DEVICE DRIVERS LAB 4

DONE BY:
A S V DHANUSH
CS20B1057

1)

Make File

Makefile ~/cs20b1057_dd_lab/lab4

```
obj-m += chr_ioctl.o
KDIR = /lib/modules/$(shell uname -r)/build
all:
	make -C $(KDIR)  M=$(shell pwd) modules

clean:
	make -C $(KDIR)  M=$(shell pwd) clean

```

chr_ioctl.c (ioctl creation file)-

Code

```
#include<linux/kernel.h>
#include<linux/init.h>
#include<linux/module.h>
#include<linux/kdev_t.h>
#include<linux/fs.h>
#include<linux/cdev.h>
#include<linux/device.h>
#include<linux/slab.h>
#include<linux/uaccess.h>
#include<linux/ioctl.h>

#define mem_size 1024  // Macro for memory size

// Define the ioctl code
#define WR_DATA _IOW('a','a',int32_t*)
#define RD_DATA _IOR('a','b',int32_t*)

int32_t val=0;

dev_t dev = 0;
static struct class *dev_class;
static struct cdev my_cdev;

uint8_t *kernel_buffer;

static int __init chr_driver_init(void);
static void __exit chr_driver_exit(void);

static int my_open(struct inode *inode, struct file *file);
static int my_release(struct inode *inode, struct file *file);
static ssize_t my_read(struct file *filp, char __user *buf, size_t len, loff_t *off);
static ssize_t my_write(struct file *filp, const char *buf, size_t len, loff_t *off);
static long my_ioctl(struct file *file,unsigned int cmd,unsigned long arg);

static struct file_operations fops=
{
        .owner          =       THIS_MODULE,
        .read           =       my_read,
        .write          =       my_write,
        .open           =       my_open,
        .unlocked_ioctl =       my_ioctl,
        .release        =       my_release,
};

static int my_open(struct inode *inode, struct file *file)
{
        // Creating physical Memory
        if((kernel_buffer = kmalloc(mem_size, GFP_KERNEL))==0)
        {
                printk(KERN_INFO"Can NOT allocate the memory to kernel ...\n");
                return -1;
        }
        printk(KERN_INFO "Device File Opened...\n");
        return 0;
}
```

execute 1st file with make cmd

```
user@user:~/cs20b1057_dd_lab/lab4$ make
make -C /lib/modules/5.15.0-58-generic/build   M=/home/user/cs20b1057_dd_lab/lab4 modules
make[1]: Entering directory '/usr/src/linux-headers-5.15.0-58-generic'
make[1]: Leaving directory '/usr/src/linux-headers-5.15.0-58-generic'
user@user:~/cs20b1057_dd_lab/lab4$ ls
a.out        chr_ioctl.ko   chr_ioctl.mod.c  chr_ioctl.o     chr_ioctl_test.c  modules.order
chr_ioctl.c  chr_ioctl.mod  chr_ioctl.mod.o  chr_ioctl_test  Makefile          Module.symvers
user@user:~/cs20b1057_dd_lab/lab4$
```

2)

chr_ioctl_test.c (user space application)

Code

```c
#include <sys/fcntl.h>
#include <sys/stat.h>
#include <sys/ioctl.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include<linux/ioctl.h>

#define WR_DATA _IOW('a','a',int32_t*)
#define RD_DATA _IOR('a','b',int32_t*)

int main()
{
        int fd;
        int32_t value, number;


        printf("\nOpening Driver\n");
        fd = open("/dev/my_device", O_RDWR);
        if(fd < 0) {
                printf("Cannot open device file...\n");
                return 0;
        }

        printf("Enter the Value to send\n");
        scanf("%d",&number);
        printf("Writing Value to Driver\n");
        ioctl(fd, WR_DATA, (int32_t*) &number);

        printf("Reading Value from Driver\n");
        ioctl(fd, RD_DATA, (int32_t*) &value);
        printf("Value is %d\n", value);

        printf("Closing Driver\n");
        close(fd);
}

```

Compile 2nd file using the below commands
cc chr_ioctl_test.c -o
chr_ioctl_testsudo ./chr_ioctl_test.o

OUTPUT

```
user@user:~/cs20b1057_dd_lab/lab4$ sudo insmod chr_ioctl.ko
user@user:~/cs20b1057_dd_lab/lab4$ gcc chr_ioctl_test.c
user@user:~/cs20b1057_dd_lab/lab4$ sudo ./a.out

Opening Driver
Enter the Value to send
12345678
Writing Value to Driver
Reading Value from Driver
Value is 12345678
Closing Driver
user@user:~/cs20b1057_dd_lab/lab4$ sudo ./a.out

Opening Driver
Enter the Value to send
dhanush
Writing Value to Driver
Reading Value from Driver
Value is 1952457568
Closing Driver
user@user:~/cs20b1057_dd_lab/lab4$
```

(Note : for non numeric values we get a junk value as the return type of the IOCTL function is of the type “int”)

```
int ioctl(int fd, int request, ...)
```

Removing the module

```
user@user:~/cs20b1057_dd_lab/lab4$ sudo rmmod chr_ioctl
user@user:~/cs20b1057_dd_lab/lab4$ dmesg|tail -1
[ 2571.060175] Device Driver is Removed Successfully...
user@user:~/cs20b1057_dd_lab/lab4$
```