目　录

第一章　PCI 总线概述

1.1　总线的基本概念

计算机总线是计算机各部件之间进行信息传输的公共通道。微型计算机系统中广泛采用总线结构，其优点是系统成本低、组态灵活、维修方便。采用总线标准设计、生产的硬件模块兼容性强，并通过系统总线可以方便地组合在一起，以构成满足不同需要的微机系统。

计算机总线技术包括通道控制功能、使用方法、仲裁方法和传输方式等。任何系统的研制和外围模块的开发，都必须服从一定的总线规范。总线的结构不同，性能差别很大。计算机总线的主要职能是负责计算机各模块间的信息传输，因此，对总线性能的衡量，也是围绕着这一职能而定义、测试和比较的。总线的传输率是其性能的主要技术指标。另外，总线的可操作性、兼容性和性能价格比，也是很重要的技术特征。

随着计算机技术的不断发展，微型计算机的体系结构发生了显著的变化。如 CPU 运行速度的提高，多处理器结构的出现，高速缓冲存储器的广泛采用等，都要求有高速的总线来传输数据，从而出现了多总线结构。多总线结构是指 CPU 与存储器、I/O 等设备之间有两种以上的总线，这样可以将慢速的设备和快速的设备挂在不同的总线上，减少总线竞争现象，使系统的效率大大提高。

在多总线结构中，局部总线(Local Bus)的发展最令人瞩目。局部总线是指来自处理器的延伸线路，与处理器同步操作。外部

设备如果直接挂到局部总线上，就能以 CPU 的速度运行。由于局部总线具有极高的数据传输率，因此在 CPU 与高速缓冲存储器(Cache)、CPU 与高速图形卡等需要高速传输信息的场合得到了广泛的应用。

1.2 PCI 总线的发展

1.2.1 PCI 总线的孕育

PCI 总线的英文全称为：Peripheral Component Interconnect Special Interest Group，简称 PCISIG。即外设部件互连。PCI 总线虽然是由 Intel 公司提出的，但说到其发展，必须由 IBM 公司说起。由于 IBM PC 机系统的开放性，全世界 PC 机的制造商纷纷向 IBM PC 标准靠拢，从而使 IBM PC 系列产品风靡全球。与此同时，Intel 公司和 Microsoft 公司也迅速发展壮大起来，对 IBM 公司构成了威胁。IBM 公司为保护自身利益，将计算机总线由 ISA 总线升级到 MCA 总线，并于 1987 年 4 月在 PS/2 机上使用。MCA 是 32 位总线，传输率为 40 MB/s，可共享资源，具有多重处理能力。为防止其它厂家的仿制，IBM 公司没有对外公开 MCA 总线的技术标准，从而使其成为专有产品。

鉴于上述原因，Compaq、AST、Epson、HP、Olivetti 和 NEL 等 9 家公司联合于 1988 年 9 月推出了一种兼容性更强的总线，即 EISA 总线。该总线除了具有与 MCA 总线完全相同的功能外，还与 ISA 总线 100%兼容。EISA 是 32 位总线，支持多处理器结构，具有较强的 I/O 扩展能力和负载能力，传输率为 33 MB/s，适用于网络服务器，高速图像处理，多媒体等领域。因 EISA 是兼容商共同推出的，所以其技术标准是公开的。

1991 年下半年，Intel 公司首先提出了 PCI 总线的概念，并与

IBM、Compaq、AST、HP、DEC 等 100 多家公司联合共谋计算机总线的发展大业，于 1993 年推出了 PC 局部总线标准——PCI 总线。

1.2.2 PCI 总线发展的动力

PCI 总线支持 64 位数据传送、多总线主控和线性突发方式(Burst)，其数据传输率为 132 MB/s。这给其发展提供了有利条件。总的来看，PCI 总线之所以能发展，其动力之一是 GUI(Graph User Interface)的发展。良好的用户接口界面的实现是以高性能的图形界面操作系统为基础的，而图形界面操作系统往往需要大容量存储器，因而，刺激了 RAM 芯片的生产，更重要的是对总线的性能提出了更高的要求。例如：

在多媒体视频图像显示中，若分辨率为 640×480，每秒 30 帧，显示彩色深度为 24 位，则

$$\text{多媒体显示卡的数据吞吐量} = 640\times 480\times 30\times 3 = 27.648\ \text{Mb/s}$$

对于具有 100 Mb/s 传输率的高速光纤网，需要总线的吞吐量为

$$100\ \text{Mb/s} = 12.5\ \text{MB/s}$$

由此可见，采用 100 Mb/s 光纤传输视频动态图像必须借助于压缩技术。

由于外围设备数据吞吐量与总线传输率之间没有严格的比例关系，一般一条总线可能挂接 3～5 个高速外设，因而总线的最大传输率应为高速外设的 3～5 倍。由此可计算出多媒体视频卡对总线最大传输率的需求为

$$\text{Tran Multimadia} = 27.648\times(3\sim5) = 82.944\sim138.24\ \text{MB/s}$$

而 100 MB/s 的高速光纤网络中，对总线最大传输率的需求为

$$\text{Tran FDDI} = 12.5\times(3\sim5) = 37.5\sim62.5\ \text{MB/s}$$

但 ISA 总线的最大传输率为 8 MB/s，EISA 总线为 33 MB/s，无法满足图形操作系统和高速网络的要求。而 PCI 总线的传输率为 132 MB/s，可满足上述要求。

另一推动 PCI 总线发展的原因是它可以降低系统成本。用大量面向 PCI 总线的处理芯片来构造系统机、工作站、外围设备及板卡，其性能优越，处理能力、传输速度都很高。反之，若不采用面向 PCI 的芯片进行设计，实现同样的功能，其成本将升高 10%～15%。

1.2.3 总线的性能指标

要评价一个总线的性能好坏，只有通过相应的指标才能做出。一般采用如下指标来进行评价：

(1) 总线宽度：数据总线的数量，用 bit(位)表示，如 8 位、16 位、32 位、64 位。

(2) 传输率：每秒钟在总线上传输的最大字节数，用 MB/s 表示，即每秒多少兆字节。若总线工作频率为 8 MHz，总线宽度为 8 位，则最大传输率为 8 MB/s。若工作频率为 33 MHz，总线宽度为 32 位，则最大传输率为 132 MB/s。

(3) 同步方式：总线上的数据与时钟同步工作的总线，称为同步总线。反之，称为异步总线。

(4) 信号线数：表明总线所需信号线数的多少，是地址线 AB、数据线 DB、控制线 CB 的总和。信号线数与性能不成正比，但与复杂程度成正比。

(5) 数据总线/地址总线的多路复用和非多路复用：地址线传输地址码，数据总线传输数据命令。为了提高总线性能，优化设计，采用了地址线和数据线共用一条物理线路，即某一时刻该线路上传输的是地址信号，而另一时刻传输的是数据信号或总线命令。这种一条总线多种用途的技术，称作多路复用。

（6）负载能力：通常指“可连接的扩增电路板数”或“可连接的扩充电路板的数量”，尽管大家沿用这一表示方法，其实并不严密，不过它基本上能反映出总线的负载能力。

（7）总线控制方式：主要指突发传输、并发工作、自动配置、中断方式、仲裁方式等。

（8）扩充电路板尺寸：表示某一总线扩展电路板的尺寸大小。

（9）其它指标：如电源电压等级，能否扩展为 64 位宽度等，也是很重要的参数。

表 1.1 简要给出几种总线的有关性能参数。

1.3 PCI 总线的特点

PCI 是先进的高性能局部总线，可同时支持多组外围设备。PCI 局部总线不受制于处理器，为中央处理器及高速外围设备提供一座桥梁，更可作为总线之间的交通指挥员，提高数据吞吐量。PCI 采用高度综合化的局部总线结构。其优化的设计可充分利用今日最先进的微处理器及个人电脑科技。它可确保电脑部件、附加卡及系统之间的运作可靠，并能完全兼容现有的 ISA/EISA/Micro Channel 扩充总线。总之，PCI 局部总线具有如下特点：

1. 高性能

PCI 是一套整体的系统解决方案，较其它只为加速图形或视频操作的局部总线优越。它能提高网络界面卡、硬盘的性能；可以出色地配合全活动影像、图形及各种高速外围设备的要求。PCI 局部总线以 33 MHz 的时钟频率操作，采用 32 位数据总线，可支持多组外围部件及附加卡。数据传送速率可高达 132 MB/s，远远超过标准 ISA 总线 5 MB/s 的速率。即使在 32 位的情况下，

表 1.1 几种计算机总线性能一览表

名　称	PC—XT	ISA(PC—AT)	EISA	STD	VESA(VL—BUS)	MCA	PCI	HI
适用机型	8086 个人计算机	80286、386、486 系列个人计算机	IBM 系列 386、 486、586 计算机	Z—80、V20、V40 IBM—PC 系列机	i486、PC—AT 兼容个人计算机	IBM 公司个人 机与工作站	P5 个人机、Power PC、Alpha 工作站	日立工作站
最大传输率	4 MB/s	16 MB/s	33 MB/s	2 MB/s	266 MB/s	40 MB/s	133 MB/s	133 MB/s
总线宽度	8 位	16 位	32 位	8 位	32 位	32 位	32 位	32 位
总线工作频率	4 MHz	8 MHz	8.33 MHz	2 MHz	66 MHz	10 MHz	0—33 MHz	20—33.3 MHz
同步方式			同步			异步	同步	
仲裁方式	集中	集中	集中	集中	集中			
逻辑时序	边缘敏感	边缘敏感		边缘敏感	电平敏感		边缘敏感	电平敏感
地址宽度	20	24	32	20			32/64	
负载能力	8	8	6	无限制	6	无限制	3	7
信号线数			143		90	109	49	137
64 位扩展	不可	不可	无规定	不可	可	可	可	无规定
自动配置	无	无		无			可	可
并发工作					可		可	可
突发方式							可	
引脚使用	非多路复用	非多路复用	非多路复用	非多路复用	非多路复用		多路复用	

也能支持奔腾(Pentium)级电脑的图形数据传送速率。

2. 线性突发传输

PCI 能支持一种称为线性突发的数据传输模式，可确保总线不断满载数据。外围设备一般会由内存某个地址顺序接收数据，这种线性或顺序的寻址方式，意味着可以由某一个地址起读写大量数据，然后每次只需将地址自动加 1，便可接收数据流内下一个字节的数据。线性突发传输能够更有效地运用总线的带宽去传送数据，以减少无谓的地址操作。

另外，PCI 最独特之处是可以支持突发读取及突发写入，这对使用高性能图形加速器尤为重要。

3. 极小的存取延误

支持 PCI 的设备，存取延误很小，能够大幅度减少外围设备取得总线控制权所需的时间。例如，连接局部网络的以太网控制器，其缓冲区随时需要由网络接收大型档案，由于要等待使用总线的批准，从而使以太网界面卡往往无法及时在缓冲区溢出之前迅速将数据送给中央处理器，网络界面卡被迫将文件内容存在额外的内存区。对于 PCI 兼容的外围设备，由于它能提供更快速的存取，因此以太网卡可及时将数据传至中央处理器，减少所需的额外内存，从而降低附加卡的整体成本。

4. 采用总线主控和同步操作

PCI 的总线主控和同步操作功能有利于 PCI 性能的改善。总线主控是大多数总线都具有的功能，目的是让任何一个具有处理能力的外围设备暂时接管总线，以加速执行高吞吐量、高优先级的任务。PCI 独特的同步操作功能可保证微处理器能够与这些总线主控同时操作，不必等待后者的完成。

5. 不受处理器限制

PCI 独立于处理器的结构，形成一种独特的中间缓冲器设计方式，将中央处理器子系统与外围设备分开。一般来说，在中央

处理总线上增加更多的设备或部件，只会降低性能和可靠程度。而有了缓冲器的设计方式，用户可随意增添外围设备，以扩展电脑系统而不必担心在不同时钟频率下会导致性能的下降。

独立于处理器的总线设计还可保证处理器技术的变化不会使任何个别系统的设计变得过时，使消费者大为受惠。

6. 适合于各种机型

PCI 局部总线不只是为标准的桌面(台式)电脑提供合理的局部总线设计，同时也适用于便携式电脑和服务器。它可为便携式电脑及笔记本电脑提供台式电脑的图形性能，又可支持 3.3 V 的电源环境，延长电池寿命，为电脑的小型化创造了良好的实现条件。PCI 可缩小零件的尺寸，减少零件的数目，从而节省了宝贵的线路板空间，可使系统设计者在其产品中加入更多功能。

在服务器环境下，PCI 支持分级式外围设备的特性，可使一个 PCI 界面支持一组级联的 PCI 局部总线；也可以使设置为多组 PCI 总线的服务器增添额外的扩展插槽，提供更多的 I/O 接口，并将高带宽与低带宽的数据分隔开来。

7. 兼容性强

由于 PCI 的设计是要辅助现有的扩展总线标准，因此它与 ISA、EISA 及 MCA 总线完全兼容。虽然现有电脑系统的插槽数目有限，但 PCI 局部总线可提供"共用插槽"，以便接插一个 PCI、ISA、EISA 及 MCA 插头。这种兼容能力能保障用户的投资，让用户在继续使用沿用的附加卡之余，又能提供额外的插槽，方便用户选用新的外围设备。

8. 预留了发展空间

PCI 总线在开发时预留了充足的发展空间，这是它的一项重要特性。例如，它支持 64 位地址/数据多路复用。这是考虑到新一代的高性能外围设备最终将需要 64 位宽的数据通道。PCI 的 64 位延伸设计，可将系统的数据传输速率提高到 264 MB/s。同

时，由于 PCI 插槽能同时接插 32 位和 64 位插卡，所以，32 位与 64 位外围设备之间的通信是在用户不知不觉间进行的，从而做到了真正的瞻前顾后兼容。

PCI 还提供了自动配置功能，从而保证了用户在安装外围卡时，不需要手工调整跨接线。

9. 低成本、高效益

PCI 的芯片将大量系统功能高度集成，节省了逻辑电路，耗用较小的线路板空间，成本降低。PCI 部件采用地址/数据线复用，从而使 PCI 部件用以连接其它部件的引脚数减至 50 以下。

10. 是立足现在放眼未来的标准

PCI 局部总线既迎合了当今的技术要求，又能满足未来的需要，是计算机界公认的最具高瞻远瞩的局部总线标准。PCI 的高性能、高效率及与现有标准的兼容性和充裕的发展潜力，是其它总线不可及的。它可作为当今及未来的设计指引。

1.4 PCI 总线的系统结构

在一个 PCI 系统中可以做到：高速外部设备和低速外部设备共存，PCI 总线与 ISA/EISA 总线并存。如图 1.1 所示。

在图 1.1 中可以看到，处理机/Cache/存储器子系统经过一个 PCI 桥连接到 PCI 总线上。此桥提供了一个低延迟的访问通路，从而使处理器能够直接访问通过它映射于存储器空间或 I/O 空间的 PCI 设备；也提供了能使 PCI 主设备直接访问主存的高速通路；该桥也能提供数据缓冲功能，以使 CPU 与 PCI 总线上的设备并行工作而不必相互等待；另外，桥可使 PCI 总线的操作与 CPU 总线分开，以免相互影响。总之，桥实现了 PCI 总线的全部驱动控制。

扩展总线桥(标准总线接口)的设置是为了能在 PCI 总线上接

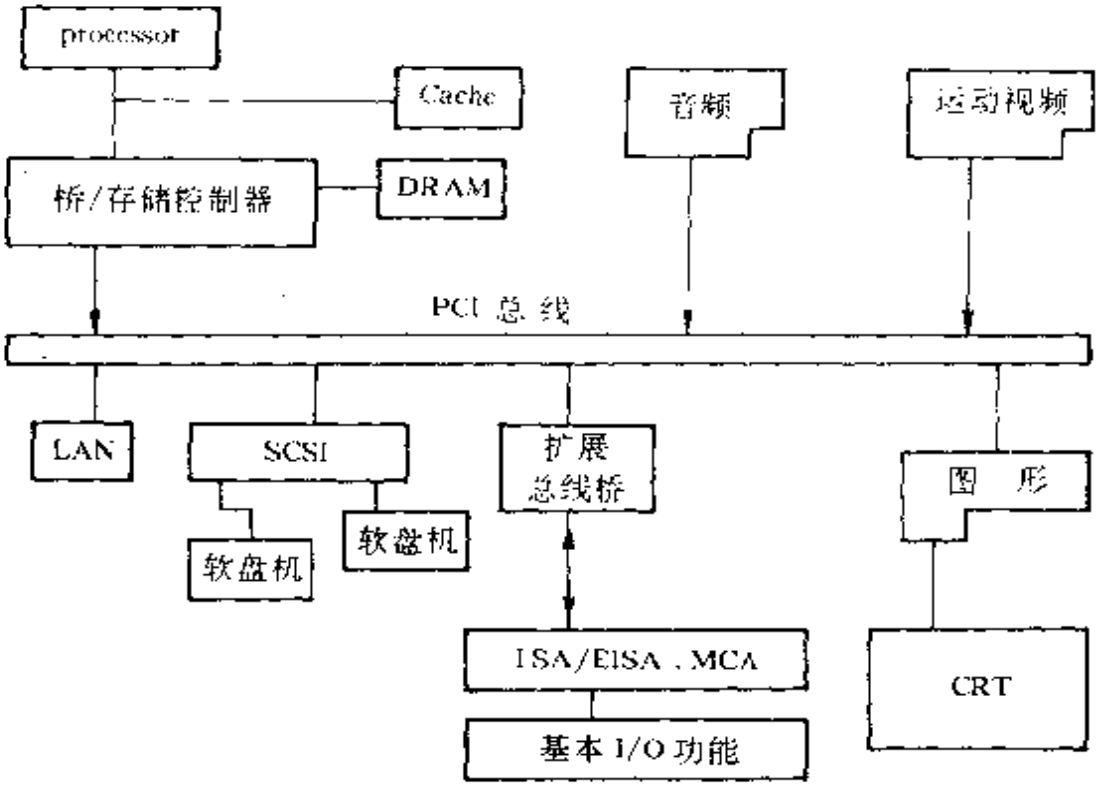


图 1.1 PCI 系统结构图

出一条标准 I/O 扩展总线，如 ISA、EISA 或 MCA 总线，从而可继续使用现有的 I/O 设备，以增加 PCI 总线的兼容性和选择范围。一般地，典型的 PCI 局部总线系统中，最多支持三个插槽(连接器)，但这样的扩充能力并不一定是必要的。PCI 接插卡连接器属于微通道(MC)类型的连接器。同样的 PCI 扩充板连接器也可用在 ISA一、EISA一及 MCA 总线的系统中。

1.5 PCI 总线的应用

对于一个新型总线标准，只有提供高性能、低成本、应用广泛、生命周期长等优点，才能成为工业标准。不但要着眼当前系统应达到新的性能价格比，更重要的是要能适应将来的系统要求，能在多种平台和体系结构上应用。图 1.2 说明了 PCI 总线的

多种应用范围。

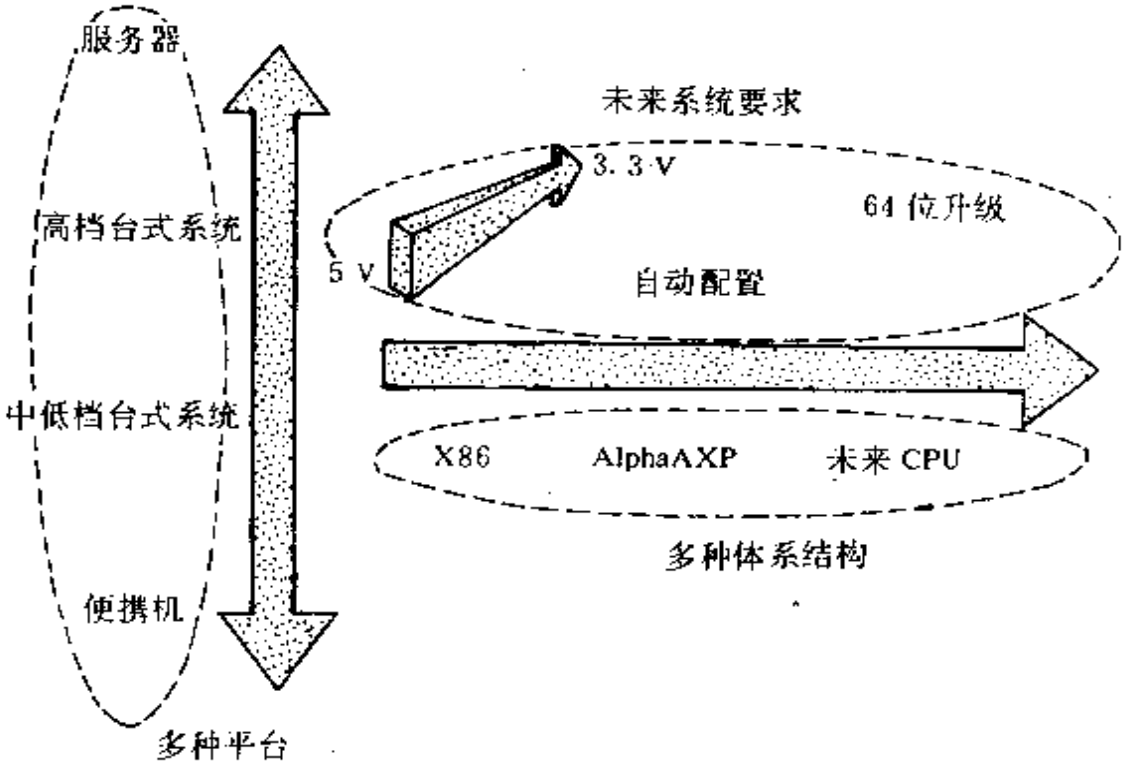


图 1.2 PCI 总线的应用范围

PCI 局部总线不仅可应用到低档至高档的台式系统上，而且也可应用在便携机乃至服务器的范围中。对于便携机应用要求 3.3 V 电源的同时，台式机应用也迫切要求由 5 V 电源改为 3 V 电源，这必须在标准中予以考虑。因此，PCI 局部总线规范中明确指定了两种电源电压，并说明了相应的转变途径。即为此定义了两种插接卡连接器：一种是 5 V 信号环境，一种是 3.3 V 信号环境；同时为这两种信号环境规定了三种插接卡电气类型，分别是 5 V 卡、3.3 V 卡和通用卡，其中通用卡是实现 5 V 到 3.3 V 过渡时使用的。

PCI 总线元件和插件接口与处理机是相互独立的，这样有助于将其应用到新型处理机上去，并适合于多种处理机体系结构的要求。同时，可使 PCI 局部总线根据 I/O 功能的需求而优化，总

线的操作与处理机/存储器子系统并行工作，以及适应图形、运动图像、LAN、SCSI、FDDI 和硬盘驱动器等多种高性能外部设备。

为了适应诸如高清晰度电视(HDTV)和三维显示等视频和多媒体显示的发展，以及高带宽 I/O 对局部总线带宽的进一步要求，PCI 局部总线定义了可对 32 位数据/地址总线进行 64 位扩展，并提供了 32 位及 64 位 PCI 局部总线设备的向前和向后的兼容性。

PCI 的自动配置功能使其应用更为方便，由于该总线标准为其元件及插件分配了相应的配置寄存器，对于一个系统只要有嵌入的自动配置软件，就可以在加电时自动配置 PCI 总线上的设备，为用户提供了很大的方便。

第二章　PCI 总线信号定义

在一个 PCI 应用系统中，如果某设备取得了总线控制权，就称其为“主设备”；而被主设备选中以进行通信的设备称为“从设备”或“目标节点”。对于相应的接口信号线，通常分为必备的和可选的两大类。如果只作为目标的设备，至少需要 47 条，若作为主设备则需要 49 条。利用这些信号线便可处理数据、地址，实现接口控制、仲裁及系统功能。下面对主设备与目标设备综合考虑，并按功能分组将这些信号表示于图 2.1 中。

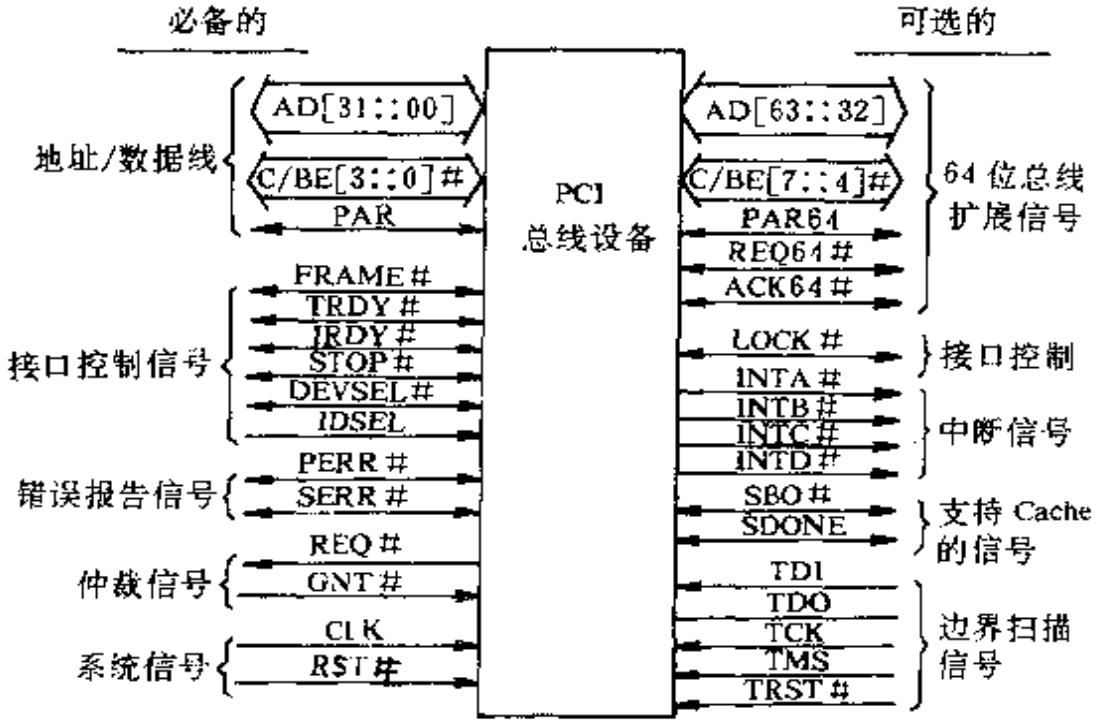


图 2.1　PCI 局部总线信号

图中 # 号表示低电平有效，否则为高电平有效。

2.1 信号类型说明

为了后续各节叙述方便，在此将 PCI 信号的所有类型规定综述如下：

IN　输入，是一标准的只作输入的信号。

OUT　输出，是一标准的图腾柱式输出驱动信号。

T/S　表示一双向的三态输入/输出信号。

S/T/S　表示一持续的并且低电平有效的三态信号。在某一时刻当且仅当只能属于一个主设备并被其驱动。这种信号从有效变为浮空(高阻状态)之前必须保证使其具有至少一个时钟周期的高电平状态。另一主设备要想驱动它，至少要等到该信号的原有驱动者将其释放(变为三态)一个时钟周期之后才能开始。同时，如果此类信号处于持续的非驱动状态时，在有新的主设备驱动它之前应采取上拉措施，并且该措施必须由中央资源提供。

OD　表示漏极开路，以线或形式允许多个设备共同驱动和分享。

2.2 PCI 总线信号定义

PCI 局部总线的信号线共有 100 根，下面按功能分组进行说明。

2.2.1 系统信号定义

CLK IN：系统时钟信号，对于所有的 PCI 设备都是输入信号。其频率最高可达 33 MHz，最低频率一般为 0 Hz(DC)，这一

频率也称为 PCI 的工作频率。对于 PCI 的其它信号，除 RST#、IRQB#、IRQC#、IRQD#之外，其余信号都在 CLK 的上升沿有效(或采样)。

RST# IN：复位信号。用来使 PCI 专用的特性寄存器和定序器相关的信号恢复规定的初始状态。至于 PCI 定序器之外的设备复位后如何变化，不属于本说明的范围。但必要的 PCI 配置寄存器，其复位状态是明确规定的。每当复位时，PCI 的全部输出信号一般都应驱动到第三态。SERR# 信号为高阻状态，SBD# 和 SDONE 可驱动到低电平(如果未提供三态输出)。REQ# 和 GNT# 必须同时驱动到第三态，不能在复位期间为高或为低。为防止 AD、C/BE# 及 PAR 在复位期间浮动，可由中心设备将它们驱动到逻辑低，但不能驱动为高电平。RST# 和 CLK 可以不同步，但要保证其撤消边沿没有反弹。当设备请求引导系统时，将响应复位，复位后响应系统引导。

2.2.2 地址和数据信号

AD[31：：00] T/S：它们是地址、数据多路复用的输入/输出信号。在 FRAME# 有效时，是地址期；在 IRDY# 和 TRDY# 同时有效时，是数据期。一个 PCI 总线的传输中包含了一个地址信号期和接着的一个(或无限个)数据期。PCI 总线支持突发方式的读写功能。

地址期为一个时钟周期，该周期中 AD[31：：00]线上含有一物理地址(32 位)。对于 I/O 操作，它是一个字节地址；若是存储器操作和配置操作，则是双字地址。

在数据期，AD[07：：00]为最低字节，AD[31：：24]为最高字节。当 IRDY# 有效时表示写数据稳定有效，而 TRDY# 有效时表示读数据稳定有效。

C/BE[3：：0]# T/S：它们是总线命令和字节使能多路复用

信号线。在地址期中，这四条线上传输的是总线命令；在数据期内，它们传输的是字节使能信号，用来表示在整个数据期中，AD[31：：00]上哪些字节为有效数据。

2.2.3 接口控制信号

FRAME# S/T/S：帧周期信号。由当前主设备驱动，表示一次访问的开始和持续时间。FRAME#的有效预示着总线传输的开始；在其存在期间，意味着数据传输继续进行；FRAME#失效后，是传输的最后一个数据期。

IRDY# S/T/S：主设备准备好信号。该信号的有效表明发起本次传输的设备能够完成一个数据期。它要与 TRDY#配合使用，二者同时有效，数据方能完整传输，否则即为等待周期。在读周期，该信号有效时，表示数据变量已在 AD[31：：00]中；在写周期，该信号有效时，表示从设备已做好接收数据的准备。

TRDY# S/T/S：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工作，也就是说，可以进行相应的数据传输。同样，该信号要与 IRDY#配合使用，二者同时有效，数据才能完整传输。在写周期内该信号有效表示从设备已做好了接收数据的准备；在读周期内，该信号有效表示有效数据已提交到 AD[31：：00]中。同理，IRDY#和 TRDY#的任何一个无效时都为等待周期。

STOP# S/T/S：停止数据传送信号。当它有效时，表示从设备要求主设备终止当前的数据传送。很显然，该信号应由从设备发出。

LOCK# S/TS：锁定信号。当该信号有效时，表示驱动它的设备所进行的操作可能需要多个传输才能完成。也就是说，对此设备的操作是排它性的。而此时，未被锁定的设备，对它的非互斥访问仍然可以进行。LOCK#信号的控制是由 PCI 总线上发起

数据传输的设备，根据它自己的约定并结合 GNT＃信号来完成的。即使有几个不同的设备在使用总线，但对 LOCK＃信号的控制权只属于一个主设备。如果某一设备具有可执行存储器，那么它也必须能实现锁定，以便实现对该存储器的完全独占性访问。对于支持锁定的目标设备，必须能提供一个互斥访问块，且该块不能小于 16 个字节。由于主桥后面是系统存储器，所以也应能实现锁定。

IDSEL IN：初始化设备选择信号。在参数配置读写传输期间，用作片选信号。

DEVSEL＃ S/T/S：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。换言之，它的有效说明总线上某处的某一设备已被选中。

2.2.4 仲裁信号

REQ＃ T/S：总线占用请求信号。该信号一旦有效即表明驱动它的设备要求使用总线。它是一个点到点的信号线，任何主设备都有其 REQ＃信号。

GNT＃ T/S：总线占用允许信号。用来向申请占用总线的设备表示，其请求已获得批准。这也是一个点到点的信号线，任何主设备都应有自己的 GNT＃信号。

2.2.5 错误报告信号

为使数据传输可靠、完整，PCI 局部总线标准要求，所有挂于其上的设备都应具有错误报告线。

PERR＃ S/T/S：数据奇偶校验错误报告信号。但该信号不报告特殊周期中的数据奇偶错。一个设备只有在响应设备选择信号（DEVSEL＃）和完成数据期之后，才能报告一个 PERR＃。对于每个数据接收设备，如果发现数据有错误，就应在数据收到后的两

个时钟周期内将 PERR＃激活。该信号的持续时间与数据期的多少有关，如果是一个数据期，则最小持续时间为一个时钟周期；若是一连串的数据期并且每个数据期都有错，那么 PERR＃的持续时间将多于一个时钟周期。由于该信号是持续的三态信号，因此，该信号在释放前必须先驱动为高电平。另外，对于数据奇偶错的报告既不能丢失也不能推迟。

SERR＃ O/D：系统错误报告信号。该信号的作用是报告地址奇偶错、特殊命令序列中的数据奇偶错，以及其它可能引起灾难性后果的系统错误。

如果设备不希望产生非屏蔽中断，就应采用其它机制来实现 SERR＃的报告。由于 SERR＃是一个漏极开路信号，因此，报告此类错误的设备只需将该信号驱动一个 PCI 周期即可。SERR＃信号的发出和时钟同步，因而满足总线上所有其它信号的建立时间和保持时间的要求。要使该信号复位，需要一个微弱的上拉作用，但这应由系统设计来提供，而不是靠报错的设备或中央资源。一般这种上拉复位需要 2～3 个时钟周期才能完成。

2.2.6 中断信号

中断在 PCI 总线中是可选项，不一定必须具有。并且中断信号属电平敏感性，低电平有效，使用漏极开路方式驱动。同时，此类信号的建立和撤消与时钟不同步。对于单功能设备，只有一条中断线，而多功能设备最多可有四条中断线。在前一种情况下，只能使用 INTA＃，其它三条中断线没有意义。所谓的多功能设备是指：将几个相互独立的功能集中在一个设备中。

PCI 局部总线中共有四条中断线，分别是：INTA＃、INTB＃、INTC＃和 INTD＃，均为 O/D(漏极开路)。其作用是：用以请求一个中断。后三个只能用于多功能设备。

一个多功能设备上的任何功能都可以连接到四条中断线的任

意一条。也就是说，各功能与中断线之间的连接是任意的，没有附加限制，二者的最终对应关系是由中断引脚寄存器来定义的。这显然提供了很大的灵活性。如果一个设备要实现一个中断，就定义为 INTA＃；要实现二个中断，就定义为 INTA＃和 INTB＃，依次类推。对于多功能设备，可以多个功能共用同一条中断线，或者各自占一条中断线，或者是两种情况的组合。但是，对于单功能设备，绝对不能在多于一条中断线上发中断请求。

系统供应商在对 PCI 连接器的各个中断信号和中断控制器进行连接时，其方法是随意的，可以是线或方式、程控电子开关方式，或者是二者的组合，这就是说，设备驱动程序对于中断共享事先无法作出任何假定。

2.2.7 其它可选信号

1. 高速缓存支持信号

为了使具有可缓存功能的 PCI 存储器能够和写穿式(Write-through)或回写式(Write-back)的 Cache 相配合工作，可缓存的 PCI 存储器应该能实现两条高速缓存(Cache)支持信号作为输入。如果可缓存的存储器位于 PCI 总线上，那么，连接回写式 Cache 和 PCI 的桥要能够将这对信号作为输出，而连接写穿式 Cache 的桥只需要实现一个信号。上述的两个信号定义如下：

SBO＃ IN/OUT：试探返回信号。当该信号有效时，表示命中了一个修改过的行。当该信号无效，而 SDONE 信号有效时，表示有一个“干净”的试探结果。

SDONE IN/OUT：监听完成信号。用来表示当前监听的状态。该信号无效时，表明监听仍在进行，否则，表明监听已经完成。

2. 64 位总线扩展信号

必须注意的是，如果要进行 64 位扩展，以下信号都要使用。

AD[63：：32] T/S：扩展的 32 位地址和数据多路复用线。

在地址期(如果使用了DAC命令且REQ64#有效时)这32条线上含有64位地址的高32位，否则，它们是保留的；在数据期，当REQ64#和ACK64#同时有效时，这32条线上含有高32位数据。

C/BE[7：：4]# T/S：总线命令和字节使能多路复用信号线。在数据期，若REQ64#和ACK64#同时有效时，该四条线上传输的是表示数据线上哪些字节是有意义的字节使能信号。如C/BE[4]#对应第四个字节，C/BE[5]#对应第五个字节。在地址期里，如果使用了DAC命令且REQ64#信号有效，则表明C/BE[7：：4]#上传输的是总线命令，否则这些位是保留的且不确定。

REQ64# S/T/S：64位传输请求。该信号由当前主设备驱动，并表示本设备要求采用64位通路传输数据。它与FRAME#有相同的时序。

ACK64# S/T/S：64位传输认可。表明从设备将用64位传输。此信号由从设备驱动，并且和DEVSEL#具有相同的时序。

PAR64 T/S：奇偶双字节校验。是AD[63：：32]和C/BE[7：：4]的校验位。当REQ64#有效且C/BE[3：：0]#上是DAC命令时，PAR64将在初始地址期之后一个时钟周期有效，并在DAC命令的第二个地址期过后的一个时钟处失效。当REQ64#和ACK64#同时有效时，PAR64在各数据期内稳定有效，并且在IRDY#或TRDY#发出后的一个时钟处失效。

PAR64信号一旦有效，将保持到数据期完成之后的一个时钟周期处。该信号与AD[63：：32]的时序相同，但拖后一个时钟周期，对于主设备是为了地址和写数据而发PAR64，从设备是为了读数据而发PAR64。

第三章　PCI 总线的操作

3.1　总线命令

总线命令的作用是用来规定主、从设备之间的传输类型，它出现于地址期的 C/BE[3：：0]#线上。这里的主设备是指通过仲裁而获得总线控制权的设备；从设备是指在 C/BE[3：：0]#上出现命令的同时，被 AD[31：：00]线上的地址所选中的设备。

3.1.1　总线命令编码

表 3.1 给出了总线命令的编码及类型说明。其中，命令编码中的“1”表示高电平，“0”表示低电平。

下面对各条命令作较为详细地说明。

表 3.1　总线命令表

C/BE[3：：0]#	命令类型说明
0　0　0　0	中断应答(中断识别)
0　0　0　1	特殊周期
0　0　1　0	I/O 读(从 I/O 口地址中读数据)
0　0　1　1	I/O 写(向 I/O 地址空间写数据)
0　1　0　0	保留
0　1　0　1	保留
0　1　1　0	存储器读(从内存空间映象中读数)
0　1　1　1	存储器写(向内存空间映象写数据)

续表

C/BE[3：：0]#	命令类型说明
1 0 0 0	保留
1 0 0 1	保留
1 0 1 0	配置读
1 0 1 1	配置写
1 1 0 0	存储器多行读
1 1 0 1	双地址周期
1 1 1 0	存储器一行读
1 1 1 1	存储器写并无效

1. 中断应答命令

中断应答命令是一个读命令，并且对中断控制器的寻址采用稳含方式，也就是说，该地址是逻辑地址而不明显地出现于地址期，回送的中断矢量的长度由字节使能信号来表示。如图 3.1 所示。图中互相指着对方尾部的两个箭头表示过渡周期，该周期的设定，是为了避免当一个设备停止驱动该信号而另一个设备开始驱动时可能发生的冲突。

图 3.1 说明了一个 X86 的中断应答在 PCI 总线上进行的情况。这仅作为一个例子，其中的字节使能是单一字节。一般情况下，字节使能信号用来确定哪些字节是传输中要涉及到的。从图中可看出，在地址期中，尽管 AD[31：：00]中不含有效地址，但必须将它们驱动到稳定的状态。在中断应答过程中，PAR 信号是有效的，而且被用来进行奇偶校验。对中断应答命令的响应只能是一个设备，并且响应的设备必须发出相应的 DEVSEL＃信号，同时在 TRDY＃信号有效时必须返回中断向量。另外，中断应答

周期与其它周期一样可插入等待周期。

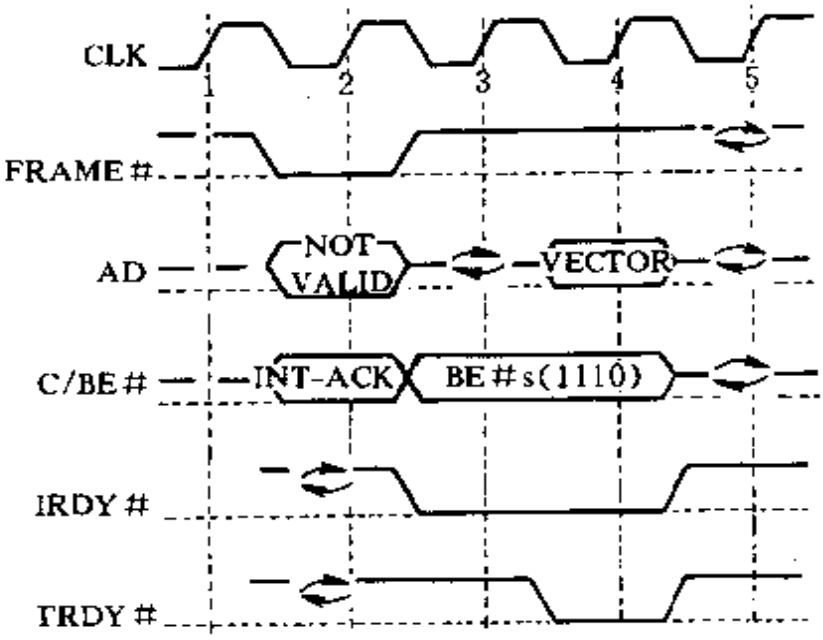


图 3.1 中断应答

由于 PCI 总线中的中断应答采用的是单个周期，这与传统的 8259 双周期应答是不同的，因此在桥路应把处理机的双周期格式变换成 PCI 上的单次格式，这只要将处理机上送出的第一个中断应答请求丢掉即可。

2. 特殊周期命令

该命令的作用是为 PCI 提供一个简单的信息广播机制。它不但能报告处理机的状态，而且可用来作为 PCI 设备间逻辑的侧面连接信号(当这些信号不要求精确的时间关系或物理信号同步时)。

特殊周期命令不包含目标地址，而是以广播的形式发给所有设备。每个接收设备必须自我确定广播的消息是否适合于它。在特殊周期命令期，不允许 PCI 设备发出 DEVSEL# 信号。也就是说，此类对话不需要目标设备的应答，同时负译码的桥也不能将特殊周期命令传到它的下级总级上，更不能跨桥传播特殊周期

命令。

一个特殊周期命令可以包含可选的由消息决定的数据，它不是由 PCI 序列器来解释，而是根据需要传给与 PCI 序列器相连的应用硬件。在大多数情况下，显式的寻址信息是由 PCI 总线上的三种物理地址空间之一来处理，而不是用特殊周期命令。

特殊周期命令也像其它总线命令一样，具有一个地址期和一个数据期。地址期开始于 FRAME # 信号的建立，结束于 FRAME # 和 IRDY # 信号的同时撤消。该命令与其它总线命令的唯一区别是没有目标设备响应信号 DEVSEL #。

该命令在地址期除了命令字外不含其它有效信息，也没有地址信息，但是，AD[31：：00]要驱动到稳定的电平，并产生奇偶校验位。在数据期，AD[31：：00]上含有消息类型和一个可选的数据字段，消息用 AD[15：：00]这低 16 位编码表示，可选的数据字段用 AD[31：：16]这高 16 位编码表示，但高 16 位并非所有消息都需要。在特殊周期中主设备可插入等待周期，而目标设备不允许插入等待周期，原因是特殊周期内无固定的从设备。消息和数据仅在 IRDY # 有效后的第一个时钟内有效，此后的数据期中所含信息及时序关系要由消息来决定。

在地址期中，当 C/BE[3：：0]# = 0001 时，表示是特殊周期命令，而 AD[31：：00]被驱动为随机值且忽略不计。在数据期，C/BE[3：：0]# 要有效，而 AD[31：：00]各位的情况如下：

AD[15：：0]含有消息编码，当 AD[15：：0] = 0000H 时为 SHUTDOWN；AD[15：：0] = 0001H 时为 HALT；AD[15：：0] = 0002H 时为 X86 有关消息；AD[15：：0] = 0003H～FFFH 保留；而 AD[31：：16]含有消息所决定的可选数据字段。

PCI 总线序列器启动该命令的方式与其它命令相同，但终止该命令要由主设备完成。当从设备得到主设备的终止信息时，就知道一次访问已完成。在这种情况下，配置空间的状态寄存器的

"接收主设备终止位"不能置 1。特殊周期命令的完成最快也要五个时钟周期，同时还要附加一个时钟周期作为下一个访问开始之前的过渡周期。

3. I/O 读命令

该命令用来从一个映射到 I/O 地址空间的设备中读取数据。AD[31：：00]上提供一个字节地址，全部 32 位必须完全译码；而字节使能信号表示传送数据的多少，必须与字节地址一致。

4. 保留命令

该类编码是为将来的用途而保留的。PCI 的任何设备都不能将它们挪作它用，任何设备也不允许对保留命令编码作出反应。如果接口中使用了一条保留命令，通常要由主设备终止操作来结束本次访问。

5. I/O 写命令

该命令用来向一个映射到 I/O 地址空间的设备写入数据。全部 32 位地址必须参加译码，字节使能信号表示数据长度，且必须和字节地址一致。

6. 存储器读命令

该命令用来从一个映射到存储器地址空间的设备读取数据。如果能保证无副作用产生时，从设备可以为该命令进行预先读取。另外，目标设备也要保证在本次 PCI 传输之后保存于临时缓冲器中的数据的一致性(包括数据次序)。这个缓冲器在任何同步事件(如更新 I/O 状态寄存器或存储器标志)通过此访问通路之前必须被置为无效。

7. 存储器写命令

该命令用来向一个映射到存储器空间的设备写入数据。当从设备发出"准备好"信号后，它已经准备对所涉及的数据的一致性(包括次序)负责。因此，对于该命令的实现可采用完全同步的方式，或采用其它方法。但应保证在任何同步事件通过该访问路径

之前使数据缓冲器被冲洗。也就是说，主设备在使用了该命令之后可以立即创造一个同步事件。

8. 配置读命令

该命令用来从每个设备的配置空间读取数据。如果一个设备的 IDSEL 引脚有效，且 AD[1：：0]=00 时，那么该设备即被选定为配置读命令的目标。在一个配置命令的地址期内，AD[7：：2]用于从每个设备的配置空间中的 64 个双字寄存器中选出一个。AD[31：：11]无意义，AD[10：：8]表示一个多功能设备的哪个功能设备被选中。

9. 配置写命令

该命令用来向每个设备的配置空间写入数据。一个设备被选中的条件是：它的 IDSEL 信号有效并且 AD[1：：0]=00。其余和配置读命令相同。

10. 存储器多行读命令

该命令的作用是试图在主设备断开连接之前预读取多行 Cache 数据。存储器控制器应保证，只要 FRAME# 有效，就连续不断地以流水方式发存储器请求。该命令预定用于大块连续数据的传输，如果有一个对软件透明的缓冲器来暂存数据的话，通过顺序地预读一个附加的 Cache 行，可使存储器系统得到某些性能上的改善。

11. 双地址周期(DAC)命令

该命令用来给支持 64 位寻址的设备发送 64 位地址。发送过程需要两个时钟周期。对于只有 32 位寻址能力的设备，不得以任何方式对该命令作出反应，只能把它当作保留命令。

12. 存储器一行读命令

该命令与存储器读命令基本相同，不同之处在于它还表示主设备试图完成多于两个 32 位的 PCI 数据期。此命令也预定用于大块连续数据的传输。这样存储器系统的性能会得到一些改进，

道理是响应一次请求不仅仅完成一个存储器读周期，而是一直读到一个 Cache 的行边界。和存储器读命令的情形一样，预取缓冲器必须在任何同步事件通过该访问通路之前变为无效。

13. 存储器写并无效命令

该命令在语义上与存储器写命令相同，不同点是它要保证最小的传输量是一个高速缓存(Cache)的行，也就是说，主设备要在一次 PCI 传输中将寻址的 Cache 行的每个字节都写入。如果要传输下一行的所有字节，允许主设备跨边界写入。该命令同时要求主设备的配置寄存器指出 Cache 行的尺寸。存储器写并无效命令也是保证 Cache 一致性的措施。

3.1.2 命令使用规则

配置读命令和配置写命令要求所有的 PCI 设备都以目标设备的形式给予响应，其它所有命令都为可选项。在 PCI 总线上执行 I/O 读写命令时，应保证其执行顺序。PCI 上的设备应具有可在定位功能或寄存器，以便通过配置寄存器将它们映射到存储器空间。所有的主、从设备都可根据需要来实现选项命令。但是，如果某个设备实现了基本的存储器命令，那么，它就必须支持所有的存储器命令，否则，就必须利用别名将这些为优化性能而设的命令(存储器一行读、存储器多行读及存储器写并无效命令)转变为基本的存储器命令。例如，一个从设备可以不实现存储器一行读命令，但是它必须能接受该命令的请求，并按存储器读命令来处理。同理，一个从设备可以不实现存储写并无效命令，但它必须能接受该命令的请求，并按存储器写命令来处理。

对于系统存储器的块数据读写，建议在主设备支持的情况下尽量采用存储器写并无效命令和存储器行读命令。如果主设备确实不能支持上述优化性能的命令，可采用存储器读写命令。

对于使用存储器读命令的主设备，所有命令都可进行任何长

度的访问，但最好遵循下述用法原则。尽管存储器写并无效命令的实现，要求具有 Cache 行长度寄存器，但存储器读命令最好也使用它。下面是在有和没有行长度寄存器情况下的用法建议：

1. 具有 Cache 行长度寄存器时

存储器读命令用来读半行或更少；存储器一行读用来读取半行以上到三行；存储器多行读用来读取三行以上。

2. 无 Cache 行长度寄存器时

存储器读命令用于 1～2 次的数据突发传输；存储器一行读用于 3～12 次的数据突发传输；存储器多行读命令用于 13 次或更多数据的突发传输。

3.2 PCI 总线协议

PCI 上的基本总线传输机制是突发成组传输。一个突发分组由一个地址期和一个(多个)数据期组成。PCI 支持存储器空间和 I/O 空间的突发传输。这里的突发传输是指主桥(位于主处理机和 PCI 总线之间)可以将多个存储器写访问在不产生副作用的前提下合并为一次传输。一个设备通过将基址寄存器的预取位置 1，来表示允许预读数据和合并写数据。一个桥可利用初始化时配置软件所提供的地址范围，来区分哪些地址空间可以合并，哪些不能合并。当遇到要写的后续数据不可预取或者一个对任何范围的读操作时，在缓冲器的数据合并操作必须停止并将以前的合并结果清洗。但其后的写操作，如果是在预取范围内，便可与更后面的写操作合并，但无论如何不能与前面合并过的数据合并。

只要处理机发出的一系列写数据(双字)所隐含的地址顺序相同，主桥路总是可以将它们组合成突发数据。例如，若处理机的写顺序是 DWORD0、DWORD2、DWORD3，那么主桥路即可将它们组合成一次突发。PCI 的突发顺序可以是 DWORD0、

DWORD1、DWORD2 到 DWORD3 结束，中间可插入未访问的 DWORD(双字)，只要对应该双字的字节使能信号不发即可。合并的原则是只要后面的 DWORD 地址比前面的高就行。当读操作不会在被寻址的从设备上引起副作用时，桥路便可将单个的处理机读请求转变为一次突发读(前述的预取)。

对于 I/O 空间的访问不能合并，因此它们一般只有一个数据期。目前，是否有处理机或总线主设备可以发出指向 I/O 空间的突发传输，还无法得知，但不能排除将来有这种设备的可能性。不能处理多个 I/O 数据期的 PCI 设备必须在第一个数据期之后断开访问，为此，桥路绝对不能把顺序的 I/O 访问合并成一个突发的访问或单个的 PCI 访问。也就是说，对于所有的 I/O 访问，在处理机产生它们的同时，该访问也必须出现在 PCI 总线上。如果一个从设备被 I/O 访问选中，在字节使能信号所代表的传输长度大于该设备能支持的长度时，从设备要用目标终止方式结束本次访问。

在 PCI 总线中，除了 RST＃、INTA＃～INTD＃之外，其它所有信号都在时钟上升沿被采样，每个信号都有相对于时钟前沿的建立和保持时间，在此期间不允许有信号跳动，该时间一过，信号的变化就无关紧要了。这种建立和保持时间对于不同的信号，其情形是不同的。对于 AD[31：：00]、AD[63：：32]、PAR、PAR64 和 IDSEL 来说，只有在一定的时钟边沿上才有上述时间的要求；对于 LOCK＃、IRDY＃、TRDY＃、FRAME＃、DEVSEL＃、STOP＃、REQ＃、GNT＃、REQ64＃、ACK64＃、SBO＃、SDONE＃、SERR＃和 PERR＃这些信号，在每个时钟前沿都有建立和保持时间；对于 C/BE[3：：0]＃、C/BE[7：：4]＃这两组信号，在传输总线命令时，要在 FRAME＃第一次建立时对应的时钟边沿上遵守建立和保持时间的关系；若传输字节使能信号时，要在完成一个地址期或数据期之后的每一个时钟边沿保证相应的建立和保持

时间。

3.2.1 PCI 总线的传输控制

PCI 总线上所有的数据传输基本上都是由以下三条信号线控制的：

FRAME＃　由主设备驱动，指明一个数据传输的起始和结束。

IRDY＃　由主设备驱动，允许插入等待周期。

TRDY＃　由从设备驱动，允许插入等待周期。

当数据有效时，数据资源需要无条件设置 XRDY＃信号(写操作为 IRDY＃，读操作为 TRDY＃)。接收方可以在适当的时间发出它的 XRDY＃信号。FRAME＃信号有效后的第一个时钟前沿是地址期的开始，此时传送地址信息和总线命令。下一个时钟前沿开始一个(多个)数据期，每逢 IRDY＃和 TRDY＃同时有效时，所对应的时钟前沿就使数据在主、从设备之间传送，在此期间，可由主设备或从设备分别利用 IRDY＃和 TRDY＃的无效而插入等待周期。

一旦主设备设置了 IRDY＃信号，将不能改变 IRDY＃和 FRAME＃，直到当前的数据期完成为止。而一个从设备一旦设置了 TRDY＃信号或 STOP＃信号，就不能改变 DEVSEL＃、TRDY＃或 STOP＃，直到当前的数据期完成。也就是说，不管是主设备还是从设备，只要承诺了的数据传输，就必须进行到底。

当到最后一次数据传输时(有时紧接地址期之后)主设备应撤消 FRAME＃信号，而建立 IRDY＃信号，表明主设备已做好了最后一次数据传输的准备，待到从设备发出 TRDY＃信号后，就说明最后一次数据传输已完成，FRAME＃和 IRDY＃信号均撤消，接口回到了空闲状态。总之，PCI 总线的传输一般遵循如下管理规则：

(1) FRAME#和 IRDY#定义了总线的忙/闲状态。当其中一个有效时，总线是忙的；两个都无效时，总线处于空闲状态。

(2) 一旦 FRAME#信号被置为无效，在同一传输期间不能重新设置。

(3) 除非设置 IRDY#信号，一般情况下不能设置 FRAME#信号无效。

(4) 一旦主设备设置了 IRDY#信号，直到当前数据期结束为止，主设备不能改变 IRDY#信号和 FRAME#信号的状态。

3.2.2 PCI 的编址

PCI 总线定义了三个物理地址空间：内存地址空间、I/O 地址空间和配置地址空间。前两个是通常都有的，第三个用以支持 PCI 的硬件配置，将在后续章节中进一步说明。

PCI 总线的编址是分布式的，每个设备都有自己的地址译码，从而省去了中央译码逻辑。PCI 支持正向和负向两种风格的地址译码，所谓正向译码就是每个设备都监视地址总线上的访问地址是否落在它的地址范围，因此速度较快；而负向译码是指该设备要接受未被其它设备在正向译码中接受的所有访问，因此，此种译码方式只能由总线上的一个设备来实现，由于它要等到总线上其它所有设备都拒绝之后才能行动，所以速度较慢。然而，负向译码对于像标准扩展总线这类设备却是很有用的，这是因为这类设备必须响应一个很零散的地址空间。但无论是正向译码的设备还是负向译码的设备都不对保留的总线命令发出 DEVSEL#响应信号。

1. I/O 地址空间

在 I/O 地址空间，全部 32 位 AD 线都被用来提供一个完整的地址编码(字节地址)，这使得要求地址精确到字节水平的设备不需要多等一个周期就可完成地址译码(产生 DEVSEL#信号)、

也使负的地址译码节省了一个时钟周期。

在I/O访问中，AD[1：：0]两位很重要．它一方面用来产生DEVSEL#信号，更值得注意的是它也表示传输涉及的最低有效字节，并且要与C/BE[3：：0]#相配合。例如，当C/BE0#有效时，那么AD[1：：0]必须为"00"；如果C/BE3#有效时，AD[1：：0]就应当为"11"。在具体访问中，每当一个从设备被地址译码选中后，便要检查字节使能信号是否与AD[1：：0]相符，如果二者矛盾，则整个访问就无法完成，此时，从设备不传送任何数据，而是以一个"目标终止"操作来结束访问。表3.2给出了AD[1：：0]和C/BE[3：：0]#的对应关系。

表3.2　AD[1：：0]与C/BE[3：：0]#对应关系表

AD1	AD0	C/BE3#	C/BE2#	C/BE1#	C/BE0#
0	0	X	X	X	0
0	1	X	X	0	1
1	0	X	0	1	1
1	1	0	1	1	1

其中：1表示高电平，0表示低电平，X表示0或1。对于任何不同于表3.2的组合状态都是非法的。

2. 内存地址空间

在存储器访问中，所有的目标设备都要检查AD[1：：0]，要么提供所要求的突发传输顺序，或者执行一目标设备断开操作。对于所有支持突发传输的设备都应能实现线性突发传输顺序。而Cache的行切换不一定必须实现。在存储器地址空间，要用AD[31：：02]译码得到一个双字地址的访问。在线性增长方式下，每个数据周期过后地址按一个DWORD(4个字)增长，直到对话结束。在存储器访问期间，AD[1：：0]的含义如下：

当AD[1：：0]＝00时，突发传输顺序为线性增长方式；AD

[1：：0]=01 时，为 Cache 行切换方式；AD[1：：0]=1X 时，为保留。

3. 配置地址空间

在配置的地址空间中，要用 AD[7：：2] 将访问落实到一个 DWORD 地址。当一个设备收到配置命令时，若 IDSEL 信号成立且 AD[1：：0]=00，则该设备即被选为访问的目标。否则就不参与当前的对话。如果译码出的命令，符合某桥路的编号，且 AD[1：：0]=01，则说明配置访问是对着该桥后面的设备。

3.2.3 字节校正

字节使能信号仅被用来指出哪些字节带了有意义的数据，在每个数据期内，可以自由改变字节使能，使之对传输数据的实际含义和有效部分进行界定，这一功能被称作“字节校正”。

由于 PCI 设备的地址译码需要 32 位地址/数据线的全部，所以在 PCI 总线上不能进行字节的交换。但是具有 64 位数据通路的主设备可以进行 DWORD(双字)的交换，这就是说，字节总是在按字节地址所规定的自然字节位置上出现。主设备可以在每个新数据期改变字节使能信号，但一定要使其在每个数据期开始的时钟前沿变为有效，且在整个数据期中保持不变。一个主设备若要在读操作中改变字节使能信号，那么在写操作中也应采用同样的时序关系。如果目标设备在读操作中特别要求字节使能信号配合的话，则在相应地数据期必须等到字节使能信号有效，方可完成传输，否则该设备必须回送全部字节。

如果当前要读取的数据来自于可缓存的存储器，则无论字节使能信号是否有效，所有的字节都必须传送。这就要求主控一方能够保证使得目标设备回送全部字节。也就是说，假如可缓存性是由数据读取发起者判断而知，则该设备要能保证使 C/BE[3：：0]# 的各位均有效，以使被选中的设备送回全部字节；反

之，要是目标一方判断而知，那么目标方就必须回送全部字节，而不必考虑字节使能信号的状态。

对于一个不支持高速缓存但支持预取的从设备(目标设备)，只要不会引起数据破坏或状态改变，也可回送全部字节而不受字节使能信号的控制。

PCI 总线允许字节使能信号以相邻或不相邻的形式进行组合。对于目标设备，即使没有字节使能信号，也必须通过发出 TRDY＃使数据传输完成，倘若是读操作还要提供奇偶位。

3.2.4 总线的驱动与过渡

为了避免多个设备同时驱动一个信号到 PCI 总线上而产生竞争，在一个设备驱动到另一个设备驱动之间设置了一个过渡期，又称交换周期，在后续章节的时序图中，用互相指着对方尾部的双箭头符号来表示。此周期在不同的信号上所出现的时刻是不同的。对于 IRDY＃、TRDY＃、STOP＃、DEVSEL＃和 ACK64＃这些信号都利用地址期作为它们的交换周期；而 FRAME＃、C/BE[3：：0]＃、AD[31：：00]、C/BE[7：：4]＃、AD[64：：32]及 REQ64＃这些信号，则是利用数据传输之间的空闲期作为交换周期；LOCK＃信号的交换周期出现于当前的控制者释放它之后的一个时钟周期；PERR＃的交换周期开始于前一个数据期过后的第四个时钟周期，相当于比 AD 线的交换周期推迟三个时钟周期。

在每个地址期和数据期中，所有的 AD 线，都必须被驱动到稳定的状态(数据)，即便是在当前数据传输中未涉及到的字节所对应的 AD 线，也不例外。这样做的目的有三：① 为了奇偶计算；② 使传送中未被包含的字节所对应的缓冲区不致于处在临界状态；③ 便于快速而相对稳定地锁定数据。在实际应用中，如果对功耗要求较大时，为尽量减少由于总线上信号开关所造成的功

耗，对当前总线周期中不用的字节最后利用与前一周期相同的数据去驱动它们，否则，没有具体要求。但奇偶位的计算包括所有的字节在内，与字节使能信号的有效与否无关。

3.3 总线上的数据传输过程

本节所给时序图主要表示总线以 32 位方式执行有关操作时，相应信号之间的关系。在具体图示中，当一信号以虚线画出时，则表示没有设备驱动它；但若此虚线处在基准位置时，仍然可表示它具有一个稳定的值；当三态信号以虚线方式画在高、低状态之间时，说明它的值是不稳定的(例如，AD 线或 C/BE # 线)；当一条实线变成连续的短线时，表明它由原来的被驱动状态变成了现在的三态；当一实线在由低向高跳变后变为连续的短线时，则说明该信号先经预充电变为高电平，然后变成三态(释放)。上述关于信号状态画法的约定在以后的章节中同样适用。

3.3.1 总线上的读操作

图 3.2 表示了总线上一次读操作中有关信号的变化情况。

从图中可看出，一旦 FRAME # 信号有效，地址期就开始，并在时钟 2 的上升沿处稳定有效。在地址期内，AD[31 : : 00]上包含有有效地址，C/BE[3 : : 0]# 上含有一个有效的总线命令。数据期是从时钟 3 的上升沿处开始的，在此期间，AD[31 : : 0]线上传送的是数据，而 C/BE # 线上的信息却指出数据线上的哪些字节是有效的(即哪几个字节是当前要传输的)。要特别指出的是无论是读操作还是后面要讲的写操作，从数据期的开始一直到传输的完成，C/BE # 的输出缓冲器必须始终保持有效状态。

图中的 DEVSEL # 信号和 TRDY # 信号是由被地址期内所发地址选中的设备(从设备)提供的，但要保证 TRDY # 在

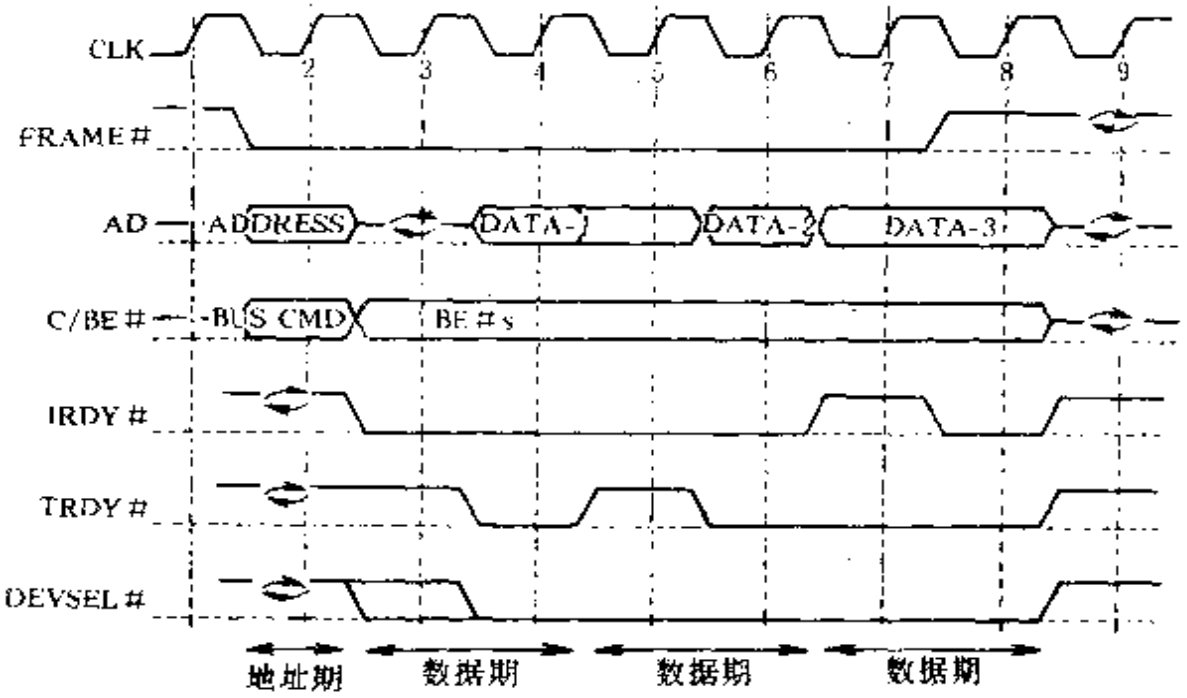


图 3.2　读操作时序

DEVSEL#之后出现。而 IRDY#信号是发起读操作的设备(主设备)根据总线的占用情况自动发出的。数据的真正传输是在 IRDY#和 TRDY#同时有效的时钟前沿进行的，这两个信号的其中之一无效时，就表示需插入等待周期，此时不进行数据传输。这说明，一个数据期可以包含一次数据传输和若干个等待周期。在图 3.2 中，时钟 4、6、8 处各进行了一次数据传输，而在时钟 3、5、7 处插入了等待周期。

在读操作中的地址期和数据期之间，AD 线上要有一个交换期，这需要由从设备利用 TRDY#强制实现(也就是 TRDY#的发出必须比地址的稳定有效晚一拍)。但在交换期过后并且有 DEVSEL#信号时，从设备必须驱动 AD 线。

在时钟 7 处尽管是最后一个数据期，但由于主设备因某种原因不能完成最后一次传输(具体表现是此时 IRDY#无效)，故 FRAME#不能撤消，只有在时钟 8 处，IRDY#变为有效后，

FRAME#信号才能撤消。

3.3.2 总线上的写操作

图 3.3 表示总线上一次写操作的时序关系。

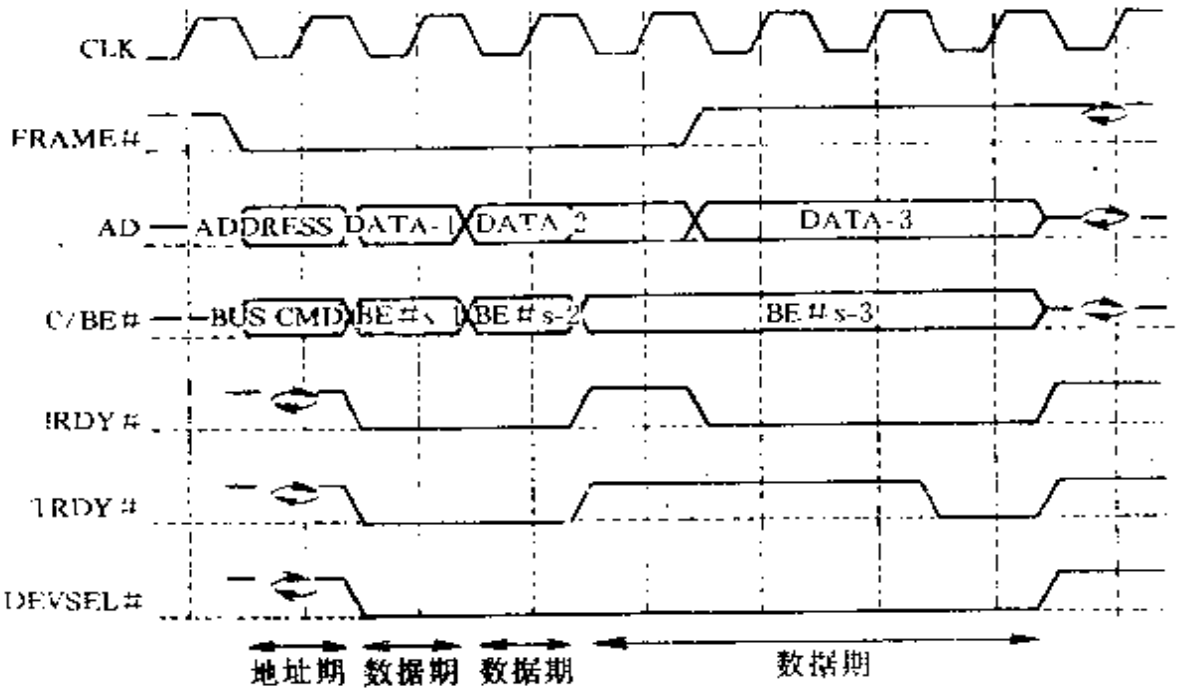


图 3.3 写操作时序

由图可知，总线上的写操作与读操作相类似，也是FRAME#信号的有效预示着地址周期的开始，且在时钟 2 的上升沿处达到稳定有效。整个数据期也与读操作基本相同，读者自行分析。只是在第三个数据期中由从设备连续插入了三个等待周期，时钟 5 处传输双方均插入了等待周期。

值得注意的是，当 FRAME# 撤消时必须要有 IRDY# 发出为前提，以表明是最后一个数据期。另外，从图 3.3 中可看出，主设备在时钟 5 处因撤消了 IRDY# 而插入等待周期，表明要写的数据将延迟发送，但此时，字节使能信号不受等待周期的影响，不得延迟发送。

要说写操作与读操作的不同点，那就是在写操作中，地址期与数据期之间没有交换周期，这是因为，在此类操作中，数据和地址是由同一个设备(主设备)发出的。

最后要强调的是：上述的读/写操作均是以多个数据期为例来说明的。如果是一个数据期时，FRAME#信号在没有等待周期的情况下，应在地址期(读操作应在交换周期)过后即撤消。若有等待周期时，读者可自行分析其时序关系。

3.3.3 传输的终止过程

无论是主设备，还是从设备，都可以提出终止一次 PCI 总线传输的要求，但要求不等于具体实施，也就是说双方均无权力单方面实施传输停止工作，需要相互配合，这一点读者可从下面的叙述中体会到。不过有一点是肯定的，那就是传输的最终停止控制要由主设备完成，这是因为传输的结束必须满足系统的要求并且应是有秩序的，这只有主设备才能做到。同时，所有传输的结束标志是 FRAME#信号和 IRDY#信号均已撤消而进入总线空闲状态(参看图 3.3 时钟 9 处)。

1. 由主设备提出的终止

主设备是通过撤消 FRAME#信号并建立 IRDY#信号来提出终止请求的。实际上，这样做的目的就是告诉从设备，现在已进入了最后的数据期，此后 IRDY#一直保持有效，直到出现 TRDY#信号，完成最后一个数据的传输。接着 IRDY#便撤消，从而达到完全终止的条件(FRAME#和 IRDY#同时无效)，结束传输，进入总线空闲状态。

读者可能要问，主设备一般在什么情况下会提出终止传输的要求呢？其原因有二：

(1) 这是一个最常见，也是一个最容易理解的原因，它就是主设备已做完要做的事。

(2) 当主设备的 GNT# 信号无效并且其内部的延时计数器已满，从而不得不终止传输，即所谓的超时。究其超时的原因，不是从设备产生的访问延迟，便是要做的操作太长。

下面通过例子来进一步说明上述两种主设备终止传输的具体情形。

如图 3.4 所示，在时钟 3 处，主设备撤消了 FRAME# 信号而建立了 IRDY#，说明它已得知目前的操作已完成，便以此方式提出终止，而此时 TRDY# 也正好有效，故最后一个数据被传输，当时钟 4 到来时，IRDY# 已撤消，因为此时传输已经完成，所以 IRDY# 也在此刻失效。但实际上在时钟 3 处，TRDY# 也可以无效，从而形成最终数据传输及终止的延迟。应注意的是此时应保证主设备发出的 IRDY# 的有效期维持到最终数据传输完成时为止。

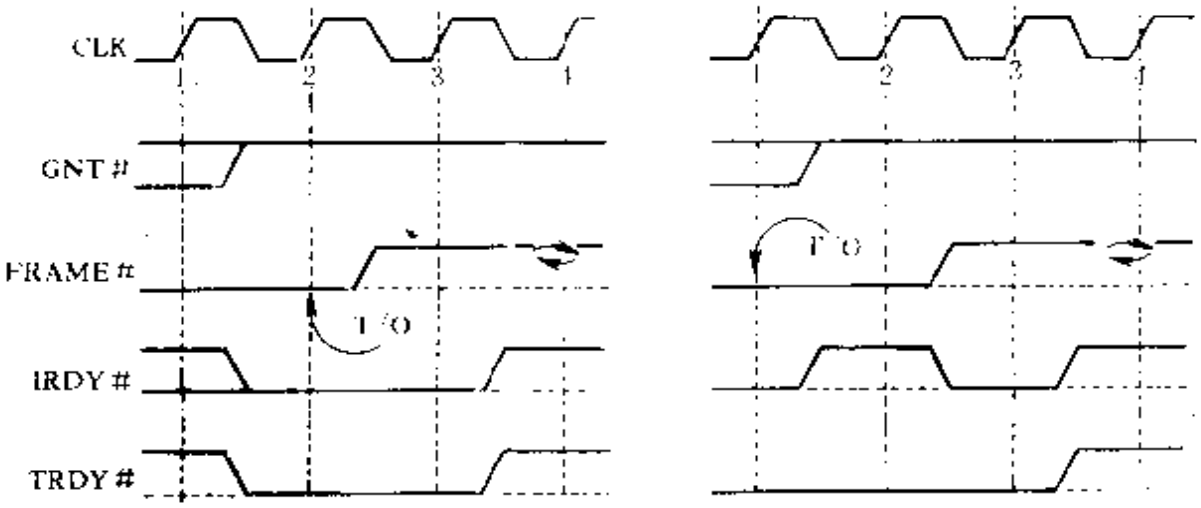


图 3.4 主设备提出的终止

图 3.4 中，也可以认为是因超时而引起的传输终止。在这种情况下，FRAME# 在时钟 3 处的无效就应看成是由于计时器已满而造成的。此时 GNT# 撤消并且主设备已准备好最终数据传输(因 IRDY# 有效)，但因 GNT# 无效，不允许使用总线，应该终止。(不过，若使用的是存储器写并无效命令，则可继续使用总

线，因该命令引起的传输不受延时计数器的限制，并且这种传输只有在一个 Cache 行的边界上才能停止进行传输）其终止的具体过程与第一种方式一样。如果 TRDY＃在时钟 2 处无效，则终止便后延，最后数据期也要后延，直到 TRDY＃有效为止。值得注意的是在这种情况下，不一定要做完原来进行的传输。

2. 由从设备提出的终止

如果从设备希望终止一次传输，就应向主设备发出 STOP＃信号以示请求。只要 STOP＃信号一有效，就必须保持到 FRAME＃信号撤消为止。在此期间，IRDY＃和 TRDY＃间的关系与 STOP＃和 FRAME＃间的关系是相互独立的，也就是说，在目标设备要求终止传输的操作中，数据的传输可有可无，具体情况取决于当时 IRDY＃和 TEDY＃的状态。但有一点应说明：当从设备发出 STOP＃信号同时又使 TRDY＃无效时，则表明从设备将不再传输任何数据，这也意味着，主设备在此时不必等待最后一次的数据传输，而使整个操作结束。

从设备之所以要求终止当前的传输操作，可能基于以下两种理由：

（1）由于死锁、某些非 PCI 资源处于非空闲状态及该设备处于互斥访问的锁定状态等原因，使得当前从设备无法进行正常的传输，不得不要求终止相应地传输操作。也就是说，从设备目前尚无数据传输。通常把这种情况也称为再试。

（2）此种情况下的终止，通常也称为断开。是由于从设备在八个时钟周期内不能对主设备作出响应，因而只好要求停止传输。但断开往往不会发生在第一个数据期，也就是说，一般在进行了一些数据传输之后才会发生。

对于大多数从设备都要求能实现再试功能，而其它终止方式可作为选项。对于哪些不支持互斥访问、不能检测死锁条件、不会陷入需要它们拒绝的一个访问之中的设备，也可将再试作为选

项。但对于主设备要能够处理从设备以任何方式提出的终止请求。

再试方式较为容易，下面我们举例再对断开方式作进一步地说明。

例 1 如图 3.5 所示。

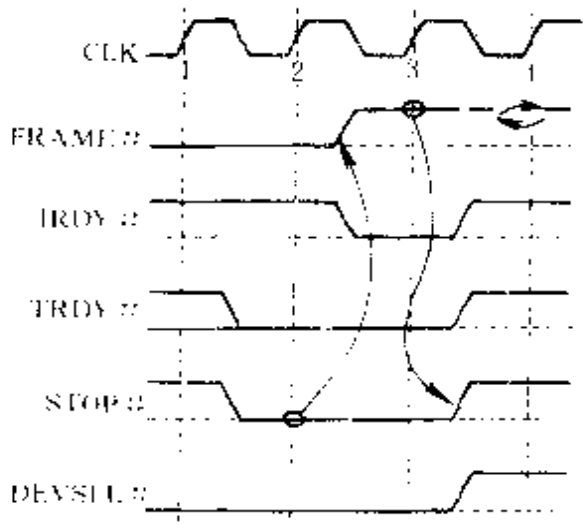


图 3.5 断开情形 1

此例中，由于从设备在发出 STOP # 信号时，同时使 TRDY # 有效，从而表明它还要进行一次数据传输才终止。从图 3.5 中可以看出，在时钟 2 处 STOP # 已稳定有效，因此接下来主设备就将 FRAME # 撤消并建立 TRDY #，表示接受了从设备的终止请求，并为最后一次数据传输做好了准备。数据的传输发生在时钟 3 处。

例 2 其时序关系如图 3.6 所示。

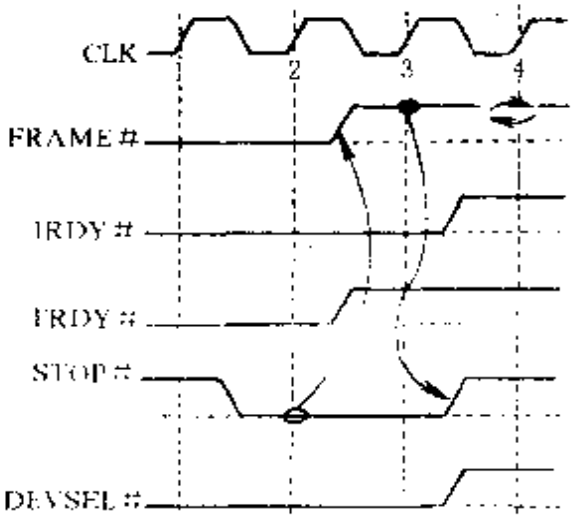


图 3.6 断开情形 2

该例与例 1 一样，在 STOP # 发出之后也有数据要传输，但不同之处在于例 1 的数据传输发生在 FRAME # 撤消之后，而本例中则发生在 FRAME # 撤消之前(时钟 2 处)。当从设备发现数据已经完成，就应把 TRDY # 撤消，也就是说，在该例中，最后一个数据期没有数据传输。

例 3 时序关系如图 3.7 所示。

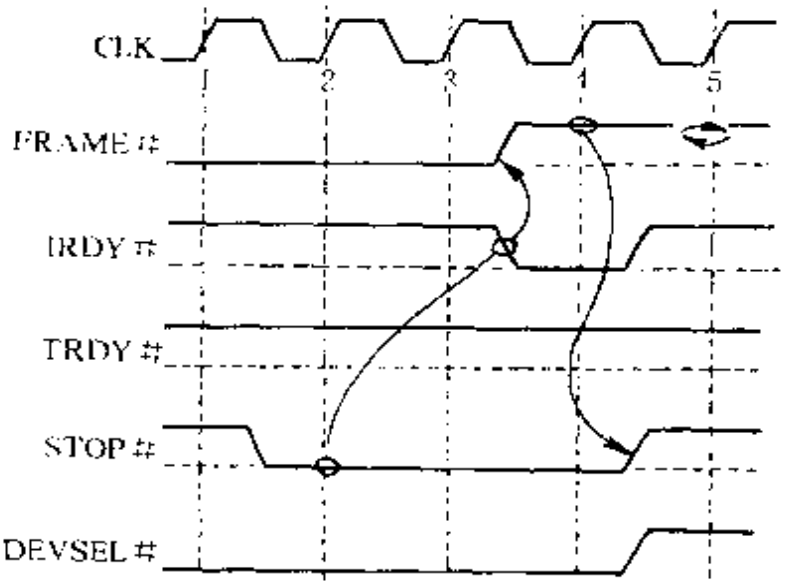


图 3.7 断开情形 3

从图 3.7 中我们可清楚地看到，在整个终止过程中，未发生任何数据传输，这也是它与前两个例子的重要区别。实际上，该例的本质是属再试方式，是断开的一种特殊情况。另外，该例中的 FRAME#的撤消被延迟到时钟 3 之后，是因为等待 IRDY#的发出。

从上面三个例子，我们也可总结出从设备终止的一些特点：

(1) 一旦 STOP#信号发出后，它就必须维持其有效状态直到 FRAME#撤消为止。

(2) STOP#信号发出后，FRAME#要尽快撤消，而 FRAME#撤消后，STOP#也必须在接下来的周期内立即无效。

(3) 在 STOP#有效期间，DEVSEL#也处于有效状态。

(4) 从设备可以决定在 STOP#发出之后是否还进行一次数据传输，若希望在停止之前进行传输，就要使 TRDY#在发 STOP#的同时有效。

(5) 从设备不能在 STOP#撤消后继续传输数据，如果主设

备要想将一被终止的传输重新启动，那它就要在稍后的时间内用下一个未传输的数据地址来启动。

(6) 在从设备发出 STOP# 信号之后的数据传送不能多于一次。

(7) 当现行的传输被从设备终止时，主设备必须撤消它的 REQ# 信号，并保持至少两个 PCI 周期；若主设备要继续完成被终止的传输，它必须在撤消 REQ# 两个时钟周期之后立即重置 REQ# 有效，否则就会失去机会，只有等到以后再次使用总线时再发 REQ# 信号。

总之，综合前述内容，在 PCI 总线上的所有传输操作中，FRAME#、IRDY#、TRDY# 和 STOP# 这几个信号一般都遵循下列规则：

(1) 当 STOP# 信号有效时，FRAME# 应在其后的 2～3 个时钟周期内尽快撤消，但在撤消时，应置 IRDY# 为有效。而目标设备应无条件地保持 STOP# 的有效状态直到 FRAME# 撤消为止。但是，一旦 FRAME# 撤消，紧跟着 STOP# 也必须撤消。

(2) 任何时钟前沿，若 STOP# 和 TRDY# 同时有效就表示是传输的最终周期，IRDY# 要在下一时钟的前沿之前撤消，也就是表示传输的结束。

(3) 对于被目标终止的对话，主设备若要继续完成它，就必须用下一个未传输数据的地址来再试访问。

(4) 一旦从设备发出了 TRDY# 或 STOP#，它就不能改变 DEVSEL#、TRDY# 和 STOP# 信号，直到当前的数据期完成。

3.4 PCI 总线的仲裁机制

为了使访问等待时间最小，PCI 的仲裁机制是基于访问而不是基于时间。总线管理必须为总线上的每一个访问执行仲裁，也

就是说，一个总线主设备要想在总线上进行访问，必须提出仲裁要求。PCI 总线执行中心仲裁方案，每个主设备都应有各自的请求线 REQ# 和批准线 GNT#，要想得到总线的控制权，必须履行相应的请求——批准手续。仲裁是"隐含的"，也就是说，一次仲裁可以在上一次访问期间完成，这样就使得仲裁的具体实现不必占用 PCI 总线周期，但是，如果在总线空闲期，就不一定采用隐含方式。中心仲裁机构必须实现一定的特殊仲裁算法，因为它是最坏情况下的仲裁基础，通常采用轮转优先级、公平性等仲裁算法。在具体实施中，系统设计者可以选择或修改算法，这是因为，仲裁算法从根本上看并不属于 PCI 总线技术规范。但有一点应注意，系统设计者在考虑具体算法时，必须为所选用的 I/O 控制器及其接插件所要求的访问延迟提供保障。关于仲裁器的实现方案，在 PCI 总线规范中未作具体规定(即是任意的)，但一定要保证在任何时间只能有一个 GNT# 信号成立。

3.4.1 仲裁协议

PCI 总线仲裁中主要利用 REQ# 和 GNT# 两个信号线实现，前者用于某一设备占用总线的请求，后者允许某一设备占用总线和应答。换而言之，一个设备依靠发出它的 REQ# 信号来请求使用总线，当仲裁器根据当时情况决定出一个设备可以使用总线时，就发出该设备的 GNT# 信号。对于一个设备，只有当它真正需要使用总线时才能发 REQ#，绝对不能利用 REQ# 把自己"停靠"在总线上。即使要实现总线的停靠，也要由仲裁器来指定缺省的总线拥有者。

仲裁器可以在任何时钟上置某一设备的 GNT# 无效。若某一设备要利用 PCI 总线传输数据时，必须确保它的 GNT# 信号在此时有效。下面给出 PCI 总线仲裁的基本协议规则：

(1) 若设置了 GNT# 信号无效而 FRAME# 有效时，当前的

数据传输是合法的且继续进行下去。

(2) 如果总线不是处在空闲状态，则一个设备的 GNT＃信号有效和另一个设备的 GNT＃信号无效之间必须有一个延迟时钟，否则会在 AD 线上和 PAR 线上出现时序竞争。

(3) 当 FRAME＃无效时，为了响应更高优先级主设备的服务，可以在任意时刻置 GNT＃和 REQ＃无效。若总线占用者在 GNT＃和 REQ＃设置后，在处于空闲状态 16 个 PCI 时钟后，仍未开始数据传输，仲裁器允许当前主机“打破”这个状态。仲裁器也可以在任意时刻移去 GNT＃，以便服务于一个更高优先级的设备。

上述三条也可以说是 GNT＃信号的撤消原则。下面通过两个设备对总线的占用情况来说明仲裁的基本过程。其时序关系如图 3.8 所示。

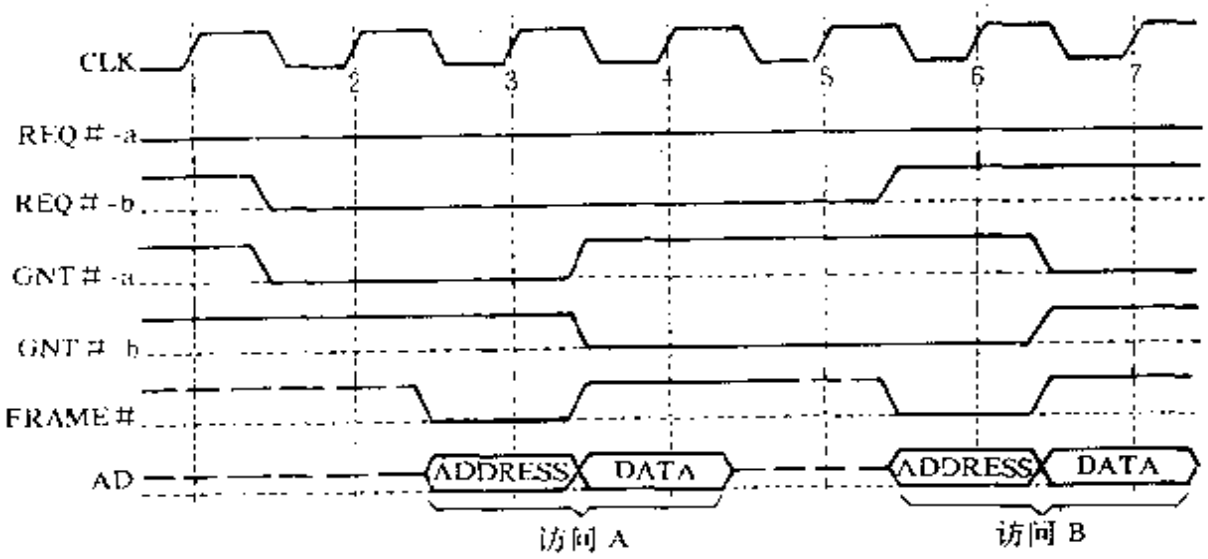


图 3.8 仲裁的基本过程

为了讨论方便，我们假设两个设备分别为 A 和 B，从上图可看出，在时钟 1 或在此之前设备 A 发出了 REQ＃－A 请求使用总线，仲裁器根据总线的使用情况，于时钟 2 处发出 GNT＃－A 信

号，表示设备 A 的请求获得批准，可以使用总线。由于此时只有 GNT#-A 有效，而 FRAME# 和 IRDY# 无效，所以设备 A 可以在时钟 2 启动一次传输。到了时钟 3，FRAME# 信号有效，设备 A 便可开始其数据的真正传送。兹于它的总线请求信号 REQ#-A 一直未撤消，说明它还要进行另外一次传输。

前面我们曾提到总线的仲裁是隐含的，因此，在设备 A 获得批准(时钟 2)到它的数据开始传输(时钟 3)期间，仲裁器同时决定出 B 设备应为下一个总线使用者。这是因为在设备 A 占用总线期间，设备 B 也提出了使用总线的请求(时钟 1 过后 REQ#-B 有效)，所以，当设备 A 在时钟 4 传完数据时，仲裁器便撤消了 GNT#-A 而置 GNT#-B 有效，允许设备 B 使用总线。这样，设备 A 在时钟 4 之后就释放总线，同时 FRAME# 和 IRDY# 也消失，从而使得所有的 PCI 设备都能够判断出当前数据传输已经结束。设备 B 在时钟 5 处成为总线的拥有者，可以启动开始传输，并在时钟 7 完成相应的传输。

另外，从图 3.8 中还可看到在时钟 6 处 REQ#-B 信号撤消而 FRAME# 信号有效，这表明设备 B 只请求进行一次数据传输，因而仲裁器便在此期间批准设备 A 为下一个主设备(由于 REQ#-A 仍然存在)。

从上面的讨论可以得出如下结论或者说是注意事项：

(1) 如果当前的总线占有者要求进行多次数据传输时，应保持其总线请求信号 REQ# 一直有效。假如没有其它设备提出请求或者该设备具有最高优先级，仲裁器便会批准它继续占用总线。

(2) 一个设备在其获得总线占有权期间只能进行一次数据传输，若想再次传输，它就要继续发出请求信号 REQ#。

(3) 一个设备可在任何时刻撤消它的 REQ# 信号，但这时仲裁器也就认为该设备不再请求使用总线，随之撤消相应的 GNT# 信号。

(4) 倘若某设备只需要占用总线做一次数据传输，那么它必须在发出 FRAMF＃信号的同时撤消 REQ＃信号。

(5) 如果一次传输因从设备发出 STOP＃而被终止的话，相应的主设备必须把它的 REQ＃信号撤消，并且要保持这种撤消状态至少达两个 PCI 周期。即使这个主设备打算完成被从设备终止的传输，它也必须先撤消 REQ＃，然后重新发出 REQ＃信号再次提出请求。否则，它可以等以后需要使用总线时再发 REQ＃。这样做的目的是为了给其它设备留出使用总线的机会，以及给前一个从设备留出时间为下一次传输做准备。

3.4.2 仲裁的停靠

所谓停靠是指总线仲裁器在没有设备使用总线或者也没有设备请求使用总线的情况下，根据一定方式选定一个设备给它发出 GNT＃信号，从而选择一个缺省的总线拥有者，这有点类似于计算机常用的缺省配置。缺省所依据的方式一般采用固定为某一设备或选择最后一次使用总线的设备等，有时也可指定自己(仲裁器)为缺省的拥有者。

当仲裁器在总线空闲期间给一个停靠的设备发出 GNT＃信号时，该设备必须在 8 个时钟周期内将其 AD[31：：00]、C/BE[3：：0]＃及 PAR 的输出缓冲器打开，但是并不要求一定要在某一个时钟周期之内将全部的输出缓冲器打开，只要在 2～3 个时钟周期内完成即可。PAR 相对于 AD 和 C/BE 信号可延迟一个时钟周期。这样做的目的是为了保证总线能安全地停靠于某个设备上，从而使总线不致于浮动。如果仲裁器不将总线实行停靠，则总线将由仲裁器所在的中央资源进行驱动。

在总线空闲时，除非仲裁器在撤消一个设备的 GNT＃信号时赶巧碰上该设备发出 FRAME＃方可进行传输，否则，无论何种情况，该设备都将失去对总线的访问权，而且应把 AD[31：：00]、

C/BE[3：：0]#及 PAR(延迟一个时钟周期)变为第三态。这一点与前述的打开它们的输出缓冲器有所不同，它要求该设备必须在一个时钟周期内禁止全部输出缓冲器(断开输出)，以防止与下一个总线拥有者发生竞争。

根据以上的情况可以得出，在 PCI 总线上，从总线空闲周期开始到仲裁完成所需的最小延迟为：

(1) 具有停靠时，停靠的设备为 0 个时钟周期，其它设备为 2 个时钟周期；

(2) 没有停靠时，每个设备都为 2 个时钟周期。

最后要说明的是，当总线停靠于一个设备时，该设置不用发 REQ#信号就可以开始一次传输(只要总线空闲且 GNT#有效)，如果它要进行多次数据传输，就需要发出 REQ#，向仲裁器提出多次传输的请求；而只要求一次传输时，就不应当发 REQ#，以免仲裁器在它不需要使用总线的情况下又给它发批准信号 GNT#。

3.5 PCI 总线的访问延迟

PCI 总线是一种吞吐量高、访问延迟小的 I/O 总线。本节主要对预测并控制该总线在最坏情况下的访问延迟之机制加以描述，以使读者在对一个独立的 PCI 环境的访问延迟进行预测时，具有较高的准确性和精确性。但是，如果系统中含有一个标准扩展总线(ISA、EISA 或 MC 总线)，那么访问延迟的预测会变得更为困难，这是因为，此时最坏情况下的访问延迟受到扩展总线或适配器的限制可能要大于 PCI 总线的限制。

3.5.1 PCI 总线上访问延迟的概念

PCI 总线上设备的访问延迟由三部分时间组成，如图 3.9 所

示。其中的第一部分为仲裁延迟，是指主设备从发出 REQ＃信号到收到 GNT＃信号这一段等待时间，其长短取决于仲裁算法、请求设备的优先级和系统的利用率。对于优先级最高的设备，一般是 2 个时钟周期；第二部分是总线获取延迟，也就是设备需要等待多长时间才能使用总线；最后一部分是目标延迟，是指目标设备为第一次数据传输发 TRDY＃信号所需的时间。

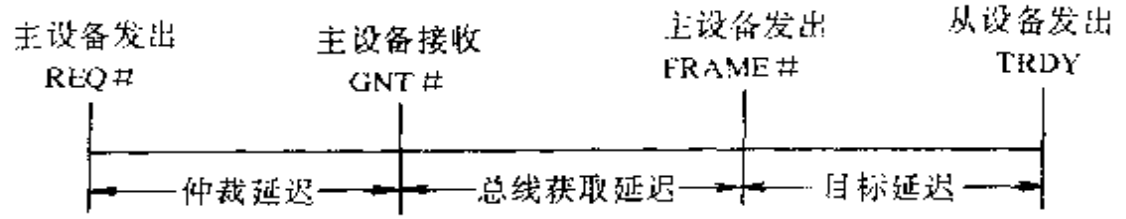


图 3.9　访问延迟的组成

只要目标设备能够发出或接收数据，则 PCI 主设备的突发传输长度不受限制，并且在没有其它设备请求使用总线的情况下，传输可以一直进行下去。为此，在 PCI 总线中采用两种方式以决定一个主设备在有其它请求的情况下占有总线的时间长短，从而使总线获取延迟成为可预测的。

1. 主设备延迟计数器(LT)

每个主设备的 LT 在该设备未发 FRAME＃信号时是被清零且挂起的。当主设备发出 FRAME＃时，它的 LT 开始计数。如果该设备在计数器溢出之前就撤消了 FRAME＃信号，则它的 LT 就不起作用。但是，一旦计满，该设备必须在它的 GNT＃撤消时立即发动传输终止。实质上，LT 的最大计数值(以时钟周期为单位)表示分配给该设备的最小保证时间片，超过这个时间，只要 GNT＃无效，就必须立刻停止占有总线。

2. 目标发动的终止

目标设备对 TRDY＃和 STOP＃两个信号要具备处理能力，使它耗费的数据期达到 N 时(N＝1，2，3，…)就结束传输。这里 N

的具体确定条件是 N＋1 个数据期的延迟增量大于 8 个时钟周期。例如，设一个 PCI 主设备从扩展总线读出要用 15 个时钟周期，利用上述规则令 N＝1，这样，到第二个数据期的访问延迟是 15 个时钟周期，因此目标设备必须在完成第一个数据期时终止传输。

对于第一个数据期的访问延迟没有限制，例如，假设在一个系统中没有从设备完成第一数据期需要 TRDY＃的延迟大于 T＋8(T 为 LT 的计数最大值)个时钟周期，根据这一假定及上面叙述的机制可以得出，主设备完成传输的最长时间为 T＋8 个时钟周期(假设它的 GNT＃在 LT 计满之前就撤消了)。若 T＝40，则最大的传输时间为 48 个时钟周期，这在 33 MHz 下，相当于 1.44 μs。

实际上，T 代表了吞吐量和访问延迟之间的折衷。例如，如果主设备和从设备都能以 0 等待周期突发传输数据，并且第一个数据期的延迟是 8 个时钟，那么当 T＝40 时可做到连发 40－8＝32 个数据期；若把 T 降为 20 则突发传输次数降为 12～14 次，但这样可以把传输的时间限制在 28 个时钟之内，即 0.84 μs。

3.5.2 访问延迟的确定

随着每一个特定系统中所用设备的不同，其具体实现中的访问延迟也会发生变化，这就要求 PCI 设备的制造者们应具备一些关于典型的访问延迟的确定措施，以使他们在决定芯片的缓冲能力时尽量减少失误，从而做到既充分又不过量。

从图 3.9 可以看出，总线上的一个访问延迟可分为三种访问延迟成分。仲裁延迟对于最高优先级的设备一般为 2 个时钟，而低优先级设备的仲裁延迟取决于较高优先级设备的数量及使用程度，但比起前者要长一些。总线获取延迟的确定分两种情况，一种是取系统中第一数据期的最长延时；另一种是当所有的第一数

据期延时都比较低时，取延时计数器(LT)的值。这里第一数据期延时是指从 FRAME# 信号有效到 TRDY # 信号发出之间的时间间隔。对于访问延迟的第三部分——目标延迟，一般取第一个数据的延迟。但是对 CPU 桥上的又经过缓冲的设备，其目标延迟可能会长一些。例如，当某设备通过有缓冲的 CPU 桥去访问主存时，该桥可能需要先冲洗缓冲器，才允许访问进行。这样一来，就使得目标延迟加大，其增加程度取决于冲洗面向何方和要冲洗缓冲区的大小。

下面给出几个例子说明几种在 PCI 上发生的不同事件序列的延迟计算方法，并且作如下的前提假设：

(1) PCI 总线的运行时钟为 33 MHz；

(2) 所有非扩展总线设备的第一数据延迟为 16 个时钟，即 0.5 μs，以后的突发传输中，数据期间隔为 8 个时钟；

(3) 扩展总线上的字节访问需要 1.5 μs，以使通过一个扩展总线进行 32 位访问时，第一数据延时达到 6 μs。

例 1 一个主设备在请求使用总线时，正好 PCI 总线处于空闲状态，也就是说不会出现与其它设备发生竞争的现象。

计算方法：仲裁延迟一般取两个时钟。因为总线空闲，故总线获取延迟为 0，目标延迟为 16 个时钟。总延迟为

$$T_L = 2+0+16 = 18 \text{ 个时钟} = 18 \times \frac{1}{33\ \text{MHz}} = 0.54\ \mu\text{s}$$

例 2 当一个主设备请求使用总线时，正好有另外一个设备刚刚开始在总线上传输数据。假定使用总线的设备要进行 16 个数据期的连发，且系统中的延时计数器 LT 为 66 个时钟，也就是 2 μs。

计算方法：由于没有其它请求等待处理，因此仲裁延迟仍取 2 个时钟；总线获取延迟包括第一数据期的 16 个时钟、LT 计满时所发的 6 个数据期耗费的 $6 \times 8 = 48$ 个时钟及为照顾最后数据

期的 8 个时钟，共计需 72 个时钟；目标延迟取 16 个时钟。在此例要特别注意仲裁延迟和总线延迟是重合的。总的访问延迟为

$$T_L = 2 + 72 + 16 - 2 = 88 \text{ 个时钟} = 2.7\ \mu s$$

例 3 一个主设备通过 CPU 桥请求总线传输，该桥有缓冲区需要冲洗，缓冲区的大小为 4 个字节，冲洗目的地为一扩展总线。

计算方法：仲裁延迟仍然是 2 个时钟；总线获取延迟为 0；目标延迟除了前述的 16 个时钟外，由于要进行缓冲区清洗，需要加长。具体情况是，CPU 桥要请求总线以进行冲洗，需 2 个时钟，整个冲洗过程需 $4 \times 1.5\ \mu s = 6\ \mu s$(200 个时钟)；原来请求使用总线的主设备需发再试信号以再次请求总线，需 2 个时钟。因此总的访问延迟为

$$T_L = 2 + 4 + 200 + 16 = 222 \text{ 个时钟} = 6.73\ \mu s$$

3.5.3 访问延迟的处理原则

PCI 总线能适合于吞吐量和访问延迟要求都比较高的设备。例如，当 CPU 发起一次屏幕刷新时，帧缓冲区的读写可能要耗用很高比例的总线带宽，但是，只要典型的访问延迟不长，偶而出现这种不常见的长的访问还是可以容忍的。在 PCI 总线中，诸如不限长度的突发传输(高带宽 I/O 块)和总线停靠(高带宽随机 I/O，如帧缓冲区读写)等特性，都有利于设备吞吐率的提高。另一方面，一个 10 Mb/s 的 LAN 设备只消耗很小的 PCI 带宽，为了保证类似设备使用的经济性，可利用 PCI 中像基于访问的仲裁和主设备延迟计时器等特性来限制最坏情况下的延迟，以尽量减少缓冲区要求，从而达到目的。例如一个深度缓冲的 FDDI 设备，本来在一次访问中可连发 128 个 DWORD(双字)，但在 PCI 总线中，可能被它的延时计时器限制而拆成几次发送，从而可使缓冲的深度(大小)减小许多。

在大多情况下，经常访问总线的设备都是对访问延迟较为敏感的设备，加之高吞吐率和低延迟往往又是互不相让的，所以PCI设备和系统的良好设计就显得尤为重要，力求做到使得跟访问延迟有关的相互操作问题最少，以使其性能价格比趋于平衡而合理。

对于大多数PCI系统，典型的访问延迟约在2 μs以下，不算太长，但是最坏情况下的访问延迟虽然少见却可能很长，并且有时无法估计。例如，通过一个桥对标准扩展总线适配器进行访问，其访问延时往往是由适配器决定而不是取决于PCI总线。为了补偿，主设备上必须提供30 μs的缓冲能力以保证其最坏情况下的访问延迟限制，这样，即使有最坏情况的不确定性，但30 μs对实现系统设计来讲应当可以提供足够的富裕度。这样做虽然有些保守，但针对最坏情况的不确定性，30 μs还是必要的。

对于一个主设备，如果可以忍受一个偶而出现的访问延迟超限并不产生错误，那么为了减小最坏情况下的设计代价，可以按小于30 μs的值进行设计。例如，声音信号这样的数据流可以容忍一次偶然的取样丢失。

有些应用(如多媒体)可能对访问延迟环境的控制要求更加严格。同时要求从设备具有快速的、可预测的、受约束的访问延迟行为。对于此类设计，建议在目标接口的设计中遵循下列原则：

(1) 单层的目标设备(单层是指一个PCI目标提供对选定资源的直接访问)应把第一个数据期延迟限制在16个时钟以内。如果因暂时的原因(DRAM刷新等)会加长访问延迟使它超过16个时钟，就立即发再试。

(2) 多层目标设备(多层是指一个目标设备为了访问选定的资源还必须取得一个独立仲裁的资源)要能够使由于资源忙碌而冲突的访问再试。例如，一个对EISA从设备进行访问的设备，当发现EISA总线被另一主设备占有时，应当立即发再试信号。

(3) 对于多层目标设备(特别是总线与总线之间的桥)的写缓冲策略应多加小心，由于写缓冲使得对访问延迟的限制变得很困难。之所以这样，是因为很强的顺序要求迫使在一条总线上排队的写操作全部完成之后，才能允许进行其它的访问。这里特别强调的是，一个宿主 CPU 到 PCI 总线的桥要能把写数据转发到一个常驻 PCI 的帧缓冲器而同时阻止发往 ISA/EISA/MC 总线的写操作。

总之，为了便于做出良好的系统级的取舍，设备供应商应当清楚地写明其设备在访问延迟方面的情况。

3.6 PCI 总线的互斥操作

PCI 总线具有互斥访问功能(机制)，但该机制并不影响非互斥访问的进行，有时也将此功能称为资源锁定。如此设置是为了将来的处理器可以在几个访问之间保持一个硬件锁，而又不会扰乱像视频数据传输这类实时数据传输。该机制的实现是基于仅仅锁定那些被原来的锁访问作为目标的 PCI 资源，同时它与现有的互斥访问软件是完全兼容的。

资源的锁定是通过 LOCK＃信号实现的。PCI 规定，凡是提供系统存储器的设备必须实现 LOCK＃信号；尤其是具有可执行存储器的设备更是如此，以保证在该存储器中进行访问的独占性。由于主桥的后面有系统存储器，所以它也必须实现 LOCK＃信号。

LOCK＃的有效预示着一个互斥访问在潜伏进行。当一个设备的 GNT＃有效时，并不能保证它对 LOCK＃的控制，为获得对 LOCK＃的控制，需将 GNT＃信号和 LOCK＃自身的规程结合起来才能实现。

如果使用了资源锁定，设备的非互斥访问可依然进行而不受限制，那怕此时有另一主设备持有对 LOCK＃的所有权也没有关

系。然而对于兼容性有较高要求时，可利用仲裁器将一资源锁定为总线锁，具体做法是在 LOCK＃信号放开之前，一直批准拥有 LOCK＃的设备对总线的互斥访问。

在资源锁中，访问的独占性是由被访问的从设备保证的，而不是依靠排斥其它设备对总线的访问。锁的粒度定义为 16 个对齐的字节，对 16 个字节内的任何字节进行互斥访问时，都会锁住 16 个字节的全部，主设备不能通过 16 个字节之外的地址去实现对 16 个字节块的锁定。对于目标设备也要能实现锁定，要求最少锁住 16 个字节块，最大到整个资源。

总之，在 PCI 总线中，LOCK＃信号应遵循以下规则：

(1) 在地址期，当 LOCK＃被置为无效时，执行访问的从设备要将自身锁住。

(2) 一旦建立了锁，目标将保持锁住状态，直到它取样发现 FRAME＃和 LOCK＃一起撤消，或者直到发出目标废止。

(3) 保证 LOCK＃信号所有者的独占性，一旦锁已建立，至少有 16 个字节，最大可锁住整个资源。

(4) 锁不能跨越设备界限。

(5) 锁操作的第一个传输必须是读传输。

(6) LOCK＃必须在地址期之后的第一个时钟上被设置，并保持设置以继续控制。

(7) 在数据期结束之前，再试被告知还有锁建立，则 LOCK＃被释放。

(8) 无论何种情况，只要一次访问被主、从设备打断时，LOCK＃必须释放。

(9) 在连续的锁操作中，LOCK＃必须被置成一个最小的空闲周期。

3.6.1 一个互斥访问的开始

如果一个设备需要进行互斥访问时，在它发 REQ# 之前应跟踪 LOCK# 的状态。在任何时候，只要 LOCK# 有效，主设备就告知 LOCK# 是忙碌的，只有在 FRAME# 和 LOCK# 同时无效时，主设备才提供 LOCK# 空闲的信息。如果 LOCK# 处于忙状态，请求互斥访问设备就应等到 LOCK# 可用时再发 REQ# 信号。

图 3.10 表示一个主设备发动一个互斥访问。

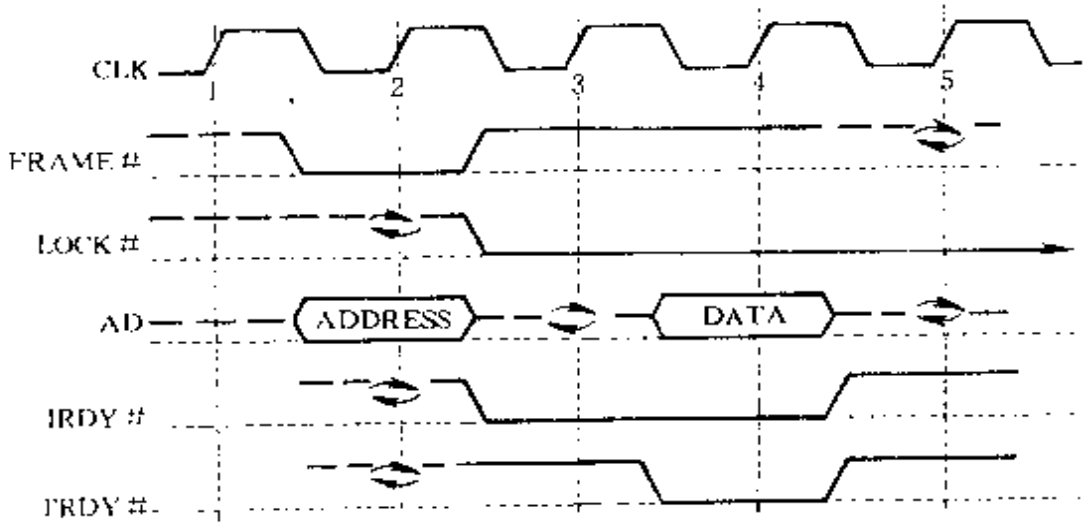


图 3.10 一个互斥访问的开始

LOCK# 在地址期之后立即发出，在互斥访问完成之前它一直受该主设备的控制。

一个被锁定的操作状态只有在第一个传输的第一数据期(IRDY# 和 TRDY# 同时有效)完成时才能建立起来。如果从设备未完成数据传输而将第一个对话再试，那么主设备就必须终止传输并释放 LOCK# 信号。反之，若第一数据期顺利完成，互斥操作就已建立，并且主设备将一直保持 LOCK# 信号的有效直到互锁操作的完成，除非由于一个错误而引起提前终止(主设备或从设备废止)。但是在锁操作建立后，目标设备终止中的再试或断开

均属于正常终止，如果这种终止情况发生，它仅表明该目标设备目前处于忙碌状态，无法完成要求的数据传输，该目标设备将会在以后空闲时接受访问，同时仍然承认以前的锁操作而排斥其它访问，相应的主设备继续实施对 LOCK＃的控制权直到当前的互斥访问完成后将其置为无效，此后，其它主设备才有可能取得对它的控制权。

3.6.2 一个互斥访问的延续

图 3.11 表示了一个主设备对一个互斥访问的继续。可以看出，在时钟 1 之前该主设备已经在进行互斥操作，因此在时钟 3 LOCK＃又一次有效就表明该设备将继续进行互斥访问。具体过程是，当主设备得到访问总线的批准时，就开始对被它以前锁住的从设备进行另外一次互斥访问，但应注意，另外一次互斥访问的锁定状态应重新建立，因此在地址期内要撤消 LOCK＃，以便锁的重建(见图 3.11 的时钟 2 处)，下一个 LOCK＃信号在时钟 3 发出，被锁设备应当接受并响应这一请求，以保持目标设备的锁状态并允许当前主设备在当前传输完成之后继续拥有对 LOCK＃信号的控制权。

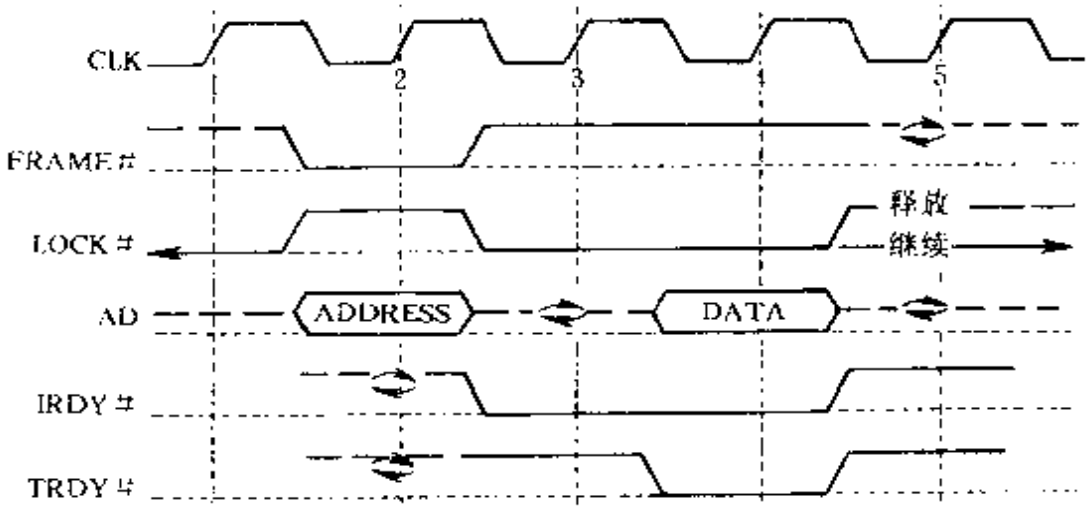


图 3.11 一个互斥访问的继续

另外，上述访问的延续也可能还不是整个互斥访问的最终完成，也就是说可能还要进一步延续。如果主设备继续锁操作时，就接着发 LOCK＃信号，当它完成锁操作时，便在最后数据期之后的时钟 5 处置 LOCK＃信号无效。

3.6.3 锁定状态下非互斥访问的进行

图 3.12 给出了一个主设备试图对一个已经被锁住的设备进行非互斥访问的时序关系。

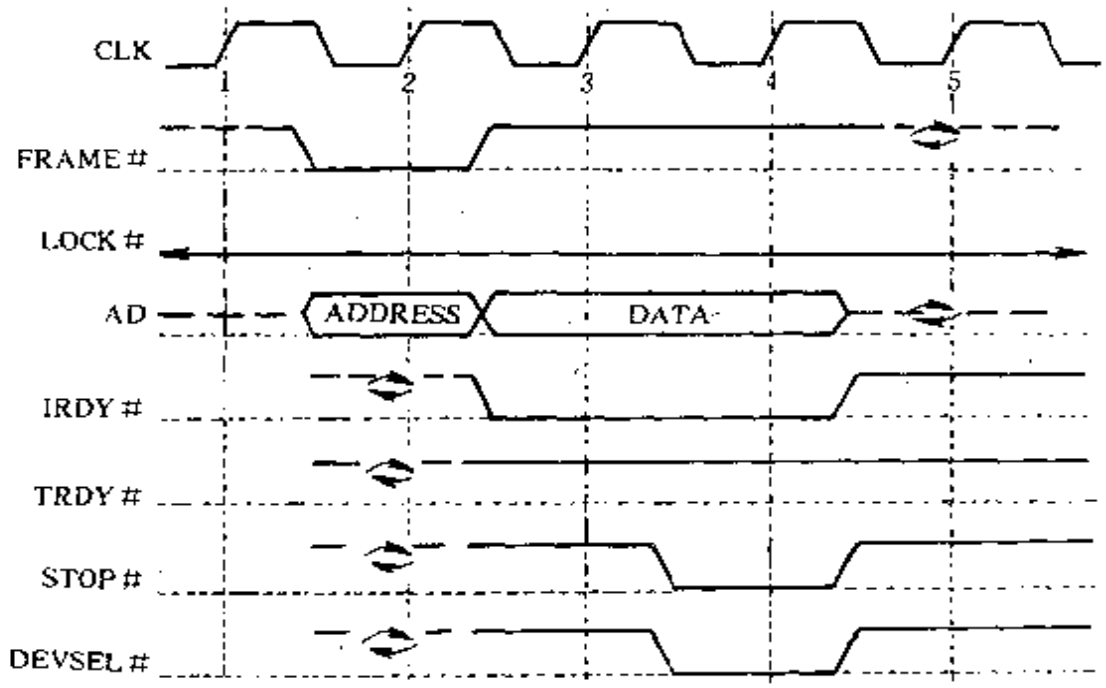


图 3.12 锁定状态下的非互斥访问

从图中可以看出，当 LOCK＃在地址期中有效且目标是锁着的时候，目标不会进行任何数据的传输，只是回答一个再试。反之，如果目标设备是未锁的，它就不会理睬 LOCK＃的有无，只是决定是否响应主设备的访问。另一方面，因为在地址期 LOCK＃和 FRAME＃均为有效状态，所以未锁的设备不会进入锁定状态。

3.6.4 互斥访问完成

在互斥访问的最后一次数据传输期间，如果目标设备接受了请求，LOCK＃信号就要撤消。作为主设备可以在互斥操作完成后的任何时刻置 LOCK＃为无效。但是最好(并不是一定)将 LOCK＃和 IRDY＃两个信号同时在互斥操作的最后数据结束后撤消，这主要是防止在任何其它时间释放 LOCK＃后，可能会造成以后的传输被不必要的再试而终止。对于被锁的节点，只要它发现 LOCK＃和 FRAME＃都无效时便可自行解锁。

如果一个主设备想要在总线上进行两个相互独立的互斥操作，它就必须能保证在这两个互斥操作之间，FRAME＃和 LOCK＃至少有一个时钟周期处于无效状态，从而保证任何被第一个互斥操作锁住的设备能在第二个互斥操作开始之前完全释放(开锁)。

3.6.5 总线的完全锁定

PCI 的资源锁可以转变为一个完全的总线锁，也就是在 LOCK＃信号有效时，仲裁器不批准任何设备的使用总线请求。如果被锁序列的第一个访问出现再试时，主设备必须撤消它的 REQ＃和 LOCK＃信号。当第一个访问完成时，总线就被完全锁定，仲裁器将不会把总线使用权批准给其它任何设备。如果在完全总线锁建好时，仲裁器已经把总线批准给了其它设备，那么这时仲裁器必须撤掉它的批准，以保证总线的完全锁定。完全总线锁可能会对系统的性能产生严重影响，尤其是视频子系统。另外，在完全总线锁状态下，所有的非互斥访问在锁操作期间都不能进行，这是显而易见的。

不论是完全总线锁还是完全资源锁，如果遇到系统中有回写式高速缓存时，必须避免死锁问题。这就意味着，支持完全总线

锁的仲裁器必须批准高速缓存设备为完成一次回写所发的请求；支持回写高速缓存的从设备即使被锁住也应允许进行回写。

3.7 PCI 总线的其它操作

3.7.1 设备选择

一个设备是否被选中，是由 DEVSEL＃信号来指示的。DEVSEL＃由当前传输中的目标设备所驱动，如图 3.13 所示。

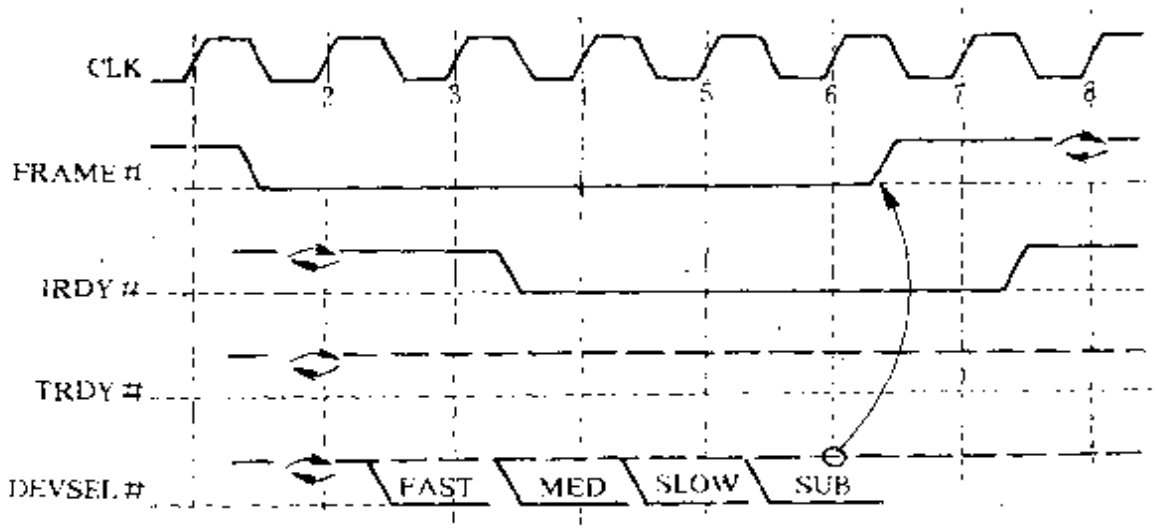


图 3.13 设备选择的时序关系

DEVSEL＃信号可在地址期之后的 1 个、2 个或 3 个时钟处被驱动，具体时刻可在配置空间的状态寄存器中指定。DEVSEL＃的发出必须早于或同时于目标设备的 TRDY＃、STOP＃或读数据的时钟边沿，也就是说，一个目标设备要先置 DEVSEL＃有效后才能发出其它目标响应信号。一旦目标设备确定了 DEVSEL＃信号，就不能在 FRAME＃被撤消而 IRDY＃有效时和最后数据期完成之前撤消它。在正常的主设备终止情况下，DEVSEL＃的撤消必须与 TRDY＃的撤消同时发生。

如果在 FRAME＃有效后的 3 个时钟周期内，没有设备发出

DEVSEL＃信号，则按负向译码的设备便可以置 DEVSEL＃有效并拥有传输的权力。倘若整个系统中没有一个负向译码的设备，则主设备就收不到有效的 DEVSEL＃信号，它便以自己的方式终止传输。

一个目标设备必须在进行了完全的地址译码之后才能发出 DEVSEL＃或任何其它的目标响应信号，否则设置 DEVSEL＃有效是非法的，因为这样可能造成冲突。在非配置命令中，目标设备必须先用 FRAME＃信号来认可 AD 线上的信息，然后才能设置 DEVSEL＃信号；而在配置命令中，目标设备要先用 FRAME＃和 AD[1：：0]来认可 IDSEL 信号后再设置 DEVSEL＃信号。一般来说，大部分目标设备都能在 FRAME＃有效之后的 1～2 个时钟周期内完成地址译码并发出 DEVSEL＃信号，如图 3.13 中的快速档和中速档。

如果一个访问中的地址落入目标节点的地址范围，这个目标就会发出 DEVSEL＃信号，表示承担这个传输访问。但是，如果主设备企图跨越资源边界以继续进行它的突发传输时，目标节点就应该发出断开信号。

当一个目标设备许诺了一个 I/O 访问，同时字节使能信号又表示出 1 个或更多的字节不在该设备的地址范围之内时，它就必须发出目标废止信号。对于一个负向译码的设备，如扩展总线桥，为了适应这类 I/O 问题，可采用如下之一的作法：

（1）凡是发现有不同的设备共用一个双字地址时，就以该地址做正向译码，然后再用字节使能信号进行检测，发现问题就实行目标废止。

（2）将整个访问都放到扩展总线上，没法服务的部分就丢掉。

3.7.2 特殊周期命令

该命令提供了 PCI 总线上简单的信息发布机制(消息广播)，用以传达处理机状态，或者发送各个从设备之间的信号。如果不要求精确的时间关系或物理信号同步的情况下，此命令也可用来在 PCI 设备之间做逻辑的侧面连接信号。

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，每个接收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；同时，采用负向译码的桥也不能把特殊周期命令传到它的次级总线上去，即特殊周期命令不能跨桥传播。

一个特殊周期命令可以含有可选的、由消息决定的数据，而且这些数据不需要由 PCI 序列器本身来解释，它是根据需要传送给与 PCI 序列器相连的硬件。通常情况下，显式寻址的消息一般不采用特殊周期命令来处理，而是利用 PCI 总线上的三种物理地址空间的其中之一来处理。

就像其它总线命令一样，特殊周期命令中也包含一个地址期和一个数据期。地址期也是在 FRAME＃信号有效之时开始，当 FRAME＃和 IRDY＃同时撤消时，也表示该命令已经完成，要说它与其它总线命令的唯一区别，那就是没有目标设备给出 DEVSEL＃响应信号，它是广播给所有设备，然后由感兴趣的设备予以接受并加以处理。

该命令在地址期中，除了发送命令之外不包含其它有效信息。但是，有一点要注意，尽管该命令不含有明显的地址信息，可 AD[31：：00]要被驱动到一个稳定的电平，同时也要产生奇偶位。在数据期内，AD[31：：00]上含有信息类型和一个选项的

数据字段，其中，低 16 位 AD[15：：00]表示信息类型；高 16 位 AD[31：：16]表示选项数据，但并不是所有的信息类型都要求高 16 位。主设备可以像对待其它总线命令一样，在特殊周期命令中插入等待周期，但是，由于没有固定的目标，所以目标设备不能插入等待周期。消息和数据只在 IRDY＃有效的第一个时钟处有效，而其后数据期中所含信息及时序关系则由消息类型决定。

在地址期中，若 C/BE[3：：0]＃＝0001，则表示特殊周期命令，此时，AD[31：：00]被驱动为随机值且必须忽略；在特殊周期命令的数据期中，C/BE[3：：0]＃要处于有效状态，而 AD[31：：00]的各位如表 3.3 所示。

表 3.3　消息类型编码表

AD[15：：00]	消息类型
0000H	SHUTDOWN
0001H	HALT
0002H	X86 有关信息
0003H～FFFFH	保留
AD[31：：16]	由消息决定的可选数据

PCI 序列器启动特殊周期命令的方式与其它的总线命令相同，但该命令的终止则要依靠主设备废止方式。相应的目标设备只要发现主设备废止时就知道访问已经完成，在此情况下，不能将配置空间的状态寄存器的“接收主设备废止位”置为“1”。一般来说，一个特殊周期命令最快可以在 5 个时钟周期内完成，同时要附加 1 个时钟周期作为下一个访问开始之前的过渡周期(交换周期)，换而言之，从一个特殊周期命令的开始到下一个访问之间，需要 6 个时钟周期。

3.7.3 数据/地址的渐进

PCI 总线上的“事务代理”将一个设备发出的信号拓展到几个时钟周期之内逐步驱动，而不是在一个时钟内完成，这种能力称为渐进(Steping)。此概念的应用，可以使一个设备以“弱的”输出缓冲将一组信号在几个时钟周期内逐渐驱动到有效状态，从而减少了由每个输出驱动所产生的地电流负载。还有一种渐进的方法是：允许设备以“强的”输出驱动在几个时钟周期内分批驱动，每个时钟周期驱动一部分信号，直到全部被驱动到有效状态为止，称为分批渐进，此方法可以减少同时开关的信号数量。

上述两种渐进方法，无论哪一种都能使一个芯片的电源线与地线数减少，从而使得设备的性能与价格之间具有取舍自由。但应注意，在采用全部渐进时，必须避免那些要求严格的控制信号之间的相互耦合，这是因为，本应在各个时钟边沿上取样的信号此时可能在一个时钟边沿上出现跳变，所以，对性能要求较高的设备应慎用这种方法。

并非所有的信号都可以渐进，只有 AD[31：：00]、AD[63：：32]、PAR、PAR64 及 IDSEL 这些信号才行，因为它们总是要由控制信号来认可，或者说，这些信号只有在它们被认可的时钟边沿上才被认为是有效的。我们知道，AD 线的确认，在地址期是依靠 FRAME＃信号，在数据期则是根据数据的传输方向分别依靠 IRDY＃或 TRDY＃信号；PAR 是在每个认可 AD 的时钟之后一个周期内隐含确认的；IDSEL 信号的确认则是通过 FRAME＃和译码得到的配置命令相结合来完成的。

图 3.14 表示了全部 AD 线的渐进情况。可以看出，一个主设备将 FRAME＃的建立一直推迟到该设备成功地驱动了全部 AD 线时为止。对于主设备来讲，一旦它的总线使用请求得到批准并且总线又处于空闲状态时，那么该设备对 AD 线和 C/BE＃线的驱

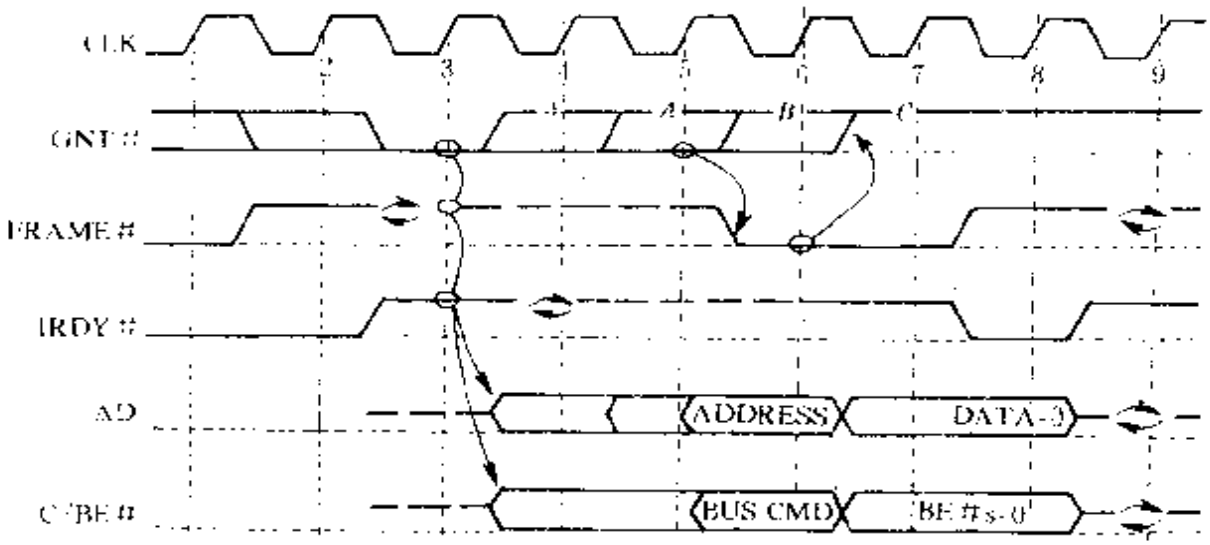


图 3.14　地址渐进的时序关系

动是允许的，也是必要的，但是它可以利用多个时钟周期来驱动使其达到有效值，然后再发 FRAME# 信号。不过，这种延迟建立 FRAME# 信号的做法，使得主设备有可能失去总线控制权。因此，对于任何主设备，GNT# 信号在 FRAME# 信号建立之前的一个时钟前沿处必须有效，如果此时(图 3.14 中标 A 的时钟边沿) GNT# 已被置为无效，仲裁器就会将总线批准给另外一个主设备使用，从而使当前主设备失掉总线控制权。从图中可明显看到，只要 GNT# 信号在 B 或 C 所对应的时钟边沿处撤消，那么 FRAME# 信号就能够建立，并且传输将继续进行。需要说明的是，如果当前主设备失去总线控制时，它应立即将 AD 线置为第三状态。

3.7.4　配置周期

PCI 的驱动软件提供了一个初始化的配置通道和一块独立的地址空间。为此，PCI 设备应具有 256 字节的配置寄存器，有关这些寄存器的说明见第六章。本节只讨论为访问 PCI 配置空间而设立的配置命令，即配置读命令和配置写命令。

3.7.4.1 配置访问

在前面的章节中我们已讨论过，正常的访问中，每个设备都对它自己的地址进行译码。然而，对于配置地址空间的访问，则要求采用另外的方法进行设备选择译码，即通过 IDSEL 引脚来选择 PCI 设备，也就是将 IDSEL 引脚作为"芯片选择"信号。对于某一 PCI 设备，只有当它的 IDSEL 引脚上有信号并且 AD[1：：0]在地址期内为 00 时，该设备才会被作为配置(读或写)命令的目标设备。64 个双字寄存器的内部寻址是通过 AD[7：：2]和字节使能信号来完成的。

配置命令跟其它总线命令一样，允许以字节、字、双字或者突发操作的方式进行访问；访问过程中的其它操作及终止方式也和一般的命令没有什么差异。如果没有设备响应配置命令，则应该由主设备废止。扩展总线桥绝对不能将配置命令传到扩展总线上去。

关于如何准确驱动 IDSEL 引脚的问题，一般由主桥或系统的设计者去决定。但是，在过去的设计中，该选择信号可以与高 21 位地址线中的其中一位相连，这是因为在配置访问中，高 21 位地址线没有别的用途。至于如何用高 21 位地址来决定 IDSEL，还没有现成的或标准的方法可遵循，所以，设备必须留有 IDSEL 引脚。换而言之，PCI 设备不能为了节省一个引脚而在内部将 IDSEL 信号连接到一条 AD 线上。唯独总线主桥可以例外，因为它可以决定如何映射 IDSEL。

AD[31：：00]在地址期中必须全部被驱动为有效状态，其中的每一条可连接一个设备，每次确立 AD[31：：11]当中的一条为 IDSEL 信号，从而可以从 21 个设备中选择一个设备进行配置读写。

上述利用高 21 位地址线来决定 IDSEL 的方法会使 AD 线上

增加一个负载，为此，可以将 IDSEL 利用电阻耦合到适当的 AD 上来解决负载加重这一问题。然而这样做，又带来了一个新的问题，那就是使得 IDSEL 的变化速率大为减慢，并且在大部分时间内处于无效的逻辑状态，如图 3.15 中的"XXXX"所示区段。

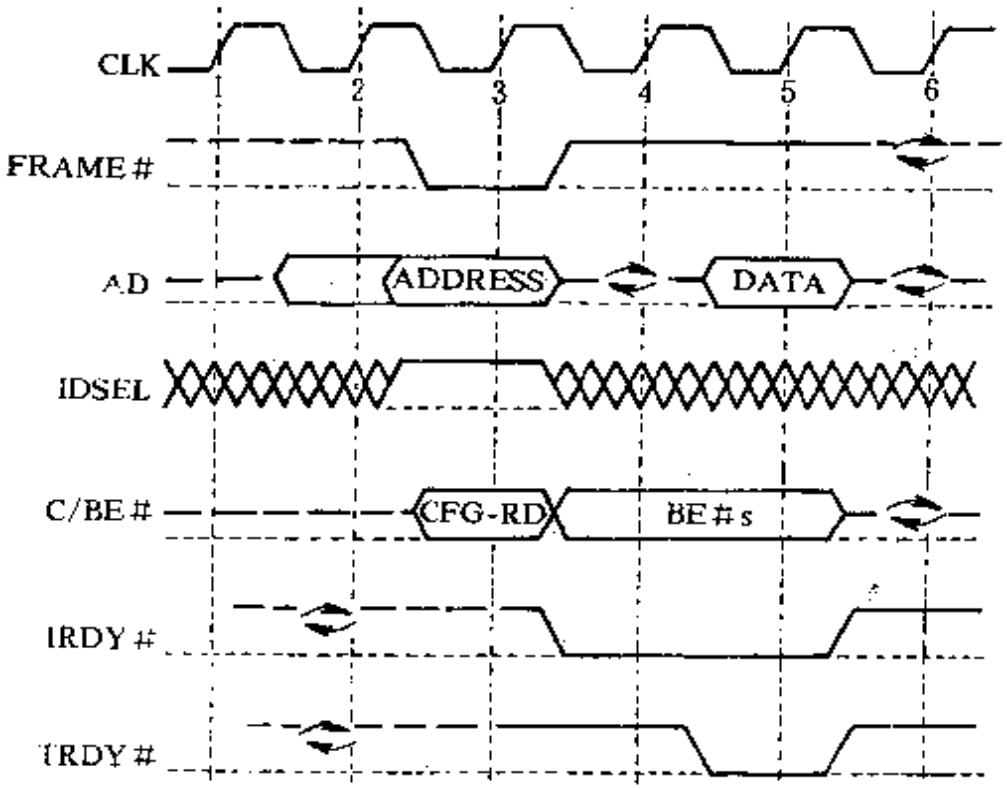
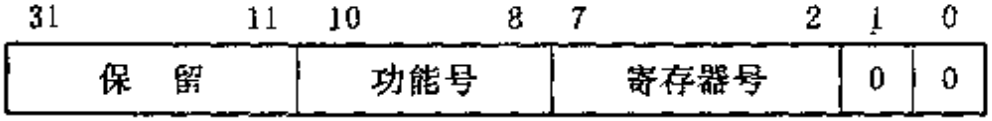


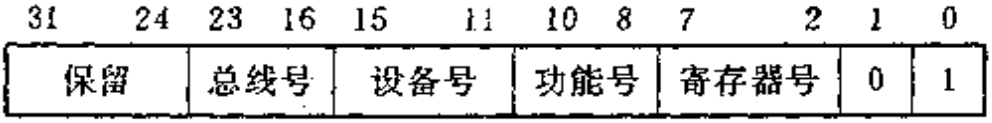
图 3.15 配置读操作时序

万幸的是，IDSEL 信号只有在一个配置命令的地址期内才会用到，所以可以在 FRAME# 建立之前的几个时钟周期内，将与 IDSEL 相连的地址线进行预先驱动，以保证 IDSEL 信号在需要被采样时是稳定有效的。地址总线的预先驱动方法与前述的地址/渐进相类似。

为了支持分层的 PCI 总线，采用了两种类型的配置访问，分别称为 0 类配置访问和 1 类配置访问，其格式如下：



0 类配置访问



1 类配置访问

从上面的格式表示中可以看出，它们的实质是表示了 AD 线在配置命令的地址期中的用法。两类访问的区别在于 AD[1：：0]线上的取值不同。对于 0 类配置访问，AD[1：：0]=00，用来在运行的 PCI 总线上选择一个设备；对于 1 类配置访问，AD[1：：0]=01，用来将一个配置请求传递到另一条 PCI 总线上。可见，在此处 AD[1：：0]相当于标志位。

两类配置访问中的寄存器编号和功能编号字段具有相同的含义；设备编号和总线编号字段只用于 1 类配置访问中；目标设备对保留字段应予以忽略，不能做出任何响应。

下面对上述各字段的具体含义做一说明：

寄存器号：选择目标设备配置空间中的一个双字(DWORD)。

功能号：用来选择多功能设备的某一个功能，有八种功能供选择。

设备号：用以在一条给定的总线上选择 32 个设备的其中之一。但是如果采用 IDSEL 与一条 AD 线相连，只能选择到 21 个设备。

总线号：用以在一个系统中从 256 条总线中选择一条。

无论是主桥还是 PCI 到 PCI 的桥，如果需要产生一个 0 类配置访问时，要用设备号决定建立哪一条 IDSEL 信号；在 AD[10：：08]上提供功能号；在 AD[7：：2]上给出寄存器号；AD[1：：0]此时必须为 00 以表示为 0 类配置访问。0 类配置访问不

会传播到本地 PCI 总线以外的总线上去，因此，必须要有一个本地设备对其作出响应，否则，主设备就会将该次访问废止。

倘若配置访问的目标不在本地总线上，而是在另一条 PCI 总线上，那就必须用 1 类配置访问。也就是说，1 类访问只适用于 PCI 到 PCI 的桥，其它所有目标设备都将其忽略，不作出任何响应。这些桥对总线号字段进行译码，以判断配置访问的目标是否驻留在自己的桥路之后。如果总线号不是指向自己桥路后面的一条总线，就将它忽略；如果访问的目标是本桥路后面的一条总线，此桥就要接受这次配置访问，其具体的响应又分为两种情况：① 如果总线编号与该桥路所带总线相符，此桥就要把这个 1 类访问变换成 0 类配置访问，也就是将 AD[1：：0]由 01 转变为 00，而 AD[10：：02]保持原值不变，同时对设备号进行译码从而选择一个 IDSEL 在相应的线上发出，开始一次配置访问；② 如果总线编号与该桥路所带总线不相符，此桥就会将配置访问直接向下一级传递。

有两种类型的设备可以响应 0 类配置访问：其一为单功能设备，是为了做到向后兼容而设定的，响应于否只需要利用它的 IDSEL 引脚和 AD[1：：0]＝00 来决定；其二为多功能设备，能够接受功能编号字段，需要利用它的 IDSEL 引脚、AD[1：：0]＝00 及 AD[10：：08]的编码值这三个条件来决定是否响应相应的配置访问。这两类设备的区别在于它们的配置空间的头标区的编码值不同。

对于多功能设备，必须能对 AD[10：：08]进行完全译码，以选择相应的功能，而且，只有那些在配置空间寄存器组中实现的功能才能得到响应。在它们所实现的功能中，必须具有 0 号功能，其它的功能都是可选的，并且功能编号可任意选择。例如，有一个 2 功能设备，那它必须有 0 号功能，而第二个功能的编号可以是 1～7 当中的任何一个号码。

配置编码(程序)将会按照设备号的顺序在总线上进行探测。如果得知某一设备是单功能设备，就不会对该设备再作进一步探测；如果发现是一多功能设备，将会对其所有的功能进行检查。

3.7.4.2 配置周期的产生

在系统中必须提供一个可以由软件产生 PCI 配置周期的机制，通常将这种机制置于主桥路当中。PCI 总线规范只对 PC—AT 兼容的系统规定了产生配置周期的机制，而对其它系统未作规定。

对于 PC—AT 兼容机，PCI 总线规范中定义了两种截然不同的机制，以允许软件产生配置周期，这两种机制称为配置机制 #1 和配置机制 #2。下面分别予以叙述。

对于配置机制 #1 要求优先考虑并实现之。将来所有的主桥都必须实现配置机制 #1，现有的桥如果有可能的话，也应该设法转变以实现它；配置机制 #2 的设立是为了做到向后兼容，因此，新的设计中绝对不能采用它。凡是在 PC—AT 兼容系统中使用的主桥，至少应实现这两种机制的其中之一。

一、配置机制 #1

在配置机制 #1 中，使用了两个 DWORD(双字)的 I/O 单元。第一单元(CF8H)表示一个可读写的寄存器，叫做 CONFIG—ADDRESS；第二个单元(CFCH)称为 CONFIG—DATA 寄存器。

访问配置空间的一般机制是：在 CONFIG—ADDRESS 中写入一个值，用以指定总线编号、设备编号、配置寄存器编号，这样一来，凡是对 CONFIG—DATA 寄存器的读写操作，都会使得桥路把 CONFIG—ADDRESS 的值转变为 PCI 总线上所要求的配置访问周期。

CONFIG—ADDRESS 寄存器为 32 位，其格式如下：

31	30 24	23 16	15 11	10 8	7 2	1	0
使能位	保留	总线号	设备号	功能号	寄存器号	0	0

其中，位 1 和位 0 两位是只读位，读出值必须为 00；位 7 到位 2 用以在设备的配置空间中选择一个 DWORD；位 10 到 位 8 用来选择一多功能设备中一个指定的功能；位 15 到位 11 用来在指定的总线上选择一个指定的设备；位 23 到位 16 用以在系统中选择一个指定的总线；位 30 到位 24 为保留位，只能读且读出值必须为全 0；最高位是使能位，用来决定何时应当把对 CONFIG—DATA 寄存器的访问转变成 PCI 总线上的配置访问。当该位为"1"时，表示允许，反之表示不允许。

任何时候，只要主桥发现一个对 CONFIG—ADDRESS 全部 DWORD 的 I/O 写命令，该桥就必须把数据存入它的 CONFIG—ADDRES 寄存器。如果主桥发现一个对 CONFIG—ADDRES 寄存器整个 DWORD 的 I/O 读命令时，它就必须送回 CONFIG—ADDRES 寄存器中的数据。对于此地址的其它任何类型的访问(即非 DWORD)，主桥必须将其按一次正常的 I/O 访问来处理，而不能产生特殊的响应。

CONFIG—ADDRES 寄存器在 I/O 空间耗费的仅仅是给定地址上的一个 DWORD(双字)，使用字节或字寄存器的设备不会受到影响，因为这些内容将会原封不动地被传递。

当一个桥发现一个 I/O 访问落入 CONFIG—DATA 地址开始的 DWORD 中，它就要检查 CONFIG—ADDRESS 寄存器中的使能标志位和总线编号字段，如果使能标志位为 1，同时总线编号与该桥路或任何在该桥路之后的总线编号相符合时，则该桥路就必须进行一次配置周期的转换。

配置周期的转换有两种类型可采用：第一种为 0 类，此类转换用在被寻址的设备位于桥路所连接的 PCI 总线上的情况下；第二种为 1 类，当设备位于桥路后面的另一条总线上时采用该类

转换。

图 3.16 表示了 0 类配置周期在桥路上的转换情况。桥路对设备编号字段进行译码并在适当的 IDSEL 线上建立信号，从而在 PCI 总线上完成一次配置访问。从图中可以看出发送到 PCI 总线的 AD 线上各部分信息的转换关系，其中 AD[10：：2]直接取自 CONFIG—ADDRESS 寄存器的相应位，AD[1：：0]=00。

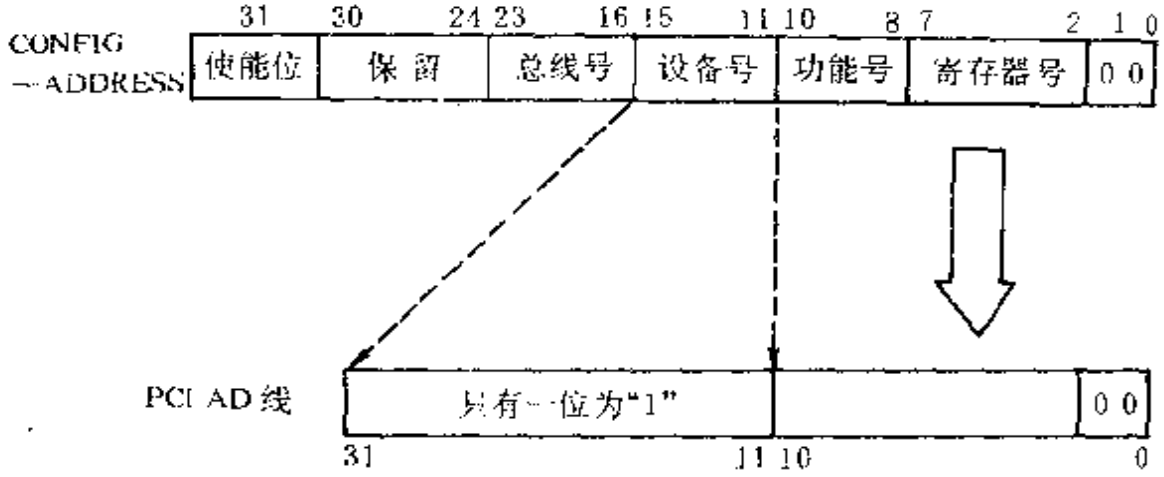


图 3.16　0 类配置周期的转换

对于 1 类配置周期的转换，其具体过程是桥路在一个配置命令的地址期中，直接将 CONFIG—ADDRESS 寄存器的内容复制到 PCI 总线的 AD 线上，并且保证使 AD[1：：0]=01。

在上述两种转换情况下，配合数据传输的字节使能信号必须从处理机总线上直接复制得到。

对于在处理机总线上具有与主桥同等功能桥路的系统，其中的同等桥路对 CONFIG—ADDRESS 的访问，一般被设计成以应答方式进行。而处理机总线上的其它桥路将监听对 CONFIG—ADDRESS 寄存器的写操作。至于对 CONFIG—DATA 寄存器的访问，一般由主桥在进行配置转换中以应答方式进行。

主桥和 PCI 到 PCI 的桥一般都要求有两个配置空间寄存器，其内容用以决定什么时候该桥路应进行配置周期的转换。一个寄

存器称为总线编号寄存器，用来指定本桥路后面直接相连的 PCI 总线的编号；另一个寄存器叫做从属总线编号寄存器，用它来指定桥路后面最后一级总线的编号。这些寄存器均由 POST 例程负责将它们置以适当的初始值。

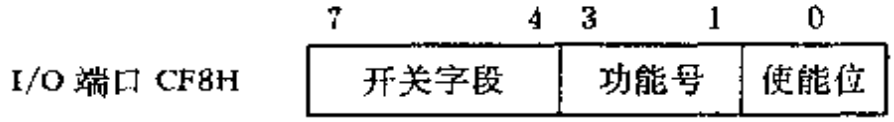
二、配置机制 #2

配置机制 #2 实际上是为 PCI 配置空间的访问提供了一种模式，在该模式中，把 PCI 的配置空间映射到 4 K 字节的 CPU I/O 空间。如果该模式处于允许状态，则对 C000H～CFFFH 范围内 I/O空间的任何 CPU 访问，都会变换成一个 PCI 配置周期。反之，当该模式设置为禁止状态时，对上述范围内 I/O 空间的所有 CPU 访问，都成为对系统中相应 I/O 端口的访问。此机制不支持处理机总线上的同等桥路。

在这种机制中，要用到两个寄存器，现分述如下：

1. 配置空间使能寄存器(CSE)

该寄存器位于 I/O 端口地址 CF8H 上，其作用是通过对该寄存器的写入来完成配置空间到 I/O 空间的映射。CSE 寄存器各位的含义如下：



可见，CSE 寄存器只有 8 位，因此，所有对该寄存器的访问必须是单一字节的操作。另外，该寄存器是一个可读可写的 I/O 端口，在逻辑上它驻留于主桥路。各位的功能是：0 位为使能位，当该位为 0 时，表示要产生配置周期，若将该位置 1，则表示要用配置机制产生特殊周期(详细说明见后)；位 1～位 3 是为配置周期提供功能号，它们在产生配置周期时被传送到 AD[10：：08]线上；位 3～位 7 叫开关字段，用来决定映射功能，可以把 I/O空间的读写变成 PCI 配置空间的读写。主桥对于读 CSE 寄存

器的响应是把最后一次写入该寄存器的数据返回。

系统复位之后，CSE 寄存器被清零，主桥进入缺省状态，此时它对所有的 I/O 访问都按正常方式处理。

2. 路径寄存器

该寄存器位于 I/O 地址 CFAH 处，用来指明访问哪一条 PCI 总线。它是一个可读可写的寄存器，且复位后的初始值为 0，也是一个 8 位的寄存器。读这个寄存器时，它返回的是上次写入的值。如果该寄存器为"00"时，则表明要访问的总线是与桥路直接相连的 PCI 总线，而且是 0 类配置访问；如果该路径寄存器的值为非零时，则形成 1 类配置访问，而且其值被映射到 AD[23：：16]线上，并在配置访问的地址期内送出。

三、配置空间的映射及配置周期的形成

下面介绍如何利用 I/O 地址、CSE 寄存器及路径寄存器进行配置空间的映射和两类配置周期的产生。

如果 CSE 寄存器的开关字段不为 0 时，说明允许桥路进行配置空间的映射，这时，桥路必须把 C000H～CFFFH 范围内的所有 I/O 访问转变为 PCI 配置周期。这种变换首先要利用 I/O 访问的 I/O 地址，很明显，由于是对 C000H～CFFFH 范围内 I/O 访问的变换，因此，I/O 地址的位 15～位 12 必然为"1100"。至于 I/O 地址的其余位，其用途是：位 11～位 8 用来对每个总线上的 16 个 PCI 设备进行寻址，也就是从 16 个设备中选择其中之一；位 7～位 2 用以选择设备配置空间中某一特定的 DWORD。

具体的变换又分为两种情况：其一是当路径寄存器为 0 时的情况，此时表明被访问的设备位于 PCI 总线编号为 0 的总线上，也就是说，该总线直接连在桥后，这时的变换结果是形成一个 0 类的配置访问，如图 3.17 所示。

其二为路径寄存器不为 0 的情况，这时说明所要访问的设备

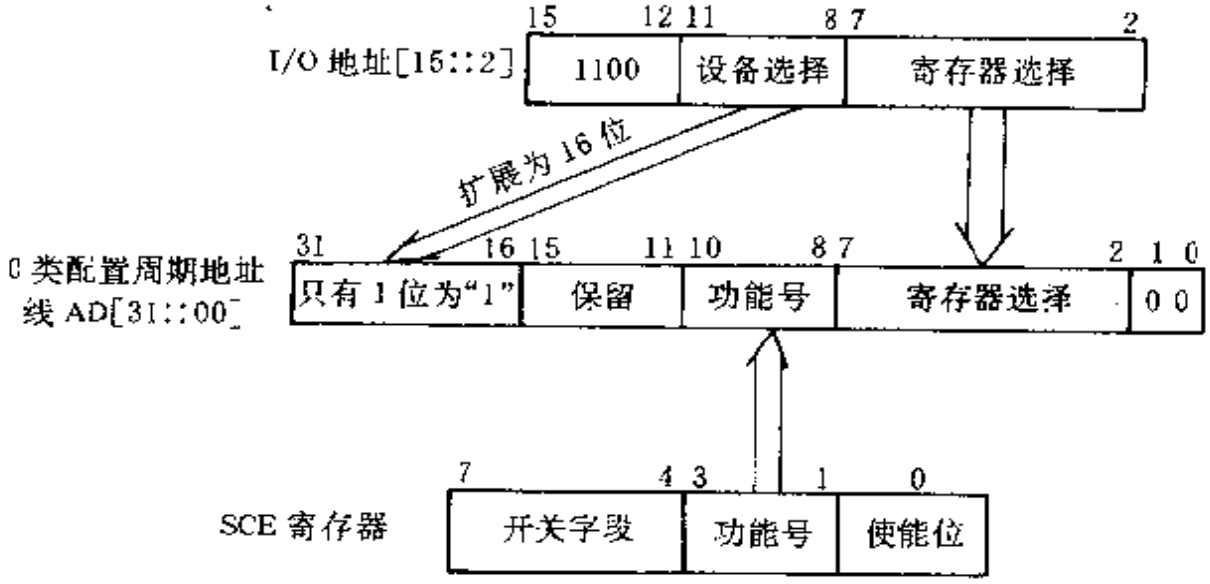


图 3.17 I/O 访问变换到 0 类配置周期

不在与桥路直接相连的 PCI 总线上，此时，该桥路应产生一个 1 类配置周期，并将路径寄存器的值映射到 PCI 总线的 AD[23：：16]线上，如图 3.18 所示。

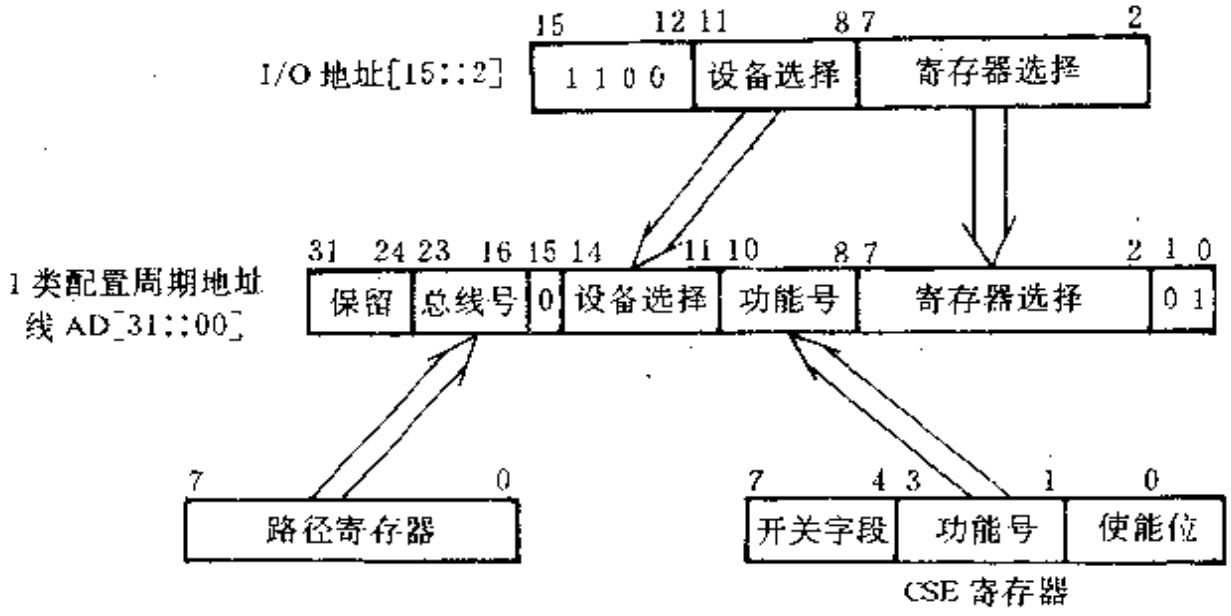


图 3.18 I/O 访问变换到 1 类配置周期

四、利用配置机制形成特殊周期

这部分内容主要说明对于具有配置机制#1和配置机制#2的主桥如何实现软件产生特殊周期，而且不需要为此目的提供一个机构给主桥。

在具有配置机制#1的主桥上，当CONFIG—ADDRESS寄存器中的总线编号等于桥路的总线号，设备编号字段和功能编号字段均为全1且寄存器编号字段为全0时，则在下一次对CONFIG—DATA寄存器进行写操作时便会形成一个特殊周期。此时，当写CONFIG—DATA寄存器时，桥路就不会发出配置写命令，而是在C/BE[3：：0]#上发出一个特殊周期命令，并在第一个数据期将I/O写数据放到AD[31：：00]线上。在这种情况下，对CONFIG--DATA的读操作结果是不确定的。

假如CONFIG—ADDRESS的总线编号和桥路的总线编号不相符时，该桥路便把对CONFIG--DATA寄存器的写命令当作1类配置周期通过PCI总线向下传递。

对于具有配置机制#2的主桥，当CSE寄存器的位0置为"1"，并且该寄存器的功能编号字段为全"1"、开关字段为非"0"时，则允许该桥路在下一次遇到CPU的I/O写访问地址为CF00H的情况下，形成一个特殊周期或者一个1类配置周期。

此时，读者不禁要问，如何判断是特殊周期还是1类配置周期呢？方法是：在上述条件满足的情况下，桥路还要检查路径寄存器的内容。如果该寄存器为全0，则主桥路在PCI总线上发出的是特殊周期命令，并在地址期内将特殊周期命令代码放到C/BE[3：：0]#线上，在第一数据期把对地址CF00H的I/O写数据放到AD[31：：00]线上；如果路径寄存器不是全0，则主桥路在PCI总线上发出的是1类配置周期，其中的设备编号字段和功能号字段(AD[15：：08])全为"1"，寄存器编号(AD[7：：2])

为 00。

在上述情况下，对地址 CXXXH 的读访问将会产生不确定的读出结果。

3.7.5 快速的背对背传输

对于一个主设备，可以采用两种方式进行快速的背对背传输，一种是访问同一设备，另一种是访问不同设备。但是不论是哪一种快速的背对背传输，只有在 TRDY #、DEVSEL # 或者 STOP # 不发生竞争的前提下才可以进行。

第一种快速的背对背传输允许将避免竞争的负担加在主设备上，而第二种则是将这种负担加于所有潜在的目标设备上。如果主设备能够保证竞争不会发生，就可以将两次传输之间的空闲周期取消，具体地讲就是，如果主设备的第二次传输与第一次传输的目标是同一个，并且第一次传输又是一个写操作的情况下，便可取消两次传输之间的空闲周期。

在第二种快速背对背传输当中，要求主设备应知道潜在目标的地址边界，否则可能发生竞争。对于主设备可以将这种快速背对背作为选项，但目标设备必须对这种快速背对背进行译码。

由于第二种快速背对背允许将避免竞争的任务放在所有潜在的目标设备上完成，因此状态寄存器当中的快速背对背允许位可以接在高电平上，但是一个总线目标设备要这样做必须满足下列要求：

(1) 目标设备保证既不会丢掉一个总线传输的开始部分，也不会丢掉地址，哪怕是传输开始之前没有一个空闲周期时也应如此。也就是说，在连续的时钟周期内，目标设备要能够跟上总线状态从一个最终数据传输(FRAME # 为高，IRDY # 为低)直接变到一个地址期(FRAME # 为低，IRDY # 为高)这样一种快速的跳变。

(2) 目标设备必须能够避免 DEVSEL#、TRDY#和 STOP#信号的冲突。如果目标设备不实现最快的 DEVSEL#响应时间，就可以保证这一点。而对于那些实现 0 等待译码的目标设备，除了下列两种情况之外，必须将上述三个信号延迟一个时钟周期再发送。

① 当前的传输紧跟在一个总线空闲周期之后，这时就不是一个背对背传输。

② 当前的目标设备已经在前次传输中设置了 DEVSEL#信号，此时，尽管属于背对背传输，但所涉及的目标设备与前次相同。

对于背对背传输需要目标机构支持才能完成的主设备，就要使用命令寄存器中的快速背对使能位(该位仅对作为总线主设备的设备有意义，且是一个可选项)。该位可读可写，如果这个使能位为"1"时，只要前一次传输是由当前总线主设备发动的写操作，那么该主设备就可以利用快速背对背的时序关系启动一次 PCI 传输，而不必考虑哪个目标设备被寻址。反之，如果这个使能位为"0"或者没有实现，那么只有在主节点能够确保新的传输所对应的目标设备与前一次相同的情况下，才可以进行快速背对背传输。该位的设置也许要由系统的 POST 例程来完成，但在设定时应确保同一条总线上的所有目标设备的快速背对背使能位均被设置。

PCI 总线之所以提供快速背对背传输功能，是由于这种配置中将 PCI 总线作为处理机到主存的通路，从而使得在"低档"系统配置中可实现性能方面的改进。尤其是这种功能的实现代价几乎可忽略不计，所以建议所有新出的目标设备或者准备用于这种配置的设备都应实现这个功能。但是，对于现有的设备，没有必要仅为此功能而进行重新设计，这是因为如此做法对现有应用可能不会带来多少好处。

在其它所有条件下，主设备必须至少插入一个总线空闲周期，即就是在不同的主设备传输之间也要保证至少有一个总线空闲周期。值得注意的是，对于多端口目标设备，当它在快速背对背传输中被锁住时，只能锁定它本身。

有一点是重要的，在快速背对背传输过程当中其设备未被涉及的节点，不能也不需要利用 FRAME＃和 IRDY＃信号来区分中间传输边界。只有在快速背对背传输期间涉及到的主设备和目标设备才需要区分这些边界。

在一次快速背对背传输过程中，主设备可以直接启动下一次传输而不用插入一个总线空闲周期，当 FRAME＃信号撤消而 IRDY＃和 TRDY＃信号都有效时，表明最后数据期已经完成。当前主设备可以在前一次传输的最后数据传输期开始之时，启动下一次传输。如图 3.19 所示。

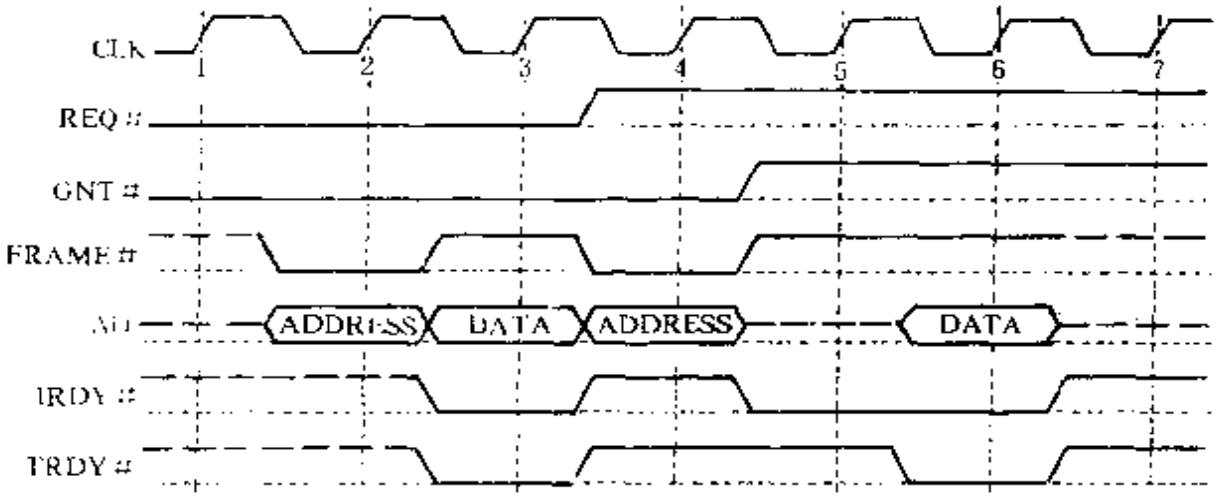


图 3.19　背对背访问的仲裁

图 3.19 中，主设备在 CLK3 处完成一个写操作，并且下一次传输的地址期紧接着在 CLK4 上出现。目标设备必须在当前传输完成之后的下一个时钟上采样 FRAME＃信号。尽管背对背操作对主设备来讲可以是选项，但目标设备对此操作必须能够译码，即使支持第二种背对背机制的目标设备也必须能够区分全部 PCI

传输的完成，并能检测到所有的地址期。如果一个目标设备通过设置 DEVSEL＃信号而确立了从属关系后，它可以对传输请求发出再试响应。

3.8 纠错功能

PCI 总线为奇偶位和其它系统错误提供了检测和报告措施。同时，这种纠错功能也覆盖了从不关心错误的设备的检测、标识和恢复，从而使得此类设备可以从奇偶错误中恢复出来，不致于影响它们的操作，因此，所有的目标设备都应能在各种传输中产生奇偶位，以满足检测、标识和恢复的灵活性。

3.8.1 奇偶校验

PCI 总线的奇偶校验提供了一种机制来决定一件事务，如果数据在它们之间正确传输，则应保证执行正确的总线操作。在 PCI 总线上，利用奇偶校验在每次传输中检查主设备是否成功地寻址到所希望的目标，以及它们之间的数据传输是否正确。为保证总线命令的正常执行和数据传输的正确性，AD[31：：00]及 C/BE[3：：0]＃各线均包括在奇偶计算之列。同时，在任何总线周期，承担驱动 AD 线的设备还应负责计算出奇偶位并驱动 PAR 线。

只要是地址期和数据期，无论 AD[31：：00]和 C/BE[3：：0]＃各线上是否携带有效信息，奇偶位都将包含它们。但是，对于那些无意义或者说实际并不传送数据的字节所对应的线，必须被驱动到稳定状态，并包含于奇偶计算之中。而在配置读写、特殊周期和中断响应命令的执行过程中，地址线的部分或者全部也是无定义的，同样，它们也要被驱动到稳定值并包含于奇偶计算之中。

总之，奇偶位要根据以下原则来产生：

(1) 奇偶校验与传输的类型或形式无关，也就是说在所有的PCI事务中均采用同样的计算方法。

(2) 在AD[31：：00]、C/BE[3：：0]#及PAR各线上，“1”的个数为偶数。

(3) 奇偶校验的产生不是可选项，它必须由所有的PCI从属设备完成。

在任何给定的总线周期内，哪个设备驱动了AD[31：：00]线，它就必须驱动PAR线，而且在时间上要比相应的地址或数据推迟一个时钟周期。图3.20表示了一个读写操作过程中奇偶校验的情况。

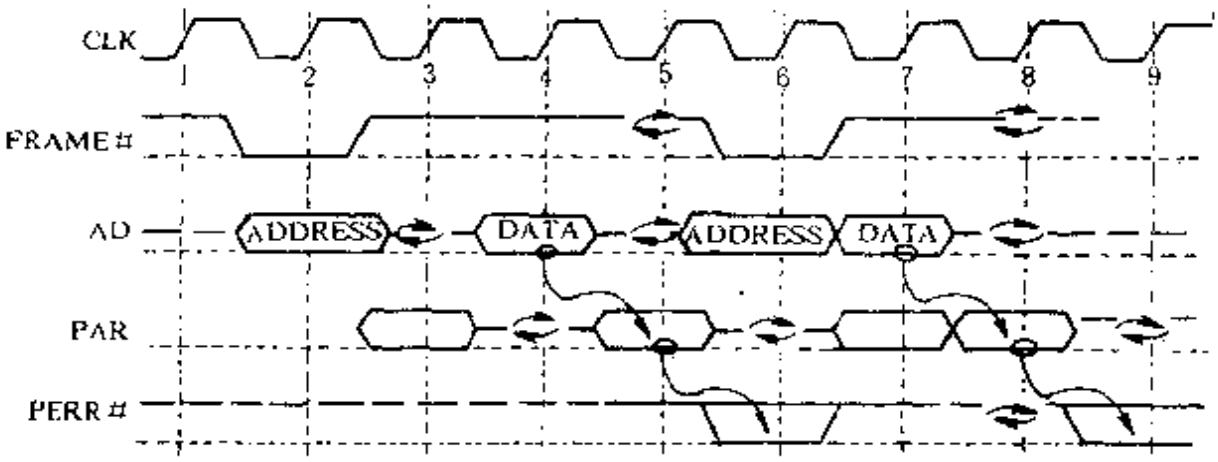


图3.20 读写操作下的奇偶校验

可以看出，主设备在时钟3和时钟7处分别为地址期而驱动了PAR线，目标设备(从设备)在时钟5处为读操作中的数据期而驱动了PAR线，接着主设备又在时钟8处为写操作的数据期驱动了PAR线。应注意的是，PAR除拖后一个时钟节拍外，其它的行为完全和AD[31：：00]相同，包括等待周期和过渡周期。

奇偶校验主要用来确定主设备是否成功地寻址到它所希望的目标设备，以及数据传输的正确于否。在PCI总线上奇偶校验的检测是必须的，但下列两类设备可以例外：

(1) 专门设计用在母板上的设备，例如芯片组。因为系统供应商对这类设备的使用具有控制权，也就是说这类设备决不会出现在接插板上。

(2) 从不涉及、包含或访问任何表示永久性、驻留系统或应用状态的数据的设备。由于这类设备只涉及暂时性的数据，而不是永久性的、驻留系统的或者应用状态之数据，所以不会因为没有检测出来的错误而引起系统完整性问题。

对于那些支持奇偶校验的设备来说，只要它检测到一个奇偶错误时，就必须把配置空间的状态寄存器的"检测奇偶错"位置成"1"。除此之外的其它操作，要根据配置空间的命令寄存器的"奇偶错误响应"位来决定。具体如何决定将在下一小节说明。

最后应特别注意一点，任何设备都可以检查出奇偶错误并在SERR＃线上进行标记，但只有主设备可以报告读数据时的奇偶错误，而不能报告写数据时的奇偶错误；反之，被选中的目标设备只能报告写数据时的奇偶错误，而不能报告读数据时的奇偶错误，这也是显而易见的。

3.8.2 错误的报告

如上文所述，PCI 提供了对奇偶错误及其它系统错误的检测和报告。应当强调指出，奇偶错误要报告到访问和设备驱动程序链上尽可能多的地方。这个错误报告链可形象地表示为：目标设备→主设备→设备驱动程序→设备管理程序→操作系统。也就是说，尽最大可能将错误向上级报告，以便在任何一级实现错误状态的恢复。对于系统错误，一般情况下，不容易将它和一个特定的访问链联系起来，因此，一旦该类错误发生，应直接报告给系统级。

在 PCI 总线中使用两个信号(引脚)来进行错误的报告。其中之一为 PERR＃线，它专门用来报告除特殊周期命令之外其它所

有传输中发生的数据奇偶校验错误，是一个只允许单一设备驱动的低电平三态信号，并且为所有的 PCI 设备共用。总线协议保证 PERR＃信号不可能被多个总线设备同时驱动，同时，也保证各个设备都遵守适当的信号过渡时间规定，从而避免发生任何驱动冲突。

另一根线是 SERR＃，用来报告其它错误，包括地址奇偶错误和特殊周期命令中的数据奇偶错误，同时也可以用于报告其它非奇偶错误或者系统错误。它是 OD 形式信号，所有的 PCI 设备都可以线或方式接入，所以此线有可能被多个设备驱动。这里应该注意，对于 OD 信号不能保证在每个时钟前沿都具有稳定的状态，因而，一旦 SERR＃线上建立了它的稳定逻辑值，就应该认为它是不间断的，只有在连续两个以上的时钟前沿处采样发现为无效状态时，才能认为它被撤消。

由于奇偶错误的报告在 PCI 总线上是必须的，因此，不管是 SEER＃还是 PERR＃引脚自然也是必须的。但是对于 3.8.1 节中所述的两类设备可以例外。

注意：所有的设备都应该能够毫不例外的产生奇偶校验；用 SERR＃报告非奇偶性的错误是可选项；在 SERR＃上发信号将产生一个不可屏蔽中断 NMI，这将是致命的错误，所以对 SERR＃信号线的使用应当谨慎才是。

3.8.2.1 在 PERR＃线上对奇偶错误的报告及其对它的响应

现在来说明对于发生于除特殊周期命令之外的一切总线操作中的数据奇偶错误，是如何进行报告并予以正确响应的。对于全部的地址奇偶错误及特殊周期命令中的数据奇偶错误都利用 SERR＃线来反映，这将在稍后进行说明。下面叙述中所有提到的奇偶错误均指数据奇偶错，并且排除特殊周期命令中的数据奇

偶错。

在 PCI 总线上利用 PERR＃引脚来标记或者反映一个发生在两个设备之间的数据奇偶错误。只有其传输数据遭到破坏的主设备才可以不采用 PERR＃线向软件报告奇偶错误。目标设备总是通过 PERR＃引脚将数据奇偶错误报告给主设备，这样就给发起访问的设备在各个硬件或软件层次上进行恢复的特权。

除了设置“奇偶错误检测”位之外，所有关于奇偶错误的报告和响应都由“奇偶错误响应”位来控制，除了 3.8.1 节中所述的两类设备外，其它一切设备都应具有此位。如果该位为“0”，设备对所有的奇偶错误都将不予理睬，而是像出错前一样完成传输；反之，当该位为“1”时，一旦出现奇偶错误，设备就必须设置 PERR＃线以进行报告。在任何情况下，“奇偶错误检测”位必须设置为“1”。至于对错误的响应往往跟具体设备有关，各不相同。

一个设备总是在发生错误的数据传输之后的两个 PCI 时钟内发出 PERR＃信号，具体参看图 3.20。对于接收数据的设备，当它发现奇偶错误时，可以发出 PERR＃信号，也可以不发 PERR＃信号，但是，一旦它发出了 PERR＃信号，就必须保持其有效状态直到实际传输之后两个时钟周期。一个主设备在任何时候只要发现 PERR＃线上有信号，就得知有数据奇偶错误发生，但要确定数据传输是否有错，还要等到传输之后两个时钟时才能得知。

在多次数据传输且未插入等待周期的情况下，PERR＃信号将在多个连续的时钟上被采样，为了与之对应，PERR＃信号有可能在其中任何一个时钟上或每个时钟上都有效。由于 PERR＃是一个一次只有一个设备驱动的低电平三态信号，因此，它必须在每个认可时钟边沿上被有源地驱动到正确的值。为了在每次总线操作结束时返回到正常状态，PERR＃必须在 AD 总线过渡周期之后两个时钟内被驱到高电平，并保持一个时钟周期，如图 3.20 中的时钟 7 处所示。PERR＃的过渡周期发生在一个时钟周期之

后，如图 3.20 的时钟 8。另外，一次传输中产生的 PERR＃要等到地址期之后至少三个时钟时才会出现。

如果主设备检测出一个数据奇偶错误，并且在读操作中发出 PERR＃信号或者在写操作中采样到 PERR＃信号时，它必须设置“数据奇偶错误报告”位(状态寄存器的位 8)，至于接下来的传输过程可以继续，也可以终止。对于目标设备，如果它在传输中检测到数据奇偶错误，就不能对“数据奇偶错误报告”位进行设置，但传输过程同样也可以继续进行或者终止。这里建议，无论主设备还是目标设备检测到奇偶错误(PERR＃有效)，最好使它们继续将传输完成。PERR＃对目标设备来说只是一个输出信号，而对于主设备却是一个输入/输出信号。

如果主设备在其传输中发现有数据奇偶错误出现，就必须通知给系统。这里建议采用由主设备发一个中断、修改一个状态寄存器或者设置一个标志的方法，去通知它的设备驱动程序。如果设备上没有这些措施，那么它只有通过发 SERR＃信号将错误处理的责任推给操作系统。

注意：系统设计者可以在中央资源中把 PERR＃转变为 SERR＃信号，从而将所有的奇偶错误报告给操作系统。

3.8.2.2 在 SERR＃线上进行错误报告及其响应

SERR＃信号用于反映全部的地址奇偶错误、特殊周期命令中的数据奇偶错误以及除奇偶错误之外的其它所有错误。

无论是主设备还是目标设备，它们都可以检测地址奇偶错误并在 SERR＃线上进行报告。不管是何种类型的错误，只有在命令寄存器中的 SERR 使能位为高电平时，才能发出 SERR＃信号。同时在发出该信号时，相应的设备必须将配置空间状态寄存器的“发出系统错误”位设置为“1”。此外，如果是奇偶错误，那么在任何情况下“奇偶错误检测”位必须设置为“1”，但错误在 SERR＃线

上的报告，要根据命令寄存器中的“奇偶错误响应”位而定。

一个被选中的设备如果检测到地址奇偶错误时，可按如下方法处理：① 可以承认这个传输并按正确地址情况终止；② 承认这个传输并以目标废止方式来结束传输；③ 不承认这个传输而让主设备废止。应当注意，由于已经检测出一个地址奇偶错误，目标设备不能利用再试或断开方式来终止传输。

在任何 PCI 传输中，SERR＃信号的发出都没有时间关系，但是应该尽快地将错误反映出来，最好是在检测到错误之后的两个时钟之内实现。只有中央资源对 SERR＃信号感兴趣并将它作为输入信号，同时将它变成一个低脉冲送给处理机。至于中央资源如何把信号发往处理机要由系统来决定，但一般可采用产生一个 NMI 中断、高优先级中断、设置一个状态位或标志这些方法的其中之一。然而，发 SERR＃信号的设备肯定希望中央资源产生一个 NMI 中断，否则，它可以采用其它方式(如状态位、标志或中断)进行报告。

当“奇偶错误响应”位为“1”，并且“SERR 使能”位也为“1”时，一个设备在以下条件下可以发 SERR＃信号。

(1) 检测到地址奇偶错误或者特殊周期中的数据奇偶错误。

(2) 当前主设备检测到一个未被其它机构报告的奇偶错误。

只要“SERR 使能”位为“1”时，一个设备可在下列条件下发 SERR＃信号：

(1) 没有驱动程序的主设备被一个不正常结束的传输所涉及。

(2) 一个灾难性的错误使得设备能否正常工作成了问题。

注意：对于执行配置命令及特殊周期命令的桥路，主设备废止不是非正常条件，SERR＃不能用于这种情况或者可以正常恢复的情况。由于发 SERR＃信号可能产生一个 NMI 中断，因此应当深思熟虑和特别小心。目标废止一般来说是一个非正常终止条

件，如果主设备无法通过它的设备驱动程序报告错误时，就可能采用 SERR＃来报告错误。

一般情况下，一个主桥应当有一个计数器，当它计满时，主桥就停止一个访问的再试。当一个访问以再试方式终止时，该计数器就进行计数，而每当主设备进行数据传输时，该计数器就被清除。这些不一定是必须的，但建议这样做，以保证不会因为一个访问的连续再试而妨碍处理机去处理一个代表错误条件的中断。

3.9 对 Cache 的支持

Cache 对共享内存的支持由 SDONE 和 SBO＃引脚来实现。它们在桥/缓存与存储请求的从设备之间传递 Cache 状态信息。桥/缓存监听 PCI 上的内存访问，并决定从设备要什么样的响应，以保证系统内存的一致性。

在低档机器或便携机系统中，系统存储器的部分或全部都可能在 PCI 总线上，这里包括 ROM 和 DRAM。它们对于处理机来说都必须是可缓存的。PCI 总线对高速缓存的支持是一个选项，其目的是为了在 PCI 存储设备与桥路或高速缓存设备之间提供一个标准的接口，以便允许监听高速缓存(Cache)一致性机制的使用。这个可选的 Cache 支持能力要求一个单级的桥路拓朴结构和一个平面地址空间(也就是一个地址对应唯一的目标，与访问源点无关)。

注意：这个支持是为简单的低档机系统而优化的，而不是为了最大限度的处理机/Cache/存储器的性能优化。

为了避免“监听过载”，对于经常访问的在配置上规定不可缓存的地址范围(如帧缓冲区)，可以在程序的控制下立即发出一个“清除监听”信号，以表示监听结果是“干净的”，或者说没有

Cache 冲突。

任何支持可缓存存储器的 PCI 设备，都必须监听 PCI Cache 支持信号并做出适当的响应。在配置上不支持 Cache 的从设备可忽略 SDONE 和 SBO# 信号，这样做可以节省一些访问延迟。

由于 PCI 总线允许无限长的突发传输，这就要求可缓存的目标设备必须能够断开在存储器范围内试图跨越 Cache 行边界的访问。这就是说，任何可缓存的目标设备要么在配置空间中实现缓存行长度寄存器，要么将此参数采用硬件方法固定下来，总之它必须知道缓存行的长度。如果允许一次突发传输跨越 Cache 行边界，则 Cache 的一致性问题可以拆开解决，也就是桥/缓存设备可以在监听过程中产生下一个 Cache 行地址以继续监听。

为了高效利用 PCI 总线，可缓存的存储器和桥/缓存设备都要对总线的操作进行跟踪。当只锁存一个地址时，有可能出现使一个可缓存的传输被延迟的情况。如果一个不可缓存的传输启动之时，正好有一个缓存设备准备监听一个地址，这时它便把传输的地址锁存起来并由 Cache 进行监听。所谓监听就是检查一下总线命令是否命中了本模块中一个修改过的 Cache 行。在回写式的 Cache 中，这个修改过的行是系统中目前唯一正常的副本，所以应当由本模块为这次总线读取提供数据，而不是由主存提供数据。但是由于当前这个传输是不可缓存的，其完成不受 SDONE 状态的影响。如果前一个监听还未完成时，就启动了下一个可缓存的传输，则该传输就要被再试，要不然它的地址就监听不到了。如果可缓存和不可缓存的传输交替进行时，可缓存的传输就可能无法进行。

为了减少可缓存传输的再试，就要求涉及可缓存的传输设备应该能够接受两个地址。这样，当第一个地址正在被监听之时，如果第二个地址出现在总线上，它也被锁存起来。一旦第一个地址的监听完成，第二个地址的监听就立即开始。要求锁存的地址

数最多为两个，因为第三个地址出现在总线上的时间不可能在第一个地址监听完成或第二个传输的终止之前；从另一角度来看，只有在前面的监听或第二个传输完成的情况下，Cache 和存储器控制器才准备好接受新的地址。

如果第二个对话是可缓存的，就要求存储器的控制器插入等待周期直到第一个监听完成。当第一个监听完成时，存储器的控制器继续工作处理对话。如果第二个传输对应一个不可缓存的地址，由于它不需监视 SDONE 和 SBO＃信号，其目标可以完成传输；如果第二个传输的目标发出 TRDY＃或 STOP＃的时间早于或同时于 SDONE 的发出时间，这就是一个不可缓存的传输，此时由于有 TRDY＃信号而不监听传输的地址，所以，任何情况下，只需要锁存两个地址就完全足够了。

3.9.1 Cache 状态的定义

PCI 总线规范中，定义了 SDONE 和 SBO＃两个信号用于在参与缓存协议的设备之间提供状态信息。当 SDONE 有效时，表示监听完成；当 SBO＃有效时，表示命中了一个修改过的行；当 SBO＃撤消而 SDONE 有效时，表示一个“干净的”监听结果。

在 PCI 总线上出现的 Cache 状态有三个。而 PCI 总线上的缓存支持信号 SDONE 和 SBO＃以如下的组合方式表示这三种状态：

STANDBY	0X
CLEAN	10
HITM	11

下面我们来讨论每种状态在被缓存桥(以下简称为缓存)驱动时所表示的含义，以及一个可缓存的存储器应当在控制器中如何解释它们。

1. STANDBY

此状态表示 Cache 已经处于三个可能的条件之一。第一个条

件是 Cache 当前并没有监听一个地址，但已经做好监听的准备。第二个条件是已经锁定了一个地址，Cache 也正在监听该地址，同时为第二个可能出现于总线上的地址的接受及锁定做好了准备。最后一个条件是 Cache 当前正在监听，而且已经锁定了第二个地址。在第一个监听完成时，如果 SDONE 和第二个地址仍然有效，便开始对第二个地址进行监听。至于 Cache 当前到底处在哪一种条件下，必须依靠存储器控制器跟踪 PCI 控制信号才能知道。

如果一个监听正在进行中，而存储器控制器是第二个传输的目标，它就必须插入等待周期直到第一个地址的监听完成才继续进行第二个传输。如果存储器控制器不是第二个地址的目标，它就必须监视总线信号以判断第二个地址是否被监听或丢弃。

2. CLEAN

此状态表示没有 Cache 冲突且存储器访问可以正常完成。这个也说明发生了以下三种情况之一：① Cache 未命中；② 在写传输中命中了一个未修改过的行；③ 在一个存储器写并无效命令中命中了一个修改过的行。以上几种情况都不需要回写。当 Cache 是当前总线主设备并且正在回写一个修改过的行时，它在地址期中将以 CLEAN 标记。

当锁定了两个地址后，Cache 有可能在连续两个时钟周期发 CLEAN 信号。第一个时钟周期 SDONE 有效表示第一个监听已经完成。如果第一个监听的结果是 CLEAN 并且 Cache 已经发起了第二个传输，那么它可以继续发 SDONE 信号以表示第二个地址的监听结果将是 CLEAN。此外，Cache 可在 CLEAN 之后发 STANDBY 以表示第二次监听正在进行，在这种情况下，STANDBY 在总线上出现的时间为 SDONE 有效后的一个时钟周期。

3. HITM

此状态表示监听命中一个修改过的行，并且 Cache 要求将这

个被监听到的行写回主存作为下一个操作。Cache 将停留在此状态上直到回写操作开始。当 HITM 在总线上出现时，其它所有可缓存的传输都将被存储器控制器以再试方式而终止。在回写一个修改过的行的期间，Cache 的状态在地址期内将从 HITM 转变为 CLEAN。所有的存储控制器在收到 HITM 信号之后都要以再试方式终止后续的可缓存传输，以便给回写让路，等到回写完成后再重新请求。

3.9.2 支持的状态转变序列

PCI 总线上，Cache 的状态转变序列有两种，分别是：

(1) STANDBY→CLEAN→[CLEAN]→STANDBY

(2) STANDBY→HITM→CLEAN→[CLEAN]→STANDBY

序列(1)是正常情况，Cache 一直处于 STANDBY 状态直到监听完成，然后发出 CLEAN 信号表示传输应该正常完成。当第一个监听完成时，如没有第二个地址等待处理的话，就回到 STANDBY 状态。如果已经锁定了第二个地址并且 Cache 不是主设备时，它也要转变到 STANDBY 状态，表示正在监听。如果 Cache 是第二个传输的主设备而且这个传输是回写一个修改过的行或者它知道监听结果是干净的，它就可以继续发 CLEAN，否则转变为 STANDBY 状态。

序列(2)是监听命中一个修改过的行的情形。一旦 Cache 确定为 HITM 状态，它就会持续此状态直到修改过的行被回写，此时，Cache 将变为 CLEAN 状态以表示回写正在进行。然后它将发 STANDBY 信号表示准备好监听一个新地址。如 Cache 是第二个传输的主设备，同时传输是回写 Cache 行或知道监听结果是 CLEAN 的情况下，可以继续续发 CLEAN，否则也转变到 STANDBY 状态。

3.9.3 时序关系

本节将对有关 Cache 操作中的一些时序关系加以讨论。在各时序图的 CLK1 上假定总线是处于空闲状态。

1. 插入等待状态直到监听完成

如图 3.21 所示，当一个地址在时钟 2 处被锁定时，一个传输便开始进行。目标设备在监听完成之前一直不发 TRDY # 信号，表示插入等待周期。在时钟 5 处采样到 SDONE 信号有效，说明监听已经完成，由于 SBO # 无效，表示这个监听结果是 CLEAN（干净的）。

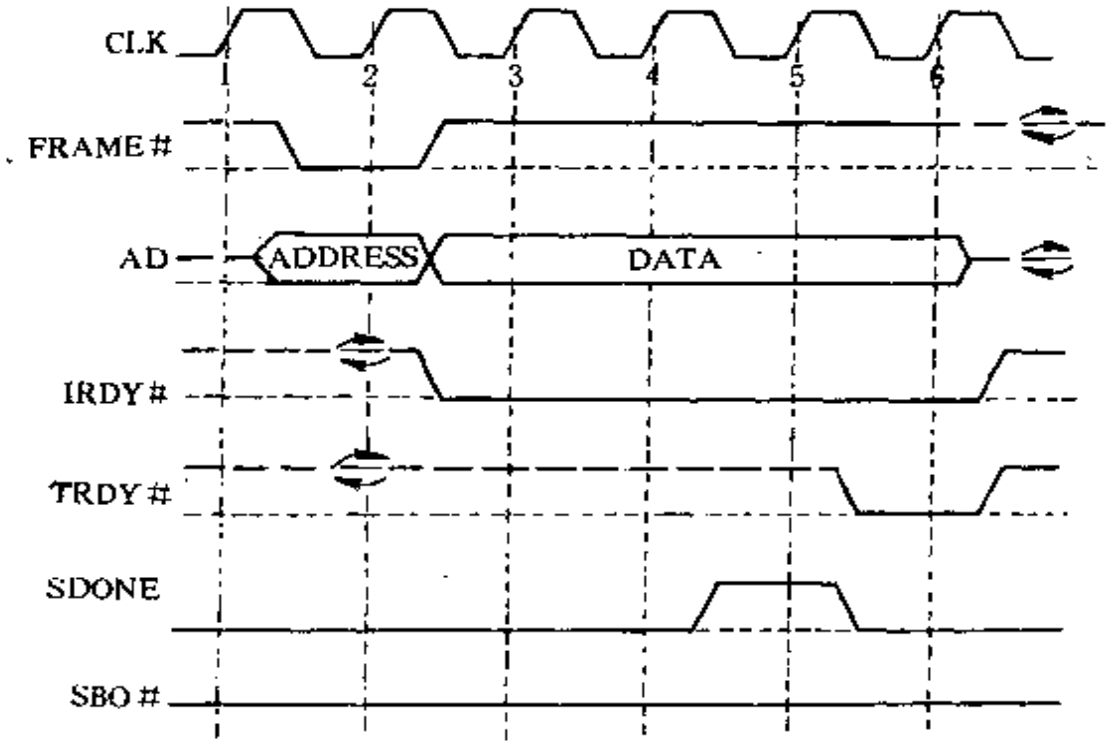


图 3.21 插入等待状态直到监听完成

2. 命中一个修改过的行及其回写

该操作的时序关系如图 3.22 所示。

同样地址在时钟 2 处被锁定，标志着第一个传输的开始。相应的目标设备插入等待周期直到 SDONE 有信号时为止。该例中，

Cache 在时钟 4 上使得 SDONE 和 SBO# 信号同时有效，表示监听命中了一个修改过的行。由于传输中的目标是可缓存的，所以它在时钟 5 上发出 STOP# 信号终止传输，以允许发 HITM 的设备能够把修改过的行回写到存储器中。对于读传输，在发出 HITM 期间，存储器控制器必须将 AD 线置为三态。当 HITM 信号出现在总线上时，所有对可缓存目标设备的传输都会被它以再试方式而终止。

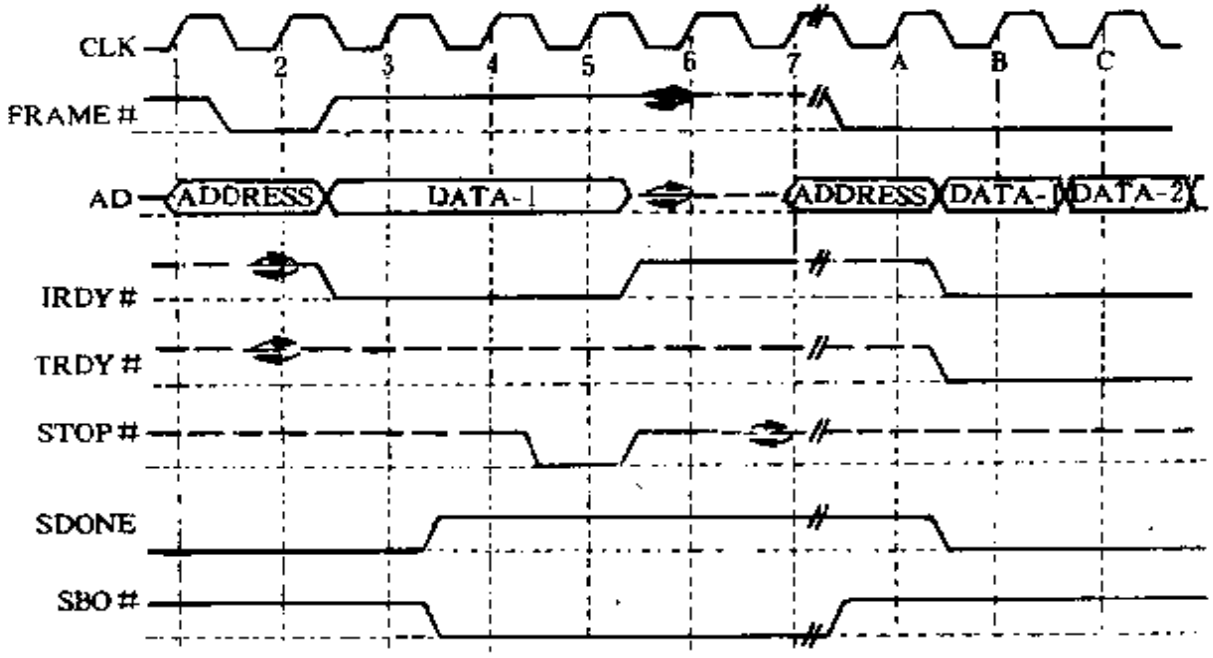


图 3.22　命中一个修改过的行及其回写

图 3.22 中的双斜杠线表示在第一个传输中自监听信号被确立以来，已经过了一段时间。在这段时间内，不可缓存的传输可以完成，而可缓存的传输由于 HITM 的确立在开始以后必须以再试而告终。回写的传输开始于时钟 A 处。从图中可以看出，在地址期中，Cache 的状态从 HITM 转变为 CLEAN 状态，这就告知存储器控制器，监听的回写已开始，要求它接受整个一行数据。

如果存储器不能完成此传输(只发生于可缓存的目标设备有一个内部冲突的情况)，它就必须插入等待周期直到能够完成为

止。即使目标是锁住的，它也必须能够接受由于对修改行的监听而引起的回写，否则，就会造成死锁。

在回写期间，无论是 Cache 还是存储器控制器均可插入等待周期。对于存储器控制器要求它能够在一次传输中接受整个一行；而 Cache 设备将提供整个一行。

在时钟 B 上，Cache 的状态又从 CLEAN 转变为 STANDBY，表示 Cache 已经为监听另一个地址做好了准备。一旦回写完成，总线又回到正常操作，又可进行可缓存的传输。回写的顺序与引起回写的传输无关。

3. 存储器写并无效命令

图 3.23 表示了一个存储器写并无效命令的情况，它仅作为一个例子。关于 Cache 对此命令的处理，有几种方法可供选择。由于主设备可以保证高速缓存行中每个字节都能被修改，所以，即就使命中了一个修改过的行，Cache 也可以简单地发出 CLEAN 信号。在本例中，Cache 在时钟 5 发 CLEAN 表示要么不命中，要么命中一个修改过并变成无效状态的行。一旦 Cache 变成 CLEAN 状态，它就准备好监听下一个地址。所以，对于 Cache 要等到它做好准备后才能发 SDONE 信号。

如果 SBO＃信号也在时钟 5 上发出，就表示监听命中了一个修改过的行并且将要进行回写操作。Cache 可以像对待其它任何命令一样对待存储器写并无效命令。

在得知监听结果之前，Cache 为采样到有效的 TRDY＃信号所等待的时间是固定的。如果 TRDY＃在监听结果出现之前有效时，建议使 Cache 不要回写相应的行并发出 CLEAN。

4. 数据传输→命令一个修改行→回写

此类操作的时序关系如图 3.24 所示。

在图 3.24 中，第一个传输开始于时钟 2，完成于时钟 3。在对第一个传输监听的过程中，另一个传输开始于时钟 5，完成于

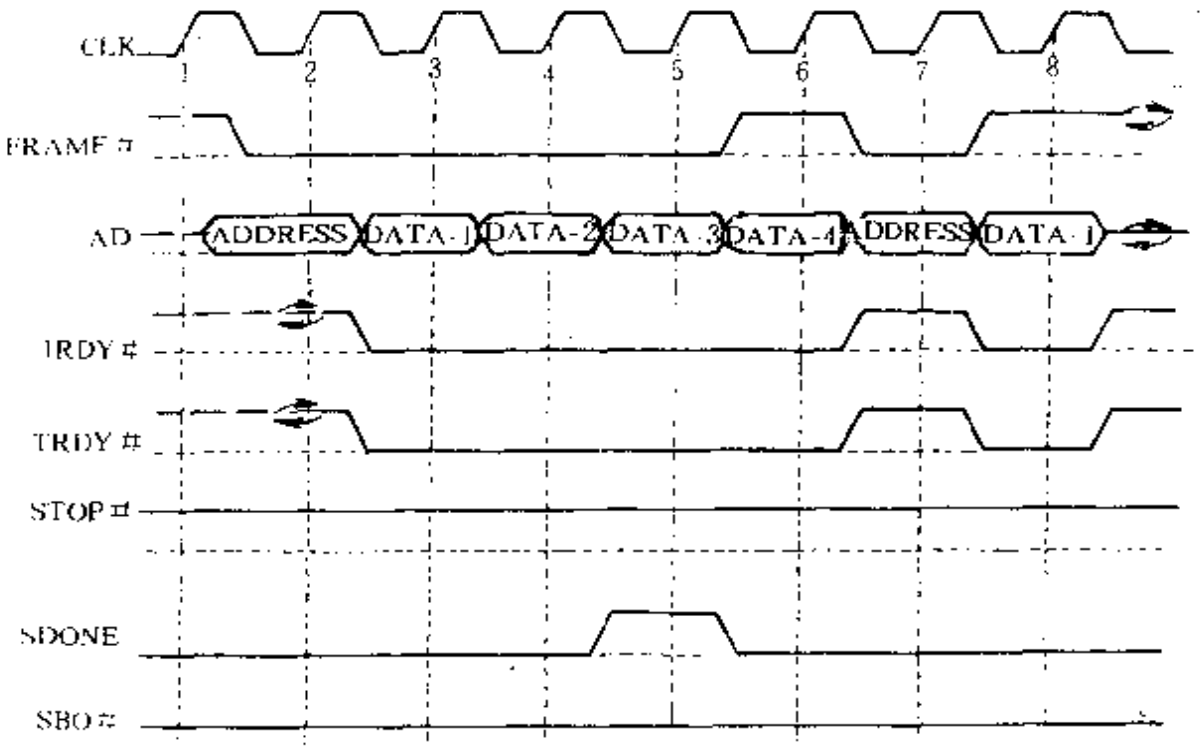


图 3.23　存储器写并无效命令

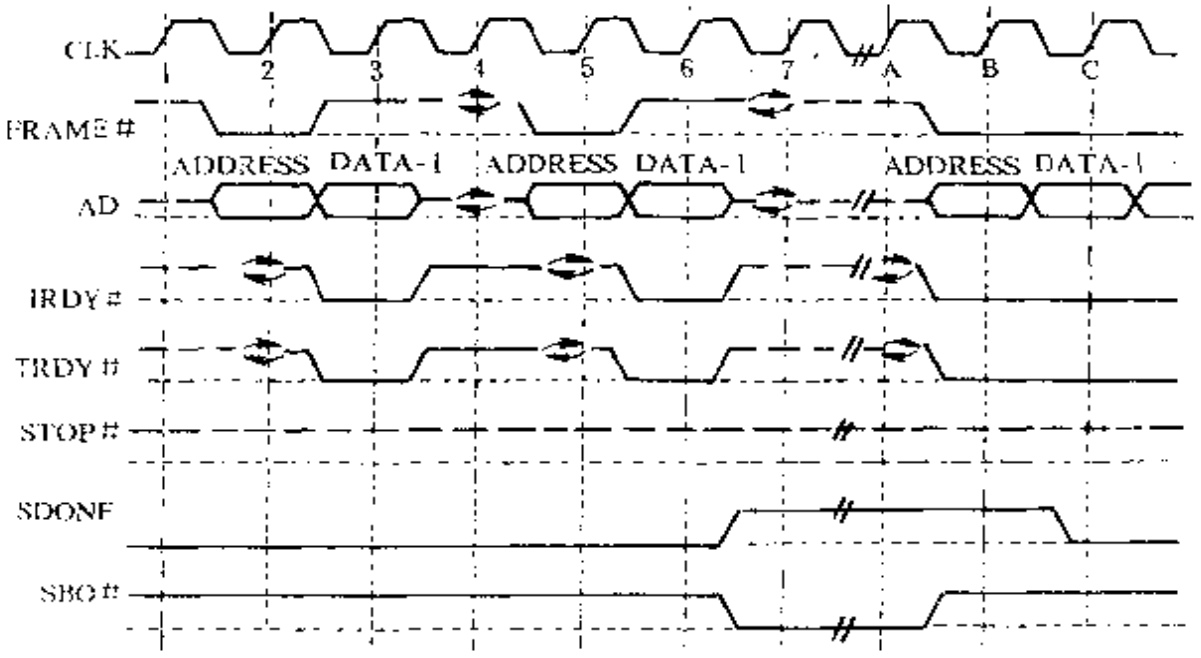


图 3.24　数据传输→监听命中→回写

时钟 6。第二个传输之所以能在第一个监听进行期间完成，说明

它肯定是一个不可缓存的传输。到了时钟 7 第一个传输的监听才完成。一旦 FRAME# 建立并且一个监听也正在进行当中时，在 SDONE 信号建立之前，SDONE 和 SBO# 的状态只对第一个地址有意义。而 SDONE 建立之后，当它下一次再发出时就属于第二个传输了。如果在图 3.24 中，SDONE 不是在时钟 7 而是在时钟 5 发出，那么对第二个传输的监听仍然没有结果。

3.9.4 对写穿式 Cache 的支持

对一个写穿式 Cache 的支持和回写式 Cache 基本相同，唯一不同之处在于不用 SBO# 信号。存储控制器监视总线并且跟踪有多少地址需要处理，最多可以有两个。每当 SDONE 信号有效时，存储控制器便允许完成另一个可缓存的传输。

在写穿式方式下，仅仅支持一种状态变换序列，即

STANDBY→CLEAN→[CLEAN]→STANDBY

由于 SBO# 信号在该方式中不使用，可由系统设计者接为高电平，所以就只有 STANDBY 和 CLEAN 之间的状态变换了。如果 Cache 是第二个传输中的主设备，则只有出现下列两种情形之一时，它才可以续续发 CLEAN，否则就转变为 STANDBY 状态。

① 传输过程是一个 Cache 行的回写；

② Cache 已经知道监听结果会是 CIEAN 状态。

这里建议可缓存的目标设备既实现 SDONE 也实现 SBO# 。

对于每个在总线上发出的 FRAME#，Cache 将在它监听完地址之后发 SDONE 信号。如果第二次 FRAME# 出现时没有 SDONE 信号，那么第二个传输若是可缓存的就不能完成，存储器控制器必须插入等待周期直到前一个监听完成，发出 SDONE 信号。如果第二个访问是不可缓存的，这个访问可以完成并且 Cache 将不对此地址进行监听。

3.9.5 支持 Cache 时的仲裁注意事项

当总线上出现 HITM 状态时，要求仲裁器应进入某种公平的仲裁算法，否则，可能会发生互锁。之所以出现这种互锁，是因为两个更高优先级的设备在访问可缓存的存储器从而使修改过的行无法进行回写所造成。当总线上出现 HITM 时，所有的可缓存访问都要被再试而终止。

建议当系统中有一个 Cache 存在时，仲裁器可以把它的 REQ # 信号线连接到固定的输入端上，这样做，就可以使得在总线上出现 HITM 时把它的优先级升成最高。以保证当有回写等待时，使被再试终止的可缓存对话减至最少，访问延迟也最少。

当系统中用一个 Cache 时，特别是回写式 Cache 时，对目标设备访问延迟的仲裁必须考虑到回写一修改行所需的时间，具体地讲，就是要有所增加。这个增加值的大小取决于回写仲裁算法。

3.10 扩充为 64 位总线

PCI 总线支持 64 位数据通路，用来提供附加的带宽，以满足一些设备的需要。对于 64 的设备需要增加 39 条引线，它们是：REQ64 #、ACK64 #、AD[63 : : 32]、C/BE[7 : : 4] # 及 PAR64，这些信号线的具体定义在前面章节中已有叙述，在此不再说明。在系统复位后，REQ64 # 通知 64 位设备，它是否连接了 64 位数据通路。很显然，如果 REQ64 # 没有信号，说明设备没有接到 64 位数据通路上；反之，当 REQ64 # 有效，则表明 64 位数据通路已投入工作。

64 位数据传输的完成，要依靠主设备和从设备之间在每个地址期动态协调进行。为达到此目的，主设备在进行 64 位传输

时发出 REQ64＃信号，而相应的从设备以 ACK64＃作为响应。REQ64＃和 ACK64＃要被接成外部上拉信号，以便它们在 32 位和 64 位设备混合使用的环境中具有正确的动作。64 位的传输协定一经约好，就要一直保持到传输的完成。

对于 32 位的设备不需要作任何修改就可以同 64 位的设备一起工作。64 位的设备在没有协商进行 64 位操作之前，其缺省工作方式应为 32 位。因此，64 位的传输对 32 的设备是完全透明的。

在 64 位传输期间，PCI 总线的所有协议和时序关系均如前所述，保持不变，只有存储器命令变为 64 位传输。中断应答和特殊周期命令是基本的 32 位命令，绝对不允许使用 REQ64＃。对于 I/O 命令和配置命令来说，若使用 64 位传输时，其带宽的要求与增加的复杂性相比是不合算的。所以，只有存储器命令支持 64 位数据传输。

无论一次传输是 32 位还是 64 位，所有的存储器命令和总线传输动作都是一样的。64 位的设备在每一个数据期可以传输 1～8 个字节，并且字节使能信号的任何组合形式都是合法的。就像在 32 位模式中一样，字节使能信号在每个数据期可以改变。发起 64 位数据传输的设备必须使用 4 个双字(8 个字节)长度的地址，而且在地址期内 AD2 必须为“0”。

64 位情况下的奇偶校验除增加了一个奇偶信号外，其它的工作情况与 32 位奇偶校验一样。PAR64 覆盖了 AD[63：：32]和 C/BE[7：：4]＃，但时序关系及其功能与 PAR 完全相同。在任何传输中，PAR64 必须在 REQ64＃有效的地址期之后一个时钟周期有效，同时，在 AD[64：：32]、C/BE[7：：4]＃及 PAR64 各线上“1”的个数也要等于偶数。

然而，在数据期，PAR64 必须由 REQ64＃和 ACK＃进行再次认可。PAR64 对于数据期来说是必须的，对于 64 位设备来说也

不是选项。

在图 3.25 和图 3.26 中，都是 64 位的主设备请求进行 64 位的传输。图 3.25 表示的是一个读操作，从设备以 ACK64＃响应。而图 3.26 所示传输为一个写操作，从设备没有给予响应，传输便自动成为 32 位。这两张图与图 3.1 和图 3.2 是一样的，只是增加了 64 位信号而已。

AD[63：：32]在地址期内是保留的，但在 64 位数据期内含有数据。C/BE[7：：4]＃在地址期也是保留的，在数据期内它是高位 4 个字节的字节使能信号。

在图 3.25 中，一个主设备发 REQ64＃(与 FRAME＃完全重复)来请求一个 64 位的读传输，从设备以 ACK64＃(与 DEVSEL＃相重合)来回答这个请求。数据期的延长是靠两个设备推迟发准备好信号来达到的。64 位信号同 32 位信号一样，要求同样的过渡周期。

在图 3.26 中，一个主设备请求 64 位传输，但目标设备不理解 REQ64＃，所以 ACK64＃由一个上拉电阻维持在无信号状态。该传输一联系到目标就变成 32 位了。对应的主设备要把 64 位传输转变为 32 位。由于这样的变换，在传输的任何数据期都可能出现没有字节使能信号的情况。所以，所有的 32 位目标设备都必须能够处理没有字节使能信号的数据期。对于这种数据期，目标设备不应当使用断开或者再试，而是应当发 TRDY＃信号并完成该数据期。

在图 3.26 中，主设备在第二个数据期把第一数据期出现于 AD[64：：32]上的数据放在 AD[31：：00]上发出，以后的数据就完全按 32 位传输。

64 位数据传输适合多数据期传送大批数据，对于单一数据期，采用 64 位传输未必能提高效率。因为在 DEVSEL＃信号送回之前主设备还不知道在哪个周期撤消，只好等待 IRDY＃。

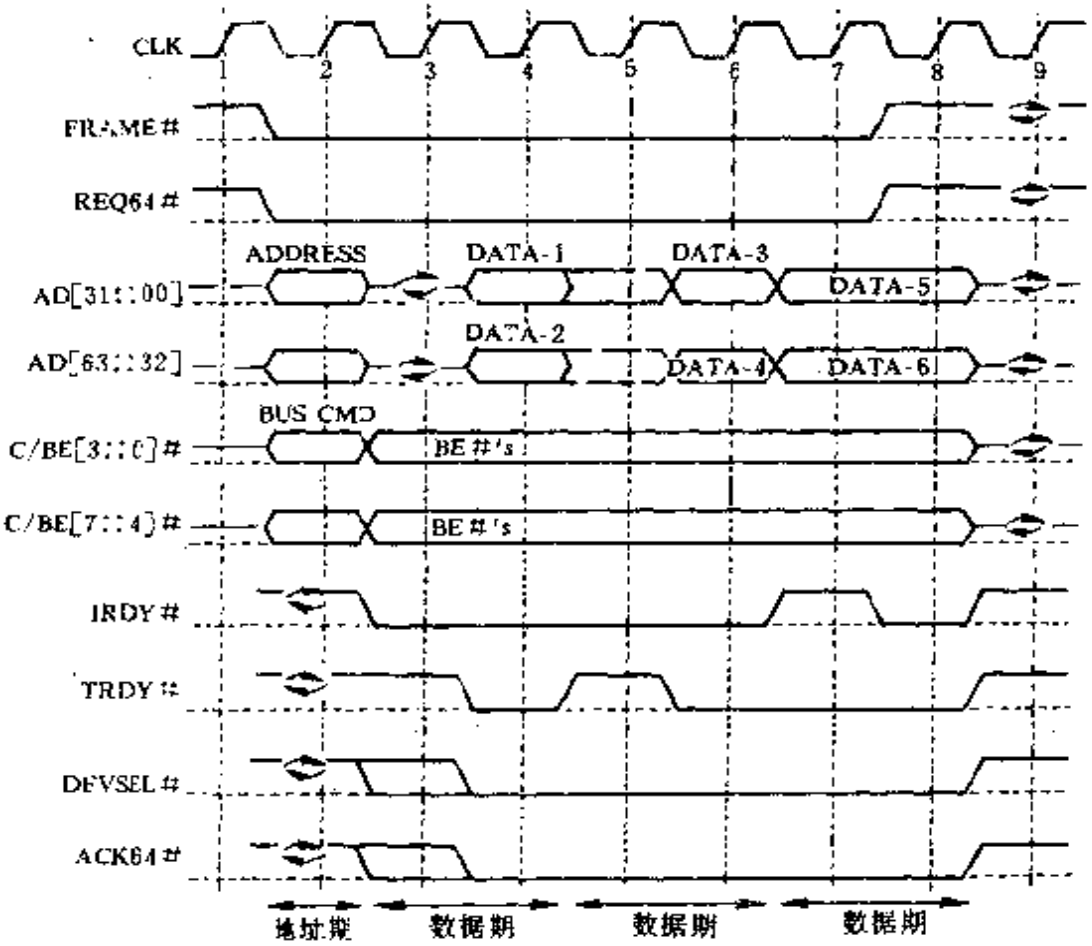


图 3.25　64 位请求—64 位传输

下面来讨论在 PCI 总线上的 64 位寻址情况。PCI 总线上定义了一个机构，用来把 64 位地址从主设备送到目标设备以支持超出 4GB 的寻址范围。一个主设备无论它支持 32 位还是 64 位数据通路，都可以发出 64 位地址。对此，也不需要增加额外的信号线。如果主、从设备都支持 64 位数据通路，则整个 64 位地址可以在一个单一时钟周期内提供。只支持 32 位寻址的目标设备将以透明方式和可发 64 位地址的主设备一起工作，但此时要映射到低 4GB 的地址空间。

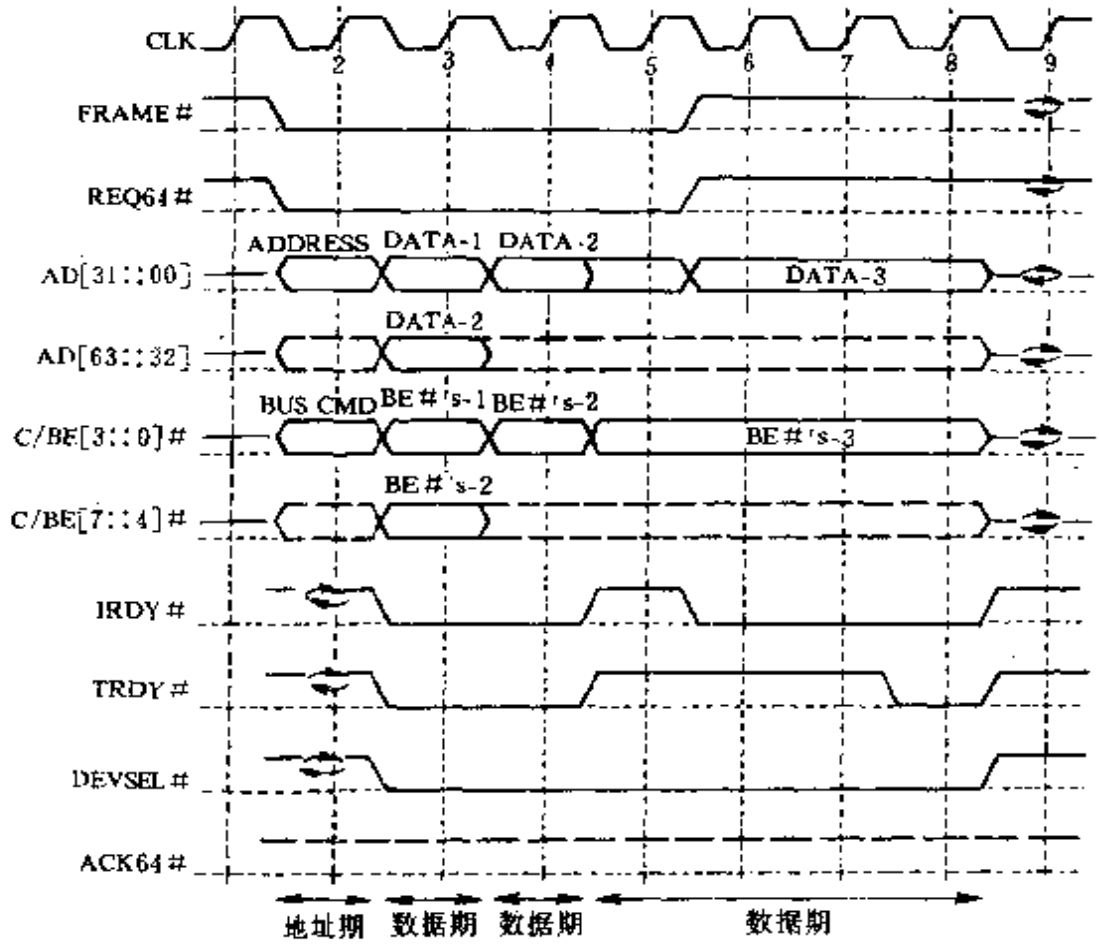


图 3.26　64 位请求—32 位传输

标准的 PCI 总线传输支持单一地址周期方式(SAC)。为了支持 64 位地址传送而又不用 64 位数据通路，采用了双地址周期(DAC)。DAC 用两个时钟周期来传送 64 位地址的全部，每个时钟传送 32 位。如果主设备采用地址渐进方式则不能实现 64 位寻址，因为无法延迟或延伸第二个地址期。

图 3.27 表示了一个 DAC 机构的情况。

在一个基本的 PCI 读传输中，地址期后面应当是一个过渡周期。而在 DAC 传输中，在标准的地址期和过渡周期之间插入了一

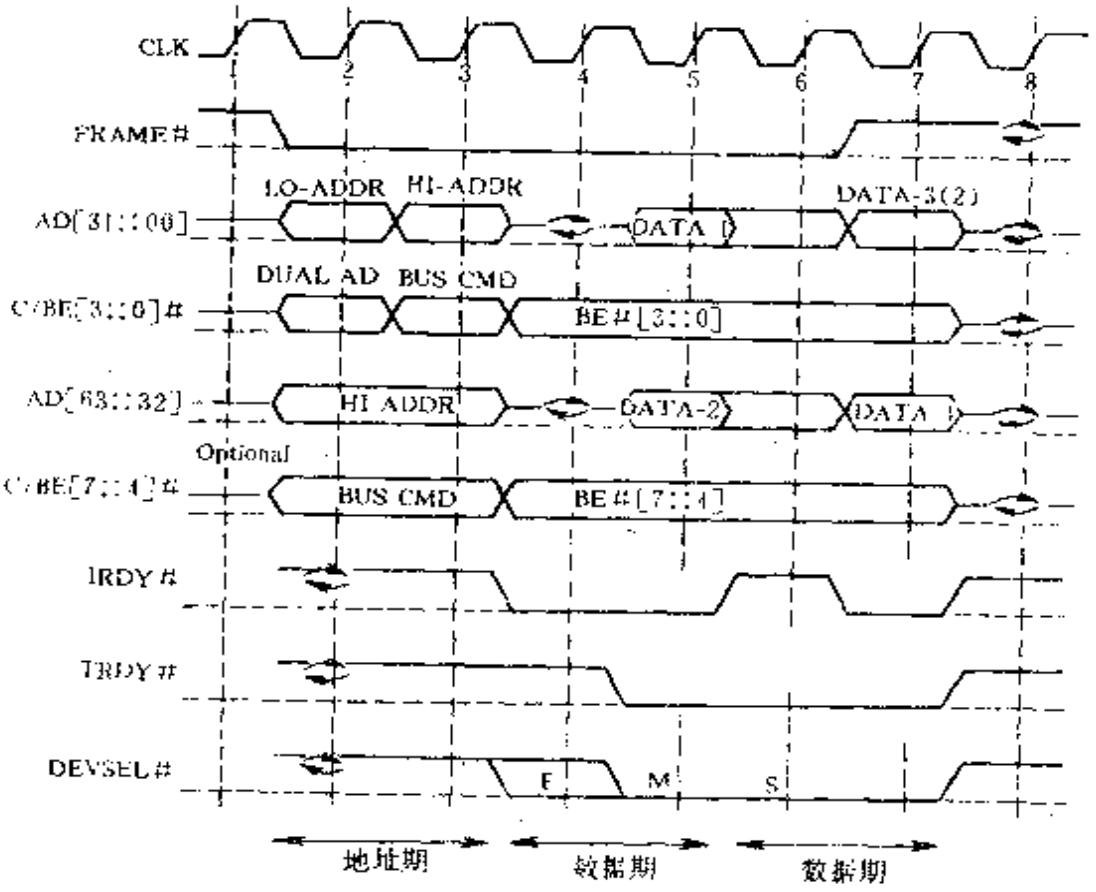


图 3.27　64 位双地址周期

个附加的地址期(如图 3.27 中的 HI－ADDR 所示)，也可称它为第二地址期。在图 3.27 中，第一和第二地址期分别出现于时钟 2 和时钟 3，而地址期与数据期之间的过渡周期顺延至时钟 4。注意：在两个地址期期间，FRAME＃信号必须有效。

为了保持 FRAME＃和 IRDY＃两个信号之间的相对关系，在 IRDY＃有效之前 FRAME＃不能撤消；写传输的数据未被提供或读传输的数据未接收之前 IRDY＃不能撤消。

DAC 运行机制：第一地址期在 C/BE[3：：0]＃上出现“1101”代码，表示双地址周期；第二地址期在 C/BE[3：：0]＃上

出现总线命令。第一地址期在 AD[31：：00]上出现地址低位，而高位在第二地址期出现。如果目标设备支持 64 位寻址，它就会按上述顺序锁定整个地址和总线命令，并决定是否发 DEVSEL#信号。

图 3.27 表示了 64 位数据通路高 32 位的情况，这只有主设备支持 64 位数据通路才会发生。在第一地址期，主设备驱动整个地址(低位在 AD[31：：00]，高位在 AD[63：：32])和总线命令(DAC 代码在 C/BE[3：：0]#上，总线命令在 C/BE[7：：4]#上)。在第二地址期，主设备驱动地址高 32 位在 AD[31：：00]和 AD[63：：32]上，同时将总线命令驱动在 C/BE[3：：0]#和 C/BE[7：：4]#上。在整个地址传输被完成之前，主设备还不能确定目标设备是否支持 64 位数据通路。所以，它必须假定目标设备是 32 位的，以使得提供的地址既适合于 32 位的目标设备同时也适合于 64 位的目标设备。

如果主、从设备都支持 64 位数据通路，那么 64 位的寻址不会被 DEVSEL#的确定产生延迟。即使从设备因访问延迟而插入了等待周期，附加的地址期也没有性能方面的影响。如果主、从设备都不支持 64 位数据通路，将会出现一个时钟的附加延迟。

对于一个支持 64 位数据通路的主设备，当高 32 位地址为 0 时，它必须产生一个 SAC 以替代 DAC，这样一来，它就可以借助于 SAC 产生 64 位地址从而与 32 位的从设备进行通信。SAC 和 DAC 两种寻址方式的确定是根据地址的大小而不是目标设备的能力。

另外，有两种基本途径可使支持 32 位寻址的主设备能够与支持 64 位寻址的目标设备进行通信：其一是可以将 64 位寻址的目标设备看作 32 位；其二是利用 DAC 来实现。

第四章　PCI 总线的电气规范

4.1　概述

PCI 总线的电气规范定义了所有 PCI 元件、系统扩展板的电气性能和约束，以及扩展板连接器的引脚分配。规范中提供了 5 V 和 3.3 V 两种信号环境，信号环境不能混合使用，也就是说，对一个给定的 PCI 总线，所有的元件必须使用同一个信号规则。但是，通过设计可使 5 V 的元件工作于 3.3 V 的信号环境，反之亦然，这一点将在后面进行说明。

PCI 总线是一个 CMOS 总线，也就是说其静态电流是非常小的。实际上，直流驱动电流主要消耗在上拉电阻上。

PCI 总线的信号驱动采用反射波方式而不是入射波。所谓的反射波方式驱动是指：总线驱动器只把总线信号的幅度驱动到所要求幅度(高电压或低电压)的一半，然后电波沿着总线向目标传播，到达目标后再向原点反射，从而使原来的电压振幅加倍以达到要求的电压级别。实际上，在这段传播时间内总线驱动处于开关范围的中间，这段时间的长短至少为 10 ns，在 33 MHz 时钟下，相当于总线周期的 1/3。PCI 的上述两个电气特征，决定了 PCI 总线，I/O 缓冲区特性的定义方式与众不同。

PCI 总线驱动器在瞬变开关上花费了较多的时间，并且直流电流又很小，从而使按直流驱动源的能力定义缓冲区的传统方法不能使用。PCI 总线驱动器的指标要用交流开关特性来定义，而不用直流驱动能力。尤其是驱动器在其整个有源开关范围内的电压电流关系为主要技术指标时，更应如此。这些电压/电流关系要

求：在典型配置为主板上有六个负载和两个扩展连接器或者两个负载和四个扩展连接器的情况下，达到可接受的开关行为。但是，也有可能达到不同或更大的配置，这取决于实际设备、布局安排及主板上的负载阻抗等因素。

4.2 PCI 元件指标

本节主要讨论 PCI 元件的电气参数和时间参数。无论是 5 V 还是 3.3 V 信号环境都有其相应的指标。5 V 环境基于绝对的开关电压，目的是为了和 TTL 开关电平兼容。而 3.3 V 环境却是基于 V_{cc}相关的开关电压，是一种最佳化的 CMOS 方法。这样做的意图是：无论在主板上还是在扩展板上，PCI 元件都可以直接相连，而不需要额外附加缓冲或采用其它的连接方式。

PCI 元件的输出缓冲要利用它们的电压/电流关系曲线来定义。但对这个关系曲线是有所限制的，换一个角度来说，就是规定了最大输出阻抗和最小输出阻抗。前者的目的是为了在典型配置下使输出达到一个可接受的第一阶段电压；而后者是为了将反射波维持在一个合理的边界之内。缓冲的上拉边和下拉边分别有不同的电压/电流曲线，这些曲线要与参数指标一同提供。

有效的缓冲强度主要由一个 AC 驱动点来定义，即规定一个可接受的第一阶段电压以及在一典型配置中达到该电压所要求的电流。而 DC 驱动点的定义必须维持静态条件，但这在 CMOS 环境下是很小的，并且不能代表实际的输出驱动强度。图 4.1 中阴影区域便是输出特性的允许范围。

直流(DC)参数在静态(DC)条件下必须是持久的；而交流(AC)参数在瞬态(AC)开关条件下必须得以保证，其时间可能占到时钟周期的 33%。所有电气参数的方向符号均以元件内部地为参考点，例如，正的电流值表示流入元件内部，而负的电流值表

示从元件中流出。

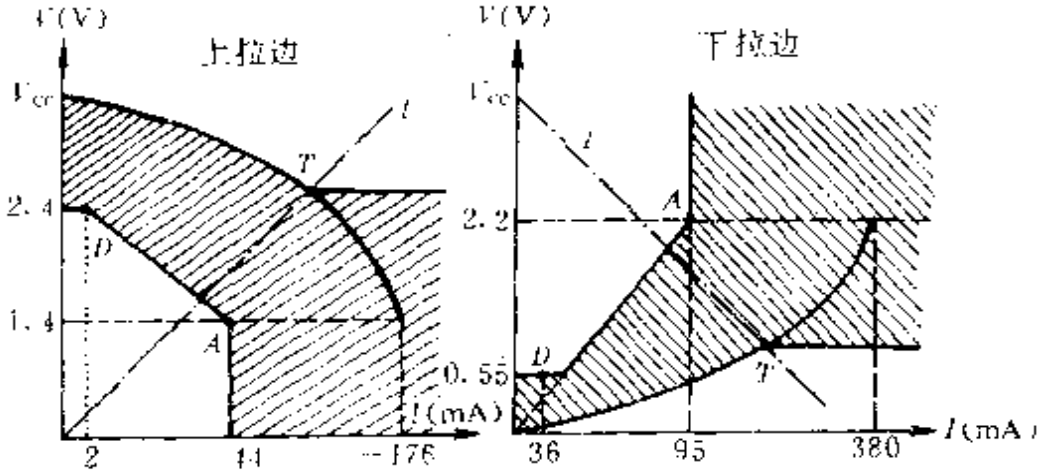


图 4.1　5 V 信号 V/I 曲线

A—交流驱动点；D—直流驱动点；T—测试点；L—22 Ω 负载线

4.2.1　5 V 信号环境下的指标

1. 直流指标

表 4.1 简要给出了 5 V 信号环境下的直流技术参数。

表 4.1 中"注"所对应的内容如下：

① 输入漏电流包括所有双向缓冲三态输出的高阻抗输出漏电流。

② 未加上拉电阻的信号必须有 3 mA 的低输出电流；要求加上拉电阻的信号必须有 6 mA 的低输出电流。对于后者包括以下信号：FRAME#、TRDY#、IRDY#、DEVSEL#、STOP#、SERR#、PERR#、LOCK#、AD[63 : : 32]、C/BE[7 : : 4]#、PAR64、REQ64#和 ACK64#。

③ 除 CLK 引脚外，一个 PCI 输入端的最大引脚电容为

表 4.1　5 V 信号下的直流指标

名　称	含　义	条　件	最小值	最大值	单位	注
V_{cc}	电源电压		4.75	5.25	V	
V_{ih}	输入高电压		2.0	$V_{cc}+0.5$	V	
V_{il}	输入低电压		−0.5	0.8	V	
I_{ih}	输入高漏电流	$V_{in}=2.7$		70	μA	①
I_{il}	输入低漏电流	$V_{in}=0.5$		−70	μA	①
V_{oh}	输出高电压	$I_{out}=-2$ mA	2.4		V	
V_{ol}	输出低电压	$I_{out}=3$ mA,6 mA		0.55	V	②
C_{in}	输入引脚电容			10	pF	③
C_{clk}	CLK 引脚电容		5	12	pF	
C_{IDSEL}	IDSEL 引脚电容			8	pF	④
L_{pin}	引脚电感			20	nH	⑤

10 pF，而对于只用在主板上的设备，可以达到 16 pF 以适应 PGA 封装。这也意味着扩展板上的元件将不用 PGA 封装，而改用 PQFP、SGA 等封装形式。

④ 这个只用作输入的引脚上电容量较低，它可以不经电阻而耦合到 AD 线上。

⑤ 这仅是一个建议，不是绝对要求。其实际值应与元件数据清单一起提供。

对于作为扩展数据通路的引脚 AD[63 :: 32]、C/BE[7 :: 4]# 和 PAR64，由于它们在 32 位设备的传输中是不用的，所以要加上拉电阻或者输入"保持器"，否则，有可能浮动达到门坎电平，而造成振荡或者通过输入缓冲泄漏大的电源电流。该上拉电阻或保持器必须是主板上(不包括扩展板)中央资源的一部分，以避免上拉电流过载。如果一个设备具有 64 位数据通路但没有连接(把一个 64 位卡插在 32 位的 PCI 插槽上)，此时，PCI 元件的主要责任是要确保它的输入不能振荡，至于通过输入缓冲泄漏电源电流这一问题就显得不重要了。

2. 交流指标

表 4.2 摘要列出了 5 V 信号环境下的交流技术指标。表达式 *A*、*B* 分别为：

表达式 $A=I_{oh}=11.9\times(V_{out}-5.25)\times(V_{out}+2.45)$

要求：$V_{CC}>V_{out}>3.1$ V

表达式 $B=I_{ol}=78.5\times V_{out}\times(4.4-V_{out})$

要求：$0V<V_{out}<0.71$ V

表中"注"的具体内容如下：

① 参见图 4.1 的 *V*/*I*(电压/电流)曲线。REQ# 和 GNT# 的开关电流特性可以是这里指定值的一半。这个指标不适合于 CLK 和 RST#，它们都是系统输出。"开关电流高"也不适用 SERR#，它是 OD 输出。

表 4.2　5 V 信号下的交流指标

名称	含　义	条　件	最小值	最大值	单位	注
I_{oh} (AC)	开关电流高	$0 \leqslant V_{out} \leqslant 1.4$	-44		mA	①
		$1.4 \geqslant V_{out} < 2.4$	$-44+(V_{out}-1.4)/0.024$	表达式 A	mA	①,②,③
	测试点	$V_{out}=3.1$		-142	mA	③
I_{ol} (AC)	开关电流低	$V_{out} \geqslant 2.2$	95		mA	①
		$2.2 > V_{out} > 0.55$	$V_{out}/0.023$	表达式 B	mA	①,③
	测试点	$V_{out}=0.71$		206	mA	③
I_{cl}	低嵌位电流	$-5 < V_{in} \leqslant -1$	$-25+(V_{in}+1)/0.015$		mA	
t_r	空载上升时间	0.4 V～2.4 V	I	5	V/ns	④
t_f	空载下降时间	2.4 V～0.4 V	I	5	V/ns	④

② 注意这一段最小电流曲线是从 AC 驱动点直接连到 DC 驱动点，而不像下拉曲线上那样朝着电源电压连线。这是为了选一个 N 沟道上拉。

③ 当驱动器驱动超过第一阶段电压时(AC 驱动点)，必须满足最大电流的要求。表达式规定的最大值应当在设计中予以满足。为了方便元件测试，对每种情况下的输出驱动都定义了一个最大电流测试点。

④ 所有的 PCI 设备都必须满足这个最小摆动速率要求(最慢的信号边沿)，最大摆动速率(最快信号边沿)是一个指导方针。元件供应商必须心中有数，信号边沿越快则越有可能在信号线上造成干扰并在系统中形成信号性错误。作为母板的设计者也应注意，上升下降时间比这里所列最大值更快的情况也是可能发生的，应在信号完整性模型中考虑到这些因素。

在进行表 4.2 所列参数的测试时，要求将输入端嵌位到地。到 5 V 电源线的嵌位是可选的，但有可能需要用它来保护 3.3 V 的输入设备。在 5 V 信号环境下决不可使用 3.3 V 电源线嵌位。当使用双电源线时，在它们之间会存在寄生的二极管通路。

3. 最大交流允许值和设备保护

这里提出一个最大交流测试指标并作为一种测试方法的建议。这是因为 PCI 环境含有许多有抗元件，而且一般都必须当作一个无端接的传输线来对待。该环境的基本要求是：一个信号只有在线的末端反射并返回到驱动点时，才被认为是完成了信号的开关。

由于这个环境以及驱动器、设备拓朴结构和板上阻抗等条件，PCI 设备引脚上的"开路"电压将超过预期的地到 V_{cc} 电压范围一个相当大的数值。不同的供应商采用不同的途径来实现 PCI 技术，因此不能认为技术上能自然地抵抗这些影响。这里给出的测试指标是基于 AC 环境的人为的最坏情况，可以用来对设备的长

期可靠性进行评价。

每个 PCI 设备的所有输入、双向的和三态的输出都应该能连续地暴露于下列测试条件中。测试等效于用一个 0 内阻的电压源去驱动一个直接连到 PCI 设备的每个输入或三态引脚上的串联电阻。电压源的波形和串接电阻的阻值如图 4.2 所示。该测试只覆盖了交流操作条件，并分为上行测试波形和下行测试波形。

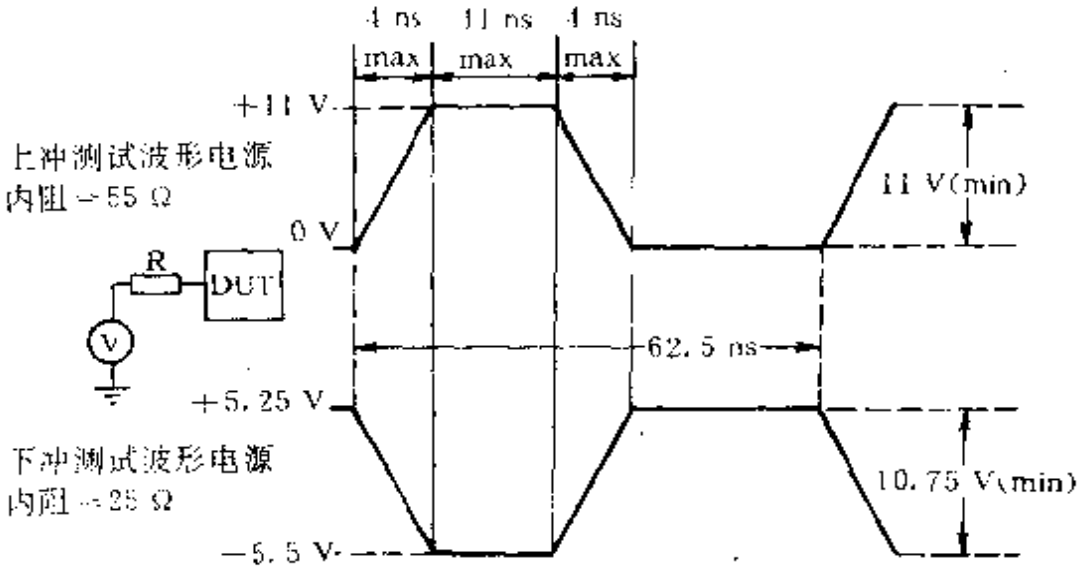


图 4.2　5 V 信号的测试波形

4.2.2　3.3 V 信号环境下的指标

1. 直流技术指标

表 4.3 摘要列出 3.3 V 信号环境下的直流技术指标。

表 4.3 中"注"一栏的具体内容对应如下：

① 这一指标应当由设计保证，是计算上拉电阻(使之上拉一个浮动网络)时应考虑的最小电压。对静态耗电敏感的应用中，应保证输入缓冲在此电压下的电流最小。

② 输入漏电流包含所有的双向三态输出缓冲的高阻抗输出

表 4.3　3.3 V 信号下的直流指标

名　称	含　义	条　件	最 小	最　大	单位	注
V_{cc}	电源电压		3.0	3.6	V	
V_{ih}	输入高电压		0.475 V_{cc}	V_{cc}+0.5	V	
V_{il}	输入低电压		−0.5	0.325 V_{cc}	V	
V_{ipu}	输入上拉电压		0.77 V_{cc}		V	①
I_{il}	输入漏电流	$0<V_{in}<V_{cc}$		±10	μA	②
V_{oh}	输出高电压	$I_{out}=-500$ μA	0.9 V_{cc}		V	
V_{ol}	输出低电压	$I_{out}=1500$ μA		0.1V_{cc}	V	
C_{in}	输入引脚电容			10	pF	③
C_{clk}	CLK 引脚电容		5	12	pF	
C_{IDSEL}	IDSEL 引脚电容			8	pF	④
L_{pin}	引脚电感			20	nH	⑤

漏电流。

③ 一个 PCI 输入端的绝对最大引脚电容量为 10 pF(CLK 除外)，对于只用在主板上的设备可以达到 16 pF，以便适应 PGA 封装。扩展板上的元件最好不用 PGA 封装，而改用 PQFP 或 SGA 等封装形式。

① 在这个只作为输入的引脚上，引脚电容量比较低，目的是为了不加电阻能够将它耦合到 AD 线上。

⑤ 这仅仅是一个建议，而不是绝对要求。其实际值应该与元件数据清单一同提供。

对于作为扩展数据通路的引脚 AD[63 :: 32]C/BE[7 :: 4]# 及 PAR64，由于它们在 32 位设备的传输中是不用的，所以要加上拉电阻或者输入"保持器"。否则，有可能浮动到门坎电平，从而造成振荡或者通过输入缓冲泄漏大的电源电流。该上拉电阻或保持器必须是主板上(不包括扩展板)中央资源的一部分，以避免上拉电流产生过载。如果一个设备具有 64 位数据通路但未连接(将一个 64 位卡插在 32 位的 PCI 插槽上)，这时，PCI 元件的主要责任是要确保它的输入不能振荡，至于通过输入缓冲泄漏电源电流这一问题就显得不重要了。

2. 交流技术指标

表 4.4 摘要列出了 3.3 V 信号环境下的交流技术指标。

表中的表达式 C、D 分别为：

表达式 $C=I_{oh}=(98.0/V_{cc})\times(V_{out}-V_{cc})\times(V_{out}+0.4\ V_{cc})$

要求：$V_{cc}>V_{out}>0.7\ V_{cc}$

表达式 $D=I_{ol}=(256/V_{cc})\times V_{out}(V_{cc}-V_{out})$

要求：$0\ V<V_{out}<0.18\ V_{cc}$

表 4.4 中"注"一栏的具体内容对应如下：

① 参见图 4.3 的 V/I 曲线。REQ# 和 GNT# 的开关电流特性可以是这里指定值的一半，也就是说，可在这些信号上使用 1/2

表 4.4　3.3 V 信号下的交流指标

名称	含　义	条　件	最　小	最　大	单位	注
I_{oh} (AC)	开关电流高	$0<V_{out}\leqslant 0.3V_{cc}$	$-12V_{cc}$		mA	①
		$0.3V_{cc}<V_{out}<0.9V_{cc}$	$-17.1(V_{cc}-V_{out})$	表达式 C	mA	①,②
	测试点	$V_{out}=0.7V_{cc}$		$-32V_{cc}$	mA	②
I_{ol} (AC)	开关电流低	$V_{cc}>V_{out}\geqslant 0.6V_{cc}$	$16V_{cc}$		mA	①
		$0.6V_{cc}>V_{out}>0.1V_{cc}$	$26.7V_{out}$	表达式 D	mA	①,②
	测试点	$V_{out}=0.18V_{cc}$		$38V_{cc}$	mA	②
I_{cl}	低嵌位电流	$-3<V_{in}\leqslant -1$	$-25+(V_{in}+1)/0.015$		mA	
t_r	空载上升时间	$0.2V_{cc}-0.6V_{cc}$	1	4	V/ns	③
t_f	空载下降时间	$0.6V_{cc}-0.2V_{cc}$	1	4	V/ns	③

大小的输出驱动器。这个指标不适合 CLK 和 RST#，因为它们都是系统输出。“开关电流高”也不适合于 SERR#，因为它是 OD 输出。

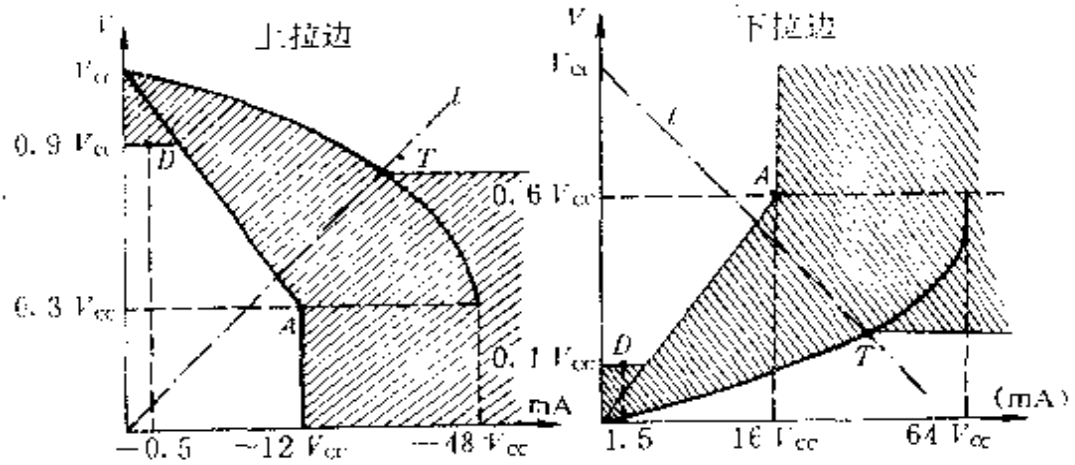


图 4.3　3.3 V 信号的 V/I 曲线

A—交流驱动点；*D*—直流驱动点；*T*—测试点；*L*—22 Ω 负载线

② 当驱动器驱动超过第一阶段电压时，必须满足最大电流要求。表达式规定的最大值应当在设计中予以满足。为了方便元件测试，对每种情况下的输出驱动都定义了一个最大电流测试点。

③ 所有的 PCI 设备都必须满足这个最小摆动速率要求(最慢的信号边沿)，最大摆动速率(最快信号边沿)是一个指导方针。元件供应商应清楚，信号边沿越快则越有可能在信号线上造成干扰并在系统中形成信号性错误。作为母板的设计者也应注意，上升下降时间比这里所列最大值更快的情况也是可能发生的，应在信号完整性模型中考虑到这些因素。

在进行表 4.4 所列参数测试时，要求将输入端嵌位到地和 V_{cc}(3.3 V)。当使用双电源线时，在它们之间会存在寄生的二极管通路。到电源的嵌位二板管及输出上拉器件，必须能够承受驱动器在变为三态之前的短路电流。

3. 最大交流允许值及设备保护

这里提出一个最大交流测试指标并作为一种测试方法的建议，该指标是基于交流环境下人为的最坏情况，模拟 PCI 总线上无端接的反射造成的各种影响，可以用来对设备的长期可靠性进行评价。

每个 PCI 设备的所有输入、双向输出和三态输出都应该能够持续地暴露于下列测试条件中。测试等效于用一个 0 内阻电压源去驱动一个直接连到 PCI 设备的每个输入或三态引脚上的串联电阻。电压源的波形和串接电阻的阻值如图 4.4 所示。

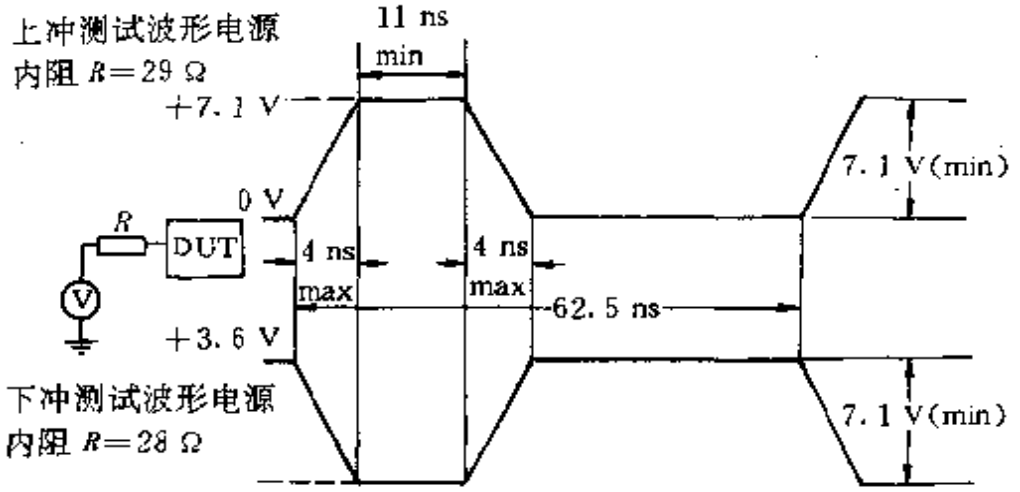


图 4.4　3.3 V 信号下的测试波形

该测试只覆盖了交流操作条件，图中的 DUT 表示被测设备和元件。

4.2.3　时间指标

1. 时钟技术指标

时钟波形必须传递到系统中每个 PCI 元件上。在扩展板上，是否符合时钟的技术指标要在扩展板上的元件处测量，而不是在连接器插槽上去测量。图 4.5 给出了在 5 V 和 3.3 V 信号环境下

各自的时钟波形和要求的测量点。

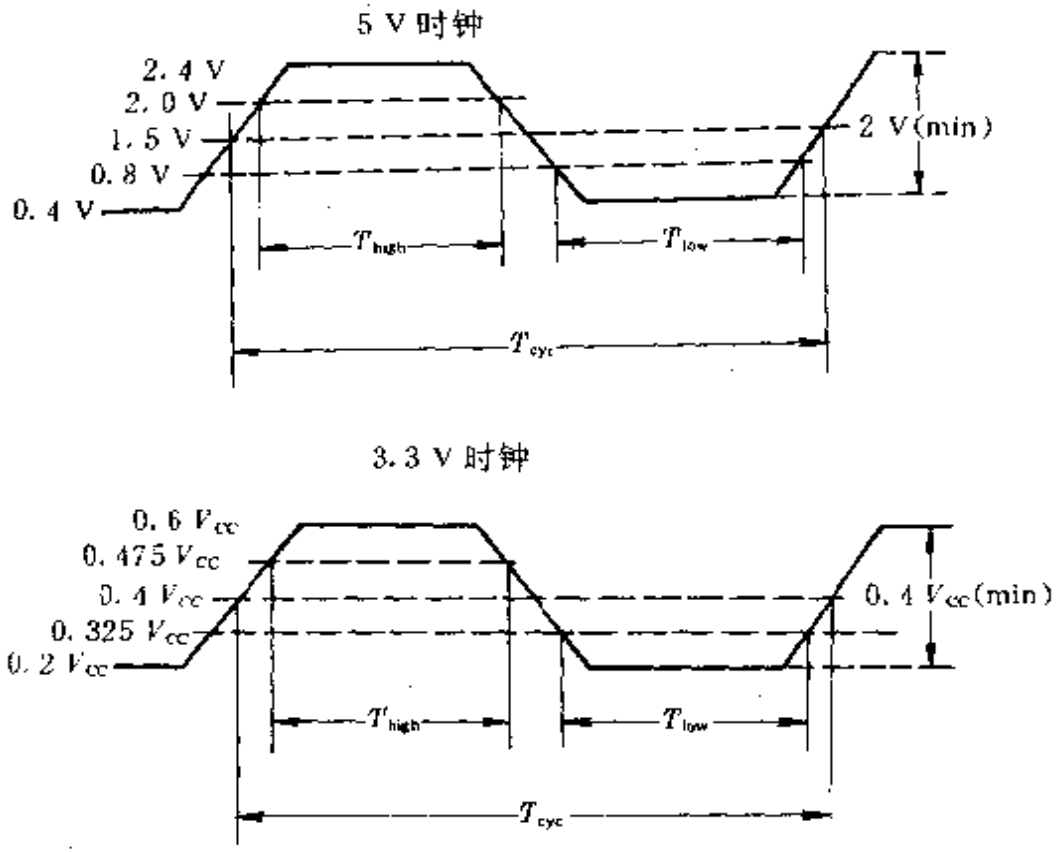


图 4.5 时钟波形

表 4.5 摘要列出了一些时钟技术指标。

表 4.5 时钟技术指标

名 称	含 义	最 小	最 大	单 位
T_{cyc}	时钟周期	30	∞	ns
T_{high}	时钟为高时间	12		ns
T_{low}	时钟为低时间	12	12	ns
	时钟摆速	1	4	V/ns

注意：

(1) T_{cyc}是时钟周期时间。一般来说，所有的 PCI 元件必须能在 0～33 MHz 范围内的任何时钟频率下工作。16 MHz 以下频段

的设备操作参数是依靠设计来保证而不是通过测试来保证的。时钟频率可在系统运行当中的任意时刻进行改变，但要保证时钟边沿是干净的并且不违反最小周期及高、低时间的规定。时钟只有在低状态时才可以被停止。如果只用于主板的元件上，则允许对规定的时间指标做一些改变。这些元件可工作于 33 MHz 及以下的任意一个单一固定频率上。

(2) 上升下降时间(摆速)指标是以 V/ns 为单位而表示的边沿变化速率，必须在图 4.5 中的最小峰--峰值部分上完全满足该变化速率。

2. 时间参数

表 4.6 给出了 5 V 和 3.3 V 信号环境下的时间指标。

表中"注"一栏的具体内容对应如下：

① 见图 4.6 中的时标测量条件。

② 最小时间是在等效负载为 0 pF 条件下测得的；最大时间是在等效负载为 50 pF 条件下测得的；实际测试电容可以有变化，但测得结果要经修正再对照此表。

③ REQ#和 GNT#是点到点信号，与总线信号在输入建立时间和输出有效时延上有差别。GNT#建立时间为 10 ns，REQ#建立时间为 12 ns，其它信号都是总线信号。

④ 见图 4.7 的时标测试条件。

⑤ RST#的有效和撤消与 CLK 不同步。

⑥ 当 RST# 有效时，所有的输出驱动都必须浮空。

3. 时标测试条件

时标的测量是在图 4.6 和图 4.7 所示的条件下进行的。在元件测试中应保证所有的时标满足最小时钟摆动速率和电压摆动，这个要求在设计过程中也应满足。此外，设计中也必须保证超出所列测试条件之外的输入操作能够满足上述要求。

上面两个图中相应参数的取值见表 4.7。

表 4.6 时标参数

名 称	含 义	最小	最大	单位	注
t_{val}	从 CLK 到总线信号输出有效	2	11	ns	①,②,③
$t_{val(ptp)}$	从 CLK 到点对点信号输出有效	2	12	ns	①,②,③
t_{on}	输出从浮动到导通	2		ns	①
t_{off}	输出从导通到浮动		28	ns	①
t_{su}	总线输入信号对 CLK 的建立时间	7		ns	③,④
$t_{su(ptp)}$	点到点输入对 CLK 的建立时间	10,12		ns	③,④
t_{rst}	电源稳定后的复位作用时间	1		ms	⑤
t_h	从 CLK 算起的输入保持时间	0		ns	④
$t_{rst-clk}$	CLK 稳定后的复位作用时间	100		μs	⑤
$t_{rst-off}$	从 RST# 有效到输出浮动之间延迟		40	ns	⑤,⑥
t_{rrsu}	REQ64# 对 RST# 的建立时间	$10T_{cyc}$		ns	
t_{rrh}	RST# 到 REQ64# 的保持时间	0	50	ns	

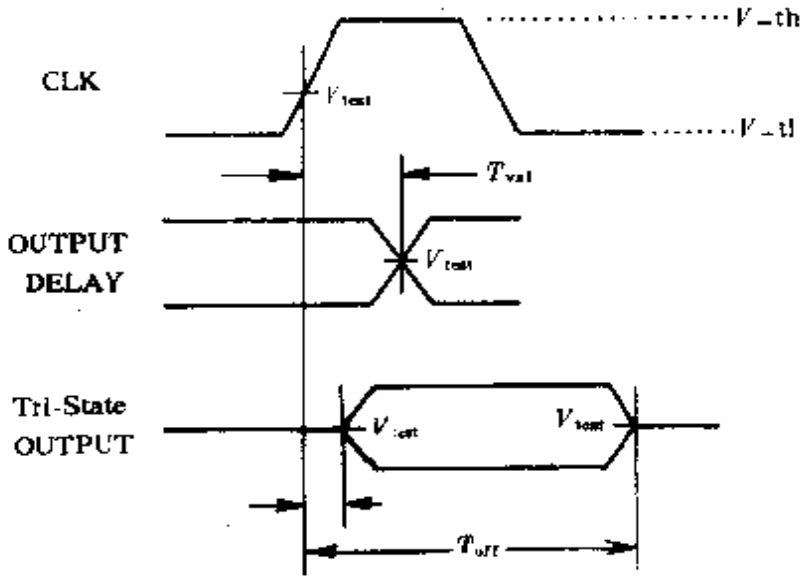


图 4.6　输出时标测试条件

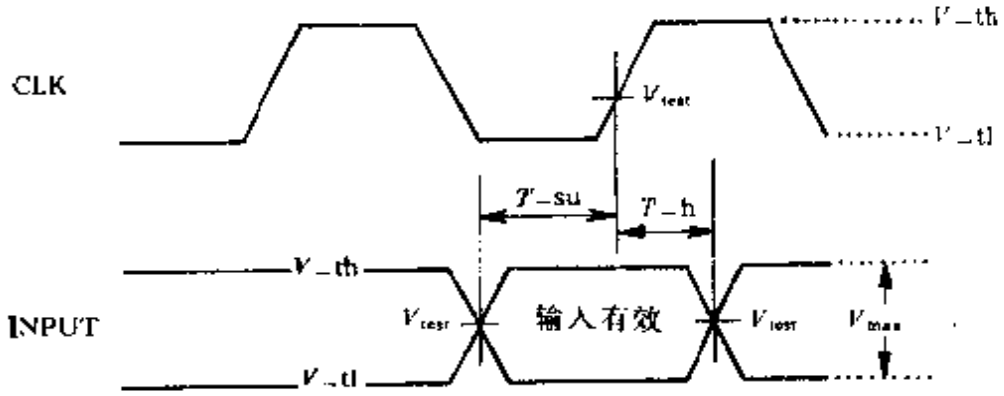


图 4.7　输入时标测试条件

表 4.7　测试条件参数

名　称	5 V 信号下	3.3 V 信号下	单位	注
V_{th}	2.4	0.6 V_{CC}	V	✓
V_{tl}	0.4	0.2 V_{CC}	V	✓
V_{test}	1.5	0.4 V_{CC}	V	
V_{max}	2.0	0.4 V_{CC}	V	✓
输入信号 边沿速率	1 V/ns			

表中“注”一栏的含义是：

5 V 环境下的输入测试是在 400 mV 的过载情况下进行的，而 3.3 V 环境下的输入测试是在 0.125 V_{cc}mV 的过载情况下进行的。时标参数必须满足这些过载条件。V_{max}是指测试输入时标所允许的最大峰一峰值。

4. 供应商提供的技术指标

在 PCI 的生存时间内，许多系统供应商将要进行 PCI 元件的板级电特性模拟。这将保证系统实现的可塑性及元件的正确使用。为了给系统商的这种努力提供方便以及提供完整的信息，元件供应商应该在他们的数据清单中提供如下信息：

(1) 所有输入引脚上的电容量。

(2) 所有输入引脚上的电感量。

(3) 在开关条件下的输出 *V/I* 曲线。对每个使用的输出类型都应给出两个曲线：一个是驱动为高时的曲线，另一个是驱动为低时的曲线，二者都应包含最好、典型、最坏三种情况。其次，超过电压范围的响应是很重要的，因此，5 V 信号的电压范围为 −5 V～10 V，而 3.3 V 信号的电压范围为 −3 V～7 V。

(4) 开关条件下的输入 *V/I* 曲线。当输出为三态时，输入机构的 *V/I* 曲线也是很重要的，也应该有最好、典型、最坏情况下的曲线，并落在 0 V～V_{cc}的范围内。

(5) 每个输出类型的无负载上升/下降时间。

(6) 完整的极限条件数据。如操作温度、非操作温度、直流最大值等等。

此外，连接器的供应商应当提供 PCI 连接器的精确仿真模型。

5. 关于元件引脚的分配

下面为 PCI 元件的引脚分配提供一个建议。由于扩展板上的总线分支线很少，如果元件的引脚分配能和板上连接器的引脚分配对应起来，将大大简化布局。只用在母板上的元件也应该遵守

同样的引脚分配方案，以使布局的分支线减少。图 4.8 给出了典型的 PQFP 封装的 PCI 元件引脚分配建议。其信号顺序与板上连接器是完全对应的(SDONE 和 SBO# 未画出)。至于电源及地线的引脚数目和排列方式与具体设备有关。

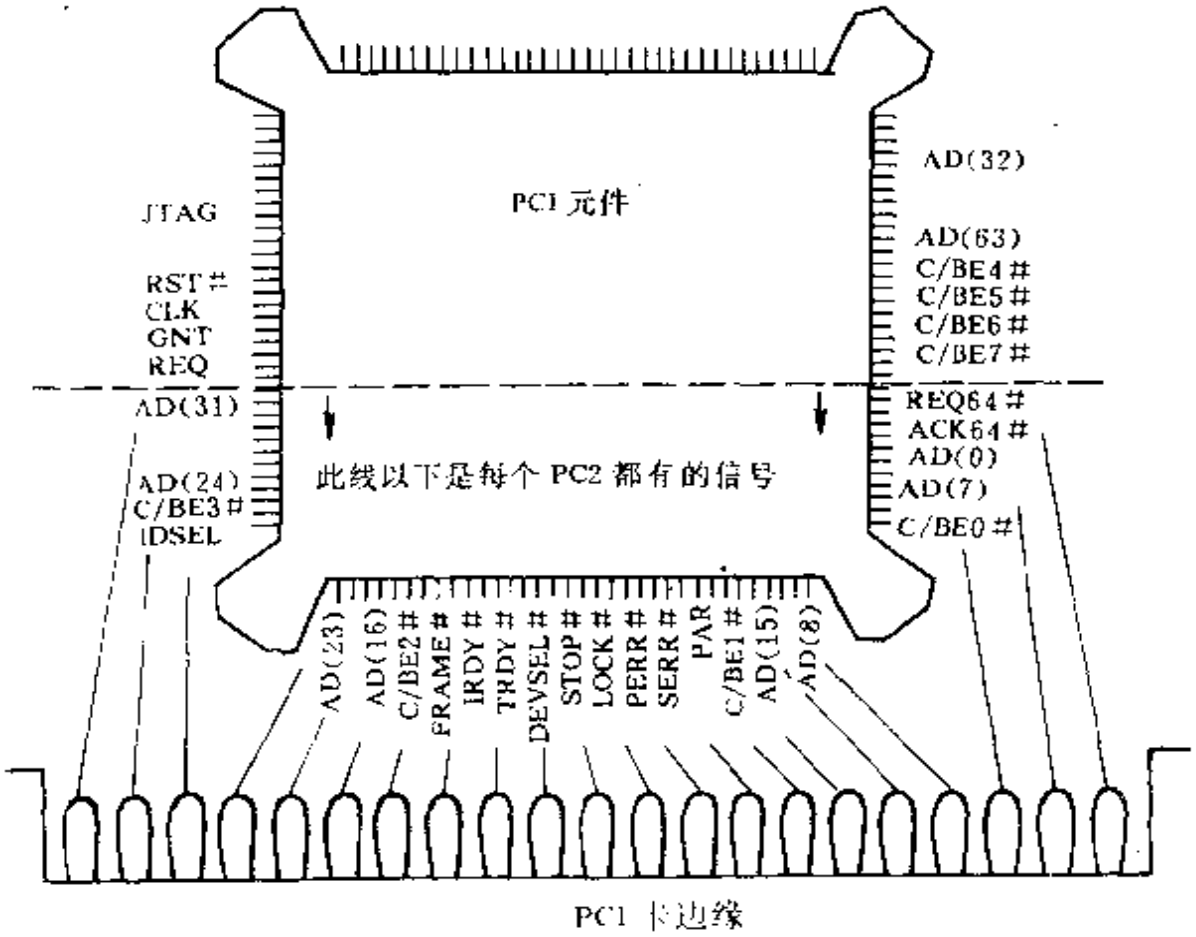


图 4.8　PQFP 封装的 PCI 元件引脚分配建议

64 位的信号应按照它们在 64 位连接器上出现的顺序，以反时针方向沿着元件边继续排列。

IDSEL 输入端尽量靠近 AD[31 :: 16]，以使 IDSEL 与其中的一条连线时所造成的附加负载小一些。该引脚上的允许电容量较

小，可能制约它在封装中的位置安排。

4.3 系统(母板)技术指标

4.3.1 时钟相位偏移

在系统板上，允许的最大时钟相位偏移是 2 ns，这个值是在任意两个元件之间测量的，而不是在连接器之间。该限制不仅适用于门限点，也适用于时钟波形上位于开关范围内的所有各点。有关波形及参数见图 4.9 及表 4.8。

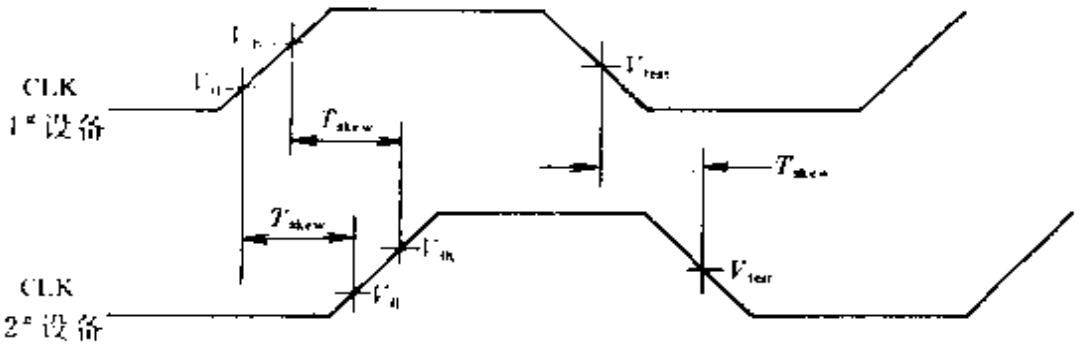


图 4.9 时钟相位偏移波形

表 4.8 时钟相位偏移参数

名 称	5 V 信号环境	3.3 V 信号环境	单位
V_{test}	1.5	$0.4V_{cc}$	V
T_{skew}	2(max)	2(max)	ns

为了保证时钟相位偏移的准确性，系统产品的开发者必须考虑扩展板上的时钟分配。

4.3.2 复位信号

PCI 复位信号 RST # 的发出和撤消均与系统时钟 CLK 不同

步。RST#信号的上升沿和下降沿在输入开关范围内必须是单调的(也就是不能出现跳动现象)。必要时，PCI 总线也可能使 RST#和 CLK 同步。

除参数 T_{fail}之外，其它有关复位时的时间参数已在表 4.6 中给出。参数 T_{fail}表示系统对电源电压失常时的反应。如果电源电压失常，杂散二极管通路有可能将导通的输出缓冲短路，因此，RST#应在电源失效时建立起来，以便将输出缓冲置为浮动状态。参数 T_{fail}的具体取值是从下面两种情况中选择最小的一个：

(1) 从电源电压偏离规定指标(超过指定的偏离范围 500 mV 以上)时算起 500 ns(最大)。

(2) 从 5 V 电源电压降到比 3.3 V 电源电压还要低 300 mV 时算起 100 ns(最大)。

在系统上电或电源失效时，系统都必须发出 RST#信号。而 PCI 元件的复位，要等到 RST#发出之后；T_{rst}和 $T_{rst-clk}$两个参数都满足要求时才算完成。图 4.10 给出了 RST#有关的时序。

从图中可以看出，REQ64#信号在复位期间用来区分是否连接有 64 位数据通路的设备。REQ64#作为总线连到母板上包括 PCI 连接器插槽在内的所有支持 64 位数据通路的设备上，并在母板上有其相应的上拉电阻。在不支持 64 位数据通路的 PCI 扩展插槽上，REQ64#不作为总线也不与它相连，但它有自己单独的上拉电阻。在 RST#信号有效期间，中央资源必须将 REQ64#驱动为低，并符合技术指标中规定的时间关系。凡是在复位期间发现 REQ64#有效的设备就被连到 64 位数据通路上，否则就不连。

复位期间，REQ64#的建立和撤消时间都是以 RST#的撤消边作为参考点，并且与 CLK 异步。

4.3.3 上拉电阻

PCI 总线的控制信号都要求有上拉电阻，这是为了保证它们

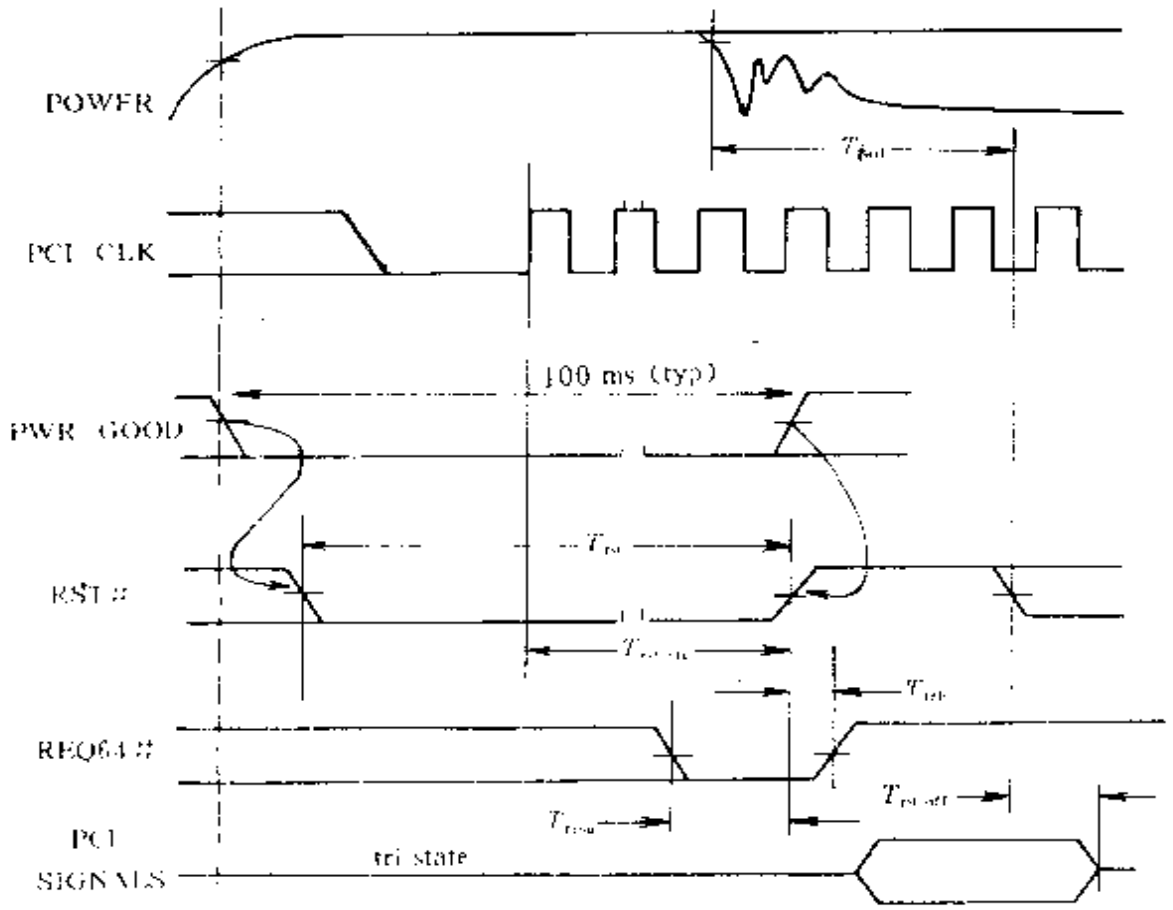


图 4.10 复位信号时序

在没有设备驱动总线的情况下仍具有稳定的值。此类信号有：FRAME #、TRDY #、IRDY #、DEVSEL #、STOP #、SERR #、PERR #以及选用的 LOCK #、REQ64 #和 ACK64 #。点到点和 32 位共享信号不需要上拉电阻，总线的停靠即可保证它们的稳定。对于 64 位数据通路的扩展信号 AD[63 :: 32]、C/BE[7 :: 4]# 及 PAQ64 #，如果连接就必须接上拉电阻，否则，元件本身要处理输入浮动问题。

下面给出计算最大上拉电阻和最小上拉电阻的公式：

$R_{max}=[V_{cc}(min)-V_x]/[负载数\times I_{il}]$

$R_{min}=[V_{cc}(max)-V_{ol}]/[I_{ol}+16\times I_{il}]$

其中：16 为最大负载数；V_x 在 5 V 信号环境下为 2.7 V，而在

3.3 V信号环境下为 0.7 V_{CC}；V_{ol}和 I_{ol}、I_{il}见前述的技术指标。

R_{max}主要由负载数来决定，而 R_{min}主要由直流低输出电流 I_{ol}决定，负载数量的影响是次要的。指标中给出的 R 最小值是以 16 个负载进行计算的(应该是最坏情况)，R 的典型值是基于 10 个负载来计算的，R 的最大值没有给出具体值，实际上取一个系统中最小负载数会使该值为最高值。具体指标见表 4.9。

表 4.9　上拉电阻的指标

信号环境	R_{min}/Ω	R 典型值/Ω	R_{max}
5 V	963	2.7 k(精度 10%)	见公式
3 V	2.42 k	8.2 k(精度 10%)	见公式

4.3.4　电源

1. 电源要求

所有的 PCI 连接器都需要四条电源线：+5 V、+3.3 V、+12 V、和 −12 V。其中 3.3 V 由需要该电压的扩展板提供，+5 V和±12 V 必须由系统提供。电源供电时的启动和停用没有明确的时序，当超过安全极限时，RST＃信号发生作用，以保障系统安全。

实现 3.3 V 信号环境的系统需要全部四条电源线，其电流要求见表 4.10。实现 5 V 信号环境的系统可以随系统提供 3.3 V 电源，也可以仅仅提供一种事后增加的措施，待需要支持 3.3 V 扩展板时再加上去。但无论如何，其它三种电源是每个系统上必备的。

在表 4.10 中也给出了每个连接器上对±12 V 电源线上的电流要求。对于 5 V 和 3.3 V 电源没有提出每个连接器上要求多大电流，而是由系统决定的。系统只为 PCI 扩展板提供一个总的电源容量，可以用仲裁方法在连接器之间分配。连接器上的 PSNTn

#引脚使系统可以估计每个板上的电源需求，并且确定当前安装的配置是否在允许的电源总容量之内。

表 4.10 电源允许范围

电 源 线	扩展卡(短卡与长卡)
5 V(±5%)	5 A_{max}(系统决定)
3.3 V(±0.3 V)	7.6 A_{max}(系统决定)
12 V(±5%)	500 mA
−12 V(±10%)	100 mA

2. 电源的顺序

对于上述四种电源，其加电或关电顺序没有特殊规定，可以按任意顺序进行。每当加电时或者 5 V(3.3 V)电源不符合指标要求时，系统必须发出 RST# 信号。在复位期间所有的 PCI 信号均被驱动到一个“安全的”状态。

3. 电源的退耦

所有的电源平面都必须对地去耦合，以便合理地处理开关电流的冲击。关于去耦合的具体方法在 PCI 规范中未作详细规定。

PCI 连接器上的 3.3 V 引脚(即使在实际中未提供电流)应提供一个交流回路，而且在母板上必须连到一起，同时最好是连到一个 3.3 V 的电源平面上。这时的对地去耦应符合高频信号技术的要求。为此，建议在 3.3 V 平面上均匀排列 12 个高速电容器，容量为 0.01 μF。

4.3.5 系统时标限制

在计算整个 PCI 负载模型时，必须十分小心或注意扩展板上最大走线长度和负载情况。同时，必须遵守在扩展板上最大引脚电容为 10 pF 的约束条件。而在主板上可以采用实际的引脚电容量。

整个时钟周期可分为四个阶段：输入建立时间(T_{su})、有效输出延迟(T_{val})、总的时钟相位偏移(T_{skew})和总线传输时间(T_{prop})。前两个参数值根据元件指标定义，后两个是系统参数。T_{prop}被定义为 10 ns，但是若降低时钟相位偏移则它可以增加到 11 ns，也就是说，T_{prop}加上 T_{skew}不能超过 12 ns，至于 T_{skew}无论如何不得超过 2 ns。图 4.11 给出了对 T_{prop}的测量波形，它开始于输出缓冲已经跨越门限点之后或者说在驱动 50 pF 负载的波形上，结束于最慢的输入跨越 V_{th}或 V_{il}之时。在计算确定这个时间点时必须小心。注意：输入缓冲时标的测量是在一定过载量的前提下进行的，这就需要保证输入缓冲的时间。

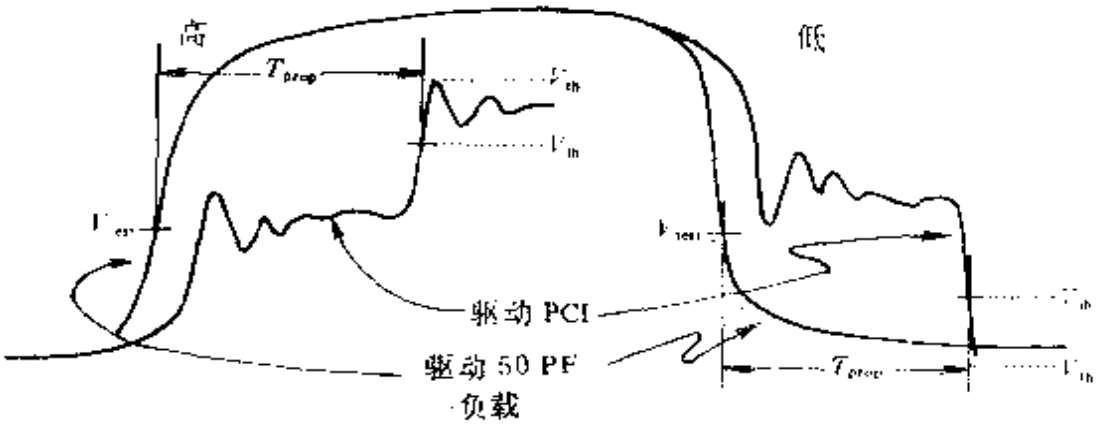


图 4.11　T_{prop}的测量波形

4.3.6　系统的物理要求

连接器上电源引脚的排列是从方便母板四层布线的目的出发的。有时可能需要在 5 V 电源平面上开出一块 3.3 V 的区域，这就要利用分裂的电源层。该区域被连到所有 3.3 V PCI 连接器引脚，同时还要连接到电源分配用的连接器引脚。尽管这些做法在技术上是标准的，但当直接在这些分裂的电源层上进行高速信号走线时有可能引起信号的完整性问题。电源层的裂变会造成阻抗

的不连续性。

建议对信号线的布局采用如下方法：在两个电源层上不要跨接高速信号(如 33 MHz)，要么将其全部布于 3.3 V 平面上，要么完全在 5 V 平面上安排它的走线。如果有的信号线必须跨越两个区域，可以将它放到板的反面使之对地形成参照，因为地线平面是整块的。如果这个办法行不通，而信号又必须跨在电源平面层的裂缝上时，就应该把两个电源平面用电容器耦合在一起(即 5 V 平面直接耦合到 3.3 V 平面)，并且每四条跨过裂缝的线就要用一个 0.01 μF 的高速电容，同时应保证该电容器的位置距离跨越点不得超过 0.25 英寸。①

上述建议不适合于低速的 ISA 总线信号。

关于母板的光板阻抗指标，在总线规范中未作具体规定。但对于系统产品的开发者应遵循下列两条重要原则：

(1) 信号线的长度和速度，必须保证能够使总线信号在 10 ns 的传播延迟时间内在总线上往返一个来回。

(2) 在总线的任何驱动点上，其负载阻抗必须能使一个 PCI 输出信号依靠一次反射便可达到输入信号的指标要求。对扩展板也是一样。

如果由于配置上的原因而使上述约束不能实现，信号的往返时间就会增加，此时可以对操作频率做一些折衷来进行弥补。

4.3.7 连接器

1. 从 5 V 到 3.3 V 的过渡

提供从 5 V 到 3.3 V 快速而容易的转换技术是 PCI 总线电气规范的一个目标。为此，PCI 定义了两种扩展板连接器，一种是基于 5 V 信号环境的，另一种是基于 3.3 V 信号环境的。同时还

① 1 英寸＝0.0254 m

定义了三种电气类型的扩展板，分别是 5 V 板、3.3 V 板和通用板(即可用于 5 V 环境又可用于 3.3 V 环境)。在连接器和板上都有相应的定位键位以防止将一个板插入不适当的位置。具体见图 4.12所示。

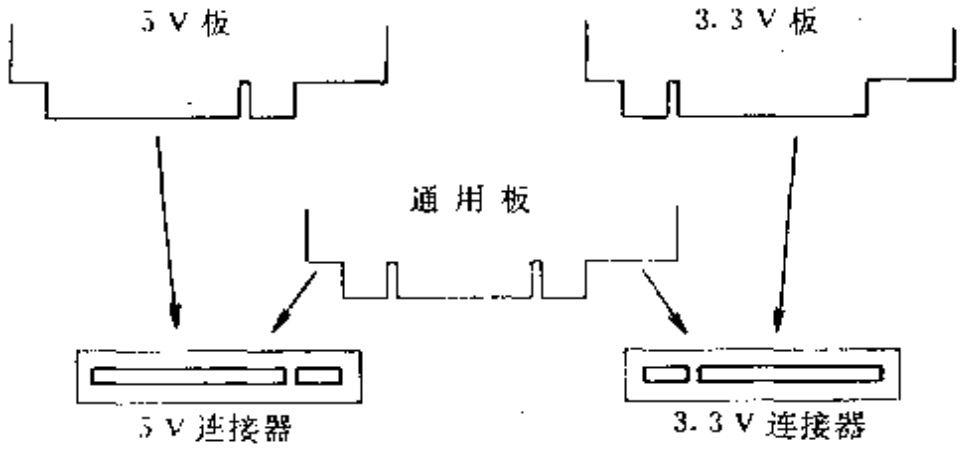


图 4.12　PCI 板与连接器

母板(包括连接器)为总线规定了信号环境，不是 5 V 便是 3.3 V。5 V 的板是按 5 V 信号环境而设计的，只能插于 5 V 的连接器；同样道理，3.3 V 板是按照 3.3 V 信号环境而设计，只能工作于 3.3 V 环境。但是通用板能够检测出当前所用的信号环境属于哪一种，并且能够自适应该环境，所以它可以随便插入任意一种连接器中。三种类型的板都具有到 5 V 和 3.3 V 电源的连接线。至于板上的元件有可能包含 5 V 的，也有可能包含 3.3 V 的，或者两种都有。各种类型板子之间的区别在于它们所用的信号协议，而不是电源线的连接，也不是其上所包含的元件技术。

通用板上的 PCI 元件必须使用符合 5 V 或者 3.3 V 信号环境的 I/O 缓冲。这种符合双重环境要求的缓冲有多种实现方法。一般来说，总是希望它是一个具有双重电压的缓冲，也就是说，利用两种电源的任何一种都可以工作。它们的电源应该来自于标有"I/O"的电源引脚上，在 PCI 连接器上，这样的电源引脚总是与所

用信号环境相对应的电源汇流排相连接。在 5 V 信号环境下这些缓冲将从 5 V 电源上得到供电，而当同一块板插入 3.3 V 连接器上时，这些缓冲便由 3.3 V 电源供电。这样一来，就可以使通用板符合任何一种信号环境。

这种过渡的意图是把 5 V 元件技术放在 3.3 V 信号环境下，而不是要求 3.3 V 元件技术能在 5 V 信号环境下工作。虽然后者可以实现，但比较困难，价格较贵，尤其是在一个无端接的模块总线环境下。最好的做法是将 5 V 元件放到 3.3 V 信号环境下工作，这既容易实现又不增加任何成本，而且在信号性能方面有一定好处。

由于第一个 PCI 元件只有 5 V 的 I/O 缓冲，所以 5 V 的板子在初期是必要的。但是所有按照 5 V 技术而设计的新元件，都应该采用双电压缓冲，而且通用板也应该如此。这样，基于 5 V 元件技术的扩展板就可以用在 5 V 和 3.3 V 两种系统中，从而使5 V 系统向 3.3 V 系统的过渡成为可能。

总的来讲，短期内将只用 5 V 的板子和连接器，但从长远看，应立足于 3.3 V 连接器，使用 5 V 元件但采用双电压缓冲的通用板和使用 3.3 V 元件的 3.3 V 板。这之间的过渡主要依靠通用板，而在整个过渡中最重要的是先从 5 V 板过渡到通用板。

2. 连接器的引脚排列

PCI 连接器包含了为 PCI 元件定义的全部信号，另外还有与连接器本身有关的两个信号 PRSNT1# 和 PRSNT2#，有关这两个信号的进一步说明见 4.4.1。由于这两个信号或者其中之一还要提供一个交流回路，所以在母板上必须单独对它们用 0.01 μF 的高速电容对地退耦。它们不能在母板上接成总线，也不能以其它方式互连。它们在母板上的用途是可选的。如果母板的设计中采用了这两个引脚来了解扩展板的情况，则每个引脚在母板上都必须有一个适当的上拉电阻(约 5 kΩ)。

母板上的PCI连接器引脚分配见表4.11。表中给出了5 V和3.3 V两种信号环境下的引脚分配。表中标有"+5 $V^{(I/O)}$"和"3.3 $V^{(I/O)}$"的引脚是专用的电源引脚，用以驱动通用板上的PCI信号，在母板上它们分别连到+5 V和+3.3 V电源层。

对于REQ64#和ACK64#在母板上的连接应特别注意，为它们而设的上拉电阻应放置在母板上而不是在扩展板上，这是为了保证不会在这两个引脚上出现多个上拉值。64位的扩展连接器在母板上不一定必备，所以将这些信号与32位连接器部分安排在一起。所有提供64位数据通路的连接器，应把这些引脚分别用各自的上拉电阻以总线方式连在一起。而对于32位连接器，这些信号线是开路的，不能连在一起，而且要有各自的上拉电阻，以便64位板子插入32位槽上时也能够正常工作。

表4.11 PCI连接器引脚分配

引脚	5 V信号环境		3.3 V信号环境		备注
	B面	A面	B面	A面	
1	−12 V	TRST#	−12 V	TRST#	32位连接器开始
2	TCK	+12 V	TCK	+12 V	
3	Ground	TMS	Ground	TMS	
4	TDO	TDI	TDO	TDI	
5	+5 V	+5 V	+5 V	+5 V	
6	+5 V	INTA#	+5 V	INTA#	
7	INTB#	INTC#	INTB#	INTC#	
8	INTD#	+5 V	INTD#	+5 V	
9	PRSNT1#	Reserved	PRSNT1#	Reserved	
10	Reserved	+5 V(I/O)	Reserved	+3.3 V(I/O)	
11	PRSNT2#	Reserved	PRSNT2#	Reserved	
12	Ground	Ground	连接器键位		3.3 V键位
13	Ground	Ground			
14	Reserved	Reserved	Reserved	Reserved	
15	Ground	RST#	Ground	RST#	

续表(一)

引脚	5 V 信号环境		3.3 V 信号环境		备注
	B 面	A 面	B 面	A 面	
16	CLK	+5 V(I/O)	CLK	+3.3 V(I/O)	
17	Ground	GNT#	Ground	GNT#	
18	REQ#	Ground	REQ#	Ground	
19	+5 V(I/O)	Reserved	+3.3 V(I/O)	Reserved	
20	AD[31]	AD[30]	AD[31]	AD[30]	
21	AD[29]	+3.3 V	AD[29]	+3.3 V	
22	Ground	AD[28]	Ground	AD[28]	
23	AD[27]	AD[26]	AD[27]	AD[26]	
24	AD[25]	Ground	AD[25]	Ground	
25	+3.3 V	AD[24]	+3.3 V	AD[24]	
26	C/BE[3]#	IDSEL	C/BE[3]#	IDSEL	
27	AD[23]	+3.3 V	AD[23]	+3.3 V	
28	Ground	AD[22]	Ground	AD[22]	
29	AD[21]	AD[20]	AD[21]	AD[20]	
30	AD[19]	Ground	AD[19]	Ground	
31	+3.3 V	AD[18]	+3.3 V	AD[18]	
32	AD[17]	AD[16]	AD[17]	AD[16]	
33	C/BE[2]#	+3.3 V	C/BE[2]#	+3.3 V	
34	Ground	FRAME#	Ground	FRAME#	
35	IRDY#	Ground	IRDY#	Ground	
36	+3.3 V	TRDY#	+3.3 V	TRDY#	
37	DEVSEL#	Ground	DEVSEL#	Ground	
38	Ground	STOP#	Ground	STOP#	
39	LOCK#	+3.3 V	LOCK#	3.3 V	
40	PERR#	SDONE	PERR#	SDONE	
41	+3.3 V	SBO#	+3.3 V	SBO#	
42	SERR#	Ground	SERR#	Ground	
43	+3.3 V	PAR	+3.3 V	PAR	

续表(二)

引脚	5 V 信号环境		3.3 V 信号环境		备注
	B 面	A 面	B 面	A 面	
44	C/BE[1]#	AD[15]	C/BE[1]#	AD[15]	
45	AD[14]	+3.3 V	AD[14]	+3.3 V	
46	Ground	AD[13]	Ground	AD[13]	
47	AD[12]	AD[11]	AD[12]	AD[11]	
48	AD[10]	Ground	AD[10]	Ground	
49	Ground	AD[09]	Ground	AD[09]	
50	键 位		Ground	Ground	5 V 键位
51			Ground	Ground	
52	AD[08]	C/BE[0]#	AD[08]	C/BE[0]#	
53	AD[07]	+3.3 V	AD[07]	+3.3 V	
54	+3.3 V	AD[06]	+3.3 V	AD[06]	
55	AD[05]	AD[04]	AD[05]	AD[04]	
56	AD[03]	Ground	AD[03]	Ground	
57	Ground	AD[02]	Ground	AD[02]	
58	AD[01]	AD[00]	AD[01]	AD[00]	
59	+5 V(I/O)	+5 V(I/O)	+3.3 V(I/O)	+3.3 V(I/O)	
60	ACK64#	REQ64#	ACK64#	REQ64#	
61	+5 V	+5 V	+5 V	+5 V	
62	+5 V	+5 V	+5 V	+5 V	32 位终
XX	键 位		键 位		64 空
63	Reserved	Ground	Reserved	Ground	64 位始
64	Ground	C/BE[7]#	Ground	C/BE[7]#	
65	C/BE[6]#	C/BE[5]#	C/BE[6]#	C/BE[5]#	
66	C/BE[4]#	+5 V(I/O)	C/BE[4]#	+3.3 V(I/O)	
67	Ground	PAR64	Ground	PAR64	
68	AD[63]	AD[62]	AD[63]	AD[62]	
69	AD[61]	Ground	AD[61]	Ground	

续表(三)

引脚	5 V 信号环境		3.3 V 信号环境		备注
	B 面	A 面	B 面	A 面	
70	+5 V(I/O)	AD[60]	+3.3 V(I/O)	AD[60]	
71	AD[59]	AD[58]	AD[59]	AD[58]	
72	AD[57]	Ground	AD[57]	Ground	
73	Ground	AD[56]	Ground	AD[56]	
74	AD[55]	AD[54]	AD[55]	AD[54]	
75	AD[53]	+5 V(I/O)	AD[53]	+3.3 V(I/O)	
76	Ground	AD[52]	Ground	AD[52]	
77	AD[51]	AD[50]	AD[51]	AD[50]	
78	AD[49]	Ground	AD[49]	Ground	
79	+5 V(I/O)	AD[48]	+3.3 V(I/O)	AD[48]	
80	AD[47]	AD[46]	AD[47]	AD[46]	
81	AD[45]	Ground	AD[45]	Ground	
82	Ground	AD[44]	Ground	AD[44]	
83	AD[43]	AD[42]	AD[43]	AD[42]	
84	AD[41]	+5 V(I/O)	AD[41]	+3.3 V(I/O)	
85	Ground	AD[40]	Ground	AD[40]	
86	AD[39]	AD[38]	AD[39]	AD[38]	
87	AD[37]	Ground	AD[37]	Ground	
88	+5 V(I/O)	AD[36]	+3.3 V(I/O)	AD[36]	
89	AD[35]	AD[34]	AD[35]	AD[34]	
90	AD[33]	Ground	AD[33]	Ground	
91	Ground	AD[32]	Ground	AD[32]	
92	Reserved	Reserved	Reserved	Reserved	
93	Reserved	Ground	Reserved	Ground	
94	Ground	Reserved	Ground	Reserved	64 位终

4.4 扩展板技术指标

4.4.1 扩展板上的引脚分配

PCI 连接器包含了为 PCI 元件定义的全部信号线。另外还有两个引脚是与连接器本身有关的，即 PRSNT1＃和 PRSNT2＃。其用途有两个：其一用来表明槽位上实际存在一块板；其二是提供该板对电源要求的有关信息。表 4.12 给出了扩展板上 PRSNT＃引脚的设置情况。

对于扩展板，必须指明板上电源功耗的最大值，而系统无论在 5 V 或 3.3 V 电源线上都必须能满足这个功耗。另外，如果扩展板是可配置的(如有存储器扩充插座等)，引脚的接法必须表示出最大配置下的总功耗要求。

表 4.12 表示信号的定义

PRSNT1＃	PRSNT2＃	扩展板配置
开路	开路	不存在扩展板
地	开路	有扩展板，最大功耗为 25 W
开路	地	有扩展板，最大功耗为 15 W
地	地	有扩展板，最大功耗为 7.5 W

不实现 JTAG 边界扫描的板子，必须把引脚 TDI 和 TDO 连接起来，以免扫描链不能断开。

表 4.13 给出了 PCI 板上的引脚分配情况。

表中标着“$V^{(I/O)}$”的引脚是专用的电源引脚，用来在通用板上区分和驱动 PCI 信号线。在通用板上，PCI 元件的 I/O 缓冲必须从这些专门的电源引脚供电，而不是从＋5 V 或＋3.3 V 电源引脚供电。

表 4.14 摘要给出了 32 位板和 64 位板（增加数量）的引脚分配。

表 4.13 PCI 板引脚分配表

引脚	5 V 板		通用板		3.3 V 板		备注
	B 面	A 面	B 面	A 面	B 面	A 面	
1	−12 V	TRST#	−12 V	TRST#	12 V	TRST#	32 位始
2	TCK	+12 V	TCK	+12 V	TCK	+12 V	
3	Ground	TMS	Ground	TMS	Ground	TMS	
4	TDO	TDI	TDO	TDI	TDO	TDI	
5	+5 V	+5 V	+5 V	+5 V	+5 V	+5 V	
6	+5 V	INTA#	+5 V	INTA#	+5 V	INTA#	
7	INTB#	INTC#	INTB#	INTC#	INTB#	INTC#	
8	INTD#	+5 V	INTD#	+5 V	INTD#	+5 V	
9	PRSNT1#	Reserved	PRSNT1#	Reserved	PRSNT1#	Reserved	
10	Reserved	+5 V	Reserved	$+V_{I/O}$	Reserved	+3.3 V	
11	PRSNT2#	Reserved	PRSNT2#	Reserved	PRSNT2#	Reserved	
12	Ground	Ground	键 位		键 位		3.3 V 键位
13	Ground	Ground	键 位		键 位		3.3 V 键位
14	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	
15	Ground	RST#	Ground	RST#	Ground	RST#	
16	CLK	+5 V	CLK	$+V_{I/O}$	CLK	+3.3 V	
17	Ground	GNT#	Ground	GNT#	Ground	GNT#	
18	REQ#	Ground	REQ#	Ground	REQ#	Ground	
19	+5 V	Reserved	$+V_{I/O}$	Reserved	+3.3 V	Reserved	
20	AD[31]	AD[30]	AD[31]	AD[30]	AD[31]	AD[30]	
21	AD[29]	+3.3 V	AD[29]	+3.3 V	AD[29]	+3.3 V	
22	Ground	AD[28]	Ground	AD[28]	Ground	AD[28]	

续表(一)

引脚	5 V板		通用板		3.3 V板		备注
	B面	A面	B面	A面	B面	A面	
23	AD[27]	AD[26]	AD[27]	AD[26]	AD[27]	AD[26]	
24	AD[25]	Ground	AD[25]	Ground	AD[25]	Ground	
25	+3.3 V	AD[24]	+3.3 V	AD[24]	+3.3 V	AD[24]	
26	C/BE[3]#	IDSEL	C/BE[3]#	IDSEL	C/BE[3]#	IDSEL	
27	AD[23]	+3.3 V	AD[23]	+3.3 V	AD[23]	+3.3 V	
28	Ground	AD[22]	Ground	AD[22]	Ground	AD[22]	
29	AD[21]	AD[20]	AD[21]	AD[20]	AD[21]	AD[20]	
30	AD[19]	Ground	AD[19]	Ground	AD[19]	Ground	
31	+3.3 V	AD[18]	+3.3 V	AD[18]	+3.3 V	AD[18]	
32	AD[17]	AD[16]	AD[17]	AD[16]	AD[17]	AD[16]	
33	C/BE[2]#	+3.3 V	C/BE[2]#	+3.3 V	C/BE[2]#	+3.3 V	
34	Ground	FRAME#	Ground	FRAME#	Ground	FRAME#	
35	IRDY#	Ground	IRDY#	Ground	IRDY#	Ground	
36	+3.3 V	TRDY#	+3.3 V	TRDY#	+3.3 V	TRDY#	
37	DEVSEL#	Ground	DEVSEL#	Ground	DEVSEL#	Ground	
38	Ground	STOP#	Ground	STOP#	Ground	STOP#	
39	LOCK#	+3.3 V	LOCK#	+3.3 V	LOCK#	+3.3 V	
40	PERR#	SDONE	PERR#	SDONE	PERR#	SDONE	
41	+3.3 V	SBO#	+3.3 V	SBO#	+3.3 V	SBO#	
42	SERR#	Ground	SERR#	Ground	SERR#	Ground	
43	+3.3 V	PAR	+3.3 V	PAR	+3.3 V	PAR	
44	C/BE[1]#	AD[15]	C/BE[1]#	AD[15]	C/BE[1]#	AD[15]	

续表(二)

引脚	5 V 板		通用板		3.3 V 板		备注
	B 面	A 面	B 面	A 面	B 面	A 面	
45	AD[14]	+3.3 V	AD[14]	+3.3 V	AD[14]	+3.3 V	
46	Ground	AD[13]	Ground	AD[13]	Ground	AD[13]	
47	AD[12]	AD[11]	AD[12]	AD[11]	AD[12]	AD[11]	
48	AD[10]	Ground	AD[10]	Ground	AD[10]	Ground	
49	Ground	AD[09]	Ground	AD[09]	Ground	AD[09]	
50	键位		键位		Ground	Ground	5V 键位
51	键位		键位		Ground	Ground	5V 键位
52	AD[08]	C/BE[0]#	AD[08]	C/BE[0]#	AD[08]	C/BE[0]#	
53	AD[07]	+3.3 V	AD[07]	+3.3 V	AD[07]	+3.3 V	
54	+3.3 V	AD[06]	+3.3 V	AD[06]	+3.3 V	AD[06]	
55	AD[05]	AD[04]	AD[05]	AD[04]	AD[05]	AD[04]	
56	AD[03]	Ground	AD[03]	Ground	AD[03]	Ground	
57	Ground	AD[02]	Ground	AD[02]	Ground	AD[02]	
58	AD[01]	AD[00]	AD[01]	AD[00]	AD[01]	AD[00]	
59	+5 V	+5 V	$+V_{I/O}$	$+V_{I/O}$	+3.3 V	+3.3 V	
60	ACK64#	REQ64#	ACK64#	REQ64#	ACK64#	REQ64#	
61	+5 V	+5 V	+5 V	+5 V	+5 V	+5 V	
62	+5 V	+5 V	+5 V	+5 V	+5 V	+5 V	32 位终
	键位		键位		键位		64 空
	键位		键位		键位		64 空
63	Reserved	Ground	Reserved	Ground	Reserved	Ground	
64	Ground	C/BE[7]#	Ground	C/BE[7]#	Ground	C/BE[7]#	64 位始

续表(三)

引脚	5 V 板		通 用 板		3.3 V 板		备 注
	B 面	A 面	B 面	A 面	B 面	A 面	
65	C/BE[6]#	C/BE[5]#	C/BE[6]#	C/BE[5]#	C/BE[6]#	C/BE[5]#	
66	C/BE[4]#	+5 V	C/BE[4]#	$+V_{I/O}$	C/BE[4]#	+3.3 V	
67	Ground	PAR64	Ground	PAR64	Ground	PAR64	
68	AD[63]	AD[62]	AD[63]	AD[62]	AD[63]	AD[62]	
69	AD[61]	Ground	AD[61]	Ground	AD[61]	Ground	
70	+5 V	AD[60]	$+V_{I/O}$	AD[60]	+3.3 V	AD[60]	
71	AD[59]	AD[58]	AD[59]	AD[58]	AD[59]	AD[58]	
72	AD[57]	Ground	AD[57]	Ground	AD[57]	Ground	
73	Ground	AD[56]	Ground	AD[56]	Ground	AD[56]	
74	AD[55]	AD[54]	AD[55]	AD[54]	AD[55]	AD[54]	
75	AD[53]	+5 V	AD[53]	$+V_{I/O}$	AD[53]	+3.3 V	
76	Ground	AD[52]	Ground	AD[52]	Ground	AD[52]	
77	AD[51]	AD[50]	AD[51]	AD[50]	AD[51]	AD[50]	
78	AD[49]	Ground	AD[49]	Ground	AD[49]	Ground	
79	+5 V	AD[48]	$+V_{I/O}$	AD[48]	+ 3.3V	AD[48]	
80	AD[47]	AD[46]	AD[47]	AD[46]	AD[47]	AD[46]	

续表(四)

引脚	5 V板		通用板		3.3 V板		备注
	B面	A面	B面	A面	B面	A面	
81	AD[45]	Ground	AD[45]	Ground	AD[45]	Ground	
82	Ground	AD[44]	Ground	AD[44]	Ground	AD[44]	
83	AD[43]	AD[42]	AD[43]	AD[42]	AD[43]	AD[42]	
84	AD[41]	+5 V	AD[41]	$+V_{I/O}$	AD[41]	+3.3 V	
85	Ground	AD[40]	Ground	AD[40]	Ground	AD[40]	
86	AD[39]	AD[38]	AD[39]	AD[38]	AD[39]	AD[38]	
87	AD[37]	Ground	AD[37]	Ground	AD[37]	Ground	
88	+5 V	AD[36]	$+V_{I/O}$	AD[36]	+3.3 V	AD[36]	
89	AD[35]	AD[34]	AD[35]	AD[34]	AD[35]	AD[34]	
90	AD[33]	Ground	AD[33]	Ground	AD[33]	Ground	
91	Ground	AD[32]	Ground	AD[32]	Ground	AD[32]	
92	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	
93	Reserved	Ground	Reserved	Ground	Reserved	Ground	
94	Ground	Reserved	Ground	Reserved	Ground	Reserved	64位终

表 4.14　32 位/64 位板引脚数摘要

引脚类型	5 V 板	通用板	3.3 V 板
地	22/16	18/16	22/16
+5 V	13/6	8/0	8/0
+3.3 V	12/0	12/0	17/6
I/O 电源	0/0	5/6	0/0
保留	6/5	6/5	6/5

表中"/"左边为 32 位板的引脚数，右边为 64 位板的引脚增加数。

4.4.2　电源要求

1. 退耦

在典型条件下，V_{cc}层到地线层之间的电容将为连接器上的V_{cc}引脚提供适量的退耦作用。从连接器根部到 V_{cc}/GND 层的引线孔之间最大走线长度限制在 0.25 英寸以下，线宽不小于 0.02 英寸。但是在通用板上，在 I/O 缓冲电源线到地线层之间没有电容，需要外加退耦电容。在标有"$+V_{(I/O)}$"的引脚上，应当用平均容量为 0.047 μF 的电容将它耦合到地。

此外，所有 3.3 V 引脚和未用的 5 V 引脚，都需要用下列方法将它们耦合到地，以保证它们继续发挥有效的交流参考作用。

(1) 每个 V_{cc}引脚上必须有退耦电容，且容量的平均值至少为 0.01 μF。

(2) 从引脚根部到电容器焊盘的走线长度不大于 0.25 英寸，线宽至少为 0.02 英寸。

(3) 多个引脚可以共用一个电容并且参与共用的引脚数不受限制，但必须满足上述(1)、(2)两点的要求。

2. 关于电源的功耗

任何 PCI 扩展板上所允许的最大功耗为 25 W，该值是指来自连接器上四条电源线功耗的总和。在最坏情况下，全部 25 W 功耗有可能取自单一的+5 V 或+3.3 V 电源。

估计许多系统不会为每根电源线都提供足 25 W 的连接器容量，这是因为大多数板子通常耗电要比 25 W 小得多。为此，在可能的情况下，耗电多于 10 W 的板子应设法维持在一种耗电不大于 10 W 的节电状态，并且每次复位后就应处于此状态。在该状态下，板子必须提供对它的 PCI 配置空间的完全访问，同时必须完成所要求的自举功能。例如一个视频板上基本的文本方式。至于板上的其它功能，必要时可以禁止。

上述的电源节电状态有多种方法可以实现，例如：

(1) 可以降低板上的时钟频率，这样做会降低性能但不限制功能。

(2) 可以用一个场效应管(FET)将一部分连到非关键部位的电源供电关闭，但这样可能会减少功能。

板子的驱动程序启动之后，便可以把该板设置为全供电、全功能/性能状态，所用的机构因设备不同而各异。在先进的电源管理系统中，在把整个板子的功能全部开动之前，可要求设备驱动程序报告它的功耗需求情况，以使系统确定有无足够的电源容量来满足当前配置中的所有板子。驱动程序必须能够精确地确定它的板子在当前配置下所需的功率要求以及从哪个电源线上取得。

4.4.3 物理要求

1. 关于走线长度的限制

从扩展卡的边缘连接器到 PCI 元件的引脚之间，其走线长度有如下限制：

(1) 在32位和64位板上，所有32位接口信号的最大走线长度为1.5英寸。

(2) 在所有的64位板上，用于64位扩展的附加信号线走线长度最大为2英寸。

(3) 无论是32位板还是64位板，其上的时钟CLK信号走线长度为2.5±0.1英寸，而且只能连到一个负载上。

2. 关于布线及布局要求

连接器上的电源引脚分配已考虑了尽量方便四层板的布线及布局。有时需要将电源层分成两部分用于+5 V和+3.3 V，即所谓的"分裂的电源层"。尽管这是一种标准技术，但是如果直接在分裂的电源层上有高速信号线跨过，有可能发生信号完整性问题。电源层的断裂处使得信号线的交流回路受阻，从而造成阻抗的不连续性。

建议在信号布局中尽量不要让高速信号跨在两个电源平面上，应该将它们全部布置在3.3 V平面上方，或者全部布在5 V平面上方。如果有的信号线不得已要跨越两个区域，可以将它放在板的另一面，使它在地线平面上方走线，因为地线平面无裂缝。如果有的信号无论采用什么办法都不能不让它跨在两个电源层平面的裂缝上，这时就应该将两个电源层平面用电容器耦合在一起，每四条跨过的信号线用一个0.01 μF的高速电容器，并且该电容器的位置距跨越点不得超过0.25英寸。

3. 关于阻抗的要求

扩展板上共享的PCI信号线的无负载特性阻抗应当控制在60～100 Ω范围内。线上的信号速率必须在150～190 PS/s范围内。

4. 关于信号的负载要求

在扩展板上，共享的PCI信号只能带一个负载。如果违反扩

展板上布线长度或负载限制将损害系统信号的完整性。特别是以下做法都是违反规定的：

（1）直接在扩展板的任何 PCI 引脚上（或经总线收发器）挂一个扩展 ROM。

（2）不经过 PCI—PCI 桥而在一个扩展板上挂两个或更多的 PCI 设备。

（3）除一个设备外又在 PCI 引脚上挂任何监听逻辑。

（4）使用了 PCI 元件组，使每个 PCI 引脚上所带负载多于一个。例如地址线和数据相互分离的元件。

（5）使用了一个引脚电容大于 10 pF 的 PCI 元件。

（6）不经 PCI—PCI 桥而在 PCI 信号线上加了上拉电阻。

第五章　PCI 总线的机械特性

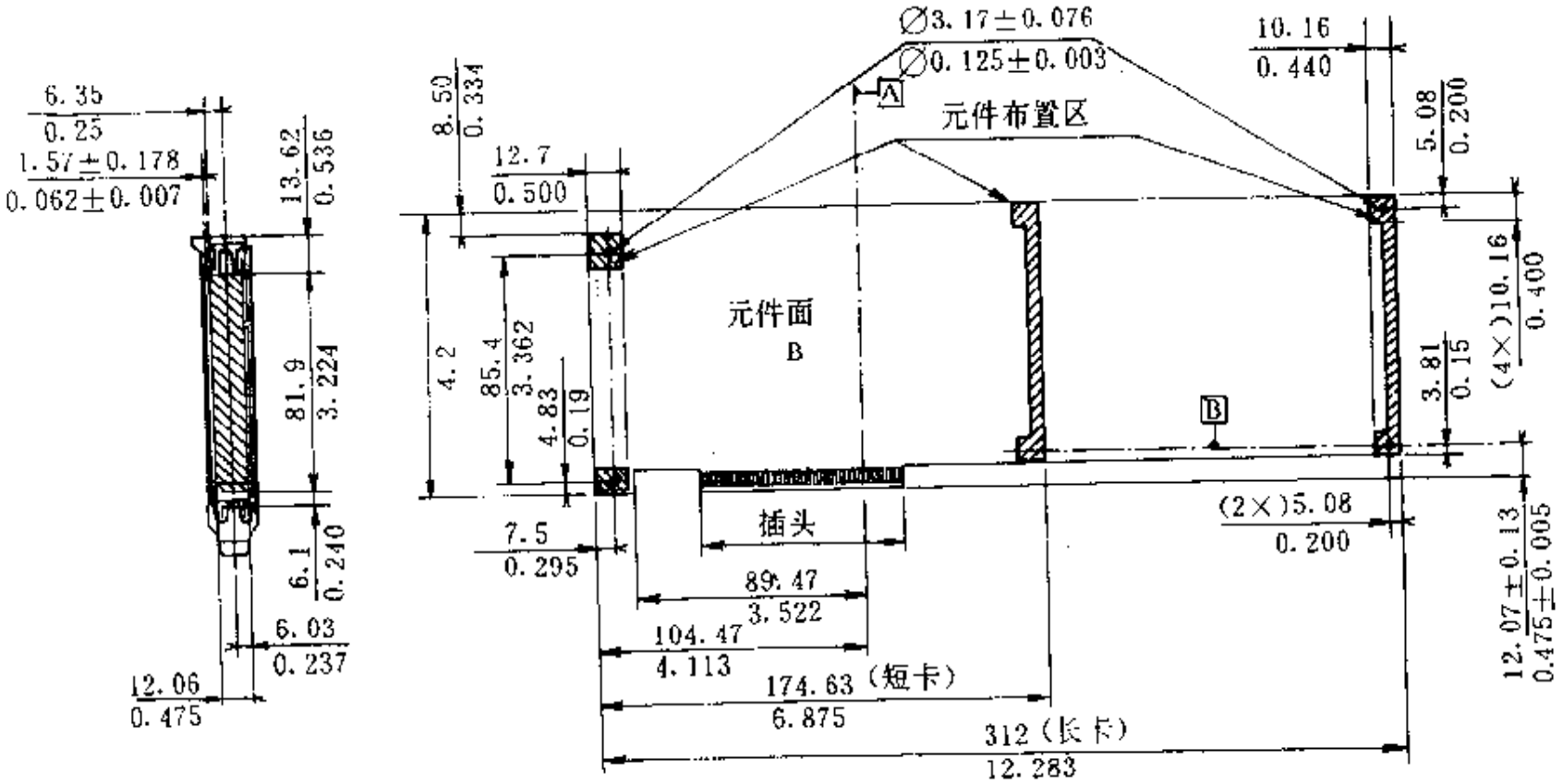
5.1　简介

PCI 扩展卡(见图 5.1 和图 5.2)采用原卡设计(Raw Card Design)思想，以便兼容于现有的 ISA、EISA 及 MCA 总线系统。PCI 扩展卡分为两种，即长卡和短卡。长卡提供多达 49 平方英寸的设计空间，用来实现一些复杂系统；短卡则提供以较小的成本和功耗来实现一定的功能和相应的系统，或者将已有系统小型化。PCI 为扩展板的连接定义了 32 位和 64 位两种接口。

PCI 扩展卡及连接器的设计关键在于 5 V 电源向 3.3 V 电源的过渡。基本的 32 位连接器之引脚逻辑数目为 124，但实际上只有 120 个引脚，其中键位占去了 4 个引脚的位置。当键位在某一位置时，连接器可接受 5 V 信号环境的板卡；若将它旋转 180°，连接器便可接受 3.3 V 信号环境的板卡。通用的 PCI 板卡，应设计成具有两个键位槽，以便插入任何一种连接器中，从而既可适应于 5 V 信号环境，又可适应于 3.3 V 信号环境。

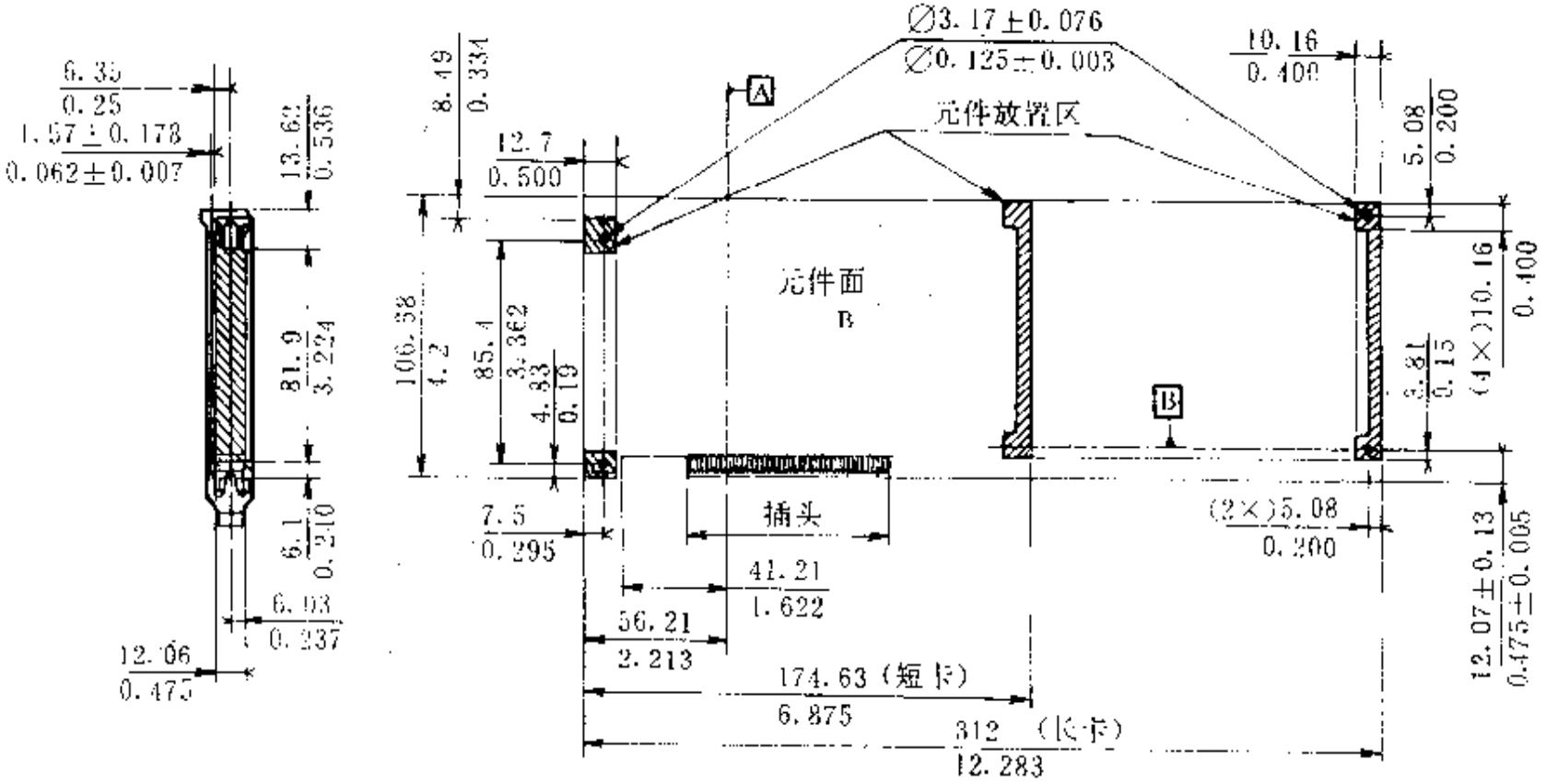
64 位扩展卡的连接器结构与 32 位相同，只是引脚数目增至 184 个。32 位连接器的系统信号是 64 位连接器的子集，所以，32 位和 64 位扩展卡可以互换使用。信号的电压级别由子集 32 位连接器键位来确定。32 位扩展卡在 64 位连接器上以 32 位方式进行数据传输，而 64 位扩展卡在 32 位连接器上时可配置成 32 位方式进行数据传输。

扩展卡的 PRSNT1＃和 PRSNT2＃两个引脚能够表明该卡的最大功耗，在初始化时，系统软件能够读取这两个引脚所对应的



图中未标公差者为：±0.127(0.005)

图 5.1　PCI 卡(5 V)



图中未标公差者为：0.127(0.005)

图 5.2 PCI 卡(3.3 V)

硬件编码，并以此来确定当前所提供的电流能否使系统可靠工作。关于电源等级及对应的编码已在第四章中作过介绍。

PCI 扩展卡包括一个供定位和插拔用的装配托架。装配托架固定于板卡上，同时在装配托架上提供了 PCI 扩展卡与系统之间的连线接口。PCI 的装配托架也可用于 ISA、EISA 及 MCA 总线系统。图 5.3 和图 5.4 是 ISA/EISA 系统的装配示意图；图 5.5 和图 5.6 为 MCA 总线系统的装配示意图。各种类型卡的托架配件必须由 PCI 来提供，以便用户用他们的系统对卡进行配置。ISA/EISA 总线的托架配件包括一个 PCI 托架，四个 4×40 的螺钉和一个板卡扩充装置。这样在 PCI 扩展板边缘固定的扩充装置通过标准 ISA 规范提供一种支持能力，可使 PCI 扩展卡和 MC 卡的装配一致。MC 托架配件包括一个 MC 托架、四个 4×40 的螺钉和一个托架槽。

PCI 扩展板的元件面和 ISA/EISA 及 MC 卡的元件面相反，也就是 PCI 卡是 ISA/EISA 及 MC 卡的镜像。在系统实现中，尽量减少所用扩展卡插槽是 PCI 总线的一个目标。在这些系统中，PCI 扩展板连接器、ISA/EISA 扩展板连接器和 MC 扩展板连接器可以共用一个插槽。用户可以在这种共享插槽中插入 PCI、ISA/EISA 或者 MC 卡。但是在同一时刻只能安装一个扩展卡。

5.2 PCI 扩展卡的物理尺寸及公差

在 PCI 扩展卡的元件面上，元件的最大高度不能超过 14.48 mm，而焊接面的元件高度不能超过 2.67 mm。在所有图中，凡是标有 A 的位置就是键位。图 5.1～图 5.7 为 PCI 扩展卡的物理尺寸。

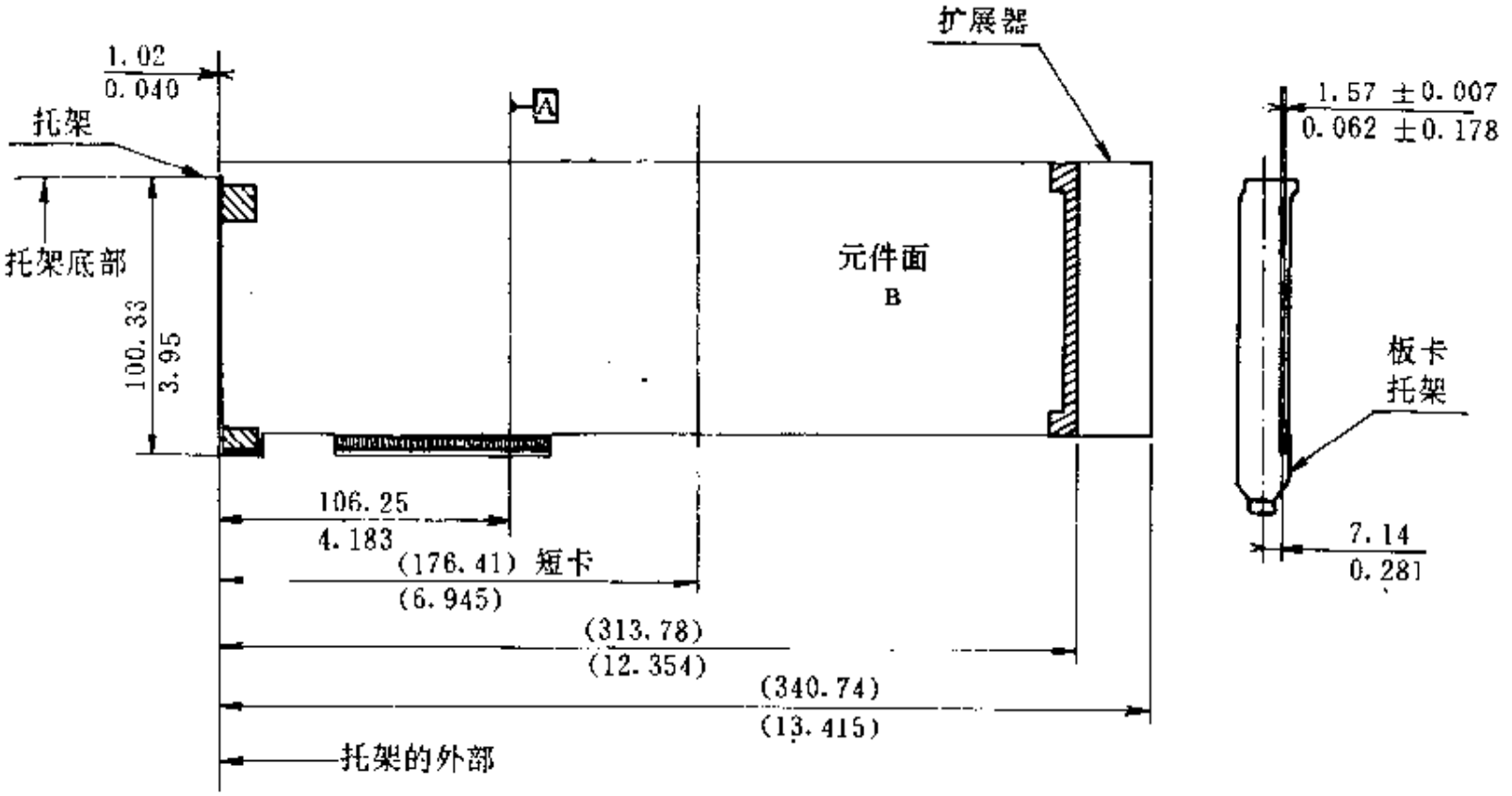

扩展器
1.02
0.040
托架
A
托架底部
100.33
3.95
元件面
B
106.25
4.183
(176.41) 短卡
(6.945)
(313.78)
(12.354)
(340.74)
(13.415)
托架的外部
1.57 ±0.007
0.062 ±0.178
板卡
托架
7.14
0.281

图 5.3　ISA 装配(5 V)

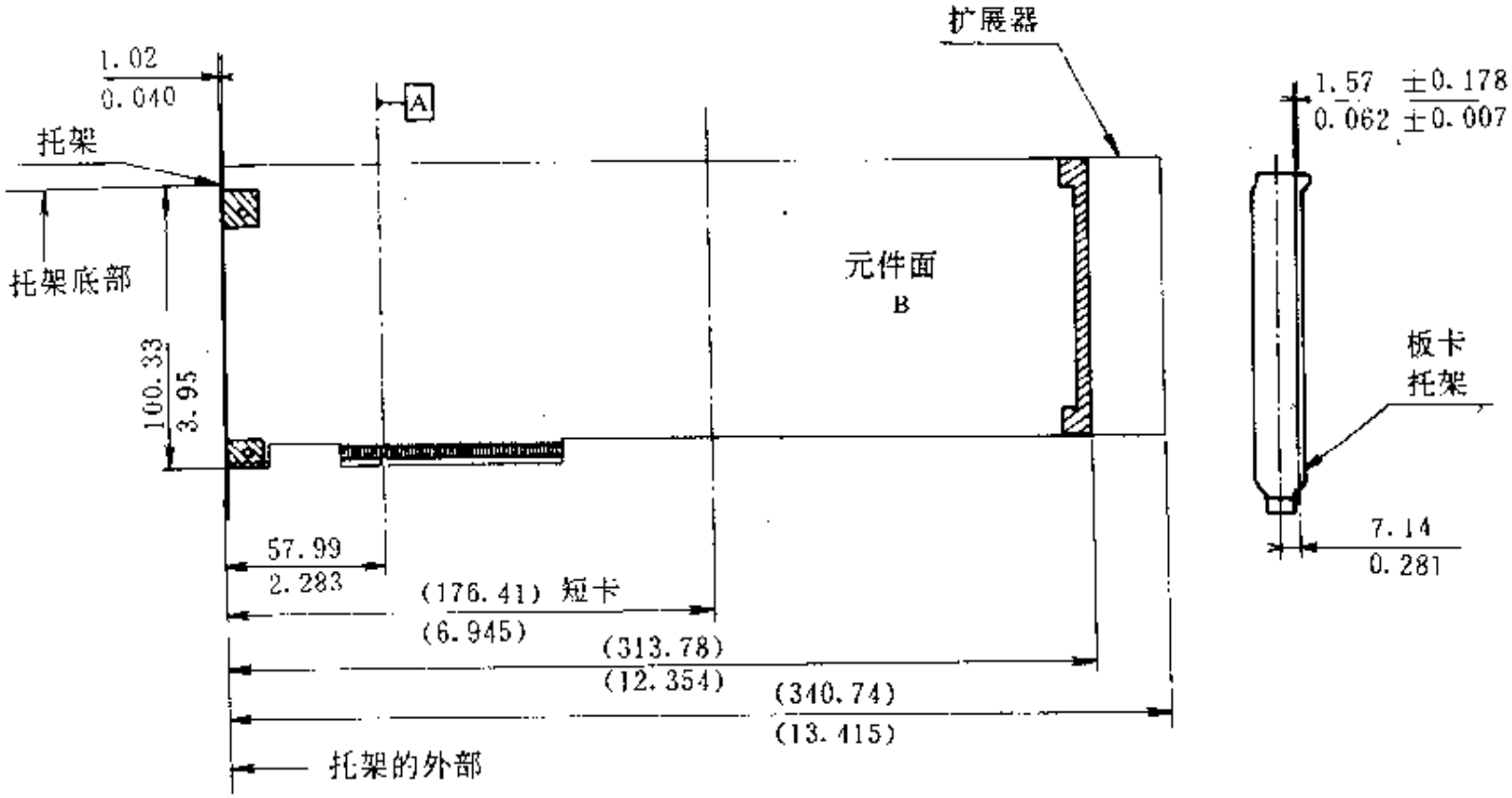


图 5.4　ISA 装配(3.3 V)

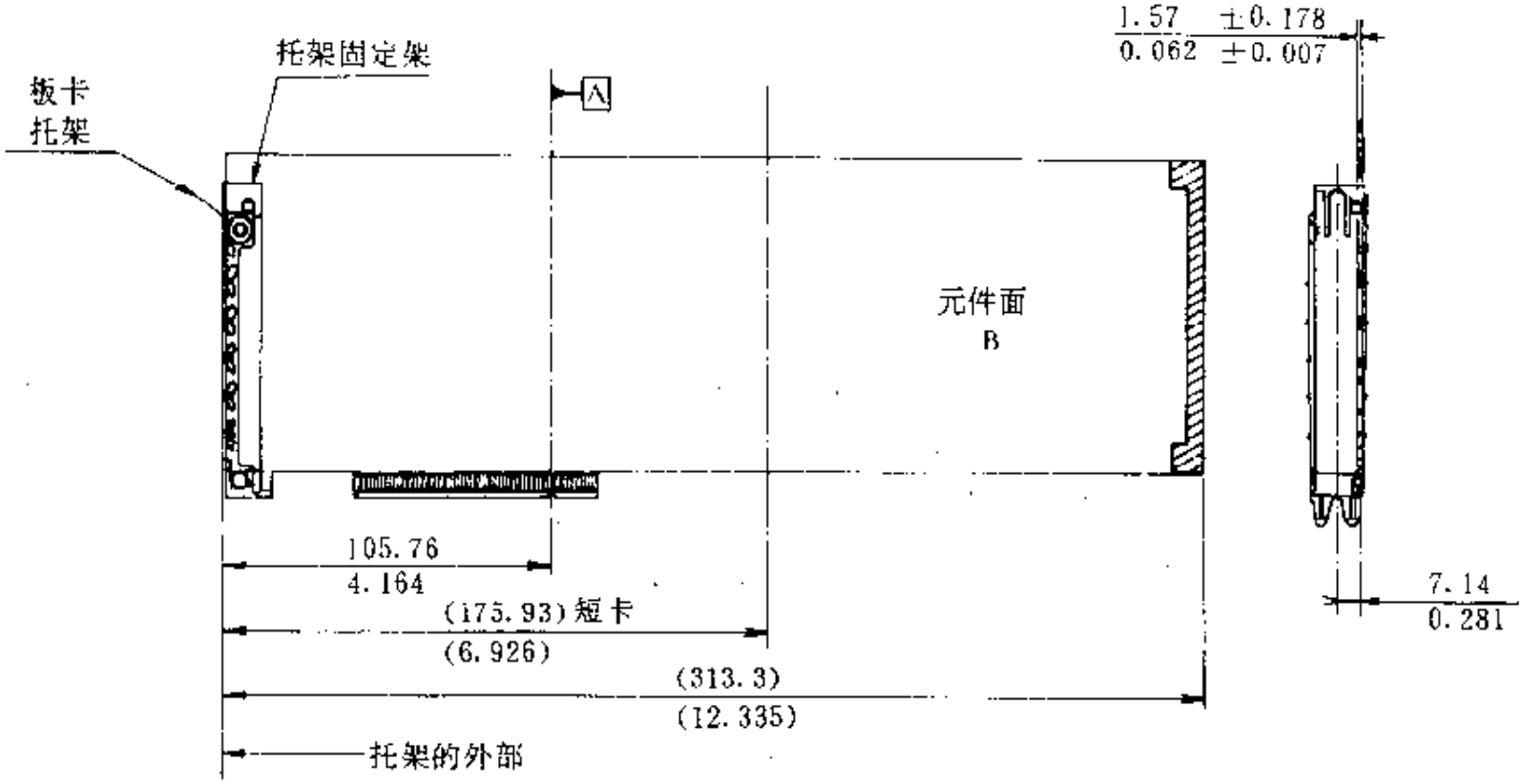

托架固定架
板卡
托架
A
1.57 ±0.178
0.062 ±0.007
元件面
B
105.76
4.164
(175.93) 短卡
(6.926)
(313.3)
(12.335)
托架的外部
7.14
0.281

图 5.5 MC 装配(5 V)

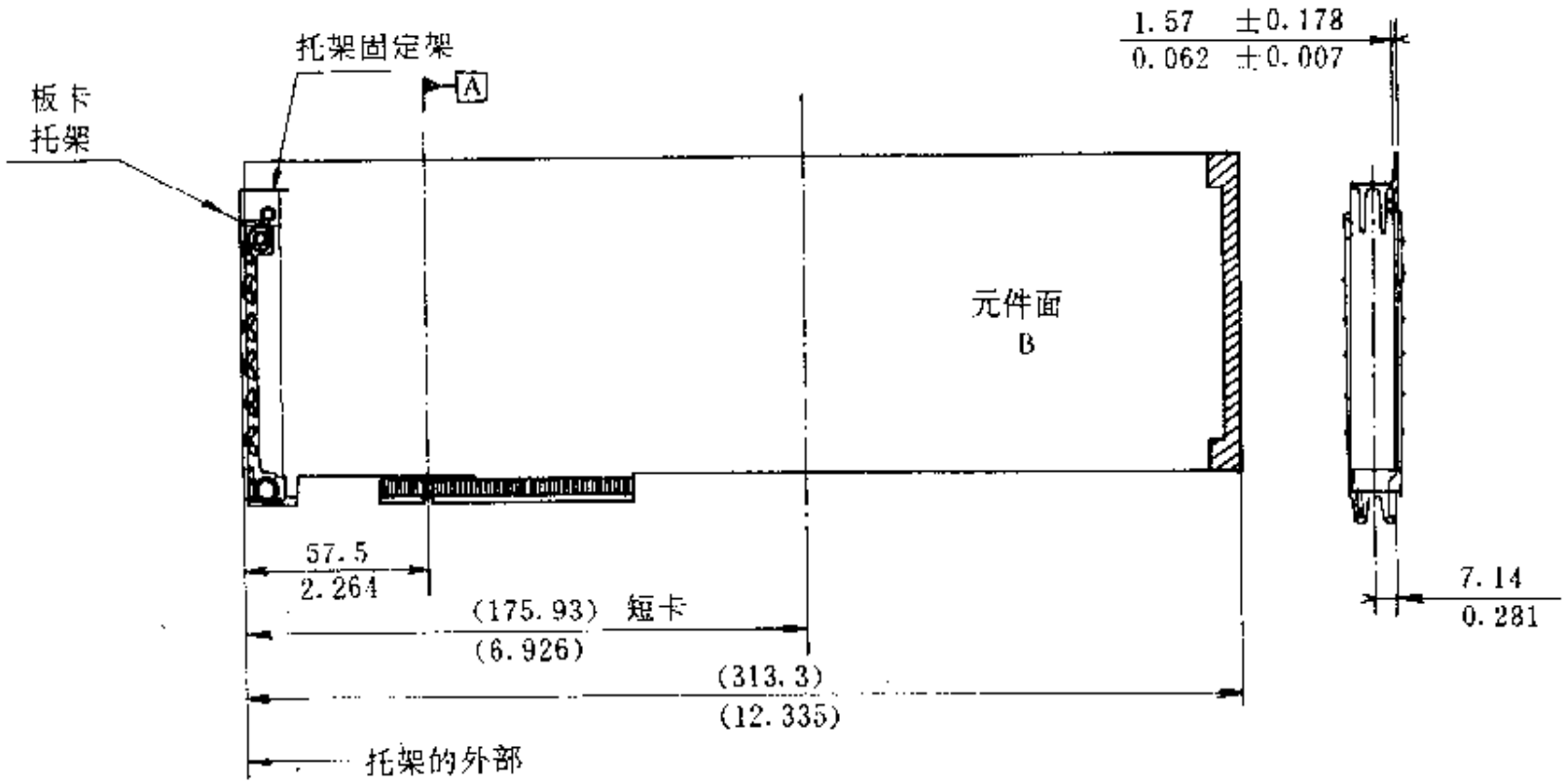

托架固定架
A
板卡
托架
1.57 ±0.178
0.062 ±0.007
元件面
B
57.5
2.264
(175.93) 短卡
(6.926)
(313.3)
(12.335)
托架的外部
7.14
0.281

图 5.6 MC 装配(3.3 V)

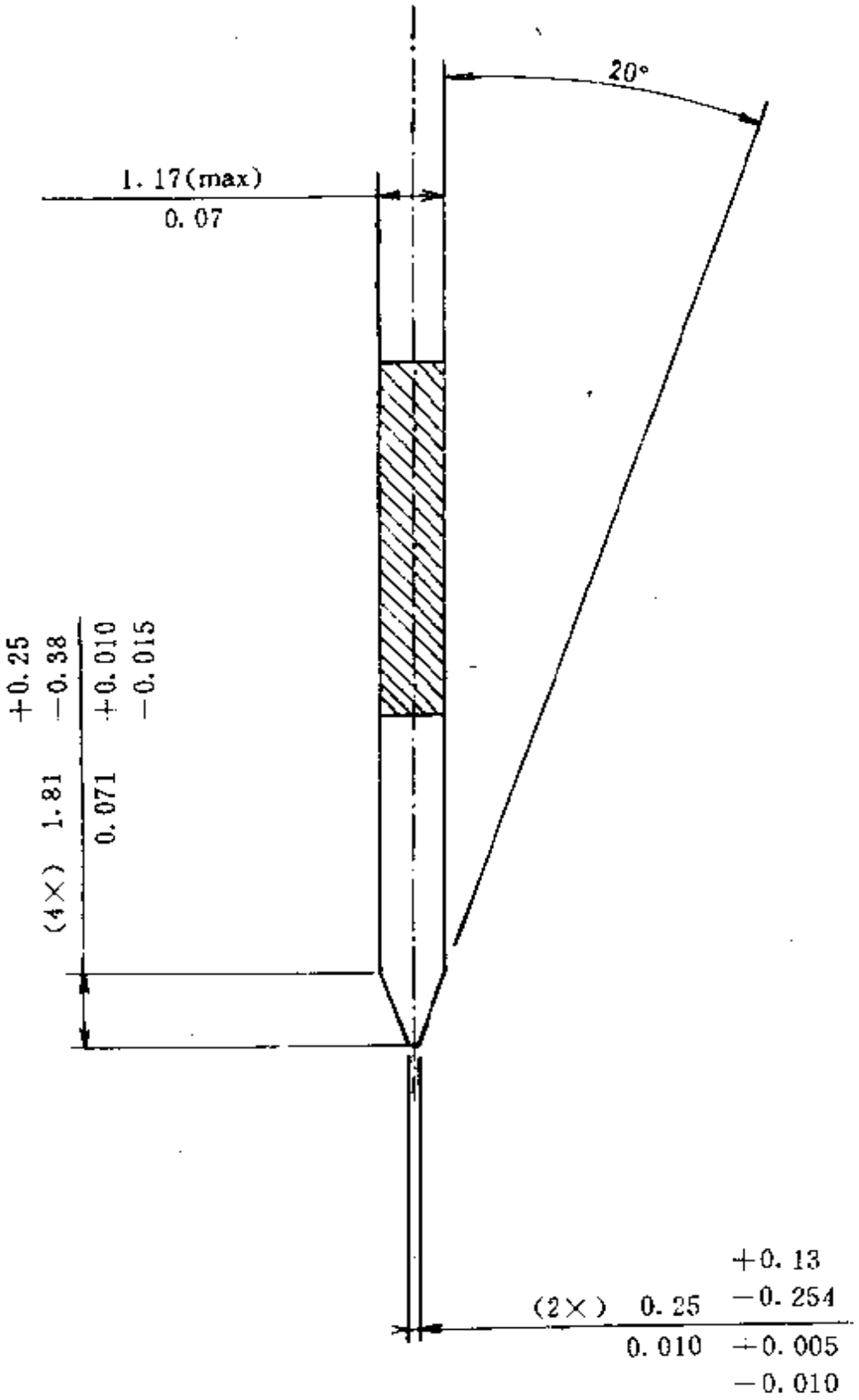


图 5.7　PCI 板卡边缘连接头斜角

5.3 连接器的物理描述

支持PCI扩展卡的连接器是由MC总线发展而来的。这是因为MC连接器的可靠性及性能已经受了实际的考验。有四种连接器可以决定PCI的实现，分别为32位、64位、5 V信号环境和3.3 V信号环境。键位决定了不同的信号环境。在同一个32位连接器上，在一个方向上接受5 V信号环境的板卡，当它旋转180°时，便可接受3.3 V信号环境的板卡。随着信号环境的变化，连接器的引脚数也随之变化，但其相对位置不变。

图5.8～图5.10表示各种连接器的尺寸。

图5.11～图5.17表示了各种板卡的边缘连接插头的尺寸和公差。

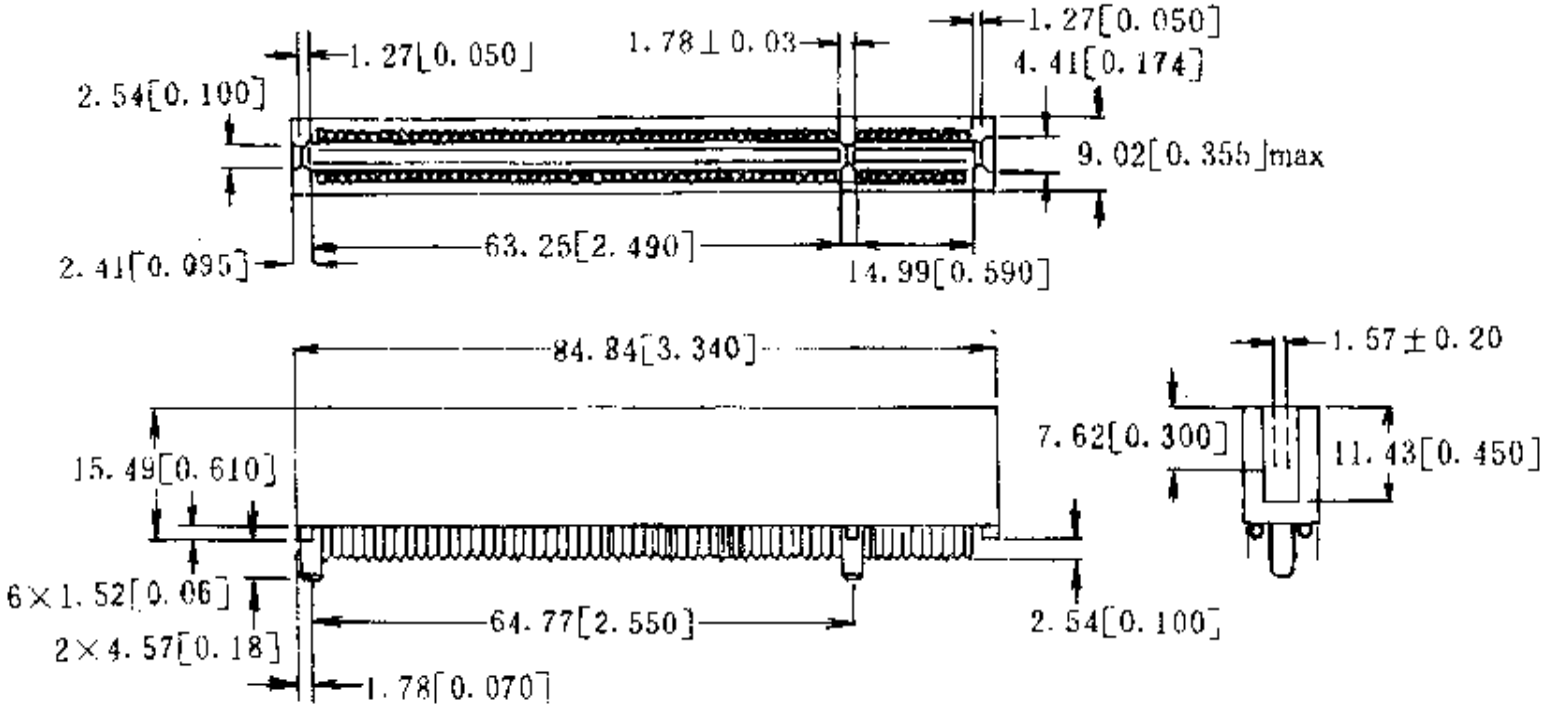

1.27[0.050]
1.78±0.03
1.27[0.050]
4.41[0.174]
2.54[0.100]
9.02[0.355]max
2.41[0.095]
63.25[2.490]
14.99[0.590]
84.84[3.340]
1.57±0.20
7.62[0.300]
11.43[0.450]
15.49[0.610]
6×1.52[0.06]
2×4.57[0.18]
64.77[2.550]
2.54[0.100]
1.78[0.070]

图 5.8　32 位连接器

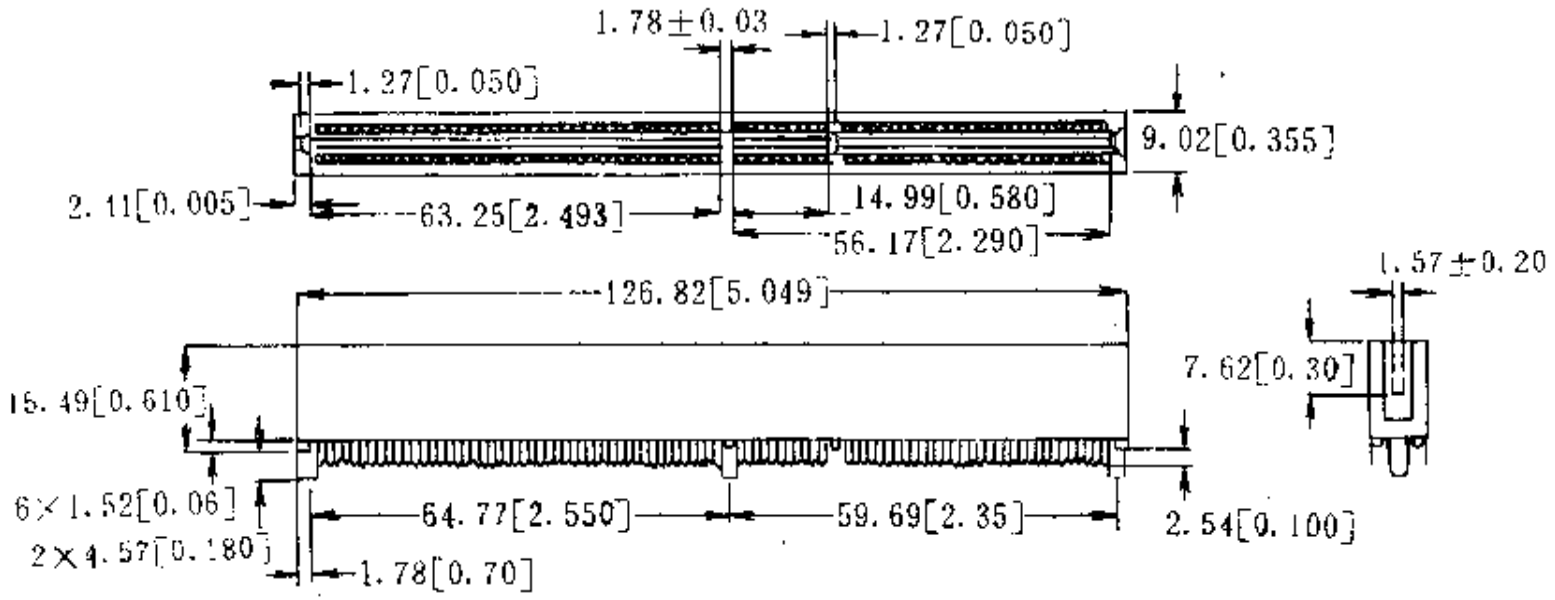


图 5.9　5 V/64 位连接器

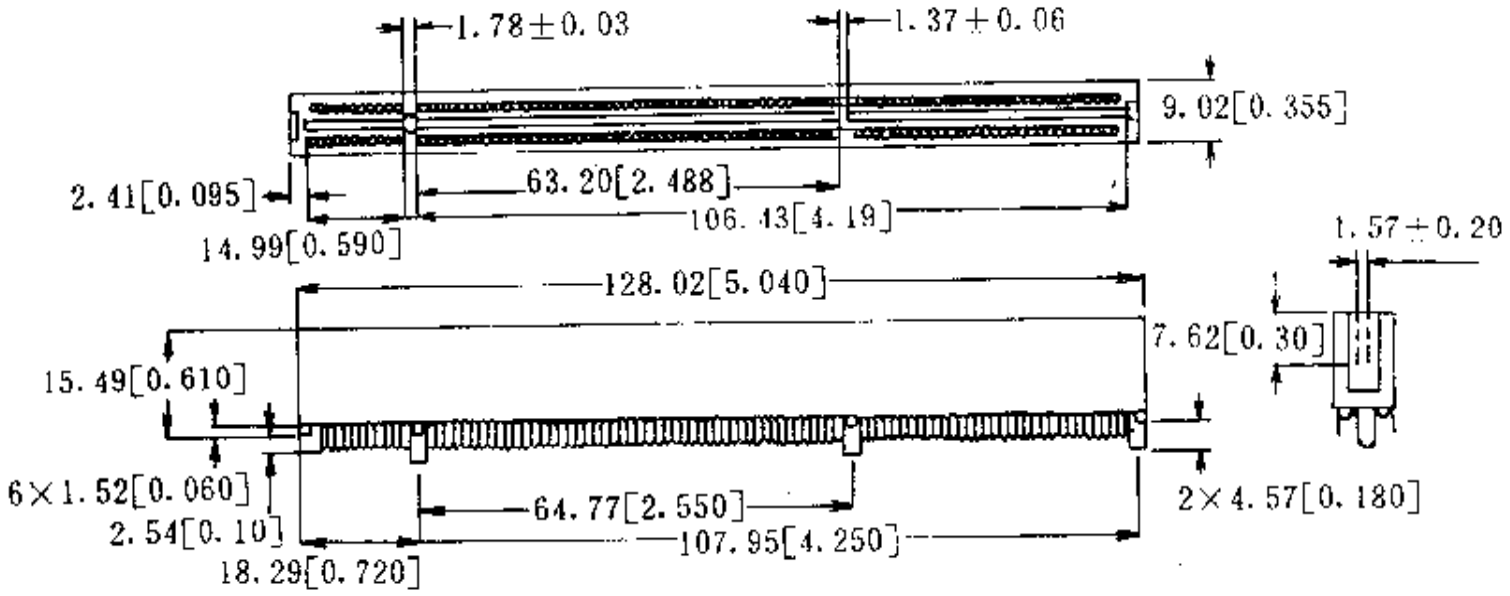


图 5.10　3.3 V/64 位连接器

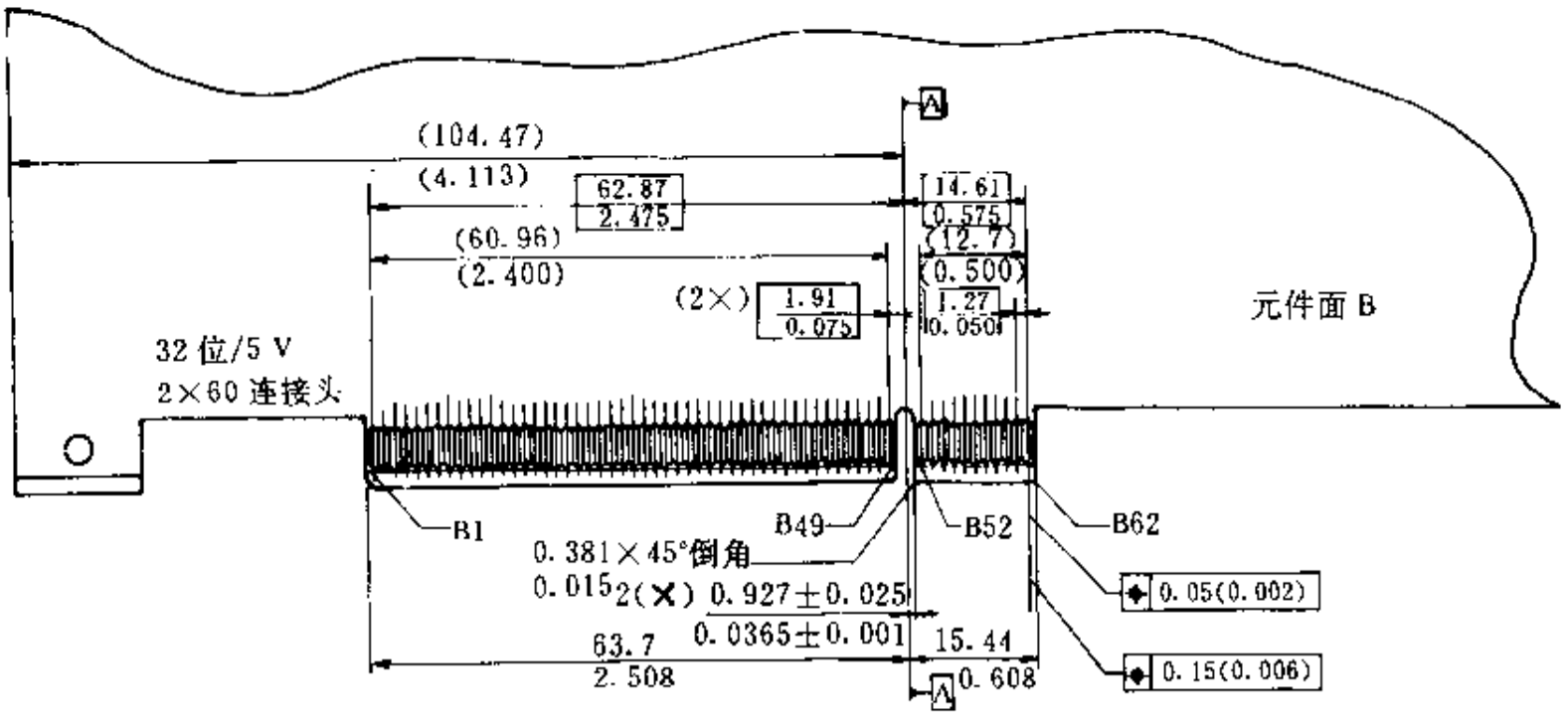


图 5.11　5 V/32 位卡边缘连接头尺寸及公差

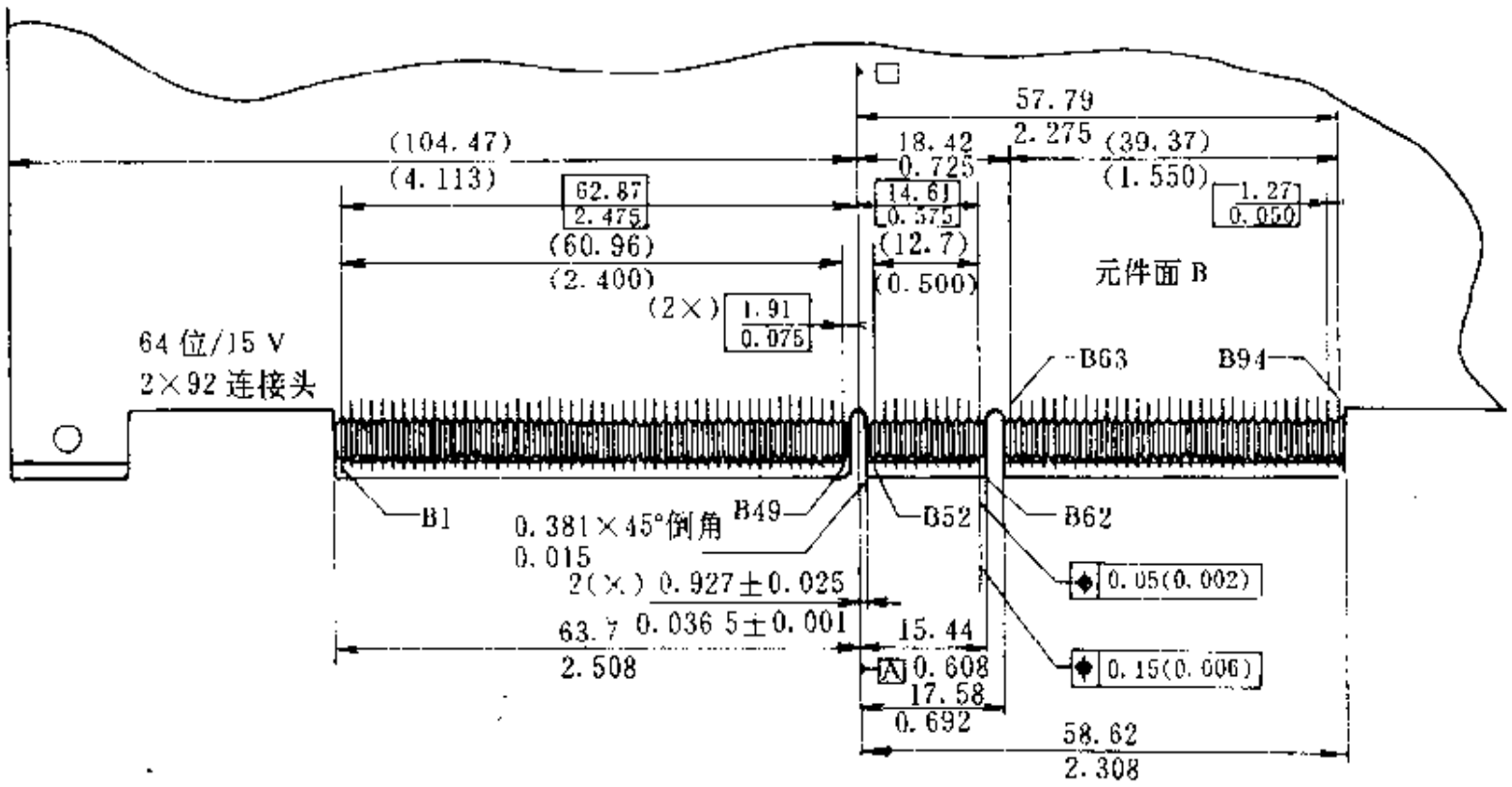


图 5.12　5 V/64 位卡边缘连接头尺寸及公差

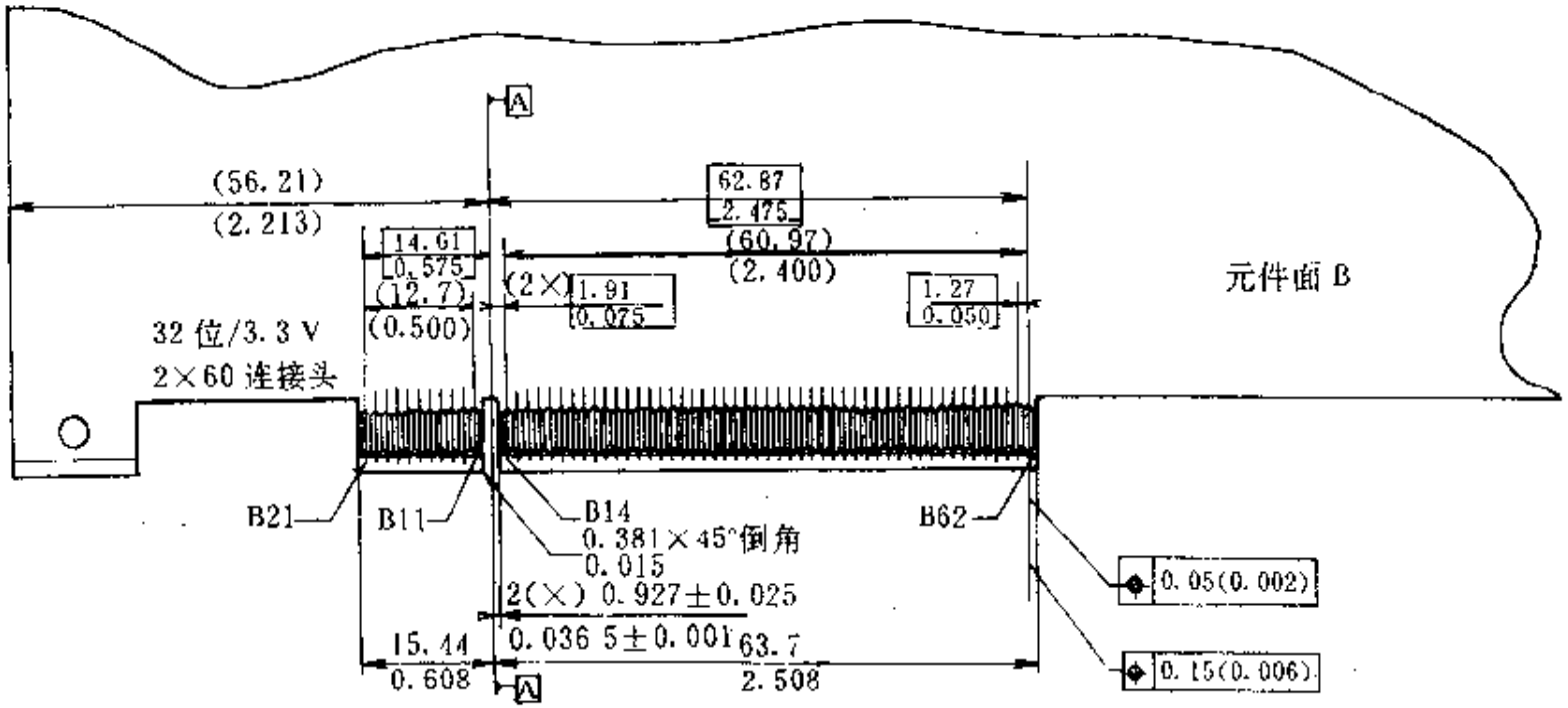


图 5.13　3.3 V/32 位卡边缘连接头尺寸及公差

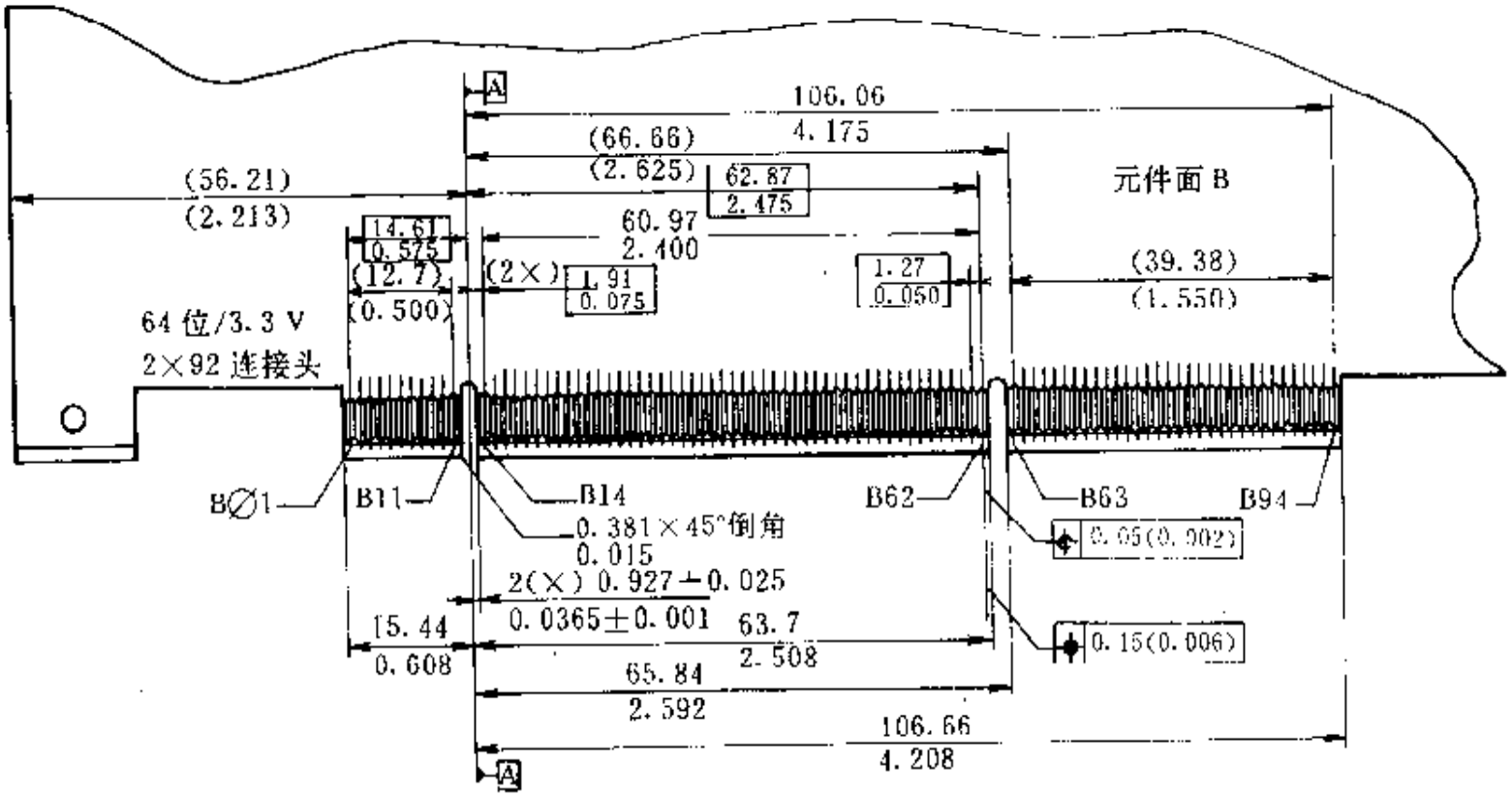


图 5.11　3.3 V/64 位卡边缘连接头尺寸及公差

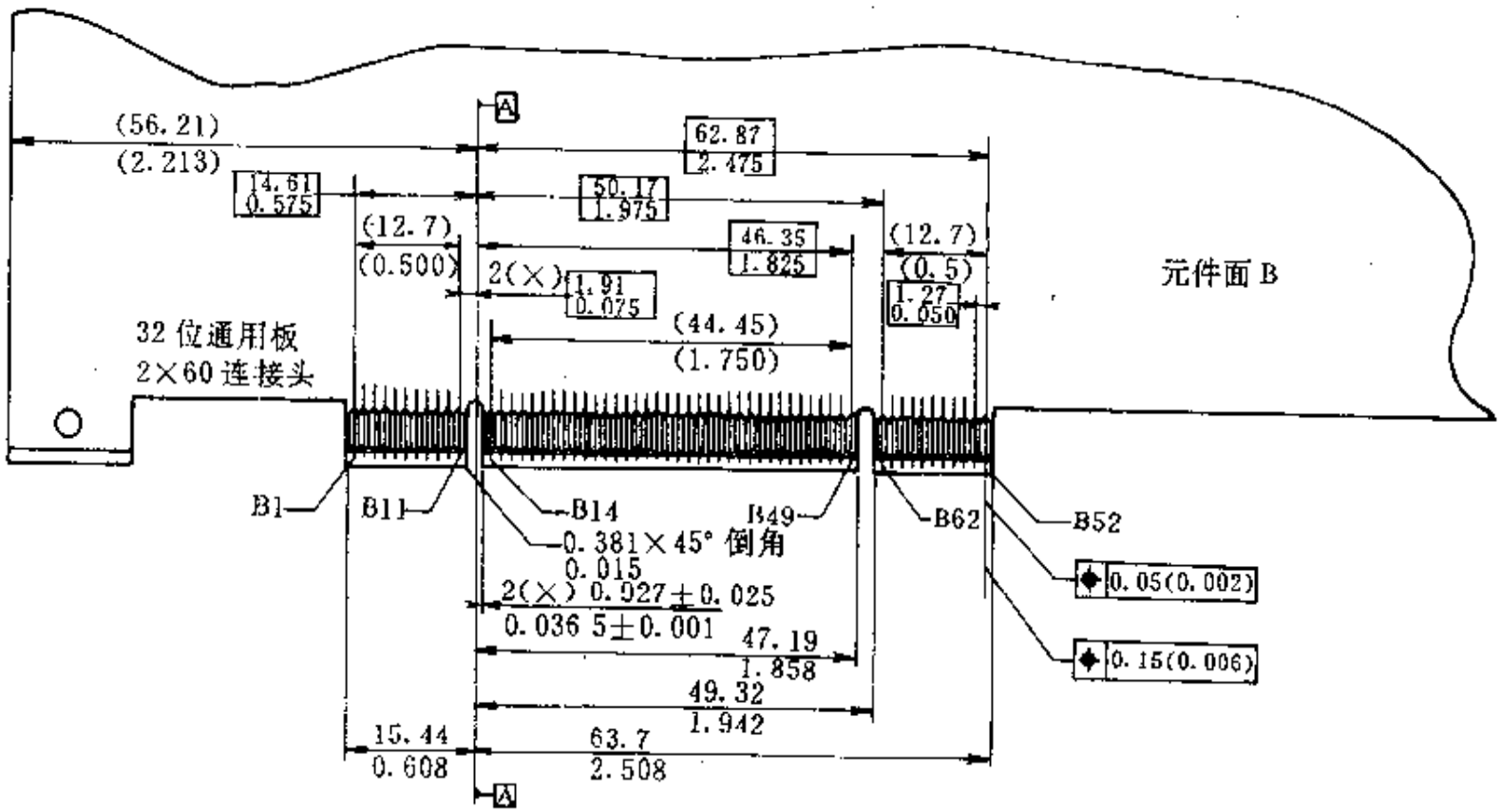


图 5.15　通用 32 位卡边缘连接头尺寸及公差

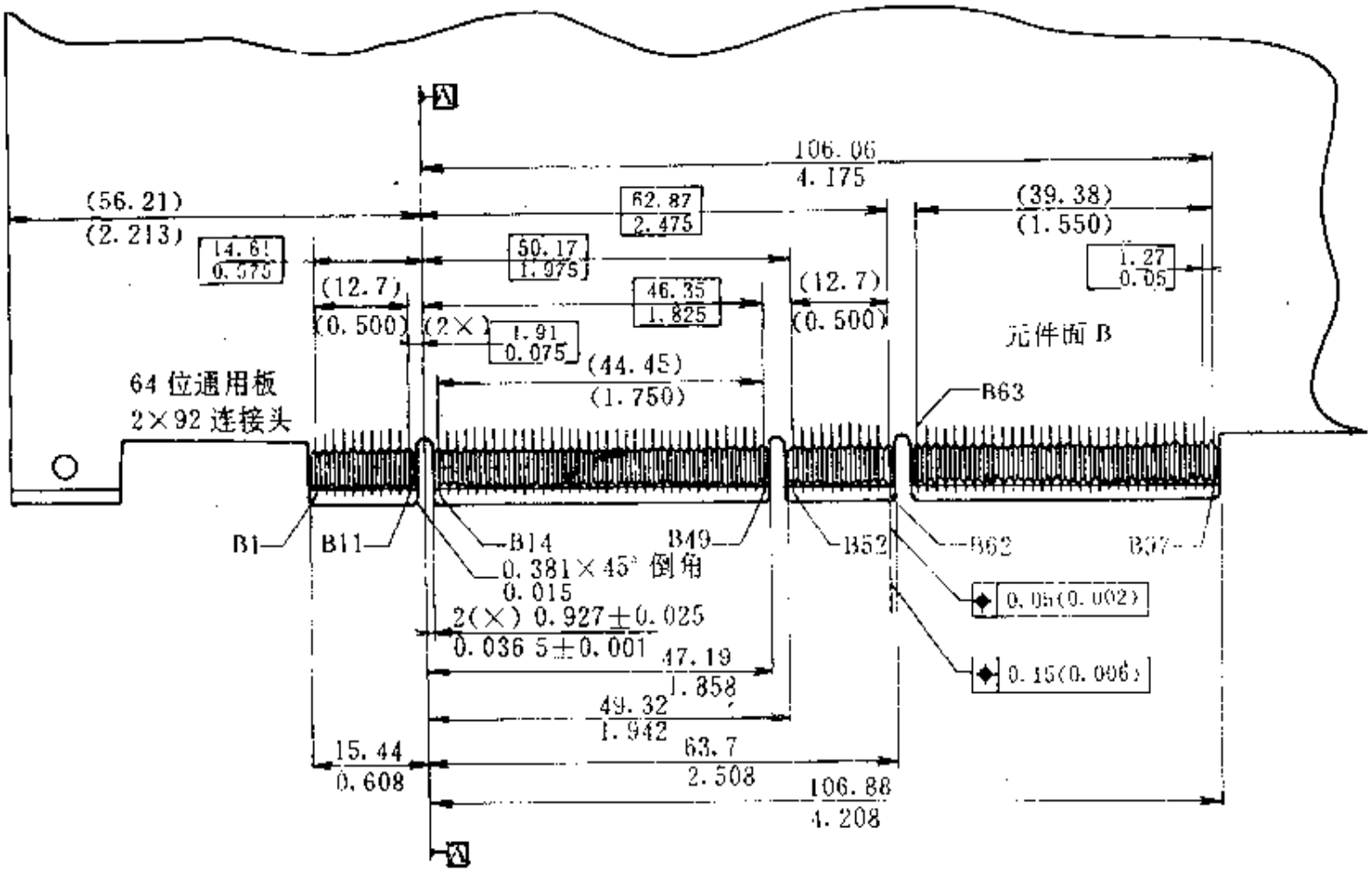


图 5.16　通用 64 位卡边缘连接头尺寸及公差

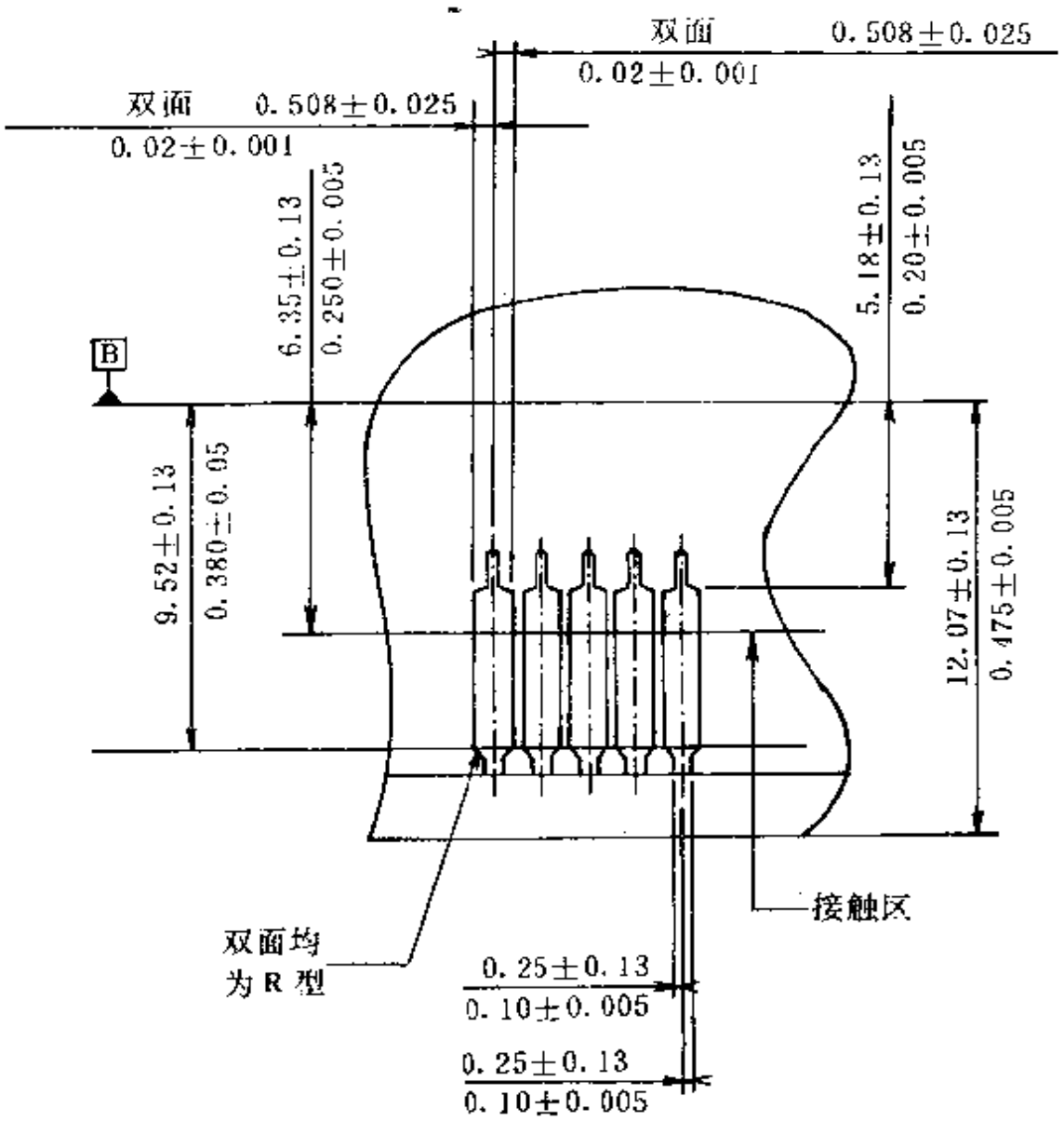


图 5.17　PCI 卡边缘连接头接触点

第六章 配 置 空 间

本章将讨论 PCI 总线设备中配置寄存器空间的编程模式和使用规则。其讨论范围只限于各种类型的系统的 PCI 设备所共有的规定，不涉及某一个特定平台的系统。与某一类 PC 相容的系统所涉及的问题(例如将各种 PCI 地址空间映射到 CPU 地址空间，访问顺序的规定以及主 CPU 到 PCI 的主桥路等)。

定义一个 PCI 配置空间的目的在于：提供一套适当的配置措施，使之满足现行的和可预见的系统配置机构，因为这些机构还在发展之中，对它们无法限定或者说对它们的用法不能加以约束。总之，提供这些配置的要求是：

(1) 充分支持未来的配置机构，使之提供：完全的设备再定位；无需用户干预 的安装、配置和引导；由与设备无关的软件进行系统地址映射。

(2) 有效地支持现有的配置机构(如 EISA 配置实用程序)。

(3) 将功能要求所造成的硅片负担最小化。

(4) 有利于设备通用性，避免由于独特的要求而排斥一些设备。

6.1 配置空间的组织

本节讨论配置空间寄存器的组织。配置空间是一容量为 256 字节并具有特定记录结构或模型的地址空间。该空间又分为头标区和设备有关区两部分。设备在每个区中只实现必要的和与之相配的寄存器。一个设备的配置空间不仅在系统自举时可以访问，在其它时间内也是可以访问的。

头标区的长度为 64 字节，每个设备都必须支持该区的寄存器分配。该区中的各个字段用来唯一地识别设备，并使设备能以一般方法控制。其余的 192 字节是因设备而异，不在本书的讨论范围之内。

系统软件可能需要扫描 PCI 总线以便确定实际上存在什么样的设备。为此，配置软件必须读取每个 PCI 槽位上的设备供应商识别代码。如果所读取的位置上不存在一个设备，则从宿主总线连到 PCI 的桥必须准确无误地报告出来。由于 OFFFH 是一个非法的供应商识别码，所以，宿主总线到 PCI 的桥可以返回一个全“1”，作为一个设备的配置空间寄存器之读出值，以表示设备不存在。

对于配置空间中保留寄存器的写操作，所有的 PCI 设备都必须将其作为空操作来处理，也就是说访问必须在总线上正常完成，但总线数据要丢弃。对于保留的或者未实现的寄存器的读访问也必须按正常方式完成，并且回送一个全“0”的数据值。

图 6.1 中给出了每个设备必须支持的 256 字节配置空间中预先定义的 64 字节头标区的布局情况。

任何因设备而异的寄存器都不在这个头标区，而必须安排在 64～255 所对应的地址空间。所有多字节的数码字段都遵循低位在后的排列顺序，也就是说，低部地址含有字段值的低位部分。对于位编码字段中某些为将来使用而保留的位，软件必须小心正确地处理它们。在读取时，软件必须以适当的屏蔽来抽取定义过的位，而不能指望保留的位有任何特定的值。而进行写操作时，软件必须保证使保留位的值不发生变化。实际上，可以先将保留的值读出来，与其它位置上的新值合并为新的数据再写入，即可达到目的。

配置空间的头标区又分为两部分。前 16 个字节的定义在各种类型的设备中都是一样的，而其余 48 个字节可以根据设备支

持的基本功能情况进行不同的布置。位于偏移地址 0EH 处的头标类型字段规定了所提供的是何种布置。目前，只有一种头标类型定义(00H)，其具体布置如图 6.1 所示。将来的修改版本要为特殊的功能(如插接存储器、PCI—PCI 桥等)而定义其它的头标类型。

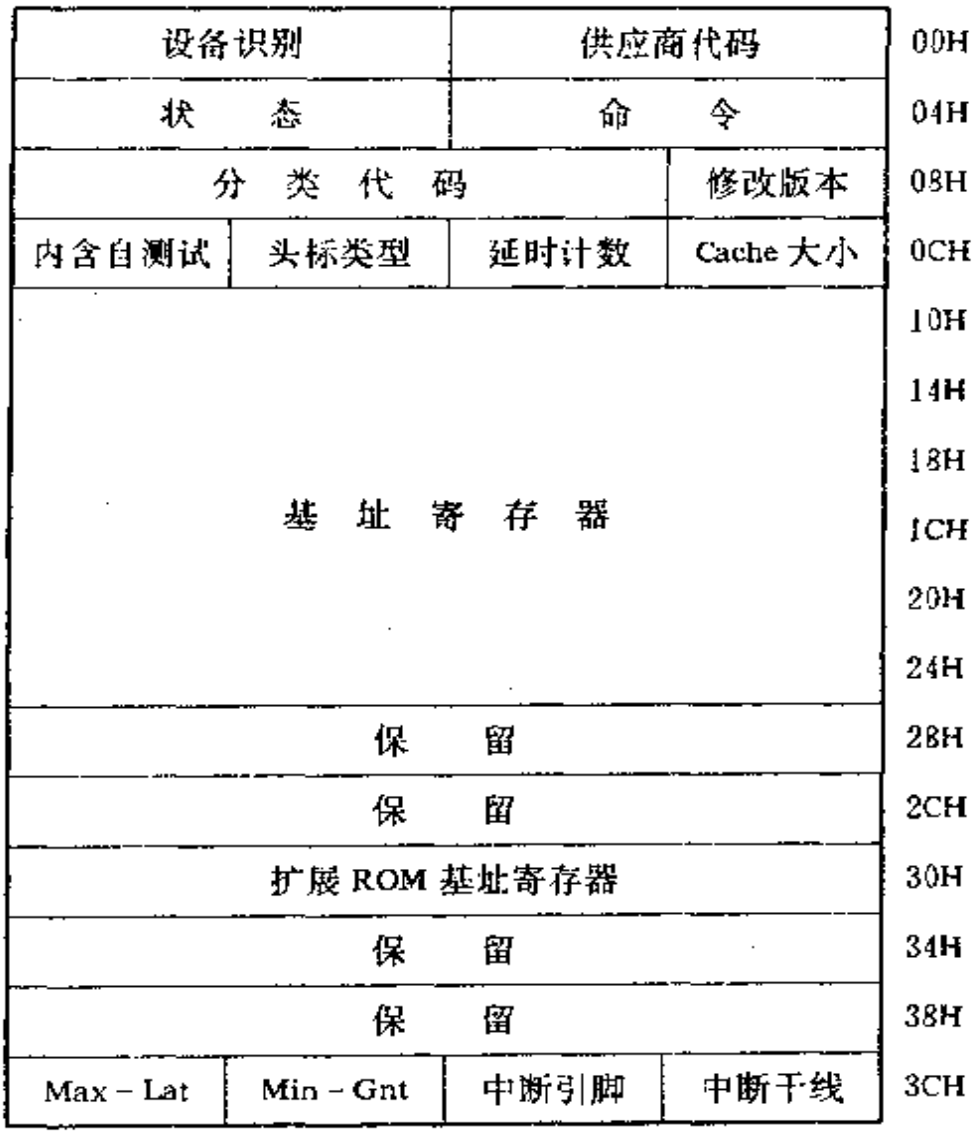


图 6.1　配置空间头标区

所有符合 PCI 要求的从设备都必须支持供应商识别、设备识别、命令和状态字段。其它寄存器的实现是可选的，也就是可以将它们当作保留的寄存器来处理，具体视设备的功能而定。如果

一个设备支持的功能是某个寄存器所管辖的，则它必须实现此寄存器，并且要符合规定的位置和功能。至于配置空间头标区内各个寄存器的功能将在 6.2 节进行说明。

6.2 配置空间的功能

PCI 总线为系统配置的简易性提供了较大的潜力，为了实现这个潜力，所有的 PCI 设备必须提供一定的功能，以使系统配置软件能够被利用。本节也将列举一些需通过配置空间头标区寄存器支持的功能。这些寄存器的精确格式(实现的位数)往往与具体设备有关，但是有些共同的规则必须遵守。所有的寄存器必须是可读的，而且其返回数据必须能够代表设备正在使用的实际值。

6.2.1 设备识别

在头标区有五个字段涉及设备的识别。所有的 PCI 设备必须实现这些字段，一般配置软件利用它们便能很容易地确定在系统的 PCI 总线上有什么样的可用设备。所有这类寄存器都是只读寄存器。下面说明它们的具体功能。

1. 供应商识别字段(Vendor ID)

该字段用以标明设备的制造者。一个有效的供应商标识由 PCI SIG 来分配，以保证它的唯一性。0FFFH 是该字段的无效值。

2. 设备识别字段(Device ID)

该字段用以标明特定的设备，具体代码由供应商来分配。

3. 修改识别字段(Revision ID)

该字段用来指定一个设备特有的修改识别代码，其值由供应商来选定，0 是一个可接受的值。该字段应当看作是设备识别字段的扩展。

4. 头标类型字段(Header Type)

该字段有两个作用，其一是用来表示配置空间内字节 10H～3FH 的布局类型；其二是用以指出设备是否包含多功能。该字段的位 7 用来标识一个多功能设备，如果此位为 0 则表示相应的设备为单功能设备，反之，若为 1 则说明该设备为多功能设备。位 6～位 0 指出字节 10H～3FH 的布局情况。现在只有一个编码 00H 对应图 6.1 的布局，其它所有编码均属保留编码。

5. 分类代码字段(Class Code)

分类代码寄存器是只读的，用来标识设备的总体功能和特定的寄存器级编程接口。该寄存器分为三段，每段占一个字节。高位字节在 0BH 处，是一个基本分类代码，对设备的功能进行粗略的分类，见表 6.1；中间字节在 0AH 处，叫做子分类代码，它对设备的功能给予更精确、更详细的分类。低位字节在 09H 处，用来标识一个特定的寄存器级编程接口(如果有的话)，以使与设备无关的软件可以与设备交互作用。

表 6.1 基本分类代码编码

基本分类	含　　义
00h	分类代码定义确定之前已有的设备
01h	海量存储控制器
02h	网络控制器
03h	显示控制器
04h	多媒体设备
05h	存储器控制器
06h	桥路设备
07h～EFh	保留
EFh	不符合上述分类的其它设备

下面给出上述已定义的基本分类代码所对应的子类和编程接口。所有没有指定的编码都是保留的。

(1) 基本类 00h：设立该类的目的是为那些在分类代码字段定义之前已经有了的设备提供向后兼容能力。新出的设备不应该使用此值，现有的设备也应尽可能将它改到一个更为合适的值。

表 6.2 为该类所对应的子类，其中未列出的值都属于保留的。

表 6.2 基本分类代码 00h 对应的子类代码

基本类	子 类	编程接口	含 义
00h	00h	00h	除 VGA 兼容设备之外的所有当前设备
	01h	00h	VGA 兼容设备

(2) 基本类 01h：该类用来定义海量存储控制器的类型，只有几个子类值有定义，没有定义特别的寄存器级编程接口，具体见表 6.3 所示。

表 6.3 基本分类代码 01h 对应的子类代码

基本类	子 类	编程接口	含 义
01h	00h	00h	SCSI 总线控制器
	01h	00h	IDE 控制器
	02h	00h	软盘控制器
	03h	00h	IPI 总线控制器
	80h	00h	其它海量存储控制器

(3) 基本类 02h：该类是为各种类型的网络控制器而定义的。只定义了几个子类值，而未定义特别的寄存器级编程接口。见表 6.4。

表 6.4 基本分类代码 02h 对应的子类代码

基本类	子 类	编程接口	含 义
02h	00h	00h	以太网控制器
	01h	00h	令牌环网控制器
	02h	00h	FDDI 控制器
	80h	00h	其它网络控制器

(4) 基本类 03h：该类代码是为了各种类型的显示控制器而定义的。定义了几个子类代码及其与之相对应的特定的已知寄存器级编程接口。见表 6.5。

表 6.5 基本分类代码 03h 对应的子类代码

基本类	子 类	编程接口	含 义
03h	00h	00h	VGA 兼容控制器
	01h	00h	XGA 控制器
	80h	00h	其它显示控制器

(5) 基本类 04h：该类是为各种多媒体设备而定义的，只定义了几个子类代码，未定义特别的寄存器级编程接口，见表 6.6。

表 6.6 基本分类代码 04h 对应的子类代码

基本类	子 类	编程接口	含 义
04h	00h	00h	视频设备
	01h	00h	音频设备
	80h	00h	其它多媒体设备

(6) 基本类 05h：该类是为各种类型的存储控制器而定义的，只定义了几个子类代码，没有定义特别的寄存器级编程接口，见表 6.7。

表 6.7 基本分类代码 05h 对应的子类代码

基本类	子 类	编程接口	含 义
05h	00h	00h	RAM
	01h	00h	FLASH
	80h	00h	其 它

(7) 基本类 06h：该类是为各类桥路设备而定义的，如果一个 PCI 设备把它一侧的 PCI 资源能够映射到另一侧，它就是 PCI 桥。只定义了几个子类，没有特定的寄存器级编程接口。见表 6.8。

表 6.8 基本分类代码 06h 对应的子类代码

基本类	子 类	编程接口	含 义
06h	00h	00h	主桥
	01h	00h	ISA 桥
	02h	00h	EISA 桥
	03h	00h	MC 桥
	04h	00h	PCI—PCI 桥
	05h	00h	PCMCIA 桥
	80h	00h	其它

6.2.2 关于设备的控制

头标区中的命令寄存器(Command)可为发出和响应 PCI 总线命令提供粗略的控制。当向这个寄存器写入 0 时，对应的设备在逻辑上除了可进行配置访问外，其它所有访问均与 PCI 总线脱开。要求所有的设备都应支持这一基本的功能。根据一个设备的功能要求，命令寄存器的个别位可以实现，也可以不实现。例如，一个不实现 I/O 空间的设备，大概不会在命令寄存器中的 0 位实现可写功能。对于典型的设备，在系统复位后该寄存器应为全“0”。关于命令寄存器的具体格式见图 6.2 所示。

图中各位的功能及含义如下：

(1) 位 0(I/O 空间控制)：控制对 I/O 空间访问的响应。当它为 0 时，禁止设备响应对 I/O 空间的访问；当其值为 1 时，则允许设备响应对 I/O 空间的访问。

(2) 位 1(存储器空间控制)：控制一个设备对存储器空间访问的响应。其值为 0 时，禁止响应；当它为 1 时，允许设备响应对存储器空间的访问。

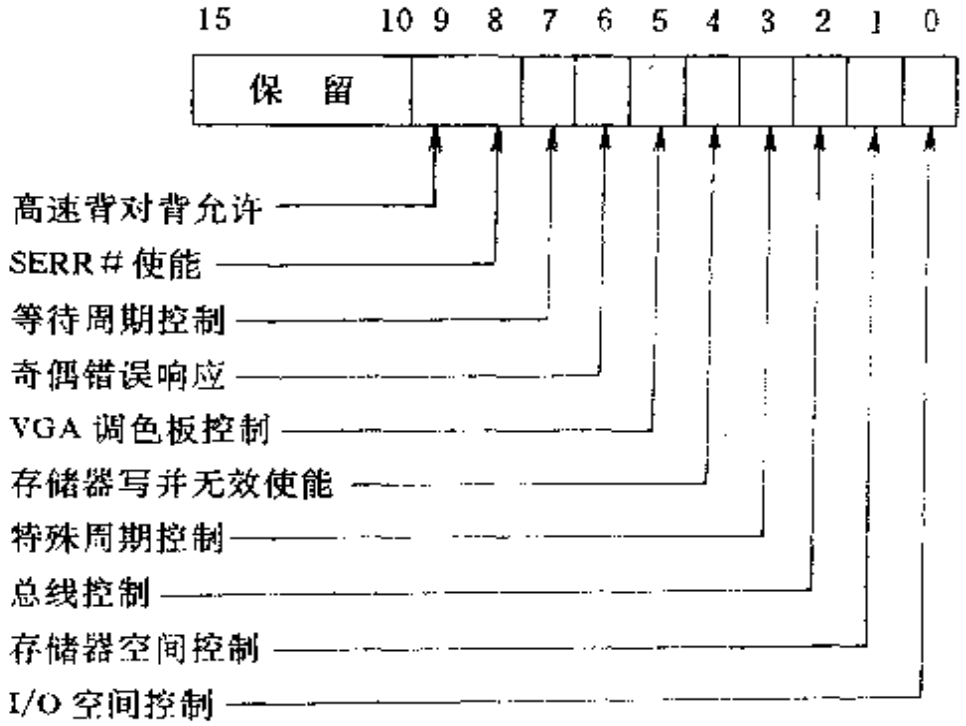


图 6.2 命令寄存器格式

(3) 位 2(总线控制)：控制一个设备在 PCI 总线上作为主设备的工作能力。该位为 0 时，禁止设备发出 PCI 访问；当它为 1 时，允许设备作为总线主设备工作。

(4) 位 3(特殊周期控制)：控制一个设备在特殊周期操作中的行为。该位为 0 时，设备将对任何特殊周期操作不予理睬；该位为 1 时，则表示允许设备参加总线上的特殊周期操作。

(5) 位 4(存储器写并无效使能)：用以决定是否允许设备发出存储器写并无效命令。该位为 0 时，主设备必须用存储器写命令来代替存储器写并无效命令；该位为 1 时，则允许主设备发此命令。RST# 之后该位为 0，凡是可以发存储器写并无效命令的主设备必须实现该位。

(6) 位 5(VGA 调色板控制)：用来控制 VGA 兼容设备如何处理对其调色板寄存器的访问。该位为 0 时，设备应当像处理任何其它访问那样来处理对调色板的访问；该位置 1 时，允许设备进

行特殊的调色板监听。很显然，VGA 兼容设备应实现这一位。

(7) 位 6(奇偶错误响应)：该位用以控制设备对奇偶错误的响应。该位为 0 时，设备必须忽略任何奇偶错误而继续正常的操作；该位置 1 时，设备必须在发现奇偶错误时，采取相应措施。该位在复位之后必须为 0。有奇偶校验的设备必须实现该位，即使奇偶校验被禁止，也要照常产生奇偶位。

(8) 位 7(等待周期控制)：该位用来决定一个设备是否可以进行地址/数据的渐近。永远不会做渐近的设备应将该位恒接为 0；反之、总是渐近的设备应将该位恒接为 1。对于上述两种可能都存在的设备，应将此位做成读写位，并且复位后该位的初始值为 1。

(9) 位 8(SERR＃使能)：该位用于控制 SERR＃驱动器。该位为 0 时禁止 SERR＃驱动器工作；该位为 1 时，则允许 SERR＃驱动器工作。复位后该位为 0。凡是有 SERR＃信号线的设备都必须实现此位。另外，只有当该位和位 6 同时为 1 时，才能报告地址奇偶错误。

(10) 位 9(高速背对背允许)：该位用于控制一个主设备是否可以对不同的设备做快速的背对背传输，该位是一个读写位并且是可选项。如果所有的目标设备都允许背对背传输时，初始化软件应将该位置 1，也就是允许主设备对不同的目标设备发出快速背对背传输；若该位为 0，则表示只允许对同一个目标设备进行快速背对背传输。

(11) 位 10～位 15：这六位为保留位。

6.2.3 关于设备状态寄存器

状态寄存器(Status)用于记录 PCI 总线有关事件的状态信息。其中每一位的具体定义见图 6.3 所示。

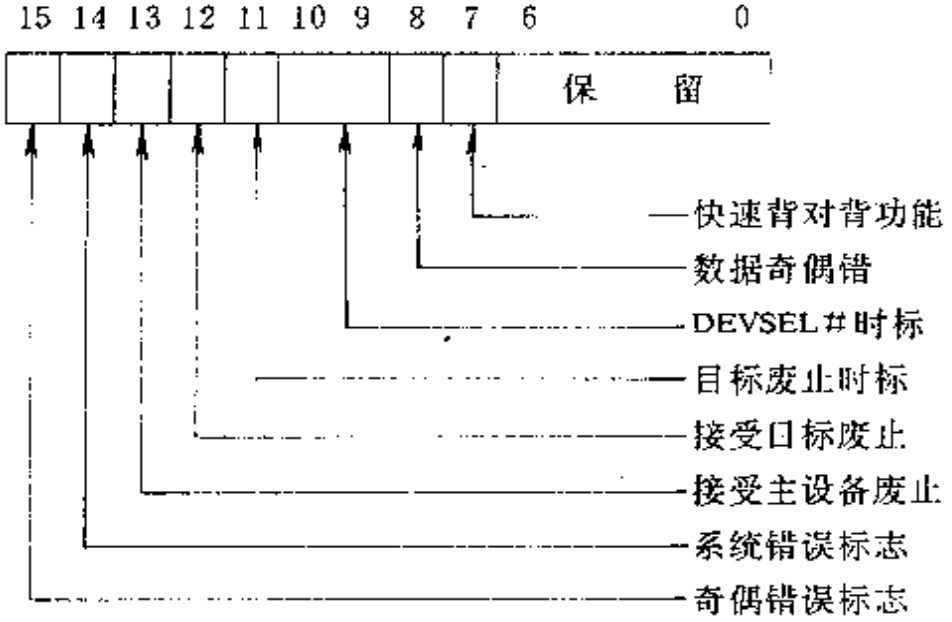


图 6.3 状态寄存器格式

设备可以根据自己的功能来决定实现状态寄存器的哪些位，不一定要实现所有的位。例如一个作为目标的设备，如果肯定不会发出目标废止，则不需要实现位 11。

系统对该寄存器的读操作无特殊要求。而对写操作，由于该寄存器的有些位是可清不可置的，因而有所不同。当向该寄存器写数并且这样的位上(可清不可置的位)所对应的数据为 1 时，则该位被清 0。例如，为了清位 14 而不影响其它位，应向该寄存器写 0100 0000 0000 0000B。

下面说明该寄存器每一位的具体功能。

(1) 位 0～位 6：该七位为保留位。

(2) 位 7(快速背对背功能)：该位是只读位并且是可选项，用来表示一个目标设备是否可以接受对不同设备的快速背对背传输。若该设备有能力接受这种传输，便将此位置 1，否则必须将它清 0。

(3) 位 8(数据奇偶错)：本位只在主设备上实现。当它为 1 时，表示有数据奇偶错误发生。但该位的置 1 需要满足三个条件：

①本设备产生了 PERR＃或者收到了总线上的 PERR＃信号；②只有总线主设备才可以将其传输中发生的奇偶错误表达到这一位上；③命令寄存器中的奇偶错误响应位为 1。

(4) 位 9～位 10(DEVSEL＃时标)：该二位编码用来表示 DEVSEL＃信号的时间关系。其编码值表示三种地址译码速度：00 表示快速，01 表示中速，10 为慢速，11 保留。二位均为只读位。除了配置读/写外，这两位必须表示设备在其它总线命令下发 DEVSEL＃所对应时间中最慢的一个。

(5) 位 11(目标废止标志)：每当目标设备用目标废止方式终止一个传输时，必须先将此位置 1。如果设备永远不会发动目标废止，就不需要实现该位。

(6) 位 12(接受目标废止)：该位必须由传输过程被目标设备以目标废止方式而终止的主设备来设置为 1。也就是说该位应由主设备置 1，并以此来表示已接受了目标设备发动的目标废止操作。所有的主设备都必须实现此位。

(7) 位 13(接受主设备废止)：该位必须由主设备在它以主设备废止方式结束传输时置 1(特殊周期除外)。所有的主设备必须实现此位。

(8) 位 14(系统错误标志)：当设备发 SERR＃信号时必将此位置 1。肯定不会发 SERR＃信号的设备不必实现该位。

(9) 位 15(奇偶错标志)：当设备检测到一个奇偶错误时必须将此位置 1，即使命令寄存器中的位 6 处于非使能状态也应如此。

6.2.4 头标区中其它寄存器的功能

本节所涉及的寄存器都是与设备有关的，只需要在提供某一功能的设备上实现其对应的寄存器即可，而并非普遍需求。

1. 缓存行长度寄存器(Cache Line Size)

该寄存器用来指定系统中高速缓存(Cache)一行的长度，以

32 字节为单位。它是一个可读可写的寄存器。所有能发存储器写并无效命令的主设备必须实现它。每个参加缓存协议的设备都要使用该寄存器，以便知道何时达到 Cache 行边界，从而对突发访问进行再试。当该寄存器为 0 时，设备可以忽略 Cache 支持信号 SBD# 和 SDONE。系统复位时应将此寄存器清 0。

2. 延迟计数器(Latency Timer)

该寄存器以 PCI 总线时钟为单位来指定 PCI 总线主设备的延迟计时值。对于主设备，只要有能力连发两个以上的数据期，就必须将该寄存器实现为可写的寄存器。而那些只能连发两个或者少于两个数据期的设备，可以只读方式实现此寄存器，但有一点，硬件接线必须保证它的值为 16 个总线时钟或者更少。一个典型的实现方法是：把它的高 5 位做成可读可写，低 3 位做成只读，这样计时器的粒度为 8 个总线时钟。系统复位时应将此寄存器清 0。

3. 内含自测试寄存器(BIST)

BIST 是一个可选寄存器，用作内含自测试的控制与状态寄存器。对于不支持 BIST 的设备，该寄存器的读出值必须恒为 0，也就是说，把它当做一个保留的寄存器来对待。当调用一个设备的内含自测试时，不能妨碍 PCI 总线的正常操作。图 6.4 给出了该寄存器的格式。

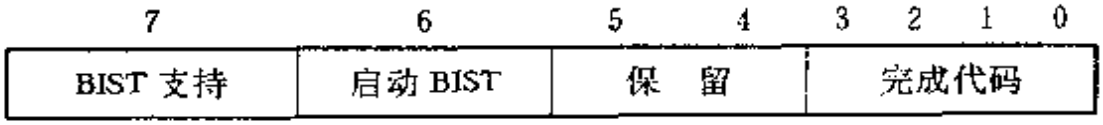


图 6.4 BIST 寄存器格式

其中各位的具体含义如下：

(1) 位 7(BIST 支持)：该位表示设备是否支持 BIST 功能。若此位读出值为 1，表示设备支持 BIST；反之，若读出值为 0，表示

设备不支持 BIST。

(2) 位 6(启动 BIST)：此位用来决定是否启动内含自测试。若向该位写 1，则启动 BIST。当 BIST 完成时设备便将此位清 0。如果经过 2 s 后，BIST 还没有完成，软件就应将此设备看作是失效的。

(3) 位 5～位 4：这两位为保留位，读出值必须为 0。

(4) 位 3～位 0(完成代码)：用来表示设备的内含自测试是否通过。若它们的值为 0 时表示设备通过了自测试；否则，表示设备失效。

4. 中断线寄存器(Interrupt Line)

中断线寄存器是一个 8 位的寄存器，用来报告中断线的连接情况。它是一个可读可写的寄存器，并且凡是使用了一个中断引脚的设备都必须实现它。POST 例程(加电自测试程序)在系统进行初始化和配置时要将中断连线信息写入该寄存器。

这个寄存器的值表示设备的中断引脚与系统中断控制器的哪一个输入端相连。设备驱动程序和操作系统可以利用这个信息来确定中断的优先级和向量。该寄存器的值要受系统体系的支配。对于 X86 兼容 PC，这个寄存器的值与标准 8259 配置中的 IRQ 编号(0～15)相对应，255 表示没有连到中断控制器，15～255 之间的值为保留值。

5. 中断引脚寄存器(Interrupt Pin)

该寄存器的值表示设备(设备功能)使用了哪个中断线。其值为 1 时表示使用 INTA#，其值为 2 时对应 INTB#，而 3 和 4 分别对应于 INTC#及 INTD#。如果设备(设备功能)没有使用中断线，则必须将该寄存器清 0。这个寄存器是一个只读寄存器。

6. MIN—GNT 和 MAX—LAT 寄存器

这两个寄存器都是只读寄存器，用来指定设备对延迟计时器

的设定值，单位为 1/4 μs。如果它们的值为 0，则表明设备对延迟计时器没有特殊的要求。

MIN—GNT 用于指定设备需要多长的突发传输时间。MAX—LAT 用来表示设备对 PCI 总线进行访问的频繁程度。

设备在指定这两个值的时候要考虑两个因素：一是要能最有效地利用 PCI 总线；二是要能有效地利用自己的内部资源。

6.2.5 基址寄存器(Base Addresses)

PCI 设备可以在地址空间中浮动是 PCI 总线中最重要的功能之一，它能够简化设备的配置过程。在系统上电时，与设备无关的系统软件必须确定有哪些设备存在，同时建立一个统一的地址映射关系并确定一个设备是否有扩展 ROM。

1. 地址映射

加电软件在引导操作系统之前必须要建立一个统一的地址映射。也就是说，必须确定在系统中有多少存储器以及系统中的 I/O控制器要求多少地址空间。当这些信息确定之后，加电软件便可以把I/O控制器映射到合理的地址空间并引导系统。为了使这种映射能够做到与相应的设备无关，从而在配置空间的头标区中安排了一个供映射时使用的基址寄存器。

在所有的基址寄存器中，位 0 均为只读位并且用来决定是存储器空间还是 I/O 空间。如果该位为 0 则是映射到存储器空间，否则，若为 1 表示映射到 I/O 地址空间。

映射到 I/O 空间的基址寄存器宽度总是 32 位，其中位 0 恒为 1(用硬件实现)，位 1 为保留位并且其读出值必须为 0，其余各位用来把设备映射到 I/O 空间。

映射到存储器空间的基址寄存器可以是 32 位宽度，也可以是 64 位宽度(支持映射到一个 64 位地址空间时)。其中位 0 也要用硬件方法使其恒为 0。而位 2～位 1 两位用来表示映射类型，具

体如下：当这两位等于 00 时，表示基址寄存器为 32 位宽，可以在 32 位表示的存储器地址范围的任何地方进行映射；等于 01 时，表示基址寄存器为 32 位宽，但必须映射到 1 MB 以下的存储器地址空间；等于 10 时，表示基地址寄存器为 64 位宽，可以映射到以 64 位表示的存储器空间的任何地方；最后一个编码 11 为保留编码。至于位 3，若数据是可预取的，就应将它置为 1，否则清 0。该寄存器的其余各位用来将一设备映射到存储器空间。

一个设备可以把一个地址范围标记为可预取的，但必须保证如此做法不会影响读操作，设备在读取时返回全部字节而与字节使能信号无关，主桥在可预取的范围内可以将处理机写操作合并而不致于引起错误。

任何设备若有一个地址范围的行为像正常的存储器，但不参加 PCI 的缓存协议，就应该标记为可预取的。例如，图形设备中的一个平面帧缓冲区。

前面已经提到存储器空间的基址寄存器宽度可以是 32 位或 64 位，但在具体实现过程中，除了低 4 位应符合上述要求外，其高位部分的实际实现位数要根据映射多大的地址空间范围来决定，32 位或 64 位的限制只是一个上限。例如，一个设备希望 1 MB 的地址空间(用 32 位基址寄存器)时，应实现寄存器的高 20 位，而其它位用硬件方法保证为 0。

加电软件通过向存储器空间基址寄存器写入全“1”然后将其读回，便可以确定设备要求多大的地址空间。与地址空间无关的位其返回值为 0。

在配置空间中，从偏移地址 10 h 开始，为基址寄存器分配了六个 DWORD 单元位置。第一个基址寄存器总是在 10 h 处，第二个基址寄存器要根据第一个的长度来决定它的偏移地址，可能是 14 h 或 18 h。依此类推，后面的基址寄存器的偏移地址都要由前面寄存器的大小来决定。

一个典型的设备将要求一个存储器范围为其控制功能服务。有的图形设备可能要用两个这样的地址空间，其中一个用于控制功能，另一个作为帧缓冲。如果一个设备希望将控制功能同时映射到存储器空间和 I/O 空间，那么它就必须实现存储器型和 I/O 型这两个基址寄存器。设备总是应该允许它们的控制功能映射到存储器地址空间中。

2. 扩展 ROM 基址寄存器(Expansion ROM Base Address)

有些 PCI 设备，尤其是那些准备用于 PC 结构插接模块上的设备，需要局部的 EPROM 作为扩展 ROM(其目的见 6.3 节)。为此，在配置空间偏移地址 30h 处开始定义了四个字节的寄存器，用来处理这个扩展 ROM 的基地址和大小。

该寄存器和 32 位基址寄存器相比，除了位的编码和用途不同之外，其它功能完全相似。它的高 21 位对应于扩展 ROM 基地址的高 21 位。一个设备实际实现的位数取决于该设备支持什么样的地址边界。例如，一个设备允许它的扩展 ROM 映射到任意 64k 边界时，它就应该实现此寄存器的最高 16 位，其它 5 位用硬件方法使它们恒为 0。凡是支持扩展 ROM 的设备，必须实现这个寄存器。

该寄存器的位 10～位 1 为保留位。与设备无关的配置软件通过对扩展 ROM 基址寄存器的地址位上写入全 1，然后再读回以确定设备支持各种地址边界。所有的无关位上都返回 0，从而有效地指出了地址边界，也就知道了设备要求的这一块地址空间的大小。

这个寄存器的位 0 用来控制相应的设备是否能够接受对其扩展 ROM 的访问。当该位为 0 时，禁止访问设备的扩展 ROM 地址空间；当该位为 1 时，允许将本寄存器的其它位作为参数进行地址译码。这样在系统配置时，就允许将一个设备用于有或者没有扩展 ROM 的环境中。命令寄存器中的存储器空间位优先于扩展

ROM 的位 0(使能位)，但是，如果存储器空间位扩展 ROM 的使能位同时为 1 时，那么设备就必须响应对其扩展 ROM 的访问。

为了减少一个设备上地址译码器的需要数量，可以使扩展 ROM 基址寄存器和其它的基址寄存器之间共享一个译码器，当扩展 ROM 译码使能位(位 0)有效时，该译码器就用于对扩展 ROM 的访问，而与设备无关的软件绝对不能通过任何其它的基址寄存器来访问相应的设备。

3. 插接存储器

2.0 版本以后的 PCI 规范中将定义一个能够处理插接存储器的机构。该机构将规定一个新的头标类型值用于指定插接存储器的配置寄存器。这些规定将允许对插接存储器设备进行自动检测、容量测定和配置设定。

6.3 PCI 扩展 ROM

在 PCI 规范中提供了一种机制，使 PCI 设备可以带一个扩展 ROM，通过执行其中存放的代码来完成与设备有关的初始化，同时也有可能完成系统引导功能。该机制允许扩展 ROM 含有几个不同的映象，以适应不同的机器和处理器结构。本节讨论有关扩展 ROM 中代码映象的放置和其它一些必要的信息。

注意：凡是支持扩展 ROM 的设备，必须允许按任意的字节组合方式可对 ROM 进行访问，特别强调的是必须支持双字访问(DWORD)。

扩展 ROM 中的信息安排要与现有的适合于 ISA、EISA 和 MC 适配器的 Intel X86 扩展 ROM 的头标区相兼容，同时还要支持其它体系结构的机器。头标区中所给信息经过了扩充，从而使适配器的功能进一步优化使用，以致于扩展 ROM 中的代码在运行期间所使用的存储器空间总量可以最小。

PCI 扩展 ROM 的头标信息支持如下功能：

(1) 提供一个长度代码，用以表示 PCI 设备的 ROM 映象在初始化时，所需连续地址空间的总量。

(2) 提供一个标记，用来表示在 ROM 地址空间上的每个 ROM 映象中的可执行代码或解释性代码属于哪一种类型。

(3) 提供了 ROM 中代码或数据的修改次数。

(4) 具有指向设备最重要的产品数据的指针。

(5) 提供它所支持的 PCI 设备的供应商识别代码和设备识别代码。

在 PCI 扩展 ROM 和标准的 ISA、EISA 及 MC 总线 ROM 的使用方法之间有一个重要的区别，那就是 PCI 扩展 ROM 代码从不在原地执行，而是将代码从 ROM 中拷贝到 RAM 中，然后在 RAM 中执行。这样便可在初始化和运行时动态地确定代码长度，并且能够改善代码的执行速度。

6.3.1 PCI 扩展 ROM 的内容

PCI 设备的扩展 ROM 能够存放多种处理机体系结构的可执行代码或解释性代码。这个可以在单个物理的 ROM 中实现，该 ROM 中可以包含为不同的系统和处理机体系结构而设的多个代码映象。每个映象必须开始于一个 512 字节边界并含有 PCI 扩展 ROM 头标。每个映象的开始位置取决于前一个映象的长度。ROM 的最后一个映象的头标中，要有一个特殊的编码以表示它是最后一个。

1. PCI 扩展 ROM 的头标格式

每个 ROM 映象中所要求的信息分为两个不同的区域：一个区域为 ROM 头标，应放在 ROM 映象的开始处；另一个区域是 PCI 数据结构，必须放在映象的第一个 64k 中。

PCI 扩展 ROM 头标的格式如表 6.9 所示，其中的偏移量是从

映象的开始处算起，并以 16 进制数表示，每个字段的长度以字节为单位来表示。

表 6.9　PCI 扩展 ROM 头标格式

偏　移	长　度	值	说　　明
0h	1	55h	ROM 标签字节 1
1h	1	AAh	ROM 标签字节 2
2h～17h	22	VV	保　留
18h～19h	2	VV	到 PCI 数据结构指针

ROM 标签字段的总长度为两个字节，第一字节的值为 55h，第二字节为 AAh。此标签在 ROM 的每个映象中都必须设置，并放在各自 ROM 地址空间的前两个字节处。

PCI 数据结构指针的长度也是两个字节，用来指出 PCI 数据结构的起始偏移地址，而且以 ROM 映象的起始位置为参考点。

2. PCI 数据结构格式

PCI 数据结构必须放在 ROM 映象的前 64k 字节内，并且要做到双字对齐。具体格式见表 6.10 所示。

表 6.10　PCI 数据结构格式

偏　移	长　度	说　　明
0	4	标签，字符串“PCID”
4	2	供应商识别码
6	2	设备识别码
8	2	对重要产品数据的指针
A	2	PCI 数据结构长度
C	1	PCI 数据结构修改
D	3	分类代码
10	2	映像长度
12	2	代码/数据的修改级别
14	1	代码类型
15	1	指示标志
16	2	保留

表中各个字段的含义如下：

(1) 标签：用四个字节提供一个唯一的 PCI 数据结构标签"PCIR"，其中一个字符占一个字节，也就是"P"在偏移 0h 处，"C"位于 1h 处，其余类推。

(2) 供应商识别码：是一个 16 位长度的字段，其内容与配置空间头标中的完全相同。

(3) 设备识别：该字段的长度为 16 位，其内容与配置空间头标中的完全一致。

(4) 重要产品数据指针：该字段的长度是 16 位，用以指出重要产品数据(VPD)的存放偏移地址。VPD 必须放在 ROM 映象的前 64k 字节中。当该指针值为 0 时，表示 ROM 映象中没有 VPD。关于 VPD 结构的具体内容有待于将来定义。

(5) PCI 数据结构长度：是一个 16 位长度的字段，表示数据结构的长度，并且以字节为单位。

(6) PCI 数据结构修改(版本)：是一个 8 位长度的字段，用以表示数据结构的修改版本号。

(7) 分类代码：该字段的长度为 24 位，其内容与配置空间中的分类代码字段相同。

(8) 映象长度：此字段的长度为两个字节，用来表示映象的长度，并以 512 字节为单位。

(9) 代码/数据版本：是一个 2 字节长度的字段，用来指出映象中代码的修改次数。

(10) 代码类型：该字段的长度为一个字节，用来表示 ROM 映象中的代码是什么类型。此代码是为特定的处理机和系统体系结构而设置，可能是可执行的二进制代码，也可能是解释性代码。其代码类型编码如下：

0　　Intel X86，PC…AT 兼容

1　　PCI 使用的 OPENBOOT 标准

2～FF　　保留

(11) 指示标志：该字段的长度为一个字节，其中的位 7 用来表示该映象是不是 ROM 中的最后一个。若该位为 1，则表示该映象是最后一个映象；若为 0，则表示后面还有映象。位 6～位 0 为保留位。

6.3.2　加电时自动测试代码(POST)

在大多数情况下，系统的 POST 代码对插接的 PCI 设备的处理相同于对那些焊接在母板上的设备的处理。但对扩展 ROM 的处理是不同的。POST 代码分两步来检测一个选项 ROM 是否存在，其中第一步确定设备是否在配置空间中实现了一个扩展 ROM 基址寄存器，如果该寄存器存在，POST 就必须将 ROM 映射到地址空间中未用的部分并将使能位置 1；第二步检查前两个字节是否为标签值 AA55h，如果是，就表示存在一个 ROM，否则就说明该设备上没有 ROM。

如果检测到设备有一 ROM，POST 就必须在该 ROM 中寻找一个具有合适的代码类型的映象，同时该映象中的供应商识别和设备识别字段也要与设备配置空间的相应字段吻合。当这个合适的映象找到之后，POST 就从 ROM 中拷贝适当数量的数据到 RAM 中，然后执行该设备的初始化代码。至于拷贝多少数据量，以及怎样执行设备初始化代码，要由代码类型来决定。

6.3.3　PC 兼容的扩展 ROM

本小节将进一步说明对 ROM 映象的要求以及在 PC 兼容系统上如何处理 ROM 映象。该说明适用于在 PCI 数据结构的代码类型字段中指定了 Intel X86 和 PC—AT 兼容的任何映象以及 PC 兼容的任何平台。

1. ROM 头标的扩展

为了保证 PC 系列机的兼容性，对 PCI 扩展 ROM 映象的标准头标进行一些扩充，即增加了两个字段。一个位于偏移地址 02h 处，其作用是为映象提供初始化长度；另一个位于偏移地址 03h 处，目的是为 ROM INIT(ROM 中断)提供入口地址。扩展后的 ROM 头标如表 6.11 所示。

表 6.11 扩展后的 ROM 头标格式

偏 移	长度	值	说 明
0h	1	55h	ROM 标签字节 1
1h	1	AAh	ROM 标签字节 2
2h	1	vv	初始化长度(代码长度以 512 字节为单位)
3h	4	vv	INIT 功能入口点，POST 对此位置做一次远调用
7h～17h	17	vv	保留(每个应用有独特数据)
18h～19h	2	vv	到 PCI 数据结构的指针

2. POST 代码的扩展

系统中的 POST 代码根据初始化长度字段所指定的字节数，将数据从 ROM 拷贝到 RAM 中，然后以 03h 处的值作入口点调用 INIT 功能。在 INIT 中断返回之前，POST 代码应将上述拷贝过来的 RAM 区保持为可写状态，以便 INIT 代码能在该区存放一些静态数据，并调整运行存储分配，使得系统在运行时耗费较少的空间。

当涉及到扩展 ROM 时，在系统 POST 代码中有一套为 PC 兼容而设的特别步骤，它们是：

(1) 根据初始化长度指定的字节数从 ROM 中拷贝数据到 RAM 中。

(2) 保持 RAM 区为可写状态并调用 INIT 功能。

(3) 利用偏移地址 02 h 处的值决定运行时需要多少存储器空间。

在系统引导之前，POST 代码必须把含有扩展 ROM 代码的 RAM 区变为只读区。

POST 代码必须用特殊的方法去处理带有扩展 ROM 的 VGA 设备。VGA 设备的扩展 BIOS 必须拷贝到 0C0000h 处。VGA 设备的识别是通过检查设备配置空间中的分类代码字段来实现的。

3. INIT 功能的扩展

在 PC 兼容的扩展 ROM 中，含有一个 INIT 功能，用来负责 I/O 设备的初始化以及为运行存储分配操作做准备。由于在执行 INIT 功能时代码所驻留的 RAM 区被保持为可写状态，因此，允许 PCI 扩展 ROM 中的 INT 功能具有某种扩展能力。

在 INIT 功能执行期间，INIT 功能可以在它的 RAM 区中存放静态参数，然后，运行存储分配 BIOS 或设备驱动程序便可调用这些参数。但是该 RAM 区在运行存储分配期间是不能写入的。

象中。

INIT 从入口处可以得到三个参数：设备的总线编号、设备编号和功能编号。这些参数可以用来访问正在被初始化的设备。它们被传送到 X86 的寄存器 AX 中，其中 AH 包含总线编号，AL 的高 5 位为设备编号，AL 的低 3 位是功能编号。

4. 映象结构

一个 PC 兼容的映象有三个长度与映象结构有关，一个是运行存储分配长度，一个是初始化长度，还有一个是映象长度。映象长度是映象的总长度，它必须大于或等于初始化长度。

初始化长度表示映象中所含的初始化代码和运行存储分配代码两部分的总和。这也是在执行初始化例行程序之前，POST 代码要拷贝到 RAM 区中的数据总量。初始化长度必须大于或等于运行存储分配长度。拷贝到 RAM 区中的初始化数据的检查和必须为 0。

运行存储分配长度是指映象中所含运行存储分配代码的总量。这也是系统运行时 POST 代码保留在 RAM 区中的数据量。这部分映象的检查和必须为 0。

如果有运行存储分配时，PCI 数据结构必须包含于映象的运行存储分配部分中，否则它必须包含于初始化部分。图 6.5 表示了扩展 ROM 中一个映象的典型布局。

6.3.4 设备驱动程序

PCI 设备有两个特征使得它的驱动程序与"标准的"或现有的设备驱动程序不同。第一个特征是 PCI 设备是可以再定位的。也就是说设备的地址空间不是硬件固定的，PCI 设备驱动程序及其它配置软件应当用该设备配置空间中的映射信息来决定将设备映射到何处。第二个特征是 PCI 中断是共享的。因为在系统实现中，很有可能将多个设备连到一条中断线上，这就要求 PCI 设备驱动

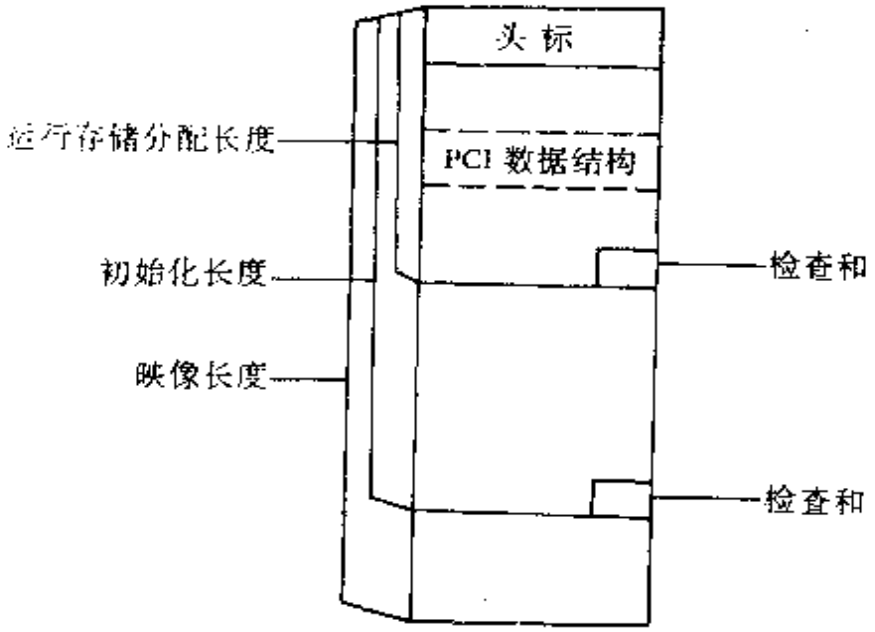


图 6.5 典型的映象布局

程序能够支持共享的中断。关于中断共享的具体实现方法因操作系统的不同而不同，在此不做说明。

有些系统可能无法保证在中断送到 CPU 之前，将数据送到主存。该问题如果处理不当，可能会引起数据一致性问题或数据的丢失。对于该问题有三种方法可以解决：

(1) 系统硬件可以保证在中断传递到处理器之前冲洗投递缓冲区(Posting Buffers)。

(2) 在中断发出之前，设备可以以中断方式对刚写的数据进行读取，这也可冲洗投递缓冲区。

(3) 在设备进行写访问之前，该设备的驱动程序可以完成对这个设备中任何寄存器的读操作，而这个读操作同样能冲洗投递缓冲区。

第七章　PCI 总线开发

计算机技术水平标志着一个国家现代化水平和国防力量的强弱。因此，我国的计算机产业不能完全依赖国外的产品，特别是军事、科研、金融行业，一定要由自己的计算机唱主角。由于计算机产品更新换代很快，所以新品的开发速度就显得尤其重要。本章将讨论有关 PCI 总线产品的开发问题。

7.1　PCI 总线的组件及其产品

7.1.1　PCI 组件概念

虽然许多厂商纷纷宣布支持 PCI 总线产品，掀起了阵阵 PCI 热潮，但市场上有关 PCI 总线的标准产品仍然不是主流。PCI 总线是先有标准然后才有产品，这和以前先有产品后有标准的模式不同。最先按 PCI 总线标准推出基础芯片产品的厂商，当然是该标准的提出者 Intel 公司，其产品是 X86 系列处理器到 PCI 总线的桥接电路。为了与 ISA/EISA 总线兼容，Intel 公司又推出了 ISA 与 PCI、EISA 与 PCI 之间进行转换的大规模集成电路。这些芯片配合起来构成了微机系统，并称之为 PCI 组件(芯片组)。芯片组有二片组或三片组不等。

另外，Digital 公司生产了一种 PCI—PCI 的桥接电路，其目的是为了改善 PCI 系统的负载能力，支持多处理并发工作。

总之，PCI 总线大规模集成电路按功能可分为三类：

(1) 各类处理器到 PCI 总线之间的大规模集成电路；

(2) 各类总线转换到 PCI 总线的大规模集成电路；

(3) 各种高速外设到 PCI 总线的大规模集成电路。

具体见表 7.1。

在表 7.1 中：A 表示处理总线与 PCI 总线桥接电路；B 表示 Cache 控制器及 Cache 容量；C 表示存储控制器及存储器容量；D 表示 PCI 总线仲裁器及支持 PCI 总线的条数。Intel 公司的组件为三片组，适用于 Pentium 处理器；IBM 公司的组件为二片组，适用于 Power PC、i486 系列处理器；DEC 公司的组件为四片组，适用于 DEC 21064 处理器。

在以 PCI 总线为系统总线的计算机系统中，允许多条总线同时存在，这就是多总线的概念。它在很大程度上提高了系统的数据处理能力，从而使高档服务器、多媒体计算机、工作站的升级换代大大加快。在多总线系统中，PCI 总线组件的使用如图 7.1 所示。

7.1.2 PCI 组件产品及供应商

Intel 公司推出的三片组 PCI 总线大规模集成电路 80434NX，可与 Pentium 处理器桥接，它自身含有 Cache 控制器、存储控制器、PCI 控制器，可以完成处理器总线到 PCI 总线的桥接控制。Cache、存储控制器使大批数据的高速传输成为可能。为解决速度匹配和并发工作问题，该公司又推出了 80433NX 数据缓冲器；为使 PCI 总线与 ISA、EISA 总线桥接并与市场上流行的产品兼容，该公司还推出了 82378EB 和 82375EB 两种芯片。Intel 公司 PCI 组件产品的有关数据见表 7.2。

除了上表之外，Intel 公司最近又推出了两种组件。其一为 82430MX 便携式 Pentium 芯片组，内含支持 DRAM、L2 Cache 和 PCI 的系统控制器 82437；PCI 到 Pentium 总线的总线桥 82438；支持 ISA 总线和 IDE 接口的设备控制器 82371，它是一个四片组。该芯片支持 512KB 的 L2 Cache 和 128 MB 的页方式或 EDO

表 7.1 PCI 总线组件一览表

1 2 3	· Cache 控制器 · Memory 控制器 · PCI 控制器	内装 A*	内装 B*	内装 C*	内装 D*	PCI—ISA	PCI—EISA	PCI—PCI	数据缓冲器	适用 CPU
Intel 组件	82434NX(1,2,3)	√	√512K	√512M	82375EB(4)		82374EB		82433NX	Pentium
	82434 2X(1,2,3)	√	√512K	√160M	82378EB(4)	82378EB			82423EX	iDx,i486 系列
IBM 组件	IBM27—82654(2,3)	82653	无	82564A 256M	82654A(6)				82653	Power 001,603,604
	IBM27—82454(2,3)	√	无	√256M	无				82454	i486,33Mhz
DEC 组件	27071 CA(1,2)	√	16M	√2G	无			21052	27071BA	21064
	27021 DA(3)	√	16M	√2G	无			21052	27071BA	21064
TI 组件	MPUT1486SXL(2,3)	√	√8K	√64M	基础 I/O PUU(2)			PCU(PCI)总线的 PC 卡控制器		T1486XL
VCSI 组件	VL82C591(1,2,3)	√	√	√1G		VL82C593			82C592	Pentium

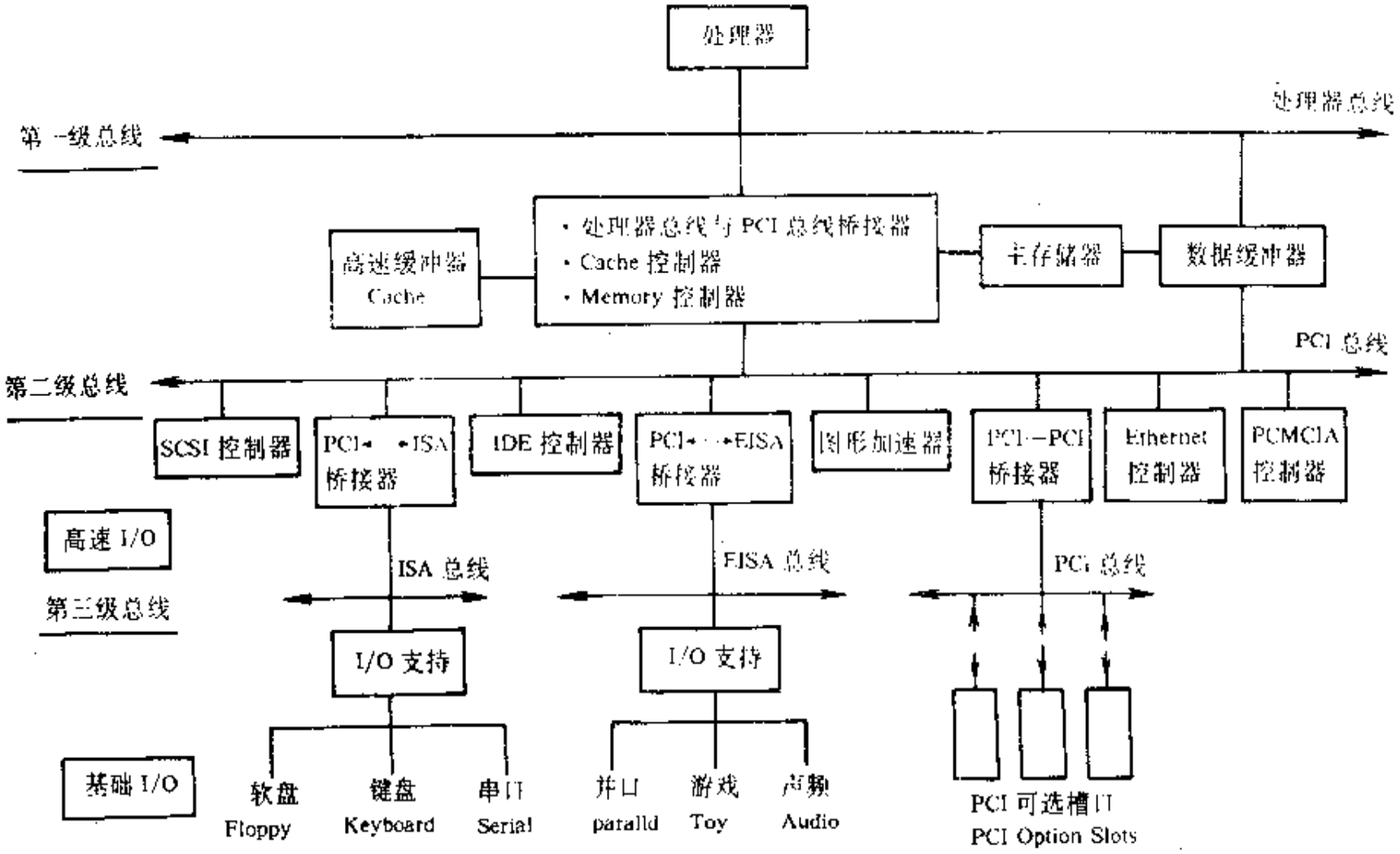


图 7.1　多总线结构及 PCI 组件应用

DRAM，可达到 66 MHz 的 Pentium 总线速率，支持即插即用。

表 7.2 Intel 组件概况

	名 称	功 能	封 装	电 源	价 格（美元/只）
一组	82434NX	Cache、Memory PCI 总线控制器	208PQFP	+5 V,3.3 V	120
	82375EB (82374EB)	PCI－EISA 总线转换	208PQFP	+5 V,3.3 V	150
	82433NX	数据缓冲器	160PQFP	+5 V,3.3 V	—
二组	82434LX	Cache、Memory PCI 控制器	208PQFP	+5 V	120
	82375EB (82374EB)	PCI－EISA 总线转换	208PQFP	+5 V	150
	82433ZX	数据缓冲器	160PQFP	+5 V	—
三组	82424ZX	Cache、Memory PCI 控制器	208PQFP	+5 V	50
	82378ZB	PCI—ISA 总线转换	208PQFP	+5 V	90
	82423EB	数据缓冲器	160PQFP	+5 V	—
四组	82425EX	系统控制器	208PQFP	+5 V	38
	82426EX	与 ISA 总线桥接	160PQFP	+5 V	—

另一种是 8240 PCI 组件，为三片组。该芯片组内含一个 Cache DRAM 控制器(CDC)、一个数据通路单元(DPU)和一个 PCI 总线与传统的 ISA 总线之间桥接所用的系统 I/O(SIO)；具有对高速缓存以回写方式或写穿方式进行操作的功能；支持 Intel 公司 486 系列的微处理器。其中 CDC 具有两个端口，一个为主端口，另一个为 PCI 端口。CDC 可配置成 Cache 方式或 4－1－2－1 DRAM 访问方式，可处理的 Cache 容量为 64 KB～512 KB，可处理的 DRAM 容量为 2 MB～128 MB。DPU 是一具有主端口、DRAM 端口和 PCI 端口的三端口器件。SIO 是一个 PCI 总线到标准总线之间进行桥接的双端口器件，具有 32 位快速 DMA 控制器。该芯片组为 208 引脚 QFP 封装形式，单价为 52.80 美元。

NCR 公司推出了 SCSI PCI 处理器。Weitek 公司推出了图形接口集成电路。Western Digital 公司推出了 AT—IDE 器件以支持 PCI 总线。随着 PCI 局部总线的进一步发展及相应组件的上市，IBM、DEC、TI、VLSI 等公司也推出了许多产品。其中美国 VISI

技术公司的产品主要有 VL82C591、VL82C592 和 VL82C593 三种。VL82C591 用于和 Pentium 处理器桥接；VL82C592 为数据缓冲器；VL82C593 是 ISA 总线转换器。

IBM 公司的 IBM27—82454 与 i486 系列处理器兼容，能在一片内完成存储控制、数据缓冲、PCI 桥接等多种功能，采用 3.3 V 电源和方形扁平封装，引脚为 240 条。

除 X86 系统处理器外，其它处理器也有相应的 PCI 总线桥接芯片组，如 DEC21064、Power PC601、Power PC603、Power PC604 等。

7.1.3 外围高速处理芯片

外围高速处理大规模集成电路由外设专业厂商开发生产，它能直接和 PCI 总线桥接，并享受 PCI 提供的高速数据传输服务。此类高速外围设备处理芯片主要有：图形加速芯片、多媒体视频动态图像芯片、IDE 控制器、SCSI—2 及 SCSI—3 控制器和 Ethernet 控制器等。

1. 图形加速芯片

• Trident 公司

其系列图像处理芯片有 8900C、8900D 和 9000B，与 SVGA 完全兼容，称为 TVGA 方式，在国内比较流行。是目前 386、486 系列微机的通用显示卡。该公司将推出 64 位的高效的 GUI 加速芯片 TGUI9660，其主要参数为：

a. 支持 1280×1024 分辨率 64 K 种颜色。

b. 支持 800×600 分辨率 16 M 真彩色。

c. 支持 Intel PCI 2.0 规范，无须外围逻辑芯片而直接挂在 PCI 总线上。

d. 可以控制 PCI 缓冲器的组合。

e. 硬件支持各种窗口图形加速功能。

f. 支持硬件游标和最新的 Window RAM。

• 美国 CIRRUS LOGIC 公司

CIRRUS LOGIC(凌云公司)的外围处理和控制方面的集成电路具有领先地位。该公司的图形加速器主要有 CL—GD5470/GD5471/GD5472，其技术指标为：

a. 是与 64 位 2D/3D VGA 兼容的窗口和复制加速器。

b. 支持分辨率 1280×1024×24 位逐行扫描，1600×1200×8 位隔行扫描。

c. 支持 16 位缓冲区。

d. 32 位的主机接口，支持 PCI 总线的突发传输。

e. 135 MHz 24 位专用调色板 DAC，GD5472。

f. 100%软件兼容及结构映射的硬件支持。

g. 0.8 μm 工艺，208 引脚 PQFP 封装。

该芯片为三片结构，其主要功能为：GL—GD571 是一高性能的 VGA 控制器，可以直接与 PCI 总线接口；GL—GD572 为调色板 DAC；GL—GD5470 为 2DGUI 和 3D 复制加速器。

另外，该公司还生产一种 VISAVL—BUS 到 PCI BUS 的总线桥 PT80C286 Juniper，其指标如下：

a. 单片式的 VL—BUS→PCI BUS 的总线桥接器。

b. 符合 PCI 总线 2.0 版标准和 VL—BUS 2.0 版标准。

c. 支持从直流到 33 MHz 的 PCI 总线频率，也支持 VL—BUS 最大频率 50 MHz(5 V)和 33 MHz(3.3V)频率。

d. 两个内置式总线仲裁器，支持四个 PCI 总线控制器。

e. 同步总线，工作于 5 V，33 MHz。

f. 0.8 μm CMOS 技术，208 引脚 PQFP 封装形式。

• 美国 S3 Incor Porated 公司

该公司的图形加速芯片主要有：Trio32 和 Trio64 图像加速芯片，基于 DRAM 的 Vision868 和基于 VRAM 的 Vision968 图形/视

频加速器。它们都支持 PCI 总线标准。下面以 Vision968 为例来说明其功能和技术指标：

a. 图形/视频双加速和 64 位视频存储接口。

b. 与主流操作系统驱动软件兼容。

c. 支持 VGA 和 SVGA 显示方式。

d. 支持 Microsoft 的 Windows 视频及 MPEG 格式。

e. 支持工业标准局部总线，32 位 PCI 总线和 VL 总线。

f. 分辨率可以为 1024×768、1280×1024、1600×1200，具有 16.7 M 种颜色。

g. 支持多媒体集成功能、彩色空间转换及图像压缩等。

h. 视频 BIOS 支持 RAM DAC。

i. 灵活的系统设计，32/64/128 位 SID RAM DAC。

j. 支持绿色 PC 和即插即用功能，208 引脚 PQFP 封装形式。

另外，美国的 ARK LOGIC 公司也推出了一种新一代 64 位 ARK 2000 PV 系列图形加速芯片，显示时钟频率高达 220 MHz，支持 1～4 M 显示缓存、1600×1280 分辨率、64 K 种颜色和 PCI 总线。

2. 多媒体动态视频显示芯片

该类芯片目前主要以 S3 公司和 ARK COGIC 公司的产品为代表。

S3 公司的多媒体加速芯片组将声音、图像合二为一，支持 PCI 总线，是三片组，主要有：S3 Trio64 V＋TV（基于 64 位 DRAM，数字视频综合图形及视频图像加速器，直接挂接 PCI 总线）、S3 Scenic/MX1、PCI MPEG—1、Audio/videv Decoder、声频/视频 MPEG—1 解码加速器、Sonic/AD AUDIO DAC 及可编程声频 DAC 等。S3 公司的多媒体加速芯片组，使原来的多媒体套板由三块合成一块卡，其 PCI 总线的界面，满足 30 帧/s 视频图像的数据传输要求。

美国 ARK LOGIC 公司的多媒体加速器芯片组主要是 ARK 2000MI，它集 GUI 图形加速器、MPEG 解压回放、16 位音频处理和视频捕捉为一体，是合四为一的多功能多媒体板卡。

3. 外部存储设备芯片

采用 PCI 总线标准的硬盘控制器芯片，可以提高系统与硬盘之间的数据吞吐量。将多种控制器功能集成在一块芯片上，可以使多种控制器共享同一 PCI 总线接口，减少所用器件数，减轻 PCI 总线的电气负载。

• IDE 控制器

通常，与 PCI 总线接口的 IDE 硬盘控制器均安装在台式个人机中，以便大幅度提高硬盘的吞吐量，使其达到 11 MB/s 的传输速度。该控制器适用于 PCI 总线的 IDE 控制。这样的 IDE 控制器由美国的 Adaptec 公司开发生产。

• SCSI 硬盘控制器

当前，高档微机的硬盘已达到 500 M 以上，使用 SCSI 硬盘接口已成为一种新时尚。为此许多外设厂家相继推出 SCSI—2、SCSI—3 硬盘控制器，最高数据传输率可达 10 MB/s 以上。

4. 网络芯片产品

在该领域，美国 DEC 公司的产品占主流。3COM 网络芯片，已被 Microdyne 公司和 SVEC 公司采用。一般将这种利用 PCI 总线技术的网卡产品称为高速网络产品。在网络上信息量急剧增大的情况下，PCI 总线网络产品的推出，将缓解不少用户的燃眉之急。

最近，Cogent Data Technologies 公司和 Znyx 公司均推出了四段 PCI 以太网适配器，其型号分别为 EM964 PCI Quartet TP 和 EtherArray 2×314。该适配器能将服务器与四个独立的以太网段相连，均支持双工方式，即用户能在以太网两条不同的通道上同时发送和接收数据。全双工方式使服务器与交换器经由一条以太网链的总吞吐量达到 20 MB/s。

上述两家公司都采用专用软件来简化安装工作。Cogent 的诊断程序能够检测到 PCI 的 BIOS 芯片是否识别到插件板。Znyx 的诊断工具根据所需的配置信息对 BIOS 编程，从而使适配器安装运行起来。两种插卡都提供了面向 Netware 3. X、4. X 以及 Microsoft Windows NT3. 1 的开放式数据链路接口(DDI)驱动程序。但 PCI 不支持 Microsoft LAN Manager、Microsoft Windows for Workgroups、Microsoft Windows NT、OS/2 LAN Server 以及 Pathworks，而 Ether Array 对上述环境全部支持，此外，它还支持 Artisoft 的 LANtastic、Unix TCP/IP for Sco Univel 及 Sun soft 应用程序。

除了上述各类芯片组件外，还有几种最新组件，这里也作一简要介绍，以供读者参考。

IBM 公司的存储器分部生产了一组 PCI 总线适配器板卡，以对其串行存储体系(SSA)进行扩充。其中最有代表性的是 PNS4—20 和 PRS4—20 RAM 适配器，通过使用 SCSI—2 的命令设置，可使它们与板外的存储设备相互通信，从而使这些 SSA 升级到具有最大数据传输率为 20 MB/s 的 SCSI。

Cypress Semiconductors 推出了一种超级 Cache 芯片组，是一种三片组。内含 128 KB 的二级 Cache 并支持容量为 1 MB 的 2L Cache。该芯片组支持 PCI 总线和 ISA 总线并具有几个标准的 PCI 控制器。其中的三个芯片分别为 CY82C692、CY82C691 和 CY82C693。其主要性能为：

a. 三片：系统控制器、数据通路/Cache 和系统 I/O 控制器。

b. 128 KB 的内部 Cache，能够以 16 K×64 位或 8 K×24 位的方式使用。

c. 支持 PCI 和 ISA 总线。

d. RTC、键盘＋鼠标控制器、IDE(CD—ROM)控制器、中断控制器及 DRAM 控制器。

e. 3－1－1－1 突发 Cache，64 位总线。

f. 支持 1 MB 的 2L Cache，768 MB 的 DRAM。

g. 208 引脚，PQFP 封装形式。

OPTi 公司最近也推出了一种型号为 Vipe—N 的芯片组，它支持 3.3 V 或 5 V 的同步 DRAM，是一种三片组。由 82C556 数据缓冲控制器、82C557 系统控制器和 82C558N 设备控制器组成。通过与该公司的 82C602 缓冲芯片相配合，开发人员可设计出一个包括局部总线 IDE 接口在内的综合性系统。

7.2 PCI 总线开发工具及用途

随着支持 PCI 总线产品开发芯片的不断涌现，相应的软件工具也得到了发展。PCI 总线分析工具已经问世，Vmetro 公司和 New BUS 公司，目前均可提供插于 PCI 总线上的状态和时钟频率分析板。本节将讨论 PCI 总线系统开发测试环境和 PCI 总线板卡开发测试环境。

PCI 总线是一种数据/地址复用总线，也是同步总线。由于频率可以在 0～33 Hz 范围内变化，所以各频段上 PCI 总线的性能并不完全一致，从而使以不同频率工作的板卡可能有不符合 PCI 总线标准的地方。从理论上讲，只要严格按照设计规范进行设计，就不会出现上述问题。但是，实际情况却不尽相同。这样，充分利用专用软件工具来帮助产品的开发和调试，就显得很重要。

1. 基于软件的总线接口模型

Synopsis/Ligic Modeling 公司开发了 VHDL/Verlog PCI 总线模型，它是一种基于 PCI 规范符合核查表的模拟环境。该环境由主/从模块和监控/仲裁模型组成。它支持频率变化的特性对测试最坏情况特别有用。通过改变某项定时参数，PCI 总线模型会产生其它相关的时间参数，并能检测出任何违反 PCI 规范的情况。

全套测试程序共有 27 个测试范例。这些范例可以按 PCI 规

范修改参数，从而生成产品开发者所需要的专用测试程序。它所归纳出的测试结果十分客观、精确。测试完成后，根据核查表，形成详细的报告清单。

该模型和全套测试程序仅仅从数字化的角度检查 PCI 总线功能是否可以运行。如果用户需要检测输出缓冲器的 V/I 曲线，可利用 Spice 模拟程序进行测试。在进行插件板设计时，Spice 模拟程序还可以帮助开发设计人员检测轨迹长度、最高频率、噪声容限、过调和弱调效果。

2. PCI 总线实验器

当一个 PCI 总线系统和插卡设计完成后，并不能说明整个工作已彻底结束。因为还有大量的测试和调试工作要做，而且，这对于一个产品来讲，是必不可少的环节。PCI 的测试和调试工作随 PCI 总线配置的复杂程度而有所不同。一般情况下的做法是：根据检测和调试中读出的数据量大小来判定 PCI 总线工具的效率。

美国 HP 公司生产的 PCI 总线实验器，能生成各类 PCI 总线操作事务，同时能生成 0～33 MHz 的规范变化并对其进行实时监控。它可帮助设计人员开发出足够的测试应用程序，以便更好地生成桥接程序和 PCI 总线主程序。

PCI 总线实验器由四部分组成：

(1) 一个基于 ISA 总线的排序卡，运行总线排序程序可以和实验器上的模式识别硬件进行对话。

(2) 实验器主板上含有总线规约状态机，事务运行存储器和触发器逻辑，还有能使实验器处于消极旁观或积极参与 PCI 事务的运行电路。

(3) 用于连接主板和开发所用 PCI 插槽的适配器 HP E2912A，可取代系统板提供的 PCI 环境。其上含有一个总线仲裁器、一个 PCI 时钟发生器和两个可插入测试卡的 PCI 插槽。

（4）一个 IEEE—488 接口板，用来将开发的 PC 微机系统和 HP16500 A/B 逻辑分析仪连接起来。通过该板可以把 PC 机的数据送给逻辑分析仪，也可以把 PC 机的数据送到总线上进行事务处理。当逻辑分析仪得到 PCI 业务数据后，将其送入编辑器加以改变，然后再送给排序程序，这时排序程序便可作为一 PCI 参与者，并在 PCI 得到事务数据时成为 PCI 总线的主程序。

3. PCI 总线预处理器

在调试产品时，用手动判定每一个临界信号波形和触发点是很困难的。而 PCI 总线预处理器可完成这一复杂、困难的工作。它不仅能和逻辑分析仪一起完成时序分析，而且还具有状态分析能力，倒顺序状态和软件分析功能；支持多种数据处理软件。

相应的产品由 Future Plus System 公司、Corelis 公司和 Biomation 公司分别推出。它们都支持 5 V 环境和 3.3 V 环境的 PCI 总线扩展槽，都以扩充卡形式和 PCI 插卡连接。

4. 板卡测试工具

在制造板卡时，要同时了解和确定板卡的电气特性。有一种电子扩展卡可用以检测 PCI 的板卡。它是一种电流传感扩展卡，专门用来检测过电流消耗。它通过模拟开关来检查板卡底部的镀金插头和顶部的连接器之间的电流消耗。

5. PCI 总线系统开发工具

PCI SIG 集团提供了一套 PCI 总线系统开发工具。包括一块专用卡和一套软件，用来调试 BIOS 和模拟 PCI 总线上不同类型的设备，以确定 POST 功能是否正常。

继 PCI SIG 集团的软件开发工具之后，其它各厂家在推出各自的 PCI 总线支持芯片的同时，也提供了设计指南及软硬件开发工具。

Advanced Micro Devices（AMD）公司、Future Domain 公司及

Sysbios Logic 公司都有相应的软件工具，以帮助用户尽快掌握 PCI 系统的 SCSI 控制器，还提供了 SCSI 全套工具和操作系统级的软件驱动程序。AMD 公司提供了一块样板。DEC 公司提供了 PCI—PCI 桥接芯片相连的开发工具。

7.3 PCI 总线产品的开发

在设计开发一个 PCI 产品时，首先必须了解可供选择的工具，工具会使人事半功倍。“工欲善其事，必先利其器”，对于现代科技产品的开发，仍具有指导意义。

从 PCI 总线目前的发展来看，PCI 总线产品主要有三类，分别是：PCI 总线基础产品，PCI 总线系统产品和 PCI 总线应用产品。

1. PCI 总线基础产品的开发

此类产品包括：①PCI 总线各类规范的制订，各类电气标准及机械标准的制订；②各类微处理器到 PCI 总线的桥接集成电路及各种总线到 PCI 总线的桥接集成电路；③PCI 总线测试工具、测试板卡、测试软件及各类应用产品的开发环境。

对于 PCI 总线基础产品的开发，国外许多厂商都投入了大量的人力物力，但各有侧重。Intel 公司的贡献在于提出了 PCI 总线的规范雏形，联合了几十家大公司，成立了 PCI SIG 集团，制订了 PCI 总线规范 2.0 版。在桥接芯片方面，Intel 公司的 X86 系列处理器和 PCI 桥接芯片组等产品都得到了用户的认可。PCI 总线基础产品的推出，统一了标准，为应用产品的开发奠定了基础，国外各公司纷纷予以响应。因此，国内厂家在 PCI 总线的产品开发方面也应迎头赶上，凡是与高速数据传输相关的产品，在技术上都应考虑选用 PCI 总线技术标准。

2. PCI 总线系统产品的开发

该类产品包括：①PCI 总线个人计算机系统；②PCI 总线网络服务器系统；③PCI 工作站系统。

从目前个人计算机市场来看，由于多媒体的日益繁荣，其家用市场越来越大，况且视频播放所需的大数据量传输能力正是 PCI 总线的特长，因此只要是家用计算机上的多媒体产品，就应考虑 PCI 总线的使用问题。

至于 PCI 总线网络服务器，由于它可以支持多总线、多 CPU 芯片，因而，许多网络产品厂家已推出了双奔腾芯片的服务器，使服务器的容错性能和吞吐量大大提高。

PCI 总线在工作站上的应用目前尚未普及，这是因为各厂家工作站产品的结构及操作系统不尽相同，同时有些工作站的专用总线的性能并不逊色于 PCI 总线。所以，在工作站上采用 PCI 总线相对来说就显得比较谨慎。

3. PCI 总线应用产品的开发

PCI 总线的应用产品主要有：①PCI 总线磁盘控制器板卡；②PCI 总线网络板卡；③PCI 总线图像板卡。

PCI 总线的盘控接口 SCSI，由于它具有比 IDE 接口传输率高的特点，同时 PCI 总线的传输率也远比 ISA 和 EISA 总线要高，所以它有十分明显的优势。市面上的 PCI 总线主板产品，有不少已把硬盘的 IDE 接口和 SCSI 接口都集成在母板上，从而减少了盘控卡和扩展槽口之间的一次转换，使盘控接口的可靠性提高、成本降低、结构简化，同时也少占用一个扩展槽口。如目前市场上见到的 B355 PCI SCSI 接口符合 SCSI—2 标准，传输率 110 MB/s，32 位 PCI 总线接口的 B350C IDE 卡，具有较大的数据吞吐量，使硬盘数据存储器的速度大大加快。

关于网卡，制造商们已开发研制出具有 10 MB/s 和 100 MB/s 两种速率的 PCI 总线网络接口卡，可用于 10 MB/s 和 100 MB/s 的

快速以太网。如果用户想把已有的网络升级到新一代局域网，首先应考虑选用 PCI 网卡。Xpoint Technologies 公司的 Switch—on—a Card 可以将服务器的吞吐量提高到 100 MB/s，并且不需要 CPU 介入，从而减轻了 CPU 的负担，使服务器从路由服务中解脱出来。目前，从网络的发展来看，其中一个很突出的矛盾是信息阻塞。在信息高速公路尚未完善的情况下，64 KB/s 还不能称为高速，10 MB/s 只能传输一般文件，而 100 MB/s 在传输视频信号时有动画效应，且图像的连续性不佳，只能借助于压缩工具，以解决网络信道负荷过重、使用效率低下的问题。但是，若选用 PCI 网卡却可以在不改变网络结构的情况下，以最少的投资提高传输带宽和改善网络性能。

在图像处理方面，相应的 PCI 总线产品已相对成熟和完善。为满足多媒体视频播放的需要，已逐步推出了支持 PCI 总线的视频加速产品和图形加速产品。图像压缩和图像加速，是从图像处理的角度来解决图像的大量传输和存储问题，而且数据传输率的大小，将直接影响图像的播放质量。但是 PCI 总线的高速、高效数据传输能力，对于 30 帧/s 的播放速度可以说是“得心应手”。总之，用现有的图像压缩解压技术和图像加速处理技术，再加上 PCI 总线技术便可实现完美的视频播放。各种电影卡和电视卡，在 PCI 总线的高速数据通道上，也可运用自如。

目前，PCI 总线技术图形加速卡有：采用 S3 公司 Trio32 和 Trio64 芯片的 VGA 显示卡；采用凌云公司 5434 64 位 GUI 加速芯片的显示卡。多媒体图形/图像加速卡应首选 DSV 3968，该板卡采用 S3 Vision 968 多媒体加速器，实现了 4M VRAM 图形/图像双加速，支持 MPEG 解码，可以加快图像播放速度，减轻 CPU 负担，在 Pentium 机上，其播放速度可达 30 帧/s，同时支持 800×600、1024×768 及 1600×1200 高分辨率真彩色逐行显示。

前面我们已介绍了有关 PCI 总线的组件、开发工具及产品开

发情况，总的来看，目前市场上各种类型的PCI产品在逐步推出，而且部分已相当成熟。但由于PCI总线是一种新型的总线标准，仍有需要完善的地方。同时，关于该总线在各种环境的应用方面，也有大量的工作可做。可有一点对开发者来讲应特别注意，那就是在任何情况下都必须严格遵守PCI总线的时序要求和相应规范。

附　录

PCI 总线操作规则

1. 一旦复位完成，应保证下列信号在所有时钟的上升沿稳定：LOCK#，IRDY#，TRDY#，REQ #，FRAME #，DEVSEL#，STOP#，GNT#，REQ64 #，ACK64 #，SBO #，SDONE#，PERR#，SERR#（只在下降沿）。

2. 保证地址/数据线在以下各种情况中的相应要求：

a. 在 FRAME#有效后的第一个时钟上，无论地址线 AD[31 :: 00]是否全部有用，它们都必须是稳定的。

b. 在 REQ64#有效后的第一个时钟上，无论地址线 AD[63 :: 32]是否全部有用，它们都必须是稳定的。

c. 在读操作中，当 TRDY#有效时，数据线 AD[31 :: 0]与字节使能无关，必须全部稳定有效；而在写操作中，当 IRDY#有效时，数据线 AD[31 :: 00]与字节使能无关，必须全部稳定有效。在其它任何时间数据线的状态都是不确定的。在读写操作中，一旦相应的TRDY#或IRDY#有效，数据线就不能发生变化直到当前数据期完成为止。

d. 在读/写传输中，当 ACK64#和 TRDY#/IRDY#有效时，数据线 AD[63 :: 32]与字节使能无关应全部稳定有效，而在其它任何时间都是不确定的。

e. 在特殊周期命令中，当 IRDY#有效时，数据线 AD[31 :: 00]在传输期间稳定有效且与字节使能无关。

f. 在读/写传输中，当 TRDY#/IRDY#有效后，不能向 PCI 总线上发异步数据。

3. 命令/字节使能线的状态应满足下述要求：

a. 作为总线命令的 C/BE[3 :: 0]# 和 C/BE[7 :: 4]#，分别在 FRAME# 和 REQ64# 初次建立时保持稳定有效并且含有相应的命令码。

b. 作为字节使能的 C/BE[3 :: 0]# 和 C/BE[7 :: 4]#，在地址期过后的时钟上以及整个数据期的每个时钟周期都是稳定有效的，并且不受等待周期插入的影响。在突发传输期间，主设备可以在每个数据期完成时，所对应的时钟上修改字节使能，但此修改值要在下一个时钟上才能有效。

4. PAR 在 AD[31 :: 00]有效后的一个时钟上稳定有效；PAR64 在 AD[63 :: 32]有效后的一个时钟上稳定有效。

5. IDSEL 只在配置访问时相应的 FRAME# 建立后的第一个时钟上稳定有效，而在其它任何时间都是不确定的。

6. 对于 RST#，IRQA#，IRQB#，IRQC# 和 IRQD# 没有限制或者说是异步的。

7. 当 FRAME# 和 IRDY# 无效而 GNT# 有效时，一个设备可以启动一次访问。

8. FRAME# 信号的初次建立就标志着一次传输的开始。

9. 在所有的 PCI 传输中，FRAME# 和 IRDY# 应符合下列条件：

a. FRAME# 和 IRDY# 定义了总线的忙/闲状态。当其中一个有效时，总线是忙的；两个都无效时，总线处于忙状态。

b. 一旦 FRAME# 被置为无效，在同一传输期间不能重新设置。

c. 除非设置 IRDY# 有效，一般情况下不能设置 FRAME# 无效。

d. 一旦主设备设置了 IRDY#，直到当前数据期结束为止，

主设备不能改变 IRDY＃和 FRAME＃的状态。

10. 当下列条件之一满足时，表明最后一个数据期已经完成：

a. FRAME＃无效而 TRDY＃有效(正常终止方式)。

b. FRAME＃无效而 STOP＃有效(目标终止方式)。

c. FRAME＃无效并且设备选择计时器已经计满(主设备废止方式)。

d. DEVSEL＃无效而 STOP＃有效(目标废止方式)。

11. 当 FRAME＃和 IRDY＃均无效时，表示传输结束。

12. 下列一般规则在所有 PCI 传输中对于 FRAME＃、IRDY＃、TRDY＃、STOP＃都有效：

a. 每当 STOP＃发出时，FRAME＃必须尽快地撤消，但要符合撤消 FRAME＃的规则，即必须发出 IRDY＃。FRAME＃的撤消应尽快在 STOP＃发出之后 2～3 个时钟周期之内实现。目标设备不能假设 STOP＃的发出和 FRAME＃的撤消之间有任何时间关系，而是必须保持 STOP＃信号一直到 FRAME＃撤消时为止。当主设备取样发现 STOP＃有效时，它就必须在有 IRDY＃的周期后面的第一个周期内将 FRAME＃撤消。IRDY＃的发出和 FRAME＃的撤消动作可以作为主设备正常的 IRDY＃行为，并根据主设备何时准备完成一次数据传输而延迟 0 个或更多个周期。然而，如果 TRDY＃无效，主设备便可立即发出 IRDY＃，因为这时不会发生数据传输。

b. STOP＃一旦建立，就必须保持到 FRAME＃撤消为止，接着 STOP＃也必须撤消。

c. 一旦目标设备发出了 TRDY＃或 STOP＃，它就不能改变 DEVSEL＃、TRDY＃和 STOP＃信号，直到当前的数据期完成。

13. 主设备和目标设备之间的数据传送发生于每个 TRDY＃和 IRDY＃同时有效的时钟沿上。

14. 当数据有效时，要求数据源无条件发出 XRDY＃信号(写传输为 IRDY＃，读传输为 TRDY＃)。接收设备也必须发出它的 XRDY＃信号。

15. 如果当前传输被目标终止时，主设备必须撤消它的 REQ＃信号至少两个 PCI 时钟周期，一个是总线进入的第一个空闲周期，另一个在此空闲周期之前或之后。

16. 一个设备通过 DEVSEL＃信号表明它是被访问的目标。

17. DEVSEL＃的发出必须早于或同时于目标使能输出时所对应的时钟边沿。

18. 一旦 DEVSEL＃建立，除非被目标废止，否则在最后一个数据期完成之前，不允许将它撤消。

19. LOCK＃信号具有独占性并且只能由一个设备驱动，当总线释放时它仍可以保留。

20. 在 PCI 总线上，一个支持 LOCK＃的目标设备必须遵守下列规则：

a. 当 LOCK＃在地址期中撤消时，被访问的设备要将自身锁住。

b. 一旦建立了锁，目标将保持锁住状态，直到取样发现 FRAME＃和 LOCK＃一起撤消或者发出目标废止。

c. 保证 LOCK＃信号所有者的独占性，一旦锁已建立，至少有 16 个字节的资源，最多可以锁住整个资源。

21. 在 PCI 总线上，使用 LOCK＃的主设备必须遵循以下规则：

a. 在锁操作期间，一个主设备只能访问一个单一的资源。

b. 一个锁不能跨越设备边界。

c. 16 个对齐的字节是一个主设备在锁操作中执行互斥时可

以计算的最大资源。对 16 字节块内任何字节的互斥访问，将会锁住整个 16 字节的块。

d. 锁操作中的第一个传输必须是读传输。

e. LOCK＃必须在紧跟地址期的时钟上被设置，并保持设置以继续控制。

f. 在数据期结束之前，如果出现再试并且锁还没有建立时，应该释放 LOCK＃。

g. 无论何时，在一存取被主、从设备打断时，必须释放 LOCK＃。

h. 在连续的锁操作中，LOCK＃必须被置成一个最小空闲周期。

22. 仲裁器可以在任何时钟置某一设备的 GNT＃信号无效。

23. GNT＃一旦建立，其撤消应符合以下规则：

a. 如果总线不是处于空闲状态，有可能一个 GNT＃的撤消时刻碰巧是另一个 GNT＃的发出时刻。否则，要求一个 GNT＃的撤消到下一个 GNT＃的发出之间要有一个时钟的延迟，以避免在 AD 线和 PAR 线上出现冲突。

b. 当 FRAME＃无信号时，GNT＃可以在任意时间撤消，以便服务于另一个主设备，或者作为对相应的 REQ＃撤消的响应。如果 GNT＃撤消而 FRAM＃有效，则总线的传输有效并可继续下去。

24. 当仲裁器向一个设备发出了 GNT＃信号并且总线处于空闲状态时，该设备必在 8 个 PCI 时钟周期内将 AD[31 :: 00]、C/BE[3 :: 0]＃和 PAR(推迟一个周期)驱动到有效状态。

25. 奇偶校验的产生应依据下述规则：

a. 不管类型及形式，在所有 PCI 事务中奇偶校验的计算方法不变。

b. AD[31 :: 00]、C/BE[3 :: 0]＃及 PAR 上“1”的个数等于

偶数。

c. AD[63 :: 32]、C/BE[7 :: 4]# 及 PAR 上"1"的个数等于偶数。

d. 奇偶校验的产生不是可选项，它必须由所有 PCI 从属设备完成。

参 考 资 料

[1] 曾凡太，李洪珍. PCI 总线发展概况. 北京：计算机世界，1996(7)：101

[2] 曾凡太，陈美金. PCI 总线产品的开发. 北京：计算机世界，1996(7)：109

[3] 林勇、周晓雁. 微机总线及其发展. 微型机与应用，1995(5)：16～17