嵌入式 Linux字符设备驱动程序设计

舒克毅,胡荣强
(武汉理工大学 自动化学院,湖北 武汉 430070)

摘要: 阐述 Linux下设备驱动及其相关的概念,概括了字符驱动程序的编写过程,对 Linux平台下字符设备驱动程序的工作机理进行了分析。结合 S3C2410开发板中 LED驱动开发对字符设备的驱动开发流程进行深论。

关键词: Linux;字符设备;驱动程序
中图分类号: TP39 文献标识码: B 文章编号: 1006-2394(2010)02-0004-02

Design of Char Drive Procedures in the Embedded Linux System

SHU Ke-yi, HU Rong-qiang
(School of Automation, Wuhan University of Technology, Wuhan 430070, China)

Abstract: the paper describes the conceptions of the drivers on Linux. By taking LED drivers on the S3C2410 embedded development board as the example, the design method of char drivers on Linux is discussed.

Key words: Linux; character device; device driver

0 引言

由于 Linux开放源代码,易于移植,资源丰富,可剪裁等优点,使它在嵌入式领域越来越流行。Linux下设备驱动程序的编写要点是为相应的设备编写初始化函数,对设备进行必要的初始化,包括硬件初始化、向内核注册驱动程序等,可见设备驱动程序实质上是一组完成不同任务的函数的集合。当应用程序需要对设备进行操作时,可以访问该设备对应的文件节点,内核将调用该设备的相关处理函数,通过这些函数所提供的功能可以使得从设备接收输入和将输出送到设备就象读写文件一样。

1 Linux系统下设备驱动及其相关的概念

在 Linux系统中,为屏蔽硬件细节,给应用程序提供一个统一的编程接口,引入了设备驱动这种手段,即无论底层硬件有何不同,都通过设备驱动程序的方式,将具体设备抽象为设备文件(网络设备例外,它抽象为接口)。设备文件不使用文件系统中的任何数据空间,它仅仅作为设备驱动访问的入口点,应用程序利用该入口点就可以像操作普通文件一样来操作设备。Linux系统中设备的基本类型有字符设备、块设备和网络设备 3种。相对来说,字符设备对应的驱动程序最为简单,90%以上的设备驱动都是针对字符设备的。可以用“ls/dev-l”命令查看设备文件的类型,以字母“c”开头的表示字符设备,以字母“b”开头的表示块设备。比如 pc机上的串口属于字符设备,硬盘属于块设备。

Linux系统为每一个设备分配了一个主设备号和次设备号,主设备号标识设备对应驱动程序,次设备号标识具体设备的实例。例如一块开发板上有 2个串口终端(/dev/ttyO, /dev/ttyO1),它们的主设备号都是 4,次设备号分别为 0和 1。每一类设备使用的主设备号是独一无二的,系统增加一个驱动程序就要赋予它一个主设备号,这一赋值过程在驱动程序的初始化过程中进行。

2 设备驱动程序的组成

设备驱动在加载时首先需要调用入口函数 init_module(),该函数完成设备驱动的初始化工作,比如寄存器置位、结构体赋值等一系列工作,其中最重要的一个工作就是向内核注册该设备,字符设备调用函数 register_chrdev()完成注册。注册成功后,该设备获得了系统分配或向系统申请的主设备号、自定义的次设备号,并建立起与设备文件的关联。设备驱动在卸载时需要回收相应的资源,将设备的响应寄存器值复位并从系统中注销该设备。系统调用部分则是对设备的操作过程,比如 open, read, write, ioct等操作。设备驱

收稿日期: 2009-09
作者简介: 舒克毅(1984-),男,硕士研究生,研究方向为计算机控制与信息系统集成。

动程序可以分成以下 3个主要部分：

（1）自动配置和初始化子程序。负责检测所需驱动的硬件设备是否存在以及是否能正常工作，这部分驱动程序仅在初始化时被调用一次。

（2）服务 I/O就是请求子程序，是驱动程序的上半部分，这部分是系统调用的结果。

（3）中断服务程序又称驱动程序的下半部分，设备在 I/O请求结束或其他状态改变时产生中断。因为设备驱动程序一般支持同一类型的若干个设备，所以调用中断服务子程序时都带有一个或多个参数以唯一标识请求服务的设备。

3 字符设备驱动程序中重要的数据结构和函数

应用程序使用统一的接口函数调用硬件驱动程序，这组接口称为系统调用，对于每个系统调用，驱动程序中都有一个与之对应的函数。对于字符设备驱动程序，这些函数集合在一个 file_operations类型的数据结构中，它定义了常见文件 I/O函数的入口。 file_operations在 Linux内核的 include/linux/fs.h文件中定义。

编写字符设备驱动程序就是为具体硬件的 file_operations结构编写各个函数，大多数的驱动程序只是利用了其中的一部分（对于驱动程序中不提供的功能，把相应位置的值设为 NULL），对于字符设备来说，要提供的主要入口有：open()、release()、read()、write()、lseek()、ioctl()等。本文中用到的主要有 open()与 ioctl()。

open()函数，对设备特殊文件进行 open()系统调用时，将调用驱动程序的 open()函数：int(* open)(struct inode*, struct file*);

ioctl()函数，该函数是特殊的控制函数，可以通过它向设备传递控制信息或从设备取得状态信息，函数原型为：int(* ioctl)(struct inode*, struct file*, unsigned int, unsigned long);

4 驱动程序的注册和卸载

驱动程序有一个初始化函数，在安装驱动程序时会调用它。在初始化函数中会将驱动程序的 file_operations与主设备号一起向内核进行注册。对字符设备使用如下函数进行注册：

int register_chrdev(unsigned int major, const char* name, struct file_operations* fops);

其中，major是为设备驱动程序向系统申请的主设备号，如果为 0则系统动态地分配 1个主设备号，name是设备名。 fops是 file_operations对各个调用入口点的说明。此函数返回 0表示成功；返回 -1是表示出错；返回 -EINVAL表示申请的主设备号非法（一般来说是主设备号大于系统所允许的最大设备号）；返回 -EBUSY表示所申请的主设备号正在被其他设备驱动程序使用。如果是动态分配主设备号成功，此函数将返回所分配的主设备号。如果 register_chrdev操作成功，设备名就会出现在 /Proc/devices文件里。

模块在调用 rmmod函数时被卸载，此时的入口点是 cleanup_module函数或宏 module_exit并在其中完成对设备的注销。类似的，字符设备的卸载函数定义为：

int unregister_chrdev(unsigned int major, const char* name);

5 LED设备驱动程序的设计

5.1 LED电路设计

S3C2410开发板的 LED电路接线如图 1所示。其中 LED1、LED2、LED3及 LED4分别接 S3C2410芯片的 I/O口 GPB5、GPB6、GPB7及 GPB8，这样就可以通过读写 GPB I/O口来控制 LED的状态。4个 LED为共阳极接法，要点亮 LED，令引脚输出 0，要熄灭 LED，令引脚输出 1。

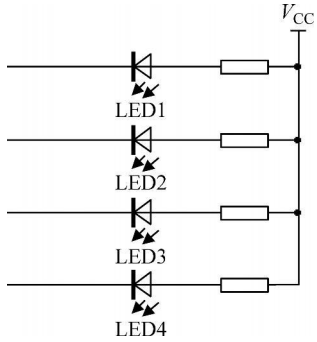

图 1　LED电路设计

5.2 LED驱动程序设计

字符设备是 Linux系统中最简单和常见的设备，应用程序可以调用与存取文件相同的系统来操作它。设备通过设备号来标识，宏定义 LED设备名称，主设备号，代码如下：

```
#define DEVICE_NAME "leds"
#define LED_MAJOR 231
```

file operations类型 S3C2410_leds_fops结构是驱动中最重要的数据结构，编写字符设备驱动的主要工作是填充其中的各个成员。比如本驱动用到的 open，ioctl成员被设为 S3C2410_leds_open，S3C2410_leds_ioctl函数，前者用来初始化 LED所用的 GPIO引脚，后者用来根据用户转入的参数设置 GPIO的输出电平。

（下转第 8页）

WLY1.twb和虚拟资源分配表 WLY1.vri环境文件中的两个工程: WLY1和 WLY2,为黑体者“WLY2”,是当前正在调试的工程。

3.3 系统调试

图 4为某型高度表的收发机各章节测试数据,测试结果显示各章测试均通过(被测为标准正常仪器)。

测试结果窗口

测试项目	结论	结果值	单..	检查标准
第一章 系统启动时间	通过	24.092150	秒	下限= 15.0000 上限= 45.0000
第二章 X6-18针工作基准电压	通过	-49.955000	伏	下限= -50.2500 上限= -49.7500
第二章 X6-9针工作基准电压	通过	25.051000	伏	下限= 24.8000 上限= 25.2000
第二章 X6-16针工作基准电压	通过	15.104000	伏	下限= 14.2000 上限= 15.8000
第二章 X6-24针工作基准电压	通过	-15.052000	伏	下限= -15.8000 上限= -14.2000
第三章 可靠性电压测量	通过	24.232000	伏	下限= 22.0000 上限= 27.0000
第四章 X6-3可靠性信号电压	通过	-0.005000	伏	下限= -2.0000 上限= 2.0000
第四章 X6-5外距电压	通过	-46.492000	伏	下限= -47.2000 上限= -46.0000
第四章 X6-6专距电压	通过	23.236000	伏	下限= 20.0000 上限= 40.0000
第五章 模拟0米高度时,外距电压值	通过	-0.768900	伏	下限= -0.8800 上限= -0.7200
第五章 模拟418米高度时,外距电压值	通过	-0.792400	伏	下限= -0.8800 上限= -0.7200
第六章 搜索灵敏度	通过	76.000000	DB	下限= 74.0000 上限= 82.0000
第六章 跟踪灵敏度	通过	74.000000	DB	下限= 72.0000 上限= 80.0000

图 4 某收发机各章节测试结果

4 小结

通常用该高度表的专用测试仪器,由测试人员手工操作时耗时 1 h左右,现应用综合自动测试平台及相应的适配器,对某型高度表的指示器及收发机分别进行了测试,测试仅需 10 min左右,测试时间大大缩短,并可对指示器及收发机分别进行测试,提高故障诊断率。通过对 10台高度表的测试,结果表明,该系统满足各种指标和测试需求,各种功能都已实现。

参考文献:

[1] 王远达,卢永吉. ATE通用平台的研究[J]. 航空兵器, 2007(10): 33-36.

[2] 刘浩,朱小平. ATLAS语言在自动测试设备 ATE中的应用实践. 计算机测量与控制, 2005, 13(2): 118-119.

[3] 王伟斌,秦红磊. 无线电高度表自动测试系统的实现[J]. 电子测量与仪器学报, 2007, 21(4): 72-76.

(郁菁编发)

(上接第 5页)

```
static struct file_operations s3c2410_leds_fops={
    owner= THIS_MODULE,
    open= s3c2410_leds_open,
    ioctl= s3c2410_leds_ioctl,
};
```

驱动程序作为内核的一部分,因此需要给其添加模块初始化函数。该函数用来完成对所控设备的初始化工作,调用 register_chrdev函数向内核注册驱动程序:将主设备号 LED_MAJOR与 file_operations结构 s3c2410_led_fops联系起来。以后应用程序操作主设备号为 LED_MAJOR的设备文件时, open, ioctl, s3c2410_led_fops中的相应成员函数就会被调用。

```
static int init s3c24xx_leds_init(void)
{
  int ret;
  ret= register_chrdev(LED_MAJOR, DEVICE_NAME, &s3c24xx_leds_
  fops);
  if(ret< 0){
    printk(DEVICE_NAME "can't register major number\n");
    return ret;
  }
    printk(DEVICE_NAME "initialized\n");
    return 0;
}
```

与模块初始化函数对应的就是模块卸载函数,需要调用 unregister_chrdev()函数:

```
    static void exit s3c24xx_leds_exit(void)
{
     unregister_chrdev(LED_MAJOR, DEVICE_NAME);
}
```

在内核启动时,使用 module_init宏指定驱动程序初始化函数, module_exit宏指定驱动程序卸载函数。

```
    module_init(s3c24xx_leds_init);
    module_exit(s3c24xx_leds_exit);
```

6 结束语

本文结合 S3C2410开发板 LED驱动程序的开发,详细讨论了嵌入式 Linux系统中字符设备驱动程序的设计方法和关键技术,对类似的其他字符设备驱动程序开发过程可以起到一定的启发作用。本文的程序经过 S3C2410开发板测试后得到预期结果。

参考文献:

[1] JONATHNA CORBET ALESSSANDRO RUBIN & JOATHAN CORBET. LINUX设备驱动程序[M]. 3版. 北京:中国电力出版社, 2006.

[2] 韦东山. 嵌入式 Linux应用开发[M]. 北京:人民邮电出版社, 2008.

(丁云编发)