Lab3-1 Report

AI 學程 312505017 馮凡哲

In driver.c

etx_write():

```c
static ssize_t etx_write(struct file *filp,
                         const char __user *buf, size_t len, loff_t *off)
{
  uint8_t rec_buf[10] = {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 : GPIO_21 Set = %c\n", rec_buf[0]);

  if (rec_buf[0] != '\0' && rec_buf[0] != '\n') {
  gpio_set_value(GPIO_12, ((rec_buf[0] >> 3) & 1) ? 1 : 0);
  gpio_set_value(GPIO_16, ((rec_buf[0] >> 2) & 1) ? 1 : 0);
  gpio_set_value(GPIO_20, ((rec_buf[0] >> 1) & 1) ? 1 : 0);
  gpio_set_value(GPIO_21, (rec_buf[0] & 1) ? 1 : 0);
  }

  return len;
}
```

主要實現功能的地方，判斷 rec_buf 內的字元不是\0 及\n 後，將 rec_buf 內的字元進行移位判斷是 1 還是 0 並寫入 gpio 腳位。

Init():

```c
if(gpio_is_valid(GPIO_12) == false){
  pr_err("GPIO %d is not valid\n", GPIO_12);
  goto r_device;
}

if(gpio_is_valid(GPIO_16) == false){
  pr_err("GPIO %d is not valid\n", GPIO_16);
  goto r_device;
}

if(gpio_is_valid(GPIO_20) == false){
  pr_err("GPIO %d is not valid\n", GPIO_20);
  goto r_device;
}

if(gpio_is_valid(GPIO_21) == false){
  pr_err("GPIO %d is not valid\n", GPIO_21);
  goto r_device;
}
```

```c
if(gpio_request(GPIO_12,"GPIO_21") < 0){
  pr_err("ERROR: GPIO %d request\n", GPIO_12);
  goto r_gpio;
}

if(gpio_request(GPIO_16,"GPIO_21") < 0){
  pr_err("ERROR: GPIO %d request\n", GPIO_16);
  goto r_gpio;
}

if(gpio_request(GPIO_20,"GPIO_21") < 0){
  pr_err("ERROR: GPIO %d request\n", GPIO_20);
  goto r_gpio;
}

if(gpio_request(GPIO_21,"GPIO_21") < 0){
  pr_err("ERROR: GPIO %d request\n", GPIO_21);
  goto r_gpio;
}
```

這兩個地方分別是確認 gpio 是否可用及向 gpio 送出 request。這個地方未多作更改，主要是加入新的 gpio。

```
gpio_direction_output(GPIO_12, 0);
gpio_direction_output(GPIO_16, 0);
gpio_direction_output(GPIO_20, 0);
gpio_direction_output(GPIO_21, 0);
```

```
gpio_export(GPIO_12, false);
gpio_export(GPIO_16, false);
gpio_export(GPIO_20, false);
gpio_export(GPIO_21, false);
```

這裡設定 gpio 為輸出，並設定不可改變輸出輸入方向。

Exit():

```
gpio_unexport(GPIO_12);
gpio_unexport(GPIO_16);
gpio_unexport(GPIO_20);
gpio_unexport(GPIO_21);
gpio_free(GPIO_12);
gpio_free(GPIO_16);
gpio_free(GPIO_20);
gpio_free(GPIO_21);
```

釋放 gpio。

In writer.c

Main():

```
int main(int argc, char **argv)
{
    char path[] = "/dev/etx_device";
    char buf[1], clear_sign='0';
    int i = 0, fd = 0;

    fd = open(path, O_WRONLY);
    if (fd == -1) {
        perror("Error opening GPIO pin");
        exit(EXIT_FAILURE);
    }

    while(argv[1][i] != '\0'){
        strncpy(buf, &argv[1][i], 1);
        if (write(fd, buf, sizeof(buf)) == -1) {
            perror("write, set pin output");
            exit(EXIT_FAILURE);
        }
        sleep(1);
        i++;
    }

    write(fd, &clear_sign, 1);

    return 0;
}
```

一開始先設定 driver 的路徑並設定清除所有 led 燈的 clear_sign，之後使用 fd 記憶打開的檔案編號。While 迴圈是主要傳送字元的地方，設定從 argv[1]依序傳送字元並等待 1 秒，最後傳送 clear_sign 清空所有 led 燈。