

Windows98 平台下 PCI 设备的软件开发

蔡骏, 蒋大明, 戴胜华

(北方交通大学电子信息工程学院, 100044, 北京)

[摘要] 针对应用日益广泛的 PCI 总线, 介绍了其概念、性能、结构及特点. 着重论述了 PCI 设备的软件开发方法, 并结合具体的硬件设备与 windows98 操作系统平台, 给出了两套完整的开发方案, 比较这两套方案, 并给出了部分关键源代码.

[关键词] 局部总线, PCI 总线, Windriver

[中图分类号] TP311.5, [文献标识码] B, [文章编号] 1672- 1292(2003)03- 0039- 04

0 引言

计算机技术的不断发展对微型计算机总线性能提出了更高的要求. 如 CPU 运行速度的不断提高, 高速缓冲存储器(cache) 的广泛采用等, 都要求有高速的总线来传输数据. 传统的总线标准如 ISA、EISA 已经远远不能满足高速数据传输的需要了, 局部总线(Local Bus) 的推出, 是一种完善的解决方案. 在局部总线的发展中, PCI 总线的发展最令人瞩目.

1 PCI 总线介绍

PCI(Peripheral Component Interconnect) 总线是一种高性能的局部总线. 局部总线是指来自处理器的延伸线路, 与处理器同步操作. 因此, PCI 总线不受制于处理器, 它为 CPU 及高速外围设备之间提供了一座桥梁, 可缩短外围设备取得总线控制权所需时间, 提高数据吞吐量; PCI 总线以 33 MHz 的时钟频率操作, 采用 32 位数据总线, 并能扩展到 64 位, 可支持多组外围部件, 数据传送速率高达 132 MB/s, 远远超过标准 ISA 总线 5 MB/s 的速率; PCI 兼容性强, 与 ISA、EISA 总线完

全兼容. 更值得一提的是 PCI 总线的“自动配置”的功能, 即 PCI 设备内有配置暂存器, 当系统启动时, 自动设定与处理器沟通的端口地址及物理地址, 不必手工调整跨接线. PCI 总线的系统结构如图 1.

从图中可以看到, CPU/Cache 通过一个 PCI 桥连接到 PCI 总线上. PCI 桥提供了一个低延迟的访问通路, 将 PCI 设备映射到 CPU 能够直接访问到的存储器空间或 I/O 空间; 也提供了 PCI 设备直接访问主存的高速通路; 该桥将 PCI 总线的操作与 CPU 总线分开, 避免了相互影响. 总之, PCI 桥实现了 PCI 总线的驱动控制; 扩展总线桥的设置, 使得 PCI 总线上能够接出一条标准 I/O 扩展总线, 从而兼容现有的

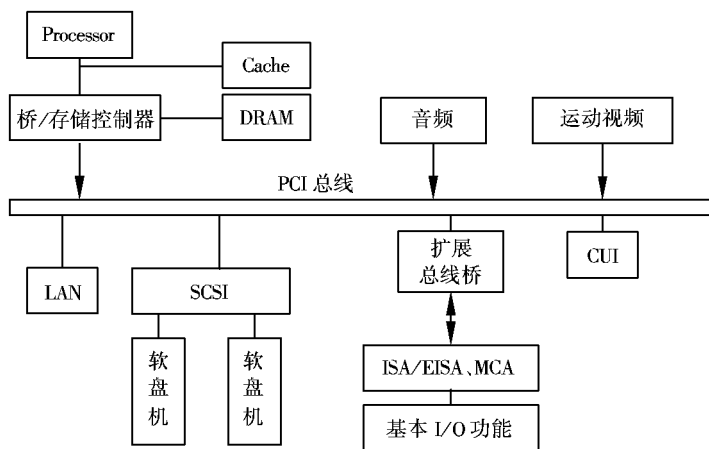


图 1 PCI 系统结构图

收稿日期: 2003- 03- 06

作者简介: 蔡骏, 1979-, 北方交通大学电子信息工程学院硕士研究生. 主要从事微机自动控制的学习和研究

ISA、EISA、MCA 总线. 由于 PCI 总线既迎合了当今的技术要求, 又能满足未来的需要, 因此是计算机界公认的局部总线标准, 得到了广泛的应用.

2 PCI 设备的软件开发

目前, 在 windows 操作系统下开发驱动硬件设备的软件, 使用微软的 DDK(driver develop kit) 工具是标准的方法, 但是工作量大, 难度高; 还可以参照总线规范, 按照已定义的软件接口规范来驱动设备, 但缺陷在于代码可读性差, 不易实现平台间的移植; 值得庆幸的是, 由于第三方软件的蓬勃发展, 现在开发设备驱动程序的难度有所降低, 通过已经提供的、功能完善的接口, 驱动开发者不需要了解硬件的实现细节就能够与硬件通信, 省时省力.

现以一个实际项目中的应用, 讨论了 PCI 设备的软件开发过程, 在项目中使用了两块 ADLINK 公司的 PCI-9114A 多功能数据采集卡. 在每块 PCI-9114A 中都有一个 8254 可编程定时器, 项目中需要利用它产生频率. 8254 定时器有 3 个独立的 16 位计数器, 每个计数器都有控制命令寄存器和 6 种选择信号, 见文献[2]. 将两块采集卡的 8254 级联, 把它们第 0 号计数器 Counter0 作为分频器, 对采集卡内部 2 MHz 时钟进行分频. 接线时将第一块卡的分频输出, 作为第二块卡内部时钟频率输入. 理论上可以实现的频率最高可达 $2\text{ MHz}/4 = 500\text{ KHz}$, 最低约为 $2\text{ M}/65535/65535 \approx 0.00046\text{ Hz}$, 输出范围、精度都足以满足项目的需要.

产生频率信号的关键就是要找到采集卡里 8254 定时器的内部寄存器的物理地址, 设置命令字和计数初值. 在 Windows98 平台下使用了下面两种方案.

2.1 利用 PCI BIOS 的开发方案

由于 PCI 设备的资源是系统自动配置的, 因此操作系统、应用程序和设备驱动程序无法直接访问 PCI 配置寄存器, 但可以通过 PCI BIOS 功能调用, 即通过 X86 系列的寄存器来传递参数, 调用 DOS 功能号得到返回参数的方式解决, 调用返回的值也保存在有关的寄存器中. 有关 PCI BIOS 功能调用的资料请参考 PCI BIOS 规范.

使用 PCI BIOS 提供的 16 位实模式接口, 16 位接口可通过调用中断 1AH 来实现, BIOS 也支持 INT 21H 工业标准的入口点来仿真中断调用. 下面以第一块 9114 为例, 给出部分查找 PCI 设备及配置空间的程序:

```
int main( int argc, char* argv[ ])
{
    DWORD chipPhy, BaseAddr;
    unsigned char BusNo, DevNo, ReturnCode;
    _asm{
        mov ah, 0b1h           //PCI BIOS 定义
        mov al, 02h           // 的入口参数
        mov cx, 09114h         //9114 的设备 ID
        mov dx, 0144ah         // 销售商 ID
        mov si, 0              // 第一块 9114 卡的索引号
        INT 1ah                //BIOS 功能调用
        mov BusNo, bh          //得到总线号
        mov DevNo, bl          // 得到设备号及功能号
    }
    _asm{
        mov ah, 0b1h
```

```

        mov al, 0ah
        mov bh, BusNo
        mov bl, DevNo
        mov di, 14h          // 偏移量 14h
        INT 1ah
        mov chipPhy, ecx      // 得到 LCR 寄存器基址
    }
- asm{
        mov ah, 0b1h
        mov al, 0ah
        mov bh, BusNo
        mov bl, DevNo
        mov di, 18h          // 偏移量 18h
        INT 1ah
        mov BaseAdr, ecx      // 得到 9114 寄存器基址
    }
}

```

在程序中得到了两个基址, 这是由 9114 卡的内部寄存器决定的. 9114 卡有 3 种不同的寄存器, 分别为 PCR(PCI Configuration Registers)、LCR(Local Configuration Registers)、9114 Registers. LCR、9114 Registers 的地址均由 PCR 获得, 它们分别位于 PCR 的偏移量 14H, 18H 处.

得到了 9114 寄存器基址后, 便可以控制 8254 的寄存器了. 在 9114 卡中, Counter0 寄存器的地址为 BaseAdr+ 20H. 要注意在向 Counter0 赋计数初值之前要先设置好 8 位控制命令字 Control Byte, 它位于 BaseAdr+ 26H 处. 因此, 通过利用 PCI BIOS 功能调用查找 PCI 设备并得到物理地址, 再由物理地址寻找设备的有关寄存器并进行读写控制, 可以实现产生频率信号.

2.2 利用 Windriver 的开发方案

利用 PCI BIOS 的缺点是需要熟悉汇编语言, 代码可读性较差; 最不方便之处是由于使用 VB6.0 编程, 无法嵌入汇编程序, 而 VB 的硬件编程能力非常有限, 要在 VB 开发环境下访问 PCI 设备, 必须有其他的软件支持. 目前, KEFTech 公司的主推产品 Windriver 是许多 PCI 设备首选驱动程序开发工具.

2.2.1 Windriver 的结构

如图 2 所示, 为了实现对硬件的访问, 用户应用程序从 Windriver 用户模式库(Windriver.h)中调用 Windriver 函数, Windriver.h 再调用 Windriver Kernel, 它运行于内核模式, 能够直接对硬件操作.

2.2.2 Windriver 开发 PCI 设备的实现

如前所述, Windriver 为 VB 提供了访问硬件的手段. Windriver 为 VB4.0 以上版本提供了一个类模块(Windriver.cls), 里面包含驱动程序接口函数以及相关的数据结构. 利用这个类模块用户就可以编写自己所需的接口来访问 PCI 设备. 下面是访问 PCI9114 卡资源的一段程序, 它能够说明, Windriver 是如何

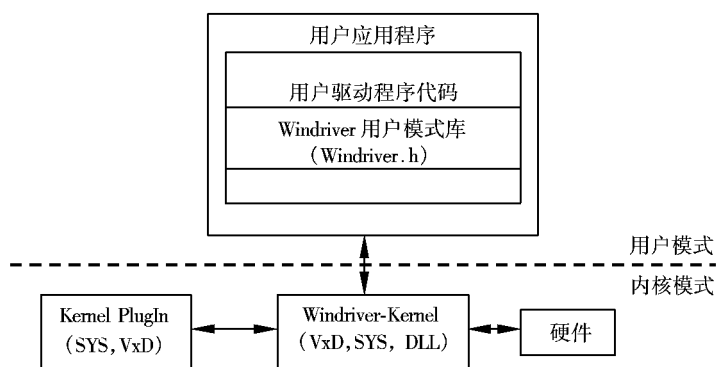


图 2 Windriver 的系统结构

访问 PCI 设备的.

```
.....  
hWD= WD_ Open          ' 得到对 windriver=kernel 操作的句柄  
Call WD_ Version( hWD, VerInfo)      ' 验证 windriver 的版本是否正确  
If WD_ Version( hWD, Verinfo) = INVALID Then  
    Goto Localerror  
End If  
PciScan. searchId. dwVendorId= &H144A      ' 9114 的 vendorId  
PciScan. searchId. dwDeviceId= 37140        ' 9114 的 deviceId  
Call WD_ PcScanCards( hWD, PciScan)        ' 查找 PCI 设备  
PciCard. pciSlot= PciScan. cardSlot( 0)     ' 选中第一块  
WD. PciGetCardInfo hWD, PciCard            ' 获取第一块 9114 的配置信息  
Card1. Card= PciCard. Card  
WD. CardRegister hWD, Card1                ' 注册第一块板卡  
.....  
.....  
WD. CardUnregister hWD, Card1              ' 释放板卡占用的内存  
WD. Close hWD                             ' 关闭句柄  
.....
```

在用 VB 编写驱动程序的过程中, 得到正确的 hWD 很重要. 因为 windriver. cls 类模块中所有的接口函数都需要使用它来访问内核模式的资源; 当驱动程序编写完毕后, 一定要释放内存和关闭 windriver 句柄.

利用 Windriver 开发 PCI 设备, 代码可读性好, 并且与汇编语言相比, 如果不慎写错, 不会对操作系统有致命的破坏性, 该程序已在项目中正常运行.

3 结束语

本文阐述了利用 PCI BIOS 和 Windriver 进行 PCI 设备开发软件的方法. 第一种方法需要了解 PCI BIOS 规范以及 X86 系列的寄存器, 在项目中无法实现代码移植; 第二种方法屏蔽了操作系统平台, 跨平台性强. 在 windows98 平台下, 作者设计的 PCI 设备的软件已能够完成 48 通道数据采集和频率发送功能, 并取得了良好的效果.

[参考文献]

- [1] Windriver5. 20 Manual. Jungo Ltd.
- [2] 戴梅萼, 史嘉权. “微型计算机技术及应用——从 16 位到 32 位”. (第二版) [M]. 北京: 清华大学出版社, 1999.
- [3] PCI BIOS Specification. Version 2. 01.

Development of PCI Devices on Windows98 Platform

Cai Jun, Jiang Daming, Dai Shenghua

(School of Electronics and Information Engineering, Northern Jiaotong University, 100044, Beijing, PRC)

A stract: This paper discusses the conception, performance, architecture and characters of PCI bus. It lays emphasis on the software methods to develop PCI devices. Besides it provides two kinds of developing methods with hardware and windows98 OS platform and shows some central codes, with the two kinds of methods compared.

Key words: local bus, PCI bus(PCI BIOS), Windriver

[责任编辑: 刘健]