Sistem Programlama Final Ödevi

“Character Device Uygulaması”

1306160048 Barış YARAR

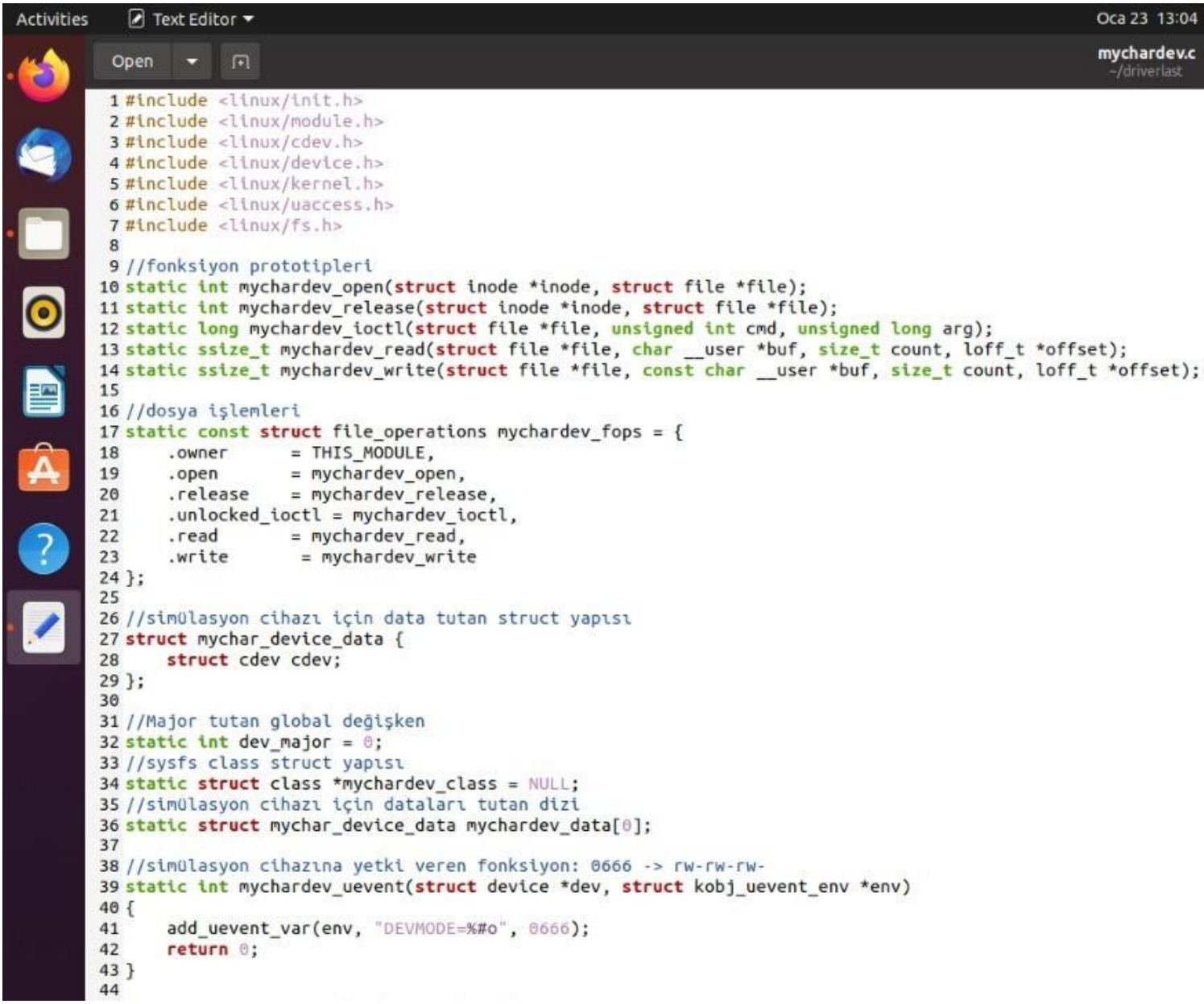
1306150015 Onur ERMAN

1306151327 Mustafa TAYYAR

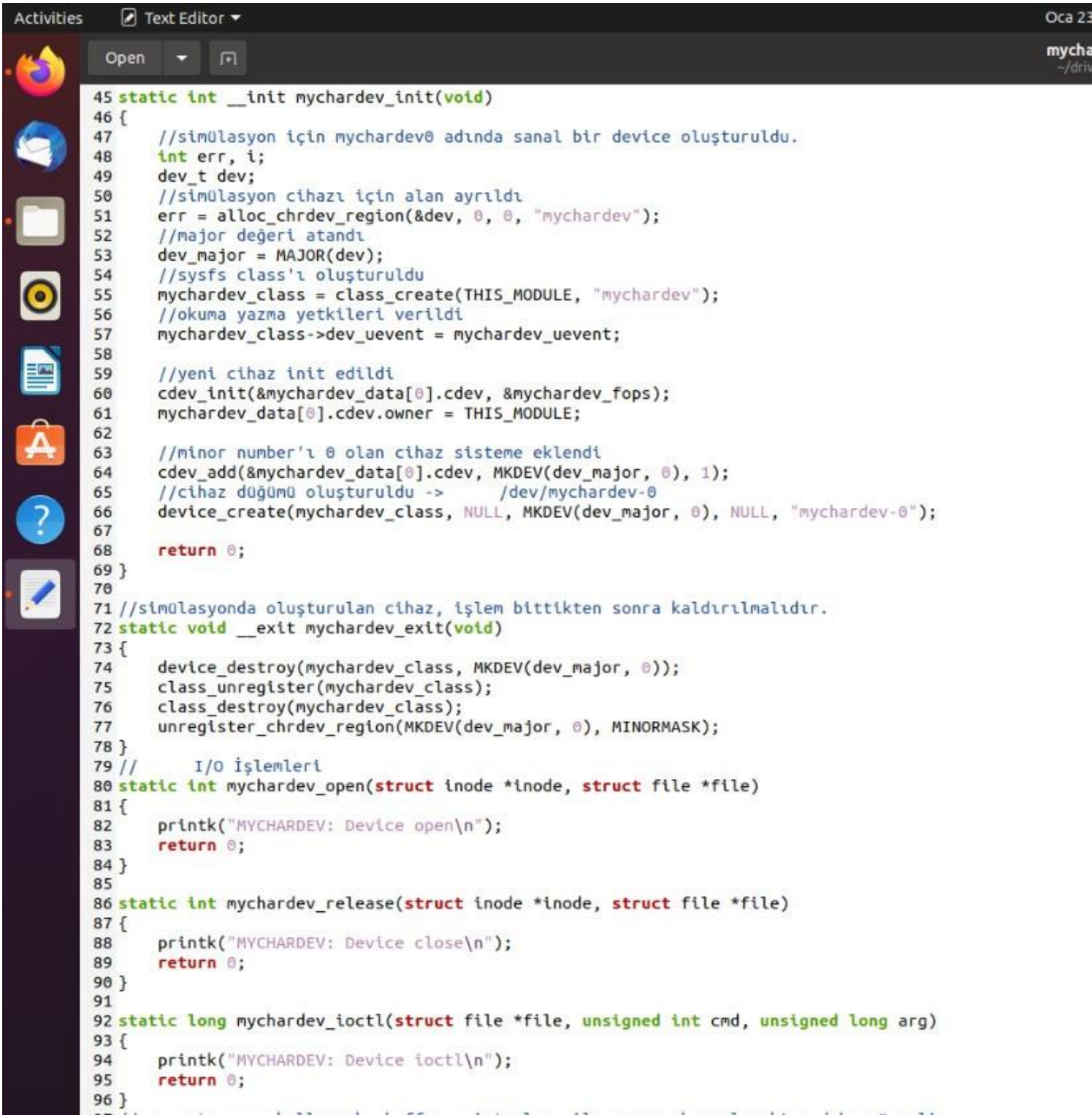
Bu projede simülasyon kullanılmıştır. Ubuntu kernel 5.9.2-generic sürümünde denenmiştir.

Arduino üzerindeki testlerimizde arduino uno cihazına veri yazdırdıktan sonra arduinonun kendini yeniden başlatması gibi birtakım sorunlar yaşayıp projenin simülasyon üzerinde gösterilmesi uygun görülmüştür.

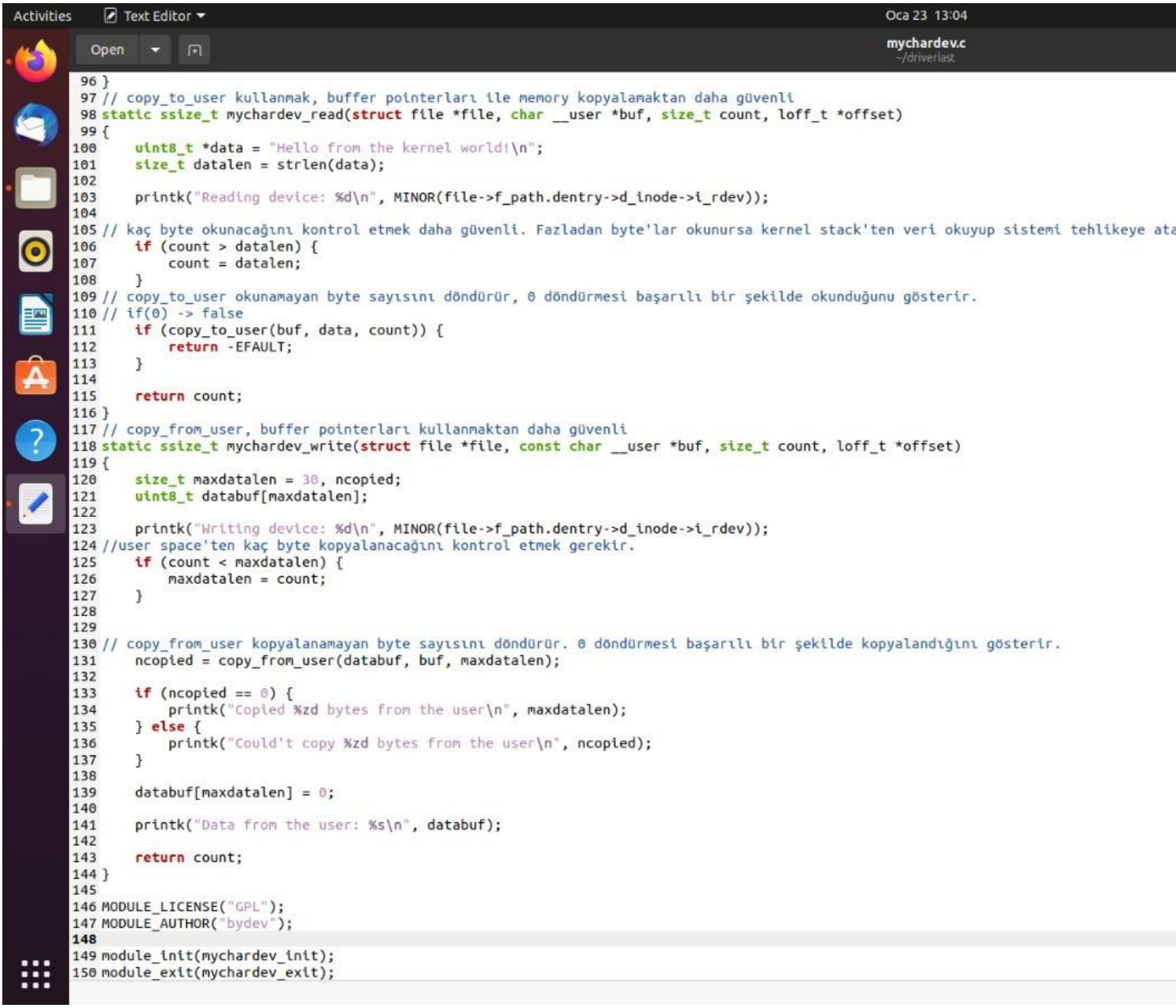
Kodlar




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

//fonksiyon prototipleri
static int mychardev_open(struct inode *inode, struct file *file);
static int mychardev_release(struct inode *inode, struct file *file);
static long mychardev_ioctl(struct file *file, unsigned int cmd, unsigned long arg);
static ssize_t mychardev_read(struct file *file, char __user *buf, size_t count, loff_t *offset);
static ssize_t mychardev_write(struct file *file, const char __user *buf, size_t count, loff_t *offset);

//dosya işlemleri
static const struct file_operations mychardev_fops = {
    .owner      = THIS_MODULE,
    .open       = mychardev_open,
    .release    = mychardev_release,
    .unlocked_ioctl = mychardev_ioctl,
    .read       = mychardev_read,
    .write       = mychardev_write
};

//simülasyon cihazı için data tutan struct yapısı
struct mychar_device_data {
    struct cdev cdev;
};

//Major tutan global değişken
static int dev_major = 0;
//sysfs class struct yapısı
static struct class *mychardev_class = NULL;
//simülasyon cihazı için dataları tutan dizi
static struct mychar_device_data mychardev_data[0];

//simülasyon cihazına yetki veren fonksiyon: 0666 -> rw-rw-rw-
static int mychardev_uevent(struct device *dev, struct kobj_uevent_env *env)
{
    add_uevent_var(env, "DEVMODE=%#o", 0666);
    return 0;
}
```




```c
static int __init mychardev_init(void)
{
    //simülasyon için mychardev0 adında sanal bir device oluşturuldu.
    int err, i;
    dev_t dev;
    //simülasyon cihazı için alan ayrıldı
    err = alloc_chrdev_region(&dev, 0, 0, "mychardev");
    //major değeri atandı
    dev_major = MAJOR(dev);
    //sysfs class'ı oluşturuldu
    mychardev_class = class_create(THIS_MODULE, "mychardev");
    //okuma yazma yetkileri verildi
    mychardev_class->dev_uevent = mychardev_uevent;

    //yeni cihaz init edildi
    cdev_init(&mychardev_data[0].cdev, &mychardev_fops);
    mychardev_data[0].cdev.owner = THIS_MODULE;

    //minor number'ı 0 olan cihaz sisteme eklendi
    cdev_add(&mychardev_data[0].cdev, MKDEV(dev_major, 0), 1);
    //cihaz düğümü oluşturuldu ->      /dev/mychardev-0
    device_create(mychardev_class, NULL, MKDEV(dev_major, 0), NULL, "mychardev-0");

    return 0;
}

//simülasyonda oluşturulan cihaz, işlem bittikten sonra kaldırılmalıdır.
static void __exit mychardev_exit(void)
{
    device_destroy(mychardev_class, MKDEV(dev_major, 0));
    class_unregister(mychardev_class);
    class_destroy(mychardev_class);
    unregister_chrdev_region(MKDEV(dev_major, 0), MINORMASK);
}
//      I/O İşlemleri
static int mychardev_open(struct inode *inode, struct file *file)
{
    printk("MYCHARDEV: Device open\n");
    return 0;
}

static int mychardev_release(struct inode *inode, struct file *file)
{
    printk("MYCHARDEV: Device close\n");
    return 0;
}

static long mychardev_ioctl(struct file *file, unsigned int cmd, unsigned long arg)
{
    printk("MYCHARDEV: Device ioctl\n");
    return 0;
}
```




```c
}
// copy_to_user kullanmak, buffer pointerları ile memory kopyalamaktan daha güvenli
static ssize_t mychardev_read(struct file *file, char __user *buf, size_t count, loff_t *offset)
{
    uint8_t *data = "Hello from the kernel world!\n";
    size_t datalen = strlen(data);

    printk("Reading device: %d\n", MINOR(file->f_path.dentry->d_inode->i_rdev));

// kaç byte okunacağını kontrol etmek daha güvenli. Fazladan byte'lar okunursa kernel stack'ten veri okuyup sistemi tehlikeye ata
    if (count > datalen) {
        count = datalen;
    }
// copy_to_user okunamayan byte sayısını döndürür, 0 döndürmesi başarılı bir şekilde okunduğunu gösterir.
// if(0) -> false
    if (copy_to_user(buf, data, count)) {
        return -EFAULT;
    }

    return count;
}
// copy_from_user, buffer pointerları kullanmaktan daha güvenli
static ssize_t mychardev_write(struct file *file, const char __user *buf, size_t count, loff_t *offset)
{
    size_t maxdatalen = 30, ncopied;
    uint8_t databuf[maxdatalen];

    printk("Writing device: %d\n", MINOR(file->f_path.dentry->d_inode->i_rdev));
//user space'ten kaç byte kopyalanacağını kontrol etmek gerekir.
    if (count < maxdatalen) {
        maxdatalen = count;
    }


// copy_from_user kopyalanamayan byte sayısını döndürür. 0 döndürmesi başarılı bir şekilde kopyalandığını gösterir.
    ncopied = copy_from_user(databuf, buf, maxdatalen);

    if (ncopied == 0) {
        printk("Copied %zd bytes from the user\n", maxdatalen);
    } else {
        printk("Could't copy %zd bytes from the user\n", ncopied);
    }

    databuf[maxdatalen] = 0;

    printk("Data from the user: %s\n", databuf);

    return count;
}

MODULE_LICENSE("GPL");
MODULE_AUTHOR("bydev");

module_init(mychardev_init);
module_exit(mychardev_exit);
```

Makefile

Activities Text Editor

Open Makefile ~/driverlast Save

```
1 BINARY      := mychardev
2 KERNEL      := /lib/modules/$(shell uname -r)/build
3 ARCH        := x86
4 C_FLAGS     := -Wall
5 KMOD_DIR    := $(shell pwd)
6 TARGET_PATH := /lib/modules/$(shell uname -r)/kernel/drivers/char
7
8 ccflags-y += $(C_FLAGS)
9
10 obj-m += mychardev.o
11
12 $(BINARY)-y := $(OBJECTS)
13
14 $(BINARY).ko:
15         make -C $(KERNEL) M=$(KMOD_DIR) modules
16
17 install:
18         cp $(BINARY).ko $(TARGET_PATH)
19         depmod -a
20
21 uninstall:
22         rm $(TARGET_PATH)/$(BINARY).ko
23         depmod -a
24
25 clean:
26         make -C $(KERNEL) M=$(KMOD_DIR) clean
```

Test Aşaması

echo “test test test” -> /dev/mychardev-0

echo “-------Selamlar-------” -> /dev/mychardev-0

Komutlarıyla mychardev-0 adlı cihaza “test test test” ve “-----Selamlar-----” yazdırılır.