

实验 21 字符驱动

21.1 本章导读

本实验给大家介绍一个完整的字符驱动，字符驱动的框架结构必须要掌握。

21.1.1 工具

21.1.1.1 硬件工具

- 1) iTOP4412 开发板
- 2) U 盘或者 TF 卡
- 3) PC 机
- 4) 串口

21.1.1.2 软件工具

- 1) 虚拟机 Vmware
- 2) Ubuntu12.04.2
- 3) 超级终端（串口助手）
- 4) 源码文件夹“char_driver”

21.1.2 预备课程

实验 20 生成字符类设备节点

21.1.3 视频资源

本节配套视频为“视频 21_字符驱动”

21.2 学习目标

本章需要学习以下内容：

完善字符设备中 file_operations 剩余的函数变量

编写简单的应用

21.3 实验操作

因为在前面杂项设备中已经介绍了这几个函数，这里不再重复。

将驱动视频教程 20 中的“create_cnode.c”改为“char_driver.c”。

如下图所示，添加打开关闭等操作的函数。

```
static int chardevnode_open(struct inode *inode, struct file *file){
    printk(KERN_EMERG "chardevnode_open is success!\n");
    return 0;
}
/*关闭操作*/
static int chardevnode_release(struct inode *inode, struct file *file){
    printk(KERN_EMERG "chardevnode_release is success!\n");
    return 0;
}
/*IO操作*/
static long chardevnode_ioctl(struct file *file, unsigned int cmd, unsigned long arg){
    printk(KERN_EMERG "chardevnode_ioctl is success! cmd is %d,arg is %d \n",cmd,arg);
    return 0;
}

ssize_t chardevnode_read(struct file *file, char __user *buf, size_t count, loff_t *f_ops){
    return 0;
}

ssize_t chardevnode_write(struct file *file, const char __user *buf, size_t count, loff_t *f_ops){
    return 0;
}

loff_t chardevnode_llseek(struct file *file, loff_t offset, int ence){
    return 0;
}
```

将结构体 file_operations my_fops 补全，如下图所示。

```
struct file_operations my_fops = {  
    .owner = THIS_MODULE,  
    .open = chardevnode_open,  
    .release = chardevnode_release,  
    .unlocked_ioctl = chardevnode_ioctl,  
    .read = chardevnode_read,  
    .write = chardevnode_write,  
    .llseek = chardevnode_llseek,  
};
```

然后修改一下 Makefile 编译文件，如下图所示。

```
#!/bin/bash  
#通知编译器我们要编译模块的哪些源码  
#这里是编译itop4412_hello.c这个文件编译成中间文件mini_linux_module.o  
obj-m += char_driver.o  
  
#源码目录变量，这里用户需要根据实际情况选择路径  
#作者是将Linux的源码拷贝到目录/home/topeet/android4.0下并解压的  
KDIR := /home/topeet/android4.0/iTop4412_Kernel_3.0  
  
#当前目录变量  
PWD ?= $(shell pwd)  
  
#make命名默认寻找第一个目标  
#make -C就是指调用执行的路径  
#$(KDIR) Linux源码目录，作者这里指的是/home/topeet/android4.0/iTop4412_Kernel_3.0  
#$(PWD) 当前目录变量  
#modules要执行的操作  
all:  
    make -C $(KDIR) M=$(PWD) modules  
  
#make clean执行的操作是删除后缀为o的文件  
clean:  
    rm -rf *.mod.c *.o *.order *.ko *.mod.o *.symvers
```

如下图所示，写一个简单的应用。

```
#include <stdio.h>

#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/ioctl.h>

main() {
    int fd;
    char *hello_node0 = "/dev/chardevnode0";
    char *hello_node1 = "/dev/chardevnode1";
    /*O_RDONLY只读打开,O_NDELAY非阻塞方式*/
    if((fd = open(hello_node0,O_RDONLY|O_NDELAY))<0) {
        printf("APP open %s failed!\n",hello_node0);
    }
    else{
        printf("APP open %s success!\n",hello_node0);
    }

    close(fd);

    if((fd = open(hello_node1,O_RDONLY|O_NDELAY))<0) {
        printf("APP open %s failed!\n",hello_node1);
    }
    else{
        printf("APP open %s success!\n",hello_node1);
    }

    close(fd);
}
```

修改完成之后，在 Ubuntu 系统下使用命令 “mkdir char_driver” 新建文件夹 “char_driver”，然后将修改好的驱动文件 “char_driver.c”、Makefile 文件拷贝到文件夹 “char_driver” 中，如下图所示。

```
root@ubuntu: /home/topeet/char_driver
root@ubuntu:/home/topeet# mkdir char_driver
root@ubuntu:/home/topeet# cd char_driver/
root@ubuntu:/home/topeet/char_driver# ls
char_driver.c  Makefile
root@ubuntu:/home/topeet/char_driver#
```

使用编译命令 “make” 编译驱动，如下图所示。

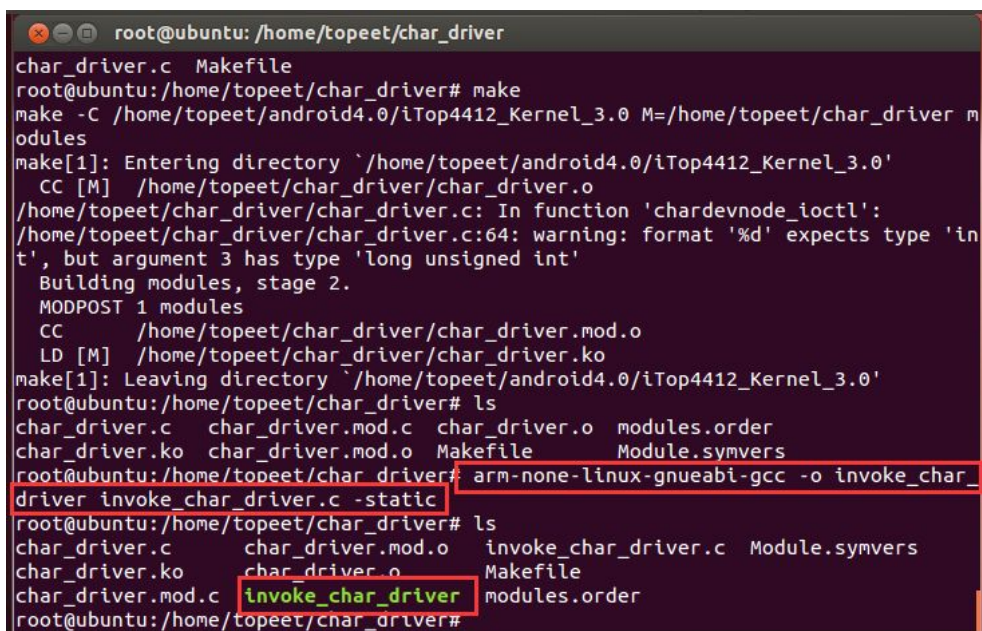
```
root@ubuntu: /home/topeet/char_driver
root@ubuntu:/home/topeet# mkdir char_driver
root@ubuntu:/home/topeet# cd char_driver/
root@ubuntu:/home/topeet/char_driver# ls
char_driver.c  Makefile
root@ubuntu:/home/topeet/char_driver# make
make -C /home/topeet/android4.0/iTop4412_Kernel_3.0 M=/home/topeet/char_driver modules
make[1]: Entering directory `/home/topeet/android4.0/iTop4412_Kernel_3.0'
  CC [M] /home/topeet/char_driver/char_driver.o
/home/topeet/char_driver/char_driver.c: In function 'chardevnode_ioctl':
/home/topeet/char_driver/char_driver.c:64: warning: format '%d' expects type 'int', but argument 3 has type 'long unsigned int'
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/topeet/char_driver/char_driver.mod.o
  LD [M] /home/topeet/char_driver/char_driver.ko
make[1]: Leaving directory `/home/topeet/android4.0/iTop4412_Kernel_3.0'
root@ubuntu:/home/topeet/char_driver# ls
char_driver.c  char_driver.mod.c  char_driver.o  modules.order
char_driver.ko  char_driver.mod.o  Makefile      Module.symvers
root@ubuntu:/home/topeet/char_driver#
```

将修改好的应用文件 “invoke_char_driver.c” 拷贝到文件夹 “char_driver” 中，使用命

令

“arm-none-linux-gnueabi-gcc -o invoke_char_driver invoke_char_driver.c -static”

编译应用，如下图所示。



```
root@ubuntu: /home/topeet/char_driver
char_driver.c Makefile
root@ubuntu:/home/topeet/char_driver# make
make -C /home/topeet/android4.0/iTop4412_Kernel_3.0 M=/home/topeet/char_driver modules
make[1]: Entering directory `/home/topeet/android4.0/iTop4412_Kernel_3.0'
  CC [M] /home/topeet/char_driver/char_driver.o
/home/topeet/char_driver/char_driver.c: In function 'chardevnode_ioctl':
/home/topeet/char_driver/char_driver.c:64: warning: format '%d' expects type 'int', but argument 3 has type 'long unsigned int'
  Building modules, stage 2.
  MODPOST 1 modules
  CC /home/topeet/char_driver/char_driver.mod.o
  LD [M] /home/topeet/char_driver/char_driver.ko
make[1]: Leaving directory `/home/topeet/android4.0/iTop4412_Kernel_3.0'
root@ubuntu:/home/topeet/char_driver# ls
char_driver.c  char_driver.mod.c  char_driver.o  modules.order
char_driver.ko  char_driver.mod.o  Makefile      Module.symvers
root@ubuntu:/home/topeet/char_driver# arm-none-linux-gnueabi-gcc -o invoke_char_driver invoke_char_driver.c -static
root@ubuntu:/home/topeet/char_driver# ls
char_driver.c  char_driver.mod.o  invoke_char_driver.c  Module.symvers
char_driver.ko  char_driver.o      Makefile
char_driver.mod.c  invoke_char_driver  modules.order
root@ubuntu:/home/topeet/char_driver#
```

将生成的驱动模块 “char_driver.ko” 以及应用 “invoke_char_driver” 拷贝到 U 盘。

启动开发板，将 U 盘插入开发板，使用命令 “mount /dev/sda1 /mnt/udisk/” 加载 U 盘，如下图所示。

```
[root@iTOP-4412]#
[root@iTOP-4412]#
[root@iTOP-4412]#
[root@iTOP-4412]#
[root@iTOP-4412]#
[root@iTOP-4412]# mount /dev/sda1 /mnt/udisk/
```

使用命令 “insmod /mnt/udisk/char_driver.ko” 加载驱动模块。

```
COM1
[root@iTOP-4412]# mount /dev/sda1 /mnt/udisk/
[root@iTOP-4412]# insmod /mnt/udisk/char_driver.ko
[ 220.579074] numdev_major is 0!
[ 220.580681] numdev_minor is 0!
[ 220.583686] adev_region req 249 !
[ 220.588512] cdev_add 0 is success!
[ 220.591897] cdev_add 1 is success!
[ 220.595303] scdev_init!
[root@iTOP-4412]#
```

使用命令 “./mnt/udisk/invoke_char_driver” 运行应用，调用驱动模块生成的设备节点，如下图所示，可以看到两个设备节点都可以正常打开，说明驱动底层的 open 函数可以正常使用。

```
[root@iTOP-4412]# ./mnt/udisk/invoke_char_driver
[ 324.627849] chardevnode_open is success!
APP open /dev/cha[ 324.635780] chardevnode_release is success!
[ 324.639936] chardevnode_open is success!
[ 324.643929] chardevnode_release is success!
rdevnode0 success!
APP open /dev/chardevnode1 success!
[root@iTOP-4412]#
```