微算機系統實習

LAB 06

組別：19

109590014　　沈煒翔

109590015　　楊挺煜

109590023　　廖堃霖

日期：111/5/23

2. 實驗

項目一

hellod.c

```c
#include <linux/kernel.h>
#include <linux/module.h>
static int __init tx2_hello_module_init(void)
{
printk("Hello, TX2 module is installed !\n");
return 0;
}
static void __exit tx2_hello_module_cleanup(void)
{
printk("Good-bye, TX2 module was removed!\n");
}
module_init(tx2_hello_module_init);
module_exit(tx2_hello_module_cleanup);
MODULE_LICENSE("GPL");
```

Makefile

```
obj-m := hellod.o
kernel_DIR = /usr/src/linux-headers-4.9.201-tegra-ubuntu18.04_aarch64/kernel-4.9/ #有可能需要更改位址

PWD := $(shell pwd)
all:
    make -C $(kernel_DIR) SUBDIRS=$(PWD)
clean:
    rm *.o *.ko *mod.C
.PHONY:
    clean
```

項目二

demo.c

```c
#include <linux/init.h>
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/fs.h>
#include <asm/uaccess.h>
#define MAJOR_NUM 60
#define MODULE_NAME "demo"

static int iCount = 0;
static char userChar[100];

static ssize_t drv_read(struct file *filp,char *buf, size_t count, loff_t *ppos){
    printk("device read\n");
    return count;
}

static ssize_t drv_write(struct file *filp, const char *buf, size_t count, loff_t *ppos)
{
    printk("device write\n");
    printk("%d\n",iCount);
    printk("W_buf_size: %d\n",(int)count);
    copy_from_user(userChar,buf,count);
    userChar[count - 1] = 0;
    printk("userChar: %s\n", userChar);
    printk("userChar: %d\n", (int)sizeof(userChar));
    iCount++;
    return count;
}

long drv_ioctl(struct file *filp, unsigned int cmd,unsigned long arg)
{
    printk("device ioctl\n");
    return 0;
}
```

```c
static int drv_open(struct inode *inode, struct file *filp)
{
    printk("device open\n");
    return 0;
}

static int drv_release(struct inode *inode, struct file *filp)
{
    printk("device close\n");
    return 0;
}

struct file_operations drv_fops =
{
        read: drv_read,
        write: drv_write,
        unlocked_ioctl: drv_ioctl,
        open: drv_open,
        release: drv_release,
};

static int demo_init(void)
{
    if(register_chrdev(MAJOR_NUM, "demo", &drv_fops)<0)
    {
        printk("<1>%s: can't get major %d\n", MODULE_NAME, MAJOR_NUM);
        return (-EBUSY);
    }
    printk("<1>%s: started\n", MODULE_NAME);
    return 0;
}


static void demo_exit(void)
{
    unregister_chrdev(MAJOR_NUM,"demo");
    printk("<1>%s: removed\n", MODULE_NAME);
}
```

```c
module_init(demo_init);
module_exit(demo_exit);
MODULE_LICENSE("GPL");
```

test.c

```c
#include<stdio.h>
int main(){
    char buf[1024] = "Data Input 123456 hello world";
    FILE *fp = fopen("/dev/demo","w+");
    if(fp == NULL){
        printf("can't open device\n");
    }
    fwrite(buf,sizeof(buf),1,fp);
    fread(buf,sizeof(buf),1,fp);
    fclose(fp);
    return 0;
}
```

Makefile

```
obj-m := demo.o
kernel_DIR = /usr/src/linux-headers-4.9.201-tegra-ubuntu18.04_aarch64/kernel-4.9/ #有可能需要更改位址

PWD := $(shell pwd)
all:
    make -C $(kernel_DIR) M=$(PWD)
clean:
    rm *.o *.ko *mod.C
.PHONY:
    clean
```

3. 實驗影片

項目一：https://youtu.be/sa6HT7EFiRQ

項目二：https://youtu.be/YMqwSyf8uNo

4. 組員貢獻

沈煒翔：34%

楊挺煜：33%

廖堃霖：33%

5. 心得

沈煒翔：

這次關於驅動程式的製作我覺得非常的有趣，以前完全沒有這種經驗，要製作的時候感覺會花很多時間，可是這次的 lab 非常的簡單，只要照著做就 ok 了，時間也沒花很多，希望下個 lab 也可以不要花我太多時間，再來是項目二上 tx2 執行時，有一點小 bug 修了 1 個半小時才修好，結果是組員指令有一個地方打錯，也查了很多關於驅動的資料，這些資料時做起來讓我感到興趣，可能之後會在自己查詢更多資料。

楊挺煜：

這次的實驗是做驅動程式，跟之前的 GPIO 有點相似的地方是要對系統的些檔案進行修改，所以實驗流程整體上不會太複雜。不果還是有點不太了解驅動這一部份，需要再花時間搞懂。

廖堃霖：

這次的實驗是學習怎麼讓系統過載的病痛時監看系統的訊息，我負責的部分還蠻順暢的算是久久一次幸運吧