

## 3G 通讯模块设备驱动设计与实现

陈佐<sup>1</sup>, 冉再<sup>1</sup>, 涂员员<sup>1</sup>, 陈汉<sup>2</sup>

(1. 湖南大学 计算机与通信学院, 湖南 长沙 410082;

2. 长沙行成信息科技有限公司, 湖南 长沙 410005)

**摘要:**随着国家 3G 牌照的发放, 支持 3G 移动通讯的设备开发已经成为通讯产业和嵌入式系统发展的热点。嵌入式 Linux 因其源代码开放、可定制等特点而在嵌入式系统开发中得到广泛应用, 而基于 3G 通讯模块的设备驱动是开发 3G 终端设备和服务的基础。文中讨论了嵌入式 Linux 环境下的设备文件管理、USB 系统结构, 详细分析了 USB 驱动骨架, 并结合 arm 系列 S3C2440 处理器和支持 TD-SCDMA 的通讯模块 LC6311, 实现嵌入式 Linux 系统下的 3G 通讯模块设备驱动开发, 并给出了测试验证实例。

**关键词:**3G; 嵌入式 Linux; LC6311; USB

**中图分类号:**TP302

**文献标识码:**A

**文章编号:**1673-629X(2011)05-0214-05

## Design and Implementation of 3rd-Generation Communication Module Driver

CHEN Zuo<sup>1</sup>, RAN Zai<sup>1</sup>, TU Yuan-yuan<sup>1</sup>, CHEN Han<sup>2</sup>

(1. School of Computer Communication, Hunan University, Changsha 410082, China;

2. Changsha Xing Cheng Information Technology Co., Ltd. Changsha 410005, China)

**Abstract:** Research and development of electronic device which supporting the 3rd-generation communication technology, has become the focus of the communication industry and embedded systems, since 3G licenses were released. As an operating system which is open source code and could be customized, Linux has widely used in embedded system. The device driver for the 3G modules, which was applied under Linux environment, was the most important for developing 3G terminal facilities and services. Discussed the management of device file and the USB system structure under embedded Linux environment. Subsequently, it analyzed the USB driver skeleton in great detail. At last, it gave the implementation of developing device drivers for 3G modules based on the S3C2440 processor and LC6311 module. At the end of this paper, some samples were used to show how to test the performance of the system.

**Key words:** 3rd-generation; embedded Linux; LC6311; universal serial bus

## 0 引言

2009 年 1 月 7 日, 工信部正式为中国移动、中国电信和中国联通发放 3 张第三代移动通信牌照, 标志着我国正式进入 3G 时代。信息产业将形成一条包括 3G 网络建设、终端设备制造、运营信息服务等在内的通信产业链, 对扩大内需、刺激社会经济产生积极影响。支持具有我国自主知识产权的 TD-SCDMA 标准的移动通讯的设备开发已经成为通讯产业和嵌入式系统发展的热点。目前, 嵌入式软件平台多种多样, 比较典型的软件平台有: Linux、wince、Palm OS 等。Linux 以其良好的实时性、高效实用性、可靠可裁剪性以及开源等特

点, 已经广泛应用于工业控制、消费电子、通信网络等领域。设备驱动程序与软硬件环境息息相关, 设备驱动的开发尤为重要, 基于 3G 通讯模块的设备驱动开发是 3G 终端设备和服务的基础。

文中介绍了在软件平台 utlinux2.6.24 和硬件平台 S3C2440A 下 LC6311 模块 USB 驱动的开发。首先, 介绍了 Linux 下设备驱动的管理方式, 设备驱动注册后会在 /dev/ 下生成对应的设备文件, 用户空间通过访问设备文件实现对设备硬件的具体操作, 由内核根据设备 ID 或设备标号调用已注册的同类型的设备驱动。因此, 设备驱动是内核与设备的接口, 它为上层用户程序提供访问硬件的入口函数。其次, 介绍了 USB (Universal Serial Bus) 系统的主体框架, 它由 USB 主机、USB 主机控制器、USB 设备及其连接构成。USB 驱动的开发就是对 USB 主机驱动的开发, USB 设备端驱动通常固化在设备硬件的存储器里。再次, 分析了内核

收稿日期: 2010-10-12; 修回日期: 2011-01-24

基金项目: 湖南省科技计划重点项目 (2010GK2002)

作者简介: 陈佐 (1979-), 男, 博士, 讲师, 主要研究领域为高性能计算、数据挖掘。

提供的 USB 驱动骨架,从中了解 USB 驱动开发的基本思想,以便更好掌握嵌入式 Linux 下 USB 驱动开发流程,对 LC6311 驱动的主要结构体和函数进行了剖析,比较了驱动加载的两种方式,最后对驱动进行了简单的测试<sup>[1]</sup>。

## 1 开发环境简介

### 1.1 硬件环境

开发硬件环境主要包括应用处理器基于 ARM 架构的 S3C2440A 和移动 TD-SCDMA 射频通讯模块 LC6311。

S3C2440A 开发板采用 ARM920T 内核,包括 16kB 指令 Cache(I-Cache)/16kB 数据 Cache(D-Cache),130 个通用 I/O,24 个外部中断源,拥有外部存储控制器,集成 LCD 专用 DMA 的 LCD 控制器,4 路拥有外部请求引脚的 DMA 控制器,三路 UART 口,两路 SPI,一个 USB1.1 主机接口,可接 usb hub,一通道 USB1.1 设备接口,8 路 10 位 ADC 和触摸屏接口,摄像头接口,以太网控制器等<sup>[2]</sup>。

LC6311 模块支持 TD-SCDMA(HSDPA)& GGE 双模制式,有手动切换和自动切换两种方式。工作频带:TD-SCDMA(2010~2025MHz),GSM(GSM850、E-GSM、DCS1800、PCS1900)。遵循 3GPP TS27.005、3GPP TS27.007、ITU-T V.25ter。多通道复用协议支持 3GPP TS 27.010,集成 TCP/IP 协议,支持飞行模式功能。TD-SCDMA 下支持多种上下行速率,典型的下行 384kbps/上行 384kbps 等。主要应用接口:UART 接口,USB 接口,USIM 卡接口,两路模拟音频输入接口,三路模拟音频输出接口,USIM 卡接口,控制信号接口等等<sup>[3]</sup>。

S3C2440 和 LC6311 的硬件连接如图 1 所示,USB 系统主机在 S3C2440 开发板上,主机通过与主机控制器相连的主 USB 口与设备端 USB 口相连,LC6311 通过 60pin 与底板基座连接,天线连接在 LC6311 上,用于 3G 或 GSM 信号的接收与发送。

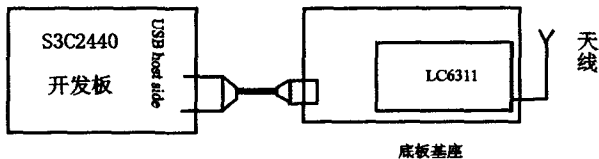


图 1 LC6311 与 S3C2440 连接图

### 1.2 软件环境

utuLinux2.6.24 内核版本比较稳定,改善了 CFS、x86-64、PPC、UML、ARM、MIPS 等架构,添加了 Tickless 支持,新增无限驱动程序及配置界面,引入了 Anti-fragmentation 补丁,使 MMC 层获得 SPI/SDIO 支持,添

加了 USB authorization、PID 及网络命名空间,针对 TCP 流量的 LRO 支持等特性,此外该版本还添加了包括显卡、SATA/IDE、网络、声卡、MTD、USB、V4L/DVB、Hwmon、I2C、蓝牙等设备的新驱动程序。通过配置内核,可以使其支持更多的功能和应用<sup>[4]</sup>。

## 2 LC6311 模块 USB 驱动设计原理

### 2.1 Linux 设备驱动管理

系统调用是内核和应用程序之间的接口,而设备驱动程序是 Linux 内核与设备硬件的软件接口。由于操作系统对硬件资源的保护性,使得用户应用程序不能直接访问硬件设备,这些硬件对于用户来说是不可见的。所以当应用进程访问硬件资源时,是通过设备驱动程序实现的,而驱动程序的调用是由内核完成的,驱动程序属于内核的一部分。Linux2.6.24 内核通过设备文件系统的方式来管理设备。各种设备都是以设备文件的形式存放在/dev 目录下,并且系统为各种设备分配了合理的主设备号和次设备号,主设备号用于标示同类型设备,次设备号用于标示同类型的不用设备。用户程序通过访问设备文件,调用 Linux 系统函数(API,应用程序编程接口),然后由内核决定调用对应设备的驱动程序以达到应用进程访问硬件的目的。

### 2.2 Linux 设备文件类型

Linux 下的设备文件有三类:字符设备、块设备、网络设备<sup>[4]</sup>。

字符设备是指在 I/O 传输过程中以字符(并不一定意味着以字节为单位,因为有的编码规则规定字符占两字节)为单位进行传输的设备,如键盘、鼠标、LED、打印机等。通常字符设备都是顺序写入和读取的,当对字符设备发出读写请求时,实际的硬件 I/O 会紧接着执行操作,从而提高字符型 I/O 操作的速度。

块设备主要是面向大量数据的传输或慢速设备而设计的,可以节省 CPU 等待时间。块设备存放在设备文件系统(/dev)中。当对块设备进行访问时,首先要开辟一段内存缓冲区,以方便数据的暂存操作。

网络设备在嵌入式 Linux 系统中比较特殊,不像字符设备或块设备那样能通过对应的设备文件去访问网络设备,内核也不是通过系统调用访问它。嵌入式 Linux 的网络系统主要是基于 BSD UNIX 的 socket 机制。

### 2.3 USB 设备驱动

USB 通用串行总线作为计算机的标准设备接口,一个 USB 系统由主机和设备组成。USB 设备驱动有主机端设备驱动程序、主机控制器驱动和设备端驱动程序之分。主机端设备驱动程序就是人们常说的驱动程序。主机控制器驱动程序主要是针对主机控制器。

设备端驱动程序属于 Firmware(设备固件程序)的一部分,提供设备端信息与主机的通信接口。USB 主机端驱动程序框架如图 2 所示。

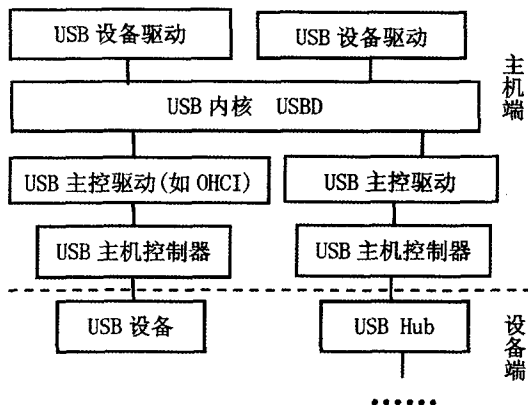


图 2 USB 主机端驱动框架

由图 2 可知,在主机端,处于最底层的是 USB 主机控制器硬件,向下与 USB 设备连接,向上为 USB 主机控制器驱动,USB 主控驱动之上是 USB 内核层,再往上就是 USB 设备驱动层。因此,要实现的 USB 设备驱动,主要控制 USB 设备如何与主机通信。USB 内核层主要完成 USB 驱动的管理和协议处理,其通过定义一些数据结构、功能函数、宏等,向上为 USB 设备驱动提供编程接口,向下为 USB 主控驱动提供编程接口,通过全局变量维护整个系统的 USB 设备信息,完成设备的热插拔控制,数据总线传输控制等。

#### 2.4 LC6311 模块的 USB 驱动开发原理

USB 驱动是内核和 USB 主控制器之间的接口,USB 核心为 USB 驱动提供访问控制 USB 硬件的接口,所以 USB 驱动的编写不需要考虑底层 USB 硬件的类型差别。虽然 USB 系统比较复杂,但是内核已经提供了一个 USB 核心子系统来处理大部分复杂问题。

USB 设备对外提供的通信进出口有三种形式:配置、接口、端点。USB 通信以端点为最小传输口,端点只能往一个方向传输数据,从主机到设备或设备到主机。USB 设备具有多个配置,每个配置一般由一个或多个接口组成,而每个接口又是由一个或多个端点构成。在 USB 核心中有一个数据缓存,当用户进程发送数据给设备或设备传送数据给用户进程时,用来缓存传输的数据。在 USB 设备端由固化程序进行数据的接收与发送工作。数据两端通过 USB 总线相连,与总线直接连接的是两端的 USB 控制芯片,进行实际的物理层数据的传输<sup>[5]</sup>。

USB 驱动程序将对应的设备注册到 USB 核心子系统中,当设备接入该 USB 系统时,由主机根据设备厂商标示和设备标示来识别该设备。USB 核心支持的设备种类罗列在一个列表中,当设备进行热插拔操作

时,脚本通过它来决定装载哪个驱动。

在 USB 驱动开发过程中,采用结构体的形式描述对 USB 硬件各种操作、状态、数据等信息。典型的 URB 结构体用来装载需要传输的数据,file-operations 结构体用来定义系统调用与具体功能函数的链接。

LC6311 模块作为 USB 设备,用户进程可以通过访问 USB 设备文件的方式与 LC6311 模块通信。而在实际的操作中,是通过串口的形式访问 LC6311 的,这是因为对串口比 USB 操作要简单。

一般 USB 驱动主要完成以下功能:对设备初始化,把数据从内核传到硬件和从硬件读取数据给内核,读取应用程序传给设备文件的数据和回送应用程序请求的数据,检查并处理硬件设备出现的错误<sup>[6]</sup>。

### 3 LC6311 模块 USB 驱动设计

USB 驱动骨架程序为编写 USB 驱动提供了良好的框架,根据具体的 USB 外设编写相应的代码,最终完成驱动的设计。

#### 3.1 usb-skeleton 驱动骨架分析

Linux 内核源码中的 usb-skeleton.c 即是 USB 驱动骨架。在这里介绍几个非常关键的数据结构<sup>[7]</sup>。

USB 设备结构体 usb\_skel,简要介绍已在结构体中:

```

struct usb_skel {
    struct usb_device *udev, /* 保存设备指针 */
    struct usb_interface *interface, /* 设备接口 */
    unsigned char *bulk_in_buffer, /* 接收数据缓冲区 */
    size_t bulk_in_size, /* 接收数据缓冲区大小 */
};

USB 驱动结构体 skel_driver:
static struct usb_driver skel_driver = {
    .name = "skeleton",
    .probe = skel_probe,
    .disconnect = skel_disconnect,
    .suspend = skel_suspend,
    .id_table = skel_table,
};
  
```

上个结构体中:name 用来告诉内核模块的名字;probe 和 disconnect 是函数指针,当设备和 id\_table 中的变量信息相同时,函数被调用;id\_table 用来告诉内核该模块支持的设备;Suspend 为暂停函数。

```

static const struct file_operations skel_fops = {
    .read = skel_read,
    .write = skel_write,
    .open = skel_open,
    .release = skel_release,
};
  
```

skel\_fops 结构体里面提供系统函数对应的接口函

数指针,内核通过它实现设备的打开关闭、读写、控制等操作。

```
static struct usb_class_driver skel_class = {
    .name = "skel%d", /* 设备名 */
    .fops = &skel_fops, /* 文件操作结构体指针 */
    .minor_base = USB_SKEL_MINOR_BASE, /* 开始次设备号 */
};
```

上为 USB 设备类驱动结构体,USB 设备类比较典型的有语音设备类、通信设备类、USB 打印机设备类、HID 设备类(HUMAN DEVICE CLASS)和存储设备类等。

```
驱动程序的设备列表 skel_table:
static struct usb_device_id skel_table[] = {
    {USB_DEVICE(USB_SKEL_VENDOR_ID, USB_SKEL_PRODUCT_ID)},
    { /* 终止输入 */ }
};
```

为了使系统支持热插拔,还需如下一行代码:  
MODULE\_DEVICE\_TABLE(usb, skel\_table);

3.2 LC6311 模块 USB 驱动设计

LC6311 模块的 USB 驱动编写主要是实现 file\_operations 结构中的对应系统调用的接口函数。这里对比骨架中的几个重要的结构体来看 LC6311 的 USB 驱动,另外简要介绍几个重要的函数。

表 1 为应用进程访问 LC6311 时的接口调用关系表,应用进程通过系统调用实现对设备的访问,file\_operations 结构体实现系统调用与设备驱动中功能函数的连接<sup>[8]</sup>。

下面是 USB 驱动结构体的定义:

```
static struct usb_driver usb_serial_driver = {
    .name = "usbserial",
    .probe = usb_serial_probe,
    .disconnect = usb_serial_disconnect,
    .suspend = usb_serial_suspend,
    .resume = usb_serial_resume,
    .no_dynamic_id = 1,
};
```

};  
USB 为设备驱动程序维护一个 usb\_driver 数据结构,当总线上有设备连接时,USB 中的 HUBD 通过该结构查找相关设备驱动,断开时,同样通过该结构断开设备的链接。

由上结构体可知,该驱动模块的名字是 usbserial。对内核进行正确的配置之后编译,能生成驱动模块 usbserial.ko。

驱动中还用到以下几个重要的数据结构<sup>[9]</sup>:  
struct tty\_struct \*tty, 此结构类型的变量被 tty 核心用来保存当前特定 tty 端口状态。结构体中的成员 struct tty\_driver driver,就是当前控制这个 tty 设备的 tty\_driver 结构。成员 void \*driver\_data,指针 tty\_driver 用来存储 tty 驱动本地的数据。

驱动通过系统调用 usb\_serial\_bus\_register() 和 usb\_serial\_bus\_deregister() 函数实现 USB 总线驱动的注册与注销,通过系统调用 tty\_register\_driver() 和 tty\_unregister\_driver() 函数实现设备驱动的注册与注销。

usb\_serial\_init() 主要完成 usbserial 设备的注册,当 LC6311 设备连接上后,在超级终端上会显示 tty-USB0-ttyUSB5 共 6 个设备文件。

usb\_serial\_exit() 注销掉 usb\_serial\_init() 函数中注册的设备驱动。

usb\_serial\_probe() 函数负责识别设备、内存的分配、创建设备请求 urb 等。

usb\_serial\_register() 函数完成 USB 总线驱动的注册。

usb\_serial\_deregister() 函数完成 USB 总线驱动的注销。

当 LC6311 连接到 USB 总线上时,处于内核中的 HUBD(HUB driver)通过中断数据传输获得设备连接信息,由 USB 内核对设备进行配置工作。USB 为 LC6311 分配在 USB 层和 HCD 层所需资源。接着,USB 调用 LC6311 设备驱动中提供的 usb\_serial\_probe 函数来查询适合该设备的驱动程序。

表 1 应用进程访问设备的接口调用

应用进程通过系统调用	标准系统调用	file-operations 结构体	系统函数对应驱动中的函数	函数功能说明
	open()		serial_open()	打开设备
	close()		serial_close()	关闭设备
	read_proc()		serial_read_proc()	/proc 读函数
	write()		serial_write()	返回缓冲空闲空间
	ioctl()		serial_ioctl()	写设备
	write_room()		serial_write_room()	读写之外的控制操作
	set_termios()		serial_set_termios()	当设备的 termios 设置被改变时, set_termios() 函数将被 tty 核心调用
	throttle()		serial_throttle()	用来帮助控制 TTY 核心的输入缓存
	unthrottle()		serial_unthrottle()	当 TTY 核心的输入缓冲空时被调用
	break_ctl()		serial_break()	线路中断控制函数,当 TTY 驱动打开或变比线路时被调用
	chars_in_buffer()		serial_chars_in_buffer()	指示缓冲中有多少字符
	tiocmget()		serial_tiocmget()	获取当前特定 TTY 设备的线路设置
	tiocmset()		serial_tiocmset()	设置当前特定 TTY 设备的线路设置

当 LC6311 断开连接时, HUBD 获得设备断开信息, 由 USB 内核释放资源, 并调用设备驱动提供的 `usb_serial_disconnect` 函数通知设备驱动程序该设备已断开。

图 3 表述了应用进程访问 USB 设备进行读写时的数据流向。当写 USB 设备时, 进程通过文件系统 (POSIX) 接口访问设备类驱动和 USBD, 设备类驱动中的 `usb_serial_probe` 函数将进程发送的数据创建成 urb 请求包, 并分配给 USB 设备的一个特定的端点, 再将 urb 包通过 USBD (USB driver) 提供的接口传递给 USBD, USBD 再将 urb 包传给 USB 主控驱动 HCD, 最后 HCD 通过主控硬件将 urb 包放在 USB 总线上, 由 USB 设备硬件接收, 当 urb 传输完成, USB 主控驱动通知 USB 设备驱动。USB 设备端的固件程序通过端点完成 urb 数据的接收。

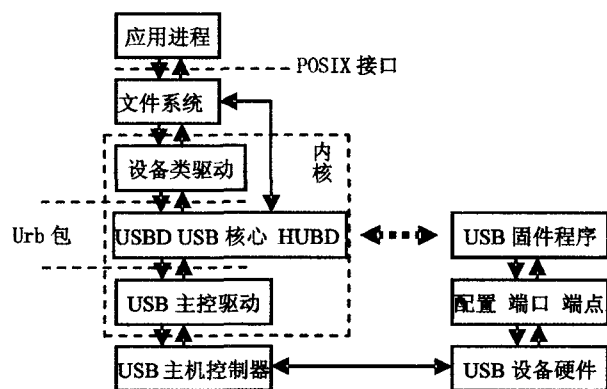


图 3 USB 系统读写数据流程图

urb 数据体包含一次数据传输所必需的信息, 主要有数据缓冲区、数据管道、数据缓冲区长度、USB 设备信息、回调函数、不同类型数据传输所需参数、控制参数等成员变量。由于 urb 在设备驱动、USB、HCD 之间传输, 所以 urb 包含在这三层驱动中传输所必需的信息。

### 3.3 内核配置

为使驱动能正确运行, 还需要内核的支持, 内核配置如下所示:

进入 Device Drivers ---->

进入 USB support ---->

进入 USB Serial Converter support ---->

选中 USB Generic Serial Driver //USB 到 serial 的驱动支持

选中 Support for Host-side USB //对 USB 主机系统的支持

选中 OHCI HCD support //选中主机控制器驱动

选中 USB Mass Storage support //对 USB 海量存储设备的支持

选中 USB Monitor

//对 USB 设备

监控的支持

若把 USB 设备当高速设备来处理, 还需进入 Block devices ---> 然后选上 Low Performance USB Block driver。若把 USB 设备当 SCSI 设备来处理, 则去掉 Low Performance USB Block driver 选项, 进入 SCSI device support --->, 选中 SCSI disk support。其他可能涉及到的功能已经在内核默认配置中已选<sup>[10]</sup>。

### 3.4 驱动测试

插入基于 USB 接口的 LC6311 设备, 进入 Linux 命令行窗口, 输入命令 `insmod usbserial.ko vendor = 0x1AB7 product = 0x6000` 来加载 LC6311 模块驱动<sup>[11]</sup>。

编写 3G 测试程序, 再交叉编译生成二进制文件, 用于驱动测试。嵌入式 Linux 2.6.24 通过 NFS 挂载的方式启动, 编写好相关配置文件后, 再运行测试程序即可<sup>[12]</sup>。

测试中的返回值 `DSQU: 28` 表示当前信号强度为 28 (信号强度范围 0-91), 在测试程序中是通过写指令 `AT+DSQ = 1` 来检测当前信号强度的, 同时说明了驱动能正常工作。

## 4 结束语

中文主要介绍了基于嵌入式 Linux 下的设备驱动的管理方式, USB 系统的构成及其驱动类别, 通过分析 USB 骨架程序, 了解 USB 驱动开发的基本流程。针对 LC6311 模块 USB 驱动, 介绍了几个重要的数据结构和系统调用。最后, 对驱动进行了测试。

目前, 3G 网络在国内应用还不够成熟, 对 3G 设备生产厂商来说存在巨大的市场。LC6311 模块作为一种支持 3G 和 GSM 通信的 modem, 以其体积、速率、功耗、可靠等方面的优势, 必将赢得广大用户的青睐。而 LC6311 模块的驱动随着终端软硬件的不同而有所区别。所以, 文中对 LC6311 模块 USB 驱动开发基本流程的分析, 对不同类型的 3G 模块驱动的开发具有良好的指导意义。

### 参考文献:

- [1] 孙天泽, 袁文菊. 嵌入式设计及 Linux 驱动开发指南-基于 ARM9 处理器 [M]. 北京: 电子工业出版社, 2008: 282-339.
- [2] S3C2440A 32-BIT RISC MICROPROCESSOR USER'S MANUAL PRELIMINARY [EB/OL]. 2004-06. <http://www.elec-fans.com>.
- [3] LC6311 硬件接口手册 [EB/OL]. 2008-11. <http://download.csdn.net/source/2092925>.
- [4] 李俊. 嵌入式 Linux 设备驱动开发详解 [M]. 北京: 人民

(下转第 222 页)

的数据库操作。

## 5 结束语

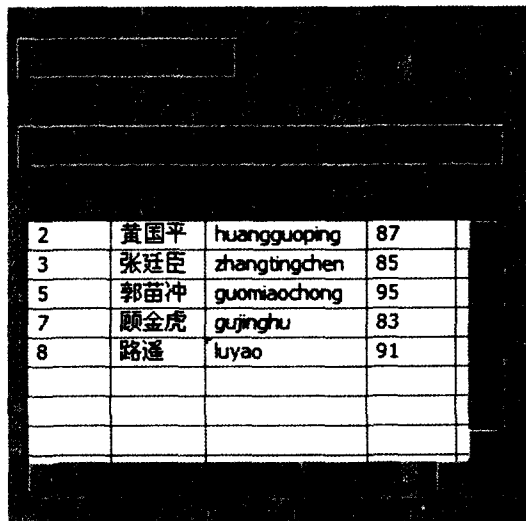
文中设计了通过免费的第三方开发工具 LabSQL, 采用 ODBC 连接 Access 数据库, 利用 Microsoft ADO 对

象和 SQL 数据库查询语言来完成对数据库的访问, 从而实现了 LabVIEW 与数据库之间的数据通信。实践证明, 此方法简洁易行, 操作方便, 是一种理想的连接和访问数据库方法。不足的是, LabSQL 不能对远程数据库进行访问, 如果需要远程访问, 可采用其他方法。



2	黄国平	huangguoping	87
3	张廷臣	zhangtingchen	85
4	付瑞	furui	48
5	郭苗冲	guomiaochong	95
6	邓阳	dengyang	73
7	顾金虎	gujinghu	83
8	路遥	luyao	91

a 显示 studentrecord 表中所有数据



2	黄国平	huangguoping	87
3	张廷臣	zhangtingchen	85
5	郭苗冲	guomiaochong	95
7	顾金虎	gujinghu	83
8	路遥	luyao	91

b 显示 studentrecord 表中 recore>80 的数据

图 4 前面板显示的查询结果

### 参考文献:

- [1] 王太成, 蔡 勇. 利用虚拟机技术完成复杂网络实验[J]. 计算机技术与发展, 2009, 19(4): 246-249.
- [2] 苏福根, 金红莉, 朱新宇. 网络虚拟实验平台在高等教育网络化、信息化中的应用研究[J]. 中国教育信息化, 2009(9): 69-70.
- [3] 安小东. 虚拟实验技术的应用研究[D]. 天津: 天津理工大学, 2007.
- [4] 毛 松. 远程虚拟电路实验仿真技术[D]. 杭州: 浙江大学, 2005.
- [5] 周 荣, 丁金华, 李明颖, 等. 在 LabVIEW 中利用 ADO 技术动态创建数据库[J]. 大连轻工业学院学报, 2005, 24(3): 207-210.
- [6] 沈 媛, 李宏坤, 王奉涛. 基于 LabSQL 访问数据库技术的优化实现与应用[J]. 仪器仪表用户, 2008, 15(1): 64-65.
- [7] 彭汉国, 艾瑞东, 王明方, 等. 基于 LabVIEW 的数据库访问技术[J]. 舰船电子工程, 2009, 29(8): 151-154.
- [8] 杨乐平. LabVIEW 高级程序设计[M]. 北京: 北京清华大学出版社, 2003.
- [9] 赵桂明, 赵质良. 在 LabVIEW 中灵活访问数据库[J]. 电子测试, 2008(9): 49-52.
- [10] 秘晓元, 张彦斌, 薛德庆, 等. LabVIEW 中利用 LabSQL 对数据库访问技术的探讨[J]. 自动化与仪器仪表, 2004(6): 54-56.
- [11] Bishop R H. LabVIEW 8, Student Edition[M]. Beijing: Publishing House of Electronics Industry, 2007.
- [12] Chang Chenhung, Eshe P, Sven K. On the assembly of experimental setups in virtual laboratory environments [J]. ASME International Mechanical Engineering Congress and Exposition, Proceedings, 2008, 7: 421-428.
- [13] Chen M, Chen S H. Virtual and Remote Control Laboratory Development[J]. IEEE Control System Magazine, 2005, 5: 78-80.

(上接第 218 页)

邮电出版社, 2008: 311-332.

- [5] Universal Serial Bus Specification Revision 2.0[S]. 2000-04.
- [6] 白复东. 嵌入式 Linux 驱动程序的开发[J]. 信息技术, 2009(9): 185-189.
- [7] 阮赐兴, 赵彦敏. Linux 下 MPC8272 的 USB 主控制器驱动设计与实现[J]. 计算机技术与发展, 2009, 19(2): 12-15.
- [8] Fliegl D. Programming Gide for Linux USB Device Drivers

[EB/OL]. 2000-12. <http://usb.as.tum.edu>.

- [9] Corbet J, Rubini A, Greg Kroah-Hartman. Linux Device Drivers[M]. [s. l.]: [s. n.], 2005.
- [10] 唐六华, 王 瑛. 嵌入式 Linux 下 USB 主机设备驱动开发[J]. 计算机技术与发展, 2009, 19(9): 181-185.
- [11] 陆智慧. USB 多端口设备 LINUX 环境使用指南[M]. 联芯科技有限公司, 2009: 1-3.
- [12] 薛园园. USB 应用开发技术大全[M]. 北京: 人民邮电出版社, 2007: 165-181, 634-645.