ARM® Cortex®-M
32位微处理器

NuMicro® Family
NUC505 系列
技术参考手册

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

目录

1 概述 13
2 特性 14
2.1 NUC505 特性 14
3 缩写词 19
3.1 缩写词 19
4 芯片信息和管脚配置 20
4.1 选型指南 20
4.1.1 NuMicro™ NUC505 系列选型指南 20
4.1.2 NuMicro™ NUC505 系列命名规则 21
4.2 管脚配置 22
4.2.1 NuMicro™ NUC505DLA LQFP 48管脚 22
4.2.2 NuMicro™ NUC505DL13Y LQFP 48管脚 23
4.2.3 NuMicro™ NUC505YLA QFN 48管脚 24
4.2.4 NuMicro™ NUC505YLA2Y QFN 48管脚 25
4.2.5 NuMicro™ NUC505DSA LQFP 64管脚 26
4.2.6 NuMicro™ NUC505DS13Y LQFP 64管脚 27
4.2.7 NuMicro™ NUC505YOY13Y QFN 88管脚 28
4.3 管脚描述 29
4.3.1 NuMicro™ NUC505DLA LQFP 48 脚管脚描述 29
4.3.2 NuMicro™ NUC505DL13Y LQFP 48 脚管脚描述 32
4.3.3 NuMicro™ NUC505YLA QFN 48脚管脚描述 38
4.3.4 NuMicro™ NUC505YLA2Y QFN 48脚管脚描述 43
4.3.5 NuMicro™ NUC505DSA LQFP 64脚管脚描述 48
4.3.6 NuMicro™ NUC505DS13Y LQFP 64 脚管脚描述 55
4.3.7 NuMicro™ NUC505YO13Y QFN 88 脚管脚描述 62
4.3.8 GPIO 多功能管脚表 71
4.3.9 GPIO 多功能管脚概述 73
5 框图 77

5.1 NuMicro™ NUC505 功能框图 77
6 功能描述 78
6.1 ARM® Cortex®-M4 内核 78
6.2 系统管理 81
6.2.1 概述 81
6.2.2 系统复位 81
6.2.3 系统上电设置 82
6.2.4 系统电源分配 82
6.2.5 系统内存映射 83
6.2.6 SRAM 内存组织 85
6.2.7 AHB 总线仲裁 87
6.2.8 寄存器映射 89
6.2.9 寄存器描述 90
6.2.10 系统定时器 (Systick) 128
6.2.11 嵌套向量中断控制器(NVIC) 133
6.2.12 系统控制寄存器映射和描述 144
6.3 时钟控制器 153
6.3.1 概述 153
6.3.2 时钟框图 154
6.3.3 时钟发生器 155
6.3.4 掉电模式时钟 156
6.3.5 寄存器映射 157
6.3.6 寄存器描述 158
6.4 通用 I/O (GPIO) 176
6.4.1 概述 176
6.4.2 特性 176
6.4.3 基本配置 177
6.4.4 功能描述 177
6.4.5 寄存器映射 178
6.4.6 寄存器描述 180
6.5 定时器控制器 (TIMER) 192
6.5.1 概述 192
6.5.2 特性 192

6.5.3 框图 193
6.5.4 功能描述 193
6.5.5 寄存器映射 197
6.5.6 寄存器描述 199
6.6 PWM 发生器和捕捉定时器 (PWM) 208
6.6.1 概述 208
6.6.2 **特性** 208
6.6.3 框图 209
6.6.4 **功能描述** 211
6.6.5 **寄存器 映射** 217
6.6.6 **寄存器 描述** 219
6.7 看门狗定时器 (WDT) 238
6.7.1 概述 238
6.7.2 **特性** 238
6.7.3 框图 238
6.7.4 **功能描述** 239
6.7.5 **寄存器 映射** 241
6.7.6 **寄存器 描述** 242
6.8 窗看门狗定时器 (WWDT) 246
6.8.1 概述 246
6.8.2 **特性** 246
6.8.3 框图 246
6.8.4 **功能描述** 246
6.8.5 **寄存器 映射** 249
6.8.6 **寄存器 描述** 250
6.9 实时时钟 (RTC) 256
6.9.1 概述 256
6.9.2 **特性** 256
6.9.3 框图 257
6.9.4 基本配置 257

6.9.5 功能描述 257
6.9.6 系统电源控制流程 261
6.9.7 寄存器 映射 263
6.9.8 寄存器 描述 265
6.10 串口控制器 (UART) 288
6.10.1 概述 288
6.10.2 特性 288
6.10.3 框图 289
6.10.4 基本配置 291
6.10.5 功能 描述 291
6.10.6 寄存器 映射 311
6.10.7 寄存器 描述 313
6.11 I²C 串行接口控制器 (主机/从机) 340
6.11.1 概述 340
6.11.2 特性 341
6.11.3 基本配置 341
6.11.4 功能描述 341
6.11.5 操作模式 344
6.11.6 协议寄存器 351
6.11.7 寄存器 映射 355
6.11.8 寄存器 描述 356
6.12 串行外围设备接口(SPI) 367
6.12.1 概述 367
6.12.2 特性 367
6.12.3 框图 367
6.12.4 功能描述 368
6.12.5 时序图 374
6.12.6 编程示例 377
6.12.7 寄存器映射 379
6.12.8 寄存器描述 380

6.13 SPI 同步串行接口控制器 (主机模式, SPIM) 392
6.13.1 概述 392
6.13.2 特性 392
6.13.3 框图 393
6.13.4 功能描述 393
6.13.5 寄存器映射 398
6.13.6 寄存器描述 399
6.14 带内部音频编解码的I²S 控制器 (I²S) 411
6.14.1 概述 411
6.14.2 特性 411
6.14.3 框图 412
6.14.4 I²S 操作时序图描述 413
6.14.5 FIFO 操作 416
6.14.6 功能描述 416
6.14.7 寄存器映射 424
6.14.8 寄存器描述 425
6.14.9 CODEC 寄存器映射 450
6.14.10 CODEC 寄存器描述 451
6.15 USB 2.0设备控制器(USBD) 466
6.15.1 概述 466
6.15.2 特性 466
6.15.3 框图 467
6.15.4 功能描述 467
6.15.5 寄存器映射 470
6.15.6 寄存器描述 476
6.16 USB 1.1 Host 控制器 (USBH) 530
6.16.1 概述 530
6.16.2 特性 530
6.16.3 框图 531
6.16.4 基本描述 532
6.16.5 功能描述 532
6.16.6 寄存器列表 533
6.16.7 寄存器描述 535

6.17 安全数字主控制器(SDHC) 567
6.17.1 概述 567
6.17.2 特性 567
6.17.3 框图和卡片脚位设定 568
6.17.4 SD 主机 DMA控制器 569
6.17.5 安全数字 SD/SDHC 控制器 570
6.17.6 寄存器列表 572
6.17.7 寄存器描述 573
6.18 12位模数转换器 (ADC) 595
6.18.1 概述 595
6.18.2 特性 595
6.18.3 框图 595
6.18.4 工作描述 596
6.18.5 寄存器列表 599
6.18.6 寄存器描述 600
7 电气特性 604
8 封装尺寸 605
8.1 LQFP 48L (7x7x1.4mm footprint 2.0mm) 605
8.2 QFN 48 (7x7x0.8mm) 606
8.3 LQFP 64L (7x7x1.4mm footprint 2.0mm) 607
8.4 QFN 88 (10x10x0.9mm) 608
9 修订历史 610

图集

图 4.1-1 NuMicro™ NUC505 系列选型代码 21
图 4.2-1 NuMicro™ NUC505DLA LQFP 48管脚图 22
图 4.2-2 NuMicro™ NUC505DL13Y LQFP 48管脚图 23
图 4.2-3 NuMicro™ NUC505YLA QFN 48管脚图 24
图 4.2-4 NuMicro™ NUC505YLA2Y QFN 48管脚图 25
图 4.2-5 NuMicro™ NUC505DSA LQFP 64管脚图 26
图 4.2-6 NuMicro™ NUC505DS13Y LQFP 64管脚图 27
图 4.2-7 NuMicro™ NUC505YO13Y QFN 88 管脚图 28
图 5.1-1 NuMicro™ NUC505功能框图 77
图 6.1-1 Cortex®-M4 模块框图 78
图 6.2-1 NuMicro™ NUC505 电源分布框图 83
图 6.2-2 SRAM 模块图 85
图 6.2-3 SRAM内存组织架构 86
图 6.2-4 矢量表模块框图 87
图 6.3-1时钟发生器全局框图 154
图 6.3-2时钟发生器框图 155
图 6.3-3 晶振振荡电路 156
图 6.4-1 I/O 引脚模块框图 176
图 6.4-2 推挽输出 177
图 6.5-1 定时器控制器框图 193
图 6.5-2 连续计数模式 195
图 6.5-3 外部捕捉模式 196
图 6.5-4 外部复位计数模式 196
图 6.6-1 PWM 发生器 0 框图 209
图 6.6-2 PWM发生器1框图 210
图 6.6-3 PWM 系统时钟源 210
图 6.6-4 PWM 边沿对齐波形图 211
图 6.6-5 PWM 双缓存时序 212
图 6.6-6 PWM 带死区的互补输出 213
图 6.6-7 捕捉操作时序 214
图 6.6-8 PWM中断架构 217

图 6.7-1 看门狗时钟控制图 238
图 6.7-2 看门狗定时器框图 239
图 6.7-3 看门狗中断和复位时序 240
图 6.8-1 窗看门狗定时器框图 246
图 6.8-2窗看门狗定时器的复位和加载过程 248
图 6.9-1 RTC模块框图 257
图 6.9-2 系统电源控制流程图 261
图 6.9-3 电源键侦测 262
图 6.10-1 UART 时钟控制图 289
图 6.10-2 UART模块框图 289
图 6.10-3 自动波特率测量 292
图 6.10-4发送延时操作 293
图 6.10-5 UART nCTS 唤醒1 294
图 6.10-6 UART nCTS 唤醒2 294
图 6.10-7 UART 数据唤醒 294
图 6.10-8自动流控框图 297
图 6.10-9 UART nCTS 自动流控使能 298
图 6.10-10 UART nRTS自动流控使能 299
图 6.10-11 UART nRTS软件方式下的自动流控 299
图 6.10-12 IrDA 控制模块框图 300
图 6.10-13 IrDA TX/RX时序图 301
图 6.10-14 LIN 的帧结构 301
图 6.10-15LIN字节结构 302
图 6.10-16 LIN 帧 ID 和奇偶格式 302
图 6.10-17 LIN模式Break侦测 304
图 6.10-18 LIN同步空间测量 306
图 6.10-19 UART_BAUD AR模式下SLVDUEN=1时UART_BAUD更新顺序 307
图 6.10-20 UART_BAUD AR模式下SLVDUEN=0时UART_BAUD更新顺序 307
图 6.10-21 RS-485自动方向控制模式下nRTS管脚驱动电平 310
图 6.10-22 RS-485软件控制模式下nRTS管脚驱动电平 310
图 6.10-23 RS-485帧结构 311
图 6.11-1 I^2C 总线时序 340

图 6.11-2 I²C 协议 342
图 6.11-3 主机发数据到从机 342
图 6.11-4主机读从机数据 342
图 6.11-5 开始和停止条件 343
图 6.11-6 I²C 总线位传送 343
图 6.11-7 I²C总线应答 344
图 6.11-8通过当前状态控制I²C 总线 345
图 6.11-9主机发送控制流程 345
图 6.11-10 主机接收控制流程 346
图 6.11-11 从机控制流程 347
图 6.11-12 GC模式 348
图 6.11-13 EEPROM 随机读 350
图 6.11-14 随机读EEPROM的协议 350
图 6.11-15 I²C数据移动方向 351
图 6.11-16 I²C 溢出计数器框图 353
图 6.12-1 SPI 模块框图 367
图 6.12-2 SPI 主机模式应用框图 368
图 6.12-3 SPI从机模式应用框图 368
图 6.12-4 一次传输中的位 369
图 6.12-5 字节重排序功能 370
图 6.12-6 字节休眠时序波形 371
图 6.12-7 FIFO 模块框图 372
图 6.12-8 主机模式下的SPI 时序 375
图 6.12-9 主机模式下的SPI 时序(交替SPICLK时钟相位) 375
图 6.12-10 从机模式下的SPI时序 376
图 6.12-11 从机模式下的SPI时序(交替SPICLK时钟相位) 376
图 6.13-1 SPIM 模块框图 393
图 6.13-2 SPIM 时序图 394
图 6.13-3 类型_1 四线模式写的编程命令流程. 396
图 6.13-4 类型_2四线模式写的编程命令流程. 396
图 6.13-5 类型_3四线模式写的编程命令流程. 397
图 6.14-1 用内部音频编解码的I²S 控制器框图 412
图 6.14-2 内部 CODEC 框图 413

图 6.14-3 I²S 总线时序图 414
图 6.14-4 MSB 对齐时序图 414
图 6.14-5 PCM-A 音频时序图 414
图 6.14-6 PCM-B音频时序图 415
图 6.14-7不同I²S 模式下FIFO 的内容 416
图 6.14-8 主机模式接口框图 418
图 6.14-9 从机模式接口框图 418
图 6.14-10 I²S 控制器中断 423
图 6.15-1 USB 设备控制器框图 467
图 6.16-1 USB 1.1 主机控制器模块框图 531
图 6.17-1 SDHC模块框图 568
图 6.17-2 DMA控制器模块框图 569
图 6.17-3 PAD表格格式 576
图 6.18-1 ADC 控制模块框图 596
图 6.18-2 转换时序框图 597
图 6.18-3 按键应用模块图 597
图 6.18-4 ADC 控制器中断 598

表集

表 2.1-1 特性表 13
表 3.1-1 缩写词 19
表 4.1-1 NuMicro™ NUC505 系列选型指南 20
表 4.3-1 NUC505 GPIO 多功能表 76
表 6.2-1 系统上电设置指南 82
表 6.2-2片上控制器地址空间分配 85
表 6.2-3固定优先级模式下AHB总线优先级顺序 88
表 6.2-4 异常模型 134
表 6.2-5 中断号表 135
表 6.2-6优先级分组 148
表 6.3-1 推荐电容值和电阻值 156
表 6.3-2掉电模式控制表 159
表 6.7-1看门狗定时器时间间隔 240
表 6.8-1 窗看门狗定时器预分频值选择 247
表 6.8-2 CMPDAT 设置范围 248
表 6.10-1 UART 接口控制管脚 291
表 6.10-2 UART 控制器波特率计算公式 292
表 6.10-3 UART控制器中断源和标志列表 296
表 6.10-4 UART线控的数据位和停止位长度设置 296
表 6.10-5 UART线控校验设置 297
表 6.10-6 LIN主模式时报头选择 303
表 6.11-1 I^2C状态码描述表 353
表 6.14-1 内部音频CODEC 的时钟参考 421
表 6.14-2 外部音频CODEC 的时钟参考 422
表 6.17-1 SD/SDHC/MMC 脚位设定 568

1 概述

NUC505 是32位ARM® Cortex®-M4F 内核、通用微控制器。

NUC505 工作频率 100 MHz，支持DSP指令和符点运算，嵌入 512K/2 M字节 SPI FLASH，128 K 字节 SRAM。其外设包括：USB Host/Device, Timers, WDT, RTC, UART, SPI, I²S, I²C, PWM Timer, GPIO, 12-bit ADC, 24-bit Audio CODEC, 低电压复位（LVR）Low 和 低电压检测（LVD）。

产品线	USB Host	USB Device	UART	I²C	I²S	SPI	PWM	ADC	Audio CODEC	RTC
NUC505	•	•	•	•	•	•	•	•	•	•

表 2.1-1 特性表

NUC505 适合以下应用:

- 热敏打印机
- GPS 追踪器 / VTDR (车辆路线记录仪)
- 2.4G 无线产品
- 音频应用
- LED 广告

2 特性

2.1 NUC505 特性

●Cortex-M4 内核

- 工作频率 100 MHz
- 有 DSP 指令，和硬件除法器
- 与 IEEE 754 兼容的符点运算单元
- 支持内存管理 (MPU)
- 一个24位系统定时器 system timer
- 支持休眠模式：WFI 和 WFE 指令
- 单周期 32-位乘法运算
- 16 级中断优先等级 (NVIC)
- 中断可软件屏蔽
- 代码可从SPI FLASH中加载，或从USB设备加载

●128K字节 SRAM

- 支持8位，16位，32位访问

●内嵌 512K/2M字节 SPI Flash

- 代码/数据空间自由分配
- 支持SWD/ICE 接口2 线ICP编程
- 支持 ISP 编程
- 支持单线，双线和4线数据访问
- 单线模式支持 100 MHz 数据传输速度
- 2线和4线模式支持 80 MHz数据传输速度

●SPIM

- 通用 SPI 主机接口协议，可外接 SPI Flash
- 支持代码保护
- SPI Flash 传到 SRAM，可由 DMA 传输
- CPU 可直接从SPI Flash 读取数据
- 支持单线，双线和四线传输模式

●时钟

- 内建 32.768 kHz RC 时钟(LIRC) 用于RTC，WDT和唤醒操作
- RTC也可用外接32768HZ晶振时钟源
- 可外接 12 MHz 晶振，PLL可倍频到240MHz

●全速 USB 2.0 host

- 传输速度1.5Mbps，或12Mbps

- 完全兼容USB1.1
- 兼容 OHCI1.0
- 支持控制，批量，中断和等时传输

●高速USB 2.0 device

- 12个端点地址，支持控制，批量，中断和等时传输
- 2K-字节 缓存
- 自动挂起功能
- 远程唤醒功能

●低压检测功能 LVD： 2.8V / 2.6V

●低压复位功能 LVR： 门限 2.4 V

●GPIO

- 两种模式：输入，输出。
- 输入可选 CMOS 和 Schmitt 模式
- 可配置边沿中断
- 5V-容限 (PA.7~PA.0 和 PD.4~PD.2 仅能 3.3 V)
- QFN88/LQFP64/LQFP48 封装，分别有 52/35/25 个GPIO

●4个24位、上计数定时器

- 带8位预分频器
- 每个定时器都有独立时钟源
- 一次定时、周期定时、反转GPIO模式和连续操作四种工作模式
- 支持引脚脉冲计数模式
- 有捕获功能，引脚信号可复位定时器

●看门狗

- 时钟源可选 LIRC(缺省值), HXT 和 LXT
- 8 个溢出时间选择：1.6毫秒 ~ 26.0秒
- 看门狗溢出可选中断或复位功能

●窗看门狗

- 时钟源可选 LIRC(缺省值), HXT 和 LXT
- 6 位计数器，带11位预分频器
- 看门狗溢出可选中断或复位功能

●三个UART：UART0，UART1 和 UART2

- UART0 16-字节 FIFO。UART1，UART2 有 64-字节 FIFO
- UART1,UART2 支持硬件流控 nCTS，nRTS
- 支持红外IrDA模式

- 485模式支持 第9位地址识别模式和方向控制
- UART1，UART2 支持 LIN 功能
- 接收线 Rx 和流控线 nCTS 可把 CPU 从掉电状态唤醒

●两个SPI

- 速度可达50MHz
- 支持主从模式，一位传输模式
- 传输位长： 8 到 32，高低位在前可配置。
- 收发各 8 级 FIFO
- 有字节重排序功能
- 字节和字间加入空闭时钟功能
- 三线传输功能

●两个I^2C

- 支持主模式收发，或从模式收发。
- 支持多主通信和总线仲裁
- 时钟可任意延长，最快1MHz
- 四个从机地址，并带有地址掩码
- 支持 SMBus 和 PMBus
- 支持多地址掉电唤醒功能

●主从I^2S

- 单、双声道可选，收发独立16 级 FiFo。
- 内建 PLL
- 传输位长： 8，16，24 和 32
- 支持 PCMA，PCMB，I^2S 和 MSB 调整格式
- 支持 DMA 传输

●4个PWM

- 支持一次PWM模式，和周期PWM模式
- 支持捕获模式

●SD2.0 Host 接口

- 1位或4位传输模式
- 支持 DMA 传输
- 速度可达 200Mbps@3.3V（50MHz，4位模式）

●RTC

- 外接 32768 晶振，硬件计算年月日时分秒星期闰年
- 有备用电池引脚 VBAT

- 32字节 备用 RAM
- 闹钟功能
- 可唤醒掉电状态的CPU
- 有掉电唤醒引脚
- 寄存器配置进入掉电功能
- 上电超时、保护电池功能

●代码保护功能

- 128-bit 密钥
- 15次错误计数

●音频编解码 Audio CODEC

- 内嵌双声道 24-bit Sigma-Delta CODEC 输出
- ADC-THDN: -80 dB。 信噪比SNR： 90 dB (预加重)
- 耳机输出 THDN：-60dB, 信噪比SNR： 93 dB (预加重)
- 采样率： 8 kHz to 96 kHz

●8路 12-bit SAR ADC

- 输入电压范围： 0~ AV_{DD}
- 保证10位精度
- 采样率：ADC1 1MSPS，ADC12~ADC7 200 kSPS
- 支持中断
- 软件可启动转换

●内建3.3V LDO

●电源管理

- 正常工作模式
  - ◆ CPU和时钟正常工作，耗电约46mA (CPU 96 MHz 时)
- 空闲模式
  - ◆ CPU 停止工作，外设工作
- 掉电模式
  - ◆ 高速时钟关闭，LXT 和 LIRC工作，SRAM 数据保存，耗电约 700 uA
- 深度掉电模式
  - ◆ 高速时钟关闭，LXT 和 LIRC工作，SRAM数据不保存，耗电约 7 uA

●工作温度范围: -40℃~+85℃

●封装

- 绿色环保封装 (RoHS)
- All Green package (RoHS)

- QFN 88-pin (10mm x 10mm)
- LQFP 64-pin (7mm x 7mm)
- LQFP 48-pin (7mm x 7mm)
- QFN 48-pin (7mm x 7mm)

3 缩写词

3.1 缩写词

缩写词	描述
ADC	Analog-to-Digital Converter
APB	Advanced Peripheral Bus
AHB	Advanced High-Performance Bus
DMA	Direct Memory Access
FIFO	First In, First Out
FPU	Floating Point Unit
GPIO	General-Purpose Input/Output
HCLK	The Clock of Advanced High-Performance Bus
HXT	12 MHz External High Speed Crystal Oscillator
ICP	In Circuit Programming
ISP	In System Programming
LDO	Low Dropout Regulator
LIN	Local Interconnect Network
LIRC	32.768 kHz Internal Low Speed RC Oscillator
LXT	32.768 kHz External Low Speed Crystal Oscillator
LVD	Low Voltage Detection
MPU	Memory Protection Unit
NVIC	Nested Vectored Interrupt Controller
PCLK	The Clock of Advanced Peripheral Bus
PLL	Phase-Locked Loop
PWM	Pulse Width Modulation
SD	Secure Digital
SPI	Serial Peripheral Interface
SPIM	Serial Peripheral Interface Master
SPS	Samples per Second
TMR	Timer Controller
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
WDT	Watchdog Timer
WWDT	Window Watchdog Timer

表 3.1-1 缩写词

4 芯片信息和管脚配置

4.1 选型指南

4.1.1 NuMicro™ NUC505 系列选型指南

[1]: 表中* √ 符号表示 NUC505DS13Y 仅支持耳机输出.

[2]: 此封装相同但是引脚并不兼容.

LQFP64*: 7x7mm

Part Number	Serial Flash (KB)	SRAM (KB)	ISP ROM (KB)	I/O	Timer (32-Bit)	Connectivity				I^2S	USB 2.0 HS Device	USB 2.0 FS Host	PWM (16-Bit)	24-Bit Audio CODEC[1]	DIGITAL MIC	ADC (12-Bit)	RTC	ISP/ICP	Package[2]
						I^2C	SD HOST	SPI	UART										
NUC505DLA	512	128	8	18	4	2	-	2	2	1	1	-	-	√	1	5CH	-	√	LQFP48
NUC505DL13Y	2048	128	8	25	4	2	1	3	3	1	1	1	4	-	1	5CH	√	√	LQFP48
NUC505YLA	512	128	8	18	4	2	-	2	2	1	1	-	-	√	1	5CH	-	√	QFN48
NUC505YLA2Y	512	128	8	25	4	2	1	3	3	1	1	1	4	-	1	5CH	√	√	QFN48
NUC505DSA	512	128	8	34	4	2	1	3	3	1	1	1	4	√	1	5CH	-	√	LQFP64*
NUC505DS13Y	2048	128	8	35	4	2	1	3	3	1	1	1	4	√*	1	8CH	√	√	LQFP64*
NUC505YO13Y	2048	128	8	52	4	2	1	3	3	1	1	2	4	√	1	8CH	√	√	QFN88

表 4.1-1 NuMicro™ NUC505 系列选型指南

4.1.2 NuMicro™ NUC505 系列命名规则

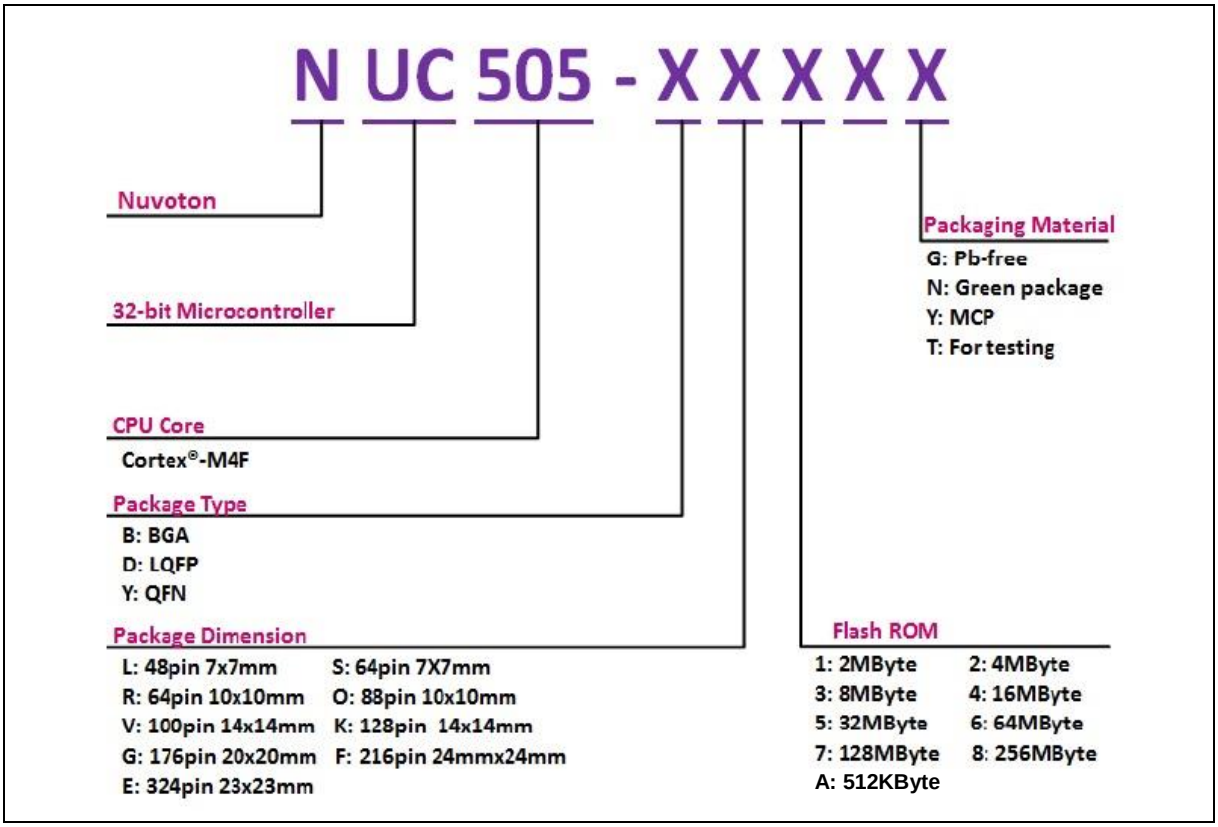


图 4.1-1 NuMicro™ NUC505 系列选型代码

4.2 管脚配置

4.2.1 NuMicro™ NUC505DLA LQFP 48管脚

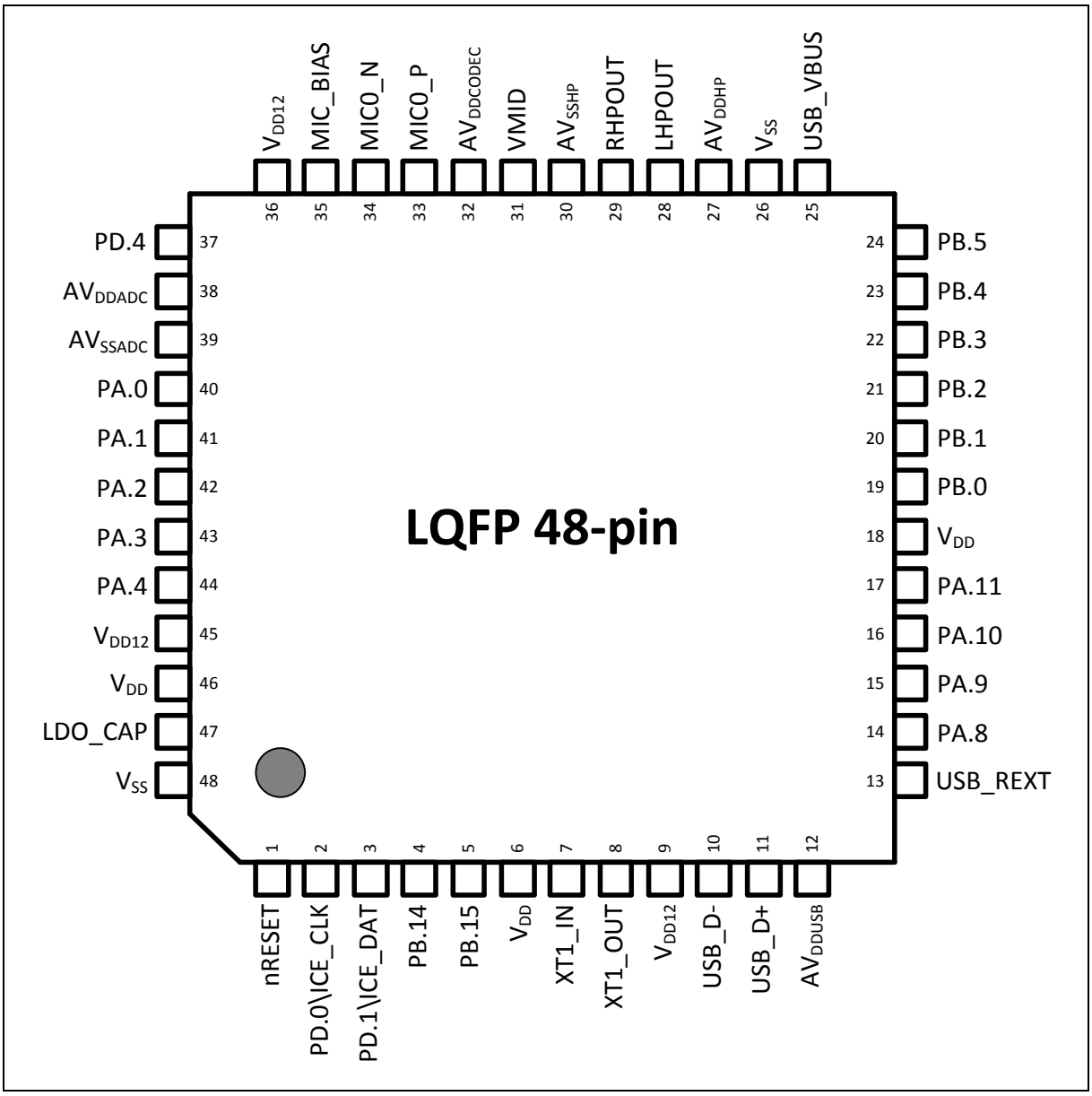


图 4.2-1 NuMicro™ NUC505DLA LQFP 48管脚图

4.2.2 NuMicro™ NUC505DL13Y LQFP 48管脚

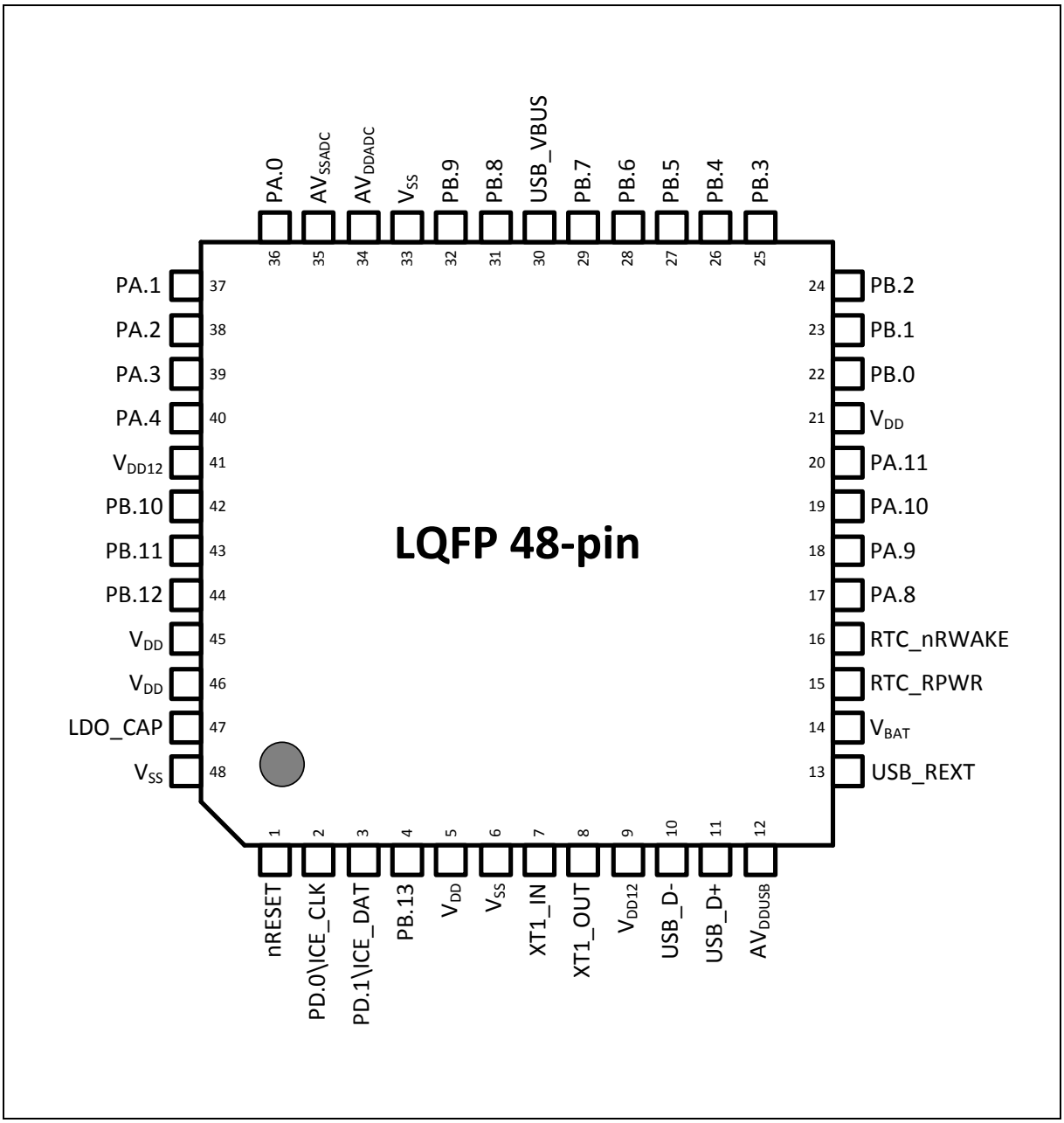


图 4.2-2 NuMicro™ NUC505DL13Y LQFP 48管脚图

4.2.3 NuMicro™ NUC505YLA QFN 48管脚

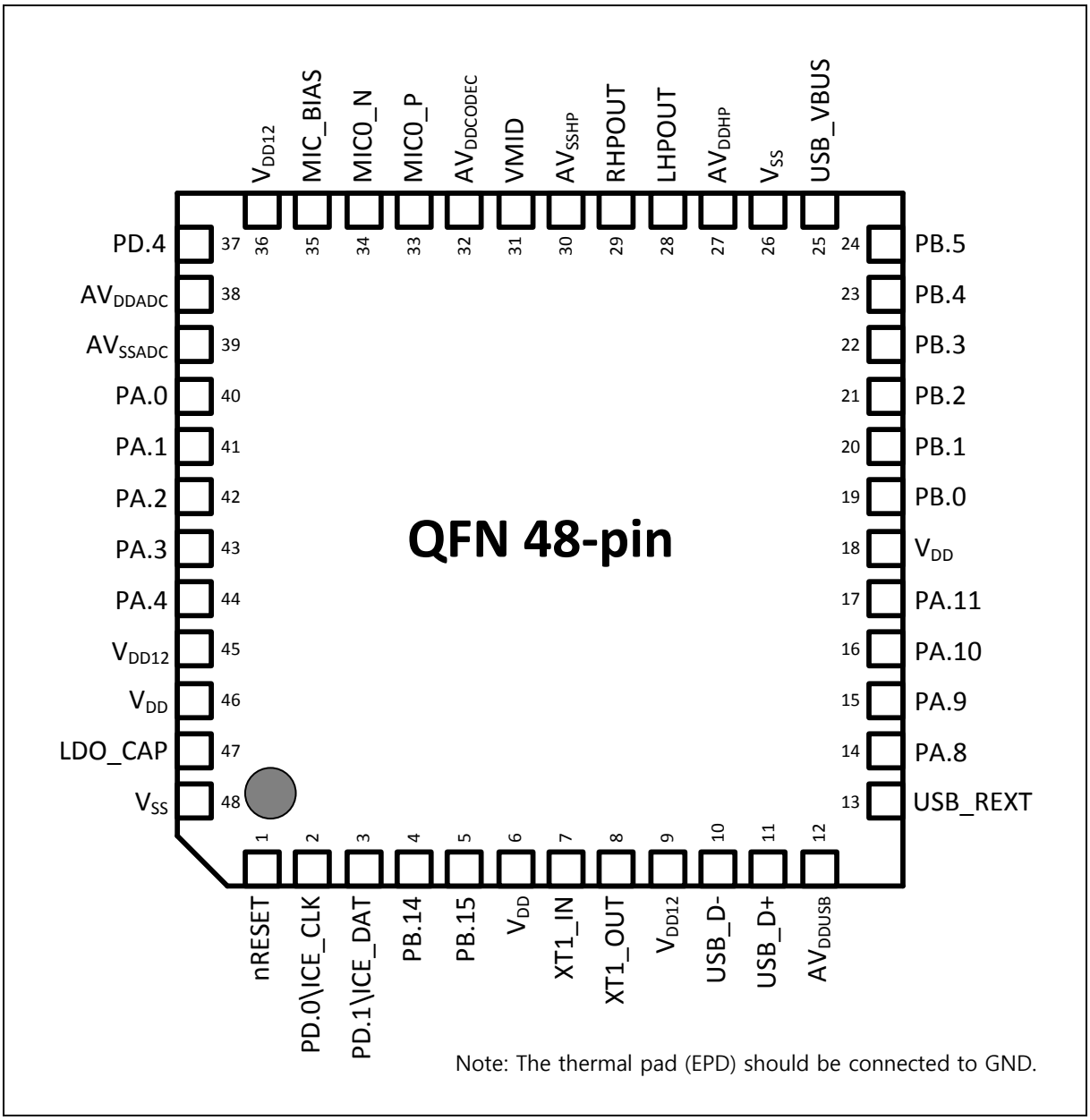
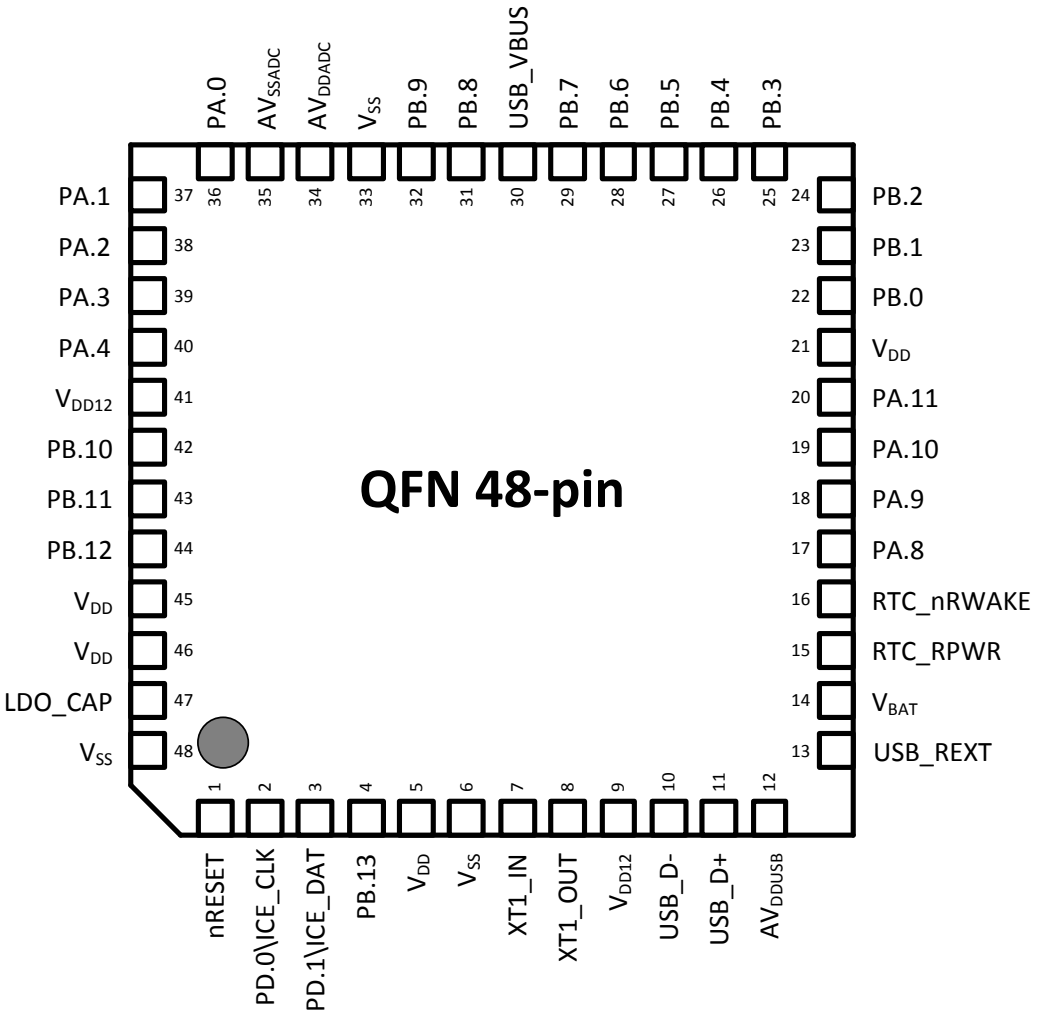


图 4.2-3 NuMicro™ NUC505YLA QFN 48管脚图

4.2.4 NuMicro™ NUC505YLA2Y QFN 48管脚



Note: The thermal pad (EPD) should be connected to GND.

图 4.2-4 NuMicro™ NUC505YLA2Y QFN 48管脚图

4.2.5 NuMicro™ NUC505DSA LQFP 64管脚

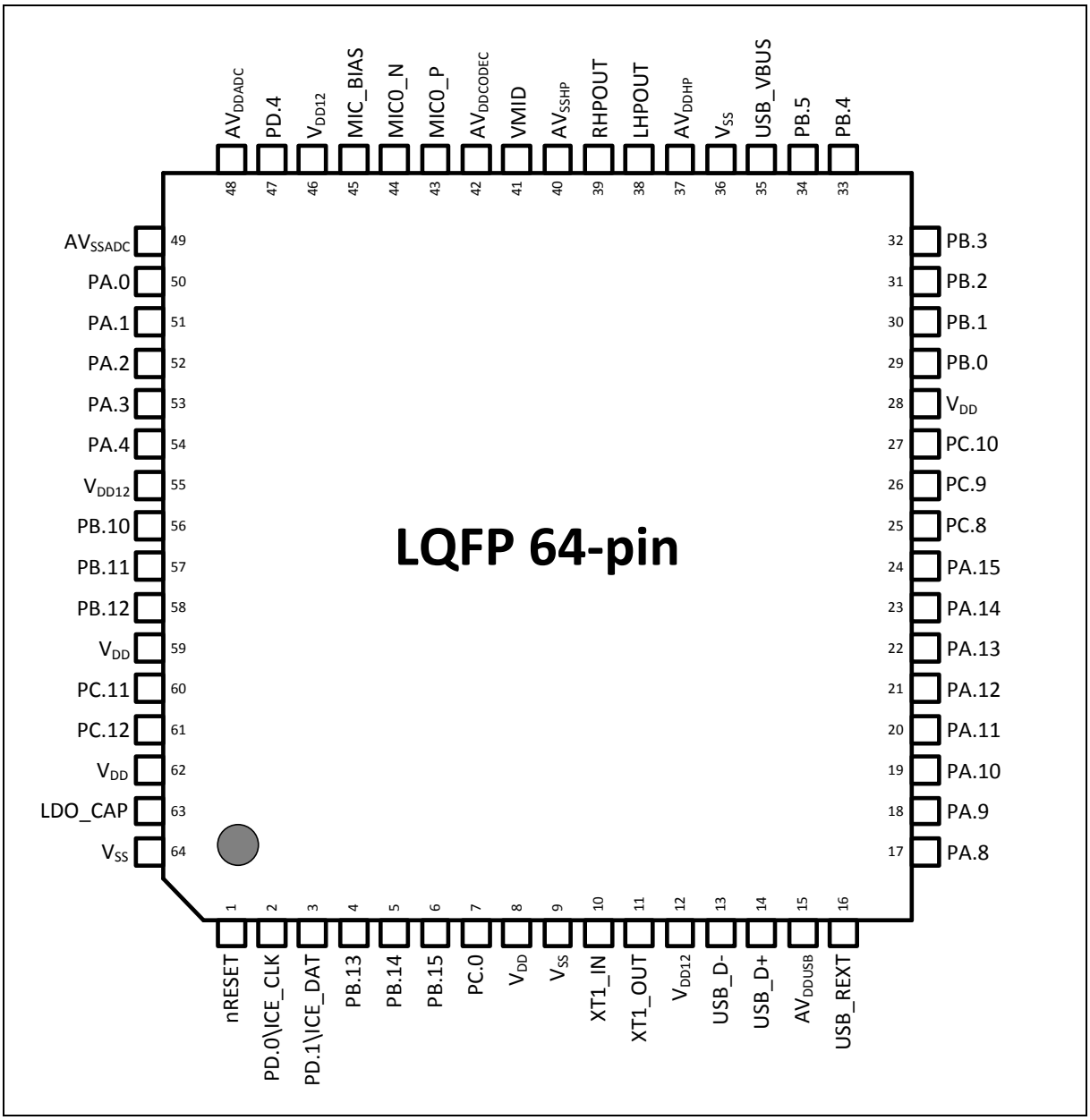


图 4.2-5 NuMicro™ NUC505DSA LQFP 64管脚图

4.2.6 NuMicro™ NUC505DS13Y LQFP 64管脚

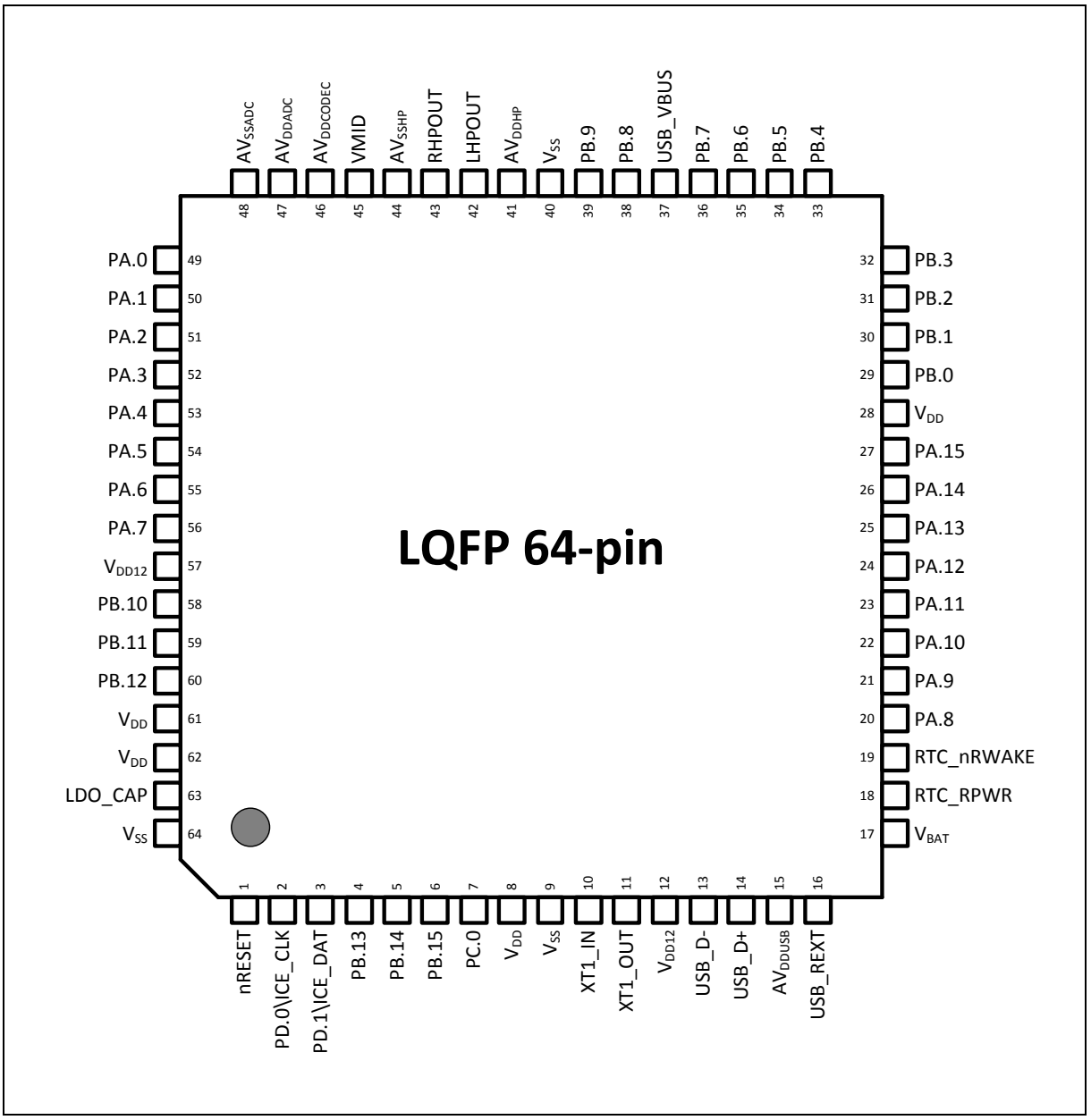


图 4.2-6 NuMicro™ NUC505DS13Y LQFP 64管脚图

4.2.7 NuMicro™ NUC505YOY13Y QFN 88管脚

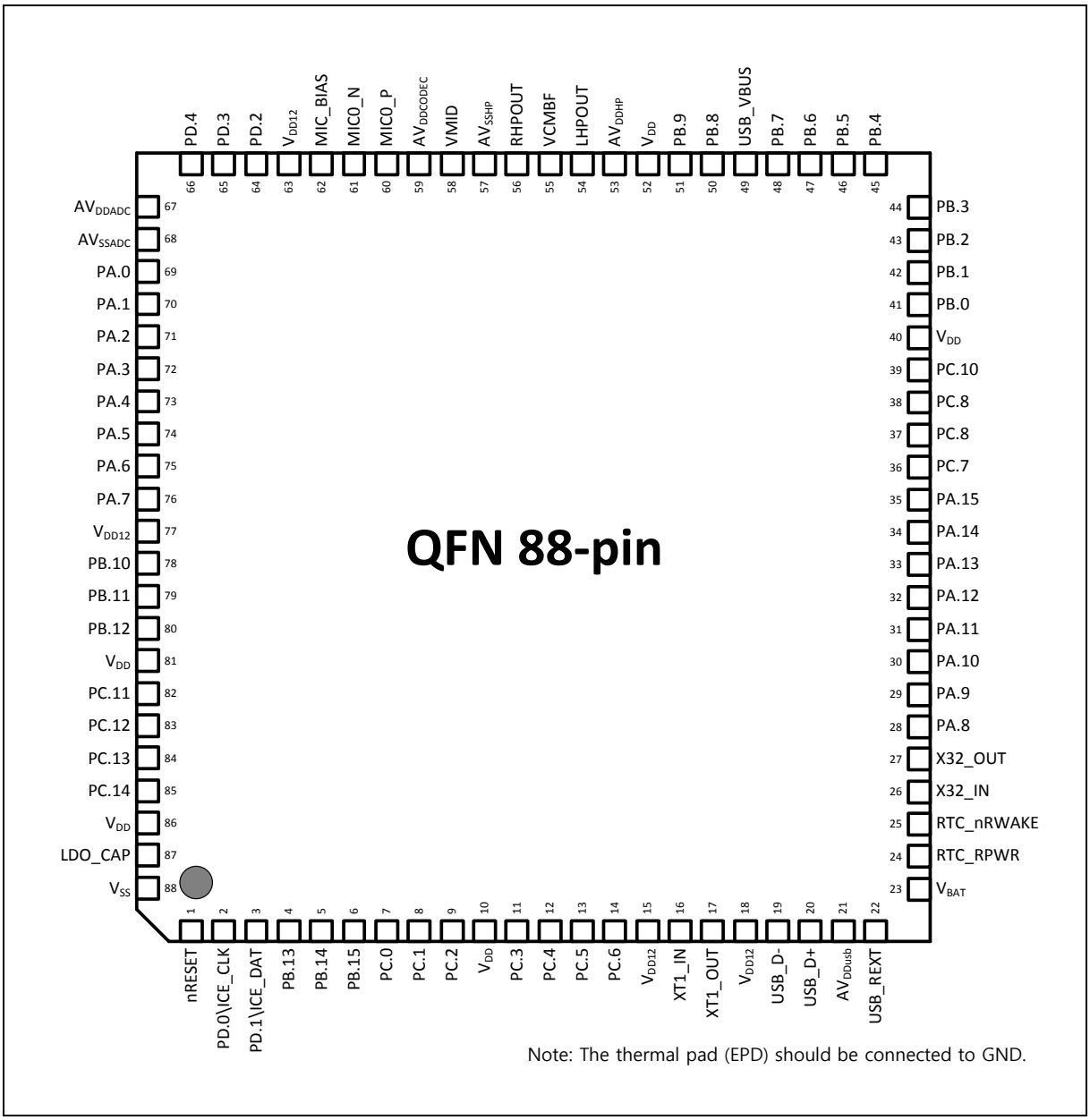


图 4.2-7 NuMicro™ NUC505YO13Y QFN 88 管脚图

4.3 管脚描述

4.3.1 NuMicro™ NUC505DLA LQFP 48 脚管脚描述

MFP* =多功能管脚(请参考 SYS_GPx_MFPL 和SYS_GPx_MFPH)

PA.0 MFP0 意思 MFP 的值配置为0，SYS_GPA_MFPL[2:0]=0x0

PA.9 MFP5 意思 MFP 的值配置为5，SYS_GPA_MFPH[6:4]=0x5

管脚号	管脚名称	管脚类型	MFP*	管脚描述
1	nRESET	I	MFP0	低复位，正常工作为高，内建上拉电阻
2	ICE_CLK	O	MFP0	仿真时钟. (仿真模式下)
	PD.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I^2C0 时钟
3	ICE_DAT	I/O	MFP0	仿真数据. (仿真模式下)
	PD.1	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I^2C0 数据
4	PB.14	I/O	MFP0	通用数字输入/输出管脚
	USBH1_D+	I/O	MFP1	USB host 1 差分 D+ 线.
	I2C1_SCL	O	MFP2	I2C1 时钟线
5	PB.15	I/O	MFP0	通用数字输入/输出管脚
	USBH1_D-	I/O	MFP1	USB host 1 差分 D- 线.
	I2C1_SDA	I/O	MFP2	I2C1 数据线
6	V_{DD}	A	MFP0	I/O 口的电源， DC 3.3V.
7	XT1_IN	I	MFP0	12 MHz 晶振引脚
8	XT1_OUT	O	MFP0	
9	V_{DD12}	A	MFP0	I/O 口的电源， DC 1.2V
10	USB_D-	A	MFP0	USB 接口
11	USB_D+	A	MFP0	
12	AV_{DDUSB}	A	MFP0	USB 电源, DC 3.3V.

13	USB_REXT	A	MFP0	12.1 KΩ 用于 USB.
14	PA.8	I/O	MFP0	通用数字输入/输出管脚
	SPIM_SS	O	MFP1	SPIM 从机片选.
	I2S_LRCLK	I/O	MFP2	I^2S 左右声道时钟.
	UART1_TXD	O	MFP3	UART1 发送.
15	PA.9	I/O	MFP0	通用数字输入/输出管脚
	SPIM_CLK	O	MFP1	SPIM 时钟.
	I2S_BCLK	I/O	MFP2	I^2S 位时钟
	UART1_RXD	I	MFP3	UART1 接收.
	SYSCFG[0]	I	MFP0	系统配置0
16	PA.10	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MOSI	I/O	MFP1	SPIM 主机发送端,从机接收端.
	I2C1_SCL	O	MFP2	I2C1 时钟.
	SD_CLK	O	MFP4	SD/SDH 时钟.
	SYSCFG[1]	I	MFP0	系统配置1.
17	PA.11	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MISO	I/O	MFP1	SPIM 主机接收,从机发送端
	I2C1_SDA	I/O	MFP2	I2C1 数据.
	SD_CMD	I	MFP4	SD/SDH 命令.
18	V_{DD}	A	MFP0	I/O 电源, DC 3.3V.
19	PB.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I2C0 时钟.
	UART0_TXD	O	MFP3	UART0.发送
	SD_DAT2	I/O	MFP4	SD/SDH 数据2.
20	PB.1	I/O	MFP0	通用数字输入/输出管脚

	I2C0_SDA	I/O	MFP2	I2C0 数据
	UART0_RXD	I	MFP3	UART0 接收.
	SD_DAT3	I/O	MFP4	SD/SDH 数据 3.
21	PB.2	I/O	MFP0	通用数字输入/输出管脚
	SPI0_SS	O	MFP1	SPI0 从机片选.
	SD_CMD	I	MFP4	SD/SDH 命令.
22	PB.3	I/O	MFP0	通用数字输入/输出管脚
	SPI0_CLK	O	MFP1	SPI0 时钟.
	SD_CLK	O	MFP4	SD/SDH 时钟.
	SYSCFG[2]	I	MFP0	系统配置 2.
23	PB.4	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MOSI	O	MFP1	SPI0 MOSI 主机发送端,从机接收端
	SYSCFG[3]	I	MFP0	系统配置3.
24	PB.5	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MISO	I	MFP1	SPI0 MISO 主机接收端,从机发送端.
	SD_nCD	I	MFP4	SD/SDH 卡检测端
25	USB_VBUS33	I	MFP0	检测USB是否插入
26	V_{SS}	A	MFP0	地
27	AV_{DDHP}	AP	VDD	解码器 耳机电源, DC 3.3V.
28	LHPOUT	A	MFP0	耳机左声道输出
29	RHPOUT	A	MFP0	耳机驱动的 VCM 输出.
30	AV_{SSHP}	A	MFP0	耳机右声道输出.
31	VMID	A	MFP0	耳机参考 电源..
32	$AV_{DDCODEC}$	A	MFP0	解码器电源DC 3.3V.
33	MIC0_P	A	MFP0	麦克风0正端

34	MIC0_N	A	MFP0	麦克风0负端
35	MIC_BIAS	A	MFP0	MIC偏值
36	V_{DD12}	A	MFP0	I/O电源, DC 1.2V
37	PD.4	I/O	MFP0	通用数字输入/输出管脚
	RLINEIN	A	MFP1	解码器右通道输入
38	AV_{DDADC}	A	MFP0	ADC电源, DC 3.3V.
39	AV_{SSADC}	A	MFP0	ADC 地
40	PA.0	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH0	A	MFP1	ADC0输入
41	PA.1	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH1	A	MFP1	ADC1.
42	PA.2	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH2	A	MFP1	ADC2
	I2S_MCLK	O	MFP2	I^2S 主时钟.
43	PA.3	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH3	A	MFP1	ADC3.
	I2S_DI	I	MFP2	I^2S 输入
44	PA.4	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH4	A	MFP1	ADC4
	I2S_DO	O	MFP2	I^2S 输出
45	V_{DD12}	A	MFP0	I/O 电源, DC 1.2V
46	V_{DD}	A	MFP0	I/O电源, DC 3.3V.
47	LDO_CAP	A	MFP0	LDO 输出端，外接退藕电容
48	V_{SS}	A	MFP0	地

4.3.2 NuMicro™ NUC505DL13Y LQFP 48 脚管脚描述

MFP* =多功能管脚(请参考 SYS_GPx_MFPL 和SYS_GPx_MFPH)

PA.0 MFP0 意思 MFP 的值配置为0，SYS_GPA_MFPL[2:0]=0x0

PA.9 MFP5 意思 MFP 的值配置为5，SYS_GPA_MFPH[6:4]=0x5

管脚号	管脚名称	管脚类型	MFP*	管脚描述
1	nRESET	I	MFP0	低复位，正常工作为高，内建上拉电阻
2	ICE_CLK	O	MFP0	仿真时钟. (仿真模式下)
	PD.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I^2C0 时钟
3	ICE_DAT	I/O	MFP0	仿真数据. (仿真模式下)
	PD.1	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I^2C0 数据
4	PB.13	I/O	MFP0	通用数字输入/输出管脚
	SPI1_MISO	I	MFP1	SPI1 主机模式输入，从机模式输出.
	USBH1_D-	I/O	MFP2	USB host 1 差分 D- 线.
	UART2_nRTS	O	MFP3	UART2 流控， 请求发送.
	PWM_CH3	I/O	MFP4	PWM 通道3
5	V_{DD}	A	MFP0	I/O 口的电源， DC 3.3V.
6	V_{SS}	A	MFP0	地.
7	XT1_IN	I	MFP0	12 MHz 晶振引脚
8	XT1_OUT	O	MFP0	
9	V_{DD12}	A	MFP0	I/O 口的电源， DC 1.2V
10	USB_D-	A	MFP0	USB 接口
11	USB_D+	A	MFP0	
12	AV_{DDUSB}	A	MFP0	USB 电源, DC 3.3V.
13	USB_REXT	A	MFP0	12.1 KΩ 用于 USB.
14	V_{BAT}	A	MFP0	RTC 电池引脚, DC 3.3V.
15	RTC_RPWR	O	MFP0	高电平使能引脚电源控制

16	RTC_nRWAKE	I	MFP0	低电平触发系统电源使能.
17	PA.8	I/O	MFP0	通用数字输入/输出管脚
	SPIM_SS	O	MFP1	SPIM 从机片选.
	I2S_LRCLK	I/O	MFP2	I^2S 左右声道时钟.
	UART1_TXD	O	MFP3	UART1 发送.
18	PA.9	I/O	MFP0	通用数字输入/输出管脚
	SPIM_CLK	O	MFP1	SPIM 时钟.
	I2S_BCLK	I/O	MFP2	I^2S 位时钟
	UART1_RXD	I	MFP3	UART1 接收.
	SYSCFG[0]	I	MFP0	系统配置0
19	PA.10	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MOSI	I/O	MFP1	SPIM 主机发送端,从机接收端.
	I2C1_SCL	O	MFP2	I2C1 时钟.
	SD_CLK	O	MFP4	SD/SDH 时钟.
	SYSCFG[1]	I	MFP0	系统配置1.
20	PA.11	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MISO	I/O	MFP1	SPIM 主机接收,从机发送端
	I2C1_SDA	I/O	MFP2	I2C1 数据.
	SD_CMD	I	MFP4	SD/SDH 命令.
21	V_{DD}	A	MFP0	I/O 电源, DC 3.3V.
22	PB.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I2C0 时钟.
	UART0_TXD	O	MFP3	UART0.发送
	SD_DAT2	I/O	MFP4	SD/SDH 数据2.
23	PB.1	I/O	MFP0	通用数字输入/输出管脚

	I2C0_SDA	I/O	MFP2	I2C0 数据
	UART0_RXD	I	MFP3	UART0 接收.
	SD_DAT3	I/O	MFP4	SD/SDH 数据 3.
24	PB.2	I/O	MFP0	通用数字输入/输出管脚
	SPI0_SS	O	MFP1	SPI0 从机片选.
	SD_CMD	I	MFP4	SD/SDH 命令.
25	PB.3	I/O	MFP0	通用数字输入/输出管脚
	SPI0_CLK	O	MFP1	SPI0 时钟.
	SD_CLK	O	MFP4	SD/SDH 时钟.
	SYSCFG[2]	I	MFP0	系统配置 2.
26	PB.4	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MOSI	O	MFP1	SPI0 MOSI 主机发送端,从机接收端
	SYSCFG[3]	I	MFP0	系统配置3.
27	PB.5	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MISO	I	MFP1	SPI0 MISO 主机接收端,从机发送端.
	SD_nCD	I	MFP4	SD/SDH 卡检测端
28	PB.6	I/O	MFP0	通用数字输入/输出管脚
	UART1_TXD	O	MFP3	UART1.发送
	SD_DAT0	I/O	MFP4	SD/SDH 数据 0.
29	PB.7	I/O	MFP0	通用数字输入/输出管脚
	UART1_RXD	I	MFP3	UART1.接收
	SD_DAT1	I/O	MFP4	SD/SDH 数据 1.
30	USB_VBUS33	I	MFP0	检测USB是否插入
31	PB.8	I/O	MFP0	通用数字输入/输出管脚
	USB_PWEN	O	MFP1	USB host 电流控制端.

	TM1_CNT_OUT	I/O	MFP2	Timer1计数输入,或反转输出
	UART1_nCTS	I	MFP3	UART1流控，清除发送.
	SD_DAT2	I/O	MFP4	SD/SDH 数据2.
32	PB.9	I/O	MFP0	通用数字输入/输出管脚
	USBH_OVD	I	MFP1	USB host 总线过压检测.
	TM1_EXT	I	MFP2	Timer1 捕获输入端.
	UART1_nRTS	O	MFP3	UART1 流控，请求发送.
	SD_DAT3	I/O	MFP4	SD/SDH 数据 3.
33	V_{SS}	A	MFP0	地
34	AV_{DDADC}	A	MFP0	模拟电源 DC 3.3V.
35	AV_{SSADC}	A	MFP0	模拟地
36	PA.0	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH0	A	MFP1	ADC0输入
37	PA.1	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH1	A	MFP1	ADC1.
38	PA.2	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH2	A	MFP1	ADC2
	I2S_MCLK	O	MFP2	I^2S 主时钟.
39	PA.3	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH3	A	MFP1	ADC3.
	I2S_DI	I	MFP2	I^2S 输入
40	PA.4	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH4	A	MFP1	ADC4
	I2S_DO	O	MFP2	I^2S 输出
41	V_{DD12}	A	MFP0	I/O 电源, DC 1.2V
42	PB.10	I/O	MFP0	通用数字输入/输出管脚

	SPI1_SS	O	MFP1	SPI1 片选.
	I2C1_SCL	O	MFP2	I2C1 时钟
	UART2_TXD	O	MFP3	UART2.发送
	PWM_CH0	I/O	MFP4	PWM0
43	PB.11	I/O	MFP0	通用数字输入/输出管脚
	SPI1_CLK	O	MFP1	SPI1 时钟
	I2C1_SDA	I/O	MFP2	I2C1 数据
	UART2_RXD	I	MFP3	UART2.接收
	PWM_CH1	I/O	MFP4	PWM1
44	PB.12	I/O	MFP0	通用数字输入/输出管脚
	SPI1_MOSI	O	MFP1	SPI1 主机输出从机输入
	USBH1_D+	I/O	MFP2	USB host-lite1 差分数据线 D+
	UART2_nCTS	I	MFP3	UART2.流控，请除发送
	PWM_CH2	I/O	MFP4	PWM2
45	V_{DD}	A	MFP0	I/O电源, DC 3.3V.
46	V_{DD}	A	MFP0	
47	LDO_CAP	A	MFP0	LDO 输出端，外接退藕电容
48	V_{SS}	A	MFP0	地

4.3.3 NuMicro™ NUC505YLA QFN 48脚管脚描述

MFP* =多功能管脚(请参考 SYS_GPx_MFPL 和 SYS_GPx_MFPH)

PA.0 MFP0 意思 MFP 的值配置为0，SYS_GPA_MFPL[2:0]=0x0

PA.9 MFP5 意思 MFP 的值配置为5，SYS_GPA_MFPH[6:4]=0x5

管脚号	管脚名称	管脚类型	MFP*	管脚描述
1	nRESET	I	MFP0	低复位，正常工作为高，内建上拉电阻
2	ICE_CLK	O	MFP0	仿真时钟. (仿真模式下)
	PD.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I^2C0 时钟
3	ICE_DAT	I/O	MFP0	仿真数据. (仿真模式下)
	PD.1	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I^2C0 数据
4	PB.14	I/O	MFP0	通用数字输入/输出管脚
	USBH1_D+	I/O	MFP1	USB host 1 差分 D+ 线.
	I2C1_SCL	O	MFP2	I2C1 时钟线
5	PB.15	I/O	MFP0	通用数字输入/输出管脚
	USBH1_D-	I/O	MFP1	USB host 1 差分 D- 线.
	I2C1_SDA	I/O	MFP2	I2C1 数据线
6	V_{DD}	A	MFP0	I/O 口的电源， DC 3.3V.
7	XT1_IN	I	MFP0	外接晶振
8	XT1_OUT	O	MFP0	
9	V_{DD12}	A	MFP0	I/O 口的电源， DC 1.2V
10	USB_D-	A	MFP0	USB 接口
11	USB_D+	A	MFP0	
12	AV_{DDUSB}	A	MFP0	USB 电源, DC 3.3V.
13	USB_REXT	A	MFP0	12.1 KΩ 用于 USB.

管脚号	管脚名称	管脚类型	MFP*	管脚描述
14	PA.8	I/O	MFP0	通用数字输入/输出管脚
	SPIM_SS	O	MFP1	SPIM 从机片选.
	I2S_LRCLK	I/O	MFP2	I^2S 左右声道时钟.
	UART1_TXD	O	MFP3	UART1 发送.
15	PA.9	I/O	MFP0	通用数字输入/输出管脚
	SPIM_CLK	O	MFP1	SPIM 时钟.
	I2S_BCLK	I/O	MFP2	I^2S 位时钟
	UART1_RXD	I	MFP3	UART1 接收.
	SYSCFG[0]	I	MFP0	系统配置0
16	PA.10	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MOSI	I/O	MFP1	SPIM 主机发送端,从机接收端.
	I2C1_SCL	O	MFP2	I2C1 时钟.
	SD_CLK	O	MFP4	SD/SDH 时钟
	SYSCFG[1]	I	MFP0	系统配置1.
17	PA.11	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MISO	I/O	MFP1	SPIM 主机接收,从机发送端
	I2C1_SDA	I/O	MFP2	I2C1 数据.
	SD_CMD	I	MFP4	SD/SDH 命令.
18	V_{DD}	A	MFP0	I/O 电源, DC 3.3V.
19	PB.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I2C0时钟
	UART0_TXD	O	MFP3	UART0 发送.
	SD_DAT2	I/O	MFP4	SD/SDH 数据2
20	PB.1	I/O	MFP0	通用数字输入/输出管脚

管脚号	管脚名称	管脚类型	MFP*	管脚描述
	I2C0_SDA	I/O	MFP2	I2C0 数据
	UART0_RXD	I	MFP3	UART0 接收
	SD_DAT3	I/O	MFP4	SD/SDH 数据3
21	PB.2	I/O	MFP0	通用数字输入/输出管脚
	SPI0_SS	O	MFP1	SPI0 片选
	SD_CMD	I	MFP4	SD/SDH 命令
22	PB.3	I/O	MFP0	通用数字输入/输出管脚
	SPI0_CLK	O	MFP1	SPI0 时钟
	SD_CLK	O	MFP4	SD/SDH 时钟
	SYSCFG[2]	I	MFP0	系统配置 2.
23	PB.4	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MOSI	O	MFP1	SPI0 主机发送端,从机接收端.
	SYSCFG[3]	I	MFP0	系统配置 3.
24	PB.5	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MISO	I	MFP1	SPI0 MISO 主机接收端,从机发送端.
	SD_nCD	I	MFP4	SD/SDH 卡检测端
25	USB_VBUS33	I	MFP0	检测USB是否插入
26	V_{SS}	A	MFP0	地
27	AV_{DDHP}	AP	VDD	解码器 耳机电源, DC 3.3V.
28	LHPOUT	A	MFP0	耳机左声道输出
29	RHPOUT	A	MFP0	耳机右声道输出.
30	AV_{SSHP}	A	MFP0	耳机解码器地
31	VMID	A	MFP0	耳机参考 电源..
32	$AV_{DDCODEC}$	A	MFP0	解码器电源DC 3.3V.

管脚号	管脚名称	管脚类型	MFP*	管脚描述
33	MIC0_P	A	MFP0	麦克风0正端
34	MIC0_N	A	MFP0	麦克风0负端
35	MIC_BIAS	A	MFP0	MIC偏值
36	V_{DD12}	A	MFP0	I/O电源, DC 1.2V
37	PD.4	I/O	MFP0	通用数字输入/输出管脚
	RLINEIN	A	MFP1	解码器右通道输入
38	AV_{DDADC}	A	MFP0	ADC电源, DC 3.3V.
39	AV_{SSADC}	A	MFP0	ADC 地
40	PA.0	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH0	A	MFP1	ADC0
41	PA.1	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH1	A	MFP1	ADC1
42	PA.2	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH2	A	MFP1	ADC2
	I2S_MCLK	O	MFP2	I^2S 主时钟.
43	PA.3	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH3	A	MFP1	ADC 3
	I2S_DI	I	MFP2	I^2S 输入.
44	PA.4	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH4	A	MFP1	ADC4
	I2S_DO	O	MFP2	I^2S输出
45	V_{DD12}	A	MFP0	I/O 电源, DC 1.2V
46	V_{DD}	A	MFP0	I/O电源 DC 3.3V.
47	LDO_CAP	A	MFP0	LDO 输出， 外接退藕电容
48	V_{SS}	A	MFP0	地

注意：QFN封装底部的散热垫（EPD）应连接到GND。

4.3.4 NuMicro™ NUC505YLA2Y QFN 48脚管脚描述

MFP* =多功能管脚(请参考 SYS_GPx_MFPL 和 SYS_GPx_MFPH)

PA.0 MFP0 意思 MFP 的值配置为0，SYS_GPA_MFPL[2:0]=0x0

PA.9 MFP5 意思 MFP 的值配置为5，SYS_GPA_MFPH[6:4]=0x5

管脚号	管脚名称	管脚类型	MFP*	管脚描述
1	nRESET	I	MFP0	低复位，正常工作为高，内建上拉电阻
2	ICE_CLK	O	MFP0	仿真时钟. (仿真模式下)
	PD.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I^2C0 时钟
3	ICE_DAT	I/O	MFP0	仿真数据. (仿真模式下)
	PD.1	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I^2C0 数据
4	PB.13	I/O	MFP0	通用数字输入/输出管脚
	SPI1_MISO	I	MFP1	SPI1 主机模式输入，从机模式输出.
	USBH1_D-	I/O	MFP2	USB host 1 差分 D- 线.
	UART2_nRTS	O	MFP3	UART2 流控， 请求发送.
	PWM_CH3	I/O	MFP4	PWM 通道3
5	V_{DD}	A	MFP0	I/O 口的电源， DC 3.3V.
6	V_{SS}	A	MFP0	地.
7	XT1_IN	I	MFP0	12 MHz 晶振引脚
8	XT1_OUT	O	MFP0	
9	V_{DD12}	A	MFP0	I/O 口的电源， DC 1.2V
10	USB_D-	A	MFP0	USB 接口
11	USB_D+	A	MFP0	
12	AV_{DDUSB}	A	MFP0	USB 电源, DC 3.3V.
13	USB_REXT	A	MFP0	12.1 KΩ 用于 USB.

14	V_{BAT}	A	MFP0	RTC 电池引脚, DC 3.3V.
15	RTC_RPWR	O	MFP0	高电平使能引脚电源控制
16	RTC_nRWAKE	I	MFP0	低电平触发系统电源使能.
17	PA.8	I/O	MFP0	通用数字输入/输出管脚
	SPIM_SS	O	MFP1	SPIM 从机片选.
	I2S_LRCLK	I/O	MFP2	I^2S 左右声道时钟.
	UART1_TXD	O	MFP3	UART1 发送.
18	PA.9	I/O	MFP0	通用数字输入/输出管脚
	SPIM_CLK	O	MFP1	SPIM 时钟.
	I2S_BCLK	I/O	MFP2	I^2S 位时钟
	UART1_RXD	I	MFP3	UART1 接收.
	SYSCFG[0]	I	MFP0	系统配置0
19	PA.10	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MOSI	I/O	MFP1	SPIM 主机发送端,从机接收端.
	I2C1_SCL	O	MFP2	I2C1 时钟.
	SD_CLK	O	MFP4	SD/SDH 时钟.
	SYSCFG[1]	I	MFP0	系统配置1.
20	PA.11	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MISO	I/O	MFP1	SPIM 主机接收,从机发送端
	I2C1_SDA	I/O	MFP2	I2C1 数据.
	SD_CMD	I	MFP4	SD/SDH 命令.
21	V_{DD}	A	MFP0	I/O 电源, DC 3.3V.
22	PB.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I2C0 时钟.
	UART0_TXD	O	MFP3	UART0.发送

	SD_DAT2	I/O	MFP4	SD/SDH 数据2.
23	PB.1	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I2C0 数据
	UART0_RXD	I	MFP3	UART0 接收.
	SD_DAT3	I/O	MFP4	SD/SDH 数据 3.
24	PB.2	I/O	MFP0	通用数字输入/输出管脚
	SPI0_SS	O	MFP1	SPI0 从机片选.
	SD_CMD	I	MFP4	SD/SDH 命令.
25	PB.3	I/O	MFP0	通用数字输入/输出管脚
	SPI0_CLK	O	MFP1	SPI0 时钟.
	SD_CLK	O	MFP4	SD/SDH 时钟.
	SYSCFG[2]	I	MFP0	系统配置 2.
26	PB.4	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MOSI	O	MFP1	SPI0 MOSI 主机发送端,从机接收端
	SYSCFG[3]	I	MFP0	系统配置3.
27	PB.5	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MISO	I	MFP1	SPI0 MISO 主机接收端,从机发送端.
	SD_nCD	I	MFP4	SD/SDH 卡检测端
28	PB.6	I/O	MFP0	通用数字输入/输出管脚
	UART1_TXD	O	MFP3	UART1.发送
	SD_DAT0	I/O	MFP4	SD/SDH 数据 0.
29	PB.7	I/O	MFP0	通用数字输入/输出管脚
	UART1_RXD	I	MFP3	UART1.接收
	SD_DAT1	I/O	MFP4	SD/SDH 数据 1.
30	USB_VBUS33	I	MFP0	检测USB是否插入

31	PB.8	I/O	MFP0	通用数字输入/输出管脚
	USB_PWEN	O	MFP1	USB host 电流控制端.
	TM1_CNT_OUT	I/O	MFP2	Timer1计数输入,或反转输出
	UART1_nCTS	I	MFP3	UART1流控，清除发送.
	SD_DAT2	I/O	MFP4	SD/SDH 数据2.
32	PB.9	I/O	MFP0	通用数字输入/输出管脚
	USBH_OVD	I	MFP1	USB host 总线过压检测.
	TM1_EXT	I	MFP2	Timer1 捕获输入端.
	UART1_nRTS	O	MFP3	UART1 流控，请求发送.
	SD_DAT3	I/O	MFP4	SD/SDH 数据 3.
33	V_{SS}	A	MFP0	地
34	AV_{DDADC}	A	MFP0	模拟电源 DC 3.3V.
35	AV_{SSADC}	A	MFP0	模拟地
36	PA.0	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH0	A	MFP1	ADC0输入
37	PA.1	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH1	A	MFP1	ADC1.
38	PA.2	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH2	A	MFP1	ADC2
	I2S_MCLK	O	MFP2	I^2S 主时钟.
39	PA.3	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH3	A	MFP1	ADC3.
	I2S_DI	I	MFP2	I^2S 输入
40	PA.4	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH4	A	MFP1	ADC4
	I2S_DO	O	MFP2	I^2S 输出

41	V_{DD12}	A	MFP0	I/O 电源, DC 1.2V
42	PB.10	I/O	MFP0	通用数字输入/输出管脚
	SPI1_SS	O	MFP1	SPI1 片选.
	I2C1_SCL	O	MFP2	I2C1 时钟
	UART2_TXD	O	MFP3	UART2.发送
	PWM_CH0	I/O	MFP4	PWM0
43	PB.11	I/O	MFP0	通用数字输入/输出管脚
	SPI1_CLK	O	MFP1	SPI1 时钟
	I2C1_SDA	I/O	MFP2	I2C1 数据
	UART2_RXD	I	MFP3	UART2.接收
	PWM_CH1	I/O	MFP4	PWM1
44	PB.12	I/O	MFP0	通用数字输入/输出管脚
	SPI1_MOSI	O	MFP1	SPI1 主机输出从机输入
	USBH1_D+	I/O	MFP2	USB host-lite1 差分数据线 D+
	UART2_nCTS	I	MFP3	UART2.流控，请除发送
	PWM_CH2	I/O	MFP4	PWM2
45	V_{DD}	A	MFP0	I/O电源, DC 3.3V.
46	V_{DD}	A	MFP0	
47	LDO_CAP	A	MFP0	LDO 输出端，外接退藕电容
48	V_{SS}	A	MFP0	地

注意：QFN封装底部的散热垫（EPD）应连接到GND。

4.3.5 NuMicro™ NUC505DSA LQFP 64脚管脚描述

MFP* =多功能管脚(请参考 SYS_GPx_MFPL 和 SYS_GPx_MFPH)

PA.0 MFP0 意思 MFP 的值配置为0，SYS_GPA_MFPL[2:0]=0x0

PA.9 MFP5 意思 MFP 的值配置为5，SYS_GPA_MFPH[6:4]=0x5

管脚号	管脚名称	管脚类型	MFP*	管脚描述
1	nRESET	I	MFP0	低复位，正常工作为高，内建上拉电阻
2	ICE_CLK	O	MFP0	仿真时钟. (仿真模式下)
	PD.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I^2C0 时钟
3	ICE_DAT	I/O	MFP0	仿真数据. (仿真模式下)
	PD.1	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I^2C0 数据
4	PB.13	I/O	MFP0	通用数字输入/输出管脚
	SPI1_MISO	I	MFP1	SPI1 主机模式输入，从机模式输出.
	USBH1_D-	I/O	MFP2	USB host 1 差分 D- 线.
	UART2_nRTS	O	MFP3	UART2 流控， 请求发送.
	PWM_CH3	I/O	MFP4	PWM 通道3
5	PB.14	I/O	MFP0	通用数字输入/输出管脚
	USBH1_D+	I/O	MFP1	USB host 1 差分 D+ 线.
	I2C1_SCL	O	MFP2	I2C1 时钟线
6	PB.15	I/O	MFP0	通用数字输入/输出管脚
	USBH1_D-	I/O	MFP1	USB host 1 差分 D- 线.
	I2C1_SDA	I/O	MFP2	I2C1 数据线
7	PC.0	I/O	MFP0	通用数字输入/输出管脚
	SD_CMD	I	MFP1	SD/SDH 命令线.
8	V_{DD}	A	MFP0	I/O 口的电源， DC 3.3V.

管脚号	管脚名称	管脚类型	MFP*	管脚描述
9	V_{SS}	A	MFP0	地
10	XT1_IN	I	MFP0	外接晶振
11	XT1_OUT	O	MFP0	
12	V_{DD12}	A	MFP0	I/O 口的电源， DC 1.2V
13	USB_D-	A	MFP0	USB 接口
14	USB_D+	A	MFP0	
15	AV_{DDUSB}	A	MFP0	USB 电源, DC 3.3V.
16	USB_REXT	A	MFP0	12.1 KΩ 用于 USB.
17	PA.8	I/O	MFP0	通用数字输入/输出管脚
	SPIM_SS	O	MFP1	SPIM 从机片选.
	I2S_LRCLK	I/O	MFP2	I^2S 左右声道时钟.
	UART1_TXD	O	MFP3	UART1 发送.
18	PA.9	I/O	MFP0	通用数字输入/输出管脚
	SPIM_CLK	O	MFP1	SPIM 时钟.
	I2S_BCLK	I/O	MFP2	I^2S 位时钟
	UART1_RXD	I	MFP3	UART1 接收.
	SYSCFG[0]	I	MFP0	系统配置0
19	PA.10	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MOSI	I/O	MFP1	SPIM 主机发送端,从机接收端.
	I2C1_SCL	O	MFP2	I2C1 时钟.
	SD_CLK	O	MFP4	SD/SDH 时钟
	SYSCFG[1]	I	MFP0	系统配置1.
20	PA.11	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MISO	I/O	MFP1	SPIM 主机接收,从机发送端
	I2C1_SDA	I/O	MFP2	I2C1 数据.

管脚号	管脚名称	管脚类型	MFP*	管脚描述
	SD_CMD	I	MFP4	SD/SDH 命令.
21	PA.12	I/O	MFP0	通用数字输入/输出管脚
	SPIM_D2	I/O	MFP1	SPIM 数据2
	TM0_CNT_OUT	I	MFP2	Timer0 计数输入,或反转输出
22	PA.13	I/O	MFP0	通用数字输入/输出管脚
	SPIM_D3	I/O	MFP1	SPIM 数据3
	TM0_EXT	I	MFP2	Timer0 捕获输入
	SD_nCD	I	MFP4	SD/SDH 卡检测
23	PA.14	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I2C0 时钟线
	SD_DAT0	I/O	MFP4	SD/SDH 数据0
24	PA.15	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I2C0 数据线
	SD_DAT1	I/O	MFP4	SD/SDH 数据1
25	PC.8	I/O	MFP0	通用数字输入/输出管脚
	I2S_MCLK	O	MFP1	I^2S 主时钟.
26	PC.9	I/O	MFP0	通用数字输入/输出管脚
	I2S_DI	I	MFP1	I^2S 输入
	TM2_CNT_OUT	I/O	MFP2	Timer2计数输入,或反转输出
	PWM_CH0	I/O	MFP3	PWM0
27	PC.10	I/O	MFP0	通用数字输入/输出管脚
	I2S_DO	O	MFP1	I^2S 输出
	TM2_EXT	I	MFP2	Timer2捕获输入
	PWM_CH1	I/O	MFP3	PWM 1
28	V_{DD}	A	MFP0	I/O 电源, DC 3.3V.

管脚号	管脚名称	管脚类型	MFP*	管脚描述
29	PB.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I2C0时钟
	UART0_TXD	O	MFP3	UART0 发送.
	SD_DAT2	I/O	MFP4	SD/SDH 数据2
30	PB.1	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I2C0 数据
	UART0_RXD	I	MFP3	UART0 接收
	SD_DAT3	I/O	MFP4	SD/SDH 数据3
31	PB.2	I/O	MFP0	通用数字输入/输出管脚
	SPI0_SS	O	MFP1	SPI0 片选
	SD_CMD	I	MFP4	SD/SDH 命令
32	PB.3	I/O	MFP0	通用数字输入/输出管脚
	SPI0_CLK	O	MFP1	SPI0 时钟
	SD_CLK	O	MFP4	SD/SDH 时钟
	SYSCFG[2]	I	MFP0	系统配置 2.
33	PB.4	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MOSI	O	MFP1	SPI0 主机发送端,从机接收端.
	SYSCFG[3]	I	MFP0	系统配置 3.
34	PB.5	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MISO	I	MFP1	SPI0 MISO 主机接收端,从机发送端.
	SD_nCD	I	MFP4	SD/SDH 卡检测端
35	USB_VBUS33	I	MFP0	检测USB是否插入
36	V_{SS}	A	MFP0	地
37	AV_{DDHP}	AP	VDD	解码器 耳机电源, DC 3.3V.

管脚号	管脚名称	管脚类型	MFP*	管脚描述
38	LHPOUT	A	MFP0	耳机左声道输出
39	RHPOUT	A	MFP0	耳机右声道输出.
40	AV_{SSHP}	A	MFP0	耳机解码器地
41	VMID	A	MFP0	耳机参考 电源..
42	$AV_{DDCODEC}$	A	MFP0	解码器电源DC 3.3V.
43	MIC0_P	A	MFP0	麦克风0正端
44	MIC0_N	A	MFP0	麦克风0负端
45	MIC_BIAS	A	MFP0	MIC偏值
46	V_{DD12}	A	MFP0	I/O电源, DC 1.2V
47	PD.4	I/O	MFP0	通用数字输入/输出管脚
	RLINEIN	A	MFP1	解码器右通道输入
48	AV_{DDADC}	A	MFP0	ADC电源, DC 3.3V.
49	AV_{SSADC}	A	MFP0	ADC 地
50	PA.0	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH0	A	MFP1	ADC0
51	PA.1	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH1	A	MFP1	ADC1
52	PA.2	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH2	A	MFP1	ADC2
	I2S_MCLK	O	MFP2	I^2S 主时钟.
53	PA.3	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH3	A	MFP1	ADC 3
	I2S_DI	I	MFP2	I^2S 输入.
54	PA.4	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH4	A	MFP1	ADC4

管脚号	管脚名称	管脚类型	MFP*	管脚描述
	I2S_DO	O	MFP2	I^2S输出
55	V_{DD12}	A	MFP0	I/O 电源, DC 1.2V
56	PB.10	I/O	MFP0	通用数字输入/输出管脚
	SPI1_SS	O	MFP1	SPI1 片选.
	I2C1_SCL	O	MFP2	I2C1 时钟
	UART2_TXD	O	MFP3	UART2.发送
	PWM_CH0	I/O	MFP4	PWM0
57	PB.11	I/O	MFP0	通用数字输入/输出管脚
	SPI1_CLK	O	MFP1	SPI1 时钟
	I2C1_SDA	I/O	MFP2	I2C1 数据
	UART2_RXD	I	MFP3	UART2 接收
	PWM_CH1	I/O	MFP4	PWM1
58	PB.12	I/O	MFP0	通用数字输入/输出管脚
	SPI1_MOSI	O	MFP1	SPI1 主机输出从机输入
	USBH1_D+	I/O	MFP2	USB host-lite1 差分数据线 D+
	UART2_nCTS	I	MFP3	UART2.流控，请除发送
	PWM_CH2	I/O	MFP4	PWM2
59	V_{DD}	A	MFP0	I/O电源 DC 3.3V.
60	PC.11	I/O	MFP0	通用数字输入/输出管脚
	I2S_LRCLK	I/O	MFP1	I^2S左右专用道时钟
	TM3_CNT_OUT	I/O	MFP2	Timer3计数输入或反转输出
	PWM_CH2	I/O	MFP3	PWM2
61	PC.12	I/O	MFP0	通用数字输入/输出管脚
	I2S_BCLK	I/O	MFP1	I^2S 位时钟.
	TM3_EXT	I	MFP2	Timer3 捕获输入

管脚号	管脚名称	管脚类型	MFP*	管脚描述
	PWM_CH3	I/O	MFP3	PWM3
62	V_{DD}	A	MFP0	电源DC 3.3V
63	LDO_CAP	A	MFP0	LDO 输出， 外接退藕电容
64	V_{SS}	A	MFP0	地

4.3.6 NuMicro™ NUC505DS13Y LQFP 64 脚管脚描述

MFP* =多功能管脚(请参考SYS_GPx_MFPL 和 SYS_GPx_MFPH)

PA.0 MFP0 意思 MFP 的值配置为0，SYS_GPA_MFPL[2:0]=0x0

PA.9 MFP5 意思 MFP 的值配置为5，SYS_GPA_MFPH[6:4]=0x5

管脚号	管脚名称	管脚类型	MFP*	管脚描述
1	nRESET	I	MFP0	低复位，正常工作为高，内建上拉电阻
2	ICE_CLK	O	MFP0	仿真时钟. (仿真模式下)
	PD.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I^2C0 时钟
3	ICE_DAT	I/O	MFP0	仿真数据. (仿真模式下)
	PD.1	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I^2C0 数据
4	PB.13	I/O	MFP0	通用数字输入/输出管脚
	SPI1_MISO	I	MFP1	SPI1 主机模式输入，从机模式输出.
	USBH1_D-	I/O	MFP2	USB host 1 差分 D- 线.
	UART2_nRTS	O	MFP3	UART2 流控， 请求发送.
	PWM_CH3	I/O	MFP4	PWM 通道3
5	PB.14	I/O	MFP0	通用数字输入/输出管脚
	USBH1_D+	I/O	MFP1	USB host 1 差分 D+ 线.
	I2C1_SCL	O	MFP2	I2C1 时钟线
6	PB.15	I/O	MFP0	通用数字输入/输出管脚
	USBH1_D-	I/O	MFP1	USB host 1 差分 D- 线.
	I2C1_SDA	I/O	MFP2	I2C1 数据线
7	PC.0	I/O	MFP0	通用数字输入/输出管脚
	SD_CMD	I	MFP1	SD/SDH 命令线.
8	V_{DD}	A	MFP0	I/O 口的电源， DC 3.3V.

9	V_{SS}	A	MFP0	地
10	XT1_IN	I	MFP0	12 MHz 晶振引脚
11	XT1_OUT	O	MFP0	
12	V_{DD12}	A	MFP0	I/O 口的电源， DC 1.2V
13	USB_D-	A	MFP0	USB 接口
14	USB_D+	A	MFP0	
15	AV_{DDUSB}	A	MFP0	USB 电源, DC 3.3V.
16	USB_REXT	A	MFP0	12.1 KΩ 用于 USB.
17	V_{BAT}	A	MFP0	RTC 电池引脚, DC 3.3V.
18	RTC_RPWR	O	MFP0	高电平使能引脚电源控制
19	RTC_nRWAKE	I	MFP0	低电平触发系统电源使能.
20	PA.8	I/O	MFP0	通用数字输入/输出管脚
	SPIM_SS	O	MFP1	SPIM 从机片选.
	I2S_LRCLK	I/O	MFP2	I^2S 左右声道时钟.
	UART1_TXD	O	MFP3	UART1 发送
21	PA.9	I/O	MFP0	通用数字输入/输出管脚
	SPIM_CLK	O	MFP1	SPIM 时钟
	I2S_BCLK	I/O	MFP2	I^2S 位时钟
	UART1_RXD	I	MFP3	UART1 接收
	SYSCFG[0]	I	MFP0	系统配置0
22	PA.10	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MOSI	I/O	MFP1	SPIM 主机发送端,从机接收端.
	I2C1_SCL	O	MFP2	I2C1 时钟
	SD_CLK	O	MFP4	SD/SDH 时钟
	SYSCFG[1]	I	MFP0	系统配置1
23	PA.11	I/O	MFP0	通用数字输入/输出管脚

	SPIM_MISO	I/O	MFP1	SPIM 主机接收,从机发送端
	I2C1_SDA	I/O	MFP2	I2C1 数据.
	SD_CMD	I	MFP4	SD/SDH 命令.
24	PA.12	I/O	MFP0	通用数字输入/输出管脚
	SPIM_D2	I/O	MFP1	SPIM 数据2
	TM0_CNT_OUT	I	MFP2	Timer0 计数输入,或反转输出
25	PA.13	I/O	MFP0	通用数字输入/输出管脚
	SPIM_D3	I/O	MFP1	SPIM 数据3
	TM0_EXT	I	MFP2	Timer0 捕获输入
	SD_nCD	I	MFP4	SD/SDH 卡检测
26	PA.14	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I2C0 时钟线
	SD_DAT0	I/O	MFP4	SD/SDH 数据0
27	PA.15	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I2C0 数据线
	SD_DAT1	I/O	MFP4	SD/SDH 命令线
28	V_{DD}	A	MFP0	I/O 电源, DC 3.3V
29	PB.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I2C0 时钟线
	UART0_TXD	O	MFP3	UART0发送
	SD_DAT2	I/O	MFP4	SD/SDH 数据 2
30	PB.1	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I2C0 数据
	UART0_RXD	I	MFP3	UART0.接收
	SD_DAT3	I/O	MFP4	SD/SDH 数据3

31	PB.2	I/O	MFP0	通用数字输入/输出管脚
	SPI0_SS	O	MFP1	SPI0 片选
	SD_CMD	I	MFP4	SD/SDH 命令
32	PB.3	I/O	MFP0	通用数字输入/输出管脚
	SPI0_CLK	O	MFP1	SPI0 时钟
	SD_CLK	O	MFP4	SD/SDH 时钟
	SYSCFG[2]	I	MFP0	系统配置 2
33	PB.4	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MOSI	O	MFP1	SPI0 主机发送端,从机接收端
	SYSCFG[3]	I	MFP0	系统配置 3
34	PB.5	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MISO	I	MFP1	SPI0 MISO 主机接收端,从机发送端.
	SD_nCD	I	MFP4	SD/SDH 卡检测端
35	PB.6	I/O	MFP0	通用数字输入/输出管脚
	UART1_TXD	O	MFP3	UART1发送
	SD_DAT0	I/O	MFP4	SD/SDH 数据 0
36	PB.7	I/O	MFP0	通用数字输入/输出管脚
	UART1_RXD	I	MFP3	UART1.接收
	SD_DAT1	I/O	MFP4	SD/SDH 数据 1
37	USB_VBUS33	I	MFP0	检测USB是否插入
38	PB.8	I/O	MFP0	通用数字输入/输出管脚
	USB_PWEN	O	MFP1	USB host 电流控制端
	TM1_CNT_OUT	I/O	MFP2	Timer1计数输入,或反转输出
	UART1_nCTS	I	MFP3	UART1流控，清除发送
	SD_DAT2	I/O	MFP4	SD/SDH 数据2

39	PB.9	I/O	MFP0	通用数字输入/输出管脚
	USBH_OVD	I	MFP1	USB host 总线过压检测
	TM1_EXT	I	MFP2	Timer1 捕获输入端
	UART1_nRTS	O	MFP3	UART1 流控，请求发送
	SD_DAT3	I/O	MFP4	SD/SDH 数据 3
40	V_{SS}	A	MFP0	地
41	AV_{DDHP}	AP	VDD	解码器 耳机电源, DC 3.3V.
42	LHPOUT	A	MFP0	耳机左声道输出
43	RHPOUT	A	MFP0	耳机右声道输出.
44	AV_{SSHP}	A	MFP0	耳机解码器地
45	VMID	A	MFP0	耳机参考 电源.
46	$AV_{DDCODEC}$	A	MFP0	解码器电源DC 3.3V.
47	AV_{DDADC}	A	MFP0	ADC电源, DC 3.3V.
48	AV_{SSADC}	A	MFP0	ADC.地
49	PA.0	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH0	A	MFP1	ADC0输入
50	PA.1	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH1	A	MFP1	ADC1.
51	PA.2	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH2	A	MFP1	ADC2
	I2S_MCLK	O	MFP2	I^2S 主时钟.
52	PA.3	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH3	A	MFP1	ADC3.
	I2S_DI	I	MFP2	I^2S 输入
53	PA.4	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH4	A	MFP1	ADC4

	I2S_DO	O	MFP2	I^2S 输出
54	PA.5	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH5	A	MFP1	ADC 5
55	PA.6	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH6	A	MFP1	ADC6.
56	PA.7	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH7	A	MFP1	ADC 7
57	V_{DD12}	A	MFP0	I/O 电源, DC 1.2V
58	PB.10	I/O	MFP0	通用数字输入/输出管脚
	SPI1_SS	O	MFP1	SPI1 片选
	I2C1_SCL	O	MFP2	I2C1 时钟
	UART2_TXD	O	MFP3	UART2.发送
	PWM_CH0	I/O	MFP4	PWM0
59	PB.11	I/O	MFP0	通用数字输入/输出管脚
	SPI1_CLK	O	MFP1	SPI1 时钟
	I2C1_SDA	I/O	MFP2	I2C1 数据
	UART2_RXD	I	MFP3	UART2.接收
	PWM_CH1	I/O	MFP4	PWM1
60	PB.12	I/O	MFP0	通用数字输入/输出管脚
	SPI1_MOSI	O	MFP1	SPI1 主机输出从机输入
	USBH1_D+	I/O	MFP2	USB host-lite1 差分数据线 D+
	UART2_nCTS	I	MFP3	UART2.流控，请除发送
	PWM_CH2	I/O	MFP4	PWM2
61	V_{DD}	A	MFP0	I/O电源, DC 3.3V.
62	V_{DD}	A	MFP0	
63	LDO_CAP	A	MFP0	LDO 输出端，外接退藕电容

64	V_{SS}	A	MFP0	地

4.3.7 NuMicro™ NUC505YO13Y QFN 88 脚管脚描述

MFP* =多功能管脚(请参考 SYS_GPx_MFPL 和 SYS_GPx_MFPH)

PA.0 MFP0 意思 MFP 的值配置为0，SYS_GPA_MFPL[2:0]=0x0

PA.9 MFP5 意思 MFP 的值配置为5，SYS_GPA_MFPH[6:4]=0x5

管脚号	管脚名称	管脚类型	MFP*	管脚描述
1	nRESET	I	MFP0	低复位，正常工作为高，内建上拉电阻
2	ICE_CLK	O	MFP0	仿真时钟. (仿真模式下)
	PD.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I^2C0 时钟
3	ICE_DAT	I/O	MFP0	仿真数据. (仿真模式下)
	PD.1	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I^2C0 数据
4	PB.13	I/O	MFP0	通用数字输入/输出管脚
	SPI1_MISO	I	MFP1	SPI1 主机模式输入，从机模式输出.
	USBH1_D-	I/O	MFP2	USB host 1 差分 D- 线.
	UART2_nRTS	O	MFP3	UART2 流控， 请求发送.
	PWM_CH3	I/O	MFP4	PWM 通道3
5	PB.14	I/O	MFP0	通用数字输入/输出管脚
	USBH1_D+	I/O	MFP1	USB host 1 差分 D+ 线.
	I2C1_SCL	O	MFP2	I2C1 时钟线
6	PB.15	I/O	MFP0	通用数字输入/输出管脚
	USBH1_D-	I/O	MFP1	USB host 1 差分 D- 线.
	I2C1_SDA	I/O	MFP2	I2C1 数据线
7	PC.0	I/O	MFP0	通用数字输入/输出管脚
	SD_CMD	I	MFP1	SD/SDH 命令线.
8	PC.1	I/O	MFP0	通用数字输入/输出管脚

管脚号	管脚名称	管脚类型	MFP*	管脚描述
	SD_CLK	O	MFP1	SD/SDH 时钟
9	PC.2	I/O	MFP0	通用数字输入/输出管脚
	SD_nCD	I	MFP1	SD/SDH 卡检测
10	V_{DD}	A	MFP0	I/O 口的电源， DC 3.3V.
11	PC.3	I/O	MFP0	通用数字输入/输出管脚
12	PC.4	I/O	MFP0	通用数字输入/输出管脚
	SD_DAT0	I/O	MFP1	SD/SDH 数据0.
13	PC.5	I/O	MFP0	通用数字输入/输出管脚
	SD_DAT1	I/O	MFP1	SD/SDH数据1.
14	PC.6	I/O	MFP0	通用数字输入/输出管脚
	SD_DAT2	I/O	MFP1	SD/SDH数据 2
15	V_{DD12}	A	MFP0	I/O 口的电源， DC 1.2V
16	XT1_IN	I	MFP0	外接晶振
17	XT1_OUT	O	MFP0	
18	V_{DD12}	A	MFP0	I/O 口的电源， DC 1.2V
19	USB_D-	A	MFP0	USB 接口
20	USB_D+	A	MFP0	
21	AV_{DDUSB}	A	MFP0	USB 电源, DC 3.3V.
22	USB_REXT	A	MFP0	12.1 KΩ 用于 USB.
23	V_{BAT}	A	MFP0	RTC 电池引脚, DC 3.3V
24	RTC_RPWR	O	MFP0	高电平使能引脚电源控制
25	RTC_nRWAKE	I	MFP0	低电平触发系统电源使能.
26	X32_IN	I	MFP0	外接32768晶振
27	X32_OUT	O	MFP0	
28	PA.8	I/O	MFP0	通用数字输入/输出管脚

管脚号	管脚名称	管脚类型	MFP*	管脚描述
	SPIM_SS	O	MFP1	SPIM 从机片选.
	I2S_LRCLK	I/O	MFP2	I^2S 左右声道时钟.
	UART1_TXD	O	MFP3	UART1 发送.
29	PA.9	I/O	MFP0	通用数字输入/输出管脚
	SPIM_CLK	O	MFP1	SPIM 时钟.
	I2S_BCLK	I/O	MFP2	I^2S 位时钟
	UART1_RXD	I	MFP3	UART1 接收.
	SYSCFG[0]	I	MFP0	系统配置0
30	PA.10	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MOSI	I/O	MFP1	SPIM 主机发送端,从机接收端.
	I2C1_SCL	O	MFP2	I2C1 时钟.
	SD_CLK	O	MFP4	SD/SDH 时钟
	SYSCFG[1]	I	MFP0	系统配置1.
31	PA.11	I/O	MFP0	通用数字输入/输出管脚
	SPIM_MISO	I/O	MFP1	SPIM 主机接收,从机发送端
	I2C1_SDA	I/O	MFP2	I2C1 数据.
	SD_CMD	I	MFP4	SD/SDH 命令.
32	PA.12	I/O	MFP0	通用数字输入/输出管脚
	SPIM_D2	I/O	MFP1	SPIM 数据2
	TM0_CNT_OUT	I	MFP2	Timer0 计数输入,或反转输出
33	PA.13	I/O	MFP0	通用数字输入/输出管脚
	SPIM_D3	I/O	MFP1	SPIM 数据3
	TM0_EXT	I	MFP2	Timer0 捕获输入
	SD_nCD	I	MFP4	SD/SDH 卡检测

管脚号	管脚名称	管脚类型	MFP*	管脚描述
34	PA.14	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I2C0 时钟线
	SD_DAT0	I/O	MFP4	SD/SDH 数据0
35	PA.15	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I2C0 数据线
	SD_DAT1	I/O	MFP4	SD/SDH 数据1
36	PC.7	I/O	MFP0	通用数字输入/输出管脚
	SD_DAT3	I/O	MFP1	SD/SDH 数据 3.
37	PC.8	I/O	MFP0	通用数字输入/输出管脚
	I2S_MCLK	O	MFP1	I^2S 主时钟.
38	PC.9	I/O	MFP0	通用数字输入/输出管脚
	I2S_DI	I	MFP1	I^2S 输入
	TM2_CNT_OUT	I/O	MFP2	Timer2计数输入,或反转输出
	PWM_CH0	I/O	MFP3	PWM0
39	PC.10	I/O	MFP0	通用数字输入/输出管脚
	I2S_DO	O	MFP1	I^2S 输出
	TM2_EXT	I	MFP2	Timer2捕获输入
	PWM_CH1	I/O	MFP3	PWM 1
40	V_{DD}	A	MFP0	I/O 电源, DC 3.3V.
41	PB.0	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SCL	O	MFP2	I2C0时钟
	UART0_TXD	O	MFP3	UART0 发送.
	SD_DAT2	I/O	MFP4	SD/SDH 数据2
42	PB.1	I/O	MFP0	通用数字输入/输出管脚
	I2C0_SDA	I/O	MFP2	I2C0 数据

管脚号	管脚名称	管脚类型	MFP*	管脚描述
	UART0_RXD	I	MFP3	UART0 接收
	SD_DAT3	I/O	MFP4	SD/SDH 数据3
43	PB.2	I/O	MFP0	通用数字输入/输出管脚
	SPI0_SS	O	MFP1	SPI0 片选
	SD_CMD	I	MFP4	SD/SDH 命令
44	PB.3	I/O	MFP0	通用数字输入/输出管脚
	SPI0_CLK	O	MFP1	SPI0 时钟
	SD_CLK	O	MFP4	SD/SDH 时钟
	SYSCFG[2]	I	MFP0	系统配置 2.
45	PB.4	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MOSI	O	MFP1	SPI0 主机发送端,从机接收端.
	SYSCFG[3]	I	MFP0	系统配置 3.
46	PB.5	I/O	MFP0	通用数字输入/输出管脚
	SPI0_MISO	I	MFP1	SPI0 MISO 主机接收端,从机发送端.
	SD_nCD	I	MFP4	SD/SDH 卡检测端
47	PB.6	I/O	MFP0	通用数字输入/输出管脚
	UART1_TXD	O	MFP3	UART1发送
	SD_DAT0	I/O	MFP4	SD/SDH 数据 0
48	PB.7	I/O	MFP0	通用数字输入/输出管脚
	UART1_RXD	I	MFP3	UART1接收
	SD_DAT1	I/O	MFP4	SD/SDH 数据 1
49	USB_VBUS33	I	MFP0	检测USB是否插入
50	PB.8	I/O	MFP0	通用数字输入/输出管脚
	USB_PWEN	O	MFP1	USB host 电流控制端.

管脚号	管脚名称	管脚类型	MFP*	管脚描述
	TM1_CNT_OUT	I/O	MFP2	Timer1计数输入,或反转输出
	UART1_nCTS	I	MFP3	UART1流控，清除发送
	SD_DAT2	I/O	MFP4	SD/SDH 数据2.
51	PB.9	I/O	MFP0	通用数字输入/输出管脚
	USBH_OVD	I	MFP1	USB host 总线过压检测
	TM1_EXT	I	MFP2	Timer1 捕获输入端.
	UART1_nRTS	O	MFP3	UART1 流控，请求发送.
	SD_DAT3	I/O	MFP4	SD/SDH 数据 3.
52	V_{DD}	A	MFP0	I/O电源 DC 3.3V.
53	AV_{DDHP}	AP	VDD	解码器 耳机电源, DC 3.3V.
54	LHPOUT	A	MFP0	耳机左声道输出
55	VCMBF	A	MFP0	内部译码器使用, 保持悬空.
56	RHPOUT	A	MFP0	耳机右声道输出.
57	AV_{SSHP}	A	MFP0	耳机解码器地
58	VMID	A	MFP0	耳机参考 电源..
59	$AV_{DDCODEC}$	A	MFP0	解码器电源DC 3.3V.
60	MIC0_P	A	MFP0	麦克风0正端
61	MIC0_N	A	MFP0	麦克风0负端
62	MIC_BIAS	A	MFP0	MIC偏值
63	V_{DD12}	A	MFP0	I/O电源, DC 1.2V
64	PD.2	I/O	MFP0	通用数字输入/输出管脚
	MIC1_P	A	MFP1	麦克风1正端
65	PD.3	I/O	MFP0	通用数字输入/输出管脚
	MIC1_N	A	MFP1	麦克风1负端

管脚号	管脚名称	管脚类型	MFP*	管脚描述
66	PD.4	I/O	MFP0	通用数字输入/输出管脚
	RLINEIN	A	MFP1	解码器右通道输入
67	AV_{DDADC}	A	MFP0	ADC电源, DC 3.3V.
68	AV_{SSADC}	A	MFP0	ADC 地
69	PA.0	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH0	A	MFP1	ADC0
70	PA.1	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH1	A	MFP1	ADC1
71	PA.2	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH2	A	MFP1	ADC2
	I2S_MCLK	O	MFP2	I^2S 主时钟.
72	PA.3	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH3	A	MFP1	ADC 3
	I2S_DI	I	MFP2	I^2S 输入.
73	PA.4	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH4	A	MFP1	ADC4
	I2S_DO	O	MFP2	I^2S输出
74	PA.5	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH5	A	MFP1	ADC 5
75	PA.6	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH6	A	MFP1	ADC6.
76	PA.7	I/O	MFP0	通用数字输入/输出管脚
	ADC_CH7	A	MFP1	ADC 7
77	V_{DD12}	A	MFP0	I/O 电源, DC 1.2V
78	PB.10	I/O	MFP0	通用数字输入/输出管脚

管脚号	管脚名称	管脚类型	MFP*	管脚描述
	SPI1_SS	O	MFP1	SPI1 片选.
	I2C1_SCL	O	MFP2	I2C1 时钟
	UART2_TXD	O	MFP3	UART2.发送
	PWM_CH0	I/O	MFP4	PWM0
79	PB.11	I/O	MFP0	通用数字输入/输出管脚
	SPI1_CLK	O	MFP1	SPI1 时钟
	I2C1_SDA	I/O	MFP2	I2C1 数据
	UART2_RXD	I	MFP3	UART2 接收
	PWM_CH1	I/O	MFP4	PWM1
80	PB.12	I/O	MFP0	通用数字输入/输出管脚
	SPI1_MOSI	O	MFP1	SPI1 主机输出从机输入
	USBH1_D+	I/O	MFP2	USB host-lite1 差分数据线 D+
	UART2_nCTS	I	MFP3	UART2.流控，请除发送
	PWM_CH2	I/O	MFP4	PWM2
81	V_{DD}	A	MFP0	I/O电源 DC 3.3V.
82	PC.11	I/O	MFP0	通用数字输入/输出管脚
	I2S_LRCLK	I/O	MFP1	I^2S左右专用道时钟
	TM3_CNT_OUT	I/O	MFP2	Timer3计数输入或反转输出
	PWM_CH2	I/O	MFP3	PWM2
83	PC.12	I/O	MFP0	通用数字输入/输出管脚
	I2S_BCLK	I/O	MFP1	I^2S 位时钟.
	TM3_EXT	I	MFP2	Timer3 捕获输入
	PWM_CH3	I/O	MFP3	PWM3
84	PC.13	I/O	MFP0	通用数字输入/输出管脚
	USBH2_D+	I/O	MFP1	USB host-lite 2差分信号 D+

管脚号	管脚名称	管脚类型	MFP*	管脚描述
85	PC.14	I/O	MFP0	通用数字输入/输出管脚
	USBH2_D-	I/O	MFP1	USB host-lite 2差分信号 D-
86	V_{DD}	A	MFP0	电源DC 3.3V
87	LDO_CAP	A	MFP0	LDO 输出， 外接退藕电容
88	V_{SS}	A	MFP0	地

注意：QFN封装底部的散热垫（EPD）应连接到GND。

4.3.8 GPIO 多功能管脚表

MPF0	MPF1	MPF2	MPF3	MPF4	Other	驱动电流
PA.0	ADC_CH0					2~16mA
PA.1	ADC_CH 1					2~16mA
PA.2	ADC_CH 2					2~16mA
PA.3	ADC_CH 3					2~16mA
PA.4	ADC_CH 4					2~16mA
PA.5	ADC_CH 5					2~16mA
PA.6	ADC_CH 6					2~16mA
PA.7	ADC_CH 7					2~16mA
PA.8	SPIM_SS	I2S_LRCLK	UART1_TXD			8mA
PA.9	SPIM_CLK	I2S_BCLK	UART1_RXD		SYSCFG[0]	8mA
PA.10	SPIM_MOSI	I2C1_SCL		SD_CLK	SYSCFG[1]	8mA
PA.11	SPIM_MISO	I2C1_SDA		SD_CMD		8mA
PA.12	SPIM_D2	TM0_CNT_OUT				8mA
PA.13	SPIM_D3	TM0_EXT		SD_nCD		8mA
PA.14		I2C0_SCL		SD_DAT0		4mA
PA.15		I2C0_SDA		SD_DAT1		4mA
PB.0		I2C0_SCL	UART0_TXD	SD_DAT2		4mA
PB.1		I2C0_SDA	UART0_RXD	SD_DAT3		4mA
PB.2	SPI0_SS			SD_CMD		4mA
PB.3	SPI0_CLK			SD_CLK	SYSCFG[2]	4mA
PB.4	SPI0_MOSI				SYSCFG[3]	4mA
PB.5	SPI0_MISO			SD_nCD		4mA
PB.6			UART1_TXD	SD_DAT0		4mA
PB.7			UART1_RXD	SD_DAT1		4mA
PB.8	USB_PWEN	TM1_CNT_OUT	UART1_nCTS	SD_DAT2		4mA
PB.9	USBH_OVD	TM1_EXT	UART1_nRTS	SD_DAT3		4mA
PB.10	SPI1_SS	I2C1_SCL	UART2_TXD	PWM_CH0		4mA
PB.11	SPI1_CLK	I2C1_SDA	UART2_RXD	PWM_CH1		4mA
PB.12	SPI1_MOSI	USBH1_D+	UART2_nCTS	PWM_CH2		8mA
PB.13	SPI1_MISO	USBH1_D-	UART2_nRST	PWM_CH3		8mA
PB.14	USBH1_D+	I2C1_SCL				8mA
PB.15	USBH1_D-	I2C1_SDA				8mA

PC.0	SD_CMD					8mA
PC.1	SD_CLK					8mA
PC.2	SD_nCD					8mA
PC.3						8mA
PC.4	SD_DAT0					8mA
PC.5	SD_DAT1					8mA
PC.6	SD_DAT2					8mA
PC.7	SD_DAT3					8mA
PC.8	I2S_MCLK					4mA
PC.9	I2S_DI	TM2_CNT_OUT	PWM_CH0			4mA
PC.10	I2S_DO	TM2_EXT	PWM_CH1			4mA
PC.11	I2S_LRCLK	TM3_CNT_OUT	PWM_CH2			4mA
PC.12	I2S_BCLK	TM3_EXT	PWM_CH3			4mA
PC.13	USBH2_D+					8mA
PC.14	USBH2_D-					8mA
PD.0		I2C0_SCL			ICE_CLK	4mA
PD.1		I2C0_SDA			ICE_DAT	4mA
PD.2					MIC1_P	2~16mA
PD.3					MIC1_N	2~16mA
PD.4					RLINEIN	2~16mA

4.3.9 GPIO 多功能管脚概述

MFP* = 多功能管脚. (请参考SYS_GPx_MFPL 和 SYS_GPx_MFPH)

PA.0 MFP0 表示 SYS_GPA_MFPL[2:0]=0x0.

PA.9 MFP5 表示 SYS_GPA_MFPH[6:4]=0x5.

组	管脚名	GPIO	MFP*	类型	描述
ADC	ADC0_CH0	PA.0	MFP1	A	ADC0 模拟输入
	ADC0_CH1	PA.1	MFP1	A	ADC1模拟输入
	ADC0_CH2	PA.2	MFP1	A	ADC2模拟输入
	ADC0_CH3	PA.3	MFP1	A	ADC3模拟输入
	ADC0_CH4	PA.4	MFP1	A	ADC4模拟输入
	ADC0_CH5	PA.5	MFP1	A	ADC5模拟输入
	ADC0_CH6	PA.6	MFP1	A	ADC6模拟输入
	ADC0_CH7	PA.7	MFP1	A	ADC7模拟输入
CODEC	MIC1_P	PD.2	MFP1	A	Audio MIC1模拟正极输入
	MIC1_N	PD.3	MFP1	A	Audio MIC1模拟负极输入
	RLINEIN	PD.4	MFP1	A	音频右通道输入
I2C0	I2C0_SCL	PA.14	MFP2	I/O	I2C0时钟管脚
	I2C0_SCL	PB.0	MFP2	I/O	I2C0时钟管脚
	I2C0_SCL	PD.0	MFP2	I/O	I2C0时钟管脚
	I2C0_SDA	PA.15	MFP2	I/O	I2C0数据输入/输出管脚
	I2C0_SDA	PB.1	MFP2	I/O	I2C0数据输入/输出管脚
	I2C0_SDA	PD.1	MFP2	I/O	I2C0数据输入/输出管脚
I2C1	I2C1_SCL	PA.10	MFP2	I/O	I2C1时钟管脚
	I2C1_SCL	PB.10	MFP2	I/O	I2C1时钟管脚
	I2C1_SCL	PB.14	MFP2	I/O	I2C1时钟管脚
	I2C1_SDA	PA.11	MFP2	I/O	I2C1数据输入/输出管脚
	I2C1_SDA	PB.11	MFP2	I/O	I2C1数据输入/输出管脚
	I2C1_SDA	PB.15	MFP2	I/O	I2C1数据输入/输出管脚
I^2S	I2S_MCLK	PA.2	MFP2	O	I^2S主机时钟输出管脚
	I2S_MCLK	PC.8	MFP1	O	I^2S 主机时钟输出管脚
	I2S_BCLK	PA.9	MFP2	I/O	I^2S 位时钟管脚
	I2S_BCLK	PC.12	MFP1	I/O	I^2S 位时钟管脚

组	管脚名	GPIO	MFP*	类型	描述
	I2S_LRCLK	PA.8	MFP2	I/O	I²S 左右通道管脚
	I2S_LRCLK	PC.11	MFP1	I/O	I²S 左右通道管脚
	I2S_DO	PA.4	MFP2	O	I²S数据输出
	I2S_DO	PC.10	MFP1	O	I²S数据输出
	I2S_DI	PA.3	MFP2	I	I²S数据输入
	I2S_DI	PC.9	MFP1	I	I²S 数据输入
ICE	ICE_CLK	PD.0	MFP0	I	串口调试时钟管脚
	ICE_DAT	PD.1	MFP0	I/O	串口调试数据管脚
PWM	PWM_CH0	PB.10	MFP4	I/O	PWM 输出/捕捉输入管脚
	PWM_CH0	PC.9	MFP3	I/O	PWM 输出/捕捉输入管脚
	PWM_CH1	PB.11	MFP4	I/O	PWM 输出/捕捉输入管脚
	PWM_CH1	PC.10	MFP3	I/O	PWM 输出/捕捉输入管脚
	PWM_CH2	PB.12	MFP4	I/O	PWM 输出/捕捉输入管脚
	PWM_CH2	PC.11	MFP3	I/O	PWM 输出/捕捉输入管脚
	PWM_CH3	PB.13	MFP4	I/O	PWM 输出/捕捉输入管脚
	PWM_CH3	PC.12	MFP3	I/O	PWM 输出/捕捉输入管脚
SPIM	SPIM_SS	PA.8	MFP1	O	SPIM从选择管脚
	SPIM_CLK	PA.9	MFP1	O	SPIM串行时钟管脚
	SPIM_MOSI	PA.10	MFP1	I/O	SPIM MOSI (主出，从入) 管脚
	SPIM_MISO	PA.11	MFP1	I/O	SPIM MISO (主入，从出) 管脚
	SPIM_D2	PA.12	MFP1	I/O	SPIM 数据2
	SPIM_D3	PA.13	MFP1	I/O	SPIM 数据3
SPI0	SPI0_SS	PB.2	MFP1	O	SPI0 从选择管脚
	SPI0_CLK	PB.3	MFP1	O	SPI0 串行时钟管脚
	SPI0_MOSI	PB.4	MFP1	I/O	SPI0 MOSI (主出，从入) 管脚
	SPI0_MISO	PB.5	MFP1	I/O	SPI0 MISO (主入，从出) 管脚
SPI1	SPI1_SS	PB.10	MFP1	O	SPI1从选择管脚
	SPI1_CLK	PB.11	MFP1	O	SPI1串行时钟管脚
	SPI1_MOSI	PB.12	MFP1	I/O	SPI1 MOSI (主出，从入) 管脚
	SPI_MISO	PB.13	MFP1	I/O	SPI1 MISO (主入，从出) 管脚
Timer	TM0_CNT_OUT	PA.12	MFP2	I/O	Timer0事件计数器输入/翻转输出管脚

组	管脚名	GPIO	MFP*	类型	描述
	TM0_EXT	PA.13	MFP2	I	Timer0外部计数器输入管脚
	TM1_CNT_OUT	PB.8	MFP2	I/O	Timer1事件计数器输入/翻转输出管脚
	TM1_EXT	PB.9	MFP2	I	Timer1外部计数器输入管脚
	TM2_CNT_OUT	PC.9	MFP2	I/O	Timer2事件计数器输入/翻转输出管脚
	TM2_EXT	PC.10	MFP2	I	Timer2外部计数器输入管脚
	TM3_CNT_OUT	PC.11	MFP2	I/O	Timer3事件计数器输入/翻转输出管脚
	TM3_EXT	PC.12	MFP2	I	Timer3外部计数器输入管脚
UART0	UART0_RXD	PB.1	MFP3	I	UART0数据接收输入管脚
	UART0_TXD	PB.0	MFP3	O	UART0数据输出管脚
UART1	UART1_RXD	PA.9	MFP3	I	UART1数据接收输入管脚
	UART1_RXD	PB.7	MFP3	I	UART1数据接收输入管脚
	UART1_TXD	PA.8	MFP3	O	UART1数据输出管脚
	UART1_TXD	PB.6	MFP3	O	UART1数据输出管脚
	UART1_nCTS	PB.8	MFP3	I	UART1清零发送输入管脚
	UART1_nRTS	PB.9	MFP3	O	UART1请求发送输出管脚
UART2	UART2_RXD	PB.11	MFP3	I	UART2数据接收输入管脚
	UART2_TXD	PB.10	MFP3	O	UART2数据输出管脚
	UART2_nCTS	PB.12	MFP3	I	UART2清零发送输入管脚
	UART2_nRTS	PB.13	MFP3	O	UART2请求发送输出管脚
USB Host Lite	USB_PWEN	PB.8	MFP1	O	USB host 电流控制端
	USBH_VOD	PB.9	MFP1	I	USB host 总线过压检测
	USBH0_D+	PC.13	MFP1	A	USB host-lite 0差分信号 D+
	USBH0_D-	PC.14	MFP1	A	USB host-lite 0差分信号 D-
	USBH1_D+	PB.12	MFP2	A	USB host-lite 1差分信号 D+
	USBH1_D+	PB.14	MFP1	A	USB host-lite 1差分信号 D+
	USBH1_D-	PB.13	MFP2	A	USB host-lite 1差分信号 D-
	USBH1_D-	PB.15	MFP1	A	USB host-lite 1差分信号 D-
SDH	SD_CLK	PA.10	MFP4	O	SD/SDH 时钟
	SD_CLK	PB.3	MFP4	O	SD/SDH 时钟
	SD_CLK	PC.1	MFP1	O	SD/SDH 时钟
	SD_CMD	PA.11	MFP4	O	SD/SDH 命令

组	管脚名	GPIO	MFP*	类型	描述
	SD_CMD	PB.2	MFP4	O	SD/SDH 命令
	SD_CMD	PC.0	MFP1	O	SD/SDH 命令
	SD_nCD	PA.13	MFP4	I	SD/SDH 卡检测
	SD_nCD	PB.5	MFP4	I	SD/SDH 卡检测
	SD_nCD	PC.2	MFP1	I	SD/SDH 卡检测
	SD_DAT0	PA.14	MFP4	I/O	SD/SDH 数据0
	SD_DAT0	PB.6	MFP4	I/O	SD/SDH 数据0
	SD_DAT0	PC.4	MFP1	I/O	SD/SDH 数据0
	SD_DAT1	PA.15	MFP4	I/O	SD/SDH 数据1
	SD_DAT1	PB.7	MFP4	I/O	SD/SDH 数据1
	SD_DAT1	PC.5	MFP1	I/O	SD/SDH 数据1
	SD_DAT2	PB.0	MFP4	I/O	SD/SDH 数据2
	SD_DAT2	PB.8	MFP4	I/O	SD/SDH 数据2
	SD_DAT2	PC.6	MFP1	I/O	SD/SDH 数据2
	SD_DAT3	PB.1	MFP4	I/O	SD/SDH 数据3
	SD_DAT3	PB.9	MFP4	I/O	SD/SDH 数据3
	SD_DAT3	PC.7	MFP1	I/O	SD/SDH 数据3

表 4.3-1 NUC505 GPIO 多功能表

5 框图

5.1 NuMicro™ NUC505 功能框图

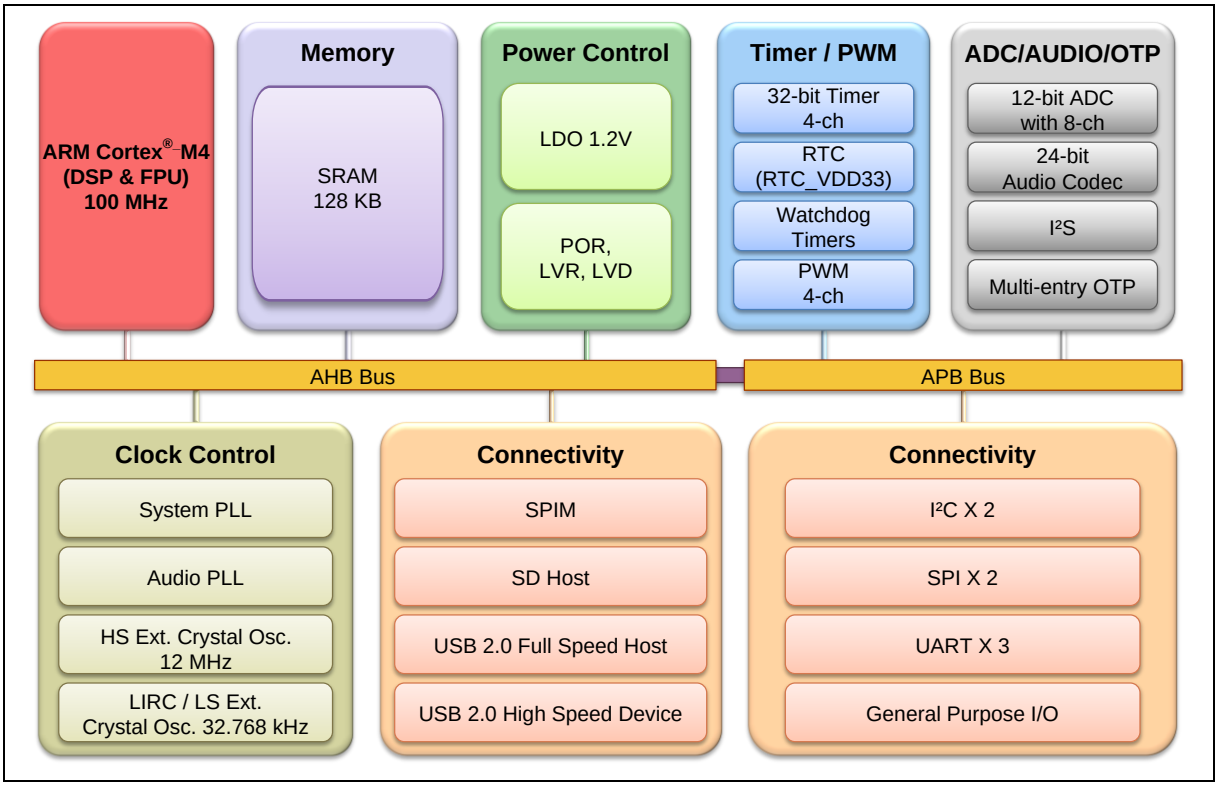


图 5.1-1 NuMicro™ NUC505功能框图

6 功能描述

6.1 ARM® Cortex®-M4 内核

Cortex®-M4处理器是32位可配置的多级流水线RISC处理器，其内置三个AMBA AHB-Lite 接口可以达到最好的并行处理性能，并包括NVIC组件。处理器配有可选硬件调试功能，可执行 Thumb指令并与其他Cortex-M 系列处理器兼容。支持两种模式–Thread 模式和Handler模式。意外时系统进入Handler模式。只能在handle 模式下处理意外返回。系统复位时，进入Thread模式。Thread模式也可由异常返回时进入。Cortex®-M4F具有Cortex®-M4处理器所有性能并另外包含浮点运算器。NUC505内置Cortex®-M4F处理器。在本文档中，所有名为Cortex®-M4的地方都指代Cortex®-M4 和Cortex®-M4F处理器。下图为处理器的功能控制图。

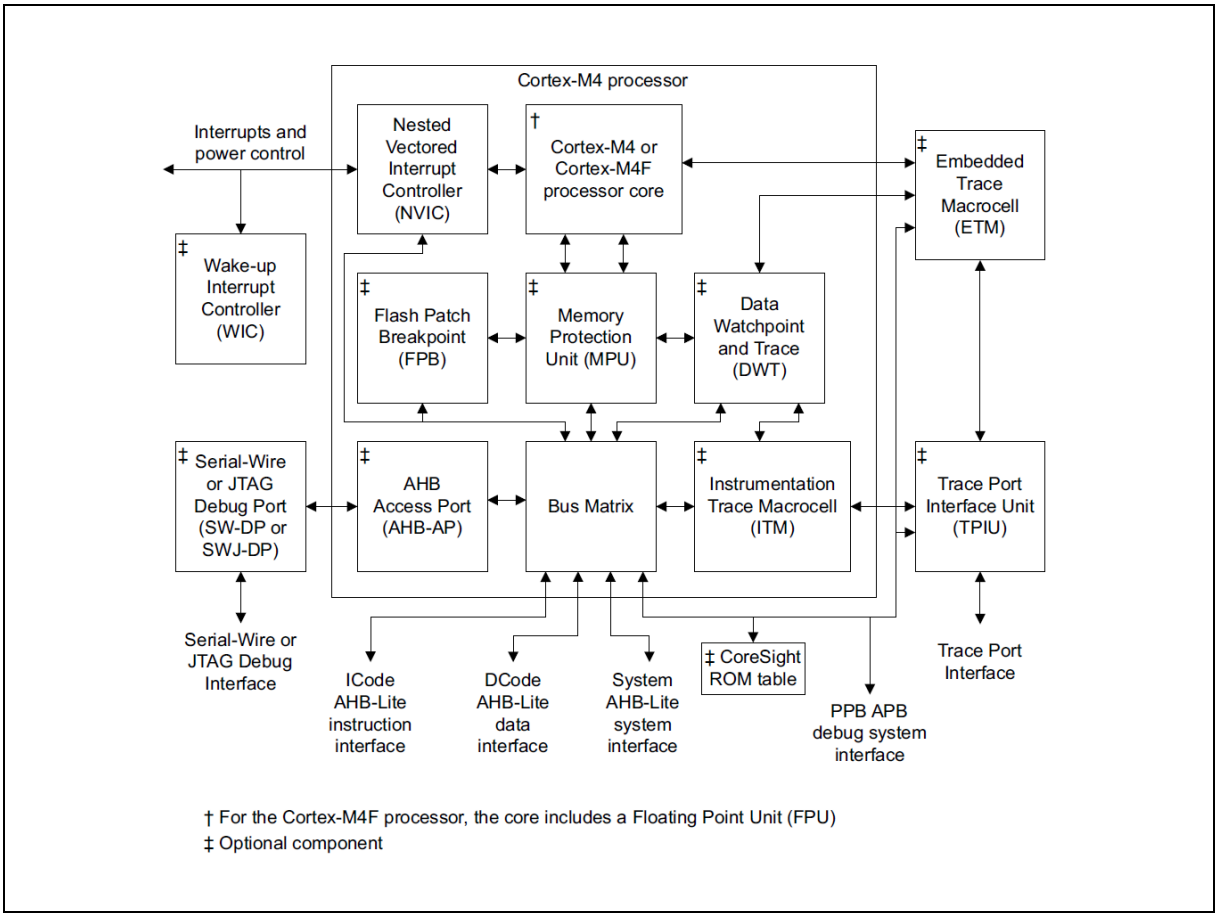


图 6.1-1 Cortex®-M4 模块框图

Cortex®-M4处理器特性：

- 低延时中断处理的低门数处理器内核, 支持：
  - Thumb指令集子集，*参看ARMv7-M架构参考手册*
  - 堆栈指针(SP)

- 硬件整数除法指令，SDIV 和 UDIV
- Handler和 Thread模式
- Thumb和 Debug状态
- 支持LDM, STM, PUSH, POP可中断-连续指令，用于低延时中断
- 低延时中断服务处理程序*(ISR)*进入和退出时，自动保存和恢复处理器状态
- 支持ARMv6 大端字节保持不变或小端数据访问
- 支持ARMv6数据不对齐访问

- Cortex®-M4F中浮点运算单元(FPU)提供:
  - 用于单精度(浮点C)数据处理的32-位指令
  - 用于提高精度的乘法累加指令(融合MAC)
  - 为变换、加法、减法、乘法运算在硬件上提供可选的加法器,除法器和平方根器
  - 为反常值和所有IEEE 舍入模式提供硬件支持
  - 32个32-位单精度寄存器,同时也可以寻址为16个双字节(double-word)寄存器
  - 分离式三级管道
- 嵌套向量中断控制器(NVIC)与处理器内核高度集成化, 以实现低延时中断处理。特性包括:
  - 外部中断。 可配置中断从1到240 (NUC505只有32个中断可配置).
  - 位优先级，配置级别从位3到位7
  - 中断优先级动态可调
  - 通过使能优先或普通中断级别选择, 可将优先级分组
  - 支持尾链(tail-chaining)和迟到中断处理,以实现背对背(back-to-back)中断处理从而无需在反复中断时不停进行保存和恢复的动作
  - 处理器状态会在中断入口时自动保存，在中断退出时自动恢复
  - 唤醒中断控制（WIC），支持超低功耗睡眠模式
- 内存保护单元(MPU)。可选的MPU用于内存保护，包括：
  - 8个内存区
  - 子区禁用(SRD)，以支持内存区域的有效应用
  - 提供备份区用于生成默认内存表属性
- 低耗调试方法，特性有：
  - 调试可访问系统中所有内存和寄存器，包括内存映射设备、内核中止时内核寄存器以及SYSRESETn宣告有效时的调试控制寄存器
  - 支持串行线调试接口（SW-DP）或串行线JTAG调试接口（SWJ-DP），NUC505只支持SW-DP
  - 可选闪存地址重载和断点模块(FPB)用于实现断点和闪存地址重载

- 可选数据观测和跟踪模块(DWT)用于实现监视点、数据跟踪和系统性能测试和统计

- 总线接口：
  - 三条高级高性能Lite总线(AHB-Lite)接口：ICode, Dcode, 和 System bus 接口
  - 基于Advanced Peripheral Bus (APB)的Private Peripheral Bus (PPB)
  - 支持bit-band，包括bit-band读写操作
  - 内存访问地址对齐
  - 为写数据提供写缓存
  - 多处理器系统中支持独占访问

6.2 系统管理

6.2.1 概述

系统管理包括以下几个部分:

- 系统复位
- 系统内存映射
- 总线仲裁算法
- 全局控制寄存器
- 系统时钟 (Systick)
- 嵌套式向量中断控制器(NVIC)
- 系统控制寄存映射和描述

6.2.2 系统复位

- 硬件复位
  - 上电复位 (POR)
  - nRESET 引脚低电平复位 (nRST)
  - 看门狗复位 (WDT)
  - 低电压复位 (LVR)
- 软件复位
  - SYSRESETREQ (AIRCR[2])
  - CPU Reset (SYS_IPRST0[0])
  - CHIPRST (SYS_IPRST0 [1])

注1: SYSRESETREQ (AIRCR[2])可以复位整个芯片包括所有外围设备, 但是不复位 SPIM 功能、向量表映射设置模块和 PA.8~PA.15 多功能设置

注2: CPU Rest (SYS_IPRST0[0]) 只可以复位 CPU

注3: CHIPRST (SYS_IPRST0[1]) 可以复位整个芯片包括所有外设

6.2.3 系统上电设置

当芯片上电或是复位时需要配置上电设置让芯片进入指定状态。由于在复位期间每个引脚在上电设置时都有对应的内部上拉电阻，如果应用需要设置为0，那么在对应的引脚上需要增加合适的下拉。

PB.4	PB.3	PA.10	PA.9	描述	寄存器映射
1	1	1	1	从内部的 MCP SPI Flash 启动	SYS_BOOTSET[3:0]
1	1	1	0	从 USB 启动	SYS_BOOTSET[3:0]
1	1	0	1	从外部 SPI Flash 启动	SYS_BOOTSET[3:0]
1	0	1	1	从 ICP 模式启动	SYS_BOOTSET[3:0]
0	1	1	1	内部 SPI Flash SWD/ICE 模式	SYS_BOOTSET[3:0]
0	1	1	0	外部 SPI Flash SWD/ICE 模式	SYS_BOOTSET[3:0]

表 6.2-1 系统上电设置指南

6.2.4 系统电源分配

本芯片中电源分配分为五个部分:

- 由 $AV_{DDCODEC,}$ AV_{DDHP}, 和 AV_{SSHP} 提供音频编码器电源，为音频编码工作提供电源
- 由AV_{DDADC} 和 AV_{SSADC} 模数转换的电源，为ADC工作提供电源
- 由V_{DD} 和 V_{SS} 提供的数字电源，提供一个固定的1.2V数字电源，用于数字部分和I/O 引脚工作
- AV_{DDUSB}提供给USB的电源，用于USB模块传输操作
- V_{BAT} 提供给电池的电源， 用于RTC和80字节的备用寄存器

内部的电压调节器、LDO和V_{DD}，要求在相应的引脚上外接电容，并尽量靠近引脚摆放。模拟电源($AV_{DDCODEC}$ 和 AV_{DDADC})要与数字电源(V_{DD})是同一个电压准位。下图列出NuMicro™ NUC505的电源分布。

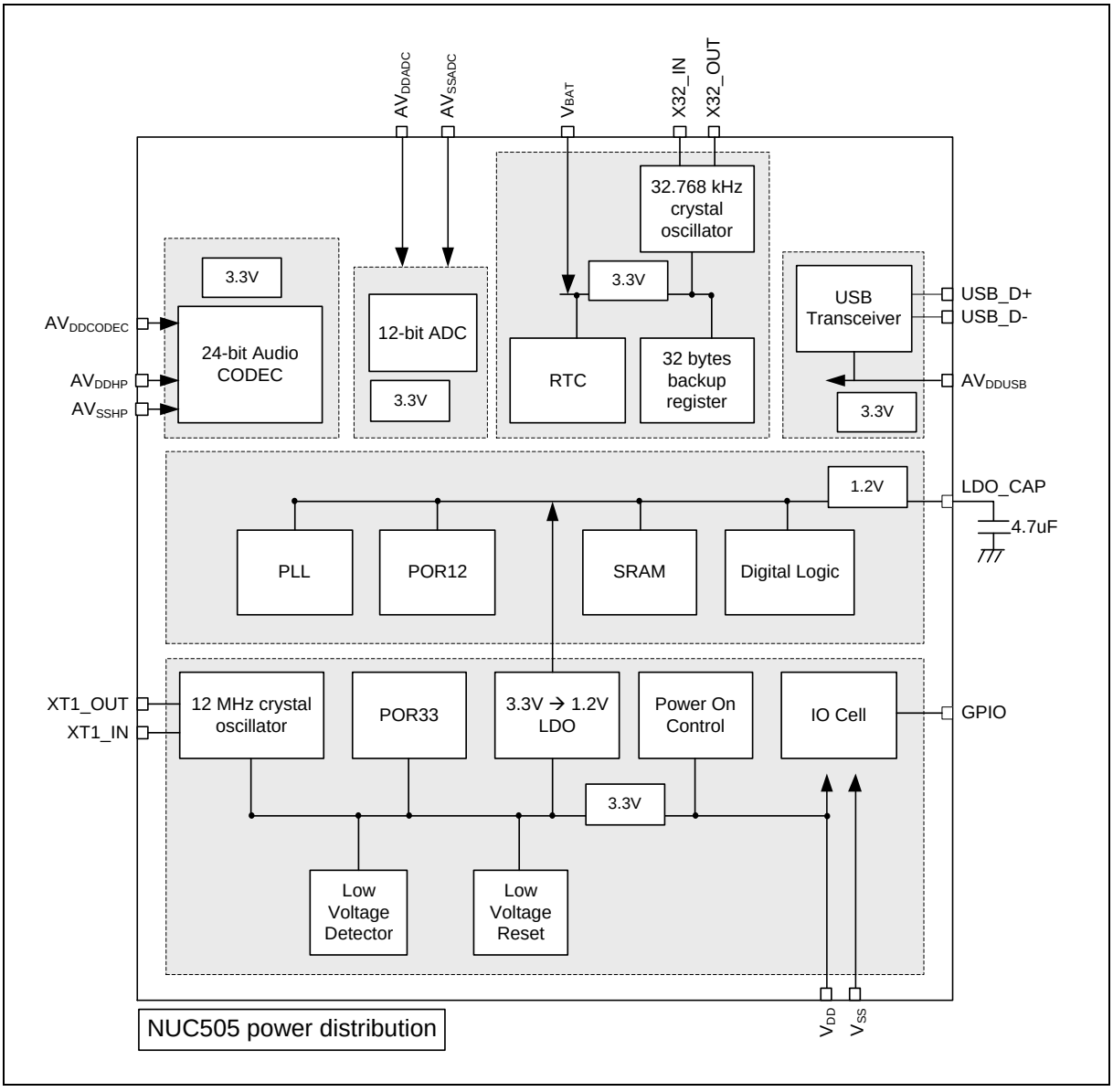


图 6.2-1 NuMicro™ NUC505 电源分布框图

6.2.5 系统内存映射

NUC505提供了4G字节寻址空间。片内控制器的内存地址分配如表6.2-2所示。对片上外设的详细寄存器定义、内存空间和编程指南，将在每个章节中详细描述。 NUC505只支持小端数据格式。

地址空间	符号	控制器
内存空间		
0x1FFF_0000 – 0x1FFF_7FFF	IBR_BA	内部启动 ROM (IBR) 内存空间

0x2000_0000 – 0x2000_7FFF	SRAM1_BA	SRAM1内存空间(32K 字节)
0x2000_8000 – 0x2000_FFFF	SRAM2_BA	SRAM2内存空间(32K 字节)
0x2001_0000 – 0x2001_7FFF	SRAM3_BA	SRAM3内存空间(32K 字节)
0x2001_8000 – 0x2001_FFFF	SRAM4_BA	SRAM4内存空间(32K 字节)
0x0000_0000 – 0x0FFF_FFFF	FLASH_BA	SPI Flash/ROM内存空间
AHB外设控制器空间(0x4000_0000 ~0x4000FFFF)		
0x4000_0000 – 0x4000_01FF	GCR_BA	全局控制寄存器
0x4000_0200 – 0x4000_02FF	CLK_BA	时钟控制寄存器
0x4000_7000 – 0x4000_7FFF	SPIM_BA	SPIM控制寄存器
0x4000_9000 – 0x4000_9FFF	USBD_BA	USB Device控制寄存器
0x4000_A000 – 0x4000_AFFF	SDH_BA	SDH控制寄存器
0x4000_B000 – 0x4000_BFFF	USBH_BA	USB Host控制寄存器
APB控制器空间(0x400E_0000~0x400E_FFFF)		
0x400E_1000 – 0x400E_1FFF	SPI1_BA	SPI1 主/从 控制寄存器(SPI1)
0x400E_2000 – 0x400E_2FFF	ADC_BA	ADC 控制寄存器
0x400E_3000 – 0x400E_3FFF	GPIO_BA	GPIO 控制寄存器
0x400E_4000 – 0x400E_4FFF	I2C0_BA	I2C0 接口 控制寄存器
0x400E_5000 – 0x400E_5FFF	I2C1_BA	I2C1 接口控制寄存器
0x400E_6000 – 0x400E_6FFF	PWM_BA	PWM 控制寄存器
0x400E_7000 – 0x400E_7FFF	RTC_BA	实时时钟(RTC)控制寄存器
0x400E_8000 – 0x400E_8FFF	I2S_BA	I^2S控制寄存器
0x400E_9000 – 0x400E_9FFF	SPI0_BA	SPI0主/从 控制寄存器(SPI0)
0x400E_A000 – 0x400E_AFFF	Timer01_BA	Timer0/Timer1控制寄存器
0x400E_B000 – 0x400E_BFFF	Timer23_BA	Timer2/Timer3控制寄存器
0x400E_C000 – 0x400E_CFFF	UART0_BA	UART0控制寄存器(正常速度)
0x400E_D000 – 0x400E_DFFF	UART1_BA	UART1控制寄存器(高速)
0x400E_E000 – 0x400E_EFFF	UART2_BA	UART2控制寄存器(高速)
0x400E_F000 – 0x400E_FFFF	WDT_BA	WDT 接口控制寄存器
系统控制器空间(0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SCS_BA	系统计时器控制寄存器
0xE000_E100 – 0xE000_ECFF	SCS_BA	外部中断控制寄存器
0xE000_ED00 – 0xE000_ED8F	SCS_BA	系统控制寄存器

表 6.2-2片上控制器地址空间分配

6.2.6 SRAM 内存组织

NUC505内置128K 字节SRAM，SRAM分成四个bank：SRAM bank0、SRAM bank1、SRAM bank2和SRAM bank3。每个bank有32K字节地址空间，可以同时访问。

- 共支持128 K字节SRAM
- 支持字节/半字/字 写操作
- 支持固定32K字节SRAM banks独立访问
- 支持地址重映射到0x1FF0_0000
- 使用向量重映射模块可以使128K字节SRAM任意内存块重映射到0x0000_0000

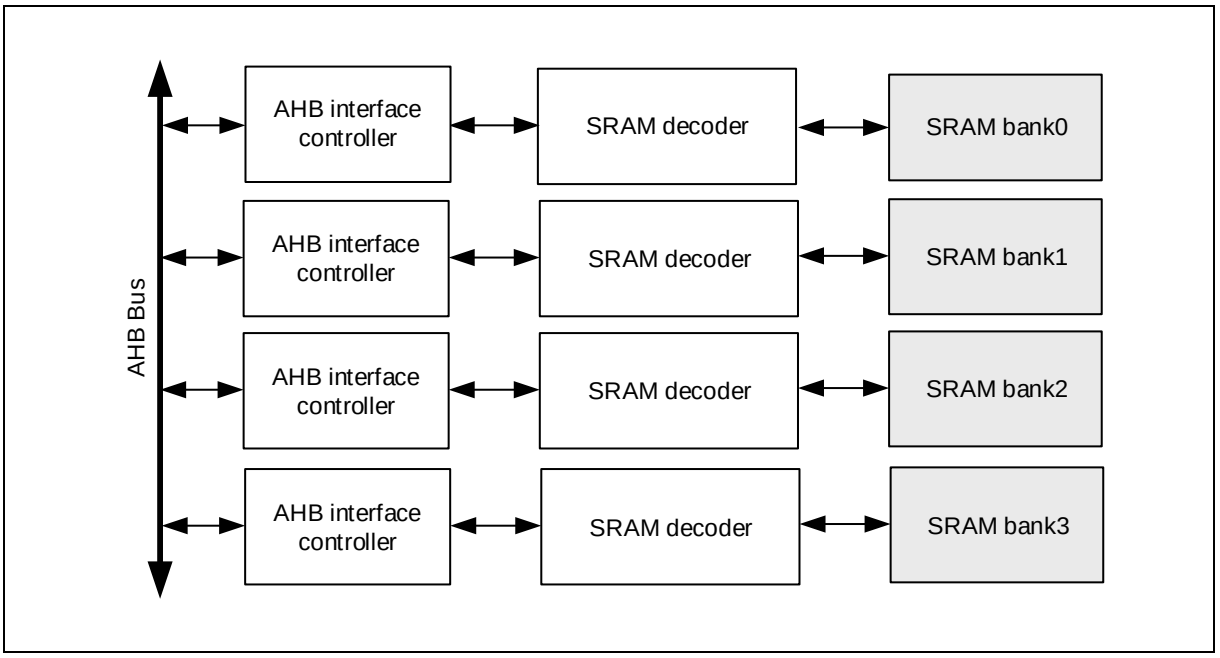


图 6.2-2 SRAM 模块图

图 6.2-3列出NUC505的SRAM组织架构。M451中共有四个SRAM bank，每个bank可寻址范围为32K字节。bank0地址空间从0x2000_0000 到 0x2000_7FFF。bank1 地址空间从 0x2000_8000 到0x2000_FFFF。Bank2 地址空间从 0x2001_0000 到0x2001_7FFF。Bank3 地址空间从 0x2001_8000到0x2001_FFFF。

每个bank的地址可以从0x2000_0000 映射到 0x1FF0_0000。CPU可以通过地址0x2000_0000 到

0x2000_7FFF或0x1FF0_0000 到 0x1FF0_7FFF访问SRAM bank0，可以通过地址0x2000_8000 到 0x2000_FFFF 或 0x1FF0_8000 到 0x1FF0_FFFF访问SRAM bank1，可以通过地址0x2001_0000 到 0x2001_7FFF或0x1FF1_0000 到 0x1FF1_7FFF访问SRAM bank2，也可以通过地址0x2001_8000 到 0x2001_FFFF 或 0x1FF1_8000 到 0x1FF1_FFFF访问SRAM bank3。

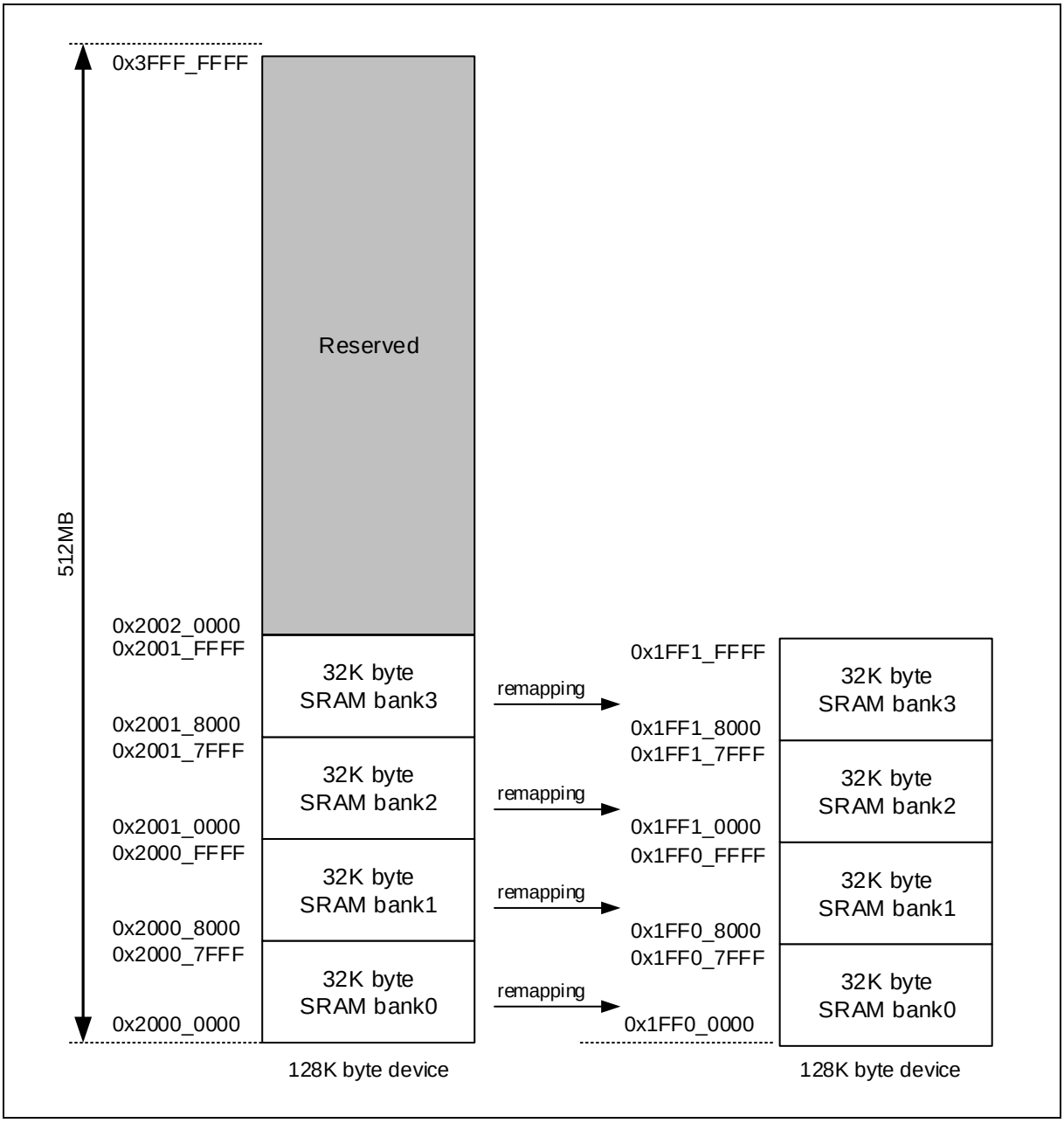


图 6.2-3 SRAM内存组织架构

图 6.2-4为矢量表模块框图。128K字节SRAM中的任意一个内存块都能被重映射到SPI flash中其起始地址是0x0000_0000。内存块的地址和大小有SYS_RVMPADDR[31:0]和SYS_RVMPLEN[31:24]两个寄存

器控制。SYS_RVMPADDR 指示内存的起始地址，SYS_RVMPLEN表明内存块的大小（单位为1K字节）

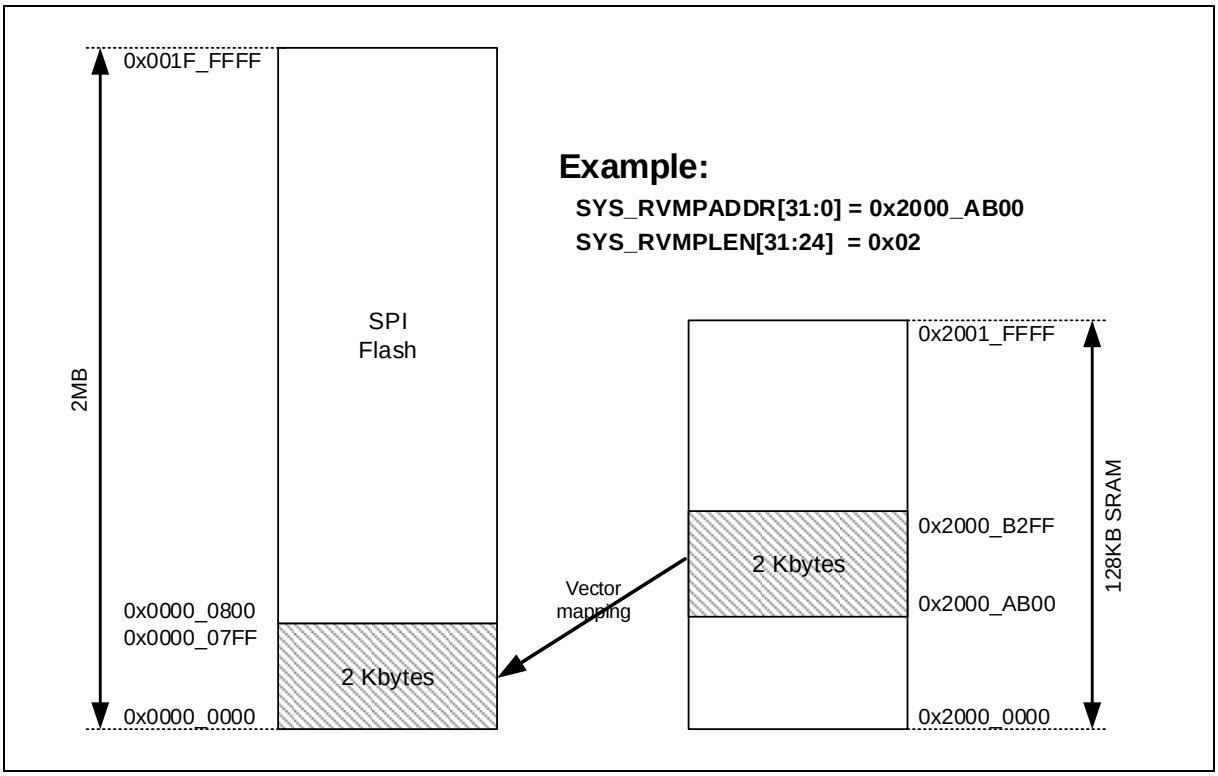


图 6.2-4 矢量表模块框图

6.2.7 AHB 总线仲裁

NUC505内部总线是AHB-Compliant总线可以连接到标准的AHB主/从接口。NUC505的AHB仲裁机制为同时访问提供了两种仲裁算法可供选择。一种是固定优先级模式，另一种是循环优先级模式。通过***SYS_AHBCTL***寄存器中**PRISEL**位选择相应的模式和类型。

6.2.7.1 固定优先级模式

如果**PRISEL** = 0则为固定优先级模式。 在AHB总线上片上主控模块的控制权的优先顺序如表6.2-3

优先级 (PRISEL = 0)	AHB 总线优先级
1 (最低)	Cortex-M4 I
2	Cortex-M4 D
3	Cortex-M4 System

4	SPIM
5	USBD
6	USBH
6	SDH
8 (最高)	I^2S

表 6.2-3固定优先级模式下AHB总线优先级顺序 如果两个或是更多的主控模块同时请求访问AHB总线时，高优先级则被允许访问AHB总线。

优先级 (PRISEL = 0)	**AHB 总线优先级**
1 (最低)	Cortex-M4 I
2	Cortex-M4 D
3	Cortex-M4 System
4	SPIM
5	USBD
6	USBH
6	SDH
8 (最高)	I^2S

表 6.2-3固定优先级模式下AHB总线优先级顺序

在固定优先级中SPI flash控制器通常是最低优先级（除CPU接口外）。NUC505提供一种机制可提高CPU请求为最高优先级。如果**CPUHPRI**位（***SYS_AHBCTL***控制寄存器bit4）置1，在没有屏蔽外部IRQ时**PRISTS**位（***SYS_AHBCTL***控制寄存器bit5）会自动置1。在这种情况下，ARM核成为最高优先级访问AHB总线。

程序员可以通过写1到**PRISTS**恢复原始优先级顺序。比如这个动作在中断服务程序结束时。注在**CPUHPRI**为1时**PRISTS**只能通过外部中断自动置1。这个并不影响程序员直接向**PRISTS**写1来提高ARM核的优先级。

6.2.7.2 循环优先级模式

如果**PRISEL** = 1则为循环优先级模式。AHB总线仲裁机制使用循环方法让每个主控模块按顺序获得使用权。也就是说请求者在获得访问权限后由最高优先级变成最低优先级。

6.2.7.3 循环规则举例

在AHB主控总线上默认优先级顺序是：I^2S>SDH > USBH > USBD >SPIM >M4(S) > M4(D) > M4(I).

6.2.8 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移地址	R/W	描述	复位值
GCR基地址: **GCR_BA = 0x4000_0000**				
SYS_PDID	GCR_BA+0x00	R	器件ID 寄存器	0x0055_0505
SYS_BOOTSET	GCR_BA+0x04	R/W	系统上电配置寄存器	0x0000_00XX
SYS_IPRST0	GCR_BA+0x08	R/W	外设复位控制寄存器 0	0x0000_0009
SYS_LVDCTL	GCR_BA+0x0C	R/W	低电压检测控制寄存器	0x0000_0009
SYS_WAKEUP	GCR_BA+0x10	R/W	唤醒控制和状态 寄存器	0x0000_0000
SYS_IPRST1	GCR_BA+0x14	R/W	外设复位控制寄存器1	0x0000_0000
SYS_NMICTL	GCR_BA+0x18	R/W	不可屏蔽中断控制寄存器	0x0000_0000
SYS_RSTSTS	GCR_BA+0x1C	R/W	复位状态寄存器	0x0000_0000
SYS_AHBCTL	GCR_BA+0x20	R/W	AHB 总线控制寄存器	0x0000_0000
SYS_GPA_MFPL	GCR_BA+0x30	R/W	GPIOA低字节复用功能控制寄存器	0x0000_0000
SYS_GPA_MFPH	GCR_BA+0x34	R/W	GPIOA高字节复用功能控制寄存器	0x0000_0000
SYS_GPB_MFPL	GCR_BA+0x38	R/W	GPIOB 低字节复用功能控制寄存器	0x0000_0000
SYS_GPB_MFPH	GCR_BA+0x3C	R/W	GPIOB高字节复用功能控制寄存器	0x0000_0000
SYS_GPC_MFPL	GCR_BA+0x40	R/W	GPIOC 低字节复用功能控制寄存器	0x0000_0000
SYS_GPC_MFPH	GCR_BA+0x44	R/W	GPIOC高字节复用功能控制寄存器	0x0000_0000
SYS_GPD_MFPL	GCR_BA+0x48	R/W	GPIOD 低字节复用功能控制寄存器	0x0000_0000
SYS_LVMPADDR	GCR_BA+0x50	R/W	加载 VECMAP 地址参数控制寄存器	0x1FFF_0000
SYS_LVMPLEN	GCR_BA+0x54	R/W	加载 VECMAP 长度参数控制寄存器	0x0000_0008
SYS_RVMPADDR	GCR_BA+0x58	R	实际 VECMAP地址参数寄存器	0x1FFF_0000
SYS_RVMPLEN	GCR_BA+0x5C	R/W	实际 VECMAP长度参数控制寄存器	0x0800_0000
SYS_EPADPUEN	GCR_BA+0x6C	R/W	内嵌 SPI Flash Pad 上拉 使能控制寄存器	0x0000_001F
SYS_GPADS	GCR_BA+0x70	R/W	GPIOA 驱动能力控制寄存器	0x0000_0000
SYS_GPAIBE	GCR_BA+0x74	R/W	GPIOA 输入缓冲使能控制寄存器	0xFFFF_0000
SYS_GPBIBE	GCR_BA+0x78	R/W	GPIOB输入缓冲使能控制寄存器	0xFFFF_0000
SYS_GPCIBE	GCR_BA+0x7C	R/W	GPIOC输入缓冲使能控制寄存器	0xFFFF_0000
SYS_GPDIBE	GCR_BA+0x80	R/W	GPIOD输入缓冲使能控制寄存器	0xFFFF_0000
SYS_GPDDS	GCR_BA+0x84	R/W	GPIOD驱动能力控制寄存器	0x0000_0000

SYS_RSTDBCNT	GCR_BA+0x100	R/W	外部 nRESET 引脚 抖动计数 控制寄存器	0x0000_04B0
SYS_RSTDBEN	GCR_BA+0x104	R/W	外部 nRESET 引脚 抖动 控制寄存器	0x0000_0000

6.2.9 寄存器描述

器件ID 寄存器(SYS_PDID)

寄存器	偏移地址	R/W	描述	复位值
SYS_PDID	GCR_BA+0x00	R	器件ID 寄存器	0x0055_0505

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

位	描述	
[31:24]	**Reserved**	保留
[23:0]	**PDID**	**产品器件识别码（只读）** 该寄存器为设备的器件号码. 软件可以读取该寄存器来识别所使用的器件。

系统上电配置寄存器(SYS_BOOTSET)

该寄存器为软件识别芯片上电设置提供特殊信息。BOOTSET[3:0]反映了上电设置引脚的状态。可以通过软件修改它们。

寄存器	偏移地址	R/W	描述	复位值
SYS_BOOTSET	GCR_BA+0x04	R/W	系统上电配置寄存器	0x0000_00XX

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

位	描述	
[31:4]	Reserved	保留
[3:0]	BOOTSET	**系统模式配置** 0110 =从带有外部SPI Flash的ICE模式启动 0111 =从带有SPI Flash的ICE模式启动 1011 =从ICP模式启动 1101 = 从外部 SPI Flash启动 1110 = 从 USB启动 1111 = 从 SPI Flash启动 **注:** 如果BOOTSET为ICE模式，软件就不能改变BOOTSET为其它模式。但是其它模式就没有这个限制

外设复位控制寄存器 0 (SYS_IPRST0)

寄存器	偏移地址	R/W	描述	复位值
SYS_IPRST0	GCR_BA+0x08	R/W	外设复位控制寄存器 0	0x0000_0009

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						CHIPRST	CPURST

位	描述	
[31:2]	Reserved	保留.
[1]	CHIPRST	**芯片复位** 该位置1将复位整个芯片包括内核和外设，2时钟周期后该位将自动被清0。 芯片复位和上电复位一样，复位整个芯片控制器，从flash中重新加载芯片配置。 0 = 芯片正常工作 1 = 触发芯片复位
[0]	CPURST	**处理器内核复位** 该位置1只复位处理器内核，2时钟周期后该位将自动被清0。 0 = 内核正常工作 1 = 触发内核复位

低电压检测控制寄存器(SYS_LVDCTL)

寄存器	偏移地址	R/W	描述	复位值
SYS_LVDCTL	GCR_BA+0x0C	R/W	低电压检测控制寄存器	0x0000_0009

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			PORENB	LVREN	LDVEN	LDVSEL	LVD_FALG

位	描述	
[31:5]	Reserved	保留
[4]	PORENB	上电复位使能 0 = 使能 (默认) 1 = 禁用
[3]	LVREN	低电压使能 0 =禁用 1 =使能 (默认) 注:电压阀值是2.4V
[2]	LVDEN	低电压侦测使能 0 =禁用 1 = 使能
[1]	LVDSEL	低电压侦测阀值选择 0 = 2.6V 1 = 2.8V
[0]	LVDIF	低电压侦测标志 0 = 电压过低 1 = 电压正常 注: 只有当LVDEN 使能，该位才有意义

唤醒控制和状态 寄存器(SYS_WAKEUP)

寄存器	偏移地址	R/W	描述	复位值
SYS_WAKEUP	GCR_BA+0x10	R/W	唤醒控制和状态 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		USBHWF	USBDWF	UART2WF	UART1WF	UART0WF	TMR3WF
23	22	21	20	19	18	17	16
TMR2WF	TMR1WF	TMR0WF	WDTWF	RTCWF	GPIOWF	I2C1WF	I2C0WF
15	14	13	12	11	10	9	8
Reserved		USBHWE	USBDWE	UART2WE	UART1WE	UART0WE	TMR3WE
7	6	5	4	3	2	1	0
TMR2WE	TMR1WE	TMR0WE	WDTWE	RTCWE	GPIOWE	I2C1WE	I2C0WE

位	描述	
[31:30]	**Reserved**	保留
[29]	**USBHWF**	**USB主机唤醒标志位** 0 = 未发生USB 主机唤醒. 1 =发生USB 主机唤醒. **注:** 写1清除该位
[28]	**USBDWF**	**USB 设备唤醒标志位** 0 = 未发生USB 设备唤醒. 1 =发生USB 设备唤醒. **注:** 写1清除该位
[27]	**UART2WF**	**UART2 唤醒标志位** 0 = 未发生UART2唤醒. 1 =发生UART2唤醒. **注:** 写1清除该位
[26]	**UART1WF**	**UART1 唤醒标志位** 0 = 未发生UART1唤醒. 1 =发生UART1唤醒. **注:** 写1清除该位
[25]	**UART0WF**	**UART0 唤醒标志位** 0 = 未发生UART0唤醒. 1 =发生UART0唤醒. **注:** 写1清除该位

[24]	**TMR3WF**	**Timer3 唤醒标志位** 0 = 未发生Timer3唤醒. 1 =发生Timer3唤醒. **注:** 写1清除该位
[23]	**TMR2WF**	**Timer2 唤醒标志位** 0 = 未发生Timer2唤醒. 1 =发生Timer2唤醒. **注:** 写1清除该位
[22]	**TMR1WF**	**Timer1 唤醒标志位** 0 = 未发生Timer1唤醒. 1 =发生Timer1唤醒. **注:** 写1清除该位
[21]	**TMR0WF**	**Timer0 唤醒标志位** 0 = 未发生Timer0唤醒. 1 =发生Timer0唤醒. **注:** 写1清除该位
[20]	**WDTWF**	**WDT 唤醒标志位** 0 = 未发生WDT唤醒. 1 =发生WDT唤醒. **注:** 写1清除该位
[19]	**RTCWF**	**RTC 唤醒标志位** 0 = 未发生RTC唤醒. 1 =发生RTC唤醒. **注:** 写1清除该位
[18]	**GPIOWF**	**GPIO 唤醒标志位** 0 = 未发生GPIO唤醒. 1 =发生GPIO唤醒. **注:** 写1清除该位
[17]	**I2C1WF**	**I2C1 唤醒标志位** 0 = 未发生I2C1唤醒. 1 =发生I2C1唤醒. **注:** 写1清除该位
[16]	**I2C0WF**	**I2C0 唤醒标志位** 0 = 未发生I2C0唤醒. 1 =发生I2C0唤醒. **注:** 写1清除该位
[15:14]	**Reserved**	保留
[13]	**USBHWE**	**USB主机唤醒使能位** 0 = USB主机唤醒功能禁用 1 = USB 主机唤醒功能使能

[12]	**USBDWE**	**USB 设备唤醒使能位** 0 = USB 设备 唤醒功能禁用 1 = USB 设备 唤醒功能使能
[11]	**UART2WE**	**UART2 唤醒使能位** 0 = UART2 唤醒功能禁用 1 = UART2 唤醒功能使能
[10]	**UART1WE**	**UART1 唤醒使能位** 0 = UART1唤醒功能禁用 1 = UART1 唤醒功能使能
[9]	**UART0WE**	**UART0 唤醒使能位** 0 = UART0唤醒功能禁用 1 = UART0 唤醒功能使能
[8]	**TMR3WE**	**Timer3 唤醒使能位** 0 = Timer3唤醒功能禁用 1 = Timer3 唤醒功能使能
[7]	**TMR2WE**	**Timer2 唤醒使能位** 0 = Timer2唤醒功能禁用 1 = Timer2 唤醒功能使能
[6]	**TMR1WE**	**Timer1 唤醒使能位** 0 = Timer1唤醒功能禁用 1 = Timer1 唤醒功能使能
[5]	**TMR0WE**	**Timer0 唤醒使能位** 0 = Timer0唤醒功能禁用 1 = Timer0 唤醒功能使能
[4]	**WDTWE**	**WDT 唤醒使能位** 0 = WDT唤醒功能禁用 1 = WDT 唤醒功能使能
[3]	**RTCWE**	**RTC 唤醒使能位** 0 = RTC唤醒功能禁用 1 = RTC 唤醒功能使能
[2]	**GPIOWE**	**GPIO 唤醒使能位** 0 = GPIO唤醒功能禁用 1 = GPIO 唤醒功能使能
[1]	**I2C1WE**	**I2C1 唤醒使能位** 0 = I2C1唤醒功能禁用 1 = I2C1 唤醒功能使能
[0]	**I2C0WE**	**I2C0 唤醒使能位** 0 = I2C0唤醒功能禁用 1 = I2C0 唤醒功能使能

外设复位控制寄存器1 (SYS_IPRST1)

置以下相应位为1会产生异步复位信号给相应的控制器。用户需要将这些位置0才能将相应的控制器从复位状态释放。

寄存器	偏移地址	R/W	描述	复位值
SYS_IPRST1	GCR_BA+0x14	R/W	外设复位控制寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
SPI1RST	SPI0RST	Reserved	ADCRST	GPIORST	Reserved	SRAMRST	SDHRST
23	22	21	20	19	18	17	16
Reserved					USBHRST	I2SRST	Reserved
15	14	13	12	11	10	9	8
Reserved		WDTPRST	TMR3RST	USBDRST	SPIMRST	I2C1RST	I2C0RST
7	6	5	4	3	2	1	0
PWMRST	UART2RST	TMR2RST	WDTFRST	TMR1RST	TMR0RST	UART1RST	UART0RST

位	描述	
[31]	SPI1RST	**SPI1 控制器复位** 0 = SPI1 控制器正常工作 1 = SPI1 控制器复位
[30]	SPI0RST	**SPI0 控制器复位** 0 = SPI1 控制器正常工作 1 = SPI1 控制器复位
[29]	Reserved	保留
[28]	ADCRST	**ADC控制器复位** 0 = ADC控制器正常工作 1 = ADC控制器复位
[27]	GPIORST	**GPIO控制器复位** 0 = GPIO控制器正常工作 1 = GPIO控制器复位
[26]	Reserved	保留
[25]	SRAMRST	**SRAM控制器复位** 0 = SRAM控制器正常工作 1 = SRAM控制器复位

[24]	**SDHRST**	**SDH控制器复位** 0 = SDH控制器正常工作 1 = SDH控制器复位
[23:19]	**Reserved**	保留
[18]	**USBHRST**	**USB主机控制器复位** 0 = USB主机控制器正常工作 1 = USB主机控制器复位
[17]	**I2SRST**	**I^{2}S控制器复位** 0 = I^{2}S控制器正常工作 1 = I^{2}S控制器复位
[16:14]	**Reserved**	保留
[13]	**WDTPRST**	**WDT控制器复位** 0 = WDT控制器正常工作 1 = WDT控制器复位
[12]	**TMR3RST**	**Timer3控制器复位** 0 = Timer3控制器正常工作 1 = Timer3控制器复位
[11]	**USBDRST**	**USB设备控制器复位** 0 = USB设备控制器正常工作 1 = USB设备控制器复位
[10]	**SPIMRST**	**SPIM控制器复位** 0 = SPIM控制器正常工作 1 = SPIM控制器复位
[9]	**I2C1RST**	**I2C1控制器复位** 0 = I2C1控制器正常工作 1 = I2C1控制器复位
[8]	**I2C0RST**	**I2C0控制器复位** 0 = I2C0控制器正常工作 1 = I2C0控制器复位
[7]	**PWMRST**	**PWM控制器复位** 0 = PWM控制器正常工作 1 = PWM控制器复位
[6]	**UART2RST**	**UART2控制器复位** 0 = UART2控制器正常工作 1 = UART2控制器复位
[5]	**TMR2RST**	**Timer2控制器复位** 0 = Timer2控制器正常工作 1 = Timer2控制器复位

[4]	**WDTFRST**	**WDT硬件控制器复位** 0 = WDT控制器正常工作 1 = WDT控制器复位
[3]	**TMR1RST**	**Timer1控制器复位** 0 = Timer1控制器正常工作 1 = Timer1控制器复位
[2]	**TMR0RST**	**Timer0控制器复位** 0 = Timer0控制器正常工作 1 = Timer0控制器复位
[1]	**UART1RST**	**UART1控制器复位** 0 = UART1控制器正常工作 1 = UART1控制器复位
[0]	**UART0RST**	**UART0控制器复位** 0 = UART0控制器正常工作 1 = UART0控制器复位

不可屏蔽中断控制寄存器(SYS_NMICTL)

寄存器	偏移地址	R/W	描述	复位值
SYS_NMICTL	GCR_BA+0x18	R/W	不可屏蔽中断控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
LVDIF	EINT3IF	EINT2IF	EINT1IF	EINT0IF	PORIF	WDTIF	RTCIF
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
LVDIEN	EINT3IEN	EINT2IEN	EINT1IEN	EINT0IEN	PORIEN	WDTIEN	RTCIEN

位	描述	
[31:24]	Reserved	保留
[23]	LVDIF	低电压侦测（LVD）中断标志位（只读） 0 =未发生LVD中断 1 =发生LVD中断
[22]	EINT3IF	外部GPIO组3中断标志位（只读） 0 =未发生GPIO组3中断 1 =发生GPIO组3中断
[21]	EINT2IF	外部GPIO组2中断标志位（只读） 0 =未发生GPIO组2中断 1 =发生GPIO组2中断
[20]	EINT1IF	外部GPIO组1中断标志位（只读） 0 =未发生GPIO组1中断 1 =发生GPIO组1中断
[19]	EINT0IF	外部GPIO组0中断标志位（只读） 0 =未发生GPIO组0中断 1 =发生GPIO组0中断
[18]	PORIF	上电复位（POR）中断标志位（只读） 0 =未发生POR中断 1 =发生POR中断
[17]	WDTIF	看门狗定时器（WDT）中断标志位（只读） 0 =未发生WDT中断 1 =发生WDT中断

[16]	**RTCIF**	**RTC中断标志位（只读）** 0 =未发生RTC中断 1 =发生RTC中断
[15:8]	**Reserved**	保留.
[7]	**LVDIEN**	**低电压侦测不可屏蔽中断(NMI)源使能位** 0 = 禁用 1 = 使能
[6]	**EINT3IEN**	**外部GPIO组3不可屏蔽中断(NMI)源使能位** 0 = 禁用 1 = 使能
[5]	**EINT2IEN**	**外部GPIO组2不可屏蔽中断(NMI)源使能位** 0 = 禁用 1 = 使能
[4]	**EINT1IEN**	**外部GPIO组1不可屏蔽中断(NMI)源使能位** 0 = 禁用 1 = 使能
[3]	**EINT0IEN**	**外部GPIO组0不可屏蔽中断(NMI)源使能位** 0 = 禁用 1 = 使能
[2]	**PORIEN**	**上电中断不可屏蔽中断(NMI)源使能位** 0 = 禁用 1 = 使能
[1]	**WDTIEN**	**WDT中断不可屏蔽中断(NMI)源使能位** 0 = 禁用 1 = 使能
[0]	**RTCIEN**	**RTC 中断不可屏蔽中断(NMI)源使能位** 0 = 禁用 1 = 使能

复位状态寄存器(SYS_RSTSTS)

寄存器	偏移地址	R/W	描述	复位值
SYS_RSTSTS	GCR_BA+0x1C	R/W	复位状态寄存器	0x0000_0000

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

位	描述	
[31:16]	**Reserved**	保留
[15:5]	**Reserved**	保留
[4]	**PORF**	**上电复位（POR）标志** 0= 没有影响 1= 上电复位(POR)发出复位信号使系统复位 **注：** 向该位写1清零。
[3]	**CPURF**	**软件复位标志** 0 =没有影响 1 = CPURST或是CHIPREST软件触发使系统复位 **注：** 向该位写1清零。
[2]	**WDTRF**	**看门狗复位标志** 0 =没有影响 1= 看门狗或窗口看门狗发出复位信号来复位系统 **注:** 向该位写1清零。
[1]	**LVRF**	**低电压（LVR）复位标志** 0 =没有影响 1 =LVR 模块发出复位信号使系统复位 **注：** 向该位写1清零。
[0]	**PINRF**	**nRESET引脚复位标志** 0= nRESET复位引脚无复位 1= nRESET复位引脚发出复位信号使系统复位 **注：** 向该位写1清零。

AHB 总线控制寄存器 (SYS_AHBCTL)

寄存器	偏移地址	R/W	描述	复位值
SYS_AHBCTL	GCR_BA+0x20	R/W	AHB 总线控制寄存器	0x0000_0000

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

位	描述	
[31:6]	Reserved	保留
[5]	PRISTS	**在使能CPUHPRI模式下中断运行状态激活位** 如果该位置1，表明CPU拥有AHB总线最高优先级。在CPUHPRI使能和外部IRQ激活时该位置1. 写1清除该位。因此在CPUHPRI使能时如果从IRQ退出时，PRISTS必须被清除了，否者CPU总是拥有AHB总线最高优先级。 0 =没有影响 1 = 激活CPU拥有AHB总线最高优先级
[4]	CPUHPRI	**提升CPU在IRQ期间的优先级使能控制位** 在实时系统中，可以减少中断延迟。该位 置1，CPU将会拥有AHB最高优先级 0 = 没有影响 1 = 使能
[3:1]	Reserved	保留
[0]	PRISEL	**AHB 总线仲裁模式控制位** 0 = 固定优先级模式 1 = 循环优先级模式 固定优先级模式的优先级顺序：I^2S > SDH > USBH > USBD > SPIM > M4(S) > M4(D) > M4(I)

GPIOA低字节复用功能控制寄存器(SYS_GPA_MFPL)

请参考**錯誤! 找不到參照來源。** GPIO 多功能引脚概要

寄存器	偏移地址	R/W	描述	复位值
SYS_GPA_MFPL	GCR_BA+0x30	R/W	GPIOA低字节复用功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	PA7MFP			Reserved	PA6MFP		
23	22	21	20	19	18	17	16
Reserved	PA5MFP			Reserved	PA4MFP		
15	14	13	12	11	10	9	8
Reserved	PA3MFP			Reserved	PA2MFP		
7	6	5	4	3	2	1	0
Reserved	PA1MFP			Reserved	PA0MFP		

位	描述	
[30:28]	PA7MFP	PA.7多功能引脚选择
[26:24]	PA6MFP	PA.6多功能引脚选择
[22:20]	PA5MFP	PA.5多功能引脚选择
[18:16]	PA4MFP	PA.4多功能引脚选择
[14:12]	PA3MFP	PA.3多功能引脚选择
[10:8]	PA2MFP	PA.2多功能引脚选择
[6:4]	PA1MFP	PA.1多功能引脚选择
[2:0]	PA0MFP	PA.0多功能引脚选择

GPIOA 高字节复用功能控制寄存器(SYS_GPA_MFPH)

请参考**錯誤! 找不到參照來源。** GPIO 多功能引脚概要

寄存器	偏移地址	R/W	描述	复位值
SYS_GPA_MFPH	GCR_BA+0x34	R/W	GPIOA 高字节复用功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	PA15MFP			Reserved	PA14MFP		
23	22	21	20	19	18	17	16
Reserved	PA13MFP			Reserved	PA12MFP		
15	14	13	12	11	10	9	8
Reserved	PA11MFP			Reserved	PA10MFP		
7	6	5	4	3	2	1	0
Reserved	PA9MFP			Reserved	PA8MFP		

位	描述	
[30:28]	PA15MFP	PA.15多功能引脚选择
[26:24]	PA14MFP	PA.14多功能引脚选择
[22:20]	PA13MFP	PA.13多功能引脚选择
[18:16]	PA12MFP	PA.12多功能引脚选择
[14:12]	PA11MFP	PA.11多功能引脚选择
[10:8]	PA10MFP	PA.10多功能引脚选择
[6:4]	PA9MFP	PA.9多功能引脚选择
[2:0]	PA8MFP	PA.8多功能引脚选择

GPIOB 低字节复用功能控制寄存器(SYS_GPB_MFPL)

请参考**錯誤! 找不到參照來源。** GPIO 多功能引脚概要

寄存器	偏移地址	R/W	描述	复位值
SYS_GPB_MFPL	GCR_BA+0x38	R/W	GPIOB 低字节复用功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	PB7MFP			Reserved	PB6MFP		
23	22	21	20	19	18	17	16
Reserved	PB5MFP			Reserved	PB4MFP		
15	14	13	12	11	10	9	8
Reserved	PB3MFP			Reserved	PB2MFP		
7	6	5	4	3	2	1	0
Reserved	PB1MFP			Reserved	PB0MFP		

位	描述	
[30:28]	PB7MFP	PB.7多功能引脚选择
[26:24]	PB6MFP	PB.6多功能引脚选择
[22:20]	PB5MFP	PB.5多功能引脚选择
[18:16]	PB4MFP	PB.4多功能引脚选择
[14:12]	PB3MFP	PB.3多功能引脚选择
[10:8]	PB2MFP	PB.2多功能引脚选择
[6:4]	PB1MFP	PB.1多功能引脚选择
[2:0]	PB0MFP	PB.0多功能引脚选择

GPIOB 高字节复用功能控制寄存器(SYS_GPB_MFPH)

请参考**錯誤! 找不到參照來源。** GPIO 多功能引脚概要

寄存器	偏移地址	R/W	描述	复位值
SYS_GPB_MFPH	GCR_BA+0x3C	R/W	GPIOB 高字节复用功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	PB15MFP			Reserved	PB14MFP		
23	22	21	20	19	18	17	16
Reserved	PB13MFP			Reserved	PB12MFP		
15	14	13	12	11	10	9	8
Reserved	PB11MFP			Reserved	PB10MFP		
7	6	5	4	3	2	1	0
Reserved	PB9MFP			Reserved	PB8MFP		

位	描述	
[30:28]	PB15MFP	PB.15多功能引脚选择
[26:24]	PB14MFP	PB.14多功能引脚选择
[22:20]	PB13MFP	PB.13多功能引脚选择
[18:16]	PB12MFP	PB.12多功能引脚选择
[14:12]	PB11MFP	PB.11多功能引脚选择
[10:8]	PB10MFP	PB.10多功能引脚选择
[6:4]	PB9MFP	PB.9多功能引脚选择
[2:0]	PB8MFP	PB.8多功能引脚选择

GPIOC 低字节复用功能控制寄存器(SYS_GPC_MFPL)

请参考**錯誤! 找不到參照來源。** GPIO 多功能引脚概要

寄存器	偏移地址	R/W	描述	复位值
SYS_GPC_MFPL	GCR_BA+0x40	R/W	GPIOC 低字节复用功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	PC7MFP			Reserved	PC6MFP		
23	22	21	20	19	18	17	16
Reserved	PC5MFP			Reserved	PC4MFP		
15	14	13	12	11	10	9	8
Reserved	PC3MFP			Reserved	PC2MFP		
7	6	5	4	3	2	1	0
Reserved	PC1MFP			Reserved	PC0MFP		

位	描述	
[30:28]	PC7MFP	PC.7多功能引脚选择
[26:24]	PC6MFP	PC.6多功能引脚选择
[22:20]	PC5MFP	PC.5多功能引脚选择
[18:16]	PC4MFP	PC.4多功能引脚选择
[14:12]	PC3MFP	PC.3多功能引脚选择
[10:8]	PC2MFP	PC.2多功能引脚选择
[6:4]	PC1MFP	PC.1多功能引脚选择
[2:0]	PC0MFP	PC.0多功能引脚选择

GPIOC高字节复用功能控制寄存器(SYS_GPC_MFPH)

请参考**錯誤! 找不到參照來源。** GPIO 多功能引脚概要

寄存器	偏移地址	R/W	描述	复位值
SYS_GPC_MFPH	GCR_BA+0x44	R/W	GPIOC 高字节复用功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	Reserved			Reserved	PC14MFP		
23	22	21	20	19	18	17	16
Reserved	PC13MFP			Reserved	PC12MFP		
15	14	13	12	11	10	9	8
Reserved	PC11MFP			Reserved	PC10MFP		
7	6	5	4	3	2	1	0
Reserved	PC9MFP			Reserved	PC8MFP		

位	描述	
[26:24]	PC14MFP	PC.14多功能引脚选择
[22:20]	PC13MFP	PC.13多功能引脚选择
[18:16]	PC12MFP	PC.12多功能引脚选择
[14:12]	PC11MFP	PC.11多功能引脚选择
[10:8]	PC10MFP	PC.10多功能引脚选择
[6:4]	PC9MFP	PC.9多功能引脚选择
[2:0]	PC8MFP	PC.8多功能引脚选择

GPIOD 低字节复用功能控制寄存器(SYS_GPD_MFPL)

请参考**錯誤! 找不到參照來源。** GPIO 多功能引脚概要

寄存器	偏移地址	R/W	描述	复位值
SYS_GPD_MFPL	GCR_BA+0x48	R/W	GPIOD 低字节复用功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				Reserved	PD4MFP		
15	14	13	12	11	10	9	8
Reserved	PD3MFP			Reserved	PD2MFP		
7	6	5	4	3	2	1	0
Reserved	PD1MFP			Reserved	PD0MFP		

位	描述	
[31:7]	Reserved	保留
[18:16]	PD4MFP	PD.4多功能引脚选择
[14:12]	PD3MFP	PD.3多功能引脚选择
[10:8]	PD2MFP	PD.2多功能引脚选择
[6:4]	PD1MFP	PD.1多功能引脚选择
[2:0]	PD0MFP	PD.0多功能引脚选择

加载 VECMAP 地址参数控制寄存器(SYS_LVMPADDR)

寄存器	偏移地址	R/W	描述	复位值
SYS_LVMPADDR	GCR_BA+0x50	R/W	加载 VECMAP 地址参数控制寄存器	0x1FFF_0000

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

位	描述	
[31:0]	ADDR	**加载VECMAP地址寄存器** 在VECMAP功能中该寄存器装载的起始地址被映射到地址0x0000_0000。只有在CPU复位或是RLDVMP被置1，该值才会被加载到SYS_RVMPADDR寄存器中 **注:**复位该寄存器只能通过CHIPRST和HW复位

加载 VECMAP 长度参数控制寄存器(SYS_LVMPLEN)

寄存器	偏移地址	R/W	描述	复位值
SYS_LVMPLEN	GCR_BA+0x54	R/W	加载 VECMAP 长度参数控制寄存器	0x0000_0008

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
LEN							

位	描述	
[31:8]	**Reserved**	保留
[7:0]	**LEN**	**LD_VECMAP 长度** 在VECMAP功能中该寄存器提供存储器长度用来映射到地址0x0000_0000。只有在CPU复位或是RLDVMP被置1，该值才会被加载到SYS_ RVMPLEN寄存器中 映射存储器长度 = VMP_LEN * 1K (字节). **注1:** 最大映射长度 128K 字节 **注2:**复位该寄存器只能通过CHIPRST和HW复位

实际 VECMAP地址参数寄存器(SYS_RVMPADDR)

寄存器	偏移地址	R/W	描述	复位值
SYS_RVMPADDR	GCR_BA+0x58	R	实际 VECMAP地址参数寄存器	0x1FFF_0000

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

位	描述	
[31:0]	ADDR	**实际 VECMAP地址参数寄存器** 在VECMAP功能中该值是实际起始地址参数用于重映射到地址0x0000_0000。（默认重映射地址是IBR_ROM起始地址） **注：**只有在CPU复位或是RLDVMP被置1，该值才会被加载

实际 VECMAP长度参数控制寄存器(SYS_RVMPLEN)

寄存器	偏移地址	R/W	描述	复位值
SYS_RVMPLEN	GCR_BA+0x5C	R/W	实际 VECMAP长度参数控制寄存器	0x0800_0000

31	30	29	28	27	26	25	24
LEN							
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							RLDVMP

位	描述	
[31:24]	**LEN**	**实际 VECMAP 长度** 在VECMAP功能中该值是实际存储器长度用来映射到地址0x0000_0000。 映射存储器长度 = VMP_LEN * 1K (字节). **注1:** 只读 **注2:** 只有在CPU复位或是RLDVMP被置1，该值才会被加载
[23:1]	**Reserved**	保留
[0]	**RLDVMP**	**加载VECMAP参数** 0 = 没有影响 1 = 加载 VECMAP 地址 和长度 **注:** 该位会自动被清0

内嵌 SPI Flash Pad 上拉 使能控制寄存器(SYS_EPADPUEN)

寄存器	偏移地址	R/W	描述	复位值
SYS_EPADPUEN	GCR_BA+0x6C	R/W	内嵌 SPI Flash Pad 上拉 使能控制寄存器	0x0000_001F

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			EPADPUEN				

位	描述	
[31:5]	Reserved	保留.
[4:0]	EPADPUEN	**内嵌 SPI Flash Pad 上拉使能控制位** 00000 =禁用所有内嵌pads上拉 10010 =只使能SPI Flash MISO pads上拉 11111 =使能所有内嵌pads上拉 Other = 保留 **注:**在掉电模式下，用户应该设置该位为0x12

GPIOA 驱动能力控制寄存器(SYS_GPADS)

寄存器	偏移地址	R/W	描述	复位值
SYS_GPADS	GCR_BA+0x70	R/W	GPIOA 驱动能力控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	PA7DS			Reserved	PA6DS		
23	22	21	20	19	18	17	16
Reserved	PA5DS			Reserved	PA4DS		
15	14	13	12	11	10	9	8
Reserved	PA3DS			Reserved	PA2DS		
7	6	5	4	3	2	1	0
Reserved	PA1DS			Reserved	PA0DS		

位	描述	
[31]	**Reserved**	保留
[30:28]	**PA7DS**	**PA.7驱动能力控制位** PA.7模拟/数字混合引脚的驱动能力 000 = 2.0 mA (默认). 001 = 6.5 mA. 010 = 8.7 mA. 011 = 13.0 mA. 100 = 15.2 mA. 101 = 19.5 mA. 110 = 21.7 mA. 111 = 26.1 mA.
[27]	**Reserved**	Reserved.
[26:24]	**PA6DS**	**PA.6驱动能力控制位** PA.6模拟/数字混合引脚的驱动能力 000 = 2.0 mA (默认). 001 = 6.5 mA. 010 = 8.7 mA. 011 = 13.0 mA. 100 = 15.2 mA. 101 = 19.5 mA. 110 = 21.7 mA. 111 = 26.1 mA.
[23]	**Reserved**	保留

[22:20]	**PA5DS**	**PA.5驱动能力控制位** PA.5模拟/数字混合引脚的驱动能力 000 = 2.0 mA (默认). 001 = 6.5 mA. 010 = 8.7 mA. 011 = 13.0 mA. 100 = 15.2 mA. 101 = 19.5 mA. 110 = 21.7 mA. 111 = 26.1 mA.
[19]	**Reserved**	保留
[18:16]	**PA4DS**	**PA.4驱动能力控制位** PA.4模拟/数字混合引脚的驱动能力 000 = 2.0 mA (默认). 001 = 6.5 mA. 010 = 8.7 mA. 011 = 13.0 mA. 100 = 15.2 mA. 101 = 19.5 mA. 110 = 21.7 mA. 111 = 26.1 mA.
[15]	**Reserved**	保留
[14:12]	**PA3DS**	**PA.3驱动能力控制位** PA.3模拟/数字混合引脚的驱动能力 000 = 2.0 mA (默认). 001 = 6.5 mA. 010 = 8.7 mA. 011 = 13.0 mA. 100 = 15.2 mA. 101 = 19.5 mA. 110 = 21.7 mA. 111 = 26.1 mA.
[11]	**Reserved**	保留.
[10:8]	**PA2DS**	**PA.2驱动能力控制位** PA.2模拟/数字混合引脚的驱动能力 000 = 2.0 mA (默认). 001 = 6.5 mA. 010 = 8.7 mA. 011 = 13.0 mA. 100 = 15.2 mA. 101 = 19.5 mA. 110 = 21.7 mA. 111 = 26.1 mA.

[7]	**Reserved**	保留
[6:4]	**PA1DS**	**PA.1驱动能力控制位** PA.1模拟/数字混合引脚的驱动能力 000 = 2.0 mA (默认). 001 = 6.5 mA. 010 = 8.7 mA. 011 = 13.0 mA. 100 = 15.2 mA. 101 = 19.5 mA. 110 = 21.7 mA. 111 = 26.1 mA.
[3]	**Reserved**	保留
[2:0]	**PA0DS**	**PA.0驱动能力控制位** PA.0模拟/数字混合引脚的驱动能力 000 = 2.0 mA (默认). 001 = 6.5 mA. 010 = 8.7 mA. 011 = 13.0 mA. 100 = 15.2 mA. 101 = 19.5 mA. 110 = 21.7 mA. 111 = 26.1 mA.

GPIOA 输入缓冲使能控制寄存器(SYS_GPAIBE)

寄存器	偏移地址	R/W	描述	复位值
SYS_GPAIBE	GCR_BA+0x74	R/W	GPIOA 输入缓冲使能控制寄存器	0xFFFF_0000

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

位	描述	
[30:24]	**SMTENx**	**SMTENx (x = 8,9,…,15)施密特触发输入缓冲使能控制位** 0 =禁用 1 =使能 (默认) **注:** 如果施密特触发和CMOS输入缓冲都设置为0，PAD的输入信号始终为0。
[23:16]	**DINONx**	**DINONx (x = 0,1,…,7) 输入缓冲使能控制位** 0 =禁用 1 =使能 (默认) **注1:** 如果设为0，PAD的输入信号始终为0。 **注2:**如果PA.0~PA.7作为模拟输入时，不要忘记禁用输入缓冲
[15:8]	**CMOSENx**	**CMOSENx (x = 8,9,…,15)CMOS 输入缓冲使能控制位** 0 =禁用(默认) 1 =使能 **注:** 如果施密特触发和CMOS输入缓冲都设置为0，PAD的输入信号始终为0。
[7:0]	**IBSELx**	**IBSELx (x = 0,1,…,7)选着施密特触发还是CMOS输入缓冲** 0 = CMOS 输入缓冲 (默认). 1 =施密特输入缓冲.

注: 施密特触发和CMOS输入缓冲不能同时被使能

GPIOB输入缓冲使能控制寄存器(SYS_GPBIBE)

寄存器	偏移地址	R/W	描述	复位值
SYS_GPBIBE	GCR_BA+0x78	R/W	GPIOB输入缓冲使能控制寄存器	0xFFFF_0000

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

位	描述	
[30:16]	SMTENx	**SMTENx (x = 0,1,…,15) 施密特触发输入缓冲使能控制位** 0 =禁用 1 =使能 (默认) **注:** 如果施密特触发和CMOS输入缓冲都设置为0，PAD的输入信号始终为0。
[15:0]	CMOSENx	**CMOSENx (x = 0,1,…,15)CMOS输入缓冲使能控制位** 0 =禁用(默认) 1 =使能 **注:** 如果施密特触发和CMOS输入缓冲都设置为0，PAD的输入信号始终为0。

注: 施密特触发和CMOS输入缓冲不能同时被使能

GPIOC输入缓冲使能控制寄存器(SYS_GPCIBE)

寄存器	偏移地址	R/W	描述	复位值
SYS_GPCIBE	GCR_BA+0x7C	R/W	GPIOC输入缓冲使能控制寄存器	0xFFFF_0000

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

位	描述	
[31]	**Reserved**	保留
[30:16]	**SMTENx**	**SMTENx (x = 0,1,…,14) 施密特触发输入缓冲使能控制位** 0 =禁用 1 =使能 (默认) **注:** 如果施密特触发和CMOS输入缓冲都设置为0，PAD的输入信号始终为0。
[15]	**Reserved**	保留
[14:0]	**CMOSENx**	**CMOSENx (x = 0,1,…,14) CMOS输入缓冲使能控制位** 0 =禁用(默认) 1 =使能 **注:** 如果施密特触发和CMOS输入缓冲都设置为0，PAD的输入信号始终为0。

注: 施密特触发和CMOS输入缓冲不能同时被使能

GPIOD输入缓冲使能控制寄存器(SYS_GPDIBE)

寄存器	偏移地址	R/W	描述	复位值
SYS_GPDIBE	GCR_BA+0x80	R/W	GPIOD输入缓冲使能控制寄存器	0xFFFF_0000

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

位	描述	
[31:21]	**Reserved**	保留
[20:18]	**DINONx**	**DINONx (x = 2,3,4) 输入缓冲使能控制位** 0 =禁用 1 =使能 (默认) **注1:** 如果设为0，PAD的输入信号始终为0。 **注2:**如果PD.2~PD.4作为模拟输入时，不要忘记禁用输入缓冲
[17:16]	**SMTENx**	**SMTENx (x = 0,1) 施密特触发输入缓冲使能控制位** 0 =禁用 1 =使能 (默认) **注:** 如果施密特触发和CMOS输入缓冲都设置为0，PAD的输入信号始终为0。
[15:5]	**Reserved**	保留
[4:2]	**IBSELx**	**IBSELx (x = 2,3,4) 选着施密特触发还是CMOS输入缓冲** 0 = CMOS 输入缓冲 (默认). 1 =施密特输入缓冲.
[1:0]	**CMOSENx**	**CMOSENx (x = 0,1)CMOS 输入缓冲使能控制位** 0 =禁用(默认) 1 =使能 **注:** 如果施密特触发和CMOS输入缓冲都设置为0，PAD的输入信号始终为0。

注: 施密特触发和CMOS输入缓冲不能同时被使能

GPIOD驱动能力控制寄存器(SYS_GPDDS)

寄存器	偏移地址	R/W	描述	复位值
SYS_GPDDS	GCR_BA+0x84	R/W	GPIOD驱动能力控制寄存器	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					PD4DS		
7	6	5	4	3	2	1	0
Reserved	PD3DS			Reserved	PD2DS		

位	描述	
[31:11]	**Reserved**	保留
[10:8]	**PD4DS**	**PD.4驱动能力控制位** PD.4模拟/数字混合引脚的驱动能力 000 = 2.0 mA (默认). 001 = 6.5 mA. 010 = 8.7 mA. 011 = 13.0 mA. 100 = 15.2 mA. 101 = 19.5 mA. 110 = 21.7 mA. 111 = 26.1 mA.
[7]	**Reserved**	保留
[6:4]	**PD3DS**	**PD.3驱动能力控制位** PD.3模拟/数字混合引脚的驱动能力 000 = 2.0 mA (默认). 001 = 6.5 mA. 010 = 8.7 mA. 011 = 13.0 mA. 100 = 15.2 mA. 101 = 19.5 mA. 110 = 21.7 mA. 111 = 26.1 mA.
[3]	**Reserved**	保留

[2:0]	**PD2DS**	**PD.2驱动能力控制位** PD.2模拟/数字混合引脚的驱动能力 000 = 2.0 mA (默认). 001 = 6.5 mA. 010 = 8.7 mA. 011 = 13.0 mA. 100 = 15.2 mA. 101 = 19.5 mA. 110 = 21.7 mA. 111 = 26.1 mA.

外部 nRESET 引脚 抖动计数 控制寄存器(SYS_RSTDBCNT)

寄存器	偏移地址	R/W	描述	复位值
SYS_RSTDBCNT	GCR_BA+0x100	R/W	外部 nRESET 引脚 抖动计数 控制寄存器	0x0000_04B0

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
RSTDBCNT							
7	6	5	4	3	2	1	0
RSTDBCNT							

位	描述	
[31:16]	Reserved	保留
[15:0]	RSTDBCNT	**外部 nRESET 引脚 抖动计数** 该16位计数器防抖动时间最多可以到5.46ms（0xFFFF）@XIN=12 MHz. **注:** 默认时间为0.1ms(0x04B0) @XIN=12 MHz.

外部 nRESET 引脚 抖动 控制寄存器(SYS_RSTDBEN)

寄存器	偏移地址	R/W	描述	复位值
SYS_RSTDBEN	GCR_BA+0x104	R/W	外部 nRESET 引脚 抖动 控制寄存器	0x0000_0000

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

位	描述	
[31:1]	Reserved	保留
[0]	RSTDBEN	外部 nRESET 引脚 抖动 控制使能位 0 = 禁用防抖动功能 1 =使能防抖动功能

6.2.10 系统定时器 (Systick)

Cortex-M4 内置了一个系统定时器：SysTick。SysTick 提供一种简单的24位写清零、递减、自装载同时具有可灵活控制机制的计数器。该计数器可用作实时操作系统(RTOS) 的滴答定时器或一个简单的计数器。

当系统定时器使能后，将从 SysTick 的当前值寄存器 (SYST_ CVR) 的值向下计数到0，并在下一个时钟周期，重新加载 SysTick 重新加载值寄存器 (SYST_ RVR) 的值，然后再随时钟递减。当计数器减到0时，标志位COUNTFLAG置位，读 COUNTFLAG 位使其清零。

复位后，SYST_ CVR的值是未知的。使能前，软件应该向寄存器写入值清零。这样确保定时器使能时以SYST_ RVR的值开始计数，而非任意值。

若SYST_ RVR的值为0，在重新加载后，定时器将保持当前值0。这个功能可以在计数器使能后用来禁用的独立功能。

详情请参考 “ARM® Cortex™-M4 Technical Reference Manual” 与 “ARM® v6-M Architecture Reference Manual”

6.2.10.1 系统定时器控制寄存器映射

R:只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移地址	R/W	描述	复位值
SYST 基地址: **SCS_BA = 0xE000_E000**				
SYST_ CSR	SCS_BA+0x10	R/W	SysTick 控制和状态寄存器	0x0000_0000
SYST_ RVR	SCS_BA+0x14	R/W	SysTick 重新加载值寄存器	0xXXXX_XXXX
SYST_ CVR	SCS_BA+0x18	R/W	SysTick 当前值寄存器	0xXXXX_XXXX

6.2.10.2 系统定时器控制寄存器描述

SysTick 控制和状态寄存器(SYST_CSR)

寄存器	偏移地址	R/W	描述	复位值
SYST_CSR	SCS_BA+0x10	R/W	SysTick 控制和状态寄存器	0x0000_0000

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

位	描述	
[31:17]	Reserved	保留
[16]	COUNTFLAG	SysTick 计数器标志位 从上次读取该寄存器后，如果定时器计数到0，则返回1 计数从1到0转变时，COUNTFLAG 置位 在读或写当前寄存器时，COUNTFLAG被清零
[15:3]	Reserved	保留
[2]	CLKSRC	SysTick时钟源选择 0 = 时钟源为(可选)外部参考时钟 1 = 内核时钟用于SysTick
[1]	TICKINT	SysTick中断使能 0 = 向下计数到 0 不会引起 SysTick 异常而挂起。软件根据 COUNTFLAG 来确定是否已经发生计数到 0。 1 = 向下计数到 0 将引起 SysTick 异常而挂起。软件清 SysTick 当前值寄存器(SYST_CVR)将不会导致 SysTick 挂起。
[0]	ENABLE	SysTick计数使能 0 = 禁用计数器 1 = 计数器运行于连拍方式 (multi-shot manner)

SysTick 重新加载值寄存器(SYST_RVR)

寄存器	偏移地址	R/W	描述	复位值
SYST_RVR	SCS_BA+0x14	R/W	SysTick 重新加载值寄存器	0xXXXX_XXXX

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

位	描述	
[31:24]	Reserved	保留
[23:0]	RELOAD	SysTick 重新加载值 当计数器达到 0 时，这个值加载到当前值寄存器

SysTick 当前值寄存器(SYST_CVR)

寄存器	偏移地址	R/W	描述	复位值
SYST_CVR	SCS_BA+0x18	R/W	SysTick 当前值寄存器	0xXXXX_XXXX

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

位	描述	
[31:24]	Reserved	保留
[23:0]	CURRENT	SysTick当前数值 当前计数值。该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，该寄存器为写清除 (write-clear)。软件写入任何值将清寄存器为 0。不支持位RAZ(参看SysTick重新加载值寄存器)

6.2.11 嵌套向量中断控制器(NVIC)

NVIC与处理器内核接口的紧密耦合，能够使中断低延时处理以及后到的中断有效处理。NVIC保持了堆栈，嵌套，中断来使能链尾中断。虽然只能在优先模式下完全访问NVIC，但在用户模式下，如果使能了配置和控制的相关寄存器，也可以产生中断进入挂起状态。任何其它用户模式下访问会导致总线故障。除有特别说明之外，用户可以使用字节，半字，字的方式访问所有的NVIC寄存器。NVIC寄存器都放在SCS(系统控制位置)的里面。所有的NVIC寄存器和系统调试寄存器都是低字节序而不管处理器的字节顺序状态。

NVIC主要特点有:

- 可实现定义的中断数量，范围在1-240个中断。
- 每个中断有可配置的0到15级的优先级，级数越高，优先级就越低，0级为最高中断优先级。
- 电平和脉冲侦测中断信号
- 动态重新分配中断的优先级
- 组优先级和子优先级域设置组优先级
- 链尾中断
- 外部的不可屏蔽中断
- 中断唤醒控制器(WIC)支持超低功耗模式

处理器在异常发生时自动压栈保存状态，异常离开时自动出栈恢复状态，而且不需要额外的指令。该机制使异常低延时处理。

*6.2.11.1*异常模式和系统中断映射

下表列出了NUC505系列支持的异常模式。所有的中断以及部分异常有16级的优先等级设置。最高优先等级可以配置为”0x00”，最低优先等级可以配置为”0xF0”(低4位一直为0)。所有可配置的中断默认优先等级为”0x00”。优先等级”0”在系统中是第4级优先级，仅次于3个系统异常“Reset”, “NMI” 和 “Hard Fault”。

当接收到中断时，处理器将自动从内存中的向量表获取中断服务程序(ISR)的起始地址。在系统复位时，向量表地址固定在0x00000000。特权模式下可以改变VTOR，来重新安置向量表的起始地址到不同的内存位置，可安置的范围为0x00000080 到 0x3FFFFF80。

向量表包括复位后堆栈指针的初始值和所有异常处理器的入口地址。下面的向量号表示进入向量表的顺序。与异常处理入口相关的向量表如下部分描述。

异常类别	向量号	向量地址	优先级
Reset	1	0x00000004	-3
NMI	2	0x00000008	-2
Hard Fault	3	0x0000000C	-1
Memory Manager Fault	4	0x00000010	可配置
Bus Fault	5	0x00000014	可配置
Usage Fault	6	0x00000018	可配置
Reserved	7 ~ 10		保留
SVCall	11	0x0000002C	可配置
Debug Monitor	12	0x00000030	可配置
Reserved	13		保留
PendSV	14	0x00000038	可配置
SysTick	15	0x0000003C	可配置
Interrupt (IRQ0 ~ IRQ31)	16 ~ 47	0x00000000 + (Vector Number)*4	可配置

表 6.2-4 异常模型

向量号	中断号 (中断寄存器中对应位)	中断名称	中断描述
0 ~ 15	-	-	系统异常
16	0	PWR_INT	上电中断
17	1	WDT_INT	看门狗定时器中断
18	2	Reserved	保留
19	3	I2S_INT	I^2S 中断
20	4	EINT0_INT	外部GPIO Group 0 中断
21	5	EINT1_INT	外部GPIO Group 1 中断
22	6	EINT2_INT	外部GPIO Group 2 中断
23	7	EINT3_INT	外部GPIO Group 3 中断
24	8	SPIM_INT	SPIM 中断
25	9	USBD_INT	USB Device 20 中断
26	10	TM0_INT	Timer0中断
27	11	TM1_INT	Timer1中断

28	12	TM2_INT	Timer2中断
29	13	TM3_INT	Timer3中断
30	14	SDH_INT	SDH中断
31	15	PWM0_INT	PWM0中断
32	16	PWM1_INT	PWM1中断
33	17	PWM2_INT	PWM2中断
34	18	PWM3_INT	PWM3中断
35	19	RTC_INT	实时时钟中断
36	20	SPI0_INT	SPI0中断
37	21	I2C1_INT	I2C1中断
38	22	I2C0_INT	I2C0中断
39	23	UART0_INT	UART0中断
40	24	UART1_INT	UART1中断
41	25	ADC_INT	ADC中断
42	26	wwdt_INT	窗口看门狗定时器中断
43	27	USBH_INT	USB Host 1.1中断
44	28	UART2_INT	UART2中断
45	29	LVD_INT	低电压检测中断
46	30	SPI1_INT	SPI1中断
47	31	Reserved	保留

表 6.2-5 中断号表

6.2.11.2 操作描述

通过写相应中断的使能寄存器或清除使能寄存器的相关位，可以使能 NVIC 中断或禁用 NVIC 中断，使能寄存器通过写 1 使能和清除使能寄存器写 1 清除使能，读取两寄存器都返回当前相应中断的使能状态。当中断禁用时，中断声明将使中断挂起，因此中断不被激活，如果在禁用时中断被激活，该中断就保持在激活状态，直到通过复位或异常返回来时清除。清使能位可以阻止新的相应中断被激活。

NVIC 中断可以使用互补的寄存器对来挂起/取消挂起以使能/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，可以写 1 使能和写 1 禁用，读取两寄存器都返回当前相应中断的状态。寄存器 Clear-Pending 在中断响应时的不影响执行状态。

可以通过更新32位寄存器中的8位字段来设置NVIC 中断的优先级（每个寄存器支持4个中断）。

与 NVIC 相关的的通用寄存器都可以在系统控制空间的一块内存区域中访问，下一节将作出描述。

6.2.11.3NVIC 控制寄存器

R: 只读, **W:** 只写, **R/W:** 读/写

寄存器	偏移地址	R/W	描述	复位值
SCS基地址: **SCS_BA = 0xE000_E000** **NVIC_IPRn_BA = 0xE000_E400+(0x04*n)** **n = 0,1..7**				
NVIC_ISER0	SCS_BA+0x100	R/W	IRQ0 ~ IRQ31设置使能控制寄存器	0x0000_0000
NVIC_ICER0	SCS_BA+0x180	R/W	IRQ0 ~ IRQ31清使能控制寄存器	0x0000_0000
NVIC_ISPR0	SCS_BA+0x200	R/W	IRQ0 ~ IRQ31设置挂起控制寄存器	0x0000_0000
NVIC_ICPR0	SCS_BA+0x280	R/W	IRQ0 ~ IRQ31清挂起控制寄存器	0x0000_0000
NVIC_IABR0	SCS_BA+0x300	R/W	IRQ0 ~ IRQ31激活位寄存器	0x0000_0000
NVIC_IPRn	NVIC_IPRn_BA+0x00	R/W	IRQ0 ~ IRQ31先级控制寄存器	0x0000_0000
NVIC_STIR	SCS_BA+0xF00	W	软件触发中断寄存器	0x0000_0000

IRQ0 ~ IRQ31设置使能控制寄存器(NVIC_ISERn)

寄存器	偏移地址	R/W	描述	复位值
NVIC_ISER0	SCS_BA+0x100	R/W	IRQ0 ~ IRQ31设置使能控制寄存器	0x0000_0000

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

位	描述	
[31:0]	SETENA	**中断设置使能位** NVIC_ISER0寄存器用来使能相应的中断，这些位表示那些中断被使能了。 写操作: 0 = 无效 1 = 使能中断 读操作: 0 = 相关中断被禁止 1 = 相关中断状态已使能

IRQ0 ~ IRQ31清使能控制寄存器(NVIC_ICERn)

寄存器	偏移地址	R/W	描述	复位值
NVIC_ICER0	SCS_BA+0x180	R/W	IRQ0 ~ IRQ31清使能控制寄存器	0x0000_0000

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

位	描述	
[31:0]	CLRENA	**中断禁止位** NVIC_ICER0寄存器用来禁止相应的中断，这些位表示那些中断被禁止了。 写操作: 0 = 无效 1 = 禁止中断 读操作: 0 = 相关中断被禁止 1 = 相关中断状态已使能

IRQ0 ~ IRQ31设置挂起控制寄存器(NVIC_ISPRn)

寄存器	偏移地址	R/W	描述	复位值
NVIC_ISPR0	SCS_BA+0x200	R/W	IRQ0 ~ IRQ31设置挂起控制寄存器	0x0000_0000

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

位	描述	
[31:0]	SETPEND	**设置中断挂起位** NVIC_ISPR0寄存器用来强制相应的中断进入挂起状态，这些位表示那些中断已挂起。 写操作: 0 = 无效 1 = 强制中断进入挂起状态 读操作: 0 = 相关中断没有挂起 1 = 相关中断已挂起

IRQ0 ~ IRQ31清挂起控制寄存器(NVIC_ICPRn)

寄存器	偏移地址	R/W	描述	复位值
NVIC_ICPR0	SCS_BA+0x280	R/W	IRQ0 ~ IRQ31清挂起控制寄存器	0x0000_0000

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

位	描述	
[31:0]	CLRPEND	**清除挂起中断位** NVIC_ICPR0寄存器用来清除进入挂起状态的中断，这些位表示那些中断已挂起。 写操作: 0 = 无效 1 = 清除进入挂起状态的中断 读操作: 0 = 相关中断没有挂起 1 = 相关中断已挂起

IRQ0 ~ IRQ31激活位寄存器(NVIC_IABRn)

寄存器	偏移地址	R/W	描述	复位值
NVIC_IABR0	SCS_BA+0x300	R/W	IRQ0 ~ IRQ31激活位寄存器	0x0000_0000

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

位	描述	
[31:0]	ACTIVE	中断激活标志位 NVIC_IABR0寄存器表示哪个中断激活了 0 =中断没有激活 1 = 中断已激活

IRQ0 ~ IRQ31中断优先级寄存器(NVIC_IPRn)

寄存器	偏移地址	R/W	描述	复位值
NVIC_IPRn	NVIC_IPRn_BA+0x00	R/W	IRQ0 ~ IRQ31中断优先级寄存器	0x0000_0000

注: n = 0,1..7

31	30	29	28	27	26	25	24
PRI_4n+3				Reserved			
23	22	21	20	19	18	17	16
PRI_4n+2				Reserved			
15	14	13	12	11	10	9	8
PRI_4n+1				Reserved			
7	6	5	4	3	2	1	0
PRI_4n+0				Reserved			

位	描述	
[31:28]	**PRI_4n+3**	**IRQ_4n+3的优先级** “0” 表示最高优先级,“15” 表示最低优先级
[27:24]	**Reserved**	保留
[23:20]	**PRI_4n+2**	**IRQ_4n+2的优先级** “0” 表示最高优先级,“15” 表示最低优先级
[19:16]	**Reserved**	保留
[15:12]	**PRI_4n+1**	**IRQ_4n+1的优先级** “0” 表示最高优先级,“15” 表示最低优先级
[11:8]	**Reserved**	保留
[7:4]	**PRI_4n+0**	**IRQ_4n+0的优先级** “0” 表示最高优先级,“15” 表示最低优先级
[3:0]	**Reserved**	保留

软件触发中断寄存器(NVIC_STIR)

寄存器	偏移地址	R/W	描述	复位值
NVIC_STIR	SCS_BA+0xF00	W	软件触发中断寄存器	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							INTID
7	6	5	4	3	2	1	0
INTID							

位	描述	
[31:9]	Reserved	保留
[8:0]	INTID	**中断 ID** 通过软件写一个ID到STIR寄存器来产生一个中断。 当寄存器SCR的USERSETMPEND位设置为1时，软件可以访问STIR寄存器。 中断ID用来触发中断。范围是0-47。例如：0x03表示中断IRQ3。

6.2.12 系统控制寄存器映射和描述

系统控制寄存器管理了 Cortex-M4 的状态和操作模式，包括 CPUID，Cortex-M4的中断优先级和 Cortex-M4 的电源管理。

详情请参考 “ARM® Cortex™-M4 Technical Reference Manual” 与 “ARM® v6-M Architecture Reference Manual”。

R: 只读, **W**: 只写, **R/W**: 读/写

寄存器	偏移地址	R/W	描述	复位值
SCS基地址: **SCS_BA = 0xE000_E000**				
ICSR	SCS_BA+0xD04	R/W	中断控制和状态寄存器	0x0000_0000
AIRCR	SCS_BA+0xD0C	R/W	应用中断和复位控制寄存器	0xFA05_0000
SCR	SCS_BA+0xD10	R/W	系统控制寄存器	0x0000_0000
SHPR1	SCS_BA+0xD18	R/W	系统处理优先级寄存器 1	0x0000_0000
SHPR2	SCS_BA+0xD1C	R/W	系统处理优先级寄存器 2	0x0000_0000
SHPR3	SCS_BA+0xD20	R/W	系统处理优先级寄存器 3	0x0000_0000

中断控制状态寄存器(ICSR)

寄存器	偏移地址	R/W	描述	复位值
ICSR	SCS_BA+0xD04	R/W	中断控制状态寄存器	0x0000_0000

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

位	描述	
[31]	**NMIPENDSET**	**NMI 设置挂起位** 写操作: 0 = 该位写 0 无效 1 = 更改 NMI 异常状态为挂起 读操作: 0 = NMI 异常没有挂起 1 = NMI 异常挂起 因为 NMI 异常是最高优先级的异常，正常情况下处理器一旦检测到这一位被置1，就会立刻进入 NMI 异常处理程序。进入处理程序后处理器会将该位清为零。这意味着只有当处理器执行 NMI 异常处理程序时再次产生 NMI 信号，NMI 异常处理程序读取这一位的值将返回 1。
[30:29]	**Reserved**	保留
[28]	**PENDSVSET**	**PendSV 设置挂起位** 写操作: 0 = 该位写 0 无效 1 = 更改 PendSV 异常状态为挂起 读操作: 0 = PendSV 异常没有挂起 1 = PendSV 异常挂起 **注:** 向该位写1是将 PendSV 异常状态设为挂起的唯一方法。

[27]	**PENDSVCLR**	**PendSV 清除挂起位** 写操作: 0 = 该位写 0 无效 1 = 移除 PendSV 异常的挂起状态 **注:**该位为只写位。当想清除 PENDSV 位时，必须同时“向 PENDSVSET 写 0 和向 PENDSVCLR写 1。
[26]	**PENDSTSET**	**SysTick异常设置挂起位** 写操作： 0 = 该位写 0 无效 1 = 更改 SysTick 异常状态为挂起 读操作： 0 = SysTick 异常没有挂起 1 = SysTick 异常挂起
[25]	**PENDSTCLR**	**SysTick异常清除挂起位** 写操作： 0 = 该位写 0 无效 1 = 移除SysTick 异常的挂起状态 **注:**该位为只写位。当想清除 PENDST 位时，必须同时“向 PENDSTSET 写0 和 向 PENDSTCLR写 1”。
[24]	**Reserved**	保留
[23]	**ISRPREEMPT**	**中断先占位 (只读)** 若置位，则挂起的异常将从调试停止状态退出，进入活动状态。
[22]	**ISRPENDING**	**中断挂起标志，不包括 NMI 和 Faults：(只读)** 0 = 中断没有挂起 1 = 中断挂起.
[21:18]	**Reserved**	保留
[17:12]	**VECTPENDING**	**最高优先级异常的异常号** 该位表示使能挂起异常中的最高优先级异常的异常号： 0 = 没有异常挂起 非 0 = 挂起使能异常中的最高优先级异常的异常号。 通过该域的值可影响BASEPRI 和 FAULTMASK寄存器，但对PRIMASK寄存器没有影响。
[11]	**RETTOBASE**	**先占激活异常指示位** 表示是否有先占激活异常 0 = 有先占激活异常要执行 1 = 没有激活异常或者当前执行异常是唯一的激活异常。
[10:6]	**Reserved**	保留
[5:0]	**VECTACTIVE**	**当前执行异常处理的异常号** 0 = Thread 模式 非 0 = 当前执行异常处理的异常号

应用程序中断和复位控制寄存器(AIRCR)

寄存器	偏移地址	R/W	描述	复位值
AIRCR	SCS_BA+0xD0C	R/W	应用程序中断和复位控制寄存器	0xFA05_0000

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

位	描述	
[31:16]	VECTORKEY	**寄存器的访问键** 当写该寄存器时，写入值必须为 0x05FA，否则写动作将是不可预测的。 VECTORKEY用于保护系统复位或异常清除状态时误写该寄存器。
[15]	ENDIANNESS	**时间次序** 0 = 低字节序 1 = 高字节序
[14:11]	Reserved	保留
[10:8]	PRIGROUP	**中断优先级分组** 该域决定抢占优先级组从非抢占优先级中分开
[7:3]	Reserved	保留
[2]	SYSRESETREQ	**系统复位请求** 向该位写 1，产生复位信号给芯片，表示有复位请求。 该位为只写位，在复位时自动清零。
[1]	VECTCLRACTIVE	**异常激活状态清除位** 对于固定和配置的异常，设置该位为1将清除所有激活状态信息。 该位为只写位，而且只有当内核停止时可以被写。 注：这是用来重新初始化堆栈调试器的职责。
[0]	Reserved	**保留**

优先级分组	二进制点	抢占优先级组位	非抢占优先级位	抢占优先级个数	非抢占优先级个数
0b000	bxxxxxxx.y	[7:1]	[0]	128	2
0b001	bxxxxxx.yy	[7:2]	[1:0]	64	4
0b010	bxxxxx.yyy	[7:3]	[2:0]	32	8
0b011	bxxxx.yyyy	[7:4]	[3;0]	16	16
0b100	bxxx.yyyyy	[7:5]	[4:0]	8	32
0b101	bxx.yyyyyy	[7:6]	[5:0]	4	64
0b110	bx.yyyyyyy	[7]	[6:0]	2	128
0b111	b.yyyyyyyy	None	[7:0]	1	256

表 6.2-6优先级分组

系统控制寄存器(SCR)

寄存器	偏移地址	R/W	描述	复位值
SCR	SCS_BA+0xD10	R/W	系统控制寄存器	0x0000_0000

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

位	描述	
[31:5]	Reserved	保留.
[4]	SEVONPEND	**挂起状态时的发送事件位** 0 = 只有使能中断或者事件可以唤醒处理器，不包括禁用中断在内。 1 = 使能事件和所有中断，包括禁用中断，都可以唤醒处理器。 当一个事件或中断进入挂起状态时，事件信号从 WFE 唤醒处理器。如果处理器不是在等待该事件，那么这个事件会被注册到并影响下一个 WFE。 处理器总是会在执行一个 SEV 指令或者一个外部事件时被唤醒。
[3]	Reserved	保留
[2]	SLEEPDEEP	**系统休眠或空闲模式选择** 控制处理器使用空闲或者休眠作为它的低功耗模式： 0 = 空闲 1 = 休眠
[1]	SLEEPONEXIT	**Sleep-On-Exit 控制使能位** 表示当从 Handler 模式切换到 Thread 模式时，是否使用 sleep-on-exit。 0 = 当切换到 Thread 模式时不空闲。 1 = 当从某个 ISR 切换到 Thread 模式时，进入空闲或者休眠。 设置该位为1 使能一个中断驱动应用，避免返回到一个空的主函数应用。
[0]	Reserved	保留

系统处理优先级寄存器1 (SHPR1)

寄存器	偏移地址	R/W	描述	复位值
SHPR1	SCS_BA+0xD18	R/W	系统处理优先级寄存器1	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留
[23:16]	**PRI_6**	**系统处理的优先级6 – UsageFault**
[15:8]	**PRI_5**	**系统处理的优先级5 – BusFault**
[7:0]	**PRI_4**	**系统处理的优先级4 – MemManage**

系统处理优先级寄存器2 (SHPR2)

寄存器	偏移地址	R/W	描述	复位值
SHPR2	SCS_BA+0xD1C	R/W	系统处理优先级寄存器2	0x0000_0000

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

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

系统处理优先级寄存器3 (SHPR3)

寄存器	偏移地址	R/W	描述	复位值
SHPR3	SCS_BA+0xD20	R/W	系统处理优先级寄存器3	0x0000_0000

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

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

6.3 时钟控制器

6.3.1 概述

时钟控制器为整个芯片提供时钟源。包括AHB、APB时钟和所有外设时钟，比如USB设备，USB主机，UART等等。有两路PLL时钟分别是PLL和APLL来源于外部HXT时钟输入。PLL为处理器工作提供内部高速时钟频率，APLL可以产生更精确地频率为音频解码提供时钟。他们还实现了功率控制功能, 包括单独的时钟打开或关闭控制寄存器、时钟源选择器和分频器。这些功能使额外功耗得到了最小化让芯片运行在合理的状态。在掉电模式下，时钟控制器会关闭外部高速晶振和内部，以降低整个系统功耗

6.3.2 时钟框图

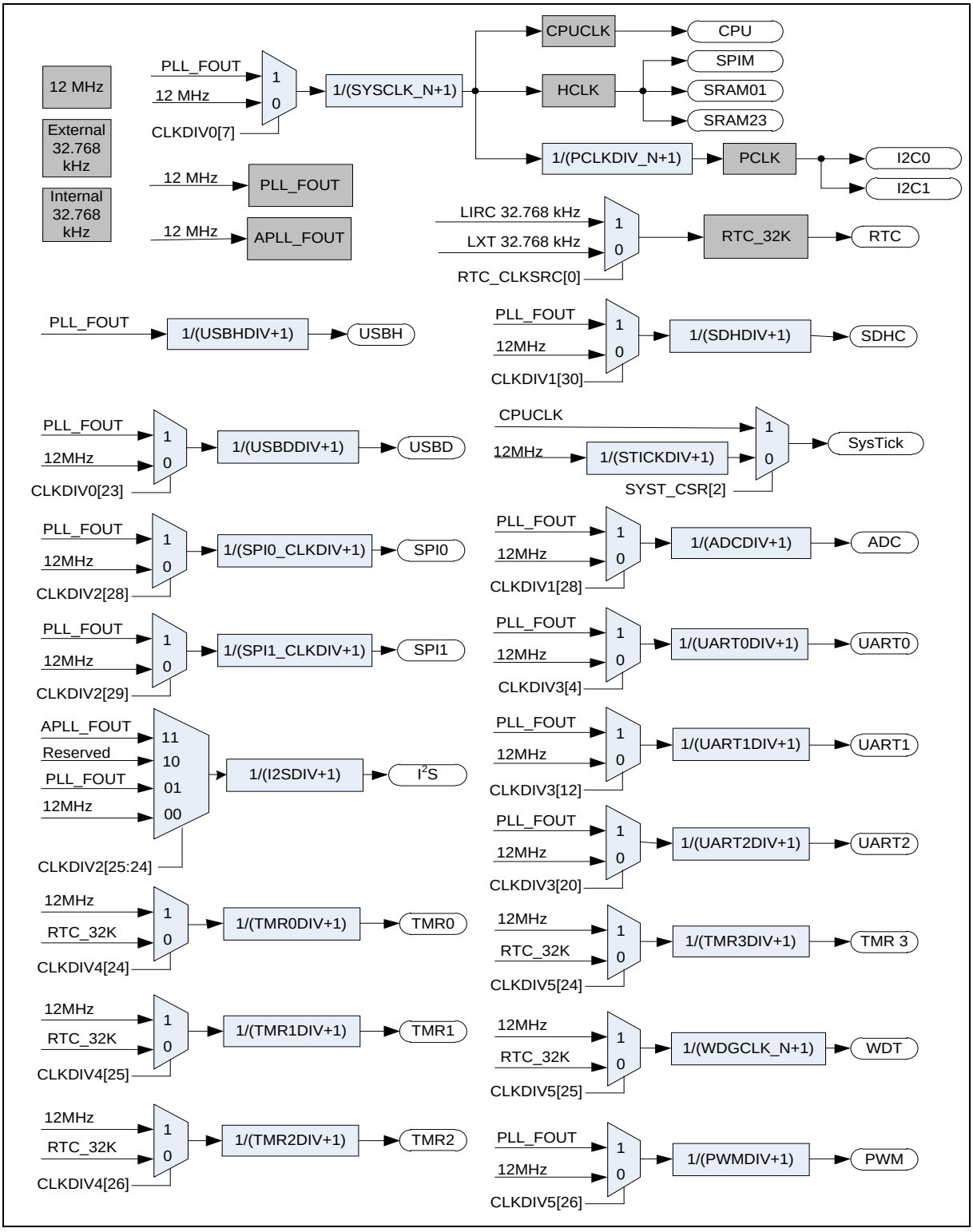


图 6.3-1时钟发生器全局框图

6.3.3 时钟发生器

时钟发生器由如下4个时钟源组成：

- 实时时钟源(RTC_CLK)可以选择外部32.768 kHz低速晶振（LXT）或是内部32.768 kHz低速RC振荡器(LIRC)
- 外部12 MHz高速晶振(HXT)
- 可编程的系统PLL输出时钟频率(PLL_FOUT)
- 可编程的音频PLL输出时钟频率(APLL_FOUT)

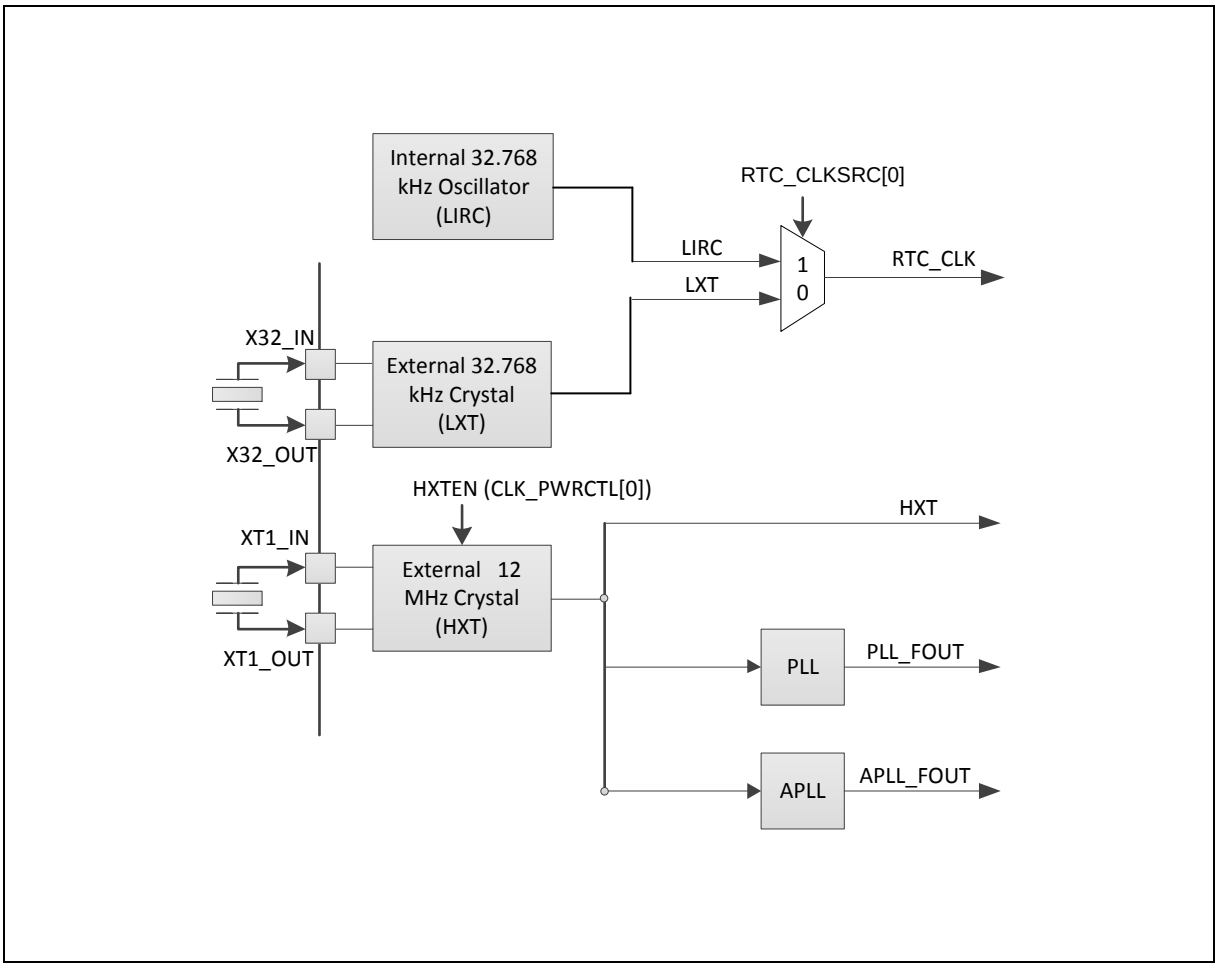


图 6.3-2时钟发生器框图

外部晶振和两个电容是连接到“XT1_IN / X32_IN”和 pad “XT1_OUT / X32_OUT”引脚上。两个电容值可能随不同的晶振而改变。相应晶振振荡器必须要选着与之相应的电容值和电阻值。下面是我们推荐电容值和电阻值。

晶振振荡器	电容值	电阻值
12 MHz	20pF	1 MΩ
32.768 kHz	33pF	10 MΩ

表 6.3-1 推荐电容值和电阻值

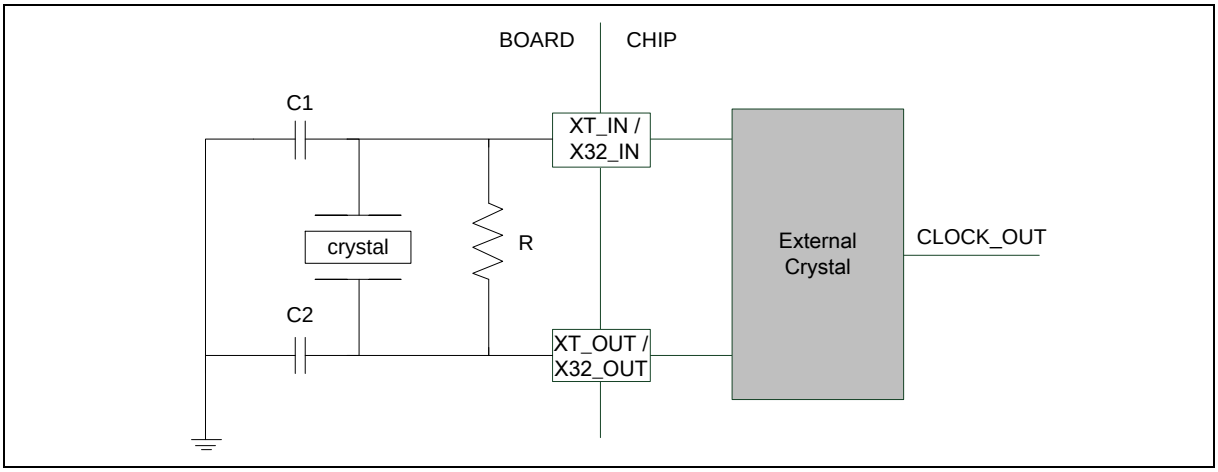


图 6.3-3 晶振振荡电路

6.3.4 掉电模式时钟

当芯片进入掉电模式，系统时钟和一些时钟源以及一些外设时钟将被关闭。也有一些时钟源与外设时钟仍在工作。

如下时钟仍在工作：

- 时钟发生器
  - 32.768 kHz内部低速RC振荡器(LIRC)
  - 32.768 kHz外部低速晶振时钟(LXT)

在掉电模式，如果发生唤醒事件，被关闭的时钟在 PDWKPSC (CLK_PWRCTL[23:8]) x 256 HXT 周期内将重新开启

6.3.5 寄存器映射

R: 只读, **W:** 只写, **R/W:** 读/写

寄存器	偏移地址	R/W	描述	复位值
CLKCTRL基地址: **CLK_BA = 0x4000_0200**				
CLK_PWRCTL	CLK_BA+0x00	R/W	系统掉电控制寄存器	0x00FF_FF03
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB设备时钟使能控制寄存器	0x0000_000F
CLK_APBCLK	CLK_BA+0x08	R/W	APB 设备时钟使能控制寄存器	0x0000_0000
CLK_CLKDIV0	CLK_BA+0x10	R/W	时钟分频数目控制寄存器0	0x0001_0000
CLK_CLKDIV1	CLK_BA+0x14	R/W	时钟分频数目控制寄存器1	0x0000_0000
CLK_CLKDIV2	CLK_BA+0x18	R/W	时钟分频数目控制寄存器2	0x0000_0000
CLK_CLKDIV3	CLK_BA+0x1C	R/W	时钟分频数目控制寄存器3	0x0000_0000
CLK_PLLCTL	CLK_BA+0x20	R/W	PLL 控制寄存器	0x0000_20CF
CLK_APLLCTL	CLK_BA+0x28	R/W	APLL 控制寄存器	0x0003_0000
CLK_CLKDIV4	CLK_BA+0x30	R/W	时钟分频数目控制寄存器4	0x0000_0000
CLK_CLKDIV5	CLK_BA+0x34	R/W	时钟分频数目控制寄存器5	0x0000_0000

6.3.6 寄存器描述

系统掉电控制寄存器(CLK_PWRCTL)

芯片时钟源来自外部晶振。可以通过HXTEN控制晶振振荡器的开/关。当关闭晶振是芯片进入掉电状态。为了避免输出不稳定的时钟给系统，时钟控制器增加了PDWKPSC计数器。在时钟计数器计数到PDWKPSC (CLK_PWRCTL[23:8]) x 256振荡周期，时钟控制器开始输出时钟给系统。

寄存器	偏移地址	R/W	描述	复位值
CLK_PWRCTL	CLK_BA+0x00	R/W	系统掉电控制寄存器	0x00FF_FF03

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

位	描述	
[31:25]	**Reserved**	保留
[24]	**PDWTCPU**	**进入掉电模式条件控制位** 0 =当HXTEN位被置1，芯片进入掉电模式 1 =当PDWTCPU和HXTEN同时置1且CPU执行WFI指令时芯片进入掉电模式
[23:8]	**PDWKPSC**	**PDWKPSC 计数器** 假设在PDWKPSC x 256 振荡周期之后HXT稳定，时钟控制器在计数器没有达到PDWKPSC x 256是不会输出时钟给系统
[7:4]	**Reserved**	保留
[3]	**PDWKIEN**	**掉电模式唤醒中断使能位** 0 = 禁用掉电模式唤醒中断 1 = 使能掉电模式唤醒中断 **注1:** 当PDWKIF 和 PDWKIEN 都为1时，该中断将产生
[2]	**PDWKIF**	**芯片掉电模式唤醒中断状态** 该位由掉电唤醒事件置位，表示从掉电模式恢复过来。。 如果GPIO, USBH, USBD, UART, TIMER, WDT, RTC 或I^2C唤醒发生，该标志置位。 **注1：** 该位写 1清零。 **注2：**只有在PDWKIEN (CLK_PWRCTL[24])置 1 后，该位才工作。

[1]	**HXTCTL**	**掉电模式下唤醒预除频计数器使能控制位** HXT预除频控制从掉电下唤醒时间。芯片会延时PDWKPSC*256周期直到HXT稳定 0 = PDWKPSC 计数器禁用. 1 = PDWKPSC计数器使能.
[0]	**HXTEN**	**晶振（掉电）控制** 0 = 晶振关闭 (掉电). 1 = 晶振打开 (正常运行).

寄存器/指令模式	HXTEN	PDWKIEN	PDWTCPU	CPU 执行 WFI 指令	禁用时钟
正常运行模式	1	0	0	NO	所有时钟由相关控制寄存器控制
掉电模式	0	1	1	YES	大部分时钟被禁用，仅LIRC/LXT 和选LIRC/LXT为时钟源的RTC/WDT/Timer外设时钟还能运行。

表 6.3-2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源唤醒芯片。用户必须在设置HXTEN位(CLK_PWRCTL[0]) 之前，使能相关的中断源及相应的NVIC IRQ 使能位 (NVIC_ISER) 从而保证芯片能进入掉电模式以及能够成功被唤醒。

AHB设备时钟使能控制寄存器(CLK_AHBCLK)

该寄存器的各位用于使能/禁用AMBA时钟、AHB总线设备时钟

寄存器	偏移地址	R/W	描述	复位值
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB设备时钟使能控制寄存器	0x0000_000F

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						USBHCKEN	Reserved
7	6	5	4	3	2	1	0
Reserved	USBDCKEN	SDHCKEN	Reserved	SPIMCKEN	ROMCKEN	SRAM23CKEN	SRAM01CKEN

位	描述	
[31:10]	**Reserved**	保留
[9]	**USBHCKEN**	**USB 主机控制器时钟使能位** 0 = 禁止USB主机外设时钟 1 = 使能USB 主机外设时钟
[8:7]	**Reserved**	保留
[6]	**USBDCKEN**	**USB 设备时钟使能位** 0 = 禁用USB 设备时钟 1 = 使能USB 设备时钟
[5]	**SDHCKEN**	**SDH 时钟使能位** 0 = 禁用SDH设备时钟 1 = 使能SDH设备时钟
[4]	**Reserved**	保留
[3]	**SPIMCKEN**	**SPIM 时钟使能位** 0 = 禁用SPIM设备时钟 1 = 使能SPIM设备时钟
[2]	**ROMCKEN**	**ROM 时钟使能位** 0 = 禁用ROM设备时钟 1 = 使能ROM设备时钟

[1]	**SRAM23CKEN**	**SRAM#2 时钟使能位** 0 = 禁用SRAM#2设备时钟 1 = 使能SRAM#2设备时钟
[0]	**SRAM01CKEN**	**SRAM#1 时钟使能位** 0 = 禁用SRAM#1设备时钟 1 = 使能SRAM#1设备时钟

APB 设备时钟使能控制寄存器(CLK_APBCLK)

该寄存器各位用于使能/禁用APB外设控制器时钟

寄存器	偏移地址	R/W	描述	复位值
CLK_APBCLK	CLK_BA+0x08	R/W	APB 设备时钟使能控制寄存器	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
ADCCKEN	I2SCKEN	UART2CKEN	UART1CKEN	UART0CKEN	SPI1CKEN	SPI0CKEN	PWMCKEN
7	6	5	4	3	2	1	0
RTCCKEN	I2C1CKEN	I2C0CKEN	WDTCKEN	TMR3CKEN	TMR2CKEN	TMR1CKEN	TMR0CKEN

位	描述	
[31:16]	**Reserved**	保留
[15]	**ADCCKEN**	**ADC时钟使能位** 0 = 禁用ADC时钟 1 = 使能ADC时钟
[14]	**I2SCKEN**	**I^2S时钟使能位** 0 = 禁用I^2S时钟 1 = 使能I^2S时钟
[13]	**UART2CKEN**	**UART2时钟使能位** 0 = 禁用UART2时钟 1 = 使能UART2时钟
[12]	**UART1CKEN**	**UART1时钟使能位** 0 = 禁用UART1时钟 1 = 使能UART1时钟
[11]	**UART0CKEN**	**UART0时钟使能位** 0 = 禁用UART0时钟 1 = 使能UART0时钟
[10]	**SPI1CKEN**	**SPI1时钟使能位** 0 = 禁用SPI1时钟 1 = 使能SPI1时钟
[9]	**SPI0CKEN**	**SPI0时钟使能位** 0 = 禁用SPI0时钟 1 = 使能SPI0时钟

[8]	**PWMCKEN**	**PWM时钟使能位** 0 = 禁用PWM时钟 1 = 使能PWM时钟
[7]	**RTCCKEN**	**RTC时钟使能位** 0 = 禁用RTC时钟 1 = 使能RTC时钟
[6]	**I2C1CKEN**	**I2C1时钟使能位** 0 = 禁用I2C1时钟 1 = 使能I2C1时钟
[5]	**I2C0CKEN**	**I2C0时钟使能位** 0 = 禁用I2C0时钟 1 = 使能I2C0时钟
[4]	**WDTCKEN**	**Watchdog 定时器时钟使能位** 0 = 禁用Watchdog定时器时钟 1 = 使能Watchdog定时器时钟
[3]	**TMR3CKEN**	**TIMER3时钟使能位** 0 = 禁用TIMER3时钟 1 = 使能TIMER3时钟
[2]	**TMR2CKEN**	**TIMER2时钟使能位** 0 = 禁用TIMER2时钟 1 = 使能TIMER2时钟
[1]	**TMR1CKEN**	**TIMER1时钟使能位** 0 = 禁用TIMER1时钟 1 = 使能TIMER1时钟
[0]	**TMR0CKEN**	**TIMER0时钟使能位** 0 = 禁用TIMER0时钟 1 = 使能TIMER0时钟

时钟分频数目控制寄存器0 (CLK_CLKDIV0)

在时钟切换前，相关时钟源（预选和新选）必须打开

寄存器	偏移地址	R/W	描述	复位值
CLK_CLKDIV0	CLK_BA+0x10	R/W	时钟分频数目控制寄存器0	0x0001_0000

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

位	描述	
[31:28]	**Reserved**	保留.
[27:24]	**USBHDIV**	**USB主机时钟除频数** 实际时钟除频数为 (USBHDIV+1)故, USBH_CLK = USBH_SrcCLK / (USBHDIV+1).
[23]	**USBDSEL**	**USB 设备时钟源选择(USBD_SrcCLK)** 0 = USB 设备时钟源为HXT 1 = USB 设备时钟源为PLL_FOUT
[22:21]	**Reserved**	保留.
[20:16]	**USBDDIV**	**USB 设备时钟除频数** 实际时钟除频数为(USBDDIV+1) 故, USBD_CLK = USBD_SrcCLK / (USBDDIV+1).
[15:12]	**Reserved**	保留
[11:8]	**PCLKDIV**	**APB_CLK时钟除频数** 实际时钟除频数为(PCLKDIV+1) 故, APB_CLK = HCLK / (PCLKDIV+1).
[7]	**HCLKSEL**	**系统时钟源选择 (SYS_SrcCLK)** 0 = 系统时钟源为 HXT. 1 =系统时钟源为PLL_FOUT.
[6:4]	**Reserved**	保留

[3:0]	**HCLKDIV**	**SYS_CLK时钟除频数** 实际时钟除频数为(HCLKDIV+1) 故, SYS_CLK = SYS_SrcCLK / (HCLKDIV+1).

时钟分频数目控制寄存器1 (CLK_CLKDIV1)

寄存器	偏移地址	R/W	描述	复位值
CLK_CLKDIV1	CLK_BA+0x14	R/W	时钟分频数目控制寄存器1	0x0000_0000

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

位	描述	
[31]	**Reserved**	保留
[30]	**SDHSEL**	**SDH 时钟源选择 (SDH_SrcCLK)** 0 = SDH 时钟源为HXT. 1 = SDH 时钟源为PLL_FOUT.
[29]	**Reserved**	保留
[28]	**ADCSEL**	**ADC时钟源选择(ADC_SrcCLK)** 0 = ADC 时钟源为HXT. 1 = ADC 时钟源为PLL_FOUT.
[27]	**Reserved**	保留
[26:16]	**SDHDIV**	**SDH时钟除频数** 实际时钟除频数为(SDHDIV+1)故, SDH_CLK = SDH_SrcCLK / (SDHDIV+1).
[15:8]	**STICKDIV**	**SYS_TICK时钟除频数** 实际时钟除频数为(STICKDIV+1) 故, SYS_TICK_CLK = SYS_TICK_SrcCLK / (STICKDIV+1).
[7:0]	**ADCDIV**	**ADC时钟除频数** 实际时钟除频数为(ADCDIV+1) 故, ADC_CLK = ADC_SrcCLK / (ADCDIV+1).

时钟分频数目控制寄存器2 (CLK_CLKDIV2)

寄存器	偏移地址	R/W	描述	复位值
CLK_CLKDIV2	CLK_BA+0x18	R/W	时钟分频数目控制寄存器2	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		SPI1SEL	SPI0SEL	Reserved		I2SSEL	
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
I2SDIV							

位	描述	
[31:30]	**Reserved**	保留
[29]	**SPI1SEL**	**SPI1 时钟源选择 (SPI1SEL)** 0 = SPI1 时钟源为HXT. 1 = SPI1 时钟源为PLL_FOUT.
[28]	**SPI0SEL**	**SPI0时钟源选择(SPI0SEL)** 0 = SPI0 时钟源为HXT. 1 = SPI0 时钟源为PLL_FOUT.
[27:26]	**Reserved**	保留
[25:24]	**I2SSEL**	**I^2S时钟源选择(I2S_SrcCLK)** 00 = I^2S 时钟源为HXT. 01 = I^2S 时钟源为PLL_FOUT. 10 = 保留 11 = I^2S 时钟源为APLL_FOUT.
[23:8]	**Reserved**	保留
[7:0]	**I2SDIV**	**I^2S时钟除频数** 实际时钟除频数为(I2SDIV+1)故, I2S_CLK = I2S_SrcCLK / (I2SDIV+1)

时钟分频数目控制寄存器3 (CLK_CLKDIV3)

寄存器	偏移地址	R/W	描述	复位值
CLK_CLKDIV3	CLK_BA+0x1C	R/W	时钟分频数目控制寄存器3	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			UART2SEL	UART2DIV			
15	14	13	12	11	10	9	8
Reserved			UART1SEL	UART1DIV			
7	6	5	4	3	2	1	0
Reserved			UART0SEL	UART0DIV			

位	描述	
[31:21]	**Reserved**	保留
[20]	**UART2SEL**	**UART2 时钟源选择(UART2_SrcCLK)** 0 = UART2 时钟源为HXT. 1 = UART2 时钟源为PLL_FOUT.
[19:16]	**UART2DIV**	**UART2时钟除频数** 实际时钟除频数为(UART2DIV +1) 故, UART2_CLK = UART2_SrcCLK / (UART2DIV +1).
[15:13]	**Reserved**	保留.
[12]	**UART1SEL**	**UART1时钟源选择(UART1_SrcCLK)** 0 = UART1 时钟源为HXT. 1 = UART1 时钟源为PLL_FOUT.
[11:8]	**UART1DIV**	**UART1时钟除频数** 实际时钟除频数为(UART1DIV +1) 故, UART1_CLK = UART1_SrcCLK / (UART1DIV +1).
[7:5]	**Reserved**	保留.
[4]	**UART0SEL**	**UART0时钟源选择(UART0_SrcCLK)** 0 = UART0 时钟源为HXT. 1 = UART0 时钟源为PLL_FOUT.
[3:0]	**UART0DIV**	**UART0时钟除频数** 实际时钟除频数为(UART0DIV +1) 故, UART0_CLK = UART0_SrcCLK / (UART0DIV +1).

PLL 控制寄存器（CLK_PLLCTL）

寄存器	偏移地址	R/W	描述	复位值
CLK_PLLCTL	CLK_BA+0x20	R/W	PLL 控制寄存器	0x0000_20CF

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

位	描述	
[31:18]	**Reserved**	保留
[17]	**PD**	**掉电模式** 0 = PLL 正常模式 1 = PLL 进入掉电模式（默认）
[16]	**BP**	**PLL旁路控制** 0 = PLL 正常模式（默认） 1 = PLL 时钟输出与PLL时钟源(FIN)输入相同
[15:13]	**OUTDIV**	**输出分频控制位(P)** 设置输出分频因素范围1到8
[12:7]	**INDIV**	**参考输入分频控制位(M)** 设置参考输入分频因素范围1到64
[6:0]	**FBDIV**	**反馈分频控制位(N)** 设置反馈分频因素范围1到128

频率计算

PLL的输入时钟频率和输出频率之间的关系取决于内部分频器结构，该分频器分频比如下。

M	INDIV[5:0]+1	输入分频
N	FBDIV[6:0]+1	反馈分频

P	OUTDIV[2:0]+1	输出分频

在锁定模式下，PLL频率的关系由以下方程定义：

$$F_{UPLL_clko} = F_{HXT} * \frac{N}{M} * \frac{1}{P}$$

约束条件：

1. $5\text{MHz} \leq \frac{F_{HXT}}{M} \leq 80\text{MHz}$
2. $300\text{MHz} \leq F_{PLL_clko} * P \leq 1000\text{MHz}$
3. $M \geq 1, N \geq 1$

其中F_{PLL_clko}是PLL输出的时钟频率。F_{HXT}是输入时钟频率。

APLL 控制寄存器(CLK_APLLCTL)

寄存器	偏移地址	R/W	描述	复位值
CLK_APLLCTL	CLK_BA+0x28	R/W	APLL 控制寄存器	0x0003_0000

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

位	描述	
[31:20]	**FRAC**	**Sigma-Delta调节器控制引脚** 设置反馈除频的小数 $Franction_number = FRAC[11:0] / 2^{12}$
[19]	**Reserved**	保留
[18]	**MODE**	**模式选择** 0 = 整数模式 1 = 小数模式
[17]	**Reserved**	保留
[16]	**PD**	**掉电使能控制** 0 = 掉电禁用 1 = 掉电使能
[15:13]	**OUTDIV**	**输出分频控制** 设置输出分频因素范围1到8 P = OUTDIV[2:0]+1.
[12:6]	**FBDIV**	**反馈分频控制** 设置反馈分频因素范围1到128 N = FBDIV[6:0]+1.
[5:0]	**INDIV**	**参考输入分频控制** 设置参考输入分频因素范围1到64 M = INDIV[5:0]+1.

整数模式下频率计算

APLL的输入时钟频率和输出频率之间的关系取决于内部分频器结构，该分频器分频比如下：

M	INDIV[5:0]+1	输入分频
N	FBDIV[6:0]+1	反馈分频
P	OUTDIV[2:0]+1	输出分频

在锁定模式下，APLL频率的关系由以下方程定义：

$$F_{APLL_clko} = F_{HXT} * \frac{N}{M} * \frac{1}{P}$$

约束条件：

1. $2.5\text{MHz} \leq \frac{F_{HXT}}{M} \leq 80\text{MHz}$
2. $200\text{MHz} \leq F_{APLL_clko} * P \leq 500\text{MHz}$
3. $M \geq 1, N \geq 1$

其中F_{APLL_clko}是PLL输出的时钟频率。F_{HXT}是输入时钟频率。

小数模式下频率计算

在该模式下，内部分频器的配置和整数模式下相同：

M	INDIV[5:0]+1	输入分频
N	FBDIV[6:0]+1	反馈分频
P	OUTDIV[2:0]+1	输出分频

不同的是通过设置FRAC[11:0]可以得到一个小数分频数N.X而不是在整数模式下的N，其中：

X=FRAC[11:0]/ 2^{12} ；小数数字

N.X=N+X ；小数分频

PLL模块和输出时钟频率的关系可以用以下公式计算：

$$F_{APLL_clko} = F_{HXT} * \frac{N.X}{M} * \frac{1}{P}$$

例子:

1. F_{HXT} = 12MHz, F_{APLL_clko} *P= 301.5MHz;
2. 计算: (12MHz / 2) * 50.25 / 1=301.5MHz

意味着 F_{HXT}/M=6MHz, M= 2, N.X=50.25, P=1

为了得到小数分频数 50.25, 使用以下方程:

$50.25 = 50+(2^{10} / 2^{12})$

故 N = 50, X=0.25, FRAC=2^{10}

3. 配置如下:

INDIV[5:0] = 000001,

FBDIV[6:0] = 0110001,

OUTDIV[2:0] = 000,

FRAC[11:0] = 010000000000.

4. 频率间隔：将 FRAC 值 2^{10} 改成 2^{10}+1 后小数分频数改变为 50.250244。F_{APLL_clko} 改变为 6MHz * 50.250244 =301.501464MHz，因此频率间隔是 301.501464 MHz – 301.5 MHz = 1.464 kHz.

时钟分频数目控制寄存器4 (CLK_CLKDIV4)

寄存器	偏移地址	R/W	描述	复位值
CLK_CLKDIV4	CLK_BA+0x30	R/W	时钟分频数目控制寄存器4	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					TMR2SEL	TMR1SEL	TMR0SEL
23	22	21	20	19	18	17	16
TMR2DIV							
15	14	13	12	11	10	9	8
TMR1DIV							
7	6	5	4	3	2	1	0
TMR0DIV							

位	描述	
[31:27]	**Reserved**	保留
[26]	**TMR2SEL**	**Timer2时钟源选择(TMR2_SrcCLK)** 0 = Timer2 时钟源为RTC_CLK. 1 = Timer2 时钟源为from HXT.
[25]	**TMR1SEL**	**Timer1时钟源选择(TMR1_SrcCLK)** 0 = Timer1 时钟源为RTC_CLK. 1 = Timer1 时钟源为 HXT.
[24]	**TMR0SEL**	**Timer0时钟源选择(TMR0_SrcCLK)** 0 = Timer0 时钟源为RTC_CLK. 1 = Timer0 时钟源为HXT.
[23:16]	**TMR2DIV**	**TMR2时钟除频数** 实际时钟除频数为(TMR2DIV+1) 故, TMR2_CLK = TMR2_SrcCLK / (TMR2CLK_N+1).
[15:8]	**TMR1DIV**	**TMR1时钟除频数** 实际时钟除频数为(TMR1DIV+1) 故, TMR1_CLK = TMR1_SrcCLK / (TMR1DIV+1).
[7:0]	**TMR0DIV**	**TMR0时钟除频数** 实际时钟除频数为(TMR0DIV+1) 故, TMR0_CLK = TMR0_SrcCLK / (TMR0DIV+1).

时钟分频数目控制寄存器5 (CLK_CLKDIV5)

寄存器	偏移地址	R/W	描述	复位值
CLK_CLKDIV5	CLK_BA+0x34	R/W	时钟分频数目控制寄存器5	0x0000_0000

31	30	29	28	27	26	25	24
Reserved					PWMSEL	WDTSEL	TMR3SEL
23	22	21	20	19	18	17	16
PWMDIV							
15	14	13	12	11	10	9	8
WDTDIV							
7	6	5	4	3	2	1	0
TMR3DIV							

位	描述	
[31:27]	Reserved	保留
[26]	PWMSEL	**PWM时钟源选择(PWM_SrcCLK)** 0 = PWM 时钟源为HXT. 1 = PWM 时钟源为PLL_FOUT.
[25]	WDTSEL	**WDT时钟源选择(WDT_SrcCLK)** 0 = WDT 时钟源为RTC_CLK. 1 = WDT 时钟源为HXT.
[24]	TMR3SEL	**Timer3时钟源选择(TMR3_SrcCLK)** 0 = Timer3 时钟源为RTC_CLK. 1 = Timer3 时钟源为HXT.
[23:16]	PWMDIV	**PWM时钟除频数** 实际时钟除频数为(PWMDIV+1) 故, PWM_CLK = PWM_SrcCLK / (PWMDIV+1).
[15:8]	WDTDIV	**WDT时钟除频数** 实际时钟除频数为(WDTDIV+1) 故, WDT_CLK = WDT_SrcCLK / (WDTDIV+1).
[7:0]	TMR3DIV	**TMR3时钟除频数** 实际时钟除频数为(TM3DIV+1) 故, TMR3_CLK = TMR3_SrcCLK / (TMR3DIV+1).

6.4 通用 I/O (GPIO)

6.4.1 概述

NUC505 系列多达52个通用I/O管脚和其他功能管脚共享，这取决于芯片的配置。52个管脚分配在PA, PB, PC和 PD四个端口上。PA和PB有16个管脚，PC有15个管脚，PF有5个管脚。每个管脚都是独立的，都有相应的寄存器位来控制管脚功能模式与数据。

I/O管脚的I/O类型可由软件独立地配置为输入或推挽式的输出模式。复位之后，所有管脚的 I/O 管脚为输人模式没有使能上拉和下拉（除PB.2上拉使能）。每个I/O管脚有一个上拉电阻接到VDD 上和一个下拉电阻接到VSS上其阻值为40K~50K。用户可以设置Px_PUEN去控制I/O脚连接上拉或是下拉。

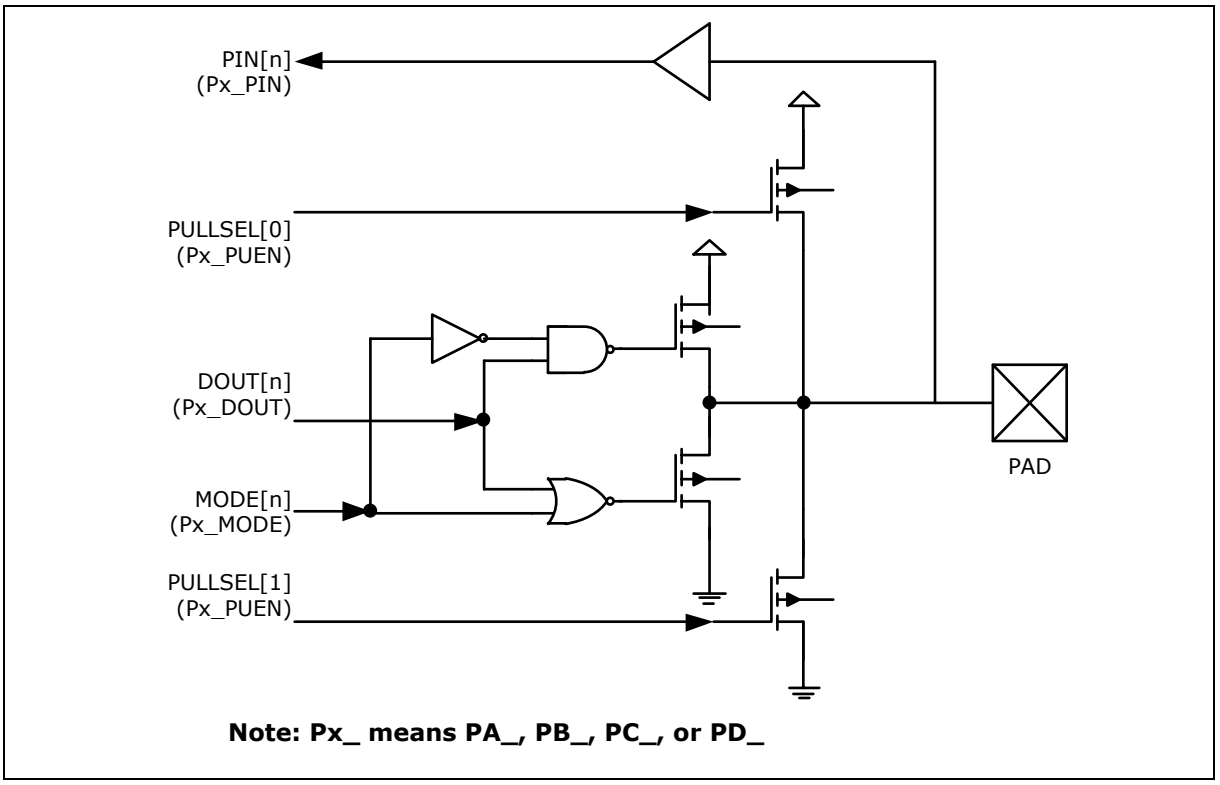


图 6.4-1 I/O 引脚模块框图

6.4.2 特性

- 二种 I/O 模式:
  - 推挽输出
  - 高阻态输入
- CMOS/施密特触发输入选择（参考6.2.9章节的SYS_GPAIBE寄存器）
- I/O可以配置为边沿触发的中断源

- I/O引脚内部有独立的上拉电阻和下拉电阻
- 使能管脚中断功能将也使能了唤醒功能。

6.4.3 基本配置

GPIO 管脚通过配置 SYS_GPA_MFPL, SYS_GPA_MFPH, SYS_GPB_MFPL, SYS_GPB_MFPH, SYS_GPC_MFPL, SYS_GPC_MFPH, 和 SYS_GPD_MFPL 寄存器来实现相应功能

6.4.4 功能描述

图 6.4-1和图 6.4-1 展示了GPIO的内部结构.它支持的输入方式和推挽输出模式。它具有独立的上拉和下拉电阻。

6.4.4.1 输入模式

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

6.4.4.2 推挽输出模式

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

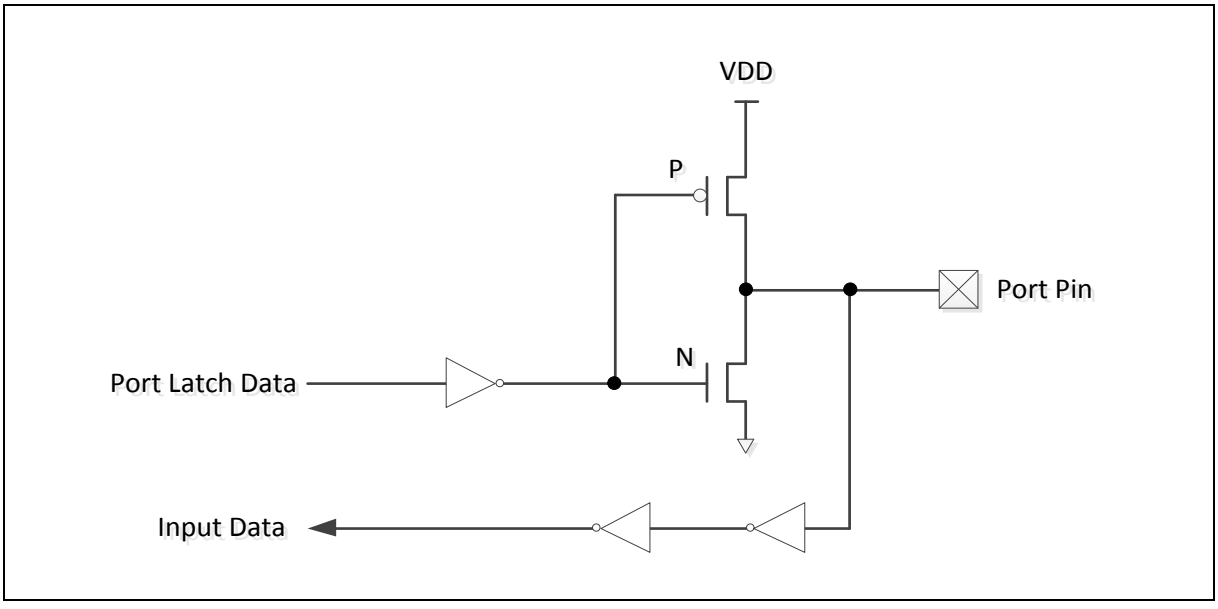


图 6.4-2 推挽输出

6.4.5 寄存器映射

R: 只读, **W:** 只写, **R/W:** 读/写

寄存器	偏移地址	R/W	描述	复位值
GPIO 基地址: **GPIO_BA = 0x400E_3000**				
PA_MODE	GPIO_BA+0x00	R/W	PA I/O 模式控制	0x0000_0000
PA_PUEN	GPIO_BA+0x04	R/W	PA I/O 上/下拉电阻控制	0x0000_0000
PA_DOUT	GPIO_BA+0x08	R/W	PA数据输出寄存器	0x0000_0000
PA_PIN	GPIO_BA+0x0C	R	PA管脚数值	0x0000_XXXX
PB_MODE	GPIO_BA+0x10	R/W	PB I/O模式控制	0x0000_0000
PB_PUEN	GPIO_BA+0x14	R/W	PB I/O上/下拉寄存器控制	0x0000_0010
PB_DOUT	GPIO_BA+0x18	R/W	PB 数据输出寄存器	0x0000_0000
PB_PIN	GPIO_BA+0x1C	R	PB 管脚数值	0x0000_XXXX
PC_MODE	GPIO_BA+0x20	R/W	PC I/O 模式控制	0x0000_0000
PC_PUEN	GPIO_BA+0x24	R/W	PC I/O上/下拉寄存器控制	0x0000_0000
PC_DOUT	GPIO_BA+0x28	R/W	PC 数据输出寄存器	0x0000_0000
PC_PIN	GPIO_BA+0x2C	R	PC 管脚数值	0x0000_XXXX
PD_MODE	GPIO_BA+0x30	R/W	PD I/O 模式控制	0x0000_0000
PD_PUEN	GPIO_BA+0x34	R/W	PD I/O上/下拉寄存器控制	0x0000_0000
PD_DOUT	GPIO_BA+0x38	R/W	PD 数据输出寄存器	0x0000_0000
PD_PIN	GPIO_BA+0x3C	R	PD 管脚数值	0x0000_00XX
GPIO_DBCTL	GPIO_BA+0x70	R/W	中断事件 (EINT) 防抖动控制	0x0000_0000
PA_INTSRCGP	GPIO_BA+0x80	R/W	PA 中断事件 (EINT) 源分组	0x0000_0000
PB_INTSRCGP	GPIO_BA+0x84	R/W	PB中断事件 (EINT) 源分组	0x5555_5555
PC_INTSRCGP	GPIO_BA+0x88	R/W	PC中断事件 (EINT) 源分组	0x2AAA_AAAA
PD_INTSRCGP	GPIO_BA+0x8C	R/W	PD中断事件 (EINT) 源分组	0x0000_03FF
PA_INTEN	GPIO_BA+0x90	R/W	PA中断使能控制寄存器	0x0000_0000
PB_INTEN	GPIO_BA+0x94	R/W	PB 中断使能控制寄存器	0x0000_0000
PC_INTEN	GPIO_BA+0x98	R/W	PC 中断使能控制寄存器	0x0000_0000
PD_INTEN	GPIO_BA+0x9C	R/W	PD 中断使能控制寄存器	0x0000_0000
GPIO_INTCTL	GPIO_BA+0xA0	R/W	中断锁存触发选择寄存器	0x0000_0000
PA_LATCHDAT	GPIO_BA+0xA4	R	PA 中断锁存值	0x0000_0000

PB_LATCHDAT	GPIO_BA+0xA8	R	PB 中断锁存值	0x0000_0000
PC_LATCHDAT	GPIO_BA+0xAC	R	PC中断锁存值	0x0000_0000
PD_LATCHDAT	GPIO_BA+0xB0	R	PD 中断锁存值	0x0000_0000
GPIO_INTSTSA_B	GPIO_BA+0xB4	R/W	PA和PB端口 EINT0~3 中断触发源指示	0x0000_0000
GPIO_INTSTSC_D	GPIO_BA+0xB8	R/W	PC和PD端口EINT0~3 中断触发源指示	0x0000_0000

6.4.6 寄存器描述

端口 A-D I/O 模式控制 (Px_MODE)

寄存器	偏移地址	R/W	描述	复位值
PA_MODE	GPIO_BA+0x00	R/W	PA I/O模式控制	0x0000_0000
PB_MODE	GPIO_BA+0x10	R/W	PB I/O模式控制	0x0000_0000
PC_MODE	GPIO_BA+0x20	R/W	PC I/O模式控制	0x0000_0000
PD_MODE	GPIO_BA+0x30	R/W	PD I/O模式控制	0x0000_0000

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

位	描述	
[31:16]	Reserved	保留
[n] n = 0,1..15	MODE[n]	**端口 A-D I/O Pin[N] 模式控制** 决定GPIO Px.n的I/O 类型. 0 = Px.n 管脚为输入模式 1 = Px.n 管脚为推挽输出模式 **注:** 端口 A/B： Max. n=15 端口C：Max. n=14 端口D：Max. n=4

端口 A-D 管脚上/下拉电阻控制(Px_PUEN)

寄存器	偏移地址	R/W	描述	复位值
PA_PUEN	GPIO_BA+0x04	R/W	PA I/O 上/下拉电阻控制	0x0000_0000
PB_PUEN	GPIO_BA+0x14	R/W	PB I/O 上/下拉电阻控制	0x0000_0010
PC_PUEN	GPIO_BA+0x24	R/W	PC I/O 上/下拉电阻控制	0x0000_0000
PD_PUEN	GPIO_BA+0x34	R/W	PD I/O 上/下拉电阻控制	0x0000_0000

31	30	29	28	27	26	25	24
PULLSEL15		PULLSEL14		PULLSEL3		PULLSEL2	
23	22	21	20	19	18	17	16
PULLSEL11		PULLSEL10		PULLSEL9		PULLSEL8	
15	14	13	12	11	10	9	8
PULLSEL7		PULLSEL6		PULLSEL5		PULLSEL4	
7	6	5	4	3	2	1	0
PULLSEL3		PULLSEL2		PULLSEL1		PULLSEL0	

位	描述	
[2n+1:2n] n = 0,1..15	**PULLSELn**	**端口 A-D 管脚上拉或下拉电阻使能控制** 00 =禁用 Px.n管脚上拉和下拉电阻 01 =使能管脚上拉电阻 10 =使能管脚下拉电阻 11 = 保留 **注1:** 端口 A/B： Max. n=15 端口C：Max. n=14 端口D：Max. n=4 **注2:** 参考 图 6.4-1

端口 A-D 数据输出值(Px_DOUT)

寄存器	偏移地址	R/W	描述	复位值
PA_DOUT	GPIO_BA+0x08	R/W	PA数据输出值	0x0000_0000
PB_DOUT	GPIO_BA+0x18	R/W	PB数据输出值	0x0000_0000
PC_DOUT	GPIO_BA+0x28	R/W	PC数据输出值	0x0000_0000
PD_DOUT	GPIO_BA+0x38	R/W	PD 数据输出值	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**
DOUT[n]							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DOUT[n]							

位	描述	
[31:16]	**Reserved**	保留
[n] n = 0,1..15	**DOUT[n]**	**端口 A-D Pin[N] 输出值** 在GPIO配置成推挽输出时，控制GPIO相应管脚的状态. 0 = GPIO配置成推挽输出时，Px.n为低 1 = GPIO配置成推挽输出时，Px.n为高 **注:** 端口 A/B： Max. n=15 端口C：Max. n=14 端口D：Max. n=4

端口 A-D 管脚数值(Px _PIN)

寄存器	偏移地址	R/W	描述	复位值
PA_PIN	GPIO_BA+0x0C	R	PA 管脚数值	0x0000_XXXX
PB_PIN	GPIO_BA+0x1C	R	PB 管脚数值	0x0000_XXXX
PC_PIN	GPIO_BA+0x2C	R	PC 管脚数值	0x0000_XXXX
PD_PIN	GPIO_BA+0x3C	R	PD 管脚数值	0x0000_00XX

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

位	描述	
[31:16]	**Reserved**	保留
[n] n = 0,1..15	**PIN[n]**	**端口 A-D Pin[N] 管脚数值** 这些位的值为各个GPIO Px.n管脚真实状态的反映。如果值为‘1’，表示相应管脚状态为高，否则为低 **注:** 端口 A/B： Max. n=15 端口C：Max. n=14 端口D：Max. n=4

中断事件 (EINT) 防抖动控制(GPIO_DBCTL)

寄存器	偏移地址	R/W	描述	复位值
GPIO_DBCTL	GPIO_BA+0x70	R/W	中断事件 (EINT) 防抖动控制	0x0000_0000

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

位	描述	
[31:8]	Reserved	保留
[7:4]	DBCLKSEL	**防抖动采样周期选择** 0000 =采样中断输入每1 APB clocks一次 0001 =采样中断输入每2 APB clocks一次 0010 =采样中断输入每4 APB clocks一次 0011 =采样中断输入每8 APB clocks一次 0100 =采样中断输入每16 APB clocks一次 0101 =采样中断输入每32 APB clocks一次 0110 =采样中断输入每64 APB clocks一次 0111 =采样中断输入每128 APB clocks一次 1000 =采样中断输入每256 APB clocks一次 1001 =采样中断输入每512 APB clocks一次 1010 =采样中断输入每1024 APB clocks一次 1011 =采样中断输入每2048 APB clocks一次 1100 =采样中断输入每4096 APB clocks一次 1101 =采样中断输入每8192 APB clocks一次 1110 =采样中断输入每16384 APB clocks一次 1111 =采样中断输入每32768 APB clocks一次
[n] n = 0,1,2,3	DBEN[n]	**EINT (EINT0~EINT3) 防抖动使能位** DBEN[n]用于使能相应EINT的防抖动功能．如果输入信号脉冲宽度不能被两个连续的防抖动采样周期所采样，则被视为信号反弹，从而不触发中断．防抖动周期由DBCLKSEL (GPIO_DBCTL [7:4])控制。 0 =禁用EINTn防抖动功能 1 = 使能EINTn防抖动功能

端口 A-D中断事件 (EINT) 源分组(Px_INTSRCGP)

寄存器	偏移地址	R/W	描述	复位值
PA_INTSRCGP	GPIO_BA+0x80	R/W	PA中断事件 (EINT) 源分组	0x0000_0000
PB_INTSRCGP	GPIO_BA+0x84	R/W	PB中断事件 (EINT) 源分组	0x5555_5555
PC_INTSRCGP	GPIO_BA+0x88	R/W	PC中断事件 (EINT) 源分组	0x2AAA_AAAA
PD_INTSRCGP	GPIO_BA+0x8C	R/W	PD中断事件 (EINT) 源分组	0x0000_03FF

31	30	29	28	27	26	25	24
INTSEL15		INTSEL14		INTSEL13		INTSEL12	
23	22	21	20	19	18	17	16
INTSEL11		INTSEL10		INTSEL9		INTSEL8	
15	14	13	12	11	10	9	8
INTSEL7		INTSEL6		INTSEL5		INTSEL4	
7	6	5	4	3	2	1	0
INTSEL3		INTSEL2		INTSEL1		INTSEL0	

位	描述	
[2n+1:2n] n=0,1..15	**INTSELn**	**选择Px.N作为中断源到EINT0, EINT1, EINT2, 或 EINT3** 00 =选择Px.N作为一个中断源到EINT0 01 =选择Px.N作为一个中断源到EINT1 10 =选择Px.N作为一个中断源到EINT2 11 =选择Px.N作为一个中断源到EINT3 **注:** 端口 A/B： Max. n=15 端口C：Max. n=14 端口D：Max. n=4

端口 A-D 中断使能控制寄存器(Px_INTEN)

寄存器	偏移地址	R/W	描述	复位值
PA_INTEN	GPIO_BA+0x90	R/W	PA 中断使能控制寄存器	0x0000_0000
PB_INTEN	GPIO_BA+0x94	R/W	PB 中断使能控制寄存器	0x0000_0000
PC_INTEN	GPIO_BA+0x98	R/W	PC 中断使能控制寄存器	0x0000_0000
PD_INTEN	GPIO_BA+0x9C	R/W	PD 中断使能控制寄存器	0x0000_0000

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

位	描述	
[n+16] n = 0,1..15	**REIEN[n]**	**端口A-D Pin[N] 输入上升沿中断使能位** RHIEN (Px_INTEN[n+16])用于使能相应GPIO Px.n输入的中断. 置'1'也可以使能管脚唤醒功能 当设置RHIEN (Px_INTEN[n+16])位为'1': 输入Px.n的状态由低电平到高电平变化时，产生中断. 0 = 禁用Px.n由低电平到高电平变化的中断 1 = 使能Px.n由低电平到高电平变化的中断 **注:** 端口 A/B： Max. n=15 端口C：Max. n=14 端口D：Max. n=4
[n] n = 0,1..15	**FEIEN[n]**	**端口 A-D Pin[N] 输入下降沿中断使能位** FLIEN (Px_INTEN[n])用于使能相应GPIO Px.n]输入的中断. 置 '1' 也可以使能管脚唤醒功能 当设置FLIEN (Px_INTEN[n])位为 '1': 输入Px.n的状态由高电平到低电平变化时，产生中断. 0 = 禁用Px.n由高电平到低电平变化的中断 1 = 使能Px.n由高电平到低电平变化的中断 **注:**

		端口 A/B： Max. n=15 端口C：Max. n=14 端口D：Max. n=4

注1: 只支持边沿触发中断

注2: 在正常工作模式，对于每个引脚FEIENn和REIENn可同时打开去检测上升沿和下降沿

注3:当引脚作为掉电唤醒源，边缘设置必须为电平触发。例如，如果一个引脚设置为上升沿，用户必须保持这个引脚为低后再进入掉电；高电平将使掉电被忽略。进入掉电后，在这个引脚上的高电平将使芯片离开掉电。

注4: 当引脚作为掉电唤醒源，如果双边沿被设置，高优先级将作为唤醒源。

中断锁存触发选择寄存器(GPIO_INTCTL)

寄存器	偏移地址	R/W	描述	复位值
GPIO_INTCTL	GPIO_BA+0xA0	R/W	中断锁存触发选择寄存器	0x0000_0000

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

位	描述	
[31:9]	Reserved	保留
[8]	INTCTL	**中断请求源控制** 0 =中断发生，GPIO中断控制器会产生1APB时钟脉冲到NVIC 1 = 当发生GPIO中断，NVIC的GPIO中断一直保持到CPU清除这些中断触发(GPIO_INTSTSA_B, GPIO_INTSTSC_D)
[n+4] n = 0,1,2,3	WKENn	**GPIO中断唤醒系统使能控制位** 0 = 无作用 1 = EINTn可以将芯片从空闲模式和掉电模式下唤醒
[n] n = 0,1,2,3	INTLHENn	**当EINTn发生时使能锁存PA.N/PB.N/PC.N/PD.N的值** 0 =无作用 1 =当EINTn中断发生，PA_LATCHDAT, PB_LATCHDAT, PC_LATCHDAT, 和 PD_LATCHDAT 寄存器将会锁存PA, PB, PC, 和PD端口值到EINTn组里

端口 A-D中断锁存值(Px_LATCHDAT)

寄存器	偏移地址	R/W	描述	复位值
PA_LATCHDAT	GPIO_BA+0xA4	R	PA 中断锁存值	0x0000_0000
PB_LATCHDAT	GPIO_BA+0xA8	R	PB 中断锁存值	0x0000_0000
PC_LATCHDAT	GPIO_BA+0xAC	R	PC 中断锁存值	0x0000_0000
PD_LATCHDAT	GPIO_BA+0xB0	R	PD 中断锁存值	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**
DATn							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATn							

位	描述	
[31:16]	**Reserved**	保留
[n] n = 0,1..15	**DATn**	**锁存 Px.N 中断值** 当 EINT (EINT0~EINT3)被GPIO_INTCTL激活，将会锁存对应 Px.n的值 **注:** 端口 A/B： Max. n=15 端口C：Max. n=14 端口D：Max. n=4

当被锁存引脚值为0时，意味着：

1. 引脚为输入模式并且为低电平
2. 引脚为输出模式故输入的值被屏蔽为 0

PA和PB端口 EINT0~3 中断触发源指示(GPIO_INTSTSA_B)

寄存器	偏移地址	R/W	描述	复位值
GPIO_INTSTSA_B	GPIO_BA+0xB4	R/W	PA和PB端口 EINT0~3 中断触发源指示	0x0000_0000

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

位	描述	
[n+16] n = 0,1..15	PBIFn	**端口 B Pin[N] 中断触发源标志** 写 : 0= 无动作 1= 清相应的中断标志 读 : 0 = PB.n没有中断 1 = PB.n产生中断
[n] n = 0,1..15	PAIFn	**端口 A Pin[N] 中断触发源标志** 写 : 0= 无动作 1= 清相应的中断标志 读 : 0 = PA.n没有中断 1 = PA.n产生中断

当相应的上升沿或下降沿的触发被设置并且引脚的状态切换确认（通过防抖动或没有防抖动）触发源将被锁定，无论其来源是一个输入或输出引脚。

PC和PD端口EINT0~3 中断触发源指示(GPIO_INTSTSC_D)

寄存器	偏移地址	R/W	描述	复位值
GPIO_INTSTSC_D	GPIO_BA+0xB8	R/W	PC和PD端口EINT0~3 中断触发源指示	0x0000_0000

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

位	描述	
[31:21]	**Reserved**	保留.
[n+16] n = 0,1..4	**PDIFn**	**端口 D Pin[N] 中断触发源标志** 写 : 0= 无动作 1= 清相应的中断标志 读 : 0 = PD.n没有中断 1 = PD.n产生中断
[15]	**Reserved**	保留.
[n] n = 0,1..14	**PCIFn**	**端口 C Pin[N] 中断触发源标志** 写 : 0= 无动作 1= 清相应的中断标志 读 : 0 = PC.n没有中断 1 = PC.n产生中断

当相应的上升沿或下降沿的触发被设置并且引脚的状态切换确认（通过防抖动或没有防抖动）触发源将被锁定，无论其来源是一个输入或输出引脚。

6.5 定时器控制器 (TIMER)

6.5.1 概述

定时器控制器有4个 32-位定时器, TIMER0 ~ TIMER3, 提供用户便捷的计数定时功能。定时器可执行很多功能，如频率测量，时间延迟，时钟发生，外部输入管脚事件计数和外部捕捉管脚脉宽测量等。.

6.5.2 特性

- 4 个 32-位定时器，带24位向上计数器和一个8位的预分频计数器
- 每个定时器都有独立的时钟源
- 提供 单次, 周期, 切换 和 连续 四种计数操作模式
- 溢出周期 = (定时器时钟周期) * (8-位分频器值+1) * CMPDAT (TIMERx_CMP[23:0])
- 最大计数周期 = (1 / T MHz) * (2^8) * (2^{24}), T 是定时器时钟周期
- 通过TIMERx_CNT (定时器数据寄存器)可读取内部 24 位向上计数器的值
- 支持外部管脚(TM0_CNT_OUT~TM3_CNT_OUT)事件计数功能
- 支持外部管脚(TM0_EXT~TM3_EXT)捕捉，可用于脉宽测量
- 支持外部管脚(TM0_EXT~TM3_EXT)捕捉，可用于复位24位向上计数器
- 如果定时器中断信号产生，支持芯片从空闲模式,掉电模式和深度掉电唤醒

6.5.3 框图

定时器控制器框图和时钟控制描述如下.

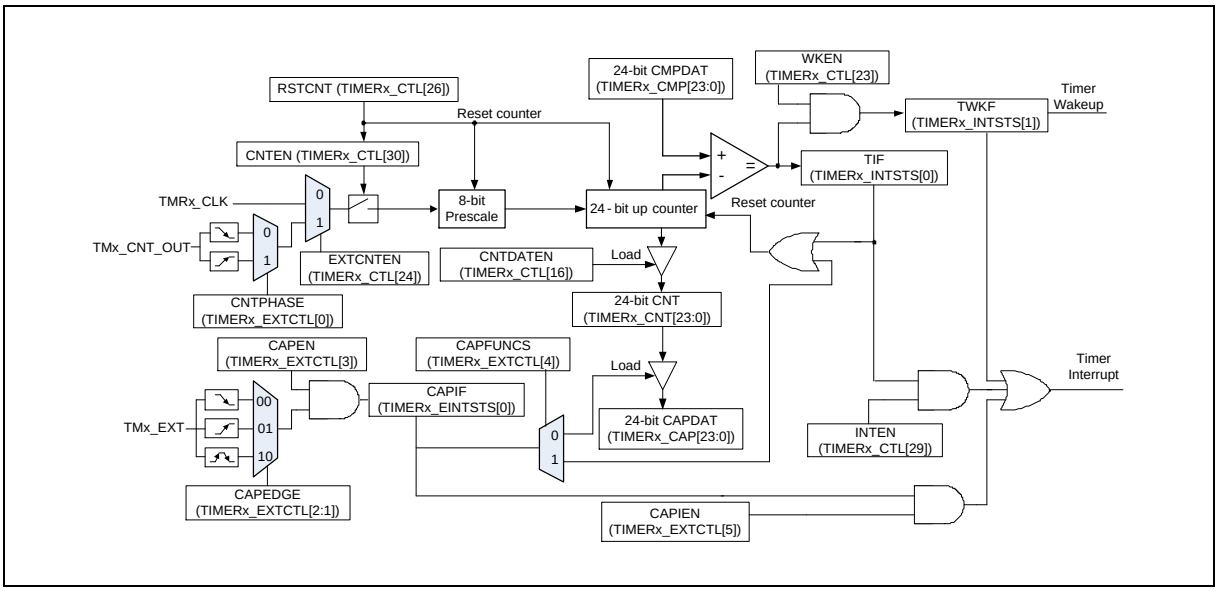


图 6.5-1 定时器控制器框图

6.5.4 功能描述

定时器控制器支持 单次, 周期, 切换 和 连续计数操作模式. 事件计数功能是用于对外部管脚进行事件/计数 ， 外部管脚捕捉功能进行间隔测量或复位计数。每个功能操作模式具体如下描述。

6.5.4.1 定时器中断标志

定时器控制器支持2个中断标志; 一个是 TIF 标志来指示当定时器计数器(TIMERx_CNT)值匹配定时器比较器(TIMERx_CMP)值 ， 另一个是CAPIF (TIMERx_EINTSTS[0]) 标志来指示当 外部管脚 TMx_EXT 变化关联的CAPEDGE (TIMERx_EXTCTL[2:1])设置.

6.5.4.2 单次模式

如果定时器控制器配置在单次模式(TIMERx_CTL[28:27] =00) 和 CNTEN (TIMERx_CTL[30]) 位置1, 定时器计数器开始上计数. 当TIMERx_CNT的值到 TIMERx_CMP 的值, TIF 标志将会置1 , TIMERx_CNT的值和CNTEN 位被定时器控制器清除后计数停止计数. 除此之外,如果INTEN (TIMERx_CTL[29])位使能,定时器中断信号发生并发给NVIC告知CPU。

在这个操作模式下, 当TIF (TIMERx_INTSTS[0])已置 1, 计数器停止并且计数器 (TIMERx_CNT 值)的值和CNTEN (TIMERx_CTL[30])被定时器控制器自动清0. 也就是说, 定时器操作计数器和比较器 TIMERx_CMP 值在程序设置定时器比较寄存器(TIMERx_CMP)的值和CNTEN (TIMERx_CTL[30])置1后仅执行一次。因此，这种操作模式叫“单次模式”。

6.5.4.3 *周期模式*

如果定时器在周期模式 (TIMERx_CTL[28:27] = 01) 和 CNTEN (TIMERx_CTL[30])位置1, 定时器计数器开始上计数. 一旦TIMERx_CNT值达到 CMPDAT 值, TIF (TIMERx_INTSTS[0]) 将置1. 如果INTEN (TIMERx_CTL[29]) 置 1, TIF (TIMERx_INTSTS[0])置1然后中断信号发生并发给NVIC 通知CPU 表明定时器计数溢出发生。如果INTEN (TIMERx_CTL[29])清0,就没有中断信号发生.

在这个模式下, 一旦TIMERx_CNT值达到 CMPDAT 值, TIF (TIMERx_INTSTS[0]) 将置1， 计数器的值 (TIMERx_CNT value) 回初始值和 CNTEN (TIMERx_CTL[30] 使能位)维持为 1 (计数器继续计数) 定时器计数器仍上计数。如果 TIF (TIMERx_INTSTS[0]) 被软件清位, 一旦TIMERx_CNT值再达到 CMPDAT 值, TIF (TIMERx_INTSTS[0]) 将再置1. 也就是说,定时器计数器和比较器TIMERx_CMP周期性操作. 定时器计数器不停止直到 CNTEN (TIMERx_CTL[30])置0. 中断信号也周期性发生。因此，这种操作模式叫“周期模式”。

6.5.4.4 *切换模式*

如果定时器工作在切换输出模式 (TIMERx_CTL[28:27] =10) 和 CNTEN (TIMERx_CTL[30]) 位置1, 定时器计数器开始上计数. 一旦TIMERx_CNT值达到 CMPDAT 值, TIF (TIMERx_INTSTS[0]) 将置1. 如果INTEN (TIMERx_CTL[29]) 置1, TIF (TIMERx_INTSTS[0])置1然后中断信号发生并发给NVIC 通知CPU 表明定时器计数溢出发生. 如果INTEN (TIMERx_CTL[29])清0,就没有中断信号发生.

在这个模式下, 一旦TIMERx_CNT值达到 CMPDAT 值, TIF (TIMERx_INTSTS[0]) 将置1，切换输出信号 (TM0_CNT_OUT~TM3_CNT_OUT 管脚) 将置1, 计数器的值 (TIMERx_CNT value) 回初始值和 CNTEN (TIMERx_CTL[30] 使能位)维持为 1 (计数器继续计数) 定时器计数器仍上计数，如果 TIF (TIMERx_INTSTS[0]) 被软件清位, 一旦TIMERx_CNT值再达到 CMPDAT 值, TIF (TIMERx_INTSTS[0]) 将再置1.切换输出(TM0_CNT_OUT~TM3_CNT_OUT 管脚) 将被置 0. 定时器计数器不停止直到 CNTEN (TIMERx_CTL[30]) 置 0. 因此, 切换输出信号 (TM0_CNT_OUT~TM3_CNT_OUT 管脚)反复变化形成50% 占空比信号. 因此,这种操作模式叫“切换输出模式”。

6.5.4.5 *连续计数模式*

如果定时器工作在连续计数模式 (TIMERx_CTL[28:27] = 11) 和 CNTEN (TIMERx_CTL[30]) 位置1, 定时器计数器开始上计数. 一旦TIMERx_CNT值达到 CMPDAT 值, TIF (TIMERx_INTSTS[0]) 将置1. 如果INTEN (TIMERx_CTL[29]) 置1, TIF (TIMERx_INTSTS[0])置1然后中断信号发生并发给NVIC 通知CPU 表明定时器计数溢出发生. 如果INTEN (TIMERx_CTL[29])清0,就没有中断信号发生.

在这个模式下, 一旦TIMERx_CNT值达到 CMPDAT 值, TIF (TIMERx_INTSTS[0]) 将置1 和 CNTEN (TIMERx_CTL[30])维持为 1定时器计数器连续计数不会重载计数器值 (TIMERx_CNT 值)直到初始值. 用户能改变不同的比较寄存器 (TIMERx_CMP)的值但不能立即停止和复位计数器计数。

例, TIMERx_CMP 值先设为 80,. 一旦TIMERx_CNT值到 80, TIF 将置 1 并 CNTEN (TIMERx_CTL[30]) 继续为1 和 TIMERx_CNT 值不回到0,计数器继续计数 81, 82, 83,⋯ 到 2^{24} -1, 0, 1, 2, 3, ⋯ 到 2^{24} -1一直继续. 接着, 如果用户改变TIMERx_CMP 值为 200 并且TIF (TIMERx_INTSTS[0]) 清除为 0, 当TIMERx_CNT的值到达200然后 TIF (TIMERx_INTSTS[0]) 再被置1. 最后用户设置 TIMERx_CMP 值为500并清除TIF (TIMERx_INTSTS[0]) 为 0, 当TIMERx_CNT的值到达500然后 TIF (TIMERx_INTSTS[0]) 再被置1. 这种模式, TIMERx_CNT 值一直保持上计数即使TIF (TIMERx_INTSTS[0]) 置 1. 因此, 这种操作模式叫“连续计数模式”。

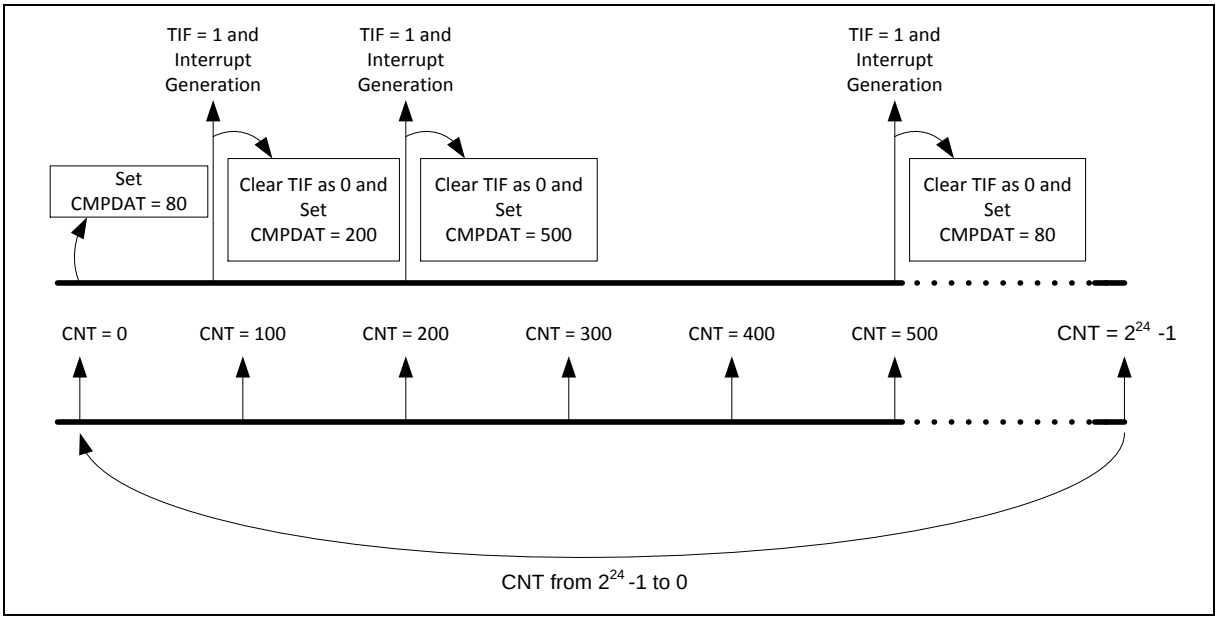


图 6.5-2 连续计数模式

6.5.4.6 *事件计数模式*

定时器控制器提供一种模式能对TMx_CNT_OUT (x= 0~3)管脚输入事件进行计数，事件次数则表现在 TIMERx_CNT的值. 这就叫事件计数功能. 在这个功能中, EXTCNTEN (TIMERx_CTL[24]) 位将被置位， 定时器时钟源可以设为HXT.

软件能使能和关闭CNTDBEN (TIMERx_EXTCTL[7])位来控制 TMx_CNT_OUT管脚防抖动电路. 如果 TMx_CNT_OUT (x= 0~3) 管脚防抖动电路关闭，输入事件的频率不能小于1/3 HXT。如果TMx_CNT_OUT (x= 0~3) 管脚防抖动电路使能，输入事件的频率小于1/8 HXT就不能保证TIMERx_CNT值正确, 软件可以设置CNTPHASE (TIMERx_EXTCTL[0]) 位来选择TMx_CNT_OUT管脚的边沿脉冲检测。

在事件计数模式, 定时器的计数器可以选择单次, 周期和连续计数模式去计数 TIMERx_CNT 值用来计数TMx_CNT_OUT (x= 0~3) 管脚.

6.5.4.7 *外部捕捉模式*

当 TMx_EXT (x= 0~3)管脚输入边沿变化时，外部捕捉功能是用来捕捉TIMERx_CNT值到TIMERx_CAP。在这种模式, CAPFUNCS (TIMERx_EXTCTL[4]) 位置0选择 TMx_EXT 变化用作外部捕捉功能和定时器的时钟源将设置为HXT.

软件能使能和关闭CAPDBEN (TIMERx_EXTCTL[6])位来控制TMx_EXT管脚防抖动电路。如果TMx_EXT管脚防抖动电路关闭，输入事件的频率不能小于1/3 HXT 。如果TMx_EXT管脚防抖动电路使能，输入事件的频率小于1/8保证捕捉功能工作正常。软件可以设置CAPEDGE (TIMERx_EXTCTL[2:1])位来选择TMx_EXT 管脚的边沿变化检测。

在外部捕捉模式, 软件不用考虑定时器计数器选择的模式，只要在TMx_EXT管脚检测到边沿变化捕捉事件就发生。

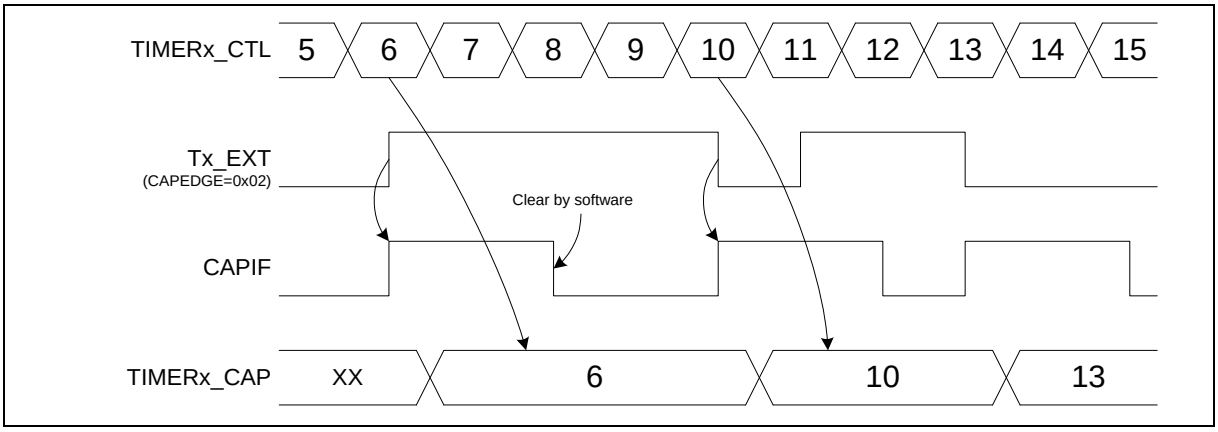


图 6.5-3 外部捕捉模式

6.5.4.8 外部复位计数模式

定时器控制器提供复位计数功能，当输入管脚TX_EXT (x= 0~3)边沿发生变化时CNT (TIMERx_CNT[23:0]) 值复位。在这个模式下，大部分设置与事件捕捉模式相同，除 CAPFUNCS (TIMERx_EXTCTL[4]) 应该设为 1 选择 TX_EXT变化时去触发复位计数器的值。

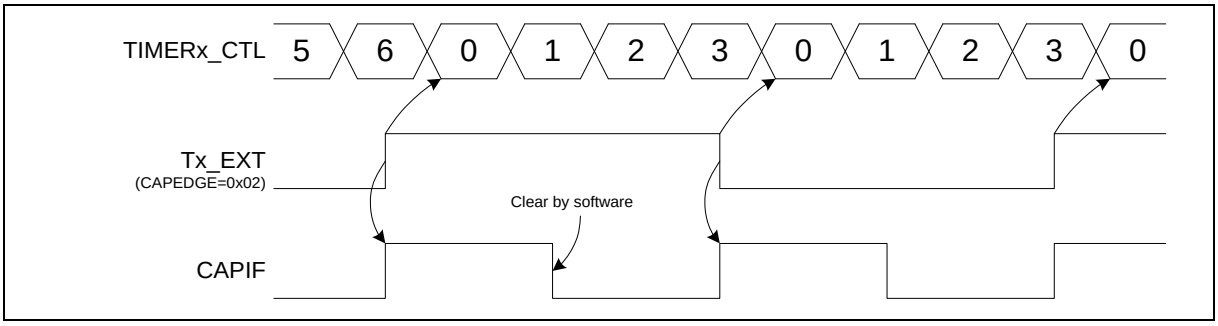


图 6.5-4 外部复位计数模式

6.5.5 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读写,

寄存器	偏移地址	R/W	描述	复位值
TIMER 基地址: **TIMER_BA01 = 0x400E_A000** **TIMER_BA23 = 0x400E_B000**				
TIMER0_CTL	TIMER_BA01+0x00	R/W	Timer0 控制和状态寄存器	0x0000_0005
TIMER0_CMP	TIMER_BA01+0x04	R/W	Timer0比较寄存器	0x0000_0000
TIMER0_INTSTS	TIMER_BA01+0x08	R/W	Timer0 中断状态寄存器	0x0000_0000
TIMER0_CNT	TIMER_BA01+0x0C	R	Timer0数据寄存器	0x0000_0000
TIMER0_CAP	TIMER_BA01+0x10	R	Timer0 捕捉数据寄存器	0x0000_0000
TIMER0_EXTCTL	TIMER_BA01+0x14	R/W	Timer0外部控制寄存器	0x0000_0000
TIMER0_EINTSTS	TIMER_BA01+0x18	R/W	Timer0外部中断状态寄存器	0x0000_0000
TIMER1_CTL	TIMER_BA01+0x20	R/W	Timer1控制和状态寄存器	0x0000_0005
TIMER1_CMP	TIMER_BA01+0x24	R/W	Timer1比较寄存器	0x0000_0000
TIMER1_INTSTS	TIMER_BA01+0x28	R/W	Timer1中断状态寄存器	0x0000_0000
TIMER1_CNT	TIMER_BA01+0x2C	R	Timer1数据寄存器	0x0000_0000
TIMER1_CAP	TIMER_BA01+0x30	R	Timer1捕捉数据寄存器	0x0000_0000
TIMER1_EXTCTL	TIMER_BA01+0x34	R/W	Timer1外部控制寄存器	0x0000_0000
TIMER1_EINTSTS	TIMER_BA01+0x38	R/W	Timer1外部中断状态寄存器	0x0000_0000
TIMER2_CTL	TIMER_BA23+0x00	R/W	Timer2控制和状态寄存器	0x0000_0005
TIMER2_CMP	TIMER_BA23+0x04	R/W	Timer2比较寄存器	0x0000_0000
TIMER2_INTSTS	TIMER_BA23+0x08	R/W	Timer2中断状态寄存器	0x0000_0000
TIMER2_CNT	TIMER_BA23+0x0C	R	Timer2数据寄存器	0x0000_0000
TIMER2_CAP	TIMER_BA23+0x10	R	Timer2捕捉数据寄存器	0x0000_0000
TIMER2_EXTCTL	TIMER_BA23+0x14	R/W	Timer2外部控制寄存器	0x0000_0000
TIMER2_EINTSTS	TIMER_BA23+0x18	R/W	Timer2外部中断状态寄存器	0x0000_0000
TIMER3_CTL	TIMER_BA23+0x20	R/W	Timer3控制和状态寄存器	0x0000_0005
TIMER3_CMP	TIMER_BA23+0x24	R/W	Timer3比较寄存器	0x0000_0000
TIMER3_INTSTS	TIMER_BA23+0x28	R/W	Timer3中断状态寄存器	0x0000_0000

TIMER3_CNT	TIMER_BA23+0x2C	R	Timer3数据寄存器	0x0000_0000
TIMER3_CAP	TIMER_BA23+0x30	R	Timer3捕捉数据寄存器	0x0000_0000
TIMER3_EXTCTL	TIMER_BA23+0x34	R/W	Timer3外部控制寄存器	0x0000_0000
TIMER3_EINTSTS	TIMER_BA23+0x38	R/W	Timer3外部中断状态寄存器	0x0000_0000

6.5.6 寄存器描述

定时器控制和状态寄存器(TIMERx_CTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CTL	TIMER_BA01+0x00	R/W	Timer0控制和状态寄存器	0x0000_0005
TIMER1_CTL	TIMER_BA01+0x20	R/W	Timer1控制和状态寄存器	0x0000_0005
TIMER2_CTL	TIMER_BA23+0x00	R/W	Timer2控制和状态寄存器	0x0000_0005
TIMER3_CTL	TIMER_BA23+0x20	R/W	Timer3控制和状态寄存器	0x0000_0005

31	30	29	28	27	26	25	24
ICEDEBUG	CNTEN	INTEN	OPMODE		RSTCNT	ACTSTS	EXTCNTEN
23	22	21	20	19	18	17	16
WKEN	Reserved						CNTDATEN
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
PSC							

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

		00 =定时器控制器工作在 单次计数模式. 01 =定时器控制器工作在周期模式 10 =定时器控制器工作在切换输出模式 11 =定时器控制器工作在连续计数模式.
[26]	**RSTCNT**	**定时器复位控制** 设置该位将复位24位向上计数器(TIMERx_CNT)的值，且如果ACTSTS (TIMERx_CTL[25])位为1，则会把CNTEN (TIMERx_CTL[30])强制清0. 0 = 无效 1 =复位8-位预分频计数器，内部24-位向上计数器的值和CNTEN位
[25]	**ACTSTS**	**定时器激活状态位（只读）** 该位表示24位向上计数器的状态。 0 = 定时器计数器未激活 1 = 定时器计数器激活
[24]	**EXTCNTEN**	**外部计数模式使能位** 该位用来使能外部计数管脚功能，当定时器用于事件计数，该位置1并选择HXT作为定时器时钟源。 0 = 禁用事件计数器模式 1 = 使能事件计数器模式
[23]	**WKEN**	**唤醒使能控制** 如果该位1, 当中断标志 TIF (TIMERx_INTSTS[0])是 1 和 INTEN (TIMERx_CTL[29])使能, 定时器中断信号将唤醒触发CPU. 0 = 如果中断信号发生唤醒触发事件禁止 1 =如果中断信号发生唤醒触发事件使能.
[22:17]	**Reserved**	保留.
[16]	**CNTDATEN**	**数据加载使能** 当该位置1，定时器计数器值(TIMERx_CNT)将连续不断地更新到监控内部24位上计数器的值. 0 = 数据寄存器加载禁止. 1 = 当定时器计数运行时数据寄存器加载使能
[15:8]	**Reserved**	保留.
[7:0]	**PSC**	**预分频计数器** 定时器的输入时钟源或事件计数源被PSC +1预分频，然后再输入到定时器。如果 该位为0，表示不分频。

定时器比较寄存器(TIMERx_CMP)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CMP	TIMER_BA01+0x04	R/W	Timer0比较寄存器	0x0000_0000
TIMER1_CMP	TIMER_BA01+0x24	R/W	Timer1比较寄存器	0x0000_0000
TIMER2_CMP	TIMER_BA23+0x04	R/W	Timer2比较寄存器	0x0000_0000
TIMER3_CMP	TIMER_BA23+0x24	R/W	Timer3比较寄存器	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留.
[23:0]	**CMPDAT**	**定时器比较值** CMPDAT是24-位比较寄存器。当内部 24-位向上计数器的值等于CMPDAT的值时，TIF (TIMERx_INTSTS[0] 定时器中断标志)将被置1。 超时溢出周期 = (定时器输入时钟周期) * (8-bit PSC + 1) * (24-bit CMPDAT) 注1：不能向CMPDAT写 0x0 或 0x1，否则内核将运行到未知状态。 注2：当定时器工作在连续计数模式，即使软件写一个新的值到CMPDAT，24-位向上计数定时器将保持继续计数。如果定时器工作在其他模式，如果软件写一个新的值到CMPDAT，定时器将使用新比较值并退出当前计数，开始重新计数。

定时器中断状态寄存器(TIMERx_INTSTS)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_INTSTS	TIMER_BA01+0x08	R/W	Timer0中断状态寄存器	0x0000_0000
TIMER1_INTSTS	TIMER_BA01+0x28	R/W	Timer1中断状态寄存器	0x0000_0000
TIMER2_INTSTS	TIMER_BA23+0x08	R/W	Timer2中断状态寄存器	0x0000_0000
TIMER3_INTSTS	TIMER_BA23+0x28	R/W	Timer3中断状态寄存器	0x0000_0000

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

位	描述	
[31:2]	**Reserved**	保留.
[1]	**TWKF**	**定时器唤醒标志** 该位表示定时器的中断唤醒标志状态。 0 = 定时器不会引起CPU 唤醒 1 =如果定时器中断信号产生， CPU 从掉电或深度掉电模式唤醒 **注:** 该位必须通过软件写1清0
[0]	**TIF**	**定时器中断标志** 当定时器TIMERx_CNT的值到达定时器比较值CMPDAT时，该位会显示其中断状态。. 0 = 无影响 1 = TIMERx_CNT与CMPDAT的值相匹配 **注**：该位写 1 清零。

定时器数据寄存器(TIMERx_CNT)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CNT	TIMER_BA01+0x0C	R	Timer0数据寄存器	0x0000_0000
TIMER1_CNT	TIMER_BA01+0x2C	R	Timer1数据寄存器	0x0000_0000
TIMER2_CNT	TIMER_BA23+0x0C	R	Timer2数据寄存器	0x0000_0000
TIMER3_CNT	TIMER_BA23+0x2C	R	Timer3数据寄存器	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留.
[23:0]	**CNT**	**定时器数据寄存器** 1.如果EXTCNTEN (TIMERx_CTL[24]) =0, 用户可以读CNT 的值来获取当前24位向上计数器的值 2.如果 EXTCNTEN (TIMERx_CTL[24])=1, 用户可以读 CNT 的值来获取当前24位事件计数器的值.

定时器捕捉数据寄存器(TIMERx_CAP)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_CAP	TIMER_BA01+0x10	R	Timer0捕捉数据寄存器	0x0000_0000
TIMER1_CAP	TIMER_BA01+0x30	R	Timer1捕捉数据寄存器	0x0000_0000
TIMER2_CAP	TIMER_BA23+0x10	R	Timer2捕捉数据寄存器	0x0000_0000
TIMER3_CAP	TIMER_BA23+0x30	R	Timer3捕捉数据寄存器	0x0000_0000

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

位	描述	
[31:24]	**Reserved**	保留.
[23:0]	**CAPDAT**	**定时器捕捉数据寄存器** 当 CAPEN (TIMERx_EXTCTL[3]) 位被置1, CAPFUNCS (TIMERx_EXTCTL[4])位为0, Tx_EXT 管脚的变化与CAPEDGE (TIMERx_EXTCTL[2:1]) 设定相匹配时, CAPIF (TIMERx_EINTSTS[0]) 将被置1且当前定时器计数器(TIMERx_CNT 值) 的值将被自动载入到CAPDAT.

定时器外部控制寄存器(TIMERx_EXTCTL)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_EXTCTL	TIMER_BA01+0x14	R/W	Timer0外部控制寄存器	0x0000_0000
TIMER1_EXTCTL	TIMER_BA01+0x34	R/W	Timer1外部控制寄存器	0x0000_0000
TIMER2_EXTCTL	TIMER_BA23+0x14	R/W	Timer2外部控制寄存器	0x0000_0000
TIMER3_EXTCTL	TIMER_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
CNTDBEN	CAPDBEN	CAPIEN	CAPFUNCS	CAPEN	CAPEDGE		CNTPHASE

位	描述	
[31:8]	Reserved	保留.
[7]	CNTDBEN	定时器计数管脚防抖动使能控制 0 = TMx_CNT_OUT (x= 0~3) 管脚防抖动禁止. 1 = TMx_CNT_OUT (x= 0~3) 管脚防抖动使能. 注: 如果该位使能TMx_CNT_OUT管脚的边沿检测将有防抖动电路
[6]	CAPDBEN	定时器捕捉管脚防抖动使能控制 0 = TMx_EXT (x= 0~3) 管脚防抖动禁止 1 = TMx_EXT (x= 0~3) 管脚防抖动使能. 注: 如果该位使能TMx_EXT管脚的边沿检测将有防抖动电路.
[5]	CAPIEN	定时器外部中断管脚防抖动使能控制 0 = TMx_EXT (x= 0~3) 外部中断管脚防抖动禁止. 1 = TMx_EXT (x= 0~3) 外部中断管脚防抖动使能. 注: CAPIEN是用来使能外部中断.如果CAPIEN 使能, 当CAPIF = 1定时器将出现一个中断. 例, 当 CAPIEN = 1, CAPEN = 1, 和 CAPEDGE = 00, TMx_EXT管脚有1到0的变化将引起CAPIF(TIMERx_EINTSTS[0])中断标志置1然后中断信号发生发送给NVIC 告知 CPU.
[4]	CAPFUNCS	定时器外部复位计数/捕捉模式选择 0 = TMx_EXT (x= 0~3)管脚变化保存24-位定时器计数器的值(TIMERx_CNT 值) 到捕捉寄存器 (TIMERx_CAP 值)， CAPIF (TIMERx_EINTSTS[0]) 置 1

		1 = TMx_EXT (x= 0~3) 管脚变化将复位24位定时器计数器的值.
[3]	**CAPEN**	**定时器外部管脚使能控制** 该位使能 CAPFUNCS (TIMERx_EXTCTL[4]) 功能在TMx_EXT 管脚. 0 = CAPFUNCS 功能在TMx_EXT (x= 0~3) 管脚忽略 1 = CAPFUNCS 功能在TMx_EXT (x= 0~3) 管脚有效.
[2:1]	**CAPEDGE**	**外部管脚边沿检测** 00 = 1到 0 的变化在 TMx_EXT (x= 0~3) 管脚将被检测 01 = 0到 1 的变化在 TMx_EXT (x= 0~3) 管脚将被检测. 10 = 1到 0或0到1 的变化在 TMx_EXT (x= 0~3) 管脚将都被检测. 11 = 保留.
[0]	**CNTPHASE**	**定时器外部脉冲计数** 该位指明检测外部计数管脚的脉冲 0 = 一个外部计数脚下降沿将计数 1 =一个外部计数脚上升沿将计数.

定时器外部中断状态寄存器(TIMERx_EINTSTS)

寄存器	偏移地址	R/W	描述	复位值
TIMER0_EINTSTS	TIMER_BA01+0x18	R/W	Timer0外部中断状态寄存器	0x0000_0000
TIMER1_EINTSTS	TIMER_BA01+0x38	R/W	Timer1外部中断状态寄存器	0x0000_0000
TIMER2_EINTSTS	TIMER_BA23+0x18	R/W	Timer2外部中断状态寄存器	0x0000_0000
TIMER3_EINTSTS	TIMER_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							**CAPIF**

位	描述	
[31:1]	**Reserved**	保留.
[0]	**CAPIF**	**定时器外部中断标志** 该位指示定时器外部中断状态. 当 CAPEN (TIMERx_EXTCTL[3]) 位置1, CAPFUNCS (TIMERx_EXTCTL[4]) 位为0, a在TMx_EXT (x= 0~3)管脚变化符合CAPEDGE (TIMERx_EXTCTL[2:1]) 设置要求, 该位被硬件置1. 0 = TMx_EXT (x= 0~3) 管脚中断没有发生. 1 = TMx_EXT (x= 0~3) 管脚中断发生. **注:** 该位写1 清除.

6.6 PWM 发生器和捕捉定时器 (PWM)

6.6.1 概述

NUC505 系列有一路 PWM发生器，支持4通道 PWM 输出或4通道输入捕捉功能，管脚共用(PWM_CH0/ PWM_CH1，PWM_CH2/PWM_CH3)。

PWM 发生器有16位PWM计数器和比较器，PWM发生器支持 2 种标准的PWM输出模式： 独立模式和带8位死区控制的互补模式。每种模式能象定时器和事件计数器一样独立应用。对PWM计数器的时钟频率范围有8位预分频器和5级(1, 1/2, 1/4, 1/8, 1/16)时钟除法器来进行调节。对每个PWM输出控制单元都支持极性反向功能。

PWM发生器也支持输入捕捉功能。当输入通道有上升变化，下降变化或两种变化发生时，锁存PWM计数器的值到相应的寄存器中。

输入捕捉功能使能后，当输入通道有上升变化产生时，锁存PWM计数器的值到RCAPDATn寄存器，当输入通道有下降变化产生时，锁存PWM计数器的值到FCAPDATn 寄存器。捕捉通道0 的中断通过程序设定CRLIEN0 (PWM_CAPCTL01[1]) (上升捕捉中断使能) 和 CFLIEN0 (PWM_CAPCTL01[2]) (下降捕捉中断使能)的值来决定中断触发的条件。捕捉通道1 的中断通过程序设定CRLIEN1 (PWM_CAPCTL01[17]) (上升捕捉中断使能) 和 CFLIEN1 (PWM_CAPCTL01[18]) (下降捕捉中断使能)的值来决定中断触发的条件。捕捉2和3通道也一样要设定CRLIEN2 (PWM_CAPCTL23[1]),CFLIEN2 (PWM_CAPCTL23[2]) 和 CRLIEN3 (PWM_CAPCTL23[17]), CFLIEN3 (PWM_CAPCTL23[18]) 相应的值。每当捕捉通道0/1/2/3产生中断时，PWM计数器0/1/2/3将会对应重置。

PWM发生器仅有4个中断，PWM0和捕捉0共享同一中断，PWM1和捕捉1共享同一中断，其它的也是一样，因此PWM功能和捕捉功能在同一通道上无法同时应用。

6.6.2 特性

6.6.2.1 PWM 功能描述

- 支持 4 通道 16 位分辨率 PWM 输出
- 支持 8-位预分频和时钟除法器
- 支持 4 个 PWM 中断源
- 支持 单次或自动加载 PWM 计数器操作模式
- 支持 8 位死区控制

6.6.2.2 捕捉功能描述

- 支持 4 通道 6 位分辨率输入捕捉
- 支持 上升沿或下降沿捕捉
- 支持 4 个捕捉中断源

6.6.3 框图

图 6.6-1 图 6.6-2 描述了一组PWM 的架构(如 PWM-定时器 0/1 是一组, PWM-定时器 2/3 是一组).

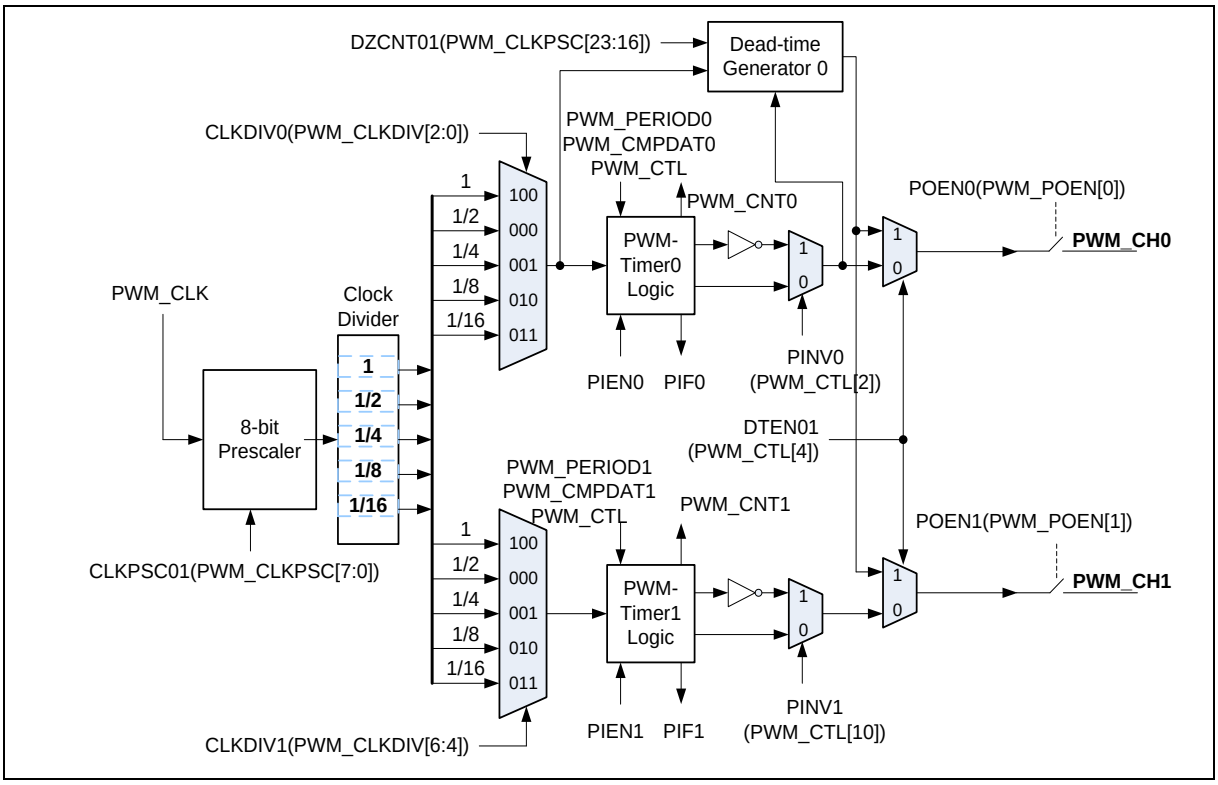


图 6.6-1 PWM 发生器 0 框图

NUC505 技术参考手册

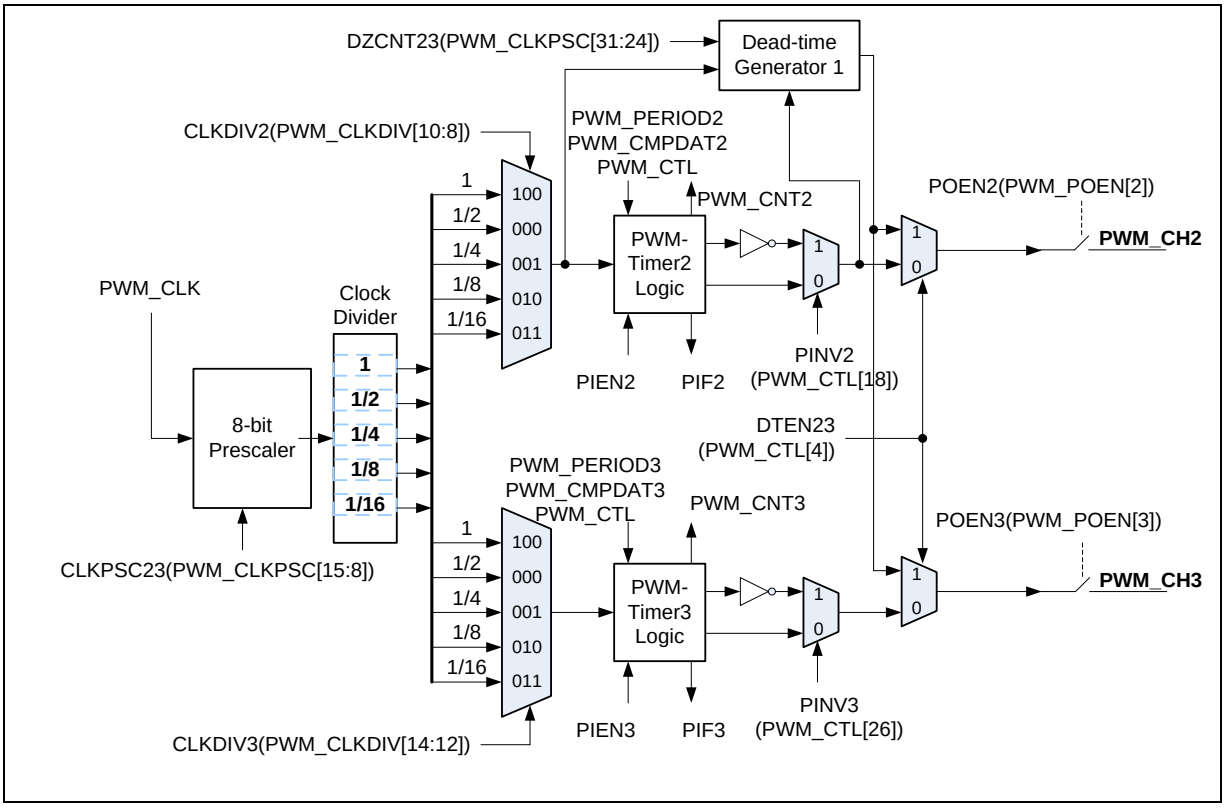


图 6.6-2 PWM发生器1框图

每个 PWM 发生器有2个时钟源输入，每个时钟源都能做PWM的时钟源如图 6.6-3所示。

当系统时钟发生变化时为保持PWM的输出占空比用户可以选择外部晶振(12 MHz)做PWM时钟，最大PWM_CLK频率和PCLK频率相同。

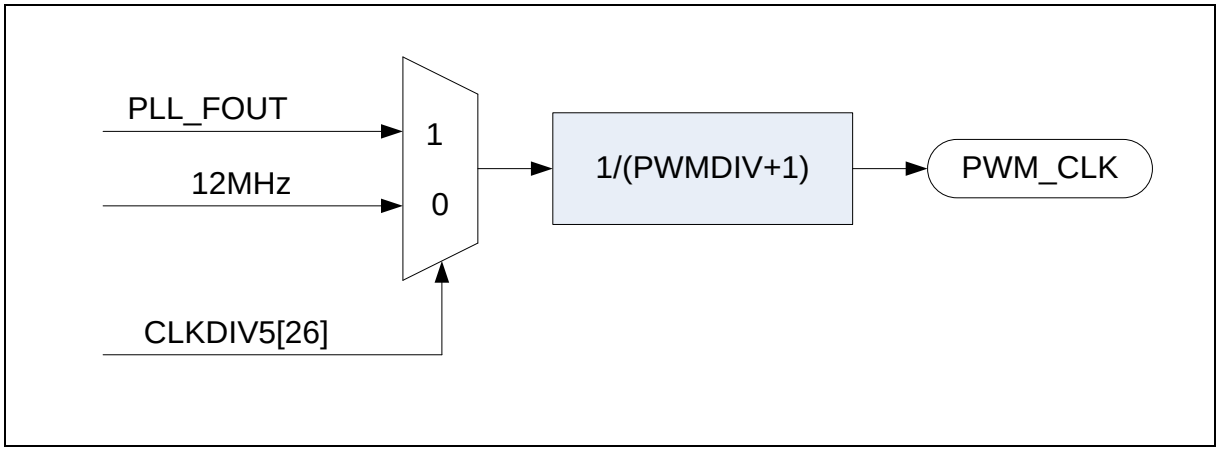


图 6.6-3 PWM 系统时钟源

6.6.4 功能描述

6.6.4.1 边沿对齐方式 (下计数器)

边沿对齐方式， 16 位 PWM 计数器是个下计数计数器，从PWM_PERIODn的值开始到0的下计数完成为PWM的周期。PWM计数器的值和PWM_CMPDATn寄存器的值在比较单元比较，当PWM计数器的值大于PWM_CMPDATn 的值输出低，等于或小于PWM_CMPDATn 的值输出高。因此，通过设置PWM_PERIODn 和PWM_CMPDATn 的值可以调整PWM的周期和占空比。 PWM遵循以下公式和边沿对齐波形图 6.6-4.

- PWM 时钟频率 = PWM_CLK/[(prescale+1)* (clock divider)]
- 占空比 = (PWM_CMPDATn + 1)/(PWM_PERIODn + 1)
- PWM_CMPDATn >= PWM_PERIODn: PWM 输出高电平
- PWM_CMPDATn < PWM_PERIODn: PWM 低电平宽 = (PWM_PERIODn - PWM_CMPDATn) 单位; PWM 高电平宽 = (PWM_CMPDATn + 1) 单位
- PWM_CMPDATn = 0: PWM 低电平宽= (PWM_PERIODn) 单位; PWM高电平宽= 1 单位
- 1 单位 = 1个 PWM 时钟周期

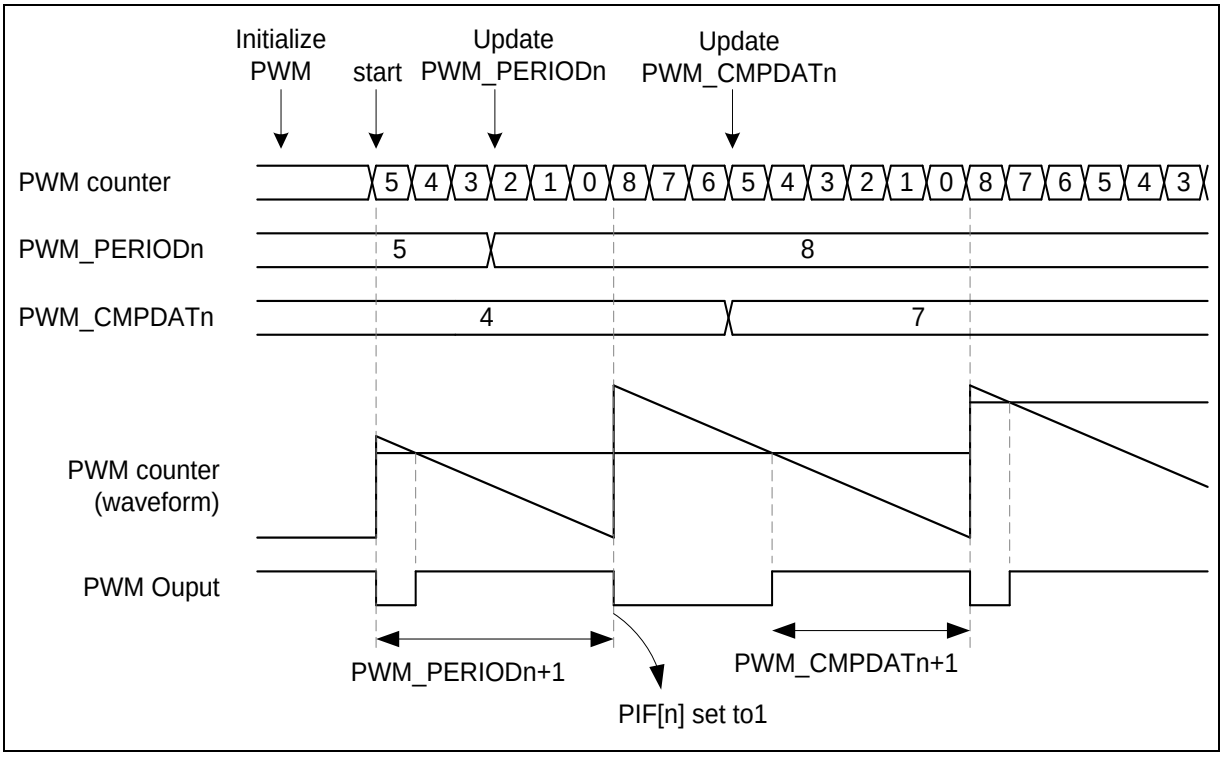


图 6.6-4 PWM 边沿对齐波形图

6.6.4.1 PWM 计数器操作模式

PWM计数器支持2种操作模式： 单次模式和自动加载模式。如果CNTMODE0 (PWM_CTL[3])位设置0时PWM将工作在单次模式，如果CNTMODE0 (PWM_CTL[3])设置1时，PWM工作在自动加载模式。在

设置PWM_PERIODn, PWM_CMPDATn 和使能PWM计数器运行（CNTENn位置1）前，设置PWM的操作模式，因为当改变PWM的操作模式时，寄存器PWM_PERIODn 和PWM_CMPDATn的值将会被清0。

在单次模式, 首先写PWM_CMPDATn 和 PWM_PERIODn 的值 ，然后再给CNTENn位置1去使能PWM计数器运行。在边沿类型下，PWM计数器从PWM_PERIODn的值向下计数到0，PWM计数器将停止，并硬件自动清除PWM_PERIODn 和PWM_CMPDATn 的值为0。

通过软件写到PWM_CMPDATn 和 PWM_PERIODn的新值是设置下一次的的周期和占空比，当下一次开始执行。一般要先写PWM_CMPDATn的值，因为当写一个非0的PERIODn的值时PWM计数器将会自动运行。

在自动加载模式, 首先写PWM_CMPDATn 和 PWM_PERIODn 的值 ，然后再给CNTENn位置1去使能PWM计数器运行。当PWM计数器的值向下计数到0时，将自动加载PWM_PERIODn 的值。如果设置PWM_PERIODn的值为0时，PWM计数器将会停止。.

6.6.4.2 PWM 双缓存

PWM 有双缓存功能，当软件改变更改PWM周期(写 PWM_PERIODn)或占空比写 PWM_CMPDATn)时，PWM周期和占空比将在下个周期加载，对当前周期的输出没有影响。双缓存时序波形如下。

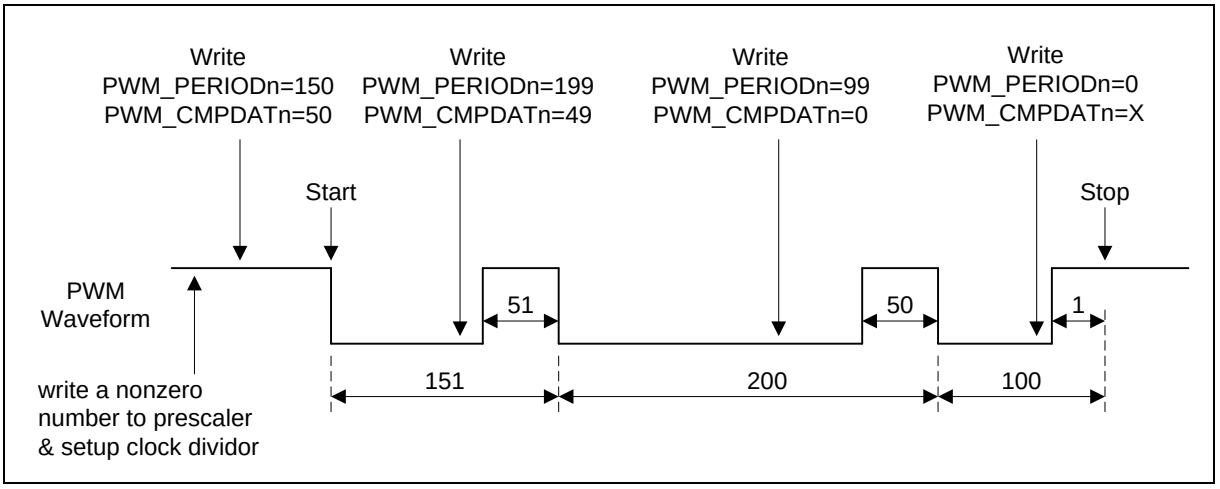


图 6.6-5 PWM 双缓存时序

6.6.4.3 PWM 输出控制单元

PWM输出控制单元包括死区控制，输出控制和极性控制。

6.6.4.3.1 死区控制

在互补输出时当输出脚开另一脚关时，死区发生器插入一个延时“死区”周期“关”到输出开脚。互补输出模式有一个8位的死区计数器，死区延时时间可以到0.

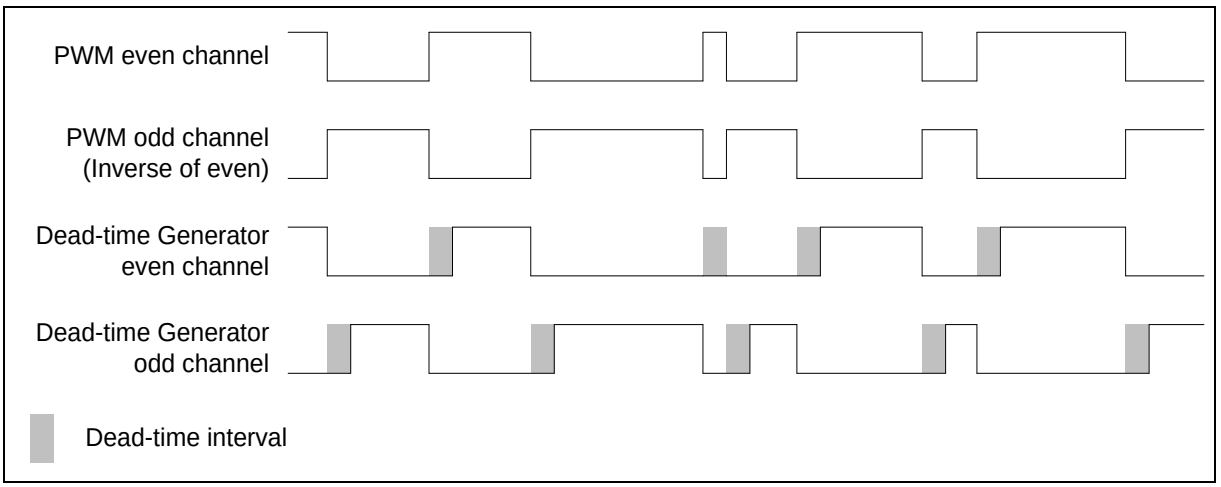


图 6.6-6 PWM 带死区的互补输出

6.6.4.3.2 *极性控制*

每个PWM端口，从PWM_CH0到 PWM_CH3都有一个独立的极性控制器用于配置PWM的输出工作状态的极性。默认的输出为高工作状态，这说明PWM的关状态为低，开状态为高。对每个PWM通道，通过设置PWM_CTL 的PINV 位的值可以改变PWM输出端的极性。

6.6.4.4 *捕捉操作*

捕捉的输入通道和PWM输出共享相同的管脚和计数器。在捕捉功能时，当输入通道有上升变化时，锁存PWM计数器的值到RCAPDATn，当输入通道有下降变化时，锁存PWM计数器的值到FCAPDATn。捕捉中断是靠设置CRLIE[n] (上升锁存中断使能) 和 CFLIE[n] (下降锁存中断使能)来设置中断发生的条件。每当捕捉事件发生触发中断时就会重载PWM_PERIODn的值到PWM计数器。注相应捕捉通道的GPIO管脚必须配置为捕捉功能(CAPINEN[n] 使能)。

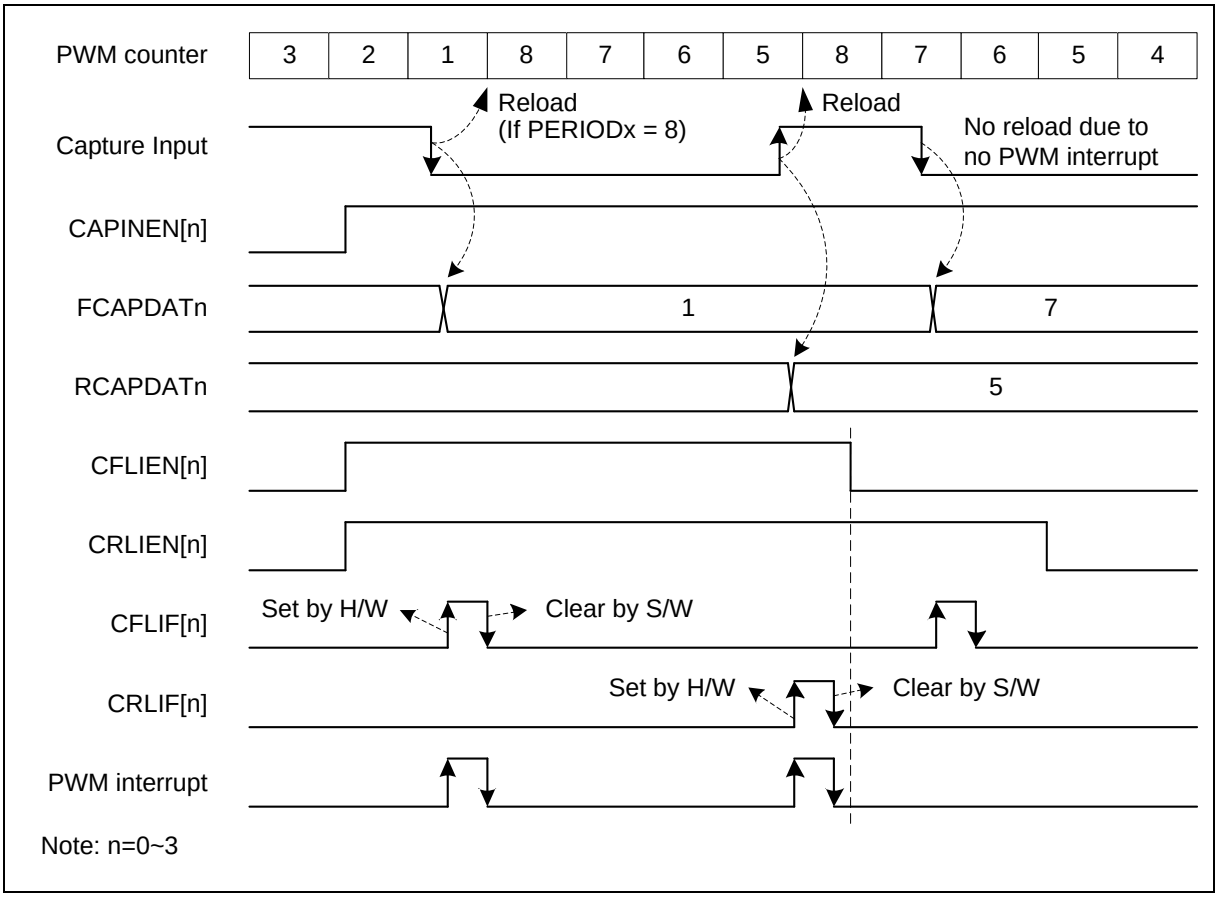


图 6.6-7 捕捉操作时序

这个例子周期PERIOD是 8:

1. 当捕捉中断标志位(CAPIFx)置1时，PWM 计数器将重新加载PWM_PERIODn的值。
2. 通道的低脉冲宽度是(PWM_PERIODn + 1 - PWM_RCAPDATn)
3. 通道的高脉冲宽度是(PWM_PERIODn + 1 - PWM_FCAPDATn)

6.6.4.5 PWM 启动程序

PWM的启动建议程序.

1. 设置时钟源除法寄存器 (PWM_CLKDIV)
2. 设置预分频器 (PWM_CLKPSC)
3. 设置PWM 计数器操作模式单次或自动加载，PWM输出极性 (PWM_CTL)
4. 设置死区控制器开或关 (PWM_CTL)
5. 设置PWM的占空比的比较寄存器 (PWM_CMPDATn)的值.
6. 设置PWM的周期的计数寄存器 (PWM_PERIODn)的值.
7. 设置中断使能寄存器 (PWM_INTEN) (可选)

8. 设置 PWM 输出使能 (PWM_POEN)

PWM_ POEN[0] 使能 -----> PWM0输出 -----> PB.10/PC.9
PWM_ POEN[1] 使能 -----> PWM1输出 -----> PB.11/PC.10
PWM_ POEN[2] 使能 -----> PWM2输出 -----> PB.12/PC.11
PWM_ POEN[3] 使能 -----> PWM3输出 -----> PB.13/PC.12

9. 设置相应PWM通道的GPIO管脚为PWM功能。
使能PWM计数器运行(PWM_CTL).

6.6.4.6 PWM单次模式的重启动程序

在单次模式下，当PWM波形产生一次后，PWM计数器将会自动停止。以下是在单次模式下的重启动PWM的程序。

1. 设置PWM占空比的比较寄存器 (PWM_CMPDATn)。.
2. 设置PWM周期的寄存器(PWM_PERIODn) 后，将产生PWM波形。

6.6.4.7 PWM 停止程序

方法 1:

设置 16-位的周期寄存器 (PWM_PERIODn) 为0， 和监测 PWM_CNTn (16位向下计数器当前值)的值. 当 PWM_CNTn 为 0, CNTENn位清0停止计数。 ***(推荐)***

方法 2:

设置 16-位的周期寄存器(PWM_PERIODn) 为 0，当中断请求发生后CNTENn位清0停止计数。. ***(推荐)***

方法 3:

CNTENn位清0停止计数。 ***(不推荐)***

因为CNTENn位清0后，立刻停止PWM信号输出，将改变PWM输出的占空比，可能会对马达的驱动电路带来损害，因此不建议用方法3。

6.6.4.8 捕捉开始程序

1. 设置时钟源除法寄存器 (PWM_CLKDIV)
2. 设置预分频器 (PWM_CLKPSC)
3. 设置输入捕捉反向开/关和捕捉功能使能 (PWM_CAPCTL)
4. 设置PWM计数器为自动加载模式 (PWM_CTL)

5. 设置PWM周期寄存器 (PWM_PERIODn)
6. 使能捕捉 (PWM_CAPCTL)
7. 使能PWM计数器运行(设置 (CNTENn = 1)

6.6.4.9 PWM 中断发生器

PWM有4个独立的中断结构图如图 6.6-8

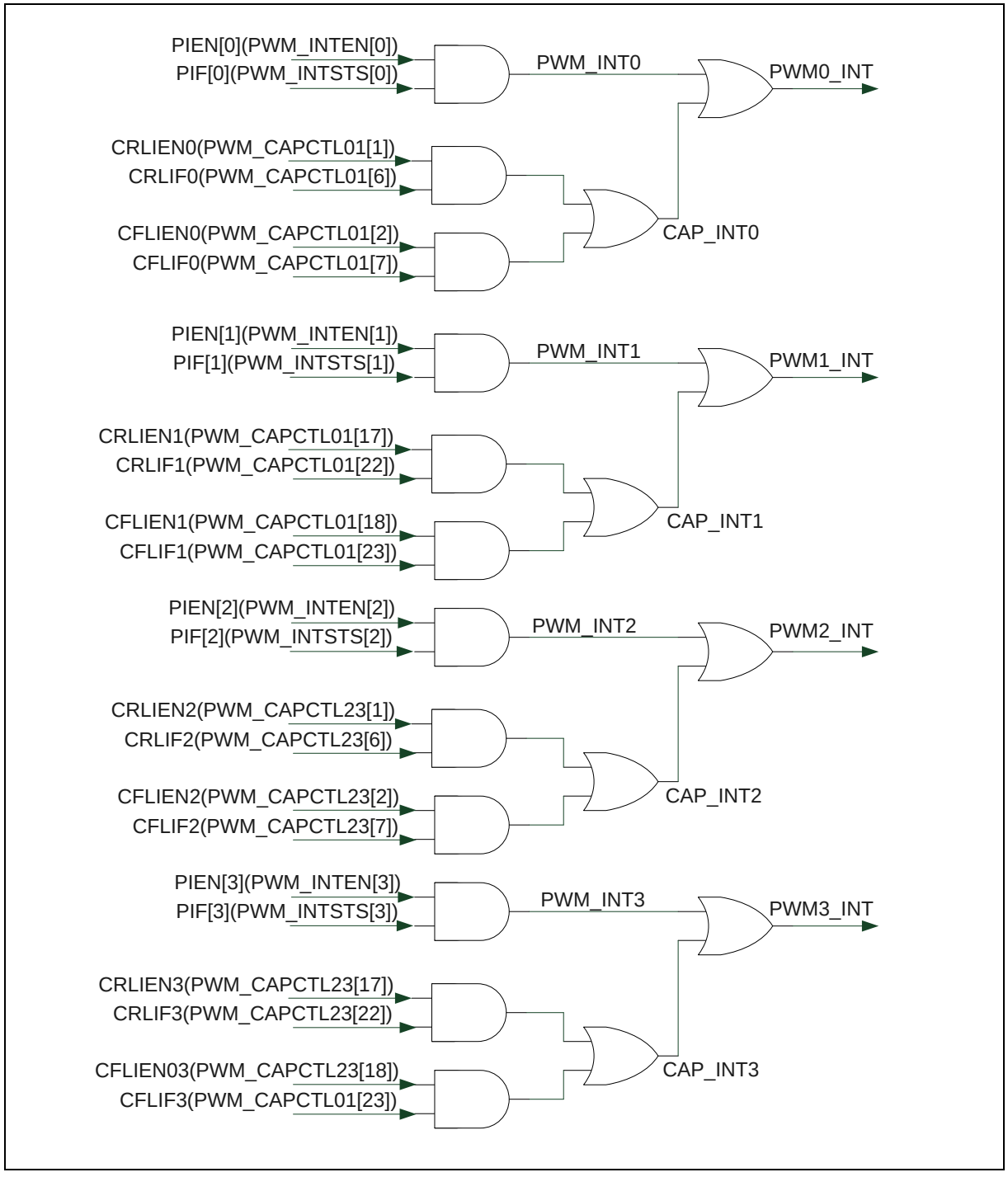


图 6.6-8 PWM中断架构

6.6.5 寄存器 映射

R: 只读, **W**: 只写, **R/W**: 读写, **C**:仅可写 0

寄存器	偏移地址	R/W	描述	复位值
PWM 基地址: **PWM_BA = 0x400E_6000**				
PWM_CLKPSC	PWM_BA+0x00	R/W	PWM 时钟预分频 寄存器	0x0000_0000
PWM_CLKDIV	PWM_BA+0x04	R/W	PWM 时钟除法选择寄存器	0x0000_0000
PWM_CTL	PWM_BA+0x08	R/W	PWM 控制 寄存器	0x0000_0000
PWM_PERIOD0	PWM_BA+0x0C	R/W	PWM 周期 寄存器 0	0x0000_0000
PWM_CMPDAT0	PWM_BA+0x10	R/W	PWM 比较 寄存器 0	0x0000_0000
PWM_CNT0	PWM_BA+0x14	R	PWM 数据 寄存器 0	0x0000_0000
PWM_PERIOD1	PWM_BA+0x18	R/W	PWM 周期 寄存器 1	0x0000_0000
PWM_CMPDAT1	PWM_BA+0x1C	R/W	PWM 比较 寄存器 1	0x0000_0000
PWM_CNT1	PWM_BA+0x20	R	PWM 数据 寄存器 1	0x0000_0000
PWM_PERIOD2	PWM_BA+0x24	R/W	PWM 周期 寄存器 2	0x0000_0000
PWM_CMPDAT2	PWM_BA+0x28	R/W	PWM 比较 寄存器 2	0x0000_0000
PWM_CNT2	PWM_BA+0x2C	R	PWM 数据 寄存器 2	0x0000_0000
PWM_PERIOD3	PWM_BA+0x30	R/W	PWM 周期 寄存器 3	0x0000_0000
PWM_CMPDAT3	PWM_BA+0x34	R/W	PWM 比较 寄存器 3	0x0000_0000
PWM_CNT3	PWM_BA+0x38	R	PWM 数据 寄存器 3	0x0000_0000
PWM_INTEN	PWM_BA+0x40	R/W	PWM 中断 使能 寄存器	0x0000_0000
PWM_INTSTS	PWM_BA+0x44	R/W	PWM中断状态 寄存器	0x0000_0000
PWM_CAPCTL01	PWM_BA+0x50	R/W	捕捉控制寄存器 0	0x0000_0000
PWM_CAPCTL23	PWM_BA+0x54	R/W	捕捉控制寄存器1	0x0000_0000
PWM_RCAPDAT0	PWM_BA+0x58	R/W	捕捉 上升锁存寄存器 (Channel 0)	0x0000_0000
PWM_FCAPDAT0	PWM_BA+0x5C	R/W	捕捉下降锁存 寄存器 (Channel 0)	0x0000_0000
PWM_RCAPDAT1	PWM_BA+0x60	R/W	捕捉上升锁存 寄存器 (Channel 1)	0x0000_0000
PWM_FCAPDAT1	PWM_BA+0x64	R/W	捕捉下降锁存 寄存器 (Channel 1)	0x0000_0000
PWM_RCAPDAT2	PWM_BA+0x68	R/W	捕捉上升锁存 寄存器 (Channel 2)	0x0000_0000
PWM_FCAPDAT2	PWM_BA+0x6C	R/W	捕捉下降锁存 寄存器 (Channel 2)	0x0000_0000

PWM_RCAPDAT3	PWM_BA+0x70	R/W	捕捉上升锁存寄存器 (Channel 3)	0x0000_0000
PWM_FCAPDAT3	PWM_BA+0x74	R/W	捕捉下降锁存 寄存器 (Channel 3)	0x0000_0000
PWM_CAPINEN	PWM_BA+0x78	R/W	捕捉 输入 使能 寄存器	0x0000_0000
PWM_POEN	PWM_BA+0x7C	R/W	PWM输出使能 寄存器	0x0000_0000

6.6.6 寄存器 描述

PWM 时钟预分频寄存器 (PWM_CLKPSC)

寄存器	偏移地址	R/W	描述	复位值
PWM_CLKPSC	PWM_BA+0x00	R/W	PWM时钟预分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DZCNT23							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DZCNT01							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
CLKPSC23							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKPSC01							

位	描述	
[31:24]	**DZCNT23**	**死区间隔寄存器 1** 这个8位决定死区的长度. 死区时间 = (DTCNT23[7:0]+1) * PWM_CLK 周期.
[23:16]	**DZCNT01**	**死区间隔寄存器 0** 这个8位决定死区的长度. 死区时间= (DTCNT01[7:0]+1) * PWM_CLK 周期.
[15:8]	**CLKPSC23**	**时钟预分频 1 PWM 计数器 2 & 3** 在供给计数器 2 & 3前，时钟输入除以 (CLKPSC23+1). 如果 CLKPSC23=0,预分频器时钟输出停止.
[7:0]	**CLKPSC01**	**时钟预分频 0 PWM 计数器 0 & 1** 在供给计数器0 & 1前，时钟输入除以(CLKPSC01+1). 如果CLKPSC01=0, 预分频器时钟输出停止

PWM时钟除法选择寄存器 (PWM_CLKDIV)

寄存器	偏移地址	R/W	描述	复位值
PWM_CLKDIV	PWM_BA+0x04	R/W	PWM时钟除法选择寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	CLKDIV3			Reserved	CLKDIV2		
7	6	5	4	3	2	1	0
Reserved	CLKDIV1			Reserved	CLKDIV0		

位	描述	
[31:15]	Reserved	保留.
[14:12]	CLKDIV3	**PWM 计数器 3时钟除法选择** PWM 计数器 3的时钟选择. 000 = PWM_CLK/2. 001 = PWM_CLK/4. 010 = PWM_CLK/8. 011 = PWM_CLK/16. 100 = PWM_CLK/1.
[11]	Reserved	保留.
[10:8]	CLKDIV2	**PWM 计数器 2时钟除法选择** PWM 计数器 2的时钟选择. (表和 CLKDIV3一样)
[7]	Reserved	保留.
[6:4]	CLKDIV1	**PWM 计数器 1时钟除法选择** PWM 计数器 1的时钟选择. (表和 CLKDIV3一样)
[3]	Reserved	保留.
[2:0]	CLKDIV0	**PWM 计数器 0时钟除法选择** PWM 计数器 0的时钟选择. (表和 CLKDIV3一样)

PWM 控制 寄存器 (PWM_CTL)

寄存器	偏移地址	R/W	描述	复位值
PWM_CTL	PWM_BA+0x08	R/W	PWM 控制 寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved				CNTMODE3	PINV3	Reserved	CNTEN3
23	22	21	20	19	18	17	16
Reserved				CNTMODE2	PINV2	Reserved	CNTEN2
15	14	13	12	11	10	9	8
Reserved				CNTMODE1	PINV1	Reserved	CNTEN1
7	6	5	4	3	2	1	0
Reserved		DTEN23	DTEN01	CNTMODE0	PINV0	Reserved	CNTEN0

位	描述	
[31:28]	**Reserved**	保留.
[27]	**CNTMODE3**	**PWM 计数器 3 自动加载模式/单次模式** 0 =单次模式. 1 = 自动加载模式. **注:** 如果这位有上升变化时PWM_PERIOD3 和 PWM_CMPDAT3将被清0.
[26]	**PINV3**	**PWM 计数器 3反向器 ON/OFF** 0 =反向器 OFF. 1 =反向器 ON.
[25]	**Reserved**	保留.
[24]	**CNTEN3**	**PWM 计数器 3 使能 控制** 0 = PWM 计数器 和时钟预分频器停止运行. 1 = PWM 计数器和时钟预分频器开始运行.
[23:20]	**Reserved**	保留.
[19]	**CNTMODE2**	**PWM 计数器 2 自动加载模式/单次模式** 0 =单次模式. 1 = 自动加载模式. **注:** 如果这位有上升变化时PWM_PERIOD2和 PWM_CMPDAT2将被清0
[18]	**PINV2**	**PWM 计数器 2反向器 ON/OFF** 0 =反向器 OFF. 1 =反向器 ON.
[17]	**Reserved**	保留.

[16]	**CNTEN2**	**PWM 计数器2使能 控制** 0 = PWM 计数器 和时钟预分频器停止运行. 1 = PWM 计数器和时钟预分频器开始运行.
[15:12]	**Reserved**	保留.
[11]	**CNTMODE1**	**PWM 计数器 1 自动加载模式/单次模式** 0 =单次模式. 1 = 自动加载模式. **注:** 如果这位有上升变化时PWM_PERIOD1和 PWM_CMPDAT1将被清0
[10]	**PINV1**	**PWM 计数器 1反向器 ON/OFF** 0 =反向器 OFF. 1 =反向器 ON.
[9]	**Reserved**	保留.
[8]	**CNTEN1**	**PWM 计数器 1 使能 控制** 0 = PWM 计数器 和时钟预分频器停止运行. 1 = PWM 计数器和时钟预分频器开始运行
[7:6]	**Reserved**	保留.
[5]	**DTEN23**	**死区1发生器使能 控制** 0 =死区发生器停止运行. 1 =死区发生器开始运行.
[4]	**DTEN01**	**死区0发生器使能 控制** 0 =死区发生器停止运行. 1 =死区发生器开始运行
[3]	**CNTMODE0**	**PWM 计数器 0 自动加载模式/单次模式** 0 =单次模式. 1 = 自动加载模式. **注:** 如果这位有上升变化时PWM_PERIOD0和 PWM_CMPDAT0将被清0
[2]	**PINV0**	**PWM 计数器 0反向器 ON/OFF** 0 =反向器 OFF. 1 =反向器 ON.
[1]	**Reserved**	保留.
[0]	**CNTEN0**	**PWM 计数器 0 使能 控制** 0 = PWM 计数器 和时钟预分频器停止运行. 1 = PWM 计数器和时钟预分频器开始运行

PWM 周期 寄存器 3~0 (PWM_PERIOD3~0)

寄存器	偏移地址	R/W	描述	复位值
PWM_PERIOD0	PWM_BA+0x0C	R/W	PWM 周期 寄存器 0	0x0000_0000
PWM_PERIOD1	PWM_BA+0x18	R/W	PWM 周期 寄存器 1	0x0000_0000
PWM_PERIOD2	PWM_BA+0x24	R/W	PWM 周期 寄存器 2	0x0000_0000
PWM_PERIOD3	PWM_BA+0x30	R/W	PWM 周期 寄存器 3	0x0000_0000

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

位	描述	
[31:16]	**Reserved**	保留.
[15:0]	**PERIOD**	**PWM 周期 寄存器** PERIOD决定 PWM波的周期. PWM 频率 = PWM_CLK/[(prescale+1)*(clock divider)*(PERIOD+1)] ● 占空比 = (CMP+1)/(PERIOD+1). ● CMP >= PERIOD: PWM 输出高. ● CMP < PERIOD: PWM 低电平宽 = (PERIOD-CMP) 单位； PWM高电平宽 = (CMP+1) 单位. ● CMP = 0: PWM 低电平宽= (PERIOD) 单位；PWM高电平宽 = 1 单位. **注:** 任何写PERIOD操作将在下个PWM周期起作用..

PWM 比较 寄存器 3~0 (PWM_CMPDAT3~0)

寄存器	偏移地址	R/W	描述	复位值
PWM_CMPDAT0	PWM_BA+0x10	R/W	PWM 比较 寄存器 0	0x0000_0000
PWM_CMPDAT1	PWM_BA+0x1C	R/W	PWM 比较 寄存器 1	0x0000_0000
PWM_CMPDAT2	PWM_BA+0x28	R/W	PWM 比较 寄存器 2	0x0000_0000
PWM_CMPDAT3	PWM_BA+0x34	R/W	PWM 比较 寄存器 3	0x0000_0000

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

位	描述	
[31:16]	**Reserved**	保留.
[15:0]	**CMP**	**PWM 比较 寄存器** CMP 决定PWM输出占空比的值. PWM 频率 = PWM_CLK/[(prescale+1)*(clock divider)*(PERIOD+1)]. ● 占空比 = (CMP+1)/(PERIOD+1). ● CMP >= PERIOD: PWM 输出高. ● CMP < PERIOD: PWM 低电平宽 = (PERIOD-CMP) unit; PWM高电平宽 = (CMP+1) 单位. ● CMP = 0: PWM 低电平宽= (PERIOD) unit; PWM高电平宽 = 1 单位. **注:** 任何写 CMP操作将在下个PWM周期起作用.

PWM 数据 寄存器 3~0 (PWM_CNT3~0)

寄存器	偏移地址	R/W	描述	复位值
PWM_CNT0	PWM_BA+0x14	R	PWM 数据 寄存器 0	0x0000_0000
PWM_CNT1	PWM_BA+0x20	R	PWM 数据 寄存器 1	0x0000_0000
PWM_CNT2	PWM_BA+0x2C	R	PWM 数据 寄存器 2	0x0000_0000
PWM_CNT3	PWM_BA+0x38	R	PWM 数据 寄存器 3	0x0000_0000

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

位	描述	
[31:16]	**Reserved**	保留.
[15:0]	**CNT**	**PWM 数据 寄存器** 用户通过CNT 来监控16位下计数器的当前值.

PWM 中断使能 寄存器 (PWM_INTEN)

寄存器	偏移地址	R/W	描述	复位值
PWM_INTEN	PWM_BA+0x40	R/W	PWM 中断 使能 寄存器	0x0000_0000

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

位	描述	
[31:4]	Reserved	保留.
[3:0]	PIEN	PWM 中断 使能使能 0 =中断关闭. 1 = 中断 使能. 注: 每位对应相应的PWM通道.

PWM 中断状态指示寄存器 (PWM_INTSTS)

寄存器	偏移地址	R/W	描述	复位值
PWM_INTSTS	PWM_BA+0x44	R/W	PWM中断状态指示寄存器	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				PIF			

位	描述	
[31:4]	Reserved	保留.
[3:0]	PIF	**PWM 中断标志** 0 =中断标志OFF. 1 = 中断标志 ON. **注1:**每位指示相应的PWM通道. **注2:** 用户对相应的位进行写1清除

捕捉 控制 寄存器 0 (PWM_CAPCTL01)

寄存器	偏移地址	R/W	描述	复位值
PWM_CAPCTL01	PWM_BA+0x50	R/W	捕捉 控制寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CFLIF1	CRLIF1	Reserved	CAPIF1	CAPEN1	CFLIEN1	CRLIEN1	CAPINV1
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CFLIF0	CRLIF0	Reserved	CAPIF0	CAPEN0	CFLIEN0	CRLIEN0	CAPINV0

位	描述	
[31:24]	**Reserved**	保留.
[23]	**CFLIF1**	**捕捉下降锁存中断标志** 0 = 没有 捕捉 下降锁存发生. 1 = 捕捉下降锁存发生, 该标志置高. **注:** 该位写1清除.
[22]	**CRLIF1**	**捕捉上升锁存中断标志** 0 = 没有 捕捉 下降锁存发生. 1 = 捕捉下降锁存发生, 该标志置高. **注:** 该位写1清除.
[21]	**Reserved**	保留.
[20]	**CAPIF1**	**捕捉 1中断指示** 0 = 中断标志 OFF. 1 =中断标志ON. **注:**如果 这位是“1”， 当下一个捕捉中断发生将不能重新加载1，该位写1清除.
[19]	**CAPEN1**	**捕捉 通道 1 功能 使能控制** 0 = 捕捉 功能关闭. 1 = 捕捉 功能使能. **注1:** 当 使能,捕捉锁存 PMW计数器1 存到PWM_RCAPDAT1 (上升) 和 PWM_FCAPDAT1 (下降). **注2:** 当关闭, 不更新 PWM_RCAPDAT1 和 PWM_FCAPDAT1, 并关闭通道1的中断
[18]	**CFLIEN1**	**通道1 下降中断 使能控制** 0 = 通道1 下降中断关闭.

		1 = 通道1 下降中断 使能. **注:** 当 使能, 如果捕捉发现通道1有下降变化，捕捉发生中断
[17]	**CRLIEN1**	**通道1 上升中断 使能控制** 0 = 通道1 上升中断关闭. 1 = 通道1 上升中断 使能. **注:** 当 使能, 如果捕捉发现通道1有上升变化，捕捉发生中断
[16]	**CAPINV1**	**捕捉 1反向器 使能 控制** 0 = 捕捉 源 反向器关闭. 1 = 捕捉源 反向器使能.从GPIO输入信号反向.
[15:8]	**Reserved**	保留.
[7]	**CFLIF0**	**捕捉下降锁存中断标志** 0 = 没有 捕捉 下降锁存发生. 1 = 捕捉下降锁存发生, 该标志置高. **注:** 该位写1清除.
[6]	**CRLIF0**	**捕捉上升锁存中断标志** 0 = 没有 捕捉 下降锁存发生. 1 = 捕捉下降锁存发生, 该标志置高. **注:** 该位写1清除
[5]	**Reserved**	保留.
[4]	**CAPIF0**	**捕捉 0中断指示** 0 = 中断标志 OFF. 1 =中断标志ON. **注:**如果 这位是“1”， 当下一个捕捉中断发生将不能重新加载1，该位写1清除
[3]	**CAPEN0**	**捕捉 通道 0 功能 使能控制** 0 = 捕捉 功能关闭. 1 = 捕捉 功能 使能. **注1:** 当 使能,捕捉锁存 PMW计数器0 存到PWM_RCAPDAT0 (上升) 和 PWM_FCAPDAT0 (下降). **注2:** 当关闭, 不更新 PWM_RCAPDAT0 和 PWM_FCAPDAT0, 并关闭通道0的中断
[2]	**CFLIEN0**	**通道0下降中断 使能控制** 0 = 通道0 下降中断关闭. 1 = 通道0下降中断 使能. **注:** 当 使能, 如果捕捉发现通道0有下降变化，捕捉发生中断
[1]	**CRLIEN0**	**通道0 上升中断 使能控制** 0 = 通道0 上升中断关闭. 1 = 通道0 上升中断 使能. **注:** 当 使能, 如果捕捉发现通道0有上升变化，捕捉发生中断

[0]	**CAPINV0**	**捕捉 0反向器 使能 控制** 0 = 捕捉 源 反向器关闭. 1 = 捕捉源 反向器使能.从GPIO输入信号反向.

捕捉 控制 寄存器 1 (PWM_CAPCTL23)

寄存器	偏移地址	R/W	描述	复位值
PWM_CAPCTL23	PWM_BA+0x54	R/W	捕捉 控制 寄存器 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
CFLIF3	CRLIF3	Reserved	CAPIF3	CAPEN3	CFLIEN3	CRLIEN3	CAPINV3
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
CFLIF2	CRLIF2	Reserved	CAPIF2	CAPEN2	CFLIEN2	CRLIEN2	CAPINV2

位	描述	
[31:24]	**Reserved**	保留.
[23]	**CFLIF3**	**捕捉下降锁存中断标志** 0 = 没有 捕捉 下降锁存发生. 1 = 捕捉下降锁存发生, 该标志置高. **注:** 该位写1清除.
[22]	**CRLIF3**	**捕捉上升锁存中断标志** 0 = 没有 捕捉 下降锁存发生. 1 = 捕捉下降锁存发生, 该标志置高. **注:** 该位写1清除.
[21]	**Reserved**	保留.
[20]	**CAPIF3**	**捕捉 3中断指示** 0 = 中断标志 OFF. 1 =中断标志ON. **注:**如果 这位是“1”， 当下一个捕捉中断发生将不能重新加载1，该位写1清除
[19]	**CAPEN3**	**捕捉 通道 3 功能 使能控制** 0 = 捕捉 功能关闭. 1 = 捕捉 功能 使能. **注1:** 当 使能,捕捉锁存 PMW计数器3 存到PWM_RCAPDAT3(上升) 和 PWM_FCAPDAT3 (下降). **注2:** 当关闭, 不更新 PWM_RCAPDAT3 和 PWM_FCAPDAT3, 并关闭通道0的中断.
[18]	**CFLIEN3**	**通道3下降中断 使能控制** 0 = 通道3 下降中断关闭.

		1 = 通道3下降中断 使能. **注:** 当 使能, 如果捕捉发现通道3有下降变化，捕捉发生中断
[17]	**CRLIEN3**	**通道3 上升中断 使能控制** 0 = 通道3 上升中断关闭. 1 = 通道3上升中断 使能. **注:** 当 使能, 如果捕捉发现通道3有上升变化，捕捉发生中断
[16]	**CAPINV3**	**捕捉 0反向器 使能 控制** 0 = 捕捉 源 反向器关闭. 1 = 捕捉源 反向器使能.从GPIO输入信号反向
[15:8]	**Reserved**	保留.
[7]	**CFLIF2**	**捕捉下降锁存中断标志** 0 = 没有 捕捉 下降锁存发生. 1 = 捕捉下降锁存发生, 该标志置高. **注:** 该位写1清除.
[6]	**CRLIF2**	**捕捉上升锁存中断标志** 0 = 没有 捕捉 下降锁存发生. 1 = 捕捉下降锁存发生, 该标志置高. **注:** 该位写1清除
[5]	**Reserved**	保留.
[4]	**CAPIF2**	**捕捉 2中断指示** 0 = 中断标志 OFF. 1 =中断标志ON. **注:**如果 这位是“1”， 当下一个捕捉中断发生将不能重新加载1，该位写1清除.
[3]	**CAPEN2**	**捕捉 通道 2功能 使能控制** 0 = 捕捉 功能关闭. 1 = 捕捉 功能 使能. **注1:** 当 使能,捕捉锁存 PMW计数器2 存到PWM_RCAPDAT2 (上升) 和 PWM_FCAPDAT2 (下降). **注2:** 当关闭, 不更新 PWM_RCAPDAT2和 PWM_FCAPDAT2, 并关闭通道0的中断
[2]	**CFLIEN2**	**通道2下降中断 使能控制** 0 = 通道2 下降中断关闭. 1 = 通道2下降中断 使能. **注:** 当 使能, 如果捕捉发现通道2有下降变化，捕捉发生中断
[1]	**CRLIEN2**	**通道2上升中断 使能控制** 0 = 通道2 上升中断关闭. 1 = 通道2 上升中断 使能. **注:** 当 使能, 如果捕捉发现通道2有上升变化，捕捉发生中断
[0]	**CAPINV2**	**捕捉 2反向器 使能 控制** 0 = 捕捉 源 反向器关闭.

		1 = 捕捉源 反向器使能.从GPIO输入信号反向

捕捉上升锁存寄存器3-0 (PWM_RCAPDAT3-0)

寄存器	偏移地址	R/W	描述	复位值
PWM_RCAPDAT0	PWM_BA+0x58	R/W	捕捉上升锁存寄存器 (Channel 0)	0x0000_0000
PWM_RCAPDAT1	PWM_BA+0x60	R/W	捕捉上升锁存寄存器 (Channel 1)	0x0000_0000
PWM_RCAPDAT2	PWM_BA+0x68	R/W	捕捉上升锁存 寄存器 (Channel 2)	0x0000_0000
PWM_RCAPDAT3	PWM_BA+0x70	R/W	捕捉上升锁存 寄存器 (Channel 3)	0x0000_0000

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

位	描述	
[31:16]	**Reserved**	保留.
[15:0]	**RCAPDAT**	**捕捉上升锁存 寄存器** 当通道0/1/2/3有上升变化时锁存PWM计数器的值.

捕捉下降锁存寄存器3-0 (PWM_FCAPDAT3-0)

寄存器	偏移地址	R/W	描述	复位值
PWM_FCAPDAT0	PWM_BA+0x5C	R/W	捕捉下降锁存 寄存器 (Channel 0)	0x0000_0000
PWM_FCAPDAT1	PWM_BA+0x64	R/W	捕捉下降锁存寄存器 (Channel 1)	0x0000_0000
PWM_FCAPDAT2	PWM_BA+0x6C	R/W	捕捉下降锁存寄存器 (Channel 2)	0x0000_0000
PWM_FCAPDAT3	PWM_BA+0x74	R/W	捕捉下降锁存寄存器 (Channel 3)	0x0000_0000

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

位	描述	
[31:16]	**Reserved**	保留.
[15:0]	**FCAPDAT**	捕捉下降锁存**寄存器** 当通道0/1/2/3有下降变化时锁存PWM计数器的值.

捕捉输入使能 寄存器 (PWM_CAPINEN)

寄存器	偏移地址	R/W	描述	复位值
PWM_CAPINEN	PWM_BA+0x78	R/W	捕捉输入使能 寄存器	0x0000_0000

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

位	描述	
[31:4]	**Reserved**	保留.
[n]=0,1,2,3	**CAPINEN[n]**	**捕捉输入 使能 控制** 0 = PWM 捕捉输入通道关闭. PWM捕捉功能输入的信号一直作为0。 1 = PWM 捕捉输入通道使能. PWM捕捉功能输入的信号是相应多功能管脚的信号。 n = 0, 捕捉通道 0是PB.10 或 PC.9. n = 1, 捕捉通道1 是PB.11 或 PC.10. n = 2, 捕捉通道2 是PB.12 或 PC.11. n = 3, 捕捉通道3是PB.13 或 PC.12.

PWM 输出使能 寄存器 (PWM_POEN)

寄存器	偏移地址	R/W	描述	复位值
PWM_POEN	PWM_BA+0x7C	R/W	PWM 输出使能 寄存器	0x0000_0000

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

位	描述	
[31:4]	**Reserved**	保留.
[3:0]	**POEN**	**PWM 计数器 输出使能** 0 = PWM 输出关闭. 1 = PWM 输出使能. **注:** 每位对应PWM相应的通道.

6.7 看门狗定时器 (WDT)

6.7.1 概述

当系统运行到一个未知状态时，通过看门狗定时器使系统复位，可以预防系统进入到无限期的死循环。此外，该看门狗定时器支持系统从掉电状态唤醒功能。

6.7.2 特性

- 18位的向上计数器设定看门狗定时器溢出时间间隔
- 溢出时间间隔(2^4 ~ 2^{18})个WDT_CLK时钟周期可选，如WDT_CLK = 32 kHz，那么溢出时间间隔是32.5 ms ~ 8.224 s
- 系统复位保持时间(1 / WDT_CLK) * 63
- 支持看门狗定时器复位延时，有 1026、130 、18 或 3 个WDT_CLK 延时时间.
- 当看门狗时钟源选择32 kHz，看门狗定时器支持唤醒功能

6.7.3 框图

看门狗定时器和时钟控制框图分别是图 6.7-1 和 图 6.7-2 。

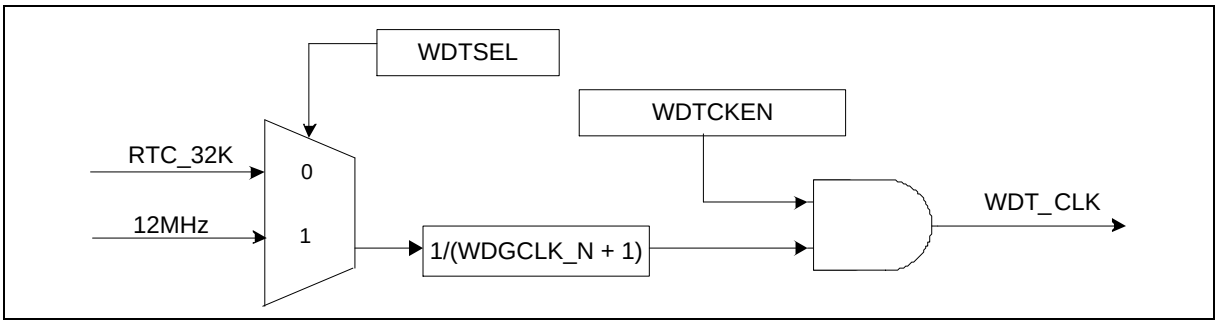
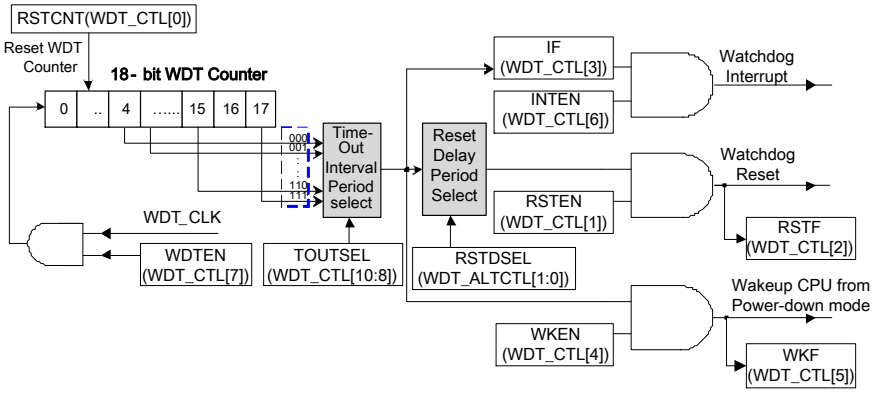


图 6.7-1 看门狗时钟控制图



注1: 看门狗复位CPU维持63 WDT_CLK时钟周期.

注2: 如果准备用看门狗定时器来唤醒掉电的系统，CPU进入掉电前就要设置与看门狗相同的时钟源,.

注3: 看门狗复位延时时间可设3/18/130/1026 WDT_CLK.

图 6.7-2 看门狗定时器框图

6.7.4 功能描述

看门狗定时器 (WDT)包括一个18位的可设置溢出间隔的上计数器。. 表 6.7-1 列出看门狗定时器溢出间隔选项， 图 6.7-3描述看门狗定时器溢出间隔和复位时序。 .

看门狗定时器溢出中断

设置 WDTEN (WDT_CTL[7]) 位为 1将使能看门狗功能并使看门狗定时器开始进行上计数。通过设置TOUTSEL (WDT_CTL[10:8])寄存器有8种溢出时间间隔可以选择，当看门狗计数器计数到你选择的TOUTSEL (WDT_CTL[10:8])的相应值时，看门狗定时器将产生溢出中断并将IF(WDT_CTL[3])标志位立即置1。

看门狗定时器复位延时和复位系统

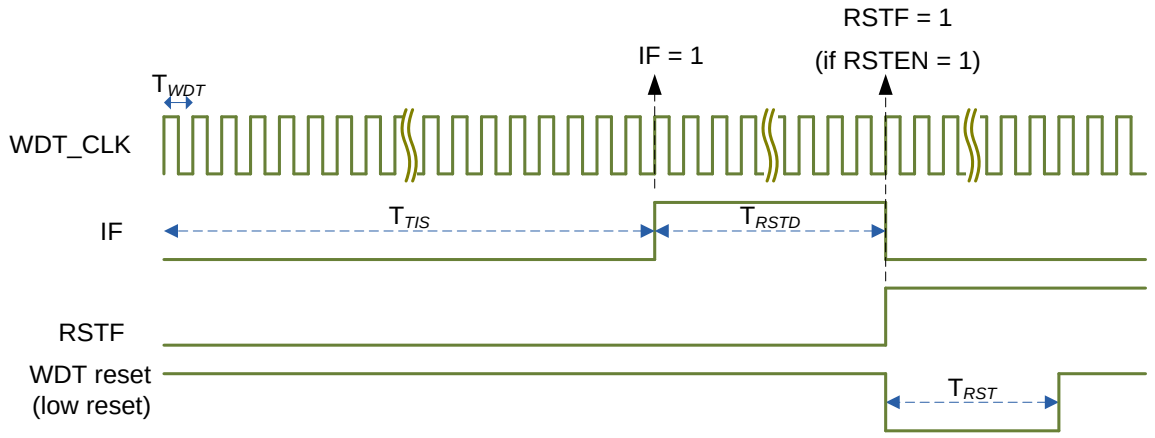
用户使能RSTCNT (WDT_CTL[0])位，看门狗IF标志被置1后 还有T_{RSTD} 延时周期用来复位看门狗18位计数器的值以防止看门狗定时器溢出复位产生。如果看门狗计数器没有在T_{RSTD}延时周期到达前被用户清除，RSTF (WDT_RSTSTS[2])位又被设置为1，RSTEN (WDT_CTL[1]) 位又被使能，芯片立即进行复位状态。参考 图 6.7-3, T_{RSTD} 将保持 63 个看门狗时钟后芯片重新开始从复位向量(0x0000_0000)运行. 看门狗定时器溢出复位芯片后RSTF标志被置1， 用户通过RSTF 标志来确认系统被看门狗定时器溢出复位与否。

看门狗定时器唤醒

如果看门狗定时器时钟源选择32 kHz, 当看门狗产生中断信号并WKEN (WDT_CTL[4])位使能时系统能从掉电模式唤醒，与此同时WKF (WDT_CTL[5]) 标志将自动置1，用户可以通过WKF标志来确认系统是否为看门狗定时器溢出产生中断而唤醒。

WTIS	溢出时间间隔 T_{TIS}	中断周期 T_{INT}	复位延时周期 T_{RSTD}
000	$2^4 * T_{WDT}$	$1024 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
001	$2^6 * T_{WDT}$	$1024 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
010	$2^8 * T_{WDT}$	$1024 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
011	$2^{10} * T_{WDT}$	$1024 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
100	$2^{12} * T_{WDT}$	$1024 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
101	$2^{14} * T_{WDT}$	$1024 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
110	$2^{16} * T_{WDT}$	$1024 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$
111	$2^{18} * T_{WDT}$	$1024 * T_{WDT}$	$(3/18/130/1026) * T_{WDT}$

表 6.7-1看门狗定时器时间间隔



- *T_{WDT} : Watchdog Clock Time Period*
- *T_{TIS} : Watchdog Time-out Interval Period ((2^4 ~ 2^{18}) * T_{WDT})*
- *T_{RSTD} : Watchdog Reset Delay Period*
  *- Selectable 3/18/130/1026 * T_{WDT} delay period controlled by RSTDSEL (WDT_ALTCTL[1:0])*
- *T_{RST} : Watchdog Reset Period (63 * T_{WDT})*

图 6.7-3 看门狗中断和复位时序

6.7.5 寄存器 映射

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移地址	R/W	描述	复位值
WDT 基地址: **WDT_BA = 0x400E_F000**				
WDT_CTL	WDT_BA+0x00	R/W	看门狗定时器控制 寄存器	0x0000_0700
WDT_ALTCTL	WDT_BA+0x04	R/W	看门狗定时器 选择寄存器	0x0000_0000
WDT_RSTSTS	WDT_BA+0x10	R	看门狗定时器复位状态寄存器	0x0000_0700

6.7.6 寄存器 描述

看门狗控制寄存器 (WDT_CTL)

寄存器	偏移地址	R/W	描述	复位值
WDT_CTL	WDT_BA+0x00	R/W	看门狗控制寄存器	0x0000_0700

31	30	29	28	27	26	25	24
ICEDEBUG	Reserved						
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved					TOUTSEL		
7	6	5	4	3	2	1	0
WDTEN	INTEN	WKF	WKEN	IF	RSTFC	RSTEN	RSTCNT

位	描述	
[31]	**ICEDEBUG**	**ICE调试模式 影响计数器功能控制** 0 = ICE调试模式下看门狗计数器受CPU被ICE暂停影响，计数器也暂停计数 1 = ICE调试模式下看门狗计数器不受CPU被ICE暂停影响，计数器仍运行
[30:11]	**Reserved**	保留.
[10:8]	**TOUTSEL**	**看门狗定时器溢出时间间隔** 这3 位 用于选择看门狗定时器时间间隔周期 000 = 2^4 *T_{WDT}. 001 = 2^6 *T_{WDT}. 010 = 2^8 *T_{WDT}. 011 = 2^{10} *T_{WDT}. 100 = 2^{12} *T_{WDT}. 101 = 2^{14} *T_{WDT}. 110 = 2^{16} *T_{WDT}. 111 = 2^{18} *T_{WDT}.
[7]	**WDTEN**	**看门狗定时器使能** 0 = 看门狗定时器关闭 (这个动作将复位内部计数器). 1 = 看门狗 使能.
[6]	**INTEN**	**看门狗定时器中断 使能** 如果该位使能，看门狗定时器溢出中断信号将产生并通知CPU. 0 = 看门狗定时器中断关闭. 1 = 看门狗定时器中断 使能.
[5]	**WKF**	**看门狗定时器 唤醒标志**

		这位表示看门狗定时器中断唤醒标志状态 0 = 看门狗不能唤醒芯片. 1 = 看门狗定时器中断信号可以将芯片从掉电状态唤醒. **注:** 该位写1清除.
[4]	**WKEN**	**看门狗定时器 唤醒功能使能** 如果此位被置1, 当IF (WDT_CTL[3])被置1且INTEN (WDT_CTL[6])被使能，看门狗定时溢出中断信号将会触发芯片唤醒。 0 =唤醒触发功能关闭，即便看门狗定时器定时溢出中断发生. 1 =唤醒触发事件使能，如果看门狗定时溢出中断产生，将触发芯片唤醒。 **注:** 芯片能被看门狗定时器定时溢出中断信号唤醒的条件是看门狗定时器时钟源必须为32 kHz晶振
[3]	**IF**	**看门狗定时器中断标志** 当看门狗定时器的计数值到达设定的时间间隔周期值时，该位被置 1 0 =看门狗定时器溢出中断没有发生. 1 = 看门狗定时器溢出中断发生. **注:** 该位被置 1清除.
[2]	**RSTFC**	**看门狗定时器复位标志清除** 该位写1清除 RSTF (WDT_RSTSTS [2]).
[1]	**RSTEN**	**看门狗复位 使能 控制** 如果看门狗计数器的值在延时周期结束前没有被清除，这位值为1时将复位芯片。 0 = 看门狗定时器复位功能关闭. 1 = 看门狗定时器复位功能使能.
[0]	**RSTCNT**	**清除看门狗计数器** 0 =无动作. 1 = 清除18-位看门狗计数器. **注:** 这位将自动被硬件清除.

看门狗定时器选择功能寄存器 (WDT_ALTCTL)

寄存器	偏移地址	R/W	描述	复位值
WDT_ALTCTL	WDT_BA+0x04	R/W	看门狗定时器选择功能 寄存器	0x0000_0000

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

位	描述	
[31:2]	**Reserved**	保留.
[1:0]	**RSTDSEL**	**看门狗定时器复位延时选择** 看门狗定时器溢出复位发生后, 在定时器复位延时周期内，软件可以清除看门狗定时器的计数器的值，以防止看门狗溢出中断复位产生。软件能设置不同的看门狗复位延时时间，对应不同看门狗复位延时时间。 00 = 看门狗定时器复位延时周期是 (1024+2) * WDT_CLK. 01 =看门狗定时器复位延时周期是(128+2) * WDT_CLK. 10 =看门狗定时器复位延时周期是(16+2) * WDT_CLK. 11 =看门狗定时器复位延时周期是(1+2) * WDT_CLK. **注:** 看门狗定时器溢出复位发生后该位将被清0.

看门狗定时器复位状态 寄存器 (WDT_RSTSTS)

寄存器	偏移地址	R/W	描述	复位值
WDT_RSTSTS	WDT_BA+0x10	R	看门狗定时器复位状态寄存器	0x0000_0700

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					**RSTF**	**Reserved**	

位	描述	
[31:3]	**Reserved**	保留.
[2]	**RSTF**	看门狗定时器复位状态标志 这位表示系统是否被看门狗定时器溢出复位. 0 = 看门狗定时器溢出复位没有发生. 1 =看门狗定时器溢出复位发生. **注:** 该位被RSTFC (WDT_CTL [2])写1清除.
[1:0]	**Reserved**	保留.

6.8 窗看门狗定时器 (WWDT)

6.8.1 概述

窗看门狗定时器(WWDT)用于在一个窗口时间内执行窗看门狗复位，以防止程序在不可预知条件下跑到一个不可控的状态。.

6.8.2 特性

- 6位向下计数器CNTDAT (WWDT_CNT[5:0]) 和6位比较值CMPDAT(WWDT_CTL[21:16])，使得窗口期更加灵活。
- 最多 11-位 WWDT时钟预分频器PSCSEL (WWDT_CTL[11:8]) 去设置WWDT溢出区间值

6.8.3 框图

窗看门狗定时器框图如下.

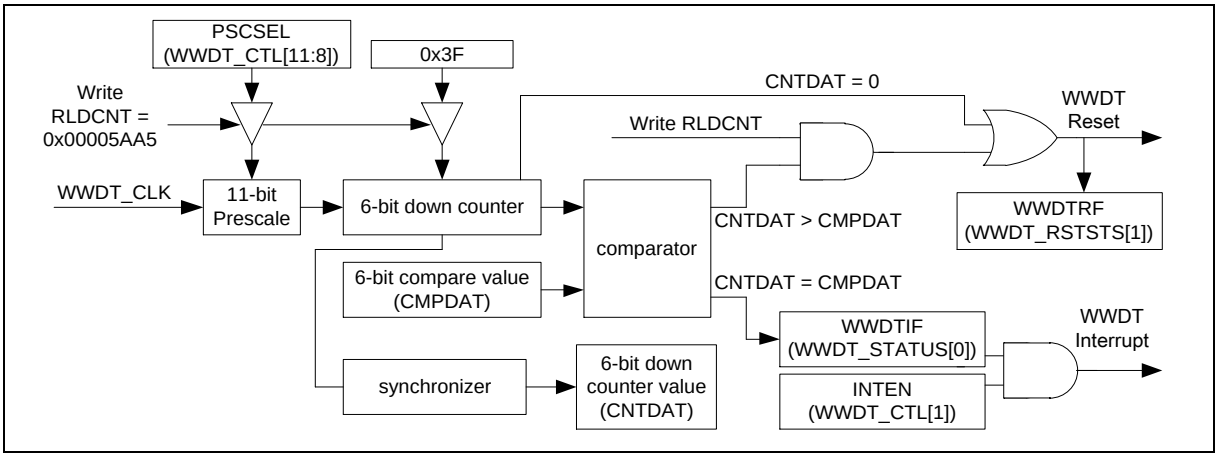


图 6.8-1 窗看门狗定时器框图

6.8.4 功能描述

窗看门狗定时器包括一个带可选择预分频值的6位下计数器来定义不同的溢出时间间隔。窗看门狗6位计数器的时钟源是以RTC的时钟(32 kHz) 为基础最多可进行11位预分频的值，该值受寄存器PSCSEL (WWDT_CTL[11:8] WWDT 预分频周期)选择的值影响， PSCSEL的选项与预分频的值对应关系如下表。.

PSCSEL	预分频器值	溢出周期	溢出间隔 (WWDT_CLK= 32 KHz)
0000	1	$1 * 64 * T_{WWDT}$	2 ms
0001	2	$2 * 64 * T_{WWDT}$	4 ms

0010	4	4 * 64 * T_{WWDT}	8 ms
0011	8	8 * 64 * T_{WWDT}	16 ms
0100	16	16 * 64 * T_{WWDT}	32 ms
0101	32	32 * 64 * T_{WWDT}	64 ms
0110	64	64 * 64 * T_{WWDT}	128 ms
0111	128	128 * 64 * T_{WWDT}	256 ms
1000	192	192 * 64 * T_{WWDT}	384 ms
1001	256	256 * 64 * T_{WWDT}	512 ms
1010	384	384 * 64 * T_{WWDT}	768 ms
1011	512	512 * 64 * T_{WWDT}	1024 ms
1100	768	768 * 64 * T_{WWDT}	1536 ms
1101	1024	1024 * 64 * T_{WWDT}	2048 ms
1110	1536	1536 * 64 * T_{WWDT}	3072 ms
1111	2048	2048 * 64 * T_{WWDT}	4096 ms

表 6.8-1 窗看门狗定时器预分频值选择

窗看门狗定时器仅可以在芯片上电或复位后通过软件设置WWDTEN (WWDT_CTL[0] WWDT 使能)位为1使能一次，WWDT 下计数器将从0x3F开始计数，不能通过软件进行停止操作直到芯片再次复位。.

在WWDT计数器下计数期间， 如果WWDT计数器的值等于CMPDAT (WWDT_CTL [21:16] WWDT窗比较寄存器)的值WWDTIF (WWDT_STATUS[0] WWDT比较匹配中断标志)置1；如果 INTEN (WWDT_CTL[1] WWDT中断使能) 被软件设置为1,当WWDT 超时中断信号发生时硬件会立即将WWDTIF置1.

当WWDT计数器的值到达0时WWDT超时复位信号将发生。在 WWDT 计数器向下计数到0之前，软件能写**0x00005AA5** 到 WWDT_RLDCNT 寄存器 重新加载 0x3F到WWDT计数器以防止当WWDT计数器的值(WWDT_CNT)等于或小于CMPDAT的值时发生超时复位， 如果WWDT计数器(WWDT_CNT)的值大于CMPDAT的值并且软件写**0x00005AA5** 到WWDT_RLDCNT 寄存器，WWDT复位信号将马上给芯片发出复位信号。图 6.8-2 显示了WWDT复位和加载的过程.

为了避免程序在运行过程中意外中止窗看门狗， WWDT_CTL仅可在芯片上电或复位后写一次，软件不能停止窗看门狗计数器运行(WWDTEN),当WWDTEN位被置1后不能改变超时周期(PSCSEL)或改变比较值(CMPDAT) 除非芯片复位。

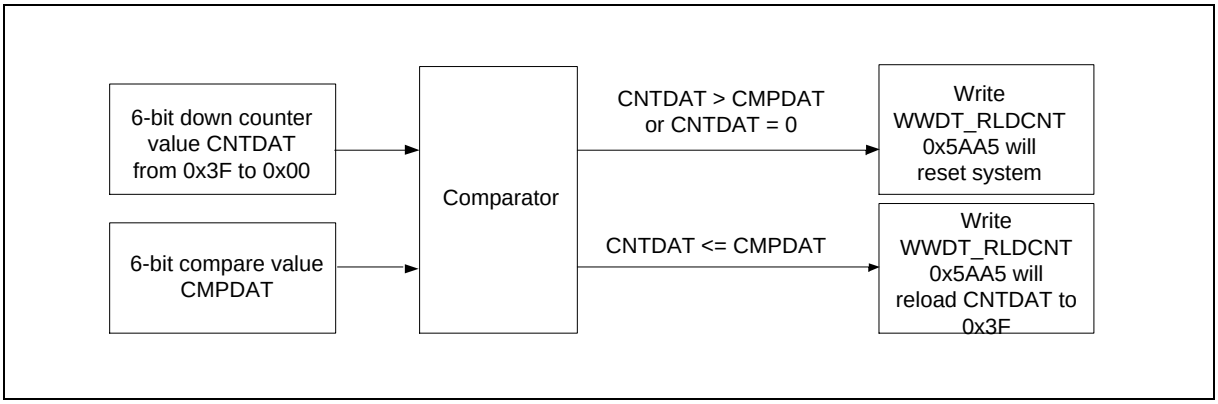


图 6.8-2窗看门狗定时器的复位和加载过程

当用户对WWDT_RLDCNT寄存器写0x00005AA5来重载WWDT的值到0x3F的时候，大概需要3个WWDT时钟周期来同步重载命令到实际执行重载操作。这意味着如果用户设置PSCSEL (WWDT_CTL[11:8]预分频周期选择)的值为0000，即计数器预分频值设置为1，CMPDAT (WWDT_CTL[21:16]比较值)寄存器的值必须大于2；否则，当WWDTIF标志产生后，写WWDT_RLDCNT来加载窗看门狗定时器的值为0x3F操作无效，WWDTIF (WWDT_STATUS[0])将会产生，窗看门狗定时器系统复位将发生。以下表显示CMPDAT的范围.

PSCSEL	预分频值	有效 CMPDAT 的值
0000	1	0x3 ~ 0x3F
0001	2	0x2 ~ 0x3F
Others	Others	0x0 ~ 0x3F

表 6.8-2 CMPDAT 设置范围

6.8.5 寄存器 映射

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移地址	R/W	描述	复位值
WWDT基地址: **WWDT_BA = 0x400E_F100**				
WWDT_RLDCNT	WWDT_BA+0x00	W	窗看门狗定时 器加载寄存器	0x0000_0000
WWDT_CTL	WWDT_BA+0x04	R/W	窗看门狗定时 器控制寄存器	0x003F_0800
WWDT_STATUS	WWDT_BA+0x08	R/W	窗看门狗定时 器状态寄存器	0x0000_0000
WWDT_CNT	WWDT_BA+0x0C	R	窗看门狗定时 器计数器值寄存器	0x0000_003F
WWDT_RSTSTS	WWDT_BA+0x18	R	窗看门狗定时 器复位状态寄存器	0x0000_0000

6.8.6 寄存器 描述

窗看门狗定时器加载寄存器 (WWDT_RLDCNT)

寄存器	偏移地址	R/W	描述	复位值
WWDT_RLDCNT	WWDT_BA+0x00	W	窗看门狗定时器加载寄存器	0x0000_0000

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

位	描述	
[31:0]	**RLDCNT**	**WWDT 加载寄存器** 对此寄存器写0x00005AA5 将会重载窗口看门狗定时器计数器的值到0x3F. **注:**用户只能当窗口看门狗计数器的值在0到CMPDAT之间才可以写RLDCNT寄存器来重载窗口看门狗计数器。如果窗口看门狗计数器的值大于CMPDAT 时写WWDT_RLDCNT寄存器，窗口看门狗定时器复位信号仍会立即产生。.

窗看门狗定时 器控制寄存器 (WWDT_CTL)

寄存器	偏移地址	R/W	描述	复位值
WWDT_CTL	WWDT_BA+0x04	R/W	窗看门狗定时 器控制寄存器	0x003F_0800

该 寄存器 仅在芯片上电或复位后写一次.

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

位	描述	
[31]	**ICEDEBUG**	**ICE 调试模式下窗看门狗计数控制位** 0 =ICE 调试模式下影响窗看门狗定时器计数，当CPU被ICE暂停后，窗看门狗向下计数器计数值将保持不变 1 = ICE 调试模式下不影响窗看门狗定时器计数，窗看门狗定时器向下计数器将会保持继续计数，不管CPU是否被ICE暂停。
[30:22]	**Reserved**	保留.
[21:16]	**CMPDAT**	**窗看门狗定时器窗口比较位** 设置此寄存器来调整有效加载窗口 **注**：当前窗看门狗定时器计数器的值在0到CMPDAT之间时，用户才可以写RLDCNT来加载窗看门狗定时器计数器的值。如果当前窗看门狗定时器计数器的值大于CMPDAT时用户写RLDCNT寄存器，窗看门狗定时器复位信号仍会立即产生。
[15:12]	**Reserved**	保留.
[11:8]	**PSCSEL**	**WWDT定时器计数器预分频周期选择** 0000 =预分频为1; 最大定时溢出周期是1 * 64 * T_{WWDT}. 0001 =预分频为2; 最大定时溢出周期是2 * 64 * T_{WWDT}. 0010 =预分频为4; 最大定时溢出周期是4 * 64 * T_{WWDT}. 0011 =预分频为8; 最大定时溢出周期是8 * 64 * T_{WWDT}. 0100 =预分频为16; 最大定时溢出周期是16 * 64 * T_{WWDT}. 0101 =预分频为32; 最大定时溢出周期是32 * 64 * T_{WWDT}. 0110 =预分频为64; 最大定时溢出周期是64 * 64 * T_{WWDT}. 0111 =预分频为128; 最大定时溢出周期是128 * 64 * T_{WWDT}. 1000 =预分频为192; 最大定时溢出周期是192 * 64 * T_{WWDT}. 1001 =预分频为256; 最大定时溢出周期是256 * 64 * T_{WWDT}.

		1010 =预分频为384; 最大定时溢出周期是384 * 64 * T_{WWDT}. 1011 =预分频为512; 最大定时溢出周期是512 * 64 * T_{WWDT}. 1100 =预分频为768; 最大定时溢出周期是768 * 64 * T_{WWDT}. 1101 =预分频为1024; 最大定时溢出周期是1024 * 64 * T_{WWDT}. 1110 =预分频为1536; 最大定时溢出周期是1536 * 64 * T_{WWDT}. 1111 =预分频为2048; 最大定时溢出周期是2048 * 64 * T_{WWDT}.
[7:2]	**Reserved**	保留.
[1]	**INTEN**	**WWDT 中断 使能 控制** 如果该位被使能，如果有WWDT计数器比较匹配中断信号产生，就会通知CPU 0 = WWDT计数器比较匹配中断禁止 1 = WWDT计数器比较匹配中断使能
[0]	**WWDTEN**	**WWDT 使能 控制** 设置该位来使能窗看门狗定时器计数器计数 0 =窗看门狗定时器计数器停止. 1 =窗看门狗定时器计数器计数开始

窗看门狗定时 器状态寄存器 (WWDT_STATUS)

寄存器	偏移地址	R/W	描述	复位值
WWDT_STATUS	WWDT_BA+0x08	R/W	窗看门狗定时 器状态寄存器	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						WWDTRFC	WWDTIF

位	描述	
[31:2]	**Reserved**	保留.
[1]	**WWDTRFC**	**WWDT 超时复位标志清除** 当窗看门狗复位发生了，寄存器 WWDTRF (WWDT_RSTSTS [1])将会置1. 对该位写1会清除WWDTRF (WWDT_RSTSTS [1]) 复位标志.
[0]	**WWDTIF**	**WWDT 比较匹配中断标志** 这位表明WWDT计数器的值与CMPDAT 匹配时产生的中断标志 0 = 无效. 1 = WWDT 计数器的值与CMPDAT 匹配. **注：** 该位写1清除该位

窗看门狗定时 器计数器值寄存器(WWDT_CNT)

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

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		CNTDAT					

位	描述	
[31:6]	Reserved	保留.
[5:0]	CNTDAT	WWDT 计数器的值 寄存器 反映出WWDT计数器当前值， 只读

窗看门狗定时 器复位状态寄存器(WWDT_RSTSTS)

寄存器	偏移地址	R/W	描述	复位值
WWDT_RSTSTS	WWDT_BA+0x18	R	窗看门狗定时 器复位状态寄存器	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						WWDTRF	Reserved

位	描述	
[31:2]	Reserved	保留.
[1]	WWDTRF	WWDT 超时复位标志 表明系统已经发生WWDT超时复位与否 0 = WWDT超时复位没有发生. 1 = WWD超时复位已发生. 注：给 WWDTRFC (WWDT_STATUS [1])写1 清除该位
[0]	Reserved	保留.

6.9 实时时钟 (RTC)

6.9.1 概述

当系统断电时实时时钟 (RTC) 控制器有独立的电源供电。RTC用外部的32.768 kHz晶振(LXT)或内部(LIRC)，提供报时时标信号和闹铃中断，时间及日历等信息的表示格式为BCD 码。可对外接晶振(LXT) 或内部振荡器 (LIRC)的频率精度进行数字频率补偿.

RTC控制器也提供32个字节的备用寄存器用于存储用户的重要信息

唤醒信号可以用于将系统从空闲模式，掉电模式或深度掉电模式唤醒。

6.9.2 特性

- 支持时间计数器 RTC_TIME (时, 分, 秒) 和日期 计数器 RTC_CAL (年, 月, 日) RTC 时间和日期校对
- 支持闹铃时间(时, 分, 秒)和日期 (年, 月, 日) 设置 RTC_TALM 和 RTC_CALM
- 可选择12-小时或 24-小时制式,参看RTC_CLKFMT寄存器
- 支持闰年自动识别,参看RTC_LEAPYEAR寄存器
- 支持 星期计数，参看RTC_WEEKDAY寄存器
- 支持RTC时钟源频率补偿功能，参看RTC_FREQADJ寄存器
- 所有时间、日期的数据格式为BCD码
- 支持 8种 RTC中时标周期间隔选择1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 和 1 秒
- 支持 RTC 报时和闹铃中断
- RTC中断信号支持芯片从空闲和掉电模式唤醒
- 支持32字节备用寄存器用RTC外部电源供电保存

6.9.3 框图

下图描述RTC的架构.

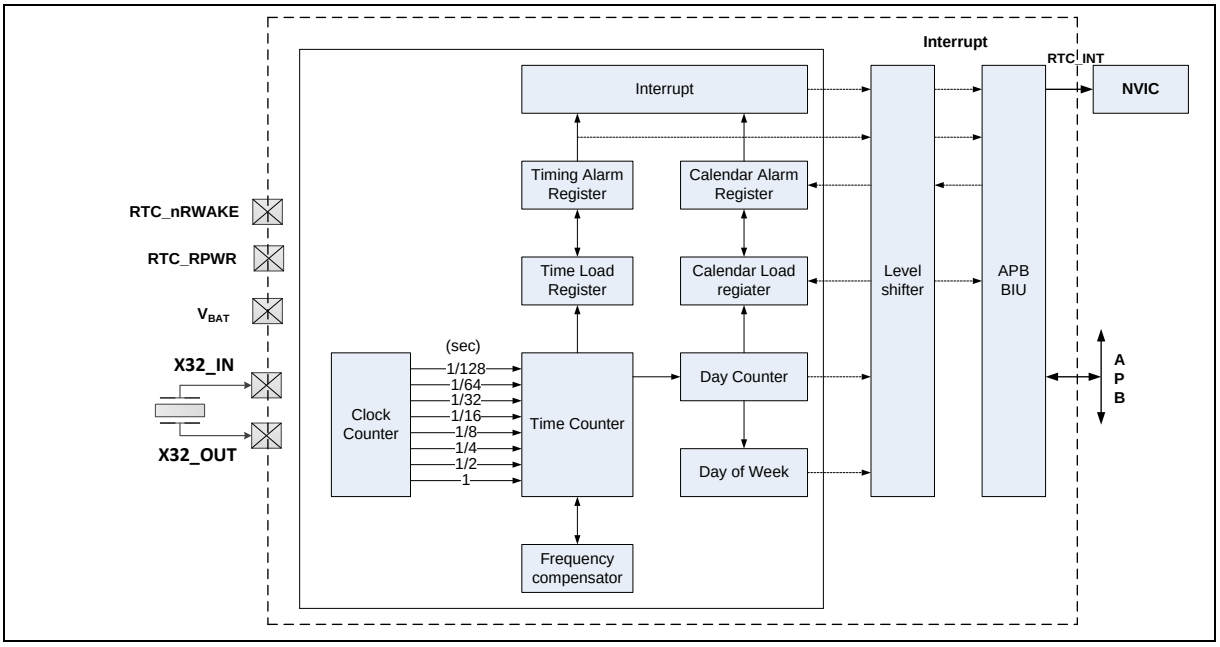


图 6.9-1 RTC模块框图

6.9.4 基本配置

当V_{BAT} 供电为3.3V时RTC的默认时钟源是内部振荡器 (LIRC) 。如果用外部晶振(LXT)，连接晶振到X32_IN 和 X32_OUT,用户设CKSRC (RTC_CLKSRC[0] 值为0关闭 内部振荡器 (LIRC)并通过CBEN (RTC_SET[1]) 的值为1来确认外部振荡器起振正常。

芯片的电源供电可以用RTC_nRWAKE 和 RTC_RPWR 来控制外部电源，当RTC发现RTC_nRWAKE使能可以通过RTC_RPWR来使能外部电源。

6.9.5 功能描述

NUC505在系统上电时,RTC_nRWAKE必须接地才可开启RTC功能。开机后,若RTC_nRWAKE管脚持续为低电平超过一段时间,系统硬件会自动关机,其时间可由POWOFFT (RTC_POWCTL[19:16]) 来设置 ,关机后RTC_nRWAKE管脚必须重新上下电才可重新启动系统。若不希望系统关机,可以在关机前设置POWOFFEN (RTC_POWCTL[2])为0来禁止RTC硬件关机功能。

6.9.5.1 RTC 初始化

当 RTC 模块上电后，RTC 处于 reset 状态。用户需要写入数据 (0x a5eb1357) 到RTC 初始化寄存器RTC_INIT (INIR [31:0])中，让RTC退出复位状态。一旦RTC_INIT被写入 0xa5eb1357，RTC 将会永远处于Active状态。用户可以读Active位 (INIR[0]) 来判断RTC是处于Active状态还是复位状态。

6.9.5.2 访问RTC 寄存器

由于RTC 时钟和系统时钟的不同，当用户对任一 RTC 寄存器进行写入时，必须等待2个 RTC 时钟

周期，寄存器内的值才会被更新。因此用户要等到标志位RTC_SYNC (SYNC[0])置1时再写另一寄存器。.

6.9.5.3 RTC 读写 使能

RWEN (RTC_RWEN[15:0]) 是控制读/写访问的寄存器对RTC有写保护功能，如果RTC_RWEN[15:0] 写入0xa965, 用户可以读访问标志位RWENF (RTC_RWEN[16]) 自动置1，来确认RTC 寄存器可以读写。

6.9.5.4 时钟频率补偿

RTC的时钟源可能不是精确的32768 Hz，RTC_FREQADJ (频率补偿寄存器) 允许软件对RTC时钟输入进行数字补偿。

按照以下例子进行RTC时钟补偿.

例:

设 ADJTRG (RTC_FREQADJ[31]) =1.

等 ADJTRG (RTC_FREQADJ[31]) 被清除

读 CALCNT (RTC_CALCNT[31:0])

通过RTC_CALCNT的值计算RTC时钟,

RTC_Clock_Rate = (PCLK/RTC_CALCNT)*32768

RTC_FREQADJ[23:8] = INTEGER = RTC_Clock_Rate（整数部分）-1

RTC_FREQADJ[7:0] = FRACTION = (RTC_Clock_Rate（小数部分）)*60-1

如果 RTC_CALCNT=50103132,就

RTC_Clock_Rate = (50000000/50103132)*32768=32700.5505

RTC_FREQADJ[23:8] = INTEGER =32699

RTC_FREQADJ[7:0] = FRACTION = 0.5505*60-1=32.03

FRACTION (RTC_FREQADJ[7:0] 应该设置32.03的近似值32.

RTC_FREQADJ[23:0] 寄存器设置 0x7FBB20

6.9.5.5 时间和日期计数器

RTC_TIME 和 RTC_CAL 是用户用来加载时间和日期的. RTC_TALM 和 RTC_CALM 是用于设置闹铃的，它们都是BCD格式.

6.9.5.6 12/24 小时制式选择

设置 24HEN (RTC_CLKFMT[0])进行12/24制式选择.

当 RTC 运行12-小时制式, RTC_TIME[21] (TENHR[1:0]高位)表示 AM/PM. (如果 RTC_TIME[21]是1,代表是PM.)

24-小时制式 (24HEN = 1)	24-小时制式 (24HEN = 1)	24-小时制式 (24HEN = 0)	12-小时制式 (24HEN = 0) (PM + 20)
0x00	0x12	0x12 (AM12)	0x32 (PM12)
0x01	0x13	0x01 (AM01)	0x21 (PM01)
0x02	0x14	0x02 (AM02)	0x22 (PM02)
0x03	0x15	0x03 (AM03)	0x23 (PM03)
0x04	0x16	0x04 (AM04)	0x24 (PM04)
0x05	0x17	0x05 (AM05)	0x25 (PM05)
0x06	0x18	0x06 (AM06)	0x26 (PM06)
0x07	0x19	0x07 (AM07)	0x27 (PM07)
0x08	0x20	0x08 (AM08)	0x28 (PM08)
0x09	0x21	0x09 (AM09)	0x29 (PM09)
0x10	0x22	0x10 (AM10)	0x30 (PM10)
0x11	0x23	0x11 (AM11)	0x31 (PM11)

6.9.5.7 星期计数器

RTC控制器提供星期计数WEEKDA (RTC_WEEKDAY[2:0]). 值 0 到 6 表示星期日到星期六。

6.9.5.8 报时周期中断

有8种报时时标周期选择 1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 和 1 秒， 通过 TICK (RTC_TICK[2:0])进行设置. 通过TICKIEN (RTC_INTEN[1]) 置 1使能报时周期中断, 中断的周期时间间隔通过RTC_TICK[2:0] 选择.

6.9.5.9 闹铃中断

当RTC_TIME 和RTC_CAL中的值等于RTC_TALM和RTC_CALM中的设定值，如果此时闹铃中断已设为使能 (ALMIEN (RTC_INTEN[0])=1)，则闹铃中断标志ALMIF (RTC_INTSTS[0])被置位同时发出闹铃中断请求。

RTC控制器提供时间屏蔽寄存器（RTC_TAMSK）和日历闹铃屏蔽寄存器（RTC_CAMSK），用于屏蔽指定数字并产生周期中断而无需改动每个闹铃中断服务程序中的闹铃匹配条件除了改变匹配条件RTC_TALM 和 RTC_CALM的值.

6.9.5.10 应用笔记

当系统电源关闭但RTC电源供电时，数据存贮在 RTC 寄存器中不会丢失除了 RTC_INTEN 和 RTC_INTSTS.因为RTC时钟和系统时钟频率的不同，当写一个新的数据到任一寄存器时，要等2个RTC时钟最少 (60us)才能更新，所以程序编写要考虑时序RTC_CLKFMT，RTC_TALM和RTC_TIME.

此外，用户必须想到RTC方面加载数据无法确认边界时间，RTC自己没有办法确认星期几和日期之

间是否合理。

1. RTC_TALM, RTC_CALM, RTC_TIME and RTC_CAL 寄存器都是BCD格式.
2. 用户一定要自己确认写给RTC的值是不是合理的，例：加载RTC_CAL 201a (year), 13 (month), 00 (day), RTC_CAL 没有办法确认 RTC_WEEKDAY, 等等.
3. 上电后寄存器的值以下表描述:

寄存器	复位值
RTC_RWEN	0
RTC_CAL	05/1/1 (年/月/日)
RTC_TIME	00:00:00 (时 : 分 : 秒)
RTC_CALM	00/00/00 (年/月/日)
RTC_TALM	00:00:00 (时 : 分 : 秒)
RTC_CLKFMT	1 (24 hr 模式)
RTC_WEEKDAY	6 (星期六)
RTC_INTEN	0
RTC_INTSTS	0
RTC_LEAPYEAR	0
RTC_TICK	0
RTC_POWCTL	50000

4. RTC_CAL 和 RTC_TALM, 仅 2位 BCD数据来表示 “年”. 2 位BCD 数xy 表示20xy, 而不是表示19xy或21xy.
5. 例12-小时制式设置

如果当前 RTC时间是 PM12:59:30 用 12-小时制式设置，RTC_TIME设置:

RTC_TIME[21:16]: 0x32 (0x12+0x20)

TENHR (RTC_TIME[21:20]) 是0x3, HR (RTC_TIME[19:16]) 是 0x2.

RTC_TIME[14:8]: 0x59

TENMIN (RTC_TIME[14:12]) 是 0x5, MIN (RTC_TIME[11:8]) 是 0x9.

RTC_TIME[6:0]: 0x30

TENSEC (RTC_TIME[6:4]) 是 0x3, SEC (RTC_TIME[3:0])是 0x0.

6.9.6 系统电源控制流程

6.9.6.1 正常系统电源控制流程

机器电源控制On/Off 流程如下.

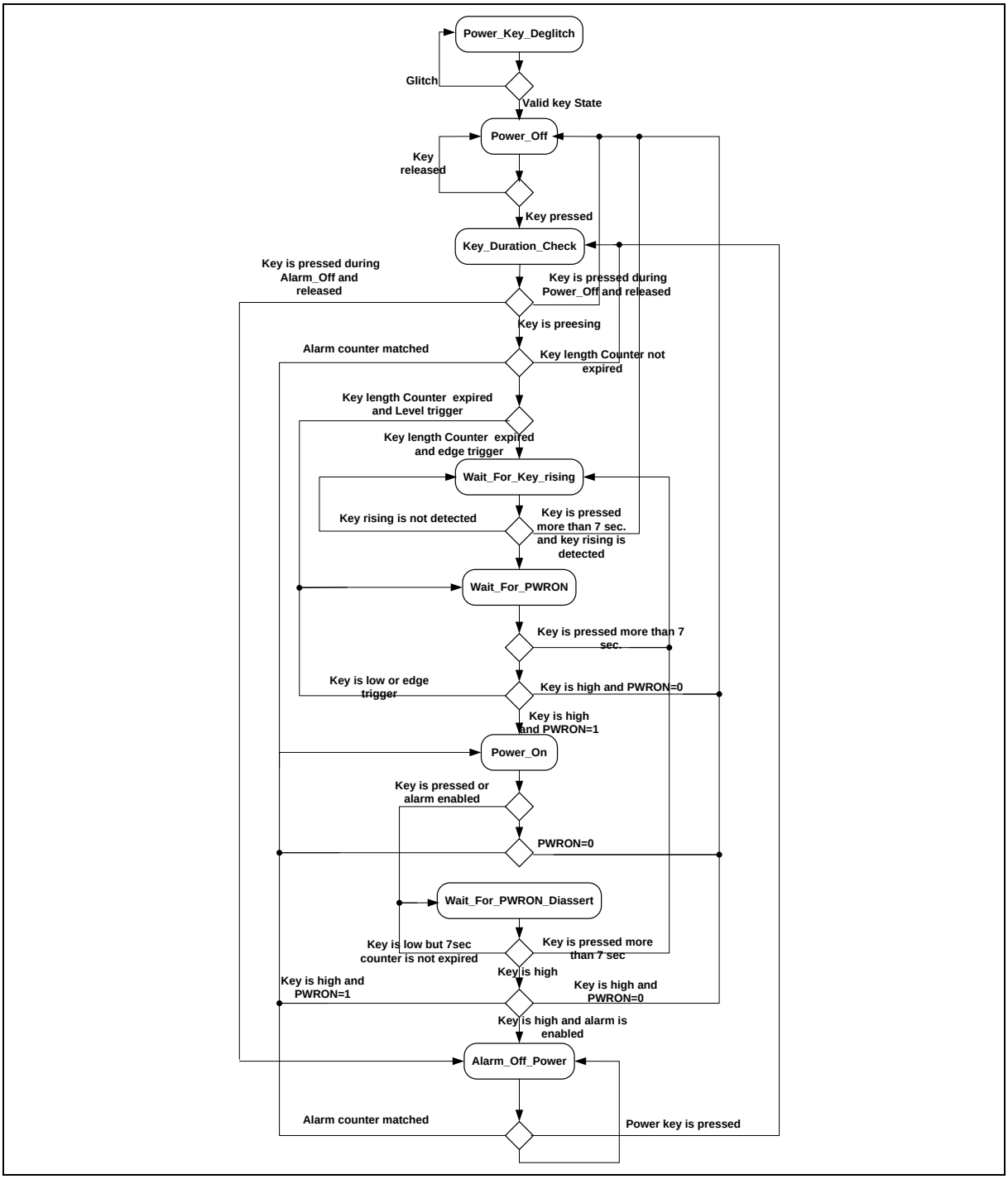


图 6.9-2 系统电源控制流程图

6.9.6.2 强制关系统电源流程

RTC 支持硬件自动关闭电源功能和软件关闭电源功能。 对于硬件关闭功能，RTC能使能和关闭POWOFFEN (RTC_POWCTL[2])， 用户按一会电源键关闭系统电源， POWOFFT (RTC_POWCTL[19:16])来设置些功能。

从设置 POWOFFT (RTC_pOWCTL[19:16])的值和按下电源按键时间如下.

Key_Pressed_Period_To_Power_Off=(POWOFFT+3) 秒.

硬件关闭电源的时序如下.

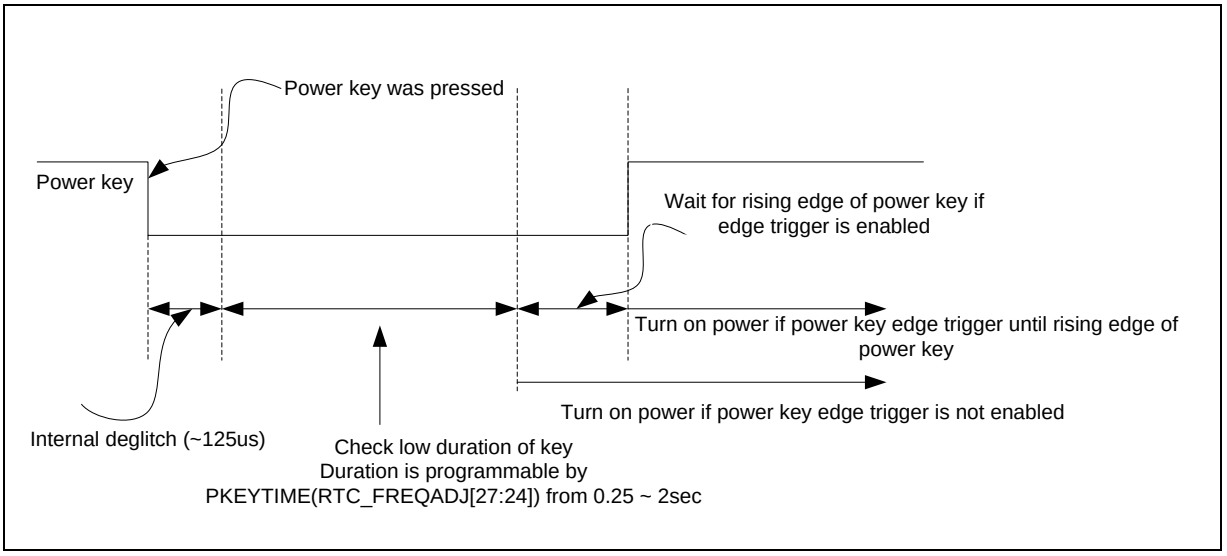


图 6.9-3 电源键侦测

6.9.7 寄存器 映射

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移地址	R/W	描述	复位值
RTC 基地址: **RTC_BA = 0x400E_7000**				
RTC_INIT	RTC_BA+0x000	R/W	RTC初始化寄存器	0x0000_0000
RTC_RWEN	RTC_BA+0x004	R/W	RTC 访问使能寄存器	0x0000_0000
RTC_FREQADJ	RTC_BA+0x008	R/W	RTC 频率补偿寄存器	0x007F_FF00
RTC_TIME	RTC_BA+0x00C	R/W	RTC时间载入寄存器	0x0000_0000
RTC_CAL	RTC_BA+0x010	R/W	RTC日期载入寄存器	0x0005_0101
RTC_CLKFMT	RTC_BA+0x014	R/W	RTC时间制式选择寄存器	0x0000_0001
RTC_WEEKDAY	RTC_BA+0x018	R/W	RTC星期寄存器	0x0000_0006
RTC_TALM	RTC_BA+0x01C	R/W	RTC 闹铃时间寄存器	0x0000_0000
RTC_CALM	RTC_BA+0x020	R/W	RTC 闹铃日期 寄存器	0x0000_0000
RTC_LEAPYEAR	RTC_BA+0x024	R	RTC 闰年 寄存器	0x0000_0000
RTC_INTEN	RTC_BA+0x028	R/W	RTC中断使能 寄存器	0x0000_0000
RTC_INTSTS	RTC_BA+0x02C	R/W	RTC中断状态寄存器	0x0000_0000
RTC_TICK	RTC_BA+0x030	R/W	RTC 报时时标 寄存器	0x0000_0000
RTC_POWCTL	RTC_BA+0x034	R/W	RTC 电源控制 寄存器	0x0005_0000
RTC_SET	RTC_BA+0x038	R/W	RTC设置寄存器	0x0000_0000
RTC_CLKSRC	RTC_BA+0x03C	R/W	RC 振荡器设置寄存器	0x0000_0001
RTC_CALCNT	RTC_BA+0x040	R	RC 振荡器校验 寄存器	0x0000_0000
RTC_SYNC	RTC_BA+0x044	R	RTC 寄存器写同步	0x0000_0000
RTC_SPR0	RTC_BA+0x054	R/W	RTC 备用 寄存器 0	0x0000_0000
RTC_SPR1	RTC_BA+0x058	R/W	RTC备用寄存器 1	0x0000_0000
RTC_SPR2	RTC_BA+0x05C	R/W	RTC备用寄存器 2	0x0000_0000
RTC_SPR3	RTC_BA+0x060	R/W	RTC备用寄存器 3	0x0000_0000
RTC_SPR4	RTC_BA+0x064	R/W	RTC备用寄存器 4	0x0000_0000
RTC_SPR5	RTC_BA+0x068	R/W	RTC备用寄存器 5	0x0000_0000
RTC_SPR6	RTC_BA+0x06C	R/W	RTC备用寄存器 6	0x0000_0000
RTC_SPR7	RTC_BA+0x070	R/W	RTC备用寄存器 7	0x0000_0000

6.9.8 寄存器 描述

RTC 初始寄存器 (RTC_INIT)

寄存器	偏移地址	R/W	描述	复位值
RTC_INIT	RTC_BA+0x000	R/W	RTC初始寄存器	0x0000_0000

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

位	描述	
[31:1]	INIT/STS	**RTC 初始化(写)** 当 RTC 模块上电后，RTC 处于复位状态。用户需要写入数据 (0xa5eb1357) 到 INIR 使得 RTC 离开复位状态。一旦 INIR 被写入 0xa5eb1357，RTC 将会一直处于非复位状态。 INIR 是只写，读取该位域将总是返回 0”. **RTC 初始化状态 (读)** [31:8]: INIT[31:8] [7:5]: RTC 按键的内部状态 [4]: 电源键的状态, 0:按下 1:弹起 [3]: pwr_key_off电源关闭的状态 [2]:电平移动复位 [1]: 电平移动 使能
[0]	ACTIVE	**RTC运行状态 (只读)** 0 = RTC复位状态. 1 = RTC正常运行状态.

RTC访问使能 寄存器 (RTC_RWEN)

寄存器	偏移地址	R/W	描述	复位值
RTC_RWEN	RTC_BA+0x004	R/W	RTC 访问 使能 寄存器	0x0000_0000

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

位	描述	
[16]	RWENF	**RTC 寄存器 访问使能 标志(只读)** 0 = RTC 寄存器 访问关闭. 1 = RTC 寄存器 访问使能. **注:** RTC_RWEN[15:0] 寄存器写入 0xA965后该位被置位，当RTC_RWEN[15:0]写入的不是0xA965该位被清除.
[15:0]	RWEN	**RTC 寄存器 访问 使能口令 (R/W)** 0xA965 =访问口令. 其它 =访问关闭.

RTC频率补偿寄存器 (RTC_FREQADJ)

寄存器	偏移地址	R/W	描述	复位值
RTC_FREQADJ	RTC_BA+0x008	R/W	RTC频率补偿 寄存器	0x007F_FF00

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

位	描述	
[31]	ADJTRG	**RTC时钟校准控制** 这校正是高时校准机构运行如果是低将自动去做RTC_CALCNT 寄存器计数有效的值. 0 = RTC 时钟校准机构关闭 1 = RTC 时钟校准机构使能.
[30]	Reserved	保留.
[29:28]	Reserved	保留.
[27:24]	PKEYTIME	**电源按键按下接通的最短时间** 电源按键按下的最短时间 = 0.25*(PKEYTIME+1) 秒.
[23:8]	INTEGER	**整数部分** 实际振荡器频率 = (RTC时钟比率的整数部分 -1).
[7:6]	Reserved	保留.
[5:0]	FRACTION	**小数部分** 公式 = (RTC时钟比率的小数部分) x 60-1. RTC_FREQADJ必须以16进制方式表示.

RTC 时间加载寄存器 (RTC_TIME)

寄存器	偏移地址	R/W	描述	复位值
RTC_TIME	RTC_BA+0x00C	R/W	RTC时间加载 寄存器	0x0000_0000

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

位	描述	
[31:22]	Reserved	保留.
[21:20]	TENHR	小时的十位数
[19:16]	HR	小时的个位数
[15]	Reserved	保留.
[14:12]	TENMIN	分钟的十位数
[11:8]	MIN	分钟的个位数
[7]	Reserved	保留.
[6:4]	TENSEC	秒的十位数
[3:0]	SEC	秒的个位数

注: RTC_TIME是BCD格式RTC没有确认值合理性机制.

RTC 日期加载 寄存器 (RTC_CAL)

寄存器	偏移地址	R/W	描述	复位值
RTC_CAL	RTC_BA+0x010	R/W	RTC 日期加载 寄存器	0x0005_0101

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

位	描述	
[31:24]	Reserved	保留.
[23:20]	TENYEAR	年的十位数
[19:16]	YEAR	年的个位数
[15:13]	Reserved	保留.
[12]	TENMON	月的十位数
[11:8]	MON	月的个位数
[7:6]	Reserved	保留.
[5:4]	TENDAY	日的十位数
[3:0]	DAY	日的个位数

注: RTC_CAL是BCD格式RTC没有确认值合理性机制.

RTC 时间制式寄存器 (RTC_CLKFMT)

寄存器	偏移地址	R/W	描述	复位值
RTC_CLKFMT	RTC_BA+0x014	R/W	RTC 时间制式寄存器	0x0000_0001

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							24HEN

位	描述	
[31:1]	**Reserved**	保留.
[0]	**24HEN**	**24 / 12小时制式** 表明 RTC_TIME 和 RTC_TALM 是24小时制式或12小时制式 0 = 12-小时制式用 AM 和 PM指示. 1 = 24-小时制式

RTC星期 寄存器 (RTC_WEEKDAY)

寄存器	偏移地址	R/W	描述	复位值
RTC_WEEKDAY	RTC_BA+0x018	R/W	RTC星期 寄存器	0x0000_0006

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					WEEKDAY		

位	描述	
[31:3]	Reserved	保留.
[2:0]	WEEKDAY	星期 寄存器 000 = 星期日 001 = 星期一 010 = 星期二 011 = 星期三 100 = 星期四 101 = 星期五. 110 = 星期六 111 = 保留

RTC 闹铃时间 寄存器 (RTC_TALM)

寄存器	偏移地址	R/W	描述	复位值
RTC_TALM	RTC_BA+0x01C	R/W	RTC闹铃时间寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
MSKHR	Reserved	TENHR		HR			
15	14	13	12	11	10	9	8
MSKMIN	TENMIN			MIN			
7	6	5	4	3	2	1	0
MSKSEC	TENSEC			SEC			

位	描述	
[31:24]	**Reserved**	保留.
[23]	**MSKHR**	**闹铃小时屏蔽** 0 = 有效. 1 = 屏蔽.
[22]	**Reserved**	保留.
[21:20]	**TENHR**	**闹铃小时十位数 (0-2)** 当 RTC是12小时制式时, RTC_TIME[21] (高位 TENHR[1:0]) 表示 AM/PM. (如果 RTC_TIME[21] 是 1,表示 PM.)
[19:16]	**HR**	**闹铃小时个位数**(0-9)
[15]	**MSKMIN**	**闹铃分钟屏蔽** 0 = 有效. 1 = 屏蔽.
[14:12]	**TENMIN**	**闹铃分钟十位数**(0-5)
[11:8]	**MIN**	**闹铃分钟个位数**(0-9)
[7]	**MSKSEC**	**闹铃秒屏蔽** 0 = 有效. 1 = 屏蔽.
[6:4]	**TENSEC**	**闹铃秒十位数**(0-5)
[3:0]	**SEC**	**闹铃秒个位数**(0-9)

注: RTC_TALM 是BCD格式RTC没有确认值合理性机制.

RTC 闹铃日期寄存器 (RTC_CALM)

寄存器	偏移地址	R/W	描述	复位值
RTC_CALM	RTC_BA+0x020	R/W	RTC 闹铃日期寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MSKWEEKDAY	WEEKDAY			Reserved			MSKYEAR
23	22	21	20	19	18	17	16
TENYEAR				YEAR			
15	14	13	12	11	10	9	8
MSKMON	Reserved		TENMON	MON			
7	6	5	4	3	2	1	0
MSKDAY	Reserved	TENDAY		DAY			

位	描述	
[31]	**MSKWEEKDAY**	**闹铃星期屏蔽** 0 = 有效. 1 = 屏蔽
[30:28]	**WEEKDAY**	星期数
[27:25]	**Reserved**	保留.
[24]	**MSKYEAR**	**闹铃年屏蔽** 0 = 有效. 1 = 屏蔽.
[23:20]	**TENYEAR**	闹铃年的十位数(0-9)
[19:16]	**YEAR**	闹铃年的个位数(0-9)
[15]	**MSKMON**	**闹铃月屏蔽** 0 = 有效. 1 = 屏蔽.
[14:13]	**Reserved**	保留.
[12]	**TENMON**	闹铃月的十位数(0-1)
[11:8]	**MON**	闹铃月的个位数(0-9)
[7]	**MSKDAY**	**闹铃日 屏蔽** 0 = 有效. 1 = 屏蔽.
[6]	**Reserved**	保留.
[5:4]	**TENDAY**	闹铃日的十位数(0-3)

[3:0]	**DAY**	闹铃日的个位数(0-9)

注:RTC_CALM是BCD格式RTC没有确认值合理性机制.

1. 当 RTC_CALM 和 RTC_TALM 都是屏蔽，闹铃将自动屏蔽

RTC 闰年 寄存器 (RTC_LEAPYEAR)

寄存器	偏移地址	R/W	描述	复位值
RTC_LEAPYEAR	RTC_BA+0x024	R	RTC闰年 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							LEAPYEAR

位	描述	
[31:1]	Reserved	保留.
[0]	LEAPYEAR	闰年寄存器 (只读) 0 = 不是闰年. 1 = 闰年.

RTC中断使能 寄存器 (RTC_INTEN)

寄存器	偏移地址	R/W	描述	复位值
RTC_INTEN	RTC_BA+0x028	R/W	RTC中断使能 寄存器	0x0000_0000

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

位	描述	
[31:2]	Reserved	保留.
[3]	RALMIEN	相对时间闹铃中断 使能控制 0 = RTC相对时间闹铃中断关闭. 1 = RTC相对时间闹铃中断使能
[2]	PKEYIEN	电源开关中断 使能 控制 0 = 电源开关按下中断关闭. 1 = 电源开关按下中断 使能.
[1]	TICKIEN	报时时标 中断使能 控制 0 = RTC 时标中断关闭. 1 = RTC 时标中断使能.
[0]	ALMIEN	闹铃中断 使能 控制 0 = RTC闹铃中断关闭. 1 = RTC闹铃中断 使能.

RTC 中断标志寄存器 (RTC_INTSTS)

寄存器	偏移地址	R/W	描述	复位值
RTC_INTSTS	RTC_BA+0x02C	R/W	RTC中断标志寄存器	0x0000_0000

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

位	描述	
[31:2]	**Reserved**	保留.
[3]	**RELALMIF**	**RTC相对时间闹铃中断指示** 0 =相对时间闹铃中断没有发生. 1 =相对的时间计数和日期计数达到指定的 RTC_TALM 和 RTC_CALM的值 . RTC闹铃中断是有效的. **注:** RTC中断发生后可以软件清除该位
[2]	**POWKEYIF**	**电源开关中断标志** 当检测RTC电源键 (RTC_nRWAKE) 被按下 , POWKEYIF (RTC_INTSYS[2]) 被置1 0 = 电源开关中断从没有发生. 1 = 电源开关有效. **注:**在RTC 中断发生后软件清除该位
[1]	**TICKIF**	**RTC 报时时标中断标志** 当 RTC 时钟时标发生，该位将置1并产生一个中断信号;当芯片在掉电模式下，如果RTC时标中断TICKIEN (RTC_INTEN[1])使能，该位置1将会从掉电状态唤醒芯片. 0 = 时标条件没有发生. 1 = 时标条件发生. **注:** 该位写1清除该位.
[0]	**ALMIF**	**RTC 闹铃中断标志** 当 RTC计数器一旦到达RTC_TIME 和RTC_CAL 的值达到了闹铃设定寄存器RTC_TALM 和 RTC_CALM 的值，如果RTC闹铃使能位ALMIEN (RTC_INTEN[0])使能，该位将被置为1并产生一个中断信号；当芯片工作在掉电模式下，将会唤醒芯片 0 = 闹铃没有达到条件. 1 =闹铃达到条件. **注:** : 该位写1清除该位.

RTC报时时标 寄存器 (RTC_TICK)

寄存器	偏移地址	R/W	描述	复位值
RTC_TICK	RTC_BA+0x030	R/W	RTC 报时时标 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					TICKSEL		

位	描述	
[31:3]	Reserved	保留.
[2:0]	TICKSEL	**时标周期 寄存器** 用于选择 RTC中断请求时标周期时间. 000 =时标是1 秒. 001 =时标是1/2 秒. 010 =时标是1/4 秒. 011 =时标是1/8 秒. 100 =时标是1/16 秒. 101 =时标是1/32 秒. 110 =时标是1/64 秒. 111 =时标是1/28 秒.

RTC电源控制寄存器 (RTC_POWCTL)

寄存器	偏移地址	R/W	描述	复位值
RTC_POWCTL	RTC_BA+0x034	R/W	RTC 电源控制 寄存器	0x0005_0000

31	30	29	28	27	26	25	24
RALMTIME							
23	22	21	20	19	18	17	16
RALMTIME				POWOFFT			
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
POWKEY	Reserved	EDGE_TRIG	RALMIEN	ALMIEN	POWOFFEN	SWPOWOFF	POWEN

位	描述	
[31:20]	**RALMTIME**	**相对闹铃时间周期 (单位秒)** 表示相对闹铃的时间周期. 最大值为 1800. 当 RALMIEN (RTC_INTEN[3]) = 0 , 该位清 0.
[19:16]	**POWOFFT**	**电源清除周期** 电源键按下后到内核电源关闭的时间周期，.时间是秒数，默认为5秒
[15:8]	**Reserved**	保留.
[7]	**POWKEY**	**电源键状态** 0 = 电源键按下状态为低. 1 = 电源键状态是高.
[6]	**Reserved**	保留.
[5]	**EDGE_TRIG**	**电源键触发模式** 0 = LEVEL TRIGGER, RTC电源键长按有效 1 = EDGE TRIGE, RTC 电源键按并释放后有效.
[4]	**RALMIEN**	**相对时间闹铃** 0 = 相对时间闹铃关闭. 1 = 相对时间闹铃使能.
[3]	**ALMIEN**	**正常时间闹铃** 0 = 正常时间闹铃关闭. 1 = 正常时间闹铃 使能.

[2]	**POWOFFEN**	**硬件电源清除使能控制** 0 = RTC_RPWR管脚 不受电源键按压影响. 1 = RTC_RPWR管脚被拉低，当电源键按下时间超过POWOFFT的值（单位秒）
[1]	**SWPOWOFF**	**软件电源关闭控制** 如果电源键按下，RTC_RPWR脚能被 该位置高清除，也能被按键清除，如果电源键没有按或没有给该位置1 RTC_RPWR脚被释放. 1 = RTC_RPWR 拉低.
[0]	**POWEN**	**电源开** RTC_RPWR 将变为高状态 当POWEN从0变为 1. **注** : RTC_RPWR 变为低的步骤 • 设 POWEN 位为 0 • POWOFFEN设为1， 电源键按下时间超过 POWOFFT设定的值. RTC 使能 有效后，该位就能得到电源状态.

RTC 设置寄存器 (RTC_SET)

寄存器	偏移地址	R/W	描述	复位值
RTC_SET	RTC_BA+0x038	R/W	RTC 设置寄存器	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			XOUTDAT	XININDAT	IOMSEL	CBEN	Reserved

位	描述	
[31:5]	**Reserved**	保留.
[4]	**XOUTDAT**	**X32_OUT PAD 状态** 当 IOMSEL = 0 (只读) 输入信号.
[3]	**XININDAT**	**X32_IN PAD 状态** 当 IOMSEL = 0 (只读)输入信号.
[2]	**IOMSEL**	**X32_IN 和 X32_OUT PAD 数字信号输入控制** 0 =数字信号输入. 1 = 晶振模式 (默认).
[1]	**CBEN**	**32768 Hz (LXT) 晶振控制** 0 = 晶振关闭. 1 =晶振 使能.
[0]	**Reserved**	保留.

RC 振荡器设置寄存器 RTC_CLKSRC)

寄存器	偏移地址	R/W	描述	复位值
RTC_CLKSRC	RTC_BA+0x03C	R/W	RC振荡器设置寄存器	0x0000_0001

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							CKSRC

位	描述	
[31:1]	Reserved	保留.
[0]	CKSRC	内部RC振荡器设置 0 = 内部RC振荡器关闭. 1 =内部RC振荡器使能.

RC 振荡器校准寄存器 (RTC_CALCNT)

寄存器	偏移地址	R/W	描述	复位值
RTC_CALCNT	RTC_BA+0x040	R	RC振荡器校准寄存器	0x0000_0000

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

位	描述	
[31:0]	CALCNT	**PCLK振荡数据在1Hz周期** 这个数据是RTC时钟分频得到. 可以用于 RTC 时钟真实时钟比率计算.

RTC 寄存器 写同步 寄存器 (RTC_SYNC)

寄存器	偏移地址	R/W	描述	复位值
RTC_SYNC	RTC_BA+0x044	R	RTC 寄存器 写同步	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							SYNC

位	描述	
[31:1]	Reserved	保留.
[0]	SYNC	RTC 寄存器写同步标志查询 0 = 寄存器 不能写. 1 = 寄存器 写完成可以再写

RTC 备用寄存器 0~7

寄存器	偏移地址	R/W	描述	复位值
RTC_SPR0	RTC_BA+0x054	R/W	RTC 备用寄存器 0	0x0000_0000
RTC_SPR1	RTC_BA+0x058	R/W	RTC 备用寄存器 1	0x0000_0000
RTC_SPR2	RTC_BA+0x05C	R/W	RTC 备用寄存器 2	0x0000_0000
RTC_SPR3	RTC_BA+0x060	R/W	RTC 备用寄存器 3	0x0000_0000
RTC_SPR4	RTC_BA+0x064	R/W	RTC 备用寄存器 4	0x0000_0000
RTC_SPR5	RTC_BA+0x068	R/W	RTC 备用寄存器 5	0x0000_0000
RTC_SPR6	RTC_BA+0x06C	R/W	RTC 备用寄存器 6	0x0000_0000
RTC_SPR7	RTC_BA+0x070	R/W	RTC备用寄存器 7	0x0000_0000

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

位	描述	
[31:0]	**RTC_SPRn**	**RTC 备用 寄存器** n = 0~7.

6.10 串口控制器 (UART)

6.10.1 概述

NUC505系列提供3个通用异步串行接口(UART). UART控制器支持标准速度UART，并提供流量控制。UART控制器的接收过程是把外设的串行数据转为并行数据，发送过程是把CPU的并行数据转成串行数据发送出去。每个UART通道支持10种类型的中断. UART控制器还支持IrDA SIR,RS-485和波特率自动测量功能。.

6.10.2 特性

- 全双工，异步通讯口
- 独立的接收/发送16/16(UART0)/ / 64/64(UART1 和 UART2)字节FIFO
- 支持硬件自动流控制(nCTS 和nRTS)仅 UART1 和 UART2
- 接收缓存触发等级的数据长度可设
- 支持每个通道波特率可单独设置
- 支持nCTS和 RX 数据触发唤醒功能
- 支持 8位接收缓存定时溢出检测功能
- 通过设置寄存器DLY (UA_TOR [15:8])，可配置两个数据之间（从上一个stop 位到下一个start 位）的传送时间间隔
- 支持波特率自动侦测
- 支持break error，frame error， parity error和收/发缓冲区溢出检测等功能
- 可编程串行接口特性
  - 数据位长度可设为5~8位
  - 可编程校验，包括奇、偶、无校验，或固定校验位生成和检测
  - 可设置停止位长度为1位，1.5位或2位
- 支持IrDA SIR 功能模式
  - 标准模式下支持3/16位宽功能
- 支持LIN 功能模式(UART1和UART2支持)
  - 支持LIN 主/从模式
  - 传输中支持break生成功能可设
  - 支持接收器break检测功能
- 支持RS-485模式
  - 支持RS-485 9-位模式
  - 支持软硬件控制nRTS管脚，用于控制RS-485传送方向

6.10.3 框图

UART时钟控制和框图分别描述图 6.10-1 和 图 6.10-2.

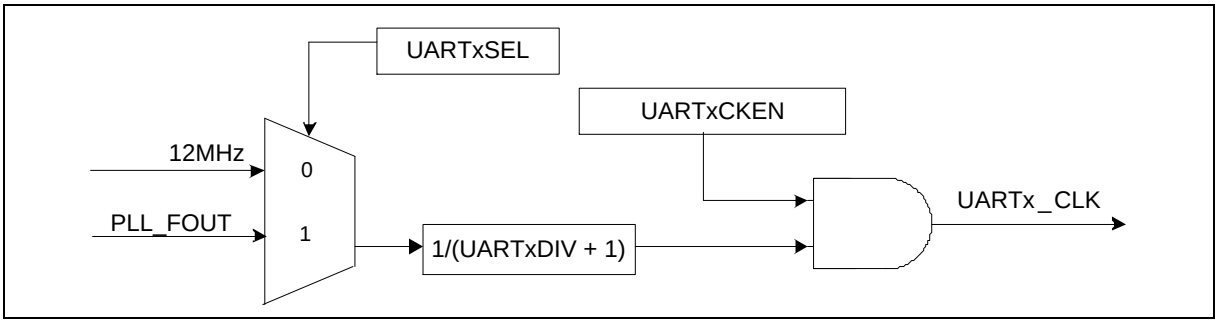


图 6.10-1 UART 时钟控制图

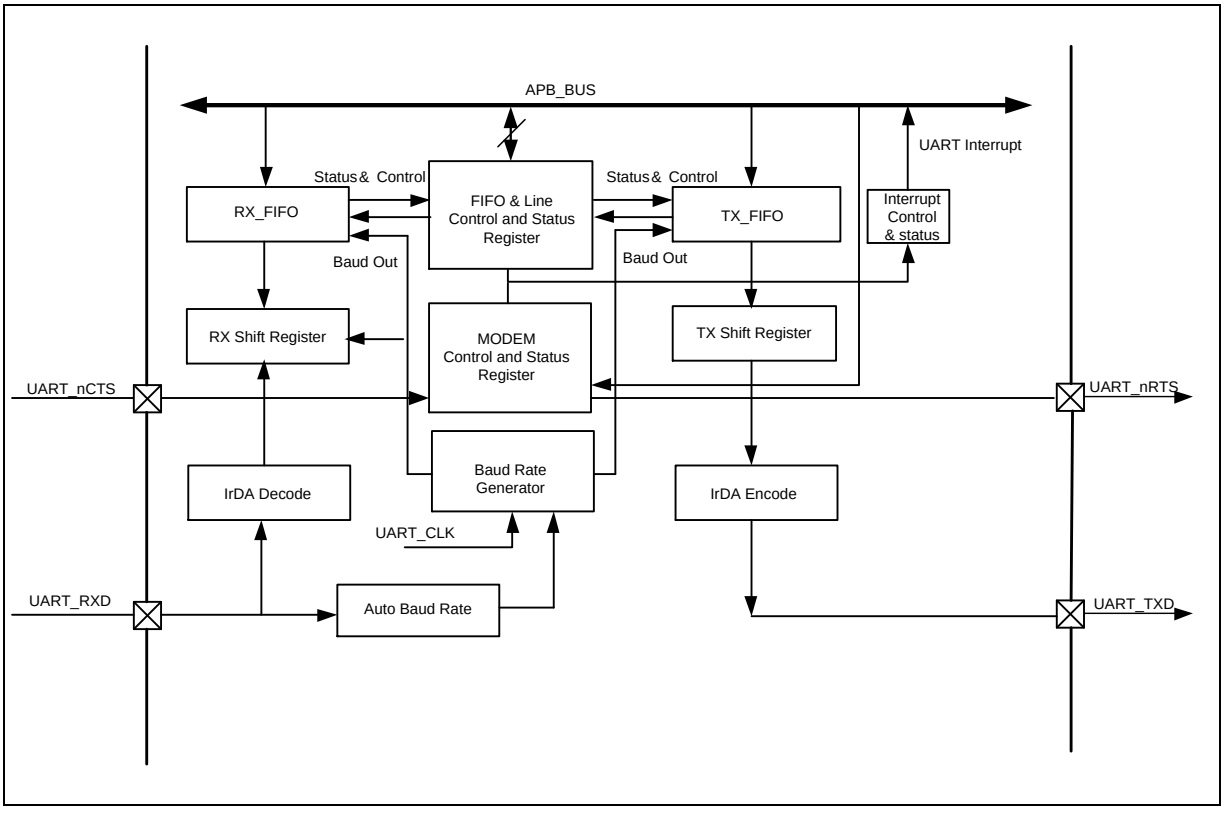


图 6.10-2 UART模块框图

每个模块功能详细描述如下:

TX_FIFO
发送口带有一个16 (UART0) / 64 (UART1和 UART2)字节的FIFO缓冲区以减少CPU中断的频率

RX_FIFO
接收口带有一个16 (UART0) / 64 (UART1和 UART2) 字节的FIFO缓冲区（另加3个出错位，, BIF (UART_FIFOSTS[6]), FEF (UART_FIFOSTS[5]), PEF (UART_FIFOSTS[4])）以减少CPU中断的频率

TX Shift 寄存器（发送移位寄存器）

该模块用于控制把并行数据串行输出

RX Shift 寄存器（接收移位寄存器）

该模块用于控制把串行数据并行输入

Modem 控制和状态 寄存器（Modem控制和状态寄存器）

该寄存器用于控制到Modem或数传机（或类似于Modem的外设）的接口

Baud Rate Generator（波特率发生器.）

通过把输入的外部时钟除频得到期望的波特率。详情请参考波特率公式

IrDA Encode（IrDA编码）

该模块为IrDA编码控制模块:

IrDA Decode（IrDA解码）

该模块为IrDA解码控制模块

FIFO & Line Control and Status 寄存器（FIFO & Line 控制和状态寄存器）
该寄存器组包括FIFO控制寄存器(UART_FIFO)，FIFO状态寄存器(UART_FIFOSTS)和line控制寄存器(UART_LINE)。定时溢出寄存器(UART_TOUT)配置时间溢出中断条件。

Auto-Baud Rate Measurement（波特率自动检测）

该模块用于自动检测波特率

Interrupt Control and Status 寄存器（中断控制和状态寄存器）

一共有10种类型的中断，包括发送FIFO空中断(THERIF)，接收达到阈值(RDAIF)中断，接收状态中断(parity error 或 framing error 或 break中断) (RLSIF)，超时中断(RXTOINT)，缓存错误中断(BUFERRINT) ，和 LIN总线中断(LININT)，数据唤醒中断，nCTS唤醒中断和自动波特率检测完成中断或波特率检测计数器溢出中断。中断使能寄存器(UART_INTEN)用于使能或禁止相应中断，中断状态寄存器(UART_INTSTS)显示哪些中断正在发生。.

6.10.4 基本配置

UART控制器管脚在SYS_GPA_MFPH, SYS_GPB_MFPL和SYS_GPB_MFPH多功能寄存器中配置。

UART控制器时钟分别由各个UART口时钟使能位控制。UAR0TCKEN (CLK_APBCLK[11]) 对应UART0, UART1CKEN(CLK_APBCLK[12]) 对应UART1, UART2CKEN(CLK_APBCLK[13]) 对应UART2。

UART控制器时钟源通过UART0SEL (CLK_CLKDIV3[4])对应UART0，UART1SEL (CLK_CLKDIV3[12]) 对应 UART1 和 UART2SEL (CLK_CLKDIV3[20]) 对应 UART2来选择。

UART控制器时钟预分频通过UART0DIV (CLK_CLKDIV3[3:0]) 对应 UART0, UART1DIV (CLK_CLKDIV3[11:8]) 对应 UART1 和 UART2DIV (CLK_CLKDIV3[19:16]) 对应 UART2.来设置。

UART接口控制器管脚描述如下：

管脚	类型	描述
UART_TXD	输出	UART 发送
UART_RXD	输入	UART 接收
UART_nCTS	输入	UART modem 清 0 发送
UART_nRTS	输出	UART modem 请求发送

表 6.10-1 UART 接口控制管脚

6.10.5 功能 描述

UART控制器支持四种功能模式，包括UART，IrDA，LIN和RS-485模式。用户可以通过对UART_FUNCSEL设置选择功能。四种功能模式将在下面的章节中描述。

6.10.5.1 UART 控制器波特率发生器

UART控制器包含一个可编程波特率发生器，其通过分频器对输入时钟源分频而得到收发数据所需的串行时钟。下表列出了各种条件下的UART波特率计算公式和参数的设置。通过设置相应的参数和寄存器可以得到零误差的波特率。IrDA功能模式，波特率发生器必须是模式0。寄存器表UART_BAUD一栏中将更详细地介绍寄存器内容。共有三种设定模式：设定UART_BAUD[29:28]=00为模式0，设定UART_BAUD[29:28]=10为模式1，设定UART_BAUD[29:28]=11为模式2。

模式	BAUDM1	BAUDM0	波特率公式
Mode 0	0	0	UART_CLK / [16 * (BRD+2)]
Mode 1	1	0	UART_CLK / [(EDIVM1+1) * (BRD+2)], EDIVM1 必须 >= 8
Mode 2	1	1	UART_CLK / (BRD+2). 如果 UART_CLK <= 3*PCLK, BRD 必须 >= 9. 如果 UART_CLK > 3*PCLK, BRD 必须 >= 3*N – 1. N 必须是一个大于或等于 UART_CLK /PCLK的整数. 例,

			如果 3*PCLK < UART_CLK =< 4*PCLK, BRD 必须 >=11. 如果4*PCLK < UART_CLK =< 5*PCLK, BRD 必须 >=14.

表 6.10-2 UART 控制器波特率计算公式

6.10.5.2 UART 控制器自动波特率功能模式

自动波特率功能可以自动测量从UART RX管脚输入数据的波特率。当自动波特率测量结束，测量结果将会放置在BRD (UART_BAUD[15:0]).中。BAUDM1 (UART_BAUD[29]) 和 BAUDM0 (UART_BAUD[28])都被自动设置为1。在自动波特率检测帧中，UART RX数据从起始位到第一个上升沿的时间由ABRD位 (UART_ALTCTL[20:19])时间设定。

通过设置ABRDEN (UART_ALTCTL[18]) 可以使能自动波特率检测功能。在初始阶段，UART RX保持为 1 。一旦检测到下降沿，即为接收到起始位。自动波特率计数器被重启并开始计数。 当检测到第一个上升沿时，自动波特率计数器将停止计数。然后，自动波特率计数器数值除以ABRD位 (UART_ALTCTL[20:19])的结果将会自动载入到BRD(UART_BAUD[15:0])中，而且ABRDEN (UART_ALTCTL[18])会被清零。一旦自动波特率测量结束，ABRDIF (UART_FIFOSTS[1])会被置位。当自动波特率计数器溢出，ABRTOIF (UART_FIFOSTS[2])将会被置位。如果使能ABRIEN (UART_INTEN[18])使能，ABRDIF(UART_FIFOSTS[1]) 或 (UART_FIFOSTS[2])将会引发自动波特率中断ABRIF(UART_ALTCTL[17])产生。

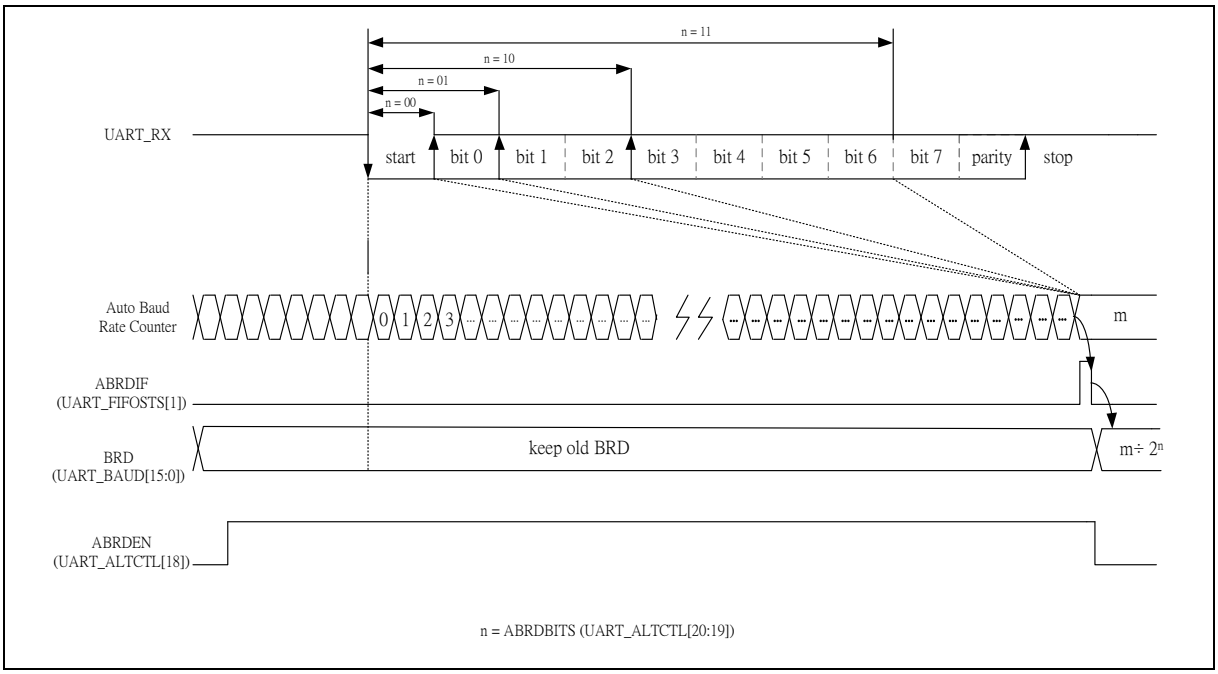


图 6.10-3 自动波特率测量

软件顺序例子:

1. 设置 ABRDBIT (UART_ALTCTL[20:19]) 去测量 UART RX 从开始到第1个数上升沿用的

时间.

2. ABRIEN (UART_INTEN[18]) 使能 自动波特率中断功能.
3. ABRDEN (UART_ALTCTL[18]) 使能自动波特率检测功能.
4. ABRDIF (UART_FIFOSTS[1]) 置1，表示自动波特率测量结束
5. 执行UART发送和接收的动作.
6. ABRDTOIF (UART_FIFOSTS[2])置 1，表示自动波特率计数器溢出.
7. 去第 2步.

6.10.5.3 UART控制器发送延时时间

UART控制器可以通过设置DLY (UART_TOUT [15:8])来控制传输过程两个数据帧之间即上一个数据帧的停止位和下一个数据帧的起始位之间的间隔。单位是波特。延时操作如UART控制器可以通过设置DLY (UART_TOUT [15:8])来控制传输过程两个数据帧之间即上一个数据帧的停止位和下一个数据帧的起始位之间的间隔。单位是波特。延时操作如 图 6.10-4

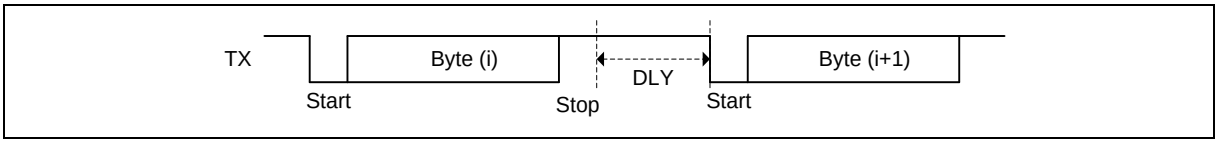


图 6.10-4发送延时操作

6.10.5.4 UART 控制器FIFO控制和状态

UART0/UART1/UART2内置一个16 (UART0) / 64 (UART1 和 UART2)字节的发送FIFO (TX_FIFO)和一个16 (UART0) / 64 (UART1 和 UART2)字节的接收FIFO (RX_FIFO), 通讯中,使用这些收发FIFO可以减少对CPU的中断次数。CPU在操作过程中任何时间都可以读取UART的状态，包括3个错误状态（校验错误，帧错误，间隔错误）如果接收的数据出现校验错误、帧或间隔错误。FIFO控制和状态也支持所有的功能模式，包括UART, IrDA, LIN和RS-485模式

6.10.5.5 UART控制器唤醒功能

UART控制器支持系统唤醒功能。唤醒功能包括nCTS和数据唤醒。 当系统在掉电模式下，nCTS脚或数据都可以唤醒系统。当传入数据唤醒系统后，传入的数据将被保存在FIFO中，并且控制器将自动清除WKDATIEN (UART_INTEN [10])。但是传入数据的第一个字节将会丢失，数据接收从第二个字节开始。下面的框图列举了唤醒功能。

nCTS 唤醒1 (nCTS 从低变高)

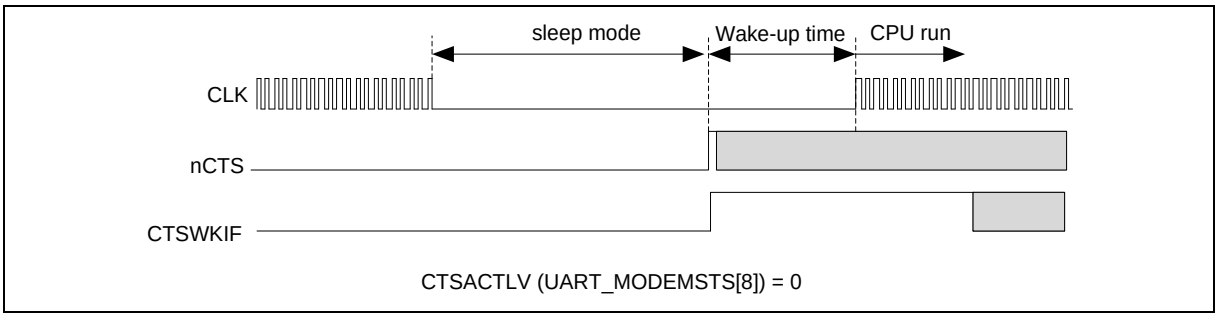


图 6.10-5 UART nCTS 唤醒1

nCTS 唤醒 2 (nCTS 从高到低)

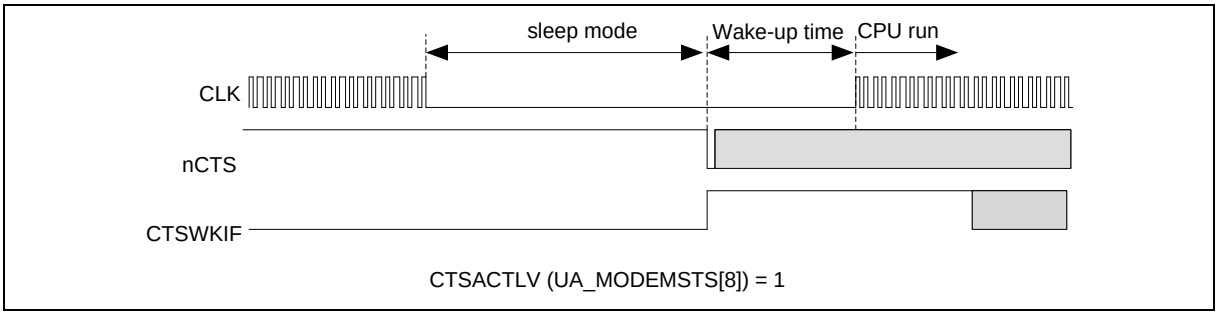


图 6.10-6 UART nCTS 唤醒2

数据唤醒

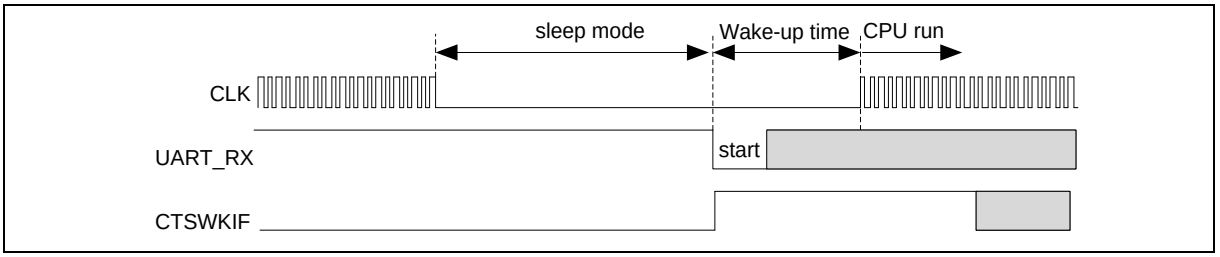


图 6.10-7 UART 数据唤醒

6.10.5.6 UART控制器中断和状态

每个UART控制器支持10种类型的中断，包括:

- 达到接收阈值水平的中断(RDAINT)
- 发送FIFO空中断(THERINT)
- Line状态中断（校验错误，帧错误，间隔错误）(RLSINT)
- MODEM状态中断(MODEMINT)
- 接收缓冲区定时溢出中断(RXTOINT)

- 缓冲区错误中断(BUFERRINT)
- LIN总线中断(LININT)
- nCTS唤醒中断(CTSWKIF)
- 数据唤醒中断(DATWKIF)
- 自动波特率中断(ABRIF)

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

中断源	中断指示	中断使能位	中断标志	清中断标志方法
接收数据空闲中断	RDAINT	RDAIEN	**RDAIF**	Read UART_DAT
发送保持寄存器空中断	THERINT	TJREIEN	**THREIF**	Write UART_DAT
接收Line状态中断	RLSINT	RLSIEN	**RLSIF** = BIF	**Write '1' to** BIF
			RLSIF = FEF	**Write '1' to** FEF
			RLSIF = PEF	**Write '1' to** PEF
			RLSIF = ADDRDETF	**写'1'到** ADDRDETF
Modem状态中断	MODEMINT	MODEMIEN	**MODEMIF** = CTSDETF	**写'1'到** CTSDETF
RX定时溢出中断	RXTOINT	RXTOIEN	**RXTOIF**	Read UART_DAT
缓冲区错误中断	BUFERRINT	BUFERRIEN	**BUFERRIF** = TXOVIF	**写'1'到** TXOVIF
			BUFERRIF = RXOVIF	**写'1'到** RXOVIF
LIN 总线中断	LININT	LIN _IEN	**LINIF =** BRKDETF	**写'1'到** LINIF **and** **写'1'到** BRKDETF
			LINIF = BITEF	**写'1'到** BITEF
			LINIF = SLVIDPEF	**写'1'到** BITEF
			LINIF = SLVHEF	**写'1'到** BITEF

			LINIF = SLVHDETF	**写'1'到** SLVHDETF
nCTS 唤醒中断	N/A	WKCTSIEN	**CTSWKIF**	**写'1'到** CTSWKIF
数据唤醒中断	N/A	WKDATIEN	**DATWKIF**	**写'1'到** DATWKIF
自动波特率中断	N/A	ABRIEN	**ABRIF =** ABRDIF	**写'1'到** ABRDIF
			ABRIF = ABRDTOIF	**写'1'到** ABRDTOIF.

表 6.10-3 UART控制器中断源和标志列表

6.10.5.7 UART 功能模式

UART控制器提供了UART功能（用户须设置FUNCSEL (UART_FUNCSEL [1:0]) 设置为 '00' 来使能UART功能模式）。UART 波特率最高速度是1 Mbps 。.

UART为全双工异步通讯接口。收发各包含一个16字节的FIFO缓冲区。用户可以设置接收时的FIFO触发阈值以及定时溢出检测时间。发送数据帧间（即从上一帧停止位到下一帧起始位）时间间隔通过DLY (UA_TOR [15:8])位可设。UART支持硬件自动流控功能，且nRTS流控触发电平可设。如果RX FIFO中数据字节数大于或等于RTSTRGLV (UART_FIFO[19:16])，nRTS将被释放。

UART线控功能

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

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

表 6.10-4 UART线控的数据位和停止位长度设置

校验类型	SPE (UART_LINE[5])	EPE (UART_LINE[4])	PBE (UART_LINE[3])	描述

无校验	x	x	0	无奇偶校验位输出
奇校验	0	0	1	奇校验位的计算方法是把数据流中的所有的1相加，使得包括校验位在内,1的总数为奇数个.
偶校验	0	1	1	偶校验位的计算方法是把数据流中的所有的1相加，使得包括校验位在内,1的总数为偶数个
奇偶校验位强制置1	1	0	1	奇偶校验位总是逻辑1 不管数据位中1的个数是奇数还是偶数，奇偶校验位永远都是逻辑1
奇偶校验位强制置0	1	1	1	奇偶校验位总是逻辑0 不管数据位中1的个数是奇数还是偶数，奇偶校验位永远都是逻辑0

表 6.10-5 UART线控校验设置

UART自动流控功能

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

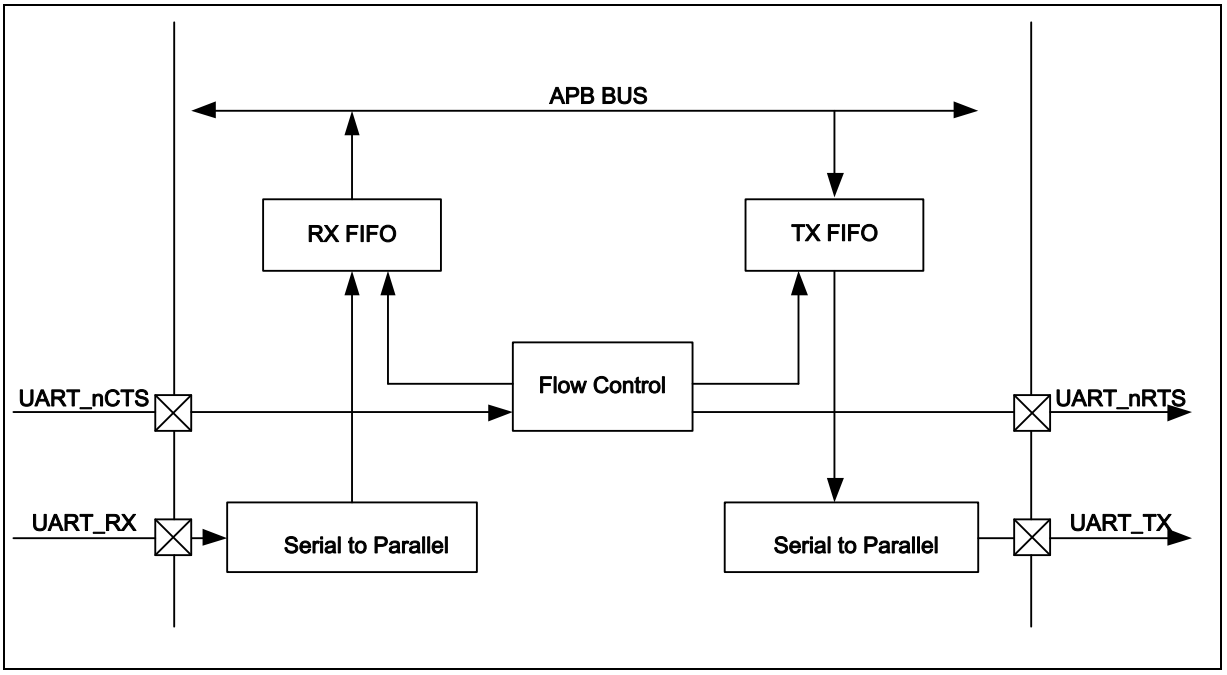


图 6.10-8自动流控框图

以下框图展示了UART CTS自动流控功能的框图。用户必须要先设置ATOCTSEN (UART_INTEN [13])以使能CTS自动流控功能。CTSACTLV (UART_MODEMSTS [8])位可以设置CTS脚输入的有

效状态。当CTS脚上任何电平变化将导致CTSDETF (UART_MODEMSTS[0])位被置1，然后TX FIFO将自动将数据发送到TX脚,并传送出去。

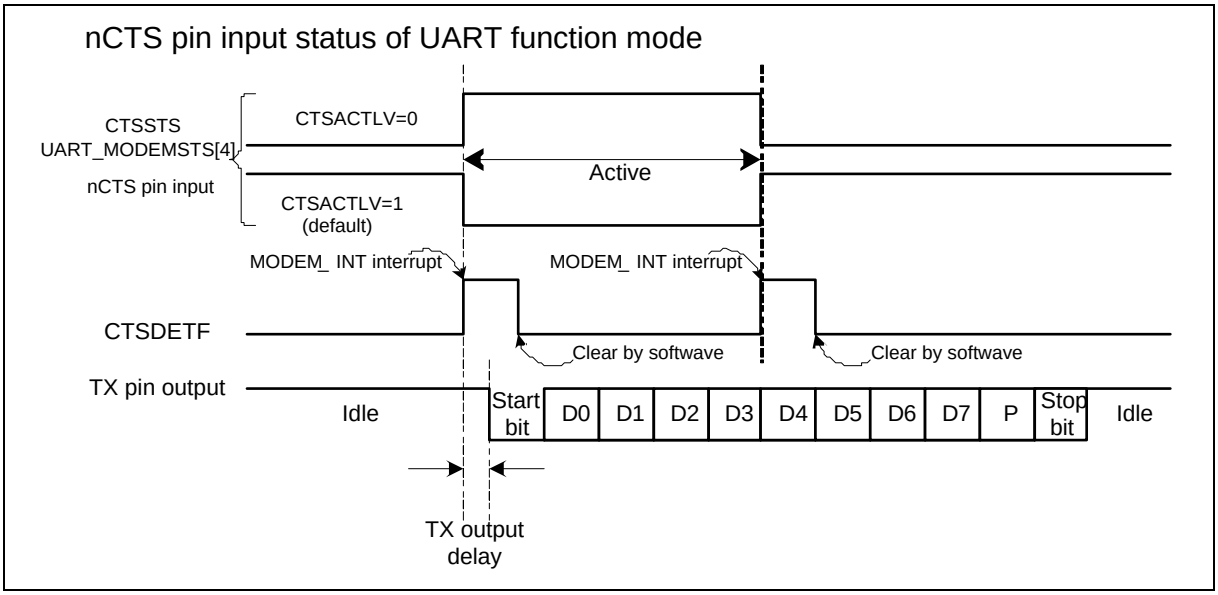


图 6.10-9 UART nCTS 自动流控使能

如下图所示，ART nRTS自动流控模式(ATORTSEN(UART_INTEN[12])=1)中，nRTS的触发阈值由UART FIFO控制寄存器的RTSTRGLV(UART_FIFO[19:16])控制。

设置RTSACTLV(UART_MODEM[9])可以控制nRTS 脚输出反向或非反向。用户可以读RTSSTS (UART_MODEM[13])来获取nRTS脚输出电压的真实逻辑状态。

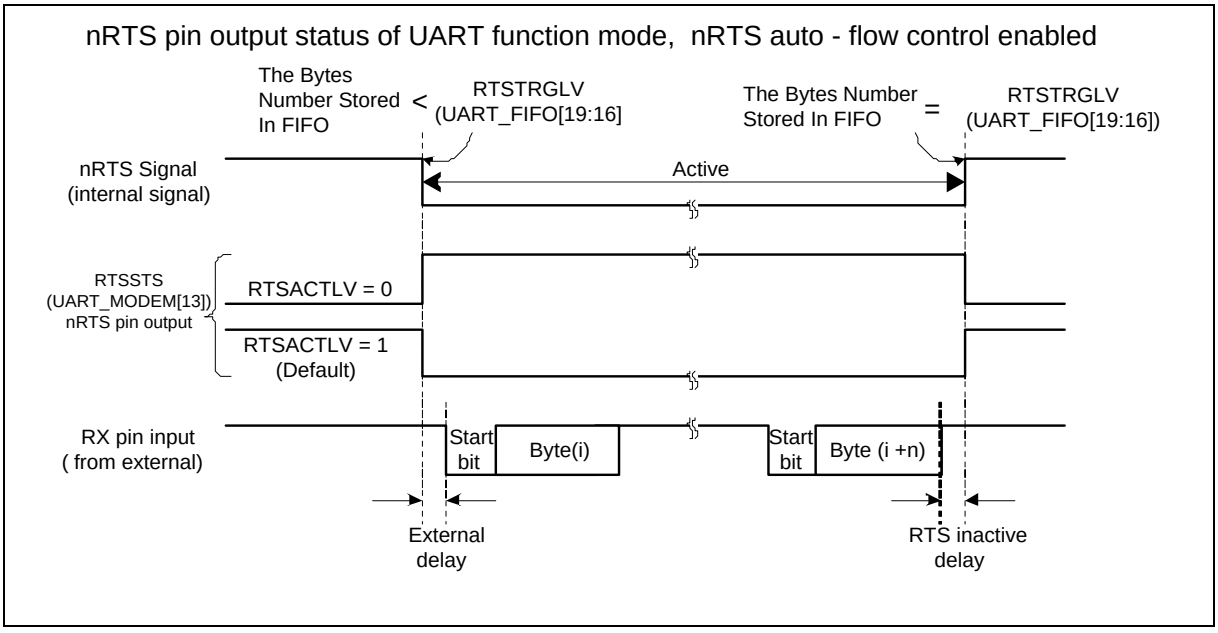


图 6.10-10 UART nRTS自动流控使能

如下图，在软件模式下(ATORTSEN(UART_INTEN[12])=0)，软件改动nRTS(UART_MODEM[1])来实现nRTS流控。

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

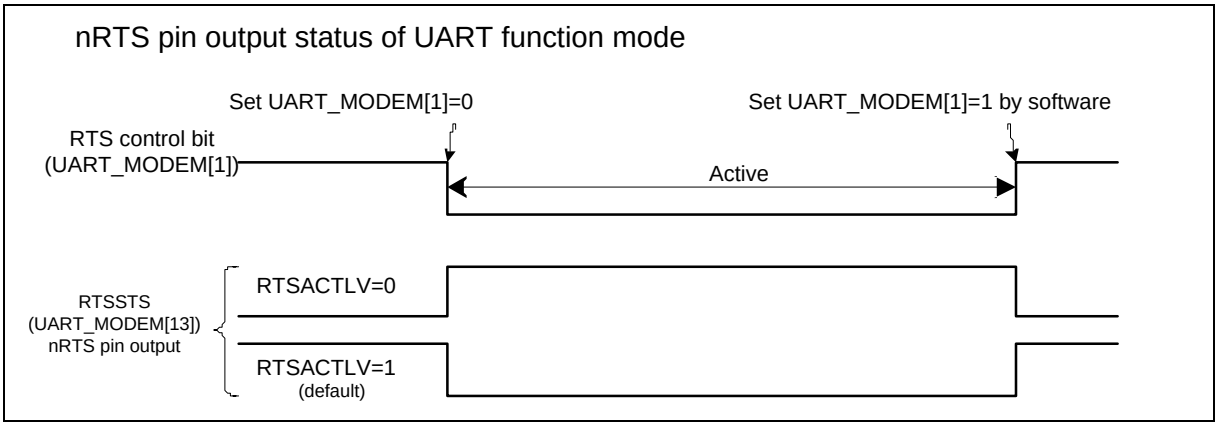


图 6.10-11 UART nRTS软件方式下的自动流控

6.10.5.8 IrDA功能模式

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

IrDA 模式下，BAUDM1 (UART_BAUD [29])需为0。

波特率= Clock / (16 * BRD +2)，这里BRD(UART_BAUD[15:0])是UA_BAUD寄存器中定义的波特率分频器。

以下框图展示了IrDA控制模块框图

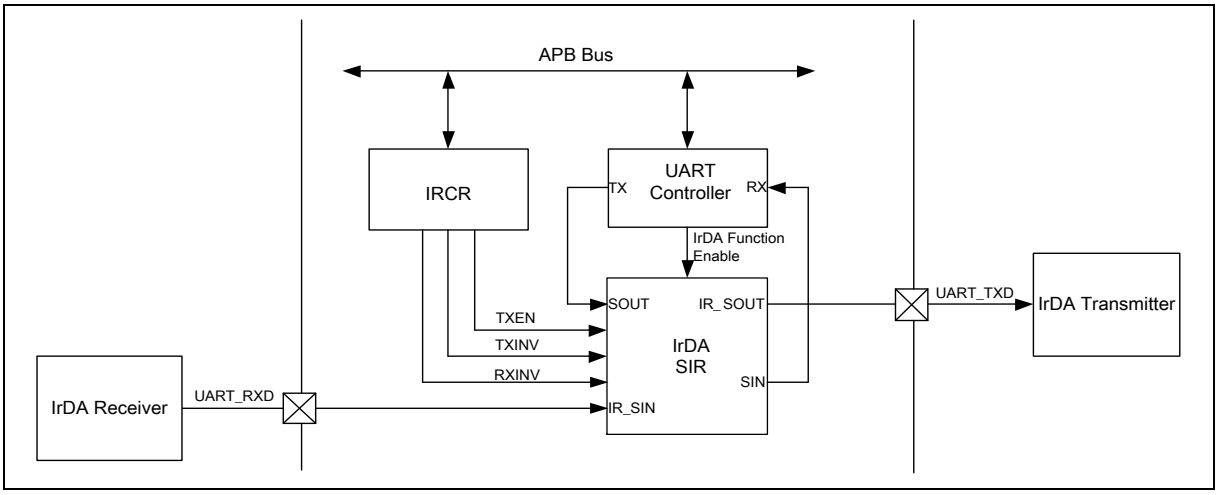


图 6.10-12 IrDA 控制模块框图

IrDA SIR发送编码

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

在正常模式下，传输脉冲的宽度为3/16 波特率周期。

IrDA SIR接收解码

IrDA SIR 接收解码器对输入管脚的(Return-to-Zero, Inverted (RZI))串行位流进行解调，并输出NRZ串行位流到 UART接收数据输入端。

当解码器输入端为低时，表明接收到一个起始位。

IrDA SIR操作

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

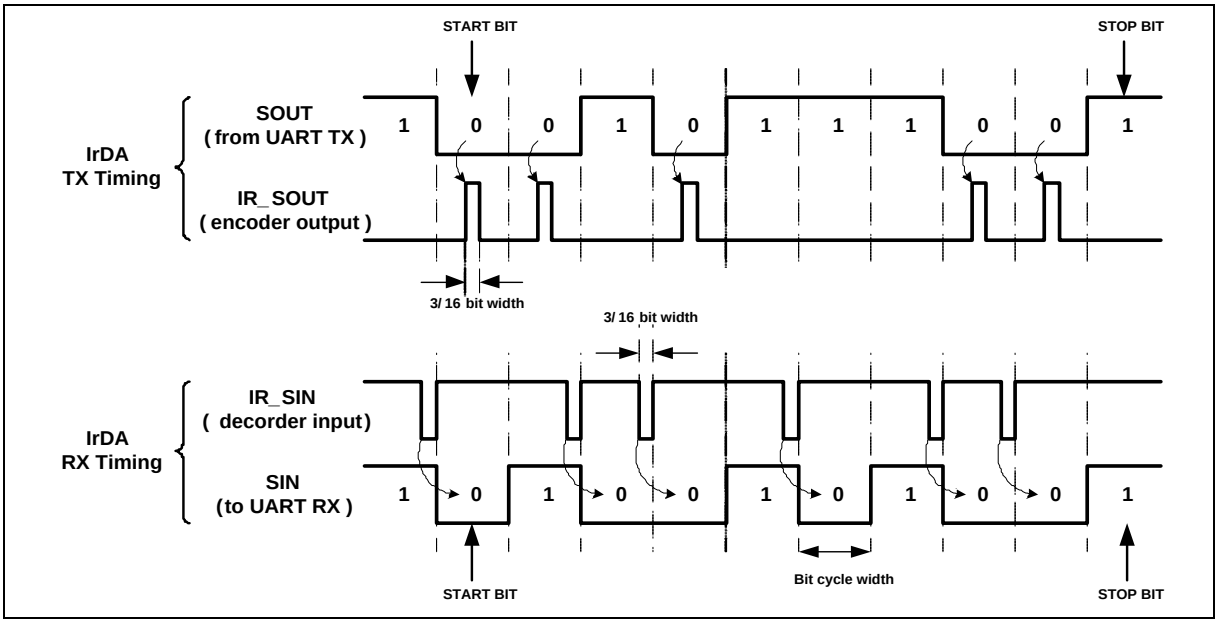


图 6.10-13 IrDA TX/RX时序图

6.10.5.9 LIN (本地互连网络) 功能模式

UART1/UART2支持LIN功能，通过设置FUNCSEL (UART_FUNCSEL[1:0])为 ’01’可以将UART设定为LIN模式。在主机模式，UART1/UART2支持LIN break/delimiter的产生以及break/delimiter的侦测，在LIN从机模式下，支持报头侦测和自动重新同步

LIN帧结构

根据LIN协议，所有的传输信息被打包为帧。一个帧由一个报头（主机任务提供）和一个紧跟其后的应答（从机任务提供）组成。报头（主机任务提供）由一个break域和一个sync（同步）域再跟一个帧识别码(frame ID)组成。帧ID仅作为定义帧的用途。从机任务负责回应相关的帧ID。响应由一个数据域和一个校验域组成。下图是LIN帧的结构。

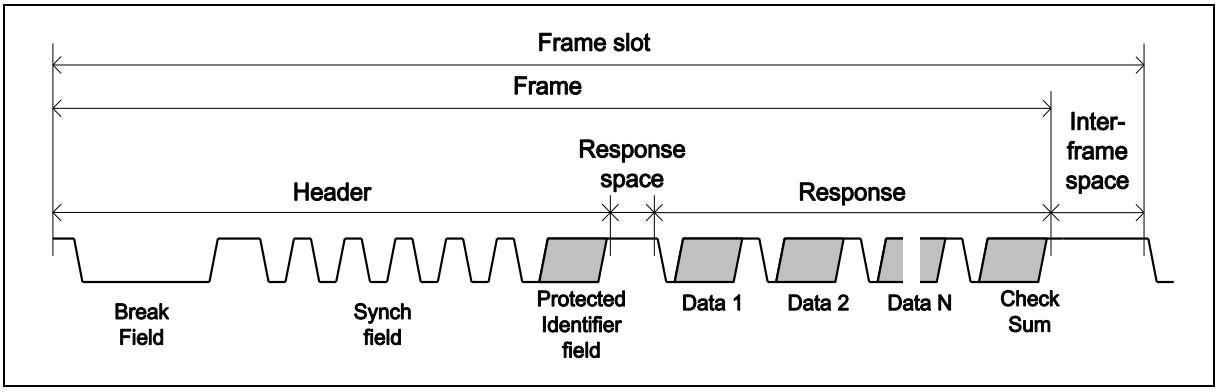


图 6.10-14 LIN 的帧结构

LIN字节结构

在LIN模式，根据LIN的标准，每个字节由值为0（显性）的START位开始，接着是8位数据位，没有校验位，LSB优先，由一个值为1（隐性）的STOP位结束。字节的结构如下：

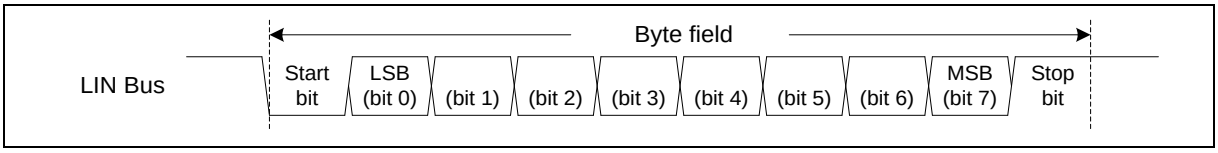


图 6.10-15LIN字节结构

LIN 帧 ID和奇偶格式

LIN模式下LIN的帧ID值如下，帧ID校验位可以通过软件或硬件产生，产生方式取决于IDPEN (UART_LINCTL[9])位的设置。

如果校验位通过硬件产生，用户需要填写ID0~ID5 (UART_LINCTL [29:24])，硬件将会计算P0 (UART_LINCTL[30]) 和 P1 (UART_LINCTL[31])，否则用户必须填写帧ID和校验位。

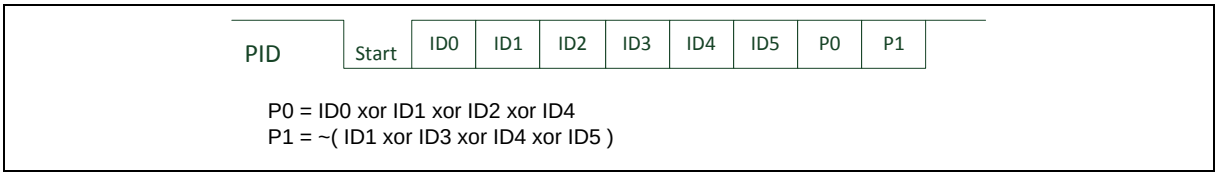


图 6.10-16 LIN 帧 ID 和奇偶格式

LIN主机模式

UART1/UART2 控制器支持 LIN 主机模式，使能并初始化 LIN 主机模式需要如下步骤：:

1. 设置UART_BAUD 寄存器设定波特率.
2. 设置WLS (UART_LINE[1:0])=‘11’配置数据长度为8位，PBE (UART_LINE[3])=0禁止奇偶校验， NSB (UART_LINE[2])=0配置1位stop位。.
3. 设置FUNCSEL (UART_FUNCSEL[1:0]) 位为“01”选择LIN功能模式.

一个完整的报头由一个break空间和同步域再跟一个帧标识符(帧ID)组成。UART1/UART2控制器可以选择三种报头发送模式，通过设置HSEL (UART_LINCTL[23:22])可以选择“break 空间” 或 “break 空间和同步空间” 或 “break空间, 同步空间和 帧 ID”作为发送的报头。如果选择的报头是“break空间”， 发送一个完整的报头到总线，软件必须依次填同步数据(0x55)和帧ID数据到UART_DAT 寄存器。如果选择的报头是“break空间和同步空间”，软件必须填帧ID数据到UART_DAT 寄存器，按顺列来发送一个完整的报头到总线。如果选择的报头是“break空间, 同步空间和 帧 ID”，硬件会自动控制报头发送顺序，但是软件必须填帧ID数据到PID (UART_LINCTL [31:24])。当选择的报头模式是“break 空间, 同步空间和 帧 ID”时，帧ID校验位可以由软件或硬件来计算，这取决于IDPEN (UART_LINCTL[9])位是否置位。

HSEL	Break空间	同步空间	帧ID
0	硬件产生	软件处理	软件处理
1	硬件产生	硬件产生	软件处理

2	硬件产生	硬件产生	硬件产生 （但软件要先填写 ID到 PID (UART_LINCTL[31:24])

表 6.10-6 LIN主模式时报头选择

当UART 工作于LIN数据传输模式时，LIN总线传输状态可以由硬件或软件监控。通过设置BITERREN (UART_LINCTL [12])为 “1”使能硬件监控。如果在LIN发送状态输入管脚(UART_RX) 状态不同于输出管脚(UART_TX) 状态，硬件会产生一个中断到CPU。软件也能通过读回UART_DAT寄存器数据监视LIN总线传输状态。以下为编程顺序示例。

在主机模式下，没有软件错误监控的步骤:

1. 填保密标识符ID到PID (UART_LINCTL[31:24])
2. 通过设置HSEL (UART_LINCTL [23:22])=10，选择硬件传输的报头空间包括“break 空间 + 同步空间 +保密标识符ID域.
3. 通过设置SENDH (UART_LINCTL[8])为1选择硬件传输报头.
4. 等待SENDH (UART_LINCTL[8])被硬件清零.
5. 等待TXEMPTYF (UART_FIFOSTS[28])被硬件置1

注1: break空间的缺省值是12 个显性位 (break空间) 和1 个隐性位 (break/sync 分隔符). 软件可以通过设定BRKFL (UART_LINCTL [19:16])和BSL (UART_LINCTL[21:20])来改变break空间的长度和break/sync分隔符的长度.

注2: break/sync分隔符缺省长度是1比特时间，字节间隔缺省也是1个比特时间。软件可以通过设定BSL (UART_LINCTL[21:20])和DLY(UART_TOUT[15:8])来改变它们之间的间隔。

注3: 如果报头包含“break空间, sync空间和帧ID空间”, 在触发硬件发送报头之前(设定SENDH (UART_LINCTL[8]) ， 软件必须填帧 ID 到PID (UART_LINCTL[31:24]) 。 根据IDPEN (UART_LINCTL[9])的设定，帧 ID 校验可以由硬件或者软件产生。如果校验由软件产生IDPEN (UART_LINCTL[9])为0, 软件必须填8 位数据 (包括 2 个校验位)到帧ID域；如果校验位由硬件产生(IDPEN (UART_LINCTL[9])= 1), 软件填 ID0~ID5就好，硬件负责计算P0和 P1。.

在主机模式下，有软件错误监控的步骤:

1. 设置UART_LINCTL [23:22]= 00，选择硬件传输的报头空间，其内容只包括 “break空间”
2. 设置BRKDETEN (UART_LINCTL[10])使能 break侦测功能
3. 设置SENDH (UART_LINCTL[8])请求硬件传输 break + break/同步 分隔符
4. 等待，直到BRKDETF (UART_LINSTS[8])标志被硬件置为 “1”.
5. 写0x55到UART_DAT寄存器发送 sync空间
6. 等待，直到RDAIF (UART_INTSTS[0])标志被硬件置为 “1” ，然后读回UART_DAT寄存器的值
7. 写密码到UART_DAT寄存器，发送头帧ID
8. 等待，直到RDAIF (UART_INTSTS[0])标志被硬件置为 “1”， 然后读回UART_DAT寄存器的值·.

LIN break和分隔符侦测

当软件通过设定BRKDETEN (UART_LINCTL[10])使能break 侦测功能时， break侦测功能被激活。break 侦测电路和UART1/UART2接收电路是完全独立的。.

当break侦测功能使能时,电路侦测UART_RX引脚的起始信号。如果UART LIN控制器侦测到显性连续大于11个比特并跟随1个隐性比特 (分隔符)，在break结束时BRKDETF (UART_LINSTS[8])标志将被置位，如果LINIEN (UART_INTEN[8])=1, LININT (UART_INTSTS[15])中断将发生，break 侦测和break 标志如下图所示。.

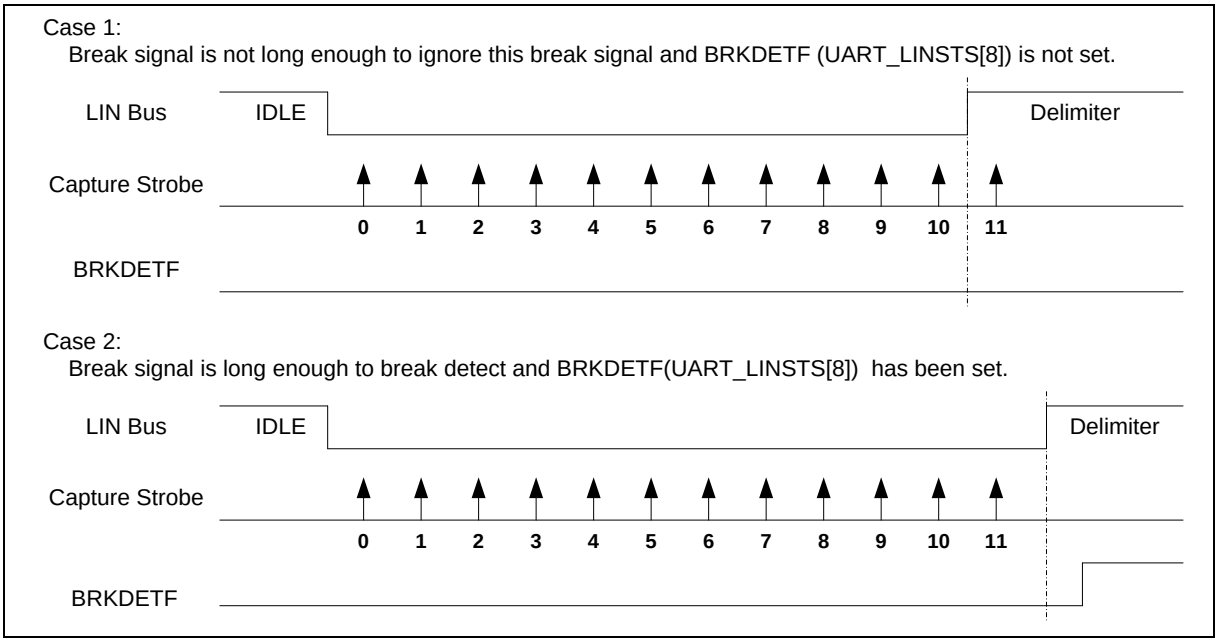


图 6.10-17 LIN模式Break侦测

LIN 从模式

UART1/UART2控制器支持LIN从模式。使能和初始化LIN从模式的必要步骤如下:

1. 设置UART_BAUD 寄存器设定波特率
2. 设定WLS (UART_LINE[1:0])= 11设定数据长度为8 比特，清PBE (UART_LINE[3])禁止校验，清NSB (UART_LINE[2])设定停止位为1比特
3. 设定FUNCSEL (UART_FUNCSEL[1:0])为“01”，选择 LIN 功能
4. 设定SLVEN (UART_LINCTL[0])为1，使能LIN 从机模式

LIN 报头接收

根据LIN 协议, 从节点必须等待从主节点接收一个有效的报头，然后应用程序可以采取以下动作 (根据主报头帧ID 的值)

- 接收响应

- 发送响应.
- 忽略响应，等待下一个报头

LIN从模式下，通过设定SLVHDEN (UART_LINCTL[10])，使能从机报头检测功能来侦测完整的报头(接收 “break 空间”, “sync空间” 和 “frame ID空间”)。接收到一个 LIN 报头后， SLVHDETF (UART_LINSTS[0])标志将被置位。如果LINIEN (UART_INTEN[8]) =1, 中断将发生。 通过设定IDPEN (UART_LINCTL[9])，使能帧ID校验检查功能。如果收到的帧ID校验位不正确时 (break 和 sync 空间正确), SLVIDPEF (UART_LINSTS[2])标志将被置1。如果LINIEN(UART_INTEN[8]) =1, 中断将发生，SLVHDETF (UART_LINSTS[0])会被置1。通过设定MUTE (UART_LINCTL[4])=1，用户也可以将 LIN 设为mute 模式。该模式只允许检测报头(break + sync + frame ID) ，不允许接收任何其它字符。为了避免比特率偏差, 控制器支持自动重同步功能，避免时钟误差错误，通过设定SLVAREN (UART_LINCTL[2])使能该特性。

LIN发送响应

LIN 从机模式可以发送响应和接收响应。当从机节点是响应的发送者时，通过将数据填到UART_DAT 寄存器发送响应；如果从机节点是响应的接收者时，从接节点从LIN总线接收数据

LIN 报头超时错误

LIN 从机控制器包含一个报头超时计数器。如果整个报头没有在57比特的最大时间限制内收到，报头错误标志SLVHEF (UART_LINSTS [1])将被置位。超时计数器在每个break侦测沿使能，在下列条件下将停止：

- LIN 帧 ID 域已经收到了
- 报头错误标志被置起
- 写 1 到SLVSYNCF (UART_LINSTS[3])，重新侦测新的报头.

LIN Mute模式和从Mute模式退出的条件

Mute模式下, LIN 从机节点需满足一定的条件时才能接收其它数据。此模式只允许侦测报头，禁止接收任何其它数据。通过设定 MUTE (UART_LINCTL[4] 使能 Mute 模式， 设置 HSEL (UART_LINCTL[23:22])来退出Mute模式。.

推荐发送校验和后设置LIN从节点进入Mute模式.

LIN 从机控制器从Mute模式退出描述如下：如果HSEL (UART_LINCTL[23:22])设为 “break 空间”，当 LIN 从机控制器检测到一个有效的LIN break + 分隔符时, 控制器将使能接收器 (退出Mute 模式) ，随后的数据(sync 数据, 帧ID , 响应数据) 将被收到 RX-FIFO中。.

如果HSEL (UART_LINCTL[23:22])设为 “break空间 和 sync空间”, 当 LIN 从机控制器检测到一个有效的LIN break + 分隔符，后面跟一个有效的同步域并且没有帧错误时，控制器将使能接收器 (退出Mute 模式)， 随后的数据(帧ID, 响应数据) 将被收到 RX-FIFO 中。如果 HSEL (UART_LINCTL[23:22])设为“break 空间, sync空间和 ID空间”, 当 LIN 从机控制器检测到一个有效的LIN break + 分隔符和有效的同步域，并且没有帧错误后面跟一个有效的帧ID域并且没有帧错误，并且收到的帧ID和PID (UART_LINCTL[31:24])中的值匹配时，控制器将使能接收器 (退出Mute

模式) 随后的数据(响应数据) 将被收到 RX-FIFO中。

非自动重新同步从机模式(NAR)

用户可以关闭自动重新同步功能来固定通信波特率。当工作在非自动重新同步模式时，软件需要一些初始化过程，初始化过程如下所示：

1. 设定UART_BAUD 寄存器设定波特率
2. 设定FUNCSEL (UART_FUNCSEL[1:0])为“01”选择 LIN 功能
3. 设定SLVAREN (UART_LINCTL[2])= 0关闭自动重新同步功能
4. 设定SLVEN (UART_LINCTL[0])为1使能 LIN 从机模式..

自动重新同步从机模式(AR)

自动重新同步模式下，每次收到同步域时，控制器将调整比特率发生器。软件需要一些初始化过程，初始化过程如下所示

1. 设定UART_BAUD 寄存器设定波特率
2. 设定UART_FUNCSEL (UART_FUNCSEL[1:0])为‘01选择 LIN 功能模式
3. 设定SLVAREN (UART_LINCTL[2])为 ‘1’使能自动重新同步功能
4. 设定SLVEN (UART_LINCTL[0])为‘1’，使能LIN从机模式

当自动重新同步功能使能后，每个LIN break空间后面，用LIN的工作时钟持续采样5个下降沿的时间，测量的结果储存在内部一个13-bit 寄存器中，在第5个下降沿结束，UART_BAUD 寄存器的值将被自动更新。如果在5个下降沿之前，测量计数器(13-bit) 溢出，报头错误标志SLVHEF (UART_LINSTS [1])将被置位’。 .

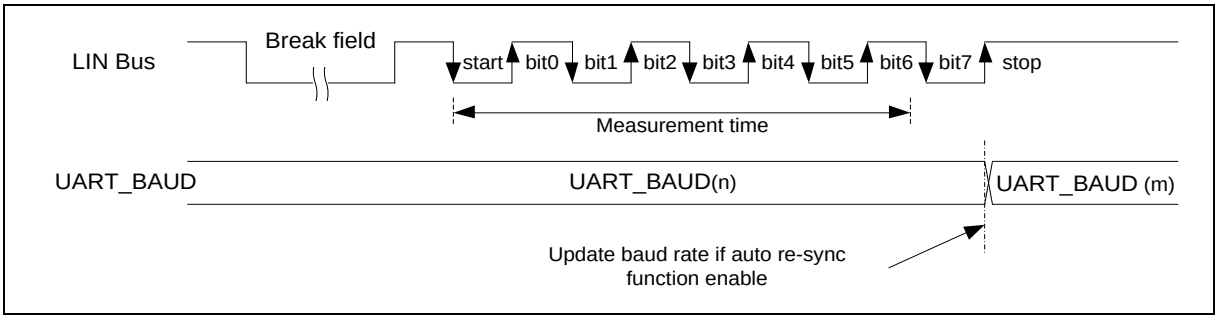


图 6.10-18 LIN同步空间测量

自动重新同步模式(AR)下，软件必须通过设定UART_BAUD 寄存器设定希望的波特率，硬件将保存到内部的TEMP_REG 寄存器, 每次LIN break空间之后, 5个下降沿被采样，测量结果保存到内部13-bit 寄存器 (BAUD_LIN) ，这个结果将自动更新到UART_BAUD 寄存器.

为了保证传输波特率,每个新的break收到之前，波特率发生器必须重新加载初始值。初始值在初始化的时候由应用程序设定（TEMP_REG）。用户可以设定SLVDUEN (UART_LINCTL [3])来使能自动加载初始波特率功能。如果SLVDUEN (UART_LINCTL [3])= 1, 当前帧结束，收到下一个字符之前，硬件将自动加载初始值到UART_BAUD寄存器，UA_BAUD 被更新之后，SLVDUEN (UART_LINCTL [3])将被自动清0。LIN波特率的更新方法如下图所示.

注1: 收到检验字符之前，推荐设定SLVDUEN比特为1.

注2: 侦测到报头错误时，用户需要对SLVSYNCF (UART_LINSTS[3])写1来重新检测报头. 当写1到该位时，硬件将重载初始波特率(TEMP_REG) 并重新搜索新的报头

注3: 自动重新同步模式下，波特率必须设定为mode2 (UART_BAUD [29]) 和 DIV_X_ONE BAUDM0 (UART_BAUD[28]) 必须都为1)

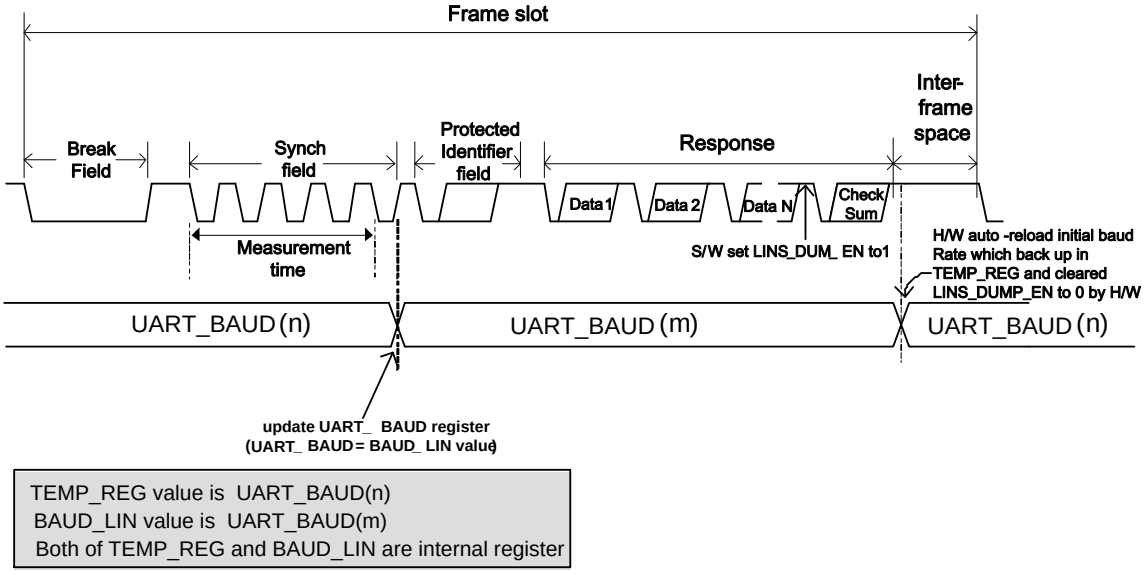


图 6.10-19 UART_BAUD AR模式下SLVDUEN=1时UART_BAUD更新顺序

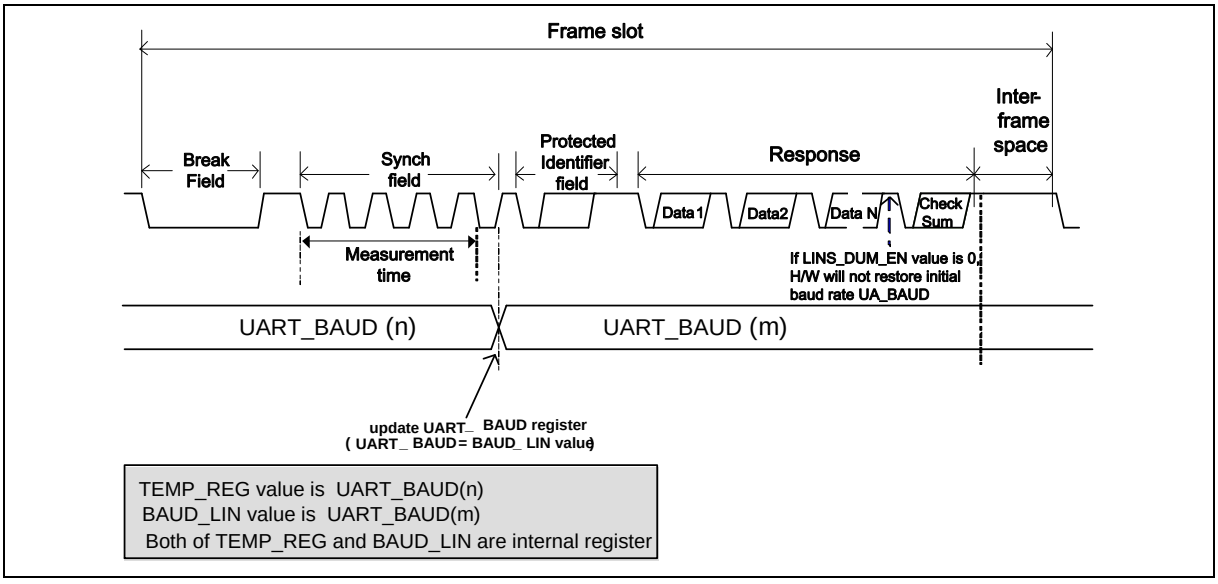


图 6.10-20 UART_BAUD AR模式下SLVDUEN=0时UART_BAUD更新顺序

同步空间误差错误

自动重新同步模式下，控制器将检测同步域的误差错误。误差错误检测比较当前波特率和接收到的

同步域的波特率。两个检测被同步执行.

检查1:根据同步域的第一个下降沿和最后一个下降沿的测量值

- 如果误差大于14.84%, 报头错误标志SLVHEF (UART_LINSTS[1])将被置位
- 如果误差小于14.06%, 报头错误标志SLVHEF (UART_LINSTS[1])不会被置位.
- 如果误差在14.84% 和 14.06%之间, 报头错误标志SLVHEF (UART_LINSTS[1])可能被置位也可能没有被置位.

检查2: 根据同步域的每一个下降沿的测量值.

- 如果误差大于18.75%,报头错误标志SLVHEF (UART_LINSTS[1])将被置位
- 如果误差小于15.62%,报头错误标志SLVHEF (UART_LINSTS[1])不会被置位
- 如果误差在18.75% 和 15.62%之间, 报头错误标志SLVHEF (UART_LINSTS[1]))可能被置位也可能没有被置位..

误差检测基于当前波特率时钟。因而，为了保证误差检测的正确性，通过设定SLVDUEN (UART_LINCTL[3])，新的break收到之前，波特率必须重新加载为初始值(每个checksum收到之前，推荐设定SLVDUEN (UART_LINCTL[3])为1)。

LIN报头错误侦测

LIN 从机模式下, 当用户通过设定SLVHDEN (UART_LINCTL[1])使能报头检测功能时, 硬件将处理报头检测流程。如果报头有错误，LIN 报头错误标志SLVHEF (UART_LINSTS[1])将被置位，如果LINIEN (UART_INTEN[8])=1，中断将发生。报头错误被检测到时，用户必须写1到SLVSYNCF (UART_LINSTS[3])来复位检测电路以重新检测新的报头

如果下列一个条件发生，LIN 报头错误标志SLVHEF (UART_LINSTS[1])将被置位

- Break分隔符太短 (小于 0.5 比特时间).
- 同步域或者帧ID域帧错误
- 同步域数据不是0x55 (非自动重新同步模式)
- 同步域误差错误(自动重新同步模式).
- 同步域测量超时 (自动重新同步模式)
- LIN报头接收超时.

6.10.5.10 RS-485 功能模式

UART控制器另一个可选择的功能是RS-485功能（用户必须设置UART_FUNCSEL [1:0] = ’11’来使能RS-485功能），方向控制则由异步串口的nRTS脚来控制。RS-485收发器的驱动控制是通过nRTS控制信号来驱动的。RS-485模式下的RX和TX大多数特性与UART相同。

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

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

.

RS-485普通多点操作模式 (NMM)

RS-485 普通多点操作模式(RS485NMM (UART_ALTCTL[8]) = 1)，首先，软件决定在检测到地址字节之前的数据是否存储到 RX-FIFO。如果软件想忽略在检测到地址之前的任何数据，流程是设置RXOFF (UART_FIFO [8])，然后使能RS485NMM (UART_ALTCTL [8])，接收器将会忽略数据，直到检测到地址字节 (bit9 =1) 并且地址字节数据存储到RX-FIFO。如果软件想接收在检测到地址字节之前的任何数据，流程是禁用RXOFF (UART_FIFO [8])，然后使能RS485NMM (UART_ALTCTL [8])，接收器将接收任何数据。

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

RS-485自动地址识别工作模式(AAD)

RS-485 自动地址识别模式(RS485AAD (UART_ALTCTL[9]) = 1)，接收器在检测到地址字节 (bit9=1) 并且地址字节数据与ADDRMV (UART_ALTCTL[31:24])的值相匹配之前将忽略所有数据。地址字节数据将存储在 RX-FIFO。所有接收字节数据将被接收并存储于 RX-FIFO 直到地址字节或数据字节与ADDRMV (UART_ALTCTL[31:24])的值不匹配。

RS-485 自动方向控制功能(AUD)

RS-485控制器的另一个功能是 RS-485 自动方向控制功能(RS485AUD (UART_ALTCTL[10) = 1)。RS-485 通过nRTS 驱动控制异步串口的控制信号，使能RS-485 驱动器。nRTS 连接到RS-485 驱动器，设置nRTS线为高（逻辑1）使能RS-485 驱动器。设置nRTS 为低（逻辑0），使驱动器进入 tri-state 状态。用户通过设置寄存器UART_MODEM 中的RTSACTLV改变nRTS 驱动电平

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

设置 RTSACTLV(UART_MODEM[9]) 可以控制 nRTS 脚的输出电平。用户可以通过读 RTSSTS(UART_MODEM[13])来获取nRTS脚上实际的输出逻辑电平

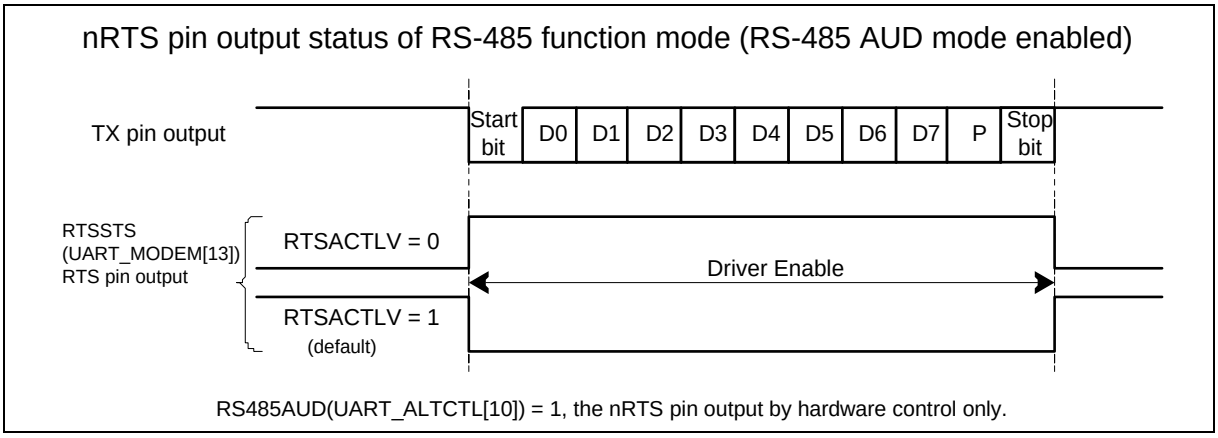


图 6.10-21 RS-485自动方向控制模式下nRTS管脚驱动电平

下图展示了通过软件控制(RS485AUD (UART_ALTCTL[10])=0)模式下，RS-485 nRTS脚的驱动电平。nRTS驱动电平通过设置RTS(UART_MODEM[1])来控制。

设置RTSACTLV (UART_MODEM[9])可以控制nRTS脚的输出与RTS(UART_MODEM[1])设置值是否相反。用户可以读RTSSTS (UART_MODEM[13])来获取nRTS脚上实际的逻辑电平.

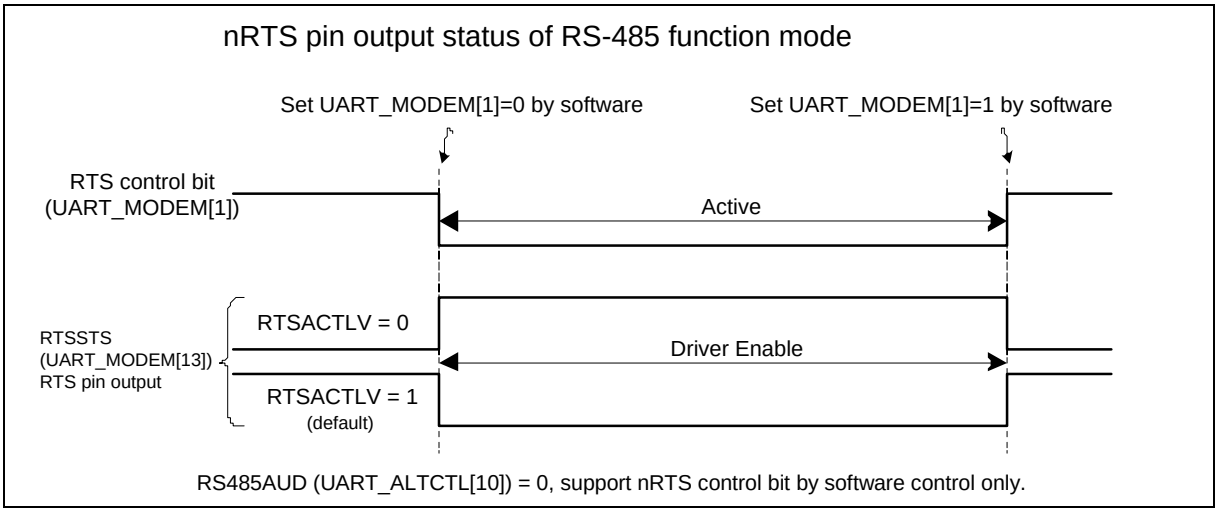


图 6.10-22 RS-485软件控制模式下nRTS管脚驱动电平

程序编写例程:

1. 设置UART_FUNCSEL 中的 FUN_SEL 选择 RS-485 功能
2. 设置RXOFF (UART_FIFO[8])，使能或禁用 RS-485 接收器
3. 设置RS485NMM (UART_ALTCTL[8])或RS485AAD (UART_ALTCTL[9])模式
4. 如果选择RS485AAD (UART_ALTCTL[9])模式 ，ADDRMV (UART_ALTCTL[31:24])需设置成自动地址匹配值
5. 设置RS485_AUD (UA_ALT_CSR[10])来决定是否为自动方向控制.

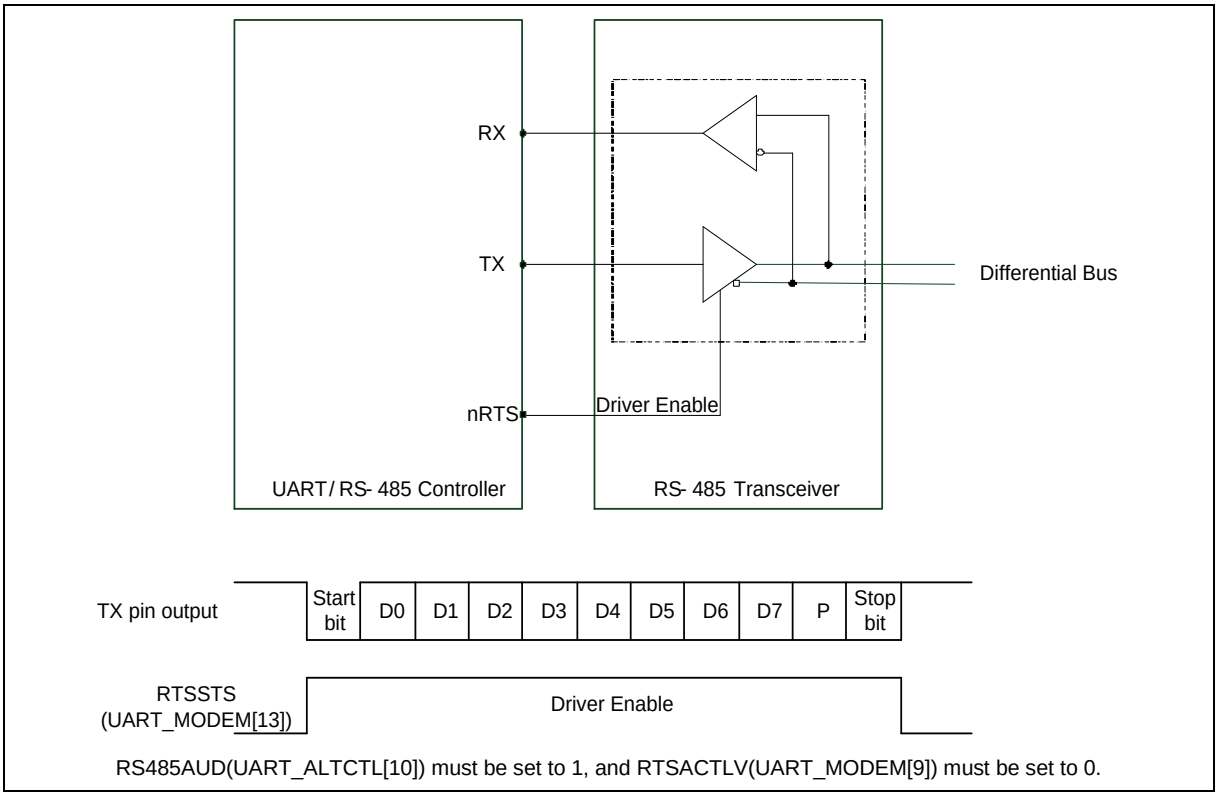


图 6.10-23 RS-485帧结构

6.10.6 寄存器 映射

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移地址	R/W	描述	复位值
UART 基地址: **UARTx_BA = 0x400E_C000+(x * 0x1000)** **x=0,1,2**				
UART_DAT	UARTx_BA+0x00	R/W	UART接收/发送数据 寄存器	不定
UART_INTEN	UARTx_BA+0x04	R/W	UART 中断 使能 寄存器	0x0000_0000
UART_FIFO	UARTx_BA+0x08	R/W	UART FIFO 控制寄存器	0x0000_0101
UART_LINE	UARTx_BA+0x0C	R/W	UART线近寄存器	0x0000_0000
UART_MODEM	UARTx_BA+0x10	R/W	UART Modem 控制 寄存器	0x0000_0200
UART_MODEMSTS	UARTx_BA+0x14	R/W	UART Modem 状态寄存器	0x0000_0110
UART_FIFOSTS	UARTx_BA+0x18	R/W	UART FIFO 状态寄存器	0x1040_4000
UART_INTSTS	UARTx_BA+0x1C	R/W	UART中断状态寄存器	0x0000_0002
UART_TOUT	UARTx_BA+0x20	R/W	UART 溢出寄存器	0x0000_0000

UART_BAUD	UARTx_BA+0x24	R/W	UART 波特率分频寄存器	0x0F00_0000
UART_IRDA	UARTx_BA+0x28	R/W	UART IrDA控制寄存器	0x0000_0040
UART_ALTCTL	UARTx_BA+0x2C	R/W	UART复用控制/状态寄存器	0x0000_000C
UART_FUNCSEL	UARTx_BA+0x30	R/W	UART功能选择寄存器	0x0000_0000
UART_LINCTL	UARTx_BA+0x34	R/W	UART LIN 控制寄存器 *	0x000C_0000
UART_LINSTS	UARTx_BA+0x38	R/W	UART LIN 状态 寄存器 *	0x0000_0000

注[*]:UART0不支持

6.10.7 寄存器 描述

UART 接收/发送数据 寄存器 (UART_DAT)

寄存器	偏移地址	R/W	描述	复位值
UART_DAT	UARTx_BA+0x00	R/W	UART接收/发送数据寄存器	不定

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**
DAT							

位	描述	
[31:8]	**Reserved**	保留.
[7:0]	**DAT**	**接收/发送缓冲区** 写操作： 写数据到该寄存器, 数据将会保存到发送FIFO. UART控制器将会通过TX脚把存放在FIFO中最前面的数据发送出去 读操作: 读该寄存器, UART 将返回从接收FIFO中接收到的8位数据(LSB 优先)..

UART 中断使能 寄存器 (UART_INTEN)

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

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					ABRIEN	Reserved	
15	14	13	12	11	10	9	8
Reserved		ATOCTSEN	ATORTSEN	TOCNTEN	WKDATIEN	WKCTSIEN	LINIEN
7	6	5	4	3	2	1	0
Reserved		BUFERRIEN	RXTOIEN	MODEMIEN	RLSIEN	THREIEN	RDAIEN

位	描述	
[31:19]	Reserved	保留.
[18]	ABRIEN	**自动波特率中断使能位** 0 = 自动波特率中断禁止 1 = 自动波特率中断使能
[17:14]	Reserved	保留.
[13]	ATOCTSEN	**nCTS自动流控使能位** 0 = nCTS自动流控禁止. 1 = nCTS自动流控使能. 当nCTS 自动流控使能后，当nCTS输入有效， UART 会发送数据到外部设备（UART将不会发送数据到外部设备直到nCTS有效）
[12]	ATORTSEN	**nRTS 自动流控使能位** 0 = nRTS 自动流控禁止 1 = nRTS 自动流控使能 当nRTS 自动流控使能后，如果RX FIFO中字节的数量等于RTSTRGLV (UART_FIFO[19:16])，UART会自动禁止nRTS信号
[11]	TOCNTEN	超时计数器使能位 0 = 超时计数器禁止 1 = 超时计数器使能.
[10]	WKDATIEN	**输入数据唤醒中断使能控制** 0 = 输入数据唤醒系统功能禁止。 1 = 输入数据唤醒系统功能使能，当系统在掉电状态时，输入数据可以将系统从掉电状态唤醒。 **注:**当输入数据唤醒动作完成并且“系统时钟”工作稳定后，硬件将清除该位。

[9]	**WKCTSIEN**	**nCTS 唤醒中断使能控制** 0 = nCTS唤醒系统功能禁止. 1 = 唤醒系统功能使能，当系统在掉电状态时，外部nCTS信号电平变化可以将系统从掉电状态唤醒。
[8]	**LINIEN**	**LIN 总线中断使能位(UART1/UART2有效)** 0 = LIN 总线中断禁止 1 = LIN 总线中断使能 **注:** 该位用于LIN 功能模式
[7:6]	**Reserved**	保留.
[5]	**BUFERRIEN**	**缓存错误中断使能** 0 = 缓冲区错误中断 禁止 1 = 缓冲区错误中断 使能
[4]	**RXTOIEN**	**RX超时中断使能位** 0 = RX 超时中断禁止 1 = RX 超时中断使能.
[3]	**MODEMIEN**	**Modem 状态中断使能位** 0 = 关闭MODEM状态中断. 1 = 使能MODEM状态中断
[2]	**RLSIEN**	**Receive 线状态中断使能位** 0 = 关闭接收线状态中断. 1 = 使能接收线状态中断
[1]	**THREIEN**	**发送保持寄存器空中断使能位** 0 = 关闭发送数据寄存器为空时中断 1 = 使能发送数据寄存器为空时中断
[0]	**RDAIEN**	**接收数据有效中断使能位** 0 = 接收数据有效中断关闭 1 = 接收数据有效中断使能.

UART FIFO控制寄存器 (UART_FIFO)

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

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

位	描述	
[31:20]	**Reserved**	保留.
[19:16]	**RTSTRGLV**	**NRTS自动流控触发阈值** 0000 = nRTS触发阈值为1 字节. 0001 = nRTS 触发阈值为 4字节. 0010 = nRTS 触发阈值为 8 字节. 0011 = nRTS 触发阈值为 14 字节. 0100 = nRTS 触发阈值为 30/14 (64 FIFO/16 FIFO). 0101 = nRTS 触发阈值为 46/14 (64 FIFO/16 FIFO). 0110 = nRTS 触发阈值为 62/14 (64 FIFO/16 FIFO). 其它 = 保留. **注:** 该区域用于自动nRTS 流控制
[15:9]	**Reserved**	保留.
[8]	**RXOFF**	**接收器禁止设置位** 是否禁用接收器（置 1 禁用接收器） 0 =接收器使能 1 =接收器禁止 **注：** 该位用于 RS-485 普通多点模式,需要在RS485NMM (UART_ALTCTL [8])之前设置
[7:4]	**RFITL**	**RX FIFO 中断 (INT_RDA) 触发级别** 当FIFO 接收字节数等于 RFITL 后，RDAIF 将被置位（如果RDAIEN (UART_INTEN [0])使能，将产生中断）. 0000 = RX FIFO 中断 触发阈值为 1 字节. 0001 = RX FIFO 中断 触发阈值为 4 字节. 0010 = RX FIFO 中断 触发阈值为 8 字节.

		0011 = RX FIFO 中断 触发阈值为 14 字节. 0100 = RX FIFO 中断 触发阈值为 30/14 (64 FIFO/16 FIFO). 0101 = RX FIFO 中断 触发阈值为 46/14 (64 FIFO/16 FIFO). 0110 = RX FIFO 中断 触发阈值为 62/14 (64 FIFO/16 FIFO). 其它 = 保留.
[3]	**Reserved**	保留.
[2]	**TXRST**	**TX 域软件复位** 当TXRST (UART_FIFO[2])置位， 发送 FIFO和 TX 内部状态机中的所有数据将被清除。 0 =不影响. 1 =该位写 1 将复位 TX 内部状态机和指针 **注：** 该位至少 3个 UART 时钟周期后会自动清0
[1]	**RXRST**	**RX 域软件复位** 当RXRST (UART_FIFO[1])被置位， 接收 FIFO和 RX 内部状态机中的所有数据将被清除。 0 = 不影响. 1 =该位写 1 将复位 RX 内部状态机和指针. **注：** 该位至少 3个 UART 时钟周期后会自动清0.
[0]	**Reserved**	保留.

UART 线控 寄存器 (UART_LINE)

寄存器	偏移地址	R/W	描述	复位值
UART_LINE	UARTx_BA+0x0C	R/W	UART线控寄存器	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	BCB	SPE	EPE	PBE	NSB	WLS	

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

[1:0]	**WLS**	**字长度选择** 设置 UART字长度. 00 = 5 位. 01 = 6 位. 10 = 7 位. 11 = 8 位.

UART Modem控制寄存器 (UART_MODEM)

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

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		RTSSTS	Reserved			RTSACTLV	Reserved
7	6	5	4	3	2	1	0
Reserved						RTS	Reserved

位	描述	
[31:14]	**Reserved**	保留.
[13]	**RTSSTS**	**nRTS 脚状态(只读)** 该位的值对应于nRTS脚的输出电平 0 = nRTS脚为低电平逻辑状态 1 = nRTS脚为高电平逻辑状态
[12:10]	**Reserved**	保留.
[9]	**RTSACTLV**	**nRTS 脚的有效电平** 该位定义了nRTS输出脚的有效电平 0 = nRTS 脚输出为高电平有效. 1 = nRTS 脚输出为低电平有效（默认）. **注1:**参考 图 6.10-10 和 图 6.10-11 UART功能模式 **注2:** 参考 图 6.10-21 和 图 6.10-22 RS-485 功能模式.
[8:2]	**Reserved**	保留.
[1]	**RTS**	**NRTS (请求发送) 信号控制** 该位直接控制内部nRTS脚信号是否有效，然后使用RTSACTLV位的配置驱动nRTS脚输出 0 = nRTS 信号有效. 1 = nRTS 信号无效. **注1:** UART 功能模式下，当nRTS自动流控被使能后，nRTS信号控制位无效 **注2:** RS-485模式下，当RS-485自动方向模式（AUD）被使能后，nRTS信号控制位无效
[0]	**Reserved**	保留.

UART Modem 状态寄存器 (UART_MODEMSTS)

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

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							CTSACTLV
7	6	5	4	3	2	1	0
Reserved			CTSSTS	Reserved			CTSDETF

位	描述	
[31:9]	Reserved	保留.
[8]	CTSACTLV	**nCTS 脚有效电平** 该位定义了nCTS输入脚的有效电平 0 = nCTS输入脚高电平有效 1 = nCTS输入脚低电平有效.(默认)
[7:5]	Reserved	保留.
[4]	CTSSTS	**NCTS脚状态(只读)** 该位对应于nCTS脚输入逻辑状态 0 = nCTS脚输入状态为低电平 1 = nCTS脚输入状态为高电平 **注:**当UART控制器外围时钟被使能，且nCTS功能被使能，该位才有效。
[3:1]	Reserved	保留.
[0]	CTSDETF	**检测到nCTS 引脚电平改变标志 (只读)** 如果nCTS输入脚上有电平变化，该位将被置位，如果MODEMIEN (UART_INTEN [3])位被置位，将会产生Modem中断。 0 = nCTS 输入脚没有电平变化. 1 = nCTS输入脚有电平变化. **注:** 该位只读, 写 “1” 清零.

UART FIFO 状态 寄存器 (UART_FIFOSTS)

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

31	30	29	28	27	26	25	24
Reserved			TXEMPTYF	Reserved			TXOVIF
23	22	21	20	19	18	17	16
TXFULL	TXEMPTY	TXPTR					
15	14	13	12	11	10	9	8
RXFULL	RXEMPTY	RXPTR					
7	6	5	4	3	2	1	0
Reserved	BIF	FEF	PEF	ADDRDETF	ABRDTOIF	ABRDIF	RXOVIF

位	描述	
[31:29]	**Reserved**	保留.
[28]	**TXEMPTYF**	**发送空标志(只读)** 当TX FIFO (UART_DAT)为空, 并且最后一个字节的STOP位也已经被发送完毕，该位被硬件置1. 0 = TX FIFO 不为空或最后一个字节的STOP位还没有发送. 1 = TX FIFO 为空而且最后一个字节的STOP位已经发送. 当TX FIFO不为空或最后一个字节的STOP位还没有被发送完毕，那么该位将被自动清零
[27:25]	**Reserved**	保留.
[24]	**TXOVIF**	**TX溢出错误中断标志 (只读)** 如果TX FIFO (UART_DAT满，如果再向UART_DAT写入数据，将会导致此位被置1. 0 = TX FIFO 未溢出 1 = TX FIFO 已溢出. **注**: 该位为只读位. 写1可清除该位
[23]	**TXFULL**	**T发送FIFO满(只读)** 此位用于指示TX FIFO是否已满. 0 = TX FIFO 未满 1 = TX FIFO 已满 **注:**当TX FIFO Buffer中的字节数量等于16/64,此位将被置1.否则被硬件清零.
[22]	**TXEMPTY**	**发送FIFO 空(只读)** 此位指示TX FIFO是否为空。 0 = TX FIFO不为空. 1 = TX FIFO为空. **注**：当TX FIFO中的最后一个字节被发送到发送移位寄存器，硬件将把此位置1.当写入数

		据到DAT中, (TX FIFO不为空)，此位被清零
[21:16]	**TXPTR**	**TX FIFO 指针(只读)** 此位指示TX FIFO缓冲区指针位置。当CPU写一个字节到UART_DAT寄存器，TXPTR将累加1.当TX FIFO发送一个字节到发送移位寄存器中，TXPTR指针将减1. TXPTR显示的最大值是15/63。当TX FIFO缓冲区所填充数据数量达到16/64，TXFULL将被置1，TXPTR被清零。如果TX FIFO中发送一个字节到发送移位寄存器，TXFULL位将被清零，TXPTR显示15/63。
[15]	**RXFULL**	**接收 FIFO 满(只读)** 该位指示RX FIFO 是否已满。 0 = RX FIFO 未满 1 = RX FIFO 已满 **注：**当RX FIFO缓冲区中的数据数量等于16/64后，此位将被置1，否则被硬件清零。
[14]	**RXEMPTY**	**接收 FIFO空(只读)** 此位指示RX FIFO是否为空。 0 = RX FIFO不为空. 1 = RX FIFO为空. **注：**当RX FIFO中最后一个字节被CPU读取后，硬件将对此位置1，UART接收到新数据后此位将被清零。
[13:8]	**RXPTR**	**RX FIFO指针(只读)** 此位指示RX FIFO缓冲区指针。当UART从外部设备接收到一个字节，RXPTR将累加1.当RX FIFO的数据被CPU读取一个字节，RXPTR将递减1. RXPTR显示的最大值是15/63。当RX FIFO的数据达到16/64，RXFULL位将被置1，RXPTR显示0，RX FIFO 当中的数据被CPU读取一个后，RXFULL 将被清零，RXPTR显示15/63。
[7]	**Reserved**	保留.
[6]	**BIF**	**Break中断标志位(只读)** 每当接收到数据输入 (RX) 维持在 “spacing state” (logic 0) 的时间长于一个全字的传输时间（即“start bit” + data 位 + parity + stop 位 的总时间），该位置 1。当 CPU 向该位写 1，该位重置。 0 =没有Break中断产生. 1 =有Break中断产生. **注：**此位只读。但是可以写1清零
[5]	**FEF**	**帧错误标志(只读)** 每当接收到的字符没有有效的 “停止位”（即跟在最后的数据位后或奇偶校验位后的数据为0）该位置 1。 0 =没有帧错误产生 1 =有帧错误产生. **注：**该位只读，但是可以写 1 清零.
[4]	**PEF**	**P奇偶校验错误标志(只读)** 每当接收到的字符没有有效的奇偶校验位，该位置 1。当 CPU 向该位写 1，该位重置。 0 =没有奇偶校验错误产生 1 =有奇偶校验错误产生

		注： 该位只读，但是可以写 1 清零。
[3]	**ADDRDETF**	**RS-485 地址数据检测标志(只读)** 0 =接收到的数据没有地址位标记 (bit 9 ='0'). 1 =接收到的数据有地址位标记(bit 9 ='1'). **注1：**此位用于RS-485功能模式，且ADDRDEN (UART_ALTCTL[15]))位被置1使能地址检测模式 **注2**：该位只读，但是可以写 1 清零。
[2]	**ABRDTOIF**	**自动波特率时间溢出中断 (只读)** 0 = 自动波特率计数器没有溢出. 1 = 自动波特率计数器溢出. **注1:**在自动波特率检测模式下，当自动波特率计数器溢出时，该位将被置为1 。 **注2:** 该位虽为只读位，但可以通过写1清除该位。
[1]	**ABRDIF**	**自动波特率检测中断(只读)** 0 = 自动波特率检测还没有完成. 1 = 自动波特率检测已经完成 **注1**：当自动波特率检测完成时该位将被置为 “1”. **注2:** 该位虽为只读位，但可以通过写1清除该位。
[0]	**RXOVIF**	**RX 溢出错误标志(只读)** 当RX FIFO溢出时被置1 如果接收到的数据数量大于RX_FIFO (UART_DAT) 16 /64字节，该位将被置位. 0 = RX FIFO未溢出. 1 = RX FIFO溢出 **注:**该位只读，但是可以写 1 清零

UART 中断状态 寄存器 (UART_INTSTS)

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

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						DATWKIF	CTSWKIF
15	14	13	12	11	10	9	8
LININT	WKINT	BUFERRINT	RXTOINT	MODEMINT	RLSINT	THREINT	RDAINT
7	6	5	4	3	2	1	0
LINIF	WKIF	BUFERRIF	RXTOIF	MODEMIF	RLSIF	THREIF	RDAIF

位	描述	
[31:18]	**Reserved**	保留.
[17]	**DATWKIF**	**数据唤醒中断标志（只读）** 当数据传入将芯片从**掉电**状态唤醒时，该位将被置位 0 = 芯片保持在**掉电**状态 1 = 数据输入将芯片从**掉电**状态唤醒 **注1**：如果WKDATIEN (UART_INTEN[10])为使能状态，将会产生唤醒中断 **注2**：该位为只读位，但可以写‘1’清零
[16]	**CTSWKIF**	**nCTS唤醒中断标志（只读）** 0 = 芯片保持在**掉电**状态 1 = nCTS将芯片从**掉电**状态唤醒 **注1：**如果WKCTSIEN (UART_INTEN[9])为使能状态，将会产生唤醒中断 **注2：**该位为只读位，但可以写‘1’清零
[15]	**LININT**	**LIN总线中断提示(只读)（UART1/UART2通道有效）** 如果LINIEN (UART_INTEN[8])都被置1，该位置 1 0 =没有LIN总线中断产生 1 =有LIN总线中断产生.
[14]	**WKINT**	**UART唤醒中断标志只读)** 当 DATWKIF 或 CTSWKIF 被置1后该位置1. 0 =没有数据或 nCTS 唤醒中断发生. 1 = 数据或nCTS唤醒中断发生.

[13]	**BUFERRINT**	**Buffer 错误中断提示(只读)** 如果BFERRIEN(UART_INTEN[5] 和BERRIF(UART_INTSTS[5])都被置1，该位置 1 0 =没有buffer错误中断产生. 1 =有buffer错误中断产生.
[12]	**RXTOINT**	**定时溢出中断提示(只读)** 如果TOUTIEN(UART_INTEN[4]) 和RXTOIF(UART_INTSTS[4])都被置1，该位置 1 0 =没有定时溢出中断产生 1 =有定时溢出中断产生.
[11]	**MODEMINT**	**MODEM状态中断提示(只读)** 如果MODEMIEN(UART_INTEN[3] 和MODEMIF(UART_INTSTS[4])都被置1，该位置 1 0 =没有Modem 中断产生. 1 =有Modem 中断产生
[10]	**RLSINT**	**接收线状态中断提示(只读)** 如果RLSIEN (UART_INTEN[2]) 和RLSIF(UART_INTSTS[2])都被置1，该位置 1 0 =没有RLS中断产生 1 =有RLS中断产生
[9]	**THREINT**	**Tr发送保持寄存器空中断提示(只读)** 如果THREIEN (UART_INTEN[1])and THREIF(UART_INTSTS[1])都被置1，该位置 1 0 =没有THRE中断产生 1 =有THRE中断产生.
[8]	**RDAINT**	**接收可用数据中断提示(只读)** 如果RDAIEN (UART_INTEN[0]) 和 RDAIF (UART_INTSTS[0])都被置1，该位置 1 0 =没有RDA中断产生. 1 =有RDA中断产生
[7]	**LINIF**	**LIN总线标志（只读）（ UART1/UART2通道有效）** 当LIN从机报头侦测(SLVHDETF (UART_LINSTS[0] =1))，LIN break 侦测(BRKDETF(UART_LINSTS[9]=1))，位错误侦测(BITEF(UART_LINSTS[9]=1，LIN从机ID校验错误(SLVIDPEF(UART_LINSTS[2] = 1)或LIN从机报头错误侦测(SLVHEF (UART_LINSTS[1]) = 1)，该位置1。如果LIN_ IEN (UA_IER [8])被使能，LIN中断产生。 0 = SLVHDETF, BRKDETF, BITEF, SLVIDPEF 和SLVHEF中没有任何一个标志产生 1 = SLVHDETF, BRKDETF, BITEF, SLVIDPEF 和SLVHEF中至少有一个标志产生 **注:**该位只读。当SLVHDETF(UART_LINSTS[0]), BRKDETF(UART_LINSTS[8]), BITEF(UART_LINSTS[9]), SLVIDPEF (UART_LINSTS[2]), SLVHEF(UART_LINSTS[1]) 和SLVSYNCF(UART_LINSTS[3])都被清0时，该位清0。
[6]	**WKIF**	**UART唤醒标志（只读）** 当DATWKIF (UART_INTSTS[17]) 或CTSWKIF(UART_INTSTS[16])为1时，该位将被置位。 0 = 没有DATWKIF 和 CTSWKIF产生 1 = DATWKIF 或 CTSWKIF 为1 **注**：该位为只读。当写1到DATWKIF DATWKIF (UART_INTSTS[17]) and CTSWKIF (UART_INTSTS[17])，将DATWKIF (UART_INTSTS[17]) and CTSWKIF (UART_INTSTS[17])清零时，该位将被清零

[5]	**BUFERRIF**	**Buffer 错误中断标志 (只读)** 当TX FIFO或RX FIFO溢出 (TXOVIF (UART_FIFOSTS[24]) 或 RXOVIF (UART_FIFOSTS[0])为1)该位置1. 当BERRIF (UART_INTSTS[5])被置位, 传输出错.如果BFERRIEN (UART_INTEN [8]))被使能, buffer 错误中断将产生 0 = 没有buffer 错误中断标志产生 1 = 有buffer 错误中断标志产生 **注：**该位为只读。当写1到RXOVIF(UART_FIFOSTS[0]) 和 TXOVIF(UART_FIFOSTS[24])，将D RXOVIF(UART_FIFOSTS[0]) 和 TXOVIF(UART_FIFOSTS[24])清零时，该位也将被清零.
[4]	**RXTOIF**	**定时溢出中断标志（只读）** 当 RX FIFO非空而且 RX FIFO无活动发生，定时溢出计数器等于TOIC 时，该位置位。如果TOUTIEN (UART_INTEN [4])使能，定时溢出中断将产生。 0 =没有定时溢出中断标志产生. 1 =有定时溢出中断标志产生 **注：**该位只读，用户可以读UART_DAT（RX处于活动状态）清除该位
[3]	**MODEMIF**	**MODEM中断标志(只读)** 当n CTS管脚状态有改变(CTSDETF (UART_MODEMSTS[0]) = 1)，该位置位。如果MODEMIEN (UART_INTEN [3])使能，Modem 中断将产生。 0 =没有Modem 中断标志产生. 1 =有Modem 中断标志产生. **注：**该位只读，当向CTSDETF(UART_MODEMSTS[0])写1将CTSDETF清零后，该位也复位为0。
[2]	**RLSIF**	**接收线中断标志(只读)** 当 RX 接收数据发生 parity error, framing error 或 break error（至少 BIF(UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]) 和PEF(UART_FIFOSTS[4])三位中有一位被置位）该位置位。如果RLSIEN (UART_INTEN [2])使能，RLS 中断将产 0 =没有RLS中断标志产生 1 =有RLS中断标志产生. **注1:** 在 RS-485功能模式，当“接收检测任一接收到的地址字节字符 (bit9 = ‘1’) 位”时，该位也会置1，同时，ADDRDETF (UART_FIFOSTS[3])位也被置1 **注2:** 该位只读，当BIF (UART_FIFOSTS[6]), FEF(UART_FIFOSTS[5]) and PEF(UART_FIFOSTS[4])三位都被清除后，该位重置为 0 **注3:** 在RS-485 功能模式下,该位只读，且当(UART_FIFOSTS[6]) , FEF(UART_FIFOSTS[5]) 和 PEF(UART_FIFOSTS[4]) 和 ADDRDETF (UART_FIFOSTS[3]) 所有位被清零，该位清零
[1]	**THREIF**	**发送保持寄存器空中断标志(只读)** 当TX FIFO中最后数据传输到发送移位寄存器时，该位置位。如果THREIEN (UART_INTEN[1])使能，THRE 中断将产生。 0 =没有THRE中断标志产生 1 =有THRE中断标志产生. **注:** 该位只读，当写数据到UART_DAT（TX FIFO 非空）时，该位将被清除
[0]	**RDAIF**	**接收可用数据中断标志(只读)** 当 RX FIFO 中的数据数量等于 RFITL时，RDAIF(UART_INTSTS[0])将被置位。如果RDAIEN (UART_INTEN [0])使能，RDA 中断将产生。 0 =没有RDA中断标志产生.

		1 =有RDA中断标志产生. **注**：该位只读，当 RX FIFO 的未读数据低于阈值(RFITL(UART_FIFO[7:4])时，该位将被清除

UART 溢出寄存器 (UART_TOUT)

寄存器	偏移地址	R/W	描述	复位值
UART_TOUT	UARTx_BA+0x20	R/W	UART 溢出寄存器	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
DLY							
7	6	5	4	3	2	1	0
TOIC							

位	描述	
[31:16]	Reserved	保留.
[15:8]	DLY	**TX延迟时间值** 该域用于设置上一次停止位和下一次开始位之间的传输延迟时间
[7:0]	TOIC	**定时溢出中断比较器** 当RX FIFO接收到一个新的数据字，定时溢出计数器重置并开始计数 (计数时钟 = 波特率)。一旦超时计数器的内容等于超时中断比较器(TOIC (UART_TOUT[7:0]))，如果此时(RXTOINT(UART_INTSTS[12]))为使能，则接收超时中断(RXTOINT(UART_INTSTS[12]))产生。接收到新的数据或RX FIFO为空将把RXTOINT(UART_INTSTS[12])清零。为了避免接收超时中断在接收到一个字符就立即产生，TOIC的值必须设置在 40 到 255 之间。例如，如果TOIC为 40，当 UART 传输设置为1 位停止位且无奇偶校验位时，在4个字符时间长度后还没收到数据，超时中断将产生。

UART 波特率分频寄存器 (UART_BAUD)

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

31	30	29	28	27	26	25	24
Reserved		BAUDM1	BAUDM0	EDIVM1			
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
BRD							
7	6	5	4	3	2	1	0
BRD							

位	描述	
[31:30]	**Reserved**	保留.
[29]	**BAUDM1**	**波特率模式选择BIT 1** 该bit为波特率模式选择bit 1。UART 提供三种波特率计算方式。 该位和BAUDM0 (UART_BAUD[28])组合选择波特率计算方式。 详细描叙在表 6.10-2中. **注：**IrDA模式时必须选择Mode0
[28]	**BAUDM0**	**波特率模式选择BIT 0** 该bit为波特率模式选择bit 0。UART 提供三种波特率计算方式。 该位和BAUDM0 BAUDM1 (UART_BAUD[29])组合选择波特率计算方式。 详细描叙在表 6.10-2.中
[27:24]	**EDIVM1**	**波特率模式1的扩展分频** 该域用于波特率计算模式1，对于波特率计算模式0和模式2 无效。详细描叙在表 6.10-2.中
[23:16]	**Reserved**	保留.
[15:0]	**BRD**	波特率分频 该域用于波特率分频。详细描叙在表 6.10-2.中

UART_IrDA 控制寄存器 (UART_IRDA)

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

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	RXINV	TXINV	Reserved			TXEN	Reserved

位	描述	
[31:7]	**Reserved**	保留.
[6]	**RXINV**	**IrDA接收輸入信号反向** 0 =接收输入信号不反向. 1 =接收输入信号反向（默认）.
[5]	**TXINV**	**IrDA发送信号反向** 0 =发送信号不反向（默认）. 1 =发送信号反向
[4:2]	**Reserved**	保留.
[1]	**TXEN**	**IrDA 接收/发送选择使能位** 0 = IrDA发送禁止，接收使能（默认） 1 = IrDA发送使能，接收禁止 在 IrDA功能模式 (FUNCSEL(UART_FUNCSEL[1:0])=10),收到的第1个数据是不完整的 所以在IrDA接收使能时第1个时先跳过.
[0]	**Reserved**	保留.

注: 在IrDA 模式下, BAUDM1 (UART_BAUD [29])寄存器必须禁止， (波特率公式必须是Clock / (16 * (BRD + 2))

UART复用控制/状态寄存器(UART_ALTCTL)

寄存器	偏移地址	R/W	描述	复位值
UART_ALTCTL	UARTx_BA+0x2C	R/W	UART复用控制/状态寄存器	0x0000_000C

31	30	29	28	27	26	25	24
ADDRMV							
23	22	21	20	19	18	17	16
Reserved			ABRD位		ABRDEN	ABRIF	Reserved
15	14	13	12	11	10	9	8
ADDRDEN	Reserved				RS485AUD	RS485AAD	RS485NMM
7	6	5	4	3	2	1	0
LINTXEN	LINRXEN	Reserved		BRKFL			

位	描述	
[31:24]	**ADDRMV**	**地址匹配值** 该位包括 RS-485 地址匹配值. **注：**该域用于 RS-485 自动地址检测模式.
[23:21]	**Reserved**	保留.
[20:19]	**ABRD位**	**自动波特率检测位长** 00 = 1-位时长，从起始位到第一个 上升沿。 输入数据格式应该为0x01。 01 = 2-位时长，从起始位到第一个 上升沿。 输入数据格式应该为0x02。 10 = 4-位时长，从起始位到第一个 上升沿。 输入数据格式应该为0x08。 11 = 8-位时长，从起始位到第一个 上升沿。 输入数据格式应该为0x80。 **注**：计算的位数包括起始位。.
[18]	**ABRDEN**	**自动波特率检测使能位** 0 = 自动波特率检测功能禁止. 1 = 自动波特率检测功能使能 **注**：自动检测结束后，该位将被自动清零。
[17]	**ABRIF**	**自动波特率中断标志（只读）** 当自动波特率检测结束，或者自动波特率计数器发生溢出，如果ABRIEN(UART_INTEN[18])为1，那么自动波特率中断将会产生. **注:** 该位为只读位，但可以通过写“1”到ABRDTOIF (UART_FIFOSTS[2]) 和ABRDIF(UART_FIFOSTS[1])将该位清零。
[16]	**Reserved**	保留.
[15]	**ADDRDEN**	**RS-485 地址检测使能位** 该位用于使能 RS-485 地址检测模式.

		0 =地址检测模式禁止. 1 =地址检测模式使能. **注：** 该位适用于 各种RS-485 操作模式
[14:11]	**Reserved**	保留.
[10]	**RS485AUD**	**RS-485 自动方向模式 (AUD)** 0 =RS-485自动方向操作模式（AUD)禁止. 1 =RS-485自动方向操作模式（AUD)使能 **注：** 仅在RS-485_AAD 或 RS-485_NMM 操作模式有效
[9]	**RS485AAD**	**RS-485自动地址检测操作模式 (AAD)** 0 = RS-485自动地址方向操作模式 (AAD)禁止. 1 =RS-485自动地址方向操作模式 (AAD)使能. **注：** 在 RS-485_NMM 操作模式下无效
[8]	**RS485NMM**	**RS-485普通多点操作模式 (NMM)** 0 = RS-485 普通多点操作模式 (NMM)禁止. 1 = RS-485 普通多点操作模式 (NMM)使能 **注：** 在 RS-485_AAD 操作模式下无效.
[7]	**LINTXEN**	**LIN TX Break 模式使能（UART1/UART2有效）** 0 = LIN TX Break 模式禁止 1 = LIN TX Break 模式使能. **注：** 当 TX break 域传输操作完成后，该位将被自动清除.
[6]	**LINRXEN**	**LIN RX 使能（UART1/UART2有效）** 0 =LIN RX 模式禁止. 1 = LIN RX 模式使能
[5:4]	**Reserved**	保留.
[3:0]	**BRKFL**	**UART LIN LIN Break 域长度（UART1/UART2有效）** 该域表示一个 4-位 LIN TX break 域数量 **注1:** 该break 域长度为BRKFL + 1 **注2:** 根据LIN 规范, 复位值是0xC (break 域长度= 13).

UART 功能选择寄存器 (UART_FUNCSEL)

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

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

位	描述	
[31:2]	Reserved	保留.
[1:0]	FUNCSEL	**功能选择** 00 = UART功能. 01 = LIN 功能（UART1/UART2有效） 10 = IrDA功能 11 = RS-485功能 .**注:** 在 IrDA功能模式 (FUNCSEL(UART_FUNCSEL[1:0])=10),收到的第1个数据是不完整的所以在IrDA接收使能时第1个时先跳过..

UART LIN 控制寄存器 (UART_LINCTL) (UART0无效)

寄存器	偏移地址	R/W	描述	复位值
UART_LINCTL	UARTx_BA+0x34	R/W	UART LIN 控制寄存器 *	0x000C_0000

注[*]:UART0 无效.

31	30	29	28	27	26	25	24
PID							
23	22	21	20	19	18	17	16
HSEL		BSL		BRKFL			
15	14	13	12	11	10	9	8
Reserved			BITERREN	RXOFF	BRKDETEN	IDPEN	SENDH
7	6	5	4	3	2	1	0
Reserved			MUTE	SLVDUEN	SLVAREN	SLVHDEN	SLVEN

位	描述	
[31:24]	**PID**	**LIN PID 位** 在LIN功能模式下，该域包含LIN帧ID值。帧ID的校验可以由软件或硬件产生，这取决于IDPEN (UART_LINCTL[9])的设置。 如果校验由硬件产生， 用户填ID0~ID5, (LIN_PID[24:29]，硬件会计算P0(PID[30]) 和P1(PID[31])，否则用户必须填帧ID和校验位 **注1:** 用户可以填任何 8-bit 值到该域，位 24 表示 ID0 (LSB 优先) **注2:** 该域可以用在 LIN 主机模式或从机模式
[23:22]	**HSEL**	**LIN帧报头选择** 00 = LIN帧报头包括 “break空间” 01 = LIN 帧报头包括 “break 空间” 和 “同步空间””. 10 = LIN 帧报头 包括 “break空间”, “同步空间” “帧ID”. 11 = 保留 **注：** LIN 主机模式时，该位用于发送帧报头LIN_SHD (UA_LIN_CTL [8]) = 1)，而在LIN从机模式时，用于指示从mute模式退出条件(LIN_MUTE_EN (UA_LIN_CTL[4] = 1)。 该位可以用于LIN主机发送帧头(SENDH (UART_LINCTL[8])) = 1)或用于提示 LIN 从机从mute模式退出条件(LIN_MUTE_EN (UA_LIN_CTL[4] = 1)。
[21:20]	**BSL**	**LIN Break/同步分隔符长度** 00 = LIN break/同步分隔符长度为1 位时间. 10 = LIN break/同步分隔符长度为2 位时间. 10 = LIN break/同步分隔符长度为3 位时间. 11 = LIN break/同步分隔符长度为4 位时间. **注:** 该位用于LIN 主机发送帧报头
[19:16]	**BRKFL**	**LIN Break 空间长度** 该域表示一个4-位 LIN TX break空间数量 **注1:** 这个寄存器是BRKFL的影子寄存器，用户可以通过设置BRKFL (UART_ALTCTL[3:0])

		或BRKFL (UART_LINCTL[19:16])对break空间的长度进行读/写。 **注2**: 该 break空间长度为BRKFL + 1 **注3:** 根据LIN 规范，复位值是 12 (break空间长度= 13).
[15:13]	**Reserved**	保留.
[12]	**BITERREN**	**位错误侦测使能** 0 =位错误侦测功能禁止. 1 =位错误侦测使能. **注:** 在 LIN功能模式下，当位错误发生时，BITEF (UART_LINSTS[9])标志会被置位。如果LINIEN (UART_INTEN[8]) = 1，会产生一个中断
[11]	**RXOFF**	**LIN接收器禁止位** 如果接收器使能(RXOFF (UART_LINCTL[11]) = 0)，所有接收到的数据字节会被接受并存储在RX-FIFO，如果接收器禁止(RXOFF (UART_LINCTL[11] = 1)，接收到的所有数据会被忽略 0 = LIN 接收器禁止. 1 = LIN 接收器使能. **注:** 该位仅工作在 LIN功能模式下有效(FUNCSEL (UART_FUNCSEL[1:0]) = 01)
[10]	**BRKDETEN**	**LIN Break 空间检测使能位** 当检测到显性位长度连续超过11位，后跟一个分隔符，在break空间结束时BRKDETF (UART_LINSTS[8])标志被置位。如果LINIEN (UART_INTEN [8])=1，将会产生中断。 0 = LIN break 空间检测禁止 1 = LIN break 空间检测使能.
[9]	**IDPEN**	**LIN ID 校验使能位** 0 = LIN 帧ID校验禁止. 1 = LIN 帧ID校验使能. **注1:** 该位可以用于LIN主机发送帧头域(SENDH (UART_LINCTL[8])) = 1 且HSEL (UART_LINCTL[23:22]) = 10)或用来使能 LIN 从机对接收到的帧 ID进行奇偶校验检查。 **注2:**该位仅在发送器报头是在HSEL (UART_LINCTL[23:22]) = 10时有用。
[8]	**SENDH**	**LIN TX 发送帧报头使能位** LIN TX 帧报头可以是“break 域” 或 “break 和同步域” 或 “break, 同步和帧 ID 域”，这取决于HSEL (UART_LINCTL[23:22])的设置 0 = LIN TX 帧报头发送禁止. 1 = LIN TX 帧报头发送使能. **注1:** 这些 寄存器是SENDH (UART_ALTCTL [7])的影子寄存器;用户可以通过SENDH (UART_ALTCTL [7]) 或SENDH (UART_LINCTL [8])进行读写设置 **注2:** 当发送器帧头域 (可能是 “break” 或 “break + 同步” 或 “break + 同步 + 帧 ID”通过HSEL (UART_LINCTL[23:22]) 选择) 传输操作完成, 该位会自动清0.
[7:5]	**Reserved**	保留.
[4]	**MUTE**	**LIN Mute 模式使能位** 0 = LIN Mute 模式禁止 1 = LIN Mute 模式使能. **注:** 退出Mute模式条件，该域的控制和相互影响在章节 6.10.5.9 (LIN从机模式).
[3]	**SLVDUEN**	**LIN从机分频器更新方式使能**

		0 = UA_BAUD 被软件更新(如果同时没有自动重同步发生) 1 = UA_BAUD在下次接收到字符的时候更新。用户必须在接收校验之前设置该位. **注1:** 该位仅在 LIN 从模式有效(SLVEN (UART_LINCTL[0]) = 1) **注2**: 该位用于LIN 从机自动重同步模式. (对于非自动重同步模式，该位需维持清零) **注3:** 该域的控制和相互作用在章节 6.10.5.9 (自动再同步的从机模式).
[2]	**SLVAREN**	**LIN 从机自动重同步模式使能位** 0 = LIN从机自动重同步模式禁止 1 = LIN从机自动重同步模式使能. **注1:** 该位仅在 LIN 从机模式(SLVEN (UART_LINCTL[0]) = 1) 有效. **注2:** 当工作在自动重同步模式，波特率的设置必须是模式2 (BAUD_M1 (UA_BAUD [29]) 和 BAUD_M0 (UA_BAUD [28])必须是 1). **注3:** 该域的控制和相互作用在章节 6.10.5.9 (自动再同步的从机模式).
[1]	**SLVHDEN**	**LIN 从机报头侦测使能位** 0 = LIN 从机报头检测禁止. 1 = LIN 从机报头检测使能. **注1:** 该位仅在 LIN 从机模式 (SLVEN (UART_LINCTL[0]) = 1) 有效. **注2:** 在 LIN功能模式， 当侦测报头(break + sync + frame ID), SLVHDETF (UART_LINSTS [0])标志会被置位，如果LINIEN (UART_INTEN[8]) = 1，会产生一个中断
[0]	**SLVEN**	**LIN 从模式使能位** 0 = LIN 从模式禁止 1 = LIN 从模式使能

UART LIN 状态寄存器 (UART_LINSTS) (UART0 无效)

寄存器	偏移地址	R/W	描述	复位值
UART_LINSTS	UARTx_BA+0x38	R/W	UART LIN 状态寄存器 *	0x0000_0000

注[*]: Not Available in UART0 Channel.

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						BITEF	BRKDETF
7	6	5	4	3	2	1	0
Reserved				SLVSYNCF	SLVIDPEF	SLVHEF	SLVHDETF

位	描述	
[31:10]	**Reserved**	保留.
[9]	**BITEF**	**位错误侦测状态标志 (只读)**. TX传输状态，硬件会监视总线状态，如果输入管脚 (SIN) 状态不等于输出管脚 (SOUT) 状态, BITEF (UART_LINSTS[9])位会被置位。 当发生位错误，如果LINIEN (UART_INTEN[8]) = 1，会产生一个中断 **注1:** 该位只读，向该位写1清0 **注2:** 该位仅当位错误侦测功能使能时有效(BITERREN (UART_LINCTL [12]) = 1)
[8]	**BRKDETF**	**LIN Break 检测标志(只读)** 当侦测到一个break，该位由硬件置位。可通过软件写1将该位清零。 0 = LIN break 没有被检测到 1 = LIN break 被检测到. **注1:** 该位只读，向该位写1清0. **注2:**该位仅当 LIN break侦测功能使能时有效(BRKDETEN (UART_LINCTL[10]) =1)
[7:4]	**Reserved**	保留.
[3]	**SLVSYNCF**	**LIN从机同步域.** 该位表示在自动重同步模式， LIN同步域正在被分析。 当接收器报头侦测到一些错误，用户必须通过写1到该位复位内部电路来重新搜索新的帧报头 0 =当前字符不在 LIN同步状态. 1 =当前字符在 LIN 同步状态. **注1：** 该位仅在 LIN 从机模式有效 (LINS_EN = 1) **注2**: 该位只读，向该位写1清0 **注3:** 当向该位写1，硬件会重载初始波特率并重新搜索新的帧报头
[2]	**SLVIDPEF**	**LIN从机 ID 校验错误标志 (只读)**

		当接收帧ID校验错误，该位由硬件置1 0 = 无效. 1 = 收到的帧ID校验错误. **注1:** 该位只读，向该位写1清0.. **注2:**该位仅在 LIN 从机模式 (SLVEN (UART_LINCTL [0])= 1)，并使能LIN帧ID校验功能 IDPEN (UART_LINCTL [9])时 有效
[1]	**SLVHEF**	**LIN从机报头错误标志 (只读).** 在LIN从机模式，当侦测到一个LIN报头错误时，该位由硬件置1，向该位写1清0。报头错误包括“break 间隔符太短（小于0.5位的时间）”， “在同步域或在识别域中帧错误，” “在非自动重同步模式同步域数据不是0x55”， “自动重同步模式同步域偏离错误”， “自动重同步模式同步域测量超时”和“LIN 报头接收超时’ 0 = LIN未检测到报头错误 1 = LIN检测到报头错误. **注1:** 该位只读，向该位写1清0 **注2:** 该位仅在 LIN 从机模式(SLVEN (UART_LINCTL [0]) = 1)，并且使能LIN 从机报头侦测功能(SLVHDEN (UART_LINCTL [1]))时有效。
[0]	**SLVHDETF**	**LIN 从机报头侦测标志（只读）** 在LIN从机模式，当侦测到一个LIN报头，该位由硬件置位，通过软件写1清0 0 = LIN 报头没有被侦测到. 1 = LIN 报头被侦测到(break + 同步+ 帧ID) **注1:** 该位只读，向该位写1清0 **注2:** 该位仅在 LIN 从机模式(SLVEN (UART_LINCTL [0]) = 1)，并且使能LIN 从机报头侦测功能(SLVHDEN (UART_LINCTL [1]))时有效。 **注3:** 当使能ID 校验IDPEN (UART_LINCTL [9])，如果硬件侦测到完整的报头(“break + sync + frame ID”)，SLVHDETF会被置位，无论帧ID是否正确

6.11 I²C 串行接口控制器 (主机/从机)

6.11.1 概述

I²C 是双线，双向串行总线，通过简单有效的连线方式实现设备间的数据交换。I2C标准是多主机总线，包括冲突检测和仲裁，以防止在两个或多个主机同时尝试控制总线时发生数据损坏。

在I²C总线上，数据通过时钟线SCL和数据线SDA在主从机间逐一字节同步传送。. 每个字节数据长度是8位. 一个SCL 时钟脉冲传输一个数据位，数据由最高位MSB 开始传输，每个字节传输后跟随一个应答位。每个位在SCL 为高时采样，因此，SDA 线可能在SCL 为低时改变，但在SCL 为高时必须保持稳定。当SCL 为高时，SDA 线上的跳变视为一个命令(START or STOP) 更多关于I²C总线时序的细节请参考下图.

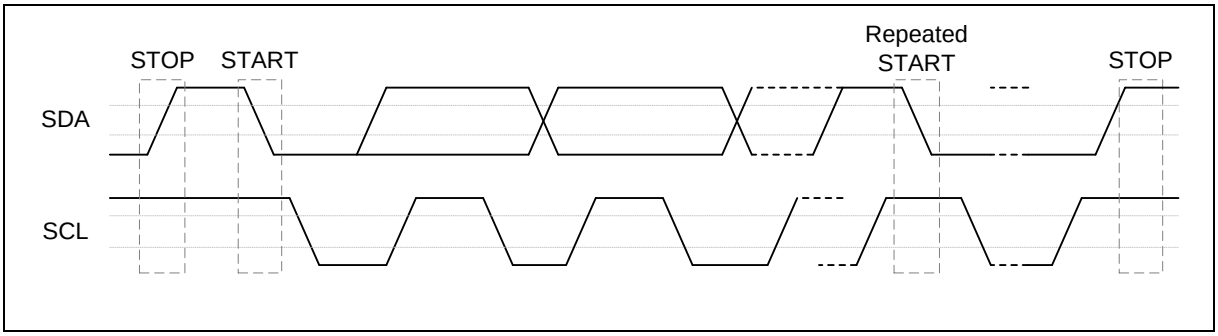


图 6.11-1 I²C 总线时序

片上I²C外设提供了一个符合I²C总线规范的串行接口。I²C端口自动处理字节传输。通过设置寄存器I2CEN (I2C_CTL[6])为'1'可使能该端口。I²C 硬件接口通过数据线SDA 和时钟线 SCL两个管脚连到I²C总线。I²C操作两个管脚需要上拉电阻，为开漏脚，当I/O 管脚作为I²C 端口使用时，用户必须事先设定I/O 管脚为I²C功能。.

6.11.2 特性

NUC505 系列提供2通道I²C总线. I2C通过SDA 及SCL两条线与连接在总线上的设备传输信息总线的主要特征有:

- 支持主机/从机模式和广播模式
- 主从机之间双向数据传输
- 总线支持多主机
- 多主机间同时传输数据仲裁，避免总线上串行数据损坏
- 总线采用串行同步时钟，可实现设备之间以不同的速率传输
- 串行时钟同步可用握手方式去暂停和恢复串行传输
- 内建14位溢出定时器，当I2C总线中止且定时器溢出，产生I2C中断.
- 可配置不同时钟以适用于可变速率控制
- 支持7位从地址模式
- 支持支持多地址识别（4组从机地址带屏蔽选项）
- 支持地址匹配唤醒功能

6.11.3 基本配置

I^2C 管脚在 SYS_GPA_MFPH, SYS_GPB_MFPL, SYS_GPB_MFPH and SYS_GPD_MFPL 多功能寄存器中配置.

多功能寄存器 SYS_GPA_MFPH[26:24], SYS_GPB_MFPL[2:0] 和 SYS_GPD_MFPL[2:0] 对应 I2C0_SCL

多功能寄存器s SYS_GPA_MFPH[30:28], SYS_GPB_MFPL[6:4] 和 SYS_GPD_MFPL[6:4] 对应 I2C0_SDA

多功能寄存器 SYS_GPA_MFPH[10:8], SYS_GPB_MFPH[10:8] 和 SYS_GPB_MFPL[26:24]对应 I2C1_SCL

多功能寄存器 SYS_GPA_MFPH[14:12], SYS_GPB_MFPH[14:12] 和 SYS_GPB_MFPL[30:28] 对应 I2C1_SDA

6.11.4 功能描述

6.11.4.1 I^2C 协议

通常标准通讯有以下4部分:

1) 起始信号或者重复起始信号
2) 从机地址传输和读/写位传输
3) 数据传送

4) 停止信号

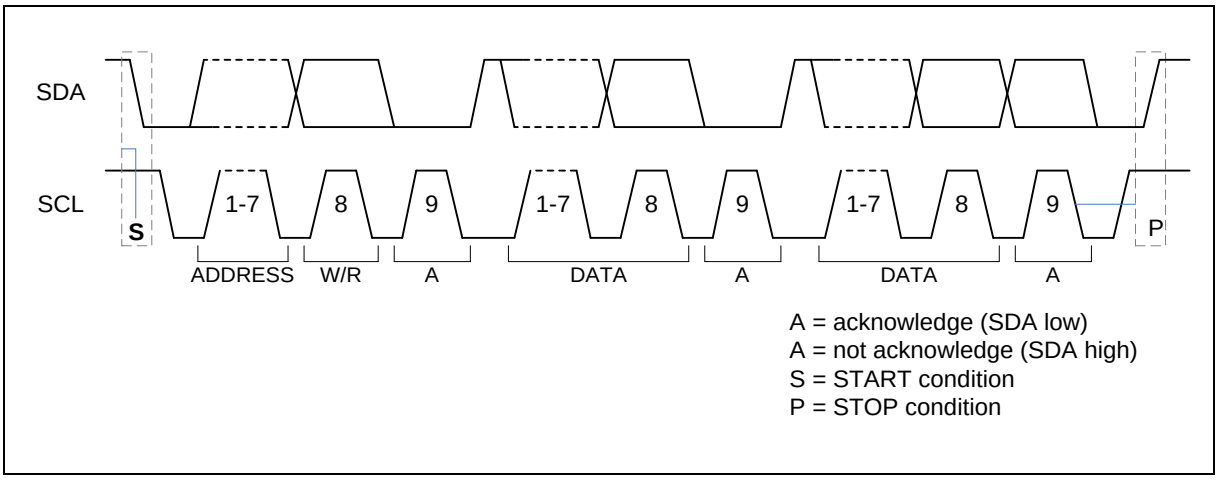


图 6.11-2 I²C 协议

6.11.4.1.1 总线上的数据传输

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

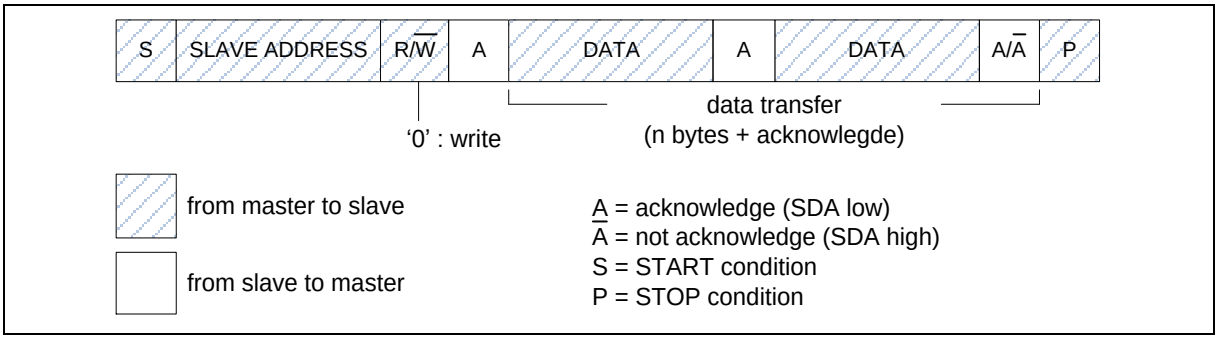


图 6.11-3 主机发数据到从机

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

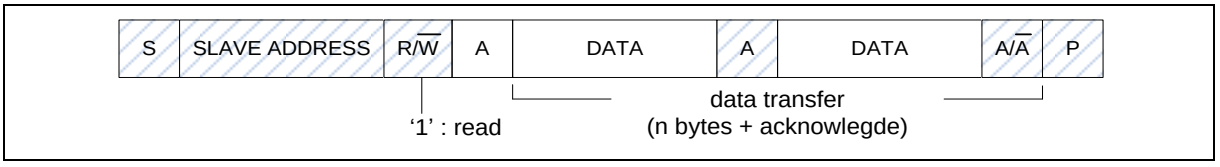


图 6.11-4主机读从机数据

6.11.4.2 开始或重复开始信号和停止信号

当总线处于释放/空闲状态下，说明没有主机设备占用总线（SDA 与SCL线同时为高），主机可以

通过发送起始(START)信号来发起传输过程。起始信号，通常表示为"S"位，当SCL 线为高时，SDA 线上信号由高至低变化，就被定义为起始信号。起始信号表示一个新的数据传输的开始。.

重复起始是另一个起始信号替代停止信号，主机用来这个方法的目的是在不释放总线情况下，对一个或多个设备能读/写操作，而不让操作被打断。用这个方法主机跟其他从机或同一个从机在不同方向传输（例如，写设备到读设备）不用释放总线。

主机通过产生一个停止信号来停止传输，停止信号可以用P-位表示，当SCL 线为高时，SDA线上信号由低至高变化，就被定义为停止信号

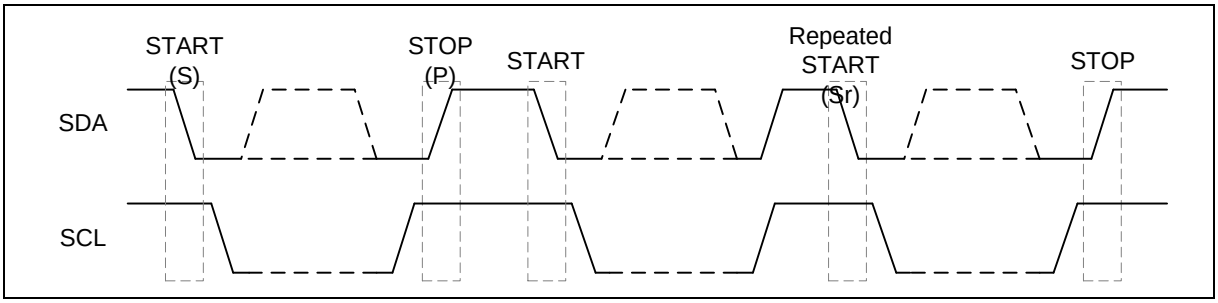


图 6.11-5 开始和停止条件

6.11.4.3 从机地址传送

起始信号后立即传输的第一个字节就是从机地址(SLA)。这是一个7位呼叫地址，后面跟随一个读/写(R/W) 位。R/W 位为从机数据传输方向的信号。系统中从机不能有2个相同的地址，仅有被主机寻址的从机设备才会通过在第9个SCL 时钟周期时将SDA 拉低作为返回应答信号。

6.11.4.4 数据传送

当从机设备地址被成功识别到，就可以根据R/W 位所决定的方向按一字节一字节方式进行数据传输。每个字节传输完后，紧接着的第9个SCL 时钟周期会有一个应答信号位。如果从机上产生无应答信号(NACK)，主机可以产生停止信号来中止数据传输或者产生重复起始信号开始新一轮数据传输周期。.

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

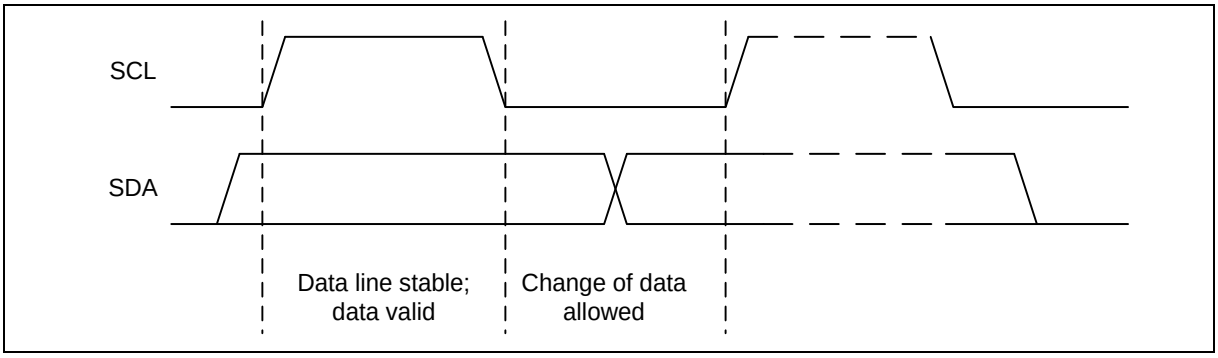


图 6.11-6 I²C 总线位传送

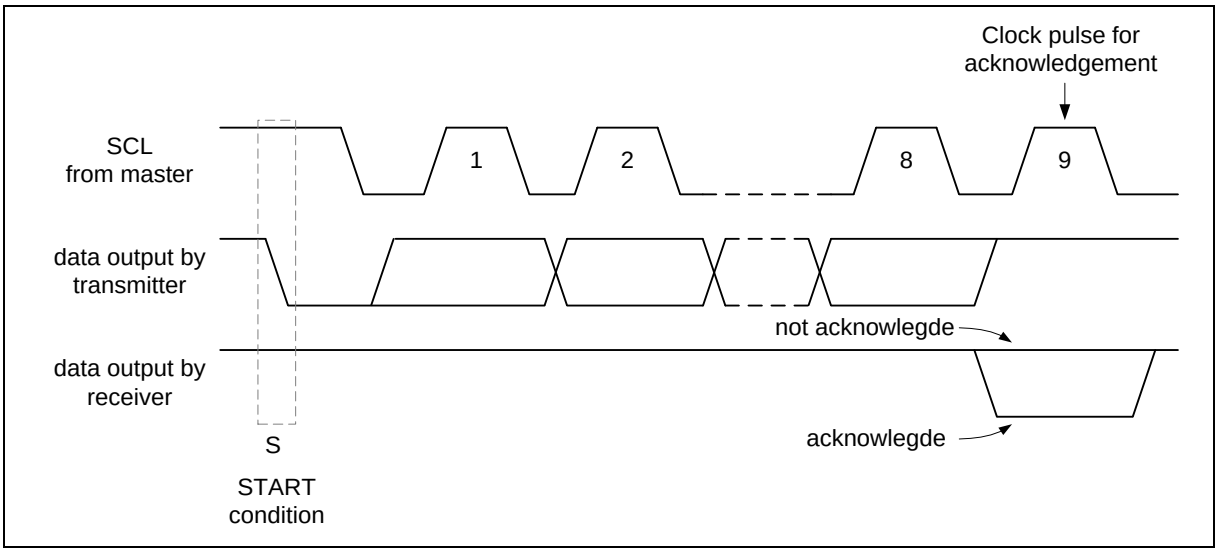


图 6.11-7 I²C总线应答

6.11.5 操作模式

片上I²C端口支持三种工作模式：主机模式，从机模式和广播呼叫模式。

应用中，I²C 端口可以作为主机和从机。在从机模式，I2C端口硬件会查找自身从机地址和广播呼叫地址，如果这两个地址的任一个被检测到，并且从机想要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个时钟发出，此时，如果中断使能，则主机和从机设备上都会发生一次中断请求。在微控制器想要成为总线主机时，在进入主机模式之前，硬件需等待到总线空闲，以保证合理的从机动作不会被打断。在主机模式下，如果总线仲裁丢失，I2C端口立即切换到从机模式，并可以在同一次串行传输过程中检测自身的从机地址。

为控制I²C总线的各种模式传输，用户需要按照寄存器I2C_STATUS的当前状态码来设置I2C_CTL, I2C_DAT寄存器。换句话说，每个I2C总线动作都要检查I2C_STATUS寄存器的当前状态，然后再设置I2C_CTL, I2C_DAT寄存器执行总线动作。最后，通过I2CSTATUS检查响应状态。

在寄存器I2C_CTL [3]的SI标志清除后，寄存器I2C_CTL的这些位STA, STO 和 AA 用来控制I²C硬件的下一个状态。当完成一个新的动作，I2C_STATUS的状态代码将被更新，I2C_CTL [3]寄存器的SI标志将被设置。如果I²C中断控制位EI (I2C_CTL [7])被设置，新状态代码对应的动作或者软件将在中断服务程序中被执行。

下图显示当前I²C状态码是0x08，然后通过设置I2C_DATA=SLA+W和(STA,STO,SI,AA) = (0,0,1,x)发送地址到I²C总线。如果总线上有从机匹配相应的地址且返回ACK，I2C_STATUS状态码更新为0x18.

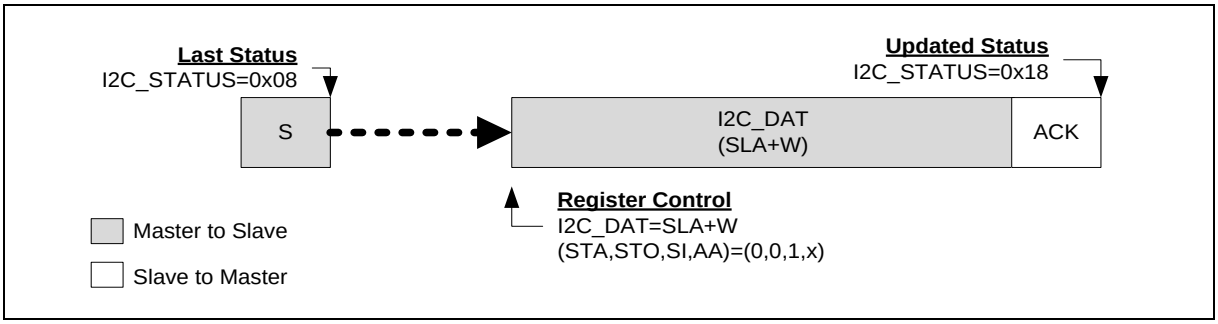


图 6.11-8通过当前状态控制I²C 总线

6.11.5.1 主机模式

下图中，是I²C主机模式所有可能的协议。用户需要遵循恰当的流程来实现I²C协议。

换句话，用户可以发送一个起始信号到总线，I²C总线将设置成主机传输模式(图 6.11-9) 或主机接收模式(图 6.11-10) 起始信号设置成功后新的状态码将是0x08。起始信号后，用户可以发送从机地址，读/写位，数据和重复起始，停止来执行I²C协议。

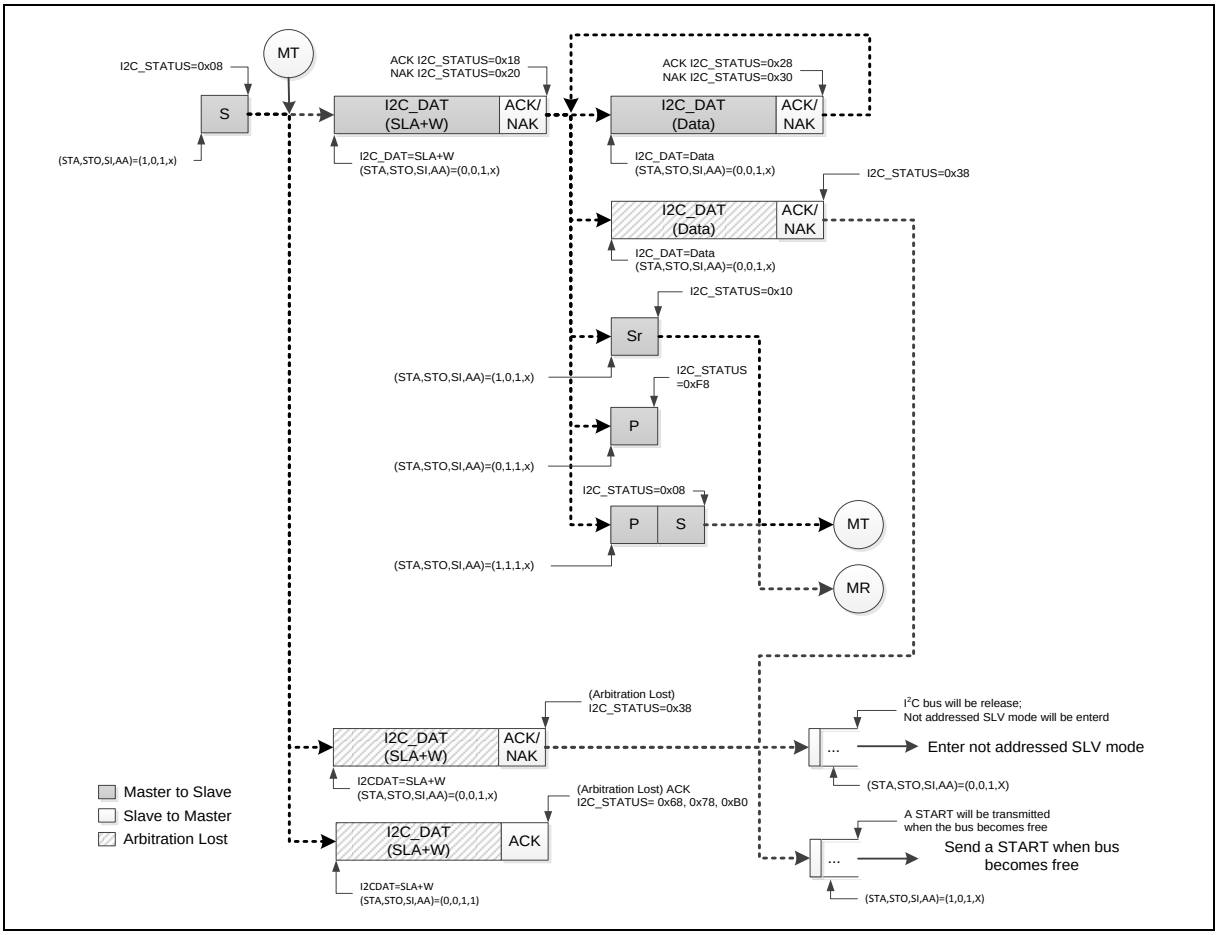


图 6.11-9主机发送控制流程

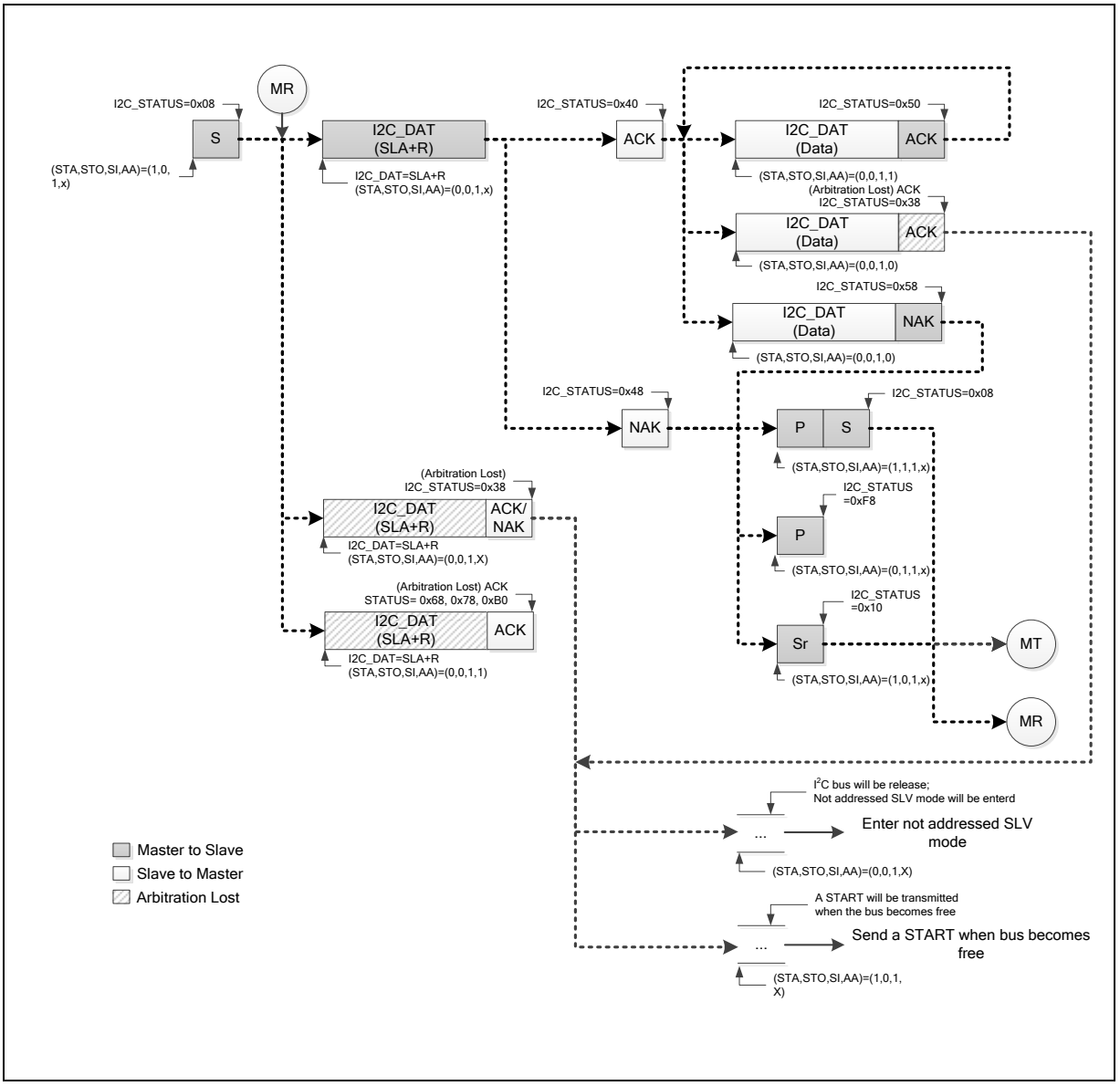


图 6.11-10 主机接收控制流程

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

6.11.5.2 从机模式

复位后默认情况下，I²C不会被寻址，并且不会识别I²C总线上的地址。用户可以通过I2C_ADDRn(n=0~3)设置从机地址和设置 (STA, STO, SI, AA) = (0, 0, 1, 1)来让I²C识别主机发送的地址. 遵循图所示为I2C从机模式的所有可能的流程。

如果在主机模式总线仲裁丢失，I²C端口立即切换到从机模式，并且在同一串行传输中识别自有的从

机地址。如果在仲裁丢失后识别到地址是SLA+W（主机想写数据到从机），状态码是0x68。如果在仲裁丢失后识别到地址是SLA+R（主机向从机读数据），状态码是0xB0。.

I²C通信期间，在从机模式下，当对SI标志SI （I2C_CTL[3]）写‘1’清除时，SCL 时钟将被释放。

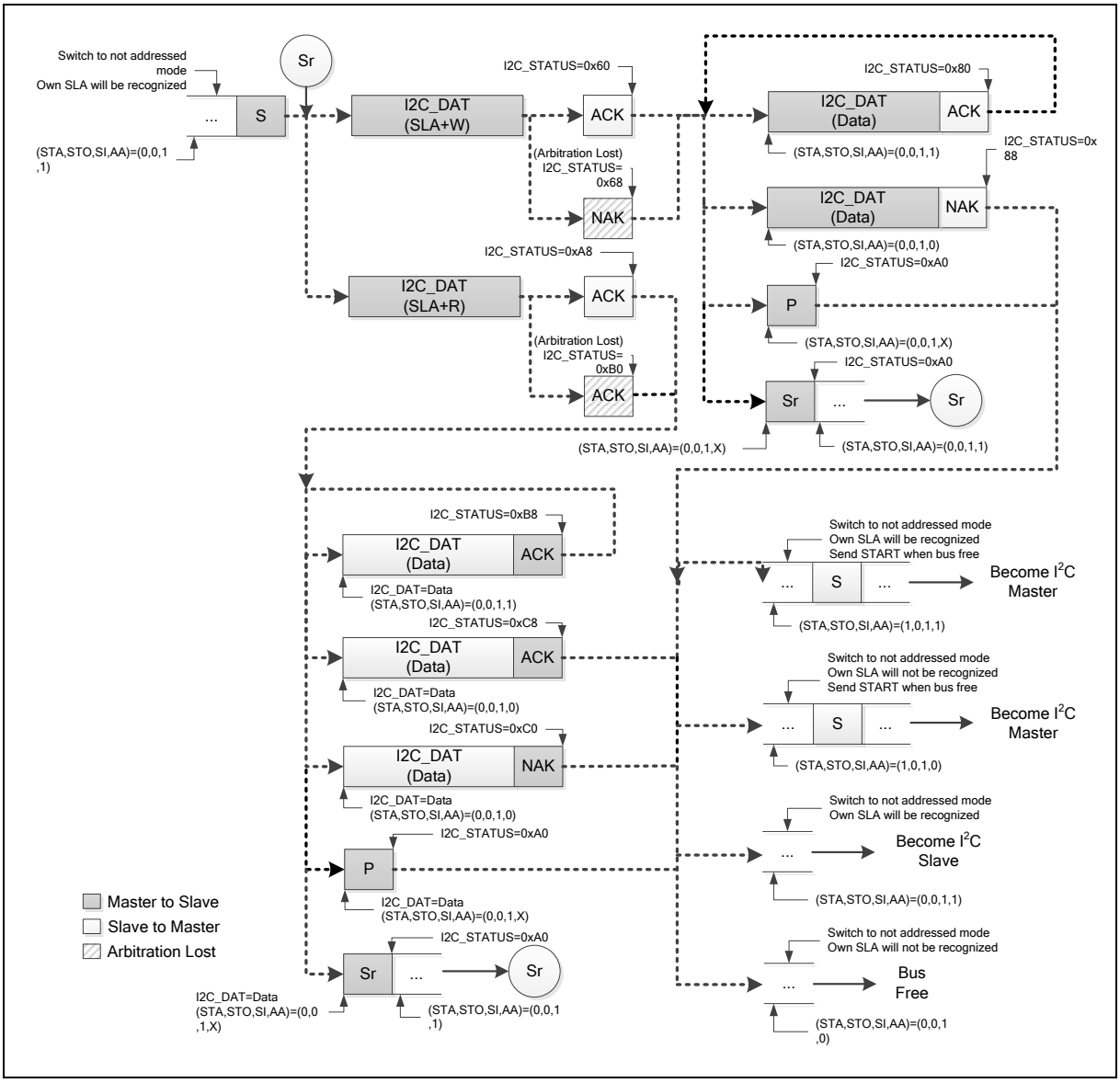


图 6.11-11 从机控制流程

如果I2C在可寻址从机模式接收数据时，收到停止或重复起始信号，状态码是0xA0。当状态码是0xA0时，用户可以遵循如上图状态码是0x88的操作。

如果I2C在可寻址从机模式传输数据时却收到停止或重复起始信号，状态码是0xA0。当状态码是0xA0时，用户可以遵循如上图状态码是0xC8的操作。

从机获得0x88, 0xC8, 0xC0 和0xA0状态后,从机可以切换到无地址模式，自身SLA不会被辨识。如果进入这种状态，从机不再接收主机任何信号或地址。在这种状态，需要复位才能离开这种状态。

6.11.5.3 广播 (GC) 模式

如果 GC位(I2C_ADDRn [0]) 被设, I²C端口硬件将响应广播呼叫地址(0x00).用户可以通过清GC位来禁止广播呼叫功能。当I²C在从机模式时且GC 位被设置，可以接收主机地址(0x00)的广播呼叫，将遵循广播模式状态。

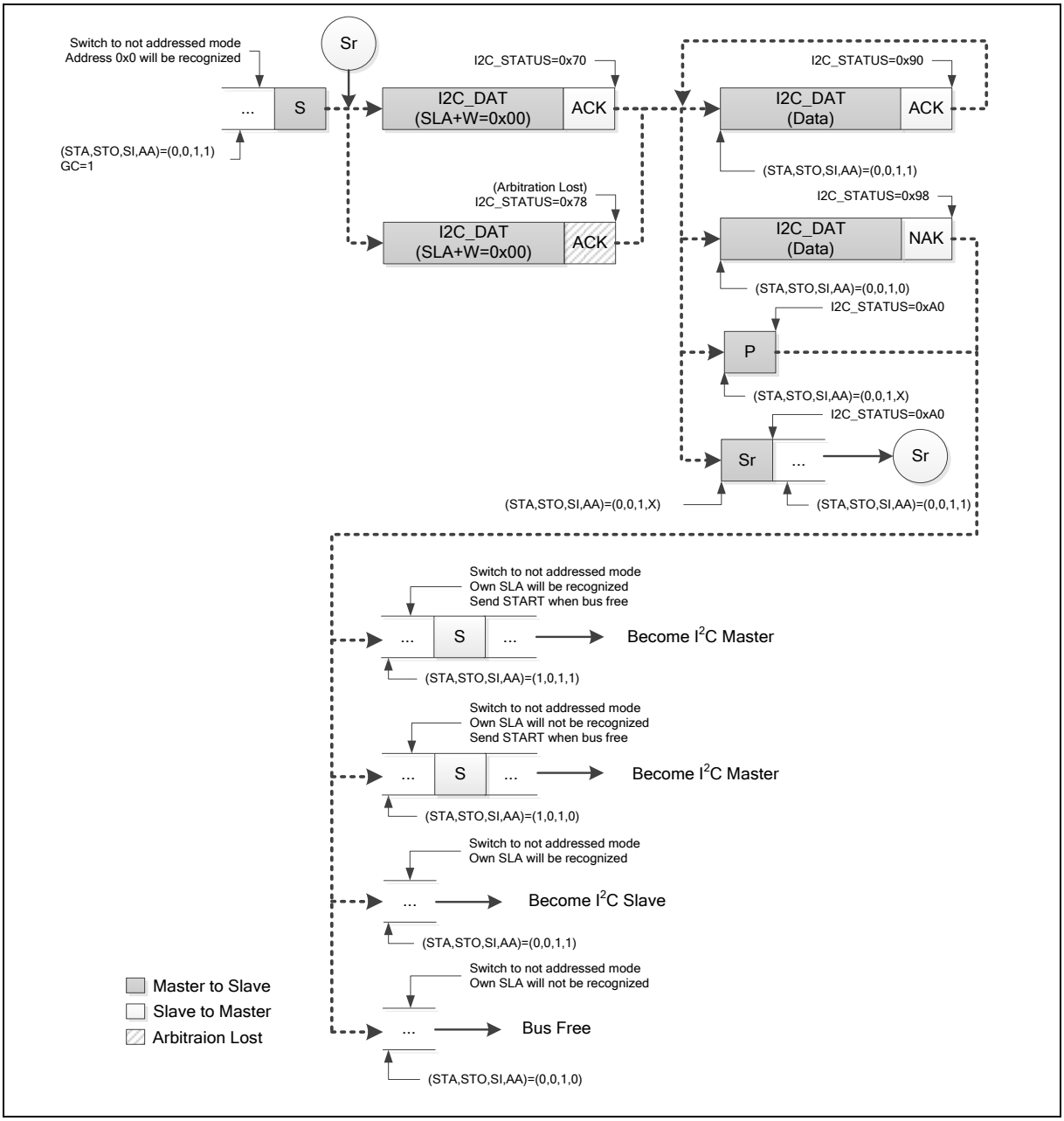


图 6.11-12 GC模式

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

从机获得0x98 和0xA0 状态后，从机会切换到无地址模式并且自身SLA将不被辨识。如果进入这种

状态，从机不再接收主机的任何信号或地址。需要复位才能离开这种状态.

6.11.5.4 多主机模式

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

- 当I2C_STATUS = 0x38代表接收到一个仲裁丢失。仲裁丢失事件可能发生在发送起始位，数据位或者停止位期间。用户可以在总线空闲时设置(STA, STO, SI, AA) = (1, 0, 1, X)来再次发送起始信号或者设置(STA, STO, SI, AA) = (0, 0, 1, X)发送停止信号来返回到无地址从机模式.
- 当I2C_STATUS = 0x00，接收到一个“总线错误”，为了从错误总线恢复到I2C总线，STO应被设置且SI应将被清0，然后对STO清0来释放总线。
  - 设置 (STA, STO, SI, AA) = (0, 1, 1, X) 停止当前传输
  - 设置 (STA, STO, SI, AA) = (0, 0, 1, X) 释放总线

6.11.5.5 EEPROM随机读取例子

通过下面的步骤来配置I²C0相关寄存器，来使用I²C从EEPROM读取数据.

1. 在 “SYS_GPB_MFPL”多功能寄存器设置为SCL (PB.0) 和 SDA (PB.1) 管脚.
2. 使能 I²C APB clock, I2C0CKEN=1在 “CLK_APBCLK[5]” 寄存器.
3. 设置 I2C0RST=1 去复位 I²C 控制器然后设置 I²C 控制器正常操作, I2C0RST=0 在 “SYS_IPRST1[8]” 寄存器.
4. 设置I2CEN (I2C_CTL[6])=1使能 I²C控制器在 “I2C_CTL” 寄存器.
5. 给定 I²C时钟分频寄存器值设I²C 时钟频率在 “I2C_CLKDIV”寄存器.
6. 设置 “NVIC_ISER” 寄存器 去 使能I²C IRQ.
7. 设置 INTEN (I2C_CTL[7])=1 去 使能 I²C中断在 “I2C_CTL” 寄存器.
8. 设 I²C地址寄存器 “I2C_ADDR0~I2C_ADDR3”.

随机读操作是访问EEPROM的其中一种方法。这个方法是允许主机访问EEPROM的任何一个地址，. 图显示对EEPROM随机读取操作.

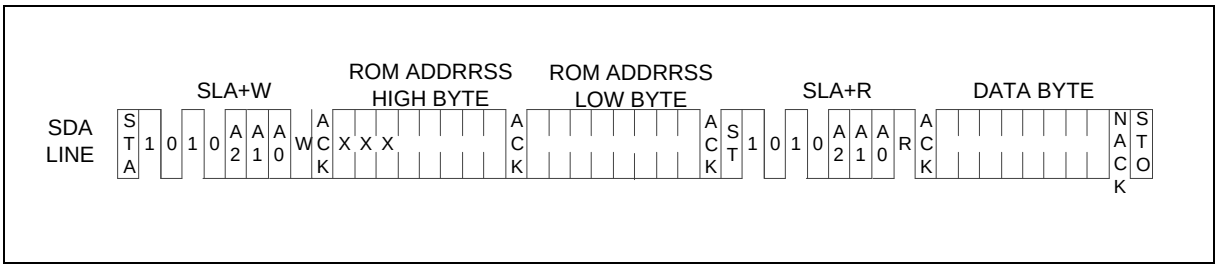


图 6.11-13 EEPROM 随机读

图显示如何用 I²C控制器去实施如何随机读EEPROM协议.

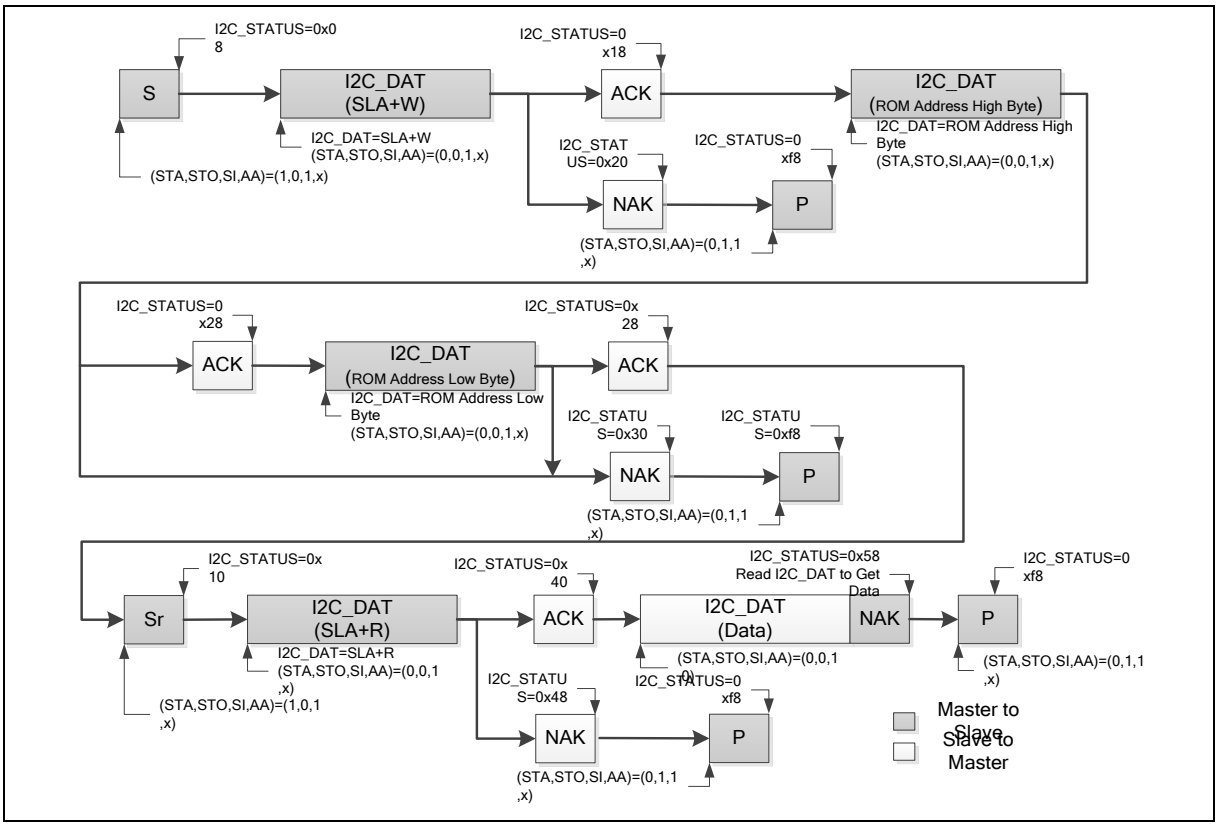


图 6.11-14 随机读EEPROM的协议

I²C控制器作为主机发送起始信号到总线，然后发送SLA+W（从机地址 + 写位）到EERPOM，跟着由两个字节数据地址来设置EEPROM被读的地址。最后，重复起始信号跟着SLA+R被发送来向EEPROM 读取数据。

6.11.6 协议寄存器

CPU的 I²C接口是通过以下的13个特殊功能寄存器: I2C_CTL (控制寄存器), I2C_STATUS (状态 寄存器), I2C_DAT (数据寄存器), I2C_ADDRn (地址寄存器, n=0~3), I2C_ADDRMSKn (地址屏蔽寄存器s, n=0~3), I2C_CLKDIV (时钟频率寄存器) and I2C_TOCTL (超时计数寄存器). 所有位 31~位 8 这些 I²C特殊功能寄存器是保留的. 这些位不能有任何功能，如果读值都是0.

当 I²C 接口被设置 I2CEN (I2C_CTL [6]) 为高使能, 内部状态由I2C_CTL 和 I²C硬件逻辑控制，当一个新状态码产生并存在I2C_STATUS中, I²C 中断标志位SI (I2C_CTL [3])将自动置1 .如果这时使能 INTEN (I2C_CTL [7]), I²C中断将发生. I2C_STATUS[7:3] 空间存贮内部状态码, I2C_STATUS的低3位一直为0，状态码一直保持到被软件清除SI。.

6.11.6.1 地址寄存器 (I2C_ADDR)

I²C端口内建4个从机地址寄存器I2C_ADDRn (n=0~3)。当I2C作为主机时，这四个寄存器的内容不相关。在从机模式下，位字段 (I2C_ADDRn [7:1]) 必须装载芯片自己的从机地址。当I2C_ADDRn地址与接收到的从机地址符合时I²C硬件会作出反应

I2C 端口支持“广播呼叫”功能。当GC位(I2C_ADDRn [0]) 被设置，I2C端口硬件将应答广播呼叫地址(00H)。清GC 位可禁用“广播呼叫”功能

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

I²C总线控制器带有四个地址掩码寄存器I2C_ADDRMSKn (n=0~3) 支持多地址识别。当地址掩码寄存器被置1，意味着收到的相应地址位是"无关的". 如果置0则收到相应地址位应跟地址寄存器完全一样。.

6.11.6.2 数据寄存器 (I2C_DAT)

该寄存器包含一个准备发送或刚接收到的一个字节的串行数据。当不在字节移位处理过程时，CPU可以直接读写访问I2C_DAT[7:0]。当I2C处于一个确定的状态并且串行中断标志(SI) 被置1，I2C_DAT[7:0]中的数据保持稳定。当数据被移出，总线上的数据同时被移入。I2C_DAT [7:0]的内容总是总线上传递的最后一个字节。

应答位由I²C硬件控制，不能被CPU 访问。串行数据在SCL线上串行时钟脉冲的上升沿移入I2C_DAT [7:0] 。当一个字节被移入I2C_DAT [7:0]，则I2C_DAT [7:0] 中的串行数据是可用的，应答位 (ACK 或NACK) 在第9个时钟脉冲由控制逻辑返回。为了在发送数据时监控总线的状态，当发送 I2C_DAT [7:0]到总线时，总线数据将同时被移入I2C_DAT[7:0]。在发送数据过程中，串行数据在 SCL 时钟脉冲的下降沿从I2C_DAT[7:0] 移出，在SCL时钟脉冲的上升沿数据移入I2C_DAT [7:0]。

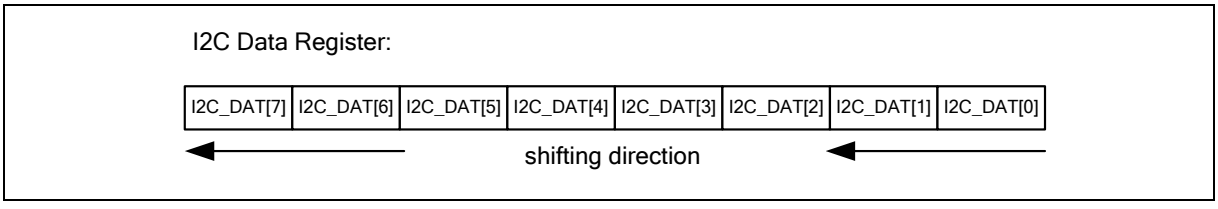


图 6.11-15 I²C数据移动方向

6.11.6.3 控制寄存器 (I2C_CTL)

CPU 能直接读写 I2C_CTL [7:0] 的8位空间。 有2位受硬件影响: 当I²C硬件发生硬件中断请求时SI位设1 , 当停止条件成立或I2CEN = 0时 STO位被清除。

INTEN 中断使能.

I2CEN I²C 功能使能. 当 I2CEN=1 I²C串行功能使能. 多功能管脚必须设置为I²C功能的SDA 和 SCL.

STA I²C 开始控制位. STA 设置1进入主机模式, 当总线空闲 I²C硬件发送开始信号或重复开始信号.

STO I²C 停止控制位.主机模式, 设置 STO发送一个停止信号到总线，I²C硬件将检查总线是不是符合停止条件，符合后硬件自动清除该位。从机模式, 设置 STO将复位 I²C 硬件到不寻址的从机模式。. 这意味着将**不再**从主机接收数据的从机状态。.

SI I²C 中断标志. 当新的 I²C 状态存于 I2C_STATUS 寄存器时, SI 被硬件置位, 如果 INTEN (I2C_CTL [7])是置位状态, I²C中断请求产生. SI 必须被软件写1清除. 所有状态都在 I2C_STATUS 寄存器中.

AA 应答控制位. 当在接收地址或数据之前AA=1,将在SCL收到应答时钟期间回送应答 (SDA低电平)，1.) 从机收到主机地址的应答2.)接收设备收到数据的应答。当在接收地址或数据之前AA=0 ， 在SCL收到应答时钟期间不回送应答 (SDA高电平).

6.11.6.4 状态寄存器 (I2C_STATUS)

I2C_STATUS [7:0] 是一个8-位只读寄存器。I2C_STATUS [7:0]共有26 个可能的状态码。所有状态码由下面表所列出。当I2C_STATUS [7:0] 的值为0xF8时，没有串行中断请求。所有其他的I2C_STATUS [7:0]的值对应I²C 的状态。当进入其中任一状态时，就会产生状态中断请求(SI=1)。在SI 被硬件置位或SI 被软件复位后一个机器周期，有效状态码出现在I2C_STATUS [7:0] 中。

此外，状态0x00表示总线错误，这会出现在起始或停止条件处在不正确的I²C格式帧。总线错误会发生在一个地址字节、一个数据字节或一个应答位的串行传输。从错误总线恢复时STO应被置位，S应被清除，来进入无地址从机模式。然后STO清除释放总线并等待下一个传输。出现总线错误操作期间I²C总线不能识别停止条件。

主机模式		从机模式	
STATUS	**描述**	**STATUS**	**描述**
0x08	开始	0xA0	从机发送重新起始或停止
0x10	主机重新开始	0xA8	从机发送地址ACK
0x18	主机发送地址 ACK	0xB0	从机发送仲裁丢失
0x20	主机发送地址NACK	0xB8	从机发送数据 ACK
0x28	主机发送数据ACK	0xC0	从机发送数据NACK
0x30	主机发送数据 NACK	0xC8	从机发送最后数据ACK
0x38	主机仲裁丢失	0x60	从机接收地址ACK

0x40	主机接收地址 ACK	0x68	从机接收仲裁丢失
0x48	主机接收地址NACK	0x80	从机接收数据ACK
0x50	主机接收数据ACK	0x88	从机接收数据 NACK
0x58	主机接收数据NACK	0x70	广播模式地址ACK
0x00	总线错误	0x78	广播模式仲裁丢失
-	-	0x90	广播模式数据ACK
-	-	0x98	广播模式数据NACK
0xF8	总线释放 主/从模式的 “0xF8”状态，都不会产生中断.		

表 6.11-1 I²C状态码描述表

6.11.6.5 I2C时钟波特率位 (I2C_CLKDIV)

当I²C 在主机模式下，I²C数据的波特率由CLK(I2C_CLKDIV [7:0])寄存器设定。在从机模式下时不需要。在从机模式下，I²C 将自动与主机I²C设备时钟频率同步。

I²C 数据波特率的设定：I²C数据波特率= (系统时钟) / (4x ((I2C_CLKDIV [7:0] +1))。如果系统时钟 = 16 MHz, I2C_CLKDIV [7:0] = 40 (0x28), 那么I²C数据波特率= 16 MHz/ (4x (40 +1)) = 97.5 K位/sec。

6.11.6.6 I²C 溢出计数器 寄存器 (I2C_TOCTL)

系统提供一个14 位超时的计数器来处理当I2C总线锁死时的情况。当超时计数器计数功能使能后，计数器开始计数直至溢出(TOIF (I2C_TOCTL[0])=1) 并向CPU 产生I2C中断或者清除TOCEN (I2C_TOCTL[2])为0停止计数。当超时计数器使能，对SI (I2C_CTL[3])标志置高会使计数器复位，再对SI 清零会重新开始计数。如果I2C总线锁死，会使I2C_STATUS 及SI 标志不再更新，该14-位超时计数器会发生溢出从而产生I2C中断通知CPU。参考下图14-位超时计数器。用户可以写1 清TOIF为0。

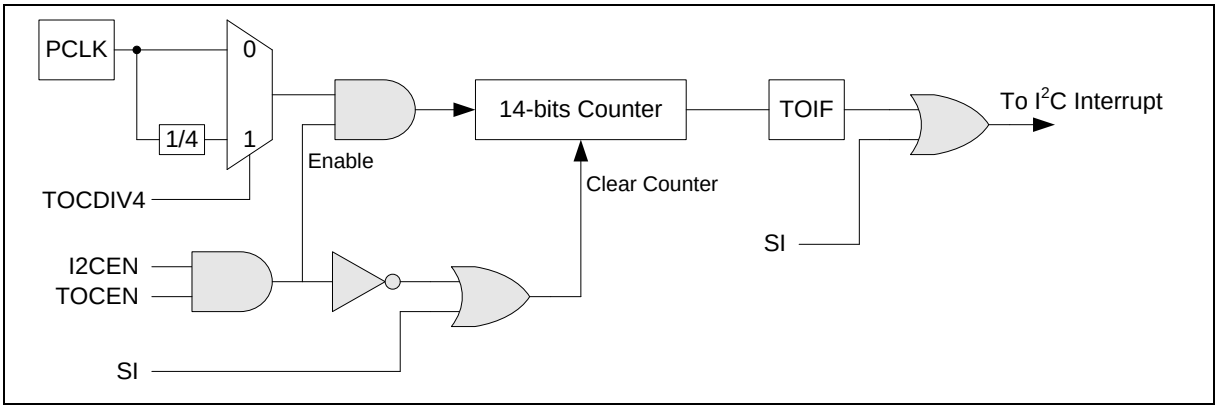


图 6.11-16 I²C 溢出计数器框图

6.11.6.7 I^2C 唤醒控制寄存器 (I2C_WKCTL)

当进入休眠模式， 其它 I^2C主机能通过地址唤醒I^2C设备芯片,用户必须进行休眠模式前进行配置设定 WKUPEN 使能I^2C 唤醒功能。

6.11.6.8 I^2C 唤醒状态寄存器 (I2C_WKSTS)

当系统被其它 I^2C 主机设备唤醒时, WKIF将指示这个事件

WKIF 唤醒中断标志

6.11.7 寄存器 映射

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移地址	R/W	描述	复位值
I2C 基地址: **I2Cx_BA = 0x400E_4000 + x * 0x1000** **x=0,1**				
I2C_CTL	I2Cx_BA+0x00	R/W	I^2C状态寄存器	0x0000_0000
I2C_ADDR0	I2Cx_BA+0x04	R/W	I^2C从机地址寄存器0	0x0000_0000
I2C_DAT	I2Cx_BA+0x08	R/W	I^2C 数据 寄存器	0x0000_0000
I2C_STATUS	I2Cx_BA+0x0C	R	I^2C状态寄存器	0x0000_00F8
I2C_CLKDIV	I2Cx_BA+0x10	R/W	I^2C 时钟分频 寄存器	0x0000_0000
I2C_TOCTL	I2Cx_BA+0x14	R/W	I^2C超时计数控制 寄存器	0x0000_0000
I2C_ADDR1	I2Cx_BA+0x18	R/W	I^2C 从机地址寄存器1	0x0000_0000
I2C_ADDR2	I2Cx_BA+0x1C	R/W	I^2C 从机地址寄存器2	0x0000_0000
I2C_ADDR3	I2Cx_BA+0x20	R/W	I^2C从机地址寄存器3	0x0000_0000
I2C_ADDRMSK0	I2Cx_BA+0x24	R/W	I^2C 从机地址屏蔽寄存器0	0x0000_0000
I2C_ADDRMSK1	I2Cx_BA+0x28	R/W	I^2C 从机地址屏蔽 寄存器1	0x0000_0000
I2C_ADDRMSK2	I2Cx_BA+0x2C	R/W	I^2C 从机地址屏蔽寄存器2	0x0000_0000
I2C_ADDRMSK3	I2Cx_BA+0x30	R/W	I^2C 从机地址屏蔽 寄存器3	0x0000_0000
I2C_WKCTL	I2Cx_BA+0x3C	R/W	I^2C 唤醒控制寄存器	0x0000_0000
I2C_WKSTS	I2Cx_BA+0x40	R/W	I^2C 唤醒状态寄存器	0x0000_0000

6.11.8 寄存器 描述

I²C控制寄存器 (I2C_CTL)

寄存器	偏移地址	R/W	描述	复位值
I2C_CTL	I2Cx_BA+0x00	R/W	I²C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
INTEN	I2CEN	STA	STO	SI	AA	Reserved	

位	描述	
[31:8]	Reserved	保留.
[7]	INTEN	**I²C 中断 使能 控制** 0 = I²C中断关闭. 1 = I²C 中断使能.
[6]	I2CEN	**I²C 控制器 使能 控制** 0 = 关闭. 1 = 使能. 设置使能 I²C 串行功能控制器。当I2CEN =1，I²C 串行功能使能。SDA和SCL对应的多功能管脚功能必须先设置为 I²C 功能。
[5]	STA	**I²C 开始控制** 设置 STA 为 1，进入主机模式，如果总线处于空闲状态，I²C 硬件会送出 START 或 重复 START 条件。
[4]	STO	**I²C 停止控制** 在主机模式，设置 STO 来传送一个 STOP 条件到总线，然后 I²C 硬件将会检查总线状况，如果检测到一个 STOP 状况，这个标志会被硬件自动清除。从机模式, 设置 STO将复位 I2C 硬件到不寻址的从机模式。. 这意味着将不再从主机接收数据的从机状态
[3]	SI	**I²C 中断标志** 当一个新的 I²C 状态出现在寄存器 I2C_STATUS 时，SI 标志由硬件置位，并且如果INTEN (I2C_CTL [7])位被置位，则产生 I²C 中断请求。SI 必须由软件通过向该位写 '1' 清零。 如果在从机读模式且ACKMEN置位，SI标志在第8个时钟周期置位，此时用户确认应答位，在第9个时钟周期时，用户可以从数据缓冲器中读取数据。
[2]	AA	**应答控制位**

		当 AA=1 先于地址或数据接收，在SCL线上的应答时钟脉冲期间将返回一个应答(SDA上为低电平)，有两种情况：1.) 从机正在应答主机发送的地址。2.) 接收设备正在应答发送设备发送的数据。当 AA = 0 先于地址或数据接收，则在SCL线上的应答时钟脉冲期间将返回一个非应答（SDA上为高电平）。
[1:0]	**Reserved**	保留.

I²C 数据寄存器 (I2C_DAT)

寄存器	偏移地址	R/W	描述	复位值
I2C_DAT	I2Cx_BA+0x08	R/W	I²C 数据 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
DAT							

位	描述	
[31:8]	Reserved	保留.
[7:0]	DAT	I²C 数据 位 Bit [7:0] 区域 8 位存放 I²C 串行端口8位传输/接收数据

I²C 状态 寄存器 (I2C_STATUS)

寄存器	偏移地址	R/W	描述	复位值
I2C_STATUS	I2Cx_BA+0x0C	R	I²C 状态 寄存器	0x0000_00F8

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
STATUS							

位	描述	
[31:8]	Reserved	保留.
[7:0]	STATUS	**I²C 状态 位** I²C的状态: 低3位一直为0. 有效的5 位包括状态码，. 参考表6.11.6.4 详细 描述.

I²C 时钟分频寄存器 (I2C_CLKDIV)

寄存器	偏移地址	R/W	描述	复位值
I2C_CLKDIV	I2Cx_BA+0x10	R/W	I²C 时钟分频寄存器	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**
DIVIDER							

位	描述	
[31:8]	**Reserved**	保留.
[7:0]	**DIVIDER**	**I²C 时钟分频位** I²C时钟波特率值 = (PCLK) / (4x (DIVIDER+1)). 最小值 DIVIDER是 4.

I²C 超时计数器寄存器 (I2C_TOCTL)

寄存器	偏移地址	R/W	描述	复位值
I2C_TOCTL	I2Cx_BA+0x14	R/W	I²C 超时计数器寄存器	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					TOCEN	TOCDIV4	TOIF

位	描述	
[31:3]	Reserved	保留.
[2]	TOCEN	**超时计数器使能 控制** 0 = 关闭. 1 = 使能. **注:** 当使能该位，则14-位溢出定时器将会在SI (I2C_CTL[3])清零后开始计数。设置 SI位为高将会复位计数器，在 SI位清零之后，计数器会重新开始计数。
[1]	TOCDIV4	**溢出定时器输入时钟除以 4** 0 = 关闭. 1 = 使能. **注:** 当 使能, 溢出超时时间放大4倍.
[0]	TOIF	**溢出标志** 当超时发生时，该位由 H/W 置位，如果此时 I²C 的中断使能位(INTEN (I2C_CTL[7]))置为1，则可引发 CPU 的中断。 **注:** 该位写1清除。

I²C从机地址寄存器 0~3 (I2C_ADDRx)

寄存器	偏移地址	R/W	描述	复位值
I2C_ADDR0	I2Cx_BA+0x04	R/W	I²C 从机地址寄存器0	0x0000_0000
I2C_ADDR1	I2Cx_BA+0x18	R/W	I²C 从机地址寄存器1	0x0000_0000
I2C_ADDR2	I2Cx_BA+0x1C	R/W	I²C从机地址 寄存器2	0x0000_0000
I2C_ADDR3	I2Cx_BA+0x20	R/W	I²C 从机地址寄存器3	0x0000_0000

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

位	描述	
[31:8]	Reserved	保留.
[7:1]	ADDR	**I²C 地址位** 主机模式下，该寄存器内容没有意义。从机模式下，高七位作为芯片本身的地址。如果地址符合，I²C 硬件将会自动应答
[0]	GC	**广播功能** 0 = 广播呼叫功能禁用 1 = 广播呼叫功能使能

I²C 从机地址屏蔽 寄存器 0~3 (I2C_ADDRMSKx)

寄存器	偏移地址	R/W	描述	复位值
I2C_ADDRMSK0	I2Cx_BA+0x24	R/W	I²C 从机地址屏蔽寄存器0	0x0000_0000
I2C_ADDRMSK1	I2Cx_BA+0x28	R/W	I²C从机地址屏蔽寄存器1	0x0000_0000
I2C_ADDRMSK2	I2Cx_BA+0x2C	R/W	I²C从机地址屏蔽寄存器2	0x0000_0000
I2C_ADDRMSK3	I2Cx_BA+0x30	R/W	I²C从机地址屏蔽寄存器3	0x0000_0000

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

位	描述	
[31:8]	**Reserved**	保留.
[7:1]	**ADDRMSK**	**I²C地址屏蔽位** 0 =掩码禁用（接收到的相应地址必须完全符合地址寄存器） 1 =掩码使能（接收到的相应地址位不予辨识） I²C 总线控制器有4个地址掩码寄存器，支持多地址识别。当地址掩码寄存器的某位被置 '1'，表示接收到的地址的相应位可忽略。如果该位被置 '0'，则表示接收到的地址的相应位必须和地址寄存器中的完全一致。
[0]	**Reserved**	保留.

I²C唤醒控制寄存器 (I2C_WKCTL)

寄存器	偏移地址	R/W	描述	复位值
I2C_WKCTL	I2Cx_BA+0x3C	R/W	I²C唤醒控制寄存器	0x0000_0000

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

位	描述	
[31:1]	Reserved	保留.
[0]	WKEN	I²C唤醒使能 控制 0 = I²C唤醒功能关闭. 1 = I²C唤醒功能使能.

I²C 唤醒状态寄存器 (I2C_WKSTS)

寄存器	偏移地址	R/W	描述	复位值
I2C_WKSTS	I2Cx_BA+0x40	R/W	I²C 唤醒状态寄存器	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							WKIF

位	描述	
[31:1]	Reserved	保留.
[0]	WKIF	**I²C 唤醒标志** 0 =没有唤醒事件发生. 1 = 从掉电状态唤醒. **注:**软件写1清除

I²C 唤醒状态寄存器 (I2C_WKSTS)

寄存器	偏移地址	R/W	描述	复位值
I2C_WKSTS	I2Cx_BA+0x40	R/W	I²C唤醒状态寄存器	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							WKIF

位	描述	
[31:1]	Reserved	保留.
[0]	WKIF	**I²C 唤醒标志** 0 = 没有唤醒发生. 1 = 从掉电状态唤醒. **注:** 软件写1清除该位

6.12 串行外围设备接口(SPI)

6.12.1 概述

串行外围设备接口(SPI)是一个工作于全双工模式的同步串行数据通讯协议。设备可工作在主/从模式，利用4线双向接口相互通讯。NUC505系列包含2组SPI控制器，当从一个外围设备接收数据时，SPI执行串-并的转换，而在数据向外围设备发送时执行并-串的转换。每组SPI控制器可以配置为主设备或从设备。

6.12.2 特性

- 支持主机和从机工作模式
- 一个事务传输的数据长度可配置为8到32位
- 提供独立的8级深度发送和接收FIFO缓存
- 支持MSB或LSB优先传输
- 支持字节重排序功能
- 支持三线，没有从机片选信号的双向接口
- 多达2组 SPI 控制器

6.12.3 框图

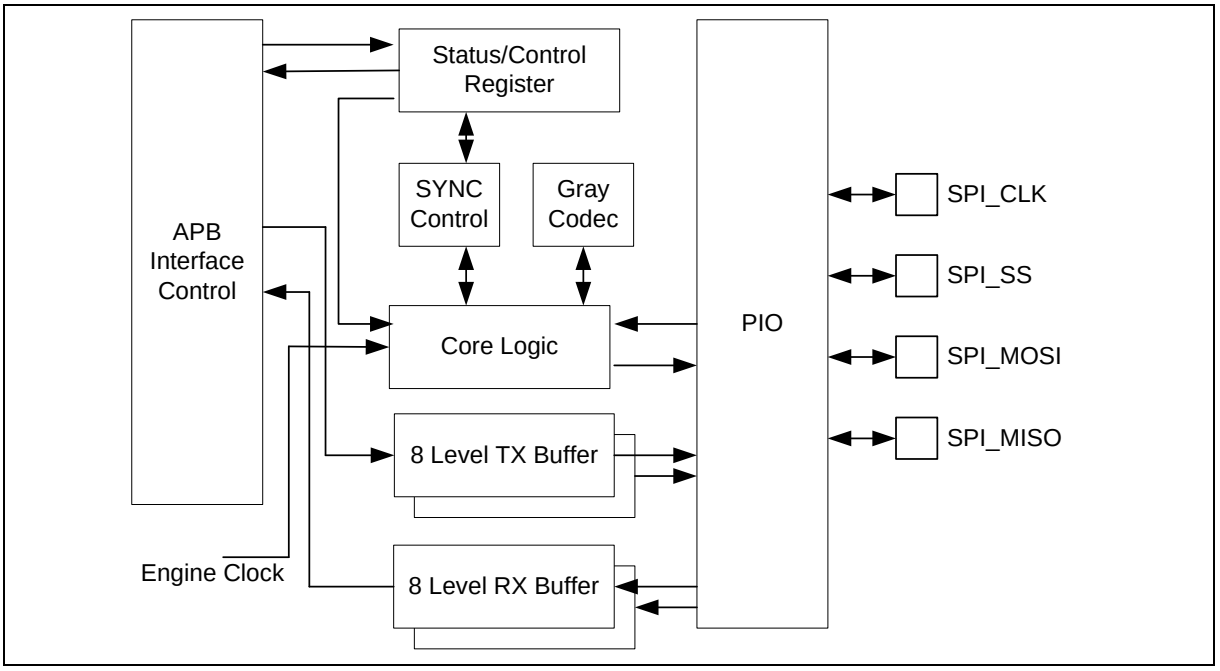


图 6.12-1 SPI 模块框图

6.12.4 功能描述

SPI外设时钟和SPI总线时钟

SPI控制器需要外设时钟来驱动SPI逻辑单元来执行数据传输。SPI 外设时钟速率决定于时钟分频器的设置。SPI0SEL (CLK_CLKDIV2[28]) 和 SPI1SEL (CLK_CLKDIV2[29]) 决定SPI0和SPI1的外设时钟源。时钟源可以设置为HXT或 PLL输出时钟。寄存器DIVIDER(SPI_CLKDIV[7:0])的设定值决定时钟速率计算的分频值。

在主机模式下，SPI总线的时钟频率等于外设的时钟速率。在通常情况下，SPI总线时钟表示为SPI时钟。在从机模式下，SPI总线时钟由片外主机设备提供。从机设备的SPI外设时钟速率必须大于所连接的主机设备的SPI总线时钟速率。不论工作于主机还是从机模式，SPI外设的时钟频率不能快于系统时钟速率。（如果外设的时钟源不同于系统时钟，SPI外设时钟不管是主机模式还是从机模式都必须不能超过系统时钟频率。）

主机/从机模式

SPI控制器可通过设置SLAVE (SPI_CTL[18])配置成主机或从机模式，来与片外的SPI从机或者主机通信。主机模式与从机模式的应用框图如下：

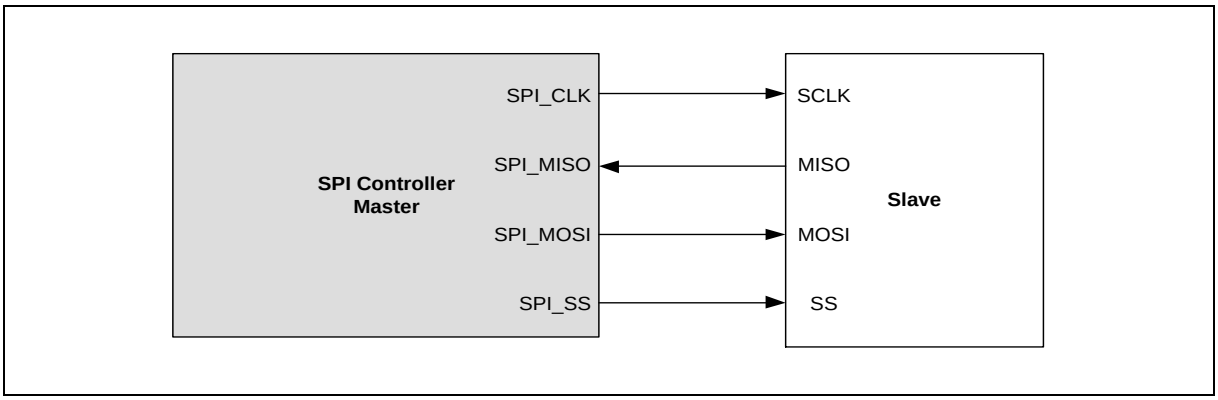


图 6.12-2 SPI 主机模式应用框图

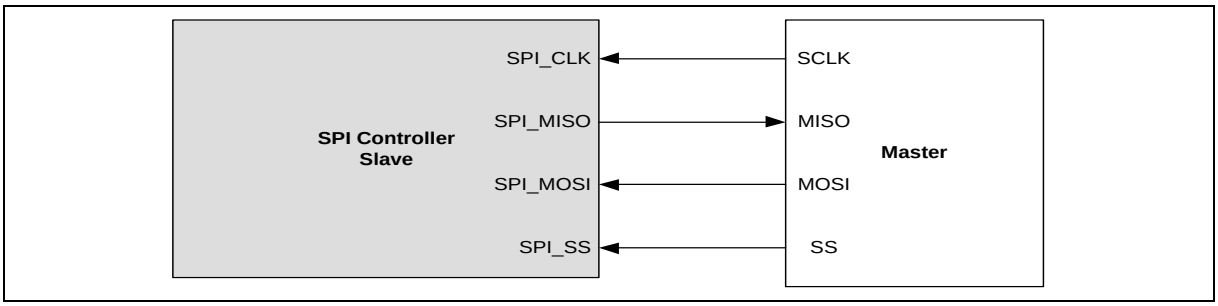


图 6.12-3 SPI从机模式应用框图

从机片选

在主机模式下，该SPI控制器能通过从机片选输出脚SPIn_SS来驱动片外从机设备。在从机模式，片外

主机设备通过SPIn_SS输入端口驱动从机片选信号到SPI控制器。在主机和从机模式下，从机选择信号的有效状态可以通过编程寄存器SSACTPOL (SPI_SSCTL[2])来设定为低有效或高有效。从机片选条件取决于所连设备的类型。

自动从机选择

在主机模式下，如果AUTOSS (SPI_SSCTL[3])设置为1，从机选择信号将会自动产生，并根据SS (SPI_SSCTL[0])是否使能，将从机选择信号输出到SPIn_SS管脚上。当SPI把数据写入FIFO时，启动数据传输，从机选择信号将由SPI控制器设置为有效状态。当SPI总线空闲时，从机选择信号将变为无效状态。如果AUTOSS (SPI_SSCTL[3])位被清除，从机选择输出信号可以通过SPI_SSCTL[0]的相关位手动地置有效或无效。从机选择输出信号的有效状态由SSACTPOL (SPI_SSCTL[2])配置。

在主机模式下，如果SUSPITV[3:0]的值小于3且AUTOSS被设置为1，在两个连续的事务之间，从机选择信号将保持有效。

在从机模式，识别从机选择信号的无效状态，在两个连续的事务之间，从机选择信号的无效周期必须大于或等于3个外设时钟。

时钟极性

CLKPOL (SPI_CTL[3])定义了SPI时钟的空闲状态，如果CLKPOL (SPI_CTL[3]) = 1，SPI输出时钟空闲状态是高电平，如果CLKPOL (SPI_CTL[3]) = 0，空闲是低电平。

发送/接收位宽度

一次传输的位宽度定义在DWIDTH (SPI_CTL[12:8])中，一次发送和接收传输宽度最大可以到32位。

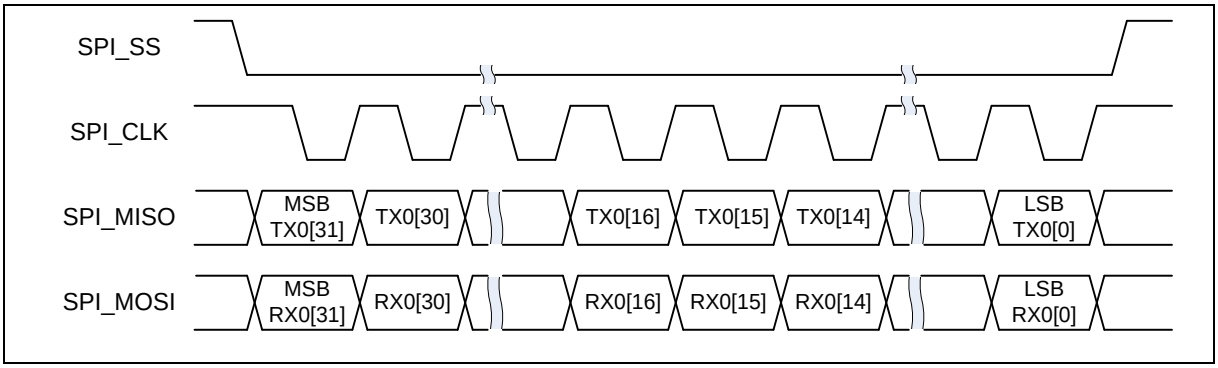


图 6.12-4 一次传输中的位

LSB/MSB 优先

LSB (SPI_CTL[13])定义一个事务中位传输顺序。 如果设定LSB (SPI_CTL[13])为1，传输顺序为LSB优先。位0首先将被传输。如果清除LSB (SPI_CTL[13])为0，传输顺序是MSB优先。

发送边沿

TXNEG (SPI_CTL[2])定义了数据是在SPI时钟的下降沿还是上升沿传输出去。

接收边沿

RXNEG (SPI_CTL[1])定义了数据在SPI时钟的下降沿还是上升沿接收。

注: TXNEG (SPI_CTL[2])和RXNEG (SPI_CTL[1])的设置是相互排斥的，换话说就是不要在相同的时钟沿发送和接收数据。

休眠间隔

在主机模式下，SUSPITV (SPI_CTL[7:4])提供在两个连续事务之间，可配置为0.5～15.5个SPI时钟周期的休眠间隔。休眠间隔是指从前一个事务的最后一个时钟沿到下一个事务的第一个时钟沿的时间间隔。SUSPITV的复位值是0x3(3.5 SPI总线时钟周期)。

字节重排序

当传输设置为 MSB 优先 (LSB = 0) 且使能REORDER (SPI_CTL[19])，在32-位模式(DWIDTH = 0)时，存储在 TX 缓存与 RX 缓存的数据将重新按 [字节0, 字节1, 字节2, 字节3] 的顺序重排列，数据将以 字节0, 字节1, 字节2, 和 字节3 的顺序进行发送/接收。如果DWIDTH设为 24-位传输模式，存储在 TX 缓存与 RX 缓存的数据将重新按 [unknown 字节, 字节0, 字节1, 字节2] 的顺序重排列。SPI 控制器将按照 字节0, 字节1, 字节2 的顺序发送/接收数据，每个字节 MSB 优先发送/接收。16-位模式的规则与上面相同。字节重排序功能只在DWIDTH为16, 24, and 32 位时适用。

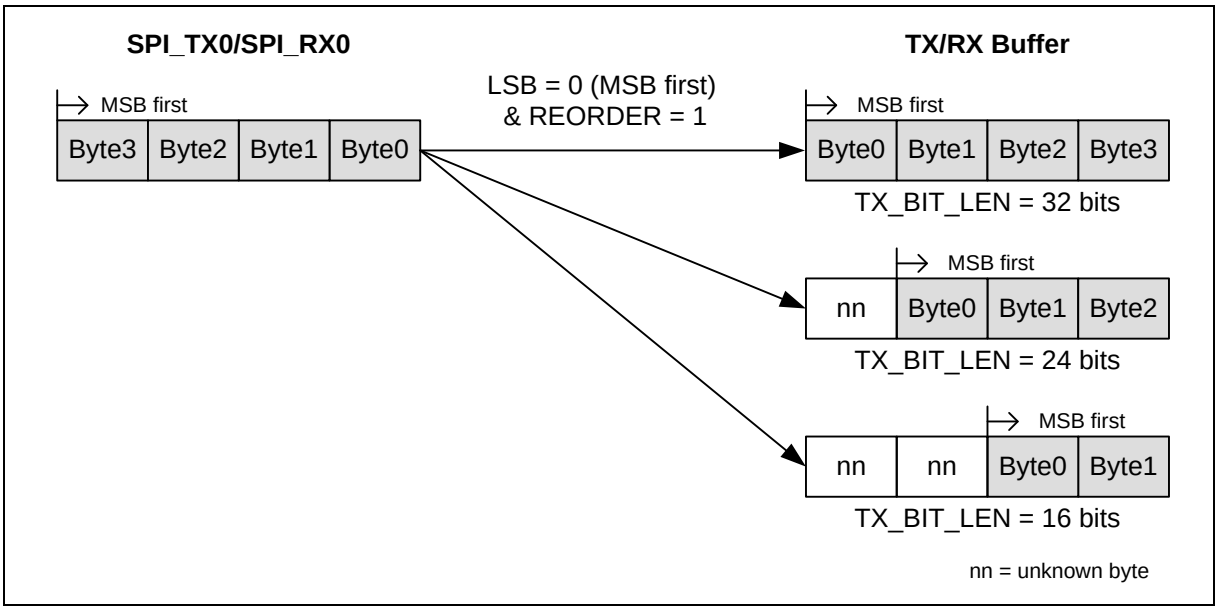


图 6.12-5 字节重排序功能

字节休眠

主机模式下，如果REORDER (SPI_CTL[19]) 被设为 1，硬件将会在两个连续传输字节之间插入一个0.5 ~ 15.5个SPI时钟周期的休眠间隔。休眠间隔时间由SUSPITV (SPI_CTL[7:4])设置。

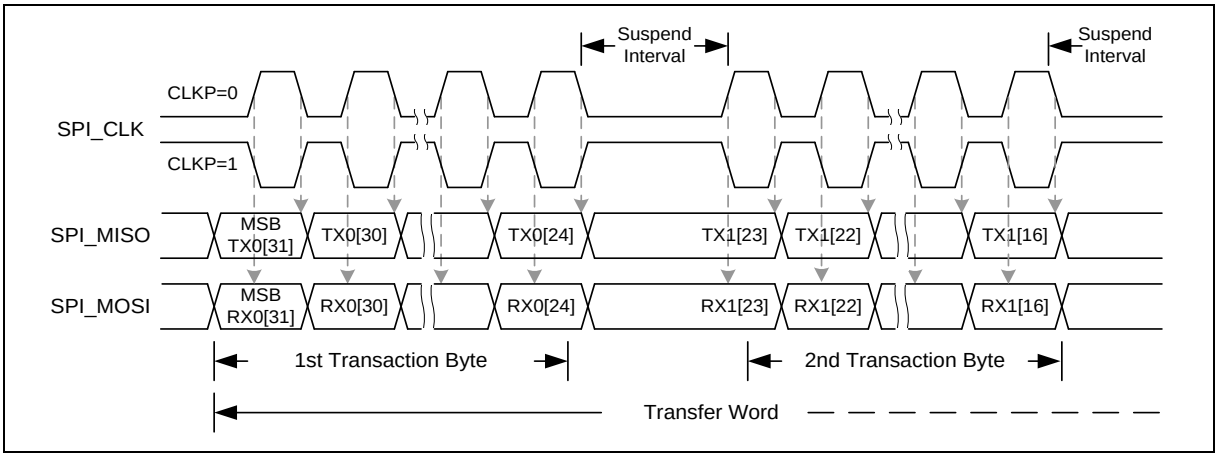


图 6.12-6 字节休眠时序波形

3线模式

当SLV3WIRE (SPI_SSCTL[4])由软件置1，使能从机3线模式时，在从机模式下SPI控制器可以工作在没有从机选择信号。SLV3WIRE (SPI_SSCTL[4])仅在从机模式有效。在与一个主机通讯时，仅需要三个管脚，SPI0_CLK, SPI0_MISO, 和 SPI0_MOSI。SPI0_SS脚可以配置成一个GPIO。当SLV3WIRE (SPI_SSCTL[4])设为1时，在SPIEN (SPI_CTL[0]) 置为1后SPI从机将开始准备发送和接收数据。

8级 FIFO 缓存

SPI控制器配备了8个32-位宽的发送和接收FIFO缓存。

发送FIFO是一个8级深度，32位宽度，先进先出的寄存器缓存。可以通过写SPI_TX寄存器事先将8个数据写入发送FIFO缓存。存放在发送FIFO缓存的数据通过发送控制逻辑进行读取和发送。如果8级发送FIFO缓存满了，TXFULL (SPI_STATUS[17])会被置1。当SPI传输逻辑单元抽出发送FIFO缓存的最后一个数据，发送FIFO缓存就为空了，TXEMPTY (SPI_STATUS[16])位被置1。注TXEMPTY (SPI_STATUS[16])标志被置1时，最后一笔传输还在进行。在主机模式，软件应该检查BUSY (SPI_STATUS[0]) 位和TXEMPTY (SPI_STATUS[16])的状态以确认SPI是否已经空闲。

接收FIFO也是一个8级深度，32位宽度，先进先出的寄存器缓存。接收控制逻辑存储SPI接收到的数据到接收FIFO缓存。接收FIFO缓存的数据可以通过软件读SPI_RX寄存器读出来。有FIFO相关的状态位，像RXEMPTY (SPI_STATUS[8]) 和RXFULL (SPI_STATUS[9])，来表明当前FIFO缓存的状态。

发送和接收的阀值可以通过设置TXTH (SPI_FIFOCTL[30:28]) 和RXTH (SPI_FIFOCTL[26:24])来设定。当存储在发送FIFO缓存的有效数据计数小于或等于TXTH (SPI_FIFOCTL[30:28])设定时，TXTHIF (SPI_STATUS[18])位会被置1。当存储在接收FIFO缓存的有效数据计数大于RXTH (SPI_FIFOCTL[26:24])设定，RXTHIF (SPI_STATUS[10])位会被置1。

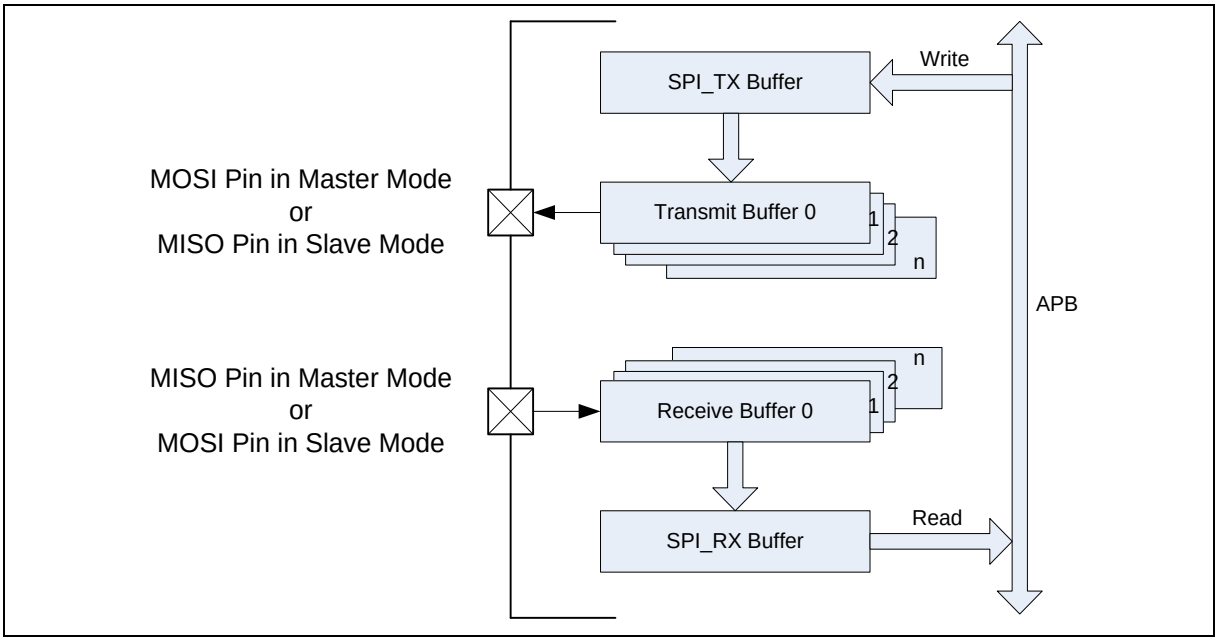


图 6.12-7 FIFO 模块框图

在主机模式，当第一个数据写入SPI_TX寄存器时，TXEMPTY(SPI_STATUS[16]) 标志将会被清除为0。只要发送FIFO不是空，传输会立即开始。用户可以立即写下一个数据到SPI_TX寄存器。SPI 控制器将会在两个连续的事务之间插入一个休眠间隔，休眠间隔的长度由SUSPITV (SPI_CTL[7:4]) 的设定值决定。只要TXFULL (SPI_STATUS[17])为0，用户可以不断的写入数据到SPI_TX寄存器。

如果要发送的数据更新及时，接下来的事务将会被自动触发。如果在所有数据传输完成之后， SPI_TX寄存器没有被更新，则传输停止。

在主机模式接收操作中，串行数据从SPIn_MISO0/1管脚接收并被存储在接收 FIFO 缓存。当接收FIFO缓存有未读数据时，RXEMPTY (SPI_STATUS[8])将清0。只要是RXEMPTY (SPI_STATUS[8])为0，用户可以通过SPI_RX寄存器来读取接收的数据。如果接收FIFO缓存包含8个未读数据，RXFULL (SPI_STATUS[9])将设置为1，SPI控制器会停止接收数据，直到SPI_RX寄存器被软件读取。

在从机模式发送操作中，当软件写数据到SPI_TX 寄存器时，数据将会被加载到发送 FIFO 缓存，而且TXEMPTY (SPI_STATUS[16]) 标志也将被设置为 0。当从设备从主机接收到时钟信号时，发送操作将会开始。只要TXFULL (SPI_STATUS[17]) 标志为 0，用户就可以写数据到SPI_TX 寄存器。在所有数据都被 SPI 发送逻辑单元发送出去后，而且用户没有再更新SPI_TX寄存器，TXEMPTY (SPI_STATUS[16]) 标志将会被设置为 1。

当从机片选信号有效时，如果没有任何数据写入SPI_TX寄存器，且串行时钟输入控制器，发生下溢标志，TXUFIF (SPI_STATUS[19])将被设置为1。在之前的条件下，当SS进入无效状态且发送下溢发生。从机模式错误1，SLVURIF (SPI_STATUS[7])会置1 。

在从机接收操作中，串行数据从SPI_MOSI0/1管脚被接收，并被存储到 SPI_RX0 寄存器。接收机制与主机模式接收操作类似。 如果接收FIFO包含8个未读数据，RXFULL(SPI_STATUS[9])标志会被置1，并且RXOVIF (SPI_STATUS[11])会被置1，如果有更多来自SPI0MOSI的串行数据被接收，接下来的数据会丢失。当从机片选进入无效状态，如果接收位计数与DWIDTH (SPI_CTL[12:8])不匹配，从机模式错误0，SLVBEIF (SPI_STATUS[6])会置1。

当从机选择信号有效，并且SLVTOCNT (SPI_SSCTL[31:16])的值不是0， 在串行时钟输入之后，SPI控

制器逻辑中的从机超时计数器会开始计数。在一次传输完成或SLVTOCNT (SPI_SSCTL[31:16])被置为0之后，该计数器会被清0。在一次传输完成之前，如果超时计数器的值大于或等于SLVTOCNT (SPI_SSCTL[31:16])的值，从机超时事件发生，并且SLVTOIF (SPI_STATUS[5])被置1 。

接收超时功能是嵌入该控制器的。当接收FIFO不为空且在主机模式接收FIFO超过64个SPI时钟周期或在从机模式超过576个SPI外设时钟没有读操作，接收超时发生，并且SLVTOIF (SPI_STATUS[5])被置1。当接收FIFO被用户读取，超时状态会自动清除。

SPI 单元传输中断

当SPI控制器完成一个单元传输，单元传输中断标志UNITIF (SPI_STATUS[1])将会被设为1。如果单元传输中断使能位UNITIEN (SPI_CTL[17]) 被置位，则单元传输中断事件将给CPU产生中断请求。单位传输中断标志位只能写 1 清零。

SPI 从机选择中断

在从机模式，当SPIEN (SPI_CTL[0])和SLAVE (SPI_CTL[18])都设置为1，且从机片选信号进入有效或无效状态时，从机片选有效或无效中断标志SSACTIF (SPI_STATUS[2])和SSINAIF (SPI_STATUS[3])将设置为1。如果SSINAIEN (SPI_SSCTL[13])或SSACTIEN (SPI_SSCTL[12])设置为1，SPI控制器将发生一个中断。

从机超时中断

在从机模式，从机超时功能为，有串行时钟输入但是超过了SLVTOCNT (SPI_SSCTL[31:16])中定义的从机外设时钟周期个数，传输事务还没有完成。

当从机片选信号有效时，而且SLVTOCNT (SPI_SSCTL[31:16])的值不为0，在串行时钟输入后，SPI控制器逻辑的从机超时计数器开始计数。在一次事务完成后或者SLVTOCNT (SPI_SSCTL[31:16])设置为0，该计数器将被清除。如果在一次事务传输完前，超时计数器的值大于或等于SLVTOCNT (SPI_SSCTL[31:16])的值，则从机超时事件发生，同时SLVTOIF (SPI_STATUS[5])将设置为1。如果SLVTOIEN (SPI_SSCTL[5])设置为1，SPI控制器将发生一个中断。

从机错误 0 中断

在从机模式，当从机选择线变成无效状态，如果发送/接收位计数与DWIDTH (SPI_CTL[12:8])不匹配，从机模式错误0，SLVBEIF (SPI_STATUS[6])位被置1 。如果SLVBEIEN (SPI_SSCTL[8])置1 ，SPI控制器会产生中断。

1.在从机发送模式，如果有位长度发送错误（位计数不匹配），用户应该置位TXRST (SPI_FIFOCTL[1]) 位并再次写发送数据重新开始下一次传输。

2.如果从机片选信号激活，但是没有任何串行时钟输入，当从机片选信号进入无效状态时，SLVBEIF (SPI_STATUS[6])也将设置为1。

从机下溢和从机错误1中断

在SPI从机模式，如果没有任何数据写入SPI_TX寄存器，当从机选择信号激活时，并且串行时钟输入控制器，TXUFIF (SPI_STATUS[19])将设置为1。

在之前的条件下，当SS变成无效状态和发送下溢发生，从机模式错误1，SLVURIF (SPI_STATUS[7])会被置1 ， 如果SLVURIEN (SPI_SSCTL[9])有设置为1，SPI控制器将发生中断。

在从机3-线模式，从机片选信号被认为是一直有效的，用户需要去查看TXUFIF (SPI_STATUS[19])位来确定是否有发生发送下溢事件。

接收溢出中断

在从机模式，如果接收FIFO缓存已有8个未读数据， RXFULL (SPI_STATUS[9])标志将会被设置为1，并且RXOVIF (SPI_STATUS[11])会被置1。如果从SPI总线上接收到更多串行数据，RXOVIF (SPI_STATUS[11])会被置1，多余的数据将会丢失。如果RXOVIEN (SPI_FIFOCTL[5])有设置为1，SPI控制器将发生中断。

接收 FIFO 超时中断

在FIFO模式，有一个超时功能来通知用户。如果在FIFO里有一个接收到的数据，在主机模式下用户超过64个SPI外设时钟周期没有去读取，或者从机模式下超过576个SPI外设时钟周期没有去读取，如果接收超时中断使能位RXTOIEN (SPI_FIFOCTL[4])有设置为1，则会向系统发出一个接收超时中断。

发送 FIFO 中断

在FIFO模式，如果发送FIFO缓存的有效数据计数少于或等于TXTH (SPI_FIFOCTL[30:28])的设定值，发送FIFO中断标志会被置1。如果发送FIFO中断位TXTHIEN (SPI_FIFOCTL[3])有设置为1，则SPI控制器会向系统产生一个发送FIFO中断。

接收 FIFO 中断

在 FIFO 模式，如果接收FIFO缓存的有效数据计数大于RXTH (SPI_FIFOCTL[26:24])的设定值，接收FIFO中断标志会被设置为1。如果接收FIFO中断位RXTHIEN (SPI_FIFOCTL[2])有设置为1，SPI控制器将会向系统产生一个接收FIFO中断。

6.12.5 时序图

从机选择信号的有效状态可以由SSACTPOL (SPI_SSCTL[2])的设置定义。串行时钟的空闲状态可以通过CLKPOL (SPI_CTL[3]) 配置为高电平或低电平。传输字段长度在DWIDTH (SPI_CTL[12:8])中定义，发送/接收数据是以 MSB 还是 LSB 优先由 LSB 位(SPI_CTL[13]) 定义。用户也可以通过设置TXNEG/RXNEG (SPI_CTL[2:1])来选择发送/接收数据时串行时钟的边沿。四种SPI 主机/从机操作时序图以及相关设置如下图。

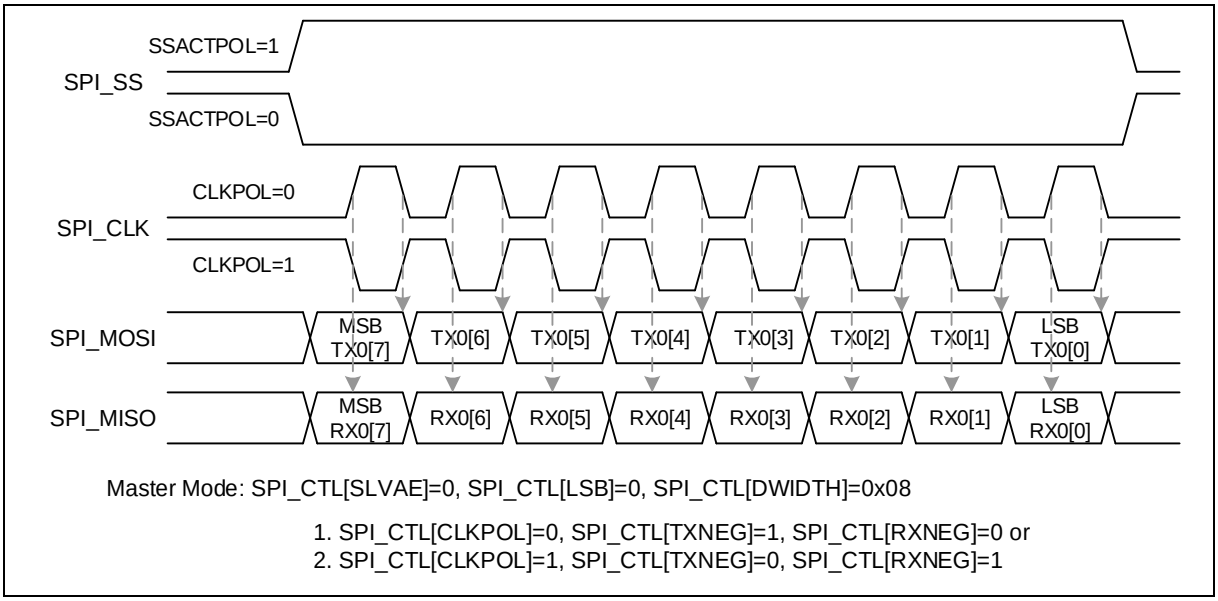


图 6.12-8 主机模式下的SPI 时序

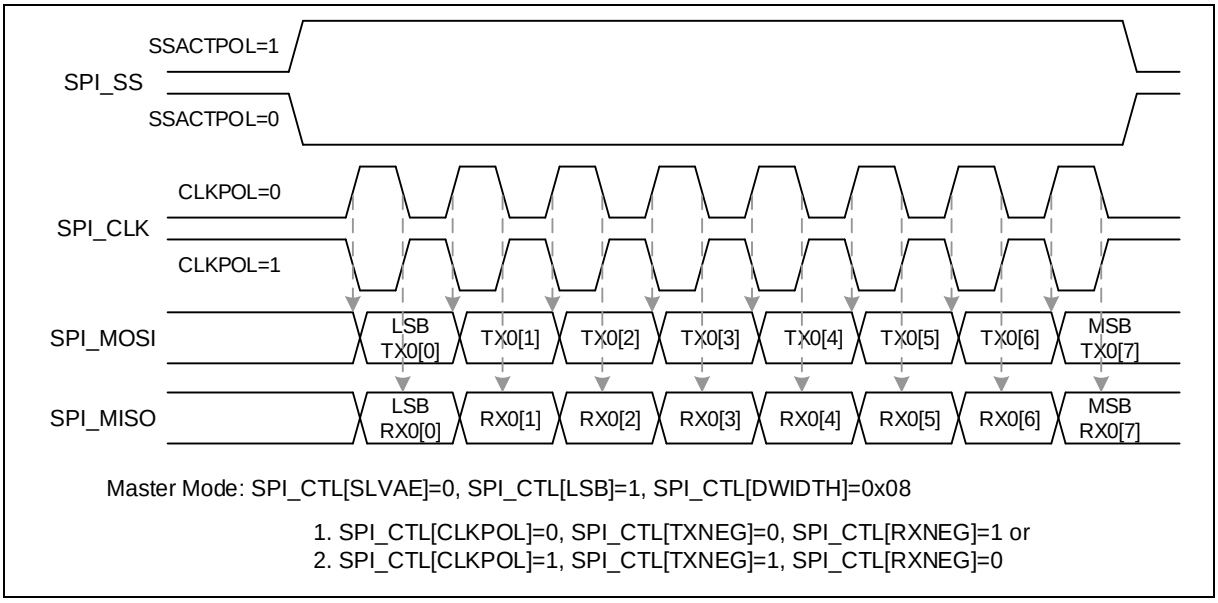


图 6.12-9 主机模式下的SPI 时序(交替SPICLK时钟相位)

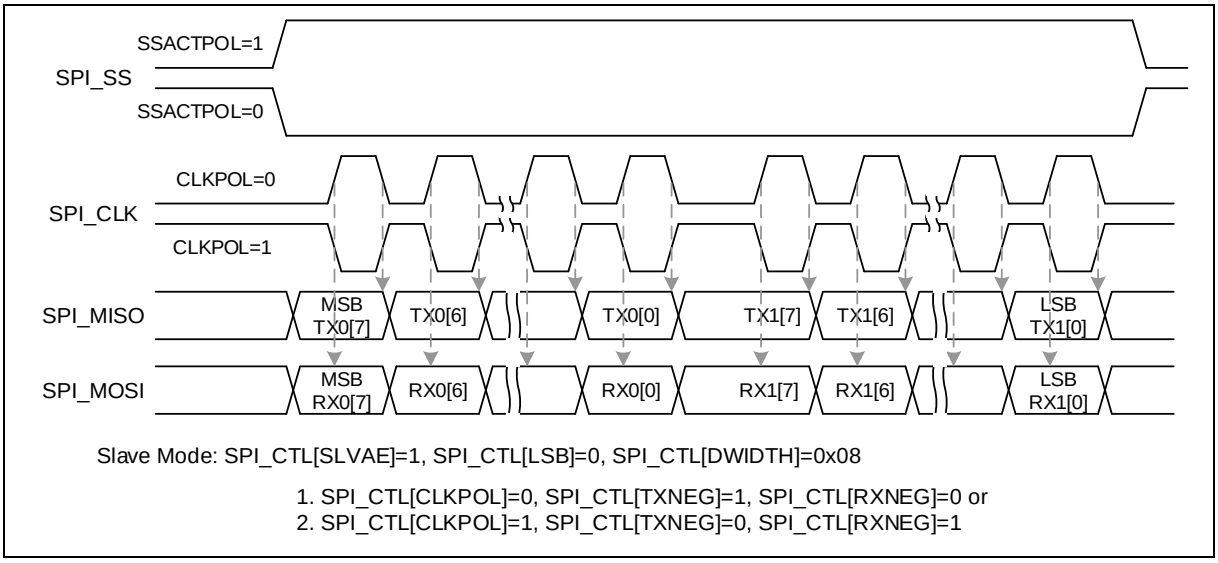


图 6.12-10 从机模式下的SPI时序

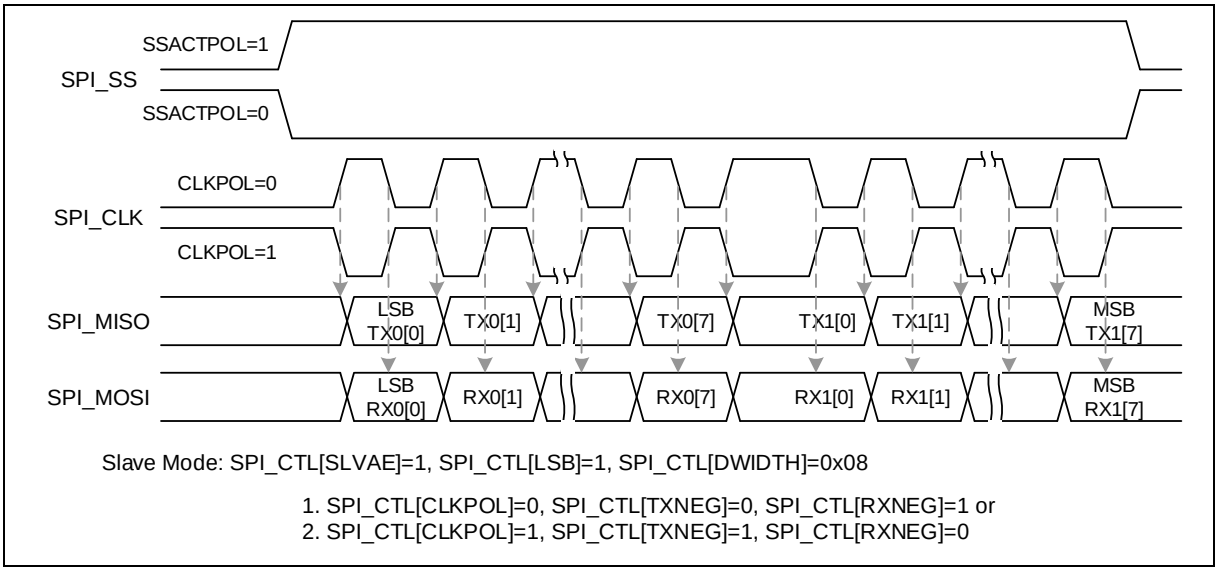


图 6.12-11 从机模式下的SPI时序(交替SPICLK时钟相位)

6.12.6 编程示例

例 1, SPI 控制器作为主机去访问一个片外从机设备，过程如下：

- 数据在SPI总线时钟上升沿锁存
- 数据在SPI总线时钟下降沿传输
- MSB 先传输
- SPI总线时钟空闲状态下为低电平
- 每次发送/接收只有一个字节
- 使用SPI从机选择管脚和一个片外从机相连。从机选择信号为低电平有效

操作流程如下：

2) 设置DIVIDER (SPI_CLKDIV[7:0]) 来确定串行时钟输出频率。
3) 写一个合适的值到SPI_SSCTL 寄存器来对主机相关配置进行设定:
    1. 清除AUTOSS (SPI_SSCTL[3])为0，来禁止自动从机选择功能。
    2. 清除SSACTPOL (SPI_SSCTL[2])为0，来设置从机选择信号有效电平为低电平。
    3. 设置SS (SPI_SSCTL[0])为1，来使能从机片选信号以激活片外从机设备。
4) 通过设置SPI_CTL寄存器来控制 SPI 主机的行为：
    1. 通过设置SLAVE (SPI_CTL[18])为0将 SPI 控制器设为主机设备。
    2. 通过设置CLKPOL (SPI_CTL[3])为0将串行时钟的空闲状态设为低电平。.
    3. 通过设置TXNEG (SPI_CTL[2])为1选择数据在串行时钟的下降沿传输。
    4. 通过设置RXNEG (SPI_CTL[1])为0选择数据在串行时钟的上升沿锁存。.
    5. 通过设置DWIDTH 位域 (SPI_CTL[12:8] = 0x08)来设定一次字传输的长度为8-位。
    6. 通过设置LSB (SPI_CTL[13])为0来设定MSB传输优先。
5) 设置SPIEN (SPI_CTL[0]) 为1来使能SPI接口的数据传输。
6) 如果 SPI 主机想要发送(写)一个字节的数据到片外从机设备，则将所要发送的数据写入SPI_TX 寄存器。
7) 等待 SPI 中断发生（如果中断使能IE位设置为1）或轮询SPIEN (SPI_CTL [0])直到被硬件自动清零。
8) 从寄存器SPI_RX[7:0].中读取接收到的一个字节数据
9) 重复步骤 5) 继续其他数据传输或设置SS [0] (SPI_SSCTL[0])为 0 来停止片外从机设备。

例 2: SPI 控制器作为从机设备，和一片外主机设备相连，外设通过 SPI 接口与片上 SPI 从机通信。过程如下：

- 数据在SPI总线时钟上升沿锁存
- 数据在SPI总线时钟下降沿传输

- LSB 先传输
- SPI总线时钟空闲状态为高电平
- 每次发送/接收一个字节
- 从机选择信号为高电平有效

操作流程如下：

1) 将从机模式的相应配置写入合适值到SPI_SSCTL 寄存器。通过设置SSACTPOL (SPI_SSCTL[2]) 为1，来选择高电平作为从机选择信号的有效输入。
2) 给SPI_CTL 寄存器写入相关的值来控制 SPI 从机的行为。
    1. 通过设置SLAVE (SPI_CTL[18]) 为1 将 SPI 控制器设为从机设备。
    2. 通过设置CLKPOL (SPI_CTL[3])为1 将串行时钟的空闲状态设为高电平。
    3. 通过设置TXNEG (SPI_CTL[2])为1 选择数据在SPI总线时钟的下降沿传输
    4. 通过设置RXNEG (SPI_CTL[1])为0 选择数据在SPI总线时钟的上升沿锁存。
    5. 通过设置DWIDTH 位域 (SPI_CTL[12:8] = 0x08)来设定一次字传输的位长度为8-位。
    6. 通过设置LSB (SPI_CTL[13])为1来设置传输为LSB优先。
3) 设置SPIEN (SPI_CTL[0]) 为1，等待片外主机设备的从机片选触发输入和SPI时钟输入来启动数据传输。
4) 如果 SPI 从机要发送(被读取)一个字节的数据到片外主机，则将所要发送的数据写入到SPI_TX 寄存器。
5) 如果 SPI 从机只是要从片外主机接收(被写入)一字节数据，用户不需要关心什么数据将被传输，SPI_TX 寄存器也不需要用户去更新。
6) 等待 SPI 中断发生（如果中断使能位UNITIEN (SPI_CTL[17])设置为1）.
7) 从寄存器SPI_RX[7:0]中读取接收到的一个字节数据.
8) 重复步骤 4) 继续其他数据传输或停止数据传输.

6.12.7 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移地址	R/W	描述	复位值
SPI 基地址: **SPI0_BA = 0x400E_9000** **SPI1_BA = 0x400E_1000**				
SPI_CTL **x = 0,1**	SPIx_BA+0x00	R/W	SPI 控制寄存器	0x0000_0034
SPI_CLKDIV **x = 0,1**	SPIx_BA+0x04	R/W	SPI 时钟除频寄存器	0x0000_0000
SPI_SSCTL **x = 0,1**	SPIx_BA+0x08	R/W	SPI 从机选择控制	0x0000_0000
SPI_FIFOCTL **x = 0,1**	SPIx_BA+0x10	R/W	SPI FIFO 控制寄存器	0x4400_0000
SPI_STATUS **x = 0,1**	SPIx_BA+0x14	R/W	SPI 状态寄存器	0x0005_0110
SPI_TX **x = 0,1**	SPIx_BA+0x20	W	SPI 数据发送寄存器	0x0000_0000
SPI_RX **x = 0,1**	SPIx_BA+0x30	R	SPI 数据接收寄存器	0x0000_0000

6.12.8 寄存器描述

SPI 控制寄存器 (SPI_CTL)

寄存器	偏移地址	R/W	描述	复位值
SPI_CTL	SPIx_BA+0x00	R/W	SPI 控制寄存器	0x0000_0034

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved				REORDER	SLAVE	UNITIEN	Reserved
15	14	13	12	11	10	9	8
Reserved		LSB	DWIDTH				
7	6	5	4	3	2	1	0
SUSPITV				CLKPOL	TXNEG	RXNEG	SPIEN

位	描述	
[31:20]	**Reserved**	保留.
[19]	**REORDER**	**字节重排序功能使能位** 0 =禁止字节重排序功能. 1 =使能字节重排序功能，在每两个字节之间插入一个字节休眠间隔。字节休眠间隔时间取决于SUSPITV (SPI_CTL[7:4])的设置。 **注:** 字节重排序仅在DWIDTH 被定义为 16 位，24 位和 32 位时有效.
[18]	**SLAVE**	**从机模式控制位** 0 = 主机模式 1 = 从机模式
[17]	**UNITIEN**	**单元传输中断使能位** 0 = 禁止SPI单元传输中断 1 = 使能SPI单元传输中断
[16:14]	**Reserved**	保留.
[13]	**LSB**	**优先发送LSB** 0 = MSB, 具体发送/接收寄存器的哪一位首先被发送/接收，取决于DWIDTH 的设定值。 1 = LSB, SPI TX寄存器的位 0，首先被发送到 SPI 数据输出管脚，从 SPI 数据输入管脚上接收到的第一个数据位将会被放置到 RX 寄存器 (SPI_RX的位0) LSB 的位置.
[12:8]	**DWIDTH**	**数据宽度** 该位域指定在一次发送/接收事务中，有多少个数据位将会被传输。最小位长是 8，最多可以达到 32 位。 DWIDTH = 0x08 … 8 位

		DWIDTH = 0x09 … 9位 …… DWIDTH = 0x1F … 31位 DWIDTH = 0x00 … 32位
[7:4]	**SUSPITV**	**休眠间隔(仅主模式)** 该四位用来配置在一次数据传输过程中连续两个发送/接收事务之间的休眠间隔。休眠间隔是从当前事务字的最后一个时钟边沿到接下来的事务的第一个边沿时钟。缺省值是 0x3. 休眠间隔的周期可以根据下面公式获得： (SUSPITV [3:0] + 0.5) * SPICLK时钟周期 例: SUSPITV = 0x0 … 0.5 SPICLK 时钟周期 SUSPITV = 0x1 … 1.5 SPICLK 时钟周期 …… SUSPITV = 0xE … 14.5 SPICLK 时钟周期 SUSPITV = 0xF … 15.5 SPICLK 时钟周期
[3]	**CLKPOL**	**时钟极性** 0 = SPI总线空闲状态的电平为低. 1 = SPI总线空闲状态的电平为高
[2]	**TXNEG**	**在下降沿发送** 0 = 在 SPI总线时钟的上升沿改变发送数据输出信号 1 = 在 SPI总线时钟的下降沿改变发送数据输出信号
[1]	**RXNEG**	**在下降沿接收** 0 = 在 SPI总线时钟的上升沿锁存接收数据输入信号. 1 = 在 SPI总线时钟的下降沿锁存接收数据输入信号.
[0]	**SPIEN**	**SPI 传输控制使能位** 0 = 禁止传输控制 1 = 使能传输控制 **注1**：在主机模式下，如果在FIFO缓存有数据，在该位设置为1后，开始传输。在从机模式下，当该位设置为1时，该设备已准备好接收数据。 **注2**：当双I/O或四I/O模式使能，不支持字节重排序功能。

SPI 分频器寄存器 (SPI_CLKDIV)

寄存器	偏移地址	R/W	描述	复位值
SPI_CLKDIV	SPIx_BA+0x04	R/W	SPI 时钟分频器寄存器	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
DIVIDER							

位	描述	
[31:8]	Reserved	保留.
[7:0]	DIVIDER	**时钟除频器** 该域的值是产生SPI主机SPI总线时钟、f_{spi_eclk},和外设时钟的频率分频器，可根据下列公式获得所期望的频率： $f_{spi_eclk} = \frac{f_{spi_clock_src}}{(DIVIDER+1)}$ $f_{spi_clock_src}$ 是外设的时钟源，在时钟控制寄存器里定义。

SPI 从机选择寄存器 (SPI_SSCTL)

寄存器	偏移地址	R/W	描述	复位值
SPI_SSCTL	SPIx_BA+0x08	R/W	SPI 从机选择控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
SLVTOCNT							
23	22	21	20	19	18	17	16
SLVTOCNT							
15	14	13	12	11	10	9	8
Reserved		SSINAIEN	SSACTIEN	Reserved		SLVURIEN	SLVBEIEN
7	6	5	4	3	2	1	0
Reserved	SLVTORST	SLVTOIEN	SLV3WIRE	AUTOSS	SSACTPOL	Reserved	SS

位	描述	
[31:16]	SLVTOCNT	**从机超时周期** 在从机模式，当在从机片选信号有效期间有总线时钟输入时，这些位表示超时周期。超时计数器的时钟源是从机外设时钟。如果该位值为0，表示禁止从机模式超时功能。
[15:14]	Reserved	保留.
[13]	SSINAIEN	**从机片选无效中断使能位** 0 = 禁止从机片选无效中断 1 = 使能从机片选无效中断
[12]	SSACTIEN	**从机片选有效中断使能位** 0 = 禁止从机片选有效中断 1 = 使能从机片选有效中断
[11:10]	Reserved	保留.
[9]	SLVURIEN	**从机模式错误1中断使能位** 0 = 禁止从机模式错误1中断 1 = 使能从机模式错误1中断
[8]	SLVBEIEN	**从机模式错误0中断使能位** 0 = 禁止从机模式错误0中断 1 = 使能从机模式错误0中断
[7]	Reserved	保留.
[6]	SLVTORST	**从机模式超时FIFO清除使能位** 0 = 当从机模式超时事件发生时，禁止FIFO清除。 1 = 当从机模式超时事件发生时，使能FIFO清除。 注：当有从机模式超时事件发生，若使能FIFO清除功能，TXRST和RXRST会自动激活

[5]	**SLVTOIEN**	**从机模式超时中断使能位** 0 = 禁止从机模式超时中断 1 = 使能从机模式超时中断.
[4]	**SLV3WIRE**	**从机3-线模式使能位** 该位用来在从机模式忽略从机选择信号，SPI控制器可以工作于3-线接口。包括SPI0_CLK, SPI0_MISO 和 SPI0_MOSI管脚。 0 = 4-线双向接口 1 = 3-线双向接口
[3]	**AUTOSS**	**自动从机片选功能使能位(仅主机模式)** 0 = 禁止自动从机片选功能。此时从机片选信号通过置位或清除SS(SPI_SSCTL[0])位来置有效或无效 1 = 使能自动从机片选功能，SS信号会自动产生。这意味着当发送或接收开始，SPI_SSCTL[0],里设置的从机选择信号会由SPI控制器自动置有效，传输完成会自动置无效
[2]	**SSACTPOL**	**从机片选有效极性位** 该位定义从机片选信号(SPIn_SS)有效的极性 0 = 从机片选信号 SPIn_SS低有效 1 = 从机片选信号 SPIn_SS高有效
[1]	**Reserved**	保留.
[0]	**SS**	**从机片选控制位 (仅主机模式)** 如果AUTOSS位设置为0 0 = 设置SPIn_SS线为无效状态 1 = 设置SPIn_SS线为激活状态 如果AUTOSS位设置为1 0 = 保持SPIn_SS线为无效状态 1 = 在数据传输期间，SPIn_SS线将自动进入激活状态，同样在其它时间将自动进入无效状态。SPIn_SS的激活状态在SSACTPOL (SPI_SSCTL[2])中指定。

SPI FIFO 控制寄存器 (SPI_FIFOCTL)

寄存器	偏移地址	R/W	描述	复位值
SPI_FIFOCTL	SPIx_BA+0x10	R/W	SPI FIFO 控制寄存器	0x4400_0000

31	30	29	28	27	26	25	24
Reserved	TXTH			Reserved	RXTH		
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved						TXFBCLR	RXFBCLR
7	6	5	4	3	2	1	0
TXUFIEN	TXUFPOL	RXOVIEN	RXTOIEN	TXTHIEN	RXTHIEN	TXRST	RXRST

位	描述	
[31]	**Reserved**	保留.
[30:28]	**TXTH**	**发送 FIFO 阈值** 如果发送FIFO缓存的有效数据个数小于或者等于TXTH的设置，TXTHIF (SPI_STATUS[18])将会被置为1，否则TXTHIF (SPI_STATUS[18])将会被置0。
[27]	**Reserved**	保留.
[26:24]	**RXTH**	**接收 FIFO阈值** 如果接收FIFO缓存的有效数据个数大于RXTH的设置，RXTHIF (SPI_STATUS[10])将会被设置为 1， 否则RXTHIF (SPI_STATUS[10])将会被设置为 0。
[23:10]	**Reserved**	保留.
[9]	**TXFBCLR**	**发送FIFO 缓存清除位** 0 = 无影响 1 = 仅清除发送FIFO **注:** 该位由硬件自动清0。
[8]	**RXFBCLR**	**接收FIFO 缓存清除位** 0 = 无影响 1 = 仅清除接收FIFO **注:** 该位由硬件自动清0。
[7]	**TXUFIEN**	**从机发送下溢中断使能位** 0 = 禁止从机发送FIFO下溢中断 1 = 使能从机发送FIFO下溢中断
[6]	**TXUFPOL**	**发送下溢数据极性** 0 = 在从机模式，如果有发送下溢事件发生，SPI数据输出保持为0。

		1 = 在从机模式，如果有发送下溢事件发生，SPI数据输出保持为1。. **注1:**.在串行时钟输入且硬件同步后，下溢事件激活。所以最先的1~3 bit （取决于系统时钟和外设时钟之间的关系）数据输出会是最后传输的数据。 **注2:**如果系统时钟的频率接近外设时钟，它们可能需要3个位时间来报告发送下溢数据输出。
[5]	**RXOVIEN**	**接收FIFO 溢出中断使能位** 0 = 禁止接收FIFO 溢出中断 1 = 使能接收FIFO 溢出中断
[4]	**RXTOIEN**	**从机接收超时中断使能位** 0 = 禁止接收超时中断 1 = 使能接收超时中断
[3]	**TXTHIEN**	**发送FIFO 阀值中断使能位** 0 = 禁止发送FIFO 阀值中断 1 = 使能发送FIFO 阀值中断
[2]	**RXTHIEN**	**接收FIFO阀值中断使能位** 0 = 禁止接收FIFO阀值中断 1 = 使能接收FIFO阀值中断
[1]	**TXRST**	**清除发送 FIFO 控制** 0 =无影响. 1 =清除发送FIFO控制，TXFULL (SPI_STATUS[17])会被清0，TXEMPTY (SPI_STATUS[16])会被置1 。该位变为1后，大约经过3个系统时钟周期+3个外设时钟周期，硬件将该位清除为0。 **注:** 如果有从机接收超时事件发生，当SLVTORST (SPI_SSCTL[6])使能时，TXRST将被设置为1.
[0]	**RXRST**	**清除接收 FIFO 控制** 0 =无影响. 1 =清除接收FIFO控制，RXFULL (SPI_STATUS[9]) 会被清0，RXEMPTY (SPI_STATUS[8])会被置1 。该位变为1后，大约经过3个系统时钟周期+3个外设时钟周期，硬件将该位清除为0。 **注:** 如果有从机接收超时事件发生，当SLVTORST (SPI_SSCTL[6])使能时，RXRST将被设置为1.

SPI 状态寄存器 (SPI_STATUS)

寄存器	偏移地址	R/W	描述	复位值
SPI_STATUS	SPIx_BA+0x14	R/W	SPI 状态寄存器	0x0005_0110

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
TXRXRST	Reserved			TXUFIF	TXTHIF	TXFULL	TXEMPTY
15	14	13	12	11	10	9	8
SPIENSTS	Reserved		RXTOIF	RXOVIF	RXTHIF	RXFULL	RXEMPTY
7	6	5	4	3	2	1	0
SLVURIF	SLVBEIF	SLVTOIF	SSLINE	SSINAIF	SSACTIF	UNITIF	BUSY

位	描述	
[31:28]	**TXCNT**	**发送 FIFO 数据个数 (只读)** 该位域表明发送FIFO缓存的有效数据个数。
[27:24]	**RXCNT**	**接收 FIFO 数据个数 (只读)** 该域表明接收 FIFO 缓存有效数据个数..
[23]	**TXRXRST**	**FIFO 清除状态 (只读)** 0 = TXRST(SPI_FIFOCTL[1]) 或 RXRST(SPI_FIFOCTL[0]).的FIFO缓存清除功能已经完成 1 = TXRST(SPI_FIFOCTL[1]) 或 RXRST(SPI_FIFOCTL[0]).的FIFO缓存清除功能正在进行 **注:** TXRST(SPI_FIFOCTL[1]) 或 RXRST(SPI_FIFOCTL[0]).的FIFO清除操作都需要3个系统时钟周期+3个外设时钟周期。用户可以检查该状态位来检测操作是否完成。
[22:20]	**Reserved**	保留.
[19]	**TXUFIF**	**发送下溢中断标志** 当发送下溢事件发生时，该位被设置为1，数据输出管脚的状态取决于TXUFPOL的设置。 0 = 无影响 1 = 当从机片选信号有效时，发送FIFO和发送移位寄存器中没有数据。 **注 1:** 该位通过写1清除。 **注 2:** 当从机片选信号有效时，如果复位从机的发送电路，在3个系统时钟周期和2个外设时钟周期后，此时复位操作已经完成，该位将被设置为1。
[18]	**TXTHIF**	**发送FIFO阀值中断标志(只读)** 0 = 发送FIFO缓存中的有效数据个数大于TXTH (SPI_FIFOCTL[30:28]).设置的值。 1 = 发送FIFO缓存中的有效数据个数少于或等于TXTH (SPI_FIFOCTL[30:28]).设置的值 注：如果TXTHIEN (SPI_FIFOCTL[3]) = 1 和 TXTHIF = 1,SPI控制器会产生SPI中断请求
[17]	**TXFULL**	**发送FIFO缓存满标志 (只读)** 0 = 发送FIFO 缓存未满

		1 = 发送FIFO 缓存已满
[16]	**TXEMPTY**	**发送FIFO缓存空标志 (只读)** 0 = 发送FIFO 缓存不空 1 = 发送FIFO 缓存已空
[15]	**SPIENSTS**	**SPI 使能状态位(只读)** 0 = 表示SPI控制器是禁止的 1 = 表示SPI控制器已激活 **注:** SPI外设时钟与系统时钟不同步。为了确保SPI控制器已经被禁止，该位指示了SPI控制器中SPIEN (SPI_CTL[0])真实状态。
[14:13]	**Reserved**	保留.
[12]	**RXTOIF**	**接收超时中断标志** 0 = 没有接收FIFO超时事件 1 = 接收 FIFO 缓存非空且主机模式下，超过64个SPI时钟周期或从机模式下超过576个SPI引擎时钟周期，接收FIFO缓存上没有读操作。当接收FIFO缓存被软件读取，超时状态会自动清0。 **注:** 该位通过写1清除
[11]	**RXOVIF**	**接收FIFO溢出中断标志** 0 = 接收FIFO没有溢出 1=接收FIFO溢出，当接收FIFO缓存已满，接下来的数据将被丢失，同时该位被设置为1. **注:** 该位通过写1清除
[10]	**RXTHIF**	**接收FIFO阀值中断标志(只读)** 0 = 接收FIFO缓存的有效数据个数少于或等于RXTH (SPI_FIFOCTL[26:24]).设置的值 1 = 接收FIFO缓存的有效数据个数大于RXTH (SPI_FIFOCTL[26:24]).设置的值 注：如果RXTHIEN (SPI_FIFOCTL[2]) = 1 和 RXTHIF = 1, SPI控制器会产生SPI中断请求
[9]	**RXFULL**	**接收FIFO 缓存满标志(只读)** 0 = 接收 FIFO 缓存未满 1 = 接收 FIFO 缓存已满
[8]	**RXEMPTY**	**接收FIFO 缓存空标志 (只读)** 0 = 接收 FIFO 缓存不空 1 = 接收 FIFO 缓存为空
[7]	**SLVURIF**	**从机模式错误1中断标志（只读）** 在从机模式，当从机片选线进入无效状态，发送下溢事件发生，该位置1 0 = 没有从机模式错误1发生 1 = 从机模式错误1发生
[6]	**SLVBEIF**	**从机模式错误0中断标志（只读）** 在从机模式，当从机片选信号线进入无效状态时，如果位计数与DWIDTH (SPI_CTL[12:8])不一致，该中断标志将被设置为1。 0 = 未发生从机模式错误0事件 1 = 发生从机模式错误0事件 **注:**如果从机片选激活，但是没有总线时钟输入，当从机片选进入无效状态时，SLVBEIF (SPI_STATUS[6])也将被设置为1。

[5]	**SLVTOIF**	**从机超时中断标志(只读)** 当从机片选信号激活和SLVTOCNT (SPI_SSCTL[31:16])的值不为0，在检测到总线时钟时，SPI控制器逻辑的从机超时计数器开始计数。在事务完成之前，如果超时计数器的值大于或等于SLVTOCNT (SPI_SSCTL[31:16])里的值。将发生从机超时中断事件。 0 = 从机超时未发生 1 = 发生从机超时 **注:** 如果DWIDTH (SPI_CTL[12:8])被设置成0x10，一次传输时间等于16个位总线时钟周期
[4]	**SSLINE**	**从机片选线总线状态(只读)** 0 = 从机片选线状态为0 1 = 从机片选线状态为1 **注:** 如果SSACTPOL (SPI_SSCTL[2])设置为0，而且SSLINE的值为1，SPI从机片选处于无效状态。
[3]	**SSINAIF**	**从机片选无效中断标志** 0 = 从机片选无效中断被清除或没有发生 1 = 发生了从机片选无效中断 **注:** 该位通过写1清除。
[2]	**SSACTIF**	**从机片选激活中断标志** 0 = 从机片选激活中断被清除或未发生 1 = 发生了从机片选激活中断 **注:** 该位通过写1清除。
[1]	**UNITIF**	**单元传输中断标志** 0 = 该位清除后，没有事务完成 1 = SPI 控制器完成一个单元传输 **注:** 该位通过写1清除
[0]	**BUSY**	**SPI 单元 总线状态 (只读)** 0 = SPI 控制器处于空闲状态 1 = SPI 控制器处于忙状态 下面列出忙状态的条件： a. SPIEN (SPI_CTL[0]) = 1 和 TXEMPTY (SPI_STATUS[16]) = 0. b. SPI 主机模式时，TXEMPTY (SPI_STATUS[16]) = 1，但是当前传输还没有完成 c. SPI 从机模式时，SPIEN (SPI_CTL[0]) = 1且当从机片选激活时，SPI内部逻辑有串行时钟输入。 d. SPI 从机模式时，SPIEN (SPI_CTL[0]) = 1且发送缓存不为空(即使从机片选无效).

SPI 数据发送寄存器 (SPI_TX)

寄存器	偏移地址	R/W	描述	复位值
SPI_TX	SPIx_BA+0x20	W	SPI 数据发送寄存器	0x0000_0000

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

位	描述	
[31:0]	TX	**数据发送寄存器** 数据发送寄存器把数据传送到8级发送FIFO缓存。数据的有效长度取决于DWIDTH (SPI_CTL[12:8])的设置。 例如，如果DWIDTH为 0x08 ， 则TX[7:0]位将会被发送。如果DWIDTH为 0x00，则 SPI 控制器会执行32-位数据传输。

SPI 数据接收寄存器 (SPI_RX)

寄存器	偏移地址	R/W	描述	复位值
SPI_RX	SPIx_BA+0x30	R	SPI 数据接收寄存器	0x0000_0000

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

位	描述	
[31:0]	RX	**数据接收寄存器** 该控制器有8级FIFO缓存。 数据接收寄存器保存从SPI数据输入管脚接收到的数据。如果RXEMPTY (SPI_STATUS[8]未设置为1， 接收FIFO缓存可以通过软件读该寄存器访问。该寄存器只读。

6.13 SPI 同步串行接口控制器 (主机模式, SPIM)

6.13.1 概述

当从一个外围设备接收数据时，SPI同步串行接口控制器执行串-并的转换，而在收到CPU数据后执行并-串的转换。该控制器可以驱动两个外设（内嵌的SPI Flash或外部的SPI Flash），扮演SPI主机的角色。当数据传输完成，它可以产生中断信号，并且可以通过写1来清除中断标志。从机片选的有效电平根据外设可以选择低或高电平有效。写一个分频值到SPIM_CTL1寄存器可以设置串行输出时钟的频率。该控制器包含4个32位发送/接收缓存，可以提供1到4个突发操作模式。每次传输的位数可以是8, 16, 24, 或 32。一次传输中最大可以连续发送/接收四个数据。

6.13.2 特性

- 支持 SPI 主机模式
- 支持 DMA 模式 (DMA 写 和 DMA 读)，直接内存映射模式 (DMM)，和I/O 模式
- 每笔传输数据可以是8, 16, 24, 或 32位宽度
- 支持标准 (1-bit), dual (2-bit), 和 quad (4-bit) I/O 传输模式
- 提供突发操作模式，一次传输中最大可以连续发送/接收四个数据。
- 两个从机/设备片选（内嵌的SPI Flash或外部的SPI Flash）
- 采用统一时钟的全同步设计架构

6.13.3 框图

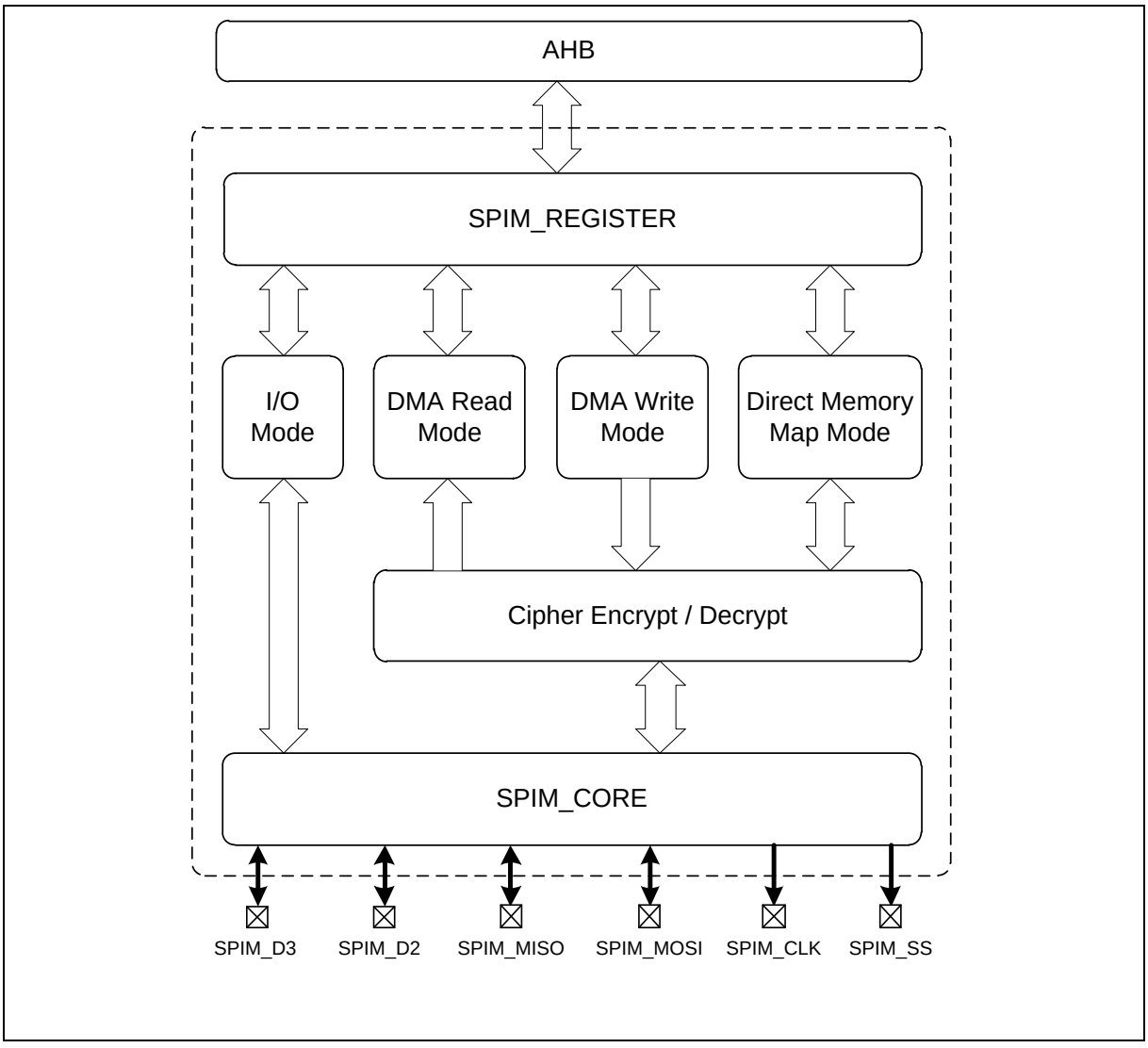


图 6.13-1 SPIM 模块框图

6.13.4 功能描述

6.13.4.1 SPIM 时序图

SPI传输时序图如下所示：

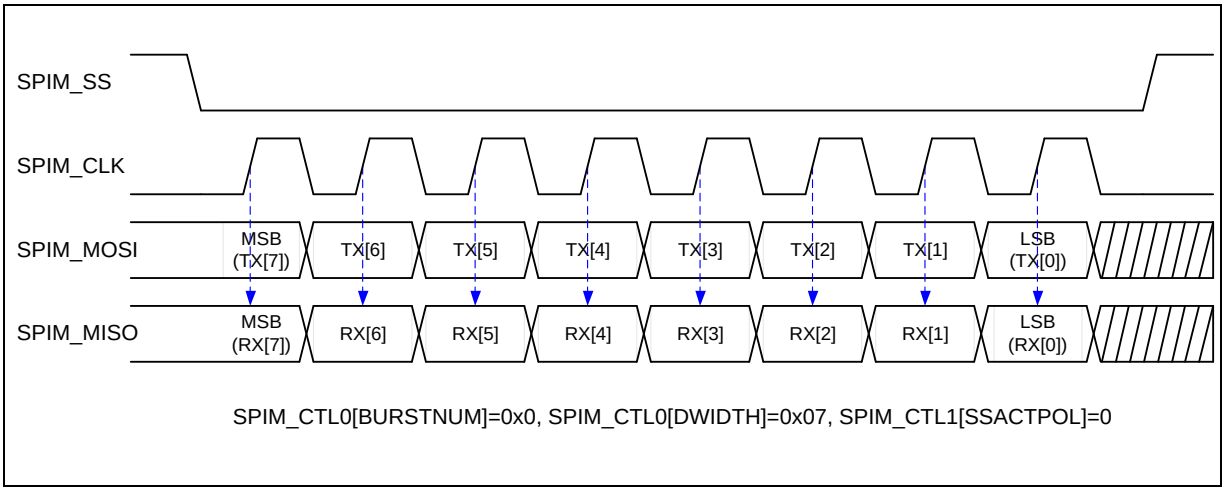


图 6.13-2 SPIM 时序图

6.13.4.2 SPIM 不带DMA的编程示例 (I/O 模式)

根据如下的要求访问设备:

- 数据高位优先传输 MSB.
- 每次发送/接收只有一个字节.
- 片选信号低有效.

你应该做如下基本的动作 (具体的步骤请参考设备的规格):

1. 写一个分频值到 DIVIDER (SPIM_CTL1[31:16]) 以确定串行时钟的频率.
2. 设置 SSACTPOL (SPIM_CTL1[5]) 为 0 和 IFSEL (SPIM_CTL1[7:6]) 来激活你想要访问的设备。
3. 当发送 (写) 数据到设备:
   - QDIODIR (SPIM_CTL0[15]) = 1
   - BURSTNUM (SPIM_CTL0[14:13]) = 0x0
   - DWIDTH (SPIM_CTL0[12:8]) = 0x07
   - 写你要发送的数据到 SPIM_TX0[7:0]
4. 当从设备接收（读）数据:
   - QDIODIR (SPIM_CTL0[15]) = 0
   - BURSTNUM (SPIM_CTL0[14:13]) = 0x0
   - DWIDTH (SPIM_CTL0[12:8]) = 0x07
5. 设置 SPIMEN (SPIM_CTL1[0])为 1，开始传输 。
6. 等待中断或轮询 SPIMEN (SPIM_CTL1[0]) 直到它变为 0
7. 当接收（读）来自设备的数据:
   - 从 SPIM_RX0[7:0] 寄存器读出接收到的数据.

6.13.4.3 带DMA的SPIM 编程示例

如果用户想用DMA的功能访问设备，三个额外的寄存器需要被配置，包括SPIM_DMATBCNT,

SPIM_SRAMADDR 和 SPIM_FADDR。DMA功能可以被用来支持加载启动代码（从外设读数据到系统内存）或从系统内存读数据保存到外设。用户必须定义源地址、长度和目的地址，然后硬件会自动搬运想要的数据长度到指定的目的地址。

6.13.4.4 代码启动过程 (DMA 读模式)

步骤 1: 通过使用I/O模式检查设备ID 确认设备已连接

步骤 2:

1. 在 SPIM_SRAMADDR 寄存器设置目标内存地址
2. 在 SPIM_DMATBCNT 寄存器设置启动代码的长度
3. 在 SPIM_FADDR 寄存器设置 SPI Flash 的起始地址.
4. 设置 OPMODE (SPIM_CTL0[23:22]) 为 DMA 读模式.
5. 设置 CMDCODE (SPIM_CTL0[31:24]) 为 0x03 命令码.
6. 设置 SPIMEN (SPIM_CTL1[0]) 启动，然后 SPIM 会从 SPIM_FADDR 到 SPIM_SRAMADDR 搬运 SPIM_DMATBCNT 中设置的传输长度的代码块
7. 等待中断或轮询 SPIMEN (SPIM_CTL1[0]) 位直到它变为 0.

如果使用的SPI Flash支持其他相关的读命令，用户也可以使用下面的读命令代码.

1. 快速读 (0Bh), 设置 CMDCODE (SPIM_CTL0[31:24])为 0x0B.
2. 快速双线读 (3Bh), 设置 CMDCODE (SPIM_CTL0[31:24]) 为 0x3B.
3. 四线读 (EBh), 设置 CMDCODE (SPIM_CTL0[31:24]) 为 0xEB.

6.13.4.5 从系统内存到外设搬运数据 (DMA 写模式)

步骤 1: 写SPI Flash之前先擦除SPI Flash（使用I/O模式）

步骤 2:

1. 发送写使能命令到 SPI Flash (使用 I/O 模式).
2. 在 SPIM_SRAMADDR 寄存器里设置源内存地址
3. 在 SPIM_DMATBCNT 寄存器里设置传输数据个数
4. 在 SPIM_FADDR 里设置 SPI Flash
5. 设置 OPMODE (SPIM_CTL0[23:22]) 为 DMA 写模式.
6. 设置 CMDCODE (SPIM_CTL0[31:24]) 为 0x02 命令代码.
7. 设置 SPIMEN (SPIM_CTL1[0]) 启动.
8. 等待中断或轮询 SPIMEN (SPIM_CTL1[0]) 直到它变为 0.

如果使用的SPI Flash 支持其他相关的写命令，用户也可以使用下面命令代码（用户需要参考SPI Flash 的规格来选择合适的四线写模式命令）

1. 四线写, 设置 CMDCODE (SPIM_CTL0[31:24]) 为 0x32 用于 类型_1 的编程命令 (参考 图 6.13-3).
2. 四线写, 设置 CMDCODE (SPIM_CTL0[31:24]) 为 0x38 用于 类型_2 的编程命令 (参考图 6.13-4).
3. 四线写, 设置 CMDCODE (SPIM_CTL0[31:24]) 为 0x40 用于 类型_3 的编程命令 (参考 图 6.13-5).

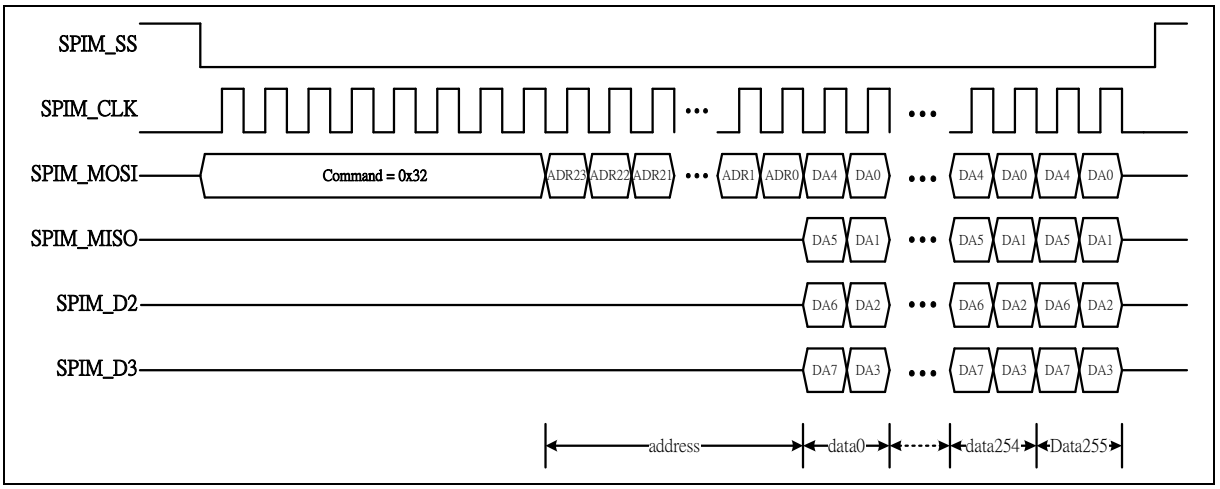


图 6.13-3 类型_1 四线模式写的编程命令流程.

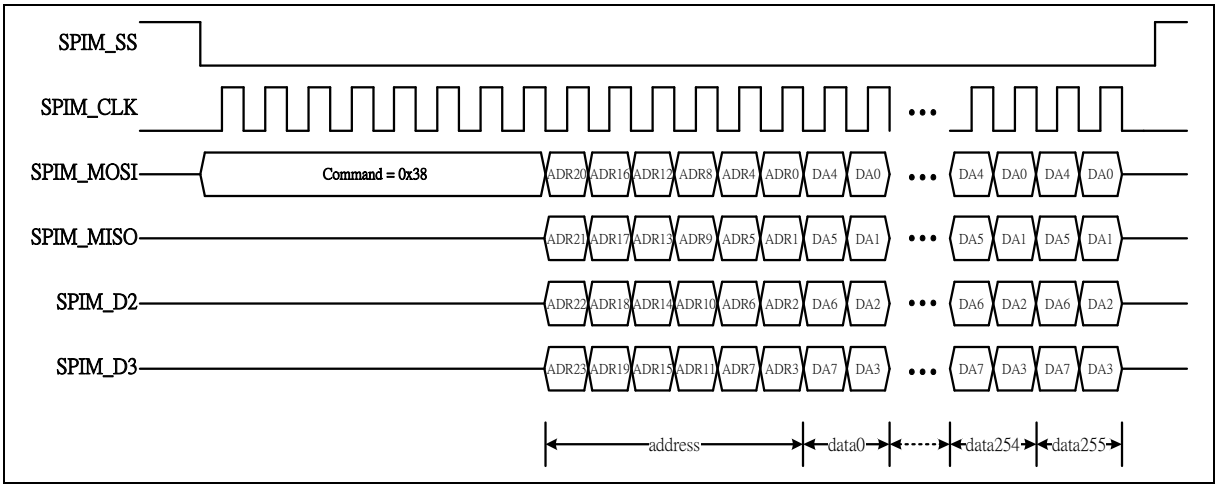


图 6.13-4 类型_2四线模式写的编程命令流程.

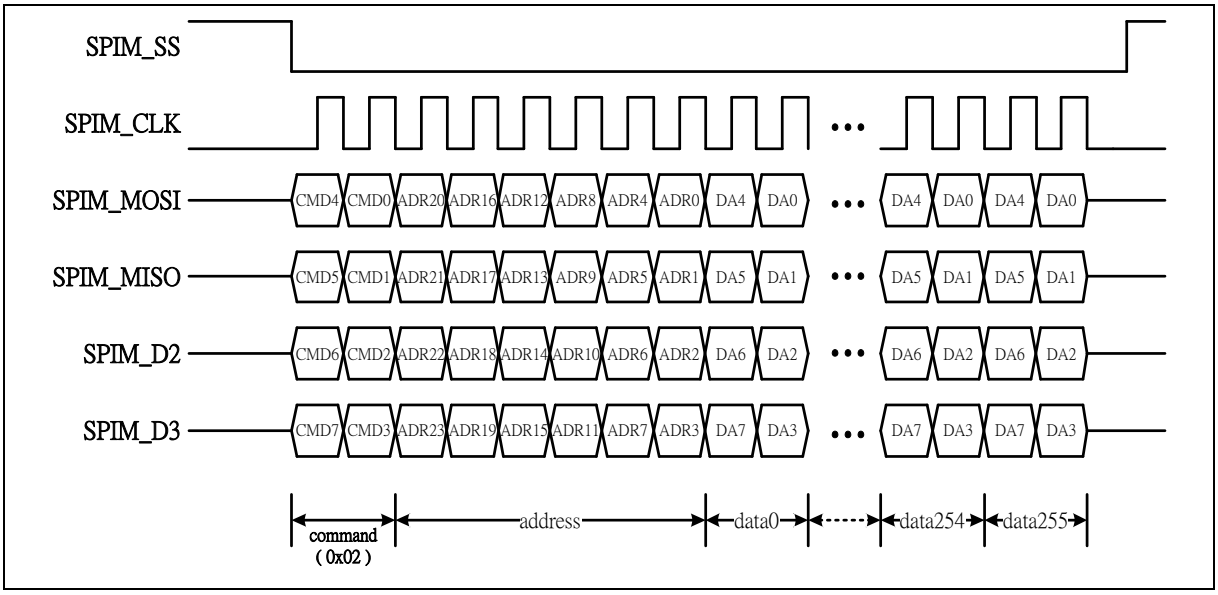


图 6.13-5 类型_3四线模式写的编程命令流程.

6.13.4.6 直接内存映射模式

在直接内存映射模式，SPI Flash可以被当作一个ROM模块。控制器会转变AHB时钟进入SPI Flash，不需要CPU设置相关的SPI Flash命令。CPU需要做的唯一设置是设置CMDCODE (SPIM_CTL0[31:24])为0x03, 0x0B, 0x3B, 或 0xEB命令代码，然后用户就可以像一个ROM模块一样访问SPI Flash。

在直接内存映射模式，在一个直接内存映射访问后，会预先取四个字的Flash数据。如果用户想要在直接内存映射访问之后修改SPIM硬件寄存器，需要等至少250个外设时钟周期（SPIM总线周期）

6.13.5 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读写, **C**: 只能写0

寄存器	偏移地址	R/W	描述	复位值
SPIM 基地址: **SPIM_BA = 0x4000_7000**				
SPIM_CTL0	SPIM_BA+0x00	R/W	控制和状态寄存器 0	0x00C0_0002
SPIM_CTL1	SPIM_BA+0x04	R/W	控制寄存器 1	0x0000_0010
SPIM_VALIDCTL	SPIM_BA+0x08	R/W	确认检查寄存器	0x0000_0000
SPIM_RXCLKDLY	SPIM_BA+0x0C	R/W	Rx 时钟延迟控制寄存器	0x0000_0000
SPIM_RX0	SPIM_BA+0x10	R	数据接收寄存器0	0x0000_0000
SPIM_RX1	SPIM_BA+0x14	R	数据接收寄存器1	0x0000_0000
SPIM_RX2	SPIM_BA+0x18	R	数据接收寄存器2	0x0000_0000
SPIM_RX3	SPIM_BA+0x1C	R	数据接收寄存器3	0x0000_0000
SPIM_TX0	SPIM_BA+0x20	R/W	数据发送寄存器0	0x0000_0000
SPIM_TX1	SPIM_BA+0x24	R/W	数据发送寄存器1	0x0000_0000
SPIM_TX2	SPIM_BA+0x28	R/W	数据发送寄存器2	0x0000_0000
SPIM_TX3	SPIM_BA+0x2C	R/W	数据发送寄存器3	0x0000_0000
SPIM_SRAMADDR	SPIM_BA+0x30	R/W	SRAM 内存地址寄存器	0x0000_0000
SPIM_DMATBCNT	SPIM_BA+0x34	R/W	DMA 传输字节数寄存器	0x0000_0000
SPIM_FADDR	SPIM_BA+0x38	R/W	SPI Flash 地址寄存器	0x0000_0000

注: 如果软件想写该外设的任何寄存器，SPIM_CTL1 寄存器的 SPIMEN 位应该要为低状态

6.13.6 寄存器描述

控制和状态寄存器 0 (SPIM_CTL0)

寄存器	偏移地址	R/W	描述	复位值
SPIM_CTL0	SPIM_BA+0x00	R/W	控制和状态寄存器 0	0x00C0_0002

31	30	29	28	27	26	25	24
CMDCODE							
23	22	21	20	19	18	17	16
OPMODE		BITMODE		SUSPITV			
15	14	13	12	11	10	9	8
QDIODIR	BURSTNUM		DWIDTH				
7	6	5	4	3	2	1	0
IF	IEN	B4ADDREN	Reserved		BALEN	Reserved	CIPHOFF

位	描述	
[31:24]	CMDCODE	**页编程命令码** 0x02 = 页编程 (用于 DMA 写模式). 0x32 = 用类型_1编程流程的四线模式的页编程(用于 DMA 写模式). 0x38 = 用类型_2编程流程的四线模式的页编程(用于 DMA 写模式). 0x40 = 用类型_3编程流程的四线模式的页编程(用于 DMA 写模式). 其他 = 保留. **读命令码** 0x03 = 标准读 (用于DMA读/DMM模式). 0x0B = 快速读 (用于DMA读/DMM模式). 0x3B = 快速两线读 (用于DMA读/DMM模式). 0xEB = 四线读 (用于DMA读/DMM模式). 其他 = 保留. **注1:** 在使用四线模式的页编程/或四线读命令之前， SPI Flash的四线模式必须先通过I/O模式使能 。 **注2:** 哪个SPI Flash支持这些命令码，请看SPI Flash 规格 **注3:** 类型_1, 类型_2, 和 类型_3 的编程流程，参考图 6.13-3, 图 6.13-4, 和 图 6.13-5.
[23:22]	OPMODE	**SPI 功能操作模式** 00 = I/O 模式. 01 = DMA 写模式. 10 = DMA 读模式. 11 = 直接内存映射模式 (DMM) (默认). **注:** 在DMA 写模式，每次操作硬件会发送正好一页的编程命令。用户必须留意跨页的情况

[21:20]	**BITMODE**	**SPI 接口位模式** 00 = 标准模式. 01 = 两线模式. 10 = 四线模式. 11 = 保留. **注:** 仅用于I/O 模式.
[19:16]	**SUSPITV**	**挂起时间间隔** 这四个位域提供一次传输中两个连续发送/接收事务之间的挂起时间间隔的配置。复位值是0x00。当BURSTNUM = 00，设置该域没有影响。想要的时间间隔可以根据下面的等式获得（从当前时钟的最后一个下降沿到下一个时钟的上升沿） (SUSPITV+2)* SCLK的周期 0x0 = 2 SCLK 时钟周期 0x1 = 3 SCLK时钟周期. …… 0xE = 16 SCLK时钟周期. 0xF = 17 SCLK时钟周期. **注:** 仅用于 I/O 模式.
[15]	**QDIODIR**	**两线或四线模式的SPI 接口方向选择** 0 = 接口信号是输入 1 = 接口信号是输出. **注:** 仅用于 I/O 模式..
[14:13]	**BURSTNUM**	**发送/接收 突发个数** 该域指定在一次传输中多少个发送/接收事务应该被连续执行 00 = 在一次传输中，仅执行一个发送/接收事务 01 = 在一次传输中，执行两个连续的发送/接收事务. 10 = 在一次传输中，执行三个连续的发送/接收事务. 11 = 在一次传输中，执行四个连续的发送/接收事务.. **注:** 仅用于 I/O 模式
[12:8]	**DWIDTH**	**发送/接收位长度** 该域指定一次发送/接收事务中多少位被发送/接收 0x7 = 8 位. 0xF = 16 为. 0x17 = 24 为. 0x1F = 32 为. 其他 = 不正确的传输结果 **注1:** 仅用于 I/O 模式 **注2:** 仅 8-, 16-, 24-, 和 32-位是允许的，其他位长度会导致不正确的传输
[7]	**IF**	**中断标志** 写操作: 0 = 没有影响. 1 = 写1清0. 读操作:

		0 = 传输还没有完成. 1 = 传输已完成.
[6]	**IEN**	**中断使能控制** 0 = SPIM 中断禁止. 1 = SPIM 中断使能.
[5]	**B4ADDREN**	**4-字节地址模式使能控制** 0 = 4-字节地址模式禁止. 1 = 4-字节地址模式使能. **注:** 用于 DMA 写/DMA 读/DMM 模式.
[4:3]	**Reserved**	保留.
[2]	**BALEN**	**在密码使能和禁止控制之间平衡AHB控制时间** 当密码使能，AHB控制信号根据编码解码的计算会延时一段时间，所以如果设置BALEN 为1， 会使密码使能和密码禁止时的AHB信号处理时间相等 **注:**.仅当密码禁止时有用
[1]	**Reserved**	保留.
[0]	**CIPHOFF**	**密码禁止控制** 0 = 密码功能使能. 1 = 密码功能禁止 **注1:** 密码功能仅可以被禁止，如果用户想在禁止操作之后使能密码功能，芯片必须重新复位。 **注2:** .如果芯片没有任何钥匙，密码会被自动禁止 **注3:** 在 ICE 模式，密码会被自动禁止.

控制寄存器 1 (SPIM_CTL1)

寄存器	偏移地址	R/W	描述	复位值
SPIM_CTL1	SPIM_BA+0x04	R/W	控制寄存器1	0x0000_0010

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

位	描述	
[31:16]	**DIVIDER**	**时钟分频器寄存器** 该域的值是系统时钟的频率分频值，用来产生SPIM_CLK引脚上输出的串行时钟，想要的频率可以根据下面的公式获得。 $f_{SPIM_CLK} = \frac{f_{SYS_CLK}}{(DIVIDER)*2}$ **注:** .当设置DIVIDER为0，SPIM_CLK的频率会等于SYS_CLK.的频率
[15:12]	**Reserved**	保留.
[11:8]	**IDLECNT**	**空闲间隔** 在 DMM 模式, IDLECNT 用来控制两次SPI Flash访问之间的最小的空闲时间. **注:** 仅用于DMM 模式.
[7:6]	**IFSEL**	**设备/从机 接口选择** 00 = SPI 接口来自GPIO. 01 = SPI 接口来自 MCP. 10 = SPI 接口来自 MCP64. 11 = 保留. **注:** MCP 和 MCP64 仅由MCP SPI Flash的布局来参考
[5]	**SSACTPOL**	**从机选择激活电平** 该位定义了设备/从机选择信号的激活电平(SPIM_SS). 0 = SPIM_SS 从机选择信号低有效. 1 = SPIM_SS从机选择信号高有效.
[4]	**SS**	**从机选择激活使能控制** 0 = SPIM_SS 是在激活的电平

		1 = SPIM_SS 是在非激活的电平 **注:**.在给定的时间，这个接口仅驱动一个设备/从机。所以在开始任何读或写传输之前，已选择的设备的从机选择必须设置成它的有效电平。
[3:1]	**Reserved**	保留.
[0]	**SPIMEN**	**运行和忙状态** 写操作: 0 = 没影响 1 =开始传输，该位正在传输过程中仍然保持1，并且在传输完成之后会自动清0 读操作: 0 = 传输已经完成 1 = 传输还没有完成. **注:** 在写1到SPIMEN位之前，所有寄存器应该设置好。在传输过程中，不应该写该外设的任何寄存器。

有效检查寄存器 (SPIM_VALIDCTL)

寄存器	偏移地址	R/W	描述	复位值
SPIM_VALIDCTL	SPIM_BA+0x08	R/W	有效检查寄存器	0x0000_0000

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

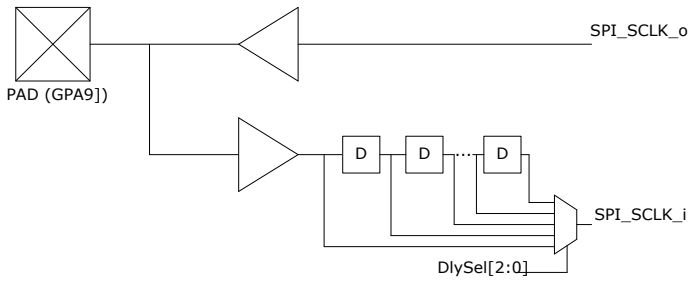
位	描述	
[31:17]	Reserved	保留.
[16]	VALIDEN	**有效使能位** 设置该位使能有效检查功能，该功能可以检查SPI Flash中的代码是否有效 0=>1 (上升沿) = 使能有效检查并清除 VALIDSTS 位. 1=>0 (下降沿) = 禁止有效检查并更新VALIDSTS 位.
[15:1]	Reserved	保留.
[0]	VALIDSTS	**有效状态位** 当VALIDEN位改变，该位会被更新。. 0 = SPI Flash 中的代码无效. 1 = SPI Flash 中的代码有效. **注:** 写 0 清除该位为 0.

Rx 时钟延迟控制寄存器(SPIM_RXCLKDLY)

Register	Offset	R/W	Description	Reset Value
SPIM_RXCLKDLY	SPIM_BA+0x0C	R/W	Rx 时钟延迟控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
DLYSEL			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							

Bits	Description	
[31:29]	DLYSEL	**Rx 采样时钟源延迟选择** 000 = 无延迟. 001 = 采样时钟延2个单元. 010 = 采样时钟延迟4个单元. 011 = 采样时钟延迟6个单元. … 111 = 采样时钟延迟14个单元
[28:0]	Reserved	保留.



Rx数据（来自SPI闪存）可能会有一些延迟。因此，我们使用DLYSEL[2:0]来调整采样时钟（SPI_SCLK_i）以存取正确的数据。

数据接收寄存器0~3 (SPIM_RX0~3)

寄存器	偏移地址	R/W	描述	复位值
SPIM_RX0	SPIM_BA+0x10	R	数据接收寄存器 0	0x0000_0000
SPIM_RX1	SPIM_BA+0x14	R	数据接收寄存器1	0x0000_0000
SPIM_RX2	SPIM_BA+0x18	R	数据接收寄存器2	0x0000_0000
SPIM_RX3	SPIM_BA+0x1C	R	数据接收寄存器3	0x0000_0000

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

位	描述	
[31:0]	**RX**	**数据接收寄存器** 数据接收寄存器保留最近一次已执行传输的接收到的数据。. 有效RX寄存器的个数在SPIM_CTL0[BURSTNUM].中指定，如果BURSTNUM > 0，接收到的数据先保存在最高有效RX寄存器中。 有效位的个数在SPIM_CTL0[DWIDTH].中指定，如果DWIDTH是16, 24, 或 32，接收到的数据先保存在RX寄存器的最低有效字节。 在一个字节中，接收到的数据先保存在RX寄存器的最高有效位。 **例 1:** 如果SPIM_CTL0[BURSTNUM] = 0x3 和 SPIM_CTL1[DWIDTH] = 0x17，接收到的数据会按顺序保存在SPIM_RX3[23:0], SPIM_RX2[23:0], SPIM_RX1[23:0], SPIM_RX0[23:0]. **例 2:** 如果 SPIM_CTL0[BURSTNUM = 0x0 和 SPIM_CTL0[DWIDTH] = 0x17, 接收到的数据会按顺序保存在 SPIM_RX0[7:0], SPIM_RX0[15:8], SPIM_RX0[23:16]. **例 3:** 如果 SPIM_CTL0[BURSTNUM = 0x0 和 SPIM_CTL0[DWIDTH] = 0x07, 接收到的数据会按顺序保存在SPIM_RX0[7], SPIM_RX0[6], …, SPIM_RX0[0].

数据发送寄存器 0~3 (SPIM_TX0~3)

寄存器	偏移地址	R/W	描述	复位值
SPIM_TX0	SPIM_BA+0x20	R/W	数据发送寄存器 0	0x0000_0000
SPIM_TX1	SPIM_BA+0x24	R/W	数据发送寄存器1	0x0000_0000
SPIM_TX2	SPIM_BA+0x28	R/W	数据发送寄存器2	0x0000_0000
SPIM_TX3	SPIM_BA+0x2C	R/W	数据发送寄存器3	0x0000_0000

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

位	描述	
[31:0]	**TX**	**数据发送寄存器** 数据发送寄存器保存下一次传输要发送的数据 有效TX寄存器的个数在SPIM_CTL0[BURSTNUM]中指定，如果BURSTNUM > 0，数据会先发送最高有效TX寄存器。 有效位的个数在SPIM_CTL0[DWIDTH].中指定。如果DWIDTH是16, 24, 或 32，会先发送TX寄存器的最低有效字节 在一个字节中，先发送TX寄存器的最高有效位 **例 1:** 如果 SPIM_CTL0[BURSTNUM] = 0x3 和 SPIM_CTL1[DWIDTH] = 0x17, 下次传输数据会按SPIM_TX3[23:0], SPIM_TX2[23:0], SPIM_TX1[23:0], SPIM_TX0[23:0] 顺序发送 **例 2:** 如果 SPIM_CTL0[BURSTNUM] = 0x0 和 SPIM_CTL0[DWIDTH] = 0x17, 下次传输数据会按SPIM_TX0[7:0], SPIM_TX0[15:8], SPIM_TX0[23:16] 顺序发送. **例 3:** 如果 SPIM_CTL0[BURSTNUM] = 0x0 和 SPIM_CTL0[DWIDTH] = 0x07, 下次传输数据会按SPIM_TX0[7], SPIM_TX0[6], …, SPIM_TX0[0] 顺序发送.

SRAM 内存地址寄存器 (SPIM_SRAMADDR)

寄存器	偏移地址	R/W	描述	复位值
SPIM_SRAMADDR	SPIM_BA+0x30	R/W	SRAM 内存地址寄存器	0x0000_0000

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

位	描述	
[31:0]	ADDR	**SRAM 内存地址** 对于 DMA 读模式,该寄存器是DMA传输的目的地址 对于 DMA 写模式, 该寄存器是DMA 传输的源地址. **注:**该地址必须是字对齐.

DMA 传输字节计数寄存器 (SPIM_DMATBCNT)

寄存器	偏移地址	R/W	描述	复位值
SPIM_DMATBCNT	SPIM_BA+0x34	R/W	DMA 传输字节计数寄存器	0x0000_0000

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

位	描述	
[31:24]	Reserved	保留.
[23:0]	CNT	DMA 传输字节计数寄存器 该位表示DMA传输的长度，计数单位是字节

SPI Flash 地址寄存器 (SPIM_FADDR)

寄存器	偏移地址	R/W	描述	复位值
SPIM_FADDR	SPIM_BA+0x38	R/W	SPI Flash 地址寄存器	0x0000_0000

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

位	描述	
[31:0]	ADDR	**SPI Flash 地址寄存器** 对于 DMA 读模式, 该寄存器是 DMA 传输的源地址. 对于 DMA 写模式, 该寄存器是DMA 传输的目的地址. **注:**该地址必须是字对齐.

6.14 带内部音频编解码的I²S 控制器 (I²S)

6.14.1 概述

该I²S控制器由到内部音频编解码或支持使用外部音频编解码的I2S协议接口组成。两个 16 字的 FIFO 分别用于接收和发送通道，可以处理8, 16, 24, 或 32位字大小的采样。

内部音频编解码的结构是一个delta-sigma的24位CODEC，带麦克风输入、线路输入、和耳机输出。

6.14.2 特性

I²S 控制器

- 支持主机模式和从机模式
- 可处理 8, 16, 24 和 32位字大小采样
- 支持单声道和立体声的音频数据
- 支持 I²S 和 MSB 对齐数据格式
- 支持 PCM-A 和 PCM-B 数据格式
- 提供两个 16 字的 FIFO 分别用于接收和发送
- 当缓存超过可编程边界时，产生中断请求
- 支持 TX DMA 功能用于发送和RX DMA 功能用于接收
- 支持 RX 数据功率测量
- 支持连接到外部音频编解码器

内部 CODEC

- 支持单声道麦克风输入和立体声音频线路输入
- 支持立体声耳机输出
- 支持立体声和单声道模式
- ADC特性
  - 总谐波失真与噪音 (THD+N): -80 dB
  - 动态范围 (DR) 和 信噪比 (SNR): 90 dB (A-Weighted)
- DAC特性 (带 32Ω 负载耳机输出)
  - 总谐波失真与噪音(THD+N): -60 dB
  - 动态范围 (DR) 和 信噪比 (SNR): 93 dB (A-Weighted)
- 支持采样率 8 kHz, 11.025 kHz, 12 kHz, 16 kHz, 22.05 kHz, 24 kHz, 32 kHz, 44.1 kHz, 48 kHz, 和 96 kHz

6.14.3 框图

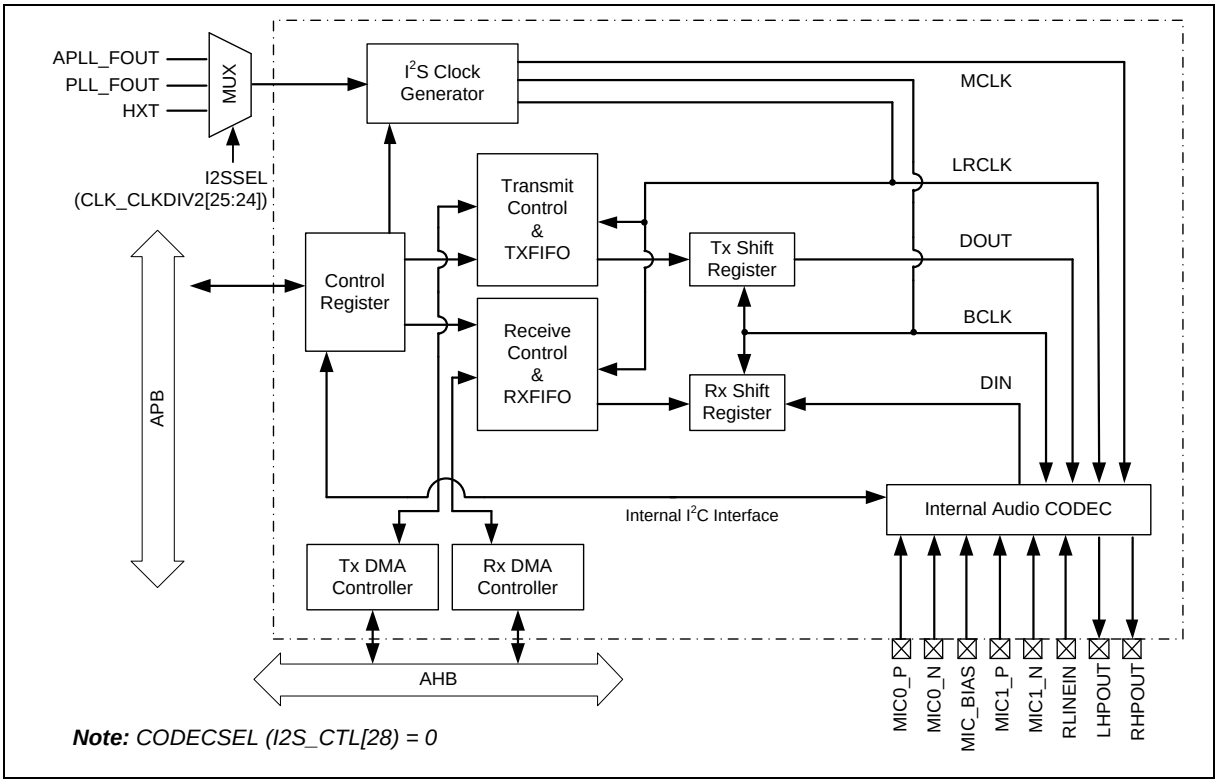


图 6.14-1 用内部音频编解码的I²S 控制器框图

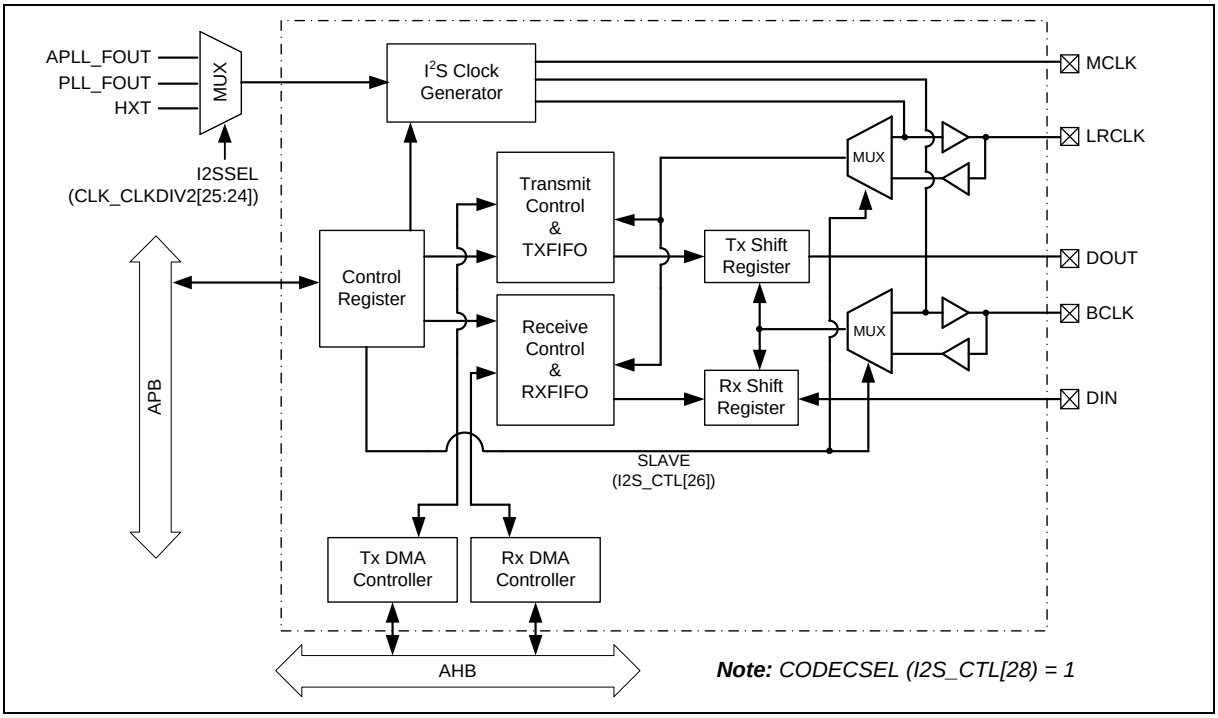


图 6.14-2用外部音频编解码的I²S 控制器框图

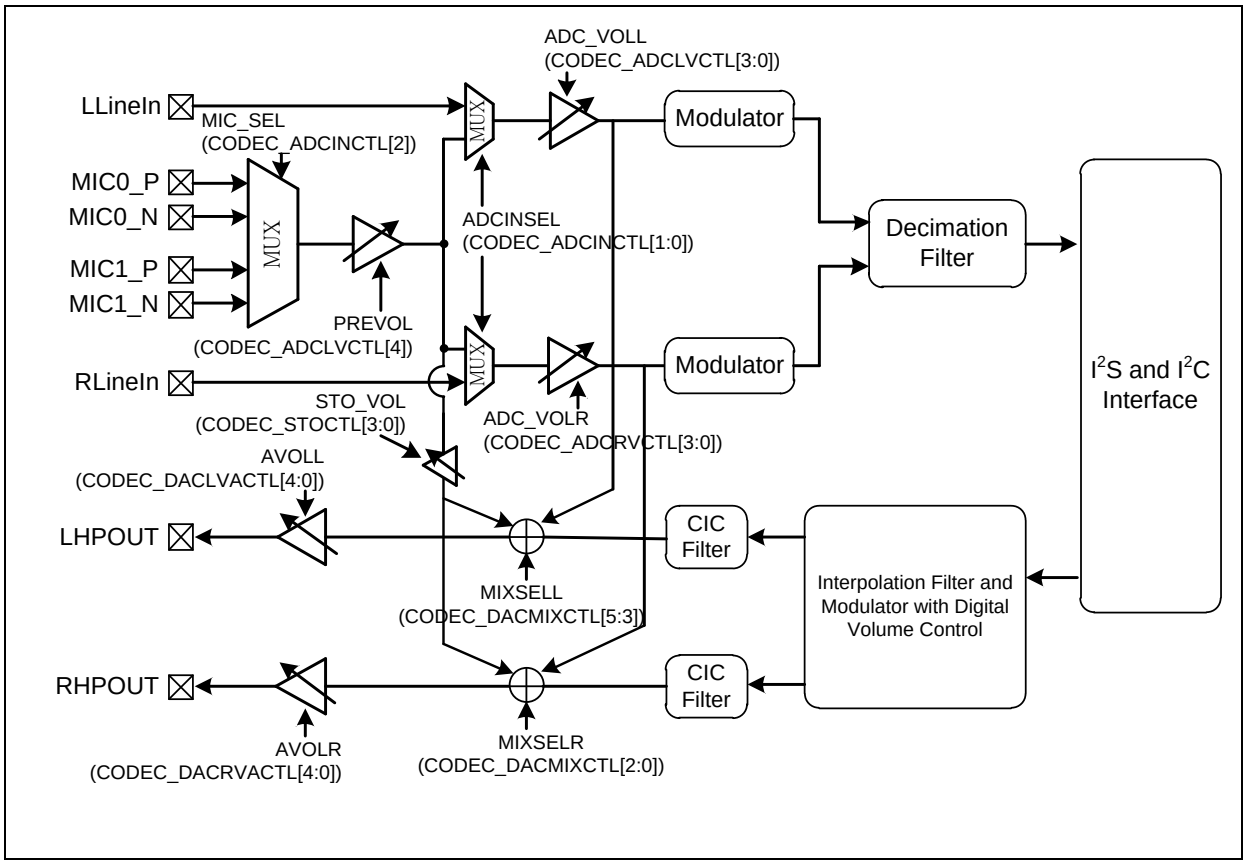
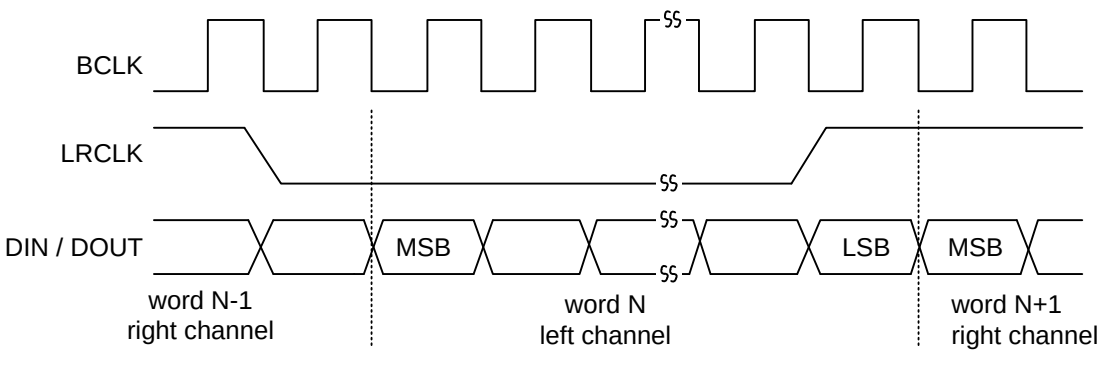


图 6.14-2 内部 CODEC 框图

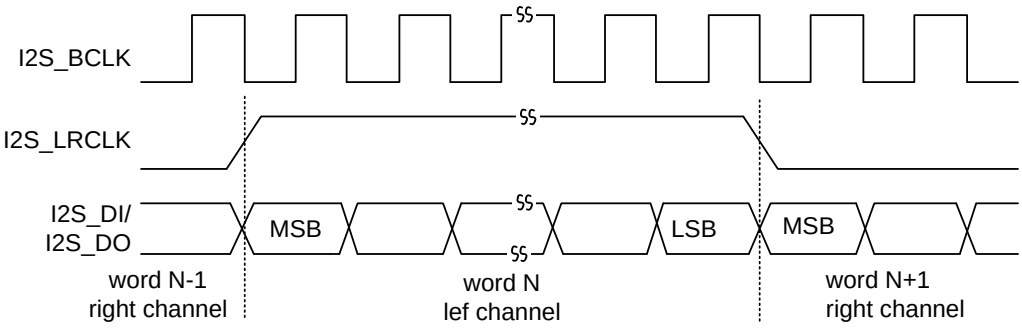
6.14.4 I²S 操作时序图描述

I²S控制器支持MSB对齐和I²S数据格式。LRCLK信号表示哪一个音频通道正在传输。音频通道的位计数由WDWIDTH (I2S_CTL[5:4])决定。传输顺序总是从最高有效位（MSB）开始。数据在上升时钟边沿读取，在下降时钟边沿发送。在I²S的数据格式，在音频通道的第一个时钟，MSB被发送并锁存。图 6.14-3表示了I²S数据格式时序图，图 6.14-4表示I²S MSB对齐的数据格式。



Note: PCMEN (I2S_CTL[24]) = 0 ; FORMAT (I2S_CTL[7]) = 0

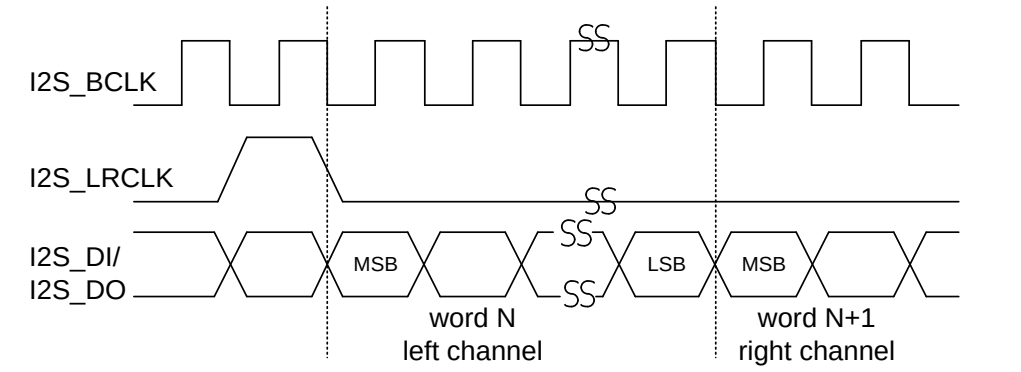
图 6.14-3 I²S 总线时序图



Note: PCMEN (I2S_CTL[24]) = 0 ; FORMAT (I2S_CTL[7]) = 1

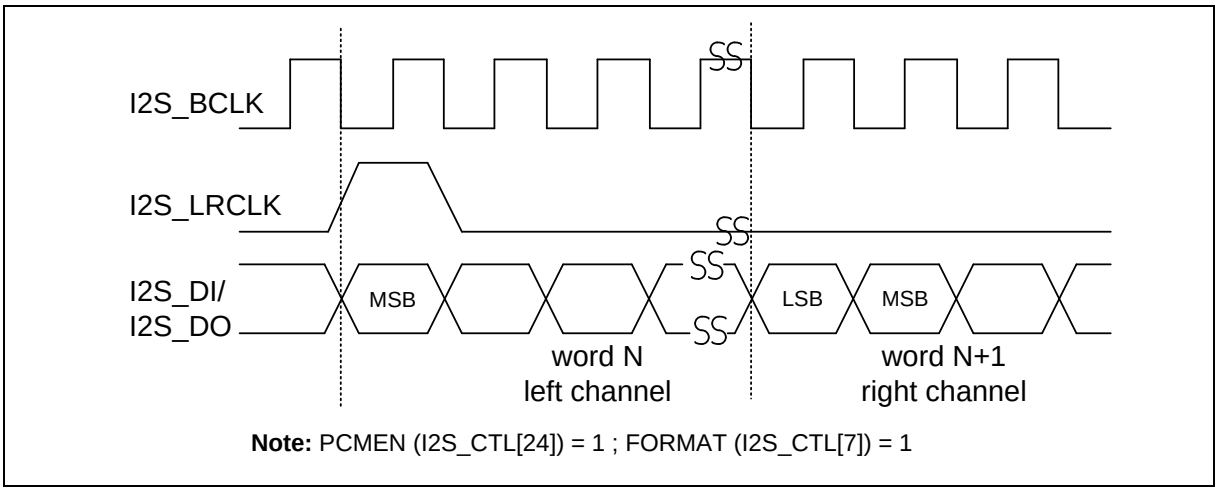
图 6.14-4 MSB 对齐时序图

I²S也支持PCM-A 和 PCM-B数据格式。LRCLK信号用一个时钟脉冲来表示新数据的开始。左通道和右通道的数据按顺序交替传输。音频通道的位计数由WDWIDTH (I2S_CTL[5:4])决定。在PCM-A数据格式，MSB在音频通道的第二个时钟被发送和锁存。PCM-B数据格式，MSB在音频通道的第一个时钟被发送和锁存。图 6.14-5 和 图 6.14-6 表示PCM-A 和 PCM-B数据格式的时序图。



Note: PCMEN (I2S_CTL[24]) = 1 ; FORMAT (I2S_CTL[7]) = 0

图 6.14-5 PCM-A 音频时序图



Note: PCMEN (I2S_CTL[24]) = 1 ; FORMAT (I2S_CTL[7]) = 1

图 6.14-6 PCM-B音频时序图

6.14.5 FIFO 操作

音频通道的字宽度可以是8, 16, 24 或 32 位。各种设置的内存排列如下所示：

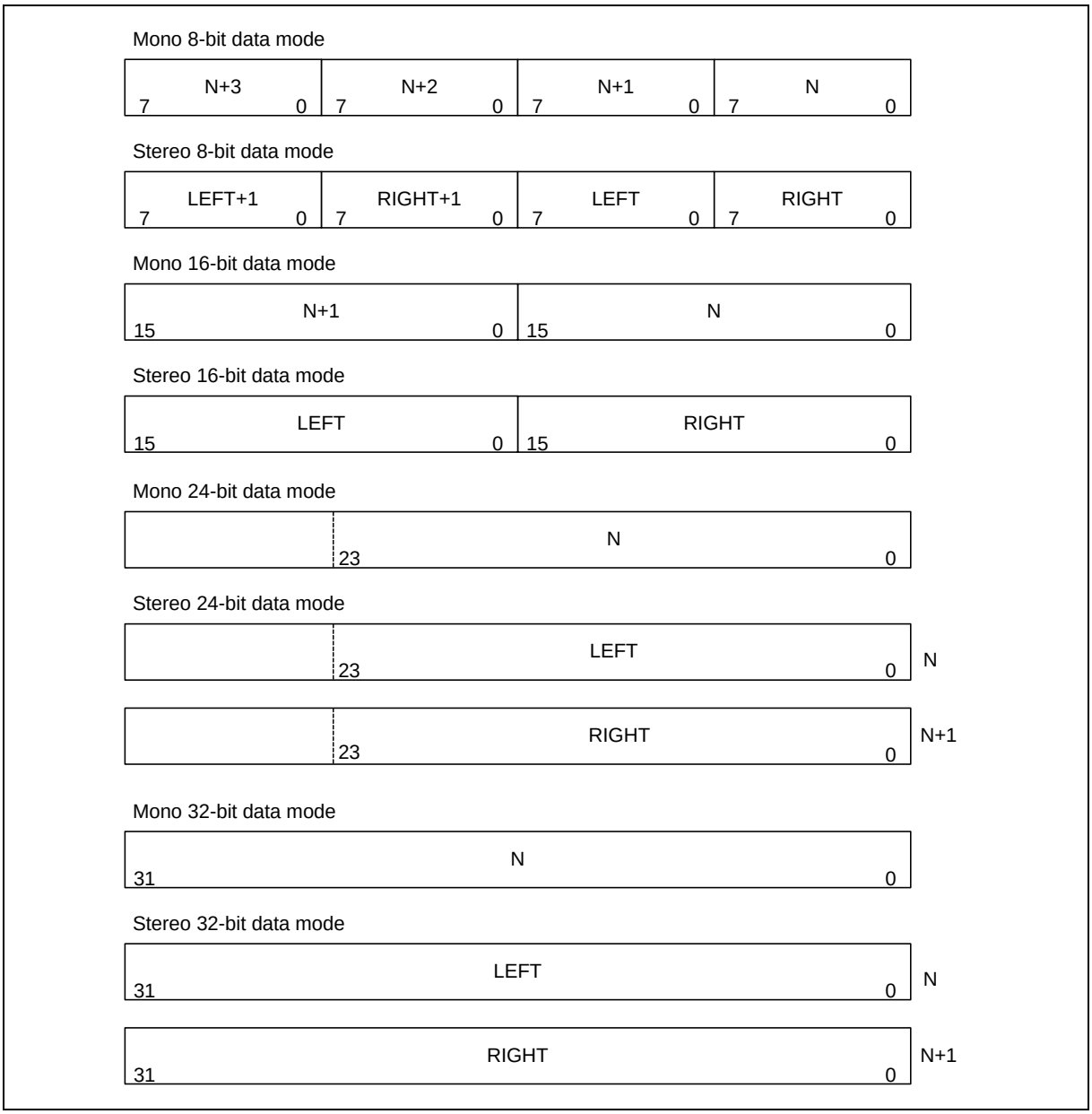


图 6.14-7不同I²S 模式下FIFO 的内容

6.14.6 功能描述

6.14.6.1 过零检测

当通过I²S功能播放声音，来自内存的数据通过CPU或DMA输出。然而，如果播放增益水平由用户在任意时刻改变，有可能会导致一些POP的噪音。因为如果输出数据不是0，输出数据经过增益的

改变点会产生一个不连续的数据。这个不连续的数据会引起POP的噪音。所以过零检测的功能可以帮助减少这种情况。如果过零检测功能使能，硬件会侦测下一个传输的数据是否是0或符号改变。如果下一个数据是0或者符号改变，过零标志LZCIF (I2S_STATUS[23]) 和RZCIF (I2S_STATUS[22])会置1 。然后下一个传输数据会被保持并自动输出数据0，直到该标志被用户清除。所以如果用户想修改播放增益，用户可以使能过零检测中断功能，LZCIEN (I2S_IEN[12]) 和RZCIEN (I2S_IEN[11]), 来指示过零检测的时间，并改变声音增益。这会减少POP的噪音。

6.14.6.2 DMA 模式

I²S功能可以使用DMA功能来访问数据。如果使能TX DMA功能，当发送FIFO没满，I²S会产生请求信号并自动从内存获取数据，直到发送FIFO满。而如果使能RX DMA功能，当接收FIFO不为空，I²S会产生请求信号，并自动搬运接收到的数据到内存，直到接收FIFO为空。所以当录音或放音的时候，用户可以使用TX DMA 和 RX DMA来分担CPU的负担。

例-1: 使用 TX DMA来播放声音，且声音数据在内存的地址从0x2001_0300 到 0x2001_03FC。

1. 根据声音格式，设置正确的值到 MONO (I2S_CTL[6])，WDWIDTH (I2S_CTL[5:4]), FORMAT (I2S_CTL[7])。
2. 设置 I2S_TXSTADDR 值为 0x2001_0300 (该寄存器要字对齐).
3. 设置 I2S_TXEADDR 值为 0x2001_03FC (该寄存器要字对齐).
4. 设置 TXDMAEN (I2S_CTL[20]) 为 1.
5. 设置 TXEN (I2S_CTL[1]) 和 I2SEN (I2S_CTL[0]) 为 1.
6. I²S 控制器会使用 TX DMA 来重复播放从地址 0x2001_0300 到 0x2001_03FC 的数据

例-2:使用 RX DMA来录音，且声音数据在内存的地址从0x2001_0300 到 0x2001_03FC

1. 根据声音格式，设置正确的值到 MONO (I2S_CTL[6]), WDWIDTH (I2S_CTL[5:4]), FORMAT (I2S_CTL[7]) 。
2. 设置 I2S_ RXSTADDR 值为 0x2001_0300 (该寄存器要字对齐).
3. 设置 I2S_ RXEADDR 值为 0x2001_03FC (该寄存器要字对齐).
4. 设置 RXDMAEN (I2S_CTL[21]) 为 1.
5. 设置 RXEN (I2S_CTL[2]) 和 I2SEN (I2S_CTL[0]) 为 1.
6. I²S 控制器会使用 RX DMA 来重复将录音数据存到地址 0x2001_0300 到 0x2001_03FC 的内存空间。

6.14.6.3 主/从 接口

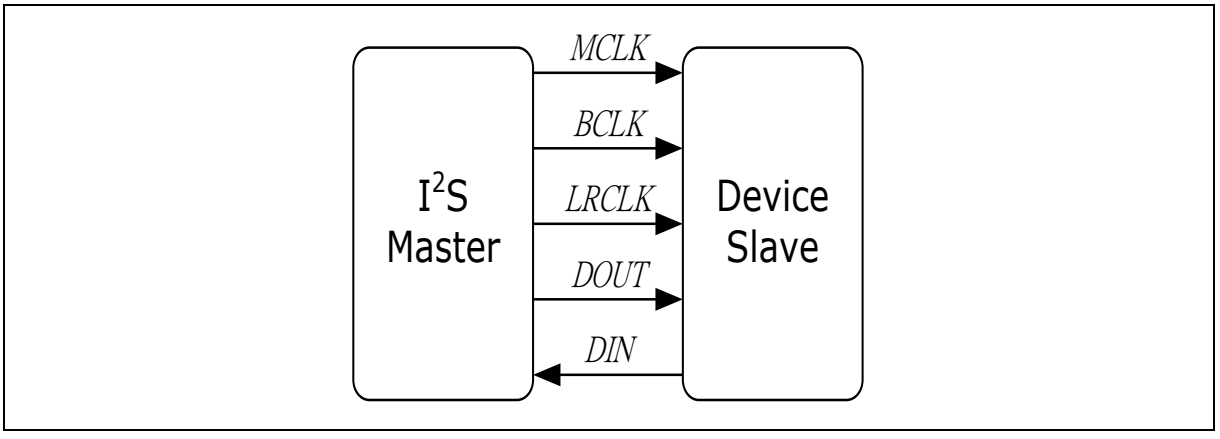


图 6.14-8 主机模式接口框图

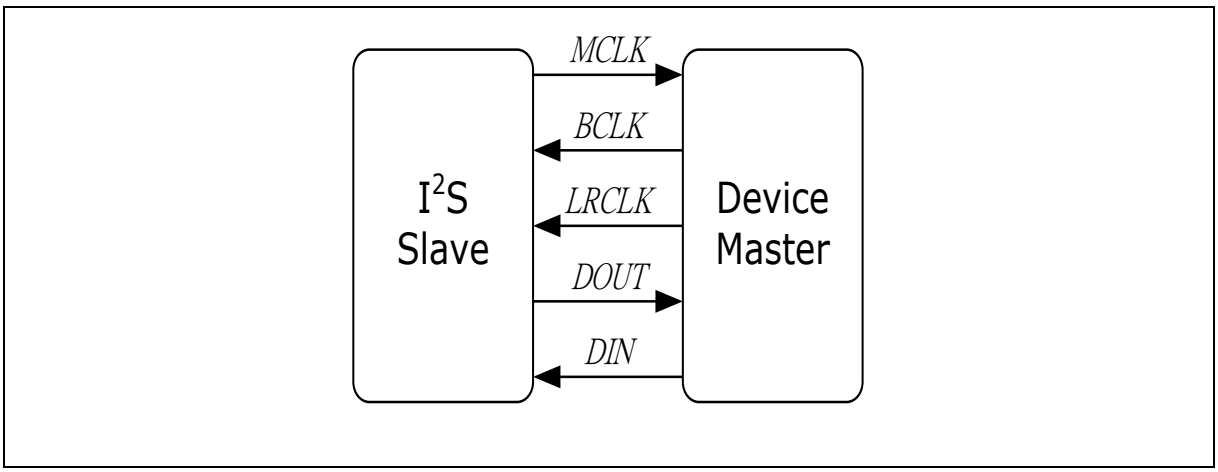


图 6.14-9 从机模式接口框图

在主机模式，MCLK, BCLK, LRCLK输出到从机设备。然而在从机模式，MCLK是输出到主机设备，BCLK 或 LRCLK是从主机设备输入。

6.14.6.4 内部音频CODEC 接口

NUC505 有一个内部音频 CODEC接口。内部音频CODEC 的控制接口是I^2C 协议，音频接口是I^2S协议(参考图 6.14-1)。所以I^2S控制器有一个虚拟的I^2C转换器模块来转换APB命令为I^2C协议命令，用以容易地读或写内部音频CODEC寄存器。下面是写和读内部音频CODEC寄存器的例子。

例 1: 写内部音频CODEC 寄存器.

如果你想改变左通道模拟增益，AVOLL (CODEC_DACLVACTL[4:0]) 为静音， (对于内部音频CODEC 寄存器，参考6.15.9 和 6.15.10部分)

1. 设置 I2SCKEN (CLK_APBCLK[14]) 为 1 来打开 I^2S 时钟.
2. 设置 I2S_CODECCTL 为 0x2081_1081。这意味着:
   a. I^2C 协议的 SCLK (F_SCLK) 的频率等于 F_I2SCLK / (32*16)。这里的 F_I2SCLK 的意思是 I^2S 外设时钟的频率。32 的值来自 I2CCKDIV (I2S_CODECCTL[30:24])。16 是固定的值。
   b. 用户想写内部音频 CODEC 寄存器，RW (I2S_CODECTL[16]) 需要等于 1。
   c. 内部音频 CODEC 地址是 0x08 且数据是 0x1F (静音).
3. 等待 BUSY (I2S_CODECCTL[31]) 变为 0 且 I^2C 命令已完成。

注1: "F_" 是什么的频率的意思.

注2: F_SCLK 应该比F_MCLK 低32倍.

注3: 对于内部音频CODEC 寄存器地址，参考6.15.9部分.

例 2: 读内部音频CODEC 寄存器.

如果你想读左通道模拟增益AVOLL (CODEC_DACLVACTL[4:0]).

1. 设置 I2SCKEN (CLK_APBCLK[14]) 为 1 来打开 I^2S 时钟.
2. 设置 I2S_CODECCTL 为 0x2080_0800， 这意味着 :
   a. I^2C 协议的 SCLK (F_SCLK) 的频率等于 F_I2SCLK / (32*16)。这里的 F_I2SCLK 的意思是 I^2S 外设时钟的频率。32 的值来自 I2CCKDIV (I2S_CODECCTL[30:24])。16 是固定的值。
   b. 用户想读内部音频 CODEC 寄存器，RW (I2S_CODECTL[16]) 需要等于 0。
   c. 内部音频 CODEC 地址是 0x08.
3. 等待 BUSY (I2S_CODECCTL[31]) 变为 0.
4. 读 DAT (I2S_CODECCTL[7:0]), 该值是左通道的模拟增益。

注1: ”F_”是什么的频率的意思.

注2: F_SCLK 应该比F_MCLK 低32倍 .

注3: 对于内部音频CODEC 寄存器地址，参考6.15.9部分.

6.14.6.5 MCLK, BCLK 和 LRCLK 的产生

I^2S 在主机模式:

如果内部音频CODEC (CODECSEL (I2S_CTL[28]) = 0)被使用，MCLK (F_MCLK)的频率必须比音频采样率(F_LRCLK)大256倍。BCLK的频率必须比F_LRCLK大64倍。64是由2*32计算来的，2的意思是立体声通道，32的意思是一个通道音频采样有32个BCLK时钟。

但是如果外部音频CODEC (CODECSEL (I2S_CTL[28] = 1)被使用，F_MCLK可以比F_LRCLK大256/128/64倍。这取决于外部音频编解码器的要求。F_BCLK必须低于F_MCLK，F_BCLK可以比F_LRCLK大2*8 / 2*16 / 2*24 / 2*32倍。

WDWIDTH = 0x0: 一个通道的音频采样有8个BCLK 周期.
WDWIDTH = 0x1: 一个通道的音频采样有16个BCLK 周期.
WDWIDTH = 0x2: 一个通道的音频采样有24个BCLK 周期.
WDWIDTH = 0x3: 一个通道的音频采样有32个BCLK 周期.

所以接下来的步骤可以用于设置BCLKDIV (I2S_CLKDIV[16:8]) 和 MCLKDIV (I2S_CLKDIV[5:0]),

内部音频 CODEC (CODECSEL = 0):

1. 选择音频采样率 (F_LRCLK)
2. F_MCLK = 256 * F_LRCLK.
3. F_BCLK = 64 * F_LRCLK.
4. 设置 BCLKDIV (I2S_CLKDIV[16:8]) 和 MCLKDIV (I2S_CLKDIV[5:0]) 来获得正确的 F_BCLK 和 F_MCLK.

$$F_MCLK = F_I2SCLK / (2*MCLKDIV) \quad (如果\ MCLKDIV \neq 0)$$
$$= F_I2SCLK \quad (如果\ MCLKDIV = 0)$$
$$F_BCLK = F_I2SCLK / (2*(BCLKDIV+1))$$

外部音频 CODEC (CODECSEL = 1):

1. 选择音频采样率 (F_LRCLK)

2. F_MCLK = 384 (或 256 或 128 或 64) * F_LRCLK. (该值取决于外部音频 CODEC 的要求，且必须大于 F_BCLK)
3. F_BCLK = 2 * 8 * F_LRCLK (如果 WDWIDTH = 0x0)
   = 2 * 16 * F_LRCLK (如果 WDWIDTH = 0x1)
   = 2 * 24 * F_LRCLK (如果 WDWIDTH = 0x2)
   = 2 * 32 * F_LRCLK (如果 WDWIDTH = 0x3)
4. 设置 BCLKDIV (I2S_CLKDIV[16:8]) 和 MCLKDIV (I2S_CLKDIV[5:0]) 来获得正确的 F_BCLK 和 F_MCLK.

   *F_MCLK = F_I2SCLK / (2*MCLKDIV) (如果 MCLKDIV ≠ 0)*
   = F_I2SCLK (如果 MCLKDIV = 0)
   F_BCLK = F_I2SCLK / (2(BCLKDIV+1))*

内部音频 CODEC: CODECSEL = 0 ; I2SSEL (CLK_CLKDIV2[25:24]) = 0x3;							
F_LRCLK (KHz)	WDWIDTH [1:0]	F_MCLK (MHz)	F_BCLK (MHz)	BCLKDIV [8:0]	MCLKDIV [5:0]	I2SDIV [7:0]	CLK_APLLCTL [31:0]
96	0x0	24.576	6.144	0x3	0x1	0x0	0x424D542
	0x1						
	0x2						
	0x3						
48	0x0	12.288	3.072	0x7	0x2	0x0	0x424D542
	0x1						
	0x2						
	0x3						
44.1	0x0	11.2896	2.8224	0x7	0x2	0x0	0x704D382
	0x1						
	0x2						
	0x3						
32	0x0	8.192	2.048	0xB	0x3	0x0	0x424D542
	0x1						
	0x2						
	0x3						
16	0x0	4.096	1.024	0x17	0x6	0x0	0x424D542
	0x1						
	0x2						
	0x3						
8	0x0	2.048	0.512	0x2F	0xC	0x0	0x424D542

	0x1						
	0x2						
	0x3						

表 6.14-1 内部音频CODEC 的时钟参考

外部音频 CODEC : CODECSEL = 1 ; I2SSEL (CLK_CLKDIV2[25:24]) = 0x3; 注: 设置 F_MCLK = 256 * F_LRCLK							
F_LRCLK (KHz)	WDWIDTH [1:0]	F_MCLK (MHz)	F_BCLK (MHz)	BCLKDIV [8:0]	MCLKDIV [5:0]	I2SDIV [7:0]	CLK_APLLCTL [31:0]
96	0x0	24.576	1.536	0xF	0x1	0x0	0x424D542
	0x1		3.072	0x7	0x1		
	0x3		6.144	0x3	0x1		
48	0x0	12.288	0.768	0x1F	0x2	0x0	0x424D542
	0x1		1.536	0xF	0x2		
	0x3		3.072	0x7	0x2		
44.1	0x0	11.2896	0.7056	0x1F	0x2	0x0	0x704D382
	0x1		1.4112	0xF	0x2		
	0x3		2.8224	0x7	0x2		
32	0x0	8.192	0.512	0x2F	0x3	0x0	0x424D542
	0x1		1.024	0x17	0x3		
	0x3		2.048	0xB	0x3		
16	0x0	4.096	0.256	0x5F	0x6	0x0	0x424D542
	0x1		0.512	0x2F	0x6		
	0x3		1.024	0x17	0x6		
8	0x0	2.048	0.128	0xBF	0xC	0x0	0x424D542
	0x1		0.256	0x5F	0xC		
	0x3		0.512	0x2F	0xC		

表 6.14-2 外部音频CODEC 的时钟参考

注: 当使用外部音频CODEC 且WDWIDTH (I2S_CTL[5:4]) = 0x2, 这比I²S 控制器工作在从机模式好。

I²S 在从机模式:

当内部音频CODEC (CODECSEL = 0)被使用，I²S不支持从机模式。如果外部音频CODEC (CODECSEL = 1)被使用，BCLK 和 LRCLK从外部音频CODEC输入。因此F_BCLK 和 F_LRCLK由外部音频CODEC控制。但是F_MCLK仍然是输出信号，并取决于外部音频CODEC的要求。

F_MCLK = F_I2SCLK / (2*MCLKDIV)　(如果 MCLKDIV ≠ 0)

= F_I2SCLK　(如果 MCLKDIV = 0)

6.14.6.6 RX 数据功率测量

I²S控制器支持RX数据功率测量功能。这可以帮助用户很容易地控制输入音频信号的增益。用户可以通

过I2S_RXLCHAVG和I2S_RXRCHAVG寄存器获得功率的平均值 。这样就可以知道输入音频信号是多大的音量。然后用户就可以控制音量增益来调整音频信号到目标水平。

功率测量的方法是在设定的窗口周期内计算RX数据的绝对值。因此需要先设置WINSEL (I2S_RXAVGCTL[3:0])寄存器。

例:

如果WINSEL (I2S_RXAVGCTL[3:0]) = 0x2,意味着平均窗口是4个采样。因此每四个输入音频信号会产生一个平均值。换句话说测量结果代表四个采样的平均功率。

6.14.6.7 中断源

图 6.14-10 表示I^{2}S 控制器的中断源。每一个中断源可以单独使能。

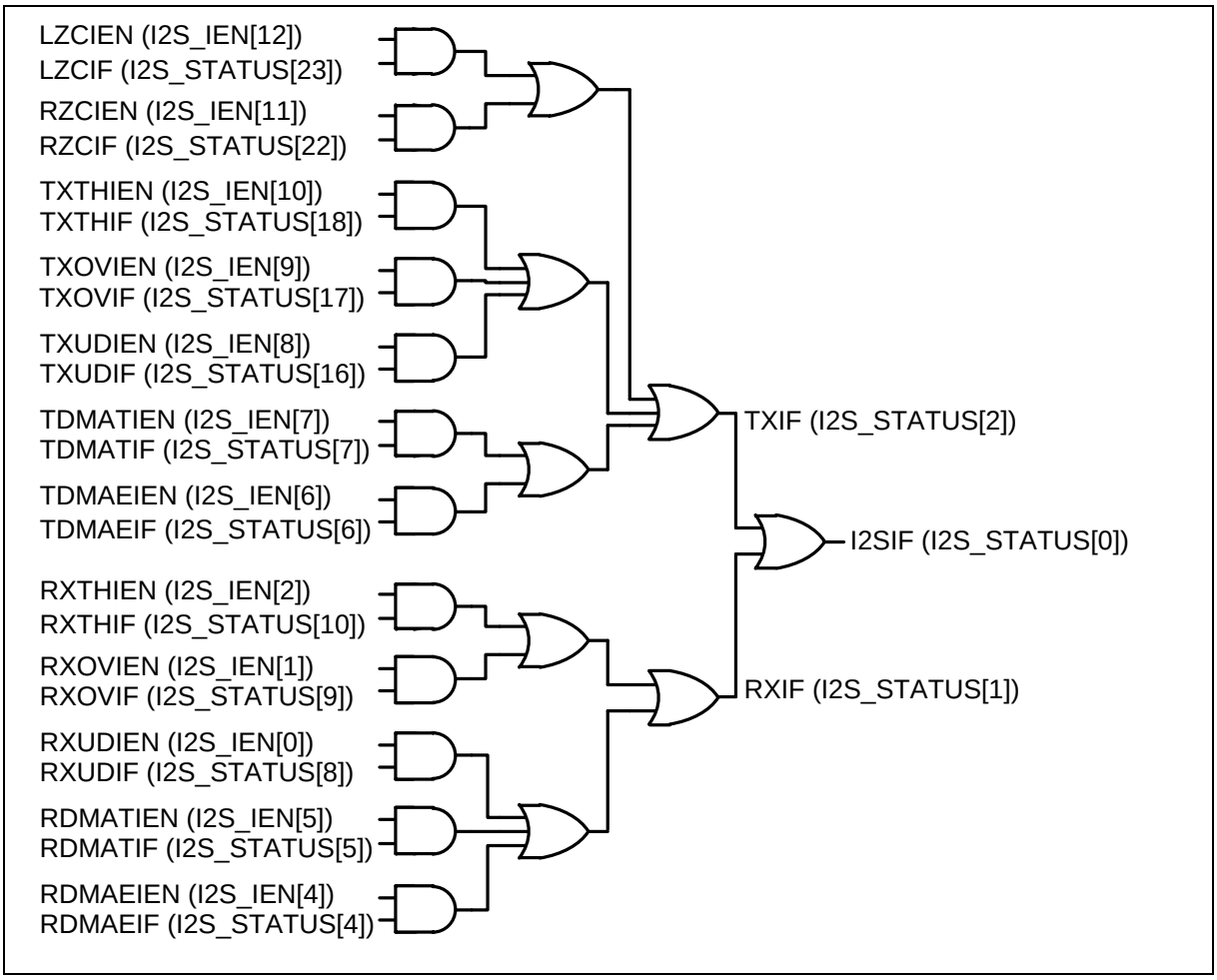


图 6.14-10 I^{2}S 控制器中断

6.14.7 寄存器映射

R: 只读, **W**: 只写, **R/W**: 读写

寄存器	偏移地址	R/W	描述	复位值
I²S 基地址: **I2S_BA = 0x400E_8000**				
I2S_CTL	I2S_BA+0x00	R/W	I²S 控制寄存器	0x2000_0000
I2S_CLKDIV	I2S_BA+0x04	R/W	I²S 时钟分频器控制寄存器	0x0000_0000
I2S_IEN	I2S_BA+0x08	R/W	I²S 中断使能寄存器	0x0000_0000
I2S_STATUS	I2S_BA+0x0C	R/W	I²S 状态寄存器	0x0014_1000
I2S_TX	I2S_BA+0x10	W	I²S 发送 FIFO 寄存器	0x0000_0000
I2S_RX	I2S_BA+0x14	R	I²S 接收 FIFO 寄存器	0x0000_0000
I2S_CODECCTL	I2S_BA+0x18	R/W	I²S 虚拟 I²C 控制寄存器	0x0000_0000
I2S_TXSTADDR	I2S_BA+0x20	R/W	I²S TX DMA 开始地址寄存器	0x2001_0000
I2S_TXTHADDR	I2S_BA+0x24	R/W	I²S TX DMA 阀值地址寄存器	0x2001_FFF0
I2S_TXEADDR	I2S_BA+0x28	R/W	I²S TX DMA 结束地址寄存器	0x2001_FFF0
I2S_TXCADDR	I2S_BA+0x2C	R	I²S TX DMA 当前地址寄存器	0x0000_0000
I2S_RXSTADDR	I2S_BA+0x30	R/W	I²S RX DMA 开始地址寄存器	0x2001_0000
I2S_RXTHADDR	I2S_BA+0x34	R/W	I²S RX DMA阀值地址寄存器	0x2001_FFF0
I2S_RXEADDR	I2S_BA+0x38	R/W	I²S RX DMA结束地址寄存器	0x2001_FFF0
I2S_RXCADDR	I2S_BA+0x3C	R	I²S RX DMA当前地址寄存器	0x0000_0000
I2S_RXAVGCTL	I2S_BA+0x40	R/W	I²S RX 数据平均值控制寄存器	0x0000_000A
I2S_RXLCHAVG	I2S_BA+0x44	R	I²S RX 左通道数据平均值	0x0000_0000
I2S_RXRCHAVG	I2S_BA+0x48	R	I²S RX右通道数据平均值	0x0000_0000

6.14.8 寄存器描述

I²S 控制寄存器 (I2S_CTL)

寄存器	偏移地址	R/W	描述	复位值
I2S_CTL	I2S_BA+0x00	R/W	I²S 控制寄存器	0x2000_0000

31	30	29	28	27	26	25	24
Reserved		CODECRST	CODECSEL	Reserved	SLAVE	MCLKEN	PCMEN
23	22	21	20	19	18	17	16
RXLCH	Reserved	RXDMAEN	TXDMAEN	RXCLR	TXCLR	LZCEN	RZCEN
15	14	13	12	11	10	9	8
RXTH				TXTH			
7	6	5	4	3	2	1	0
FORMAT	MONO	WDWIDTH		MUTE	RXEN	TXEN	I2SEN

位	描述	
[31:30]	**Reserved**	保留.
[29]	**CODECRST**	**内部 Codec 硬件复位控制** 0 = 复位操作. 1 = 正常操作.
[28]	**CODECSEL**	**内部 Codec 或外部 Codec 选择** 0 = I²S接口连接到内部codec. 1 = I²S 接口连接到外部codec.
[27]	**Reserved**	保留.
[26]	**SLAVE**	**从机选择** I²S 可以工作为主机或从机，对主机模式,I2S_BCLK 和 I2S_LRCLK 信号输出到CODEC。在从机模式， I2S_BCLK 和 I2S_LRCLK 引脚从 CODEC接收信号。. 0 = 主机模式. 1 = 从机模式. **注:** 如果使用内部codec，I²S必须是主机模式
[25]	**MCLKEN**	**主机时钟使能控制** 0 = I²S 主机时钟输出禁止 1 = I²S 主机时钟输出使能 **注1:** I2S_MCLK 总是输出. **注2:** I2S_MCLK 频率由MCLKDIV[5:0].控制

[24]	**PCMEN**	**PCM 接口使能控制** 0 = I^2S 接口. 1 = PCM 接口.
[23]	**RXLCH**	**接收左通道使能控制** 当选择单声道模式(MONO = 1), 如果RXLCH被设置为0，则I2S 将接收右声道数据, 而如果RXLCH被设置为1，I2S接收左声道数据. 0 = 当选择单声道模式时，接收右声道数据. 1 = 当选择单声道模式时，接收左声道数据.
[22]	**Reserved**	保留.
[21]	**RXDMAEN**	**RX DMA 使能控制 (录音路径)** 0 = RX DMA 模式禁止. 1 = RX DMA 模式使能. **注:** 仅当RXDMAEN从低到高， I2S_RXSTADDR会更新新的配置。因此如果你想改变I2S_RXSTADDR，应该确认RXDMAEN是禁止的。
[20]	**TXDMAEN**	**TX DMA 使能控制 (发送路径)** 0 = TX DMA 模式禁止. 1 = TX DMA 模式使能. **注:** 仅当TXDMAEN从低到高， I2S_TXSTADDR会更新新的配置。因此如果你想改变I2S_TXSTADDR，应该确认TXDMAEN是禁止的。
[19]	**RXCLR**	**清除接收 FIFO** 0 = 无影响. 1 = 接收 FIFO 被清除. **注1:** 该位自动清零 **注2:** 如果清除接收FIFO ，RXCNT (I2S_STATUS[27:24])返回0，接收 FIFO 变为空.
[18]	**TXCLR**	**清除发送FIFO** 0 = 无影响. 1 = 发送 FIFO 被清除. **注1:** 该位自动清零 **注2:** 如果清除发送FIFO ，TXCNT (I2S_STATUS[31:28])返回0，发送 FIFO 变为空.
[17]	**LZCEN**	**使能左声道过零检测** 如果该位置 1，当左声道数据符号位改变或下一个移位数据的数据位都为0，则寄存器 I2S_STATUS 的LZCIF标志被置为 1。 0 = 禁用左声道过零检测 1 = 使能左声道过零检测
[16]	**RZCEN**	**使能右声道过零检测** 如果该位置 1，当右声道数据符号位改变或下一个移位数据的数据位都为0，则寄存器 I2S_STATUS 的RZCIF标志被置为 1。 0 = 禁用右声道过零检测 1 = 使能右声道过零检测

[15:12]	**RXTH**	**接收 FIFO 阈值水平** 当缓存中的接收字等于或大于阈值水平时，RXTHIF 标志置位. 0000 =接收 FIFO 中有 1个 字数据 0001 =接收 FIFO 中有 2个 字数据 0010 =接收 FIFO 中有 3个 字数据 0011 =接收 FIFO 中有 4个 字数据. 0100 =接收 FIFO 中有 5个 字数据. 0101 =接收 FIFO 中有 6个 字数据. 0110 =接收 FIFO 中有 7个 字数据. 0111 =接收 FIFO 中有 8个 字数据. 1000 =接收 FIFO 中有 9个 字数据. 1001 =接收 FIFO 中有 10个 字数据. 1010 =接收 FIFO 中有 11个 字数据. 1011 =接收 FIFO 中有 12个 字数据. 1100 =接收 FIFO 中有 13个 字数据. 1101 =接收 FIFO 中有 14个 字数据. 1110 =接收 FIFO 中有 15个 字数据 1111 =接收 FIFO 中有 16个 字数据.
[11:8]	**TXTH**	**发送 FIFO 阈值水平** 如果发送 FIFO 中剩下的数据字 (32 位) 等于或小于阈值水平，则TXTHIF标志置位. 0000 =发送 FIFO 中有 0个字数据 0001 =发送 FIFO 中有 1 个字数据. 0010 =发送 FIFO 中有 2 个字数据 0011 =发送 FIFO 中有 3 个字数据. 0100 =发送 FIFO 中有 4 个字数据. 0101 =发送 FIFO 中有 5 个字数据. 0110 =发送 FIFO 中有 6个字数据. 0111 =发送 FIFO 中有 7个字数据 1000 =发送 FIFO 中有 8 个字数据. 1001 =发送 FIFO 中有 9个字数据. 1010 =发送 FIFO 中有 10个字数据. 1011 =发送 FIFO 中有 11 个字数据. 1100 =发送 FIFO 中有 12 个字数据. 1101 =发送 FIFO 中有 13 个字数据. 1110 =发送 FIFO 中有 14 个字数据. 1111 =发送 FIFO 中有 15个字数据.

[7]	**FORMAT**	**数据格式选择** 如果 PCMEN=0,. 0 = I²S 数据格式. 1 = MSB 对齐数据格式. 如果PCMEN=1,. 0 = PCM 模式 A. 1 = PCM模式 B.
[6]	**MONO**	**单声道数据** 0 = 数据为立体声格式 1 = 数据为单声道格式， **注:**当芯片录音数据，且MONO = 1，RXLCH (I2S_CTL[23]) 会控制录音右声道数据或左声道数据的选择 。
[5:4]	**WDWIDTH**	**字宽度** 00 = 数据为 8-bit. 01 = 数据为16-bit. 10 = 数据为24-bit. 11 = 数据为32-bit.
[3]	**MUTE**	**使能发送静音** 0 = 发送缓存中的数据到发送通道. 1 = 发送数据 固定为0
[2]	**RXEN**	**接收使能** 0 = 禁用数据接收. 1 = 使能数据接收.
[1]	**TXEN**	**发送使能** 0 = 禁用数据发送. 1 = 使能数据发送.
[0]	**I2SEN**	**使能 I²S 控制器** 0 = 禁用. 1 = 使能.

I2S时钟分频寄存器(I2S_CLKDIV)

寄存器	偏移地址	R/W	描述	复位值
I2S_CLKDIV	I2S_BA+0x04	R/W	I^2S 时钟分频控制寄存器	0x0000_0000

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

位	描述	
[31:17]	Reserved	保留.
[16:8]	BCLKDIV	位时钟分频 如果 I^2S 工作于主机模式，用户客户编程这些位来产生BCLK的频率。在从机模式，BCLK的频率由主设备控制。 F_BCLK = F_I2SCLK / (2*(BCLKDIV+1)).
[7:6]	Reserved	保留.
[5:0]	MCLKDIV	主时钟分频 如果F_I2SCLK 是 (2*MCLKDIV)*256*F_LRCLK ， 软件可以编程这些位来产生256*F_LRCLK的时钟频率作为主机时钟输出到音频codec 。但是如果MCLKDIV被置为0. MCLK和I2SCLK输入一样。 例如，如果采样率是24 kHz kHz (F_LRCLK = 24 kHz) 并且 F_I2SCLK 是 12.288 MHz, MCLKDIV应该被置1来得到256*F_LRCLK的频率(12.288 MHz= (2*1*256*24) kHz). F_MCLK = F_I2SCLK / (2*MCLKDIV) (当 MCLKDIV >= 1). F_MCLK = F_I2SCLK (当 MCLKDIV = 0).

I²S 中断使能 (I2S_IEN)

寄存器	偏移地址	R/W	描述	复位值
I2S_IEN	I2S_BA+0x08	R/W	I²S 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			LZCIEN	RZCIEN	TXTHIEN	TXOVIEN	TXUDIEN
7	6	5	4	3	2	1	0
TDMATIEN	TDMAEIEN	RDMATIEN	RDMAEIEN	Reserved	RXTHIEN	RXOVIEN	RXUDIEN

位	描述	
[31:13]	Reserved	保留.
[12]	LZCIEN	**左声道过零检测中断使能** 如果该位设为 1 且左声道有过零信号，则中断产生. 0 = 禁用中断 1 = 使能中断
[11]	RZCIEN	**右声道过零检测中断使能** 如果该位设为 1 且右声道有过零信号，则中断产生. 0 = 禁用中断 1 = 使能中断
[10]	TXTHIEN	**发送 FIFO 阈值水平中断使能** 如果该位被设为1，且发送 FIFO 中的数据字小于或等于TXTH[3:0].，则中断发生. 0 = 禁用中断. 1 = 使能中断.
[9]	TXOVIEN	**发送 FIFO 溢出中断使能** 如果该位被设为1，且发送 FIFO 的溢出标志被置为 1，则中断发生 0 = 禁用中断. 1 = 使能中断.
[8]	TXUDIEN	**发送 FIFO 下溢中断使能** 如果该位被设为1，且发送 FIFO 的下溢标志被置为 1，则中断发生 0 = 禁用中断. 1 = 使能中断.

[7]	**TDMATIEN**	**TX DMA阀值中断使能控制** 如果该位被置为1且DMA当前地址等于I2S_TXTHADDR寄存器，中断发生。 0 = 禁用中断. 1 = 使能中断
[6]	**TDMAEIEN**	**TX DMA 结束中断使能控制** 如果该位被置1，且DMA当前地址等于I2S_TXEADDR寄存器，中断发生。 0 = 禁用中断. 1 = 使能中断
[5]	**RDMATIEN**	**RX DMA 阀值中断使能控制** 如果该位被置为1且DMA当前地址等于I2S_ RXTHADDR寄存器，中断发生。 0 = 禁用中断. 1 = 使能中断
[4]	**RDMAEIEN**	**RX DMA 结束中断使能控制** 如果该位被置1，且DMA当前地址等于I2S_ RXEADDR寄存器，中断发生。 0 = 禁用中断. 1 = 使能中断
[3]	**Reserved**	保留.
[2]	**RXTHIEN**	**接收 FIFO 阀值水平中断使能控制** 当接收FIFO的数据字等于或大于RXTH[3:0]，且RXTHIF位被设置为1，如果RXTHIEN位使能，中断发生。 0 = 禁用中断. 1 = 使能中断
[1]	**RXOVIEN**	**接收 FIFO 溢出中断使能控制** 0 = 禁用中断. 1 = 使能中断
[0]	**RXUDIEN**	**接收 FIFO 下溢中断使能** 如果软件读接收FIFO，当它读空，. I2S_STATUS中的标志RXUDIF被设置为1 0 = 禁用中断. 1 = 使能中断

I²S 状态寄存器 (I2S_STATUS)

寄存器	偏移地址	R/W	描述	复位值
I2S_STATUS	I2S_BA+0x0C	R/W	I²S 状态寄存器	0x0014_1000

31	30	29	28	27	26	25	24
TXCNT				RXCNT			
23	22	21	20	19	18	17	16
LZCIF	RZCIF	TXBUSY	TXEMPTY	TXFULL	TXTHIF	TXOVIF	TXUDIF
15	14	13	12	11	10	9	8
Reserved			RXEMPTY	RXFULL	RXTHIF	RXOVIF	RXUDIF
7	6	5	4	3	2	1	0
TDMATIF	TDMAEIF	RDMATIF	RDMAEIF	RIGHT	TXIF	RXIF	I2SIF

位	描述	
[31:28]	TXCNT	**发送 FIFO 水平** 这些位表示在发送FIFO中字的个数 0000 = 无数据或16 个字 (需要检查 TX 满标志). 0001 =在发送FIFO中有1个字 …. 1111 =在发送FIFO中有15个字 **注:** 如果 TXFULL 标志是 1, 并且 TXCNT = 0x0，意味着有16 个字在FIFO.中
[27:24]	RXCNT	**接收 FIFO 水平** 这些位表示在接收FIFO中字的个数 0000 =无数据或16 个字 (需要检查 RX 满标志). 0001 =在接收FIFO中有1个字. …. 1111 =在接收FIFO中有15个字. **注:** 如果 RXFULL 标志是 1, 并且 RXCNT = 0x0. 意味着有16 个字在FIFO.中
[23]	LZCIF	**左声道过零标志** 该位表示左声道下一个采样数据符号位改变或所有数据位为 0. 0 =没有过零 1 =检测到左声道过零 软件可以写 1 清除该位.

[22]	**RZCIF**	**右声道过零标志** 该位表示右声道下一个采样数据符号位改变或所有数据位为 0. 0 =没有过零 1 =检测到右声道过零 软件可以写 1 清除该位.
[21]	**TXBUSY**	**发送忙** 当发送 FIFO 和移位缓存中的所有数据都被移出时，该位清零。当第一个数据加载到移位缓存时，该位置 1. 0 = 发送移位缓存为空 1 = 发送移位缓存忙 该位只读.
[20]	**TXEMPTY**	**发送 FIFO 空** 该位反映发送 FIFO 中的数据字数为 0 0 = 非空 1 = 空 该位只读.
[19]	**TXFULL**	**发送 FIFO 满** 该位反映发送 FIFO 中的数据字数为16 0 = 未满. 1 = 满. 该位只读
[18]	**TXTHIF**	**发送 FIFO 阈值标志** 当发送 FIFO 中的数据字数等于或低于TXTH[3:0]中设定的阈值时，TXTHIF位变为 1。该位将保持 1 直到软件写数据到I2S_TX寄存器后TXCNT[3:0]高于TXTH[3:0] 0 =发送FIFO 中的数据字数高于阈值水平 1 =发送FIFO 中的数据字数等于或低于阈值水平 该位只读
[17]	**TXOVIF**	**发送 FIFO 溢出标志** 当发送 FIFO 已满时写数据到该 FIFO，该位被置为 1 0 = 没有溢出 1 = 溢出 软件可以写 1 清除该位.
[16]	**TXUDIF**	**发送 FIFO 下溢标志** 当发送 FIFO 为空时，移位逻辑硬件从数据 FIFO 读数据，会导致该位被置为1. 0 = 没有下溢 1 = 下溢 软件可以写 1 清除该位.
[15:13]	**Reserved**	保留.

[12]	**RXEMPTY**	**接收 FIFO 空** 该位反映接收 FIFO 中的数据字数为0 0 = FIFO非空 1 = FIFO空 该位只读.
[11]	**RXFULL**	**接收 FIFO 满** 该位反映接收 FIFO 中的数据字数为16 0 = 未满. 1= 满. 该位只读
[10]	**RXTHIF**	**接收 FIFO 阈值标志** 当接收 FIFO 中数据字等于或高于 RXTH[3:0] 中所设定的阈值时，RXTHIF位 变为 1。该位将保持 1直到软件读I2S_RX寄存器后RXCNT[3:0]低于RXTH[3:0] 0 = 接收FIFO 中数据字数低于阈值水平 1 = 接收FIFO 中数据字数等于或高于阈值水平 该位只读
[9]	**RXOVIF**	**接收 FIFO 溢出标志** 当接收 FIFO 已满，接收硬件还尝试写数据到接收 FIFO，则该位被置为1，第一个缓存中数据被覆盖. 0 = 没发生溢出 1 = 发生溢出 软件可以写 1 清除该位.
[8]	**RXUDIF**	**接收 FIFO 下溢标志** 当接收 FIFO 为空时，读接收 FIFO，则该位置 1，表示发生下溢. 0 = 没发生下溢 1 = 发生下溢 软件可以写 1 清除该位.
[7]	**TDMATIF**	**TX DMA 等于阀值地址中断标志** 如果 TX DMA当前地址等于I2S_TXTHADDR寄存器，该中断标志会置1 。如果TDMATIEN置1，一个NVIC中断会产生。. 0 = 没有 TX 阀值中断. 1 = 产生TX 阀值中断. 软件可以写 1 清除该位.
[6]	**TDMAEIF**	**TX DMA 等于结束地址中断标志** 如果TX DMA当前地址等于I2S_TXEADDR寄存器，该中断标志置1，如果TDMAEIEN置位，一个NVIC中断会产生。 0 = 没有 TX 结束中断. 1 = TX 结束中断. 软件可以写 1 清除该位.

[5]	**RDMATIF**	**RX DMA 等于阀值地址中断标志** 如果RX DMA当前地址等于I2S_ RXTHADDR寄存器，该中断标志会置1 。如果RDMATIEN置1，一个NVIC中断会产生。. 0 = 没有RX阀值中断. 1 = 产生RX阀值中断. 软件可以写 1 清除该位.
[4]	**RDMAEIF**	**RX DMA 等于结束地址中断标志** 如果RX DMA当前地址等于I2S_ RXEADDR寄存器，该中断标志置1，如果RDMAEIEN置位，一个NVIC中断会产生。 0 = 没有RX结束中断. 1 = RX结束中断. 软件可以写 1 清除该位.
[3]	**RIGHT**	**右声道** 该位表示当前发送的数据属于右声道 0 = 左声道 1 = 右声道 该位只读
[2]	**TXIF**	**I^2S 发送中断** 0 = 无发送中断发生 1 = 发送中断发生 如果LZCIF, RZCIF, TXTHIF, TXOIF, TXUDIF, TDMATIF, 和 TDMAEIF中任何一个发生置位，该标志被触发。 该位只读
[1]	**RXIF**	**I^2S 接收中断** 0 = 无接收中断发生 1 = 接收中断发生 如果RXTHIF, RXOVIF, RXUDIF, RDMATIF, 和 RDMAEIF中任何一个发生置位，该标志被触发。 该位只读
[0]	**I2SIF**	**I^2S 中断标志** 0 = 无 I^2S 中断发生 1 = I^2S 中断发生 如果TXIF 或 RXIF位使能，该标志被触发。 该位只读.

I²S 发送 FIFO 寄存器 (I2S_TX)

寄存器	偏移地址	R/W	描述	复位值
I2S_TX	I2S_BA+0x10	W	I²S 发送 FIFO 寄存器	0x0000_0000

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

位	描述	
[31:0]	TX	**发送 FIFO 寄存器** I²S包含16个字(16x32 bit)数据FIFO，用于数据发送。写数据到该寄存器准备传输数据。剩余的字个数由I2S_STATUS中TXCNT[3:0] 指示。

I²S 接收 FIFO 寄存器 (I2S_RX)

寄存器	偏移地址	R/W	描述	复位值
I2S_RX	I2S_BA+0x14	R	I²S 接收 FIFO 寄存器	0x0000_0000

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

位	描述	
[31:0]	RX	**接收 FIFO 寄存器** I²S包含16个字(16x32 bit)数据FIFO，用于数据接收。读该寄存器来得到FIFO中的数据。剩余的字个数由I2S_STATUS中RXCNT [3:0] 指示。

I²S 虚拟 I²C 控制寄存器 (I2S_CODECCTL)

寄存器	偏移地址	R/W	描述	复位值
I2S_CODECCTL	I2S_BA+0x18	R/W	I²S 虚拟 I²C 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
BUSY	I2CCKDIV						
23	22	21	20	19	18	17	16
DEVID							RW
15	14	13	12	11	10	9	8
ADDR							
7	6	5	4	3	2	1	0
DAT							

位	描述	
[31]	**BUSY**	**忙标志** 如果'I2S_CODECCTL'寄存器已经被写，硬件会改变命令为I²C格式，因为内部音频codec接口是I²C。然而I²C的速度是慢的，所以该位用来指示I²C命令的结束。 0 = I²C命令结束 1 = I²C 命令没有结束.
[30:24]	**I2CCKDIV**	**SCK 时钟分频器** 控制 SCK 时序参数 SCK 频率是 (F_I2SCLK / (I2CCKDIV * 16)). **注1:** 不能为0. **注2:** F_SCK必须低于或等于F_MCLK / 16.
[23:17]	**DEVID**	**内部音频 CODEC 设备 ID** 该参数应该设置为40H.
[16]	**RW**	**读或写命令** 控制该命令来从内部音频CODEC 读取数据或写数据到内部音频CODEC 0 = CODEC.从内部音频CODEC 读取数据 1 =写数据到内部音频CODEC
[15:8]	**ADDR**	**地址信息** 该参数用来从内部音频CODEC读数据或写数据到内部音频CODEC
[7:0]	**DAT**	**数据信息** 该参数用来从内部音频CODEC读数据或写数据到内部音频CODEC

I²S TX DMA 开始地址寄存器 (I2S_TXSTADDR)

寄存器	偏移地址	R/W	描述	复位值
I2S_TXSTADDR	I2S_BA+0x20	R/W	I²S TX DMA 开始地址寄存器	0x2001_0000

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

位	描述	
[31:0]	**ADDR**	**TX DMA 开始地址寄存器** **注:**该地址是字对齐

I²S TX DMA 阀值地址寄存器 (I2S_TXTHADDR)

寄存器	偏移地址	R/W	描述	复位值
I2S_TXTHADDR	I2S_BA+0x24	R/W	I²S TX DMA 阀值地址寄存器	0x2001_FFF0

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

位	描述	
[31:0]	ADDR	TX DMA 阀值地址寄存器 **注:** 该地址是字对齐

I²S TX DMA 结束地址寄存器 (I2S_TXEADDR)

寄存器	偏移地址	R/W	描述	复位值
I2S_TXEADDR	I2S_BA+0x28	R/W	I²S TX DMA 结束地址寄存器	0x2001_FFF0

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

位	描述	
[31:0]	ADDR	**TX DMA 结束地址寄存器** **注1:** 该地址是字对齐 **注2:**如果WDWIDTH[1:0] 等于 0x2 或 0x3，用户必须设置正确的结束地址以避免立体声模式在右声道和左声道之间交换

I²S TX DMA 当前地址寄存器 (I2S_TXCADDR)

寄存器	偏移地址	R/W	描述	复位值
I2S_TXCADDR	I2S_BA+0x2C	R	I²S TX DMA 当前地址寄存器	0x0000_0000

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

位	描述	
[31:0]	ADDR	TX DMA 当前地址寄存器

I²S RX DMA 开始地址寄存器 (I2S_RXSTADDR)

寄存器	偏移地址	R/W	描述	复位值
I2S_RXSTADDR	I2S_BA+0x30	R/W	I²S RX DMA 开始地址寄存器	0x2001_0000

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

位	描述	
[31:0]	ADDR	RX DMA 开始地址寄存器 **注:** 该地址是字对齐

I²S RX DMA 阀值地址寄存器 (I2S_RXTHADDR)

寄存器	偏移地址	R/W	描述	复位值
I2S_RXTHADDR	I2S_BA+0x34	R/W	I²S RX DMA 阀值地址寄存器	0x2001_FFF0

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

位	描述	
[31:0]	**ADDR**	**RX DMA 阀值地址寄存器** **注:** 该地址是字对齐

I²S RX DMA 结束地址寄存器 (I2S_RXEADDR)

寄存器	偏移地址	R/W	描述	复位值
I2S_RXEADDR	I2S_BA+0x38	R/W	I²S RX DMA 结束地址寄存器	0x2001_FFF0

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

位	描述	
[31:0]	ADDR	**RX DMA 结束地址寄存器** **注1:** 该地址是字对齐 **注2:**如果WDWIDTH[1:0] 等于 0x2 或 0x3，用户必须设置正确的结束地址以避免立体声模式在右声道和左声道之间交换

I²S RX DMA 当前地址寄存器 (I2S_RXCADDR)

寄存器	偏移地址	R/W	描述	复位值
I2S_RXCADDR	I2S_BA+0x3C	R	I²S RX DMA 当前地址寄存器	0x0000_0000

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

位	描述	
[31:0]	ADDR	RX DMA 当前地址寄存器

I²S RX 数据平均控制寄存器 (I2S_RXAVGCTL)

寄存器	偏移地址	R/W	描述	复位值
I2S_RXAVGCTL	I2S_BA+0x40	R/W	I²S RX 数据平均控制寄存器	0x0000_000A

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				WINSEL			

位	描述	
[31:0]	Reserved	保留.
[3:0]	WINSEL	**RX 数据平均窗口选择** 0000 = 平均窗口是 1 (2^0) 个采样. 0001 =平均窗口是2 (2^1) 个采样. 0010 =平均窗口是4 (2^2) 个采样. … 1110 =平均窗口是16384 (2^14) 个采样. 1111 =平均窗口是32768 (2^15) 个采样. **注:** 每一个窗口大小采样会产生一个平均结果

I²S RX 左声道数据平均值 (I2S_RXLCHAVG)

寄存器	偏移地址	R/W	描述	复位值
I2S_RXLCHAVG	I2S_BA+0x44	R	I²S RX 左声道数据平均值	0x0000_0000

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

位	描述	
[31:0]	RESULT	**RX 左声道数据平均结果** 左声道接收到数据的平均结果 **注:** 如果 MONO (I2S_CTL[6]) = 1, 平均结果仅在该寄存器，无论RXLCH (I2S_CTL[23]) = 1 或 0。.

I²S RX 右声道数据平均值 (I2S_RXRCHAVG)

寄存器	偏移地址	R/W	描述	复位值
I2S_RXRCHAVG	I2S_BA+0x48	R	I²S RX 右声道数据平均值	0x0000_0000

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

位	描述	
[31:0]	**RESULT**	**RX 右声道数据平均结果** 右声道接收到数据的平均结果 **注:** 如果 MONO (I2S_CTL[6]) = 1, 该寄存器无用。

6.14.9 CODEC 寄存器映射

该部分描述内部音频CODEC。内部音频CODEC寄存器不能被CPU直接访问。如何配置它们，请参考6.15.6.4部分

寄存器	偏移地址	R/W	描述	复位值
CODEC_56ADCCTL	0x00	RW	CODEC ADC 控制寄存器	0x10
CODEC_DACCTL	0x01	RW	CODEC DAC控制寄存器	0x00
CODEC_CTL	0x02	RW	CODEC控制寄存器	0x80
CODEC_DACLVCTL	0x03	RW	CODEC 左声道音量控制寄存器	0x80
CODEC_DACRVCTL	0x04	RW	CODEC右声道音量控制寄存器	0x80
CODEC_DACLVACTL	0x08	RW	CODEC左声道模拟量控制寄存器	0x00
CODEC_DACRVACTL	0x09	RW	CODEC 右声道模拟量控制寄存器	0x00
CODEC_DACMIXCTL	0x0A	RW	CODEC DAC 输出模拟混合控制寄存器	0x09
CODEC_DACPDCTL	0x0B	RW	CODEC模拟模块的 DAC 掉电控制寄存器	0xFF
CODEC_PWRCTL	0x0D	RW	CODEC 电源控制寄存器	0x3A
CODEC_ADCINCTL	0x0E	RW	CODEC ADC 输入控制寄存器	0x00
CODEC_ADCPDCTL	0x0F	RW	CODEC ADC 掉电控制寄存器	0xFF
CODEC_ADCLVCTL	0x10	RW	CODEC ADC 左通道增益控制寄存器	0x00
CODEC_ADCRVCTL	0x11	RW	CODEC ADC右通道增益控制寄存器	0x00
CODEC_STOCTL	0x12	RW	CODEC ADC Sidetone 音量控制寄存器	0x00

6.14.10 CODEC 寄存器描述

CODEC ADC 控制寄存器 (CODEC_ADCCTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_ADCCTL	0x00	RW	CODEC ADC 控制寄存器	0x10

7	6	5	4	3	2	1	0
ADCEN	HPFEN	Reserved	STEREO	Reserved			

位	描述	
[7]	**ADCEN**	**CODEC ADC 使能控制** 0 = 数字 ADC 功能禁止. 1 = 数字ADC 功能使能
[6]	**HPFEN**	**高通滤波器使能控制** 0 = 高通滤波器禁止. 1 = 高通滤波器使能
[5]	**Reserved**	保留.
[4]	**STEREO**	**CODEC ADC 输出通道模式** 0 = 左单声道 (右声道输出和左声道一样信号). 1 = 立体声模式
[3:0]	**Reserved**	保留.

CODEC DAC 控制寄存器 (CODEC_DACCTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_DACCTL	0x01	RW	CODEC DAC 控制寄存器	0x00

7	6	5	4	3	2	1	0
DACEN	Reserved	DAC_MODE		DAC_DMUTEL	DAC_DMUTER	Reserved	

位	描述	
[7]	**DACEN**	**CODEC DAC 使能控制** 0 = 数字 DAC 功能禁止 1 = 数字 DAC 功能使能.
[6]	**Reserved**	保留.
[5:4]	**DAC_MODE**	**CODEC DAC 输出立体声或单声道信号控制** 00 = 输出立体声信号模式. 01 = 输出左单声道信号模式. (右声道输出和左声道一样信号). 10 = 输出右单声道信号模式. (左声道输出和右声道一样信号). 11 = 输出 (左声道信号 + 右声道信号)/2 单通道模式
[3]	**DAC_DMUTEL**	**左通道数字静音使能控制** 0 = 通道数字输出静音禁止. 1 = 通道数字输出静音使能.
[2]	**DAC_DMUTER**	**右通道数字静音使能控制** 0 = 通道数字输出静音禁止. 1 = 通道数字输出静音使能.
[1:0]	**Reserved**	保留.

CODEC 控制寄存器 (CODEC_CTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_CTL	0x02	RW	CODEC 控制寄存器	0x80

7	6	5	4	3	2	1	0
SRESET	SLAVE	Reserved					

位	描述	
[7]	SRESET	**CODEC 软复位控制位** 0 = 复位 AD/DA滤波器和I^2S 部分, I^2C 模块除外. 1 = 正常工作.
[6]	SLAVE	**I^2S 接口模式选择** 0 = I^2S 接口工作在主机模式 1 = I^2S接口工作在从机模式.
[5:0]	Reserved	保留.

CODEC DAC 左声道音量控制寄存器 (CODEC_DACLVCTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_DACLVCTL	0x03	RW	CODEC DAC 左声道音量控制寄存器	0x80

7	6	5	4	3	2	1	0
DACVOLL							

位	描述	
[7:0]	DACVOLL	**DAC 左声道音量控制寄存器** 该值可以配置为从0x00 到 0x80，步进是1/128 * (音量最大范围). **注:** 默认设置是0x80 (最大音量).

CODEC DAC右声道音量控制寄存器(CODEC_DACRVCTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_DACRVCTL	0x04	RW	CODEC DAC右声道音量控制寄存器	0x80

7	6	5	4	3	2	1	0
DACVOLR							

位	描述	
[7:0]	DACVOLR	**DAC右声道音量控制寄存器** 该值可以配置为从0x00 到 0x80，步进是1/128 * (音量最大范围). **注:** 默认设置是0x80 (最大音量).

CODEC DAC 左声道模拟量控制寄存器 (CODEC_DACLVACTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_DACLVACTL	0x08	RW	CODEC DAC左声道模拟量控制寄存器	0x00

7	6	5	4	3	2	1	0
Reserved			AVOLL				

位	描述	
[7:5]	**Reserved**	保留.
[4:0]	**AVOLL**	**DAC左声道模拟量控制寄存器** 该值可以设置为从0x00 到 0x1F, 步进是 -2dB. 0x00 = 0 dB. 0x01 = -2 dB. 0x02 = -4 dB. … 0x1E = -60 dB. 0x1F = 模拟量静音

CODEC DAC右声道模拟量控制寄存器(CODEC_DACRVACTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_DACRVACTL	0x09	RW	CODEC DAC 右声道模拟量控制寄存器	0x00

7	6	5	4	3	2	1	0
Reserved			AVOLR				

位	描述	
[7:5]	**Reserved**	保留.
[4:0]	**AVOLR**	**DAC右声道模拟量控制寄存器** 该值可以设置为从0x00 到 0x1F, 步进是 -2dB. 0x00 = 0 dB. 0x01 = -2 dB. 0x02 = -4 dB. … 0x1E = -60 dB. 0x1F = 模拟量静音

CODEC DAC输出模拟混合控制寄存器(CODEC_DACMIXCTL)

Register	Offset	R/W	Description	Reset Value
CODEC_DACMIXCTL	0x0A	RW	CODEC DAC 输出模拟混合控制寄存器	0x09

7	6	5	4	3	2	1	0
Reserved		MIXSELL			MIXSELR		

Bits	Description	
[7:6]	**Reserved**	Reserved.
[5:3]	**MIXSELL**	**DAC 输出左通道 模拟混合控制** 000 = 无信号选择. 001 = 左DAC线输出信号选择. 010 = 左模拟线输入信号选择 100 = ADC MIC 信号选择 011 = 左DAC线输出信号及左模拟线输入信号混合 110 = 左模拟输入及 ADC MIC信号混合 101 = 左DAC线输出 及 ADC MIC信号混合 111 = 左DAC线输出, 左模拟线输入及 ADC MIC信号混合. **Note:** 左模拟线输入信能设定来源从 LLineIn 或MIC.
[2:0]	**MIXSELR**	**DAC 输出右通道 模拟混合控制** 000 = 无信号选择. 001 = 右DAC线输出信号选择. 010 = 右模拟线输入信号选择 100 = ADC MIC 信号选择 011 = 右DAC线输出信号及右模拟线输入信号混合 110 = 右模拟输入及 ADC MIC信号混合 101 = 右DAC线输出 及 ADC MIC信号混合 111 = 右DAC线输出, 右模拟线输入及 ADC MIC信号混合. **Note:** 右模拟线输入信能设定来源从 RLineIn 或 MIC.

CODEC DAC 模拟模块的掉电控制寄存器 (CODEC_DACPDCTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_DACPDCTL	0x0B	RW	CODEC DAC 掉电控制寄存器	0xFF

7	6	5	4	3	2	1	0
Reserved		HPROI	PDCHG	PWDNL	PWDNR	PDPAL	PDPAR

位	描述	
[7:6]	**Reserved**	保留.
[5]	**HPROI**	**耳机输出阻抗控制寄存器** 0 = 耳机输出阻抗是1 KΩ. 1 =耳机输出阻抗是40 KΩ. **注:** .当耳机模拟模块在掉电的时候，该位是有作用的。
[4]	**PDCHG**	**VMID 预充电电容控制** 0 = VMID 预充电电容禁止 1 = VMID 预充电电容使能.
[3]	**PWDNL**	**DAC 左声道模拟模块电源控制** 0 = 通道电源使能. 1 = 通道电源禁止
[2]	**PWDNR**	**DAC 右声道模拟模块电源控制** 0 = 通道电源使能 1 = 通道电源禁止.
[1]	**PDPAL**	**DAC 左耳机输出模拟模块电源控制** 0 = 耳机模拟模块电源使能 1 = 耳机模拟模块电源禁止
[0]	**PDPAR**	**DAC 右耳机输出模拟模块电源控制** 0 = 耳机模拟模块电源使能 1 = 耳机模拟模块电源禁止

CODEC 电源控制寄存器 (CODEC_PWRCTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_PWRCTL	0x0D	RW	CODEC 电源控制寄存器	0x3A

7	6	5	4	3	2	1	0
Reserved				PDBIAS	RESADJ		

位	描述	
[7:4]	**Reserved**	保留.
[3]	**PDBIAS**	**CODEC 参考 (MIC_BIAS) 电源控制** 0 = 参考电源使能 1 = 参考电源禁止
[2:0]	**RESADJ**	**当前偏流电阻器选择** 000 = 最小偏流电阻器 001 = 最大偏流电阻器 010 = 中大偏流电阻器 100 = 中小偏流电阻器 其它 = 保留.

CODEC ADC 输入控制寄存器 (CODEC_ADCINCTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_ADCINCTL	0x0E	RW	CODEC ADC 输入控制寄存器	0x00

7	6	5	4	3	2	1	0
Reserved					MIC_SEL	ADCINSEL	

位	描述	
[7:3]	**Reserved**	保留.
[2]	**MIC_SEL**	**麦克风输入源选择** 0 = 输入信号来自 MIC0. 1 = 输入信号来自MIC1.
[1:0]	**ADCINSEL**	**模拟输入信号源选择** 00 = 模拟输入信号来自线路输入 10 = 模拟输入信号来自麦克风输入 其它 = 保留..

CODEC ADC 掉电控制寄存器 (CODEC_ADCPDCTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_ADCPDCTL	0x0F	RW	CODEC ADC 掉电控制寄存器	0xFF

7	6	5	4	3	2	1	0
Reserved		PDMIC	Reserved	PDPGAL	PDPGAR	PDL	PDR

位	描述	
[7:6]	**Reserved**	保留.
[5]	**PDMIC**	**麦克风偏流的电源控制和右声道线路输入选择** 0 = MIC-BIAS引脚用于麦克风偏流功能，且偏流电源使能。 1 = MIC-BIAS 引脚用于右线路输入功能，且偏流电源禁止。
[4]	**Reserved**	保留.
[3]	**PDPGAL**	**左通道模拟增益模块电源控制寄存器** 0 = 增益模块电源使能 1 = 增益模块电源禁止
[2]	**PDPGAR**	**右通道模拟增益模块电源控制寄存器** 0 = 增益模块电源使能 1 = 增益模块电源禁止
[1]	**PDL**	**左通道 Sigma-Delta 模块电源控制** 0 = Sigma-Delta模块电源使能. 1 = Sigma-Delta 模块电源禁止
[0]	**PDR**	**右通道 Sigma-Delta 模块电源控制** 0 = Sigma-Delta模块电源使能. 1 = Sigma-Delta 模块电源禁止

CODEC ADC 左通道增益控制寄存器 (CODEC_ADCLVCTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_ADCLVCTL	0x10	RW	CODEC ADC左通道增益控制寄存器	0x00

7	6	5	4	3	2	1	0
Reserved			PREVOL	ADC_VOLL			

位	描述	
[7:5]	**Reserved**	保留.
[4]	**PREVOL**	**麦克风输入信号预增益增强控制寄存器** 0 = 预增益是 0 dB. 1 = 预增益是20 dB.
[3:0]	**ADC_VOLL**	**ADC 左通道增益控制** 该值可以设置成从 0x0 到 0xF, 步进是1.6dB. 0x0 = 0 dB. 0x1 = 1.6 dB. 0x2 = 3.2 dB. … 0xE = 22.4 dB. 0xF = 静音

CODEC ADC右通道增益控制寄存器(CODEC_ADCRVCTL)

寄存器	偏移地址	R/W	描述	复位值
CODEC_ADCRVCTL	0x11	RW	CODEC ADC 右通道增益控制寄存器	0x00

7	6	5	4	3	2	1	0
Reserved				ADC_VOLR			

位	描述	
[7:4]	**Reserved**	保留.
[3:0]	**ADC_VOLR**	**ADC右通道增益控制** 该值可以设置成从 0x0 到 0xF, 步进是1.6dB. 0x0 = 0 dB. 0x1 = 1.6 dB. 0x2 = 3.2 dB. … 0xE = 22.4 dB. 0xF = 静音

CODEC ADC Sidetone 音量控制寄存器(CODEC_STOCTL)

Register	Offset	R/W	Description	Reset Value
CODEC_STOCTL	0x12	RW	CODEC ADC Sidetone 音量控制寄存器	0x00

7	6	5	4	3	2	1	0
Reserved				STO_VOL			

Bits	Description	
[7:4]	**Reserved**	Reserved.
[3:0]	**STO_VOL**	**ADC Sidetone 音量控制寄存器** 音量设定值可以设定从0x0 到 0xF, 每个阶数为 -1.6dB. 0x0 = 0 dB. 0x1 = -1.6 dB. 0x2 = -3.2 dB. … 0xE = -22.4 dB. 0xF = 静音.

6.15 USB 2.0设备控制器(USBD)

6.15.1 概述

USB设备控制器接口含AHB总线和UTMI总线。USB控制器包含AHB主机接口和AHB从机接口。CPU通过AHB从机接口编程USB控制器寄存器。对于IN或OUT传输，USB设备控制器需要通过AHB主机接口写数据到内存或从内存读数据。USB设备控制器与USB2.0规范兼容，除控制端点外，它包含12个可配置的端点。这些端点可以配置为批量，中断或等时传输。USB控制器有一个内建的DMA来减轻CPU的负担。

6.15.2 特性

- 兼容USB 2.0版本规范
- 除控制端点外，支持12个可配置的端点。
- 每一个端点可以是同步，批量或中断传输，方向可以是IN或OUT 。
- IN端点的三种不同的操作模式－ 自动确认模式, 手动确认模式, Fly 模式
- 支持 DMA 操作
- 2048字节可配置RAM 用于端点缓存
- 支持端点最大包大小到1024字节

6.15.3 框图

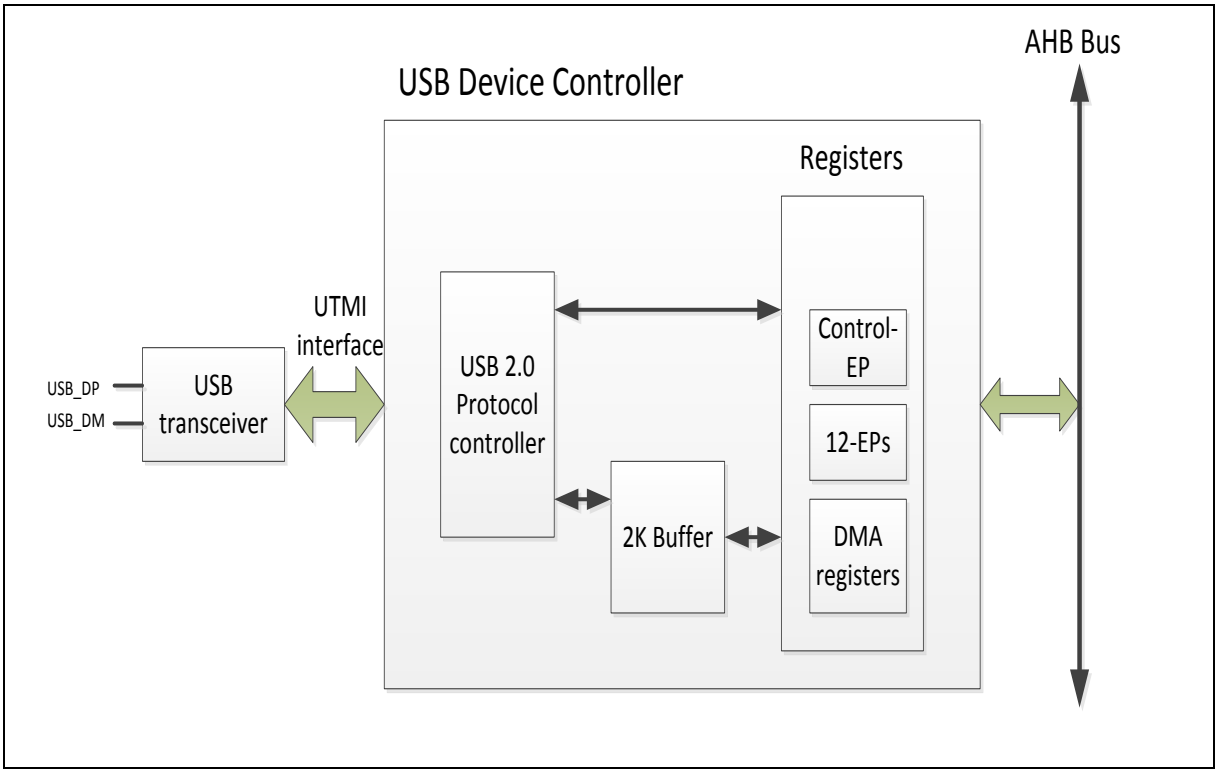


图 6.15-1 USB 设备控制器框图

6.15.4 功能描述

6.15.4.1 不同IN传输模式的操作

任何IN传输的数据要写入到内部缓存，当收到in-token，数据会依次发送到主机。CPU确认发送到主机的数据，有三种不同的模式。

- 自动确认模式
- 手动确认模式
- Fly模式

6.15.4.2 自动确认模式

如果一个端点选择工作在自动确认模式，端点仅在数据有效载荷等于EPMPS寄存器时响应。端点控制器等待直到数据量等于EPMPS的值，然后确认数据。如果CPU需要在传输结束的时候发送一个短包，USBD_EPxRSPCTL[6]寄存器中SHORTTXEN位应该置位。当该位置位，任何缓存中剩余的数据会被确认并在接下来的in-token发送到主机。

该模式需要最少的CPU干涉，大部分工作由USB设备控制器来完成。当发送到主机的数据有效载荷总是等于MPS大小，可以选择该模式。

SHORTTXEN	缓存中有效数据	数据发送/NAK 发送
0	< 最大包大小	NAK 发送
0	>= 最大包大小	最大包大小的数据有效荷载
1	< 最大包大小	小于最大包大小的有效数据
1	>=最大包大小	最大包大小发送的数据有效载荷

6.15.4.3 手动确认模式

如果端点选择工作在手动确认模式，CPU每次确认缓存中的数据，端点响应。CPU必须写数据到缓存，然后写数据的个数到EPxTXCNT寄存器。一旦缓存中的数据由写入数据个数到EPxTXCNT寄存器确认，数据会在收到in-token时发送到主机。

该模式每次传输需要CPU干预，但是很有用，如果每次发送的数据个数不是固定的，那么它由CPU来决定。

EPxTXCNT 写	缓存中有效数据	数据发送/NAK 发送
NO	-	NAK
YES	EPxTXCNT	EPxTXCNT 发送的数据有效载荷

6.15.4.4 Fly模式

Fly模式是最简单的操作模式，没有确认的步骤。缓存由CPU来填。如果主机发送一个in-token，缓存中的数据会自动确认并发送到主机。如果缓存中的数据大小超过最大包大小，控制器会自动打包成等于MPS并发送到主机。

该模式需要最少的CPU干预，该模式最适用于同步数据传输，因为该传输的速度比包大小重要。

缓存中有效数据	数据发送
< 最大包大小	数据有效发送
>=最大包大小	

6.15.4.5 分散-聚集功能

当分散聚集的DMA功能使能，SGEN被置高且USBD_DMACNT设置成8个字节，DMA会被使能来获取描述符，该描述符描述了真实的内存地址和长度。该描述符会使用一个8字节格式，如下：

	格式			
	[31]	[30]	[29:0]	
Word0	MEM_ADDR[31:0]			
Word1	EOT	RD	保留	Count[19:0]

MEM_ADDR: 它指定了内存地址 (AHB 地址).

EOT: 传输结束，当该位被置成高，意味着这是最后的描述符。

RD: “1” 意思是从内存读数据到缓存， “0” 意思是从缓存读数据到内存。

6.15.5 寄存器映射

寄存器	偏移地址	R/W	描述	复位值
USBD 基地址: **USBD_BA = 0x4001_9000**				
USBD_GINTSTS	USBD_BA+0x000	R	全局中断状态寄存器	0x0000_0000
USBD_GINTEN	USBD_BA+0x008	R/W	全局中断使能寄存器	0x0000_0001
USBD_BUSINTSTS	USBD_BA+0x010	R/W	USB 总线中断状态寄存器	0x0000_0000
USBD_BUSINTEN	USBD_BA+0x014	R/W	USB总线中断使能寄存器	0x0000_0040
USBD_OPER	USBD_BA+0x018	R/W	USB 操作寄存器	0x0000_0002
USBD_FRAMECNT	USBD_BA+0x01C	R	USB 帧计数寄存器	0x0000_0000
USBD_FADDR	USBD_BA+0x020	R/W	USB 功能地址寄存器	0x0000_0000
USBD_TEST	USBD_BA+0x024	R/W	USB 测试模式寄存器	0x0000_0000
USBD_CEPDAT	USBD_BA+0x028	R/W	控制端点数据缓存	0x0000_0000
USBD_CEPCTL	USBD_BA+0x02C	R/W	控制端点控制寄存器	0x0000_0000
USBD_CEPINTEN	USBD_BA+0x030	R/W	控制端点中断使能控制寄存器	0x0000_0000
USBD_CEPINTSTS	USBD_BA+0x034	R/W	控制端点中断状态	0x0000_1800
USBD_CEPTXCNT	USBD_BA+0x038	R/W	控制端点IN传输数据计数	0x0000_0000
USBD_CEPRXCNT	USBD_BA+0x03C	R	控制端点OUT传输数据计数	0x0000_0000
USBD_CEPDATCNT	USBD_BA+0x040	R	控制端点数据计数	0x0000_0000
USBD_SETUP1_0	USBD_BA+0x044	R	Setup1 & Setup0 字节	0x0000_0000
USBD_SETUP3_2	USBD_BA+0x048	R	Setup3 & Setup2字节	0x0000_0000
USBD_SETUP5_4	USBD_BA+0x04C	R	Setup5 & Setup4字节	0x0000_0000
USBD_SETUP7_6	USBD_BA+0x050	R	Setup7 & Setup6字节	0x0000_0000
USBD_CEPBUFSTART	USBD_BA+0x054	R/W	控制端点RAM开始地址寄存器	0x0000_0000
USBD_CEPBUFEND	USBD_BA+0x058	R/W	控制端点RAM结束地址寄存器	0x0000_0000
USBD_DMACTL	USBD_BA+0x05C	R/W	DMA 控制状态寄存器	0x0000_0000
USBD_DMACNT	USBD_BA+0x060	R/W	DMA 计数寄存器	0x0000_0000
USBD_EPADAT	USBD_BA+0x064	R/W	端点 A数据寄存器	0x0000_0000
USBD_EPAINTSTS	USBD_BA+0x068	R/W	端点 A中断状态寄存器	0x0000_0003
USBD_EPAINTEN	USBD_BA+0x06C	R/W	端点A 中断使能寄存器	0x0000_0000
USBD_EPADATCNT	USBD_BA+0x070	R	端点A 数据有效计数寄存器	0x0000_0000
USBD_EPARSPCTL	USBD_BA+0x074	R/W	端点A 响应控制寄存器	0x0000_0000

USBD_EPAMPS	USBD_BA+0x078	R/W	端点A最大包大小寄存器	0x0000_0000
USBD_EPATXCNT	USBD_BA+0x07C	R/W	端点A 传输计数寄存器	0x0000_0000
USBD_EPACFG	USBD_BA+0x080	R/W	端点A 配置寄存器	0x0000_0012
USBD_EPABUFSTART	USBD_BA+0x084	R/W	端点A RAM 开始地址寄存器	0x0000_0000
USBD_EPABUFEND	USBD_BA+0x088	R/W	端点A RAM 结束地址寄存器	0x0000_0000
USBD_EPBDAT	USBD_BA+0x08C	R/W	端点 B数据寄存器	0x0000_0000
USBD_EPBINTSTS	USBD_BA+0x090	R/W	端点 B中断状态寄存器	0x0000_0003
USBD_EPBINTEN	USBD_BA+0x094	R/W	端点B 中断使能寄存器	0x0000_0000
USBD_EPBDATCNT	USBD_BA+0x098	R	端点B 数据有效计数寄存器	0x0000_0000
USBD_EPBRSPCTL	USBD_BA+0x09C	R/W	端点B 响应控制寄存器	0x0000_0000
USBD_EPBMPS	USBD_BA+0x0A0	R/W	端点B最大包大小寄存器	0x0000_0000
USBD_EPBTXCNT	USBD_BA+0x0A4	R/W	端点B 传输计数寄存器	0x0000_0000
USBD_EPBCFG	USBD_BA+0x0A8	R/W	端点B 配置寄存器	0x0000_0022
USBD_EPBBUFSTART	USBD_BA+0x0AC	R/W	端点B RAM 开始地址寄存器	0x0000_0000
USBD_EPBBUFEND	USBD_BA+0x0B0	R/W	端点B RAM 结束地址寄存器	0x0000_0000
USBD_EPCDAT	USBD_BA+0x0B4	R/W	端点 C数据寄存器	0x0000_0000
USBD_EPCINTSTS	USBD_BA+0x0B8	R/W	端点 C中断状态寄存器	0x0000_0003
USBD_EPCINTEN	USBD_BA+0x0BC	R/W	端点C 中断使能寄存器	0x0000_0000
USBD_EPCDATCNT	USBD_BA+0x0C0	R	端点C 数据有效计数寄存器	0x0000_0000
USBD_EPCRSPCTL	USBD_BA+0x0C4	R/W	端点C 响应控制寄存器	0x0000_0000
USBD_EPCMPS	USBD_BA+0x0C8	R/W	端点C最大包大小寄存器	0x0000_0000
USBD_EPCTXCNT	USBD_BA+0x0CC	R/W	端点C 传输计数寄存器	0x0000_0000
USBD_EPCCFG	USBD_BA+0x0D0	R/W	端点C 配置寄存器	0x0000_0032
USBD_EPCBUFSTART	USBD_BA+0x0D4	R/W	端点C RAM 开始地址寄存器	0x0000_0000
USBD_EPCBUFEND	USBD_BA+0x0D8	R/W	端点C RAM 结束地址寄存器	0x0000_0000
USBD_EPDDAT	USBD_BA+0x0DC	R/W	端点 D数据寄存器	0x0000_0000
USBD_EPDINTSTS	USBD_BA+0x0E0	R/W	端点 D中断状态寄存器	0x0000_0003
USBD_EPDINTEN	USBD_BA+0x0E4	R/W	端点D 中断使能寄存器	0x0000_0000
USBD_EPDDATCNT	USBD_BA+0x0E8	R	端点D 数据有效计数寄存器	0x0000_0000
USBD_EPDRSPCTL	USBD_BA+0x0EC	R/W	端点D 响应控制寄存器	0x0000_0000
USBD_EPDMPS	USBD_BA+0x0F0	R/W	端点D最大包大小寄存器	0x0000_0000
USBD_EPDTXCNT	USBD_BA+0x0F4	R/W	端点D 传输计数寄存器	0x0000_0000

USBD_EPDCFG	USBD_BA+0x0F8	R/W	端点D 配置寄存器	0x0000_0042
USBD_EPDBUFSTART	USBD_BA+0x0FC	R/W	端点D RAM 开始地址寄存器	0x0000_0000
USBD_EPDBUFEND	USBD_BA+0x100	R/W	端点DRAM 结束地址寄存器	0x0000_0000
USBD_EPEDAT	USBD_BA+0x104	R/W	端点 E数据寄存器	0x0000_0000
USBD_EPEINTSTS	USBD_BA+0x108	R/W	端点 E中断状态寄存器	0x0000_0003
USBD_EPEINTEN	USBD_BA+0x10C	R/W	端点E 中断使能寄存器	0x0000_0000
USBD_EPEDATCNT	USBD_BA+0x110	R	端点E 数据有效计数寄存器	0x0000_0000
USBD_EPERSPCTL	USBD_BA+0x114	R/W	端点E 响应控制寄存器	0x0000_0000
USBD_EPEMPS	USBD_BA+0x118	R/W	端点E最大包大小寄存器	0x0000_0000
USBD_EPETXCNT	USBD_BA+0x11C	R/W	端点E 传输计数寄存器	0x0000_0000
USBD_EPECFG	USBD_BA+0x120	R/W	端点E 配置寄存器	0x0000_0052
USBD_EPEBUFSTART	USBD_BA+0x124	R/W	端点E RAM 开始地址寄存器	0x0000_0000
USBD_EPEBUFEND	USBD_BA+0x128	R/W	端点E RAM 结束地址寄存器	0x0000_0000
USBD_EPFDAT	USBD_BA+0x12C	R/W	端点 F数据寄存器	0x0000_0000
USBD_EPFINTSTS	USBD_BA+0x130	R/W	端点 F中断状态寄存器	0x0000_0003
USBD_EPFINTEN	USBD_BA+0x134	R/W	端点F中断使能寄存器	0x0000_0000
USBD_EPFDATCNT	USBD_BA+0x138	R	端点F数据有效计数寄存器	0x0000_0000
USBD_EPFRSPCTL	USBD_BA+0x13C	R/W	端点F 响应控制寄存器	0x0000_0000
USBD_EPFMPS	USBD_BA+0x140	R/W	端点F最大包大小寄存器	0x0000_0000
USBD_EPFTXCNT	USBD_BA+0x144	R/W	端点F 传输计数寄存器	0x0000_0000
USBD_EPFCFG	USBD_BA+0x148	R/W	端点F 配置寄存器	0x0000_0062
USBD_EPFBUFSTART	USBD_BA+0x14C	R/W	端点F RAM 开始地址寄存器	0x0000_0000
USBD_EPFBUFEND	USBD_BA+0x150	R/W	端点FRAM 结束地址寄存器	0x0000_0000
USBD_EPGDAT	USBD_BA+0x154	R/W	端点 G数据寄存器	0x0000_0000
USBD_EPGINTSTS	USBD_BA+0x158	R/W	端点 G中断状态寄存器	0x0000_0003
USBD_EPGINTEN	USBD_BA+0x15C	R/W	端点G 中断使能寄存器	0x0000_0000
USBD_EPGDATCNT	USBD_BA+0x160	R	端点G 数据有效计数寄存器	0x0000_0000
USBD_EPGRSPCTL	USBD_BA+0x164	R/W	端点G 响应控制寄存器	0x0000_0000
USBD_EPGMPS	USBD_BA+0x168	R/W	端点G最大包大小寄存器	0x0000_0000
USBD_EPGTXCNT	USBD_BA+0x16C	R/W	端点G 传输计数寄存器	0x0000_0000
USBD_EPGCFG	USBD_BA+0x170	R/W	端点G 配置寄存器	0x0000_0072
USBD_EPGBUFSTART	USBD_BA+0x174	R/W	端点G RAM 开始地址寄存器	0x0000_0000

USBD_EPGBUFEND	USBD_BA+0x178	R/W	端点G RAM 结束地址寄存器	0x0000_0000
USBD_EPHDAT	USBD_BA+0x17C	R/W	端点 H数据寄存器	0x0000_0000
USBD_EPHINTSTS	USBD_BA+0x180	R/W	端点 H中断状态寄存器	0x0000_0003
USBD_EPHINTEN	USBD_BA+0x184	R/W	端点H 中断使能寄存器	0x0000_0000
USBD_EPHDATCNT	USBD_BA+0x188	R	端点H 数据有效计数寄存器	0x0000_0000
USBD_EPHRSPCTL	USBD_BA+0x18C	R/W	端点H 响应控制寄存器	0x0000_0000
USBD_EPHMPS	USBD_BA+0x190	R/W	端点H最大包大小寄存器	0x0000_0000
USBD_EPHTXCNT	USBD_BA+0x194	R/W	端点H 传输计数寄存器	0x0000_0000
USBD_EPHCFG	USBD_BA+0x198	R/W	端点H 配置寄存器	0x0000_0082
USBD_EPHBUFSTART	USBD_BA+0x19C	R/W	端点H RAM 开始地址寄存器	0x0000_0000
USBD_EPHBUFEND	USBD_BA+0x1A0	R/W	端点H RAM 结束地址寄存器	0x0000_0000
USBD_EPIDAT	USBD_BA+0x1A4	R/W	端点I 数据寄存器	0x0000_0000
USBD_EPIINTSTS	USBD_BA+0x1A8	R/W	端点 I 中断状态寄存器	0x0000_0003
USBD_EPIINTEN	USBD_BA+0x1AC	R/W	端点I 中断使能寄存器	0x0000_0000
USBD_EPIDATCNT	USBD_BA+0x1B0	R	端点I 数据有效计数寄存器	0x0000_0000
USBD_EPIRSPCTL	USBD_BA+0x1B4	R/W	端点I 响应控制寄存器	0x0000_0000
USBD_EPIMPS	USBD_BA+0x1B8	R/W	端点I 最大包大小寄存器	0x0000_0000
USBD_EPITXCNT	USBD_BA+0x1BC	R/W	端点I 传输计数寄存器	0x0000_0000
USBD_EPICFG	USBD_BA+0x1C0	R/W	端点I 配置寄存器	0x0000_0092
USBD_EPIBUFSTART	USBD_BA+0x1C4	R/W	端点I RAM 开始地址寄存器	0x0000_0000
USBD_EPIBUFEND	USBD_BA+0x1C8	R/W	端点I RAM 结束地址寄存器	0x0000_0000
USBD_EPJDAT	USBD_BA+0x1CC	R/W	端点 J 数据寄存器	0x0000_0000
USBD_EPJINTSTS	USBD_BA+0x1D0	R/W	端点 J中断状态寄存器	0x0000_0003
USBD_EPJINTEN	USBD_BA+0x1D4	R/W	端点J 中断使能寄存器	0x0000_0000
USBD_EPJDATCNT	USBD_BA+0x1D8	R	端点J 数据有效计数寄存器	0x0000_0000
USBD_EPJRSPCTL	USBD_BA+0x1DC	R/W	端点J 响应控制寄存器	0x0000_0000
USBD_EPJMPS	USBD_BA+0x1E0	R/W	端点J最大包大小寄存器	0x0000_0000
USBD_EPJTXCNT	USBD_BA+0x1E4	R/W	端点J 传输计数寄存器	0x0000_0000
USBD_EPJCFG	USBD_BA+0x1E8	R/W	端点J 配置寄存器	0x0000_00A2
USBD_EPJBUFSTART	USBD_BA+0x1EC	R/W	端点J RAM 开始地址寄存器	0x0000_0000
USBD_EPJBUFEND	USBD_BA+0x1F0	R/W	端点J RAM 结束地址寄存器	0x0000_0000
USBD_EPKDAT	USBD_BA+0x1F4	R/W	端点 K数据寄存器	0x0000_0000

USBD_EPKINTSTS	USBD_BA+0x1F8	R/W	端点 K中断状态寄存器	0x0000_0003
USBD_EPKINTEN	USBD_BA+0x1FC	R/W	端点K 中断使能寄存器	0x0000_0000
USBD_EPKDATCNT	USBD_BA+0x200	R	端点K 数据有效计数寄存器	0x0000_0000
USBD_EPKRSPCTL	USBD_BA+0x204	R/W	端点K 响应控制寄存器	0x0000_0000
USBD_EPKMPS	USBD_BA+0x208	R/W	端点K最大包大小寄存器	0x0000_0000
USBD_EPKTXCNT	USBD_BA+0x20C	R/W	端点K 传输计数寄存器	0x0000_0000
USBD_EPKCFG	USBD_BA+0x210	R/W	端点K 配置寄存器	0x0000_00B2
USBD_EPKBUFSTART	USBD_BA+0x214	R/W	端点K RAM 开始地址寄存器	0x0000_0000
USBD_EPKBUFEND	USBD_BA+0x218	R/W	端点K RAM 结束地址寄存器	0x0000_0000
USBD_EPLDAT	USBD_BA+0x21C	R/W	端点 L数据寄存器	0x0000_0000
USBD_EPLINTSTS	USBD_BA+0x220	R/W	端点 L中断状态寄存器	0x0000_0003
USBD_EPLINTEN	USBD_BA+0x224	R/W	端点L 中断使能寄存器	0x0000_0000
USBD_EPLDATCNT	USBD_BA+0x228	R	端点L 数据有效计数寄存器	0x0000_0000
USBD_EPLRSPCTL	USBD_BA+0x22C	R/W	端点L 响应控制寄存器	0x0000_0000
USBD_EPLMPS	USBD_BA+0x230	R/W	端点L最大包大小寄存器	0x0000_0000
USBD_EPLTXCNT	USBD_BA+0x234	R/W	端点L 传输计数寄存器	0x0000_0000
USBD_EPLCFG	USBD_BA+0x238	R/W	端点L 配置寄存器	0x0000_00C2
USBD_EPLBUFSTART	USBD_BA+0x23C	R/W	端点L RAM 开始地址寄存器	0x0000_0000
USBD_EPLBUFEND	USBD_BA+0x240	R/W	端点L RAM 结束地址寄存器	0x0000_0000
USBD_UVCHDAT0	USBD_BA+0x248	R/W	USB 头字0	0x0000_0000
USBD_UVCHDAT1	USBD_BA+0x24C	R/W	USB 头字1	0x0000_0000
USBD_UVCHDAT2	USBD_BA+0x250	R/W	USB 头字2	0x0000_0000
USBD_UVCEPAHCNT	USBD_BA+0x254	R/W	端点 A 头计数	0x0000_0000
USBD_UVCEPBHCNT	USBD_BA+0x258	R/W	端点 B头计数	0x0000_0000
USBD_UVCEPCHCNT	USBD_BA+0x25C	R/W	端点 C 头计数	0x0000_0000
USBD_UVCEPDHCNT	USBD_BA+0x260	R/W	端点 D头计数	0x0000_0000
USBD_UVCEPEHCNT	USBD_BA+0x264	R/W	端点 E 头计数	0x0000_0000
USBD_UVCEPFHCNT	USBD_BA+0x268	R/W	端点 F 头计数	0x0000_0000
USBD_UVCEPGHCNT	USBD_BA+0x26C	R/W	端点 G 头计数	0x0000_0000
USBD_UVCEPHHCNT	USBD_BA+0x270	R/W	端点 H 头计数	0x0000_0000
USBD_UVCEPIHCNT	USBD_BA+0x274	R/W	端点 I 头计数	0x0000_0000
USBD_UVCEPJHCNT	USBD_BA+0x278	R/W	端点 J 头计数	0x0000_0000

USBD_UVCEPKHCNT	USBD_BA+0x27C	R/W	端点 K 头计数	0x0000_0000
USBD_UVCEPLHCNT	USBD_BA+0x280	R/W	端点 L 头计数	0x0000_0000
USBD_DMAADDR	USBD_BA+0x700	R/W	AHB DMA 地址寄存器	0x0000_0000
USBD_PHYCTL	USBD_BA+0x704	R/W	USB PHY 控制寄存器	0x0000_0420

6.15.6 寄存器描述

全局中断状态寄存器 (USBD_GINTSTS)

寄存器	偏移地址	R/W	描述	复位值
USBD_GINTSTS	USBD_BA+0x000	R	全局中断状态寄存器	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		EPLIF	EPKIF	EPJIF	EPIIF	EPHIF	EPGIF
7	6	5	4	3	2	1	0
EPFIF	EPEIF	EPDIF	EPCIF	EPBIF	EPAIF	CEPIF	USBIF

位	描述	
[31:14]	**Reserved**	保留.
[13]	**EPLIF**	**端点 L 中断** 当置位，应该读对应的端点L的中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生
[12]	**EPKIF**	**端点 K 中断** 当置位，应该读对应的端点K的中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生
[11]	**EPJIF**	**端点 J 中断** 当置位，应该读对应的端点J的中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生
[10]	**EPIIF**	**端点 I 中断** 当置位，应该读对应的端点I 的中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生
[9]	**EPHIF**	**端点 H 中断** 当置位，应该读对应的端点H的中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生

[8]	**EPGIF**	**端点 G 中断** 当置位，应该读对应的端点G的中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生
[7]	**EPFIF**	**端点 F 中断** 当置位，应该读对应的端点F的中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生
[6]	**EPEIF**	**端点 E 中断** 当置位，应该读对应的端点E的中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生
[5]	**EPDIF**	**端点 D 中断** 当置位，应该读对应的端点D的中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生
[4]	**EPCIF**	**端点 C 中断** 当置位，应该读对应的端点C的中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生
[3]	**EPBIF**	**端点 B 中断** 当置位，应该读对应的端点B的中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生
[2]	**EPAIF**	**端点 A 中断** 当置位，应该读对应的端点A的中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生
[1]	**CEPIF**	**控制端点中断** 该位表示控制端点的中断状态。当置位，应该读控制端点的状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生
[0]	**USBIF**	**USB 中断** 该位表示USB指定事件端点的中断状态。当置位，应该读USB中断状态寄存器来确定中断的发生。 0 = 没有中断事件发生 1 = 相关的中断事件发生

全局中断使能寄存器 (USBD_GINTEN)

寄存器	偏移地址	R/W	描述	复位值
USBD_GINTEN	USBD_BA+0x008	R/W	全局中断使能寄存器	0x0000_0001

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		EPLIEN	EPKIEN	EPJIEN	EPIIEN	EPHIEN	EPGIEN
7	6	5	4	3	2	1	0
EPFIEN	EPEIEN	EPDIEN	EPCIEN	EPBIEN	EPAIEN	CEPIEN	USBIEN

位	描述	
[31:14]	**Reserved**	保留.
[13]	**EPLIEN**	**端点L的中断使能控制** 该位置位使能当该端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能
[12]	**EPKIEN**	**端点K的中断使能控制** 该位置位使能当该端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能
[11]	**EPJIEN**	**端点J的中断使能控制** 该位置位使能当该端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能
[10]	**EPIIEN**	**端点I 的中断使能控制** 该位置位使能当该端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能
[9]	**EPHIEN**	**端点H的中断使能控制** 该位置位使能当该端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能

[8]	**EPGIEN**	**端点G的中断使能控制** 该位置位使能当该端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能
[7]	**EPFIEN**	**端点F的中断使能控制** 该位置位使能当该端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能
[6]	**EPEIEN**	**端点E的中断使能控制** 该位置位使能当该端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能
[5]	**EPDIEN**	**端点D的中断使能控制** 该位置位使能当该端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能
[4]	**EPCIEN**	**端点C的中断使能控制** 该位置位使能当该端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能
[3]	**EPBIEN**	**端点B的中断使能控制** 该位置位使能当该端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能
[2]	**EPAIEN**	**端点A的中断使能控制** 该位置位使能当该端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能
[1]	**CEPIEN**	**控制端点中断使能位** 该位置位使能当控制端点的中断挂起时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能
[0]	**USBIEN**	**USB 中断使能位** 该位置位使能当总线USB事件发生时产生一个局部中断。 0 = 相关中断禁止 1 = 相关中断使能

USB 总线中断状态寄存器 (USBD_BUSINTSTS)

寄存器	偏移地址	R/W	描述	复位值
USBD_BUSINTSTS	USBD_BA+0x010	R/W	USB 总线中断状态寄存器	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							VBUSDETIF
7	6	5	4	3	2	1	0
Reserved	PHYCLKVLDIF	DMADONEIF	HISPDIF	SUSPENDIF	RESUMEIF	RSTIF	SOFIF

位	描述	
[31:9]	**Reserved**	保留.
[8]	**VBUSDETIF**	**VBUS侦测中断状态** 0 = 没有 VBUS 插入 1 = VBUS 插入. 写1清除该位为0
[7]	**Reserved**	保留.
[6]	**PHYCLKVLDIF**	**可用时钟中断** 0 = 可用时钟无效 1 = 来自收发器的可用时钟有效 写1清除该位为0
[5]	**DMADONEIF**	**DMA 完成中断** 0 = 没有 DMA 传输完成 1 = DMA传输完成 写1清除该位为0
[4]	**HISPDIF**	**高速配置** 0 = 没有侦测到有效的高速复位协议 1 = 有效的高速复位协议完成，设备被配置为高速。 写1清除该位为0

[3]	**SUSPENDIF**	**挂起请求** 该位默认是1，必须在USB复位之前写1清除该位。当来自主机的USB挂起请求被侦测到，该位也会置位。 0 =没有来自主机的USB挂起请求被侦测到 1= 来自主机的USB挂起请求被侦测到 写1清除该位为0.
[2]	**RESUMEIF**	**恢复** 当该位置1表示设备恢复发生 0 = 没发生设备恢复 1 = 发生设备恢复 写1清除该位为0
[1]	**RSTIF**	**复位状态** 该位置位表示任意一个USB根端口复位结束 0 = 没有USB根端口复位结束 1 = USB根端口复位结束 写1清除该位为0
[0]	**SOFIF**	**SOF接收控制** 该位表示接收到起始帧包 0 = 没有接收到起始帧包 1 = 接收到起始帧包 写1清除该位为0

USB 总线中断使能寄存器 (USBD_BUSINTEN)

寄存器	偏移地址	R/W	描述	复位值
USBD_BUSINTEN	USBD_BA+0x014	R/W	USB 总线中断使能寄存器	0x0000_0040

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							VBUSDETIEN
7	6	5	4	3	2	1	0
Reserved	PHYCLKVLDIEN	DMADONEIEN	HISPDIEN	SUSPENDIEN	RESUMEIEN	RSTIEN	SOFIEN

位	描述	
[31:9]	**Reserved**	保留.
[8]	**VBUSDETIEN**	**VBUS 侦测中断使能控制** 该位使能VBUS 悬空侦测中断 0 = VBUS 悬空侦测中断禁止 1 = VBUS悬空侦测中断使能
[7]	**Reserved**	保留.
[6]	**PHYCLKVLDIEN**	**可用时钟中断** 该位使能可用时钟中断 0 = 可用时钟中断禁止 1 = 可用时钟中断使能
[5]	**DMADONEIEN**	**DMA 完成中断** 该位使能DMA完成中断 0 = DMA 完成中断禁止 1 = DMA完成中断使能
[4]	**HISPDIEN**	**高速配置** 该位使能高速配置中断 0 = 高速配置中断禁止 1 = 高速配置中断使能
[3]	**SUSPENDIEN**	**挂起请求** 该位使能挂起中断 0 = 挂起中断禁止 1 = 挂起中断使能

[2]	**RESUMEIEN**	**恢复** 该位使能恢复中断 0 = 恢复中断禁止. 1 = 恢复中断使能.
[1]	**RSTIEN**	**复位状态** 该位使能USB复位中断 0 = USB复位中断禁止 1 = USB复位中断使能
[0]	**SOFIEN**	**SOF 中断** 该位使能 SOF 中断. 0 = SOF 中断禁止. 1 = SOF 中断使能

USB 操作寄存器 (USBD_OPER)

寄存器	偏移地址	R/W	描述	复位值
USBD_OPER	USBD_BA+0x018	R/W	USB 操作寄存器	0x0000_0002

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					CURSPD	HISPDEN	RESUMEEN

位	描述	
[31:3]	Reserved	保留.
[2]	CURSPD	当前 USB 速度 0 = 设备配置成了全速 1 = USB控制器配置成了高速
[1]	HISPDEN	USB 高速 0 =在复位协议期间， USB设备控制器压制chirp-sequence，因此允许USB设备控制器配置成全速模式，即使它连到USB2.0主机。. 1 =在复位协议期间， USB设备控制器初始化一个chirp-sequence
[0]	RESUMEEN	产生恢复事件 0 = 没有恢复顺序被初始化到主机 1 =.如果设备远程唤醒使能，一个恢复顺序被初始化到主机。该位自我清除。

USB 帧计数寄存器 (USBD_FRAMECNT)

寄存器	偏移地址	R/W	描述	复位值
USBD_FRAMECNT	USBD_BA+0x01C	R	USB 帧计数寄存器	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		FRAMECNT					
7	6	5	4	3	2	1	0
FRAMECNT					MFRAMECNT		

位	描述	
[31:14]	Reserved	保留.
[13:3]	FRAMECNT	**帧计数器** 该域包含从最近的起始帧包开始的帧计数
[2:0]	MFRAMECNT	**微帧计数** 该域包含在帧计数器域的微帧数量

USB 功能地址寄存器 (USBD_FADDR)

寄存器	偏移地址	R/W	描述	复位值
USBD_FADDR	USBD_BA+0x020	R/W	USB 功能地址寄存器	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	FADDR						

位	描述	
[31:7]	Reserved	保留.
[6:0]	FADDR	USB 功能地址 该域包含当前USB设备地址 。当侦测到一个根端口复位，该域被清零。

USB 测试模式寄存器 (USBD_TEST)

寄存器	偏移地址	R/W	描述	复位值
USBD_TEST	USBD_BA+0x024	R/W	USB 测试模式寄存器	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					**TESTMODE**		

位	描述	
[31:3]	**Reserved**	保留.
[2:0]	**TESTMODE**	**测试模式选择** 000 = 正常操作. 001 = Test_J. 010 = Test_K. 011 = Test_SE0_NAK. 100 = Test_Packet. 101 = Test_Force_Enable. 110 = 保留. 111 = 保留. **注:** 当侦测到一个根端口复位，该域被清零。.

控制端点数据缓存 (USBD_CEPDAT)

寄存器	偏移地址	R/W	描述	复位值
USBD_CEPDAT	USBD_BA+0x028	R/W	控制端点数据缓存	0x0000_0000

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

位	描述	
[31:0]	**DAT**	**控制端点数据缓存** 控制端点缓存传输中的数据缓存（读或写） 仅支持字或字节访问

控制端点控制寄存器 (USBD_CEPCTL)

寄存器	偏移地址	R/W	描述	复位值
USBD_CEPCTL	USBD_BA+0x02C	R/W	控制端点控制寄存器	0x0000_0000

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

位	描述	
[31:4]	Reserved	保留.
[3]	FLUSH	**CEP-FLUSH 位** 0 =没有包缓存和它对应的USBD_CEPDATCNT寄存器被清除 1 =包缓存和它对应的USBD_CEPDATCNT寄存器被清除，该位自我清除。
[2]	ZEROLEN	**零包长度** 该位仅在自动确认模式有效 0 = 在数据阶段，IN token 的时候，没有零长度包到主机。 1 =在数据阶段，IN token 的时候，USB设备控制器可以发送零长度包到主机 。一旦零长度包被发送，该位清零。因此本地CPU不需要再一次写它清零。
[1]	STALLEN	**Stall 使能控制** 当这个stall置位，这之后，控制端点发送一个stall握手来回应任何IN或OUT token。这典型地用来回应无效/不支持的请求。当该位被置位，NAK清除位必须同时被清除，因为NAK清除位比STALL位有更高的优先级。该位在收到下一个setup-token的时候会自动清除。因此本地CPU不需要再一次写它清零。 0 =不会发送stall握手来回应之后的任何IN或OUT token。 1 =控制端点发送一个stall握手来回应之后的任何IN或OUT token。
[0]	NAKCLR	**NAK控制** 在任何控制传输中，该位扮演重要的角色。 0 =该位通过写0清除，USB设备控制器会以NAK来响应后来的状态阶段，该机制会保持住主机，防止其进入下一个请求，直到本地CPU也准备好处理下一个请求。 1 = 当收到一个setup token ， 该位由USB设备控制器置1。基于这个请求，本地CPU 可以做 它自己的事情，然后清除该位。.

控制端点中断使能控制寄存器 (USBD_CEPINTEN)

寄存器	偏移地址	R/W	描述	复位值
USBD_CEPINTEN	USBD_BA+0x030	R/W	控制端点中断使能控制寄存器	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			BUFEMPTYIEN	BUFFULLIEN	STSDONEIEN	ERRIEN	STALLIEN
7	6	5	4	3	2	1	0
NAKIEN	RXPKIEN	TXPKIEN	PINGIEN	INTKIEN	OUTTKIEN	SETUPPKIEN	SETUPTKIEN

位	描述	
[31:13]	**Reserved**	保留.
[12]	**BUFEMPTYIEN**	**缓存空中断** 0 = 在控制端点，缓存空中断禁止 1= 在控制端点，缓存空中断使能
[11]	**BUFFULLIEN**	**缓存满中断** 0 = 在控制端点，缓存满中断禁止 1= 在控制端点，缓存满中断使能
[10]	**STSDONEIEN**	**状态完成中断** 0 = 在控制端点，状态完成中断禁止 1= 在控制端点，状态完成中断使能
[9]	**ERRIEN**	**USB 错误中断** 0 = 在控制端点，USB错误中断禁止 1= 在控制端点，USB错误中断使能
[8]	**STALLIEN**	**STALL 发送中断** 0 = 在控制端点，STALL发送中断禁止 1= 在控制端点，STALL发送中断使能
[7]	**NAKIEN**	**NAK 发送中断** 0 = 在控制端点，NAK发送中断禁止 1= 在控制端点，NAK发送中断使能

[6]	**RXPKIEN**	**数据包接收中断** 0 = 在控制端点，数据接收中断禁止 1= 在控制端点，数据接收中断使能
[5]	**TXPKIEN**	**数据包发送中断** 0 = 在控制端点，数据包发送中断禁止 1= 在控制端点，数据包发送中断使能
[4]	**PINGIEN**	**Ping Token 中断** 0 = 在控制端点，ping token中断禁止 1= 在控制端点，ping token中断使能
[3]	**INTKIEN**	**In Token 中断** 0 = 在控制端点，IN token中断禁止 1= 在控制端点，IN token中断使能
[2]	**OUTTKIEN**	**Out Token 中断** 0 = 在控制端点，OUT token中断禁止 1= 在控制端点，OUT token中断使能
[1]	**SETUPPKIEN**	**Setup 包中断** 0 = 在控制端点，SETUP 包中断禁止 1= 在控制端点，SETUP 包中断使能
[0]	**SETUPTKIEN**	**Setup Token中断** 0 = 在控制端点，SETUP token中断禁止 1= 在控制端点，SETUP token中断使能

控制端点中断状态 (USBD_CEPINTSTS)

寄存器	偏移地址	R/W	描述	复位值
USBD_CEPINTSTS	USBD_BA+0x034	R/W	控制端点中断状态	0x0000_1800

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			BUFEMPTYIF	BUFFULLIF	STSDONEIF	ERRIF	STALLIF
7	6	5	4	3	2	1	0
NAKIF	RXPKIF	TXPKIF	PINGIF	INTKIF	OUTTKIF	SETUPPKIF	SETUPTKIF

位	描述	
[31:13]	**Reserved**	保留.
[12]	**BUFEMPTYIF**	**缓存空中断** 0 = 控制端点缓存不为空 1 =控制端点缓存空 **注:** 写1清除该位为0.
[11]	**BUFFULLIF**	**缓存满中断** 0 = 控制端点缓存不满 1 =控制端点缓存满 **注:** 写1清除该位为0.
[10]	**STSDONEIF**	**状态完成中断** 0 = 没有USB传输成功完成 1 = USB传输的状态阶段已经成功完成 **注:** 写1清除该位为0.
[9]	**ERRIF**	**USB错误中断** 0 = 传输过程中，没有错误发生 1 =传输过程中，有错误发生 **注:** 写1清除该位为0.
[8]	**STALLIF**	**STALL 发送中断** 0 = 没有 stall-token被发送来回应IN/OUT token. 1 =一个 stall-token被发送来回应IN/OUT token. **注:** 写1清除该位为0.

[7]	**NAKIF**	**NAK 发送中断** 0 = 没有NAK -token被发送来回应IN/OUT token. 1 =一个NAK -token被发送来回应IN/OUT token. **注:** 写1清除该位为0.
[6]	**RXPKIF**	**数据包接收中断** 0 = 对于一个OUT-token，没有从主机成功接收数据包和发送ACK到主机 1 = 对于一个OUT-token，从主机成功接收到数据包且发送ACK到主机 **注:** 写1清除该位为0.
[5]	**TXPKIF**	**数据包发送中断** 0 = 对于一个IN -token，没有数据包成功发送到主机和收到ACK-token 1 = 对于一个IN -token，数据包成功发送到主机且收到ACK-token **注:** 写1清除该位为0.
[4]	**PINGIF**	**Ping Token 中断** 0 = 控制端点没有接收到来自主机的ping token 1 =控制端点接收到来自主机的ping token **注:** 写1清除该位为0.
[3]	**INTKIF**	**In Token 中断** 0 =控制端点没有接收到来自主机的IN token 1 =控制端点接收到来自主机的IN token **注:** 写1清除该位为0.
[2]	**OUTTKIF**	**Out Token 中断** 0 =控制端点没有接收到来自主机的OUT token 1 =控制端点接收到来自主机的OUT token **注:** 写1清除该位为0.
[1]	**SETUPPKIF**	**Setup 包中断** 在下一个setup包可以被接收之前，该位必须清0（通过写1）。如果该位没有被清零，在setup包缓存中，连续的setup包会被覆盖。 0 =没有接收到来自主机的setup 包 1 =接收到来自主机的setup 包 **注:** 写1清除该位为0.
[0]	**SETUPTKIF**	**Setup Token 中断** 0 = 没有setup token 被接收到. 1 = setup token 被接收到，写1清除该状态位 **注:** 写1清除该位为0.

控制端点IN传输数据计数 (USBD_CEPTXCNT)

寄存器	偏移地址	R/W	描述	复位值
USBD_CEPTXCNT	USBD_BA+0x038	R/W	控制端点IN传输数据计数	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**
TXCNT							

位	描述	
[31:8]	**Reserved**	保留.
[7:0]	**TXCNT**	**In-传输数据计数** 对于控制端点没有模式选择（但是它操作像手动模式）。对于in-token，本地CPU必须填要发送的数据到控制端点缓存，并写字节个数到该寄存器。当零写入该域，一个零长度包被发送到主机。当写入该寄存器的个数大于MPS，仅发送MPS个数据。

控制端点OUT传输数据计数 (USBD_CEPRXCNT)

寄存器	偏移地址	R/W	描述	复位值
USBD_CEPRXCNT	USBD_BA+0x03C	R	控制端点OUT传输数据计数	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**
RXCNT							

位	描述	
[31:8]	**Reserved**	保留.
[7:0]	**RXCNT**	**Out-传输数据计数** 在控制传输期间，out传输的情况下，USB设备控制器保持的接收到数据的个数。

控制端点数据计数 (USBD_CEPDATCNT)

寄存器	偏移地址	R/W	描述	复位值
USBD_CEPDATCNT	USBD_BA+0x040	R	控制端点数据计数	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
DATCNT							
7	6	5	4	3	2	1	0
DATCNT							

位	描述	
[31:16]	Reserved	保留.
[15:0]	DATCNT	**控制端点数据计数** USB设备控制器保持的控制端点的数据个数

Setup1 & Setup0 字节 (USBD_SETUP1_0)

寄存器	偏移地址	R/W	描述	复位值
USBD_SETUP1_0	USBD_BA+0x044	R	Setup1 & Setup0 字节	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**
SETUP1							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETUP0							

位	描述	
[31:16]	**Reserved**	保留.
[15:8]	**SETUP1**	**Setup 字节 1[15:8]** 该寄存器提供最后一次收到的setup包的字节1。对于标准的设备请求，下面请求码信息被返回。 00000000 = Get Status. 00000001 = Clear Feature. 00000010 = 保留 00000011 = Set Feature. 00000100 = 保留. 00000101 = Set Address. 00000110 = Get Descriptor. 00000111 = Set Descriptor. 00001000 = Get Configuration. 00001001 = Set Configuration. 00001010 = Get Interface. 00001011 = Set Interface. 00001100 = Synch Frame.

[7:0]	**SETUP0**	**Setup 字节 0[7:0]** 该寄存器提供最后一次收到的setup包的字节0。对于标准的设备请求，下面请求类型信息被返回。 Bit 7(Direction): 0 = Host to device. 1 = Device to host. Bit 6-5 (Type): 00 = Standard. 01 = Class. 10 = Vendor. 11 = 保留. Bit 4-0 (Recipient) 00000 = Device. 00001 = nterface. 00010 = Endpoint. 00011 =:其它. 其他 =:保留.

Setup3 & Setup2 字节 (USBD_SETUP3_2)

寄存器	偏移地址	R/W	描述	复位值
USBD_SETUP3_2	USBD_BA+0x048	R	Setup3 & Setup2 字节	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
SETUP3							
7	6	5	4	3	2	1	0
SETUP2							

位	描述	
[31:16]	Reserved	保留.
[15:8]	SETUP3	**Setup 字节 3 [15:8]** 该寄存器提供最后一次收到的setup包的字节3。对于标准的设备请求，wValue域最高有效字节被返回。
[7:0]	SETUP2	**Setup 字节 2 [7:0]** 该寄存器提供最后一次收到的setup包的字节2 。对于标准的设备请求，wValue域最低有效字节被返回。

Setup5 & Setup4 字节 (USBD_SETUP5_4)

寄存器	偏移地址	R/W	描述	复位值
USBD_SETUP5_4	USBD_BA+0x04C	R	Setup5 & Setup4 字节	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
SETUP5							
7	6	5	4	3	2	1	0
SETUP4							

位	描述	
[31:16]	Reserved	保留.
[15:8]	SETUP5	Setup 字节 5[15:8] 该寄存器提供最后一次收到的setup包的字节5。对于标准的设备请求，wIndex域最高有效字节被返回。
[7:0]	SETUP4	Setup 字节 4[7:0] 该寄存器提供最后一次收到的setup包的字节4。对于标准的设备请求，wIndex域最低有效字节被返回。

Setup7 & Setup6 字节 (USBD_SETUP7_6)

寄存器	偏移地址	R/W	描述	复位值
USBD_SETUP7_6	USBD_BA+0x050	R	Setup7 & Setup6 字节	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**
SETUP7							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SETUP6							

位	描述	
[31:16]	**Reserved**	保留.
[15:8]	**SETUP7**	**Setup 字节 7[15:8]** 该寄存器提供最后一次收到的setup包的字节7。对于标准的设备请求，wLength域最高有效字节被返回。
[7:0]	**SETUP6**	**Setup 字节 6[7:0]** 该寄存器提供最后一次收到的setup包的字节6。对于标准的设备请求，wLength域最低有效字节被返回。

控制端点RAM开始地址寄存器 (USBD_CEPBUFSTART)

寄存器	偏移地址	R/W	描述	复位值
USBD_CEPBUFSTART	USBD_BA+0x054	R/W	控制端点RAM开始地址寄存器	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				SADDR			
7	6	5	4	3	2	1	0
SADDR							

位	描述	
[31:12]	Reserved	保留.
[11:0]	SADDR	**控制端点开始地址** 对于控制端点，这是RAM空间分配的开始地址

控制端点RAM 结束地址寄存器 (USBD_CEPBUFEND)

寄存器	偏移地址	R/W	描述	复位值
USBD_CEPBUFEND	USBD_BA+0x058	R/W	控制端点 RAM 结束地址寄存器	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				EADDR			
7	6	5	4	3	2	1	0
EADDR							

位	描述	
[31:12]	Reserved	保留.
[11:0]	EADDR	**控制端点结束地址** 对于控制端点，这是RAM空间分配的结束地址

DMA 控制状态寄存器 (USBD_DMACTL)

寄存器	偏移地址	R/W	描述	复位值
USBD_DMACTL	USBD_BA+0x05C	R/W	DMA 控制状态寄存器	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
DMARST	SGEN	DMAEN	DMARD	EPNUM			

位	描述	
[31:8]	Reserved	保留.
[7]	DMARST	**复位 DMA 状态机** 0 = 没有复位DMA状态机 1 = 复位DMA状态机
[6]	SGEN	**Scatter Gather功能使能** 0 = Scatter gather 功能禁止 1 = Scatter gather 功能使能
[5]	DMAEN	**DMA 使能位** 0 = DMA 功能禁止 1 = DMA 功能使能
[4]	DMARD	**DMA 操作** 0 =操作是DMA写（从USB缓存读）。在执行DMA写操作之前，DMA会根据EPNM的设置检查端点数据有效计数(USBD_EPxDATCNT)。 1 = 操作是 DMA 读 (写数据到 USB 缓存).
[3:0]	EPNUM	**DMA 端点地址位** 用来定义端点地址

DMA 计数寄存器 (USBD_DMACNT)

寄存器	偏移地址	R/W	描述	复位值
USBD_DMACNT	USBD_BA+0x060	R/W	DMA 计数寄存器	0x0000_0000

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

位	描述	
[31:20]	Reserved	保留.
[19:0]	DMACNT	DMA 传输计数 要执行的DMA操作的传输计数写入该寄存器

端点 A~L 数据寄存器 (USBD_EPADAT~ USBD_EPLDAT)

寄存器	偏移地址	R/W	描述	复位值
USBD_EPADAT	USBD_BA+0x064	R/W	端点 A 数据寄存器	0x0000_0000
USBD_EPBDAT	USBD_BA+0x08C	R/W	端点 B 数据寄存器	0x0000_0000
USBD_EPCDAT	USBD_BA+0x0B4	R/W	端点 C 数据寄存器	0x0000_0000
USBD_EPDDAT	USBD_BA+0x0DC	R/W	端点 D 数据寄存器	0x0000_0000
USBD_EPEDAT	USBD_BA+0x104	R/W	端点 E 数据寄存器	0x0000_0000
USBD_EPFDAT	USBD_BA+0x12C	R/W	端点 F 数据寄存器	0x0000_0000
USBD_EPGDAT	USBD_BA+0x154	R/W	端点 G 数据寄存器	0x0000_0000
USBD_EPHDAT	USBD_BA+0x17C	R/W	端点 H 数据寄存器	0x0000_0000
USBD_EPIDAT	USBD_BA+0x1A4	R/W	端点 I 数据寄存器	0x0000_0000
USBD_EPJDAT	USBD_BA+0x1CC	R/W	端点 J 数据寄存器	0x0000_0000
USBD_EPKDAT	USBD_BA+0x1F4	R/W	端点 K 数据寄存器	0x0000_0000
USBD_EPLDAT	USBD_BA+0x21C	R/W	端点 L 数据寄存器	0x0000_0000

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

位	描述	
[31:0]	**EPDAT**	**端点 A~L 数据寄存器** 对于缓存处理事务，端点 A~L 数据缓存 (读或写). 仅支持字或字节访问

端点 A~L 中断状态寄存器 (USBD_EPAINTSTS~ USBD_EPLINTSTS)

寄存器	偏移地址	R/W	描述	复位值
USBD_EPAINTSTS	USBD_BA+0x068	R/W	端点 A 中断状态寄存器	0x0000_0003
USBD_EPBINTSTS	USBD_BA+0x090	R/W	端点 B 中断状态寄存器	0x0000_0003
USBD_EPCINTSTS	USBD_BA+0x0B8	R/W	端点 C 中断状态寄存器	0x0000_0003
USBD_EPDINTSTS	USBD_BA+0x0E0	R/W	端点 D 中断状态寄存器	0x0000_0003
USBD_EPEINTSTS	USBD_BA+0x108	R/W	端点 E 中断状态寄存器	0x0000_0003
USBD_EPFINTSTS	USBD_BA+0x130	R/W	端点 F 中断状态寄存器	0x0000_0003
USBD_EPGINTSTS	USBD_BA+0x158	R/W	端点 G 中断状态寄存器	0x0000_0003
USBD_EPHINTSTS	USBD_BA+0x180	R/W	端点 H 中断状态寄存器	0x0000_0003
USBD_EPIINTSTS	USBD_BA+0x1A8	R/W	端点 I 中断状态寄存器	0x0000_0003
USBD_EPJINTSTS	USBD_BA+0x1D0	R/W	端点 J 中断状态寄存器	0x0000_0003
USBD_EPKINTSTS	USBD_BA+0x1F8	R/W	端点 K 中断状态寄存器	0x0000_0003
USBD_EPLINTSTS	USBD_BA+0x220	R/W	端点 L 中断状态寄存器	0x0000_0003

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			SHORTRXIF	ERRIF	NYETIF	STALLIF	NAKIF
7	6	5	4	3	2	1	0
PINGIF	INTKIF	OUTTKIF	RXPKIF	TXPKIF	SHORTTXIF	BUFEMPTYIF	BUFFULLIF

位	描述	
[31:13]	**Reserved**	保留.
[12]	**SHORTRXIF**	**批量输出短包接收** 0 = 没有批量输出短包被接收到 1 = 接收到批量输出短包 (包括零长度包). 写1清除该位为0

[11]	**ERRIF**	**ERR 发送** 0 = 传输中没有任何错误 1 = 传输中发生错误 写1清除该位为0
[10]	**NYETIF**	**NYET 发送** 0 = 在RAM中的有效空间是充足的，来容纳下次即将到来的数据包 1 =在RAM中的有效空间不是充足的，来容纳下次即将到来的数据包 写1清除该位为0
[9]	**STALLIF**	**USB STALL 发送** 0 =最后的USB包可以被接收 1 =最后的USB包不可以被接收. 写1清除该位为0
[8]	**NAKIF**	**USB NAK 发送** 0 = 最后的USB IN包可以被提供 1 = 最后的USB IN包不可以被提供. 写1清除该位为0
[7]	**PINGIF**	**PING Token 中断** 0 = 来自主机的数据 PING token 还没有被接收到 1 = 来自主机的数据 PING token被接收到 写1清除该位为0
[6]	**INTKIF**	**数据 IN Token 中断** 0 =没有来自主机的数据 IN token 被接收到 . 1 =来自主机的数据 IN token 被接收到 写1清除该位为0
[5]	**OUTTKIF**	**数据** 0 =没有来自主机的数据 OUT token 被接收到 1 =来自主机的数据 OUT token 被接收到。 该位也由PING token置位（仅在高速模式） 写1清除该位为0
[4]	**RXPKIF**	**数据包接收中断** 0 = 没有来自主机的数据包通过端点被接收到 1 = 来自主机的数据包通过端点被接收到 写1清除该位为0
[3]	**TXPKIF**	**数据包发送中断** 0 = 没有数据包被从端点到主机发送 1 = 数据包被从端点到主机发送 写1清除该位为0

[2]	**SHORTTXIF**	**短包传输中断** 0 =最后包长度不小于最大包大小(EPMPS). 1 =最后包长度小于最大包大小(EPMPS). 写1清除该位为0
[1]	**BUFEMPTYIF**	**缓存空** 对于IN端点，所有FIFO字节都是可用的 0 =端点缓存不为空 1 = 端点缓存空 对于 OUT 端点: 0 = 最近选择的缓存还没有数到0 1 =最近选择的缓存数到0，本地没有缓存可用 (没东西可读). **注:**该位只读
[0]	**BUFFULLIF**	**缓存满** 对于IN端点，当前选择的缓存满，或没有缓存可用给本地端写（没有空间写）。对于OUT端点，在本地端有缓存可用，且有FIFO所有字节可用于读（整个包都是可用于读的） 0 = 端点包缓存不是满的 1 =端点包缓存是满的. **注:**该位只读

端点 A~L 中断使能控制寄存器 (USBD_EPAINTEN~ USBD_EPLINTEN)

寄存器	偏移地址	R/W	描述	复位值
USBD_EPAINTEN	USBD_BA+0x06C	R/W	端点 A 中断使能寄存器	0x0000_0000
USBD_EPBINTEN	USBD_BA+0x094	R/W	端点 B 中断使能寄存器	0x0000_0000
USBD_EPCINTEN	USBD_BA+0x0BC	R/W	端点 C 中断使能寄存器	0x0000_0000
USBD_EPDINTEN	USBD_BA+0x0E4	R/W	端点 D 中断使能寄存器	0x0000_0000
USBD_EPEINTEN	USBD_BA+0x10C	R/W	端点 E 中断使能寄存器	0x0000_0000
USBD_EPFINTEN	USBD_BA+0x134	R/W	端点 F 中断使能寄存器	0x0000_0000
USBD_EPGINTEN	USBD_BA+0x15C	R/W	端点 G中断使能寄存器	0x0000_0000
USBD_EPHINTEN	USBD_BA+0x184	R/W	端点 H 中断使能寄存器	0x0000_0000
USBD_EPIINTEN	USBD_BA+0x1AC	R/W	端点 I 中断使能寄存器	0x0000_0000
USBD_EPJINTEN	USBD_BA+0x1D4	R/W	端点 J 中断使能寄存器	0x0000_0000
USBD_EPKINTEN	USBD_BA+0x1FC	R/W	端点 K 中断使能寄存器	0x0000_0000
USBD_EPLINTEN	USBD_BA+0x224	R/W	端点 L 中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved			SHORTRXIEN	ERRIEN	NYETIEN	STALLIEN	NAKIEN
7	6	5	4	3	2	1	0
PINGIEN	INTKIEN	OUTTKIEN	RXPKIEN	TXPKIEN	SHORTTXIEN	BUFEMPTYIEN	BUFFULLIEN

位	描述	
[31:13]	Reserved	保留.
[12]	SHORTRXIEN	**批量输出短包中断使能位** 当置位，对于该端点，总线上批量输出短包发生的时候，该位使能本地中断置位。 0 = 批量输出中断禁止 1 = 批量输出中断使能

[11]	**ERRIEN**	**ERR 中断使能位** 当置位，对于该端点，总线上错误条件发生的时候，该位使能本地中断置位。 0 = 错误事件中断禁止 1 = 错误事件中断使能
[10]	**NYETIEN**	**NYET中断使能位** 当置位，对于该端点，总线上NYET条件发生的时候，该位使能本地中断置位。 0 = NYET条件中断禁止 1 = NYET条件中断使能
[9]	**STALLIEN**	**USB STALL 发送中断使能位** 当该位置位，当stall token发送到主机，该位使能本地中断置位。 0 = STALL token 中断禁止 1 = STALL token 中断使能
[8]	**NAKIEN**	**USB NAK 发送中断使能位** 当该位置位，当NAK token发送到主机，该位使能本地中断置位。 0 = NAK token中断禁止. 1 = NAK token中断使能
[7]	**PINGIEN**	**PING Token中断使能位** 当该位置位，当来自主机的PING token被接收到，该位使能本地中断置位。 0 = PING token 中断禁止 1 = PING token 中断使能
[6]	**INTKIEN**	**数据 IN Token 中断使能位** 当该位置位，当来自主机的IN token被接收到，该位使能本地中断置位。 0 = 数据 IN token中断禁止. 1 = 数据 IN token中断使能.
[5]	**OUTTKIEN**	**数据 OUT Token中断使能位** 当该位置位，当来自主机的数据OUT token被接收到，该位使能本地中断置位。 0 = 数据 OUT token 中断禁止 1 = 数据 OUT token中断使能.
[4]	**RXPKIEN**	**数据包接收中断使能位** 当该位置位，当数据包被发送到主机，该位使能本地中断置位。 0 = 数据包发送到主机中断禁止 1 = 数据包发送到主机中断使能
[3]	**TXPKIEN**	**数据包发送中断使能位** 当该位置位，当来自主机的数据包被接收，该位使能本地中断置位。 0 = 来自主机的数据包接收中断禁止 1 = 来自主机的数据包接收中断使能

[2]	**SHORTTXIEN**	**短包传输中断使能位** 当该位置位，当一个短数据包已经传输到主机或从主机接收到 ，该位使能本地中断置位。 0 = 短数据包中断禁止 1 =短数据包中断使能
[1]	**BUFEMPTYIEN**	**缓存空中断** 当该位置位，当总线侦测到缓存空条件 ，该位使能本地中断置位。 0 = 缓存空中断禁止 1 = 缓存空中断使能.
[0]	**BUFFULLIEN**	**缓存满中断** 当该位置位，当总线侦测到缓存满条件 ，该位使能本地中断置位。 0 = 缓存满中断禁止 1 = 缓存满中断使能.

端点 A~L 数据可用计数寄存器 (USBD_EPADATCNT~ USBD_EPLDATCNT)

寄存器	偏移地址	R/W	描述	复位值
USBD_EPADATCNT	USBD_BA+0x070	R	端点 A 数据可用计数寄存器	0x0000_0000
USBD_EPBDATCNT	USBD_BA+0x098	R	端点 B 数据可用计数寄存器	0x0000_0000
USBD_EPCDATCNT	USBD_BA+0x0C0	R	端点 C 数据可用计数寄存器	0x0000_0000
USBD_EPDDATCNT	USBD_BA+0x0E8	R	端点 D 数据可用计数寄存器	0x0000_0000
USBD_EPEDATCNT	USBD_BA+0x110	R	端点 E 数据可用计数寄存器	0x0000_0000
USBD_EPFDATCNT	USBD_BA+0x138	R	端点 F 数据可用计数寄存器	0x0000_0000
USBD_EPGDATCNT	USBD_BA+0x160	R	端点 G 数据可用计数寄存器	0x0000_0000
USBD_EPHDATCNT	USBD_BA+0x188	R	端点 H 数据可用计数寄存器	0x0000_0000
USBD_EPIDATCNT	USBD_BA+0x1B0	R	端点 I 数据可用计数寄存器	0x0000_0000
USBD_EPJDATCNT	USBD_BA+0x1D8	R	端点 J 数据可用计数寄存器	0x0000_0000
USBD_EPKDATCNT	USBD_BA+0x200	R	端点 K 数据可用计数寄存器	0x0000_0000
USBD_EPLDATCNT	USBD_BA+0x228	R	端点 L 数据可用计数寄存器	0x0000_0000

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

位	描述	
[31]	**Reserved**	保留.
[30:16]	**DMALOOP**	**DMA 循环** 该寄存器保持DMA循环到完成，每一个循环意味着32个字节传输
[15:0]	**DATCNT**	**数据计数** 对于IN端点(EPDIR(USBD_EPxCFG[3] 为高.)，该寄存器返回IN端点包缓存的有效字节的数量 对于OUT端点(EPDIR(USBD_EPxCFG[3] 为低.)，该寄存器返回主机输出传输中接收到有效字节个数。

端点 A~L 响应控制寄存器 (USBD_EPARSPCTL~ USBD_EPLRSPCTL)

寄存器	偏移地址	R/W	描述	复位值
USBD_EPARSPCTL	USBD_BA+0x074	R/W	端点 A 响应控制寄存器	0x0000_0000
USBD_EPBRSPCTL	USBD_BA+0x09C	R/W	端点 B 响应控制寄存器	0x0000_0000
USBD_EPCRSPCTL	USBD_BA+0x0C4	R/W	端点 C 响应控制寄存器	0x0000_0000
USBD_EPDRSPCTL	USBD_BA+0x0EC	R/W	端点 D 响应控制寄存器	0x0000_0000
USBD_EPERSPCTL	USBD_BA+0x114	R/W	端点 E 响应控制寄存器	0x0000_0000
USBD_EPFRSPCTL	USBD_BA+0x13C	R/W	端点 F 响应控制寄存器	0x0000_0000
USBD_EPGRSPCTL	USBD_BA+0x164	R/W	端点 G 响应控制寄存器	0x0000_0000
USBD_EPHRSPCTL	USBD_BA+0x18C	R/W	端点 H 响应控制寄存器	0x0000_0000
USBD_EPIRSPCTL	USBD_BA+0x1B4	R/W	端点 I 响应控制寄存器	0x0000_0000
USBD_EPJRSPCTL	USBD_BA+0x1DC	R/W	端点 J 响应控制寄存器	0x0000_0000
USBD_EPKRSPCTL	USBD_BA+0x204	R/W	端点 K 响应控制寄存器	0x0000_0000
USBD_EPLRSPCTL	USBD_BA+0x22C	R/W	端点 L 响应控制寄存器	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
DISBUF	SHORTTXEN	ZEROLEN	HALT	TOGGLE	MODE		FLUSH

位	描述	
[31:8]	Reserved	保留.
[7]	DISBUF	**缓存禁止位** 该位用来接收未知大小OUT短包。接收到的包大小参考USBD_EPxDATCNT寄存器 0 =.当批量输出短包被接收到缓存没有禁止 1 =. 当批量输出短包被接收到缓存禁止

[6]	**SHORTTXEN**	**短包传输使能位** 该位仅用于自动确认方法的情况。该位置位用来确认缓存中的任何不等于端点MPS的剩余数据，并且发送在最近一次传输。一旦数据包被发送，该位清除。 0 =没有确认缓存中的任何不等于端点MPS的剩余数据， 1 =确认缓存中的任何不等于端点MPS的剩余数据，
[5]	**ZEROLEN**	**零长度** 该位用来发送一个0长度包来响应IN-token。当该位置1，在接收到IN-token的时候，一个0长度包发送到主机。一旦0长度数据包被发送，该位清0。 0 =在接收到IN-token的时候，0长度包没有发送到主机。 1 =在接收到IN-token的时候，0长度包发送到主机。
[4]	**HALT**	**端点暂停** 该位用来发送STALL握手来响应来自主机的token。当端点Set Feature (ep_halt) 被本地CPU侦测到，必须写1到该位。 0 =不发送STALL握手来响应来自主机的token。 1 =发送STALL握手来响应来自主机的token。
[3]	**TOGGLE**	**端点 Toggle** 该位用来清除端点数据toggle位。读该位返回当前端点数据toggle位的状态。 在接收到来自主机的Set Interface 请求 或 Clear Feature (ep_halt)请求的情况下，本地CPU可以使用该位来初始化端点toggle。仅当toggle位是1，该位才可以被写进反写数据bit[3].。 0 = 不清除端点数据toggle 位 1 =清除端点数据toggle 位
[2:1]	**MODE**	**模式控制** 这两位决定IN端点的操作模式 00 = 自动确认模式 01 = 手动确认模式 10 = Fly 模式. 11 = 保留. **注:**对于OUT端点，这些位是无效的。当保留模式被选择，自动确认模式会被激活。
[0]	**FLUSH**	**缓存清除** 写1到该位会引起包缓存被清除，且对应的EP_AVAIL 寄存器被清除。该位自我清除。该位应该在一个配置事件之后写 0 = 包缓存没有被清除 1 =包缓存被用户清除

端点 A~L 最大包大小寄存器 (USBD_EPAMPS~ USBD_EPLMPS)

寄存器	偏移地址	R/W	描述	复位值
USBD_EPAMPS	USBD_BA+0x078	R/W	端点 A 最大包大小寄存器	0x0000_0000
USBD_EPBMPS	USBD_BA+0x0A0	R/W	端点 B 最大包大小寄存器	0x0000_0000
USBD_EPCMPS	USBD_BA+0x0C8	R/W	端点 C 最大包大小寄存器	0x0000_0000
USBD_EPDMPS	USBD_BA+0x0F0	R/W	端点 D 最大包大小寄存器	0x0000_0000
USBD_EPEMPS	USBD_BA+0x118	R/W	端点 E 最大包大小寄存器	0x0000_0000
USBD_EPFMPS	USBD_BA+0x140	R/W	端点 F 最大包大小寄存器	0x0000_0000
USBD_EPGMPS	USBD_BA+0x168	R/W	端点 G 最大包大小寄存器	0x0000_0000
USBD_EPHMPS	USBD_BA+0x190	R/W	端点 H 最大包大小寄存器	0x0000_0000
USBD_EPIMPS	USBD_BA+0x1B8	R/W	端点 I 最大包大小寄存器	0x0000_0000
USBD_EPJMPS	USBD_BA+0x1E0	R/W	端点 J 最大包大小寄存器	0x0000_0000
USBD_EPKMPS	USBD_BA+0x208	R/W	端点 K 最大包大小寄存器	0x0000_0000
USBD_EPLMPS	USBD_BA+0x230	R/W	端点 L 最大包大小寄存器	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					**EPMPS**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPMPS							

位	描述	
[31:11]	**Reserved**	保留.
[10:0]	**EPMPS**	**端点最大包大小** 该域定义了端点的最大包大小

端点 A~L传输计数寄存器 (USBD_EPATXCNT~ USBD_EPLTXCNT)

寄存器	偏移地址	R/W	描述	复位值
USBD_EPATXCNT	USBD_BA+0x07C	R/W	端点 A 传输计数寄存器	0x0000_0000
USBD_EPBTXCNT	USBD_BA+0x0A4	R/W	端点 B 传输计数寄存器	0x0000_0000
USBD_EPCTXCNT	USBD_BA+0x0CC	R/W	端点 C 传输计数寄存器	0x0000_0000
USBD_EPDTXCNT	USBD_BA+0x0F4	R/W	端点 D 传输计数寄存器	0x0000_0000
USBD_EPETXCNT	USBD_BA+0x11C	R/W	端点 E 传输计数寄存器	0x0000_0000
USBD_EPFTXCNT	USBD_BA+0x144	R/W	端点 F 传输计数寄存器	0x0000_0000
USBD_EPGTXCNT	USBD_BA+0x16C	R/W	端点 G 传输计数寄存器	0x0000_0000
USBD_EPHTXCNT	USBD_BA+0x194	R/W	端点 H 传输计数寄存器	0x0000_0000
USBD_EPITXCNT	USBD_BA+0x1BC	R/W	端点 I 传输计数寄存器	0x0000_0000
USBD_EPJTXCNT	USBD_BA+0x1E4	R/W	端点 J 传输计数寄存器	0x0000_0000
USBD_EPKTXCNT	USBD_BA+0x20C	R/W	端点 K 传输计数寄存器	0x0000_0000
USBD_EPLTXCNT	USBD_BA+0x234	R/W	端点 L 传输计数寄存器	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					**TXCNT**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXCNT							

位	描述	
[31:11]	**Reserved**	保留.
[10:0]	**TXCNT**	**端点传输计数** 对于IN端点，在手动确认方法情况下，该域定义发送到主机的字节总数。 对于 OUT 端点, 该域无效。.

端点 A~L 配置寄存器 (USBD_EPACFG~ USBD_EPLCFG)

寄存器	偏移地址	R/W	描述	复位值
USBD_EPACFG	USBD_BA+0x080	R/W	端点 A 配置寄存器	0x0000_0012
USBD_EPBCFG	USBD_BA+0x0A8	R/W	端点 B 配置寄存器	0x0000_0022
USBD_EPCCFG	USBD_BA+0x0D0	R/W	端点 C 配置寄存器	0x0000_0032
USBD_EPDCFG	USBD_BA+0x0F8	R/W	端点 D 配置寄存器	0x0000_0042
USBD_EPECFG	USBD_BA+0x120	R/W	端点 E 配置寄存器	0x0000_0052
USBD_EPFCFG	USBD_BA+0x148	R/W	端点 F 配置寄存器	0x0000_0062
USBD_EPGCFG	USBD_BA+0x170	R/W	端点 G 配置寄存器	0x0000_0072
USBD_EPHCFG	USBD_BA+0x198	R/W	端点 H 配置寄存器	0x0000_0082
USBD_EPICFG	USBD_BA+0x1C0	R/W	端点 I 配置寄存器	0x0000_0092
USBD_EPJCFG	USBD_BA+0x1E8	R/W	端点 J 配置寄存器	0x0000_00A2
USBD_EPKCFG	USBD_BA+0x210	R/W	端点 K 配置寄存器	0x0000_00B2
USBD_EPLCFG	USBD_BA+0x238	R/W	端点 L 配置寄存器	0x0000_00C2

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						**Reserved**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EPNUM				**EPDIR**	**EPTYPE**		**EPEN**

位	描述	
[31:8]	**Reserved**	保留.
[7:4]	**EPNUM**	**端点数** 该域选择端点号，有效号是1 到 15. 不支持两个端点有相同的端点号
[3]	**EPDIR**	**端点方向** 0 = 输出端点 (主机 OUT 到设备). 1 = 输入端点 (主机 IN 到设备).

[2:1]	**EPTYPE**	**端点类型** 该域选择端点类型。端点0强制为控制类型。 00 = 保留. 01 = 批量. 10 = 中断. 11 = 同步.
[0]	**EPEN**	**端点有效** 当置位，该位使能该端点。该位对端点0无效，端点0总是使能的。 0 = 端点禁止 1 = 端点使能.

端点 A~L RAM 开始地址寄存器 (USBD_EPABUFSTART~ USBD_EPLBUFSTART)

寄存器	偏移地址	R/W	描述	复位值
USBD_EPABUFSTART	USBD_BA+0x084	R/W	端点 A RAM 开始地址寄存器	0x0000_0000
USBD_EPBBUFSTART	USBD_BA+0x0AC	R/W	端点 B RAM 开始地址寄存器	0x0000_0000
USBD_EPCBUFSTART	USBD_BA+0x0D4	R/W	端点 C RAM 开始地址寄存器	0x0000_0000
USBD_EPDBUFSTART	USBD_BA+0x0FC	R/W	端点 D RAM 开始地址寄存器	0x0000_0000
USBD_EPEBUFSTART	USBD_BA+0x124	R/W	端点 E RAM 开始地址寄存器	0x0000_0000
USBD_EPFBUFSTART	USBD_BA+0x14C	R/W	端点 F RAM 开始地址寄存器	0x0000_0000
USBD_EPGBUFSTART	USBD_BA+0x174	R/W	端点 G RAM 开始地址寄存器	0x0000_0000
USBD_EPHBUFSTART	USBD_BA+0x19C	R/W	端点 H RAM 开始地址寄存器	0x0000_0000
USBD_EPIBUFSTART	USBD_BA+0x1C4	R/W	端点 I RAM 开始地址寄存器	0x0000_0000
USBD_EPJBUFSTART	USBD_BA+0x1EC	R/W	端点 J RAM 开始地址寄存器	0x0000_0000
USBD_EPKBUFSTART	USBD_BA+0x214	R/W	端点 K RAM 开始地址寄存器	0x0000_0000
USBD_EPLBUFSTART	USBD_BA+0x23C	R/W	端点 L RAM 开始地址寄存器	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				**SADDR**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SADDR							

位	描述	
[31:12]	**Reserved**	保留.
[11:0]	**SADDR**	**端点开始地址** 这是为端点A~L分配RAM空间的起始地址

端点 A~L RAM 结束地址寄存器 (USBD_EPABUFEND~ USBD_EPLBUFEND)

寄存器	偏移地址	R/W	描述	复位值
USBD_EPABUFEND	USBD_BA+0x088	R/W	端点 A RAM 结束地址寄存器	0x0000_0000
USBD_EPBBUFEND	USBD_BA+0x0B0	R/W	端点 B RAM 结束地址寄存器	0x0000_0000
USBD_EPCBUFEND	USBD_BA+0x0D8	R/W	端点 C RAM 结束地址寄存器	0x0000_0000
USBD_EPDBUFEND	USBD_BA+0x100	R/W	端点 D RAM 结束地址寄存器	0x0000_0000
USBD_EPEBUFEND	USBD_BA+0x128	R/W	端点 E RAM 结束地址寄存器	0x0000_0000
USBD_EPFBUFEND	USBD_BA+0x150	R/W	端点 F RAM 结束地址寄存器	0x0000_0000
USBD_EPGBUFEND	USBD_BA+0x178	R/W	端点 G RAM 结束地址寄存器	0x0000_0000
USBD_EPHBUFEND	USBD_BA+0x1A0	R/W	端点 H RAM 结束地址寄存器	0x0000_0000
USBD_EPIBUFEND	USBD_BA+0x1C8	R/W	端点 I RAM 结束地址寄存器	0x0000_0000
USBD_EPJBUFEND	USBD_BA+0x1F0	R/W	端点 J RAM 结束地址寄存器	0x0000_0000
USBD_EPKBUFEND	USBD_BA+0x218	R/W	端点 K RAM 结束地址寄存器	0x0000_0000
USBD_EPLBUFEND	USBD_BA+0x240	R/W	端点 L RAM 结束地址寄存器	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				**EADDR**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
EADDR							

位	描述	
[31:12]	**Reserved**	保留.
[11:0]	**EADDR**	**端点结束地址** 这是为端点A~L分配RAM空间的结束地址

USB 头字0 (USBD_UVCHDAT0)

寄存器	偏移地址	R/W	描述	复位值
USBD_UVCHDAT0	USBD_BA+0x248	R/W	USB 头字0	0x0000_0000

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

位	描述	
[31:0]	DAT	第一个头数据 (字节 0 先被发送)

USB 头字1 (USBD_UVCHDAT1)

寄存器	偏移地址	R/W	描述	复位值
USBD_UVCHDAT1	USBD_BA+0x24C	R/W	USB 头 字1	0x0000_0000

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

位	描述	
[31:0]	DAT	第二个头数据 (字节 0 先被发送)

USB 头字2 (USBD_UVCHDAT2)

寄存器	偏移地址	R/W	描述	复位值
USBD_UVCHDAT2	USBD_BA+0x250	R/W	USB 头字2	0x0000_0000

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

位	描述	
[31:0]	DAT	第三个头数据 (字节 0 先被发送)

端点 A~L 头计数寄存器 (USBD_UVCEPAHCNT~ USBD_UVCEPLCNT)

寄存器	偏移地址	R/W	描述	复位值
USBD_UVCEPAHCNT	USBD_BA+0x254	R/W	端点 A 头计数	0x0000_0000
USBD_UVCEPBHCNT	USBD_BA+0x258	R/W	端点 B 头计数	0x0000_0000
USBD_UVCEPCHCNT	USBD_BA+0x25C	R/W	端点 C 头计数	0x0000_0000
USBD_UVCEPDHCNT	USBD_BA+0x260	R/W	端点 D 头计数	0x0000_0000
USBD_UVCEPEHCNT	USBD_BA+0x264	R/W	端点 E 头计数	0x0000_0000
USBD_UVCEPFHCNT	USBD_BA+0x268	R/W	端点 F 头计数	0x0000_0000
USBD_UVCEPGHCNT	USBD_BA+0x26C	R/W	端点 G 头计数	0x0000_0000
USBD_UVCEPHHCNT	USBD_BA+0x270	R/W	端点 H 头计数	0x0000_0000
USBD_UVCEPIHCNT	USBD_BA+0x274	R/W	端点 I 头计数	0x0000_0000
USBD_UVCEPJHCNT	USBD_BA+0x278	R/W	端点 J 头计数	0x0000_0000
USBD_UVCEPKHCNT	USBD_BA+0x27C	R/W	端点 K 头计数	0x0000_0000
USBD_UVCEPLHCNT	USBD_BA+0x280	R/W	端点 L 头计数	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				**CNT**			

位	描述	
[31:4]	**Reserved**	保留.
[3:0]	**CNT**	**这是端点A~L 的头计数** 头计数必须是偶数

AHB 地址寄存器 (USBD_DMAADDR)

寄存器	偏移地址	R/W	描述	复位值
USBD_DMAADDR	USBD_BA+0x700	R/W	AHB DMA 地址寄存器	0x0000_0000

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

位	描述	
[31:0]	**DMAADDR**	**DMAADDR** 该寄存器指定地址，哪一个DMA必须读/写。地址必须字(32-bit)对齐 。

USB PHY 控制寄存器 (USBD_PHYCTL)

寄存器	偏移地址	R/W	描述	复位值
USBD_PHYCTL	USBD_BA+0x704	R/W	USB PHY 控制寄存器	0x0000_0420

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

位	描述	
[31]	VBUSDET	**VBUS 状态** 0 = VBUS还没有侦测到 1 = VBUS 侦测到
[30:25]	Reserved	保留.
[24]	WKEN	**唤醒使能位** 0 = 唤醒功能禁止. 1 = 唤醒功能使能
[23:10]	Reserved	保留.
[9]	PHYEN	**PHY 挂起使能位** 0 = USB PHY 挂起 1 = USB PHY 没有挂起
[8]	DPPUEN	**DP 上拉** 0 = D+上的上拉电阻禁止. 1 = D+上的上拉电阻使能.
[7:4]	Reserved	保留.
[3]	LOWPWREN	**PHY 低功耗模式使能位 (低有效)** USB PHY HS 低功耗模式配置控制位. W当该位为0时，USB PHY处于低功耗模式，USB PHY TX仅在USB设备控制器（USBD）将数据传输到USB总线时使能。 在这种模式下，HS信号质量不能保证。 0 = USB PHY 低功耗模式使能. 1 = USB PHY 低功耗模式禁止.
[2:0]	Reserved	保留.

6.16 USB 1.1 Host 控制器 (USBH)

6.16.1 概述

NUC505 系列包含一个 USB 1.1 Host 控制器 (USBH) ，支持开放主机控制接口(OpenHCI, OHCI)规范，提供寄存器级的控制操作，可以管理USB从机和数据传输。

USB主机控制器内部包含一个根Hub端口，一个可以在内存和USB总线上实时传输数据的DMA通道，以及端口电源控制和过流检测的功能。

USB主机控制器可以检测USB从设备的插拔操作，控制数据传输，收集状态使能USB总线，对插入的从机提供电源控制和过流检测功能。

6.16.2 特性

- 支持通用串行总线（USB）标准1.1。
- 支持开放主机控制接口(OpenHCI) 标准 1.0。
- 支持全速 (12Mbps) 和低速 (1.5Mbps) 两种USB 设备。
- 支持控制传输，批量传输，中断传输和同步传输。
- 内部根 Hub。
- LQFP48 或 LQFP64封装有一个USB主机接口 ，QFN88封装有两个USB主机接口。
- 提供端口电源控制和过流检测功能。
- 提供实时数据传输的DMA功能。

6.16.3 框图

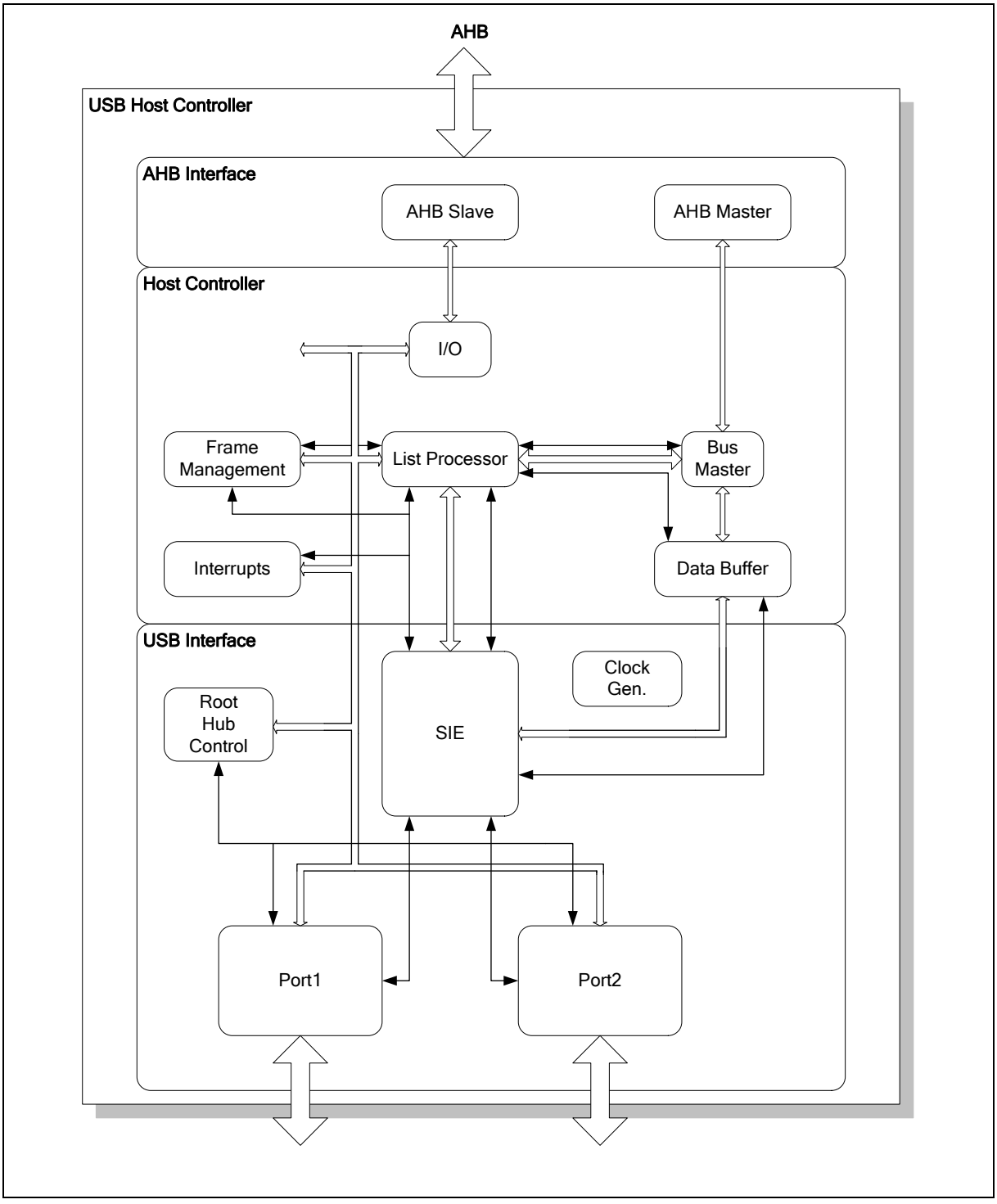


图 6.16-1 USB 1.1 主机控制器模块框图

6.16.4 基本描述

USBH 的时钟源从PLL分频而来。用户需要在使能USBH之前设定好PLL相关配置。 设置 USBHCKEN (CLK_AHBCLK[4]) 位为1 来使能 USBH时钟，然后设定 4位的预分频USBDIV (CLK_CLKDIV0[7:4])来产生合适的 USBH 时钟频率.合适的 USBH时钟频率是 48 MHz。

6.16.5 功能描述

6.16.5.1 AHB 接口

OpenHCI主控制器通过AHB总线接入系统，要求同时支持主/从总线操作。作为主机时，主控制器负责AHB总线和端点描述符（EDs）及传输描述符(TDs)的数据传输, 以及内存与数据缓存之间的数据传输。作为从机时，主控制器监控AHB总线上的数据命令，并决定何时回应这些命令。通过AHB总线的从机接口，可以配置和非实时操作主控制器的操作寄存器

6.16.5.2 主机控制器

主机控制器包含5个功能单元：

包括列表处理，帧管理，中断处理，主控制器总线主机以及数据缓存。

列表处理功能管理主控制器驱动的数据结构，并协调主控制器内的所有动作。

帧管理负责管理USB和OpenHCI规定的帧的特定任务，这些任务是：

1) 管理OpenHCI帧特定的操作寄存器
2) 最大数据包计数器操作
3) USB对SIE传送请求的帧验证
4) 向SIE产生SOF token 请求

主控制器与主控驱动间的通讯是通过中断方式实现的，主控制器有几种事件可以触发中断，每个事件会在HcInterruptStatus寄存器设置相应的标志位。

在数据通路中主机总线控制器是核心单元，它协调所有与AHB接口的操作，总线控制器有两个数据来源，列表处理器和数据缓存引擎。数据缓存作为总线控制器和SIE间的数据接口，是一个64字节双向异步FIFO和一个DWORD AHB保持寄存器的组合

6.16.5.3 USB 接口

USB接口包含集成的根Hub和两个外部接口，端口1和端口2，以及串行接口引擎（SIE）和USB时钟产生单元。负责主控制器发出的总线传输请求及USB协议定义的集线器和端口的管理。

SIE负责管理USB所有的传输行为，控制总线协议，数据包打包和拆包，数据并串转换，CRC校验，比特填充，NRZI译码。所有USB的传输都是由列表处理器和帧管理器发起的。

根集线器是一个分别控制的端口集合，它通过一些相同的功能模块维持所有端口的控制/状态。

6.16.6 寄存器列表

R: 只读, **W**: 只可写, **R/W**: 可读写

寄存器	偏移地址	R/W	描述	复位值
USBH 基地址: **USBH_BA = 0x4000_9000**				
HcRevision	USBH_BA+0x000	R	主控制器版本寄存器	0x0000_0110
HcControl	USBH_BA+0x004	R/W	主控制器控制寄存器	0x0000_0000
HcCommandStatus	USBH_BA+0x008	R/W	主控制器命令状态寄存器	0x0000_0000
HcInterruptStatus	USBH_BA+0x00C	R/W	主控制器中断状态寄存器	0x0000_0000
HcInterruptEnable	USBH_BA+0x010	R/W	主控制器中断使能寄存器	0x0000_0000
HcInterruptDisable	USBH_BA+0x014	R/W	主控制器中断禁止寄存器	0x0000_0000
HcHCCA	USBH_BA+0x018	R/W	主控制器通信控制区内存地址寄存器	0x0000_0000
HcPeriodCurrentED	USBH_BA+0x01C	R/W	主控制器当前周期端点描述符地址寄存器	0x0000_0000
HcControlHeadED	USBH_BA+0x020	R/W	主控制器控制端点描述符首地址寄存器	0x0000_0000
HcControlCurrentED	USBH_BA+0x024	R/W	主控制器控制端点描述符当前地址寄存器	0x0000_0000
HcBulkHeadED	USBH_BA+0x028	R/W	主控制器批量端点描述符首地址寄存器	0x0000_0000
HcBulkCurrentED	USBH_BA+0x02C	R/W	主控制器批量端点描述符当前地址寄存器	0x0000_0000
HcDoneHead	USBH_BA+0x030	R/W	主控制器已完成传输描述符首地址寄存器	0x0000_0000
HcFmInterval	USBH_BA+0x034	R/W	主控制器帧间隔寄存器	0x0000_2EDF
HcFmRemaining	USBH_BA+0x038	R	主控制器帧剩余时间寄存器	0x0000_0000
HcFmNumber	USBH_BA+0x03C	R	主控制器帧计数器	0x0000_0000
HcPeriodicStart	USBH_BA+0x040	R/W	主控制器周期列表开始处理寄存器	0x0000_0000
HcLSThreshold	USBH_BA+0x044	R/W	主控制器低速度门槛寄存器	0x0000_0628
HcRhDescriptorA	USBH_BA+0x048	R/W	主控制器根集线器描述符A寄存器	0x0000_0902
HcRhDescriptorB	USBH_BA+0x04C	R/W	主控制器根集线器描述符B寄存器	0x0000_0000
HcRhStatus	USBH_BA+0x050	R/W	主控制器根集线器状态寄存器	0x0000_0000
HcRhPortStatus1	USBH_BA+0x054	R/W	主控制器根集线器端口状态[1]	0x0000_0000
HcRhPortStatus2	USBH_BA+0x058	R/W	主控制器根集线器端口状态[2]	0x0000_0000
HcPhyControl	USBH_BA+0x200	R/W	主控制器PHY 控制寄存器	0x0000_0000
HcMiscControl	USBH_BA+0x204	R/W	主控制器多功能控制寄存器	0x0000_0000

6.16.7 寄存器描述

主控制器版本寄存器(HcRevision)

寄存器	偏移地址	R/W	描述	复位值
HcRevision	USBH_BA+0x000	R	主控制器版本寄存器	0x0000_0110

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**
REV							

位	描述	
[31:8]	**Reserved**	Reserved.
[7:0]	**REV**	**版本号** 表示硬件匹配的OpenHCI规格版本号 主控制器支持1.1版本规范 (X.Y = XYh).

主控制器控制寄存器(HcControl)

寄存器	偏移地址	R/W	描述	复位值
HcControl	USBH_BA+0x004	R/W	主控制器控制寄存器	0x0000_0000

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

位	描述	
[31:9]	**Reserved**	Reserved.
[8]	**Reserved**	Reserved.
[7:6]	**HCFS**	**主控制器功能状态** 设置主控制器状态，当控制器从下行端口收到RESUME信号后，可以强制将状态从USB暂停状态改变为USB恢复状态 可以有如下状态： 00 = USBRESET. 01 = USBRESUME. 10 = USBOPERATIONAL. 11 = USBSUSPEND.
[5]	**BLE**	**批量列表使能位** 0 = 下一个SOF (Start-Of-Frame)后禁止处理批量列表 1 = 使能下一帧处理批量列表.
[4]	**CLE**	**控制列表使能位** 0 = 下一个SOF (Start-Of-Frame)后禁止处理控制列表 1 = 使能下一帧处理控制列表
[3]	**IE**	**同步列表使能位** ISOEn及PLE (HcControl[2])为高使能主控制器处理同步列表，ISOEn 或 PLE (HcControl[2])为低禁止主控制器处理同步列表 0 = 下一个SOF (Start-Of-Frame)后禁止处理同步列表 1 = 如果PLE (HcControl[2])也为高，使能下一帧处理同步列表

[2]	**PLE**	**周期列表使能位** 为高时，使能处理周期（中断和同步）列表，主控制器在每一帧要开始周期传输之前都会检测该位 0 =下一个SOF (Start-Of-Frame)后禁止处理周期（中断和同步）列表 1 =使能下一帧处理周期列表 **注:**如果用户想使能同步列表，必须设置PLE 及IE (HcControl[3])为高.
[1:0]	**CBSR**	**控制/批量处理比例** 这里定义了控制/批量端点描述符的处理时间比例。在处理非周期列表之前，主控制器(HC)会将代表已处理过的控制端点描述符的内部计数器跟这里定义的比例做比对，以决定是继续处理控制端点描述符还是切换到处理批量端点描述符。当处理跨越多个帧时，内部计数器会被保存。如果发生复位，主控制器驱动(HCD)负责保存这个值。 00 = 控制端点描述符与批量端点描述符处理时间的比值是1:1. 01 = 控制端点描述符与批量端点描述符处理时间的比值是 2:1. 10 = 控制端点描述符与批量端点描述符处理时间的比值是 3:1. 11 = 控制端点描述符与批量端点描述符处理时间的比值是4:1.

主控制器命令状态寄存器(HcCommandStatus)

寄存器	偏移地址	R/W	描述	复位值
HcCommandStatus	USBH_BA+0x008	R/W	主控制器命令状态寄存器	0x0000_0000

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

位	描述	
[31:18]	**Reserved**	Reserved.
[17:16]	**SOC**	**调度超时计数** 每次发生调度超时错误这里会加一，复位为00b,计数到11b后复位。就算SO (HcInterruptStatus[0])被置位了，发生调度超时错误时这里还会加一
[15:3]	**Reserved**	Reserved.
[2]	**BLF**	**批量列表满** 设置为1表示批量列表上有使能的传输描述符。此位可能被软件或者主控制器设置，主控制器开始处理批量列表时将此位清零 0 = 没有发现使能的传输描述符或者主控制器开始处理批量列表了. 1 = 在批量列表发现或者添加了使能的传输描述符.
[1]	**CLF**	**控制列表满** 设置为1表示控制列表上有使能的传输描述符。此位可能被软件或者主控制器设置，主控制器开始处理控制列表时将此位清零 0 =没有发现使能的传输描述符或者主控制器开始处理控制列表了 1 =在控制列表发现或者添加了使能的传输描述符..
[0]	**HCR**	**主控制器复位** 软件设置此位为1复位主控制器，复位操作完成后主控制器将此位清零 设置此位时不会复位根集线器，也没有复位信号发到下行端口 0 =主控制器不在软件复位状态 1 = 主控制器在软件复位状态

主控制器中断状态寄存器(HcInterruptStatus)

寄存器	偏移地址	R/W	描述	复位值
HcInterruptStatus	USBH_BA+0x00C	R/W	主控制器中断状态寄存器	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	RHSC	FNO	Reserved	RD	SF	WDH	SO

位	描述	
[31:7]	**Reserved**	Reserved.
[6]	**RHSC**	**根集线器状态改变** 当寄存器HcRhStatus或HcRhPortStatus1的内容改变时，该位被设置 0 = 寄存器HcRhStatus和HcRhPortStatus1的内容没有改变. 1 = 寄存器HcRhStatus或HcRhPortStatus1的内容改变了.
[5]	**FNO**	**帧计数器溢出** 当帧计数器的Bit15从1变为0，或者从0变为1时该位被设置 0 =帧计数器的Bit15没有变化 1 =帧计数器的Bit15产生了从1到0，或者从0到1的变化.
[4]	**Reserved**	Reserved.
[3]	**RD**	**唤醒检测** 主控制器检测到下行端口有唤醒信号时设置该位. 0 = 下行端口没有唤醒信号 1 = 下行端口有唤醒信号.
[2]	**SF**	**帧开始** 帧管理模块发出'帧开始'信号时设置此位，同时主控制器将产生一个SOF token 0 =没有帧开始信号 1 =有帧开始信号，同时主控制器产生一个SOF token

[1]	**WDH**	**完成的传输描述符头地址写回** 主控制器将完成了的传输描述符头地址HcDoneHead写到寄存器HccaDoneHead时此位置高，此位被清零前HccaDoneHead不能再次被更新 0 =主控制器没有更新HccaDoneHead. 1 =主控制器写HcDoneHead 到 HccaDoneHead.
[0]	**SO**	**调度溢出** 当列表处理器调度溢出时设置此位. 0 = 没有发生调度溢出 1 = 发生了调度溢出

主控制器中断使能寄存器(HcInterruptEnable)

寄存器	偏移地址	R/W	描述	复位值
HcInterruptEnable	USBH_BA+0x010	R/W	主控制器中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MIE	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	RHSC	FNO	Reserved	RD	SF	WDH	SO

位	描述	
[31]	**MIE**	**主中断使能位** 此位是全局中断使能位，写1后使能本寄存器其他位的中断使能 写操作: 0 = 没有影响 1 = 如果寄存器HcInterruptEnable相应的位为1，则使能由以下原因造成的中断：RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 以及 SO (HcInterruptStatus[0]) 读操作: 0 =就算HcInterruptEnable相应位为1，由以下原因造成的中断也不会被使能： RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 以及 SO (HcInterruptStatus[0]) 1 = 如果HcInterruptEnable相应位为1，由以下原因造成的中断将被使能： RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 以及 SO (HcInterruptStatus[0])
[30:7]	**Reserved**	Reserved.
[6]	**RHSC**	**根集线器状态改变中断使能位** 写操作: 0 = 没有影响. 1 = 使能RHSC (HcInterruptStatus[6])产生的中断 读操作: 0 = RHSC (HcInterruptStatus[6])产生的中断将被禁止 1 = RHSC (HcInterruptStatus[6])产生的中断将被使能

[5]	**FNO**	**帧计数溢出中断使能位** 写操作: 0 = 没有影响. 1 = 使能FNO (HcInterruptStatus[5]) 产生的中断 读操作: 0 = FNO (HcInterruptStatus[5])产生的中断被禁止. 1 = FNO (HcInterruptStatus[5])产生的中断被使能
[4]	**Reserved**	Reserved.
[3]	**RD**	**唤醒检测中断使能位** 写操作: 0 = 没有影响. 1 = 使能RD (HcInterruptStatus[3]) 产生的中断 读操作: 0 = RD (HcInterruptStatus[3])产生的中断被禁止. 1 = RD (HcInterruptStatus[3])产生的中断被使能
[2]	**SF**	**帧开始中断使能** 写操作: 0 = 没有影响. 1 =使能 SF (HcInterruptStatus[2]) 产生的中断 读操作: 0 = SF (HcInterruptStatus[2])产生的中断被禁止. 1 = SF (HcInterruptStatus[2])产生的中断被使能
[1]	**WDH**	**完成的传输描述符头地址写回中断使能位** 写操作: 0 = 没有影响. 1 = 使能WDH (HcInterruptStatus[1])产生的中断 读操作: 0 = WDH (HcInterruptStatus[1])产生的中断被禁止. 1 = WDH (HcInterruptStatus[1])产生的中断被使能
[0]	**SO**	**调度溢出中断使能位** 写操作: 0 = 没有影响. 1 = 使能SO (HcInterruptStatus[0]) 产生的中断 读操作: 0 = SO (HcInterruptStatus[0]) 产生的中断被禁止. 1 = SO (HcInterruptStatus[0]) 产生的中断被使能

主控制器中断禁止寄存器(HcInterruptDisable)

寄存器	偏移地址	R/W	描述	复位值
HcInterruptDisable	USBH_BA+0x014	R/W	主控制器中断禁止寄存器	0x0000_0000

31	30	29	28	27	26	25	24
MIE	**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	**RHSC**	**FNO**	**Reserved**	**RD**	**SF**	**WDH**	**SO**

位	描述	
[31]	**MIE**	**主中断禁止位** 此位是全局中断禁止位，写1后禁止所有中断 写操作: 0 = 没有影响 1 = 就算寄存器HcInterruptEnable相应的位为1，由以下原因造成的中断也会被禁止：RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 以及 SO (HcInterruptStatus[0]) 读操作: 0 = 就算HcInterruptEnable相应位为1，由以下原因造成的中断也不会被使能： RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 以及 SO (HcInterruptStatus[0]) 1 =如果HcInterruptEnable相应位为1，由以下原因造成的中断被使能： RHSC (HcInterruptStatus[6]), FNO (HcInterruptStatus[5]), RD (HcInterruptStatus[3]), SF (HcInterruptStatus[2]), WDH (HcInterruptStatus[1]) 以及 SO (HcInterruptStatus[0])
[30:7]	**Reserved**	Reserved.
[6]	**RHSC**	**根集线器状态改变中断禁止位** 写操作: 0 = 没有影响. 1 = RHSC (HcInterruptStatus[6])产生的中断将被禁止 读操作: 0 = RHSC (HcInterruptStatus[6])产生的中断被禁止 1 = RHSC (HcInterruptStatus[6])产生的中断被使能

[5]	**FNO**	**帧计数溢出中断禁止位** 写操作: 0 = 没有影响. 1 = FNO (HcInterruptStatus[5]) 产生的中断禁止 读操作: 0 = FNO (HcInterruptStatus[5])产生的中断被禁止 1 = FNO (HcInterruptStatus[5])产生的中断被使能.
[4]	**Reserved**	Reserved.
[3]	**RD**	**唤醒检测中断禁止位** 写操作: 0 = 没有影响. 1 = RD (HcInterruptStatus[3]) 产生的中断禁止 读操作: 1 = RD (HcInterruptStatus[3])产生的中断被使能. 0 = RD (HcInterruptStatus[3])产生的中断被禁止
[2]	**SF**	**帧开始中断禁止** 写操作: 0 = 没有影响. 1 = SF (HcInterruptStatus[2]) 产生的中断禁止 读操作: 1 = SF (HcInterruptStatus[2])产生的中断被使能. 0 = SF (HcInterruptStatus[2])产生的中断被禁止
[1]	**WDH**	**完成的传输描述符头地址写回中断禁止位** 写操作: 0 = 没有影响. 1 = WDH (HcInterruptStatus[1])产生的中断禁止 读操作: 1 = WDH (HcInterruptStatus[1])产生的中断被使能. 0 = WDH (HcInterruptStatus[1])产生的中断被禁止
[0]	**SO**	**调度溢出中断禁止位** 写操作: 0 = 没有影响. 1 = SO (HcInterruptStatus[0]) 产生的中断禁止 读操作: 1 = SO (HcInterruptStatus[0]) 产生的中断被使能. 0 = SO (HcInterruptStatus[0]) 产生的中断被禁止

主控制器通信控制区内存地址寄存器(HcHCCA)

寄存器	偏移地址	R/W	描述	复位值
HcHCCA	USBH_BA+0x018	R/W	主控制器通信控制区内存地址寄存器	0x0000_0000

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

位	描述	
[31:8]	**HCCA**	**主控制器通信控制区** 主控制器通信控制区的基地址指针
[7:0]	**Reserved**	Reserved.

主控制器当前周期端点描述符地址寄存器(HcPeriodCurrentED)

寄存器	偏移地址	R/W	描述	复位值
HcPeriodCurrentED	USBH_BA+0x01C	R/W	主控制器当前周期端点描述符地址寄存器	0x0000_0000

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

位	描述	
[31:4]	**PCED**	**当前周期端点描述符** 当前同步或中断端点描述符的物理地址指针
[3:0]	**Reserved**	Reserved.

主控制器控制端点描述符首地址寄存器(HcControlHeadED)

寄存器	偏移地址	R/W	描述	复位值
HcControlHeadED	USBH_BA+0x020	R/W	主控制器控制端点描述符首地址寄存器	0x0000_0000

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

位	描述	
[31:4]	CHED	Control Head ED 控制列表第一个端点描述符的物理地址指针
[3:0]	Reserved	Reserved.

主控制器控制端点描述符当前地址寄存器(HcControlCurrentED)

寄存器	偏移地址	R/W	描述	复位值
HcControlCurrentED	USBH_BA+0x024	R/W	主控制器控制端点描述符当前地址寄存器	0x0000_0000

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

位	描述	
[31:4]	CCED	**Control Current Head ED** 控制列表当前端点描述符物理地址指针
[3:0]	Reserved	Reserved.

主控制器批量端点描述符首地址寄存器(HcBulkHeadED)

寄存器	偏移地址	R/W	描述	复位值
HcBulkHeadED	USBH_BA+0x028	R/W	主控制器批量端点描述符首地址寄存器	0x0000_0000

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

位	描述	
[31:4]	**BHED**	**批量端点描述符首地址** 批量列表端点描述符首地址指针.
[3:0]	**Reserved**	Reserved.

主控制器批量端点描述符当前地址寄存器(HcBulkCurrentED)

寄存器	偏移地址	R/W	描述	复位值
HcBulkCurrentED	USBH_BA+0x02C	R/W	主控制器批量端点描述符当前地址寄存器	0x0000_0000

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

位	描述	
[31:4]	**BCED**	**批量端点描述符当前地址** 批量列表当前描述符物理地址指针
[3:0]	**Reserved**	保留.

主控制器已完成传输描述符首地址寄存器(HcDoneHead)

寄存器	偏移地址	R/W	描述	复位值
HcDoneHead	USBH_BA+0x030	R/W	主控制器已完成传输描述符首地址寄存器	0x0000_0000

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

位	描述	
[31:4]	**DH**	**已完成传输描述符首地址** 加入完成队列的最新完成的传输描述符物理地址指针
[3:0]	**Reserved**	保留.

主控制器帧间隔寄存器(HcFmInterval)

寄存器	偏移地址	R/W	描述	复位值
HcFmInterval	USBH_BA+0x034	R/W	主控制器帧间隔寄存器	0x0000_2EDF

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

位	描述	
[31]	**FIT**	**帧间隔反转** 当主控制器驱动加载一个新的值到FI (HcFmInterval[13:0])中，此位反转 0 =主控制驱动没有加载新的值到FI (HcFmInterval[13:0]). 1 =主控制驱动加载新的值到FI (HcFmInterval[13:0]).
[30:16]	**FSMPS**	**最大数据包** 这里定义了帧开始时加载到最大数据包计数器中的值
[15:14]	**Reserved**	保留.
[13:0]	**FI**	**帧间隔** 这里以位时间（bit time）定义了帧的长度为（位时间-1），例如，一帧12,000位时间这里应当存储11999

主控制器帧剩余时间寄存器(HcFmRemaining)

寄存器	偏移地址	R/W	描述	复位值
HcFmRemaining	USBH_BA+0x038	R	主控制器帧剩余时间寄存器	0x0000_0000

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

位	描述	
[31]	**FRT**	**帧剩余反转** 当FR (HcFmRemaining[13:0])到0时，此位从FIT (HcFmInterval[31])加载
[30:14]	**Reserved**	保留.
[13:0]	**FR**	**帧剩余** 当主控制器在USBOPERATIONAL状态时，FR每个12MHz时钟周期减1，当减到0时（帧结束）计数器重新装载帧间隔，此外，当主控制器从其他状态切换到USBOPERATIONAL时计数器也会重新装载

主控制器帧计数器(HcFmNumber)

寄存器	偏移地址	R/W	描述	复位值
HcFmNumber	USBH_BA+0x03C	R	主控制器帧计数器	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**
FN							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
FN							

位	描述	
[31:16]	**Reserved**	保留.
[15:0]	**FN**	**帧计数器** 在FR (HcFmRemaining[13:0])重新装载时此计数器加一，计数从‘FFFFh’ 到 ‘0h’循环

主控制器周期列表开始处理寄存器(HcPeriodicStart)

寄存器	偏移地址	R/W	描述	复位值
HcPeriodicStart	USBH_BA+0x040	R/W	主控制器周期列表开始处理寄存器	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		**PS**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PS							

位	描述	
[31:14]	**Reserved**	保留.
[13:0]	**PS**	**周期列表开始处理** 一个帧中，列表处理器检测PS决定何时开始处理周期列表

主控制器低速度门槛寄存器(HcLSThreshold)

寄存器	偏移地址	R/W	描述	复位值
HcLSThreshold	USBH_BA+0x044	R/W	主控制器低速度门槛寄存器	0x0000_0628

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				**LST**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LST							

位	描述	
[31:12]	**Reserved**	保留.
[11:0]	**LST**	**低速门槛** 主控制器驱动在启动低速传输前先将LST的值与FR (HcFmRemaining[13:0])比较只有当FR (HcFmRemaining[13:0])>=LST的时候才会启动低速传输

主控制器根集线器描述符A寄存器(HcRhDescriptorA)

寄存器	偏移地址	R/W	描述	复位值
HcRhDescriptorA	USBH_BA+0x048	R/W	主控制器根集线器描述符A寄存器	0x0000_0902

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			NOCP	OCPM	Reserved	Reserved	PSM
7	6	5	4	3	2	1	0
NDP							

位	描述	
[31:13]	**Reserved**	保留.
[12]	**NOCP**	**无过电流保护** 此位描述了当根集线器端口过电流是否会被报告 0 = 过电流状态被报告 1 = 过电流状态不会被报告
[11]	**OCPM**	**过电流保护模式** 此位描述了根集线器过电流状态如何被报告，此位只有在NOCP (HcRhDescriptorA[12])被清零时有效 0 =所有下行端口过电流状态都会被报告 1 =每个端口过电流状态被单独报告
[10:9]	**Reserved**	保留.
[8]	**PSM**	**电源切换模式** 此位用来定义根集线器电源切换的方式 0 = 所有端口同时供电 1 = 每个端口可单独供电
[7:0]	**NDP**	**下行端口数量** 主控制器支持2个下行端口，但是在此芯片中只有1个端口可用

主控制器根集线器描述符B寄存器(HcRhDescriptorB)

寄存器	偏移地址	R/W	描述	复位值
HcRhDescriptorB	USBH_BA+0x04C	R/W	主控制器根集线器描述符B寄存器	0x0000_0000

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

位	描述	
[31:16]	**PPCM**	**端口电源控制屏蔽** 电源切换模式PSM为1时，这里每一位决定一个端口是否受所有端口同时供电命令的影响，为1时各个端口受单独供电命令影响，为0时所有端口受同时供电命令影响。 0 = 所有端口受同时供电命令影响 1 = 各个端口受单独供电命令影响 **注:** PPCM[15:2] 及 PPCM[0] 是保留位.
[15:0]	**Reserved**	保留.

主控制器根集线器状态寄存器(HcRhStatus)

寄存器	偏移地址	R/W	描述	复位值
HcRhStatus	USBH_BA+0x050	R/W	主控制器根集线器状态寄存器	0x0000_0000

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

位	描述	
[31]	**CRWE**	**清除远程唤醒使能位** 此位用来清除DRWE (HcRhStatus[15]). 此位只能读到0 写操作: 0 = 没有影响 1 = 清除 DRWE (HcRhStatus[15]).
[31:18]	**Reserved**	保留.
[17]	**OCIC**	**过电流显示改变** 当OCI (HcRhStatus[1])发生变化时此位由硬件置1，此位写1清0 0 = OCI (HcRhStatus[1]) 没有变化. 1 = OCI (HcRhStatus[1]) 改变了
[16]	**LPSC**	**打开所有端口供电** 当(PSM (HcRhDescriptorA[8]) = 0)时，此位用来使能所有端口供电 此位总是读为0 写操作: 0 = 没有影响 1 = 所有端口供电

[15]	**DRWE**	**设备远程唤醒使能位** 此位使能连接状态改变（HcRhPortStatus1[16]）作为远程唤醒事件 写操作： 0 = 没有影响. 1 = 使能连接状态改变作为远程唤醒事件 读操作: 0 = 禁止连接状态改变作为远程唤醒事件 1 = 使能连接状态改变作为远程唤醒事件
[14:2]	**Reserved**	保留.
[1]	**OCI**	**过电流显示** 此位显示了过电流的状态，只有在NOCP (HcRhDesA[12]) 及 OCPM (HcRhDesA[11])为0时有效 0 = 没有过电流状况 1 = 出现过电流状况
[0]	**LPS**	**切断所有端口电源** 当电源模式为(PSM (HcRhDescriptorA[8]) = 0)时，此位写1将切断所有端口电源 此位只能读为0. 写操作: 0 = 没影响. 1 = 切断所有端口电源.

主控制器根集线器端口状态寄存器 (HcRhPrt [1:2])

寄存器	偏移地址	R/W	描述	复位值
HcRhPortStatus1	USBH_BA+0x054	R/W	主控制器根集线器端口状态寄存器[1]	0x0000_0000
HcRhPortStatus2	USBH_BA+0x058	R/W	主控制器根集线器端口状态寄存器[2]	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved			PRSC	OCIC	PSSC	PESC	CSC
15	14	13	12	11	10	9	8
Reserved						LSDA	PPS
7	6	5	4	3	2	1	0
Reserved			PRS	POCI	PSS	PES	CCS

位	描述	
[31:21]	**Reserved**	保留.
[20]	**PRSC**	**端口复位状态改变** 此位显示端口复位信号完成 此位写1清0 0 = 端口复位信号没有完成 1 = 端口复位信号完成
[19]	**OCIC**	**端口过电流显示改变** 当 POCI (HcRhPortStatus1[3])改变时，此位置1 写1清0 0 = POCI (HcRhPortStatus1[3]) 没有改变 1 = POCI (HcRhPortStatus1[3]) 改变了
[18]	**PSSC**	**端口暂停状态改变** 此位表示端口有完整的唤醒时序 写1清0 0 = 端口唤醒没有发生 1 = 端口唤醒
[17]	**PESC**	**端口使能状态改变** 该位表示端口由于硬件事件而被禁止（PES (HcRhPortStatus1[1])=0）了 写1清0 0 = PES (HcRhPortStatus1[1]) 没有改变. 1 = PES (HcRhPortStatus1[1]) 改变了

[16]	**CSC**	**连接状态改变** 此位显示检测到连接或者断开事件发生 (CCS (HcRhPortStatus1[0]) 改变). 写1清0 0 = 没有连接/断开事件发生(CCS (HcRhPortStatus1[0]) 不变). 1 =硬件检测到连接/断开事件发生(CCS (HcRhPortStatus1[0]) 改变).
[15:10]	**Reserved**	保留.
[9]	**LSDA**	**低速设备插入（读）或切断端口电源（写）** 该位定义了插入设备的速度，只有在CCS (HcRhPortStatus1[0])为1是有效。此位也用来切断端口电源 写操作: 0 = 没有影响. 1 = 清除 PPS (HcRhPortStatus1[8]). 读操作: 0 = 全速设备 1 = 低速设备
[8]	**PPS**	**端口电源状态** 此位显示了端口的电源状态，不管电源切换模式 写操作: 0 = 没有影响. 1 = 端口电源使能. 读操作: 0 = 端口电源禁止. 1 = 端口电源使能.
[7:5]	**Reserved**	保留.
[4]	**PRS**	**端口复位状态** 此位反映了端口的复位状态 写操作: 0 = 没有影响. 1 = 设置端口复位. 读操作： 0 = 端口复位信号没有使能. 1 = 端口复位信号使能.
[3]	**POCI**	**端口过电流显示（读）或 清除暂停状态（写）** 此位反应了端口过电流的状态，只有在NOCP (HcRhDescriptorA[12]) 为 0 且OCPM (HcRhDescriptorA[11]) 为1时有效 该位也可用于启动端口原状态。 写操作: 0 = 没有影响 1 = 清除暂停状态. 读操作: 0 = 没有过电流状况 1 = 有过电流状况

[2]	**PSS**	**端口暂停状态** 此位显示端口是暂停的 写操作: 0 = 没有影响 1 = 设置端口暂停 读操作: 0 = 端口没有暂停 1 = 端口暂停
[1]	**PES**	**端口使能状态** 写操作： 0 = 没有影响 1 = 设置端口使能. 读操作: 0 = 端口禁止. 1 = 端口使能.
[0]	**CCS**	**当前连接状态（读） 或 清除端口使能位（写）** 写操作: 0 = 没有影响 1 = 清除端口使能. 读操作: 0 = 没有连接的设备. 1 = 设备连接了

主控制器PHY 控制寄存器(HcPhyControl)

寄存器	偏移地址	R/W	描述	复位值
HcPhyControl	USBH_BA+0x200	R/W	主控制器PHY 控制寄存器	0x0000_0000

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

位	描述	
[31:28]	Reserved	保留.
[27]	STBYEN	**USB 收发器待机使能位** 此位控制USB收发器是否可以进入待机模式以减小功耗 0 = USB收发器永远不会进入待机模式 1 = 当端口在断电模式下（端口电源不使能），USB收发器将进入待机模式
[26:0]	Reserved	保留.

主控制器多功能控制寄存器(HcMiscControl)

寄存器	偏移地址	R/W	描述	复位值
HcMiscControl	USBH_BA+0x204	R/W	主控制器多功能控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved						DPRT2	DPRT1
15	14	13	12	11	10	9	8
Reserved							SIEPD
7	6	5	4	3	2	1	0
Reserved			PCAL	OCAL	Reserved	ABORT	DBR16

位	描述	
[31:18]	**Reserved**	保留.
[17]	**DPRT2**	**禁止端口2** 此位控制连接USB主机与收发器之间的端口2是否使能，如果连接被禁止，USB主机将不会收到任何USB总线事件 设置此位为2，不管USB主机作何操作，端口1的收发器将强制进入待机模式 0 = USB主机和端口2的收发器间的连接使能 1 = USB主机和端口2的收发器间的连接禁止，收发器将强制进入待机模式
[16]	**DPRT1**	**禁止端口 1** 此位控制连接USB主机与收发器之间的端口1是否使能，如果连接被禁止，USB主机将不会收到任何USB总线事件 设置此位为1，不管USB主机作何操作，端口1的收发器将强制进入待机模式 0 = USB主机和端口1的收发器间的连接使能 1 = USB主机和端口1的收发器间的连接禁止，收发器将强制进入待机模式
[15:9]	**Reserved**	保留.
[8]	**SIEPD**	**SIE 管道禁止位** 当此位置位，等待所有使能的USB总线完成之后再回报完成状态给列表处理器 此位是用来避免1.5MHz与12MHz时钟切换的时候带来的潜在的问题
[7:5]	**Reserved**	保留.
[4]	**PCAL**	**端口电源控制低使能** 这一位用来控制端口电源控制单元控制外部电源芯片的信号极性 0 = 端口电源控制信号高有效. 1 =端口电源控制信号低有效

[3]	**OCAL**	**过电流低有效** 此位控制外部电源IC过电流指示的极性 0 =过电流标志高有效 1 = 过电流标志低有效
[2]	**Reserved**	保留.
[1]	**ABORT**	**AHB 总线错误响应** 此位显示AHB总线收到错误响应 0 = 没有收到错误响应 1 = 收到了错误响应
[0]	**DBR16**	**数据缓存区16字节** 此位为1，数据缓存器为16字节，为0，大小是32 字节

6.17 安全数字主控制器(SDHC)

6.17.1 概述

安全数字主控制器 (SDH Controller)包含一个DMAC单元 (直接内存存取控制器)和一个SD单元 。DMAC单元在系统内存和共享缓存（128 字节）之间提供DMA (直接内存存取)功能，SD单元控制SD/SDHC/MMC的接口。SD控制器支持SD/SDHC/MMC接口，并且可以通过DMAC来实现卡片与系统内存之间的快速传输。

6.17.2 特性

- 支持一个DMA通道。
- 支持硬件Scatter-Gather 功能.
- 提供一个128 字节 大小的共享缓存用来在卡片和系统内存之间传递数据
- 提供DMAC寄存器读写和数据传输接口.
- 支持SD/SDHC/MMC 卡.
- 系统时钟HCLK频率需要高于SDHC时钟频率

6.17.3 框图和卡片脚位设定

模块框图和卡片脚位设定如下：

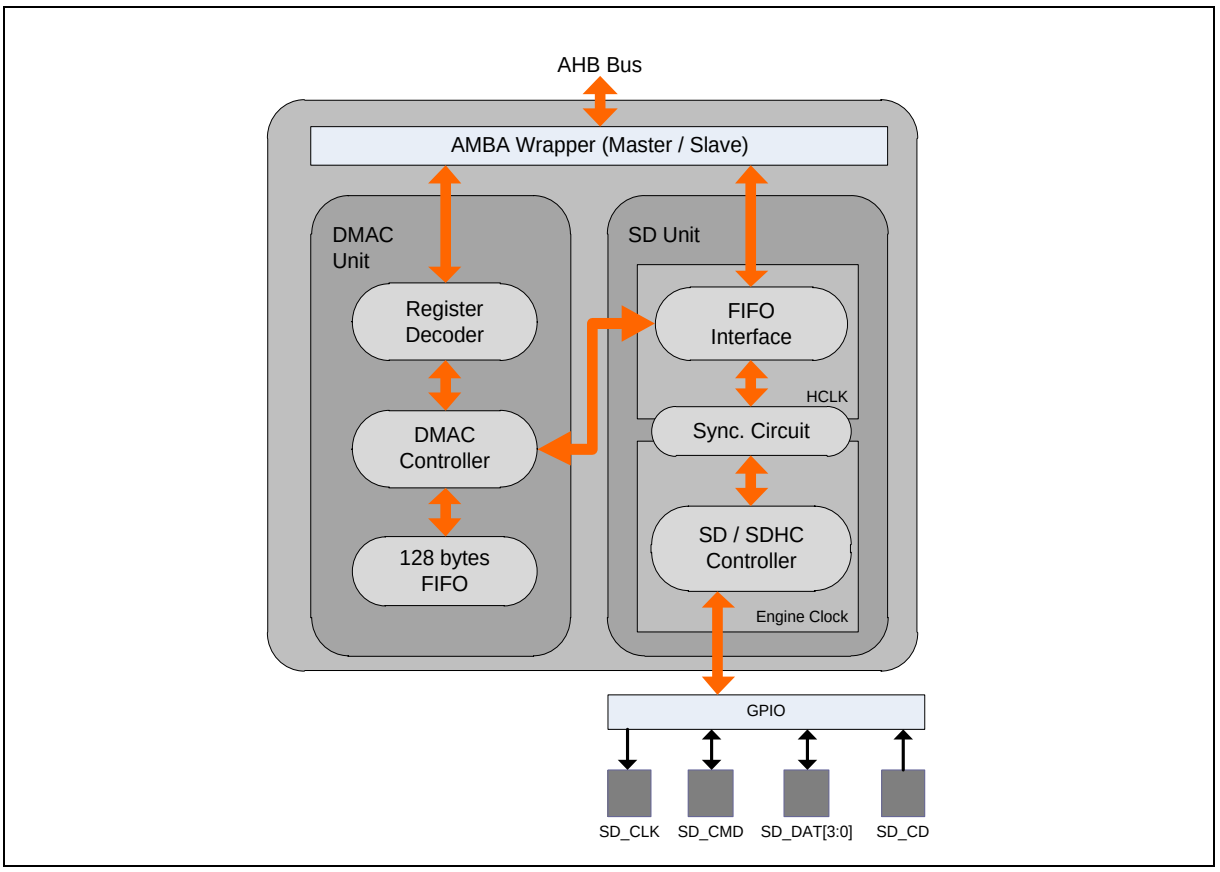


图 6.17-1 SDHC模块框图

名称	描述
SD_DAT0	SD Data (bit 0)
SD_DAT1	SD Data (bit 1)
SD_DAT2	SD Data (bit 2)
SD_DAT3	SD Data (bit 3)
SD_CMD	SD Command / Response
SD_CLK	SD Clock pin
SD_CD	Card Detect (Source can be GPIO or DAT3 (in SDIER)

表 6.17-1 SD/SDHC/MMC 脚位设定

6.17.4 SD 主机 DMA控制器

SD主机DMA控制器提供DMA (直接内存存取)功能，可以在系统内存 (例如. SRAM) 和共享缓存 (128 字节)直接传送数据。程序仅需要填写起始地址，然后使能DMAC，DMAC会自动传送数据。

6.17.4.1 特性

- 支持一路DMA通道，地址不需要字对齐。
- 支持 SD/SDHC/MMC 卡字节访问。
- 支持硬件 Scatter-Gather 功能。
- 内含一块123 字节的共享缓存。

6.17.4.2 模块框图

DMA控制器模块框图如下：

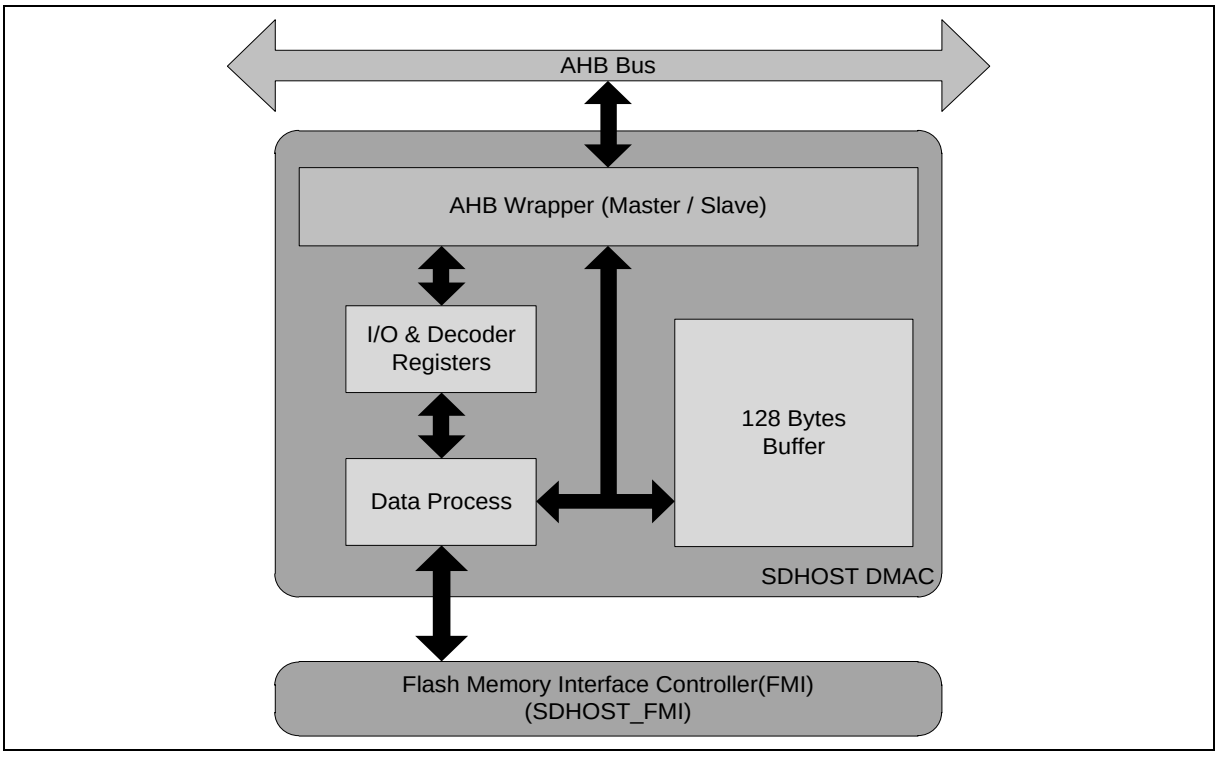


图 6.17-2 DMA 控制器模块框图

6.17.4.3 编程流程

这有一份不打开Scatter-Gather 功能的DMA编程流程：

1. DMAEN (SDH_DMACTL[0])置1 使能 DMAC.
2. 在 SDH_DMASA 填写 DMAC 相关起始地址.

3. 使能 SDHOST_FMI 开始 DMA 传输.
4. 等待直到 SDHOST_FMI 指示完成.

这有一份打开Scatter-Gather 功能的DMA编程流程：.

1. DMAEN (SDH_DMACTL[0])置1 使能 DMAC，并且 SGEN (SDH_DMACTL[3]) 置1使能 Scatter-Gather 功能.
2. 在 SDH_DMASA 里的物理地址描述符 (PAD)表格填写 DMAC 相关起始地址.
3. 当 SDH_DMASA[0] 是 1, PAD 读取顺序不对; 如果为0，读取顺序是正确的。在这个功能下第一次不能给最低位写0或写1，这一位只在 PAD 表格(图 6.17-3 PAD 图 6.17-3).里有效。
4. 使能 SDHOST_FMI 开始 DMA 传输.
5. 等待直到 SDHOST_FMI 指示完成.

6.17.5 安全数字 SD/SDHC 控制器

SD/SDHC/MMC 控制器通过 DMAC 实现系统内存和卡之间的快速数据存取.。DMAC内置一个128字节的缓存来临时保存数据。因为DMAC只有一路通道，所以同时只能使能一个接口

6.17.5.1 特性

- 提供DMAC接口用来寄存器读写和数据传输
- 支持 SD/SDHC/MMC 卡
- 支持 SD/SDHC/MMC 烧写时序
- 提供128 字节的共享缓存，用来在系统内存和卡之间传递数据。
- HCLK的频率一定要高于SD引擎的工作频率

6.17.5.2 功能描述

SD控制器提供一个SD端口-端口0， 支持1bit、4bit两种总线模式

SD控制器有单独的时钟源 SDCLK，可以与HCLK完全异步。程序可以任意设定SDCLK，但是HCLK的频率一定要高于SDCLK。

SD控制器可以产生SD卡所有种类的48-bit指令，而且可以检索所有SD卡应答。当接收到应答后，内容被存储在SDH_RESP0 和 SDH_RESP1，并且SD控制器会计算CRC-7校验值。如果CRC-7校验值不对， CRCIF (SDH_INTSTS[1]) 会置位 ，CRC7 (SDH_INTSTS[2]) 会置0。 如果应答是R1b， 当接收到应答软件需要通知SD卡发送一个忙信号到信号线DAT0，程序需要轮询这个信号状态直到其变高。当应答是R3， CRC-7校验是无效的，但SD控制器仍然会计算CRC-7校验值并得到不正确的结果，程序需要忽略这个错误并且清除 CRCIF 标志 (SDH_INTSTS[1]).

SD控制器包含两个状态机：指令、应答部分和数据部分。指令应答部分，触发SDH_CTL 的COEN, RIEN, R2EN, CLK74OEN 和 CLK8OEN这些位。如果程序使能所有这些位，操作顺序需要CLK74OEN → COEN →RIEN /R2EN → CLK8OEN。注 RIEN 和 R2EN 不能同时触发。数据部分，有DIEN 和 DOEN 可以选择。程序每次只可触发其中一个。如果 DIEN 触发，SD控制器会等待信号线DAT0上的开始信号，然后开始接受来自SD卡的设定数目的数据。接收到数据之后，SD控制器会计算CRC-16的校验值。如果校验值不正确，CRCIF (SDH_INTSTS[1])位会置位，并且 CRC16 (SDH_INTSTS[3]) 会置0。如果DOEN触发，SD控制器将等待应答完成，然后发送设定数目的数据到SD卡。数据发送出去之后，SD控制器会得到SD卡计算之后的CRC校验状态值。如果正确这个值 ‘010’，如果是其他值CRCIF (SDH_INTSTS[1])会置位，状态值会存到 CRCSTS (SDH_INTSTS[6:4]) 里面。

如果 R2EN 触发，SD控制器会从SD卡接收R2 (136 位) 的应答，还有 CRC-7 校验值，最后一位会丢弃 。接收到的数据会放在DMAC的缓存, 从偏移地址 0x0开始的地方。

SD控制器也提供多个数据块传输的功能(设定BLKLEN t来设定数据块长度).。程序可以使用这个功能加速数据的传输。如果 CRC7, CRC16 或 CRC 状态不正确，SD控制器会停止传输，然后设定CRCIF (SDH_INTSTS[1])为1，当这种情况发生程序需要复位 引擎。

当应答在SD引擎里时有硬件的超时机制，程序可以在TOUT里设定24bit的超时数值，根据这个数值SD控制器可以决定超时时间的长度。

6.17.6 寄存器列表

R: 只读, **W**: 只可写, **R/W**: 可读写

寄存器	地址偏移	R/W	描述	Reset 值
SDH 基地址: **SDH_BA = 0x4000_A000**				
SDH_FB_n **n=0,1..31**	SDH_BA+0x000+0x4 * n	R/W	SDH 收发闪存缓存	0x0000_0000
SDH_DMACTL	SDH_BA+0x400	R/W	DMAC控制和状态寄存器	0x0000_0000
SDH_DMASA	SDH_BA+0x408	R/W	DMAC 传输起始地址寄存器	0x0000_0000
SDH_DMABCNT	SDH_BA+0x40C	R	DMAC 传输字节数寄存器	0x0000_0000
SDH_DMAINTEN	SDH_BA+0x410	R/W	DMAC 中断使能控制寄存器	0x0000_0001
SDH_DMAINTSTS	SDH_BA+0x414	R/W	DMAC 中断状态寄存器	0x0000_0000
SDH_GCTL	SDH_BA+0x800	R/W	总体控制和状态寄存器	0x0000_0000
SDH_GINTEN	SDH_BA+0x804	R/W	总体中断控制寄存器	0x0000_0001
SDH_GINTSTS	SDH_BA+0x808	R/W	总体中断状态寄存器	0x0000_0000
SDH_CTL	SDH_BA+0x820	R/W	SD 控制和状态寄存器	0x0101_0000
SDH_CMDARG	SDH_BA+0x824	R/W	SD 指令参数寄存器	0x0000_0000
SDH_INTEN	SDH_BA+0x828	R/W	SD 中断控制寄存器	0x0000_0A00
SDH_INTSTS	SDH_BA+0x82C	R/W	SD 中断状态寄存器	0x000X_008C
SDH_RESP0	SDH_BA+0x830	R	SD 接收应答标记寄存器 0	0x0000_0000
SDH_RESP1	SDH_BA+0x834	R	SD 接受应答标记寄存器 1	0x0000_0000
SDH_BLEN	SDH_BA+0x838	R/W	SD 数据块长度寄存器	0x0000_01FF
SDH_TOUT	SDH_BA+0x83C	R/W	SD 应答/数据输入 超时寄存器	0x0000_0000

6.17.7 寄存器描述

6.17.7.1 DMAC 寄存器描述

SDH收发闪存缓存(SDH_FB_n)

寄存器	偏移地址	R/W	描述	复位值
SDH_FB_n n=0,1..31	SDH_BA+0x000+0x4 * n	R/W	SDH 收发闪存缓存	0x0000_0000

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

位	描述	
[31:0]	DATA	SDH 收发缓存 这个缓存用来收发数据，可以被CPU或是DMAC访问

DMAC控制和状态寄存器(SDH_DMACTL)

寄存器	偏移地址	R/W	描述	复位值
SDH_DMACTL	SDH_BA+0x400	R/W	DMAC 控制和状态寄存器	0x0000_0000

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

位	描述	
[31:10]	**Reserved**	保留.
[9]	**DMABUSY**	**DMA 传输正在进行** 这一位指示DMA是否正在传输. 0 = DMA传输不在进行中. 1 = DMA 传输正在进行
[8:4]	**Reserved**	保留.
[3]	**SGEN**	**Scatter-Gather 功能使能控制** 0 = Scatter-gather 功能禁止 (DMAC 从SDH_DMASA里设定的起始地址开始传输单独一个数据块). 1 = Scatter-gather 功能使能 (DMAC把SDH_DMASA 设定的起始地址作为物理地址描述符（PAD）表格的起始地址，PAD的格式会在下面描述).
[2]	**Reserved**	保留.
[1]	**DMARST**	**软件引擎复位** 0 = 没有影响 1 = 复位内部状态机和指针。控制寄存器的内容不会被清除。这一位会在接下来的几个时钟周期内自动清除 **注:软件复位DMA模块**
[0]	**DMAEN**	**DMAC引擎使能控制** 0 = DMAC 禁止. 1 = DMAC 使能. **注1:** 如果这一位清0, DMAC 会忽略所有DMA请求，并且让总线主控进入空闲模式 **注2:** 如果目标异常中止发生，此位清零

DMAC传输起始地址寄存器(SDH_DMASA)

寄存器	偏移地址	R/W	描述	复位值
SDH_DMASA	SDH_BA+0x408	R/W	DMAC 传输起始地址寄存器	0x0000_0000

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

位	描述	
[31:1]	**DMASA**	**DMA 传输起始地址** 这个区域的值补上最低位的0作为DMAC数据收发的系统内存（SRAM）的起始地址 如果 DMAC不是在一般模式，这个值作为物理地址描述（PAD）表格的起始地址
[0]	**ORDER**	**确定PAD 存取是否合法** 0 = PAD表格读取合法. 1 = PAD 读取不合法. **注:** 这一位只在scatter-gather模式下当 SGEN = 1.时有效

注: SRAM的起始地址必须字对齐，例如 0x0000_0000, 0x0000_0004…

PAD表格的格式如下。注PAD的字节数需要与DMAC中填写的字节数相符。最后一个描述符的EOT位要设置为1。

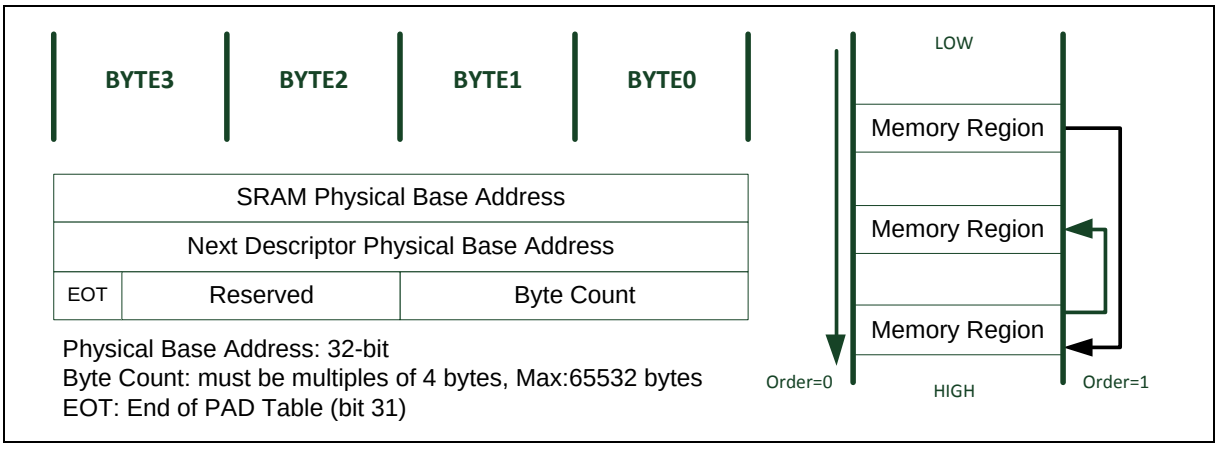


图 6.17-3 PAD表格格式

DMAC传输字节数寄存器 (SDH_DMABCNT)

寄存器	偏移地址	R/W	描述	复位值
SDH_DMABCNT	SDH_BA+0x40C	R	DMAC 传输字节数寄存器	0x0000_0000

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

位	描述	
[31:26]	Reserved	保留.
[25:0]	BCNT	DMA 传输字节数寄存器 （只读） 这个区域的值指示DMAC传输中还有多少字节需要传输。这个值只在DMAC传输时有效，否则，这个值会是0.

DMAC中断使能控制寄存器(SDH_DMAINTEN)

寄存器	偏移地址	R/W	描述	复位值
SDH_DMAINTEN	SDH_BA+0x410	R/W	DMAC 中断使能控制寄存器	0x0000_0001

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						WEOTIEN	ABORTIEN

位	描述	
[31:2]	Reserved	保留.
[1]	WEOTIEN	遇到错误 EOT 中断使能控制 0 = 禁止遇到错误EOT时产生中断 1 = 使能遇到错误EOT时产生中断
[0]	ABORTIEN	DMA 读写目标异常中止中断使能控制 0 = 当DMA传输时，目标异常中止中断禁止. 1 = 当DMA传输时，目标异常中止中断使能.

DMAC中断状态寄存器(SDH_DMAINTSTS)

寄存器	偏移地址	R/W	描述	复位值
SDH_DMAINTSTS	SDH_BA+0x414	R/W	DMAC 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						WEOTIF	ABORTIF

位	描述	
[31:2]	Reserved	保留.
[1]	WEOTIF	**遇到错误EOT中断标志** 当使能 DMA Scatter-Gather 功能，在DMA传输完成以前遇到了EOT标志 (这意味着PAD表格里的传输数量小于DMAC中设定的传输数量)，这一位会置位。 0 = 没有在DMA传输完成前遇到了EOT. 1 =在DMA传输完成前遇到了EOT. **注:** 此位只读，写1清0.
[0]	ABORTIF	**DMA 读写目标异常中止中断标志** 0 = 没有接收到总线错误. 1 = 接收到总线错误. **注:** 此位只读，写1清0 .

注: 当DMAC总线总控接收到错误返回。意味着发生了目标异常中止。DMAC会停止传输并把这个状态报给程序，然后去进入空闲模式。当目标异常中止发生或者WEOTIF置位，程序需要复位DMAC和外设，然后重新传输这些数据.

6.17.7.2 SD 寄存器描述

总体控制和状态寄存器 (SDH_GCTL)

寄存器	偏移地址	R/W	描述	复位值
SDH_GCTL	SDH_BA+0x800	R/W	总体控制和状态寄存器	0x0000_0000

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

位	描述	
[31:2]	Reserved	保留.
[1]	SDEN	SD功能使能控制 0 = SD 功能禁止. 1 = SD 功能使能.
[0]	GCTLRST	软件引擎复位 0 = 没有影响. 1 = 复位SDH引擎。控制寄存器不会被清除。这一位会在接下来的几个时钟内自动清除。

总体中断控制寄存器(SDH_GINTEN)

寄存器	偏移地址	R/W	描述	复位值
SDH_GINTEN	SDH_BA+0x804	R/W	总体中断控制寄存器	0x0000_0001

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							DTAIEN

位	描述	
[31:1]	Reserved	保留.
[0]	DTAIEN	DMAC 读写异常中止中断控制 0 = DMAC 读写异常中止中断禁止. 1 = DMAC 读写异常中止中断使能.

总体中断状态寄存器 (SDH_GINTSTS)

寄存器	偏移地址	R/W	描述	复位值
SDH_GINTSTS	SDH_BA+0x808	R/W	总体中断状态寄存器	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							DTAIF

位	描述	
[31:1]	Reserved	保留.
[0]	DTAIF	**DMAC 读写目标异常终止中断标志 (只读)** 这位指示DMAC当DMA读写操作时从内部AHB总线接收到一个错误反馈。当目标异常中止发生后，需要复位所有的引擎. 0 = 没接收到总线错误反馈. 1 =接收到总线错误反馈. **注:** 此位只读，写1清0.

不管中断是否使能，当相关事件发生时，中断标志都会置位.

SD控制和状态寄存器(SDH_CTL)

寄存器	偏移地址	R/W	描述	复位值
SDH_CTL	SDH_BA+0x820	R/W	SD 控制和状态寄存器	0x0101_0000

31	30	29	28	27	26	25	24
Reserved				SDNWR			
23	22	21	20	19	18	17	16
BLKCNT							
15	14	13	12	11	10	9	8
DBW	CTLRST	CMDCODE					
7	6	5	4	3	2	1	0
CLKKEEP0	CLK8OEN	CLK74OEN	R2EN	DOEN	DIEN	RIEN	COEN

位	描述	
[31:28]	**Reserved**	Reserved.
[27:24]	**SDNWR**	**数据块写操作的NWR参数** 这一位是数据块写操作的NWR参数，单位是SD 的时钟周期。实际的时钟周期个数是SDNWR+1.
[23:16]	**BLKCNT**	**收发数据块大小** 这个区域决定收发数据的数据块大小，在 READ_MULTIPLE_BLOCK 和 WRITE_MULTIPLE_BLOCK指令 下，程序可以利用这个功能加速数据传输来提高效能，不要在这个区域填0 . 在 READ_MULTIPLE_BLOCK and WRITE_MULTIPLE_BLOCK 指令下, 实际的长度是 BLKCNT * (BLKLEN +1).
[15]	**DBW**	**SD 数据宽度 (1-Bit / 4-Bit 模式选择)** 0 = 数据宽度为 1-bit. 1 = 数据宽度为 4-bit.
[14]	**CTLRST**	**软件引擎复位** 0 = 无影响. 1 = 复位内部状态机和计数器. 控制寄存器的内容不会清除 (但 RIEN, DIEN, DOEN 和 R2EN 会被清除). 这一位在几个时钟周期内会自动清0
[13:8]	**CMDCODE**	**SD 指令码** 这个寄存器是SD的指令码 (0x00 – 0x3F).
[7]	**CLKKEEP0**	**SD端口0时钟使能控制** 0 = SD 主控制器自动设置时钟输出与关闭. 1 = SD 时钟一直输出.

[6]	**CLK8OEN**	**产生8个时钟周期输出使能控制** 0 = 无影响. (请通过设置 SDH_CTL[CTLRST]来清这一位.) 1 = 使能，SD主控制器会输出8个时钟周期. **注:** 当操作结束之后，此位会自动清零，所以，不要对这位写0（控制器会工作不正常）.
[5]	**CLK74OEN**	**产生74个时钟周期输出使能控制** 0 = 无影响. (请通过设置 SDH_CTL[CTLRST]来清这一位.) 1 = 使能，SD主控制器会输出74个时钟周期. **注:** 当操作结束之后，此位会自动清零，所以，不要对这位写0（控制器会工作不正常）
[4]	**R2EN**	**应答 R2输入使能控制** 0 = 无影响. (请通过设置 SDH_CTL[CTLRST]来清这一位.) 1 = 使能, SD 控制器等待从SD卡接收 R2应答，并把数据存到 DMAC的闪存缓存 (计算 CRC-7). **注:** 当操作结束之后，此位会自动清零，所以，不要对这位写0（控制器会工作不正常）.
[3]	**DOEN**	**数据输出使能控制** 0 = 无影响. (请通过设置 SDH_CTL[CTLRST]来清这一位.) 1 = 使能，SD主控制器将发送数据和CRC-16 值到SD卡. **注:** 当操作结束之后，此位会自动清零，所以，不要对这位写0（控制器会工作不正常）.
[2]	**DIEN**	**数据输入使能控制** 0 = 无影响. (请通过设置 SDH_CTL[CTLRST]来清这一位.) 1 = 使能，SD主控制器将等待从SD卡接收数据和CRC-16 值. **注:** 当操作结束之后，此位会自动清零，所以，不要对这位写0（控制器会工作不正常）.
[1]	**RIEN**	**应答输入使能控制** 0 = 无影响. (请通过设置 SDH_CTL[CTLRST]来清这一位.) 1 = 使能, SD主控制器准备从SD卡接收应答. **注:** 当操作结束之后，此位会自动清零，所以，不要对这位写0（控制器会工作不正常）.
[0]	**COEN**	**指令输出使能控制** 0 = 无影响. (请通过设置 SDH_CTL[CTLRST]来清这一位.) 1 =使能 , SD 主控制器会向SD卡输出指令. **注:** 当操作结束之后，此位会自动清零，所以，不要对这位写0（控制器会工作不正常）.

SD指令参数寄存器 (SDH_CMDARG)

寄存器	偏移地址	R/W	描述	复位值
SDH_CMDARG	SDH_BA+0x824	R/W	SD指令参数寄存器	0x0000_0000

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

位	描述	
[31:0]	ARGUMENT	**SD 指令参数** 这个寄存器包含一个32-bit 值作为主控制器到SD卡的指令参数 。.在触发 SDH_CTL [COEN] 以前, 程序需要先填好这个值

SD中断控制寄存器 (SDH_INTEN)

寄存器	偏移地址	R/W	描述	复位值
SDH_INTEN	SDH_BA+0x828	R/W	SD 中断控制寄存器	0x0000_0A00

31	30	29	28	27	26	25	24
Reserved	CDSRC0	Reserved					
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved	Reserved	DITOIEN	RTOIEN	Reserved	SDHIEN0	Reserved	CDIEN0
7	6	5	4	3	2	1	0
Reserved						CRCIEN	BLKDIEN

位	描述	
[31]	**Reserved**	保留.
[30]	**CDSRC0**	**SD0 卡片检测信号选择** 0 = 来自 SD0卡片的 DAT3 管脚. 主控制器需要有时终来读DAT3管脚上的状态，需要确认SDH_CTL[CLKKEEP0] 是 1 保证时钟存在. 1 =来自GPIO管脚.
[29:15]	**Reserved**	保留.
[14]	**Reserved**	保留.
[13]	**DITOIEN**	**数据输入超时中断使能控制** 使能或关闭当数据输入超时时SD控制器产生中断，超时时间在 TOUT里设定. 0 = 禁止数据输入超时中断. 1 = 使能数据输入超时中断.
[12]	**RTOIEN**	**应答超时中断使能控制** 使能或关闭当接收应答或R2超时时SD控制器是否产生中断。 超时时间在 TOUT里设定. 0 =应答超时中断禁止. 1 = 使能应答超时中断.
[11]	**Reserved**	保留.
[10]	**SDHIEN0**	**SDH 端口0中断使能控制** 使能或禁止当SDH 卡 0从DAT【1】发送中断到SDH时，是否产生中断. 0 = SDH端口0中断禁止. 1 = SDH端口0中断使能.
[9]	**Reserved**	保留.

[8]	**CDIEN0**	**SD0卡片侦测中断使能控制** 使能或禁止当检测到卡片插入或弹出时是否产生中断. 0 = SD0 卡片侦测中断禁止. 1 = SD0卡片侦测中断使能.
[7:2]	**Reserved**	保留.
[1]	**CRCIEN**	**CRC7, CRC16 和 CRC 状态错误中断使能控制** 0 = SD 主控制器当发生 CRC7, CRC16 和 CRC状态错误时不产生中断. 1 = SD主控制器当发生 CRC7, CRC16 和 CRC状态错误时产生中断.
[0]	**BLKDIEN**	**快传输完成中断使能控制** 0 = SD 控制器当传输完成后不会产生中断. 1 = SD 控制器当传输完成后产生中断.

SD中断状态寄存器(SDH_INTSTS)

寄存器	偏移地址	R/W	描述	复位值
SDH_INTSTS	SDH_BA+0x82C	R/W	SD中断状态寄存器	0x000X_008C

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved					DAT1STS	Reserved	CDSTS0
15	14	13	12	11	10	9	8
Reserved		DITOIF	RTOIF	Reserved	SDHIF0	Reserved	CDIF0
7	6	5	4	3	2	1	0
DAT0STS	CRCSTS			CRC16	CRC7	CRCIF	BLKDIF

位	描述	
[31:19]	**Reserved**	保留.
[18]	**DAT1STS**	**SD端口 DAT1 管脚状态 (只读)** 这一位显示SD端口的DAT1的管脚状态.
[17]	**Reserved**	保留.
[16]	**CDSTS0**	**SD0卡片侦测状态 (只读)** 这一位显示SD0的卡片侦测状态，用来检测卡 如果 CDSRC0 (SDH_INTEN[30]) = 0， DAT3 f用来卡片侦测:. 0 = 卡片弹出. 1 = 卡片插入. 如果CDSRC0 (SDH_INTEN[30]) = 1, GPIO用来侦测:. 0 = 卡片插入. 1 = 卡片弹出.
[15:14]	**Reserved**	保留.
[13]	**DITOIF**	**数据输入超时中断标志 (只读)** 这一位指示SD主控制器计算输入超时时间是否超过（等待开始位）. 0 = 没有超时. 1 = 数据输入超时. **注:** 此位只读，写1清0.

[12]	**RTOIF**	**响应超时中断标志(只读)** 这一位指示SD主控制器计算等待应答或R2超时时间是否超过（等待开始位）. 0 = 没有超时. 1 = 响应超时. **注:** 此位只读，写1清0..
[11]	**Reserved**	保留.
[10]	**SDHIF0**	**SDH 0 中断标志 (只读)** 此位指示 SDH 卡片 0向控制器申请中断，这个中断是电平触发，在清这个标志之前，需要把SDHIEN0 (SDH_INTEN[10])先关掉. 0 = SDH 卡片 0没有申请中断. 1 = SDH 卡片 0 申请中断. **注:** 此位只读，写1清0
[9]	**Reserved**	保留.
[8]	**CDIF0**	**SD0 卡片侦测中断标志 (中断)** 此位指示 SD 卡 0 有插入或弹出，只有当CDIEN0 (SDH_INTEN[8]) 置 1,此位有效 0 =没有卡片插入或弹出. 1 = SD0有卡片插入或弹出. **注:** 此位只读，写1清0.
[7]	**DAT0STS**	**当前选择的SD端口 DAT0 管脚状态(只读)** 此位显示当前选择的SD端口DAT0的管脚状态
[6:4]	**CRCSTS**	**数据输出CRC状态值 (只读)** SD 主控制器会记录输出数据的CRC状态值。程序可以根据这一位判断数据输出的错误类型。. 010 = 有效 CRC 状态值. 101 = 无效 CRC 状态值. 111 = SD 卡 发生程序错误 Others = 保留.
[3]	**CRC16**	**数据输入CRC-16 校验状态 (只读)** SD主控制器会校验输入数据的CRC-16的校验值. 0 = CRC-16 传输不正确. 1 = CRC-16 传输正确.
[2]	**CRC7**	**CRC-7 校验状态 (只读)** SD host 计算每个响应的CRC-7的校验值。如果响应不包含CRC-7部分（如R3），程序需要关掉CRCIEN (SDH_INTEN[1]) 然后忽略此位. 0 = CRC-7 传输不正确. 1 = CRC-7 传输正确.

[1]	**CRCIF**	**CRC7, CRC16 和 CRC 状态错误中断标志 (只读)** 这一位指示有响应CRC错误或数据输入CRC错误或数据输出CRC状态错误发生，当CRC错误发生，程序需要复位SD引擎。有一些响应（如R3）没有CRC-7部分，SD主控制器仍然会计算CRC-7校验值，产生CRC错误，此位也会置位。在这种情况下，程序忽略这个错误并手动清除此位。. 0 =没有CRC错误发生. 1 = 有CRC错误发生. **注:** 此位只读，写1清0.
[0]	**BLKDIF**	**块传输完成中断标志 (只读)** 此位指示SD控制器完成了所有的数据输入输出的传输。如果在多数据块传输时有CRC16错误或CRC状态值错误发生，传输不正确，此位仍然会置1. 0 = 还没有完成. 1 = 已完成. **注:** 此位只读，写1清0.

SD接收应答标记寄存器0 (SDH_RESP0)

寄存器	偏移地址	R/W	描述	复位值
SDH_RESP0	SDH_BA+0x830	R	SD 接收应答标记寄存器 0	0x0000_0000

31	30	29	28	27	26	25	24
RESPTK0							
23	22	21	20	19	18	17	16
RESPTK0							
15	14	13	12	11	10	9	8
RESPTK0							
7	6	5	4	3	2	1	0
RESPTK0							

位	描述	
[31:0]	RESPTK0	**SD 接收应答标记 0** 当RIEN (SDH_CTL[1])为1时SD 主控制器会接收应答标记来得到回复 。这个区域包含应答标记的47-16位。.

SD 接收应答标记寄存器1 (SDH_RESP1)

寄存器	偏移地址	R/W	描述	复位值
SDH_RESP1	SDH_BA+0x834	R	SD 接收应答标记寄存器1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	22	21	20	19	18	17	16
Reserved							
15	14	13	12	11	10	9	8
Reserved							
7	6	5	4	3	2	1	0
RESPTK1							

位	描述	
[31:8]	Reserved	保留.
[7:0]	RESPTK1	SD Receiving Response Token 1 当RIEN (SDH_CTL[1])为1时SD 主控制器会接收应答标记来得到回复 。这个区域包含应答标记的15-8位.

SD 数据块长度寄存器(SDH_BLEN)

寄存器	偏移地址	R/W	描述	复位值
SDH_BLEN	SDH_BA+0x838	R/W	SD 数据块长度寄存器	0x0000_01FF

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					BLKLEN		
7	6	5	4	3	2	1	0
BLKLEN							

位	描述	
[31:11]	Reserved	保留.
[10:0]	BLKLEN	**SD 数据块长度字节数** 11位的值是SD传输数据块长度的字节数，实际的长度是BLKLEN +1. **注:** 默认的 SD数据块长度是 512 字节.

SD应答和数据输入超时寄存器(SDH_TOUT)

寄存器	偏移地址	R/W	描述	复位值
SDH_TOUT	SDH_BA+0x83C	R/W	SD 应答和数据输入超时寄存器	0x0000_0000

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

位	描述	
[31:24]	Reserved	保留.
[23:0]	TOUT	**SD 应答和数据输入超时时间** 24位的值用来设定应答和数据输入的超时时间。SD主控制器会等待开始位直到超时时间结束 。时钟周期由SD引擎时钟频率决定。不要设定过小的值，不然在超时前会收不到输入或应答。. **注:** 在这个寄存器填0会禁止硬件超时功能.

6.18 12位模数转换器 (ADC)

6.18.1 概述

NUC505系列包含一个12位的逐次逼近式的模数转换器 (ADC) ，有八个单端输入通道(ADC_CH0, ADC_CH1, … ADC_CH7). ADC_CH0 内部有一个 10 kΩ 电阻分压用来做电池检测。 ADC_CH2 也支持键盘比较器功能。用户可以通过设定SWTRG (ADC_CTL[0])来控制AD转换

6.18.2 特性

- 模拟电压输入范围: 0~AV_{DDADC}.
- 12-bit 解析度 10-bit 精度保证.
- 多达8路的单端输入通道
- ADC 时钟最高可达 16 MHz.
- ADC_CH1 通道最高可达1 MSPS的采样率.
- ADC_CH2, …ADC_CH7 通道最高可达200 kSPS的采样率.
- ADC内部采样时间可配置.
- 支持键盘比较器 (ADC_CH2).
- 内建 10 kΩ 电阻分压用作电池检测 (ADC_CH0).

6.18.3 框图

图 6.18-1 是整个ADC 控制的模块框图。ADC 8路单端模拟信号输入。通道选择由CHSEL (ADC_CTL[18:16])决定。

ADC_CH0 内建 10 kΩ 电阻分压用来做电池电压检测。分压电阻开关由CHSEL (ADC_CTL[18:16])决定。如果 CHSEL 是 0x0, t开关短接，相应的，如果CHSEL不是 0x0, 开关断开。

ADC_CH1 支持高采样率 (1 MSPS).。采样率由ADC的时钟频率确定 Conversion rate = (ADC clock frequency) / 16

“16” 意味着一次 A/D 采样转换需要 16个时钟。ADC_CH2 ~ ADC_CH7可达 200 kSPS 的采样率。ADC_CH0 因为10K电阻的关系需要更多的转换时间，,ADC_CH0 的采样率最高 100 KSPS.

ADC_CH2 也支持按键应用。 ADC 有按键比较器模块。当使能了按键比较器模块(设定 PDKEY (ADC_CTL[5]) 为 1), 如果ADC_CH2 不等于 AV_{DDADC}, 会设置标志 KEYIF (ADC_INTCTL[1]). KEYIF (ADC_INTCTL[1]) 写1清0.

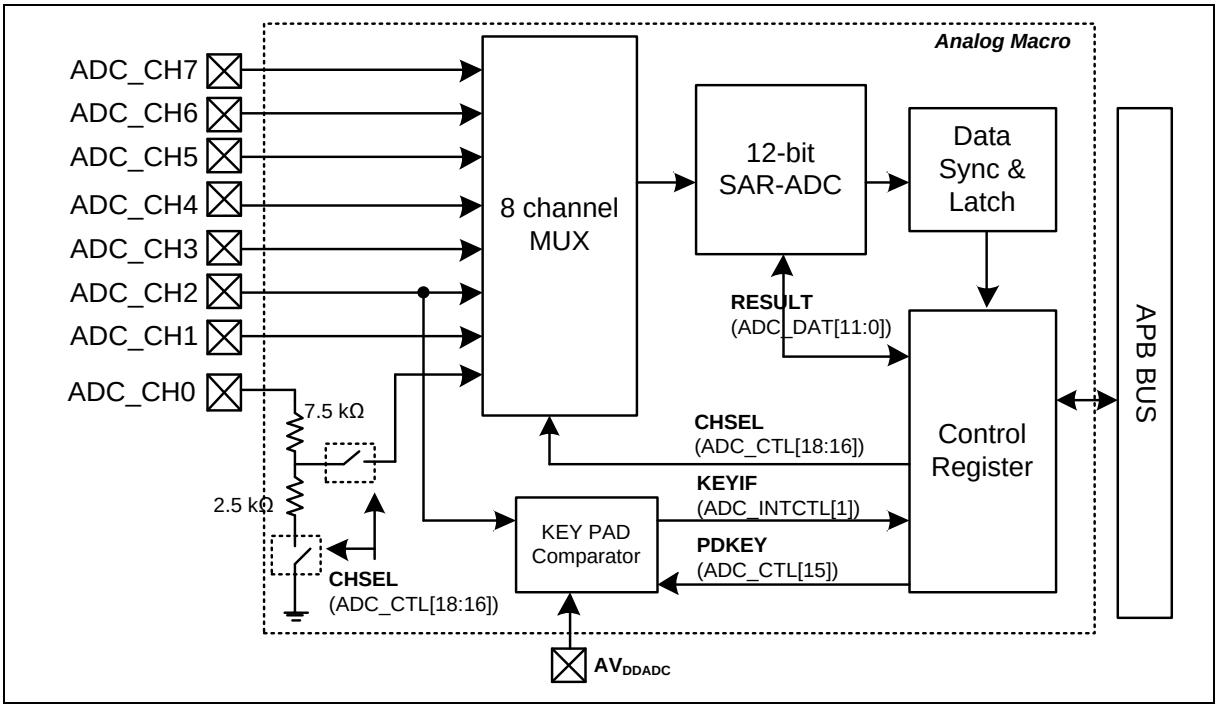


图 6.18-1 ADC 控制模块框图

6.18.4 工作描述

6.18.4.1 工作时序描述

在高速ADC时钟频率下的A/D转换，如果输入通道有比较高的阻抗比较长的充电时间，那么采样时时间有可能会不够长。用户可以通过设定 EXTSMPT (ADC_CTL[31:24])增加采样时间. 增加的采样时间长度可以是 0 ~255个 ADC时钟. 增加采样时间如图示 图 6.18-2. 如图 图 6.18-2, 一次 A/D 转换需要 16 个 ADC 时钟。

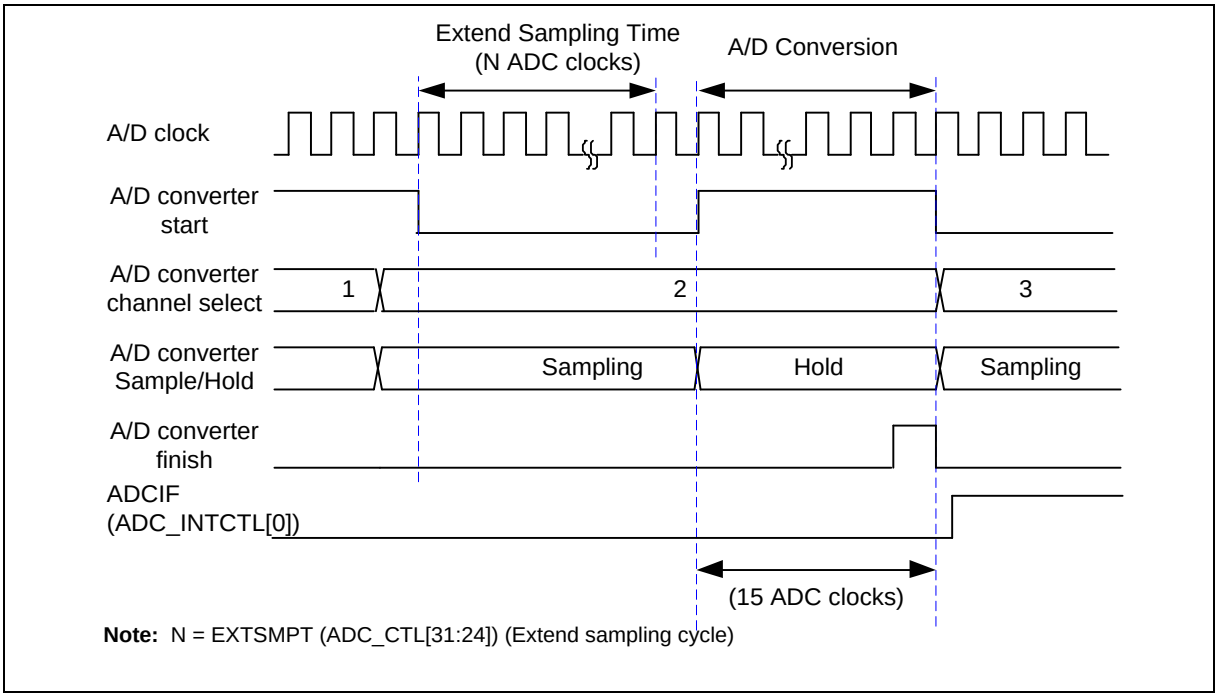


图 6.18-2 转换时序框图

6.18.4.2 按键应用

图 6.18-3 展示按键应用的示例电路。如果一个按键被按下， ADC_CH2上的电压会低于 AV_{DDADC}。按键比较器会比较ADC_CH2 和 AV_{DDADC}的电压， KEYIF (ADC_INTCTL[1]) 会置1.

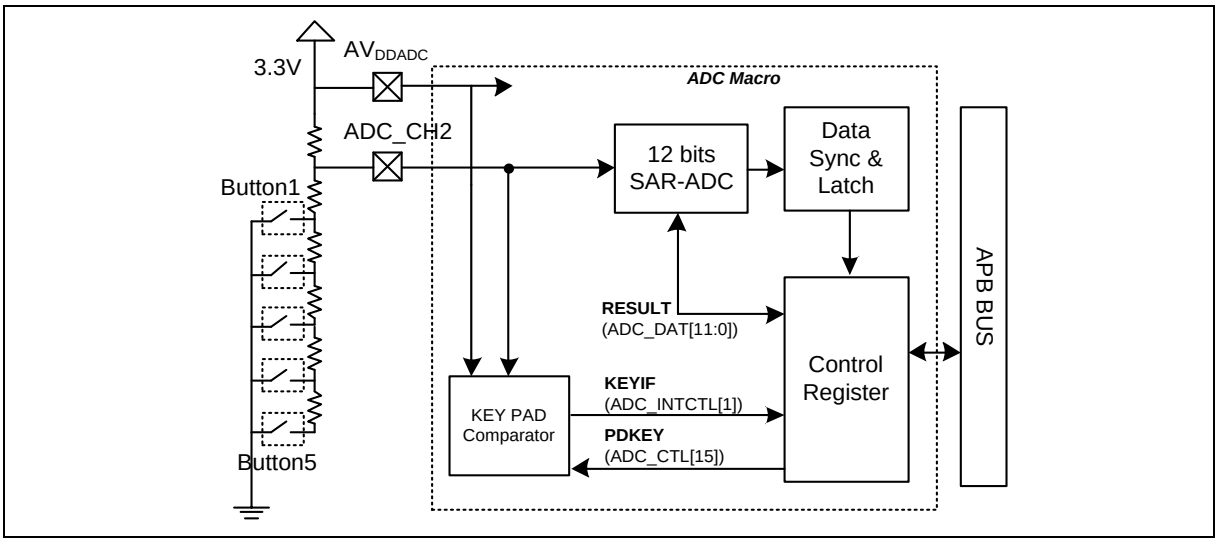


图 6.18-3 按键应用模块图

6.18.4.3 中断源

图 6.18-4 展示 ADC 的中断源。用户可以控制KEYIEN (ADC_INTCTL[9]) 和ADCIEN (ADC_INTCTL[8])

来单独选择中断源。

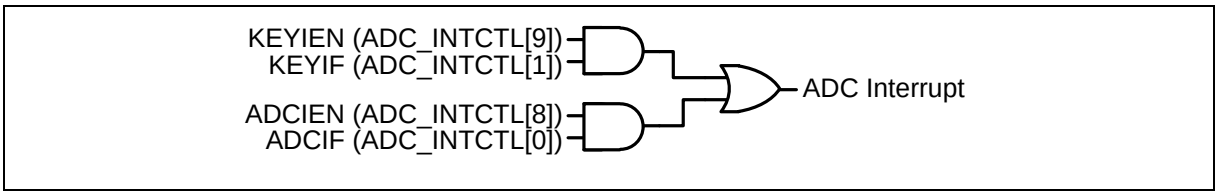


图 6.18-4 ADC 控制器中断

6.18.5 寄存器列表

R: 只读, **W**: 只可写, **R/W**: 可读写

寄存器	地址偏移	R/W	描述	复位值
ADC 基地址: **ADC_BA = 0x400E_2000**				
ADC_CTL	ADC_BA+0x00	R/W	ADC 控制寄存器	0x0400_AE00
ADC_INTCTL	ADC_BA+0x04	R/W	ADC 中断状态寄存器	0x0000_0000
ADC_DAT	ADC_BA+0x08	R	ADC 数据寄存器	0x0000_0000

6.18.6 寄存器描述

ADC控制寄存器 (ADC_CTL)

寄存器	偏移地址	R/W	描述	复位值
ADC_CTL	ADC_BA+0x00	R/W	ADC 控制寄存器	0x0400_AE00

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

位	描述	
[31:24]	**EXTSMPT**	**ADC 增加采样时间** 当高频率下的AD转换，高输入阻抗时采样时间可能会不够。程序可以增加额外的采样时间。. **注:** 单位是ADC时钟周期.
[23:19]	**Reserved**	保留.
[18:16]	**CHSEL**	**模拟输入信号选择** 000 = ADC_CH0. 001 = ADC_CH1. 010 = ADC_CH2. 011 = ADC_CH3. 100 = ADC_CH4. 101 = ADC_CH5. 110 = ADC_CH6. 111 = ADC_CH7. **注1:** ADC_CH0 用来做电池电压测量，内建了10K的电阻分压 **注2:** 用户需要特别注ADC_CH0的电流损耗，因为默认选择 ADC_CH0, 而 ADC_CH0 内建了分压电阻。. 如果 ADC_CH0 连接到电池，这是一个漏电的路径。所以当电池测量完成后，用户可以通过设定 CHSEL换到其他通道来切换通路.
[15]	**PDKEY**	**掉电键盘检测** 0 = 禁止掉电键盘检测. 1 = 使能掉电键盘检测.

[14]	**Reserved**	保留.
[13]	**PD**	**ADC掉电** 0 = ADC 正常工作. 1 = ADC 进入掉电状态. **注1:** ADC 在触发转换前必须先上电 **注2:** 在设定PD从1到0的时候需要等待 100ms让模拟电路稳定 。.
[12:1]	**Reserved**	保留.
[0]	**SWTRG**	**A/D 转换开始** 触发一次 A/D 转换. 0 = A/D 转换空闲. 1 = 开始转换. **注:** 这一位会自动清0.

ADC中断状态寄存器(ADC_INTCTL)

寄存器	偏移地址	R/W	描述	复位值
ADC_INTCTL	ADC_BA+0x04	R/W	ADC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						KEYIEN	ADCIEN
7	6	5	4	3	2	1	0
Reserved						KEYIF	ADCIF

位	描述	
[31:10]	Reserved	保留.
[9]	KEYIEN	**按键中断使能控制** 0 = 禁止按键中断. 1 = 使能按键中断.
[8]	ADCIEN	**ADC 中断使能控制** 0 = 禁止ADC转换结束中断. 1 = 使能ADC转换结束中断.
[7:2]	Reserved	保留.
[1]	KEYIF	**按键中断标志** 在按键检测流程里， he KEYIF标记按键状态。如果ADC_CH2 用做键盘应用，当ADC_CH2 不等于 AV_{DDADC},中断标志位是1. 0 = 按键没有按下 1 = 按键按下.
[0]	ADCIF	**ADC 转换完成中断标志** 当完成了一次转换流程， ADCIF 会置1，如果ADCIEN 是1，会产生中断到 NVIC. 0 = ADC 转换完成标志没有设置. 1 = ADC 转换完成标志已设置.

ADC数据寄存器(ADC_DAT)

寄存器	偏移地址	R/W	描述	复位值
ADC_DAT	ADC_BA+0x08	R	ADC数据寄存器	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				RESULT			
7	6	5	4	3	2	1	0
RESULT							

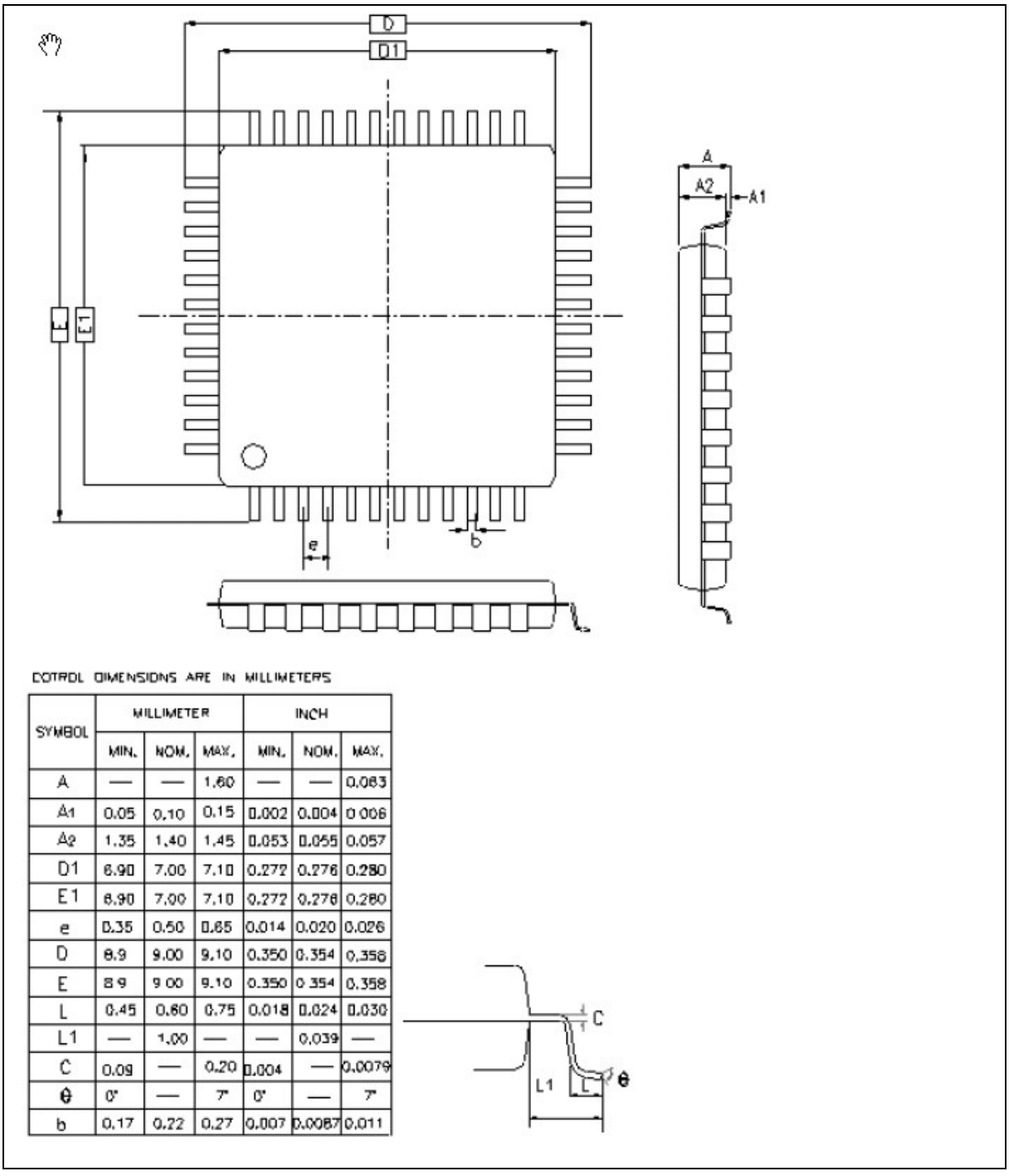
位	描述	
[31:12]	Reserved	保留.
[11:0]	RESULT	**A/D 转换结果** 这个寄存器包含ADC的转换结果，当A/D 转换完成, 12-bit ADC的转换结果会以无符号数的形式存入RESULT[11:0].

7 电气特性

NUC505系列的电气特性，请参考 NuMicro™ NUC505 Series Datasheet.

8 封装尺寸

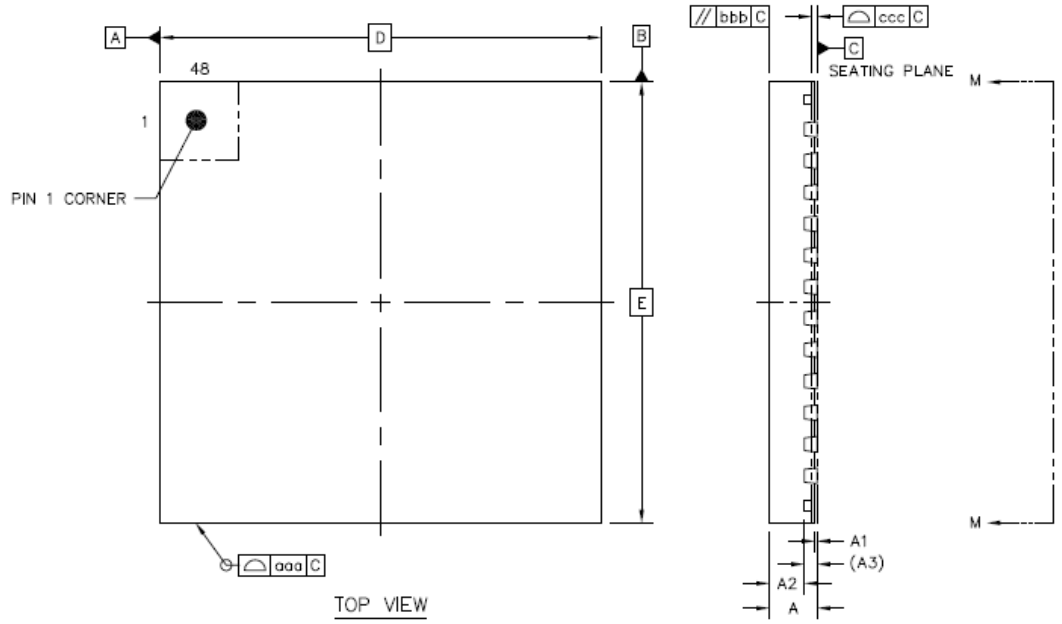
8.1 LQFP 48L (7x7x1.4mm footprint 2.0mm)



CONTROL DIMENSIONS ARE IN MILLIMETERS

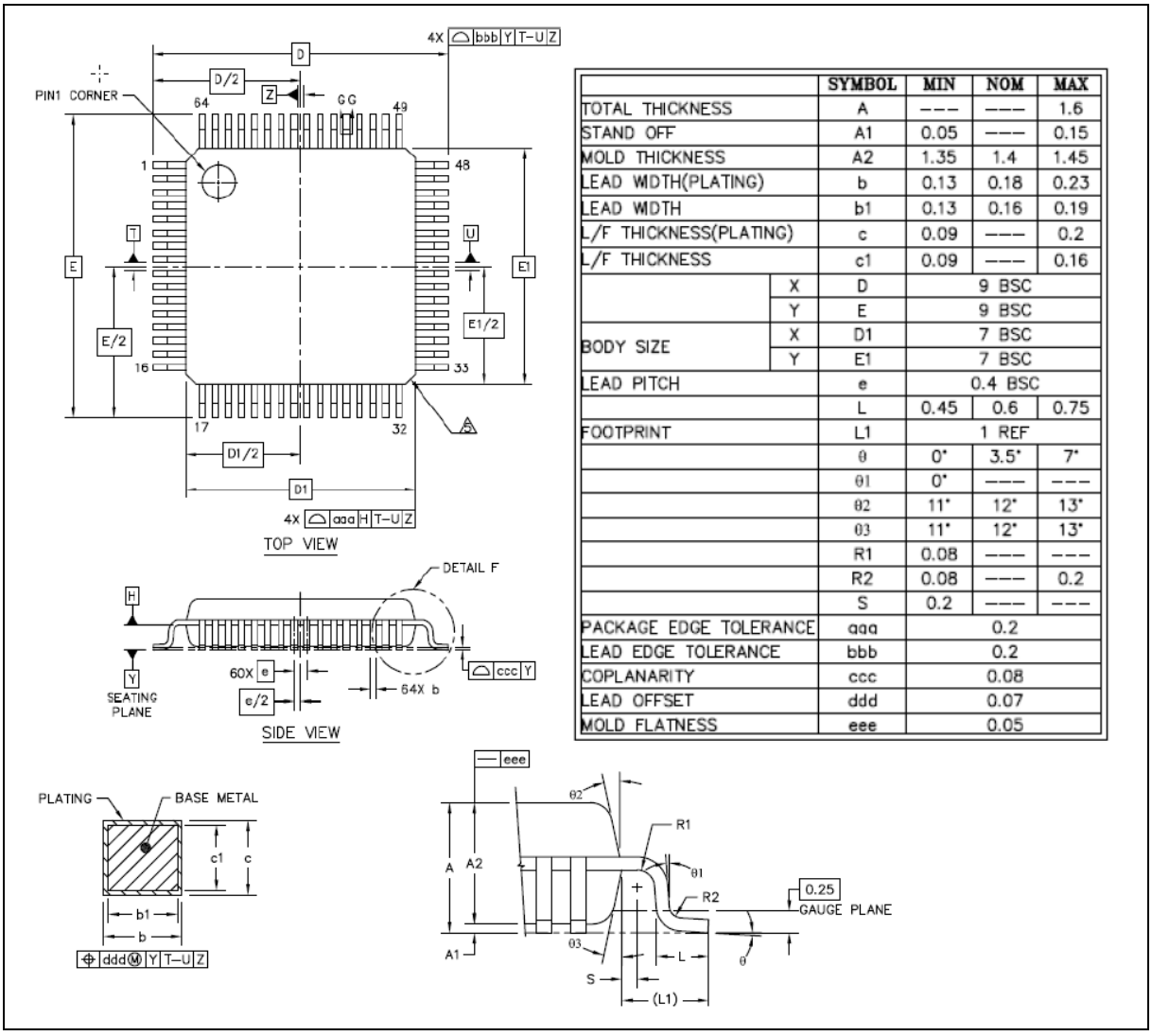
SYMBOL	MILLIMETER			INCH		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	—	—	1.60	—	—	0.063
A1	0.05	0.10	0.15	0.002	0.004	0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
D1	6.90	7.00	7.10	0.272	0.276	0.280
E1	6.90	7.00	7.10	0.272	0.276	0.280
e	0.35	0.50	0.65	0.014	0.020	0.026
D	8.9	9.00	9.10	0.350	0.354	0.358
E	8.9	9.00	9.10	0.350	0.354	0.358
L	0.45	0.60	0.75	0.018	0.024	0.030
L1	—	1.00	—	—	0.039	—
C	0.09	—	0.20	0.004	—	0.0079
θ	0°	—	7°	0°	—	7°
b	0.17	0.22	0.27	0.007	0.0087	0.011

8.2 QFN 48 (7x7x0.8mm)



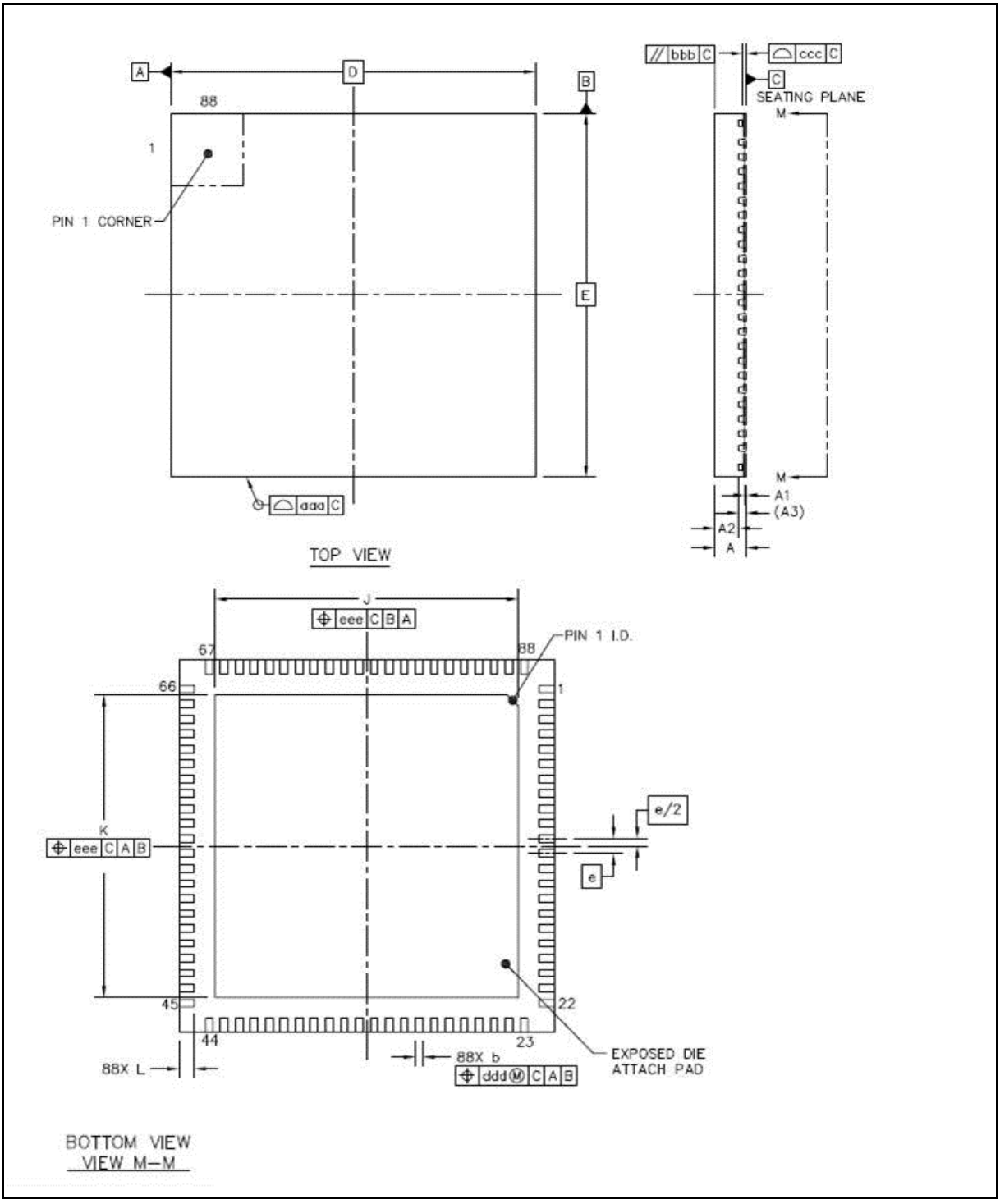
		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.7	0.75	0.8
STAND OFF		A1	0	0.035	0.05
MOLD THICKNESS		A2	---	0.55	---
L/F THICKNESS		A3		0.203 REF	
LEAD WIDTH		b	0.2	0.25	0.3
BODY SIZE	X	D		7 BSC	
	Y	E		7 BSC	
LEAD PITCH		e		0.5 BSC	
EP SIZE	X	J	5.2	5.3	5.4
	Y	K	5.2	5.3	5.4
LEAD LENGTH		L	0.35	0.4	0.45
PACKAGE 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	
		S		0.45 REF	

8.3 LQFP 64L (7x7x1.4mm footprint 2.0mm)



		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	---	---	1.6
STAND OFF		A1	0.05	---	0.15
MOLD THICKNESS		A2	1.35	1.4	1.45
LEAD WIDTH(PLATING)		b	0.13	0.18	0.23
LEAD WIDTH		b1	0.13	0.16	0.19
L/F THICKNESS(PLATING)		c	0.09	---	0.2
L/F THICKNESS		c1	0.09	---	0.16
BODY SIZE	X	D	9 BSC		
	Y	E	9 BSC		
	X	D1	7 BSC		
	Y	E1	7 BSC		
LEAD PITCH		e	0.4 BSC		
		L	0.45	0.6	0.75
FOOTPRINT		L1	1 REF		
		θ	0°	3.5°	7°
		θ1	0°	---	---
		θ2	11°	12°	13°
		θ3	11°	12°	13°
		R1	0.08	---	---
		R2	0.08	---	0.2
		S	0.2	---	---
PACKAGE EDGE TOLERANCE		aaa	0.2		
LEAD EDGE TOLERANCE		bbb	0.2		
COPLANARITY		ccc	0.08		
LEAD OFFSET		ddd	0.07		
MOLD FLATNESS		eee	0.05		

8.4 QFN 88 (10x10x0.9mm)



		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.8	0.85	0.9
STAND OFF		A1	0	0.035	0.05
MOLD THICKNESS		A2	---	0.65	0.67
L/F THICKNESS		A3		0.203 REF	
LEAD WIDTH		b	0.15	0.2	0.25
BODY SIZE	X	D		10 BSC	
	Y	E		10 BSC	
LEAD PITCH		e		0.4 BSC	
EP SIZE	X	J	8	8.1	8.2
	Y	K	8	8.1	8.2
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	

NOTES

1.0 COPLANARITY APPLIES TO LEADS, CORNER LEADS AND DIE ATTACH PAD.

9 修订历史

日期	版本	描述
2014.10.30	1.01	1. 初始发布
2015.05.08	1.02	1.加入图 6.14 2 内部 CODEC 框图.
2015.09.10	1.03	1. 加入新型号: NUC505DLA, NUC505YLA, and NUC505DSA 在章节 4. 2. 更新在新的型号内部SPI 闪存容量512 KB.
2015.11.04	1.04	1. 添加了一条说明，表示NUC505DS13Y仅支持4.1.1节中的耳机输出 2. 添加了一个说明，在4.1.1节中相同封装但引脚并不兼容 3. 加入8.2节QFN 48（7x7x0.8mm）封装规格 4. 在4.1.1，4.2.4和4.3.4节中增加了NUC505YLA2Y新型号 5. 在6.15.6节中增加了LOWPWREN（USBD_PHYCTL [3]）的描述 6. 分别使用空闲模式和掉电模式替换睡眠模式和深度睡眠模式的电源名称
2016.05.09	1.05	1. 为QFN 48/88引脚封装的引脚图和引脚说明添加了一个注释。QFN封装底部的散热垫（EPD）应连接到GND.
2016.12.02	1.06	1. 修正4.2.4/4.2.5/4.2.6节中管脚配置以及4.3.5节中管脚描述 2. 修正NUC505DLA 和NUC505YLA的SPI组数为2. 3. 修改引脚描述从VBUS为VBUS33.
2017.08.10	1.07	1. 修正VBUS33引脚描述. 2. 修正6.2.4节中USB模块供电描述. 3. 修正6.14.9节中CODEC寄存器名称. 4. 修正6.3.2节中USB Host时钟源为仅能为PLL. 5. 修正4.3.7节中VCMBF引脚描述. 6. 修正6.16.7节中寄存器HCFS内容描述. 7. 在6.13.6节中添加了SPIM寄存器的描述.
2019.01.16	1.08	1. 在6.9.5节中添加了RTC初始化启动的设定流程. 2. 修正6.12.2节中SPI特性描述 3. 修正图 6.3 1时钟发生器全局框图

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.