

Ver 1.1

300MHz 抗辐照 SPARC CPU

产品使用手册

产品型号: BM3823AMCCRH



北京微电子技术研究所



版本控制页

版本号	发布日期	更改章节	更改说明	备注
1.0	2019.3.26			
1.1	2020.7.29	第 4 章 附录 9	1.增加 4.7.3 小节; 2.增加附录 9.2 和 9.3 小节; 3.全文勘误。	



目 录

目 录	3
一、产品概述	12
1.1 产品特点	12
1.2 产品用途及应用范围	14
1.3 惯例和术语表	14
1.4 免责声明	16
二、产品工作条件	17
2.1 绝对最大额定值	17
2.2 推荐工作条件	17
三、封装及引出端说明	18
3.1 引出端排列	18
3.2 引脚信号描述	18
3.3 外形尺寸说明	50
四、产品功能	51
4.1 产品结构框图及核心	51
4.1.1 整数单元	52
4.1.2 浮点处理单元	52
4.1.3 工作模式	53
4.1.4 产品复位	53
4.1.5 产品异常	53
4.1.6 容错设计	55
4.2 Cache	55
4.2.1 概述	55
4.2.2 Cache 缓存操作	55



4.2.3 Cache 容错功能.....	56
4.2.4 Cache Tag 信息说明.....	57
4.2.5 Cache 寄存器地址映射.....	57
4.3 MMU.....	58
4.3.1 概述.....	58
4.3.2 MMU 映射操作.....	59
4.3.3 MMU 寄存器地址映射.....	60
4.4 中断和陷阱.....	61
4.4.1 概述.....	61
4.4.2 陷阱类型.....	62
4.4.3 中断.....	65
4.5 DMA 控制器.....	69
4.5.1 概述.....	69
4.5.2 功能描述.....	70
4.5.3 DMA 传输配置流程.....	70
4.5.5 DMA 中断.....	76
4.5.6 软件握手控制 DMA 传输.....	77
4.5.7 DMA 寄存器地址映射.....	80
4.6 Ethernet MAC.....	82
4.6.1 概述.....	82
4.6.2 功能描述.....	82
4.6.3 关键接口引脚说明.....	83
4.6.4 系统连接.....	84
4.6.5 MAC 编程使用.....	85
4.6.6 描述符说明.....	86
4.6.7 寄存器地址映射.....	88
4.7 DDR2 存储器控制器.....	89
4.7.1 概述.....	89



4.7.2 功能描述.....	89
4.7.3 软硬件配置.....	90
4.7.4 时序.....	94
4.7.5 DDR2 控制器的寄存器地址映射.....	96
4.7.6 DDR2 PHY 的寄存器地址映射.....	101
4.8 PCI 总线控制器.....	104
4.8.1 概述.....	104
4.8.2 AHB-PCI 模块结构.....	105
4.8.3 PCI 接口复位.....	105
4.8.4 PCI 接口 AHB 侧地址空间.....	106
4.8.5 PCI 仲裁器.....	107
4.8.6 PCI 侧寄存器空间.....	108
4.8.7 地址空间映射.....	109
4.8.8 DMA.....	112
4.8.9 PCI 模块中断.....	115
4.9 内部存储器接口.....	117
4.9.1 概述.....	117
4.9.2 内部存储器地址空间分配.....	117
4.9.3 内部存储器寄存器地址映射.....	118
4.9.4 内部存储器 EDAC 进 trap 情况.....	118
4.10 外部存储器接口.....	118
4.10.1 概述.....	118
4.10.2 外部存储器控制器地址空间.....	119
4.10.3 RAM 接口.....	126
4.10.4 PROM 接口.....	136
4.10.5 I/O 空间接口.....	139
4.10.6 访问错误.....	142
4.10.7 错误管理.....	142



4.10.8 写保护	144
4.11 GPIO.....	145
4.11.1 概述.....	145
4.11.2 数据模式和 IO 模式.....	145
4.11.3 复用功能.....	147
4.11.4 GPIO 寄存器地址映射	148
4.12 SPI 总线控制器.....	148
4.12.1 概述.....	148
4.12.2 SPI 控制器的配置	149
4.12.3 SPI 接口时序描述	149
4.12.4 SPI 寄存器地址映射	151
4.13 I ² C 总线控制器	151
4.13.1 概述.....	151
4.13.2 模块功能.....	152
4.13.3 数据传输.....	153
4.13.4 I ² C 寄存器地址映射.....	155
4.14 UART.....	156
4.14.1 概述.....	156
4.14.2 UART1、UART2.....	156
4.14.3 UART3、UART4.....	158
4.14.4 UART 寄存器地址映射	161
4.15 定时器和看门狗	162
4.15.1 预分频器.....	162
4.15.2 定时器.....	162
4.15.3 看门狗.....	163
4.15.4 定时器和看门狗的寄存器地址映射	163
4.16 计数器.....	163
4.16.1 概述.....	163



4.16.2 计数器功能.....	163
4.16.3 计数器的寄存器地址映射	164
4.17 PWM 定时器.....	165
4.17.1 PWM 定时器功能	165
4.17.2 PWM 定时器输出	165
4.17.3 PWM 寄存器地址映射	166
4.18 调试支持单元	167
4.18.1 概述.....	167
4.18.2 调试通讯链接.....	174
4.18.3 启动调试.....	176
4.18.4 调试支持单元寄存器地址映射	176
4.19 时钟配置.....	176
4.19.1 概述.....	176
4.19.2 PLL 管脚说明及使用要求.....	177
4.19.3 PLL 频率计算.....	179
4.20 JTAG 接口	180
4.20.1 概述.....	180
4.20.2 JTAG 使用要求	180
五、产品电特性.....	182
六、典型应用.....	200
6.1 开发板说明	200
6.2 开发板参考电路	202
6.2.1 电源模块.....	202
6.2.2 时钟倍频电路及其工作模式	204
6.2.3 DSU 串口电路.....	206
6.2.4 FLASH	206
6.2.5 SRAM	207



6.2.7 SDRAM	208
6.2.8 DDR2 SDRAM	209
6.2.9 Ethernet MAC	210
七、应用注意事项	211
7.1 产品应用说明	211
7.2 对电源的要求	212
7.3 编程说明	213
7.3.1 BM3823A 汇编编程注意事项	213
7.3.2 BM3823A 软件示例	216
7.4 产品防护	222
7.4.1 电装及防护措施	222
7.4.2 包装	222
7.4.3 运输和贮存	222
八、用户关注产品信息	224
8.1 产品标识	224
8.2 研制生产单位联系方式	224
附录 1 警示页	225
附录 2 寄存器列表	226
2.1 0x80000000 地址上的寄存器	226
2.2 0x81000000 地址上的寄存器	227
附录 3 寄存器描述	230
3.1 系统相关寄存器	230
3.2 Cache 相关寄存器	238
3.3 MMU 相关寄存器	240
3.4 中断相关寄存器	243



3.5 DMA 相关寄存器.....	错误！未定义书签。
3.6 Ethernet MAC 相关寄存器	266
3.6.1 Ethernet DMA 相关寄存器	266
3.6.2 GMAC 相关寄存器	279
3.6.3 发送描述符相关寄存器	291
3.6.4 接收描述符相关寄存器	298
3.7 DDR2 相关寄存器.....	303
3.7.1 DDR2 控制器寄存器	303
3.7.2 DDR2 PHY 寄存器	312
3.7.3 OCD_PU/PD	324
3.7.4 TSEL/DS	325
3.8 PCI 相关寄存器.....	325
3.9 内存控制器相关寄存器	345
3.10 外存控制器相关寄存器	346
3.11 GPIO 相关寄存器.....	355
3.12 SPI 相关寄存器	361
3.13 I2C 相关寄存器.....	364
3.14 UART 相关寄存器	373
3.15 定时器相关寄存器	385
3.16 计数器相关寄存器	389
3.17 PWM 相关寄存器	391
3.18 DSU 相关寄存器	394
3.19 写保护相关寄存器	399
附录 4 寄存器初始化	401
4.1 寄存器初始化简介	401
4.2 推荐寄存器初始化过程	401
4.2.1 内核寄存器初始化	401



4.2.2 片上寄存器初始化	407
4.2.3 用户程序相关寄存器初始化	409
附录 5 软件开发规范	411
5.1 通用编程规范	411
5.1.1 寄存器窗口与函数调用	411
5.1.2 堆栈	416
5.1.3 陷阱和中断	417
5.1.4 Cache 刷新	417
5.2 引发 Trap0x1a 陷阱条件	418
5.3 引发 Trap0x09 陷阱条件	419
附录 6 SPARC V8 介绍	420
6.1 SPARC V8 概述	420
6.2 数据类型	420
6.3 寄存器	421
6.3.1 IU 寄存器	422
6.3.2 FPU 寄存器	422
6.3.3 寄存器推荐使用方法汇总	423
6.4 指令	424
6.4.1 指令格式	424
6.4.2 指令分类	424
6.5 陷阱	425
6.5.1 精确陷阱	425
6.5.2 中断陷阱	426
6.5.3 延迟陷阱	426
6.5.4 陷阱控制	426
6.5.5 陷阱识别	427



附录 7 PROM、SRAM、IO、SDRAM 访问时序.....	428
附录 8 与 BM3803FMGRH 差异性.....	434
附录 9 PCI、DDR2 及 MAC 不使用时相关管脚接法说明.....	436
9.1 PCI 不使用时管脚连接说明	436
9.2 DDR2 不使用时管脚连接说明	437
9.3 MAC 不使用时管脚连接说明	438

一、产品概述

BM3823AMCCRH 是一款 SPARC V8 体系结构的 32 位高性能抗辐射微处理器，集成高性能整数处理单元、浮点处理单元、独立的指令和数据 Cache、256KB 片上存储器、DDR2 存储器控制器、Ethernet MAC、PCI 总线控制器以及其它丰富的外设接口，与 PROM 等相关外围电路组成完整的单板计算机系统，满足空间恶劣环境应用要求，可广泛应用于高可靠的信息处理系统及测控系统。

1.1 产品特点

- ◆ 最大工作频率：300MHz
- ◆ 主要性能指标：0.85DMIPS/MHz（Dhrystone2.1），0.23MFLOPS/MHz（Whetstone）
- ◆ 兼容 SPARC V8 体系结构的 32 位微处理器内核：
 - ✓ 符合 IEEE754 标准的 32/64 位浮点处理单元（FPU）
 - ✓ 32KB 指令 Cache，16KB 数据 Cache
 - ✓ 支持系统级容错
- ◆ 内嵌硬件乘/除法器
- ◆ 外部存储器控制器
 - ✓ 支持 8/16/32 位宽 PROM、SRAM、I/O 空间
 - ✓ 支持 32 位宽 SDRAM
 - ✓ 支持 8/32 位宽 EDAC
- ◆ 256KB 片上存储器，支持 EDAC 功能
- ◆ PCI 总线控制器
 - ✓ 符合 PCI 2.3 协议规范
 - ✓ 支持主/从桥模式
 - ✓ 支持 DMA
- ◆ 内嵌 4 通道带 FIFO 的 DMA 控制器
- ◆ 中断控制器
 - ✓ 15 路一级中断，其中 7 路外部中断（1 路不可屏蔽，6 路可屏蔽）
 - ✓ 20 路二级中断

- ◆ DDR2 存储器控制器（含控制器和 PHY）
- ◆ Ethernet MAC
 - ✓ 符合 IEEE802.3 协议规范
 - ✓ 支持 10/100/1000Mbps 三种数据传输速率
 - ✓ 支持 MII/GMII、RMII、TBI 接口
- ◆ 2 路 SPI 总线控制器
- ◆ 2 路 I²C 总线控制器（主从模式）
- ◆ 4 路 UART，其中 2 路带 32 字节的接收/发送 FIFO
- ◆ 6 路 32 位计数器
- ◆ 6 路 24 位 PWM 定时器：带外部时钟输入和启停控制输入信号
- ◆ 2 路 32 位通用定时器
- ◆ 1 路看门狗
- ◆ 32 路 GPIO
- ◆ 硬件调试单元
 - ✓ 缓存跟踪功能
 - ✓ 4 个硬件观测点
- ◆ 容错设计
 - ✓ 三模冗余
 - ✓ 纠一检二的 EDAC 编解码
 - ✓ 奇偶校验
- ◆ 工作电压：
 - ✓ I/O 电源电压：3.3V $\pm$ 0.33V、1.8V $\pm$ 0.18V（DDR2）
 - ✓ 内核电源电压：1.2V $\pm$ 0.12V
- ◆ 工作温度：-55℃ $\sim$ +125℃
- ◆ 存储温度：-65℃ $\sim$ +150℃
- ◆ 功耗：小于 4W
- ◆ 抗辐射指标：
 - ✓ 抗总剂量： $\geq 1 \times 10^3$ Gy（Si）

- ✓ 单粒子锁定（SEL）阈值： $\geq 75\text{MeV cm}^2/\text{mg}$
- ✓ 单粒子功能在轨错误率（GEO 轨道）： $\leq 5 \times 10^{-5}$ 次/天 器件
- ◆ ESD 抗静电能力：2000V
- ◆ 质量等级：GJB 597B B 级/CAST C 级
- ◆ 封装形式：CCGA717
- ◆ 封装尺寸：35mm × 35mm
- ◆ 重量：16.6g~20.6g
- ◆ IEEE 1149.1 JTAG 接口
- ◆ 软件开发环境：
 - ✓ SPARC 集成开发环境（SPE-C）
 - ✓ SPARC 软件开发环境（SparcSDE）

1.2 产品用途及应用范围

BM3823AMCCRH 是基于 SPARC V8 体系结构的 32 位高性能抗辐射处理器，集成高性能整数处理单元、浮点处理单元、独立的指令和数据 Cache、硬件乘/除法器、256KB 片上存储器、DDR2 存储器控制器、高速 Ethernet MAC、DMA 控制器、PCI 总线控制器、中断控制器及 SPI 等其它丰富的外设接口，与 PROM 等相关外围电路组成完整的单板计算机系统，满足空间恶劣环境应用要求，可广泛应用于高可靠的信息处理系统及测控系统。

1.3 惯例和术语表

本节定义了本手册使用的惯例和术语表。如表 1-3-1 所示。

表1-3-1 惯例和术语表

惯例/术语	描述
ASI	Address Space Identifier, 地址空间标志符
ASR	Ancillary State Register, 辅助状态寄存器
Cache	高速缓冲存储器
CPU	Central Processing Unit, 中心处理单元
DSU	Debug Support Unit, 调试支持单元
Flash	闪存
FPU	Floating Point Unit, 浮点单元

惯例/术语	描述
FSR	Floating Point State Register, 浮点状态寄存器
GPIO	General Purpose Input Output, 通用输入输出
HSC	High Speed Counter, 高速计数单元
IU	Integer Unit, 整数单元
JTAG	Joint Test Action Group, 联合测试工作组
LMR	Load Mode Register, 加载模式寄存器
LRU	Least Recently Used, 最近最少使用
LSB	Least Significant Bit, 最低有效位
MSB	Most Significant Bit, 最高有效位
PC/nPC	Program Counter/next Program Counter, 程序计数器
PCI	Peripheral Component Interconnect, 外设部件互连
PLL	Phase Locked Loop, 锁相环
PROM	Programmable Read-Only Memory, 可编程只读存储器
PSR	Processor State Register, 处理器状态寄存器
PTO	Pulse Train Output, 高速脉冲串输出
PWM	Pulse Width Modulation, 脉冲宽度调制
RISC	Reduced Instruction Set Computer, 精简指令集处理器
SDRAM	Synchronous Dynamic Random Access Memory, 同步动态随机存储器
SoC	System On Chip, 片上单元
SPARC	Scalable Processor ARChitecture, 可扩充处理器架构
SPI	Serial Peripheral Interface, 串行外设接口
SRAM	Static Random Access Memory, 静态随机存取存储器
TBR	Trap Base Register, 异常基址寄存器
UART	Universal Asynchronous Receiver and Transmitter, 通用异步收发器
WIM	Windows Invalid Mask Register, 窗口无效寄存器
MAC	Media Access Controller, 媒体访问控制器

⚠ 注意

- 1、定义“惯例/术语”之后，本文其他地方使用“惯例/术语”时均统一引用方式，例如按照本表格，表达“浮点状态寄存器”时，均使用“FSR”。
- 2、寄存器复位值按照十六进制统一写法“0xXXX0XXXX”，“0x”表示16进制，每



4bit 中有 1bit 不定值，即写为“X”，4bit 全部为固定值则写为相应 16 进制数值。

3、本文中的寄存器名字的表述或引用使用加粗的表示方式。

4、本文中对寄存器中的字段的表述或引用使用斜体的表示方式。

1.4 免责声明

本手册版权归北京微电子技术研究所所有，并保留一切权利。未经书面许可，任何单位、组织和个人不得将此文档中的任何部分公开、转载或以其他方式散发给第三方，否则将追究其法律责任。

本手册版本将不定期更新，请在使用本产品之前联系本单位销售部门获取本手册的最新版本。

用户因未严格按本手册要求保存、使用本产品，致使产品工作异常或损坏，造成任何直接或间接损失，本单位不承担任何责任。

除本手册说明之外，请勿接受第三方指导或参考第三方资料对本产品进行操作，用户对本手册有疑问之处请与本单位销售部门联系。

二、产品工作条件

2.1 绝对最大额定值

绝对最大额定值如下：

- ◆ 1.2V 内核电源电压范围：-0.3V~+1.8V；
- ◆ 1.8V I/O 电源电压范围：-0.3V~+2.5V；
- ◆ 3.3V I/O 电源电压范围：-0.3V~+4.0V；
- ◆ 存储温度范围：-65℃~+150℃。

2.2 推荐工作条件

推荐工作条件如下：

- ◆ 1.2V 内核电源电压范围：1.08V~1.32V；
- ◆ 1.8V I/O 电源电压范围：1.62V~1.98V；
- ◆ 3.3V I/O 电源电压范围：2.97V~3.63V；
- ◆ 工作温度范围：-55℃~+125℃。

三、封装及引出端说明

3.1 引出端排列

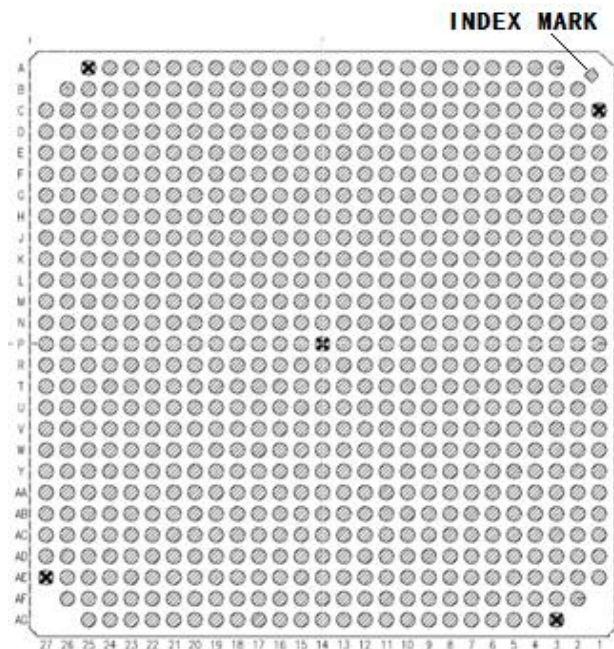


图3-1-1 引出端排列图（底视图）

3.2 引脚信号描述

表3-2-1 引脚信号描述

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
1	clk	AG12	I		系统时钟
2	resetn	C22	I		系统复位，低电平有效，需在时钟有效后至少保持 10 个时钟周期
3	errorn	G19	O(OD)		处理器系统错误指示，低电平输出有效
4	timerup[5]	F16	I	D	PWM 定时器 6 外部实启控制输入，高电平有效，配置为不使用外部实启功能时接 1k Ω 电阻下拉
5	timerup[4]	B16	I	D	PWM 定时器 5 外部实启控制输入，高电平有效，配置为不使用外部实启功能时接 1k Ω 电阻下拉
6	timerup[3]	E16	I	D	PWM 定时器 4 外部实启控制输入，高电平有效，配置为不使用外部实启功能时接 1k Ω 电阻下拉

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
7	timerup[2]	B19	I	D	PWM 定时器 3 外部实启控制输入, 高电平有效, 配置为不使用外部实启功能时接 1k Ω 电阻下拉
8	timerup[1]	G16	I	D	PWM 定时器 2 外部实启控制输入, 高电平有效, 配置为不使用外部实启功能时接 1k Ω 电阻下拉
9	timerup[0]	A18	I	D	PWM 定时器 1 外部实启控制输入, 高电平有效, 配置为不使用外部实启功能时接 1k Ω 电阻下拉
10	clke[5]	E17	I		PWM 定时器 6 外部时钟输入管脚, 不使用时接 4.7k Ω 电阻下拉
11	clke[4]	C19	I		PWM 定时器 5 外部时钟输入管脚, 不使用时接 4.7k Ω 电阻下拉
12	clke[3]	F17	I		PWM 定时器 4 外部时钟输入管脚, 不使用时接 4.7k Ω 电阻下拉
13	clke[2]	B17	I		PWM 定时器 3 外部时钟输入管脚, 不使用时接 4.7k Ω 电阻下拉
14	clke[1]	E18	I		PWM 定时器 2 外部时钟输入管脚, 不使用时接 4.7k Ω 电阻下拉
15	clke[0]	C20	I		PWM 定时器 1 外部时钟输入管脚, 不使用时接 4.7k Ω 电阻下拉
16	pwmtimer0	F18	O		PWM 定时器 1 输出
17	pwmtimer1	B18	O		PWM 定时器 2 输出
18	pwmtimer2	D19	O		PWM 定时器 3 输出
19	pwmtimer3	B21	O		PWM 定时器 4 输出
20	pwmtimer4	G17	O		PWM 定时器 5 输出
21	pwmtimer5	A19	O		PWM 定时器 6 输出
22	cnt1	F19	I	D	计数器 1 输入, 不使用时接 1k Ω 电阻下拉
23	cnt2	A20	I	D	计数器 2 输入, 不使用时接 1k Ω 电阻下拉
24	cnt3	D20	I	D	计数器 3 输入, 不使用时接 1k Ω 电阻下拉
25	cnt4	C21	I	D	计数器 4 输入, 不使用时接 1k Ω 电阻下拉
26	cnt5	F20	I	D	计数器 5 输入, 不使用时接 1k Ω 电阻下拉
27	cnt6	A21	I	D	计数器 6 输入, 不使用时接 1k Ω 电阻下拉

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
28	tck	D21	I	D	JTAG 时钟，不使用时接 1kΩ 电阻下拉
29	tdi	B20	I	U	JTAG 数据输入，不使用时需要接 1kΩ 电阻上拉
30	tdo	E20	O(三态)		JTAG 数据输出
31	tms	G18	I	U	JTAG 模式选择，不使用时需要接 1kΩ 电阻上拉
32	trst	B22	I	U	JTAG 复位，低电平有效，不使用时需要接 1kΩ 电阻下拉
33	pio[0]	E7	I/O		通用 I/O，与 PROM 宽度定义复用，上电复位时为输入，从外部 PROM 启动时，与 pio[1]配合使用，用于配置 PROM 数据位宽。”1x”表示 32 位宽，”01”表示 16 位宽，”00”表示 8 位宽。上电启动需要按照实际需求进行配置。处理器启动后该管脚作为正常 I/O 使用
34	pio[1]	B7	I/O		通用 I/O，与 PROM 宽度定义复用，上电复位时为输入，从外部 PROM 启动时，与 pio[0]配合使用，用于配置 PROM 数据位宽。”1x”表示 32 位宽，”01”表示 16 位宽，”00”表示 8 位宽。上电启动需要按照实际需求进行配置。处理器启动后该管脚作为正常 I/O 使用
35	pio[2]	G9	I/O		通用 I/O，与 PROM EDAC 使能定义复用，上电复位为输入，用于配置 MECFG1 的 romEEEn 位，为”1”时表示 PROM EDAC 使能。上电启动后该管脚作为正常 I/O 使用
36	pio[3]	D5	I/O		通用 I/O，与串口 1/2 外部可选时钟信号复用
37	pio[4]	D7	I/O		通用 I/O
38	pio[5]	B8	I/O		通用 I/O
39	pio[6]	G8	I/O		通用 I/O
40	pio[7]	C5	I/O		通用 I/O
41	pio[8]	E8	I/O		通用 I/O，与 CTSN2 管脚复用，详见手册 UART2 部分说明
42	pio[9]	A8	I/O		通用 I/O，与 RTSN2 管脚复用，详见手册 UART2 部分说明

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
43	pio[10]	F8	I/O		通用 I/O，与 RXD2 管脚复用，详见手册 UART2 部分说明
44	pio[11]	B4	I/O		通用 I/O，与 TXD2 管脚复用，详见手册 UART2 部分说明
45	pio[12]	D8	I/O		通用 I/O，与 CTSN1 管脚复用，详见手册 UART1 部分说明
46	pio[13]	B9	I/O		通用 I/O，与 RTSN1 管脚复用，详见手册 UART1 部分说明
47	pio[14]	F9	I/O		通用 I/O，与 RXD1 管脚复用，详见手册 UART1 部分说明
48	pio[15]	B5	I/O		通用 I/O，与 TXD1 管脚复用，详见手册 UART1 部分说明
49	pio[16]	E9	I/O		通用 I/O
50	pio[17]	A9	I/O		通用 I/O
51	pio[18]	F10	I/O		通用 I/O
52	pio[19]	C6	I/O		通用 I/O
53	pio[20]	D9	I/O		通用 I/O
54	pio[21]	B10	I/O		通用 I/O
55	pio[22]	G11	I/O		通用 I/O
56	pio[23]	C7	I/O		通用 I/O
57	pio[24]	E10	I/O		通用 I/O
58	pio[25]	A10	I/O		通用 I/O
59	pio[26]	F11	I/O		通用 I/O
60	pio[27]	C8	I/O		通用 I/O
61	pio[28]	D10	I/O		通用 I/O
62	pio[29]	B11	I/O		通用 I/O
63	pio[30]	G12	I/O		通用 I/O
64	pio[31]	C9	I/O		通用 I/O
65	gmac_clk_pmarx0	C15	I		以太网 TBI 接口 PHY 给 MAC 的奇时钟 62.5MHz，不使用时接 4.7kΩ 电阻下拉

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
66	gmac_clk_pmarx1	D15	I		以太网 TBI 接口 PHY 给 MAC 的偶时钟 62.5MHz, 不使用时接 4.7kΩ 电阻下拉
67	gmac_clk_rmii	A15	O		以太网 RMII 接口 MAC 对 PHY 输出时钟 50MHz
68	gmac_clk_tx_125	A17	O		以太网 TBI 接口 MAC 给 PHY 的 125MHz 发送 时钟
69	gmac_clk_tx	E15	O		以太网 GMII/MII 接口 MAC 给 PHY 的发送时钟
70	gmac_intf_sel[0]	A16	I	D	以太网 PHY 接口模式选择, 不使用时接 1kΩ 电 阻下拉
71	gmac_intf_sel[1]	D18	I	D	以太网 PHY 接口模式选择, 不使用时接 1kΩ 电 阻下拉
72	gmac_intf_sel[2]	C18	I	D	以太网 PHY 接口模式选择, 不使用时接 1kΩ 电 阻下拉
73	gmac_MDC	C16	O		以太网管理数据时钟信号, 用于控制器操作管理 PHY 时数据传输时钟同步
74	gmac_MDIO	F15	I/O		以太网管理数据信号, 用于控制器操作管理 PHY 数据传输, 不使用时接 4.7kΩ 电阻下拉
75	gmac_pcs_aquired_sync_o	C14	O		以太网 TBI 非标准 PHY 接口, 物理编码子层 (PCS) 已获得同步的标识, 高电平表示 PCS 接 口已同步到接收码组运行差异 (用户无需关心)
76	gmac_pcs_en_cdet_o	D14	O		以太网 TBI 非标准 PHY 接口, PCS 检测使能信 号, 复位后为低。高电平表示使能 PCS 检测功能
77	gmac_pcs_ewrap_o	B15	O		以太网 TBI 非标准 PHY 接口, 物理编码子层回 卷使能, 高电平使能 PHY 回环, 此时 PHY 将发 送序列数据回送接收路径
78	gmac_pcs_lck_ref_o	H14	O		以太网 TBI 非标准 PHY 接口, PCS 锁定参考信 号, 该信号反映了 PCS 控制寄存器的 LR 位, 复 位后为低
79	gmac_phy_clk_rx	D17	I		以太网 GMII/MII 外部 PHY 给 MAC 的接收时钟 (125/25/2.5MHz), 不使用时接 4.7kΩ 电阻下拉
80	gmac_phy_clk_tx	C17	I		以太网 GMII/MII 外部 PHY 给 MAC 的发送时钟 (25MHz/2.5MHz), 不使用时接 4.7kΩ 电阻下拉

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
81	gmac_phy_col_i	E11	I		以太网 PHY 发送/接收通道忙指示信号，在 GMII/MII 模式时有效，由 PHY 提供，高电平有效，不使用时接 4.7kΩ 电阻下拉
82	gmac_phy_crs_i	A11	I		以太网 PHY 发送/接收通道冲突指示信号，在 GMII/MII 模式有效，由 PHY 提供，高电平有效，不使用时接 4.7kΩ 电阻下拉
83	gmac_phy_rxd_i[0]	F12	I		以太网 PHY 接口接收数据（低位数据），不使用时接 4.7kΩ 电阻下拉
84	gmac_phy_rxd_i[1]	C10	I		以太网 PHY 接口接收数据，不使用时接 4.7kΩ 电阻下拉
85	gmac_phy_rxd_i[2]	D11	I		以太网 PHY 接口接收数据，不使用时接 4.7kΩ 电阻下拉
86	gmac_phy_rxd_i[3]	B12	I		以太网 PHY 接口接收数据，不使用时接 4.7kΩ 电阻下拉
87	gmac_phy_rxd_i[4]	G13	I		以太网 PHY 接口接收数据，不使用时接 4.7kΩ 电阻下拉
88	gmac_phy_rxd_i[5]	C11	I		以太网 PHY 接口接收数据，不使用时接 4.7kΩ 电阻下拉
89	gmac_phy_rxd_i[6]	E12	I		以太网 PHY 接口接收数据，不使用时接 4.7kΩ 电阻下拉
90	gmac_phy_rxd_i[7]	A12	I		以太网 PHY 接口接收数据（高位数据），不使用时接 4.7kΩ 电阻下拉
91	gmac_phy_rxdv_i	F13	I		以太网 PHY 接收数据有效指示信号，高电平有效，由 PHY 提供，不使用时接 4.7kΩ 电阻下拉
92	gmac_phy_rxer_i	C12	I		以太网 PHY 接收数据错误指示信号，高电平有效，由 PHY 提供，不使用时接 4.7kΩ 电阻下拉
93	gmac_phy_txd_o[0]	D12	O		以太网 PHY 接口发送数据（低位数据）
94	gmac_phy_txd_o[1]	B13	O		以太网 PHY 接口发送数据
95	gmac_phy_txd_o[2]	E14	O		以太网 PHY 接口发送数据
96	gmac_phy_txd_o[3]	C13	O		以太网 PHY 接口发送数据
97	gmac_phy_txd_o[4]	E13	O		以太网 PHY 接口发送数据

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
98	gmac_phy_txd_o[5]	A13	O		以太网 PHY 接口发送数据
99	gmac_phy_txd_o[6]	F14	O		以太网 PHY 接口发送数据
100	gmac_phy_txd_o[7]	B14	O		以太网 PHY 接口发送数据（高位数据）
101	gmac_phy_txen_o	D13	O		以太网 PHY 发送数据使能，高电平有效
102	gmac_phy_txer_o	A14	O		以太网 PHY 发送数据错误，高电平有效
103	gmac_ptp_aux_ts_trig_i	D16	I		以太网辅助映射记录事件的触发信号，上升沿触发，不使用时接 4.7kΩ 电阻下拉
104	gmac_tbi_sigdet_i	G14	I		以太网 TBI 信号检测信号，高电平有效，不使用时接 4.7kΩ 电阻下拉
105	irqin[0]	C23	I	D	外部中断输入 0，不使用时接 1kΩ 电阻下拉
106	irqin[1]	E21	I	D	外部中断输入 1，不使用时接 1kΩ 电阻下拉
107	irqin[2]	A23	I	D	外部中断输入 2，不使用时接 1kΩ 电阻下拉
108	irqin[3]	E22	I	D	外部中断输入 3，不使用时接 1kΩ 电阻下拉
109	irqin[4]	B23	I	D	外部中断输入 4，不使用时接 1kΩ 电阻下拉
110	irqin[5]	D22	I	D	外部中断输入 5，不使用时接 1kΩ 电阻下拉
111	nmi	A22	I	D	非屏蔽中断，不使用时接 1kΩ 电阻下拉
112	spi_clk0	D23	O		第 0 路 SPI 的输出时钟信号
113	spi_clk1	C24	O		第 1 路 SPI 的输出时钟信号
114	spi_csn0_0	F21	O		第 0 路 SPI 通过片选外部可接两个外设，SPI0 的第 0 个片选信号，低电平有效
115	spi_csn0_1	A24	O		第 0 路 SPI 通过片选外部可接两个外设，SPI0 的第 1 个片选信号，低电平有效
116	spi_csn1_0	A25	O		第 1 路 SPI 通过片选外部可接两个外设，SPI1 的第 0 个片选信号，低电平有效
117	spi_csn1_1	D24	O		第 1 路 SPI 通过片选外部可接两个外设，SPI1 的第 1 个片选信号，低电平有效
118	spi_miso0	B24	I		两线模式时，表示 SPI0 的数据输入；单线模式时无效
119	spi_miso1	G21	I		两线模式时，表示 SPI1 的数据输入；单线模式时无效

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
120	spi_mosi0	G20	I/O		两线模式时，表示 SPI0 的数据输出；单线模式时为双向数据
121	spi_mosi1	B25	I/O		两线模式时，表示 SPI1 的数据输出；单线模式时为双向数据
122	i2c_hsmode0	L21	O		第 0 路 I ² C (I ² C0) 模式指示信号，高电平为高速模式，低电平表示标准/快速模式
123	i2c_hsmode1	F23	O		第 1 路 I ² C (I ² C1) 模式指示标识，高电平为高速模式，低电平表示标准/快速模式
124	i2c_mcs0	E25	O		第 0 路 I ² C (I ² C0) 在高速模式下的电流源使能指示信号，任何时候在高速模式，只有 1 个主机的电流源有效，高电平有效。（该信号仅输出作为指示信号，不连接 I ² C 外部设备，HS 模式时从第一个字节数据传输开始变高，数据传输结束后变低）
125	i2c_mcs1	H21	O		第 1 路 I ² C (I ² C1) 在高速模式下的电流源使能指示信号，任何时候在高速模式，只有 1 个主机的电流源有效，高电平有效。（该信号仅输出作为指示信号，不连接 I ² C 外部设备，HS 模式从第一字节数据传输开始变高，传输结束后变低）
126	i2c_scl1	D26	I/O(OD)		I ² C1 总线时钟信号，空闲时保持高电平，板级设计时接 4.7K Ω -10k Ω 电阻上拉
127	i2c_sda1	G22	I/O(OD)		I ² C1 总线数据信号，空闲时保持高电平，板级设计时接 4.7K Ω -10k Ω 电阻上拉
128	i2c_scl0	F24	I/O(OD)		I ² C0 总线时钟信号，空闲时保持高电平，板级设计时接 4.7K Ω -10k Ω 电阻上拉
129	i2c_sda0	L20	I/O(OD)		I ² C0 总线数据信号，空闲时保持高电平，板级设计时接 4.7K Ω -10k Ω 电阻上拉
130	timer0	K21	O		定时器 1 中断输出，高电平有效
131	timer1	E24	O		定时器 2 中断输出，高电平有效
132	wdogn	J21	O(OD)		看门狗输出，低电平有效
133	rx3	D25	I	U	串口 3 的接收数据信号

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
134	rx4	H20	I	U	串口 4 的接收数据信号
135	tx3	F22	O		串口 3 的发送数据信号
136	tx4	K20	O		串口 4 的发送数据信号
137	address[0]	AA3	O		地址总线第 0 位（最低位）
138	address[1]	U7	O		地址总线第 1 位
139	address[2]	Y2	O		地址总线第 2 位
140	address[3]	V8	O		地址总线第 3 位
141	address[4]	AA4	O		地址总线第 4 位
142	address[5]	W6	O		地址总线第 5 位
143	address[6]	AA1	O		地址总线第 6 位
144	address[7]	W8	O		地址总线第 7 位
145	address[8]	AB3	O		地址总线第 8 位
146	address[9]	V7	O		地址总线第 9 位
147	address[10]	AA2	O		地址总线第 10 位
148	address[11]	Y5	O		地址总线第 11 位
149	address[12]	AB4	O		地址总线第 12 位
150	address[13]	Y6	O		地址总线第 13 位
151	address[14]	AB1	O		地址总线第 14 位
152	address[15]	AA5	O		地址总线第 15 位
153	address[16]	AD1	O		地址总线第 16 位
154	address[17]	Y7	O		地址总线第 17 位
155	address[18]	AB2	O		地址总线第 18 位
156	address[19]	AE1	O		地址总线第 19 位
157	address[20]	AD2	O		地址总线第 20 位
158	address[21]	W7	O		地址总线第 21 位
159	address[22]	AC1	O		地址总线第 22 位
160	address[23]	AE2	O		地址总线第 23 位
161	address[24]	AC3	O		地址总线第 24 位
162	address[25]	Y8	O		地址总线第 25 位
163	address[26]	AC2	O		地址总线第 26 位

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
164	address[27]	AC4	O		地址总线第 27 位（最高位）
165	bexcn	P5	I	U	总线异常，低电平有效，用于 PROM、SRAM 或 I/O 空间，不使用时需要接 1K Ω 电阻上拉
166	brdyn	R1	I	U	外部数据准备好信号，低电平有效，用于 SRAM BANK 4 和 I/O 空间，不使用时接 1K Ω 电阻上拉
167	data[0]	U3	I/O		数据总线第 0 位（最低位）
168	data[1]	R8	I/O		数据总线第 1 位
169	data[2]	T2	I/O		数据总线第 2 位
170	data[3]	U5	I/O		数据总线第 3 位
171	data[4]	U4	I/O		数据总线第 4 位
172	data[5]	U6	I/O		数据总线第 5 位
173	data[6]	U1	I/O		数据总线第 6 位
174	data[7]	R9	I/O		数据总线第 7 位
175	data[8]	V3	I/O		数据总线第 8 位
176	data[9]	T7	I/O		数据总线第 9 位
177	data[10]	U2	I/O		数据总线第 10 位
178	data[11]	T9	I/O		数据总线第 11 位
179	data[12]	V4	I/O		数据总线第 12 位
180	data[13]	T8	I/O		数据总线第 13 位
181	data[14]	V1	I/O		数据总线第 14 位
182	data[15]	Y9	I/O		数据总线第 15 位
183	data[16]	W3	I/O		数据总线第 16 位
184	data[17]	V5	I/O		数据总线第 17 位
185	data[18]	V2	I/O		数据总线第 18 位
186	data[19]	W11	I/O		数据总线第 19 位
187	data[20]	W4	I/O		数据总线第 20 位
188	data[21]	W5	I/O		数据总线第 21 位
189	data[22]	W1	I/O		数据总线第 22 位
190	data[23]	W10	I/O		数据总线第 23 位
191	data[24]	Y3	I/O		数据总线第 24 位

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
192	data[25]	V6	I/O		数据总线第 25 位
193	data[26]	W2	I/O		数据总线第 26 位
194	data[27]	W9	I/O		数据总线第 27 位
195	data[28]	Y4	I/O		数据总线第 28 位
196	data[29]	U8	I/O		数据总线第 29 位
197	data[30]	Y1	I/O		数据总线第 30 位
198	data[31]	V9	I/O		数据总线第 31 位（最高位）
199	ios[0]	N4	O		IO 片选 0，高电平有效
200	ios[1]	P6	O		IO 片选 1，高电平有效
201	ios[2]	P1	O		IO 片选 2，高电平有效
202	ios[3]	N7	O		IO 片选 3，高电平有效
203	ios[4]	P4	O		IO 片选 4，高电平有效
204	ios[5]	P7	O		IO 片选 5，高电平有效
205	iosn[0]	P2	O		IO 片选 0，与 ios[0]反相，低电平有效
206	iosn[1]	N6	O		IO 片选 1，与 ios[1]反相，低电平有效
207	iosn[2]	R3	O		IO 片选 2，与 ios[2]反相，低电平有效
208	iosn[3]	R6	O		IO 片选 3，与 ios[3]反相，低电平有效
209	iosn[4]	P3	O		IO 片选 4，与 ios[4]反相，低电平有效
210	iosn[5]	N5	O		IO 片选 5，与 ios[5]反相，低电平有效
211	oe	AF2	O		ROM 或 IO 读使能，高电平有效
212	oen	AA6	O		ROM 或 IO 读使能，与 oe 反相，低电平有效
213	pardata[0]	T3	I/O		外存校验数据第 0 位（低位数据）
214	pardata[1]	T6	I/O		外存校验数据第 1 位
215	pardata[2]	R2	I/O		外存校验数据第 2 位
216	pardata[3]	R5	I/O		外存校验数据第 3 位
217	pardata[4]	T4	I/O		外存校验数据第 4 位
218	pardata[5]	R7	I/O		外存校验数据第 5 位
219	pardata[6]	T1	I/O		外存校验数据第 6 位
220	pardata[7]	T5	I/O		外存校验数据第 7 位（高位数据）
221	ramoen[0]	AG5	O		SRAM bank0 输出使能，低电平有效

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
222	ramoen[1]	AB5	O		SRAM bank1 输出使能，低电平有效
223	ramoen[2]	AE5	O		SRAM bank2 输出使能，低电平有效
224	ramoen[3]	AE6	O		SRAM bank3 输出使能，低电平有效
225	ramoen[4]	AG4	O		SRAM bank4 输出使能，低电平有效
226	ramsn[0]	AA8	O		SRAM bank0 片选信号，低电平有效
227	ramsn[1]	AE4	O		SRAM bank1 片选信号，低电平有效
228	ramsn[2]	AD6	O		SRAM bank2 片选信号，低电平有效
229	ramsn[3]	AG3	O		SRAM bank3 片选信号，低电平有效
230	ramsn[4]	AA9	O		SRAM bank4 片选信号，低电平有效
231	rbn[0]	AG7	O		SRAM 读操作和写操作的字节选通，低电平有效，对应数据管脚 data[31:24]
232	rbn[1]	AB6	O		SRAM 读操作和写操作的字节选通，低电平有效，对应数据管脚 data[23:16]
233	rbn[2]	AF6	O		SRAM 读操作和写操作的字节选通，低电平有效，对应数据管脚 data[15:8]
234	rbn[3]	AD8	O		SRAM 读操作和写操作的字节选通，低电平有效，对应数据管脚 data[7:0]
235	romsn[0]	R4	O		ROM bank0 片选，低电平有效
236	romsn[1]	P8	O		ROM bank1 片选，低电平有效
237	rwcn[0]	AG6	O		SRAM 字节控制的写使能，低电平有效，对应数据管脚 data[31:24]
238	rwcn[1]	AA10	O		SRAM 字节控制的写使能，低电平有效，对应数据管脚 data[23:16]
239	rwcn[2]	AF5	O		SRAM 字节控制的写使能，低电平有效，对应数据管脚 data[15:8]
240	rwcn[3]	AE7	O		SRAM 字节控制的写使能，低电平有效，对应数据管脚 data[7:0]
241	spw_clk	AF4	O		SpaceWire 时钟。
242	spw_hack	AD5	I		SpaceWire 应答，高电平表示数据准备好，不使用时接 4.7KΩ 电阻下拉

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
243	write_1553	AF3	O		1553 接口读写使能，高电平表示写有效
244	write	AE3	O		ROM 或 I/O 空间或 SRAM 写使能，高电平有效
245	writen_1553	AA7	O		1553 接口读写使能，与 write_1553 反相，低电平表示写有效
246	writen	AD4	O		ROM 或 I/O 空间或 SRAM 写使能，与 write 反相，低电平有效
247	sdcasn	AG8	O		SDRAM 列地址选择，低电平有效
248	sdclk	AG11	O		SDRAM 时钟
249	sdcsn[0]	AF7	O		SDRAM bank0 片选，低电平有效
250	sdcsn[1]	AE8	O		SDRAM bank1 片选，低电平有效
251	sddqm[0]	AB10	O		SDRAM 字节选通，低电平选通有效，对应数据管脚 data[7:0]
252	sddqm[1]	AF11	O		SDRAM 字节选通，低电平选通有效，对应数据管脚 data[15:8]
253	sddqm[2]	AD10	O		SDRAM 字节选通，低电平选通有效，对应数据管脚 data[23:16]
254	sddqm[3]	AB9	O		SDRAM 字节选通，低电平选通有效，对应数据管脚 data[31:24]
255	sdrasn	AB7	O		SDRAM 行地址选择，低电平有效
256	sdwen	AG10	O		SDRAM 写使能，低电平有效
257	pad_mem_address[0]	AE22	O		DDR2 行列地址第 0 位（低位地址）
258	pad_mem_address[1]	AE23	O		DDR2 行列地址第 1 位
259	pad_mem_address[2]	AF20	O		DDR2 行列地址第 2 位
260	pad_mem_address[3]	AD23	O		DDR2 行列地址第 3 位
261	pad_mem_address[4]	AC21	O		DDR2 行列地址第 4 位
262	pad_mem_address[5]	AE24	O		DDR2 行列地址第 5 位
263	pad_mem_address[6]	AF21	O		DDR2 行列地址第 6 位
264	pad_mem_address[7]	AC24	O		DDR2 行列地址第 7 位
265	pad_mem_address[8]	AD22	O		DDR2 行列地址第 8 位
266	pad_mem_address[9]	AC25	O		DDR2 行列地址第 9 位

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
267	pad_mem_address[10]	AF23	O		DDR2 行列地址第 10 位
268	pad_mem_address[11]	AB24	O		DDR2 行列地址第 11 位
269	pad_mem_address[12]	AC22	O		DDR2 行列地址第 12 位
270	pad_mem_address[13]	AB25	O		DDR2 行列地址第 13 位
271	pad_mem_address[14]	AF25	O		DDR2 行列地址第 14 位（高位地址）
272	pad_mem_bank[0]	Y24	O		DDR2 BANK 地址第 0 位（低位）
273	pad_mem_bank[1]	AC26	O		DDR2 BANK 地址第 1 位
274	pad_mem_bank[2]	Y25	O		DDR2 BANK 地址第 2 位（高位）
275	pad_mem_cas_n	AF19	O		DDR2 列地址选通信号，低电平有效
276	pad_mem_cke	AE21	O		DDR2 时钟使能信号，高电平有效
277	pad_mem_cs_n[0]	AC18	O		DDR2 片选 0，低电平有效
278	pad_mem_cs_n[1]	AE20	O		DDR2 片选 1，低电平有效
279	pad_mem_data[0]	AE16	I/O		DDR2 数据第 0 位（低位数据）
280	pad_mem_data[1]	AC16	I/O		DDR2 数据第 1 位
281	pad_mem_data[2]	AC15	I/O		DDR2 数据第 2 位
282	pad_mem_data[3]	AF15	I/O		DDR2 数据第 3 位
283	pad_mem_data[4]	AG16	I/O		DDR2 数据第 4 位
284	pad_mem_data[5]	AE17	I/O		DDR2 数据第 5 位
285	pad_mem_data[6]	AF16	I/O		DDR2 数据第 6 位
286	pad_mem_data[7]	AD16	I/O		DDR2 数据第 7 位
287	pad_mem_data[8]	AE18	I/O		DDR2 数据第 8 位
288	pad_mem_data[9]	AF17	I/O		DDR2 数据第 9 位
289	pad_mem_data[10]	AD17	I/O		DDR2 数据第 10 位
290	pad_mem_data[11]	AG17	I/O		DDR2 数据第 11 位
291	pad_mem_data[12]	AG18	I/O		DDR2 数据第 12 位
292	pad_mem_data[13]	AE19	I/O		DDR2 数据第 13 位
293	pad_mem_data[14]	AF18	I/O		DDR2 数据第 14 位
294	pad_mem_data[15]	AD20	I/O		DDR2 数据第 15 位
295	pad_mem_data[16]	V24	I/O		DDR2 数据第 16 位
296	pad_mem_data[17]	AB26	I/O		DDR2 数据第 17 位

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
297	pad_mem_data[18]	W25	I/O		DDR2 数据第 18 位
298	pad_mem_data[19]	W27	I/O		DDR2 数据第 19 位
299	pad_mem_data[20]	V27	I/O		DDR2 数据第 20 位
300	pad_mem_data[21]	U24	I/O		DDR2 数据第 21 位
301	pad_mem_data[22]	AA26	I/O		DDR2 数据第 22 位
302	pad_mem_data[23]	V26	I/O		DDR2 数据第 23 位
303	pad_mem_data[24]	T24	I/O		DDR2 数据第 24 位
304	pad_mem_data[25]	Y26	I/O		DDR2 数据第 25 位
305	pad_mem_data[26]	T25	I/O		DDR2 数据第 26 位
306	pad_mem_data[27]	U27	I/O		DDR2 数据第 27 位
307	pad_mem_data[28]	T27	I/O		DDR2 数据第 28 位
308	pad_mem_data[29]	R24	I/O		DDR2 数据第 29 位
309	pad_mem_data[30]	W26	I/O		DDR2 数据第 30 位
310	pad_mem_data[31]	P24	I/O		DDR2 数据第 31 位（高位数据）
311	pad_mem_ecc[0]	AB23	I/O		DDR2 ECC 校验数据第 0 位（低位）
312	pad_mem_ecc[1]	AA24	I/O		DDR2 ECC 校验数据第 1 位
313	pad_mem_ecc[2]	AD26	I/O		DDR2 ECC 校验数据第 2 位
314	pad_mem_ecc[3]	AA25	I/O		DDR2 ECC 校验数据第 3 位（高位）
315	pad_mem_dm[0]	AD15	I/O		DDR2 双向数据屏蔽信号
316	pad_mem_dm[1]	AC17	I/O		DDR2 双向数据屏蔽信号
317	pad_mem_dm[2]	U26	I/O		DDR2 双向数据屏蔽信号
318	pad_mem_dm[3]	T26	I/O		DDR2 双向数据屏蔽信号
319	pad_mem_ecc_dm	AF22	I/O		DDR2 ECC 数据掩码信号
320	pad_mem_dqs_m[0]	AB13	I/O		DDR2 控制器和 PHY 数据同步奇信号 0
321	pad_mem_dqs_p[0]	AB14	I/O		DDR2 控制器和 PHY 数据同步偶信号 0
322	pad_mem_dqs_m[1]	AD18	I/O		DDR2 控制器和 PHY 数据同步奇信号 1
323	pad_mem_dqs_p[1]	AD19	I/O		DDR2 控制器和 PHY 数据同步偶信号 1
324	pad_mem_dqs_m[2]	V25	I/O		DDR2 控制器和 PHY 数据同步奇信号 2
325	pad_mem_dqs_p[2]	U25	I/O		DDR2 控制器和 PHY 数据同步偶信号 2
326	pad_mem_dqs_m[3]	R25	I/O		DDR2 控制器和 PHY 数据同步奇信号 3

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
327	pad_mem_dqs_p[3]	R26	I/O		DDR2 控制器和 PHY 数据同步偶信号 3
328	pad_mem_ecc_dqs_m	AD24	I/O		DDR2 控制器和 PHY 校验位同步奇信号
329	pad_mem_ecc_dqs_p	AD25	I/O		DDR2 控制器和 PHY 校验位同步奇信号
330	pad_mem_clk[0]	AG23	O		DDR2 片选 0 使用的差分时钟奇信号
331	pad_mem_clk_n[0]	AG22	O		DDR2 片选 0 使用的差分时钟偶信号
332	pad_mem_clk[1]	AG21	O		DDR2 片选 1 使用的差分时钟奇信号
333	pad_mem_clk_n[1]	AG20	O		DDR2 片选 1 使用的差分时钟偶信号
334	pad_mem_odt[0]	Y27	O		DDR2 片选 0 对应的片内终结信号 (on-die termination), 高电平有效
335	pad_mem_odt[1]	W24	O		DDR2 片选 1 对应的片内终结信号 (on-die termination), 高电平有效
336	pad_mem_ras_n	AD21	O		DDR2 行地址选通, 低电平有效
337	pad_mem_reset_n	AG19	O		DDR 复位信号, DDR2 不起作用, 悬空
338	pad_mem_we_n	AA27	O		DDR2 写使能, 低电平有效
339	pci_ad[0]	N3	I/O		PCI 总线地址/数据信号第 0 位 (低位)
340	pci_ad[1]	N2	I/O		PCI 总线地址/数据第 1 位
341	pci_ad[2]	N1	I/O		PCI 总线地址/数据信号第 2 位
342	pci_ad[3]	M1	I/O		PCI 总线地址/数据信号第 3 位
343	pci_ad[4]	M2	I/O		PCI 总线地址/数据信号第 4 位
344	pci_ad[5]	M3	I/O		PCI 总线地址/数据信号第 5 位
345	pci_ad[6]	M4	I/O		PCI 总线地址/数据信号第 6 位
346	pci_ad[7]	M5	I/O		PCI 总线地址/数据信号第 7 位
347	pci_ad[8]	L6	I/O		PCI 总线地址/数据信号第 8 位
348	pci_ad[9]	L5	I/O		PCI 总线地址/数据信号第 9 位
349	pci_ad[10]	L4	I/O		PCI 总线地址/数据信号第 10 位
350	pci_ad[11]	L3	I/O		PCI 总线地址/数据信号第 11 位
351	pci_ad[12]	L2	I/O		PCI 总线地址/数据信号第 12 位
352	pci_ad[13]	L1	I/O		PCI 总线地址/数据信号第 13 位
353	pci_ad[14]	K1	I/O		PCI 总线地址/数据信号第 14 位
354	pci_ad[15]	K2	I/O		PCI 总线地址/数据信号第 15 位

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
355	pci_ad[16]	H1	I/O		PCI 总线地址/数据信号第 16 位
356	pci_ad[17]	H2	I/O		PCI 总线地址/数据信号第 17 位
357	pci_ad[18]	H3	I/O		PCI 总线地址/数据信号第 18 位
358	pci_ad[19]	H4	I/O		PCI 总线地址/数据信号第 19 位
359	pci_ad[20]	H5	I/O		PCI 总线地址/数据信号第 20 位
360	pci_ad[21]	H6	I/O		PCI 总线地址/数据信号第 21 位
361	pci_ad[22]	G6	I/O		PCI 总线地址/数据信号第 22 位
362	pci_ad[23]	G5	I/O		PCI 总线地址/数据信号第 23 位
363	pci_ad[24]	G2	I/O		PCI 总线地址/数据信号第 24 位
364	pci_ad[25]	G1	I/O		PCI 总线地址/数据信号第 25 位
365	pci_ad[26]	F1	I/O		PCI 总线地址/数据信号第 26 位
366	pci_ad[27]	F2	I/O		PCI 总线地址/数据信号第 27 位
367	pci_ad[28]	F3	I/O		PCI 总线地址/数据信号第 28 位
368	pci_ad[29]	F4	I/O		PCI 总线地址/数据信号第 29 位
369	pci_ad[30]	F5	I/O		PCI 总线地址/数据信号第 30 位
370	pci_ad[31]	E4	I/O		PCI 总线地址/数据信号第 31 位（高位）
371	pci_arb_gnt_n[0]	B2	O		PCI 总线占用允许（主模式），低电平有效
372	pci_arb_gnt_n[1]	B6	O		PCI 总线占用允许（主模式），低电平有效
373	pci_arb_gnt_n[2]	C3	O		PCI 总线占用允许（主模式），低电平有效
374	pci_arb_gnt_n[3]	A5	O		PCI 总线占用允许（主模式），低电平有效
375	pci_arb_gnt_n[4]	D4	O		PCI 总线占用允许（主模式），低电平有效
376	pci_arb_gnt_n[5]	A4	O		PCI 总线占用允许（主模式），低电平有效
377	pci_arb_gnt_n[6]	E5	O		PCI 总线占用允许（主模式），低电平有效
378	pci_host	G7	I	U	PCI 主桥/从桥模式配置，低电平表示主桥模式，高电平表示从桥模式
379	pci_arb_req_n[0]	A3	I		PCI 总线占用请求信号（主模式），低电平有效
380	pci_arb_req_n[1]	H7	I		PCI 总线占用请求信号（主模式），低电平有效
381	pci_arb_req_n[2]	E6	I		PCI 总线占用请求信号（主模式），低电平有效
382	pci_arb_req_n[3]	H8	I		PCI 总线占用请求信号（主模式），低电平有效
383	pci_arb_req_n[4]	F7	I		PCI 总线占用请求信号（主模式），低电平有效

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
384	pci_arb_req_n[5]	H9	I		PCI 总线占用请求信号（主模式），低电平有效
385	pci_arb_req_n[6]	F6	I		PCI 总线占用请求信号（主模式），低电平有效
386	pci_cbe_n[0]	M6	I/O		PCI 命令与字节使能复用信号
387	pci_cbe_n[1]	K3	I/O		PCI 命令与字节使能复用信号
388	pci_cbe_n[2]	J1	I/O		PCI 命令与字节使能复用信号
389	pci_cbe_n[3]	G3	I/O		PCI 命令与字节使能复用信号
390	pci_clk_in	E1	I		PCI 时钟输入，33MHz 或 66MHz
391	pci_devsel_n	J5	I/O		PCI 目标设备准备好标识信号，有效时表示目标设备准备好，低电平有效
392	pci_frame_n	J2	I/O		PCI 数据开始传输标识，启动 PCI 事务时该信号为低电平，有效；结束 PCI 事务时，该信号无效
393	pci_gnt_in_n	E2	I		PCI 总线申请占用允许（从模式）
394	pci_idsel_in	G4	I		PCI 目标设备选择信号
395	pci_inta_n	C1	I/O(OD)		PCI 总线中断 A，低电平有效（从桥模式输出信号，主桥模式输入信号）
396	pci_intb_n	C2	I		PCI 总线中断 B
397	pci_intc_n	D3	I		PCI 总线中断 C
398	pci_intd_n	D2	I		PCI 总线中断 D
399	pci_irdy_n	J3	I/O		PCI 主设备准备好标识信号
400	pci_par	K4	I/O		PCI 奇偶校验信号
401	pci_perr_n	K6	I/O		PCI 校验错误指示信号
402	pci_req_n	E3	O		PCI 总线申请占用请求（从模式）
403	pci_rst_n	D1	I/O		PCI 复位信号，低电平有效。（从桥模式输入信号，主桥模式输出信号）
404	pci_serr_n	K5	I/O(OD)		PCI 系统错误指示信号
405	sim_fast_reset	A6	I	D	测试信号，板级设计时接 1K Ω 电阻下拉
406	pci_stop_n	J6	I/O		PCI 目标设备停止数据传输
407	pci_trdy_n	J4	I/O		PCI 目标设备数据准备好标识信号
408	dsuact	G10	O		硬件调试响应指示信号有效时表示处理器处于调试模式，高电平有效

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
409	dsubre	B3	I		硬件调试模式控制输入，当 dsuen 为高电平，且在系统复位结束前三个时钟周期 dsubre 为高电平，处理器在复位结束后运行四条指令后立即进入调试模式
410	dsuen	C4	I		硬件调试模式使能，高电平时表示有效，此时若调试模式触发条件满足，处理器进入调试模式
411	dsurx	A7	I	U	硬件调试串口接收输入
412	dsutx	D6	O		硬件调试串口发送输出
413	pcinouse	AF10	I	U	PCI 使能开关，低电平使能 pci，高电平禁能 pci
414	bist_mode	AG9	I	D	MBIST 测试模式使能信号，高电平有效，板级设计时接 1K Ω 电阻下拉
415	scan_enable	AC5	I	D	SCAN 测试使能，高电平有效，板级设计时接 1K Ω 电阻下拉
416	scan_mode	AD9	I	D	SCAN 模式的选择，高电平有效，板级设计时接 1K Ω 电阻下拉
417	iddq_enable	AE9	I		IDDQ 测试使能，高电平有效，板级设计时接 4.7K Ω 电阻下拉
418	vss	N25	GND		地
419	vss	R21	GND		地
420	test1_clk	M27	I	D	测试用时钟，板级设计时接 1K Ω 电阻下拉
421	resetfull	E23	O		内部输出的复位指示信号
422	vss	R22	GND		地
423	io_vdd	R23	P		数字 3.3V 数字 3.3V IO 电源（数字 3.3V 电源）
424	io_vdd	R27	P		数字 3.3V IO 电源
425	io_vdd	P26	P		数字 3.3V IO 电源
426	vss	P25	GND		地
427	vdd	P27	P		数字 1.2V CORE 电源（数字 1.2V 电源）
428	vss	R20	GND		地
429	vdd	L27	P		数字 1.2V CORE 电源
430	N/C	M20	-		N/C

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
431	io_vdd	H24	P		数字 3.3V IO 电源
432	test2_clk	E27	I	D	测试用时钟，板级设计时接 1K Ω 电阻下拉
433	vss	M23	GND		地
434	vss	H26	GND		地
435	io_vdd	M22	P		数字 3.3V IO 电源
436	io_vdd	F25	P		数字 3.3V IO 电源
437	io_vdd	N21	P		数字 3.3V IO 电源
438	io_vdd	G24	P		数字 3.3V IO 电源
439	io_vdd	K22	P		数字 3.3V IO 电源
440	io_vdd	D27	P		数字 3.3V IO 电源
441	io_vdd	J22	P		数字 3.3V IO 电源
442	io_vdd	C27	P		数字 3.3V IO 电源
443	io_vdd	H22	P		数字 3.3V IO 电源
444	io_vdd	G23	P		数字 3.3V IO 电源
445	io_vdd	M21	P		数字 3.3V IO 电源
446	io_vdd	L22	P		数字 3.3V IO 电源
447	vss	K26	GND		地
448	io_vdd	N22	P		数字 3.3V IO 电源
449	io_vdd	N23	P		数字 3.3V IO 电源
450	io_vdd	F26	P		数字 3.3V IO 电源
451	vss	P22	GND		地
452	vss	J26	GND		地
453	vdd	H25	P		数字 1.2V CORE 电源
454	io_vdd	J25	P		数字 3.3V IO 电源
455	vdd	K27	P		数字 1.2V CORE 电源
456	pll_bp	AF13	I		PLL 时钟旁路控制信号，高电平时旁路 PLL，片内时钟直接由晶振提供；低电平时，片内时钟由晶振经过 PLL 电路后提供
457	ddr_pll[0]	AC7	I		DDR2 时钟 PLL 分频系数，[0]为最低位，[2]为最高位，3 位不可同时配置成 0

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
458	ddr_pll[1]	AE11	I		DDR2 时钟 PLL 分频系数, [0]为最低位, [2]为最高位, 3 位不可同时配置成 0
459	ddr_pll[2]	AA12	I		DDR2 时钟 PLL 分频系数, [0]为最低位, [2]为最高位, 3 位不可同时配置成 0
460	pll_din[0]	AD11	I		主时钟 PLL 分频系数, [0]为最低位, [2]为最高位, 3 位不可同时配置成 0
461	pll_din[1]	AB12	I		主时钟 PLL 分频系数, [0]为最低位, [2]为最高位, 3 位不可同时配置成 0
462	pll_din[2]	AF12	I		主时钟 PLL 分频系数, [0]为最低位, [2]为最高位, 3 位不可同时配置成 0
463	pll_fout4phasepd	AC12	I		主时钟、DDR2 时钟、MAC 时钟三个 PLL 的相位时钟掉电使能信号, 高电平有效; 测试时使用, 板级设计时通过 4.7K Ω 电阻上拉, 以节省功耗
464	pll_lock_ddr	AD14	O		DDR2 时钟 PLL 锁定指示信号, 高有效
465	pll_lock_mac	AG14	O		MAC 时钟 PLL 锁定指示信号, 高有效
466	pll_lock_main	AD7	O		主时钟 PLL 锁定指示信号, 高有效
467	pll_min[0]	AE10	I		主时钟 PLL 倍频系数, [0]为最低位, [2]为最高位, 3 位不可同时配置成 0
468	pll_min[1]	AD12	I		主时钟 PLL 倍频系数, [0]为最低位, [2]为最高位, 3 位不可同时配置成 0
469	pll_min[2]	AC6	I		主时钟 PLL 倍频系数, [0]为最低位, [2]为最高位, 3 位不可同时配置成 0
470	pll_opt	AD3	I		PLL 测试加固模式使能信号, 高电平有效。pll_opt 为低时, 时钟质量更好一些, 在高频下表现更明显, 使用时通过 4.7K Ω 电阻下拉
471	pll_out_ddr_d8	AC13	O		DDR2 时钟 PLL 的 8 分频时钟输出
472	pll_out_mac_d8	AE13	O		MAC 时钟 PLL 的 8 分频时钟输出
473	pll_out_main_d8	AD13	O		主时钟 PLL 的 8 分频时钟输出
474	N/C	AC8	-		N/C
475	pll_pd_main	AG13	I		主时钟 PLL 掉电使能, 掉电后没有主时钟输出,

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
					高电平有效
476	pll_pd_dds	AB11	I		DDR2 时钟 PLL 掉电使能，掉电后没有 DDR2 时钟输出，高电平有效
477	pll_pd_mac	AE12	I		MAC 时钟 PLL 掉电使能，掉电后没有 MAC 时钟输出，高电平有效
478	N/C	J27	-		N/C
479	vss	G27	GND		地
480	N/C	G26	-		N/C
481	vss	P23	GND		地
482	N/C	K25	-		N/C
483	N/C	P21	-		N/C
484	io_vdd	M24	P		数字 3.3V IO 电源
485	N/C	L24	-		N/C
486	N/C	E26	-		N/C
487	N/C	H27	-		N/C
488	N/C	K24	-		N/C
489	vss	K23	GND		地
490	io_vdd	F27	P		数字 3.3V IO 电源
491	N/C	G25	-		N/C
492	N/C	H23	-		N/C
493	io_vdd	J23	P		数字 3.3V IO 电源
494	N/C	J24	-		N/C
495	vss	L23	GND		地
496	N/C	N27	-		N/C
497	vss	M26	GND		地
498	N/C	L25	-		N/C
499	io_vdd	L26	P		数字 3.3V IO 电源
500	vss	T23	GND		地
501	N/C	N26	-		N/C
502	N/C	M25	-		N/C

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
503	N/C	N24	-		N/C
504	pll_io_vdd	AC10	P		锁相环数字 3.3V IO 电源（模拟 3.3V 电源）
505	pll_io_vss	AG15	GND		锁相环 IO 地
506	pll_vddh	AE15	P		锁相环模拟电源（模拟 3.3V 电源）
507	pll_vddpost_ddr	AC9	P		锁相环 DDR 内核电源（数字 1.2V 电源）
508	pll_vddpost_mac	AC14	P		锁相环 MAC 内核电源（数字 1.2V 电源）
509	pll_vddpost	AE14	P		锁相环 MAIN 内核电源（数字 1.2V 电源）
510	pll_vddref	AF14	P		锁相环参考电源（数字 1.2V 电源）
511	pll_vss	AA13	GND		锁相环内核地
512	io_vdd	AA15	P		数字 3.3V IO 电源
513	io_vdd	AF24	P		数字 3.3V IO 电源
514	io_vdd	C25	P		数字 3.3V IO 电源
515	io_vdd	G15	P		数字 3.3V IO 电源
516	io_vdd	H12	P		数字 3.3V IO 电源
517	io_vdd	J11	P		数字 3.3V IO 电源
518	io_vdd	J14	P		数字 3.3V IO 电源
519	io_vdd	J18	P		数字 3.3V IO 电源
520	io_vdd	K13	P		数字 3.3V IO 电源
521	io_vdd	K15	P		数字 3.3V IO 电源
522	io_vdd	K17	P		数字 3.3V IO 电源
523	io_vdd	L9	P		数字 3.3V IO 电源
524	io_vdd	L12	P		数字 3.3V IO 电源
525	io_vdd	L16	P		数字 3.3V IO 电源
526	io_vdd	L18	P		数字 3.3V IO 电源
527	io_vdd	M8	P		数字 3.3V IO 电源
528	io_vdd	M11	P		数字 3.3V IO 电源
529	io_vdd	M15	P		数字 3.3V IO 电源
530	io_vdd	N19	P		数字 3.3V IO 电源
531	io_vdd	P9	P		数字 3.3V IO 电源
532	io_vdd	P11	P		数字 3.3V IO 电源

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
533	io_vdd	R12	P		数字 3.3V IO 电源
534	io_vdd	R17	P		数字 3.3V IO 电源
535	io_vdd	R19	P		数字 3.3V IO 电源
536	io_vdd	T10	P		数字 3.3V IO 电源
537	io_vdd	U11	P		数字 3.3V IO 电源
538	io_vdd	U16	P		数字 3.3V IO 电源
539	io_vdd	V10	P		数字 3.3V IO 电源
540	io_vdd	V12	P		数字 3.3V IO 电源
541	io_vdd	V19	P		数字 3.3V IO 电源
542	io_vdd	W13	P		数字 3.3V IO 电源
543	io_vdd	Y10	P		数字 3.3V IO 电源
544	io_vdd	Y12	P		数字 3.3V IO 电源
545	io_vdd	Y13	P		数字 3.3V IO 电源
546	vdd_io	AA17	P		数字 1.8V DDR2 IO 电源
547	vdd_io	AB16	P		数字 1.8V DDR2 IO 电源
548	vdd_io	AB18	P		数字 1.8V DDR2 IO 电源
549	vdd_io	AB22	P		数字 1.8V DDR2 IO 电源
550	vdd_io	AC27	P		数字 1.8V DDR2 IO 电源
551	vdd_io	AE26	P		数字 1.8V DDR2 IO 电源
552	vdd_io	L11	P		数字 1.8V DDR2 IO 电源
553	vdd_io	M13	P		数字 1.8V DDR2 IO 电源
554	vdd_io	M17	P		数字 1.8V DDR2 IO 电源
555	vdd_io	N10	P		数字 1.8V DDR2 IO 电源
556	vdd_io	N14	P		数字 1.8V DDR2 IO 电源
557	vdd_io	N16	P		数字 1.8V DDR2 IO 电源
558	vdd_io	P15	P		数字 1.8V DDR2 IO 电源
559	vdd_io	P20	P		数字 1.8V DDR2 IO 电源
560	vdd_io	R11	P		数字 1.8V DDR2 IO 电源
561	vdd_io	R14	P		数字 1.8V DDR2 IO 电源
562	vdd_io	R16	P		数字 1.8V DDR2 IO 电源

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
563	vdd_io	T13	P		数字 1.8V DDR2 IO 电源
564	vdd_io	T15	P		数字 1.8V DDR2 IO 电源
565	vdd_io	T22	P		数字 1.8V DDR2 IO 电源
566	vdd_io	U14	P		数字 1.8V DDR2 IO 电源
567	vdd_io	U21	P		数字 1.8V DDR2 IO 电源
568	vdd_io	V20	P		数字 1.8V DDR2 IO 电源
569	vdd_io	W17	P		数字 1.8V DDR2 IO 电源
570	vdd_io	W19	P		数字 1.8V DDR2 IO 电源
571	vdd_pre	AB17	P		数字 1.8V DDR2 PRE 电源
572	vdd_pre	AB21	P		数字 1.8V DDR2 PRE 电源
573	vdd_pre	AE25	P		数字 1.8V DDR2 PRE 电源
574	vdd_pre	L8	P		数字 1.8V DDR2 PRE 电源
575	vdd_pre	N12	P		数字 1.8V DDR2 PRE 电源
576	vdd_pre	P13	P		数字 1.8V DDR2 PRE 电源
577	vdd_pre	V21	P		数字 1.8V DDR2 PRE 电源
578	vdd_pre	V23	P		数字 1.8V DDR2 PRE 电源
579	vdd_pre	W22	P		数字 1.8V DDR2 PRE 电源
580	vref_0	W21	P		DDR2 电源 REF (0.9V)
581	ddr_add_vref	AC11	P		DDR2 电源 REF (0.9V)
582	vref_vss_0	W20	GND		DDR2 地 REF
583	vref_vss_1	AB15	GND		DDR2 地 REF
584	vss_io	AA18	GND		DDR2 地 IO
585	vss_io	AA23	GND		DDR2 地 IO
586	vss_io	AB19	GND		DDR2 地 IO
587	vss_io	AB27	GND		DDR2 地 IO
588	vss_io	AC20	GND		DDR2 地 IO
589	vss_io	AE27	GND		DDR2 地 IO
590	vss_io	AG25	GND		DDR2 地 IO
591	vss_io	H10	GND		DDR2 地 IO
592	vss_io	H18	GND		DDR2 地 IO

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
593	vss_io	J16	GND		DDR2 地 IO
594	vss_io	K7	GND		DDR2 地 IO
595	vss_io	L14	GND		DDR2 地 IO
596	vss_io	P12	GND		DDR2 地 IO
597	vss_io	P16	GND		DDR2 地 IO
598	vss_io	R18	GND		DDR2 地 IO
599	vss_io	T14	GND		DDR2 地 IO
600	vss_io	T19	GND		DDR2 地 IO
601	vss_io	U18	GND		DDR2 地 IO
602	vss_io	U23	GND		DDR2 地 IO
603	vss_io	V15	GND		DDR2 地 IO
604	vss_io	V17	GND		DDR2 地 IO
605	vss_io	W18	GND		DDR2 地 IO
606	vss_io	Y14	GND		DDR2 地 IO
607	vss_io	Y17	GND		DDR2 地 IO
608	vss_pre	AA19	GND		DDR2 地 PRE
609	vss_pre	AA21	GND		DDR2 地 PRE
610	vss_pre	AB20	GND		DDR2 地 PRE
611	vss_pre	AC23	GND		DDR2 地 PRE
612	vss_pre	AD27	GND		DDR2 地 PRE
613	vss_pre	J7	GND		DDR2 地 PRE
614	vss_pre	K8	GND		DDR2 地 PRE
615	vss_pre	K10	GND		DDR2 地 PRE
616	vss_pre	K12	GND		DDR2 地 PRE
617	vss_pre	M7	GND		DDR2 地 PRE
618	vss_pre	Y16	GND		DDR2 地 PRE
619	vss_pre	Y18	GND		DDR2 地 PRE
620	vss_pre	Y22	GND		DDR2 地 PRE
621	vdd	AA20	P		数字 1.2V CORE 电源
622	vdd	AG24	P		数字 1.2V CORE 电源

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
623	vdd	C26	P		数字 1.2V CORE 电源
624	vdd	E19	P		数字 1.2V CORE 电源
625	vdd	H11	P		数字 1.2V CORE 电源
626	vdd	H15	P		数字 1.2V CORE 电源
627	vdd	H17	P		数字 1.2V CORE 电源
628	vdd	J9	P		数字 1.2V CORE 电源
629	vdd	J13	P		数字 1.2V CORE 电源
630	vdd	J19	P		数字 1.2V CORE 电源
631	vdd	K16	P		数字 1.2V CORE 电源
632	vdd	K19	P		数字 1.2V CORE 电源
633	vdd	L7	P		数字 1.2V CORE 电源
634	vdd	L15	P		数字 1.2V CORE 电源
635	vdd	L17	P		数字 1.2V CORE 电源
636	vdd	M10	P		数字 1.2V CORE 电源
637	vdd	M14	P		数字 1.2V CORE 电源
638	vdd	M16	P		数字 1.2V CORE 电源
639	vdd	M19	P		数字 1.2V CORE 电源
640	vdd	N9	P		数字 1.2V CORE 电源
641	vdd	N18	P		数字 1.2V CORE 电源
642	vdd	N20	P		数字 1.2V CORE 电源
643	vdd	P17	P		数字 1.2V CORE 电源
644	vdd	P19	P		数字 1.2V CORE 电源
645	vdd	R10	P		数字 1.2V CORE 电源
646	vdd	T11	P		数字 1.2V CORE 电源
647	vdd	T17	P		数字 1.2V CORE 电源
648	vdd	T20	P		数字 1.2V CORE 电源
649	vdd	U9	P		数字 1.2V CORE 电源
650	vdd	U12	P		数字 1.2V CORE 电源
651	vdd	U19	P		数字 1.2V CORE 电源
652	vdd	V13	P		数字 1.2V CORE 电源

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
653	vdd	W14	P		数字 1.2V CORE 电源
654	vdd	W16	P		数字 1.2V CORE 电源
655	vdd	W23	P		数字 1.2V CORE 电源
656	vdd	Y11	P		数字 1.2V CORE 电源
657	vdd	Y15	P		数字 1.2V CORE 电源
658	vdd	Y19	P		数字 1.2V CORE 电源
659	vdd	Y21	P		数字 1.2V CORE 电源
660	vss	AA14	GND		地
661	vss	AA22	GND		地
662	vss	AC19	GND		地
663	vss	AF26	GND		地
664	vss	B26	GND		地
665	vss	H13	GND		地
666	vss	H16	GND		地
667	vss	H19	GND		地
668	vss	J8	GND		地
669	vss	J10	GND		地
670	vss	J12	GND		地
671	vss	J15	GND		地
672	vss	J17	GND		地
673	vss	J20	GND		地
674	vss	K9	GND		地
675	vss	K11	GND		地
676	vss	K14	GND		地
677	vss	K18	GND		地
678	vss	L10	GND		地
679	vss	L13	GND		地
680	vss	L19	GND		地
681	vss	M9	GND		地
682	vss	M12	GND		地

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
683	vss	M18	GND		地
684	vss	N8	GND		地
685	vss	N11	GND		地
686	vss	N13	GND		地
687	vss	N15	GND		地
688	vss	N17	GND		地
689	vss	P10	GND		地
690	vss	P14	GND		地
691	vss	P18	GND		地
692	vss	R13	GND		地
693	vss	R15	GND		地
694	vss	T12	GND		地
695	vss	T16	GND		地
696	vss	T18	GND		地
697	vss	T21	GND		地
698	vss	U10	GND		地
699	vss	U13	GND		地
700	vss	U15	GND		地
701	vss	U17	GND		地
702	vss	U20	GND		地
703	vss	U22	GND		地
704	vss	V11	GND		地
705	vss	V14	GND		地
706	vss	V16	GND		地
707	vss	V18	GND		地
708	vss	W12	GND		地
709	vss	W15	GND		地
710	vss	Y20	GND		地
711	vss	Y23	GND		地
712	N/C	AA16	-		N/C

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
713	N/C	V22	-		N/C
714	N/C	AA11	-		N/C
715	N/C	AB8	-		N/C
716	N/C	AF8	-		N/C
717	N/C	AF9	-		N/C

注：

数字 3.3V 电源（V_{D33}）包括：io_vdd

模拟 3.3V 电源（V_{A33}）包括：pll_io_vdd、pll_vddh

数字 1.8V 电源（V₁₈）包括：vdd_pre、vdd_io

数字 1.2V 电源（V₁₂）包括：pll_vddpost_ddr、pll_vddpost_mac、pll_vddpost、pll_vddref、vdd

0.9V 信号（V_{18/2}）包括：vref_0、ddr_add_vref

数字地（GND）包括：vss、vref_vss_0、vref_vss_1、vss_io、vss_pre、pll_vss、pll_io_vss

注：

(1) “上拉/下拉”列中的“—”表示内部 pad 无上拉/下拉；

(2) “上拉/下拉”列中的“上拉/下拉”表示内部 pad 有上拉/下拉，U 表示上拉，D 表示下拉，内部上拉/下拉阻值为 15KΩ~120KΩ。



注意 一般情况下，内部上下拉的管脚，如果不使用，外部不需要进行上下拉处理，

但是考虑到内部上下拉阻值比较高，如果外部干扰太大，有一定的风险受到影响，所以推荐用户将不使用的输入管脚进行外部上下拉处理。但是考虑到板级设计电阻使用过多的情况，部分上下拉管脚可以不必要再做外部上下拉处理，必须要上下拉的管脚如下表 3-2-2 所示。

表3-2-2 必须上下拉管脚列表

序号	管脚名称	引出端	I/O 类型	上拉/ 下拉	说明
1	trst	B22	I	U	JTAG 复位,低电平有效, 不使用时需要接 1k Ω 电阻下拉
2	nmi	A22	I	D	非屏蔽中断, 不使用时接 1k Ω 电阻下拉
3	dsurx	A7	I	U	硬件调试串口接收输入, 在运行模式时接 1k Ω 电阻上拉。
4	pcinouse	AF10	I	U	PCI 使能开关, 不使用时接 1k Ω 电阻上拉
5	bist_mode	AG9	I	D	MBIST 测试模式使能信号, 高电平有效, 板级设计时接 1K Ω 电阻下拉
6	scan_enable	AC5	I	D	SCAN 使能, 高电平有效, 板级设计时接 1K Ω 电阻下拉
7	scan_mode	AD9	I	D	SCAN 模式的选择, 高电平有效, 板级设计时接 1K Ω 电阻下拉

BM3823AMCCRH 输出信号驱动见表 3-2-3 所示。

表3-2-3 输出信号驱动列表

I _{OH}	-0.5mA	-0.5mA (双向输出)	无 (OD 双向)	-4mA (三态输出)	无 (OD 输出)
I _{OL}	1.5mA	1.5mA (双向输出)	1.5mA (OD 双向)	4mA (三态输出)	4mA (OD 输出)
数量	8	44	2	1	2
管脚名	pci_req_n pci_arb_gnt_n[6:0]	pci_par pci_perr_n pci_stop_n pci_devsel_n pci_trdy_n pci_irdy_n pci_frame_n pci_cbe_n[3:0] pci_ad[31:0] pci_rst_n	pci_inta_n pci_serr_n	tdo	errorn wdogn

表3-2-3（续）

I_{OH}	-8mA			-8mA (双向输出)	无 (OD 双向)
I_{OL}	8mA			8mA (双向输出)	8mA (OD 双向)
数量	103			75	4
管脚名	gmac_phy_txd_o[7:0] gmac_phy_txen_o gmac_phy_txer_o gmac_pcs_aquired_sync_o gmac_pcs_en_cdet_o gmac_pcs_ewrap_o gmac_pcs_lck_ref_o gmac_clk_rmii gmac_clk_tx gmac_clk_tx_125 gmac_mdc	address[27:0] romsn[1:0] iosn[5:0] ios[5:0] oen oe writen write ramsn[4:0] ramoen[4:0] rwen[3:0] rbn[3:0] spw_clk resetfull	i2c_mcs0 i2c_hsmode0 i2c_mcs1 i2c_hsmode1 spi_csn0_0 spi_csn0_1 spi_clk0 spi_csn1_0 spi_csn1_1 spi_clk1 sdcasn[1:0] sdrasn sdcasn sdwen sddqm[3:0]	pio[31:0] data[31:0] pardata[7:0] gmac_mdio spi_mosi0 spi_mosi1	i2c_scl0 i2c_sda0 i2c_scl1 i2c_sda1

表3-2-3（续）

I_{OH}	-4mA			-4mA (双向输出)	-16mA
I_{OL}	4mA			4mA (双向输出)	16mA
数量	20		31	51	1
管脚名	pll_out_main_d8 pll_out_ddr_d8 pll_out_mac_d8 pll_lock_main pll_lock_ddr pll_lock_mac timer0 timer1 writen_1553 write_1553	pwmtimer0 pwmtimer1 pwmtimer2 pwmtimer3 pwmtimer4 pwmtimer5 txd3 txd4 dsutx dsuact	pad_mem_address[14:0] pad_mem_bank[2:0] pad_mem_cas_n pad_mem_cke pad_mem_clk[1:0] pad_mem_clk_n[1:0] pad_mem_cs_n[1:0] pad_mem_odt[1:0] pad_mem_ras_n pad_mem_reset_n pad_mem_we_n	pad_mem_ecc_dm pad_mem_data[31:0] pad_mem_dm[3:0] pad_mem_dqs_m[3:0] pad_mem_dqs_p[3:0] pad_mem_ecc[3:0] pad_mem_ecc_dqs_m pad_mem_ecc_dqs_p	sdclk

3.3 外形尺寸说明

BM3823AMCCRH 采用 717 引出端柱栅阵列封装 (CCGA717), 外壳外形及尺寸符合下图 3-3-1 和表 3-3-1 规定。

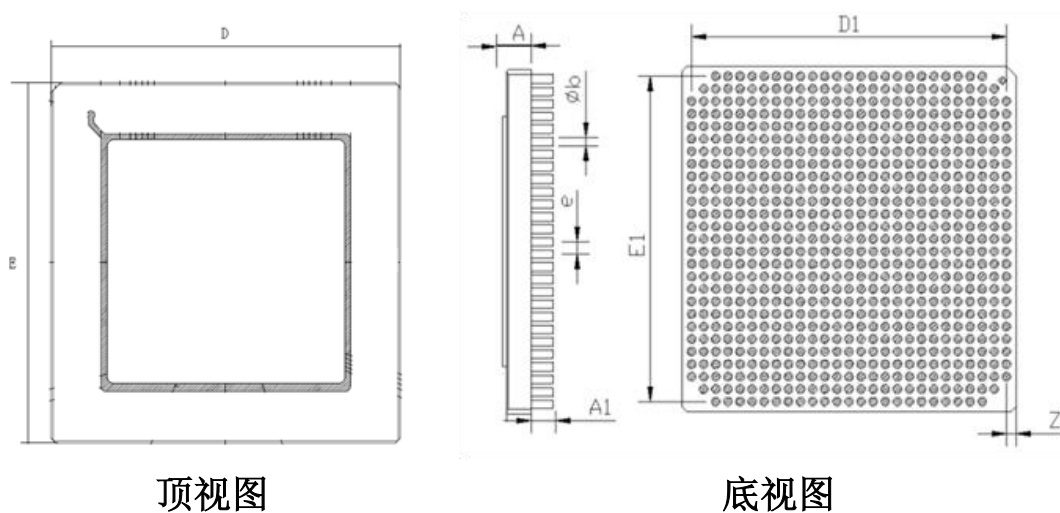


图 3-3-1 外形尺寸图

表3-3-1 外形尺寸说明

单位: mm

尺寸符号	数 值		
	最 小	公 称	最 大
A	2.50	3.05	3.50
A1	2.00	2.20	2.40
Z	0.70	0.94	1.30
D	34.35	35.0	35.65
E	34.35	35.0	35.65
D1	—	33.02	—
E1	—	33.02	—
e	—	1.27	—
φb	0.41	0.51	0.61

四、产品功能

BM3823AMCCRH 是基于 SPARC V8 体系结构的 32 位高性能抗辐射处理器，集成高性能整数处理单元、浮点处理单元、独立的指令和数据 Cache、硬件乘/除法器、256KB 片上存储器、DDR2 存储器控制器、高速 Ethernet MAC、DMA 控制器、PCI 总线控制器、中断控制器及 SPI 等其它丰富的外设接口，与 PROM 等相关外围电路组成完整的单板计算机系统，满足空间恶劣环境应用要求，可广泛应用于高可靠的信息处理系统及测控系统。

4.1 产品结构框图及核心

结构框图如图 4-1-1 所示。

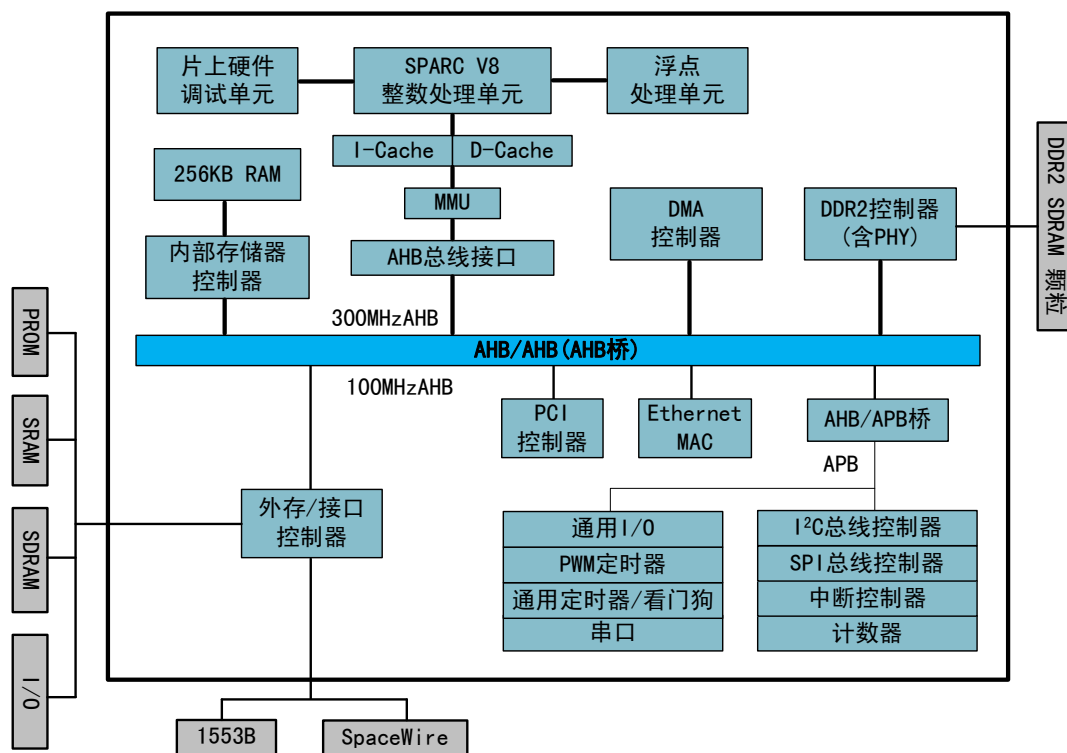


图4-1-1 BM3823AMCCRH电路结构框图

BM3823AMCCRH（以下简称 BM3823A）处理器核心的主要功能是确保程序的正确执行。BM3823A 处理器核心具有两种状态：用户态和管理态。通过处理器状态寄存器中 S 字段可配置处理器处于用户态还是管理态。在管理态下，处理器能够执行包括特权指令在内的所有指令；在用户态下，试图执行特权指令将会触发管理软件的陷阱。

本节除了介绍处理器核心部件整数单元（IU）和浮点处理单元（FPU）之外，还介绍了

处理器的启动方式、复位、异常和容错处理。

4.1.1 整数单元

BM3823A 包含一个整数处理单元 IU，IU 有自己独立的 32 位寄存器。

IU 相关的寄存器及状态寄存器说明详见附录 3.1。

IU 实现了 SPARC V8 标准定义的整数指令集，其特性如下：

- ◆ 6 级指令流水：IU 单元使用 6 级指令流水线，以接近每时钟周期执行一条指令的速率执行指令；
- ◆ 8 个寄存器窗口；
- ◆ 硬件乘/除法器：可以实现两个 32 位整数相乘，得到 64 位乘积，以及 64 位被除数与 32 位除数相除，得到 32 位商；
- ◆ 寄存器三模冗余结构，寄存器堆 EDAC 容错。

IU 执行的指令分为 6 种功能类别：加载/存储，算术/逻辑/移位，控制转移，读/写控制寄存器，浮点操作及其它指令。指令的执行周期如下表 4-1-1 所示。

表4-1-1 IU指令执行周期（假设Cache命中）

指令名称	执行周期
JMPL（跳转指令）	2
LDD（双字加载指令）	2
ST（单字存储指令）	2
STD（双字存储指令）	3
SMUL/UMUL（有符号/无符号整数乘法指令）	4
SDIV/UDIV（有符号/无符号整数除法指令）	35
TICC（陷阱指令）	4
LDSTUB（原子加载/存储指令）	3
其他指令	1

4.1.2 浮点处理单元

BM3823A 包含一个浮点处理单元 FPU，FPU 有自己独立的 32 位寄存器。FPU 浮点数据格式和指令集符合 ANSI/IEEE754-1985 标准。

FPU 实现了 SPARC V8 标准定义的浮点指令，可以处理单精度数和双精度数，但不支持四精度浮点指令操作。浮点指令的执行与 IU 串行，当 FPU 中指令没有执行完时，IU 处于等待状态。

FPU 包括 32 个 32 位通用浮点寄存器（%f0~%f31）。与 IU 的寄存器窗口不同，FPU 没有窗口机制，所有函数执行时访问的是同样的 32 个浮点寄存器。浮点寄存器能够被 FPop（FPop1/FPop2 格式）指令，浮点加载/存储指令（LDF，LDDF，STF，STDF）读写访问。

FPU 通过 IU 进行存储器访问，相关的异常有 fp_disable 和 fp_exception 两种入口，异常的处理由 IU 完成。浮点处理单元相关寄存器及状态寄存器说明详见附录 3.1。

4.1.3 工作模式

BM3823A 有两种工作模式：运行模式和调试模式。

◆ 运行模式

dsuen 和 dsubre 管脚设置为低电平，复位后进入运行模式，从地址 0x0 开始执行程序。0x0 地址属于 PROM，需要通过 pio[1: 0]正确配置 PROM 宽度：“00”表示 8 位，“01”表示 16 位，“1x”表示 32 位。当使用带 EDAC 功能的 8 位 PROM 启动时，因为校验位的空间起始于 PROM Bank 的最大地址处，所以在启动前不需要确定 PROM 空间范围。

◆ 调试模式

dsuen 和 dsubre 管脚设置为高电平，复位后从地址 0x0 开始执行 4 条指令后进入调试模式，此时可以通过调试串口调试 BM3823A。

4.1.4 产品复位

BM3823A 为低电平复位。在时钟有效后 resetn 信号保持 10 个时钟周期的低电平，就可以完成复位。复位信号拉高后，经过 3 个 sdclk 时钟周期的上升沿，处理器给出 PROM 地址信号，再经过 1 个 sdclk 时钟周期的上升沿，处理器给出 PROM 片选和读使能信号。

4.1.5 产品异常

BM3823A 的陷阱以及优先级见表 4-1-2。

表4-1-2 BM3823A的陷阱以及优先级

陷阱	陷阱类型	优先级	说明
reset	0x00	1	上电复位
write_error	0x1A	2	写错误
instruction_access_error	0x01	3	无法被纠正的取指错误
privileged_instruction	0x03	4	用户模式下执行特权指令
illegal_instruction	0x02	5	UNIMP 或其他未实现的命令
fp_disabled	0x04	6	不使能 FPU 时执行浮点指令
cp_disabled	0x24	6	不使能协处理器时执行协处理器指令
watchpoint_detected	0x0B	7	探测到观察点
window_overflow	0x05	8	SAVE 进入无效窗口
window_underflow	0x06	8	RESTORE 进入无效窗口
register_hardware_error	0x20	9	检测到不可纠正的寄存器堆错误
mem_address_not_aligned	0x07	10	访问存储器时地址未对齐
fp_exception	0x08	11	FPU 异常
data_access_exception	0x09	13	LOAD 指令访问错误
tag_overflow	0x0A	14	Tagged 运算溢出
divide_exception	0x2A	15	被 0 除
trap_instruction	0x80~0xFF	16	软件异常指令 TA
interrupt_level_15	0x1F	17	异步中断 15
interrupt_level_14	0x1E	18	异步中断 14
interrupt_level_13	0x1D	19	异步中断 13
interrupt_level_12	0x1C	20	异步中断 12
interrupt_level_11	0x1B	21	异步中断 11
interrupt_level_10	0x1A	22	异步中断 10
interrupt_level_9	0x19	23	异步中断 9
interrupt_level_8	0x18	24	异步中断 8
interrupt_level_7	0x17	25	异步中断 7
interrupt_level_6	0x16	26	异步中断 6
interrupt_level_5	0x15	27	异步中断 5
interrupt_level_4	0x14	28	异步中断 4
interrupt_level_3	0x13	29	异步中断 3
interrupt_level_2	0x12	30	异步中断 2
interrupt_level_1	0x11	31	异步中断 1

4.1.6 容错设计

容错设计采用三模冗余和 EDAC 技术，对 IU 内部寄存器采用三模冗余结构，对寄存器堆（Regfile）中的数据采用 Hamming 码进行 EDAC 保护，可以检测两位错，纠正一位错。

当检测到可以纠正的数据错误时，流水线断流，将纠正后的数据按原读出地址写回 Regfile，然后重启流水线，从断流位置重新运行程序；当检测到不可纠正的数据错误时，采用陷阱方式处理，引起 0x20 陷阱。

ASR16 和 ASR17 控制 Regfile 的 EDAC 功能，相关寄存器说明详见附录 3.1。

4.2 Cache

4.2.1 概述

BM3823A 的处理器内核采用哈佛结构，具有独立的指令 Cache 和数据 Cache。为了提高处理器内核的速度，指令 Cache 和数据 Cache 都采用了多路组相联（multi-set-caches）技术。Cache 的替换算法采用 LRU 算法。数据 Cache 容量为 16KB，指令 Cache 容量为 32KB。

4.2.2 Cache 缓存操作

外部存储器的部分区域及内部存储器区域能通过 Cache 进行缓存，可缓存的地址空间包括 PROM 和 RAM 区域，具体说明见表 4-2-1。

表4-2-1 Cache缓存的地址空间列表

地址范围	区域	能否通过 Cache 缓存
0x0000_0000 ~ 0x0FFF_FFFF	PROM 区域	能
0x1000_0000 ~ 0x1FFF_FFFF	I/O 空间区域	否
0x2000_0000 ~ 0x3FFF_FFFF	外部 RAM 区域	能
0xa000_0000 ~ 0xa003_FFFF	内部 RAM 区域	能
其它	其他存储区域	否

Cache 在使用时通过使用不同的 ASI 映射表示不同的 Cache 操作，ASI 映射方式见表 4-2-2。

表4-2-2 ASI映射情况说明

ASI	Usage
0x0, 0x1, 0x2, 0x3	强制Cache不命中，若数据被Cache缓存过则更新Cache，若数据不在Cache中但地址可缓存则分配新Cache块进行替换
0x4, 0x7	强制Cache不命中，若数据被Cache缓存过则更新Cache
0x5	刷新指令Cache
0x6	刷新数据Cache
0x8, 0x9, 0xA, 0xB	正常Cache访问
0xC	访问指令Cache tags
0xD	访问指令Cache data
0xE	访问数据Cache tags
0xF	访问数据Cache data

利用 LDA/STA 指令，可以访问不同的地址空间。如果 Cache 使能，通过 ASI 0x8 - ASI 0xB 可以正常访问指令和数据；通过 ASI 0xC - ASI 0xF 可以诊断指令 Cache 和数据 Cache。指令 Cache 采用 4 路组相联方式，每组（set）的大小为 8KB。数据 Cache 采用 2 路组相联方式，每组（set）的大小为 8KB。通过 ASI 0xC - ASI 0xF 访问指令 Cache 和数据 Cache 时，除了要给 ASI 命令提供物理地址外，还需提供 set 的值，具体公式如下所示：

$$\text{ASI命令使用的地址} = (\text{set值} \ll 13) + (\text{物理地址低13位})$$

其中，“set 值 $\ll 13$ ”是 set 空间的基地址，“物理地址低 13 位”是偏移地址。对于数据 Cache，set 值可以为“0”或“1”，对于指令 Cache，set 值可以为“0”或“1”或“2”或“3”。

从数据 Cache 中读取 tag 和 data 时，需分别读 set 为“0”和“1”两种情况下的结果，其中有且仅有一种情况下结果正确。

从指令 Cache 中读取 tag 和 data 时，需分别读 set 为“0”、“1”、“2”和“3”四种情况下的结果，其中有且仅有一种情况下结果正确。

4.2.3 Cache 容错功能

Cache 采用了奇偶校验的容错策略，当奇偶校验检测出数据/指令出错时，将直接从存储器中读取数据/指令，并将正确的数据/指令写入相应的 Cache 中。

Cache 容错功能的诊断主要通过 **Cache 容错控制寄存器**实现。主要功能包括对不同 Cache 区域的造错以及错误计数，测试 Cache 容错功能的正确性。本功能主要用于 Cache 容错功能的测试，用户可以了解，但是使用时不需要关心。

4.2.4 Cache Tag 信息说明

指令 Cache 和数据 Cache 是用户关心的两个缓存，除了 Cache 存储体存储的数据之外，有些时候用户会关心 Tag 信息，本节主要描述指令 Cache 和数据 Cache 中的 Tag 信息。

ICache TAG 信息：

31: 13	12: 10	9	8	7: 0
TAG[18:0]	3'b000	LRR	LOCK	VALID[7:0]

- TAG[18:0]: 表示写入的 32 位数据中 TAG 位数据，共 19 比特
- LRR: 表示 ICache 的替换策略标识位
- LOCK: 表示 ICache 保护锁定标识位
- VALID: 表示在写入 32 位数据之后 8 个 WORD 对应的有效位标识

DCache TAG 信息：

31: 13	12: 10	9	8	7: 4	3: 0
TAG[18:0]	3'b000	LRR	LOCK	4'b0000	VALID[3:0]

- TAG[18:0]: 表示写入的 32 位数据中 TAG 位数据，共 19 比特
- LRR: 表示 DCache 的替换策略标识位
- LOCK: 表示 DCache 保护锁定标识位
- VALID: 表示在写入 32 位数据之后 4 个字对应的有效位标识

4.2.5 Cache 寄存器地址映射

Cache 寄存器地址映射见表 4-2-3，相关寄存器详细说明见附录 3.2。

表4-2-3 Cache寄存器地址映射

寄存器名称	地址	复位值 ^注
Cache 控制寄存器	0x80000014	0xfd0000X0
Cache 容错控制寄存器 1	0x80000110	0x20000000
Cache 容错控制寄存器 2	0x80000114	0x00000000
Cache 容错控制寄存器 3	0x80000118	0x00000000

注：“复位值”按照十六进制统一写法“0xfd0000X0”，“0x”表示 16 进制，每 4bit 中只要有 1bit 不定值，即写为“X”，4bit 全部为固定值则写为相应 16 进制数值。

注意

1、Cache 在使能前要首先对 Cache 进行刷新，防止由于之前 Cache 里已经缓存的内容导致指令或数据出错。

2、Cache 刷新过程中，Cache 控制寄存器的 IP 和 DP 位会变为‘1’，此时不可对 Cache 操作；Cache 刷新完成以后，IP 和 DP 位会变为‘0’，此时方可对 Cache 进行操作。

4.3 MMU

4.3.1 概述

BM3823A 中的 MMU 采用页式虚拟存储管理技术，通过虚拟内存到主存映射的 3 级页表查询来完成虚拟地址到物理地址的转换。MMU 从 CPU 得到虚拟地址并将它转换为物理地址后送给主存储器，主存储器再将地址中的数据发送给 CPU。MMU 用 Context Number（0~255）来标识每个进程地址空间。MMU 通过三级页表完成虚实地址转换，第一级和第二级页表会包含一个指向下一级页表的 PTD（页表描述符），第三级的页表项往往是指向物理页面的 PTE（页表项）。第一级或者第二级的页表项也可以是一个 PTE。

PTD 包括页表的物理地址并定义了 Context 表，第一级页表和二级页表中各项的格式。PTD 格式如下：

PTP[31:2]	ET[1:0]

- PTP：页表指针----下一级页表基地址的物理地址；当进行不命中处理的时候，PTP 会出现在物理地址总线的 2 至 31 位。
- ET：用于将 PTD 与 PTE 区分开来。ET 值为 0 时表示对应的虚拟地址的范围目前还没有映射到物理地址，ET 值为 1 时代表 PTD，ET 值为 2 时代表 PTE。

PTE 用来指示该页的物理地址以及该页是否可以访问。PTE 格式如下：

4' b0000	PPN[27:8]	C[7]	M[6]	R[5]	ACC[4:2]	ET[1:0]

- PPN：物理页号，该页 32 位物理地址的高 24 位。当地址变换完成的时候物理页

号出现在物理地址总线的 12 至 31 位。

- C: 可缓存, 如果该位为 1, 那么该页可以被指令或者数据 cache 缓存。
- M: 修改位, 当对页进行了写访问的时候 MMU 会将该位设置为 1。
- R: 访问位, 当页被访问的时候 MMU 会将该位置为 1。
- ACC: 访问许可位, 用来表示是否允许访问该页进行想要的处理。ACC 说明见表 4-3-1。
- ET: 用于将 PTD 与 PTE 区分开来, ET 值为 2 时代表 PTE。

表4-3-1 ACC说明

ACC	Accesses Allowed	
	User access	Supervisor access
0	Read Only	Read Only
1	Read/Write	Read/Write
2	Read/Execute	Read/Execute
3	Read/Write/Execute	Read/Write/Execute
4	Execute Only	Execute Only
5	Read Only	Read/Write
6	No Access	Read/Execute
7	No Access	Read/Write/Execute

MMU 通过执行一次 STA 指令来完成一次刷新操作, 将 PTE/PTD 从 TLB 中删除, 刷新的相应地址由 VFPA (虚拟刷新地址) 确定。MMU 刷新的地址包含以下内容:

VFPA[31:12]	Type[11:8]	保留[7:0]

- VFPA: 虚拟刷新地址。
- Type: 刷新的类型, Type 为 4 时刷新全部 TLB。

4.3.2 MMU 映射操作

MMU 使能时, 通过使用不同的 ASI 映射表示不同的操作, ASI 映射方式见表 4-3-2。

表4-3-2 ASI映射情况说明

ASI	Usage
0x0, 0x1, 0x2, 0x3	强制Cache不命中，若数据被Cache缓存过则更新Cache，若数据不在Cache中但地址可缓存则分配新Cache块进行替换
0x4, 0x7	强制Cache不命中，若数据被Cache缓存过则更新Cache
0x5	刷新指令cache
0x6	刷新数据cache
0x8, 0x9, 0xA, 0xB	正常Cache访问
0xC	诊断指令Cache tags
0xD	诊断指令Cache data
0xE	诊断数据Cache tags
0xF	诊断数据Cache data
0x18	MMU刷新
0x19	MMU寄存器访问
0x1C	MMU旁路

利用 LDA/STA 指令，可以访问不同的地址空间。如果 MMU 使能，通过 ASI 0x18、ASI 0x19、ASI 0x1C 可以进行 MMU 相关操作。

4.3.3 MMU 寄存器地址映射

MMU 有五种内部寄存器。**MMU 控制寄存器**包括了一般 MMU 的控制和状态标志。**Context 寄存器**中存储了当前的进程标志符，**Context 表指针寄存器**存储了存储器中 Context 表的基址指针。如果发生一个 MMU 错误，导致错误发生的地址就存放在**错误地址寄存器**（Fault Address Register）中，而且可以根据**错误状态寄存器**（Fault Status Register）来确定错误的原因。CPU 可以通过外围设备直接访问所有的内部 MMU 寄存器。对于 MMU 寄存器，其地址映射如表 4-3-3 所示。

表4-3-3 MMU寄存器地址映射

MMU寄存器（ASI=0x19时访问）	映射地址
MMU控制寄存器	0x000
Context表指针寄存器	0x100
Context寄存器	0x200
错误状态寄存器	0x300
错误地址寄存器	0x400

MMU 内部寄存器的访问比较特殊，需要使用 ASI 指令（ASI 0x19），寄存器的读取需要用到 LDA 命令，写寄存器需要用到 STA 指令。

例如：

- 读取 MMU 控制寄存器时，程序为：

```
asm("lda [%g0] 0x19, %g1");
```

- 写 MMU 控制寄存器时（假如写入值为 0x83），程序为：

```
asm("set 0x83, %g1; sta %g1, [%g0] 0x19");
```

MMU 相关寄存器的详细说明见附录 3.3。

4.4 中断和陷阱

4.4.1 概述

BM3823A 支持异常处理机制，即陷阱处理。陷阱分为精确陷阱、延迟陷阱和中断。精确陷阱和延迟陷阱由与特定指令相关的硬件引起，在引起陷阱产生的指令执行期间发生，即同步陷阱；中断是由外部事件引起的，中断的产生不与任何特定的指令相关，并在指令执行完成后发生。

陷阱被几个寄存器控制：**处理器状态寄存器 PSR** 的陷阱使能位 *ET* 控制陷阱和外部中断请求，PSR 的处理器中断等级域 *PIL* 指定可响应的外部中断请求级别，**浮点状态寄存器 FSR** 的陷阱使能屏蔽域 *TEM* 控制浮点陷阱请求。

处理器状态寄存器 PSR 中的 *ET* 位必须为 1，陷阱才会被响应。*ET*=1 时，IU 在执行指令期间根据未决的陷阱优先次序来响应陷阱，只有最高优先级的陷阱请求被响应。当在同一时钟周期内发生多个同步陷阱时，具有最高优先级的陷阱被响应，而较低优先级陷阱被忽略；

当有多个未处理的外部中断请求时，较低优先级的中断请求会被保持。

◆ ET 和 PIL 控制

陷阱要正常响应，PSR 中的 ET 位必须为 1。

ET=1，如果发生陷阱，处理器按照陷阱处理程序响应陷阱。ET=0，如果发生陷阱，处理器进入错误模式。

IU 根据陷阱优先级区分陷阱响应次序，只响应最高优先级的陷阱。当处理最高优先级中断时，低优先级中断请求保持在**中断请求寄存器**中。

ET=1，如果发生中断，中断控制器根据**中断屏蔽和优先级寄存器**中的 ILEVEL 和 IMASK 字段比较，送出级别最高的中断，然后处理器比较 PSR 中的 PIL 字段和中断序号，如果中断序号大于 PIL 字段值，则处理器响应该中断。ET=0，所有中断不响应，中断请求保持在**中断请求寄存器**中。

◆ TEM 控制

浮点陷阱发生时，如果相应**浮点状态寄存器 FSR**中的 TEM 字段置 1，则使能浮点陷阱，此时若 ET=1，则处理器按照浮点陷阱处理程序响应浮点陷阱，且引起陷阱的指令的目的寄存器、FSR 的 FCC 和 AEXC 字段保持不变。如果 TEM 字段置 0，则屏蔽浮点陷阱，此时，产生的陷阱会被记录在 FSR 的 AEXC 字段中，且引起陷阱的指令的目的寄存器、FSR 的 FCC 字段被更新。

4.4.2 陷阱类型

陷阱类型符合 SPARC V8 规范，主要陷阱见表 4-4-1。

表4-4-1 陷阱类型

陷阱	TT	描述	优先级
reset	0x00	复位	1
data_store_error	0x1A	数据存储错误	2
instruction_access_exception	0x01	取指错误	3
privileged_instruction	0x03	用户模式下执行特权指令	4
illegal_instruction	0x02	UNIMP或其他未实现的命令	5
fp_disabled	0x04	不使能FPU时执行浮点指令	6
cp_disabled	0x24	不使能协处理器时执行协处理器指令	6

陷阱	TT	描述	优先级
watchpoint_detected	0x0B	指令或数据观察点匹配	7
window_overflow	0x05	SAVE进入无效窗口	8
window_underflow	0x06	RESTORE进入无效窗口	8
register_hardware_error	0x20	检测到不可纠正的寄存器堆错误	9
mem_address_not_aligned	0x07	访问存储器时地址未对齐	10
fp_exception	0x08	FPU异常	11
data_access_exception	0x09	加载/存储指令执行时访问数据错误	13
tag_overflow	0x0A	tagged运算溢出	14
divide_exception	0x2A	被0除	15
trap_instruction	0x80 -0xFF	软件异常指令(TA)	16

- ◆ reset: 复位陷阱由外部复位请求引起。它使得处理器在实际地址 0x0 开始执行。
复位陷阱产生后, 将使得 **PSR** 中的 **ET** 和 **S** 位分别被初始化为 0 和 1。
- ◆ write_error: 数据存储到存储器时发生错误。
- ◆ instruction_access_exception: 无法被纠正的取指错误, 或者取指时 MMU 地址映射失败。
- ◆ privileged_instruction: 当 **PSR** 中的管理位为 0 (即处于用户模式) 而执行特权指令时产生的异常。
- ◆ illegal_instruction: 执行带有未实现操作数的指令, 或者 UNIMP 指令以及会引起处理器非法状态的指令时发生的陷阱。
- ◆ fp-disabled: 当 FPU 无效或者不存在时执行 FPU 指令产生的陷阱。
- ◆ cp-disabled: 当协处理器无效或者不存在时执行协处理器指令产生的陷阱。
- ◆ watchpoint_detected: 指令返回存储器地址或加载/存储数据存储器地址, 这些地址与以前加载的实现相关的观察点寄存器的内容相匹配。
- ◆ window_overflow: SAVE 指令使得当前窗口指针 CWP 指向无效窗口时引起的陷阱。
- ◆ window_underflow: RESTORE 或 RETT 指令使得 CWP 指向无效窗口引起的陷阱。
- ◆ register_hardware_error: 检测到不可纠正的寄存器堆错误, 在读寄存器时发生。
- ◆ mem_address_not_aligned: 加载/存储指令产生与指令地址未正确对齐的存储器地

址或 JMPL, RETT 指令产生字未对齐地址时产生的陷阱。

- ◆ fp_exception: 当陷阱使能屏蔽 TEM 为 1 时, FPU 指令产生 IEEE_754 陷阱, 或 FPU 指令未实现, 或 FPU 指令不完整, 或在 FPU 中存在序列或者硬件错误。
- ◆ data_access_exception: 执行加载数据指令时, 发生数据访问错误, 或者数据访问 MMU 地址映射失败。
- ◆ tag_overflow: 执行标记的算术指令并且算术上溢发生, 或操作数的标记位至少有一位不为 0 时, 产生的陷阱。
- ◆ divide_exception: 执行整数除指令时被 0 除产生的陷阱。
- ◆ trap_instruction: 软件陷阱, 执行 Ticc 指令并且陷阱条件为真时产生的陷阱。

当在同一时钟周期内发生多个同步陷阱时, 具有最高优先级的陷阱被命中, 而较低优先级陷阱被忽略。

在浮点状态寄存器 FSR 中, 有 IEEE_754 浮点数异常相关字段 (Floating-Point Exception Fields), 实现如下:

NVM	OFM	UFM	DZM	NXM
27	26	25	24	23

图4-4-1 FSR中的陷阱允许屏蔽字段TEM

NVA	OFA	UFA	DZA	NXA
9	8	7	6	5

图4-4-2 FSR已发生的异常位字段AEXC

NVC	OFC	UFC	DZC	NXC
4	3	2	1	0

图4-4-3 FSR当前异常位字段CEXC

FSR_invalid(NVC, NVA): 对于将要执行的操作, 操作数为非法操作数。比如, $0 \div 0$ 和 $\infty - \infty$ 运算都是无效运算。该位为 1 表示非法操作数, 为 0 表示合法操作数。

FSR_overflow(OFC, OFA): 在某种数据格式中, 舍入后的结果可能会在数量级上大于正常数值的上限。1 表示上溢, 0 表示未上溢。

FSR_underflow(UFC, UFA): 在某种数据格式中, 舍入后的结果可能会不精确而且会在数量级上小于正常数值的下限。1 表示下溢, 0 表示未下溢。

FSR_division-by-zero(DZC, DZA): $X \div 0$, X 为非正常操作数或者正常操作数时, 对该位置位。注意 $0 \div 0$ 运算不会将 DZC 置位。DZC 在除数为 0 时被置 1, 除数不为 0 时被置 0。

FSR_inexact(NXC, NXA): 舍入后的结果与正确的无限精确结果不同时进行置位。置 1 表示不精确结果, 置 0 表示精确结果。

FSR 的 *TEM* 位, 五种浮点异常使能屏蔽位, *TEM* 字段如图 4-4-1 所示, 如果浮点操作指令产生了一个或多个异常, 并且对应于这一个或多个异常的 *TEM* 位置 1, 则允许产生 IEEE_754 类型的浮点异常陷阱, *TEM* 置 0 可以禁止 IEEE_754 类型的陷阱。

FSR 的 *AEXC* 位, 已发生异常位, 如图 4-4-2 所示, 当浮点陷阱被 *TEM* 屏蔽时, 浮点陷阱就会累计在 *AEXC* 中。一条浮点操作指令执行时, 将 *TEM* 和 *CEXC* 两个字段进行逻辑“与”操作, 如果结果非零, 将会产生 fp_exception 陷阱; 否则, 新的 *CEXC* 字段与 *AEXC* 字段进行逻辑“或”操作后将结果保留到 *AEXC* 字段中。

FSR 的 *CEXC* 位, 当前异常位, 如图 4-4-3 所示, 表示由于执行浮点操作指令, 产生了一个或者多个 IEEE_754 浮点异常。如果没有异常, 对应的位会被清 0。

4.4.3 中断

BM3823A 支持 15 种一级中断及两个优先层次, 一级中断的 10 号中断扩展出 20 个二级中断。中断都是由外部请求引起, 与指令操作无关。中断产生时, **中断强制和未决寄存器**中的相应 *IPEND* 位被置位。中断控制器将当前优先级最高的未屏蔽的中断送给 IU。当 IU 确认中断后, 则**中断强制和未决寄存器**中的相应 *IPEND* 位自动清零。在**中断强制和未决寄存器**中可以通过置位 *IFORCE* 强制中断。这时 IU 确认中断后, **中断强制和未决寄存器**中的相应 *IFORCE* 位清零, 而不是 *IPEND* 位清零。中断控制器的结构框图见图 4-4-4。

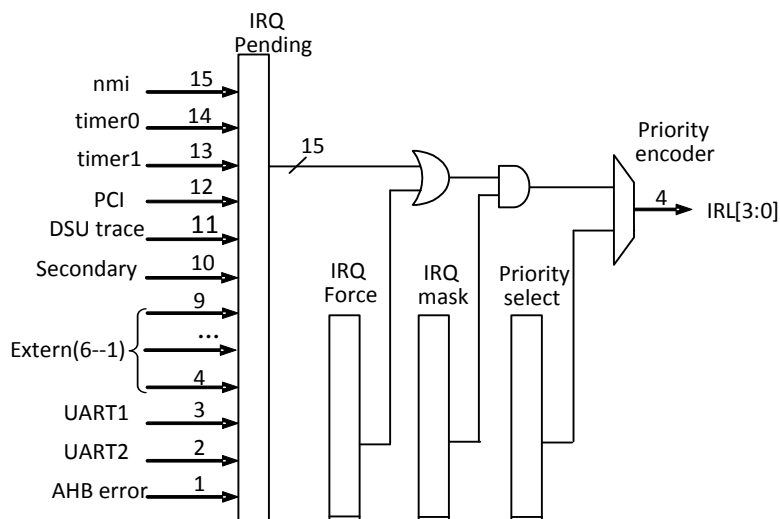


图4-4-4 中断控制器结构框图

4.4.3.1 一级中断

一级中断列表如表 4-4-2 所示。

表4-4-2 一级中断描述

中断号	TT（类型）	中断源
15	0x1F	非屏蔽中断
14	0x1E	timer0
13	0x1D	timer1
12	0x1C	PCI
11	0x1B	DSU trace buffer
10	0x1A	二级中断控制器
9	0x19	外部中断 6
8	0x18	外部中断 5
7	0x17	外部中断 4
6	0x16	外部中断 3
5	0x15	外部中断 2
4	0x14	外部中断 1
3	0x13	UART1
2	0x12	UART2
1	0x11	AHB error

一级中断寄存器映射表如表 4-4-3 所示，寄存器说明详见附录 3.4。

表4-4-3 一级中断寄存器地址映射对照表

寄存器名称	寄存器地址	复位值
中断屏蔽和优先级寄存器	0x80000090	0x80008000
中断强制和未决寄存器	0x80000094	0x00000000
六路中断子优先级寄存器	0x80000098	0x00000000
中断清除寄存器	0x8000009C	0x00000000

4.4.3.2 二级中断

BM3823A 中大多数外围设备都使用了二级中断控制器扩展中断。二级中断产生时，二级中断未决寄存器中的相应 IPEND2 位被置位，同时，一级中断强制和未决寄存器的 10 号中断置位。中断控制器将当前优先级最高的未屏蔽的二级中断送给 IU。当 IU 确认中断后，二级中断未决寄存器的相应位不会自动清零，必须软件清零二级中断请求，并且，软件清零一级中断请求寄存器的 10 号中断位。

二级中断控制器共扩展了 20 个中断，相应的中断状态、请求和屏蔽寄存器只有 20 位有效，其余位保留。二级中断分配见表 4-4-4：

表4-4-4 二级中断分配

中断号	中断源
21	UART4
20	UART3
19	保留
18	保留
17	O2Tbridge1
16	T2Obridge0
15	pwm5
14	pwm4
13	pwm3
12	pwm2
11	pwm1

中断号	中断源
10	pwm0
9	DDR
8	MAC_lpi
7	MAC_sbd
6	DMA
5	Parallel I/O[3]
4	Parallel I/O[2]
3	Parallel I/O[1]
2	Parallel I/O[0]
1	I2c_int1
0	I2c_int0

二级中断寄存器映射表如表 4-4-5 所示，寄存器说明详见附录 3.4。

表4-4-5 二级中断寄存器地址映射对照表

寄存器名称	寄存器地址	复位值
二级中断屏蔽寄存器	0x800000B0	0x00000000
二级中断未决寄存器	0x800000B4	0x00000000
二级中断状态寄存器	0x800000B8	0x00000000
二级中断清除寄存器	0x800000B8	0x00000000

4.4.3.3 外部中断

BM3823A 在同一时刻支持 7 个外部中断：其中 1 个非屏蔽外部中断，中断号是 15；6 个可屏蔽外部中断，中断号是 4，5，6，7，8，9。

外部中断滤波模块对外部中断输入信号进行滤波处理，主要用于去除中断输入信号上的干扰，并且可以通过**外部中断配置寄存器**定义干扰脉冲的宽度，从 0 个到 3 个周期；同时能够灵活配置各路中断的触发方式：低电平、高电平、上升沿、下降沿。

外部中断寄存器映射表如表 4-4-6 所示，寄存器说明详见附录 3.4。

表4-4-6 外部中断寄存器地址映射对照表

寄存器名称	寄存器地址	复位值
外部中断配置寄存器	0x800003C0	0x00000000

4.4.3.4 中断优先级

BM3823A 的中断优先级可以配置，**中断屏蔽和优先级寄存器**的中断级别 *ILEVEL* 字段配置一级中断的优先级，级别 1 的优先级高于级别 0。在同一级别中，序号大的具有较高优先级，一级中断 15 (TT=0x1F) 具有最高优先级，一级中断 1 (TT=0x11) 具有最低优先级。一级中断 15 是不可屏蔽的。

对于 15 个一级中断，优先级通过 *ILEVEL* 和中断序号来决定，优先级次序为 *ILEVEL*>中断序号。

对于 20 个二级中断，一级中断序号固定是 10 号，所以 20 个二级中断的 *ILEVEL* 和一级中断序号是固定的，在二级中断序号中，序号大的优先级高，二级中断 21 具有最高优先级，二级中断 0 具有最低优先级。

对于 7 个外部中断，1 个非屏蔽中断的优先级固定为最高优先级，6 个可屏蔽外部中断的优先级通过 *ILEVEL*、子优先级和中断序号来决定，优先级次序为 *ILEVEL*>子优先级>中断序号。*ILEVEL* 级别 1 的优先级高于级别 0，子优先级可以通过**六路中断子优先级寄存器**调整，可配置为 0 到 5，数值大的优先级大，不能配置为无效优先级 6 和 7，在同一 *ILEVEL* 级别和同一子优先级别中，中断序号大的具有较高优先级。

4.5 DMA 控制器

4.5.1 概述

DMA (Direct Memory Access)，即直接存储器访问，是一种不需要 CPU 参与的传送操作。在 CPU 不占用总线的情况下，总线处于空闲状态，此时 DMA 可以通过硬件电路实现数据的直接快速传输，能够满足高速 I/O 设备的要求，也有利于 CPU 效率的发挥。

DMA 传输的源和目的可以是存储器或外设。BM3823A 的 DMA 控制器提供了 4 个 DMA 通道，每个通道的操作都是相互独立的，4 个通道的优先权通过内部寄存器配置。结构示意图如图 4-5-1 所示。

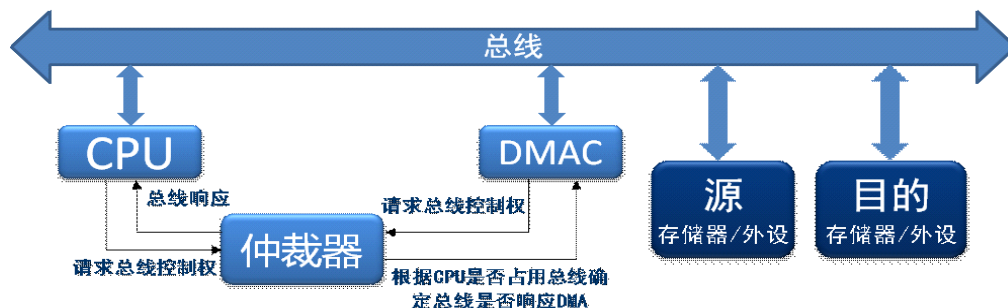


图4-5-1 DMA控制器结构示意图

4.5.2 功能描述

DMA 控制器包含以下功能特点：

- ◆ 支持 single 传输和 burst 传输
- ◆ 支持存储器到存储器、存储器到外设、外设到存储器、外设到外设的 DMA 传输，DMA 控制器与外设之间通过总线桥相互传输数据
- ◆ 流控可以是 DMAC 或外设：当块大小在通道使能前确定时，DMAC 应作为流控，块大小需提前配置到 **CTLn.BLOCK_TS**；当块大小在通道使能前未知时，源外设或目的外设应作为流控
- ◆ 传输的源和目的可以在不同的总线层
- ◆ 源、目的地址可配置
- ◆ 传输大小可配置：单块最大传输 4095 个字
- ◆ 传输宽度可配置：支持字节、半字和字传输
- ◆ 传输方式可配置：支持地址连续传输、地址自动重加载传输和地址来自链接列表传输
- ◆ 地址变化方式可配置：地址递增、递减或不变
- ◆ 4 通道互相独立，通道优先权可配置，默认通道编号大的优先权高
- ◆ 每通道有各自的 FIFO 供源和目的使用，FIFO 深度可配置

4.5.3 DMA 传输配置流程

(1) 启动 DMA 传输前，首先要读**通道使能寄存器 ChEnReg**，选择一条可用的通道，**CH_EN** 位为 1 表明该通道当前已被占用，**CH_EN** 位为 0 表明该通道当前未使用，可以被选用来开启新的 DMA 传输。假设通道 **n** (**n=0~3**) 当前未使用，选用通道 **n** 来执行当前传输；

(2) 选定通道 **n** 之后，写 **DMA 中断清除寄存器** 对应通道 **n** 的 **CLEAR** 位，清中断，然

后可以通过读 **DMA 原始中断状态寄存器** 和 **DMA 中断状态寄存器** 中通道 n 的对应位来确定中断是否已清除；

(3) 写 **DMA 中断屏蔽寄存器** 中通道 n 的对应位来取消中断屏蔽，使能中断；

(4) 根据传输需求配置 **SARn** 和 **DARn** 寄存器，用于设置传输的源起始地址和目的起始地址；

(5) 通过**链表项寄存器 LLPn**、**控制寄存器 CTLn_1** 和 **CTLn_2**、**配置寄存器 CFGn_1** 和 **CFGn_2** 配置传输方式、流控、传输宽度和传输块大小等；

(6) 判断是否为链接列表传输，如果是则配置链接列表及**链表项寄存器 LLPn**，链接列表配置的具体方法见 4.5.4.3 节；如果不是则直接跳到下一步；

(7) 通过配置 **DMA 配置寄存器 DmaCfgReg** 和**通道使能寄存器 ChEnReg**，使能 DMA 传输；

(8) 软件握手控制的 DMA 传输，需通过软件握手寄存器实现，具体配置方式见 4.5.6。

注意 使能 DMA 传输时，必须先使能 DMAC（即先配置 **DmaCfgReg**），再使能选用的 DMA 通道（即再配置 **ChEnReg**）。

配置流程如下图 4-5-2 所示。

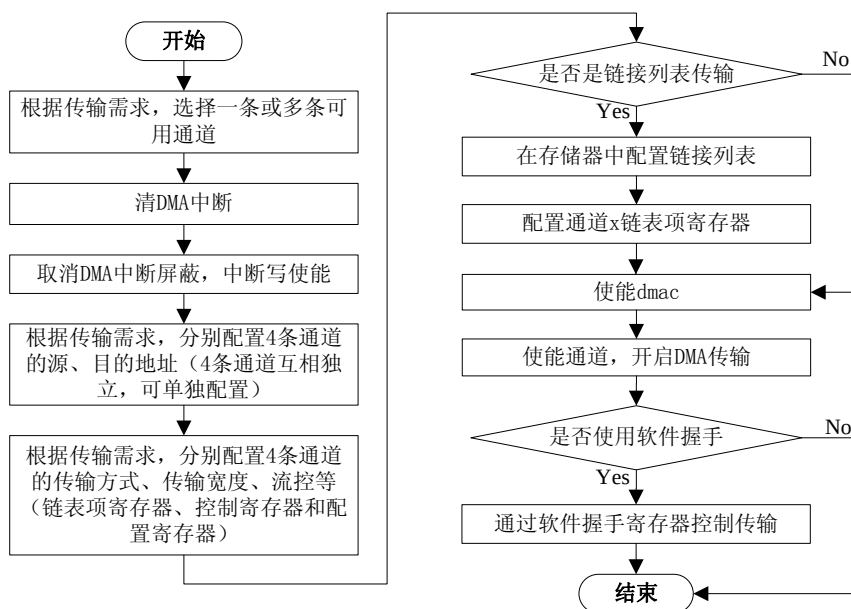


图4-5-2 DMA配置流程

下面分别根据不同的地址更新方式，举例说明 DMA 传输的配置流程。

4.5.3.1 地址连续的 DMA 传输

地址连续的 DMA 传输的配置流程如下：

- (1) 在**通道源地址寄存器 SAR_n**中写源起始地址，在**通道目的地址寄存器 DAR_n**中写目的起始地址；
- (2) 通过将**通道控制寄存器 CTL_n**的 *LLP_SRC_EN*、*LLP_DST_EN* 置为 0，配置源和目的地址均来自于链接列表；
- (3) 将**通道配置寄存器 CFG_{n_1}**的 *RELOAD_DST*、*RELOAD_SRC* 置为 0，配置源和目的地址均不采用自动重加载；
- (4) 不使能链接列表（通道链表项寄存器 **LLP_n** 置为 0）；
- (5) 在 **CTL_{n_1}** 和 **CTL_{n_2}** 中配置 DMA 传输的控制信息，包括传输类型、源和目的为存储器或非存储器外设、流控方式；配置传输特性，包括源和目的传输宽度（如表 4-5-2 所示）、源和目的连接的 master 层、源和目的地址的变化方式（递增、递减或不变）；
- (6) 在**通道配置寄存器 CFG_{n_1}**和 **CFG_{n_2}**中写通道 n 配置信息；
- (7) 配置 **DmaCfgReg** 的 *DMA_EN* 为 1，使能 DMA 控制器；配置 **ChEnReg** 中通道 n 对应的 *CH_EN_WE* 和 *CH_EN* 位，使能 DMA 传输。DMAC 在每次块（single 或 burst）处理结束后应答，完成块传输；传输完成时，硬件发起中断禁能通道 n。

4.5.3.2 地址自动重加载的 DMA 传输

地址自动重加载的 DMA 传输配置流程如下：

- (1) 在**通道源地址寄存器 SAR_n**中写源起始地址，在**通道目的地址寄存器 DAR_n**中写目的起始地址；
- (2) **CFG_{n_1}**寄存器的 *RELOAD_SRC* 和 *RELOAD_DST* 位决定了源和目的地址是否自动重加载，需根据传输需求变换配置值，置为 1 表示相应源或目的地址采用自动重加载模式，为 0 表示不采用自动重加载模式。源采用自动重加载模式时 **CTL_{n_1}**寄存器的 *LLP_SRC_EN* 位需位置为 0；目的采用自动重加载模式时 **CTL_{n_1}**寄存器的 *LLP_DST_EN* 位需位置为 0；
- (3) 若源和目的均不使用链接列表传输，不使能链接列表（通道链表项寄存器 **LLP_n** 置为 0）；
- (4) 在 **CTL_{n_1}** 和 **CTL_{n_2}** 中配置 DMA 传输的控制信息，包括传输类型、源和目的

为存储器或非存储器外设、流控方式；配置传输特性，包括源和目的传输宽度（如表 4-5-2 所示）、源和目的连接的 master 层、源和目的地址的变化方式（递增、递减或不变）；

（5）在**通道配置寄存器 CFGn_1** 和 **CFGn_2** 中写通道 n 配置信息；

（6）配置 **DmaCfgReg** 的 **DMA_EN** 为 1，使能 DMA 控制器；配置 **ChEnReg** 中通道 n 对应的 **CH_EN_WE** 和 **CH_EN** 位，使能 DMA 传输。DMAC 在每次块（single 或 burst）处理结束后应答，硬件置块完成中断，DMAC 会自动重加载 **CTLn_1** 和 **CTLn_2** 寄存器，如果是源自动重加载则同时重加载 **SARn** 寄存器，如果是目的自动重加载则同时重加载 **DARn** 寄存器。当源或目的使用了链接列表传输时，如果 **LLPn 寄存器** 的 **LOC** 位为 0，则硬件置传输完成中断并禁能通道；如果 **LOC** 位不为 0，DMA 传输继续，每次块传输完成时执行自动重加载并检测当前 **LOC** 位的值。为了检测到传输完成，可以响应块完成或传输完成中断，也可以监测 **RawTfr** 寄存器相对应的 **RAW** 位是否被外设置位；传输完成时，硬件发起中断禁能通道 n。

注意

1、如果检测 **RawTfr[n]**，必须保证在通道使能前已经写 **ClearTfr[n]**来清除传输完成中断（n=0~3）。

2、DMA 传输最后一个块完成前，软件必须清除 **CFGn_1** 寄存器中的 **RELOAD_SRC** 位。

4.5.3.3 地址来自链接列表的 DMA 传输

地址来自链接列表的 DMA 传输配置流程与前两种传输方式类似，不同的是如果源或目的地址需要来自链接列表，则需要配置链接列表 LLI。

地址连续的 DMA 传输的配置流程如下：

（1）在**通道源地址寄存器 SARn** 中写源起始地址，在**通道目的地址寄存器 DARn** 中写目的起始地址；

（2）根据源或目的地址更新方式需求配置 **CTLn_1** 寄存器的 **LLP_SRC_EN**、**LLP_DST_EN** 位为 1 或 0，为 1 表示相应源或目的地址来自于链接列表，为 0 表示不是来自链接列表。**CTLn_1** 寄存器其他位及 **CTLn_2** 的配置信息不会被链接列表传输使用，链接列表传输会读取 **LLI.CTLn_1** 和 **LLI.CTLn_2** 当中配置的控制信息。源采用链接列表模式时，**CFGn_1** 寄存器的 **RELOAD_SRC** 位需置为 0；目的采用链接列表模式时，**CFGn_1** 寄存器

的 *RELOAD_DST* 位需置为 0；

(3) 在 *LLPn* 寄存器中写第一个链接地址 *LLPn(0)*；

(4) 完成链接列表配置：

链接列表的内容及配置方式如下：

每个链接列表项包含以下内容：*LLI.SARn*、*LLI.DARn*、*LLI.LLPn*、*LLI.CTLn_1*、*LLI.CTLn_2*、*LLI.SSTATn*、*LLI.DSTATn*，分别表示下一块传输的源地址、目的地址、链接列表基址、控制寄存器配置以及当前块传输的源、目的状态信息存储，如图 4-5-3 所示。

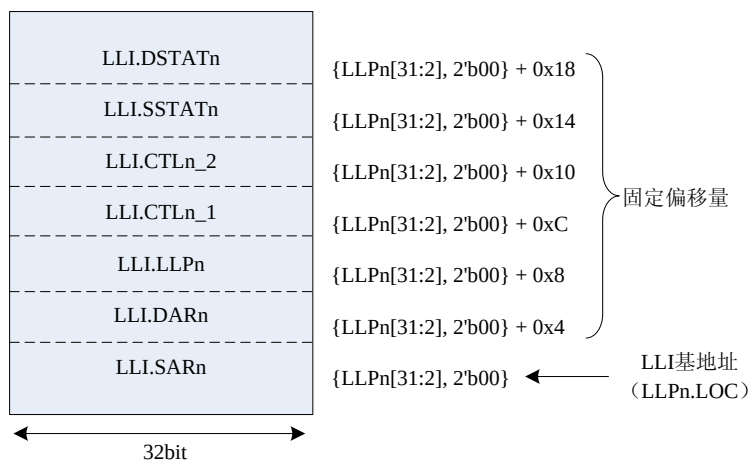


图4-5-3 单个链接列表项结构示意图

链接列表项的数量无限制，两个连接列表项 *LLI(m)*和 *LLI(m+1)*之间通过 *LLPn(m)*连接 (*m* 为自然数)，如图 4-5-4 所示。

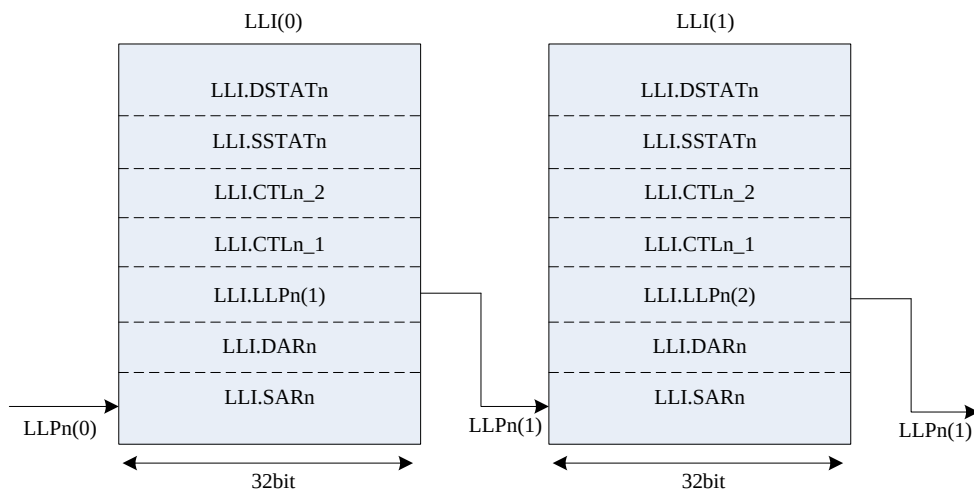


图4-5-4 多个链接列表项结构示意图

配置链接列表时,首先需要规划出足够的存储空间,能够完整存放需要配置的连接列表,任一程序运行过程中不会被占用的空间均可。然后根据传输需求配置源起始地址、目的起始地址、控制信息及下一链接列表基址。

需要特别注意的是:

- ◆ **LLI.SSTATn** 和 **LLI.DSTATn** 寄存器不需要配置,而是在使能源状态写回或目的状态写回时,每次块传输结束后由硬件自动写入;
- ◆ **LLI.SARn**、**LLI.DARn**、**LLI.LLPn**、**LLI.CTLn_1** 和 **LLI.CTLn_2** 寄存器的内容都需要用户自行配置,且无论是否用到都必须配置;
- ◆ **LLI.CTLn_1** 寄存器的配置需要符合链接列表传输的要求;最后一个 LLI 的 **LLI.CTLn_1** 寄存器, **LLP_SRC_EN** 和 **LLP_DST_EN** 都要置为 0。

所有链接列表项的配置需要在 DMA 传输开始之前完成,且链接列表传输开始前,**LLI.CTLn_2** 寄存器的 **DONE** 位需要被清除。启动 DMA 传输时, DMA 控制器首次取 LLI 的地址即为 **LLPn(0)**,通道寄存器 **SARn**、**DARn**、**LLPn**、**CTLn_1** 和 **CTLn_2** 的值会根据 **LLPn(0)** 指向的寄存器 **LLI.SARn**、**LLI.DARn**、**LLI.LLPn**、**LLI.CTLn_1** 和 **LLI.CTLn_2** 自动重配置。

(5) 在 **CTLn_1** 和 **CTLn_2** 中配置 DMA 传输的控制信息,包括传输类型、源和目的为存储器或非存储器外设、流控方式;配置传输特性,包括源和目的传输宽度(如表 4-5-2 所示)、源和目的连接的 master 层、源和目的地址的变化方式(递增、递减或不变);

(6) 在通道配置寄存器 **CFGn_1** 和 **CFGn_2** 中写通道 n 配置信息;

(7) 配置 **DmaCfgReg** 的 **DMA_EN** 为 1,使能 DMA 控制器;配置 **ChEnReg** 中通道 n 对应的 **CH_EN_WE** 和 **CH_EN** 位,使能 DMA 传输。DMAC 在每次块(single 或 burst)处理结束后应答,硬件置块完成中断,DMAC 会自动从当前 **LLPn** 寄存器指向的位置取下一个 LLI,并自动重配置 **SARn**、**DARn**、**LLPn**、**CTLn_1** 和 **CTLn_2** 通道寄存器。如果 **LLPn** 寄存器的 **LOC** 位为 0,则硬件置传输完成中断并禁能通道;如果 **LOC** 位不为 0, DMA 传输继续,每次块传输完成时执行自动重加载并检测当前 **LOC** 位的值。DMA 传输会持续到 DMA 控制器检测到当前的传输块是最后一个块。为了检测到传输完成,可以响应块完成或传输完成中断,也可以监测 **RawTfr** 寄存器相对应的 **RAW** 位是否被外设置位;传输完成时,硬件发起中断禁能通道 n。



注意 如果源和目的中只有一方采用链接列表模式，每次通道寄存器自动重配置时，链接列表中的 **LLI.SARn**、**LLI.DARn**、**LLI.LLPn**、**LLI.CTLn_1** 和 **LLI.CTLn_2** 寄存器内容依然都会被取出，但若源端未采用链接列表模式，则 **LLI.SARn** 的内容不会被写入 **SARn** 寄存器使用，若目的端未采用链接列表模式，则 **LLI.DARn** 的内容不会被写入 **DARn** 寄存器使用。

4.5.4 DMA 中断

对于每个通道，有五种中断源：

- ◆ **IntTfr-DMA** 传输完成中断，向目的外设的 DMA 传输完成时产生该中断。
- ◆ **IntBlock**-块传输完成中断，该中断在 DMA 块传输完成时产生，给目的外设。
- ◆ **IntSrcTran**-源处理完成中断。源端（存储器除外）的握手接口发出的 single/burst 处理请求完成最后一次 AHB 传输时，产生该中断。
- ◆ **IntDstTran**-目的处理完成中断。目的端（目的是存储器除外）的握手接口发出的 single/burst 处理请求完成最后一次 AHB 传输时，产生该中断。
- ◆ **IntErr**-错误中断，当 DMA 传输过程中 HRESP 总线上的 AHB slave 给出一个 ERROR 响应时，产生此中断，同时 DMA 传输取消，该通道被禁能。



注意 如果源或目的是存储器，那么 **IntSrcTran**/**IntDstTran** 中断会被忽略。

当通道使能中断产生时，对中断的处理遵循如下原则：（1）中断事件存储在**原始中断状态寄存器**；（2）**原始中断状态寄存器**的内容被**中断屏蔽寄存器**屏蔽；（3）屏蔽的中断存储在**中断状态寄存器**；（4）**中断状态寄存器**的内容用于驱动 **IntBlock**、**IntDstTran**、**IntErr**、**IntSrcTran**、**IntTfr** 端口信号；（5）向**中断清除寄存器**的相应位写入数据会清除相同时钟周期下的**原始中断状态寄存器**和**中断状态寄存器**中的中断。

五个 Status 寄存器中每一个的内容按位相或，会在**联合状态寄存器**中给对应的每种中断类型生成一位，这就是 **StatusInt**。



注意 为了使中断能够传输通过**原始中断状态寄存器**，**CTLx** 寄存器的 **INT_EN** 位必须设置为 1，并且相关中断必须非屏蔽。

4.5.5 软件握手控制 DMA 传输

握手接口用于在传输时控制 single 或 burst 传输流。非存储器外设可以通过软件握手接口请求 DMA 传输，外设通过握手接口告诉 DMAC 已准备好通过总线传输或接收数据，软件握手接口通过寄存器实现。软件握手寄存器包括：

- ◆ LstDstReg-末位目的处理请求寄存器
- ◆ LstSrcReg-末位源处理请求寄存器
- ◆ ReqDstReg-目的软件处理请求寄存器
- ◆ ReqSrcReg-源软件处理请求寄存器
- ◆ SglReqDstReg-目的 single 处理请求寄存器
- ◆ SglReqSrcReg-源 single 处理请求寄存器

流控是外设或 DMAC 时握手接口的操作是不同的。

当外设不是流控时，DMAC 试图使用尽可能小的总线带宽来有效传输数据。通常，DMAC 会试图发起 burst 传输，并且，在可能的情况下装满或搬空通道 FIFO——假设软件没有限制 burst 传输长度。

当外设是流控时，它控制块的长度并告诉 DMAC 什么时候传输完成——通过说明当前操作是块内最后一次操作实现。若外设是流控时块大小不是 $CTLn.SRC\_MSIZE / CTLn.DST\_MSIZE$ 的整数倍，那么在结尾处外设需要使用 single 传输或预终止的 burst 传输来完成块传输。当外设是流控时，它会告诉 DMAC 发起 single 传输或 burst 传输，DMAC 会使用尽可能大的 burst 长度。

DMAC 可以锁定对 master 总线接口的仲裁，这样一条 DMA 通道就固定给某个 master 口使用了。DMA 还可以发出 AMBA hlock 信号来锁定 AHB 系统仲裁。

当 slave 外设需要 DMAC 发起 DMA 传输时，它首先向 CPU 或中断控制器发出一个中断来发起请求。然后中断服务程序通过软件握手寄存器来初始化和控制一次 DMA 传输。



注意 CFGn 寄存器的 HS_SEL_SRC/HS_SEL_DST 位域必须置 1 来使能软件握手。置 $CFGn.HS\_SEL\_SRC$ 为 1 使能通道 n 的源端软件握手；置 $CFGn.HS\_SEL\_DST$ 为 1 使能通道 n 的目的端软件握手。

在介绍软外设不是流控的件握手操作之前，首先来介绍 Single 传输区域的概念。

4.5.5.1 Single 传输区域

Single 传输区域是 DMAC 使用 single 处理来完成块传输的时段，该时段以外使用 burst 处理。

DMA 块传输存在无法仅通过 burst 处理完成的情况，如块大小不是 burst 处理长度的整数倍时。这时，块传输使用 burst 处理直到块中剩余数据不够一次 burst 操作，然后，DMAC 通过 **SglReqSrcReg/SglReqDstReg** 发出“single”状态标志并通过 single 处理完成后续传输。

外设声明一次 single 状态标志告知 DMAC 还有足够的数据或空间来完成到源外设或给目的外设的 single 传输。

Single 传输区域仅适用不作流控的外设，何时会进入 Single 传输区域取决于什么是流控。

DMAC 是流控：在源 block 剩余要传字节数小于源 burst 传输大小时，源外设进入 Single 传输区域，若块传输大小是源 burst 传输大小的整数倍，则源不会进入 Single 传输区域，源块传输仅使用 burst 处理；在目的 block 剩余要传字节数小于目的 burst 传输大小时，目的外设进入 Single 传输区域，若块传输大小是目的 burst 传输大小的整数倍，则目的不会进入 Single 传输区域，目的块传输仅使用 burst 处理。当外设在 Single 传输区域以外时，DMAC 仅响应 burst 处理请求，无论外设是否知道现在处在 Single 传输区域，它都必须发出 burst 处理请求，否则 DMA 块传输会被挂起。一旦进入 Single 传输区域，DMAC 会通过 single 处理完成块传输。

外设是流控：若源外设是流控，当流控发起块中的最后一次传输并且目的块中剩余要传输的数据小于目的 burst 传输大小时，目的外设进入 Single 传输区域；若目的外设是流控，当流控发起块中的最后一次传输并且源块中剩余要传输的数据小于源 burst 传输大小时，源外设进入 Single 传输区域。

4.5.5.2 外设不是流控时的软件握手配置

当外设不是流控时，**末位处理请求寄存器 LstSrcReg** 和 **LstDstReg** 无效，其值被忽略。

1、外设不在 Single 传输区域的传输配置

给 **ReqSrcReg[n] / ReqDstReg[n]** 写 1 就是发起一次 burst 处理请求，n 是通道编号。然而为了开始一次 burst 处理，软件必须给 **SglReqSrcReg[n] / SglReqDstReg[n]** 写 1。**SglReqSrcReg[n] / SglReqDstReg[n]** 和 **ReqSrcReg[n] / ReqDstReg[n]** 写 1 的配置顺序没有要求，但必须给两组寄存器都配置才能初始化一次 burst 处理。

2、外设在 Single 传输区域的传输配置

必须先给 **SglReqSrcReg[n] / SglReqDstReg[n]** 写 1 来发起一次 single 处理。Single 处理完成时，硬件会清除 **SglReqSrcReg[n] / SglReqDstReg[n]** 和 **ReqSrcReg[n] / ReqDstReg[n]** 寄存器。而在 single 处理发起后，给 **ReqSrcReg[n] / ReqDstReg[n]** 写 1 会被忽略，请求的 burst 处理不会被执行。

软件会检测 **SglReqSrcReg[n] / SglReqDstReg[n]** 和 **ReqSrcReg[n] / ReqDstReg[n]** 寄存器的相关通道位。如果都是 0，那么所有 burst 或 single 处理都已完成。或者，使能和非屏蔽 **IntSrcTran** 或 **IntDstTran** 中断，可以在源或目的处理完成时发起中断。

4.5.5.3 外设是流控时的软件握手配置

当外设是流控时，它会控制块传输的长度、告诉 DMAC 发起 single 传输或 burst 传输，以及块传输何时结束。DMAC 会使用可用的最大 burst 长度。由于流控外设会发起 single 处理，这里不必像 DMAC 作流控时一样设置 Single 传输区域的概念。

给源/目的软件处理请求寄存器 **ReqSrcReg / ReqDstReg** 写 1 初始化一次处理请求；传输类型是 single 或 burst 取决于 **SglReqSrcReg / SglReqDstReg** 的对应位。当 **SglReqSrcReg[n] / SglReqDstReg[n]=1** 时，向 **ReqSrcReg[n] / ReqDstReg[n]** 写 1 表示软件在通道 n 上发起一次 single 处理请求；当 **SglReqSrcReg[n] / SglReqDstReg[n]=0** 时，向 **ReqSrcReg[n] / ReqDstReg[n]** 写 1 表示软件在通道 n 上发起一次 burst 处理请求。

如果 **LstSrcReg[n] / LstDstReg[n]** 寄存器置 1，向 **ReqSrcReg[n] / ReqDstReg[n]** 写 1 表示当前请求是通道 n 上当前块的最后一次传输请求。

SglReqSrcReg / SglReqDstReg 和 **LstSrcReg / LstDstReg** 寄存器必须优先于 **ReqSrcReg / ReqDstReg** 寄存器配置。

Single 或 burst 传输结束后，**ReqSrcReg / ReqDstReg** 寄存器的相应通道位会被清除。软件可以通过该位来检测传输完成。或者，使能和非屏蔽 **IntSrcTran** 或 **IntDstTran** 中断，可以在源或目的处理完成时发起中断。

当外设是流控，且块大小不是 **CTLn.SRC_MIZE** 或 **CTLn.DST_MIZE** 的整数倍时，软件必须通过 single 处理来完成块传输。

4.5.6 DMA 寄存器地址映射

DMA 集群其地址映射如下表 4-5-1 所示，寄存器说明详见附录 3.5。

表4-5-1 DMA寄存器地址映射对照表

寄存器名称	地址	复位值
通道 0 源地址寄存器 SAR0	0xB8000000	0x00000000
通道 0 目的地址寄存器 DAR0	0xB8000008	0x00000000
通道 0 链表项寄存器 LLP0	0xB8000010	0x00000000
通道 0 控制寄存器 1 CTL0_1	0xB8000018	0x00304801
通道 0 控制寄存器 2 CTL0_2	0xB800001C	0x00000002
通道 0 源状态寄存器 SSTAT0	0xB8000020	0x00000000
通道 0 目的状态寄存器 DSTAT0	0xB8000028	0x00000000
通道 0 源状态地址寄存器 SSTATAR0	0xB8000030	0x00000000
通道 0 目的状态地址寄存器 DSTATAR0	0xB8000038	0x00000000
通道 0 配置寄存器 1 CFG0_1	0xB8000040	0x00000e00
通道 0 配置寄存器 2 CFG0_2	0xB8000044	0x00000004
通道 1 源地址寄存器 SAR1	0xB8000058	0x00000000
通道 1 目的地址寄存器 DAR1	0xB8000060	0x00000000
通道 1 链表项寄存器 LLP1	0xB8000068	0x00000000
通道 1 控制寄存器 1 CTL1_1	0xB8000070	0x00304801
通道 1 控制寄存器 2 CTL1_2	0xB8000074	0x00000002
通道 1 源状态寄存器 SSTAT1	0xB8000078	0x00000000
通道 1 目的状态寄存器 DSTAT1	0xB8000080	0x00000000
通道 1 源状态地址寄存器 SSTATAR1	0xB8000088	0x00000000
通道 1 目的状态地址寄存器 DSTATAR1	0xB8000090	0x00000000
通道 1 配置寄存器 1 CFG1_1	0xB8000098	0x00000e20
通道 1 配置寄存器 2 CFG1_2	0xB800009C	0x00000004
通道 2 源地址寄存器 SAR2	0xB80000B0	0x00000000
通道 2 目的地址寄存器 DAR2	0xB80000B8	0x00000000
通道 2 链表项寄存器 LLP2	0xB80000C0	0x00000000
通道 2 控制寄存器 1 CTL2_1	0xB80000C8	0x00304801
通道 2 控制寄存器 2 CTL2_2	0xB80000CC	0x00000002

寄存器名称	地址	复位值
通道 2 源状态寄存器 SSTAT2	0xB80000D0	0x00000000
通道 2 目的状态寄存器 DSTAT2	0xB80000D8	0x00000000
通道 2 源状态地址寄存器 SSTATAR2	0xB80000E0	0x00000000
通道 2 目的状态地址寄存器 DSTATAR2	0xB80000E8	0x00000000
通道 2 配置寄存器 1 CFG2_1	0xB80000F0	0x00000e40
通道 2 配置寄存器 2 CFG2_2	0xB80000F4	0x00000004
通道 3 源地址寄存器 SAR3	0xB8000108	0x00000000
通道 3 目的地址寄存器 DAR3	0xB8000110	0x00000000
通道 3 链表项寄存器 LLP3	0xB8000118	0x00000000
通道 3 控制寄存器 1 CTL3_1	0xB8000120	0x00304801
通道 3 控制寄存器 2 CTL3_2	0xB8000124	0x00000002
通道 3 源状态寄存器 SSTAT3	0xB8000128	0x00000000
通道 3 目的状态寄存器 DSTAT3	0xB8000130	0x00000000
通道 3 源状态地址寄存器 SSTATAR3	0xB8000138	0x00000000
通道 3 目的状态地址寄存器 DSTATAR3	0xB8000140	0x00000000
通道 3 配置寄存器 1 CFG3_1	0xB8000148	0x00000e60
通道 3 配置寄存器 2 CFG3_2	0xB800014C	0x00000004
DMA 原始传输完成中断状态寄存器 RawTfr	0xB80002C0	0x00000000
DMA 原始块完成中断状态寄存器 RawBlock	0xB80002C8	0x00000000
DMA 原始源完成中断状态寄存器 RawSrcTran	0xB80002D0	0x00000000
DMA 原始目的完成中断状态寄存器 RawDstTran	0xB80002D8	0x00000000
DMA 原始错误中断状态寄存器 RawErr	0xB80002E0	0x00000000
DMA 传输完成中断状态寄存器 StatusTfr	0xB80002E8	0x00000000
DMA 块完成中断状态寄存器 StatusBlock	0xB80002F0	0x00000000
DMA 源完成中断状态寄存器 StatusSrcTran	0xB80002F8	0x00000000
DMA 目的完成中断状态寄存器 StatusDstTran	0xB8000300	0x00000000
DMA 错误中断状态寄存器 StatusErr	0xB8000308	0x00000000
DMA 传输完成中断屏蔽寄存器 MaskTfr	0xB8000310	0x00000000
DMA 块完成中断屏蔽寄存器 MaskBlock	0xB8000318	0x00000000
DMA 源完成中断屏蔽寄存器 MaskSrcTran	0xB8000320	0x00000000
DMA 目的完成中断屏蔽寄存器 MaskDstTran	0xB8000328	0x00000000

寄存器名称	地址	复位值
DMA 错误中断屏蔽寄存器 MaskErr	0xB8000330	0x00000000
DMA 传输完成中断清除寄存器 ClearTfr	0xB8000338	0x00000000
DMA 块完成中断清除寄存器 ClearBlock	0xB8000340	0x00000000
DMA 源完成中断清除寄存器 ClearSrcTran	0xB8000348	0x00000000
DMA 目的完成中断清除寄存器 ClearDstTran	0xB8000350	0x00000000
DMA 错误中断清除寄存器 ClearErr	0xB8000358	0x00000000
联合状态寄存器 StatusInt	0xB8000360	0x00000000
源软件处理请求寄存器 ReqSrcReg	0xB8000368	0x00000000
目的软件处理请求寄存器 ReqDstReg	0xB8000370	0x00000000
源 single 处理请求寄存器 SglReqSrcReg	0xB8000378	0x00000000
目的 single 处理请求寄存器 SglReqDstReg	0xB8000380	0x00000000
末位源处理请求寄存器 LstSrcReg	0xB8000388	0x00000000
末位目的处理请求寄存器 LstDstReg	0xB8000390	0x00000000
DMA 配置寄存器 DmaCfgReg	0xB8000398	0x00000000
通道使能寄存器 ChEnReg	0xB80003A0	0x00000000



注意 DMAC 返回 ERROR 响应表示一次违规寄存器访问，包括访问未定义的寄存器地址。因此访问 DMA 寄存器时，请注意不要访问无效地址，包括 0xB8000368~0xB80003FC、0xB80003A8~0xB80003F8。

4.6 Ethernet MAC

4.6.1 概述

BM3823A 的 Ethernet MAC 实现网络接口数据的接收和发送，支持 MII/GMII、TBI 接口类型可配置，支持 10/100/1000Mbps 数据传输率，支持全双工、半双工工作模式，支持 EEE（Energy Efficient Ethernet）能效以太网，支持回环功能。

4.6.2 功能描述

- ◆ 符合 IEEE802.3-2008 协议标准
- ◆ 支持 10/100/1000Mbps 速率选择
- ◆ 支持 MII/GMI 和 TBI 接口
- ◆ 支持全双工/半双工的工作模式

- ◆ 提供 MDIO 接口，用于管理数据传输
- ◆ 支持对输出帧添加 CRC 校验
- ◆ 支持端口全双工模式下的回环操作
- ◆ 支持数据包过滤功能
- ◆ 支持数据帧长可编程，支持标准帧和 Jumbo 帧
- ◆ 支持流量控制
- ◆ 支持 VLAN 以太网功能
- ◆ 支持辅助映射和时间戳功能
- ◆ 支持以太网拥堵调试
- ◆ 支持 EEE 能效以太网功能
- ◆ 发送 FIFO 大小为 16KB，接收 FIFO 大小为 32KB

4.6.3 关键接口引脚说明

Ethernet MAC 管脚包括数据信号和控制信号，关键信号详细说明如下表 4-6-1 所示。

表4-6-1 MAC关键管脚接口描述

名称	类型	描述
phy_intf_sel_i[2:0]	in	PHY 接口选择信号 000: GMII/MII 011: TBI 备注：仅支持以上 2 种配置值，其余配置不支持
phy_txd_o[7:0]	out	PHY 发送数据 GMII: [7:0]位全部用来发送数据，MII: [3:0]位用来发送数据 TBI: 对应发送码组数据的[7:0]位
phy_txen_o	out	PHY 发送数据使能信号，该信号由 MAC 发出 GMII/MII: 高电平表明数据传输有效 TBI: 对应发送码组数据的比特[8]位
phy_txer_o	out	PHY 发送数据错误标识信号，该信号由 MAC 发出 GMII/MII: 高电平表示发送数据错误（10Mbps 速率下，该信号不起作用） TBI: 对应发送码组数据的比特[9]位
phy_rxd_i[7:0]	in	PHY 接收数据

名称	类型	描述
		GMII: [7:0]位全部用来接收数据 MII: [3:0]位用来接收数据 TBI: 对应接收码组数据的[7:0]位
phy_rxdv_i	in	PHY 接收数据有效标识信号, 该信号由 PHY 输出 GMII/MII: 高电平表示接收数据有效 TBI: 对应接收码组数据的的比特[8]位
phy_rxer_i	in	PHY 接收数据错误标识信号, 该信号由 PHY 输出 GMII/MII: 高电平表示接收数据错误 (10Mbps 速率下, 该信号不起作用) TBI: 对应接收码组数据的的比特[9]位
phy_crs_i	in	PHY 载波侦测信号。该信号只在 GMII/MII 模式且为半双工工作时有效。当发送/接收通道有一个忙时, 信号有效; 当发送/接收通道均空闲时, 信号无效。由 PHY 输出, 不同步于参考时钟。
phy_col_i	in	PHY 冲突检测信号。该信号只在 GMII/MII 模式且为半双工工作时有效。当通道上有冲突时, 该信号为高电平, 由 PHY 输出。

4.6.4 系统连接

BM3823A 中仅包含以太网控制器 (Ethernet MAC), 使用时外部需要接 PHY 芯片和网口组成单端网络系统, 如图 4-6-1 所示是一个简单的两个 BM3823A 系统以太网连接示意图。

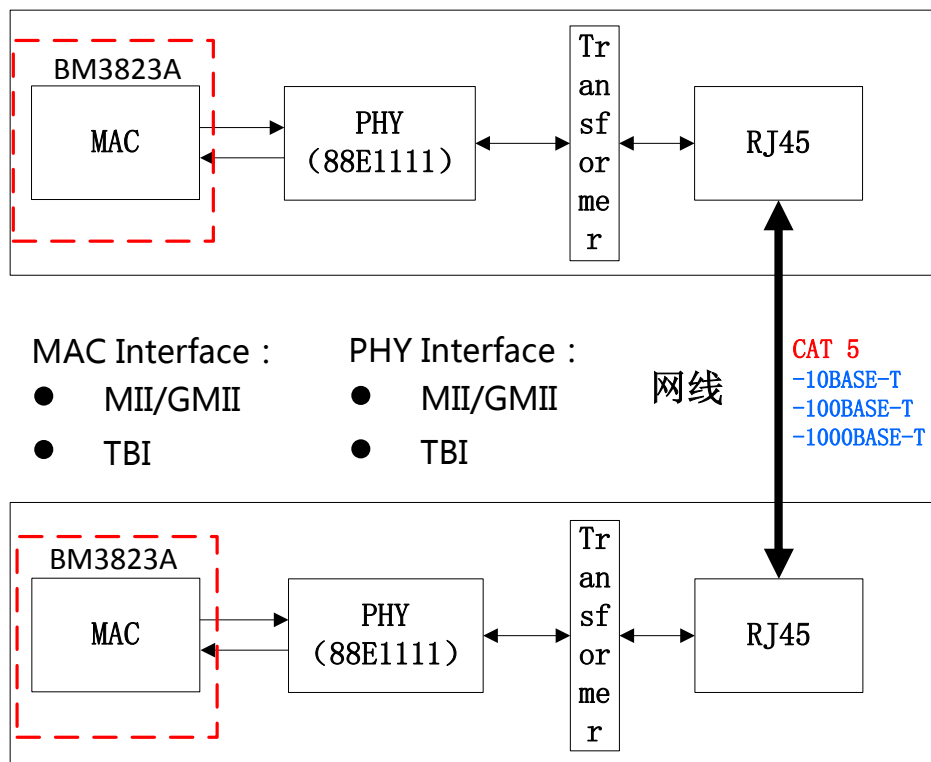


图4-6-1 MAC使用外部连接示意图

图中红色虚框内表示 BM3823A，MAC 为 BM3823A 中的以太网控制器，系统连接时，需要在外边连接 PHY（如：88E1111 芯片）、变压器（Transformer）以及网口（RJ45）即可组成单端以太网组件，通过网线将两个网口连接即可实现以太网数据收发功能。



注意 以太网控制器模块不使用时各管脚的连接方式见附录 9.3。

4.6.5 MAC 编程使用

MAC 在使用前需要进行初始化配置，包括 DMA（Ethernet MAC 内部自带的 DMA）寄存器配置和 GMAC 寄存器配置两部分。

◆ 初始化 DMA 配置流程：

- 1) 软件复位: 复位 GMAC 所有内部寄存器和逻辑。将**总线模式寄存器**的 SWR 位置 1；
- 2) 等待复位完成: 查询**总线模式寄存器**的 SWR 位，该位清 0 后表示复位完成；
- 3) 通过**总线模式寄存器**配置 burst 模式和长度；
- 4) 为发送和接收配置合适的描述符链，确保 DMA 拥有接收描述符（描述符的 bit31），当用 OSF 模式时，至少需要 2 个描述符；

- 5) 在重复使用任何一个描述符之前，确保软件在一个链中创建了三个或更多不同的发送和接收描述符；
- 6) 通过发送和接收描述符基地址，初始化接收和发送描述符列表地址，即分别配置**接收描述符列表地址寄存器**和**发送描述符列表地址寄存器**；
- 7) 通过配置**中断使能寄存器**来使能相关中断；
- 8) 配置**操作模式寄存器**，初始化发送和接收数据的传输方式，通过将 *SR* 位和 *ST* 位置 1 来开始 DMA 的接收和发送。

◆ 初始化 MAC 配置流程：

在 DMA 初始化之后，初始化 MAC。如果在 DMA 设立之前 MAC 已经初始化完成，那么使能 MAC 接收器只能在 DMA 激活之后（即下述最后一步）。否则，接收帧将会溢出接收 FIFO。只有在 TBI 模式时，才操作第一步和第二步，否则操作第三步和第四步。

- 1) 配置**自协商控制寄存器**的 *ANE* 位为 1，使能自动协商功能，也可以根据实际需求配置 *RAN* 和 *ELE* 位（只有 TBI 模式需要本步骤）；
- 2) 检查**自协商状态寄存器**的 *ANC* 位判断自协商是否完成。*ANC* 置高时表示完成自协商，*LS* 位置高表示链路连接成功；
- 3) 配置**GMII 地址寄存器**的 *PA* 位和 *GR* 位建立 MAC 与 PHY 之间的寄存器访问，可以根据实际需求对 PHY 寄存器进行读写访问；
- 4) **GMII 数据寄存器**需要与 **GMII 地址寄存器**联合使用，获得 MAC 与 PHY 的连接数据、操作速度、操作模式等信息；
- 5) 配置**MAC 地址 0 高位寄存器**和**MAC 地址 0 低位寄存器**。如果在 consultant 配置时有超过一个 MAC 地址被使能，那么设定合适的 MAC 地址；
- 6) 通过配置**MAC 帧过滤寄存器**选择适当的数据帧的过滤模式；
- 7) 通过**MAC 配置寄存器**可以进行数据速率、全双工、回环等配置，通过将 *TE* 位和 *RE* 位都置高，使能发送和接收。

4.6.6 描述符说明

在以太网子系统中，DMA 进行数据传输是基于链接的描述符列表的。描述符在系统的存储空间中进行创建即可。BM3823A 处理器中，MAC 描述符分为发送描述符和接收描述符，

每个描述都含有两个 buffer，两个字节计数 buffer，两个地址指针。

发送描述符和接收描述符为 32 字节（8 个字）和 16 字节（4 个字）两种模式进行选择：

- ◆ 当时间戳使能（**时间戳控制寄存器配置**）或校验码转移使能（**MAC 配置寄存器配置**），软件要分配 32 字节存储给每个描述符。此时需软件设置 **MAC 配置寄存器** 的第 7 比特位为高电平，描述符为 8 个字：DES0-DES7；
- ◆ 当是时间戳或校验码转移不使能，则描述符仅需前 4 个字，此时软件设置 **MAC 配置寄存器** 的第 7 比特位为低电平，描述符为 4 个字：DES0-DES3。

发送描述符和接收描述地址映射关系如下表 4-6-2 和表 4-6-3 所示，具体描述见附录 3.6。

注意 由于大小端问题，用户在使用附录 3.6 中的描述符时需要进行大小端转换。

表4-6-2 Ethernet GMAC发送描述符寄存器地址映射

寄存器名称	地址
发送描述符 0 寄存器（TDES0）	BASE_ADDR1 ^注
发送描述符 1 寄存器（TDES1）	BASE_ADDR1 + 0x4
发送描述符 2 寄存器（TDES2）	BASE_ADDR1 + 0x8
发送描述符 3 寄存器（TDES3）	BASE_ADDR1 + 0xC
发送描述符 4 寄存器（TDES4）	BASE_ADDR1 + 0x10
发送描述符 5 寄存器（TDES5）	BASE_ADDR1 + 0x14
发送描述符 6 寄存器（TDES6）	BASE_ADDR1 + 0x18
发送描述符 7 寄存器（TDES7）	BASE_ADDR1 + 0x1C

注：“地址”一栏中的 BASE_ADDR1 是发送描述寄存器的首地址，由用户自行定义，无固定地址，8 个描述符地址连续递增。

表4-6-3 Ethernet GMAC接收描述符寄存器地址映射

寄存器名称	地址
接收描述符 0 寄存器（RDES0）	BASE_ADDR2 ^注
接收描述符 1 寄存器（RDES1）	BASE_ADDR2 + 0x4
接收描述符 2 寄存器（RDES2）	BASE_ADDR2 + 0x8
接收描述符 3 寄存器（RDES3）	BASE_ADDR2 + 0xC
接收描述符 4 寄存器（RDES4）	BASE_ADDR2 + 0x10
接收描述符 5 寄存器（RDES5）	BASE_ADDR2 + 0x14

寄存器名称	地址
接收描述符 6 寄存器 (RDES6)	BASE_ADDR2 + 0x18
接收描述符 7 寄存器 (RDES7)	BASE_ADDR2 + 0x1C

注：“地址”一栏中的 BASE_ADDR2 是接收描述寄存器的首地址，由用户自行定义，无固定地址，8 个描述符地址连续递增。

4.6.7 寄存器地址映射

Ethernet MAC 相关寄存器地址映射见表 4-6-4 和表 4-6-5，详细说明见附录 3.6。

表4-6-4 Ethernet DMA寄存器地址映射

寄存器名称	地址	复位值
总线模式寄存器	0xB0001000	0x00020101
发送查询命令寄存器	0xB0001004	0x00000000
接收查询命令寄存器	0xB0001008	0x00000000
接收描述符列表地址寄存器	0xB000100C	0x00000000
发送描述符列表地址寄存器	0xB0001010	0x00000000
DMA 状态寄存器	0xB0001014	0x00000000
操作模式寄存器	0xB0001014	0x00000000
中断使能寄存器	0xB000101C	0x00000000
丢失帧及缓存溢出计数寄存器	0xB0001020	0x00000000
接收中断看门狗寄存器	0xB0001024	0x00000000
AHB 状态寄存器	0xB000102C	0x00000000
当前主发送描述符寄存器	0xB0001048	0x00000000
当前主接收描述符寄存器	0xB000104C	0x00000000
当前主发送缓存地址寄存器	0xB0001050	0x00000000
当前主接收缓存地址寄存器	0xB0001054	0x00000000

表4-6-5 Ethernet GMAC寄存器地址映射

寄存器名称	地址	复位值
MAC 配置寄存器	0xB0000000	0x00000000
MAC 帧过滤寄存器	0xB0000004	0x00000000

寄存器名称	地址	复位值
GMII 地址寄存器	0xB0000010	0x00000000
GMII 数据寄存器	0xB0000014	0x00000000
流量控制寄存器	0xB0000018	0x00000000
总线模式寄存器	0xB0000024	0x00000000
MAC 地址 0 高位寄存器	0xB0000040	0x8000FFFF
MAC 地址 0 低位寄存器	0xB0000044	0xFFFFFFFF
MAC 地址 1 高位寄存器	0xB0000048	0x0000FFFF
MAC 地址 1 低位寄存器	0xB000004C	0xFFFFFFFF
自协商控制寄存器	0xB00000C0	0x00000001
自协商状态寄存器	0xB00000C4	0x00000108
看门狗超时设定寄存器	0xB00000DC	0x00000000

4.7 DDR2 存储器控制器

4.7.1 概述

DDR2 存储器控制器包含控制器和 PHY 两部分，支持对符合 JEDEC 标准的动态存储器 DDR2 SDRAM 的存取控制。

- 支持最大存储空间容量：1GB（0x4000-0000 – 0x7FFF-FFFF）
- 支持存储数据位宽：32bit
- 支持 DDR2 SDRAM 最高工作频率：266MHz，即 533Mbps 数据率
- 支持最大片选：2 个
- 支持突发长度：4、8
- 支持“纠一检二”的 ECC 功能

4.7.2 功能描述

DDR2 存储器控制器实现了 CPU 等主设备对 DDR2 SDRAM 存储器的数据访问。通过 CPU 配置 DDR2 相关时序参数寄存器，可以支持符合 JEDEC 标准的 DDR2 SDRAM 器件。

4.7.2.1 输入频率配置

DDR2 的 PLL 与主时钟 PLL、GMAC 的 PLL 共用一个参考时钟 F_{REF} 。

DDR_PLL 输出频率计算公式如下：

$$DDR\_pll\_out = \frac{F_{REF} \times 16 \times ddr\_pll\_min}{ddr\_pll\_din}$$

要求 1: 参考时钟 F_{REF} 的频率输入范围: 1MHz~800MHz, 整数模式;

要求 2: $400MHz \leq F_{REF} \times 16 \times ddr\_pll\_min \leq 1600MHz$;

要求 3: DDR_PLL 输出频率要求为不超过 266MHz, 推荐频率为 200MHz 或 266MHz。

输入参考时钟 F_{REF} (clk)、ddr_pll_min (pll_min[2:0]) 和 ddr_pll_din (ddr_pll_din[2:0]) 推荐配置表 4-7-1 如下。

表4-7-1 DDR2频率配置推荐表

F_{REF} (clk) (MHz)	ddr_pll_min (pll_min[2:0])	ddr_pll_din (ddr_pll_din[2:0])	ddr_pll_out (MHz)
25	2	3	266
25	2	4	200

4.7.2.2 ECC 功能

BM3823A 的 DDR2 存储器控制器支持可配置的错误检测与纠正 (Error Checking and Correcting, ECC) 功能。每 32bit 数据对应 4bit 校验码, ECC 能够纠正数据中的 1bit 错误, 能够检测数据中的 2bit 错误。1bit 错误指的是 32bit 数据位和 4bit 校验位总共出现 1bit 错误, 2bit 错误指的是 32bit 数据位和 4bit 校验位总共出现 2bit 错误。

使用 DDR2 的 ECC 功能时, 需要搭配存放 4 位宽 DDR2 SDRAM 校验位的存储体。通过配置 **DDR_CTL_43** 的 **ECC_EN** 为高电平可以使能 ECC 功能。

4.7.3 软硬件配置

DDR2 存储器控制器使用时首先需要在硬件上连接外围 DDR2 SDRAM 存储颗粒, 按照需求调整 DDR2 PLL 时钟频率; 之后通过配置控制器和 PHY 的相关寄存器, 完成软件初始化操作; 最后可以对 DDR2 存储器进行数据访问。

4.7.3.1 硬件连接说明

DDR2 存储器控制器包括了控制器和 PHY 两个部分, 使用时外围需要通过相关管脚连接符合标准的 DDR2 SDRAM 存储器。DDR2 管脚包括地址信号、数据信号和控制信号, 关键信号详细说明如下表 4-7-2 所示。

表4-7-2 DDR2关键管脚接口描述

名称	类型	描述
pad_mem_address[14:0]	out	DDR2 行列地址信号, 共 15 位, 由于 DDR2 空间为 1GB, 根据行列地址计算, 地址信号最高位[14]位用不到
pad_mem_bank[2:0]	out	DDR2 BANK 地址信号, 共 3 位
pad_mem_data[31:0]	in/out	DDR2 数据信号, 32 位宽
pad_mem_dm[3:0]	out	DDR2 双向数据屏蔽信号
pad_mem_dqs_m[3:0]	in/out	DDR2 数据选通差分对信号, 共 4 组, 每一组对应数据位 8 比特, [0]对应数据的[7:0], 依此类推
pad_mem_dqs_p[3:0]	in/out	
pad_mem_ecc[3:0]	in/out	DDR2 数据校验位, 4 位宽
pad_mem_ecc_dm	out	DDR2 ECC 校验数据屏蔽
pad_mem_ecc_dqs_m	in/out	DDR2 ECC 校验数据选通差分对信号
pad_mem_ecc_dqs_p	in/out	
pad_mem_cke	out	DDR2 时钟使能信号
pad_mem_clk[1:0]	out	DDR2 差分时钟信号, 两组时钟分别对应 DDR2 的两个片选外设
pad_mem_clk_n[1:0]	out	
pad_mem_cs_n[1:0]	out	DDR2 片选信号, 共两个, 每个片选可接 1 个存储颗粒, 低电平有效
pad_mem_cas_n	out	DDR2 列地址片选, 低电平有效
pad_mem_ras_n	out	DDR2 行地址片选, 低电平有效
pad_mem_we_n	out	DDR2 写使能信号, 低电平有效
pad_mem_odt[1:0]	out	DDR2 内部核心终结, 该信号共两位, 分别对应 DDR2 的两个片选外设。 DDR2 内部集成了终结电阻, 当 DDR2 工作时系统会吧终结电阻器屏蔽, 当暂时不工作时则会打开终结电阻器以减少信号的反射。可以通过 ODT 同时管理所有引脚的信号终结。

注意

- 1、DDR2 的 ODT 功能是否使能, 可以通过寄存器进行配置;
- 2、ODT 内部终结电阻阻值大小有 50Ω、75Ω 多种选择, 可以通过寄存器进行配置;
- 3、DDR2 使用时如果使能了 ODT 功能, 则外部硬件连接上仅需要将 ODT 输出连接到

存储颗粒即可；如果不使能 ODT 功能，在外部硬件连接时，需要在 DDR2 管脚间手动加入终结电阻，减少信号反射；

4、DDR2 存储器控制器模块不使用时各管脚的连接方式见附录 9.2。

4.7.3.2 DDR2 PCB 设计参考约束

由于 DDR2 的频率相对较高，为了更好的使用 DDR2，在进行原理图和 PCB 设计时需要遵守一定的设计规则，包括差分对走线等长、增加匹配电阻等，设计规则共 7 条，供用户设计时参考。

1) 一般信号线 50 欧姆阻抗匹配，差分信号对 100 欧姆阻抗匹配；

2) 差分时钟信号 pad_mem_clk 和 pad_mem_clk_n 等长控制在+5mil 内；

3) pad_mem_address[13:0]、pad_mem_bank[2:0]、pad_mem_cs_n、pad_mem_cas_n、pad_mem_ras_n、pad_mem_cke、pad_mem_odt 和 pad_mem_we_n，均以 pad_mem_clk 时钟线长度为基准+/-200mil；

4) 信号 pad_mem_dqs_p[3:0]、pad_mem_ecc_dqs_p 以 pad_mem_clk 时钟线长度为基准+/-200mil（原理图中以*_dqs_p 和*_dqs_m 结尾的为一对差分信号，如 pad_mem_dqs_p[0] 和 pad_mem_dqs_m[0]）；

5) 每组数据线如：pad_mem_data[7:0]为一组，对应 pad_mem_dm[0]、差分线 pad_mem_dqs_p[0] 和 pad_mem_dqs_m[0]，pad_mem_data[7:0] 和 pad_mem_dm[0] 均以 pad_mem_dqs_p[0]线长为基准+/-25mil；

同理：

第二组：pad_mem_data[15:8]为一组，对应 pad_mem_dm[1]、差分线 pad_mem_dqs_p[1] 和 pad_mem_dqs_m[1]，pad_mem_data[15:8]和 pad_mem_dm[1]均以 pad_mem_dqs_p[1]线长为基准+/-25mil；

第三组：pad_mem_data[23:16]为一组，对应 pad_mem_dm[2]、差分线 pad_mem_dqs_p[2] 和 pad_mem_dqs_m[2]，pad_mem_data[23:16]和 pad_mem_dm[2]均以 pad_mem_dqs_p[2]线长为基准+/-25mil；

第四组：pad_mem_data[31:24]为一组，对应 pad_mem_dm[3]、差分线 pad_mem_dqs_p[3] 和 pad_mem_dqs_m[3]，pad_mem_data[31:24]和 pad_mem_dm[3]均以 pad_mem_dqs_p[3]线长为基准+/-25mil；

第五组: pad_mem_ecc[3:0]为一组, 对应 pad_mem_ecc_dm、差分线 pad_mem_ecc_dqs_p 和 pad_mem_ecc_dqs_m, pad_mem_ecc[3:0]和 pad_mem_ecc_dm 均以 pad_mem_ecc_dqs_p 线长为基准 $\pm 25\text{mil}$;

6) 所有的差分信号线要求等长;

7) 数据线和时钟线不要太近, pad_mem_clk 走线不要太长, 时钟线不要穿过数据线。

4.7.3.3 软件初始化

DDR2 存储器控制器必须经过软件初始化后, 才可以正常访问。初始化操作从软件设置 **DDR_CTL_00** 寄存器 (地址为 0x88000000) 的 **START** 位为 1 时开始, 在设置 **START** 为 1 之前, 必须将所有寄存器配置为正确的值。

初始化步骤如下:

1) 软件向所有控制器和 **PHY** 相关的寄存器写入正确的配置值, 但是需保持 **DDR_CTL_00** 寄存器 (地址为 0x88000000) 的 **START** 位为 0;

2) 配置所有相关寄存器, 包括控制器寄存器 **DDR_CTRL_*** (共 143 个) 和 **PHY** 寄存器 **DDR_PHY_*** (共 88 个);

3) 完毕后, 软件设置 **DDR_CTL_00** 寄存器 (地址为 0x88000000) 的 **START** 位为 1, 开始 DDR2 的初始化操作;

4) 软件读取 **DDR_CTL_66** 寄存器 (地址为 0x88000108) 的值, 当 **INT_STATUS[9]** 位置为 1 时, 表明 DDR2 初始化完成;

5) 可以开始 DDR2 正常操作。

注意

1、DDR2 相关寄存器包括控制器和 **PHY** 共 231 个, 配置时需要根据不同的 DDR2 SDRAM 颗粒手册进行合理配置, 目前常用颗粒型号较少, 时序要求类似, 为了方便用户使用, 产品方提供通用用户初始化配置文件, 包括全部寄存器配置和初始化操作, 用户仅需在提供文件基础上, 根据实际需求进行微小调整即可;

2、用户需要根据实际板级情况, 通过调整输入、输出阻抗值调整 DDR2 接口驱动能力, 满足使用需求, 具体可以配置 **PHY** 寄存器 **DDR_PHY_82~DDR_PHY_87** 的 **OCD_PU**、**OCD_PD**、**TSEL** 和 **DS** 位阈;

3、推荐配置 OCD_PU 和 OCD_PD 值为“0111”或“1000”；

4、TSEL 表示输入阻抗，DS 表示输出阻抗，参考附件 3.7.4 列表信息进行阻抗配置，阻抗值越小，管脚驱动能力越强，但是功耗会有变大，用户使用时根据实际情况，在满足驱动能力的情况下尽量配置大阻抗，降低功耗。

4.7.4 时序

4.7.4.1 上电及初始化

DDR2 存储器控制器的上电初始化如下图 4-7-1 所示：

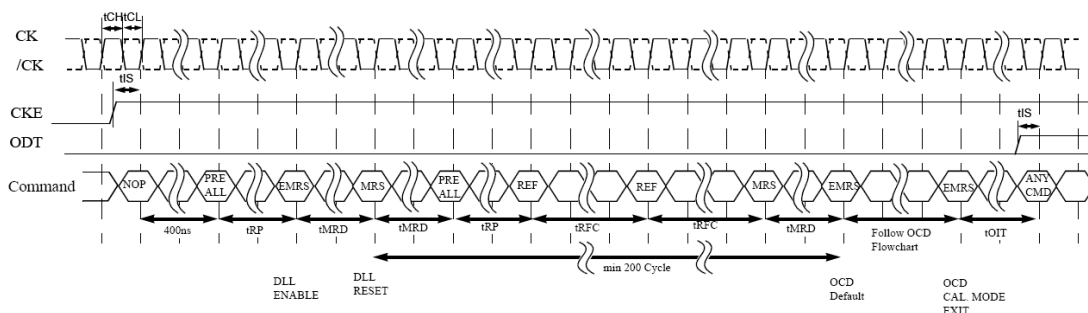


图4-7-1 DDR2存储器控制器初始化时序图

- 1) 时钟开始，并保持稳定；
- 2) 电源和时钟稳定至少 $200\mu s$ ，发送 NOP 或 Deselect 命令，CKE 拉高；
- 3) 等待至少 400ns，发送预充（Precharge）命令，400ns 期间仍发送 NOP 或 Deselect 命令；
- 4) 发送 EMRS 命令给 EMR2 (BA2=0, BA1=1, BA0=0)；
- 5) 发送 EMRS 命令给 EMR3 (BA2=0, BA1=1, BA0=1)；
- 6) 发送 EMRS 命令使能 DLL (BA2=0, BA1=0, BA0=1, A15-A13=000, A9-A7=000, A0=0)；
- 7) 发送 MRS 命令复位 DLL (BA2=0, BA1=0, BA0=0, A15-A13=000, A8=1)；
- 8) 发送 precharge all 命令；
- 9) 发送两个或以上自动刷新（auto-refresh）命令；
- 10) 发送 MRS 命令初始化设备操作 (A8=0)；
- 11) 等待至少 200 个时钟周期，执行 OCD 校准；
- 12) DDR2 SDRAM 准备好，可以正常操作。

4.7.4.2 读时序

DDR2 SDRAM 读操作时序如图 4-7-2 所示，CMD (Command, CMD) 由 CS#、RAS#、CAS#、WE#组成。对于读操作，CS#=0、RAS#=1、CAS#=0、WE#=1。

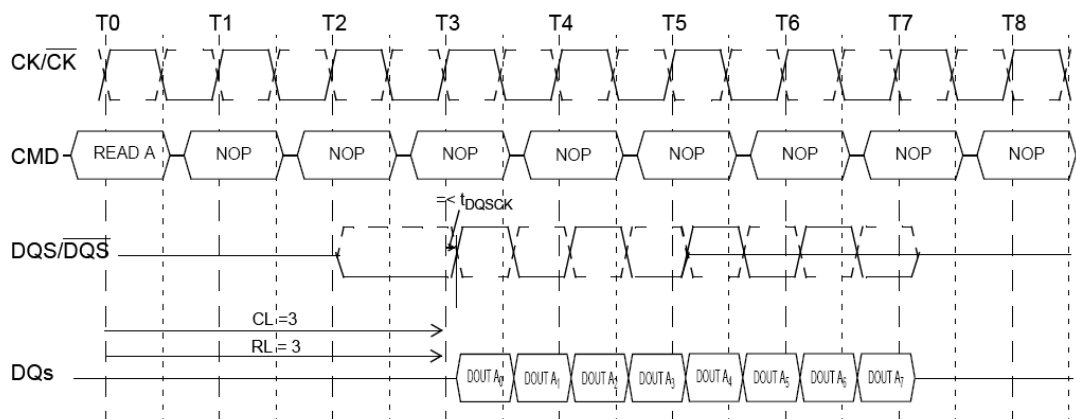


图4-7-2 DDR2 SDRAM读操作时序图 (CL=3, RL=3, Burst Length=8)

4.7.4.3 写时序

DDR2 SDRAM 写操作时序如图 4-7-3 所示，CMD (Command, CMD) 由 CS#、RAS#、CAS#、WE#组成。对于写操作，CS#=0、RAS#=1、CAS#=0、WE#=0。

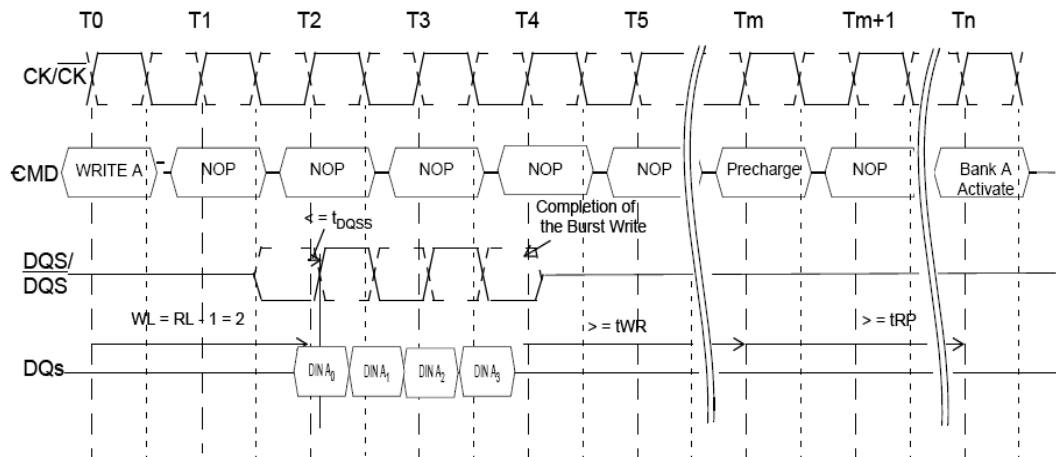


图4-7-3 DDR2 SDRAM写操作时序图 (CL=3, WL=3, Burst Length=4)

另外，与读操作不同的是，写操作需要 DM 信号来标识写操作的掩码，即需要写入的字节数。DM 与 DQ 信号同步关系如图 4-7-4 所示。

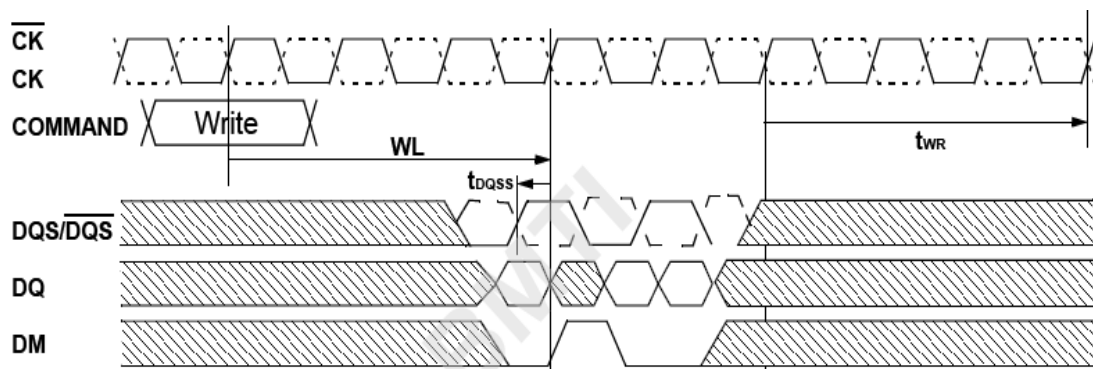


图4-7-4 DM与DQ信号的同步关系图

4.7.5 DDR2 控制器的寄存器地址映射

DDR2 控制器的寄存器地址映射如表 4-7-3 所示，相关寄存器说明详见附录 3.7.1。

其中：

模式寄存器配置相关寄存器：DDR_CTL_32-DDR_CTL_38；

BIST 测试相关寄存器：DDR_CTL_39-DDR_CTL_42、DDR_CTL_72-DDR_CTL_77；

ECC 功能相关寄存器：DDR_CTL_43-DDR_CTL_53；

ZQ 校准相关寄存器：DDR_CTL_54 - DDR_CTL_58；

中断相关寄存器：DDR_CTL_66 - DDR_CTL_68；

超地址范围检查相关寄存器：DDR_CTL_69 - DDR_CTL_71；

表4-7-3 DDR2控制器相关寄存器地址映射对照表

寄存器名称	地址
DDR_CTL_00	0x88000000
DDR_CTL_01	0x88000004
DDR_CTL_02	0x88000008
DDR_CTL_03	0x8800000c
DDR_CTL_04	0x88000010
DDR_CTL_05	0x88000014
DDR_CTL_06	0x88000018
DDR_CTL_07	0x8800001c
DDR_CTL_08	0x88000020
DDR_CTL_09	0x88000024

寄存器名称	地址
DDR_CTL_10	0x88000028
DDR_CTL_11	0x8800002c
DDR_CTL_12	0x88000030
DDR_CTL_13	0x88000034
DDR_CTL_14	0x88000038
DDR_CTL_15	0x8800003c
DDR_CTL_16	0x88000040
DDR_CTL_17	0x88000044
DDR_CTL_18	0x88000048
DDR_CTL_19	0x8800004c
DDR_CTL_20	0x88000050
DDR_CTL_21	0x88000054
DDR_CTL_22	0x88000058
DDR_CTL_23	0x8800005c
DDR_CTL_24	0x88000060
DDR_CTL_25	0x88000064
DDR_CTL_26	0x88000068
DDR_CTL_27	0x8800006c
DDR_CTL_28	0x88000070
DDR_CTL_29	0x88000074
DDR_CTL_30	0x88000078
DDR_CTL_31	0x8800007c
DDR_CTL_32	0x88000080
DDR_CTL_33	0x88000084
DDR_CTL_34	0x88000088
DDR_CTL_35	0x8800008c
DDR_CTL_36	0x88000090
DDR_CTL_37	0x88000094
DDR_CTL_38	0x88000098
DDR_CTL_39	0x8800009c
DDR_CTL_40	0x880000a0

寄存器名称	地址
DDR_CTL_41	0x880000a4
DDR_CTL_42	0x880000a8
DDR_CTL_43	0x880000ac
DDR_CTL_44	0x880000b0
DDR_CTL_45	0x880000b4
DDR_CTL_46	0x880000b8
DDR_CTL_47	0x880000bc
DDR_CTL_48	0x880000c0
DDR_CTL_49	0x880000c4
DDR_CTL_50	0x880000c8
DDR_CTL_51	0x880000cc
DDR_CTL_52	0x880000d0
DDR_CTL_53	0x880000d4
DDR_CTL_54	0x880000d8
DDR_CTL_55	0x880000dc
DDR_CTL_56	0x880000e0
DDR_CTL_57	0x880000e4
DDR_CTL_58	0x880000e8
DDR_CTL_59	0x880000ec
DDR_CTL_60	0x880000f0
DDR_CTL_61	0x880000f4
DDR_CTL_62	0x880000f8
DDR_CTL_63	0x880000fc
DDR_CTL_64	0x88000100
DDR_CTL_65	0x88000104
DDR_CTL_66	0x88000108
DDR_CTL_67	0x8800010c
DDR_CTL_68	0x88000110
DDR_CTL_69	0x88000114
DDR_CTL_70	0x88000118
DDR_CTL_71	0x8800011c

寄存器名称	地址
DDR_CTL_72	0x88000120
DDR_CTL_73	0x88000124
DDR_CTL_74	0x88000128
DDR_CTL_75	0x8800012c
DDR_CTL_76	0x88000130
DDR_CTL_77	0x88000134
DDR_CTL_78	0x88000138
DDR_CTL_79	0x8800013c
DDR_CTL_80	0x88000140
DDR_CTL_81	0x88000144
DDR_CTL_82	0x88000148
DDR_CTL_83	0x8800014c
DDR_CTL_84	0x88000150
DDR_CTL_85	0x88000154
DDR_CTL_86	0x88000158
DDR_CTL_87	0x8800015c
DDR_CTL_88	0x88000160
DDR_CTL_89	0x88000164
DDR_CTL_90	0x88000168
DDR_CTL_91	0x8800016c
DDR_CTL_92	0x88000170
DDR_CTL_93	0x88000174
DDR_CTL_94	0x88000178
DDR_CTL_95	0x8800017c
DDR_CTL_96	0x88000180
DDR_CTL_97	0x88000184
DDR_CTL_98	0x88000188
DDR_CTL_99	0x8800018c
DDR_CTL_100	0x88000190
DDR_CTL_101	0x88000194
DDR_CTL_102	0x88000198

寄存器名称	地址
DDR_CTL_103	0x8800019c
DDR_CTL_104	0x880001a0
DDR_CTL_105	0x880001a4
DDR_CTL_106	0x880001a8
DDR_CTL_107	0x880001ac
DDR_CTL_108	0x880001b0
DDR_CTL_109	0x880001b4
DDR_CTL_110	0x880001b8
DDR_CTL_111	0x880001bc
DDR_CTL_112	0x880001c0
DDR_CTL_113	0x880001c4
DDR_CTL_114	0x880001c8
DDR_CTL_115	0x880001cc
DDR_CTL_116	0x880001d0
DDR_CTL_117	0x880001d4
DDR_CTL_118	0x880001d8
DDR_CTL_119	0x880001dc
DDR_CTL_120	0x880001e0
DDR_CTL_121	0x880001e4
DDR_CTL_122	0x880001e8
DDR_CTL_123	0x880001ec
DDR_CTL_124	0x880001f0
DDR_CTL_125	0x880001f4
DDR_CTL_126	0x880001f8
DDR_CTL_127	0x880001fc
DDR_CTL_128	0x88000200
DDR_CTL_129	0x88000204
DDR_CTL_130	0x88000208
DDR_CTL_131	0x8800020c
DDR_CTL_132	0x88000210
DDR_CTL_133	0x88000214

寄存器名称	地址
DDR_CTL_134	0x88000218
DDR_CTL_135	0x8800021c
DDR_CTL_136	0x88000220
DDR_CTL_137	0x88000224
DDR_CTL_138	0x88000228
DDR_CTL_139	0x8800022c
DDR_CTL_140	0x88000230
DDR_CTL_141	0x88000234
DDR_CTL_142	0x88000238

4.7.6 DDR2 PHY 的寄存器地址映射

BM3823A 的 DDR2 PHY 包含 4 个数据块(DATA SLICE)和 1 个 ECC 块(ECC SLICE), 共有 88 个配置寄存器。具体分配如下:

- **DDR_PHY_00 — DDR_PHY_15**, 对应 DATA SLICE 0;
- **DDR_PHY_16 — DDR_PHY_31**, 对应 DATA SLICE 1;
- **DDR_PHY_32 — DDR_PHY_47**, 对应 DATA SLICE 2;
- **DDR_PHY_48 — DDR_PHY_63**, 对应 DATA SLICE 3;
- **DDR_PHY_64 — DDR_PHY_79**, 对应 ECC SLICE;
- **DDR_PHY_80 — DDR_PHY_87**, 全局控制所有 SLICE。

下面以 DATA SLICE 0 对应的 16 个寄存器为例, 对 PHY 相关配置寄存器进行简单介绍。

DDR_PHY_00: DQ 时序配置;

DDR_PHY_01: DQS 时序配置;

DDR_PHY_02: LOOPBACK 自测试配置;

DDR_PHY_03: 读控制;

DDR_PHY_04: 主 DLL 控制;

DDR_PHY_05: 从 DLL 控制;

DDR_PHY_06: 输入使能 IE (Input Enable) 时序配置;

DDR_PHY_07 — DDR_PHY_10: 只读寄存器, 反映 PHY 和 DLL 的工作状态;

DDR_PHY_11 — DDR_PHY_15: 保留寄存器。

DDR2 PHY 的寄存器地址映射如表 4-7-4 所示，寄存器说明详见附录 3.7.2。

表4-7-4 DDR2 PHY寄存器地址映射对照表

寄存器名称	地址
DDR_PHY_00	0x88000400
DDR_PHY_01	0x88000404
DDR_PHY_02	0x88000408
DDR_PHY_03	0x8800040C
DDR_PHY_04	0x88000410
DDR_PHY_05	0x88000414
DDR_PHY_06	0x88000418
DDR_PHY_07	0x8800041C
DDR_PHY_08	0x88000420
DDR_PHY_09	0x88000424
DDR_PHY_10	0x88000428
DDR_PHY_16	0x88000440
DDR_PHY_17	0x88000444
DDR_PHY_18	0x88000448
DDR_PHY_19	0x8800044C
DDR_PHY_20	0x88000450
DDR_PHY_21	0x88000454
DDR_PHY_22	0x88000458
DDR_PHY_23	0x8800045C
DDR_PHY_24	0x88000460
DDR_PHY_25	0x88000464
DDR_PHY_26	0x88000468
DDR_PHY_32	0x88000480
DDR_PHY_33	0x88000484
DDR_PHY_34	0x88000488
DDR_PHY_35	0x8800048C
DDR_PHY_36	0x88000490
DDR_PHY_37	0x88000494
DDR_PHY_38	0x88000498

寄存器名称	地址
DDR_PHY_39	0x8800049C
DDR_PHY_40	0x880004A0
DDR_PHY_41	0x880004A4
DDR_PHY_42	0x880004A8
DDR_PHY_48	0x880004C0
DDR_PHY_49	0x880004C4
DDR_PHY_50	0x880004C8
DDR_PHY_51	0x880004CC
DDR_PHY_52	0x880004D0
DDR_PHY_53	0x880004D4
DDR_PHY_54	0x880004D8
DDR_PHY_55	0x880004DC
DDR_PHY_56	0x880004E0
DDR_PHY_57	0x880004E4
DDR_PHY_58	0x880004E8
DDR_PHY_64	0x88000500
DDR_PHY_65	0x88000504
DDR_PHY_66	0x88000508
DDR_PHY_67	0x8800050C
DDR_PHY_68	0x88000510
DDR_PHY_69	0x88000514
DDR_PHY_70	0x88000518
DDR_PHY_71	0x8800051C
DDR_PHY_72	0x88000520
DDR_PHY_73	0x88000524
DDR_PHY_74	0x88000528
DDR_PHY_80	0x88000540
DDR_PHY_81	0x88000544
DDR_PHY_82	0x88000548
DDR_PHY_83	0x8800054C
DDR_PHY_84	0x88000550
DDR_PHY_85	0x88000554
DDR_PHY_86	0x88000558

寄存器名称	地址
DDR_PHY_87	0x8800055C



注意 DDR2 的 OCD_PU、OCD_PD、TSEL 及 DS 等具体配置值说明见附录 3.7.3

和 3.7.4，DDR2 各管脚的相关配置项在 **DDR_PHY_82-DDR_PHY_87** 中。

4.8 PCI 总线控制器

4.8.1 概述

BM3823A 的 PCI 接口为 32 位 AHB to PCI 总线桥, 实现了 AHB2.0 总线和 PCI2.3 33MHz PCI 总线之间的协议转换。BM3823A 的 PCI 接口可配置成主桥形式 (Host Bridge) 或从桥形式 (Guest Bridge)，对应 PCI 总线上的 Host 设备和 Guest 设备。此模块实现了标准 32 位 PCI 接口的所有必选信号和可选信号中的全部 PCI 中断信号。

PCI 总线上的 Host 设备和 Guest 设备如图 4-8-1。图 4-8-1 左侧，BM3823A 处于 Host 模式，配置总线上其他 PCI 设备，并为其他设备提供总线仲裁。图 4-8-1 右侧，BM3823A 处于 Guest 模式，被 PCI Host 设备配置，在总线 Host 的管理下进行总线通信。

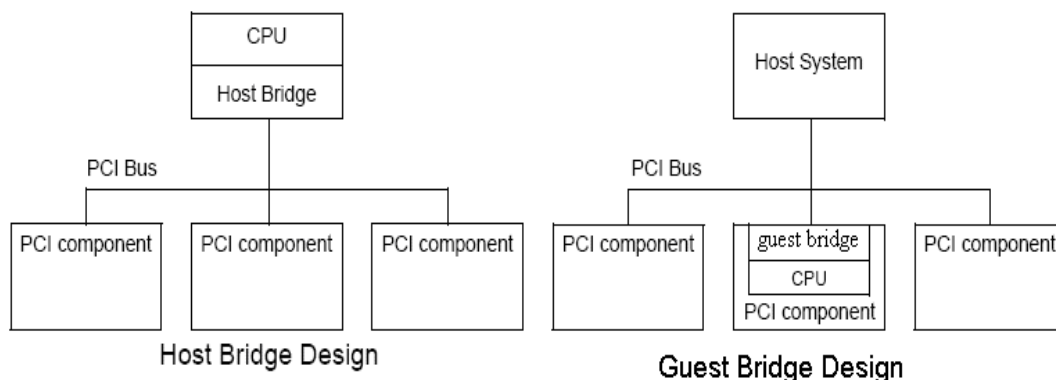


图4-8-1 PCI总线设备

4.8.2 AHB-PCI 模块结构

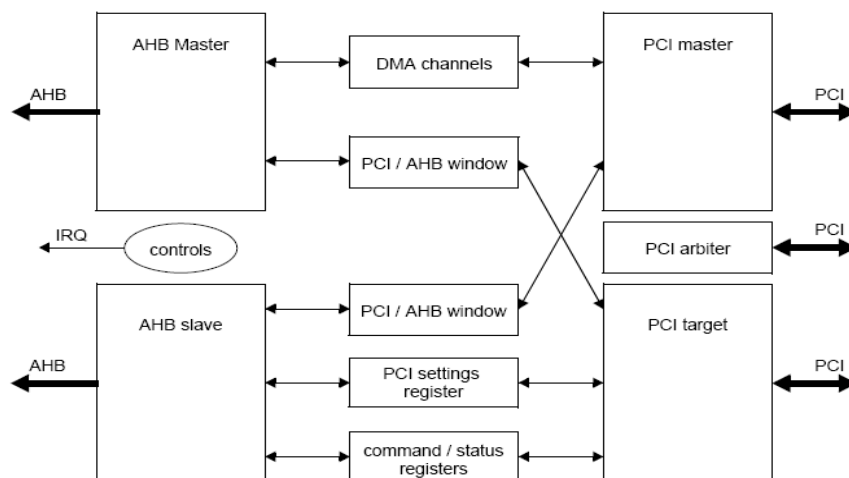


图4-8-2 AHB-PCI 桥模块结构框图

PCI 接口结构如图 4-8-2 所示，PCI master/target 接口将 AHB 系统连接到 PCI 总线上；BM3823A 通过 AHB slave 模块控制和配置 AHB-PCI 桥模块；BM3823A 可通过 DMA 通道配置和访问 PCI 总线设备。Host 模式下的 AHB-PCI 桥包含一个内嵌的 PCI 仲裁模块，此模块可管理 7 个外部 PCI 设备。PCI-AHB 地址空间映射即 PCI-AHB window 使 PCI 总线 Master 设备可以直接访问 CPU 存储器和 AHB 设备。

AHB 总线 Master 设备可以通过 AHB-PCI 地址空间映射即 AHB-PCI window 读写存储器和 I/O 空间来访问 PCI 总线设备。

4.8.3 PCI 接口复位

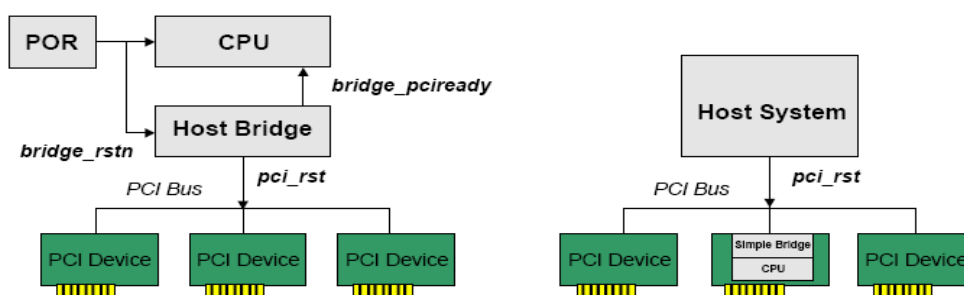


图4-8-3 PCI总线复位

如图 4-8-3 所示，Host 模式下 BM3823A 的 PCI 接口，在 BM3823A 电路复位时发出 PCI 总线复位信号，并报告 PCI 总线复位是否完成。BM3823A 向 AHB 地址 0x8000-00D0 寄存

器写入 0x05 可以使 PCI 总线复位。

Guest 模式下 BM3823A 的 PCI 接口，在接收到 PCI 总线上的复位信号时 PCI 侧逻辑复位，接收到 BM3823A 复位信号时 AHB 侧逻辑复位。

4.8.4 PCI 接口 AHB 侧地址空间

BM3823A 的 PCI 接口，映射到四块独立的 AHB 总线空间，如图 4-8-4 所示。

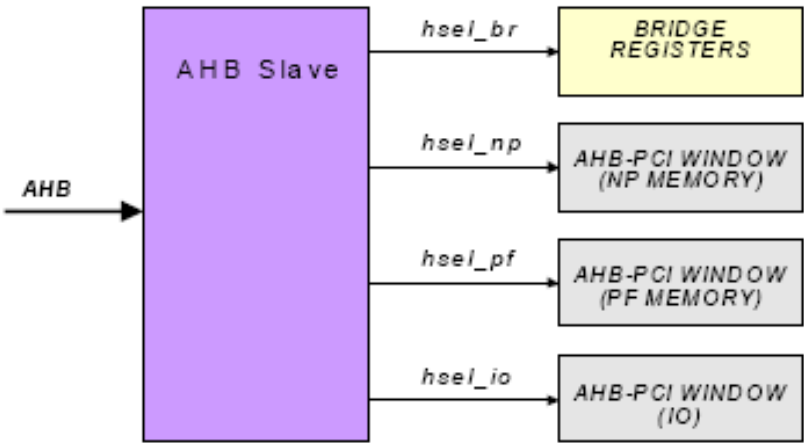


图4-8-4 AHB-PCI地址映射

空间分配如下：

寄存器空间（bridge registers）：0xC000-0000 ~ 0xC7FF-FFFF

AHB-PCI 不可预取地址空间映射（NP）：0xD000-0000 ~ 0xDFFF-FFFF

AHB-PCI 可预取地址空间映射（PF）：0xE000-0000 ~ 0xFFFF-FFFF

AHB-PCI I/O 地址空间映射（I/O）：0xC800-0000 ~ 0xCFFF-FFFF

其中 AHB 侧桥寄存器空间，基地址为 0xC000-0000，偏移地址如表 4-8-1 所示。

表4-8-1 AHB侧PCI相关映射寄存器

Byte Offset	Mode	Name	Description
000h	R/W	WDMA_PCI_ADDR	Write DMA start address on the PCI bus
004h	R/W	WDMA_AHB_ADDR	Write DMA transfer start address on the AHB bus
008h	R/W	WDMA_CONTROL	Write DMA size & control
00Ch ... 01Ch	reserved		
020h	R/W	RDMA_PCI_ADDR	Read DMA start address on the PCI bus

Byte Offset	Mode	Name	Description
024h	R/W	RDMA_AHB_ADDR	Read DMA transfer start address on the AHB bus
028h	R/W	RDMA_CONTROL	Read DMA size & control
02Ch ... 03Ch		reserved	
040h	R/W	CPU_IMASK	Interrupt mask
044h	R/W/C	CPU_ISTATUS	Interrupt status
048h	W	CPU_ICMD	Interrupt command
04Ch	R	CPU_VERSION	Bridge version and miscellaneous information
050h ... 06Ch		reserved	
070h	R/W	PCIAHB_ADDR_NP	PCI-AHB window non-prefetchable range control
074h	R/W	PCIAHB_ADDR_PF	PCI-AHB window prefetchable range control
078h	R/W	PCIAHB_TIMER	PCI-AHB window discard timer
07Ch	R/W	AHBPCI_TIMER	AHB-PCI window discard timer
080h	R/W/C	PCI_CONTROL	PCI control bits
084h	R or R/W	PCI_DV	PCI device and vendor ID
088h	R or R/W	PCI_SUB	PCI subsystem device and vendor ID
08Ch	R or R/W	PCI_CREV	PCI class code and revision ID
090h	R/W/C	PCI_BROKEN	PCI arbiter broken master register
094h	R or R/W	PCIAHB_SIZ_NP	PCI-AHB window non-prefetchable range size
098h	R or R/W	PCIAHB_SIZ_PF	PCI-AHB window prefetchable range size
09Ch ...3FCh		reserved	

4.8.5 PCI 仲裁器

PCI 仲裁模块在 BM3823A 处于 PCI Host 模式时有效，此模块提供 BM3823A 的 PCI 接口自身仲裁信号和最多七个其他 PCI 设备的仲裁信号。此模块的功能是：

- ◆ 使能 PCI 总线 Master 设备以开始 DMA 传输；

- ◆ 避免 PCI 总线冲突；
- ◆ 保证所有的 PCI 总线 Master 设备以同等的机会有规则地传送数据。

仲裁器能发现屏蔽并报告出错的 PCI 总线设备，以使其不影响 PCI 系统的性能。如果一个 PCI Master 设备请求 PCI 总线 16 个时钟周期以上而不开始一次总线操作，仲裁器就认为此 PCI 总线设备功能不正常。

PCI_Broken 寄存器的低 8 位反映 PCI 总线 Master 设备的状态，其中任 1 位为 1 则说明对应的 PCI Master 设备功能不正常而且被禁止访问 PCI 总线。向此位写 1，则重新使能相应的 PCI Master 设备。

4.8.6 PCI 侧寄存器空间

Guest 模式下，BM3823A 的 PCI 接口实现了 PCI 标准配置空间，除此之外，PCI 侧寄存器还包括标准 PCI 中断的桥寄存器和 PCI 版本寄存器(这些寄存器地址在 0x80 ~ 0x8c 之间)。PCI 侧寄存器的地址空间如表 4-8-2 所示。

表4-8-2 PCI配置寄存器地址空间

Offset	Mode	Name	Description
00h...3Ch	—	—	Standard PCI Header
40h...7Ch	—	—	reserved
80h	R/W	PCI_IMASK	Interrupt mask
84h	R/W/C	PCI_ISTATUS	Interrupt status
88h	W	PCI_ICMD	Interrupt command
8Ch	R	PCI_VERSION	Bridge version and miscellaneous information
90h...FCh	—	—	reserved

PCI 接口处于 Guest 模式时，实现了 PCI2.3 协议规定的标准 PCI 配置寄存器中，六个 PCI 地址空间定义寄存器 bar0---bar5 中的 bar3-bar5，如表 4-8-3 所示。

表4-8-3 PCI目标接口地址空间

Space	Size	Resource
BAR0...BAR2		reserved

Space	Size	Resource
BAR3	user defined	PCI-AHB window non-prefetchable area
BAR4	user defined	PCI-AHB window prefetchable area
BAR5	256 bytes	same as configuration space (Guest Bridge only)

当 BM3823A 的 PCI 接口处于 Host 模式时，并不存在 PCI 侧寄存器，但可以通过配置读写对其他 PCI 总线设备进行配置。

PCI Host 设备为 PCI 总线上的 BM3823A PCI Guest 设备资源分配地址空间，包括 Memory 和 I/O，需设置 PCI 配置空间的 BAR3、BAR4、BAR5 三个基地址寄存器，BM3823A PCI 设备配置空间中，PCI 头标区偏移 0x1c、0x20、0x24 的三个寄存器，即 BAR0，BAR1，BAR2 三个基地址寄存器未实现；其中 BAR5 为配置空间 PCI 总线基地址，这样可以让 BM3823A PCI Guest 通过 Memory 读写取代通过配置读写来访问其他设备配置空间中的各种寄存器的方式，如读写中断寄存器，或者重新配置配置空间中的 BAR3 或/和 BAR4 寄存器以实现地址空间重新分配。中断相关寄存器使用见图 4-8-11。

4.8.7 地址空间映射

4.8.7.1 AHB-PCI 地址空间映射

AHB-PCI 地址空间映射将 AHB 总线读写操作转化为 PCI 总线读写操作。

AHB 总线 Master 设备使用 AHB-PCI 地址空间映射直接访问 PCI 设备。AHB-PCI 地址空间映射捕获指向 PCI 接口 AHB 空间的 AHB 操作，并将这些操作传输到 PCI 总线上。这些操作并不需要 BM3823A 的 IU 等其他模块介入。

AHB-PCI 地址空间映射如图 4-8-5 所示，包括不可预取存储器映射空间、可预取存储器映射空间，可预取 I/O 映射空间三部分，AHB 侧地址范围在 PCI 接口 AHB 侧地址范围之内。

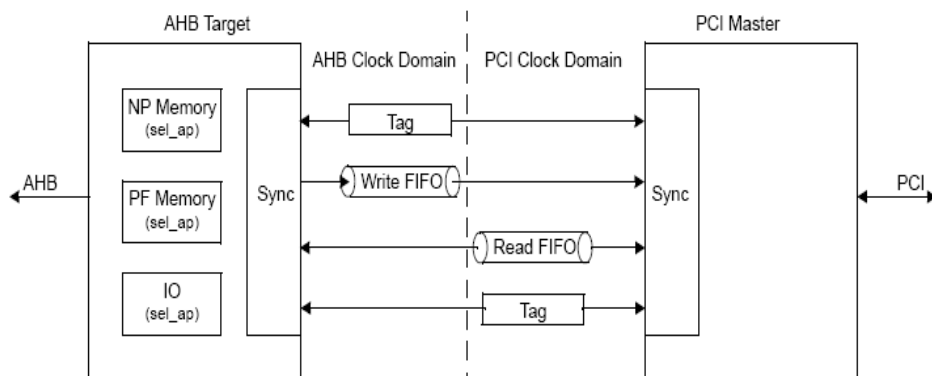


图4-8-5 AHB-PCI地址空间映射

AHB-PCI 地址空间映射使用扁平寻址方式，AHB 总线操作的地址与其转化成的 PCI 总线操作的地址相同。

具体说，BM3823A 访问 AHB 地址空间段 0xC800-0000 ~ 0xFFFF-FFFF 时，可以自动产生相应的 PCI 总线操作，即 PCI 接口访问 PCI 地址空间 0xC800-0000 ~ 0xFFFF-FFFF 地址时：

- ◆ 0xC800-0000 ~ 0xCFFF-FFFF 落在 AHB-PCI I/O 地址空间映射；将产生 PCI I/O 读写操作。
- ◆ 0xE000-0000 ~ 0xFFFF-FFFF 落在 AHB-PCI 可预取地址空间映射；将产生 PCI 可预取存储器读写操作。
- ◆ 0xD000-0000 ~ 0xDFFF-FFFF 落在 AHB-PCI 不可预取地址空间映射。将产生 PCI 不可预取存储器读写操作。

如上所述，BM3823A 可以利用地址空间映射机制，通过 AHB 总线读写操作直接访问 0xC800-0000 ~ 0xFFFF-FFFF 范围的 PCI 总线地址，这个范围内的资源也可以通过 DMA 访问，包括可预取地址空间映射和不可预取地址空间映射以及 I/O 地址空间映射，其余的 PCI 地址空间则只能通过 PCI 接口的 DMA 机制访问。

4.8.7.2 PCI-AHB 地址空间映射

PCI-AHB 地址空间映射将 PCI 总线读写操作转化为 AHB 总线读写操作。

PCI 总线的其他 Master 设备通过 PCI-AHB 地址空间映射直接访问 BM3823A 本地存储器和片内 AHB 总线设备。PCI-AHB 地址空间映射捕获指向自己 PCI 空间的 PCI 操作，并将这些操作传输到 AHB 总线上。同 AHB-PCI 地址空间映射类似，这些操作并不将 IU 等其他模块牵连其中。

PCI-AHB 地址空间映射如图 4-8-6 所示，包括 Bar3、Bar4、Bar5 三部分，分别为不可预取 PCI 存储器映射空间、可预取 PCI 存储器映射空间和 PCI 接口 PCI 侧寄存器空间。地址转换的机制随 PCI 接口模式不同而不同。

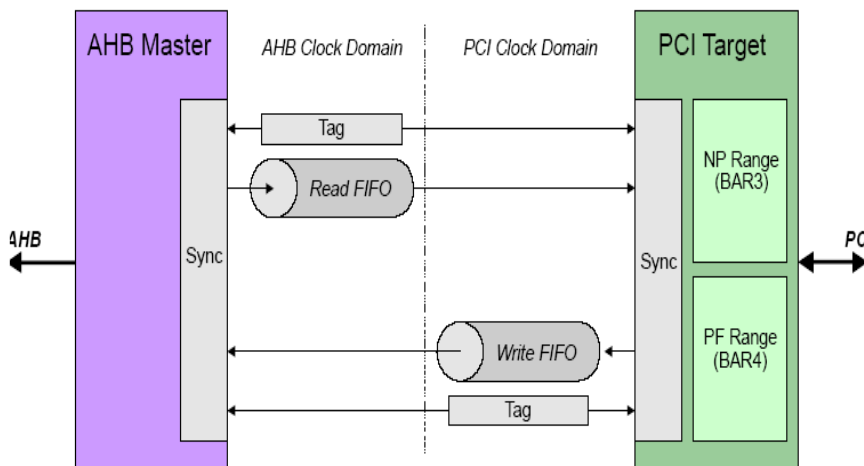


图4-8-6 PCI-AHB地址映射

PCI-AHB 地址空间映射包括 Bar3、Bar4、Bar5 三部分：

- ◆ 不可预取存储器映射空间被映射到 PCI 设备的 BAR3，处理指向此空间范围内的 PCI 操作不会进行数据预取。
- ◆ 可预取存储器映射空间被映射到 PCI 设备的 BAR4，处理指向此空间范围内的 PCI 操作会自动进行数据预取。
- ◆ PCI 接口 PCI 侧寄存器空间在 PCI 总线上被映射成 BAR5。这样使用存储器读写也可以修改 PCI 接口配置寄存器。

Bar3 和 Bar4 的大小在 256 字节和 1G 字节之间。其对应的 AHB 侧地址使用 AHB 侧寄存器定义。分别由 PCIAHB_ADDR_NP 和 PCIAHB_ADDR_PF 定义其 AHB 基地址，PCIAHB_SIZ_NP 和 PCIAHB_SIZ_PF 两个寄存器定义其大小。

在 Host 模式下，PCI-AHB 地址空间映射不进行地址转换，即 PCI 总线读写操作和其转换成的 AHB 总线读写操作目标地址一致。反过来说 Host 模式下 PCI 接口的 PCI-AHB 地址空间映射的 PCI 侧地址范围由 AHB 侧寄存器定义。

在 Guest 模式下，PCI 接口的 PCI-AHB 地址空间映射的 PCI 侧地址范围由 BAR3 base address、BAR4 base address、BAR5 base address 决定。PCI 空间和 AHB 空间之间存在如下

转换：

不可预取空间：

$AHB\ address = PCIAHB\_ADDR\_NP + (PCI\ address - BAR3\ base\ address);$

可预取空间：

$AHB\ address = PCIAHB\_ADDR\_PF + (PCI\ address - BAR4\ base\ address)。$

4.8.8 DMA

PCI 接口的很大一部分功能是通过 DMA 机制进行的，DMA 通道如图 4-8-7 所示，在 DMA 机制中 DMA 控制器在控制 PCI 总线的同时也控制 AHB 总线，实现独立于 CPU 的数据传输。

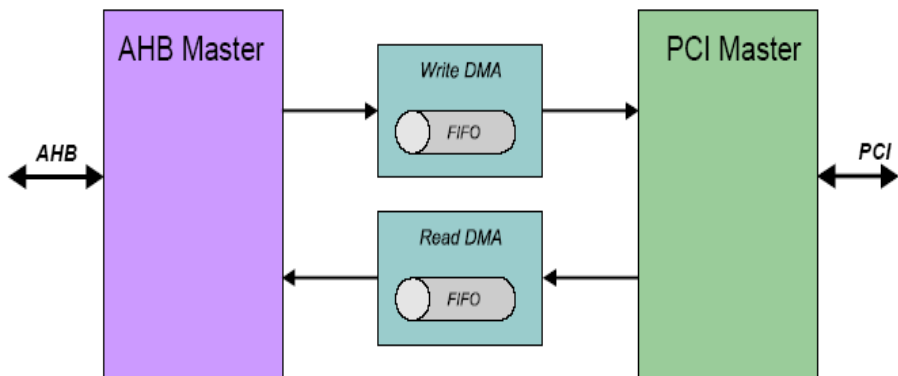


图4-8-7 DMA通道

DMA 机制可以在 AHB 总线和 PCI 总线之间实现大量的 Memory 或 I/O 数据传输而仅需要少量的 CPU 干预，也可以执行配置周期和 I/O 操作。

CPU 分配和初始化一段存储器缓冲区，然后初始化设置 DMA 寄存器，并指定 PCI 交易的类型和目标地址。交易被初始化为 AHB Master 后，CPU 再无任何动作，PCI Master 会自动执行所需要的 PCI 总线数据交易，除非用户终止 DMA 传输。DMA 传输状态将报告给 CPU_ISTATUS 寄存器，可以通过此寄存器引起 BM3823A 处理器中断。

在 DMA 操作中，DMA 读是指从 PCI 地址空间中向 BM3823A 的 AHB 地址空间中读取数据；DMA 写是指把 BM3823A 的 AHB 地址空间中的数据写入 PCI 地址空间。

发起 DMA 操作需要三个寄存器：PCI 地址寄存器、AHB 地址寄存器和 DMA 控制寄存器。DMA 写传输需要配置的寄存器有：PCI 地址寄存器，地址为 0xC000_0000；AHB 地址

寄存器，地址为 0xC000_0004；DMA 写控制寄存器，地址为 0xC000_0008。DMA 读传输需要配置的寄存器有：PCI 地址寄存器，地址为 0xC000_0020；AHB 地址寄存器，地址为 0xC000_0024；DMA 读控制寄存器，地址为 0xC000_0028。DMA 传输的 AHB 地址寄存器 WDMA_AHB_ADDR、RDMA_AHB_ADDR 会随着 DMA 的进行自动增长，直至 DMA 结束。

DMA 控制寄存器 DMA_CONTROL 定义了 PCI 总线上数据传输的 PCI 命令和控制操作，结构如图 4-8-8 所示。

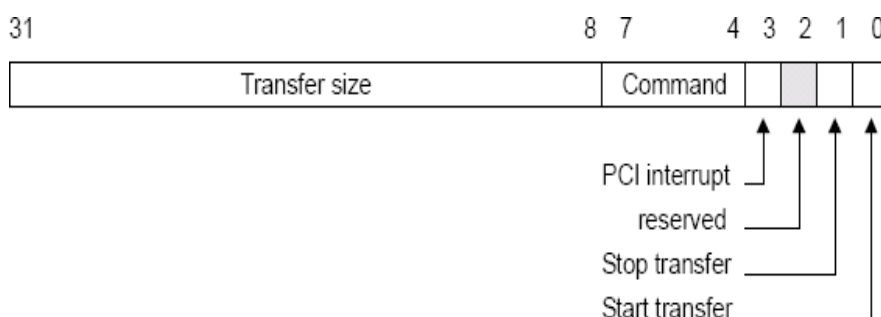


图4-8-8 DMA控制寄存器

- ◆ Bit[3] 置 1 可以使 DMA 完成时在 PCI 总线上产生中断信号。
- ◆ Bit[1] 置 1 可以终止正在进行的 DMA。
- ◆ Bit[0] 置 1 可以启动 DMA，完成时自动清零，DMA 完成/退出时会在中断状态寄存器相应的位置 1，使用时在 DMA 开始之后，等待并判断 DMA 完成时可以查询此位。

Transfer size 为 DMA 传输字节长度，最大为 16MB，取值为 0x1-0xFFFFFC；

Command 部分表示发起的 DMA 操作类型，如表 4-8-4 所示。

表4-8-4 DMA操作类型

Channel	Command	Description
Read DMA	0000	Interrupt Acknowledge
	0010	I/O Read
	0110	Memory Read
	1010	Configuration Read
	1100	Memory Read Multiple

Channel	Command	Description
Write DMA	0001	Special Cycle
	0011	I/O Write
	0111	Memory Write
	1011	Configuration Write

DMA 写操作示例：

```
*(unsigned int *)0xc0000000 = 0xc0000000; // Write DMA start address on the PCI bus
```

```
*(unsigned int *)0xc0000004 = 0x20000000; // Write DMA transfer start address on the AHB bus
```

```
*(unsigned int *)0xc0000008 = 0x10071; // Write DMA size & control
```

发起把 AHB 地址 0x2000-0000 处 0x100 字节写往 PCI 地址 0xC000-0000 处的 DMA Memory Write 操作。

DMA 读操作示例：

```
*(unsigned int *)0xc0000020 = 0x20000000;
```

```
*(unsigned int *)0xc0000024 = 0x20001000;
```

```
*(unsigned int *)0xc0000028 = 0x100a1;
```

则读出了槽位地址在 0x2000-0000 处的 PCI 设备配置空间中 0x100 字节的内容。

BM3823A 的 PCI Host 设备通过配置 DMA 进行 PCI 配置时，应尽量使用单字 DMA 读写。配置 DMA 读写时 PCI 地址的确定由 PCI 总线底板的槽位及选择关系决定。

通过 DMA 操作产生 PCI 总线类型 0 和类型 1 配置周期时的 PCI 总线地址格式如图 4-8-9 所示。

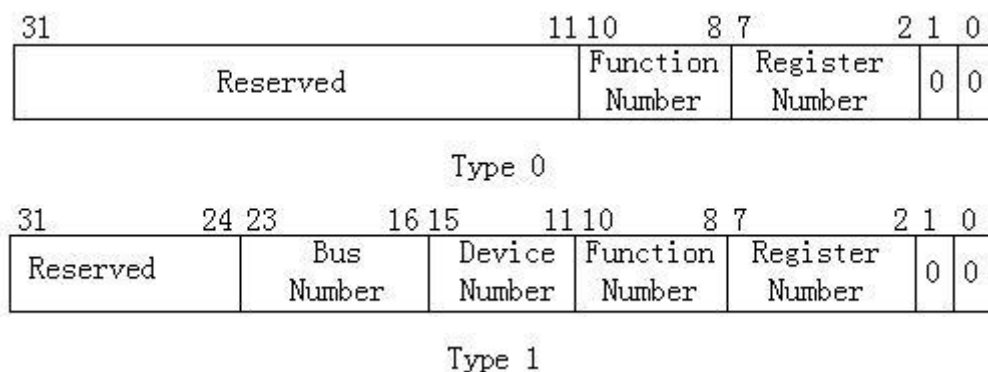


图4-8-9 Type 0和Type 1地址空间配置

BM3823A 的 AHB 总线为大端总线，而 PCI 总线为小端总线，所以他们之间的 DMA 传输需按照一定的要求来保证数据顺序一致。

当 BM3823A 的 DMA 目标或源地址为 32 位的 RAM 空间且进行 DMA 传输的 PCI 设备为两个 BM3823A PCI 设备时，需遵循以下原则以避免字节序错位：

- ◆ PCI 地址和 AHB 地址后两位相等时，进行任意类型字节长度的 DMA；
- ◆ PCI 地址和 AHB 地址 2 字节即半字对齐时，DMA 长度为 2 或 2 的倍数；
- ◆ PCI 地址和 AHB 地址 4 字节即字对齐时，DMA 长度为 4 或 4 的倍数。

当 BM3823A 的 DMA 目标或源地址为 8 位 RAM 空间时，则需要 PCI 地址和 AHB 地址 4 字节对齐，DMA 长度为 4 字节或 4 的倍数。

推荐使用 PCI 地址和 AHB 地址 4 字节对齐，DMA 长度为 4 字节或 4 的倍数的 DMA 方式，此时无论何种情况都不会被大小端问题影响。

4.8.9 PCI 模块中断

PCI 模块中断既可以通过 AHB 总线向 BM3823A 的 IU 模块发出中断请求，也可以在 Guest 模式下通过 PCI 总线发出中断或响应来自 Host 模式 PCI 总线的中断请求。前者称为处理器中断，后者称为 PCI 中断。

4.8.9.1 处理器中断

PCI 接口的处理器中断为 BM3823A 处理器的 14 号中断，PCI 部分配置为 Host 或 Guest 时，其实现稍有不同，如图 4-8-10 所示。AHB 侧有三个中断相关寄存器：

- ◆ 0xC000-0040 : CPU_IMASK CPU 中断掩码寄存器。
- ◆ 0xC000-0044 : CPU_ISTATUS CPU 中断状态寄存器。
- ◆ 0xC000-0048 : CPU_ICMD CPU 中断命令寄存器。

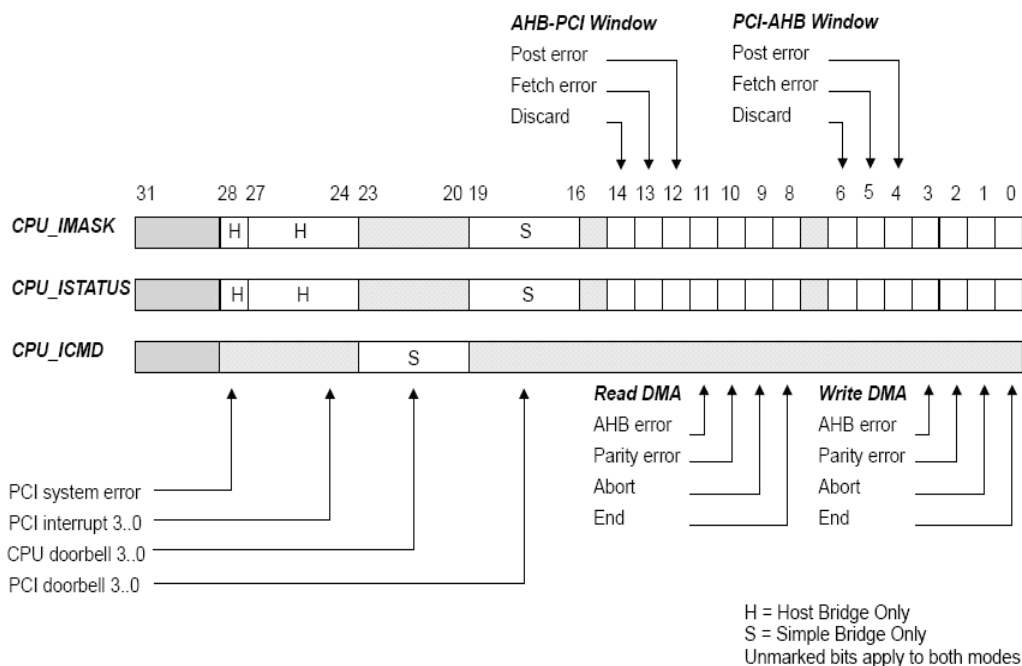


图4-8-10 CPU中断寄存器

- ◆ **CPU_ISTATUS:** CPU 中断状态寄存器，此寄存器可读/可写/可清零。相应的中断源被激活，状态寄存器相应位自动置位为 1，各个中断源是相互独立的且多个中断源可以同时触发。CPU 可以监控并清除状态位，写 1 清除相应状态位，写 0 无效。
- ◆ **CPU_IMASK:** CPU 中断掩码寄存器，此寄存器可读/可写。寄存器置位使能相应的中断源，清零屏蔽相应的中断源。上述操作不影响 CPU_ISTATUS 寄存器各位的值：也就是说被屏蔽的中断仍会报告给相应的状态位。如果一个或多个中断源被触发且未被屏蔽，则 PCI 桥就会向 CPU 申请中断。
- ◆ **CPU_ICMD:** CPU 中断命令寄存器，只写寄存器，用于人为使能一些中断源。读操作返回 0。写 1 置位，写 0 无效。

4.8.9.2 PCI 中断

PCI 侧配置空间中也有三个中断寄存器，PCI_IMASK、PCI_ISTATUS、PCI_ICMD，分别为 PCI 中断掩码寄存器，PCI 中断状态寄存器，PCI 中断命令寄存器。这三个寄存器的 CPU Doorbell 位和 PCI Doorbell 位和 CPU 中断寄存器的对应位配合使用产生中断请求，这些位只在 Guest 模式下有效。这三个寄存器只能通过 PCI 总线读写，如图 4-8-11 所示。

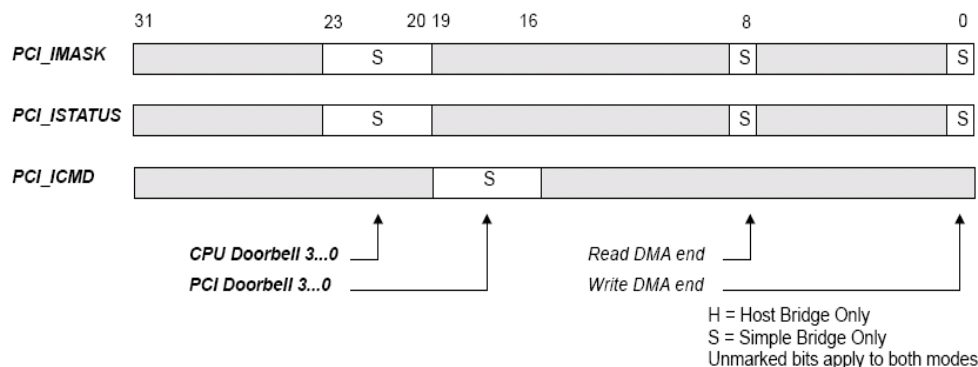


图4-8-11 PCI中断寄存器

- ◆ **PCI Doorbell:** PCI 设备可以通过访问此位向 BM3823A 中断控制器发出中断请求。若 PCI Doorbell 中断在 CPU_IMASK 寄存器中使能，此时置位 PCI Doorbell 位，则 BM3823A 的 CPU 中断中的 PCI Doorbell 中断有效。该位只写，写 0 无效，写 1 置位。此位只在 BM3823A 配置为 Guest bridge 模式下有效。
- ◆ **CPU Doorbell:** CPU Doorbell 中断由 BM3823A 写 CPU_ICMD 对应位触发。若此时 PCI_IMASK 寄存器中断源被使能，则 BM3823A 可以通过 PCI 总线 INTA# 发送用户自定义的中断，并访问 PCI_ISTATUS 寄存器查询中断类型。此位只在 BM3823A 配置为 Guest bridge 模式下有效。
- ◆ **Read DMA end 和 Write DMA end 的使用和 CPU Doorbell 类似，也是通过 PCI 总线的 INTA# 信号向 PCI 总线 Host 设备发出中断信号，中断由 DMA 完成事件触发。此位只在 BM3823A 配置为 Guest bridge 模式下有效。**

PCI 中断使用说明：PCI 中断为电平中断，处理器响应 14 号中断即 PCI 中断后，需要软件首先清除中断源 PCI 模块内的中断请求，即向 CPU_ISTATUS 寄存器中相应位写 1，然后清除 14 号中断，从而清除相应 PCI 中断。

4.9 内部存储器接口

4.9.1 概述

内部存储器容量为 256KB。支持纠一检二的 EDAC 功能。数据位为 32 位，对应校验位为 7 位。

4.9.2 内部存储器地址空间分配

表4-9-1 内部存储器地址空间分配

地址范围	大小	映射
0xA000-0000 - 0xA003-FFFF	256KB	内部 SRAM

4.9.3 内部存储器寄存器地址映射

内部存储器寄存器地址映射见表 4-9-2，详细说明见附录 3.9。

表4-9-2 GPIO寄存器地址映射表

寄存器名称	地址	复位值 ^注
内部存储器配置寄存器	0x80000030	0x000000XX

注：“复位值”按照十六进制统一写法“0xXXX0XXXX”，“0x”表示 16 进制，每 4bit 中有 1bit 不定值，即写为“X”，4bit 全部为固定值则写为相应 16 进制数值。

4.9.4 内部存储器 EDAC 进 trap 情况

当内存访问出现 EDAC 错误时，会进入 trap 并记录错误状态，记录错误状态的寄存器为 **AHB 失效地址寄存器 FAILAR**(0x8000-000C)、**AHB 状态寄存器 FAILSR**(0x8000-0010)。

表4-9-3 内存控制器EDAC进trap和中断1以及错误状态位的机制表

条件		结果			
数据错误 位数	EDAC 使能 配置	是否进入 trap	是否进入中断 1	FAILSR 的 CE 位	FAILSR 的 NE 位
一位	使能	否	是	1	0
二位	使能	是	是	0	1

4.10 外部存储器接口

4.10.1 概述

BM3823A 通过外部存储器控制器可以访问 PROM、存储器映射的 I/O 设备、SRAM 和 SDRAM。系统可配两个 PROM、6 个 I/O 设备、5 个 SRAM 以及两个 SDRAM。其中 I/O 设备支持 SpaceWire 逻辑接口，用于对 SpaceWire 通讯控制器进行访问控制。I/O 设备支持 1553 逻辑接口，用于实现对 1553B 通讯控制器 16 位 buffer 模式的访问控制。

支持片选数量和数据位宽见表 4-10-1。

表4-10-1 外部存储器类型表

存储器类型	片选数量	数据位宽
PROM	2	32、16、8
I/O	6	32、16、8
SRAM	5	32、16、8
SDRAM	2	32

外部存储器接口配置寄存器包括：MCFG1、MCFG2、MCFG3、MECFG1、MECFG2。其中 MCFG1 主要用于配置 PROM 和 I/O 设备，MCFG2 用于配置 SRAM 和 SDRAM，MCFG3 用于配置 SDRAM 和 I/O 设备，MECFG1、MECFG2 用于容错配置。

4.10.2 外部存储器控制器地址空间

外部存储器的地址空间为 1G 字节，地址空间映射见表 4-10-2。

表4-10-2 外部存储器地址空间

地址范围	大小	映射
0x0000-0000~0x0FFF-FFFF	256MB	PROM
0x1000-0000~0x1FFF-FFFF	256MB	I/O
0x2000-0000~0x3FFF-FFFF	512MB	SRAM/SDRAM

4.10.2.1 PROM 地址空间分配

PROM 最大支持 2 个片选，每个片选容量无需配置，每个片选容量最大为 128MB。

表4-10-3 PROM地址空间

大小	起始地址	结束地址	片选信号
128MB	0x0000-0000	0x07FF-FFFF	romsn[0]
128MB	0x08000-0000	0x0FFF-FFFF	romsn[1]

4.10.2.2 I/O 地址空间分配

I/O 设备支持 6 个片选，容量无需配置，片选容量最大为 32MB 或 64MB。

表4-10-4 I/O地址空间

大小	起始地址	结束地址	片选信号
32MB	0x1000-0000	0x11FF-FFFF	iosn[0]
32MB	0x1200-0000	0x13FF-FFFF	iosn[1]
32MB	0x1400-0000	0x15FF-FFFF	iosn[2]
32MB	0x1600-0000	0x17FF-FFFF	iosn[3]
64MB	0x1800-0000	0x1BFF-FFFF	iosn[4]
64MB	0x1C00-0000	0x1FFF-FFFF	iosn[5]

4.10.2.3 SRAM 和 SDRAM 空间分配

RAM 地址范围可达 512MB 字节，支持两种 RAM 类型：SRAM 和 SDRAM。SRAM 和 SDRAM 地址分配见表 4-10-5。

表4-10-5 SRAM和SDRAM地址分配

SRAM 不使能位 (SRDIS)	SDRAM 使能位 (SDREN)	地址空间	RAM 类型
0 或 1	0	0x2000-0000~0x2fff-ffff	SRAM0、1、2、3
		0x3000-0000~0x3fff-ffff	SRAM4
0	1	0x2000-0000~0x2fff-ffff	SRAM0、1、2、3
		0x3000-0000~0x3fff-ffff	SDRAM
1	1	0x2000-0000~0x3fff-ffff	SDRAM

SRAM 最大支持 5 个片选，前 4 个片选容量大小可配置，范围是 8KB~256MB。第 5 个片选容量无需配置，第 5 个片选容量最大 256MB。

SDRAM 最大支持 2 个片选，每个片选容量大小可配置，范围是 4MB~512MB。

4.10.2.3.1 SRAM 使能、SDRAM 使能

● SRAM 地址空间分配

表4-10-6 SRAM地址空间（SRAM使能、SDRAM使能）

SRBS 配置	大小	起始地址	结束地址	片选信号
1111	256MB	0x2000-0000	0x2FFF-FFFF	ramsn[0]
1110	128MB	0x2000-0000	0x27FF-FFFF	ramsn[0]
	128MB	0x2800-0000	0x2FFF-FFFF	ramsn[1]
1101	64MB	0x2000-0000	0x23FF-FFFF	ramsn[0]
	64MB	0x2400-0000	0x27FF-FFFF	ramsn[1]
	64MB	0x2800-0000	0x2BFF-FFFF	ramsn[2]
	64MB	0x2C00-0000	0x2FFF-FFFF	ramsn[3]
1100	32MB	0x2000-0000	0x21FF-FFFF	ramsn[0]
	32MB	0x2200-0000	0x23FF-FFFF	ramsn[1]
	32MB	0x2400-0000	0x25FF-FFFF	ramsn[2]
	32MB	0x2600-0000	0x27FF-FFFF	ramsn[3]
1011	16MB	0x2000-0000	0x20FF-FFFF	ramsn[0]
	16MB	0x2100-0000	0x21FF-FFFF	ramsn[1]
	16MB	0x2200-0000	0x22FF-FFFF	ramsn[2]
	16MB	0x2300-0000	0x23FF-FFFF	ramsn[3]
1010	8MB	0x2000-0000	0x207F-FFFF	ramsn[0]
	8MB	0x2080-0000	0x20FF-FFFF	ramsn[1]
	8MB	0x2100-0000	0x217F-FFFF	ramsn[2]
	8MB	0x2180-0000	0x21FF-FFFF	ramsn[3]
1001	4MB	0x2000-0000	0x203F-FFFF	ramsn[0]
	4MB	0x2040-0000	0x207F-FFFF	ramsn[1]
	4MB	0x2080-0000	0x20BF-FFFF	ramsn[2]
	4MB	0x20C0-0000	0x20FF-FFFF	ramsn[3]
1000	2MB	0x2000-0000	0x201F-FFFF	ramsn[0]
	2MB	0x2020-0000	0x203F-FFFF	ramsn[1]
	2MB	0x2040-0000	0x205F-FFFF	ramsn[2]
	2MB	0x2060-0000	0x207F-FFFF	ramsn[3]
0111	1MB	0x2000-0000	0x200F-FFFF	ramsn[0]
	1MB	0x2010-0000	0x201F-FFFF	ramsn[1]

SRBS 配置	大小	起始地址	结束地址	片选信号
	1MB	0x2020-0000	0x202F-FFFF	ramsn[2]
	1MB	0x2030-0000	0x203F-FFFF	ramsn[3]
0110	512KB	0x2000-0000	0x2007-FFFF	ramsn[0]
	512KB	0x2008-0000	0x200F-FFFF	ramsn[1]
	512KB	0x2010-0000	0x2017-FFFF	ramsn[2]
	512KB	0x2018-0000	0x201F-FFFF	ramsn[3]
0101	256KB	0x2000-0000	0x2003-FFFF	ramsn[0]
	256KB	0x2004-0000	0x2007-FFFF	ramsn[1]
	256KB	0x2008-0000	0x200B-FFFF	ramsn[2]
	256KB	0x200C-0000	0x200F-FFFF	ramsn[3]
0100	128KB	0x2000-0000	0x2001-FFFF	ramsn[0]
	128KB	0x2002-0000	0x2003-FFFF	ramsn[1]
	128KB	0x2004-0000	0x2005-FFFF	ramsn[2]
	128KB	0x2006-0000	0x2007-FFFF	ramsn[3]
0011	64KB	0x2000-0000	0x2000-FFFF	ramsn[0]
	64KB	0x2001-0000	0x2001-FFFF	ramsn[1]
	64KB	0x2002-0000	0x2002-FFFF	ramsn[2]
	64KB	0x2003-0000	0x2003-FFFF	ramsn[3]
0010	32KB	0x2000-0000	0x2000-7FFF	ramsn[0]
	32KB	0x2000-8000	0x2000-FFFF	ramsn[1]
	32KB	0x2001-0000	0x2001-7FFF	ramsn[2]
	32KB	0x2001-8000	0x2001-FFFF	ramsn[3]
0001	16KB	0x2000-0000	0x2000-3FFF	ramsn[0]
	16KB	0x2000-4000	0x2000-7FFF	ramsn[1]
	16KB	0x2000-8000	0x2000-BFFF	ramsn[2]
	16KB	0x2000-C000	0x2000-FFFF	ramsn[3]
0000	8KB	0x2000-0000	0x2000-1FFF	ramsn[0]
	8KB	0x2000-2000	0x2000-3FFF	ramsn[1]
	8KB	0x2000-4000	0x2000-5FFF	ramsn[2]
	8KB	0x2000-6000	0x2000-7FFF	ramsn[3]

● SDRAM 地址空间分配

表4-10-7 SDRAM地址空间（SRAM使能、SDRAM使能）

SDRBS 配置	大小	起始地址	结束地址	片选信号
110	256MB	0x3000-0000	0x3FFF-FFFF	sdcsn[0]
101	128MB	0x3000-0000	0x37FF-FFFF	sdcsn[0]
	128MB	0x3800-0000	0x3FFF-FFFF	sdcsn[1]
100	64MB	0x3000-0000	0x33FF-FFFF	sdcsn[0]
	64MB	0x3400-0000	0x37FF-FFFF	sdcsn[1]
011	32MB	0x3000-0000	0x31FF-FFFF	sdcsn[0]
	32MB	0x3200-0000	0x33FF-FFFF	sdcsn[1]
010	16MB	0x3000-0000	0x30FF-FFFF	sdcsn[0]
	16MB	0x3100-0000	0x31FF-FFFF	sdcsn[1]
001	8MB	0x3000-0000	0x307F-FFFF	sdcsn[0]
	8MB	0x3080-0000	0x30FF-FFFF	sdcsn[1]
000	4MB	0x3000-0000	0x303F-FFFF	sdcsn[0]
	4MB	0x3040-0000	0x307F-FFFF	sdcsn[1]

4.10.2.3.2 SRAM 使能、SDRAM 不使能

● SRAM 地址空间分配

表4-10-8 前4个SRAM地址空间（SRAM使能、SDRAM不使能）

SRBS 配置	大小	起始地址	结束地址	片选信号
1111	256MB	0x2000-0000	0x2FFF-FFFF	ramsn[0]
1110	128MB	0x2000-0000	0x27FF-FFFF	ramsn[0]
	128MB	0x2800-0000	0x2FFF-FFFF	ramsn[1]
1101	64MB	0x2000-0000	0x23FF-FFFF	ramsn[0]
	64MB	0x2400-0000	0x27FF-FFFF	ramsn[1]
	64MB	0x2800-0000	0x2BFF-FFFF	ramsn[2]
	64MB	0x2C00-0000	0x2FFF-FFFF	ramsn[3]
1100	32MB	0x2000-0000	0x21FF-FFFF	ramsn[0]

SRBS 配置	大小	起始地址	结束地址	片选信号
	32MB	0x2200-0000	0x23FF-FFFF	ramsn[1]
	32MB	0x2400-0000	0x25FF-FFFF	ramsn[2]
	32MB	0x2600-0000	0x27FF-FFFF	ramsn[3]
1011	16MB	0x2000-0000	0x20FF-FFFF	ramsn[0]
	16MB	0x2100-0000	0x21FF-FFFF	ramsn[1]
	16MB	0x2200-0000	0x22FF-FFFF	ramsn[2]
	16MB	0x2300-0000	0x23FF-FFFF	ramsn[3]
1010	8MB	0x2000-0000	0x207F-FFFF	ramsn[0]
	8MB	0x2080-0000	0x20FF-FFFF	ramsn[1]
	8MB	0x2100-0000	0x217F-FFFF	ramsn[2]
	8MB	0x2180-0000	0x21FF-FFFF	ramsn[3]
1001	4MB	0x2000-0000	0x203F-FFFF	ramsn[0]
	4MB	0x2040-0000	0x207F-FFFF	ramsn[1]
	4MB	0x2080-0000	0x20BF-FFFF	ramsn[2]
	4MB	0x20C0-0000	0x20FF-FFFF	ramsn[3]
1000	2MB	0x2000-0000	0x201F-FFFF	ramsn[0]
	2MB	0x2020-0000	0x203F-FFFF	ramsn[1]
	2MB	0x2040-0000	0x205F-FFFF	ramsn[2]
	2MB	0x2060-0000	0x207F-FFFF	ramsn[3]
0111	1MB	0x2000-0000	0x200F-FFFF	ramsn[0]
	1MB	0x2010-0000	0x201F-FFFF	ramsn[1]
	1MB	0x2020-0000	0x202F-FFFF	ramsn[2]
	1MB	0x2030-0000	0x203F-FFFF	ramsn[3]
0110	512KB	0x2000-0000	0x2007-FFFF	ramsn[0]
	512KB	0x2008-0000	0x200F-FFFF	ramsn[1]
	512KB	0x2010-0000	0x2017-FFFF	ramsn[2]
	512KB	0x2018-0000	0x201F-FFFF	ramsn[3]
0101	256KB	0x2000-0000	0x2003-FFFF	ramsn[0]
	256KB	0x2004-0000	0x2007-FFFF	ramsn[1]
	256KB	0x2008-0000	0x200B-FFFF	ramsn[2]

SRBS 配置	大小	起始地址	结束地址	片选信号
	256KB	0x200C-0000	0x200F-FFFF	ramsn[3]
0100	128KB	0x2000-0000	0x2001-FFFF	ramsn[0]
	128KB	0x2002-0000	0x2003-FFFF	ramsn[1]
	128KB	0x2004-0000	0x2005-FFFF	ramsn[2]
	128KB	0x2006-0000	0x2007-FFFF	ramsn[3]
0011	64KB	0x2000-0000	0x2000-FFFF	ramsn[0]
	64KB	0x2001-0000	0x2001-FFFF	ramsn[1]
	64KB	0x2002-0000	0x2002-FFFF	ramsn[2]
	64KB	0x2003-0000	0x2003-FFFF	ramsn[3]
0010	32KB	0x2000-0000	0x2000-7FFF	ramsn[0]
	32KB	0x2000-8000	0x2000-FFFF	ramsn[1]
	32KB	0x2001-0000	0x2001-7FFF	ramsn[2]
	32KB	0x2001-8000	0x2001-FFFF	ramsn[3]
0001	16KB	0x2000-0000	0x2000-3FFF	ramsn[0]
	16KB	0x2000-4000	0x2000-7FFF	ramsn[1]
	16KB	0x2000-8000	0x2000-BFFF	ramsn[2]
	16KB	0x2000-C000	0x2000-FFFF	ramsn[3]
0000	8KB	0x2000-0000	0x2000-1FFF	ramsn[0]
	8KB	0x2000-2000	0x2000-3FFF	ramsn[1]
	8KB	0x2000-4000	0x2000-5FFF	ramsn[2]
	8KB	0x2000-6000	0x2000-7FFF	ramsn[3]

表4-10-9 第5个SRAM地址空间（SRAM使能、SDRAM不使能）

大小	起始地址	结束地址	片选信号
256MB	0x6000-0000	0x6FFF-FFFF	ramsn[4]

4.10.2.3.3 SRAM 不使能、SDRAM 使能

- SDRAM 地址空间分配

表4-10-10 SDRAM地址空间（SRAM不使能、SDRAM使能）

SDRBS 配置	大小	起始地址	结束地址	片选信号
111	512MB	0x2000-0000	0x3FFF-FFFF	sdcsn[0]
110	256MB	0x2000-0000	0x2FFF-FFFF	sdcsn[0]
	256MB	0x3000-0000	0x3FFF-FFFF	sdcsn[1]
101	128MB	0x2000-0000	0x27FF-FFFF	sdcsn[0]
	128MB	0x2800-0000	0x2FFF-FFFF	sdcsn[1]
100	64MB	0x2000-0000	0x23FF-FFFF	sdcsn[0]
	64MB	0x2400-0000	0x27FF-FFFF	sdcsn[1]
011	32MB	0x2000-0000	0x21FF-FFFF	sdcsn[0]
	32MB	0x2200-0000	0x23FF-FFFF	sdcsn[1]
010	16MB	0x2000-0000	0x20FF-FFFF	sdcsn[0]
	16MB	0x2100-0000	0x21FF-FFFF	sdcsn[1]
001	8MB	0x2000-0000	0x207F-FFFF	sdcsn[0]
	8MB	0x2080-0000	0x20FF-FFFF	sdcsn[1]
000	4MB	0x2000-0000	0x203F-FFFF	sdcsn[0]
	4MB	0x2040-0000	0x207F-FFFF	sdcsn[1]

4.10.2.3.4 SRAM 不使能、SDRAM 不使能

此种情况等同于 SRAM 使能、SDRAM 不使能的情况。

4.10.3 RAM 接口

RAM 支持两种 RAM 类型：SRAM 和 SDRAM。

4.10.3.1 SRAM 接口

SRAM 接口支持 5 个 SRAM 片选，SRAM 控制信号包括：片选 `ramsn[4:0]`、读使能 `ramoen[4:0]` 和写使能 `rwen[3:0]`。片选 `ramsn[4:0]` 与读使能 `ramoen[4:0]` 一一对应，写使能 `rwen[3:0]` 为所有片选共用。

表4-10-11 SRAM片选和读写使能对应关系表

片选	读使能	写使能
ramsn[4]	ramoen[4]	rwten[3:0]
ramsn[3]	ramoen[3]	
ramsn[2]	ramoen[2]	
ramsn[1]	ramoen[1]	
ramsn[0]	ramoen[0]	

前 4 个 SRAM 片选容量大小通过配置寄存器 MCFG2[12:9]进行配置，范围为 8K 字节到 256M 字节，当配置为 256M 字节时，片选 ramsn[3:0]只留下 ramsn[1:0]。第 5 个 SRAM 片选为 ramsn[4]，容量大小固定为 256M 字节，起始地址为 0x3000-0000。

SRAM 数据总线的位宽，支持 8 位、16 位和 32 位，通过配置寄存器 MCFG2[5:4]进行配置，其值是“00”表示 8 位总线，“01”表示 16 位总线，“1x”表示 32 位总线。

当 SRAM 配置为 8 位数据宽度时，SRAM 存储体数据管脚连接处理器的数据管脚为 data[31:24]，SRAM 存储体地址管脚最低位连接处理器的地址管脚为 address[0]。8 位数据宽度 SRAM 的 EDAC 功能同管脚 pdata[7:0]无关。

当 SRAM 配置为 16 位数据宽度时，SRAM 存储体数据管脚连接处理器的数据管脚为 data[31:16]，SRAM 存储体地址管脚最低位连接处理器的地址管脚为 address[1]。

当 SRAM 配置为 32 位数据宽度时，SRAM 存储体数据管脚连接处理器的数据管脚为 data[31:0]，SRAM 存储体地址管脚最低位连接处理器的地址管脚为 address[2]。如果 32 位数据宽度 SRAM 使用 EDAC 功能，处理器的校验位管脚 pdata[7:0]连接 SRAM 校验位存储体。

SRAM 连接示意图见图 4-10-1、图 4-10-2、图 4-10-3。

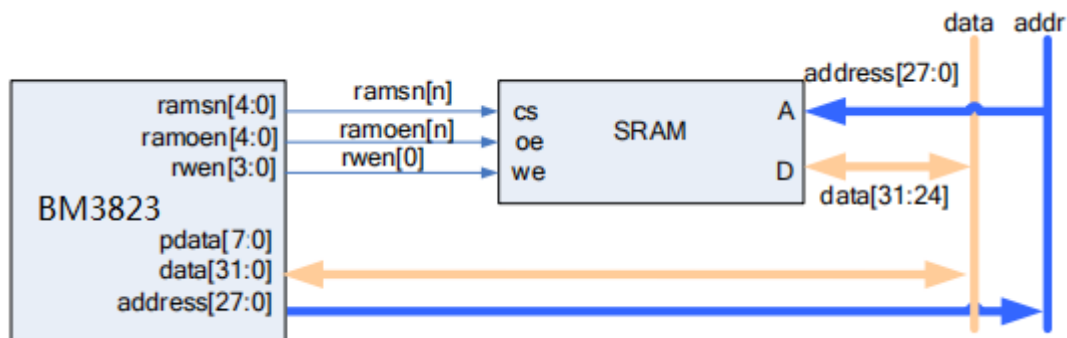


图4-10-1 8位宽SRAM连接示意图

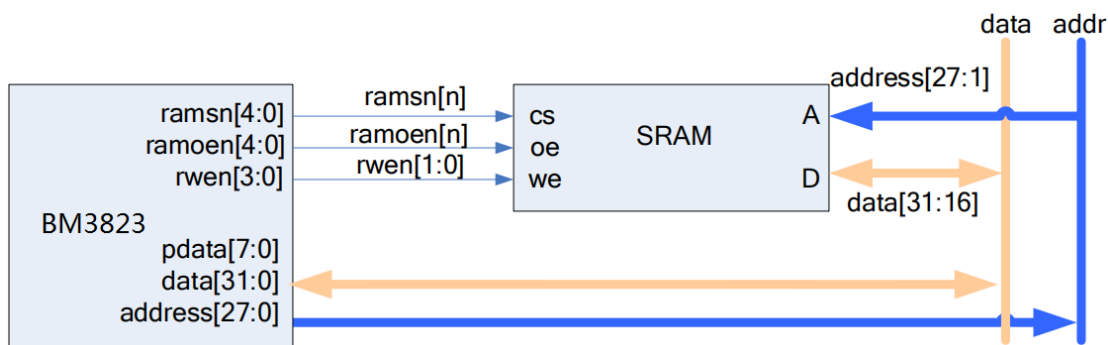


图4-10-2 16位宽SRAM连接示意图

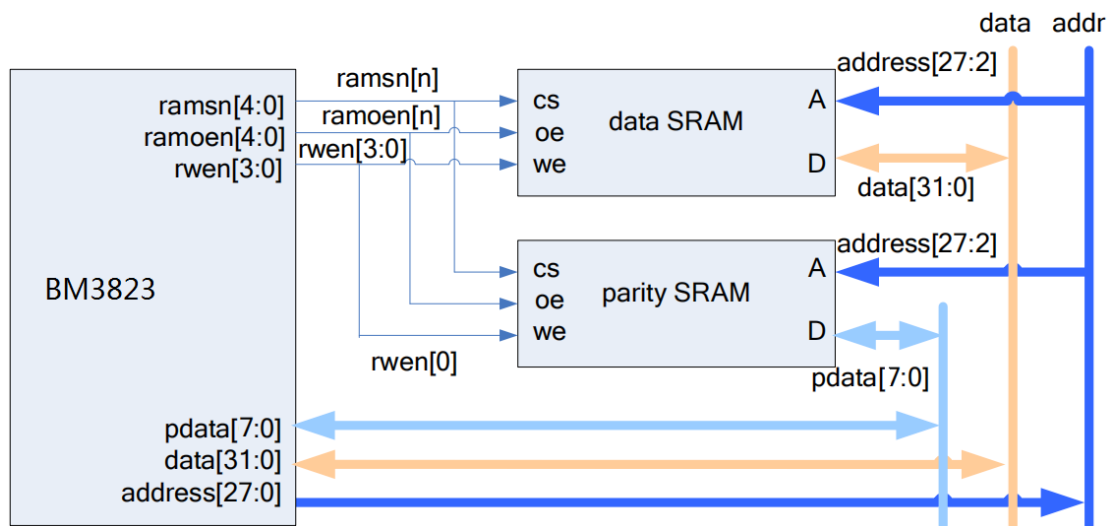


图4-10-3 32位宽SRAM连接示意图

SRAM 写操作分“读改写”模式和“写选通”模式两种，通过配置寄存器 MCFG2[6] 进行配置。

MCFG2[6]配置为高时，SRAM 写操作为“读改写”模式，rwen[3:0]信号会同时有效，或同时无效，当 SRAM 配置为 32 位数据宽度时进行字节或半字写访问，先读字节或半字所在的字，然后控制器在片内改写字内的字节或半字，再把整个字写到 SRAM 中；当 SRAM 配置为 16 位数据宽度时进行字节写访问时，先读字节所在的半字，然后控制器在片内改写半字内的字节，再把整个半字写到 SRAM 中。

MCFG2[6]配置为低时，SRAM 写操作为“写选通”模式，rwen[3:0]信号会分别有效或无效，当出现字节或半字写访问时，字节或半字对应的 rwen 信号会有效，其它 rwen 信号会无效。

rwen[3:0]与 data[31:0]对应关系，见表 4-10-12。

表4-10-12 SRAM写信号与数据字节对应表

SRAM 写信号	对应数据字节
rwen[0]	data[31:24]
rwen[1]	data[23:16]
rwen[2]	data[15:8]
rwen[3]	data[7:0]

对于第五个 SRAM，还可以通过 brdyn 管脚，来延迟适配读写访问时间。第五个 SRAM 对 brdyn 管脚的使用需要通过配置寄存器 MCFG2[7]进行配置，当 MCFG2[7]配置为“1”时，SRAM 读写结束前，需要 brdyn 管脚为低电平。

SRAM 的读写访问时序，见图 4-10-4、图 4-10-5、图 4-10-6、图 4-10-7、图 4-10-8。

SRAM 读访问时序中“lead_out”周期中的地址值，不保持上一个时钟周期（图中“data2”周期）中的地址值，而是上一个时钟周期（图中“data2”周期）中处理器内部总线地址值。

SRAM 的读访问时序如下图所示：

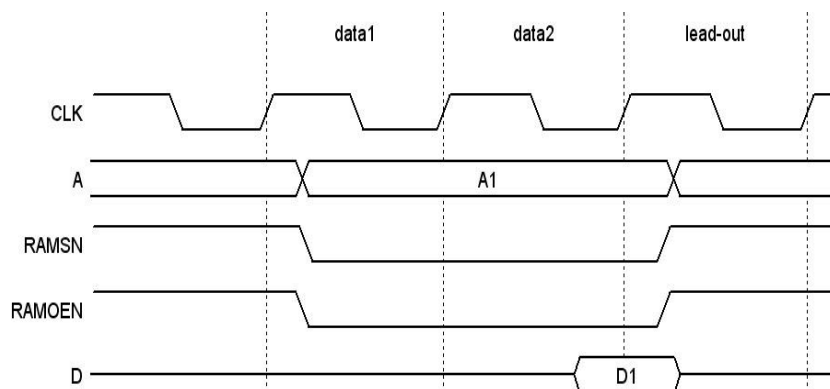


图4-10-4 SRAM读访问时序（EDAC不使能，0等待周期）

SRAM 读访问，EDAC 不使能时，在片选从有效变为无效前的时钟上升沿采集数据。

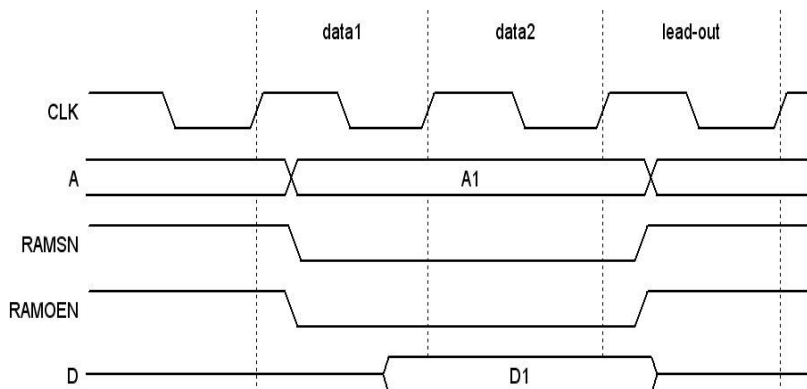


图4-10-5 SRAM读访问时序（EDAC使能，0等待周期）

SRAM 读访问，EDAC 使能时比不使能时，早一个时钟周期采集数据。

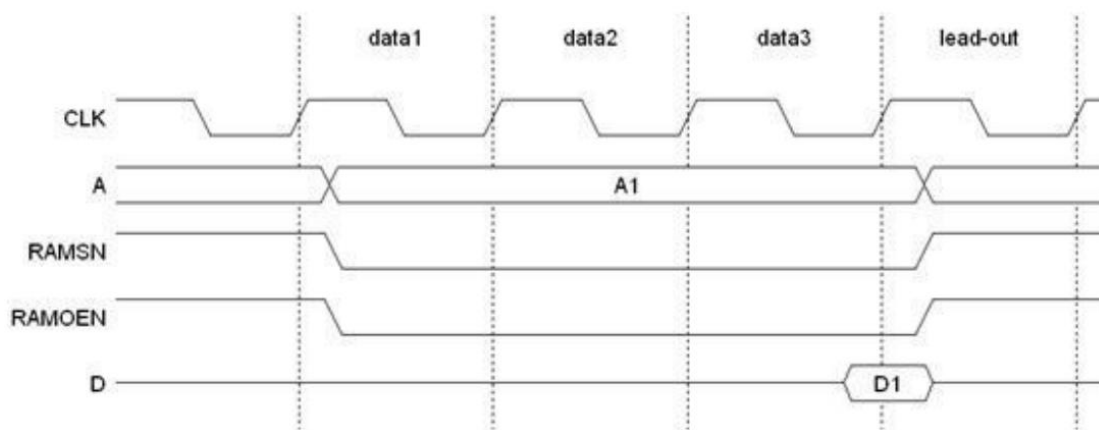


图4-10-6 SRAM读访问时序（EDAC不使能，1等待周期）

SRAM 的写访问时序如下图所示：

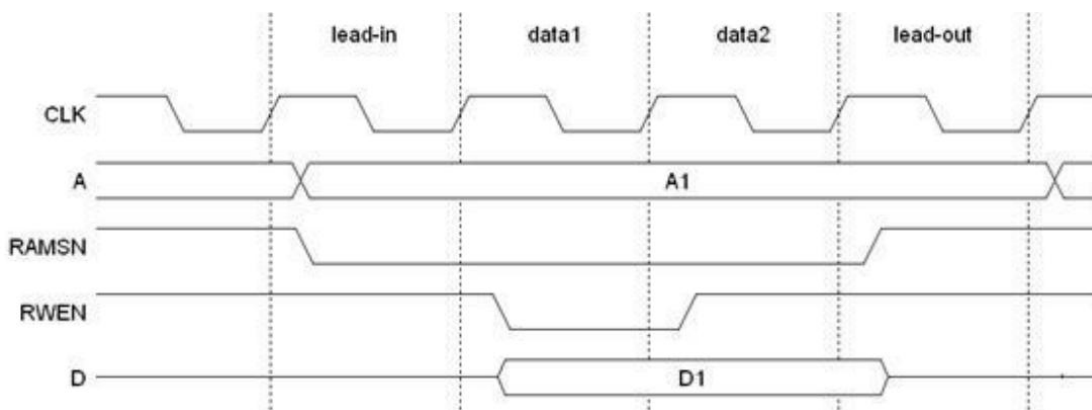


图4-10-7 SRAM写访问时序（0等待周期）

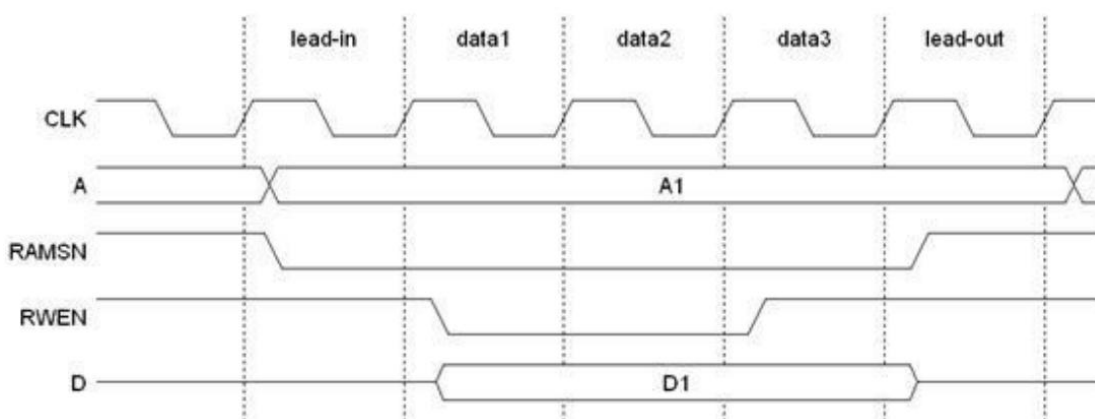


图4-10-8 SRAM写访问时序（1等待周期）

4.10.3.2 SDRAM 接口

同步动态随机存储器接口支持 2 个片选，SDRAM 控制信号包括：片选 $\text{sdcsn}[1:0]$ 、行地址选择 sdrasn 、列地址选择 sdcasn 、写使能 sdwen 、数据屏蔽 $\text{sddqm}[3:0]$ 和时钟 sdclk 。每个 SDRAM 片选容量通过配置寄存器 $\text{MCFG2}[25:23]$ 配置，范围按照两倍递进从 4M 到 512M 字节。控制器支持 64M、256M 和 512M 的器件，这些器件列地址范围从 8 到 12 位，行地址位最大 13，片内 4 个块。SDRAM 只支持 32 位数据总线。

SDRAM 存储体的地址管脚应连接到处理器的 $\text{address}[14:2]$ ，SDRAM 存储器的块地址应连接到处理器的 $\text{address}[16:15]$ 。存储体地址管脚少于 13 位时，连接处理器管脚 $\text{address}[14:2]$ 的低位。

SDRAM 存储体的数据管脚应连接到处理器的 $\text{data}[31:0]$ ，如果 SDRAM 使用 EDAC 功

能，处理器的校验位管脚 pdata[7:0]连接 SDRAM 校验位存储体。

SDRAM 连接示意图见图 4-10-9。

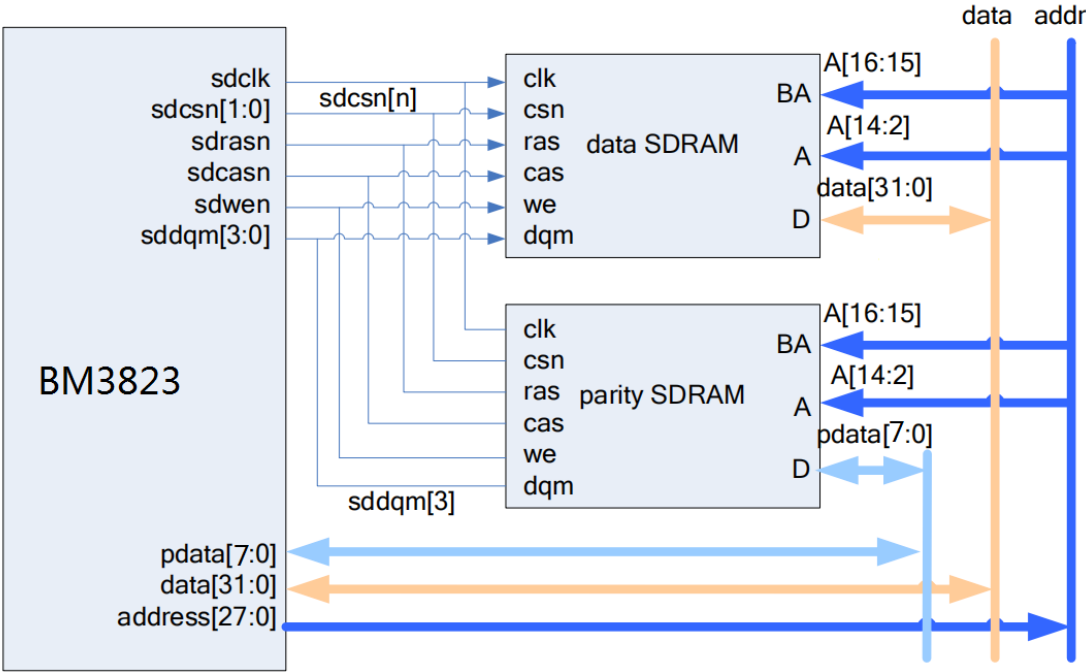


图4-10-9 SDRAM连接示意图

数据屏蔽 sddqm 管脚，也叫数据选通管脚，“1”表示屏蔽，“0”表示选通。sddqm[3:0]与 data[31:0]对应关系，见表 4-10-13。

表4-10-13 SDRAM数据屏蔽与数据字节对应表

数据屏蔽信号	对应数据字节
sddqm[3]	data[31:24]
sddqm[2]	data[23:16]
sddqm[1]	data[15:8]
sddqm[0]	data[7:0]

4.10.3.2.1 SDRAM 时序参数的配置

根据不同的 SDRAM 器件，通过配置寄存器 MCFG2、MCFG3 配置 SDRAM 时序参数见表 4-10-14。

表4-10-14 SDRAM可编程时序参数表

功能	参数	范围	单位
CAS延迟	T_{CAS}	2-3	时钟周期
预充	T_{RP}	2-3	时钟周期
自动更新命令周期	T_{RFC}	3-11	时钟周期
自动更新间隔	$T_{REFRESH}$	10-32768	时钟周期

4.10.3.2.2 SDRAM 命令

SDRAM 控制器能够发送三种 SDRAM 命令。通过对配置寄存器 MCFG2[20:19]写入非零值来发送 SDRAM 命令：

“01”：发送预充命令；

“10”：发送自动刷新命令；

“11”：发送加载模式寄存器 LMR 命令。

每个命令执行完后，MCFG2[20:19]自动清零。

通过执行 LMR 命令，为 SDRAM 存储体配置访问模式，包括 CAS 延时周期和其它固定模式。其它固定模式为全页 burst 读、单一写、顺序 burst。CAS 延时周期会按照配置寄存器 MCFG2[26]，即 SdrCAS，为“0”，对应 CAS 延时周期为 2，为“1”，对应 CAS 延时周期为 3。如果要修改 CAS 延时配置，必须执行 LMR 命令。

执行 LMR 命令时处理器地址、输出值以及它们与 SDRAM 存储器的地址、模式寄存器位对应关系见表 4-10-15。

表4-10-15 执行LMR命令时地址管脚情况

SDRAM		处理器	
地址	模式寄存器	地址	输出值
BA1	14	A16	0
BA0	13	A15	0
A12	12	A14	0
A11	11	A13	0
A10	10	A12	0

SDRAM		处理器	
地址	模式寄存器	地址	输出值
A9	9	A11	1
A8	8	A10	0
A7	7	A9	0
A6	6	A8	0
A5	5	A7	1
A4	4	A6	SDRCAS
A3	3	A5	0
A2	2	A4	1
A1	1	A3	1
A0	0	A2	1

控制器提供周期刷新功能。通过对 MCFG2[31]刷新使能位（SDRREF）配置为“1”来使能周期刷新功能。刷新命令会对两个 SDRAM 片选同时发送。通过 MCFG3[26:12]刷新计数器重载值配置两个自动刷新命令之间的间隔周期。

根据 SDRAM 类型，自动刷新命令间隔周期为 7.8 μ s 或 15.6 μ s，即对应 100MHz 频率下的 780 或 1560 个时钟周期。

自动刷新命令间隔周期计算公式如下：

刷新间隔周期=（重载值+1）/时钟频率

4.10.3.2.3 SDRAM 初始化

通过对 MCFG2[14]配置为“1”来使能 SDRAM 后，同时对两个 SDRAM 片选自动执行 SDRAM 初始化序列命令：依次执行预充命令、两个自动刷新命令和 LMR 命令。

4.10.3.2.4 SDRAM 读访问

读周期包括三个步骤：

第一步：对所需的块和行执行 ACTIVE 命令；

第二步：经过 CAS 延时后，执行 READ 命令；

第三步：以预充命令结束读周期。

一次访问结束后，会关闭当前行，如果内部总线发出突发模式访问请求则执行突发模式读操作。

SDRAM 读时序 (Burstlength=1;CL=3)，见图 4-10-10。

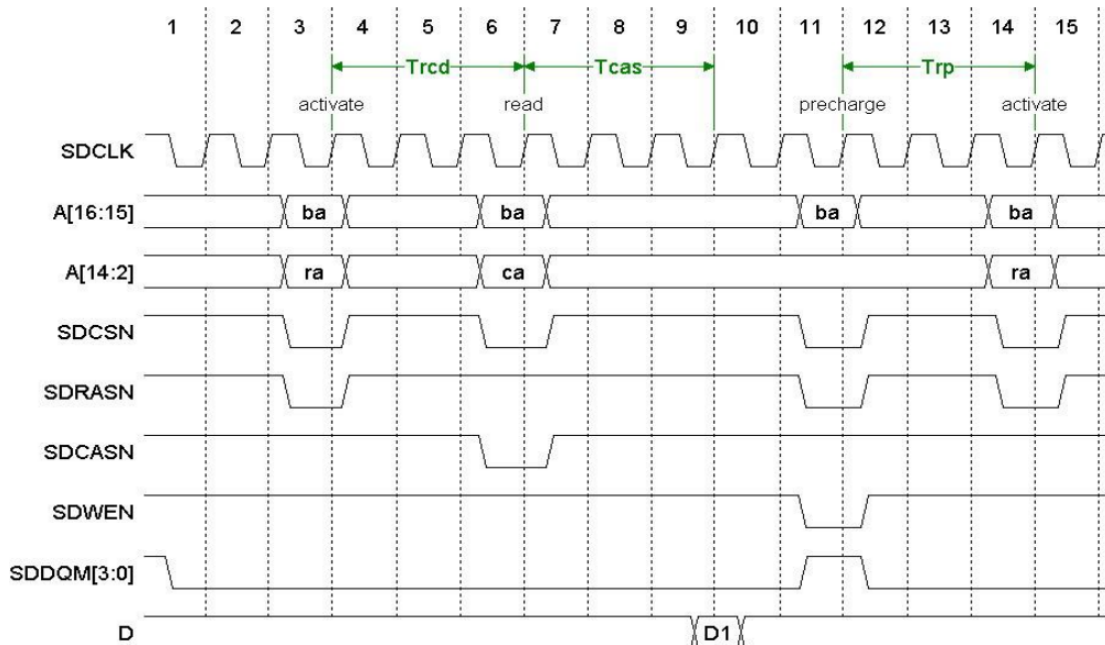


图4-10-10 SDRAM读访问时序

4.10.3.2.5 SDRAM 写访问

写周期包括三个步骤：

第一步：对所需的块和行执行 ACTIVE 命令；

第二步：经过 CAS 延时后，执行 WRITE 命令；

第三步：以预充命令结束写周期。

内部总线的突发写操作会产生没有空闲周期的突发写命令。

SDRAM 写时序 (Burstlength=1;CL=3)，见图 4-10-11。

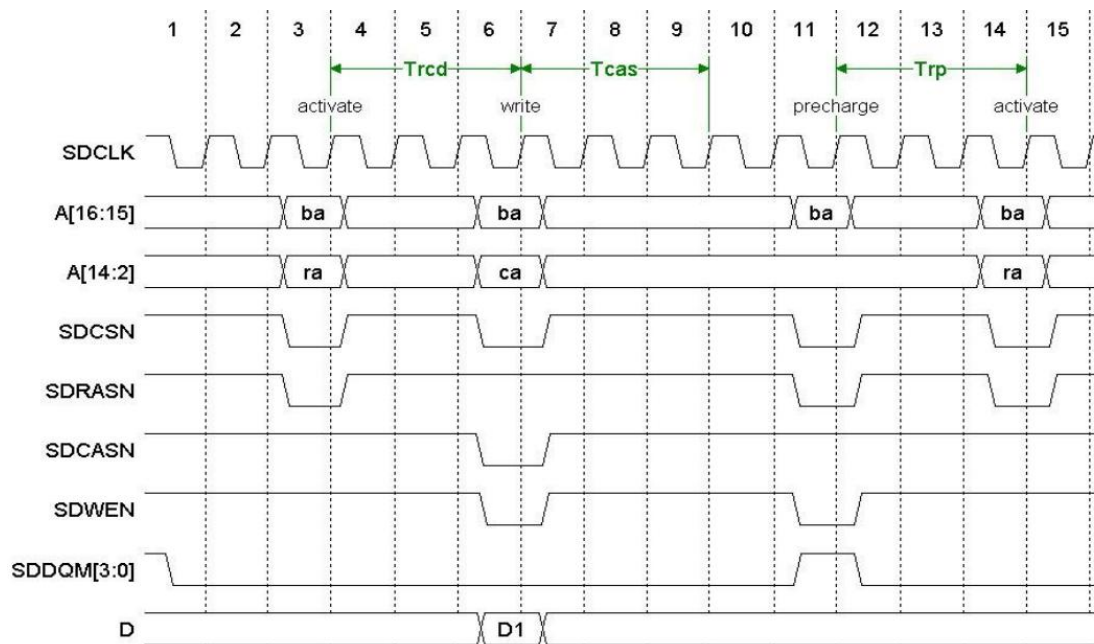


图4-10-11 SDRAM写访问时序

4.10.4 PROM 接口

PROM 接口支持 2 个 PROM 片选，每个片选容量无需配置，每个片选容量最大 128M 字节。PROM 控制信号包括：片选 romsn[1:0]、读使能 oen、写使能 writen。

通过配置寄存器 MCFG1[16:12]配置 PROM 写等待周期，通过配置寄存器 MCFG1[4:0]配置 PROM 读等待周期，等待周期配置范围为 0 到 31。通过配置寄存器 MCFG1[11]配置 PROM 的写使能。通过配置寄存器 MCFG1[9:8]配置 PROM 的数据宽度。

当 PROM 配置为 8 位数据宽度时，PROM 存储体数据管脚连接处理器的数据管脚为 data[31:24]，PROM 存储体地址管脚最低位连接处理器的地址管脚为 address[0]。8 位数据宽度 PROM 的 EDAC 功能同管脚 pdata[7:0]无关。

当 PROM 配置为 16 位数据宽度时，PROM 存储体数据管脚连接处理器的数据管脚为 data[31:16]，PROM 存储体地址管脚最低位连接处理器的地址管脚为 address[1]。

当 PROM 配置为 32 位数据宽度时，PROM 存储体数据管脚连接处理器的数据管脚为 data[31:0]，PROM 存储体地址管脚最低位连接处理器的地址管脚为 address[2]。如果 32 位数据宽度 PROM 使用 EDAC 功能，处理器的校验位管脚 pdata[7:0]连接 PROM 校验位存储体。

PROM 连接示意图见图 4-10-12、图 4-10-13、图 4-10-14。

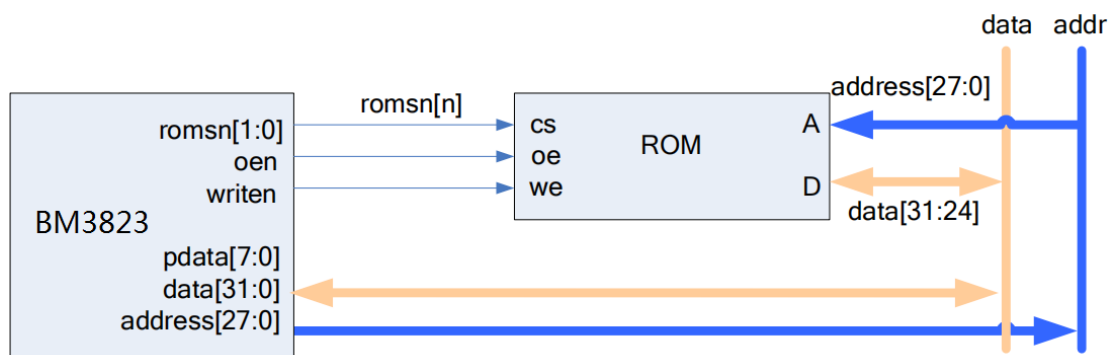


图4-10-12 8位宽PROM连接示意图

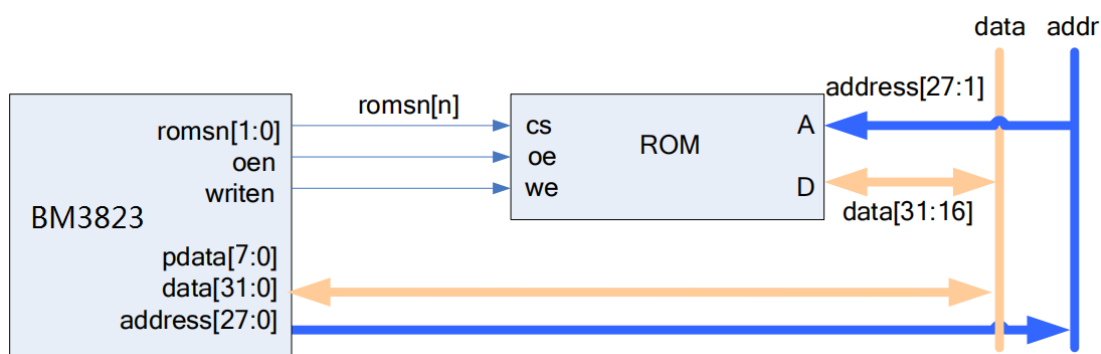


图4-10-13 16位宽PROM连接示意图

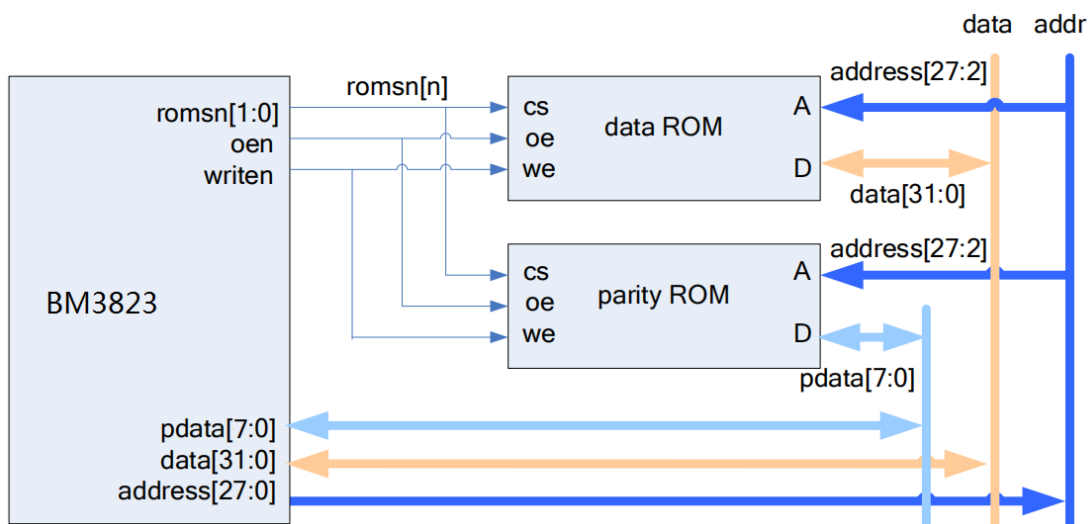


图4-10-14 32位宽PROM连接示意图

PROM 的读写访问接口时序，见图 4-10-15、图 4-10-16、图 4-10-17、图 4-10-18、图 4-10-19。

PROM 读访问时序中“lead_out”周期中的地址值，不保持上一个时钟周期（图中“data2”

周期)中的地址值,而是上一个时钟周期(图中“data2”周期)中处理器内部总线地址值。

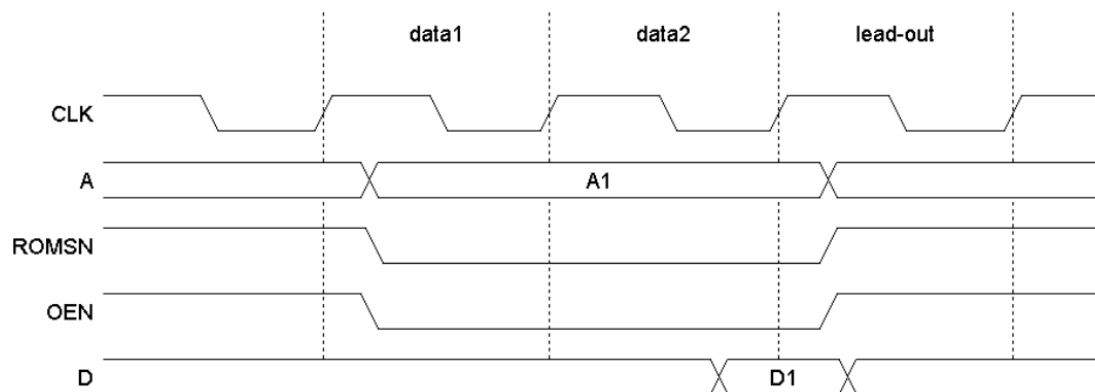


图4-10-15 PROM读访问时序 (EDAC不使能, 0个等待周期)

PROM 读访问, EDAC 不使能时, 在片选从有效变为无效前的时钟上升沿采集数据。

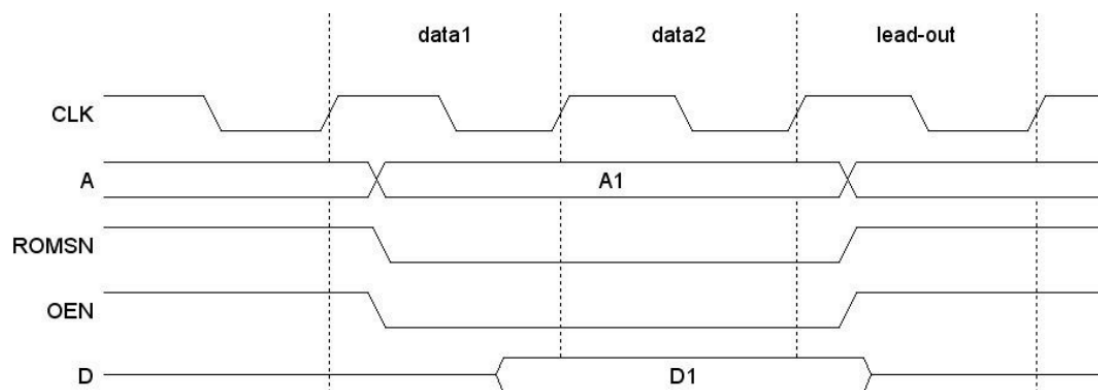


图4-10-16 PROM读访问时序 (EDAC使能, 0等待周期)

PROM 读访问, EDAC 使能时比不使能时, 早一个时钟周期采集数据。

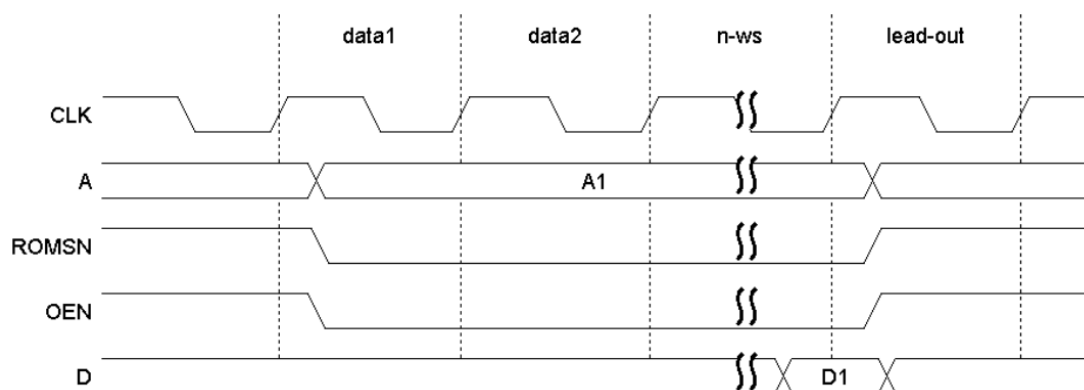


图4-10-17 PROM读访问时序 (EDAC不使能, n个等待周期)

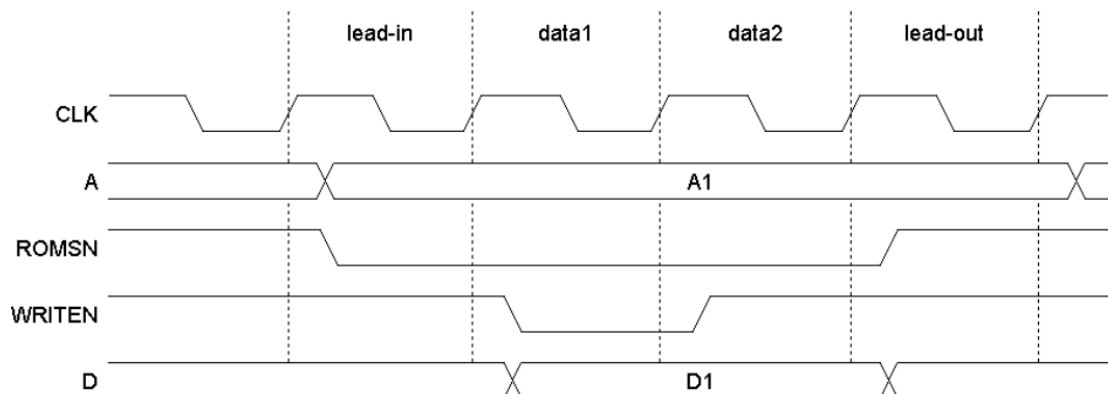


图4-10-18 PROM写访问时序（0个等待周期）

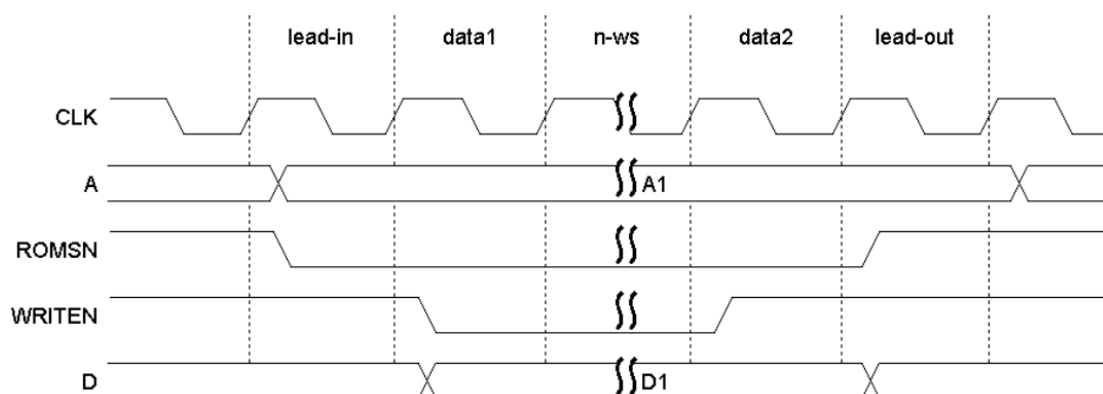


图4-10-19 PROM写访问时序（n个等待周期）

4.10.5 I/O 空间接口

I/O 空间设备接口支持 6 个片选。I/O 设备地址范围是 0x1000-0000~0x1FFF-FFFF。I/O 设备访问控制信号包括片选 iosn、读使能 oen、写使能 writen。

通过配置寄存器 MCFG1[28:27]配置 I/O 设备数据总线的宽度（“00”=8，“01”=16，“1x”=32）。

当 I/O 设备配置为 8 位数据宽度时，I/O 设备数据管脚连接处理器的数据管脚为 data[31:24]，I/O 设备地址管脚最低位连接处理器的地址管脚为 address[0]。

当 I/O 设备配置为 16 位数据宽度时，I/O 设备数据管脚连接处理器的数据管脚为 data[31:16]，I/O 设备地址管脚最低位连接处理器的地址管脚为 address[1]。

当 I/O 设备配置为 32 位数据宽度时，I/O 设备数据管脚连接处理器的数据管脚为 data[31:0]，I/O 设备地址管脚最低位连接处理器的地址管脚为 address[2]。

I/O 设备数据总线宽度配置为 8、16 位，跟配置为 32 位的读写访问时序相同，区别仅仅是数据总线宽度不同。

I/O 设备数据总线宽度配置为 8 位时，只支持 LDUB 和 STB 指令。

I/O 设备数据总线宽度配置为 16 位时，只支持 LDUH 和 STH 指令。

I/O 设备连接示意图见图 4-10-20、图 4-10-21、图 4-10-22。

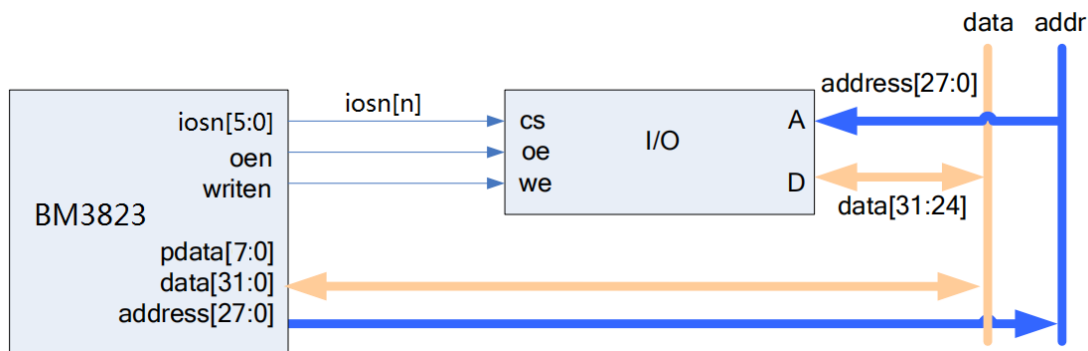


图4-10-20 8位宽I/O设备连接示意图

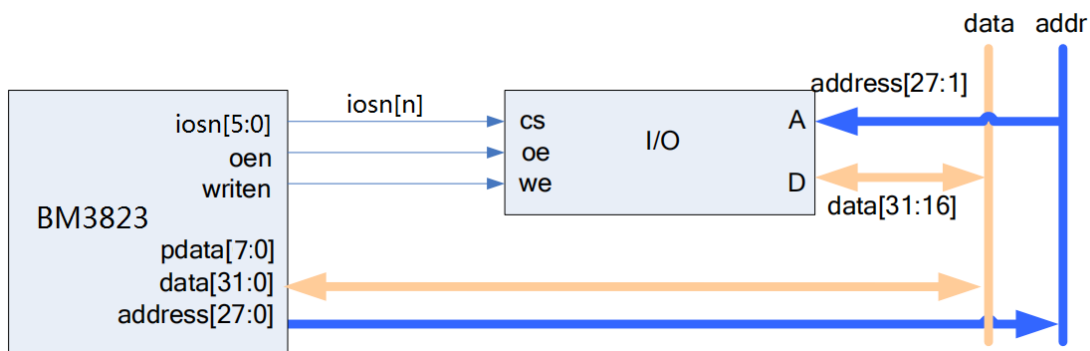


图4-10-21 16位宽I/O设备连接示意图

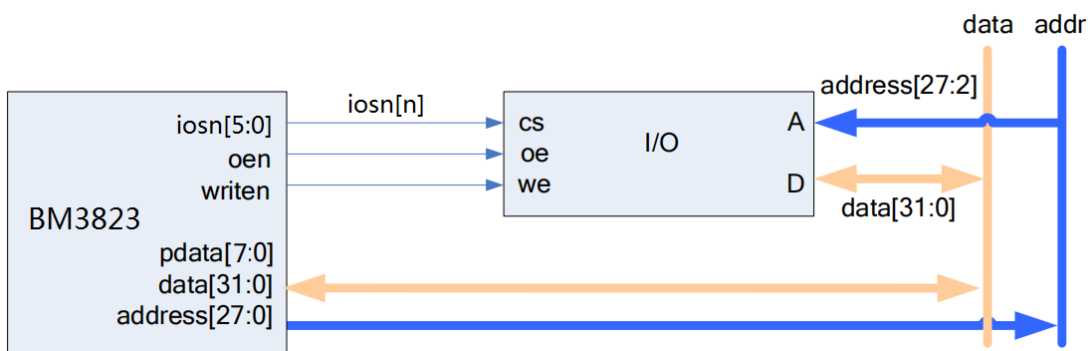


图4-10-22 32位宽I/O设备连接示意图

通过配置寄存器 MCFG1[28:27]配置数据总线的宽度，MCFG1[26]配置 I/O 设备 BRDYN

使能，MCFG1[24:20]配置读写等待周期，即读使能周期和写使能周期，MCFG1[19]配置 I/O 设备使能。I/O 设备访问时序，见图 4-10-23、图 4-10-24。

I/O 空间访问时序如下图所示：

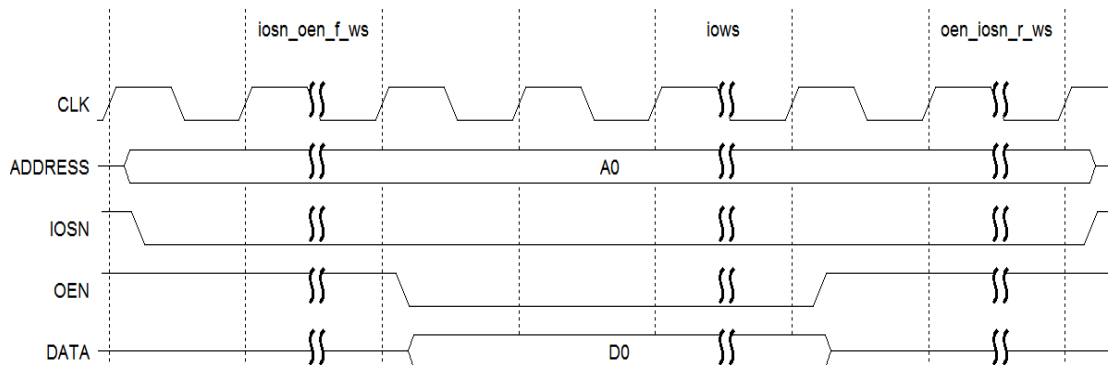


图4-10-23 I/O读访问时序（n等待周期）

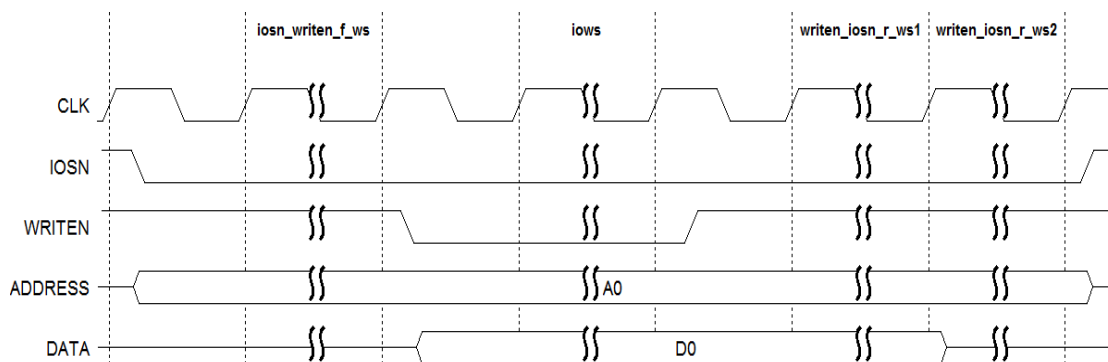


图4-10-24 I/O写访问时序（n等待周期）

图 4-10-23 中的 iosn_oen_f_ws 和 oen_iosn_r_ws 两个值在 MCFG3 中进行配置；图 4-10-24 中的 iows 在 MCFG1 中进行配置，iosn_writen_f_ws 和 writen_iosn_r_ws 在 MCFG3 中进行配置，writen_iosn_r_ws1 和 writen_iosn_r_ws2 符合如下公式：

当 writen_iosn_r_ws 为奇数时：

$$\text{writen\_iosn\_r\_ws1} = \frac{\text{writen\_iosn\_r\_ws} - 1}{2}$$

$$\text{writen\_iosn\_r\_ws2} = \frac{\text{writen\_iosn\_r\_ws} + 1}{2}$$

当 writen_iosn_r_ws 为偶数时：

$$\text{writen\_iosn\_r\_ws1} = \frac{\text{writen\_iosn\_r\_ws}}{2} - 1$$

$$\text{writen\_iosn\_r\_ws2} = \frac{\text{writen\_iosn\_r\_ws}}{2} + 1$$

当 `writen_iosn_r_ws` 为 0 时：

`writen_iosn_r_ws1 = 0, writen_iosn_r_ws2 = 0`

I/O 设备访问的读使能周期和写使能周期可以通过 `MCFG1[23:20]IOWS` 配置位来配置，最大可以配置为“15”，如果通过配置 `IOWS` 最大“15”仍不能满足 I/O 空间设备访问，可以使用 `BRDYN` 功能。I/O 空间的 `BRDYN` 功能通过 `MCFG1[26]Brdy` 配置位来配置，“1”表示使能 I/O 空间的 `BRDYN` 功能，“0”表示不使能 I/O 空间的 `BRDYN` 功能。在使能 I/O 空间 `BRDYN` 功能情况下，I/O 空间读写访问时，经过 `IOWS` 配置的等待周期后，采集 `brdyn` 管脚，如果采集到 `brdyn` 管脚为低电平后，结束等待，且在 `brdyn` 管脚为低电平后一个时钟周期后，读写信号结束。

4.10.6 访问错误

处理器通过 `BEXCN` 信号指示访问错误，适用的外部存储器类型包括 `PROM`、`SRAM`、I/O 设备。如果通过设置 `MCFG1` 中的 `bexc` 位为 1 使能访问错误，接受数据的同时接受 `BEXCN` 管脚信号。如果访问外部设备时 `BEXCN` 信号为低，内部总线会收到错误响应。

若处于取指操作，则进入 `trap 0x01`；

若处于读数据操作，则进入 `trap 0x09`；

若处于写数据操作，则进入 `trap 0x1A`。

4.10.7 错误管理

`BM3823A` 处理器对外部存储器访问具有 `EDAC`（纠错检错）功能，能够纠正 32 位字中的一个错误，能够检测 32 位字中的两个错误。一个错误具体是指 32 位数据位和 7 位校验位总共出现 1 位错误，两个错误具体是指 32 位数据位和 7 位校验位总共出现 2 位错误。检错和纠错时，对数据位和校验位同等对待，当检错时，不论错误出现在数据位或校验位，都检错，当纠正 1 位错误时，不论 1 位错误出现在数据位或校验位都纠正。

处理器对 8 位的 `PROM`、32 位的 `RPOM`、8 位的 `SRAM`、32 位的 `SRAM` 和 `SDRAM` 存储器访问具有 `EDAC` 功能，需要特别注意的是：（1）对于 8 位宽 `PROM`，当使能 `EDAC` 功能时，`bank0` 最大支持 128MB，`bank1` 最大支持 32MB；（2）对于 8 位宽 `SRAM`，仅有 `bank0` 和 `bank4` 支持 `EDAC` 功能。

32 位的 `PROM`、32 位的 `SRAM` 和 32 位的 `SDRAM` 实现 `EDAC` 功能需要分别搭配 8 位

的 PROM、8 位的 SRAM 和 8 位的 SDRAM 校验位存储器。

8 位的 PROM 和 8 位的 SRAM 实现 EDAC 功能不需要搭配额外的校验位存储器，而是数据位和校验位存放在同一个存储体当中，且校验位地址相比校验位对应的数据位地址是倒序排列的，即：数据位在存储体中从前向后依次放置，对应的 7 位校验位（占一个字节）从存储体最后一个字节位置开始依次向前存放。注意不要在存放校验位的存储空间部分存放数据，避免造成数据和校验位的冲突。

4.10.7.1 EDAC 编码匹配注意事项说明

对 8 位 SRAM、32 位 SRAM 和 32 位 SDRAM 的访问，打开 EDAC 使能前，要保证存储器中数据位和校验位的 EDAC 编码匹配，避免由于字节或半字写操作，造成存储器中数据位和校验位的 EDAC 编码不匹配。

4.10.7.2 EDAC 编码

32 位数据位对应 7 位校验位生成公式如下：

$$PD0 = D0 \wedge D4 \wedge D6 \wedge D7 \wedge D8 \wedge D9 \wedge D11 \wedge D14 \wedge D17 \wedge D18 \wedge D19 \wedge D21 \wedge D26 \wedge D28 \wedge D29 \wedge D31。$$
$$PD1 = D0 \wedge D1 \wedge D2 \wedge D4 \wedge D6 \wedge D8 \wedge D10 \wedge D12 \wedge D16 \wedge D17 \wedge D18 \wedge D20 \wedge D22 \wedge D24 \wedge D26 \wedge D28。$$
$$\sim PD2 = D0 \wedge D3 \wedge D4 \wedge D7 \wedge D9 \wedge D10 \wedge D13 \wedge D15 \wedge D16 \wedge D19 \wedge D20 \wedge D23 \wedge D25 \wedge D26 \wedge D29 \wedge D31。$$
$$\sim PD3 = D0 \wedge D1 \wedge D5 \wedge D6 \wedge D7 \wedge D11 \wedge D12 \wedge D13 \wedge D16 \wedge D17 \wedge D21 \wedge D22 \wedge D23 \wedge D27 \wedge D28 \wedge D29。$$
$$PD4 = D2 \wedge D3 \wedge D4 \wedge D5 \wedge D6 \wedge D7 \wedge D14 \wedge D15 \wedge D18 \wedge D19 \wedge D20 \wedge D21 \wedge D22 \wedge D23 \wedge D30 \wedge D31。$$
$$PD5 = D8 \wedge D9 \wedge D10 \wedge D11 \wedge D12 \wedge D13 \wedge D14 \wedge D15 \wedge D24 \wedge D25 \wedge D26 \wedge D27 \wedge D28 \wedge D29 \wedge D30 \wedge D31。$$
$$PD6 = D0 \wedge D1 \wedge D2 \wedge D3 \wedge D4 \wedge D5 \wedge D6 \wedge D7 \wedge D24 \wedge D25 \wedge D26 \wedge D27 \wedge D28 \wedge D29 \wedge D30 \wedge D31。$$

公式中“~”表示“非”，“^”表示“异或”。

4.10.8 写保护

4.10.8.1 概述

写保护功能对 RAM 空间(0x2000-0000 开始到 0x3FFF-FFFF)的写操作进行保护。

4.10.8.2 功能说明

支持两个写保护寄存器，**写保护寄存器 WPR1**（0x8100-001C）和**写保护寄存器 WPR2**（0x8100-0020），分别对不同的地址空间进行写保护。

WPR[31]，即 *EN*，“1”表示使能，“0”表示不使能。

WPR[30]，即 *BP*，“1”表示 block 保护模式，“0”表示 segment 保护模式。

WPR[29]，保留。

WPR[28:15]，即 *TAG*，比较地址。

WPR[14]，保留。

WPR[13:0]，即 *MASK*，掩码地址。

对 RAM 进行写操作，地址 ADDR[29:15]同 *TAG* 进行“异或”，再同 *MASK* 进行“与”，结果为 *result*。

满足写操作被保护的条件：

（1）当两个写保护单元只有一个使能，则配置使能的写保护单元，满足写操作被保护的条件是：

- *BP* 配置为“1”（block 模式）时，如果 *result* 为零，则写操作被保护。
- *BP* 配置为“0”（segment 模式）时，如果 *result* 不为零，则写操作被保护。

表4-10-16 只有1个写保护单元使能时的写保护条件

<i>BP</i> 配置	写操作被保护的条件
“1”（block 模式）	<i>result</i> 为零
“0”（segment 模式）	<i>result</i> 不为零

（2）当两个写保护单元都配置使能，则满足写操作被保护的条件是：

- 如果两个单元都是 block 模式，则任何一个单元的 *result* 为零，则写操作被保护。
- 如果两个单元都是 segment 模式，则两个单元的 *result* 都不为零，则写操作被保护。

- 如果两个单元分别配置为 block 模式和 segment 模式，则配置为 segment 模式的单元的 result 不为零，则写操作被保护。

表4-10-17 两个写保护单元都使能时的写保护条件

保护单元 1 的 BP 配置	保护单元 2 的 BP 配置	写操作被保护的条件
“1” (block 模式)	“1” (block 模式)	result1 为零, 或者 result2 为零
“0” (segment 模式)	“0” (segment 模式)	result1 不为零, 并且 result2 不为零
“1” (block 模式)	“0” (segment 模式)	result2 不为零
“0” (segment 模式)	“1” (block 模式)	result1 不为零

当写操作被保护时，写操作地址值会保存到 AHB 失效地址寄存器 (0x8100-000C)，写操作状态会保存到 AHB 状态寄存器 (0x8100-0010)。

4.11 GPIO

4.11.1 概述

通用接口 GPIO 实现了 32 路供用户自定义使用的双向 IO 端口，其中 12 路 GPIO 端口采用功能复用设计，当它们全复用时，其余 20 路可作用户定义的 I/O；不复用时 32 路 GPIO 均可用作 IO。32 路 GPIO 均可独立配置为输入或输出。

4.11.2 数据模式和 IO 模式

根据中断的不同，GPIO 可工作在 IO 模式和数据模式，通过配置 **GPIO 输入滤波分频与中断模式寄存器**的模式域 M 设置工作模式，0 为数据模式，1 为 IO 模式。

4.11.2.1 IO 模式

在 IO 模式下，不复用的 IO 接口均可作为通用 IO 使用，每个端口都可读/写且相互独立。当端口配置成输入时，读 **GPIO 输入输出数据寄存器**中 IODATAx (x=0~31) 位，则返回当前端口值；当配置成输出时，若 IODATAx (x=0~31) 位置 1，端口 x 被拉高，若 IODATAx (x=0~31) 位置 0，端口 x 被拉低。

GPIO 端口可以配置成输入，用于捕获外部器件中断，通过设置 **GPIO 中断寄存器**，可支持四个中断：GPIO 中断 1~4。可以配置 GPIO 端口与中断的映射关系，32 路 0-31 中任意一路可以映射为中断 0-3 中的任何一个中断。可独立配置 0-3 号 4 个中断的触发方式，并有

中断使能控制。

通过读/写 **GPIO 输入输出数据寄存器**完成数据读/写操作，通过配置 **GPIO 输入输出方向控制寄存器**完成端口输入/输出方向设置，通过设置 **GPIO 中断寄存器**完成 32 路端口的中断使能和状态配置。**GPIO 中断寄存器**每 8bit 一组，共分 4 组：[7:0]、[15:8]、[23:16]、[31:24]，每组功能包括中断使能位 *EN*，边沿/电平触发方式位 *LE*、极性位 *PL*、端口选择位 *ISEL*，其中端口选择共 5 位信号，用于选择 32 位 GPIO 端口中任意一路相应管脚作为中断输入。

注意

- 1、不要用同一路 GPIO 管脚同时触发多个 GPIO 中断，会给中断处理带来麻烦。
- 2、当 GPIO 中断采用电平触发方式时，以低电平为例，只要相应的 GPIO 端口接收到低电平，就会产生中断请求，处理器响应中断后，会自动清除中断请求，但如果此时 GPIO 端口仍接收到低电平，会再次产生中断请求，此时，用户中断服务程序中须增加清除中断请求操作，即写**中断清除寄存器**相应位，且要确保清除中断请求操作前 GPIO 端口的低电平撤销。

4.11.2.2 数据模式

选择数据模式时，GPIO 接口方向必须配置为输入，此时每 8 路 GPIO 接口对应一个固定的数据状态 GPIO 中断，即 0-7 路对应中断 0，8-15 路对应中断 1，16-23 路对应中断 2，24-31 路对应中断 3。

在数据模式下会对接口输入的数据进行滤波处理，滤波宽度可以通过配置 **GPIO 输入滤波分频与中断模式寄存器**实现，滤波采样时钟周期是滤波分频值加一个主时钟周期。对于每一个状态变化，以 0 变 1 为例，原状态为 0，要连续采样到三次 1，状态才变成 1。例如：主时钟 100MHz，周期为 10ns，滤波分频值为 9，那么滤波采用时钟是 $100\text{MHz}/(9+1)=10\text{MHz}$ ，滤波周期是 $10\text{ns}\times(9+1)=100\text{ns}$ ，采样 3 次为 $100\text{ns}\times3=300\text{ns}$ ，只有每次变化超过 300ns，才能确保变化被 GPIO 电路正确接收。

完成滤波分频配置后，使能滤波功能，**数据寄存器**才开始接收数据，*COUNTER* 开始计数。每 8bit 数据为一组，即 0-7bit 为一组，8-15bit 为一组，16-23bit 为一组，24-31bit 为一组，高 32bit 依次类推，每采样到一组数据状态变化，产生一个数据中断，并将数据缓存，最多可缓存 7 个最新数据，并记录 256 次变化。

数据寄存器 0 中包含数据 2、数据 1、数据 0、计数值，数据寄存器 1 中包含数据 6、数据 5、数据 4、数据 3，其中数据 0 为最新数据，数据 1 次之，依次类推。COUNTER 具有读清功能，即在读取完 COUNTER 的计数值之后，COUNTER 值清零。

注意 当读清操作与数据变化同时发生时，读清优先级高，COUNTER 清零；当前数据变化被检测后，新数据存入缓存队列，不产生中断，COUNTER 计数值不能更新。

4.11.3 复用功能

0-15 低 16 位 GPIO 除了作为通用 I/O，部分端口可复用为其他功能。处于输入状态时，管脚上的信号同时送入 GPIO 模块和复用功能模块。处于输出状态时，若复用功能使能，输出到管脚上的信号来自于复用功能模块，若复用功能不使能，输出管脚上的信号来自于 GPIO 模块。复用功能如表 4-11-1 所示：

表4-11-1 GPIO管脚复用说明

GPIO 管脚	复用管脚	类型	复用管脚说明	复用管脚使能条件
pio[15]	txd1	O	UART1 发送数据	UART1 发送使能
pio[14]	rx1d	I	UART1 接收数据	UART1 接收使能
pio[13]	rts1	O	UART1 请求发送	UART1 流控使能
pio[12]	cts1	I	UART1 清除发送	UART1 流控使能
pio[11]	txd2	O	UART2 发送数据	UART2 发送使能
pio[10]	rx2d	I	UART2 接收数据	UART2 接收使能
pio[9]	rts2	O	UART2 请求发送	UART2 流控使能
pio[8]	cts2	I	UART2 清除发送	UART2 流控使能
pio[3]	uart clk	I	可选 UART 时钟	串口外部时钟使能
pio[2]	PROM EDAC	I	PROM EDAC 使能	上电复位时，该引脚值会配置给 MECFG1 的 romEEn，为 1 表示 PROM EDAC 使能，为 0 表示 PROM EDAC 不使能。复位后该管脚作为正常 IO 使用。
pio[1:0]	PROM Width	I	定义启动 PROM 宽度	从外部 PROM 启动时，通过设置 pio[1:0]来配置 PROM 存储器的数据宽度。00:8bit, 01:16bit, 1x:32bit。启动后该管脚作为正常 IO 使用。

4.11.4 GPIO 寄存器地址映射

GPIO 寄存器地址映射见表 4-11-2，详细说明见附录 3.6。

表4-11-2 GPIO寄存器地址映射表

寄存器名称	地址	复位值 ^注
GPIO 输入输出数据寄存器	0x81000400	0xFFFFFFFF
GPIO 输入输出方向控制寄存器	0x81000408	0x00000000
GPIO 输入滤波分频与中断模式寄存器	0x81000410	0x00000000
GPIO 滤波时钟使能寄存器	0x81000414	0x00000000
GPIO 中断寄存器	0x81000418	0xFFFFFFFF
0 路数据寄存器 0	0x81000420	0x00000000
0 路数据寄存器 1	0x81000424	0x00000000
1 路数据寄存器 0	0x81000428	0x00000000
1 路数据寄存器 1	0x8100042C	0x00000000
2 路数据寄存器 0	0x81000430	0x00000000
2 路数据寄存器 1	0x81000434	0x00000000
3 路数据寄存器 0	0x81000438	0x00000000
3 路数据寄存器 1	0x8100043C	0x00000000

注：“复位值”按照十六进制统一写法“0xXXX0XXX”，“0x”表示 16 进制，每 4bit 中有 1bit 不定值，即写为“X”，4bit 全部为固定值则写为相应 16 进制数值。

4.12 SPI 总线控制器

4.12.1 概述

BM3823A 有 2 路 SPI 总线控制器。SPI 总线控制器主要特性包括：（1）时钟频率可编程（0x2 到 0x3ff 分频）；（2）LSB/MSB 工作方式可选；（3）支持时钟有效沿设置；（4）支持时钟空闲电平设置；（5）支持数据位长 8，9，10.....32bits 可配置；（6）支持单线/双线工作模式选择；（7）支持单主多从选择，即每路 SPI 分别可以接 2 个从设备。

SPI 共有 4 根信号线，分别是：串行数据线 MISO（双线模式下主入从出，单线模式下无效），串行数据线 MOSI（双线模式下主出从入，单线模式下为双向数据），时钟线 SPI_CLK，设备选择线 SPI_CSN（BM3823A 每路 SPI 有 2 个片选：SPI_CSN0，SPI_CSN1）。

4.12.2 SPI 控制器的配置

SPI 采用主从模式结构，可以支持多 Slave 模式应用，仅支持单 Master。时钟由 Master 控制，在时钟移位脉冲下，数据按位传输，通过配置寄存器可以选择高位优先传输或者低位优先传输。SPI 有两种工作模式，常用的是双线模式，此时 SPI 数据线是单向的，有两根串行数据线分别用来进行 Master 和 Slave 之间的数据相互传输；另一种是单线模式，此时仅有一根双向数据线 MOSI，MOSI 承担 Master 到 Slave 和 Slave 到 Master 的数据传输。SPI 控制器的时钟及工作模式可进行配置。

◆ 时钟极性选择

SPI 设备使能之后，数据没有进行传输时，SCLK 处于空闲电平，通过配置 **SPI 控制寄存器** 中的空闲状态极性控制位 *CPOL*，可以选择空闲电平是高电平还是低电平。

◆ 时钟相位选择

SPI 控制器可以通过配置 **SPI 控制寄存器** 中的 *CPHA* 控制位来选择数据接收端设备的采样时刻，可以在时钟的奇数边沿进行采样，也可以在时钟的偶数边沿进行采样。

◆ 多 Slave 模式

SPI 支持多 Slave 的工作模式（2 个 Slave），此时两个 Slave 共享时钟线、数据线，可以直接并接在一起，两个 Slave 的片选线则单独与 Master 连接，受 Master 控制。在一段时间之内，Master 只能通过某一片选激活其中一个 Slave 进行数据传输，而此时其他 Slave 的时钟线和数据线端口则都保持高阻状态。

4.12.3 SPI 接口时序描述

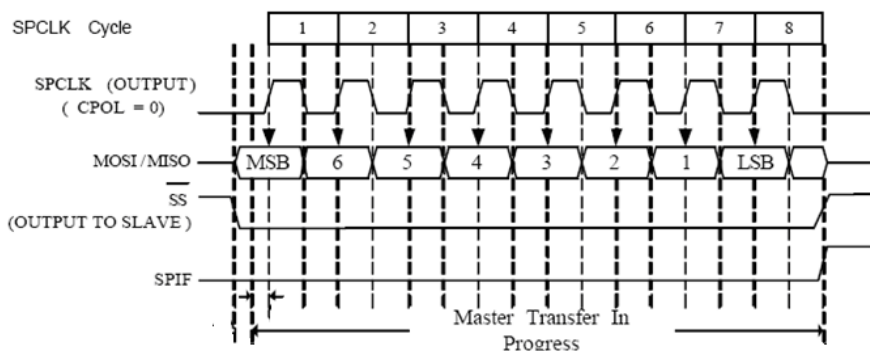


图4-12-1 (a) 主机传送模式 (CPOL=0, CPHA=0)

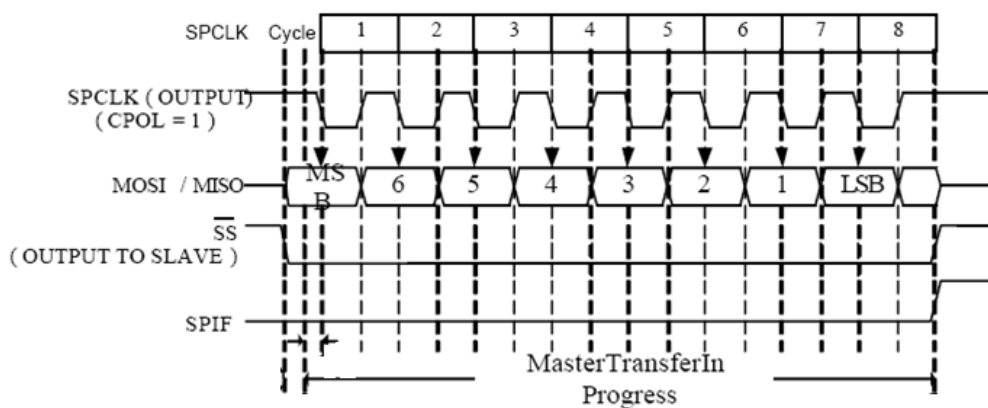


图4-12-1 (b) 主机传送模式 (CPOL=1, CPHA=0)

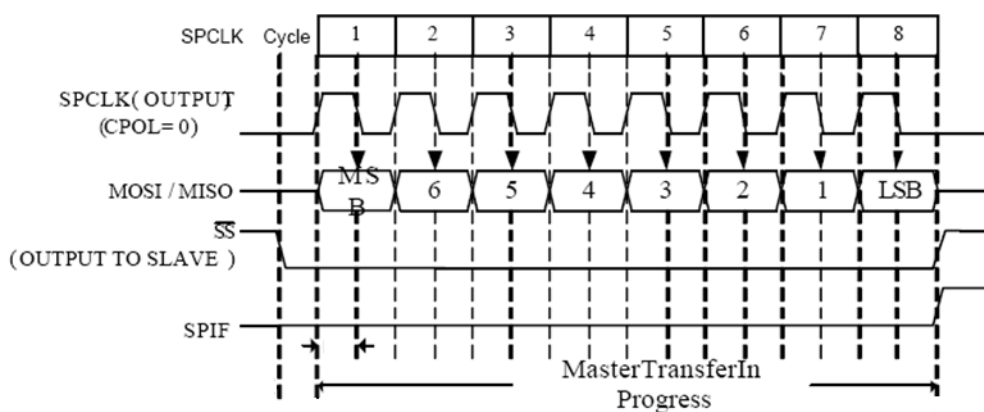


图4-12-1 (c) 主机传送模式 (CPOL=0, CPHA=1)

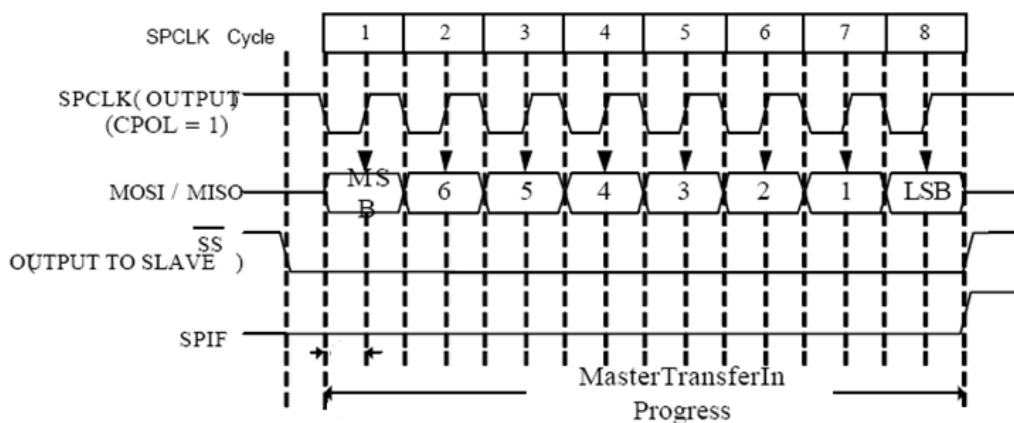


图4-12-1 (d) 主机传送模式 (CPOL=1, CPHA=1)

图4-12-1 SPI接口时序关系图

4.12.4 SPI 寄存器地址映射

SPI 寄存器地址映射见表 4-12-1，详细说明见附录 3.12。

表4-12-1 SPI寄存器地址映射

表4-12-1 (a) SPI0寄存器地址映射

寄存器名称	地址	复位值 ^注
SPI0 控制寄存器	0x81000560	0x00000011
SPI0 状态寄存器	0x81000564	0x00000020
SPI0 数据寄存器	0x81000568	0x00000000
SPI0_CLK 相对主钟分频寄存器	0x8100056C	0x00000006
SPI0 数据位长寄存器	0x81000570	0x00000008
SPI0 片选寄存器	0x81000574	0x00000003

表4-12-1 (b) SPI1寄存器地址映射

寄存器名称	地址	复位值 ^注
SPI1 控制寄存器	0x81000580	0x00000011
SPI1 状态寄存器	0x81000584	0x00000020
SPI1 数据寄存器	0x81000588	0x00000000
SPI1_CLK 相对主钟分频寄存器	0x8100058C	0x00000006
SPI1 数据位长寄存器	0x81000590	0x00000008
SPI1 片选寄存器	0x81000594	0x00000003

注：“复位值”按照十六进制统一写法“0xXXX0XXX”，“0x”表示 16 进制，每 4bit 中有 1bit 不定值，即写为“X”，4bit 全部为固定值则写为相应 16 进制数值。

4.13 I²C 总线控制器

4.13.1 概述

BM3823A 共实现了 2 路主从模式 I²C 总线控制器，串行 8 位双向数据传输速率在快速模式下可达 400KBits/s，高速模式下速率可达 3.4MBits/s（产品双向传输时对从设备 I²C 速率要求高，且波形受板级上拉电阻与负载的容性影响，不易形成方波，有超过 1MBits/s 传输速率要求时建议使用 SPI 替代 I²C）。

I²C 总线控制器主要特性包括：（1）时钟频率可编程；（2）响应应答信号可编程；（3）

支持可自动取消的丢失仲裁中断；（4）支持起始/停止/重复起始/响应条件产生；（5）支持起始/停止/重复起始检测；（6）支持总线忙检测；（7）支持 7 位/10 位地址模式；（8）支持从机模式。

4.13.2 模块功能

通常，标准通讯包括以下四个方面：（1）起始信号的产生；（2）从设备地址传输；（3）数据传输；（4）停止信号的产生。首先生成一个数据传输起始条件，之后依次传输从设备地址和数据，最后产生停止信号，具体见图 4-13-1。

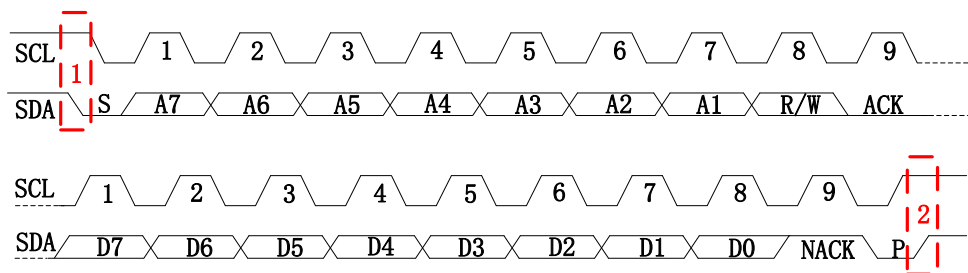


图4-13-1 I²C数据传输通讯示意图

（1）起始信号

在数据线 SDA 和时钟线 SCL 都为高时，总线处在空闲状态，表示没有主设备控制总线，等待各个主设备发送起始信号启动一个数据传输操作。在 SCL 为高时，SDA 由高到低的跳变表示产生一个起始信号，意味着一个新数据传输将开始，见图 4-13-1 中的 1 号红框。如果没有产生停止条件且再次产生一个起始信号，则表示产生了重复起始条件，主机可以寻址另一个从机或有不同的传输方向。

（2）从设备地址传输

主机发出起始信号后，接着传输的第一个字节是从机（7 位地址的 Slave 设备）地址，高 7 位表示从设备地址，最低位 8 位是读/写位（高电平为读，低电平为写）。读/写位确定了从设备数据传输方向。一个系统中从设备的地址都是唯一的，不同的从设备地址不同（具体可以参考从设备的 Datasheet 手册），只有与主机发出的地址相匹配的从设备才会在下一个时钟周期（即第 9 个 SCL）把数据线拉低，作为一个应答信号，通知主机操作有效，之后进入数据传输状态。

（3）数据传输

所有的传输可分为数据读取和写入，不论读取或写入数据，Master 会先发出 7 个比特的地址位，先发送 MSB，最后发送 LSB；之后发出 Write 或 Read 信号，其中高电平表示 Read，低电平表示 Write；然后 Slave 必须发出一个响应信号 ACK，低电平有效。如果是 Write 操作，Master 就把希望传输的数据以 MSB 先、LSB 后的顺序传给 Slave，然后由 Slave 响应一个 ACK；如果是 Read 的话，就由 Slave 响应相对应地址的数据，然后由 Master 响应一个 ACK，最后 Master 发出结束信号。如果 Slave 响应的是 NACK，Master 可以选择结束信号，然后再重传一次作为响应。

当 Slave 没有足够的时间接收数据时，可以把 SCL 拉低，此时时钟信号停止振荡，Master 需要将最后一次传送的比特位再传送一次。如果读取或写入的数据不止一个，只要 Master 不产生停止条件，则数据传输就会继续下去，Slave 需自动把地址加 1。

（4）停止信号

当 SCL 是高电平时，SDA 由低电平向高电平切换时表示停止条件，此时主设备产生停止信号来结束通讯。

4.13.3 数据传输

（1）写 I²C 从设备

I²C 从设备器件 7 位地址 0x22，向该从设备器件写 N 个数据，I²C 写流程命令如下：

- 主设备

- 1) 设置**速率寄存器**；

- 2) 写从机地址 ((slave address) <<1) + Write Command 到**数据寄存器**；设置**控制寄存器**中的 START 位；

- 3) 等待**中断标志寄存器**中的 BYTE_INT 位置位，向数据寄存器写入一个 Bit[3]为“1”的数据，此时进入高速模式，hsmode 信号拉高（如果不需要进入高速模式，此步骤忽略）；

- 4) 等待**中断标志寄存器**中的 BYTE_INT 位置位，将待发送的数据写入**数据寄存器**，重复这个过程，直到最后一个数据（第 N 个）被发送；

- 5) 设置**控制寄存器**中的 STOP 位，完成数据发送。

- 从设备（中断方式）

- 1) 设置**速率寄存器**，使能 I²C **中断标志寄存器**中的 SLAVE_INT；

2) 写 0x0 到从地址寄存器 1, 0x22 到从地址寄存器 0;

3) 设置控制寄存器中的 *ACK_EN* 位为 1;

4) 等待中断标志寄存器中的 *SLAVE_INT* 位置位, 表示 I²C 作为从设备被校验成功, 并且作为从设备发送器, 读取数据寄存器, 可以根据读取数据的最低位确定正确的传输方向(是读还是写);

5) 当中断标志寄存器中的 *BYTE_INT* 位置位时, 读取数据寄存器, 这个过程重复执行, 直到停止或重新开始条件被检测到。

(2) 读 I²C 从设备

确定 I²C 从设备器件地址 0x22, 连续读 N 个数据:

● 主设备

1) 设置控制寄存器中的 *ACK_EN* 位, 设置速率寄存器;

2) 写从机地址 ((slave address) <<1) + Read Command 到数据寄存器; 设置控制寄存器中的 *START* 位;

3) 等待中断标志寄存器中的 *BYTE_INT* 位置位, 当该中断触发时, 读取数据寄存器 (该值不是接收的第一个数据), 清除 *BYTE_INT* 位;

4) 等待中断标志寄存器中的 *BYTE_INT* 位置位, 当该中断触发时, 读取数据寄存器, I²C 会自动发送确认; 重复执行这个过程, 直到接收完第 N-1 个数据为止;

5) 清除控制寄存器中的 *ACK_EN* 位, 一直等待, 直到中断标志寄存器中的 *BYTE_INT* 位置位, 读取数据寄存器 (最后一个数据), I²C 不再发送确认;

6) 设置控制寄存器中的 *STOP* 位, 完成数据接收。

● 从设备 (中断方式)

1) 设置速率寄存器, 使能 I²C 中断标志寄存器中的 *SLAVE_INT*;

2) 写 0x0 到从地址寄存器 1, 0x22 到从地址寄存器 0;

3) 设置控制寄存器中的 *ACK_EN* 位为 1;

4) 等待中断标志寄存器中的 *SLAVE_INT* 位置位, 表示 I²C 作为从设备被校验成功, 并且作为从设备发送器, 读取数据寄存器, 可以根据读取数据的最低位确定正确的传输方向(是读还是写);

5) 写待发送数据到数据寄存器中, 等待中断标志寄存器中的 *BYTE_INT* 位置位, 重复

这个过程，直到收到未确认中断为止（中断标志寄存器中的 $NOACK\_INT$ 位为 1），传输完成。

4.13.4 I²C 寄存器地址映射

I²C 寄存器地址映射见表 4-13-1，详细说明见附录 3.13。

表4-13-1 I²C寄存器地址映射

表4-13-1（a） I²C0寄存器地址映射

寄存器名称	地址	复位值
I ² C0 中断标志寄存器	0x81000800	0x00000000
I ² C0 控制寄存器	0x81000804	0x00000000
I ² C0 速率寄存器	0x81000808	0x00000000
I ² C0 数据寄存器	0x8100080C	0x00000000
I ² C0 从地址寄存器 1	0x81000810	0x00000000
I ² C0 从地址寄存器 0	0x81000814	0x00000000
I ² C0 状态寄存器	0x81000818	0x00000000
I ² C0 中断屏蔽寄存器	0x8100081C	0x00000000
I ² C0 存储器件地址寄存器	0x81000820	0x00000000
I ² C0 存储地址寄存器 1	0x81000824	0x00000000
I ² C0 存储地址寄存器 2	0x81000828	0x00000000
I ² C0 存储模式选择寄存器	0x8100082C	0x00000000
I ² C0 频率寄存器	0x81000830	0x000000ff

表4-13-1（b） I²C1寄存器地址映射

寄存器名称	地址	复位值
I ² C1 中断标志寄存器	0x81000900	0x00000000
I ² C1 控制寄存器	0x81000904	0x00000000
I ² C1 速率寄存器	0x81000908	0x00000000
I ² C1 数据寄存器	0x8100090C	0x00000000
I ² C1 从地址寄存器 1	0x81000910	0x00000000
I ² C1 从地址寄存器 0	0x81000914	0x00000000
I ² C1 状态寄存器	0x81000918	0x00000000
I ² C1 中断屏蔽寄存器	0x8100091C	0x00000000

寄存器名称	地址	复位值
I ² C1 存储器件地址寄存器	0x81000920	0x00000000
I ² C1 存储地址寄存器 1	0x81000924	0x00000000
I ² C1 存储地址寄存器 2	0x81000928	0x00000000
I ² C1 存储模式选择寄存器	0x8100092C	0x00000000
I ² C1 频率寄存器	0x81000930	0x000000ff

4.14 UART

4.14.1 概述

BM3823A 片内集成四路 UART。其中，UART1、UART2 没有 FIFO、具有硬件流控模式、管脚与 GPIO 复用；UART3、UART4 具有 FIFO、没有硬件流控模式、管脚独立。

4.14.2 UART1、UART2

UART1、UART2 由发送保持寄存器、接收保持寄存器、发送移位寄存器、接收移位寄存器、波特率产生模块和控制模块组成；控制主要通过对 UART1、UART2 的控制寄存器、状态寄存器、分频器重载寄存器、数据寄存器进行操作完成。UART1、UART2 的结构框图如图 4-14-1 所示：

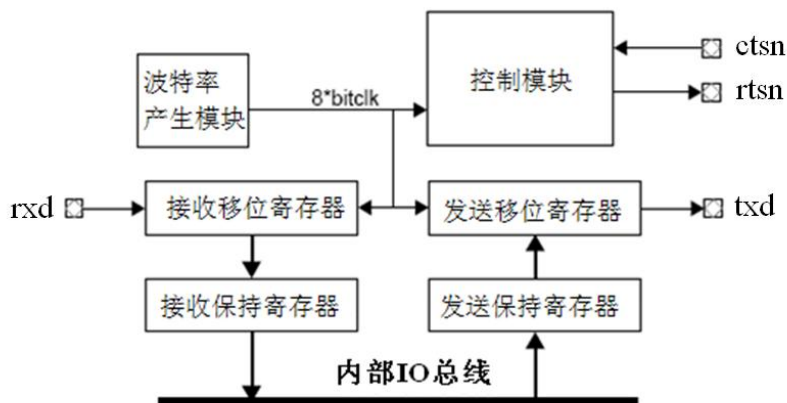


图4-14-1 UART1、UART2结构框图

4.14.2.1 发送操作

在 UART1、UART2 控制寄存器中置发送使能 TE 位，可以进行发送操作。当进行数据发送时，将数据从发送保持寄存器传输到发送移位寄存器，从串行输出口 txd 管脚输出串行

数据流，首先发送的是数据最低位。输出串行数据流的同时，自动在每 8 位数据前增加一位起始位、数据后增加一位停止位和可选的奇偶校验位。数据帧如图 4-14-2 和图 4-14-3 所示：

无奇偶校验的数据帧：

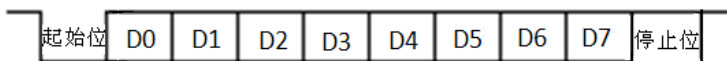


图4-14-2 UART1、UART2无奇偶校验的数据帧格式

包含奇偶校验的数据帧：

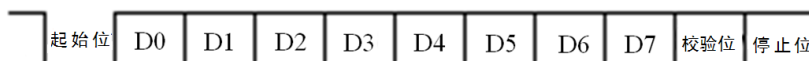


图4-14-3 UART1、UART2包含奇偶校验的数据帧格式

在停止位之后，如果在发送保持寄存器中没有出现新的字符，那么发送串行数据输出口仍然保持高电平，并使 **UART 状态寄存器** 的发送移位寄存器空 **TS** 位为 1。

当在发送保持寄存器中还有待发送的字符时，将继续发送并清除 **TS** 位。如果发送中关闭发送功能，则将在完成本次发送后停止发送。

4.14.2.2 接收操作

在 **UART1、UART2 控制寄存器** 中置接收使能 **RE** 位，可以进行接收操作。接收端等待在接收数据输入引脚上出现一个从高电平到低电平变化的开始位。若 **RE** 使能，则将在半个时钟之后采样串行输入状态。如果串行输入采样为高，即开始位无效，继续搜寻有效的开始位；如果串行输入采样为低，则认为开始位有效，继续在每个时钟间隔内采样串行输入数据，接收移位寄存器串行接收到一帧数据后，将数据送入接收保持寄存器。

接收时，首先接收数据的最低位。收到的数据送入接收保持寄存器后，同时 **UART 状态寄存器** 的数据准备好 **DR** 位变为 1。当接收保持寄存器和接收移位寄存器中都有未读取的数据，且又检测到新的数据时，溢出错 **OV** 位置 1，并且新数据将写入接收移位寄存器。

4.14.2.3 流控

若使能流控，则 **ctsn** 输入必须为低电平时才能进行发送。若发送过程中，**ctsn** 变高，将完成当前帧的发送，之后串行输出保持无效状态，直到 **ctsn** 重新变为低电平；接收时，当监测到有效的起始位时，**rtsn** 变为高电平。当从接收保持寄存器中读出数据后，并且没有要接收的数据，**rtsn** 变为低电平。

如果 ctsn 连接到了接收端的 rtsn，则可以有效地避免数据溢出。

4.14.2.4 回环

若 **UART 控制寄存器**的 *LB* 位置位，UART1、UART2 将进入回环模式。在该模式下，发送端输出将在内部与接收端输入相连，rtsn 也将连接 ctsn，从而可进行回环测试，验证接收/发送端以及相关软件的工作情况。在该模式下，输出保持无效状态，不能向外发送数据。

4.14.2.5 波特率产生

UART1、UART2 通过一个 12 位递减计数的分频器来产生需要的波特率。当外部时钟 *EC* 不置位时，分频器由系统时钟控制，在每次下溢时产生 UART TICK 信号，同时重载 UART1、UART2 分频值。UART TICK 信号的频率是所需波特率的 8 倍。若 *EC* 位置位，则 UART 分频器通过 PIO[3]，由外部时钟触发，在这种情况下，PIO[3]的频率必须小于系统时钟频率的一半。

UART1、UART2 的波特率产生公式如下：

$$\text{分频器重载值} + 1 = ((\text{系统时钟} * 10) / (\text{波特率} * 8) - 5) / 10$$

4.14.2.6 中断产生

在以下的情况下，UART1、UART2 将产生一个中断：

当发送使能、发送中断使能、流控不使能，发送保持寄存器数据不为空，产生一个中断；
当接收使能，接收中断使能，接收移位寄存器接收到一个完整的数据后，产生一个中断。

4.14.2.7 使用说明

在进行串口初始化时，须对 **UART 状态寄存器**以及 **UART 控制寄存器**进行配置，并读两次数据寄存器来清空 **UART 数据寄存器**，保证初始状态正确。若使用中断模式接收数据，且在中断屏蔽状态下接收了数据，则在串口中断服务程序中须增加两次读数据寄存器的操作，保证数据接收正确。

4.14.3 UART3、UART4

UART3、UART4 带有 32 字节的接收 FIFO 和发送 FIFO，并可以分别工作在中断模式或查询模式下。

UART3、UART4 带有一个可配置的波特率发生器，每一帧数据包含 1 位的起始位，8 位宽的数据，1 位或 2 位的停止位，和可配置触发条件的中断。

每个 UART 由发送 FIFO、接收 FIFO、发送移位寄存器、接收移位寄存器、波特率产生器和控制模块组成；控制主要通过对波特率分频寄存器、发送/接收设置寄存器、状态寄存器、数据寄存器、清状态寄存器、发送结束寄存器和中断使能寄存器进行操作完成。基本结构如图 4-14-4 所示：

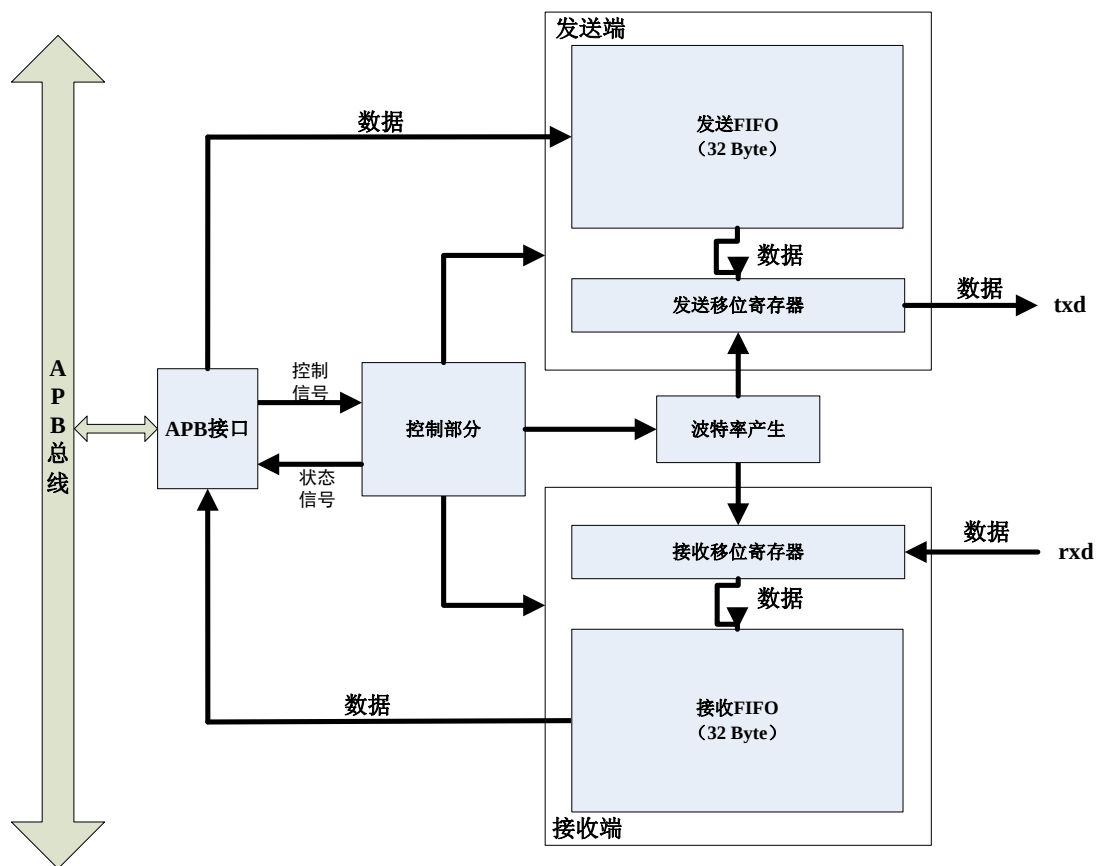


图4-14-4 UART3、UART4结构框图

4.14.3.1 发送操作

UART3、UART4 的接收 FIFO 未滿时，向接收移位寄存器写入的数据会送入 FIFO。当发送 FIFO 不空时，串口自动把数据送到发送移位寄存器，组帧从 txd 发送出去。UART3、UART4 帧的数据位为 8 比特长，可以选择 1 位或 2 位停止位。发送时，首先发送的是数据最低位。当 FIFO 没有空余位置时，不能将新数据写入发送 FIFO。

数据帧格式如下图 4-14-5 和图 4-14-6 所示：

数据帧，无奇偶校验：

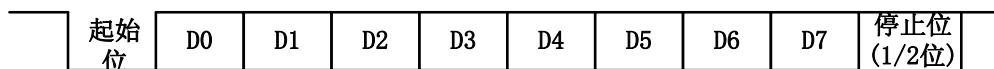


图4-14-5 UART3、UART4无奇偶校验数据帧格式

数据帧，包含奇偶校验：

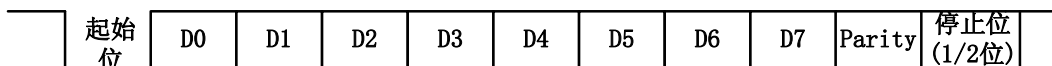


图4-14-6 UART3、UART4包含奇偶校验数据帧格式

4.14.3.2 接收操作

当 UART3、UART4 rxd 上收到数据时，数据通过接收移位寄存器传入接收 FIFO，接收时首先接收数据的低位，从 **UART 数据寄存器** 可读到 FIFO 中最先收到的数据。接收数据送入接收移位寄存器时，判断是否出现停止位错，并对 **UART 状态寄存器** 的相应位置位。当接收 FIFO 全满时，再有新数据送入则产生溢出错误。

4.14.3.3 回环

若 **UART 发送/接收设置寄存器** 的 LB 位置“1”，UART3、UART4 将进入回环模式。在该模式下，发送端输出将在内部与接收端输入相连，从而可进行回环测试，验证接收/发送端以及相关软件的工作情况。在该模式下，输出保持无效状态，不能向外发送数据。

4.14.3.4 波特率产生

UART3、UART4 的波特率根据如下公式设定：

波特率分频寄存器值 + 1 = 主时钟频率 ÷ 波特率 ÷ 16（余数 < 8，则向下取整；余数 ≥ 8，则向上取整）

波特率分频寄存器上电复位值为 0x104（40MHz 下 9600 波特率）。

4.14.3.5 中断产生

在以下的情况下，UART3、UART4 将产生一个中断：

- 发送全空中断使能且 **UART 状态寄存器** 的 NTD 位为 1 时，发送 FIFO 从有数据变为空；
- 接收全满中断使能时，接收 FIFO 全满；
- 配置发送 FIFO 为 1/4、1/2、3/4 空中断时，如果发送 FIFO 满足该状态，则产生中

断；

- 配置接收 FIFO 为 1/4、1/2、3/4 满中断时，如果接收 FIFO 满足该状态，则产生中断；
- 接收全满中断使能时，接收 FIFO 全满；
- 停止位错误中断使能时，产生停止位错；
- 超时中断使能时，产生超时错。

UART3 的 3/4 满中断使用举例：

- 允许一级中断列表的 10 号中断，屏蔽其他一级中断；
- 允许二级中断的 UART3 中断，屏蔽其他二级中断；
- 设置 **UART3 波特率分频寄存器**；
- 设置 **UART3 发送/接收设置寄存器**。

以上条件正确设置后，若 FIFO 空间满足 3/4 满条件，则正确响应 UART3 中断。若想连续响应 UART3 中断，中断产生后，在中断处理函数中需要将 FIFO 中的数据取出，若不取出，则中断不再产生。

4.14.4 UART 寄存器地址映射

UART 的寄存器地址映射如表 4-14-1 所示。寄存器说明详见附录 3.14。

表4-14-1 UART寄存器地址映射表

寄存器名称	地址	复位值 ^注
UART1 数据寄存器	0x81000070	0x000000XX
UART1 状态寄存器	0x81000074	0x00000006
UART1 控制寄存器	0x81000078	0x000000XX
UART1 分频器重载寄存器	0x8100007C	0x00000XXX
UART2 数据寄存器	0x81000080	0x000000XX
UART2 状态寄存器	0x81000084	0x00000006
UART2 控制寄存器	0x81000088	0x000000XX
UART2 分频器重载寄存器	0x8100008C	0x00000XXX
UART3 波特率分频寄存器	0x810001C0	0x00000104
UART3 发送/接收设置寄存器	0x810001C4	0x00000000

寄存器名称	地址	复位值 ^注
UART3 状态寄存器	0x810001C8	0x00002000
UART3 数据寄存器	0x810001CC	0x000000XX
UART3 清状态寄存器	0x810001D0	0x00000000
UART3 发送结束寄存器	0x810001D4	0x00000000
UART3 中断使能寄存器	0x810001D8	0x00000000
UART4 波特率分频寄存器	0x810001E0	0x00000104
UART4 发送/接收设置寄存器	0x810001E4	0x00000000
UART4 状态寄存器	0x810001E8	0x00002000
UART4 数据寄存器	0x810001EC	0x000000XX
UART4 清状态寄存器	0x810001F0	0x00000000
UART4 发送结束寄存器	0x810001F4	0x00000000
UART4 中断使能寄存器	0x810001F8	0x00000000

注：“复位值”按照十六进制统一写法“0xXXX0XXXX”，“0x”表示 16 进制，每 4bit 中有 1bit 不定值，即写为“X”，4bit 全部为固定值则写为相应 16 进制数值。

4.15 定时器和看门狗

4.15.1 预分频器

BM3823A 的定时器单元包含两个普通定时器和一个看门狗定时器。这三个定时器均为 32 位定时器，共用同一个 10 位预分频器。预分频器由系统时钟控制，系统时钟 clk 上升沿到达时，预分频器的计数值减 1，判断预分频器的计数值是否溢出，如果溢出，**定时器预分频重载寄存器**自动重载计数值，同时为定时器和看门狗产生一个计数 TICK 信号。预分频器有效的分频值等于预分频重载计数值加 1。

两个普通定时器和看门狗定时器共用同一个减一计数器，预分频器重载计数值最小为 3。

4.15.2 定时器

定时器 1、定时器 2 分别包含 3 个寄存器：计数寄存器，重载寄存器，控制寄存器。**定时器控制寄存器**通过设置使能位 EN 来控制定时器是否使能。当预分频器产生 TICK 信号时，定时器的计数值减 1，定时器下溢时，产生中断，该中断能够被**中断屏蔽和优先级寄存器**屏蔽。此时，如果设置**定时器控制寄存器**的自动重载使能位 RL 为 1，定时器将重新装载寄存器值，定时器继续工作；如果 RL 为 0，定时器停止工作。通过设置**定时器控制寄存器**中的

LD 位，定时器可在任何时候强制重载计数值。定时器有效的分频值等于定时器重载计数值加 1。

4.15.3 看门狗

看门狗定时器在系统复位后，处于不使能状态，通过设置**定时器 1 控制寄存器**中的 **WTDEN** 使能位使能看门狗，当预分频器产生 TICK 信号时，**看门狗计数值寄存器**的计数值减 1，计数值减为 0 时，看门狗输出的外部信号 WDOG 为低电平。系统应在看门狗计数值减为 0 之前，更新**看门狗计数值寄存器**的计数值。看门狗有效的分频值等于看门狗计数值寄存器值加 1。

4.15.4 定时器和看门狗的寄存器地址映射

定时器和看门狗的寄存器地址映射如表 4-15-1 所示，具体说明详见附录 3.15。

表4-15-1 定时器和看门狗寄存器地址映射对照表

寄存器名称	地址	复位值 ^注
定时器 1 计数寄存器	0x81000040	0xFFFFFFFF
定时器 1 重载寄存器	0x81000044	0xFFFFFFFF
定时器 1 控制寄存器	0x81000048	0x0000000X
看门狗计数值寄存器	0x8100004C	0xFFFFFFFF
定时器 2 计数寄存器	0x81000050	0xFFFFFFFF
定时器 2 重载寄存器	0x81000054	0xFFFFFFFF
定时器 2 控制寄存器	0x81000058	0x0000000X
定时器预分频计数寄存器	0x81000060	0x00000000
定时器预分频重载寄存器	0x81000064	0x000003FF

注：“复位值”按照十六进制统一写法“0xXXX0XXXX”，“0x”表示 16 进制，每 4bit 中有 1bit 不定值，即写为“X”，4bit 全部为固定值则写为相应 16 进制数值。

4.16 计数器

4.16.1 概述

BM3823A 包含 6 路 32 位的计数器，实现外部输入端口的计数功能，可对由外部管脚来的正脉冲驱动计数，来一次正脉冲，计数加 1，最大值为 0xFFFFFFFF。

4.16.2 计数器功能

BM3823A 包含 6 路 32 位的计数器，可对由外部管脚来的正脉冲驱动计数，来一次正脉

冲，计数加 1；

在计数之前，可对外部脉冲采样频率进行分频；

设置所需外部脉冲的采样次数时，可以设置采样次数，最少采样 3 次；

软件控制计数器开始/停止；软件控制写计数器数据寄存器，将计数器清零。

当采样计数次数为 n ，采样脉宽为 m 个时钟周期（宽 T ）时，应避免输入介于 $(n-1) \times m \times T$ 到 $n \times m \times T$ 秒之间的脉冲，防止计数出错。

计数器计满数之后会从 0 重新开始计数，并不产生中断。

输入端的脉冲宽度必须大于 $3 \times (\text{滤波分频值} + 1)$ 个主时钟周期才能保证正确的滤波输出波形，这样才能保证计数器的计数值的正确。

4.16.3 计数器的寄存器地址映射

6 路 32 位计数器的每路都有对应的控制寄存器、数据寄存器以及分频寄存器，可以单独进行控制。计数器的寄存器地址映射如表 4-16-1 所示，寄存器说明详见附录 3.16。

表 4-16-1 计数器寄存器地址映射对照表

寄存器名称	地址	复位值
计数器 1 控制寄存器	0x81000210	0x00000000
计数器 1 数据寄存器	0x81000214	0x00000000
计数器 1 分频寄存器	0x81000218	0x00000000
计数器 2 控制寄存器	0x81000220	0x00000000
计数器 2 数据寄存器	0x81000224	0x00000000
计数器 2 分频寄存器	0x81000228	0x00000000
计数器 3 控制寄存器	0x81000230	0x00000000
计数器 3 数据寄存器	0x81000234	0x00000000
计数器 3 分频寄存器	0x81000238	0x00000000
计数器 4 控制寄存器	0x81000240	0x00000000
计数器 4 数据寄存器	0x81000244	0x00000000
计数器 4 分频寄存器	0x81000248	0x00000000
计数器 5 控制寄存器	0x81000250	0x00000000
计数器 5 数据寄存器	0x81000254	0x00000000
计数器 5 分频寄存器	0x81000258	0x00000000
计数器 6 控制寄存器	0x81000260	0x00000000

寄存器名称	地址	复位值
计数器 6 数据寄存器	0x81000264	0x00000000
计数器 6 分频寄存器	0x81000268	0x00000000

4.17 PWM 定时器

4.17.1 PWM 定时器功能

BM3823A 有 6 路 24 位 PWM 定时器，每路定时器单独用自己的预分频器，预分频值的范围为 0 到 0x3FF，预分频的有效周期数为预分频值加 1。

预分频器为一个 10 位的减 1 计数器，可以由系统时钟定时，每个时钟周期减 1，当预分频计数器产生下溢时，定时器的值减 1；预分频重载值自动重新装载。

当定时器计数溢出时，将向外部输出信号，如果允许，将产生中断。

定时器输出信号方式有三种：单脉冲方式；电平翻转输出即定时到达重新计数时，装载相同的预置值，输出电平反相；PWM 输出即在总的定时时间不变的条件下，可软件设置两个预置值，当定时到达重新计数时，交替装载两个不同的预置值，从而确定占空比，总的定时时间一定。

实时启动能力：每一个定时器都有一个外部 timerup 管脚，作为输入信号可以启动定时器和关闭定时器，软件可设置能否允许外部信号对定时器实时启动。

每 1 路都可以软件配置用外部时钟定时，还是用内部时钟；当使用外部时钟定时时，将不再进行预分频。外部时钟周期必须为内部时钟周期的两倍及以上的整数倍。

开启并使用定时器，需配置 **PWM 定时器控制寄存器** 的使能位，并在相应的 MASK 位同时为 1 时才能有效写入；关闭定时器时，相应的 MASK 位同时为 1 时才能关闭。

4.17.2 PWM 定时器输出

定时器有三种输出方式，其波形如图 4-17-1 到图 4-17-3 所示：

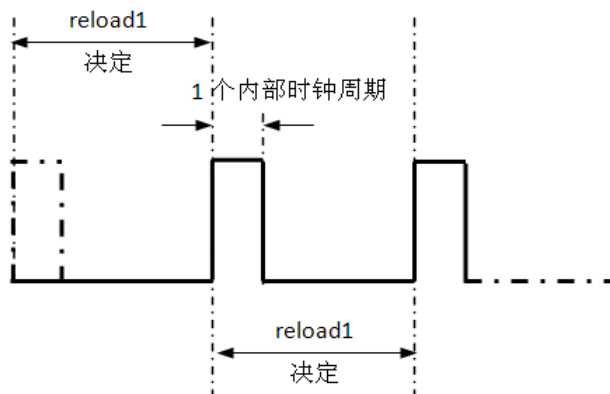


图4-17-1 PWM定时器的脉冲输出

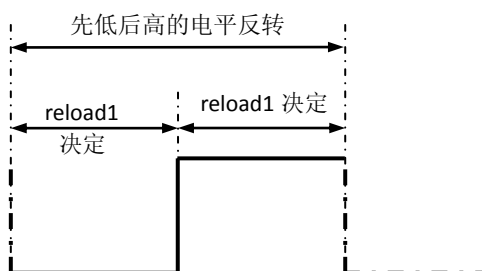


图4-17-2 PWM定时器的电平反转输出

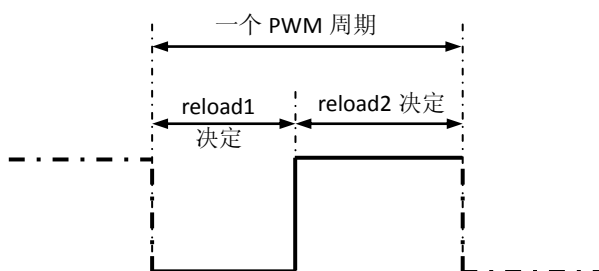


图4-17-3 PWM定时器的PWM输出

4.17.3 PWM 寄存器地址映射

PWM 寄存器地址映射见表 4-17-1，详细说明见附录 3.17。

表4-17-1 PWM寄存器地址映射

寄存器名称	地址	复位值
PWM 定时器控制寄存器	0x81000300	0x00000000
PWM 定时器 1 计数寄存器	0x81000310	0x00000000
PWM 定时器 1 预置值寄存器 1	0x81000314	0x80000000
PWM 定时器 1 预置值寄存器 2	0x81000318	0x00000000

寄存器名称	地址	复位值
PWM 定时器 1 配置寄存器	0x8100031C	0x00000000
PWM 定时器 2 计数寄存器	0x81000320	0x00000000
PWM 定时器 2 预置值寄存器 1	0x81000324	0x80000000
PWM 定时器 2 预置值寄存器 2	0x81000328	0x00000000
PWM 定时器 2 配置寄存器	0x8100032C	0x00000000
PWM 定时器 3 计数寄存器	0x81000330	0x00000000
PWM 定时器 3 预置值寄存器 1	0x81000334	0x80000000
PWM 定时器 3 预置值寄存器 2	0x81000338	0x00000000
PWM 定时器 3 配置寄存器	0x8100033C	0x00000000
PWM 定时器 4 计数寄存器	0x81000340	0x00000000
PWM 定时器 4 预置值寄存器 1	0x81000344	0x80000000
PWM 定时器 4 预置值寄存器 2	0x81000348	0x00000000
PWM 定时器 4 配置寄存器	0x8100034C	0x00000000
PWM 定时器 5 计数寄存器	0x81000350	0x00000000
PWM 定时器 5 预置值寄存器 1	0x81000354	0x80000000
PWM 定时器 5 预置值寄存器 2	0x81000358	0x00000000
PWM 定时器 5 配置寄存器	0x8100035C	0x00000000
PWM 定时器 6 计数寄存器	0x81000360	0x00000000
PWM 定时器 6 预置值寄存器 1	0x81000364	0x80000000
PWM 定时器 6 预置值寄存器 2	0x81000368	0x00000000
PWM 定时器 6 配置寄存器	0x8100036C	0x00000000

4.18 调试支持单元

4.18.1 概述

BM3823A 具有调试支持单元（Debug Support Unit, DSU），方便软件在目标硬件上进行调试。通过 DSU 可以将 BM3823A 置于调试模式，允许 BM3823A 中所有寄存器及 Cache 存储器的读写访问。DSU 同时具有一个大小为 512×16 字节的跟踪缓冲区 Tracebuffer，保存已经被执行的指令和总线上传过来的数据。调试通讯链接实现了一个简单的读/写协议并使用标准的异步串口通讯。

DSU 控制寄存器在任何时候都可以被访问。而 BM3823A 的寄存器和 Cache 只能在 BM3823A 进入调试模式时被访问。Tracebuffer 只能在跟踪被禁用或者跟踪完成时被访问。在调试模式下，DSU 控制 BM3823A，流水线暂停。

仅当 DSU 的外部引脚 dsuen 使能的情况下，才能进入调试模式。dsuen 引脚拉高，使能调试模式。

使能调试模式后，通过下列任一事件 BM3823A 可以进入调试模式：

- ◆ 执行一个断点指令 ta 1；
- ◆ 观测点命中；
- ◆ 在系统复位结束前三个时钟周期 dsubre 为高电平，处理器在复位结束后运行四条指令后立即进入调试模式；
- ◆ 设置 **DSU 控制寄存器**中的 BN 位；
- ◆ 进入错误模式的陷阱；
- ◆ **DSU 控制寄存器**中定义的一个或多个陷阱触发；
- ◆ 执行一个单步操作；
- ◆ DSU 断点命中。

当进入调试模式后，BM3823A 发生以下改变：

- ◆ **程序计数器 PC** 和 **nPC** 被存入临时寄存器（可通过调试单元访问）；
- ◆ DSU 状态指示信号 dsuact 输出高电平；
- ◆ 时钟单元停止，冻结定时器和看门狗。

通过清除 **DSU 控制寄存器**中的 BN 位或取消 dsuen 信号可以使 BM3823A 恢复运行模式，此时定时器恢复计时，BM3823A 从保存的 **PC** 和 **nPC** 处开始执行。BM3823A 进入陷阱后也可以进入调试模式，例如：应用程序运行中，发生了非屏蔽陷阱，此时应用程序中止，暂停执行。

4.18.1.1 时间标志

DSU 时间标志计数器的值作为执行操作的时间参考。BM3823A 运行时，计数器每时钟周期加一。BM3823A 进入调试模式时计数器停止，BM3823A 重新开始运行时计数器重新开始计数。

4.18.1.2 Tracebuffer

Tracebuffer 是一个环形缓冲区，存储执行的指令或者内部总线上传输的数据，其大小为 512 行，每行 16 字节。Tracebuffer 由 **DSU 控制寄存器**和 **Tracebuffer 控制寄存器**控制。当 BM3823A 进入调试模式时，跟踪暂停。

Tracebuffer 有三种工作模式：

- ◆ 指令跟踪模式（只记录执行的指令）
- ◆ 总线跟踪模式（只记录内部总线上传输的数据）
- ◆ 混合跟踪模式（同时记录执行指令和总线数据）

Tracebuffer 控制单元包含了两个计数器（总线计数器和指令计数器），表示当前跟踪记录写入 Tracebuffer 的位置。由于 Tracebuffer 是环形的，因此计数器值实际上指向 Tracebuffer 当前记录位置，每次对 Tracebuffer 的记录完成后，计数器值自动增加。

（1）指令跟踪

Tracebuffer 控制寄存器中的 *TI* 位置 1，使能指令跟踪模式。

在指令跟踪期间，除多周期指令，一条指令占用缓冲区的一行。多周期指令占用跟踪缓冲区两行或三行：

- ◆ 对于存储指令，Tracebuffer 中第一行[95:64]用于记录存储地址；第二行相同字段存储写入的数据。对 STD 指令，需要使用三行记录，第一行[95:64]字段，记录存储地址，第二行相同字段存储写入数据的高 32bit，第三行相同字段存储写入数据的低 32bit。当一行的 bit 126 置 1 时，表示这是一条指令的第二行或第三行记录。
- ◆ 双字加载指令（LDD）使用两行记录，第一行[95:64]字段存储读出数据高 32bit，第二行相同字段存储读出数据的低 32bit。
- ◆ 乘法和除法指令使用两行记录。对乘法指令，第一行[95:64]字段记录计算结果的高 32bit，第二行[95:64]字段记录计算结果的低 32bit；对除法指令，第一行[95:64]字段记录符号扩展，第二行[95:64]字段记录 32bit 运算结果。Bit 126 置位，表示同一指令记录的第二行。
- ◆ 双精度浮点指令使用两行记录。第一行[95:64]字段存储计算结果的高 32bit，第二行[95:64]字段存储计算结果的低 32bit。Bit 126 置位，表示同一指令记录的第二行。

记录格式如图 4-18-1 和表 4-18-1 所示：

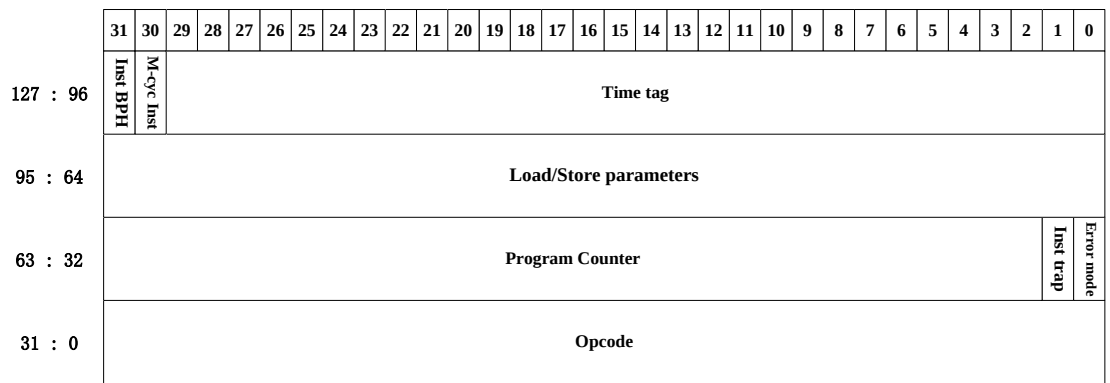


图4-18-1 指令跟踪记录格式

表4-18-1 指令跟踪记录格式

比特位	含义	说明
127	指令断点命中	如果出现DSU指令断点命中，设置为‘1’
126	多周期指令	多周期指令（LDD，ST或FPOP）的第二、三行设置为‘1’
125:96	时间标志	时间标志计数器的值
95:64	Load/Store参数	指令结果、存储地址或者存储数据
63:34	程序计数器	程序计数器（不记录最低两位，因为它们总是0）
33	指令陷阱	如果被跟踪的指令进入陷阱则设置为‘1’
32	处理器错误模式	如果被跟踪的指令引起处理器进入错误模式则设置为‘1’
31:0	Opcode	指令操作码

当跟踪被冻结时，产生 11 号中断。

（2）总线跟踪

将 **Tracebuffer 控制寄存器**的 TA 设置为 1，可以使能总线跟踪模式。

在总线跟踪期间，Tracebuffer 的一行记录一次内部总线上的操作。记录格式如图 4-18-2

和表 4-18-2 所示：

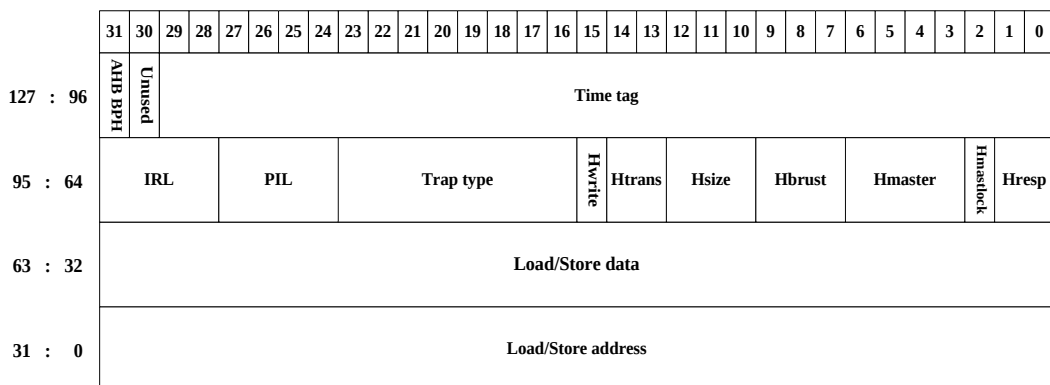


图4-18-2 总线跟踪记录格式

表4-18-2 总线跟踪记录格式

比特位	含义	说明
127	AHB 断点命中	如果 DSU AHB 断点命中则设为 ‘1’
126	-	未使用
125:96	时间标志	时间标志计数器的值
95:92	IRL	中断请求输入
91:88	PIL	中断级别(psr.pil)
87:80	trap type	陷阱类型(psr.tt)
79	Hwrite	AHB HWRITE
78:77	Htrans	AHB HTRANS
76:74	Hsize	AHB HSIZE
73:71	Hburst	AHB HBURST
70:67	Hmaster	AHB HMASTER
66	Hmastlock	AHB HMASTLOCK
65:64	Hresp	AHB HRESP
63:32	Load/Store data	AHB HRDATA or HWDATA
31:0	Load/Store address	AHB HADDR

(3) 混合跟踪

在混合模式下，Tracebuffer 被分为两半：指令存在低一半 Tracebuffer，总线传输存在高一半 Tracebuffer。AHB 索引计数器的 MSB 位自动保持为高，指令索引计数器的 MSB 保持为低，即 AHB 跟踪记录的起始地址为 0x9001-1000，指令跟踪记录的起始地址为 0x9001-0000。

(4) DSU 映射

表 4-18-3 DSU 映射表

地址	寄存器
0x8000-00c4	DSU 串口状态寄存器
0x8000-00c8	DSU 串口控制寄存器
0x8000-00cc	DSU 串口波特率重载寄存器
0x9000-0000	DSU 控制寄存器
0x9000-0004	Trace buffer 控制寄存器
0x9000-0008	Time tag counter
0x9000-0010	AHB 断点地址 1
0x9000-0014	AHB 掩码 1
0x9000-0018	AHB 断点地址 2
0x9000-001C	AHB 掩码 2
0x9001-0000~0x9001-1FFF	Trace buffer ⁽¹⁾
0x9002-0000~0x9002-029F	IU/FPU 寄存器堆
0x9008-0000	Y 寄存器
0x9008-0004	PSR 寄存器
0x9008-0008	WIM 寄存器
0x9008-000C	TBR 寄存器
0x9008-0010	PC 寄存器
0x9008-0014	NPC 寄存器
0x9008-0018	FSR 寄存器
0x9008-001C	DSU 陷阱寄存器 ⁽²⁾
0x9008-0060	ASR24
0x9008-0064	ASR25
0x9008-0068	ASR26
0x9008-006C	ASR27
0x9008-0070	ASR28
0x9008-0074	ASR29
0x9008-0078	ASR30
0x9008-007C	ASR31
0x9010-0000~0x9010-7FFF	指令 Cache tags ⁽³⁾

地址	寄存器
0x9010-0000~0x9010-1FFF	第一路
0x9010-2000~0x9010-3FFF	第二路
0x9010-4000~0x9010-5FFF	第三路
0x9010-6000~0x9010-7FFF	第四路
0x9014-0000~0x9014-7FFF	指令 Cache 数据
0x9014-0000~0x9014-1FFF	第一路
0x9014-2000~0x9014-3FFF	第二路
0x9014-4000~0x9014-5FFF	第三路
0x9014-6000~0x9014-7FFF	第四路
0x9018-0000~0x9018-3FFF	数据 Cache tags ⁽⁴⁾
0x9018-0000~0x9018-1FFF	第一路
0x9018-2000~0x9018-3FFF	第二路
0x901C-0000~0x901C-3FFF	数据 Cache 数据
0x901C-0000~0x901C-1FFF	第一路
0x901C-2000~0x901C-3FFF	第二路

注：(1)每行 128 位，共 512 行。

(2)DSU 陷阱寄存器是一个只读寄存器，指出哪种 SPARC 陷阱导致处理器进入调试模式。当通过设置

DSU 控制寄存器中的 BN 位强制进入调试模式时，陷阱类型是 0x0b。

DSU 陷阱寄存器

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																			EM	Trap type								保留			
r																															
x																												0000			

位号	助记符	描述
31:29	—	保留
12	EM	错误模式 (EM)，如果陷阱使得处理器进入错误模式，则该位置为 1。
11:4	TT	8 位 SPARC 陷阱类型。
3:0	—	保留

(3)由于 Cache 结构，一行指令 Cache 的 TAG 在 Tracebuffer 中的映射为连续的 8 个字，读写 8 个字中

的任何一个都是对同一个 TAG 操作。

(4)由于 Cache 结构，一行数据 Cache 的 TAG 在 Tracebuffer 中的映射为连续的 4 个字，读写 4 个字中的任何一个都是对同一个 TAG 操作。

(5) DSU 断点

DSU 包含了两个断点寄存器来匹配任何内部的 AHB 总线地址或者执行的指令。断点命中的典型应用是冻结 Tracebuffer，但也可以把 BM3823A 置于调试模式。

冻结操作可以被延迟，通过把 **DSU 控制寄存器**中的 *TDELAY* 字段置为非零值来实现。这种情况下，*TDELAY* 的值会在每一次额外的跟踪后减一，递减至零时冻结 Tracebuffer。如果 **DSU 控制寄存器**中的 *BT* 置 1，则 Tracebuffer 冻结时，DSU 强制 BM3823A 进入调试模式。因为流水延迟，在 BM3823A 被置于调试模式前，最多有 4 个额外的指令可能被执行。针对每个断点，都可以使用屏蔽寄存器，针对某块地址进行屏蔽，仅有屏蔽寄存器中相应位为 1 时，才会在断点检测时进行比较。在执行指令时中断，应置 *EN* 位；在 AHB 加载或存储访问时中断，应置 *LD* 或 *ST* 位。

4.18.2 调试通讯链接

调试通讯链接包含一个串口，支持异步通信协议传输访问参数和数据。链接命令由一个控制字节、32 位地址和可选的写入数据组成。如果 **DSU 控制寄存器**的 *LR* 位置位，则每次 AHB 传输之后发送响应的字节。如果未设置 *LR* 位，则写访问不返回任何数据，读访问只返回读到的数据。

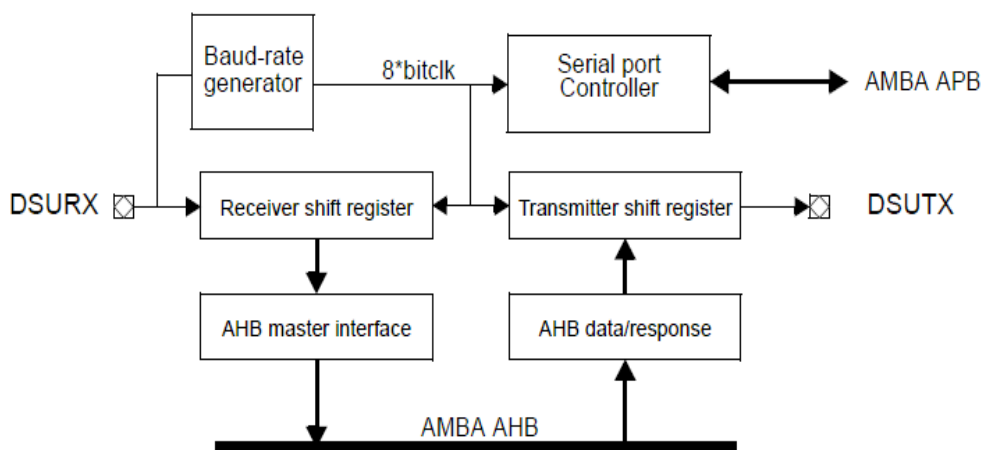


图 4-18-3 DSU 通讯链接框图

4.18.2.1 数据帧

数据以 8-bit 为基础进行发送，数据帧格式如图 4-18-4 所示：

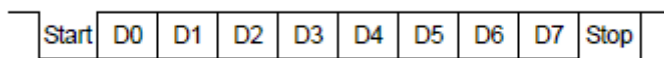
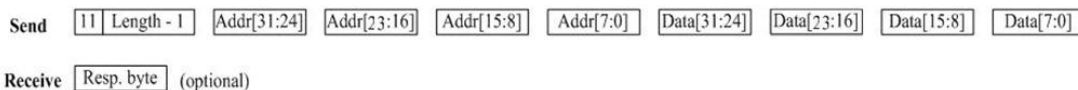


图 4-18-4 DSU 串口数据帧

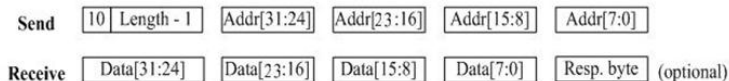
4.18.2.2 调试控制命令

通过通讯链路，可以对内部总线上任何地址进行读/写操作。通过设置长度字段为 $n-1$ （其中 n 标明传输字的长度），可以进行块传输。写访问时，发送控制字节、目标地址段首地址以及需要写入的数据，数据写入地址以目标地址段首地址为基地址，以字为单位自动递增。读访问时，发送控制字节和目标地址段首地址，返回相应大小的数据。DSU 命令如图 4-18-5 所示。

DSU 写命令



DSU 读命令



Response byte encoding
bit 7:3 = 000000
bit 2 = DMODE
bit 1:0 = HRESP

图 4-18-5 DSU 命令

4.18.2.3 串口通信时钟

时钟控制，每次下溢时产生一个串口 tick。每次下溢之后分频器重新载入串口重载寄存器的值。串口 tick 频率应该是期望波特率的 8 倍。

如果不进行软件编程，波特率能够被自动检测，这通过寻找接收数据（2-bit 周期）的两个下降沿的最短周期实现。当发现三个相同的 2-bit 周期时，相应的分频器重载值锁存入重载寄存器，设置 **DSU 串口控制寄存器** 中的 *BL* 位。如果 *BL* 位通过软件复位，波特率检测过程重新开始。当接收器收到一个 ‘break’ 时，波特率检测重新开始，这允许外部的发送器改变波特率。为了正确的检测波特率，在复位或者发送 ‘break’ 之后应该向接收器发送值 0x55。

分频值计算公式如下：

$$\text{scaler} = \frac{\frac{\text{sysclk} \times 10}{\text{baudrate} \times 8} - 5}{10}$$

4.18.3 启动调试

将 dsuen 和 dsubre 管脚同时置为高电平时，在复位后 BM3823A 执行 0x0 地址处的四条指令后进入调试模式。此时可以通过 DSU 通信接口初始化系统，下载程序进行调试。

4.18.4 调试支持单元寄存器地址映射

调试支持单元的寄存器地址映射见表 4-18-4，详细说明见附录 3.18。

表 4-18-4 DSU 寄存器地址映射对照表

寄存器名称	地址	复位值 ^注
DSU 控制寄存器	0x90000000	0xXXX0XXXX
Trace buffer 控制寄存器	0x90000004	0x0XXXXXXXX
DSU Trace 时间标志寄存器	0x90000008	0x00000000
AHB 断点寄存器 1	0x90000010	0xFFFFFFFF
AHB 屏蔽地址寄存器 1	0x90000014	0xFFFFFFFF
AHB 断点寄存器 2	0x90000018	0xFFFFFFFF
AHB 屏蔽地址寄存器 2	0x9000001c	0xFFFFFFFF
DSU 串口状态寄存器	0x800000c4	0x00000006
DSU 串口控制寄存器	0x800000c8	0x00000000
DSU 串口波特率重载寄存器	0x800000cc	0x0003FFFF

注：“复位值”按照十六进制统一写法“0xXXX0XXXX”，“0x”表示 16 进制，每 4bit 中有 1bit 不定值，即写为“X”，4bit 全部为固定值则写为相应 16 进制数值。

4.19 时钟配置

4.19.1 概述

BM3823A 的时钟配置是通过 PLL 实现的，处理器内部使用的时钟信号可以由外部时钟输入 clk 提供或者 clk 经过内部 PLL 处理后提供，处理器内部共有三个 PLL 单元，分别是系统时钟单元 MAIN_PLL、DDR2 时钟单元 DDR_PLL、MAC 时钟单元 MAC_PLL。

4.19.2 PLL 管脚说明及使用要求

PLL 内核电压为数字 1.2V，PLL IO 电压为模拟 3.3V，PLL 关键管脚共 21 个，具体说明见表 4-19-1。管脚 clk 为 MAIN_PLL、DDR_PLL、MAC_PLL 提供外部时钟输入，PLL 使能时，clk 输入时钟频率范围为 5MHz~100MHz，推荐输入晶振频率为 25MHz。

注意

- 1、如果使用 Ethernet MAC 功能，则输入晶振 clk 要求一定是 25MHz；
- 2、PLL IO 电压为模拟 3.3V，推荐与数字 3.3V 电源分开设计或者使用磁珠等进行隔离，目前数字、模拟 3.3V 合用一个电源时还未出现问题。

表 4-19-1 PLL 相关管脚使用说明

管脚	I/O 类型	上拉/下拉	功能说明
clk	I	—	系统时钟，PLL 不使能时直接输出给系统使用；PLL 使能时，为 PLL 的输入时钟，经过 PLL 处理后输出给系统。
pll_bp	I	—	PLL 时钟旁路控制，高电平时旁路 PLL，片内时钟由输入晶振 clk 提供；低电平时，片内时钟由输入 clk 经过 PLL 处理后提供。
ddr_pll[2:0]	I	—	DDR2 时钟 PLL 分频系数，[0]为最低位，[2]为最高位，3 位不可同时配置成 0。
pll_din[2:0]	I	—	系统主时钟 PLL 分频系数，[0]为最低位，[2]为最高位，3 位不可同时配置成 0。
pll_fout4phasepd	I	—	主时钟、DDR2 时钟及 MAC 时钟三个 PLL 的 8 分频时钟掉电使能，有效时三个 PLL 的 8 分频时钟没有输出，高电平有效（测试用管脚，用户不用关心，接 4.7kΩ 电阻上拉即可，节省功耗）。
pll_lock_ddr	O	—	DDR2 时钟 PLL 锁定指示信号，高电平时表示 DDR2 PLL 时钟已经锁定。
pll_lock_mac	O	—	MAC 时钟 PLL 锁定指示信号，高电平时

管脚	I/O 类型	上拉/ 下拉	功能说明
			表示 MAC PLL 时钟已经锁定。
pll_lock_main	O	—	主时钟 PLL 锁定指示信号, 高电平时表示主系统 PLL 时钟已经锁定。
pll_min[2:0]	I	—	主系统 PLL 和 DDR2 PLL 共用的时钟倍频系数, [0]为最低位, [2]为最高位, 3 位不可同时配置成 0。
pll_opt	I	—	PLL 测试加固模式使能, 高电平有效。低电平时, 输出时钟质量更好, 在高频下表现更明显。测试用信号, 用户板级设计时通过 4.7kΩ 电阻下拉即可。
pll_out_ddr_d8	O	—	DDR2PLL 输出 8 分频时钟。
pll_out_mac_d8	O	—	MACPLL 输出 8 分频时钟。
pll_out_main_d8	O	—	主系统 PLL 输出 8 分频时钟。
pll_pd_main	I	—	主系统时钟 PLL 掉电使能, 有效时主系统 PLL 不输出主系统时钟信号, 高电平有效。不使用系统时钟时可以通过上拉该信号降低功耗。
pll_pd_ddr	I	—	DDR2 时钟 PLL 掉电使能, 有效时 DDR2 PLL 不输出 DDR2 时钟信号, 高电平有效。不使用 DDR2 时, 该信号仍然要下拉, 这样可以保证不使用 DDR2 时所有双向管脚都为输出状态, 板级设计时可以悬空。
pll_pd_mac	I	—	MAC 时钟 PLL 掉电使能, 有效时 MAC PLL 不输出 MAC 时钟信号, 高电平有效。不使用 MAC 模块时, 可以通过上拉该信号降低功耗。

⚠ 注意

- 1、pll_opt 为测试使用, 板级设计时通过 4.7kΩ 电阻下拉;
- 2、pll 输出时钟信号 pll_out_main_d8、pll_out_ddr_d8、pll_out_mac_d8 板级设计时悬空;

3、当 pll_bp 设置为低电平时，时钟由内部 PLL 处理后提供。为保证 PLL 稳定工作，外部时钟 clk 的选择必须同时满足以下三个条件：

- ①输入频率范围 5MHz~100MHz；
- ②占空比范围 45%-55%；
- ③ $25\text{MHz} \leq f_{\text{clk}} \times \text{pll\_min} \leq 100\text{MHz}$ ；

4、pll_min[2:0]、pll_din[2:0]、ddr_plld[2:0]不能设置为 0。

4.19.3 PLL 频率计算

MAIN_PLL 输出频率 $f_{\text{main\_pll}}$ 的计算公式为：

$$f_{\text{main\_pll}} = f_{\text{clk}} \times \text{pll\_min} \times 16 \div \text{pll\_din}$$

系统时钟时 MAIN_PLL 的 2 分频时钟输出，因此系统时钟频率 f_{system} 的计算公式为：

$$f_{\text{system}} = f_{\text{main\_pll}} \div 2 = f_{\text{clk}} \times \text{pll\_min} \times 8 \div \text{pll\_din}$$

DDR2 时钟是 DDR_PLL 的输出频率 $f_{\text{ddr\_pll}}$ ，计算公式为：

$$f_{\text{ddr\_pll}} = f_{\text{clk}} \times \text{pll\_min} \times 16 \div \text{ddr\_plld}$$

MAC 时钟是 MAC_PLL 的输出频率 $f_{\text{mac\_pll}}$ ，计算公式为：

$$f_{\text{mac\_pll}} = f_{\text{clk}} \times 20$$

管脚 pll_out_main_d8、pll_out_ddr_d8、pll_out_mac_d8 分别为三个 PLL：MAIN_PLL、DDR_PLL、MAC_PLL 的 8 分频时钟测试输出信号，其输出频率分别为：

$$f_{\text{pll\_out\_main\_d8}} = f_{\text{main\_pll}} \div 8$$

$$f_{\text{pll\_out\_ddr\_d8}} = f_{\text{ddr\_pll}} \div 8$$

$$f_{\text{pll\_out\_mac\_d8}} = f_{\text{mac\_pll}} \div 8$$

注意

- 1、上述公式中用到的 f_{clk} 表示输入时钟信号 clk，即晶振输入；
- 2、pll_min表示管脚配置 pll_min[2:0]，pll_din表示管脚配置 pll_din[2:0]，ddr_plld表示管脚配置 ddr_plld[2:0]；
- 3、pll_out_main_d8、pll_out_ddr_d8、pll_out_mac_d8 为测试用时钟，用户不需要关心，板级设计时悬空即可。

4.20 JTAG 接口

4.20.1 概述

BM3823A 实现了符合 IEEE 1149.1 JTAG 规范的标准接口，此接口主要用于板级 JTAG 测试。

4.20.2 JTAG 使用要求

JTAG 接口包括 5 个管脚，如下表 4-20-1 所示。

表 4-20-1 JTAG 管脚

管脚名称	I/O 类型	上拉/下拉	说明
tck	I	下拉	tck 是时钟信号，tck 与系统时钟独立。
tms	I	上拉	tms 是模式控制信号，与 tck 同步。
tdi	I	上拉	tdi 是串行输入数据信号，tdi 与 tck 同步。
tdo	O	——	tdo 是串行输出数据信号，tdo 与 tck 同步。
trst	I	上拉	trst 用于复位 JTAG，低电平有效，trst 与 tck 异步。不使用 JTAG 时该信号通过 1k Ω 电阻下拉。

注意 不使用 JTAG 时需确保 trst 下拉，JTAG 不使用时推荐接法如图 4-20-1 所示。

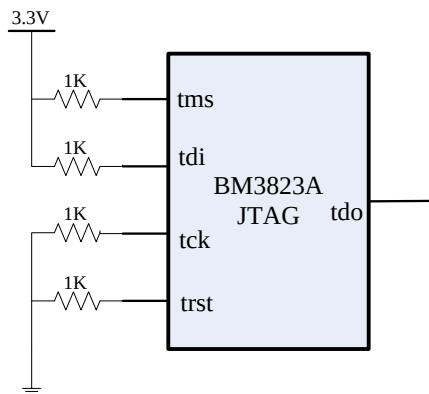


图 4-20-1 不使用 JTAG 时推荐的板级连接方式

不使用 JTAG 时，tms 和 tdi 管脚通过 1K 上拉电阻连接到 3.3V 的电源，tck 和 trst 通过 1K 下拉电阻接地。

使用 JTAG 时，四个输入管脚中的 tms、tdi、tck 直接连接其外部 JTAG 输入信号，trst 管脚推荐采用如图 4-20-2 的接法。

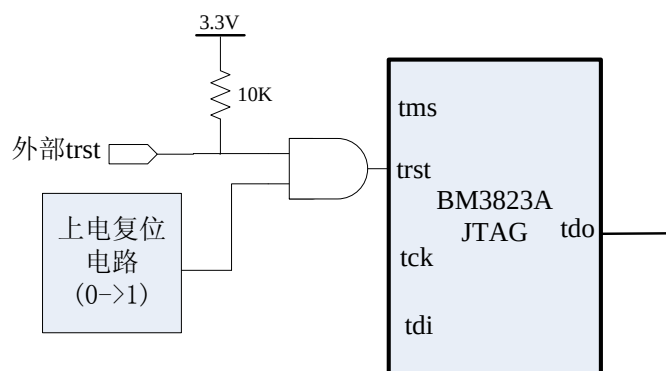


图 4-20-2 使用 JTAG 时推荐的 trst 管脚连接方式

图 4-20-2 中上电复位电路为板级上电复位电路，“外部 trst”为外部系统 JTAG 输入信号中的 trst，并通过 10K 电阻上拉，上电复位电路的输出和“外部 trst”作为与门的输入，与门的输出连接 BM3823A 的 trst 管脚。

五、产品电特性

表 5-1 直流参数、功能、开关参数

参数	符号	条件 (除另有规定外, -55℃≤T _A ≤125℃, 1.08V≤V ₁₂ ≤1.32V, 1.62V≤V ₁₈ ≤1.98V, 2.97V≤V ₃₃ ≤3.63V)		分组	极限值		单位	
					最小	最大		
输出高电平电压	V _{OH}	V ₁₂ =1.08V, V ₁₈ =1.62V, V ₃₃ =2.97V,	I _{OH} =-0.5 mA	A1 A2 A3	2.7	—	V	
输出低电平电压	V _{OL}		测所有适用的 输出端。		I _{OH} = -4、-8、-16 mA	2.4		—
					I _{OH} = -4 mA	0.95		—
		I _{OL} =1.5 mA			—	0.3	V	
		I _{OL} = 4、8、16 mA			—	0.5		
		I _{OL} = 4 mA			—	0.65		
输入高电平电压	V _{IH}	测所有适用的 输入端。	3.3V 输入端		2.0	—	V	
输入低电平电压	V _{IL}		1.8V 输入端		1.1	—	V	
			3.3V 输入端		—	0.8	V	
			1.8V 输入端		—	0.65	V	
输入高电平漏电流	I _{IH}	V ₁₂ = 1.32V, V ₁₈ = 1.98V, V ₃₃ = 3.63V, V _I = 3.63V	T _A =25℃		—	0.1	μA	
			T _A =-55℃或 T _A =125℃		—	1	μA	
上拉输入高电平漏电流	I _{IHPU}	V ₁₂ = 1.32V, V ₁₈ = 1.98V, V ₃₃ = 3.63V, V _I = 3.63V			—	5	μA	
下拉输入高电平漏电流	I _{IHPD}				—	200	μA	
输入低电平漏电流	I _{IL}	V ₁₂ = 1.32V, V ₁₈ = 1.98V, V ₃₃ = 3.63V, V _I = 0V	T _A =25℃		—	0.1	μA	
			T _A =-55℃或 T _A =125℃		—	1	μA	
上拉输入低电平漏电流	I _{ILPU}	V ₁₂ = 1.32V, V ₁₈ = 1.98V, V ₃₃ = 3.63V, V _I = 0V			—	200	μA	
下拉输入低电平漏电流	I _{ILPD}				—	5	μA	

表 5-1（续）

参数	符号	条件 (除另有规定外, -55℃≤T _A ≤125℃, 1.08V≤V ₁₂ ≤1.32V, 1.62V≤V ₁₈ ≤1.98V, 2.97V≤V ₃₃ ≤3.63V)			分组	极限值		单位
						最小	最大	
三态输出高电平漏电流	I _{OZH}	V ₁₂ = 1.32V, V ₁₈ = 1.98V, V ₃₃ = 3.63V	V _O =3.63V	25℃	A1 A2 A3	—	0.1	μA
				-55℃ 或 125℃		—	1	μA
三态输出低电平漏电流	I _{OZL}		V _O =0V	25℃		—	0.1	μA
				-55℃ 或 125℃		—	1	μA
待机电源电流	I _{DD(SB)}	V ₁₂ = 1.32V, V ₁₈ = 1.98V, V ₃₃ = 3.63V, (各时钟信号不工作, 处理器处于待机状态)	V ₁₂ 端			—	100	mA
			V ₁₈ 端			—	40	mA
			V _{33D} 端			—	30	mA
			V _{33A} 端			—	1	mA
输入电容	C _{IN}	T _A =25℃			A4	—	20	pF
输出电容	C _{IO}	T _A =25℃				—	20	pF
动态电源电流	I _{DD(dy)}	V ₁₂ = 1.32V, V ₁₈ = 1.98V,	V ₁₂ 端		A4 A5	—	800	mA



			V_{18} 端		—	200	mA

			V _{33D} 端		—	300	mA

		V_{33A} 端		—	10	mA
功能测试		$V_{12}=1.08V, V_{18}=1.62V, V_{33}=2.97V, V_{IH}=V_{33}, V_{IL}=0V, V_{OUT}=V_{33}/2$	A7 A8A A8B	—	—	—
		$V_{12}=1.2V, V_{18}=1.8V, V_{33}=3.3V, V_{IH}=V_{33}, V_{IL}=0V, V_{OUT}=V_{33}/2$				
		$V_{12}=1.32V, V_{18}=1.98V, V_{33}=3.63V, V_{IH}=V_{33}, V_{IL}=0V, V_{OUT}=V_{33}/2$				
功能测试频率	f	$V_{12}=1.08V, V_{18}=1.62V, V_{33}=2.97V, V_{IH}=V_{33}, V_{IL}=0V, V_{OUT}=V_{33}/2$		—	300	MHz
		$V_{12}=1.2V, V_{18}=1.8V, V_{33}=3.3V, V_{IH}=V_{33}, V_{IL}=0V, V_{OUT}=V_{33}/2$				
		$V_{12}=1.32V, V_{18}=1.98V, V_{33}=3.63V, V_{IH}=V_{33}, V_{IL}=0V, V_{OUT}=V_{33}/2$				

表 5-1（续）

参数	符号	条件 (除另有规定外, $-55^{\circ}\text{C} \leq T_A \leq 125^{\circ}\text{C}$, $1.08\text{V} \leq V_{12} \leq 1.32\text{V}$, $1.62\text{V} \leq V_{18} \leq 1.98\text{V}$, $2.97\text{V} \leq V_{33} \leq 3.63\text{V}$)	分组	极限值		单位
				最小	最大	
pll 旁路情况 clk 周期	t_1	$V_{12}=1.08\text{V}$, $V_{18}=1.62\text{V}$, $V_{33}=2.97\text{V}$	A9 A10 A11	10	1000	ns
pll 允许下的 clk 周期	t_{1p}			10	200	ns
pll 旁路情况 clk 高低脉宽	t_2			4.5	550	ns
pll 允许下的 clk 高低脉宽	t_{2p}			4.5	110	ns
sdclk 周期	t_3			10	—	ns
address[27:0]输出延时	t_{10}			0.5	8	ns
data[31:0] and pardata[7:0] 写输出延时	t_{11}			0.5	8	ns
data[31:0] and pardata[7:0] 读建立时间	t_{12}			4.5	—	ns
data[31:0] and pardata[7:0] 读保持时间	t_{13}			0	—	ns
data[31:0] and pardata[7:0] 写输出保持时间	t_{14}			0.5	8	ns
oen 和 oe 输出延时	t_{15}			0.5	10	ns
writen、write、writen_1553、write_1553 输出延时	t_{17}			0.5	10	ns
romsn[1:0]输出延时	t_{18}			0.5	10	ns
ramsn[4:0]输出延时	t_{19}			0.5	10	ns
ramoen[4:0]输出延时	t_{20}			0.5	10	ns
rwen[3:0]、rbn[3:0]输出延时	t_{21}			0.5	10	ns
iosn[5:0]、ios[5:0]输出延时	t_{22}			—	10	ns
brdyn 建立时间	t_{23}			4	—	ns
brdyn 保持时间	t_{24}			0	—	ns
sdcasn 输出延时	t_{25}			0.5	8	ns
sdcsn[1:0]输出延时	t_{26}			0.5	8	ns
sdrasn 输出延时	t_{27}			0.5	8	ns
sdwen 输出延时	t_{28}			0.5	8	ns
sddqm[3:0]输出延时	t_{29}			0.5	8	ns
pio[31:0]输出延时	t_{31}			0.5	10	ns
pio[31:0]建立时间	t_{32}			4.5	—	ns
pio[31:0]读保持时间	t_{33}			0	—	ns
pio[31:0]写保持时间	t_{34}			0.5	—	ns
bexcn 建立时间	t_{35}			4	—	ns
bexcn 保持时间	t_{36}			0	—	ns

表 5-1（续）

参数	符号	条件 (除另有规定外, $-55^{\circ}\text{C} \leq T_A \leq 125^{\circ}\text{C}$, $1.08\text{V} \leq V_{12} \leq 1.32\text{V}$, $1.62\text{V} \leq V_{18} \leq 1.98\text{V}$, $2.97\text{V} \leq V_{33} \leq 3.63\text{V}$)	分组	极限值		单位
				最小	最大	
pci_clk 周期	t_{101}	$V_{12}=1.08\text{V}$, $V_{18}=1.62\text{V}$, $V_{33}=2.97\text{V}$	A9 A10 A11	30	—	ns
pci_clk 低脉宽和高脉宽	t_{102}			14.5	—	ns
pci_ad[31:0] 和 pci_cbe_n[3:0] 输出延时	t_{110}			4	12	ns
pci_ad[31:0] 和 pci_cbe_n[3:0] 建立时间	t_{111}			6	—	ns
pci_ad[31:0] 和 pci_cbe_n[3:0] 保持时间	t_{112}			0	—	ns
pci_frame_n, pci_par, pci_perr_n, pci_serr_n, pci_stop_n, 和 pci_devsel_n 输出延时	t_{113}			4	11	ns
pci_iridy_n 和 pci_trdy_n 输出延时	t_{114}			4	11	ns
pci_req_n 输出延时	t_{115}			4	12	ns
pci_frame_n, pci_par, pci_perr_n, pci_serr_n, pci_stop_n, pci_idsel_n,和 pci_devsel_n 建立时间	t_{116}			7	—	ns
pci_iridy_n 和 pci_trdy_n 建立时间	t_{117}			7	—	ns
pci_gnt_in_n 建立时间	t_{118}			6	—	ns
pci_frame_n, pci_par, pci_perr_n, pci_serr_n, pci_stop_n, pci_idsel_n,和 pci_devsel_n 保持时间	t_{119}			0	—	ns
pci_iridy_n 和 pci_trdy_n 保持时间	t_{120}			0	—	ns
pci_gnt_in_n 保持时间	t_{121}			0	—	ns

表 5-1（续）

参数	符号	条件 (除另有规定外, -55℃≤T _A ≤125℃, 1.08V≤V ₁₂ ≤1.32V, 1.62V≤V ₁₈ ≤1.98V, 2.97V≤V ₃₃ ≤3.63V)	分组	极限值		单位
				最小	最大	
gmac_phy_clk_rx 周期	t ₂₀₁	V ₁₂ =1.08V, V ₁₈ =1.62V, V ₃₃ =2.97V	A9 A10 A11	40	—	ns
gmac_phy_rxdv_i、gmac_phy_rxer_i 和 gmac_phy_rxd_i[3:0]建立时间	t ₂₀₂			10	—	ns
gmac_phy_rxdv_i、gmac_phy_rxer_i 和 gmac_phy_rxd_i[3:0]保持时间	t ₂₀₃			10	—	ns
gmac_phy_clk_tx 周期	t ₂₀₄			40	—	ns
gmac_phy_txd_o[3:0]输出延时	t ₂₀₅			1	10	ns
gmac_phy_txen_o 和 gmac_phy_txer_o 输出延时	t ₂₀₆			1	10	ns
gmac_phy_crs_i 和 gmac_phy_col_i 建 立时间	t ₂₀₇			10	—	ns
gmac_phy_crs_i 和 gmac_phy_col_i 保 持时间	t ₂₀₈			10	—	ns
ddr 控制、地址的建立时间	t ₃₀₁			0.2	—	ns
ddr 控制、地址的保持时间	t ₃₀₂			0.275	—	ns
ddr 的 dqs 的建立时间	t ₃₀₃			0.6	—	ns
ddr 的 dqs 的保持时间	t ₃₀₄			0.6	—	ns
ddr 中数据的建立时间	t ₃₀₅			0.1	—	ns
ddr 中数据的保持时间	t ₃₀₆			0.175	—	ns
ddr 中数据之间的组内偏差	t ₃₀₇			—	0.34	ns

注意

1、待机电源电流指处理器无时钟待机状态时的功耗，动态电源电流是处理器工作在 300MHz 最高频率时的功耗，上表中的指标是处理器最差工作条件时的界限值；

2、漏电指标中，上拉输入低电平漏电和下拉输入高电平漏电电流为 200 μA ，相对其他漏电较大。因为首先增加上下拉相当于并联电阻，会导致漏电流增大；其次上拉低电平和下拉高电平相当于形成了低阻通路，漏电流又有一定增大，故而此时漏电最大。

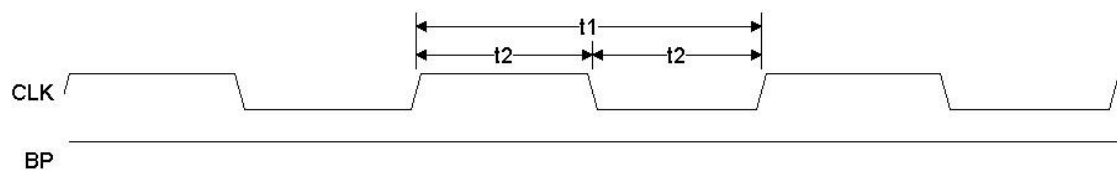


图 5-1 PLL 旁路情况下的时钟信号

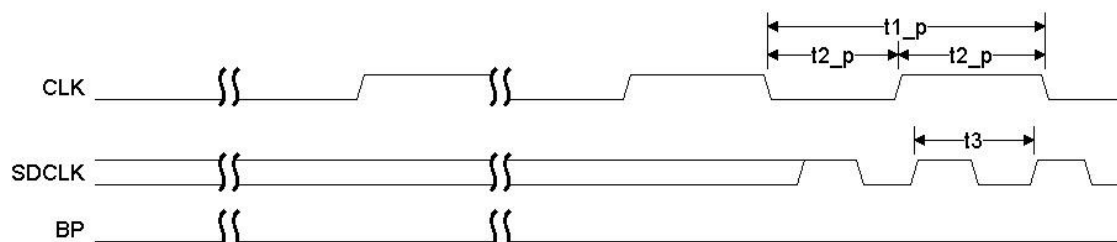


图 5-2 使用 PLL 情况下的时钟信号

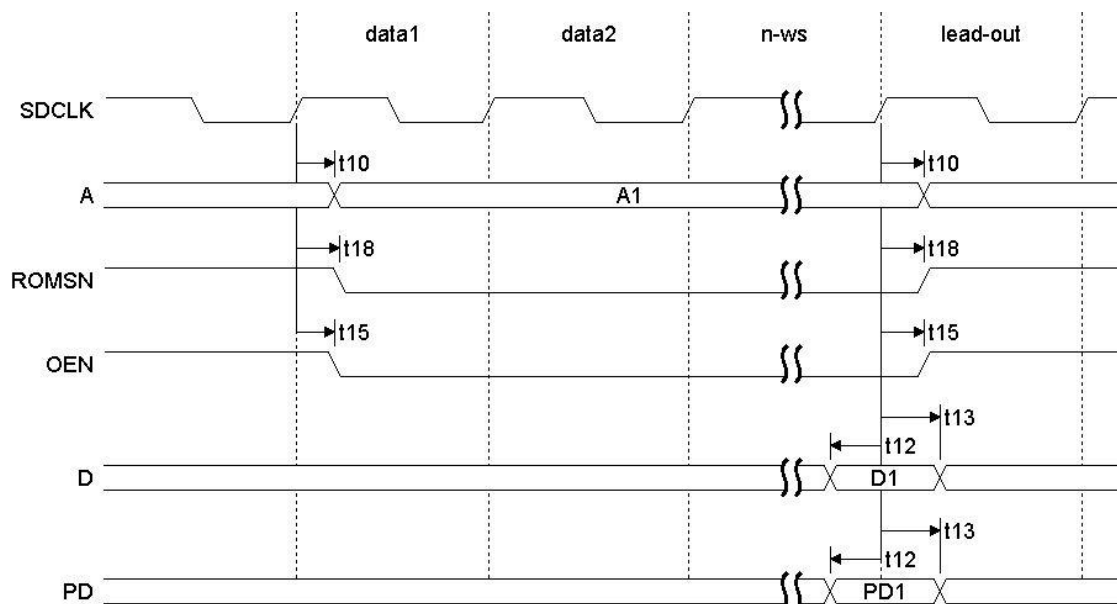


图 5-3 32 位 PROM 读时序

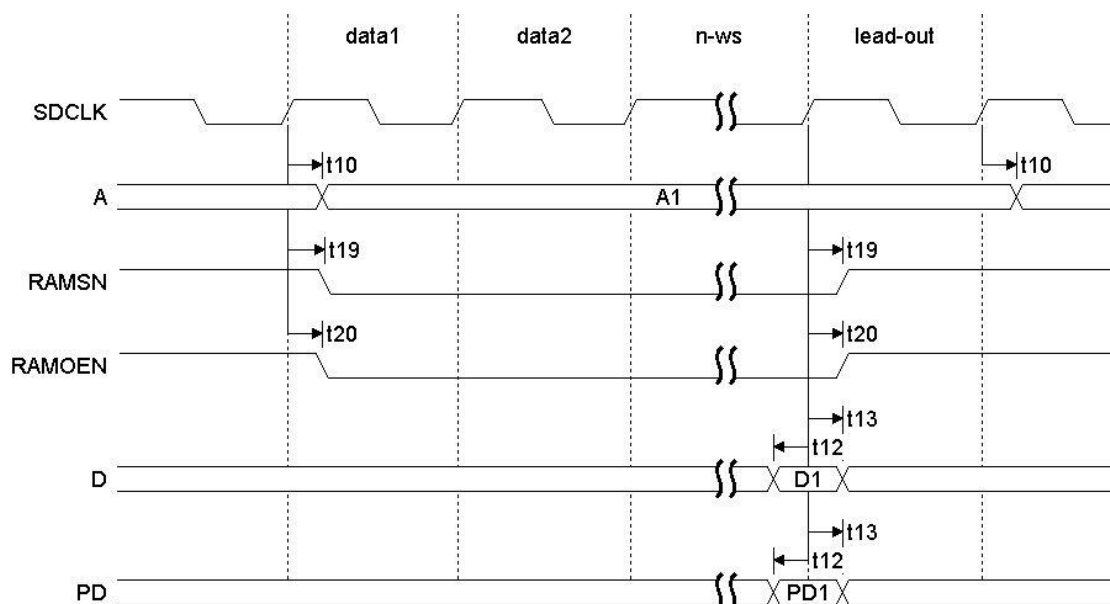


图 5-4 32 位 SRAM 读时序

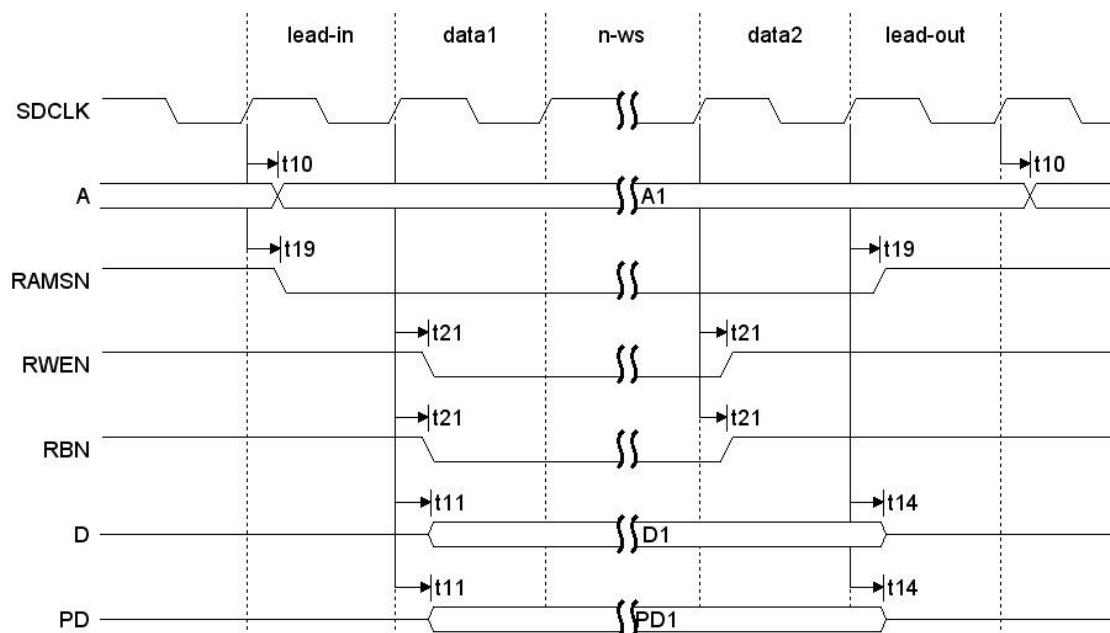


图 5-5 32 位 SRAM 写时序

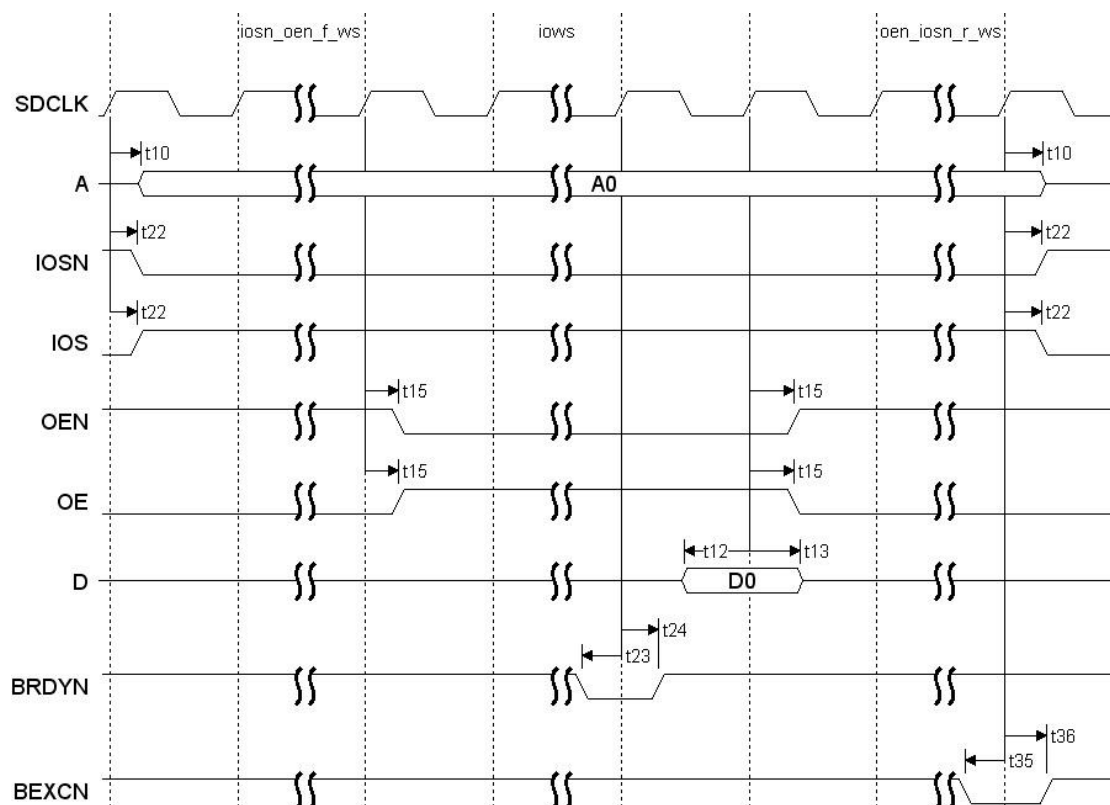


图 5-6 32 位 I/O 读时序

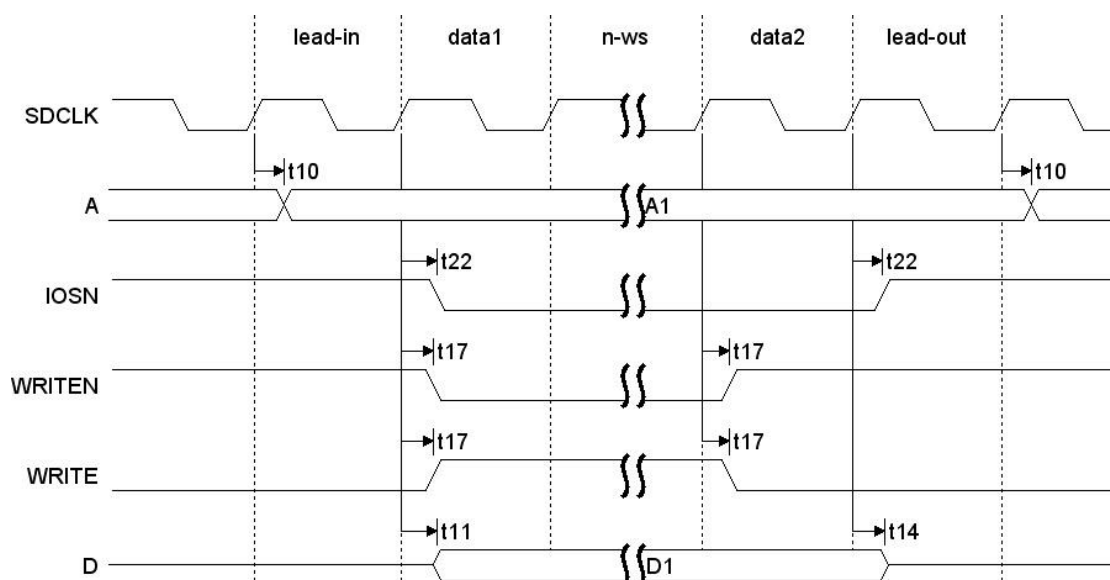


图 5-7 32 位 I/O 写时序

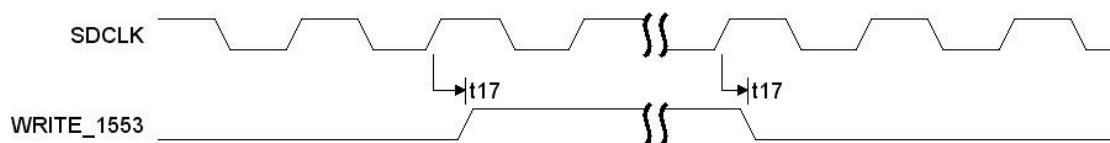


图 5-8 write_1553 时序

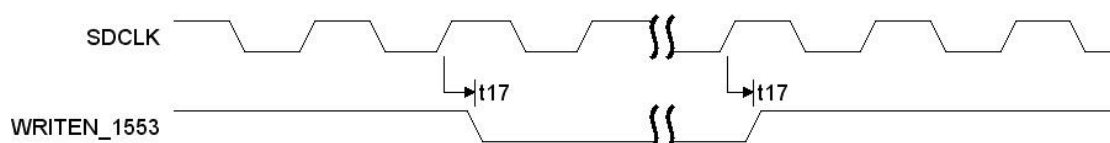


图 5-9 writen_1553 时序

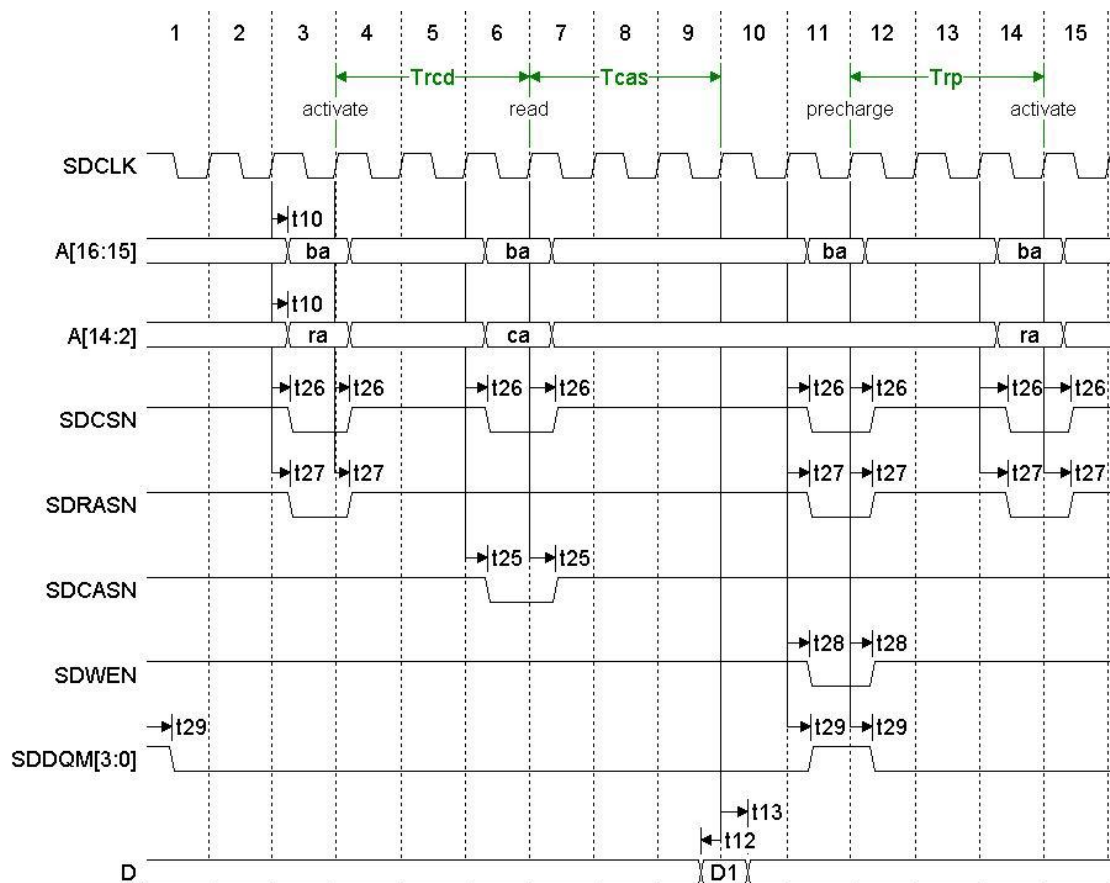


图 5-10 SDRAM 读时序

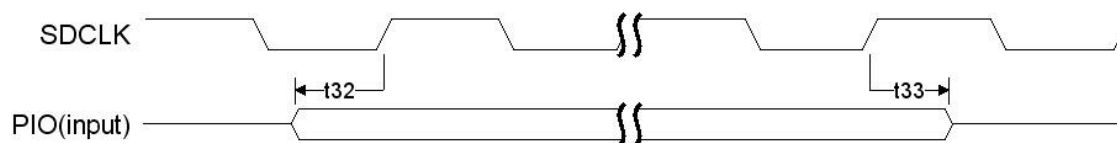


图 5-11 PIO 输入时序

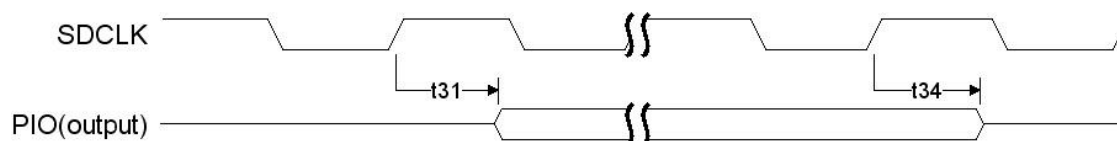


图 5-12 PIO 输出时序

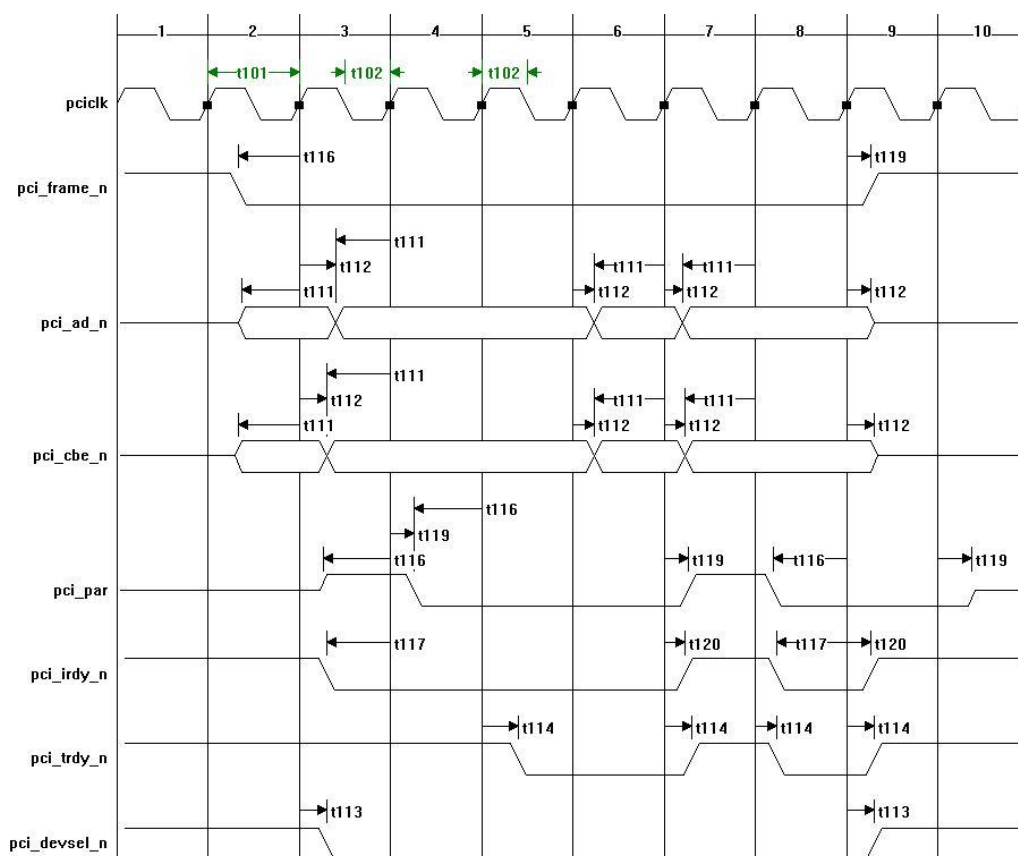


图 5-13 PCI 写时序

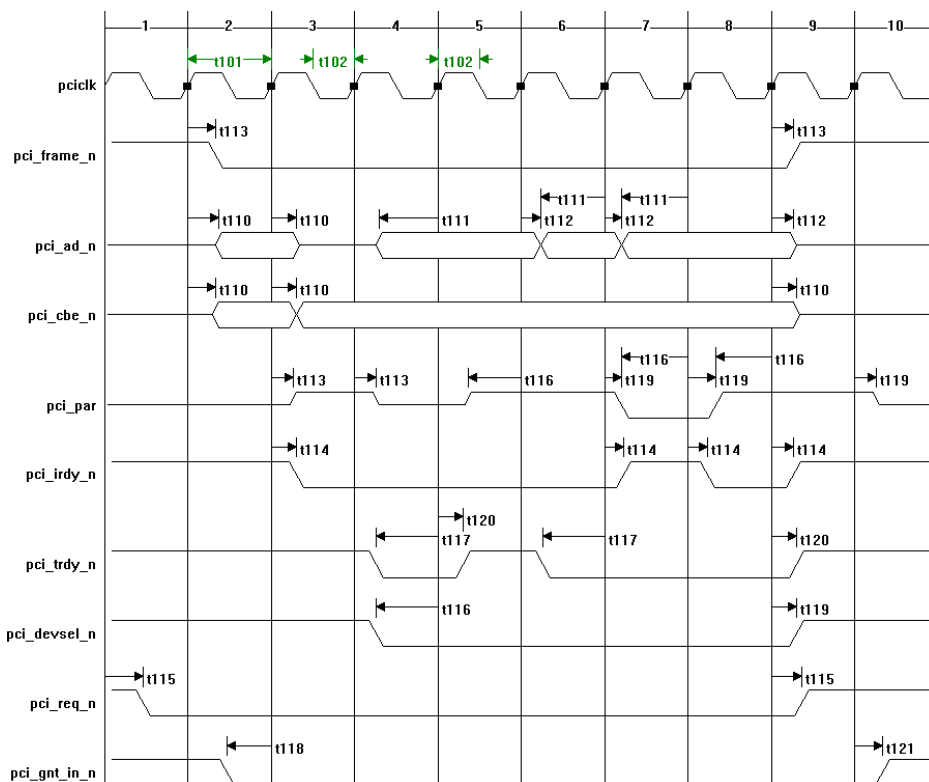


图 5-14 PCI 读时序

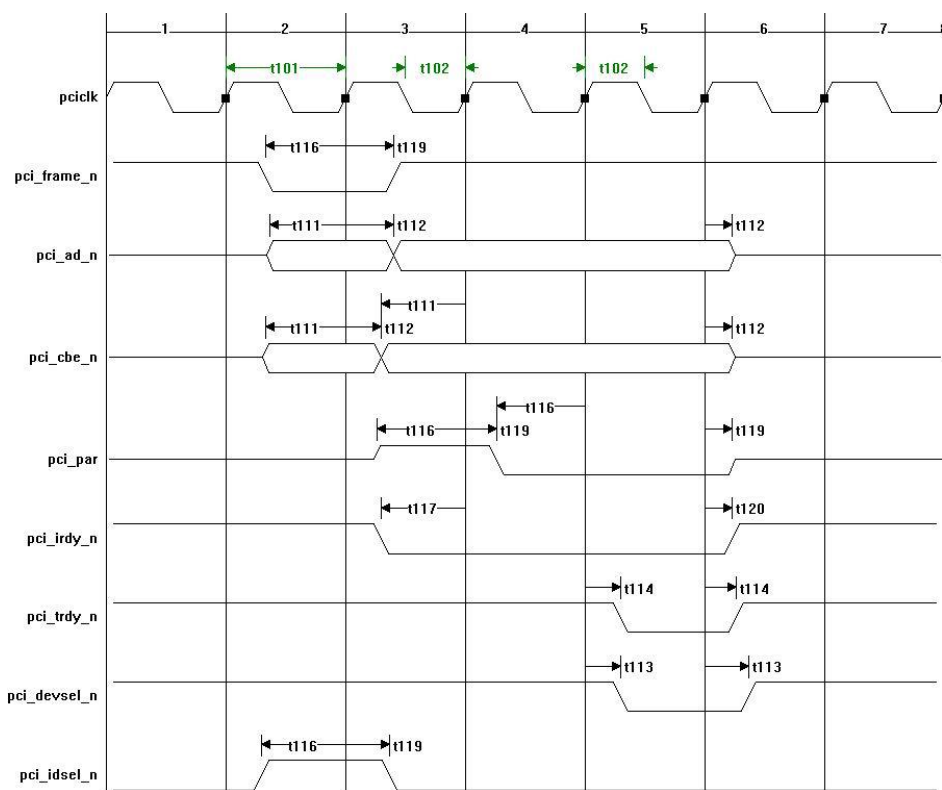


图 5-15 PCI 配置写时序

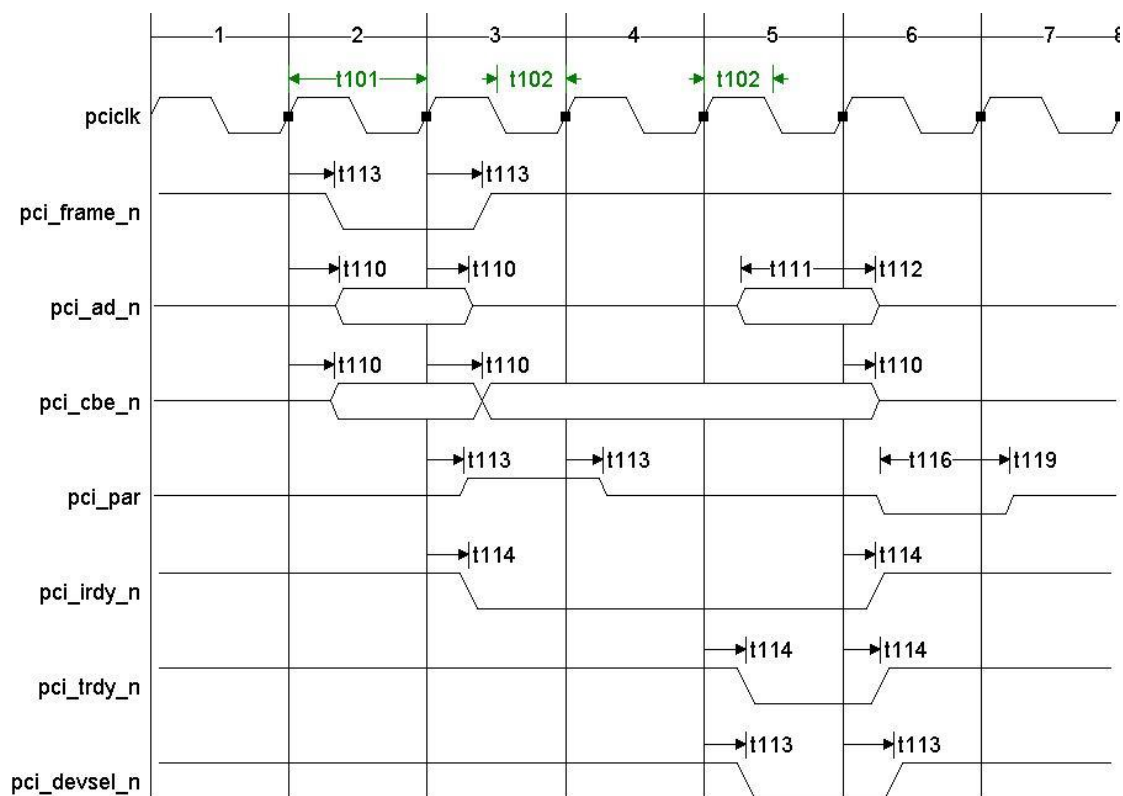


图 5-16 PCI 配置读时序

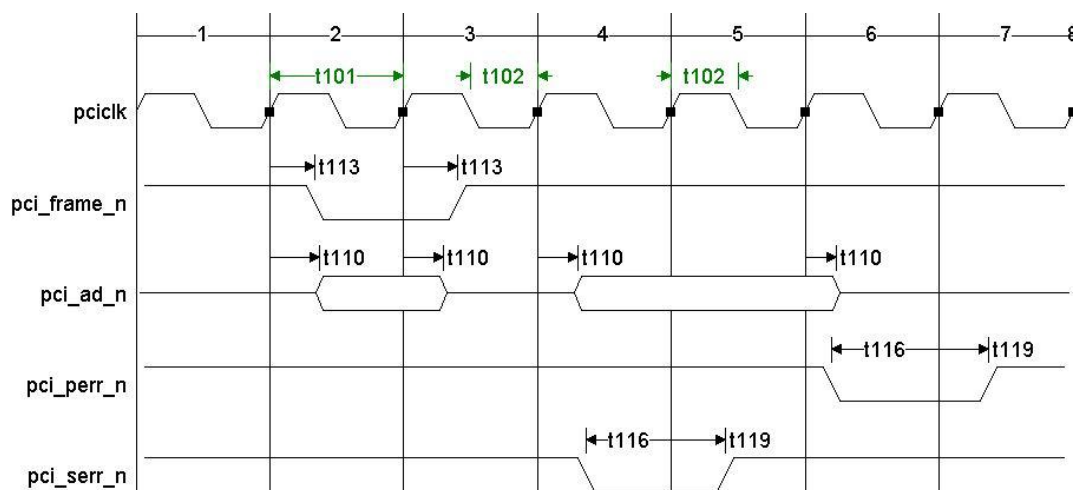
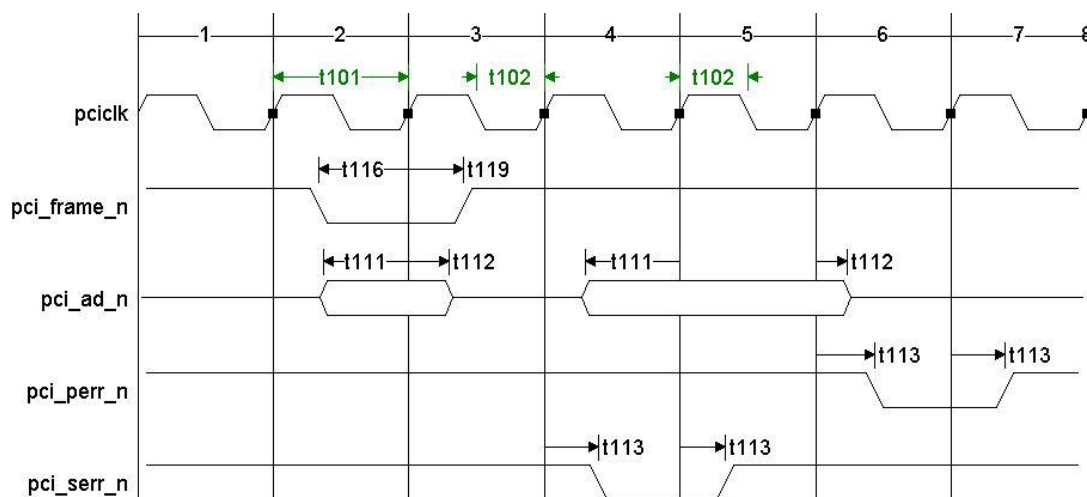
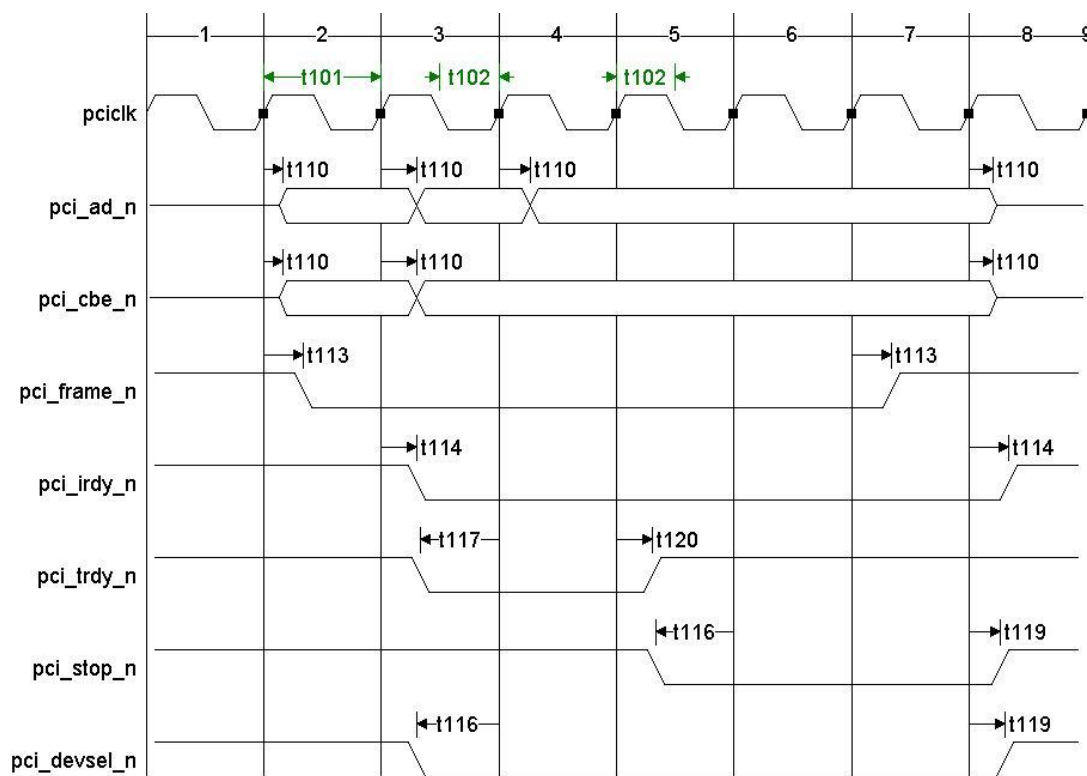


图 5-17 pci_perr_n 和 pci_serr_n 输入时序


图 5-18 pci_perr_n 和 pci_serr_n 输出时序

图 5-19 pci_stop_n 输入时序

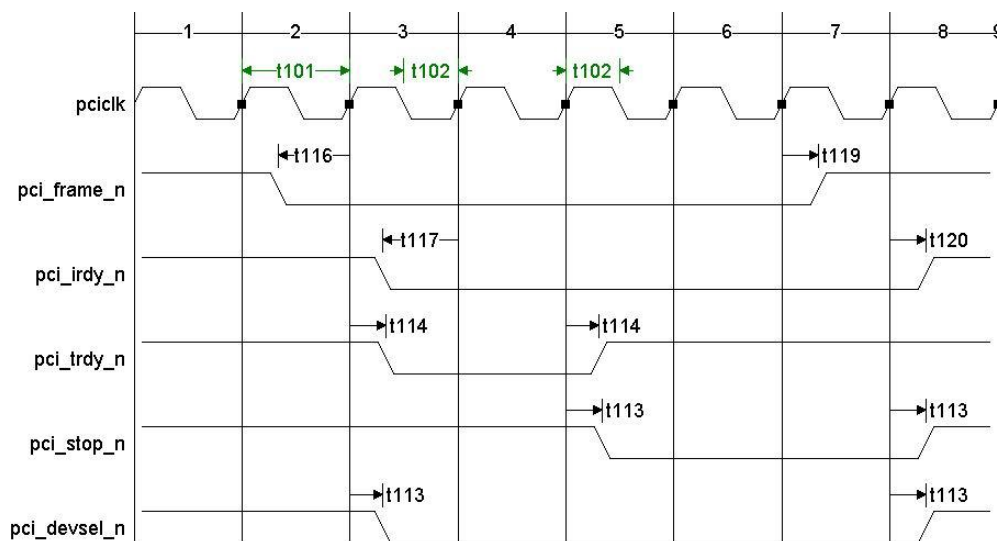


图 5-20 pci_stop_n 输出时序

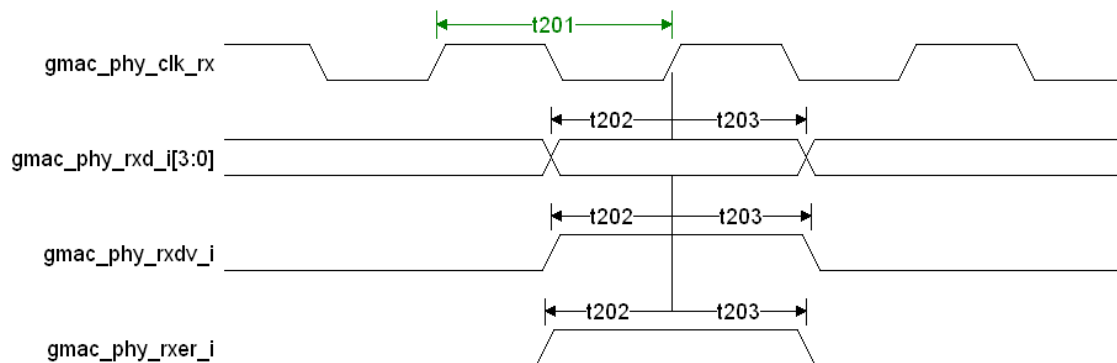


图 5-21 MII 接收时序

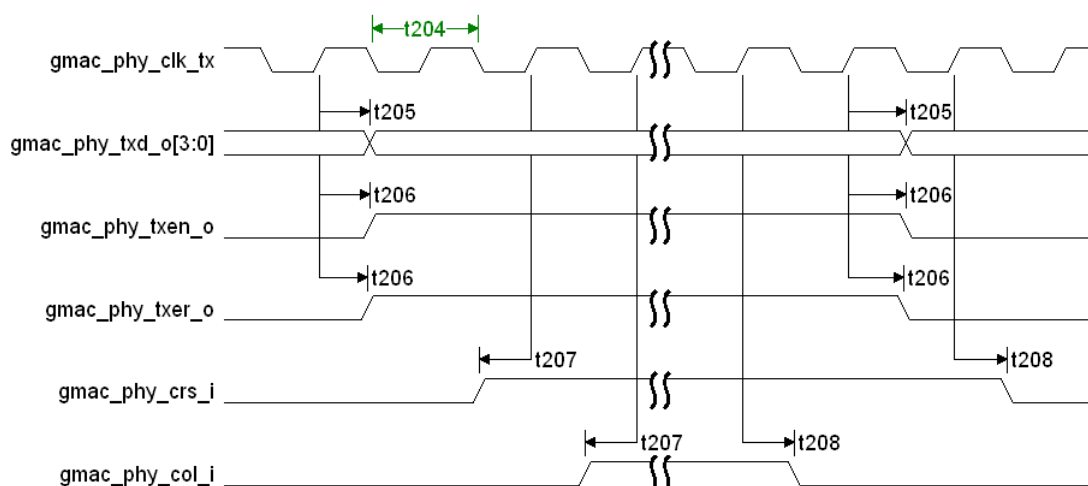


图 5-22 MII 发送时序

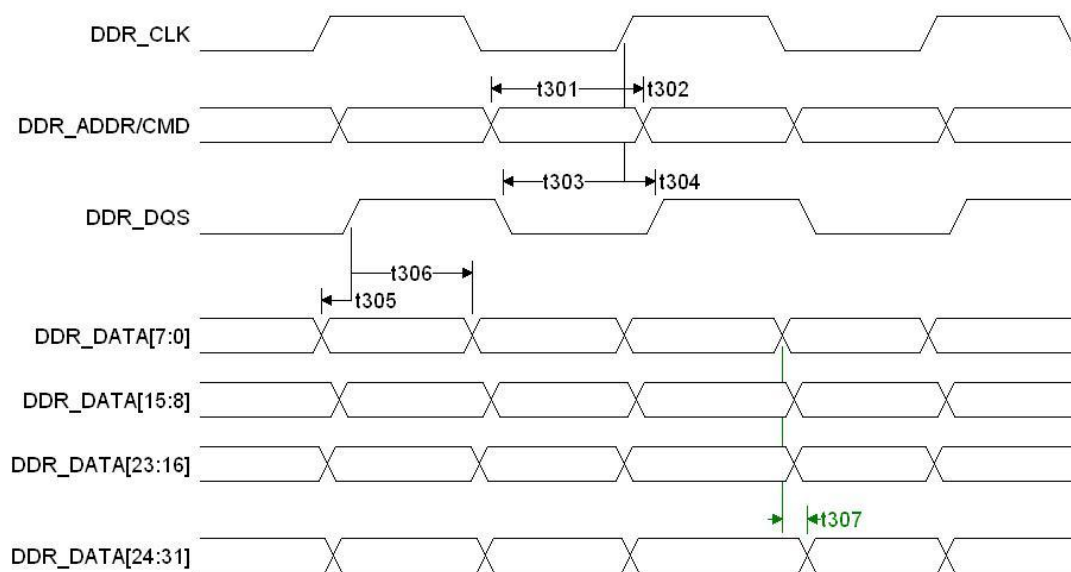


图 5-23 DDR2 时序

六、典型应用

选择 BM3823A 的典型功能，设计完成开发板。主要包括如下功能：

- ◆ 外部提供+5V 的电压输入；
- ◆ BM3823A 主频 300MHz；
- ◆ 板载 6MB FLASH（4MB 用于存储一般数据，2MB 用于存储校验数据）；
- ◆ 板载 3MB SRAM（2MB 用于存储一般数据，1MB 用于存储校验数据）；
- ◆ 板载 32MB SDRAM（16MB 存储一般数据，16MB 存储校验数据）；
- ◆ 板载 384MB DDR2 颗粒（256MB 存储一般数据，128MB 存储校验数据）；
- ◆ 提供 3 个通用异步串行接口，1 个 DSU 调试端口；
- ◆ 提供 GPIO 扩展接口；
- ◆ 提供 1 路 10/100/1000M 网络接口；
- ◆ 提供 2 路 SPI 接口（主模式）；
- ◆ 提供 2 路 I²C 总线接口（主/从模式）。

6.1 开发板说明

BM3823A 开发板外围硬件框图如图 6-1-1 所示。

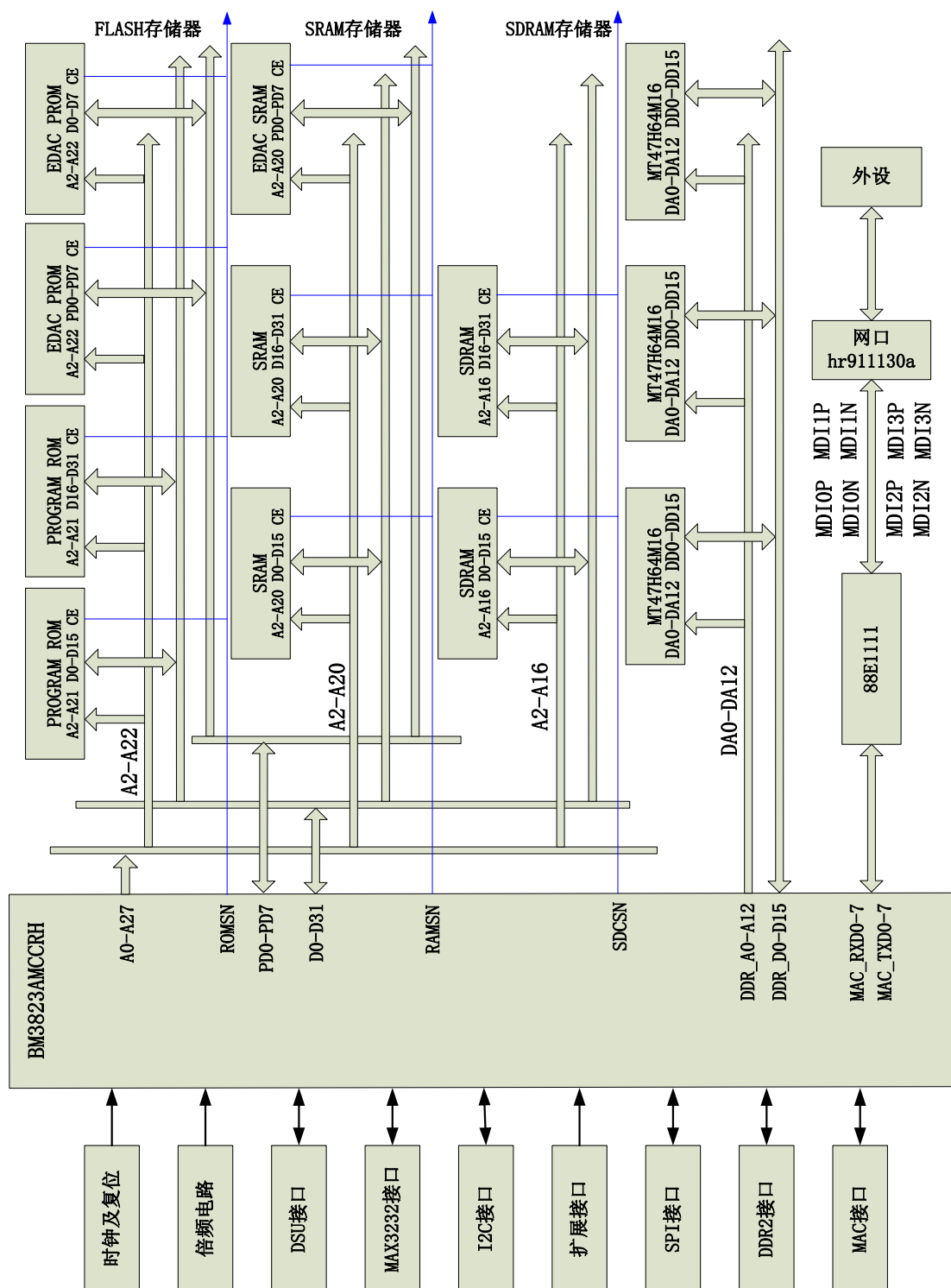


图 6-1-1 开发板外围硬件结构框图

6.2 开发板参考电路

6.2.1 电源模块

开发板采用 5V 电源供电，通过电源芯片分别输出模拟 3.3V（AVDD3.3V），数字 3.3V（DVDD3.3V），数字 1.8V（DVDD1.8V），数字 1.2V（DVDD1.2V），模拟 1.2V（AVDD1.2V）电源电压，以及网络接口器件的数字 2.5V（DVDD2.5V）和模拟 2.5V（AVDD2.5V）电源电压。原理图如图 6-2-1 所示。

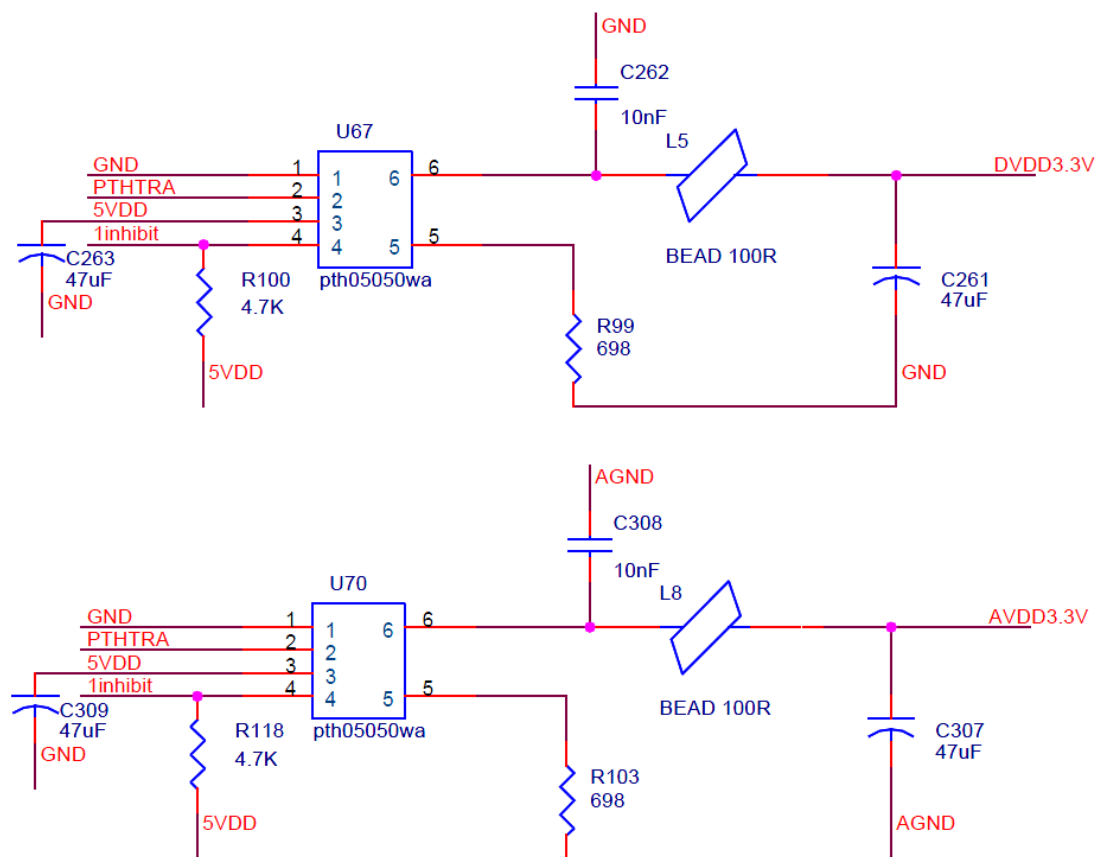


图 6-2-1 (a) 3.3V 电源原理图

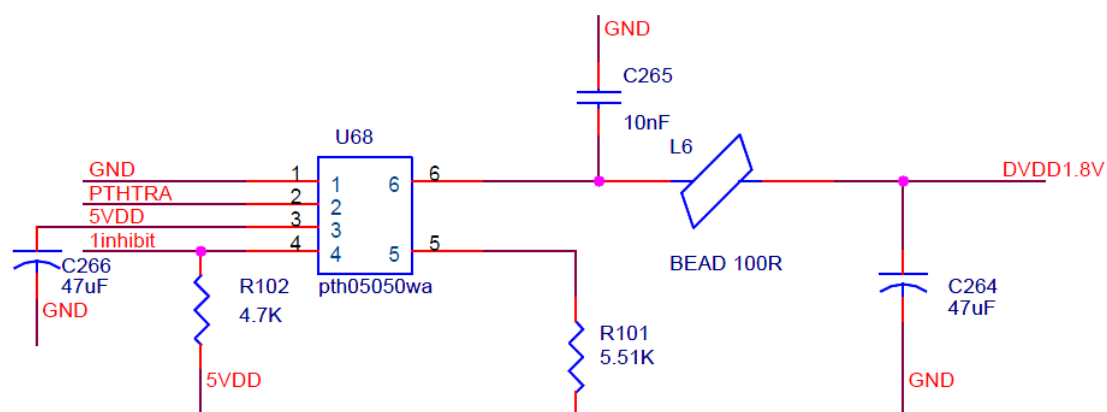


图 6-2-1 (b) 1.8V 电源原理图

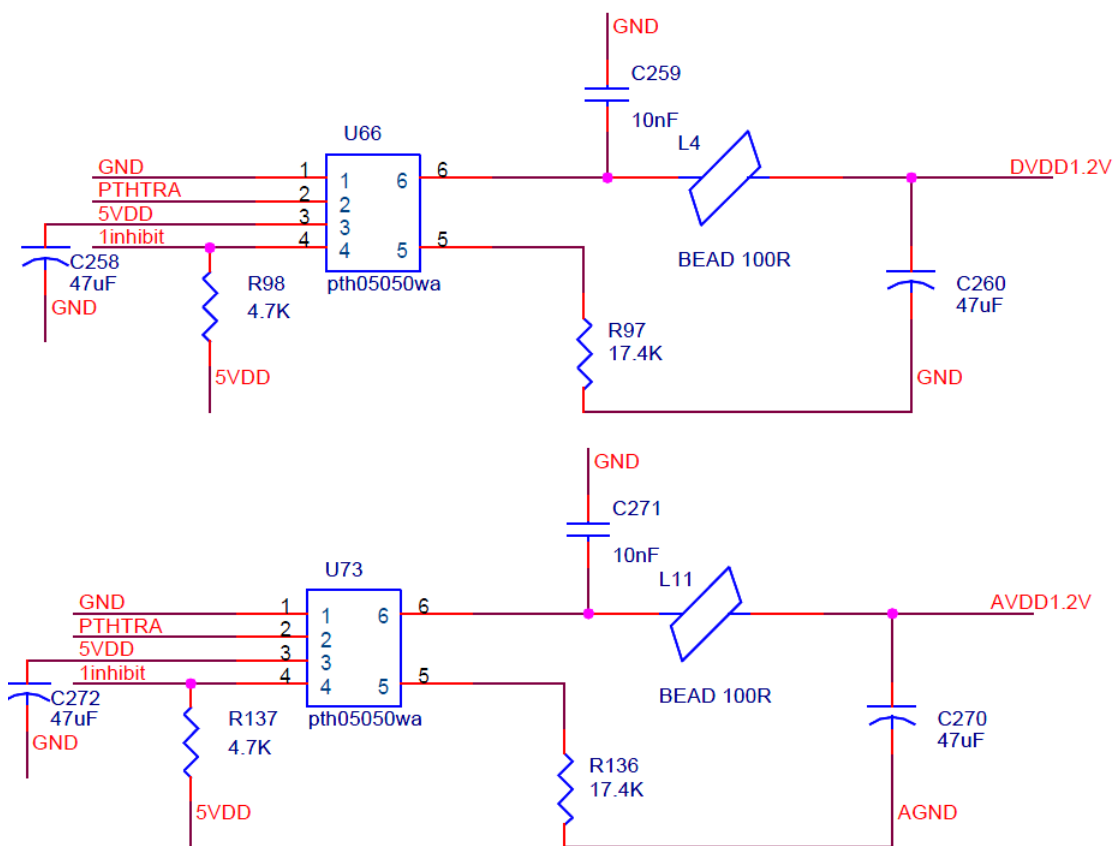


图 6-2-1 (c) 1.2V 电源原理图

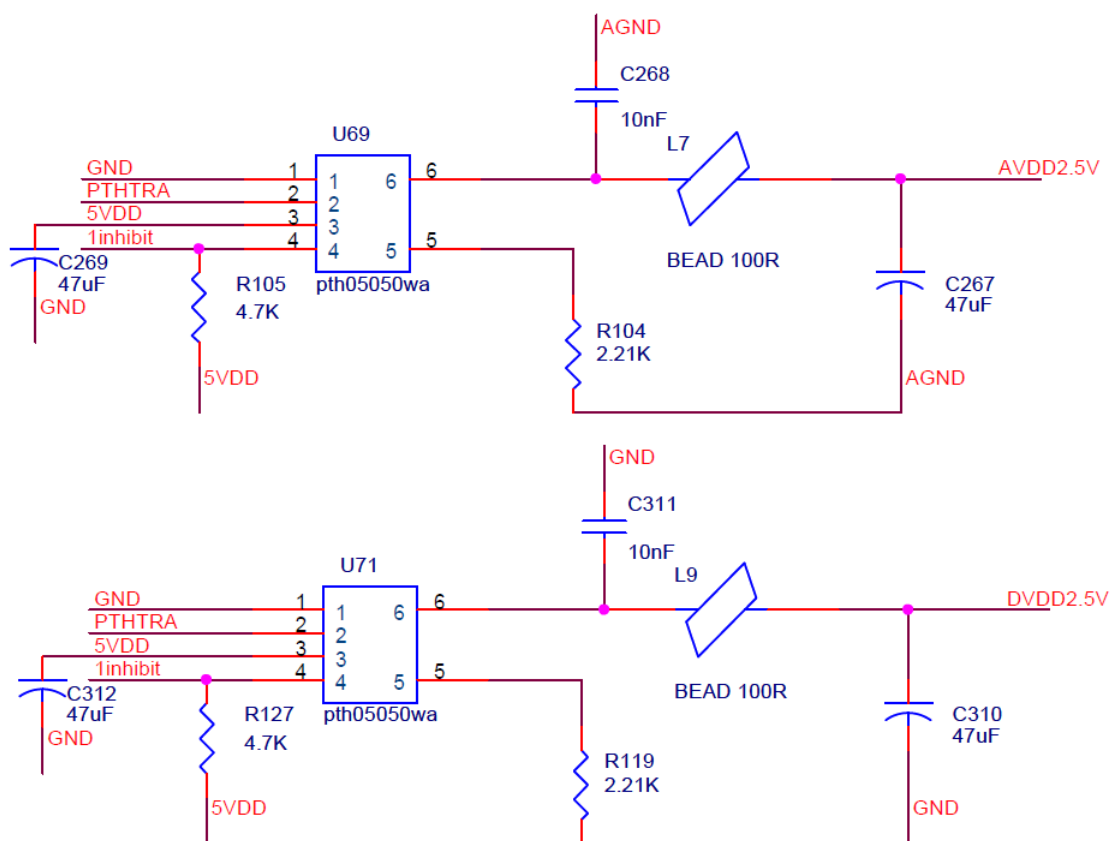


图 6-2-1 (d) 2.5V 电源原理图

图 6-2-1 电源原理图

6.2.2 时钟倍频电路及其工作模式

BM3823A 内部集成了倍频电路 (PLL)，PLL 电路可以配置处理器系统时钟、DDR 时钟和 MAC 时钟。PLL 电路是否进行锁相环时钟倍频操作，取决于时钟倍频电路是否使能，pll_bp 为 0，PLL 倍频使能；pll_bp 为 1，PLL 倍频禁止。

系统时钟由外部输入频率 clk、倍频系数 pll_min 和分频系数 pll_din 共同决定(其中 pll_min、pll_din 的范围在 1~7 之间，由硬件管脚进行配置)。系统时钟频率最大 300MHz。

系统时钟、DDR2 时钟及 MAC 时钟计算方法参考 4.19 章节时钟配置介绍，本开发板采用推荐的 25MHz 时钟晶振作为 clk 输入，DDR 时钟配置为 200MHz。MAC 时钟为 500MHz。

具体模式如表 6-2-1 所示：

表 6-2-1 PLL 信号配置管脚说明

符号	使用说明
clk	本板接 25MHz 晶振
pll_bp	拨向 ON 端时，是低电平；拨向数字端，是高电平。
pll_din_ddr[0]	[2:0]权位递减。拨向 ON 端时，是低电平；拨向数字端，是高电平
pll_din_ddr[1]	
pll_din_ddr[2]	
pll_din[0]	[2:0]权位递减，拨向 ON 端时，是低电平；拨向数字端，是高电平
pll_din[1]	
pll_din[2]	
pll_fout4phasepd	测试管脚，用户使用时上拉。拨向 ON 端时，是低电平；拨向数字端，是高电平
pll_lock_ddr	ddr 时钟 pll 稳定指示管脚
pll_lock_mac	mac 时钟 pll 稳定指示管脚
pll_lock_main	主时钟 pll 稳定指示管脚
pll_min[0]	[2:0]权位递减，拨向 ON 端时，是低电平；拨向数字端，是高电平
pll_min[1]	
pll_min[2]	
pll_opt	PLL 测试加固模式使能，高电平有效。低电平时，输出时钟质量更好，在高频下表现更明显。测试用信号，用户使用时下拉即可。拨向 ON 端时，是低电平下拉；拨向数字端，是高电平上拉。
pll_out_ddr_d8	测试管脚，可悬空
pll_out_mac_d8	测试管脚，可悬空
pll_out_main_d8	测试管脚，可悬空
pll_pd_main	用户使用 MAIN PLL 时下拉，拨向 ON 端时，是低电平；拨向数字端，是高电平
pll_pd_ddr	用户使用 DDR PLL 模块时下拉，拨向 ON 端时，是低电平；拨向数字端，是高电平
pll_pd_mac	用户使用 MAC PLL 模块时下拉，拨向 ON 端时，是低电平；拨向数字端，是高电平

注：在本电路板上，默认 CLK 连接 25MHz 有源晶振。

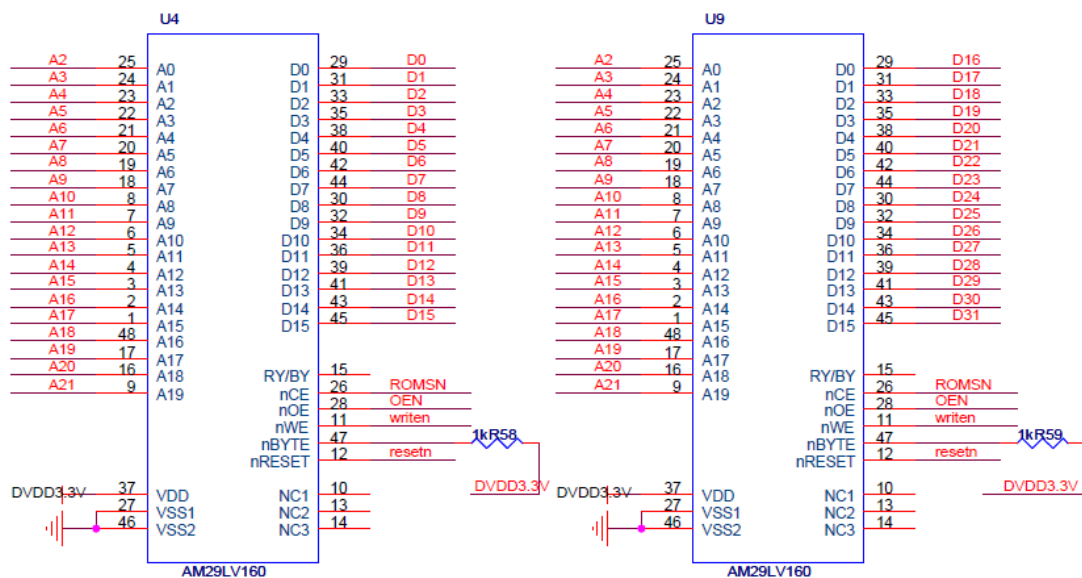


图 6-2-3 FLASH 系统参考原理图

6.2.5 SRAM

板载的 SRAM 系统采用三片型号为 IS61LV51216 (512K×16) 的 SRAM 组成 32 位宽 SRAM 系统, U2、U3 和 U7 为 SRAM, 其中 U2 和 U3 存放 32 位的数据存储器, U7 存放校验数据。SRAM 器件的高低字节选通信号通过 $rbn[3:0]$ 控制。SRAM 地址空间为: 0x2000-0000 到 0x201F-FFFF。原理图如图 6-2-4 所示。

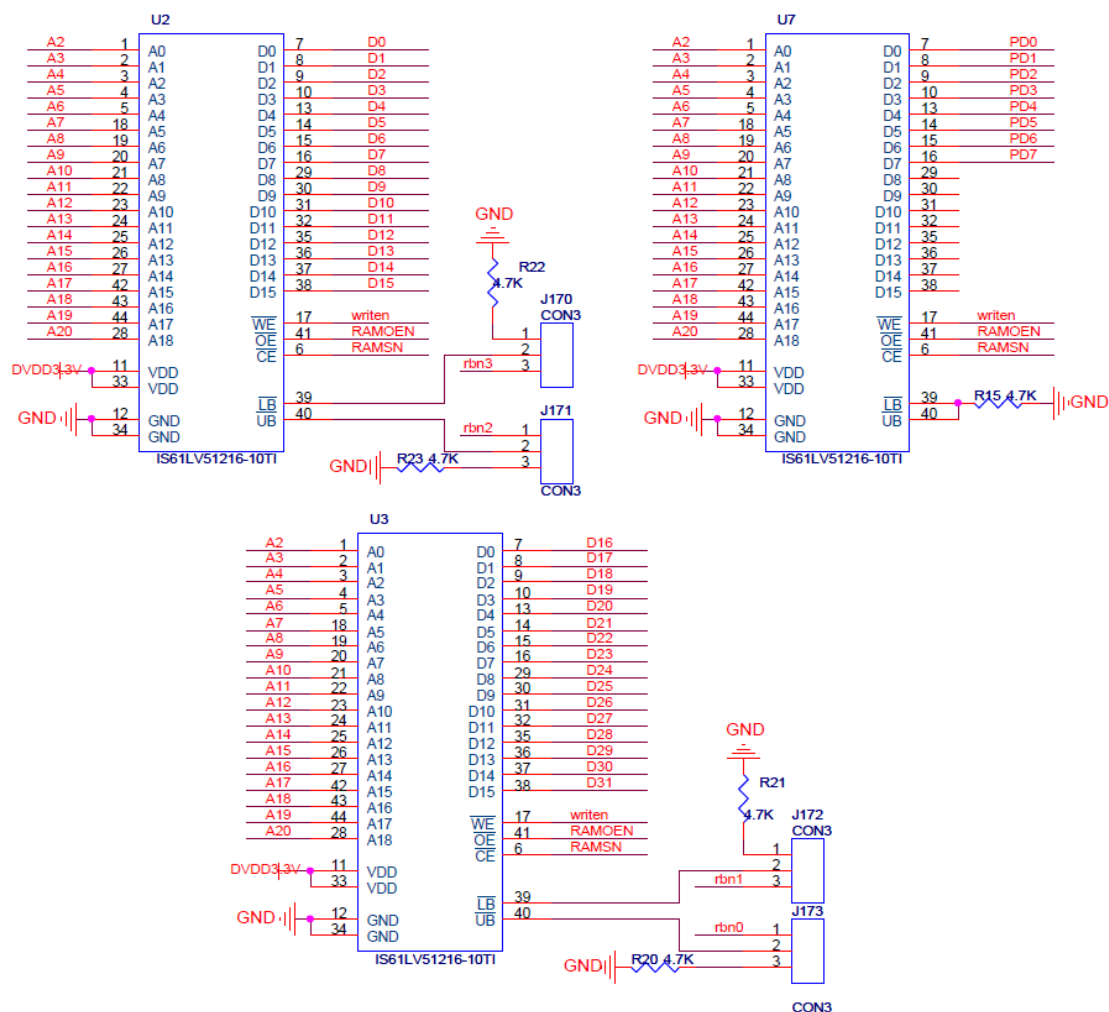


图 6-2-4 SRAM 系统参考原理图

6.2.7 SDRAM

板载的 SDRAM 采用两片型号为 IS42S32400(4M×32)的 SDRAM 器件，U82 和 U83，U82 作为数据存储区，U83 作为校验存储区。连接不同的 Bank 控制信号，SDRAM 地址空间不同。当选用 Bank0 时（sdcsn 选通 0），其地址空间为：0x3000-0000 到 0x30FF-FFFF；当选用 Bank1 时（sdcsn 选通 1），SDRAM 地址空间为：0x3100-0000 到 0x31FF-FFFF。原理图如图 6-2-5 所示。

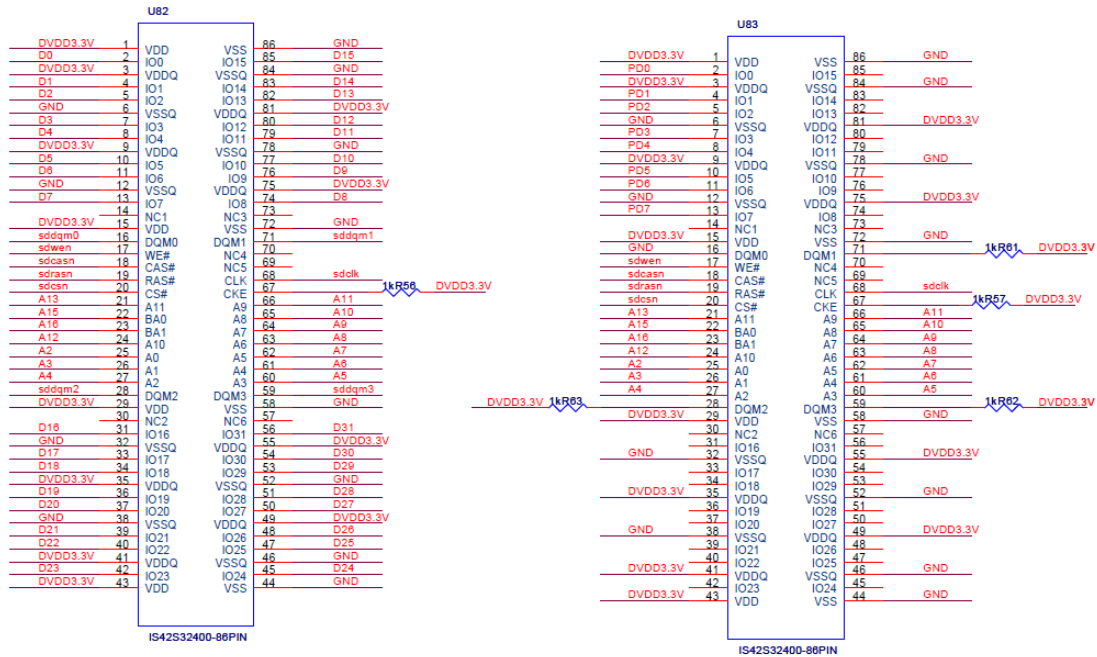


图 6-2-5 SDRAM 系统参考原理图

6.2.8 DDR2 SDRAM

板载 DDR2 SDRAM 的器件为三片 mt47h64m16 (64M×16)，其中两片组成 32 位位宽的数据存储器，另外一片存放 ECC 数据，地址范围为 0x40000000-0x4ffffff。原理图如图 6-2-6 所示。

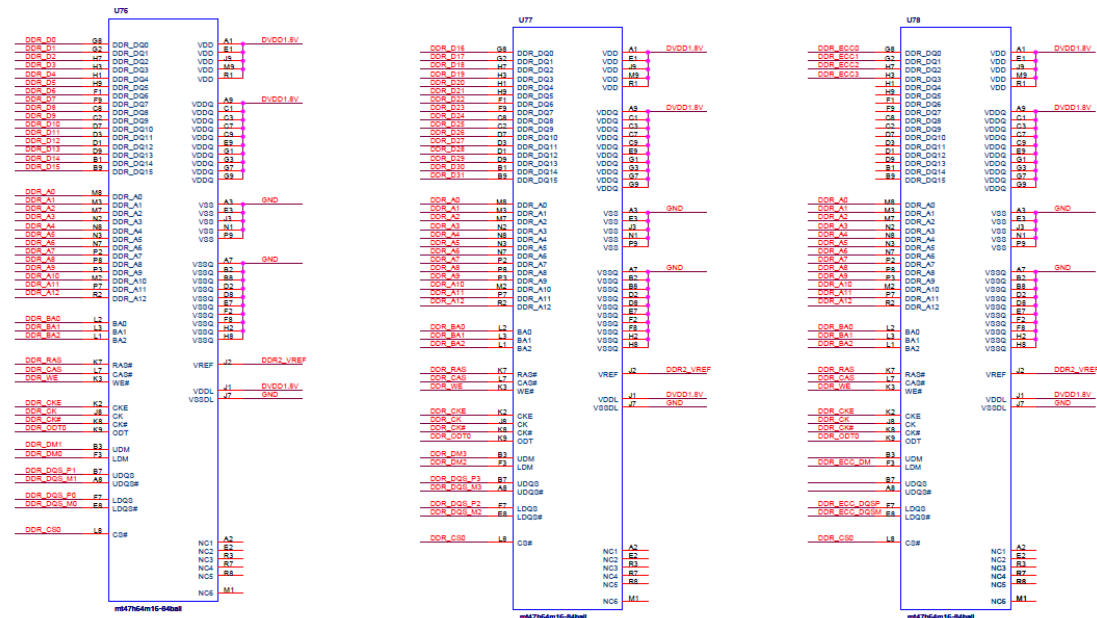


图 6-2-6 DDR2 系统参考原理图

6.2.9 Ethernet MAC

BM3823A 的 Ethernet MAC 支持 10/100/1000Mbps 三种数据传输速率，符合 IEEE802.3-2008 协议标准，通过以太网实现主机对数据的收发；PHY 接口支持 MII、GMII、RMII、TBI 传输模式；发送 FIFO 大小为 16KB，接收 FIFO 大小为 32KB。其中 MII 和 RMII 都支持 10/100Mbps 的传输速率，GMII 和 TBI 支持 1000Mbps 传输速率。10Mbps 速率下时钟频率为 2.5MHz，100Mbps 速率下时钟频率为 25MHz，1000Mbps 速率下时钟频率为 125MHz。

开发板中 88E1111-128pin 是以太网 PHY 芯片，满足 802.3 以太网传输协议标准，与 BM3823A 的时钟、接收数据、发送数据以及有关控制信号连接。hr911130a_rj45 是网络接口器件，与 PHY 芯片输出的差分对数据连接，通过网线进行数据的传输。由于内部 pll 的设计以及以太网正常工作的时钟频率要求，采用 25MHz 的输入晶振。

七、应用注意事项

7.1 产品应用说明

1. 在产品使用过程中如有任何不明确之处，请及时反馈。
2. 当系统复位时，需先复位外围器件，然后复位本电路。
3. 电源上电顺序严格按照章节 7.2，禁止出现内核电源先于 I/O 电源上电的情况。
4. 利用外部复位引脚 `resetn` 进行复位时，需要在时钟有效后保持复位信号至少 10 个时钟周期。
5. 为保证 pll 稳定工作，外部时钟 `clk` 的选择必须同时满足以下三个条件：输入频率范围 5MHz-100MHz；占空比范围 45%-55%；倍频后的时钟范围 25MHz-100MHz。
6. 测试管脚 `bist_mode`、`scan_mode`、`scan_enable`、`iddq_enable` 必须通过 1K Ω 电阻下拉。
7. 运行模式时，必须将 `dsuen` 和 `dsubre` 管脚置为低电平，`dsurx` 置为高电平。
8. PROM、SRAM、I/O 读访问时序中“`lead_out`”周期中的地址值，不保持上一个时钟周期中的地址值，而是上一个时钟周期中处理器内部总线地址值。
9. 当 SDRAM 和 SRAM 都配置使能，并且 SDRAM 的 bank 的 size 配置为 256MB 时，应当用处理器的管脚 `sdcsn[0]` 连接 SDRAM 存储器的片选管脚。
10. PCI 模块操作不支持字节、半字操作，只允许字操作，操作地址必须字对齐。
11. 电路采用 CMOS 工艺生产，使用时需注意静电防护。工作台应有可靠接地，并配有防静电设备，操作人员要佩戴防静电护腕。对电路进行测量前需先将所使用的测量探针接地，插拔电路时应使用专用工具进行操作，避免用手直接接触电路，以免造成污染或破坏。
12. 波峰焊工艺注意事项
 - (1) 推荐采用合金成分为 37%Pb63%Sn 的焊料。
 - (2) 器件在拾取和安装过程中应避免因外力造成的引线变形、损伤或表面镀层脱落。
 - (3) 波峰焊前，应对器件进行预热，避免焊接过程中应力过大造成器件损坏。波峰焊时，器件实际达到的峰值温度建议不超过 220℃，锡炉温度在 230℃到 250℃之间。波峰焊温度曲线如下图 7-1-1 所示：

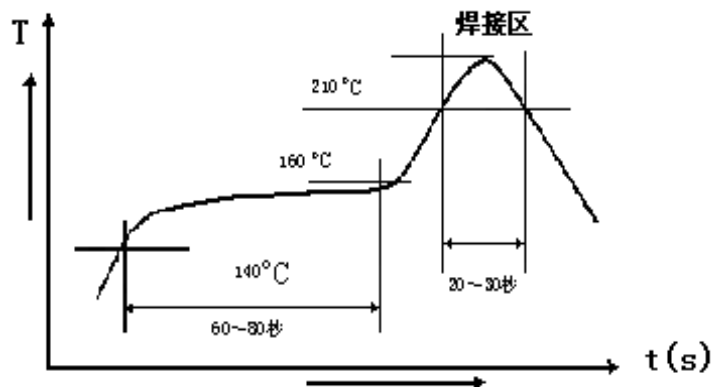


图 7-1-1 波峰焊温度曲线

13. 手工焊接注意事项

- (1) 器件在安装过程中应避免因外力造成引线发生变形损伤或表面镀层脱落。
- (2) 焊接时烙铁温度范围：240℃～300℃。
- (3) 单根引线焊接时间：5 秒～10 秒，烙铁头温度越高焊接时间要越短。

7.2 对电源的要求

BM3823A 共有 4 路电源，分别是数字 3.3V I/O 电源、模拟 3.3V I/O 电源、数字 1.8V I/O 电源、数字内核电源。BM3823A 对电源的要求和推荐电路如下。

➤ 工作电压要求

数字 3.3V I/O 和模拟 3.3V I/O 工作电压是 $3.30V \pm 0.33V$ ，数字 1.8V I/O 工作电压是 $1.80V \pm 0.18V$ ，数字内核工作电压是 $1.20V \pm 0.12V$ 。

➤ 上电过冲电压要求

上电时 3.3V 的数字 I/O 和模拟 I/O 电压过冲不得超过 4V，1.8V 的数字 I/O 电压过冲不得超过 2.5V，数字内核电压过冲不得超过 1.8V。

➤ 上电顺序要求

为保证 BM3823A 正常、安全启动，4 路电源必须严格按照正确顺序上电，具体要求是：

- 1) 数字 I/O 电源与模拟 I/O 电源同时上电（合称 I/O 电源）；
- 2) I/O 电源先于内核电源上电。

⚠ 注意

- 1、禁止上电过程中内核电压高于 I/O 电压超过 0.5V；

2、I/O 电源先于内核电源上电，但是 I/O 电源和内核电源之间的间隔要小于 2ms；

3、不满足上电顺序要求时可能会出现瞬时大电流，上电时可能对芯片造成损伤，但是只要成功上电，芯片可以正常工作，则对后续没有运行没有其他影响。

7.3 编程说明

7.3.1 BM3823A 汇编编程注意事项

BM3823A 是基于 SPARC V8 体系结构的 32 位 RISC 嵌入式处理器，指令兼容《The SPARC Architecture Manual Version 8》。SPARC V8 共定义了 246 条指令，分为五种基本指令类型，分别为数据传输指令、算术/逻辑操作指令、控制传输指令、混合指令和浮点操作指令。

BM3823A 根据具体设计特点，实现了 SPARC V8 定义的 179 条指令，其中数据传输指令 21 条、算术/逻辑操作指令 35 条、控制传输指令 83 条、混合指令 17 条、浮点操作指令 23 条。如表 7-3-1 所示。

BM3823A 未实现的指令有 67 条，包括协处理器指令 41 条、MMU 相关指令 1 条、四精度浮点指令 14 条、单精度浮点相乘得双精度浮点指令 1 条、浮点队列指令 1 条、非 32 位交换空间加载/存储指令 9 条。如表 7-3-2 所示。

表 7-3-1 BM3823A 实现指令

指令类型	实现指令
数据传输指令	LDSB、LDSH、LDUB、LDUH、LD、LDD、STB、STH、ST、STD、LDSTUB、SWAP、LDF、LDDF、LDFSR、STF、STDF、STFSR、LDA、STA、SWAPA
算术/逻辑操作指令	AND、ANDcc、ANDN、ANDNcc、OR、ORcc、ORN、ORNcc、XOR、XORcc、XNOR、XNORcc、SLL、SRL、SRA、ADD、ADDcc、ADDX、ADDXcc、TADDcc、TADDccTV、SUB、SUBcc、SUBX、SUBXcc、TSUBcc、TSUBccTV、UMUL、SMUL、UMULcc、SMULcc、UDIV、SDIV、UDIVcc、SDIVcc
控制传输指令	BA、BN、BNE、BE、BG、BLE、BGE、BL、BGU、BLEU、BCC、BCS、BPOS、BNEG、BVC、

指令类型	实现指令
	BVS、BA,a、BN,a、BNE,a、BE,a、BG,a、BLE,a、BGE,a、BL,a、BGU,a、BLEU,a、BCC,a、BCS,a、BPOS,a、BNEG,a、BVC,a、BVS,a、FBA、FBN、FBU、FBG、FBUG、FBL、FBUL、FBLG、FBNE、FBE、FBUE、FBGE、FBUGE、FBLE、FBULE、FBO、FBA,a、FBN,a、FBU,a、FBG,a、FBUG,a、FBL,a、FBUL,a、FBLG,a、FBNE,a、FBE,a、FBUE,a、FBGE,a、FBUGE,a、FBLE,a、FBULE,a、FBO,a、CALL、JMPL、RETT、TA、TN、TNE、TE、TG、TLE、TGE、TL、TGU、TLEU、TCC、TCS、TPOS、TNEG、TVC、TVS
混合指令	RDY、RDASR、RDPSR、RDWIM、RDTBR、WRY、WRASR、WRPSR、WRWIM、WRTBR、FLUSH、SETHI、NOP、UNIMP、MULScC、SAVE、RESTORE
浮点操作指令	FMOV _s 、FNEG _s 、FABS _s 、FSQRT _s 、FSQRT _d 、FADD _s 、FADD _d 、FSUB _s 、FSUB _d 、FMUL _s 、FMUL _d 、FDIV _s 、FDIV _d 、FiTO _s 、FdTO _s 、FiTO _d 、FsTO _d 、FsTO _i 、FdTO _i 、FCMP _s 、FCMP _d 、FCMP _{Es} 、FCMP _{Ed}

表 7-3-2 BM3823A 未实现指令

指令类型	未实现指令
协处理器指令	CBA、CBN、CB3、CB2、CB23、CB1、CB13、CB12、CB123、CB0、CB03、CB02、CB023、CB01、CB013、CB012、CBA,a、CBN,a、CB3,a、CB2,a、CB23,a、CB1,a、CB13,a、CB12,a、CB123,a、CB0,a、CB03,a、CB02,a、CB023,a、CB01,a、CB013,a、CB012,a、LDC、LDDC、LDCSR、STC、STDC、STCSR、STDCQ、CPop1、CPop2
MMU 相关指令	STBAR

指令类型	未实现指令
四精度浮点指令	FSQRTq、FADDq、FSUBq、FMULq、FDIVq、FdMULq、FqTOs、FqTOd、FiTOq、FsTOq、FdTOq、FqTOi、FCMPq、FCMPEq
单精度浮点相乘得双精度浮点指令	FsMULd
浮点队列指令	STDFQ
非 32 位交换空间存储/加载指令	LDSBA、LDSHA、LDUBA、LDUHA、LDDA、STBA、STHA、STDA、LDSTUBA

上述 BM3823A 实现的指令是《The SPARC Architecture Manual Version 8》手册中明确规定的实现相关指令，编程要遵守手册定义。

(1) 写状态寄存器指令是延迟指令，若用户软件在这些延迟指令后三指令周期内访问或隐含访问这些状态寄存器的指令，须（增加 3 条指令如 nop）满足该寄存器前一指令写入三指令后生效的要求，确保后续指令访问到的是已更新的状态寄存器。

一般的指令写法是：

```
mov reg_or_imm, %X(1)
```

```
nop
```

```
nop
```

```
nop
```

或者：

```
wr %g0, reg_or_imm, %X(1)
```

```
nop
```

```
nop
```

```
nop
```

注：

(1)%X 是%psr、%wim、%tbr 寄存器其中之一。

(2)半字的访问必须在 2-byte 边界上对齐（aligned）；字的访问（包括取指令），必须在 4-byte 边界上对齐；双字的访问必须在 8-byte 边界上对齐。一个没有正确对齐的地址会让 load 或 store 指令产生一个 mem_address_not_aligned 陷阱。

7.3.2 BM3823A 软件示例

电路应用示例是针对用户的普通应用，根据电路特征所给出的应用代码片段，描述了主要功能模块的使用方法和操作手段，作为电路用户手册的补充，可作为基本操作的参考。

BM3823A 为核心构建的单板计算机系统，软件开发应根据 SPARC V8 手册、BM3823A 电路用户手册要求进行；在实际应用中，可根据需要修改使用本示例中的代码片段。

7.3.2.1 片上寄存器访问

片上寄存器是片上模块（包括存储器控制、中断控制、I/O、定时器、串口等）的控制寄存器的集合；其在片上地址空间中的编址部分从 0x81000000 开始，部分从 0x80000000 开始。

片上寄存器的访问，类似于对统一编址的存储器的访问，如向 Cache 控制寄存器地址 0x80000014 写十六进制数据 0x81000f 打开 Cache 功能模块(缓存、指令 burst、数据 snoop)，C 语言程序的访问方式有多种：

a)、直接使用地址

```
*(unsigned int *)0x80000014 = 0x81000f;
```

b)、使用宏引用来访问片上寄存器

```
#define CACHE_CFG 0x80000014
```

```
*(unsigned int *)CACHE_CFG = 0x81000f;
```

或

```
#define CACHE_CFG *(volatile unsigned int *)0x80000014
```

```
CACHE_CFG = 0x81000f;
```

c)、使用结构体指针来访问

对片上寄存器（全部或部分）定义相应的数据结构（结构体），

```
struct lregs{  
    volatile unsigned int memcfg1;  
    volatile unsigned int memcfg2;  
    volatile unsigned int memcfg3;  
    volatile unsigned int ahbfailaddr;
```



```
volatile unsigned int ahbstatus;

volatile unsigned int cachectrl;    /*offset=0x14*/

.....    /*...其它...*/

};
```

定义指向结构体的指针 regs（指向 0x80000000 开始的寄存器区）

```
struct lregs *regs = (struct lregs*)0x80000000;
```

如 cache 打开可以用：

```
regs->cachectrl = 0x81000f;
```

将变量定义为 volatile 类型，是说这个变量可能会被意想不到的语句改变，这样，编译器就不会去假设这个变量的值。精确地说就是，优化器在用到这个变量时必须每次都小心地重新读取这个变量的值，而不是使用保存在寄存器里的备份，这样可以避免错误地引用寄存器的值，这在读取的时候更可靠一些。

7.3.2.2 存储器控制器

存储器控制寄存器，用来定义存储器控制器在控制各种不同类型存储器（包括 ROM、SRAM）时控制的参数（设备是否可用、总线宽度、读写等待周期、时序规定、存储器大小（或 bank 大小））等。

在实际配置时，ROM 控制参数在系统上电时硬件配置应完全满足对 rom 空间 BANK0 的访问，设置或修改对任一区域的控制器参数最好在访问此区域之前进行；且不对别的区域的访问造成不良影响：如修改 IO 时影响了 ROM。

如在使用外部 PROM 进行 boot 启动时，需要配置 SRAM 存储器控制器（bank 大小为 2M 字节，16 位、读写等待为 5 个周期，读改写使能），程序如下：

```
#define MEM_CFG2 *(volatile unsigned int *)0x81000004

MEM_CFG2 = 0x00029055; /*2MB、16 位、5 等待周期，读改写*/
```

7.3.2.3 Cache 控制

BM3823A 有 4 路指令 Cache（共 32KB），2 路数据 Cache（共 16KB），都采用了多路组相联（multi-set-Caches）技术。打开 Cache 缓存功能及指令 Cache 的突发取指功能可以提高程序性能。

指令 Cache 没有 snoop 功能，程序中修改指令内容应该增加 flush 指令以确存储器和

Cache 的一致性。

汇编指令 flush

C 语言语句: asm(“flush”);

在 flush 之后, 等待 256 个时钟周期, 可达到存储体和 Cache 的一致。使用插入 nop 指令的方法, 或者执行一段循环等待指令, 期间避免访问被修改的相关代码即可。

7.3.2.4 中断和陷阱

BM3823A 支持中断和陷阱两种异常, 陷阱是由于特定指令相关的硬件引起, 在引起异常产生的指令运行期间发生; BM3823A 的外部事件可以产生中断类型的异常, 中断的产生不与任何特定的指令相关, 并在指令执行期间发生。

当陷阱发生时, 与之唯一对应的识别码写入 TBR 的 8 位 *TT* 字段, 处理器自动保存 PC 到 %l1, nPC 到 %l2, 程序以 TBR 的值为目的地址进行跳转到陷阱向量表, 然后跳转到陷阱处理程序。由于 TBR 的低四位为 0, 因此陷阱向量表的每一入口包含陷阱处理程序的前 4 条指令 (16 字节), 陷阱向量表共定义了 256 个陷阱, 占用 4KB 地址。陷阱向量表定义如下:

```
sethi %hi(_trap_table), %g1; /*陷阱向量表首地址赋于 TBR*/
mov    %g1, %tbr;
nop; nop; nop;
/*绑定陷阱向量表*/
#define TRAP(H)    mov %psr,%l0; sethi %hi(H), %l4; jmp %l4+%lo(H); nop;
#define TRAP_ENTRY(H)    mov %psr,%l0; sethi %hi(H), %l4; jmp %l4+%lo(H);
mov %tbr, %l3;
TRAP(_window_overflow);          ! 05 window_overflow
...
TRAP_ENTRY(_ISR_Handler)        ! tt=0x11 level 1
...
/*陷阱处理程序*/
_window_overflow:
...
```

```
jmp %l1 ! Re-execute save.
```

```
rett %l2
```

当发生浮点异常陷阱（ $tt=0x08$ ）时， nPC 保存在寄存器 $\%l2$ 中，对此寄存器中保存的数值减四可以获得引起浮点异常陷阱指令的 PC 值。同时，当发生 IEEE754 型浮点异常时，由于异常打断了正常的处理过程，浮点异常状态位 $CEEXEC$ 字段不更新；若希望获得引起浮点异常陷阱的具体原因，可关闭 $\%fsr$ 中 TEM 字段并重新执行引起陷阱的指令，从而更新 $CEEXEC$ 字段，得到具体原因。

BM3823A 可以处理两级中断； $IPEND$ 显示了当前等待处理的中断状态；写**中断强制和未决寄存器**可以人为触发中断，而**中断清除寄存器**可以人为清除中断 $IPEND$ 。

BM3823A 响应一级中断时相应中断标识会被自动清除；二级中断处理则不会。

另外，寄存器 $0x800000B8$ 读取时是二级中断状态，写入时是清除二级中断，意义不同，需要注意。

7.3.2.5 定时器

BM3823A 具有 2 路 32 位通用定时器。

通用定时器共用一个预分频器。预分频器每个时钟周期减一，当产生下溢时，由**预分频重载寄存器**自动重新装载，所以有效的分频值等于**预分频重载寄存器**值加 1。定时器计数值在每个 $TICK$ 信号到来时减一，当产生下溢时，产生相应的中断，然后由各自的重载寄存器自动重新装载，有效的 $TICK$ 等于重载寄存器值加 1。所以每次中断时间间隔（以定时器 2 为例，且两个初始值都等于重载值）为：

$$(\text{预分频重载值}+1) \times (\text{定时器重载值}+1) / \text{处理器频率} \quad (\text{单位秒})$$

如主频 300MHz，定时器时钟频率 100MHz，定时器 20ms 中断设置为：

```
#define CPU_FREQ 100000000 /* 处理器主频 300MHz */
#define TIMER_SCALE_CNT *(volatile unsigned int *)0x81000060
#define TIMER_SCALE_LOAD *(volatile unsigned int *)0x81000064
#define TIMER0_CNT *(volatile unsigned int *)0x81000040
#define TIMER0_LOAD *(volatile unsigned int *)0x81000044
#define TIMER0_CTRL *(volatile unsigned int *)0x81000048
```

```
TIMER_SCALE_LOAD = CPU_FREQ/1000000 -1; /*预分频重载,每 us 溢出*/
TIMER_SCALE_CNT   = CPU_FREQ/1000000 -1; /*预分频计数初值*/
TIMER0_LOAD       = 10000 -1; /*定时器重载, 每 10000us*/
TIMER0_CNT        = 10000 -1; /*定时器计数初值*/
TIMER0_CTRL       = 0x7; /*开始定时, 自动重载, 允许定时器发出中断*/
```

7.3.2.6 串口

BM3823A 提供了 2 个不带 FIFO 的串口（支持全双工发送和接收数据）。波特率由分频器对系统时钟分频产生，设置串口 1 的参数示例如下：

```
#define BAUD_RATE 115200

#define UART_SCALE  (((CPU_FREQ*10) / (8 * BAUD_RATE))-5)/10

#define UART1_SCALE_SET    *(volatile unsigned int *)0x8100007C

#define UART1_STATUS      *(volatile unsigned int *)0x81000074

#define UART1_CTRL        *(volatile unsigned int *)0x81000078

#define GPIO_DIR          *(volatile unsigned int *)0x81000408

UART1_SCALE_SET  = UART_SCALE ;//115200bps

UART1_STATUS  = 0; /*清除状态*/

UART1_CTRL      = 3; /*收发使能*/

GPIO_DIR      = 0x00008000; /*串口 1 发送、接收需要设置复用 GPIO 的方向，且不考虑其它 GPIO 使用*/
```

串口 3、4 是具有 FIFO 的串口，可以使用中断方式发送和接收字符，也可以使用查询方式。当接收 FIFO 中有数据时，其状态寄存器的“已接收字节数”位域表明 FIFO 中已收到的字节数，最大为 32。

```
#define UART3_STATUS    *(volatile unsigned int *)0x810001c8

if (UART3_STATUS & 0x3F){ /*说明接收 fifo 有数据*/

    //读取或返回状态

}
```

据此判断是否有数可读取，程序读取或 DSU 调试读取一次，已接收字节数将减一，直

至 0 说明 FIFO 中无数据。

7.3.2.7 GPIO

GPIO 作为普通 IO 使用时，其**方向控制寄存器**和**数据寄存器**要配合使用，先由**方向控制寄存器**设置方向，再由**数据寄存器**发送或读入数据，这些均与中断设置无关。

如 GPIO 中，高 16 位（31-16）设置为输出，低 16 位（15-0）设置为输入，如下：

```
#define GPIO_DIR      *(volatile unsigned int *)0x81000408
```

```
#define GPIO_DATA      *(volatile unsigned int *)0x81000400
```

```
GPIO_DIR = 0xffff0000;
```

此时，从 GPIO15-GPIO0 读入的数据：

```
unsigned int i = GPIO_DATA; /*使用时取其低 16 位，即低 16 位值等于当前给  
GPIO15-GPIO0 管脚的赋值，管脚接高电平则值为 1，接低电平则值为 0*/
```

向数据寄存器中写数，会从 GPIO31-GPIO16 输出数据，但不影响从 GPIO15-GPIO0 读入的数据。

```
GPIO_DATA = 0x5555AAAA; /*高 16 位输出，此时高 16 位 GPIO_DATA [31:16]的值为 0x5555，而低 16 位 GPIO_DATA [15:0]的值保持为读入的数据不变*/
```

7.3.2.8 PCI

PCI 进行读写操作时一般采用 DMA 操作，需要配置 PCI 起始地址、AHB 起始地址以及 DMA 读写控制。

DMA 写操作示例：

```
*(unsigned int *)0xc0000000 = 0xc0000000; // DMA 写传输 PCI 总线起始地址
```

```
*(unsigned int *)0xc0000004 = 0x20000000; // DMA 写传输 AHB 总线起始地址
```

```
*(unsigned int *)0xc0000008 = 0x10071; //写 DMA 的空间大小和控制字
```

发起将 AHB 地址 0x2000-0000 处 0x100 字节写往 PCI 地址 0xC000-0000 处的 DMA Memory Write 操作。

DMA 读操作示例：

```
*(unsigned int *)0xc0000020 = 0x20000000; // DMA 读传输 PCI 总线起始地址
```

```
*(unsigned int *)0xc0000024 = 0x20001000; // DMA 读传输 AHB 总线起始地址
```

```
*(unsigned int *)0xc0000028 = 0x100a1; //读 DMA 的空间大小和控制字
```

发起将槽位地址在 0x2000-0000 处的 PCI 设备配置空间中 0x100 字节的内容读入 AHB 地址 0x2000-1000 处的 DMA Memory Read 操作。

BM3823A PCI Host 设备通过配置 DMA 进行 PCI 配置时,应尽量使用单字 DMA 读写。配置 DMA 读写时 PCI 地址的确定由 PCI 总线底板的槽位及选择关系决定。

7.4 产品防护

7.4.1 电装及防护措施

器件应采取防静电措施进行操作。推荐下列操作措施:

- a) 器件应在防静电的工作台上操作;
- b) 试验设备和器具应接地;
- c) 不能直接用手触摸器件引线,应佩戴防静电指套和腕带;
- d) 器件应存放在防静电材料制成的容器中;
- e) 生产、测试、使用及流转过过程工作区域内应避免使用能引起静电的塑料、橡胶或丝织物;
- f) 相对湿度应尽可能保持在 30%~70%。

7.4.2 包装

器件包装应至少满足以下要求:

- a) 由无腐蚀的材料制成;
- b) 具有足够的强度,能够经得起搬运过程中的震动和冲击;
- c) 用防静电材料涂敷过或浸渍过,具备足够的防静电能力;
- d) 能够牢固的把所装器件支撑在一定的位罝;
- e) 能保持器件引线不发生变形;
- f) 没有锋利的棱角;
- g) 能安全容易的移动、检查和替换器件;
- h) 一般不使用聚氯乙烯、氯丁橡胶、乙烯树脂和聚硫化物等材料,也不允许使用有硫、盐、酸、碱等腐蚀成分的材料,使用具有低放气指数、低尘粒脱落的材料制造为宜。

7.4.3 运输和贮存

器件在运输和贮存过程中,至少应满足以下要求:



a) 运输：在避免雨、雪直接影响的条件下，装有产品的包装箱可以用任何运输工具运输。但不能和带有酸性、碱性和其它腐蚀性物体堆放在一起。

b) 贮存：包装好的产品应贮存在环境温度为 $16^{\circ}\text{C}\sim 28^{\circ}\text{C}$ ，相对湿度不大于 $30\%\sim 70\%$ ，周围没有酸、碱或其它腐蚀性气体且通风良好的库房里。

八、用户关注产品信息

8.1 产品标识

产品标识按下图 8-1-1 所示。第一行“BM3823AMCCRH”为器件型号；第二行“XXXX”为生产批次标识，“XXXX”为两位年度数字加两位周数数字组合成的四位数字，△△为 ESD 等级标志；第三行中的“YYY”为电路序号。

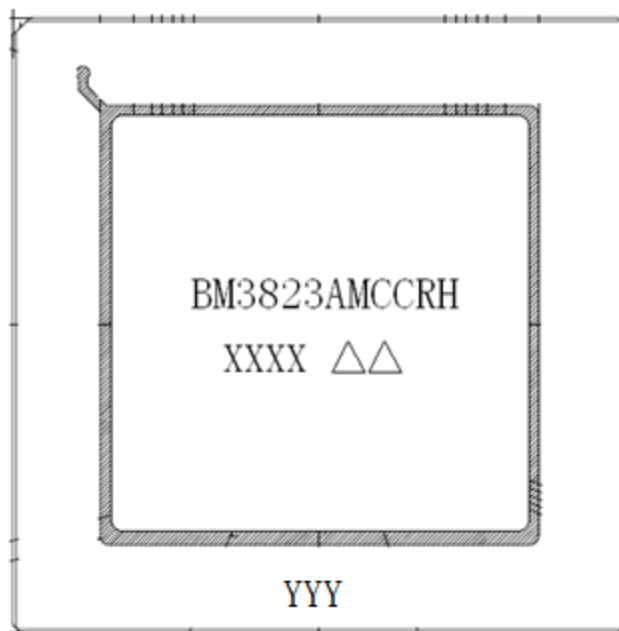


图 8-1-1 器件标识示意图

8.2 研制生产单位联系方式

通信地址：北京市丰台区东高地四营门北路 2 号

邮政编码：100076

联系部门：市场二部电话/传真：010-67968115-6313/010-68757706

SoC（MCU）事业部 刘亚丽 电话：010-67968115-8215

附录 1 警示页

- 1、在产品使用过程中如有任何不明确之处，请及时反馈。
- 2、若产品出现问题需要解焊，请提前经过产品技术人员确认再行解焊。
- 3、不使用 JTAG 功能时，trst 管脚必须通过 1K Ω 电阻下拉。
- 4、电源上电顺序：先 I/O 电压上电，后内核电压上电，禁止出现内核电源先于 I/O 电源上电的情况。
- 5、输入管脚不得悬空，当输入管脚不使用时，需根据管脚功能上拉或下拉，避免悬空。
- 6、输出管脚不使用时，禁止与其他管脚连接，包括电源和地。
- 7、程序编写时禁止访问无响应地址，例如 PCI 功能不使用时，无响应地址范围：
0xC0000000~0xFFFFFFFF。

附录 2 寄存器列表

2.1 0x80000000 地址上的寄存器

地址	寄存器描述	地址	寄存器描述
0x8000000C	高速总线 AHB 失效地址寄存器	0x800000B8	二级中断状态寄存器（只读）
0x80000010	高速总线 AHB 状态寄存器	0x800000B8	二级中断清除寄存器（只写）
0x80000014	Cache 控制寄存器	0x800000C4	DSU 串口状态寄存器
0x80000024	产品配置寄存器	0x800000C8	DSU 串口控制寄存器
0x80000030	内部存储器配置寄存器 1	0x800000CC	DSU 串口波特率重载寄存器
0x80000090	中断屏蔽和优先级寄存器	0x800000D0	PCI 总线复位寄存器
0x81000094	中断强制和未决寄存器	0x80000110	Cache 容错控制寄存器 1
0x80000098	六路中断子优先级寄存器	0x80000114	Cache 容错控制寄存器 2
0x8000009C	中断清除寄存器	0x80000118	Cache 容错控制寄存器 3
0x800000B0	二级中断屏蔽寄存器	0x800003C0	外部中断配置寄存器
0x800000B4	二级中断未决寄存器		

2.2 0x81000000 地址上的寄存器

地址	寄存器描述	地址	寄存器描述
0x81000000	外部存储器配置寄存器 1	0x81000104	外部存储器容错配置寄存器 2
0x81000004	外部存储器配置寄存器 2	0x810001C0	UART3 波特率分频寄存器
0x81000008	外部存储器配置寄存器 3	0x810001C4	UART3 发送/接收设置寄存器
0x8100000C	低速总线 AHB 失效地址寄存器	0x810001C8	UART3 状态寄存器
0x81000010	低速总线 AHB 状态寄存器	0x810001CC	UART3 数据寄存器
0x8100001C	写保护寄存器 1	0x810001D0	UART3 清状态寄存器
0x81000020	写保护寄存器 2	0x810001D4	UART3 发送结束寄存器
0x81000040	定时器 1 计数寄存器	0x810001D8	UART3 中断使能寄存器
0x81000044	定时器 1 重载寄存器	0x810001E0	UART4 波特率分频寄存器
0x81000048	定时器 1 控制寄存器	0x810001E4	UART4 发送/接收设置寄存器
0x8100004C	看门狗计数值寄存器	0x810001E8	UART4 状态寄存器
0x81000050	定时器 2 计数寄存器	0x810001EC	UART4 数据寄存器
0x81000054	定时器 2 重载寄存器	0x810001F0	UART4 清状态寄存器
0x81000058	定时器 2 控制寄存器	0x810001F4	UART4 发送结束寄存器
0x81000060	定时器预分频计数寄存器	0x810001F8	UART4 中断使能寄存器
0x81000064	定时器预分频重载寄存器	0x81000210	计数器 1 控制寄存器
0x81000070	UART1 数据寄存器	0x81000214	计数器 1 数据寄存器
0x81000074	UART1 状态寄存器	0x81000218	计数器 1 分频寄存器
0x81000078	UART1 控制寄存器	0x81000220	计数器 2 控制寄存器
0x8100007C	UART1 分频器重载寄存器	0x81000224	计数器 2 数据寄存器
0x81000080	UART2 数据寄存器	0x81000228	计数器 2 分频寄存器
0x81000084	UART2 状态寄存器	0x81000230	计数器 3 控制寄存器
0x81000088	UART2 控制寄存器	0x81000234	计数器 3 数据寄存器
0x8100008C	UART2 分频器重载寄存器	0x81000238	计数器 3 分频寄存器
0x81000100	外部存储器容错配置寄存器 1	0x81000240	计数器 4 控制寄存器

地址	寄存器描述	地址	寄存器描述
0x81000244	计数器 4 数据寄存器	0x81000350	PWM 定时器 5 计数寄存器
0x81000248	计数器 4 分频寄存器	0x81000354	PWM 定时器 5 预置值寄存器 1
0x81000250	计数器 5 控制寄存器	0x81000358	PWM 定时器 5 预置值寄存器 2
0x81000254	计数器 5 数据寄存器	0x8100035C	PWM 定时器 5 配置寄存器
0x81000258	计数器 5 分频寄存器	0x81000360	PWM 定时器 6 计数寄存器
0x81000260	计数器 6 控制寄存器	0x81000364	PWM 定时器 6 预置值寄存器 1
0x81000264	计数器 6 数据寄存器	0x81000368	PWM 定时器 6 预置值寄存器 2
0x81000268	计数器 6 分频寄存器	0x8100036C	PWM 定时器 6 配置寄存器
0x81000300	PWM 定时器控制寄存器	0x81000400	GPIO 输入输出数据寄存器
0x81000310	PWM 定时器 1 计数寄存器	0x81000408	GPIO 方向控制寄存器
0x81000314	PWM 定时器 1 预置值寄存器 1	0x81000410	GPIO 滤波分频与中断寄存器
0x81000318	PWM 定时器 1 预置值寄存器 2	0x81000414	GPIO 滤波时钟使能寄存器
0x8100031C	PWM 定时器 1 配置寄存器	0x81000418	GPIO 中断寄存器
0x81000320	PWM 定时器 2 计数寄存器	0x81000420	0 路数据寄存器 0
0x81000324	PWM 定时器 2 预置值寄存器 1	0x81000424	0 路数据寄存器 1
0x81000328	PWM 定时器 2 预置值寄存器 2	0x81000428	1 路数据寄存器 0
0x8100032C	PWM 定时器 2 配置寄存器	0x8100042C	1 路数据寄存器 1
0x81000330	PWM 定时器 3 计数寄存器	0x81000430	2 路数据寄存器 0
0x81000334	PWM 定时器 3 预置值寄存器 1	0x81000434	2 路数据寄存器 1
0x81000338	PWM 定时器 3 预置值寄存器 2	0x81000438	3 路数据寄存器 0
0x8100033C	PWM 定时器 3 配置寄存器	0x8100043C	3 路数据寄存器 1
0x81000340	PWM 定时器 4 计数寄存器	0x81000560	SPI0 控制寄存器
0x81000344	PWM 定时器 4 预置值寄存器 1	0x81000564	SPI0 状态寄存器
0x81000348	PWM 定时器 4 预置值寄存器 2	0x81000568	SPI0 数据寄存器
0x8100034C	PWM 定时器 4 配置寄存器	0x8100056C	SPI0_CLK 相对主钟分频寄存器
0x81000570	SPI0 数据位长寄存器	0x81000824	I ² C0 存储地址寄存器 1

地址	寄存器描述	地址	寄存器描述
0x81000574	SPI0 片选寄存器	0x81000828	I ² C0 存储地址寄存器 2
0x81000580	SPI1 控制寄存器	0x8100082C	I ² C0 存储模式选择寄存器
0x81000584	SPI1 状态寄存器	0x81000830	I ² C0 频率寄存器
0x81000588	SPI1 数据寄存器	0x81000900	I ² C1 中断标志寄存器
0x8100058C	SPI1_CLK 相对主钟分频寄存器	0x81000904	I ² C1 控制寄存器
0x81000590	SPI1 数据位长寄存器	0x81000908	I ² C1 速率寄存器
0x81000594	SPI1 片选寄存器	0x8100090C	I ² C1 数据寄存器
0x81000800	I ² C0 中断标志寄存器	0x81000910	I ² C1 从地址寄存器 1
0x81000804	I ² C0 控制寄存器	0x81000914	I ² C1 从地址寄存器 0
0x81000808	I ² C0 速率寄存器	0x81000918	I ² C1 状态寄存器
0x8100080C	I ² C0 数据寄存器	0x8100091C	I ² C1 中断屏蔽寄存器
0x81000810	I ² C0 从地址寄存器 1	0x81000920	I ² C1 存储器件地址寄存器
0x81000814	I ² C0 从地址寄存器 0	0x81000924	I ² C1 存储地址寄存器 1
0x81000818	I ² C0 状态寄存器	0x81000928	I ² C1 存储地址寄存器 2
0x8100081C	I ² C0 中断屏蔽寄存器	0x8100092C	I ² C1 存储模式选择寄存器
0x81000820	I ² C0 存储器件地址寄存器	0x81000930	I ² C1 频率寄存器

附录 3 寄存器描述

寄存器图例

位号

域名

读/写

复位后的初始值

31	30	29	28	27	26	25	24	23	...	...	...	...	9	8	7	6	5	4	3	2	1	0	
域				保留										位									
r = 只读				w = 只写										r/w = 可读可写									
0	4			1	x = 未定义或复位后无影响																		

3.1 系统相关寄存器

1. 处理器状态寄存器（Processor State Register, PSR）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IMPL				VER				N	Z	V	C	保留						EC	EF	PIL				S	PS	ET	保留		CWP		
r				r				r/w				r						r/w	r/w	r/w				r/w			r		r/w		
0000				1000				xxxx				0	0	1	0	0	0	0	1	xxxx				1	1	0	00		xxx		

- ◆ IMPL[31:28]: 处理器实现的等级，只读，其值为 4'b0000。
- ◆ VER[27:24]: 版本号，其值为 4'b1000。
- ◆ ICC[23:20]: 整数单元的条件码，被以字母 CC 结尾的指令操作。N，表示 ALU 的逻辑运算单元的 32 位结果是否为负数，1 表示结果是负数；Z，表示运算结果是否为零，1 表示结果为零；V，表示运算结果是否超出 32 位数的范围，1 表示结果溢出；C，表示运算的结果是否引起进位，1 表示结果有进位。
- ◆ EC[13]: 表示是否使能协处理器，1 表示使能。
- ◆ EF[12]: 表示是否使能浮点处理器，1 表示使能。
- ◆ PIL[11:8]: 中断级别，处理器只能处理高于 PIL 级别的中断，非屏蔽中断（中断级别为 15 的中断）不受 PIL 字段控制。
- ◆ S[7]: 指定处理器处于用户态还是管理态，1 表示处于管理态。
- ◆ PS[6]: 保留最近一次中断发生时 S 位中的值。
- ◆ ET[5]: 当 ET=0 时禁止陷阱，当 ET=1 时表示使能陷阱。
- ◆ CWP[2:0]: 当前窗口指针，可以识别当前寄存器窗口。

2. 窗口无效屏蔽寄存器（Windows Invalid Mask Register, WIM）

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

BM3823A 具有 8 个寄存器窗口，因此 WIM[31]-WIM[8] 的值始终为零。WIM[0]-WIM[7] 随着程序的运行而变化，用 WIM[n] 表示，其中 $0 \leq n \leq 7$ 。WIM[n]=0 表示窗口 n 有效，WIM[n]=1 表示窗口 n 无效。

对于 IU 中的每一个寄存器窗口，都对应一个有效状态位，即 WIM[n] 对应 CWP=n 的寄存器窗口。通过指令 RDWIM 和 WRWIM 实现对 WIM 的读写操作。

3. 异常基址标志寄存器（Trap Base Register, TBR）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBA												TT								ZERO											
r/w												r								r											
0												0								0000											

- ◆ TBA[31:12]：陷阱的基地址，这个基址由软件设置，通过 RDTBR 和 WRTBR 指令实现对 TBA 字段的读写。
- ◆ TT[11:4]：陷阱类型字段，当陷阱发生时硬件对这个字段进行写入，保持值不变，直到下一个陷阱发生。
- ◆ ZERO[3:0]：全零。

4. 乘法/除法寄存器（Multiply/Divide Register, Y）

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

32 位的 Y 寄存器在乘法运算时，其内容为整型乘积的高 32 位；在除法运算时，存放整型被除数的高 32 位。Y 寄存器由 RDY 和 WRY 指令读写。

5. 程序计数器（PC）

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

PC 为 32 位寄存器，保存 IU 正在执行指令的地址。

6. 程序计数器（nPC）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Y																															
r/w																															
0x4																															

nPC 为 32 位寄存器，保存着将要被执行的下一条指令的地址（发生 trap 的情况除外）。

7. Regfile EDAC 辅助状态寄存器 ASR16

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

◆ CB[31:29]: EDAC 的控制位，其值含义如下：

“000/001/011/100/110/111”，“错误”计数；

“010”，“错误”计数清零；

“101”，打开造错功能，包括校验位造错和数据位造错。

◆ [28:23]: 保留位；

◆ TCB[22:16]: 校验位造错，可读写；

◆ CNT[15:0]: 记录发生一位错误的次数，只读。

8. Regfile EDAC 造错控制寄存器 ASR17

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

◆ DCB[31:0]: 数据位造错，可读写；

BM3823A 的寄存器堆 (Regfile) 采用“纠一检二”的 EDAC 机制，即纠正一位错，检查两位错。ASR16 和 ASR17 控制 Regfile 的 EDAC 功能，硬件复位时，硬件对 ASR16 造错控制位[31:29]、校验位[23:16]、计数位[15:0]清零，也对 ASR17 清零。当电路硬件或软件设置 ASR16[31:29]=101（造错功能使能）时，则电路进入造错功能模式，此时写入到 Regfile 寄存器的值，是写入值与造错辅助寄存器 ASR17 值异或的结果，可能造成写入到 Regfile 寄存器的值与预期值不符，再读取 Regfile 寄存器时，检测到 Regfile 寄存器的 32 位数据和 7 位校验位不匹配；如果检测到两位错，则进入 TT=0x20 陷阱（检测到不可纠正的寄存器堆错误），如果有多位错，则可能引发 TT=0x07 陷阱（访问存储器时地址未对齐）或者 TT=0x09 陷阱(数据访问错误)或者运算、取数据错误。

在对 Regfile 寄存器赋值前，必须首先设置 ASR16 不使能造错，否则，会造成 Regfile EDAC 错。

9. 观察点寄存器 (ASR24、ASR26、ASR28、ASR30)

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

◆ WADDR[31:2]: 表示观察点的地址范围；

◆ [1]: 保留位；

◆ IF[0]: 取值使能位，‘1’表示使能，‘0’表示禁能。

10. 屏蔽地址寄存器（ASR25、ASR27、ASR29、ASR31）

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

- ◆ WMASK[31:2]: 与观察点的比较位，‘1’表示比较使能，‘0’表示比较禁能；
- ◆ DL[1]: 加载数据命中使能位，‘1’表示使能加载，‘0’表示禁能加载；
- ◆ DS[0]: 存储数据命中使能位，‘1’表示使能，表示禁能。

11. 浮点状态寄存器（Floating Point State Register, FSR）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD	保留	TEM				NS	保留	VER				FTT				保留	FCC	AEXC				CEXC									
r/w	r	r/w				r	r	r				r				r	r/w	r/w				r/w									
xx	00	xxxxx				0	00	000				xxx				00	00	xxxxx				xxxxx									

- ◆ RD[31:30]: 舍入方向，根据 ANSI/IEEE 的 754-1985 标准选择 FPU 浮点算术操作时的舍入方向。

表 A3-1-1 FSR 的舍入方式选择（RD）字段

RD Round Toward:
0 Nearest (even, if tie)
1 0
2 $+\infty$
3 $-\infty$

- ◆ [29:28]: 保留。
- ◆ TEM[27:23]: 异常使能标志，使能由 FPOps 引起的异常。该值同 CEXC（当前异常域值）相与以决定是否对 IU 产生浮点异常。所有使能域的命名与 CEXC 中一致，“0”不使能异常，“1”使能异常。
- ◆ NS[22]: FPU 实现标志。
- ◆ [21:20]: 保留。
- ◆ VER[19:17]: FPU 设计实现版本，只读，其值为 3b'000。

- ◆ FTT[16:14]: 浮点异常类型位, 识别浮点异常类型, 在指令 STFSR 或者另一条 FPop 指令执行之前, FTT 域值表示当前浮点异常类型。
- ◆ [13:12]: 保留。
- ◆ FCC[11:10]: 包含浮点条件码内容。浮点比较指令 (FCMP, FCMPE) 用于更新 FCC。指令 STFSR 和 LDFSR 用于读写 FCC。指令 FBfcc 基于 FCC 的值进行分支控制。
- ◆ AEXC[9:5]: TEM 不使能 fp_exception 时, 保存 IEEE_754 浮点异常。一条浮点操作指令完成之后, 将 TEM 和 CEXC 两个字段进行逻辑与操作, 如果结果非零, 将会产生一个 fp_exception 异常陷阱; 否则, 新的 CEXC 字段与 AEXC 字段进行或操作后将结果保留到 AEXC 字段中。这样, 当中断被屏蔽的时候, 异常就会保存在 AEXC 中。
- ◆ CEXC[4:0]: 表示最近执行的 Fpop 指令引起的一个或多个 IEEE 浮点异常。异常不发生时, 其对应位被清零。

12. 产品配置寄存器 (Product Config Register, PCR)

Address = 0x8000-0024

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MMU	DSU	SDCTRL	WTPNB			IMAC	NWINDOWS					ICSZ			ILSZ		DCSZ			DLSZ		DIVINST	MULINST	WDOG	MEMSTAT	FPU		PCI		WPRT	
r	r	r	r			r	r					r			r		r			r	r	r	r	r		r		r			
1	1	1	1	0	0	0	0	0	1	1	1	0	1	1	1	1	0	1	1	1	0	1	1	1	1	0	1	1	1	0	1

- ◆ MMU[31]: “1”, 表明具有 MMU;
- ◆ DSU[30]: “1”, 表明具有 DSU;
- ◆ STRCTRL[29]: “1”, 表明存在 SDRAM 功能;
- ◆ WTPNB[28:26]: “100”, 表明具有 4 个 watchpoint;
- ◆ IMAC[25]: “0”, 表明不支持 MAC 指令;
- ◆ NWINDOWS[24:20]: “00111”, 表明具有 8 寄存器窗口数目;
- ◆ ICSZ[19:17]: “011”, 指令 Cache 的大小;
- ◆ ILSZ[16:15]: “11”, 指令 Cache 的 LINE 的大小;
- ◆ DCSZ[14:12]: “011”, 数据 Cache 的大小;

- ◆ DLSZ[11:10]: “10”，数据 Cache 的 LINE 的大小；
- ◆ DIVINST[9]: “1”，表明具有硬件除法器；
- ◆ MULINST[8]: “1”，表明具有硬件乘法器；
- ◆ WDOG[7]: “1”，表明具有看门狗；
- ◆ MEMSTAT[6]: “1”，表明具有总线状态寄存器；
- ◆ FPU[5:4]: “01”，表明具有浮点处理器；
- ◆ PCI[3:2]: “11”，表明具有 PCI；
- ◆ WPRT[1:0]: “01”，表明具有写保护功能。

13. 高速总线 AHB 失效地址寄存器

Address = 0x8000-000C

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

- ◆ FA[31:0]: 记录当访问发生错误时的总线地址。

14. 高速总线 AHB 状态寄存器

Address = 0x8000-0010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																						CE	NE	RW	HMAST				HSIZE		
r																						r/w	r/w	r/w	r/w				w		
0																						0	0	x	xxxx				xxx		

- ◆ [31:10]: 保留，读出为 0。
- ◆ CE[9]: 可纠正的错误发生，“1”为发生可纠正的错误，“0”为未发生可纠正的错误。
当有可纠正的错误发生后，需要软件清零。
- ◆ NE[8]: 错误状态，“1”为有错误，“0”为无错误。当有错误发生后，需要软件清零。
- ◆ RW[7]: 读写访问，“1”为读，“0”为写。
- ◆ HMAST[6:3]: 记录访问失效时的 HMASTER，即总线主设备的值。

- ◆ HSIZE[2:0]: HSIZE 传输大小,“000”表示字节,“001”表示半字,“010”表示字,“011”表示双字。

15. 低速总线 AHB 失效地址寄存器

Address = 0x8100-000C

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

- ◆ FA[31:0]: 记录当访问发生错误时的总线地址。

16. 低速总线 AHB 状态寄存器

Address = 0x8100-0010

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																						CE	NE	RW	HMAST				HSIZE		
r																						r/w	r/w	r/w	r/w				w		
0																						0	0	x	xxxx				xxx		

- ◆ [31:10]: 保留,读出为 0。
- ◆ CE[9]: 可纠正的错误发生,“1”为发生可纠正的错误,“0”为未发生可纠正的错误。
当有可纠正的错误发生后,需要软件清零。
- ◆ NE[8]: 错误状态,“1”为有错误,“0”为无错误。当有错误发生后,需要软件清零。
- ◆ RW[7]: 读写访问,“1”为读,“0”为写。
- ◆ HMAST[6:3]: 记录访问失效时的 HMASTER,即总线主设备的值。
- ◆ HSIZE[2:0]: HSIZE 传输大小,“000”表示字节,“001”表示半字,“010”表示字,“011”表示双字。

3.2 Cache 相关寄存器

1. Cache 控制寄存器

Address = 0x8000-0014

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DRP		IRP		ISTS		DSTS		DS	FD	FI	保留				IB	IP	DP	保留						DF	IF	DCS		ICS			
r		r		r		r		r/w	w	w	r				r/w	r	r	r						r/w	r/w		r/w		r/w		
11		11		11		01		0	0	0	0000				0	x	x	0						x	x		00		00		

- ◆ DRP[31:30]: 数据 Cache 的替换策略，读出为“11”；
- ◆ IRP[29:28]: 指令 Cache 的替换策略，读出为“11”；
- ◆ ISTS[27:26]: 指令 Cache 的相联数，读出为“11”；
- ◆ DSTS[25:24]: 数据 Cache 的相联数，读出为“01”；
- ◆ DS[23]: 数据 Cache 的 Snoop 使能功能位侦听使能控制位。使能时，若其他主设备对可 Cache 内存写操作，Cache 中相应数据会无效；
- ◆ FD[22]: Flush 数据 Cache。若置位将 flush 数据 Cache；
- ◆ FI[21]: Flush 指令 Cache。若置位将 flush 指令 Cache；
- ◆ [20:17]: 保留位，读出为 0；
- ◆ IB[16]: 指令 burst 取，若置位将允许突发取指令；
- ◆ IP[15]: 指令 Cache 刷新进行中（IP），该位为“1”表示正在进行指令 Cache 的刷新；
- ◆ DP[14]: 数据 Cache 刷新进行中（DP），该位为“1”表示正在进行数据 Cache 的刷新；
- ◆ [13:6]: 保留位，读出为 0；
- ◆ DF[5]: 数据 Cache 中断冻结（DF），如果置位，数据 Cache 将在发生中断时，自动冻结；
- ◆ IF[4]: 指令 Cache 中断冻结（IF），如果置位，指令 Cache 将在发生中断时，自动冻结；
- ◆ DCS[3:2]: 数据 Cache 的状态，表明当前数据 Cache 的状态。
“01”冻结、“11”使能，“X0”禁止。
- ◆ ICS[1:0]: 指令 Cache 的状态，表明当前指令 Cache 的状态。

“01”冻结、“11”使能，“X0”禁止。

2. Cache 容错控制寄存器 1

Address = 0x8000-0110

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VER			保留																CMEEN		CNTRST	CPSEL		保留				EPOS			
r			r																r/w		r/w	r/w	r				r/w				
001			0																000		0	00		0000				0000			

- ◆ VER[31:29]: Cache 版本号寄存器，只读，其值为 3'b001。
- ◆ [28:14]: 保留位，读出为 0。
- ◆ CMEEN[13:11]: Cache Make Error Enable, Cache 造错使能标志位。“011”表示造错开启，其它值表示不造错。
- ◆ CNTRST[10]: CNT Reset, Cache 校验计数器清零标志。值为 1 时，清除所有错误计数器。
- ◆ CPSEL[9:8]: Cache Parity Select, Cache 制造错误的目标区域选择。“00”表示指令 Cache 的 data 区；“01”表示指令 Cache 的 tag 区；“10”表示数据 Cache 的 data 区；“11”表示数据 Cache 的 tag 区。
- ◆ [7:4]: 保留位。
- ◆ EPOS[3:0]: Error Position, Cache 造错位高 4 位。当目标区域选择为 data 区，[3:0]对应数据的 4 个校验位；当目标区域选择为 tag 区，[3:0]对应 tag 的 4 个校验位。

3. Cache 容错控制寄存器 2

Address = 0x8000-0114

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

- ◆ EPOS[31:0]: Error Position, Cache 造错位低 32 位。

当 Cache 造错使能，且目标区域为 data 或 tag 区，则[31:0]对应 data[31:0]；如果目标区域为 tag 区，则按照下图确定：



LOCK (1位)	CTX[7:0]	DTAG (19位)	VALID (4位)	数据TAG
LOCK (1位)	CTX[3:0]	DTAG (19位)	VALID (8位)	指令TAG
CCR2[31: 0] 造错bit位置标志				

4. Cache 容错控制寄存器 3

Address = 0x8000-0118

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

- ◆ DTAGCNT[31:26]: DCache 的 tag 错误计数器;
- ◆ DDATACNT[25:16]: DCache 的 data 错误计数器;
- ◆ ITAGCNT[15:10]: ICache 的 tag 错误计数器;
- ◆ IDATACNT[9: 0]: ICache 的 data 错误计数器。

3.3 MMU 相关寄存器

1. MMU 控制寄存器

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IMPL				VER				SC												PSO		保留				NF		E			
r				r				r		r		r		r		r				r/w		r				r/w		r/w			
0000				0000				101		101		1		0		0				x		00000				0		0			

- ◆ IMPL[31:28]: MMU 的特定实现位, 只读。
- ◆ VER[27:24]: MMU 版本, 只读。
- ◆ SC[23:8]: 系统控制位, 只读。
- ◆ PSO[7]: 决定处理器看到的存储器模式是部分存储顺序 (PSO=1) 还是全存储顺序 (PSO=0)。

- ◆ [6:2]: 保留位，读出为 0。
- ◆ NF[1]: “无错误”位。当 NF=0 时，MMU 发现的任何错误都会修改 FSR 和 FAR 并且会向处理器提供一个错误；当 NF=1 时，访问 ASI 0x9 指令的错误处理与 NF=0 时一样，而访问其它的 ASI 时产生的错误会修改 FSR 和 FAR，但是不会向处理器提供一个错误。
- ◆ E[0]: MMU 使能位，E 值为 1 时使能 MMU。

2. Context 表指针寄存器

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTP																															保留
r/w																															r
x																															00

- ◆ CTP[31:2]: Context 表指针，指向物理存储器中的 Context 表。Context 表通过 Context 寄存器的内容进行索引。
- ◆ [1:0]: 保留，只读。

3. Context 寄存器

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

- ◆ [31:8]: 保留。
- ◆ CN[7:0]: Context number，进程号。

4. MMU 错误状态寄存器

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留														EBE				L		AT		FT		FAV		OW					
r														r				r		r		r		r		r					
0														0				0		000		000		0		0					

- ◆ [31:18]: 保留。
- ◆ EBE[17:10]: 默认为 0。

- ◆ L[9:8]: 错误发生的表项所在的页表级数。L 定义如下:

L	Level
0	Entry in Context Table
1	Entry in Level-1 Page Table
2	Entry in Level-2 Page Table
3	Entry in Level-3 Page Table

- ◆ AT[7:5]: 访问类型, 定义了导致错误发生的访问操作类型。AT 定义如下:

AT	Access Type
0	Load from User Data Space
1	Load from Supervisor Data Space
2	Load/Execute from User Instruction Space
3	Load/Execute from Supervisor Instruction Space
4	Store to User Data Space
5	Store to Supervisor Data Space
6	Store to User Instruction Space
7	Store to Supervisor Instruction Space

- ◆ FT[4:2]: 错误类型, 定义如下:

FT	Fault Type
0	None
1	Invalid address error
2	Protection error
3	Privilege violation error
4	Translation error
5	Access bus error

- ◆ FAV[1]: 如果错误地址寄存器有效的话, FAV 为“1”。对于指令错误, 不需要将错误地址寄存器设为有效。而对于数据或者地址变换错误必须将错误地址寄存器设为有效。
- ◆ OW[0]: 如果错误状态寄存器已经不止一次的被指令访问错误写入, 则 OW 为“1”, 说明先前的错误状态已丢失。数据访问错误会覆盖以前的指令访问错误, 并清除 OW 位。

5. MMU 错误地址寄存器

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

◆ [31:0]: 错误地址寄存器包含错误状态寄存器记录错误的虚拟地址。

3.4 中断相关寄存器

1. 中断屏蔽和优先级寄存器

Address = 0x8000-0090

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

位号	助记符	描述
31	LEVEL[15]	15 号非屏蔽中断的优先级等级固定为 1
30:17	LEVEL[14:1]	中断优先级, 1=中断是属于优先级 1, 0=属于优先级 0, 优先级 1 高于优先级 0
15	IMASK[15]	15 号中断为非屏蔽中断, 屏蔽位固定为 1
14:1	IMASK[14:1]	中断屏蔽, 0=屏蔽中断, 1=允许中断

2. 中断强制和未决寄存器

Address = 0x8000-0094

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

位号	助记符	描述
31:17	IFORCE[15:1]	强制中断，1=中断强制，0=无强制中断
15:1	IPEND[15:1]	中断未决状态，1=有未处理的中断，0=没有未处理的中断

3. 六路中断子优先级寄存器

Address = 0x8000-0098

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留 2								INT6				INT5				INT4				INT3				INT2				INT1			
r								r/w				r/w				r/w				r/w				r/w				r/w			
0								0				0				0				0				0				0			

位号	助记符	描述
23:20	INT6	INT6 的优先级域，最高位 23 保留，优先级可配置为 000 到 101，110 和 111 为无效子优先级，配置为无效子优先级时不响应中断
19:16	INT5	INT5 的优先级域，最高位 19 保留，优先级可配置为 000 到 101，110 和 111 为无效子优先级，配置为无效子优先级时不响应中断
15:12	INT4	INT4 的优先级域，最高位 15 保留，优先级可配置为 000 到 101，110 和 111 为无效子优先级，配置为无效子优先级时不响应中断
11:8	INT3	INT3 的优先级域，最高位 11 保留，优先级可配置为 000 到 101，110 和 111 为无效子优先级，配置为无效子优先级时不响应中断
7:4	INT2	INT2 的优先级域，最高位 7 保留，优先级可配置为 000 到 101，110 和 111 为无效子优先级，配置为无效子优先级时不响应中断
3:0	INT1	INT1 的优先级域，最高位 3 保留，优先级可配置为 000 到 101，110 和 111 为无效子优先级，配置为无效子优先级时不响应中断

4. 中断清除寄存器

Address = 0x8000-009C

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

位号	助记符	描述
15:1	ICLEAR[15:1]	中断清除域，只写，置 1 将清除中断未决寄存器中的相应位，置 0 无效，读出为全零

5. 二级中断屏蔽寄存器

Address = 0x8000-00B0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								IMASK2																							
r								r/w																							
0								0																							

位号	助记符	描述
25:0	IMASK[25:0]	中断屏蔽，0=屏蔽中断，1=允许中断

6. 二级中断未决寄存器

Address = 0x8000-00B4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								IPEND2																							
r								r/w																							
0								0																							

位号	助记符	描述
25:0	IPEND[25:0]	中断未决，1=有未处理中断，0=没有未处理中断，写该域时不会清除原有的中断

7. 二级中断状态寄存器

Address = 0x8000-00B8

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

位号	助记符	描述
5	IP	中断未决状态，1=有未处理的未屏蔽中断；0=没有未处理的未屏蔽中断
4:0	ILR[4:0]	当前等待处理的中断号，表示当前未被屏蔽的最高优先级的二级中断

8. 二级中断清除寄存器

Address = 0x8000-00B8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								ICLEAR2																							
r								r/w																							
0								0																							

位号	助记符	描述
25:0	ICLEAR2[25:0]	中断清除，置 1 将会清除中断未决寄存器中相应位，写 0 无效

9. 外部中断配置寄存器

Address = 0x8000-03C0

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

位号	助记符	描述
27: 26	F	nmi 滤波配置, 00 滤小于一个周期的脉冲, 01 滤小于两个周期的脉冲, 10 滤小于三个周期的脉冲, 11 滤小于四个周期的脉冲
25: 24	T	nmi 触发方式, 00 代表上升沿触发, 01 代表下降沿触发, 10 代表低电平触发, 11 代表高电平触发
23: 22	F	int6 滤波配置, 00 滤小于一个周期的脉冲, 01 滤小于两个周期的脉冲, 10 滤小于三个周期的脉冲, 11 滤小于四个周期的脉冲
21: 20	T	int6 触发方式, 00 代表上升沿触发, 01 代表下降沿触发, 10 代表低电平触发, 11 代表高电平触发
19: 18	F	int5 滤波配置, 00 滤小于一个周期的脉冲, 01 滤小于两个周期的脉冲, 10 滤小于三个周期的脉冲, 11 滤小于四个周期的脉冲
17: 16	T	int5 触发方式, 00 代表上升沿触发, 01 代表下降沿触发, 10 代表低电平触发, 11 代表高电平触发
15: 14	F	int4 滤波配置, 00 滤小于一个周期的脉冲, 01 滤小于两个周期的脉冲, 10 滤小于三个周期的脉冲, 11 滤小于四个周期的脉冲
13: 12	T	int4 触发方式, 00 代表上升沿触发, 01 代表下降沿触发, 10 代表低电平触发, 11 代表高电平触发
11: 10	F	int3 滤波配置, 00 滤小于一个周期的脉冲, 01 滤小于两个周期的脉冲, 10 滤小于三个周期的脉冲, 11 滤小于四个周期的脉冲
9: 8	T	int3 触发方式, 00 代表上升沿触发, 01 代表下降沿触发, 10 代表低电平触发, 11 代表高电平触发
7: 6	F	int2 滤波配置, 00 滤小于一个周期的脉冲, 01 滤小于两个周期的脉冲, 10 滤小于三个周期的脉冲, 11 滤小于四个周期的脉冲
5: 4	T	int2 触发方式, 00 代表上升沿触发, 01 代表下降沿触发, 10 代表低电平触发, 11 代表高电平触发
3: 2	F	int1 滤波配置, 00 滤小于一个周期的脉冲, 01 滤小于两个周期的脉冲, 10 滤小于三个周期的脉冲, 11 滤小于四个周期的脉冲
1: 0	T	int1 触发方式, 00 代表上升沿触发, 01 代表下降沿触发, 10 代表低电平触发, 11 代表高电平触发

3.5 DMA 相关寄存器

1. DmaCfgReg-DMA 配置寄存器 (0xB800-0398)

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

位号	助记符	描述
0	DMA_EN	DMAC 使能位，1=DMAC 使能，0=DMAC 禁能。如果在通道使用过程中全局通道使能位被禁能了，那么 DmaCfgReg.DMA_EN 位仍然会返回 1，表示仍有通道在使用，直到硬件终止了所有通道的操作，DmaCfgReg.DMA_EN 位才会返回 0。

2. ChEnReg-通道使能寄存器（0xB800-03a0）

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

如果软件需要配置一条新的通道，就读该寄存器来找出当前未使用的通道；然后按照配置或默认的优先级来使能一条未使用的通道。

位号	助记符	描述
11:8	CH_EN_WE	通道写使能。
3:0	CH_EN	通道使能/禁能。置 1 使能通道，清零禁能通道。DMA 传输中的最后一次 AMBA 传输到目的完成之后，ChEnReg.CH_EN 位由硬件自动清零，以禁能相应通道。所以软件可以根据该位判断该通道什么时候可以用于新的 DMA 传送。通道使能位 ChEnReg.CH_EN 只有在相应通道写使能位 ChEnReg.CH_EN_WE 在同一 AHB 写传输总线上被声明时，才会被写入。

当全局 DMAC 通道使能位 DmaCfgReg[0]为 0 时，该寄存器的所有位清零。如果全局通道使能位是 0，那么对 ChEnReg 寄存器的写操作会被忽略，而读操作总会读到 0。

◆ DMA 通道寄存器（n=0~3）

DMA 通道寄存器包含以下寄存器（其中 n=0~3）：

CFGn – 通道 n 配置寄存器

CTLn – 通道 n 控制寄存器

DARn – 通道 n 目的地址寄存器

DSTATn–通道 n 目的状态寄存器

DSTATARn –通道 n 目的状态地址寄存器

LLPn –通道 n 链表项寄存器

SARn –通道 n 源地址寄存器

SSTATn –通道 n 源状态寄存器

SSTATARn –通道 n 源状态地址寄存器

SARn, DARn, LLPn, CTLn 和 CFGn 通道寄存器应该优先设置以使能通道。但如果在数据传输开始时发生了 LLI 更新，就不需要优先配置 SARn 和 DARn 再使能通道了。在通道使能后写寄存器 SARn, DARn, LLPn, CTLn, SSTATn, DSTATn, SSTATARn 或 DSTATARn 寄存器是违规寄存器操作。

3. SARn-通道 n 源地址寄存器（0xB800-0000, 0xB800-0058, 0xB800-00b0, 0xB800-0108）

（n=0~3，4 个地址依次对应通道 0~3）

在 DMA 通道使能前，起始源地址由软件配置，或者由 LLI 更新产生。在 DMA 传输过程中，该寄存器会跟随当前 AHB 传输源地址刷新。

注意：必须根据 CTLn_1.SRC_TR_WIDTH 配置 SAR 地址对齐。

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

位号	助记符	描述
31:0	SAR	当前 DMA 传输的源地址。每次源传输后自动更新。块传输时,CTLn_1 寄存器的 SINC 字段决定地址在每次源传输过程中是增加、减少还是不变。

4. DARN-通道 n 目的地址寄存器(0xB800-0008, 0xB800-0060, 0xB800-00b8, 0xB800-0110)

(n=0~3, 4 个地址依次对应通道 0~3)

在 DMA 通道使能前,起始目的地址由软件配置,或者由 LLI 更新产生。在 DMA 传输过程中,该寄存器会跟随当前 AHB 传输的目的地址刷新。

注意:必须根据 CTLn_1.DST_TR_WIDTH 配置 DAR 地址对齐。

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

位号	助记符	描述
31:0	DAR	当前 DMA 传输的目的地址。每次目的传输后自动更新。块传输时,CTLn_1 寄存器的 DINC 字段决定地址在每次目的传输过程中是增加,减少还是不变。

◆ SAR/DAR 寄存器硬件重排列

特殊情况下,在多块 DMA 传输中,在一个块结束,下一个块开始时,目的地址会发生偏移。当这种情形出现时,在下一个块传输开始前,DMAC 重排列目的地址。

例如,块长度是 9,源传输宽度是 16(半字),目的传输宽度是 32(字)——目的配置为连续的块传输——那么源输出的长度是 9 个半字共 4.5 个字,目的执行四次字传输和一次半字传输来完成到目的的块传输。由于最后一次 AMBA 传输是半字,在目的块传输的最后,地址排列为 16 位传输。这就与 32 位的目的传输大小偏离了。然而对于目的连续的多块传输,DMAC 会重排列 DAR 地址到最接近的 32 位地址(如果地址设置为增加则提前下一个 32 位地址,如果地址设置为减少则推后下一个 32 位地址)。

在下列 DMA 传输设定下,DMAC 会自动重排列目的地址:

- 目的端是连续多块传输，且
- $DST\_TR\_WIDTH > SRC\_TR\_WIDTH$ ，且
- 在 SRC_TR_WIDTH 或 DST_TR_WIDTH 配置为字节或半字的传输中，
($BLOCK\_TS * SRC\_TR\_WIDTH$)/ DST_TR_WIDTH 结果为非整数。

5. LLPn-通道 n 链表项寄存器 (0xB800-0010, 0xB800-0068, 0xB800-00c0, 0xB800-0118) (n=0~3, 4 个地址依次对应通道 0~3)

注意：如果块链接使能，该寄存器必须在通道使能前配置，并指向存储器中的 LLI。

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

位号	助记符	描述
31:2	LOC	当块链接使能时，表示存储器中下一个 LLI 的起始地址。注意起始地址的两个 LSB 并未存储，因为地址假定是沿着 32 位边界排列。LLI 访问总是 32 位宽，并沿着 32 位边界排列，不可以被改变或配置成任何非 32 位形式。
0	LMS	Master 选择列表。指示存储下一链接列表的存储器件所在的 AHB 层/接口。0 = AHB master 1；1 = AHB master 2。

LLP 寄存器有两项功能：

- 因子 LLP.LOC 用于配置传输是否使用链接列表。如果 LLP.LOC 设置为 0x0，那么用链接列表的传输被禁能了。该寄存器必须比通道使能优先配置以设置传输类型。
- LLP.LOC!=0 包含了对下一块链接 LLI 的指针。LLPn 寄存器还可以指出块传输完成之后控制写回和源/目的状态信息出现的地址。

6. CTLn_1-通道 n 控制寄存器 1 (0xB800-0018, 0xB800-0070, 0xB800-00C8, 0xB800-0120) (n=0~3, 4 个地址依次对应通道 0~3)

该寄存器包含控制 DMA 传输的字段。

块链接使能时，CTLn_1 寄存器块描述符（链接列表项 linked list item - LLI）的一部分，

并且它可以在一个 DMA 传输的基本块和块之间变化，即每次链接时可以给该寄存器配置不同的值，所有值需先配入链接列表，传输过程中自动更新。

注意该寄存器需要在通道使能前配置。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LLP_SRC_EN		LLP_DST_EN		SMS		DMS		TT_FC		SRC_MSIZ		DST_MSIZ		SINC		DINC		SRC_TR_WIDTH		DST_TR_WIDTH		INT_EN									
r		r/w		r/w		N/A		r/w		N/A		r/w		r		r/w		r/w		r/w		r/w		r/w		r/w		r/w		r/w	
0		0		0		0		0		0		3		0		1		1		0		0		0		0		0		1	

位号	助记符	描述
28	LLP_SRC_EN	当 LLP_SRC_EN=1, 且 LLPn.LOC 非零时, 源端块链接使能。
27	LLP_DST_EN	当 LLP_DST_EN=1, 且 LLPn.LOC 非零时, 目的端块链接使能。
25	SMS	源 master 选择。指示源器件（外设或存储器）接入的 master 接口层。0 = AHB master 1; 1 = AHB master 2。
23	DMS	目的 master 选择。指示目的器件（外设或存储器）接入的 master 接口层。0 = AHB master 1; 1 = AHB master 2。
22:20	TT_FC	传输类型和流控, 译码关系见表 A3-5-3。支持以下传输类型：存储器到存储器；存储器到外设；外设到存储器；外设到外设。流控可以是 DMAC 或外设。对于使用链接列表操作的多块传输, TT_FC 必须在该多块传输过程中, 在所有块中保持不变, 复位初值为 3。
16:14	SRC_MSIZ	源 burst 操作长度。当软件握手接口传来源 burst 处理请求时, 需要从源端读出多少个宽度为 CTLn.SRC_TR_WIDTH 的数据。配置值与传输数据量的关系见表 A3-5-2。复位初值为 1。
13:11	DST_MSIZ	目的 burst 操作长度。当软件握手接口传来目的 burst 处理请求时, 需要向目的端写入多少个宽度为 CTLn.DST_TR_WIDTH 的数据。配置值与传输数据量的关系见表 A3-5-2。复位初值为 1。
10:9	SING	源地址增量。每次源传输时, 表明源地址是增加还是减少。如果器件从地址修正的源外设 FIFO 中预取数据, 那么设置该区域为“不变 (No change)”。00 = 增加; 01 = 减小; 1x

位号	助记符	描述
		= 不变。增加或减少是为了对齐到下一条 CTLn_1.SRC_TR_WIDTH 边界。
8:7	DING	目标地址增量。每次目标传输时，表明目标地址是增加还是减少。如果器件从地址修正的目标外设 FIFO 中预取数据，那么设置该区域为“不变 (No change)”。00 = 增加；01 = 减小；1x = 不变。增加或减少是为了对齐到下一条 CTLn_1.DST_TR_WIDTH 边界。
6:4	SRC_TR_WIDTH	源传输宽度，配置值与传输宽度的关系见表 A3-5-1。与 AHB 总线“hsize”对应。对于非存储器外设，一般是指外设（源）FIFO 的宽度。这个位宽不能大于 32（master1 或 master2 层的接口数据位宽）复位值是编码值。
3:1	DST_TR_WIDTH	目的传输宽度，配置值与传输宽度的关系见表 A3-5-1。与 AHB 总线“hsize”对应。对于非存储器外设，一般是指外设（源）FIFO 的宽度。这个位宽不能大于 32（master1 或 master2 层的接口数据位宽）复位值是编码值。
0	INT_EN	中断使能位。如果置位，所有的中断源都使能。该位是通道所有中断的全局屏蔽位，复位初值为 1。

表 A3-5-1 CTLn.SRC_TR_WIDTH 和 CTLn.DST_TR_WIDTH 译码

CTLn.SRC_TR_WIDTH/ CTLn.DST_TR_WIDTH	大小 (bit)
000	8
001	16
010	32
011	无意义
1xx	无意义

表 A3-5-2 CTLn.SRC_MSIZ 和 CTLn.DST_MSIZ 译码

CTLn.SRC_MSIZ/ CTLn.DST_MSIZ	需要传输的数据量（单位为 CTLn.SRC_TR_WIDTH 或 CTLn.DST_TR_WIDTH）
000	1

CTLn.SRC_MSIZE/ CTLn.DST_MSIZE	需要传输的数据量（单位为 CTLn.SRC_TR_WIDTH 或 CTLn.DST_TR_WIDTH）
001	4
010	8
011	16
100	32
101	64
110	128
111	256

表 A3-5-3 CTLn. TT_FC 译码

CTLn.TT_FC	传输类型	流控
000	存储器到存储器	DMAC
001	存储器到外设	DMAC
010	外设到存储器	DMAC
011	外设到外设	DMAC
100	外设到存储器	外设
101	外设到外设	源外设
110	存储器到外设	外设
111	外设到外设	目的外设

7. CTLn_2-通道 n 控制寄存器 2(0xB800-001C, 0xB800-0074, 0xB800-00CC, 0xB800-0124)
(n=0~3, 4 个地址依次对应通道 0~3)

该寄存器包含控制 DMA 传输的字段。

块链接使能时，CTLn_2 寄存器是块描述符（链接列表项 linked list item - LLI）的一部分，并且它可以在一个 DMA 传输的基本块和块之间变化，即每次链接时可以给该寄存器配置不同的值，所有值需先配入链接列表，传输过程中自动更新。

如果写回状态使能，控制寄存器 CTLn_2，会在块传输结束时被写入系统存储器中 LLI 的控制寄存器区域。

注意该寄存器需要在通道使能前配置。



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

位号	助记符	描述
12	DONE	该位置 1 时，如果写回状态使能，CTLn_2 会在块传输结束时被写入寄存器 LLI CTLn_2，同时 LLI CTLx_2.DONE 位会被置高。软件可以通过检测 LLI CTLn_2.DONE 位来判断块传输是否完成。在配置使用链接列表的传输时，LLI CTLn_2.DONE 位应该配置为 0。
11:0	BLOCK_TS	块传输大小。当 DMAC 是流控时，在通道使能前需写该字段来指定块大小。BLOCK_TS 中写的数字表示每个块传输的总数，复位初值为 2。

状态寄存器 SSTATn、DSTATn、SSTATARn 和 DSTATARn 通常用于外设发起的 DMA 传输，用于跟踪源外设或目的外设的控制或传输状态。

8. SSTATn-通道 n 源状态寄存器 (0xB800-0020, 0xB800-0078, 0xB800-00d0, 0xB800-0128)
(n=0~3, 4 个地址依次对应通道 0~3)

每个块传输结束时，硬件可以根据 SSTATARn 寄存器的目录指向的地址找回源状态信息。然后，在下一个块传输开始之前，这些状态信息存储在 SSTATn 寄存器，并写出到 LLI 的 SSTATn 寄存器。

注意：该寄存器是源状态信息在向 LLI 的 SSTATn 寄存器传递过程中的临时占位符。源状态信息可以通过软件从 LLI 的 SSTATn 寄存器中找回，而不是由 DMAC slave 接口读本寄存器得到。DMAC 返回 ERROR 响应表示一次违规寄存器访问，包括访问无效寄存器地址。

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



位号	助记符	描述
31:0	SSTAT	硬件根据 SSTATARn 寄存器的值指向的地址找回源状态信息。

9. DSTATn-通道 n 目的状态寄存器（0xB800-0028，0xB800-0080，0xB800-00d8，0xB800-0130）（n=0~3，4 个地址依次对应通道 0~3）

每个块传输结束时，硬件可以根据 DSTATARn 寄存器的目录指向的地址找回目的状态信息。然后，在下一个块传输开始之前，这些状态信息会存储在 DSTATn 寄存器，并写出到 LLI 的 DSTATn 寄存器。

注意：该寄存器是目的状态信息在向 LLI 的 DSTATn 寄存器传递过程中的临时占位符。目的状态信息可以通过软件从 LLI 的 DSTATn 寄存器中找回，而不是由 DMAC slave 接口读本寄存器得到。DMAC 返回 ERROR 响应表示一次违规寄存器访问，包括访问无效寄存器地址。

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

位号	助记符	描述
31:0	DSTAT	硬件根据 DSTATARn 寄存器的值指向的地址找回目的状态信息。

10. SSTATARn-通道 n 源状态地址寄存器（0xB800-0030，0xB800-0088，0xB800-00e0，0xB800-0138）（n=0~3，4 个地址依次对应通道 0~3）

每个块传输结束时，硬件可以根据 SSTATARn 寄存器的值指向的用户定义地址找回源状态信息。用户可以选择系统存储器中的任意位置（例如源外设的控制寄存器），提供一个 32 位的值来表示源传输状态。

注意：DMAC 返回 ERROR 响应表示一次违规寄存器访问，包括访问无效寄存器地址。

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

位号	助记符	描述
31:0	SSTATAR	由硬件可以预取源状态信息处给出，寄存在 SSTATn 寄存器并在下一次块传输开始前写入 LLI 的 SSTATn 寄存器。

11. DSTATARn-通道 n 目的状态地址寄存器 (0xB800-0038, 0xB800-0090, 0xB800-00e8, 0xB800-0140) (n=0~3, 4 个地址依次对应通道 0~3)

每个块传输结束时，硬件可以根据 DSTATARn 寄存器的值指向的用户定义地址找回目的状态信息。用户可以选择系统存储器中的任意位置（例如目的外设的控制寄存器），提供一个 32 位的值来表示目的传输状态。

注意：DMAC 返回 ERROR 响应表示一次违规寄存器访问，包括访问无效寄存器地址。

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

位号	助记符	描述
31:0	DSTATAR	由硬件可以预取目的状态信息处给出，寄存在 DSTATn 寄存器并在下一次块传输开始前写入 LLI 的 DSTATn 寄存器。

12. CFGn_1-通道 n 配置寄存器 1 (0xB800-0040, 0xB800-0098, 0xB800-00f0, 0xB800-0148) (n=0~3, 4 个地址依次对应通道 0~3)

该寄存器用于配置 DMA 传输。通道配置寄存器对多块传输的所有块保持一致，不可中途改变。

注意：该寄存器需要在通道使能前配置。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
REL_OAD_DS_T	REL_OAD_SR_C	0 0 0																		HS_SEL_SR_C	HS_SEL_DS_T	FIFO_EMPTY	CH_SUSP	CH_PRIOR				0 0 0						
r/w	r/w	N/A																		r/w	r/w	r/w	r/w	r/w				N/A						
0	0	0																		1	1	1	0	x(0 0 0 0)				0						

位号	助记符	描述
31	RELOAD_DST	目的自动加载。若该位置 1，在每次块传输或多块传输的结尾，DARn 寄存器自动重加载配置的初值，一个新的块传输会被初始化；若该位置 0，则不使用目的自动重加载模式。复位初值为 0。
30	RELOAD_SRC	源自动加载。若该位置 1，在每次块传输或多块传输的结尾，SARn 寄存器自动重加载配置的初值，一个新的块传输会被初始化；若该位置 0，则不使用源自动重加载模式。复位初值为 0。
11	HS_SEL_SRC	源软件握手接口使能。0 表示禁能，忽略源端的软件处理请求；1 表示使能，本通道源端的软件握手接口有效。当源端是存储器时，此位无效。复位初值为 1。
10	HS_SEL_DST	目的软件握手接口使能。0 表示禁能，忽略目的端的软件处理请求；1 表示使能，本通道目的端的软件握手接口有效。当目的端是存储器时，此位无效。复位初值为 1。
9	FIFO_EMPTY	表示 FIFO 通道中是否还有数据，只读。可以与 CFGn_1.CH_SUSP 结合使用来彻底禁能一条通道。0 = FIFO 通道为空；1 = FIFO 通道非空，复位初值为 1。
8	CH_SUSP	通道延缓。延迟所有源端传来的 DMA 数据传输，直到该位被清零。不能保证当前操作能完成。可以与 CFGn_1.FIFO_EMPTY 结合使用，在不丢失数据的情况下彻底禁能一条通道。0 = 无延迟；1 = 延迟源端的 DMA 传输。复位初值为 0。
7:5	CH_PRIOR	通道优先权。3 代表最高优先级，0 代表最低优先级，由于本模块只有 4 条有效通道，[7]位无效。初始化值等于通道编号，即 Chan0=0，Chan1=1，Chan2=2，Chan3=3。优先权配置相同时，默认优先级先后次序为 Chan0，Chan1，Chan2，Chan3。复位初值同通道编号。

13. CFGn_2-通道 n 配置寄存器 2(0xB800-0044, 0xB800-009C, 0xB800-00f4, 0xB800-014C)
(n=0~3, 4 个地址依次对应通道 0~3)

该寄存器用于配置 DMA 传输。通道配置寄存器对多块传输的所有块保持一致，不可中途改变。

注意：该寄存器需要在通道使能前配置。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																								SS_UPD_EN	DS_UPD_EN	PROTCTL		FIFO_MODE	FCMODE		
N/A																								r/w	r/w	r/w		r/w	r/w		
0																								0	0	1		0	0		

位号	助记符	描述
6	SS_UPD_EN	源状态刷新使能。当 SS_UPD_EN 为高时，源状态信息只能从 SSTATARn 寄存器指向的地址取出，存储在 SSTATn 寄存器，并写出到 LLI 中的 SSTATn。
5	DS_UPD_EN	目的状态刷新使能。当 DS_UPD_EN 为高时，目的状态信息只能从 DSTATARn 寄存器指向的地址取出，存储在 DSTATn 寄存器，并写出到 LLI 中的 DSTATn。
4:2	PROTCTL	保护控制位，用于驱动 AHB HPROT[3:1]总线。AMBA 规范指出，HPROT 的缺省值表明一次无 Cache、无缓冲、特殊数据的访问，复位值用于表示这样的访问。HPROT[0]固定为 1 因为所有传输都是数据访问，没有操作码预取。该寄存器位与 HPROT[3:1]master 接口信号一一对应，见表 A3-5-4，复位初值为 1。
1	FIFO_MODE	FIFO 模式选择。在一次 burst 传输请求响应之前，确定需要多大的 FIFO 空间或数据。0 = 空间/数据可用于传输宽度确定的 single 传输；1 = 目的传输的可用数据大于等于半个 FIFO 深度，或源传输的可用空间大于等于半个 FIFO 深度。例外情况出现在 burst 传输请求的末端，或块传输的末端。
0	FCMODE	流程控制模式。当目的外设是流控时，决定源操作请求什么时候被响应。 0=源操作请求一出现就被响应，允许数据预取；

位号	助记符	描述
		1=只有在目的操作请求出现的时候才会响应源操作请求。这种模式下，源端传输的数据量是受限的，以确保在块传输被目的终止之前能传输到目的端。不支持数据预取。

表 A3-5-4 PROTCTL 字段与 HPROT 的匹配

1'b1	HPROT[0]
CFGx_2.PROTCTL[1]	HPROT[1]
CFGx_2.PROTCTL[2]	HPROT[2]
CFGx_2.PROTCTL[3]	HPROT[3]

14. DMA 原始中断状态寄存器 RawTfr, RawBlock, RawSrcTran, RawDstTran, RawErr (0xB80002c0, 0xB80002c8, 0xB80002d0, 0xB80002d8, 0xB80002e0) (地址与寄存器按序一一对应)

中断事件在屏蔽之前存储在**原始中断状态寄存器: RawTfr, RawBlock, RawSrcTran, RawDstTran, RawErr** 中。每个**原始中断状态寄存器**给每个通道分配一位；例如，RawTfr[2]是通道 2 原始传输完成中断。

这些寄存器中的每一位可以通过向 **ClearTfr, ClearBlock, ClearSrcTran, ClearDstTran, ClearErr 寄存器**的相应位置写 1 清零。

注意：仅在软件测试时才允许对这些寄存器的写访问。正常操作时，不允许对这些寄存器的写操作。

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

位号	助记符	描述
3:0	RAW	原始中断状态。

15. DMA 中断状态寄存器 StatusTfr, StatusBlock, StatusSrcTran, StatusDstTran, StatusErr (0xB800-02e8, 0xB800-02f0, 0xB800-02f8, 0xB800-0300, 0xB800-0308) (地址与寄存器按序一一对应)

屏蔽之后所有通道的所有中断事件存储在这些**中断状态寄存器: StatusTfr, StatusBlock, StatusSrcTran, StatusDstTran, StatusErr** 中。每个**中断状态寄存器**给每个通道分配一位, 例如, StatusTfr[2]是通道 2 状态传输完成中断。这些寄存器的内容用于产生中断信号 (int 或 int_n 总线, 取决于中断极性)。

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

位号	助记符	描述
3:0	STATUS	中断状态。

16. DMA 中断屏蔽寄存器 MaskTfr, MaskBlock, MaskSrcTran, MaskDstTran, MaskErr (0xB800-0310, 0xB800-0318, 0xB800-0320, 0xB800-0328, 0xB800-0330) (地址与寄存器按序一一对应)

原始中断状态寄存器的内容会被**中断屏蔽寄存器: MaskTfr, MaskBlock, MaskSrcTran, MaskDstTran, MaskErr** 的对应位屏蔽。每个**中断屏蔽寄存器**给每个通道分配一位; 例如, MaskTfr[2]是通道 2 传输完成中断的屏蔽位。

当 DMA 通道的源外设是存储器时, 源操作完成中断 MaskSrcTran[i]必须被屏蔽掉, 以免 int_combined(_n)信号错误的触发中断。类似的, 当 DMA 通道的目的外设是存储器时, 目的处理完成中断 MaskDstTran[i]必须被屏蔽掉, 以免 int_combined(_n)信号错误的触发中断。

只有当 INT_MASK_WE 字段的屏蔽写使能位在同一 AHB 写传输中声明时, 通道 INT_MASK 位才可以写入。这使得软件在不履行读修正写操作时就可以设置屏蔽位。例如, 写十六进制 01x1 到 MaskTfr 寄存器会向 MaskTfr[0]位写入 1, 而 MaskTfr[7:1]位则不会被写入; 写十六进制 00xx 则 MaskTfr[7:0]位都不会被写入。

向这些寄存器的任意位写 1 可以解除对相应位的屏蔽，也就是允许 DMAC 设置状态寄存器和 int_*端口信号的相应位。

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

位号	助记符	描述
11:8	INT_MASK_WE	中断屏蔽写使能。0 = 写禁能；1 = 写使能，只写，读出为 0。
3:0	INT_MASK	中断屏蔽。0 = 屏蔽；1 = 非屏蔽。

17. DMA 中断清除寄存器 ClearTfr, ClearBlock, ClearSrcTran, ClearDstTran, ClearErr (0xB800-0338, 0xB800-0340, 0xB800-0348, 0xB800-0350, 0xB800-0358) (地址与寄存器按序一一对应)

向中断清除寄存器: ClearTfr, ClearBlock, ClearSrcTran, ClearDstTran, ClearErr 的任意位写 1 后, 原始中断状态寄存器和中断状态寄存器中的相应位会被清零。每个中断清除寄存器给每个通道分配一位, 例如, ClearTfr[2]是通道 2 传输完成中断的清除位。写零无影响。这些寄存器不可读。

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

位号	助记符	描述
3:0	CLEAR	中断清除。0 = 无影响；1 = 中断清除，只写。

18. StatusInt-联合状态寄存器 (0xB800-0360)

五个状态寄存器: StatusTfr, StatusBlock, StatusSrcTran, StatusDstTran, StatusErr



中的任意一个的内容按位或，会在联合状态寄存器（StatusInt）中给每种中断类型产生一个独立位。该寄存器只读。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																										ERR	DST T	SRC T	BLO CK	TFR	
r																										r	r	r	r	r	
0																										0	0	0	0	0	

位号	助记符	描述
4	ERR	StatusErr 寄存器内容的“或”；
3	DSTT	StatusDstTran 寄存器内容的“或”；
2	SRCT	StatusSrcTran 寄存器内容的“或”；
1	BLOCK	StatusBlock 寄存器内容的“或”；
0	TFR	StatusTfr 寄存器内容的“或”。

19. ReqSrcReg-源软件处理请求寄存器（0xB800-0368）

该寄存器给每条通道设置一位。当通道 n 源端的软件握手未使能时，忽略 ReqSrcReg[n]。

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

位号	助记符	描述
11:8	SRC_REQ_WE	源请求写使能。0 = 写禁能；1 = 写使能，只写，读出为 0。
3:0	SRC_REQ	源软件处理请求。0 = 无请求；1 = 请求传输。

只有在 ChEnReg 寄存器中的通道使能，且同一 AHB 写操作中 SRC_REQ_WE 使能时，相应通道 SRC_REQ 位才会被写入。例如，写十六进制 0101 会向 ReqSrcReg[0]位写入 1，而 ReqSrcReg[7:1]位则保持不变；写十六进制 00xx 则 ReqSrcReg[7:0]位都不会被写入。这使得在没有读改写操作时，软件也可以设置 ReqSrcReg 寄存器中的某一位。



此寄存器的功能与源外设是否作流控有关，具体见 4.5.6 节。

20. ReqDstReg-目的软件处理请求寄存器（0xB800-0370）

该寄存器给每条通道设置一位。当通道 n 源端的软件握手未使能时，忽略 ReqDstReg[n]。

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

位号	助记符	描述
11:8	DST_REQ_WE	目的请求写使能。0 = 写禁能；1 = 写使能，只写，读出为 0。
3:0	DST_REQ	目的软件处理请求。0 = 无请求；1 = 请求传输。

只有在 ChEnReg 寄存器中的通道使能，且同一 AHB 写操作中 DST_REQ_WE 使能时，相应通道 DST_REQ 位才会被写入。

此寄存器的功能与目的外设是否作流控有关，具体见 4.5.6 节。

21. SglReqSrcReg-源 single 处理请求寄存器（0xB800-0378）

该寄存器给每条通道设置一位。当通道 n 源端的软件握手未使能时，忽略 SglReqSrcReg[n]。

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

位号	助记符	描述
11:8	SRC_SGLREQ_WE	源 single 请求写使能。0 = 写禁能；1 = 写使能，只写，读出为 0。
3:0	SRC_SGLREQ	源 single 请求。0 = 无请求；1 = 请求传输。

只有在 ChEnReg 寄存器中的通道使能，且同一 AHB 写操作中 SRC_SGLREQ_WE 使能时，相应通道 SRC_SGLREQ 位才会被写入。

此寄存器的功能与源外设是否作流控有关，具体见 4.5.6 节。

22. SglReqDstReg-目的 single 处理请求寄存器（0xB800-0380）

该寄存器给每条通道设置一位。当通道 n 源端的软件握手未使能时，忽略 SglReqDstReg[n]。

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

位号	助记符	描述
11:8	DST_SGLREQ_WE	目的 single 请求写使能。0 = 写禁能；1 = 写使能，只写，读出为 0。
3:0	DST_SGLREQ	目的 single 请求。0 = 无请求；1 = 请求传输。

只有在 ChEnReg 寄存器中的通道使能，且同一 AHB 写操作中 DST_SGLREQ_WE 使能时，相应通道 DST_SGLREQ 位才会被写入。

此寄存器的功能与目的外设是否作流控有关，具体见 4.5.6 节。

23. LstSrcReg-末位源处理请求寄存器（0xB800-0388）

该寄存器给每条通道设置一位。当通道 n 源端的软件握手未使能或通道 n 的源不是流控时，忽略 LstSrcReg[n]。

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

位号	助记符	描述
11:8	LSTSRC_WE	末位源处理请求写使能。0 = 写禁能；1 = 写使能，只写，读出为 0。
3:0	LSTSRC	末位源处理请求。0 = 无请求；1 = 请求传输。

只有在 ChEnReg 寄存器中的通道使能，且同一 AHB 写操作中 LSTSRC_WE 使能时，相应通道 LSTSRC 位才会被写入。

24. LstDstReg-末位目的处理请求寄存器（0xB800-0390）

该寄存器给每条通道设置一位。当通道 n 源端的软件握手未使能或通道 n 的源不是流控时，忽略 LstDstReg[n]。

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

位号	助记符	描述
11:8	LSTDST_WE	末位目的处理请求写使能。0 = 写禁能；1 = 写使能，只写，读 出为 0。
3:0	LSTDST	末位目的处理请求。0 = 无请求；1 = 请求传输。

只有在 ChEnReg 寄存器中的通道使能，且同一 AHB 写操作中 LSTDST_WE 使能时，相应通道 LSTDST 位才会被写入。

3.6 Ethernet MAC 相关寄存器

3.6.1 Ethernet DMA 相关寄存器

1. 总线模式寄存器（Bus Mode Reg）（0xB000-1000）

建议软件初始化赋值：0x00020101（必须根据板级实际情况赋值）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RIB	保留	PRWG		TXPR	MB	AAL	PBLx8	USP	RPBL						FB	PR	PBL						ATD S	DSL					DA	SWR	
r/w	ro	r/w		r/w	r/w	r/w			r/w						r/w	r/w	r/w						r/w	r/w					r/w	r/s	
0	0	00		0	0	000			01H						0	00	01H						0	00H					0	1	

- ◆ RIB[31]: 重新建立 INCRx Burst。该位置高时，AHB 主得到一个 EBT(retry,split,losing grant) AHB 主接口重新建立 INCRx burst。默认情况下重建 INCR burst。
- ◆ [30]: 保留位，只读。

- ◆ PRWG[29:28]: channel0 通道优先权值。00: 权值为 1; 01: 权值为 2; 10: 权值为 3; 11: 权值为 4。
- ◆ TXPR[27]: 发送优先级。置 1 表示, 在系统总线仲裁时, 发送 DMA 的优先级高于接收 DMA。
- ◆ MB[26]: Mixed burst。该位为 1 且 FB 为 0 时, AHB 主接口开始所有大于 16 的 burst 长度的 INCR, 反之, AHB 主接口开始小于等于 16 的 INCRx 和 SINGLE。
- ◆ AAL[25]: 地址对齐 beats, 当该位为高且 FB 为高, AHB 接口生产的所有 bursts 从 LSbits 开始对齐。
- ◆ PBLx8[24]: 当该位为高, 该位乘以 PBL 值 (bit[22: 17]和 bit[13:8]) 的 8 倍。因此 DMA 传输数据为 8, 16, 32, 64, 128, 256 拍是根据 PBL 的值。
- ◆ USP[23]: 用不同的 PBL 值, 该位为高时, 配置 RxDMA 用 bits[22:17]为 PBL 的值; 当为低时, bits[13:8]为 PBL 的值, 用于所有的 DMA。
- ◆ RPBL[22:17]: Rx DMA PBL, 表示一个 RxDMA 发送中可发的最大拍。可设置为 1, 2, 4, 8, 16 和 32。只有当 USP 为高时, 该区域有效。
- ◆ FB[16]: FB Fixed burst, 该信号控制 AHB 主接口实现固定 burst 传输。置 1, AHB 接口只用 SINGLE, INCR4, INCR8, INCR16。置 0, AHB 接口只用 SINGLE 和 INCR。
- ◆ PR[15:14]: 优先比率, 控制 RxDMA 和 TxDMA 仲裁权值的优先比率。只有当 bit1 (DA) 复位时, 该位有效。Rx:Tx 还是 Tx:Rx 根据 bit27 (TXPR) 复位与否确定。

00: 优先比率 1: 1	01: 优先比率 2: 1
10: 优先比率 3: 1	11: 优先比率 4: 1
- ◆ PBL[13:8]: PBL 可编 BURSR 长度, 该位表示在一个 DMA 传输中最大的 burst 长度。可为 1, 2, 4, 8, 16 和 32。当 USP 为高时, PBL 值只应用于 TxDMA 传输。当拍数大于 32 时, 将首先设置为 PBLx8 模式, 然后再设置 PBL。
- ◆ ATDS[7]: 改变描述符大小, Advanced timestamp feature 中, 将描述符提升到 32bytes (8dwords)。
- ◆ DSL[6:2]: 描述符跳跃长度, 该位表示两个不连接的描述之间的位, 如果 DSL 等于 0, 则两个描述符的相邻的闭环模式。
- ◆ DA[1]: DMA 仲裁表, 该位定义了 channel0 的发送和接收路径的仲裁表。

0: 权值为 Rx:Tx 或者 Tx: Rx, 优先级根据[15:14]的优先比率, 优先权值根据 bit27

1: 固定优先级 当 bit27 为高时, 发送路径的优先级高于接收路径的优先级, 亦或反之。

- ◆ SWR[0]: 软件复位, 该位为高所有 DMA 的控制器的寄存器和逻辑复位。在复位程序完成后, 该位自动清 0。

2. 发送查询命令寄存器 (Transmit Poll Demand Reg) (0xB000-1004)

建议软件初始化赋值: 0x0

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

- ◆ TPD[31:0]: 发送查询命令。该命令用来启动在 suspend 模式下的 Tx DMA。当该寄存器写入任何值, DMA 读 REG18(CURRENT HOST transmit descriptor reg)指向的描述符。如果该描述符不可用, 发送返回到 SUSPEND 状态, 且 REG5(Status reg)的 bit2 置 1。如果该描述符有效, 则发送重新开始。

3. 接收查询命令寄存器 (Receive Poll Demand Reg) (0xB000-1008)

建议软件初始化赋值: 0x0

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

- ◆ RPD[31:0]: 接收查询命令。该命令用来启动在 suspend 模式下的 Rx DMA。当该寄存器写入任何值, DMA 读 REG18(CURRENT HOST transmit descriptor reg)指向的描述符。如果该描述符不可用, 发送返回到 SUSPEND 状态, 且 REG5(Status reg)的 bit2 置 1。如果该描述符有效, 则发送重新开始。

4. 接收描述符列表地址寄存器 (Receive descriptor list address Reg) (0xB000-100C)

建议软件初始化赋值: 0x0

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

- ◆ RDESLA[31:0]: 开始接收表。该寄存器包括了接收描述符表的第一个描述符的基础地址。LSBbit 在 32bit, 64bit 或 128bit 时置 (1:0, 2:0, 3:0) 被忽略, 且被 DMA 置全 0, 因此这里 LSBbit 为只读。

5. 发送描述符列表地址寄存器 (Transmit descriptor list address Reg) (0xB000-1010)

建议软件初始化赋值: 0x0

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

- ◆ TDESLA[31:0]: 开始发送表。该寄存器包括了发送描述符表的第一个描述符的基础地址。LSBbit 在 32bit, 64bit 或 128bit 时置 (1:0, 2:0, 3:0) 被忽略, 且被 DMA 置全 0, 因此这里 LSBbit 为只读。

6. 状态寄存器 (Status Reg) (0xB000-1014)

建议软件初始化赋值: 0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
保留	GLPII	TTI	GPI	GMI	GLI	EB			TS			RS			NIS	AS	ERI	FBI	保留			ETI	RWT	RPS	RU	RI	UNF	OVE	TJT	TU	TPS	TI		
r0															r_ss_wc				r0			r_ss_wc												
0																																		

- ◆ [31]: 保留位, 只读。
- ◆ GLPII[30]: GMAC LPI 中断, 该位表示 MAC 中的 LPI 逻辑的一个中断。复位后, 软件必须读相应的寄存器得到准确的中断原因, 并将其清掉。只读。
- ◆ TTI[29]: timestamp trigger interrupt, 时间戳触发中断。该位表示 Timestamp Generator block 的一个中断, 软件必须读相应的寄存器得到准确的中断原因, 并将其复位清掉。该位只有在 IEEE 1588 Timestamp 使能后才有效。只读。

- ◆ GPI[28]: GMAC PMT interrupt, 软件必须读 PMT control and status reg 寄存器得到准确的中断原因, 并将其复位清掉。当该位为高时, sbd_intr_o 为高。只读。
- ◆ GMII[27]: GMAC MMC 中断。该位表示 MMC 模块的一个中断, 软件必须读相应的寄存器得到准确的中断原因, 并将其复位清掉。当该位为高时, sbd_intr_o 为高。只有当 MMC 有效时, 该位有效。只读。
- ◆ GLI[26]: GMAC Line interface interrupt。只读。某些配置下的中断。
- ◆ EB[25:23]: Error bits。该位表示引起 bus 错误的错误类型, 不产生中断, 只读。
 - 000: 在 Rx DMA 写数据时发生错误
 - 011: 在 Tx DMA 读数据时发生错误
 - 100: 在 Rx DMA 描述符写时发生错误
 - 101: 在 Tx DMA 描述符写时发生错误
 - 110: 在 Rx DMA 描述符读时发生错误
 - 111: 在 Tx DMA 描述符读时发生错误
- ◆ TS[22:20]: transmit process state, 该位表示发送 DMA FSM 状态, 不产生中断, 只读。
 - 000: 停止; 复位或停止发送命令
 - 001: 运行; 取发送传输描述符
 - 010: 运行; 等待状态
 - 011: 运行; 从主存储器读数据并且排队到 Tx FIFO
 - 100: TIME_STAMP 写状态
 - 101: 保留位
 - 110: 中断; 发送描述符不可用或发送缓冲下溢
 - 111: 运行; 结束发送描述符
- ◆ RS[19:17]: Receive Process state, 该位表示接收 DMA FSM 状态, 不产生中断, 只读。
 - 000: 停止; 复位或停止接收命令
 - 001: 运行; 取接收传输描述符
 - 010: 保留位

011: 运行; 等待接收包

100: 中断; 接收描述符不可用

101: 运行; 结束接收描述符

110: TIME_STAMP 写状态

111: 运行; 从接收缓冲传输接收数据包到主存储器

- ◆ NIS[16]: Normal Interrupt Summary, 该位的值来自 Reg5 寄存器 (REG7 相应位使能)。

Reg5[0]: 发送中断

Reg5[2]: 发送缓冲不可用

Reg5[6]: 接收中断

Reg5[14]: 过早接收中断

- ◆ AIS[15]: Abnormal Interrupt Summary, 不正常中断汇总。

Reg5[1]: 发送进程停止

Reg5[3]: 发送抖动超时

Reg5[4]: 接收 FIFO 下溢

Reg5[5]: 发送下溢

Reg5[7]: 接收缓冲不可用

Reg5[8]: 接收进程停止

Reg5[9]: 接收看门口超时

Reg5[10]: 过早发送中断

Reg5[13]: 重大总线错误

- ◆ ERI[14]: Early Receive Interrupt, 该位表示 DMA 填充数据包的第一个数据 buffer。

当软件为该位写 1 或者 bit6(RI)置位, 该位清零。

- ◆ FBI[13]: Fatal bus error interrupt, 该位表示一个 bus 错误发生, 如 bit[25:23]所述。

当该位被置位, 相关的 DMA 关闭所有的 bus 通道。

- ◆ [12:11]: 保留位, 只读。

- ◆ ETI[10]: Early transmit Interrupt, 该位表示需要发送的帧被过早发送到 MTL Tx FIFO。

- ◆ RWT[9]: Receive watchdog timeout, 当置位, 该为表示在接收当前帧时接收看门狗

计数器终止，以及当前主在看门狗超时后被截断。

- ◆ RPS[8]: Receive Process Stopped, 该位置位表示接收程序进入停止状态。
- ◆ RU[7]: Receive buffer unavailable, 该位表示主机在 Receive list 中拥有下一个描述符, 且 DMA 还未获得它。接收程序被暂停。要重启程序接收描述符, 主机需要改变描述符的所有权且发起一个 Receive poll Demand。如果没有发起 Receive poll Demand, 当接收到下一个可识别的帧时, Receive Process 重启。只有当前一个 Receive Descriptor 被 DMA 所有时, 该位置位。
- ◆ RI[6]: Receive interrupt, 该位表示帧接收完成。当接收完成时, RDES1 的 bit31 在最后一个 Descriptor 复位, 且新的帧状态信息在 descriptor 被更新。
- ◆ UNF[5]: Transmit Underflow, 该位表示发送 buffer 在帧发送时发生一个下溢。发送暂停且给出一个下溢错误 TDES0[1]置位。
- ◆ OVF[4]: Receive overflow, 该位表示在帧接收的时候, 接收 buffer 有一个溢出。如果部分帧被传输到了应用设备, 溢出状态 RDES0[11]设置为 1。
- ◆ TJT[3]: Transmit Jabber Timeout, 该位表示当帧大小超过 2048bytes 时, Transmit Jabber Timer 终止。当 Jabber Timeout 发生, 发送进程中断且进入停止状态, 引起 Transmit Jabber Timeout TDES0[14]标志置位。
- ◆ TU[2]: Transmit Buffer Unavailable, 该位表示主机在 Transmit list 中拥有下一个描述符, 且 DMA 还未获得它。发送程序被暂停。Bit[22:20]解释了 Transmit process state。要重启程序接收描述符, 主机需要通过将 TDE0[31]置位来改变描述符的所有权, 且发起一个 Transmit poll Demand。
- ◆ TPS [1]: Transmit Process Stopped, 当发送停止时, 该位置 1。
- ◆ TI [0]: Transmit interrupt, 该位表示帧发送完成。当发送完成, TDES0 的 bit31 复位, 描述符中的新的帧状态信息被更新。

7. 操作模式寄存器 (Operation Reg) (0xB000-1018)

建议软件初始化赋值: 0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
保留					DT	RSF	DFF	RFA ₂	RFD ₂	TSF	FTF	保留				TTC			ST	RFD	RFA		EEC	FEF	FUF	DGF	RTC		OSF	SR	保留	
r					r/w	r/w	r/w	r/w	r/w	r/w	r/w s/sc	r				r/w			r/w	r/w	r/w		r/w	r/w	r/w	r/w	r/w	r/w		r/w	r/w	r
00000					0	0	0	0	1	0	0	000				000			0	00	00		0	0	0	0	00		0	0	0	0

- ◆ [31:27]: 保留位，只读。
- ◆ DT[26]: Disable Dropping of TCP/IP Checksum Error Frames，当该位置位时，如果只有被 Receive Checksum offload 检测到错误，MAC 不会删除帧。这些帧在 MAC 接收的以太网帧上没有错误，却只有在封装载荷上有错误。当该位复位，所有错误帧将在 FEF 复位时被删除。
- ◆ RSF[25]: Receive Store and Forward，当该位置位，MTL 从 Rx FIFO 中读一帧数据只有在帧完成写入 Rx FIFO 中，忽略 RTCbits。当该位复位时，Rx FIFO 操作在 cut-through 模式，符合 RTCbits 定义的阈值。
- ◆ DFF[24]: Disable Flushing of Received Frames，当该位置位时，由于接收描述符或 buffer 不可用，RxDMA 不会 flush 任何数据帧。
- ◆ RFA₂[23]: MSB of Threshold for activating flow control，如果 GMAC 配置为大于等于 8KB，无论在半双工还是全双工模式，该位置位将为有效流量控制提供附加阈值。该位以及 RFA bits[10:9]共同提供以下阈值：

100: 5KB	101: 6KB	110: 7KB	111: 保留
- ◆ RFD₂[22]: MSB of Threshold for deactivating flow control，如果 GMAC 配置为大于等于 8KB，无论在半双工还是全双工模式，该位置位将为无效流量控制提供附加阈值。该位以及 RFD bits[12:11]共同提供以下阈值：

100: 5KB	101: 6KB	110: 7KB	111: 保留
- ◆ TSF[21]: Transmit store and forward，该位置位，存储于 Tx FIFO 中的一个完整数据帧开始发送，bit[16:14]定义的 TTC 值忽略。只有当发送停止，该位的值才会变化。
- ◆ FSF[20]: Flush Transmit FIFO，该位置位，Tx FIFO 控制器逻辑被复位为默认值，因此所有在 Tx FIFO 中的数据将会丢失。当 flushing mode 完成时，该位被内部清掉。只有当该位被清掉，REG6 才能被写。已被 MAC 发送器接收的数据将不会被刷掉。
- ◆ [19:17]: 保留位，只读。

- ◆ TTC[16:14]: Transmit Threshold Control, 该位控制 MTL Tx FIFO 的阈值。当帧大小比阈值大时发送开始。所有帧长度比比阈值小的帧被发送。只有到 bit21(TSF)复位时, 该位可用。

000: 64 001: 128 010: 192 011: 256

100: 40 101: 32 110: 24 111: 16

- ◆ ST[13]: Start or stop transmission commend, 该位置位, 发送程序进入 Running mode, DMA 在当前位置检测发送表, 描述符不仅从表中的当前位置 (transmit list bade address set by reg4) 采集, 而且从当前传输结束的保留位置采集。如果 DMA 没有获得当前描述符, 发送将进入延迟状态, 且 reg5 的 bit2 将被置位。只有当发送停止时, Start transmission commend 有效。当该位复位, 发送程序在完成当前帧传输后进入 stopped state。Transmit list 保存下一个描述符位置, 当发送重启后该位置变为当前位置。改变 list 的地址, 需要配置 reg4。只有在当前帧发送完成或发送进程进入 suspend state 时, stop transmission commend 有效。

- ◆ RFD[12:11]: Threshold for Deactivating Flow control (全双工、半双工)。该位控制 Rx FIFO 阈值 (流量控制无效激活)。

00: 1KB 01: 2KB 10: 3KB 11: 4KB

- ◆ RFA[10:9]: Threshold for activating Flow Control(全双工、半双工)。该位控制 Rx FIFO 阈值 (当流量控制有效激活)。

00: 1KB 01: 2KB 10: 3KB 11: 4KB

- ◆ EFC[8]: Enable HW flow control, 该位置位, 流量控制信号操作基于 fill-level RxFIFO 使能。当该位复位, 流量控制操作不使能。当 RxFIFO 小于 4KB 时, 该位不能用。

- ◆ FEF[7]: Forward Error Frames, 该位复位, RxFIFO 在错误状态下丢帧。但如果帧的开始字节指针 (write) 已经指向读控制器方向 (在 threshold mode 下), 那么该帧丢失。

- ◆ FUF[6]: Forward undersized Good Frames, 该位置位, Rx FIFO Forward undersized frames (帧无错误且帧长度小于 64bytes) 包括 pad-bytes 和 CRC。

- ◆ DGF[5]: Drop Giant Frames, 该位置位时, MAC 丢失 Rx FIFO 的接收超帧。

- ◆ RTC[4:3]: Receive Threshold Control, 这两位控制 MTL Rx FIFO 的阈值。当 Rx FIFO

的帧大小比阈值大，向 DMA 发送开始。帧长度比阈值短的帧自动发送。如果配置 Rx FIFO 为 128bytes，11 的值不可用。只有当 RSF 位为 0 时，这两位有效。00: 64; 01: 32; 10: 96; 11: 128。

- ◆ OSF[2]: Operate on Second Frame，该位置位，在获得第一个数据帧之前，DMA 发送第二个发送数据帧。
- ◆ SR[1]: Start or stop Receive，该位置位，receive process 进入 Running state。DMA 试图从 Receive list 获得 descriptor，且处理当前帧。Descriptor 采集 list 中的当前位置（地址见 REG3），或者采集 receive process 当前停止的位置。如果 DMA 没有拥有 descriptor，接收进入延迟状态且 REG5 的 bit7 置位。只有当接收停止时，start receive command 才有效。当该位复位，Rx DMA 在接收完当前帧后停止。下一个 Descriptor 位置被保存在 Receive list 中，且 Receive process 重启后变成当前位置。Stop Receive command 只有在 receive process 在 running 或 suspended 状态是才有效。
- ◆ [0]: 保留位，只读。

8. 中断使能寄存器（Interrupt Enable Reg）（0xB000-101C）

建议软件初始化赋值：0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																NIE	AIE	ER E	FB E	保留	ET E	RW E	RSE	RUE	RIE	UN E	OV E	TJE	TU E	TS E	TIE
r																r/w	r/w	r/w	r/w	r	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
0000000000000000																0	0	0	0	00	0	0	0	0	0	0	0	0	0	0	0

- ◆ [31:17]: 保留位，只读。
- ◆ NIE[16]: Normal Interrupt Summary Enable，该位置位，Normal Interrupt Summary 使能。该位复位，Normal Interrupt Summary 不使能。该位使能 reg5 的中断如下：
 - Reg5[0]: 发送中断
 - Reg5[2]: 发送缓冲不可用
 - Reg5[6]: 接收中断
 - Reg5[14]: 提前接收中断
- ◆ AIE[15]: Abnormal interrupt summary enable，该位置位，Abnormal interrupt summary 使能；该位复位，Abnormal interrupt summary 不使能。该位使能 reg5 的中断如下：

Reg5[1]: 发送进程停止

Reg5[3]: 发送抖动超时

Reg5[4]: 接收溢出

Reg5[5]: 发送下溢

Reg5[7]: 接收缓冲区不可用

Reg5[8]: 接收进程停止

Reg5[9]: 接收看门狗超时

Reg5[10]: 提前发送中断

Reg5[13]: 严重总线错误

- ◆ ERE[14]: Early receive interrupt enable, 该位置位, 且 bit16 NIE 置位, Early receive interrupt 使能; 该位复位, Early receive interrupt 不使能。
- ◆ FBE[13]: Fatal bus error enable, 该位置位, 且 bit15 AIE 置位, Fatal bus error 使能; 该位复位, Fatal bus error 不使能。
- ◆ [12:11]: 保留位, 只读。
- ◆ ETE[10]: Early transmit interrupt enable, 该位置位, 且 bit15 AIE 置位, Early transmit interrupt 使能; 该位复位, Early transmit interrupt 不使能。
- ◆ RWE[9]: Receive watchdog timeout enable, 该位置位, 且 bit15 AIE 置位, Receive watchdog timeout 使能; 该位复位, Receive watchdog timeout 不使能。
- ◆ RSE[8]: Receive stopped enable, 该位置位, 且 bit15 AIE 置位, Receive stopped 中断使能; 该位复位 Receive stopped 中断不使能。
- ◆ RUE[7]: Receive buffer unavailable enable, 该位置位, 且 bit15 AIE 置位, Receive buffer unavailable 中断使能; 该位复位 Receive buffer unavailable 中断不使能。
- ◆ RIE[6]: Receive interrupt enable, 该位置位, 且 bit16 NIE 置位, Receive interrupt 使能; 该位复位, Receive interrupt 不使能。
- ◆ UNE[5]: underflow interrupt enable, 该位置位, 且 bit15 AIE 置位, underflow interrupt 中断使能; 该位复位 underflow interrupt 中断不使能。
- ◆ OVE[4]: overflow interrupt enable, 该位置位, 且 bit15 AIE 置位, overflow interrupt 中断使能; 该位复位 overflow interrupt 中断不使能。

- ◆ TJE[3]: Transmit jabber timeout enable, 该位置位, 且 bit15 AIE 置位, Transmit jabber timeout 中断使能; 该位复位 Transmit jabber timeout 中断不使能。
- ◆ TUE[2]: Transmit buffer unavailable enable, 该位置位, 且 bit16 NIE 置位, Transmit buffer unavailable 中断使能; 该位复位, Transmit buffer unavailable 中断不使能。
- ◆ TSE[1]: Transmit stopped enable, 该位置位, 且 bit15 AIE 置位, Transmit stopped 中断使能; 该位复位 Transmit stopped 中断不使能。
- ◆ TIE[0]: Transmit interrupt enable, 该位置位, 且 bit16 NIE 置位, Transmit interrupt 使能; 该位复位, Transmit interrupt 不使能。

9. 丢失帧及缓存溢出计数寄存器 (0xB000-1020)

建议软件初始化赋值: 0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留			OVFCNTOVF	OVFFRMCNT												MISCNTOVF	MISFRMCNT														
r			r/ss/rc	r/ss/rc												r/ss/rc	r/ss/rc														
000			0	0												0	0														

- ◆ [31:29]: 保留位, 只读。
- ◆ OVFCNTOVF[28]: Overflow bit for FIFO overflow counter, 该位置位, bit[27:17] OVFFRMCNT 溢出, 因此, Rx FIFO 在 OVFFRMCNT 最大值时溢出。
- ◆ OVFFRMCNT[27:17]: Overflow frame counter, 该位表示应用时丢失的帧的数量。每当 MTL FIFO 溢出时该计数器增加。当 mci_be_i[2]=1 时, 该寄存器清零。
- ◆ MISCNTOVF[16]: Overflow bit for missed frame counter, 该位置位, bit[15:0] MISFRMCNT 溢出, 因此, 由于 Host receive buffer 在 MISFRMCNT 最大值时不可用, 所以 DMA 丢弃到来的帧。
- ◆ MISFRMCNT[15:0]: Missed frame conter, 由于 Host receive buffe 不可用, 该位表示控制器最大丢帧的数量。每当 DMA 丢弃到来的帧时, 该计数器增加。当 mci_be_i[0]=1 时, 该寄存器清零。

10. 接收中断看门狗计时器寄存器 (Receive Interrupt Watchdog Timer Reg) (0xB000-1024)

建议软件初始化赋值: 0x0

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

- ◆ [31:8]: 保留位，只读。
- ◆ RIWT [7:0]: RI Watchdog Timer Count，当看门狗定时器复位时，该位表示系统时钟周期乘以 256 的值。看门狗定时器在 RxDMA 完成一帧数据的传输后，由设定的值进行触发；这里 RI 状态位不置位，因为相关的设置在描述符 RDES1[31]上。

11. AHB 状态寄存器（AHB Status Reg）（0xB000-102C）

建议软件初始化赋值：0x0

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

- ◆ [31:2]: 保留位，只读。
- ◆ AXIRDSTS[1]: AXI Master Read Channel Status，该位为高表示 AXI 主的读通道激活且传输数据。
- ◆ AXWHSTS[0]: AXI Master Write Channel or AHB Master Status，该位为高，AXI 主的写通道激活且传输数据在 GMAC-AXI 配置情况下。在 GMAC-AHB 配置下，表示 AHB 主接口 FSMs 在非空闲状态。

12. 当前主发送描述符寄存器（Current host transmit descriptor Reg）（0xB000-1048）

建议软件初始化赋值：0x0

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

- ◆ CURTDESAPTR[31:0]: 该寄存器指向当前发送描述符 DMA 读地址的开始地址。指针在 DMA 操作时更新。只读。

13. 当前主接收描述符寄存器（Current host receive descriptor Reg）（0xB000-104C）

建议软件初始化赋值：0x0

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

- ◆ CURRDESAPTR[31:0]：该寄存器指向当前接收描述符 DMA 读地址的开始地址。指针在 DMA 操作时更新。只读。

14. 当前主发送缓存地址寄存器（Current host transmit buffer address Reg）（0xB000-1050）

建议软件初始化赋值：0x0

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

- ◆ CURTBUFAPTR[31:0]：该寄存器指向当前发送缓存 DMA 读地址的开始地址。指针在 DMA 操作时更新。只读。

15. 当前主接收缓存地址寄存器（Current host receive buffer address Reg）（0xB000-1054）

建议软件初始化赋值：0x0

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

- ◆ [31:0]：该寄存器指向当前接收缓存 DMA 读地址的开始地址。指针在 DMA 操作时更新。只读。

3.6.2 GMAC 相关寄存器

1. MAC 配置寄存器（MAC Configuration Reg）（0xB000-0000）

建议软件初始化赋值：0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留	SARC			TWOKPE	SFTERR	CST	TC	WD	JD	BE	JE	IFG			DCRS	PS	FES	DO	LM	DM	IPC	DR	LUD	ACS	BL		DC	TE	RE	PRELEN	
	r	r/w			r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w			r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w		r/w	r/w	r/w	r/w		
0	000			0	0	0	0	0	0	0	0	000			0	0	0	0	0	0	0	0	0	0	00		0	0	0	0	

- ◆ [31]: 保留位，只读。
- ◆ SARC[30:28]: Source Address Insertion or Replacement Control，该位控制源地址插入或者所有传输帧的重置。Bit30 位的值时根据 bits[29:28]。bits[29:28]=10 时，Bit30 置 0，表示 MAC 将内容插入 MAC address0 reg（REG16、17）。bits[29:28]=10 时，Bit30 置 1，表示使能在 core configuration 时的 MAC address reg1 选项，MAC 将内容插入 MAC address 1 reg（REG18、19）。
- ◆ TWOKPE[27]: IEEE 802.3as Support for 2k packets。该位置位，表示 MAC 帧正常包最大可到 2000bytes 长度。这里定义了不同情况下的超帧。
- ◆ SFTERR[26]: SMII Force Transmit Error，没有选择 SMII phy 时，该位为保留位，只读。
- ◆ CST[25]: CRC Stripping for Type Frame，该位置位，帧（长度大于等于 1536）的后 4bytes（FCS）在前一帧到达应用时被删除。该功能在 Type2 Checksum offload engine 使能时有效，在 IP Checksum engine（type1）不可用。
- ◆ TC[24]: Transmit configuration in RGMII，SGMII，SMII。该位置位，使能 RGMII，SGMII，SMII 信息。
- ◆ WD[23]: Watchdog Disable，该位置位，MAC 在接收端不使能看门狗定时器。MAC 可以接收最多 16383bytes 帧。该位复位，MAC 不允许一个接收帧大于 2048bytes（当 JE 为高时，该值为 10240bytes）或 reg55（Watchdog timeout reg）定义的值。在看门狗限制 bytes 大小后，MAC 会不接收超过阈值的所有 bytes。
- ◆ JD[22]: Jabber Disable，该位置位，MAC 在发送端不使能 Jabber timer。MAC 可以发送最多 16383bytes 帧。该位复位，MAC 不允许一个发送帧大于 2048bytes（当 JE 为高时，该值为 10240bytes）或 reg55（Watchdog timeout reg）定义的值，在 Jabber timer 限制 bytes 大小后，MAC 会不发送超过阈值的所有 bytes。
- ◆ BE[21]: Frame burst enable，该位置位，MAC 允许发送在 GMII 半双工模式下 frame

burst。在 10、100Mbps 或全双工模式下保留，只读。

- ◆ JE[20]: Jumbo Frame Enable, 该位置位, MAC 允许 9018bytes 的 Jumbo frame 在接收帧状态不报告巨帧错误。
- ◆ IFG[19:17]: inter-Frame Gap, 该位控制传输过程中两帧之间 IFG 最小值。
- ◆ DCRS[16]: Disable Carrier Sense during transmission, 该位为高, MAC 发送器在半双工模式下忽略(G)MII CRS 信号。该位为低, MAC 发送器生产错误。该位在全双工配置时保留, 只读。
- ◆ PS[15]: Port Select, 该位表示以太网通道速度。0: 1000M 以太网; 1: 10M/100M 以太网。
- ◆ FES[14]: Speed, 该位选择在 MII, RMII, RGMII, SGMII, RevMII 接口速度, 0: 10M; 1: 100M。
- ◆ DO[13]: Disable receive own, 该位置位, MAC 在 phy_txen_o 半双工有效时, 不使能帧接收。该位复位, MAC 接收所有 PHY 给出的数据包。该位只工作在半双工模式, 全双工模式时该位保留, 只读。
- ◆ LM[12]: Loopback mode, 该位置位, MAC 在 GMII/MII 模式下操作 Loopback mode。
- ◆ DM[11]: Duplex Mode, 该位置位, MAC 在全双工模式下可以同步的发送或接收数据。在全双工模式下该位只读, 默认值为 1。
- ◆ IPC[10]: Checksum offload, 该位只有在配置了 IP Checksum offload 可用, 否则位保留位, 只读。
- ◆ DR[9]: Disable Retry, 该位置位, MAC 只发送。当冲突发生在 GMII/MII 接口上, MAC 忽略当前帧传输且报告一个 Frame abort 在冲突错误时的帧传输状态。
- ◆ LUD[8]: Link up or down, 该位表示在 RGMII, SGMII, SMII 接口时链路连接或断开。0: 连接; 1: 断开。
- ◆ ACS[7]: Automatic pad or CRC stripping, 该位置位, 只有在到来帧长度小于 1536bytes 时, MAC 丢弃 pad 或 FCS。该位复位, MAC 通过所有到来的帧。
- ◆ BL[6:5]: Back-off limit, 该位置应用于半双工模式, 全双工模式时位保留位, 只读。
- ◆ DC[4]: Deferral check, 该位置应用于半双工模式, 全双工模式时位保留位, 只读。
- ◆ TE[3]: Transmit enable, 该位置位, MAC 的发送状态机在 GMII, MII 模式下发送

使能。该位复位，在完成当前帧发送后 MAC 的发送状态机不使能，也不再发送其他帧。

- ◆ RE[2]: Receive enable, 该位置位，MAC 的接收状态机在 GMII, MII 模式下接收使能。该位复位，在完成当前帧接收后 MAC 的接收状态机不使能，也不再接收其他帧。
- ◆ PRELEN[1:0]: Preamble length for transmit frames, 该位控制加在每个发送帧签名的字节数据，只有 MAC 在全双工模式下前置字节才会减少。

2. MAC 帧过滤寄存器 (MAC Frame Filter Reg) (0xB000-0004)

建议软件初始化赋值: 0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RA	保留									DNTU	IPFE	保留			VTFE	保留					HPF	SAF	SAIF	PCF		DBF	PM	DAIF	HMC	HUC	PR
r/w	r									r/w	r/w	r			r/w	r					r/w	r/w	r/w	r/w		r/w	r/w	r/w	r/w	r/w	r/w
0	000000000									0	0	000			0	00000					0	0	0	00		0	0	0	0	0	0

- ◆ RA[31]: receive all. 该位置位，表示 MAC 接收模块接收所有的帧。该位复位，表示接收模块只接收通过 SA/DA 滤波的帧。
- ◆ [30:22]: 保留位，只读。
- ◆ DNTU[21]: drop non-TCP/UDP over IP frames. 该位置位，使能 MAC 丢弃非 TCP/UDP 的 IP 报文。
- ◆ IPFE[20]: layer3 and layer4 filter enable. 该位置位，使能 MAC 丢弃不符合 layer3 和 layer4 过滤的帧。
- ◆ [19:17]: 保留位，只读。
- ◆ VTFE[16]: VLAN tag filter enable. 该位置位，使能 MAC 丢弃不符合 VLAN tag 的 VLAN tagged 帧。
- ◆ [15:11]: 保留位，只读。
- ◆ HPF[10]: hash or perfect filter. 该位置位，且 HMC 和 HUC 位置位，配置地址过滤器通过既符合哈希过滤又符合完美过滤的帧。该位复位且 HMC 和 HUC 位，只通过符合哈希过滤的帧。
- ◆ SAF[9]: source address filter enable. 该位置位，MAC 将接收帧与 SA 寄存器进行比

较，如果一致则接收该帧，如果不一致，则丢弃该帧。

- ◆ SAIF[8]: SA inverse filter。该位置位，MAC 将接收帧与 SA 寄存器进行比较，如果一致则丢弃该帧，如果不一致，则接收该帧。
- ◆ PCF[7:6]: pass control frames。该位控制所有的控制帧（包括 unicast，multicast 中断帧）
 - 00: MAC 过滤掉所有控制帧
 - 01: MAC 接收除中断帧外的所有控制帧
 - 10: MAC 接收所有控制帧
 - 11: MAC 只接收通过地址过滤的帧
- ◆ DBF[5]: disable broadcast frames。该位置位，AFM 模块组织所有广播帧。该位复位，AFM 模块通过所有广播站。
- ◆ PM[4]: pass all multicast。该位置位，通过所有具有组播目的地址的接收帧。该位复位，根据 HMC 位来过滤组播帧。
- ◆ DAIF[3]: DA inverse filter。该位置位，MAC 将接收帧与 DA 寄存器进行比较，如果一致则丢弃该帧，如果不一致，则接收该帧。
- ◆ HMC[2]: hash multicast。该位置位，MAC 通过哈希表对组播帧进行 DA 过滤。该位复位，MAC 对组播帧进行完美 DA 过滤。
- ◆ HUC[1]: hash unicast。该位置位，MAC 通过哈希表对单播帧进行 DA 过滤。该位复位，MAC 对单播帧进行完美 DA 过滤。
- ◆ PR[0]: promiscuous mode。该位置位，地址过滤模块通过所有帧，无论是 DA 还是 SA。当该位为 1 时，SA/DA 过滤失败状态将被清零。

3. GMII 地址寄存器（GMII address Reg）（0xB000-0010）

建议软件初始化赋值：0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																PA				GR				CR				GW	GB		
r																r/w				r/w				r/w				r/w	r/w s/s		
0000000000000000																00000				00000				0000				0	0		

- ◆ [31:16]: 保留位，只读。
- ◆ PA[15:11]: Physical Layer address，该位表示 32 位的 PHY 设备通道。
- ◆ GR[10:6]: GMII reg，该位在可选的 PHY 设备上选择 GMII 寄存器。
- ◆ CR[5:2]: CSR Clock Range，根据 CSR 时钟频率选择想要的 MDC 时钟频率：

0000: CSR clock=60-100MHz	MDC clock=CSR clock/42
0001: CSR clock=100-150MHz	MDC clock=CSR clock/62
0010: CSR clock=20-35MHz	MDC clock=CSR clock/16
0011: CSR clock=35-60MHz	MDC clock=CSR clock/26
0100: CSR clock=150-250MHz	MDC clock=CSR clock/102
0101: CSR clock=250-300MHz	MDC clock=CSR clock/124
0110, 0111: 保留位	

如果芯片支持更高的 MDC clock，那么 MDC clock 等于：

1000: CSR clock/4
1001: CSR clock/6
1010: CSR clock/8
1011: CSR clock/10
1100: CSR clock/12
1101: CSR clock/14
1110: CSR clock/16
1111: CSR clock/18

- ◆ GW[1]: GMII write，该位置位，表示 GMII 对 PHY 时写操作；该位复位，GMII 对 PHY 是读操作。
- ◆ GB[0]: GMII Busy，在写 MAC reg4 和 reg5 之前，该位应该读位 0；软件可设置该位为 1 表示进程中有个读或写该 PHY 通道。只有当 MAC 将该位清掉以后，reg5 (GMII data) 才可用。

4. GMII 数据寄存器 (GMII data Reg) (0xB000-0014)

建议软件初始化赋值：0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																GD															
r																r_w															
0000000000000000																0000000000000000															

- ◆ [31:16]: 保留位，只读。
- ◆ GD[15:0]: GMII Data, 包含在一个管理读操作之后从 PHY 读出的 16bit 数据值，以及在管理写操作之前写入 PHY 的 16bit 数据值。

5. 流量控制寄存器 (Flow control Reg) (0xB000-0018)

建议软件初始化赋值: 0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PT																保留				DZPQ	保留	LT		UP	RFE	TTE	FCB_BPA				
r/w																r				r/w	r	r/w		r/w	r/w	r/w	r/w	r/w			
0000000000000000																00000000				0	0	00		0	0	0	0	0			

- ◆ PT[31:16]: Pause time, 表示保持在 Pause time 用到的发送控制帧的值。如果 Pause time 被配置为双同步与 (G) MII 时钟域, 那么连续写该寄存器只发生在 4 个时钟周期之后。
- ◆ [15:8]: 保留位，只读。
- ◆ DZPQ[7]: Disable zero-quanta pause, 该位置位表示在无效的 FIFO 流量控制信号上不使能自动生成 zero-quanta pause。该位复位, 正常使能生成 zero-quanta pause。
- ◆ [6]: 保留位，只读。
- ◆ LT[5:4]: Pause low threshold, 在输入流量控制信号 mti_flowctrl_i (sbd_flowctrl_i) 检测到自动重发 Pause frame 时, 配置 Pause time 的阈值。该阈值应该小于 bit[31:16] 配置的 Pause time。

00: PT-4 slot times

01: PT-28 slot times

10: PT-144 slot times

11: PT-256 slot times

- ◆ UP[3]: Unicast pause frame detect, 在 IEEE STD 802.3 中用到多点传输时, 需要 pause frame (暂停帧)。该位置位, MAC 可在多点传输地址栈上检测暂停帧。该位复位, MAC 只能在唯一的多点传说地址上检测暂停帧。
- ◆ RFE[2]: Receive flow control enable, 该位置位, MAC 解码接收的暂停帧, 且在暂停时间不使能发送端; 该位复位, 解码暂停帧功能不可用。
- ◆ TFE[1]: Transmit flow control enable, 在全双工模式下, 该位置位, MAC 使能流量控制操作发送暂停帧; 该位复位, MAC 不使能流量控制操作, 也不发送暂停帧。
- ◆ FCB_BPA[0]: Flow control busy or backpressure activate, 在全双工模式下, 该位在写流量控制寄存器之前应该读出 0。初始化暂停帧, 应用应该设置为 1。

6. 总线模式寄存器 (Debug Reg) (0xB000-0024)

建议软件初始化赋值: 0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留						TXSTSTS	TXFSTS	保留	TWCSTS	TRCSTS	TXPAUSED	TFCSTS	保留						RXSTSTS	保留		RRCSTS	RWCSTS	保留	RFCSTS	RPESTS					
r						r	r	r	r	r	r	r	r						r	r		r	r	r	r	r					
000000						0	0	0	0	00	0	00	0	000000						00	0	00	0	0	00	0					

- ◆ [31:26]: 保留位, 只读。
- ◆ TXSTSFSTS[25]: MTL Txstatus FIFO full status, 该位为高表示 MTL TxStatus FIFO 为满。因此 MTL 不接受任何帧发送。该位在 GMAC-AHB 配置情况下为保留位。
- ◆ TXFSTS[24]: MTL Tx FIFO Not Empty status, 该位为高, 表示 MTL Tx FIFO 不空且在发送时一些数据丢失。
- ◆ [23]: 保留位, 只读。
- ◆ TWCSTS[22]: MTL Tx FIFO Write Controller Status, 该位为高, 表示 MTL Tx FIFO 写控制器激活, 且正在发送数据到 Tx FIFO。
- ◆ TRCSTS[21:20]: MTL Tx FIFO Read Controller Status, 表示 Tx FIFO 读控制器的状态。

00: 空闲状态;

01: 读状态;

10: 等待 MAC 发送器的 TxStatus;

11: 写以接收的 TxStatus, 或刷新 Tx FIFO。

- ◆ TXPAUSED[19]: MAC transmit in pause, 该位为高, 表示 MAC 发送器在暂停状态 (全双工模式下), 因此不发送任何帧。
- ◆ TFCSTS[18:17]: MAC Transmit Frame Controller status, 该位表示 MAC 发送帧控制模块。
 - 00: 空闲状态;
 - 01: 等待之前帧状态/IFG/反馈周期结束;
 - 10: 生成并发送一个暂停帧;
 - 11: 发送输入帧。
- ◆ TPESTS[16]: MAC GMII or MII Transmit protocol engine status, 该位为高表示 MAC GMII 或 MII Transmit protocol engine 在发送数据, 不在空闲状态。
- ◆ [15:10]: 保留位, 只读。
- ◆ RXFSTS[9:8]: MTL Rx FIFO Fill-level status, 该位给出了 Rx FIFO 的 fill_level 状态。
 - 00: Rx FIFO 空;
 - 01: Rx FIFO fill_level 低于流量控制不激活的阈值;
 - 10: Rx FIFO fill_level 高于流量控制激活的阈值;
 - 11: Rx FIFO 满。
- ◆ [7]: 保留位, 只读。
- ◆ RRCSTS[6:5]: MTL Rx FIFO Read controller state, 下面给出了 Rx FIFO 读控制器状态。
 - 00: 空闲状态;
 - 01: 读帧数据;
 - 10: 读帧状态;
 - 11: 刷新帧数据和状态。
- ◆ RWCSTS[4]: MTL Rx FIFO Write Controller Active Status, 该位为高表示 MTL Rx FIFO 写控制器激活且向 FIFO 发送一个接收到的帧。
- ◆ [3]: 保留位, 只读。
- ◆ RFCFCSTS[2:1]: MAC Receive Frame FIFO Controller Status, 该位为高表示 MAC

帧控制模块的小 FIFO 读写控制器在激活状态。RPESTS[1]表示小 FIFO 读控制器；RPESTS[0]表示小 FIFO 写控制器。

- ◆ RPESTS[0]: MAC GMII or MII Receive protocol engine status, 该位为高表示 MAC GMII/MII Receive protocol engine 在接收数据状态, 不在空闲状态。

7. MAC 地址 0 高位寄存器 (MAC address0 High Reg) (0xB000-0040)

建议软件初始化赋值: 0x8000FFFF

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

- ◆ AE[31]: Address enable, 地址使能, 该位一直为 1, 只读。
- ◆ [30:16]: 保留位, 只读。
- ◆ ADDRHI[15:0]: MAC Address0[47:32], 表示 MAC 地址前 6 字节的高 16bits[47:32], MAC 用该 16 位对接收帧进行滤波, 且给 Transmit Flow control(pause)frame 插入 MAC 地址。

8. MAC 地址 0 低位寄存器 (MAC address0 low Reg) (0xB000-0044)

建议软件初始化赋值: 0xFFFFFFFF

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

- ◆ ADDRLO[31:0]: MAC Address0[31:0], 包含前 6-byte MAC 地址的低 32 位。MAC 用该 32 位对接收帧进行滤波, 且给 Transmit Flow control(pause)frame 插入 MAC 地址。

9. MAC 地址 1 高位寄存器 (MAC address1 high Reg) (0xB000-0048)

建议软件初始化赋值: 0x0000FFFF

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AE	SA	MBC						保留								ADDRHI															
r/w	r/w	r/w						r								r/w															
0	0	000000						00000000								0xFFFF															

- ◆ AE[31]: Address enable, 该位置位, 地址滤波器模块用第二个 6-byte MAC 地址完美滤波。该位复位, 地址滤波器模块忽略滤波。
- ◆ SA[30]: Source address, 该位置位, MAC Address[47:0]与接收帧的 SA 比较。该位复位, MAC Address[47:0]与接收帧的 DA 比较。
- ◆ MBC[29:24]: Mask byte control, 高表示 MAC 不与相关接收帧的 DA 或 SA 比较。每一位控制屏蔽不同的字节。

Bit29: Reg18[15:8]

Bit29: Reg18[7:0]

Bit29: Reg18[31:24]

...

Bit29: Reg19[7:0]

- ◆ [23:16]: 保留位, 只读。
- ◆ ADDRHI[15:0]: MAC Address1[47:32], 该位包含 MAC 地址的第二个 6-byte 的高 16 位[47:32]。

10. MAC 地址 1 低位寄存器 (MAC address1 low Reg) (0xB000-004C)

建议软件初始化赋值: 0xFFFFFFFF

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

- ◆ ADDRLO[31:0]: MAC Address[31:0], 包含 MAC 地址的第二个 6-byte 的低 32 位[31:0]。

11. 自协商控制寄存器 (AN control Reg) (0xB000-00C0)

建议软件初始化赋值: 0x00000001

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留													SGMRAL	LR	ECD	保留	ELE	保留	ANE	保留	RAN	保留									
r													r/w	r/w	r/w	r	r/w	r	r/w	r	r/ws/sc	r/w									
0000000000000													0	0	0	0	0	0	0	00	0	000000000									

- ◆ [31:19]: 保留位，只读。
- ◆ SGMRAL[18]: SGMII RAL Control, 该位置位, SGMRAL 必须用在 MAC 与 MAC 直连的情况下, 没有 PHY, 且速度固定。该位复位, SGMRAL 操作根据 SGMII PHY 接收的链路速度状态。
- ◆ LR[17]: Lock to reference, 该位置位, 表示使能 PHY 位 125MHz。该位控制 TBI, RTBI, SGMII 的 pcs_lck_ref_o 信号。
- ◆ ECD[16]: Enable comma detect, 该位置位, 表示使能 PHY 的逗号检测和字同步。该位控制 TBI, RTBI, SGMII 的 pcs_en_cdet_o 信号。
- ◆ [15]: 保留位，只读。
- ◆ ELE[14]: External loopback enable, 该位置位, 引起 PHY 将发送数据返回到接收路径, 且 pcs_ewrap_o 信号高有效。
- ◆ [13]: 保留位，只读。
- ◆ ANE[12]: Auto_negotiation enable, 该位置位, 使能 MAC 的 auto_negotiation。
- ◆ [11:10]: 保留位，只读。
- ◆ RAN[9]: Restart auto_negotiation, 该位置位, 且 bit12 ANE 置位, 引起 auto_negotiation 重新开始。在 auto_negotiation 开始之后, 该位自清除。在正常模式下, 该位为 0。
- ◆ [8:0]: 保留位，只读。

12. 自协商状态寄存器 (AN status Reg) (0xB000-00C4)

建议软件初始化赋值: 0x00000008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
保留																							ES	保留	ANC	保留	ANA	LS	保留								
r																							r	r	r	r	r	r/ss/c/llo	r								
0x0																							0	00	0	0	1	0	00								

- ◆ [31:9]: 保留位，只读。
- ◆ ES[8]: Extended status，选择 SGMII 接收时，该位为低。只读。
- ◆ [7:6]: 保留位，只读。
- ◆ ANC[5]: Auto_negotiation complete，该位置位，表示 Auto_negotiation 完成。
- ◆ [4]: 保留位，只读。
- ◆ ANA[3]: Auto_negotiation ability，MAC 支持 Auto_negotiation 模式时，该位一直位高。
- ◆ LS[2]: Link status，该位表示无论数据通道连接还是断开，对于 TBI，RTBI，SGMII 接口，如果 ANEG 发生，链路就是断的，数据不能通过链路传输。
- ◆ [1:0]: 保留位，只读。

13. 看门狗超时设定寄存器（Watchdog timeout Reg）（0xB000-00DC）

建议软件初始化赋值：0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留															PWE	保留	WTO														
r															r/w	r	r/w														
0															0	00	0														

- ◆ [31:17]: 保留位，只读。
- ◆ PWE[16]: Programmable watchdog enable，该位置位，且 MAC reg0 的 bit23 复位，WTO 用作一个接收帧的看门狗超时设定。当该位清零，一帧接收数据的看门狗超时设定有 reg0 的 bit23 和 bit20 控制。
- ◆ [15:14]: 保留位，只读。
- ◆ WTO[13:0]: Watchdog timeout，PWE 置位且 MAC reg0 的 bit23 复位，WTO 用作一个接收帧的看门狗超时设定。如果接收帧的长度超过该区域的长度，该帧终止，并变成一个错误帧。

3.6.3 发送描述符相关寄存器

1. 发送描述符 0（Transmit Descriptor 0，TDES0）（BASE_ADDR1^注）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15
OWN	IC	LS	FS	DC	DP	TTSE	CRC	CIC	TER	TCH	VLIC	TTSS	IHE	ES		

14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JT	FF	IPE	LOC	NC	LC	EC	VF	CC				ED	UF	DB

位	描述
31	OWN: Own Bit 置位时，该位表示描述符归 DMA 所有。当此位重置时，表示描述符归主机所有。DMA 完成帧传输或完全读取描述符中分配的缓冲区时，DMA 清除此位。必须在所有后续描述符之后设置了属于同一帧的设置帧的第一个描述符的所有权位。这避免了在获取描述符和设置所有权位的驱动程序之间可能存在竞争条件。
30	IC: Interrupt on Completion 置位时，该位在发送当前帧后设置发送中断（寄存器 5[0]）。该位仅在最后一个段位（TDES0[29]）置 1 时有效。
29	LS: Last Segment 置位时，该位指示缓冲区包含帧的最后一段。设置此位时，TDES1 中的 TBS1 或 TBS2 字段应具有非零值。
28	FS: First Segment 置位时，该位表示缓冲区包含帧的第一段。
27	DC: Disable CRC 当该位置 1 时，MAC 不会在发送结束时附加循环冗余校验（CRC）帧。仅在设置了第一个段（TDES0[28]）时才有效。
26	DP: Disable Pad 设置时，MAC 不会自动将填充添加到短于 64 字节的帧。当该位复位时，DMA 会自动将填充和 CRC 添加到短于 64 字节的帧中，并且尽管存在 DC（TDES0[27]）位的状态，仍会添加 CRC 字段。仅在设置了第一个段（TDES0[28]）时才有效。
25	TTSE: Transmit Timestamp Enable 置 1 时，该位使能 IEEE1588 硬件时间戳，用于引用的发送帧描述。仅当在核心配置期间选择了启用 IEEE1588 时间戳选项且设置了第一段控制位（TDES0[28]）时，此字段才有效。

位	描述
24	CRC: CRC Replacement Control 置位时，MAC 用重新计算的 CRC 字节替换发送的数据包的最后四个字节。该主机应确保在从发送缓冲区传输的帧中存在 CRC 字节。在核心期间选择“在 TX 上启用 SA，VLAN 和 CRC 插入”选项时，此位有效配置和第一段控制位（TDES0[28]）置位。此外，仅更换 CRC 当位 27（DC）设置为 1 时。
23:22	CIC: Checksum Insertion Control 这些位控制校验和计算和插入。以下列表描述了位编码： ■2'b00：禁用校验和插入。 ■2'b01：仅启用 IP 标头校验和计算和插入。 ■2'b10：启用 IP 头校验和和有效负载校验和计算和插入，但伪头校验和不是用硬件计算的。 ■2'b11：启用 IP 头校验和和有效负载校验和计算和插入，以及伪头校验和以硬件计算。在核心有效配置期间，当选择启用传输完整 TCP/IP 校验和（类型 2）选项时，这个字段有效并且此第一字段控制位（TDES0[28]）置位。
21	TER: Transmit End of Ring 置位时，该位指示描述符列表到达其最终描述符。DMA 返回基地列表的地址，创建描述符环。
20	TCH: Second Address Chained 置位时，该位表示描述符中的第二个地址是下一个描述符地址而不是第二个缓冲区地址。当 TDES0[20]置位时，TBS2（TDES1[28:16]）是“不关心”值。TDES0[21]优先于 TDES0[20]。
19:18	VLIC: VLAN Insertion Control 置位时，这些位请求 MAC 在发送帧之前执行 VLAN 标记或取消标记。如果为 VLAN 标记修改帧，MAC 自动重新计算并替换 CRC 字节。 以下列表描述了这些位的值： ■2'b00：不添加 VLAN 标记。 ■2'b01：传输前从帧中删除 VLAN 标记。此选项仅应与。一起使用 VLAN 帧。 ■2'b10：插入 VLAN 标签，其标签值在寄存器 353 中编程（VLAN 标签包含或更换注册）。 ■2'b11：使用寄存器 353（VLAN 标记包含或替换登记册）中编程的标记值替换帧中的 VLAN 标记。此选项仅应与 VLAN 帧一起使用。 在核心有效配置期间，当启用 SA，VLAN 和 CRC 插入时，这个字段有效并且此第一字段控制位（TDES0[28]）置位。
17	TTSS: Transmit Timestamp Status

位	描述
	该字段用作状态位以指示针对所描述的发送帧捕获时间戳。 当该位置 1 时，TDES2 和 TDES3 具有为发送帧捕获的时间戳值。 这个字段是仅在描述符的 Last Segment 控制位 (TDES0[29]) 置位时有效。
16	IHE: IP Header Error 置位时，该位指示 MAC 发送器检测到 IP 数据报报头中的错误。 该发送器检查 IPv4 数据包中的报头长度与从应用程序接收到的报头字节数，如果存在不匹配，则表示错误状态。对于 IPv6 帧，如果主标头长度不是 40 个字节则会报告了报头错误。此外，IPv4 或 IPv6 的以太网长度/类型字段值必须与数据包收到的 IP 头版本匹配。对于 IPv4 帧，如果指示 Header Length 字段的值小于 0x5，错误状态也是如此。 该位仅在启用 Tx 校验和卸载时有效。否则，它是保留的。如果 COE 检测到 IP 头错误，如果以太网类型字段指示 IPv4 有效负载，它仍会插入 IPv4 报头校验和。
15	ES: Error Summary Indicates the logical OR of the following bits: ■TDES0[14]: Jabber Timeout ■TDES0[13]: Frame Flush ■TDES0[11]: Loss of Carrier ■TDES0[10]: No Carrier ■TDES0[9]: Late Collision ■TDES0[8]: Excessive Collision ■TDES0[2]: Excessive Deferral ■TDES0[1]: Underflow Error ■TDES0[16]: IP Header Error ■TDES0[12]: IP Payload Error
14	JT: Jabber Timeout 置位时，该位表示 MAC 发送器经历了 jabber 超时。该位仅在寄存器 0 (MAC 配置寄存器) 的 Bit22 (Jabber 禁用) 未被设置时置位。
13	FF: Frame Flushed 置位时，该位表示由于由 CPU 给出软件 Flush 命令，DMA 或 MTL 刷新了帧。
12	IPE: IP Payload Error 置位时，该位指示 MAC 发送器检测到 TCP，UDP 或 ICMP IP 数据报有效负载中的错误。发送器检查 IPv4 或 IPv6 报头中接收的有效负载长度与从应用程序收到的 TCP，UDP 或 ICMP 报文字节的实际数量在发生不匹配时发出错误状态。
11	LOC: Loss of Carrier

位	描述
	置位时，该位表示帧传输期间发生载波丢失（即 phy_crs_i 在帧传输期间，信号对于一个或多个传输时钟周期是无效的。这仅适用于当 MAC 以半双工模式操作时，无冲突地发送的帧。
10	NC: No Carrier 置位时，该位指示在传输期间来自 PHY 的载波侦听信号未被置位。
9	LC: Late Collision 置位时，该位表示帧传输中止因冲突窗口（64 字节时，包括前导码，MII 模式和 512 字节时间，包括前同步码和载波扩展，在 GMII 模式下）。如果设置了下溢错误位，则该位无效。
8	EC: Excessive Collision 置位时，该位表示在尝试 16 次连续冲突后传输中止传输当前帧。如果设置了寄存器 0（MAC 配置寄存器）中的位 9（禁止重试）位，在第一次冲突后设置该位，并中止帧的传输。
7	VF: VLAN Frame 置位时，该位表示发送的帧是 VLAN 类型的帧。
6:3	CC: Collision Count(Status field) 这些状态位表示在帧发送之前发生的冲突数。当设置过度冲突位（TDES0[8]）时该位无效。核心仅更新此状态字段仅在半双工模式或者 SLOTNUM: AV 模式下的插槽号控制位。这些位指示应从由 TDES2 或 TDES3 处理相应缓冲区中提取数据的时隙间隔。当获取发送描述符时，DMA 将该字段中的时隙号值与时隙进行比较核心中保留的间隔（寄存器 11xx）。仅当存在匹配时，它才从缓冲区中获取数据值。这些位仅对 AV 通道有效（不是通道 0）。
2	ED: Excessive Deferral 置位时，该位表示由于超过 24,288 位的过度延迟，传输已结束。如果位 4（延迟检查）位为 1 时（在 1,000-Mbps 模式下为 155,680 位，或者如果启用了巨型帧）寄存器 0（MAC 配置寄存器）设置为高电平。
1	UF: Underflow Error 置位时，该位指示 MAC 中止帧，因为数据从主机到达存储较晚。下溢错误表示 DMA 在发送帧时遇到空发送缓冲区。传输过程进入 Suspended 状态并设置 Transmit Underflow（寄存器 5[5]）和发送中断（寄存器 5[0]）。
0	DB: Deferred Bit 置位时，该位表示由于载波的存在，MAC 在传输之前推迟。这个 bit 仅在半双工模式下有效。



2. 发送描述符 1 (Transmit Descriptor 1, TDES1) (BASE_ADDR1+0x4)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SAIC			TBS2										保留			TBS1															

位	描述
31:29	SAIC: SA Insertion Control 这些位请求 MAC 使用 MAC 地址 0 寄存器中给出值添加或替换以太网帧中的源地址字段。如果在帧中修改源地址字段，则 MAC 自动重新计算并替换 CRC 字节。 位 31 指定用于源地址插入或替换的 MAC 地址寄存器 (1 或 0) 值。以下列表描述了位[30:29]的值： ■2'b00: 不包括源地址。 ■2'b01: 包含或插入源地址。为了可靠传输，应用程序必须提供没有源地址的框架。 ■2'b10: 替换源地址。为了可靠传输，应用程序必须提供源地址的帧。 ■2'b11: 保留。 当在核心配置期间启用 SA VLAN 和 TX 上的 CRC 插入时，这些位在 GMAC-DMA, GMAC-AXI 和 GMAC-AHB 配置中有效，并且第一段控制位 TDES0[28]) 已设定。
28:16	TBS2: Transmit Buffer 2 Size 该字段表示以字节为单位的第二个数据缓冲区大小如果设置了 TDES0[20]，则该字段无效。
15:13	保留位。
12:0	TBS1: Transmit Buffer 1 Size 这些位表示第一个数据缓冲区字节大小，以字节为单位。如果此字段为 0，则 DMA 忽略此缓冲区和使用缓冲区 2 或下一个描述符，具体取决于 TCH 的值 (TDES0[20])。

3. 发送描述符 2 (Transmit Descriptor 2, TDES2) (BASE_ADDR1+0x8)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
B1AP																															



位	描述
31:0	B1AP: Buffer 1 Address Pointer 这些位指示缓冲区 1 的物理地址。缓冲区地址对齐没有限制。

4. 发送描述符 3 (Transmit Descriptor 3, TDES3) (BASE_ADDR1+0xC)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
B2AP (NDA)																															

位	描述
31:0	B2AP (NDA): Buffer 2 Address Pointer(Next Descriptor Address) 缓冲区 2 地址指针 (下一个描述符地址) 当使用描述符环结构时, 表示缓冲区 2 的物理地址。如果是第二个地址链接 (TDES1[24]) 位置 1, 该地址包含指向 Next 的物理存储器的指针描述符存在。仅当 TDES1[24]置 1 时, 缓冲区地址指针必须与总线宽度对齐。(LSB 在内部被忽略。)

5. 发送描述符 4 (Transmit Descriptor 4, TDES4) (BASE_ADDR1+0x10)

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

位	描述
31:0	保留。

6. 发送描述符 5 (Transmit Descriptor 5, TDES5) (BASE_ADDR1+0x14)

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

位	描述
31:0	保留。

7. 发送描述符 6 (Transmit Descriptor 6, TDES6) (BASE_ADDR1+0x18)

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

位	描述
31:0	TTSL: Transmit Frame Timestamp Low 该字段由 DMA 更新，其中对应的时间戳记的最低有效 32 位发送帧。仅当设置了描述符中的 Last Segment 位 (LS) 时，此字段才具有时间戳并且时间戳状态 (TTSS) 位置 1。

8. 发送描述符 7 (Transmit Descriptor 7, TDES7) (BASE_ADDR1+0x1C)

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

位	描述
31:0	TTSH: Transmit Frame Timestamp High 该字段由 DMA 更新，其中对应的时间戳记的最低有效 32 位发送帧。仅当设置了描述符中的 Last Segment 位 (LS) 时，此字段才具有时间戳并且时间戳状态 (TTSS) 位置 1。

注：BASE_ADDR1 是发送描述寄存器的首地址，用户自行定义，无固定地址。

3.6.4 接收描述符相关寄存器

1. 接收描述符 0 (Receive Descriptor 0, RDES0) (BASE_ADDR2^注)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OWN	AFM	FL														ES	DE	SAF	LE	OE	VL AN	FS	LS	TA_ GF	LC	FT	RWT	RE	DE	CE	ESA

位	描述
31	OWN: Own Bit. 当置为高时，表示该描述符由以太网的 DMA 所拥有；当该位为低时，表明该描述符由主设备所拥有。在完成数据帧接收或当该描述符相关的 buffer 满时，DMA 清除该位。
30	AFM[30]: Destination Address Filter Fail. 当置为高时，表明一帧数

位	描述
	据在 MAC 中的目的地址滤波过程中失败。
29:16	FL[29:16]: Frame Length。表明了传输到主存中的接收数据帧的字节长度；当 RDES0[8]置为高，且 RDES0[14]或者 RDES0[11]为低时，该区域有效。
15	ES[15]: Error Summary。表明以下几个 bit 位的逻辑或： ■ RDES0[1]: CRC 错误 ■ RDES0[3]: 接收错误 ■ RDES0[4]: 看门狗超时 ■ RDES0[6]: 延迟冲突 ■ RDES0[7]: 巨大帧 ■ RDES0[11]: 上溢错误 ■ RDES0[14]: 描述符错误 ■ RDES4[4:3]: IP 头或者有效载荷错误 ■ 当 RDES0[8]置为高时该区域有效
14	DE[14]: Descriptor Error。当置为高时，表明一帧数据被一帧不适应当前描述符 buffer 中打断，且 DMA 不拥有下一个描述符。帧被打断。仅当 RDES0[8]置高时该位有效。
13	SAF[13]: Source Address Filter Fail。当置高时，该位表明数据帧中的 SA 区域在以太网的 SA 滤波时失败了。
12	LE[12]: Length Error。该位置高时，表明接收到的数据帧的实际长度，且长度/类型区域不相符。该位仅在 RDES0[5]为低时才有效。
11	OE[11]: Overflow Error。该位置高时，表明接收到的数据帧由于在 MTL 中溢出 buffer 而被损坏。
10	VLAN[10]: VLAN Tag。该位置高时，表明该描述符指向的数据帧是被 MAC 标记的 VLAN 帧。
9	FS[9]: First Descriptor。该位置高时，表明该描述符包含数据帧的第一个 buffer。如果第一个 buffer 的大小为 0，第二个 buffer 包含数据帧的开始。如果第二个 buffer 的大小也是 0，则下一个描述符包含数据帧的开始。
8	LS[8]: Last Descriptor。该位置高时，表明该描述符所指向的 buffer 是数据帧的最后 buffer。
7	TA_GF[7]: Timestamp Available or Giant Frame。该位置高时表明时间

位	描述
	戳的快速映像被写入到 RDES6 和 RDES7 中，仅当 RDES0[8]为高时该位有效；此外，该位置高还表明了巨大帧状态。长度超过 1518 字节（VLAN 为 1522 字节）的普通数据帧以及长度超过 9018 字节（VLAN 为 9022 字节）的 Jumbo 数据帧。
6	LC[6]: Late Collision。该位置高表明在半双工模式下接收数据时发生延迟冲突。
5	FT[5]: Frame Type。该位置高时，表明接收到的数据帧时以太网类型帧（LT 区域大于或者等于 1536）。该位为低时，表明接收到的数据帧是一个 IEEE802.3 数据帧。对于长度小于 14 字节的小数据帧该位无效。
4	RWT[4]: Receive Watchdog Timeout。该位置高时，表明在接收当前数据帧时，接收看门狗计数器已经计满，且在看门狗超时后该数据帧会被截断。
3	RE[3]: Receive Error。该位置高时，表明在数据帧接收过程中，当 gmii_rxdv_i 为高时，gmii_rxer_i 也为高。该错误同样包括半双工的 GMII 模式下的载波扩展错误。
2	DE[2]: Dribble Bit Error。该位置高时，表明接收数据帧有一个非整数倍数的字节数。当 MII 模式下才有效。
1	CE[1]: CRC Error。该位置高时，表明接收数据帧时发生 CRC 校验错误。该位仅当 RDES0[8]置高时才有效。
0	ESA[0]: Extended Status Available。该位置高时表明 RDES4（描述符扩展状态）有效。当 RDES0[8]置高时才有效。当 RDES0[30]置高时无效。

2. 接收描述符 1（Receive Descriptor 1, RDES1）（BASE_ADDR2+0x4）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIC	保留	RBS2														RER	RCH	保留	RBS1												

位	描述
31	DIC（Disable Interrupt on Completion）：该位的设置会阻止设置状态寄存器中 RI 位的接收。即，不使能该帧 RI 产生的中断。

位	描述
	注意：该位仅在最后描述符位（RDES0[8]）设置时才有效。
30:29	保留位。
28:16	RBS2（Receive Buffer 2 Size）：表示第 2 个数据缓存的大小，单位是字节。该缓存大小必须是 4、8 或者 16 的倍数，具体依赖于总线宽度（分别对应 32、64 或者 128），而不用管 RDES3（buffer2 的地址指针）的值是否总线宽度对齐。否则，结果未知。如果设置了 RDES1[14]，该字段无效。
15	RER（Receive End of Ring）：设置该位表示描述符列表到达它的最后一个。DMA 返回列表的基地址，创建一个描述符环。
14	RCH（Second Address Chained）：设置该位表示描述符中的第二个地址是下一个描述符的地址（the Next Descriptor address），而不是第二个缓存地址。当设置该位时，RBS2（RDES1[28:16]）的值不用管。RDES1[15]优先于 RDES1[14]。
13	保留位。
12:0	RBS1（Receive Buffer 1 Size）：表示第 1 个数据缓存的大小，单位是字节。该缓存大小必须是 4、8 或者 16 的倍数，具体依赖于总线宽度（分别对应 32、64 或者 128），而不用管 RDES2（buffer1 的地址指针）的值是否对齐。否则，结果未知。如果该字段为 0，DMA 忽略该缓存，而根据 RCH（bit 14）的值来使用 Buffer2 或者下一个描述符。

3. 接收描述符 2（Receive Descriptor 2，RDES2）（BASE_ADDR2+0x8）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
B1AP																															

位	描述
31:0	Buffer1 Address Pointer：表示 Buffer1 的物理地址。除了以下条件该地址没有其他对齐约束：当 RDES2 值用于存储帧头时，DMA 使用配置值来产生地址。帧头被传输期间，DMA 将 RDES2[3:0，2:0 或者 1:0]写 0，但是帧数据以每实际缓存地址指针位移。如果地址指针指向的缓存存储的是帧的中间或者最后部分，那么 DMA 忽略 RDES2[3:0，2:0 或者 1:0]（对应总线宽度 128，64 或者 32）。

4. 接收描述符 3（Receive Descriptor 3，RDES3）（BASE_ADDR2+0xC）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
B2AP (NDA)																															

位	描述
31:0	Buffer2 Address Pointer (Next Descriptor Address): 表示环形描述符结构中 Buffer2 的物理地址。如果设置了 Second Address Chained (RDES1[24]) 位, 该地址指向下一个描述符的物理地址。 如果设置了 RDES1[24]位, 缓存 (下一个描述符) 地址指针必须总线宽度对齐 (RDES3[3, 2 或者 1:0]=0, 对应于总线宽度 128, 64 或者 32。忽略 LSBs。)然而, 当重置 RDES1[24]位, 除了以下条件 RDES3 值没有其他约束: 当 RDES3 值用于存储帧头时, DMA 使用配置值来产生缓存地址。帧头被传输期间, DMA 将 RDES2[3:0, 2:0 或者 1:0]写 0, 但是帧数据以每实际缓存地址指针位移。如果地址指针指向的缓存存储的是帧的中间或者最后部分, 那么 DMA 忽略 RDES3[3, 2 或者 1:0] (对应总线宽度 128, 64 或者 32)。

5. 接收描述符 4 (Receive Descriptor 4, RDES4) (BASE_ADDR2+0x10)

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

位	描述
31:0	保留。

6. 接收描述符 5 (Receive Descriptor 5, RDES5) (BASE_ADDR2+0x14)

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

位	描述
31:0	保留。

7. 接收描述符 6 (Receive Descriptor 6, RDES6) (BASE_ADDR2+0x18)

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

位	描述
31:0	保留。

8. 接收描述符 7（Receive Descriptor 7，RDES7）（BASE_ADDR2+0x1C）

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

位	描述
31:0	保留。

注：BASE_ADDR2 是发送描述寄存器的首地址，用户自行定义，无固定地址。

3.7 DDR2 相关寄存器

3.7.1 DDR2 控制器寄存器

1. DDR_CTL_00

Address = 0x8800-0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
VERSION																- 保留				DRAM_CLASS				保留								START
r																				r/w												r/w
0x2041																				0												0

位号	助记符	描述
31:16	VERSION	控制器版本号，读出为 0x2041。只读。
11:8	DRAM_CLASS	DRAM 类型，“0x4”表示 DDR2，“0x6”表示 DDR3，其他状态保留。本产品仅可配置为 0x4。
0	START	“1”表示控制器启动 DRAM 初始化。

2. DDR_CTL_01

Address = 0x8800-0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
READ_DATA_FIFO_DEPTH								保留								MAX_CS_REG	保留				MAX_COL_REG				保留			MAX_ROW_REG			
r																r					r							r			
0x00																0x1					0xb							0x10			

位号	助记符	描述
31:24	READ_DATA_FIFO_DEPTH	控制器读数据 FIFO 深度。只读。
17:16	MAX_CS_REG	最大有效片选信号数量。只读。
11:8	MAX_COL_REG	DRAM 列地址最大宽度。只读。
3:0	MAX_ROW_REG	DRAM 行地址最大宽度。只读。

3. DDR_CTL_02

Address = 0x8800-0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MEMCD_RMODW_FIFO_DEPTH								WRITE_DATA_FIFO_PTR_WIDTH								WRITE_DATA_FIFO_DEPTH								READ_DATA_FIFO_PTR_WIDTH							
r								r								r								r							
0x00								0x0								0x0								0x0							

位号	助记符	描述
31:24	MEMCD_RMODW_FIFO_DEPTH	控制器读/改/写 FIFO 深度。只读。
23:16	WRITE_DATA_FIFO_PTR_WIDTH	控制器写数据 FIFO 指针的宽度。只读。
15:8	WRITE_DATA_FIFO_DEPTH	控制器写数据 FIFO 深度。只读。
7:0	READ_DATA_FIFO_PTR_WIDTH	控制器读数据 FIFO 指针的宽度。只读。

4. DDR_CTL_43

Address = 0x8800-00AC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留							ECC_WRITEBACK_EN	XOR_CHECK_BITS								保留							FWC	保留							ECC_EN
							r/w	r/w															w								r/w
							0x0	0x0															0x0								0x0

位号	助记符	描述
24	ECC_WRITEBACK_EN	ECC 自动回写使能位，ECC 使能条件下，如果该位使能，读操作发生 1bit 可纠正的错误时，自动写回正确的数据，“1”表示使能，“0”表示禁能。
23:16	XOR_CHECK_BITS	写检查使能条件下，下一个写操作产生的校验位与该位域的值进行异或，产生的结果作为新的校验码写入存储体。xor_check_bits[7:0]对应 64bit 数据。
8	FWC	写检查使能位，该位使能时，下一个写操作产生的校验位与 xor_check_bits[7:0]进行异或，产生的结果作为新的校验码写入存储体，“1”表示使能，“0”表示禁能。只可写，读出始终为 0。当下一个写操作完成时，该位自动清零。
0	ECC_EN	ECC 使能位，“1”表示使能，“0”表示禁能。

5. DDR_CTL_44

Address = 0x8800-00B0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																															ECC_DISABLE_W_UC_ERR
																															r/w
																															0x0

位号	助记符	描述
0	ECC_DISABLE_W_UC_ERR	读改写操作时，发生不可纠正的错误，自动毁坏整个 64bit 数据对应的 ECC 码，“1”表示禁能，“0”表示使能。

6. DDR_CTL_45

Address = 0x8800-00B4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_U_ADDR [31:0]																															
r																															
0x0																															

位号	助记符	描述
31:0	ECC_U_ADDR	记录 2bit 不可纠正的 ECC 错误发生时的读数据地址，只读。

7. DDR_CTL_46

Address = 0x8800-00B8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																ECC_U_SYND								保留							
																r															
																0x0															

位号	助记符	描述
15:8	ECC_U_SYND	记录 2bit 不可纠正的 ECC 错误发生时的读数据校正因子，只读。
0	ECC_U_ADDR	记录 2bit 不可纠正的 ECC 错误发生时的读数据地址，只读。

8. DDR_CTL_47

Address = 0x8800-00BC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_U_DATA [31:0]																															
r																															
0x0																															

位号	助记符	描述
31:0	ECC_U_DATA[31:0]	记录 2bit 不可纠正的 ECC 错误发生时的低 32bit 数据, 只读。

9. DDR_CTL_48

Address = 0x8800-00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_U_DATA [63:32]																															
r																															
0x0																															

位号	助记符	描述
31:0	ECC_U_DATA[63:32]	记录 2bit 不可纠正的 ECC 错误发生时的高 32bit 数据, 只读。

10. DDR_CTL_49

Address = 0x8800-00C4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_C_ADDR [31:0]																															
r																															
0x0																															

位号	助记符	描述
31:0	ECC_C_ADDR	记录 1bit 可纠正的 ECC 错误发生时的读数据地址, 只读。

11. DDR_CTL_50

Address = 0x8800-00C8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0			
保留																ECC_C_SYND								保留										ECC_C_ADDR [32]
																r																		r
																0x0																		0x0

位号	助记符	描述
15:8	ECC_C_SYND	记录 1bit 可纠正的 ECC 错误发生时的读数据校正因子，只读。
0	ECC_C_ADDR	记录 1bit 可纠正的 ECC 错误发生时的读数据地址，只读。

12. DDR_CTL_51

Address = 0x8800-00CC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_C_DATA [31:0]																															
r																															
0x0																															

位号	助记符	描述
31:0	ECC_C_DATA[31:0]	记录 1bit 可纠正的 ECC 错误发生时的低 32bit 数据，只读。

13. DDR_CTL_52

Address = 0x8800-00D0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_C_DATA [63:32]																															
r																															
0x0																															

位号	助记符	描述
31:0	ECC_C_DATA[63:32]	记录 1bit 可纠正的 ECC 错误发生时的高 32bit 数据，只读。

14. DDR_CTL_53

Address = 0x8800-00D4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留					ECC_C_ID										保留					ECC_U_ID											
					r															r											
					0x0															0x0											

位号	助记符	描述
25:16	ECC_C_ID	记录 1bit 可纠正的 ECC 错误发生时的源 ID，只读。
9:0	ECC_U_ID	记录 2bit 不可纠正的 ECC 错误发生时的源 ID，只读。

15. DDR_CTL_66

Address = 0x8800-0108

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留					INT_STATUS																										
					r																										
					0x0																										

位号	助记符	描述
26	INT_STATUS	所有低位 bit 的逻辑或，只读。
25	INT_STATUS	用户发起的 DLL 重新同步完成，只读。
24	INT_STATUS	初始化完成后检测到 dfi_init_complete 信号状态发生变化，只读。
23	INT_STATUS	INHIBIT_DRAM_CMD 置位阻止命令队列成功，只读。
22	INT_STATUS	寄存器接口发起的写模式寄存器完成，可以发送另一个模式寄存器的写操作，只读。
21	INT_STATUS	低功耗接口（Low Power Interface, LPI）发生超时错误，只读。
20	INT_STATUS	平衡操作完成，只读。
19	INT_STATUS	请求平衡操作，只读。

位号	助记符	描述
18	INT_STATUS	DFI 更新错误，只读。 错误信息可以查看 UPDATE_ERROR_STATUS 参数。
17	INT_STATUS	写平衡错误，只读。 错误信息可以查看 WRLVL_ERROR_STATUS 参数。
16	INT_STATUS	读平衡门训练错误，只读。 错误信息可以查看 RDLVL_ERROR_STATUS 参数。
15	INT_STATUS	读平衡错误，只读。 错误信息可以查看 RDLVL_ERROR_STATUS 参数。
14	INT_STATUS	CL 配置为 3 的同时 ODT 使能，只读。 不支持这种配置情况。
13	INT_STATUS	用户编程了一种无效配置，只读。 举例：突发长度=2 时，设置 param_reduc；突发长度=2 时，配置 1:2 的 MC:PHY 时钟比。
12	INT_STATUS	检测到 wrap 时钟跨页，只读。 这种情况可能导致存储器数据毁坏，不允许发生。
11	INT_STATUS	BIST 操作完成，只读。
10	INT_STATUS	低功耗操作完成，只读。
9	INT_STATUS	控制器初始化完成，只读。
8	INT_STATUS	端口命令通道发生错误，只读。
7	INT_STATUS	存在一个以上未能执行 ECC 写回命令，只读。
6	INT_STATUS	检测到多个不可纠正的 ECC 错误，只读。
5	INT_STATUS	检测到一个不可纠正的 ECC 错误，只读。
4	INT_STATUS	检测到多个可纠正的 ECC 错误，只读。
3	INT_STATUS	检测到一个可纠正的 ECC 错误，只读。
2	INT_STATUS	发生多个已定义的物理存储空间以外的访问，只读。
1	INT_STATUS	发生一个已定义的物理存储空间以外的访问，只读。
0	INT_STATUS	DFI 总线上存储器复位有效，只读。

16. DDR_CTL_67

Address = 0x8800-010C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留						INT_ACK																									
						w																									
						0x0																									

位号	助记符	描述
25:0	INT_ACK	“1”表示清零 int_status 参数相应位，只写，读出为 0x0。

17. DDR_CTL_68

Address = 0x8800-0120

位号	助记符	描述
26:0	INT_MASK	只读，中断状态mask

位号	助记符	描述
26	INT_MASK	“0”表示中断屏蔽由 bit[25:0]分别控制； “1”表示屏蔽所有中断，与 bit[25:0]的配置无关。
25:0	INT_MASK	“0”表示不屏蔽 INT_STATUS 相应位的中断； “1”表示屏蔽 INT_STATUS 相应位的中断。

18. DDR_CTL_118

Address = 0x8800-01D8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DLL_RST_ADJ_DLY								DLL_RST_DELAY																保留							MEM_RST_VALID
r/w								r/w																							r
0x0								0x0																							0x0

位号	助记符	描述
31:24	DLL_RST_ADJ_DL Y	start 信号有效与 DLL 复位信号 dll_rst_n 有效之间的最小时钟周期数长度。
23:8	DLL_RST_DELAY	DLL 复位信号 dll_rst_n 有效的最小时钟周期数。
7:1	保留	
0	MEM_RST_VALID	表示 mem_rst_valid 信号的状态，只读。

3.7.2 DDR2 PHY 寄存器

1. DDR_PHY_00/16/32/48/64

Address = 0x8800-0400/440/480/4C0/500

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEN_PHY_DQ_TIMING_REG_X																															
r/w																															
0x0																															

位号	助记符	描述
15:12	tssel_start	定义 DQ 动态终端选择使能时间，每 bit 变化表示 1/2 周期。
11:8	tssel_end	定义 DQ 动态终端选择禁能时间，每 bit 变化表示 1/2 周期。
6:4	oe_start	定义 DQ 输出使能窗口起始点，每 bit 变化表示 1/2 周期，默认值 0x2。
2:0	oe_end	定义 DQ 输出使能窗口结束点，每 bit 变化表示 1/2 周期，该值最小配置为 bit[6:4]+2，默认值 0x7。

2. DDR_PHY_01/17/33/49/65

Address = 0x8800-0404/444/484/4C4/504

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEN_PHY_DQS_TIMING_REG_X																															
r/w																															
0x0																															

位号	助记符	描述
7:4	oe_start	定义 DQS 输出使能窗口起始点，每 bit 变化表示 1/2 周期，默认值 0x2。
3:0	oe_end	定义 DQ 输出使能窗口结束点，每 bit 变化表示 1/2 周期，该值最小配置为 bit[7:4]+2，默认值 0x7。

3. DDR_PHY_02/18/34/50/66

Address = 0x8800-0408/448/488/4C8/508

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEN_PHY_GATE_LPBK_CTRL_REG_X																															
r/w																															
0x0																															

位号	助记符	描述
31	sw_dqs_phase_bypass	设置 half_cycle_shift 的值。“0”表示 half_cycle_shift 的值由鉴相电路决定，“1”表示 half_cycle_shift 的值由延时线延迟设置决定，延时线设置为 0x0 – 0x7 时，half_cycle_shift = 0，延时线设置为 0x80 – 0xff 时，half_cycle_shift = 1。
29	enable_half_cas	
27	sw_half_cycle_shift	当 DDR2_PHY_ctrl_reg 的 bit8 (en_sw_half_cycle) 为 1 时，该值为 1 时，表示在写平衡延时链中 clock 与 clk_wr dqs 之间加入 1/2 时钟周期延时，为 0 时，无影响。
25	wrlvl_clkperiod_delay	
18:16	param_phase_detect_sel	设置插入鉴相器模块触发器间的延时单元数，默认值为 0x1。用于解决 clk (180 度相移) 与 clk_wr 之间的跨时钟域问题，如果 DLL 锁定条件不满足，则需要增加延时单元。“000”表示 1 个延时单元，...，“111”表示 8 个延时单元。
15:13	lpbk_err_check_timing	设置 LFSR 和回路错误检测逻辑之间的延时。内部

位号	助记符	描述
		loopback 时，一般设为 4；外部 loopback 时，设为 7-rd_dly_sel。
12	lpbk_fail_muxsel	配置 den_phy_obs_reg_0_X 寄存器 bit[23:8](lpbk_dq_data)和 bit[5:4](lpbk_dm_data)的输出类型，“0”表示输出期望值，“1”表示输出实际值。
11:10	lpbk_ctrl	loopback 控制。“00”表示正常操作模式，“01”表示使能 loopback 写模式，“10”表示停止 loopback，“11”表示清零 loopback 寄存器。
9	lpbk_internal	控制 loopback 读通路。“0”表示外部 loopback，“1”表示内部 loopback。
8	lpbk_en	控制内部写通路。“0”表示正常操作，“1”表示使能 loopback。
7:5	gate_error_delay	
4:3	gate_close_cfg	设置 DQS 门关闭延时。当 IO_Pad_Delay + (Flight Time * 2) + IO_Pad_Input_Delay + tDQSCK(max)大于 2 个时钟周期时，该值必须设置为非 0 值。 “00” = 1 cycle, “01” = 1 cycle, “10” = 2 cycles, “11” = 3 cycles。
2:0	gate_cfg	设置 dfi_rddata_en 信号的延时时间。 当 IO pad 具备输入使能功能时，该值配置为 Read Latency - 1 - round up(IE delay / tCK)； 当 IO pad 不具备输入使能功能时，该值配置为 Read Latency - 1； 当使能 loopback 功能时，该值配置为 0。

4. DDR_PHY_03/19/35/51/67

Address = 0x8800-040C/44C/48C/4CC/50C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEN_PHY_READ_CTRL_REG_X																															
r/w																															
0x0																															

位号	助记符	描述
3:0	rd_dly_sel	io_datain FIFO 时序的读延时设置，设置为 ASYNC_DELAY_max/tCK 向上取整。 内部 loopback 时，一般设为 3； 外部 loopback 时，设为 3+lpkb_diff，其中 lpkb_diff = (IO_Delay_out + IO_Delay_in + 0.75*clkperiod)/clock 取整。

5. DDR_PHY_04/20/36/52/68

Address = 0x8800-0410/450/490/4D0/510

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PHY_DLL_MASTER_CTRL_REG_X																															
r/w																															
0x0																															

位号	助记符	描述
28:25	pop_dly_sel	设置从 gather FIFO 中移除数据的延时，设置为 rd_dly_sel[3:0]的最大值 + 1。
23	param_dll_bypass_mode	控制主/从 DLL 的旁路模式。为“1”时，主 DLL 禁能，主延时线仅插入一个延时单元，从延时线以绝对延时单元译码延时，dll_lock 参数强制为 1；为“0”时，DLL 工作在正常模式，从延时线设置被译码为主延时线的分数延时。
22:20	param_phase_detect_sel	设置插入鉴相器模块触发器间的延时单元数。用于解决 clk 与 clk_wrdqs 之间的跨时钟域问题，“000”表示 1 个

位号	助记符	描述
		延时单元，...，“111”表示 8 个延时单元。
18:16	param_dll_lock_num	该值设置为 N 时，表示 lock_inc_dbg/ lock_dec_dbg 连续出现 N 个“1”时，dll_unlock_cnt 的值加 1。
7:0	param_dll_start_point	DLL 初始延时值。基于给定的延时单元最大值，该值需小于 7/8 个时钟周期。举例：时钟频率为 400MHz（时钟周期为 2.5ns），给定的延时单元最大值为 80ps，则该值最大为 $(2.5 \times 7/8) / 0.08 = 27$ 。该值设置越大，则 DLL 锁定时间越短，但可能会导致锁定错误；相反，该值设置越小，DLL 锁定时间越长，但不会出现锁定错误，例如 0x4。

6. DDR_PHY_05/21/37/53/69

Address = 0x8800-0414/454/494/4D4/514

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PHY_DLL_SLAVE_CTRL_REG_X																															
r/w																															
0x0																															

位号	助记符	描述
23	phase_clk_wrds	控制写数据通路 half_cycle_shift 的值。延时位于前半周期时，该值为“0”，延时位于后半周期时，该值为“1”。
15:8	param_dll_dqs_delay	bypass 模式下，控制 clk_wr 延时，从而调整写 DQ 的时序，每 bit 代表 1/256 时钟周期。

7. DDR_PHY_06/22/38/54/70

Address = 0x8800-0418/458/498/4D8/518

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEN_PHY_OBS_REG_0_X																															
r/w																															
0x0																															

位号	助记符	描述
31:29	ie_end_cfg	
28	ie_always_on	
15:10	dq_ie_select	定义 DQ 输入使能开始和结束位置; bit[15:13]: 定义 DQ IE 开始; bit[12:10]: 定义 DQ IE 结束。
9:4	dqs_ie_select	定义 DQS 输入使能开始和结束位置; bit[9:7]: 定义 DQS IE 开始; bit[6:4]: 定义 DQS IE 结束。
3:0	rddata_en_ie_dly	dqi_rddata_en 信号延时时钟周期数, 用于标记正确情况下 DFI 读数据位置。设置与 DDR_PHY_80 bit[19:16] (rddata_en_ie_dly) 相同值即可。

8. DDR_PHY_07/23/39/55/71

Address = 0x8800-041C/45C/49C/4DC/51C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PHY_OBS_REG_0_X																															
r																															
0x0																															

位号	助记符	描述
24	dqs_error_out	逻辑门强制关闭的状态信号。“0”表示正常操作,“1”表示逻辑门被强制关闭。只读。
23:8	lpbk_dq_data	DDR2_phy_gate_lpbk_ctrl_reg_X 寄存器的 bit12 (lpbk_fail_muxsel) 为 0 时, 该位域输出实际数据; 为 0

		时，该位域输出期望数据。只读。
5:4	lpbk_dm_data	DDR2_phy_gate_lpbk_ctrl_reg_X 寄存器的 bit12 (lpbk_fail_muxsel) 为 0 时，该位域输出实际的 DM；为 0 时，该位域输出期望的 DM。只读。
3:0	lpbk_status	bit[1]: “0” 表示最近的 loopback 测试没有错误；“1” 表 示最近的 loopback 测试包含数据错误。 bit[0]: “0” 表示不处于 loopback 模式；“1” 表示处于 loopback 模式。只读。

9. DDR_PHY_08/24/40/56/72

Address = 0x8800-0420/460/4A0/4E0/520

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PHY_DLL_OBS_REG_0_X																															
r																															
0x0																															

位号	助记符	描述
31:24	lock_inc_dbg	只读。记录主 DLL 相位测试时最新的 8 次采样结果，bit24 对应最近的一次采样，bit25 对应次近的一次采样，以此 类推。当 lock_inc_dbg 与 lock_dec_dbg 的对应 bit 为 “00” 时，主 DLL 工作精度最高。
23:16	lock_dec_dbg	只读。记录主 DLL 相位测试时最新的 8 次采样结果，bit16 对应最近的一次采样，bit17 对应次近的一次采样，以此 类推。当 lock_inc_dbg 与 lock_dec_dbg 的对应 bit 为 “00” 时，主 DLL 工作精度最高。
15:8	dll_lock_value	只读。DLL 锁定时，需要的延时单元数。
7:3	dll_unlock_cnt	只读。该 param_dll_lock_num 设置为 N 时，表示 lock_inc_dbg/ lock_dec_dbg 连续出现 N 个 “1” 时， dll_unlock_cnt 的值加 1。
2:1	DLL locked mode	只读。DLL 锁定时的模式： “00” 表示 Full clock 模式；

		“01” 保留； “10” 表示 half clock 模式； “11” 表示 Saturation 模式。
0	dll_lock	只读。主 DLL 状态：“0” 表示主 DLL 未锁定，“1” 表示主 DLL 锁定。

10. DDR_PHY_09/25/41/57/73

Address = 0x8800-0424/464/4A4/4E4/524

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PHY_DLL_OBS_REG_1_X																															
r																															
0x0																															

位号	助记符	描述
23:16	decoder_out_wr	clk_wr 延时链的编码值。只读。
7:0	decoder_out_rd	读延时链的编码值。只读。

11. DDR_PHY_10/26/42/58/74

Address = 0x8800-0428/468/4A8/4E8/528

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PHY_DLL_OBS_REG_2_X																															
r																															
0x0																															

位号	助记符	描述
15:8	decoder_out_gate	门编码值。只读。
7:0	decoder_out_wr_dqs	clk_wr dqs 延时链编码值。只读。

12. DDR_PHY_80

Address = 0x8800-0540

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEN_PHY_CTRL_REG																															
r/w																															
0x0																															

位号	助记符	描述
19:16	rddata_en_ie_dly	PHY 动态输入使能逻辑使能时，延迟 dfi_rddata_en 信号的额外时钟周期数。与 den_phy_ie_timing_reg_X 的 bit[3:0]（rddata_en_ie_dly）配置为相同的值。
12	ddr3_mode	使能额外的 DQS 脉冲信号产生，用于 DDR3 或 LPDDR3 控制器。
8	en_sw_half_cycle	使能软件半周期移位。该值决定写数据在 PHY clock 的上升沿或下降沿传送至 clk_wrdqs 域。 “0”表示硬件自动控制写平衡延时线移位； “1”表示 ew_half_cycle_shift 控制写平衡延时线移位。 推荐配置硬件自动控制方式。
5	send_rdlvl_valid	
4	clear_fifo	data gather FIFO 的紧急复位信号，“1”表示清零 gather FIFO。
3:0	read_data_valid_delay	延时 dfi_rddata_en 信号 N 个时钟周期，产生 dfi_rddata_valid，N 为该位域的配置值。一般配置为 rd_dly_sel（DDR_PHY_03/19/35/51/67 的 bit[3:0]）的最大值+2。

13. DDR_PHY_81

Address = 0x8800-0544

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DEN_PHY_LP_WAKEUP_REG																															
r/w																															
0x0																															

定义PHY从低功耗模式唤醒需要的时钟周期数。

位号	助记符	描述
9	lp_clk_io_ctrl_mask	“0”表示 Deep-Sleep 模式下，时钟管脚 OE/IE 禁能； “1”表示时钟管脚 OE/IE 不受 LPI 控制。
8	lp_io_ctrl_mask	“0”表示管脚 OE/IE 由 LPI 控制； “1”表示管脚 OE/IE 不受 LPI 控制。
4	lp_clk_sel	定义 Light-Sleep 模式下，存储器时钟的行为。 “0”表示时钟开启，“1”表示时钟关闭。
3:0	lp_wakeup_reg	用于与 dfi_lp_wakeup 信号比较，决定 PHY 进入哪种低功耗模式。 dfi_lp_wakeup≤lp_wakeup_reg 时，PHY 将进入 Light-Sleep 模式； dfi_lp_wakeup>lp_wakeup_reg 时，PHY 将进入 Deep-Sleep 模式。

14. DDR_PHY_82

Address = 0x8800-0548

31:18	17:15	14:12	11:9	8:6	5:3	2:0
Reserved	tssel_off_value_data	tssel_rd_value_data	tssel_off_value_dqs	tssel_rd_value_dqs	tssel_off_value_dm	tssel_rd_value_dm
r	r/w	r	r/w	r/w	r/w	r/w
0x0	0x0	0x0	0x0	0x0	0x0	0x0

位号	助记符	描述
17:15	tssel_off_value_data	详见附录 3.7.4 TSEL/DS 描述。
14:12	tssel_rd_value_data	
11:9	tssel_off_value_dqs	
8:6	tssel_rd_value_dqs	
5:3	tssel_off_value_dm	

2:0	tssel_rd_value_dm	

15. DDR_PHY_83

Address = 0x8800-054C

31:16	15:12	11:9	8:3	2:0
Reserved	fbdk.OCD_PU	fbdk.OCD_PD	Reserved	fbdk.DS
r	r/w	r/w	r	r/w
0x0	0x0	0x0	0x0	0x0

位号	助记符	描述
15:12	fbdk.OCD_PU	上拉电阻校准值。 详见附录 3.7.3 OCD_PU 描述。
11:9	fbdk.OCD_PD	下拉电阻校准值。 详见附录 3.7.3 OCD_PD 描述。
2:0	fbdk.DS	输出阻抗 Ron 的值。 详见附录 3.7.4 DS 描述。

16. DDR_PHY_84

Address = 0x8800-0550

31:28	27:24	23:19	18:16	15:12	11:8	7:3	2:0
clk.OCD_PU	clk.OCD_PD	Reserved	clk.DS	addr/cmd.OCD_PU	addr/cmd.OCD_PD	Reserved	addr/cmd.DS
r/w	r/w	r	r/w	r/w	r/w	r	r/w
0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0

位号	助记符	描述
31:28	clk.OCD_PU	上拉电阻校准值。 详见附录 3.7.3 OCD_PU 描述。
27:24	clk.OCD_PD	下拉电阻校准值。 详见附录 3.7.3 OCD_PD 描述。
18:16	clk.DS	输出阻抗 Ron 的值。 详见附录 3.7.4 DS 描述。
15:12	addr/cmd.OCD_PU	上拉电阻校准值。 详见附录 3.7.3 OCD_PU 描述。

位号	助记符	描述
11:9	addr/cmd.OCD_PD	下拉电阻校准值。 详见附录 3.7.3 OCD_PD 描述。
2:0	addr/cmd.DS	输出阻抗 Ron 的值。 详见附录 3.7.4 DS 描述。

17. DDR_PHY_85

Address = 0x8800-0554

31:28	27:24	23:19	18:16	15:12	11:8	7:3	2:0
dqs.OCD_PU	dqs.OCD_PD	Reserved	dqs.DS	data.OCD_PU	data.OCD_PD	Reserved	data.DS
r/w	r/w	r	r/w	r/w	r/w	r	r/w
0x0	0x0	0x0	0x0	0x0	0x0	0x0	0x0

位号	助记符	描述
31:28	dqs.OCD_PU	上拉电阻校准值。 详见附录 3.7.3 OCD_PU 描述。
27:24	dqs.OCD_PD	下拉电阻校准值。 详见附录 3.7.3 OCD_PD 描述。
18:16	dqs.DS	输出阻抗 Ron 的值。 详见附录 3.7.4 DS 描述。
15:12	data.OCD_PU	上拉电阻校准值。 详见附录 3.7.3 OCD_PU 描述。
11:9	data.OCD_PD	下拉电阻校准值。 详见附录 3.7.3 OCD_PD 描述。
2:0	data.DS	输出阻抗 Ron 的值。 详见附录 3.7.4 DS 描述。

18. DDR_PHY_86

Address = 0x8800-0558

31:3	2:0
Reserved	fdbk.tsel
r	r/w
0x0	0x0

位号	助记符	描述
2:0	fdbk.tsel	输入阻抗 R_{TT} 的值。 详见附录 3.7.4 TSEL 描述。

19. DDR_PHY_87

Address = 0x8800-055C

31:3	2:0
Reserved	clk/addr/cmd.tsel
r	r/w
0x0	0x0

位号	助记符	描述
2:0	clk/addr/cmd.tsel	时钟、地址线、命令线的输入阻抗 R_{TT} 值。 详见附录 3.7.4 TSEL 描述。

3.7.3 OCD_PU/PD

OCD_PU[3:0]	Function for “Ron”and “ODT”pull up leg	OCD_PD[3:0]	Function for “Ron”and “ODT”pull down leg
0000	-7 step	0000	-7 step
0001	-6 step	0001	-6 step
0010	-5 step	0010	-5 step
0011	-4 step	0011	-4 step
0100	-3 step	0100	-3 step
0101	-2 step	0101	-2 step

0110	-1 step	0110	-1 step
0111	Normal Ron & ODT	0111	Normal Ron & ODT
1000	Normal Ron & ODT	1000	Normal Ron & ODT
1001	+1 step	1001	+1 step
1010	+2 step	1010	+2 step
1011	+3 step	1011	+3 step
1100	+4 step	1100	+4 step
1101	+5 step	1101	+5 step
1110	+6 step	1110	+6 step
1111	+7 step	1111	+7 step

注：Ron=200Ω，step=6.5Ω。

3.7.4 TSEL/DS

TSEL[2:0]	输入阻抗 R_{TT} (Ω)	DS[2:0]	输出阻抗 Ron (Ω)
000	High-Z	000	High-Z
010	100	010	200
001	$1/2 \times 100$	001	$1/2 \times 200$
011	$1/3 \times 100$	011	$1/3 \times 200$
101	$1/4 \times 100$	101	$1/4 \times 200$
110	$1/5 \times 100$	110	$1/5 \times 200$
100	$1/6 \times 100$	100	$1/6 \times 200$
111	$1/7 \times 100$	111	$1/7 \times 200$

3.8 PCI 相关寄存器

1. PCI 总线复位寄存器 (PCI_RESET)

Address = 0x8000-00D0

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

◆ [31:4]：保留。

- ◆ RESET[3:0]: PCI 总线软件复位寄存器。在主桥模式下, 写 0x5 产生 PCI 软件复位, 复位 PCI 接口 PCI 时域的寄存器及 PCI 总线上的设备, 写其他值无效; 在从桥模式下, 该寄存器无效。

2. DMA 写操作 PCI 总线起始地址寄存器 (WDMA_PCI_ADDR)

Address = 0xC000-0000

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

- ◆ ADDR[31:0]: DMA 写操作在 PCI 总线上起始地址, 在 DMA 写操作过程中, DMA 控制器将数据写入该寄存器配置的地址, 随数据传输该寄存器值自动增加。建议使用时, 寄存器后两位 (ADDR[1:0]) 设为 0。

3. DMA 写操作 AHB 总线起始地址寄存器 (WDMA_AHB_ADDR)

Address = 0xC000-0004

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

- ◆ ADDR[31:0]: DMA 写操作在 AHB 总线上起始地址, 在 DMA 写操作过程中, DMA 控制器从该寄存器配置的地址读取数据, 随着数据传输该寄存器值自动增加。建议使用时, 寄存器后两位 (ADDR[1:0]) 设为 0。

4. DMA 写操作控制寄存器 (WDMA_CONTROL)

Address = 0xC000-0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRANSFER SIZE																								COMMAND				PCI INTERRUPT	保留	STOP TRANSFER	START TRANSFER
r/w																								r/w				r/w	r	r/w	r/w
0																								0000				0	0	0	0

- ◆ TRANSFER SIZE[31:8]: DMA 写传操作时传递的字节数, 范围为 1-0xFFFFFFFF 字节。
建议进行 DMA 操作时, 以整字 (4 个字节) 倍数长度进行数据传输。
- ◆ COMMAND[7:4]: DMA 写操作在 PCI 总线上传送命令。按正文表 4-8-4 写 DMA 设置 PCI 命令, 设置其他 PCI 命令无效。
- ◆ PCI INTERRUPT[3]: 在从桥模式下, 该寄存器配置为 1, 在 DMA 写操作正常完成时产生中断请求, DMA 写传输异常退出时不产生中断请求; 该寄存器配置为 0, 在 DMA 写操作正常完成时不发出中断请求。在主桥模式下, 该寄存器无效。
- ◆ STOP TRANSFER[1]: DMA 写操作停止位, 写 1 停止当前的 DMA 写传输, 写 0 无效。该寄存器写 1 后自动清 0, 即读出值为 0。
- ◆ START TRANSFER[0]: DMA 写操作起动位, 写 1 发起 DMA 写操作, 写 0 无效。DMA 传输完成后此位自动清 0, 因此可通过此位判断 DMA 写传输是否完成。

5. DMA 读操作 PCI 总线起始地址寄存器 (RDMA_PCI_ADDR)

Address = 0xC000-0020

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

- ◆ ADDR[31:0]: DMA 读操作在 PCI 总线上起始地址, 在 DMA 读操作过程中, DMA 控制器从该寄存器配置的地址读出数据, 随数据传输该寄存器值自动增加。建议使用时, 寄存器后两位 (ADDR[1:0]) 设为 0。

6. DMA 读操作 AHB 总线起始地址寄存器 (RDMA_AHB_ADDR)

Address = 0xC000-0024

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

- ◆ ADDR[31:0]: DMA 读操作在 AHB 总线上起始地址, 在 DMA 读操作过程中, DMA 控制器将数据写入该寄存器配置的地址, 随着数据传输该寄存器值自动增加。建议使用时, 寄存器后两位 (ADDR[1:0]) 设为 0。

7. DMA 读传输控制寄存器 (RDMA_CONTROL)

Address = 0xC000-0028

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRANSFER SIZE																								COMMAND				PCI INTERRUPT	保留	STOP TRANSFER	START TRANSFER
r/w																								r/w				r/w	r	r/w	r/w
0																								0000				0	0	0	0

- ◆ TRANSFER SIZE[31:8]: DMA 读操作时传递的字节数, 范围为 1-0xFFFFFFFF 字节。建议进行 DMA 操作时, 以整字 (4 个字节) 倍数长度进行数据传输。
- ◆ COMMAND[7:4]: DMA 读操作时在 PCI 总线上传送命令。按正文表 4-8-4 读 DMA 设置 PCI 命令, 设置其他 PCI 命令无效。
- ◆ PCI INTERRUPT[3]: 从桥模式下, 该寄存器配置为 1, 在 DMA 读操作正常完成时产生中断请求, DMA 读操作异常退出时不产生中断请求; 该寄存器配置为 0, 在 DMA 写操作正常完成时不发出中断请求。在主桥模式下, 该寄存器无效。
- ◆ STOP TRANSFER[1]: DMA 读传输停止位, 写 1 停止当前的 DMA 读传输, 写 0 无效。该寄存器写 1 后自动清 0, 即读出值为 0。
- ◆ START TRANSFER[0]: DMA 读操作起动位, 写 1 发起 DMA 读传输, 写 0 无效, DMA 传输完成后此位自动清 0, 因此可通过此位判断 DMA 读传输是否完成。

8. CPU 中断掩码寄存器 (CPU_IMASK)

Address = 0xC000-0040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留			PCI SYSTEM ERROR	保留				保留				保留	AHB-PCI WINDOW			READ DMA				保留	PCI-AHB WINDOW		WRITE DMA								
													DISCARD	FETCH ERROR	POST ERROR	AHB ERROR	PARITY ERROR	ABORT	END		DISCARD	FETCH ERROR	POST ERROR	AHB ERROR	PARITY ERROR	ABORT	END				
r/w			r/w	r/w				r/w				r/w	r/w	r/w	r/w	r/w	r	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	
000			0	0000				0000				0000			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- ◆ 保留位[31:30]: 在主桥模式下, 该寄存器可读可写; 在从桥模式下, 该寄存器只读, 读出值为 0。
- ◆ 保留位[29]: 在从桥模式下, 该寄存器可读可写; 在主桥模式下, 该寄存器只读, 读出值为 0。
- ◆ PCI SYSTEM ERROR[28], PCI 系统错误, 表示在 PCI 总线传输地址周期出现奇偶校验错, 1=中断使能, 0=中断屏蔽。在主桥模式下, 该寄存器可读可写。在从桥模式下, 该寄存器只读, 读出值为 0。
- ◆ PCI INTERRUPT[27:24]: PCI 中断, 外部 PCI 设备通过中断管脚 (pci_inta_n, pci_intb_n, pci_intc_n, pci_intd_n) 向处理器发起中断请求, 1=中断使能, 0=中断屏蔽。在主桥模式下, 该寄存器可读可写; 在从桥模式下, 该寄存器无效, 读出值为 0。
- ◆ PCI DOORBELL[19:16]: 外部 PCI 设备通过 PCI DOORBELL 位向处理器发起中断请求, 1=中断使能, 0=中断屏蔽。在从桥模式下, 该寄存器可读可写; 在主桥模式下, 该寄存器无效, 读出值为 0。
- ◆ AHB-PCI WINDOW DISCARD[14]: AHB-PCI 窗口丢弃, 表示处理器通过 AHB-PCI 窗口发起读操作, 超过 AHB-PCI 窗口定时寄存器 (AHBPCI_TIMER) 设置的时钟周期个数, 操作仍未完成, 此次操作被丢弃。1=中断使能, 0=中断屏蔽。
- ◆ AHB-PCI WINDOW FETCH ERROR[13]: AHB-PCI 窗口取值错误, 表示通过 AHB-PCI 窗口传输数据时, 从外部 PCI 设备读取数据时发生错误。1=中断使能, 0=中断屏蔽。

- ◆ AHB-PCI WINDOW POST ERROR[12]: AHB-PCI 窗口存储错误,表示通过 AHB-PCI 窗口传输数据时,数据写入外部 PCI 设备时发生错误。1=中断使能,0=中断屏蔽。
- ◆ READ DMA AHB ERROR[11]: DMA 读操作 AHB 错误,表示 DMA 控制器向 AHB 地址写入数据时发生错误。1=中断使能,0=中断屏蔽。
- ◆ READ DMA PARITY ERROR[10]: DMA 读操作数据奇偶错,表示 DMA 读传输过程中,从外部 PCI 总线设备读取数据时,发生了奇偶校验错。1=中断使能,0=中断屏蔽。
- ◆ READ DMA ABORT[9]: DMA 读操作异常终止,表示在 DMA 读传输过程中,PCI 总线上发生了目标设备终止或主设备终止。1=中断使能,0=中断屏蔽。
- ◆ READ DMA END[8]: DMA 读操作结束,表示 DMA 读操作正常完成或当前正在进行的 DMA 读操作被停止。1=中断使能,0=中断屏蔽。
- ◆ PCI-AHB WINDOW DISCARD[6]: PCI-AHB 窗口丢弃,表示外部 PCI 总线设备通过 PCI-AHB 窗口发起读操作,超过 PCI-AHB 窗口定时寄存器 (PCIAHB_TIMER) 设置的时钟周期个数,操作仍未完成,此次操作被丢弃。1=中断使能,0=中断屏蔽。
- ◆ PCI-AHB WINDOW FETCH ERROR[5]: PCI-AHB 窗口取值错误,表示通过 PCI-AHB 窗口传输数据时,从 AHB 地址空间读取数据时发生错误。1=中断使能,0=中断屏蔽。
- ◆ PCI-AHB WINDOW POST ERROR[4]: PCI-AHB 窗口存储错误,表示通过 PCI-AHB 窗口传输数据时,数据写入 AHB 地址空间时发生错误。1=中断使能,0=中断屏蔽。
- ◆ WRITE DMA AHB ERROR[3]: DMA 写操作 AHB 错误,表示 DMA 控制器从 AHB 地址空间读数据时发生错误。1=中断使能,0=中断屏蔽。
- ◆ WRITE DMA PARITY ERROR[2]: DMA 写操作数据奇偶错,表示 DMA 写传输过程中,向外部 PCI 总线设备写入数据时,发生了奇偶校验错。1=中断使能,0=中断屏蔽。
- ◆ WRITE DMA ABORT[1]: DMA 写操作异常终止,表示在 DMA 写传输过程中,PCI 总线上发生了目标设备终止或主设备终止。1=中断使能,0=中断屏蔽。
- ◆ WRITE DMA END[0]: DMA 写操作结束,表示 DMA 写操作正常结束或当前正在进行的 DMA 写操作被停止。1=中断使能,0=中断屏蔽。

9. CPU 中断状态寄存器 (CPU_ISTATUS)

Address = 0xC000-0044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留			PCI SYSTEM ERROR	PCI INTERRUPT				保留				PCI DOORBELL				保留	AHB-PCI WINDOW		READ DMA			PCI-AHB WINDOW				WRITE DMA					
																	DISCARD	FETCH ERROR	POST ERROR	AHB ERROR	PARITY ERROR									ABORT	END
																						保留							DISCARD		
r			r/w				r				r/w				r	r/w															
000			000000				0000				0000				0	0							0								

位号	助记符	描述
28	PCI SYSTEM ERROR	在主桥模式下，1表示PCI系统错误，在PCI总线传输地址周期出现奇偶错。写1清0，写0无效。在从桥模式下，该寄存器无效，读出值为0。
27:24	PCI INTERRUPT	在主桥模式下，1表示相应PCI总线设备发起的处理器中断有效。写1相应位清0，写0无效。在从桥模式下，该寄存器无效，读出值为0。
19:16	PCI DOORBELL	在从桥模式下，1表示PCI DOORBELL相应位有效。写1相应位清0，写0无效。在主桥模式下，该寄存器无效，读出值为0。
14	AHB-PCI WINDOW DISCARD	1表示AHB-PCI窗口丢弃。写1清0，写0无效。
13	AHB-PCI WINDOW FETCH ERROR	1表示AHB-PCI窗口取值错误。写1清0，写0无效
12	AHB-PCI WINDOW POST ERROR	1表示AHB-PCI窗口存储错误。写1清0，写0无效
11	READ DMA AHB ERROR	1表示DMA读操作AHB错误。写1清0，写0无效
10	READ DMA PARITY ERROR	1表示DMA读操作数据奇偶校验错误。写1清0，写0无效
9	READ DMA ABORT	1表示DMA读操作异常终止。写1清0，写0无效
8	READ DMA END	1表示DMA读操作结束。写1清0，写0无效
6	PCI-AHB WINDOW DISCARD	1表示PCI-AHB窗口丢弃。写1清0，写0无效
5	PCI-AHB WINDOW FETCH ERROR	1表示PCI-AHB窗口取值错误。写1清0，写0无效
4	PCI-AHB WINDOW POST ERROR	1表示PCI-AHB窗口存储错误。写1清0，写0无效
3	WRITE DMA AHB ERROR	1表示DMA写操作AHB错误。写1清0，写0无效
2	WRITE DMA PARITY ERROR	1表示DMA写操作数据奇偶校验错误。写1清0，写0无效
1	WRITE DMA ABORT	1表示DMA写异常终止。写1清0，写0无效
0	WRITE DMA END	1表示DMA写结束。写1清0，写0无效

10. CPU 中断命令寄存器 (CPU_ICMD)

Address = 0xC000-0048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								CPU DOORBELL				保留																			
r								w				r																			
0								0000				0																			

◆ CPU DOORBELL[23:20]：在从桥模式下，CPU 中断命令寄存器，用于人为发出一些 PCI 中断。写 1 使能，写 0 无效，读出值为 0。在主桥模式下，该寄存器无效。该寄存器与 PCI 中断掩码寄存器的[23:20]位一一对应，即 20 位对应 20 位、21 位对应 21 位、22 位对应 22 位、23 位对应 23 位。当配置 PCI 中断掩码寄存器[23:20]的某位为 1，CPU 向该寄存器的相应位写 1，则 BM3823A 通过管脚 pci_inta_n 向 PCI 总线发出中断请求。

11. CPU 状态及版本寄存器（CPU_VERSION）

Address = 0xC000-004C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
保留																TYPE				VERSION															
r																r				r															
0																xxxx				0x380															

位号	助记符	描述
15:12	TYPE	表示PCI接口模式 1 = 32位从桥模式 2 = 32位主桥模式 管脚pci_host接低电平，为2；接高电平，为1
11:0	VERSION	PCI桥版本, 由IC厂商定义

12. 不可预取范围 PCI-AHB 窗口范围控制寄存器（PCIAHB_ADDR_NP）

Address = 0xC000-0070

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															EN
r/w																							r								r/w
0x800000																							0								1

◆ ADDR[31:1]：PCI 设备通过 PCI-AHB 窗口访问处理器地址空间的基地址。在主桥

和从桥模式下，该寄存器功能不同。在主桥模式下，PCI-AHB 窗口采用扁平的寻址方式，PCI 总线读写操作的地址直接转化成内部 AHB 总线操作的地址。该寄存器定义了不可预取存储器空间，相当于定义了 PCI 基地址寄存器 3（BAR3），外部 PCI 设备通过访问该寄存器定义的地址空间，就实现了对处理器 AJ 地址空间的访问。在从桥模式下，PCI 设备通过 PCI-AHB 窗口访问处理器地址空间，地址需要进行变换。PCI 基地址寄存器 3（BAR3）由外部 PCI 主机系统进行配置，处理器通过该寄存器设置转换后的 AHB 地址，PCI-AHB 地址空间转换如下：AHB 地址 = PCIAHB_ADDR_NP + （PCI 操作地址 - PCI 基地址 3（BAR3））外部 PCI 设备访问 PCI 基地址空间 BAR3，就实现了对 AHB_ADDR_NP 定义的 AHB 地址空间的访问。

- ◆ EN[0]: PCI-AHB 地址空间使能位。1=使能 PCI 总线操作到 AHB 操作的转换，0=PCI 总线操作以目标异常终止（Target Abort）结束。

13. PCI-AHB 窗口可预取范围控制寄存器（PCIAHB_ADDR_PF）

Address = 0xC000-0074

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															EN
r/w																r															r/w
0x400000																0															1

- ◆ ADDR[31:1]: PCI 设备通过 PCI-AHB 窗口访问处理器地址空间的基地址。在主桥和从桥模式下，该寄存器功能不同。在主桥模式下，PCI-AHB 窗口采用扁平的寻址方式，PCI 总线读写操作的地址直接转化成内部 AHB 总线操作的地址。该寄存器定义了可预取存储器空间，相当于定义了 PCI 基地址寄存器 4（BAR4），外部 PCI 设备通过访问该寄存器定义的地址空间，就实现了对处理器 AHB 地址空间的访问。在从桥模式下，PCI 设备通过 PCI-AHB 窗口访问处理器地址空间，地址需要进行变换。PCI 基地址寄存器 4（BAR4）由外部 PCI 主机系统进行配置，处理器通过该寄存器设置转换后的 AHB 地址，PCI-AHB 地址空间转换如下：AHB 地址 = PCIAHB_ADDR_NP + （PCI 操作地址 - PCI 基地址 4（BAR4））外部 PCI 设备访问 PCI 基地址空间 BAR4，就实现了对 AHB_ADDR_PF 定义的 AHB 地址空间的访问。

- ◆ EN[0]: PCI-AHB 地址空间使能位。1=使能 PCI 总线操作到 AHB 操作的转换, 0=PCI 总线操作以目标异常终止 (Target Abort) 结束。

14. PCI-AHB 窗口丢弃定时寄存器 (PCIAHB_TIMER)

Address = 0xC000-0078

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
保留																				DISCARD TIMER																
r																				r/w																
0																				0x100																

位号	助记符	描述
11:0	DISCARD TIMER	清空交易定时位, 防止PCI总线锁死。PCI总线主设备通过PCI-AHB窗口发起读操作, 超过该寄存器定义的PCI时钟周期个数操作仍未完成, 此次操作丢弃。

15. AHB-PCI 窗口的丢弃定时寄存器 (AHBPCI_TIMER)

Address = 0xC000-007C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0					
保留																				DISCARD TIMER																
r																				r/w																
0																				0x100																

位号	助记符	描述
11:0	DISCARD TIMER	清空交易定时位, 防止PCI总线锁死。处理器通过通过AHB-PCI窗口读PCI总线上的设备, 超过该寄存器定义的系统时钟周期个数操作仍未完成, 此次操作丢弃。

16. PCI 控制寄存器

Address = 0xC000-0080

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

- ◆ [31:3]: 保留。

- ◆ MasterEN[2]: 主设备能力状态位, 1=具备主设备能力, 0=不具备。该位为 1, 表示处理器能够作为主设备对 PCI 总线上的目标设备发起读写操作。在主桥模式下, 该位一直为 1; 在从桥模式下, 该位复位后值为 0, 通过 PCI 状态和命令寄存器[2]位进行配置, PCI 状态和命令寄存器[2]位配置为 1, 则此位也为 1。
- ◆ MemoryEN[1]: 存储器地址空间使能位, 1=使能, 0=禁止。主桥模式下, 该位总置 1; 从桥模式下, 该位复位后为 0, 通过 PCI 状态和命令寄存器[1]位进行配置, PCI 状态和命令寄存器[1]位配置为 1, 则此位也为 1。
- ◆ Bridge PCI Ready[0]: 在主桥模式下, 系统复位后此位为 0, 225 个 PCI 时钟周期之后置为 1, 处理器进行 PCI 总线配置访问之前必须等此位为 1; 在从桥模式下, 该位复位后一直为 1。

17. PCI 设备和供应商标识寄存器 (PCI_DV)

Address = 0xC000-0084

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DV																															
r/w																															
0x55551556																															

位号	助记符	描述
31:0	DV	PCI 设备和供应商信息位

18. PCI 子系统设备和子系统供应商配置寄存器 (PCI_SUB)

Address = 0xC000-0088

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

位号	助记符	描述
31:16	SUBSYSTEM ID	PCI 子系统设备信息位
15:0	SUBSYSTEM VENDOR ID	PCI 子系统供应商信息位

19. PCI 分类代码和版本配置寄存器 (PCI_CREV)

Address = 0xC000-008C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLASS CODE																								REVISION ID							
r/w																								r/w							
0x040000																								0							

位号	助记符	描述
31: 8	CLASS CODE	PCI分类代码
7: 0	REVISION ID	版本信息位

20. PCI 仲裁状态寄存器 (PCI_BROKEN)

Address = 0xC000-0090

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

位号	助记符	描述
7:0	STATUS	在主桥模式下，该寄存器可读可写，表示PCI总线上Master设备的状态。1=表示相应的PCI Master设备被禁止，表明严重的系统错误。写1清0，读出值为0，写0无效。 在从桥模式下，该寄存器无效，只读，读出值为0。

21. PCI-AHB 窗口不可预取空间大小配置寄存器 (PCIAHB_SIZ_NP)

Address = 0xC000-0094

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIZ																															
r/w																															
0xFFFC00																								0							

位号	助记符	描述
31:0	SIZ	设置不可预取空间大小。低8位恒为0，最小范围为0x100字节。设定空间大小时，低位应设置成全0，高位设置成全1，如0的个数为n，则空间大小为2 ⁿ 字节。

22. PCI-AHB 窗口可预取空间大小配置寄存器 (PCIAHB_SIZ_PF)

- ◆ SINGALED SYSTEM ERROR[30]: 发出系统错误。当设备驱动 SERR 有效时, 将此位置 1。1=出现系统错误, 0=传输正常。该位写 1 清 0, 写 0 无效。
- ◆ RECEIVED MASTER ABORT[29]: 收到主设备异常终止。当总线主设备发起的交易以主设备异常终止结束, 将此位置 1。1=主设备异常终止, 0=传输正常该位写 1 清 0, 写 0 无效。
- ◆ RECEIVED TARGET ABORT[28]: 收到目标设备异常终止, 总线主设备发起总线传输被目标设备异常终止时, 总线主设备必须将此位置 1。1=收到目标异常终止, 0=传输正常。该位写 1 清 0, 写 0 无效。
- ◆ SINGALED TARGET ABORT[27]: 目标设备异常终止。目标设备异常终止一个 PCI 总线传输, 目标设备必须将此位置为 1。1=目标异常终止, 0=传输正常。该位写 1 清 0, 写 0 无效。
- ◆ DEVESEL TIMING[26:25]: 设备选择定时, 表示作为目标设备有效 DEVSEL 的速度。该寄存器只读, 读出值为 0x1, 表示中速。
- ◆ MASTER DATA PARITY ERROR[24]: 主设备奇偶校验错。1=总线出现奇偶校验错, 0=未出现奇偶校验错。该位写 1 后值为 0, 写 0 对值无影响。满足下面三个条件该寄存器置 1:
 - a) 桥作为总线主设备, 是 PCI 总线访问的发起者。
 - b) 命令寄存器 PCI_SCR bit[6] PARITY ERROR RESPONSE 设置为 1。
 - c) 通过自行设置错 PERR (在读操作的时候) 有效或者监测到目标设备设置 (在写的时候) PERR 有效。
- ◆ FAST BACK-TO-BACK ENABLE[23]: 目标设备应答背靠背传输的能力。1=具备, 0=不具备。
- ◆ [22:20]: 保留位。
- ◆ INTERRUPT STATUS[19]: 中断状态。1=PCI 中断信号有效, 0=PCI 中断信号无效。
- ◆ [18:11]: 保留位。
- ◆ INTERRUPT DISABLE[10]: 中断输出禁止位。1=禁止输出 PCI 中断信号, 0=允许 PCI 中断信号输出。
- ◆ [9]: 保留位。

- ◆ SERR ENABLE[8]: SERR 信号驱动使能。1=允许驱动 SERR 信号线, 0=不能驱动。
- ◆ [7]: 保留位。
- ◆ PARITY ERROR RESPONSE[6]: 奇偶校验错误响应。1=使能奇偶校验检查, 0=忽略奇偶错误。
- ◆ [5:3]: 保留位。
- ◆ BUS MASTER[2]: 总线主控能力。1=允许作为主设备操作 PCI 总线, 0=禁止作为主设备操作 PCI 总线。
- ◆ MEMEORY SPACE[1]: 存储器空间访问应答。1=允许访问, 0=禁止访问。
- ◆ [0]: 保留位。

25. PCI 分类代码和版本配置寄存器 (PCI_CREV)

Address = PCI 配置空间 0x08

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Class Code																								Revision ID							
r																								r							
0x040000																								0							

位号	助记符	描述
31: 8	CLASS CODE	PCI分类代码。该寄存器值由地址为0xC000_008C的寄存器设定
7: 0	REVISION ID	版本信息位。该寄存器值由地址为0xC000_008C的寄存器设定

26. PCI Latency Timer 寄存器 (PCI_LT)

Address = PCI 配置空间 0x0C

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

位号	助记符	描述
15:8	LATENCY TIMER	该寄存器定义了总线主设备一直保有总线控制权的PCI时钟周期个数, 。

27. PCI 基地址寄存器 0 (BAR0)

Address = PCI 配置空间 0x10

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

28. PCI 基地址寄存器 1 (BAR1)

Address = PCI 配置空间 0x14

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

29. PCI 基地址寄存器 2 (BAR2)

Address = PCI 配置空间 0x18

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

30. PCI 基地址寄存器 3 (BAR3)

Address = PCI 配置空间 0x1C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE ADDRESS																												PF	TYPE	MSI	
r/w																								r				r	r	r	
0																								0000				0	0	0	

- ◆ BASE ADDRESS[31:4]: 存储器基地址空间。外部 PCI 设备访问处理器地址空间的基地址，空间大小由寄存器 PCI-AHB 窗口不可预取范围屏蔽寄存器 (PCIAHB_SIZ_NP) 设定。
- ◆ PF[3]: 只读，读出值为 0，表明该地址空间不支持预存取。
- ◆ TYPE[2:1]: 只读，读出值为 00，表明支持 32 位寻址。

- ◆ MSI[0]: 只读, 读出值为 0, 表明地址空间为存储器空间。

31. PCI 基地址寄存器 4 (BAR4)

Address = PCI 配置空间 0x20

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE ADDRESS																												PF	TYPE	MSI	
r/w																								r				r	r	r	
0																								0000				1	0	0	

- ◆ BASE ADDRESS[31:4]: 存储器基地址空间。外部 PCI 设备访问处理器地址空间的基地址, 空间大小由寄存器 PCI-AHB 窗口可预取范围屏蔽寄存器(PCIAHB_SIZ_PF)设定。
- ◆ PF[3]: 只读, 读出值为 1, 表明该地址空间支持预存取。
- ◆ TYPE[2:1]: 只读, 读出值为 00, 表明支持 32 位寻址。
- ◆ MSI[0]: 只读, 读出值为 0, 表明地址空间为存储器空间。

32. PCI 基地址寄存器 5 (BAR5)

Address = PCI 配置空间 0x24

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE ADDRESS																												PF	TYPE	MSI	
r/w																								r				r	r	r	
0																								0000				0	0	0	

- ◆ BASE ADDRESS[31:4]: 配置寄存器基地址空间。外部 PCI 设备可通过该地址空间访问片内 PCI 侧寄存器, 空间大小为 256 字节。
- ◆ PF[3]: 只读, 读出值为 0, 表明该地址空间不支持预存取。
- ◆ TYPE[2:1]: 只读, 读出值为 00, 表明支持 32 位寻址。
- ◆ MSI[0]: 只读, 读出值为 0, 表明地址空间为存储器空间。

33. PCI Cardbus CIS Pointer 寄存器

Address = PCI 配置空间 0x28

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

34. PCI 子系统设备和子系统供应商配置寄存器 (PCI_SUB)

Address = PCI 配置空间 0x2C

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

位号	助记符	描述
31:16	SUBSYSTEM ID	PCI子系统设备信息位，该寄存器值由地址0xC000-0088的寄存器设定。
15:0	SUBSYSTEM VENDOR ID	PCI子系统供应商信息位，该寄存器值由地址0xC000-0088的寄存器设定。

35. PCI 扩展 ROM 基地址寄存器

Address = PCI 配置空间 0x30

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

36. PCI 能力指针寄存器

Address = PCI 配置空间 0x34

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																															
r																															
0																								0x78							

37. PCI 保留寄存器

Address = PCI 配置空间 0x38

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

38. PCI 中断线及中断引脚寄存器 (PCI_INTLP)

Address = PCI 配置空间 0x3C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																INTERRUPT PIN								INTERRUPT LINE							
r																r								r/w							
0																0x1								0							

位号	助记符	描述
15:8	INTERRUPT PIN	中断引脚寄存器。0x1表示设备使用pci_inta_n作为中断输出引脚
7:0	INTERRUPT LINE	中断线寄存器，中断信号pci_inta_n连接到主机的中断控制器哪个管脚上。

39. PCI 中断掩码寄存器 (PCI_IMASK)

Address = PCI 配置空间 0x80

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
保留								CPU DOORBELL				保留										READ DMA END	保留								WRITE DMA END		
r								r/w				r											r/w	r								r/w	
0								0				0											0	0								0	

位号	助记符	描述
23:20	CPU DOORBELL	在从桥模式下，CPU DOORBELL位中断使能位。1=中断使能，0=禁止中断。
8	READ DMA END	在从桥模式下，DMA读操作结束中断使能位。1=中断使能，0=中断禁止。
0	WRITE DMA END	在从桥模式下，DMA写操作结束使能位。1=中断使能，0=中断禁止。

40. PCI 中断状态寄存器 (PCI_ISTATUS)

Address = PCI 配置空间 0x84

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								CPU DOORBELL				保留								READ DMA END		保留								WRITE DMA END	
r								r/w				r								r/w		r								r/w	
0								0000				0								0		0								0	

位号	助记符	描述
23:20	CPU PC DOORBELL	在从桥模式下，1表示对应的CPU DOORBELL位有效。写1则相应位清0，写0无效。
8	READ DMA END	在从桥模式下，1表示DMA读操作结束。写1清0，写0无效。
0	WRITE DMA END	在从桥模式下，1表示DMA写操作结束。写1清0，写0无效。

41. PCI 中断命令寄存器（PCI_ICMD）

Address = PCI 配置空间 0x88

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留												PCI DOORBELL				保留															
r												w				r															
0												0000				0															

位号	助记符	描述
19: 16	PCI DOORBELL	在从桥模式下，PCI DOORBELL位。该字段只写，读出的值为0，写1使相应位有效，发出中断请求。

42. PCI 状态及版本信息寄存器（PCI_VERSION）

Address = PCI 配置空间 0x8C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留												TYPE				VERSION															
r												r				r															
0												xxxx				0x380															

位号	助记符	描述
15:12	TYPE	表示PCI接口模式 1 = 32位从桥模式 2 = 32位主桥模式 管脚pci_host接低电平，为2；接高电平，为1
11:0	VERSION	PCI桥版本，由IC厂商定义

3.9 内存控制器相关寄存器

1. 内部存储器配置寄存器 1 (IMCFG)

Address = 0x8000-0030

bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
WS		PDD				PDP	保留
0	0	0	0	0	0	1	0

bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
保留							
0	0	0	0	0	0	0	0

bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
保留					EWB		ERB
0	0	0	0	0	0	0	0

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
EEn	TCB						
0	x	x	x	x	x	x	x

Bit31:30 WS

内部同步 SRAM 读写访问等待周期数。初始值为 0。

(“00” =0, “01” =1, “10” =2, “11” =3)

Bit29:26 PDD

数据位低功耗分区开关。初始值为 0。“1”为低功耗打开，“0”为低功耗关闭。

Bit29 对应 0xA000-0000 - 0xA000-FFFF;

Bit28 对应 0xA001-0000 - 0xA001-FFFF;

Bit27 对应 0xA002-0000 - 0xA002-FFFF;

Bit26 对应 0xA003-0000 - 0xA003-FFFF。

Bit25 PDP

校验位低功耗分区开关。初始值为 1。“1”为低功耗打开，“0”为低功耗关闭。

Bit24:11 保留

保留。

Bit10:9 EWB

EDAC 写旁路使能，“11”为使能，“00”、“01”、“10”为不使能。初始值为“00”。

如果 EWB 使能，则在写校验位时，用 imcfg.tcb 来替换校验位。

Bit8 ERB

EDAC 读旁路使能，“1”为使能，“0”为不使能。初始值为 0。

如果 ERB 使能，则在读操作时，读入的校验位存入 imcfg.tcb。

Bit7 EEn

内部 SRAM 存储器 EDAC 使能，“1”为使能，“0”为不使能。初始值为 0。

Bit6:0 TCB

测试校验位。

如果 EWB 使能，则在写校验位时，用此字段来替换校验位。

如果 ERB 使能，则在读操作时，读入的校验位存入此字段。

3.10 外存控制器相关寄存器

1. 外部存储器配置寄存器 1

Address = 0x8100-0000

bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
FDiv			IOwdh		Brdy	Bexc	IOWS
0	0	0	x	x	0	0	1

bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
IOWS				IOen	SpEn	SpCkPo	PRWWS
1	1	1	1	0	0	0	1

bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8

PRWWS				PRWen	保留	PRwdh	
1	1	1	1	0	0	Gpio[1]	Gpio[0]

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
保留			PRRWS				
0	0	0	1	1	1	1	1

Bit31:29 FDiv

SpaceWire 输出时钟的分频系数，相比外部存储器控制器输入时钟。初始值为 0。

0: 8 分频

1: 同频

2: 2 分频

3: 3 分频

4: 4 分频

5: 5 分频

6: 6 分频

7: 7 分频

Bit28:27 IOwdh

I/O 设备总线宽度（“00”=8，“01”=16，“1x”=32）。

Bit26 Brdy

总线准备好（BRDYN）使能，用于 I/O 设备。“1”为使能，“0”为不使能。

Bit25 Bexc

总线错误（BEXCN）使能，用于 PROM、I/O 和 SRAM。“1”为使能，“0”为不使能。

Bit24:20 IOWS

I/O 设备访问等待周期数。初始值为 31。

（“00000”=0，“00001”=1，“00010”=2，...，“11111”=31）

Bit19 IOen

I/O 设备访问使能，“1”为使能，“0”为不使能。

Bit18 SpEn

SpaceWire 设备访问使能，“1”为使能，I/O 设备数据准备好信号使用 Spw_hack 管脚，“0”为不使能，I/O 设备数据准备好信号使用 brdyn 管脚。初始值为“0”。

Bit17 SpCkPo

SpaceWire 输出时钟的极性。“0”和“1”两种配置的输出时钟关系为反向关系。初始值为“0”。

Bit16:12 PRWWS

PROM 写等待周期数。初始值为 31。

(“00000”=0, “00001”=1, “00010”=2, ..., “11111”=31)

Bit11 PRWen

PROM 写使能，“1”为使能，“0”为不使能。

Bit10 保留

保留。

Bit9:8 PRwdh

PROM 数据宽度 (“00”=8, “01”=16, “1x”=32)。

此字段的复位初值由 GPIO 的第 1、0 两位配置。

Bit7:5 保留

保留。

Bit4:0 PRRWS

PROM 读等待周期数。初始值为 31。

(“00000”=0, “00001”=1, “00010”=2, ..., “11111”=31)

2. 外部存储器配置寄存器 2

Address = 0x8100-0004

bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
Sdref	Trp	Trfc			SdrCAS	Sdrbs	
0	1	1	1	1	1	0	0

bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
Sdrbs	Sdrcls		Sdrcmd		SrWWS		

0	1	0	0	0	1	1	1

bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
SrWWS	Sdren	Srdis	SrBS				保留
1	0	0	x	x	x	x	0

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
Brdy	RMW	Srwdh		SrRWS			
x	x	x	x	1	1	1	1

Bit31 Sdref

SDRAM 刷新，“1”为使能，“0”为不使能。

Bit30 Trp

SDRAM 预充时间。“1”，则 Trp 为 3 个时钟周期，“0”，则 TRP 为 2 个时钟周期。

Bit29:27 Trfc

SDRAM 刷新命令时间。Trfc 为（设置数值+3）个系统时钟周期。

Bit26 SdrCAS

SDRAM CAS 延时。“1”，则 CAS 延时为 3 个时钟周期，“0”，则 CAS 延时为 2 个时钟周期。

Bit25:23 Sdrbs

SDRAM BANK 的大小。（“000”=4Mbyte，“001”=8Mbyte，“010”=16Mbyte，…，“111”=512Mbyte）。

Bit22:21 Sdrcls

SDRAM Column 的大小。（“00”=256，“01”=512，“10”=1024，

当[22:21]=“11”并且[25:23]=“111”时，为 4096，

当[22:21]=“11”并且[25:23]≠“111”时，为 2048。）

Bit20:19 Sdrcmd

SDRAM 命令。当写入非 0 数值时，将产生 SDRAM 命令。

“01”=PRECHARGE，

“10” =AUTO-REFRESH,

“11” =LOAD-COMMAND-REGISTER,

命令执行后，此字段清零。

Bit18:15 SrWWS

SRAM 写等待周期数，初始值为 15。（“0000” =0, “0001” =1, “0010” =2, ..., “1111” =15）

Bit14 Sdren

SDRAM 使能。“1” 为使能，“0” 为不使能。

Bit13 Srdis

SRAM 不使能。“1” 为不使能，“0” 为使能。

Bit12:9 SrBS

SRAM bank 大小。定义每个 SRAM 大小。

（“0000” =8Kbyte, “0001” =16Kbyte, ..., “1111” =256Mbyte）

Bit8 保留

保留。

Bit7 Brdy

总线准备好（BRDYN）使能。使能 BRDYN 用于第 5 bank 的 SRAM。“1” 为使能，“0” 为不使能。

Bit6 RMW

“读改写” 方式使能，“1” 为使能，“0 为不使能。

Bit5:4 Srwdh

SRAM 数据总线宽度（“00” =8, “01” =16, “1x” =32）。

Bit3:0 SrRWS

SRAM 读等待周期数，初始值为 15。（“0000” =0, “0001” =1, “0010” =2, ..., “1111” =15）

3. 外部存储器配置寄存器 3

Address = 0x8100-0008

bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24

writen_iosn_r_ws				保留	SdrRV		
0	0	0	0	0	x	x	x

bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
SdrRV							
x	x	x	x	x	x	x	x

bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
SdrRV				iosn_writen_f_ws			
x	x	x	x	0	0	0	0

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
oen_iosn_r_ws				iosn_oen_f_ws			
0	0	0	0	0	0	0	0

Bit31:28 writen_iosn_r_ws

I/O 空间写访问，从 writen 上升沿到 iosn 上升沿等待周期数。

（“0000”=0，“0001”=1，“0010”=2，...，“1111”=15）。

Bit27 保留

保留。

Bit26:12 SdrRV

SDRAM 刷新计数器重载值。AUTO-REFRESH 命令间隔时间计算公式为：

$$t_{\text{REFRESH}} = ((\text{reload value}) + 1) / \text{SYSCLK}.$$

Bit11:8 iosn_writen_f_ws

I/O 空间写访问，从 iosn 下降沿到 writen 下降沿等待周期数。

（“0000”=0，“0001”=1，“0010”=2，...，“1111”=15）。

Bit7:4 oen_iosn_r_ws

I/O 空间读访问，从 oen 上升沿到 iosn 上升沿等待周期数。

（“0000”=0，“0001”=1，“0010”=2，...，“1111”=15）。

Bit3:0 iosn_oen_f_ws

I/O 空间读访问，从 iosn 下降沿到 oen 下降沿等待周期数。

（“0000”=0，“0001”=1，“0010”=2，...，“1111”=15）。

4. 外部存储器容错配置寄存器 1

Address = 0x8100-0100

bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
Sr1EEEn	Sr2EEEn	Sr3EEEn	Sr4EEEn	Sr5EEEn	romEEEn	SdEEEn	OED
0	0	0	0	0	Gpio[2]	0	0

bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
EWB		ERB	WP	tcb_addr_en	保留		
0	0	0	0	1	0	0	0

bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
保留						TCBarea	
0	0	0	0	0	0	1	1

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
TCB							
x	x	x	x	x	x	x	x

Bit31 Sr1EEEn

SRAM bank1 EDAC 使能，“1”为使能，“0”为不使能。

Bit30 Sr2EEEn

SRAM bank2 EDAC 使能，“1”为使能，“0”为不使能。

Bit29 Sr3EEEn

SRAM bank3 EDAC 使能，“1”为使能，“0”为不使能。

Bit28 Sr4EEEn

SRAM bank4 EDAC 使能，“1”为使能，“0”为不使能。

Bit27 Sr5EEEn

SRAM bank5 EDAC 使能，“1”为使能，“0”为不使能。

Bit26 romEEn

PROM EDAC 使能，“1”为使能，“0”为不使能。复位初值通过 GPIO[2]来配置。如果 GPIO[2]为‘1’，则 romEEn 配置位在上电复位时为‘1’。

Bit25 SdEEn

SDRAM EDAC 使能，“1”为使能，“0”为不使能。

Bit24 OED

Only Error Detection, 只检错不纠错使能，“1”为使能，“0”为不使能。如果 OED 使能，在 EDAC 使能时，遇到单位错，只检错不纠错。

Bit23:22 EWB

EDAC 写旁路使能，“11”为使能，“00”、“01”、“10”为不使能。如果 EWB 使能，则写校验位时，用 mecfg1.tcb 来替换校验位。（当 EWB 使能时，tcb_addr_en 使能，则用 tcb 字段写校验位时，需要数据地址等于 MECFG2 的 tcb_addr，如果 tcb_addr_en 不使能，则用 tcb 字段写校验位时，数据地址不限制。）

Bit21 ERB

EDAC 读旁路使能，“1”为使能，“0”为不使能。当 ERB 使能时，tcb_addr_en 使能，则读校验位时选择 tcb_addr 设置地址方式来存入 tcb 字段；tcb_addr_en 不使能，则读校验位时选择 tcb_area 设置区域方式来存入 tcb 字段。在 tcb_area 设置区域方式下，对于 PROM、第 5 个 SRAM、对应 TCBarea 字段的前 4 个 SRAM 和对应 TCBarea 字段的 SDRAM，在读操作时，读入的校验位存入 mecfg1.tcb。

Bit20 WP

Write Parity bit, 在写操作时，写校验位使能，“1”为使能，“0”为不使能。

Bit19 tcb_addr_en

Tcb address enable, 读旁路 erb 和写旁路 ewb 通过设置地址方式使能，“1”为使能，“0”为不使能。

当 ERB 使能时，tcb_addr_en 使能，则读校验位时选择 tcb_addr 设置地址方式来存入 tcb 字段，如果 tcb_addr_en 不使能，则读校验位时选择 tcb_area 设置区域方式来存入 tcb 字段。

当 EWB 使能时，tcb_addr_en 使能，则用 tcb 字段写校验位时，需要数据地址等于

MECFG2 的 tcb_addr，如果 tcb_addr_en 不使能，则用 tcb 字段写校验位时，数据地址不限制。

Bit18:10 保留

保留。

Bit9:8 TCBarrea

将每个 bank(第 5 个 bank 除外)的 SRAM 和每个 bank 的 SDRAM 平均分成 4 份区域。在通过设置此字段而指向的区域中，在读操作时，被访问的数据的校验位才可以存入 TCB 字段。PROM 和第 5 个 bank 的 SRAM 不受 TCBarrea 字段控制。

Bit7:0 TCB

测试校验位。

如果 MECFG1 中 ERB 使能，则在读操作时，读入的校验位存入此字段。(当 ERB 使能，如果 tcb_addr_en 不使能，那么对于 PROM、第 5 个 SRAM、对应 TCBarrea 字段的前 4 个 SRAM 和对应 TCBarrea 字段的 SDRAM，在读操作时，读入的校验位存入此字段。当 ERB 使能时，如果 tcb_addr_en 使能，则读校验位时选择 tcb_addr 设置地址方式来存入 tcb 字段。在 tcb_addr 方式下，在读操作时，当数据地址[31:2]与 MECFG2 的 tcb_addr[31:2]相同时，读入的校验位存入 MECFG1 的 tcb 字段。)

如果 MECFG1 中 EWB 使能，则写校验位时，用此字段来替换校验位。(当 EWB 使能时，如果 tcb_addr_en 使能，则用 tcb 字段写校验位时，需要数据地址等于 MECFG2 的 tcb_addr，如果 tcb_addr_en 不使能，则用 tcb 字段写校验位时，数据地址不限制。)

5. 外部存储器容错配置寄存器 2

Address = 0x8100-0104

bit 31	bit 30	bit 29	bit 28	bit 27	bit 26	bit 25	bit 24
tcb_addr							
0	0	0	0	0	0	0	0

bit 23	bit 22	bit 21	bit 20	bit 19	bit 18	bit 17	bit 16
tcb_addr							

0	0	0	0	0	0	0	0

bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
tcb_addr							
0	0	0	0	0	0	0	0

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
tcb_addr						保留	
0	0	0	0	0	0	0	0

Bit31:2 tcb_addr

设置 ERB 时被读校验位的数据所在地址（条件：ERB 使能，且 tcb_addr_en 使能）。

设置 EWB 时被写校验位的数据所在地址（条件：EWB 使能，且 tcb_addr_en 使能）。

Bit1:0 保留

保留。

3.11 GPIO 相关寄存器

1. GPIO 输入输出数据寄存器（0x8100-0400）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IODATA1																															
r/w																															
xxxxxxx																															

位号	助记符	描述
31: 0	IODATA1	GPIO[31:0]输入或输出数据，可读可写，对应 IO 配置为输出时，执行写操作，将所写数据作为 GPIO[31:0]输出，对应 IO 配置为输入时，执行读操作，读出从 GPIO[31:0]输入的数据。上电复位后，值不定。

2. GPIO 输入输出方向控制寄存器（0x8100-0408）



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IODIR1																															
r/w																															
0																															

位号	助记符	描述
31: 0	IODIR1	GPIO[31:0]方向控制，1=输出，0=输入，可读可写，上电复位后，值为全 0，即配置 32 路接口为输入。

3. GPIO 输入滤波分频与中断模式寄存器（0x8100-0410）

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

位号	助记符	描述
24	M	GPIO 中断模式配置，0=数据模式，1=IO 模式，上电复位后，值为 0。
23:0	PRESCALE	GPIO 输入滤波分频配置，可将系统时钟 1/（PRESCALE+1）分频（其中 PRESCALE 的值需换算为十进制，最大分频值为 2^{24} ），上电复位后，值为全 0。

4. GPIO 输入滤波时钟使能寄存器（0x8100-0414）

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

位号	助记符	描述
3:0	EN	8 路分频时钟使能信号，0=禁止，1=使能，每位对应一路，上电复位后，值为全 0，即禁止所有分频时钟输出。

5. GPIO 中断寄存器（0x8100-0418）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	LE	PL	ISE3					EN	LE	PL	ISE2					EN	LE	PL	ISE1					EN	LE	PL	ISE0				
r/w	r/w	r/w	r/w					r/w	r/w	r/w	r/w					r/w	r/w	r/w	r/w					r/w	r/w	r/w	r/w				
0	x	x	x					0	x	x	x					0	x	x	x					0	x	x	x				

位号	助记符	描述
31	EN	GPIO 中断 3 使能信号，1=使能，0=禁止，上电复位后，值为 0；
30	LE	GPIO 中断 3 触发方式，[30:29]：00=低电平触发，01=高电平触发，10=下降沿触发，11=上升沿触发，上电复位后，初值不定；
29	PL	
28:24	ISEL3	GPIO 中断 3 对应输入接口选择，5 位译码对应低 32 个输入接口，上电复位后，初值不定；
23	EN	GPIO 中断 2 使能信号，1=使能，0=禁止，上电复位后，值为 0；
22	LE	GPIO 中断 2 触发方式，[22:21]：00=低电平触发，01=高电平触发，10=下降沿触发，11=上升沿触发，上电复位后，初值不定；
21	PL	
20:16	ISEL2	GPIO 中断 2 对应输入接口选择，5 位译码对应低 32 个输入接口，上电复位后，初值不定；
15	EN	GPIO 中断 1 使能信号，1=使能，0=禁止，上电复位后，值为 0；
14	LE	GPIO 中断 1 触发方式，[14:13]：00=低电平触发，01=高电平触发，10=下降沿触发，11=上升沿触发，上电复位后，初值不定；
13	PL	
12:8	ISEL1	GPIO 中断 1 对应输入接口选择，5 位译码对应低 32 个输入接口，上电复位后，初值不定；
7	EN	GPIO 中断 0 使能信号，1=使能，0=禁止，上电复位后，值为 0；

6	LE	GPIO 中断 0 触发方式，[6:5]：00=低电平触发，01=高电平触发，10=下降沿触发，11=上升沿触发，上电复位后，初值不定；
5	PL	
4:0	ISEL0	GPIO 中断 0 对应输入接口选择，5 位译码对应低 32 个输入接口，上电复位后，初值不定。

6. 0 路数据寄存器 0 (0x8100-0420)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA2								DATA1								DATA0								COUNTER							
r								r								r								r							
0								0								0								0							

位号	助记符	描述
31:24	DATA2	0 路缓存数据 DATA2，只读，上电复位后，值为全 0；
23:16	DATA1	0 路缓存数据 DATA1，只读，上电复位后，值为全 0；
15:8	DATA0	0 路缓存数据 DATA0，只读，上电复位后，值为全 0；
7:0	COUNTER	0 路缓存数据计数器，只读，上电复位后，值为全 0，执行读操作后自动清零，计数值达到 0xff 时代表已经计满或者溢出。

7. 0 路数据寄存器 1 (0x8100-0424)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA6								DATA5								DATA4								DATA3							
r								r								r								r							
0								0								0								0							

位号	助记符	描述
31:24	DATA6	0 路缓存数据 DATA6，只读，上电复位后，值为全 0；
23:16	DATA5	0 路缓存数据 DATA5，只读，上电复位后，值为全 0；

15:8	DATA4	0 路缓存数据 DATA4，只读，上电复位后，值为全 0；
7:0	DATA3	0 路缓存数据 DATA3，只读，上电复位后，值为全 0。

8. 1 路数据寄存器 0 (0x8100-0428)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA2								DATA1								DATA0								COUNTER							
r								r								r								r							
0								0								0								0							

位号	助记符	描述
31:24	DATA2	1 路缓存数据 DATA2，只读，上电复位后，值为全 0；
23:16	DATA1	1 路缓存数据 DATA1，只读，上电复位后，值为全 0；
15:8	DATA0	1 路缓存数据 DATA0，只读，上电复位后，值为全 0；
7:0	COUNTER	1 路缓存数据计数器，只读，上电复位后，值为全 0，执行读操作后自动清零，计数值达到 0xff 时代表已经计满或者溢出。

9. 1 路数据寄存器 1 (0x8100-042C)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA6								DATA5								DATA4								DATA3							
r								r								r								r							
0								0								0								0							

位号	助记符	描述
31:24	DATA6	1 路缓存数据 DATA6，只读，上电复位后，值为全 0；
23:16	DATA5	1 路缓存数据 DATA5，只读，上电复位后，值为全 0；
15:8	DATA4	1 路缓存数据 DATA4，只读，上电复位后，值为全 0；
7:0	DATA3	1 路缓存数据 DATA3，只读，上电复位后，值为全 0。

10. 2 路数据寄存器 0 (0x8100-0430)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA2								DATA1								DATA0								COUNTER							
r								r								r								r							
0								0								0								0							

位号	助记符	描述
31:24	DATA2	2 路缓存数据 DATA2，只读，上电复位后，值为全 0；
23:16	DATA1	2 路缓存数据 DATA1，只读，上电复位后，值为全 0；
15:8	DATA0	2 路缓存数据 DATA0，只读，上电复位后，值为全 0；
7:0	COUNTER	2 路缓存数据计数器，只读，上电复位后，值为全 0，执行读操作后自动清零，计数值达到 0xff 时代表已经计满或者溢出。

11. 2 路数据寄存器 1 (0x8100-0434)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA6								DATA5								DATA4								DATA3							
r								r								r								r							
0								0								0								0							

位号	助记符	描述
31:24	DATA6	2 路缓存数据 DATA6，只读，上电复位后，值为全 0；
23:16	DATA5	2 路缓存数据 DATA5，只读，上电复位后，值为全 0；
15:8	DATA4	2 路缓存数据 DATA4，只读，上电复位后，值为全 0；
7:0	DATA3	2 路缓存数据 DATA3，只读，上电复位后，值为全 0。

12. 3 路数据寄存器 0 (0x8100-0438)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA2								DATA1								DATA0								COUNTER							
r								r								r								r							
0								0								0								0							

位号	助记符	描述
31:24	DATA2	3 路缓存数据 DATA2，只读，上电复位后，值为全 0；
23:16	DATA1	3 路缓存数据 DATA1，只读，上电复位后，值为全 0；
15:8	DATA0	3 路缓存数据 DATA0，只读，上电复位后，值为全 0；
7:0	COUNTER	3 路缓存数据计数器，只读，上电复位后，值为全 0，执行读操作后自动清零，计数值达到 0xff 时代表已经计满或者溢出。

13. 3 路数据寄存器 1（0x8100-043C）

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA6								DATA5								DATA4								DATA3							
r								r								r								r							
0								0								0								0							

位号	助记符	描述
31:24	DATA6	3 路缓存数据 DATA6，只读，上电复位后，值为全 0；
23:16	DATA5	3 路缓存数据 DATA5，只读，上电复位后，值为全 0；
15:8	DATA4	3 路缓存数据 DATA4，只读，上电复位后，值为全 0；
7:0	DATA3	3 路缓存数据 DATA3，只读，上电复位后，值为全 0。

3.12 SPI 相关寄存器

1. SPI0 控制寄存器、SPI1 控制寄存器

Address = 0x8100-0560、Address = 0x8100-0580

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																											BIDIROE	SPC0	CPOL	CPHA	LSBEF
r																											r/w	r/w	r/w	r/w	r/w
0																											1	0	0	0	1

- ◆ [31:5]: 保留位，写无效，读出为 0；
- ◆ BIDIROE[4]: 输出使能位，主模式控制 SPI_MOSI，从模式控制 SPI_MISO。为 0 禁止输出使能，1 使能输出。复位为 1；
- ◆ SPC0[3]: 串口工作方式控制位。为 0 时串口为单向数据传输（双线模式），为 1 时串口为双向 I/O 数据传输（单线模式）。复位为 0；
- ◆ CPOL[2]: SPI 时钟极性选择位。0 表示时钟高有效，空闲模式时 SPI_CLK 为低；为 1 时表示时钟低有效，空闲模式 SPI_CLK 为高。复位为 0；
- ◆ CPHA[1]: SPI 时钟相位选择位。0 表示数据采样发生在 SPI_CLK 的奇数边沿（时钟有效时第一个边沿为 1，依次往后数），为 1 表示数据采样发生在 SPI_CLK 偶数边沿。复位为 0；
- ◆ LSBEF[0]: LSB-first 使能位。为 0 时，数据传输低位优先，为 1 时数据传输高位优先。数据寄存器的 MSB 为 Bit7，复位为 1。

2. SPI0 状态寄存器、SPI1 状态寄存器

Address = 0x8100-0564、Address = 0x8100-0584

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

- ◆ [31:8]: 保留位，写无效，读出为 0；
- ◆ SPIF[7]: 数据中断标志。0 表示数据传输未完成；1 表示数据传输完成，即接收数

据寄存器收到新数据。数据寄存器读、写后该位清 0。复位为 0；

- ◆ [6]: 保留位，写无效，读出为 0；
- ◆ SPTEF[5]: 数据寄存器清空中断标志。0 表示数据寄存器非空；1 表示数据寄存器为空。复位为 1；
- ◆ [4:0]: 保留位，写无效，读出为 0。

3. SPI0 数据寄存器、SPI1 数据寄存器

Address = 0x8100-0568、Address = 0x8100-0588

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

- ◆ DATA[31:0]: SPI 数据，可读写。可读数据位长与数据位长寄存器配合使用，如果数据位长为 32 位，则可读数据位长为 32 位（数据位长最小为 8 位，最大为 32 位）。复位为 0。

4. SPI0_CLK 相对主钟分频寄存器、SPI1_CLK 相对主钟分频寄存器

Address = 0x8100-056C、Address = 0x8100-058C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																						FREQ									
r																						r/w									
0																						10'h006									

- ◆ [31:10]: 保留位，写无效，读出为 0；
- ◆ FREQ[9:0]: SPI_CLK 相对主时钟的分频值（分频值 $N=[0x2:0x3ff]$ ）。分频值为 N 时，SPI_CLK 为主时钟的 N 分频；复位为 6（10'h006）。

5. SPI0 数据位长寄存器、SPI1 数据位长寄存器

Address = 0x8100-0570、Address = 0x8100-0590

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																										LENGTH					
r																										r/w					
0																										6'h08					

◆ [31:6]: 保留位，写无效，读出为 0；

◆ LENGTH[5:0]: SPI 数据长度（最大为 32，最小为 8），可读写，复位为 8（6'h08）。

6. SPI0 片选寄存器、SPI1 片选寄存器

Address = 0x8100-0574、Address = 0x8100-0594

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																										CSN					
r																										r/w					
0																										2'b11					

◆ [31:2]: 保留位，写无效，读出为 0；

◆ CSN[1:0]: SPI 片选位，低电平有效。复位为 3（2'b11）。

3.13 I²C 相关寄存器

1. I²C0 中断标志寄存器、I²C1 中断标志寄存器

Address = 0x8100-0800、Address = 0x8100-0900

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

◆ [31:6]: 保留位。写无效，读出为 0；

◆ SLAVE_INT[5]: 从模式地址中断，从机模式情况下，被其他主机地址校验成功时，该位被设置为 1，读数据寄存器时，该位被清零；

- ◆ **BYTE_INT[4]**: byte 中断, 该位表示, I²C 总线在基本模式下已经发送或者接收完成一组 8 位数据。在存储器模式情况下, 该位不起作用。在其它情况下, 该位被置为 0, 该位在读写访问数据寄存器时被置 0;
- ◆ **SEQ_INT[3]**: 序列中断, 该位表示, 存储器模式情况下, 一组序列传输完成后, 该位被置 1, 该位在基本模式情况下不起作用, 该位在读中断标志寄存器时被置 0;
- ◆ **ARBLOST_INT[2]**: 仲裁丢失中断, 该位表示在器件丢失仲裁时, 该位被置 1, 该位在读中断标志寄存器时被置 0;
- ◆ **NOACK_INT[1]**: 未响应中断, 该位表示, 当 I²C 总线在完成一个 byte 数据传输后未收到响应时, 该位被置 1, 该位在读中断标志寄存器时被置 0;
- ◆ **INT[0]**: 中断请求, 该位表示中断标志寄存器中所有非屏蔽位的逻辑相或, 由此位送至外部中断输出。

2. I²C0 控制寄存器、I²C1 控制寄存器

Address = 0x8100-0804、Address = 0x8100-0904

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																									GCA_EN	ACK_EN	STOP	START	LSA_DDR		
r																									r/w	r/w	r/w	r/w	r/w		
0																									0	0	0	0	0		

- ◆ **[31:5]**: 保留位。写无效, 读出为 0;
- ◆ **GCA_EN[4]**: 广播呼叫响应使能, 若 GCA_EN=1, 产生对广播呼叫地址的响应信号 (器件接收广播呼叫地址);
- ◆ **ACK_EN[3]**: 响应使能, 当 ACK_EN=1 时, 对接收数据产生一个响应信号, 该位必须要在进入 “产生响应 (generate ack)” 状态之前进行设置;
- ◆ **STOP[2]**: 停止请求, 当 STOP=1 且该模块作为主机时, I²C 总线将在 byte 传输完成后的最近一个周期产生 STOP 条件, 同时该位将在 STOP 条件产生之后被自动置 0;
- ◆ **START[1]**: 开始请求, 当 START=1 时, 若该模块作为主机, I²C 总线将在 byte 传输完成后的最近一个周期产生 START 条件; 若该模块作为从机或者进入存储模式时, I²C 总线将在 “等开始 (wait for start)” 状态使能之后, 才会产生 START 条件; 该位将在 START 条件产生之后被自动置 0;

- ◆ LSADDR[0]: 长从机地址, 当 LSADDR=1 时, 其自身从机地址为 10 位; 当 LSADDR=0 时, 其自身的从地址为 7 位。

3. I²C0 速率寄存器、I²C1 速率寄存器

Address = 0x8100-0808、Address = 0x8100-0908

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																								HSC LK2	HSC LK1	HSC LK0	FSC LK4	FSC LK3	FSC LK2	FSC LK1	FSC LK0
r																								r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
0																								0	0	0	0	0	0	0	0

- ◆ [31:8]: 保留位。写无效, 读出为 0;
- ◆ HSCLK2[7]: 可读写, HS 模式下 SCL 分频比例因子的 bit2;
- ◆ HSCLK1[6]: 可读写, HS 模式下 SCL 分频比例因子的 bit1;
- ◆ HSCLK0[5]: 可读写, HS 模式下 SCL 分频比例因子的 bit0;
- ◆ FSCLK4[4]: 可读写, F/S 模式下 SCL 分频比例因子的 bit4;
- ◆ FSCLK3[3]: 可读写, F/S 模式下 SCL 分频比例因子的 bit3;
- ◆ FSCLK2[2]: 可读写, F/S 模式下 SCL 分频比例因子的 bit2;
- ◆ FSCLK1[1]: 可读写, F/S 模式下 SCL 分频比例因子的 bit1;
- ◆ FSCLK0[0]: 可读写, F/S 模式下 SCL 分频比例因子的 bit0。

SPEED 寄存器中的数据用以配置时钟分频比例。F/S 工作模式下, 若[4:0]为全 1, 则工作频率取决于 FREQ 寄存器, 其他情况则根据该五位值决定 I²C 的主机模式工作频率, 如表 A3-13-1 所示; HS 工作模式下, 若第一串字符发送了 HS 模式识别码 (8'b000001xxx, 发送时仍在 F/S 模式, 发送完成后进入 HS 模式), 若[7:5]三位为全 1, 则其工作频率取决于 FREQ 寄存器, 其他情况下根据该三位的值决定 I²C 在主机 HS 模式下的工作频率, 如表 A3-13-2 所示。

表 A3-13-1 F/S 工作模式下主机模式工作频率表

SPEED 寄存器低五位	分频比例
00000	66
00001	130
00010	194
00011	258

00100	322
00101	386
...	...
...	$66+64 \times n$
...	...
11100	1858
11101	1922
11110	1986
11111	取决于 FREQ

表 A3-13-2 HS 工作模式下主机模式工作频率表

SPEED 寄存器高三位	分频比例
000	6
001	9
010	12
011	15
100	18
101	21
110	24
111	取决于 FREQ

I²C 的工作频率可以通过 SPEED 进行配置(F/S 模式下需 SPEED 寄存器第[4:0]不为全 1, HS 模式下需 SPEED 寄存器第[7:5]三位不为全 1)。

①F/S 模式:

假设主时钟频率 Fclk, I²C 工作频率为 Fi2c, 则可以通过计算公式得到两者关系:

$$Fi2c = Fclk / (66+64 \times SPEED[4:0])$$

如 100MHz 时钟, SPEED[4:0]被置为 0xe (962 分频), I²C 的时钟频率为 104KHz。

②HS 模式:

若第一串字符发送的是高速模式识别码 (发送时, SCL 时钟频率取决于此前设定的 SPEED 或 FREQ 寄存器), 则在第二个发送的数据时的 SCL 速度即基于 SPEED 寄存器的[7:5]位 (此三位不为全 1), 根据计算公式可以得到高速模式下主时钟频率 Fclk, I²C 工作频率为 Fi2c 之间的关系:

$$Fi2c = Fclk / (6 + 3 * SPEED[7:5])$$

如 30MHz 时钟，SPEED[7:5]被置为 0x1（9 分频），I²C 的时钟频率为 3.333MHz。

4. I²C0 数据寄存器、I²C1 数据寄存器

Address = 0x8100-080C、Address = 0x8100-090C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																								DATA							
r																								r/w							
0																								00000000							

- ◆ [31:8]: 保留位。写无效，读出为 0；
- ◆ DATA[7:0]: bit7 对应串行数据 MSB bit7，bit0 对应串行数据 LSB bit0；该寄存器包含即将发送或者刚刚接收的一个 byte 数据，该寄存器可读/写，MSB 最先发出，LSB 最后发出。

5. I²C0 从地址寄存器 1、I²C1 从地址寄存器 1

Address = 0x8100-0810、Address = 0x8100-0910

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
保留																															ADR9	ADR8
r																															r/w	r/w
0																															0	0

- ◆ [31:2]: 保留位。写无效，读出为 0；
- ◆ ADR9[1]: 10 位地址模式的地址 bit9，MSB；
- ◆ ADR8[0]: 10 位地址模式的地址 bit8。

6. I²C0 从地址寄存器 0、I²C1 从地址寄存器 1

Address = 0x8100-0814、Address = 0x8100-0914

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
保留																								ADR7	ADR6	ADR5~1				ADR0			
r																								r/w	r/w	r/w				r/w			
0																								0	0	00000				0			

- ◆ [31:8]: 保留位。写无效，读出为 0；
- ◆ ADR7[7]: 10 位地址模式的地址 bit7；
- ◆ ADR6[6]: 7 位地址模式的地址 bit6，MSB；也是 10 位地址模式的地址 bit6；



- ◆ ADR5~ADR1[5:1]: 地址 bit5-bit1;
- ◆ ADR0[0]: 地址 bit0, LSB。

7. I²C0 状态寄存器、I²C1 状态寄存器

Address = 0x8100-0818、Address = 0x8100-0918

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																											MODE	STATE			
r																											r	r			
0																											00	000			

- ◆ [31:5]: 保留位。写无效，读出为 0;
- ◆ MODE[4:3]: I²C 总线控制器的工作状态，复位后为 ‘00’;
 - ‘00’: 从机模式
 - ‘01’: 主机模式
 - ‘10’: 存储模式
 - ‘11’: 在 bit2-bit0= ‘111’ 时，代表产生了停止条件
- ◆ STATE[2:0]: 与 I²C 总线控制器的工作模式相关，三位组合成八种状态，从 ‘000’ 到 ‘111’，具体状态描述参见以表 A3-13-3，复位后为 ‘000’。

状态寄存器分多个工作模式：从机工作模式、主机工作模式和存储器模式。三个工作模式下，状态寄存器的各个位有不同的意义。下面分别就三个工作模式下的状态寄存器进行说明。状态寄存器为只读存储器，bit31-bit5 为保留位，bit4-bit0 代表工作模式和工作状态，通过查询和判断 bit4-bit0 的数据，可以对比状态表 A3-13-3，获知当前的工作状态。

表 A3-13-3 状态寄存器读出数据与状态对照表

[4:3]	[2:0]	状态	描述
从机模式			
00	000	wait for start	等待 START 条件
	001	receive address 0	接收第一个地址 byte
	010	slave address ack	作为从机，响应接收到的地址
	011	receive address 1	接收第二个地址的 byte
	100	slave receive data	作为从机，接收数据
	101	slave generate ack	作为从机，响应接收到的数据



[4:3]	[2:0]	状态	描述
	110	slave send data	作为从机，发送数据
	111	wait for condition	等待 START 或 STOP 条件
主机模式			
01	000	generate START 0	START 条件产生
	001	send master code	高速模式下发送模式识别码
	010	generate START 1	产生 re-START 条件
	011	send address	基本模式下发送从机地址
	100	master receive data	作为主机接收数据
	101	master generate ack	作为主机响应数据
	110	master send data	作为主机发送数据
	111	wait for request	作为主机等待 START 或 STOP 状态请求
存储器模式			
10	000	mem device sel1	存储器模式下，发送从器件地址
	001	mem sends maddr	发送存储器内部地址
	010	mem gen START	存储器模式下，产生 re-START 条件
	011	mem device sel2	再次发送器件地址
	100	mem receive data	存储器模式下，接收数据
	101	mem generate ack	存储器模式下，响应数据
	110	mem send data	存储器模式下，发送数据
	111	reserved	保留状态
其他			
11	111	generate STOP	产生了停止条件
	xxx	reserved	保留状态

8. I²C0 中断屏蔽寄存器、I²C1 中断屏蔽寄存器

Address = 0x8100-081C、Address = 0x8100-091C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																										SLAVE_MASK	BYTE_MASK	SEQ_MASK	ARBLOS_T_MASK	NOACK_MASK	INT_MASK
r																										r/w	r/w	r/w	r/w	r/w	r/w
0																										0	0	0	0	0	0

◆ [31:6]R: 保留位。写无效，读出为 0;

◆ SLAVE_MASK[5]: 从地址中断屏蔽，当 SLAVE_MASK=1 时，党自身被寻址后不

产生中断信号，上电初值为 0；

- ◆ BYTE_MASK[4]: byte 中断屏蔽，BYTE_MASK=1 时，当发送或接收完一 byte 数据后，不产生中断，上电初值为 0；
- ◆ SEQ_MASK[3]: 序列中断屏蔽，SEQ_MASK=1 时，当接收完一个序列的数据后，不产生中断，上电初值为 0；
- ◆ ARB_MASK[2]: 仲裁丢失屏蔽，ARB_MASK=1 时，当仲裁丢失时不产生中断，上电初值为 0；
- ◆ NOACK_MASK[1]: 错误条件屏蔽，NOACK_MASK=1 时，当不响应情况发生时不产生中断，上电初值为 0；
- ◆ INT_MASK[0]: 中断请求屏蔽，INT_MASK =1 时，所有中断均不产生，上电初值为 0。

9. I²C0 存储器件地址寄存器、I²C1 存储器件寄存器

Address = 0x8100-0820、Address = 0x8100-0920

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																								DEV_SEL							
r																								r/w							
0																								0000000							

- ◆ [31:7]: 保留位。写无效，读出为 0；
- ◆ DEV_SEL6-0[6:0]: 存储器件的地址，bit6-bit0，bit6 作为 MSB，bit0 为 LSB，可读写，上电初值为全 0，仅在存储器模式下可用，最高位先发送，最低位最后发送。

10. I²C0 存储地址寄存器 1、I²C1 存储地址寄存器 1

Address = 0x8100-0824、Address = 0x8100-0924

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																								MADDR7~MADDR0							
r																								r/w							
0																								00000000							

- ◆ [31:8]: 保留位。写无效，读出为 0；
- ◆ MADDR7-MADDR0[7:0]: bit7 对应存储单元地址的 MSB bit7，bit0 对应存储单元地址的 LSB bit0，可读写，上电初值为全 0，仅在存储模式下可读写的特殊功能寄存

器，最先传输 MSB，最后传输 LSB。

11. I²C0 存储地址寄存器 2、I²C1 存储地址寄存器 2

Address = 0x8100-0828、Address = 0x8100-0928

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																								MADDR15~MADDR8							
r																								r/w							
0																								00000000							

- ◆ [31:8]: 保留位。写无效，读出为 0;
- ◆ MADDR15-MADDR8[7:0]: bit7 对应存储单元地址的 MSB bit15, bit0 对应存储单元地址的 LSB bit8，可读写，上电初值为全 0，仅在存储模式下可读写的特殊功能寄存器，最先传输 MSB，最后传输 LSB。

12. I²C0 存储模式选择寄存器、I²C1 存储模式选择寄存器

Address = 0x8100-082C、Address = 0x8100-092C

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

- ◆ [31:4]: 保留位。写无效，读出为 0;
- ◆ MEM_EN[3]: 存储器支持模式使能，可读写，当 MEM_EN=1 时，存储器支持模式开启，MEM_EN=0 时，工作在基本模式（主模式或者从模式），可读写，上电初值为 0;
- ◆ DATASIZE[2]: 数据区大小，当 DATASIZE=1 时，数据区为 16 bytes，DATASIZE=0 时，数据区为 1 byte，仅在存储模式下可读写，上电初值为 0;
- ◆ ADDRSIZE[1]: 地址区大小，当 ADDRSIZE=1 时，地址区为 2 bytes，ADDRSIZE=0 时，地址区为 1 byte，仅在存储模式下可读写，上电初值为 0;
- ◆ DIR[0]: 传输方向，当 DIR=1 时，I²C 总线控制器读取寄存器，DIR=0 时，I²C 总线控制器写寄存器，仅在存储模式下可读写，上电初值为 0。

13. I²C0 频率寄存器、I²C1 频率寄存器

Address = 0x8100-0830、Address = 0x8100-0930

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																								FREQ							
r																								r/w							
0																								11111111							

- ◆ [31:8]: 保留位。写无效，读出为 0；
- ◆ FREQ[7:0]: 用于主模式下的 SCL 时钟，可读写，初值为 0xff，F/S 模式下，若 SPEED 寄存器的低 5 位为全 1，则 SCL 的频率基于 FREQ 的分频值；H/S 模式下，若 SPEED 寄存器的第 7-5 位为全 1，则 SCL 的频率基于 FREQ 的分频值，如表 A3-13-4 所示。
该寄存器不能为全 0，全 0 时因周期与系统主时钟频率接近，无法进行 I²C 通信。

表 A3-13-4 主机模式关于 FREQ 寄存器的工作频率表

FREQ	分频比例
0x00	4
0x01	12
0x02	20
0x03	28
0x04	36
...	...
...	4+8*n
...	...
0xfe	2036
0xff	2044

I²C 的工作频率可以通过 FREQ 进行配置（F/S 模式下需 SPEED 寄存器[4:0]为全 1，HS 模式下需 SPEED 寄存器[7:5]三位为全 1，假设主时钟频率 Fclk，I²C 工作频率为 Fi2c，则可以通过计算公式得到两者关系：

$$Fi2c = Fclk / (4 + 8 * FREQ),$$

如 30MHz 时钟，FREQ 被置为 0x25（37），I²C 的时钟频率为 100KHz。

3.14 UART 相关寄存器

1. UART1 数据寄存器

Address = 0x8100-0070

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

位号	助记符	描述
7:0	RTD	接收保持寄存器（读访问）/发送保持寄存器（写访问）

2. UART1 状态寄存器

Address = 0x8100-0074

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																								FE	PE	OV	BR	TH	TS	DR	
r																								r/w	r/w	r/w	r/w	r	r	r	
0																								0	0	0	0	1	1	0	

位号	助记符	描述
6	FE	‘1’ 表示检测到帧错误
5	PE	‘1’ 表示检测到奇偶校验错误
4	OV	‘1’ 表示由于数据溢出而丢失一个或更多的字符
3	BR	‘1’ 表示接收到 BREAK
2	TH	‘1’ 表示发送保持寄存器为空
1	TS	‘1’ 表示发送移位寄存器为空
0	DR	‘1’ 表示可以在接收保持寄存器中获取新数据

3. UART1 控制寄存器

Address = 0x8100-0078

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																							EC	LB	FL	PE	PS	TI	RI	TE	RE
r																							r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
0																							0	0	0	x	x	x	x	0	0

位号	助记符	描述
8	EC	外部时钟使能，若为‘1’，UART 分频脉冲将从 PIO[3]管脚获取
7	LB	回环使能，若为‘1’，将进入回环模式
6	FL	流控使能，若为‘1’，可以使用 ctsn/rtsn 进行流控制
5	PE	奇偶校验使能，若为‘1’，允许进行奇偶校验
4	PS	奇偶校验选择，0 偶校验，1 奇校验
3	TI	发送中断使能，若为‘1’，允许产生发送中断
2	RI	接收中断使能，若为‘1’，允许产生接收中断
1	TE	发送使能，若为‘1’，可以发送数据
0	RE	接收使能，若为‘1’，可以接收数据

4. UART1 分频器重载寄存器

Address = 0x8100-007C

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

位号	助记符	描述
11:0	SRV	分频器重载值。 重载值= ((系统时钟*10) / (波特率*8) -5) /10。

5. UART2 数据寄存器

Address = 0x8100-0080

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

位号	助记符	描述
7:0	RTD	接收保持寄存器（读访问）/发送保持寄存器（写访问）

6. UART2 状态寄存器

Address = 0x8100-0084

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																								FE	PE	OV	BR	TH	TS	DR	
r																								r/w	r/w	r/w	r/w	r	r	r	
0																								0	0	0	0	1	1	0	

位号	助记符	描述
6	FE	‘1’ 表示检测到帧错误
5	PE	‘1’ 表示检测到奇偶校验错误
4	OV	‘1’ 表示由于数据溢出而丢失一个或更多的字符
3	BR	‘1’ 表示接收到 BREAK
2	TH	‘1’ 表示发送保持寄存器为空
1	TS	‘1’ 表示发送移位寄存器为空
0	DR	‘1’ 表示可以在接收保持寄存器中获取新数据

7. UART2 控制寄存器

Address = 0x8100-0088

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																							EC	LB	FL	PE	PS	TI	RI	TE	RE
r																							r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
0																							0	0	0	x	x	x	x	0	0

位号	助记符	描述
8	EC	外部时钟使能，若为‘1’，UART 分频脉冲将从 PIO[3]管脚获取
7	LB	回环使能，若为‘1’，将进入回环模式
6	FL	流控使能，若为‘1’，可以使用 ctsn/rtsn 进行流控制
5	PE	奇偶校验使能，若为‘1’，允许进行奇偶校验
4	PS	奇偶校验选择，0 偶校验，1 奇校验
3	TI	发送中断使能，若为‘1’，允许产生发送中断
2	RI	接收中断使能，若为‘1’，允许产生接收中断
1	TE	发送使能，若为‘1’，可以发送数据
0	RE	接收使能，若为‘1’，可以接收数据

8. UART2 分频器重载寄存器

Address = 0x8100-008C

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

位号	助记符	描述
11:0	SRV	分频器重载值。 重载值= ((系统时钟*10) / (波特率*8) -5) /10。

9. UART3 波特率分频寄存器

Address = 0x8100-01C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																BAUD_FREQ															
r																r/w															
0																0x104															

位号	助记符	描述
15:0	BAUD_FREQ	波特率分频值。上电复位值为 0x104 (40MHz 下 9600 波特率)。

10. UART3 发送/接收设置寄存器

Address = 0x8100-01C4

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

位号	助记符	描述
5:4	FC	FIFO 触发条件。 2'b00 接收全满 (32 字节), 或发送全空中断 2'b01 接收 1/4 满 (8 字节), 或发送 1/4 空中断 2'b10 接收 1/2 满 (16 字节), 或发送 1/2 空中断 2'b11 接收 3/4 满 (24 字节), 或发送 3/4 空中断 任何设置均有接收全满、发送全空、停止位错以及超时中断
3	LB	自发自收模式。为 1 表示自发自收模式, 为 0 表示正常工作模式
2	STOP	停止位选择。为 1 表示 2bits 停止位, 为 0 表示 1bit 停止位
1	PS	奇偶校验选择位, 1 奇校验, 0 偶校验
0	PE	奇偶校验使能位, 1 表示使能校验位, 0 表示不使能校验位

11. UART3 状态寄存器

Address = 0x8100-01C8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留									NTD	TO	SE	PE	OE	TE_NTB	RF	保留			TXD_CNT				保留		RXD_CNT						
r									r	r	r	r	r	r	r	r			r				r		r						
0									0	0	0	0	0	1	0	0	“100000”			0				0		0					

位号	助记符	描述
22	NTD	no_tx_data, 发送结束标志, 标志位置“1”, 表示所有发送数据均已写入发送缓存, 当写入新的发送数据时, 该位自动置0。上电复位值为0。
21	TO	Time_out, 接收超时标志, 为1表示当接收缓存不为空的一定时间内, 没有读取接收缓存的操作, 同时请求超时中断。上电复位值为0。
20	SE	Stop_error, 停止位错误标志, 为1表示接收数据停止位出错, 并一直保持, 直至写清状态寄存器。上电复位值为0。
19	PE	parity_error, 校验位错误标志, 为1表示检测到奇偶校验出错, 并一直保持, 直至写清状态寄存器, 0表示没有检测到奇偶校验错误。上电复位值为0。
18	OE	Overflow_error, 接收数据溢出标志, 为1表示接收数据溢出, 并一直保持, 直至写清状态寄存器。上电复位值为0。
17	TE_NTB	发送缓存全空或忙标志, 为1表示发送缓存全空, 为0表示发送缓存忙。上电复位值为1。
16	RF	Rx_full, 接收缓存全满标志, 为1表示接收缓存全满。上电复位值为0。
13:8	TXD_CNT	剩余可发送字节数(缓冲剩余字节数)。上电复位值为6'b100000, 最大值为32。
5:0	RXD_CNT	已接收字节数(缓冲已有字节数)。 上电复位值为6'b000000, 最大值为32。

12. UART3 数据寄存器

Address = 0x8100-01CC

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

位号	助记符	描述
7:0	DATA	接收数据（读访问）
7:0	DATA	接收数据（写访问）

13. UART3 清状态寄存器

Address = 0x8100-01D0

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

位号	助记符	描述
31:0	CLRS	清状态寄存器，写入有效，清除接收停止位错、接收校验位错误标志、接收溢出错状态标志。

14. UART3 发送结束寄存器

Address = 0x8100-01D4

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

位号	助记符	描述
31:0	WEND	写发送数据结束，写入有效，表示所有发送数据均已写入发送缓存，当发送空时，不再提出中断；写入新的发送数据时自动取消。

15. UART3 中断使能寄存器

Address = 0x8100-01D8

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

位号	助记符	描述
6	OT_INT	over_time_int, 超时中断使能
5	PE_INT	parity_error_int, 校验位错误中断使能
4	SE_INT	stop_error_int, 停止位错误中断使能
3:2	R	保留位，读出为 0
1	RF_INT	rx_full_int, 接收全满中断使能
0	TE_INT	tx_empty_int, 发送全空中断使能

16. UART4 波特率分频寄存器

Address = 0x8100-01E0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																BAUD_FREQ															
r																r/w															
0																0x104															

位号	助记符	描述



15:0	BAUD_FREQ	波特率分频值。上电复位值为 0x104 (40MHz 下 9600 波特率)。

17. UART4 发送/接收设置寄存器

Address = 0x8100-01E4

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

位号	助记符	描述
5:4	FC	FIFO 触发条件。 2'b00 接收全满 (32 字节), 或发送全空中断 2'b01 接收 1/4 满 (8 字节), 或发送 1/4 空中断 2'b10 接收 1/2 满 (16 字节), 或发送 1/2 空中断 2'b11 接收 3/4 满 (24 字节), 或发送 3/4 空中断 任何设置均有接收全满、发送全空、停止位错以及超时中断
3	LB	自发自收模式。为 1 表示自发自收模式, 为 0 表示正常工作模式
2	STOP	停止位选择。为 1 表示 2bits 停止位, 为 0 表示 1bit 停止位
1	PS	奇偶校验选择位, 1 奇校验, 0 偶校验
0	PE	奇偶校验使能位, 1 表示使能校验位, 0 表示不使能校验位

18. UART4 状态寄存器

Address = 0x8100-01E8

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留									NTD	TO	SE	PE	OE	TE_NTB	RF	R			TXD_CNT					R	RXD_CNT						
r									r	r	r	r	r	r	r	r			r					r	r						
0									0	0	0	0	0	1	0	0			“100000”					0	0						

位号	助记符	描述



位号	助记符	描述
22	NTD	no_tx_data, 发送结束标志, 标志位置“1”, 表示所有发送数据均已写入发送缓存, 当写入新的发送数据时, 该位自动置 0。上电复位值为 0。
21	TO	Time_out, 接收超时标志, 为 1 表示当接收缓存不为空的一定时间内, 没有读取接收缓存的操作, 同时请求超时中断。上电复位值为 0。
20	SE	Stop_error, 停止位错误标志, 为 1 表示接收数据停止位出错, 并一直保持, 直至写清状态寄存器。上电复位值为 0。
19	PE	parity_error, 校验位错误标志, 为 1 表示检测到奇偶校验出错, 并一直保持, 直至写清状态寄存器, 0 表示没有检测到奇偶校验错误。上电复位值为 0。
18	OE	Overflow_error, 接收数据溢出标志, 为 1 表示接收数据溢出, 并一直保持, 直至写清状态寄存器。上电复位值为 0。
17	TE_NTB	发送缓存全空或忙标志, 为 1 表示发送缓存全空, 为 0 是表示发送缓存忙。上电复位值为 1。
16	RF	Rx_full, 接收缓存全满标志, 为 1 表示接收缓存全满。上电复位值为 0。
13:8	TXD_CNT	剩余可发送字节数(缓冲剩余字节数)。上电复位值为 6'b100000, 最大值为 32。
5:0	RXD_CNT	已接收字节数 (缓冲已有字节数)。 上电复位值为 6'b000000, 最大值为 32。

19. UART4 数据寄存器

Address = 0x8100-01EC

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

位号	助记符	描述
7:0	DATA	接收数据（读访问）
7:0	DATA	接收数据（写访问）

20. UART4 清状态寄存器

Address = 0x8100-01F0

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

位号	助记符	描述
31:0	CLRS	清状态寄存器，写入有效，清除接收停止位错、接收校验位错误标志、接收溢出错状态标志。

21. UART4 发送结束寄存器

Address = 0x8100-01F4

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

位号	助记符	描述
31:0	WEND	写发送数据结束，写入有效，表示所有发送数据均已写入发送缓存，当发送空时，不再提出中断；写入新的发送数据时自动取消。

22. UART4 中断使能寄存器

Address = 0x8100-01F8

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

位号	助记符	描述
6	OT_INT	over_time_int, 超时中断使能
5	PE_INT	parity_error_int, 校验位错误中断使能
4	SE_INT	stop_error_int, 停止位错误中断使能
3:2	R	保留位, 读出为 0
1	RF_INT	rx_full_int, 接收全满中断使能
0	TE_INT	tx_empty_int, 发送全空中断使能

3.15 定时器相关寄存器

1. 定时器 1 计数寄存器

Address = 0x8100-0040

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIM0VAL																															
r/w																															
x																															

位号	助记符	描述
31:0	TIM0VAL	定时器 1 当前计数值, 预分频器下溢时, 该值减 1。可读可写, 复位初值为 X

2. 定时器 1 重载寄存器

Address = 0x8100-0044

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIM0RLD																															
r/w																															
x																															

位号	助记符	描述
31:0	TIM0RLD	定时器 1 重新加载值。如果在控制寄存器中设置了 RL0 位，那么在定时器溢出后，将自动加载该值。可读可写，复位初值为 X

3. 定时器 1 控制寄存器

Address = 0x8100-0048

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																									WTDEN0	保留	LD0	RL0	EN0		
r																									r/w	r	r/w	r/w	r/w		
0																									0	0	0	x	0		

位号	助记符	描述
4	WTDEN0	看门狗使能位。置 1 时使能看门狗，置 0 时则关闭看门狗。可读可写，复位初值为 0
2	LD0	定时器 1 重载位。此位置“1”时，定时器 1 计数寄存器将加载定时器 1 重载寄存器的值，并继续计数。在该位为“1”期间，定时器的溢出将不会产生中断。该位在设置“1”后的下一个周期将自动清“0”。复位初值为 X
1	RL0	此位的值决定了定时器在溢出后是继续进行还是停止。如果该位为“1”，那么在定时器 1 溢出后，将重新装载重载寄存器中的重载值，继续下一次定时；如果该位为“0”，在溢出后，定时器 1 将停止。复位初值为 X
0	EN0	定时器 1 使能位。此位为“1”时，定时器使能。复位初值为“0”

4. 看门狗计数值寄存器

Address = 0x8100-004C

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

位号	助记符	描述
31:0	WDC	看门狗当前计数值，预分频器下溢时，该值减 1。该位可读可写。复位初值为全“1”

5. 定时器 2 计数寄存器

Address = 0x8100-0050

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

位号	助记符	描述
31:0	TIMIVAL	定时器 2 当前计数值，预分频器下溢时，该值减 1。可读可写，复位初值为 X

6. 定时器 2 重载寄存器

Address = 0x8100-0054

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

位号	助记符	描述
31:0	TIM1RLD	定时器 2 重新加载值。如果在控制寄存器中设置了 RL1 位，那么在定时器溢出后，将自动加载该值。可读可写，复位初值为 X

7. 定时器 2 控制寄存器

Address = 0x8100-0058

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留																													LD1	RL1	EN1
r																													r/w	r/w	r/w
0																													0	x	0

位号	助记符	描述
2	LD1	定时器 2 重载位。此位置“1”时，定时器 2 计数寄存器将加载定时器 2 重载寄存器的值，并继续计数。在该位为“1”期间，定时器的溢出将不会产生中断。该位在设置“1”后的下一个周期将自动清“0”。复位初值为 0
1	RL1	此位的值决定了定时器在溢出后是继续进行还是停止。如果该位为“1”，那么在定时器 2 溢出后，将重新装载重载寄存器中的重载值，继续下一次定时；如果该位为“0”，在溢出后，定时器 2 将停止。复位初值为 X
0	EN1	定时器 2 使能位。此位为“1”时，定时器使能。复位初值为“0”

8. 定时器预分频计数寄存器

Address = 0x8100-0060

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
保留																						SCALERCNT											
r																						r/w											
0																						0x0											

位号	助记符	描述
9:0	SCALERCNT	预分频当前计数值位。该值每个时钟周期减“1”，在溢出后，自动重新加载预分频重载值寄存器中的重载值。有效的预分频的周期为（SCALERCNT+1）个系统时钟周期。该位可读可写，复位初值为全0。

9. 定时器预分频重载寄存器

Address = 0x8100-0064

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
保留																						SCALERLOAD															
r																						r/w															
0																						0x3FF															

位号	助记符	描述
9:0	SCALERLOAD	预分频重载值。当预分频计数器溢出后，自动加载该值。可读可写。复位初值为‘0x3FF’。

3.16 计数器相关寄存器

1. 计数器控制寄存器

Address = 0x8100-0210、0x8100-0220、0x8100-0230、0x8100-0240、0x8100-0250、

0x8100-0260

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

位号	助记符	描述
2:1	PULSEWIDTH	计数器脉冲宽度设定，为外部管脚脉冲计数时，设定需要的最小脉冲宽度，“00”对脉冲采样三次得到相同值（高电平

位号	助记符	描述
		或者低电平）认为值（高电平或者低电平）有效，“01”为采样四次，“10”为采样五次，“11”为采样六次。复位为“00”。
0	EN	计数器使能位，置 1 时，计数使能，计算外部管脚的脉冲数，可读写，复位为 0。

2. 计数器数据寄存器

Address = 0x8100-0214、0x8100-0224、0x8100-0234、0x8100-0244、0x8100-0254、0x8100-0264

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

位号	助记符	描述
31:0	CNT	计数器数据位，可读写，复位为 0。

3. 计数器分频寄存器

Address = 0x8100-0218、0x8100-0228、0x8100-0238、0x8100-0248、0x8100-0258、0x8100-0268

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

位号	助记符	描述
27:16	SCLRPRLD	计数器分频重载值。可读写，复位后为 0。
11:0	SCLRCNT	计数器分频计数值。可读写，复位后为 0。

3.17 PWM 相关寄存器

1. PWM 定时器控制寄存器

Address = 0x8100-0300

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

位号	助记符	描述
21:16	MASK	对应6路PWM定时器的使能位的写屏蔽位。复位全部为‘0’。Bit16 对应定时器 1，bit17 对应定时器 2，……，bit21 对应定时器 6。
5:0	EN	对应6路PWM定时器的使能位。置‘1’时，使能相应的定时器，复位全部为‘0’。bit0，对应定时器 1，bit1 对应定时器 2，……，bit5 对应定时器 6。定时器配置寄存器的 RL 位为‘0’时，当定时器计数寄存器溢出后会引起中断，中断会导致定时器控制寄存器的相应的 EN 位被清零，即使对应的 MASK 位为 0，EN 位也会被清零，定时器被关闭。 开启并使用定时器，需配置定时器寄存器的使能位，并在相应的 MASK 位同时为 1 时才能有效写入。若使能定时器 1，需将 bit0 置 1，与定时器 1 对应的 MASK 位置 1，则向寄存器写入 0x00010001 即可完成配置(若关闭则写 0x00010000)；若直接向寄存器写 0x00000001，由于 bit0 对应的 MASK 位 bit16 不同时为 1，则写入至 bit0 的数据无效，定时器不能开启。

2. PWM 定时器计数寄存器

Address = 0x8100-0310、0x8100-0320、0x8100-0330、0x8100-0340、0x8100-0350、0x8100-0360

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

位号	助记符	描述
23:0	TIMERCNT	PWM 定时器的计数值寄存器，定时器 1 预分频器下溢时，该值减 1。可读写，复位后为 0

3. PWM 定时器预置值寄存器 1

Address = 0x8100-0314、0x8100-0324、0x8100-0334、0x8100-0344、0x8100-0354、0x8100-0364

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LD	保留							TIMERLOAD1																							
r/w	r							r/w																							
1	0							0																							

位号	助记符	描述
31	LD	加载位，置 ‘0’ 时，定时器 1 计数寄存器将定时器 1 预置值 1 载入。复位后为 ‘1’。若该位置 0，则在置 0 后的下一个时钟周期后自动复位为 1。
23:0	TIMERLOAD1	定时器 1 预置值 1。可读可写，复位后为 0。当定时器 1 输出模式选为单脉冲或者电平翻转时，此寄存器用来设置定时的长度，即定时器 1 计数多少次后发生翻转；当定时器 1 输出模式选为 PWM 输出方式时，该寄存器对应低电平输出长度。与定时器 1 预置值寄存器 2 相互配合，输出一定占空比的波形。有效的计数周期为 TIMERLOAD1+1 个系统时钟周期。

4. PWM 定时器预置值寄存器 2

Address = 0x8100-0318、0x8100-0328、0x8100-0338、0x8100-0348、0x8100-0358、0x8100-0368

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留								TIMERLOAD2																							
r								r/w																							
0								0																							

位号	助记符	描述
23:0	TIMERLOAD2	定时器预置值 2。可读写，复位后为 0。在定时器 1 的输出模式选为 PWM 输出时，该寄存器对应高电平输出长度，与定时器 1 预置值寄存器 1 配合输出一定占空比的波形。在输出模式选为单脉冲方式或者电平翻转方式时，该寄存器不参与定时。该值参与定时时，有效的计数周期为 TIMERLOAD2 + 1 个系统时钟周期。

5. PWM 定时器配置寄存器

Address = 0x8100-031C、0x8100-032C、0x8100-033C、0x8100-034C、0x8100-035C、0x8100-036C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留		SCLRPLD										保留		SCLCNT										STAT	保留		CS	OT		SS	RL
r		r/w										r		r/w										r	r	r/w	r/w	r/w	r/w		
0		0										0		0										0	0	0	0	0	0		

位号	助记符	描述
29:20	SCLRPLD	定时器预分频重载值。可读写，复位后为 0。
17:8	SCLCNT	定时器预分频计数值，每时钟周期自动减 1，有效预分频值为该值加 1。可读写，复位后为 0。
7	STAT	定时器状态位，可读不可写，0 表示定时器 1 不工作，1 表示定时器 1 正在定时。
4	CS	定时器时钟选择，选择定时器 1 由哪个时钟驱动定时。1 为外部时钟，0 为内部时钟。复位后为 '0'。
3:2	OT	定时器输出信号方式，00 为单脉冲，01 为电平，10 为 PWM。

位号	助记符	描述
		复位后为“00”。
1	SS	定时器启动信号选择位。1 为外部信号实启，0 为软件配置启动。复位后为‘0’。
0	RL	定时器重载位。置位时，定时器计数寄存器溢出后自动将预置值寄存器的值载入。复位后为‘0’。RL 位为‘0’时，当定时器 1 计数寄存器溢出后会引起中断，中断会导致由软件配置启动的定时器控制寄存器的相应的 EN 位被清零或外部信号实启的定时器选择位被清零，导致软件配置启动或外部信号实启的定时器被关闭。

3.18 DSU 相关寄存器

1. DSU 控制寄存器

Address = 0x9000-0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
保留			DCNT									RE	DR	LR	SS	PE	EE	EB	DM	DE	BZ	BX	BB	BN	BS	BW	BE	FT	BT	TM	TE	
r			r/w									w	r/w	r/w	r/w	r	r	r	r	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w
0			x									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- ◆ [31:29]: 保留位。
- ◆ DCNT[28:20]: AHB Trace 延时计数器，延时冻结 Trace 功能的计数值，与 DE 位配合使用，可读写。
- ◆ RE[19]: 错误模式恢复位，如果置位为 1，则将使处理器从错误模式中复位。
- ◆ DR[18]: 调试模式响应位，如果置位为 1，则在处理器进入调试模式时从 DSU 串口输出高电平响应。可读写。
- ◆ LR[17]: 调试模式下 DSU 读响应位，如果置位为 1，则在 DSU 读命令时从 DSU 串口输出高电平响应。可读写。
- ◆ SS[16]: 单步执行控制位，如果置位为 1，则控制处理器执行一条指令后将重新返回

调试状态。可读写。

- ◆ PE[15]: 处理器错误模式状态位, 当处理器是错误模式时为 1, 否则为 0。只读。
- ◆ EE[14]: 外部 DSUEN 信号指示位。只读。
- ◆ EB[13]: 外部 DSUBRE 信号指示位。只读。
- ◆ DM[12]: 调试模式指示位, 与外部 DSUACT 信号作用相同。只读。
- ◆ DE[11]: 延时计数模式使能位, 如果置位为 1, 则在 DSU 断点命中时延时计数器开始减 1 计数, 当计数值为 0 时 AHB Trace 功能才会冻结, 否则立即冻结。
- ◆ BZ[10]: 处理器能被挂起的条件位, 如果置位为 1, 则表示特定陷阱才能引起处理器进入调试模式, 此特殊陷阱为不包括特权指令、FPU 禁止、窗口溢出、异步中断、ticc 陷阱的陷阱。
- ◆ BX[9]: 处理器能被挂起的条件位, 如果置位为 1, 则表示任意陷阱都能引起处理器进入调试模式。
- ◆ BB[8]: 处理器能被挂起的条件位, 如果置位为 1, 则在 DSU 断点命中时将强制处理器进入调试模式。
- ◆ BN[7]: 处理器能被挂起的条件位, 如果置位为 1, 则立即进入调试模式, 否则恢复执行模式。
- ◆ BS[6]: 处理器能被挂起的条件位, 如果置位为 1, 则在执行中断指令 (ta 1 指令) 时进入调试模式。
- ◆ BW[5]: 处理器能被挂起的条件位, 如果置位为 1, 则在 IU 观测点命中时进入调试模式 (trap 0xb)。
- ◆ BE[4]: 处理器能被挂起的条件位, 如果置位为 1, 则当处理器进入错误状态时进入调试模式 (trap in trap)。
- ◆ FT[3]: 定时器时钟冻结位, 如果置位为 1, 则在调试模式下定时器处于挂起状态, 从而应用程序的时间能保持不变。
- ◆ BT[2]: 处理器能被挂起的条件位, 如果置位为 1, 则在 trace 延时功能使能条件下 trace 功能冻结时才会进入调试模式。
- ◆ TM[1]: 总线追踪延时或指令追踪延时设置位, 如果置位为 1, 则在混合追踪模式下进行 AHB 总线功能延时冻结, 否则进行指令追踪功能延时冻结。

- ◆ TE[0]: Trace 功能使能位, 如果置位为 1, 则表示使能 trace 功能。

2. Trace buffer 控制寄存器

Address = 0x9000-0004

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留						TA	TI	保留			AHB INDEX						保留			INST INDEX											
r						r/w	r/w	r			r/w						r			r/w											
0						x	x	000			x						000			x											

- ◆ [31:26]: 保留位。
- ◆ TA[25]: AHB 总线追踪使能位。
- ◆ TI[24]: 指令追踪使能位。
- ◆ [23:21]: 保留位。
- ◆ AHB INDEX[20:12]: 当前所追踪总线信息存放位置索引。
- ◆ [11:9]: 保留位。
- ◆ INST INDEX[8:0]: 当前所追踪指令信息存放位置索引。

3. DSU Trace 时间标志寄存器

Address = 0x9000-0008

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留		DSU TIME TAG VALUE																													
r		r/w																													
00		0																													

- ◆ [31:30]: 保留。
- ◆ DSU TIME TAG VALUE[29:0]: DSU 跟踪信息时间标志计数值。

4. DSU 断点寄存器 1

Address = 0x9000-0010

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

- ◆ BADD[31:2]: 断点地址。
- ◆ [1]: 保留，读出为 0。
- ◆ EN[0]: 指令执行时使能比较位，本功能禁止使用。

5. DSU 屏蔽地址寄存器 1

Address = 0x9000-0014

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

- ◆ BMA[31:2]: 屏蔽地址位。与 DSU 断点寄存器配合使用，如果某位为 0，则会在判断断点命中时忽略 DSU 断点寄存器的相应位，表明该位不同也能命中。
- ◆ LD[1]: 如果置位为 1，则读数据时进行地址比较。
- ◆ ST[0]: 如果置位为 1，则写数据时进行地址比较。

6. DSU 断点寄存器 2

Address = 0x9000-0018

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

- ◆ BADD[31:2]: 断点地址。
- ◆ [1]: 保留，读出为 0。

- ◆ EN[0]: 指令执行时使能比较位, 本功能禁止使用。

7. DSU 屏蔽地址寄存器 2

Address = 0x9000-001C

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

- ◆ BMA[31:2]: 屏蔽地址位。与 DSU 断点寄存器配合使用, 如果某位为 0, 则会在判断断点命中时忽略 DSU 断点寄存器的相应位, 表明该位不同也能命中。
- ◆ LD[1]: 如果置位为 1, 则读数据时进行地址比较。
- ◆ ST[0]: 如果置位为 1, 则写数据时进行地址比较。

8. DSU 串口状态寄存器

Address = 0x8000-00C4

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
保留																										FE	保留	OV	BR	TH	TS	DR
r																										r/w	r	r/w	r/w	r	r	r
0																										0	0	0	0	1	1	0

- ◆ [31:7]: 保留位, 读出为 0。
- ◆ FE[6]: 数据帧错误指示, 1 表示错误。
- ◆ [5]: 保留位, 读出为 0。
- ◆ OV[4]: 数据溢出, 1 表示溢出。
- ◆ BR[3]: 1 表示接收到 break, 0 表示没有接收到 break。
- ◆ TH[2]: 发送保持寄存器空指示, 1 表示为空。
- ◆ TS[1]: 发送移位寄存器空指示, 1 表示为空。
- ◆ DR[0]: 置为 1 表示数据准备好, 但未被总线读出。

9. DSU 串口控制寄存器

Address = 0x8000-00C8

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

- ◆ [31:2]: 保留位。
- ◆ BL[1]: 波特率锁定指示, 在波特率自动检测完成之后, 1 表示锁定。
- ◆ EN[0]: 接收发送使能, 当置位为 1 时, 则允许接收和发送。系统自动置位, 也可以手动写置位。

10. DSU 串口波特率重载寄存器

Address = 0x8000-00CC

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
保留														SRV																	
r														w																	
0														0x3FFFB																	

- ◆ [31:18]: 保留位。
- ◆ SRV[17:0]: Scaler Reload Value, 波特率重载值。可以由系统自动重置, 或手动重置。

3.19 写保护相关寄存器

1. 写保护寄存器 1 (Write Protect Register1, WPR1)

Address = 0x8100-001C

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
E N	B P	保留	TAG														保留	MASK													
r/ w	r/ w	r	r/w														r	r/w													
0	x	0	XXXXXXXXXXXXX														0	XXXXXXXXXXXXX													

- ◆ EN[31]: 写保护使能, “1” 为使能, “0” 为不使能。

- ◆ BP[30]: block 保护模式, “1” 为 block 保护模式, “0” 为 segment 保护模式。
- ◆ [29]: 保留, 读出为 0。
- ◆ TAG[28:15]: 比较地址。
- ◆ [14]: 保留, 读出为 0。
- ◆ MASK[13:0]: 掩码地址。

2. 写保护寄存器 2 (Write Protect Register2, WPR2)

Address = 0x8100-0020

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
E N	B P	保留	TAG														保留	MASK															
r/ w	r/ w	r	r/w														r	r/w															
0	x	0	XXXXXXXXXXXXXXXX														0	XXXXXXXXXXXXXXXX															

- ◆ EN[31]: 写保护使能, “1” 为使能, “0” 为不使能。
- ◆ BP[30]: block 保护模式, “1” 为 block 保护模式, “0” 为 segment 保护模式。
- ◆ [29]: 保留, 读出为 0。
- ◆ TAG[28:15]: 比较地址。
- ◆ [14]: 保留, 读出为 0。
- ◆ MASK[13:0]: 掩码地址。

附录 4 寄存器初始化

4.1 寄存器初始化简介

BM3823A 实现了内核寄存器和片上寄存器，根据寄存器的实现和应用要求，相关寄存器上电复位后应软件初始化，为用户程序设置一个固定的 CPU 初始环境，确保应用程序不发生异常。

初始化寄存器的基本要求是，在写 REGFILE 寄存器（含窗口寄存器、浮点数据寄存器）之前应先初始化 ASR16 寄存器。

根据寄存器功能，可把寄存器分为三部分：内核寄存器、片上寄存器、用户应用程序相关寄存器。依据使用习惯，寄存器初始化过程可按此三部分顺次初始化（实际对配置顺序不做强制要求，以满足应用要求为目标）。流程示意图如图 A4-1-1 所示。

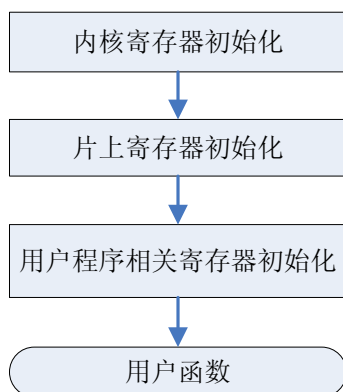


图 A4-1-1 启动程序初始化流程

4.2 推荐寄存器初始化过程

4.2.1 内核寄存器初始化

内核寄存器包括IU及FPU相关的寄存器以及辅助寄存器。如窗口寄存器、浮点数据寄存器、PSR、WIM、TBR、FSR、Y寄存器、ASR16、ASR17、ASR24-ASR31寄存器。初始化示意流程图如下所示。

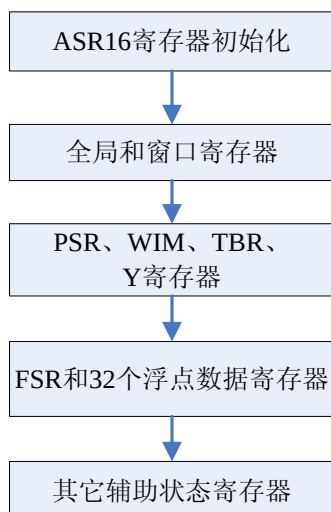


图 A4-2-1 内核寄存器初始化流程

首先需要初始化的寄存器为ASR16，因为硬件已经考虑了ASR16寄存器的初始化，所以即使ASR16寄存器没有软件初始化，上电后处理器也不会处在REGFILE造错模式。

4.2.1.1 辅助状态寄存器 ASR16

CPU 上电复位后写内核寄存器前首先需要对 ASR16 进行初始化，即关闭造错功能，初始化值推荐为 0x0（确保高 3 位不处于造错模式），示例代码如下所示（叹号!之后的语句为本行注释）：

```

mov %g0, %asr16      ! %asr16清零
nop;nop;nop;

```

4.2.1.2 IU 相关的寄存器

（1）全局和窗口寄存器

在寄存器堆中的寄存器，包括：全局寄存器%g1-%g7，寄存器窗口中的局部寄存器%i0-%i7、输出寄存器%o0-%o7（与%i0-%i7 在交叠窗口中物理实现为同一寄存器），32 个浮点寄存器（%f0-%f31），由于采用了 EDAC 设计，要求在读之前确保其 32 位数据和 7 位校验码匹配，读访问之前完成初始化即可。启动时软件初始化这些寄存器，可以确保应用程序未写先读发生异常。

这些窗口寄存器可以全部设置为 0x0。初始化时当前窗口指向窗口 0，trap 禁止并设置处理器处于管理态（处理器状态寄存器 PSR 的初值设置为 0x10c0），然后将全局寄存器%g1~%g7 清零，旋转窗口，依次把每个窗口的 local 寄存器、input/output 寄存器清零（可

以初始化为其它值，清零更简单)，示例如下：

```
set 0x10c0,%g1
mov %g1,%psr
nop
nop
nop
mov %g0,%wim !WIM设置为不存在无效窗口，避免旋转窗口时上下溢
nop !三周期内指令不涉及WIM，此处3条nop可省
nop !本附录中的例子并不是所有状态寄存器后都增加了三周期
nop
mov %g0,%g1
mov %g0,%g2
mov %g0,%g3
mov %g0,%g4
mov %g0,%g5
mov %g0,%g6
mov %g0,%g7
! init the local/in/out registers
mov 0x8,%g1 !寄存器窗口为8个
1: mov %g0,%l0 !设置标号1
mov %g0,%l1
mov %g0,%l2
mov %g0,%l3
mov %g0,%l4
mov %g0,%l5
mov %g0,%l6
mov %g0,%l7
mov %g0,%i0
```

```
mov %g0, %i1
mov %g0, %i2
mov %g0, %i3
mov %g0, %i4
mov %g0, %i5
mov %g0, %i6
mov %g0, %i7
subcc %g1,1,%g1
save      !旋转窗口
bne 1b    !%g1不为0（以%psr.icc.Z判断），则回跳到标号1处
nop
```

（2）其它 IU 寄存器

初始化窗口寄存器时，PSR 寄存器的值设定为 0x10c0（使能浮点单元、关闭 trap 使能、设置处理器处于管理态、当前窗口指针指向 0 窗口），即在“全局和窗口寄存器初始化”一节的设置值即满足要求。

窗口无效屏蔽寄存器（WIM），由于%psr.CWP 已设置为 0，也就是当前窗口指向 0 窗口，为了最大化利用窗口寄存器，一般把 WIM 设置为窗口 1，即 WIM 设置为 2，示例代码如下：

```
set 2, %g1
mov %g1, %wim
nop
nop
nop
```

异常基址寄存器（TBR）定义了响应 trap/中断时系统的跳转目标的基地址，必须是一个 4K 字节地址对齐的值，一般是 trap 表的标号。示例如下：

```
sethi %hi(_trap_table), %g1    !或者用绝对值，如set 0x20000000,%g1
mov %g1, %tbr
nop
```

```
nop
```

```
nop
```

乘法/除法寄存器（Y）的初始值可以初始化为 0：

```
wr %g0,0,%y
```

```
nop
```

```
nop
```

```
nop
```

4.2.1.3 FPU 相关的寄存器

（1）浮点状态寄存器

在使用浮点指令之前必须先使能浮点处理单元（一般在 PSR 中已经将其使能），之后配置**浮点状态寄存器 FSR**。该寄存器初始值必需从存储器中载入，如先定义一个 WORD 型（32 位）变量（初始值设置为 0），之后使用该变量初始化 FSR。参考代码如下：

```
.data !该变量数据段，如果不在数据段，也可以使用
```

```
.global _fsrinit !声明变量，名字叫_fsrinit
```

```
.align 4 !该变量存放的地址是四字节对齐
```

```
_fsrinit:
```

```
.word 0x0 !变量的值，也是 fsr 寄存器的初值
```

```
set _fsrinit,%g1 !_fsrinit 变量存放的地址
```

```
ld [%g1], %fsr
```

```
nop
```

```
nop
```

```
nop
```

（2）浮点数据寄存器

%f0~%f31 寄存器可以从存储器中初始化，如先定义一个 WORD 型变量（可设初始值为 0x0），之后使用该变量初始化。参考代码为：

```
.global _fpdata ! 定义变量_fpdata
```

```
.align 4 !四字节对齐
```

`_fpdata:`

`.word 0` !浮点寄存器的值

`set _fpdata,%g1`

`ld [%g1], %f0` !其它 f1-f31 也如此, 作为目的寄存器进行初始化或者将 `_fpdata` 定义为 8 字节对齐数据, 使用 `ldd` 指令一次初始化连续两个浮点数据寄存器。

`.align 8` !八字节对齐

`_fpdata:`

`.word 0,0` !浮点寄存器的值, 占两个 word

`set _fpdata,%g1`

`ldd [%g1], %f0` !将 f0,f1 初始化为 0

`ldd [%g1], %f2` !将 f2,f3 初始化为 0, 其它用 f4-f30 的寄存器如此初始化

当初始化完 %f0 寄存器后, 可以使用浮点操作指令, 初始化其它浮点寄存器, 由于不访问数据存储器, 速度稍快, 如:

`fmovs %f0,%f1` !浮点数据寄存器 %f1 作为目的寄存器完成 %f1 初始化, 但在浮点协处理器被使能之前, 不能使用浮点指令(初始化浮点寄存器)。

4.2.1.4 其他辅助状态寄存器

辅助寄存器中 ASR17、ASR24、ASR26、ASR28、ASR30 和 ASR25、ASR27、ASR29、ASR31 寄存器, 一般设置为 0, 无配置顺序要求, 参考代码如下:

`mov %g0,%asr17`

`mov %g0,%asr24`

`mov %g0,%asr26`

`mov %g0,%asr28`

`mov %g0,%asr30`

`mov %g0,%asr25`

`mov %g0,%asr27`

`mov %g0,%asr29`

`mov %g0,%asr31`

4.2.2 片上寄存器初始化

片上寄存器包括外部存储器及容错配置寄存器、Cache控制及容错寄存器、中断控制寄存器、GPIO、PCI相关寄存器、定时器、串口、以太网、I²C、SPI、PWM定时器以及DSU寄存器。这些寄存器可根据应用需要初始化，流程示意图如图A4-2-2所示。

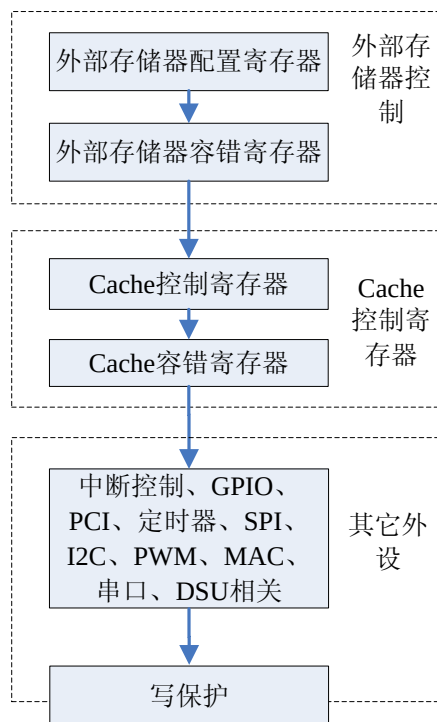


图 A4-2-2 片上寄存器初始化流程

4.2.2.1 外部存储器配置寄存器

外部存储器接口配置通过**外部存储器配置寄存器** MCFG1、MCFG2、MCFG3 来实现，初始值需要根据外部存储器系统设计来初始化，是与硬件相关的。如果硬件设计不包括 SDRAM 存储器，则可以不设置 MCFG3。

外部存储器容错配置寄存器实现对外部存储器的纠检错功能的控制，包括外部存储器容错配置寄存器 MECFG1、MECFG2；在不造错的模式下，后三个寄存器不需要配置。在 RAM 空间 EDAC 使能时，需要初始化 MECFG1 (0x81000100)。

如RAM存储器系统设计为无SDRAM，32位SRAM置于bank 0（另有8位校验SRAM），bank大小为2M字节，读、写等待周期分别设置为3个CPU周期，且该BANK使能EDAC。则需要初始化MCFG2和MECFG1。参考代码为：

```
set 0x81000000, %g1
```

```
set 0x00019063, %g2 !使能 EDAC 时, 可不明确指出使用读改写模式
```

```
st %g2, [%g1+0x4]
```

```
set 0x80400000, %g2 !设置为 0x80000000 也满足应用要求,
```

!仅考虑 sram, 不应影响 SDRAM 和 PROM 的 EDAC

```
st %g2, [%g1+0x100]
```

上电后 MCFG1 可以不重新初始化, 除非用到 IO 空间或修改 ROM 空间的读写等待周期; 或者程序跳转到其它存储器空间后修改 ROM 空间的其它属性。

4.2.2.2 Cache 控制寄存器

Cache 相关寄存器包括 **Cache 控制寄存器**和 **Cache 容错寄存器**。初始化时可根据实际需要选择使能 Cache 或禁止 Cache, 使能的参考代码为:

```
set 0x80000000, %g1
```

```
set 0x0000000f, %g2 ! 使能指令和数据 Cache
```

```
st %g2, [%g1+0x14]
```

如果要禁止 Cache, 直接向 CCR 寄存器写 0 即可。

```
set 0x80000000, %g1
```

```
st %g0, [%g1+0x14]
```

Cache 刷新操作在使能 Cache 前进行。

Cache 容错控制寄存器 1、2、3 的上电默认值即可正常使用, 不需要初始化。

4.2.2.3 中断控制寄存器

中断控制主要包括中断相关的四个寄存器, 在初始化时需要清除**中断强制和未决寄存器**, 并屏蔽全部中断。初始化参考代码为:

```
set 0x80000000, %g1
```

```
st %g0, [%g1+0x90] ! level 和 mask 均置 0
```

```
st %g0, [%g1+0x94] !清除所有强制中断
```

```
set 0xffff, %g2
```

```
st %g2, [%g1+0x9c] !清除所有未决中断
```


4.2.2.4 写保护寄存器

写保护功能对 RAM 空间的意外写操作进行保护，支持两个写保护寄存器分别对不同地址区域进行写保护。相关寄存器为**写保护寄存器 1**和**写保护寄存器 2**；初始化为禁止写保护的参考代码为：

```
set    0x81000000, %g1
st     %g0, [%g1+0x1c]
st     %g0, [%g1+0x20]
```

4.2.2.5 其它控制寄存器

GPIO 方向寄存器在上电复位后为输入，**GPIO 中断配置寄存器**的复位初始值也是不产生中断的设置，使用时根据需要初始化。

串口中，复位后控制寄存器值为发送/接收不使能，不使用可不进行初始化。

定时器和看门狗上电后是不使能的，不使用可以不进行初始化。

系统寄存器是只读的，不需要初始化。

PCI 模块上电复位后，与 CPU 中断相关的三个寄存器和两个 DMA 读写控制寄存器复位值均为 0，不使用可不进行初始化。

DSU 在运行模式不需要使用，禁止 DSU 串口发送接收（调试程序时不能禁止）参考代码如下：

```
set    0x80000000, %g1
st     %g0, [%g1+0xc8]
```

其它外设的相关配置寄存器根据实际使用需求进行配置，不使用时将有关功能配置为不使能状态。

4.2.3 用户程序相关寄存器初始化

完成处理器内核寄存器和片上寄存器的初始化后，可以进行应用程序相关寄存器的初始化，主要是指指定栈的栈底，并使能中断和TRAP。示例如图A4-2-3所示。

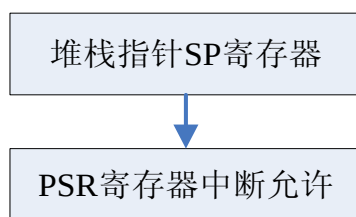


图 A4-2-3 用户程序相关寄存器初始化

根据应用情况设置堆栈栈底指针，在本示例中把堆栈指针设置为SRAM存储器的高端位置，如SRAM空间大小为1MB，初始化参考代码如下所示。

```
set 0x20FFF00, %g1 !预留了0x100字节在存储器高地址处
```

```
mov %g1, %sp
```

然后使能系统的TRAP允许并向用户主函数如main跳转。

```
set 0x10e0, %g1,
```

```
mov %g1, %psr
```

```
nop
```

```
nop
```

```
call main !向用户函数如main函数跳转
```

```
nop
```

```
nop
```

片上外设相关的寄存器，在使用时应根据应用要求进行相应设置。在使用中断时，一般先初始化中断源设备的相关寄存器，再设置中断控制器相关的寄存器。

附录 5 软件开发规范

5.1 通用编程规范

5.1.1 寄存器窗口与函数调用

5.1.1.1 函数调用

函数调用的一般原则：

- (1) 调用函数通过 CALL 指令使 PC 指针分支到被调函数的入口地址。
- (2) 被调函数使用 SAVE 指令获得新的寄存器窗口，并取得新的堆栈空间。
- (3) 被调函数使用“RET; RESTORE”指令组合返回调用函数。

例 1：调用函数：main（）；被调函数：printf（）

main:

call printf ! 调用 printf（）

nop ! 延迟一条指令

nop ! 位置

...

printf:

save %sp, -1024, %sp ! 获得新寄存器窗口

! 和新堆栈空间

...

ret

restore ! 被调函数返回

! main（）位置

5.1.1.2 参数传递

参数传递的一般规则：

- (1) 调用函数将参数写入当前寄存器窗口的输出寄存器中（%o0~%o5）。如果参数多于 6 个，则从第 7 个参数开始写入堆栈。
- (2) 调用函数使用 CALL 指令调用被调函数。
- (3) 被调函数使用 SAVE 获得新寄存器窗口和堆栈后，从新寄存器窗口的输入寄存器

(%i0~%i5) 取得参数。如果参数多于 6 个，则从调用函数的堆栈取得后面的参数。

(4) 被调函数将返回值放入输入寄存器。返回调用函数后，调用函数从输出寄存器中取得返回值。

例 2:

```
int reg_window(int a1,int a2,int a3,int a4, int a5, int a6, int a7,int a8)
{
    int b;

    b=a1+a2+a3+a4+a5+a6+a7+a8;

    return b;
}

main()
{
    int c1=1,c2=2,c3=3,c4=4,c5=5,c6=6,c7=7,c8=8;

    int sum = 0;

    sum = reg_window(c1,c2,c3,c4,c5,c6,c7,c8);

    sum = sum + 1;
}
```

以上两函数编译后生成:

<main>:

```
save    %sp, -152, %sp
mov     1, %o0
st      %o0, [ %fp + -12 ]
mov     2, %o0
st      %o0, [ %fp + -16 ]
mov     3, %o0
st      %o0, [ %fp + -20 ]
```

```
mov 4, %o0
st %o0, [ %fp + -24 ]
mov 5, %o0
st %o0, [ %fp + -28 ]
mov 6, %o0
st %o0, [ %fp + -32 ]
mov 7, %o0
st %o0, [ %fp + -36 ]
mov 8, %o0
st %o0, [ %fp + -40 ]
clr [ %fp + -44 ]
ld [ %fp + -36 ], %o0
st %o0, [ %sp + 0x5c ] ! 参数 c7, 堆栈传递
ld [ %fp + -40 ], %o0
st %o0, [ %sp + 0x60 ] ! 参数 c8, 堆栈传递
ld [ %fp + -12 ], %o0 ! 参数 c1
ld [ %fp + -16 ], %o1 ! 参数 c2
ld [ %fp + -20 ], %o2 ! 参数 c3
ld [ %fp + -24 ], %o3 ! 参数 c4
ld [ %fp + -28 ], %o4 ! 参数 c5
ld [ %fp + -32 ], %o5 ! 参数 c6
call 40001d30 <reg_window>
nop
st %o0, [ %fp + -44 ] ! 从输出寄存器
! 取得返回值 b→sum
ld [ %fp + -44 ], %o0
add %o0, 1, %o1
st %o1, [ %fp + -44 ]
```

ret

restore

<reg_window>:

```
save    %sp, -112, %sp
st    %i0, [ %fp + 0x44 ]    ! 输入参数, c1→a1
st    %i1, [ %fp + 0x48 ]    ! 输入参数, c2→a2
st    %i2, [ %fp + 0x4c ]    ! 输入参数, c3→a3
st    %i3, [ %fp + 0x50 ]    ! 输入参数, c4→a4
st    %i4, [ %fp + 0x54 ]    ! 输入参数, c5→a5
st    %i5, [ %fp + 0x58 ]    ! 输入参数, c6→a6

ld    [ %fp + 0x44 ], %o0
ld    [ %fp + 0x48 ], %o1
add    %o0, %o1, %o0
ld    [ %fp + 0x4c ], %o1
add    %o0, %o1, %o0
ld    [ %fp + 0x50 ], %o1
add    %o0, %o1, %o0
ld    [ %fp + 0x54 ], %o1
add    %o0, %o1, %o0
ld    [ %fp + 0x58 ], %o1
add    %o0, %o1, %o0
ld    [ %fp + 0x5c ], %o1    ! 输入参数, c7→a7
add    %o0, %o1, %o0
ld    [ %fp + 0x60 ], %o1    ! 输入参数, c8→a8
add    %o0, %o1, %o0
st    %o0, [ %fp + -12 ]
ld    [ %fp + -12 ], %o0
```

```
mov  %o0, %i0    ! 返回值 b
b    40001d9c <reg_window+0x6c>
nop
ret
restore
```

5.1.1.3 C 语言、汇编语言函数混合调用

C 语言与汇编语言函数相互调用时同样遵循上述参数传递原则。不同的是，需要手工确定参数地址和顺序。

例 3，C 程序调用汇编语言函数：

使用汇编语言定义数据向左移位函数 `asm_shift_function(p1,p2)`；//`p1`=被位移的数据；`p2`=位移量。例中使用 `asm_shift_function()` 将 1 左移 2 位。代码如下：

```
asm("
asm_shift_function:
save %sp, -112, %sp
sll %i0, %i1, %i0    ! 从输入寄存器中取得输入参数
ret
restore
");

main()
{
    int result = 0;
    result = asm_shift_function(1,2); // 输入参数
}
```

汇编后生成：

```
<asm_shift_function>: ! 编译器不修改汇编语言编写的代码
save  %sp, -112, %sp
sll  %i0, %i1, %i0
```

```
ret
```

```
restore
```

```
<main>:
```

```
save %sp, -152, %sp
```

```
clr [%fp + -48]
```

```
mov 1, %o0 ! 按照 C 代码顺序将参数放入输出寄存器
```

```
mov 2, %o1 ! 按照 C 代码顺序将参数放入输出寄存器
```

```
call 40001da4 <asm_shift_function>
```

```
nop
```

```
st %o0, [%fp + -48] ! 从输出寄存器中取得函数返回值
```

```
ret
```

```
restore
```

5.1.2 堆栈

SPARC V8 标准定义的程序堆栈由堆指针（%fp）和栈指针（%sp）组成，由高地址向低地址生长。

堆栈使用方法如图 A5-1-1 所示。

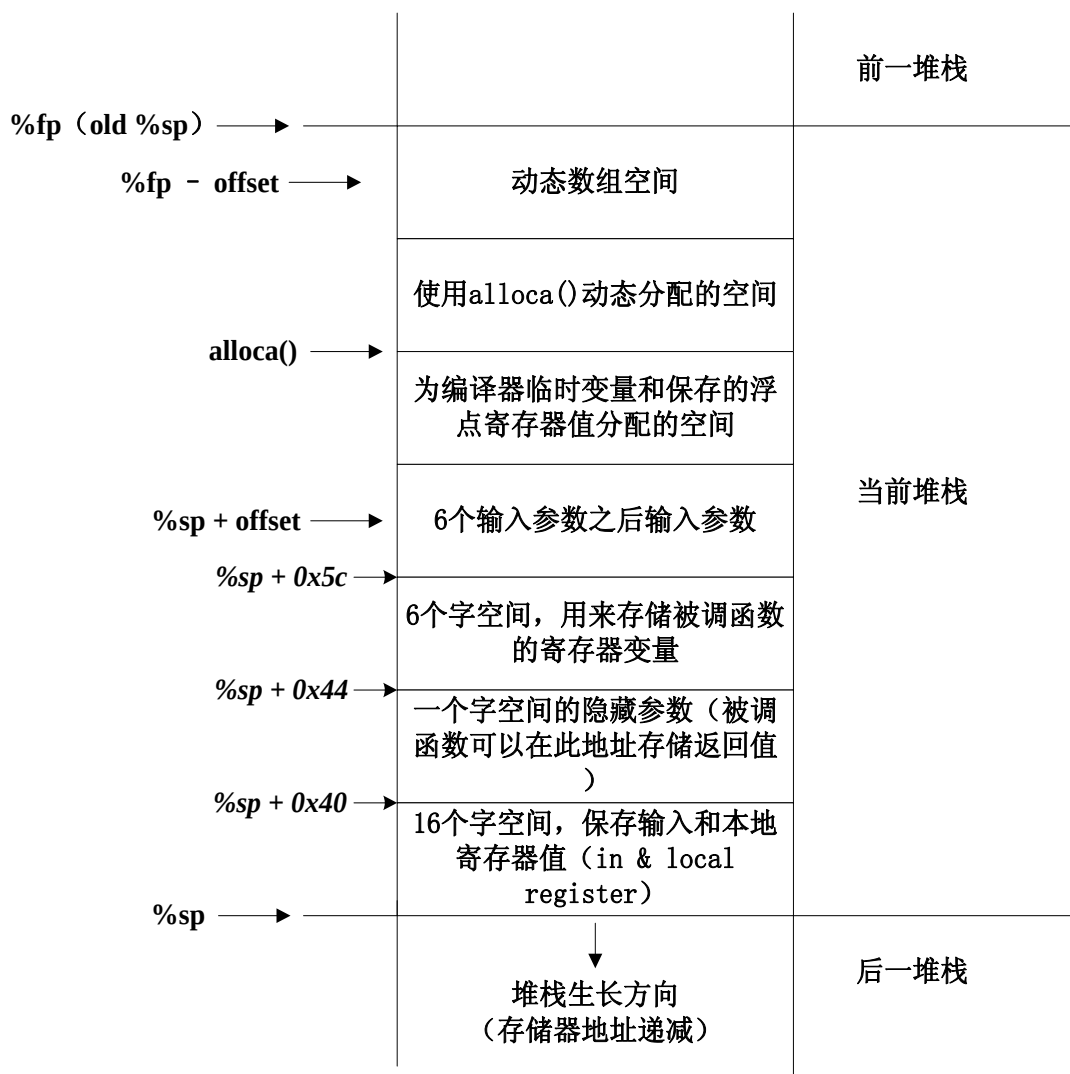


图 A5-1-1 堆栈使用方法

每次分配一个寄存器窗口，同时也分配一段存储器空间作为堆栈。由于堆栈向上生长，因此通常程序段首地址在 ROM 或 RAM 的地址低端，堆栈段首地址在 RAM 的地址高端，以达到 RAM 利用率最高。

5.1.3 陷阱和中断

5.1.4 Cache 刷新

BM3823A 可以通过指令刷新 Cache，刷新完成后，Cache 的 tag 位全部清零。

5.1.4.1 刷新方法

◆ 指令 Cache 刷新（以下三种方法均能实现）

(1) 执行 flush 指令；

(2) **Cache 控制寄存器**的 21 位写 1;

(3) 执行 asi=0x5 的 sta 指令 (写任意地址)。

◆ 数据 Cache 刷新 (以下三种方法均能实现)

(1) 执行 flush 指令;

(2) **Cache 控制寄存器**的 22 位写 1;

(3) 执行 asi=0x6 的 sta 指令 (写任意地址)。

◆ 刷新状态判断

Cache 控制寄存器的 14、15 位为 1 分别表示数据 Cache、指令 Cache 正在刷新, 为 0 表示刷新结束或没有刷新动作。

5.1.4.2 刷新步骤

(1) 关闭 Cache;

(2) 刷新 Cache;

(3) 判断 Cache 刷新状态, 等待刷新结束;

(4) 打开 Cache。

5.2 引发 Trap0x1a 陷阱条件

(1) 当**外部存储器配置寄存器** MCFG1(0x8100-0000)的[11]位 (PROM write enable) 配置为“0”时, 表示 PROM 写操作不使能, 如果出现对 PROM 进行写操作, 则会产生 trap0x1a 陷阱。

(2) 当**写保护寄存器**配置成写保护模式时, 即**写保护寄存器** WPR1(0x8100-001C)、WPR2(0x8100-0020)的[31]位 (写保护使能) 配置为“1”时, 如果出现对写保护地址进行写操作, 则会产生 trap0x1a 陷阱。

(3) 当**外部存储器容错配置寄存器**配置为 EDAC 使能时, 即**外部存储器容错配置寄存器** MECFG1(0x8100-0100)的[31]位 (SRAM bank1 EDAC 使能)、[30]位 (SRAM bank2 EDAC 使能)、[29]位 (SRAM bank3 EDAC 使能)、[28]位 (SRAM bank4 EDAC 使能)、[27]位 (SRAM bank5 EDAC 使能)、[26]位 (PROM EDAC 使能)、[19]位 (SDRAM EDAC 使能) 配置为“1”时, 如果外部存储器访问地址中的数据有错误 (在纠错模式下有 2 位错误, 在检错模式下有 1 位或 2 为错误), 在字节写或半字写操作情况, 会产生 trap0x1a 陷阱。

(4) 当配置**外部存储器配置寄存器** MCFG1(0x8100-0000)的[25]位 bus error enable 为“1”

时，表示总线错误使能，在对外部存储器进行写操作时，如果系统输入管脚 bexc_n 为“0”时，会产生 trap0x1a 陷阱。

5.3 引发 Trap0x09 陷阱条件

1. 当外部存储器容错配置寄存器配置为 EDAC 使能时，即外部存储器容错配置寄存器 MECFG1 (0x81000100)的[31:27]位（各位分别是 SRAM bank1-bank5 的 EDAC 使能）、[26]位（PROM EDAC 使能）、[19]位（SDRAM EDAC 使能）配置为“1”时，如果外部存储器访问地址中的数据有错误(在纠错模式下有 2 位错误，在检错模式下有 1 位或 2 位错误)，在读操作情况，会产生 Trap0x09 陷阱。

2. 当配置外部存储器配置寄存器 MCFG1(0x81000000)的[25]位 bus error enable 为“1”时，表示总线错误使能。在对外部存储器进行读操作时，如果系统输入管脚 bexc_n 为“0”时，会产生 Trap0x09 陷阱。

3. 当 PCI 配置为主模式且 PCI 总线上没有 PCI 从设备时，对 AHB-PCI 地址空间进行读操作，即 AHB-PCI 不可预取地址空间、AHB-PCI 可预取地址空间、AHB-PCI I/O 地址空间，会产生 trap09 陷阱。

附录 6 SPARC V8 介绍

6.1 SPARC V8 概述

SPARC 是一种精简指令集（RISC）指令集架构。根据性能、价格等不同需求，SPARC 体系架构有不同的实现方式。SPARC 的设计目标是优化编译器、简化流水线的硬件实现。

SPARC 架构产品的优势在于执行速度和开发周期。SPARC 主要包括下面几个特性：

- 32 位线性地址空间，理论上可寻址 4GB 空间；
- 简洁的指令格式：均为 32 位宽，在内存中以 32 位对齐存储，诸多的指令只有三种存储格式，不同的存储格式其操作码和寄存器地址的存放位置固定；
- 只有 load 和 store 命令可以访问内存；
- 少数几种寻址模式：寄存器+寄存器、寄存器+立即数；
- 三寄存器寻址：一般都是将两个寄存器中的内容作相应运算后将结果放到第三个寄存器中；
- 一个大的窗口寄存器文件：在任一时刻，每个程序可以访问 8 个全局整数寄存器和一个由 24 个寄存器组成的寄存器窗口。所谓的窗口寄存器可以理解为存储着进程参数、局部变量和返回地址的存储区；
- 独立的浮点寄存器文件：可由软件设置成单（32 位）、双（64 位）、四（128 位）三种不同精度；
- 延迟控制转移特性：指示 CPU 延迟一条指令执行或跳过下一条指令，由控制转移指令的“annul”位决定执行还是跳过；
- 快速的陷阱处理：陷阱的处理存放在向量表中，陷阱处理时将旋转到下一个新的寄存器窗口。

6.2 数据类型

SPARC 架构可以识别三种基本的数据类型（大端存储模式）：

有符号整数：8、16、32、64 位长（最高位为符号位）。

无符号整数：8、16、32、64 位长。

浮点数：32、64、128 位长。

整数和浮点数的存储格式分别如图 A6-2-1 和图 A6-2-2 所示。

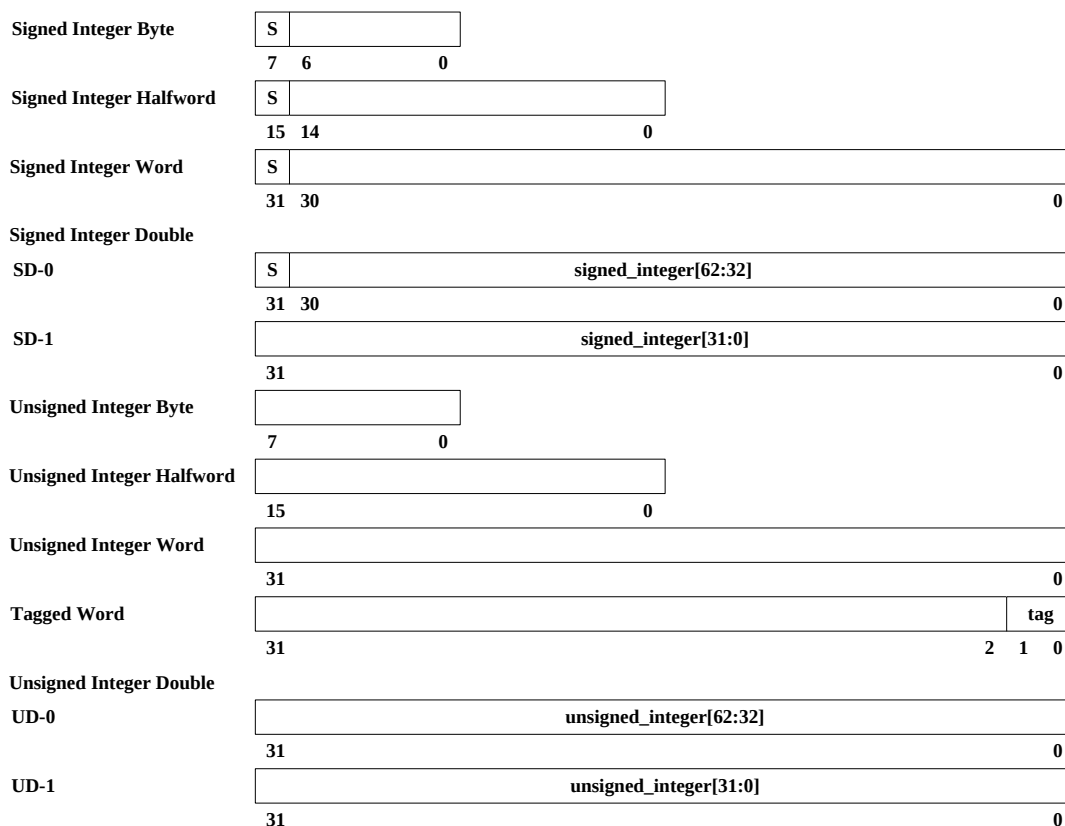


图 A6-2-1 整数存储格式

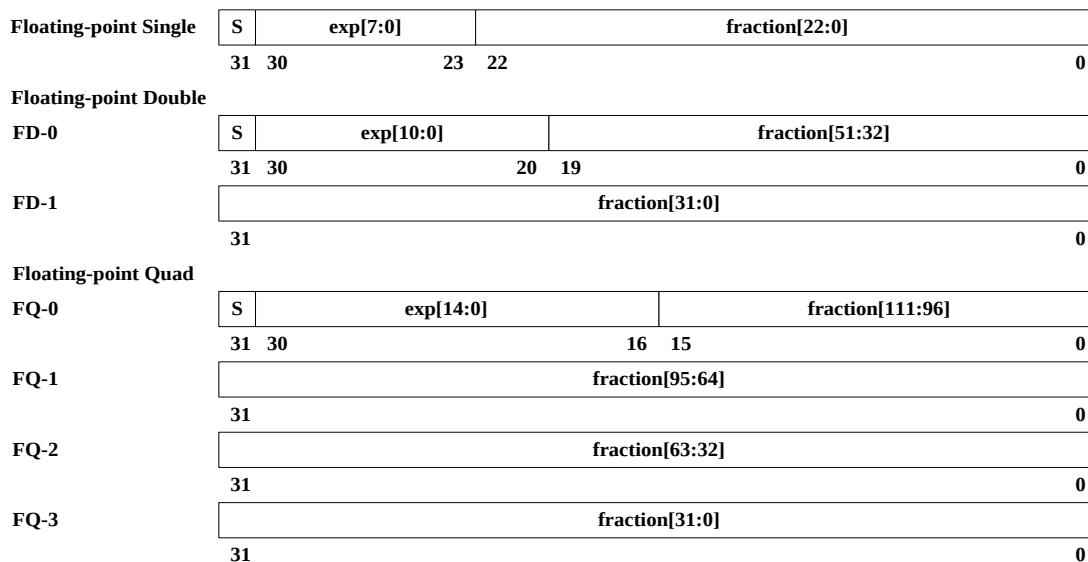


图 A6-2-2 浮点数存储格式

6.3 寄存器

处理器的寄存器包括两种类型：通用寄存器（或叫工作区数据寄存器）和控制/状态寄

寄存器。

通用寄存器：IU 和 FPU 的通用寄存器分别叫 r 寄存器和 f 寄存器。

控制状态寄存器：IU 的控制状态寄存器包括**处理器状态寄存器 PSR**、**窗口无效屏蔽寄存器 WIM**、**陷阱基址寄存器 TBR**、**乘/除寄存器 Y**、**程序计数器 PC/nPC** 和**辅助状态寄存器 ASR**。

FPU 的控制/状态寄存器是**浮点处理器状态寄存器 FSR**。

6.3.1 IU 寄存器

IU 包括 40 到 520 个通用 32 位 r 寄存器，这些 r 寄存器被分成 8 个全局寄存器，一系列由 16 个寄存器组成的寄存器组，这 16 个寄存器又可分成 8 个输入寄存器和 8 个局部寄存器。

Windowed Register Address	r Register Address
in[0] – in[7]	r[24] – r[31]
local[0] – local[7]	r[16] – r[23]
out[0] – out[7]	r[8] – r[15]
global[0] – global[7]	r[0] – r[7]

图 A6-3-1 窗口编址

窗口 r 寄存器：任一时刻，一条指令可以访问 8 个全局寄存器和 24 个窗口寄存器，每个寄存器窗口由 8 个输入寄存器、8 个局部寄存器和共享相邻寄存器组的 8 个输入寄存器（作为本窗口的输出寄存器）组成。当前窗口由 CWP 指针指定。

窗口的重叠：窗口的重叠指定是当前的窗口可以访问 CWP+1 窗口的输出寄存器，作为本窗口的输入寄存器；当前窗口的输出寄存器可以作为 CWP-1 窗口的输入寄存器，每个窗口的 8 个 local 寄存器是唯一的。窗口按一个环顺序排列，也就是说最后一个窗口指针加 1 指向 0 窗口。另外 save 命令将使 CWP 减 1，restore 命令将使 CWP 加 1。

特殊 r 寄存器：有四个特殊的 r 寄存器的用法是固定的：（1）如果 r[0] 作为源操作数或是作为 store 的目的地址，那么 r[0] 将被当作 0 来读写，如果 r[0] 作为目的操作数，那么相应的操作将被忽略。（2）call 指令将把自身的地址写进 r[15] 寄存器。（3）陷阱发生的时候：PC 和 nPC 中的内容将会自动复制到 r[17] 和 r[18] 当中。

6.3.2 FPU 寄存器

FPU 总共包含 32 个 32 位浮点 f 寄存器，标号从 f[0] 到 f[31]，f 寄存器可以通过浮点数

操作指令访问。与 r 寄存器不同的是这 32 个 f 寄存器都可以访问。另外，一个 f 寄存器可以存放 1 个单精度浮点数，对于双精度和四精度的浮点数的存放，则分别需要两个和四个相邻的浮点寄存器。

6.3.3 寄存器推荐使用方法汇总

表 A6-3-1 寄存器推荐使用方法汇总表

in	%i7	(%r31)	返回地址 - 8
	%fp, %i6	(%r30)	堆指针
	%i5	(%r29)	输入参数 6
	%i4	(%r28)	输入参数 5
	%i3	(%r27)	输入参数 4
	%i2	(%r26)	输入参数 3
	%i1	(%r25)	输入参数 2
	%i0	(%r24)	输入参数 1 / 向调用函数返回的值
local	%l7	(%r23)	本地变量 7
	%l6	(%r22)	本地变量 6
	%l5	(%r21)	本地变量 5
	%l4	(%r20)	本地变量 4
	%l3	(%r19)	本地变量 3
	%l2	(%r18)	本地变量 2
	%l1	(%r17)	本地变量 1
	%l0	(%r16)	本地变量 0
out	%o7	(%r15)	临时值 / CALL 指令的地址
	%sp, %o6	(%r14)	栈指针
	%o5	(%r13)	输出参数 6
	%o4	(%r12)	输出参数 5
	%o3	(%r11)	输出参数 4
	%o2	(%r10)	输出参数 3
	%o1	(%r9)	输出参数 2
	%o0	(%r8)	输出参数 1 / 从被调函数返回的值
global	%g7	(%r7)	全局变量 7 (SPARC ABI: 保留)
	%g6	(%r6)	全局变量 6 (SPARC ABI: 保留)

	%g5	(%r5)	全局变量 5 (SPARC ABI: 保留)
	%g4	(%r4)	全局变量 4 (SPARC ABI: 全局 reg 型变量)
	%g3	(%r3)	全局变量 3 (SPARC ABI: 全局 reg 型变量)
	%g2	(%r2)	全局变量 2 (SPARC ABI: 全局 reg 型变量)
	%g1	(%r1)	临时值
	%g0	(%r0)	0
state	%y		Y 寄存器 (用于乘/除法)
	(icc field of %psr)		整数状态位
	(fcc field of %fsr)		浮点数状态位
floating point	%f31		浮点值
	:		:
	:		:
	%f0		浮点值

6.4 指令

指令由 CPU 从内存中读取并执行，PC 和 nPC 分别提供当前指令的地址和下一条指令的地址。指令被编码成三种格式存储，按功能可以分成 6 类，有 72 条基本的指令操作。

6.4.1 指令格式

SPARC 架构指令存储的格式可以分成如下三种，各位的含义如下图 A6-4-1 所标。

Format 1(op=1): CALL

op	disp30																													
31	29																													0

Format 2(op=0): SETHI&Branches(Bicc,FBfcc,CBccc)

op	rd		op2	imm22																									
op	a	cond	op2	disp22																									
31	29	28	24	21																									0

Format 3(op=2 or 3): Remaining instructions

op	rd	op3	rs1	i=0	asi	rs2
op	rd	op3	rs1	i=1	simm13	
op	rd	op3	rs1	opf		rs2
31	29	24	18	13	12	4
0						

图 A6-4-1 指令存储位含义

6.4.2 指令分类

SPARC 指令集可以分成 6 类，分别为：

1) 加载/存储指令 load/store:

唯一可以访问内存的指令。加载/存储指令利用两个 *r* 寄存器或是一个 *r* 寄存器和一个符号扩展的 13 位立即数来计算字节对齐的 32 位内存地址。

2) 整数算术指令:

该类指令通常都是三寄存器地址指令, 即对两个寄存器的数据进行运算。然后将结果放到第三个寄存器中。源操作数一个恒为 *r* 寄存器的数据, 另一个源操作数依赖于指令的第 *i* 位, 可能为 *r* 寄存器的数据 (*i* 位为 0) 也可能为符号扩展的 13 位立即数 (*i* 位为 1)。

3) 控制传输指令

该类指令修改 *nPC* 的值, 也就是改变程序的执行路线。有五类基本的控制转移命令类型:

条件转移 *Bicc*、*FBfcc*

调用 *call* and *link*

跳转 *jump* and *link*

陷阱返回 *RETI*

陷阱 *Ticc*

4) 读写控制寄存器指令

该类指令访问程序可见的状态寄存器, 将状态寄存器的值读入 *r* 寄存器或是将 *r* 寄存器的值写入状态寄存器。对于读/写辅助状态寄存器的指令, 如果访问的寄存器是特权寄存器, 那么该条指令将是特权指令。

5) 浮点数操作指令

除了浮点数比较指令 (修改 *FSR* 的 *fcc* 位) 和浮点数交换命令 (利用一个源操作数和一个目标操作数), 其他的浮点操作数指令基本也都是三操作数指令。所有的浮点操作数指令 (*Fpop*) 都清除 *FTT* 位、置位 *CEXC*。

6) 协处理器操作

协处理器操作指令由附加的协处理器执行。

6.5 陷阱

6.5.1 精确陷阱

精确陷阱由特定的指令产生, 在程序状态被该指令修改之前由 *CPU* 捕获, 当精确陷阱

发生的时候，有几个条件或是状态必须要保证和保持：

- a) PC 指向产生该陷阱的指令，nPC 指向捕获该陷阱后 CPU 将要执行的指令地址；
- b) 引起陷阱的指令之前的一条指令将完全执行；
- c) 产生该陷阱的指令的后面一条指令不执行。

6.5.2 中断陷阱

中断陷阱受 **PSR** 的处理器中断优先级 **PIL** 位和陷阱使能 **ET** 位的联合控制。中断陷阱的产生可能是以下的原因造成：

- a) 外部中断请求；
- b) 与之前指令无关的异常；
- c) 先前执行指令引起的异常。

6.5.3 延迟陷阱

延迟陷阱也是由特定的指令产生的，与精确陷阱不同的是，延迟陷阱的发生可能在程序可见的状态改变之后。

6.5.4 陷阱控制

1. ET 和 PIL 控制

陷阱要正常工作，**PSR** 中的 **ET** 位必须为 1。

当 **ET=1** 时，**IU** 根据陷阱表区分陷阱的优先次序，只响应最高优先级的陷阱。对中断响应来说，**IU** 比较中断请求等级与 **PSR** 中 **PIL** 字段。如果中断请求等级大于或等于 **PIL** 字段值，且未屏蔽此中断，则处理器响应该中断请求。当处理最高优先级中断时，低优先级中断请求保持在中断请求寄存器中。

当 **ET=0** 时，如果发生陷阱，处理器进入错误模式。

2. TEM 控制

如果 **TEM** 字段的特定位置 1，与之对应的浮点异常发生时，允许引起浮点异常陷阱 (**fp_exception**)。

如果 **TEM** 字段的特定位置 0，不允许引起浮点异常陷阱。此时，产生的异常会被记录在 **FSR** 的“产生浮点异常”字段中 (**AEXC**)。

如果没有屏蔽浮点异常陷阱，则引起浮点陷阱时，被中断执行的指令的目的寄存器、**FCC**

和 AEXC 字段保持不变。如果浮点异常被屏蔽，引起陷阱的指令的目的寄存器的 FCC 和 AEXC 字段被更新。

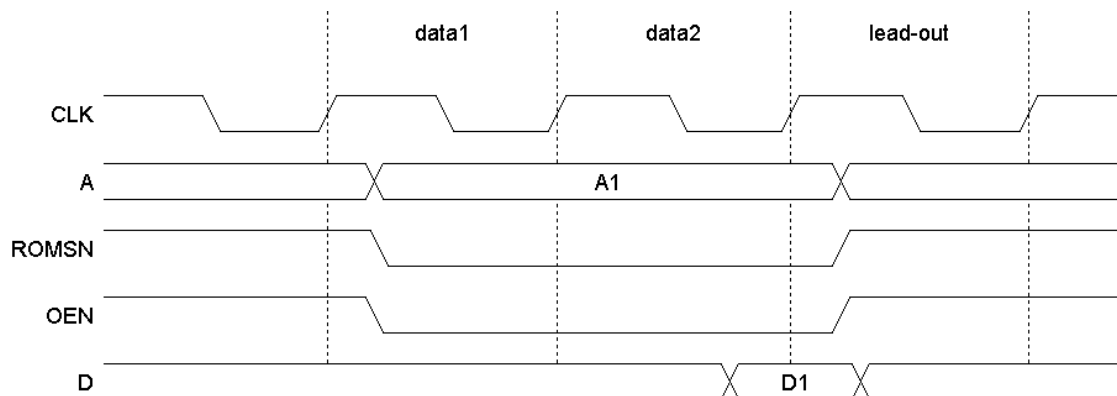
6.5.5 陷阱识别

陷阱基地址寄存器 **TBR** 中的“陷阱基地址”字段 **TBA** 保存陷阱向量表基地址的高 20 位。

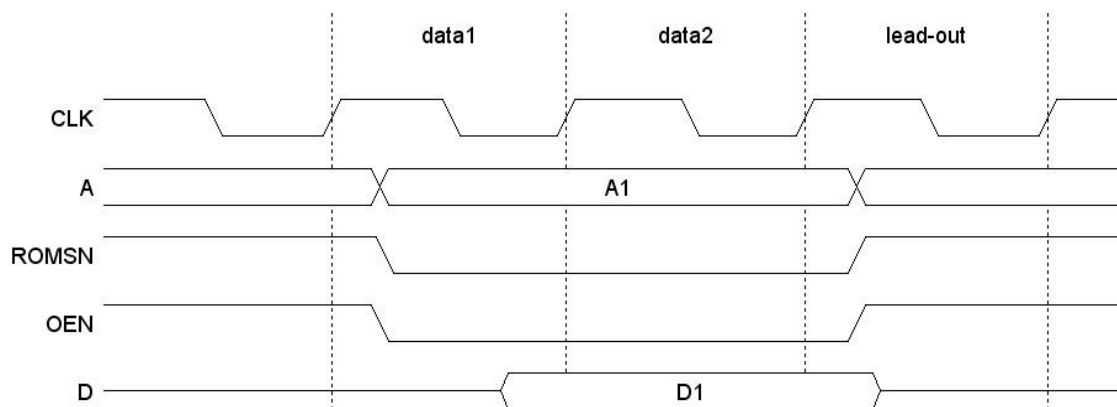
当陷阱发生时，与之唯一对应的识别码写入 **TBR** 的 8 位 **TT** 字段，程序以 **TBR** 的值为目的地址进行跳转。因为 **TBR** 的低四位为 0，因此陷阱向量表的每一入口包含中断处理程序的前 4 条指令（16 个字节）。

附录 7 PROM、SRAM、IO、SDRAM 访问时序

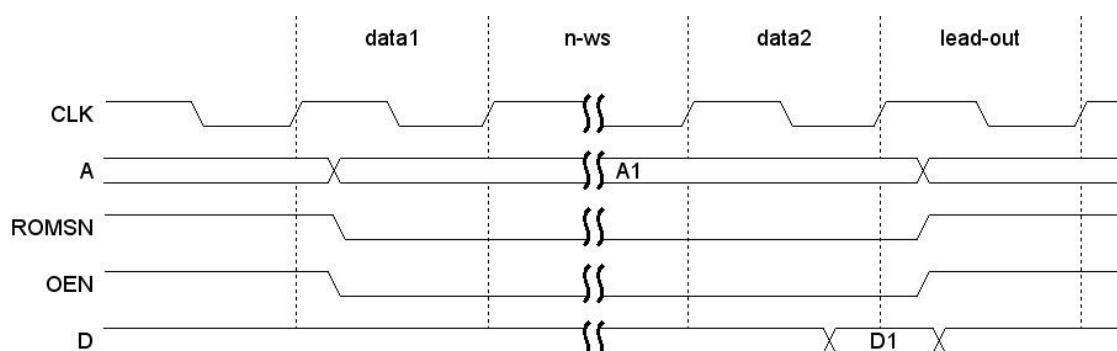
1. PROM 读时序（EDAC 不使能，0 等待周期）



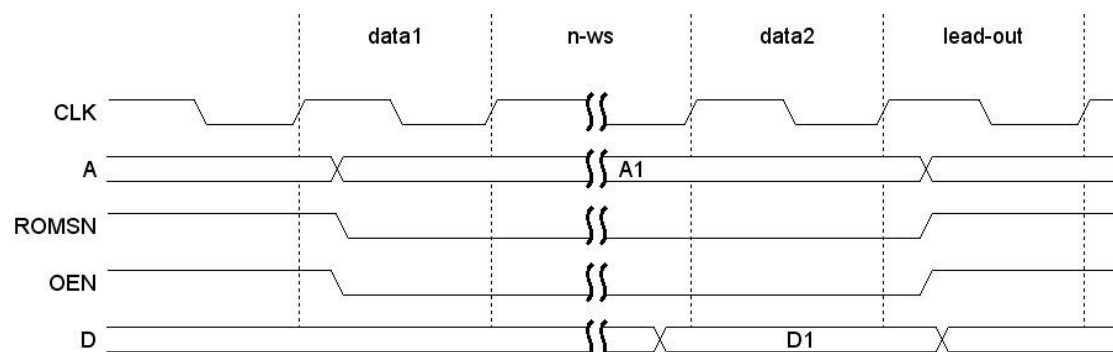
2. PROM 读时序（EDAC 使能，0 等待周期）



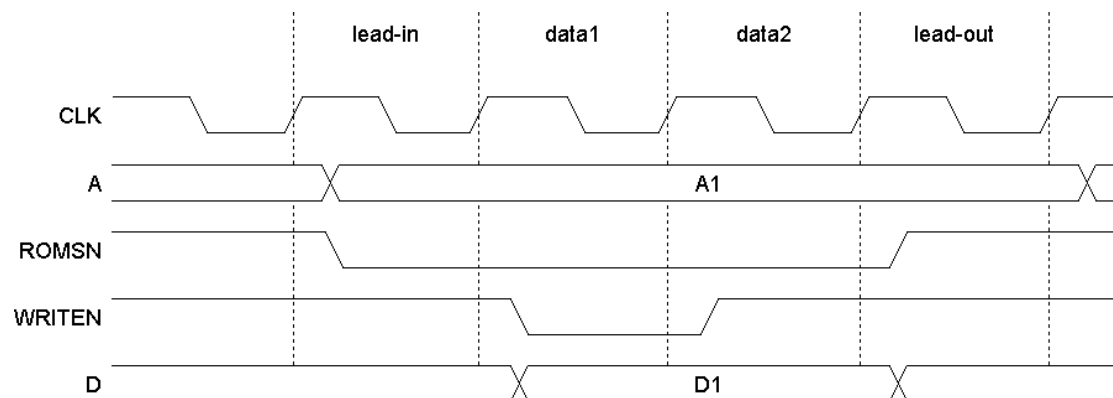
3. PROM 读时序（EDAC 不使能，n 个等待周期）



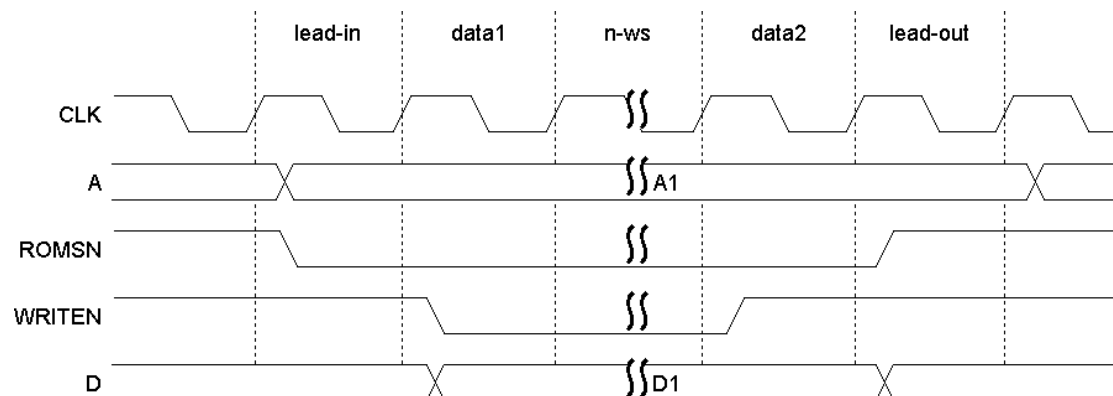
4. PROM 读时序 (EDAC 使能, n 个等待周期)



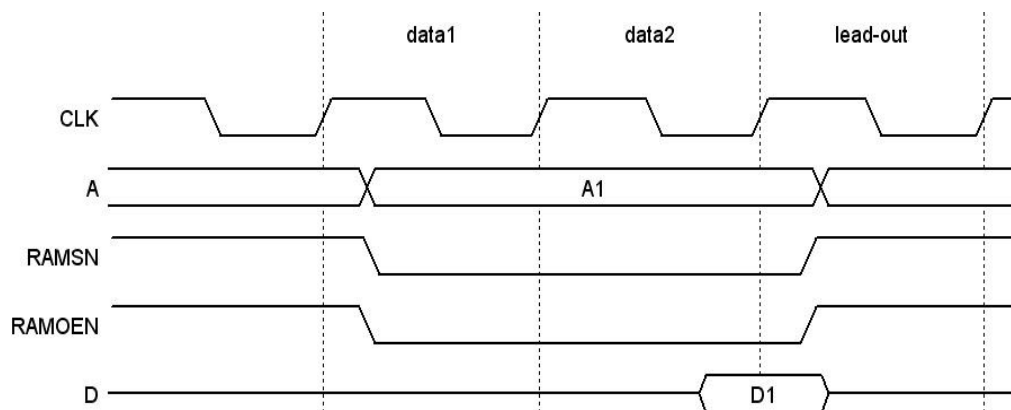
5. PROM 写时序 (0 个等待周期)



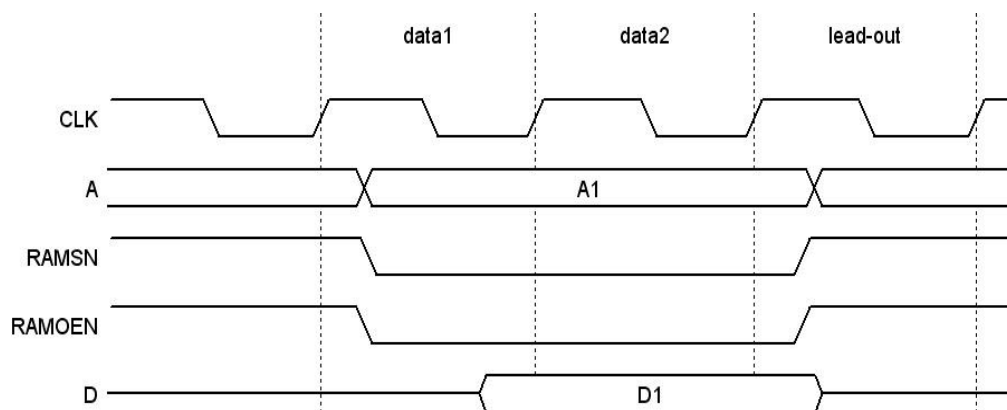
6. PROM 写时序 (n 个等待周期)



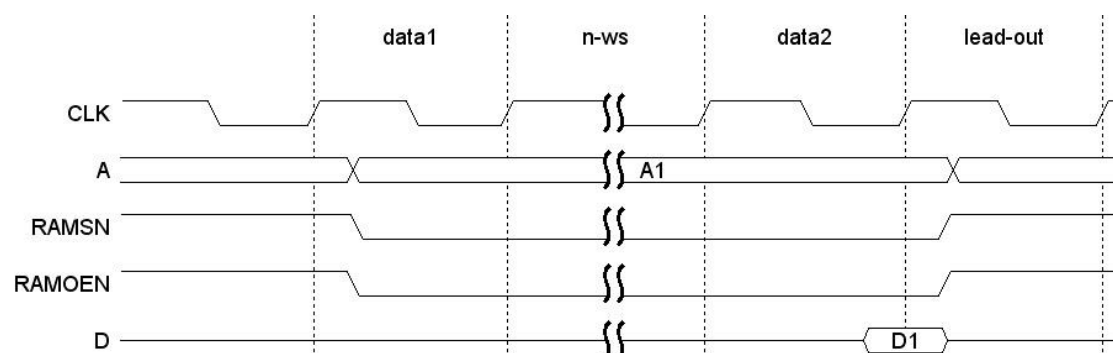
7. SRAM 读访问时序（EDAC 不使能，0 等待周期）



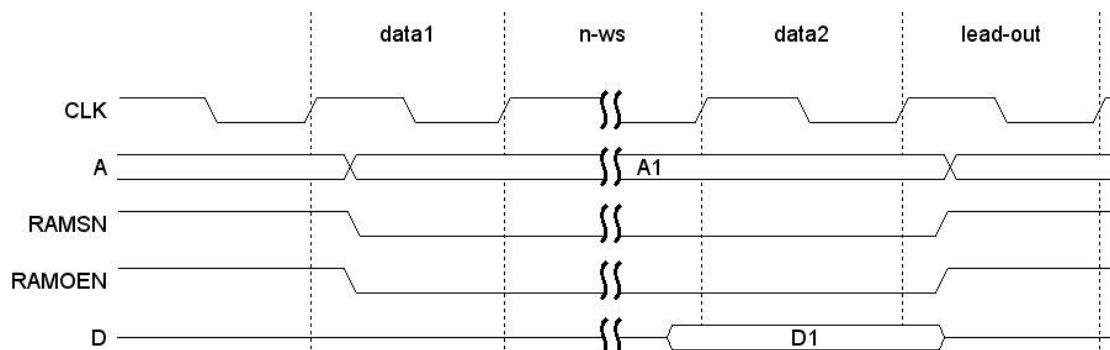
8. SRAM 读访问时序（EDAC 使能，0 等待周期）



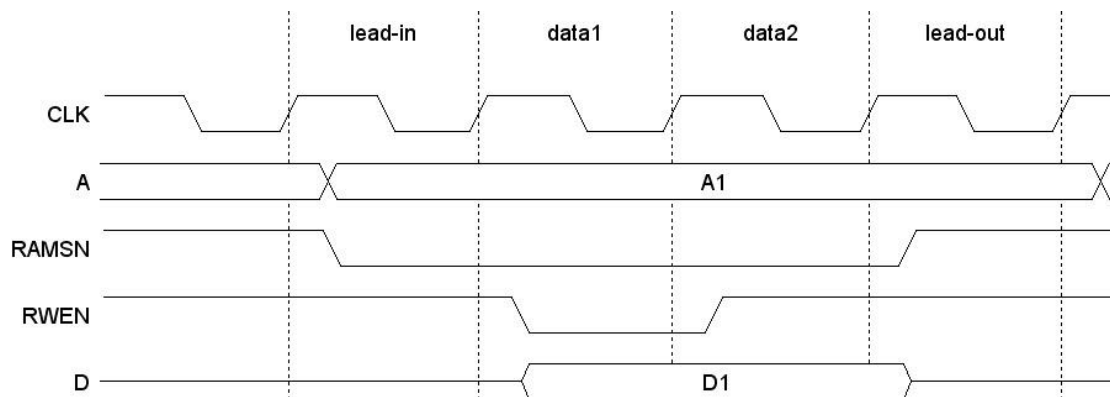
9. SRAM 读访问时序（EDAC 不使能，n 等待周期）



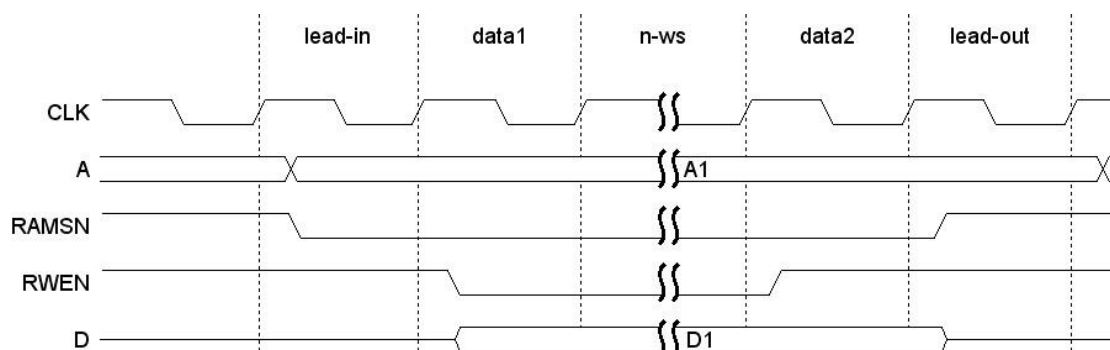
10. SRAM 读访问时序 (EDAC 使能, n 等待周期)



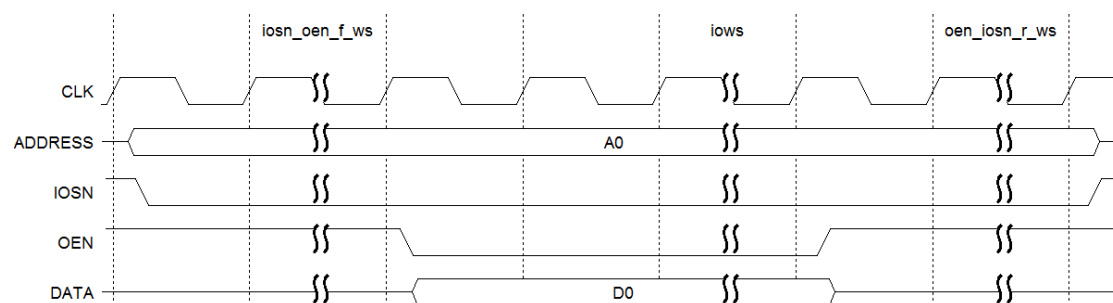
11. SRAM 写访问时序 (0 等待周期)



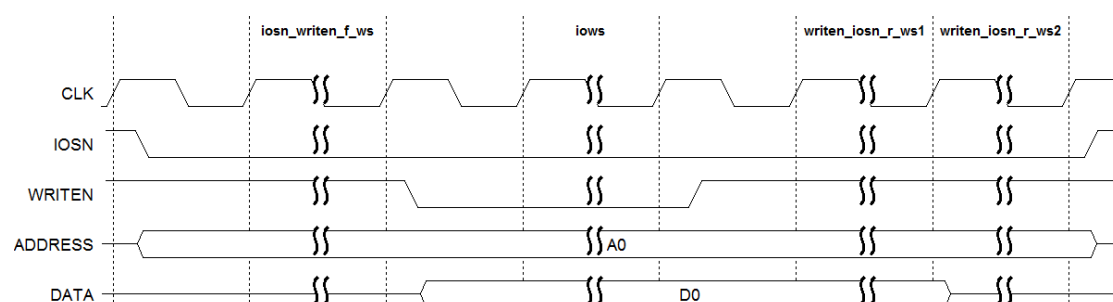
12. SRAM 写访问时序 (n 等待周期)



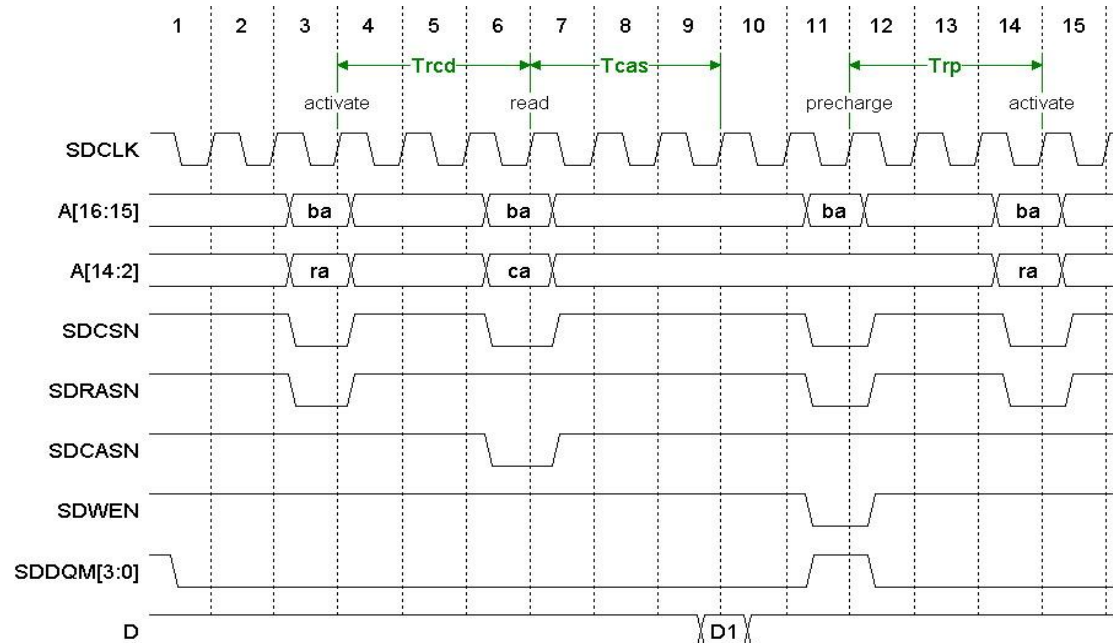
13. I/O 读访问时序 (n 等待周期)



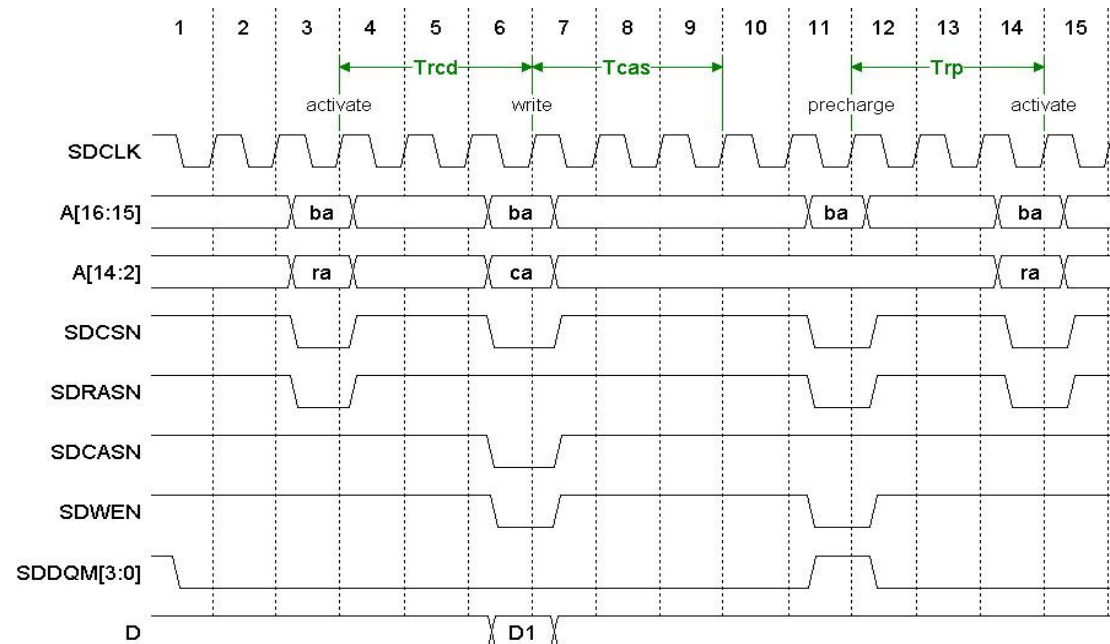
14. I/O 写访问时序 (n 等待周期)



15. SDRAM 读时序 (Burst length = 1; CL = 3)



16. SDRAM 写时序 (Burst length = 1; CL = 3)



附录 8 与 BM3803FMGRH 差异性

序号	描述	BM3803FMGRH	BM3823AMCCRH	备注
1	频率提升	100MHz	300MHz	3823A 主频提升 3 倍
2	内部 RAM	无	有	BM3823A 增加了 256KB 的内部 RAM
3	MMU	无	有	BM3823A 增加了 MMU 功能
4	DMA 控制器	无	有	BM3823A 增加了 DMA 控制器
5	DDR2 存储器控制器	无	有	BM3823A 增加了 DDR2 存储器控制器
6	I ² C	无	有	BM3823A 增加了 I ² C 总线控制器
7	SPI	无	有	BM3823A 增加了 SPI 总线控制器
8	Ethernet MAC	无	有	BM3823A 增加了 10/100/1000Mbps 的 Ethernet MAC
9	UART	3 路	4 路	BM3803F 是 3 路不带 FIFO 串口, BM3823A 是 2 路不带 FIFO 串口和 2 路带 FIFO 的串口
10	中断控制器	1 级中断	2 级中断	BM3823A 有 2 级中断, 1 级中断 15 个, 同时扩展了 20 个 2 级中断
11	外存地址空间	2GB 外存空间	1GB 外存空间	BM3823A 的外存空间压缩了一半, 剩余 1GB 留给了 DDR2 SDRAM
12	PCI 总线控制器	无 pcinouse	有 pcinouse	BM3823A 增加了 pcinouse 管脚, 提高了 PCI 不使用时管脚处理的灵活性
13	封装	CPGA391	CCGA717	—

14	电压域	2 个电压阈： IO 电压：3.3V 内核电压：1.8V	3 个电压阈： IO 电压：3.3V DDR2 IO 电压： 1.8V 内核电压：1.2V	—
15	功耗	1W	小于 4W	—
16	外形尺寸	47mm×47mm	35mm×35mm	—
17	抗辐射指标	单粒子功能在轨 错误率（GEO 轨 道）：优于 8×10^{-5} 次/天•器件	单粒子功能在轨错 误率（GEO 轨道）： 优于 5×10^{-5} 次/天• 器件	—

附录 9 PCI、DDR2 及 MAC 不使用时相关管脚接法说明

9.1 PCI 不使用时管脚连接说明

BM3823A 的 PCI 支持主从两种模式，为确保 PCI 功能不对 BM3823A 其余功能正常使用及长期可靠性造成影响，PCI 模块在不使用时，必须保证 PCI 所有输入都上拉或下拉，pcinouse 管脚接高，避免管脚悬空产生多余功耗和干扰，避免对长期使用可靠性产生影响。不使用 PCI 时相关管脚连接要求如表 A9-1-1 所示。

表 A9-1-1 PCI 不使用时接法

管脚名称	不使用时连接方法
pcinouse	上拉（通过 1K Ω 电阻接 3.3V 电源）
pci_ad[31:0]	悬空
pci_arb_gnt_n[6:0]	悬空
pci_host	上拉（通过 1K Ω 电阻接 3.3V 电源）
pci_arb_req_n[6:0]	上拉（通过 4.7K Ω 电阻接 3.3V 电源）
pci_cbe_n[3:0]	悬空
pci_clk_in	上拉或下拉（通过 4.7K Ω 电阻接 3.3V 电源或 GND）
pci_devsel_n	悬空
pci_frame_n	悬空
pci_gnt_in_n	上拉（通过 4.7K Ω 电阻接 3.3V 电源）
pci_idsel_in	下拉（通过 4.7K Ω 电阻接 GND）
pci_inta_n	悬空
pci_intb_n	上拉（通过 4.7K Ω 电阻接 3.3V 电源）
pci_intc_n	上拉（通过 4.7K Ω 电阻接 3.3V 电源）
pci_intd_n	上拉（通过 4.7K Ω 电阻接 3.3V 电源）
pci_irdy_n	悬空
pci_par	悬空
pci_perr_n	悬空
pci_req_n	悬空
pci_rst_n	悬空
pci_serr_n	悬空
sim_fast_reset	下拉（通过 1K Ω 电阻接 GND）
pci_stop_n	悬空
pci_trdy_n	悬空

9.2 DDR2 不使用时管脚连接说明

为确保 DDR2 功能不对 BM3823A 其余功能正常使用及长期可靠性造成影响，DDR2 模块在不使用时，必须保证 DDR2 所有输入都上拉或下拉，避免管脚悬空产生多余功耗和干扰，避免对长期使用可靠性产生影响。然而 DDR2 很多双向管脚，为了保证 DDR2 的双向管脚处于输出状态，不使用 DDR2 时相关管脚连接要求如表 A9-2-1 所示。

表 A9-2-1 DDR2 不使用时接法

管脚名称	不使用时接法
pll_vddpost_ddr	接数字 1.2V
vdd_pre、vdd_io	数字 1.8V
vref_0、ddr_add_vref	数字 0.9V，如果板级设计没有 0.9V，则通过 4.7K Ω 电阻下拉
vref_vss_0、vref_vss_1、vss_pre、vss_io	接 GND
pll_pd_ddr	下拉（通过 4.7K Ω 电阻接 GND）
ddr_pll[2:0]	接非全 0，推荐接“001~101”
pad_mem_address[14:0]	悬空
pad_mem_bank[2:0]	悬空
pad_mem_data[31:0]	悬空
pad_mem_dm[3:0]	悬空
pad_mem_dm[3:0]	悬空
pad_mem_dqs_m[3:0]	悬空
pad_mem_dqs_p[3:0]	悬空
pad_mem_ecc[3:0]	悬空
pad_mem_ecc_dm	悬空
pad_mem_ecc_dqs_m	悬空
pad_mem_ecc_dqs_p	悬空
pad_mem_cke	悬空
pad_mem_clk[1:0]	悬空
pad_mem_clk_n[1:0]	悬空
pad_mem_cs_n[1:0]	悬空
pad_mem_cas_n	悬空
pad_mem_ras_n	悬空
pad_mem_we_n	悬空
pad_mem_odt[1:0]	悬空

9.3 MAC 不使用时管脚连接说明

为确保 MAC 功能不对 BM3823A 其余功能正常使用及长期可靠性造成影响，MAC 模块在不使用时，必须保证 MAC 所有输入都上拉或下拉，避免管脚悬空产生多余功耗和干扰，避免对长期使用可靠性产生影响。不使用 MAC 时相关管脚连接要求如表 A9-3-1 所示。

表 A9-3-1 MAC 不使用时接法

管脚名称	不使用时接法
pll_vddpost_mac	接数字 1.2V
pll_pd_mac	上拉（通过 4.7K Ω 电阻接 3.3V）
gmac_clk_pmarx0	下拉（通过 4.7K Ω 电阻接 GND）
gmac_clk_pmarx1	下拉（通过 4.7K Ω 电阻接 GND）
gmac_clk_rmii	悬空
gmac_clk_tx_125	悬空
gmac_clk_tx	悬空
gmac_intf_sel[2:0]	下拉（通过 1K Ω 电阻接 GND）
gmac_mdc	悬空
gmac_mdio	下拉（通过 4.7K Ω 电阻接 GND）
gmac_pcs_aquired_sync_o	悬空
gmac_pcs_en_cdet_o	悬空
gmac_pcs_ewrap_o	悬空
gmac_pcs_lck_ref_o	悬空
gmac_phy_clk_rx	下拉（通过 4.7K Ω 电阻接 GND）
gmac_phy_clk_tx	下拉（通过 4.7K Ω 电阻接 GND）
gmac_phy_col_i	下拉（通过 4.7K Ω 电阻接 GND）
gmac_phy_crs_i	下拉（通过 4.7K Ω 电阻接 GND）
gmac_phy_rxd_i[7:0]	下拉（通过 4.7K Ω 电阻接 GND）
gmac_phy_rxdv_i	下拉（通过 4.7K Ω 电阻接 GND）
gmac_phy_rxer_i	下拉（通过 4.7K Ω 电阻接 GND）
gmac_phy_txd_o[7:0]	悬空
gmac_phy_txen_o	悬空
gmac_phy_txer_o	悬空
gmac_ptp_aux_ts_trig_i	下拉（通过 4.7K Ω 电阻接 GND）
gmac_tbi_sigdet_i	下拉（通过 4.7K Ω 电阻接 GND）