

ARM Cortex™ -M0

32位微控制器

NuMicro™ NUC122 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.

目录

1 概述	10
2 特性	11
2.1 NuMicro™ NUC122 特征	11
3 选型表和引脚图	14
3.1 NuMicro™ NUC122 产品选型指南	14
3.2 引脚图	15
3.2.1 NuMicro™ NUC122 LQFP 64-pin	15
3.2.2 NuMicro™ NUC122 LQFP 48-pin	16
3.2.3 NuMicro™ NUC122 QFN 33-pin	17
3.3 引脚功能描述	18
3.3.1 NuMicro™ NUC122 引脚定义	18
4 NUC122 框图	22
5 功能描述	23
5.1 ARM® Cortex™ -M0 内核	23
5.2 系统管理器	25
5.2.1 概述	25
5.2.2 系统复位	25
5.2.3 系统电源分配	25
5.2.4 系统内存映射	27
5.2.5 系统管理器控制寄存器	29
5.2.6 系统定时器 (SysTick)	53
5.2.7 嵌套向量中断控制器 (NVIC)	58
5.2.8 系统控制寄存器	81
5.3 时钟控制器	90
5.3.1 概述	90
5.3.2 时钟发生器	92
5.3.3 系统时钟与 SysTick 时钟	93
5.3.4 外围设备时钟	93
5.3.5 掉电模式时钟	93
5.3.6 寄存器映射	94
5.3.7 寄存器描述	95
5.4 USB 设备控制器 (USB)	110
5.4.1 概述	111
5.4.2 特征	111
5.4.3 框图	112

5.4.4 功能描述	112
5.4.5 寄存器映射	117
5.4.6 寄存器描述	119
5.5 通用 I/O (GPIO).....	137
5.5.1 概述和特征	137
5.5.2 功能描述	137
5.5.3 寄存器映射	138
5.5.4 寄存器描述	143
5.6 I ² C总线控制器 (Master/Slave) (I ² C)	155
5.6.1 概述	155
5.6.2 协议寄存器	158
5.6.3 寄存器映射	162
5.6.4 寄存器描述	163
5.6.5 操作模式	171
5.6.6 数据传输5种操作	172
5.7 PWM 发生器和捕捉定时器 (PWM)	178
5.7.1 简介	178
5.7.2 特性	179
5.7.3 框图	180
5.7.4 功能描述	182
5.7.5 寄存器映射	188
5.7.6 寄存器描述	190
5.8 实时时钟 (RTC).....	211
5.8.1 简介	211
5.8.2 特征	211
5.8.3 框图	212
5.8.4 功能描述	213
5.8.5 寄存器映射	215
5.8.6 寄存器描述	216
5.9 串行外围设备接口 (SPI).....	230
5.9.1 概述	230
5.9.2 特征	230
5.9.3 框图	231
5.9.4 功能描述	232
5.9.5 编程例程	243
5.9.6 寄存器映射	245
5.9.7 寄存器描述	246

5.10 定时器控制器 (TMR)	256
5.10.1 概述	256
5.10.2 特征	256
5.10.3 框图	257
5.10.4 功能描述	258
5.10.5 寄存器映射	260
5.10.6 寄存器描述	261
5.11 看门狗定时器 (WDT)	266
5.11.1 特征	267
5.11.2 框图	268
5.11.3 寄存器映射	269
5.11.4 寄存器描述	270
5.12 UART 接口控制器 (UART)	272
5.12.1 概述	272
5.12.2 特征	273
5.12.3 框图	275
5.12.4 IrDA 模式	278
5.12.5 RS-485 功能模式	280
5.12.6 寄存器映射	281
5.12.7 寄存器描述	283
5.13 PS/2 设备控制器 (PS2D)	306
5.13.1 概述	306
5.13.2 特性	306
5.13.3 系统框图	307
5.13.4 功能描述	308
5.13.5 寄存器映射	312
5.13.6 寄存器描述	313
6 FLASH 内存控制器 (FMC)	320
6.1 概述	320
6.2 特征	320
6.3 框图	321
6.4 Flash 内存结构	322
6.5 启动选择	324
6.6 数据 Flash	324
6.7 用户配置	325
6.8 在系统编程 (ISP)	327

6.8.1 ISP程序	327
6.9 寄存器映射	330
6.10 寄存器描述	331
7 电气特性	338
7.1 绝对最大额定值	338
7.2 DC电气特性	339
7.2.1 NuMicro™ NUC122 DC 电气特性	339
7.3 AC 电气特性	343
7.3.1 外部 4~24 MHz 高速晶振交流特征	343
7.3.2 外部 4~24 MHz 高速晶振	343
7.3.3 外部 32.768 KHz 低速晶振	344
7.3.4 内部 22.1184 MHz 高速振荡器	344
7.3.5 内部 10 KHz 低速振荡器	344
7.4 模拟特性	345
7.4.1 LDO和电源管理规格	345
7.4.2 低压复位规格	346
7.4.3 欠压检测规格	346
7.4.4 上电复位规格 (5 V)	346
7.4.5 USB PHY 规格	347
7.5 SPI 动态特性	348
7.5.1 数据输入输出的动态特性	348
8 封装定义	350
8.1 64L LQFP (7x7x1.4 mm footprint 2.0 mm)	350
8.2 48L LQFP (7x7x1.4 mm footprint 2.0 mm)	351
8.3 33L QFN (5x5x0.8 mm)	352
9 版本历史	353

图

图 4-1 NuMicro™ NUC122 框图	22
图 5-1 功能框图	23
图 5-2 NuMicro™ NUC122 电源分配图	26
图 5-3 时钟发生器全局框图	91
图 5-4 时钟发生器框图	92
图 5-5 系统时钟框图	93
图 5-6 SysTick 时钟控制框图	93
图 5-7 USB 框图	112
图 5-8 唤醒中断的操作流程	114
图 5-9 端点 SRAM 的结构	115
图 5-10 数据传入时间里传输流程图	115
图 5-11 数据输出图	116
图 5-12 推挽输出	137
图 5-13 开漏输出	138
图 5-14 准双端 I/O 模式	138
图 5-15 I ² C 总线时序	155
图 5-16 I ² C 协议	156
图 5-17 主机向从机传输数据	156
图 5-18 主机由从机读取地址	157
图 5-19 START 和 STOP 条件	157
图 5-20 I ² C 总线上位传输	158
图 5-21 I ² C 总线上应答信号	158
图 5-22 I ² C 数据移位方向	159
图 5-23 I ² C 超时计数器框图	161
图 5-24 传输流程图	172
图 5-25 主机传输模式	173
图 5-26 主机接收模式	174
图 5-27 从机传输模式	175
图 5-28 从机接收模式	176
图 5-29 全呼模式	177
图 5-30 PWM 发生器 0 时钟源控制	180

图 5-31 PWM 发生器 0 结构框图	180
图 5-32 PWM 发生器 2 时钟源控制.....	181
图 5-33 PWM 发生器 2 结构框图	181
图 5-34 PWM 定时器内部比较器输出.....	182
图 5-35 PWM 定时器操作时序	183
图 5-36 PWM 双缓存图解.....	183
图 5-37 PWM 控制输出占空比	184
图 5-38 PWM 对输出带死区发生器操作	184
图 5-39 捕捉操作时序	185
图 5-40 PWM A组 PWM-定时器中断结构图.....	186
图 5-41 RTC 框图	212
图 5-42 SPI 框图.....	231
图 5-43 SPI 主机模式应用框图.....	232
图 5-44 SPI 从机模式应用框图.....	232
图 5-45 可调串行时钟频率.....	234
图 5-46 32-Bit in one Transaction	234
图 5-47 一次传输两个报文(Burst Mode).....	235
图 5-48 Byte 重排序	236
图 5-49 字节休眠时序波形.....	237
图 5-50 FIFO 模式框图	238
图 5-51 FIFO 模式时序图	239
图 5-52 SPI 在主机模式下的时序	240
图 5-53 SPI 主机模下的时序 (Alternate Phase of SPICLK)	241
图 5-54 SPI 在从机模式下的时序	241
图 5-55 SPI 在从机模式下的时序 (Alternate Phase of SPICLK)	242
图 5-56 定时器控制器框图	257
图 5-57 定时器控制的时钟源	257
图 5-58 连续计数模式	259
图 5-59 中断定时和复位信号时序	267
图 5-60 看门狗定时时钟控制	268
图 5-61 看门狗定时器框图	268
图 5-62 UART 时钟控制图	275

图 5-63 UART 框图.....	275
图 5-64 自动流控制框图	277
图 5-65 IrDA 框图	278
图 5-66 IrDA TX/RX 时序图.....	279
图 5-67 RS-485 帧结构	281
图 5-68 PS/2 设备框图	307
图 5-69 设备向主机传输数据格式.....	309
图 5-70 主机向设备传输的数据格式.....	309
图 5-71 PS/2 Bit数据格式.....	310
图 5-72 PS/2 总线时序	310
图 5-73 PS/2 数据格式	311
图 6-1 Flash内存控制框图	321
图 6-2 Flash内存组织	323
图 6-3 Flash内存结构	324
图 6-4 ISP操作时序图.....	327
图 6-5 ISP操作流图	328
图 7-1 典型晶振应用电路.....	343
图 7-2 SPI 主机动态特性时序图	349
图 7-3 SPI 从机动态特性时序图	349

表格

表 1-1 所支持的接口列表.....	10
表 5-1 片上控制器的地址空间分配	28
表 5-2 异常模式	59
表 5-3 系统中断映射	60
表 5-4 向量表格式.....	60
表 5-5 掉电模式控制表	97
表 5-6 字节排序和字节休眠条件.....	237
表 5-7 看门狗定时溢出间隔选择.....	266
表 5-8 UART 波特率公式.....	272
表 5-9 UART 波特率设置表	273
表 5-10 在软件模式下UART中断源和标志表.....	298
表 5-11 波特率方程表	301
表 6-1 内存地址表.....	322
表 6-2 ISP 模式.....	329

1 概述

NuMicro™ NUC122 Advanced Line内嵌Cortex™-M0核，最高可运行至60 MHz，内建32K/64K字节的Flash存储器，以及4K/8K字节SRAM，4K字节用于存储ISP引导代码的ROM，4K字节的数据FLASH，另外还有丰富的外设，如定时器，看门狗定时器，RTC，UART，SPI，I²C，PWM定时器，GPIO，USB 2.0 全速从机模式，低电压复位控制和欠压检测功能。

Product Line	UART	SPI	I ² C	USB	PS/2
NUC122	Y	Y	Y	Y	Y

表 1-1 所支持的接口列表

2 特性

2.1 NuMicro™ NUC122 特征

- 内核
 - ARM® Cortex™-M0内核最高运行60 MHz
 - 一个 24-位系统定时器
 - 低功耗掉电模式
 - 单指令周期32位硬件乘法器
 - 嵌套向量中断控制器NVIC 用于控制32个中断源，每个中断源可设置4个优先级
 - 支持串行线调试（SWD）带2个观察点/4个断点
- 内建LDO, 宽电压工作范围为2.5 V 到 5.5 V
- Flash 存储器
 - 32K/64K字节FLASH用于存储程序代码
 - 4KB FLASH用于存储ISP引导代码
 - 支持在系统编程 (ISP)方式更新应用程序
 - 支持512 字节单页擦除
 - 4K字节数据FLASH
 - 通过SWD/ICE接口，支持2线 ICP升级方式
 - 支持外部编程器并行高速编程模式
- SRAM 存储器
 - 4K/8K 字节内建SRAM
- 时钟控制
 - 针对不同应用可灵活选择时钟
 - 内部 22.1184 MHz 高速振荡器可用于系统运行
 - ◆ 在+25 °C , $V_{DD} = 3.3$ V时, 精度校正到± 1 %
 - ◆ 在-40 °C ~ +85 °C 和 $V_{DD} = 2.5$ V ~ 5.5 V范围内, 精度为± 5 %
 - 内部低功耗 10 KHz 低速振荡器用于看门狗及掉电模式唤醒等功能
 - 支持一组PLL, 高至 60MHz, 用于高性能的系统运行
 - 外部 4~24 MHz 高速晶振输入用于精准的时序运作
 - 外部 32.768 KHz 低速晶振输入用于RTC及低功耗模式操作
- GPIO
 - 四种I/O模式:
 - ◆ 准双向模式
 - ◆ 推挽输出模式
 - ◆ 开漏输出模式
 - ◆ 高阻输入模式
 - TTL/Schmitt触发输入可选
 - I/O管脚可被配置为边沿/电平触发模式的中断源
 - 支持大电流驱动/灌入I/O模式

- Timer
 - 支持4组32位定时器，每个定时器包括一个24位向上计数定时器和一个8位预分频计数器
 - 计数自动加载
- Watch Dog Timer
 - 多个时钟源可选
 - 从1.6ms到26316.8ms有8个可选的定时溢出周期(取决于所选的时钟源)
 - WDT可用作掉电模式的唤醒
 - 看门狗定时溢出的中断/复位选择
- RTC
 - 通过频率补偿寄存器(FCR) 支持软件频率补偿功能
 - 支持RTC计数(秒, 分, 小时) 及万年历功能(日, 月, 年)
 - 支持闹铃寄存器 (秒, 分, 小时, 日, 月, 年)
 - 可选择为12小时制或24小时制
 - 闰年自动识别
 - 支持周期时间滴答中断
 - 支持唤醒功能
- PWM/Capture
 - 内建两个16位PWM产生器,可输出4路PWM或2组互补配对PWM
 - 每个PWM产生器配有一个时钟源选择器, 一个时钟分频器, 一个8位时钟预分频和一个用于互补配对PWM的死区发生器
 - 4路16位捕捉定时器(共享PWM定时器)提供8路输入的上升/下降沿的捕捉功能
 - 支持捕捉(Capture)中断
- UART
 - 两组UART控制器
 - 支持流控(TXD, RXD, CTS and RTS)
 - UART 带16-字节FIFO用于标准模式
 - 支持IrDA(SIR)协议
 - 支持 RS-485 9 位模式和方向控制.
 - 可编程波特率发生器频率高至1/16系统时钟
- SPI
 - 两组SPI控制器
 - 主机速率高至 25 Mbps , 从机高至12Mbps (5V工作电压)
 - 支持 SPI 主机/从机模式
 - 全双工同步串行数据传输
 - 可变数据长度(从1位至32位)传输模式
 - 可设置MSB 或LSB 优先的传输模式
 - 当作为主机时2条从机片选线, 作为从机时1条从机片选线
 - 支持32位传输模式下的字节睡眠模式

- I²C
 - 一组I²C 设备
 - 支持主/从机模式
 - 主从机之间双向数据传输
 - 多主机总线支持(无中心主机)
 - 多主机同时传输数据时仲裁，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
 - 串行同步时钟可作为握手方式控制总线上数据暂停及恢复传送
 - 可编程的时钟适用于不同速率控制
 - I²C总线上支持多地址识别(4个从机地址带mask选项)
- USB 2.0全速从机模式
 - USB 2.0 12Mbps
 - 内建USB收发器
 - 四个中断事件，一个中断源
 - 支持控传输，批量传输，中断传输和同步传输
 - 支持6组可编程端点(endpoints)
 - 512字节内部SRAM作为USB的缓存区
 - 支持远程唤醒功能
- 欠压检测(Brown-out detector)
 - 支持四级检测电压: 4.5 V/3.8 V/2.7 V/2.2 V
 - 支持欠压中断和复位选择
- 内建LDO
- 低压复位
- 工作温度: -40 °C ~ 85 °C
- 封装:
 - 无铅封装 (RoHS)
 - LQFP 64-pin (7X7mm)
 - LQFP 48-pin
 - QFN 33-pin

3 选型表和引脚图

3.1 NuMicro™ NUC122 产品选型指南

Part number	Flash (KB)	ISP ROM (KB)	SRAM (KB)	I/O	Timer	Connectivity						I ² S	Comp.	PWM	ADC	RTC	ISP ICP	Package
						UART	SPI	I ² C	USB	LIN	PS/2							
NUC122ZD2AN	64 KB	4KB	8 KB	up to 18	4x32-bit	1	2	1	1	-	-	-	-	-	-	-	v	QFN33
NUC122ZC1AN	32 KB	4KB	4 KB	up to 18	4x32-bit	1	2	1	1	-	-	-	-	-	-	-	v	QFN33
NUC122LD2AN	64 KB	4KB	8 KB	up to 30	4x32-bit	2	2	1	1	-	1	-	-	4	-	v	v	LQFP48
NUC122LC1AN	32 KB	4KB	4 KB	up to 30	4x32-bit	2	2	1	1	-	1	-	-	4	-	v	v	LQFP48
NUC122SD2AN	64 KB	4KB	8 KB	up to 41	4x32-bit	2	2	1	1	-	1	-	-	4	-	v	v	LQFP64
NUC122SC1AN	32 KB	4KB	4 KB	up to 41	4x32-bit	2	2	1	1	-	1	-	-	4	-	v	v	LQFP64

3.2 引脚图

3.2.1 NuMicro™ NUC122 LQFP 64-pin

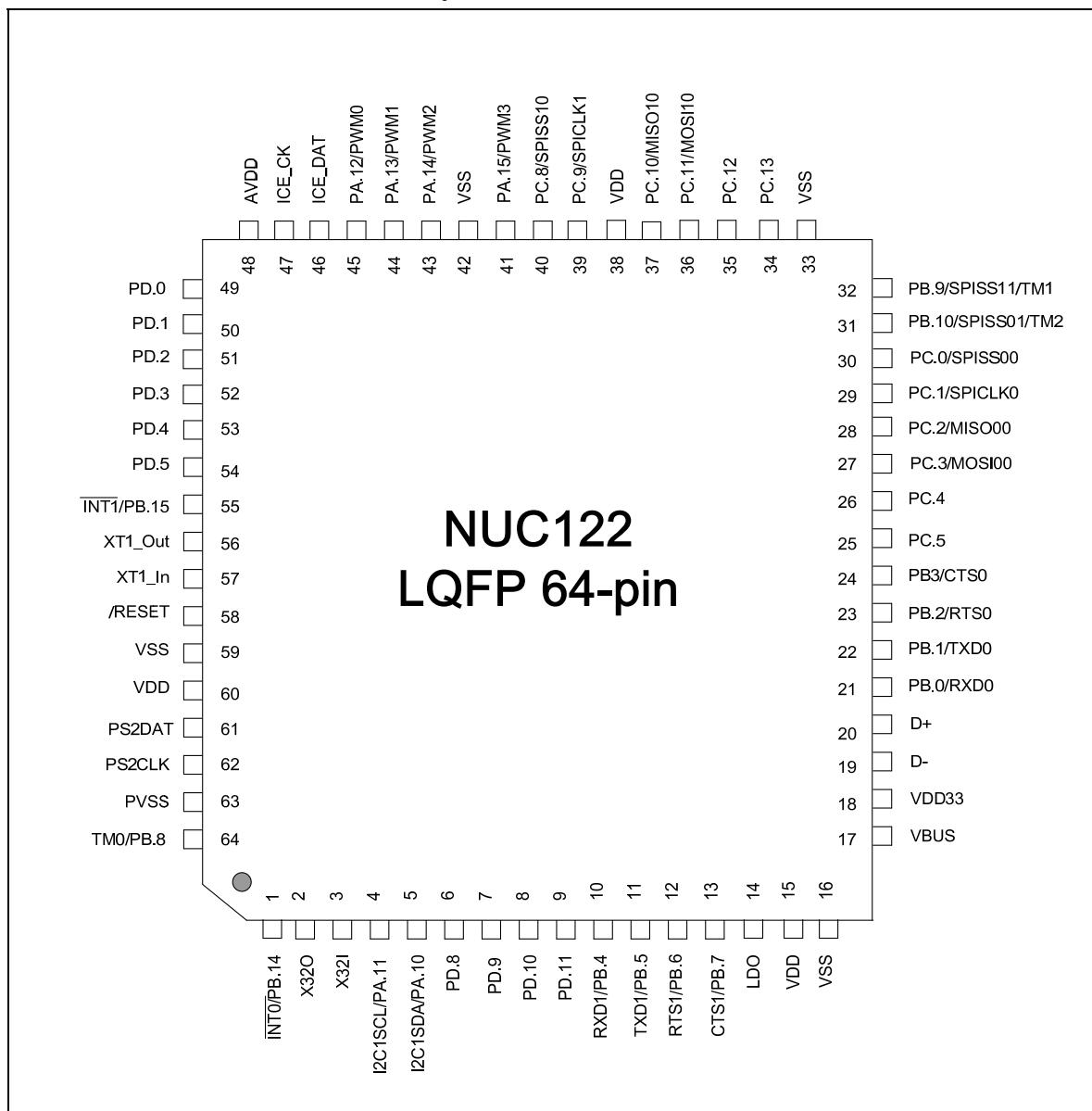


图 3-1 NuMicro™ NUC122 LQFP 64-pin 引脚图

3.2.2 NuMicro™ NUC122 LQFP 48-pin

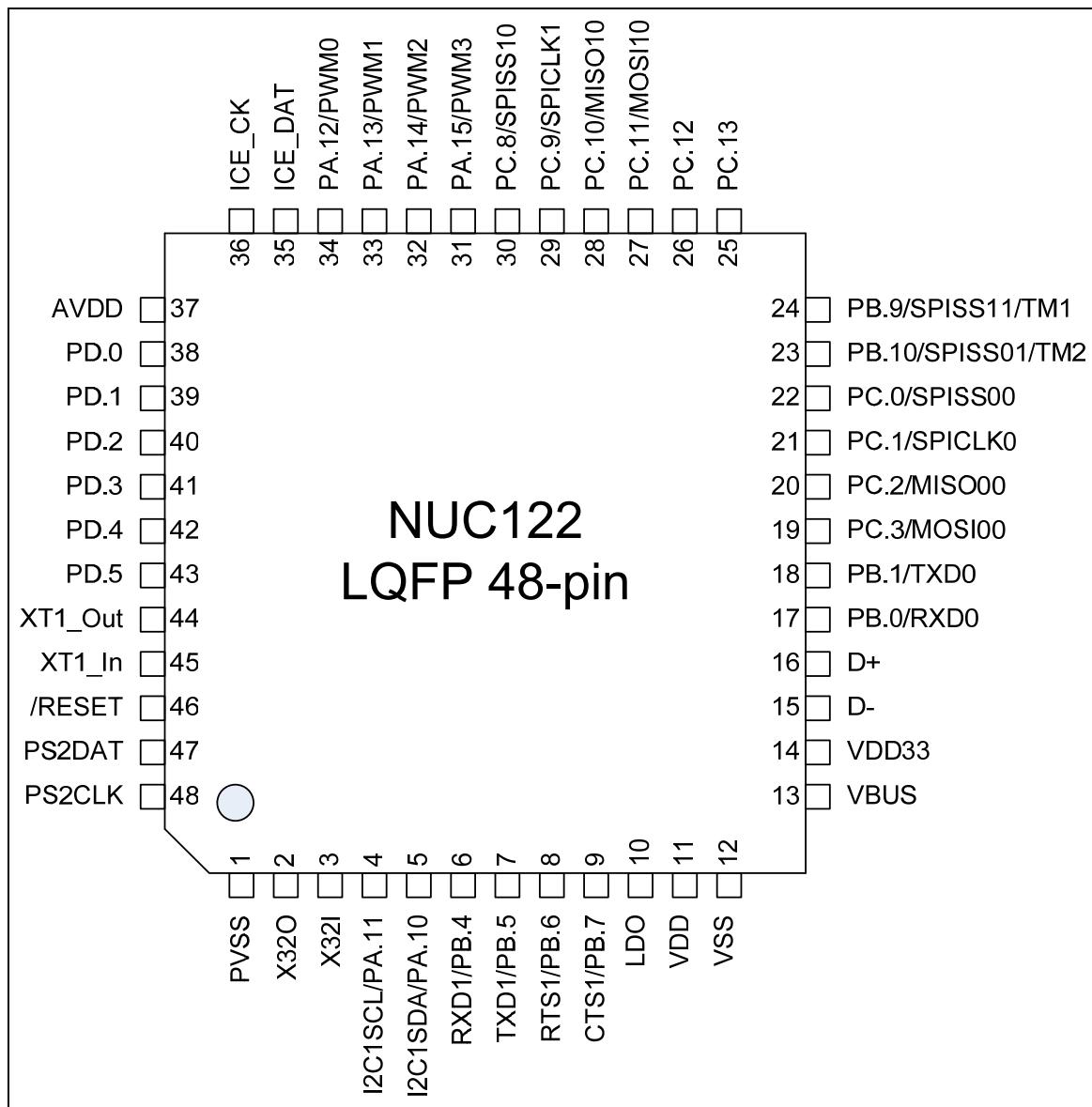


图 3-2 NuMicro™ NUC122 LQFP 48-pin 引脚图

3.2.3 NuMicro™ NUC122 QFN 33-pin

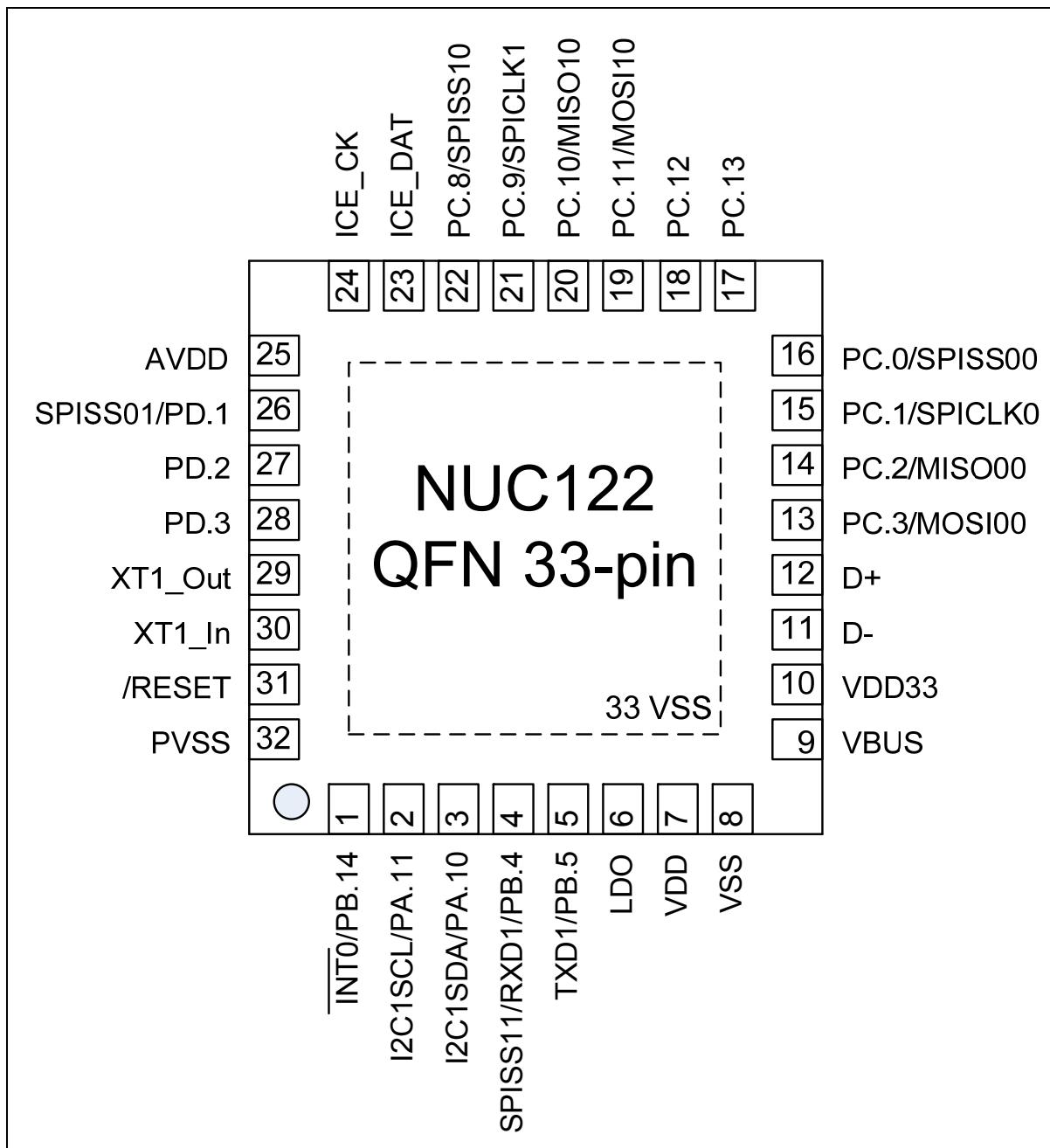


图3-3 NuMicro™ NUC122 QFN 33-pin 引脚图

3.3 引脚功能描述

3.3.1 NuMicro™ NUC122 引脚定义

管脚号.			管脚名	管脚类型		描述
LQFP 64	LQFP 48	QFN 33				
1		1	PB.14	I/O		通用IO口
			/INT0	I		/INT0: 外中断 0 输入引脚
2	2		X32O	O		32.768 KHz 低速晶体输出引脚
3	3		X32I	I		32.768 KHz 低速晶体输入引脚
4	4	2	PA.11	I/O		通用IO口
			I2C1SCL	I/O		I2C1SCL: I ² C1 时钟引脚
5	5	3	PA.10	I/O		通用IO口
			I2C1SDA	I/O		I2C1SDA: I ² C1 数据输入输出引脚
6			PD.8	I/O		通用IO口
7			PD.9	I/O		通用IO口
8			PD.10	I/O		通用IO口
9			PD.11	I/O		通用IO口
10	6	4	PB.4	I/O		通用IO口
			RXD1	I		RXD1: UART1 的接收引脚
			SPISS11	I/O		SPISS11: SPI1 从器件选择 (仅对于 QFN33)
11	7	5	PB.5	I/O		通用IO口
			TXD1	O		TXD1: UART1 的数据发送
12	8		PB.6	I/O		通用IO口
			RTS1	O		RTS1: UART1 的请求发送
13	9		PB.7	I/O		通用IO口
			CTS1	I		CTS1: UART1 的清除发送
14	10	6	LDO	P		LDO 输出, 请接 10 UF 退耦电容
15	11	7	VDD	P		IO口和LDO的电源
16	12	8	VSS	P		地

管脚号.			管脚名	管脚类型		描述
LQFP 64	LQFP 48	QFN 33				
17	13	9	VBUS	P		电源: 从USB 主机来的供给 HUB.
18	14	10	VDD33	P		内部 3.3 V 稳压输出, 请接 10 UF 退耦电容
19	15	11	D-	USB		USB 差分信号 D-
20	16	12	D+	USB		USB 差分信号 D+
21	17		PB.0	I/O		通用IO口
			RXD0	I		RXD0: UART0 的数据接收
22	18		PB.1	I/O		通用IO口
			TXD0	O		TXD0: UART0 的数据发送
23			PB.2	I/O		通用IO口
			RTS0	O		RTS0: UART0 的请求发送
24			PB.3	I/O		通用IO口
			CTS0	I		CTS0: UART0 的清除发送
25			PC.5	I/O		通用IO口
26			PC.4	I/O		通用IO口
27	19	13	PC.3	I/O		通用IO口
			MOSI00	O		MOSI00: SPI0 MOSI 主模式输出, 从模式输入
28	20	14	PC.2	I/O		通用IO口
			MISO00	I		MISO00: SPI0 MISO 主模式输入, 从模式输出
29	21	15	PC.1	I/O		通用IO口
			SPICLK0	I/O		SPICLK0: SPI0 时钟
30	22	16	PC.0	I/O		通用IO口
			SPISS00	I/O		SPISS00: SPI0 从器件片选
31	23		PB.10	I/O		通用IO口
			TM2	I/O		TM2: Timer2 反转输出/或计数输入
			SPISS01	I/O		SPISS01: SPI0 第二个从器件片选
32	24		PB.9	I/O		通用IO口

管脚号.			管脚名	管脚类型		描述
LQFP 64	LQFP 48	QFN 33				
			TM1	I/O		TM1: Timer1 反转输出/或计数输入
			SPISS11	I/O		SPISS11: SPI1 第二个从器件选择
33			VSS	P		地
34	25	17	PC.13	I/O		通用IO口
35	26	18	PC.12	I/O		通用IO口
36	27	19	PC.11	I/O		通用IO口
			MOSI10	O		MOSI10: SPI1 MOSI 主模式输出, 从模式输入
37	28	20	PC.10	I/O		通用IO口
			MISO10	I		MISO10: SPI1 MISO 主模式输入, 从模式输出
38			VDD	P		通用IO口电源
39	29	21	PC.9	I/O		通用IO口
			SPICLK1	I/O		SPICLK1: SPI1 时钟
40	30	22	PC.8	I/O		通用IO口
			SPISS10	I/O		SPISS10: SPI1 从器件片选
41	31		PA.15	I/O		通用IO口
			PWM3	O		PWM3 输出
42			VSS	P		Ground
43	32		PA.14	I/O		通用IO口
			PWM2	O		PWM2 输出
44	33		PA.13	I/O		通用IO口
			PWM1	O		PWM1 输出
45	34		PA.12	I/O		通用IO口
			PWM0	O		PWM0 输出
46	35	23	ICE_DAT	I/O		调试数据线
47	36	24	ICE_CK	I		调试时钟线
48	37	25	AVDD	AP		模拟电源
49	38		PD.0	I/O		通用IO口

管脚号.			管脚名	管脚类型		描述
LQFP 64	LQFP 48	QFN 33				
50	39	26	PD.1	I/O		通用IO口
			SPISS01	I/O		SPISS01: SPI0 第二个片选 (仅对于 QFN33)
51	40	27	PD.2	I/O		通用IO口
52	41	28	PD.3	I/O		通用IO口
53	42		PD.4	I/O		通用IO口
54	43		PD.5	I/O		通用IO口
55			PB.15	I/O		通用IO口
			/INT1	I		/INT1: 外中断 1 输入
56	44	29	XT1_OUT	O		晶体输出
57	45	30	XT1_IN	I		晶体输入
58	46	31	/RESET	I		复位脚, 内带上拉, 低有效.
59		33	VSS	P		地
60			VDD	P		通用IO 口电源
61	47		PS2DAT	I/O		PS/2 数据线
62	48		PS2CLK	I/O		PS/2 时钟线
63	1	32	PVSS	P		PLL 地
64			PB.8	I/O		通用IO口
			TM0	O		TM0: Timer0 反转输出/或计数输入

注: I—数字输入, O—数字输出, AI—模拟输入, P—电源脚, AP—模拟电源

4 NUC122 框图

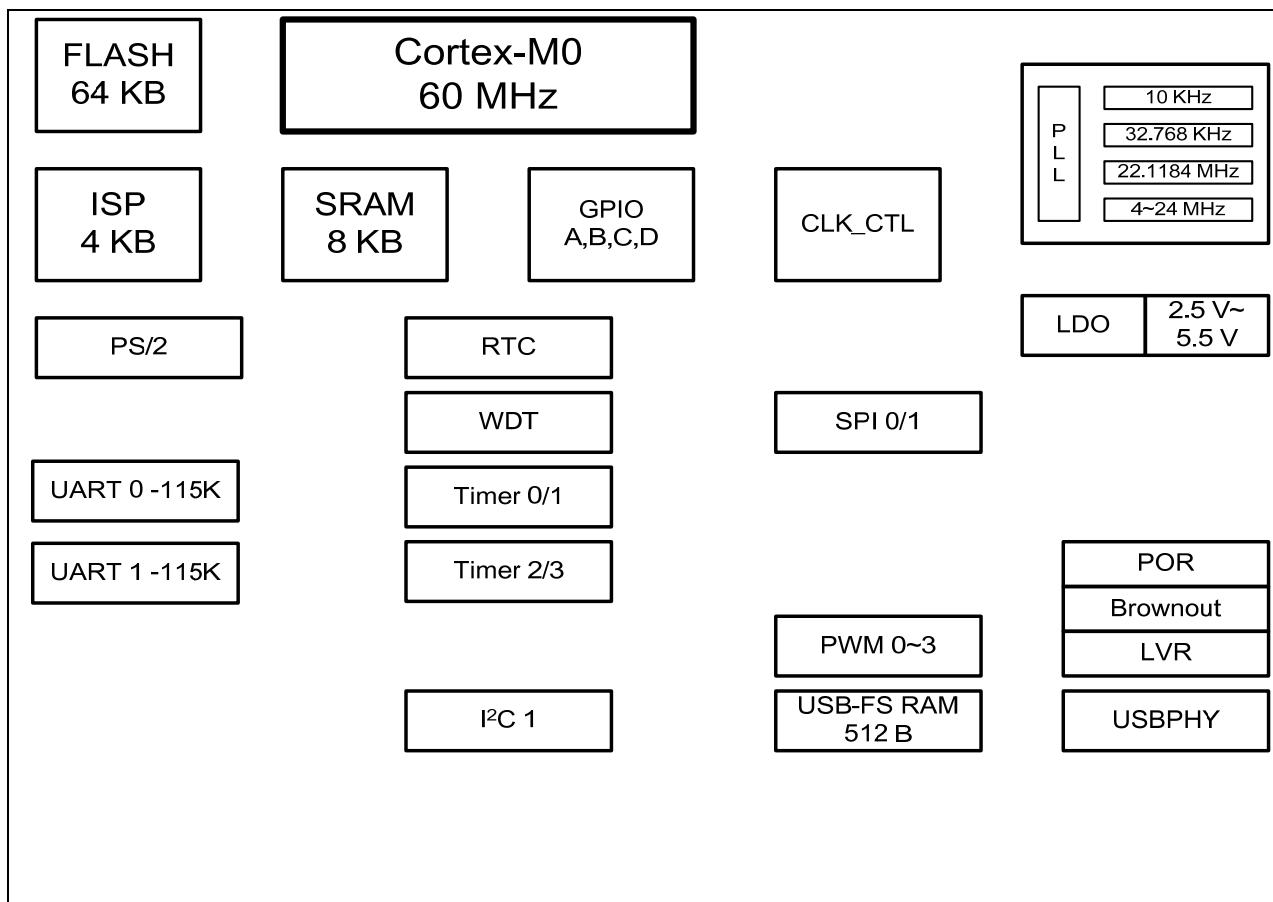


图 4-1 NuMicro™ NUC122 框图

5 功能描述

5.1 ARM® Cortex™-M0 内核

Cortex™-M0处理器是32位可配置的多级流水线RISC处理器。它有AMBA AHB-Lite接口和嵌套向量中断控制器（NVIC）。具有可选的硬件调试功能，可以执行Thumb指令，并与其它Cortex™-M系列兼容。支持两种模式-Thread 模式与 Handler 模式。异常时系统进入Handler 模式。从Handler模式返回时，执行异常返回。复位时系统进入Thread 模式。Thread 模式也可由异常返回时进入。图 5-1 为处理器的功能图。

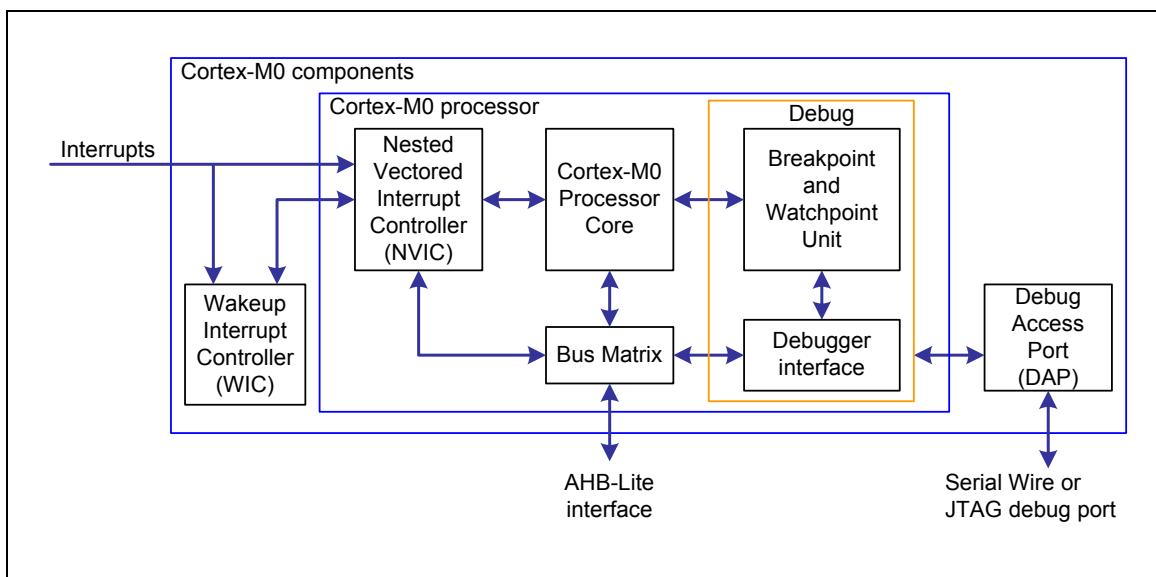


图 5-1 功能框图

设备提供：

- 低门数处理器特征：
 - ◆ ARM® v6-M Thumb® 指令集
 - ◆ Thumb-2 技术
 - ◆ ARM® v6-M 兼容 24-位 SysTick 定时器
 - ◆ 32-位 硬件乘法器
 - ◆ 系统接口支持小端（little-endian）数据访问
 - ◆ 准确而及时的中断处理能力
 - ◆ 加载、存储多个数据和多周期乘法指令可被终止然后重新开始从而实现快速中断处理
 - ◆ C 应用程序二进制接口的异常兼容模式（C-ABI）。这个 ARM® v6-M 的模式允许用户使用纯 C 函数实现中断处理
 - ◆ 使用中断唤醒（WFI）与事件唤醒（WFE）指令进入低功耗的休眠模式，或者从中断退出休眠模式

- NVIC特征:
 - ◆ 32 个外部中断，每个中断具有4级优先级
 - ◆ 专用的不可屏蔽中断（NMI）.
 - ◆ 同时支持电平和脉冲中断触发
 - ◆ 中断唤醒控制器(WIC), 支持极低功耗休眠模式.
- 调试支持
 - ◆ 四个硬件断点.
 - ◆ 两个观察点.
 - ◆ 用于非侵入式代码分析的程序计数采样寄存器（PCSR）.
 - ◆ 单步和向量捕获能力.
- 总线接口:
 - ◆ 提供简单的集成到所有系统外设和存储器的单一32位AMBA-3 ABH-Lite系统接口.
 - ◆ 支持DAP(Debug Access Port)的单一32位的从机端口.

5.2 系统管理器

5.2.1 概述

系统管理器包括如下功能:

- 系统复位
- 系统内存映射
- 产品ID、芯片复位、模块功能复位和多功能管脚控制的系统管理寄存器
- 系统定时器(SysTick)
- 嵌套向量中断控制器(NVIC)
- 系统控制寄存器

5.2.2 系统复位

以下事件可产生复位位信号，由哪个事件引起复位可从寄存器 RSTSRC 中读出。

- 上电复位
- 复位脚 (/RESET) 上有低电平
- 看门狗复位
- 低压复位
- 欠压检测器复位
- Cortex™ -M0 复位
- 系统复位

系统复位和上电复位都可以让包括片内外设在内的整个芯片复位，上电复位会复位外部晶振电路和位 ISPCON.BS，系统复位不会

5.2.3 系统电源分配

该器件的电源分为三个部分.

- 由AVDD 和AVSS提供的模拟电源，为模拟部分工作提供电压.
- 由VDD与VSS提供的数字电源，提供一个固定的1.8V的数字电源，用于数字操作和I/O引脚的内部稳压电源.
- VBUS提供给USB 的电源，用于USB模块传输操作.

内部电压调节器输出，LDO和VDD33，需要在相应的引脚上外接电容，

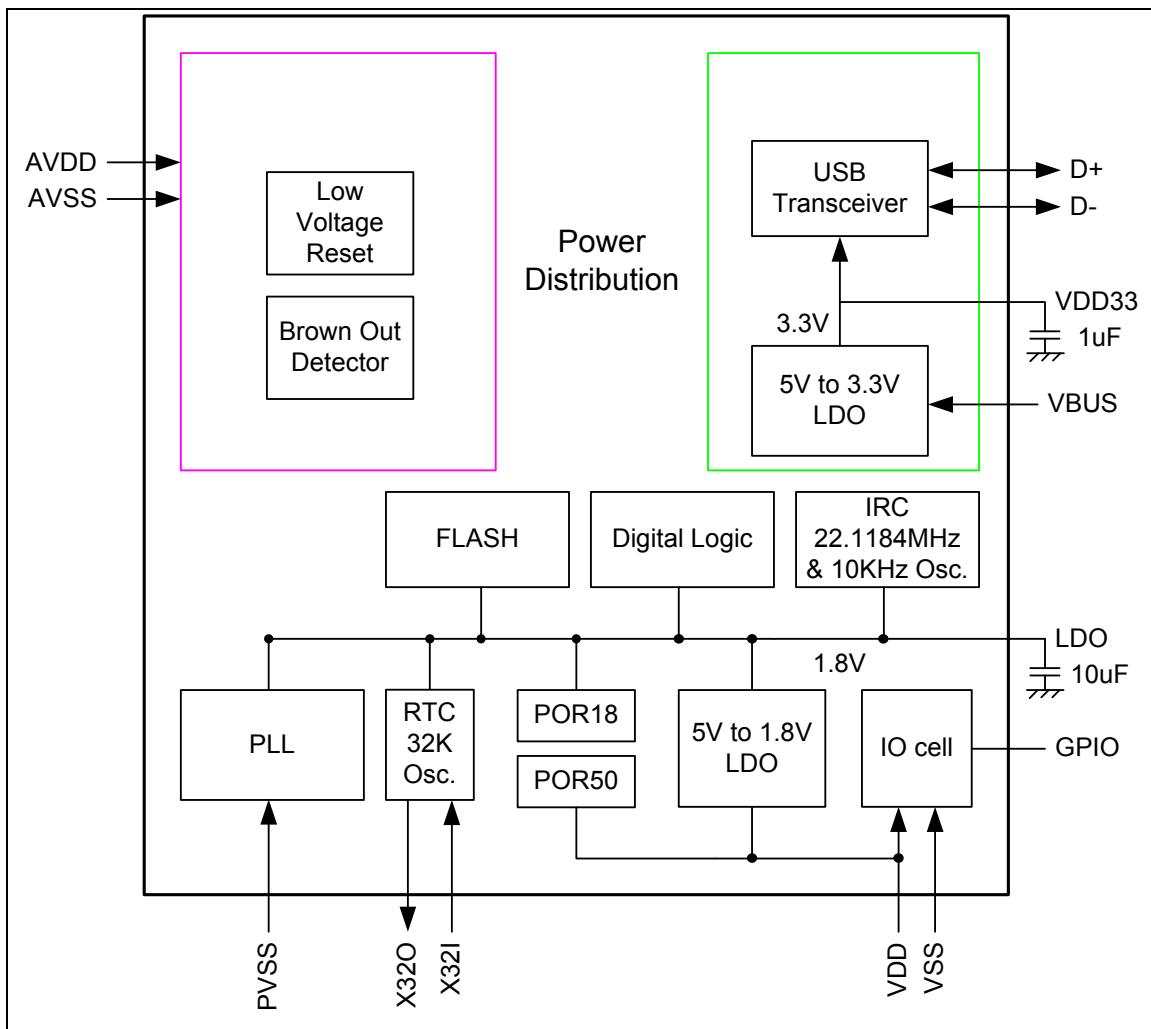


图 5-2 NuMicro™ NUC122 电源分配图

5.2.4 系统内存映射

NuMicro™ NUC122 提供 4G 字节的寻址空间。内存地址分配情况见下表。对各片上外设的详细的寄存器描述，内存空间，和编程指南，稍后章节将有详细描述。 NuMicro™ NUC122 仅支持小端数据格式。

地址空间	标志	控制
Flash & SRAM 内存空间		
0x0000_0000 – 0x0000_FFFF	FLASH_BA	FLASH 内存空间 (64KB)
0x2000_0000 – 0x2000_1FFF	SRAM_BA	SRAM 内存空间 (8KB)
AHB 控制空间 (0x5000_0000 – 0x501F_FFFF)		
0x5000_0000 – 0x5000_01FF	GCR_BA	系统全局控制寄存器
0x5000_0200 – 0x5000_02FF	CLK_BA	时钟控制寄存器
0x5000_0300 – 0x5000_03FF	INT_BA	中断多路控制寄存器
0x5000_4000 – 0x5000_7FFF	GPIO_BA	通用 IO 口控制寄存器
0x5000_C000 – 0x5000_FFFF	FMC_BA	Flash 内存控制寄存器
APB1 控制空间 (0x4000_0000 ~ 0x400F_FFFF)		
0x4000_4000 – 0x4000_7FFF	WDT_BA	看门狗控制寄存器
0x4000_8000 – 0x4000_BFFF	RTC_BA	RTC 控制寄存器
0x4001_0000 – 0x4001_3FFF	TMR01_BA	Timer0/Timer1 控制寄存器
0x4003_0000 – 0x4003_3FFF	SPI0_BA	SPI0 主从功能控制寄存器
0x4003_4000 – 0x4003_7FFF	SPI1_BA	SPI1 主从功能控制寄存器
0x4004_0000 – 0x4004_3FFF	PWMA_BA	PWM0/1/2/3 控制寄存器
0x4005_0000 – 0x4005_3FFF	UART0_BA	UART0 控制寄存器
0x4006_0000 – 0x4006_3FFF	USBD_BA	USB 2.0 FS 设备控制寄存器
APB2 控制空间 (0x4010_0000 ~ 0x401F_FFFF)		
0x4010_0000 – 0x4010_3FFF	PS2_BA	PS/2 接口控制寄存器
0x4011_0000 – 0x4011_3FFF	TMR23_BA	Timer2/Timer3 控制寄存器
0x4012_0000 – 0x4012_3FFF	I2C1_BA	I ² C1 接口控制寄存器

0x4015_0000 – 0x4015_3FFF	UART1_BA	UART1 控制寄存器
系统控制空间(0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SCS_BA	系统定时器控制寄存器
0xE000_E100 – 0xE000_ECFF	SCS_BA	外中断控制寄存器
0xE000_ED00 – 0xE000_ED8F	SCS_BA	系统控制寄存器

表 5-1 片上控制器的地址空间分配

5.2.5 系统管理器控制寄存器

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

寄存器	偏移量	R/W	描述	复位后的值
GCR_BA = 0x5000_0000				
PDID	GCR_BA+0x00	R	器件 ID 寄存器	0x0014_0018 ^[1]
RSTSRC	GCR_BA+0x04	R/W	系统复位源寄存器	0x0000_00XX
IPRSTC1	GCR_BA+0x08	R/W	外设复位控制寄存器 1	0x0000_0000
IPRSTC2	GCR_BA+0x0C	R/W	外设复位控制寄存器 2	0x0000_0000
BODCR	GCR_BA+0x18	R/W	欠压检测控制寄存器	0x0000_008X
PORCR	GCR_BA+0x24	R/W	上电复位控制寄存器	0x0000_00XX
GPA_MFP	GCR_BA+0x30	R/W	GPIOA 多功能和输入类型控制寄存器	0x0000_0000
GPB_MFP	GCR_BA+0x34	R/W	GPIOB 多功能和输入类型控制寄存器	0x0000_0000
GPC_MFP	GCR_BA+0x38	R/W	GPIOC 多功能和输入类型控制寄存器	0x0000_0000
GPD_MFP	GCR_BA+0x3C	R/W	GPIOD 多功能和输入类型控制寄存器	0x0000_0000
ALT_MFP	GCR_BA+0x50	R/W	复用多功能引脚控制寄存器	0x0000_0000
REGWRPROT	GCR_BA+0x100	R/W	寄存器写保护寄存器	0x0000_0000

Note: [1] Dependents on part number.



器件ID寄存器 (PDID)

寄存器	偏移量	R/W	描述	复位后的值
PDID	GCR_BA+0x00	R	器件 ID 寄存器	0x0014_0018 ^[1]

[1]每个器件都有一个独一无二的默认复位值.

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

Bits	描述	
[31:0]	PDID	产品器件识别码 该寄存器反映器件的器件号码。S/W可以读该寄存器来识别所使用的器件.

系统复位源寄存器 (RSTSRC)

该寄存器提供一些信息用于识别引起芯片上次复位操作的复位源.

寄存器	偏移量	R/W	描述	复位后的值
RSTSRC	GCR_BA+0x04	R/W	系统复位源寄存器	0x0000_00XX

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

Bits	描述	
[31:8]	保留	保留
[7]	RSTS_CPU	如果软件写1到CPU_RST(IPRTC1[1]), 复位Cortex-M0 CPU的核和Flash控制器, 硬件会把RSTS_CPU标志位置'1' 1= Cortex™-M0 CPU 内核与FMC因为软件置CPU_RST为1而复位. 0= CPU无复位 向该位写1清零.
[6]	保留	保留
[5]	RSTS_SYS	RSTS_SYS标志位由来自MCU Cortex™-M0 的“复位信号”置位, 以表示当前的复位源. 1 = Cortex™-M0 因为软件向SYSRESTREQ(AIRCR[2])写1, 发出复位信号而复位系统,(AIRCR[2]寄存器地址 = 0xE000ED0C). 0 = Cortex™ -M0无复位 向该位写1清零.
[4]	RSTS_BOD	RSTS_BOD标志位由欠压检测模块的“复位信号”置位, 用于表示当前复位源. 1 =欠压检验模块发出复位信号使系统复位 0 = BOD无复位

		向该位写1清零.
[3]	RSTS_LVR	RSTS_LVR标志位由低压复位模块的“复位信号”置位，用于表示当前复位源。 1= 低压 LVR 模块发出复位信号使系统复位. 0= LVR无复位 向该位写1清零.
[2]	RSTS_WDT	RSTS_WDT标志位由看门狗模块的“复位信号”置位，用于说明当前复位源. 1: 看门狗模块发出复位信号使系统复位. 0: 没有看门狗复位信号 向该位写1清零.
[1]	RSTS_RESET	RSTS_RESET 标志位由/RESET脚的“复位信号”置位，用于说明当前复位源. 1: /RESET脚上发出复位信号使系统复位. 0: 没有/RESET复位信号 向该位写1清零.
[0]	RSTS_POR	RSTS_POR 标志由POR的“复位信号”或CHIP_RST(IPRSTC1[0])位置位，用于说明当前的复位源. 1 = 上电复位 (POR) 或 CHIP_RST 发出复位信号使系统复位. 0 =没有POR或CHIP_RST复位信号 向该位写1清零.

外设复位控制寄存器 1 (IPRSTC1)

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

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

Bits	描述	
[31:2]	保留	保留
[1]	CPU_RST	<p>CPU 内核复位 (写保护位) 设置该位仅复位CPU内核和FMC，该位在2个时钟周期后自动清零 该位受保护，编程该位时，需要依次向0x5000_0100写入"59h", "16h", "88h", 参考寄存器 REGWRPROT,地址GCR_BA + 0x100. 1 = CPU 复位 0 = CPU 正常工作</p>
[0]	CHIP_RST	<p>CHIP 复位 (写保护位) 设置该位复位整个芯片，包括CPU内核和所有外设，该位在2个时钟周期后，自动清零 CHIP_RST 与上电复位一样，所有芯片控制器都复位，芯片设置从flash重新加载。 CHIP_RST 与 SYSRESETREQ的区别，参考section 5.2.2 该位受保护，编程该位时，需要依次向0x5000_0100写入"59h", "16h", "88h", 参考寄存器 REGWRPROT,地址GCR_BA + 0x100 1 = CHIP 复位 0 = CHIP 正常工作</p>



外设复位控制寄存器 2 (IPRSTC2)

置”1”，产生异步复位信号给相应的IP控制器。该位置0相应的IP控制器从复位状态恢复

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

31	30	29	28	27	26	25	24
保留				USBD_RST	保留		
23	22	21	20	19	18	17	16
PS2_RST	保留		PWM03_RST	保留		UART1_RST	UART0_RST
15	14	13	12	11	10	9	8
保留	保留	SPI1_RST	SPI0_RST	保留		I2C1_RST	保留
7	6	5	4	3	2	1	0
保留		TMR3_RST	TMR2_RST	TMR1_RST	TMR0_RST	GPIO_RST	保留

Bits	描述	
[31:28]	保留	保留
[27]	USBD_RST	USB 设备控制器复位 1 = USB设备控制器复位 0 = USB 设备控制器正常工作
[26:24]	保留	保留
[23]	PS2_RST	PS/2 控制器复位 1 = PS/2控制器复位 0 = PS/2控制器正常工作
[22:21]	保留	保留
[20]	PWM03_RST	PWM03 控制器复位 1 = PWM03 控制器复位 0 = PWM03 控制器正常工作
[19:18]	保留	保留
[17]	UART1_RST	UART1 控制器复位 1 = UART1 控制器复位 0 = UART1 控制器正常工作

[16]	UART0_RST	UART0 控制器复位 1 = UART0 控制器复位 0 = UART0 控制器正常工作
[15:14]	保留	保留
[13]	SPI1_RST	SPI1 控制器复位 1 = SPI1 控制器复位 0 = SPI1 控制器正常工作
[12]	SPI0_RST	SPI0 控制器复位 1 = SPI0 控制器复位 0 = SPI0 控制器正常工作
[11:10]	保留	保留
[9]	I2C1_RST	I²C1 控制器复位 1 = I ² C1 控制器复位 0 = I ² C1 控制器正常工作
[8:6]	保留	保留
[5]	TMR3_RST	Timer3 控制器复位 1 = Timer3 控制器复位 0 = Timer3 控制器正常工作
[4]	TMR2_RST	Timer2 控制器复位 1 = Timer2 控制器复位 0 = Timer2 控制器正常工作
[3]	TMR1_RST	Timer1 控制器复位 1 = Timer1 控制器复位 0 = Timer1 控制器正常工作
[2]	TMR0_RST	Timer0 控制器复位 1 = Timer0 控制器复位 0 = Timer0 控制器正常工作
[1]	GPIO_RST	GPIO 控制器复位 1 = GPIO 控制器复位 0 = GPIO 控制器正常工作
[0]	保留	保留

欠压检测控制寄存器 (BODCR)

BODCR 控制寄存器的部分位在flash配置时（config0寄存器）已经初始化，部分位是受保护的位。编程这些写保护的位，需要向控制寄存器0x5000_0100依次写入“59h”，“16h”“88h”，参考REGWRPROT (GCR_BA+0x100)

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

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

Bits	描述	
[31:8]	保留	保留
[7]	LVR_EN	<p>低压复位使能 (写保护位) 当输入电源电压低于LVR电路设置时，LVR复位芯片。默认使能低电压复位功能。</p> <p>1= 使能低电压复位功能，使能该位100us后，低电压复位输出稳定，LVR功能生效。(默认)。 0= 禁用低电压复位功能</p> <p>该位受保护，编程该位时，需要依次向0x5000_0100写入"59h", "16h", "88h"，参考寄存器 REGWRPROT,地址GCR_BA + 0x100</p>
[6]	BOD_OUT	<p>欠压检测输出的状态位 1 =欠压检测输出状态为 1, 表示检测到的电压低于 BOD_VL的设置. 若 BOD_EN 是“0”，禁用BOD功能，该位通常响应必定为 “0” 0 =欠压检测输出状态为0. 表示检测电压高于BOD_VL的设置或BOD_EN为0</p>
[5]	BOD_LPM	<p>低压模式下的欠压检测 (写保护位) 1 = 使能 BOD 低功耗模式 0 = BOD 工作于正常模式(默认) BOD 在正常模式下消耗电流约为100 uA, 低功耗模式下减少到当前的1/10, 但BOD响应速度变慢.</p>

		该位受保护，编程该位时，需要依次向0x5000_0100写入"59h", "16h", "88h"来禁用寄存器保护。参考寄存器 REGWRPROT,地址GCR_BA + 0x100.															
[4]	BOD_INTF	<p>欠压检测中断标志</p> <p>1= 欠压检测到VDD 下降到BOD_VL 的设定电压或VDD 升到BOD_VL 的设定电压，该位设置为1，如果欠压中断被使能，则发生欠压中断.</p> <p>0= 没有检测到任何电压由 VDD 下降或上升至BOD_VL 设定值.</p> <p>软件写1将该位清零.</p>															
[3]	BOD_RSTEN	<p>欠压复位使能 (写保护位)</p> <p>1 =使能欠压复位功能</p> <p>当同时使能欠压检测功能 (BOD_EN 为高) 和BOD 复位功能(BOD_RSTEN 为高)时，如果检测到电压低于threshold (BOD_OUT 为高)，BOD 将发送信号复位芯片</p> <p>0 = 使能欠压中断功能</p> <p>当同时使能BOD功能 (BOD_EN high) 和 BOD 中断功能(BOD_RSTEN)，如果BOD_OUT为高BOD 将产生中断. BOD 中断保持直到BOD_EN 被设置为0. 可以通过禁止NVIC BOD中断或禁止BOD功能(设置 BOD_EN low)封锁BOD中断.</p> <p>默认值由用户在配置flash控制寄存器时config0 bit[20]设置.</p> <p>该位受保护，修改该位时，需要依次向0x5000_0100写入"59h", "16h", "88h"来禁用寄存器保护，参考寄存器 REGWRPROT,地址GCR_BA + 0x100.</p>															
[2:1]	BOD_VL	<p>欠压检测Threshold 电压选择 (写保护位)</p> <p>缺省值 由用户在配置FLASH控制寄存器config0 bit[22:21]时设定</p> <p>该位受保护，修改该位时，需要依次向0x5000_0100写入"59h", "16h", "88h"来禁用寄存器保护，参考寄存器 REGWRPROT,地址GCR_BA + 0x100.</p> <table border="1"> <thead> <tr> <th>BOV_VL[1]</th> <th>BOV_VL[0]</th> <th>Brown out voltage</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1</td> <td>4.5</td> </tr> <tr> <td>1</td> <td>0</td> <td>3.8V</td> </tr> <tr> <td>0</td> <td>1</td> <td>2.7V</td> </tr> <tr> <td>0</td> <td>0</td> <td>2.2V</td> </tr> </tbody> </table>	BOV_VL[1]	BOV_VL[0]	Brown out voltage	1	1	4.5	1	0	3.8V	0	1	2.7V	0	0	2.2V
BOV_VL[1]	BOV_VL[0]	Brown out voltage															
1	1	4.5															
1	0	3.8V															
0	1	2.7V															
0	0	2.2V															
[0]	BOD_EN	<p>欠压检测使能 (写保护位)</p> <p>缺省值 由用户在配置FLASH控制寄存器config0 bit[23]时置位</p> <p>1 = 使能欠压检测功能</p> <p>0 =禁止欠压检测功能</p> <p>该位受保护，修改该位时，需要依次向0x5000_0100写入"59h", "16h", "88h"来禁用寄存器保护，参考寄存器 REGWRPROT,地址GCR_BA + 0x100.</p>															

上电复位控制寄存器 (PORCR)

寄存器	偏移量	R/W	描述	复位后的值
PORCR	GCR_BA+0x24	R/W	上电复位控制寄存器	0x0000_00XX

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

Bits	描述	
[31:16]	保留	保留
[15:0]	POR_DIS_CODE	<p>上电复位使能控制 (写保护位)</p> <p>上电时，POR电路产生复位信号使整个芯片复位，但是电源部分的干扰可能引起POR重新有效。用户可以将POR_DIS_CODE 设置为0x5AA5，禁用POR内部电路，以免造成不可预知的干扰，当设置POR_DIS_CODE 为其他值，或者由芯片的其他复位功能引起复位时，POR功能重新有效，这些复位功能包括：/RESET引脚复位，看门狗，LVR复位，BOD复位，ICE复位命令和软件复位</p> <p>该位受保护，修改该位时，需要依次向0x5000_0100写入"59h", "16h", "88h"来禁用寄存器保护，参考寄存器 REGWRPROT,地址GCR_BA + 0x100</p>

GPIOA 多功能管脚控制寄存器 (GPA_MFP)

寄存器	偏移量	R/W	描述	复位后的值
GPA_MFP	GCR_BA+0x30	R/W	GPIOA 多功能与输入类型控制寄存器	0x0000_0000

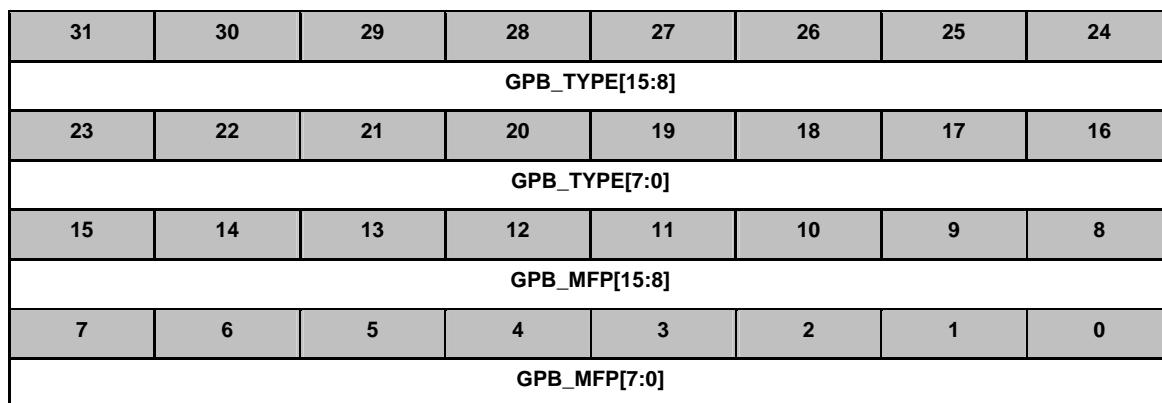
31	30	29	28	27	26	25	24
GPA_TYPE[15:8]							
23	22	21	20	19	18	17	16
GPA_TYPE[7:0]							
15	14	13	12	11	10	9	8
GPA_MFP[15:8]							
7	6	5	4	3	2	1	0
GPA_MFP[7:0]							

Bits	描述												
[31:16]	GPA_TYPEn 1 = 使能GPIOA[15:0] I/O Schmitt触发输入 0 = 禁止GPIOA[15:0] I/O Schmitt触发输入												
[15]	GPA_MFP15 PA.15 Pin功能选择 该pin功能取决于GPA_MFP15 与 ALT_MFP[9] <table border="1" style="margin-left: 20px;"> <tr><td>ALT_MFP[9]</td><td>GPA_MFP[15]</td><td>PA.15功能</td></tr> <tr><td>x</td><td>0</td><td>GPIO</td></tr> <tr><td>0</td><td>1</td><td>PWM3 (PWM)</td></tr> <tr><td>1</td><td>1</td><td>保留</td></tr> </table>	ALT_MFP[9]	GPA_MFP[15]	PA.15功能	x	0	GPIO	0	1	PWM3 (PWM)	1	1	保留
ALT_MFP[9]	GPA_MFP[15]	PA.15功能											
x	0	GPIO											
0	1	PWM3 (PWM)											
1	1	保留											
[14]	GPA_MFP14 PA.14 Pin功能选择 该pin功能取决于GPA_MFP14, 与 ALT_MFP[11] <table border="1" style="margin-left: 20px;"> <tr><td>ALT_MFP[11]</td><td>GPA_MFP[14]</td><td>PA.14功能</td></tr> <tr><td>x</td><td>0</td><td>GPIO</td></tr> <tr><td>0</td><td>1</td><td>PWM2 (PWM)</td></tr> <tr><td>1</td><td>1</td><td>保留</td></tr> </table>	ALT_MFP[11]	GPA_MFP[14]	PA.14功能	x	0	GPIO	0	1	PWM2 (PWM)	1	1	保留
ALT_MFP[11]	GPA_MFP[14]	PA.14功能											
x	0	GPIO											
0	1	PWM2 (PWM)											
1	1	保留											
[13]	GPA_MFP13 PA.13 Pin功能选择 该pin功能取决于 GPA_MFP13 , 与 ALT_MFP[11] <table border="1" style="margin-left: 20px;"> <tr><td>ALT_MFP[11]</td><td>GPA_MFP[13]</td><td>PA.13 功能</td></tr> </table>	ALT_MFP[11]	GPA_MFP[13]	PA.13 功能									
ALT_MFP[11]	GPA_MFP[13]	PA.13 功能											

		x	0	GPIO	
		0	1	PWM1 (PWM)	
		1	1	保留	
[12]	GPA_MFP12	PA.12 Pin功能选择 该pin功能取决于GPA_MFP12 , ALT_MFP[11].			
		ALT_MFP[11]	GPA_MFP[12]	PA.12 功能	
		x	0	GPIO	
		0	1	PWM0 (PWM)	
		1	1	保留	
[11]	GPA_MFP11	PA.11 Pin 功能选择 该pin功能取决于GPA_MFP11 与 ALT_MFP[11].			
		ALT_MFP[11]	GPA_MFP[11]	PA.11 功能	
		x	0	GPIO	
		0	1	SCL1 (I²C)	
		1	1	保留	
[10]	GPA_MFP10	PA.10 Pin 功能选择 该pin功能取决于 GPA_MFP10 与 ALT_MFP[11].			
		ALT_MFP[11]	GPA_MFP[10]	PA.10 功能	
		x	0	GPIO	
		0	1	SDA1 (I²C)	
		1	1	保留	
[9:0]	保留	保留			

GPIOB 多功能管脚控制寄存器 (GPB_MFP)

寄存器	偏移量	R/W	描述	复位后的值
GPB_MFP	GCR_BA+0x34	R/W	GPIOB 多功能与输入类型控制寄存器	0x0000_0000



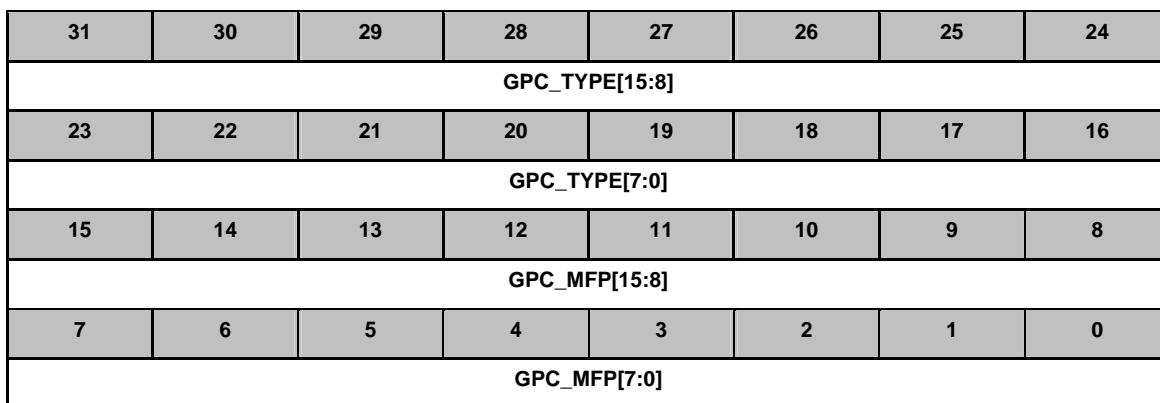
Bits	描述			
[31:16]	GPB_TYPEn			
	1 = 使能GPIOB[15:0] I/O Schmitt触发输入 0 = 禁止GPIOB[15:0] I/O Schmitt触发输入			
[15]	GPB_MFP15			
	PB.15 Pin 功能选择 1 = PB.15作为外部中断 INT1 0 = PB.15作为GPIOB[15]			
[14]	GPB_MFP14	PB.14 Pin功能选择 该引脚功能取决于 GPB_MFP14 and ALT_MFP[3]		
		ALT_MFP[3]	GPB_MFP[14]	PB.14 功能
		X	0	GPIO
		0	1	/INT0
[13: 11]	保留	保留		
		PB.10 Pin功能选择 该引脚功能取决于 GPB_MFP10 and PB10_S01 (ALT_MFP[0]).		
[10]	GPB_MFP10	ALT_MFP[0]	GPB_MFP[10]	PB.10 功能
		X	0	GPIO
		0	1	TM2

		1	1	SPISS01 (SPI0) (依具体型号而定)	
[9]	GPB_MFP9	PB.9 Pin功能选择 该引脚功能取决于 GPB_MFP9 和 PB9_S11 (ALT_MFP[1]).	ALT_MFP[1]	GPB_MFP[9]	PB.9 功能
		X	0	GPIO	
		0	1	TM1	
		1	1	SPISS11 (SPI1) 此功能依具体型号而定	
[8]	GPB_MFP8	PB.8 Pin功能选择 1 = PB.8作为 TMO (定时器/计数器外部触发时钟输入) 0 = PB.8作为 GPIOB[8]			
[7]	GPB_MFP7	PB.7 Pin功能选择 该引脚功能取决于GPB_MFP7 和 ALT_MFP[16].	ALT_MFP[16]	GPB_MFP[7]	PB.7 功能
		0	0	GPIO	
		0	1	CTS1 (UART1)	
		1	X	保留	
[6]	GPB_MFP6	PB.6 Pin功能选择 该引脚功能取决于GPB_MFP6和 ALT_MFP[17].	ALT_MFP[17]	GPB_MFP[6]	PB.6 功能
		0	0	GPIO	
		0	1	RTS1 (UART1)	
		1	X	保留	
[5]	GPB_MFP5	PB.5 Pin功能选择 1 = PB.5作为 UART1 TXD 0 = PB.5作为 GPIOB[5]			
[4]	GPB_MFP4	PB.4 Pin功能选择 1 = PB.4作为 UART1 RXD 0 = PB.4作为 GPIOB[4] QFN33封装的按下表	ALT_MFP[15]	GPB_MFP[4]	PB.4 function

		x	0	GPIO	
		0	1	RXD (UART1)	
		1	1	SPISS11 (SPI1) (the validity of this function is depended on part no)	
[3]	GPB_MFP3	PB.3 Pin功能选择 该引脚功能取决于GPB_MFP3 和 ALT_MFP[11].			
		ALT_MFP[11]	GPB_MFP[3]	PB.3 功能	
		x	0	GPIO	
		0	1	CTS0 (UART0)	
		1	1	保留	
[2]	GPB_MFP2	PB.2 Pin功能选择 该引脚功能取决于GPB_MFP2 和 ALT_MFP[11].			
		ALT_MFP[11]	GPB_MFP[2]	PB.2 功能	
		x	0	GPIO	
		0	1	RTS0 (UART0)	
		1	1	保留	
[1]	GPB_MFP1	PB.1 Pin功能选择 1 = PB.1作为 UART0 TXD 0 = PB.1作为GPIOB[1]			
[0]	GPB_MFP0	PB.0 Pin功能选择 1 = PB.0作为 UART0 RXD 0 = PB.0作为GPIOB[0]			

GPIOC 多功能管脚控制寄存器 (GPC_MFP)

寄存器	偏移量	R/W	描述	复位后的值
GPC_MFP	GCR_BA+0x38	R/W	GPIOC 多功能与输入类型控制寄存器	0x0000_0000



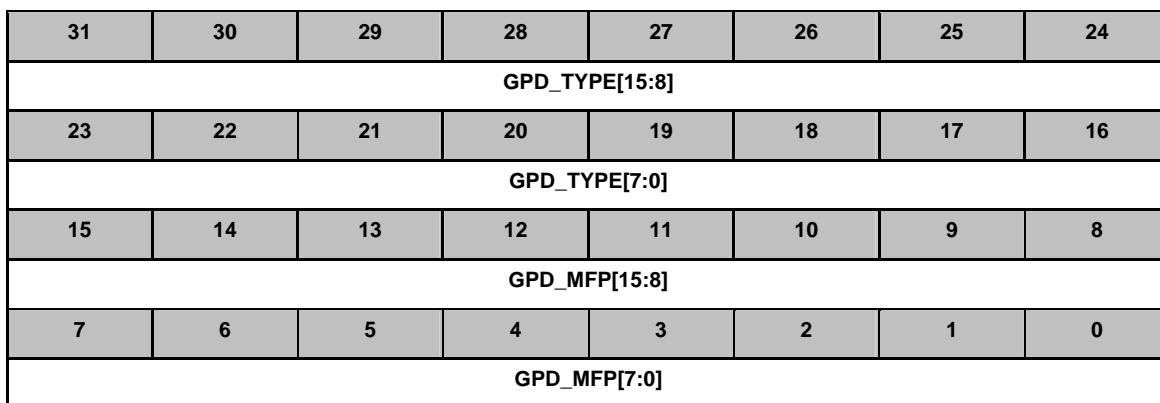
Bits	描述														
[31:16]	GPC_TYPEn	1 = 使能GPIOC[15:0] I/O Schmitt触发输入 0 = 禁止GPIOC[15:0] I/O Schmitt触发输入													
[15:14]	保留	保留													
[13]	GPC_MFP13	PC.13 Pin 功能选择 做GPIOC13用, GPC_MFP[13]和 ALT_MFP[21]必须都写入0													
[12]	GPC_MFP12	PC.12 Pin 功能选择 做GPIOC12用, GPC_MFP[12]和 ALT_MFP[13]必须都写入0													
[11]	GPC_MFP11	PC.11 功能选择 <table border="1"> <tr> <td>ALT_MFP[19]</td> <td>GPC_MFP[11]</td> <td>PC.11功能选择</td> </tr> <tr> <td>x</td> <td>0</td> <td>GPIO</td> </tr> <tr> <td>0</td> <td>1</td> <td>MOSI10 (SPI1)</td> </tr> <tr> <td>1</td> <td>1</td> <td>保留</td> </tr> </table>	ALT_MFP[19]	GPC_MFP[11]	PC.11功能选择	x	0	GPIO	0	1	MOSI10 (SPI1)	1	1	保留	
ALT_MFP[19]	GPC_MFP[11]	PC.11功能选择													
x	0	GPIO													
0	1	MOSI10 (SPI1)													
1	1	保留													
[10]	GPC_MFP10	PC.10 功能选择 <table border="1"> <tr> <td>ALT_MFP[18]</td> <td>GPC_MFP[10]</td> <td>PC.10 function</td> </tr> <tr> <td>x</td> <td>0</td> <td>GPIO</td> </tr> <tr> <td>0</td> <td>1</td> <td>MISO10 (SPI1)</td> </tr> <tr> <td>1</td> <td>1</td> <td>保留</td> </tr> </table>	ALT_MFP[18]	GPC_MFP[10]	PC.10 function	x	0	GPIO	0	1	MISO10 (SPI1)	1	1	保留	
ALT_MFP[18]	GPC_MFP[10]	PC.10 function													
x	0	GPIO													
0	1	MISO10 (SPI1)													
1	1	保留													

[9]	GPC_MFP9	PC.9 功能选择		
		ALT_MFP[17]	GPC_MFP[9]	PC.9 function
		x	0	GPIO
		0	1	SPICLK1 (SPI1)
[8]	GPC_MFP8	PC.8 Pin功能选择		
		ALT_MFP[16]	GPC_MFP[8]	PC.8 功能
		x	0	GPIO
		0	1	SPISS10 (SPI1)
[7: 6]	保留	保留		
[5]	GPC_MFP5	PC.5 Pin 功能选择 做GPIOC5用, GPC_MFP[5]要写入0		
[4]	GPC_MFP4	PC.4 Pin 功能选择 做GPIOC4用, GPC_MFP[4]要写入0		
[3]	GPC_MFP3	PC.3 Pin功能选择		
		ALT_MFP[8]	GPC_MFP[3]	PC.3 功能
		x	0	GPIO
		0	1	MOSI00 (SPI0)
[2]	GPC_MFP2	PC.2 Pin功能选择.		
		ALT_MFP[7]	GPC_MFP[2]	PC.2 功能
		x	0	GPIO
		0	1	MISO00 (SPI0)
[1]	GPC_MFP1	PC.1 Pin功能选择		
		ALT_MFP[6]	GPC_MFP[1]	PC.1 功能
		x	0	GPIO
		0	1	SPICLK0 (SPI0)
[0]	GPC_MFP0	PC.0 Pin功能选择		

ALT_MFP[5]	GPC_MFP[0]	PC.0 功能
x	0	GP O
0	1	SPISS00 (SPI0)
1	1	保留

GPIOD 多功能控制寄存器 (GPD_MFP)

寄存器	偏移量	R/W	描述	复位后的值
GPD_MFP	GCR_BA+0x3C	R/W	GPIOD 多功能与输入类型控制寄存器	0x0000_0000



Bits	描述	
[31:16]	GPD_TYPEn	1 = 使能GPIOD[15:0] I/O Schmitt触发输入 0 = 禁止GPIOD[15:0] I/O Schmitt触发输入
[15: 12]	保留	保留
[11]	GPD_MFP11	PD.11 Pin功能选择 对于PD11功能, GPD_MFP[11] 和 ALT_MFP[21]要同时写0
[10]	GPD_MFP10	PD.10 Pin功能选择 对于PD10功能, GPD_MFP[10] 和 ALT_MFP[20]要同时写0
[9]	GPD_MFP9	PD.9 Pin功能选择 对于PD9功能, GPD_MFP[9] 和 ALT_MFP[19]要同时写0
[8]	GPD_MFP8	PD.8 Pin功能选择 对于PD8功能, GPD_MFP[8] 和 ALT_MFP[18]要同时写0
[7: 6]	保留	保留
[5]	GPD_MFP5	PD.5 Pin功能选择 PD.5功能, 但要对GPD_MFP[5]写0
[4]	GPD_MFP4	PD.4 Pin功能选择 PD.4功能, 但要对GPD_MFP[4]写0
[3]	GPD_MFP3	PD.3 Pin功能选择

		PD.3功能，但要对GPD_MFP[3]写0
[2]	GPD_MFP2	PD.2 Pin功能选择 PD.2功能，但要对GPD_MFP[2]写0
[1]	GPD_MFP1	PD.1 Pin功能选择 1 = PD.1作为SPI0 的 SS01功能 (此功能依型号而定) 0 = PD.1作为GPIOD[1]
[0]	GPD_MFP0	PD.0 Pin功能选择 PD.0功能，但要对GPD_MFP[0]写0

复用多功能管脚控制寄存器 (ALT_MFP)

寄存器	偏移量	R/W	描述	复位后的值
ALT_MFP	GCR_BA+0x50	R/W	复用多功能管脚控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
ALT_MFP[15]	保留						
7	6	5	4	3	2	1	0
保留						PB9_S11	PB10_S01

Bits	描述		
[31:16]	保留	保留, 写0	
[15]	ALT_MFP[15]	The PB.4 pin function depends on GPB_MFP4 and ALT_MFP[15].	
		ALT_MFP[15]	GPB_MFP4
		x	0
		0	UART1 RX
		1	SPISS11 (SPI1,仅QFN33)
[14:2]	保留	保留, 写0	
[1]	PB9_S11	位 PB9_S11 与 GPB_MFP[9] 决定PB.9 的功能.	
		PB9_S11	GPB_MFP[]
		x	0
		0	TM1
		1	SPISS11 (SPI1)
[0]	PB10_S01	位 PB10_S01 与 GPB_MFP[10] 决定PB.10 的功能.	
		PB10_S01	GPB_MFP[10]
		x	0
		GPIO	

		0	1	TM2	
		1	1	SPISS01 (SPI0)	



寄存器写保护控制寄存器 (REGWRPROT)

有些系统控制寄存器需要被保护起来，以防止误操作而影响芯片运行，这些寄存器在上电复位到用户解锁之前是锁定的。用户可以连续依次写入“59h”，“16h”“88h”到寄存器REGWRPROT（地址：0x5000_0100）解锁。在这三个数据之间写入任何其他数据，不同时序或写入其他地址都会中止整个时序，导致无法解锁。

解锁后，用户可以检测解锁指示位：0x5000_0100 的bit0，“1”表示已经解锁，“0”表示锁定。用户可以更新目标寄存器的值，向“0x5000_0100”写入任何值，就可以重锁保护寄存器。

该位寄存器用于禁止/使能保护寄存器，读取得到REGPROTDIS状态

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

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

Bits	描述	
[31:16]	保留	保留
[7:0]	REGWRPROT	<p>寄存器写保护码 (Write only) 写保护寄存器可以通过写入“59h”，“16h”，“88h”解除锁定状态。这个时序完成之后，REGPROTDIS 位将被置1，写保护寄存器可以正常写入数据。</p>
[0]	REGPROTDIS	<p>寄存器写保护 (Read only) 1 = 解锁定以写入受保护的寄存器 0 = 锁定受保护的寄存器，不能向受保护寄存器写入数据。 受保护的寄存器有： IPRSTC1: 地址 0x5000_0008 BODCR: 地址 0x5000_0018 PORCR: 地址 0x5000_0024</p>

	<p>PWRCON: 地址0x5000_0200 (在电源唤醒中断被清时, bit[6]不受保护)</p> <p>APBCLK bit[0]: 地址0x5000_0208 (bit[0] 是看门狗时钟使能)</p> <p>CLKSEL0: 地址0x5000_0210 (选择 HCLK 与CPU STCLK 时钟源)</p> <p>CLKSEL1 bit[1:0]: 地址0x5000_0214 (用于看门狗时钟源选择)</p> <p>ISPCON: 地址0x5000_C000 (Flash ISP 控制寄存器)</p> <p>WTCR: 地址0x4000_4000</p> <p>FATCON: 地址0x5000_C018</p>
--	--



5.2.6 系统定时器 (SysTick)

Cortex™-M0 包含系统定时器SysTick，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。该计数器可用于多种用途，比如：

- RTOS中的节拍定时器。
- 对内核时钟计数的高速报警定时器.
- 做为可变时间闹钟或做为信号定时器，周期依时钟源而定。
- 简单的计数器，用于软件定时
- 控制和状态寄存器里的位COUNTFLAG 可用于查寻时间是否到定时周期

使能后，定时器从SysTick 当前值寄存器(SYST_CVR)的值向下计数到0，并在下一个时钟周期，重新加载寄存器(SYST_RVR) 的值。当计数器减到0时，标志位COUNTFLAG置位，读COUNTFLAG位使其清零。

复位后，SYST_CVR 的值未知。使能前，软件应该向寄存器写入值清0. 这样确保定时器以SYST_RVR中的值计数，而非任意值。

若SYST_RVR 是0，在重新加载后，定时器将保持当前值0。这个功能可以在计数器使能后用来禁止计数的功能。

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

5.2.6.1 系统定时器控制寄存器映射

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

寄存器	偏移量	R/W	描述	复位后的值
SCS_BA = 0xE000_E000				
SYST_CSR	SCS_BA+0x10	R/W	SysTick 控制与状态寄存器	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

5.2.6.2 系统定时器控制寄存器描述

SysTick控制与状态寄存器 (SYST_CSR)

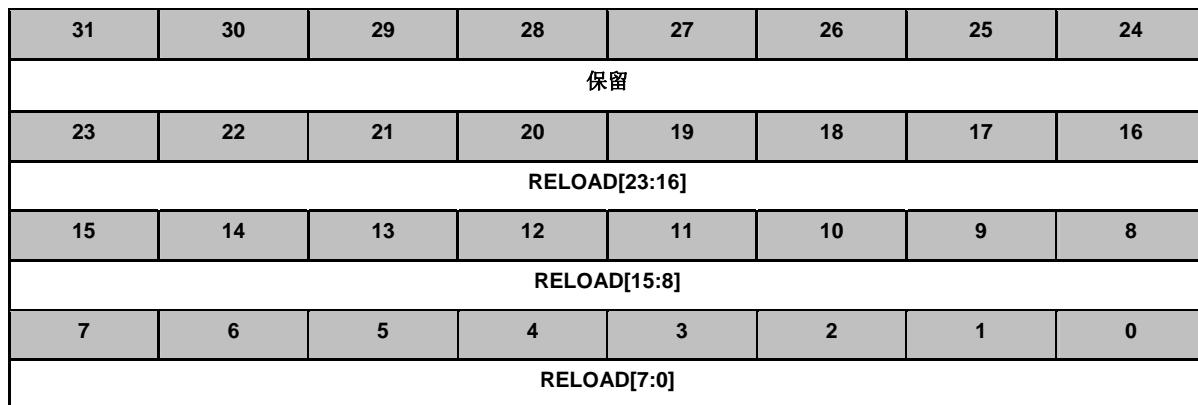
寄存器	偏移量	R/W	描述	复位后的值
SYST_CSR	SCS_BA+0x10	R/W	SysTick 控制与状态寄存器	0x0000_0000

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

Bits	描述	
[31:17]	保留	保留
[16]	COUNTFLAG	从上次该寄存器读取后，如果定时器计数到0，则返回1. 计数由1到0时，COUNTFLAG 置位。 在读或向当前值寄存器 (SYST_CVR) 写时，COUNTFLAG 被清零.
[15:3]	保留	保留
[2]	CLKSRC	1 = 内核时钟作SysTick. 0 = 时钟源为外部参考时钟
[1]	TICKINT	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 = 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.
[0]	ENABLE	1 : 计数器运行于multi-shot manner. 0 : 禁止计数器

SysTick重新加载值寄存器 (SYST_RVR)

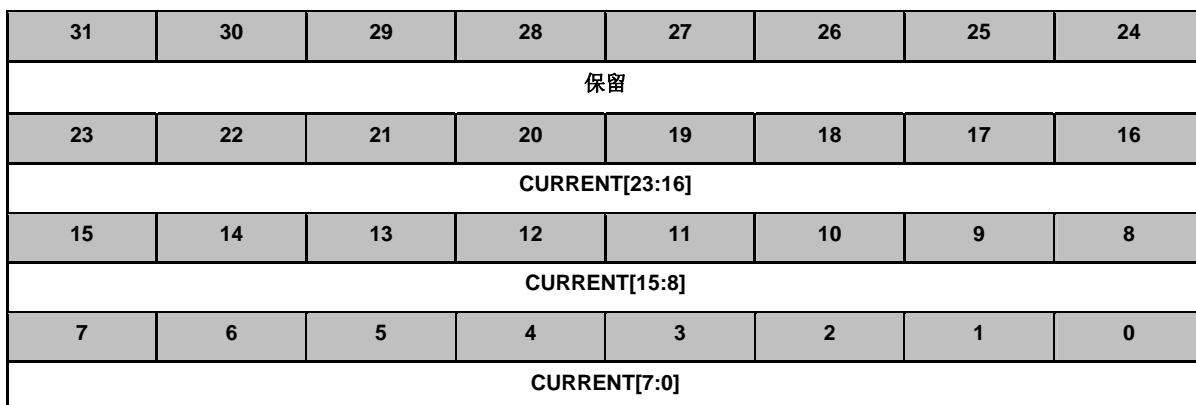
寄存器	偏移量	R/W	描述	复位后的值
SYST_RVR	SCS_BA+0x14	R/W	SysTick重新加载值寄存器	0XXXX_XXXX



Bits	描述	
[31:24]	保留	保留
[23:0]	RELOAD	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	偏移量	R/W	描述	复位后的值
SYST_CVR	SCS_BA+0x18	R/W	SysTick当前值寄存器	0XXXX_XXXX



Bits	描述	
[31:24]	保留	保留
[23:0]	CURRENT	当前计数值，为采样时刻的计数器的值，计数器不提供读修改写保护功能，该寄存器为write-clear.软件写入任何值将清寄存器为0

5.2.7 嵌套向量中断控制器 (NVIC)

Cortex™-M0提供中断控制器，用于总体管理异常，称之为“嵌套向量中断控制器(NVIC)”。 NVIC和处理器内核紧密相连，它提供以下特征：

- 支持嵌套和向量中断
- 自动保存和恢复处理器状态
- 动态改变优先级
- 简化的和确定的中断时间

NVIC依照优先级处理所有支持的异常，所有异常在“处理器模式”处理。 NVIC结构支持32[IRQ[31:0]]个离散中断，每个中断可以支持4级离散中断优先级。所有的中断和大多数系统异常可以配置为不同优先级。当中断发生时，NVIC将比较新中断与当前中断的优先级，如果新中断优先级高，则立即处理新中断。

发生中断时，ISR的开始地址可从内存的向量表中取得。不需要确定哪个中断被响应，也不要软件分配相关中断服务程序（ISR）的开始地址。当开始地址取得时，NVIC将自动保存处理状态到栈中，包括以下寄存器“PC, PSR, LR, R0~R3, R12”的值。在ISR结束时，NVIC 将从栈中恢复相关寄存器的值，进行正常操作，因此花费少量且确定的时间处理中断请求。

NVIC支持末尾链接“Tail Chaining”，有效处理背对背中断“back-to-back interrupts”，即无需保存和恢复当前状态从而减少在切换当前ISR时的延迟时间。NVIC还支持迟到“Late Arrival”，改善同时发生的ISR的效率。当较高优先级中断请求发生在当前ISR开始执行之前（保持处理器状态和获取起始地址阶段），NVIC将立即处理更高优先级的中断，从而提高了实时性。

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

5.2.7.1 异常模式和系统中断映射

NuMicro™ NUC122 支持表 5-2 所列的异常模式。与所有中断一样，软件可以对其中一些中断设置4级优先级。最高优先级为“0”，最低优先级为“3”，所有用户可配置的优先级的默认值为“0”。注意：优先级为“0”在整个系统中为第4优先级，排在“Reset”，“NMI”与“Hard Fault”之后。

异常名称	向量号	优先级
Reset	1	-3
NMI	2	-2
Hard Fault	3	-1
保留	4 ~ 10	保留
SVCALL	11	可配置
保留	12 ~ 13	保留
PendSV	14	可配置
SysTick	15	可配置

Interrupt (IRQ0 ~ IRQ31)	16 ~ 47	可配置
--------------------------	---------	-----

表 5-2 异常模式

向量号	中断号 (Bit in Interrupt Registers)	中断名称	源 IP	中断描述
0 ~ 15	-	-	-	系统异常
16	0	BOD_OUT	Brown-Out	欠压检测中断
17	1	WDT_INT	WDT	看门狗定时器中断
18	2	EINT0	GPIO	PB.14脚上的外部信号中断
19	3	EINT1	GPIO	PB.15脚上的外部信号中断
20	4	GPAB_INT	GPIO	PA[15:0]/PB[13:0] 的外部信号中断
21	5	GPCD_INT	GPIO	PC[15:0]/PD[15:0] 的外部信号中断
22	6	PWMA_INT	PWM0~3	PWM0, PWM1, PWM2 与 PWM3 中断
23	7	保留	保留	保留
24	8	TMR0_INT	TMR0	Timer 0中断
25	9	TMR1_INT	TMR1	Timer 1中断
26	10	TMR2_INT	TMR2	Timer 2中断
27	11	TMR3_INT	TMR3	Timer 3中断
28	12	UART0_INT	UART0	UART0 中断
29	13	UART1_INT	UART1	UART1中断
30	14	SPI0_INT	SPI0	SPI0中断
31	15	SPI1_INT	SPI1	SPI1中断
32	16	保留	保留	保留
33	17	保留	保留	保留
34	18	保留	保留	保留
35	19	I2C1_INT	I ² C1	I ² C1中断
36	20	保留	保留	保留
37	21	保留	保留	保留
38	22	保留	保留	保留

39	23	USB_INT	USBD	USB 2.0 FS设备中断
40	24	PS2_INT	PS/2	PS/2 中断
41	25	保留	保留	保留
42	26	保留	保留	保留
43	27	保留	保留	保留
44	28	PWRWU_INT	CLKC	掉电唤醒中断
45	29	保留	保留	保留
46	30	保留	保留	保留
47	31	RTC_INT	RTC	RTC中断

表 5-3 系统中断映射

5.2.7.2 向量表

响应中断时，处理器自动从内存的向量表中取出ISR的起始地址。对于ARM® v6-M，向量表的基址为0x00000000。向量表包括复位后堆栈的初始值以及所有异常处理器的入口地址。向量号表示处理异常的先后次序。

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

表 5-4 向量表格式

5.2.7.3 操作说明

通过写相应中断使能置位寄存器或清使能寄存器，可以使能NVIC中断或禁用NVIC中断，这些寄存器通过写1使能和写1清零，读取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清除。清使能位可以阻止新的相应中断被激活。

使能/禁用NVIC 中断，可使用寄存器 Set-Pending Register（写1禁用）与 Clear-Pending（写1使能），读这些寄存器返回相应中断的挂起状态。寄存器 Clear-Pending 在中断响应时，不影响执行状态。

可通过配置32位寄存器中的8位字段（每个寄存器支持4个中断）来设置NVIC中断的优先级。

与NVIC相关的寄存器都可以在内存的某一块寄存器区域中设置，下一节将作出描述。

5.2.7.4 NVIC 控制寄存器

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

寄存器	偏移量	R/W	描述	复位后的值
SCS_BA = 0xE000_E000				
NVIC_ISER	SCS_BA+0x100	R/W	IRQ0 ~ IRQ31 设置使能控制寄存器	0x0000_0000
NVIC_ICER	SCS_BA+0x180	R/W	IRQ0 ~ IRQ31 清使能控制寄存器	0x0000_0000
NVIC_ISPR	SCS_BA+0x200	R/W	IRQ0 ~ IRQ31 设置挂起控制寄存器	0x0000_0000
NVIC_ICPR	SCS_BA+0x280	R/W	IRQ0 ~ IRQ31 清挂起控制寄存器	0x0000_0000
NVIC_IPR0	SCS_BA+0x400	R/W	IRQ0 ~ IRQ3 优先级控制寄存器	0x0000_0000
NVIC_IPR1	SCS_BA+0x404	R/W	IRQ4 ~ IRQ7 优先级控制寄存器	0x0000_0000
NVIC_IPR2	SCS_BA+0x408	R/W	IRQ8 ~ IRQ11 优先级控制寄存器	0x0000_0000
NVIC_IPR3	SCS_BA+0x40C	R/W	IRQ12 ~ IRQ15 优先级控制寄存器	0x0000_0000
NVIC_IPR4	SCS_BA+0x410	R/W	IRQ16 ~ IRQ19 优先级控制寄存器	0x0000_0000
NVIC_IPR5	SCS_BA+0x414	R/W	IRQ20 ~ IRQ23 优先级控制寄存器	0x0000_0000
NVIC_IPR6	SCS_BA+0x418	R/W	IRQ24 ~ IRQ27 优先级控制寄存器	0x0000_0000
NVIC_IPR7	SCS_BA+0x41C	R/W	IRQ28 ~ IRQ31 优先级控制寄存器	0x0000_0000


IRQ0 ~ IRQ31设置使能控制寄存器 (NVIC_ISER)

寄存器	偏移量	R/W	描述	复位的值
NVIC_ISER	SCS_BA+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[31:24]							
23	22	21	20	19	18	17	16
SETENA [23:16]							
15	14	13	12	11	10	9	8
SETENA [15:8]							
7	6	5	4	3	2	1	0
SETENA[7:0]							

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

IRQ0 ~ IRQ31清使能控制寄存器(NVIC_ICER)

寄存器	偏移量	R/W	描述	复位后的值
NVIC_ICER	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[31:24]							
23	22	21	20	19	18	17	16
CLRENA [23:16]							
15	14	13	12	11	10	9	8
CLRENA [15:8]							
7	6	5	4	3	2	1	0
CLRENA[7:0]							

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

IRQ0 ~ IRQ31设置/挂起控制寄存器 (NVIC_ISPR)

寄存器	偏移量	R/W	描述	复位后的值
NVIC_ISPR	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[31:24]							
23	22	21	20	19	18	17	16
SETPEND [23:16]							
15	14	13	12	11	10	9	8
SETPEND [15:8]							
7	6	5	4	3	2	1	0
SETPEND [7:0]							

Bits	描述	
[31:0]	SETPEND	软件写1，由软件控制发起相应中断.每位代表从IRQ0 ~ IRQ31 的中断号(向量号： 16 ~ 47). 写0无效 读取寄存器返回当前等待处理的中断状态.

IRQ0 ~ IRQ31清挂起控制寄存器 (NVIC_ICPR)

寄存器	偏移量	R/W	描述	复位后的值
NVIC_ICPR	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 [31:24]							
23	22	21	20	19	18	17	16
CLRPEND [23:16]							
15	14	13	12	11	10	9	8
CLRPEND [15:8]							
7	6	5	4	3	2	1	0
CLRPEND [7:0]							

Bits	描述
[31:0]	CLRPEND 写1清除，由软件控制清除等待处理的中断，每位代表从IRQ0 ~ IRQ31的中断号(向量号： 16 ~ 47). 写0无效。 读取寄存器返回当前等待处理的中断状态。

IRQ0 ~ IRQ3中断优先级寄存器 (NVIC_IPR0)

寄存器	偏移量	R/W	描述	复位后的值
NVIC_IPR0	SCS_BA+0x400	R/W	IRQ0 ~ IRQ3 中断优先级控制寄存器	0x0000_0000

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

Bits	描述	
[31:30]	PRI_3	IRQ3优先级 “0”表示最高优先级& “3”表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_2	IRQ2优先级 “0”表示最高优先级& “3”表示最低优先级
[21:16]	保留	保留
[15:14]	PRI_1	IRQ1优先级 “0”表示最高优先级& “3”表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_0	IRQ0优先级 “0”表示最高优先级& “3”表示最低优先级
[5:0]	保留	保留


IRQ4 ~ IRQ7中断优先级寄存器 (NVIC_IPR1)

寄存器	偏移量	R/W	描述	复位后的值
NVIC_IPR1	SCS_BA+0x404	R/W	IRQ4 ~ IRQ7 中断优先级寄存器	0x0000_0000

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

Bits	描述	
[31:30]	PRI_7	IRQ7优先级 “0”表示最高优先级&“3”表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_6	IRQ6优先级 “0”表示最高优先级&“3”表示最低优先级
[21:16]	保留	保留
[15:14]	PRI_5	IRQ5优先级 “0”表示最高优先级&“3”表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_4	IRQ4优先级 “0”表示最高优先级&“3”表示最低优先级
[5:0]	保留	保留

IRQ8 ~ IRQ11 中断优先级寄存器 (NVIC_IPR2)

寄存器	偏移量	R/W	描述					复位后的值
NVIC_IPR2	SCS_BA+0x408	R/W	IRQ8 ~ IRQ11 中断优先级寄存器					0x0000_0000

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

Bits	描述	
[31:30]	PRI_11	IRQ11优先级 “0”表示最高优先级& “3”表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_10	IRQ10优先级 “0”表示最高优先级& “3”表示最低优先级
[21:16]	保留	保留
[15:14]	PRI_9	IRQ9优先级 “0”表示最高优先级& “3”表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_8	IRQ8优先级 “0”表示最高优先级& “3”表示最低优先级
[5:0]	保留	保留

IRQ12 ~ IRQ15中断优先级寄存器 (NVIC_IPR3)

寄存器	偏移量	R/W	描述					复位后的值
NVIC_IPR3	SCS_BA+0x40C	R/W	IRQ12 ~ IRQ15 中断优先级寄存器					0x0000_0000

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

Bits	描述	
[31:30]	PRI_15	IRQ15优先级 “0”表示最高优先级& “3”表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_14	IRQ14优先级 “0”表示最高优先级& “3”表示最低优先级
[21:16]	保留	保留
[15:14]	PRI_13	IRQ13优先级 “0”表示最高优先级& “3”表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_12	IRQ12优先级 “0”表示最高优先级& “3”表示最低优先级
[5:0]	保留	保留


IRQ16 ~ IRQ19中断优先级寄存器 (NVIC_IPR4)

寄存器	偏移量	R/W	描述					复位后的值
NVIC_IPR4	SCS_BA+0x410	R/W	IRQ16 ~ IRQ19 中断优先级寄存器					0x0000_0000

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

Bits	描述	
[31:30]	PRI_19	IRQ19优先级 “0”表示最高优先级& “3”表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_18	IRQ18优先级 “0”表示最高优先级& “3”表示最低优先级
[21:16]	保留	保留
[15:14]	PRI_17	IRQ17优先级 “0”表示最高优先级& “3”表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_16	IRQ16优先级 “0”表示最高优先级& “3”表示最低优先级
[5:0]	保留	保留


IRQ20 ~ IRQ23中断优先级寄存器 (NVIC_IPR5)

寄存器	偏移量	R/W	描述					复位后的值
NVIC_IPR5	SCS_BA+0x414	R/W	IRQ20 ~ IRQ23 中断优先级寄存器					0x0000_0000

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

Bits	描述	
[31:30]	PRI_23	IRQ23优先级 “0”表示最高优先级& “3”表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_22	IRQ22优先级 “0”表示最高优先级& “3”表示最低优先级
[21:16]	保留	保留
[15:14]	PRI_21	IRQ21优先级 “0”表示最高优先级& “3”表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_20	IRQ20优先级 “0”表示最高优先级& “3”表示最低优先级
[5:0]	保留	保留


IRQ24 ~ IRQ27中断优先级寄存器 (NVIC_IPR6)

寄存器	偏移量	R/W	描述					复位后的值
NVIC_IPR6	SCS_BA+0x418	R/W	IRQ24 ~ IRQ27 中断优先级寄存器					0x0000_0000

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

Bits	描述	
[31:30]	PRI_27	IRQ27优先级 “0”表示最高优先级&“3”表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_26	IRQ26优先级 “0”表示最高优先级&“3”表示最低优先级
[21:16]	保留	保留
[15:14]	PRI_25	IRQ25优先级 “0”表示最高优先级&“3”表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_24	IRQ24优先级 “0”表示最高优先级&“3”表示最低优先级
[5:0]	保留	保留

IRQ28 ~ IRQ31中断优先级寄存器 (NVIC_IPR7)

寄存器	偏移量	R/W	描述	复位后的值
NVIC_IPR7	SCS_BA+0x41C	R/W	IRQ28 ~ IRQ31 中断优先级寄存器	0x0000_0000

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

Bits	描述	
[31:30]	PRI_31	IRQ31优先级 “0”表示最高优先级& “3”表示最低优先级
[29:24]	保留	保留
[23:22]	PRI_30	IRQ30优先级 “0”表示最高优先级& “3”表示最低优先级
[21:16]	保留	保留
[15:14]	PRI_29	IRQ29优先级 “0”表示最高优先级& “3”表示最低优先级
[13:8]	保留	保留
[7:6]	PRI_28	IRQ28优先级 “0”表示最高优先级& “3”表示最低优先级
[5:0]	保留	保留

5.2.7.5 中断源控制寄存器

除了NVIC相关的中断控制寄存器外，NuMicro™ NUC122 还有一些特殊寄存器便于中断处理，包括“中断源识别”，“NMI 源选择”与“中断测试模式”。描述如下。

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

寄存器	偏移量	R/W	描述	复位后的值
INT_BA = 0x5000_0300				
IRQ0_SRC	INT_BA+0x00	R	IRQ0 (BOD) 中断源识别	0XXXXX_XXXX
IRQ1_SRC	INT_BA+0x04	R	IRQ1 (WDT) 中断源识别	0XXXXX_XXXX
IRQ2_SRC	INT_BA+0x08	R	IRQ2 (EINT0) 中断源识别	0XXXXX_XXXX
IRQ3_SRC	INT_BA+0x0C	R	IRQ3 (EINT1) 中断源识别	0XXXXX_XXXX
IRQ4_SRC	INT_BA+0x10	R	IRQ4 (GPA/B) 中断源识别	0XXXXX_XXXX
IRQ5_SRC	INT_BA+0x14	R	IRQ5 (GPC/D) 中断源识别	0XXXXX_XXXX
IRQ6_SRC	INT_BA+0x18	R	IRQ6 (PWMA) 中断源识别	0XXXXX_XXXX
IRQ7_SRC	INT_BA+0x1C	R	IRQ7 (保留) 中断源识别	0XXXXX_XXXX
IRQ8_SRC	INT_BA+0x20	R	IRQ8 (TMR0) 中断源识别	0XXXXX_XXXX
IRQ9_SRC	INT_BA+0x24	R	IRQ9 (TMR1) 中断源识别	0XXXXX_XXXX
IRQ10_SRC	INT_BA+0x28	R	IRQ10 (TMR2) 中断源识别	0XXXXX_XXXX
IRQ11_SRC	INT_BA+0x2C	R	IRQ11 (TMR3) 中断源识别	0XXXXX_XXXX
IRQ12_SRC	INT_BA+0x30	R	IRQ12 (URTO) 中断源识别	0XXXXX_XXXX
IRQ13_SRC	INT_BA+0x34	R	IRQ13 (URT1) 中断源识别	0XXXXX_XXXX
IRQ14_SRC	INT_BA+0x38	R	IRQ14 (SPI0) 中断源识别	0XXXXX_XXXX
IRQ15_SRC	INT_BA+0x3C	R	IRQ15 (SPI1) 中断源识别	0XXXXX_XXXX
IRQ16_SRC	INT_BA+0x40	R	IRQ16 (保留) 中断源识别	0XXXXX_XXXX
IRQ17_SRC	INT_BA+0x44	R	IRQ17 (保留) 中断源识别	0XXXXX_XXXX
IRQ18_SRC	INT_BA+0x48	R	IRQ18 (保留) 中断源识别	0XXXXX_XXXX
IRQ19_SRC	INT_BA+0x4C	R	IRQ19 (I ² C1) 中断源识别	0XXXXX_XXXX
IRQ20_SRC	INT_BA+0x50	R	IRQ20 (保留) 中断源识别	0XXXXX_XXXX
IRQ21_SRC	INT_BA+0x54	R	IRQ21 (保留) 中断源识别	0XXXXX_XXXX
IRQ22_SRC	INT_BA+0x58	R	IRQ22 (保留) 中断源识别	0XXXXX_XXXX
IRQ23_SRC	INT_BA+0x5C	R	IRQ23 (USBD) 中断源识别	0XXXXX_XXXX

IRQ24_SRC	INT_BA+0x60	R	IRQ24 (PS/2) 中断源识别	0XXXXX_XXXX
IRQ25_SRC	INT_BA+0x64	R	IRQ25 (保留)中断源识别	0XXXXX_XXXX
IRQ26_SRC	INT_BA+0x68	R	IRQ26 (保留) 中断源识别	0XXXXX_XXXX
IRQ27_SRC	INT_BA+0x6C	R	IRQ27 (保留)中断源识别	0XXXXX_XXXX
IRQ28_SRC	INT_BA+0x70	R	IRQ28 (PWRWU) 中断源识别	0XXXXX_XXXX
IRQ29_SRC	INT_BA+0x74	R	IRQ29 (保留) 中断源识别	0XXXXX_XXXX
IRQ30_SRC	INT_BA+0x78	R	IRQ30 (保留) 中断源识别	0XXXXX_XXXX
IRQ31_SRC	INT_BA+0x7C	R	IRQ31 (RTC) 中断源识别	0XXXXX_XXXX
NMI_SEL	INT_BA+0x80	R/W	NMI中断源选择控制寄存器	0000_0000
MCU_IRQ	INT_BA+0x84	R/W	MCU IRQ号识别寄存器	0000_0000

中断源识别 寄存器 (IRQn_SRC)

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

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

Bits	Address	INT-Num	描述
[2:0]	INT_BA+0x00	0	Bit2: 0 Bit1: 0 Bit0: BOD_INT
[2:0]	INT_BA+0x04	1	Bit2: 0 Bit1: 0 Bit0: WDT_INT
[2:0]	INT_BA+0x08	2	Bit2: 0 Bit1: 0 Bit0: EINT0 –PB.14上的外部中断0
[2:0]	INT_BA+0x0C	3	Bit2: 0 Bit1: 0 Bit0: EINT1 –PB.15上的外部中断1
[2:0]	INT_BA+0x10	4	Bit2: 0 Bit1: GPB_INT Bit0: GPA_INT
[2:0]	INT_BA+0x14	5	Bit2: 0 Bit1: GPD_INT

			Bit0: GPC_INT
[3:0]	INT_BA+0x18	6	Bit3: PWM3_INT Bit2: PWM2_INT Bit1: PWM1_INT Bit0: PWM0_INT
[3:0]	INT_BA+0x1C	7	Bit3: 0 Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x20	8	Bit2: 0 Bit1: 0 Bit0: TMR0_INT
[2:0]	INT_BA+0x24	9	Bit2: 0 Bit1: 0 Bit0: TMR1_INT
[2:0]	INT_BA+0x28	10	Bit2: 0 Bit1: 0 Bit0: TMR2_INT
[2:0]	INT_BA+0x2C	11	Bit2: 0 Bit1: 0 Bit0: TMR3_INT
[2:0]	INT_BA+0x30	12	Bit2: 0 Bit1: 0 Bit0: URT0_INT
[2:0]	INT_BA+0x34	13	Bit2: 0 Bit1: 0 Bit0: URT1_INT
[2:0]	INT_BA+0x38	14	Bit2: 0 Bit1: 0 Bit0: SPI0_INT
[2:0]	INT_BA+0x3C	15	Bit2: 0 Bit1: 0 Bit0: SPI1_INT
[2:0]	INT_BA+0x40	16	Bit2: 0 Bit1: 0

			Bit0: 0
[2:0]	INT_BA+0x44	17	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x48	18	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x4C	19	Bit2: 0 Bit1: 0 Bit0: I2C1_INT
[2:0]	INT_BA+0x50	20	保留
[2:0]	INT_BA+0x54	21	保留
[2:0]	INT_BA+0x58	22	保留
[2:0]	INT_BA+0x5C	23	Bit2: 0 Bit1: 0 Bit0: USBD_INT
[2:0]	INT_BA+0x60	24	Bit2: 0 Bit1: 0 Bit0: PS2_INT
[2:0]	INT_BA+0x64	25	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x68	26	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x6C	27	Bit2: 0 Bit1: 0 Bit0: 0
[2:0]	INT_BA+0x70	28	Bit2: 0 Bit1: 0 Bit0: PWRWU_INT
[2:0]	INT_BA+0x74	29	Bit2: 0 Bit1: 0 Bit0: 0

[2:0]	INT_BA+0x78	30	保留
[2:0]	INT_BA+0x7C	31	Bit2: 0 Bit1: 0 Bit0: RTC_INT

NMI中断源选择控制寄存器(NMI_SEL)

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

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

Bits	描述	
[31:8]	保留	保留
[7]	INT_TEST	中断测试模式 (写保护位)
[6:5]	保留	保留
[4:0]	NMI_SEL	NMI 中断源选择 通过设置NMI_SEL可以在外围设备中断中选择Cortex™-M0 的 NMI 中断.

MCU中断请求源寄存器(MCU_IRQ)

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

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

Bits	描述	
[31:0]	MCU_IRQ	<p>MCU中断请求寄存器</p> <p>MCU_IRQ 从外围设备收集所有中断，并向Cortex™-M0内核产生同步中断，生成中断的模式有正常模式与测试模式。</p> <p>MCU_IRQ 从每个外围设备收集所有中断和同步这些中断，然后触发Cortex™-M0中断。</p> <p>MCU_IRQ[n] 是“0”时：置 MCU_IRQ[n] 为“1”，Cortex™-M0 NVIC[n]发生一个中断。</p> <p>MCU_IRQ[n] 是“1”时：(意味着有中断请求) 置MCU_IRQ[n]为1将清除中断；置MCU_IRQ[n]为0无效</p>

5.2.8 系统控制寄存器

系统控制寄存器控制了Cortex™-M0的状态和操作模式，包括CPUID，Cortex™-M0中断优先级和Cortex™-M0电源管理

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

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

寄存器	偏移量	R/W	描述	复位后的值
SCS_BA = 0xE000_E000				

CPUID	SCS_BA+0xD00	R	CPUID寄存器	0x410C_C200
ICSR	SCS_BA+0xD04	R/W	中断控制和状态寄存器	0x0000_0000
AIRCR	SCS_BA+0xD0C	R/W	中断与复位控制寄存器	0xFA05_0000
SCR	SCS_BA+0xD10	R/W	系统控制寄存器	0x0000_0000
SHPR2	SCS_BA+0xD1C	R/W	系统处理器优先级寄存器2	0x0000_0000
SHPR3	SCS_BA+0xD20	R/W	系统处理器优先级寄存器3	0x0000_0000

CPUID寄存器(CPUID)

寄存器	偏移量	R/W	描述	复位后的值
CPUID	SCS_BA+0xD00	R	CPUID寄存器	0x410C_C200

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

Bits	描述	
[31:24]	IMPLEMENTER	由ARM分配执行码. (ARM = 0x41)
[23:20]	保留	保留
[19:16]	PART	ARM® v6-M 读取值为0xC
[15:4]	PARTNO	读回值为 0xC20.
[3:0]	REVISION	读回值为 0x0

中断控制和状态寄存器(ICSR)

寄存器	偏移量	R/W	描述				复位后的值
ICSR	SCS_BA+0xD04	R/W	中断控制和状态寄存器				0x0000_0000

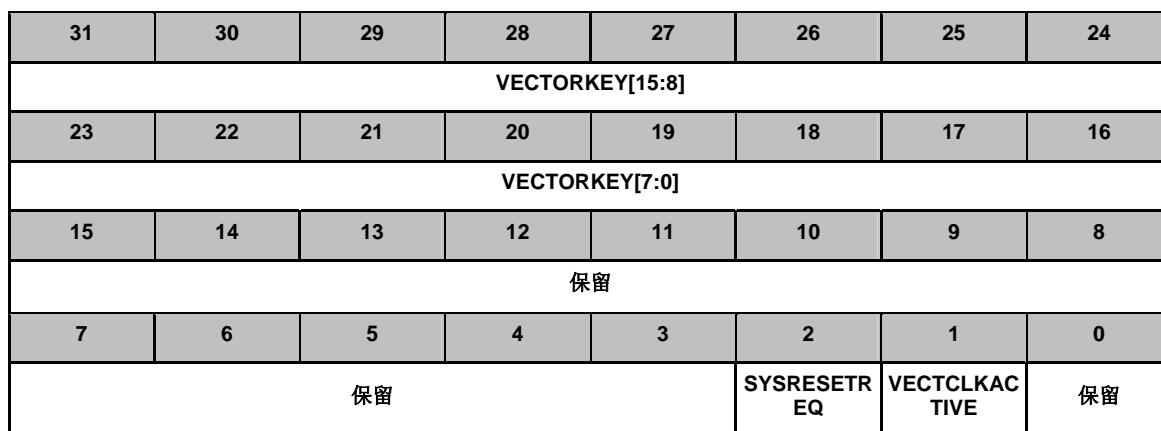
31	30	29	28	27	26	25	24
NMIPENDSET	保留		PENDSVSET	PENDSVCLR	PENDSTSET	PENDSTCLR	保留
23	22	21	20	19	18	17	16
ISRPREEMPT	ISRPENDING	保留				VECTPENDING[5:4]	
15	14	13	12	11	10	9	8
VECTPENDING[3:0]				保留			
7	6	5	4	3	2	1	0
保留		VECTACTIVE[5:0]					

Bits	描述	
[31]	NMIPENDSET	NMI 设置挂起位 写： 0 = 该位写0无效 1 = 更改 NMI 异常状态为挂起 读： 0 = NMI 异常没有挂起 1 = NMI 异常挂起 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]	保留	保留
[28]	PENDSVSET	PendSV 设置挂起位 写： 0 = 该位写0无效 1 = 更改 PendSV 异常状态为挂起 读： 0 = PendSV 异常没有挂起 1 = PendSV 异常挂起 向该位写1是将 PendSV 异常状态设为挂起的唯一方法
[27]	PENDSVCLR	PendSV 清除挂起位

		写: 0 = 该位写0无效 1 = 移除 PendSV 异常的挂起状态 只写位
[26]	PENDSTSET	SysTick 异常设置挂起位 写: 0 = 该位写0无效 1 = 更改SysTick 异常状态为挂起 读: 0 = SysTick 异常没有挂起 1 = SysTick 异常挂起 设置挂起SysTick. 可将当前状态读回 (挂起为1, 否则是0).
[25]	PENDSTCLR	SysTick 异常清除挂起位 写: 0 = 该位写0无效 1 = 移除 SysTick 异常的挂起状态 只写位, 读该寄存器的值是未知的
[24]	保留	保留
[23]	ISRPREEMPT	如果置位, 挂起异常生效, 由调试停止状态退出. 只读位
[22]	ISRPENDING	中断挂起标志, 不包括 NMI 和 Faults 0 = 中断没有挂起 1 = 中断挂起 只读位
[21:18]	保留	保留
[17:12]	VECTPENDING	表示最高优先级挂起异常的异常号 0 = 没有异常挂起。 非0 = 最高优先级挂起异常的异常号 只读位
[11:6]	保留	保留
[5:0]	VECTACTIVE	当前执行异常处理的异常号 0 = 线程模式 非0 = 当前执行异常处理的异常号. 只读位

中断和复位控制寄存器(AIRCR)

寄存器	偏移量	R/W	描述	复位后的值
AIRCR	SCS_BA+0xD0C	R/W	中断和复位控制寄存器	0xFA05_0000



Bits	描述	
[31:16]	VECTORKEY	写该寄存器时，该值为0x05FA, 否则写动作将产生不可预测的结果.
[15:3]	保留	保留
[2]	SYSRESETREQ	该位写1, 产生复位信号给芯片表示有复位请求. 该位只写, 在复位时自动清零.
[1]	VECTCLRACTIVE	该位置1, 清除所有固定的和可配置异常的活动状态. 该位只写, 只有在内核挂起时可写. 注: 调试器负责重新初始化堆栈.
[0]	保留	保留



系统控制寄存器(SCR)

寄存器	偏移量	R/W	描述	复位后的值
SCR	SCS_BA+0xD10	R/W	系统控制寄存器	0x0000_0000

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

Bits	描述	
[31:5]	保留	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]	保留	保留
[2]	SLEEPDEEP	控制处理器使用休眠或者深度休眠作为它的低功耗模式 0 = 休眠 1 = 深度休眠
[1]	SLEEPONEXIT	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 模式时不休眠 1 = 当从某个ISR 切换到 Thread 模式时，进入休眠或者深度休眠 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]	保留	保留

系统处理器优先级寄存器2 (SHPR2)

寄存器	偏移量	R/W	描述	复位后的值
SHPR2	SCS_BA+0xD1C	R/W	系统处理器优先级寄存器 2	0x0000_0000

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

Bits	描述	
[31:30]	PRI_11	系统处理器的优先级11 – SVCall “0” 表示最高优先级 & “3” 表示最低优先级
[29:0]	保留	保留

系统处理器优先级寄存器3 (SHPR3)

寄存器	偏移量	R/W	描述	复位后的值
SHPR3	SCS_BA+0xD20	R/W	系统处理器优先级寄存器 3	0x0000_0000

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

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

5.3 时钟控制器

5.3.1 概述

时钟控制器为整个芯片提供时钟源，包括系统时钟和所有外围设备时钟，该控制器还通过个别时钟的关或开，时钟源选择和分频器来进行功耗控制。CPU使能PWR_DOWN_EN位后，CPU Cortex™-M0 内核执行WFI指令，芯片将进入掉电模式。等唤醒中断发生将退出掉电模式。在掉电模式下，时钟控制器关闭外部 4~24 MHz 高速晶振和内部 22.1184 MHz 高速振荡器，以降低整个系统的功耗。

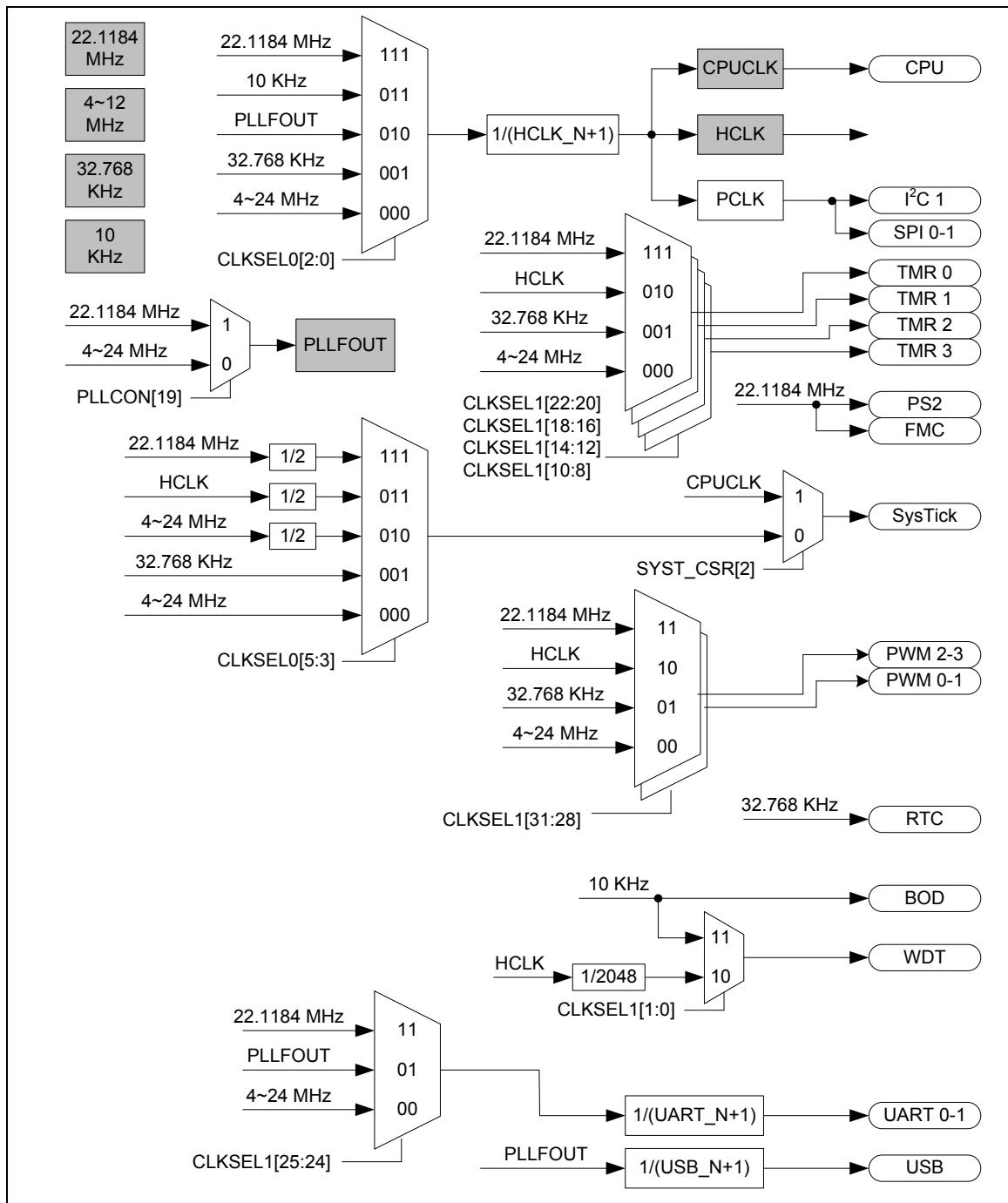


图 5-3 时钟发生器全局框图

5.3.2 时钟发生器

时钟发生器由如下5个时钟源组成：

- 一个外部 32.768 kHz 低速晶振
- 一个外部 4~24 MHz 高速晶振
- 一个可编程的 PLL FOUT(PLL 由外部 4~24 MHz 高速晶振和内部 22.1184 MHz 高速振荡器提供时钟源)
- 一个内部 22.1184 MHz 高速振荡器
- 一个内部 10 kHz 低速振荡器

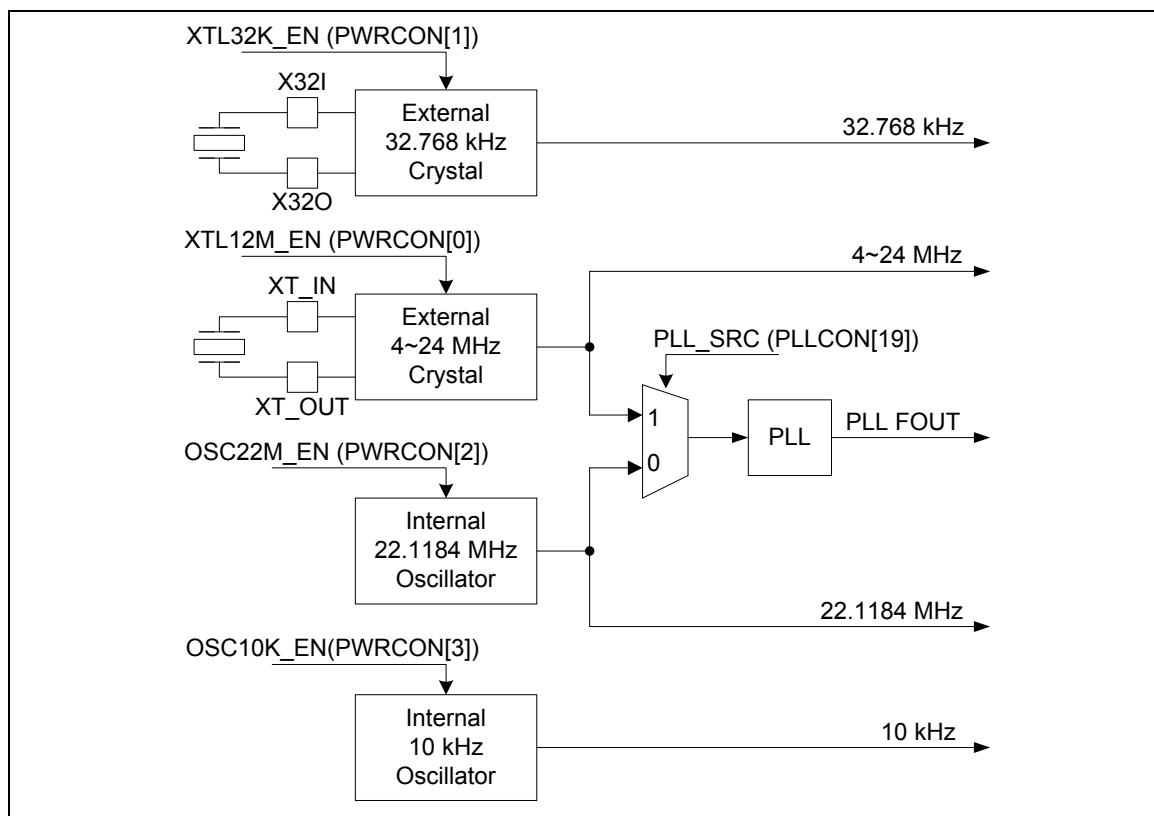


图 5-4 时钟发生器框图

5.3.3 系统时钟与 SysTick 时钟

系统时钟有5个时钟源，由时钟发生器产生。时钟源切换取决于寄存器HCLK_S(CLKSEL0[2:0]), 如图 5-5所示。

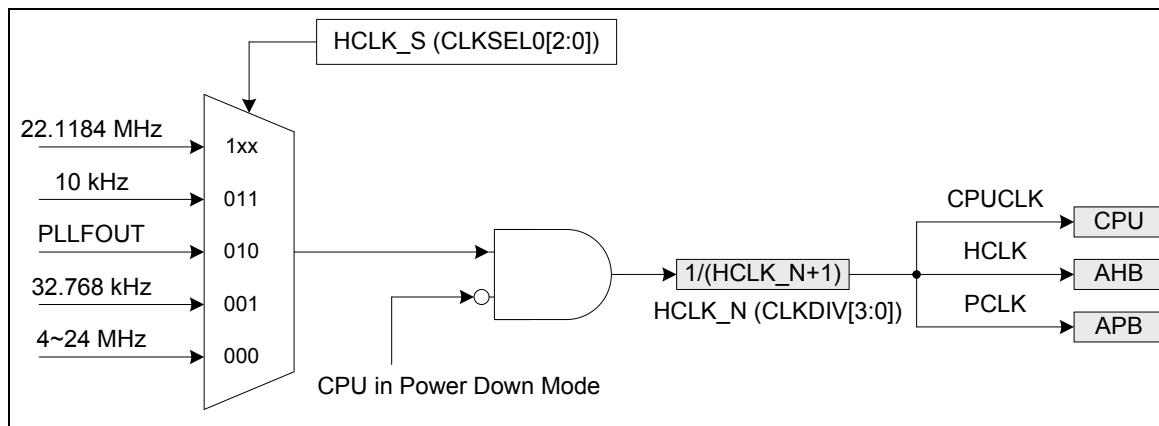


图 5-5 系统时钟框图

Cortex™-M0内核的SysTick 时钟源可以选择CPU时钟或外部时钟(SYST_CSR[2]).如果使用外部时钟, SysTick 时钟 (STCLK) 有5个时钟源，由时钟发生器产生。时钟源切换取决于寄存器 STCLK_S(CLKSEL0[5:3]). 框图如图 5-6.

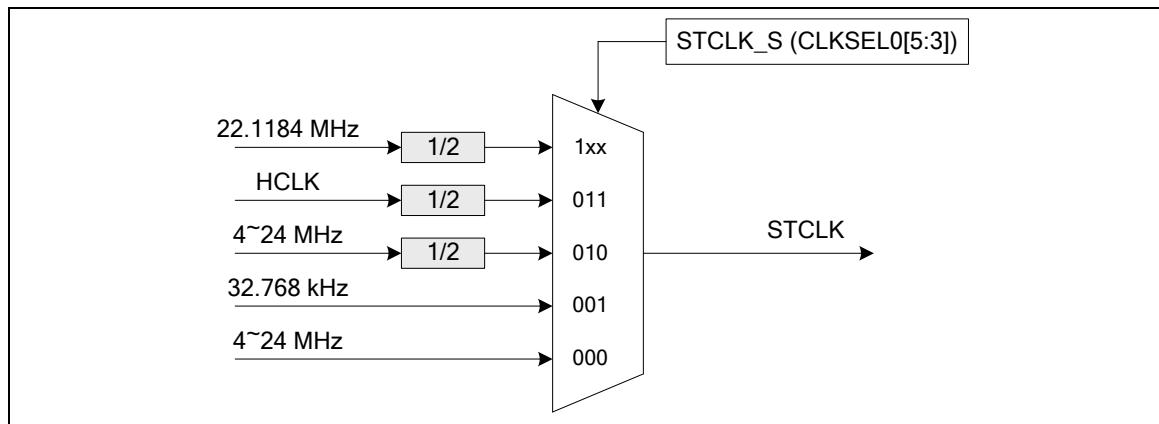


图 5-6 SysTick 时钟控制框图

5.3.4 外围设备时钟

不同的外设，其外围设备时钟有不同的时钟源切换. 参阅5.3.7节寄存器CLKSEL1 & CLKSEL2。

5.3.5 掉电模式时钟

当芯片进入掉电模式后，一些时钟源、外设时钟和系统时钟被关闭，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
 - ◆ 内部 10 KHz 低速振荡器时钟
 - ◆ 外部 32.768 KHz 低速晶振时钟
- 外设时钟 (当这些WDT采用内部10 KHz低速振荡器作为时钟源时或RTC采用外部32.768 KHz低速晶振作为时钟源时)

5.3.6 寄存器映射

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

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

5.3.7 寄存器描述

掉电控制寄存器 (PWRCON)

除BIT[6]外, PWRCON的其他位都受保护, 解锁这些位, 需要向地址0x5000_0100写入“59h”, “16h”, “88h”. 参考寄存器REGWRPROT, 其地址是GCR_BA + 0x100

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

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
PWR_DOWN_EN	PD_WU_STS	PD_WU_INT_EN	PD_WU_DLY	OSC10K_EN	OSC22M_EN	XTL32K_EN	XTL12M_EN

Bits	描述	
[31:9]	保留	保留
[8]	PD_WAIT_CPU	<p>控制进入掉电模式的条件 (写保护位) 1 = 在PWR_DOWN_EN置1且执行WFI指令时, 芯片进入掉电模式下. 0 = 在PWR_DOWN_EN 置1时, 芯片进入掉电模式</p>
[7]	PWR_DOWN_EN	<p>系统掉电模式使能位 (写保护位) 该位置“1”, 使能芯片的掉电模式, 激活芯片的掉电依赖于PD_WAIT_CPU 位 (a) 如果PD_WAIT_CPU 为“0”,在置位PWR_DOWN_EN 后, 芯片立即进入掉电. (b) 如果PD_WAIT_CPU 为“1”, 直到CPU的休眠模式有效时,芯片仍在运行, 然后才掉电 芯片由掉电唤醒, 该位自动清零, 在下次掉电时, 用户需要重新置位该位 掉电模式下, 外部 4~24 MHz 高速晶振与内部 22.1184 MHz 高速振荡器被禁止, 但外部 32.768 KHz 低速晶振与内部 10 KHz 低速振荡器的使能不受该位控制 掉电时, PLL 与系统时钟被禁, 忽略时钟源选择. 如果外设时钟源的为32.768 KHz或10 kHz时钟, 外设的时钟不受掉电模式的控制.</p>

		1 = 芯片立即进入掉电模式 或等待CPU休眠命令WFI 0 = 执行WFI，芯片工作于正常模式或芯片进入idle mode (空闲模式)
[6]	PD_WU_STS	芯片掉电唤醒状态标志 设置“掉电唤醒”，从掉电模式恢复 GPIO, USB, UART, WDT, ACMP, BOD 或 RTC 被唤醒，该标志置位 写 1 清该位.
[5]	PD_WU_INT_EN	掉电模式唤醒的中断使能 (写保护位) 0 = 禁止 1 = 使能 当PD_WU_STS 和 PD_WU_INT_EN 都为高时，产生中断.
[4]	PD_WU_DLY	使能唤醒延时计数器 (写保护位) 芯片由掉电模式唤醒时，时钟控制会延迟一定时钟周期以等待系统时钟稳定. 当芯片工作在外部 4~24 MHz 高速晶振的条件下，延迟时钟周期为 4096 时钟周期，当芯片工作在内部22.1184 MHz 时，延迟256 时钟周期. 1 = 使能时钟周期的延迟 0 = 禁止时钟周期的延迟
[3]	OSC10K_EN	内部 10 KHz 低速振荡器控制 (写保护位) 1 = 使能内部 10 KHz 低速振荡器 0 = 禁止内部 10 KHz 低速振荡器
[2]	OSC22M_EN	内部 22.1184 MHz 高速振荡器控制 (写保护位) 1 = 使能内部 22.1184 MHz 高速振荡器 0 = 禁止内部 22.1184 MHz 高速振荡器
[1]	XTL32K_EN	外部 32.768 KHz 低速晶振控制 (写保护位) 1 = 使能外部 32.768 KHz 低速晶振(正常操作) 0 = 禁止外部 32.768 KHz 低速晶振
[0]	XTL12M_EN	外部 4~24 MHz 高速晶振控制 (写保护位) 该位的缺省值由flash控制器设置，用户配置寄存器config0 [26:24]. 当缺省时钟源为4~24 MHz晶振. 该位自动置1 1 = 使能外部 4~24 MHz 高速晶振 0 = 禁止外部 4~24 MHz 高速晶振

模式	寄存器/指令	PWR_DOWN_EN	PD_WAIT_CPU	CPU run WFI 指令	禁止时钟
正常运行模式	0	0	NO	通过控制寄存器禁止所有时钟	
空闲模式 (CPU进入休眠模式)	0	0	YES	仅禁止CPU时钟	
掉电模式	1	0	NO	大部分时钟停止运行，仅外部10 KHz/32.768 KHz及RTC/WDT可以运行.	
掉电模式 (CPU进入深度休眠模式)	1	1	YES	大部分时钟停止运行，仅外部10 KHz/32.768 KHz及RTC/WDT可以运行.	

表 5-5 掉电模式控制表

芯片进入掉电模式后，要由某个中断唤醒，所以应在进入掉电模式前，使能相应中断NVIC_ISER

AHB 设备时钟使能控制寄存器 (AHBCLK)

该寄存器各位用于使能/禁止系统时钟。

寄存器	偏移量	R/W	描述	复位后的值
AHBCLK	CLK_BA+0x04	R/W	AHB 设备时钟使能控制寄存器	0x0000_0001

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

Bits	描述	
[31:3]	保留	保留
[2]	ISP_EN	Flash ISP控制器时钟使能控制 1 = 使能 the Flash ISP engine 时钟。 0 = 禁止 the Flash ISP engine 时钟
[1:0]	保留	保留



APB 设备时钟使能控制寄存器 (APBCLK)

该寄存器各位用于使能/禁止外设控制器时钟.

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

31	30	29	28	27	26	25	24
PS2_EN	保留			USBD_EN	保留		
23	22	21	20	19	18	17	16
保留		PWM23_EN	PWM01_EN	保留		UART1_EN	UART0_EN
15	14	13	12	11	10	9	8
保留		SPI1_EN	SPI0_EN	保留		I2C1_EN	保留
7	6	5	4	3	2	1	0
保留		TMR3_EN	TMR2_EN	TMR1_EN	TMR0_EN	RTC_EN	WDT_EN

Bits	描述	
[31]	PS2_EN	PS/2 时钟使能控制 1 = 使能 PS/2 时钟 0 = 禁止 PS/2 时钟
[30: 28]	保留	保留
[27]	USBD_EN	USB 2.0 FS 设备控制器时钟使能控制 1 = 使能 USB 时钟 0 = 禁止 USB 时钟
[26:22]	保留	保留
[21]	PWM23_EN	PWM_23 时钟使能控制 1 = 使能 PWM23 时钟 0 = 禁止 PWM23 时钟
[20]	PWM01_EN	PWM_01 时钟使能控制 1 = 使能 PWM01 时钟 0 = 禁止 PWM01 时钟
[19: 18]	保留	保留
[17]	UART1_EN	UART1 时钟使能控制

		1 = 使能 UART1 时钟 0 = 禁止 UART1 时钟
[16]	UART0_EN	UART0 时钟使能控制 1 = 使能 UART0 时钟 0 = 禁止 UART0 时钟
[15: 14]	保留	保留
[13]	SPI1_EN	SPI1 时钟使能控制 1 = 使能 SPI1 时钟 0 = 禁止 SPI1 时钟
[12]	SPI0_EN	SPI0 时钟使能控制 1 = 使能 SPI0 时钟 0 = 禁止 SPI0 时钟
[11:10]	保留	保留
[9]	I2C1_EN	I²C1 时钟使能控制 1 = 使能 I ² C1 时钟 0 = 禁止 I ² C1 时钟
[5]	TMR3_EN	Timer3 时钟使能控制 1 = 使能 Timer3 时钟 0 = 禁止 Timer3 时钟
[4]	TMR2_EN	Timer2 时钟使能控制 1 = 使能 Timer2 时钟 0 = 禁止 Timer2 时钟
[3]	TMR1_EN	Timer1 时钟使能控制 1 = 使能 Timer1 时钟 0 = 禁止 Timer1 时钟
[2]	TMR0_EN	Timer0 时钟使能控制 1 = 使能 Timer0 时钟 0 = 禁止 Timer0 时钟
[1]	RTC_EN	Real-Time-Clock APB 接口时钟控制 该位仅用于控制RTC APB时钟， RTC时钟源由外部 32.768 KHz 低速晶振提供。 1 = 使能 RTC 时钟 0 = 禁止 RTC 时钟

[0]	WDT_EN	看门狗定时器时钟使能控制 (写保护位) 该位是受保护的位，编程时，需要向0x5000_0100 依次写入“59h”，“16h”，“88h”来解锁定，参考寄存器 REGWRPROT（地址GCR_BA+0x100）。 缺省值由flash控制设置，用户可配置寄存器congig0 bit[31] 1 = 使能看门狗定时器时钟 0 = 禁止看门狗定时器时钟
-----	---------------	--

时钟状态寄存器 (CLKSTATUS)

该寄存器各位用于监控芯片时钟源是否稳定，时钟切换是否失败

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

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
CLK_SW_FAIL	保留		OSC22M_STB	OSC10K_STB	PLL_STB	XTL32K_STB	XTL12M_STB

Bits	描述	
[31:8]	保留	保留
[7]	CLK_SW_FAIL	<p>时钟切换失败标志 (写保护位) 1 = 时钟切换失败 0 = 时钟切换成功 该位在软件切换系统时钟源时更新时，如果切换的目标时钟稳定，该位设置为1'b0. 如果切换目标时钟不稳定，该位设置为1. 该位写1清零.</p>
[6:5]	保留	保留
[4]	OSC22M_STB	<p>内部 22.1184 MHz 高低速振荡器时钟源稳定标志 1 = OSC22M 时钟稳定 0 = OSC22M 时钟不稳定或没有使能 该位只读</p>
[3]	OSC10K_STB	<p>内部 10 KHz 低速振荡器时钟源稳定标志 1 = OSC10K 时钟稳定 0 = OSC10K 时钟不稳定或没有使能 该位只读</p>

[2]	PLL_STB	PLL 时钟源稳定标志 1 = PLL 时钟稳定 0 = PLL 时钟不稳定或没有使能 该位只读
[1]	XTL32K_STB	外部 32.768 KHz 低速晶振时钟源稳定标志 1 = XTL32K 时钟稳定 0 = XTL32K 时钟不稳定或没有使能 该位只读
[0]	XTL12M_STB	外部 4~24 MHz 高速晶振时钟源稳定标志 1 = XTL12M 时钟稳定 0 = XTL12M 时钟不稳定或没有使能 该位只读

时钟源选择控制寄存器 0 (CLKSEL0)

寄存器	偏移量	R/W	描述	复位后的值
CLKSEL0	CLK_BA+0x10	R/W	时钟源选择控制寄存器 0	0xFFFF_FFFX

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

Bits	描述	
[31:6]	保留	保留
[5:3]	STCLK_S	<p>Cortex™-M0 SysTick 时钟源选择 (写保护位) 如果 SYST_CSR[2]=0, SysTick 使用下表的时钟源 该位受保护, 修改该位时, 需要依次向0x5000_0100写入"59h", "16h", "88h", 参 考寄存器 REGWRPROT, 地址GCR_BA + 0x100.</p> <p>000 = 外部 4~24 MHz 高速晶振 001 = 外部 32.768 KHz 低速晶振 010 = 外部 4~24 MHz 高速晶振/2分频 011 = HCLK/2分频 111 = 内部 22.1184 MHz 高速振荡器/2分频</p>
[2:0]	HCLK_S	<p>HCLK时钟源选择 (写保护位) 在时钟切换到相关时钟源(预选和新选)时必须打开 任何复位后, 加载用户配置寄存器CFOSC(Config0[26:24])的值, 缺省值可为 000b 或 111b. 该位受保护, 修改该位时, 需要依次向0x5000_0100写入"59h", "16h", "88h", 参 考寄存器 REGWRPROT, 地址GCR_BA + 0x100.</p> <p>000 = 外部 4~24 MHz 高速晶振 001 = 外部 32.768 KHz 低速晶振 010 = PLL 时钟 011 = 内部 10 KHz 低速振荡器 111 = 内部 22.1184 MHz 高速振荡器</p>

时钟源选择控制寄存器 1 (CLKSEL1)

在时钟切换到相关的时钟源前，必须打开相关的时钟源(预选和新选)。

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

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

Bits	描述
[31:30]	PWM23_S PWM2 与 PWM3 的时钟源选择 PWM2与 PWM3使用相同的时钟源和相同的分频 00 = 外部 4~24 MHz 高速晶振 01 = 外部 32.768 KHz 低速晶振 10 = HCLK 11 = 内部 22.1184 MHz 高速振荡器
[29:28]	PWM01_S PWM0 与 PWM1 的时钟源选择 PWM0与 PWM1使用相同的时钟源和相同的分频 00 = 外部 4~24 MHz 高速晶振 01 = 外部 32.768 KHz 低速晶振 10 = HCLK 11 = 内部 22.1184 MHz 高速振荡器
[27:26]	保留
[25:24]	UART_S UART 时钟源选择 00 = 外部 4~24 MHz 高速晶振 01 = PLL 11 = 内部 22.1184 MHz 高速振荡器
[23]	保留

[22:20]	TMR3_S	TIMER3 时钟源选择 000 = 外部 4~24 MHz 高速晶振 001 = 外部 32.768 KHz 低速晶振 010 = HCLK 011 = 保留 111 = 内部 22.1184 MHz 高速振荡器
[19]	保留	保留
[18:16]	TMR2_S	TIMER2 时钟源选择 000 = 外部 4~24 MHz 高速晶振 001 = 外部 32.768 KHz 低速晶振 010 = HCLK 011 = 保留 111 = 内部 22.1184 MHz 高速振荡器
[15]	保留	保留
[14:12]	TMR1_S	TIMER1 时钟源选择 000 = 外部 4~24 MHz 高速晶振 001 = 外部 32.768 KHz 低速晶振 010 = HCLK 011 = 保留 111 = 内部 22.1184 MHz 高速振荡器
[11]	保留	保留
[10:8]	TMR0_S	TIMER0 时钟源选择 000 = 外部 4~24 MHz 高速晶振 001 = 外部 32.768 KHz 低速晶振 010 = HCLK 011 = 保留 111 = 内部 22.1184 MHz 高速振荡器
[7:2]	保留	保留
[1:0]	WDT_S	看门狗定时器时钟源选择 (写保护位) 该位受保护, 修改该位时, 需要依次向0x5000_0100写入"59h", "16h", "88h", 参考寄存器 REGWRPROT, 地址GCR_BA + 0x100. 00 = 保留 01 = 保留 10 = HCLK/2048 clock 11 = 内部 10 KHz 低速振荡器

时钟分频寄存器 (CLKDIV)

寄存器	偏移量	R/W	描述	复位后的值
CLKDIV	CLK_BA+0x18	R/W	时钟分频寄存器	0x0000_0000

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

Bits	描述	
[31:12]	保留	保留
[11:8]	UART_N	UART 时钟源的时钟除频数 UART 时钟频率 = (UART 时钟源频率) / (UART_N + 1)
[7:4]	USB_N	USB 时钟由PLL时钟除频数 USB时钟频率 = (PLL 频率) / (USB_N + 1)
[3:0]	HCLK_N	HCLK 时钟源的时钟除频数 HCLK 时钟频率 = (HCLK 时钟源频率) / (HCLK_N + 1)

PLL控制寄存器 (PLLCON)

PLL的参考时钟源来自外部 4~24 MHz 高速晶振或内部 22.1184 MHz 高速振荡器。该寄存器用于控制PLL的输出频率和PLL的操作模式

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

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

Bits	描述	
[31:20]	保留	保留
[19]	PLL_SRC	PLL 时钟源选择 1 = PLL 时钟源为内部 22.1184 MHz 高速振荡器 0 = PLL时钟源为外部 4~24 MHz 高速晶振
[18]	OE	PLL OE (FOUT enable) 引脚控制 0 = 使能PLL FOUT 1 = PLL FOUT 为低
[17]	BP	PLL 旁路控制 0 = PLL 正常模式 (default) 1 = PLL输出时钟频率与输入时钟相同(XTALin)
[16]	PD	掉电模式 如果设置PWRCON寄存器的 PWR_DOWN_EN 位为1, PLL将进入掉电模式 0 = PLL正常模式 1 = PLL掉电模式(default)
[15:14]	OUT_DV	PLL 输出分频控制引脚 参考下表公式

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

输出时钟频率设置

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

约束条件:

1. $3.2MHz < F_{IN} < 150MHz$

2. $800KHz < \frac{FIN}{2 * NR} < 7.5MHz$

3. $100MHz < FCO = FIN \times \frac{NF}{NR} < 500MHz$

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

默认频率设置

默认值: 0xC22E

 $F_{IN} = 12MHz$ $NR = (1+2) = 3$ $NF = (46+2) = 48$ $NO = 4$

$F_{OUT} = 12/4 \times 48 \times 1/3 = 48MHz$

48 MHz	50 MHz	60 MHz
0xC22E	0xC230	0xC23A

5.4 USB 设备控制器 (USB)

5.4.1 概述

该器件有一组全速USB 2.0设备控制器和收发器。符合USB 2.0规范，支持control/bulk/interrupt/isochronous传输类型。

在该设备控制器里，包含两个主接口：APB总线与由USB PHY收发器出来的USB总线。CPU通过APB总线编程控制寄存器。在该控制器内置有512字节的SRAM作为数据缓存。输入或输出传输，需要通过APB接口或SIE向SRAM写数据或从SRAM读数据。用户需要通过BUFSEGx为每个端点缓存设置有效的SRAM地址。

USB设备控制器共有6个可配置的端点，每个端点可以配置为IN, OUT类型。每个端点可以配置为control/bulk/interrupt/isochronous传输类型。在接收或发送数据包之前，要先配置好设备地址(主机SET_ADDRESS之后)和每个端点的端点号，在最大有效负荷寄存器(MXPLDx)中定义每个端点的最大收/发长度及处理主机与设备的信号交换。

该控制器有4个不同的中断事件。包括唤醒功能，设备插拔事件，USB事件(如IN ACK, OUT ACK等)，和BUS事件(如suspend和resume, 等)。任何事件都可以引起一个中断，用户只需要在中断事件状态寄存器(USB_INTSTS)里检查相关事件标志以得知发生何种中断，然后检测相关USB端点状态寄存器(USB_EPSTS)以得知在该端点上发生哪种事件。

USB设备有一个软件禁用功能，用于模拟设备插拔主机的情况。如果用户使能DRVSE0位(USB_DRVSE0)，USB控制器将USB_DP和USB_DM输出低电平禁止其功能。在关闭DRVSE0位后，主机将重新枚举USB设备。

参考：通用串行总线规格修订版 1.1

5.4.2 特征

该通用串行总线(USB)用一个连接器来连接所有的USB外设到主机系统。下面是USB的一些特征。

- 兼容USB 2.0全速规格
- 提供1个中断向量，4个中断事件(WAKEUP, FLDET, USB 和 BUS)
- 支持Control/Bulk/Interrupt/Isochronous传输类型
- 在没有总线活动超过3ms后支持暂停功能
- 为可配置的Control/Bulk/Interrupt/Isochronous传输类型提供6个端点和最大512字节的缓存
- 提供远程唤醒功能

5.4.3 框图

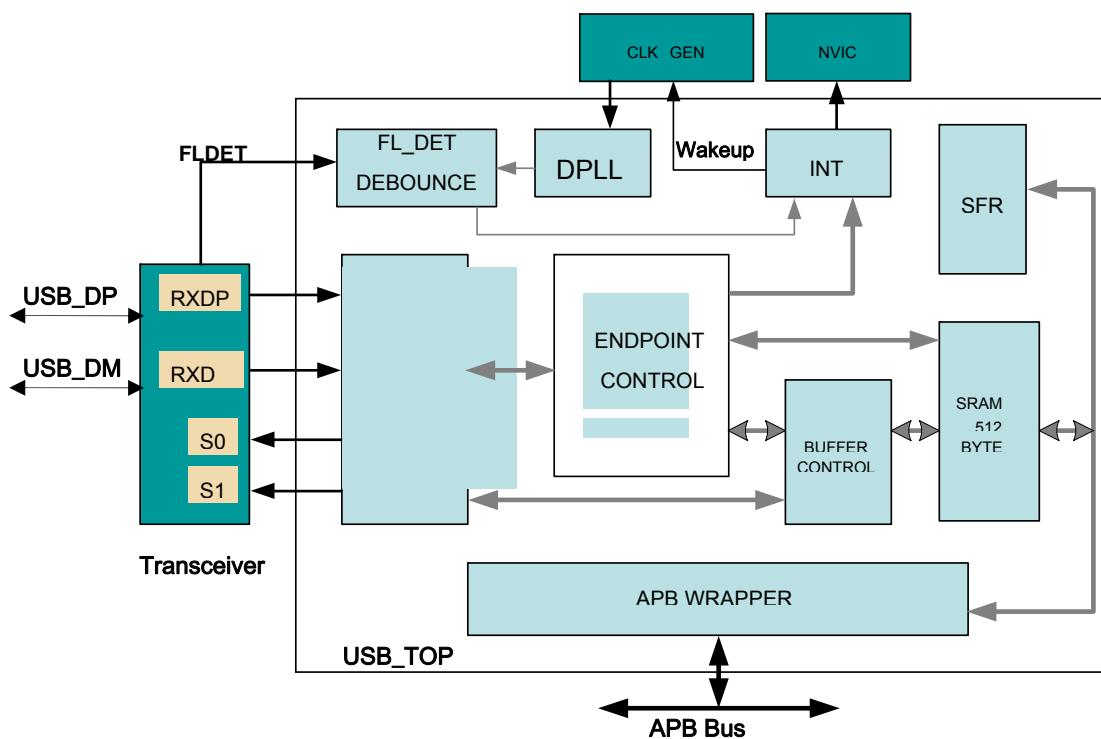


图 5-7 USB 框图

5.4.4 功能描述

5.4.4.1 SIE (Serial Interface Engine)

SIE是设备控制器的前端，处理大多数的USB 协议包。SIE典型的包括信号向上发送到达事务处理层面。处理功能包括：

- 包识别，事务排序
- SOP, EOP, RESET, RESUME信号检测/产生
- Clock/Data分离
- NRZI 数据编解码与比特填塞（bit-stuffing）
- CRC产生和校验 (for Token and Data)

- Packet ID (PID) 产生和校验/解码
- 串-并/并-串转换

5.4.4.2 端点控制

该控制器有6个端点. 每个端点可以配置成Control, Bulk, Interrupt, 或 Isochronous传输类型. 所有操作包括Control, Bulk, Interrupt 和 Isochronous 传输都在该模块内执行. 也可以用来管理数据同步时序, 端点状态控制, 当前端点地址, 当前事务状态和每个端点的数据缓冲状态.

5.4.4.3 数字锁相环

USB数据的比特率为12 MHz, DPLL采用的48 MHz的频率由时钟控制器产生, 用来锁定RXDP与RXDM的输入数据, 12 MHz的比特率时钟也是由DPLL转换来的.

5.4.4.4 插拔去抖动

USB设备可以进行热插拔操作, 为了监测USB设备被拔出时的状态, 设备控制器提供了硬件去抖动以防止在USB插拔时产生的抖动问题, 悬空检测中断产生于USB进行插拔操作的10ms后, 用户可以通过读取“USB_FLDET”寄存器的值, 来应答USB插拔. “FLODET” 标志代表在当前总线上没有经过去抖动处理的状态. 若用户要通过这个标志来检测USB的状态 (不通过中断的方式), 则需要添加软件去抖动功能。

5.4.4.5 中断

该USB提供1个中断信号有4个中断事件的中断向量 (WAKEUP, FLDET, USB, BUS). WAKEUP 中断被用来在掉电模式下唤醒系统时钟。(掉电模式功能在系统掉电控制寄存器PWRCON中定义.) FLDET 中断用于 USB 插拔, USB 事件告知用户一些USB请求, 如IN ACK, OUT ACK等, BUS 事件告知用户一些总线事件, 如 暂停、复位等. 用户必须在USB设备控制器的中断使能寄存器 (USB_INTEN) 设置相应的位以使能USB中断。

唤醒中断会出现在芯片进入掉电模式且唤醒事件发生时. 在芯片进入掉电模式后, D+和D-的任何变化都能唤醒芯片 (假定USB唤醒功能使能时). 若有非正常的变化, 例如噪音造成的变化, 将唤醒系统并发生唤醒中断。在USB唤醒超过 20 ms后, 若没有其它USB中断事件出现, 唤醒中断将发生。下图为唤醒中断的控制流程。

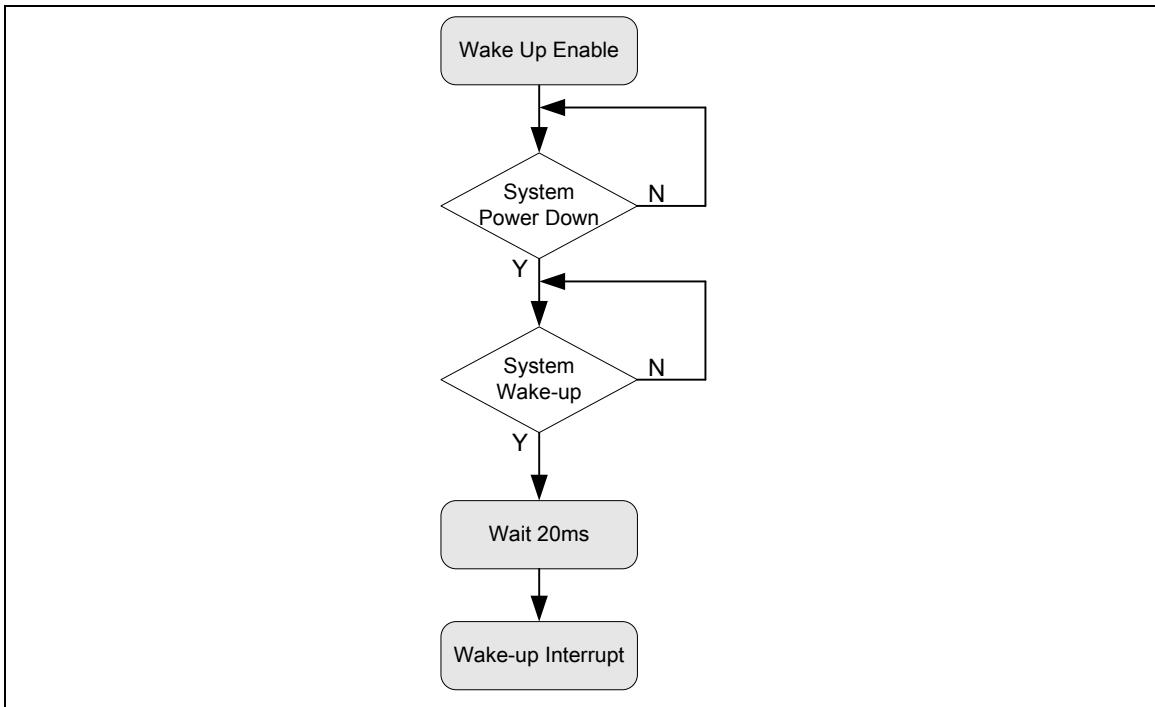


图 5-8 唤醒中断的操作流程

USB 中断告知用户所有总线上的 USB 事件，用户可以读特殊功能寄存器 EPSTS (USB_EPSTS[25:8]) 和 EPEVT5~0 (USB_INTSTS[21:16]) 来获知是哪类请求，对哪个端点采取响应。

与USB中断相似，BUS中断告之用户一些总线事件，如USB复位，中止，暂停和恢复，用户可以读特殊功能寄存器“USB_ATTR”获取总线事件。

5.4.4.6 省电

当芯片进入掉电模式时，USB 自动关闭PHY收发器省电，此外，在特殊环境下，用户可以给特殊功能寄存器USB_ATTR [4] 写入“0”关闭PHY进入省电状态。

5.4.4.7 缓冲控制

USB控制器有512 bytes SRAM和6个端点共享该缓冲。在USB功能有效前，用户需要在缓冲段寄存器配置每个端点的有效起始地址。BUFFER CONTROL 用于控制每个端点的有效起始地址和SRAM 的大小（在寄存器MXPLD定义）。

Error! Reference source not found. 根据BUFSEG和MXPLD寄存器的内容描述每个端点的起始地址。如果BUFSEG0 设置为0x08h 和 MXPLD0 设置为0x40h, 端点0的SRAM 大小从USB_BA + 0x108h开始，到 USB_BA + 0x148h结束。(注：USB SRAM 基地址为USB_BA + 0x100h)。

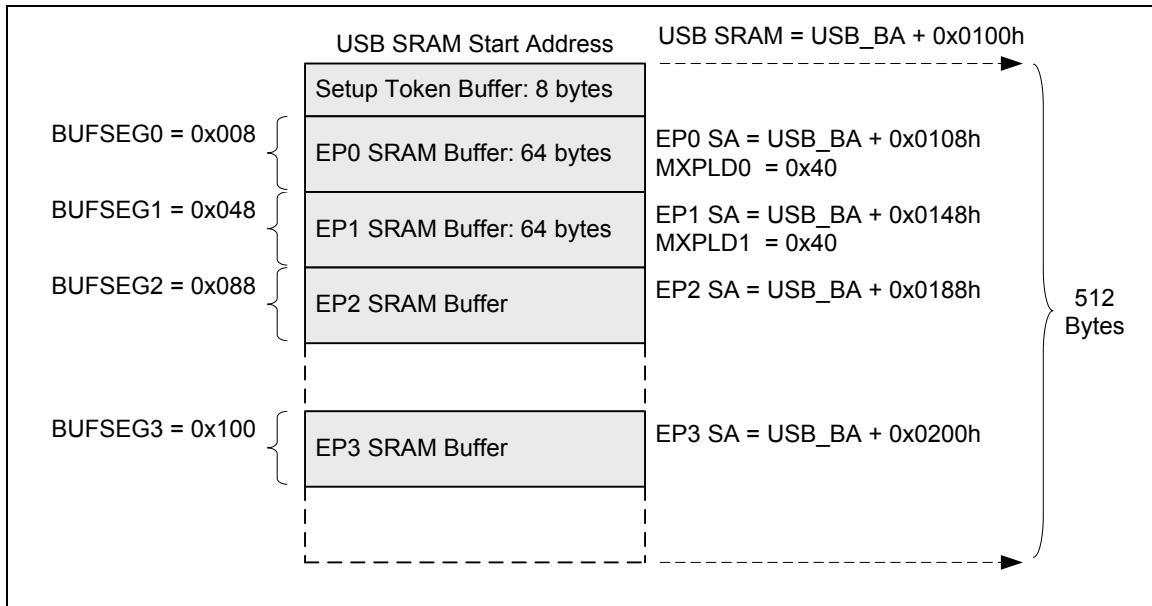


图 5-9 端点 SRAM 的结构

5.4.4.8 与USB外设通信处理

用户可以采用中断或检测USB_INTSTS来监测USB通信，在USB通信发生时，USB_INTSTS由硬件设置，并向CPU发送中断请求（如果相关中断使能），在没有中断时，用户可以检测USB_INTSTS来获取事件，以下是控制流的中断使能。

USB主机向设备控制器请求数据时，用户需要预先准备相关的数据到指定的端点缓存。在将数据写入缓冲区后，用户需要写入实际数据长度到指定的MAXPLD寄存器。一旦这个寄存器被写入数据，内部信号“In_Rdy”会被设置，当收到主机发送的相关IN token之后，缓冲数据将被立刻传送。在传送指定数据之后，信号“In_Rdy”会由硬件自动清除。

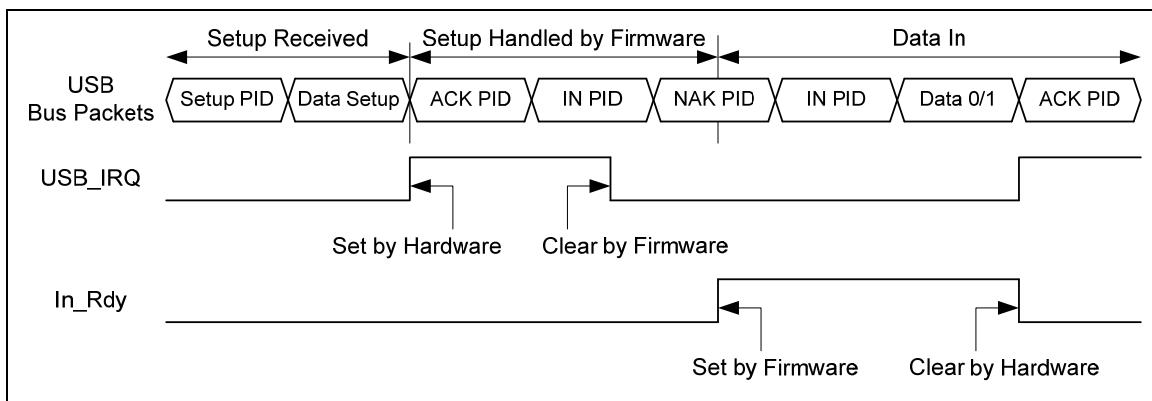


图 5-10 数据传入时间里传输流程图

USB主机要发送数据到设备控制器的OUT端点，硬件将这些数据存在指定的端点缓存里，通信完成

后，硬件在MAXPLD记录数据长度，并清除“Out_Rdy”信号，这避免硬件在用户没有取走当前数据时接收下一个数据。一旦用户处理了这次通信时，由软件写入相关的寄存器“MAXPLD”来设置“Out_Rdy”信号以接收下一次通信

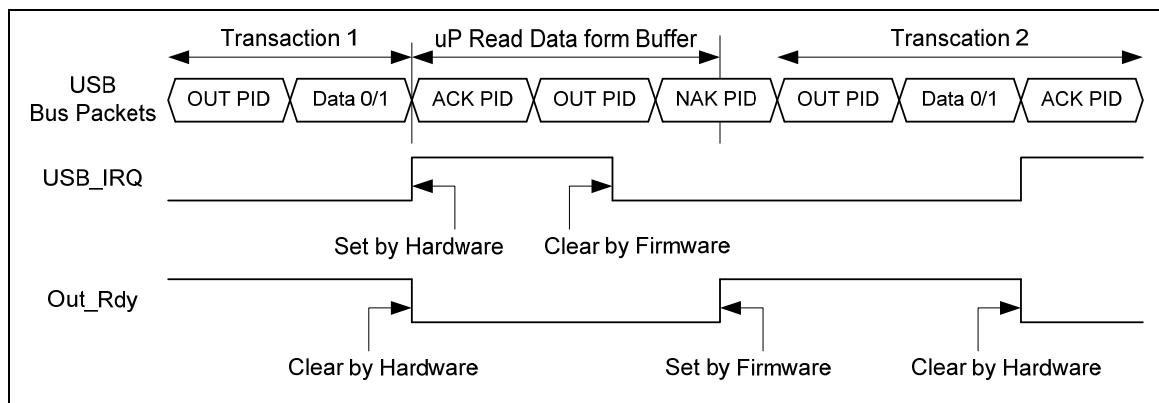


图 5-11 数据输出图



5.4.5 寄存器映射

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

寄存器	偏移量	R/W	描述	复位后的值
USB_BA = 0x4006_0000				
USB_INTEN	USB_BA+0x000	R/W	USB 中断使能寄存器	0x0000_0000
USB_INTSTS	USB_BA+0x004	R/W	USB 中断事件状态寄存器	0x0000_0000
USB_FADDR	USB_BA+0x008	R/W	USB 设备功能地址寄存器	0x0000_0000
USB_EPSTS	USB_BA+0x00C	R	USB 端点状态寄存器	0x0000_0000
USB_ATTR	USB_BA+0x010	R/W	USB 总线状态和归属寄存器	0x0000_0040
USB_FLDET	USB_BA+0x014	R	USB 悬空检测寄存器	0x0000_0000
USB_BUFSEG	USB_BA+0x018	R/W	Setup Token 缓存偏移寄存器	0x0000_0000
USB_BUFSEG0	USB_BA+0x020	R/W	端点0 的缓存偏移寄存器	0x0000_0000
USB_MXPLD0	USB_BA+0x024	R/W	端点0 的最大有效载荷	0x0000_0000
USB_CFG0	USB_BA+0x028	R/W	端点0 的配置	0x0000_0000
USB_CFGP0	USB_BA+0x02C	R/W	端点0 的设置延迟与清除 In/Out 准备控制寄存器	0x0000_0000
USB_BUFSEG1	USB_BA+0x030	R/W	端点1 的缓存偏移寄存器	0x0000_0000
USB_MXPLD1	USB_BA+0x034	R/W	端点1 的最大有效载荷	0x0000_0000
USB_CFG1	USB_BA+0x038	R/W	端点1 的配置	0x0000_0000
USB_CFGP1	USB_BA+0x03C	R/W	端点1 的设置延迟与清除 In/Out 准备控制寄存器	0x0000_0000
USB_BUFSEG2	USB_BA+0x040	R/W	端点2 的缓存偏移寄存器	0x0000_0000
USB_MXPLD2	USB_BA+0x044	R/W	端点2 的最大有效载荷	0x0000_0000
USB_CFG2	USB_BA+0x048	R/W	端点2 的配置	0x0000_0000
USB_CFGP2	USB_BA+0x04C	R/W	端点2 的设置延迟与清除 In/Out 准备控制寄存器	0x0000_0000
USB_BUFSEG3	USB_BA+0x050	R/W	端点3 的缓存偏移寄存器	0x0000_0000
USB_MXPLD3	USB_BA+0x054	R/W	端点3 的最大有效载荷	0x0000_0000
USB_CFG3	USB_BA+0x058	R/W	端点3 的配置	0x0000_0000
USB_CFGP3	USB_BA+0x05C	R/W	端点3 的设置延迟与清除 In/Out 准备控制寄存器	0x0000_0000
USB_BUFSEG4	USB_BA+0x060	R/W	端点4 的缓存偏移寄存器	0x0000_0000
USB_MXPLD4	USB_BA+0x064	R/W	端点4 的最大有效载荷	0x0000_0000

USB_CFG4	USB_BA+0x068	R/W	端点4 的配置	0x0000_0000
USB_CFGP4	USB_BA+0x06C	R/W	端点4 的设置延迟与清除 In/Out 准备控制寄存器	0x0000_0000
USB_BUFSEG5	USB_BA+0x070	R/W	端点5 的缓存偏移寄存器	0x0000_0000
USB_MXPLD5	USB_BA+0x074	R/W	端点5 的最大有效载荷	0x0000_0000
USB_CFG5	USB_BA+0x078	R/W	端点5 的配置	0x0000_0000
USB_CFGP5	USB_BA+0x07C	R/W	端点5 的设置延迟与清除 In/Out 准备控制寄存器	0x0000_0000
USB_DRVSE0	USB_BA+0x090	R/W	USB 驱动 SE0 控制寄存器	0x0000_0001

内存类型	地址	大小	描述
USB_BA = 0x4006_0000			
SRAM	USB_BA+0x100 ~ USB_BA+0x2FF	512 Bytes	SRAM用于整个端点缓冲. 参考5.4.4.7 的端点SRAM结构和描述.

5.4.6 寄存器描述

USB中断使能寄存器 (USB_INTEN)

寄存器	偏移量	R/W	描述	复位后的值
USB_INTEN	USB_BA+0x000	R/W	USB 中断使能寄存器	0x0000_0000

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

Bits	描述	
[31:16]	保留	保留
[15]	INNAK_EN	<p>激活收到 IN token 的时候的 NAK 中断 1 = 当它置1并且收到INtoken时应答NAK，将发生INNAK中断，NAK状态被更新到断点状态寄存器USB_EPSTS。 0 = 当它置0并且收到IN token应答NAK时不会发生中断，NAK状态不会被更新到断点状态寄存器USB_EPSTS。</p>
[14:9]	保留	保留
[8]	WAKEUP_EN	<p>唤醒功能使能控制 1 = 使能USB 唤醒CPU功能 0 = 禁止USB 唤醒CPU功能</p>
[7:4]	保留	保留
[3]	WAKEUP_IE	<p>使能 USB 唤醒 CPU 中断 1 = 使能唤醒CPU中断 0 = 禁止唤醒CPU中断</p>
[2]	FLDET_IE	<p>使能悬空检测中断 1 = 使能悬空检测中断 0 = 禁止悬空检测中断</p>

[1]	USB_IE	使能 USB 事件中断 1 = 使能 USB 事件中断 0 = 禁止USB 事件中断
[0]	BUS_IE	使能总线事件中断 1 = 使能总线事件中断 0 = 禁止总线事件中断

USB 中断事件状态寄存器 (USB_INTSTS)

该寄存器是USB中断事件的标志寄存器，通过向相应位写1实现清零。

寄存器	偏移量	R/W	描述	复位后的值
USB_INTSTS	USB_BA+0x004	R/W	USB 中断事件标志	0x0000_0000

31	30	29	28	27	26	25	24
SETUP	保留						
23	22	21	20	19	18	17	16
保留		EPEVT5	EPEVT4	EPEVT3	EPEVT2	EPEVT1	EPEVT0
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留				WAKEUP_STS	FLDET_STS	USB_STS	BUS_STS

Bits	描述	
[31]	SETUP	Setup 事件状态 1 = Setup事件发生，向USB_INTSTS[31]写1清标志 0 = 无Setup事件
[30:22]	保留	保留
[21]	EPEVT5	端点 5 的 USB 事件状态 1 = 端点5上的USB事件，检查 USB_EPSTS[25:23] 可知何种USB事件发生，通过向USB_INTSTS[21] 或 USB_INTSTS[1]写1清零 0 = 端点5没有事件发生
[20]	EPEVT4	端点 4 的 USB 事件状态 1 = 端点4上的USB事件，检查 USB_EPSTS[22:20]可知何种USB事件发生，通过向USB_INTSTS[20] 或 USB_INTSTS[1]写1清零 0 = 端点4没有事件发生
[19]	EPEVT3	端点 3 的 USB 事件状态 1 = 端点3上的USB事件，检查 USB_EPSTS[19:17]可知何种USB事件发生，通过向USB_INTSTS[19] 或 USB_INTSTS[1]写1清零 0 = 端点3没有事件发生
[18]	EPEVT2	端点 2 的 USB 事件状态

		1 = 端点2上的USB事件, 检查 USB_EPSTS[16:14]可知何种USB事件发生, 通过向USB_INTSTS[18] 或 USB_INTSTS[1]写1清零 0 = 端点2没有事件发生
[17]	EPEVT1	端点 1 的 USB 事件状态 1 = 端点1上的USB事件, 检查 USB_EPSTS[13:11]可知何种USB事件发生, 通过向USB_INTSTS[17] 或 USB_INTSTS[1]写1清零 0 = 端点1没有事件发生
[16]	EPEVT0	端点 0 的 USB 事件状态 1 = 端点0上的USB事件, 检查 USB_EPSTS[10:8]可知何种USB事件发生, 通过向USB_INTSTS[16] 或 USB_INTSTS[1]写1清零 0 = 端点1没有事件发生
[15:4]	保留	保留
[3]	WAKEUP_STS	唤醒中断状态 1 = 唤醒事件发生, 向USB_INTSTS[3]写1清零 0 = 无唤醒中断状态
[2]	FLDET_STS	悬空检测中断状态 1 = USB总线上有连接/分离事件, 向USB_INTSTS[2]写1清零. 0 = USB总线上没有连接/分离事件
[1]	USB_STS	USB 事件中断状态 USB 事件包括Setup Token, IN Token, OUT ACK, ISO IN, or ISO OUT. 1 = USB 事件发生, 检查EPSTS0~5[2:0] 可知何种USB事件发生, 向USB_INTSTS[1] 或 EPSTS0~5 和 SETUP (USB_INTSTS[31])写1清零 0 = 无USB事件发生
[0]	BUS_STS	总线中断状态 总线事件意味着总线上存在一定挂起或重起功能. 1 = 总线事件发生; 检查USB_ATTR[3:0] 可知何种总线事件发生, 向USB_INTSTS[0]写1清零. 0 = 无总线事件发生

USB 设备功能地址寄存器 (USB_FADDR)

在USB总线上作为设备地址的七位值.

寄存器	偏移量	R/W	描述	复位后的值
USB_FADDR	USB_BA+0x008	R/W	USB 设备功能地址寄存器	0x0000_0000

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

Bits	描述	
[31:7]	保留	保留
[6:0]	FADDR	USB 设备地址。在主机分配地址之后，把地址写在这个寄存器

USB 端点状态寄存器 (USB_EPSTS)

寄存器	偏移量	R/W	描述	复位后的值
USB_EPSTS	USB_BA+0x00C	R	USB端点状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留						EPSTS5[2:1]	
23	22	21	20	19	18	17	16
EPSTS5[0]	EPSTS4[2:0]			EPSTS3[2:0]			EPSTS2[2]
15	14	13	12	11	10	9	8
EPSTS2[1:0]		EPSTS1[2:0]			EPSTS0[2:0]		
7	6	5	4	3	2	1	0
OVERRUN	保留						

Bits	描述	
[31:26]	保留	保留
[25:23]	EPSTS5	<p>端点 5 总线状态 这些位用于表示该端点当前的状态 000 = In ACK 001 = In NAK 010 = Out Packet Data0 ACK 110 = Out Packet Data1 ACK 011 = Setup ACK 111 = Isochronous transfer end</p>
[22:20]	EPSTS4	<p>端点 4 总线状态 这些位用于表示该端点当前的状态 000 = In ACK 001 = In NAK 010 = Out Packet Data0 ACK 110 = Out Packet Data1 ACK 011 = Setup ACK 111 = Isochronous transfer end</p>
[19:17]	EPSTS3	<p>端点 3 总线状态 这些位用于表示该端点当前的状态</p>

		000 = In ACK 001 = In NAK 010 = Out Packet Data0 ACK 110 = Out Packet Data1 ACK 011 = Setup ACK 111 = Isochronous transfer end
[16:14]	EPSTS2	端点 2 总线状态 这些位用于表示该端点当前的状态 000 = In ACK 001 = In NAK 010 = Out Packet Data0 ACK 110 = Out Packet Data1 ACK 011 = Setup ACK 111 = Isochronous transfer end
[13:11]	EPSTS1	端点 1 总线状态 这些位用于表示该端点当前的状态 000 = In ACK 001 = In NAK 010 = Out Packet Data0 ACK 110 = Out Packet Data1 ACK 011 = Setup ACK 111 = Isochronous transfer end
[10:8]	EPSTS0	端点 0 总线状态 这些位用于表示该端点当前的状态 000 = In ACK 001 = In NAK 010 = Out Packet Data0 ACK 110 = Out Packet Data1 ACK 011 = Setup ACK 111 = Isochronous transfer end
[7]	OVERRUN	Overrun 表示接收到的数据超过最大值。 1 = 表示主机Out Data大于寄存器MXPLD 的Max Payload 或Setup Data大于8 Bytes 0 = 没有溢出
[6:0]	保留	保留

USB 总线状态和归属寄存器 (USB_ATTR)

寄存器	偏移量	R/W	描述	复位后的值
USB_ATTR	USB_BA+0x010	R/W	USB 总线状态和归属寄存器	0x0000_0040

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留					BYTEM	PWRDN	DPPU_EN
7	6	5	4	3	2	1	0
USB_EN	保留	RWAKEUP	PHY_EN	TIMEOUT	RESUME	SUSPEND	USRST

Bits	描述	
[31:11]	保留	保留
[10]	BYTEM	CPU 访问 USB SRAM 字节大小模式选择 1 = Byte Mode: 从CPU到USB SRAM传输的大小仅为Byte. 0 = Word Mode: 从CPU到USB SRAM传输的大小仅为 Word (4 bytes) .
[9]	PWRDN	PHY 发送器掉电, 低电平有效 1 = 打开PHY相关电路 0 = 关闭PHY相关电路
[8]	DPPU_EN	使能 USB_DP 的上拉电阻 1 = USB_DP 的上拉电阻有效 0 = 禁止USB_DP 的上拉电阻
[7]	USB_EN	使能 USB 控制器 1 = 使能USB 控制器 0 = 禁止USB控制器
[6]	保留	保留
[5]	RWAKEUP	使能远程唤醒 1 = 使USB总线到K状态 (USB_DP low, USB_DM: high), 用于远程唤醒 0 = 将USB总线由K状态释放

[4]	PHY_EN	使能 PHY 发送器功能 1 = 使能PHY 送发器功能 0 = 禁止PHY 发送器功能
[3]	TIMEOUT	超时状态 1 = 总线没有响应超过18 bits 0 = 没有超时 该位只读
[2]	RESUME	重启状态 1 = 从挂起状态重启 0 = 没有总线重启 该位只读
[1]	SUSPEND	挂起状态 1 = 总线空闲超过 3ms, 或线缆拔出或主机休眠 0 = 总线没有挂起 该位只读
[0]	USBRST	USB 复位状态 1 = SE0 (single-ended 0)超过2.5us 总线复位 0 = 总线无复位 该位只读

悬空检测寄存器 (USB_FLDET)

寄存器	偏移量	R/W	描述	复位后的值
USB_FLDET	USB_BA+0x014	R	悬空检测寄存器	0x0000_0000

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

Bits	描述	
[31:1]	保留	保留
[0]	FLDET	设备悬空检测 1 = USB控制器接入总线时，该位置1 0 = USB控制器没有接入USB HOST

缓存偏移寄存器 (USB_BUFSEG)

仅用于Setup token.

寄存器	偏移量	R/W	描述	复位后的值
USB_BUFSEG	USB_BA+0x018	R/W	缓冲分割寄存器	0x0000_0000

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

Bits	描述	
[31:9]	保留	保留
[8:3]	BUFSEG	Setup token有效的开始地址是： USB_SRAM 地址 + { BUFSEG[8:3], 3'b000 } USB_SRAM 地址 = USB_BA + 0x100h. Note: It is used for Setup token only.
[2:0]	保留	保留

缓存偏移寄存器 (BUFSEGx) x = 0~5

寄存器	偏移量	R/W	描述	复位后的值
USB_BUFSEG0	USB_BA+0x020	R/W	端点 0 的缓冲分割	0x0000_0000
USB_BUFSEG1	USB_BA+0x030	R/W	端点 1 的缓冲分割	0x0000_0000
USB_BUFSEG2	USB_BA+0x040	R/W	端点 2 的缓冲分割	0x0000_0000
USB_BUFSEG3	USB_BA+0x050	R/W	端点 3 的缓冲分割	0x0000_0000
USB_BUFSEG4	USB_BA+0x060	R/W	端点 4 的缓冲分割	0x0000_0000
USB_BUFSEG5	USB_BA+0x070	R/W	端点 5 的缓冲分割	0x0000_0000

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

Bits	描述	
[31:9]	保留	保留
[8:3]	BUFSEGx	端点有效的开始地址是： USB_SRAM 地址 + { BUFSEG[8:3], 3'b000 } USB_SRAM 地址 = USB_BA + 0x100h. 参考5.4.4.7 端点的SRAM结构及其说明.
[2:0]	保留	保留

最大有效载荷寄存器 (USB_MXPLDx) x = 0~5

寄存器	偏移量	R/W	描述	复位后的值
USB_MXPLD0	USB_BA+0x024	R/W	端点 0 最大有效载荷	0x0000_0000
USB_MXPLD1	USB_BA+0x034	R/W	端点 1 最大有效载荷	0x0000_0000
USB_MXPLD2	USB_BA+0x044	R/W	端点 2 最大有效载荷	0x0000_0000
USB_MXPLD3	USB_BA+0x054	R/W	端点 3 最大有效载荷	0x0000_0000
USB_MXPLD4	USB_BA+0x064	R/W	端点 4 最大有效载荷	0x0000_0000
USB_MXPLD5	USB_BA+0x074	R/W	端点 5 最大有效载荷	0x0000_0000

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

Bits	描述	
[31:9]	保留	保留
[8:0]	MXPLD	<p>最大有效载荷 用于定义发送到主机(IN token)或从主机接收到(OUT token)的数据长度. 也表示端点准备发送或接收数据. (1). CPU写该寄存器, IN token, MXPLD 的值用于定义要发送的数据长度并表示数据缓冲已经准备好. OUT token, 表示控制准备接收主机的数据, MXPLD 的值表示从主机发送过来的最大数据长度. (2). CPU读该寄存器, IN token, MXPLD 的值 表示发送到主机的数据长度 OUT token, MXPLD 的值表示从主机接收到的实际数据长度. 一旦MXPLD 被写, 数据包将在IN/OUT token到达后立即发送/接收.</p>

配置寄存器 (USB_CFGx) x = 0~5

寄存器	偏移量	R/W	描述	复位后的值
USB_CFG0	USB_BA+0x028	R/W	端点 0 的配置	0x0000_0000
USB_CFG1	USB_BA+0x038	R/W	端点 1 的配置	0x0000_0000
USB_CFG2	USB_BA+0x048	R/W	端点 2 的配置	0x0000_0000
USB_CFG3	USB_BA+0x058	R/W	端点 3 的配置	0x0000_0000
USB_CFG4	USB_BA+0x068	R/W	端点 4 的配置	0x0000_0000
USB_CFG5	USB_BA+0x078	R/W	端点 5 的配置	0x0000_0000

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

Bits	描述	
[31:10]	保留	保留
[9]	CSTALL	清 STALL 响应 1 = 在setup阶段允许自动清stall 0 = 禁止自动清 stall
[8]	保留	保留
[7]	DSQ_SYNC	数据时序同步 1 = DATA1 PID 0 = DATA0 PID 在IN token的传输中指明DATA0 或 DATA1 PID. H/W 基于该位自动触发.
[6:5]	STATE	端点状态 00 = 禁止端点 01 = 输出端点

		10 = 输入端点 11 = 无定义
[4]	ISOCH	Isochronous 端点 该位设置端点为Isochronous 端点, 无握手. 1 = Isochronous 端点 0 = 非Isochronous 端点
[3:0]	EP_NUM	端点号 用于定义当前端点的端点号

额外配置寄存器 (USB_CFGPx) x = 0~5

寄存器	偏移量	R/W	描述	复位后的值
USB_CFGP0	USB_BA+0x02C	R/W	设置端点 0 自动回 STALL 与清除 In/Out 准备好控制寄存器	0x0000_0000
USB_CFGP1	USB_BA+0x03C	R/W	设置端点 1 自动回 STALL 与清除 In/Out 准备好控制寄存器	0x0000_0000
USB_CFGP2	USB_BA+0x04C	R/W	设置端点 2 自动回 STALL 与清除 In/Out 准备好控制寄存器	0x0000_0000
USB_CFGP3	USB_BA+0x05C	R/W	设置端点 3 自动回 STALL 与清除 In/Out 准备好控制寄存器	0x0000_0000
USB_CFGP4	USB_BA+0x06C	R/W	设置端点 4 自动回 STALL 与清除 In/Out 准备好控制寄存器	0x0000_0000
USB_CFGP5	USB_BA+0x07C	R/W	设置端点 5 自动回 STALL 与清除 In/Out 准备好控制寄存器	0x0000_0000

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

Bits	描述	
[31:2]	保留	保留
[1]	SSTALL	<p>设置 STALL</p> <p>1 = 设置设备自动响应STALL 0 = 禁止设备响应STALL</p>
[0]	CLRRDY	<p>清除准备</p> <p>用户设置寄存器MXPLD, 表示端点准备好发送或接收数据. 如果用户想要清除准备好标志, 用户可以设置该位为1, 该位自动清0.</p> <p>IN token, 写1清IN token时发送数据到USB的准备信号.</p> <p>OUT token, 写1清OUT token时从USB接收数据的准备信号.</p>

		这位只能写入1，且其返回值始终为0。
--	--	--------------------

驱动 SE0 寄存器 (USB_DRVSE0)

寄存器	偏移量	R/W	描述	复位后的值
USB_DRVSE0	USB_BA+0x090	R/W	USB PHY 驱动 SE0	0x0000_0001

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留							DRVSE0

Bits	描述	
[31:1]	保留	保留
[0]	DRVSE0	<p>USB 总线驱动 SE0 在 USB_DP and USB_DM 都拉低时，为SE0 1 = USB PHY驱动SE0 0 = 无</p>

5.5 通用 I/O (GPIO)

5.5.1 概述和特征

NuMicro™ NUC122 有 41 个通用 I/O 引脚，可以和其他功能引脚共享，分配在GPIOA, GPIOB, GPIOC, 与 GPIOD 四个端口上，每个端口最多 16 个引脚。每个引脚都是独立的，都有相应的寄存器位来控制引脚功能模式与数据。

I/O 类型可独立地配置为输入，输出，开漏或准双端模式。复位之后，所有引脚的 I/O 引脚类型均为准双向模式，端口数据寄存器 $\text{GPIO}_x\text{-DOUT}[15:0]$ 的值为 0x000_FFFF。VDD 从 5.0 V 到 2.5 V 时，I/O 口上拉阻值为 $110\text{K}\Omega \sim 300\text{K}\Omega$ 。

5.5.2 功能描述

5.5.2.1 输入模式说明

设置 $\text{GPIO}_x\text{-PMD}$ ($\text{PMD}_n[1:0]$) 为 00b， GPIO_x port [n] 为输入模式，I/O 引脚为三态（高阻），没有输出驱动能力。 $\text{GPIO}_x\text{-PIN}$ 的值反映相应端口的状态。

5.5.2.2 输出模式的说明

设置 $\text{GPIO}_x\text{-PMD}$ ($\text{PMD}_n[1:0]$) 为 01b， GPIO_x port [n] 为输出模式，I/O 支持数字输出功能，有 source/sink 电流能力。 $\text{GPIO}_x\text{-DOUT}$ 相应位的值被送到相应引脚上。

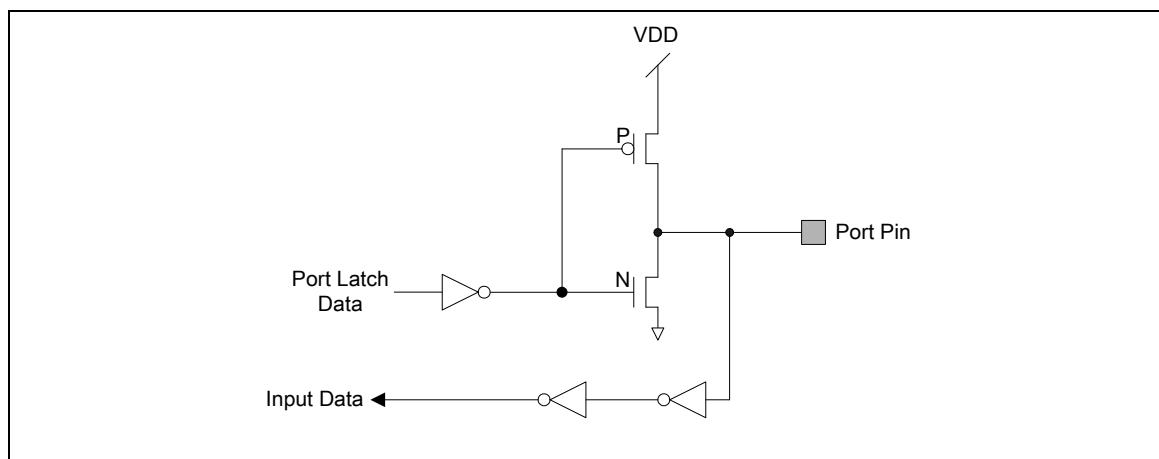


图 5-12 推挽输出

5.5.2.3 开漏模式的说明

设置 GPIOx_PMD (PMDn[1:0]) 为 10b, GPIOx port [n] 为开漏模式且 I/O 引脚数字输出功能仅支持灌电流，需要一个外加上拉接电阻用于驱动到高电平。如果 GPIOx_DOUT 相应位 bit [n] 的值为“0”，引脚上输出低。如果 GPIOx_DOUT 相应位 bit [n] 的值为“1”，该引脚输出为高，可以由外部上拉电阻控制。

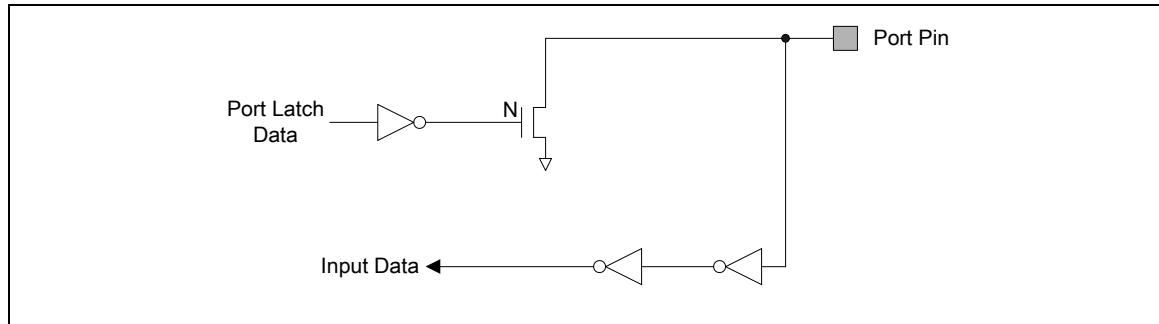


图 5-13 开漏输出

5.5.2.4 准双端模式的说明

设置 GPIOx_PMD (PMDn[1:0]) 为 11b, GPIOx port [n] 为准双端模式，I/O 同时支持数字输出和输入功能，但 source 电流仅达数百 uA。要实现数字输入，需要先将 GPIOx_DOUT 相应位置 1。准双端输出是 80C51 及其派生产品所共有的模式。若 GPIOx_DOUT 相应位 bit[n] 为“0”，引脚上输出为“低”。若 GPIOx_DOUT 相应位 bit[n] 为“1”，该引脚将检测引脚值。若引脚值为高，没有任何动作，若引脚值为低，该引脚在 2 个时钟周期内强制置高，然后禁止强输出驱动，引脚状态由内部上拉电阻控制。注：准双端模式的电流大小仅有 200 uA 到 30 uA(VDD 的电压从 5.0 V 到 2.5 V)。

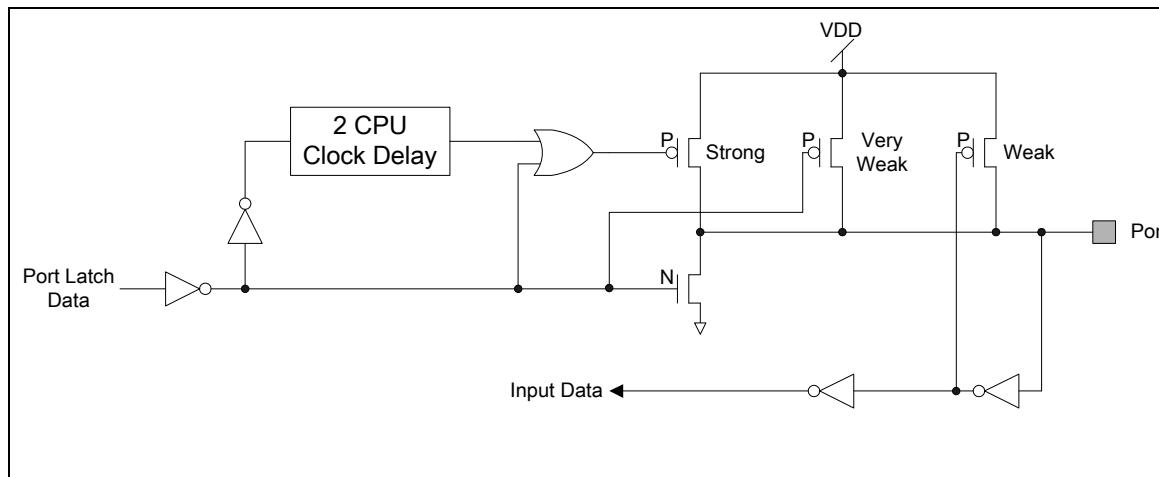


图 5-14 准双端 I/O 模式

5.5.3 寄存器映射

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



寄存器	偏移量	R/W	描述	复位后的值
GP_BA = 0x5000_4000				
GPIOA_PMD	GP_BA+0x000	R/W	GPIO 端口 A Bit 模式控制	0xFFFF_FFFF
GPIOA_OFFD	GP_BA+0x004	R/W	GPIO 端口 A Bit OFF 数字使能	0x0000_0000
GPIOA_DOUT	GP_BA+0x008	R/W	GPIO 端口 A 数据输出值	0x0000_FFFF
GPIOA_DMASK	GP_BA+0x00C	R/W	GPIO 端口 A 数据输出写屏蔽	0x0000_0000
GPIOA_PIN	GP_BA+0x010	R	GPIO 端口 A 管脚数值	0x0000_XXXX
GPIOA_DBEN	GP_BA+0x014	R/W	GPIO 端口 A 去抖动使能	0x0000_0000
GPIOA_IMD	GP_BA+0x018	R/W	GPIO 端口 A 中断模式控制	0x0000_0000
GPIOA_IEN	GP_BA+0x01C	R/W	GPIO 端口 A 中断使能	0x0000_0000
GPIOA_ISRC	GP_BA+0x020	R/W	GPIO 端口 A 中断源标志	0xXXXX_XXXX
GPIOB_PMD	GP_BA+0x040	R/W	GPIO 端口 B Bit 模式使能	0xFFFF_FFFF
GPIOB_OFFD	GP_BA+0x044	R/W	GPIO 端口 B Bit OFF 数字使能	0x0000_0000
GPIOB_DOUT	GP_BA+0x048	R/W	GPIO 端口 B 数据输出值	0x0000_FFFF
GPIOB_DMASK	GP_BA+0x04C	R/W	GPIO 端口 B 数据输出写屏蔽	0x0000_0000
GPIOB_PIN	GP_BA+0x050	R	GPIO 端口 B 管脚数值	0x0000_XXXX
GPIOB_DBEN	GP_BA+0x054	R/W	GPIO 端口 B 去抖动使能	0x0000_0000
GPIOB_IMD	GP_BA+0x058	R/W	GPIO 端口 B 中断模式控制	0x0000_0000
GPIOB_IEN	GP_BA+0x05C	R/W	GPIO 端口 B 中断使能	0x0000_0000
GPIOB_ISRC	GP_BA+0x060	R/W	GPIO 端口 B 中断源标志	0xXXXX_XXXX
GPIOC_PMD	GP_BA+0x080	R/W	GPIO 端口 C Bit 模式使能	0xFFFF_FFFF
GPIOC_OFFD	GP_BA+0x084	R/W	GPIO 端口 C Bit OFF 数字使能	0x0000_0000
GPIOC_DOUT	GP_BA+0x088	R/W	GPIO 端口 C 数据输出值	0x0000_FFFF
GPIOC_DMASK	GP_BA+0x08C	R/W	GPIO 端口 C 数据输出写屏蔽	0x0000_0000
GPIOC_PIN	GP_BA+0x090	R	GPIO 端口 C 管脚数值	0x0000_XXXX
GPIOC_DBEN	GP_BA+0x094	R/W	GPIO 端口 C 去抖动使能	0x0000_0000
GPIOC_IMD	GP_BA+0x098	R/W	GPIO 端口 C 中断模式控制	0x0000_0000
GPIOC_IEN	GP_BA+0x09C	R/W	GPIO 端口 C 中断使能	0x0000_0000
GPIOC_ISRC	GP_BA+0xA0	R/W	GPIO 端口 C 中断源标志	0xXXXX_XXXX

GPIOD_PMD	GP_BA+0x0C0	R/W	GPIO 端口 D Bit 模式使能	0xFFFF_FFFF
GPIOD_OFFD	GP_BA+0x0C4	R/W	GPIO 端口 D Bit OFF 数字使能	0x0000_0000
GPIOD_DOUT	GP_BA+0x0C8	R/W	GPIO 端口 D 数据输出值	0x0000_FFFF
GPIOD_DMASK	GP_BA+0x0CC	R/W	GPIO 端口 D 数据输出写屏蔽	0x0000_0000
GPIOD_PIN	GP_BA+0x0D0	R	GPIO 端口 D 管脚数值	0x0000_XXXX
GPIOD_DBEN	GP_BA+0x0D4	R/W	GPIO 端口 D 去抖动使能	0x0000_0000
GPIOD_IMD	GP_BA+0x0D8	R/W	GPIO 端口 D 中断模式控制	0x0000_0000
GPIOD_IEN	GP_BA+0x0DC	R/W	GPIO 端口 D 中断使能	0x0000_0000
GPIOD_ISRC	GP_BA+0x0E0	R/W	GPIO 端口 D 中断源标志	0xXXXX_XXXX
DBNCECON	GP_BA+0x180	R/W	去抖动循环控制	0x0000_0020
GPIOA0_DOUT	GP_BA+0x200	R/W	GPIO PA.0 数据输出/输入值	0x0000_0001
GPIOA1_DOUT	GP_BA+0x204	R/W	GPIO PA.1 数据输出/输入值	0x0000_0001
GPIOA2_DOUT	GP_BA+0x208	R/W	GPIO PA.2 数据输出/输入值	0x0000_0001
GPIOA3_DOUT	GP_BA+0x20C	R/W	GPIO PA.3 数据输出/输入值	0x0000_0001
GPIOA4_DOUT	GP_BA+0x210	R/W	GPIO PA.4 数据输出/输入值	0x0000_0001
GPIOA5_DOUT	GP_BA+0x214	R/W	GPIO PA.5 数据输出/输入值	0x0000_0001
GPIOA6_DOUT	GP_BA+0x218	R/W	GPIO PA.6 数据输出/输入值	0x0000_0001
GPIOA7_DOUT	GP_BA+0x21C	R/W	GPIO PA.7 数据输出/输入值	0x0000_0001
GPIOA8_DOUT	GP_BA+0x220	R/W	GPIO PA.8 数据输出/输入值	0x0000_0001
GPIOA9_DOUT	GP_BA+0x224	R/W	GPIO PA.9 数据输出/输入值	0x0000_0001
GPIOA10_DOUT	GP_BA+0x228	R/W	GPIO PA.10 数据输出/输入值	0x0000_0001
GPIOA11_DOUT	GP_BA+0x22C	R/W	GPIO PA.11 数据输出/输入值	0x0000_0001
GPIOA12_DOUT	GP_BA+0x230	R/W	GPIO PA.12 数据输出/输入值	0x0000_0001
GPIOA13_DOUT	GP_BA+0x234	R/W	GPIO PA.13 数据输出/输入值	0x0000_0001
GPIOA14_DOUT	GP_BA+0x238	R/W	GPIO PA.14 数据输出/输入值	0x0000_0001
GPIOA15_DOUT	GP_BA+0x23C	R/W	GPIO PA.15 数据输出/输入值	0x0000_0001
GPIOB0_DOUT	GP_BA+0x240	R/W	GPIO PB.0 数据输出/输入值	0x0000_0001
GPIOB1_DOUT	GP_BA+0x244	R/W	GPIO PB.1 数据输出/输入值	0x0000_0001
GPIOB2_DOUT	GP_BA+0x248	R/W	GPIO PB.2 数据输出/输入值	0x0000_0001

GPIOB3_DOUT	GP_BA+0x24C	R/W	GPIO PB.3 数据输出/输入值	0x0000_0001
GPIOB4_DOUT	GP_BA+0x250	R/W	GPIO PB.4 数据输出/输入值	0x0000_0001
GPIOB5_DOUT	GP_BA+0x254	R/W	GPIO PB.5 数据输出/输入值	0x0000_0001
GPIOB6_DOUT	GP_BA+0x258	R/W	GPIO PB.6 数据输出/输入值	0x0000_0001
GPIOB7_DOUT	GP_BA+0x25C	R/W	GPIO PB.7 数据输出/输入值	0x0000_0001
GPIOB8_DOUT	GP_BA+0x260	R/W	GPIO PB.8 数据输出/输入值	0x0000_0001
GPIOB9_DOUT	GP_BA+0x264	R/W	GPIO PB.9 数据输出/输入值	0x0000_0001
GPIOB10_DOUT	GP_BA+0x268	R/W	GPIO PB.10 数据输出/输入值	0x0000_0001
GPIOB11_DOUT	GP_BA+0x26C	R/W	GPIO PB.11 数据输出/输入值	0x0000_0001
GPIOB12_DOUT	GP_BA+0x270	R/W	GPIO PB.12 数据输出/输入值	0x0000_0001
GPIOB13_DOUT	GP_BA+0x274	R/W	GPIO PB.13 数据输出/输入值	0x0000_0001
GPIOB14_DOUT	GP_BA+0x278	R/W	GPIO PB.14 数据输出/输入值	0x0000_0001
GPIOB15_DOUT	GP_BA+0x27C	R/W	GPIO PB.15 数据输出/输入值	0x0000_0001
GPIOC0_DOUT	GP_BA+0x280	R/W	GPIO PC.0 数据输出/输入值	0x0000_0001
GPIOC1_DOUT	GP_BA+0x284	R/W	GPIO PC.1 数据输出/输入值	0x0000_0001
GPIOC2_DOUT	GP_BA+0x288	R/W	GPIO PC.2 数据输出/输入值	0x0000_0001
GPIOC3_DOUT	GP_BA+0x28C	R/W	GPIO PC.3 数据输出/输入值	0x0000_0001
GPIOC4_DOUT	GP_BA+0x290	R/W	GPIO PC.4 数据输出/输入值	0x0000_0001
GPIOC5_DOUT	GP_BA+0x294	R/W	GPIO PC.5 数据输出/输入值	0x0000_0001
GPIOC6_DOUT	GP_BA+0x298	R/W	GPIO PC.6 数据输出/输入值	0x0000_0001
GPIOC7_DOUT	GP_BA+0x29C	R/W	GPIO PC.7 数据输出/输入值	0x0000_0001
GPIOC8_DOUT	GP_BA+0x2A0	R/W	GPIO PC.8 数据输出/输入值	0x0000_0001
GPIOC9_DOUT	GP_BA+0x2A4	R/W	GPIO PC.9 数据输出/输入值	0x0000_0001
GPIOC10_DOUT	GP_BA+0x2A8	R/W	GPIO PC.10 数据输出/输入值	0x0000_0001
GPIOC11_DOUT	GP_BA+0x2AC	R/W	GPIO PC.11 数据输出/输入值	0x0000_0001
GPIOC12_DOUT	GP_BA+0x2B0	R/W	GPIO PC.12 数据输出/输入值	0x0000_0001
GPIOC13_DOUT	GP_BA+0x2B4	R/W	GPIO PC.13 数据输出/输入值	0x0000_0001
GPIOC14_DOUT	GP_BA+0x2B8	R/W	GPIO PC.14 数据输出/输入值	0x0000_0001
GPIOC15_DOUT	GP_BA+0x2BC	R/W	GPIO PC.15 数据输出/输入值	0x0000_0001

GPIOD0_DOUT	GP_BA+0x2C0	R/W	GPIO PD.0 数据输出/输入值	0x0000_0001
GPIOD1_DOUT	GP_BA+0x2C4	R/W	GPIO PD.1 数据输出/输入值	0x0000_0001
GPIOD2_DOUT	GP_BA+0x2C8	R/W	GPIO PD.2 数据输出/输入值	0x0000_0001
GPIOD3_DOUT	GP_BA+0x2CC	R/W	GPIO PD.3 数据输出/输入值	0x0000_0001
GPIOD4_DOUT	GP_BA+0x2D0	R/W	GPIO PD.4 数据输出/输入值	0x0000_0001
GPIOD5_DOUT	GP_BA+0x2D4	R/W	GPIO PD.5 数据输出/输入值	0x0000_0001
GPIOD6_DOUT	GP_BA+0x2D8	R/W	GPIO PD.6 数据输出/输入值	0x0000_0001
GPIOD7_DOUT	GP_BA+0x2DC	R/W	GPIO PD.7 数据输出/输入值	0x0000_0001
GPIOD8_DOUT	GP_BA+0x2E0	R/W	GPIO PD.8 数据输出/输入值	0x0000_0001
GPIOD9_DOUT	GP_BA+0x2E4	R/W	GPIO PD.9 数据输出/输入值	0x0000_0001
GPIOD10_DOUT	GP_BA+0x2E8	R/W	GPIO PD.10 数据输出/输入值	0x0000_0001
GPIOD11_DOUT	GP_BA+0x2EC	R/W	GPIO PD.11 数据输出/输入值	0x0000_0001
GPIOD12_DOUT	GP_BA+0x2F0	R/W	GPIO PD.12 数据输出/输入值	0x0000_0001
GPIOD13_DOUT	GP_BA+0x2F4	R/W	GPIO PD.13 数据输出/输入值	0x0000_0001
GPIOD14_DOUT	GP_BA+0x2F8	R/W	GPIO PD.14 数据输出/输入值	0x0000_0001
GPIOD15_DOUT	GP_BA+0x2FC	R/W	GPIO PD.15 数据输出/输入值	0x0000_0001

5.5.4 寄存器描述

GPIO 端口 [A/B/C/D] I/O 模式控制 (GPIOx_PMD)

寄存器	偏移量	R/W	描述	复位后的值
GPIOA_PMD	GP_BA+0x000	R/W	GPIO 端口 A Pin I/O 模式控制	0xFFFF_FFFF
GPIOB_PMD	GP_BA+0x040	R/W	GPIO 端口 B Pin I/O 模式控制	0xFFFF_FFFF
GPIOC_PMD	GP_BA+0x080	R/W	GPIO 端口 C Pin I/O 模式控制	0xFFFF_FFFF
GPIOD_PMD	GP_BA+0x0C0	R/W	GPIO 端口 D Pin I/O 模式控制	0xFFFF_FFFF

31	30	29	28	27	26	25	24
PMD15		PMD14		PMD13		PMD12	
23	22	21	20	19	18	17	16
PMD11		PMD10		PMD9		PMD8	
15	14	13	12	11	10	9	8
PMD7		PMD6		PMD5		PMD4	
7	6	5	4	3	2	1	0
PMD3		PMD2		PMD1		PMD0	

Bits	描述	
[2n+1:2n]	PMDn	GPIOx I/O Pin[n] 模式控制 决定GPIOx的I/O 类型。 00 = GPIO port [n] 引脚为输入模式 01 = GPIO port [n] 引脚为输出模式 10 = GPIO port [n] 引脚为开漏模式 11 = GPIO port [n] 引脚为准双向模式

GPIO 端口 [A/B/C/D] Bit OFF 数字寄存器使能 (GPIOx_OFFD)

寄存器	偏移量	R/W	描述	复位后的值
GPIOA_OFFD	GP_BA+0x004	R/W	GPIO 端口 A Bit OFF 数字使能	0x0000_0000
GPIOB_OFFD	GP_BA+0x044	R/W	GPIO 端口 B Bit OFF 数字使能	0x0000_0000
GPIOC_OFFD	GP_BA+0x084	R/W	GPIO 端口 C Bit OFF 数字使能	0x0000_0000
GPIOD_OFFD	GP_BA+0x0C4	R/W	GPIO 端口 D Bit OFF 数字使能	0x0000_0000

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

Bits	描述	
[31:16]	OFFD	GPIOx Pin[n] Bit OFF数字输入通道使能 用于控制GPIO的数字输入通道是否使能。如果输入为模拟信号，用户可以关闭输入通道防止漏电 1 = 禁止IO的数字输入通道 (数字输入拉低) 0 = 使能IO数据输入通道
[15:0]	保留	保留

GPIO 端口 [A/B/C/D] 数据输出值 (GPIOx_DOUT)

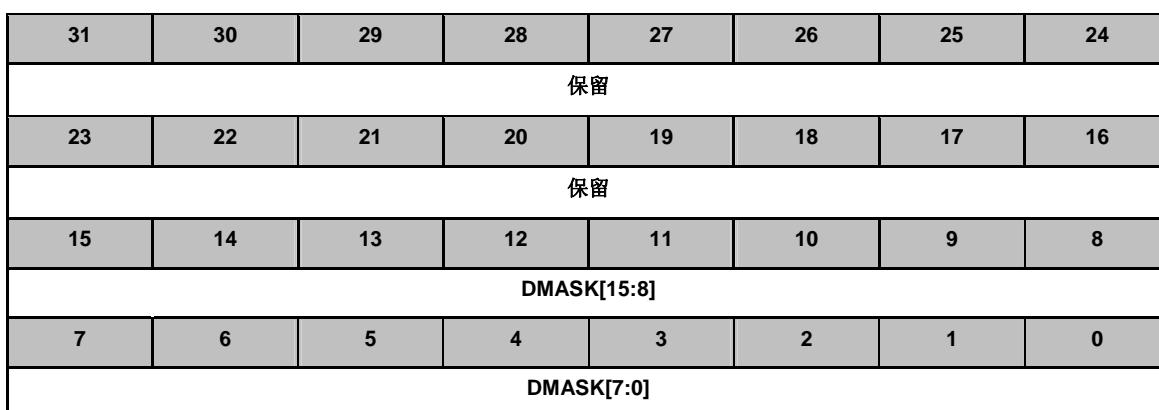
寄存器	偏移量	R/W	描述	复位后的值
GPIOA_DOUT	GP_BA+0x008	R/W	GPIO 端口 A 数据输出值	0x0000_FFFF
GPIOB_DOUT	GP_BA+0x048	R/W	GPIO 端口 B 数据输出值	0x0000_FFFF
GPIOC_DOUT	GP_BA+0x088	R/W	GPIO 端口 C 数据输出值	0x0000_FFFF
GPIOD_DOUT	GP_BA+0x0C8	R/W	GPIO 端口 D 数据输出值	0x0000_FFFF

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

Bits	描述	
[31:16]	保留	保留
[n]	DOUT[n]	GPIOx Pin[n] 输出值 在GPIO配置成输出，开漏和准双向模式时，控制GPIO的状态。 1 = GPIO配置成输出，开漏和准双向模式时，GPIO port [A/B/C/D] Pin[n] 为高 0 = GPIO配置成输出，开漏和准双向模式时，GPIO port [A/B/C/D] Pin[n] 为低

GPIO 端口 [A/B/C/D] 数据输出写屏蔽 (GPIOx_DMASK)

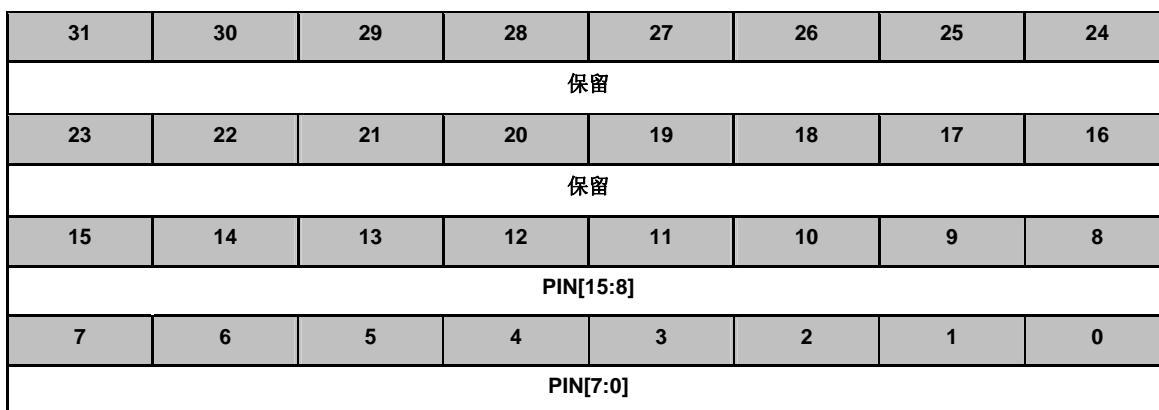
寄存器	偏移量	R/W	描述	复位后的值
GPIOA_DMASK	GP_BA+0x00C	R/W	GPIO 端口 A 数据输出写屏蔽	0xFFFF_0000
GPIOB_DMASK	GP_BA+0x04C	R/W	GPIO 端口 B 数据输出写屏蔽	0xFFFF_0000
GPIOC_DMASK	GP_BA+0x08C	R/W	GPIO 端口 C 数据输出写屏蔽	0xFFFF_0000
GPIOD_DMASK	GP_BA+0x0CC	R/W	GPIO 端口 D 数据输出写屏蔽	0xFFFF_0000



Bits	描述	
[31:16]	保留	保留
[n]	DMASK[n]	<p>端口 [A/B/C/D] 数据输出写屏蔽</p> <p>用于保护相应寄存器GPIOx_DOUT bit[n]. 在设置DMASK bit[n] 为“1”，相应DOUTn bit 被保护，写信号被屏蔽时，不能向保护位写数据</p> <p>1 = 保护相应的GPIO_DOUT[n] 位</p> <p>0 = 更新相应的GPIO_DOUT[n] 位</p>

GPIO 端口 [A/B/C/D] 管脚数据 (GPIOx_PIN)

寄存器	偏移量	R/W	描述	复位后的值
GPIOA_PIN	GP_BA+0x010	R	GPIO 端口 A 管脚数据	0x0000_XXXX
GPIOB_PIN	GP_BA+0x050	R	GPIO 端口 B 管脚数据	0x0000_XXXX
GPIOC_PIN	GP_BA+0x090	R	GPIO 端口 C 管脚数据	0x0000_XXXX
GPIOD_PIN	GP_BA+0x0D0	R	GPIO 端口 D 管脚数据	0x0000_XXXX



Bits	描述	
[31:16]	保留	保留
[n]	PIN[n]	端口 [A/B/C/D] 管脚数据 这些位的值为各个GPIO真实状态的反映

GPIO 端口 [A/B/C/D] 去抖动使能 (GPIOx_DBEN)

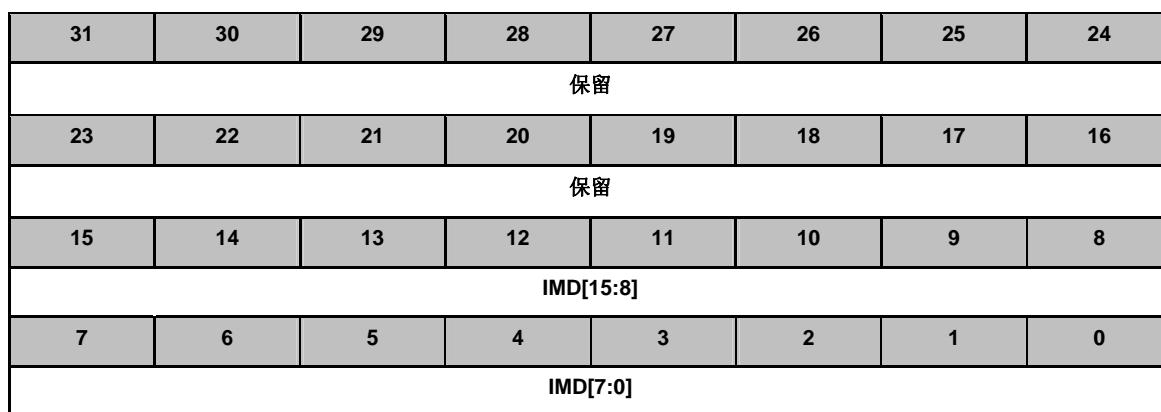
寄存器	偏移量	R/W	描述	复位后的值
GPIOA_DBEN	GP_BA+0x014	R/W	GPIO 端口 A 去抖动使能	0xFFFF_0000
GPIOB_DBEN	GP_BA+0x054	R/W	GPIO 端口 B 去抖动使能	0xFFFF_0000
GPIOC_DBEN	GP_BA+0x094	R/W	GPIO 端口 C 去抖动使能	0xFFFF_0000
GPIOD_DBEN	GP_BA+0x0D4	R/W	GPIO 端口 D 去抖动使能	0xFFFF_0000

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

Bits	描述	
[31:16]	保留	保留
[n]	DBEN[n]	<p>端口 [A/B/C/D] 输入信号去抖动使能</p> <p>DBEN[n]用于使能相应位的去抖动功能。如果信号脉冲宽度不能被两个连续的去抖动采样周期所采样，则被视为信号反弹，从而不触发中断。</p> <p>DBEN[n] 仅用于边沿触发“edge-trigger”中断，不能电平“level trigger”触发中断</p> <p>0 = 禁用 bit[n] 去抖动功能 1 = 使能 bit[n] 去抖动功能</p> <p>去抖动功能对于边沿触发中断有效，对于电平触发中断模式，去抖动功能全能位不起作用。</p>

GPIO 端口 [A/B/C/D] 中断模式控制 (GPIOx_IMD)

寄存器	偏移量	R/W	描述	复位后的值
GPIOA_IMD	GP_BA+0x018	R/W	GPIO 端口 A 中断模式控制	0xFFFF_XXXX
GPIOB_IMD	GP_BA+0x058	R/W	GPIO 端口 B 中断模式控制	0xFFFF_XXXX
GPIOC_IMD	GP_BA+0x098	R/W	GPIO 端口 C 中断模式控制	0xFFFF_XXXX
GPIOD_IMD	GP_BA+0xD8	R/W	GPIO 端口 D 中断模式控制	0xFFFF_XXXX



Bits	描述	
[31:16]	保留	保留
[n]	IMD[n]	<p>端口 [A/B/C/D] 边沿或电平检测中断控制</p> <p>IMD[n] 用于控制电平触发或边沿触发中断。若为边沿触发中断，触发源可由去抖动控制，如果是电平触发中断，输入源由一个HCLK时钟采样并产生中断。</p> <p>0 = 边沿触发中断 1 = 电平触发中断</p> <p>如果设置引脚为电平触发模式，则在寄存器GPIOX_IEN中，只能设置一个电平(高电平或者低电平)；若既设置为电平触发，又设置为边沿触发，设置被忽略，不会产生中断。</p> <p>去抖动功能对于边沿触发中断有效，对于电平触发中断无效。</p>

GPIO 端口 [A/B/C/D] 中断使能控制 (GPIOx_IEN)

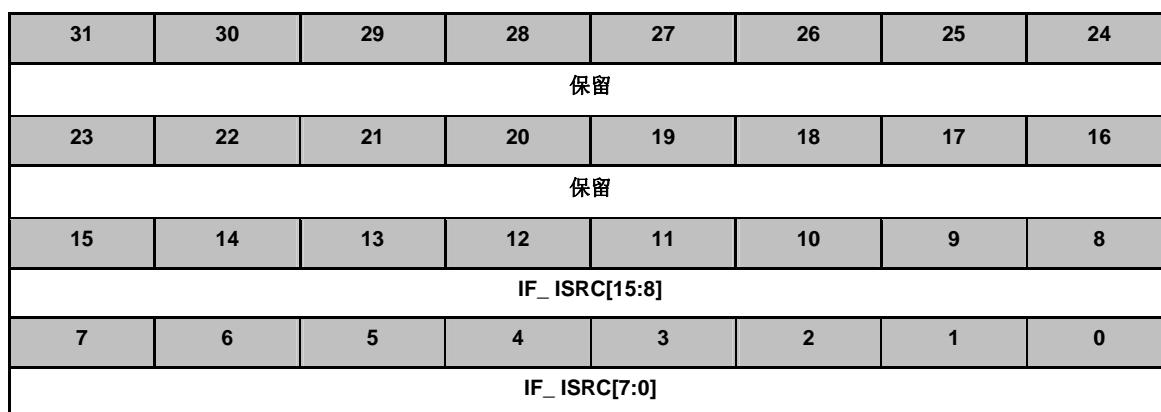
寄存器	偏移量	R/W	描述	复位后的值
GPIOA_IEN	GP_BA+0x01C	R/W	GPIO 端口 A 中断使能	0x0000_0000
GPIOB_IEN	GP_BA+0x05C	R/W	GPIO 端口 B 中断使能	0x0000_0000
GPIOC_IEN	GP_BA+0x09C	R/W	GPIO 端口 C 中断使能	0x0000_0000
GPIOD_IEN	GP_BA+0x0DC	R/W	GPIO 端口 D 中断使能	0x0000_0000

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

Bits	描述
[n+16]	<p>IR_EN[n] 使能端口 [A/B/C/D] 输入上升沿或输入高电平的中断 IR_EN[n] 用于使能相应GPIO_PIN[n]输入的中断. 置“1”也可以使能引脚唤醒功能 当设置 IR_EN[n] 位为“1”： 如果中断是电平触发模式，输入PIN[n]的状态为高电平时，产生中断。 如果中断是边沿触发模式，输入PIN[n]的状态由低电平到高电平变化时，产生中断。 1 = 使能PIN[n]高电平或由低电平到高电平变化的中断 0 = 禁用PIN[n]高电平或由低电平到高电平变化的中断</p>
[n]	<p>IF_EN[n] 使能端口 [A/B/C/D] 输入下降沿或输入低电平的中断 IF_EN[n] 用于使能相应GPIO_PIN[n]输入的中断. 置“1”也可以使能引脚唤醒功能 当设置 IF_EN[n] 位为“1”： 如果中断是电平触发模式，输入PIN[n]的状态为低电平时，产生中断。 如果中断是边沿触发模式，输入PIN[n]的状态由高电平到低电平变化时，产生中断。 1 = 使能PIN[n]低电平或由高电平到低电平变化的中断 0 = 禁用PIN[n]低电平或由高电平到低电平变化的中断</p>

GPIO 端口 [A/B/C/D] 中断触发源 (GPIOx_ISRC)

寄存器	偏移量	R/W	描述	复位后的值
GPIOA_ISRC	GP_BA+0x020	R/W	GPIO 端口 A 中断触发源指标	0x0000_0000
GPIOB_ISRC	GP_BA+0x060	R/W	GPIO 端口 B 中断触发源指标	0x0000_0000
GPIOC_ISRC	GP_BA+0x0A0	R/W	GPIO 端口 C 中断触发源指标	0x0000_0000
GPIOD_ISRC	GP_BA+0x0E0	R/W	GPIO 端口 D 中断触发源指标	0x0000_0000



Bits	描述	
[31:16]	保留	保留
[n]	ISRC[n]	<p>端口 [A/B/C/D] 中断触发源指标</p> <p>读：</p> <p>1 = GPIOx[n]产生中断 0 = GPIOx[n]没有中断</p> <p>写：</p> <p>1= 清相应的中断标志 0= 无动作</p>

中断去抖动周期控制 (DBNCECON)

寄存器	偏移量	R/W	描述	复位后的值
DBNCECON	GP_BA+0x180	R/W	外部中断去抖动控制	0x0000_0020

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

Bits	描述																	
[31:6]	保留	保留																
[5]	ICLK_ON	<p>中断时钟On模式 如果GPIO pin[n]中断功能被关闭，设置该位为“0”将关闭中断产生时钟电路 1 = 总是使能中断产生时钟电路 0 = 如果中断GPIOA/B/C/D[n]被禁止，关闭相应的时钟电路</p>																
[4]	DBCLKSRC	<p>去抖动计数器时钟源选择 1 = 去抖动计数器时钟源为内部 10 KHz 时钟 0 = 去抖动计数器时钟源为 HCLK</p>																
[3:0]	DBCLKSEL	<p>去抖动采样周期选择</p> <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>DBCLKSEL</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>采样中断信号输入, 每 1 clock 一次</td> </tr> <tr> <td>1</td> <td>采样中断信号输入, 每 2 clocks 一次</td> </tr> <tr> <td>2</td> <td>采样中断信号输入, 每 4 clocks 一次</td> </tr> <tr> <td>3</td> <td>采样中断信号输入, 每 8 clocks 一次</td> </tr> <tr> <td>4</td> <td>采样中断信号输入, 每 16 clocks 一次</td> </tr> <tr> <td>5</td> <td>采样中断信号输入, 每 32 clocks 一次</td> </tr> <tr> <td>6</td> <td>采样中断信号输入, 每 64 clocks 一次</td> </tr> </tbody> </table>	DBCLKSEL	Description	0	采样中断信号输入, 每 1 clock 一次	1	采样中断信号输入, 每 2 clocks 一次	2	采样中断信号输入, 每 4 clocks 一次	3	采样中断信号输入, 每 8 clocks 一次	4	采样中断信号输入, 每 16 clocks 一次	5	采样中断信号输入, 每 32 clocks 一次	6	采样中断信号输入, 每 64 clocks 一次
DBCLKSEL	Description																	
0	采样中断信号输入, 每 1 clock 一次																	
1	采样中断信号输入, 每 2 clocks 一次																	
2	采样中断信号输入, 每 4 clocks 一次																	
3	采样中断信号输入, 每 8 clocks 一次																	
4	采样中断信号输入, 每 16 clocks 一次																	
5	采样中断信号输入, 每 32 clocks 一次																	
6	采样中断信号输入, 每 64 clocks 一次																	

	7	采样中断信号输入, 每128 clocks一次	
	8	采样中断信号输入, 每256 clocks一次	
	9	采样中断信号输入, 每 2*256 clocks一次	
	10	采样中断信号输入, 每4*256clocks一次	
	11	采样中断信号输入, 每8*256 clocks一次	
	12	采样中断信号输入, 每16*256 clocks一次	
	13	采样中断信号输入, 每32*256 clocks一次	
	14	采样中断信号输入, 每64*256 clocks一次	
	15	采样中断信号输入, 每128*256 clocks一次	

GPIO 端口 [A/B/C/D] I/O 位输出/输入控制 (GPIOxx_DOUT)

寄存器	偏移量	R/W	描述	复位后的值
GPIOAx_DOUT	GP_BA+0x200 - GP_BA+0x23C	R/W	GPIO 端口 A Pin I/O 位输出/输入控制	0x0000_0001
GPIOBx_DOUT	GP_BA+0x240 - GP_BA+0x27C	R/W	GPIO 端口 B Pin I/O 位输出/输入控制	0x0000_0001
GPIOCx_DOUT	GP_BA+0x280 - GP_BA+0x2BC	R/W	GPIO 端口 C Pin I/O 位输出/输入控制	0x0000_0001
GPIO Dx_DOUT	GP_BA+0x2C0 - GP_BA+0x2FC	R/W	GPIO 端口 D Pin I/O 位输出/输入控制	0x0000_0001

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

Bits	描述	
[31:1]	保留	保留
[0]	GPIOxx_DOUT	<p>GPIOxx I/O Pin 位输出/输入控制 设置该位可以控制GPIO位的输出值 1 = 设置相应GPIO位为高 0 = 设置相应GPIO位为低 例如: 写GPIOA0_DOUT即把值写到GPIOA_DOUT[0]位上, 读GPIOA0_DOUT即读取GPIOA_PIN[0]的值.</p>

5.6 I²C总线控制器 (Master/Slave) (I²C)

5.6.1 概述

I²C为双线，双向串行总线，通过简单有效的连线方式实现器件间的数据交换。标准I²C是多主机总线，包括冲突检测和仲裁以防止在两个或多个主机试图同时控制总线时发生的数据损坏，串行，8位双向数据传输。

数据在主机与从机之间通过SCL时钟线控制在SDA数据线上实现一字节一字节的同步传输，每个字节为8位长度，一个SCL时钟脉冲传输一个数据位，数据由最高位MSB开始传输，每个传输字节后跟随一个应答位，每个位在SCL为高时采样；因此，SDA线只有在SCL为低时才可以改变，在SCL为高时SDA保持稳定。当SCL为高时，SDA线上的跳变视为命令中断（START 或 STOP）。参考下图 I²C 总线时序。

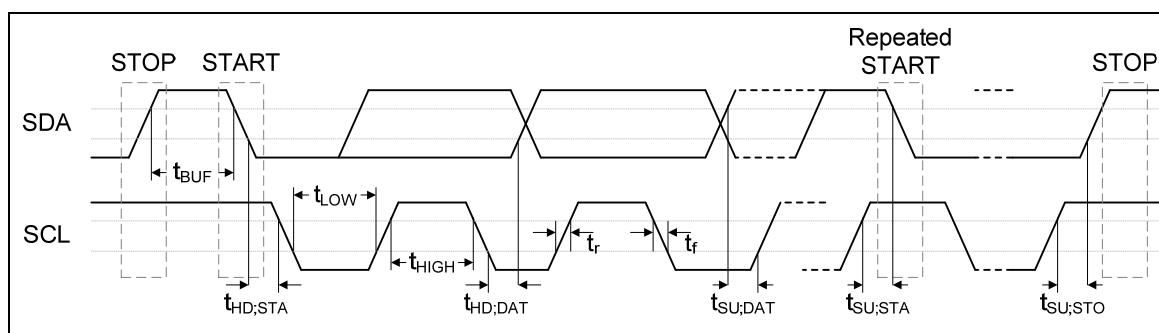


图 5-15 I²C 总线时序

片上I²C逻辑提供符合I²C总线标准的串连标准接口。I²C端口自动处理字节传输，将I2CON的ENS1位设置为1，即可使能该端口。I²C H/W 接口通过SDA与 SCL两个引脚连到I²C总线。用于I²C操作的两个引脚需要上拉电阻，因为这两个引脚为开漏脚。在I/O引脚作为I²C 端口使用时，用户必须预先设置引脚功能为I²C功能。

I²C总线通过SDA 及 SCL与连接在总线上的设备间传输数据，总线的主要特征：

- 支持主机模式或从机模式
- 主从机之间双向数据传输
- 多主机总线支持 (无中心主机)
- 总线上多主机间连续传输数据，实现无中断仲裁
- 总线上不同传输速率设备间，实现同步传输
- 总线上数据暂停传送及恢复传送时，实现同步时钟功能
- 内建14位溢出计数器，当I²C总线上闲置并计数器溢位，产生I²C中断.
- 需要外部上拉确保高电平输出
- 可编辑的时钟适用于不同速率控制
- 支持7位地址模式

- I²C总线上支持多地址辨识(4组从机地址带mask选项)

5.6.1.1 I²C 协议

通常标准I²C传输协议包含四个部分

- 1) 起始信号或重复起始信号
- 2) 从机地址传输和R/W位传输
- 3) 数据传输
- 4) 停止信号

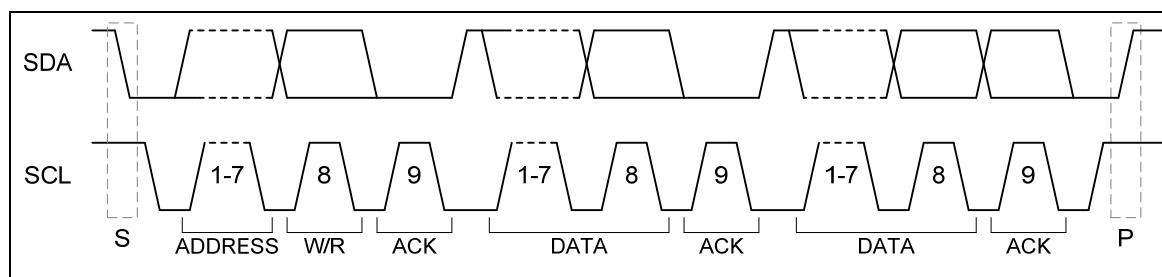


图 5-16 I²C 协议

5.6.1.2 I²C总线上数据传输

主机发出 从机接收7位地址(一个字节)

传输方向未改变

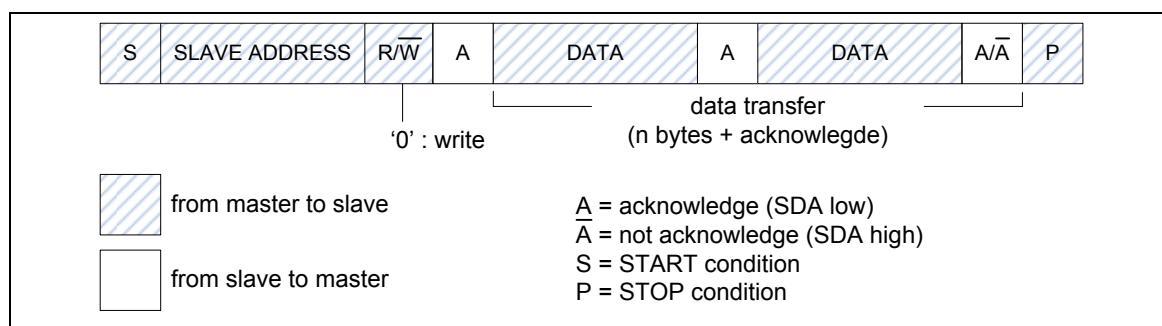


图 5-17 主机向从机传输数据

第一个字节后（从机地址）主机紧接着由从机读取数据

传输方向改变

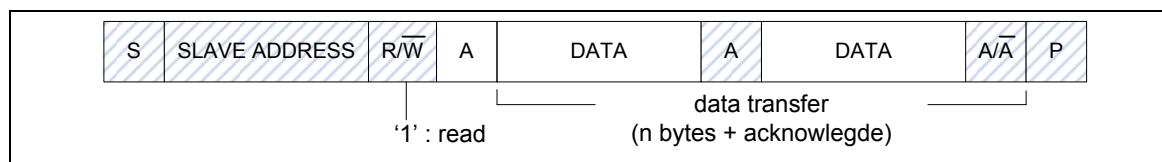


图 5-18 主机由从机读取地址

5.6.1.3 起始位或重复起始信号

当总线处于空闲状态下，说明没有主机对总线发起传输请求(SCL和SDA线同时为高)，主机可以通过发送一个START信号来发起传输请求。起始信号，通常表示为**S-bit**，当SCL线为高时，SDA线上信号由高至低，标示总线上产生起始信号，新的传输开始。

重复起始信号 (**Sr**) 即在两个START信号之间没有STOP信号。主机采用这种方法与另一个从机或相同的从机以不同传输方向进行通信(例如：从写入设备到从设备读出)而不释放总线。

STOP 信号

主机向总线发出停止信号结束数据传送。停止信号，通常用**P-bit**表示，当SCL线为高时，SDA线上出现由低到高的信号，被定义为停止信号。

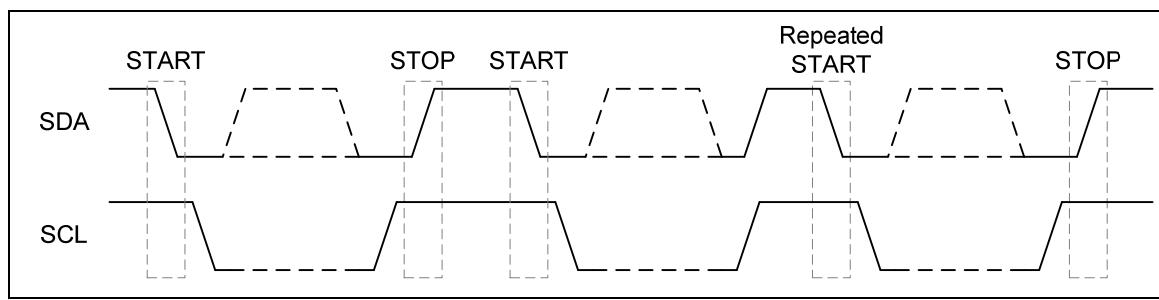


图 5-19 START 和 STOP 条件

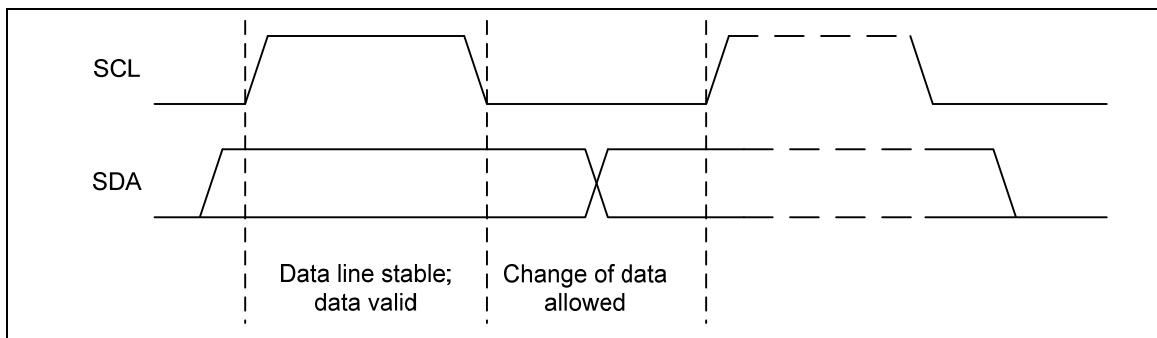
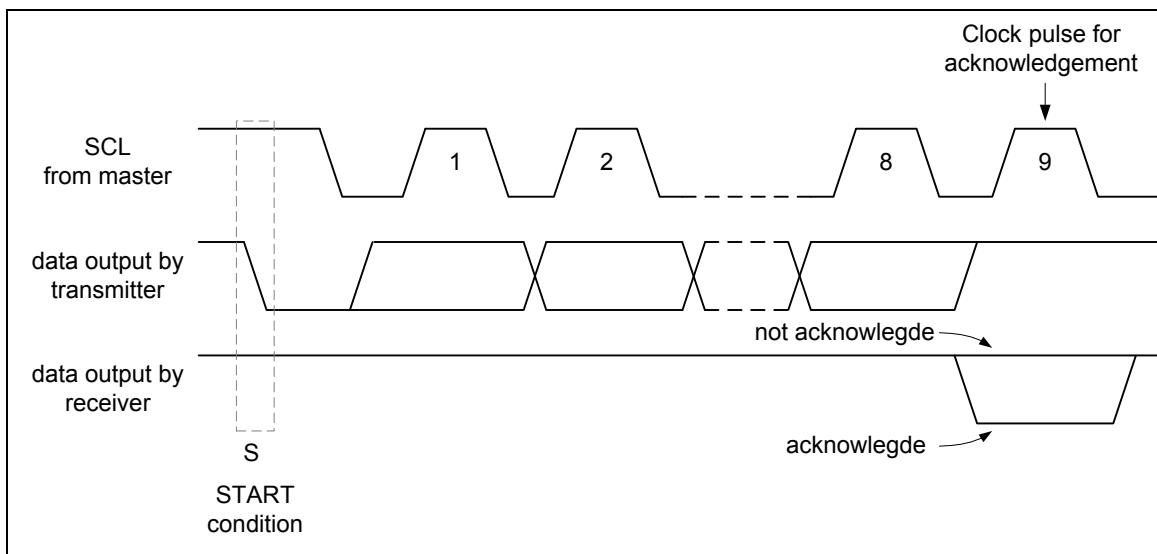
5.6.1.4 从机地址传输

START信号后马上传输的第一个字节就是从机地址。这是一个7位从设备地址加一个R/W位，R/W位控制从机的信号传输方向。系统中没有两个从机有相同的地址，只有被主机寻址的从机会通过在第9个SCL时钟周期将SDL置低电平作为应答。

5.6.1.5 数据传输

当从机地址被成功识别，就可以根据RW所决定的方向，开始一字节一字节的数据传输，每个传输字节最后带一个第9时钟周期的响应信号，如果从机上产生无响应信号(**NACK**)，主机可以产生停止信号来退出数据传输，或者产生重复起始信号开始新一轮的数据传输。

当主机作为接收器件时，发生无响应信号(**NACK**)，从机释放SDA线，使主机产生停止信号或重复起始信号。

图 5-20 I²C 总线上位传输图 5-21 I²C 总线上应答信号

5.6.2 协议寄存器

CPU 和 I²C 通过下列 13 个特殊功能寄存器通讯：I2CON（控制寄存器），I2CSTATUS（状态寄存器），I2CDAT（数据寄存器），I2CADDRn（地址寄存器，n=0~3），I2CADRMn（地址 mask 寄存器，n=0~3），I2CLK（时钟速率寄存器）和 I2CTOC（定时计数寄存器）。所有 I²C 的特殊功能寄存器的第 8 位至第 31 位都是保留的，不具备任何功能，返回值为 0。

当 ENS1(I2CON[6]) 置 1，I²C 口使能后，内部状态由 I2CON 和 I²C 逻辑硬件控制。当有新的状态码产生后，会存储到位字段 I2CSTATUS，I²C 中断标志 (SI(I2CON[3])) 也会自动置起。若此时 EI(I2CON[7]) 被设定为高，I²C 中断会被响应。I2CSTATUS[7:3] 内存储状态码，在 SI 由软件清除之前，I2CSTATUS 最低三位始终保持低电平。寄存器地址为 0x4002_0000 和 0x4012_0000。

5.6.2.1 地址寄存器(I2CADDR)

I²C 口内建4个从机地址寄存器I2CADDRn(n=0~3)。当I²C作为主机时，这四个寄存器的内容不相关。在从机模式下，位字段I2CADDRn[7:1] 必须装载芯片自己的从机地址。当I2CADDR 地址与接收的从机地址符合时，I²C硬件起作用。

I²C口支持“广播呼叫”功能。当GC位(I2CADDRn [0])被置起，I²C端口硬件将应答广播呼叫的地址，清GC位可禁止“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，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总线控制器有4个地址掩码寄存器I2CADRMn(n=0~3)支持多地址识别。当地址掩码寄存器置1，表示可以忽略接收到的相应地址位。当该位保持0，说明接收到的响应地址完全符合地址寄存器内的值。

5.6.2.2 数据寄存器 (I2CDAT)

该寄存器包含一个准备发送或刚接收到的一个字节的数据。只要不在移位处理的过程，CPU可以直接读写访问I2CDAT[7:0]。当I²C处于一个确定的状态并且串行中断标志(SI)被设置，I2CDAT[7:0]中的数据就一直是稳定的。在数据被移出的过程中，总线上的数据同时被移入。I2CDAT的内容一直是总线上出现的最后一个字节。因此，在仲裁失败时，主发射机到从接收机传输的模式下，I2CDAT[7:0]中的数据仍保持正确。

移位寄存器包含 I2CDAT[7:0] 和应答位-9位，应答位由 I²C 的硬件控制，CPU 不能访问。I2CDAT[7:0] 中的串行数据和应答位在串行时钟SCL线的上升沿移出。当一个字节被移位到I2CDAT[7:0]后，I2CDAT[7:0]中的串行数据是可以使用的，应答位(ACK或NACK)在第9个时钟返回。串行数据在每一个下降沿(SCL时钟)从I2CDAT[7:0]移出输出，在每一个上升沿(SCL时钟)数据移进I2CDAT[7:0]。

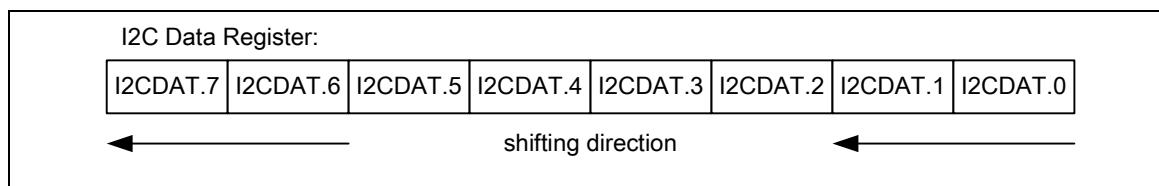


图 5-22 I²C 数据移位方向

5.6.2.3 控制寄存器(I2CON)

CPU 可以直接读或写寄存器I2CON [7:0]。两个受硬件影响的位是：在I²C硬件请求串行中断时SI置位；在STOP状态出现在总线上时清STO位。当ENS1 = "0"时，STO位也会被清除。

EI 使能中断。

ENS1 设置使能I²C串行功能控制器。当ENS1=1，使能I²C 串行功能。SDA与SCL的多功能引脚必须设置成 I²C功能。

STA I²C START 控制位。置1，STA为高进入主机模式。如果总线为空闲，发送一个STOP或repeat STOP状态到总线。

STO I²C STOP 控制位。I²C为主机模式时，STO位置'1'，设置STO来传送一个STOP状态到总线。

线, I²C硬件将检测总线状态, 如果一个STOP状态被检测到, 这个标志将被硬件自动清除。在从机模式, 设置STO复位I²C硬件来定义“无编址”从机模式, 这表示在从机接收模式不再从主机传送装置接收数据。

- SI I²C中断标志, 当一个新的I²C状态出现在寄存器I2CSTATUS, SI标志由硬件设置, 并且如果EI (I2CON [7])位被设置, 即产生I²C中断请求。SI标志由硬件置'1', SI必须由软件通过将该位置'1'来清'0'。所有的状态均在5.6.6节列出
- AA 应答控制位. 当AA=1 先于地址或数据的接收, 在以下两种情况: 1.) 从机正在应答主机发送的地址, 2.) 接收设备正在应答发送设备发送的数据, 在SCL线上的应答时钟脉冲期间将返回一个应答 (SDA上的低电平)

5.6.2.4 状态寄存器(I2CSTATUS)

I2CSTATUS [7:0] 是一个8-位只读寄存器。低3位一直为0。I2CSTATUS [7:3]是状态码。有26个可能的状态码, 当 I2CSTATUS[7:0] 的内容是 F8H, 没有串行中断请求。所有的其它 I2CSTATUS[7:3]值对应I²C 端口的状态。当每一个进入状态, 就会产生状态中断请求(SI = 1). 在SI 被硬件置'1' 或SI被软件清除之后1个机器周期, 有效状态码出现在I2CSTATUS[7:3]中.

另外, 00H状态表示总线错误。要从总线错误中恢复I²C, 应该设置STO, 清除SI以进入无编址模式。然后清除STO, 释放总线, 等待新的通信。当总线产生错误, I²C总线不能识别停止状态。总线错误发生在START或STOP条件出现在帧结构不正确的位置。不正确的位比如是在串行传输地址字节、数据字节或应答位期间。

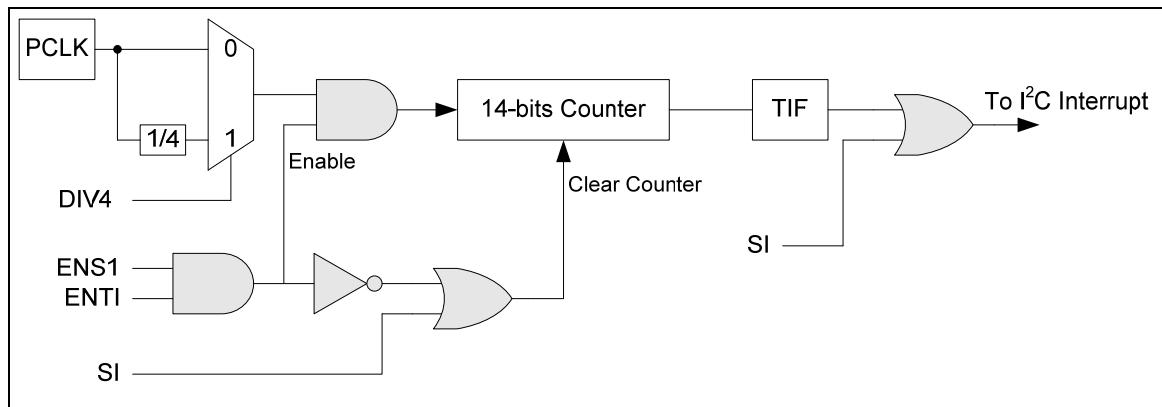
5.6.2.5 I²C时钟波特率位 (I2CLK)

当I²C在主机模式下, I²C数据的波特率由I2CLK[7:0]寄存器设定。I²C在从机模式下时是不重要的; 在从机模式下, I²C将自动与主机I²C设备时钟频率同步, 可高达1 MHz。

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

5.6.2.6 I²C超时计数寄存器(I2CTOC)

当总线被锁死时, MCU提供一个14位超时的计数器。当计数功能使能后, 计数器开始计数直至发生超时, TIF置1并要求MCU产生I²C中断或者清除ENT1为0关闭计数功能。当超时计数器使能, 对SI标志置高会使计数器复位, 再对SI清除为0会重新开始计数。如果I²C总线锁死, 会使I2CSTATUS 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 图 5- 14位超时计数器. 用户写1清TIF为0.

图 5-23 I²C 超时计数器框图

5.6.3 寄存器映射

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

寄存器	偏移量	R/W	描述	复位后的值
I2C_BA = 0x4012_0000				
I2CON	I2Cx_BA+0x00	R/W	I ² C 控制寄存器	0x0000_0000
I2CADDR0	I2Cx_BA+0x04	R/W	I ² C 从机地址寄存器 0	0x0000_0000
I2CDAT	I2Cx_BA+0x08	R/W	I ² C 数据寄存器	0x0000_0000
I2CSTATUS	I2Cx_BA+0x0C	R	I ² C 状态寄存器	0x0000_00F8
I2CLK	I2Cx_BA+0x10	R/W	I ² C 时钟时钟分频寄存器	0x0000_0000
I2CTOC	I2Cx_BA+0x14	R/W	I ² C 超时控制寄存器	0x0000_0000
I2CADDR1	I2Cx_BA+0x18	R/W	从机地址寄存器 1	0x0000_0000
I2CADDR2	I2Cx_BA+0x1C	R/W	从机地址寄存器 2	0x0000_0000
I2CADDR3	I2Cx_BA+0x20	R/W	从机地址寄存器 3	0x0000_0000
I2CADM0	I2Cx_BA+0x24	R/W	从机隐藏地址寄存器 0	0x0000_0000
I2CADM1	I2Cx_BA+0x28	R/W	从机隐藏地址寄存器 1	0x0000_0000
I2CADM2	I2Cx_BA+0x2C	R/W	从机隐藏地址寄存器 2	0x0000_0000
I2CADM3	I2Cx_BA+0x30	R/W	从机隐藏地址寄存器 3	0x0000_0000

5.6.4 寄存器描述

I²C控制寄存器 (I2CON)

寄存器	偏移量	R/W	描述	复位后的值
I2CON	I2C_BA+0x00	R/W	I ² C 控制寄存器	0x0000_0000

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

Bits	描述	
[31:8]	保留	保留
[7]	EI	使能中断 1 = 使能I ² C中断功能 0 = 禁止I ² C中断功能
[6]	ENSI	I ² C 控制器使能位 1 = 使能 0 = 禁止 当ENSI置1, I ² C串行功能使能, SDA和SCL对应的GPIO管脚必须被设置成SDA, SCL功能.
[5]	STA	I ² C 起始控制位 STA置1, 进入主机模式, 如果总线处于空闲状态, I ² C硬件会送出起始信号或重复起始信号.
[4]	STO	I ² C 停止标志 在主机模式下设置STO将在I ² C总线上产生STOP时序, 硬件会监测总线上的状态, 当检测到STOP状态时, 自动将此位清零。在从机模式下设置STO将复位I ² C硬件为“未定址”模式, 也就是说它已经退出了从主机接收数据的从机接收模式
[3]	SI	I ² C 中断标志位. 在I2CSTATUS寄存器上出现一个新的I ² C信号时, SI将由硬件置位, 如果EI(I2CON [7])位被置'1', I ² C中断请求, 可以产生中断。SI必须由软件清除, 之

		后I ² C控制器才会进行下一步动作。
[2]	AA	接收应答标志位 0 = 在下面情况下，在应答时钟脉冲时，SCL上没有应答信号(SDA上高电平): 1) I ² C为主机接收模式，已经接收一个数据。2) I ² C为可寻址的从机接收模式，已经接收一个数据。 1 = 在下面情况下，在应答时钟脉冲时，SCL上没有应答信号(SDA上高电平): 1) 接收到自己的地址；2) I ² C为主机接收模式，已经接收一个数据。3) I ² C为可寻址的从机接收模式，已经接收一个数据。
[1:0]	保留	保留

I²C 数据寄存器 (I2CDAT)

寄存器	偏移量	R/W	描述	复位后的值
I2CDAT	I2C_BA+0x08	R/W	I ² C 数据寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
I2CDAT[7:0]							

Bits	描述	
[31:8]	保留	保留
[7:0]	I2CDAT	I ² C 数据寄存器 Bit [7:0] 为8位I ² C数据.

I²C 状态寄存器 (I2CSTATUS)

寄存器	偏移量	R/W	描述	复位后的值
I2CSTATUS	I2C_BA+0x0C	R/W	I ² C 状态寄存器	0x0000_00F8

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
I2CSTATUS[7:3]					0	0	0

Bits	描述	
[31:8]	保留	保留
[7:0]	I2CSTATUS	<p>I²C 状态寄存器</p> <p>低三位始终是0；高5位包含状态码，状态码有26可能；当I2CSTATUS的值是F8H，表示没有串行断请求；其它的所有的I2CSTATUS值可以反映I²C的状态。当进入这些状态时会产生一个状态中断请求(SI=1)。一个有效的状态码在SI被硬件设为'1'后一个周期内反映到I2CSTATUS中；在SI被软件清'0'后一个周期内反映到I2CSTATUS中。另外，状态码是00H时表示总线错误；当'起始'或'结束'时出现帧结构错误时会产生总线错误。</p>

I²C 时钟分频寄存器 (I2CLK)

寄存器	偏移量	R/W	描述	复位后的值
I2CLK	I2C_BA+0x10	R/W	I ² C 时钟分频寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
I2CLK[7:0]							

Bits	描述	
[31:8]	保留	保留
[7:0]	I2CLK	I ² C 时钟分频寄存器 I ² C波特率 = (system clock) / (4x(I2CLK+1))

I²C 超时计数寄存器 (I2CTOC)

寄存器	偏移量	R/W	描述	复位后的值
I2CTOC	I2C_BA+0x14	R/W	I ² C 超时计数寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留					ENTI	DIV4	TIF

Bits	描述	
[31:3]	保留	保留
[2]	ENTI	超时计数使能/禁用 1 = 使能 0 = 禁用 SI被清0, 14位超时计数寄存器开始计数, 再对SI置1会使计数器复位, 并重新打开计数功能。
[1]	DIV4	超时计数输入时钟除4 1 = 使能 0 = 禁用 使能后, 溢出时间/4
[0]	TIF	超时标志 I ² C的定时器溢出时, 硬件置位, 若允许中断 (EI=1) 则产生中断。 写1清0.

I²C 从机地址寄存器 (I2CADDRx)

寄存器	偏移量	R/W	描述	复位后的值
I2CADDR0	I2C_BA+0x04	R/W	从机地址寄存器 0	0x0000_0000
I2CADDR1	I2C_BA+0x18	R/W	从机地址寄存器 1	0x0000_0000
I2CADDR2	I2C_BA+0x1C	R/W	从机地址寄存器 2	0x0000_0000
I2CADDR3	I2C_BA+0x20	R/W	从机地址寄存器 3	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
I2CADDR[7:1]							GC

Bits	描述	
[31:8]	保留	保留
[7:1]	I2CADDR	I²C 地址寄存器 在主机模式下寄存器的值没有意义. 在从机模式下, 高7位作为MCU本身地址。如果地址符合硬件会自动应答.
[0]	GC	广播呼叫功能. 0 = 禁用广播呼叫功能. 1 = 允许广播呼叫功能.

I²C 从机地址掩码寄存器 (I2CADMx)

寄存器	偏移量	R/W	描述	复位后的值
I2CADM0	I2C_BA+0x24	R/W	I ² C 从机地址掩码寄存器 0	0x0000_0000
I2CADM1	I2C_BA+0x28	R/W	I ² C 从机地址掩码寄存器 1	0x0000_0000
I2CADM2	I2C_BA+0x2C	R/W	I ² C 从机地址掩码寄存器 2	0x0000_0000
I2CADM3	I2C_BA+0x30	R/W	I ² C 从机地址掩码寄存器 3	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
I2CADM[7:1]							保留

Bits	描述	
[31:8]	保留	保留
[7:1]	I2CADM	I²C 地址掩码寄存器: 1 = 使能掩码 (接收到的相应地址位不予辨识) 0 = 禁用掩码 (接收到的地址必须完全符合正确的地址内容) I ² C总线支持多地址辨识。当打开掩码时，接收到的从机地址相应位是否正确不予处理，当选择为禁用掩码时，从机地址相应位必须完全符合其真实的地址才给与响应。
[0]	保留	保留

5.6.5 操作模式

5种操作模式：主机/传输，主机/接收，从机/传输，从机/接收和广播呼叫模式。

在实际应用中，I²C端口可以作为主机和从机。在从机模式，I²C端口寻找自身从机地址和广播呼叫地址，如果有地址被检测到，从机准备接收或发送数据（通过设置AA位），应答信号在第9个时钟脉冲时发送，此时，如果中断已经使能，则发生中断请求。在主控芯片要成为总线主机时，在进入主机模式之前，硬件等待总线空闲以使从机动作不被中断，在主机模式，如果总线仲裁失败，I²C立即切换到从机模式，并检测自身从机地址。

5.6.5.1 主机传输模式

当SCL线上输出时钟信号时，SDA线上输出数据。置位STA，主机产生起始信号，主机向总线传输从机地址（一般为7位）+方向位。此时方向位(R/W)为0，表示写入(W)。因此传输内容为SLA+W，形成完整的8位数据。然后等待接收应答(ACK)信号。接收到应答信号后，向被寻址到的从机发送数据，等待ACK信号。输出起始或停止信号，标志开始传输或者停止传输

5.6.5.2 主机接收模式

(R/W)内容为1，表示读入(R)。此时传输内容为SLA+R。当SCL线上输出时钟信号时，SDA线上输出数据。一次接收8位串行数据，每接收到一个字节，响应一个接收标志，输出起始或者停止信号，标志开始传输或者停止传输。

5.6.5.3 从机接收模式

从串行线上接收SDA和SCL信号。每接收完一个字节，发送一个响应位，并辨识是否有起始或结束标志。由硬件识别从机地址。

5.6.5.4 从机传输模式

第一个字节是在从机接收模式接收并处理的，然后传输方向位被翻转，当SCL线上接收到时钟信号时，SDA脚上输出数据，并辨识是否有起始或停止标志。

5.6.6 数据传输 5 种操作

5种操作模式：主机/传输，主机/接收，从机/传输，从机/接收和广播呼叫模式。在SI位清除后，I²C中的STA, STO和AA位将决定I²C硬件的下一个状态。一个新的动作完成后，将更新一个新的状态码并置位SI标志。如果I²C中断控制位EI (I2CON [7]) 置位，可在中断服务子程序中根据新的状态码执行相应的动作。

数据传输每种模式见 图 5-25 到 图 5-29.

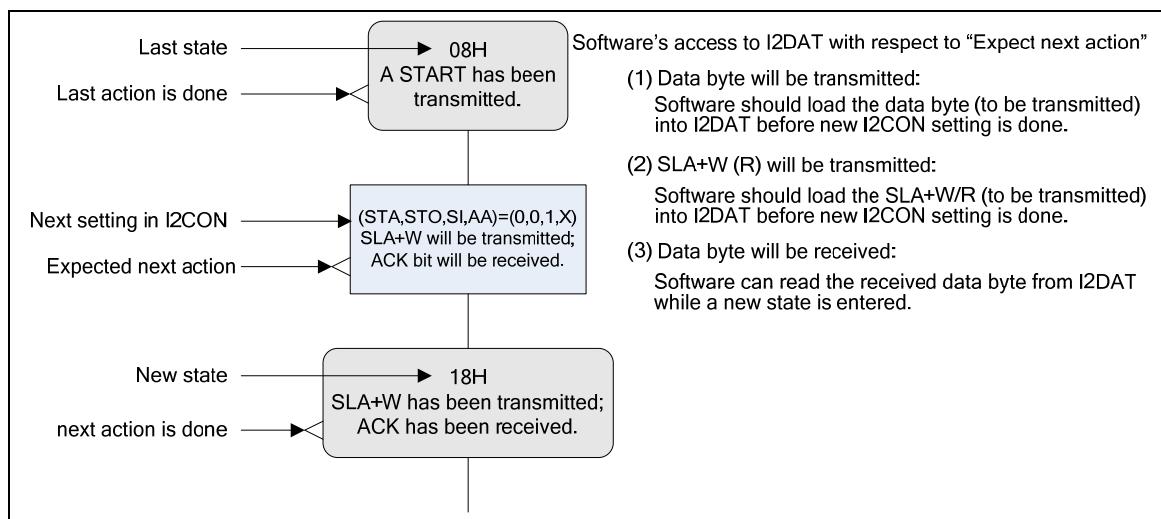


图 5-24 传输流程图

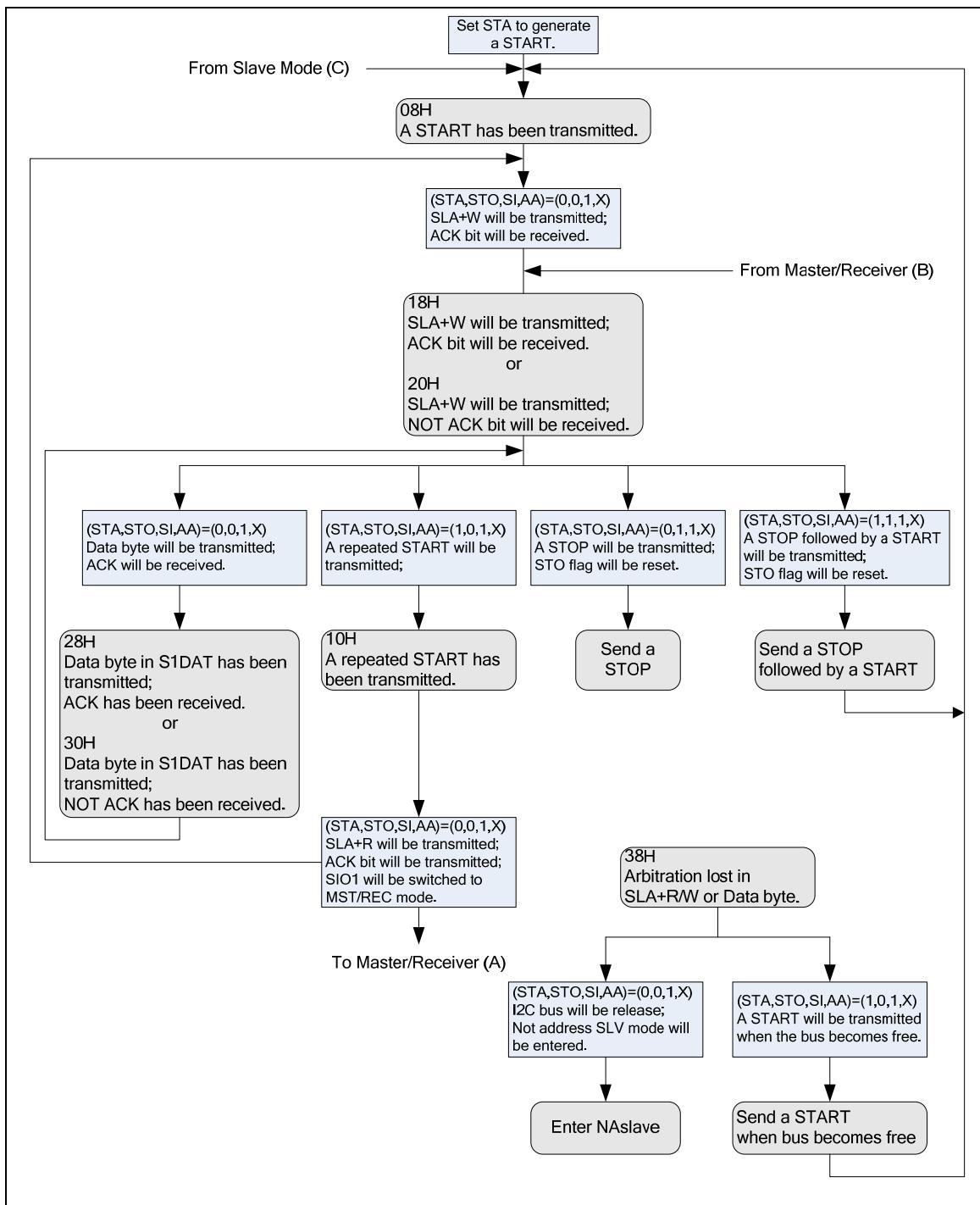


图 5-25 主机传输模式

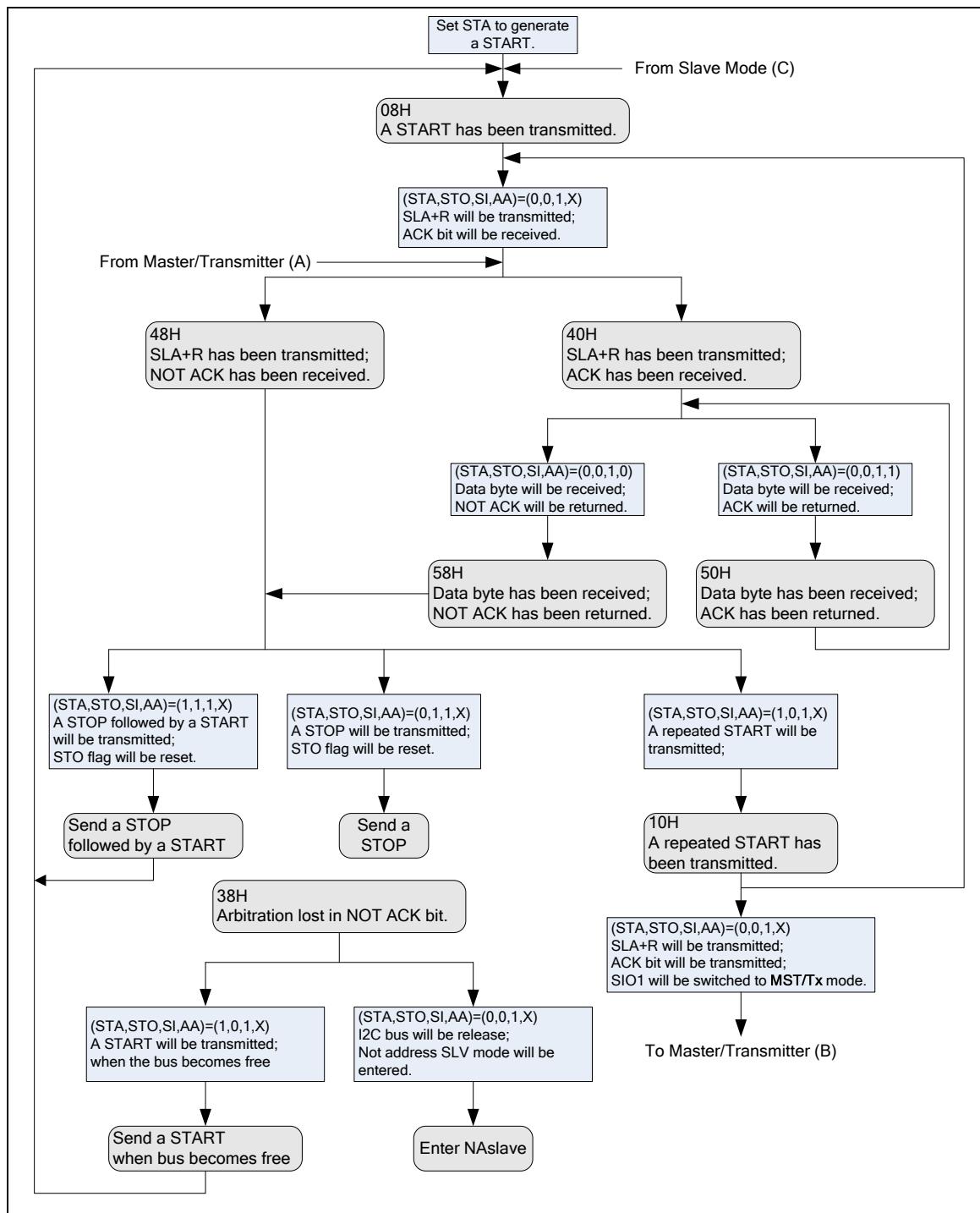


图 5-26 主机接收模式

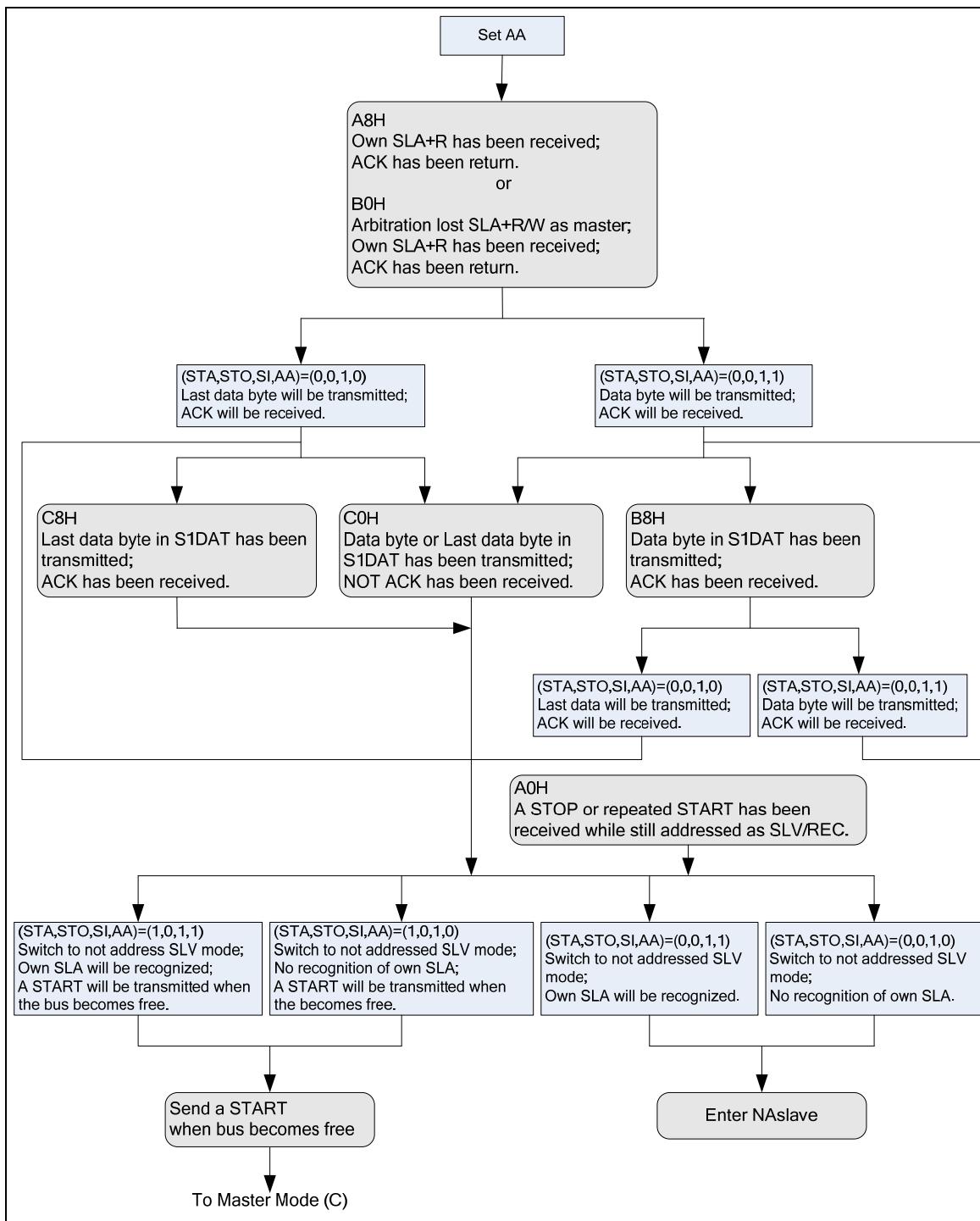


图 5-27 从机传输模式

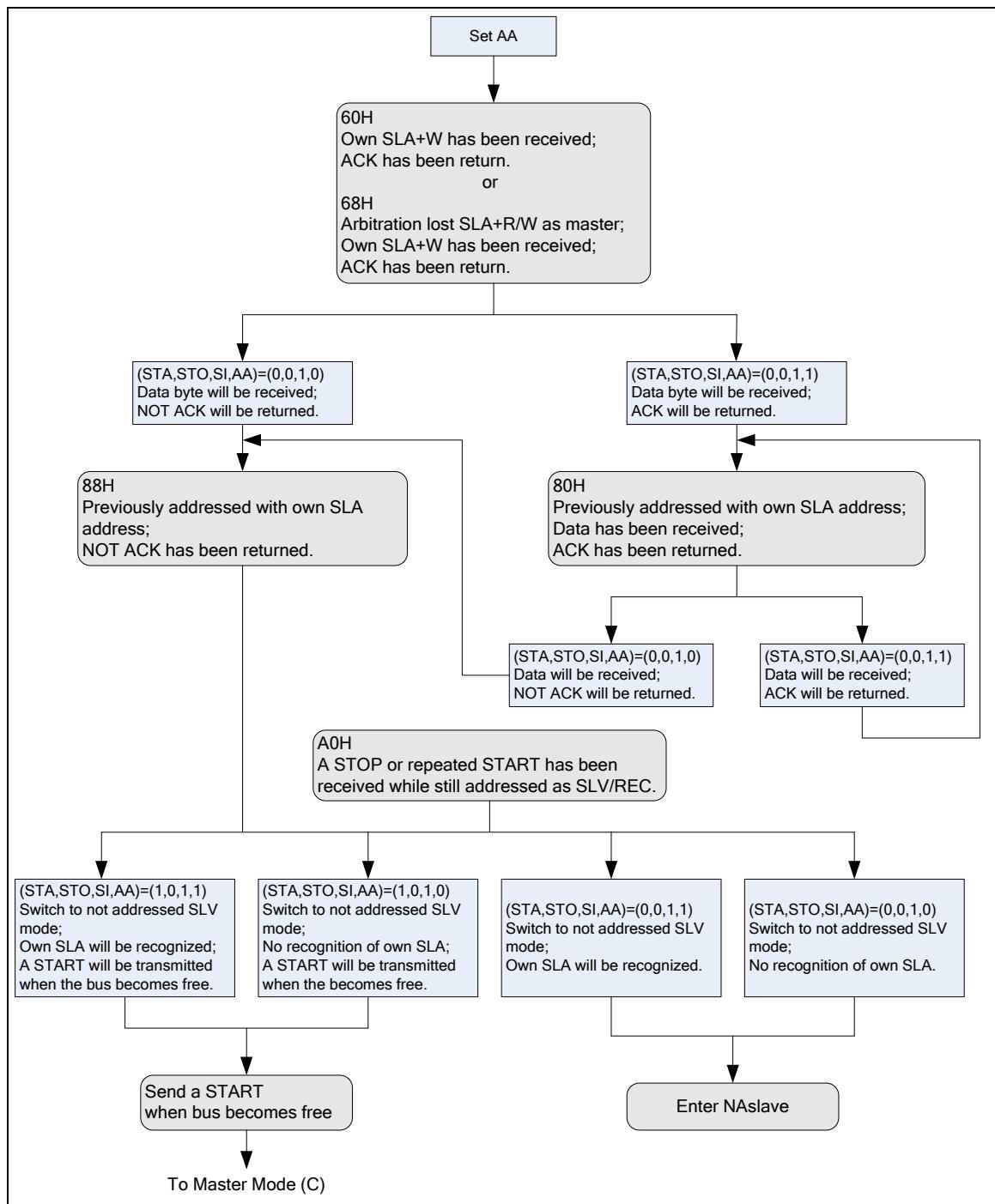


图 5-28 从机接收模式

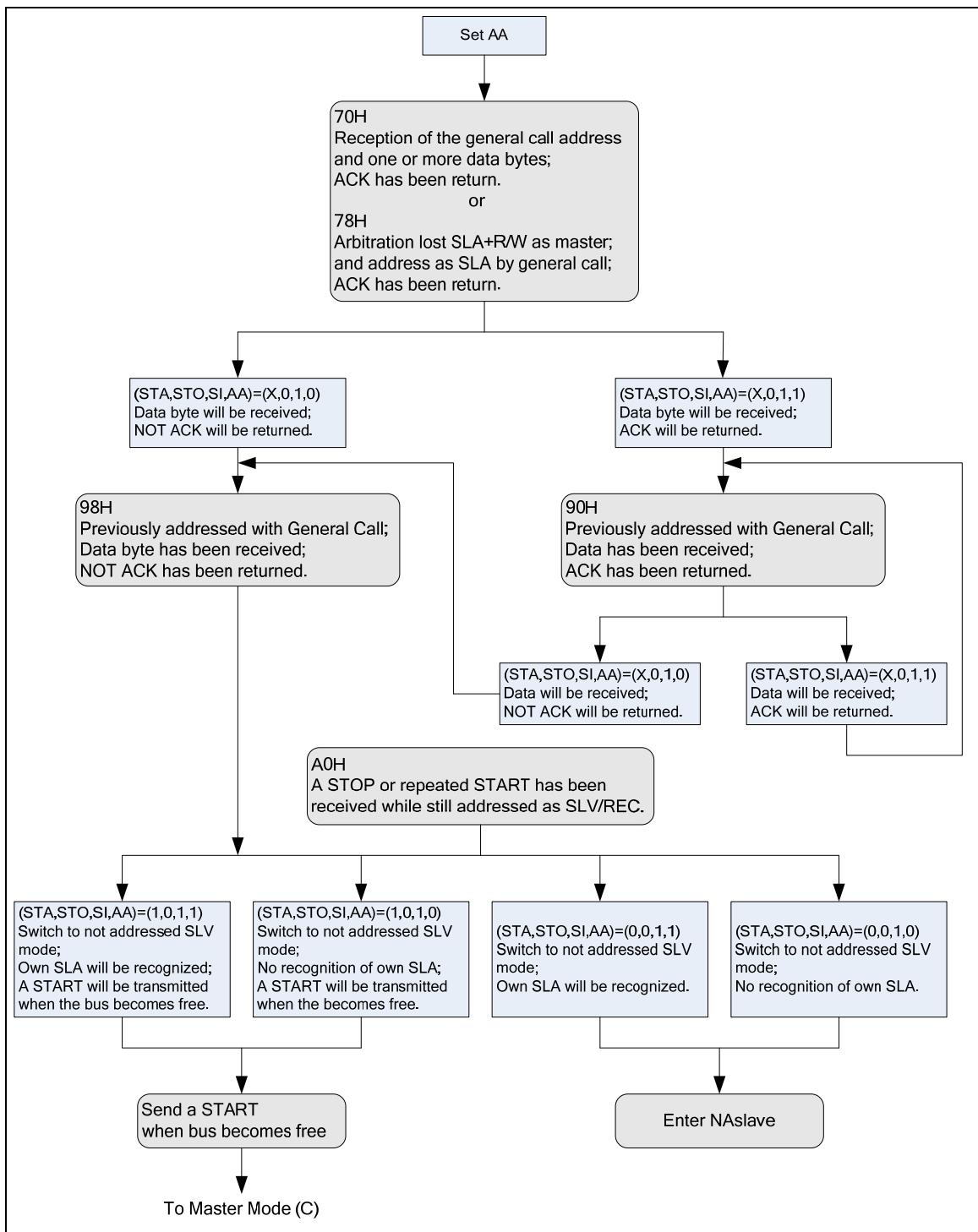


图 5-29 全呼模式

5.7 PWM 发生器和捕捉定时器 (PWM)

5.7.1 简介

NuMicro™ NUC122有1组PWM，2个PWM发生器，可配置成4个独立的PWM输出PWM0~PWM3，或2组互补的PWM对，(PWM0, PWM1), (PWM2, PWM3)，有可编程的死区发生器。

每个PWM发生器带有一个8位预分频器，一个5级分频器(1, 1/2, 1/4, 1/8, 1/16)，两个PWM定时器包括2个时钟选择，两个16位向下计数器用于PWM周期控制，两个16位比较计数器用于PWM 占空比控制，还有一个死区发生器。2个PWM发生器提供8个独立的中断标志，当PWM向下计数周期达到零时触发中断。PWM发生器可以定义为单次触发模式或连续输出PWM波形模式。

当PCR.DZEN01置位，PWM0 与 PWM1形成互补的PWM周期，这一对PWM的周期，占空比和死区时间由PWM0定时器和死区发生器0决定. 同样 PWM 互补对(PWM2, PWM3)由 PWM2定时器和死区发生器2控制。

为防止PWM输出抖动不稳定波形，16位向下计数计数器和16位比较计数器采用两级缓存。当用户向计数器/比较计数器内写入值后，只有当计数器/比较计数器的值计数到0后，新值才会被写入计数器/比较器。该两级缓存可以避免PWM输出时产生干扰波形。

当16位向下计数计数器达到0时，中断请求产生。如果PWM定时器被定义为连续模式，当向下计数器达到0时，会自动重新导入设定值(CNRx)并从新开始运行下一个周期。如果定时器设为单触发模式，向下计数器停止计数，并产生中断请求。

比较计数器数据用于设定脉宽，计数控制逻辑在计数器计数到比较值时将PWM输出变高。

使能模块的输入捕捉功能，PWM输出引脚切换作捕捉功能。捕捉器0和PWM0使用同一个定时器，捕捉器1和PWM1使用另一组定时器，以此类推。在使用捕捉功能之前，必须预先配置PMW定时器。输入端口有上升沿时，PWM计数器的值锁存入CRLR，输入端口有下降沿时PWM计数器锁存入CFLR。设定CCR0.CRL_IE0[1] (上升沿触发中断有效)和CCR0.CFL_IE0[2]] (下降沿触发中断有效),可以使捕捉器通道0作为中断源。同样设定CCR0.CRL_IE1[17] 和CCR0.CFL_IE1[18]，可以设定通道1。依照上述方法，通过设定相对应CCR2的寄存器可设定捕捉通道2和3。捕捉模块触发中断时，PWM计数器将被置为初值。

最大的捕捉频率由捕捉中断延迟决定. 捕捉中断发生时，软件至少执行以下三步: 第一步，读PIIRx 获取中断源，第二步，读CRLRx/CFLRx(x=0~3) 获取捕捉值和写1清PIIRx为0. 如果中断延迟花T0完成，捕捉信号在(T0)间隔内必须不能改变. 此条件下，最大捕捉频率为1/T0.

例如:

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

因此最大捕捉频率为 $1/900\text{ns} \approx 1000\text{ kHz}$

5.7.2 特性

5.7.2.1 PWM功能特性:

- PWM 组有两个PWM发生器. 每个PWM支持8位预分频，一个时钟除频，两个PWM定时器，一个死区发生器和两个PWM输出.
- 最高16位解析度
- PWM 中断与PWM周期同步
- One-shot 或 Auto-reload模式
- 可支持4路PWM通道或2对PWM通道

5.7.2.2 捕捉功能特征:

- 与PWM发生器共用定时器模块
- 支持4个捕捉输入通道， 4 个捕捉输入通道与4 PWM 输出通道共享
- 每个通道支持1个上升沿锁存寄存器(CRLR)，一个下降沿锁存寄存器 (CFLR) 和捕捉中断标志 (CAPIFx)

5.7.3 框图

图 5-30 到图 5-33 描述 PWM 对的结构(PWM-Timer 0&1 为一对, PWM-Timer 2&3 为另一对).

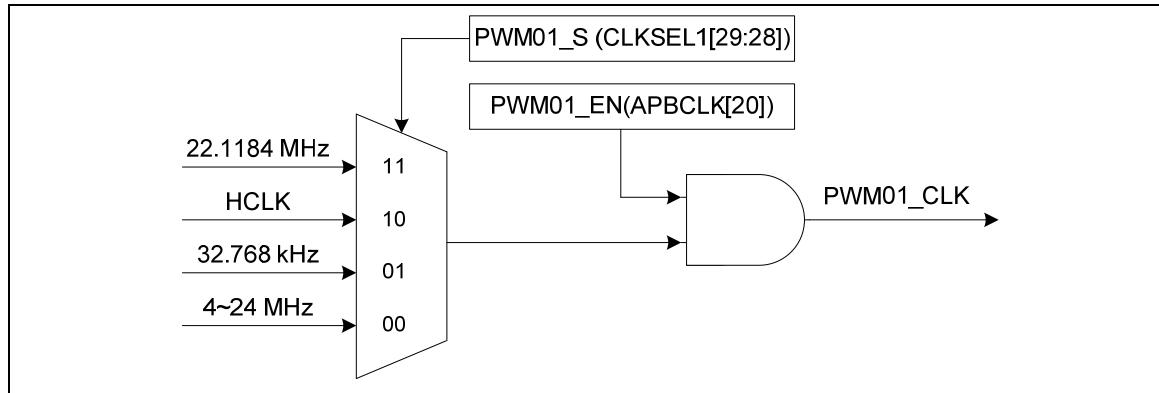


图 5-30 PWM 发生器 0 时钟源控制

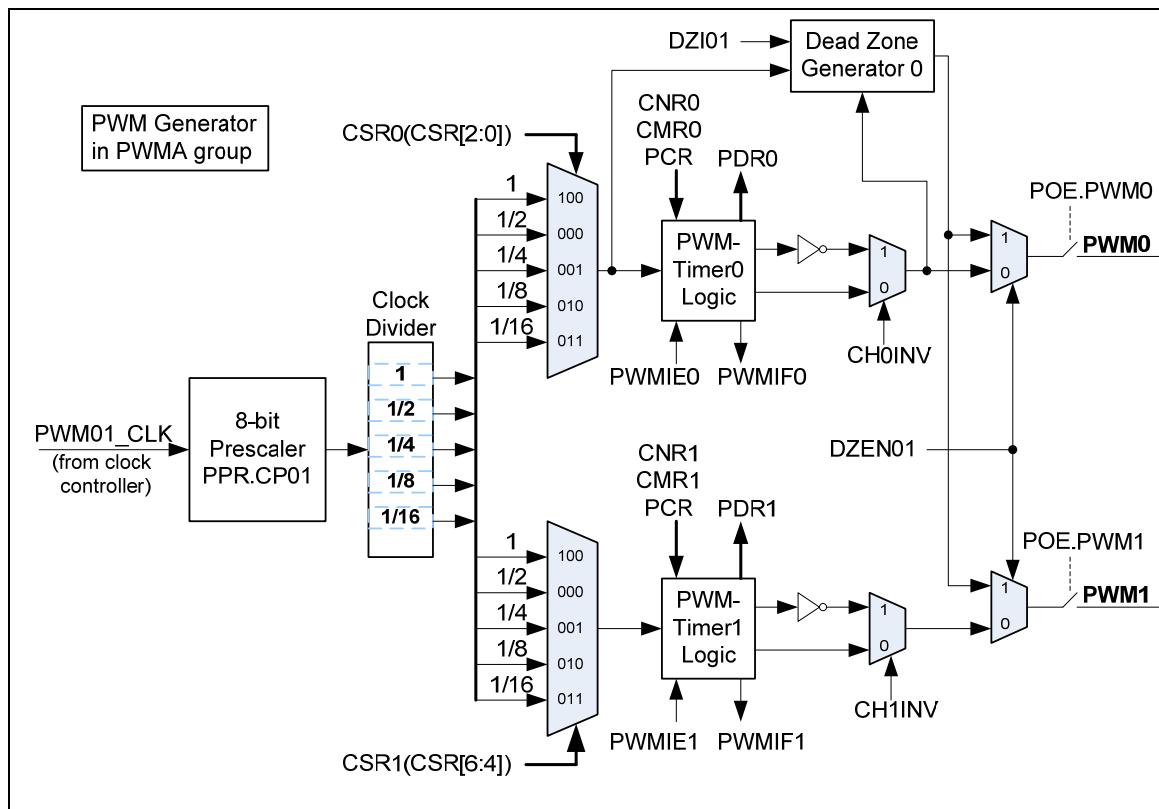


图 5-31 PWM 发生器 0 结构框图

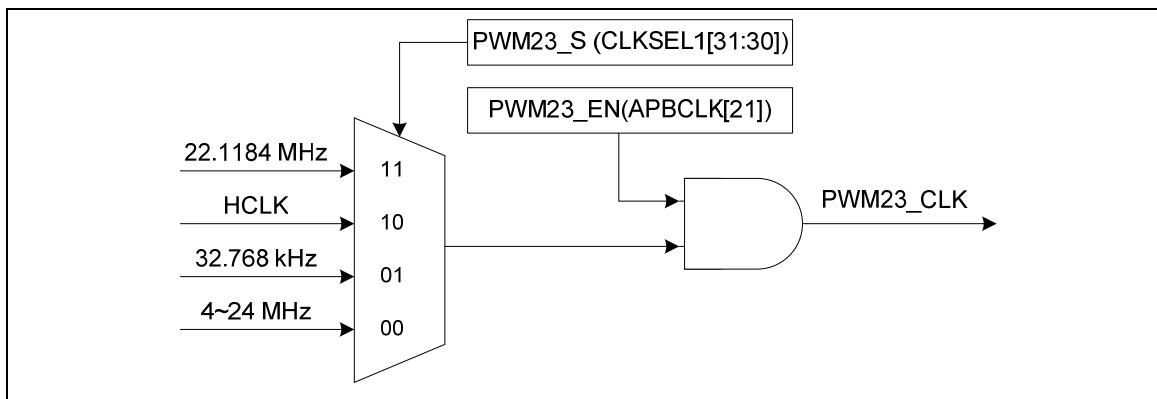


图 5-32 PWM 发生器 2 时钟源控制

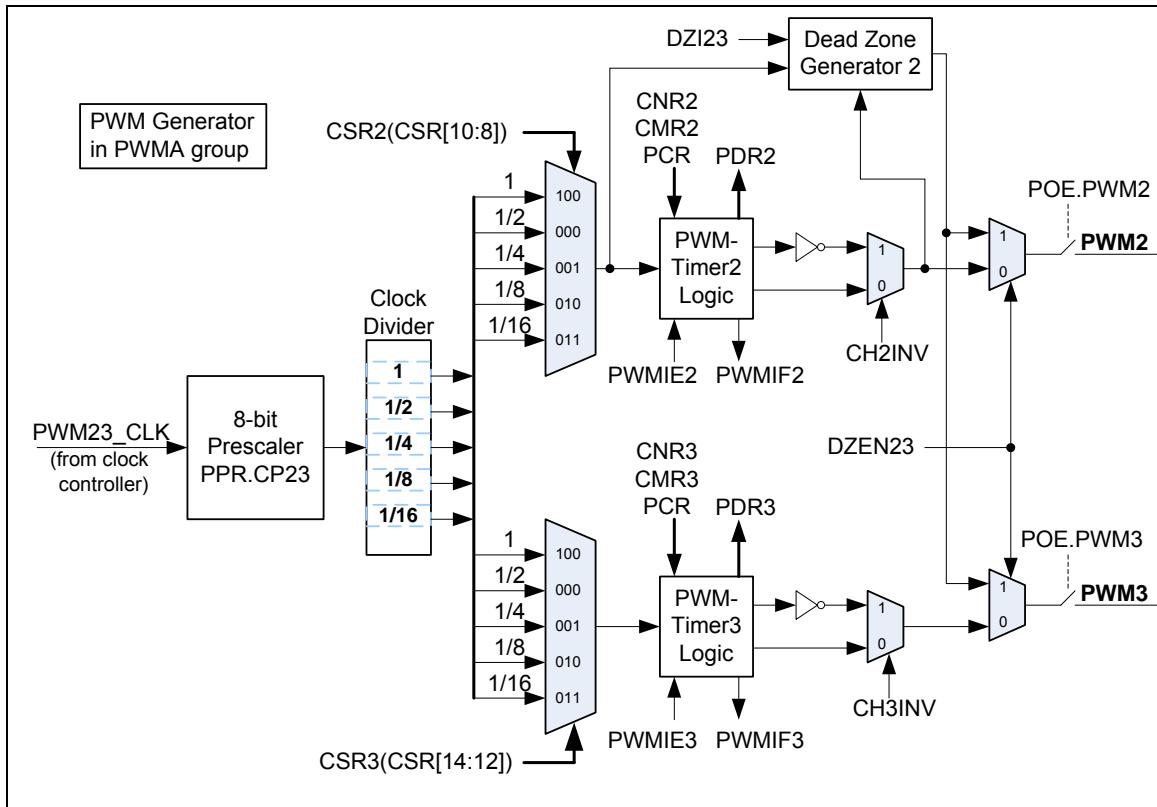


图 5-33 PWM 发生器 2 结构框图

5.7.4 功能描述

5.7.4.1 PWM-定时器操作

PWM 周期 和占空比控制由向下计数的PWM寄存器(CNR)以及PWM比较寄存器(CMR)控制。PWM 定时器工作时序如图 5-35 脉宽调制的公式如下，相应的周期如图 5-34. 注：在PWM功能使能前，MCU相应GPIO管脚应配置成PWM功能(使能 POE 并禁止 CAPENR).

- PWM 频率 = $\text{PWM}_{xy_CLK}/[(\text{prescale}+1)*(\text{clock divider})*(\text{CNR}+1)]$; xy代表01, 23, 取决于所选择的PWM通道.
- 占空比 = $(\text{CMR}+1)/(\text{CNR}+1)$
- $\text{CMR} \geq \text{CNR}$: PWM 输出为高
- $\text{CMR} < \text{CNR}$: PWM低脉宽= $(\text{CNR}-\text{CMR})$ unit¹; PWM高脉宽= $(\text{CMR}+1)$ unit
- $\text{CMR} = 0$: PWM低脉宽 = (CNR) unit; PWM高脉宽= 1 unit

Note: [1] Unit = 一个 PWM 时钟周期.

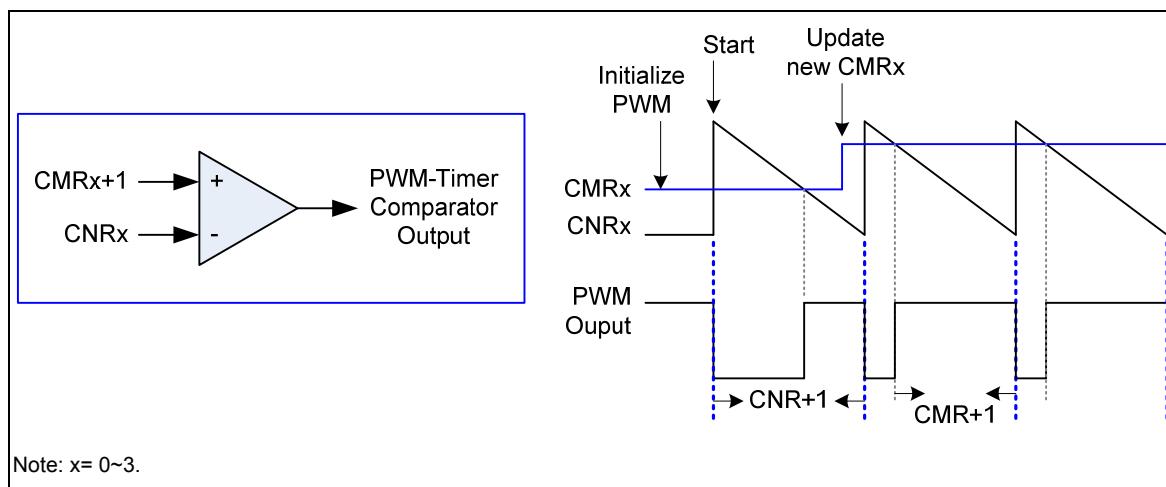


图 5-34 PWM 定时器内部比较器输出

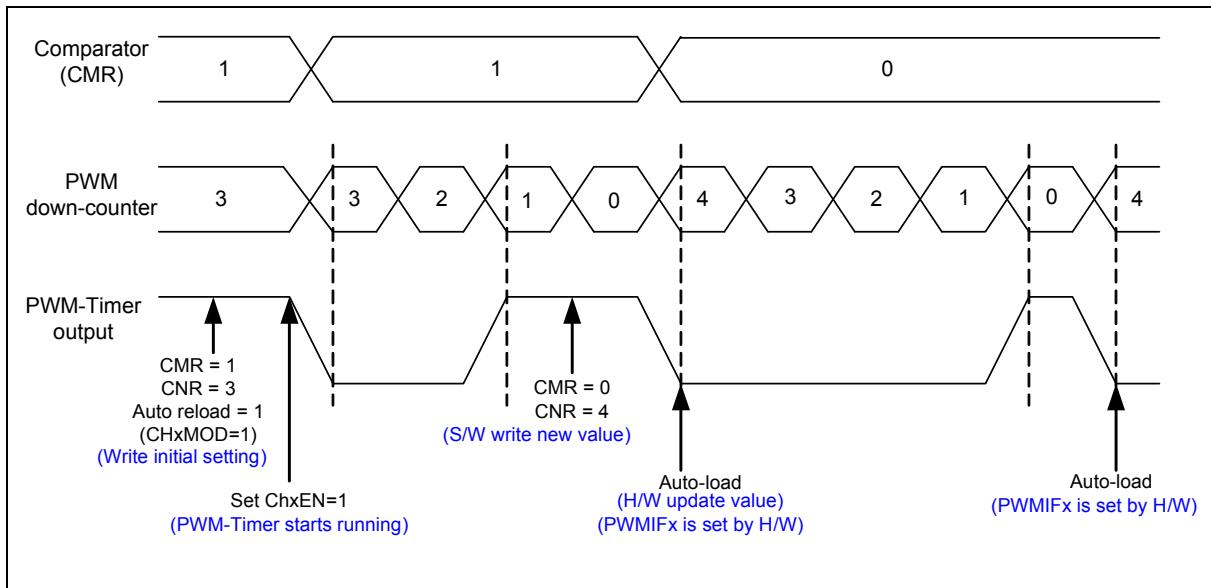


图 5-35 PWM 定时器操作时序

5.7.4.2 PWM 双缓存，自动重载以及单触发模式

PWM 定时器具有双缓存功能。寄存器预先设定的值，在一个周期完成后，可以自动重载。PWM 计数器值写入 CNRx，并可从 PDRx 内读出。

PWM 控制寄存器(PCR) 的 CH0MOD 位定义 PWM0 是自动重载模式或是单触发模式。自动重载模式下，当 PWM 计数器计到 0，MCU 自动重载 CNR0 值到 PWM 计数器。如果 CNR0 设定为 0，PWM 计数器计数到 0 后，暂停运行状态。如果此时 CH0MOD 设定为 0，计数器停止运行。PWM1~PWM3 运行状态与 PWM0 相同。

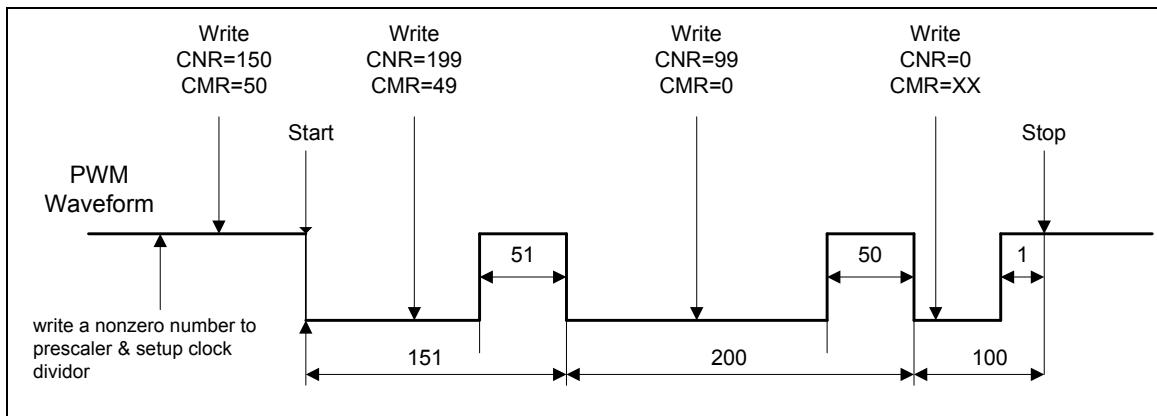


图 5-36 PWM 双缓存图解

5.7.4.3 可调占空比

双缓存允许 CMRx 字当前运行时改写。下一个周期内值被导入运行。

文件更新日期：6月21日, 2011

版本 V1.07

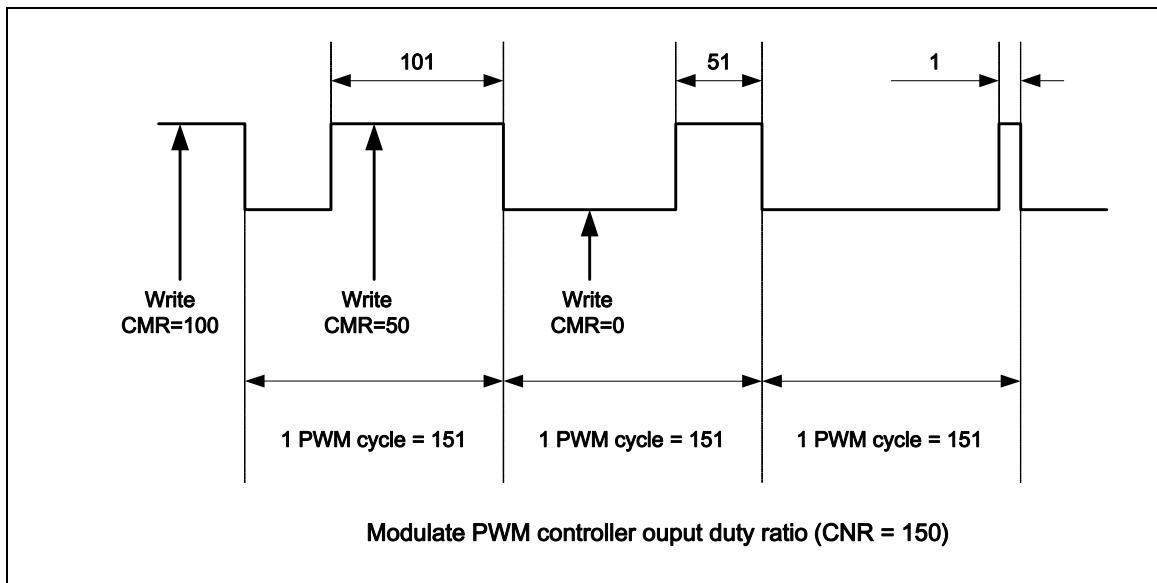


图 5-37 PWM 控制输出占空比

5.7.4.4 死区发生器

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

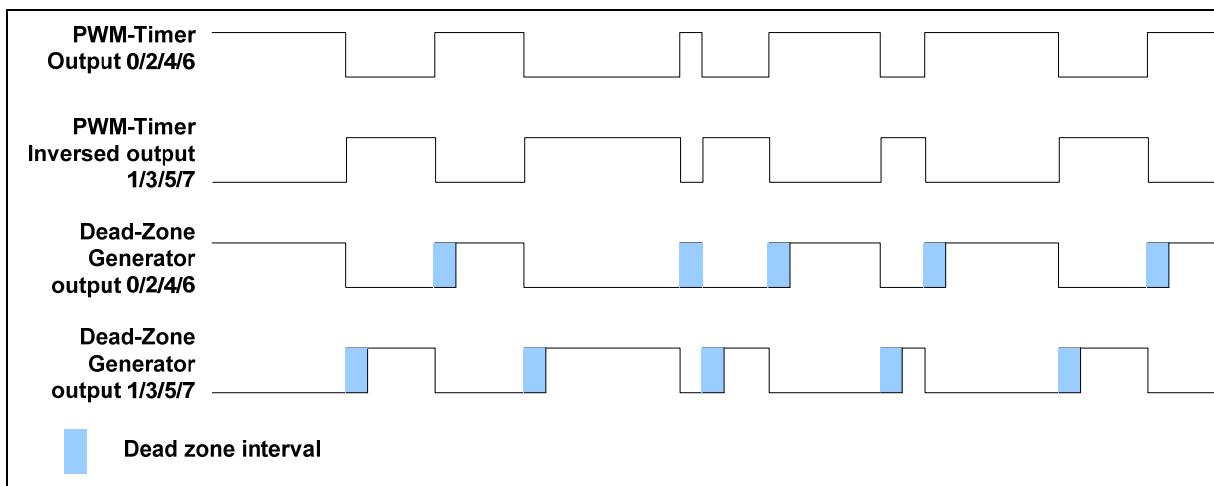


图 5-38 PWM 对输出带死区发生器操作

5.7.4.5 捕捉模式

捕捉器 0 和 PWM0 使用同一个定时器，捕捉器 1 和 PWM1 使用另一组定时器，以此类推。在使用捕捉功能之前，必须预先配置 PMW 定时器。当输入信号有上升沿转变时，PWM 计数器的值将存入 CRLRx 寄存器；当输入信号有下降沿转变时，PWM 计数器的值将存入 CFLRx 寄存器。设定 CCR0[1]（上升沿触发中断有效）和 CCR0[2]（下降沿触发中断有效），可以使捕捉器通道 0 作为中断源。

同样设定CCR0[17] 和CCR0[18], 可以设定通道1。通道2 和3同样通过设定CCR1[1],CCR1[2] 和CCR1[17], CCR1[18] 无论捕捉模块何时触发中断, PWM计数器都将重置。注: 在捕捉功能使能前, 相应的管脚必须配置为输入模式(禁止POE并使能CAPENR).

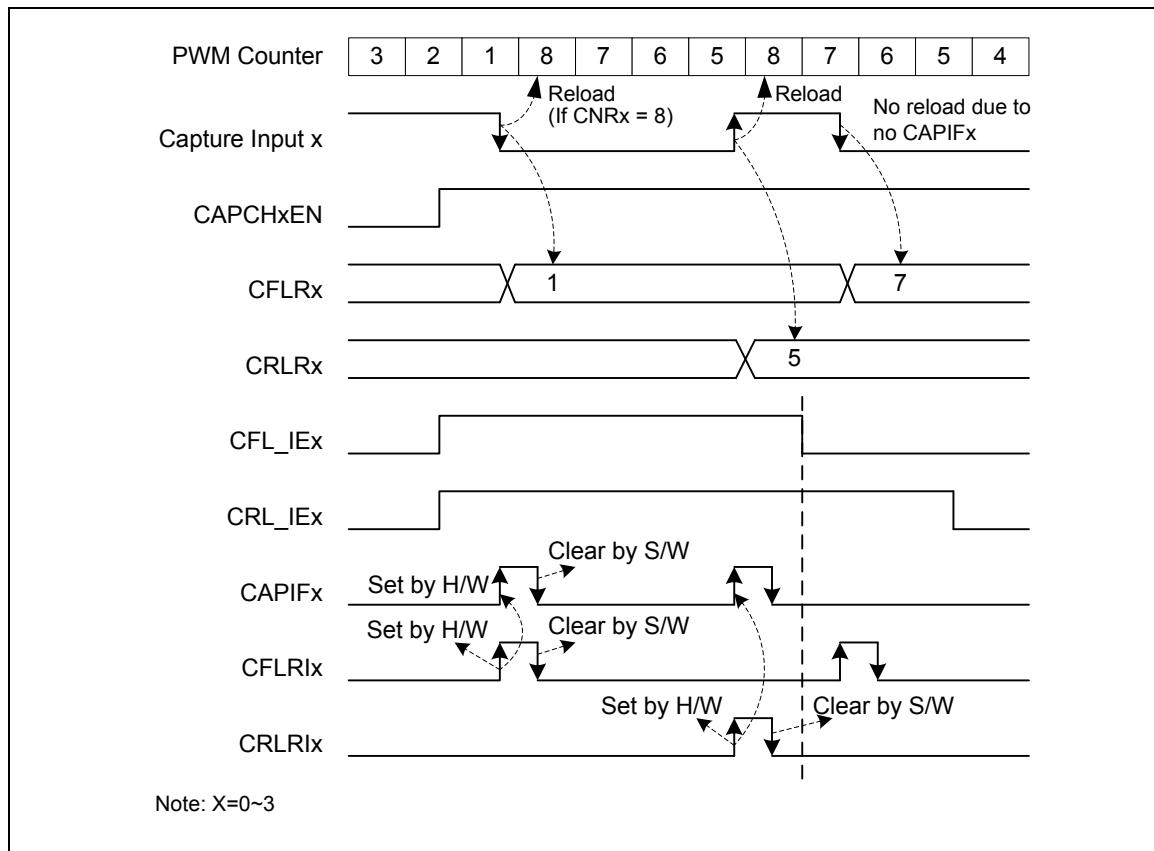


图 5-39 捕捉操作时序

在上述范例中, CNR 为8:

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

5.7.4.6 PWM-定时器中断结构

提供四个PWM中断，**PWM0_INT~PWM3_INT**，对NVIC来说。 PWM 0 与捕捉器0共用同一组中断。 PWM1 与捕捉器1 共用同一组中断，以此类推。因此，在同一个通道PWM功能和捕捉功能不能同时发生。图 5-40 描述了PWM定时器中断的结构。

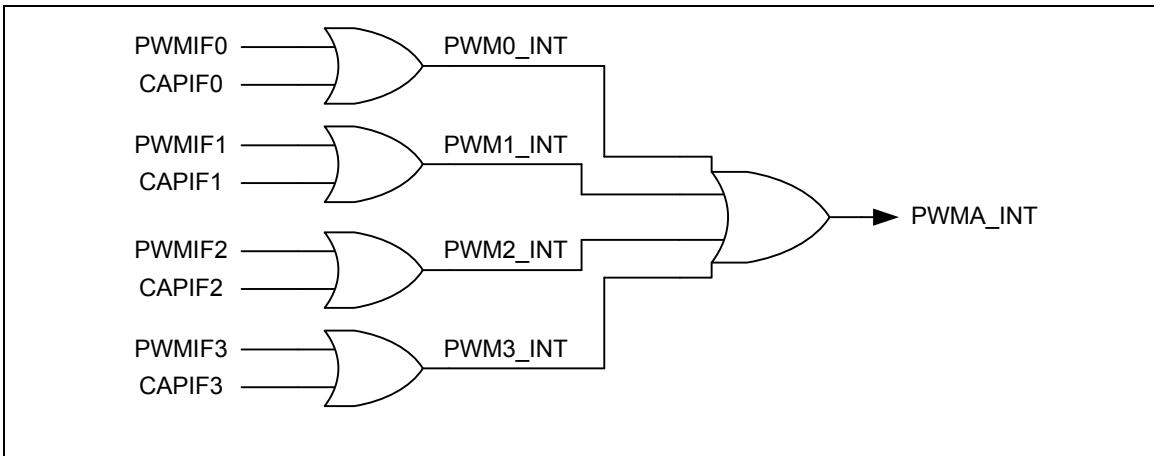


图 5-40 PWM A 组 PWM-定时器中断结构图

5.7.4.7 PWM-定时器开启步骤

下列步骤用于启动PWM.

1. 配置时钟选择 (CSR)
2. 配置预分频(PPR)
3. 配置反转打开/关闭，死区打开/关闭，自动重载/单触发模式以及PWM定时器关闭(PCR)
4. 配置比较器寄存器(CMR) 设定PWM 占空比.
5. 配置PWM计数器寄存器 (CNR) 设定PWM 周期.
6. 配置中断使能寄存器 (PIER)
7. 配置相应GPIO脚为PWM功能 (enable POE and disable CAPENR) 用于PWM通道.
8. 使能PWM定时器开始运行 (Set CHxEN = 1 in PCR)

5.7.4.8 PWM-定时器关闭步骤

方式 1:

设定 16-位向下计数计数器(CNR)为0，并查看PDR状态。当PDR达到0，关闭PWM定时器 (PCR的CHxEN位). (**推荐**)

方式 2:

设定16位向下计数计数器(CNR)为0，当中断条件发生。在中断内关闭PWM定时器(PCR的CHxEN位). (**推荐**)

方式 3:

直接关闭PWM定时器(PCR的CHxEN位). (**不推荐**)

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

5.7.4.9 捕捉开始步骤

1. 配置时钟选择(CSR)
2. 配置预分频(PPR)
3. 配置通道使能，上升/下降沿中断使能以及输入信号反转打开/关闭(CCR0, CCR1)
4. 配置PWM计数器寄存器(CNR)
5. 配置相应的GPIO为捕捉功能(禁用POE和使能 CAPENR) 用于相应的PWM通道.
6. 使能PWM定时器运行(Set CHxEN = 1 in PCR)

5.7.5 寄存器映射

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

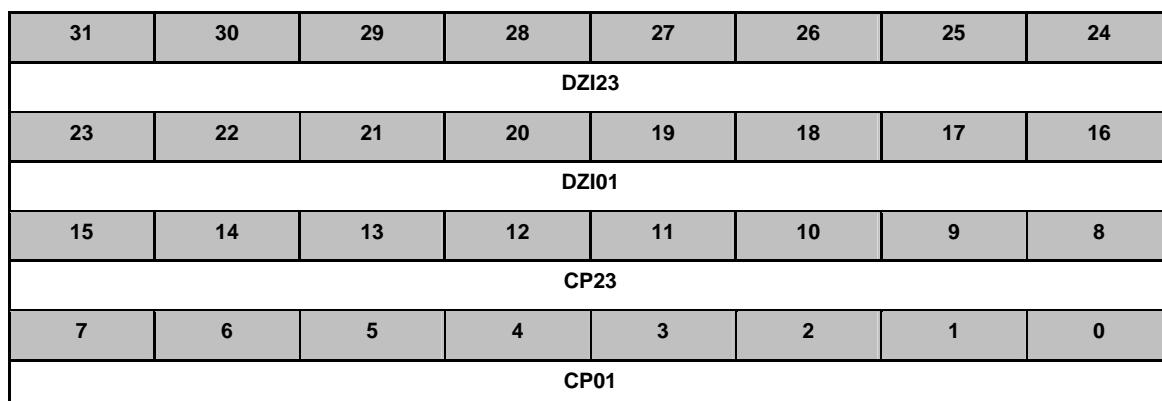
寄存器	偏移量	R/W	描述	复位后的值
PWMA_BA = 0x4004_0000 (PWM group A)				
PPR	PWMA_BA+0x00	R/W	PWM A 组预分频寄存器	0x0000_0000
CSR	PWMA_BA+0x04	R/W	PWM A 组时钟选择寄存器	0x0000_0000
PCR	PWMA_BA+0x08	R/W	PWM A 组控制寄存器	0x0000_0000
CNR0	PWMA_BA+0x0C	R/W	PWM A 组计数寄存器0	0x0000_0000
CMR0	PWMA_BA+0x10	R/W	PWM A 组比较寄存器0	0x0000_0000
PDR0	PWMA_BA+0x14	R	PWM A 组数据寄存器0	0x0000_0000
CNR1	PWMA_BA+0x18	R/W	PWM A 组计数寄存器1	0x0000_0000
CMR1	PWMA_BA+0x1C	R/W	PWM A 组比较寄存器1	0x0000_0000
PDR1	PWMA_BA+0x20	R	PWM A 组数据寄存器1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWM A 组计数寄存器2	0x0000_0000
CMR2	PWMA_BA+0x28	R/W	PWM A 组比较寄存器2	0x0000_0000
PDR2	PWMA_BA+0x2C	R	PWM A 组数据寄存器2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWM A 组计数寄存器3	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWM A 组比较寄存器3	0x0000_0000
PDR3	PWMA_BA+0x38	R	PWM A 组数据寄存器3	0x0000_0000
PIER	PWMA_BA+0x40	R/W	PWM A 组中断使能寄存器	0x0000_0000
PIIR	PWMA_BA+0x44	R/W	PWM A 组中断标志寄存器	0x0000_0000
CCR0	PWMA_BA+0x50	R/W	PWM A 组捕捉控制寄存器0	0x0000_0000
CCR2	PWMA_BA+0x54	R/W	PWM A 组捕捉控制寄存器2	0x0000_0000
CRLR0	PWMA_BA+0x58	R	PWM A 组捕捉上升沿锁存寄存器 (Channel 0)	0x0000_0000
CFLR0	PWMA_BA+0x5C	R	PWM A 组捕捉下降沿锁存寄存器 (Channel 0)	0x0000_0000
CRLR1	PWMA_BA+0x60	R	PWM A 组捕捉上升沿锁存寄存器 (Channel 1)	0x0000_0000
CFLR1	PWMA_BA+0x64	R	PWM A 组捕捉下降沿锁存寄存器 (Channel 1)	0x0000_0000
CRLR2	PWMA_BA+0x68	R	PWM A 组捕捉上升沿锁存寄存器 (Channel 2)	0x0000_0000
CFLR2	PWMA_BA+0x6C	R	PWM A 组捕捉下降沿锁存寄存器 (Channel 2)	0x0000_0000

CRLR3	PWMA_BA+0x70	R	PWM A 组捕捉上升沿锁存寄存器 (Channel 3)	0x0000_0000
CFLR3	PWMA_BA+0x74	R	PWM A 组捕捉下降沿锁存寄存器 (Channel 3)	0x0000_0000
CAPENR	PWMA_BA+0x78	R/W	PWM A 组捕捉输入 0~3 使能寄存器	0x0000_0000
POE	PWMA_BA+0x7C	R/W	PWM A 组使能通道 0~3 输出	0x0000_0000

5.7.6 寄存器描述

PWM 预分频寄存器 (PPR)

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



Bits	描述
[31:24]	DZI23 通道2与通道3的死区间隔 (PWM A 组的 PWM2 和 PWM3 pair) 该8位寄存器决定死区长度. 单位死区时间长度由相关CSR位决定.
[23:16]	DZI01 通道0与通道1的死区间隔 (PWM A 组的 PWM0 和 PWM1 pair) 该8位寄存器决定死区长度. 单位死区时间长度由相关CSR位决定.
[15:8]	CP23 时钟预分频 2 (PWM-timer2 & 3 for group A) 在选定到相应的PWM定时器之前时钟输入被(CP23 + 1)除频 如果CP23=0, 预分频器2输出时钟停止。相应PWM定时器也停止.
[7:0]	CP01 时钟预分频 0 (PWM-timer 0 & 1 for group A) 在选定到相应的PWM定时器之前时钟输入被(CP01 + 1)除频 如果CP01=0, 预分频器0输出时钟停止。相应PWM定时器也停止.

PWM 时钟选择寄存器 (CSR)

寄存器	偏移量	R/W	描述	复位后的值
CSR	PWMA_BA+0x04	R/W	PWM A 组时钟选择寄存器	0x0000_0000

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

Bits	描述													
[31:15]	保留	保留												
[14:12]	CSR3	PWM 定时器 3 时钟源选择 (PWM timer 3 for group A) 选择 PWM 定时器的时钟输入. <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>CSR3 [14:12]</td> <td>输入时钟除频</td> </tr> <tr> <td>100</td> <td>1</td> </tr> <tr> <td>011</td> <td>16</td> </tr> <tr> <td>010</td> <td>8</td> </tr> <tr> <td>001</td> <td>4</td> </tr> <tr> <td>000</td> <td>2</td> </tr> </table>	CSR3 [14:12]	输入时钟除频	100	1	011	16	010	8	001	4	000	2
CSR3 [14:12]	输入时钟除频													
100	1													
011	16													
010	8													
001	4													
000	2													
[11]	保留	保留												
[10:8]	CSR2	PWM 定时器 2 时钟源选择 (PWM timer 2 for group A) 选择 PWM 定时器的时钟输入. (表与 CSR3 相同)												
[7]		保留												
[6:4]	CSR1	PWM 定时器 1 时钟源选择 (PWM timer 1 for group A) 选择 PWM 定时器的时钟输入.												

		(表与 CSR3相同)
[3]	保留	保留
[2:0]	CSR0	PWM 定时器 0 时钟源选择 (PWM timer 0 for group A) 选择PWM定时器时钟输入. (表与 CSR3相同)

PWM控制寄存器 (PCR)

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

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

Bits	描述	
[31:28]	保留	保留
[27]	CH3MOD	PWM-定时器 3 自动加载/单触发模式 (PWM timer 3 for group A) 1 = 自动重载模式 0 = 单触发模式 注: 如果该位变化, 会使CNR3 和CMR3 清位.
[26]	CH3INV	PWM-定时器 3 输出反向使能 (PWM timer 3 for group A) 1 = 反转打开 0 = 反转关闭
[25]	保留	保留
[24]	CH3EN	PWM-定时器 3 使能 (PWM timer 3 for group A) 1 = 使能相应的PWM定时器运行 0 = 停止相应的PWM定时器运行
[23:20]	保留	保留
[19]	CH2MOD	PWM-定时器 2 自动重载/单触发模式 (PWM timer 2 for group A) 1 = 自动重载模式 0 = 单触发模式 注: 如果该位变化, 会使CNR2和CMR2清位.

[18]	CH2INV	PWM-定时器 2 输出反向使能(PWM timer 3 for group A) 1 = 反向打开 0 = 反向关闭
[17]	保留	保留
[16]	CH2EN	PWM-定时器 2 使能 (PWM timer 2 for group A) 1 = 使能相应的PWM定时器运行 0 = 停止相应的PWM定时器运行
[15:12]	保留	保留
[11]	CH1MOD	PWM-定时器 1 自动加载/单触发模式(PWM timer 1 for group A) 1 = 自动重载模式 0 = 单触发模式 注: 如果该位变化，会使CNR1和CMR1清位.
[10]	CH1INV	PWM-定时器 1 输出反向使能 (PWM timer 1 for group A) 1 = 反转打开 0 = 反转关闭
[9]	保留	保留
[8]	CH1EN	PWM-定时器 1 使能 (PWM timer 1 for group A) 1 = 使能相应的PWM定时器运行 0 = 停止相应的PWM定时器运行
[7:6]	保留	保留
[5]	DZEN23	死区 2 发生器使能 (PWM2 and PWM3 pair for PWM group A) 1 = 使能 0 = 禁用 注: 当死区发生器使能， PWM2和PWM3将成为相应一对输出，
[4]	DZEN01	死区 0 发生器使能 (PWM0 and PWM1 pair for PWM group A) 1 = 使能 0 = 禁用 注: 当死区发生器使能， PWM0和PWM1将成为相应一对输出
[3]	CH0MOD	PWM-定时器 0 自动加载/单触发模式 (PWM timer 0 for group A) 1 = 自动重载模式 0 = 单触发模式 注: 如果该位变化，会使CNR0和CMR0清位.

[2]	CH0INV	PWM-定时器 0 输出反向使能 (PWM timer 0 for group A) 1 = 反转打开 0 = 反转关闭
[1]	保留	保留
[0]	CH0EN	PWM-定时器 0 使能 (PWM timer 0 for group A) 1 = 使能相应的PWM定时器运行 0 = 停止相应的PWM定时器运行

PWM计数器寄存器3-0 (CNR3-0)

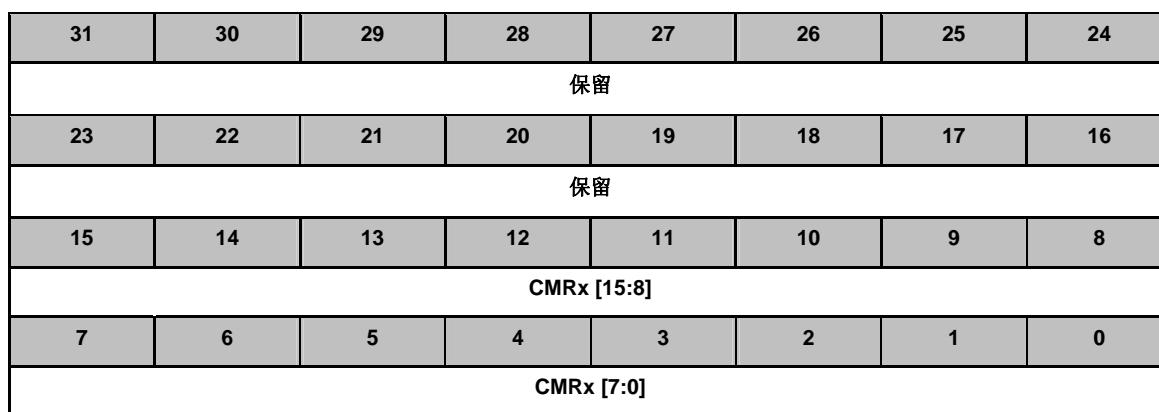
寄存器	偏移量	R/W	描述	复位后的值
CNR0	PWMA_BA+0x0C	R/W	PWM A 组计数器寄存器 0	0x0000_0000
CNR1	PWMA_BA+0x18	R/W	PWM A 组计数器寄存器 1	0x0000_0000
CNR2	PWMA_BA+0x24	R/W	PWM A 组计数器寄存器 2	0x0000_0000
CNR3	PWMA_BA+0x30	R/W	PWM A 组计数器寄存器 3	0x0000_0000

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

Bits	描述	
[31:16]	保留	保留
[15:0]	CNRx	<p>PWM 定时器载入值 CNR 决定PWM的周期.</p> <ul style="list-style-type: none"> ● PWM 频率 = PWMxy_CLK/[(prescale+1)*(clock divider)*(CNR+1)]; xy 代表01, 23, 取决于所选择的PWM通道. ● 占空比= (CMR+1)/(CNR+1). ● CMR >= CNR: PWM输出高. ● CMR < CNR: PWM 低脉宽= (CNR-CMR) unit; PWM高脉宽= (CMR+1) unit. ● CMR = 0: PWM低脉宽 = (CNR) unit; PWM 高脉宽 = 1 unit (Unit = one PWM clock cycle) <p>注: CNR写入数据后将在下一个PWM周期生效.</p>

PWM比较寄存器3-0 (CMR3-0)

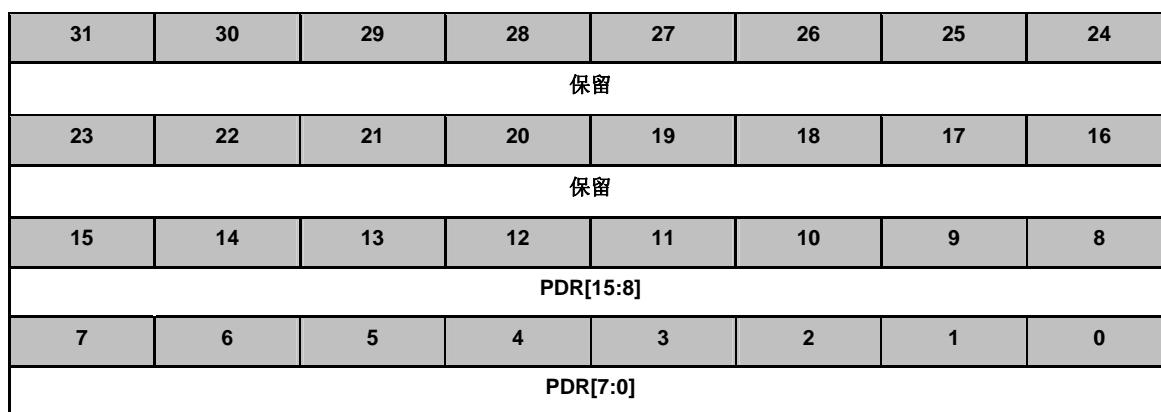
寄存器	偏移量	R/W	描述	复位后的值
CMR0	PWMA_BA+0x10	R/W	PWM A 组比较寄存器 0	0x0000_0000
CMR1	PWMA_BA+0x1C	R/W	PWM A 组比较寄存器 1	0x0000_0000
CMR2	PWMA_BA+0x28	R/W	PWM A 组比较寄存器 2	0x0000_0000
CMR3	PWMA_BA+0x34	R/W	PWM A 组比较寄存器 3	0x0000_0000



Bits	描述	
[31:16]	保留	保留
[15:0]	CMRx	<p>PWM 比较寄存器</p> <p>CMR 决定 PWM 的占空比.</p> <ul style="list-style-type: none"> ● PWM 频率 = PWMxy_CLK/[(prescale+1)*(clock divider)*(CNR+1)]; xy 代表 01, 23, 取决于所选择的PWM通道. ● 占空比= (CMR+1)/(CNR+1). ● CMR >= CNR: PWM输出高. ● CMR < CNR: PWM低脉宽= (CNR-CMR) unit; PWM高脉宽= (CMR+1) unit. ● CMR = 0: PWM低脉宽= (CNR) unit; PWM高脉宽= 1 unit (Unit = one PWM clock cycle) <p>Note: CNR写入数据后将在下一个PWM周期生效.</p>

PWM数据寄存器3-0 (PDR 3-0)

寄存器	偏移量	R/W	描述	复位后的值
PDR0	PWMA_BA0+0x14	R	PWM A 组数据寄存器 0	0x0000_0000
PDR1	PWMA_BA0+0x20	R	PWM A 组数据寄存器 1	0x0000_0000
PDR2	PWMA_BA0+0x2C	R	PWM A 组数据寄存器 2	0x0000_0000
PDR3	PWMA_BA0+0x38	R	PWM A 组数据寄存器 3	0x0000_0000



Bits	描述	
[31:16]	保留	保留
[15:0]	PDRx	PWM 数据寄存器 用来查询16位计数器当前值.



PWM中断使能寄存器(PIER)

寄存器	偏移量	R/W	描述	复位后的值
PIER	PWMA_BA+0x40	R/W	PWM A 组中断使能寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留				PWMIE3	PWMIE2	PWMIE1	PWMIE0

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



PWM 中断标志寄存器 (PIIR)

寄存器	偏移量	R/W	描述	复位后的值
PIIR	PWMA_BA+0x44	R/W	PWM A 组中断标志寄存器	0x0000_0000

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

Bits	描述	
[31:4]	保留	保留
[3]	PWMIF3	PWM 通道 3 中断状态 当PWM3的中断使能位PWMIE3为1、且PWM计数器向下计数至0时，硬件将该位置1。软件写1清除该位
[2]	PWMIF2	PWM 通道 2 中断状态 当PWM2的中断使能位PWMIE2为1、且PWM计数器向下计数至0时，硬件将该位置1。软件写1清除该位
[1]	PWMIF1	PWM 通道 1 中断状态 当PWM1的中断使能位PWMIE1为1、且PWM计数器向下计数至0时，硬件将该位置1。软件写1清除该位
[0]	PWMIF0	PWM 通道 0 中断状态 当PWM0的中断使能位PWMIE0为1、且PWM计数器向下计数至0时，硬件将该位置1。软件写1清除该位

注：用户可以写PIIR的对应位清除中断标志



捕捉控制寄存器(CCR0)

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

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

Bits	描述	
[31:24]	保留	保留
[23]	CFLRI1	CFLR1 锁定标志位 当 PWM 组输入通道1发生下降沿变化时, CFLR1 锁定PWM向下计数器的值, 硬件置位。 软件写1清该位.
[22]	CRLRI1	CRLR1 锁定标志位 当 PWM 组输入通道1发生上升沿变化时, CRLR1 锁定PWM向下计数器的值, 硬件置位。 软件写1清该位.
[21]	保留	保留
[20]	CAPIF1	通道 1 捕捉中断标志 如果PWM组通道1上升锁定中断使能(CRL_IE1=1), PWM组通道1发生上升沿转变会使CAPIF1 为高; 同样, 如果 PWM 组通道 1 下降锁定中断使能(CFL_IE1=1), 下降沿会使CAPIF1置高. 写 1 清该位为0
[19]	CAPCH1EN	通道 1 捕捉功能使能 1 = 使能PWM组通道1的捕捉功能 0 = 禁用PWM组通道1的捕捉功能

		当使能时，捕捉功能将锁定PWM计数器，并存储到CRLR (上升锁存) 或CFLR (下降锁存). 当禁用时，捕捉器不更新CRLR 或CFLR, 并关闭通道1中断.
[18]	CFL_IE1	通道 1 下降锁定中断使能 1 = 使能下降沿锁存中断 0 = 禁用下降沿锁存中断 使能时，如果检测到PWM组通道1有下降沿转变，捕捉产生中断.
[17]	CRLIE1	通道 1 上升锁定中断使能 1 = 使能上升沿锁存中断 0 = 禁用上升沿锁存中断 使能时，如果检测到PWM组通道1有上升沿转变，捕捉产生中断.
[16]	INV1	使能通道 1 反转 1 = 反转打开。输入到寄存器的信号与通道上的实际信号电平反向 0 = 反转关闭
[15:8]	保留	保留
[7]	CFLRI0	CFLR0 锁定标志位 当PWM组输入通道0发生下降沿转变时，CFLR0 锁定PWM向下计数值，硬件置位。 软件写1清该位.
[6]	CRLRI0	CRLR0 锁定标志位 当PWM组输入通道0发生上升沿转变时，CRLR0锁定PWM向下计数值，硬件置位。 软件写1清该位.
[5]	保留	保留
[4]	CAPIFO	通道 0 捕捉中断标志 如果PWM组通道0上升锁定中断使能(CRLIE0=1)，PWM组通道0发生上升沿转变会使CAPIFO 为高；同样，如果 PWM 组通道 0 下降锁定中断使能(CFLIE0=1)，下降沿转变会使CAPIFO置高. 写 1 清该位为0
[3]	CAPCH0EN	通道 0 捕捉功能使能 1 = 使能PWM组通道0的捕捉功能 0 = 禁用PWM组通道0的捕捉功能 当使能时，捕捉功能锁定PWM计数器，并存储到CRLR (上升锁存) 或CFLR (下降锁存). 当禁用时，捕捉器不更新CRLR 或CFLR, 并关闭通道0中断.

[2]	CFL_IE0	通道 0 下降锁定中断使能 1 = 使能下降沿锁存中断 0 = 禁用下降沿锁存中断 使能时，如果检测到PWM组通道0有下降沿转变，捕捉产生中断.
[1]	CRL_IE0	通道 0 上升锁定中断使能 1 = 使能上升沿锁存中断 0 = 禁用上升沿锁存中断 使能时，如果检测到PWM组通道0有上升沿转变，捕捉产生中断.
[0]	INV0	使能通道 0 反向 1 = 反转打开。输入到寄存器的信号与通道上的实际信号电平反向 0 = 反转关闭



捕捉控制寄存器(CCR2)

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

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

Bits	描述	
[31:24]	保留	保留
[23]	CFLRI3	CFLR3 锁定标志位 当 PWM 组输入通道3发生下降沿转变时, CFLR3 锁定PWM向下计数器的值, 硬件置位。 软件写1清该位.
[22]	CRLRI3	CRLR3 锁定标志位 当 PWM 组输入通道3发生上升沿转变时, CRLR3 锁定PWM向下计数器的值, 硬件置位。 软件写1清该位.
[21]	保留	保留
[20]	CAPIF3	通道 3 捕捉中断标志 如果PWM组通道3上升锁定中断使能(CRL_IE3=1), PWM组通道3发生上升沿转变时会将CAPIF3 置高; 同样, 如果PWM组通道3下降锁定中断使能(CFL_IE3=1), 下降沿会使CAPIF3置高. 写 1 清该位为0
[19]	CAPCH3EN	通道 3 捕捉功能使能 1 = 使能PWM组通道3的捕捉功能 0 = 禁用PWM组通道3的捕捉功能 当使能时, 捕捉功能锁定PWM计数器, 并存储到CRLR (上升锁存) 或CFLR (下降锁存).

		当禁用时，捕捉器不更新CRLR 或CFLR, 并关闭通道3中断
[18]	CFL_IE3	通道 3 下降锁定中断使能 1 = 使能向下锁定中断 0 = 禁用向下锁定中断 使能时，如果检测到PWM组通道3有下降沿转变，捕捉产生中断.
[17]	CRLIE3	通道 3 上升锁定中断使能 1 = 使能向上锁定中断 0 = 禁用向上锁定中断 使能时，如果检测到PWM组通道3有上升沿转变，捕捉产生中断.
[16]	INV3	使能通道 3 反向 1 = 反转打开。输入到寄存器的信号与通道上的实际信号电平反向
[15:8]	保留	保留
[7]	CFLRI2	CFLR2 锁定标志位 当PWM组输入通道0发生下降沿转变时，CFLR2 锁定PWM向下计数值，硬件置位。 软件写1清该位.
[6]	CRLRI2	CRLR2 锁定标志位 当PWM组输入通道0发生上升沿转变时，CRLR2锁定PWM向下计数值，硬件置位。 软件写1清该位.
[5]	保留	保留
[4]	CAPIF2	通道2 捕捉中断标志 如果PWM组通道2上升锁定中断使能(CRLIE2=1), PWM组通道2发生上升沿转变时会使CAPIF2 为高；同样，如果PWM组通道2下降锁定中断使能(CFLIE2=1)，下降沿会使CAPIF2置高。 写1清该位为0
[3]	CAPCH2EN	通道2 捕捉功能使能 1 = 使能PWM组通道2的捕捉功能 0 = 禁用PWM组通道2的捕捉功能 当使能时，捕捉功能锁定PWM计数器，并存储到CRLR (上升锁存) 或CFLR (下降锁存). 当禁用时，捕捉器不更新CRLR 或CFLR, 并关闭通道2中断.
[2]	CFLIE2	通道2 下降锁定中断使能 1 = 使能向下锁定中断

		0 = 禁用向下锁定中断 使能时，如果检测到PWM组通道2有下降沿转变，捕捉产生中断.
[1]	CRL_IE2	通道2 上升锁定中断使能 1 = 使能向上锁定中断 0 = 禁用向上锁定中断 使能时，如果检测到PWM组通道2有上升沿转变，捕捉产生中断.
[0]	INV2	使能通道2 反向 1 = 反转打开。输入到寄存器的信号与通道上的实际信号电平反向 0 = 反转关闭

捕捉上升沿锁存寄存器 3-0 (CRLR3-0)

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

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

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

捕捉下降沿锁存寄存器3-0 (CFLR3-0)

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

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

Bits	描述	
[31:16]	保留	保留
[15:0]	CFLRx	捕捉下降沿锁存寄存器 通道 0/1/2/3 有下降沿转变时，锁存 PWM 计数器。

捕捉输入使能寄存器(CAPENR)

寄存器	偏移量	R/W	描述	复位后的值
CAPENR	PWMA_BA+0x78	R/W	PWM A 组捕捉输入 0~3 使能寄存器	0x0000_0000

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

Bits	描述	
[31:4]	保留	保留
[3:0]	CAPENR	<p>捕捉输入使能寄存器 4组捕捉输入。Bit0~Bit3 用于控制每个输入的打开 / 关闭。 0 = 关闭 (PWMx 多功能引脚输入不会影响输入捕捉功能.) 1 = 打开 (PWMx多功能引脚输入影响输入捕捉功能.)</p> <p>CAPENR</p> <p>Bit xxx1 → 捕捉通道0 从PA.12输入 Bit xx1x → 捕捉通道1 从PA.13输入 Bit x1xx → 捕捉通道2 从PA.14输入 Bit 1xxx → 捕捉通道3 从PA.15输入</p>



PWM 输出使能寄存器(POE)

寄存器	偏移量	R/W	描述	复位后的值
POE	PWMA_BA+0x7C	R/W	PWM A 组通道 0~3 输出使能寄存器	0x0000_0000

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

Bits	描述	
[31:4]	保留	保留
[3]	PWM3	<p>通道3 输出使能寄存器 1 = 使能PWM通道3输出 0 = 禁用PWM通道3输出 注: :相应的GPIO管脚特性要根据相应的PWM要求配置</p>
[2]	PWM2	<p>通道2 输出使能寄存器 1 = 使能PWM通道2输出 0 = 禁用PWM通道2输出 注: :相应的GPIO管脚特性要根据相应的PWM要求配置</p>
[1]	PWM1	<p>通道1 输出使能寄存器 1 = 使能PWM通道1输出 0 = 禁用PWM通道1输出 注: :相应的GPIO管脚特性要根据相应的PWM要求配置</p>
[0]	PWM0	<p>通道0 输出使能寄存器 1 = 使能PWM通道0输出 0 = 禁用PWM通道0输出 注: :相应的GPIO管脚特性要根据相应的PWM要求配置</p>

5.8 实时时钟 (RTC)

5.8.1 简介

实时时钟 (RTC)模块用于记录实时时间及日历功能。时钟源由外部 32.768 KHz 低速晶振提供，管脚为X32I 及X32O 或者外接32.768 KHz 频率的信号源。RTC支持时间格式 (秒, 分, 时)寄存器 (TLR)以及日历格式(日, 月, 年)寄存器(CLR)。数据格式由BCD格式存取。模块支持闹钟功能, (TAR)寄存器用于指示时间闹铃, (CAR)用于指示日期闹铃。

RTC 模块支持时间记步及闹钟中断，通过设定TTR.TTR[2:0]，中断提供8级选择1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 及1 秒。当中断使能(RIER.AIER=1)，并且RTC 计数器内的值TLR 和CLR 与 TAR 和 CAR 相等时，中断标志 (RIIR.AIF)将被置，中断发生。如果唤醒芯片功能使能(TWKE(TTR[3])=1)时，当RTC时间与闹钟匹配时将芯片从空闲或掉电模式唤醒。

5.8.2 特征

- 支持时间计数 (秒, 分, 时)以及日历计数(日, 月, 年)，用户可用来查看时间
- 闹钟寄存器(秒, 分, 时, 日, 月, 年)
- 12-小时或 24-小时模式可选择
- 闰年自动识别
- 一周天数计数器
- 频率补偿寄存器 (FCR)
- 所有时间日期由BCD码组成
- 提供8级时间记步周期选择1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 及1 秒
- 支持RTC定时滴答和闹钟定时中断
- 支持芯片唤醒功能 (空闲或掉电模式)

5.8.3 框图

RTC模块框图如下：

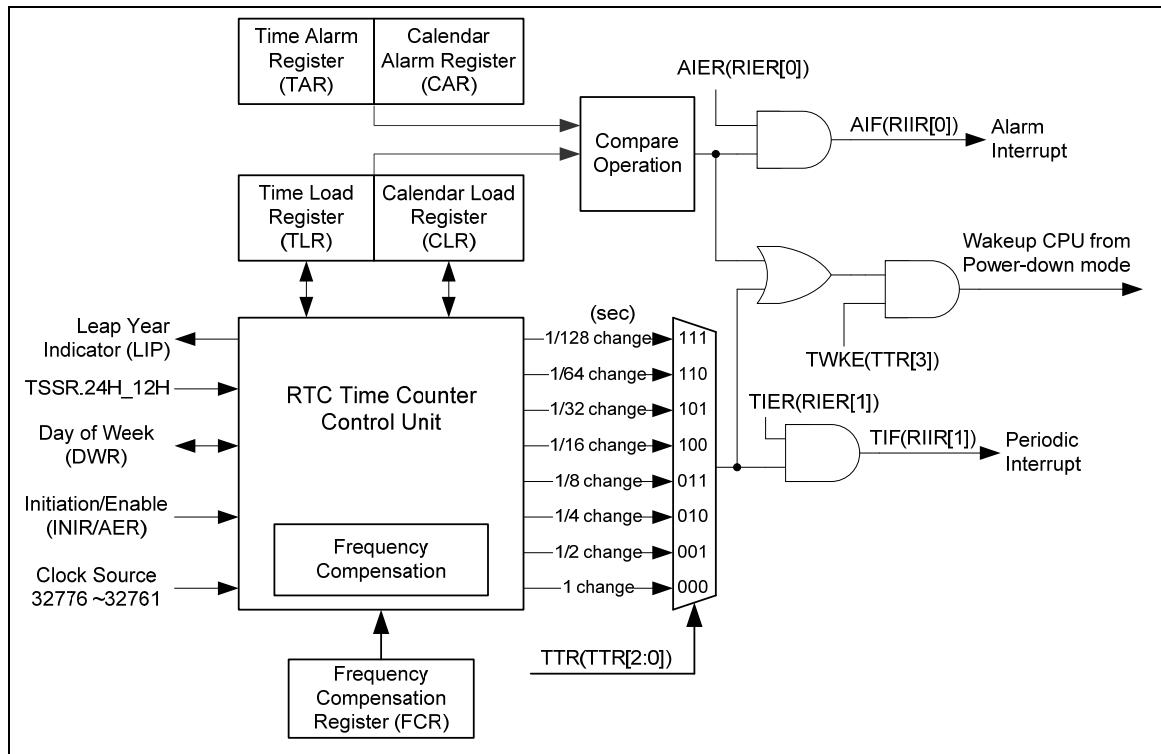


图 5-41 RTC 框图

5.8.4 功能描述

5.8.4.1 访问 RTC 寄存器

由于RTC时钟与系统时钟不同，用户对RTC寄存器写入后，2个RTC时钟周期(60us)后，寄存器内的值才会被更新。

另外，用户必须保证RTC模块载入合理数据，RTC不会对DWR和CLR内的合理性进行确认。

5.8.4.2 RTC 初始化

RTC模块上电后，RTC处于复位状态，用户需写数据(0xa5eb1357)至寄存器INIR让RTC退出reset状态。一旦INIR 被写入0xa5eb1357，RTC将永久正常工作下去。

5.8.4.3 RTC 读/写使能

寄存器AER 位 15~0 作为RTC寄存器允许读/写口令，用于避免对RTC的误写。要访问RTC的寄存器AER 位15~0 必须写入0xa965以打开访问限制。然后会有效512 个RTC时钟 (大约15ms)，之后将自动关闭。用户可通过读取AER.ENF 标志位来确认RTC是否正常工作.

5.8.4.4 频率补偿

RTC FCR 允许软件对时钟输入进行频率补偿，时钟输入频率必须在32776 Hz 至32761 Hz之间. 用户可用频率计测GPIO上的RTC时钟的频率，并存储在Flash中，当再次上电时用于数据检索。下列为频率输入的举例.

例 1:

频率计数测量: 32773.65 Hz (> 32768 Hz)

整数部分: 32773 => 0x8005

FCR.Integer = 0x05 – 0x01 + 0x08 = 0x0c

分数部分: 0.65 x 60 = 39 => 0x27

FCR.Fraction = 0x27

例 2

频率计数测量: 32765.27Hz (≤ 32768 Hz)

整数部分: 32765 => 0x7FFD

FCR.Integer = 0x0A – 0x01 – 0x08 = 0x04

分数部分: 0.27 x 60 = 16.2=> 0x10

FCR.Fraction = 0x10

5.8.4.5 时间和日历计数

TLR 和 CLR 用于载入时间和日历。TAR 和CAR 用于闹钟。BCD码格式。

5.8.4.6 12/24小时时间切换选择

根据TSSR位0 选择12/24小时格式.

5.8.4.7 一周日期计数器

RTC提供一周计日格式寄存器DWR. 内部的值由0至6 用于表示周日至周六.

5.8.4.8 时间周期中断

通过选择TTR.TTR[2:0], 中断具有8级选项1/128, 1/64, 1/32, 1/16, 1/8, 1/4, 1/2 以及1秒。当周期中断寄存器RIER.AIER 置 1, MCU根据TTR寄存器内设定的值发生中断.

5.8.4.9 时间闹钟中断

当 RTC 计数器 TLR 和 CLR 等于闹钟设定时间 TAR 和 CAR, 如果已设定中断使能(RIER.AIER=1), 闹钟中断标志(RIER.AIER=1)被置位.

5.8.4.10 应用指南:

1. TAR, CAR, TLR和CLR 寄存器为BCD格式.
2. 用户需要确认, 填入的数据为有效格式。例如, CLR = 201a (年), 13 (月), 00 (日), 或CLR与DWR不匹配, 等.
3. 复位状态:

寄存器	复位状态
AER	0
CLR	05/1/1 (年/月/日)
TLR	00:00:00 (小时:分:秒)
CAR	00/00/00 (年/月/日)
TAR	00:00:00 (小时:分:秒)
TSSR	1 (24小时制)
DWR	6 (星期六)
RIER	0
RIIR	0
LIR	0
TTR	0

4. 在TLR 和TAR中, 仅2位BCD码用于指示“年”。目前仅支持20xx年份, 不支持19xx或21xx年.

5.8.5 寄存器映射

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

寄存器	偏移量	R/W	描述	复位后的值
RTC_BA = 0x4000_8000				
INIR	RTC_BA+0x00	R/W	RTC 初始寄存器	0x0000_0000
AER	RTC_BA+0x04	R/W	RTC 访问(读写)使能寄存器	0x0000_0000
FCR	RTC_BA+0x08	R/W	RTC 频率补偿寄存器	0x0000_0700
TLR	RTC_BA+0x0C	R/W	时间载入寄存器	0x0000_0000
CLR	RTC_BA+0x10	R/W	日历载入寄存器	0x0005_0101
TSSR	RTC_BA+0x14	R/W	时间格式选择寄存器	0x0000_0001
DWR	RTC_BA+0x18	R/W	一周日期寄存器	0x0000_0006
TAR	RTC_BA+0x1C	R/W	时间闹钟寄存器	0x0000_0000
CAR	RTC_BA+0x20	R/W	日历闹钟寄存器	0x0000_0000
LIR	RTC_BA+0x24	R	闰年指示寄存器	0x0000_0000
RIER	RTC_BA+0x28	R/W	RTC 中断使能寄存器	0x0000_0000
RIIR	RTC_BA+0x2C	R/W	RTC 中断指示寄存器	0x0000_0000
TTR	RTC_BA+0x30	R/W	RTC 时钟节拍寄存器	0x0000_0000

5.8.6 寄存器描述

RTC初始化寄存器 (INIR)

寄存器	偏移量	R/W	描述	复位后的值
INIR	RTC_BA+0x00	R/W	RTC 初始化寄存器	0x0000_0000

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

Bits	描述	
[31:0]	INIR	RTC 初始化 芯片上电时，RTC处于复位状态，用户必须写入(0xA5EB1357)到INIR 让RTC控制器退出复位状态。
[0]	Active	RTC 活动状态 (只读) 0: RTC 在复位状态 1: RTC 正常运行

RTC 访问使能寄存器 (AER)

寄存器	偏移量	R/W	描述	复位后的值
AER	RTC_BA+0x04	R/W	RTC 访问使能寄存器	0x0000_0000

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

Bits	描述		
[31:17]	保留	保留	
[16]	ENF	RTC 寄存器访问使能标志 (Read only) 1 = RTC 寄存器读/写使能 0 = RTC 寄存器读/写禁止 对AER[15:0]载入0xA965可对该位置位。512个RTC时钟后，或者当AER[15:0]不等于0xA965时，自动清除。	
		Register	AER.ENF
		1	0
		INIR	R/W
		AER	R/W
		FCR	R/W
		TLR	R/W
		CLR	R/W
		TSSR	R/W
		DWR	R/W

		RIER	R/W	R/W	
		RIIR	R/W	R/W	
		TTR	R/W	-	
[15:0]	AER	RTC 寄存器访问使能密码 (Write only) 0xA965 = 允许RTC读写 Others = 禁止RTC读写			

RTC 频率补偿寄存器 (FCR)

寄存器	偏移量	R/W	描述	复位后的值
FCR	RTC_BA+0x08	R/W	频率补偿寄存器	0x0000_0700

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

Bits	描述			
[31:12]	保留	保留		
[11:8]	INTEGER	整数部分		
		整数部分值	FCR[11:8]	整数部分值
		32776	1111	32768
		32775	1110	32767
		32774	1101	32766
		32773	1100	32765
		32772	1011	32764
		32771	1010	32763
		32770	1001	32762
		32769	1000	32761
[7:6]	保留	保留		
[5:0]	FRACTION	分数部分 公式= (分数部分值) × 60 注: 分数部分必须按照Hex格式数据填入。参考0范例。		

注: RTC访问使能后, 该寄存器值可被读回

RTC 时间载入寄存器 (TLR)

寄存器	偏移量	R/W	描述	复位后的值
TLR	RTC_BA+0x0C	R/W	时间载入寄存器	0x0000_0000

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

Bits	描述	
[31:22]	保留	保留
[21:20]	10HR	10-Hour Time Digit (0~2)
[19:16]	1HR	1-Hour Time Digit (0~9)
[15]	保留	保留
[14:12]	10MIN	10-Min Time Digit (0~5)
[11:8]	1MIN	1-Min Time Digit (0~9)
[7]	保留	保留
[6:4]	10SEC	10-Sec Time Digit (0~5)
[3:0]	1SEC	1-Sec Time Digit (0~9)

注:

1. TLR 为BCD计数方式， RTC不对载入的值进行检测.
2. 括号内列出的为可接受的数值.



RTC 日历载入寄存器 (CLR)

寄存器	偏移量	R/W	描述	复位后的值
CLR	RTC_BA+0x10	R/W	日历载入寄存器	0x0005_0101



Bits	描述	
[31:24]	保留	保留
[23:20]	10YEAR	10-Year Calendar Digit (0~9)
[19:16]	1YEAR	1-Year Calendar Digit (0~9)
[15:13]	保留	保留
[12]	10MON	10-Month Calendar Digit (0~1)
[11:8]	1MON	1-Month Calendar Digit (0~9)
[7:6]	保留	保留
[5:4]	10DAY	10-Day Calendar Digit (0~3)
[3:0]	1DAY	1-Day Calendar Digit (0~9)

注:

1. CLR 为BCD计数方式， RTC不对载入的值进行检测。
2. 括号内列出的为可接受的数值。

RTC 时间格式选择寄存器 (TSSR)

寄存器	偏移量	R/W	描述	复位后的值
TSSR	RTC_BA+0x14	R/W	时间格式选择寄存器	0x0000_0001

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留							
7	6	5	4	3	2	1	0
保留							
24H_12H							

Bits	描述																																																				
[31:1]	保留																																																				
[0]	<p>24H_12H</p> <p>24-小时 / 12-小时模式选择 用于标示TLR 和 TAR 为 24-小时 或12-小时模式 1 = 选择24-小时制时间模式 0 = 选择12-小时制时间模式，带AM /PM 指示</p> <table border="1"> <thead> <tr> <th>24-小时制</th> <th>12-小时制</th> <th>24-小时制</th> <th>12-小时制 (PM 时间+ 20)</th> </tr> </thead> <tbody> <tr> <td>00</td> <td>12(AM12)</td> <td>12</td> <td>32(PM12)</td> </tr> <tr> <td>01</td> <td>01 (AM01)</td> <td>13</td> <td>21 (PM01)</td> </tr> <tr> <td>02</td> <td>02(AM02)</td> <td>14</td> <td>22(PM02)</td> </tr> <tr> <td>03</td> <td>03(AM03)</td> <td>15</td> <td>23(PM03)</td> </tr> <tr> <td>04</td> <td>04 (AM04)</td> <td>16</td> <td>24 (PM04)</td> </tr> <tr> <td>05</td> <td>05(AM05)</td> <td>17</td> <td>25(PM05)</td> </tr> <tr> <td>06</td> <td>06(AM06)</td> <td>18</td> <td>26(PM06)</td> </tr> <tr> <td>07</td> <td>07(AM07)</td> <td>19</td> <td>27(PM07)</td> </tr> <tr> <td>08</td> <td>08(AM08)</td> <td>20</td> <td>28(PM08)</td> </tr> <tr> <td>09</td> <td>09(AM09)</td> <td>21</td> <td>29(PM09)</td> </tr> <tr> <td>10</td> <td>10 (AM10)</td> <td>22</td> <td>30 (PM10)</td> </tr> <tr> <td>11</td> <td>11 (AM11)</td> <td>23</td> <td>31 (PM11)</td> </tr> </tbody> </table>	24-小时制	12-小时制	24-小时制	12-小时制 (PM 时间+ 20)	00	12(AM12)	12	32(PM12)	01	01 (AM01)	13	21 (PM01)	02	02(AM02)	14	22(PM02)	03	03(AM03)	15	23(PM03)	04	04 (AM04)	16	24 (PM04)	05	05(AM05)	17	25(PM05)	06	06(AM06)	18	26(PM06)	07	07(AM07)	19	27(PM07)	08	08(AM08)	20	28(PM08)	09	09(AM09)	21	29(PM09)	10	10 (AM10)	22	30 (PM10)	11	11 (AM11)	23	31 (PM11)
24-小时制	12-小时制	24-小时制	12-小时制 (PM 时间+ 20)																																																		
00	12(AM12)	12	32(PM12)																																																		
01	01 (AM01)	13	21 (PM01)																																																		
02	02(AM02)	14	22(PM02)																																																		
03	03(AM03)	15	23(PM03)																																																		
04	04 (AM04)	16	24 (PM04)																																																		
05	05(AM05)	17	25(PM05)																																																		
06	06(AM06)	18	26(PM06)																																																		
07	07(AM07)	19	27(PM07)																																																		
08	08(AM08)	20	28(PM08)																																																		
09	09(AM09)	21	29(PM09)																																																		
10	10 (AM10)	22	30 (PM10)																																																		
11	11 (AM11)	23	31 (PM11)																																																		

RTC一周日期寄存器 (DWR)

寄存器	偏移量	R/W	描述	复位后的值
DWR	RTC_BA+0x18	R/W	一周日期寄存器	0x0000_0006

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

Bits	描述																	
[31:3]	保留	保留																
[2:0]	DWR	<p>一周日期寄存器</p> <table border="1"> <thead> <tr> <th>值</th> <th>一周日期</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>星期日</td> </tr> <tr> <td>1</td> <td>星期一</td> </tr> <tr> <td>2</td> <td>星期二</td> </tr> <tr> <td>3</td> <td>星期三</td> </tr> <tr> <td>4</td> <td>星期四</td> </tr> <tr> <td>5</td> <td>星期五</td> </tr> <tr> <td>6</td> <td>星期六</td> </tr> </tbody> </table>	值	一周日期	0	星期日	1	星期一	2	星期二	3	星期三	4	星期四	5	星期五	6	星期六
值	一周日期																	
0	星期日																	
1	星期一																	
2	星期二																	
3	星期三																	
4	星期四																	
5	星期五																	
6	星期六																	

RTC 时间闹钟寄存器 (TAR)

寄存器	偏移量	R/W	描述	复位后的值
TAR	RTC_BA+0x1C	R/W	时间闹钟寄存器	0x0000_0000

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

Bits	描述	
[31:22]	保留	保留
[21:20]	10HR	10-Hour Time Digit of Alarm Setting (0~2)
[19:16]	1HR	1-Hour Time Digit of Alarm Setting (0~9)
[15]	保留	保留
[14:12]	10MIN	10-Min Time Digit of Alarm Setting (0~5)
[11:8]	1MIN	1-Min Time Digit of Alarm Setting (0~9)
[7]	保留	保留
[6:4]	10SEC	10-Sec Time Digit of Alarm Setting (0~5)
[3:0]	1SEC	1-Sec Time Digit of Alarm Setting (0~9)

注:

1. TAR 为BCD计数方式， RTC不对载入的值进行检测。
2. 括号内列出的为可接受的数值。
3. RTC单位使能后，该寄存器值可被读回。

RTC 日历闹钟寄存器 (CAR)

寄存器	偏移量	R/W	描述	复位后的值
CAR	RTC_BA+0x20	R/W	日历闹钟寄存器	0x0000_0000

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

Bits	描述	
[31:24]	保留	保留
[23:20]	10YEAR	10-Year Calendar Digit of Alarm Setting (0~9)
[19:16]	1YEAR	1-Year Calendar Digit of Alarm Setting (0~9)
[15:13]	保留	保留
[12]	10MON	10-Month Calendar Digit of Alarm Setting (0~1)
[11:8]	1MON	1-Month Calendar Digit of Alarm Setting (0~9)
[7:6]	保留	保留
[5:4]	10DAY	10-Day Calendar Digit of Alarm Setting (0~3)
[3:0]	1DAY	1-Day Calendar Digit of Alarm Setting (0~9)

注:

1. CAR 为BCD计数方式， RTC不对载入的值进行检测。
2. 括号内列出的为可接受的数值。
3. RTC单位使能后， 该寄存器值可被读回

RTC 闰年指示寄存器 (LIR)

寄存器	偏移量	R/W	描述	复位后的值
LIR	RTC_BA+0x24	R	RTC 闰年指示寄存器	0x0000_0000

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

Bits	描述	
[31:1]	保留	保留
[0]	LIR	闰年指示寄存器 (Real only). 1 = 表示本年为闰年 0 = 表示本年非闰年

RTC 中断使能寄存器 (RIER)

寄存器	偏移量	R/W	描述	复位后的值
RIER	RTC_BA+0x28	R/W	RTC 中断使能寄存器	0x0000_0000

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

Bits	描述	
[31:2]	保留	保留
[1]	TIER	时钟节拍中断使能 1 = RTC 时钟节拍中断使能 0 = RTC 时钟节拍中断禁止
[0]	AIER	闹钟中断使能 1 = RTC 闹钟中断使能 0 = RTC 闹钟中断禁止



RTC 中断指示寄存器 (RIIR)

寄存器	偏移量	R/W	描述	复位后的值
RIIR	RTC_BA+0x2C	R/W	RTC中断指示寄存器	0x0000_0000

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

Bits	描述	
[31:2]	保留	保留
[1]	TIF	RTC 时钟节拍中断标志 当RTC 时钟节拍中断使能, (RIER.TIER=1), RTC根据TTR[2:0]设定的周期性使TIF置1, 该位通过软件清除。 1= 如果RIER.TIER=1, 表示有RTC时钟节拍中断请求 0= 没有中断请求.
[0]	AIF	RTC 闹钟中断标志 当RTC 闹钟中断使能, (RIER.AIER=1), RTC 根据TLR和CLR是否达到TAR 和 CAR的设定对AIF置1。该位通过软件清除。 1= 如果RIER.AIER=1, 表示有RTC闹钟中断请求 0= 没有中断请求.

RTC 时钟节拍寄存器 (TTR)

寄存器	偏移量	R/W	描述	复位后的值
TTR	RTC_BA+0x30	R/W	RTC时钟节拍寄存器	0x0000_0000

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

Bits	描述																			
[31:4]	保留	保留																		
[3]	TWKE	<p>RTC 定时器唤醒使能 如果在芯片进入掉电模式之前，TWKE 被置位，当RTC或闹钟匹配发生时，芯片被RTC控制器唤醒。 1= 使能将芯片由掉电模式唤醒。 0= 禁止唤醒功能. 注: 1. 时间计数根据 TTR[2:0] 描述配置</p>																		
[2:0]	TTR	<p>时钟节拍寄存器 根据TTR寄存器设定产生中断。注: RTC 访问使能后，该寄存器值可被读回.</p> <table border="1"> <tr> <td>TTR[2:0]</td> <td>时钟节拍(秒)</td> </tr> <tr> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1/2</td> </tr> <tr> <td>2</td> <td>1/4</td> </tr> <tr> <td>3</td> <td>1/8</td> </tr> <tr> <td>4</td> <td>1/16</td> </tr> <tr> <td>5</td> <td>1/32</td> </tr> <tr> <td>6</td> <td>1/64</td> </tr> <tr> <td>7</td> <td>1/128</td> </tr> </table>	TTR[2:0]	时钟节拍(秒)	0	1	1	1/2	2	1/4	3	1/8	4	1/16	5	1/32	6	1/64	7	1/128
TTR[2:0]	时钟节拍(秒)																			
0	1																			
1	1/2																			
2	1/4																			
3	1/8																			
4	1/16																			
5	1/32																			
6	1/64																			
7	1/128																			

5.9 串行外围设备接口 (SPI)

5.9.1 概述

SPI 接口是工作于全双工模式下的同步串行数据传输接口。包括2组SPI控制器，将从外设得到的数据进行串并转换，或将数据进行并串转换，发送到外设。每组SPI控制器可以被作为一个主机，驱动最多2个外部从设备；还可以被设置为外围设备的从机。

该控制器支持不同串行时钟以适应不同应用。

5.9.2 特征

- 支持二组SPI传送控制器
- 支持主/从机模式
- 支持1位传输模式
- 最大传输字可达32位，一次最多可传输2个字，即一次最多可传输64位数据
- 支持burst操作模式，在一次传输过程中，发送/接收可执行两次字传输
- 支持MSB 或 LSB 为最先传输模式
- 主机模式下有2条设备/从机选择线，从机模式下有1条设备/从机选择线
- 支持数据寄存器字节重排序
- 支持字节或字休眠模式
- 主机模式下，支持不同输出串行时钟频率
- 主机模式下支持两个可编程的串行时钟频率
- 支持FIFO 模式

5.9.3 框图

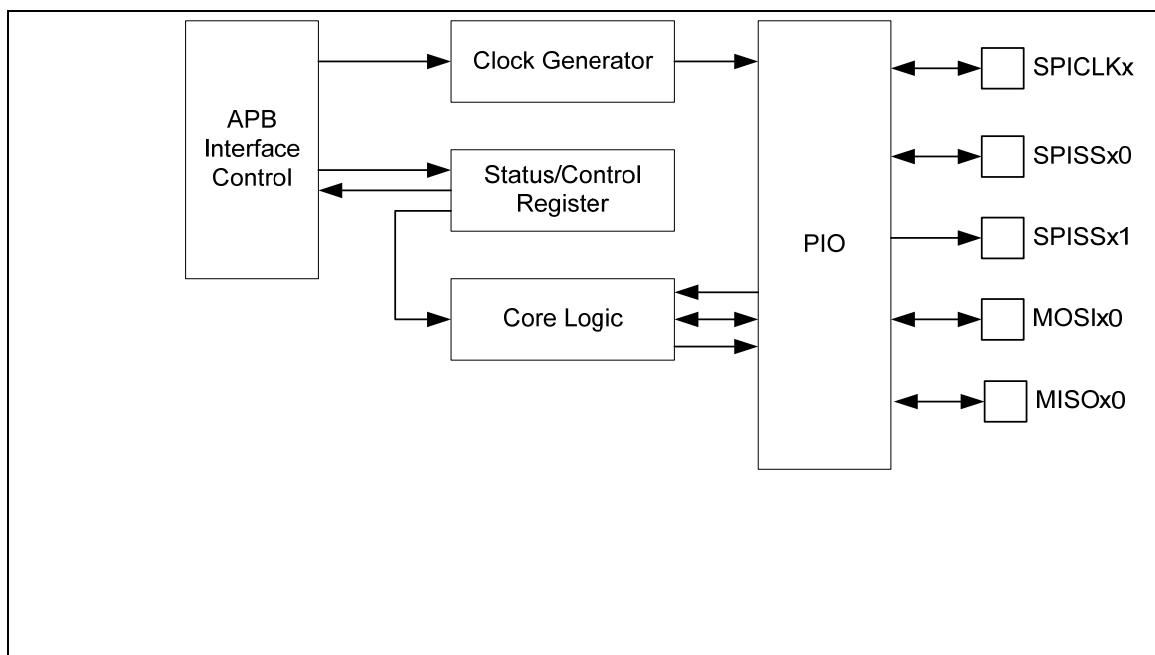


图 5-42 SPI 框图

5.9.4 功能描述

主/从模式

SPI控制器可通过设置SLAVE 位(SPI_CNTRL[18])，配置为主机或从机模式，与外设从机或主机通信。当配置为从机模式时，SPI总线不支持多从机运作，当SPISSx引脚处于无效状态时，SPICLKx引脚必须为空闲状态。主机模式与从机模式的应用框图如下：

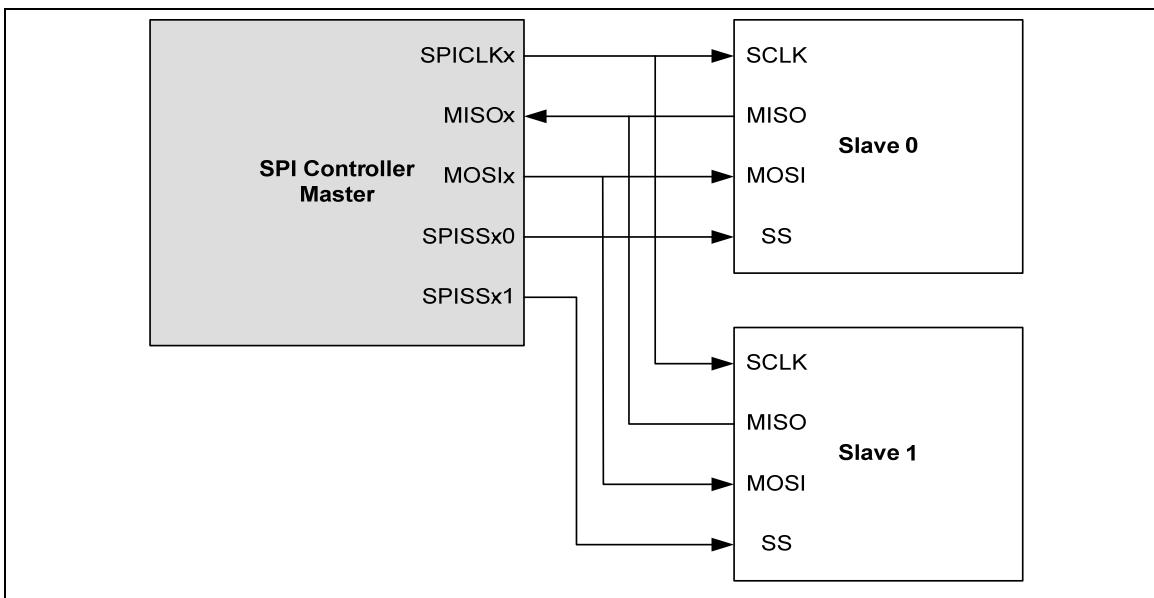


图 5-43 SPI 主机模式应用框图

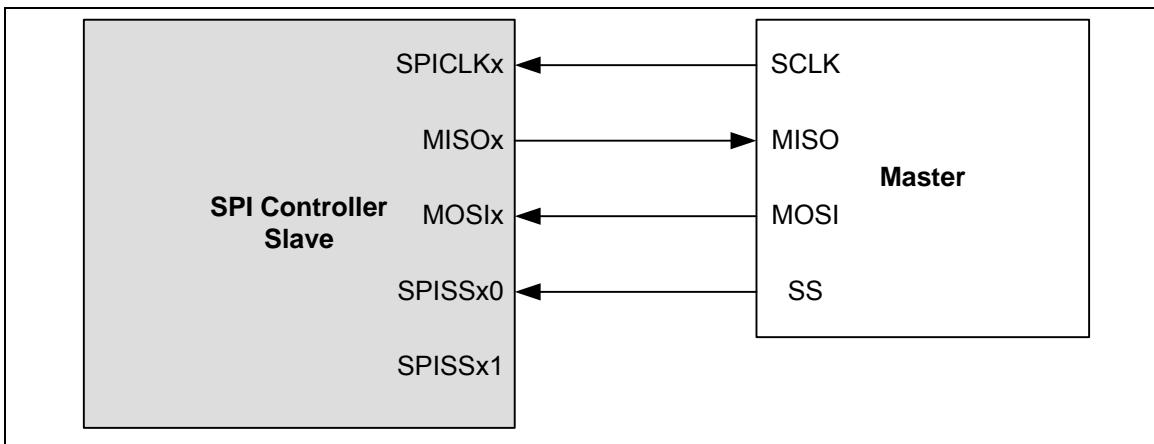


图 5-44 SPI 从机模式应用框图

从机选择

在主机模式下，SPI控制器能通过从机选择输出脚SPISSx0 与 SPISSx1选择两个从机外围设备，从机模式下，外围的主机设备驱动从机信号是通过SPISSx0 输入到SPI控制器的。在主机/从机模式下，从机选择信号的有效电平可以在SS_LVL 位 (SPI_SSR[2])被编程为低有效或高有效，SS_LTRIG 位(SPI_SSR[4])定义从机选择信号SPISSx0/1 为电平触发还是边缘触发。触发条件的选

择取决于所连接的从机/主机的外设类型.

从模式下, 如果SS_LTRIG配置成电平触发, 在一次发送结束, 或者接收结束时, LTRIG_FLAG 位(SPI_SSR[5]) 用于标志接收数目与接收位是否满足TX_NUM 和 TX_BIT_LEN设定的要求.

电平触发/边沿触发

从机模式下, 从机选择信号可配置成电平触发或边沿触发. 当为边沿触发时, 数据从检测到有效边沿时刻开始传输, 在检测到无效边沿信号时结束. 如果主机不发送无效边沿信号给从机, 传输过程就不能完成, 从机的中断标志位也不能置位。当为电平触发时, 有两个条件可以中止传输, 并置位中断标志位. 第一个条件, 如果主机设置从机选择脚为无效电平, 从机将中止当前传输, 并置位中断标志位. 用户可以读LTRIG_FLAG检查数据是否完全传输. 第二个条件是如果传输数目与TX_NUM 和 TX_BIT_LEN的设置相匹配, 从机中断将置位。

自动从机选择

在主机模式下, 如果置位AUTOSS (SPI_SSR[3]), 将自动产生从机选择信号, 并根据SSR[0] (SPI_SSR[0]) 与 SSR[1] (SPI_SSR[1])是否使能, 将从机选择信号输出到SPISSx0与SPISSx1引脚上. 意思是, 在通过设置GO_BUSY 位 (SPI_CNTRL[0])使发送/接收开始时, 被SSR[1:0]选择的从机选择信号引脚将由SPI控制器设置为有效状态, 而当数据传输结束时, SPI控制器会自动将从机选择信号引脚设置为无效状态。当ASS位清零时, 可以通过软件自行设置或清零SPI_SSR[1:0]的相关位来决定从机选择引脚输出为有效状态或无效状态。在SS_LVL 位 (SPI_SSR[2])定义了从机选择输出信号的有效(active)状态.

串行时钟

在主机模式下, 向寄存器DIVIDER1 (SPI_DIVIDER[15:0])写入除数来编程串行时钟的输出频率到SPICLK输出口。如果VARCLK_EN bit (SPI_CTL[23])使能, 也支持可调串行时钟. 此时, 串行时钟的输出频率可依据的DIVIDER1 (SPI_DIVIDER[15:0]) 和 DIVIDER2 (SPI_DIVIDER[31:16])值被编程为两种不同频率的一种. 个别串行时钟的周期取决于寄存器SPI_VARCLK的配置.

在从机模式下, 外设主机设备通过SPICLK引脚输入串行时钟到SPI控制器

可调串行时钟频率

主机模式下，如果使能VARCLK_EN (SPI_CNTRL[23])，串行时钟的输出频率可随着VARCLK寄存器的配置而变化。如果VARCLK的该位为‘0’，输出频率取决于DIVIDER (SPI_DIVIDER[15:0])，如果VARCLK 的该位‘1’，输出频率取决于DIVIDER2 (SPI_DIVIDER[31:16])。图 5-45 为 (SPICLK), VARCLK, DIVIDER 和DIVIDER2的时序关系。VARCLK两位决定一个时钟周期。VARCLK[31:30]定义了SPICLK的第一个时钟周期。VARCLK[29:28] 定义了SPICLK的第二个时钟周期。VARCLK中定义时钟源时必须在切换为另一个时钟源的前一周期被设置。例如，如果SPICLK有5个 CLK1 周期，VARCLK 将在VARCLK的MSB设置为9个 ‘0’。第10个位 设置为 ‘1’以切换下一个时钟源为CLK2。注意当使能VARCLK_EN 位，TX_BIT_LEN 必须设置成0x10 (仅16位模式)。

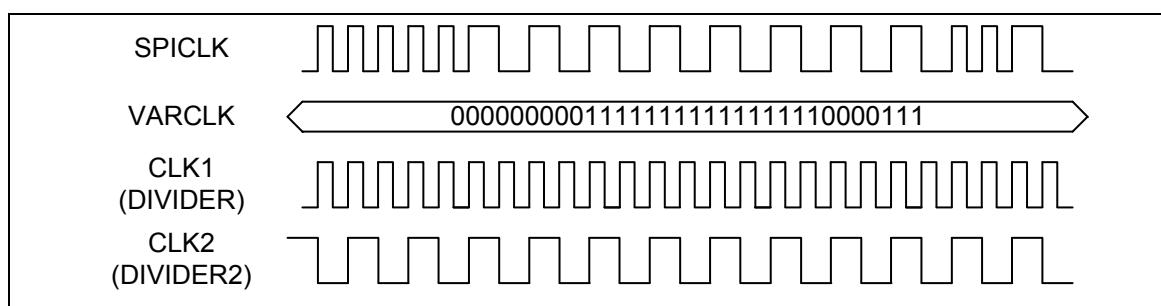


图 5-45 可调串行时钟频率

时钟极性

CLKP 位 (SPI_CNTRL[11]) 定义串行时钟的空闲状态。当 CLKP = 1，输出SPICLK为空闲高电平状态，否则CLKP = 0时，输出SPICLK为空闲低电平状态。

发送/接收位长度

传输字的长度在Tx_BIT_LEN 位(SPI_CNTRL[7:3])中定义。发送和接收时传输字可达32位长度。

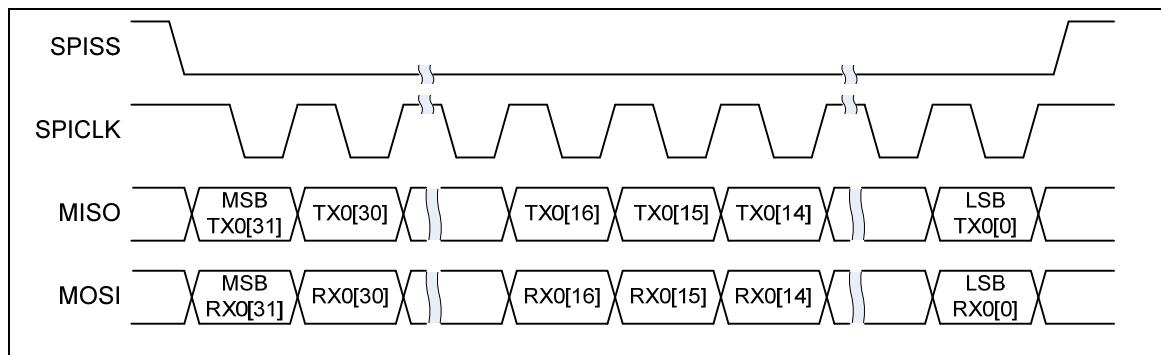


图 5-46 32-Bit in one Transaction

Burst 模式

SPI 控制器通过设置 TX_NUM bit field (SPI_CNTRL[9:8]) 为 0x01 切换到突发模式。突发模式下，SPI 可以一次发送/接收两个报文。SPI 突发波形如下：

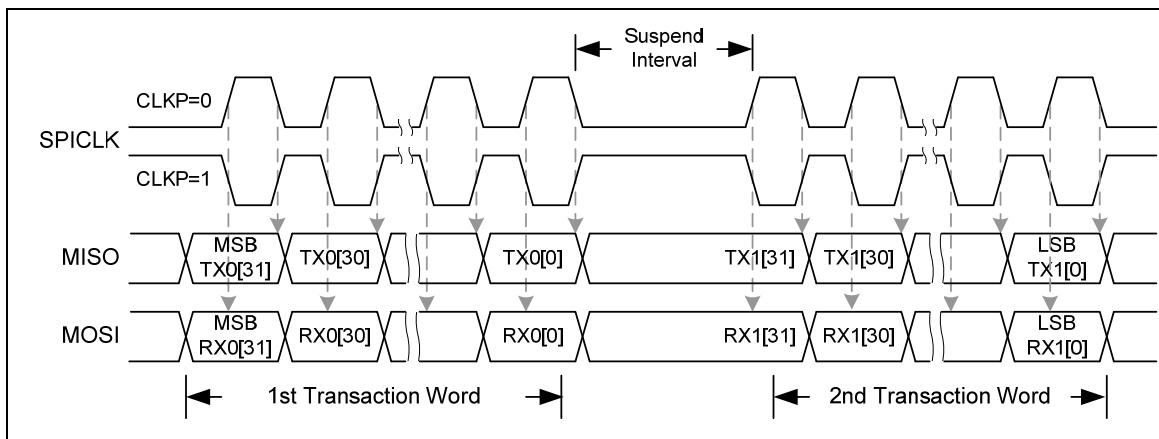


图 5-47 一次传输两个报文(Burst Mode)

LSB First

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

发送边界

Tx_NEG 位 (SPI_CNTRL[2]) 定义数据发送是在串行时钟 SPICLK 的负边缘还是正边缘。

接收边界

Rx_NEG 位 (SPI_CNTRL[1]) 定义数据接收是在串行时钟 SPICLK 的负边缘还是正边缘。

注：TX_NEG 与 RX_NEG 不可设为相同。

字休眠

在主机模式下，SLEEP (SPI_CNTRL[15:12]) 的 4 位可配置 2~17 个串行时钟周期为两个连续传输字之间的休眠间隔。如果 CLKP = 0，休眠间隔指从前一次传输字的最后一个下降时钟沿到下一次传输字的第一个上升时钟沿，如果 CLKP = 1，间隔时间为前一次传输字的最后一个上升沿到下一次传输字的第一个下降沿。SLEEP 的缺省值为 0x0 (2 个串行时钟周期)，如果 Tx_NUM = 0x00 时，设置这些位不会影响数据传输过程。

Byte 重排序

当设置为MSB优先时（LSB = 0），且使能REORDER，在Tx_BIT_LEN = 32 位模式时，存储在TX缓存与RX缓存的数据将重新按[BYTE0, BYTE1, BYTE2, BYTE3]的顺序排列，数据将以BYTE0, BYTE1, BYTE2, BYTE3的顺序发送/接收。在Tx_BIT_LEN = 24 位模式时，存储在TX缓存与RX缓存的数据将重新按[unknown byte, BYTE0, BYTE1, BYTE2]的顺序排列，数据将以BYTE0, BYTE1, BYTE2的顺序发送接收。16位模式与上面相同。

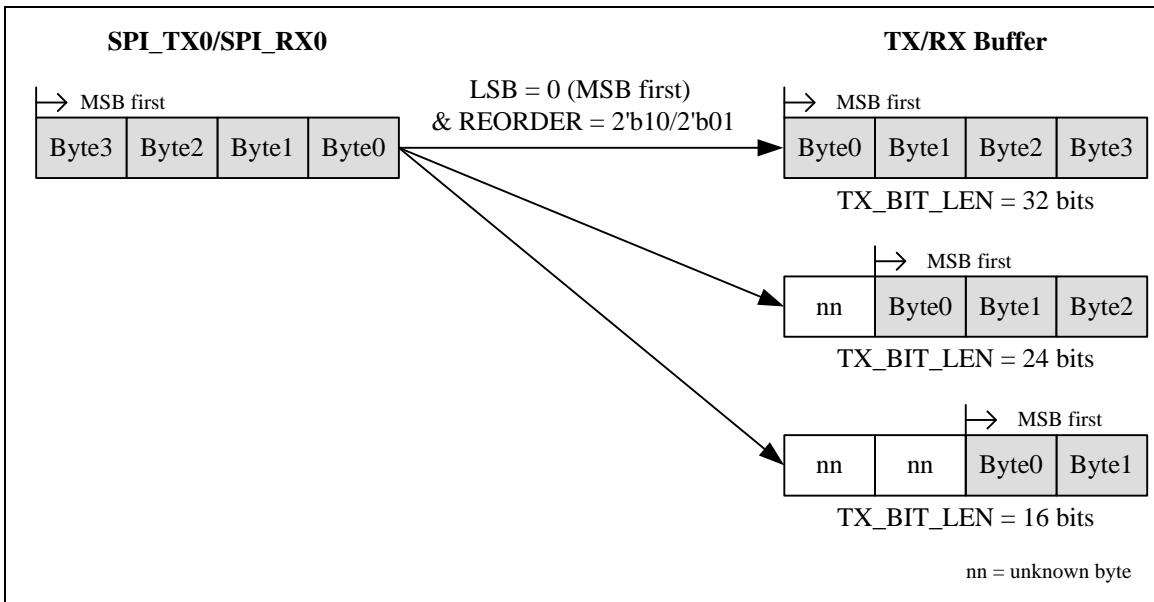


图 5-48 Byte 重排序

字节休眠

主机模式下，设置(SPI_CNTRL[19])为1，硬件将在两个连续传输字节之间插入2~17个串行时钟周期的休眠间隔。字节休眠的设置与字休眠的设置一样都是用SP_CYCLE，注意当使能BYTE_SLEEP, Tx_BIT_LEN 必须设置为0x00 (32 bits per transfer word)。

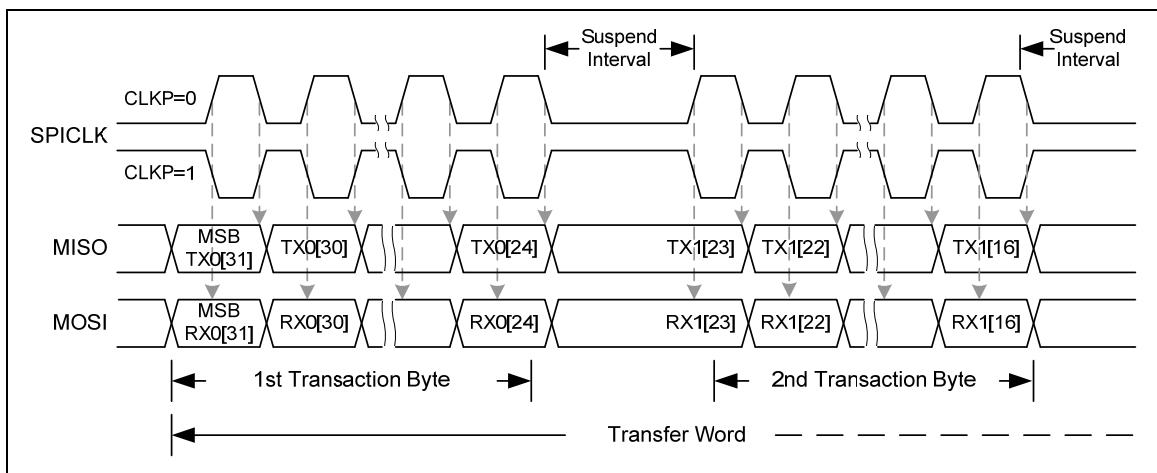


图 5-49 字节休眠时序波形

REORDER	描述
00	禁止字节重排序功能和字节休眠
01	使能字节重排序功能并在每个字节之间插入一个(2~17 SPICLK)的休眠间隔。必须设置TX_BIT_LEN 为 0x00 (32 bits/ word)
10	使能字节重排序功能，禁止字节休眠功能
11	禁止字节重排序功能，在每个字节之间插入休眠间隔 (2~17 SPICLK)。必须设置TX_BIT_LEN 为0x00 (32 bits/ word)

表 5-6 字节排序和字节休眠条件

中断

数据传输完毕时，每一个SPI控制器会产生独立的中断，相应的中断标志IF (SPI_CNTRL[16])被置位，如果IE (SPI_CNTRL[17]) 使能，则中断事件标志将向CPU提出中断请求。中断标志可由软件写1清零

FIFO 模式

SPI_CNTRL[21]置1时，SPI支持FIFO模式。若不用FIFO，每次要在发送完成后才可以再写发送缓冲器SPI_TX0。使用FIFO，可提前将下个数据写入SPI_TX0，GO_BUSY位由硬件控制，软件改变GO_BUSY位，将导致发送错误。

从模式时，若位FIFO置1，GO_BUSY位由硬件自动置1。若要停止从模式数据发送，必须把位FIFO和位GO_BUSY都清0。

在FIFO置位之前，必须先写一个数到发送缓冲器SPI_TX0/1，TX_EMPTY会清0，FIFO置位后会立即启动发送，数据被送到TX0/1。这时可以再写入下一个待发数据到SPI_TX0/1，SPI控制器会在两个字节间插入空闲，只要TX_FULL为0，就可以继续往发送缓冲器里写数据。

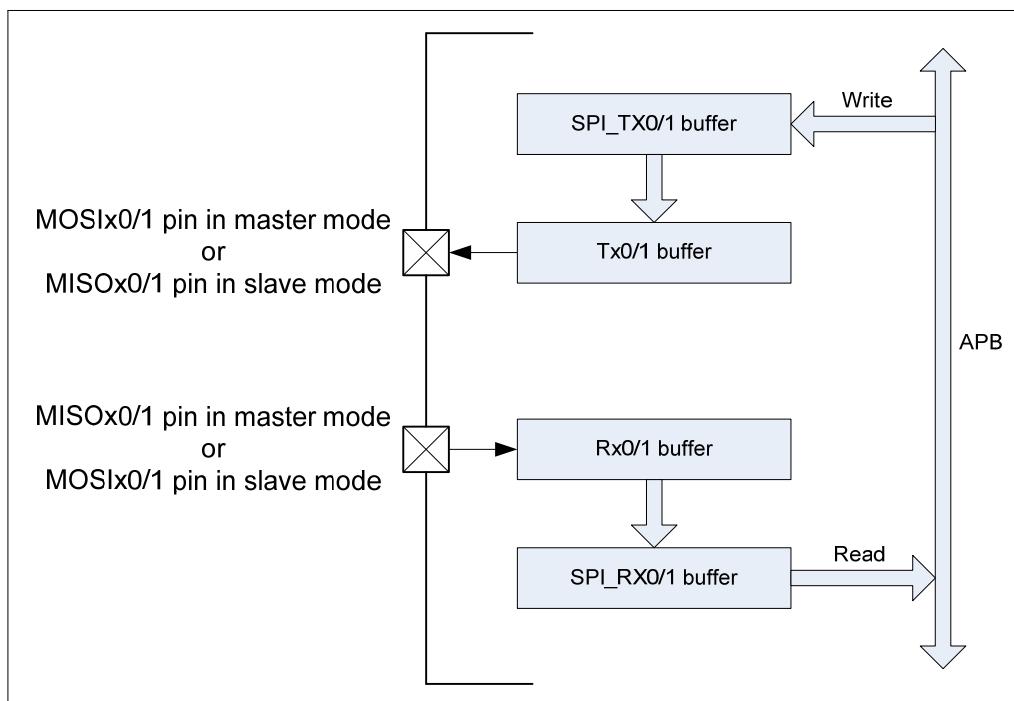


图 5-50 FIFO 模式框图

主模式接收时，若收到一个数据，且RX_EMPTY为1，收到放入Rx0/1的数据就会放入SPI_RX0/1，并且RX_EMPTY清0，可从SPI_RX0/1中读取收到的数据。

注意：主模式使用FIFO时，除了清0 FIFO位之外，不要改变SPI_CNTRL的值。且要SPI_CNTRL的IE位要清0，不用中断方式收发，而根据RX_EMPTY,RX_FULL,TX_EMPTY,TX_FULL对收发缓冲器进行读写

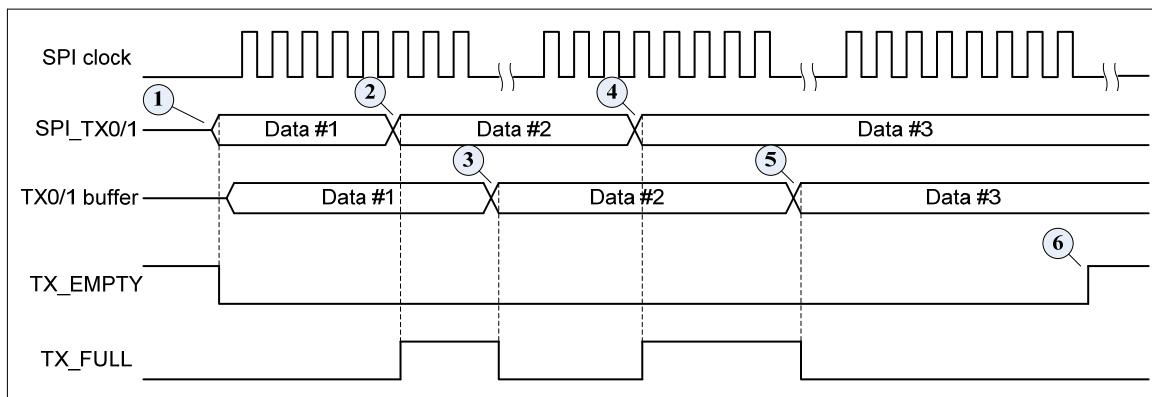


图 5-51 FIFO 模式时序图

时序图

从机选择信号(SPISSx)的有效状态可以在SS_LVL 位 (SPI_SSR[2])及SS_LTRIG 位 (SPI_SSR[4])的配置中决定.串行时钟 (SPICLK)的空闲状态可以通过CLKP 位 (SPI_CNTRL[11])配置为高电平或低电平. 在Tx_BIT_LEN (SPI_CNTRL[7:3])中定义传输字的长度, 在Tx_NUM (SPI_CNTRL[8])中定义传输的数目, 在LSB (SPI_CNTRL[10])中定义发送/接收数据是以MSB或LSB优先. Tx_NEG/Rx_NEG(SPI_CNTRL[2:1])可以设置用户选择发送/接收数据时串行时钟的边沿. 四个SPI时序图及相关设置如下.

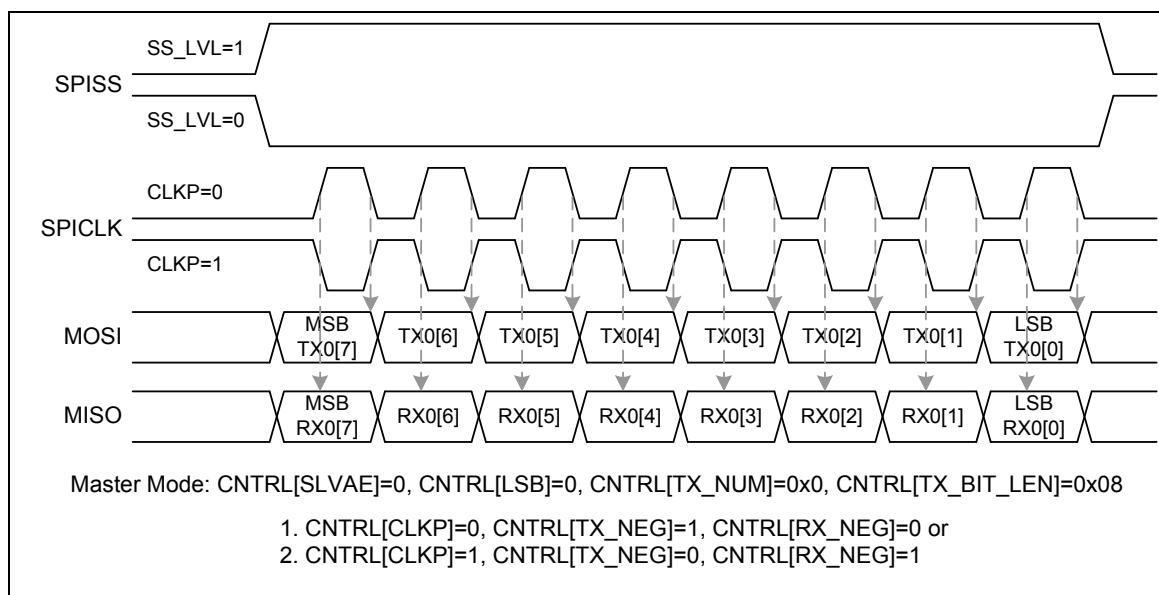


图 5-52 SPI 在主机模式下的时序

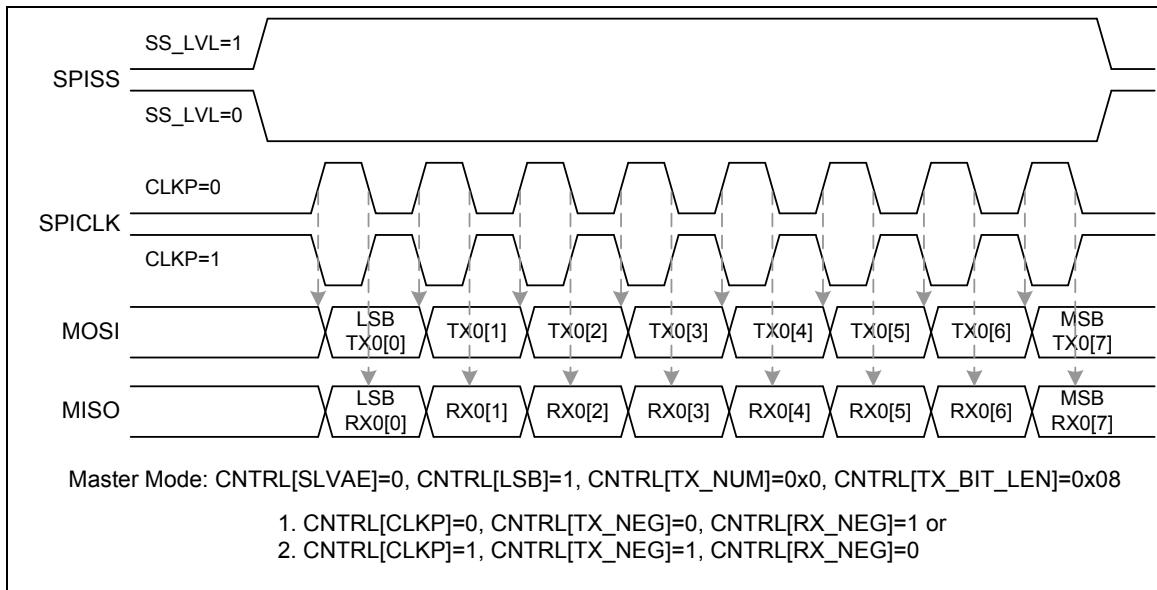


图 5-53 SPI 主机模下的时序 (Alternate Phase of SPICLK)

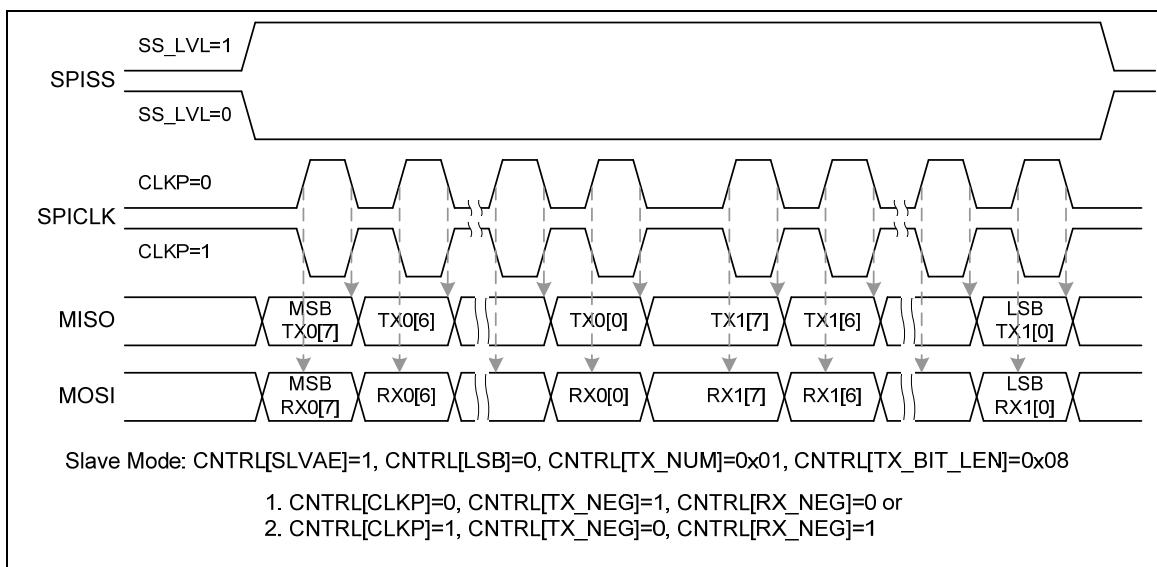


图 5-54 SPI 在从机模式下的时序

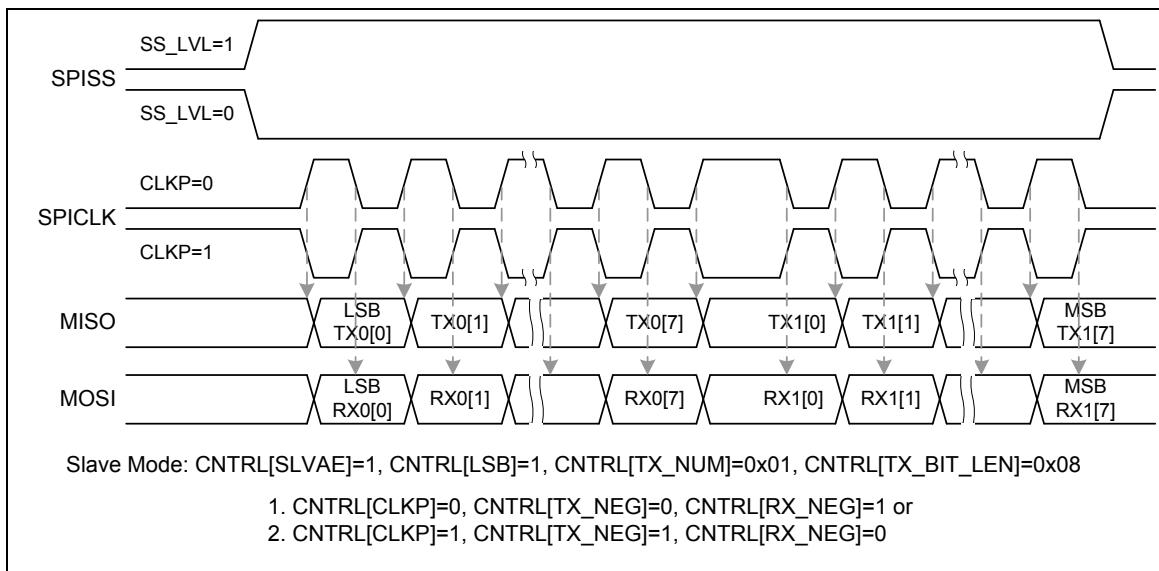


图 5-55 SPI 在从机模式下的时序 (Alternate Phase of SPICLK)

5.9.5 编程例程

例 1, SPI作为主机时，与一个从机设备通信如下：

- 数据在时钟上升沿锁存
- 数据在时钟下将沿传输
- MSB先传送
- SPICLK 空闲模式为低电平状态
- 每次输出/接收一个字节
- 使用第一个 SPI 从机选择引脚和一个片外从机相连。从机选择信号的有效状态为低电平

操作流程如下：

- 1) 设定寄存器DIVIDER 的SPI_DIVIDER[15:0]来决定串行时钟输出频率.
- 2) 将主机模式的相应设置写入SPI_SSR寄存器,
 1. 本例中禁止自动从机选择位ASS (SPI_SSR[3] = 0),
 2. 设置SS_LVL (SPI_SSR[2] = 0) 和 SS_LTRIG (SPI_SSR[4] = 1)使从机选择信号为低电平触发输出.
 3. 设置相关从机选择寄存器SSR[0] (SPI_SSR[0])激活片外从机设备，选择哪个从机选择信号将被输出到IO引脚上
- 3) 将主机模式的相应设置写入寄存器SPI_CNTRL.
 1. 通过SLAVE 位 (SPI_CNTRL[18] = 0)设置SPI控制器为主机设备
 2. 通过CLKP 位 (SPI_CNTRL[11] = 0) 设置串行时钟空闲状态为低电平
 3. 通过TX_NEG 位 (SPI_CNTRL[2] = 1)选择数据在串行时钟的负边沿传输
 4. 通过RX_NEG 位 (SPI_CNTRL[1] = 0)选择数据在串行时钟的正边沿锁存
 5. 通过TX_BIT_LEN 位域 (SPI_CNTRL[7:3] = 0x08)设置传输字的长度为8位
 6. 通过 TX_NUM 位域 (SPI_CNTRL[9:8] = 0x0) 设置为一次字传输
 7. 通过 LSB 位 (SPI_CNTRL[10] = 0) 设置为 MSB 传输优先，不必理会 SP_CYCLE 位域 (SPI_CNTRL[15:12])的设置，因为在本例中没有使用burst 模式
- 4) 如果SPI主机是要发送一个字节的数据到外设，则将所要发送的数据写入寄存器TX0[7:0] (SPI_TX0[7:0]).
- 5) 如果SPI主机只是要从外设接收一个字节的数据，不必管被传输出去的数据是什么，只需要向寄存器SPI_TX0[7:0]写入0XFF.
- 6) 使能GO_BUSY 位(SPI_CNTRL[0] = 1)，以开始 SPI 接口的数据传输。
- 7) 等到SPI中断发生 (IE使能)，或检测GO_BUSY 位直到被硬件自动清零.
- 8) 从寄存器RX0[7:0] (SPI_RX0[7:0])读出所接收到的一个字节的数据.

- 9) 重复步聚 4) 继续其他数据的传输或设置SSR[0] 为0 以停止外设.

例 2, SPI 控制器作为从机设备, 由外设控制, 外设通过SPI接口与片上 SPI 从机通信:

- 数据在时钟上升沿锁存
- 数据在时钟下降沿传输
- LSB先传送
- SPICLK空闲模式为高电平
- 每次输出/接收一个字节
- 从机选择信号为高电平触发

操作流程如下:

- 1) 设置从机有效电位 SS_LVL (SPI_SSR[2] = 1) 与从机选择电平触发位 SS_LTRIG (SPI_SSR[4] = 1)使从机选择信号为高电平触发.
- 2) 将从机模式的相应设置写入寄存器SPI_CNTRL.
 1. 通过SLAVE 位 (SPI_CNTRL[18] = 1)设置SPI控制器为从机设备
 2. 通过CLKP 位 (SPI_CNTRL[11] = 1)选择串行时钟空闲状态
 3. 通过 TX_NEG 位 (SPI_CNTRL[2] = 1)选择数据传输发生在串行时钟负边沿
 4. 通过RX_NEG 位 (SPI_CNTRL[1] = 0)选择数据锁存在串行时钟的正边沿
 5. 通过TX_BIT_LEN 位域 (SPI_CNTRL[7:3] = 0x08) 设置字传输长度为 8 位
 6. 通过TX_NUM 位(SPI_CNTRL[9:8] = 0x0)设置为仅一次字传输
 7. 通过 LSB 位 (SPI_CNTRL[10] = 1) 设置为LSB传输优先 , 不必理会SP_CYCLE 位域 (SPI_CNTRL[15:12])的设置, 因为在本例中没有使用burst模式
- 3) 如果SPI从机是要发送一个字节的数据到外设主机, 则将所要发送的数据写入到寄存器 TX0[7:0] (SPI_Tx0[7:0]).
- 4) 如果SPI从机仅从只是要外设主机接收一字节数据, 用户不必关心什么数据将被传输, 只需要向寄存器SPI_Tx0[7:0]写入0XFF.
- 5) 使能GO_BUSY bit (SPI_CNTRL[0] = 1) , 等到外设的从机选择触发输入和串行时钟输入, 开始数据传输到SPI接口.
- 6) 等到SPI中断发生 (IE使能) , 或检测GO_BUSY 位直到被硬件自动清零.
- 7) 在寄存器RX[7:0] (SPI_RX0[7:0])读出所接收到的一个字节的数据.
- 8) 重复步聚3) 继续其他数据传输或禁用GO_BUSY 位停止数据传输.

5.9.6 寄存器映射

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

寄存器	偏移量	R/W	描述	复位后的值
SPI0_BA = 0x4003_0000				
SPI1_BA = 0x4003_4000				
SPI_CTRL	SPIx_BA+0x00	R/W	控制及状态寄存器	0x0500_0004
SPI_DIVIDER	SPIx_BA+0x04	R/W	时钟除频寄存器	0x0000_0000
SPI_SSR	SPIx_BA+0x08	R/W	从机选择寄存器	0x0000_0000
SPI_RX0	SPIx_BA+0x10	R	接收数据寄存器0	0x0000_0000
SPI_RX1	SPIx_BA+0x14	R	接收数据寄存器1	0x0000_0000
SPI_TX0	SPIx_BA+0x20	W	数据发送寄存器0	0x0000_0000
SPI_TX1	SPIx_BA+0x24	W	数据发送寄存器1	0x0000_0000
SPI_VARCLK	SPIx_BA+0x34	R/W	可调时钟类型寄存器	0x007F_FF87

注 1: 由软件编写 CNTRL 寄存器, GO_BUSY 位必须最后写入.

5.9.7 寄存器描述

SPI 控制与状态寄存器 (SPI_CNTRL)

寄存器	偏移量	R/W	描述	复位后的值
SPI_CNTRL	SPIx_BA+0x00	R/W	控制与状态寄存器	0x0500_0004

31	30	29	28	27	26	25	24
保留				TX_FULL	TX_EMPTY	RX_FULL	RX_EMPTY
23	22	21	20	19	18	17	16
VARCLK_EN	保留	FIFO	REORDER		SLAVE	IE	IF
15	14	13	12	11	10	9	8
SP_CYCLE				CLKP	LSB	TX_NUM	
7	6	5	4	3	2	1	0
TX_BIT_LEN					TX_NEG	RX_NEG	GO_BUSY

Bits	描述	
[31:28]	保留	保留
[27]	TX_FULL	发送数据缓冲器满状态标志 (只读) 1 = 指示发送数据缓冲器满 0 = 指示发送数据缓冲器非满
[26]	TX_EMPTY	发送数据缓冲器空状态标志 (只读) 1 = 指示发送数据缓冲器空 0 = 指示发送数据缓冲器非空
[25]	RX_FULL	接收数据缓冲器满状态标志 (只读) 1 = 指示接收数据缓冲器满 0 = 指示接收数据缓冲器不满
[24]	RX_EMPTY	接收数据缓冲器空状态标志 (只读) 1 = 指示接收数据缓冲器空 0 = 指示接收数据缓冲器非空
[23]	VARCLK_EN	可调时钟使能 (Master only) 1 = 串行时钟输出频率是可调的. 输出频率由VARCLK, DIVIDER, 和 DIVIDER2 的值决定. 0 = 串行时钟输出频率固定, 只由DIVIDER的值决定.

		注：当使能VARCLK_EN 位, TX_BIT_LEN 必须设置为0x10 (16 bits mode)
[22]	保留	必须保持为 0
[21]	FIFO	FIFO 模式 1 = 使能FIFO模式 0 = 不使能FIFO模式
[20:19]	REORDER	重排序模式选择 00 = 禁用字节重排序和字节休眠功能. 01 = 使能字节重排序功能, 在每个字节之间插入一个字节的休眠间隔(2~17 SPICLK cycles). 必须设置 TX_BIT_LEN 为 0x00. (32 bits/word) 10 = 使能字节重排序功能, 禁用字节休眠功能. 11 = 禁用字节重排序功能, 在每个字节之间插入一个字节的休眠间隔(2~17 SPICLK cycles). 必须设置 TX_BIT_LEN 为 0x00. (32 bits/word) 註： 1. 当使能字节重排功能时, TX_BIT_LEN必须设置为 16, 24或32 2. 当运作于从机模式且从机选择设置为电平触发时, 若使能字节休眠功能, 主机输出的从机选择信号在连续的四个字节传输过程中, 必须维持在 active 状态
[18]	SLAVE	从机模式 1 = 从机模式 0 = 主机模式
[17]	IE	中断使能 1 = 使能SPI 中断 0 = 禁用SPI 中断
[16]	IF	中断标志 1 = 表示传输完成 0 = 表示传输没有完成. 注：该位写1清零.
[15:12]	SP_CYCLE	休眠间隔 (Master only) 这四位用于配置休眠间隔. 如果CLKP = 0, 休眠间隔是指从当前传输的最后一次下降时钟沿到连续传输的第一次上升时钟沿. 如果CLKP = 1, 休眠间隔则指从上升时钟沿到下降时钟沿. 默认值为0x0. 当 TX_NUM = 00b, 设置该位对传输没有影响. A: 突发模式 $(SP_CYCLE[3:0] + 2) * \text{SPI 时钟周期}$ 例如： $SP_CYCLE = 0x0 \dots 2 \text{ 个SPI 时钟周期}$

		<p>SP_CYCLE = 0x1 ... 3个SPI 时钟周期 SP_CYCLE = 0xE ... 16个SPI 时钟周期 SP_CYCLE = 0xF ... 17个SPI 时钟周期 B: FIFO模式</p> <p>1. SP_CYCLE 不能设置为 1. 2. 如果 SP_CYCLE = 2 ~ 15,挂起时间为 (SP_CYCLE[3:0] + 3) *系统时钟周期 + 1个SPI 时钟周期 比如: SP_CYCLE = 0x2 ... 5系统时钟周期 + 1个SPI 时钟周期 SP_CYCLE = 0xE ... 17系统时钟周期 + 1个SPI 时钟周期 SP_CYCLE = 0xF ... 18系统时钟周期 + 1个SPI 时钟周期</p> <p>3. 如果 SP_CYCLE = 0,挂起时间等于 35 *系统时钟周期 + 1个SPI 时钟周期</p>
[11]	CLKP	<p>时钟极性</p> <p>1 = 空闲时SCLK 低电平. 0 = 空闲时SCLK 高电平</p>
[10]	LSB	<p>优先传送 LSB</p> <p>1 =优先发送 LSB (SPI_TX0/1的比特0), 并且接收到的第一个放到 Rx 寄存器的LSB位置 (SPI_RX0/1的比特0). 0 =优先发送/接收MSB (根据CNTRL寄存器的Tx_BIT_LEN 决定SPI_TX0/1和 SPI_RX0/1的第一个位的編號).</p>
[9:8]	TX_NUM	<p>发送/接收数量</p> <p>该寄存器用于标示一次成功传输中，传输的数量。</p> <p>00 = 每次传输仅完成一次发送/接收 01 = 每次传输完成两次发送/接收 10 = 保留. 11 = 保留.</p> <p>注 当运作于从机模式且从机选择设置为电平触发时，若TX_NUM设置为01主机输出的从机选择信号在连续的两个传输过程中，必须维持在active状态</p>
[7:3]	TX_BIT_LEN	<p>传输位长度</p> <p>该寄存器用于标示一次传输中，完成的传输长度，最高纪录32位.</p> <p>TX_BIT_LEN = 0x01 ... 1 bit TX_BIT_LEN = 0x02 ... 2 bits TX_BIT_LEN = 0x1F ... 31 bits TX_BIT_LEN = 0x00 ... 32 bits</p>

[2]	TX_NEG	发送数据边沿反向位 1 = 在SPICLK下降沿改变发送数据输出信号 0 =在SPICLK上升沿改变发送数据输出信号
[1]	RX_NEG	负边沿接收 1 = 在SPICLK的下降沿锁定接收数据输入信号 0 =在SPICLK的上升沿锁定接收数据输入信号
[0]	GO_BUSY	通讯或忙状态标志 1 = 主机模式下, 写1开始SPI数据传输; 从机模式下, 写1表示从机准备好与主机通信。 0 = 写0停止数据传输, 即使正在发送。 数据传输过程中, 该位值保持为1, 传输完成后, 该位自动清零。 注: 1, 在写1到GO_BUSY之前, SPI其它寄存器都应该配置完成. 2, FIFO模式, 此位由硬件控制, 软件不要改变此位。

SPI 除频寄存器 (SPI_DIVIDER)

寄存器	偏移量	R/W	描述	复位后的值
SPI_DIVIDER	SPIx_BA+0x04	R/W	时钟除频寄存器 (Master only)	0x0000_0000

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

Bits	描述	
[31:16]	DIVIDER2	<p>时钟除频2 寄存器 (Master only)</p> <p>这些位于设置第2个频率除频器的值. 在SPICLK引脚产生的串行时钟, 可根据下列方程获得所期望的频率:</p> $f_{sclk} = \frac{f_{pclk}}{(DIVIDER2 + 1) * 2}$ <p>当VARCLK_EN置1时, 此设置才有意义.</p>
[15:0]	DIVIDER	<p>时钟除频寄存器 (Master only)</p> <p>这些位于设置频率除频器的值. 在SPICLK引脚所输出的串行时钟. 可根据下列方程获得所期望的频率:</p> $f_{sclk} = \frac{f_{pclk}}{(DIVIDER + 1) * 2}$ <p>从机模式下, SPI所能接受的最高时钟频率是PCLK的1/5.</p>

SPI 从机选择寄存器 (SPI_SSR)

寄存器	偏移量	R/W	描述	复位后的值
SPI_SSR	SPI0_BA+0x08	R/W	从机选择寄存器	0x0000_0000

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

Bits	描述	
[31:6]	保留	保留
[5]	LTRIG_FLAG	<p>电平触发标志 在从机模式下SS_LTRIG置位时，该标志能够表示接收位的数量是否达到要求。 1: 接收数量和接收位达到TX_NUM 及 TX_BIT_LEN内的设置值. 0: 接收数量或接收位没有符合设置值。. 注：该位只读</p>
[4]	SS_LTRIG	<p>从机电平触发选择 (Slave only) 1: 从机选择信号为电平触发。根据 SS_LVL选择是高电平/低电平触发 0: 从机输入边沿触发。此为该位默认值.</p>
[3]	AUTOSS	<p>自动从机选择 (Master only) 1 = 该位置位， SPI_SS[1:0]信号自动产生。意思是在通过设置GO_BUSY 位 (SPI_CNRRL[0])使发送/接收开始时，被SSR[1:0]选择的从机选择信号引脚将由SPI控制器设置为active状态. 而当数据传输结束时，SPI控制器会自动将从机选择引脚设置为inactive状态. 0 = 该位清位，可以通过软件自行设置或清零SPI_SSR[1:0]的相关位来决定从机选择引脚输出为active状态或inactive状态.</p>
[2]	SS_LVL	<p>从机选择触发电平选择 定义从机选择信号(SPISSx0/1)的有效电平. 1 = 从机选择信号SPISSx0/1 在高电平/上升沿有效</p>

		0 = 从机选择信号SPISSx0/1 在低电平/下降沿有效.
[1:0]	SSR	<p>从机选择寄存器 (Master only)</p> <p>当 AUTOSS 位被清除，对 SSR[1:0] 任何一位写 1，将会激活所对应的 SPISSx0/1 线，写 0 则 SPISSx0/1 线呈现非活动状态 (inactive state)。</p> <p>当 AUTOSS 位被设置，对 SSR[1:0] 任何一位写 0，将会使该位所对应的 SPISSx0/1 引脚输出 inactive state；对 SSR[1:0] 任何一位写 1，该位所对应的 SPISSx0/1 引脚的输出状态将由硬件自动控制，数据传输时有效，其它时段无效。</p> <p>由 SS_LVL 决定活动状态 (active state) 的信号类型。</p> <p>注： SPISSx0 在从机模式下被定义为从机选择输入</p>

SPI 数据接收寄存器 (SPI_RX)

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

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

Bits	描述	
[31:0]	RX	<p>数据接收寄存器</p> <p>数据接收寄存器内保存最后一次传输所接收的数据。数据的有效长度根据 SPI_CTRL 寄存器内定义的长度决定。例如，Tx_BIT_LEN 设定为 0x08 且 Tx_NUM 设定为 0x0，Rx0[7:0] 内保存传输数据，其它位的值为不确定的数值。</p> <p>注：数据接收寄存器为只读寄存器。</p>

SPI 数据发送寄存器(SPI_TX)

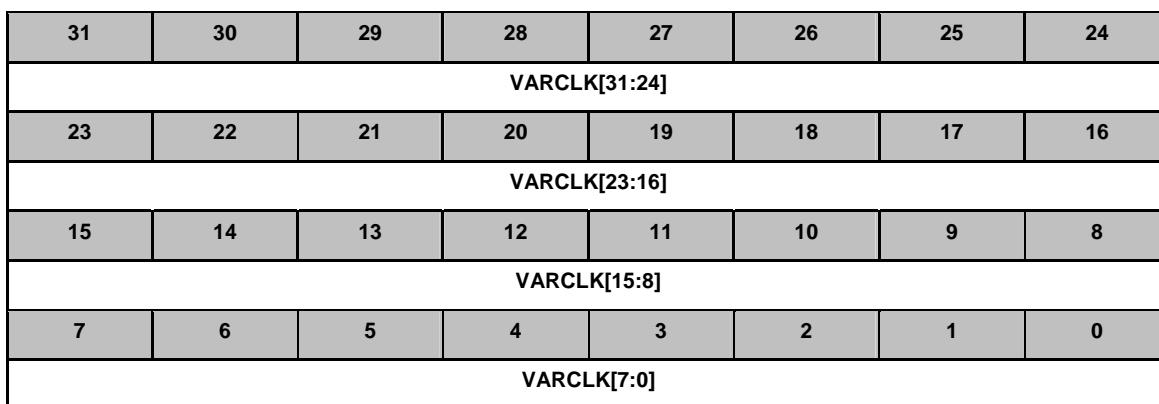
寄存器	偏移量	R/W	描述	复位后的值
SPI_TX0	SPIx_BA+0x20	W	数据发送寄存器 0	0x0000_0000
SPI_TX1	SPIx_BA+0x24	W	数据发送寄存器 1	0x0000_0000

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

Bits	描述	
[31:0]	TX	<p>数据发送寄存器</p> <p>数据发送寄存器内存储下一次被发送的数据。数据的有效长度根据CNTRL寄存器内定义的长度决定。</p> <p>例如，Tx_BIT_LEN 设定为 0x08 且 Tx_NUM 设定为 0x0，Tx0[7:0] 内的数据将被发送。如果 Tx_BIT_LEN 设定为 0x00 且 Tx_NUM 设定为 0x1，SPI 控制器将以相同的设置连续发送/接收 2 个 32 位数据。传送的顺序是先 Tx0[31:0] 然后 Tx1[31:0]。</p>

SPI 可调时钟类型寄存器 (SPI_VARCLK)

寄存器	偏移量	R/W	描述	复位后的值
SPI_VARCLK	SPIx_BA+0x34	R/W	可调时钟类型寄存器	0x007F_FF87



Bits	描述	
[31:0]	VARCLK	<p>可调时钟类型</p> <p>该寄存器的值定义了SPI时钟的频率类型。如果VARCLK的类型为‘0’，SPICLK的输出频率取决于DIVIDER的值。如果VARCLK的类型为‘1’，SPICLK的输出频率取决于DIVIDER2的值。参考寄存器SPI_DIVIDER。</p> <p>详细参考可调串行时钟的章节。</p>

5.10 定时器控制器 (TMR)

5.10.1 概述

定时器模块包含4组32位定时器，TIMER0~TIMER3，提供用户便捷的计数定时功能。定时器模块可支持例如频率测量，计数，间隔时间测量，时钟产生，延迟时间等功能。定时器可在计时溢出时产生中断信号，也可在操作过程中提供计数的当前值。

5.10.2 特征

- 4 组 32-位定时器，带24位向上定时器和一个8位的预分频计数器
- 每个定时器都有独立的时钟源
- 提供one-shot, periodic, toggle 和 auto-reload 计数操作模式
- 超时周期 = (输入的定时器时钟周期) * (8-bit预分频计数器 + 1) * (24-bit TCMP)
- 最大计数周期= $(1 / 25 \text{ MHz}) * (2^8) * (2^{24})$, 如果定时器时钟周期是 25 MHz
- 通过 TDR (定时器数据寄存器) 可读取内部24位向上计数器的值

5.10.3 框图

每个通道带一个8位预分频计数器，一个24位向上计数器，一个24位比较寄存器和一个中断请求信号。参阅图 5-57 的定时器控制框图。每个通道有4个时钟源选项，图 5-56 为时钟源控制功能。

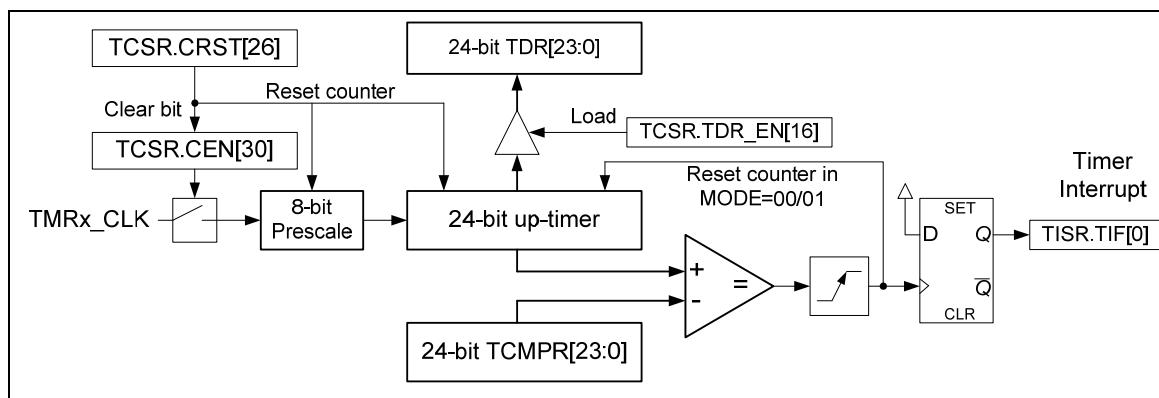


图 5-56 定时器控制器框图

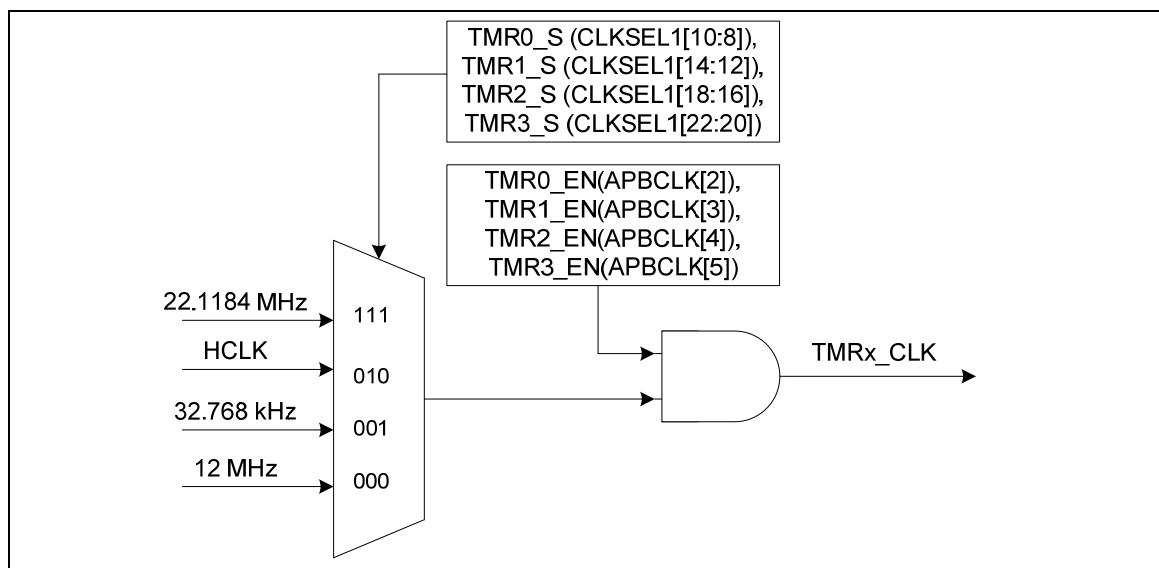


图 5-57 定时器控制的时钟源

5.10.4 功能描述

定时器控制器提供one-shot, period 和 toggle, auto-reload 模式操作。每种操作功能模式如下所示:

5.10.4.1 One -Shot 模式

如果定时器工作在单周期（one-shot）模式且CEN (定时器使能位) 置1，定时器的计数器开始计数。一旦定时器计数器的值达到定时器比较寄存器 (TCMPR) 的值，且IE (中断使能位) 置1，则定时器中断标志置位，产生中断信号并送到NVIC通知CPU，表明定时器计数发生溢出。如果IE (interrupt enable bit) 置0，无中断信号发生。在此工作模式下，一旦定时器计数器的值达到定时器比较寄存器(TCMPR) 的值时定时器计数操作停止，该比较的操作仅执行一次。因此，该操作称为One-Shot 模式。

5.10.4.2 Periodic 模式

如果定时器工作在周期模式且CEN (定时器使能位)置1，定时器计数器开始计数。一旦定时器计数器的值达到定时器比较寄存器 (TCMPR)的值，且IE (中断使能位) 设置为1'b1，则定时器中断标志置位且产生中断信号，并发送到NVIC通知CPU，表示定时器计数溢出发生。如果IE (中断使能位) 设置为0，无中断信号发生。在该工作模式下，一旦定时器计数器的值达到定时器比较器寄存器 (TCMPR) 的值，定时器计数器的值返回计数初始值且CEN 保持为1 (持续使能计数)。定时器计数器再次计数。如果软件清除中断标志，一旦定时器计数器的值与定时器比较寄存器(TCMPR)的值匹配且IE (中断使能位)设置为1'b1，产生中断信号并送到NVIC通知CPU。也就是说，定时器计数与TCMPR比较的操作是周期性的。直到CEN设置为0，定时器计数操作才会停止，中断信号的产生也是周期性的。因此，这种操作模式称为Periodic 模式。

5.10.4.3 Toggle 模式

如果定时器工作在toggle 模式且CEN (定时器使能位)置1，定时器计数器开始计数。一旦定时器计数器的值与定时器比较寄存器TCMPR的值匹配时，且IE (中断使能位)设置为 1'b1，则定时器中断标志置位，产生中断信号并送到NVIC通知CPU。表示定时器发生计数溢出。相应toggle输出(tout)信号置1。在这种操作模式，一旦定时器计数器的值与定时器比较寄存器TCMPR的值匹配，定时器计数器的值返回到计数初始值且 CEN 保持为 1 (持续使能计数)。定时器计数器重新计数。如果中断标志由软件清除，一旦定时器计数器的值与定时器比较寄存器中TCMPR的值匹配且IE (中断使能位) 置1，则定时器中断标志置位，发生中断信号，并送到NVIC再次通知CPU。相应 toggle输出 (tout)信号置0。定时器计数操作在CEN设置为0之后才停止。因此，toggle输出 (tout)信号 以50%的占空比反复改变。所以这种操作模式称为Toggle 模式。

5.10.4.4 连续记数模式

CEN=1时，工作于连续计数模式当TDR=TCMPR时，若允许中断，则会发生中断，但计数器不会停止，一直计数到 $2^{24}-1$ ，然后再回到0，开始新一轮计数。改变TCMPR的值不必停止Timer。

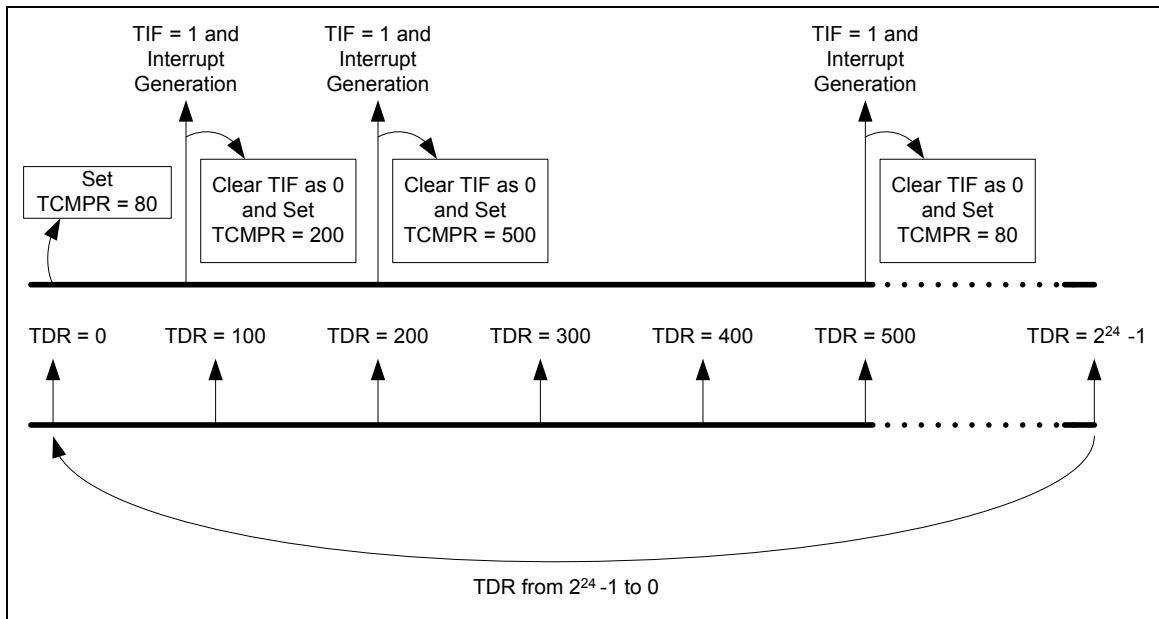


图 5-58 连续计数模式

5.10.4.5 事件计数功能

对TM0~TM3引脚进行计数，外部信号的频率必须小于HCLK的1/3。

5.10.5 寄存器映射

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

寄存器	偏移量	R/W	描述	复位后的值
TMR_BA01 = 0x4001_0000				
TMR_BA23 = 0x4011_0000				
TCSR0	TMR_BA01+0x00	R/W	Timer0 控制和状态寄存器	0x0000_0005
TCMPR0	TMR_BA01+0x04	R/W	Timer0 比较寄存器	0x0000_0000
TISR0	TMR_BA01+0x08	R/W	Timer0 中断状态寄存器	0x0000_0000
TDR0	TMR_BA01+0x0C	R	Timer0 数据寄存器	0x0000_0000
TCSR1	TMR_BA01+0x20	R/W	Timer1 控制和状态寄存器	0x0000_0005
TCMPR1	TMR_BA01+0x24	R/W	Timer1 比较寄存器	0x0000_0000
TISR1	TMR_BA01+0x28	R/W	Timer1 中断状态寄存器	0x0000_0000
TDR1	TMR_BA01+0x2C	R	Timer1 数据寄存器	0x0000_0000
TCSR2	TMR_BA23+0x00	R/W	Timer2 控制和状态寄存器	0x0000_0005
TCMPR2	TMR_BA23+0x04	R/W	Timer2 比较寄存器	0x0000_0000
TISR2	TMR_BA23+0x08	R/W	Timer2 中断状态寄存器	0x0000_0000
TDR2	TMR_BA23+0x0C	R	Timer2 数据寄存器	0x0000_0000
TCSR3	TMR_BA23+0x20	R/W	Timer3 控制和状态寄存器	0x0000_0005
TCMPR3	TMR_BA23+0x24	R/W	Timer3 比较寄存器	0x0000_0000
TISR3	TMR_BA23+0x28	R/W	Timer3 中断状态寄存器	0x0000_0000
TDR3	TMR_BA23+0x2C	R	Timer3 数据寄存器	0x0000_0000

5.10.6 寄存器描述

定时器控制寄存器 (TCSR)

寄存器	偏移量	R/W	描述				复位后的值
TCSR0	TMR_BA01+0x00	R/W	Timer0 控制与状态寄存器				0x0000_0005
TCSR1	TMR_BA01+0x20	R/W	Timer1 控制与状态寄存器				0x0000_0005
TCSR2	TMR_BA23+0x00	R/W	Timer2 控制与状态寄存器				0x0000_0005
TCSR3	TMR_BA23+0x20	R/W	Timer3 控制与状态寄存器				0x0000_0005

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

Bits	描述	
[31]	DBGACK_TMR	仿真器 (ICE) 调试模式响应禁止 (写保护位) 0 = 仿真器调试模式响应影响定时器计数. 当调试器调试模式响应时, 定时器计数器将被固定住. 1 = 仿真器调试模式响应禁止 无论调试器调试模式响应与否, 定时器计数器将持续计数下去.
[30]	CEN	计数器使能位 1 = 开始计数 0 = 停止/暂停计数 注1: 设置CEN为1, 使能24位计数器从上次停止的计数值继续计数. 注2: 在 one-shot 模式下 (MODE[28:27]=00b), 当相应的定时中断产生时 (IE[29]=1b), 该位由硬件自动清零.
[29]	IE	中断使能 1 = 使能定时器中断 0 = 禁用定时器中断

		当定时器中断使能，当计数值与TCMPR寄存器内数值相同时，触发中断.
[28:27]	MODE	定时器工作模式
		模式 定时器工作模式
		00 当定时器定义为单触发模式(one-shot)时，定时器溢出仅触发中断一次 (IE 使能)，进入中断后CEN自动清除为0.
		01 当定时器定义为周期模式(period)时，定时器每次溢出都触发中断 (IE 使能).
		10 当定时器工作在toggle 模式，中断信号周期性产生 (如果IE 使能). 且相应的信号 (tout)以占空比为50%周期改变.
		11 定时器工作在auto-reload 计数模式. 相应的中断在TDR =TCMP时产生 (如果IE使能); 然而，24位的向上定时器继续计数而不会复位.
[26]	CRST	计数器重置 设置该位将重置定时器计数器，预分频和使CEN为0. 0 = 无动作. 1 = 重置定时器的预分频计数器，内部24位向上计数器和CEN位
[25]	CACT	定时器工作状态 该位表示当前定时器计数器的状态。 0 = 定时器未工作 1 = 定时器工作中
[24]	CTB	计数模式使能位 该位是计数模式使能位. 当定时器用作事件计数器时，该位需要设置成 1 且定时器作为事件计数器外部触发引脚. 1 = 使能计数器模式 0 = 禁用计数器模式
[23:17]	保留	保留
[16]	TDR_EN	数据锁存使能 当置位TDR_EN，计数器运行时，TDR (Timer数据寄存器) 将被24位向上计数器的值 不断更新 1 = Timer数据寄存器 更新使能 0 = Timer数据寄存器 禁止更新
[15:8]	保留	保留
[7:0]	PRESCALE	预分频计数器 时钟输入根据Prescale +1进行预分频. 如果PRESCALE数值为0，不进行预分频.

定时器比较寄存器 (TCMPR)

寄存器	偏移量	R/W	描述	复位后的值
TCMPR0	TMR_BA01+0x04	R/W	Timer0 比较寄存器	0x0000_0000
TCMPR1	TMR_BA01+0x24	R/W	Timer1 比较寄存器	0x0000_0000
TCMPR2	TMR_BA23+0x04	R/W	Timer2 比较寄存器	0x0000_0000
TCMPR3	TMR_BA23+0x24	R/W	Timer3 比较寄存器	0x0000_0000

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

Bits	描述	
[31:24]	保留	保留
[23:0]	TCMP	<p>定时器比较值</p> <p>TCMP是24位比较寄存器。当内部24位向上计数器的值与TCMP的值相当时，如果TCSR.IE[29]=1，就产生定时器中断请求。TCMP的值为定时器计数周期。</p> <p>定时溢出周期= (Period of timer clock input) * (8-bit Prescale + 1) * (24-bit TCMP)</p> <p>注1: 不能在TCMP里写0x0或0x1，否则内核将运行到未知状态。</p> <p>注2: 无论CEN为0或1，软件向该寄存器写入新的值，TIMER将使用新的值并退出当前计数，开始重新计数。</p>

定时器中断状态寄存器 (TISR)

寄存器	偏移量	R/W	描述	复位后的值
TISR0	TMR_BA01+0x08	R/W	Timer0 中断状态寄存器	0x0000_0000
TISR1	TMR_BA01+0x28	R/W	Timer1 中断状态寄存器	0x0000_0000
TISR2	TMR_BA23+0x08	R/W	Timer2 中断状态寄存器	0x0000_0000
TISR3	TMR_BA23+0x28	R/W	Timer3 中断状态寄存器	0x0000_0000

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

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

定时器数据寄存器 (TDR)

寄存器	偏移量	R/W	描述	复位后的值
TDR0	TMR_BA01+0x0C	R/W	Timer0 数据寄存器	0x0000_0000
TDR1	TMR_BA01+0x2C	R/W	Timer1 数据寄存器	0x0000_0000
TDR2	TMR_BA23+0x0C	R/W	Timer2 数据寄存器	0x0000_0000
TDR3	TMR_BA23+0x2C	R/W	Timer3 数据寄存器	0x0000_0000

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

Bits	描述	
[31:24]	保留	保留
[23:0]	TDR	定时器数据寄存器 TCSR.TDR_EN 置1时，内部24位定时器的值加载到TDR中，用户可以读取该寄存器的值。

5.11 看门狗定时器(WDT)

看门狗定时器是在软件出问题时执行系统复位功能，可以防止系统无限止地挂机，除此之外，看门狗定时器还可将芯片由掉电模式唤醒。看门狗定时器包含一个18位的自动运行的计数器，可编程其定时溢出间隔。表 5-7 为看门狗定时溢出间隔选择，图 5-59 为看门狗中断信号与复位信号的时序。

设置WTE(WDTCR[7])使能看门狗定时器和WDT计数器开始计数。当计数器达到选择的定时溢出间隔，看门狗定时器中断标志WTIF被立即置位，并请求WDT中断（如果看门狗定时器中断使能位WTIE置位），同时紧接着会有一个指定周期(1024 * TWDT)的延时。用户必须在指定周期(1024 * TWDT)内设置WTR(WDTCR[0]) (Watchdog timer reset)为高，重置18位WDT计数器，防止芯片复位，WTR在WDT计数重置后自动由硬件清零。通过设置WTIS(WDTCR[10:8])选择8个定时溢出间隔。如果在特殊延迟时间终止后，如果WDT计数没有被清零，看门狗定时器将置看门狗定时器复位标志(WTRF)为高并使芯片复位。这个复位将持续63个WDT时钟，然后芯片重启，并从复位向量(0x0000 0000)执行程序。WTRF将不被看门狗复位清零。用户可用软件拉低WTRF。WDT提供唤醒功能，当芯片在掉电状态，且看门狗定时器唤醒功能使能位(WDTR[4])置位，如果在超出指定延迟时间后WDT计数器没有被清零，芯片将由掉电状态唤醒。

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

表 5-7 看门狗定时溢出间隔选择

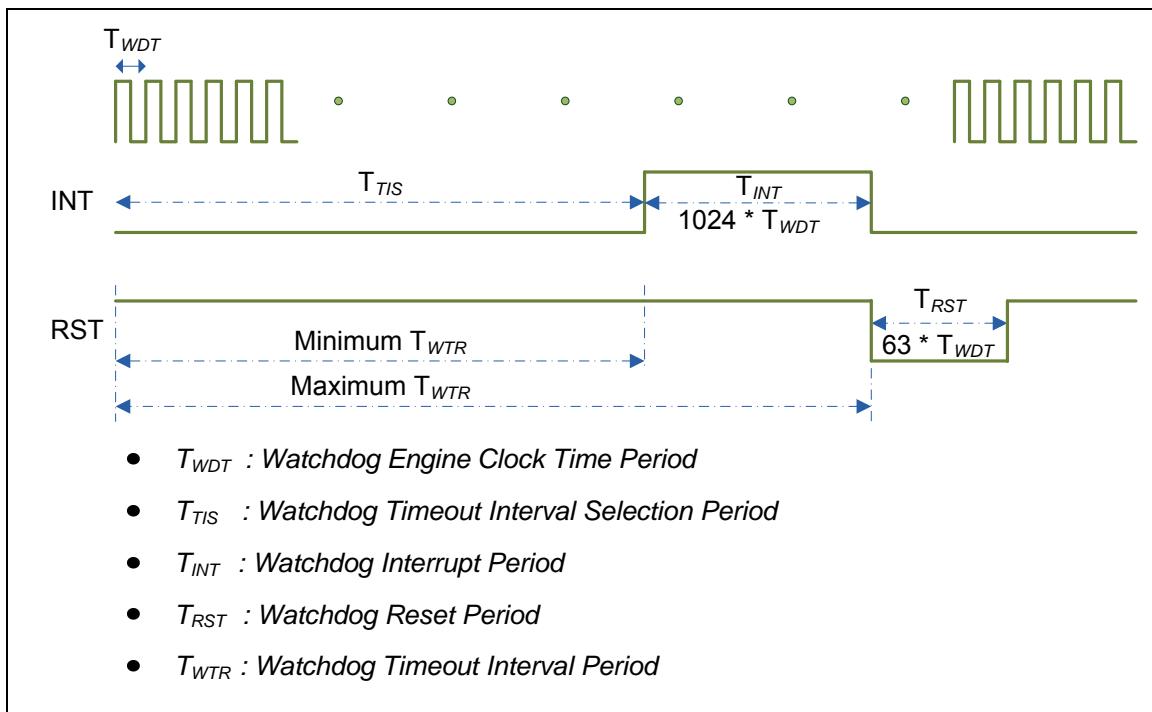


图 5-59 中断定时和复位信号时序

5.11.1 特征

- 18-位计数器以防止芯片跑飞.
- 可选择的定时溢出间隔 ($2^4 \sim 2^{18}$)， 定时溢出间隔为 $104\ ms \sim 26.3168\ s$ (if $WDT_CLK = 10\ kHz$).
- 复位周期 = $(1 / 10\ kHz) * 63$, if $WDT_CLK = 10\ kHz$.

5.11.2 框图

看门狗定时器时钟控制和框图如下所示。

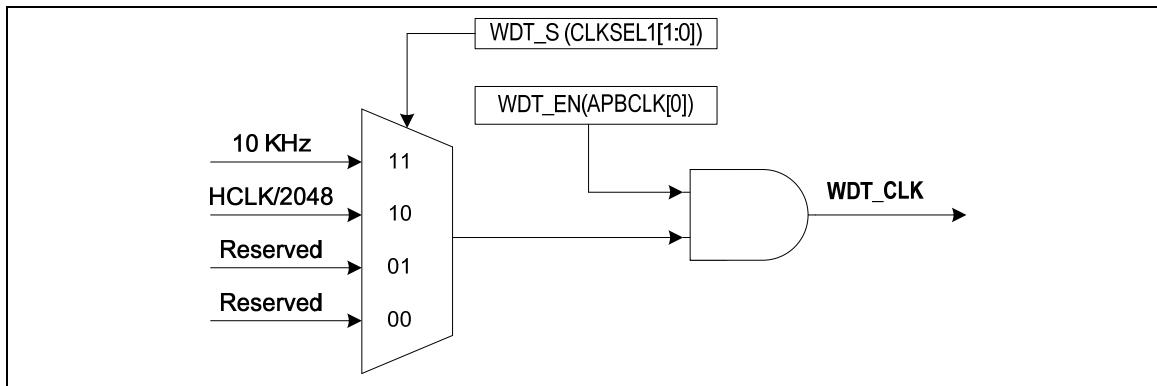


图 5-60 看门狗定时时钟控制

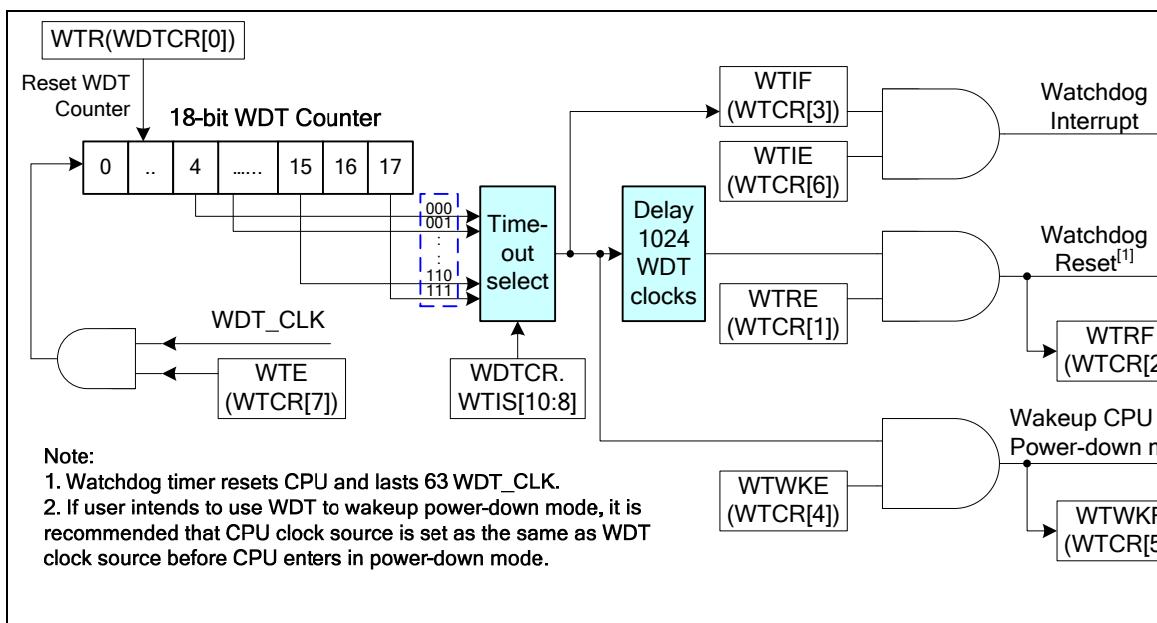


图 5-61 看门狗定时器框图

5.11.3 寄存器映射

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

寄存器	偏移量	R/W	描述	复位后的值
WDT_BA = 0x4000_4000				
WTCR	WDT_BA+0x00	R/W	看门狗定时器控制寄存器	0x0000_0700

5.11.4 寄存器描述

看门狗定时器控制寄存器(WTCR)

寄存器	偏移量	R/W	描述	复位后的值
WTCR	WDT_BA+0x00	R/W	看门狗定时器控制寄存器	0x0000_0700

注: 该寄存器所有位受保护, 修改该位时, 需要依次向0x5000_0100写入"59h", "16h", "88h", 参考寄存器 REGWRPROT, 地址 GCR_BA + 0x100.

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

Bits	描述																																				
[31]	DBGACK_WDT 仿真器 (ICE) 调试模式响应禁止 (写保护位) 0 = 仿真器调试模式响应影响看门狗定时器计数. 当调试器调试模式响应时, 看门狗定时器计数器将被固定住. 1 = 仿真器调试模式响应禁止 无论调试器调试模式响应与否, 看门狗定时器计数器将持续计数下去																																				
[30:11]	保留																																				
[10:8]	WTIS 看门狗定时器间隔选择 (写保护位) 这3位选择看门狗定时器的定时溢出间隔. <table border="1" style="margin-left: 20px;"> <tr> <th>WTIS</th> <th>Timeout Interval Selection</th> <th>Interrupt Period</th> <th>WTR Timeout Interval (WDT_CLK=10 KHz)</th> </tr> <tr> <td>000</td> <td>$2^4 * T_{WDT}$</td> <td>$(1024) * T_{WDT}$</td> <td>1.6 ms ~ 104 ms</td> </tr> <tr> <td>001</td> <td>$2^6 * T_{WDT}$</td> <td>$(1024) * T_{WDT}$</td> <td>6.4 ms ~ 108.8 ms</td> </tr> <tr> <td>010</td> <td>$2^8 * T_{WDT}$</td> <td>$(1024) * T_{WDT}$</td> <td>25.6 ms ~ 128 ms</td> </tr> <tr> <td>011</td> <td>$2^{10} * T_{WDT}$</td> <td>$(1024) * T_{WDT}$</td> <td>102.4 ms ~ 204.8 ms</td> </tr> <tr> <td>100</td> <td>$2^{12} * T_{WDT}$</td> <td>$(1024) * T_{WDT}$</td> <td>409.6 ms ~ 512 ms</td> </tr> <tr> <td>101</td> <td>$2^{14} * T_{WDT}$</td> <td>$(1024) * T_{WDT}$</td> <td>1.6384 s ~ 1.7408 s</td> </tr> <tr> <td>110</td> <td>$2^{16} * T_{WDT}$</td> <td>$(1024) * T_{WDT}$</td> <td>6.5536 s ~ 6.656 s</td> </tr> <tr> <td>111</td> <td>$2^{18} * T_{WDT}$</td> <td>$(1024) * T_{WDT}$</td> <td>26.2144 s ~ 26.3168 s</td> </tr> </table>	WTIS	Timeout Interval Selection	Interrupt Period	WTR Timeout Interval (WDT_CLK=10 KHz)	000	$2^4 * T_{WDT}$	$(1024) * T_{WDT}$	1.6 ms ~ 104 ms	001	$2^6 * T_{WDT}$	$(1024) * T_{WDT}$	6.4 ms ~ 108.8 ms	010	$2^8 * T_{WDT}$	$(1024) * T_{WDT}$	25.6 ms ~ 128 ms	011	$2^{10} * T_{WDT}$	$(1024) * T_{WDT}$	102.4 ms ~ 204.8 ms	100	$2^{12} * T_{WDT}$	$(1024) * T_{WDT}$	409.6 ms ~ 512 ms	101	$2^{14} * T_{WDT}$	$(1024) * T_{WDT}$	1.6384 s ~ 1.7408 s	110	$2^{16} * T_{WDT}$	$(1024) * T_{WDT}$	6.5536 s ~ 6.656 s	111	$2^{18} * T_{WDT}$	$(1024) * T_{WDT}$	26.2144 s ~ 26.3168 s
WTIS	Timeout Interval Selection	Interrupt Period	WTR Timeout Interval (WDT_CLK=10 KHz)																																		
000	$2^4 * T_{WDT}$	$(1024) * T_{WDT}$	1.6 ms ~ 104 ms																																		
001	$2^6 * T_{WDT}$	$(1024) * T_{WDT}$	6.4 ms ~ 108.8 ms																																		
010	$2^8 * T_{WDT}$	$(1024) * T_{WDT}$	25.6 ms ~ 128 ms																																		
011	$2^{10} * T_{WDT}$	$(1024) * T_{WDT}$	102.4 ms ~ 204.8 ms																																		
100	$2^{12} * T_{WDT}$	$(1024) * T_{WDT}$	409.6 ms ~ 512 ms																																		
101	$2^{14} * T_{WDT}$	$(1024) * T_{WDT}$	1.6384 s ~ 1.7408 s																																		
110	$2^{16} * T_{WDT}$	$(1024) * T_{WDT}$	6.5536 s ~ 6.656 s																																		
111	$2^{18} * T_{WDT}$	$(1024) * T_{WDT}$	26.2144 s ~ 26.3168 s																																		

[7]	WTE	看门狗定时器使能 (写保护位) 0 = 禁用看门狗定时器功能(该动作重置内部计数器) 1 = 使能看门狗定时器
[6]	WTIE	看门狗定时器中断使能 (写保护位) 0 = 禁用看门狗定时器中断 1 = 使能看门狗定时器中断
[5]	WTWKF	看门狗定时器唤醒标志 如果看门狗定时器引起芯片从掉电模式下唤醒，该位将被置高。必须由软件向该位写1清零 0 = 看门狗定时器不能引起芯片唤醒. 1 = 芯片由空闲或掉电模式被看门狗定时溢出唤醒.
[4]	WTWKE	看门狗定时器唤醒功能使能位 (写保护位) 0 = 禁用看门狗唤醒芯片功能. 1 = 使能看门狗唤醒芯片功能.
[3]	WTIF	看门狗定时器中断标志 如果看门狗定时器中断使能，该位表示看门狗定时器中断发生，如果没有使能看门狗定时器中断，该位表示定时溢出周期已过。 0= 不发生看门狗定时器中断 1= 发生看门狗定时器中断 注: 写1到该位，清零.
[2]	WTRF	看门狗定时器复位标志 当看门狗定时器溢出引发复位，该位被置位，通过读取该位可以确认复位是否由看门狗引起。该位通过写1手动清除，如果 WTRE 禁止，看门狗定时器溢出对该位无影响。 0 = 复位不是由看门狗定时器产生。 1 = 看门狗定时器引发复位 注: 该位只读，由软件清零.
[1]	WTRE	看门狗定时器复位使能 (写保护位) 设定该位使能看门狗定时器复位功能。 0 = 禁用看门狗定时器复位功能 1 = 使能看门狗定时器复位功能
[0]	WTR	清看门狗定时器 (写保护位) 置位该位清看门狗定时器. 0 = 无动作 1 = 重置看门狗定时器 注: 几个时钟周期后该位自动清零

5.12 UART 接口控制器 (UART)

NuMicro™ NUC122提供2个UART通道，支持普通速度，另外UART0与UART1支持流控制。

5.12.1 概述

通用异步收/发器(UART0/1)支持IrDA SIR 功能和RS-485模式功能。每个通道都有 7 种类型的中断，它们是发送FIFO 空中断(INT_THRE), 接收门限到达中断(Int_RDA), 线路状态中断 (校验错,帧错或break 中断) (INT_RLS) , 接收超时中断(INT_Tout), MODEM /唤醒 状态中断(INT_Modem) , 缓冲错误中断(INT_BUF_ERR)。中断号13 (中断向量29) 支持UART0/1的中断. 参考NVIC章节对系统中断列表的描述。

UART0/1 接口控制器内嵌一个16-byte 发送FIFO (TX_FIFO) 和 16-byte 接收 FIFO (RX_FIFO) 来降低CPU的中断数量； 在操作过程中CPU可以随时读UART的状态. 报告的状态信息包括已经被UART执行的传输操作的类型和条件，也包括4 种错误条件(parity error, overrun error, framing error, 或break中断). UART包括一个可编程的波特率发生器，它可以将输入晶振除以一个除数来得到收发器需要的时钟. 波特率公式是 $Baud\ Rate = UART_CLK / M * [BRD + 2]$. 其中M和BRD在波特率分频寄存器UA_BAUD中定义，见表 5-8.

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

表 5-8 UART 波特率公式

System clock = 22.1184 MHz			
Baud rate	Mode0	Mode1	Mode2
921600	x	A=0,B=11	A=22
460800	A=1	A=1,B=15 A=2,B=11	A=46
230400	A=4	A=4,B=15 A=6,B=11	A=94
115200	A=10	A=10,B=15 A=14,B=11	A=190
57600	A=22	A=22,B=15 A=30,B=11	A=382
38400	A=34	A=62,B=8 A=46,B=11 A=34,B=15	A=574

19200	A=70	A=126,B=8 A=94,B=11 A=70,B=15	A=1150
9600	A=142	A=254,B=8 A=190,B=11 A=142,B=15	A=2302
4800	A=286	A=510,B=8 A=382,B=11 A=286,B=15	A=4606

表 5-9 UART 波特率设置表

UART0/1 有流控 /CTS 和 /RTS，使能流程时，/RTS 无效则UART将不接收数据，直到 UART向外发送/RTS信号。当 Rx FIFO 中的数据个数和RTS_TRI_LEVEL (UA_FCR [19:16])相等时，UART向外发/RTS无效信号。发送时，如果 /CTS 无效，UART 将不向外发送数据。

UA_FUN_SEL[IrDA_EN] 置1时，UART工作在红外模IrDA (SIR, 串行红外)。该模式有1 个开始位，8个 数据位，和1个 停止位。最大数据速率 为 115.2 Kbps (半双工). IrDA SIR 包括 IrDA SIR 编码/解码协议。IrDA SIR 物理层规定在传输和接收之间至少有10ms 延时。这个由软件来完成。

NuMicro™ NUC122 UART控制器的另一个功能是RS-485 9位模式功能，由RTS脚控制方向或由软件编程(PB.2 for RTS0 and PB.6 for RTS1) 执行该功能. RS-485 模式由设置寄存器UA_FUN_SEL来选择.. 在 RS-485 模式, RX和TX的许多特性与UART相同。

5.12.2 特征

- 全双工，异步通信
- 收发各有16个字节FIFO
- 支持硬件流控(CTS/RTS)，触发电平可编程
- 每个通道独立的可编程波特率发生器
- CTS有唤醒CPU功能
- 支持8位接收超时功能
- 发送字节时，UA_TOR [DLY]可设置字节时间间隔。
- 支持break error, 帧错和收发缓冲溢出检测功能
- 5. 完全可编程的串口特性
 - 支持5, 6, 7, 8位的数据位
 - 支持校验位：奇校验，偶校验，无校验，或校验位粘着产生和侦测
 - 支持 1, 1.5, 或 2 位停止位

支持红外模式

- 普通模式下支持 3/16比特时间

支持 RS-485功能模式.

- 支持 RS-485 9bit 模式
- 由RTS引脚提供硬件或软件的方向控制

5.12.3 框图

UART时钟控制和框图见图 5-62 和 图 5-63.

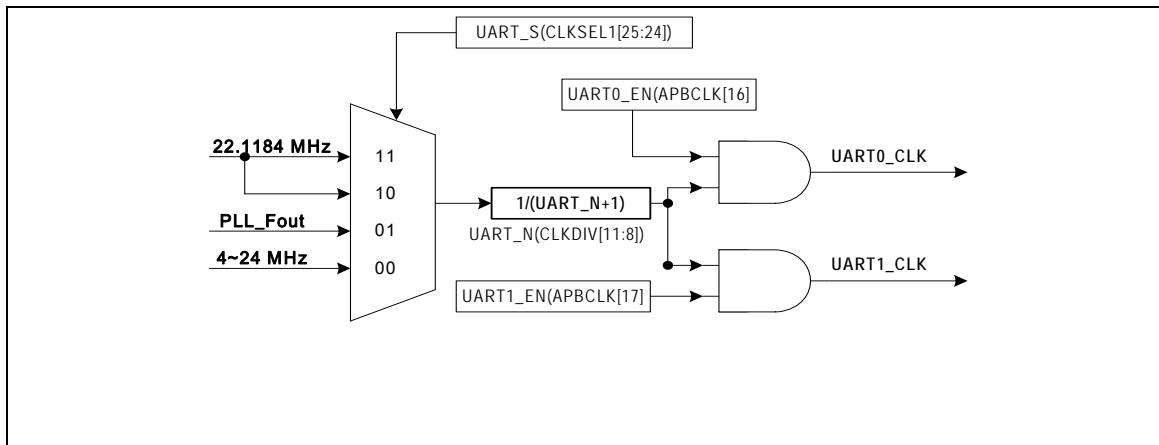


图 5-62 UART 时钟控制图

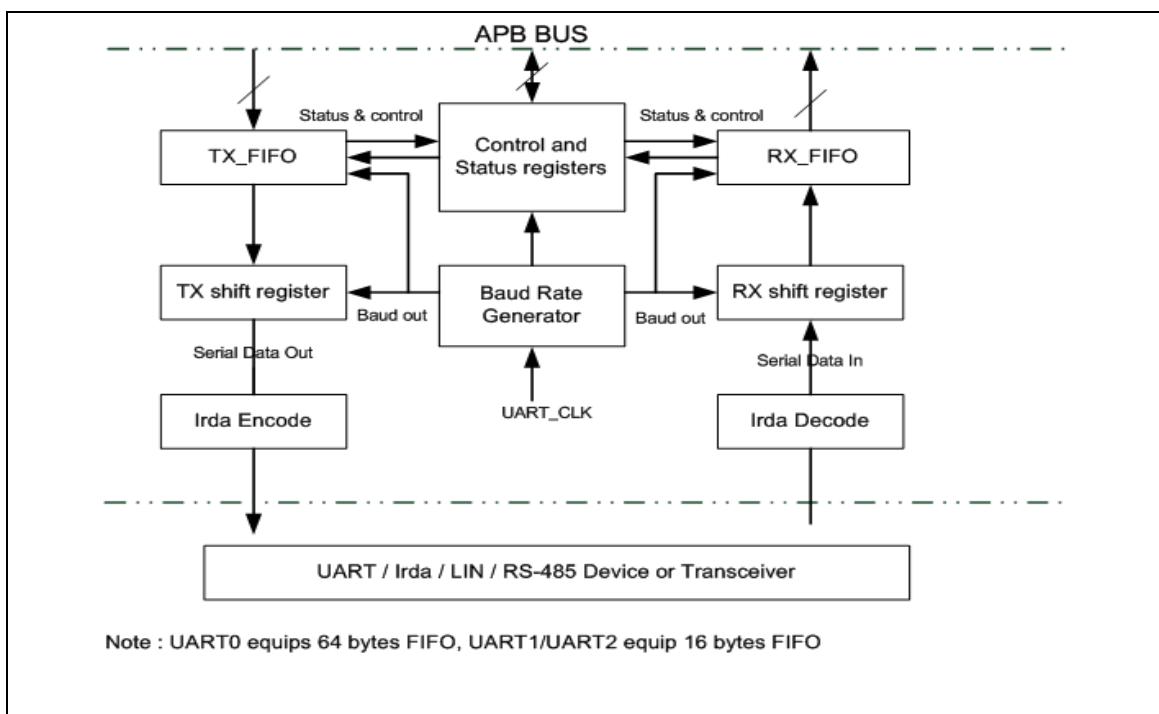


图 5-63 UART 框图

TX_FIFO

发送用一个16个字节的FIFO做缓存来降低CPU的中断数量.

RX_FIFO

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

TX shift Register

移位发送的数据串行输出.

RX shift Register

串行移位接收到的数据.

Modem控制寄存器

这个寄存器控制MODEM 或者数据集或者一个MODEM模拟器的接口.

波特率发生器

将外部时钟除以一个除数来产生期望的内部时钟, 参考波特率方程.

IrDA 编码

IrDA 编码控制模块.

IrDA 解码

IrDA 解码控制模式.

控制和状态寄存器

此寄存器设定, 包括 FIFO 控制寄存器 (UA_FCR), FIFO 状态寄存器(UA_FSR), 和线路控制寄存器 (UA_LCR)应用于传输和接收. 时间超时控制寄存器(UA_TOR) 应用于识别时间超时中断控制. 该寄存器包括中断控制使能寄存器 (UA_IER) 和中断状态寄存器 (UA_ISR) 来使能或者禁止中断响应并且识别发生的中断. 有六种中断: FIFO为空中断 (INT_THRE) , 接收开始中断 (INT_RDA) , line 状态中断 (overrun error or 校验 error or framing error or break interrupt) (INT_RLS) , 定时溢出中断 (INT_Tout) , MODEM/唤醒状态中断 (INT_Modem) , 和缓冲错误中断 (INT_Buf_Err) .

下图为自动流控制框图.

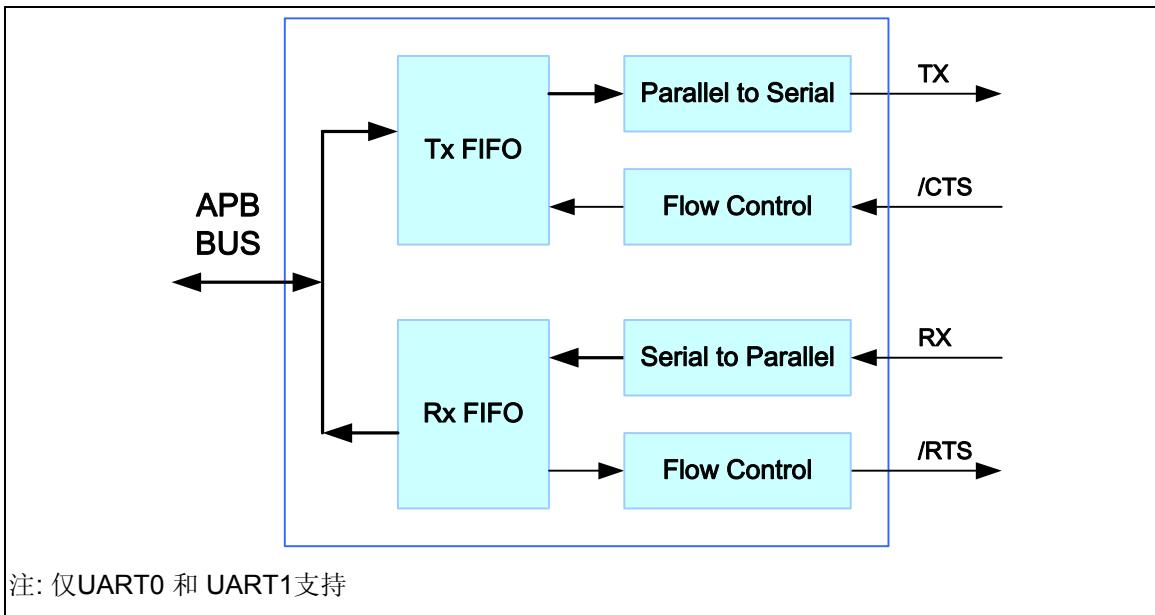


图 5-64 自动流控制框图

5.12.4 IrDA 模式

UART 支持 IrDA SIR (Serial Infrared) 传输编码和接收解码，通过设定 UA_FUN_SEL 寄存器的 IrDA_EN 位来选择 IrDA 模式。

IrDA 模式下，UA_BAUD[DIV_X_EN] 寄存器需禁止。

波特率 = Clock / (16 * BRD)，BRD 为波特率分频 UA_BAUD 寄存器。

下图为 IrDA 控制框图。

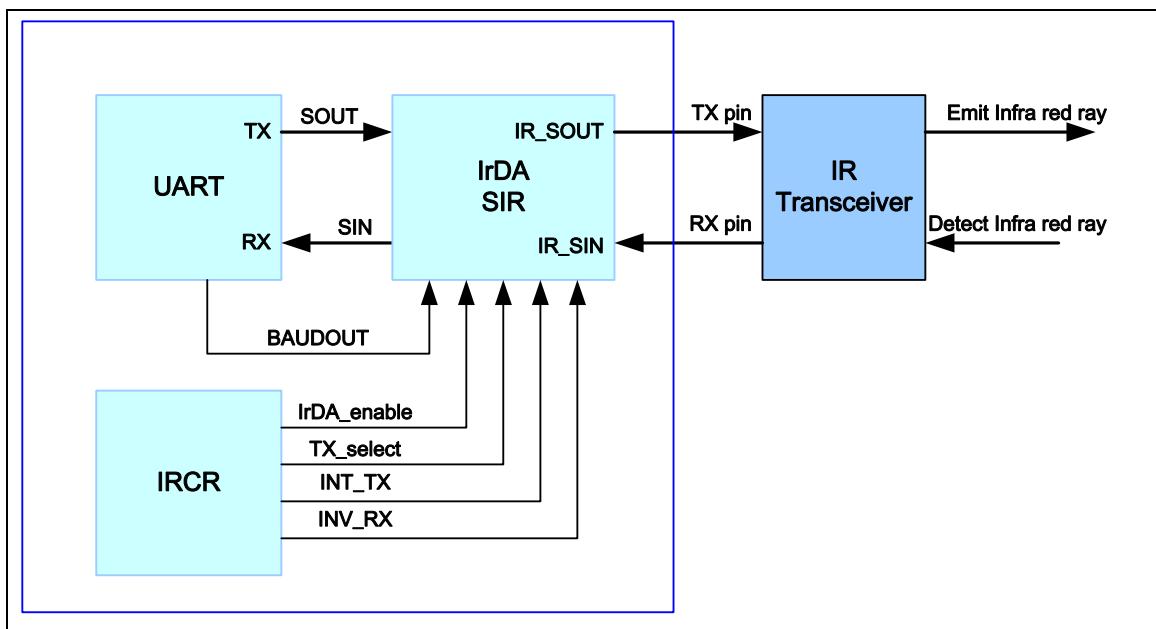


图 5-65 IrDA 框图

5.12.4.1 IrDA SIR 传输编码

IrDA SIR 传输编码调制 Non-Return-to Zero (NRZ) 传输位从 UART 输出。IrDA SIR 物理层指定使用反向归零调制编码 (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发射二极管。

在正常模式下，传输脉冲的宽度为 3/16 波特率周期。

5.12.4.2 IrDA SIR 接收解码

SIR 接收解码器对来自红外接收器的归零位比特流进行解调，并将接收到的 NRZ 串行比特流输出到 UART。在空闲状态里，解码器输入通常是高，用于标记状态。(因此，IRCR bit 6 默认设定为1)

当解码器输入为低时，表明接收到一个起始位

5.12.4.3 IrDA SIR 运作

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：

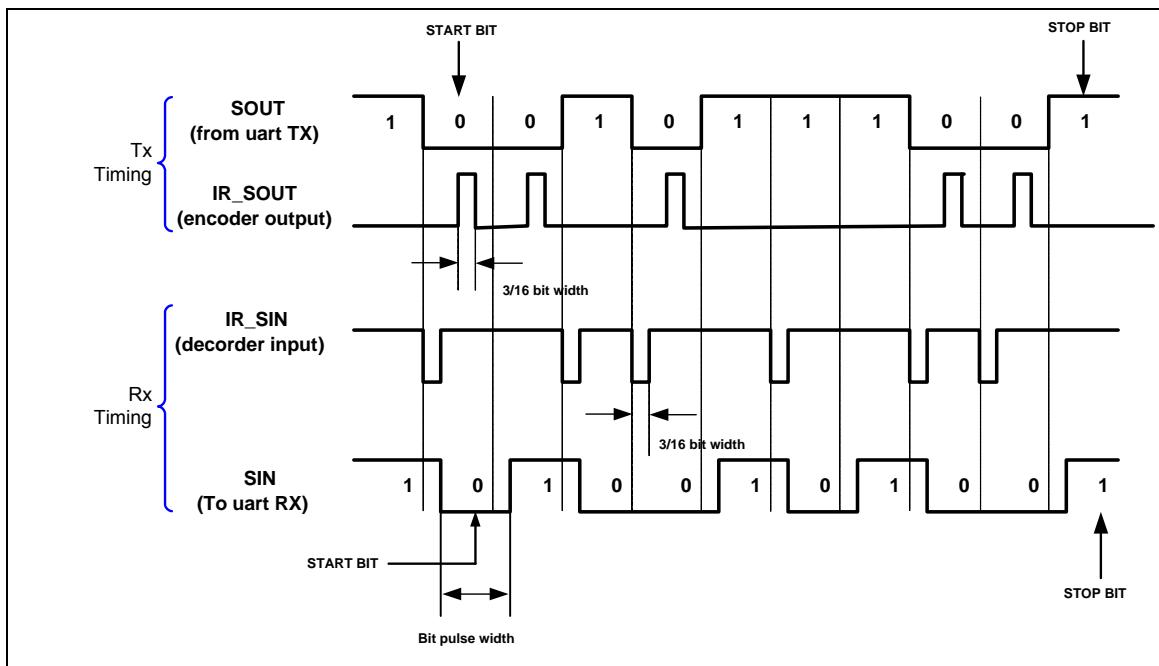


图 5-66 IrDA TX/RX 时序图

5.12.5 RS-485 功能模式

UART支持RS-485 9 位模式功能. 设置UA_FUN_SEL选择RS-485 mode. RS-485 通过RTS驱动控制异步串口的控制信号, 使能RS-485 驱动器. RS-485 模式, RX与TX的许多特性与UART一样.

RS-485 发送模式, 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优先级(bit 9th) 为 1标志地址特性,对于数据特性, 优先级设置为0. 设置寄存器UA_LCR 控制第9位(PBE , EPE 和 SPE置位, 第9位发送0 , PBE 和 SPE 置位, EPE清零, 第9位发送1). 该控制器支持三种操作模式: RS-485 普通模式(NMM), RS-485 自动地址识别模式 (AAD) 和RS-485 自动方向控制模式(AUD), 可通过UA_RS-485_CSR 的设置选择其中一种工作模式, 通过设置UA_TOR [DLY] 可以设置上一个停止与下一个开始位之间的延迟时间..

RS-485 普通多点操作模式(NMM)

RS-485普通多点操作模式, 首先, 软件决定在检测到地址字节之前哪些地数据存储到RX-FIFO, 如果软件在检测到地址之前想忽略任何数据, 流设置UART_FCR[RS485_RX_DIS]使能UA_RS-485[RS485_NMM], 接收器会忽略数据, 直到检测到地址字节(bit9 =1)并且地址字节数据存储到RX-FIFO. 在检测到地址字节之前, 软件想接收任何数据, 流禁止UART_FCR [RS485_RX_DIS], 使能UA_RS-485[RS485_NMM], 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 , 设置UA_RS-485_CSR [RX_DIS], 软件决定是否使能或禁止接收器接收数据. 如果使能接收器, 就会接收所有字节数据并存储在RX-FIFO, 如果禁止接收器, 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. 如果设置寄存器UA_RS-485_CSR [RX_DIS]禁止接收器, 当检测到下一个地址字节, 控制器将清UA_RS-485_CSR [RX_DIS]且地址字节数据将存储在RX-FIFO.

RS-485自动地址识别模式(AAD)

RS-485 自动地址识别模式, 接收器在检测到地址字节 (bit9=1) 并且地址字节数据与UA_RS-485[ADDR_MATCH]的值相匹配之前将忽略所有数据. 地址字节数据存储在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UA_RS-485[ADDR_MATCH] 的值不匹配.

RS-485自动方向模式(AUD)

RS-485控制器的另一个功能是自动方向控制. RS-485 通过RTS驱动控制异步串口的控制信号, 使能RS-485 驱动器. RS-485 模式. RTS 连接到RS-485 驱动器, 设置RTS线为高 (逻辑1) ,使能RS-485 驱动器. 设置RTS为低 (逻辑0) , 使驱动器进入tri-state状态. 用户通过设置寄存器UA_MCR 中的LEV_RTS位改变 RTS 驱动电平.

编程流程:

1. 设置寄存器UA_FUN_SEL中的FUN_SEL位选择RS-485.
2. 设置寄存器UA_FCR 中的RX_DIS 位使能或禁止RS-485 接收器
3. 设置RS-485_NMM 或 RS-485_AAD 模式.
4. 如果选择RS-485_AAD 模式, ADDR_MATCH设置成自动地址匹配值.
5. 若为自动方向模式, 设置RS-485_AUD.

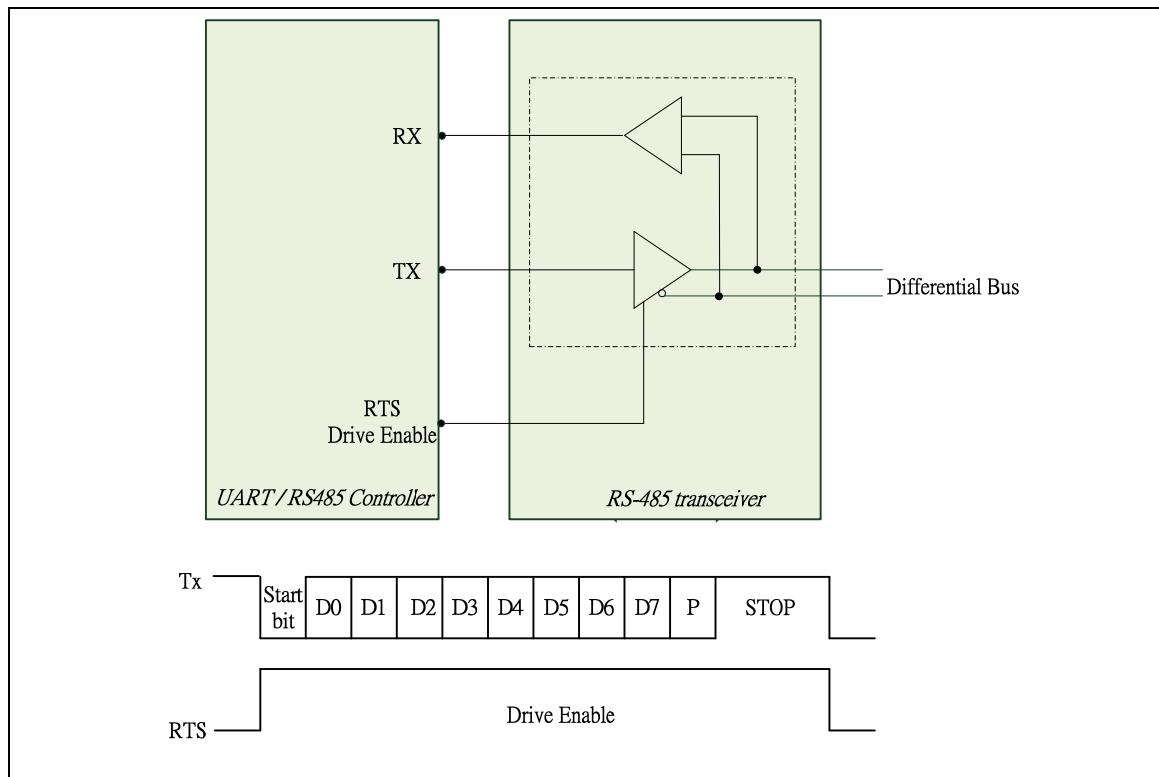


图 5-67 RS-485 帧结构

5.12.6 寄存器映射

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

寄存器	偏移量	R/W	描述	复位后的值
UART Base Address :				
Channel0 : UART0_BA (High Speed) = 0x4005_0000				
Channel1 : UART1_BA (Normal Speed)= 0x4015_0000				
UA_RBR	UART0_BA+0x00	R	UART0 接收数据缓冲寄存器.	Undefined
	UART1_BA+0x00	R	UART1 接收数据缓冲寄存器.	Undefined
UA_THR	UART0_BA+0x00	W	UART0 发送保持寄存器	Undefined
	UART1_BA+0x00	W	UART1 发送保持寄存器	Undefined
UA_IER	UART0_BA+0x04	R/W	UART0 中断使能寄存器	0x0000_0000
	UART1_BA+0x04	R/W	UART1 中断使能寄存器	0x0000_0000
UA_FCR	UART0_BA+0x08	R/W	UART0 FIFO 控制寄存器	0x0000_0000
	UART1_BA+0x08	R/W	UART1 FIFO 控制寄存器	0x0000_0000

UA_LCR	UART0_BA+0x0C	R/W	UART0 线控制寄存器	0x0000_0000
	UART1_BA+0x0C	R/W	UART1 线控制寄存器	0x0000_0000
UA_MCR	UART0_BA+0x10	R/W	UART0 Modem 控制寄存器	0x0000_0000
	UART1_BA+0x10	R/W	UART1 Modem 控制寄存器	0x0000_0000
UA_MSR	UART0_BA+0x14	R/W	UART0 Modem 状态寄存器	0x0000_0000
	UART1_BA+0x14	R/W	UART1 Modem 状态寄存器	0x0000_0000
UA_FSR	UART0_BA+0x18	R/W	UART0 FIFO 状态寄存器	0x1040_4000
	UART1_BA+0x18	R/W	UART1 FIFO 状态寄存器	0x1040_4000
UA_ISR	UART0_BA+0x1C	R/W	UART0 Interrupt 状态寄存器	0x0000_0002
	UART1_BA+0x1C	R/W	UART1 Interrupt 状态寄存器	0x0000_0002
UA_TOR	UART0_BA+0x20	R/W	UART0 定时溢出寄存器	0x0000_0000
	UART1_BA+0x20	R/W	UART1 定时溢出寄存器	0x0000_0000
UA_BAUD	UART0_BA+0x24	R/W	UART0 波特率分频寄存器	0x0F00_0000
	UART1_BA+0x24	R/W	UART1 波特率分频寄存器	0x0F00_0000
UA_IRCR	UART0_BA+0x28	R/W	UART0 IrDA 控制寄存器	0x0000_0040
	UART1_BA+0x28	R/W	UART1 IrDA 控制寄存器	0x0000_0040
UA_ALT_CSR	UART0_BA+0x2C	R/W	UART0 复用控制/状态寄存器	0x0000_0000
	UART1_BA+0x2C	R/W	UART1 复用控制/状态寄存器	0x0000_0000
UA_FUN_SEL	UART0_BA+0x30	R/W	UART0 功能选择寄存器	0x0000_0000
	UART1_BA+0x30	R/W	UART1 功能选择寄存器	0x0000_0000

5.12.7 寄存器描述

接收缓冲寄存器 (UA_RBR)

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



Bits	描述	
[31:8]	保留	保留
[7:0]	RBR	接收缓冲寄存器 (Read only) 读此寄存器, UART 将返回一组从 Rx pin接收到的 8-位数据 (LSB first).



发送保持寄存器 (UA_THR)

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

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

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



中断使能寄存器 (UA_IER)

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

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

Bits	描述	
[31:14]	保留	保留
[13]	AUTO_CTS_EN	<p>CTS 自动流控制使能 1 = 使能 CTS 自动流控制. 0 = 禁用 CTS 自动流控制.</p> <p>当 CTS 自动溢出控制使能当侦测到CTS信号UART将向外部驱动器发送数据(UART 将不发送数据 直到CTS 被侦测).</p>
[12]	AUTO_RTS_EN	<p>RTS 自动流控制使能 1 = 使能 RTS 自动流控制. 0 = 禁用 RTS 自动流控制.</p> <p>当 RTS 自动流使能,当Rx FIFO 的数值和 UA_FCR[RTS_Tri_Lev]相等时, UART 将禁止 RTS 信号.</p>
[11]	TIME_OUT_EN	<p>Time Out 计数器使能 1 = 使能 Time-out 计数器. 0 = 禁用 Time-out 计数器</p>
[10:7]	保留	保留
[6]	WAKE_EN	唤醒功能使能

		0 = 禁用UART 唤醒功能 1 =使能唤醒功能, 当系统在掉电模式下, 外部 CTS 的改变将 芯片 从掉电模式下唤醒.
[5]	BUF_ERR_IEN	Buffer Error 中断使能 1 = 使能INT_Buf_err中断 0 = 禁用INT_Buf_err中断
[4]	RTO_IEN	RX Time Out 中断使能 1 = 使能 INT_TOUT 0 = 屏蔽 INT_TOUT
[3]	MODEM_IEN	Modem 中断状态使能 1 = 使能 INT_MODEM 0 = 屏蔽INT_MODEM
[2]	RLS_IEN	线上接收中断状态使能 1 =使能INT_RLS 0 =屏蔽INT_RLS
[1]	THRE_IEN	发送保持寄存器空中断使能 1 =使能INT_THRE 0 = 屏蔽INT_THRE
[0]	RDA_IEN	可接收数据中断使能. 1 = 使能 INT_RDA 0 = 屏蔽 INT_RDA



FIFO 控制寄存器 (UA_FCR)

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

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

Bits	描述																			
[31:20]	保留	保留																		
[19:16]	RTS_TRILEV	RTS 触发自动流程控制使用 <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>RTS_TRILEV</th> <th>Trigger Level (Bytes)</th> </tr> </thead> <tbody> <tr> <td>0000</td> <td>01</td> </tr> <tr> <td>0001</td> <td>04</td> </tr> <tr> <td>0010</td> <td>08</td> </tr> <tr> <td>0011</td> <td>14</td> </tr> <tr> <td>0100</td> <td>30/14 (高速/常态)</td> </tr> <tr> <td>0101</td> <td>46/14 (高速/常态)</td> </tr> <tr> <td>0110</td> <td>62/14 (高速/常态)</td> </tr> <tr> <td>others</td> <td>62/14 (高速/常态)</td> </tr> </tbody> </table> 注: 该寄存器用于自动RTS流控制.	RTS_TRILEV	Trigger Level (Bytes)	0000	01	0001	04	0010	08	0011	14	0100	30/14 (高速/常态)	0101	46/14 (高速/常态)	0110	62/14 (高速/常态)	others	62/14 (高速/常态)
RTS_TRILEV	Trigger Level (Bytes)																			
0000	01																			
0001	04																			
0010	08																			
0011	14																			
0100	30/14 (高速/常态)																			
0101	46/14 (高速/常态)																			
0110	62/14 (高速/常态)																			
others	62/14 (高速/常态)																			
[15:9]	保留	保留																		
[8]	RX_DIS	接收器禁用寄存器. 是否禁用接收器 (set 1 is disable receiver) 1 = 禁用接收器																		

		<p>0 = 使能接收器 注: 该位用于RS-485 普通多点模式. 需要在UA_ALT_CSR [RS-485_NMM]之前设置.</p>																		
[7:4]	RFITL	<p>RX FIFO 中断 (INT_RDA) 触发级别 FIFO 接收字节数等于 RFITL 后RDA_IF 将被置位 (如果UA_IER [RDA_IEN]使能, 将产生中断).</p> <table border="1"> <thead> <tr> <th>RFITL</th><th>INTR_RDA触发级别 (Bytes)</th></tr> </thead> <tbody> <tr><td>0000</td><td>01</td></tr> <tr><td>0001</td><td>04</td></tr> <tr><td>0010</td><td>08</td></tr> <tr><td>0011</td><td>14</td></tr> <tr><td>0100</td><td>30/14 (高速/常态)</td></tr> <tr><td>0101</td><td>46/14 (高速/常态)</td></tr> <tr><td>0110</td><td>62/14 (高速/常态)</td></tr> <tr><td>others</td><td>62/14 (高速/常态)</td></tr> </tbody> </table>	RFITL	INTR_RDA触发级别 (Bytes)	0000	01	0001	04	0010	08	0011	14	0100	30/14 (高速/常态)	0101	46/14 (高速/常态)	0110	62/14 (高速/常态)	others	62/14 (高速/常态)
RFITL	INTR_RDA触发级别 (Bytes)																			
0000	01																			
0001	04																			
0010	08																			
0011	14																			
0100	30/14 (高速/常态)																			
0101	46/14 (高速/常态)																			
0110	62/14 (高速/常态)																			
others	62/14 (高速/常态)																			
[3] 保留 保留																				
<p>TX软件复位 当 Tx_RST 置位, FIFO 传输和Rx 内部状态将被清 0. 1 = 该位置位将复位 Tx 内部状态机和指针. 0 = 该位写 0 将无效. 注: 至少 3 UART时钟周期 自动清 0.</p>																				
<p>RX软件复位 当 Rx_RST 置位, FIFO 传输和Rx 内部状态将被清 0. 1 = 该位置位将复位 Rx 内部状态机和指令状态. 0 = 该位写 0 将无效. 注: 至少 3 UART时钟周期 自动清 0.</p>																				
[0] 保留 保留																				

Line 控制寄存器 (UA_LCR)

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

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

Bits	描述	
[31:7]	保留	保留
[6]	BCB	钳制控制位 该位置位时，串行数据输出 (Tx) 将被迫间隔发送数据 (logic 0). 该位仅作用于 Tx 对传输逻辑不起作用。
[5]	SPE	Stick 奇偶使能 1 = 当PBE和EPE置位时，Parity Bit傳送和檢驗時為1，.当PBE置位和EPE被清除時，Parity Bit傳送和檢驗時為0 0 = 禁止 stick 奇偶使能
[4]	EPE	Even 奇偶使能 1 = 偶數位檢驗和傳輸 Parity Bit 0 = 奇數位檢驗和傳輸 Parity Bit 该位仅当 bit 3 (parity bit enable) 位 置位有效.
[3]	PBE	奇偶使能位 0 = 当传输时 奇偶位 无(transmit data) 产生或检测 (receive data). 1 = 該位被置起時，會產生或檢測Parity bit (串行数据中的"last data word bit" 和 "stop bit"之間的位).
[2]	NSB	停止位数目

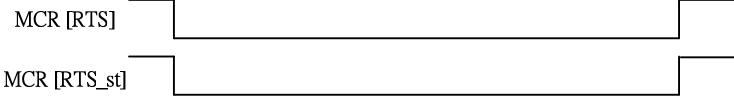
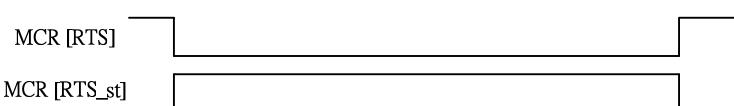
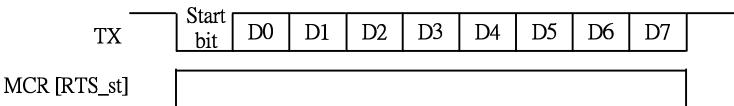
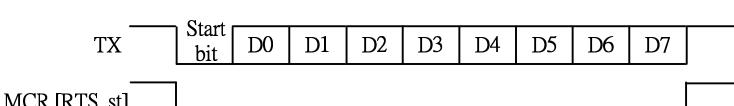
		0= 传递数据时 1“STOP bit”产生 1=传递数据时 1.5 “STOP bit”产生(5-bit word 长度被选择); 选择6-, 7- 和 8-位 word 长度.产生2 “STOP bit”										
[1:0]	WLS	字长度选择 <table border="1"><thead><tr><th>WLS[1:0]</th><th>Character length</th></tr></thead><tbody><tr><td>00</td><td>5 bits</td></tr><tr><td>01</td><td>6 bits</td></tr><tr><td>10</td><td>7 bits</td></tr><tr><td>11</td><td>8 bits</td></tr></tbody></table>	WLS[1:0]	Character length	00	5 bits	01	6 bits	10	7 bits	11	8 bits
WLS[1:0]	Character length											
00	5 bits											
01	6 bits											
10	7 bits											
11	8 bits											

**MODEM 控制寄存器 (UA_MCR)**

寄存器	偏移量	R/W	描述	复位后的值
UA_MCR	UART0_BA+0x10	R/W	UART0 Modem 控制寄存器	0x0000_0000
	UART1_BA+0x10	R/W	UART1 Modem 控制寄存器	0x0000_0000

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

Bits	描述	
[31:14]	保留	保留
[13]	RTS_ST	RTS Pin状态 (Read only) 该位表示 RTS 引脚状态.
[12:10]	保留	保留

		RTS 触发电平 该位改变 RTS 触发电平。 1= 高电平触发 0= 低电平触发 <i>UART Mode : MCR[Lev_RTS]=1</i>  <i>UART Mode : MCR[Lev_RTS]=0</i>  <i>RS-485 Mode : MCR[Lev_RTS]=1</i>  <i>RS-485 Mode : MCR[Lev_RTS]=0</i> 
[9]	LEV_RTS	
[8:2]	保留	保留
[1]	RTS	RTS (Request-To-Send) 信号 0: 使能 RTS 管脚为 1 (如果 Lev_RTS 设定低电平触发). 1: 使能 RTS 管脚为 0 (如果 Lev_RTS 设定低电平触发). 0: 使能 RTS 管脚为 0 (如果 Lev_RTS 设定高电平触发). 1: 使能 RTS 管脚为 1 (如果 Lev_RTS 设定高电平触发).
[0]	保留	保留

Modem 状态寄存器 (UA_MSR)

寄存器	偏移量	R/W	描述	复位后的值
UA_MSR	UART0_BA+0x14	R/W	UART0 Modem 状态寄存器	0x0000_0000
	UART1_BA+0x14	R/W	UART1 Modem 状态寄存器	0x0000_0000

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

Bits	描述	
[31:9]	保留	保留
[8]	LEV_CTS	CTS 触发电平 该位可改变 CTS 触发电平。 1=高电平触发 0=低电平触发
[7:5]	保留	保留
[4]	CTS_ST	CTS Pin 状况 (Read only) 该位表示 CTS 管脚状态
[3:1]	保留	保留
[0]	DCTSF	侦测 CTS 状态改变标志位 (Read only) 只要 CTS 输入状态改变 该位置位，可向 CPU 产生中断请求(使能UA_IER [MODEM_IEN])。 注：该位只读，可写‘1’清除。



FIFO 状态寄存器 (UA_FSR)

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

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

Bits	描述	
[31:29]	保留	保留
[28]	TE_FLAG	传输清空标志位 (Read only) 当 Tx FIFO(UA_THR) 清空并且最后一个字节的STOP 位传输完毕, 该位由硬件自动置位. 当 Tx FIFO(UA_THR) 未清空或最后一字节的STOP 位未传输, 该位由硬件自动清空. 注: 该位只读.
[27:25]	保留	保留
[24]	TX_OVER_IF	TX 溢出 Error 中断标志位 (Read only) 若 Tx FIFO(UA_THR) 充满, 另外写 UA_THR 将引起置1. 注: 该位只读, 可通过写'1' 清除该位.
[23]	TX_FULL	传输 FIFO 满 (Read only) 该位表示 Tx FIFO 是否充满. 当 TX_POINTER 和 64/16(UART0/UART1)相等 该位置位, 反之由硬件清除.
[22]	TX_EMPTY	发送 FIFO 为空 (Read only) 该位表示 Tx FIFO 是否空. 当 Tx FIFO 中最后一个数据传输到 Shift 寄存器, 硬件置位该位. 当写数据到 THR (Tx FIFO not empty) 该位清除.

[21:16]	TX_POINTER	TX FIFO Pointer (Read only) 该位表示Tx FIFO 缓冲指针. 当CPU 写1个字节到 UA_THR, Tx_Pointer 增 1. 当 Tx FIFO 传输 1 位到 Shift 寄存器, Tx_Pointer 减1.
[15]	RX_FULL	接收 FIFO 满 (Read only) 该位表示 Rx FIFO 是否充满. 当 RX_POINTER和 64/16(UART0/UART1)相等 该位置位, 反之由硬件清除.
[14]	RX_EMPTY	接收 FIFO 为空 (Read only) 该位表示 Rx FIFO是否为空. 当 Rx FIFO 最后字节 从CPU中读取, 硬件置位该位. 当 UART 接收到新数据 该位清除.
[13:8]	RX_POINTER	RX FIFO Pointer (Read only) 该位表示 Rx FIFO 缓冲指示器. 当UART 从外部驱动器接收到 1字节数据, Rx_Pointer增 1. 当 Rx FIFO 通过 CPU 读 1字节数据, Rx_Pointer 减1.
[7]	保留	保留
[6]	BIF	钳制中断标志位 (Read only) 当"spacing state" (logic 0) 内数据的长度大于输入全字传输(即"start bit" + data bits + parity + stop bits的所有时间)的时间, 该位置1。 注: 该位只读, 软件写“1”清除该位.
[5]	FEF	Framing Error 标志位 (Read only) 若接收字符中无有效" stop bit" 该位将置位, CPU 写 1到该位复位. 注: 该位只读, 软件写“1”清零
[4]	PEF	Parity Error 标志位 (Read only) 若 接收字符中无有效"parity bit" 该位将置位, CPU 写 1 到 该位 复位. 注: 该位只读, 软件写“1”清零.
[3]	RS485_ADD_DEF	RS-485 地址字节检测标志 (Read only) 在RS-485模式, 接收器检测到任何接收的地址字节(bit9 = '1') bit", 该位设置 为1且置位UA_ALT_CSR [RS-485_ADD_EN], 在CPU写1到该位时复位. 注: 该位用于RS-485功能模式. 注: 该位只读, 软件写“1”清零.
[2:1]	保留	保留
[0]	RX_OVER_IF	RX Overflow Error IF (Read only) 当 Rx FIFO 溢出, 该位置位. 当接收到的数据大于 Rx FIFO(UA_RBR) 范围, 64/16 UART0/UART1, 该位 置位. 注: 该位只读, 软件写“1”清零.

中断状态控制寄存器 (UA_ISR)

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

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

Bits	描述	
[31:14]	保留	保留
[13]	BUF_ERR_INT	<p>Buffer Error 中断指示器 (Read only) 将BUF_ERR_IEN 和 BUF_ERR_IF都置1， 该位置位. 1 = 产生buffer error 中断 0 = 没有buffer error 中断</p>
[12]	TOUT_INT	<p>Time Out 中断指示器 (Read only) 将TOUT_IEN 和 TOUT_IF都置1， 该位置位. 1 = 产生Tout 中断 0 = 没有 Tout 中断</p>
[11]	MODEM_INT	<p>MODEM 状态中断指示器 (Read only). 将MODEM_IEN 和 MODEM_IF都置1， 该位置位. 1 = 产生Modem中断 0 = 没有Modem中断</p>
[10]	RLS_INT	<p>线上数据接收中断状态指示中断控制器 (Read only). 将RLS_IEN 和 RLS_IF都设置为1， 该位置位. 1 = 产生RLS 中断 0 = 没有RLS 中断</p>

[9]	THRE_INT	发送保持寄存器 清空中断状态指示中断控制器 (Read only). 将THRE_IEN 和 THRE_IF都置1, 该位置位. 1 = 产生THRE 中断 0 = 没有THRE 中断
[8]	RDA_INT	接收数据可用中断状态指示中断控制器 (Read only). 将RDA_IEN 和RDA_IF都置1, 该位置位. 1 = 产生RDA 中断 0 = 没有RDA 中断
[7:6]	保留	保留
[5]	BUF_ERR_IF	Buffer Error 中断标志位 (Read only) 当 TX 或 RX FIFO溢出 (Tx_Over_IF 或 Rx_Over_IF置位) ,该位置位. 当 Buf_Err_IF置位, 传输可能不正常. 若UA_IER [BUF_ERR_IEN] 使能, buffer error 中断产生. 注: Tx_Over_IF 和 Rx_Over_IF 被清除 该位清除.
[4]	TOUT_IF	Time Out 中断标志位 (Read only) 当Rx FIFO 未清空,同时RX FIFO没有溢出, 计数器的值和TOIC 相等, 该位置位. 若 UA_IER [TOUT_IEN] 使能, Tout 中断产生. 注: 用户可读 UA_RBR (Rx is in active) 清该位.
[3]	MODEM_IF	MODEM 中断标志位 (Read only) 当 CTS pin 状态改变(DCTS=1) 该位置位. 若UA_IER [MODEM_IEN]使能, Modem 中断产生. 注: 该位只读, 当写 1 到DCTS 该位清除.
[2]	RLS_IF	Receive Line 中断标志位 (Read only). 当 Rx接收数据有parity error, framing error 或 break error (至少 3 位, BIF, FEF 和 PEF, 置位)该位清空.若 IER[RLS_IEN]使能, RLS 中断产生. 注: 在RS-485模式, 该位包括“接收检测地址字节(bit9 = '1')” 注: 该位只读, 当 BIF, FEF 和PEF 清除 该位复位.
[1]	THRE_IF	Transmit Holding 寄存器 清空中断标志位 (Read only). 当Tx FIFO 最后数据传输到 Transmitter Shift 寄存器 该位置位. 若 UA_IER [THRE_IEN]置位, THRE 中断产生. 注: 该位只读 写数据到 THR (Tx FIFO not empty) 该位清空.
[0]	RDA_IF	Receive Data Available 中断标志位 (Read only). 当 Rx FIFO 数据和 RFITL 相等, RDA_IF 将置位. 若UA_IER [RDA_IEN]使能, RDA 中断产生. 注: 该位只读, Rx FIFO 跌落到 极限 级别 该位清除 (RFITL).

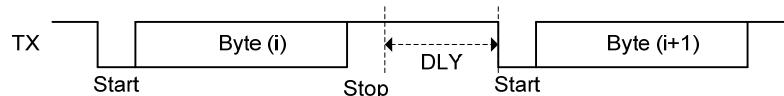
UART 中断源	中断使能位	中断指示 中断控制	中断标志位	标志位清除
Buffer Error Interrupt INT_BUF_ERR	BUF_ERR_IEN	BUF_ERR_INT	BUF_ERR_IF = (TX_OVER_IF or RX_OVER_IF)	Write '1' to TX_OVER_IF/ RX_OVER_IF
RX Timeout Interrupt INT_TOUT	RTO_IEN	TOUT_INT	TOUT_IF	Read UA_RBR
Modem Status Interrupt INT_MODEM	MODEM_IEN	MODEM_INT	MODEM_IF = (DCTS/DSR)	Write '1' to DCTS/DSR
Receive Line Status Interrupt INT_RLS	RLS_IEN	RLS_INT	RLS_IF = (BIF or FEF or PEF)	Write '1' to BIF/FEF/PEF
Transmit Holding Register Empty Interrupt INT_THRE	THRE_IEN	THRE_INT	THRE_IF	Write UA_THR
Receive Data Available Interrupt INT_RDA	RDA_IEN	RDA_INT	RDA_IF	Read UA_RBR

表 5-10 在软件模式下 UART 中断源和标志表

时间溢出寄存器 (UA_TOR)

寄存器	偏移量	R/W	描述	复位后的值
UA_TOR	UART0_BA+0x20	R/W	UART0 时间溢出寄存器	0x0000_0000
	UART1_BA+0x20	R/W	UART1 时间溢出寄存器	0x0000_0000



Bits	描述	
[31:16]	保留	保留
[15:8]	DLY	<p>TX 延迟时间值 该位用在编程上一次停止位与下一次开始位之间的传输延迟时间.</p> 
[7:0]	TOIC	<p>Time Out 中断比较器 当 RX FIFO 接收到新数据时间溢出计数器复位并开始计数(the counting clock = baud rate). 一旦 time out 计数器 (TOUT_CNT) 和中断比较器 (TOIC) 相等, 接收 time out 中断产生 (INTR_TOUT) 若 UA_IER [RTO_IEN]使能. 新的输入数据或RX FIFO为空可清INT_TOU.</p>

波特率分频寄存器 (UA_BAUD)

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

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

Bits	描述	
[31:30]	保留	保留
[29]	DIV_X_EN	分频 X 使能 BRD = 波特率分频, 波特率= $\text{Clock} / [M * (\text{BRD} + 2)]$; 默认 M 为 16. 1 = 使能分频 X (the equation of M = X+1, but DIVIDER_X [27:24] must ≥ 8). 0 = 禁用分频 X (the equation of M = 16) 注: 在IrDA 模式下 该位禁止.
[28]	DIV_X_ONE	Divider X equal 1 1 = Divider M = 1 (the equation of M = 1, but BRD [15:0] must ≥ 3). 0 = Divider M = X (the equation of M = X+1, but DIVIDER_X[27:24] must ≥ 8) 参考表 5-11
[27:24]	DIVIDER_X	Divider X 波特率分频 M = X+1.
[23:16]	保留	保留
[15:0]	BRD	波特率 分频 波特率分频位

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

表 5-11 波特率方程表

IrDA 控制寄存器 (IRCR)

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

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

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

注：在 IrDA 模式，寄存器 UA_BAUD[DIV_X_EN]必须禁止 (波特方程必须Clock / 16 * (BRD))

UART 复用控制/状态寄存器 (UA_ALT_CSR)

寄存器	偏移量	R/W	描述	复位后的值
UA_ALT_CSR	UART0_BA+0x2C	R/W	UART0 复用控制/状态寄存器	0x0000_0000
	UART1_BA+0x2C	R/W	UART1 复用控制/状态寄存器	0x0000_0000

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

Bits	描述	
[31:24]	ADDR_MATCH	地址匹配寄存器 该位包括RS-485 地址匹配值. 注: 该位用于RS-485自动地址检测模式.
[23:16]	保留	保留
[15]	RS485_ADD_EN	RS-485 地址检测使能 该位用于使能RS-485 地址检测使能模式. 1 = 使能地址检测模式 0 = 禁用地址检测模式 注: 该位用于RS-485任意 操作模式.
[14:11]	保留	保留
[10]	RS485_AUD	RS-485 自动方向模式 (AUD) 1 = 使能RS-485 自动方向操作模式(AUO) 0 = 禁用RS-485 自动方向操作模式 (AUO) 注: 在RS-485_AAD 或 RS-485_NMM 操作模式有效.
[9]	RS485_AAD	RS-485 自动地址检测操作模式 (AAD) 1 = 使能 RS-485 自动地址方向操作模式 (AAD)

		0 = 禁用 RS-485自动地址方向操作模式 (AAD) 注: 在 RS-485_NMM 操作模式下无效.
[8]	RS485_NMM	RS-485 普通多点操作模式 (NMM) 1 = 使能RS-485 普通多点操作模式 (NMM) 0 = 禁用 RS-485普通多点操作模式 (NMM) 注: 在 RS-485_AAD 操作模式下无效.
[7:0]	保留	保留

UART 功能选择寄存器 (UA_FUN_SEL)

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

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

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

5.13 PS/2 设备控制器 (PS2D)

5.13.1 概述

PS/2 设备控制器为PS/2通讯提供基本时序控制。所有在设备和主机之间的通讯都是通过CLK 和 DATA 引脚控制。不同于 PS/2 键盘和鼠标设备控制器，接收/传输代码需要固件进行代码转换成有意义的代码。在接收到发送请求后启动控制器发送CLK 信号，但是在通信过程中主机拥有最终的控制权。主机发送到设备的数据是在上升沿读取，设备发送到主机的数据在上升沿之后被改变，设备向主机发送数据。16 个字节的 FIFO 应用减少CPU的介入。S/W 可选择 1 ~ 16 字节的连续传输。

5.13.2 特性

- 主机通讯禁止和发送侦测请求
- 接收帧错误侦测
- 可编程1 ~ 16 字节传输缓冲以减少 CPU 干扰
- 双数据缓冲功能
- S/W override总线

5.13.3 系统框图

PS/2 设备驱动器 包括APB 界面和时序逻辑，用于DATA 和 CLK lines.

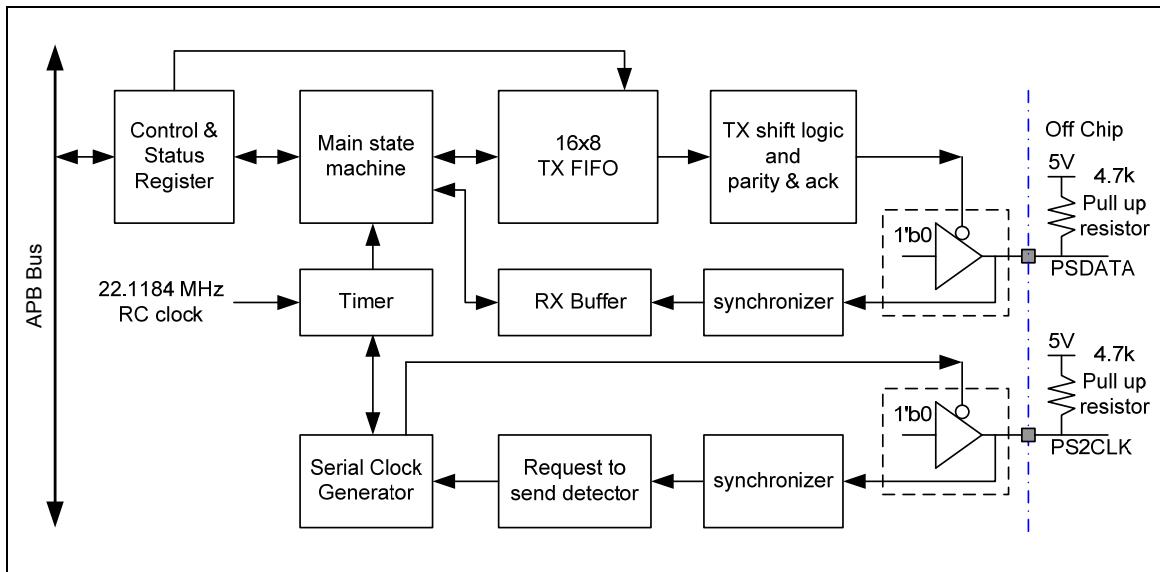


图 5-68 PS/2 设备框图

5.13.4 功能描述

5.13.4.1 通信

PS/2 设备具有双向同步串行协议.当总线为 "Idle" 模式并且两条线都为高 (open-collector), 该状态为设备允许开始 DATA 传输唯一条件. 主机在总线上有最终的控制权 , 并且任何时刻都可以通过下拉CLK line禁止通讯.

设备始终产生CLK信号. 如果主机需要发送 DATA, 下拉CLK line 为低, 禁止从设备进行通讯. 主机随后将DATA 拉低并且释放CLK. 这是"Request-to-Send" 状态, 通知设备开始发送 CLK 脉冲.

DATA	CLK	总线状态
High	High	Idle
High	Low	禁止通讯
Low	High	主机请求发送

所有数据每次传输 1 字节, 每字节包括 11-12 位. 如下:

- 1个开始位. 一直为 0
- 8个数据位, 先是最低位
- 1 个奇偶位(odd parity)
- 1 个停止位. 该位一直为1
- 1 个应答位 (主机~设备通讯)

如果有偶数个1奇偶校验位将置1, 如果有奇数个1奇偶校验位将清0, 数据位中的1加上奇偶位合计成奇数置1, 这应用于error 侦测. 设备需要检查该位, 如果该位错误, 将做接收错误的响应. 主机可通过拉底Clock line 至少100ms 来禁止通讯.如果11th 时钟脉冲之前停止传输, 设备将退出当前传输 , 当主机释放Clock, 设备将准备重新传输当前数据.为了有充足的时间应用于主机软件解码, 通过设定RXINT位封锁传输逻辑, S/w 必须清RXINT位来开始传输. S/W 可写 CLRFIFO 为 1 来复位 FIFO 指针.

设备向主机传输

设备应用于连续的 11-位 架构. 如下:

- 1 个开始位. 内容始终为 0
- 8 个数据位. 最高位最先传输
- 1个奇偶位(odd parity)
- 1个停止位. 该位内容一直为1

当CLK为高, 设备写 1 位数据到总线, CLK为低时, 通过主机读数据.见图 5-.

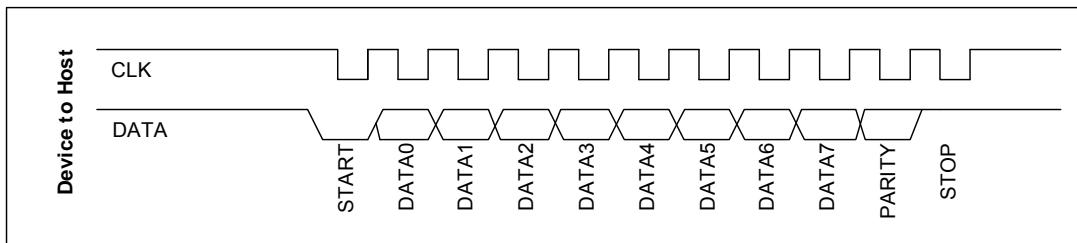


图 5-69 设备向主机传输数据格式

主机向设备传输：

首先, PS/2 设备一直产生 CLK 信号. 如果主机希望发送DATA, 首先需设定CLK 和DATA在 "请求发送" 状态如下:

- 拉低CLK 至少100 us 禁止通讯
- 应用 "Request-to-send" 拉 DATA 为低, 释放CLK

设备在不超过10 ms以内需不间断的监控状态.当设备监控到此状态, 将开始产生CLK 信号和8位DATA位 1 位停止位.当 CLK线为低时主机改变DATA 线.当 CLK 为高, 设备读取数据.

接收到停止位后设备发出应答信号, 使DATA线为低并且产生CLK 脉冲. 如果第十一个CLK 脉冲后主机未释放DATA 线, 设备将继续产生CLK脉冲直到 DATA线释放 (设备将产生一次 error.).

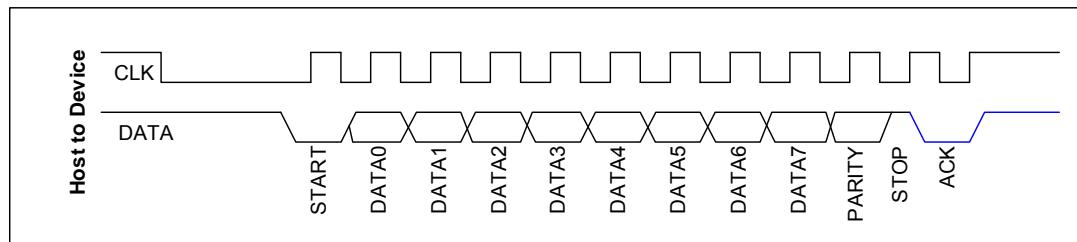


图 5-70 主机向设备传输的数据格式

主机和外设 DATA 和 CLK 详细时序框图如下：

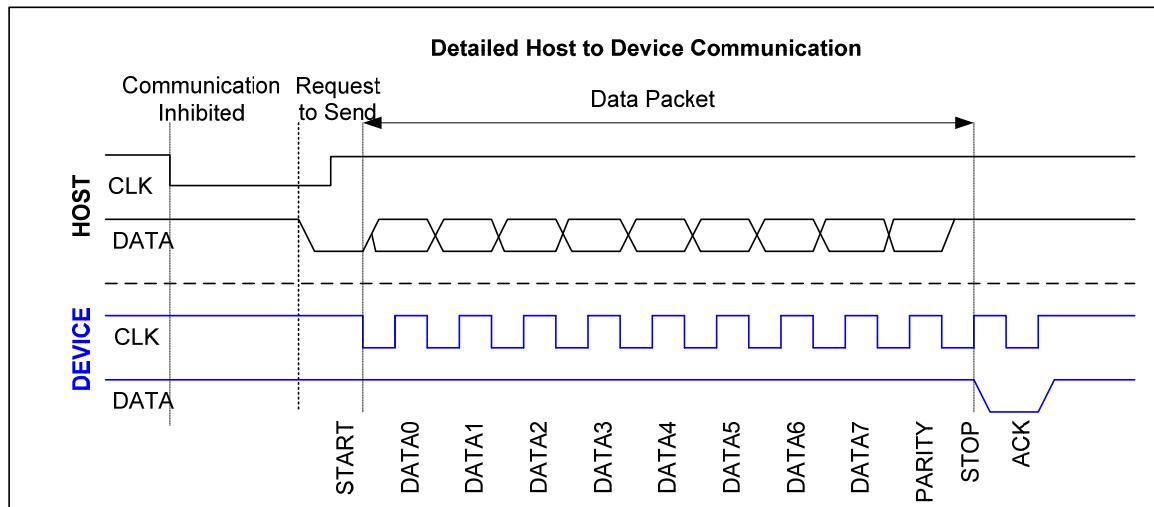


图 5-71 PS/2 Bit 数据格式

5.13.4.2 PS/2总线时序规范

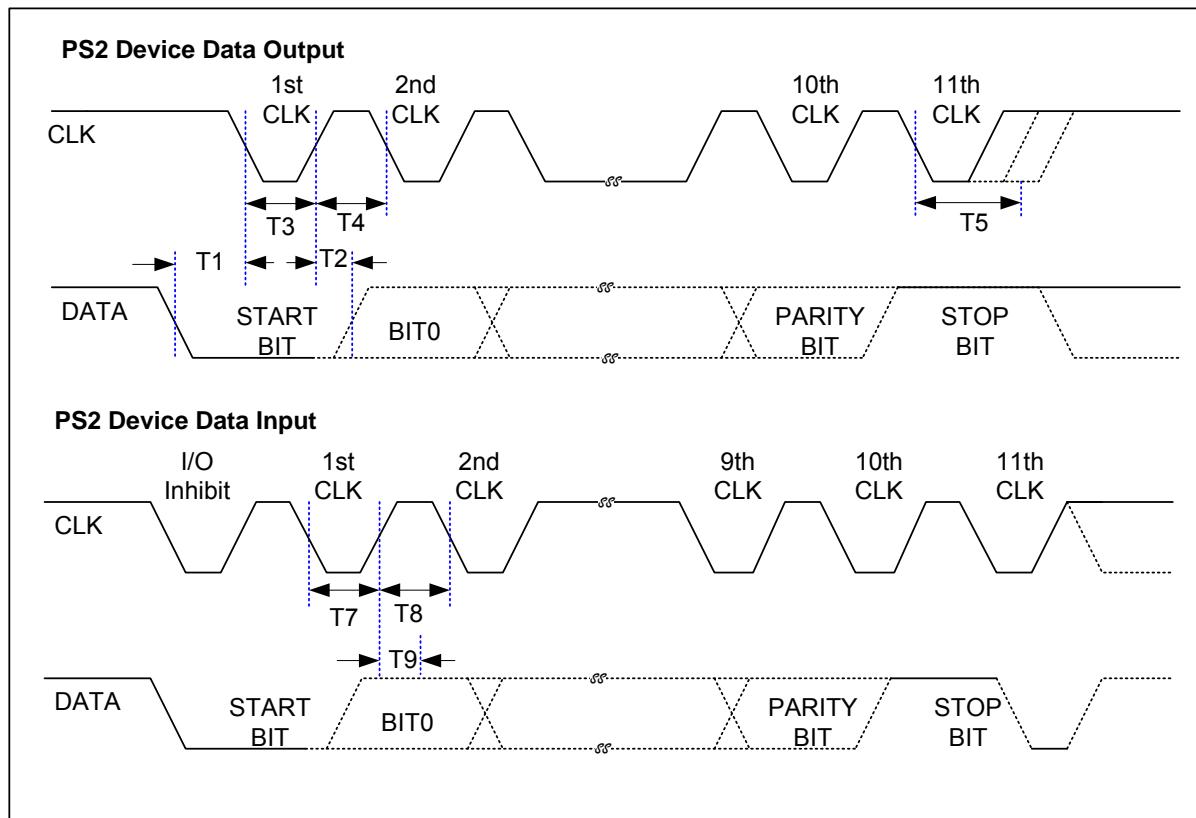


图 5-72 PS/2 总线时序

符号	时序参数	Min	Max
T1	CLK 下降沿 DATA 传输	5us	25us
T2	CLK 上升沿 DATA 传输	5us	T4-5us
T3	CLK 持续无效	30us	50us
T4	CLK持续有效	30us	50us
T5	11 th clock后 辅助设备禁止其他传输	>0	50us
T7	CLK 持续无效	30us	50us
T8	CLK持续有效	30us	50us
T9	CLK 传输从无效到有效时间, 辅助设备取样时间	5us	25us

5.13.4.3 TX FIFO操作

写PS2TXDATA0寄存器将触发设备和主机通讯. 在向TX FIFO写发送数据之前, S/W需要定义 TXFIFO的长度. 在写PS2TXDATA0寄存器後100us, 第一个字节的start bit才会被传送到总线上。如果要传送的数据超过四个字节, S/W在第四个字节数据传输完成前可写剩余数据到PS2TXDATA1-3. 2次连续字节将延时100us.

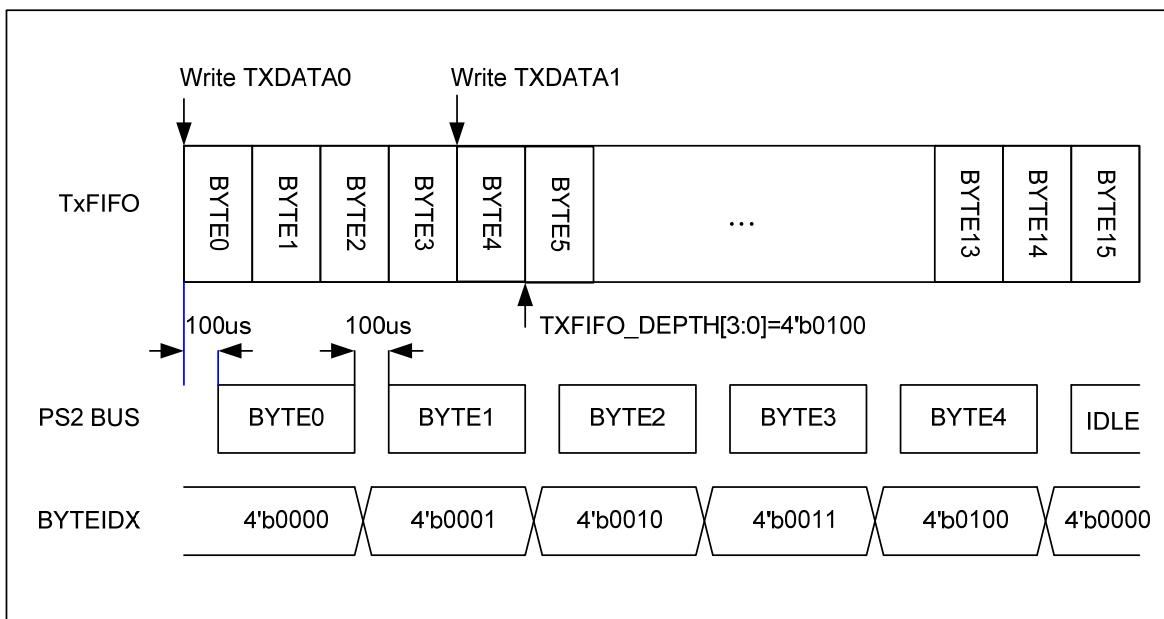


图 5-73 PS/2 数据格式

5.13.5 寄存器映射

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

寄存器	偏移量	R/W	描述	复位后的值
PS2_BA: 0x4010_0000				
PS2CON	PS2_BA+0x00	R/W	PS/2 控制寄存器	0x0000_0000
PS2TXDATA0	PS2_BA+0x04	R/W	PS/2 发送数据寄存器 0	0x0000_0000
PS2TXDATA1	PS2_BA+0x08	R/W	PS/2 发送数据寄存器 1	0x0000_0000
PS2TXDATA2	PS2_BA+0x0C	R/W	PS/2 发送数据寄存器 2	0x0000_0000
PS2TXDATA3	PS2_BA+0x10	R/W	PS/2 发送数据寄存器 3	0x0000_0000
PS2RXDATA	PS2_BA+0x14	R	PS/2 接收数据寄存器	0x0000_0000
PS2STATUS	PS2_BA+0x18	R/W	PS/2 状态寄存器	0x0000_0083
PS2INTID	PS2_BA+0x1C	R/W	PS/2 中断判定寄存器	0x0000_0000

5.13.6 寄存器描述

PS/2 控制寄存器 (PS2CON)

寄存器	偏移量	R/W	描述	复位后的值
PS2CON	PS2_BA + 0x00	R/W	PS/2 控制寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留				FPS2DAT	FPS2CLK	OVERRIDE	CLRFIFO
7	6	5	4	3	2	1	0
ACK	TXFIFO_DEPTH				RXINTEN	TXINTEN	PS2EN

Bits	描述	
[31:12]	保留	保留
[11]	FPS2DAT	<p>Force PS2DATA Line 若OVERRIDE =1, 它将强制 PS2DATA低或高, 忽略设备控制器的内部状态 1 = 强制 PS2DATA 高 0 = 强制 PS2DATA 低</p>
[10]	FPS2CLK	<p>Force PS2CLK Line 若OVERRIDE =1,它将强制 PS2CLK 线为低或高, 忽略设备控制器的内部状态 1 = 强制 PS2CLK 高 0 = 强制 PS2CLK 低</p>
[9]	OVERRIDE	<p>软件无视 PS2 CLK/DATA 引脚 状态 1 = PS2CLK 和 PS2DATA pins受S/W控制 0 = PS2CLK 和 PS2DATA pins受内部状态机控制.</p>
[8]	CLRFIFO	<p>清 TX FIFO 向该位写1将终止设备向主机传输. TXEMPTY 位 PS2STATUS 将置1并且 BYTEINDEX 将清0, 无论数据缓冲器中是否有数据. 数据缓冲器将不被清除. 1 = 清 FIFO</p>

		0 = 无效
[7]	ACK	<p>应答使能</p> <p>1 = 如果奇偶error 或停止位未接收到, 在12th 时钟将不发送应答.</p> <p>0 = 在12th 时钟从主机到设备通讯中一直发送应答位.</p>
[6:3]	TXFIFODIPTH	<p>FIFO 传输数据长度</p> <p>16 位缓冲器 应用于数据传输. S/W 可定义 FIFO 长度从 1 ~ 16 字节.</p> <p>0 = 1字节 1 = 2字节 ... 14 = 15字节 15 = 16字节</p>
[2]	RXINTEN	<p>使能接收中断</p> <p>1 = 使能数据接收完成中断 0 = 禁用数据接收完成中断</p>
[1]	TXINTEN	<p>使能传输中断</p> <p>1 = 使能数据传输完成中断 0 = 禁用数据传输完成中断</p>
[0]	PS2EN	<p>使能 PS/2 设备</p> <p>使能 PS/2 设备</p> <p>1 = 使能 0 = 禁用</p>

PS/2 TX数据寄存器 0-3 (PS2TXDATA0-3)

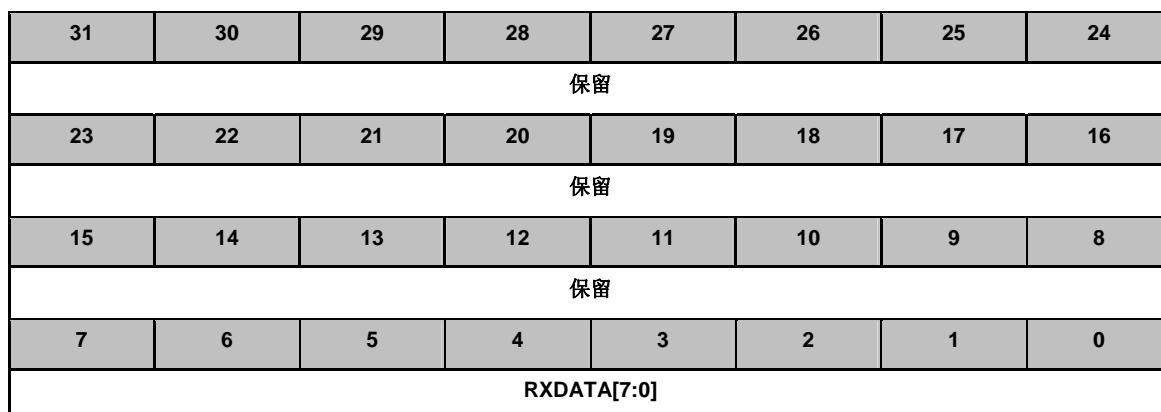
寄存器	偏移量	R/W	描述	复位后的值
PS2TXDATA0	PS2_BA + 0x04	R/W	PS/2 发送数据寄存器 0	0x0000_0000
PS2TXDATA1	PS2_BA + 0x08	R/W	PS/2 发送数据寄存器 1	0x0000_0000
PS2TXDATA2	PS2_BA + 0x0C	R/W	PS/2 发送数据寄存器 2	0x0000_0000
PS2TXDATA3	PS2_BA + 0x10	R/W	PS/2 发送数据寄存器 3	0x0000_0000

31	30	29	28	27	26	25	24
PS2TXDATAx[31:24]							
23	22	21	20	19	18	17	16
PS2TXDATAx[23:16]							
15	14	13	12	11	10	9	8
PS2TXDATAx[15:8]							
7	6	5	4	3	2	1	0
PS2TXDATAx[7:0]							

Bits	描述	
[31:0]	PS2TXDATAx	发送数据 如果总线在IDLE状态向寄存器写数据来进行设备和主机通讯。S/w在写数据到TX 缓冲器前需使能 PS2EN.

PS/2 Receiver 数据寄存器 (PS2RXDATA)

寄存器	偏移量	R/W	描述	复位后的值
PS2RXDATA	PS2_BA + 0x14	R/W	PS/2 接收数据寄存器	0x0000_0000



Bits	描述	
[31:8]	保留	保留
[7:0]	PS2RXDATA	接收数据 在主机到设备传输中，在应答位发送后，接收数据将从shift 寄存器拷贝到PS2RXDATA 寄存器. CPU 需在下一字节接收完成前读取此寄存器，否则数据将被改写而无效，并且RXOVF 位 PS2STATUS[6]将置 1

PS/2 状态寄存器 (PS2STATUS)

寄存器	偏移量	R/W	描述	复位后的值
PS2STATUS	PS2_BA + 0x18	R/W	PS/2 状态寄存器	0x0000_0000

31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留				BYTEIDX[3:0]			
7	6	5	4	3	2	1	0
TXEMPTY	RXOVF	TXBUSY	RXBUSY	RXPARTY	FRAMERR	PS2DATA	PS2CLK

Bits	描述			
[31:12]	保留	保留		
[11:8]	BYTEIDX	字节目录 表示哪个数据字节在 shift 寄存器 .当 FIFO 所有数据传输完毕，该位清 0..		
		BYTEIDX	DATA Transmit	BYTEIDX
		0000	TXDATA0[7:0]	1000
		0001	TXDATA0[15:8]	1001
		0010	TXDATA0[23:16]	1010
		0011	TXDATA0[31:24]	1011
		0100	TXDATA1[7:0]	1100
		0101	TXDATA1[15:8]	1101
		0110	TXDATA1[23:16]	1110
		0111	TXDATA1[31:24]	1111
[7]	TXEMPTY	TX FIFO 空 如果PS2EN 使能当 S/w 写任何数据到 PS2TXDATA0-3 将置TXEMPTY 位为 0， 当传输 数据的字节数和 FIFODEPTH 相等， TXEMPTY 将置 1. 1 = FIFO 为空 0 = 数据传输 只读位.		

[6]	RXOVF	RX Buffer 改写 1 =PS2RXDATA 数据被新来的数据改写. 0 =无 改写 写 1 清该位
[5]	TXBUSY	发送忙 该位表示 PS/2 驱动器当前正在发送数据. 1 = 当前正在发送数据 0 = 空闲状态. 只读位.
[4]	RXBUSY	接收忙 该位表示 PS/2 驱动器当前正在接收数据. 1 = 当前正在接收数据. 0 =空闲状态. 只读位.
[3]	RXPARTY	接收奇偶 该位表示最后接收数据奇偶位. (odd parity). 只读位.
[2]	FRAMERR	帧错误 对于主机与设备间的通信，如果未接收到STOP位表示其为帧错误，DATA线在12th时钟后仍保持低状态. 此时，S/w无视PS2CLK一直发送时钟直到PS2DATA释放成高状态后才停止发送时钟. 然后，驱动器发送“再传”命令到主机. 1 = 出现帧错误 0 = 未发生帧错误 写 1 清该位.
[1]	PS2DATA	DATA Pin 状态 该位表示同步和取样后PS2DATA线的状态.
[0]	PS2CLK	CLK Pin 状态 该位表示同步后PS2CLK线的状态.

PS/2 中断识别寄存器 (PS2INTID)

寄存器	偏移量	R/W	描述	复位后的值
PS2INTID	PS2_BA + 0x1C	R/W	PS/2 中断识别寄存器	0x0000_0000

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

Bits	描述	
[31:2]	保留	保留
[1]	TXINT	传输中断 当 STOP 位传输后, 该位置 1. 如果 TXINTEN 位置 1, 中断产生. 1 = 传输中断发生 0 = 无中断 写 1 清该位为 0.
[0]	RXINT	接收中断 在主机向设备传输时, 当应答位发送, 该位置位.如果 RXINTEN置 1, 中断产生. 1 = 接收中断发生 0 = 无中断 写 1 清该位为 0.

6 FLASH 内存控制器 (FMC)

6.1 概述

NuMicro™ NUC122 具有64/32k 字节的片上FLASH，用于存储应用程序（APROM），用户可以通过ISP更新FLASH中的程序。在系统编程（ISP）允许用户更新焊接在PCB板上的芯片中的程序。上电后，通过设置Config0确定 Cortex™-M0 CPU 从APROM或LDROM读取代码。NuMicro™ NUC122为用户提供4K字节的数据FLASH用于在芯片断电之前存储一些应用所需的数据。

6.2 特征

- 零等待状态，可达 60 MHz 的连续的地址访问
- 64/32KB 应用程序内存 (APROM)
- 4KB 在系统编程 (ISP) 加载程序内存 (LDROM)
- 固定的4KB数据FLASH，带有512字节页擦除单元
- 在系统编程 (ISP) 用于更新片上FLASH

6.3 框图

FLASH内存控制器由AHB从接口，ISP控制逻辑，烧写接口和FLASH宏接口时序控制逻辑组成。FLASH内存控制器框图如下所示：

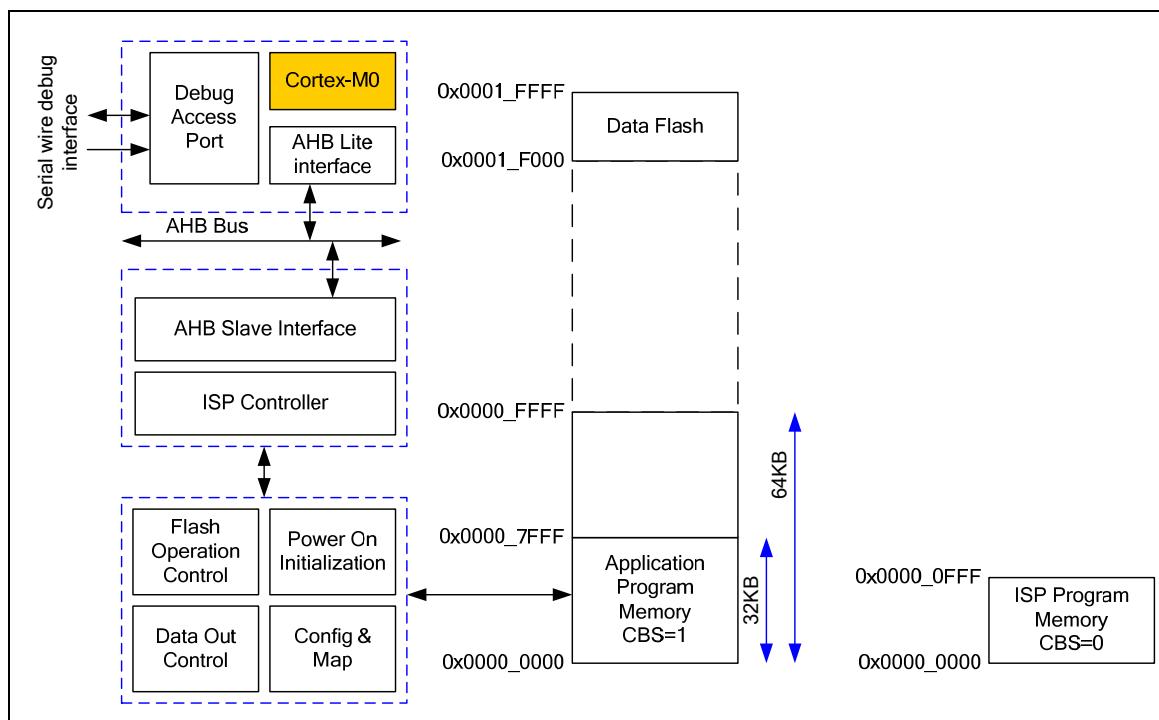


图 6-1 Flash 内存控制框图

6.4 Flash 内存结构

NuMicro™ NUC122 FLASH 内存由程序内存（64/32kB APROM），数据FLASH，ISP内存（LDROM），以及用户配置区域组成。用户配置区域提供几个字控制系统逻辑，如flash安全加密，启动选择，欠压电平，数据FLASH地址等。在上电期间，由FLASH内存加载到相应的控制寄存器中，在芯片上PCB之前，用户可以用烧写器来设置这些位，.数据FLASH的大小为4KB，开始地址为0x0001_F000。

区块名称	大小	开始地址	结束地址
AP-ROM	32/64 KB	0x0000_0000	0x0000_7FFF (32KB) 0x0000_FFFF (64KB)
保留 for future use	960 KB	0x0001_0000	0x000F_FFFF
Data Flash	4 KB	0x0001_F000	0x0001_FFFF
LD-ROM	4 KB	0x0010_0000	0x0010_0FFF
User Configuration	1 words	0x0030_0000	0x0030_0000

表 6-1 内存地址表

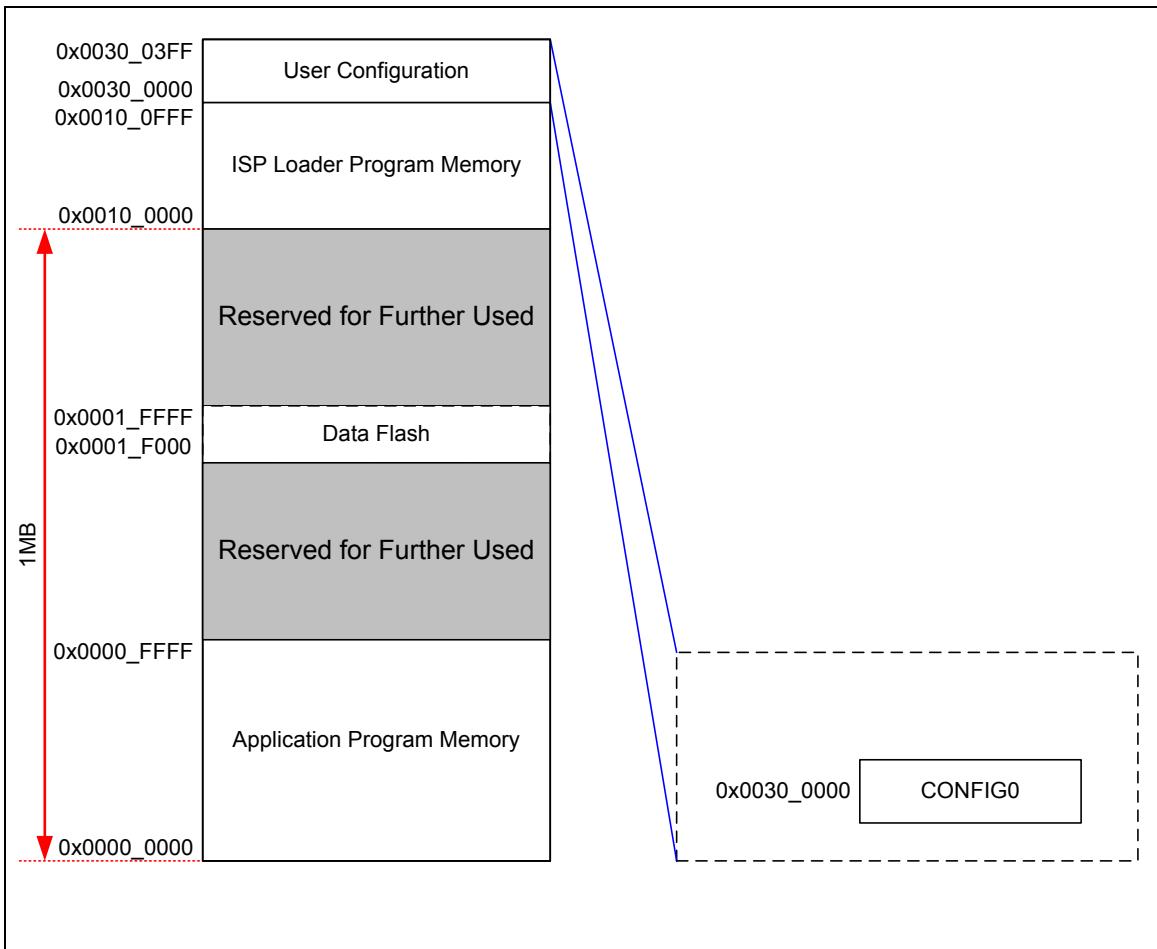


图 6-2 Flash 内存组织

6.5 启动选择

NuMicro™ NUC122 提供在系统编程 (ISP) 特征，允许用户直接更新PCB板上芯片中的程序。提供 4kB 程序内存用于存储ISP固件。用户可以通过设置Config0中的CBS位来选择从APROM还是从LDROM取指令开始程序。

6.6 数据 Flash

NuMicro™ NUC122 为用户提供数据FLASH。通过ISP过程读/写。擦除单元为512字节。若要改变一个字，需要先把所有128个字拷贝到另外页或SRAM中。数据FLASH的大小为4KB，开始地址为0x0001_F000。

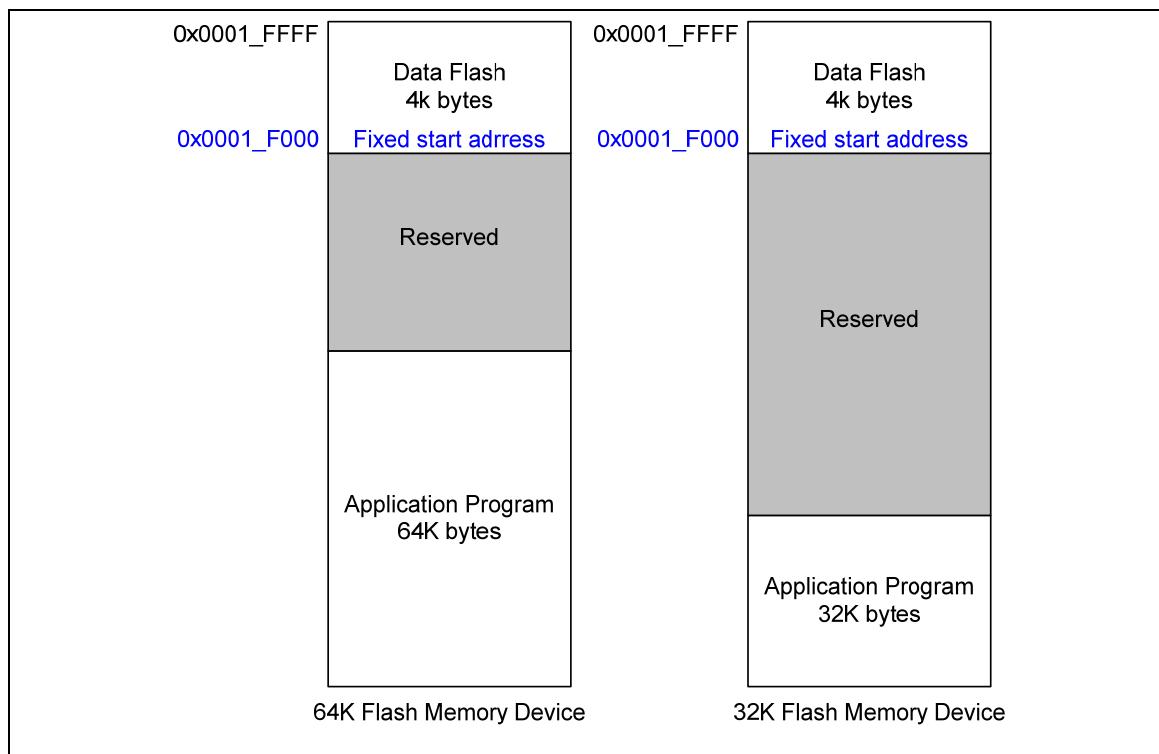


图 6-3 Flash 内存结构

6.7 用户配置

Config0 (Address = 0x0030_0000)

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

Bits	描述									
[31:29]	保留	保留								
[28]	CKF	XT1 时钟滤波使能 0 = 禁用 XT1 时钟滤波 1 = 使能 XT1 时钟滤波								
[27]	保留	保留								
[26:24]	CFOSC	复位后CPU时钟源选择 <table border="1"> <tbody> <tr> <td>FOSC[2:0]</td> <td>时钟源</td> </tr> <tr> <td>000</td> <td>外部 4~24 MHz 高速晶振时钟</td> </tr> <tr> <td>111</td> <td>内部 22.1184 MHz 高速振荡器时钟</td> </tr> <tr> <td>Others</td> <td>保留</td> </tr> </tbody> </table> 复位后，CFOSC的值将被加载到系统寄存器CLKSEL0.HCLK_S[2:0].	FOSC[2:0]	时钟源	000	外部 4~24 MHz 高速晶振时钟	111	内部 22.1184 MHz 高速振荡器时钟	Others	保留
FOSC[2:0]	时钟源									
000	外部 4~24 MHz 高速晶振时钟									
111	内部 22.1184 MHz 高速振荡器时钟									
Others	保留									
[23]	CBODEN	欠压检测使能 0= 上电后，使能欠压检测 1= 上电后，禁用欠压检测								

[22:21]	CBOV1-0	欠压电压选择
		CBOV1 CBOV0 Brown out voltage
		1 1 4.5V
		1 0 3.8V
		0 1 2.7V
		0 0 2.2V
[20]	CBORST	欠压复位使能 0 = 上电后，使能欠压复位 1 = 上电后，禁用欠压复位
[19:8]	保留	保留
[7]	CBS	启动选项 0 = 芯片由LDROM启动 1 = 芯片由APROM启动
[6:2]	保留	保留
[1]	LOCK	安全加密 0 = 加密FLASH数据 1 = 解除Flash 数据加密 当FLASH数据被加密，仅有器件ID， Config0 和Config1 可被烧写器和ICP通过串口调试接口读取. 其他数据锁定为0xFFFFFFFF. 无论数据是否锁定，ISP都可以读取
[0]	保留	保留

6.8 在系统编程 (ISP)

程序内存和数据FLASH 支持硬件编程和在系统编程 (ISP). 硬件编程模式采用批量写，以方便量产阶段进行编程。若产品还在开发阶段或终端用户需要升级固件时，硬件编程模式不是很方便，ISP 模式能更好地适用于这种情况。NuMicro™ NUC122 支持 ISP 模式，即通过软件控制来对器件重新编程。而且，这也使得更新固件得以广泛应用。

ISP 可以在没有将微控器从系统中取下来的情况下执行编程。各种接口使得LDROM更容易更新程序代码。最常用的方法是通过UART和在LDROM中的固件执行ISP，PC一般都是通过串口传输新的APROM代码。LDROM接收后，通过ISP，重新对APROM编程。Nuvoton 提供用于NuMicro™ NUC122 的ISP 固件和 PC 应用程序。这让用户可以通过Nuvoton ISP工具非常方便地执行ISP.

6.8.1 ISP程序

NuMicro™ NUC122 支持从 APROM 或 LDROM 启动，而这是首先由用户对CBS位进行配置来定义的。用户想更新APROM中的应用程序时，可以写BS=1，并开始软件复位使芯片由LDROM启动。开始ISP功能的第一步是向ISPEN写1。在向ISPCON寄存器写数据之前，S/W 需要向全局控制寄存器（GCR, 0x5000_0100）的寄存器 RegLockAddr 写入0x59, 0x16 和 0x88，这个过程用于保护FLASH内存免受因为在上电或断电期间无意识的写操作而造成的损坏。

向ISPGO写入数据后，要检查几个错误条件。如果错误条件产生时，ISP操作失败，其失败标志置位，ISPFF标志由软件清零，而不会在下次ISP操作时被覆盖，即使ISPFF保持为“1”，下一次ISP也可以开始。如果ISPFF被设置为1了，建议在每次ISP操作后，由软件检查ISPFF并将其清零。

ISPGO置位，CPU将等待ISP操作结束，在此期间，外设仍然在工作，如果有中断请求时，CPU仍然会先执行完ISP后再响应中断。

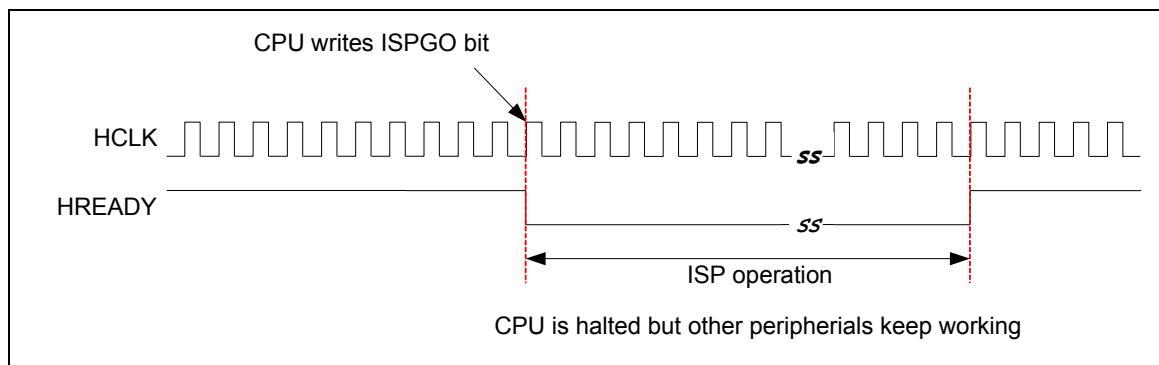


图 6-4 ISP 操作时序图

注 NuMicro™ NUC122 允许用户通过ISP更新CONFIG的值。

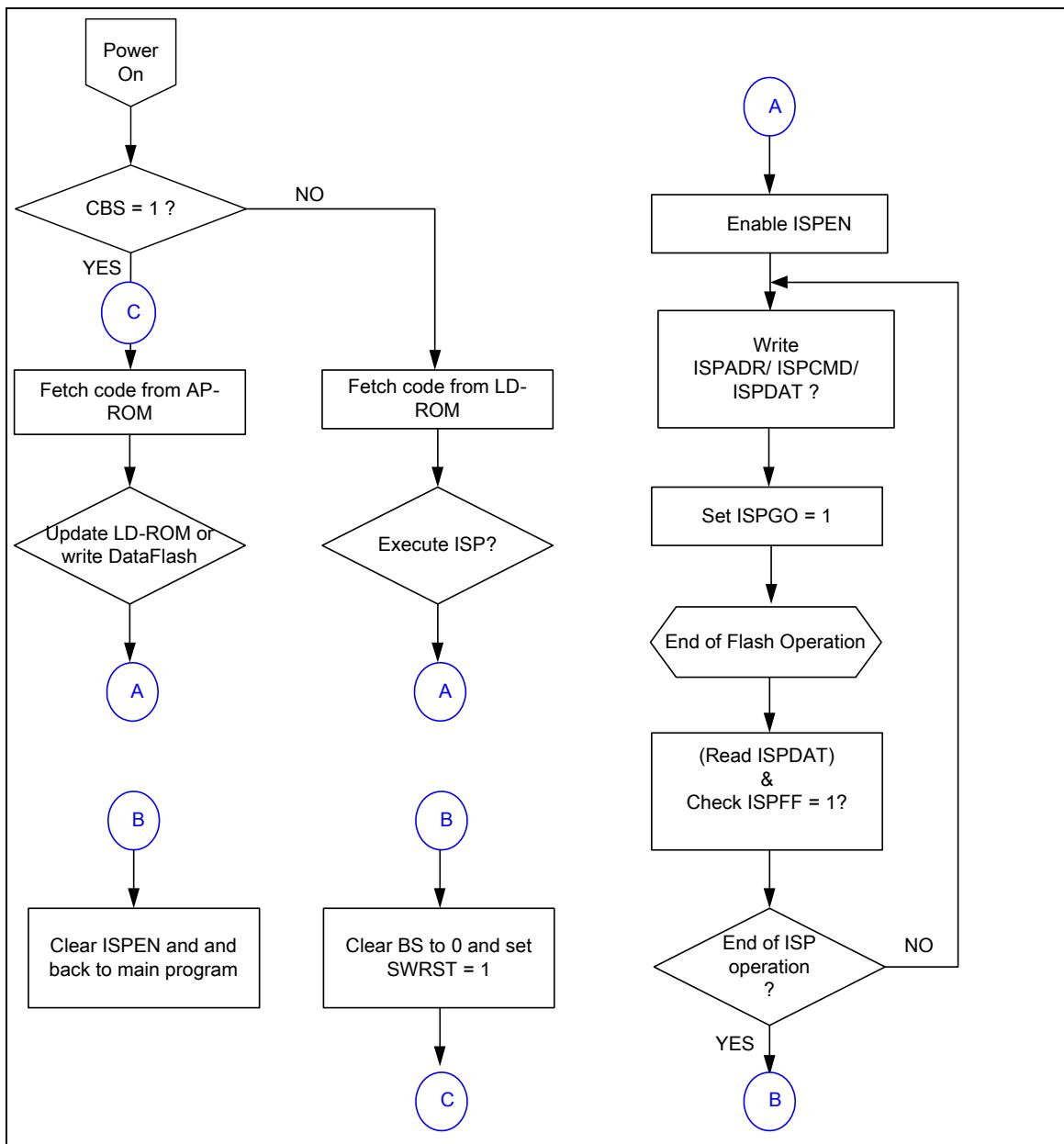


图 6-5 ISP 操作流图

ISP Mode	ISPCMD			ISPADR			ISPDAT
	FOEN	FCEN	FCTRL[3:0]	A21	A20	A[19:0]	D[31:0]
FLASH Page Erase	1	0	0010	0	A20	Address in A[19:0]	x
FLASH Program	1	0	0001	0	A20	Address in A[19:0]	Data in D[31:0]
FLASH Read	0	0	0000	0	A20	Address in A[19:0]	Data out D[31:0]
CONFIG Page Erase	1	0	0010	1	1	Address in A[19:0]	X
CONFIG Program	1	0	0001	1	1	Address in A[19:0]	Data in D[31:0]
CONFIG Read	0	0	0000	1	1	Address in A[19:0]	Data out D[31:0]

表 6-2 ISP 模式

6.9 寄存器映射

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

寄存器	偏移量	R/W	描述	复位后的值
Base Address (FMC_BA) : 0x5000_C000				
ISPCON	FMC_BA+0x000	R/W	ISP 控制寄存器	0x0000_0000
ISPADR	FMC_BA+0x004	R/W	ISP 地址寄存器	0x0000_0000
ISPDAT	FMC_BA+0x008	R/W	ISP 数据寄存器	0x0000_0000
ISPCMD	FMC_BA+0x00C	R/W	ISP 命令寄存器	0x0000_0000
ISPTRG	FMC_BA+0x010	R/W	ISP 触发寄存器	0x0000_0000
FATCON	FMC_BA+0x018	R/W	FLASH 访问窗口控制寄存器	0x0000_0000

6.10 寄存器描述

ISP 控制寄存器 (ISPCON)

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

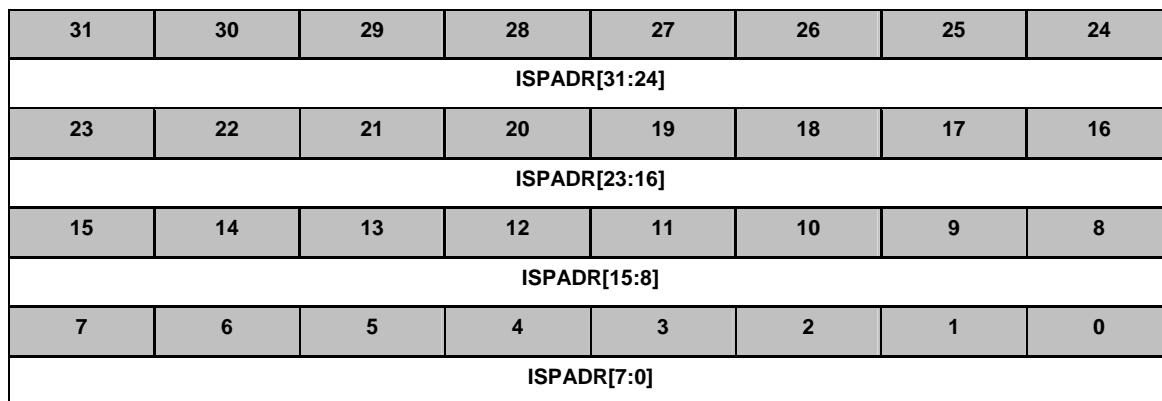
31	30	29	28	27	26	25	24
保留							
23	22	21	20	19	18	17	16
保留							
15	14	13	12	11	10	9	8
保留	ET			保留	PT		
7	6	5	4	3	2	1	0
保留	ISPFF	LDUEN	CFGUEN	保留		BS	ISOPEN

Bits	描述						
[31:15]	保留	保留					
[14:12]	ET[2:0]	Flash 擦除时间 (写保护位)					
		ET[2]	ET[1]	ET[0]	擦除时间 (ms)		
		0	0	0	20 (default)		
		0	0	1	25		
		0	1	0	30		
		0	1	1	35		
		1	0	0	3		
		1	0	1	5		
		1	1	0	10		
		1	1	1	15		
[11]	保留	保留					

[8:10]	PT[2:0]	Flash 编程时间 (写保护位)																																		
		<table border="1"> <thead> <tr> <th>PT[2]</th><th>PT[1]</th><th>PT[0]</th><th>编程时间 (us)</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>40</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>45</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>50</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>55</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>20</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>25</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>30</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>35</td></tr> </tbody> </table>	PT[2]	PT[1]	PT[0]	编程时间 (us)	0	0	0	40	0	0	1	45	0	1	0	50	0	1	1	55	1	0	0	20	1	0	1	25	1	1	0	30	1	1
PT[2]	PT[1]	PT[0]	编程时间 (us)																																	
0	0	0	40																																	
0	0	1	45																																	
0	1	0	50																																	
0	1	1	55																																	
1	0	0	20																																	
1	0	1	25																																	
1	1	0	30																																	
1	1	1	35																																	
[7] 保留 保留																																				
ISP失败标志 (写保护位) 当ISP满足下列条件时，该位由硬件置位： (1) APROM 写入本身. (2) LDROM 写入本身. (3) 如果CFGUEN设置为0， CONFIG 被擦除或编程. (4) 定义地址无效，如超过正常范围. 写 1 清标志.																																				
LDROM更新使能 (写保护位) LDROM 更新使能位。 1 = 当在APROM中运行时， LDROM 可以被更新. 0 = 禁止LDROM更新																																				
使能由ISP更新配置位 (写保护位) 1 = 使能ISP更新配置位 0 = 禁用ISP更新配置位																																				
[3:2] 保留 保留																																				
启动选择 (写保护位) 置位/清零该位选择下次是由LDROM启动还是由APROM启动，该位可作为MCU启动状态标志,用于检查MCU是由LDROM还是APROM启动的. 这一位在上电复位时被初始化为Config0的CBS位的反转值，在其他复位时保持不变 1 = 由LDROM启动 0 = 由APROM启动																																				
ISP 使能(写保护位) ISP 使能位，设置该位可以使能ISP功能. 1 = 使能 ISP 功能 0 = 禁用 ISP 功能																																				

ISP 地址 (ISPADR)

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



Bits	描述							
[31:0]	ISPADR	ISP地址 NuMicro™ NUC122 最大内置16kx32 的flash, 仅支持字编程. 执行ISP功能时, ISPARD[1:0] 必须为00b.						

ISP 数据寄存器 (ISPDAT)

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

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

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

ISP 命令 (ISPCMD)

寄存器	偏移量	R/W	描述	复位后的值
ISPCMD	FMC_BA+ 0x0C	R/W	ISP 命令寄存器	0x0000_0000

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

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

ISP 触发控制寄存器 (ISPTRG)

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

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

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

Flash 访问时间控制寄存器 (FATCON)

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

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

Bits	描述	
[31:7]	保留	保留
[6]	MFOM	中频优化模式(写何位) 如果操作频率在20M~40MHz, 此位写1, 芯片的工作效率更高, 工作频率若高于40MHz LFOM和MFOM必须都写0
[5]	保留	保留
[4]	LFOM	若芯片工作频率小于20MHz, 此位写1芯片的工作效率更高, 若工作频率高于40MHz, LFOM和MFOM必须都写为0
[3: 0]	保留	保留

7 电气特性

7.1 绝对最大额定值

参数	符号	最小值	最大值	单位
直流电源电压	VDD-VSS	-0.3	+7.0	V
输入电压	VIN	VSS-0.3	VDD+0.3	V
晶振频率	1/t _{CLCL}	4	24	MHz
工作温度	TA	-40	+85	°C
贮存温度	TST	-55	+150	°C
V _{DD} 最大流入电流		-	120	mA
V _{SS} 最大流出电流			120	mA
单一管脚最大灌电流			35	mA
单一管脚最大源电流			35	mA
所有管脚最大灌电流总合			100	mA
所有管脚最大源电流总合			100	mA

注: 上表所列的条件中, 其极限值可能对器件的稳定有反作用.

7.2 DC电气特性

7.2.1 NuMicro™ NUC122 DC 电气特性

(VDD-VSS=3.3V, TA = 25°C, FOSC = 60 MHz在无特别说明的情况下)

参数	符号.	明细表				测试条件
		最小值	典型值	最大值	单位	
工作电压	V _{DD}	2.5		5.5	V	V _{DD} = 2.5V ~ 5.5V up to 60 MHz
LDO输出电压	V _{LDO}	1.6	1.8	2.1	V	V _{DD} > 2.5V
模拟工作电压	A _{VDD}	2.1		V _{DD}	V	
普通模式下的工作电流 (60 MHz)	I _{DD1}		26		mA	V _{DD} = 5.5V@60 MHz, enable all IP and PLL, XTAL=12 MHz
	I _{DD2}		21		mA	V _{DD} = 5.5V@60 MHz, disable all IP and enable PLL, XTAL=12 MHz
	I _{DD3}		24		mA	V _{DD} = 3.3V@60 MHz, enable all IP and PLL, XTAL=12 MHz
	I _{DD4}		19		mA	V _{DD} = 3.3V@60 MHz, disable all IP and enable PLL, XTAL=12 MHz
普通模式下的工作电流 (12 MHz)	I _{DD5}		6.5		mA	V _{DD} = 5.5V@12 MHz, enable all IP and disable PLL, XTAL=12 MHz
	I _{DD6}		5		mA	V _{DD} = 5.5V@12 MHz, disable all IP and disable PLL, XTAL=12 MHz
	I _{DD7}		4.5		mA	V _{DD} = 3.3V@12 MHz, enable all IP and disable PLL, XTAL=12 MHz
	I _{DD8}		3.5		mA	V _{DD} = 3.3V@12 MHz, disable all IP and disable PLL, XTAL=12 MHz
普通模式下的工作电流 (4 MHz)	I _{DD9}		3.5		mA	V _{DD} = 5.5V@4 MHz, enable all IP and disable PLL, XTAL=4 MHz

参数	符号	明细表				测试条件
		最小值	典型值	最大值	单位	
空闲模式下的工作电流 (60 MHz)	I _{DD10}		3		mA	V _{DD} = 5.5V@4 MHz, disable all IP and disable PLL, XTAL=4 MHz
	I _{DD11}		3		mA	V _{DD} = 3.3V@4 MHz, enable all IP and disable PLL, XTAL=4 MHz
	I _{DD12}		2		mA	V _{DD} = 3.3V@4 MHz, disable all IP and disable PLL, XTAL=4 MHz
空闲模式下的工作电流 (12 MHz)	I _{IDLE1}		17		mA	V _{DD} = 5.5V@60 MHz enable all IP and PLL, XTAL=12 MHz
	I _{IDLE2}		12		mA	V _{DD} =5.5V@60 MHz, disable all IP and enable PLL, XTAL=12 MHz
	I _{IDLE3}		15		mA	V _{DD} = 3.3V@60 MHz, enable all IP and PLL, XTAL=12 MHz
	I _{IDLE4}		11		mA	V _{DD} = 3.3V@60 MHz, disable all IP and enable PLL, XTAL=12 MHz
空闲模式下的工作电流 (4 MHz)	I _{IDLE5}		4.5		mA	V _{DD} = 5.5V@12 MHz, enable all IP and disable PLL, XTAL=12 MHz
	I _{IDLE6}		3.5		mA	V _{DD} = 5.5V@12 MHz disable all IP and disable PLL, XTAL=12 MHz
	I _{IDLE7}		3		mA	V _{DD} = 3.3V@12 MHz, enable all IP and disable PLL, XTAL=12 MHz
	I _{IDLE8}		2		mA	V _{DD} = 3.3V@12 MHz, disable all IP and disable PLL, XTAL=12 MHz
空闲模式下的工作电流 (4 MHz)	I _{IDLE9}		3		mA	V _{DD} = 5.5V@4 MHz, enable all IP and disable PLL, XTAL=4 MHz

参数	符号	明细表				测试条件
		最小值	典型值	最大值	单位	
掉电模式下工作电流 (掉电模式)	I _{IDLE10}		2.5		mA	V _{DD} = 5.5V@4 MHz, disable all IP and disable PLL, XTAL=4 MHz
	I _{IDLE11}		2		mA	V _{DD} = 3.3V@4 MHz, enable all IP and disable PLL, XTAL=4 MHz
	I _{IDLE12}		1		mA	V _{DD} = 3.3V@4 MHz, disable all IP and disable PLL, XTAL=4 MHz
PA, PB, PC, 输入电流 PD (准双向模式)	I _{PWD1}		13		μA	V _{DD} = 5.5V, RTC OFF, No load @ Disable BOV function
	I _{PWD2}		12		μA	V _{DD} = 3.3V, RTC OFF, No load @ Disable BOV function
	I _{PWD3}		15		μA	V _{DD} = 5.5V, RTC run , No load @ Disable BOV function
	I _{PWD4}		13		μA	V _{DD} = 3.3V, RTC run , No load @ Disable BOV function
PA, PB, PC, 输入电流 PD (准双向模式)	I _{IN1}	-60	-	+15	μA	V _{DD} = 5.5 V, V _{IN} = 0 V or V _{IN} =V _{DD}
输入电流/RESET ^[1]	I _{IN2}	-55	-45	-30	μA	V _{DD} = 3.3 V, V _{IN} = 0.45 V
PA, PB, PC, PD, 输入漏 电流	I _{LK}	-2	-	+2	μA	V _{DD} = 5.5V, 0<V _{IN} <V _{DD}
PA~PD逻辑1至0转换时 电流 (准双向模式)	I _{TL} ^[3]	-650	-	-200	μA	V _{DD} = 5.5V, V _{IN} <2.0V
PA, PB, PC, PD输入低电 压(TTL 输入)	V _{IL1}	-0.3	-	0.8	V	V _{DD} = 4.5V
		-0.3	-	0.6		V _{DD} = 2.5V
PA, PB, PC, PD输入高电 压(TTL 输入)	V _{IH1}	2.0	-	V _{DD} +0.2	V	V _{DD} = 5.5V
		1.5	-	V _{DD} +0.2		V _{DD} = 3.0V
PA, PB, PC, PD 输入低 电压(Schmitt 输入)	V _{IL2}	-0.5		0.4 V _{DD}	V	
PA, PB, PC, PD 输入高 电压(Schmitt 输入)	V _{IH2}	0.6 V _{DD}		V _{DD} +0.5	V	
PA~PD迟滞电压 (Schmitt 输入)	V _{HY}		0.2V _{DD}		V	

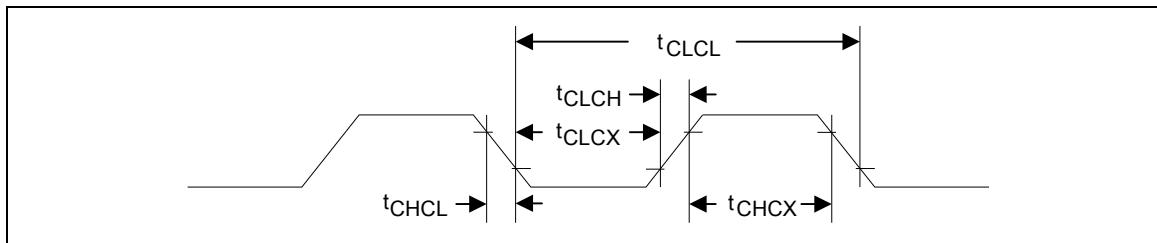
参数	符号	明细表				测试条件
		最小值	典型值	最大值	单位	
RESET脚 负向门槛电压 (Schmitt输入)	V_{ILS}	-0.5	-	$0.3V_{DD}$	V	
RESET脚 正向门槛电压 (Schmitt输入)	V_{IHS}	$0.7V_{DD}$	-	$V_{DD}+0.5$	V	
PA, PB, PC, PD源电流 (准双向模式)	I_{SR11}	-300	-370	-450	μA	$V_{DD} = 4.5V, V_S = 2.4V$
	I_{SR12}	-50	-70	-90	μA	$V_{DD} = 2.7V, V_S = 2.2V$
	I_{SR12}	-40	-60	-80	μA	$V_{DD} = 2.5V, V_S = 2.0V$
PA, PB, PC, PD源电流 (推挽模式)	I_{SR21}	-22	-28	-32	mA	$V_{DD} = 4.5V, V_S = 2.4V$
	I_{SR22}	-4	-6	-8	mA	$V_{DD} = 2.7V, V_S = 2.2V$
	I_{SR22}	-3	-5	-7	mA	$V_{DD} = 2.5V, V_S = 2.0V$
PA, PB, PC, PD灌电流 (准双向及推挽模式)	I_{SK1}	10	17	20	mA	$V_{DD} = 4.5V, V_S = 0.45V$
	I_{SK1}	7	10	13	mA	$V_{DD} = 2.7V, V_S = 0.45V$
	I_{SK1}	6	9	12	mA	$V_{DD} = 2.5V, V_S = 0.45V$
BOV_VL [1:0] =00b 时欠压电压	$V_{BO2.2}$	2.1	2.2	2.3	V	
BOV_VL [1:0] =01b 时欠压电压	$V_{BO2.7}$	2.6	2.7	2.8	V	
BOV_VL [1:0] =10b 时欠压电压	$V_{BO3.8}$	3.6	3.75	3.9	V	
BOV_VL [1:0] =11b 时欠压电压	$V_{BO4.5}$	4.2	4.4	4.6	V	
BOD电压迟滞范围	V_{BH}	30	-	150	mV	$V_{DD} = 2.5V\sim 5.5V$

注:

1. /RESET管脚为 Schmitt 触发 输入模式.
2. 晶振输入为CMOS 输入.
3. 当PA, PB, PC, 及 PD 管脚被外部由1驱动到0时, 可作为输出电流的源端, 在 $VDD=5.5V$ 时, 输出电流达到最大值, Vin 接近2V.

7.3 AC 电气特性

7.3.1 外部 4~24 MHz 高速晶振交流特征



注：占空比为50%。

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

7.3.2 外部 4~24 MHz 高速晶振

参数	条件	最小值	典型值	最大值	单位
输入时钟频率	外部晶振	4	12	24	MHz
温度	-	-40	-	85	℃

7.3.2.1 典型晶振应用电路

晶振	C1	C2	R
4 MHz ~ 24 MHz	不需要	不需要	不需要

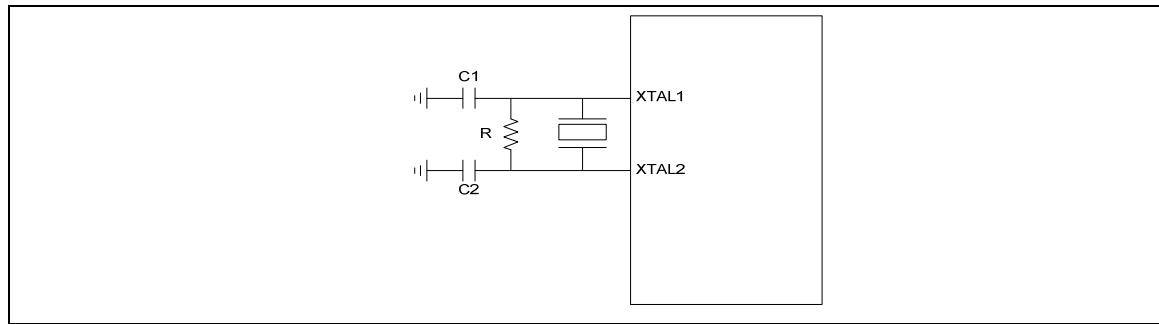


图 7-1 典型晶振应用电路

7.3.3 外部 32.768 KHz 低速晶振

参数	条件	最小值	典型值	最大值	单位
输入时钟频率	外部晶振	-	32.768	-	KHz
温度	-	-40	-	85	°C

7.3.4 内部 22.1184 MHz 高速振荡器

参数	条件	最小值	典型值	最大值	单位
中心频率	-	-	22.1184	-	MHz
校验内部振荡器频率	+25 °C; V _{DD} = 3.3 V	-1	-	+1	%
	-40 °C ~ +85 °C; V _{DD} = 2.5 V ~ 5.5 V	-5	-	+5	%

7.3.5 内部 10 KHz 低速振荡器

参数	条件	最小值	典型值	最大值	单位
中心频率	-	-	10	-	KHz
校验内部振荡器频率	+25 °C; V _{DD} = 5 V	-30	-	+30	%
	-40 °C ~ +85 °C; V _{DD} = 2.5 V ~ 5.5 V	-50	-	+50	%

注：内部的工作电压来自LDO

7.4 模拟特性

7.4.1 LDO和电源管理规格

参数	最小值	典型值	最大值	单位	备注
输入电压	2.5	5	5.5	V	V_{DD} 输入电压
输出电压	1.6	1.8	2.1	V	$V_{DD} > 2.5V$
温度	-40	25	85	°C	
静态电流 (PD=0)	-	100	-	uA	
静态电流 (PD=1)	-	5	-	uA	
Iload (PD=0)	-	-	100	mA	
Iload (PD=1)	-	-	100	uA	
Cbp	-	4.7	-	uF	Resr=1ohm

注:

1. 建议接一颗10uF 或更大的电容和一颗 100nF 旁路电容在VDD与VSS之间.
2. 为保证电源稳定, 要在LDO与VSS之间接一颗10uF 或更大的电容. 再加一颗100nF 的旁路电容在LDO与VSS之间有助于抑制输出噪声.



7.4.2 低压复位规格

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

7.4.3 欠压检测规格

参数	条件	最小值	典型值	最大值	单位
静态电流	AVDD=5.5V	-	-	140	μA
温度	-	-40	25	85	°C
欠压电压	BOV_VL[1:0]=11	4.2	44	4.6	V
	BOV_VL [1:0]=10	3.6	3.75	3.9	V
	BOV_VL [1:0]=01	2.6	2.7	2.8	V
	BOV_VL [1:0]=00	2.1	2.2	2.3	V
迟滞	-	30	-	150	mV

7.4.4 上电复位规格 (5 V)

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



7.4.5 USB PHY 规格

7.4.5.1 USB DC电气特性

符号	参数	条件	最小值	典型值	最大值	单位
V_{IH}	输入高 (driven)		2.0			V
V_{IL}	输入低				0.8	V
V_{DI}	差分输入	$ PDP-PADM $	0.2			V
V_{CM}	差分同模范围	Includes V_{DI} range	0.8		2.5	V
V_{SE}	单端接收器极限		0.8		2.0	V
	接收器滞后			200		mV
V_{OL}	输出低 (driven)		0		0.3	V
V_{OH}	输出高 (driven)		2.8		3.6	V
V_{CRS}	输出信号串扰电压		1.3		2.0	V
R_{PU}	上拉电阻		1.425		1.575	kΩ
R_{PD}	下拉电阻		14.25		15.75	kΩ
V_{TRM}	上行端口上的上拉电阻的极限电压 (R_{PU})		3.0		3.6	V
Z_{DRV}	驱动输出阻抗*	静态驱动*		10		Ω
C_{IN}	发射器容值	Pin to GND			20	pF

*驱动输出阻抗不包括串联电阻阻抗。

7.4.5.2 USB全速驱动器电气特性

符号	参数	条件	最小值	典型值	最大值	单位
T_{FR}	上升时间	$C_L=50p$	4		20	ns
T_{FF}	下降时间	$C_L=50p$	4		20	ns
T_{FRFF}	上升与下降时间比值	$T_{FRFF}=T_{FR}/T_{FF}$	90		111.11	%

7.4.5.3 USB 功耗

符号	参数	条件	最小值	典型值	最大值	单位
I_{VDDREG} (全速)	VDDD 和 VDDREG 供给电流(稳态)	待机	50			uA
		输入模式				
		输出模式				

7.5 SPI 动态特性

7.5.1 数据输入输出的动态特性

符号	参数	最小值	典型值	最大值	单位
SPI 主机模式 (VDD = 4.5V ~ 5.5V, 30pF 负载电容)					
t_{DS}	数据准备时间	16	10	-	ns
t_{DH}	数据保持时间	0	-	-	ns
t_v	数据输出有效时间	-	5	8	ns
SPI 主机模式 (VDD = 3.0V ~ 3.6V, 30pF 负载电容)					
t_{DS}	数据准备时间	20	13	-	ns
t_{DH}	数据保持时间	0	-	-	ns
t_v	数据输出有效时间	-	7	14	ns
SPI 从机模式 (VDD = 4.5V ~ 5.5V, 30pF 负载电容)					
t_{DS}	数据准备时间	0	-	-	ns
t_{DH}	数据保持时间	$2^{*}PCLK+4$	-	-	ns
t_v	数据输出有效时间	-	$2^{*}PCLK+11$	$2^{*}PCLK+20$	ns
SPI 从机模式 (VDD = 3.0V ~ 3.6V, 30pF 负载电容)					
t_{DS}	数据准备时间	0	-	-	ns
t_{DH}	数据保持时间	$2^{*}PCLK+8$	-	-	ns
t_v	数据输出有效时间	-	$2^{*}PCLK+20$	$2^{*}PCLK+32$	ns

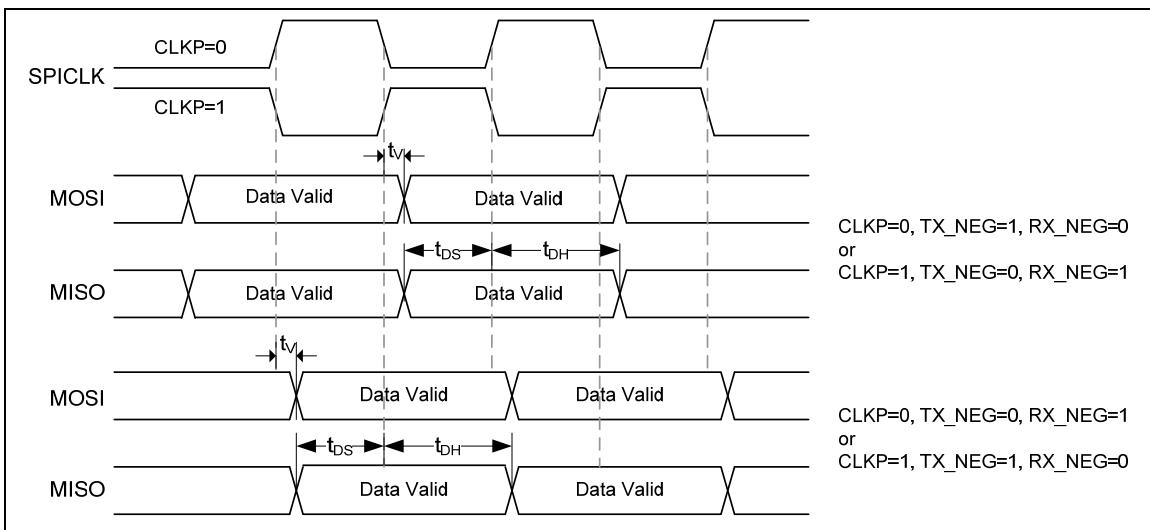


图 7-2 SPI 主机动态特性时序图

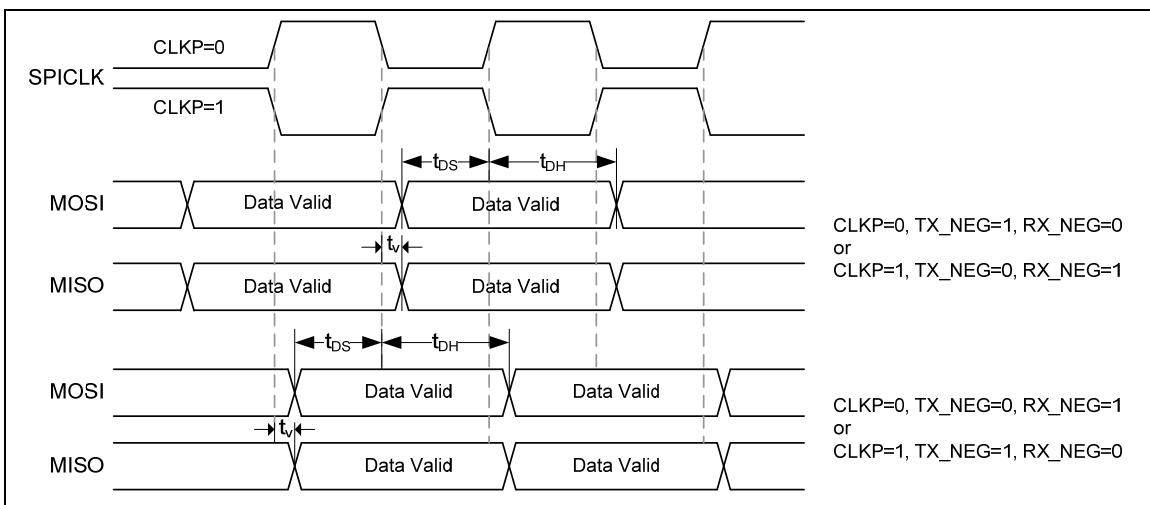
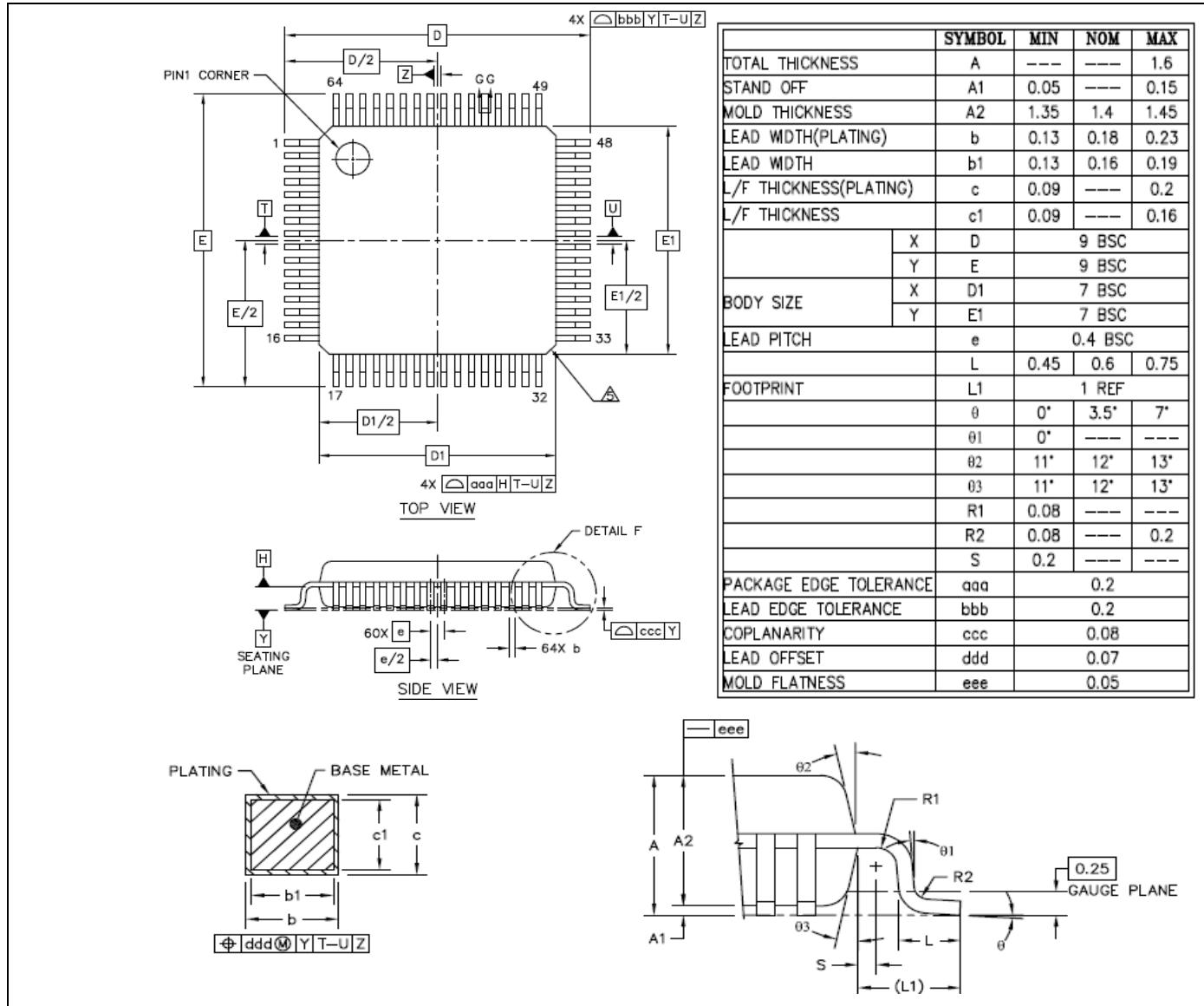


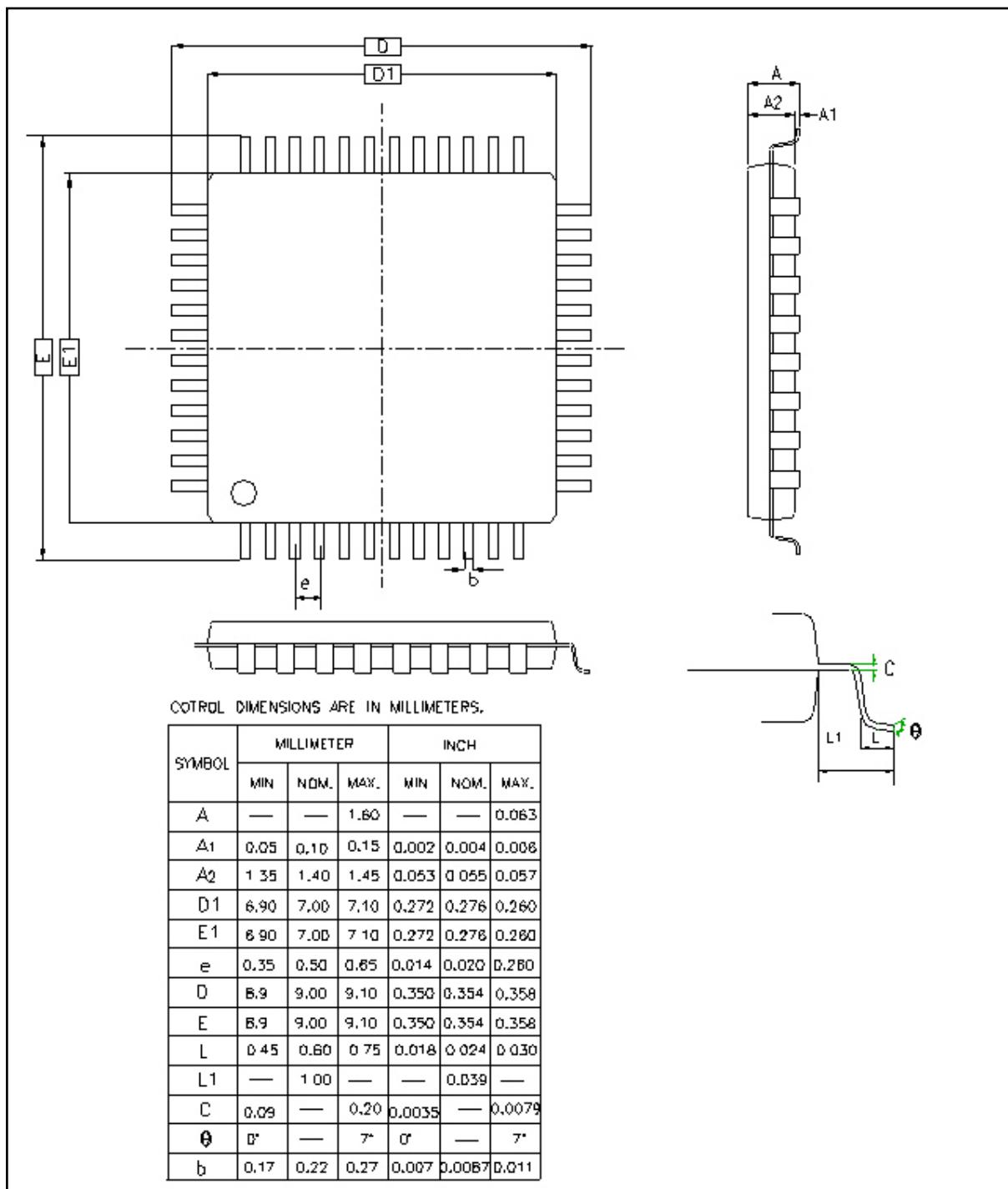
图 7-3 SPI 从机动态特性时序图

8 封装定义

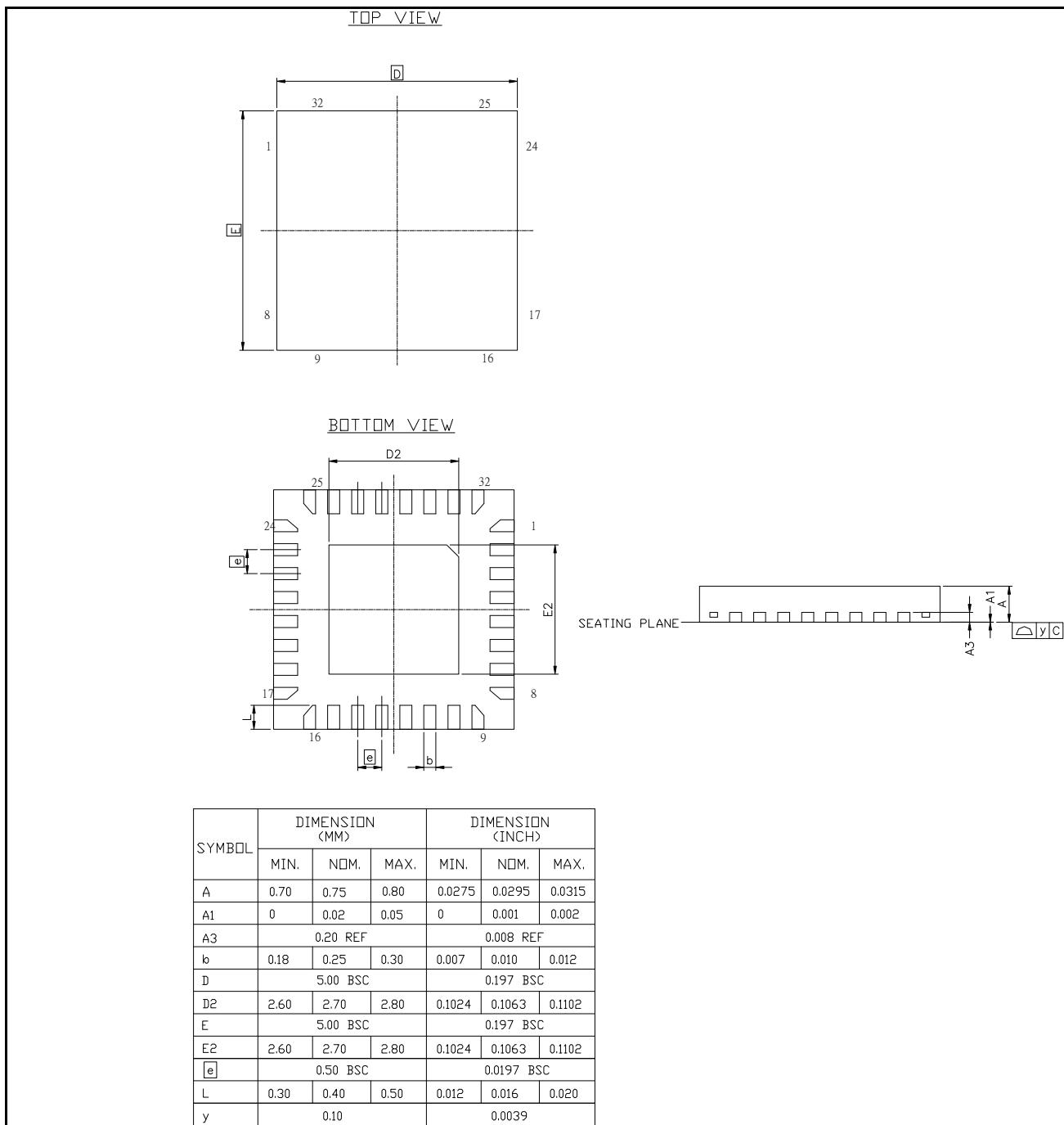
8.1 64L LQFP (7x7x1.4 mm footprint 2.0 mm)



8.2 48L LQFP (7x7x1.4 mm footprint 2.0 mm)



8.3 33L QFN (5x5x0.8 mm)



9 版本历史

版本	日期	页码/章节	描述
V1.00	2011-1-18	-	初次发行版本
V1.01	2011-2-11	第5章	1. 修正了寄存器CLKSEL1中TMRx_S 的描述。
V1.02	2011-3-14	第3章 第5章 第7章 第8章	增加了 LQFP 64-pin 对于 7x7x1.4mm 封装. (NUC122SD2AN, NUC122SC1AN) 2. 修正了 LQFP 64-pin Pin 框图. 3. 修正了Timer在CLKSEL1中的时钟源框图 4. 去掉了RCADJ控制寄存器. 5. 修正了SPI控制寄存器和功能描述 6. 更新了直流和交流电气特性. 7. 更新了 LQFN 48封装尺寸.
V1.03	2011-3-31	第2章 第3章 第4章 第5章 第8章	1. 去掉了 LQFP 64-pin 10x10x1.4mm 封装. 2.把“12 MHz”替换成“4~24 MHz” 3. 在TCSR[31] 和 WTCR[31]控制寄存器中，增加了 ICE 调试响应控制位
V1.04	2011-4-29	第1章 第2章 第3章 第5章 第6章 第7章	1. 修正了 GPIOxn_DOUT 的描述. (x=A~D, n=0~15) 2.更新了 RTC 框图 . 3. 更新了 LDO参数表和电源管理. 4. 去掉了UART中的LIN功能. 5. 改正了寄存器FPS2CLK中的“PS2DAT”为 “PS2CLK” . 6. Timer控制时钟源框图中，改5个选项为4个选项. 7. 修改了寄存器I2CSTATUS 的复位值. 8.在PWM章节概述中，改“PWM_CRLx/PWM_CFLx(x=0~3)” 为 “CRLRx/CFLRx(x=0~3)” 9. 修改了寄存器UA_LCR 中SPE, EPE, PBE 和 NSB 位的描述 10.改RIIR 和 TTR 为可读写. 11. 修改了 SPI的编程示例. 12. 在系统时钟和节拍时钟一章，框图中改 “1xx” 为 “111” 13.增加了寄存器 GPIOx_DMASK (x=A~D) 的备注 14. 增加了时钟发生器的整体框图. 15.增加了Timer的功能描述 16. 修改了寄存器 ICSR 和 SCR 的描述. 17.在引脚配置和描述里，把“RX0/1” 和 “TX0/1” 改为 “RXD0/1” 和 “TXD0/1” 18. 修改了寄存器UA_TOR中TOIC位 的描述. 19.修改了寄存器FATCON 的描述. 20. 修改了I2C操作模式的5个状态流程图
V1.05	2011-3-30	第3章 第5章	1. 修正了LQFP48封装中PIN17, PIN18的描述 2. 修正了PIIR的描述

V1.06	2011-6-8	第2章 第7章	1. 修正在特性表的“Clock Control”一栏，修改22.1184Mhz高速振荡器的校正条件 2. 修正内部22.1184MHz高速振荡器的规格说明
V1.07	2011-6-21	第2章 第5章	1, 在SPI特性表里面，修改了SPI通信速率。 2, 修改了CH0MOD, CH1MOD, CH2MOD, CH3MOD, TIF和CURRENT的描述。 3, 修改了SPI的功能描述，和GO_BUSY, SSR寄存器描述



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.*