lab3-1_report

題目需求: 透過 writer 將數字寫到 driver 當中，將數字從 driver 中讀出來並使用 GPIO 點亮 LED 燈，逐秒顯示**二進制**學號。

如何實作 Writer:

在 Linux 中 “Everything is seen as a file , include device” ，因此可以透過讀寫文件的方式讓 user space 和 kernel space 進行資料的傳遞，因此透過 lab3-1_writer.c， 開啟一個名為”/dev/etx_device”可讀寫的文件，並讀取命令列中 所輸入的學號，最後逐個逐秒傳送至 driver,待所有資料傳送完畢後關閉文件,如圖 1。

```
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <fcntl.h>
#include <sys/ioctl.h>
#include <unistd.h>


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

    int fd;
    char buf[9] = {0};

    fd = open("/dev/etx_device", O_RDWR);
    if(fd < 0)
        perror("open");

    sprintf(buf, argv[1]
    for(int i=0;i<9; i++){

        write(fd,&buf[i], 1);
        sleep(1);
    }



    if(fd > 0)
        close(fd);

    return 0;
}
```

圖 1 、lab3-1_writer.c

如何實作 Driver: 大致上與 lab3-1 上課內容差不多 只對 GPIO 的數量及 ext_write 進行修改，而程式運行流程如下，insmod 後 透過 etx_driver_init 取得 Major numer、建立字元設備、確認 GPIO 狀況、設定 GPIO 為輸出。一但有指令寫入 "/dev/etx_device" 就會呼叫 etx_write 將讀取到的字元根據情況設定所需的 GPIO 腳位(奇數必亮最低位元，數字 2、3、6、7 必亮第二個位元，數字 4、5、6、7 必亮第三個位元，數字 8、9 必亮第四個位元)，待 Writer 傳送完畢所有資料後 會關閉文件 此時就會進入 etx_release 將燈號熄滅(為了消除學號最後一碼會持續亮的情形)，最後進行 rmmod 就會執行 etx_driver_xxit 釋放所有 insmod 時建立的資源。

```
static ssize_t etx_write(struct file *filp,
                         const char __user *buf, size_t len, loff_t *off)
{
  uint8_t rec_buf[1] = {0};

    if( copy_from_user( rec_buf, buf, len ) > 0) {
        pr_err("ERROR: Not all the bytes have been copied from user\n");
     }

     pr_info("Write Function : %c \n", rec_buf[0]);

    gpio_set_value(GPIO_12, 0);
    gpio_set_value(GPIO_16, 0);
    gpio_set_value(GPIO_20, 0);
    gpio_set_value(GPIO_21, 0);

     if (rec_buf[0]=='1'||rec_buf[0]=='3'||rec_buf[0]=='5'||rec_buf[0]=='7'||rec_buf[0]=='9'){
     gpio_set_value(GPIO_12, 1);
     }
     if (rec_buf[0]=='2'||rec_buf[0]=='3'||rec_buf[0]=='6'||rec_buf[0]=='7'){
     gpio_set_value(GPIO_16, 1);
         }
     if (rec_buf[0]=='4'||rec_buf[0]=='5'||rec_buf[0]=='6'||rec_buf[0]=='7'){
         gpio_set_value(GPIO_20, 1);
          }
     if (rec_buf[0]=='8'||rec_buf[0]=='9'){
         gpio_set_value(GPIO_21, 1);
         }



  return len;
}
```

圖 2、 etx_write

```
static int etx_release(struct inode *inode, struct file *file)
{


    gpio_set_value(GPIO_12, 0);
    gpio_set_value(GPIO_16, 0);
    gpio_set_value(GPIO_20, 0);
    gpio_set_value(GPIO_21, 0);
  pr_info("Device File Closed...!!!\n");
  return 0;
}
```

圖 2、 etx_release