基于 VxWorks 的 PCI 总线多功能数据采集卡驱动开发

张 原，邹程帅，张 帅，宋 鹏
（西北工业大学 电子信息学院，陕西 西安 710129）

摘要： VxWorks 是 WindRiver(风河)公司开发的嵌入式实时操作系统(RTOS)，由于它的高实时性，所以广泛地应用于军事、工业控制、通信等领域；分析了 VxWorks 下 PCI 总线多功能数据采集卡的实现方法；以 ADLINK 的 PCI7396 数据采集卡为例，介绍 PCI 总线设备的配置空间，包括它的结构及访问方法，重点介绍 PCI 总线设备在 VxWorks 下驱动程序的开发步骤及编程要点，并对开发过程中的关键部分给予代码说明；在某综合控制系统中，开发的驱动程序运行稳定、可靠。

关键词： VxWorks；PCI；数据采集卡；驱动
中图分类号： TP336 **文献标识码：** A **文章编号：** 1674-6236(2012)12-0039-03

VxWorks-based PCI bus multifunction data acquisition card driven development

ZHANG Yuan, ZOU Cheng-shuai, ZHANG Shuai, SONG Peng
(*School of Electronics and Information, Northwestern Polytechnical University, Xi'an* 710129, *China*)

Abstract: VxWork s is an embedded Real Time Operation System (RTOS) developed by WindRiver company, because of its high real-time,it widely used in military, industrial control, communications and other fields; The multifunction data acquisition card of the PCI bus implementation on VxWorks is analyzed; Taking ADLINK PCI7396 data acquisition card as an example,introducing the configuration space of the PCI bus devices including its structure and access methods , focusing on the PCI bus device driver in the VxWorks development process and programming essentials , the key part of the development process will give the code as an explaining; The driver is developed in an integrated control system . It is stable and reliable.

Key words: VxWorks; PCI; data acquisition card; driver

VxWorks 是一种高性能的嵌入式实时操作系统(RTOS)，它由 WindRiver 公司开发，具有小巧的内核，可根据需要进行裁减；它还获得广泛的硬件支持，像 X86 系列的 CPU，Motorola 68K 系列的 CPU，Motorola/IBM Power PC 等等；它还具有很高的可靠性和实时性，像美国登陆的火星探测器使用的就是 VxWorks 操作系统；它还有其它的很多优点，因此被广泛地使用在通信、军事、航空、控制等高精尖技术以及实时性要求极高的领域中。

PCI 总线由于其即插即用、独立于微处理器、通用性好、具有很高的兼容性等等优良的特点，使得它成为运用最广泛的局部总线标准，而且具有很大的发展潜力。使它成为具有很好发展潜力的局部总线标。多功能数据采集卡在工业控制中有着广泛的运用，它是信号和嵌入式处理器的有效交互的工业控制系统中的重要环节，特别是它的中断功能能提供很有效的实时性。在 VxWorks 下，一个重要的问题就是如何开发出高效率的数据采集卡驱动，只有驱动稳定了，才能使系统高效地运转。文中结合 Adlink 公司的 7396 芯片，探讨在 VxWorks 下基于 PENTIUM CPU 的 PCI 多功能数据采集卡的驱动开发。

1 VxWorks 下的设备驱动程序

1.1 VxWorks 下的 I/O 系统

I/O 系统全称为 (Input/Output) 输入输出系统，在 VxWorks 系统中，I/O 系统向用户屏蔽了硬件层，为用户提供了一个统一的标准接口，使得应用层的用户只要了解 I/O 系统的标准接口使用方法，就可以正确地操作外部设备[1]。I/O 系统为设备提供了 7 个标准的 I/O 接口函数：creat（...）、delete（...）、open（...）、close（...）、read（...）、write（...）、ioctl（...）。驱动程序设计者只要根据实际项目、工程的需要设计完成相应功能的接口函数，然后在使用时 I/O 系统就可以把应用程序的 I/O 请求转发给相应的设计好的设备驱动程序进行处理。在 VxWorks 系统中 I/O 系统是通过维护文件描述符表、设备描述符表和驱动程序列表这 3 张表格来实现对驱动程序的管理的。

1.2 VxWorks 设备驱动程序的分类

在 VxWorks 系统中，输入/输出设备从宏观上分为 3 种类型：字符设备、块设备和网络设备。依据设备的类型，VxWorks 下设备驱动程序的管理也被划分成 3 种模块：字符设备驱动程序模块、块设备驱动程序模块、网络设备驱动程

收稿日期：2012-03-31 **稿件编号：**201203227
作者简介：张 原(1969—)，男，陕西西安人，博士，副教授。研究方向：嵌入式系统，计算机软件，通信系统控制。

序模块[2]。从架构上说,字符设备驱动程序的实现相对来说是最简单的,它向上只是与 I/O 系统接口,所以只需要提供内个接口就可以了。块设备驱动程序相对于字符设备来说相对的复杂,它还要与中间的文件系统相连,为文件系统提供服务,块设备的最大特点就是以“块”为单位进行操作,在读取其中的数据时,必须将所在块的数据完全读出。网络设备则提供了另外的接口,相对于字符设备和块设备来说,处于相对底层,这是由网络协议的复杂性决定的。为了更好地支持网络设备,VxWorks 在网络协议层和网络设备驱动程序之间增加了 MUX 接口[3]。

1.3 驱动程序和 I/O 系统

驱动程序的结构包括 3 个部分:初始化部分,函数功能部分和中断服务程序 ISR。初始化部分主要是初始化硬件,分配硬件所需要的系统资源。函数功能主要是根据实际的需要,完成系统指定的功能。中断服务程序主要是响应外部中断,使系统能够快速地对外部交互作出反应,由于中断服务程序要求尽可能地反应快,所以中断服务程序应该尽可能简单。

VxWorks 中设备驱动程序与 I/ O 系统的关系非常简单。对于块设备来说, 应用程序通过 I/O 系统访问文件系统, 而后由文件系统调用驱动程序访问设备。而其它的非块设备则是 I/O 系统直接调用驱动程序访问设备。图 1 显示了应用程序、I/ O 系统和设备驱动程序之间的关系。

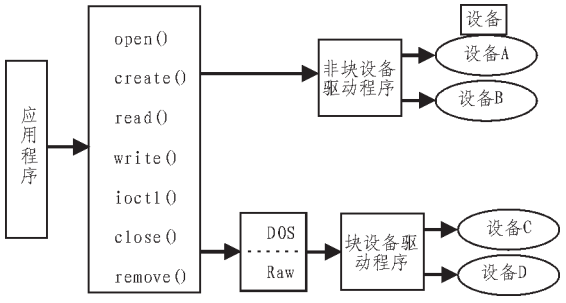

图 1 应用程序、I/O 系统和设备驱动程序之间关系

Fig. 1 Relationship of Application, the I/O systems and device drivers

2 多功能数据采集卡 7396 简介

PCI-7396 是专门为工业应用而设计的 96 位并行数字输入/输出(DIO)卡。PCI-7396 是模拟 4 个 8255 可编程外围 接口(PPI)芯片。每个 PPI 提供 3 个 8 位可同步访问的 DIO 端口。总共有 12 个可独立配置为输入或输出的端口。

PCI-7396 产品具有通过外部触发来锁定数字输入数据的特点,同时提供状态改变(COS)中断,这意味着当 任何数字输入状态改变的时候,中断就会发生。用户可以通过用跳线设置上拉/下拉电阻轻松地以用户自定义 状态(高或低)来设定 PCI-7396 数字 I/O 上电状态。

7396 有 3 种类型的寄存器:PCI 配置寄存器, 它是要实现板卡的功能所要访问和操作的寄存器; 本地配置寄存器,其是它就是操作 9050 所要访问的寄存器; 还有一个是 PCI-6308 寄存器。

3 PCI 配置空间

PCI(Peripheral Component Interconnect),外部设备互联标准,是由 Intel 公司提出的一种局部总线标准。每个 PCI 设备有 3 种物理空间:配置空间、存储器空间和 I/O 空间。配置空间是长度 256 字节的一段连续空间 (16 个 32 位寄存器)其中前 64 个字节为头标,其余 192 字节为设备相关信息。在 64 字节的头标中,前 16 字节的定义是确定的,后 48 字节的具体含义因设备而异。配置空间头标区如图 2 所示。配置空间中的一个重要部分是基地址寄存器(BaseAddresss Register), 它的内容是 PCI 设备的地址空间映射到系统地址空间的起始物理地址。其中,bit0=1 表示 IO 空间映射,bit0=0 表示存储器空间映射。所有 PCI 设备必须实现存储器空间映射。通过向 BAR 写全 1 即可确定所需地址空间的大小[4]。在 VxWorks 下要访问一个 PCI 设备, 只需要知道该设备的厂商号和设备号。

设备识别			供应商代码	00H
状态			命令	04H
分类代码			版本	08H
内含自测试	头标类型	延时计数	Cache大小	0CH
基址寄存器				10H 14H 18H 1CH 20H 24H
CarBus CIS指针				28H
子系统代码		子系统供应代码		2CH
扩展 ROM基地址				30H
保留		能力指针		34H
保留				38H
MAX.LAT	MIN.CNT	中断引脚	中断线	3CH

图 2 PCI 配置头空间

Fig. 2 PCI configuration header space

4 数据采集卡设备驱动的实现

Adlink 公司的 7396 数据采集卡是 PCI 设备,PCI 设备驱动程序属于 VxWorks 体系结构中的 I/O 系统部分,它往上为应用程序提供 API 接口,往下通过 BSP 访问 PCI 设备[5]。基于 VxWorks 的 PCi 设备驱动程序开发流程如图 3 所示。分为 4 个步骤:①创建设备;②根据 PCI 设备的配置参数,对 PCI 设备编写功能函数程序;③编写测试程序进行功能测试;④驱动程序工作正常可靠, 即可发布驱动程序, 将其加载入 VxWorks 操作系统内核,完成驱动程序的开发[6]。

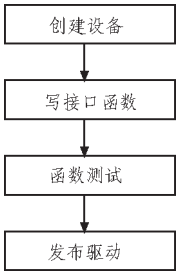

图 3 开发流程

Fig. 3 Development process

在前面的一节中提到过，要访问一个 PCI 设备，首先要知道它的厂商号和设备号，在实际工作中，可以通过 WinDriver 这个在 windows 平台下运行的编写驱动的软件获得，这样比较方便和直观，通过它，得到 7396 的厂商号（Dev7396_VENDOR_ID）为 0x144a，设备号（Dev7396_DEVICE_ID）为 0x7396。知道厂商号和设备号后，就可以利用 VxWorks 提供的函数 pciFindDevice（）获得设备的总线号、设备编号和功能号。然后调用 pciConfigInLong（）获得多功能数据采集卡设备的配置空间、内存空间、I/O 空间的地址；pciConfigInByte 得到设备中断号。7396 设备没有要访问的内存空间，所以没有涉及到内存的映射和操作。主要的实现代码如下所示：

```
/* 获取设备的总线号、设备号、功能号 */
pciFindDevice (Dev7396_VENDOR_ID, Dev7396_DEVICE_ID,0, &Dev7396Config.bus, &Dev7396Config.dev, &Dev7396Config.func)
/* 读取数据寄存器的基地址 */
pciConfigInLong (Dev7396Config.bus, Dev7396Config.dev, Dev7396Config.func, PCI_CFG_BASE_ADDRESS_2, &iobaseCsr);
iobaseCsr &= PCI_IOBASE_MASK;
gDev7396CardInfo.iobase = iobaseCsr;
printf("io address is 0x%x...\n",iobaseCsr);
/* 读取 9050 的基地址 */
pciConfigInLong (Dev7396Config.bus, Dev7396Config.dev, Dev7396Config.func, PCI_CFG_BASE_ADDRESS_1, &localAddr);
localAddr &= PCI_IOBASE_MASK;
gDev7396CardInfo.ioLoclBase = localAddr;
printf("local address is 0x%x...\n",localAddr);
```

在读取设备的基地址时，要特别注意结合设备相关的说明书进行，在前面的章节中提到过，7396 设备包含 3 个寄存器空间（详见 2 多功能数据采集卡 7396 简介），所以读出来的基地址空间有可能是 3 个里面的一个。

```
/* 使能 I/O 空间 */
pciConfigOutWord (Dev7396Config.bus, Dev7396Config.dev, Dev7396Config.func, PCI_CFG_COMMAND, PCI_CMD_IO_ENABLE | PCI_CMD_MASTER_ENABLE );
/* 读取中断 */
pciConfigInByte (Dev7396Config.bus, Dev7396Config.dev, Dev7396Config.func, PCI_CFG_DEV_INT_LINE, &irq);
gDev7396CardInfo.intNum = irq;
/* 连接中断向量 */
pciIntConnect ((VOIDFUNCPTR*)INUM_TO_IVEC(gDev7396CardInfo.intNum + 0x20), (VOIDFUNCPTR)intRoutine, 0);
```

由于使用的是 PENTIUM 系列的 CPU 来进行板卡驱动的开发，所以在边接中断向量的时候，中断号要加上 0x20。

```
/* 使能中断 */
sysIntEnablePIC(irq);
/* 设置 9050 的中断控制寄存器 */
sysOutLong (gDev7396CardInfo.ioLoclBase + ICR_9050, 0x00000049);
```

特别要注意操作 9050 的控制寄存器的偏移为 0x4c 的地方的操作，它是要实现中断必须要进行的操作，没有操作正确的话，中断是不能够正确运行的，而且 9050 的这个寄存器的各个位和 9052 还是不一样的，虽然都是 PLX 公司的产品。

读取完上面的 I/O 基地址和相应的中断后，就可以操作 I/O 空间了，这要根据相应板卡说明书上的基地址偏移来操作，而且还要注意设置端口是输入端口还是输出端口。中断服务程序在 VxWorks 下也有严格的要求，要注意。

5　结束语

文中结合具体的 Adlink 公司的 7396 数据采集卡，介绍了在 VxWorks 下编写 PCI 设备驱动的相关方法以及要注意的事项，并介绍了 VxWorks 的 I/O 系统和驱动相关的理论知识，它们是开发驱动的基础。根据上面方法开发出的 7396 驱动已经在某综合控制系统中稳定地运行。

参考文献：

[1] 徐惠民. 基于VxWorks的嵌入式系统及实验[M]. 北京:邮电大学出版社,2007.

[2] 周启平,张杨. VxWorks下设备驱动程序及BSP开发指南[M]. 北京:中国电力出版社,2004.

[3] 张杨,于银涛. VxWorks内核、设备驱动与BSP开发详解[M]. 北京:人民邮电出版社,2011.

[4] 张豫榕,董磊. VxWorks的PCI配置方法和应用实例[J]. 电子与电脑,2005(1):81-85.
ZHANG Yu-rong,DONG Lei. VxWorks PCI configuration and application examples[J]. Compotech China,2005(1):81-85.

[5] 欧峰,吴成富,段晓军,等. 基于VxWorks的多串口卡驱动程序设计[J]. 测控技术,2008,27(10):71-74.
OU Feng,WU Cheng-fu,DUAN Xiao-jun,et al. Driver design for the multi-serial-port card based on VxWorks[J]. Measurement and Control Technology,2008,27(10):71-74.

[6] 李寒冰,刘庆想,李相强,等. 基于VxWorks的PCI设备驱动程序设计[J]. 强激光与粒子束,2011,23(11):3091-3094.
LI Han-bing,LIU Qing-xiang,LI Xiang-qiang,el al. Design of peripheral component interconnect device driver based on VxWorks[J]. High Power Laser and Particle Beams,2011,23(11):3091-3094.