Em Powers
920734539
meganpowers3
Assignment 6 – Device Driver

Project Overview

Assignment 6 – Device Driver. This project represents a simple device driver with functionality for open, read, write, ioctl, and release methods. There is a driver program that acts as a mediary between the kernel and user space, allowing the user to enter two numbers and an operation that will be processed in the kernel space and outputted as the result of the calculation. It supports basic functionality for add, subtract, multiply, and divide operations, and prints out the current program status in the kernel. This device driver is a simple integer calculator that passes an operator and two numbers to the device, performs a calculation, and outputs the answer.

Description of Work Completed

For this project, I imported the project outline from GitHub and had to decide how to construct it. I ended up determining that a main class and separate driver program bundled in the same directory was best, although I could have placed the driver program in a separate directory. For the sake of convenience, I decided to place it in the same directory as my main device driver file. After importing the project, I changed the Makefile to reflect the names of the device driver object and .ko files as opposed to the provided.

Within the main device file, I had to implement functionality for at least open, release, read, write, and ioctl. I also included an init and exit function for the driver. My work completed on each of these functions was as follows:

- devchar_init
    - The purpose of this method is to initialize the device driver with the major number – or representative number for the driver in the system – as well as the device class, and the device itself. This included the use of the printk function to present alerts in the kernel showing the progress of device initialization, as well as error checking as to whether the kernel is building successfully.
- devchar_exit
    - This function destroys and unregisters the device, the class, and the major number from the system.
- dev_open
    - Counts the amount of times a device has been opened and alerts the user as to that information.

- dev_read
    - Provides functionality for copying the contents of a file pointed to by filep from kernel space to user space though the copy_to_user function. Alerts the user as to any errors that occur in the process.
- dev_write
    - Provides functionality for copying the two numbers to be calculated, alongside the operator. Increments a counter keeping track of what number is what – i.e. 0 indicating the operator, 1 indicating the left hand side of the operation, and 2 indicating the right hand side.
- dev_ioctl
    - This provides the actual calculator interface. It takes the value of operators gotten by dev_write and summarily performs the operation on them specified by the user.
- dev_release
    - This functionality alerts users that the device has been successfully closed and released.

I also had to import kernel modules, name the device and class, and provide module licenses. Alongside, I initialized variables pertaining to the calculator and class, as well as instantiating _IOW/_IOR functionality for the ioctl. Finally, I instantiated methods and mapped them to file operations.

For my tester, I had to import in libraries, instantiate _IOW and _IOR for the ioctl, as well as define the numbers that would be used in the calculator. In main, I alerted the user to the fact that their code was starting, as well as opening the sampledriver device. After error checking, I made calls to printf, scanf, and write to take in the user's calculated numbers and output the result by using read. Finally, I closed the device driver.

Project Issues

While I did not have extreme difficulty grasping the concept of a Linux kernel because I knew what it was from prior, actually implementing it proved to be a large challenge. I had direction for how to run a program in kernelspace, but actually building the program with all of the extra steps, including performing insmod to load the device driver, was something I struggled with. When I was first testing my program and figuring out what I wished it to do, I was not aware that I had to rmmod, or unload the device driver every time I wished for the kernelspace part of the program to be loaded again.

Additionally, I faced difficulties in having the program compile as the syntax of kernel functions was so different from user functions. One example where I faced difficulties in naming conventions was even getting my program to run because I had changed the device name to something different while it was still loaded in the /dev file. It took me a while to figure out that I needed to change the name in the code to the name of the device before I ran it. Another example was in using printk and KERN_LOG to display the current results of my program.

Another issue I faced was in determining exactly what device driver I wanted to code. There were a few examples given but the possibilities were so vast that I wanted to try a USB driver, a file driver, a network device driver, and so on. A lot of these possibilities were too intensive for two weeks. Originally, I wanted to do a buffered IO device driver from files like our assignments, but I ran into difficulties implementing it such as maintaining a struct for the device that caused me to rethink the trajectory of my project as I had been spending too much time on trying to make a complex problem work. I did recycle most of my code into the current driver.

I also was unable to make a calculator that included decimals as they are not supported in the Linux kernel. As a result, it is a purely integer calculator right now.

Finally, I had trouble in assigning functionality to the various requirements. Init seemed fairly straightforward, but figuring out what to do with ioctl especially was something I had trouble figuring out. I realized eventually that splitting the calculator functionality between ioctl and write was best practice, so I did this.

How to Build/Load/Interact with Driver

How to Build

1. The device driver code is provided in the assignment-6-device-driver-meganpowers3 folder. The makefile is configured to make both a .o and .ko file for the device_driver_main.c file.
2. The driver_tester.c file already contains functionality to load the sampledriver.
3. cd to the assignment-6-device-driver-meganpowers folder
4. Run make clean or make

```
student@student-VirtualBox:~$ cd Desktop
student@student-VirtualBox:~/Desktop$ cd assignment-6-device-driver-meganpowers3studen
t@student-VirtualBox:~/Desktop/assignment-6-device-driver-meganpowers3$ make
make -C /lib/modules/`uname -r`/build M=/home/student/Desktop/assignment-6-device-driv
er-meganpowers3 modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-52-generic'
  CC [M]  /home/student/Desktop/assignment-6-device-driver-meganpowers3/device_driver_
main.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/student/Desktop/assignment-6-device-driver-meganpowers3/device_driver_
main.mod.o
  LD [M]  /home/student/Desktop/assignment-6-device-driver-meganpowers3/device_driver_
main.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-52-generic'
cc driver_tester.c -o test
```

How to Load

1. Run sudo insmod device_driver_main.ko to load the device driver
2. Run sudo ./test

```
student@student-VirtualBox:~/Desktop/assignment-6-device-driver-meganpowers3$ sudo insmod device_
driver_main.ko
[sudo] password for student:
student@student-VirtualBox:~/Desktop/assignment-6-device-driver-meganpowers3$ sudo ./test
```

How to Interact

1. Enter a number between 1-4 , press enter
    a. 1 – Add
    b. 2 – Subtract
    c. 3 – Multiply
    d. 4 – Divide
2. Enter a left hand side , press enter
3. Enter a right hand side , press enter
4. Driver closes automatically

```
student@student-VirtualBox:~/Desktop/assignment-6-device-driver-meganpowers3$ sudo ./test
Starting device test code example...
1. Add
2. Subtract
3. Multiply
4. Divide
1
Enter first value : 3
Enter second value : 4
Writing to driver
Reading value from driver
Value is : 7
Closing driver
student@student-VirtualBox:~/Desktop/assignment-6-device-driver-meganpowers3$ sudo ./test
Starting device test code example...
1. Add
2. Subtract
3. Multiply
4. Divide
2
Enter first value : 88
Enter second value : 54
Writing to driver
Reading value from driver
Value is : 34
Closing driver
student@student-VirtualBox:~/Desktop/assignment-6-device-driver-meganpowers3$ sudo ./test
Starting device test code example...
1. Add
2. Subtract
3. Multiply
4. Divide
3
Enter first value : 4
Enter second value : 7
Writing to driver
Reading value from driver
Value is : 28
Closing driver
student@student-VirtualBox:~/Desktop/assignment-6-device-driver-meganpowers3$ sudo ./test
Starting device test code example...
1. Add
2. Subtract
3. Multiply
4. Divide
4
Enter first value : 12
Enter second value : 2
Writing to driver
Reading value from driver
Value is : 6
Closing driver
```

How to Unload

1. Run sudo rmmod device_driver_main.ko

```
student@student-VirtualBox:~/Desktop/assignment-6-device-driver-meganpowers3$ sudo rmmod device_
driver_main.ko
```

Kernel Output

Run dmesg command

```
  40.217436] rfkill: input handler disabled
 328.965162] SampleDevice: Initializing the Sample Device LKM...
 328.965165] SampleDevice: registered correctly with major number 240
 328.965171] SampleDevice: device class registered correctly
 328.965280] SampleDevice: device class created correctly
 408.683805] SampleDevice: Device has been opened 1 time(s)
 412.225152] VALUE = 0
 414.185136] VALUE = 0
 414.185139] VALUE = 0
 414.185143] SampleDevice: Sent our result of size 4 to the user
 414.185151] SampleDevice: Device has been successfully closed.
 422.313070] SampleDevice: Device has been opened 2 time(s)
 425.660712] VALUE = 0
 430.366120] VALUE = 0
 430.366122] VALUE = 0
 430.366126] SampleDevice: Sent our result of size 4 to the user
 430.366133] SampleDevice: Device has been successfully closed.
 434.306114] SampleDevice: Device has been opened 3 time(s)
 436.333565] VALUE = 0
 437.615414] VALUE = 0
 437.615416] VALUE = 0
 437.615421] SampleDevice: Sent our result of size 4 to the user
 437.615430] SampleDevice: Device has been successfully closed.
 569.257542] SampleDevice: Goodbye from the LKM!
 921.654062] SampleDevice: Initializing the Sample Device LKM...
 921.654065] SampleDevice: registered correctly with major number 240
 921.654071] SampleDevice: device class registered correctly
 921.654108] SampleDevice: device class created correctly
 927.329346] SampleDevice: Device has been opened 1 time(s)
 986.161053] VALUE = 0
 989.189756] VALUE = 0
 989.189759] VALUE = 0
 989.189764] SampleDevice: Sent our result of size 4 to the user
 989.189782] SampleDevice: Device has been successfully closed.
1003.509851] SampleDevice: Device has been opened 2 time(s)
1005.110370] VALUE = 7
1009.425599] VALUE = 7
1009.425601] VALUE = 7
1009.425609] SampleDevice: Sent our result of size 4 to the user
1009.425618] SampleDevice: Device has been successfully closed.
1013.420644] SampleDevice: Device has been opened 3 time(s)
1017.095558] VALUE = 34
1019.073240] VALUE = 34
1019.073242] VALUE = 34
1019.073246] SampleDevice: Sent our result of size 4 to the user
1019.073253] SampleDevice: Device has been successfully closed.
1023.695446] SampleDevice: Device has been opened 4 time(s)
1025.242219] VALUE = 28
1029.297826] VALUE = 28
1029.297828] VALUE = 28
1029.297832] SampleDevice: Sent our result of size 4 to the user
1029.297839] SampleDevice: Device has been successfully closed.
1119.192586] SampleDevice: Goodbye from the LKM!
```