OS Assignment 1

Part 2

Meenakshi Madhu

B180390CS – Group 04

Qn. Create a simple device driver (character device) for the current compiled kernel and test it with your sample application.

We have implemented a Linux device driver for a character device that reverses an input string given by the user.

For the purpose of carrying out this assignment and get a clear understanding of how a device driver works, I created and implemented a simple character device with the basic read and write operations, following which the team decided and discussed on the functionality of reversing an input string and individually implemented the program in our own systems.

Creating a device file:

The major and minor numbers of the device are allocated dynamically instead of statically. Hence mknod command need not be used.

The driver code:

Includes module_init, module_exit functions, along with the open, close, read, write functions. The functionality of the device is to get an input string from the user (through the user program). The write function call writes it into a write buffer in the user space, which is then copied onto a kernel space buffer. This is then reversed and stored in a string in the kernel space.

When the read function is called, the reversed string is read from the string in kernel space and displayed as the output.

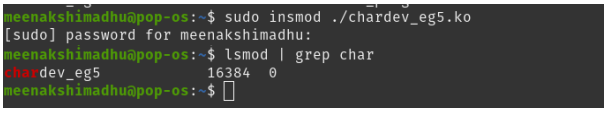
Compiling the driver:

For the compilation of the linux module, we use the kernel Makefile, and the make command.

```
meenakshimadhu@pop-os:~$ make -C /lib/modules/$(uname -r)/build M=$PWD modules
make: Entering directory '/usr/src/linux-headers-5.4.0-7642-generic'
  CC [M]  /home/meenakshimadhu/chardev_eg5.o
/home/meenakshimadhu/chardev_eg5.c: In function 'my_write5':
/home/meenakshimadhu/chardev_eg5.c:80:19: warning: unused variable 'i' [-Wunused-variable]
   80 |  short count = 0, i = 100;
      |                   ^
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/meenakshimadhu/chardev_eg5.mod.o
  LD [M]  /home/meenakshimadhu/chardev_eg5.ko
make: Leaving directory '/usr/src/linux-headers-5.4.0-7642-generic'
meenakshimadhu@pop-os:~$ ls
```

After compilation, a .ko (kernel object) file gets created.

Loading the module:

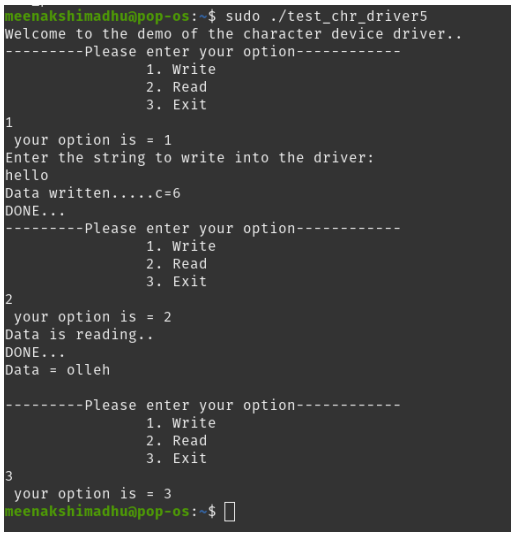


The .ko module is uploaded into the kernel using the insmod command. lsmod command can be used to check the module.

The command rmmod is used in case we need to remove/unload the module from the kernel.

Creating the user program:

Since our functionality was to reverse an input string, we have two options, to write the string in reverse, to read and display the string that was written in reverse along with an exit option. The user program is written as a C program which is compiled normally using gcc or cc.



References:

https://www.youtube.com/channel/UCQ-NwyLyw_-FUQrvXmyW_BA/videos

https://medium.com/@tp4460/linux-kernel-module-programming-a-simple-device-driver-and-a-user-level-program-accessing-it-d29753086407