-1SPI 框图

6.7.4 SPI 功能描述

主机/从机模式

SPI控制器可通过设置**SLAVE** 位(SPI_CNTRL[18])被配置为主机或从机模式，来与片外SPI从机或主机设备通讯。主机模式与从机模式下的应用框图如下所示。

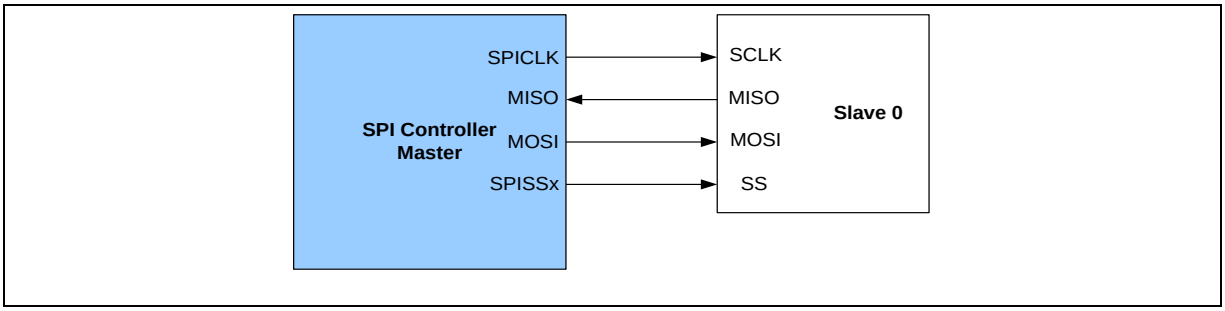


图6.7.4-1 SPI主机模式应用框图

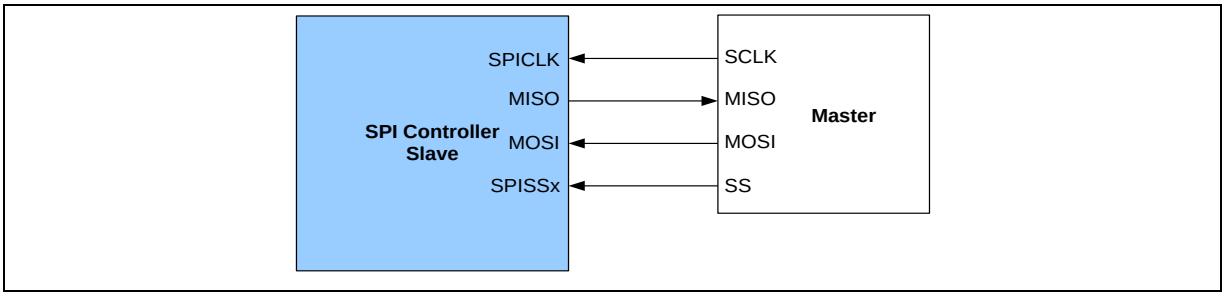


图6.7.4-2 SPI从机模式应用框图

从机选择

在主机模式下，SPI控制器能通过从机片选输出脚SPISS0或者SPISS1驱动一个片外从机设备。从机模式下，片外的主机设备驱动从机片选信号通过SPISS0或者SPISS1输入到SPI控制器。在主机/从机模式下，从机选择信号的有效电平可以在**SS_LVL**位 (SPI_SSR[2])被编程为低有效或高有效，**SS_LTRIG** 位 (SPI_SSR[4])配置从机选择信号SPISS 为电平触发或边沿触发。触发条件的选择取决于所连接的外围从机/主机的设备类型.

从模式下，如果SS_LTRIG bit被配置为电平触发，LTRIG_FLAG比特(SPI_SSR[5])可以用来指示一次传输完成时，是否接收次数和接收到的bit数满足TX_NUM 和 TX_BIT_LEN的定义（一次传输完成意味着从设备片选已经处于非有效状态或者）或者SPI 控制器已经完成一次传输）

电平触发/边沿触发

在从机模式下，从机选择信号可以配置成电平触发或边沿触发。边沿触发，数据传输从有效边沿开始，到出现一个无效边沿结束。如果主机不发送无效边沿信号给从机，传输将不能完成，从机的中断标志将不会被置位。电平触发，下面两个情况可以终止传输过程，并使从机的中断标志被置位。 一，如果主机设置从机片选管脚为非有效电平，将迫使从机终止当前传输而不管已经传输多少位，并且中断标志将被置位。用户可以读取LTRIG_FLAG位的状态来判断数据是否传输完毕。二，如果传输位数与TX_NUM 和 TX_BIT_LEN 的设置匹配时，从机的中断标志将被置位。

自动从机选择

在主机模式下，如果AUTOSS (SPI_SSR[3])置位，从机片选信号自动产生，并根据**SSR[0]** (SPI_SSR[0])是否使能，输出到SPISS0/1引脚上，这意味着，从机选择信号（由SSR[0]寄存器使能）由SPI控制器在发送/接收开始（通过置位GO_BUSY位(SPI_CNTRL[0])实现）时置为有效，在传输结束时置为无效。当AUTOSS位清零时，可以手动置位与清零寄存器SPI_SSR[0]的相关位，来发出或取消从机选择输出信号。从机选择输出信号的有效电平在SS_LVL 位 (SPI_SSR[2])指定。

串行时钟

在主机模式下，配置DIVIDER器寄存器(SPI_DIVIDER [15:0])来编程由SPICLK引脚输出的串行时钟频率。如果VARCLK_EN bit (SPI_CNTRL[23])使能，串行时钟也支持可变频率功能，在这种情况下，串行时钟每一位的输出频率可被编程为一种频率或两种不同的频率，这取决于DIVIDER和DIVIDER2(SPI_DIVIDER[31:16])的设置。每一位的可变频率是由VARCLK(SPI_VARCLK[3:0])寄存器定义的。

在从机模式下，片外主机设备通过此SPI控制器的SPICLK输入口驱动串行时钟输入.

可变串行时钟频率

在主机模式下，如果可变时钟使能位VARCLK_EN (SPI_CNTRL [23])使能，串行时钟的输出可被编程为可变频率模式。频率格式在寄存器VARCLK (SPI_VARCLK [31:0])里定义。如果VARCLK 的某位为‘0’，输出频率取决于DIVIDER (SPI_DIVIDER[15:0])，如果VARCLK某位为‘1’, 输出频率取决于DIVIDER2 (SPI_DIVIDER[31:16])。图6.7.4-3为串行时钟(SPICLK), VARCLK, DIVIDER和DIVIDER2之间的时序关系。VARCLK中两位联合确定一个时钟周期。位域VARCLK [31:30]确定SPICLK的第一个时钟周期，位域VARCLK [29:28]确定SPICLK的第二个时钟周期，以此类推。时钟源的选择在VARCLK中定义，且必须在下一个时钟选择前1个周期被置位。例如，如果在SPICLK中有5个CLK1，VARCLK将在

MSB设置9个‘0’，第10个将设置为‘1’，以切换到下一个时钟源CLK2。注意当使能VARCLK_EN 位，TX_BIT_LEN 必须设置成0x10 (仅16 bits 模式).

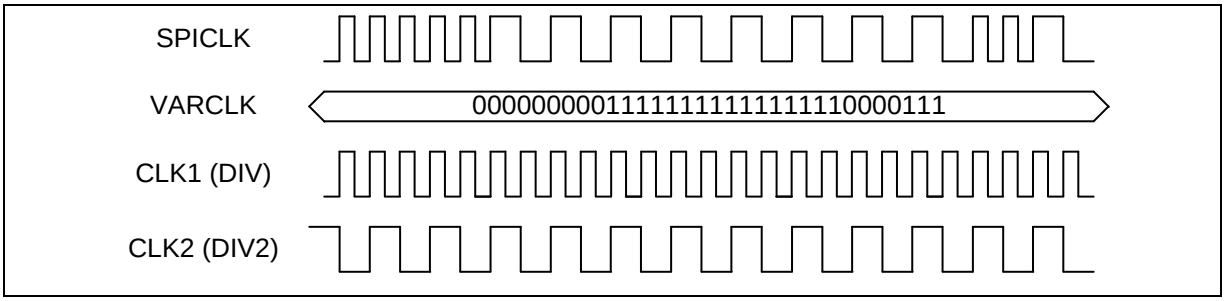


图 6.7.4-3 可调串行时钟频率

时钟极性

在主机模式下，CLKP 位 (SPI_CNTRL[11])定义串行时钟的空闲状态。如果**CLKP** = 1，输出SPICLK在空闲状态下为高电平。**CLKP** = 0时，输出SPICLK在空闲状态下为低电平。

发送/接收位长度

传输字的长度在**Tx_BIT_LEN** 位域(SPI_CNTRL[7:3])中配置。对于发送和接收，一个传输字的比特长度可被配置为最多32位。

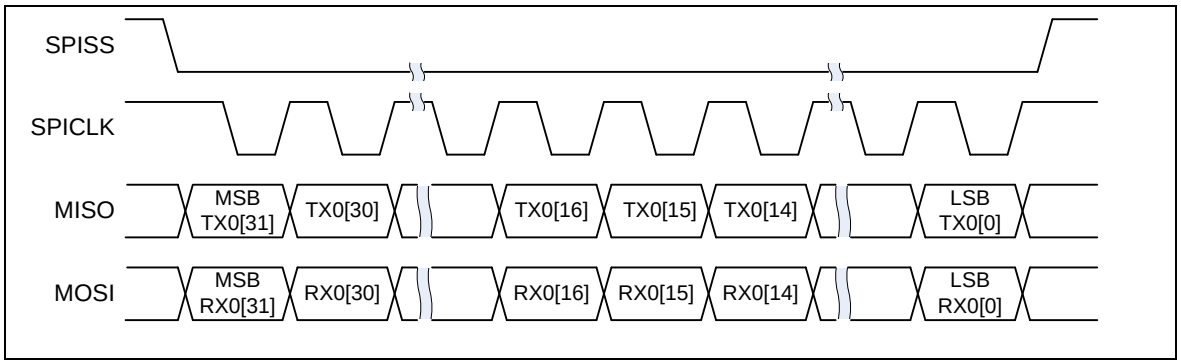


图 6.7.4-4 一次传输32-Bit

Burst 模式

SPI可通过设置TX_NUM (SPI_CNTRL [9:8])为0X01，切换到burst模式. burst 模式下, SPI 可以在一次传输中进行两笔发送/接收处理。 SPI burst 模式波形图如下:

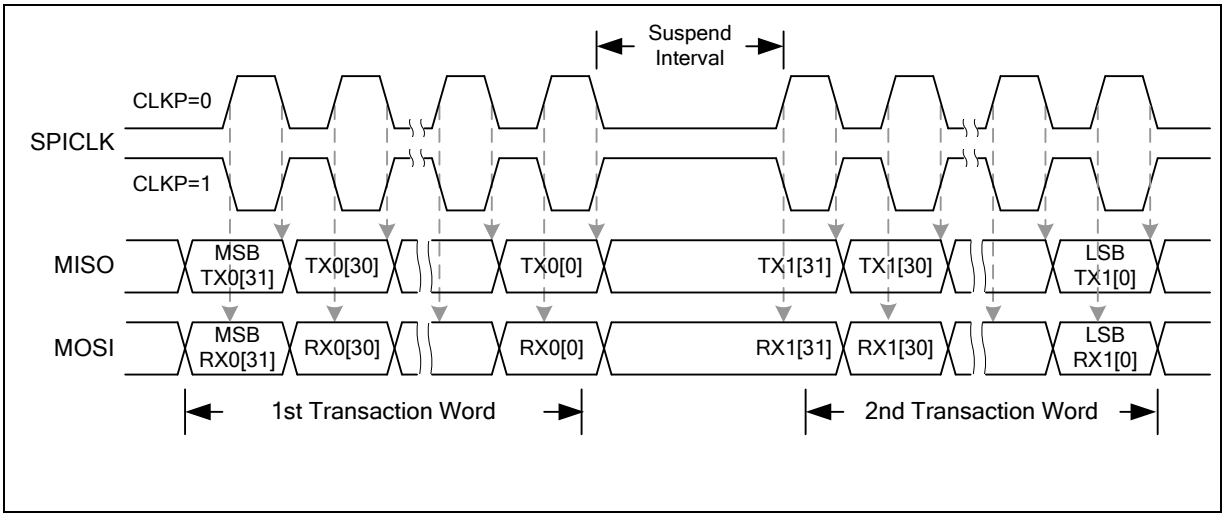


图6.7.4-5 一次传输两笔 (Burst Mode)

LSB 优先

LSB 位(SPI_CNTRL[10]) 定义是从LSB还是从MSB开始发送/接收数据.

发送边沿

Tx_NEG 位 (SPI_CNTRL[2]) 定义数据发送是在串行时钟SPICLK的下降沿还是上升沿.

接收边沿

Rx_NEG 位 (SPI_CNTRL[1]) 定义数据接收是在串行时钟SPICLK的下降沿还是上升沿.

字休眠

在主机模式下，SP_CYCLE (SPI_CNTRL[15:12])的4位负责配置在两个连续传输字之间的休眠间隔，有效值为2～17个串行时钟周期。休眠间隔指从前一次传输字的最后一个时钟下降沿到下一次传输字的第一个时钟上升沿（**CLKP** = 0）；如果**CLKP** = 1，间隔为前一次传输字的上升沿到下一次传输字的下降沿。 SP_CYCLE的默认值为0x0 (2 个串行时钟周期)，如果**Tx_NUM** = 0x00，设置这些位对数据传输过程没有任何影响。

字节重排序

当传输被设置为MSB优先(LSB = 0)，并且REORDER被使能时，TX_BIT_LEN = 32位模式下，存储在TX缓存与RX缓存中的数据将按[BYTE0, BYTE1, BYTE2, BYTE3]的次序重新排列，发送/接收数据将变成BYTE0, BYTE1, BYTE2, BYTE3的顺序。如果**Tx_BIT_LEN** 被设置为24-位模式，TX 缓存与RX 缓存的数据将被重新排列为[unknown byte, BYTE0, BYTE1, BYTE2]， BYTE0，BYTE1和BYTE2将按MSB优先的方式一步一步的被发送/接收。16位模式下规则与上述相同。

其实字节重排序功能，就是数据大端(big-endian)和小端(little-endian)之间的转换

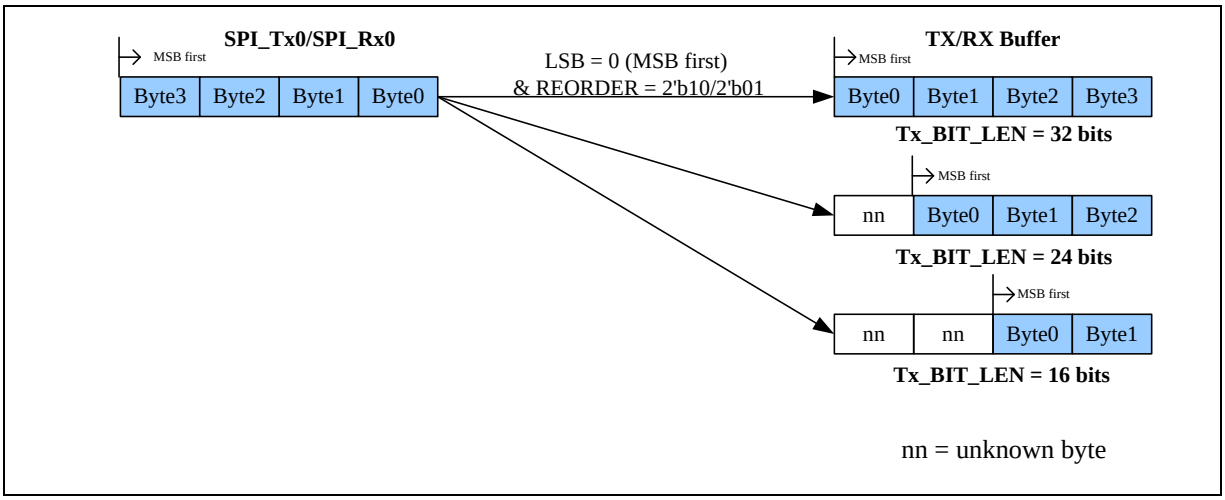


图 6.7.4-6 字节重排列

字节休眠

主机模式下，如果SPI_CNTRL[19]被设置为1，硬件将在一个传输字的两个连续传输字节之间插入2～17个串行时钟周期的休眠间隔。字节休眠的设定与字休眠设定一样，二者使用共同的位域SP_CYCLE， 注意当使能字节休眠功能时，TX_BIT_LEN必须被设置为0x00 (一个传输字32位)。

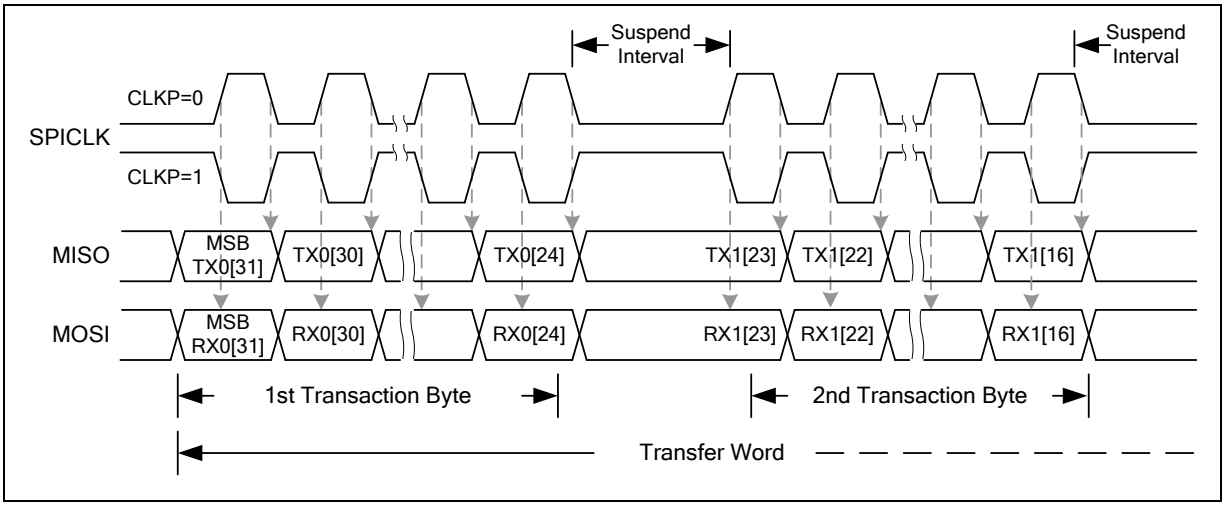


图6.7.4-7 字节休眠时序波形

REORDER	描述
00	禁用字节重排序功能和字节休眠.
01	使能字节重排序功能，并在每个字节之间插入一个字节休眠间隔（2~17串行时钟周期）. TX_BIT_LEN 的设置必须配置成0x00 (32 bits/ word)
10	使能字节重排序功能但禁用字节休眠功能.
11	禁用字节重排序功能，但在每个字节之间插入一个休眠间隔（2~17串行时钟周期）. TX_BIT_LEN的设置必须配置成 0x00 (32 bits/ word)

表6.7-1 字节顺序和字节休眠条件

无从设备片选模式 (3-线模式)

从模式下，可以忽略从设备片选信号。SPI控制器可以工作在无从设备片选信号模式 (3-线模式) ，当MCU工作在从设备模式下时，接口可以只包含 SPICLK, SPI_MISO, 和 SPI_MOSI 3根线. 当NOSLVSEL bit 被设为 1时, GO_BUSY bit被设为1并且串行时钟出现以后，控制器就开始发送/接收数据. 在无从设备片选信号模式下, SS_LTRIG, SPI_SSR[4], 都应该被设为1

.

中断

数据传输完毕时，每一个SPI控制器会产生一个独立的中断，并且各自的中断事件标志**IF** (SPI_CNTRL[16])将会被置位。如果中断使能位**IE** (SPI_CNTRL[17]) 置位，则中断将发生。中断标志只能通过向其写1清零。

3-线模式下，当传输开始并且接收到的数据满足TX_BIT_LEN 和 TX_NUM 中的定义时，SLV_START_INTSTS中断标志将被设。如果接收到的bits数小于设定值，并且没有串行时钟输入超过用户设定的时间，用户可以设定SLV_ABORT bit来迫使当前的传输完成，然后用户将得到一个传输完成中断.

6.7.5 SPI 时序波形图

在主机/从机模式下，设备/从机选择信号(SPISS) 的有效电平可以在**SS_LVL** 位 (SPI_SSR[2])被编程为低电平有效或高电平有效，但是SPISSx0/1是电平触发还是边沿触发在SS_LTRIG 位 (SPI_SSR[4])中定义。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。在**Tx_BIT_LEN** (SPI_CNTRL[7:3])中配置传输字的长度，在**Tx_NUM** (SPI_CNTRL[8])中配置传输的次数，在**LSB** bit (SPI_CNTRL[10])中配置发送/接收数据是MSB还是LSB优先。用户还可以在寄存器Tx_NEG/Rx_NEG (SPI_CNTRL[2:1])中选择在时钟的上升沿还是下降沿发送/接收数据。主机/从机的四种SPI操作时序图和相关的设定如下所示。

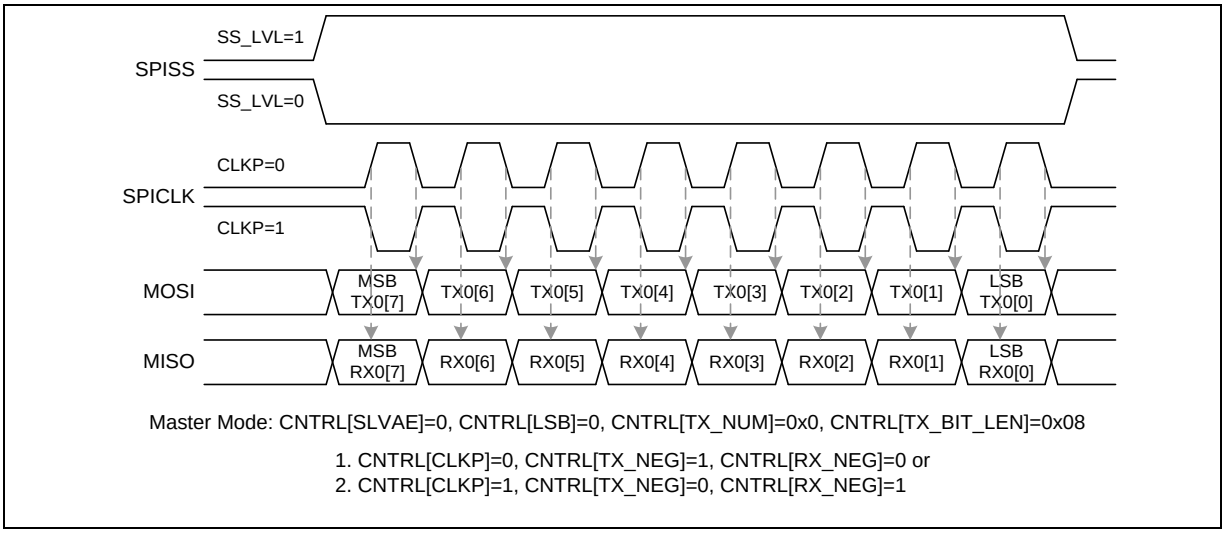


图 6.7.5-1 主机模式下 SPI 时序

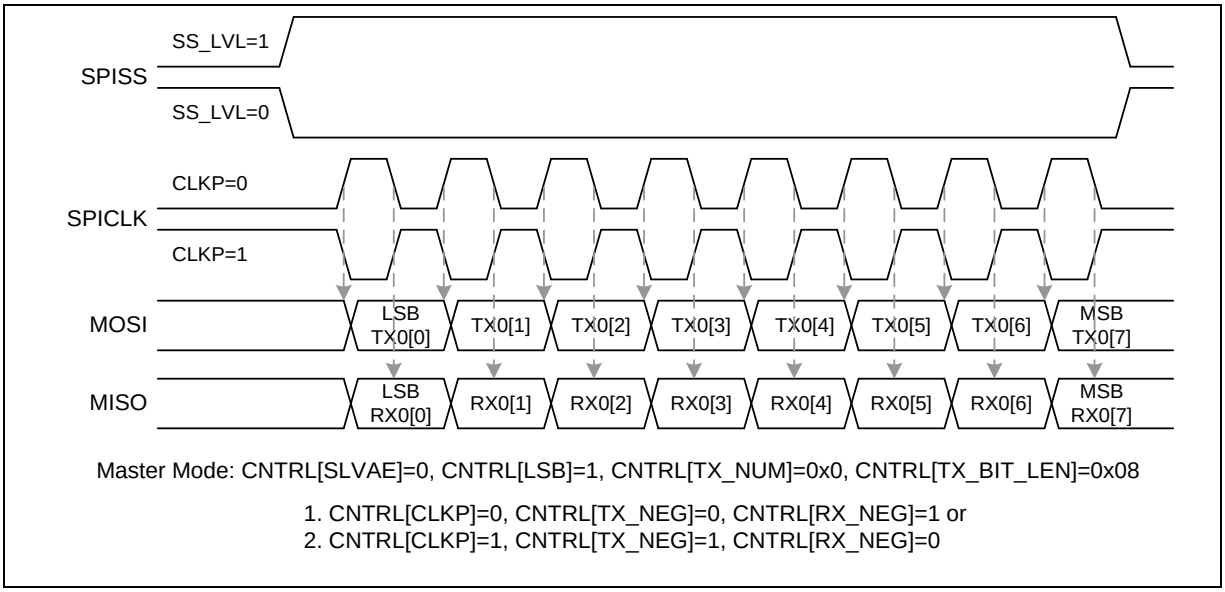


图 6.7.5-2 主机模式下 SPI 时序(Alternate Phase of SPICLK)

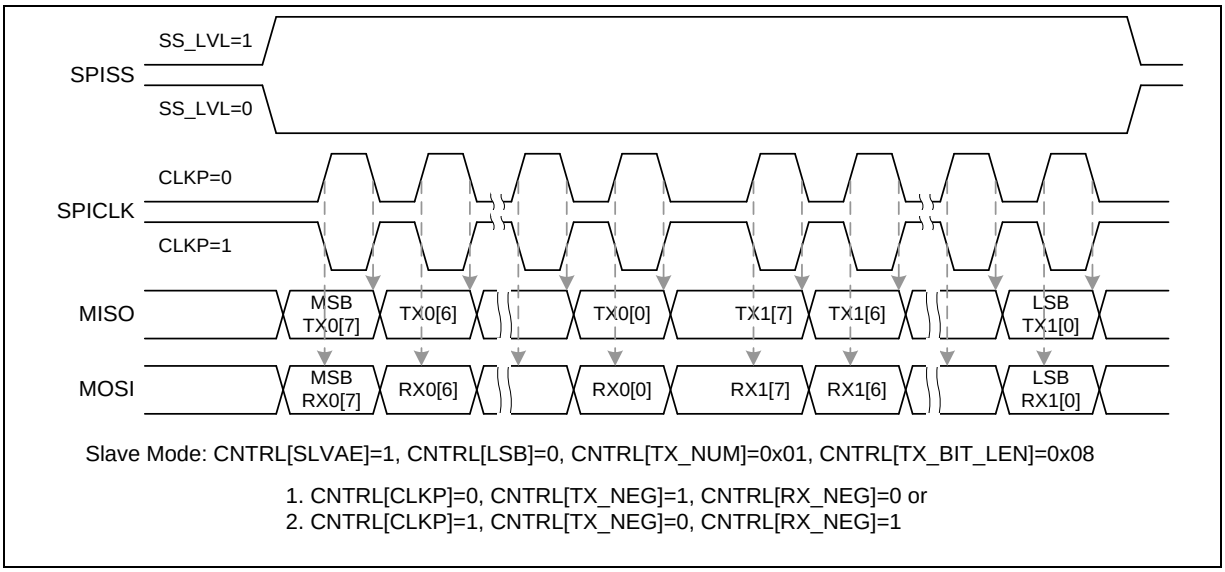


图 6.7.5-3 从机模式下 SPI 时序

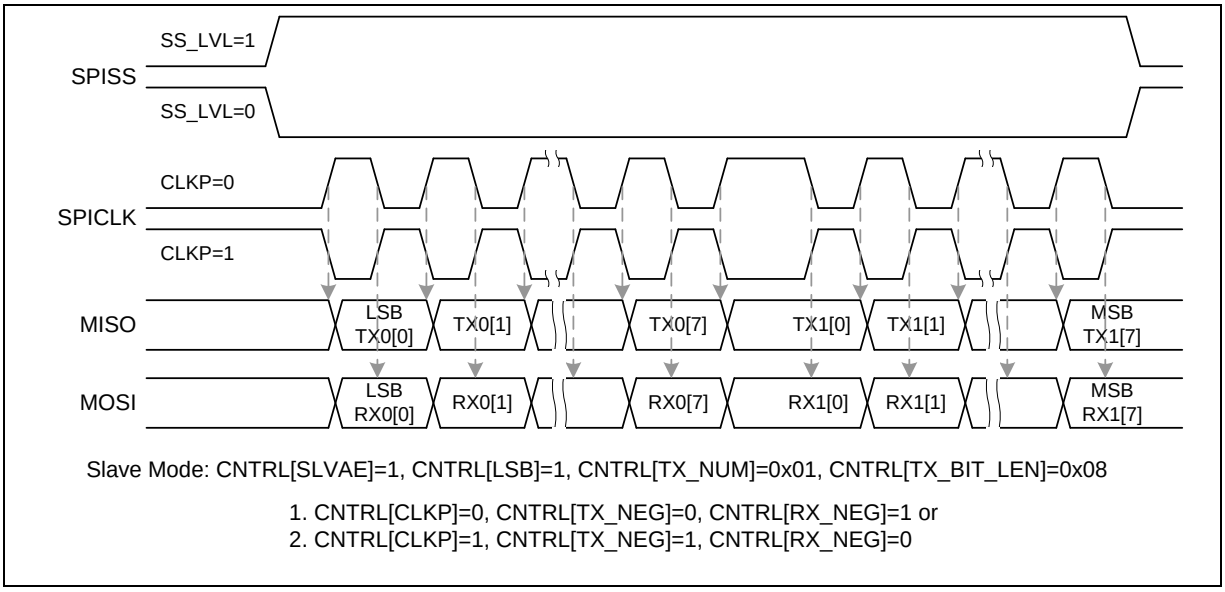


图 6.7.5-4 从机模式下 SPI 时序(Alternate Phase of SPICLK)

6.7.6 SPI 编程范例

■ 例 1, SPI 控制器作为主机，按如下说明访问一个片外从机设备：

- 数据在时钟上升沿锁存
- 数据在时钟下将沿传输
- MSB优先
- 空闲时SPICLK为低电平
- 每次发送/接收一个字节
- 从机选择信号低有效

基本上，在执行如下这些步骤之前，还请详细查阅片外从机的说明。

1) 设置DIVIDER (SPI_DIVIDER[15:0])寄存器，配置串行时钟输出频率.
2) 向SPI_SSR写入适当的值，用于主机模式的相应设置
   1. 禁用自动片选AUTOSS (SPI_SSR[3] = 0)
   2. 配置从机片选有效电平位**SS_LVL** (SPI_SSR[2] = 0)，低电平有效
   3. 通过设置从机片选寄存器位SSR[0] (SPI_SSR[0])，来使从机片选信号在I/O引脚上输出有效，以激活片外从机设备
3) 向寄存器SPI_CNTRL 写入相应设置，控制SPI主机动作
   1. 通过**SLAVE** 位 (SPI_CNTRL[18] = 0)设置SPI控制器为主机设备
   2. 通过**CLKP**位(SPI_CNTRL[11] = 0) 设置串行时钟空闲状态为低
   3. 通过**Tx_NEG**位(SPI_CNTRL[2] = 1)选择数据在串行时钟的下降沿发送
   4. 通过**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**Tx_BIT_LEN**位域 (SPI_CNTRL[7:3] = 0x08)设置传输字的长度为8位
   6. 通过 **Tx_NUM** (SPI_CNTRL[9:8] = 0x0) 设置为一次字传输
   7. 通过 **LSB** 位 (SPI_CNTRL[10] = 0) 设置为MSB优先传输，不必关心SP_CYCLE (SPI_CNTRL[15:12])的设置，因为在本例中没有burst模式
4) 如果SPI主机要发送一个字节的数据到片外从机设备，把将要发送的字节数据写入寄存器Tx0[7:0] (SPI_Tx0[7:0])。
5) 如果SPI主机要从外设接收一个字节的数据，不必关心什么数据被传输，只需要向寄存器SPI_Tx0[7:0]写入0XFF .
6) 使能**GO_BUSY** 位(SPI_CNTRL[0] = 1)，以开始数据传输。
7) 等待SPI中断发生（IE使能），或检测**GO_BUSY** 位直到其被硬件自动清零
8) 从寄存器**Rx0[7:0]** (SPI_Rx0[7:0])读出所接收到的一个字节的数据.

9) 跳转到步聚 4) 继续其他数据的传输或设置**SSR[0]**为0 取消片选，以停止外设访问.

■ 例 2, SPI控制器作为从机设备，由片外主机设备控制，片外主机设备依如下说明通过SPI接口访问片上SPI从机：

- 数据在时钟上升沿锁存
- 数据在时钟下降沿传输
- LSB优先
- 空闲时SPICLK为高电平
- 每次输出/接收一个字节
- 从机片选信号高电平有效

基本上，在执行如下这些步骤之前，还请详细查阅片外主机的说明。

1) 向SPI_SSR写入适当的值，用于从机模式的相应设置。写SS_LVL (SPI_SSR[2] = 1)配置从机片选有效电平为高电平，写**SS_LTRIG** (SPI_SSR[4] = 1).配置从机片选为电平触发

2) 向寄存器SPI_CNTRL 写入相应配置以控制SPI从机

   1. 通过**SLAVE**位(SPI_CNTRL[18] = 1)设置SPI控制器为从机设备
   2. 通过**CLKP**位(SPI_CNTRL[11] = 1)选择串行时钟空闲状态为高
   3. 通过 **Tx_NEG**位(SPI_CNTRL[2] = 1)选择数据在串行时钟下降沿发送
   4. 通过**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**Tx_BIT_LEN**位field (SPI_CNTRL[7:3] = 0x08)设置字传输长度为8位
   6. 通过**Tx_NUM** (SPI_CNTRL[9:8] = 0x0)设置为仅一次字传输
   7. 通过 **LSB** 位 (SPI_CNTRL[10] = 1) 设置为 LSB 优先传输，不必关心 SP_CYCLE (SPI_CNTRL[15:12])的设置，因为在本例中没有burst模式

3) 如果SPI从机要发送(被读取)一个字节数据到片外主机设备，把将要发送的数据写入寄存器Tx0[7:0] (SPI_Tx0[7:0])。

4) 如果SPI从机仅从外设主机接收一字节数据，用户不必关心什么数据将被传输，只需要向寄存器SPI_Tx0[7:0]写入0XFF.

5) 使能**GO_BUSY** bit (SPI_CNTRL[0] = 1) ，等待外设的从机选择触发输入和串行时钟输入，开始数据传输到SPI接口。

   --等待SPI中断发生（IE使能），或检测**GO_BUSY** 位直到其被硬件自动清零--

6) 从**Rx[7:0]** (SPI_Rx0[7:0]) 寄存器中读出所接收到的一个字节的数据

7) 跳转到步聚3) 继续其他数据传输或禁用**GO_BUSY** 位停止数据传输.

6.7.7 SPI 串行总线控制寄存器映射

R: 只读, **W**: 只写, **R/W**: 可读写

寄存器	偏移量	R/W	描述	复位后的值
SPI0_BA = 0x4003_0000 SPI1_BA = 0x4003_4000				
SPI_CNTRL	SPIx_BA + 0x00	R/W	控制及状态寄存器	0x0500_0004
SPI_DIVIDER	SPIx_BA + 0x04	R/W	时钟分频寄存器	0x0000_0000
SPI_SSR	SPIx_BA + 0x08	R/W	从机片选寄存器	0x0000_0000
SPI_RX0	SPIx_BA + 0x10	R	数据接收寄存器0	0x0000_0000
SPI_RX1	SPIx_BA + 0x14	R	数据接收寄存器1	0x0000_0000
SPI_TX0	SPIx_BA + 0x20	W	数据发送寄存器0	0x0000_0000
SPI_TX1	SPIx_BA + 0x24	W	数据发送寄存器1	0x0000_0000
SPI_VARCLK	SPIx_BA + 0x34	R/W	可变时钟模型控制寄存器	0x007F_FF87
SPI_CNTRL2	SPIx_BA+0x3C	R/W	Control and Status Register 2	0x0000_0000

注 1: 当软件编程CNTRL 寄存器时, GO_BUSY 位必须最后写入.

6.7.8 SPI 控制寄存器描述

SPI 控制与状态寄存器(SPI_CNTRL)

寄存器	偏移量	R/W	描述	复位后的值
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							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
VARCLK_EN	保留		**REORDER**		**SLAVE**	**IE**	**IF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SP_CYCLE				**CLKP**	**LSB**	**TX_NUM**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TX_BIT_LEN					**TX_NEG**	**RX_NEG**	**GO_BUSY**

Bits	描述	
[31:24]	保留	保留
[23]	**VARCLK_EN**	**可变时钟使能 (仅主机)** 0 = 串行时钟输出仅由DIVIDER的值决定. 1 = 串行时钟输出可变. 输出频率由VARCLK, DIVIDER, 和 DIVIDER2的值决定. 注：当使能VARCLK_EN, TX_BIT_LEN 必须设置成 0x10 (16 bits 模式)
[22:21]	保留	保留
[20:19]	**REORDER**	**重排序模式选择** 00 = 禁用字节重排序和字节休眠功能. 01 = 使能字节重排序，并在每个字节之间插入一个字节休眠间隔 (2~17 串行时钟周期). TX_BIT_LEN 必须设置成0x00. (32 bits/word) 10 = 使能字节重排序功能，但禁用字节休眠功能. 11 = 禁用字节重排序功能，但在每个字节之间插入一个休眠间隔(2~17 串行时钟周期). TX_BIT_LEN 必须设置成0x00. (32 bits/word)
[18]	**SLAVE**	**从机模式选择**

Bits	描述	
		0 = 主机模式 1 = 从机模式
[17]	**IE**	**中断使能** 0 = 禁用SPI 中断. 1 = 使能SPI中断.
[16]	**IF**	**中断标志** 0 = 表示传输未结束 1 = 表示传输完成 **注：**该位写1清零.
[15:12]	**SP_CYCLE**	**休眠间隔 (仅主机模式)** 该四位用于设定在两次连续传输间的间隔时间。如果CLKP=0，间隔时间从当前传输的最后一个时钟下降沿到下次传输的第一个时钟上升沿。如果CLKP=1，间隔时间从时钟上升沿到时钟下降沿。默认值为0x0。当Tx_NUM = 00, 该位无效。下列公式可获得所需的间隔时间。 ■ 用于配置字节传输间隔和burst模式的传输间隔: (SP_CYCLE[3:0] + 2) * SPI时钟周期+1个系统时钟周期 例如： SP_CYCLE = 0x0 … 2个SPI时钟周期+1个系统时钟周期 SP_CYCLE = 0x1 … 3个SPI时钟周期+1个系统时钟周期 …… SP_CYCLE = 0xe … 16个SPI时钟周期+1个系统时钟周期 SP_CYCLE = 0xf … 17个SPI时钟周期+1个系统时钟周期 如果SPI时钟频率等于系统时钟频率，就是说DIV_ONE特性被使能，burst模式间隔周期为 **(SP_CYCLE[3:0] * 2 + 3.5) * 系统时钟周期**
[11]	**CLKP**	**时钟极性** 0 = 空闲时SPICLK 为低电平 1 = 空闲时SPICLK 为高电平.
[10]	**LSB**	**优先传送LSB** 0 = 优先发送/接收MSB (具体是SPI_TX0/1和SPI_RX0/1 寄存器的哪一位取决于TX_BIT_LEN的值). 1 = 优先发送 LSB (SPI_TX0/1的bit 0), 接收到的首位数居被送入 Rx 寄存器的LSB位置(SPI_RX0/1的bit 0)..

Bits	描述	
[9:8]	**TX_NUM**	**发送/接收数量** 该寄存器用于标示一次成功传输中，传输的笔数。 00 = 每次传输仅完成一次发送/接收 01 = 每次传输完成两次连续的发送/接收 10 = 保留. 11 = 保留. 注意: 从机模式下，电平触发时如果 TX_NUM 等于 01, 在连续的数据传输之间，从设备片选引脚必须保持有效
[7:3]	**TX_BIT_LEN**	**传输位长度** 该寄存器用于说明一次传输中，传输的bit长度，最高为32位。 Tx_BIT_LEN = 0x01 … 1位 Tx_BIT_LEN = 0x02 … 2位 …… Tx_BIT_LEN = 0x1f … 31位 Tx_BIT_LEN = 0x00 … 32位
[2]	**TX_NEG**	**下降沿发送数据** 0 = 发送的数据在SPICLK的上升沿改变。 1 = 发送的数据在SPICLK的下降沿改变.
[1]	**RX_NEG**	**下降沿接收数据** 0 = 接收到的数据在SPICLK上升沿锁存 1 = 接收到的数据在SPICLK下降沿锁存.
[0]	**GO_BUSY**	**开始传输和忙状态标志** 0 =当SPI正在传输时，对该位写0会使数据传输停止 1 =主机模式下，对该位写1开启SPI数据传输；从机模式下，对该位写1表明从机已准备好与主机的通讯。 注: 在对CNTRL寄存器的GO_GOBY置1之前，必须先配置相应的寄存器。已经开始传输才对其他位进行配置，无法影响当前传输.

SPI 分频寄存器 (SPI_DIVIDER)

寄存器	偏移量	R/W	描述	复位后的值
SPI_DIVIDER	SPIx_BA + 0x04	R/W	时钟分频寄存器(仅主机模式)	0x0000_0000

31	30	29	28	27	26	25	24
DIVIDER2[15:8]							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DIVIDER2[7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DIVIDER[15:8]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVIDER[7:0]							

Bits	描述	
[31:16]	**DIVIDER2**	**时钟分频器2寄存器**（仅主机模式) 这个域的值是第二个频率除数，用于产生串行时钟输出SPICLK。可以根据下列方程获得期望的频率: $f_{sclk} = \frac{f_{psclk}}{(DIVIDER2+1)*2}$ 如果VARCLK_EN被清除成0，这个域的设定没有意义
[15:0]	**DIVIDER**	**时钟分频器寄存器**(仅主机模式) 这个域的值是频率除数，用于产生串行时钟输出SPICLK。根据下列方程获得期望的频率 $f_{sclk} = \frac{f_{psclk}}{(DIVIDER+1)*2}$ 从机模式下，由主机提供的SPI时钟周期，应该大于或等于PCLK周期的5倍. 换言之，输入的SPI时钟的最大频率为从机PCLK的1/5.

SPI从机片选寄存器(SPI_SSR)

寄存器	偏移量	R/W	描述	复位后的值
SPI_SSR	SPI0_BA + 0x08	R/W	从机片选寄存器	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**

Bits	描述	
[31:6]	保留	保留
[5]	**LTRIG_FLAG**	**电平触发标志** 在从机模式下SS_LTRIG置位，该标志能够表示接收到的位数量是否达到要求。 1: 接收数量和接收位长度达到TX_NUM 及TX_BIT_LEN定义的值. 0: 接收数量或接收位长度没有达到TX_NUM 及TX_BIT_LEN定义的值。. 注：该位只读
[4]	**SS_LTRIG**	**从机电平触发选择（从机模式）** 0: 从机片选信号为边沿触发。根据 SS_LVL选择是上升沿/下降沿触发 1: 从机片选信号为电平触发。根据 SS_LVL选择是高电平/低电平触发.
[3]	**AUTOSS**	**自动从机片选(主机模式)** 0 = 该位清0，从机选择信号是否有效，由设置或清除SSR[0]寄存器决定。. 1 = 该位置位，SPISS0/1信号自动产生。这说明在SSR[0]寄存器内的从机选择信号，在置位GO_BUSY开始发送/接收时由SPI控制器自动置为有效，，并且在传输结束后自动变为无效。
[2]	**SS_LVL**	**从机选择激活电平** 该位决定SPISS0/1寄存器内从机选择信号的激活电平 0 = SPISS0/1 从机片选为低电平/下降沿时有效。 1 = SPISS0/1从机片选为高电平/上升沿时有效。

Bits	描述	
[1]	保留	保留
[0]	**SSR**	**从机片选寄存器(主机模式)** 当AUTOSS位被清除，对SSR位写1，将会激活SPISSx0/1线，写0片选变为无效状态。 当AUTOSS位被置1，对SSR位写1，将会使SPISSx线上在传输/接受数据时自动驱动至激活状态。在其他时间驱动至非有效状态(由SS_LVL决定有效电平). 注：SPISSx通常在从机模式下被定义为设备/从机片选输入.

SPII数据接收寄存器 (SPI_RX)

寄存器	偏移量	R/W	描述	复位后的值
SPI_RX0	SPIx_BA + 0x10	R	数据接收寄存器 0	0x0000_0000
SPI_RX1	SPIx_BA + 0x14	R	数据接收寄存器1	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]							

Bits	描述	
[31:0]	**RX**	**数据接收寄存器** 数据接收寄存器内保存最后一次传输所接收的数据。数据的长度根据SPI_CNTRL寄存器内配置的长度决定。例如，Tx_BIT_LEN 设定为 0x08 且 Tx_NUM 设定为0x0, Rx0[7:0] 内保存收到的数据，其它bit为无效值. **注:** 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	偏移量	R/W	描述	复位后的值
SPI_TX0	SPIx_BA + 0x20	W	数据发送寄存器0	0x0000_0000
SPI_TX1	SPIx_BA + 0x24	W	数据发送寄存器1	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]							

Bits	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。数据的长度根据CNTRL寄存器内配置的长度决定。例如，Tx_BIT_LEN设定为 0x08 且Tx_NUM 设定为0x0, Tx0[7:0] 内的数据将被发送。如果[Tx_BIT_LEN 设定为 0x00 且Tx_NUM 设定为0x1,模块将用同种设置连续进行两笔32位数据发送/接收，（顺序是先TX0[31:0],再TX1[31:0]）。

SPI 可变时钟模型寄存器(SPI_VARCLK)

寄存器	偏移量	R/W	描述	复位后的值
SPI_VARCLK	SPIx_BA + 0x34	R/W	可变时钟模型寄存器	0x007F_FF87

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

Bits	描述	
[31:0]	**VARCLK**	可调时钟类型 该值为SPI时钟频率模型. 如果VARCLK为'0', SPICLK的输出频率取决于DIVIDER的值 . 如果VARCLK 为 '1', SPICLK的输出频率取决于DIVIDER2的值. 参考寄存器 SPI_DIVIDER. 参考上面可变时钟时序框图。

SPI 控制和状态寄存器2 (SPI_CNTRL2)

Register	Offset	R/W	Description	Reset Value
SPI_CNTRL2	SPIx_BA+0x3C	R/W	第二个控制与状态寄存器	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				SLV_START_INTSTS	SSTA_INTEN	SLV_ABORT	NOSLVSEL
7	6	5	4	3	2	1	0
保留							DIV_ONE

Bits	描述	
[31:12]	保留	保留
[11]	SLV_START_INTSTS	**从机开始传输状态** 从模式下，没有片选信号时（3线模式）用来指示传输已经开始 1 =从模式下，没有片信号号时（3线模式）用来指示传输已经开始，当传输结束或者写1可以清除该位. 0 = 传输还没有开始.
[10]	SSTA_INTEN	**从机开始传输中断使能** 从模式下，没有片选信号时（3线模式），如果设定该位为1，当传输开始时中断将发生。如果传输开始以后，在用户定义的时间内没有发生传输结束中断，用户可以设定SLV_ABORT bit来迫使传输结束. 1 = 使能传输开始中断。当传输结束或者SLV_START_INTSTS bit 被清0，这个bit将被清成0. 0 = 禁止传输开始中断.
[9]	SLV_ABORT	**从模式下，没有从设备片选信号时，用来终止传输** 正常情况下，当接收到的数据满足TX_BIT_LEN 和 TX_NUM 的设定时，中断将发生. 如果接收到的bits 小于请求并且没有串行时钟输入超时用户设定的时间，用户可以设定这个bit为1来迫使当前传输结束，然后用户将得到一

Bits	描述	
		个传输结束中断. 注意: 当终止事件发生以后，硬件自动将这个bit清0
[8]	**NOSLVSEL**	**无从机片选模式** 从模式下，用来忽略从机片选信号。SPI控制器可以工作在3线模式，包括SPICLK, SPI_MISO, 和 SPI_MOSI. 0 = 控制器为4-线双向接口. 1 = 从模式下，控制器为3-线状向接口。当这个 bit被设为 “1” 时, 设定 GO_BUSY bit为1并且有串行时钟输入，控制器就开始收/发数据 注意: 无从机片选信号时，SS_LTRIG, SPI_SSR[4], 应该被设为 1.
[7:1]	**保留**	保留
[0]	**DIV_ONE**	**SPI 时钟除频控制** 0 = SPI 时钟频率由SPI_DIVIDER 寄存器决定. 1 = 使能 DIV_ONE 特性. SPI 时钟频率等于系统时钟频率. 注意: 1. 当这个bit 设为1时, REORDER 域和VARCLK_EN 域必须都被配置为0. 换句话说，字节重新排序，字节传输暂停和可变时钟功能必须关闭. 2. 当这个bit 设为1时, TX_BIT_LEN 不可以设为 1.

6.8 定时器控制器

6.8.1 概述

定时器控制器包括4组32位的定时器，TIMER0~TIMER3, 方便用户的定时控制应用。定时器模块可支持例如频率测量，计数，间隔时间测量，时钟产生，延迟时间等功能。定时器可在计时溢出时产生中断信号，也可在计数过程中提供计数的当前值。

6.8.2 特征

- 4 组 32-位定时器，带24位向上定时器和一个8位的预分频计数器
- 每个定时器都有独立的时钟源
- 4种工作模式：单脉冲模式(one-shot), 周期模式(periodic),反转输出模式(toggle)和连续计数(continuous counting)模式操作模式
- 超时周期= (定时器时钟源的周期) * (8-bit 预分频 + 1) * (24-bit TCMP)
- 最大计数周期 = (1 / T MHz) * (2^8) * (2^{24}), T 是定时器时钟源的周期
- 24位向上计数器的值，通过TDR（定时器数据寄存器）可读取
- 支持事件计数功能，可以数外部输入信号的事件个数
- 支持输入捕捉功能，可以捕捉或者复位计数器的值

6.8.3 定时器方框图

每个通道带一个8位预分频计数器，一个24位向上计数器，一个24位比较寄存器和一个中断请求信号。参考图6.8.3-1的定时器控制框图。每个通道有4个时钟源选项，图 6.8-2说明了时钟源控制功能。 软件可以编程配置8位预分频系数来确定24位向上计数器的时钟周期。

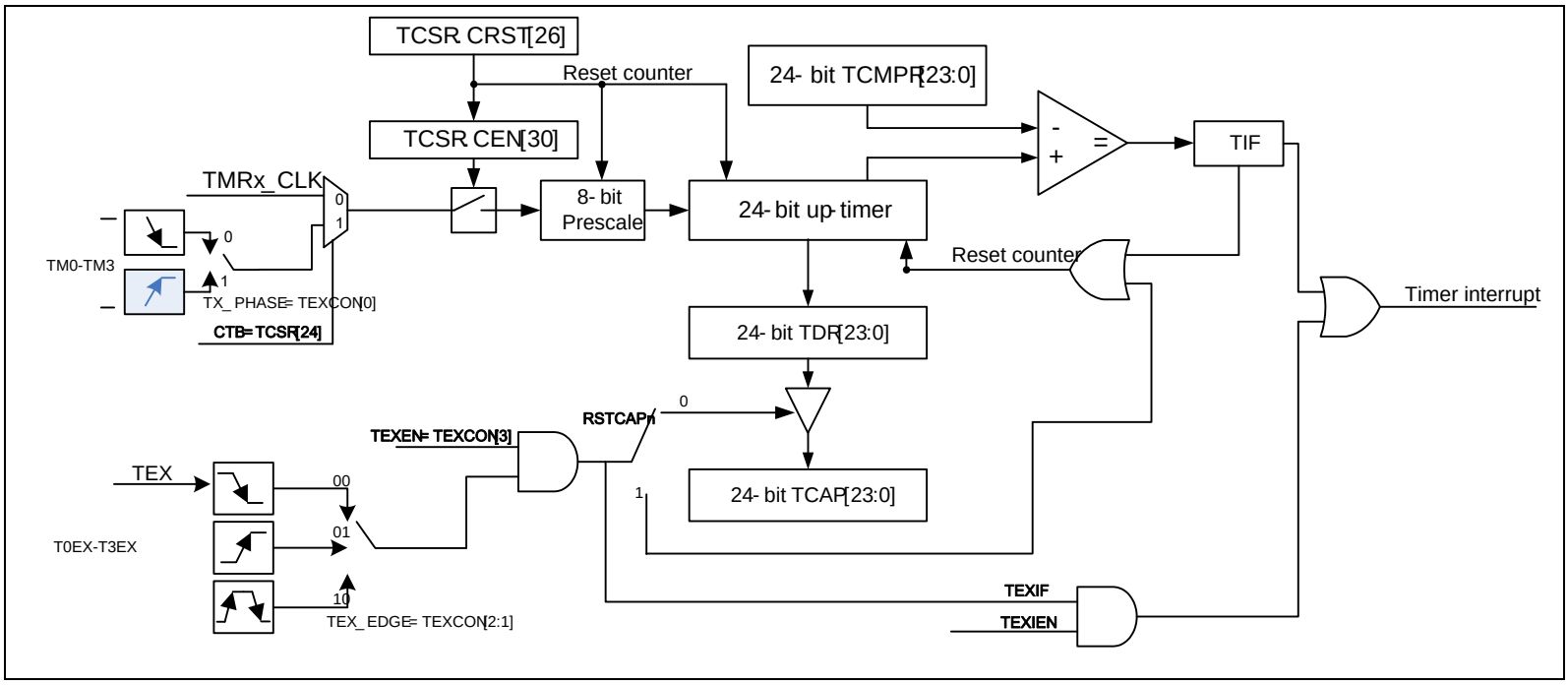


图 6.8.3-1 定时器控制器框图

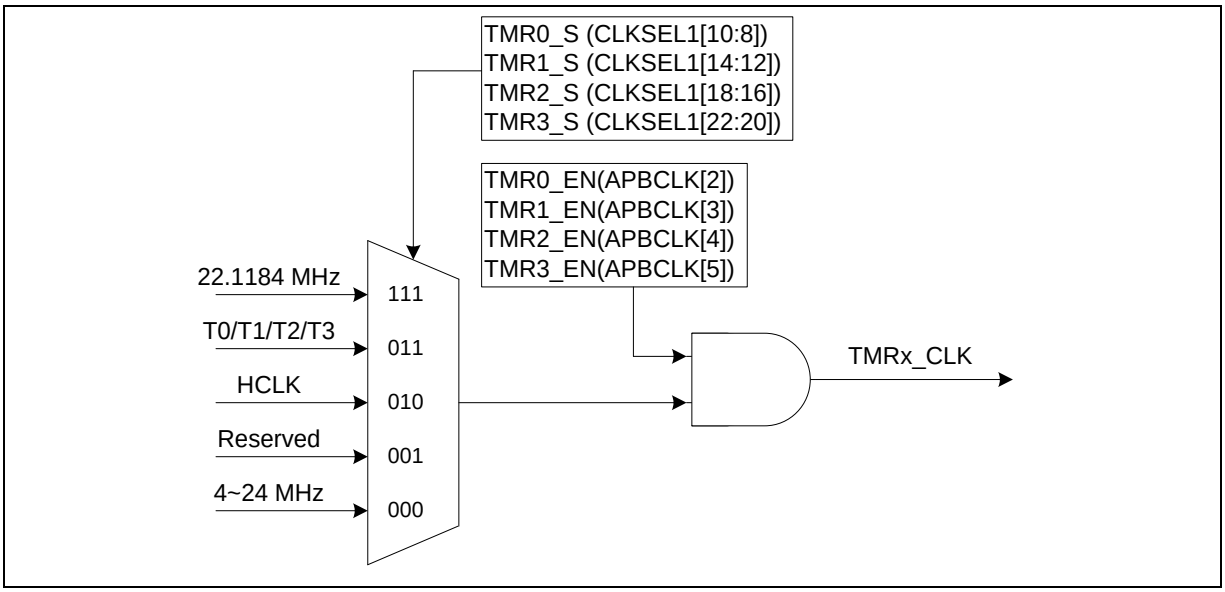


图 6.8.3-2 定时器控制的时钟源

6.8.4 功能描述

定时器控制器提供4种工作模式，单脉冲(one-shot)模式、周期(periodic)模式、反转（toggle）和 连续计数(continuous counting)模式。. 每种操作模式如下所述:

6.8.4.1 一般定时器功能

单脉冲模式

如果定时器工作在单脉冲 模式且CEN (定时器使能位) 置1, 定时器的计数器开始计数. 一旦定时器计数器的值达到定时器比较器寄存器 (TCMPR) 的值, 且IE (中断使能位) 置1, 则定时器中断标志置位，产生中断信号并送到NVIC通知CPU. 表明定时器计数发生溢出. 如果IE (interrupt enable bit) 置0, 无中断信号产生. 在此工作模式下, 一旦定时器计数器的值达到定时器比较寄存器 (TCMPR) 的值, 定时器计数器的值返回初始值且CEN (定时器使能位)由定时器控制器清零. 一旦定时器计数器的值达到定时器比较寄存器(TCMPR) 的值，定时器计数操作停止,。也就是说, 在编程比较寄存器（TCMPR）的值与CEN(定时器使能位)置1后，定时器操作定时器计数和与TCMPR值的比较仅执行一次.因此，该操作称为单脉冲 模式.

周期模式

如果定时器工作在周期模式且CEN (定时器使能位)置1, 定时器计数器开始计数. 一旦定时器计数器的值达到定时器比较寄存器 (TCMPR)的值, 且IE (中断使能位) 设置为1, 则定时器中断标志置位且产生中断信号，并发送到NVIC通知CPU. 表示定时器计数溢出发生. 如果IE (中断使能位)设置为0, 无中断信号发生. 在该工作模式下, 一旦定时器计数器的值达到定时器比较器寄存器(TCMPR) 的值, 定时器计数器的值返回计数初始值且CEN 保持为1 (持续使能计数). 定时器计数器重新开始计数. 如果软件清除中断标志, 一旦定时器计数器的值与定时器比较寄存器(TCMPR)的值匹配且IE (中断使能位)设置为1 中断标志置位，产生中断信号并送到NVIC再次通知CPU. 也就是说, 定时器操作定时器计数和与TCMPR比较功能是周期性进行的. 直到CEN设置为0，定时器计数操作才会停止.中断信号的产生也是周期性的. 因此, 这种操作模式称为周期模式.

开关模式

如果定时器工作在开关模式且CEN (定时器使能位)置1, 定时器计数器开始计数. 一旦定时器计数器的值与定时器比较寄存器TCMPR的值匹配时, 且IE (中断使能位)设置为 1, 则定时器中断标志置位, 产生中断信号并送到NVIC通知CPU. 表示定时器发生计数溢出. 相应开关输出(tout) 信号置1. 在这种操作模式, 一旦定时器计数器的值与定时器比较寄存器TCMPR的值匹配, 定时器计数器的值返回到计数初始值且 CEN 保持为 1 (持续使能计数). 定时器计数器重新开始计数. 如果中断标志由软件清除，一旦定时器计数器的值与定时器比较寄存器中TCMPR的值匹配且IE (中断使能位) 置1, 则定时器中断标志置位，发生中断信号，并送到NVIC再次通知CPU. 相应开关输出 (tout)信号置0. 定时器计数操作在CEN设置为0之后才停止. 因此, 开关输出 (tout)信号 以50%的占空比反复改变. 所以这种操作模式称为开关模式.

连续计数模式

如果定时器工作在连续计数模式且CEN (定时器使能位)置1, 如果IE(中断使能位)设置为1, 当TDR = TCMPR 时，相关的中断信号产生。用户可以立即改变TCMPR的值，而不需要禁用或重启定时器计数。例如，TCMPR的值先被设置为80(TCMPR的值应当小于2^{24} -1并且大于1)， 当TDR的值等于80时，如果IE (中断使能位)设置为1，定时器产生中断，TIF(定时器中断标志)将被置位，产生中断

信号并送到NVIC通知CPU，且CEN 保持为 1 (持续使能计数)，但是TDR的值不会返回到零，而是继续计数81, 82, 83, · · · to 2^{24} -1, 0, 1, 2, 3, · · · to 2^{24} -1。接下来，如果用户设置TCMPR为200，且TIF被清零，当TDR的值达到200，定时器中断发生，TIF被置位，产生中断信号并送到NVIC再次通知CPU。最后，用户设置TCMPR为500，并再一次清零TIF，当TDR的值达到500，定时器中断发生，TIF被置位，产生中断信号并送到NVIC通知CPU。从应用的角度看，中断的产生取决于TCMPR。在该模式下，定时器计数是连续的，所以这种操作模式被称为连续计数模式.

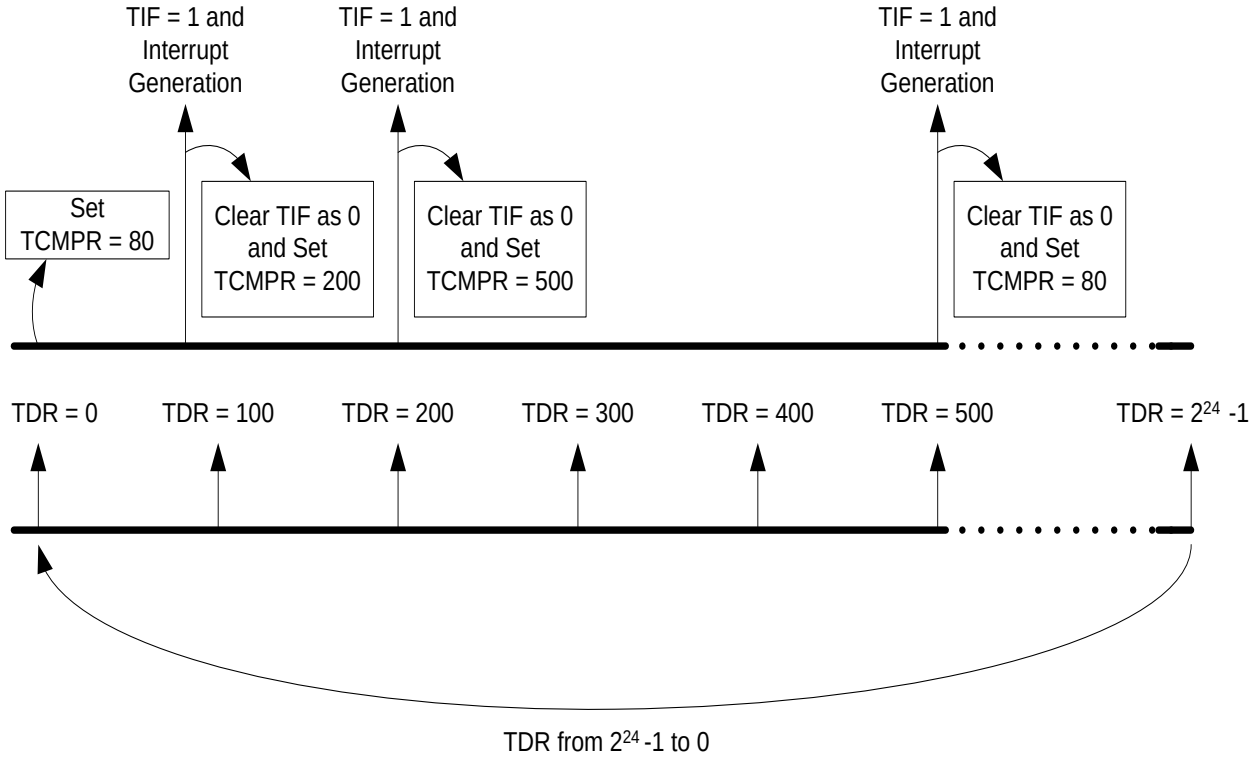


图6.8.4-1 连续计数模式

6.8.4.2 事件计数功能

定时器提供一个功能可以用来数外部T0~T3引脚的事件个数，被称为事件计数功能。事件计数功能中，时钟源, TMRx_CLK, 如 **錯誤! 找不到參照來源。** 可以是 HCLK。 TEXCONx[7]提供 T0~T3 使能或者禁止防弹跳功能。通过TEXCONx[0]可以设定T0~T3 下降沿或者上升沿计数。如果禁止防弹跳功能，事件计数源的频率应该小于HCLK 频率的1/3。如果使能防弹跳功能，事件计数源的频率应该小于HCLK 频率的1/8 。否则，返回的TDR的值将不正确.

6.8.4.3 外部捕获/复位功能

计数器提供输入捕获功能，可以用来捕获或者复位计数器的值。如果TEXEN (定时器外部引脚使能) 被设为 1 并且 RSTCAPSEL 为 0，当TEX(定时器外部引脚)引脚触发条件满足时，定时器计数器的值 (TDR) 将被捕获到TCAP 寄存器。共有4个TEX 源来自指定的引脚, T0EX~T3EX 引脚。如果 TEXEN 为 1 并且 RSTCAPSEL 也为1，当TEX(定时器外部引脚)引脚触发条件满足时，TDR 将被复位成0。TEX 触发沿可以通过TEX_EDGE配置。当TEX 触发发生时, TEXIF (定时器外部中断标志) 将被设为 1，如果TEXIEN (定时器外部中断使能)为1, 中断将发生。通过TEXCONx[6]可以使能/禁止 T0EX~T3EX 捕获防弹跳功能。如果禁止TEX防弹跳功能，TEX输入信号的频率应该小于 HCLK 频率的1/3；如果禁止TEX防弹跳功能，TEX输入信号的频率应该小于HCLK 频率的1/8.

6.8.5 定时器控制器寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位后的值
TMR_BA01 = 0x4001_0000 **TMR_BA23 = 0x4011_0000**				
TCSR0	TMR_BA01+00	R/W	Timer0控制和状态寄存器	0x0000_0005
TCMPR0	TMR_BA01+04	R/W	Timer0比较寄存器	0x0000_0000
TISR0	TMR_BA01+08	R/W	Timer0 中断状态寄存器	0x0000_0000
TDR0	TMR_BA01+0C	R	Timer0 数据寄存器	0x0000_0000
TCAP0	TMR_BA01+0x10	R	Timer0 捕获数据寄存器	0x0000_0000
TEXCON0	TMR_BA01+0x14	R/W	Timer0 外部控制寄存器	0x0000_0000
TEXISR0	TMR_BA01+0x18	R/W	Timer0 外部中断状态寄存器	0x0000_0000
TCSR1	TMR_BA01+20	R/W	Timer1 控制和状态寄存器	0x0000_0005
TCMPR1	TMR_BA01+24	R/W	Timer1 比较寄存器	0x0000_0000
TISR1	TMR_BA01+28	R/W	Timer1 中断状态寄存器	0x0000_0000
TDR1	TMR_BA01+2C	R	Timer1 数据寄存器	0x0000_0000
TCAP1	TMR_BA01+0x30	R	Timer1捕获数据寄存器	0x0000_0000
TEXCON1	TMR_BA01+0x34	R/W	Timer1外部控制寄存器	0x0000_0000
TEXISR1	TMR_BA01+0x38	R/W	Timer1外部中断状态寄存器	0x0000_0000
TCSR2	TMR_BA23+00	R/W	Timer2 控制和状态寄存器	0x0000_0005
TCMPR2	TMR_BA23+04	R/W	Timer2 比较寄存器	0x0000_0000
TISR2	TMR_BA23+08	R/W	Timer2 中断状态寄存器	0x0000_0000
TDR2	TMR_BA23+0C	R	Timer2 数据寄存器	0x0000_0000
TCAP2	TMR_BA23+0x10	R	Timer2捕获数据寄存器	0x0000_0000
TEXCON2	TMR_BA23+0x14	R/W	Timer2外部控制寄存器	0x0000_0000
TEXISR2	TMR_BA23+0x18	R/W	Timer2外部中断状态寄存器	0x0000_0000
TCSR3	TMR_BA23+20	R/W	Timer3 控制和状态寄存器	0x0000_0005
TCMPR3	TMR_BA23+24	R/W	Timer3 比较寄存器	0x0000_0000

寄存器	偏移量	R/W	描述	复位后的值
TISR3	TMR_BA23+28	R/W	Timer3 中断状态寄存器	0x0000_0000
TDR3	TMR_BA23+2C	R	Timer3 数据寄存器	0x0000_0000
TCAP3	TMR_BA23+0x30	R	Timer3捕获数据寄存器	0x0000_0000
TEXCON3	TMR_BA23+0x34	R/W	Timer3外部控制寄存器	0x0000_0000
TEXISR3	TMR_BA23+0x38	R/W	Timer3外部中断状态寄存器	0x0000_0000

定时器控制寄存器 (TCSR)

寄存器	偏移量	R/W	描述	复位后的值
TCSR0	TMR_BA01+000	R/W	Timer0 控制与状态寄存器	0x0000_0005
TCSR1	TMR_BA01+020	R/W	Timer1控制与状态寄存器	0x0000_0005
TCSR2	TMR_BA23+000	R/W	Timer2控制与状态寄存器	0x0000_0005
TCSR3	TMR_BA23+020	R/W	Timer3控制与状态寄存器	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**
保留							**TDR_EN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PRESCALE[7:0]							

Bits	描述	
[31]	**DBGACK_TMR**	**ICE 调试模式下禁止应答 (写保护)** 0 = ICE 调试模式应答影响定时器计数. ICE调试模式下，定时器计数器的值将维持不变. 1 = ICE 调试模式应答关闭. 无论是否在ICE调试模式下，定时器计数器的值都保持正常计数.
[30]	**CEN**	**计数器使能位** 0 = 停止/暂停计数 1 = 开始计数 **注1**: 在停止状态，设置CEN为1，使能24位计数器从上次停止的计数值继续计数. **注 2**: 在单脉冲模式下 (MODE[28:27]=00b)，当相应的定时中断产生时 (IE[29]=1b)，该位由硬件自动清零.
[29]	**IE**	**中断使能** 0 = **禁用**定时器中断 1 = **使能**定时器中断

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td></td><td></td><td>如果定时器中断使能，当计数值与TCMPR寄存器内数值相同时，触发中断.</td></tr>
<tr><td>[28:27]</td><td>MODE</td><td>定时器工作模式
<table>
<tr><th>模式</th><th>定时器工作模式</th></tr>
<tr><td>00</td><td>当定时器配置为单脉冲模式(one-shot)时，仅触发一次中断(IE 使能),然后CEN自动清除为0.</td></tr>
<tr><td>01</td><td>当定时器配置为周期模式(period)时，中断(IE 使能)将周期性产生.</td></tr>
<tr><td>10</td><td>定时器工作于反转输出模式. 如果IE使能，中断将周期性产生. 反转信号 (tout) 不停输出，占空比为50%.</td></tr>
<tr><td>11</td><td>定时器工作在连续计数模式。当TDR等于TCMPR时，中断将产生（如果IE使能）. 然而, 24-bit上数计数器仍继续上数. 请参考錯誤! 找不到參照來源。“连续计数模式”.</td></tr>
</table></td></tr>
<tr><td>[26]</td><td>CRST</td><td>计数器复位
设置该位将复位定时器计数器，预分频并使CEN为0.
0 = 无动作.
1 = 复位定时器的预分频计数器，内部24位向上计数器和CEN位</td></tr>
<tr><td>[25]</td><td>CACT</td><td>定时器工作状态（只读）
该位表示当前定时器计数器的状态。
0 = 定时器未工作。
1 = 定时器工作中。</td></tr>
<tr><td>[24]</td><td>CTB</td><td>计数模式使能
这个位是计数模式使能位。当计数器用作事件计数功能时，这个bit应该被设为1。计数器检测的外部引脚相位可以同过TX_PHASE域设定为上升沿或者下降沿.
1 = 使能计数功能
0 = 关闭计数功能</td></tr>
<tr><td>[23:17]</td><td>保留</td><td>保留</td></tr>
<tr><td>[16]</td><td>TDR_EN</td><td>数据读取使能
当置位TDR_EN，TDR (Timer数据寄存器) 将不断更新为24位向上计数器的值
1 = 使能Timer数据寄存器 更新
0 = 禁用Timer数据寄存器 更新</td></tr>
<tr><td>[15:8]</td><td>保留</td><td>保留</td></tr>
<tr><td>[7:0]</td><td>PRESCALE</td><td>预分频计数器
时钟输入根据Prescale+1进行预分频。如果PRESCALE =0，不进行预分频.</td></tr>
</table>

定时器比较寄存器(TCMPR)

寄存器	偏移量	R/W	描述	复位后的值
TCMPR0	TMR_BA01+004	R/W	Timer0 比较寄存器	0x0000_0000
TCMPR1	TMR_BA01+024	R/W	Timer1比较寄存器	0x0000_0000
TCMPR2	TMR_BA23+004	R/W	Timer2比较寄存器	0x0000_0000
TCMPR3	TMR_BA23+024	R/W	Timer3比较寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
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]							

Bits	描述	
[31:24]	保留	保留
[23:0]	**TCMP**	**定时器比较值** TCMP是24位比较寄存器. 当内部24位向上计数器的值与TCMP的值匹配时, 如果TCSR.IE[29]=1，就产生定时器中断请求. TCMP的值为定时器计数周期. 定时溢出周期= (定时器输入时钟周期) * (8-bit Prescale + 1) * (24-bit TCMP) **注1:** 不能在TCMP里写0x0或0x1，否则内核将运行到未知状态. **注2:** 如果定时器工作在连续计数模式，用户写一个新的值到TCMP，24位上数计数器将保持继续上数；如果计数器工作带其它模式，软件向该寄存器写入新的值，24-bit上数计数器将被复位并重新开始计数.

定时器中断状态寄存器(TISR)

寄存器	偏移量	R/W	描述	复位后的值
TISR0	TMR_BA01+08	R/W	Timer0 中断状态寄存器	0x0000_0000
TISR1	TMR_BA01+28	R/W	Timer1 中断状态寄存器	0x0000_0000
TISR2	TMR_BA23+08	R/W	Timer2 中断状态寄存器	0x0000_0000
TISR3	TMR_BA23+28	R/W	Timer3 中断状态寄存器	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							**TIF**

Bits	描述	
[31:1]	保留	保留
[0]	**TIF**	**定时器中断标志** 定时器中断状态位. 当内部24位计数器与TCMP的值匹配时，TIF由硬件置位，写1清该位.

定时器数据寄存器(TDR)

寄存器	偏移量	R/W	描述	复位后的值
TDR0	TMR_BA01+0C	R/W	Timer0数据寄存器	0x0000_0000
TDR1	TMR_BA01+2C	R/W	Timer1数据寄存器	0x0000_0000
TDR2	TMR_BA23+0C	R/W	Timer2数据寄存器	0x0000_0000
TDR3	TMR_BA23+2C	R/W	Timer3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**
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]							

Bits	描述	
[31:24]	保留	保留
[23:0]	**TDR**	定时器数据寄存器 1. CTB (TCSR[24]) = 0 : TDR为上数计数器的值，TCSR.TDR_EN 置1时，内部24位定时器的值可以从TDR读到，用户可以读取该寄存器的值获取24位计时器的值。 2. CTB (TCSR[24]) = 1 : TDR 为 24- bits 上数事件计数器的值. TCSR.TDR_EN 置1时，用户可以读TDR的值取得当前事件计数器的值

定时器捕捉数据寄存器 (TCAP)

寄存器	偏移	R/W	描述	复位后的值
TCAP0	TMR_BA01+0x10	R/W	Timer0 捕获数据寄存器	0x0000_0000
TCAP1	TMR_BA01+0x30	R/W	Timer1捕获数据寄存器	0x0000_0000
TCAP2	TMR_BA23+0x10	R/W	Timer2捕获数据寄存器r	0x0000_0000
TCAP3	TMR_BA23+0x30	R/W	Timer3捕获数据寄存器	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]							

Bits	描述	
[31:24]	保留	保留
[23:0]	TCAP	定时器捕获数据寄存器 当 TEXEN (TEXCON[3]) 为1, RSTCAPSEL (TTXCON[4]) 为 0时, TEX 引脚上的信号发生了TEX_EDGE(TEXCON[2:1])说明的转变时，内部 24-bit 上数值将被加载到 TCAP. 用户可以读这个寄存器得到计数器的值.

定时器外部控制寄存器 (TEXCON)

寄存器	偏移	R/W	描述	复位后的值
TEXCON0	TMR_BA01+0x14	R/W	Timer0 外部控制寄存器	0x0000_0000
TEXCON1	TMR_BA01+0x34	R/W	Timer1外部控制寄存器	0x0000_0000
TEXCON2	TMR_BA23+0x14	R/W	Timer2外部控制寄存器	0x0000_0000
TEXCON3	TMR_BA23+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

Bits	描述	
[31:8]	保留	保留
[7]	TCDB	定时器计数引脚防反弹使能 1 = 使能防反弹功能 0 = 关闭防反弹功能 如果这个bit 被使能, T0~T3 引脚的输入信号边沿将被防反弹电路检测.
[6]	TEXDB	定时器捕获引脚防反弹使能 1 =使能防反弹功能 0 = 关闭防反弹功能 如果这个bit 被使能, T0EX~T3EX 引脚的输入信号边沿将被防反弹电路检测
[5]	TEXIEN	定时器捕获引脚外部中断使能 1 = 使能定时器外部中断 0 = 禁止定时器外部中断 如果定时器外部中断被使能，当TEX引脚上发生TEX_EDGE(TEXCON[2:1])设定的信号转变时，中断将发生 例如, 当 TEXIEN = 1, TEXEN = 1, TEX_EDGE = 00时，TEX引脚上1 到 0的

Bits	描述	
		信号转变将导致TEXIF(TEXISR[0]) 中断标志被设，然后中断将发生.
[4]	**RSTCAPSEL**	**定时器外部复位计数器/捕获模式选择** 1 = TEX 上的信号转变被用作定时器计数器复位功能. 0 = TEX 上的信号转变被用作定时器捕获功能.
[3]	**TEXEN**	**定时器外部引脚使能.** 这个bit用来使能TEX引脚上的复位/捕获功能. 1 = TEX引脚上的信号转变将导致定时器计数器捕获或者复位. 0 = TEX 引脚被忽略.
[2:1]	**TEX_EDGE**	**定时器外部捕获引脚检测边沿设定** 00 = TEX引脚上1到 0 的转变将被检测. 01 = TEX引脚上0到 1 的转变将被检测. 10 = TEX引脚上1到 0和0到1 的转变都将被检测. 11 = 保留.
[0]	**TX_PHASE**	**定时器外部计数相位** 这个比特用来指示定时器外部计数功能计数的信号相位. 1 = 外部计数引脚上一个上升沿转变将被计数. 0 = 外部计数引脚上一个下降沿转变将被计数.

定时器外部中断状态寄存器 (TEXISR)

寄存器	偏移	R/W	描述	复位后的值
TEXISR0	TMR_BA01+0x18	R/W	Timer0 外部中断状态寄存器	0x0000_0000
TEXISR1	TMR_BA01+0x38	R/W	Timer1外部中断状态寄存器	0x0000_0000
TEXISR2	TMR_BA23+0x18	R/W	Timer2外部中断状态寄存器	0x0000_0000
TEXISR3	TMR_BA23+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

Bits	描述	
[31:1]	保留	保留
[0]	TEXIF	定时器外部中断标志 这个比特指示定时器的外部中断状态. 当 TEXEN (TEXCON[3]) 等于 1，TEX 引脚上发生 TEX_EDGE(TEXCON[2:1]) 设定的信号转变时，硬件将置位这个bit。这个bit写1清0. 例如， TEXEN = 1, TEX_EDGE = 00, TEX引脚上一个 1 到 0的转变将导致 TEXIF 被置.

6.9 看门狗定时器 (WDT)

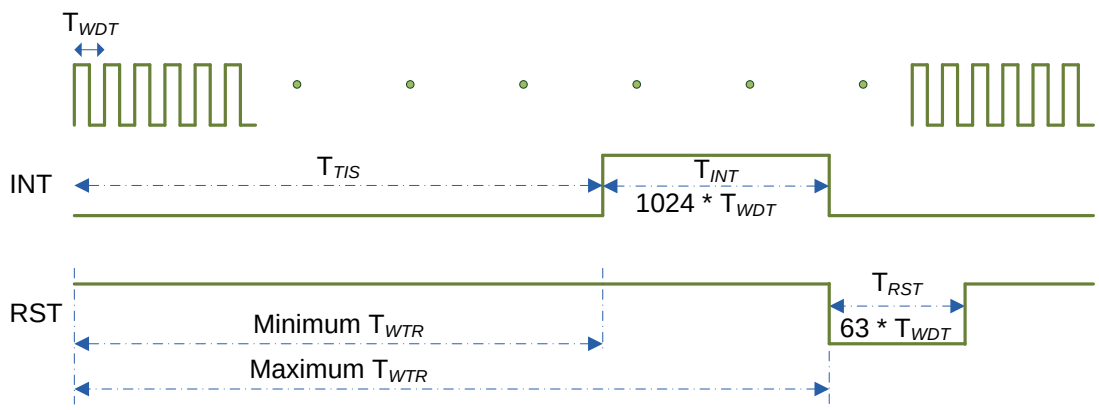
6.9.1 概述

看门狗定时器用于在软件运行至未知状态时执行系统复位功能，可以防止系统无限制地挂机，除此之外，看门狗定时器还可将CPU由掉电模式唤醒。看门狗定时器包含一个18位的自由运行的计数器，可编程其定时溢出间隔.

设置WTE(WDTCR[7])使能看门狗定时器，WDT计数器开始向上计数. 当计数器达到选择的定时溢出间隔，如果看门狗定时器中断使能位WTIE置位，看门狗定时器中断标志WTIF被立即置位，并请求WDT中断，同时，跟随在时间溢出事件之后有一个指定延时(1024 * T_{WDT})，用户必须在该延时时间结束前设置WTR(WDTCR[0]) (看门狗定时器复位)为高，重置18位WDT计数器，防止CPU复位。WTR在WDT计数重置后自动由硬件清零。通过设置WTIS(WDTCR[10:8])可选择8个带有指定延时的定时溢出间隔. 如果在特定延迟时间终止后，WDT计数没有被清零，看门狗定时器将置位看门狗定时器复位标志(WTRF)并使CPU复位. 这个复位将持续63个WDT时钟，然后CPU重启，并从复位向量(0x0000 0000)开始执行程序. 看门狗复位后WTRF位不会被清除。用户可用软件查询WTFR来识别复位源。WDT还提供唤醒功能。当芯片掉电，且看门狗唤醒使能位 (WDTR[4])置位时，如果WDT计数器达到由WTIS (WDTCR [10:8])定义的时间间隔时，芯片就会由掉电状态唤醒。第一个例子，如果WTIS被设置为000，CPU从掉电状态被唤醒的时间间隔是2^4 * T_{WDT}。当掉电命令被软件设置，CPU进入掉电状态，在2^4 * T_{WDT}时间过后，CPU由掉电状态唤醒。第二个例子，如果WTIS被设置为111，CPU从掉电状态被唤醒的时间间隔是2^{18} * T_{WDT}。当掉电命令被软件设置，CPU进入掉电状态，在2^{18} * T_{WDT}时间过后，CPU由掉电状态唤醒。注意，如果WTRE (WDTCR [1])被置位，再CPU被唤醒之后，软件应当尽可能快的通过置位WTR(WDTCR [0])来清零看门狗定时器计数器，否则，如果在从CPU唤醒到软件清零看门狗定时器计数器的时间超过1024 * T_{WDT}，看门狗定时器计数器没有通过置位WTR(WDTCR [0])被清零，看门狗定时器将复位CPU。

WTIS	Timeout Interval Selection T_{TIS}	Interrupt Period T_{INT}	WTR Timeout Interval (WDT_CLK=10 kHz) MIN. T_{WTR} ~ MAX. T_{WTR}
000	2^4 * T_{WDT}	1024 * T_{WDT}	1.6 ms ~ 104 ms
001	2^6 * T_{WDT}	1024 * T_{WDT}	6.4 ms ~ 108.8 ms
010	2^8 * T_{WDT}	1024 * T_{WDT}	25.6 ms ~ 128 ms
011	2^{10} * T_{WDT}	1024 * T_{WDT}	102.4 ms ~ 204.8 ms
100	2^{12} * T_{WDT}	1024 * T_{WDT}	409.6 ms ~ 512 ms
101	2^{14} * T_{WDT}	1024 * T_{WDT}	1.6384 s ~ 1.7408 s
110	2^{16} * T_{WDT}	1024 * T_{WDT}	6.5536 s ~ 6.656 s
111	2^{18} * T_{WDT}	1024 * T_{WDT}	26.2144 s ~ 26.3168 s

表6.9-1看门狗超时间隔选择



- T_{WDT} : *Watchdog Engine Clock Time Period*
- T_{TIS} : *Watchdog Timeout Interval Selection Period*
- T_{INT} : *Watchdog Interrupt Period*
- T_{RST} : *Watchdog Reset Period*
- T_{WTR} : *Watchdog Timeout Interval Period*

图 6.9.1-1 中断时序与复位信号时序

6.9.2 特征

- 18-位自由运行的计数器以防止CPU在延迟时间结束之前发生看门狗定时器复位。
- 超时间隔可选(2^4 ~ 2^18)，超时时间范围在104 ms ~ 26.3168 s (如果WDT_CLK = 10 KHz).
- 复位周期 = (1 / 10 kHz) * 63, 如果WDT_CLK = 10 KHz.

6.9.3 WDT 框图

看门狗定时器时钟控制和框图如图6.9.3-1所示。

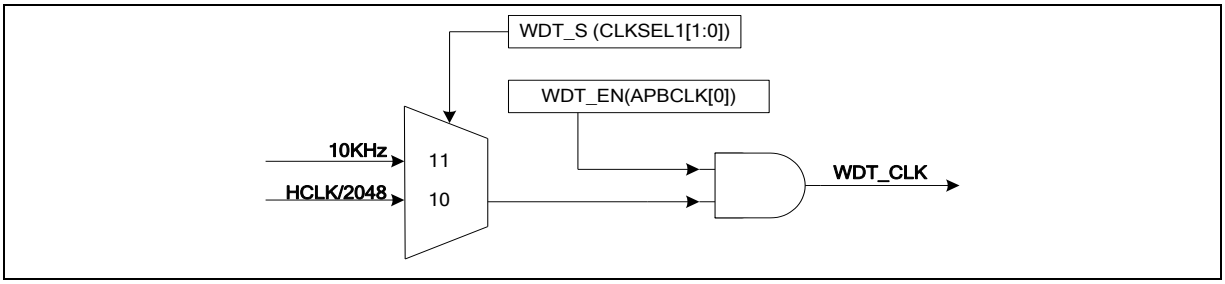


图 6.9.3-1 看门狗定时器时钟控制

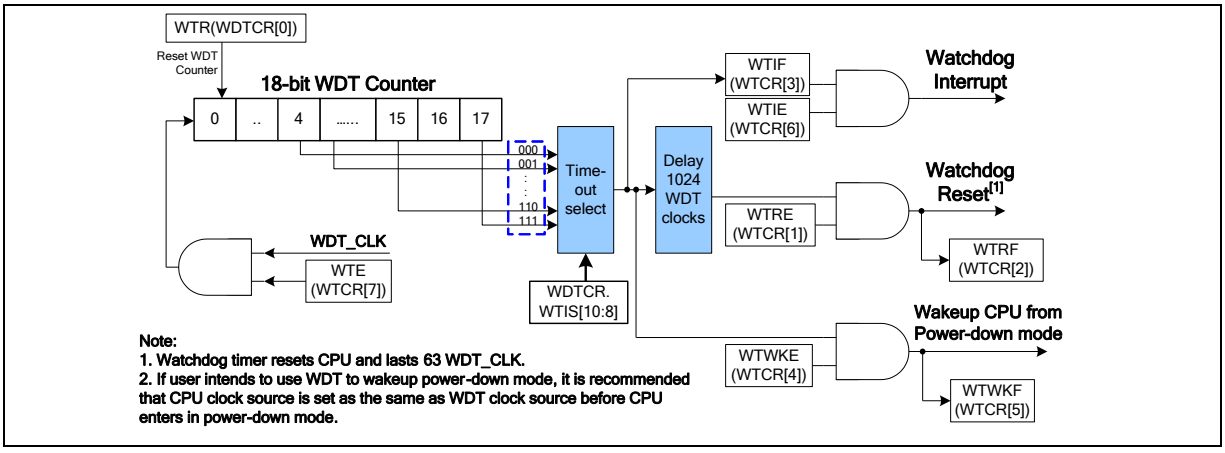


图 6.9.3-2 看门狗定时器框图

6.9.4 看门狗定时器控制寄存器映射

R: 只读, **W:** 只写, **R/W:** 可读写

寄存器	偏移量	R/W	描述	复位后的值
WDT_BA = 0x4000_4000				
WTCR	WDT_BA+00	R/W	看门狗定时器控制寄存器	0x0000_0700

看门狗定时器控制寄存器r **(WTCR)**

寄存器	偏移量	R/W	描述	复位后的值
WTCR	WDT_BA+000	R/W	看门狗定时器控制寄存器	0x0000_0700

注:

该寄存器所有位都写保护. 要编程时，需要开锁时序，依次向寄存器REGWRPROT写入**"59h"**, **"16h"**, 与 **"88h"** ，**RegLockAddr** 的地址为**GCR_BA + 0x100**

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**

Bits	描述	
[31]	DBGACK_WDT	**ICE调试模式应答禁止 (写保护)** 0 = ICE 调试模式应答影响看门狗定时器计数. 当ICE调试模式应答时，看门狗定时器计数器的值将维持不变 1 = ICE 调试模式应答禁止. 无论ICE调试模式应答与否，看门狗定时器计数器将继续正常计数。
[30:11]	保留	保留

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td>[10:8]</td><td>WTIS</td><td>
看门狗定时器间隔选择
选择看门狗定时器的定时溢出间隔.
<table>
<tr><th>WTIS</th><th>超时间隔选择</th><th>中断周期</th><th>WTR 超时间隔 (WDT_CLK=10 kHz)</th></tr>
<tr><td>000</td><td>2^4 * T_{WDT}</td><td>1024 * T_{WDT}</td><td>1.6 ms ~ 104 ms</td></tr>
<tr><td>001</td><td>2^6 * T_{WDT}</td><td>1024 * T_{WDT}</td><td>6.4 ms ~ 108.8 ms</td></tr>
<tr><td>010</td><td>2^8 * T_{WDT}</td><td>1024 * T_{WDT}</td><td>25.6 ms ~ 128 ms</td></tr>
<tr><td>011</td><td>2^{10} * T_{WDT}</td><td>1024 * T_{WDT}</td><td>102.4 ms ~ 204.8 ms</td></tr>
<tr><td>100</td><td>2^{12} * T_{WDT}</td><td>1024 * T_{WDT}</td><td>409.6 ms ~ 512 ms</td></tr>
<tr><td>101</td><td>2^{14} * T_{WDT}</td><td>1024 * T_{WDT}</td><td>1.6384 s ~ 1.7408 s</td></tr>
<tr><td>110</td><td>2^{16} * T_{WDT}</td><td>1024 * T_{WDT}</td><td>6.5536 s ~ 6.656 s</td></tr>
<tr><td>111</td><td>2^{18} * T_{WDT}</td><td>1024 * T_{WDT}</td><td>26.2144 s ~ 26.3168 s</td></tr>
</table>
</td></tr>
<tr><td>[7]</td><td>WTE</td><td>
使能看门狗定时器
0 = 禁用看门狗定时器功能(该动作复位内部计数器)
1 = 使能看门狗定时器
</td></tr>
<tr><td>[6]</td><td>WTIE</td><td>
看门狗定时器中断使能
0 = 禁用看门狗定时器中断
1 =使能看门狗定时器中断
</td></tr>
<tr><td>[5]</td><td>WTWKF</td><td>
看门狗定时器唤醒标志
如果看门狗定时器引起CPU从掉电模式下唤醒，该位将被置高。
0 = 看门狗定时器不能引起CPU唤醒.
1 = CPU 由休眠或掉电模式被看门狗定时溢出唤醒
注：写1清零.
</td></tr>
<tr><td>[4]</td><td>WTWKE</td><td>
看门狗定时器唤醒功能使能位
0 = 禁用看门狗唤醒CPU功能。
1 =使能看门狗唤醒CPU功能.
</td></tr>
<tr><td>[3]</td><td>WTIF</td><td>
看门狗定时器中断标志
如果看门狗定时器中断使能，该位由硬件置位表示看门狗定时器中断已发生。
0= 没有发生看门狗定时器中断
1= 发生看门狗定时器中断
</td></tr>
</table>

Bits	描述	
		注: 写1清零.
[2]	**WTRF**	**看门狗定时器复位标志** 当看门狗定时器引发复位，该位被置位，通过读取该位可以确认复位是否由看门狗引起。该位软件写1清零。如果**WTRE**禁用，看门狗定时器溢出对该位无影响。 0 = 复位不是由看门狗定时器产生。 1 = 看门狗定时器引发复位 **注**: 写1清零.
[1]	**WTRE**	**看门狗定时器复位使能** 设定该位使能看门狗定时器复位功能。 0 = 禁用看门狗定时器复位功能 1 = 使能看门狗定时器复位功能
[0]	**WTR**	**复位看门狗定时器的计数器（写保护）** 设置该位将复位看门狗定时器的计数器. 0: 写0无效 1: 复位看门狗定时器的值 **NOTE**: 写1清零.

6.10 UART 接口控制器

NuMicro M051™ 提供2个通用异步收/发器（UART）通道，UART0～1支持普通速度UART，并支持流控制.

6.10.1 概述

通用异步收/发器(UART) 对从外设收到的数据执行串到并的转换，对从CPU发送的数据执行并到串的转换。该串口同时支持IrDA SIR 功能和RS-485模式. 每个UART通道有5种类型的中断, 它们是, 发送FIFO空中断(Int_THRE), 接收阀值到达中断(Int_RDA), 线状态中断 (奇偶校验错误或者帧错误或者break中断)(Int_RLS) , 接收缓冲器超时中断(Int_Tout), 调制解调器/唤醒状态中断(Int_Modem)，缓冲错误中断(INT_BUF_ERR) 和LIN 接收到 break 域中断 (INT_LIN_RX_BREAK)。UART0的中断号为12（中断向量为28），UART1的中断号为13（中断向量29），参考嵌套向量中断控制器.

UART0~1内嵌一个16字节发送FIFO (TX_FIFO) 和一个16字节接收 FIFO (RX_FIFO)，可以降低CPU的中断次数。 CPU可以随时读UART的状态。返回的状态信息包括正在被UART执行的传输操作的类型和条件, 在接收数据时还可能发生3个错误(奇偶校验错误、帧错误、break中断)状况。UART包括一个可编程的波特率发生器，它可以将输入时钟分频来得到收发器需要的时钟. 波特率公式是 Baud Rate = UART_CLK / M * [BRD + 2]. 其中M和BRD在波特率分频寄存器UA_BAUD中定义. 表6.10-1列出了不同条件下波特率方程，表6.10-2列出了UART波特率设置表。

Mode	DIV_X_EN	DIV_X_ONE	Divider X	BRD	波特率公式
0	0	0	B	A	UART_CLK / [16 * (A+2)]
1	1	0	B	A	UART_CLK / [(B+1) * (A+2)] , B must >= 8
2	1	1	Don’t care	A	UART_CLK / (A+2), A must >=3

表 6.10-1 UART 波特率方程

系统时钟 = 22.1184 MHz			
波特率	模式0	模式1	模式2
921600	x	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

系统时钟 = 22.1184 MHz			
波特率	模式0	模式1	模式2
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波特率设置表

UART0与UART1 控制器支持自动流控制功能，它使用 2 种低电平信号, /CTS (clear-to-send，允许发送)和 /RTS (request-to-send，请求发送)，来控制UART 和外部驱动器(ex: Modem)之间的数据流传递. 当自动流控功能使能时，UART被禁止接收数据直到UART 向外发出/RTS信号. 当Rx FIFO中字节数量和RTS_TRI_LEV (UA_FCR [19:16])的值相等时, /RTS信号变成无效. 当UART控制器从外部驱动器侦测到 /CTS有效时，UART 向外发送数据 如果 /CTS 无效, UART 将不向外发送数据.

UART 控制器提供 串行 IrDA (SIR, 串行红外) 功能 (用户需置位rDA_EN (UA_FUN_SEL[1:0])使能 IrDA 功能). SIR 规范定义短程红外异步串行传输模式为1 bit开始位, 8 bit数据位, 和1 bit停止位. 最大数据速率为 115.2 Kbps (半双工). IrDA SIR模块包括一个IrDA SIR协议编码/解码器. IrDA SIR 只是半双工协议. 因此不能同时发送和接收数据. IrDA SIR 物理层规定在发送和接收之间至少要有10ms传输延时. 该特性必须由软件实现.

UART控制器的另一功能是支持RS-485 9位模式, 由RTS控制方向或通过软件编程GPIO (P0.3 对应于 RTS0 and P0.1 对应于 RTS 1) 执行该功能. RS-485模式通过设置UA_FUN_SEL寄存器选定。 使用来自异步串行口的RTS控制信号来使能RS-485驱动器，执行RS-485驱动器控制。在RS-485模式下，RX与TX的许多特性与UART相同。

6.10.2 特性

- 全双工，异步通信
- 独立的接收/发送16字节 (UART0/UART1) FIFO数据装载区
- 支持硬件自动流控制/流控制功能(CTS, RTS)和可编程的RTS流控制触发电平(UART0 与UART1 支持)
- 可编程的接收缓冲触发级别
- 每个通道都支持独立的可编程的波特率发生器
- 支持CTS 唤醒功能
- 支持8位接收缓冲超时功能
- 通过设置UA_TOR [DLY] 可以编程在上一个停止与下一个开始位之间数据发送的延迟时间
- 支持break错误，帧错误，奇偶校验错误和接收/发送缓冲溢出检测功能
- 完全可编程的串行接口特性
  - 可编程的数据位, 5, 6, 7, 8位
  - 可编程的奇偶校验位, 偶校验、奇校验、无校验位或stick校验位 发生和检测
  - 可编程停止位, 1, 1.5, 或 2 停止位 产生
- 支持IrDA SIR 功能
  - 普通模式下支持 3/16位持续时间
- 支持LIN功能
  - 支持LIN主/从模式
  - 支持发送端可编程的break产生功能
  - 支持接收端break检测功能
- 支持RS-485 模式.
  - 支持 RS-485 9位模式
  - 支持硬件或软件编程RTS引脚控制收发器的传输方向

6.10.3 UART 框图

UART 时钟控制和框图如图6.10.3-1和 图6.10.3-2.

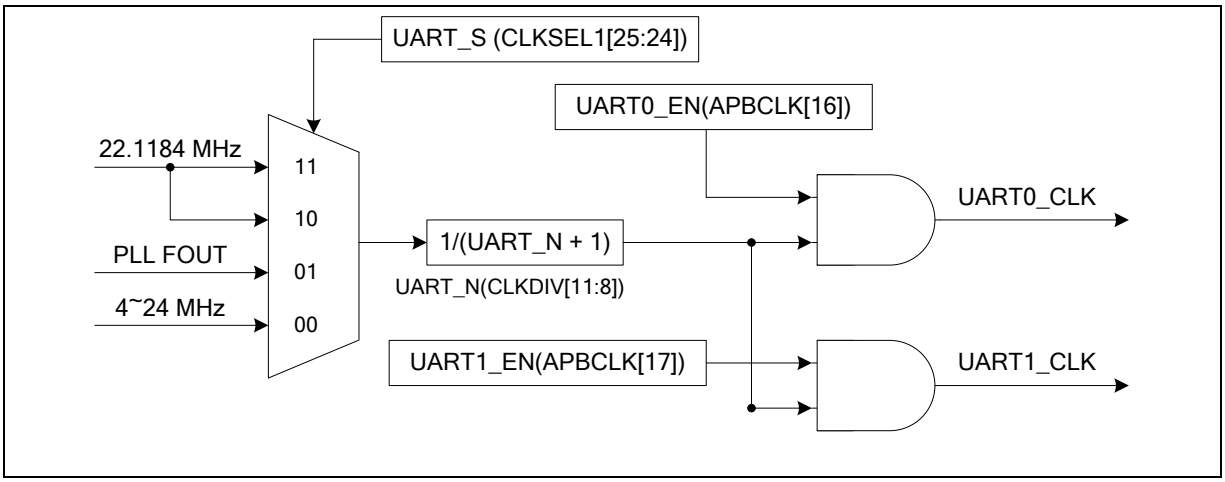


图 6.10.3-1 UART 时钟控制框图

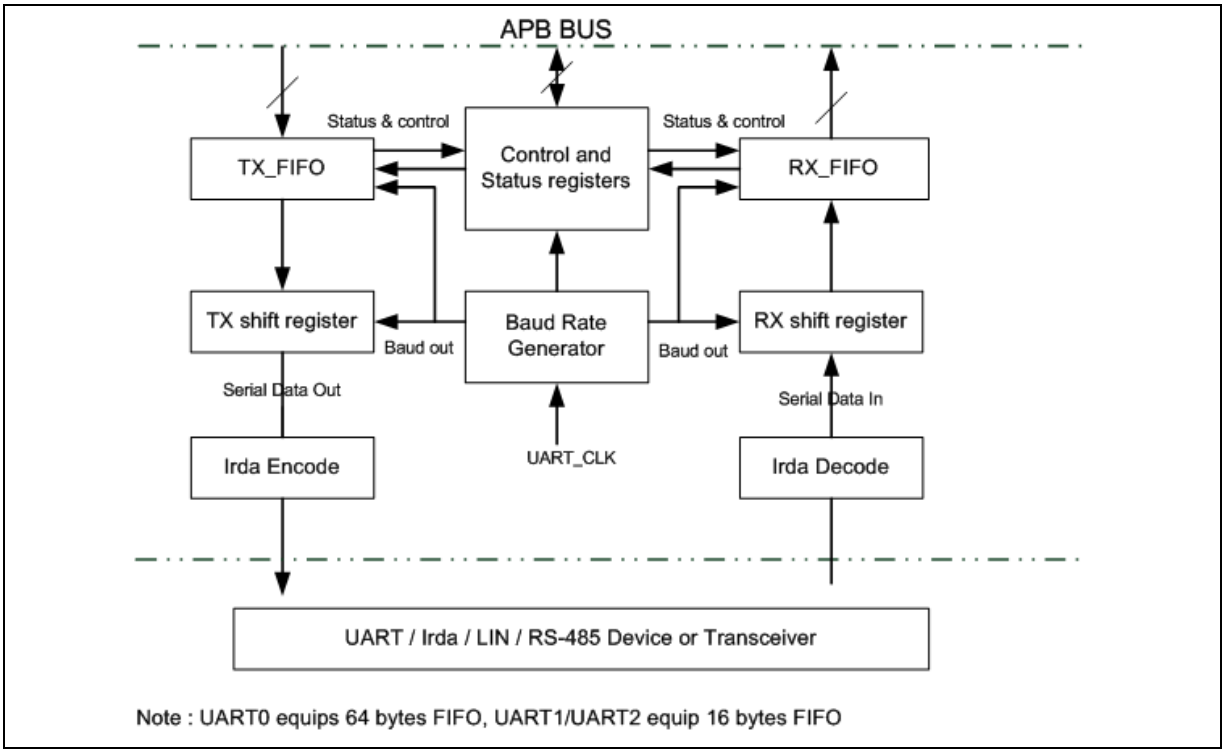


图 6.10.3-2 UART 框图

每个方块的细节如下所述：

TX_FIFO
发送用一个16字节的FIFO做缓存来降低CPU的中断数量.

RX_FIFO
接收用一个16字节(每个字节加3个比特的错误比特)的FIFO做缓存来降低CPU的中断数量.

TX移位寄存器
此模块控制移动正在发送的数据串行输出.

RX移位寄存器
此模块控制移动正在接收的数据串行输入.

Modem控制寄存器
该寄存器控制与MODEM 或者数据传输转换器(或者一个MODEM模拟器)的接口.

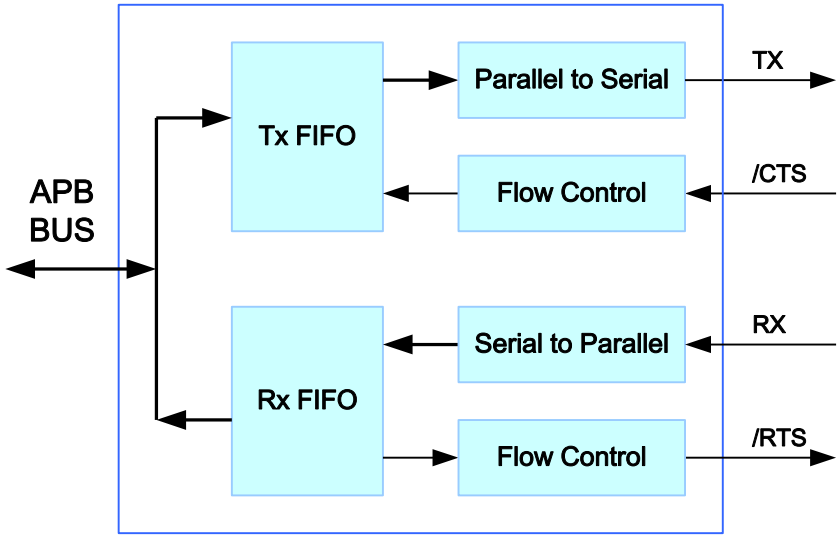
波特率发生器
将外部时钟除以一个除数来获得需要的波特率时钟,参考波特率方程.

IrDA 编码
IrDA 编码控制模块.

IrDA 解码
IrDA 解码控制模块.

控制和状态寄存器
该域是用于发送器和接收器的寄存器组, 包括 FIFO 控制寄存器 (UA_FCR), FIFO 状态寄存器 (UA_FSR), 和线控制寄存器 (UA_LCR)。超时控制寄存器 (UA_TOR) 应用于标识超时中断产生的条件. 该寄存器组还包括中断控制使能寄存器 (UA_IER) 和中断状态寄存器 (UA_ISR) 来使能或者禁用中断响应并且识别发生的中断. 有7种中断: FIFO为空中断（INT_THRE），接收阀值到达中断（INT_RDA），线状态中断 (校验错误，帧错误和break中断)（INT_RLS）, 超时中断（INT_Tout），MODEM/唤醒状态中断（INT_Modem）和缓冲错误中断（INT_Buf_Err）.

图6.10.3-3为自动流控制框图.



Note: Only supported in UART0 and UART1

图 6.10.3-3 自动流控制框图

6.10.4 IrDA 模式

UART 支持 IrDA SIR (串行红外) 发送编码 和接收解码, IrDA 模式通过设定 **UA_FUN_SEL** 寄存器的 **IrDA_EN** 位为'10'来选择.

IrDA 模式下, UA_BAUD[DIV_X_EN] 位需禁用.

波特率 = Clock / (16 * BRD), BRD 为UA_BAUD 寄存器中的波特率分频器.

下图为IrDA 控制框图.

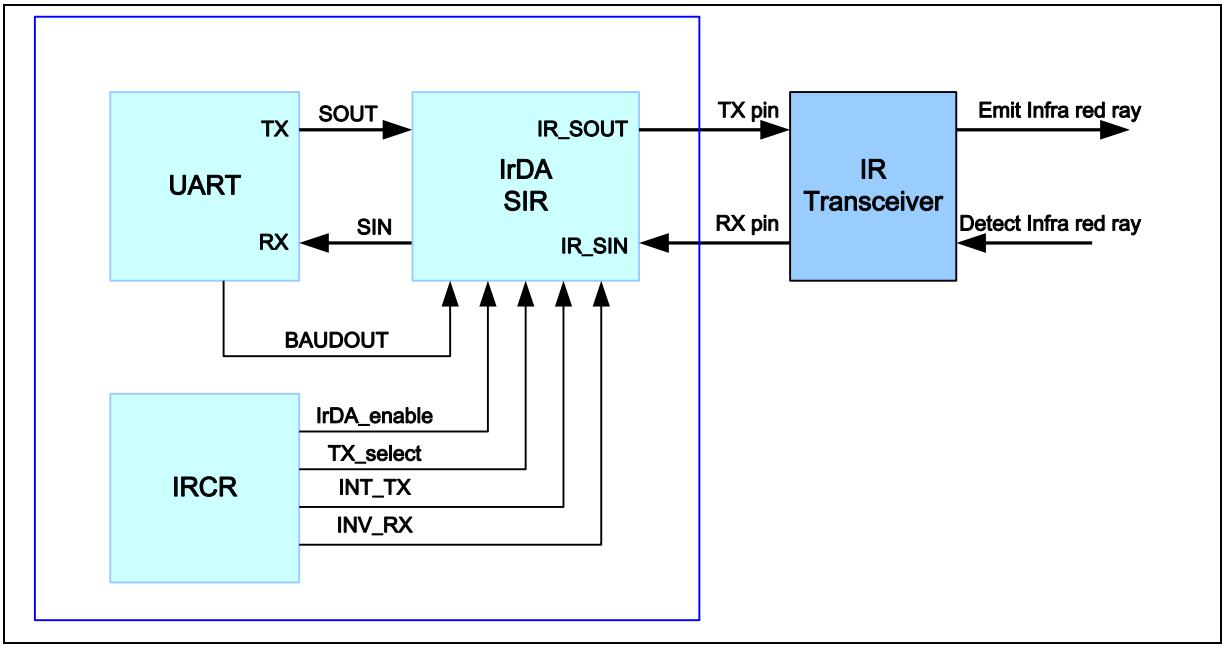


图 6.10.4-1 IrDA 框图

6.10.4.1 IrDA SIR发送编码器

IrDA SIR发送编码器以非归零(NRZ) 调制方式从 UART 输出比特流。. IrDA SIR 物理层指定使用归零, 反向 (RZI) 调制方式，代表逻辑0作为红外光脉冲。被调制的输出脉冲流被发送到外部输出驱动器和红外发射二极管

在正常模式下, 传输脉冲的宽度为 3/16 波特率周期.

6.10.4.2 IrDA SIR接收解码器

IrDA SIR 接收解码器以归零return-to-zero方式解调由输入探测器输入的比特流，并输出 NRZ比特流到 UART 作为数据输入. 解码器在空闲模式输入通常为高。 (因为IRCR bit 6 默认设定为高)

当解码器输入低时，表示一个起始位。

6.10.4.3 IrDA SIR操作

IrDA SIR 编码/解码器提供UART数据流和半双工串行SIR接口间互相转换的功能。. 图6.10.4-2是IrDA编码/解码器波形图

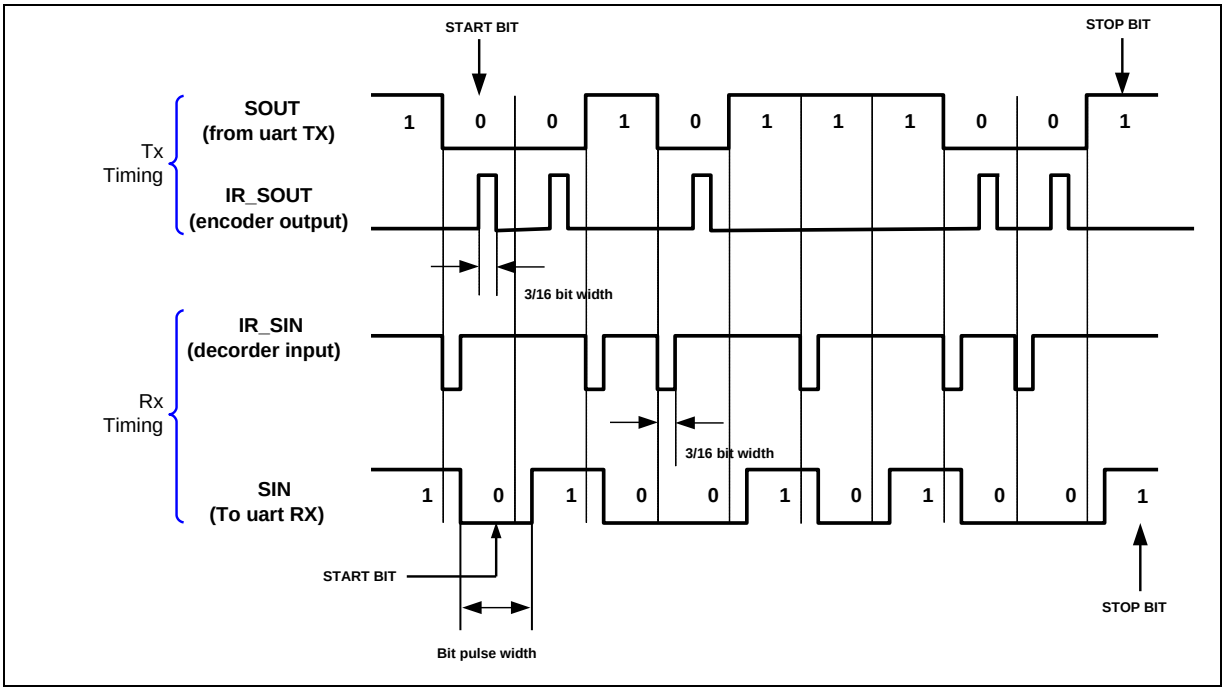


图 6.10.4-2 IrDA TX/RX 时序框图

6.10.5 LIN (Local Interconnection Network) 模式

UART 支持 LIN 功能, 可以通过设定**UA_FUN_SEL[1:0]** 为 '01'来选择LIN功能. LIN 模式下，根据LIN的标准，每个字节包含1个值为0的start bit (dominant), 后面是8 个 bits 的数据(LSB is first)，最后以1个值为1的stop bit (recessive) 结束. 下图是LIN的帧结构:

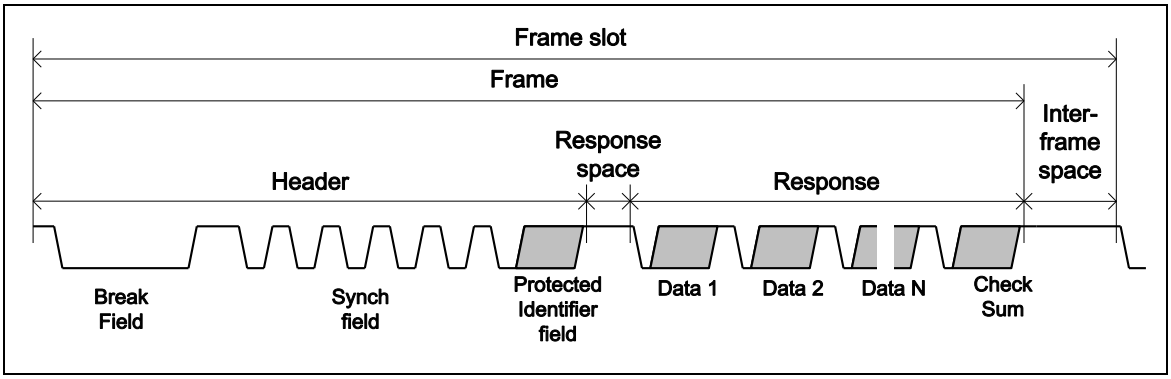


图 6.10.5-1 LIN 帧结构

LIN总线发送(TX)编程流程如下

1. 设定**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 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，请求Identifier 域被发送
5. 设定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 将被自动清为0).
6. 当THR中字节的 STOP bit 被发送到总线上, 硬件将设定UA_FSR中的 TE_FLAG 为 1.
7. 填I 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. 设定 **UA_FUN_SEL[1:0]** 为 '01' 使能LIN 总线模式.
2. 设定UA_LIN_BCNT寄存器的LIN_RX_EN 来使能 LIN RX 模式.
3. 等待UA_ISR 中的LIN_RX_BREAK_IF 标志被置，接收到Break 域.
4. 等待UA_ISR中的 RDA_IF 标志被置，从 UR_RBR 寄存器读回收到的数据.

6.10.6 RS-485 模式

UART 支持RS-485 9位模式. RS-485模式可以通过设置UA_FUN_SEL寄存器来选择RS-485功能。使用来自异步串行口的RTS控制信号来使能RS-485驱动器，执行RS-485驱动器控制。在RS-485模式下，RX与TX的许多特性与UART相同。

RS-485 模式下，控制器可以配置成可寻址的RS-485从机，RS-485主机发送器将通过设置校验位 (第9位)为1来识别一个地址字符。对于数据字符，校验位设置为0. 软件可通过设置寄存器UA_LCR 控制第9位 (**PBE , EPE 和 SPE置位, 第9位发送0 , PBE 和 SPE 置位，EPE清零, 第9位发送1**). 该控制器支持三种操作模式：RS-485 普通多点模式(NMM), RS-485 自动地址识别模式 (AAD) 和RS-485 自动方向控制模式(AUD), 可通过编程UA_RS-485_CSR寄存器选择任何工作模式, 通过设置UA_TOR [DLY] 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 普通多点模式 (NMM)

RS-485 普通多点模式, 首先必须软件配置在地址位之前检测到的数据是否存储于RX-FIFO中。如果软件想忽略地址位之前检测到的所有数据，则流程为设置UART_FCR[RS485_RX_DIS]和使能UA_RS-485[RS485_NMM]，这样，接收器忽略所有数据直至检测到地址字节（bit9=1）并将地址字节数据存储于RX-FIFO中。如果软件想接收检测到地址位之前的所有数据，则流程为禁止UART_FCR [RS485_RX_DIS]和使能UA_RS-485[RS485_NMM]，这样，接收器就会接收所有数据。如果检测到地址位，RS-485控制器会向CPU产生一个中断，软件可通过设定UA_RS-485_FCR [RX_DIS]来决定是否使能接收器来接收接下来的数据字节。如果使能接收器接收，所有接收的数据都将被接收并存储于RX-FIFO中，如果禁止接收数据，则接收到的所有数据都将被忽略直至下一个地址位被检测到。若软件设置UA_RS-485_CSR [RX_DIS]关闭接收功能，当检测到下一个地址字节时，控制器会自动清UA_RS-485_FCR [RX_DIS]位（就是说，自动打开接收功能），并将地址字节数据存储到RX-FIFO.

RS-485 自动地址识别模式 (AAD)

RS-485自动地址识别模式下, 接收器在检测到地址字节（bit9=1）并且地址字节数据与UA_ALT_CSR [ADDR_MATCH]的值相匹配之前，忽略所有数据. 然后地址字节数据将被存储在RX-FIFO. 其后所有接收到的字节数据将被接收，并存储于RX-FIFO 直到收到的地址字节不匹配UA_ALT_CSR [ADDR_MATCH] 的值为止.

RS-485 自动方向模式 (AUD)

RS-485控制器的另一个功能是自动方向控制. 使用来自异步串行口的RTS控制信号来使能RS-485驱动器，执行RS-485驱动器控制. RTS线被连接到RS-485驱动器使能，以便设置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.

编程流程示例:

1. 设置寄存器UA_FUN_SEL中的FUN_SEL位选择RS-485功能.

2. 设置寄存器UA_FCR 中的RX_DIS 位使能或禁止RS-485 接收数据

3. 设置RS-485_NMM 或 RS-485_AAD 模式.

4. 如果选择RS-485_AAD 模式, ADDR_MATCH设置成自动地址匹配值.

5. 设置RS-485_AUD选择自动方向控制.

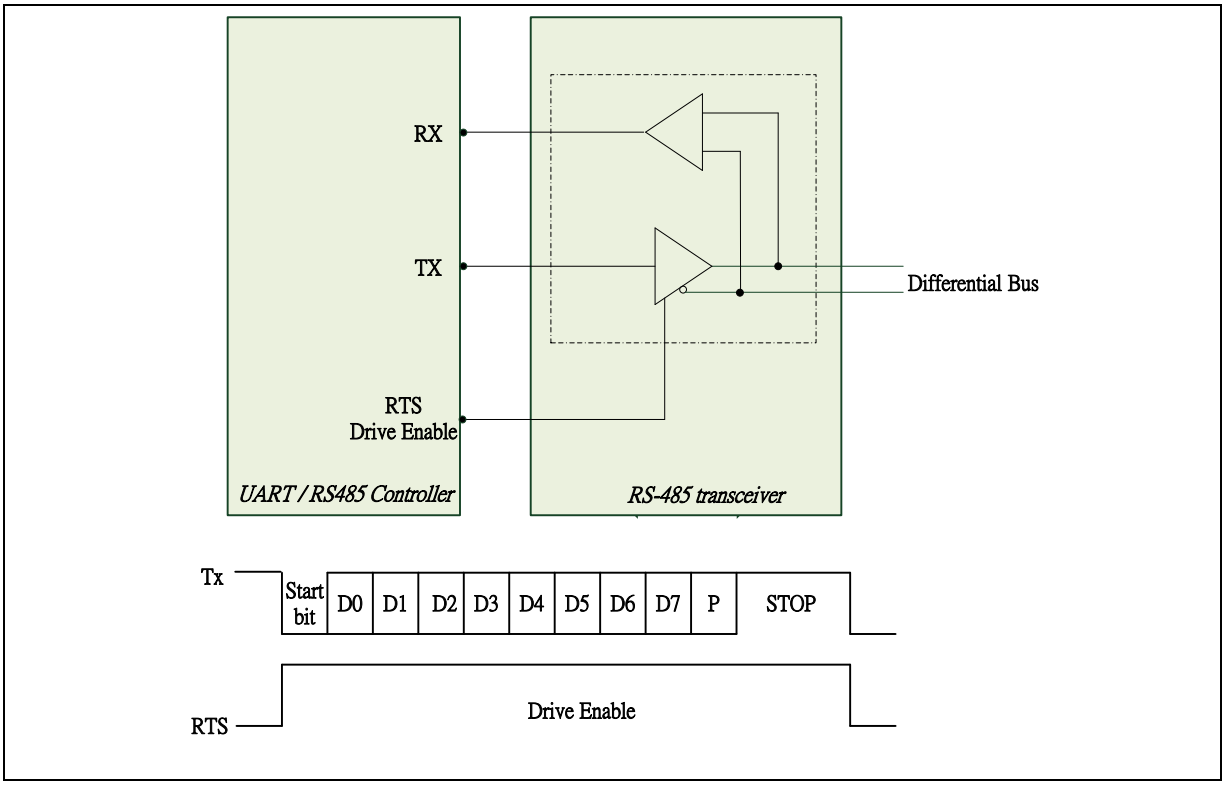


图 6.10.6-1 RS-485 帧结构

6.10.7 UART 接口控制寄存器映射

R:只读, **W**: 只写, **R/W**: 可读写

寄存器	偏移量	R/W	描述	复位后的值
UART 基地址: **Channel0 : UART0_BA = 0x4005_0000** **Channel1 : UART1_BA = 0x4015_0000**				
UA_RBR	UART0_BA+0x00	R	UART0接收数据缓存寄存器.	Undefined
	UART1_BA+0x00	R	UART1接收数据缓存寄存器.	Undefined
UA_THR	UART0_BA+0x00	W	UART0发送保持寄存器.	Undefined
	UART1_BA+0x00	W	UART1发送保持寄存器.	Undefined
UA_IER	UART0_BA+0x04	R/W	UART0中断使能寄存器.	0x0000_0000
	UART1_BA+0x04	R/W	UART1中断使能寄存器.	0x0000_0000
UA_FCR	UART0_BA+0x08	R/W	UART0 FIFO 控制寄存器.	0x0000_0000
	UART1_BA+0x08	R/W	UART1 FIFO 控制寄存器.	0x0000_0000
UA_LCR	UART0_BA+0x0C	R/W	UART0 Line 控制寄存器.	0x0000_0000
	UART1_BA+0x0C	R/W	UART1 Line 控制寄存器.	0x0000_0000
UA_MCR	UART0_BA+0x10	R/W	UART0 Modem 控制寄存器.	0x0000_0000
	UART1_BA+0x10	R/W	UART1 Modem 控制寄存器.	0x0000_0000
UA_MSR	UART0_BA+0x14	R/W	UART0 Modem 状态寄存器.	0x0000_0000
	UART1_BA+0x14	R/W	UART1 Modem 状态寄存器.	0x0000_0000
UA_FSR	UART0_BA+0x18	R/W	UART0 FIFO 状态寄存器.	0x1040_4000
	UART1_BA+0x18	R/W	UART1 FIFO 状态寄存器.	0x1040_4000
UA_ISR	UART0_BA+0x1C	R/W	UART0 中断 状态寄存器.	0x0000_0002
	UART1_BA+0x1C	R/W	UART1 中断 状态寄存器.	0x0000_0002
UA_TOR	UART0_BA+0x20	R/W	UART0 超时寄存器	0x0000_0000
	UART1_BA+0x20	R/W	UART1 超时寄存器	0x0000_0000
UA_BAUD	UART0_BA+0x24	R/W	UART0 波特率分频寄存器	0x0F00_0000
	UART1_BA+0x24	R/W	UART1 波特率分频寄存器	0x0F00_0000

寄存器	偏移量	R/W	描述	复位后的值
UA_IRCR	UART0_BA+0x28	R/W	UART0 IrDA 控制寄存器.	0x0000_0040
	UART1_BA+0x28	R/W	UART1 IrDA 控制寄存器.	0x0000_0040
UA_ALT_CSR	UART0_BA+0x2C	R/W	UART0 控制/状态寄存器	0x0000_0000
	UART1_BA+0x2C	R/W	UART1控制/状态寄存器	0x0000_0000
UA_FUN_SEL	UART0_BA+0x30	R/W	UART0 功能选择寄存器	0x0000_0000
	UART1_BA+0x30	R/W	UART1功能选择寄存器	0x0000_0000

6.10.8 UART 接口控制寄存器描述

接收缓冲寄存器(UA_RBR)

寄存器	偏移量	R/W	描述	复位后的值
UA_RBR	UART0_BA+0x00	R	UART0接收缓冲寄存器	Undefined
	UART1_BA+0x00	R	UART1接收缓冲寄存器	Undefined

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							

Bits	描述	
[31:8]	保留	保留
[7:0]	**RBR**	接收缓冲寄存器（只读） 通过读此寄存器, UART 将返回一个从 Rx引脚接收到的 8-位数据 (LSB优先).

发送保持寄存器(UA_THR)

寄存器	偏移量	R/W	描述	复位后的值
UA_THR	UART0_BA+0x00	W	UART0发送保持寄存器	Undefined
	UART1_BA+0x00	W	UART1发送保持寄存器	Undefined

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							

Bits	描述	
[31:8]	保留	保留
[7:0]	THR	发送保持寄存器 通过写该寄存器, UART 将通过Tx引脚 (LSB优先) 发送 8-位数据.

中断使能寄存器(UA_IER)

寄存器	偏移量	R/W	描述	复位后的值
UA_IER	UART0_BA+0x04	R/W	UART0中断使能寄存器	0x0000_0000
	UART1_BA+0x04	R/W	UART1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

Bits	描述	
[31:14]	保留	保留
[13]	**AUTO_CTS_EN**	**CTS 自动流控制使能** 1 = 使能 CTS 自动流控制. 0 = 禁用 CTS 自动流控制. 当 CTS 自动流控制使能, 当CTS输入有效时UART将向外部驱动器发送数据 (直到CTS 有效UART 才会开始机发送数据).
[12]	**AUTO_RTS_EN**	**RTS 自动流控制使能** 1 = 使能 RTS 自动流控制. 0 = 禁用 RTS自动流控制. 当 RTS 自动流控使能时, 如果Rx FIFO中接收的字节数和 UA_FCR[RTS_Tri_Lev]相等, UART 将使RTS信号失效
[11]	**TIME_OUT_EN**	**超时计数器使能** 1 = 使能超时计数器. 0 = 禁用超时计数器.
[10:9]	保留	保留
[8]	**LIN_RX_BRK_IEN**	**LIN 接收到 Break 域检测中断使能**

Bits	描述	
		1 = 使能 Lin 总线接收到break 域中断 0 = 禁止 Lin 总线接收break 域中断 **注意:** 这个域只是给LIN功能使用.
[7]	**保留**	保留
[6]	**WAKE_EN**	**唤醒 CPU 功能使能** 0 = 禁用 UART 唤醒 CPU 功能 1 = 使能唤醒功能, 当系统在深度睡眠模式下, 外部 /CTS 信号改变将从深度睡眠模式下唤醒CPU.
[5]	**BUF_ERR_IEN**	**缓冲错误中断使能** 1 = 使能 INT_BUF_ERR 0 = 禁止 INT_BUF_ERR
[4]	**RTO_IEN**	**接收超时中断使能** 0 = 禁用INT_tout中断 1 = 使能INT_tout 中断
[3]	**MODEM_IEN**	**调制解调器中断状态使能** 0 = 禁用off INT_MOS中断 1 = 使能INT_MOS中断
[2]	**RLS_IEN**	**接收线上状态中断使能** 0 = 禁用INT_RLS中断 1 = 使能INT_RLS中断
[1]	**THRE_IEN**	**发送保持寄存器空中断使能** 0 = 禁用 INT_THRE中断 1 = 使能INT_THRE中断
[0]	**RDA_IEN**	**接收数据可得中断使能** . 0 = 禁用INT_RDA中断 1 = 使能INT_RDA中断

FIFO 控制寄存器 (UA_FCR)

寄存器	偏移量	R/W	描述	复位后的值
UA_FCR	UART0_BA+0x08	R/W	UART0 FIFO控制寄存器	0x0000_0000
	UART1_BA+0x08	R/W	UART1 FIFO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				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

Bits	描述	
[31:20]	保留	保留
[19:16]	RTS_TRI_LEV	自动流控下RTS触发级别 接收FIFO中的字节数 RTS_TRI_LEV / 触发级别(Bytes) 0000 / 01 0001 / 04 0010 / 08 0011 / 14 **注:** 该寄存器用于自动RTS流控制..
[15:9]	保留	保留
[8]	RX_DIS	关闭接收寄存器. 接收器禁用或使能(置1禁止接收) 1: 禁用接收器 0: 使能接收器 **注:** 该位用于RS-485 普通模式. 必须在设置UA_ALT_CSR [RS-485_NMM]之前被

Bits	描述	
		设置好.
[7:4]	**RFITL**	**RX FIFO 中断 (INT_RDA)触发级别** FIFO 接收字节数与RFITL匹配时，RDA_IF 将被置位 (如果UA_IER [RDA_IEN]使能, 将发生中断). **RFITL** / **中断触发级别(Bytes)** 0000 / 01 0001 / 04 0010 / 08 其它 / 14
[3]	**保留**	保留
[2]	**TFR**	**TX软件复位** 当 Tx_RST 置位, 发送FIFO中的所有字节和Tx内部状态机将被清零 0 = 该位写 0 将无效. 1 =该位置位将复位 Tx 内部状态机和指针. **注:** 该位自动清零需要至少3个UART时钟周期
[1]	**RFR**	**Rx 软件复位** 当 Rx_RST 置位, 接收FIFO中所有字节和Rx内部状态机都将被清零 0 = 该位写 0 将无效. 1 = 该位置位将复位 Rx 内部状态机和指令状态. **注:** 该位自动清零需要至少3个UART时钟周期
[0]	**保留**	保留

Line Control Register (UA_LCR)

寄存器	偏移量	R/W	描述	复位后的值
UA_LCR	UART0_BA+0x0C	R/W	UART0线控制寄存器	0x0000_0000
	UART1_BA+0x0C	R/W	UART1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**	

Bits	描述	
[31:7]	保留	保留
[6]	**BCB**	**Break控制位** 该位置位, 串行数据输出 (Tx) 将被迫处于Spacing State (逻辑0). 该位仅作用于Tx对传输逻辑不起作用.
[5]	**SPE**	**Stick 校验使能** 0 = 禁用 stick 校验 1 = 当 PBE , EPE 和 SPE 置位，校验位发送0，当 PBE 和 SPE 置位 并且 EPE 清除，校验位发送1
[4]	**EPE**	**偶校验使能** 0：数据位和校验位中共有奇数个逻辑1被传输和检测 . 1：数据位和校验位中共有偶数个逻辑1被传输和检测 该位仅当第三位(校验位使能位) 置位才有效..
[3]	**PBE**	**校验使能位** 0 = 当传输时校验位不发送(只发送数据)或检测 (只接收数据). 1 = 在串行数据的最后一位和停止位之间生成（发送）和检测（接收）校验位

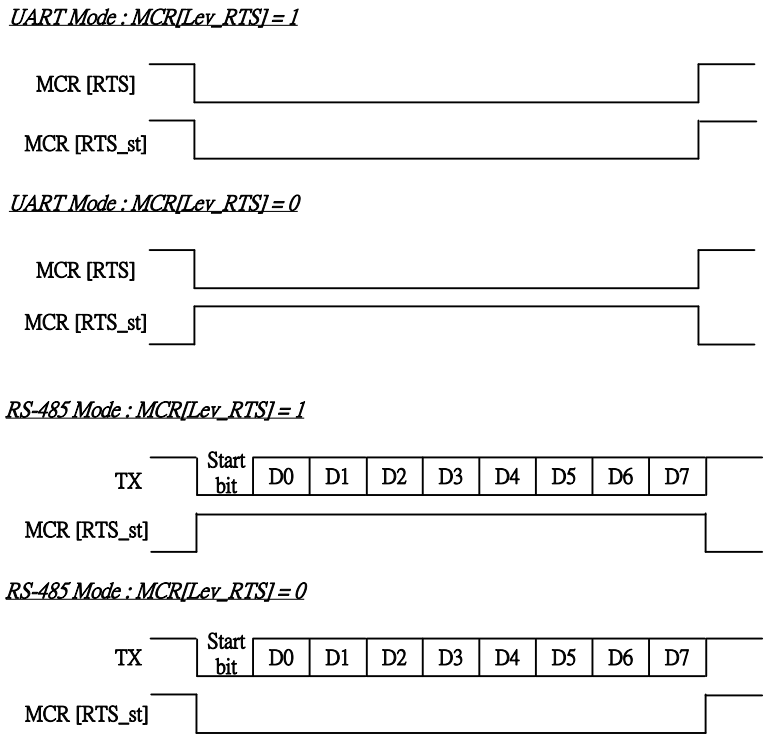
Bits	描述	
[2]	**NSB**	**“STOP bit” 个数** 0= 传递数据时产生 1个停止位 1= 传递数据时产生1.5个停止位 (5位字长度被选择时); 6, 7-和 8位字长度被选择时，产生2个停止位.
[1:0]	**WLS**	**字长度选择** **WLS[1:0]** / **字长度** 00 / 5 bits 01 / 6 bits 10 / 7 bits 11 / 8 bits

MODEM 控制寄存器 (UA_MCR)

寄存器	偏移量	R/W	描述	复位后的值
UA_MCR	UART0_BA+0x10	R/W	UART0调制解调器控制寄存器	0x0000_0000
	UART1_BA+0x10	R/W	UART1 调制解调器控制寄存器	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		RTS_ST	保留			LEV_RTS	保留
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留						RTS	保留

Bits	描述	
[31:14]	保留	保留
[13]	**RTS_ST**	**RTS 引脚的状态(只读)** 该位表示 RTS 引脚状态.
[12:10]	保留	保留
[9]	**LEV_RTS**	**RTS 触发电平** 该位可以设定 RTS 引脚的触发电平. 0= 低电平触发 1= 高电平触发

Bits	描述	
		
[8:2]	保留	保留
[1]	**RTS**	**RTS (Request-To-Send) 信号** 0 = 驱动 RTS 管脚为 1 (如果 **Lev_RTS** 设为低电平触发). 1 = 驱动 RTS 管脚为 0 (如果 **Lev_RTS** 设为低电平触发). 0 = 驱动 RTS 管脚为 0 (如果 **Lev_RTS** 设为高电平触发). 1 = 驱动 RTS 管脚为 1 (如果 **Lev_RTS** 设为高电平触发).
[0]	保留	保留

Modem状态寄存器 (UA_MSR)

寄存器	偏移量	R/W	描述	复位后的值
UA_MSR	UART0_BA+0x14	R/W	UART0 调制解调器状态寄存器	0x0000_0000
	UART1_BA+0x14	R/W	UART1 调制解调器状态寄存器	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							**LEV_CTS**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留			**CTS_ST**	保留			**DCTSF**

Bits	描述	
[31:9]	保留	保留
[8]	**LEV_CTS**	**CTS 触发电平** 该位可改变 CTS 触发电平控制TX_FIFO发送数据. 0=低电平触发 1=高电平触发
[7:5]	保留	保留
[4]	**CTS_ST**	**CTS 引脚状态(只读)** 当UART时钟使能，并且多功能引脚已经配置为CTS时，该位表示 CTS 引脚的状态.
[3:1]	保留	保留
[0]	**DCTSF**	**检测侦测 CTS 状态改变标志位(只读)** 只要 CTS 输入状态改变 该位置位，并且在UA_IER[MODEM_IEN]置位时还会向CPU产生调制解调器中断 **注:** 该位只读, 可写 '1' 清除.

FIFO 状态寄存器(UA_FSR)

寄存器	偏移量	R/W	描述	复位后的值
UA_FSR	UART0_BA+0x18	R/W	UART0 FIFO状态寄存器.	0x1040_4000
	UART1_BA+0x18	R/W	UART1 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	RS-485_ADD_DETF	保留		RX_OVER_IF

Bits	描述	
[31:29]	保留	保留
[28]	TE_FLAG	**发送器空闲标志位 (只读)** 当 Tx FIFO(UA_THR) 为空或最后一个字节的停止位被传送到之后，，该位由硬件自动置位. 当 Tx FIFO(UA_THR) 不为空或最后一个字节未传输完时，该位由硬件保持为0 注: 该位只读..
[27:24]	保留	保留
[24]	TX_OVER_IF	**发送缓冲溢出中断标志 (只读)** 如果TX FIFO (UA_THR) 已满l, 再写数据到UA_THR 将导致这个bit被置为1. **注意:** 这个bit 是只读的, 但是可以通过写'1' 清除.
[23]	TX_FULL	**发送 FIFO 满(只读)** 该位表示TX FIFO是否已满. 当TX_POINTER 等于16时，这个bit将被置为1，否则硬件将自动清成0.
[22]	TX_EMPTY	**发送FIFO 为空(只读)** 该位表示 Tx FIFO 是否为空. 当 Tx FIFO的最后一个字节传输到发送移位寄存器时，硬件置位该位. 当写数据到

Bits	描述	
		THR (Tx FIFO非空) 时硬件将自动清除该位.
[21:16]	**TX_POINTER**	**TX FIFO指针(只读)** 该位表示Tx FIFO 缓冲指针. 当CPU 写 1个字节到 UA_THR 时, Tx_Pointer 增加 1. 当 Tx FIFO 传输 1 个字节到发送移位寄存器时, Tx_Pointer 减1.
[15]	**RX_FULL**	**接收 FIFO 满 (只读)** 该位表示RX FIFO 是否已满. 当 RX_POINTER 等于 16 时，这个bit将被置位，否则硬件自动清为0.
[14]	**RX_EMPTY**	**接收FIFO 为空(只读)** 该位表示 Rx FIFO是否为空. 当 Rx FIFO 中最后一个字节 被CPU读出, 硬件置位该位. 当 UART 接收到新数据时 该位自动清除.
[13:8]	**RX_POINTER**	**Rx FIFO 指针(只读)** 该位表示 Rx FIFO 缓冲指针. 当UART 从外部设备接收到 1 个字节数据时, Rx_Pointer增加 1. 当 Rx FIFO 被 CPU 读走 1个字节数据时, Rx_Pointer 减1.
[7]	**保留**	保留
[6]	**BIF**	**Break中断标志位（只读）** 当接收数据的输入(Rx)保持在”Spacing State”(逻辑0)状态的时间大于整个字(即起始位 + 数据位 + 校验位 + 停止位的所有时间)的传输时间，该位置1。无论CPU何时向该位写1都会将该位清0。 注：该位只读，但可以写1清零.
[5]	**FEF**	**帧错误 标志位（只读）** 当接收的字符串没有正确的停止位(即检测到跟在最后一个数据位或校验位后面的停止位为逻辑0)时，该位置位。该位在CPU向其写1时清零 注：该位只读，但可以写1清零.
[4]	**PEF**	**奇偶校验错误 标志位(只读)** 当接收到的字符串的校验位无效时，该位将置位, CPU 写 1 到 该位 复位 注：该位只读，但可以写1清零.
[3]	**RS-485_ADD_DETF**	**RS-485地址字节检测标志 (只读)** RS-485模式下，如果UA_ALT_CSR [RS-485_ADD_EN]为1，只要接收器检测到地址字节（接收到的字节第9位为1），该位将置位。该位写1清0. **注：** 该位只用于RS-485 模式. **注：** 该位只读，但可写1清零.
[2:1]	**保留**	保留

Bits	描述	
[0]	**RX_OVER_IF**	**RX 溢出标志 (只读)** 当RX FIFO 溢出时，这个位将被置位. 如果接收到的字节数大于RX_FIFO (UA_RBR) 的大小，UART0/UART1 都为16个字节，这个bit 将被置位. **注意:** 这个 bit 是只读的，,但是用户可以写1清0.

中断状态控制寄存器(UA_ISR)

寄存器	偏移量	R/W	描述	复位后的值
UA_ISR	UART0_BA+0x1C	R/W	UART0中断状态寄存器	0x0000_0002
	UART1_BA+0x1C	R/W	UART1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

Bits	描述	
[31:16]	保留	保留
[15]	LIN_RX_BREAK_INT	**LIN 总线接收到Break 域中断标志(只读)** 如果 LIN_RX_BRK_IEN 和 LIN_RX_BREAK_IF 都被设为1，这个bit将被设为1. 1 = 发生了LIN 接收到Break 中断 0 = LIN接收到Break 中断没有发生
[14]	保留	保留
[13]	BUF_ERR_INT	**缓冲错误中断标志 (只读)** 如果 BUF_ERR_IEN 和 BUF_ERR_IF 都被设为 1，这个bit将被设为1. 1 = 发生了缓冲错误中断 0 = 缓冲错误中断没有发生
[12]	TOUT_INT	**超时中断标志(只读)** 如果TOUT_IEN 和 TOUT_IF都被设为1，这个bit将被设为1
[11]	MODEM_INT	**调制解调器中断标志(只读).** 如果MODEM_IEN 和 MODEM_IF都被设为1，这个bit将被设为1
[10]	RLS_INT	**接收 Line状态中断标志(只读).**

Bits	描述	
		如果RLS_IEN 和RLS_IF都被设为1，这个bit将被设为1
[9]	**THRE_INT**	**发送保持寄存器空中断标志 (只读).** 如果THRE_IEN 和 THRE_IF都被设为1，这个bit将被设为1
[8]	**RDA_INT**	**接收数据可得中断标志 (只读).** 如果RDA_IEN 和 RDA_IF都被设为1，这个bit将被设为1
[7]	**LIN_RX_BREAK_IF**	**LIN 总线接收到Break 域标志 (只读)** 当 RX 接收到 LIN Break 域时，这个bit将被置位. 如果 UA_IER [LIN_RX_BRK_IEN] 被使能， LIN 接收到 Break 域中断将发生. **注意:** 这个bit是只读的，但是可以写”1”清0.
[6]	**保留**	保留
[5]	**BUF_ERR_IF**	**缓冲错误中断标志 (只读)** 当 TX 或者 RX 缓冲溢出，或者 Break 中断标志，或者校验错误标志，或者帧错误标志(TX_OVER_IF 或者 RX_OVER_IF 或者 BIF 或者 PEF 或者FEF) 被置位时，这个bit将被置位. 当 BUF_ERR_IF 被置位时，代表传输不正确。如果 UA_IER [BUF_ERR_IEN] 被使能, 缓冲错误中断将发生.
[4]	**TOUT_IF**	**超时中断标志 (只读)** 当 Rx FIFO非空且无动作时间超时计数器和TOIC 相等 该位置位. 若UA_IER [TOUT_IEN] 使能, 超时中断将产生. **注:** 该位只读，用户可读UA_RBR (Rx is in active)来将该bit清0.
[3]	**MODEM_IF**	**调制解调器中断标志 (只读)** 当 CTS引脚状态(DCTSF=1)改变 该位置位.若UA_IER [MODEM_IEN]使能, 调制解调器中断产生. **注:** 写1清该位到0.
[2]	**RLS_IF**	**接收线 状态标志 (只读).** 当 Rx 接收数据有奇偶校验错误、帧错误、Break错误时 ,该位置位, (至少3个bit中的一个, BIF, FEF 和 PEF, 置位). 若UA_IER [RLS_IEN] 使能, RLS 中断产生. **注:** 在RS-485模式, 该位包括 “接收器检测到地址字节 (第9位为1) “ **注:** 这个bit是只读的，当所有bit ：BIF, FEF 和 PEF都被清为0时，该位将被清为0.
[1]	**THRE_IF**	**发送保持寄存器空标志 (只读).** 当TX FIFO 的最后一个数据发送到发送器移位寄存器，该位置位. 如果UA_IER [THRE_IEN] 使能, THRE中断产生. **注:** 该位只读，写数据到THR 清零该位 (TX FIFO 非空).
[0]	**RDA_IF**	**接收数据标志(只读).** 当RX FIFO 中的字节数等于RFITL , RDA_IF 置位. 如果使能 UA_IER [RDA_IEN],

Bits	描述	
		RDA 中断产生. **注:** 该位只读，当RX FIFO 的的未读取字节数少于阀值(RFITL)时该位清零

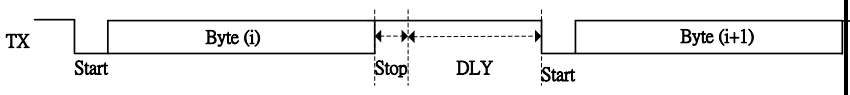
UART 中断源	中断使能位	中断指示 中断控制	中断标志位	标志位清除
LIN 接收到Break 域中断	LIN_RX_BRK_IEN	LIN_RX_BREAK_INT	**LIN_RX_BREAK_IF**	写 ‘1’ 到 LIN_RX_BREAK_IF
缓冲错误中断 INT_BUF_ERR	BUF_ERR_IEN	BUF_ERR_INT	**BUF_ERR_IF** = (TX_OVER_IF or RX_OVER_IF or BIF or PEF or FEF)	**写1到** UA_FCR [RFR]
RX超时t中断 INT_TOUT	RTO_IEN	TOUT_INT	**TOUT_IF**	读UA_RBR
调制解调器 状态中断 INT_MODEM	MODEM_IEN	MODEM_INT	**MODEM_IF** = (DCTSF)	写‘1’到DCTSF
接收线状态中断 INT_RLS	RLS_IEN	RLS_INT	**RLS_IF** = (BIF or FEF or PEF)	写‘1’到BIF/FEF/PEF
发送保持寄存器为空中断 INT_THRE	THRE_IEN	THRE_INT	**THRE_IF**	写UA_THR
接收数据中断 INT_RDA	RDA_IEN	RDA_INT	**RDA_IF**	读UA_RBR

表 6.10-3 UART 中断源和标志表（ Software Mode）

超时寄存器 (UA_TOR)

寄存器	偏移量	R/W	描述	复位后的值
UA_TOR	UART0_BA + 0x20	R/W	UART0 超时寄存器	0x0000_0000
	UART1_BA + 0x20	R/W	UART1 超时寄存器	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**
保留	**TOIC**						

Bits	描述	
[31:16]	保留	保留
[15:8]	**DLY**	**TX 延迟时间值** 该位用于编程上一停止位与下一开始位之间的延迟时间. 
[6:0]	**TOIC**	**超时比较器** 当 RX FIFO 接收到新数据后定时计数器复位和开始计数(定时器时钟频率=波特率). 一旦定时溢出计数器 (TOUT_CNT)和超时比较器 (TOIC) 相等, 且 UA_IER [RTO_IEN]使能，接收器超时中断产生 (INTR_TOUT) . 一个新的输入数据或RX FIFO 为空将清除 INT_TOUT .为了避免收到一个字节马上发生超时中断，TOIC的值应该设为40 到255之间。例如：如果TOIC设为40，没有校验位并且1个停止位的情况下，四个字节接收时间之内没有收到一个字节，超时中断发生

波特率分频寄存器(UA_BAUD)

寄存器	偏移量	R/W	描述	复位后的值
UA_BAUD	UART0_BA+0x24	R/W	UART0波特率分频寄存器	0x0F00_0000
	UART1_BA+0x24	R/W	UART1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							

Bits	描述	
[31:30]	保留	保留
[29]	DIV_X_EN	**分频 器X 使能** BRD = 波特率分频值, 波特率方程如下： **波特率= Clock / [M * (BRD + 2)]** ; 默认 M 为 16. 0 = 禁用分频器 X (M = 16) 1 = 使能分频器 X (M = X+1, 但是 DIVIDER_X [27:24] 必须 >= 8). 详见下表6.10-4 **注**: 在IrDA 模式下 该位禁用.
[28]	DIV_X_ONE	**分频系数X等于1** 0 =分频系数M = X (M = X+1,但DIVIDER_X[27:24]必须>=8) 1 = 分频系数M = 1 (M = 1, 但BRD [15:0] 必须>=3). 参考下表6.10-4.
[27:24]	DIVIDER_X	**分频 X** 波特率分频： M = X+1.
[23:16]	保留	保留
[15:0]	BRD	**波特率分频器**

Bits	描述	
		这些位表示波特率分频器

模式	DIV_X_EN	DIV_X_ONE	DIVIDER X	BRD	波特率公式
0	Disable	0	B	A	UART_CLK / [16 * (A+2)]
1	Enable	0	B	A	UART_CLK / [(B+1) * (A+2)] , B must >= 8
2	Enable	1	Don’t care	A	UART_CLK / (A+2), A must >=3

表 6.10-4 波特率方程表

IrDA 控制器寄存器 (IRCR)

寄存器	偏移量	R/W	描述	复位后的值
UA_IRCR	UART0_BA+0x28	R/W	UART0 IrDA控制寄存器.	0x0000_0040
	UART1_BA+0x28	R/W	UART1 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

Bits	描述	
[31:7]	保留	保留
[6]	**INV_RX**	**INV_RX** 1= Rx 输入信号反转 0= 无反转
[5]	**INV_TX**	**INV_TX** 1= Tx 输出信号反转 0=无反转
[4:2]	保留	保留
[1]	**TX_SELECT**	**TX_SELECT** 1: 使能IrDA 发送器 0: 使能 IrDA 接收器
[0]	保留	保留

注 : 在 IrDA 模式, 寄存器 UA_BAUD[DIV_X_EN]必须禁用 (波特率方程为Clock / 16 * (BRD)

UART 控制/状态寄存器 (UA_ALT_CSR)

寄存器	偏移量	R/W	描述	复位后的值
UA_ALT_CSR	UART0_BA+0x2C	R/W	UART0 控制/状态寄存器	0x0000_0000
	UART1_BA+0x2C	R/W	UART1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
RS-485_ADD_EN	保留				RS-485_AUD	RS-485_AAD	RS-485_NMM
7	6	5	4	3	2	1	0
LIN_TX_EN	LIN_RX_EN	保留		UA_LIN_BKFL			

Bits	描述	
[31:24]	**ADDR_MATCH**	**地址匹配值寄存器** 该位包含RS-485 的地址. **注: 该位用于**RS-485自动地址识别模式.
[23:16]	保留	保留
[15]	**RS-485_ADD_EN**	**RS-485 地址识别使能** 该位用于使能RS-485地址识别模式. 1: 使能地址识别模式 0: 禁用地址识别模式 **注:** 该位用于RS-485的所有模式.
[14:11]	保留	保留
[10]	**RS-485_AUD**	**RS-485 自动方向控制模式 (AUD)** 1: 使能RS-485 自动方向控制模式 (AUO) 0: 禁用 RS-485 自动方向操作模式(AUO) **注:** RS-485_AAD 或 RS-485_NMM 操作模式下有效.
[9]	**RS-485_AAD**	**RS-485 自动地址识别操作模式 (AAD)** 1: 使能RS-485 自动地址识别操作(AAD)

Bits	描述	
		0: 禁用 RS-485 自动地址识别操作模式(AAD) **注:** RS-485_NMM 操作模式下不要使能该位.
[8]	**RS-485_NMM**	**RS-485 普通操作模式 (NMM)** 1: 使能 RS-485 普通操作模式 (NMM) 0: 禁用 RS-485 普通操作模式 (NMM) **注:** RS-485_AAD 操作模式下不要使能该位.
[7]	**LIN_TX_EN**	**LIN 使能发送 Break 域** 1 = 使能 LIN 发送 Break 域. 0 = 禁止 LIN 发送 Break 域. **注意:** 当发送 break 域完成时，这个 bit 将被自动清除.
[6]	**LIN_RX_EN**	**LIN 接收使能** 1 = 使能 LIN 接收模式. 0 = 禁止 LIN 接收模式.
[5:4]	保留	保留
[3:0]	**UA_LIN_BKFL**	**UART LIN Break 域长度** 这个4-bit的域用来设定LIN 发送 break 域的长度 **注意:** break 域的长度为 UA_LIN_BKFL + 2

UART 功能选择寄存器 (UA_FUN_SEL)

寄存器	偏移量	R/W	描述	复位后的值
UA_FUN_SEL	UART0_BA+0x30	R/W	UART0功能选择寄存器	0x0000_0000
	UART1_BA+0x30	R/W	UART1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	

Bits	描述	
[31:2]	保留	保留
[1:0]	**FUN_SEL**	功能选择 00 = UART 01 = LIN 10 = IrDA 11 = RS-485

6.11 模拟数字转换(ADC)

6.11.1 概述

NuMicro M051™系列包含 一个8通道12位的逐次逼近式 模拟 – 数字转换器 (SAR A/D转换器). A/D 转换器支持 四种工作模式: 单次转换模式、突发转换模式、单周期扫描模式和连续扫描模式. A/D 转换可以通过软件和外部STADC/P3.2引脚启动。

6.11.2 特征

- 模拟输入电压范围: 0~0～AVDD(最大5.0V).
- 12位分辨率和10位精度保证.
- 最多 8 路单端模拟输入通道或4路差分模拟输入通道.
- 最大 ADC 时钟频率 16MHz.
- 高达760k SPS 转换速率.
- 四种操作模式
  - 单次转换模式：A/D在指定通道完成一次转换.
  - 单周期扫描模式：A/D 转换在所有指定通道完成一次转换（从低序号通道到高序号通道）.
  - 连续扫描模式：A/D 转换器连续执行单周期扫描模式直到软件停止A/D转换.
  - 突发模式：A/D转换在指定单个通道连续进行，并将结果顺序地存入FIFO.
- A/D转换开始条件
  - 软件向ADST 位写1
  - 外部引脚STADC触发
- 每个通道的转换结果存储在相应数据寄存器内，并带有有效或溢出标志.
- 转换结果可和指定的值相比较， 当转换结果和比较寄存器的设定值相匹配时，用户可设定是否产生中断请求.
- 通道 7 支持 3种输入源：外部模拟电压, 内部带隙电压和温度传感器的输出.

6.11.3 ADC 框图

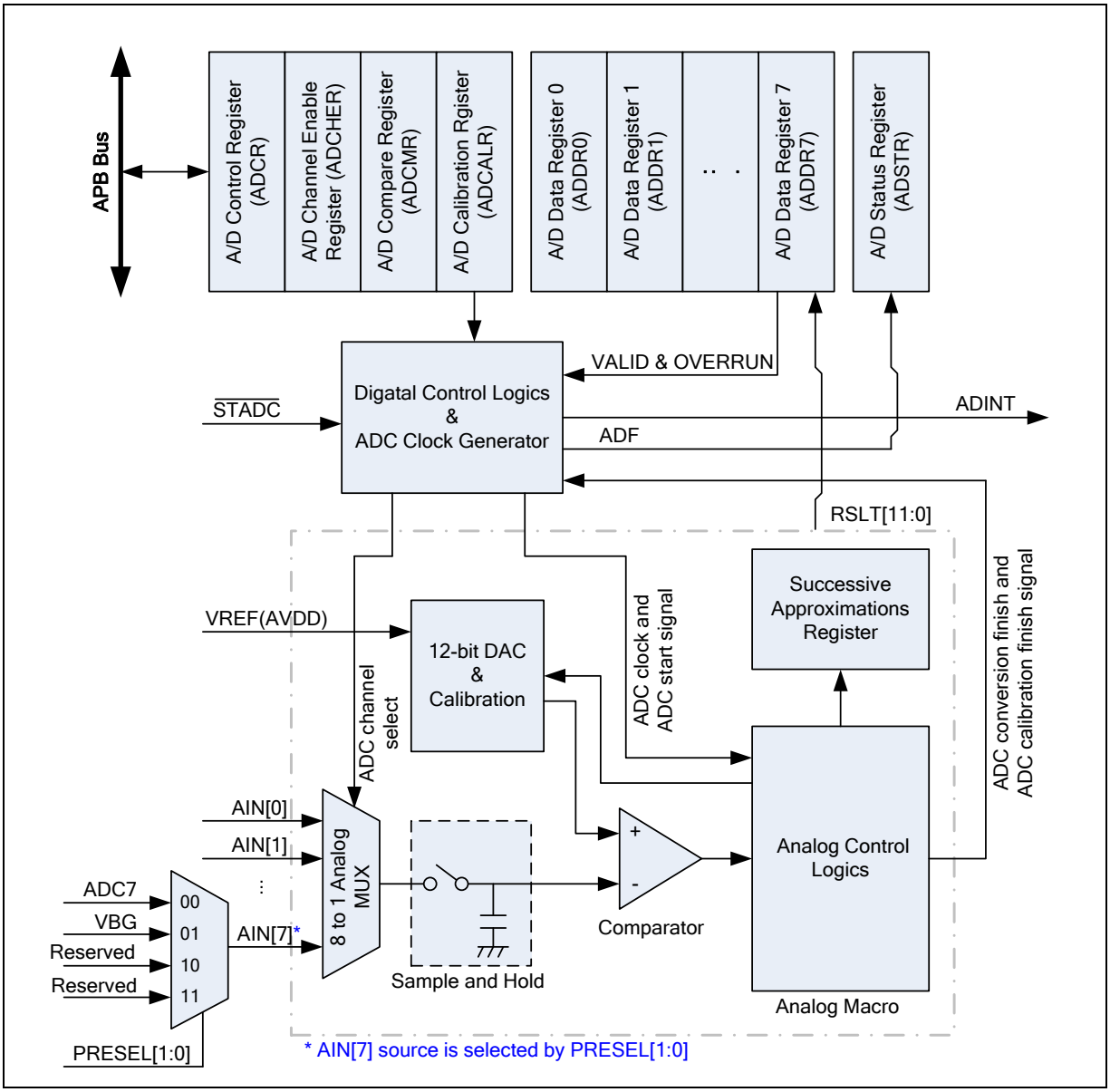


图 6.11.3-1ADC 控制器框图

6.11.4 ADC 操作步骤

A/D转换器通过逐次逼近的方式运行，分辨率为12位。ADC 共有4种操作模式: 单次转换模式、突发转换模式、单周期扫描模式和连续扫描模式. 当改变工作模式或使能的模拟输入通道时, 为了防止错误的操作, 改变之前软件需清 ADST 位为 0 (ADCR 寄存器).

6.11.4.1 ADC 时钟发生器

最大采样率达760K. ADC有三个时钟源，可由ADC_S (CLKSEL[3:2])选择, ADC时钟频率由一个8位分频器按如下公式进行8位预分频:

ADC 时钟频率 = (ADC 时钟源频率) / (ADC_N+1);
8位ADC_N 在寄存器CLKDIV[23:16]中.

通常来说，软件可以设置ADC_S 与 ADC_N 获得 16MHZ 或稍低于16MHZ的频率。

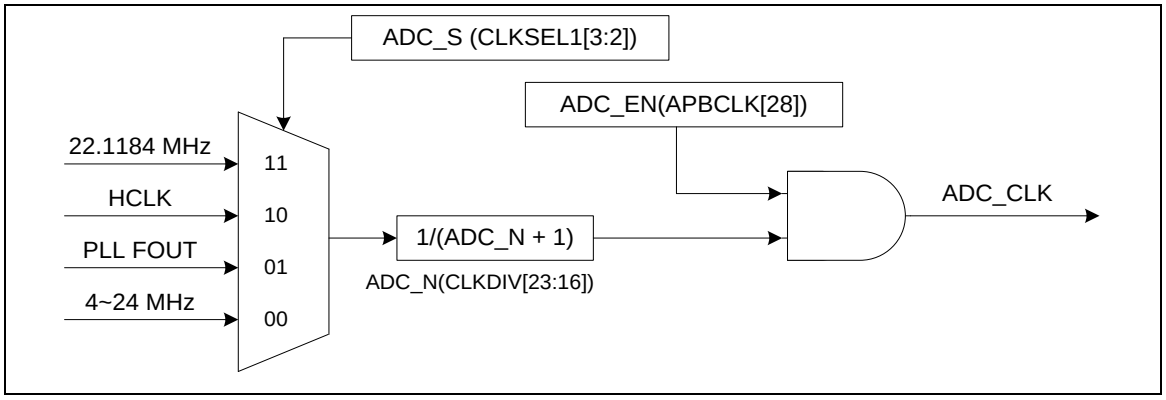


图 6.11.4-1 ADC 时钟控制

6.11.4.2 单次转换模式

在单次转换模式下, A/D 转换只在指定的通道上执行一次，操作流程如下:

1. 当通过软件或外部触发输入使 ADCR 的ADST 置位时，A/D转换开始.
2. 当 A/D 转换完成, A/D 转换的结果将存储于相应通道的A/D数据寄存器中
3. A/D 转换完成, ADSR 的ADF 位置1. 若此时ADIE 位置1, 将产生ADC中断.
4. A/D转换期间，ADST 位保持为1. A/D 转换结束, ADST 位自动清 0 ，A/D 转换器进入空闲状态。注意：在硬件清除ADST bit之后，ADST bit必须保持0至少1个ADC时钟周期，才能再次将ADST置为1，否则，A/D转换器可能不工作

注: 在单次转换模式时，如果软件使能多于一个通道，序号最小的通道被转换，其他通道被忽略.

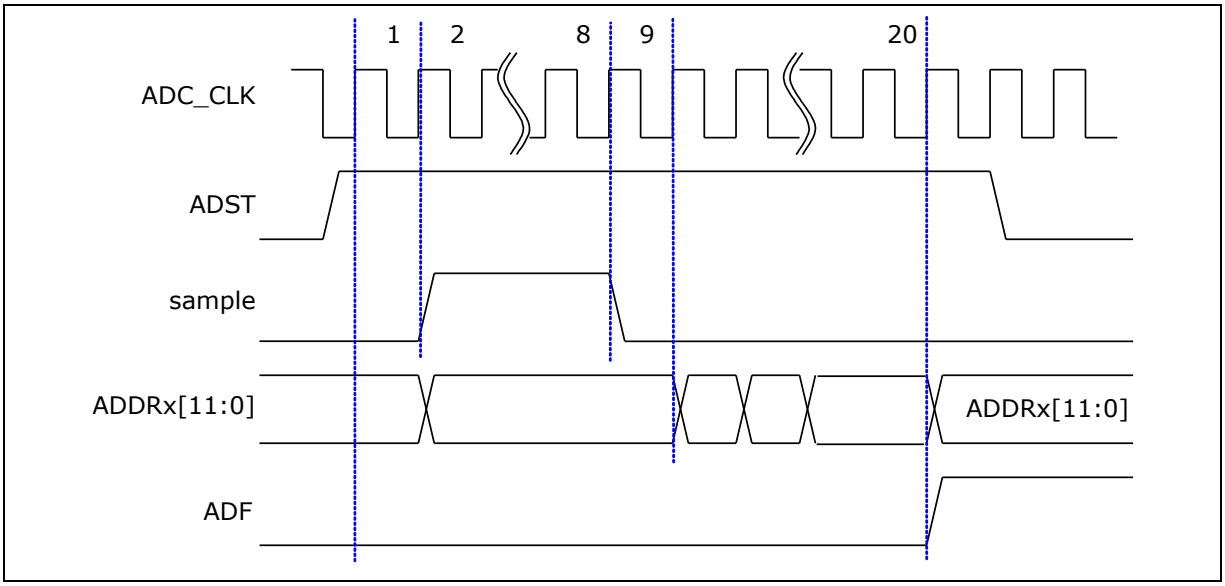


图 6.11.4-2 单次转换模式时序图

6.11.4.3 突发模式

在突发模式下, A/D 转换将采样并转换指定的单个通道,并有序的存储在FIFO中（最多8次采样）. 操作步骤如下:

1. 软件或外部触发置ADCR的ADST位为1, 在序号最小的通道上A/D 转换开始.
2. 当指定通道的A/D转换完成后，结果按序存入FIFO，可以从A/D数据寄存器0得到.
3. 当FIFO中存储了多于4个采样值时，ADSR的ADF位置1. 如果此时ADIE位置1，在A/D转换完成时就会发生ADC中断.
4. 只要ADST位保持为1，步骤2到步骤3会一直重复. 当ADST位清零时，A/D转换停止，A/D转换器进入空闲状态.

注:在突发模式下，如果软件使能多个通道, 则序号最小的通道将被转换，其他通道忽略.

6.11.4.4 单周期扫描模式

在单周期扫描模式下，从被使能的最小序号通道到最大序号通道都将进行一次A/D转换，具体流程如下:

1. 软件或外部触发使ADCR 寄存器的 ADST 位置1, 从最小序号通道开始 A/D 转换.
2. 每个使能的通道 A/D 转换完成后, A/D 转换结果将依次存放到相应的数据寄存器中.
3. 当所有使能的通道转换都完成后，ADSR的ADF位置1，如果ADC中断使能，则ADC中断发生。
4. A/D 转换结束后, ADST 位自动清 0 ，A/D 转换器进入空闲状态. 如果在所有被使能通道完成转换前 ADST 清 0 ，A/D 转换将结束当前转换，并且序号最小的通道的结果将不可预知。**注意**：在硬件清除ADST bit之后，ADST bit必须保持0至少1个ADC时钟周期，才能再次将ADST置为1，否则，A/D转换器可能不工作

使能通道(0, 2, 3 and 7) 的单周期扫描模式时序图如下:

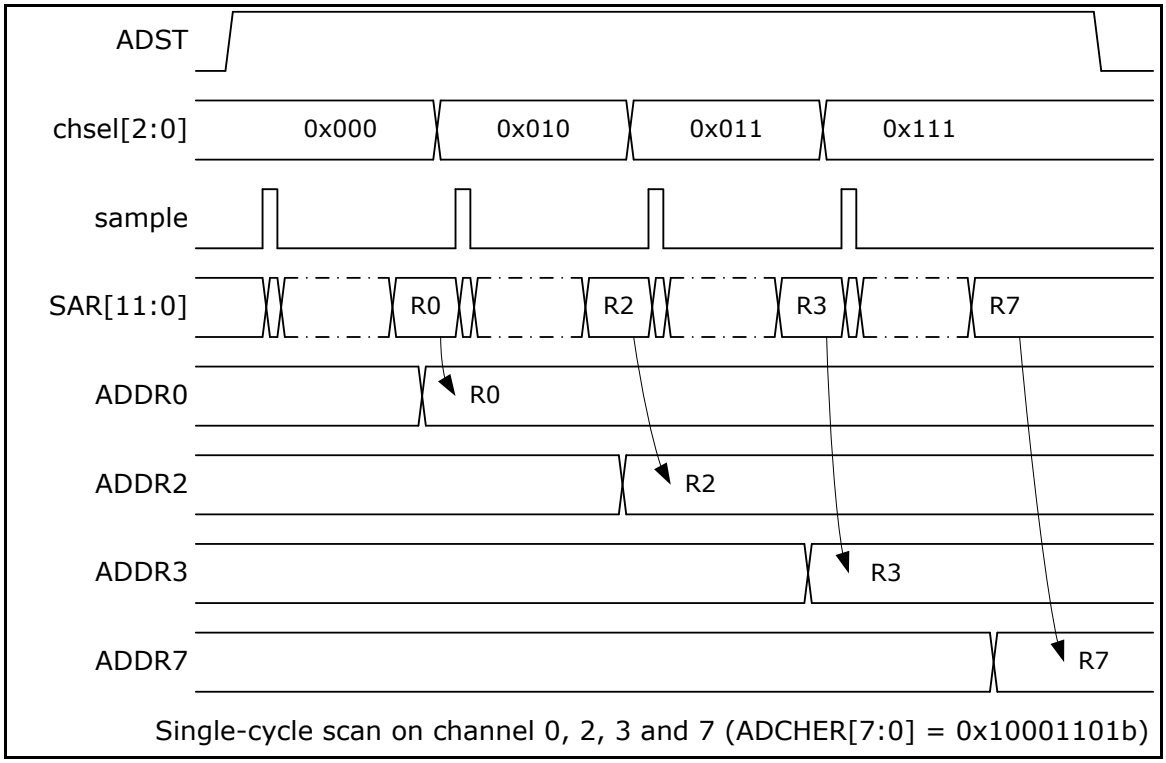


图 6.11.4-3 使能通道上单周期扫描时序图

6.11.4.5 连续扫描模式

在连续扫描模式下，A/D转换在通过ADCHER寄存器中的CHEN位使能的通道上顺序进行(最多8个ADC通道). 操作步骤如下:

1. 通过软件或外部触发使ADCR 寄存器的 ADST 位置位，从最小序号通道到最大序号通道的 A/D 转换开始.
2. 每路 A/D 转换完成后, A/D 转换结果将存储到相应的数据寄存器中.
3. 当被使能的所有通道都完成了一次转换后， ADF 位 （ADSR 寄存器）置1. 如果ADC中断使能，则ADC中断发生。如果软件没有清零ADST位，则在使能的最小通道号上的转换又一次开始.
4. 只要ADST位保持为1，步骤2到步骤3会一直重复.当ADST清0, ADC 控制器将结束当前转换，被使能的最小序号ADC通道的结果将不可预料。

使能通道(0, 2, 3和7) 连续扫描模式时序图如下:

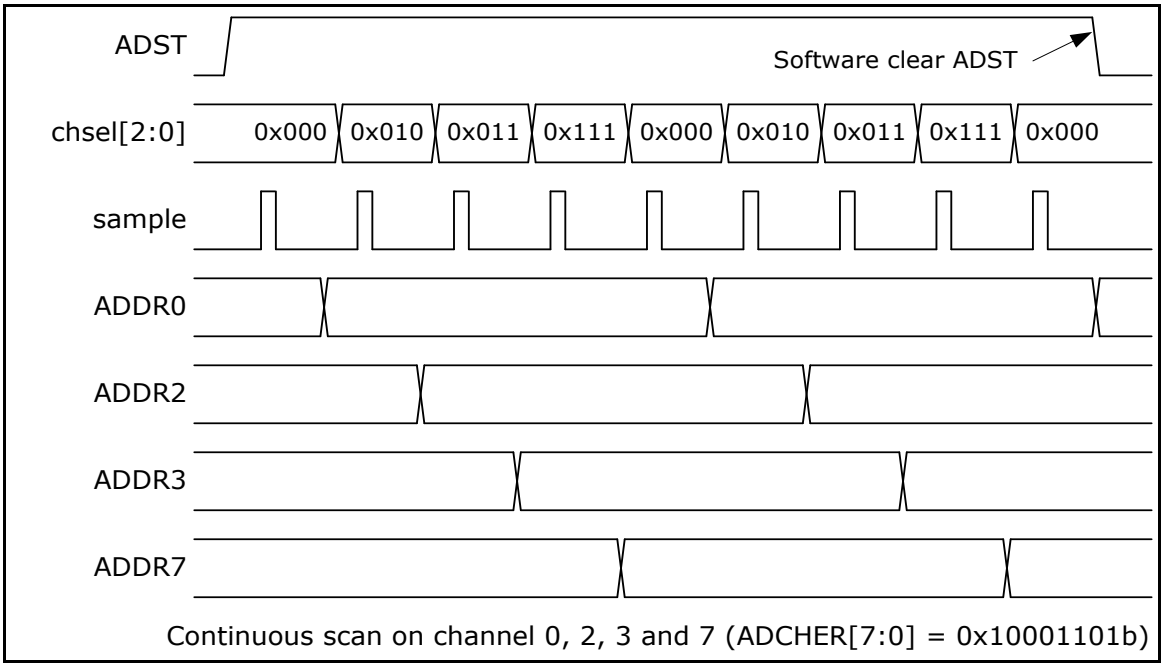


图6.11.4-4 使能通道的连续扫描时序图

6.11.4.6 外部触发输入采样和 A/D 转换时间

单周期扫描模式下，A/D 转换可通过外部引脚触发。当 ADCR.TRGEN 置位，使能 ADC 外部触发功能以后，配置 TRGS[1:0] 位为 00b选择从 STADC 引脚输入外部触发. 软件可以设定 TRGCOND[1:0] 选择触发方式为**上升沿/下降 沿 或低**电平**/高**电平触发.若选择 电平触发, STADC 需保持定义的电平状态 至少 8 个PCLK周期. 在第9个PCLK时钟时ADST 位置位，开始转换，电平触发模式下，如果外部触发输入保持为有效状态，转换连续进行. 仅当外部触发条件消失才停止。若选择边沿触发模式，高或低电平状态至少需保持 4 PLCKs周期. 脉冲低于该值时，将被忽略.

6.11.4.7 比较模式下 AD转换结果监控

NuMicro M051™ 系列提供2个比较寄存器 ADCMPR0 和ADCMPR1 ，来监控来自A/D转换模块的最多两个指定通道的转换结果, 参考图6.11.4-5。可通过软件设定CMPCH(ADCMPRx[5:0])选择监控通道，CMPCOND位用于检查转换结果小于或大于等于在CMPD[11:0]中指定的值. 当CMPCH指定的通道完成转换时，比较就被自动触发且执行一次。 当比较结果和设定值相匹配, 比较匹配计数器将加 1, 否则比较匹配计数器就清0。当计数器的值和设定值（CMPMATCNT+1）匹配，CMPF 位将置 1, 如果CMPIE 置位　将产生ADC_INT 中断请求. 在扫描模式下，软件可使用该功能来监控外部模拟输入引脚电压变化而不会增加程序负载。.具体逻辑框图如图图6.11.4-5所示:

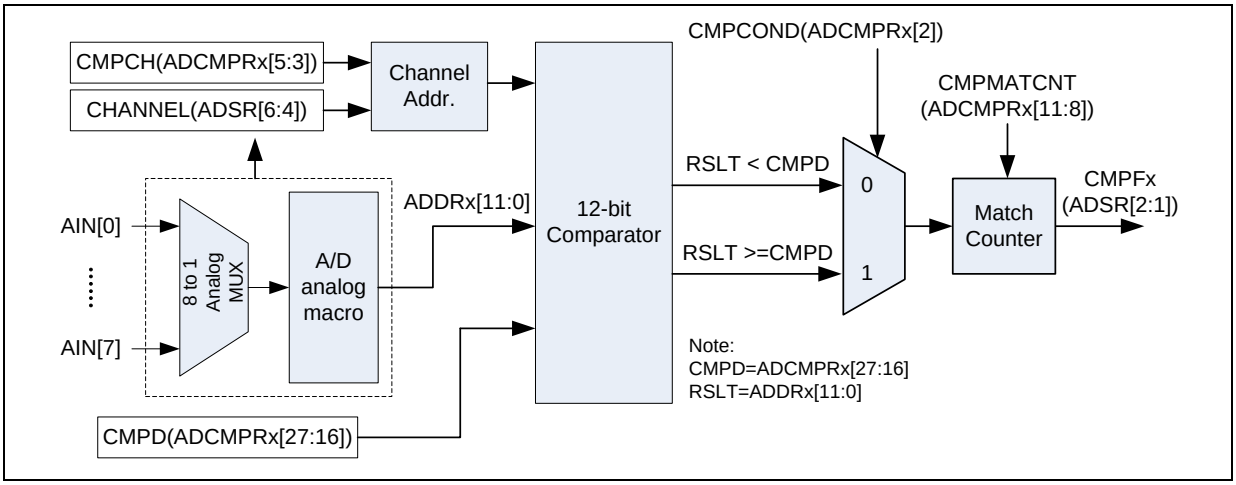


图 6.11.4-5 A/D 转换结果监控框图

6.11.4.8 中断源

ADC中断有三个中断源，A/D 转换结束时，A/D转换结束标志ADF将会被置位。CMPF0和CMPF1是比较功能的比较中断标志，当转换结果满足ADCMPR0/1的设定值，相应的标志将被置位。当ADF，CMPF0和CMPF1这三个标志位有其中一个置位，且相应的中断使能位ADCR寄存器的ADIE位，或者ADCMPR0/1中的CMPIE位被置位，ADC中断将会产生。软件可清零中断标志来撤销中断。

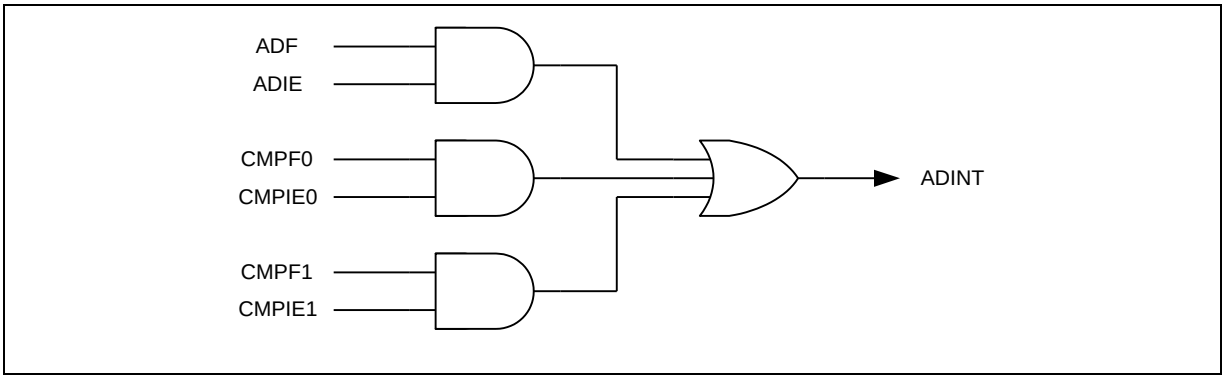


图 6.11.4-6 A/D 控制器中断

6.11.5 ADC 寄存器映射

R: 只读, W: 只写, R/W: 可读写，C:仅在值为0时可写

寄存器	偏移量	R/W	描述	复位后的值
ADC_BA = 0x400E_0000				
ADDR0	ADC_BA+0x00	R	A/D数据寄存器 0	0x0000_0000
ADDR1	ADC_BA+0x04	R	A/D数据寄存器 1	0x0000_0000
ADDR2	ADC_BA+0x08	R	A/D数据寄存器 2	0x0000_0000
ADDR3	ADC_BA+0x0C	R	A/D数据寄存器 3	0x0000_0000
ADDR4	ADC_BA+0x10	R	A/D数据寄存器 4	0x0000_0000
ADDR5	ADC_BA+0x14	R	A/D数据寄存器 5	0x0000_0000
ADDR6	ADC_BA+0x18	R	A/D数据寄存器 6	0x0000_0000
ADDR7	ADC_BA+0x1C	R	A/D数据寄存器 7	0x0000_0000
ADCR	ADC_BA+0x20	R/W	A/D控制寄存器	0x0000_0000
ADCHER	ADC_BA+0x24	R/W	A/D 通道使能寄存器	0x0000_0000
ADCMPR0	ADC_BA+0x28	R/W	A/D 比较寄存器0	0x0000_0000
ADCMPR1	ADC_BA+0x2C	R/W	A/D比较寄存器1	0x0000_0000
ADSR	ADC_BA+0x30	R/W	A/D 状态寄存器	0x0000_0000

6.11.6 ADC 寄存器描述

A/D数据寄存器 (ADDR0 ~ ADDR7)

寄存器	偏移量	R/W	描述	复位后的值
ADDR0	ADC_BA+0x00	R	A/D数据寄存器 0	0x0000_0000
ADDR1	ADC_BA+0x04	R	A/D数据寄存器 1	0x0000_0000
ADDR2	ADC_BA+0x08	R	A/D数据寄存器 2	0x0000_0000
ADDR3	ADC_BA+0x0c	R	A/D数据寄存器 3	0x0000_0000
ADDR4	ADC_BA+0x10	R	A/D数据寄存器 4	0x0000_0000
ADDR5	ADC_BA+0x14	R	A/D数据寄存器 5	0x0000_0000
ADDR6	ADC_BA+0x18	R	A/D数据寄存器 6	0x0000_0000
ADDR7	ADC_BA+0x1C	R	A/D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**
保留				**RSLT [11:8]**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RSLT [7:0]							

Bits	描述	
[31:18]	保留	**-**
[17]	**VALID**	**有效标志位(只读)** 1 = RSLT[11:0] 中的数据 有效. 0 = RSLT[11:0] 中的数据 无效. 相应模拟通道 转换完成后，该位将置位，读ADDR 寄存器后，该位由硬件清除.
[16]	**OVERRUN**	**溢出标志位(只读)** 1 = RSLT[11:0] 中的数据发生覆盖. 0 = RSLT[11:0] 中的数据没有发生覆盖.

Bits	描述	
		新的转换结果装载至寄存器之前，若 RSLT[11:0] 的数据没有被读走，，OVERRUN 将置 1， 读ADDR 寄存器后，该位由硬件清除..
[15:12]	保留	-
[11:0]	**RSLT**	**A/D 转换**结果 包括 12 位AD 转换结果.

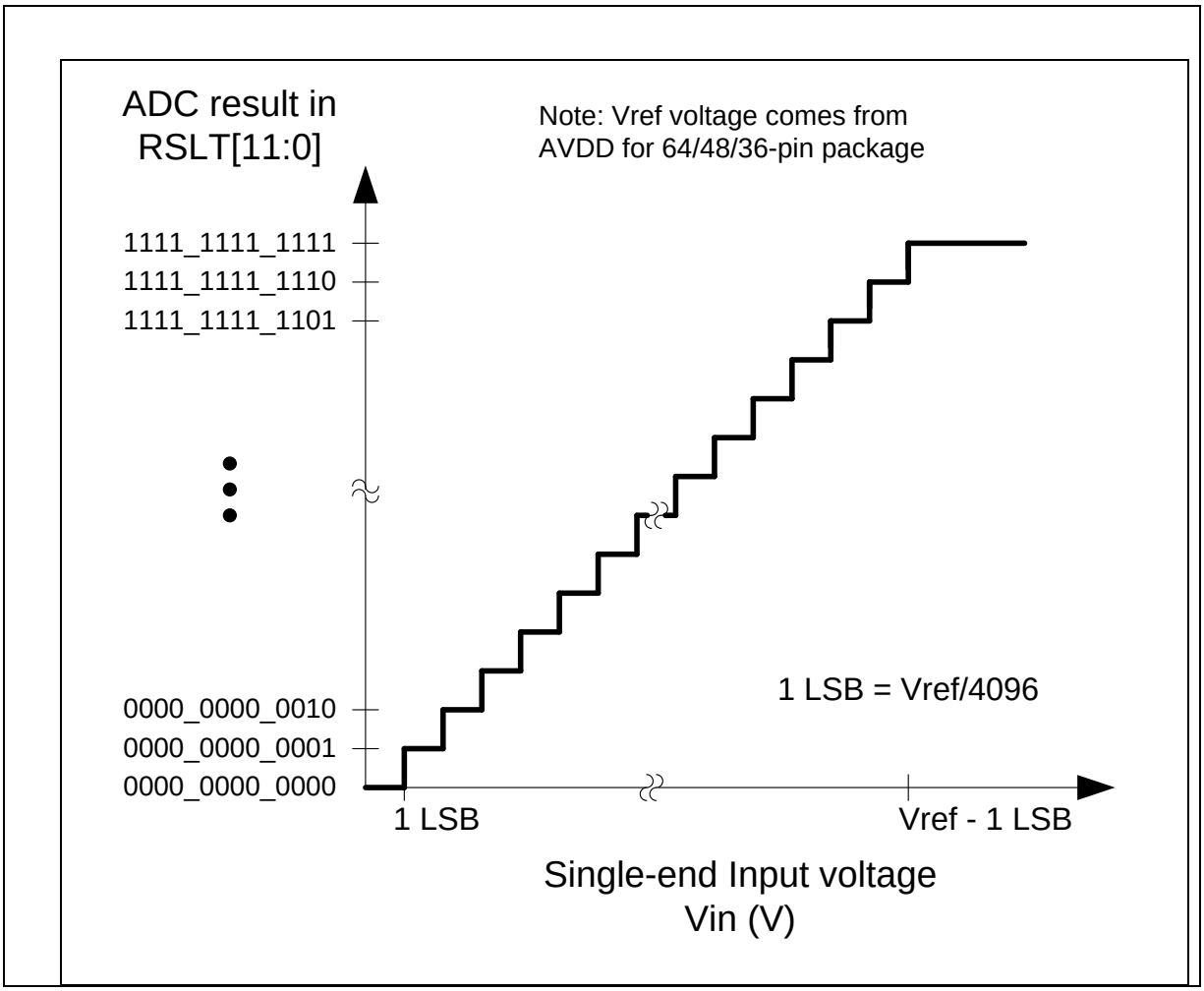


图 6.11.6-1 ADC 单端输入转换电压和转换结果映射图

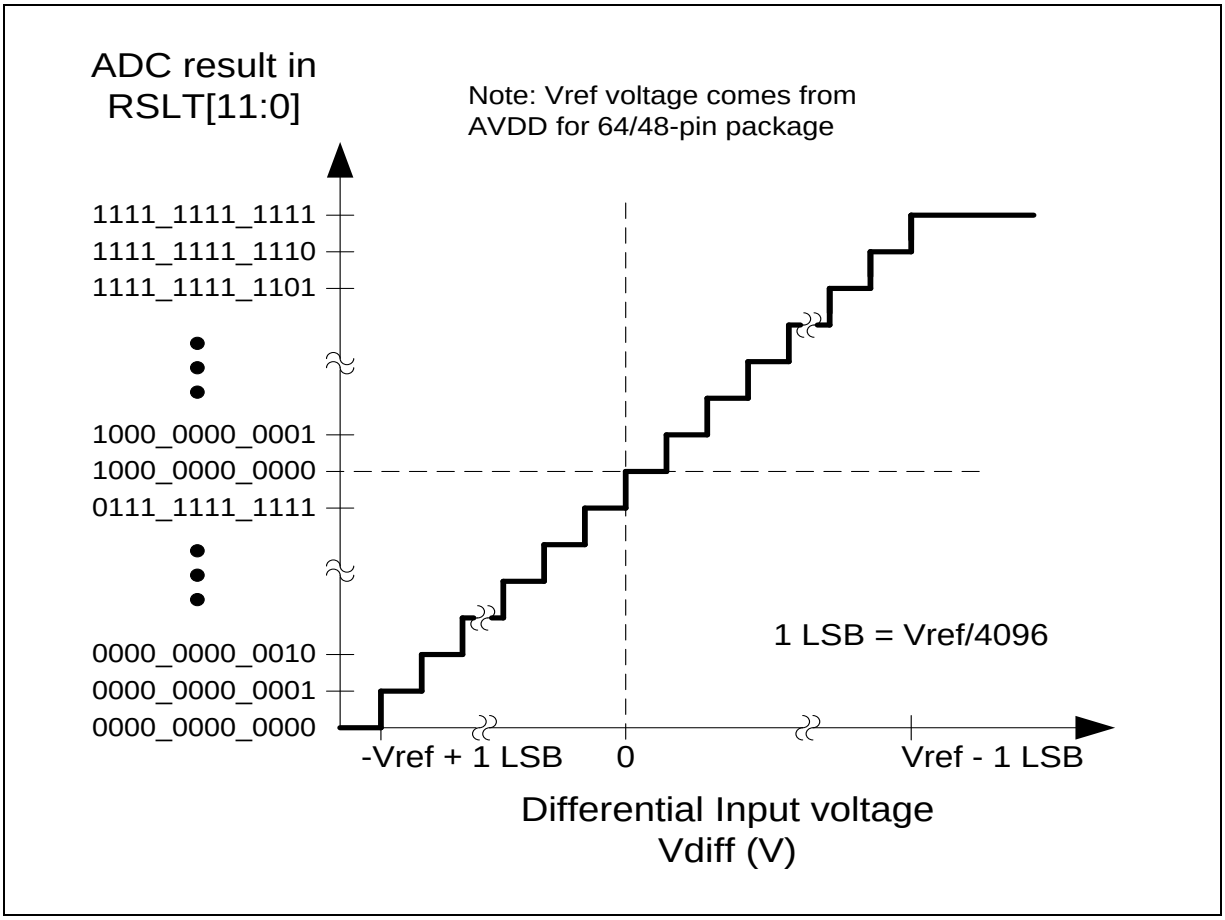


图6.11.6-2 ADC差分输入转换电压和转换结果映射图

A/D 控制寄存器 (ADCR)

寄存器	偏移量	R/W	描述	复位后的值
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**

Bits	描述	
[31]	**DMOF**	**A/D 差分输入模式输出格式** 1 = A/D 转换结果被存在ADDRx寄存器的RSLT中，格式为转换结果2的补数. 0 = /D 转换结果被存在ADDRx寄存器的RSLT中，格式为无符号数
[31:12]	保留	-
[11]	**ADST**	**A/D 转换开始** 1 =转换开始. 0 = 转换结束或A/D进入空闲状态. ADST 位置位有下列 2 种方式: 软件设定和外部STADC引脚. 单次转换模式和单周期扫描模式下，转换完成后，ADST 将被硬件自动清除。在连续扫描和burst模式下, A/D 转换将一直进行直到软件写0到该位或系统复位.
[10]	**DIFFEN**	**A/D 差分输入模式使能** 1 = A/D为差分输入模式 0 = A/D为单端输入模式

差分配对输入通道	模拟输入	
	V_{plus}	V_{minus}
0	AIN0	AIN1

Bits	描述	
		<table><tr><td>1</td><td>AIN2</td><td>AIN3</td></tr><tr><td>2</td><td>AIN4</td><td>AIN5</td></tr><tr><td>3</td><td>AIN6</td><td>AIN7</td></tr></table> **差分输入电压(V_{diff}) = V_{plus} - V_{minus}**, V_{plus} 和 V_{minus} 为差分输入两个引脚上的电压 注: 在差分输入模式下, 只有两个相关的偶数通道需要在ADCHER使能. 转换结果将放置于相应的使能通道的寄存器里。
[9]	保留	-
[8]	**TRGEN**	**外部触发使能** 使能或禁止外部STADC触发A/D 转换 1= 使能 0= 禁用 只有在单周期扫描模式下，才支持外部STADC引脚触发A/D转换
[7:6]	**TRGCOND**	**外部触发条件** 该 2 位 决定外部STADC 引脚触发为电平触发还是边沿触发。 电平触发时，该信号必须保持至少8 个PCLKS的稳定时间；边沿触发下，至少保持4 个PCLKs 的高电平或低电平状态. 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引脚启动 其它 =保留 改变 TRGS 前，软件需要先关闭TRGE 和 ADST. 在硬件触发模式下, ADST 由外部引脚STADC触发置位.
[3:2]	**ADMD**	**A/D 转换模式** 00 = 单次转换 01 = 突发转换 10 = 单周期扫描 11 = 连续扫描 当改变转换模式时, 软件要首先禁用 ADST 位 注：在突发模式下，A/D转换结果总是存储在数据寄存器0中
[1]	**ADIE**	**A/D 中断使能** 1 = 使能 A/D 中断功能 0 = 禁用 A/D 中断功能 如果ADIE置位，A/D 转换结束将产生中断请求.

Bits	描述	
[0]	**ADEN**	**A/D 转换使能** 1 = 使能 0 = 禁用 开始 A/D 转换功能时,该位需 置位。该位为 0 将关闭 A/D 转换模拟电路的电源供给.

A/D通道使能寄存器(ADCHER)

寄存器	偏移量	R/W	描述	复位后的值
ADCHER	ADC_BA+0x24	R/W	A/D通道使能	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**
CHEN7	**CHEN6**	**CHEN5**	**CHEN4**	**CHEN3**	**CHEN2**	**CHEN1**	**CHEN0**

Bits	描述	
[31:10]	保留	-
[9:8]	**PRESEL[1:0]**	**模拟输入通道7选择** 00= 外部模拟输入 01= 内部参考源电压 10= 内部温度传感器 11= 保留 注意：当软件选择内部band-gap电压作为ADC通道7的输入源时，ADC时钟频率需要小于300KHz
[7]	**CHEN7**	**模拟输入通道 7使能** 1 = 使能 0 = 禁用
[6]	**CHEN6**	**模拟输入通道 6使能** 1 = 使能 0 = 禁用
[5]	**CHEN5**	**模拟输入通道 5使能** 1 = 使能 0 = 禁用
[4]	**CHEN4**	**模拟输入通道 4使能** 1 = 使能

Bits	描述	
		0 = 禁用
[3]	**CHEN3**	**模拟输入通道 3使能** 1 = 使能 0 = 禁用
[2]	**CHEN2**	**模拟输入通道 2使能** 1 = 使能 0 = 禁用
[1]	**CHEN1**	**模拟输入通道 1使能** 1 = 使能 0 = 禁用
[0]	**CHEN0**	**模拟输入通道 0使能** 1 = 使能 0 = 禁用

A/D 比较寄存器 0/1 (ADCMPR0/1)

寄存器	偏移量	R/W	描述	复位后的值
ADCMPR0	ADC_BA+0x28	R/W	A/D比较寄存器0	0x0000_0000
ADCMPR1	ADC_BA+0x2C	R/W	A/D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	CPMEN

Bits	描述	
[31:28]	保留	-
[27:16]	**CMPD**	**比较数值** 此12 位数值将和 指定通道的转换结果相比较, 当DMOF为0时，ADC比较器将CMPD和无符号数的转换结果比较，CMPD应该填无符号数 当DMOF为1时，ADC比较器将CMPD和2的补数的转换结果比较，CMPD应该填2的补数
[15:12]	保留	-
[11:8]	**CMPMATCNT**	**比较匹配次数** 当指定A/D通道的转换值和比较条件CMPCOND[2]相匹配, 内部计数器将相应的加1.当内部计数器的值达到设定值(CMPMATCNT +1) 时, 硬件将置位CMPF 位.
[5:3]	**CMPCH**	**Compare 通道选择** 000 = 选择比较通道0 的转换结果. 001 = 选择比较通道1 的转换结果. 010 = 选择比较通道2 的转换结果. 011 = 选择比较通道3 的转换结果. 100 = 选择比较通道4 的转换结果. 101 = 选择比较通道5 的转换结果.

Bits	描述	
		110 = 选择比较通道6 的转换结果. 111 = 选择比较通道7 的转换结果.
[2]	**CMPCOND**	**比较条件** 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(ADCMPRx[27:16])，内部匹配计数器加1. 0= 设置比较条件为当12 位A/D 转换结果小于12 位CMPD(ADCMPRx[27:16])，内部匹配计数器减1. **注:** 当内部计数器的值达到(CMPMATCNT +1), CMPF 置位.
[1]	**CMPIE**	**比较中断使能** 1 = 使能比较功能中断 0 = 禁用比较功能中断 如果使能比较功能，且比较条件与CMPCOND和CMPMATCNT的设置匹配, CMPF位有效, 同时, 如果CMPIE 置1, 产生比较中断请求.
[0]	**CMPEN**	**比较使能** 1 = 使能比较. 0 = 禁用比较. 当转换数据装载到ADDR 寄存器时，该位置位将使能ADC控制器将CMPD[11:0]与特定通道的转换值进行比较。

A/D 状态寄存器 (ADSR)

寄存器	偏移量	R/W	描述	复位后的值
ADSR	ADC_BA+0x30	R/W	ADC状态寄存器	undefined

31	30	29	28	27	26	25	24
保留							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
OVERRUN							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VALID							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留	**CHANNEL**			**BUSY**	**CMPF1**	**CMPF0**	**ADF**

Bits	描述	
[31:24]	保留	-
[23:16]	**OVERRUN**	**溢出标志 (只读)** 为ADDRx的OVERRUN位的镜像 当ADC工作于突发模式时，若FIFO溢出, OVERRUN[7:0] 全部置1.
[15:8]	**VALID**	**数据有效标志位(只读)** ADDRx的VALID位的镜像 当ADC工作于突发模式时，若FIFO有效 VALID [7:0] 全部置1.
[7]	保留	-
[6:4]	**CHANNEL**	**当前转换通道** 这3位在BUSY=1时 表示进行转换中的通道. 当BUSY=0, 表示可进行下次转换的通道. 只读位.
[3]	**BUSY**	**忙/空闲** 1 = A/D 转换器忙碌 0 = A/D 转换器空闲 该位是ADST 位(ADCR). 的镜像 只读位.

Bits	描述	
[2]	**CMPF1**	**比较标志位** 当选择的 A/D通道转换结果和ADCMPR1相匹配时，该位置1.。写 1 清该位. 1 = ADDR 转换结果和 ADCMPR1相匹配 0 = ADDR 转换结果和 ADCMPR1不匹配
[1]	**CMPF0**	**比较标志位** 当选择的 A/D通道转换 结果和ADCMPR 0相匹配时，该位置1。 写 1 清该位. 1 = ADDR 转换结果和 ADCMPR0相匹配 0 = ADDR 转换结果和 ADCMPR0不匹配
[0]	**ADF**	**A/D转换结束标志位** 状态标志位 指示A/D 转换结束. ADF 在下列三个条件时置位: 1. 单次转换模式下A/D转换结束时 2. 扫描模式下在所有指定通道A/D转换结束时. 3. 突发模式下，FIFO存储多于4个转换结果 该标志写1清零.

6.12 模拟比较器 (ACMP)

6.12.1 概述

NuMicro M051™ 系列包含两个比较器. 比较器可以一些不同的配置下使用。当正端输入大于负端输入时，比较器输出逻辑"1"，否则输出"0"。每个比较器可以配置当比较器输出值改变时发生中断。框图如图 6.12.3-1所示.

6.12.2 特性

- 模拟输入电压范围: 0~5.0V
- 支持迟滞功能
- 两个模拟比较器负端可以选择输入内部参考电压
- 两个比较器共享一个中断向量

6.12.3 框图

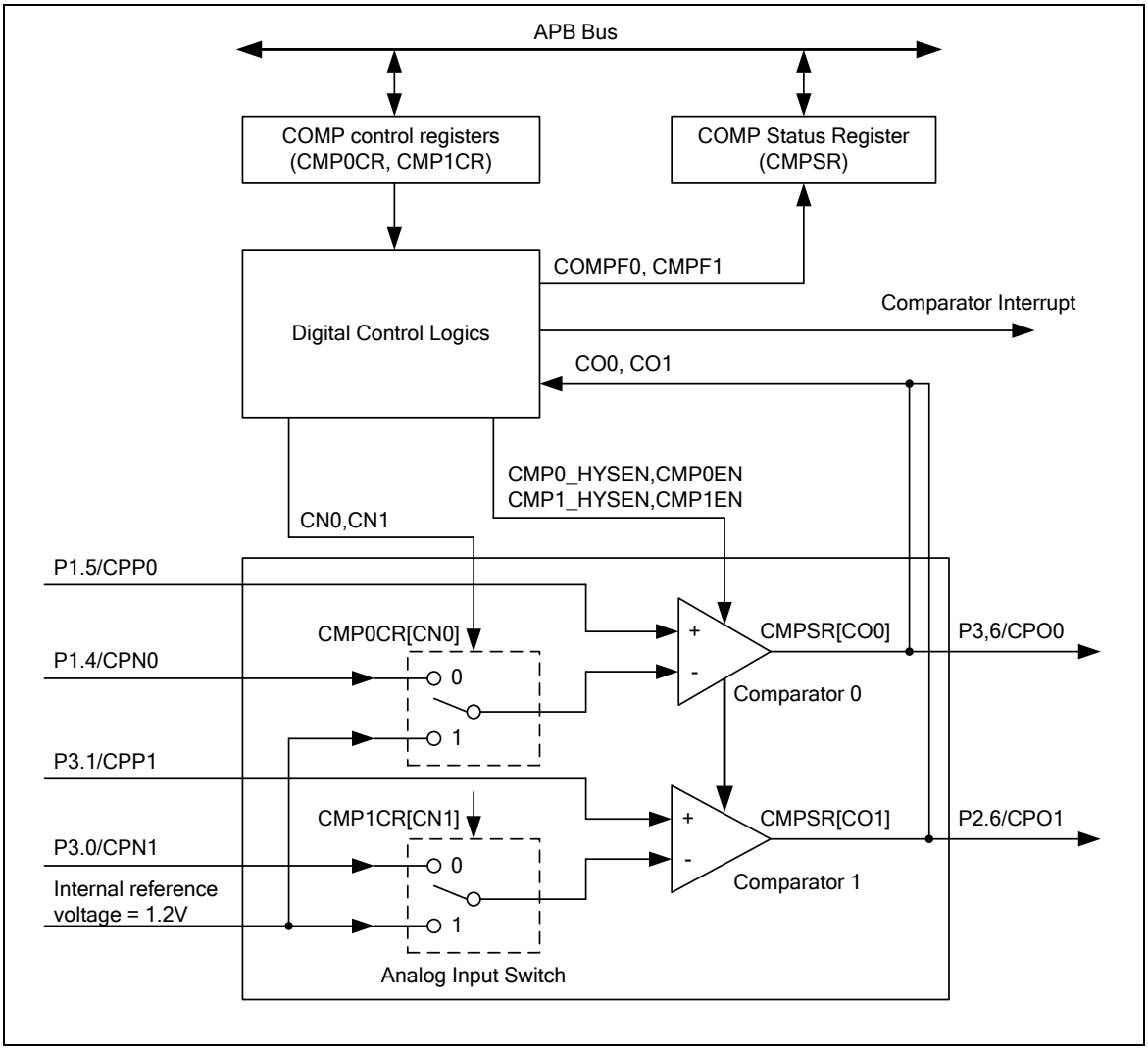


图 6.12.3-1 模拟比较器框图

6.12.4 功能描述

6.12.4.1 *中断源*

比较器由PCLK采样在CMPSR寄存器的CO1(CO2)bit产生输出。如果寄存器CMP0CR (CMP1CR)中的CMP0IE (CMP1IE) bit 为1，比较器输出CO0 (CO1)的值改变将导致比较器标志位CMPF0 (CMPF1) 被设，比较器中断将发生。软件可以写 0 到 CMP0 和 CMPF1 来停止中断请求.

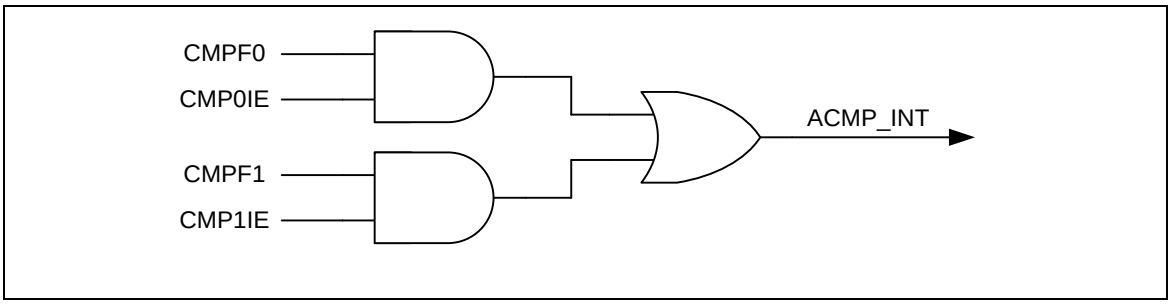


图 6.12.4-1 比较器中断源

6.12.5 寄存器映射

R: read only, **W**: write only, **R/W**: both read and write

寄存器	偏移	R/W	描述	复位后的值
CMP_BA = 0x400D_0000				
CMP0CR	CMP_BA+0x00	R/W	比较器0 控制寄存器	0x0000_0000
CMP1CR	CMP_BA+0x04	R/W	比较器1 控制寄存器	0x0000_0000
CMPSR	CMP_BA+0x08	R/W	比较器状态寄存器	0x0000_0000

6.12.6 寄存器描述

CMP0 控制寄存器 (CMP0CR)

寄存器	偏移	R/W	描述	复位后的值
CMP0CR	CMP_BA+0x00	R/W	比较器0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			CMP0CN	保留	CMP0_HYSEN	CMP0IE	CMP0EN

Bits	描述	
[31:5]	保留	保留
[4]	CMP0CN	比较器0 负端输入选择 1 = 负端选择输入内部比较器参考电压 0 =比较器参考引脚CPN0作为负端输入
[3]	保留	保留
[2]	CMP0_HYSEN	使能比较器0迟滞 功能 1 = 使能 CMP0 迟滞功能，典型范围20mV. 0 = 关闭 CMP0 迟滞功能 (默认).
[1]	CMP0IE	使能比较器0的中断 1 = 使能 CMP0 的中断 0 = 禁止 CMP0 中断功能 如果CMP0IE 等于 1，比较器转换结束时，中断将发生

Bits	描述	
[0]	**CMP0EN**	**使能比较器0** 1 = 使能 0 = 禁止 CMP0EN 置位以后，比较器输出需要等10 us 的稳定时间

CMP1 控制寄存器(CMP1CR)

寄存器	偏移	R/W	描述	复位后的值
CMP1CR	CMP_BA+0x04	R/W	比较器1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			CN1	保留	CMP1_HYSEN	CMP1IE	CMP1EN

Bits	描述	
[31:5]	保留	保留
[4]	**CN1**	**比较器1负端输入选择** 1 = 负端选择输入内部比较器参考电压 0 =比较器参考引脚CPN0作为负端输入
[3]	**保留**	保留
[2]	**CMP1_HYSEN**	**使能比较器1迟滞 功能** 1 = 使能 CMP1 迟滞功能，典型范围20mV. 0 = 关闭 CMP1 迟滞功能 (默认).
[1]	**CMP1IE**	**使能比较器1的中断** 1 = 使能 CMP1 的中断 0 = 禁止 CMP1 中断功能 如果CMP1IE 等于 1，比较器转换结束时，中断将发生.

Bits	描述	
[0]	**CMP1EN**	**使能比较器1** 1 = 使能 0 = 禁止 CMP1EN 置位以后，比较器输出需要等10 us 的稳定时间

CMP 状态寄存器 (CMPSR)

寄存器	偏移	R/W	描述	复位后的值
CMPSR	CMP_BA+0x08	R/W	比较器状态寄存器	undefined

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				CO1	CO0	CMPF1	CMPF0

Bits	描述	
[31:4]	保留	保留
[3]	**CO1**	**比较器1的输出** 这个bit显示当前比较器的比较结果，用户可以通过读这个bit查看比较器的输出。当比较器被关闭时(CMP1EN = 0)此bit将被清0.
[2]	**CO0**	**比较器0的输出** 这个bit显示当前比较器的比较结果，用户可以通过读这个bit查看比较器的输出。当比较器被关闭时(CMP0EN = 0)此bit将被清0.
[1]	**CMPF1**	**比较器1标志位** 当比较器1输出状态改变时，这个bit将被置位。如果CMP1IE使能，中断将发生。. 这个bit写1清0.
[0]	**CMPF0**	**比较器0标志位** 当比较器1输出状态改变时，这个bit将被置位。如果CMP0IE使能，中断将发生。. 这个bit写1清0.

6.13 外部总线接口 (EBI)

6.13.1 概述

NuMicro M051™ 系列配备一个外部总线接口 (EBI) , 以供外部设备使用.

为节省外部设备与芯片的连接引脚数，EBI支持地址总线与数据总线复用模式. 且地址锁存使能 (ALE)信号支持区分地址与数据周期.

6.13.2 特性

外部总线接口有下列功能:

1. 支持外部设备最大64K字节 (8位数据宽度)/128K字节(16位数据宽度)
2. 外部总线基本时钟频率可调 (MCLK)
3. 支持8位或 16 位数据宽度
4. 数据访问时间 (tACC), 地址锁存使能时间(tALE) 和地址保持时间(tAHD) 可调
5. 支持地址总线和数据总线复用以节省地址管脚
6. 空闲周期可配置用于不同的访问条件：写命令结束(W2X), 连续读(R2R)。

6.13.3 EBI 框图

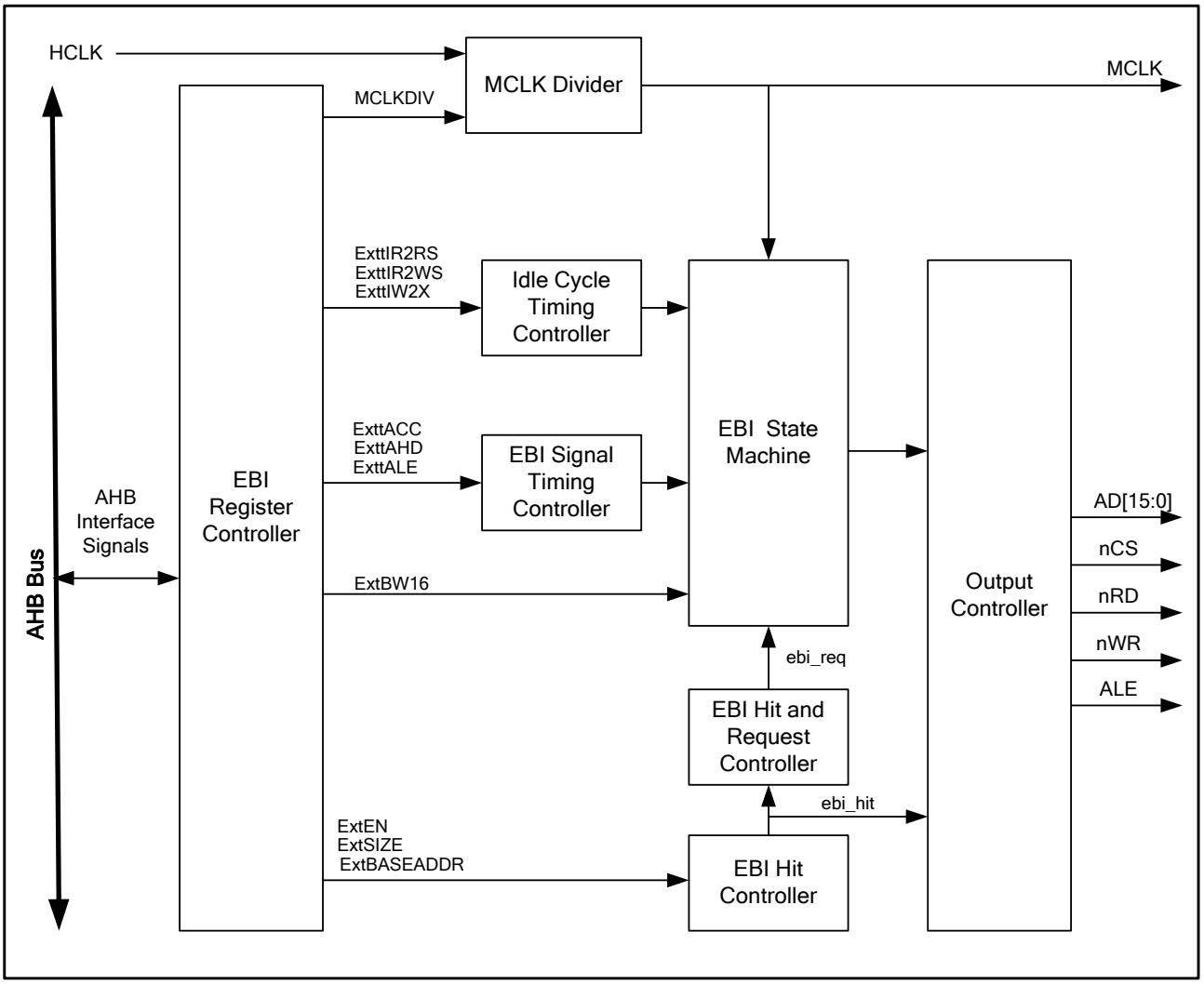


图6.13.3-1 EBI 框图

6.13.4 操作步骤

6.13.4.1 EBI 区域和地址Hit

NuMicro M051™ 系列EBI地址映射在0x6000_0000 ~ 0x6001_FFFF，总共存储器空间为128K字节。当系统请求的地址命中EBI的存储空间时, 相应的EBI片选信号有效，EBI状态机开始工作.

对于8位设备(64Kbyte), EBI把该64K字节的设备同时映射到地址0x6000_0000 ~ 0x6000_FFFF 和 0x6001_0000 ~ 0x6001_FFFF.

6.13.4.2 EBI 数据宽度连接

NuMicro M051™ 系列EBI支持地址总线和数据总线复用的设备。 对于地址总线与数据总线分开的外部设备，与设备的连接需要额外的逻辑器件来锁存地址. 这种情况下, ALE需要连接到锁存器（如74HC373）上. AD为锁存器 的输入引脚, 锁存器的输出连接到外部设备的地址总线上. 对于16位设备， AD[15:0] 由地址线与16位数据线共享. 对于 8位设备, 仅AD[7:0] 由地址线与8位数据线共享, AD [15:8]则只作为地址线直接与8位设备连接.

对于8位数据宽度, NuMicro M051™ 系统地址[15:0] 作为设备地址[15:0]. 对于16位数据宽度, NuMicro M051™ 系统地址[16:1] 作为设备地址[15:0] ,在 NuMicro M051™ 系统中地址位bit [0] 不用.

EBI bit width	System address (AHBADR)	EBI address (AD)
8 bit	AHBADR[15:0]	AD[15:0]
16 bit	AHBADR[16:1]	AD[15:0]

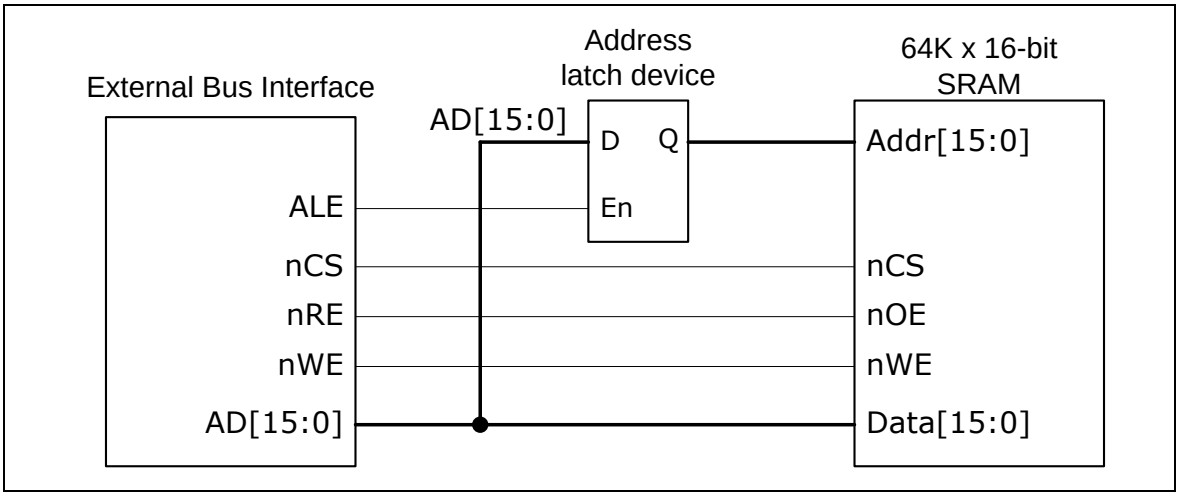


图6.13.4-1 16位EBI数据宽度与16位器件连接

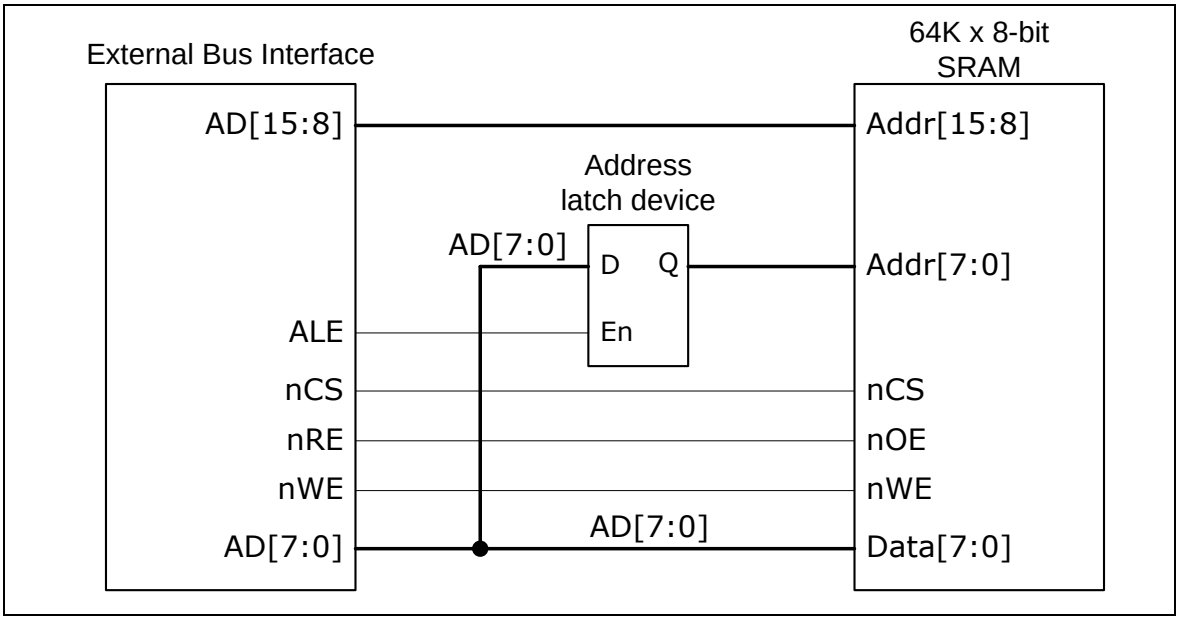


图6.13.4-2 8位EBI数据宽度与8位设备连接

当系统访问数据宽度大于EBI的数据宽度时, EBI控制器通过多次执行EBI访问来完成操作. 例如, 如果系统通过EBI设备请求32位数据, 如果EBI为8位数据宽度，EBI控制器将访问4次来完成操作.

6.13.4.3 EBI 操作控制

MCLK 控制

NuMicro M051™ 系列中, EBI工作时，通过MCLK同步所有EBI信号. 当NuMicro M051™ 系列连接到工作频率较低的外部设备时, MCLK 可以通过设置寄存器EBICON 中的MCLKDIV分频，最小可达HCLK/32. 因此, NuMicro M051™ 可以适用于宽频率范围的EBI 设备. 如果 MCLK被设置为HCLK/1, EBI 信号由MCLK的上升沿同步, 其他情况下，EBI信号由MCLK的下降沿同步.

操作与访问时序控制

开始访问时, 片选 (nCS)置低并等待一个MCLK地址建立时间(tASU)以使地址稳定. 地址稳定后ALE置高并保持一段时间 (tALE) 以用于地址锁存. 地址锁存后, ALE 置低并等待一个MCLK 的周期锁存保持时间 (tLHD) 和另一个MCLK 的周期 (tA2D) 用于总线转换（地址到数据）。然后当读操作时nRD 置低或写操作时nWR置低。在访问保持时间tACC(用于读取输出稳定或者写完成)之后nRD和nWR被拉高。之后，EBI 信号保持数据访问时间(tAHD)，然后片选信号置高, 地址由当前访问控制释放.

NuMicro M051™ 系列提供灵活的EBI 时序控制以用于不同的外部设备. 在NuMicro M051™ EBI 的时序控制中, tASU, tLHD 和 tA2D固定为1个MCLK周期, tAHD 可以通过设置寄存器EXTIME的ExttAHD在1~8个MCLK周期内调节, tACC可以通过设置寄存器EXTIME的ExttACC在1~32 个MCLK周期内调节, tALE可以通过寄存器EBICON的tALE在1~8 个MCLK 周期内调节.

参数	值	单位	描述
tASU	1	MCLK	地址锁存建立时间.
tALE	1 ~ 8	MCLK	ALE 高电平时间. 由EBICON的ExttALE控制.
tLHD	1	MCLK	地址锁存保持时间.
tA2D	1	MCLK	地址到数据的延迟 (总线转换时间).
tACC	1 ~ 32	MCLK	数据访问时间. 由EXTIME的ExttACC控制.
tAHD	1 ~ 8	MCLK	数据访问保持时间. 由EXTIME 的ExttAHB 控制.
IDLE	0 ~ 15	MCLK	空闲周期.由EXTIME 的ExtIR2R 和 ExtIW2X 控制.

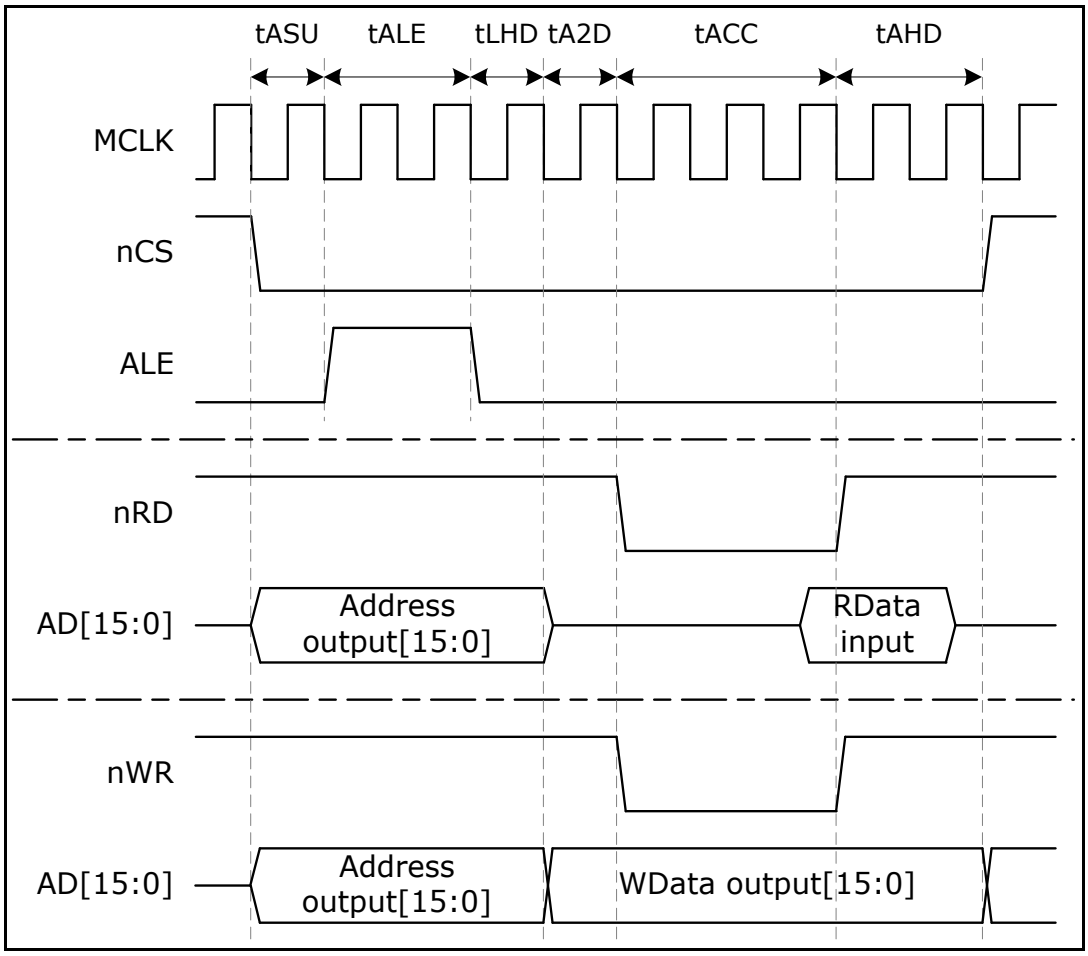


图6.13.4-3 16位数据宽度的时序控制波形

上述时序波形是以16位数据宽度为例. 此例中, AD 总线用作地址 [15:0] 和数据 [15:0]. 当 ALE 置高, AD 为地址输出. 在地址锁存后, ALE 置低并且 AD 总线转换成高阻以等待设备输出数据（在读取访问操作时），或用于写数据输出.

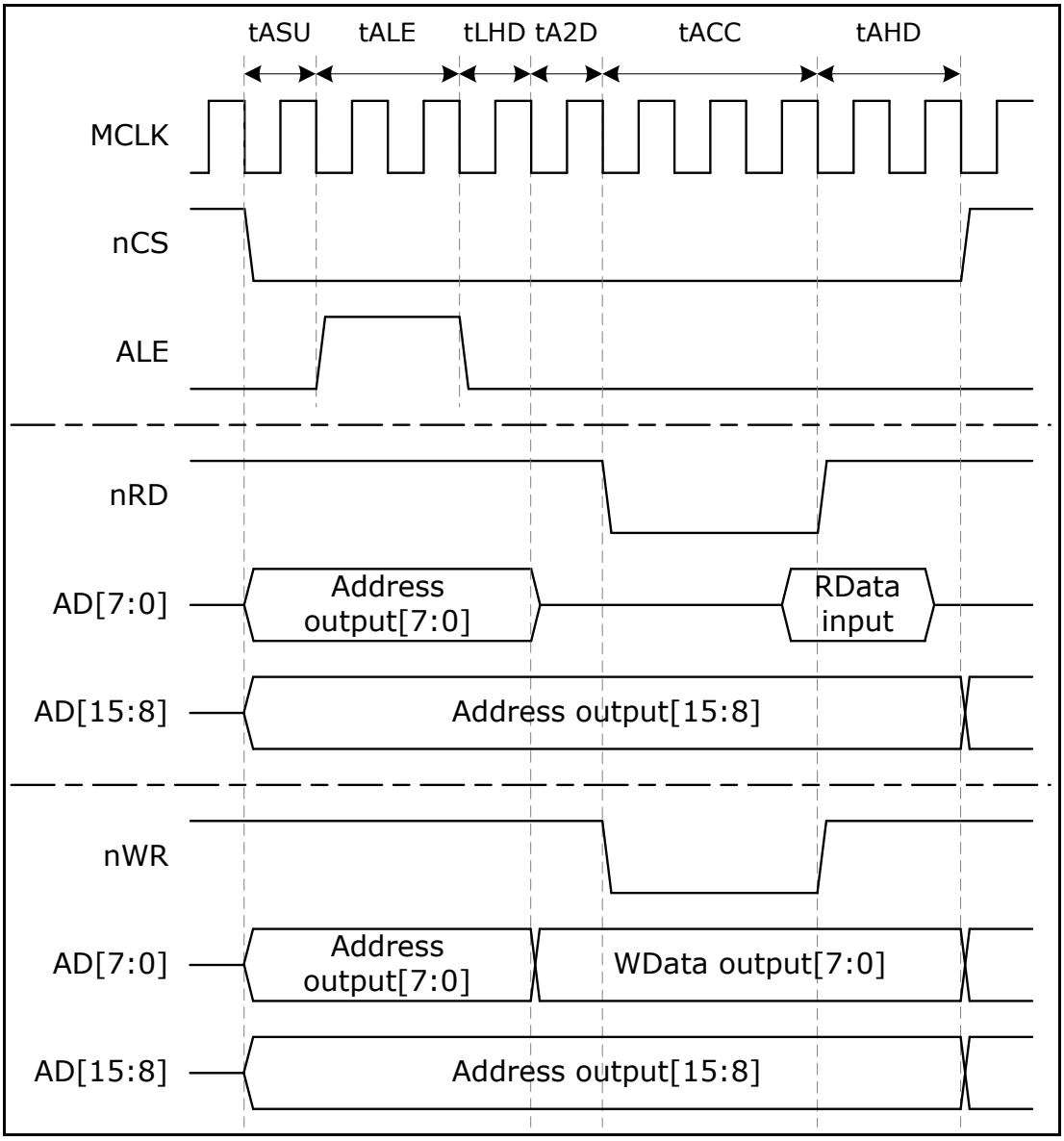


图6.13.4-4 8位数据宽度时序控制波形

上述时序波形是以8位数据宽度为例. 与16位数据宽度不同的是AD[15:8]的使用. 在8位数据宽度时, AD[15:8] 固定为地址位 [15:8] 的输出，所以外部锁存仅需要8位宽度.

插入空闲周期

当EBI连续访问时，如果设备访问速度远低于系统工作速度，可能会发生总线冲突. NuMicro M051™ 支持额外空闲周期以解决该问题. 在空闲周期, EBI的所有控制信号无效. 图**錯誤! 找不到參照來源。**为空闲周期波形图:

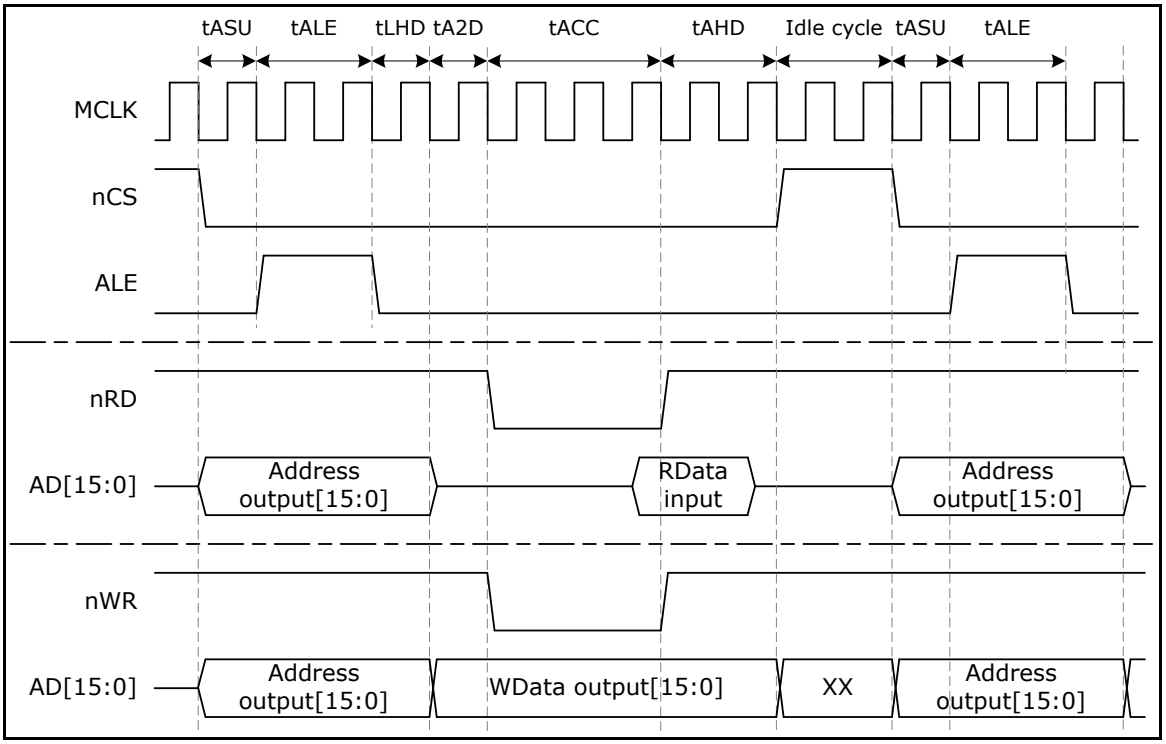


图6.13.4-5 插入空闲周期的时序控制波形

在下面两种情况下，EBI可插入空闲周期:

- 写访问之后
- 读访问之后与下一个读访问之前

通过设置寄存器EXTIME的ExtIW2X, ExtIR2R，空闲周期可设定为0~15 MCLK.

6.13.5 EBI 控制器寄存器映射

R: 只读, **W**: 只写, **R/W**: 可读写

寄存器	偏移量	R/W	描述	复位后的值
EBI_CTL_BA = 0x5001_0000				
EBICON	EBI_CTL_BA+0x00	R/W	外部总线接口通用控制寄存器	0x0000_0000
EXTIME	EBI_CTL_BA+0x04	R/W	外部总线接口时序控制寄存器	0x0000_0000

6.13.6 EBI 控制器寄存器描述

外部总线接口控制寄存器(EBICON)

寄存器	偏移量	R/W	描述	复位后的值
EBICON	EBI_CTL_BA+0x00	R/W	外部总线接口控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留					**ExttALE**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
保留					**MCLKDIV**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
保留						**ExtBW16**	**ExtEN**

Bits	描述	
[31:19]	保留	保留
[18:16]	**ExttALE**	**ALE的扩展时间** 通过ExttALE控制地址锁存ALE 时间宽度 (tALE) tALE = (ExttALE+1)*MCLK
[15:11]	保留	保留
[10:8]	**MCLKDIV**	外部输出时钟分频器

<table>
<tr><th>Bits</th><th colspan="2">描述</th></tr>
<tr><td></td><td></td><td>由 MCLKDIV 控制EBI输出时钟的频率，见下表:
<table>
<tr><th>MCLKDIV</th><th>输出频率 (MCLK)</th></tr>
<tr><td>000</td><td>HCLK/1</td></tr>
<tr><td>001</td><td>HCLK/2</td></tr>
<tr><td>010</td><td>HCLK/4</td></tr>
<tr><td>011</td><td>HCLK/8</td></tr>
<tr><td>100</td><td>HCLK/16</td></tr>
<tr><td>101</td><td>HCLK/32</td></tr>
<tr><td>11X</td><td>默认</td></tr>
</table>
注: 默认输出时钟为HCLK/1</td></tr>
<tr><td>[7:2]</td><td>保留</td><td>保留</td></tr>
<tr><td>[1]</td><td>ExtBW16</td><td>EBI 数据宽度为 16位
该位配置数据总线是8位还是16位宽度.
0 = EBI 数据宽度为8位
1 = EBI 数据宽度为16位</td></tr>
<tr><td>[0]</td><td>ExtEN</td><td>EBI 使能
该位使能EBI功能.
0 = 禁用EBI功能
1 = 使能EBI功能</td></tr>
</table>

外部总线接口时序控制寄存器(EXTIME)

寄存器	偏移量	R/W	描述	复位后的值
EXTIME	EBI_CTL_BA+0x04	R/W	外部总线接口时序控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留				**ExtIR2R**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
保留							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ExtIW2X				保留	**ExttAHD**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ExttACC					保留		

Bits	描述	
[31:28]	保留	保留
[27:24]	**ExtIR2R**	**读与读之间的空闲状态周期** 当读完成且下一个动作也是读时, 插入空闲状态周期且nCS在ExtIR2R非零时返回高. 空闲状态周期 = (ExtIR2R*MCLK)
[23:16]	保留	保留
[15:12]	**ExtIW2X**	**写之后的空闲状态周期** 当写完成, 插入空闲状态且nCS在ExtIW2X非零时返回高. 空闲状态周期 = (ExtIW2X*MCLK)
[11]	保留	保留
[10:8]	**ExttAHD**	**EBI 数据访问保持时间** ExttAHD 配置数据访问保持时间(tAHD). tAHD = (ExttAHD +1) * MCLK
[7:3]	**ExttACC**	**EBI 数据访问时间** ExttACC 配置数据访问时间 (tACC). tACC = (ExttACC +1) * MCLK
[2:0]	保留	保留

6.14 Flash 内存控制器(FMC)

6.14.1 概述

NuMicro M051™ 系列具有64K/32K/16K/8K字节的片上FLASH EPROM，用于存储应用程序（APROM）,用户可以通过ISP/IAP更新FLASH中的程序. 在系统编程 (ISP) 允许用户更新焊接在PCB板上的芯片中的程序。上电后，通过设置Config0的启动选择位（CBS）决定Cortex-M0 CPU从APROM还是LDROM读取代码. 此外，NuMicro M051™系列为用户提供额外的4k字节的数据FLASH，以供用户在64/32/16/8K字节APROM的M051™系列芯片上系统掉电之前存储数据。

6.14.2 特性

- 高达50MHz的零等待连续地址读访问
- 64/32/16/8KB 应用程序存储空间(APROM)
- 4kB 在系统编程 (ISP) 空间(LDROM)
- 固定的4kB数据FLASH，带有512字节页擦除单位
- 在系统编程(ISP)/在应用编程(IAP)更新片上Flash EPROM

6.14.3 FMC 框图

FLASH存储器控制器由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组成. FLASH存储器控制器框图如图6.13.3-1所示:

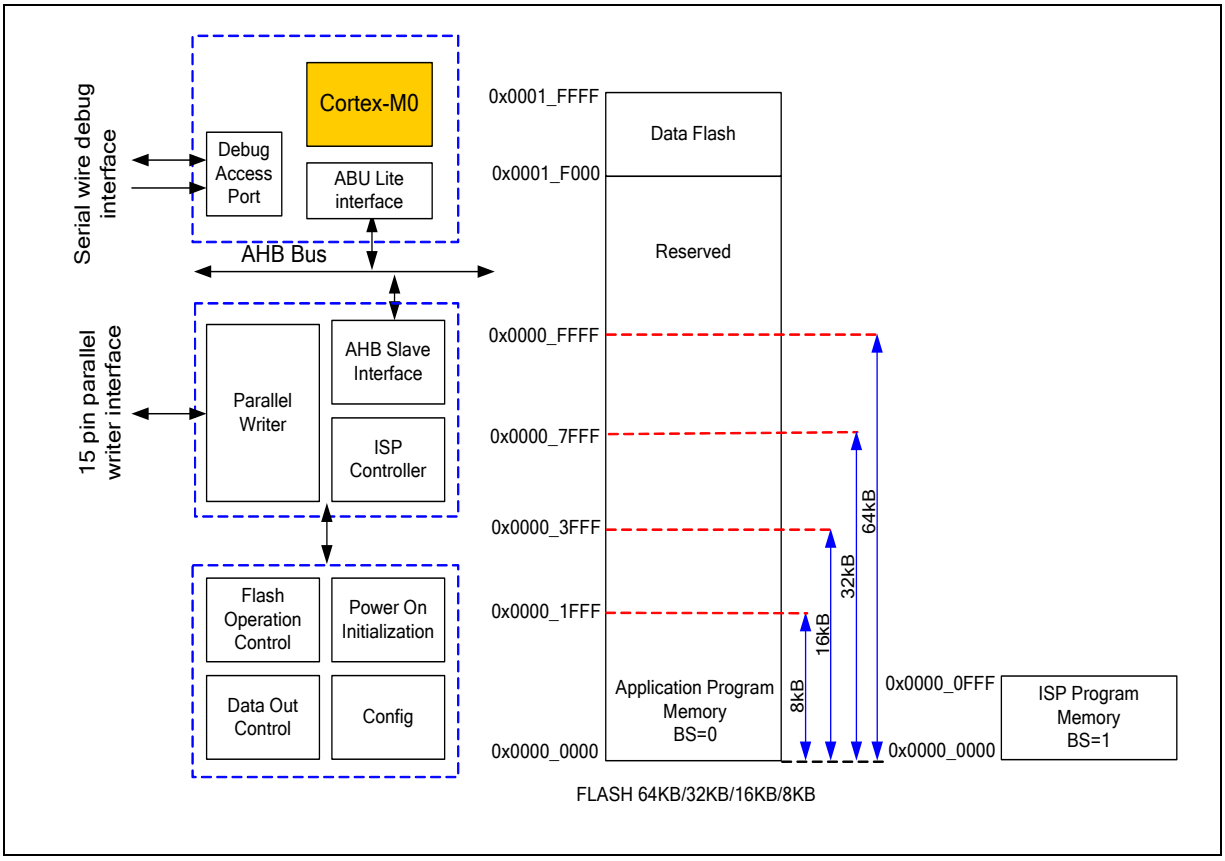


图 6.14.3-1 Flash 存储器控制器框图

6.14.4 FMC 组织结构

NuMicro M051™ 的flash存储器由程序存储器(64/32/16/8KB)，数据FLASH，ISP加载程序存储器，和用户配置区组成. 用户配置区提供几个字节来控制系统逻辑，如flash安全加密，启动选择，欠压电平等。用户配置块的作用类似保险丝用于上电时的缺省设定。在上电期间，从FLASH存储器被加载到相应的控制寄存器中，用户可根据应用需求在芯片贴到PCB板上之前通过烧写器设置这些位，或者在芯片已经贴到板子上之后通过ICP/ISP来修改。数据FLASH的开始地址和大小可由用户根据应用配置，但是对于64/32/16/8KB的FLASH存储器设备，其大小为4KB，开始地址为0x0001_F000.

区块名称	大小	开始地址	结束地址
AP-ROM	8/16/32/64KB	0x0000_0000	0x0000_1FFF (8KB) 0x0000_3FFF (16KB) 0x0000_7FFF (32KB) 0x0000_FFFF (64KB)
Data Flash	4KB	0x0001_F000	0x0001_FFFF
LD-ROM	4KB	0x0010_0000	0x0010_0FFF
User Configuration	1 Words	0x0030_0000	0x0030_0000

表 6.14-1 Flash存储器地址映射

Flash存储器组织结构如下所示:

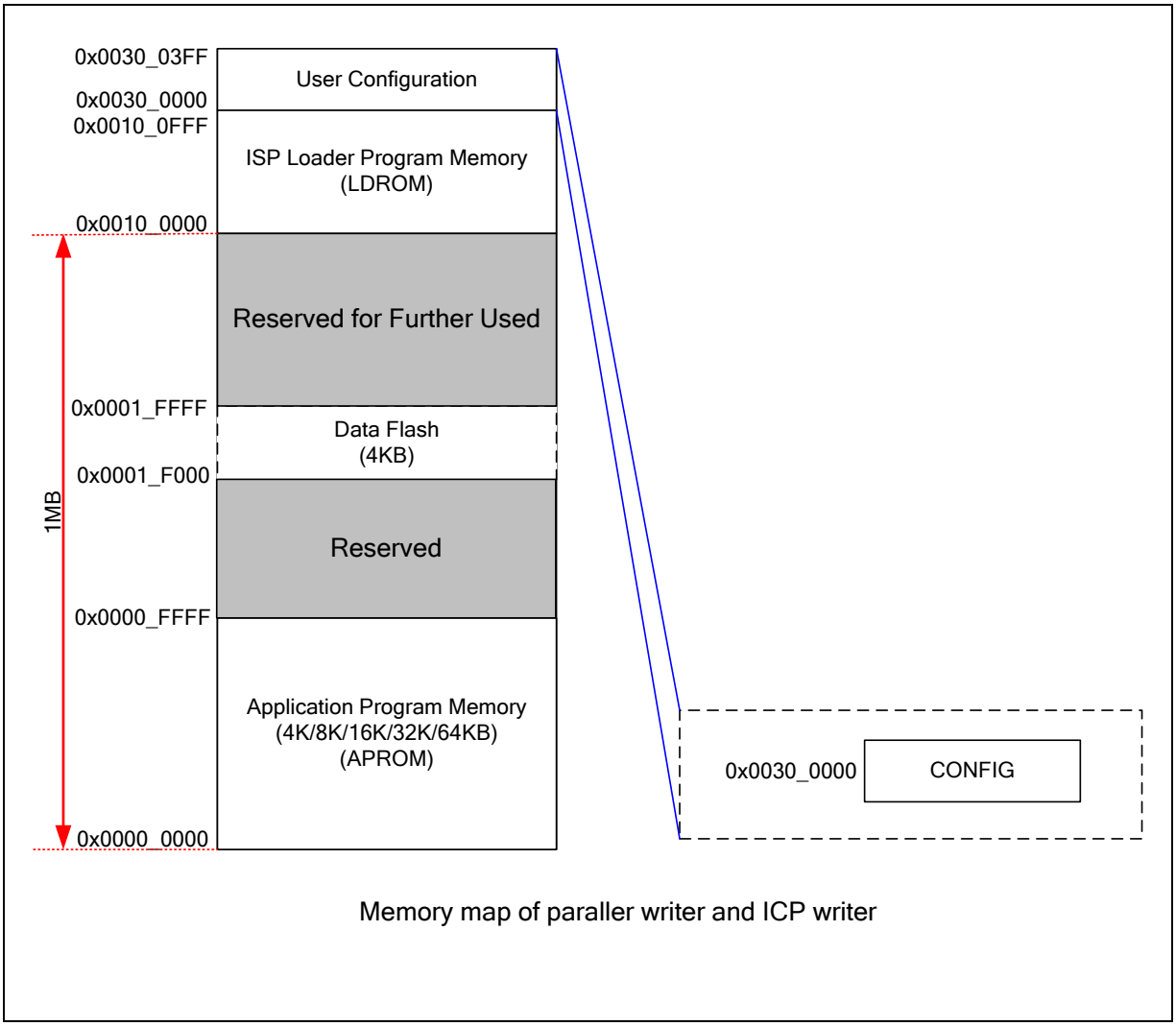


图 6.14.4-1 Flash 存储器组织结构

6.14.5 启动选择

NuMicro M051™提供在系统编程 (ISP) 特性，允许用户直接更新PCB板上芯片中的程序。提供4kB 程序存储区专门用于存储ISP固件. 用户设置Config0的(CBS)以选择从APROM或LDROM启动.

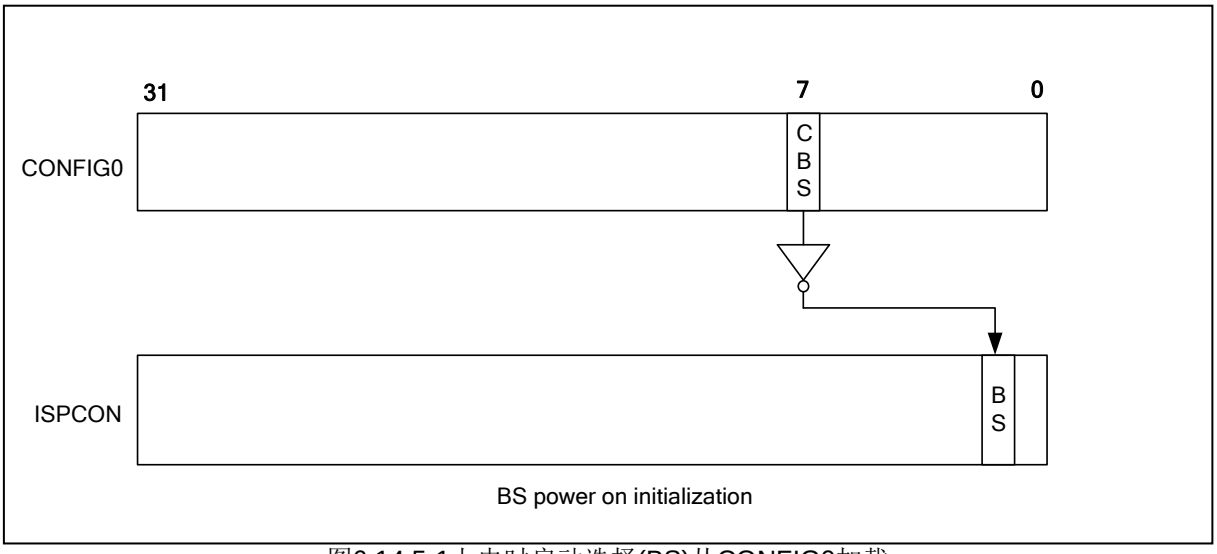


图6.14.5-1上电时启动选择(BS)从CONFIG0加载

6.14.6 Data Flash

NuMicro M051™为用户提供数据FLASH用于存储数据。通过ISP过程读/写. 擦除单位为512字节. 若要改变一个字，需要先把所有128个字拷贝到另外页或SRAM中. 对于8/16/32/64KB的flash设备，数据FLASH的大小为4KB，开始地址固定在0x0001_F000.

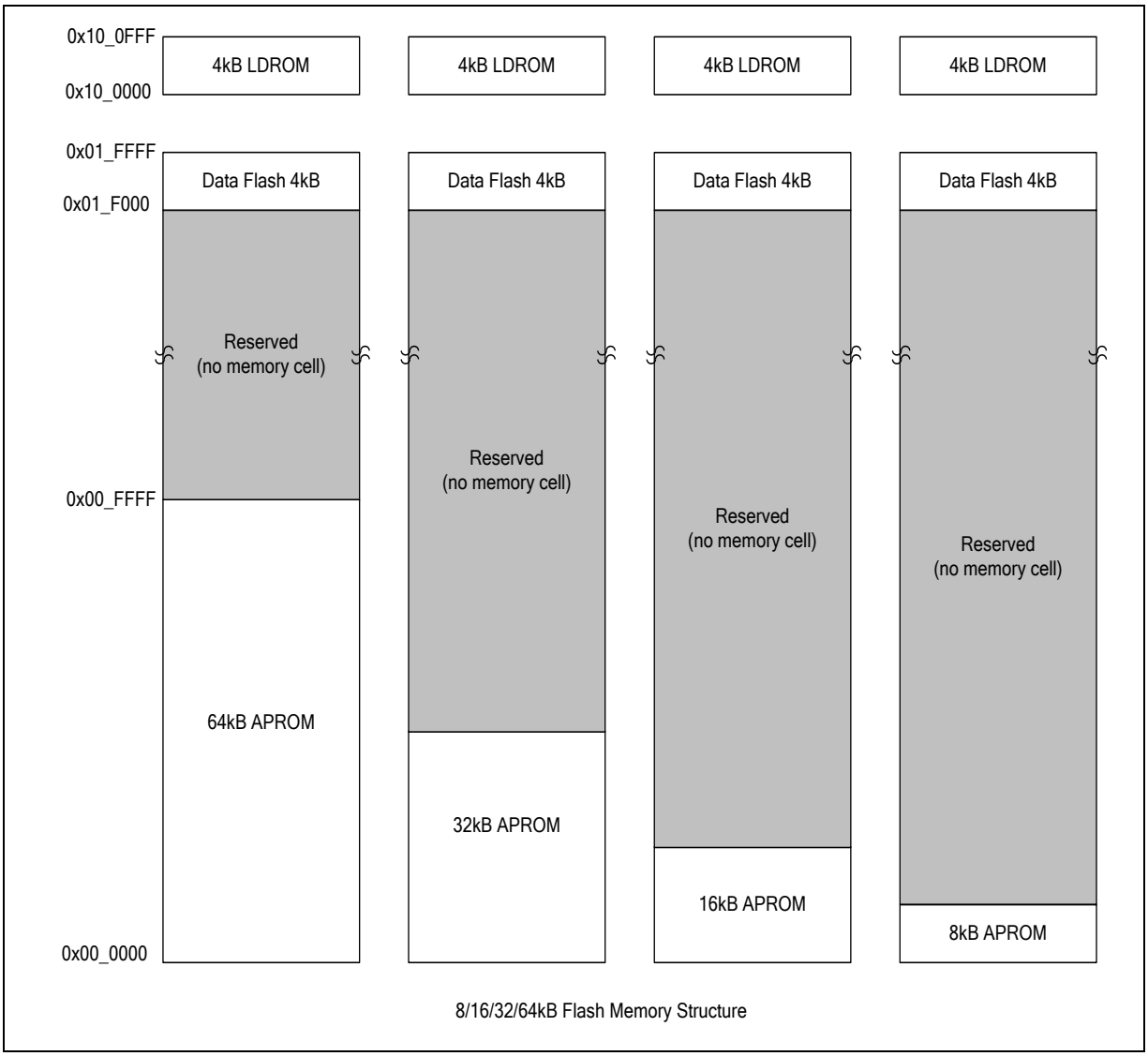


图 6.14.6-1 Flash 存储器结构

6.14.7 在系统编程(ISP)

程序存储区和数据FLASH都支持硬件编程和在系统编程 (ISP). 硬件编程模式在该产品进入批量生产状态时采用gang-writer批量写，以减小编程开销和上市时间。但是，若产品还在开发阶段或终端用户需要升级固件时，硬件编程模式不是很方便，ISP模式能更好地适用于这种情况。NuMicro M051™支持 ISP模式，即通过软件控制来对设备重新编程. 而且，这种更新应用程序固件的能力使得应用更加广泛。

ISP 可以在不用将微控制器从系统中取下来的情况下执行编程. 各种接口使得LDROM更容易更新程序代码. 最常用的方法是通过UART连接到LDROM中的固件来执行ISP，PC一般都是通过串口传输新的APROM代码. LDROM接收后，通过ISP命令，重新对APROM编程。Nuvoton 提供用于NuMicro M051™的ISP 固件和 PC 应用程序. 用户采用Nuvoton ISP工具可以非常方便地执行ISP.

ISP过程

NuMicro M051™ 从APROM还是LDROM启动由用户配置位(CBS)定义。 用户想更新APROM中的应用程序时，可以写BS=1，并开始软件复位使芯片由LDROM启动. 向ISPEN写入1启动ISP功能. 在向ISPCON寄存器写数据之前，S/W 需要向全局控制寄存器（GCR, 0x5000_0100）的REGWRPROT寄存器写入0x59, 0x16 和 0x88，这个过程用于保护FLASH存储器免受意外更改.

启动ISPGO之后，需要检查几个错误条件. 如果错误条件产生，ISP操作失败，其失败标志置位，ISPFF标志由软件清零，而不会在下次ISP操作时被覆盖，即使ISPFF保持为“1“，下一次ISP也可以开始. 建议在每次ISP操作后，通过软件检查ISPFF位，如果ISPFF被被置为1了，就将其清零。.

当ISPGO置位时，CPU将等待ISP操作结束，在此期间，外设仍然正常工作，如果有中断请求时，CPU仍然会先执行完ISP后再响应中断.

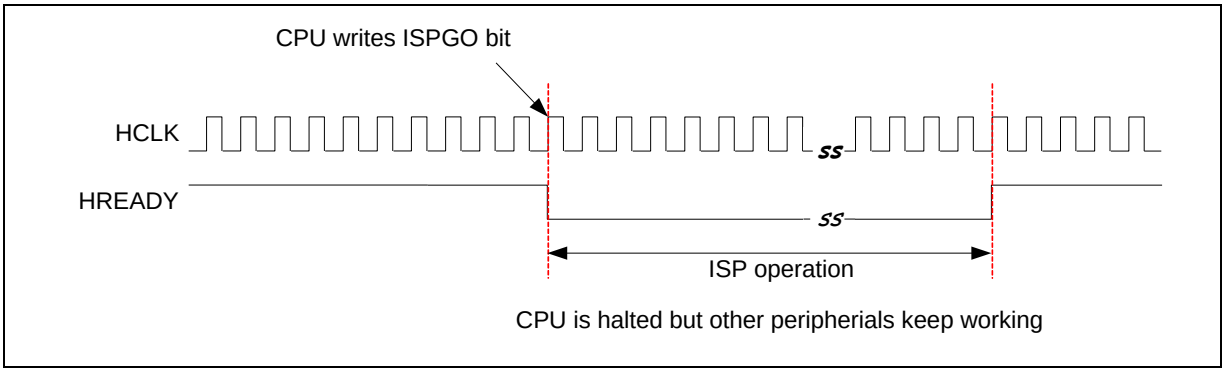


图6.14.7-1 ISPGo 时序图

注：NuMicro M051™允许用户通过ISP更新CONFIG的值, 但是基于对应用程序的安全考虑，软件在擦除CONFIG时，要先页擦除APROM，否则CONFIG不能被擦除.

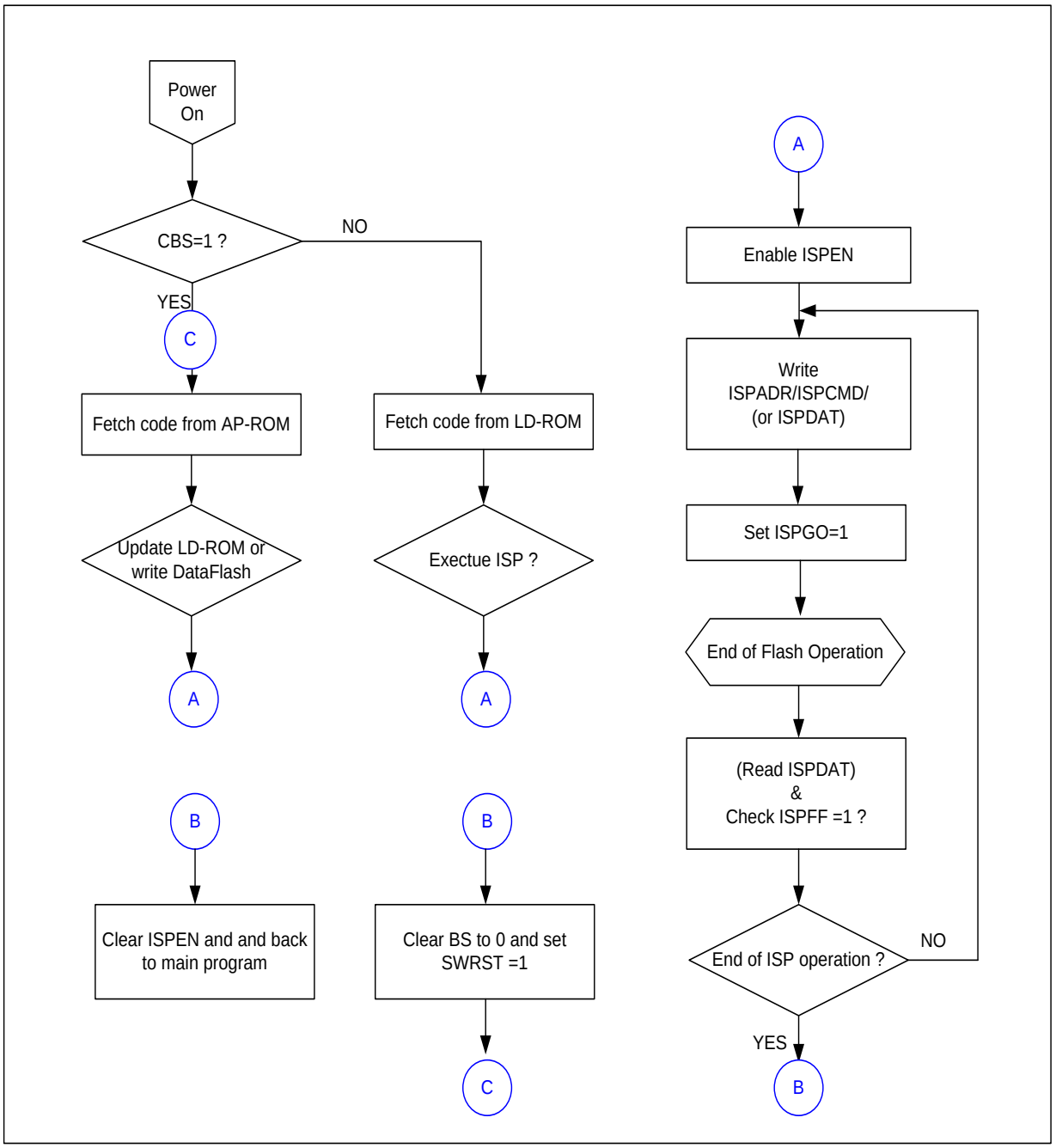


图6.14.7-2 ISP 软件编程流程

ISP Mode	ISPCMD			ISPADR			ISPDAT
	FOEN	FCEN	FCTRL[3:0]	A21	A20	A[19:0]	D[31:0]
Read Company ID	0	0	1011	x	x	x	Data out D[31:0] = 0x0000_00DA
Read Unique ID	0	0	0100	x	x	Address in A[19:0] = 0x00000 0x00004 0x00008	Data out D[31:0] = Unique ID
FLASH Page Erase	1	0	0010	0	A20[#1]	Address in A[19:0]	x
FLASH Program	1	0	0001	0	A20[#1]	Address in A[19:0]	Data in D[31:0]
FLASH Read	0	0	0000	0	A20[#1]	Address in A[19:0]	Data out D[31:0]
CONFIG Page Erase	1	0	0010	1	1	Address in A[19:0]	x
CONFIG Program	1	0	0001	1	1	Address in A[19:0]	Data in D[31:0]
CONFIG Read	0	0	0000	1	1	Address in A[19:0]	Data out D[31:0]

表 6.14-2 ISP模式

注1: A20=0 对应于 APROM和DATA, A20=1, 对应于 LDROM

6.14.8 FMC 控制寄存器映射

R: 只读, W: 只写, R/W: 可读写

寄存器	偏移量	R/W	描述	复位后的值
基地址 (FMC_BA) : 0x5000_C000				
ISPCON	FMC_BA+0x000	R/W	ISP控制寄存器	0x0000_0000
ISPADR	FMC_BA+0x004	R/W	ISP地址寄存器	0x0000_0000
ISPDAT	FMC_BA+0x008	R/W	ISP数据寄存器	0x0000_0000
ISPCMD	FMC_BA+0x00C	R/W	ISP命令寄存器	0x0000_0000
ISPTRG	FMC_BA+0x010	R/W	ISP触发寄存器	0x0000_0000
DFBADR	FMC_BA+0x014	R	数据Flash 启始地址	0x0000_0000 0x0001_F000
FATCON	FMC_BA+0x018	R/W	FLASH访问时间控制寄存器	0x0000_0000

6.14.9 FMC 控制器寄存器描述

ISP 控制寄存器(ISPCON)

寄存器	偏移量	R/W	描述	复位后的值
ISPCON	FMC_BA+0x00	R/W	ISP 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	ISPFF	LDUEN	CFGUEN	APUEN	保留	BS	ISPEN

Bits	描述	
[31:7]	保留	保留
[6]	ISPFF	**ISP失败标志（写保护）** 当ISP满足下列条件时，该位由硬件置位: (1) APUEN 等于0时，APROM对自身写入. (2) LDROM对自身写入. (3) 目标地址无效，如超过正常范围. 注：该bit写 1 清零.
[5]	LDUEN	**LDROM更新使能（写保护）** LDROM 更新使能位. 1 = MCU在APROM中运行时，LDROM可以被更新. 0 = LDROM不能被更新
[4]	CFGUEN	**配置区更新使能（写保护）** 写1使能S/W通过ISP更新配置位，不管此时程序是运行在APROM还是LDROM. 1 = 使能配置区更新

Bits	描述	
		0 = 禁止配置区更新
[2]	**保留**	保留
[1]	**BS**	**启动选择（写保护）** 该位为保护位，置位/清零该位选择下次复位时是由LDROM启动还是由APROM启动，该位可作为MCU启动状态标志,用于检查MCU是由LDROM还是APROM启动的. **上电复位后(POR或者CHIP复位)，该位初始值为config0的CBS的值取反; 其他复位时该值保持不变。** 1 = 由LDROM启动 0 = 由 APROM启动
[0]	**ISPEN**	**ISP 使能（写保护）** 该位是保护位，ISP 使能位，设置该位可以使能ISP功能. 1 = 使能 ISP 功能 0 = 禁用 ISP 功能

ISP 地址 (ISPADR)

寄存器	偏移量	R/W	描述	复位后的值
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** NuMicro M051™系列内置16k x 32 的flash, 仅支持字编程. 执行ISP功能时，ISPARD[1:0] 必须为00b.

ISP 数据寄存器(ISPDAT)

寄存器	偏移量	R/W	描述	复位后的值
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 (ISPCMD)

寄存器	偏移量	R/W	描述	复位后的值
ISPCMD	FMC_BA+ 0x0C	R/W	ISP命令寄存器	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		**FOEN**	**FCEN**	**FCTRL3**	**FCTRL2**	**FCTRL1**	**FCTRL0**

Bits	描述	
[31:6]	保留	保留
[5:0]	**FOEN, FCEN, FCTRL**	**ISP 命令** ISP命令表如下： 操作模式 / FOEN / FCEN / FCTRL[3:0] 读: 0 0 0 0 0 0 读UID: 0 0 0 1 0 0 编程: 1 0 0 0 0 1 页擦除: 1 0 0 0 1 0

ISP触发控制寄存器(ISPTRG)

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

31	30	29	28	27	26	25	24
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							ISPGO

Bits	描述	
[31:1]	保留	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. 1 = ISP 正在执行 0 = ISP 操作结束

数据FLASH基地址寄存器(DFBADR)

寄存器	地址	R/W/C	描述	复位后的值
DFBADR	FMC_BA+ 0x14	R	数据FLASH基地址	0x0001_F000

31	30	29	28	27	26	25	24
DFBA[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开始地址寄存器,只读. 对于 8/16/32/64KB flash 器件，数据flash的大小为4KB ，启始地址固定为0x0001_F000.

Flash访问时间控制寄存器 (FATCON)

寄存器	偏移量	R/W	描述	复位后的值
FATCON	FMC_BA + 0x18	R/W	Flash访问时间控制寄存器	0x0000_0080

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			LFOM	保留			

Bits	描述	
[31:8]	保留	保留
[7]	保留	总是为1
[6:5]	保留	保留
[4]	**LFOM**	低频优化模式(写保护位) 当芯片操作频率低于25MHz时，通过设置该位，系统可以工作的更有效率 1 = 使能flash低频优化模式 0 = 禁用flash低频优化模式
[3:0]	保留	保留

7 用户 配置区

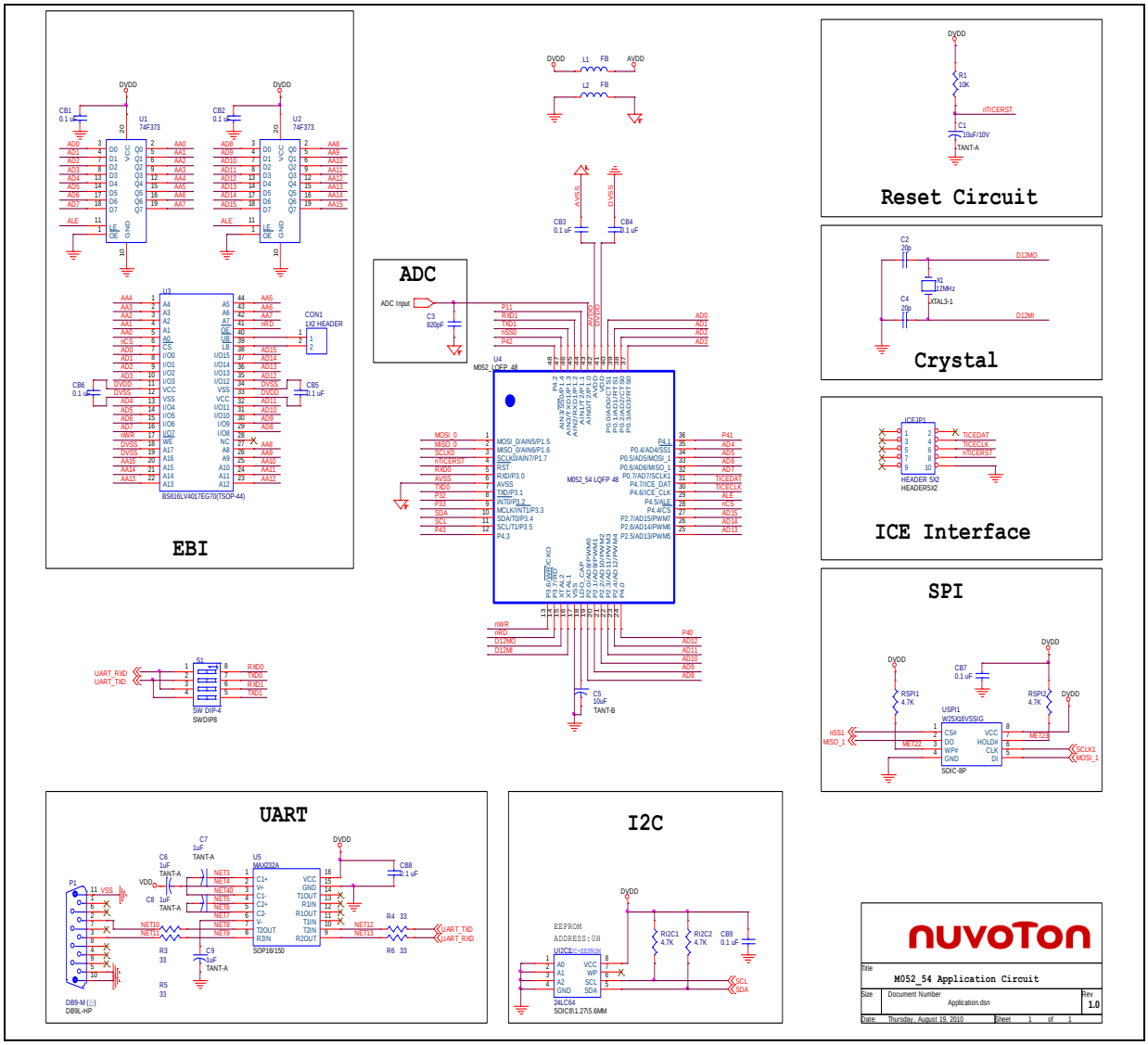
CONFIG (地址= 0x0030_0000)

31	30	29	28	27	26	25	24
保留			CKF	保留	CFOSC		
23	22	21	20	19	18	17	16
CBODEN	CBOV1	CBOV0	CBORST	保留			
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
CBS	保留					LOCK	保留

Bits	描述	
[31:29]	保留	保留
[28]	CKF	**XT1时钟滤波器使能** 0 = 禁用XT1时钟滤波器 1 = 使能 XT1时钟滤波器
[27]	保留	保留
[26:24]	CFOSC	**复位后CPU 时钟源选择** FOSC[2:0] — 时钟源 000 — 外部晶振时钟（4 ~ 24MHz） 111 — 内部 RC 22.1184 MHz 振荡器时钟 其他 — 保留 复位发生后， CFOSC 的值将被加载到CLKSEL0.HCLK_S[2:0].
[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]	CBOV1-0	**欠压电压选择** CBOV1 — CBOV0 — 欠压电压 1 — 1 — 4.5V

Bits	描述	
		1 0 3.8V 0 1 2.7V 0 0 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
[19:8]	**保留**	保留
[7]	**CBS**	**芯片启动选择** 0 = 芯片从LDROM启动 1 = 芯片从APROM启动
[6:2]	**保留**	保留
[1]	**LOCK**	**安全锁** 0 = Flash 数据锁定 1 = Flash 数据不锁定. 当锁定了flash数据, 仅有器件ID, Config0 和 Config1 可以通过烧录器和ICP通过串行调试接口读出。其他数据锁定在0xFFFFFFFF。ISP 可以不管LOCK是否锁定都能读出数据.
[0]	**保留**	保留

8 典型应用电路


EBI
ADC
Reset Circuit
Crystal
ICE Interface
SPI
UART
I2C
M052_54 Application Circuit

9 电气特性

9.1 绝对最大额定值

参数	符号	最小值	最大值	单位
直流电源电压	VDD–VSS	-0.3	+7.0	V
输入电压	VIN	VSS-0.3	VDD+0.3	V
晶振频率	$1/t_{CLCL}$	0	40	MHz
工作温度	TA	-40	+85	°C
贮存温度	TST	-55	+150	°C
VDD最大流入电流		-	120	mA
VSS最大流出电流			120	mA
单一管脚最大灌电流			35	mA
单一管脚最大流出电流			35	mA
所有管脚最大灌电流总和			100	mA
所有管脚最大输出电流总和			100	mA

注: 上表所列的条件中，其极限值可能对设备的稳定有反作用.

9.2 DC 电气特性

(在无特别说明的情况下，VDD-VSS=2.5V~5.5V, TA = 25°C, F_{OSC} = 50Mhz.)

参数	符号.	明细表				测试条件
		最小值	典型值	最大值	单位	
工作电压	V_{DD}	2.5		5.5	V	VDD =2.5V ~ 5.5V ，最高可达50 MHz
LDO 输出电压	V_{LDO}	1.7	1.8	1.9	V	V_{DD} ≥ 2.5V
带隙电压	V_{BG}	-5%	1.20	+5%	V	V_{DD} =2.5V ~ 5.5V
模拟操作电压	AV_{DD}	0		V_{DD}	V	
模拟参考电压	Vref	0		AV_{DD}	V	
正常运行模式下的工作电流@ 50Mhz	IDD1		20.6		mA	VDD= 5.5V@50 MHz, 使能所有的IP和PLL XTAL=12 MHz
	IDD2		14.4		mA	VDD=5.5V@50 MHz, 禁用所有的IP，使能PLL XTAL=12 MHz
	IDD3		18.9		mA	VDD = 3.3V@50 MHz, 使能所有的IP和PLL XTAL=12 MHz
	IDD4		12.8		mA	VDD = 3.3V@50 MHz, 禁用所有的IP，使能PLL XTAL=12 MHz
正常运行模式下的工作电流@ 22Mhz	IDD5		6.2		mA	VDD = 5.5V@ 22MHz, 使能所有的IP和内部22M，禁用PLL
	IDD6		3.4		mA	VDD = 5.5V@22 MHz, 禁用所有的IP和PLL, 使能内部22M
	IDD7		6.1		mA	VDD = 3.3V@22 MHz, 使能所有的IP和内部22M，禁用PLL
	IDD8		3.4		mA	VDD = 3.3V@22 MHz, 禁用所有的IP和PLL, 使能内部22M

参数	符号.	明细表				测试条件
		最小值	典型值	最大值	单位	
正常运行模式下的工作电流@ 12Mhz	IDD9		5.3		mA	V_{DD} = 5.5V@12MHz, 使能所有IP，关闭PLL, XTAL=12MHz
	IDD10		3.7		mA	V_{DD} = 5.5V@12MHz, 关闭所有IP 和PLL, XTAL=12MHz
	IDD11		4.0		mA	V_{DD} = 3.3V@12MHz, 使能所有IP，关闭 PLL, XTAL=12MHz
	IDD12		2.3		mA	V_{DD} = 3.3V@12MHz, 关闭所有IP和 PLL, XTAL=12MHz
正常运行模式下的工作电流@ 4Mhz	IDD13		3.4		mA	VDD = 5.5V@4 MHz, 使能所有的IP，禁用PLL, XTAL=4MHz
	IDD14		2.6		mA	VDD = 5.5V@4 MHz, 禁用所有的IP和PLL, XTAL=4MHz
	IDD15		2.0		mA	VDD = 3.3V@4 MHz, 使能所有的IP，禁用PLL, XTAL=4MHz
	IDD16		1.3		mA	VDD = 3.3V@4 MHz, 禁用所有的IP和PLL, XTAL=4 MHz
正常运行模式下的工作电流@10Khz	IDD17		98.7		uA	V_{DD} = 5.5V@10KHz, 使能所有的IP和内部10K, 禁用 PLL
	IDD18		97.4		uA	V_{DD} = 5.5V@10KHz, 禁用所有IP 和PLL，使能内部10K
	IDD19		86.4		uA	V_{DD} = 3.3V@10KHz, 使能所有的IP和内部10K, 禁用 PLL
	IDD20		85.2		uA	V_{DD} = 3.3V@10KHz, 禁用所有IP 和PLL，使能内部10K
空闲模式下的工作电流@ 50Mhz	IIDLE1		16.2		mA	VDD= 5.5V@50 MHz, 使能所有的IP和PLL XTAL=12 MHz
	IIDLE2		10.0		mA	VDD=5.5V@50 MHz, 禁用所有的IP，使能PLL XTAL=12MHz

参数	符号.	明细表				测试条件
		最小值	典型值	最大值	单位	
	IIDLE3		14.6		mA	VDD = 3V@50 MHz, 使能所有的IP和PLL XTAL=12 MHz
	IIDLE4		8.5		mA	VDD = 3V@50 MHz, 禁用所有的IP，使能PLL XTAL=12 MHz
空闲模式下的工作电流@22Mhz	IIDLE5		4.3		mA	V_{DD} = 5.5V@22MHz, enable all IP and IRC22M, disable PLL
	IIDLE6		1.5		mA	V_{DD} =5.5V@22MHz, disable all IP and enable IRC22M, disable PLL
	IIDLE7		4.2		mA	V_{DD} = 3.3V@22MHz, enable all IP and IRC22M, disable PLL
	IIDLE8		1.4		mA	V_{DD} = 3.3V@22MHz, disable all IP and enable IRC22M, disable PLL
空闲模式下的工作电流@12Mhz	IIDLE9		4.3		mA	VDD = 5.5V@12 MHz, 使能所有的IP，禁用PLL, XTAL=12 MHz
	IIDLE10		2.6		mA	VDD = 5.5V@12 MHz, 禁用所有的IP和PLL, XTAL=12 MHz
	IIDLE11		2.9		mA	VDD = 3V@12 MHz, 使能所有的IP，禁用PLL, XTAL=12 MHz
	IIDLE12		1.3		mA	VDD = 3V@12 MHz, 禁用所有的IP和PLL, XTAL=12 MHz
空闲模式下的工作电流@4Mhz	IIDLE13		3.0		mA	VDD = 5.5V@4 MHz, 使能所有的IP，禁用PLL, XTAL=4 MHz
	IIDLE14		2.3		mA	VDD = 5.5V@4MHz, 禁用所有的IP和PLL, XTAL=4 MHz
	IIDLE15		1.7		mA	VDD = 3.3V@4 MHz, 使能所有的IP，禁用PLL, XTAL=4 MHz
	IIDLE16		1.0		mA	VDD = 3.3V@4 MHz, 禁用所有的IP和PLL, XTAL=4 MHz

参数	符号.	明细表				测试条件
		最小值	典型值	最大值	单位	
空闲模式下的工作电流 @10Khz	IIDLE17		97.8		uA	V_{DD} = 5.5V@10KHz, 使能所有IP 和内部10K, 禁止 PLL
	IIDLE18		96.5		uA	V_{DD} = 5.5V@10KHz, 关闭所有IP ， 使能内部10K晶振, 禁止 PLL
	IIDLE19		85.5		uA	V_{DD} = 3.3V@10KHz, 使能所有IP 和内部10K, 禁止 PLL
	IIDLE20		84.4		uA	V_{DD} = 3.3V@10KHz, 关闭所有IP ， 使能内部10K晶振, 禁止 PLL
掉电模式下待机电流（深度睡眠模式）	IPWD1		10		μA	VDD = 5.5V, 无负载 @禁用BOV 功能
	IPWD2		10		μA	VDD = 3.0V, 无负载 @禁用BOV 功能
P0/1/2/3/4输入电流（准双向模式）	IIN1	-75	-	+15	μA	V_{DD} = 5.5V, VIN = 0V or VIN= V_{DD}
P0/1/2/3/4输入漏电流	ILK	-1	-	+1	μA	V_{DD} = 5.5V, 0<VIN< V_{DD}
P0/1/2/3/4输入低电压 (TTL 输入)	VIL1	-0.3	-	0.8	V	VDD = 4.5V
		-0.3	-	0.6		VDD = 2.5V
P0/1/2/3/4输入高电压 (TTL 输入)	VIH1	2.0	-	V_{DD} +0.2	V	VDD = 5.5V
		1.5	-	V_{DD} +0.2		VDD =3.0V
输入低电压XT1[*2]	VIL3	0	-	0.8	V	VDD = 4.5V
		0	-	0.4		VDD = 2.5V
输入高电压XT1[*2]	VIH3	3.5	-	V_{DD} +0.2	V	VDD = 5.5V
		2.4	-	V_{DD} +0.2		VDD = 3.0V
/RESET脚 负向门槛电压 (Schmitt输入)	VILS	-0.5	-	0.2V_{DD}	V	
RESET脚 正向门槛电压 (Schmitt输入)	VIHS	0.7V_{DD}	-	V_{DD}+0.5	V	
/RST 脚内部上拉电阻	RRST	40		150	KΩ	
P0/1/2/3/4负向门槛电压 (Schmitt输入)	VILS	-0.5	-	0.3V_{DD}	V	
P0/1/2/3/4正向门槛电压	VIHS	0.7V_{DD}	-	V_{DD}	V	

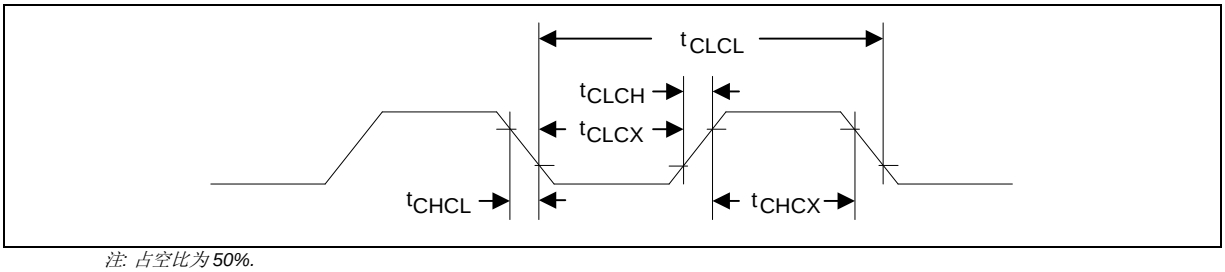
参数	符号.	明细表				测试条件
		最小值	典型值	最大值	单位	
(Schmitt输入)				+0.5		
P0/1/2/3/4 源电流(准双向模式)	ISR11	-300	-370	-450	μA	V_{DD} = 4.5V, VS = 2.4V
	ISR12	-50	-70	-90	μA	V_{DD} = 2.7V, VS = 2.2V
	ISR13	-40	-60	-80	μA	V_{DD} = 2.5V, VS = 2.0V
P0/1/2/3/4源电流(推挽模式)	ISR21	-20	-24	-28	mA	V_{DD} = 4.5V, VS = 2.4V
	ISR22	-4	-6	-8	mA	V_{DD} = 2.7V, VS = 2.2V
	ISR23	-3	-5	-7	mA	V_{DD} = 2.5V, VS = 2.0V
P0/1/2/3/4灌电流(准双向模式和推挽模式)	ISK11	10	16	20	mA	V_{DD} = 4.5V, VS = 0.45V
	ISK12	7	10	13	mA	V_{DD} = 2.7V, VS = 0.45V
	ISK13	6	9	12	mA	V_{DD} = 2.5V, VS = 0.45V
欠压电压 BOV_VL [1:0] =00b	VBO2.2	2.0	2.2	2.4	V	V_{DD} =5.5V
欠压电压 BOV_VL [1:0] =01b	VBO2.7	2.5	2.7	2.9	V	V_{DD} =5.5V
欠压电压 BOV_VL [1:0] =10b	VBO3.8	3.5	3.7	3.9	V	V_{DD} =5.5V
欠压电压 BOV_VL [1:0] =11b	VBO4.5	4.1	4.3	4.5	V	V_{DD} =5.5V
BOD电压迟滞范围	VBH	30	-	150	mV	V_{DD} = 2.5V~5.5V

注意：

1. /RST 脚为史密特触发输入.
2. XTAL1 为CMOS输入.
3. P0, P1, P2, P3 和 P4管脚被外部由1驱动到0时， 可作来输出电流的源端，在V_{DD}=5.5V时, 当Vin 接近2V时，输出电流达到最大值，

9.3 AC 电气特性

9.3.1 外部高速晶振



注: 占空比为 50%.

参数	符号	最小值	典型值	最大值	单位	条件
时钟高电平时间	t_{CHCX}	20	-	-	nS	
时钟低电平时间	t_{CLCX}	20	-	-	nS	
时钟上升沿时间	t_{CLCH}	-	-	10	nS	
时钟下降沿时间	t_{CHCL}	-	-	10	nS	

9.3.2 外部振荡器

参数	条件	最小值	典型值	最大值	单位
输入时钟频率	外部高速晶振	4	12	24	MHz
温度	-	-40	-	85	℃
VDD	-	2.5	5	5.5	V
工作电流	12 MHz@ V_{DD} = 5V	-	1	-	mA

9.3.3 外部高速晶振的典型应用电路

晶振	C1	C2
4 MHz ~ 24 MHz	可选 (取决于晶振规格)	

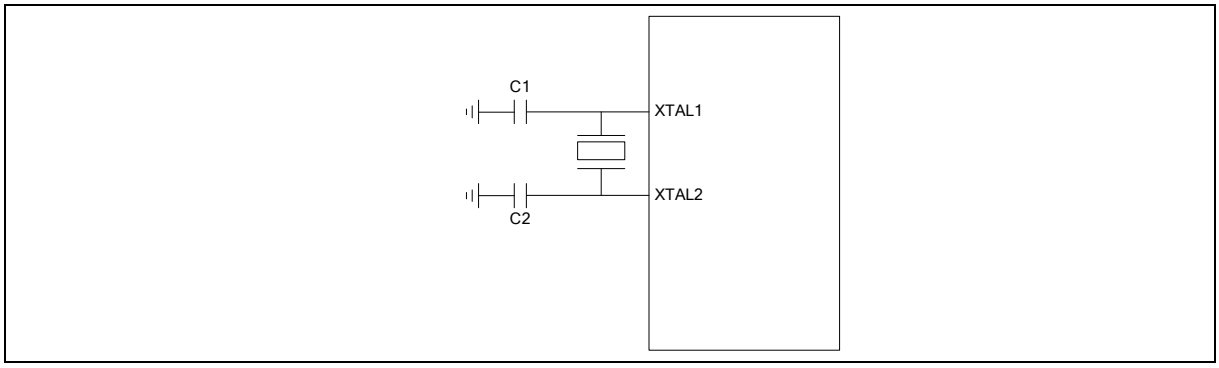


图 9.3.3-1 典型晶振应用电路

9.3.4 内部 22.1184 MHz RC 振荡器

参数	条件	最小值	典型值	最大值	单位
中心频率	-	-	22.1184		MHz
校准之后	+25° C ; V_{DD} =5V	-3	-	+3	%
	-40° C ~+85° C ; V_{DD}=2.5V~5.5V	-5	-	+5	%
工作电流	V_{DD} =5V	-	500	-	uA

9.3.5 内部 10kHz RC 振荡器

参数	条件	最小值	典型值	最大值	单位
电压[1]	-	2.5	-	5.5	V
中心频率	-	-	10	-	kHz
校准之后	+25° C ; V_{DD} =5V	-30	-	+30	%
	-40° C ~+85° C ; V_{DD}=2.5V~5.5V	-50	-	+50	%
工作电流	V_{DD} =5V	-	5	-	uA

注:

1. 内部的工作电压来自LDO.

9.4 模拟量特性

9.4.1 12-bit SARADC 规格

参数	符号	最小值	典型值	最大值	单位
分辨率	-	-	-	12	Bit
非线性差分误差	DNL	-	**±1.2**	-	LSB
非线性整型误差	INL	-	**±1.2**	-	LSB
偏移误差	EO	-	**+3**	+5	LSB
增益误差 (传输增益)	EG	-	**-4**	-6	-
一致性	-		保证		-
ADC 时钟频率	FADC	-	-	16	MHz
转换时间	TADC	-	13	-	Clock
采样率	FS	-	-	760	K SPS
工作电压	V_{LDO}	-	1.8	-	V
	VADD	**3**	**-**	**5.5**	V
工作电流(平均)	IDD	-	0.5	-	mA
	IDDA	-	1.5	-	mA
输入电压范围	VIN	0	-	AVDD	V
电容	CIN	-	5	-	pF

9.4.2 LDO 规格 & Power 管理

参数	最小值	典型值	最大值	单位	备注
输入电压	2.5	5	5.5	V	V_{DD} 输入电压
输出电压	-10%	1.8	+10%	V	LDO输出电压
温度	-40	25	+85	℃	
C	-	1u	-	F	Resr=1ohm

注:

1、建议接一颗100nF 旁路电容在VDD引脚与最近的VSS引脚之间.

2、为保证电源稳定，要在LDO与最近的VSS之间接一颗1uF 或更大的电容.

9.4.3 低压复位规格

参数	条件	最小值	典型值	最大值	单位
操作电压	-	2.5	5	5.5	V
温度	-	-40	25	85	℃
静态电流	VDD5V=5.5V	-	-	5	uA
极限电压	温度=25°C	1.7	2.0	2.3	V
	温度=-40°C	-	2.3	-	V
	温度=85°C	-	1.8	-	V
迟滞	-	0	0	0	V

9.4.4 欠压检测规格

参数	条件	最小值	典型值	最大值	单位
操作电压	-	2.5	-	5.5	V
静态电流	AVDD=5.5V	-	-	140	μA
温度	-	-40	25	85	℃
欠压电压	BOV_VL[1:0]=11	4.1	4.3	4.5	V
	BOV_VL [1:0]=10	3.5	3.7	3.9	V
	BOV_VL [1:0]=01	2.5	2.7	2.9	V
	BOV_VL [1:0]=00	2.0	2.2	2.4	V
迟滞	-	30m	-	150m	V

9.4.5 上电复位规格(5V)

参数	条件	最小值	典型值	最大值	单位
温度		-40	25	85	℃
复位电压	V+	-	2	-	V
静态电流	Vin>复位电压	-	1	-	nA

9.4.6 温度传感器规格

参数	条件	最小值.	典型值.	最大值.	单位
工作电压[1]		1.62	1.8	1.98	V
温度		-40	-	85	℃
增益		-1.72	-1.76	-1.80	mV/℃
偏移	Temp=0 ℃	717	725	733	mV

Note[1]: Internal operation voltage comes from LDO.

9.4.7 比较器规格

参数	条件	最小值.	典型值.	最大值.	单位
温度	-	-40	25	85	℃
V_{DD}	-	2.4	3	5.5	V
V_{DD} 电流	-	-	40	80	uA
输入偏移电压	-		10	20	mV
输出摆幅	-	0.1	-	V_{DD} -0.1	V
输入范围	-	0.1	-	V_{DD} -0.1	V
DC 增益	-	-	70	-	dB
传播延迟	@VCM=1.2 V and VDIFF=0.1 V	-	200	-	ns
迟滞	@VCM=0.2 V ~ V_{DD} -0.2V	-	±10	-	mV
稳定时间	@CINP=1.3 V CINN=1.2 V	-	-	2	us

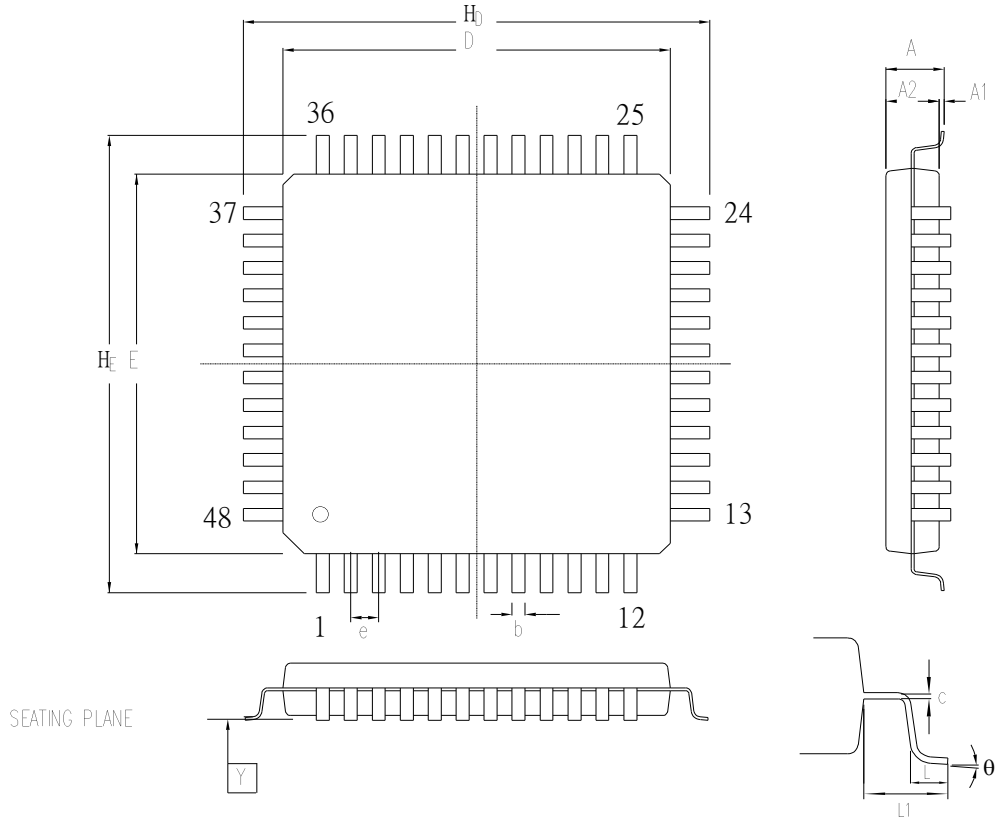
9.5 Flash DC 电器特性

符号	参数	条件	最小值.	典型值.	最大值.	单位
N_{endu}	Endurance		100000			cycles[1]
T_{ret}	保存时间	Temp=85 ℃	10			year
T_{erase}	页擦除时间		19	20	21	ms
T_{mess}	Mess erase time		30	40	50	ms
T_{prog}	编程时间		38	40	42	us
V_{DD}	工作电压		1.62	1.8	1.98	V[2]
I_{dd1}	读电流				0.25	mA
I_{dd2}	编程/擦除电流				7	mA
I_{pd}	掉电模式电流			1	20	uA

1. 编程/擦除周期数.
2. V_{DD} 是芯片 LDO 的输出电压源.
3. Guaranteed by design, not test in production

10 封装尺寸

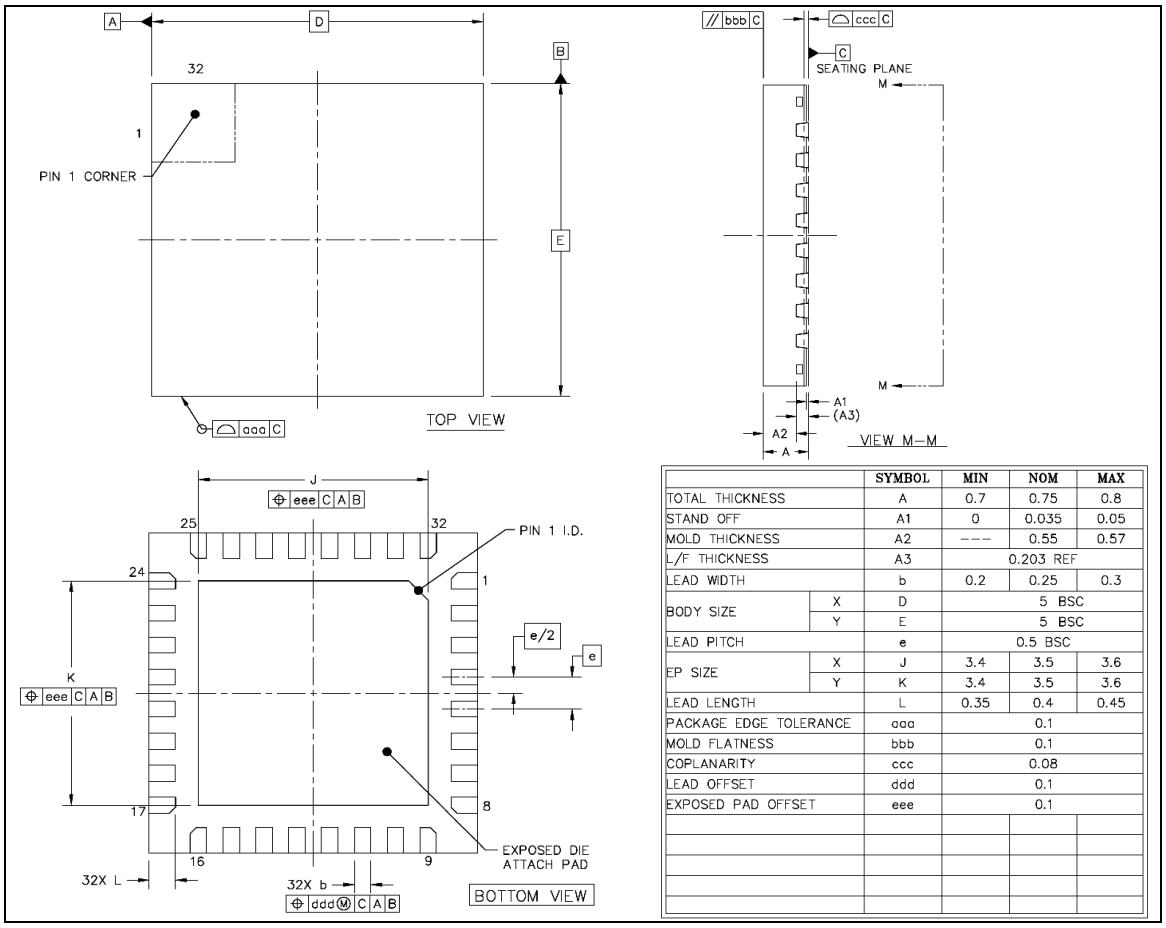
10.1 LQFP-48 (7x7x1.4mm^2 Footprint 2.0mm)



Controlling dimension : Millimeters

Symbol	Dimension in inch			Dimension in mm		
	Min	Nom	Max	Min	Nom	Max
A	—	—	—	—	—	—
A_1	0.002	0.004	0.006	0.05	0.10	0.15
A_2	0.053	0.055	0.057	1.35	1.40	1.45
b	0.006	0.008	0.010	0.15	0.20	0.25
c	0.004	0.006	0.008	0.10	0.15	0.20
D	0.272	0.276	0.280	6.90	7.00	7.10
E	0.272	0.276	0.280	6.90	7.00	7.10
e	0.014	0.020	0.026	0.35	0.50	0.65
H_D	0.350	0.354	0.358	8.90	9.00	9.10
H_E	0.350	0.354	0.358	8.90	9.00	9.10
L	0.018	0.024	0.030	0.45	0.60	0.75
L_1	—	0.039	—	—	1.00	—
Y	—	—	0.004	—	—	0.10
θ	0	—	7	0	—	7

10.2 QFN-33 (5X5 mm^2, Thickness 0.8mm, Pitch 0.5 mm)



		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.7	0.75	0.8
STAND OFF		A1	0	0.035	0.05
MOLD THICKNESS		A2	---	0.55	0.57
L/F THICKNESS		A3		0.203 REF	
LEAD WIDTH		b	0.2	0.25	0.3
BODY SIZE	X	D		5 BSC	
	Y	E		5 BSC	
LEAD PITCH		e		0.5 BSC	
EP SIZE	X	J	3.4	3.5	3.6
	Y	K	3.4	3.5	3.6
LEAD LENGTH		L	0.35	0.4	0.45
PACKAGE EDGE TOLERANCE		aaa		0.1	
MOLD FLATNESS		bbb		0.1	
COPLANARITY		ccc		0.08	
LEAD OFFSET		ddd		0.1	
EXPOSED PAD OFFSET		eee		0.1	

11 版本历史

版本	日期	页	描述
V1.03	2012年4月10日,	-	初次发行版本

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.