龙芯 2E 北桥用户手册

2007-5-24

版权声明

目录

第一章 概述

1.1 简介

整个北桥是为龙芯 2E 处理器专门设计的，为基于龙芯 2E 处理器的系统提供解决方案。非常适合用于高性能、低功耗的产品设计。

片内系统总线采用 WishBone 总线标准，其数据宽度为 64 位，数据吞吐量很高。

Linux 操作系统运行稳定。

1.2 特征

► 处理器接口

- ♠ 支持 64 位 SysAD 接口标准
- ♠ 采用高效的传输分离技术
- ♠ 支持突发传输
- ♠ 缺省总线频率为 66MHz
- ♠ 支持龙芯 2E 的复位时序

► PCI 接口

- ♠ 符合 PCI2.2 版本规范
- ♠ 32 位/33MHz 总线接口
- ♠ PCI 仲裁支持 8 个主设备

► Local IO 接口

- ♠ 支持 8 位/16 位数据总线
- ♠ 支持 19 位地址总线（可扩展到 26 位）
- ♠ 1M 启动 ROM 存储空间
- ♠ 可扩展 60M 的 ROM 存储空间
- ♠ 可提供 4 个 IO 片选和 2 个 ROM 片选

► GPIO 模块

- ♠ 集成 7 个 gpin 管脚
- ♠ gpin 一般用于输入中断源
- ♠ 集成 9 个 gpio 管脚

► 中断控制器

- ♠ 支持 32 个中断源
- ♠ 不支持优先级仲裁
- ♠ 中断输入可编程为电平有效或触发有效

► FPGA 实现

- ♠ 有利于系统调试
- ♠ 可扩展性非常强
- ♠ 可以调节 IO 标准

第二章 系统概述

2.1 北桥结构

北桥内部主要模块为：龙芯 2E 的接口（即 SysAD 接口和复位信号），Local IO 接口模块，中断控制器模块，内部寄存器模块以及 PCI Master 和 PCI target 模块，模块之间采用 WishBone 协议互连。如下图 2.1 系统框图所示：

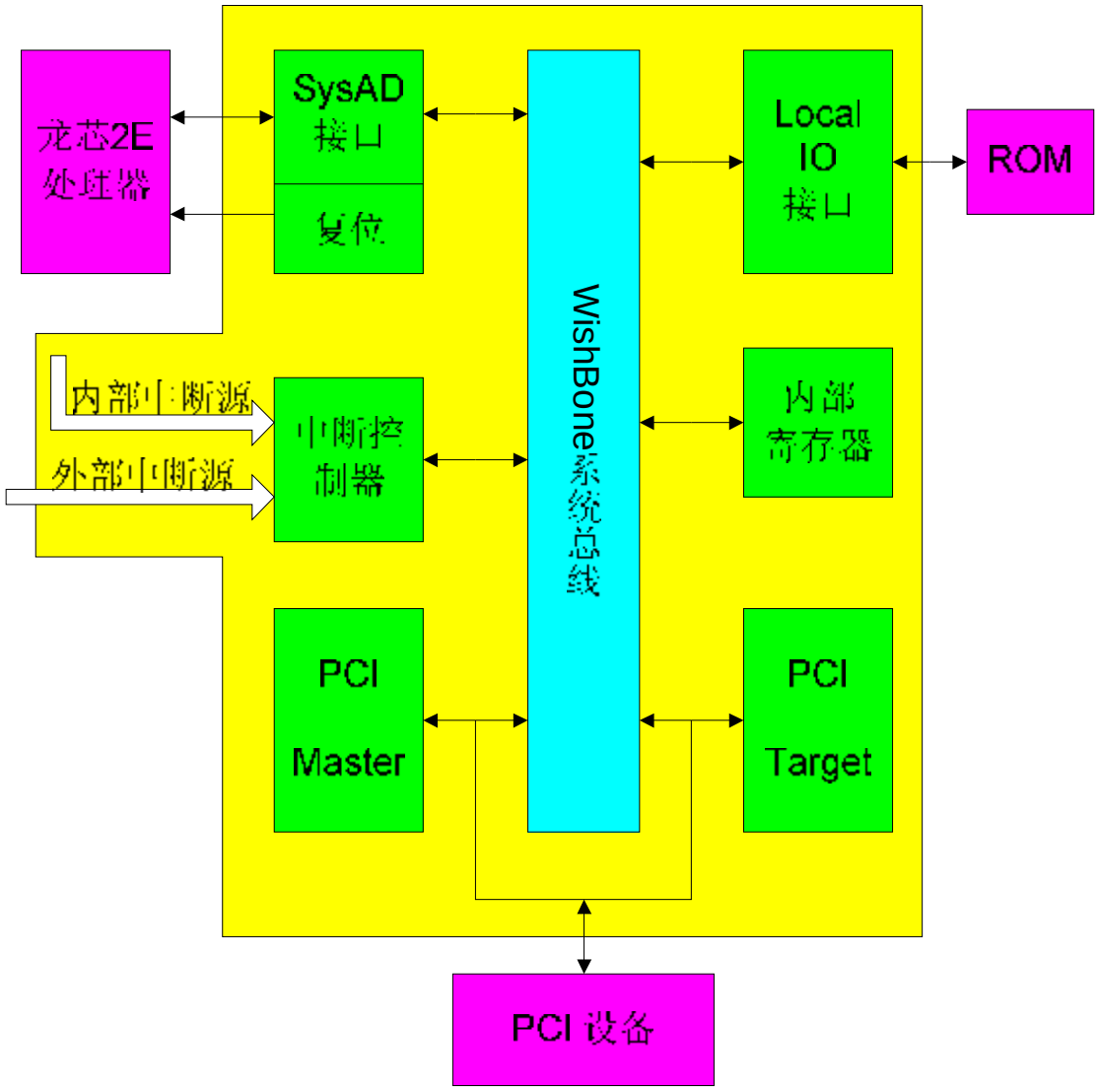


图 2.1 北桥结构框图

2.2 系统结构

由于龙芯 2E 处理器集成了 DDR 控制器，所以北桥不用再外挂内存。基于龙芯 2E 处理器和北桥的系统基本结构如下图所示：

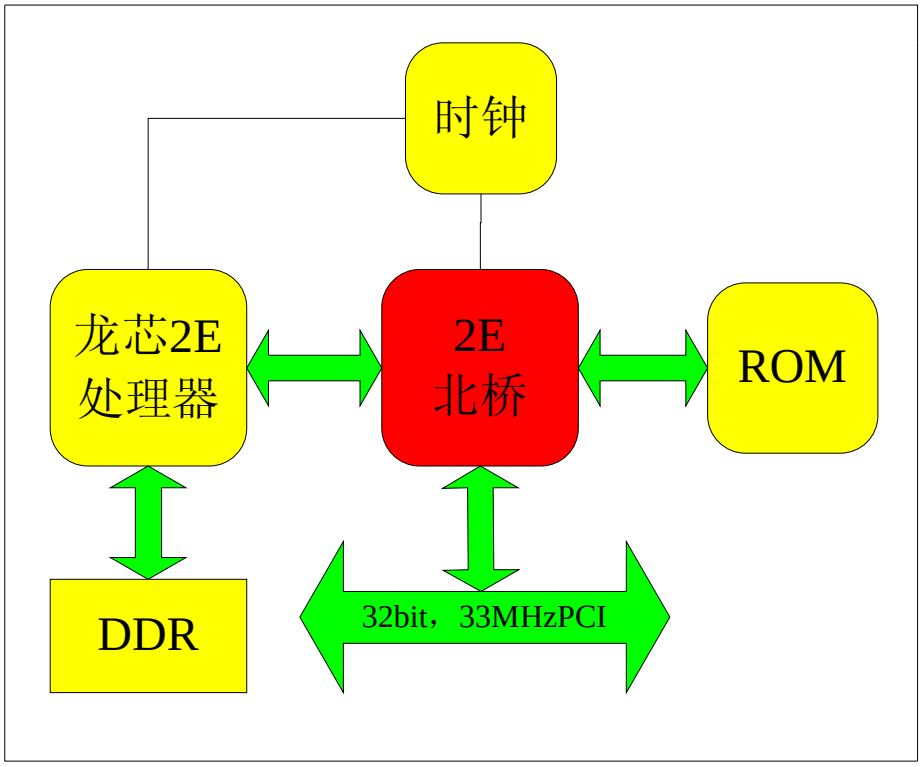


图 2.2 系统结构图

如上图所示，北桥处理的主要任务是：

（1）、管理 CPU 复位时序；

（2）、系统从 Local ROM 启动；

（3）、中断管理控制；

（4）、通用 I/O 口；

（5）、实现 PCI 接口；

（6）、实现 SysAD 接口；

2.3 地址空间

北桥将处理器地址空间分为 11 个部分，主要有内存空间、PCI 设备空间、IO 设备空间、ROM 空间以及寄存器空间五大类。详细的地址分配见下表：

表 2.3 地址空间分配

地址范围		大小/B	分类	功能
起始	0x0000_0000	256M	Memory	内存空间
结束	0x0FFF_FFFF			
起始	0x1000_0000	64M	PCI_Lo0	PCI 内存空间
结束	0x13FF_FFFF			
起始	0x1400_0000	64M	PCI_Lo1	
结束	0x17FF_FFFF			
起始	0x1800_0000	64M	PCI_Lo2	
结束	0x1BFF_FFFF			
起始	0x1C00_0000	60M	ROM	ROM 空间
结束	0x1FBF_FFFF			
起始	0x1FC0_0000	1M	Boot	启动地址空间
结束	0x1FCF_FFFF			
起始	0x1FD0_0000	1M	PCI_IO	PCI IO 空间
结束	0x1FDF_FFFF			
起始	0x1FE0_0000	256	2E-NB	北桥 PCI 配置空间
结束	0x1FE0_00FF			
起始	0x1FE0_0100	256	2E-NB	北桥内部寄存器
结束	0x1FE0_01FF			
起始	0x1FE8_0000	512K	PCI	PCI 设备配置空间
结束	0x1FEF_FFFF			
起始	0x1FF0_0000	1M	Local IO	IO 设备空间
结束	0x1FFF_FFFF			
起始	0x2000_0000	1.5G	PCI_1.5G	PCI 空间
结束	0x7FFF_FFFF			
起始	0x8000_0000	2G	PCI_2G	PCI 空间
结束	0xFFFF_FFFF			

需要注意的两点：

（1）：MIPS 指令集的启动地址为 0xBFC0_0000，龙芯 2E 也是类 MIPS 指令集的处理器，所以其启动地址向量也是 0xBFC0_0000。

（2）：由于龙芯 2E 集成了 DDR 控制器，所以访问低 256M 内存空间的时候，一般不会发送到北桥处理。

2.4 接口概述

龙芯 2E 处理器集成了 DDR 控制器，所以北桥不再包含内存控制器，这样北桥的主要接口就主要包括：处理器接口，PCI 接口，Local IO 接口，以及其他信号线。

2.4.1 处理器接口

处理器接口主要包括北桥与处理器之间数据传输的 64 位的 SysAD 总线接口以及处理器的复位信号和中断信号。

SysAD 总线是 MIPS 处理器常用的一种总线接口，其数据和地址复用，数据宽度为 64 位，地址宽度为 32 位（可扩展至 36 位）。主要负责北桥与处理器之间数据传输。在目前的北桥设计中，SysAD 总线缺省频率为 66MHz。其总的数据吞吐量可达到 4224Mb/s。

在由 SysAD 传输数据的同时，也采用 FIFO 缓冲实现了总线分离传输（Split transaction）技术。从而使得总线的效率更高。

处理器的复位信号和中断信号则分别由北桥内部的复位状态机和中断控制处理模块来提供。

2.4.2 PCI 接口

北桥实现的 PCI 总线的数据宽度为 32 位，频率为 33MHz，符合 PCI2.2 版本规范。北桥内部集成了 PCI 总线仲裁，支持 8 个 PCI 主设备的请求。片外 PCI 设备的 IO 空间，存储器空间，配置空间分别被映射到系统总线上的三段地址空间。（详细见系统地址分配）。

2.4.3 Local IO 接口

2E 北桥集成了 local I/O 总线，这是一套典型的 intel 风格的并行总线，有两个片选信号：nRomCS 和 nIOCS ，以及 8bit 数据，19bit 地址总线；支持存储空间高达 4Mb。

并行总线接口控制器是挂在北桥内部 WishBone 总线上的一个从设备。主要由挂在总线上的 CPU 接口端的 Wishbone 主设备访问。

2.4.3 GPIO

GPIO，通用输入输出口（**general-purpose I/O port**）。可根据需要将 GPIO 管脚编程为输出或输出。北桥集成了 7 个 GPIN 管脚和 9 个 GPIO 管脚。GPIN 用于通用输入。

第三章 信号定义

3.1 信号组图

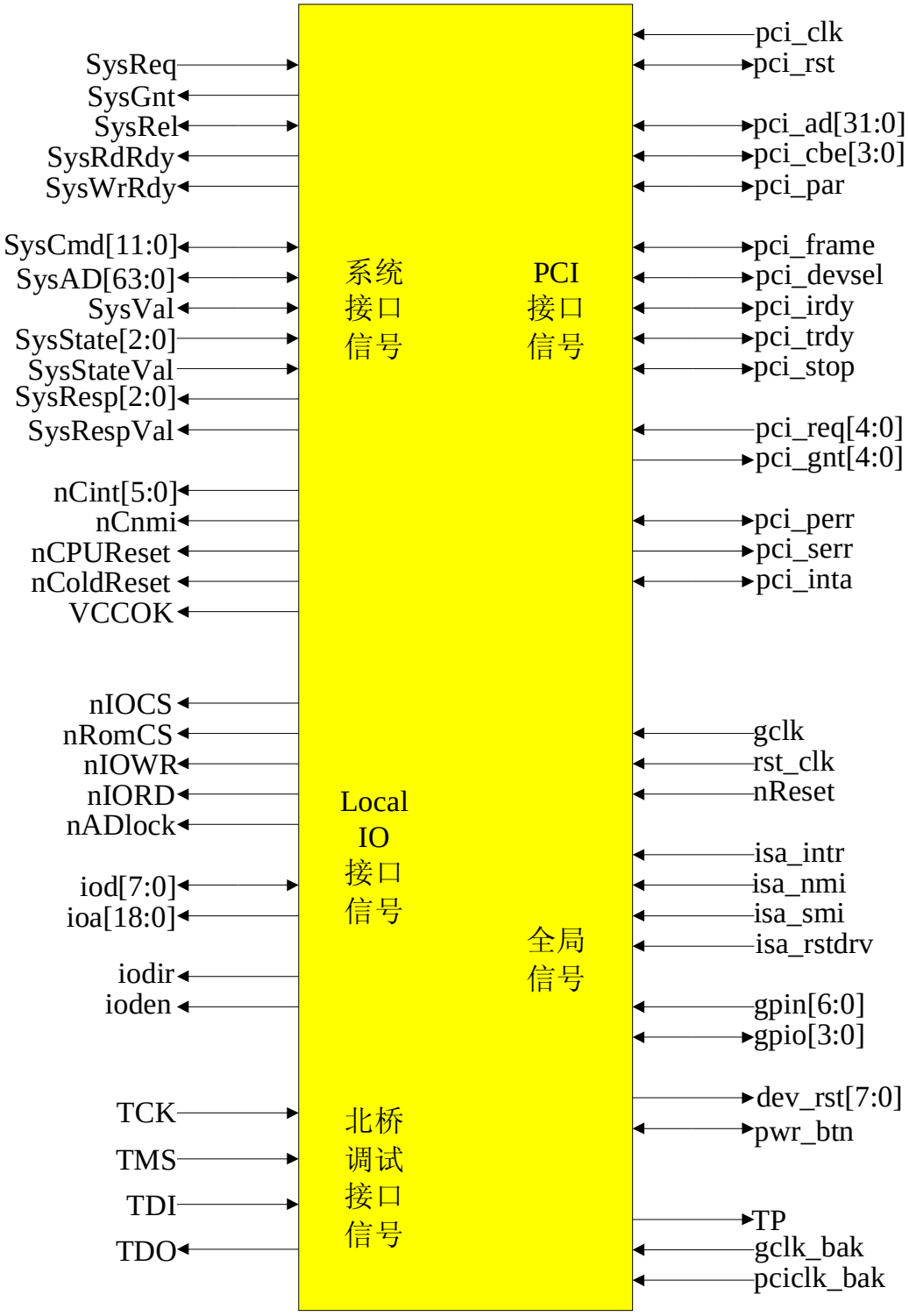


图 3.1 北桥信号组图

如图 3.1 所示，定义了北桥信号和描述了外部接口。主要分为 5 部分：系统接口，主要包括龙芯 2E 处理器 SYSAD 接口和处理器的复位信号。PCI 接口，北桥 PCI 既可以作为 PCI master，又可以作为 PCI target。Local IO 接口，以及全局信号；全局信号包括了 GPIN 信号，GPIO 信号，时钟，复位，中断等信号；此外还有北桥的 JTAG 调试接口。具体信号的描述见下一小节 3.2 信号描述

3.2 信号描述

3.2.1 处理器接口信号

表 3.2.1 处理器接口信号

管脚名称	位置	方向	电压标准	功能描述
SysAD [63:0]	D21,D19,E20,G17,F21,G21,J19,H17,H16,J17,J18,H18,R12,J15,R15,R16,P9,V14,U14,T16,AA15,T15,U15,AA14,R14,W15,W14,AA13,AB13,T12,V11,W11,AB20,Y17,AA18,AA19,AB17,AA17,AA20,AB19,V15,AB18,W16,AA16,AB16,Y14,AB15,R13,AB14,U13,AB12,AA12,Y9,U10,R11AB10,AA10,V9,W9,AB8,AA8,AA7,W8,AA6	inout	2.5 V	**系统地址/数据**复用总线。在一次传输周期中，第一个时钟周期传输为地址，接下来传输的则为数据。对于读操作，当前面的读操作没有完成，下一次读操作可以继续发出，因为该接口总线设有缓冲，可以接受 8 个事务传输。
SysCmd [11:0]	C20,C19,D22,C21,D20,F22,E21,F20,M19,M18,G18,G22	inout	2.5 V	**系统总线命令/数据标识**复用总线。当传输地址时为总线命令，当传输数据时，则表示数据类型。
SysReq	R9	input	2.5 V	**处理器总线请求**信号。当CPU需要占用总线时，需要先向北桥发出请求信号，得到允许后即可将请求信号撤消。
SysGnt	V8	output	2.5 V	**北桥对处理器请求的应答信号。**北桥设置该信号有效，CPU获得总线使用权。
SysRel	AB11	inouy	2.5 V	**系统总线释放信号.**当总线占有者完成总线数

				据传输时，则将SysRel置低一个周期，表示释放总线/
SysRdRdy	Y5	output	2.5 V	**读准备好信号**。处理器对北桥读之前先要判断北桥的读准备好信号是否有效，如果无效，则必须等待该信号有效后才能进行操作。
SysWrRdy	AA5	output	2.5 V	**写准备好信号。**处理器对北桥写之前先要判断北桥的写准备好信号是否有效，如果无效，则必须等待该信号有效后才能进行操作。
SysResp [2:0]	U8,U9,T8	output	2.5 V	**桥片对写操作的请求号释放总线.**
SysRespVal	R10	output	2.5 V	**桥片对写操作的请求号释放有效标识**
SysState [2:0]	Y6,P8,AA0	input	2.5 V	**处理器对外部写操作的请求号的释放**
SysStateVal	AA4	input	2.5 V	**处理器对外部写操作的请求号释放的有效标识**
SysVal	C22	inout	2.5 V	**系统总线传输有效标识信号。**当总线传输中的地址或数据有效，则置该信号有效。表示当前传输的地址或数据是有效的，应给予处理。
nCint [5:0]	R19,R18,R17, P18,P17,P15	output	2.5 V	**中断信号，**提供给处理器的中断信号
nCnmi	N22	output	2.5 V	**不可屏蔽中断。**提供给处理器的不可屏蔽中断
VCCOK	N15	output	2.5 V	**处理器电源稳定**。当处理器电源稳定的提供大于 100ms 的至少 90%的标准电压时，该信号设为无效。
nColdReset	T7	output	2.5 V	**开机复位或冷复位**时，北桥将其设置为有效。
nCPUReset	W7	output	2.5 V	**CPU系统软复位信号**

3.2.2 PCI 接口信号

表 3.2.2 PCI 接口信号

管脚名称	位置	方向	电压标准	功能描述
PCI_CLK	L22	input	LVTTL	**PCI 时钟**。33MHz
PCI_RST	C13	inout	LVTTL	**PCI 复位**。当北桥为 PCI master 时，该信号为输出，驱动其他外部设备。当为 PCI target 时则为输入。
PCI_AD [31:0]	V2,T1,P2,W1, T2,W2,Y1,U2, N3,H1,N2,Y2, R2,U1,R1,V1, G6,D1,E2,E1, W5,R7,T5,T3, U4,P5,R5,R6, N4,P6N6,E4	inout	3.3-V PCI	**PCI 地址/数据**复用总线。PCI 传输的第一个时钟周期为物理地址；以后的为数据。当北桥为 PCI master 时，在地址和写数据时, pci_ad 为输出，读数据时则为输入。当北桥为 PCI target 时，在地址和写数据时为输入，读数据时为输出。
PCI_CBE [3:0]	P1,H2,G5,T6	inout	3.3-V PCI	**PCI 总线命令/字节使能**复用信号线。在地址期中，传输的是总线命令，在数据期内，传输的是字节使能信号。
PCI_PAR	W3	inout	3.3-V PCI	**PCI 奇偶效验位**
PCI_FRAME	Y4	inout	3.3-V PCI	**PCI 桢周期**信号。该信号有效表示总线传输开始，在其存在期间，数据传输继续进行，失效后，是传输的最后一个数据期。通常连接一个 2.7K 的上拉电阻。
PCI_Devsel	E3	inout	3.3-V PCI	**PCI 设备选择**信号。该信号有效时，表示驱动它的设备已成为当前访问的从设备。通常连接一个 2.7K 的上拉电阻。
PCI_IRDY	F1	inout	3.3-V PCI	**PCI 主设备准备好**信号。该信号有效表明发起本次传输的设备能够完成一个数据期。读周期该信号有效，表示数据已在总线上，写周期表示已作好接受数据准备。通常连接一个 2.7K 的上拉电阻。
PCI_TRDY	F2	inout	3.3-V PCI	**PCI 从设备准备好**信号。该信号有效表示从设备已作好完成当前数据传输的准备。在写周期该信号有效表示从设

				备作好接受数据准备；在读周期则表示数据已经提交到总线上。通常连接一个 2.7K 的上拉电阻。
PCI_STOP	Y3	inout	3.3-V PCI	**PCI 停止数据传输**信号。该信号有效时，表示从设备要求主设备终止当前的数据传输。通常连接一个 2.7K 的上拉电阻。
PCI_REQ [4:0]	A6,A5,A4，A3	input	LVTTL	**PCI 总线请求**信号。该信号有效表示驱动它的设备要求使用总线。是一个点对点的信号线，任何主设备都有其请求信号。
PCI_GNT [4:0]	A10,A9,A8,A7	output	LVTTL	**PCI 总线占用允许**信号。该信号有效表示申请总线使用权的设备已经获得批准。同请求信号一样，也是点对点信号线，每个主设备都有其 GNT 信号。
PCI_INTA	G11	inout	LVTTL	**PCI 中断**
PCI_PERR	D2	inout	3.3-V PCI	**PCI 奇偶效验错误报告**信号。对于数据接受设备，应在数据收到后的两个时钟周期内将该信号激活。对数据奇偶错误的报告不能丢失，也不能推迟。通常连接一个 2.7K 的上拉电阻。
PCI_SERR	W4	output	3.3-V PCI	**PCI 系统错误报告**信号。通常连接一个 2.7K 的上拉电阻。
pciclk_bak	L21	input	LVTTL	**PCI 时钟反馈**输入

3.2.3 Local IO 接口信号

表 3.2.3 Local IO 接口信号

管脚名称	位置	方向	电压标准	功能描述
nROM_CS	B19	output	LVTTL	**ROM 片选**信号。选择外部启动 Flash 设备。低电平有效。
nIO_CS	B18	output	LVTTL	**IO 片选**信号。选择简单 IO 设备，如 UART 等。
nIOWR	B17	output	LVTTL	**写使能**信号。该信号在写操作时有效。低电平有效
nIORD	B16	output	LVTTL	**读使能**信号。该信号在读操作时有效。低电平有效
nADLock	B15	output	LVTTL	**地址锁存**信号。
ioa[18:0]	H14,H13,H12,G16,G15,G12,F15,F14,F13,F12,E15,E14,D16,D15,D14,C18,C17,C14,B20	output	LVTTL	**地址总线**。
iod[7:0]	A20,A19,A18,A17,A16,A15,A14,A13	inout	LVTTL	**数据总线**。
ioden	B13	output	LVTTL	**数据使能**信号。控制外部双向收发设备的输出方向。
iodir	B14	output	LVTTL	**数据方向**信号。控制外部双向收发设备的传输方向。

3.2.4 全局信号

全局信号主要有给北桥外部设备的复位驱动信号，时钟信号，复位信号，测试信号以及通用输入输出信号。详细内容见下表

表 3.2.4 全局信号

管脚名称	位置	方向	电压标准	功能描述
dev_rst [7:0]	B11,B9,B8,B7,B6,B4,F11,B10	output	LVTTL	设备复位驱动信号
gclk	M21	input	LVTTL	全局时钟
gclk_bak	M22	input	LVTTL	全局反馈时钟
gpin[6:0]	E9,E8,J1,J2,J4,L1,E7	input	LVTTL	中断输入信号
gpio[3:0]	C16,H10,H9,H8	bidir	LVTTL	通用输入/输出信号.
rst_clk	J14	output	LVTTL	复位时钟.
TP	H11	output	LVTTL	北桥测试点.
nReset	B3	input	LVTTL	复位信号. 低电平有效

3.2.5 南桥复位与中断信号

一般南桥有很多的中断信号，除了把普通的中断信号引入中断控制器处理之外，也把一些重要的中断信号如 isa_nmi、isa_smi、isa_intr 等跳过中断控制直接通过北桥送给处理器处理。下表列出南桥的一些重要的中断信号和复位信号。

注意：在现阶段，与北桥搭配使用的南桥为 VIA 的 VT82C686B 芯片，这些信号都是参照该芯片得来。如果南桥芯片更换，这写信号可能会不同，所以用户自己选择南桥进行系统开发则应根据具体的情况具体处理

表 3.2.5 南桥复位与中断信号

管脚名称	位置	方向	电压标准	功能描述
isa_intr	D7	input	LVTTL	南桥中断输入
isa_nmi	D8	input	LVTTL	南桥不可屏蔽中断输入
isa_rstdrv	A11	input	LVTTL	南桥复位驱动输入
isa_smi	D9	input	LVTTL	南桥系统管理中断输入

3.2.6 电源和地

表 3.2.6 电源和地

管脚名称	位置	方向	功能描述
VCCA_PLL1	U7	power	锁相环 1 模拟电源，1.2V
VCCA_PLL2	F16	power	锁相环 2 模拟电源，1.2V
VCCA_PLL3	E6	power	锁相环 3 模拟电源，1.2V
VCCA_PLL4	U16	power	锁相环 4 模拟电源，1.2V
VCCD_PLL1	U6	power	锁相环 1 电源，1.2V
VCCD_PLL2	F17	power	锁相环 2 电源，1.2V
VCCD_PLL3	E5	power	锁相环 3 电源，1.2V
VCCD_PLL4	U17	power	锁相环 4 电源，1.2V
VCCINT	J10,J11,J12,J13,K9,K14,L9,L14,M9,M14,N9,N14,P10,P11,P12,P13	power	北桥核电压， 1.2V
VCCIO1	AA1,M3,P7,T4	power	北桥 IO 电压，3.3V.
VCCIO2	B1,J7,L3	power	北桥 IO 电压，3.3V.
VCCIO3	A2,C6,C11,E10,G9	power	北桥 IO 电压，3.3V.
VCCIO4	A21,C12,D17,E13,G14	power	北桥 IO 电压，3.3V.
VCCIO5	B22,G19,J16,L20	power	北桥 IO 电压，2.5V
VCCIO6	AA22,M20,P16,T19	power	北桥 IO 电压，2.5V
VCCIO7	AB21,T14,V13,W17,Y12	power	北桥 IO 电压，2.5V
VCCIO8	AB2,T9,V10,W6,Y11	power	北桥 IO 电压，2.5V
GND_PLL1	U5,V5	gnd	锁相环 1 地信号
GND_PLL2	E17,F18	gnd	锁相环 2 地信号
GND_PLL3	F5,F6	gnd	锁相环 3 地信号
GND_PLL4	T17,V18	gnd	锁相环 4 地信号
GNDA_PLL1	V7	gnd	锁相环 1 模拟地信号
GNDA_PLL2	E16	gnd	锁相环 2 模拟地信号
GNDA_PLL3	F7	gnd	锁相环 3 模拟地信号
GNDA_PLL4	V16	gnd	锁相环 4 模拟地信号
GND	A1,A22,AA2,AA21,AB1,AB22,B2,B21,C5,C8,C15,D10,D13,D18,F19,G4,G10,G13,H20,K3,K7,K10,K11,K12,K13,K16,K19,L10,L11,L12,L13,M4,M10,M11,M12,M13,N7,N10,N11,N12,N13,N16,N19,R3,T10,T13,T20,V3,V6,V17,W10,W13,W19,Y8,Y15	gnd	地

3.2.7 没有使用的管脚

NC 指芯片管脚没有连接，不能使用；而没有使用的 IO 管脚则指北桥没有使用到该管脚，如果北桥有改变，这些没有使用的管脚可以重新使用，方便系统升级，改版等。没有使用的输入管脚则是用于输入的管脚没有使用，同样如果有需要，可以重新使用。这些没有使用输入或输入输出管脚都接地，并设置为输入信号。这些管脚的使用与否由北桥开发者来维护。详细的管脚列表如下：

表 3.2.7 没有使用的管脚

管脚名称	位置	功能描述
NC	G1,G2,H21,H22,J3,J5,J6,J8,J9,K8,K15, K17,K18,L7,L15,L16,L17,M7,M8,M15,M16 N5,N8,P4,P14,P19,P20,P21,P22,R4,W18	无连接
GND*	AA3,AA11,AB3,AB4,AB5,AB6,AB7,AB9,B5, C1,C2,C7,C9,C10,D3,D4,D5,D6,D11,E11,E1 8,E19,E22,F3,F4,F8,F9,F10,G3,G7,G8,G20, H3,H4,H5,H6,H7,H15,H19,J20,J21,J22,K20, K21,K22,L8,L18,L19,M5,M6,N1,N21,P3,R8, R20,R21,R22,T11,T18,T21,T22,U3,U18,U19 ,U20,U21,U22,V4,V19,V20,V21,V22,W21,W 22,Y7,Y10,Y13,Y16,Y18,Y19,Y20,Y21,Y22	没有使用的 IO 管脚接地
GND+	A12,B12,D12,E12,L2,M1,M2,U11,U12,V12, W12	没有使用的输入管脚接地

第四章 模块功能描述

4.1 SysAD 接口模块

SysAD 总线是一套数据和地址复用的总线。龙芯 2E 的接口总线是 64bit，缺省配置为 66MHz 的 SysAD 总线。

4.1.1 龙芯 2E 与北桥信号连接

北桥与龙芯 2E 处理器之间的信号连接如下表所示：

表 4.1.1 龙芯 2E 与北桥信号连接

龙芯 2E 接口信号	北桥接口信号
SysAD[63:0] (I/O)	SysAD[63:0] (I/O)
SysADC[7:0] (I/O)	XXX
SysCmd[11:0] (I/O)	SysCmd[11:0] (I/O)
SysCmdPar (I/O)	XXX
SysReq# (I)	SysReq(O)
SysGnt# (I)	SysGnt (O)
WrRdy# (I)	SysWrRdy (O)
RdRdy# (I)	SysRdRdy (O)
SysValid# (I/O)	SysVal (I/O)
Release# (I/O)	SysRel (I/O)
SysResp[2:0] (I)	SysResp[2:0] (O)
SysRespVal# (I)	SysRespVal (O)
SysState[2:0] (O)	SysState[2:0] (I)
SysRespVal# (O)	SysRespVal (I)
SysStatePar (O)	XXX
SysRst# (I)	nCPUReset (O)
ColdRst# (I)	nColdReset (O)
VCCOK (I)	VCCOK (O)
NMI# (I)	nCnmi (O)
INT[5:0]# (I)	nCint[5:0] (O)

4.1.2 Syscmd 信号功能

为了提高系统总线的效率，龙芯2E 处理器系统总线采取了Split transaction 技术，为了实现该技术，处理器系统接口部件包含了总线请求管理缓冲、处理器读/写请求缓冲、外部读/写请求缓冲、处理器对外部读请求的数据响应缓冲、外部设备对处理器读请求的数据响应缓冲5个缓冲，为匹配龙芯2E系统总线能同时处理8个事务的能力，每个缓冲的深度也为8。

北桥接口部分也必须实现这样的结构提高总线效率，所以北桥部分以FIFO的结构缓冲最多8个处理器访问操作。每个事务的标示可以在Syscmd总线中表示。Syscmd信号组中各位定义如下表所示。

表4.1.2 Syscmd信号定义编码

总线 周期 类别	命令 类别	Syscmd[11:0]中各位的意义											
		11	10	9	8	7	6	5	4	3	2	1	0
处理器地址周期	块读	0	Request_number			0	X	0	X		Data_size		
	非块读	0	Request_number			0	X	1	X		Data_size		
	块写	0	Request_number			1	X	0	X		Data_size		
	非块写	0	Request_number			1	X	1	X		Data_size		
处理器数据周期	非块写	1	Request_number			X	X	X	Data_type		X		
	块写	1	Request_number			X	X	X	Data_type		X		
	数据响应	1	Request_number			X	0/1	X	Data_type		X		
外部地址周期	块读	0	Request_number			0	X	0	X		Data_size		
	非块读	0	Request_number			0	X	1	X		Data_size		
	块写	0	Request_number			1	X	0	X		Data_size		
	非块写	0	Request_number			1	X	1	X		Data_size		
外部数据周期	非块写	1	Request_number			X	X	X	Data_type		X		
	块写	1	Request_number			X	X	X	Data_type		X		
	数据响应	1	Request_number			X	0/1	X	Data_type		X		

其中：

(1) 外部信号周期和处理器信号周期中的命令格式完全一致；

(2) Request_number 与 MIPS R10000 中的 Request_number 的意义一样，只是在系统总线写操作命令中也包含 Request number；

(3) X 表示无意义；

(4) 在数据响应周期，Syscmd[6]标识对应于读操作的数据返回是否正确，即 bus error 信号。其中，Syscmd[6]＝0：表示数据是正确的；Syscmd[6]＝1：表示数据是不正确的。该信号主要用于处理器所发出的无效读请求的响应标识。

(5) Data_type 标识数据是与写操作对应还是与读操作的数据返回对应，并且还用来标识是否是最后一个数据。

Data_type ＝ 2’b00 ： 写请求对应数据，且不是最后一个数据；

Data_type ＝ 2’b01 ： 读请求对应数据，且不是最后一个数据；

Data_type ＝ 2’b10 ： 写请求对应数据，而且是最后一个数据；

Data_type ＝ 2’b11 ： 读请求对应数据，而且是最后一个数据。

(6) Data_size 标识读写操作对应的数据大小。

Data_size ＝ 3’b000 ： 一字节；

Data_size ＝ 3’b001 ： 二字节；

Data_size ＝ 3’b010 ： 三字节；

Data_size ＝ 3’b011 ： 四字节；

Data_size ＝ 3’b100 ： 五字节；

Data_size ＝ 3’b101 ： 六字节；

Data_size ＝ 3’b110 ： 七字节；

Data_size ＝ 3’b111 ： 八字节。

4.1.3 Sysstate 以及 Sysresp 信号组的意义

Sysstate 信号组用来实现处理器对外部写操作对应的 request number 的释放；Sysresp 信号组用来通知处理器一个 request number 被释放。

Sysstate 信号组由写操作对应的处理器完成对该写操作的处理时给出，该信号组直接与北桥相连，sysstateval 信号低电平标识有效，Sysstate[2:0]＝requst_number。

Sysresp 信号组由北桥接收到有效的 Sysstate 信号组后给出，sysrespeval 信号低电平标识有效，Sysresp[2:0]＝requst_number。

4.1.4 系统总线仲裁

龙芯 2E 处理器的系统总线使用权的仲裁是由北桥实现。系统总线使用权的仲裁是由北桥通过 SysReq、SysGnt、SysRel 这三个信号来完成的。

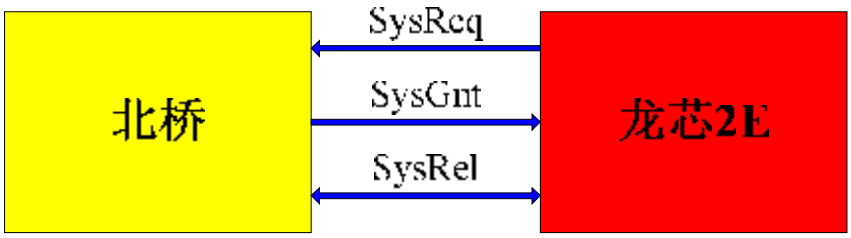


图 4.1.4 北桥与处理器总线仲裁连接

系统总线仲裁遵守如下规则：

从**北桥角度**来看：

（1）：在系统初始化的时候，总线的使用权被赋予北桥。

（2）：处理器通过 SysReq 向北桥提出使用总线的请求。如果此时北桥自身不需要使用总线（没有等待返回的处理器请求的数据、没有等待发出的 pci 读写请求），北桥将系统的使用权交给处理器（同时有效 SysGnt 和 SysRel）。

（3）：在缺省的情况下（北桥已经拥有总线使用权并且处理器没有请求总线使用权），北桥拥有总线的使用权。

（4）：在处理器占有总线时，如果北桥希望拥有总线使用权，北桥将先置 SysGnt 无效并等待处理器有效 SysRel（处理器释放总线）。系统总线使用权的变更仅仅依靠 SysRel 的有效来判断。

（5）：SysGnt 在某种程度上相当于北桥的请求信号（北桥在需要使用总线的时候要先置 SysGnt 无效）。

（6）：北桥在拥有系统总线使用权时如果没有处理器的请求，北桥不会主动放弃系统总线使用权。

从**处理器角度**来看：

（1）：如果当前处理器处于从设备状态（Slave State），此时，一个处理器请求或者数据响应已经准备好，并且系统总线资源有空闲（即：系统总线上有空闲的事务ID（Request Number）；对于读操作来说，RdRdy 有效；对于写操作来说，WrRdy 有效，等等），这样，处理器就发出有效的SysReq信号（该信号成为低电平），如果上述情况不满足，处理器将不会发出有效的SysReq 信号；

（2）：处理器等待有效的SysGnt 信号（SysGnt 为低电平）；

（3）：当处理器发现北桥插入有效的SysGnt 信号以后，隔两个系统总线时钟周期，处理器将SysReq 置为高电平，即获得系统总线使用权以后，不再申请系统总线使用权；

（4）：在处理器观察到北桥插入有效的Release 信号以后，隔两个系统总线时钟周期，

处理器进入主设备状态（Master State）；

（5）：一旦处理器进入主设备状态（Master State），除非处理器观察到北桥将其SysGnt信号置为无效（高电平），否则，处理器一直处于主设备状态；

（6）：当发现北桥将其SysGnt 信号置为无效（高电平），处理器在完成当前总线事务以后，立即发出有效的Release 信号（低电平有效），处理器处于从设备状态（Slave State）。

系统总线仲裁时序图

下图给出了系统总线使用权仲裁协议。其中，P0 代表当前处理器处于主设备状态（Master State），EA 代表北桥处于主设备状态，BR 代表块读（Block Read）,SR 代表单字读（Single Word Read），RD 代表读响应数据（Response Data），RLD 代表对应于块读请求的最后一个双字响应（Response Last Data）。

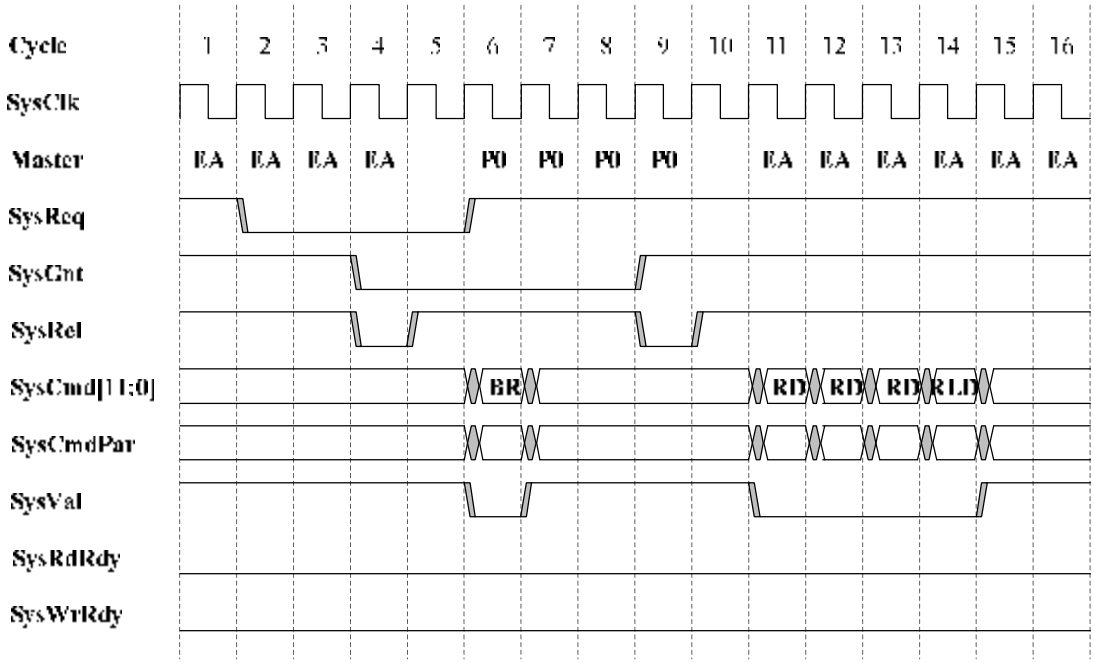


图**4.1.4** 系统总线使用权仲裁协议

4.1.5 CPU 的请求和北桥的响应

北桥在接收到处理器的读写请求后，北桥的 cpu 接口将这些请求转换为北桥内部模块互连使用的 wb 总线请求。由于我们在系统接口上最多支持 8 个 outstanding 的请求，我们先将接收到的请求和请求附带的数据（写请求时）放入一个 8 项的缓冲后在对接收到的请求进行转换。

由于龙芯 2E 的北桥没有内存控制器，双字/单字/部分字 读请求属于不正常的请求。但是为了处理器功能的验证，龙芯 2E 的北桥可以正确接收并转换块写请求，但对块读请求，北桥将返回全 0。

当写请求转换后的 wb 总线上的写操作完成，北桥的 cpu 接口通过 SysResp 释放已经完成的写请求的请求号。

当读请求转换后的 wb 总线上的读操作完成时，北桥的 cpu 接口将返回的读数据和对应的请求号放入一个 8 项的读返回缓冲中。当北桥获得系统总线使用权北桥并且内部总裁应该做数据响应时，北桥在系统总线上返回读数据。

北桥的 cpu 接口模块主要的存储资源包括一个 8 项的 cpu 请求缓冲（记录：地址、请求号、写数据）和一个 8 项的读数据返回缓冲（记录：请求号、返回的数据）。这两个缓冲占用了北桥 cpu 接口的决大部分寄存器资源。

处理器读操作协议

龙芯2E处理器系统总线读操作分为：块读（Block Read）和双字（Double）/字（Single）/部分字（Partial-Word）读请求，共两类读请求。无论是块读还是双字/字/部分字读请求，都是由地址周期和数据周期组成的。

处理器在如下条件满足的情况下可以发出系统总线读操作：

（1）：处理器系统接口处于主设备状态（Master State）；

（2）：系统总线上有空闲的事务ID（Request Number）；

（3）：读准备好信号（SysRdRdy）有效

在处理器系统总线读操作地址周期，处理器系统总线上的操作如下：

（1）：将系统命令总线的最高位（SysCmd[11]）置低，标识地址周期；

（2）：在系统命令总线的第10～第8 位（SysCmd[10:8]）上发出当前读操作对应的事务ID（即Request Number）；

（3）：将系统命令总线的第7 位（SysCmd[7]）置低，标识当前系统总线操作为读操作；

（4）：对于块读，将系统命令总线的第5 位（SysCmd[5]）置低；对于非块读，将系统命令总线的第5 位（SysCmd[5]）置高；

（5）：对于非块读，将系统命令总线的第2～第0 位（SysCmd[2:0]）置为读操作对应的字节数。其中，SysCmd[2:0]与读操作字节数对应关系见表4.1.5.1；

（6）：在系统地址/数据总线（SysAD）上发出读操作对应的地址；

（7）：将SysValid 置低，标识当前读操作有效。

表4.1.5.1 SysCmd[2:0]与读操作字节数对应关系

SysCmd[2:0]	数据大小	SysCmd[2:0]	数据大小
000	单字节	001	双字节
010	三字节	011	字
100	五字节	101	六字节
110	七字节	111	双字

图4.1.5.1 给出了处理器读操作协议在系统总线上的地址周期示意以及北桥响应处理器读操作协议在系统总线上的数据周期示意。

在图中P0代表当前处理器**P0** 是系统总线的主设备（即：Master ＝ P0）。在系统总线主设备为**P0** 时，系统总线上的读操作是由处理器**P0** 发起的。 **BR** 代表块读操作（BlockRead），**SR** 代表单个数据周期的读操作（Single Read），单个数据周期的读操作包括**1**～**8** 个字节的读操作。

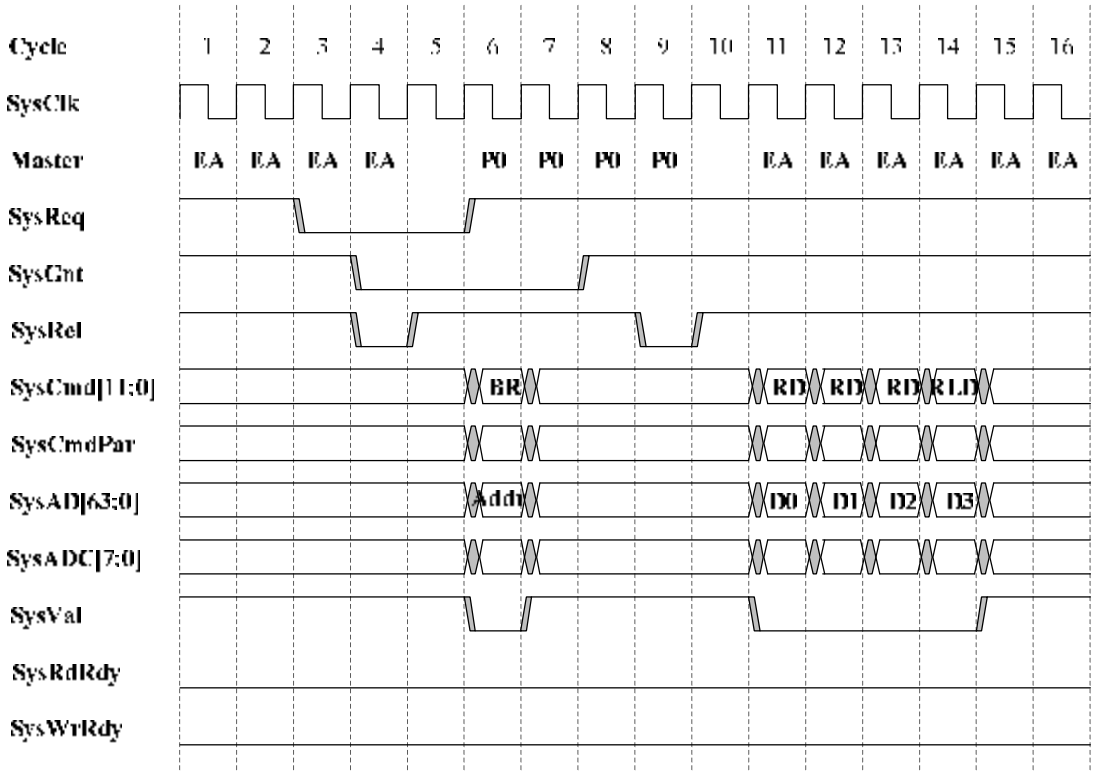


图 4.1.5.1 北桥响应 CPU 读请求时序图

处理器写操作协议

龙芯2E 处理器系统总线写操作分为：块写（Block Read）和双字（Double）/字（Single）/部分字（Partial-Word）写请求，共两类写请求。无论是块写还是双字/字/部分字写请求，都是由地址周期和数据周期组成的，与读操作不同，写操作的命令周期（地址周期）和数据周期是连续的。

处理器在如下条件满足的情况下可以发出系统总线写操作：

（1）：处理器系统接口处于主设备状态（Master State）；

（2）：系统总线上有空闲的事务ID（Request Number）；

（3）：写准备好信号（SysWrRdy）有效；

在处理器系统总线写操作地址周期，处理器系统总线上的操作如下：

（1）：将系统命令总线的最高位（SysCmd[11]）置低，标识地址周期；

（2）：在系统命令总线的第10～第8 位（SysCmd[10:8]）上发出当前写操作对应的事务ID（即Request Number）；

（3）：将系统命令总线的第7位（SysCmd[7]）置高，标识当前系统总线操作为写操作；

（4）：对于块写，将系统命令总线的第5 位（SysCmd[5]）置低；对于非块写，将系统命令总线的第5 位（SysCmd[5]）置高；

（5）：对于非块写，将系统命令总线的第2～第0 位（SysCmd[2:0]）置为写操作对应的字节数。其中，SysCmd[2:0]与写操作字节数对应关系见表5.6.2；

（6）：在系统地址/数据总线（SysAD）上发出写操作对应的地址；

（7）：将SysValid 置低，标识当前写操作有效。

在处理器系统总线写操作数据周期，处理器系统总线上的操作如下：

（1）：将系统命令总线的最高位（SysCmd[11]）置高，标识数据周期；

（2）：在系统命令总线的第4～第3 位（SysCmd[4:3]）标识数据是与写操作对应还是与读操作的数据返回对应，并且还用来标识是否是块操作中的最后一个数据。具体标识意义见表4.1.5.2；

（3）：在数据周期，数据输出有效信号（SysValid）一直置低，标识当前写数据有效。

表4.1.5.2 SysCmd[4:3]与写操作字节数对应关系

SysCmd[4:3]	含义
00	写请求对应数据，且不是最后一个数据
01	读请求对应数据，且不是最后一个数据
10	写请求对应数据，而且是最后一个数据
11	读请求对应数据，而且是最后一个数据

图4.1.5.2给出了处理器块写操作协议在系统总线上的传输示意。在图中P0 代表当前处理器P0 是系统总线的主设备（即：Master ＝ P0）。在系统总线主设备为P0 时，系统总线上的读操作是由处理器P0 发起的。 BW 代表块写操作（BlockWrite），WD 代表块写操作对应的数据（Write Data），WLD 代表块写操作对应的最后一个数据（Write Last Data）。

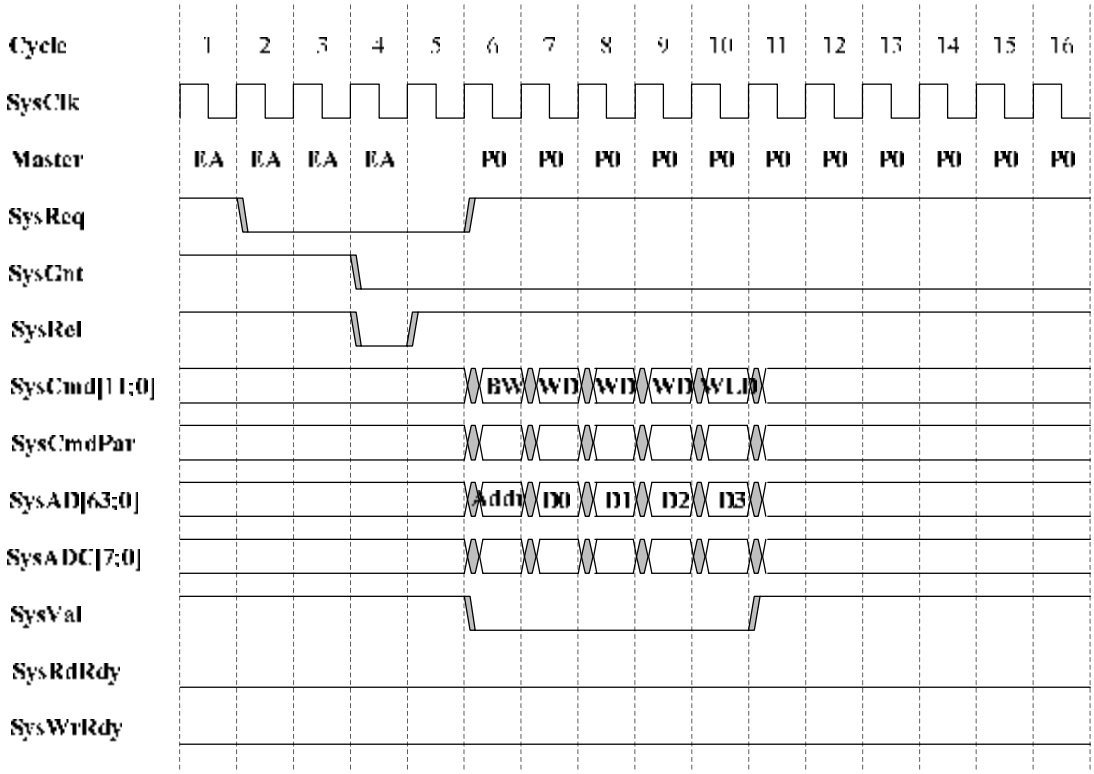


图 4.1.5.2 CPU 写请求时序图

4.1.6 北桥请求和 CPU 的响应

北桥的请求包括：

北桥双字/单字/部分字写请求

北桥块写请求

北桥块读请求

PCI 设备的读写请求经过北桥 PCI 模块的转换变为内部 wb 总线的请求。由于在系统接口上 CPU 只接收两种长度（块写的数据长度是 4，双字/单字/部分字 写请求的数据长度是 1）的写。在进行写操作时 pci 模块出来的 CAB_O 用来表明是否进行块写（高有效），write_num_o 表明非块写时写数据的长度。北桥的 cpu 接口直接将 wb 总线上的块写转换为对处理器的块写请求。对于不是块的写请求，北桥的 cpu 接口将 wb 的每一个写数据转换为一个处理器的双字/单字/ 写。由于系统总线上没有相应的位选信号，北桥将那些不能转换成双字/单字/ 写的写数据转换成相应的 byte 写。

当处理器处理完接收的写请求后，处理器会通过 SysState 释放相应写请求的请求号。

Pci 设备的读请求被统一转换为系统总线上的块读请求。北桥发出的块读请求的地址是 cache line 对齐的。当处理器将读请求的数据返回给北桥时，北桥的 cpu 接口将相应的读数据返回给 PCI 模块。

北桥读操作

系统总线上北桥读操作分为：块读（Block Read）和双字（Double）/字（Single）/部分字（Partial-Word）读请求，共两类读请求。无论是块读还是双字/字/部分字读请求，都是由地址周期和数据周期组成的。

北桥在如下条件满足的情况下可以发出系统总线读操作：

（1）：北桥系统接口处于主设备状态（Master State）；

（2）：系统总线上有空闲的事务ID（Request Number）；

在北桥系统总线读操作地址周期，系统总线上的操作如下：

（1）：将系统命令总线的最高位（SysCmd[11]）置低，标识地址周期；

（2）：在系统命令总线的第10～第8 位（SysCmd[10:8]）上发出当前读操作对应的事务ID（即Request Number）；

（3）：将系统命令总线的第7位（SysCmd[7]）置低，标识当前系统总线操作为读操作；

（4）：对于块读，将系统命令总线的第5 位（SysCmd[5]）置低；对于非块读，将系统命令总线的第5 位（SysCmd[5]）置高；

（5）：对于非块读，将系统命令总线的第2～第0 位（SysCmd[2:0]）置为读操作对应的字节数。

（6）：在系统地址/数据总线（SysAD）上发出读操作对应的地址；

（7）：将SysValid 置低，标识当前读操作有效。

处理器响应北桥读请求操作协议

处理器的数据响应是由北桥对处理器所管辖的内存空间进行读操作而引起的。

处理器在如下条件满足的情况下可以发出数据响应操作：

处理器系统接口处于主设备状态（Master State）；

在处理器系统总线数据响应周期，处理器系统总线上的操作如下：

（**1**）：将系统命令总线的最高位（SysCmd[11]）置高，标识数据周期；

在系统命令总线的第4～第3 位（SysCmd[4:3]）标识数据是否是块操作的最后一个数据。

（**1**）：在数据周期，数据输出有效信号（SysValid）一直置低，标识当前写数据有效。

图4.1.6.1 给出了北桥读操作协议在系统总线上的地址周期示意以及处理器数据响应协议在系统总线上的示意。在图中P0 代表当前处理器P0 是系统总线的主设备（即：Master ＝ P0）。EA 代表当前时刻外部设备是系统总线的主设备（即：Master ＝ EA）。在系统总线主设备为EA 时，系统总线上的读操作是由北桥EA 发起， BR 代表块读操作（Block Read），RD 代表读操作对应的数据（Read Data），RLD 代表读操作对应的最后一个数据（Read Last Data）。

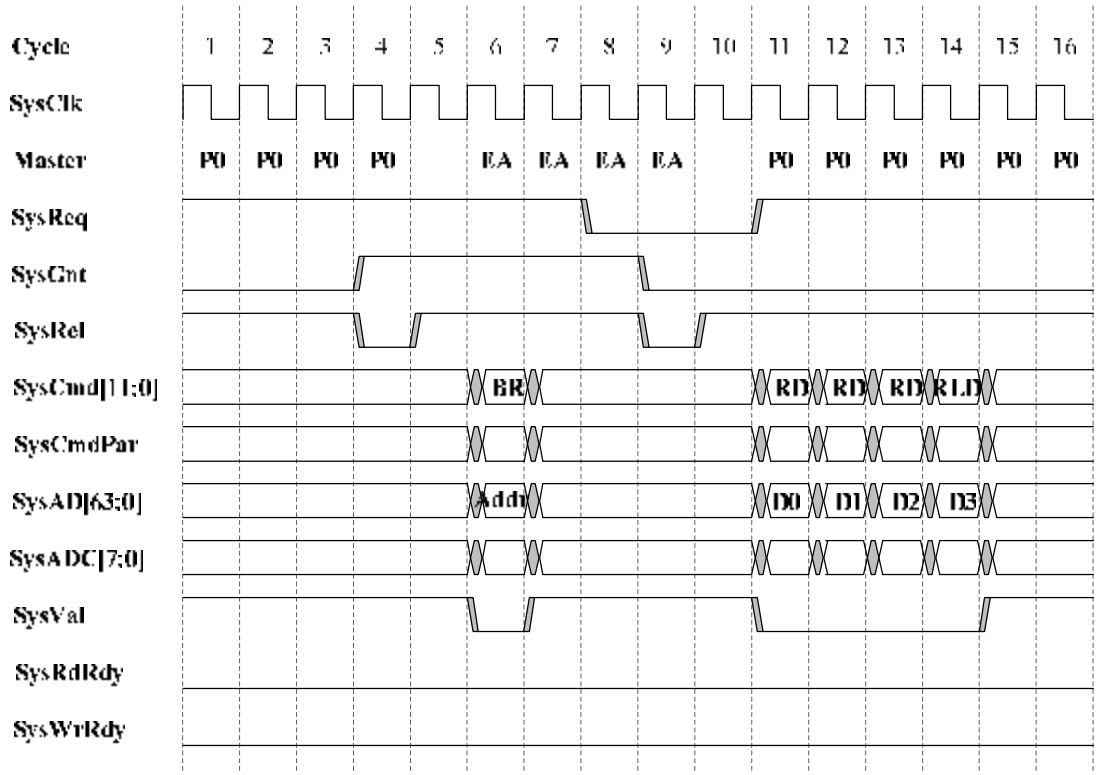


图 4.1.6.1 北桥读请求 CPU 响应时序图

北桥写操作协议

系统总线上北桥写操作分为：块写（**Block Read**）和双字（**Double**）/字（**Single**）/部分字（**Partial-Word**）写请求，共两类写请求。无论是块写还是双字/字/部分字写请求，都是由地址周期和数据周期组成的，而且写操作的命令周期（地址周期）和数据周期是连续的。

北桥在如下条件满足的情况下可以发出系统总线写操作：

（**1**）：北桥系统接口处于主设备状态（**Master State**）；

（**2**）：系统总线上有空闲的事务**ID**（**Request Number**）；

在北桥系统总线写操作地址周期，系统总线上的操作如下：

（**1**）：将系统命令总线的最高位（**SysCmd[11]**）置低，标识地址周期；

（**2**）：在系统命令总线的第**10**～第**8** 位（**SysCmd[10:8]**）上发出当前写操作对应的事务**ID**（即**Request Number**）；

（**3**）：将系统命令总线的第**7**位（**SysCmd[7]**）置高，标识当前系统总线操作为写操作；

（**4**）：对于块写，将系统命令总线的第**5** 位（**SysCmd[5]**）置低；对于非块写，将系统命令总线的第**5** 位（**SysCmd[5]**）置高；

（**5**）：对于非块写，将系统命令总线的第**2**～第**0** 位（**SysCmd[2:0]**）置为写操作对应的字节数。

（**6**）：在系统地址/数据总线（**SysAD**）上发出写操作对应的地址；

（**7**）：将**SysValid** 置低，标识当前写操作有效

在系统总线上北桥写操作数据周期，系统总线上的操作如下：

（**1**）：将系统命令总线的最高位（**SysCmd[11]**）置高，标识数据周期；

（**2**）：在系统命令总线的第**4**～第**3** 位（**SysCmd[4:3]**）标识数据是与写操作对应还是与读操作的数据返回对应，并且还用来标识是否是块操作中的最后一个数据。

（**3**）：在数据周期，数据输出有效信号（**SysValid**）一直置低，标识当前写数据有效。

图**4.1.6.2** 给出了北桥块写操作协议在系统总线上的传输。在图中P0 代表当前处理器P0 是系统总线的主设备（即：Master ＝ P0）。EA 代表当前时刻外部设备是系统总线的主设备（即：Master ＝ EA）。在系统总线主设备为EA 时，系统总线上的写操作是由北桥EA 发起，BW 代表块读操作（Block Write），WD 代表写操作对应的数据（Write Data），WLD 代表写操作对应的最后一个数据（Write Last Data）。系统总线上传输的是北桥的块写操作。

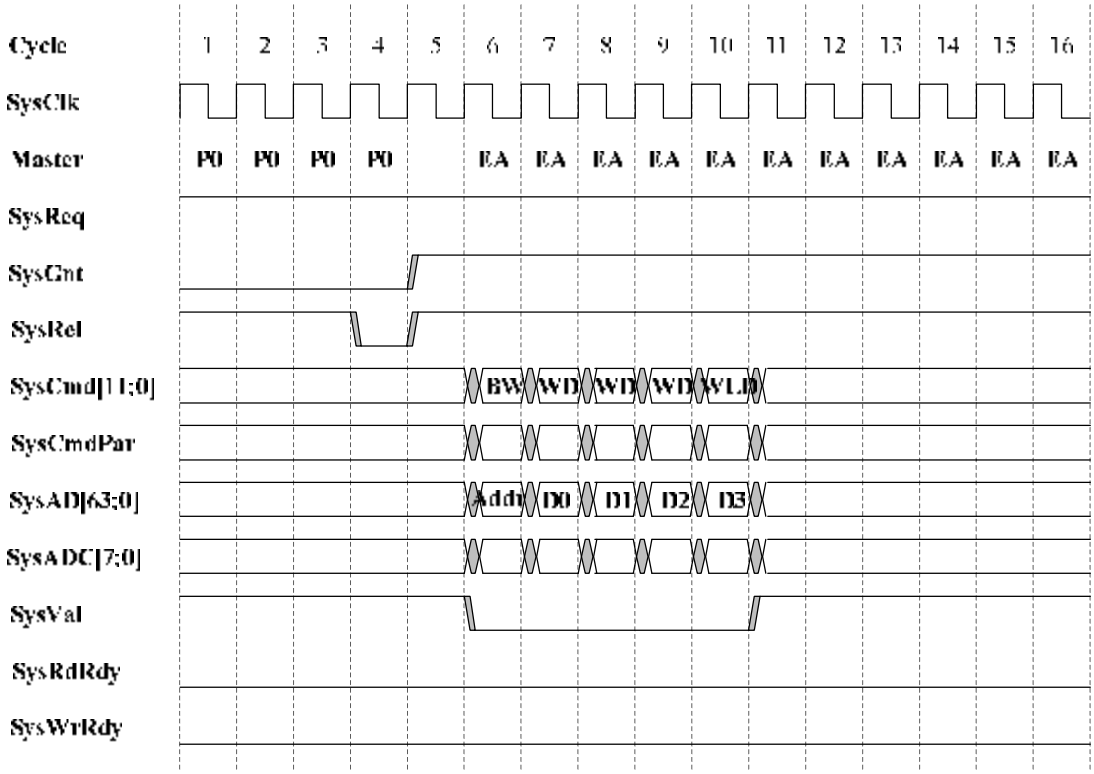


图 **4.1.6.2** 北桥写请求时序图

4.2 处理器复位模块

龙芯 2E 处理器有三种复位信号：VCCOK，ColdRst#，SysRst#。

VCCOK：上电复位，系统上电时发生，初始化处理器内部状态机；

ColdRst#：冷复位，重新启动所有时钟并彻底初始化处理器内部状态机，但此时电源保持稳定；

SysRst#：热复位，重新启动处理器，但此时时钟与处理器内部状态机的状态不受影响。

处理器对三个复位信号的要求：

VCCOK 要求：当下列条件成立时，VCCOK 开始有效并开始初始化处理器基础操作模式的配置。

（1）：输入时钟 SysClk 已经稳定的输入了 100ms；

（2）：标准为+3.3V 的供电电源（VccIO）已经提供了 100ms 的大于 3.0V 的电压。

（3）：标准为+3.3V 的供电电源（VccIOP）已经提供了 100ms 的大于 3.0V 的电压。

（4）：标准为+3.3V 的供电电源（VccInt）已经提供了 100ms 的大于 3.0V 的电压。

（5）：标准为+3.3V 的供电电源（VccIntP）已经提供了 100ms 的大于 3.0V 的电压。

ColdRst#要求：上电复位或冷复位时外部设备（北桥）将 ColdRst#信号置低有效；当 ColdRst#信号跳高无效撤消时，必须同步于 SysClk 时钟信号。

SysRst#要求：对于任何复位方式，外部设备（北桥）都必须设置 SysRst#为有效（低电平有效），当冷复位时，SysRst#信号的有效设置可以同步于 SysClk，也可以异步于 SysClk，而当热复位时，SysRst#信号的有效设置就必须同步于 SysClk 时钟，其无效设置也必须同步于 SysClk 时钟。

对于这三个复位信号的要求，北桥集成了专用逻辑电路模块进行了处理。使用该北桥的用户只需要将北桥输出的三个复位信号和处理器的三个输入复位对应连接即可，外接 4.7K 至 10K 的上拉电阻。复位信号的连接如下图.所示：

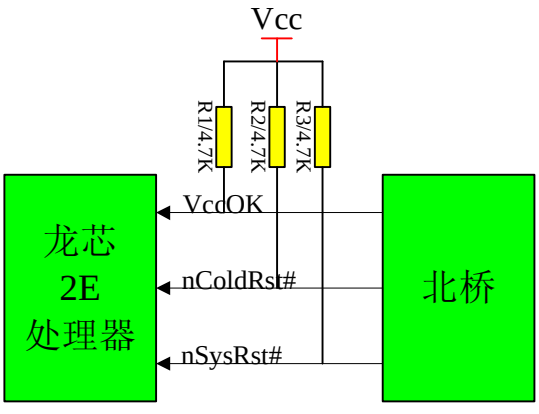


图 4.2 北桥与处理器复位信号连接图

4.3 PCI 模块

北桥实现的 PCI 总线的数据宽度为 32 位，频率为 33MHz，符合 PCI2.2 版本规范。

4.3.1 PCI 访问内存

由于龙芯 2E 集成了 DDR 内存控制器。所以当 PCI 设备访问内存的时候，其路径比较长，首先将 PCI 访问转成 WishBone，然后 WishBone 转成 SysAD，再由 SysAD 访问内存。其结构图如下：

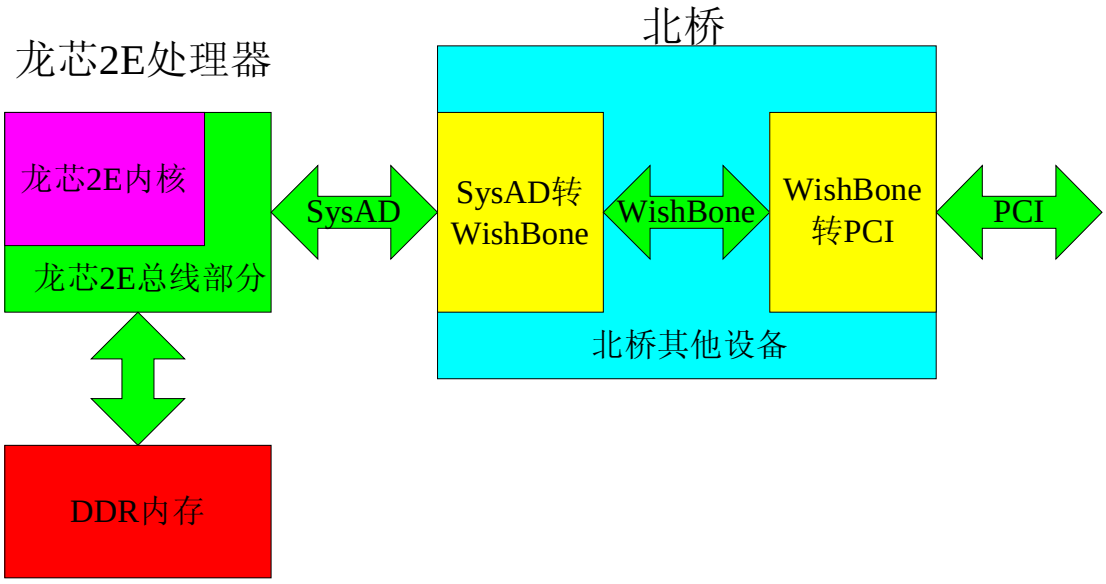


图 4.3.1 PCI 访问内存路径图

如上图所示，处理器访问内存的数据总线是 64 位的，SysAD 总线也是 64 位的，WishBone 总线也是 64 位的，PCI 总线是 32 位的。SysAD 总线支持总线突发传输，其突发长度为 4 个双字，既 32 个字节。PCI 总线也支持突发传输。

外部 PCI 设备访问内存时，北桥可以映射高达 256MB 的内存空间。但在实际应用中除了映射整个地址空间外没有做任何事情，所以，在默认的情况下，北桥只会提供一个 8MB 的窗口。这样北桥可以通过对寄存器 **Pcimembasecfg** 设置一个偏移量，从而将窗口重新映射到北桥的内存空间。

关于 **Pcimembasecfg** 寄存器的详细功能与定义请参考第五章第 5.4.1 小节关于该寄存器的介绍。

4.3.2 WishBone 转 PCI

WishBone 转 PCI 模块内部则使用了双向各两个 FIFO 来传输数据，这样就增加传输速度和效率。WishBone 转 PCI 模块部分结构如下图：

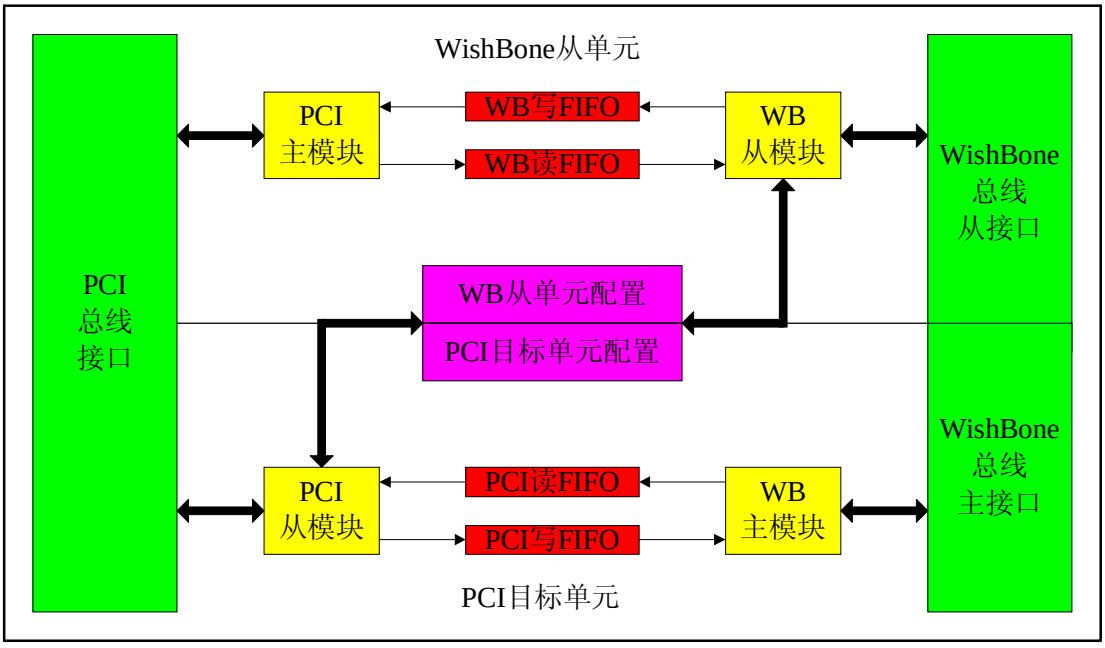


图 4.3.2 WishBone 转 PCI 结构框图

如上图所示，PCI 该部分有两个独立单元组成，PCI 目标模块和 WishBone 从单元模块。每个单元自身都有一套完整的功能支持 WishBone 和 PCI 总线之间的传输。WishBone 从单元可作为该模块中 WishBone 侧的从设备和 PCI 侧的主设备；PCI 目标单元则作为 PCI 侧的目标设备和 WishBone 侧的主设备。PCI 目标单元实现 PCI 总线从设备接口和 WishBone 总线的主设备接口；而 WishBone 从单元则用于实现 WishBone 总线从接口和 PCI 总线的主设备接口。

4.3.2 PCI 仲裁

北桥集成了 PCI 仲裁器，能处理 8 个 PCI 主设备的请求。其中一个来自北桥内部，7 个来自北桥外部。PCI 仲裁机制采用轮转优先级仲裁算法，使得每个 PCI 主设备都能够公平的得到总线使用权。下表为 PCI 总线申请者列表。

表 4.3.2 PCI 总线请求设备

总线申请设备	请求	允许
北桥 PCI	Pci_req[0]#	Pci_gnt[0]#
外部设备#0	Pci_req[1]#	Pci_gnt[1]#
外部设备#1	Pci_req[2]#	Pci_gnt[2]#
外部设备#2	Pci_req[3]#	Pci_gnt[3]#
外部设备#3	Pci_req[4]#	Pci_gnt[4]#
外部设备#4	Pci_req[5]#	Pci_gnt[5]#
外部设备#5	Pci_req[6]#	Pci_gnt[6]#
外部设备#6	Pci_req[7]#	Pci_gnt[7]#

北桥暂时引出 5 组 PCI_Req 和 PCI_Gnt 信号以提供给外部 PCI 设备使用。如果某些系统对要求更多的 PCI_Req 和 PCI_Gnt 信号，可给予支持。

4.3.3 PCI 总线命令

北桥 PCI 总线主要实现了 IO 读写，存储器读写，配置读写以及特殊指令周期等总线命令，其他的则暂时不支持。如下表为 PCI 总线命令列表

表 4.3.3 PCI 总线命令

命令编码	PCI 总线命令	北桥 PCI 支持
0000	中断应答	暂不支持
0001	特殊周期	支持
0010	I/O 读	支持
0011	I/O 写	支持
0100	保留	不支持
0101	保留	不支持
0110	存储器读	支持
0111	存储器写	支持
1000	保留	不支持
1001	保留	不支持
1010	配置读	支持
1011	配置写	支持
1100	存储器多行读	不支持
1101	双地址周期	不支持
1110	存储器一行读	不支持
1111	存储器写并无效	不支持

4.3.4 访问 PCI 配置空间

当处理器访问外部 PCI 设备的配置空间时，需要在 PCI 总线的地址周期上发送一个专用值。PCI 协议中该值定义如下：

当为“**type0**”型访问时的格式

[31:11]	[10:8]	[7:2]	1	0
系统给出正确的 IDSEL	功能号	寄存器号	0	0

当为“**type1**”型访问时的格式

[31:24]	[23:16]	[15:11]	[10:8]	[7:2]	1	0
不需要 IDSEL	总线号	设备号	功能号	寄存器号	0	0

为了正确的产生上面所述的总线值与配置时序，北桥使用寄存器 **pcimap_cfg** 用于控制外部 PCI 设备配置空间的访问。该寄存器的详细功能详见第五章第 4 小节中的 PCI 配置空间地址映射寄存器。

4.3.5 访问外部 PCI 设备

当处理器访问外部 PCI 设备的存储器空间或 IO 空间。其地址需要从系统地址转化到 PCI 地址。

访问 PCI 存储器空间时，将系统地址和 **Pcimap** 寄存器的值进行一定的处理生成 PCI 总线上需要的地址。处理过程为：输入地址和掩码地址的反进行逻辑与，同时传输地址与掩码地址进行逻辑与，这两个结果再进行逻辑或就形成访问 PCI 存储器空间的 PCI 地址。（输入地址指系统地址，掩码地址为常数，传输地址高位地址的则由 **Pcimap** 提供，低位地址为“0”。掩码地址常数值的大小跟要访问的 PCI 空间大小相关）。

北桥的 PCI 存储空间共有三个 64MB 部分。**Pcimap** 寄存器则负责提供三个传输地址的高位部分。具体的功能定义请参考第五章第 4 小节关于 **Pcimap** 的详细描述。

访问 PCI IO 空间时，与上同理，不过由于 IO 空间只有一部分，没有设专门的寄存器来进行控制。

在小节 WishBone 转 PCI 中的结构图中所描述，对外部 PCI 设备进行写操作总是允许的，除非 PCI 写 FIFO 已满。PCI 传输支持 32 位，16 位和 8 位的的数据传输

需要注意的一点是系统地址空间标示的 PCI_1.5 和 PCI_2 两段地址空间暂时没有映射到 PCI 空间上。

4.3.6 特殊周期命令

北桥 PCI 实现了特殊周期命令。由于特殊周期命令的目的是提供一种消息广播机制，因此该命令的发送中没有明显的目标地址，而是广播给所有的设备，每个接受设备必须自我确定广播的消息是否适合它。PCI 总线上的设备也不能设置 DEVSEL#信号作为对特殊周期命令的响应，也就是说在此类传输中，不需要任何目标握手信号；

北桥实现了该命令，上层软件只需要在寄存器 special_cycle 写入要广播的内容，就可以在总线上实现所需要的总线特殊周期。由于特殊周期也包含了一个地址期和一个数据期。而地址则可以是任意值，但要保持稳定。因此默认情况下，当 PCI 上传输特殊周期命令时，地址总是为 0。特殊周期命令时序如下图：

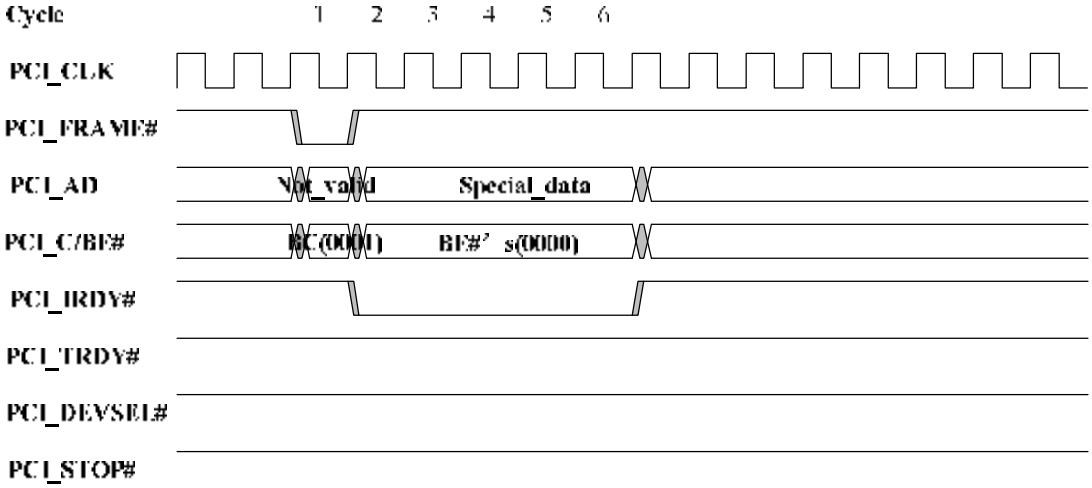


图 4.3.6 特殊周期命令时序图

PCI 特殊周期命令的终止要依靠主设备的废止方式，此时，相应的目标设备就知道访问已经完成。特殊周期命令会以 master abort 的方式报告给上层软件。

4.4 Local IO 接口模块

Local IO 是用来访问并行总线设备的总线接口。主要提供了地址总线，数据总线以及片选、读使能、写使能、以及地址锁定，数据方向，数据使能等控制信号总线，既可以访问 intel 风格的并行总线设备，又可以访问 Motorola 风格的并行总线设备。

Local IO 控制器是挂在内部 WishBone 总线上的一个从设备。Local IO 总线的配置寄存器 **iodevcfg** 是配置片外设备访问速度的寄存器。该寄存器的详细功能描述请参考第五章第 5 小节 Local IO 配置寄存器部分。

4.4.1 片外设备读写速度的配置

Local IO 总线上的设备的读写速度取决于北桥与处理器之间的总线传输速率。也既北桥内部系统总线的时钟频率。

当北桥内部系统总线时钟频率为 **90~166MHz**（时钟周期 **6ns ~11ns**）时，速度等级定为 0 级；为 **66.7~83.3MHz**（时钟周期 **12ns~15ns**）时，速度等级定为 1 级；为 **32~61.5MHz**（时钟周期 **16ns~31ns**）时速度等级定为 2 级；小于 **31MHz**（时钟周期大于 **32ns**）时速度等级则定为 3 级。

对于慢速 ROM 存储器，当速度等级分别为 0、1、2、3 级时，对系统总线的分频系数分别为 15、9、7、4；而对于快速 ROM 存储器，当速度等级分别为 0、1、2、3 级时，对系统总线的分频系数则分别为 10、5、4、3；

对于慢速 IO 设备，当速度等级分别为 0、1、2、3 级时，对系统总线的分频系数分别为 32、25、20、10；而对于快速 IO 设备，当速度等级分别为 0、1、2、3 级时，对系统总线的分频系数则分别为 15、8、6、3；

北桥内部有一个配置选择控制信号 **cfg_sel**（不可读写，由北桥综合时定义生成，一般默认值为‘0’），当 **cfg_sel** 为低电平时，**iodevcfg**[31:26]上电复位值为 **6’b001010**；当 **cfg_sel** 为高电平时，**iodevcfg**[31:26]上电复位值为 Local IO 总线 **iod**[5:0]的值。

当处理器前端总线时钟频率为 **90~166MHz**（时钟周期 **6ns ~11ns**）时，速度等级为 0 级；为 **66.7~83.3MHz**（时钟周期 **12ns~15ns**）时，速度等级为 1 级；为 **32~61.5MHz**（时钟周期 **16ns~31ns**）时为 2 级；小于 **31MHz**（时钟周期大于 **32ns**）时则为 3 级。

iodevcfg[31:27]为 4’b0010 或着 **iodevcfg**[31:28]为 3’b000 时，速度等级为 0 级；**iodevcfg**[31:27]为 4’b0011 时，速度等级为 1 级；为 **iodevcfg**[31:30]2’b01 速度等级为 2 级；**iodevcfg**[31:26]为其他值时速度等级为 3 级。

一般情况下，系统总线为 66MHz，**iodevcfg**[31:26] 上电复位值为 **6’b001010**；有特殊需求的系统需要则需要特殊处理。

4.5 GPIO 模块

北桥集成了 GPIOEnable 和 GPIODATA 两个控制寄存器对 gpin[6:0]和 gpio[8:0]16 个 IO 管脚进行控制。其中 gpin[6:0]为输入管脚，gpio[8:0]则是输入输出复用的管脚。

gpin[6:0]通常用于中断信号输入。如果用做其他用途（非中断信号输入），则在使用的时候，需要将中断使能寄存器 inten 上相对应的位清零，这样，输入信号就不会被误认为是中断信号。

gpio[8:0]编程为输入信号的时候，同 gpin 用法相同，即可以用做中断信号输入，也可以用做其他用途。当 gpio 不用于中断信号输入时，也要将 inten（中断使能寄存器）上相对应的位清零，避免产生错误中断。

GPIOEnable 用于控制 GPIO 的输入输出方向，GPIODATA 用于读 gpin 的值和读写 gpio 的值，具体的功能定义请详见 5.7 小节 GPIO 控制寄存器。

当 gpin 和 gpio 用于中断信号输入时，中断信号会锁存到中断状态寄存器上，具体的相互关系请详见图 4.6.1 中断处理结构图。

4.6 中断控制器模块

本小节主要介绍中断控制器对中断的处理。

4.6.1 中断处理

中断控制模块一共有 7 个寄存器对进入中断处理模块的 32 个中断源进行处理。这 7 个中断寄存器分别为：中断状态寄存器（intisr），中断使能寄存器（inten），设置中断使能寄存器（intenset），清零中断使能寄存器（intenclr），中断源极性寄存器（intpol），中断源触发方式寄存器（intedge）和中断导向寄存器（intsteer）。

当输入中断源的中断条件成立时，则将相应的中断状态寄存器位置‘1’。中断源极性寄存器和中断触发方式寄存器控制中断源是采用何种方式产生有效中断，中断使能寄存器，设置中断使能寄存器和清零中断使能寄存器则用来控制中断源是否被允许产生中断。

关于这些寄存器的详细功能见 5.6 小节中断控制寄存器。

如下图所示，中断处理模块能处理 32 个中断输入，这 32 个中断信号通过中断导向寄存器的控制分别输入到 nCint[0]和 nCint[1]；当中断信号相应的中断导向位为“0”时，则通过 nCint[0]输出，中断导向位为“1”则通过 nCint[1]输出。（中断导向寄存器功能详见小节 5.6 中断控制寄存器）而 nCint[5:2]和 nCnmi 还可以接受处理 5 个中断信号。所以北桥一共能处理 37 个中断。同时提供给处理器 7 个中断信号。

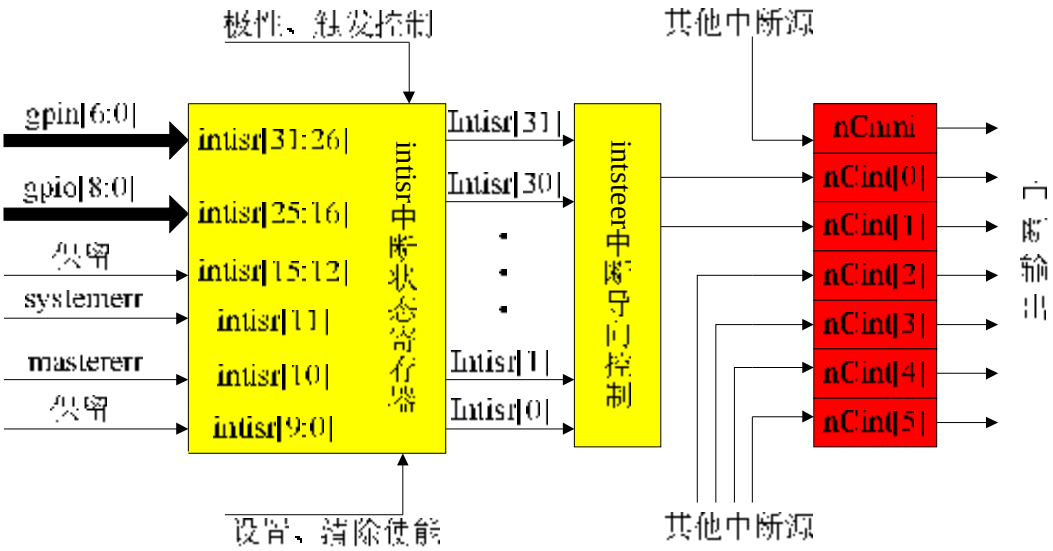


图 4.6.1 中断处理结构图

4.6.2 中断导向处理

图 4.6.2 为中断导向控制结构图，如图所示，中断导向寄存器每个位将其对应的中断输入信号分别导向两个部分。当中断导向位为“0”时，相应的中断信号被导入到 nCint[0]前的逻辑或运算部分；而中断导向位为“1”时，则中断信号被导入到 nCint[1]前的逻辑或运算部分。

所以，无论任何一个中断信号有效，都会通过 nCint[0]或 nCint[1]向处理器发起中断，处理器接受到中断信号后，则通过查询中断状态寄存器 intisr 查明是哪个中断有效，并转入中断处理服务程序。

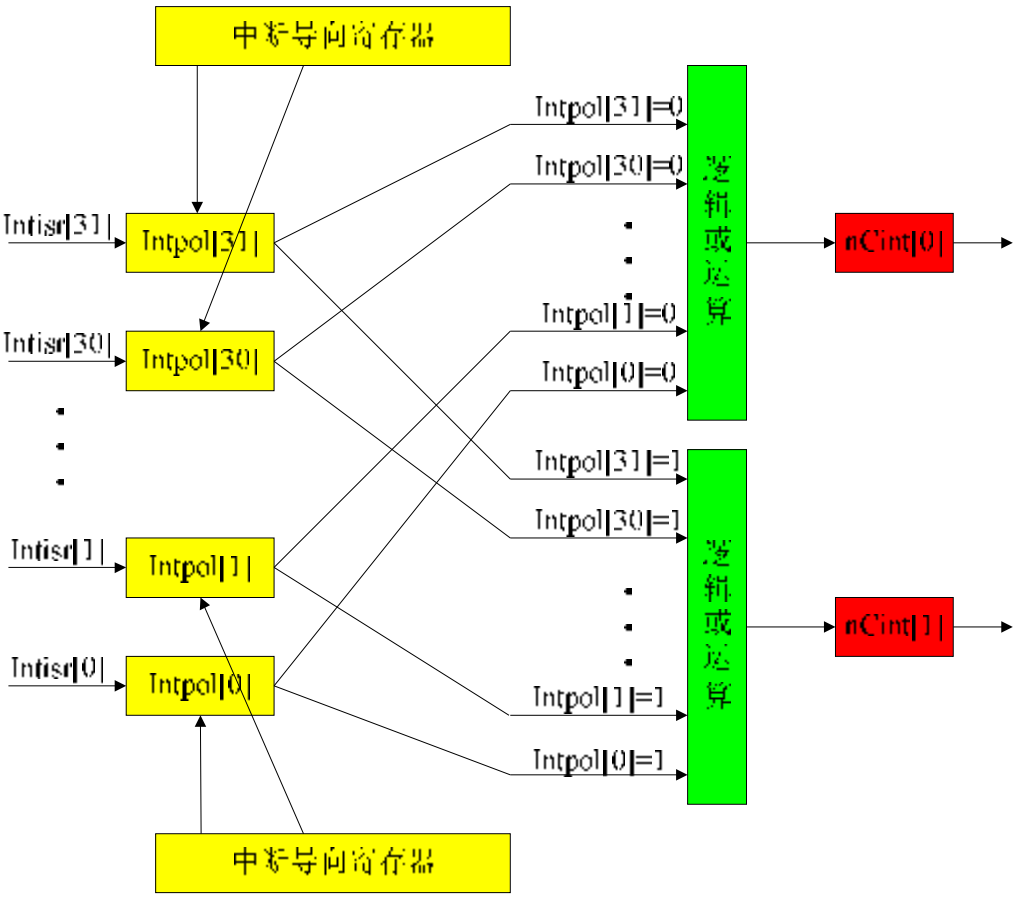


图 4.6.2 中断导向控制图

4.6.3 中断源列表

中断控制寄存器的每个相同的位都对应着相同的中断源输入。对于中断控制器处理的32 个中断源信号如下表所列：

表 4.6.3 中断源

位	中断源	中断描述
[31:25]	gpin[6:0]	来自通用输入管脚 gpin[6:0]的中断信号
[24:16]	gpio[8:0]	来自通用输入输出管脚 gpio[8:0]的中断信号
[15]	xxx	没有使用
[14]	inttimer	没有使用
[13]	retryerr	没有使用
[12]	dramperr	没有使用
[11]	systemerr	
[10]	mastererr	PCI 传输错误产生的中断信号
[9]	pciirq	没有使用
[8:0]	xxx	没有使用

4.6.4 中断控制器初始化

为避免无效的中断干扰输入，系统启动时应需要对中断控制器的 intpol 寄存器和 intedge 寄存器进行配置，通过对 intclr 写入全‘1’从而屏蔽所有的中断源，这样就确保所有边沿检测锁存器的值为‘0’；然后再通过 CPU 使能所有的中断，这样每个设备就会正确的驱动的的中断信号。

第五章 寄存器描述

5.0 寄存器地址空间

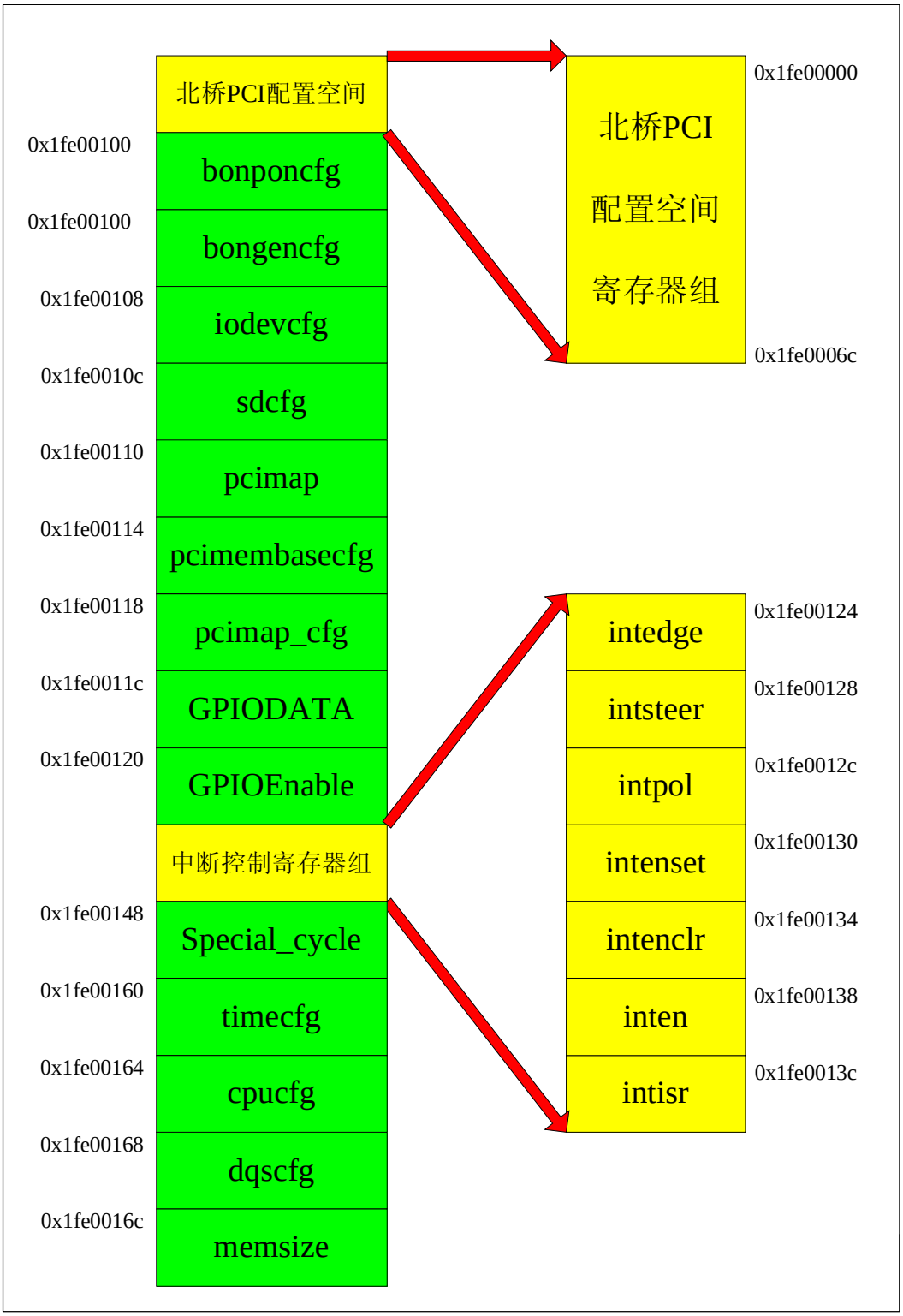


图 5.0 寄存器摘要图

5.1 CPU 配置寄存器

5.1.1 cpucfg 寄存器配置

物理地址：**0x1FE0_0160**
读写类型：可读可写
复位值： **0x0000_0000**

cpucfg 寄存器图

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

cpucfg 位描述

位	名称	描述
[31:0]	Reserved	保留；

5.2 SDRAM 配置寄存器

5.2.1 sdcfg 寄存器配置

物理地址：**0x1FE0_010C**
读写类型：可读可写
复位值： **0x255E_0091**

sdcfg 寄存器图

X	X	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Unused		dimm0	dimm1		interleave	ddr type				tREF					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
tREF						tRCD	tRFC		tRAS	tCAS		tWR	tRP	tRC	

sdcfg 位描述

位	名称	描述
[31:30]	Null	没有使用；
[29]	dimm0	保留
[28:27]	dimm1	保留
[26]	interleave	保留
[25:22]	ddr type	保留
[21:10]	tREF	保留
[9]	tRCD	保留
[8:7]	tRFC	保留
[6]	tRAS	保留
[5:4]	tCAS	保留
[3]	tWR	保留
[2]	tRP	保留
[1:0]	tRC	保留

5.2.2 dqscfg 寄存器配置

物理地址：**0x1FE0_0168**
读写类型：可读可写
复位值： **0x0000_0008**

dqscfg 寄存器图

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

dqscfg 位描述

位	名称	描述
[31:0]	Reserved	保留；

5.2.3 memsize 寄存器配置

物理地址：**0x1FE0_016C**
读写类型：可读可写
复位值： **0x1000_0000**

memsize 寄存器图

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

memsize 位描述

位	名称	描述
[31:0]	Reserved	保留；

5.3 PCI 配置空间寄存器

表 5.3 PCI 配置寄存器

Device ID 00D5h		Vendor ID DF53h		00h
Status		Command		04h
Class Code 060000h			Revision 01h	08h
0	Header Type 00h	Latency Timer	Cache Line Size	0Ch
Base Address Register0		pci_base_addr0		10h
Base Address Register1		pci_base_addr1		14h
Base Address Register2		pci_base_addr2		18h
Base Address Register3		pci_base_addr3		1Ch
Base Address Register4		pci_base_addr4		20h
Base Address Register5		pci_base_addr5		24h
XXX				28h
0		0		2Ch
XXX				30h
Unused				34h
Reserved				38h
Max_Lat	3Ch	Interrupt pin 01h	Interrupt line	3Ch
Mask Address Register0		pci_mask_addr0		40h
Mask Address Register1		pci_mask_addr1		44h
Mask Address Register2		pci_mask_addr2		48h
Mask Address Register3		pci_mask_addr3		4Ch
Mask Address Register4		pci_mask_addr4		50h
Mask Address Register4		pci_mask_addr5		54h
Trans Address Register0		pci_trans_addr0		58h
Trans Address Register1		pci_trans_addr1		5Ch
Trans Address Register2		pci_trans_addr2		60h
Trans Address Register3		pci_trans_addr3		64h
Trans Address Register4		pci_trans_addr4		68h
Trans Address Register4		pci_trans_addr5		6Ch

5.3.1 北桥的 Device ID 和 Vendor ID 寄存器

由于北桥是采用 FPGA 来实现的，没有制造商和供应商。这两个 ID 寄存器为只读寄存器，返回值为：Dveice ID **0x00D5**，Vendor ID **0xDF53**

5.3.2 地址寄存器组

基地址寄存器

PCI 配置空间寄存器表所列，北桥提供了 5 个 **base** 地址寄存器。（实际中只用到了前 3 个）。

基地址寄存器的格式如下：

[31:4]	3	[2:1]	0
Base Address	P	00	0

“P”位用于指定内存空间。

掩码地址寄存器

北桥提供了 5 个 **mask** 掩码地址寄存器，实际只用了前三个。

第一个 **mask** 的最高 4 位为有效值；

第二个 **mask** 的最高 9 位为有效值；

第三个 **mask** 的最高 20 位为有效值；

传输地址寄存器

北桥提供了 5 个 **trans** 掩码地址寄存器，实际只用了前三个。

第一个 **trans** 的最高 4 位为有效值；

第二个 **trans** 的最高 9 位为有效值；

第三个 **trans** 的最高 20 位为有效值；

每一组 base/mask/trans 构成一组映射空间。具体的地址生成请参见本章 5.4.2 小节关于外部 PCI 访问北桥地址转换的介绍。

5.4 PCI 控制寄存器

当片外 PCI 设备访问北桥的时候，其地址需要经过 **Pcimembasecfg** 寄存器的控制与映射才能生成正确的系统的本地地址。

5.4.1 PCI 内存基地址配置寄存器

物理地址：**0x1FE0_0114**
读写类型：可读可写
复位值： **0x0000_0000**

Pcimembasecfg 寄存器图

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Null								Pcibase1 options							
								io	unused	trans					mark
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pcibase1 options				Pcibase0 options											
mark				io	unused	trans						mark			

Pcimembasecfg 位描述

位	名称		描述
[31:24]	Null		没有使用；默认为“0”
[23]	Pcibase1 Options	io	IO 或内存设备选择位； 为“0”时，表示将该窗口映射到内存空间； 为“1”时，表示将该窗口映射到 IO 空间； 默认值为“0”；暂不支持映射到 IO 空间。
[22]		unused	没有使用；
[21:17]		trans	传输地址；参与本地地址的生成过程。
[16:12]		mark	掩码地址；参与本地地址的生成过程。
[11]	Pcibase0 Options	io	IO 或内存设备选择位； 为“0”时，表示将该窗口映射到内存空间； 为“1”时，表示将该窗口映射到 IO 空间； 默认值为“0”；暂不支持映射到 IO 空间。
[10]		unused	没有使用；
[9:5]		trans	传输地址；参与本地地址的生成过程。
[4:0]		mark	掩码地址；参与本地地址的生成过程。

5.4.2 外部 PCI 访问北桥地址转换

在上一小节中，描述了 **Pcimembasecfg** 寄存器的各个位的功能。由 PCI 地址转换成本地系统地址的过程如下：

由于北桥最大提供 256MB 的空间，所以 PCI 地址的[31:28]不起作用，将该 4 位省略。那么[27:23]则定义 8MB 的窗口映射到 256MB 空间的那一段。PCI 地址的[27:23]位首先跟掩码地址（**Pcimembasecfg** 寄存器中 **mask** 的值）进行逻辑“与”运算，再与传输地址（**Pcimembasecfg** 寄存器中 **trans** 的值）进行逻辑“或”运算，得到的结果就是本地地址的[27:23]的地址位，那么 PCI 地址的[22:0]则直接赋给本地地址的[22:0]位。这样结合起来就生成本地地址的[27:0]位。

在生成系统本地地址[27:0]位的同时，配置空间的 base/trans/mask 寄存器组也在参与运算。PCI 的高位地址首先配置空间提供的 mask 地址进行逻辑“与”，而配置空间的 base 地址与 mask 地址也进行逻辑“与”，如果两者相同，则该映射空间被命中。输出地址的高位[27:23]则由 **Pcimembasecfg** 寄存器与 PCI 地址生成的[27:23]提供，而[31:28]位则由配置空间的 trans 地址提供。低[27:0]则由 PCI 地址提供。这样就生成了完整的访问内存的 32 位地址。

5.4.3 PCI 地址映射寄存器

物理地址：**0x1FE0_0110**
读写类型：可读可写
复位值： **0x0000_0000**

Pcimap 寄存器图

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Null													Pcimap_2		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pcimap_lo2				Pcimap_lo1						Pcimap_lo0					

Pcimap 位描述

位	名称	功能
[31:19]	Null	没有使用；
[18]	Pcimap_2	保留；
[17:12]	Pcimap_lo2	将标有 PCI_Lo2 的 64MB 内存空间映射到处理器的地址空间，可以分配给任何 64MB 的 PCI 内存空间。 出现在 PCI 总线的地址总线组成如下：低 26 位为处理器的物理地址的低 26 位，高 6 位则来自 pcimap_lo2; pcimap_lo2 是访问 PCI_Lo2 空间的窗口，可以放置在 PCI 空间任何一个 64MB 的边界上。
[11:6]	Pcimap_lo1	将标有 PCI_Lo1 的 64MB 内存空间映射到处理器的地址空间，可以分配给任何 64MB 的 PCI 内存空间。 出现在 PCI 总线的地址总线组成如下：低 26 位为处理器的物理地址的低 26 位，高 6 位则来自 pcimap_lo1; pcimap_lo1 是访问 PCI_Lo1 空间的窗口，可以放置在 PCI 空间任何一个 64MB 的边界上。
[5:0]	Pcimap_lo0	将标有 PCI_Lo0 的 64MB 内存空间映射到处理器的地址空间，可以分配给任何 64MB 的 PCI 内存空间。 出现在 PCI 总线的地址总线组成如下：低 26 位为处理器的物理地址的低 26 位，高 6 位则来自 pcimap_lo0; pcimap_lo0 是访问 PCI_Lo0 空间的窗口，可以放置在 PCI 空间任何一个 64MB 的边界上。

5.4.4 PCI 配置空间地址映射寄存器

物理地址：**0x1FE0_0118**
读写类型：可读可写
复位值： **0x0_0000**

Pcimap_cfg 寄存器图

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Null															Type
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AD16UP															

Pcimap_cfg 位描述

位	名称	功能
[16]	Type	为‘0’表示为“type0”型的正常的配置周期； 为‘1’表示为“type1”型的 PCI-to-PCI 配置周期；
[15:0]	AD16UP	在配置周期中，该16位的数据映射为AD总线高位[31:16]位的数据值。 当对系统物理地址空间 0x01FE8_0000 到 0x01FE_FFFF 范围进行读写操作，既是对外部 PCI 配置空间访问，会被映射到 PCI 总线上的配置空间访问，并驱动 PCI 总线。

当对外部 PCI 配置空间访问时，形成的 PCI 配置访问的地址周期的格式如下：

[31:16]	[15:2]	1	0
Pcimap_cfg[15:0]	Sys_Addr[15:2]	0	Pcimap_cfg[16]

5.4.5 PCI 特殊周期命令寄存器

物理地址：**0x1FE0_0148**
读写类型：只写
复位值： **0x0000_0000**

Spcial_cycle 寄存器图

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

Spcial_cycle 位描述

位	名称	功能
[31:16]	消息内容	由消息决定的可选数据
[15:0]	消息类型	该 16 位为消息类型编码。具体的编码参考 PCI 协议 2.2 版本

5.5 Local IO 配置寄存器

Local IO 配置寄存器为 **iodevcfg，**其功能主要是对外挂在 Local IO 总线上的设备的读写速度进行配置。

5.5.1 iodevcfg 寄存器

物理地址：**0x1FE0_0008**
读写类型：可读可写
复位值： **0x2BFF_8010**

iodevcfg 寄存器图

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
cpuclockperiod						Null									
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Null					Speedbit_cs3	Null		Speedbit_cs2	Null		Speedbit_cs1	Null		Speedbit_cs0	Null

iodevcfg 位描述

位	名称	描述
[31:26]	cpuclockperiod	上电复位时对 **iodevcfg** 寄存器[31:26]位进行配置，从而确定外部 Local IO 总线上设备的读写速度等级。根据读写速度等级确定外设的读写速度。 **Iodevcfg**[31:26]上电缺省配置值为 6’b001010； 关于详细配置见 **4.4.1** 小节的介绍。
[25:16]	Null	没有使用；
[15:11]	Null	没有使用；
[10]	Speedbit_cs3	值为“1”，表示设备为高速设备；读写周期默认为 200ns； 值为“0”，表示设备为低速设备；读写周期默认为 600ns； 默认值为“0”，低速设备。
[9:8]	Null	没有使用；
[7]	Speedbit_cs2	值为“1”，表示设备为高速设备；读写周期默认为 200ns； 值为“0”，表示设备为低速设备；读写周期默认为 600ns； 默认值为“0”，低速设备。
[6:5]	Null	没有使用；
[4]	Speedbit_cs1	值为“1”，表示设备为高速设备；读写周期默认为 200ns； 值为“0”，表示设备为低速设备；读写周期默认为 600ns； 默认值为“1”，高速设备。
[3;2]	Null	没有使用；
[1]	Speedbit_cs0	值为“1”，表示设备为高速设备；读写周期默认为 200ns； 值为“0”，表示设备为低速设备；读写周期默认为 600ns； 默认值为“0”，低速设备。
[0]	Null	没有使用；

5.6 中断控制寄存器

北桥集成了一个中断控制器，包含 7 个控制寄存器，中断状态寄存器（intisr），中断使能寄存器（inten），设置中断使能寄存器（intenset），清零中断使能寄存器（intenclr），中断源极性寄存器（intpol），中断源触发方式寄存器（intedge）和中断导向寄存器（intsteer）。

同时向处理器提交 7 个中断信号。

5.6.1 intisr 中断状态寄存器

物理地址：**0x1FE0_013C**

读写类型：只读

复位值： XXX

intisr 寄存器图

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

intisr 位描述

位	中断源	名称	描述
[31:25]	Gpin[6:0]	intisr	当一个中断源的中断条件满足时，**intisr** 寄存器中对应的位被置位**(**置 **1)**。 如果外部中断被配置成边沿触发方式，该寄存器得到该中断源对应的内部锁存器的状态。
[25:16]	Gpio[8:0]		
[15:12]	Reserved		
[11]	Systemerr		
[10]	Mastererr		
[9]	Reserved		
[8:0]	Null		

5.6.2 inten 中断使能寄存器

物理地址：**0x1FE0_0138**
读写类型：不能直接写，可读
复位值： **0x0000_0000**

inten 寄存器图

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

inten 位描述

位	中断源	名称	描述
[31:25]	Gpin[6:0]	inten	如果一个中断源被允许产生中断，那么其在 inten 寄存器中对应的位被置位。 中断使能寄存器的置位和复位(置零，清位)分别通过 intenset 寄存器和 intenclr 寄存器实现，不能直接写 inten 寄存器。
[25:16]	Gpio[8:0]		
[15:12]	Reserved		
[11]	Systemerr		
[10]	Mastererr		
[9]	Reserved		
[8:0]	Null		

5.6.3 intenset 设置中断使能寄存器

物理地址：**0x1FE0_0130**
读写类型：只写
复位值： **0x0000_0000**

intenset 寄存器图

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

intenset 位描述

位	中断源	名称	描述
[31:25]	Gpin[6:0]	intenset	在该寄存器的某些位写入 **1**，则会将 **inten** 寄存器中对应的位置位**(**置为 **1)**； 如果写入 **0**，那么 **inten** 中对应的位保持不变。
[25:16]	Gpio[8:0]		
[15:12]	Reserved		
[11]	Systemerr		
[10]	Mastererr		
[9]	Reserved		
[8:0]	Null		

5.6.4 intenclr 清零中断使能寄存器

物理地址：**0x1FE0_0134**
读写类型：只写
复位值： **0x0000_0000**

intenclr 寄存器图

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

intenclr 位描述

位	中断源	名称	描述
[31:25]	Gpin[6:0]	intenclr	在该寄存器的某些位写入 **1**，则会将 **inten** 寄存器中对应的位复位**(**置为 **0)**；如果写入 **0**，那么 **inten** 中对应的位保持不变。 如果采用边沿触发方式，则在 **intenclr** 寄存器的某位写入 **1**，则会将对应的边沿触发检测锁存器的值复位。
[25:16]	Gpio[8:0]		
[15:12]	Reserved		
[11]	Systemerr		
[10]	Mastererr		
[9]	Reserved		
[8:0]	Null		

5.6.5 intpol 中断源极性寄存器

物理地址：**0x1FE0_012C**
读写类型：可读可写
复位值： **0x0000_0000**

intpol 寄存器图

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

intpol 位描述

位	中断源	名称	描述
[31:25]	Gpin[6:0]	intpol	值为 **1** 表示对应的中断源高电平有效，值为 **0** 表示对应的中断源低电平有效。 系统复位**(reset)**时 **intpol** 寄存器的值全为 **0**，因此需要对 **intpol** 寄存器进行编程，消除高电平有效的中断源在低电平时**(**没有中断**)**的假中断现象。 注意，所有的内部中断信号高电平有效**(**采用边沿触发方式**)**，同时，**intpol** 寄存器的部分位可以修改**(**不可修改的位即使修改它也不会有任何效果，如果去读它还是原来的值**)**。
[25:16]	Gpio[8:0]		
[15:12]	Reserved		
[11]	Systemerr		
[10]	Mastererr		
[9]	Reserved		
[8:0]	Null		

5.6.6 intedge 中断源触发方式寄存器

物理地址：**0x1FE0_0124**
读写类型：可读可写
复位值： **0x0000_0000**

intedge 寄存器图

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

intedge 位描述

位	中断源	名称	描述
[31:25]	Gpin[6:0]	intedge	中断触发方式寄存器； 值为 1 表示对应的中断源采用边沿触发方式，值为 0 表示对应的中断源采用电平触发方式。 电平触发方式时直接使用输入信号，边沿触发方式时选用边沿检测锁存器的状态。对 intedge 寄存器进行编程(改变)时，边沿检测锁存器的状态保持不变。 注意，该寄存器的部分位可以修改；除了 timer 中断、retryerr、systemerr、mastererr，目前的逻辑都采用电平触发方式。
[25:16]	Gpio[8:0]		
[15:12]	Reserved		
[11]	Systemerr		
[10]	Mastererr		
[9]	Reserved		
[8:0]	Null		

5.6.7 intsteer 中断导向寄存器

物理地址：**0x1FE0_0128**
读写类型：可读可写
复位值： **0x0000_0000**

intsteer 寄存器图

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

intsteer 位描述

位	中断源	名称	描述
[31:25]	Gpin[6:0]	intsteer	中断导向寄存器； 中断源对应位的值为 1 表示该中断源产生的中断向引脚 nCint1(对应 CPU 的 int1#引脚)输出，值为 0 表示中断向引脚 nCint0(对应 CPU 的 int0#引脚)输出。 注意，如果将 bongencfg 寄存器的 irqa_from_int1 位置 1，那么中断输出信号 nCint1 同时向 PCI 的引脚 INTA 输出。
[25:16]	Gpio[8:0]		
[15:12]	Reserved		
[11]	Systemerr		
[10]	Mastererr		
[9]	Reserved		
[8:0]	Null		

5.7 GPIO 配置寄存器

GPIO 配置寄存器有两个 GPIODATA 寄存器和 GPIOEnable 寄存器。GPIODATA 是反映管脚 GPIN[7:0]和 GPIO[8:0]的值；GPIOEnable 则控制 GPIO[8:0]的输入输出方向。具体功能定义见下表。

5.7.1 GPIODATA 寄存器配置

物理地址：**0x1FE0_011C**
读写类型：可读可写
复位值： **0x0000_01FF**

GPIODATA 寄存器图

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

GPIODATA 位描述

位	名称	描述
[31:25]	gpins	只读；通用输入管脚 GPIN[6:0]的值
[24:16]	gpios	只读；通用输入输出管脚 GPIO[8:0]的输入值
[15:9]	Reserved	保留；默认值全为“1”
[8:0]	GPIODATA	可读写；通用输入输出管脚 GPIO[8:0]的输出值

5.7.2 GPIOEnable 寄存器配置

物理地址：**0x1FE0_0120**
读写类型：可读可写
复位值： **0x0000_01FF**

GPIOEnable 寄存器图

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

GPIOEnable 位描述

位	名称	描述
[31:9]	Reserved	保留；默认值全为“0”
[8:0]	GPIOEnable	GPIO 方向使能信号； 每个位上的值为“0”时，表示相应 GPIO 管脚信号为输出； 每个位上的值为“1”时，表示想应 GPIO 管脚信号为输入； 默认值全为“1”；

5.8 Timercfg 配置寄存器

物理地址：**0x1FE0_0160**
读写类型：可读可写
复位值： **0x0000_0000**

timercfg 寄存器图

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

timercfg 位描述

位	名称	描述
[31:0]	Reserved	保留；

5.9 上电配置寄存器

上电配置寄存器和上电配置值寄存器是两个控制系统启动的寄存器；主要用来配置上电启动时 ROM 的选择，ROM 的数据宽度，读取 ROM 内容的速度，以及 PCI 设备复位信号的控制等等；

5.9.1 上电配置寄存器

物理地址：**0x1FE0_0104**
读写类型：可读可写
复位值：

bonponcfg 寄存器图

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rserved				RomCS1fast	RomCS0fast	RomCS1width	RomCS0width	romboot		Reserved		pcireset	Reserved		

bonponcfg 位描述

位	名称	描述
[31:12]	Reserved	保留；没有使用
[11]	RomCS1fast	根据该选择信号来选择是否对 ROM 进行高速的读操作； 低速操作时间是 160ns； 高速操作时间是 90ns； 当从 ROM 读取数据时，不能改变该位的值； 当 cfg_sel 是低的时候，默认值为“11”；否则默认值为等效于启动时 iod[11:10]的值； 龙芯 2E 北桥只有一个 Romcs ，所以 romCs1fast 没有使用；
[10]	RomCS0fast	
[9]	RomCS1width	这两位为只读位；（数据宽度选择）‘1’ 选择 16 位的数据宽度，‘0’ 则选择 8 位的数据宽度； 当 cfg_sel 是低的时候，默认值为“00”；否则默认值为等效于启动时 iod[9:8]的值； 龙芯 2E 北桥只有一个 Romcs ，所以 romCs1fast 没有使用；
[8]	RomCS0width	
[7:6]	romboot	处理器启动选择；就是选择从哪部分存储区域来启动。一般都是从 0x1fc00000 作为启动地址选择。具体选择如下： “00”： 根据 romcs1 从 ROM 低 4M 存储空间启动；

		“01”： 根据 romcs0 从 ROM 高 4M 存储空间启动； “10” ： 从本地 SDRAM 启动。（注意：只有当 SDRAM 中存在部分带有 MIPS 启动代码的系统，才能从该区域启动；） 还可以从下面所述区域启动； “11” ： 当 cfg_sel 是低的时候，默认值为“01”；否则默认值为等效于启动时 iod[7:6]的值； 现阶段的龙芯 2E 的北桥只有一个 romcs ，启动选择不应被改变；
[5:4]	Reserved	保留；没有使用
[3]	pcireset	如果龙芯 2E 北桥作为 PCI 总线控制器（例如：控制 PCI 设备复位信号），这位就是设置该功能的有效位置。 ‘0’表示启动低电平有效的 PCI 复位信号；‘1’表示忽略； 默认值为‘0’；
[2:0]	Reserved	保留；没有使用

5.9.2 上电默认值配置寄存器

物理地址：**0x1FE0_0100**
读写类型：可读可写
复位值： **0x0000_1384**

bongencfg 寄存器图

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

bongencfg 位描述

位	名称	描述
[31:18]	Null	没有使用；
[17:4]	bongencfg	保留；默认值为 14’b00_0001_0011_1000；
[3]	bongencfg	bongencfg[3]则连接到上电启动信号管脚（pwr_btn）上；
[2]	bongencfg	当处理器复位模块检测到 bongencfg[2]的上升跳变沿时，就会生成处理器软复位信号；
[1:0]	bongencfg	保留；默认值为 2’b01；

第六章 FPGA 配置

6.1 FPGA 配置与配置芯片

北桥是采用 FPGA 芯片 EP2C20F484 以及 FPGA 配置芯片 EPCS4 来实现的。FPGA 用于逻辑设计的实现，配置芯片用于存储配置数据。

6.1.1 FPGA 配置信号

FPGA 的配置信号主要是将片外配置芯片存储的配置数据在上电的时候配置到 FPGA 芯片，从而使 FPGA 芯片正常工作。

表 6.1.1 FPGA 配置信号

管脚名称	位置	方向	电压标准	功能描述
ASDO	C4	output	LVTTL	连接配置芯片 ADSI
nCSO	W20	output	2.5 V	连接配置芯片 nCS
DATA0	K4	input	LVTTL	连接配置芯片 DATA
DCLK	L6	output	LVTTL	连接配置芯片 DCLK
TCK	K2	input	LVTTL	JTAG 时钟信号
TMS	K5	input	LVTTL	JTAG 模式信号
TDI	L5	input	LVTTL	JTAG 数据输入
TDO	K6	output	LVTTL	JTAG 数据输出
CONF_DONE	N18	bidir	LVTTL	配置控制信号，应用时上拉。
MSEL[1:0]	N17,M17	input	LVTTL	模式选择信号，有效值为[10]
nCE	K1	input	LVTTL	配置控制信号，应用时下拉。
nSTATUS	N20	bidir	LVTTL	配置控制信号，应用时上拉。
nCONFIG	L4	bidir	LVTTL	配置控制信号，应用时上拉。

这里只是列出这些管脚的连接方式。具体的功能定义请参考 Altera 公司的芯片文档。

6.1.2 配置芯片

配置芯片 EPCS4 容量为 4Mbit，其封装为 SOI8 型，管脚分配如下图：

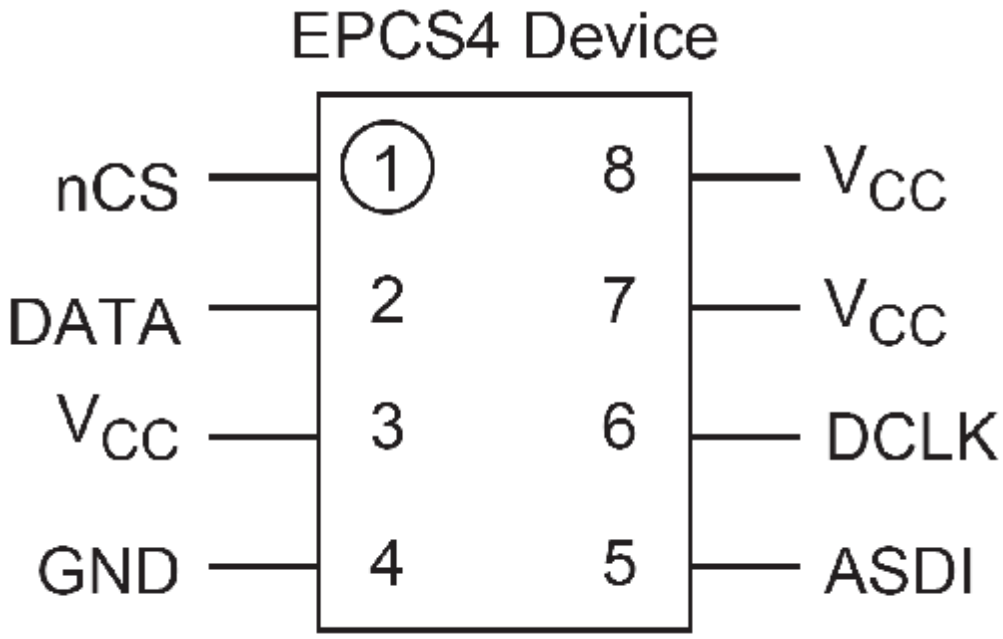


图 6.1.2 EPCS4 管脚分配图

配置芯片与 FPGA 管脚连接如下表：

表 6.1.2 FPGA 与配置芯片管脚连接

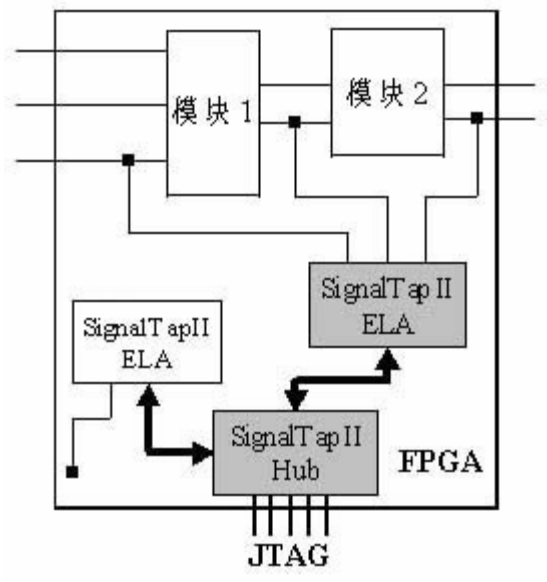
FPGA 芯片	**配置芯片**
nCSO	nCS
DATA0	DATA
DCLK	DCLK
ASDO	ASDI

6.2 北桥调试介绍

由于北桥采用 FPGA 实现，所以在系统研发期间，可以使用 SignTapII 信号抓取工具抓取北桥所有 I/O 引脚和内部任意结点的信号波形。

6.2.1 SignTapII 工具介绍

Sign TapII 是 Altera 公司的 QuartusII 软件中集成的一种基于逻辑分析核的嵌入式逻辑分析仪。它具有普通逻辑分析仪的功能，包括触发、数据采集和存储等。利用逻辑分析核，用户可以访问 FPGA 器件内部所有信号和节点。如下图所示：



由于龙芯 2E 北桥采用的是 Altera 公司的 FPGA 实现的，因此在产品研发期间，尤其是硬件调试阶段可以借助该工具缩减系统开发的调试时间。

6.2.2 调试流程简介

使用 **SignalTap II** 的一般流程是：设计人员在完成设计并编译工程后，建立 **SignalTap II (.stp)**文件并加入工程、配置 **STP** 文件、编译并下载设计到 **FPGA**、在 **Quartus II** 软件中显示被测信号的波形、在测试完毕后将该逻辑分析仪从项目中删除。以下描述设置 **SignalTap II** 文件的基本流程：

1. 设置采样时钟。采样时钟决定了显示信号波形的分辨率，它的频率要大于被测信号的最高频率，否则无法正确反映被测信号波形的变化。**SignalTap II** 在时钟上升沿将被测信号存储到缓存。

2. 设置被测信号。可以使用 **Node Finder** 中的 **SignalTap II** 滤波器查找所有预综合和布局布线后的 **SignalTap II** 节点，添加要观察的信号。逻辑分析器不可测试的信号包括：逻辑单元的进位信号、**PLL** 的时钟输出、**JTAG** 引脚信号、**LVDS**（低压差分）信号。

3．配置采样深度、确定 **RAM** 的大小。**SignalTap II** 所能显示的被测信号波形的时间长度为 **Tx**，计算公式如下：

Tx=N×Ts ； **N** 为缓存中存储的采样点数，**Ts** 为采样时钟的周期。

4．设置 **buffer acquisition mode**。**buffer acquisition mode** 包括循环采样存储、连续存储两种模式。循环采样存储也就是分段存储，将整个缓存分成多个片段**(segment)**，每当触发条件满足时就捕获一段数据。该功能可以去掉无关的数据，使采样缓存的使用更加灵活。

5．触发级别。**SignalTap II** 支持多触发级的触发方式，最多可支持 **10** 级触发。

6．触发条件。可以设定复杂的触发条件用来捕获相应的数据，以协助调试设计。当触发条件满足时，在 **signalTap** 时钟的上升沿采样被测信号。

完成 **STP** 设置后，将 **STP** 文件同原有的设计下载到 **FPGA** 中，在 **Quartus II** 中 **SignalTap II** 窗口下查看逻辑分析仪捕获结果。**SignalTap II** 可将数据通过多余的 **I/O** 引脚输出，以供外设的逻辑分析器使用。

SignalTAP II 工具可以单独使用，也可以嵌入到 QuartusII 中使用。对于基于龙芯 2E 的系统学习人员和研发人员，采用该工具可以更加方便、快速的了解底层硬件的结构与原理。

第七章 封装

7.1 封装外形尺寸表

表 7.1 封装外形尺寸表

封装外形尺寸表			
符号	毫米		
	最小	典型	最大
A	---	---	2.60
A1	0.30	---	---
A2	---	---	2.20
A3	---	---	1.80
D	23.00 BSC		
E	23.00 BSC		
b	0.50	0.60	0.70
e	1.00 BSC		

7.2 封装视图

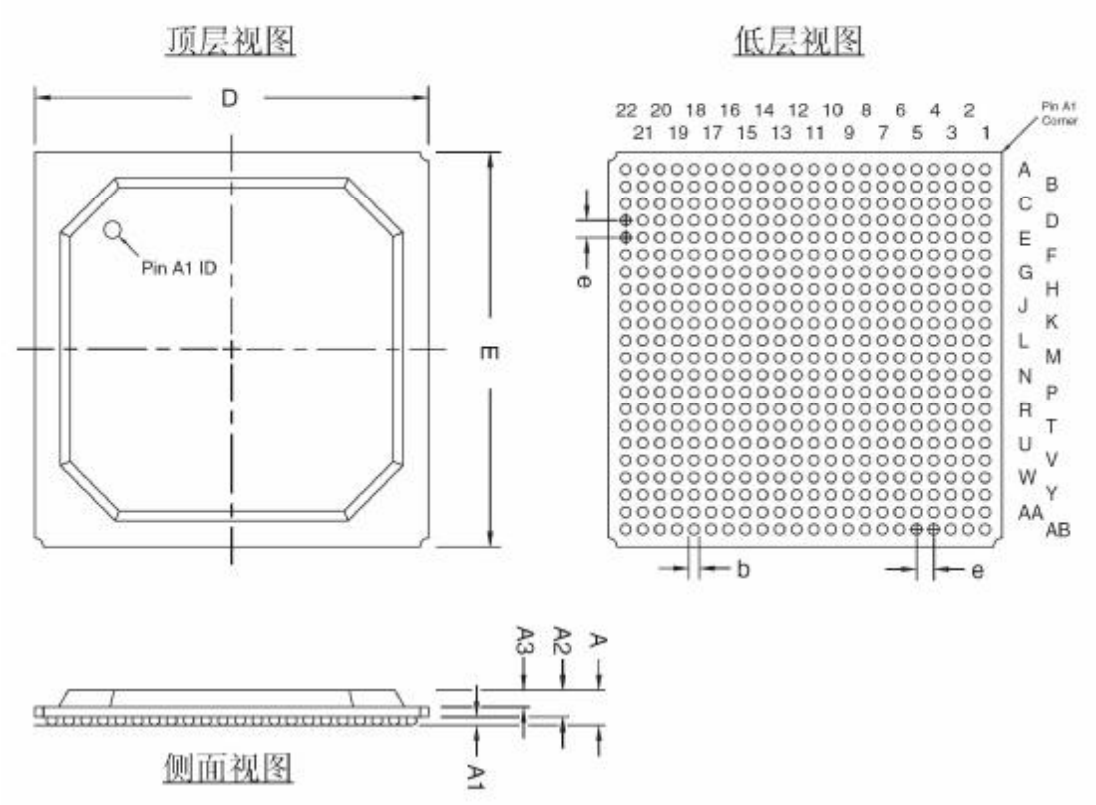


图 7.2 封装视图