中山大學

SUN YAT-SEN UNIVERSITY

实 验 报 告

课程名称：操作系统

姓　　名：方桂安

学　　号：**20354027**

专业班级：**2020** 级智能科学与技术

任课教师：吴贺俊

2022 年 12 月 17 日

实验报告成绩评定表

评定项目	内　　容	满 分	评　分	总　分
实验态度	态度端正、遵守纪律、出勤情况	10		
实验过程	按要求完成算法设计、代码书写、注释清晰、运行结果正确	30		
实验记录	展示讲解清楚、任务解决良好、实验结果准确	20		
报告撰写	报告书写规范、内容条理清楚、表达准确规范、上交及时、无抄袭，抄袭记 0 分，提供报告供抄袭者扣分。	40		
评语： **指导老师签字：**　　年　　月　　日				

实验报告正文

实验八　I/O 系统 - 设备管理

一、　实验目的

- 编写字符设备驱动程序，要求能对该字符设备执行打开、读、写、I/O 控制和关闭五个基本操作。并通过应用程序，测试添加的字符设备的正确性。
- 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。

二、　实验内容

1.　任务描述

计算机系统中使用各种各样的设备。大多数设备属于存储设备（磁盘、磁带）、传输设备（网络连接、蓝牙）和人机交互设备（屏幕、键盘、鼠标、音频输入输出）。

在 Linux 系统中，一切都可以看做文件，文档、目录、键盘、监视器、硬盘、可移动媒体设备、打印机、调制解调器、虚拟终端，还有进程间通信（IPC）和网络通信等输入/输出资源都是定义在文件系统空间下的字节流。

Linux 根据设备共同特征将其划分为三大类型：字符设备；块设备；网络设备。

- 字符设备是以字节为单位进行 I/O 传输，这种字符流的传输率通常比较低。常见字符设备有鼠标、键盘、触摸屏等；
- 块设备是以块为单位传输的，常见块设备是磁盘；

- 网络设备是一类比较特殊的设备，涉及到网络协议层，因此将其单独分成一类设备。

2. 实验方案

编写字符设备驱动程序

（1）编写字符驱动源程序，即编写内核模块 chardev.c 文件和 Makefile 文件。使用 make 命令编译驱动模块。

（2）使用 insmod 命令安装驱动模块。

（3）创建字符设备文件，并查看创建的设备文件。

（4）编写一个测试程序 test.c，访问这个设备文件。用 gcc 编译这个文件，然后运行。

（5）使用 rmmod 卸载模块。

（6）使用 rm 命令删除创建的设备文件。

编写块设备驱动程序

（1）编写设备驱动源程序，即编写内核模块 simp_blkdev.c 文件和 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 列出当前的块设备信息。

（5）格式化设备 simp_blkdev。

（6）创建挂载点并挂载该设备。

（7）查看模块使用情况，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 卸载模块。

三、 实验记录

1. 编写字符设备驱动程序

首先编写 chardev.c 文件和 Makefile 文件，用 make 命令编译

```
root@Enderfga-PC:~/chardev# make
make -C /lib/modules/5.15.79.1-microsoft-standard-WSL2/build M=/root/chardev modules
make[1]: Entering directory '/root/source/WSL2-Linux-Kernel-linux-msft-wsl-5.15.79.1'
  CC [M]  /root/chardev/chardev.o
/root/chardev/chardev.c: In function ‘device_read’:
/root/chardev/chardev.c:49:2: warning: ignoring return value of ‘copy_to_user’, declared with attribute warn_unused_resu
lt [-Wunused-result]
   49 |  copy_to_user(buffer, msg_Ptr, length);
      |  ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
  MODPOST /root/chardev/Module.symvers
  CC [M]  /root/chardev/chardev.mod.o
  LD [M]  /root/chardev/chardev.ko
  BTF [M] /root/chardev/chardev.ko
make[1]: Leaving directory '/root/source/WSL2-Linux-Kernel-linux-msft-wsl-5.15.79.1'
root@Enderfga-PC:~/chardev#
```

使用 insmod chardev.ko 命令安装编译好的字符驱动模块。并通过 lsmod 命令查看是否装载成功：

```
root@Enderfga-PC:~/chardev# insmod chardev.ko
root@Enderfga-PC:~/chardev# lsmod | grep chardev
chardev                16384  0
root@Enderfga-PC:~/chardev#
```

使用 dmesg 查看系统分配的主设备号，下图输出的“241” 即主设备号：

```
[  247.766668] <1> I was assigned major number 241
[  247.766671] <1> the drive, create a dev file
[  247.767175] <1> mknod /dev/hello c 241 0.
[  247.767844] <1> I was assigned major number 241
[  247.767845] <1> the device file
[  247.768535] <1> Remove the file device and module when done
root@Enderfga-PC:~/chardev#
```

根据输出的主设备号，利用 mknod 命令创建设备：

```
root@Enderfga-PC:~/chardev# mknod /dev/hello c 241 0
root@Enderfga-PC:~/chardev#
```

编写一个测试程序 test.c，访问这个设备文件。用 gcc 编译这个文件，然后

运行，如图实验结果说明字符驱动设备正常工作。

```
root@Enderfga-PC:~/chardev# gcc test.c -o test
root@Enderfga-PC:~/chardev# ./test
I already told you 0 times Hello world

root@Enderfga-PC:~/chardev# ./test
I already told you 1 times Hello world

root@Enderfga-PC:~/chardev# ./test
I already told you 2 times Hello world

root@Enderfga-PC:~/chardev# ./test
I already told you 3 times Hello world

root@Enderfga-PC:~/chardev# ./test
I already told you 4 times Hello world

root@Enderfga-PC:~/chardev#
```

实验成功，卸载模块和设备。

四、 总结与讨论

本次实验虽然只完成了**编写字符设备驱动程序**的部分，但也让我收获了许多知识：

1. 了解字符设备驱动程序的基本结构。字符设备驱动程序通常包含初始化、打开、读取、写入、关闭和控制等函数。这些函数被用于在应用程序和设备之间传递数据。
2. 充分理解设备文件系统。在写字符设备驱动程序时，应该对设备文件系统有所了解，包括如何在系统中创建设备文件、如何使用设备文件访问设备、如何通过设备文件与设备交互等。
3. 理解设备节点和字符设备编号。在写字符设备驱动程序时，应该知道设备节点是什么，以及如何使用设备节点与设备交互。同时，应该知道字符设备编号是什么，以及如何使用字符设备编号访问设备。