HW 2 Documentation
Philippe Proctor

Setting up char device question: To create the device file in /dev, the command sudo mknod */dev / driver_setup* must be executed. This hooks up the allocated major node to our specified character device.

1. **Source code for make file:**

```
KERNEL_DIR = /lib/modules/$(shell uname -r)/build
PWD := $(shell pwd)

obj-m += driver_setup.o

default:
	$(MAKE) -C $(KERNEL_DIR) SUBDIRS=$(PWD) modules

clean:
	$(MAKE) -C $(KERNEL_DIR) SUBDIRS=$(PWD) clean
```

Source code for module:

```
/*
 * Philippe Proctor
 * 4/17/2019
 * ECE 373
 *
 * Hw 2 part 1: Setup driver module
 */

#include <linux/module.h>
#include <linux/types.h>
#include <linux/kdev_t.h>
#include <linux/fs.h>
#include <linux/cdev.h>
#include <linux/slab.h>
#include <linux/uaccess.h>

#define DEVCNT 1
#define DEVNAME "p_dev"

/*Start of private driver data struct, passed around inside driver*/


static struct mydev_dev {
	struct cdev cdev;
	int syscall_val;
} mydev;

static dev_t mydev_node;
```

```
/*Allows user to change param w/ in driver module*/
static int input = 40;
module_param(input, int, S_IRUSR | S_IWUSR);

static int p_dev_open(struct inode *inode, struct file *file){

	printk(KERN_INFO "Succesfully opened");
	mydev.syscall_val = input;

	return 0;

}

//Read callback
static ssize_t p_dev_read(struct file *file, char __user *buf, size_t len, loff_t *offset){

	/*Get local kernel buffer set aside */
	int ret;


	/*Safety check that offset is less than int */
	if(*offset >= sizeof(int)){
		return 0;
	}

	/*Check if user passed null value to buffer */
	if(!buf) {
		ret = -EINVAL;
		goto out;		//undo allocations
	}

	printk(KERN_INFO "Inside read function");

	/*Copy value from driver struct to user space */
	if(copy_to_user(buf, &mydev.syscall_val, sizeof(int))){
		ret = -EFAULT;
		goto out;
	}

	/*Update offset by */
	ret = sizeof(int);
	*offset += len;

	printk(KERN_INFO "User got from us %d\n", mydev.syscall_val);
out:
	return ret;
};
```

```
static ssize_t p_dev_write(struct file *file, const char __user *buf, size_t len, loff_t *offset ){

	/*Local kernel memory*/
	char *kern_buf;
	int ret;

	printk(KERN_INFO "Inside write function");

	/*Check if device passed null value to buffer*/
	if(!buf){
		ret = -EINVAL;
		goto out;
	}

	/*Get memory to copy into...*/
	kern_buf = kmalloc(len, GFP_KERNEL);


	if(copy_from_user(kern_buf, buf, len)){
		ret = -EFAULT;
		goto mem_out;
	}

	ret = len;

	mydev.syscall_val = *kern_buf;

	printk(KERN_INFO "Userspace wrote %s to us\n",kern_buf );

mem_out:
	kfree(kern_buf);

out:
	return ret;

}


static struct file_operations mydev_fops = {
	//fields
	.owner = THIS_MODULE,
	.open = p_dev_open,
	.read = p_dev_read,
	.write = p_dev_write,

};

static int __init p_device_init(void) {
```

```
	printk(KERN_INFO "p_device loading...");

	if (alloc_chrdev_region(&mydev_node, 0, DEVCNT, DEVNAME)) {
		printk(KERN_ERR "alloc_chrdev_region() failed!\n");
		return -1;
	}

	printk(KERN_INFO "Allocated %d devices at major:%d\n",DEVCNT, MAJOR(mydev_node));


	/*Initialize the char device and add it to the kernel*/

	cdev_init(&mydev.cdev, &mydev_fops);
	mydev.cdev.owner = THIS_MODULE;

	if (cdev_add(&mydev.cdev, mydev_node, DEVCNT)) {
		printk(KERN_ERR "cdev_add() failed!\n");


		/*clean up chrdev alloc */
		unregister_chrdev_region(mydev_node, DEVCNT);
		return -1;
	}

	return 0;
}

static void __exit p_device_exit(void) {


	/*destroy the cdev */
	cdev_del(&mydev.cdev);

	/*clean up the devices */
	unregister_chrdev_region(mydev_node, DEVCNT);

	printk(KERN_INFO "%s module unloaded!\n", DEVNAME);
}

MODULE_AUTHOR("Philippe Proctor");
MODULE_LICENSE("GPL");
MODULE_VERSION("0.2");
module_init(p_device_init);
module_exit(p_device_exit);
```

2. Userspace program:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <fcntl.h>
#include <sys/select.h>
#include <errno.h>

#define PATH_READ "/dev/driver_setup"
#define PATH_WRITE "/sys/module/driver_setup/parameters"


int main(int argc, char **argv){

	int fd,rea,wri;
	int count = 2;
	int input_buf[1] = {0};
	int output_buf[1] = {7};

	if((fd = open(PATH_READ,O_RDWR /* O_RDWR*/)) < 0){
		printf("File not found.\n");
		perror("Dev open:");
		return errno;
	}


	if((rea = read(fd, input_buf, count)) < 0){
		printf("Bytes not read,rea = %d\n", rea);
		perror("Dev read");
		return errno;
	}

	printf("Value read from driver: %d\n",input_buf[0]);

	if((wri = write(fd, output_buf, count)) < 0){
		printf("Bytes not written, wri = %d\n",wri);
		perror("Dev write");
		return errno;

	}

	printf("Value written to driver: %d\n",output_buf[0]);

	if((rea = read(fd, input_buf, count)) < 0){
		printf("Bytes not read,rea = %d\n", rea);
		perror("Dev read");
		return errno;
```

```
        }

        printf("Value read from driver: %d\n",input_buf[0]);


        close(fd);
}
```

3A. Typescript of loading module before parameter:

```
Script started on 2019-04-21 16:59:28-0700
pproctor@pproctor-MacBook: ~/Documents/ECE373/hw2 sudo insmod driver_setup.ko
pproctor@pproctor-MacBook: ~/Documents/ECE373/hw2 sudo mknod dev/driver_setup c 237 0
pproctor@pproctor-MacBook: ~/Documents/ECE373/hw2 cd /dev
pproctor@pproctor-MacBook: /dev sudo chmod 777 driver_setup
pproctor@pproctor-MacBook: /dev cd Documents/ECE373/hw2
pproctor@pproctor-MacBook: ~/Documents/ECE373/hw2 ./tests
Value read from driver: 40
Value written to driver: 7
Value read from driver: 7
pproctor@pproctor-MacBook: ~/Documents/ECE373/hw2 exit
exit

Script done on 2019-04-21 17:01:34-0700
```

3B. Typescript of loading module after parameter:

```
Script started on 2019-04-21 17:21:08-0700
pproctor@pproctor-MacBook: ~/Documents/ECE373/hw2 cd /sys/module/driver_setup/parameters/
pproctor@pproctor-MacBook: /sys/module/driver_setup/parameters sudo su
[sudo] password for pproctor:
root@pproctor-MacBook: /sys/module/driver_setup/parameters cat input
40
root@pproctor-MacBook: /sys/module/driver_setup/parameters echo 97 > input
root@pproctor-MacBook: /sys/module/driver_setup/parameters cat input
97
root@pproctor-MacBook: /sys/module/driver_setup/parameters exit
exit
pproctor@pproctor-MacBook: cd Documents/ECE373/hw2
pproctor@pproctor-MacBook: ~/Documents/ECE373/hw2 ./test_script
Value read from driver: 97
Value written to driver: 7
Value read from driver: 7
pproctor@pproctor-MacBook: cd /sys/module/driver_setup/parameters
pproctor@pproctor-MacBook: /sys/module/driver_setup/parameters sudo su
root@pproctor-MacBook: /sys/module/driver_setup/parameters cat input
97
root@pproctor-MacBook: /sys/module/driver_setup/parameters exit
pproctor@pproctor-MacBook: exit
exit

Script done on 2019-04-21 17:23:04-0700
```

4. Typescript showing loaded module:

```
Script started on 2019-04-22 12:35:52-0700
pproctor@pproctor-MacBook: ~/Documents/ECE373/hw2 cd /proc
pproctor@pproctor-MacBook: /proc vim devices
 12  10 misc
 13  13 input
 14  21 sg
 15  29 fb
 16  81 video4linux
 17  89 i2c
 18  99 ppdev
 19 108 ppp
 20 116 alsa
 21 128 ptm
 22 136 pts
 23 180 usb
 24 189 usb_device
 25 204 ttyMAX
 26 216 rfcomm
 27 226 drm
 28 237 p_dev  <------****my module
 29 238 hidraw
 30 239 media
 31 240 aux
 32 241 iio
 33 242 mei
pproctor@pproctor-MacBook /procKexit
exit

Script done on 2019-04-22 12:36:19-0700
```