苏州大学实验报告

院、系	计算机学院	年级专业	20计科	姓名	柯骅	学号	2027405033
课程名称	操作系统课程实践					成绩	
指导教师	李培峰	同组实验者	无		实验日期	2023.05.25	

实 验 名 称 实验10 设备管理

一、 实验目的

1. 了解 Linux 字符设备管理机制。
2. 学习字符设备的基本管理方法。
3. 学会编写简单的字符设备驱动程序的方法。
4. 了解 Linux 块设备管理机制。
5. 学习块设备的基本管理方法。
6. 学会编写一个简单的块设备驱动程序。

二、 实验内容

1. （编写字符设备驱动程序）

（1） 编写字符设备驱动程序，要求能对字符设备执行打开、读、写、I/O 控制和关闭这 5 项基本操作。

（2） 编写一个应用程序，测试添加的字符设备和块设备驱动程序的正确性。

2. （编写块设备驱动程序）

编写一个简单的块设备驱动程序，实现一套内存中的虚拟磁盘驱动器，并通过实际操作验证块设备驱动是否可以正常工作。

三、 实验步骤

1. （编写字符设备驱动程序）

本实验涉及的操作需要管理员权限，因此我们需要切换到 root 权限或使用 sudo 命令。

具体的操作步骤如下。

（1）编写设备驱动源程序，即编写内核模块文件 chardev.c 和 Makefile 文件。

（2）使用 make 命令编译驱动模块。

（3）使用 insmod 命令安装驱动模块。

（4）创建字符设备文件，方法是使用 mknod 命令，语法格式如下。

```
# mknod /dev/文件名 c 主设备号 次设备号
```

然后使用如下命令查看所创建的字符设备文件。

```
# ls /dev
```

（5）编写测试程序 test.c，访问创建的字符设备文件，并使用 gcc 编译这个字符设备文件，然后运行。

（6）使用 rmmod 卸载模块。

（7）使用 rm 命令删除所创建的字符设备文件。

chardev.c 文件、Makefile 文件、测试文件 test.c 示例详见附件 test10/chardev/chardev.c、test10/chardev/Makefile、test10/chardev/test.c

教务处制

2. （编写块设备驱动程序）

本实验涉及的操作需要管理员权限，因此需要切换到 root 权限或使用 sudo 命令。具体的操作步骤如下。

（1）编写设备驱动源程序，即编写内核模块文件 simp_blkdev.c 和 Makefile 文件。
（2）使用 make 命令编译驱动模块。
（3）使用 insmod 命令安装驱动模块。
（4）使用 lsblk 命令列出当前的块设备信息。
（5）格式化设备 simp_blkdev。
（6）创建挂载点并挂载块设备。
（7）查看模块使用情况，会发现模块已被调用。
（8）对块设备驱动进行调用测试。
（9）取消挂载，查看模块调用结果。
（10）使用 rmmod 命令卸载模块。

simp_blkdev.c 文件、Makefile 文件详见附件 test10/simp_blkdev/simp_blkdev.c、test10/simp_blkdev/Makefile

四、 实验结果

1. （编写字符设备驱动程序）

(1) 使用如下命令编译字符设备驱动模块

```
# make
```

运行结果如下：

```
root@ubuntu:/home/kh/test/test10/chardev# make
make -C /lib/modules/5.4.0-148-generic/build M=/home/kh/test/test10/chardev modules
make[1]: 进入目录“/usr/src/linux-headers-5.4.0-148-generic”
  Building modules, stage 2.
  MODPOST 1 modules
make[1]: 离开目录“/usr/src/linux-headers-5.4.0-148-generic”
```

编译成功

(2) 使用 insmod 命令安装编译好的字符驱动模块，使用 lsmod | grep chardev 命令可以查看该模块是否装载成功，命令如下：

```
# insmod chardev.ko
# lsmod | grep chardev
```

运行结果如下：

```
root@ubuntu:/home/kh/test/test10/chardev# insmod chardev.ko
root@ubuntu:/home/kh/test/test10/chardev# lsmod | grep chardev
chardev                16384  0
```

成功查找到名称为 chardev 的模块，表示该模块装载成功

(3) 使用 dmesg 命令查看系统分配的主设备号，命令如下：

```
# dmesg
```

结果如下：

```
[ 1692.866215] <1> I was assigned major number 240
[ 1692.866216] <1> the drive, create a dev file
[ 1692.866217] <1> mknod /dev/hello c 240 0.
[ 1692.866218] <1> I was assigned major number 240
[ 1692.866218] <1> the device file
[ 1692.866218] <1> Remove the file device and module when done
```

图中的 240 即主设备号

(4) 根据输出的主设备号，利用 mknod 命令创建设备，主设备号为 240，执行命令如下：

```
# mknod /dev/hello c 240 0
```

运行结果如下：

```
root@ubuntu:/home/kh/test/test10/chardev# mknod /dev/hello c 240 0
root@ubuntu:/home/kh/test/test10/chardev#
```

(5) 编译并运行测试程序 test.c，命令如下：

```
# gcc test.c -o test
# ./test
# ./test
```

运行结果如下：

```
root@ubuntu:/home/kh/test/test10/chardev# gcc test.c -o test
root@ubuntu:/home/kh/test/test10/chardev# ./test
I already told you 0 times Hello world

root@ubuntu:/home/kh/test/test10/chardev# ./test
I already told you 1 times Hello world

root@ubuntu:/home/kh/test/test10/chardev# ./test
I already told you 2 times Hello world

root@ubuntu:/home/kh/test/test10/chardev# ./test
I already told you 3 times Hello world
```

至此，字符设备工作正常，实验成功。

(6) 当不再需要该字符设备时，卸载模块和设备，命令如下：

```
# rm /dev/hello //删除设备
# ls -l /dev | gerp hello //查找设备
# rmmod chardev //卸载模块
# lsmod | grep chardev //查找模块
```

运行结果如下：

```
root@ubuntu:/home/kh/test/test10/chardev# rm /dev/hello
root@ubuntu:/home/kh/test/test10/chardev# ls -l /dev | grep hello
root@ubuntu:/home/kh/test/test10/chardev# rmmod chardev
root@ubuntu:/home/kh/test/test10/chardev# lsmod | grep chardev
root@ubuntu:/home/kh/test/test10/chardev#
```

在卸载删除后并没有查找到相应模块或设备，表明模块和设备卸载成功。

2. （编写块设备驱动程序）

(1) 使用 make 命令编译块设备驱动模块，命令如下：

```
# make
```

结果如下：

```
kh@ubuntu:~/test/test10/simp_blkdev$ make
make -C /lib/modules/4.16.10/build M=/home/kh/test/test10/simp_blkdev modules
make[1]: 进入目录“/usr/src/linux-4.16.10”
Makefile:976: "Cannot use CONFIG_STACK_VALIDATION=y, please install libelf-dev, libelf-devel or elfutils-libelf-devel"
  CC [M]  /home/kh/test/test10/simp_blkdev/simp_blkdev.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/kh/test/test10/simp_blkdev/simp_blkdev.mod.o
  LD [M]  /home/kh/test/test10/simp_blkdev/simp_blkdev.ko
make[1]: 离开目录“/usr/src/linux-4.16.10”
```

目录中出现 simp_blkdev.ko，编译成功

(2) 挂载块设备驱动模块 simp_blkdev.ko，并使用 lsmod | grep simp_bikdev 命令查看是否挂载成功，命令如下：

```
# insmod simp_blkdev.ko  //装载模块
# lsmod | grep simp_blkdev  //查找模块
```

结果如下：

```
root@ubuntu:/home/kh/test/test10/simp_blkdev# insmod simp_blkdev.ko
root@ubuntu:/home/kh/test/test10/simp_blkdev# lsmod | grep simp_blkdev
simp_blkdev         52445184  0
```

教务处制

发现查找到相应模块，表明装载成功，且使用者为 0。

(3) 使用 lsblk 命令列出当前的块设备信息，命令如下：

```
# lsblk
```

结果如下：

```
loop25         7:25   0 302.2M  1 loop /snap/code/128
loop26         7:26   0  63.3M  1 loop /snap/core20/1879
loop27         7:27   0 302.2M  1 loop /snap/code/129
loop28         7:28   0  55.7M  1 loop /snap/core18/2745
sda            8:0    0    55G  0 disk
└─sda1         8:1    0    55G  0 part /
sr0           11:0    1  1024M  0 rom
simp_blkdev   72:0    0    50M  0 disk
root@ubuntu:/home/kh/test/test10/simp_blkdev#
```

可以发现刚添加的设备 simp_blkdev，且其大小为 50 MB。

(4) 格式化设备 simp_blkdev，命令如下：

```
# mkfs.ext3 /dev/simp_blkdev //表示在块设备 simp-blkdev 上建立 ext3 文件系统。
```

结果如下：

```
root@ubuntu:/home/kh/test/test10/simp_blkdev# mkfs.ext3 /dev/simp_blkdev
mke2fs 1.44.1 (24-Mar-2018)
创建含有 51200 个块（每块 1k）和 12824 个inode的文件系统
文件系统UUID：7bd989f1-0e54-49f7-9e8c-d3db2a105f5e
超级块的备份存储于下列块：
        8193, 24577, 40961

正在分配组表：  完成
正在写入inode表：  完成
创建日志（4096 个块）  完成
写入超级块和文件系统账户统计信息：  已完成
```

创建成功

(5) 创建挂载点并挂载块设备，命令如下：

```
# mkdir -p /mnt/temp1
# mount /dev/simp_blkdev /mnt/temp1
# mount | grep simp_blkdev
```

结果如下：

```
root@ubuntu:/home/kh/test/test10/simp_blkdev# mount /dev/simp_blkdev /mnt/temp1
root@ubuntu:/home/kh/test/test10/simp_blkdev# mount | grep simp_blkdev
/dev/simp_blkdev on /mnt/temp1 type ext3 (rw,relatime,data=ordered)
```

挂载成功。

(6) 再次查看模块使用情况，命令如下：

```
# lsmod | grep simp_blkdev
```

结果如下：

```
root@ubuntu:/home/kh/test/test10/simp_blkdev# lsmod | grep simp_blkdev
simp_blkdev         52445184  1
```

发现模块已被调用，且使用者为 1，如图 12.11 所示。

(7) 对块设备驱动进行调用测试。进入/mnt/temp1 目录，并尝试复制文件到该目录下，以验证是否可以使用该设备，命令如下：

```
# cp /etc/init.d/* /mnt/temp1/  //表示将另一个目录中的文件复制到挂载目录下。
# ls /mnt/temp1/  //表示查看复制进来的文件。
```

结果如下：

```
root@ubuntu:/home/kh/test/test10/simp_blkdev# cp /etc/init.d/* /mnt/temp1/
root@ubuntu:/home/kh/test/test10/simp_blkdev# ls /mnt/temp1/
acpid            cups-browsed      lost+found       saned
alsa-utils       dbus              networking       speech-dispatcher
anacron          dns-clean         network-manager  spice-vdagent
apparmor         gdm3              open-vm-tools    udev
apport           grub-common       plymouth         ufw
avahi-daemon     hwclock.sh        plymouth-log     unattended-upgrades
bluetooth        irqbalance        pppd-dns         uuidd
console-setup.sh kerneloops        procps           whoopsie
cron             keyboard-setup.sh rsync            x11-common
cups             kmod              rsyslog
```

复制成功。

(8) 查看资源使用情况，命令如下：

```
# df -h //表示查看资源使用情况
```

结果如下：

```
/dev/loop25       303M  303M     0  100% /snap/code/128
/dev/loop27       303M  303M     0  100% /snap/code/129
tmpfs             393M   16K  393M    1% /run/user/121
tmpfs             393M   40K  393M    1% /run/user/1000
tmpfs             393M     0  393M    0% /run/user/0
/dev/simp_blkdev   45M  935K   41M    3% /mnt/temp1
```

可以发现，新增的文件系统的资源使用情况为 3%。

(9) 删除文件并再次查看资源使用情况，如图 12.14 所示。

```
# rm -rf /mnt/temp1/*  //表示删除挂载目录中的所有文件。
# df -h | grep simp //表示再次查看资源使用情况，结果变成了 2%。
```

结果如下：

```
root@ubuntu:/home/kh/test/test10/simp_blkdev# rm -rf /mnt/temp1/*
root@ubuntu:/home/kh/test/test10/simp_blkdev# df -h | grep simp
/dev/simp_blkdev   45M  815K   42M    2% /mnt/temp1
```

可以发现，新增的文件系统的资源使用情况为变成了 2%

(10) 取消挂载，查看模块调用情况，命令如下：

```
# umount /mnt/temp1/
# lsmod | grep simp_blkdev
```

结果如下：

```
root@ubuntu:/home/kh/test/test10/simp_blkdev# umount /mnt/temp1/
root@ubuntu:/home/kh/test/test10/simp_blkdev# lsmod | grep simp_blkdev
simp_blkdev         52445184  0
```

可以发现，调用数已从 1 变回 0 。

(11) 卸载模块并查找，命令如下：

```
# rmmod simp_blkdev  //卸载模块
# lsmod | grep simp_blkdev  //查找模块
```

结果如下：

```
root@ubuntu:/home/kh/test/test10/simp_blkdev# rmmod simp_blkdev
root@ubuntu:/home/kh/test/test10/simp_blkdev# lsmod | grep simp_blkdev
root@ubuntu:/home/kh/test/test10/simp_blkdev#
```

可以发现，模块一旦被卸载，就再也找不到了，

五、 实验思考与总结

1. （编写字符设备驱动程序）修改测试文件，实现向字符设备写数据。

```c
int main()
{
    char buf[4096] = {"I have already told you 1 time hello world"};
    int fd =open("/dev/hello",O_RDWR );
    int ret = write(fd ,buf ,sizeof(buf));
    buf[ret] = '\0';
    printf("%s\n",buf);
}
```

2. （编写块设备驱动程序）分析字符设备与块设备的驱动程序，指出它们在实现过程中的异同点。

- 相同点：设备驱动程序编写实现的大体框架相同，都是先要编写设备驱动源代码，然后使用 make 命令编译成内核驱动模块，再使用 insmod 命令安装模块，最后测试模块。
- 不同点：字符设备驱动的源代码与块设备驱动程序的源代码不同，Makefile 文件内容不同。安装完模块后，这两个设备的使用方式也不同。

3. 实验总结

本次实验是 Linux 设备管理的实验，在编写字符设备驱动程序中，我们编写了一个字符设备驱动程序，实现了对字符设备执行打开、读、写、I/O 控制和关闭这 5 项基本操作，同时编写了一个测试程序，成功测试了添加的字符设备和块设备驱动程序的正确性。

在编写块设备驱动程序实验中，编写了一个简单的块设备驱动程序，实现了内存中的虚拟磁盘驱动器，并通过实际操作验证块设备驱动是否可以正常工作。本次实验的两种方法都是通过编写模块、装载模块的方式来实现对应功能的，区别在于字符设备驱动和块设备驱动程序具体的实现方法不同。在实验的过程中，我了解了 Linux 字符设备管理机制与基本管理方法，对于 Linux 块设备管理机制有了更深的理解，同时学会了编写一个简单的块设备驱动程序的方法。