

微算機系統實習

第 13 組專案報告

LAB 02

組別： 13

班級、姓名與學號：

醫工三 B812110004 葉芸茜

醫工三 B812110011 湯青秀

日期：2024.03.18

一、實驗內容：

1. 寫一個 C/C++ 程式透過輸入參數(argv)控制 GPIO 上的 4 顆 LED 燈狀態
2. 將彩虹排線插入 TX2 J21 腳位，4 顆 LED 燈插入程式對應的 Pin，使用 C/C++ 寫一個程式控制 GPIO 上的 4 個 LED
3. 使用跨平台編譯(在電腦或 Ubuntu 虛擬機編譯出可在嵌入式平台上執行的執行檔，再放入嵌入式平台上執行)
4. 基本題
 - LED[編號] on/off 可以控制該編號的 LED 狀態，其他 LED 狀態不變
 - Ex ./L2Program LED1 on (第一顆燈狀態改為亮，其他狀態不變)
5. 進階題
 - LED 1 & LED 2 為第一組而 LED 3 & LED 4 為第二組
 - 第一組及第二組，兩組分開亮燈/滅燈，一組亮一組滅交替閃爍 5 次

二、實驗過程及結果：

1. 實驗過程

(1) 基本題

(a) 將彩虹排線插入 TX2 J21 腳位



(b) 4 顆 LED 燈插入程式對應的 Pin



彩虹排線對應腳位：

1	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39
2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38	40

本次實驗中使用 4 個腳位為

7(VCC):6(GND)

19(VCC):20(GND)

35(VCC):34(GND)

40(VCC):39(GND)

(c) 撰寫 C++ 程式碼

```
int main(int argc, char *argv[]){
    cout << argv[1][3] << argv[2] << endl;
    if(argv[2][1] == 'n'){
        switch(argv[1][3]){
            case '1':
                gpio_export(396);
                gpio_set_dir(396, "out");
                gpio_set_value(396,1);
                break;
            case '2':
                gpio_export(429);
                gpio_set_dir(429, "out");
                gpio_set_value(429,1);
                break;
            case '3':

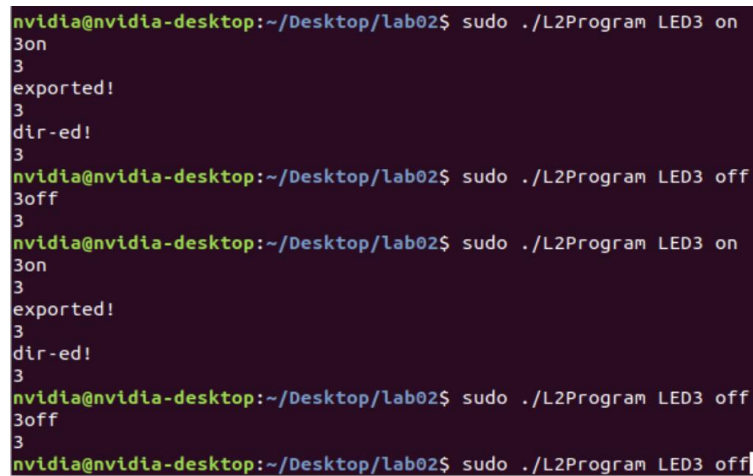
                gpio_export(395);
                gpio_set_dir(395, "out");
                gpio_set_value(395,1);
                break;
            case '4':
                gpio_export(393);
                gpio_set_dir(393, "out");
                gpio_set_value(393,1);
                break;
        }
    }
    else{
```

(d) 編譯程式碼

aarch64-linux-gnu-g++ -std=c++11 -o <輸出的執行檔名稱> <cpp 檔案名稱>

(e) 將執行檔放到跨平台執行

(f) 輸入指令，查看結果



```
nvidia@nvidia-desktop:~/Desktop/lab02$ sudo ./L2Program LED3 on
3on
3
exported!
3
dir-ed!
3
nvidia@nvidia-desktop:~/Desktop/lab02$ sudo ./L2Program LED3 off
3off
3
nvidia@nvidia-desktop:~/Desktop/lab02$ sudo ./L2Program LED3 on
3on
3
exported!
3
dir-ed!
3
nvidia@nvidia-desktop:~/Desktop/lab02$ sudo ./L2Program LED3 off
3off
3
nvidia@nvidia-desktop:~/Desktop/lab02$ sudo ./L2Program LED3 off
```

(2) 進階題

(a) 撰寫 C++ 程式碼

```
using namespace std;

int gpio_export(unsigned int gpio);
int gpio_unexport(unsigned int gpio);
int gpio_set_dir(unsigned int gpio, string dirStatus);
int gpio_set_value(unsigned int gpio, int value);

int main(int argc, char *argv[]) {
    cout << argv[1][3] << argv[2] << endl;
    if(argv[1][0] == 'L'){
        if(argv[2][1] == 'n'){
            switch(argv[1][3]){
                case '1':
                    gpio_export(396);
                    gpio_set_dir(396, "out");
                    gpio_set_value(396,1);
                    break;

                case '2':
                    gpio_export(429);
                    gpio_set_dir(429, "out");
                    gpio_set_value(429,1);
                    break;

                case '3':
                    gpio_export(395);
                    gpio_set_dir(395, "out");
                    gpio_set_value(395,1);
                    break;

                case '4':
                    gpio_export(393);
                    gpio_set_dir(393, "out");
                    gpio_set_value(393,1);
                    break;
            }
        }
        else{
            switch(argv[1][3]){
                case '1':
                    gpio_unexport(393);
                    break;

                case '2':
                    gpio_unexport(395);
                    break;

                case '3':
                    gpio_unexport(396);
                    break;

                case '4':
                    gpio_unexport(429);
                    break;
            }
        }
    }
    for (int i = 0; i < (int)argv[2][0] - 48; i++){
        cout << "i:" << i << "int:" << (int)argv[2][0] << endl;
        if(i % 2 == 0){
            gpio_export(396);
            gpio_set_dir(396, "out");
            gpio_set_value(396,1);
            gpio_export(429);
            gpio_set_dir(429, "out");
            gpio_set_value(429,1);

            gpio_set_value(395, 0);
            gpio_unexport(395);
            gpio_set_value(393, 0);
            gpio_unexport(393);
            sleep(1);
        }
        else{
            gpio_export(395);
            gpio_set_dir(395, "out");
            gpio_set_value(395,1);
            gpio_export(393);
            gpio_set_dir(393, "out");
            gpio_set_value(393,1);

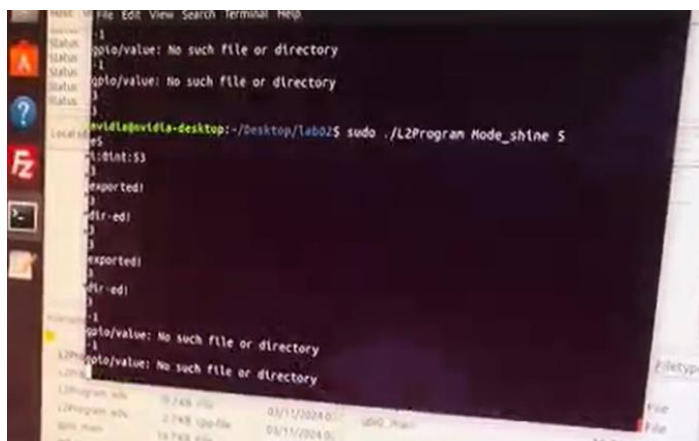
            gpio_set_value(396, 0);
            gpio_unexport(396);
            gpio_set_value(429, 0);
            gpio_unexport(429);
            sleep(1);
        }
    }
}
```

(b) 編譯程式碼

```
nvidia@ubuntu:~/Desktop/lab02$ aarch64-linux-gnu-g++ -o L2Program_adv L2Program_adv.cpp
nvidia@ubuntu:~/Desktop/lab02$ aarch64-linux-gnu-g++ -o L2Program_adv L2Program_adv.cpp
nvidia@ubuntu:~/Desktop/lab02$ aarch64-linux-gnu-g++ -o L2Program_adv L2Program_adv.cpp
nvidia@ubuntu:~/Desktop/lab02$ aarch64-linux-gnu-g++ -o L2Program_adv L2Program_adv.cpp
nvidia@ubuntu:~/Desktop/lab02$ aarch64-linux-gnu-g++ -o L2Program_adv L2Program_adv.cpp
nvidia@ubuntu:~/Desktop/lab02$ aarch64-linux-gnu-g++ -o L2Program_adv L2Program_adv.cpp
nvidia@ubuntu:~/Desktop/lab02$ aarch64-linux-gnu-g++ -o L2Program L2Program.cpp
nvidia@ubuntu:~/Desktop/lab02$
```

(c) 將執行檔放到跨平台執行

(d) 輸入指令，查看結果(詳見影片操作)



2. 預期實驗結果

(1) 執行檔可在跨平台 TX2 順利執行

(2) 程式可根據輸入指令控制特定 LED 燈的開啟及關閉

(3) 基本題中，當輸入指令 ./[執行檔名] [LED 序號] [on/off] 時，可以

控制該編號的 LED 狀態，且其他 LED 狀態不變。

- (4)進階題中，當輸入指令./[執行檔名] Mode_Shine [閃爍次數]時，第一組及第二組 LED 燈將依序交替閃爍數次(分別亮燈及滅燈)

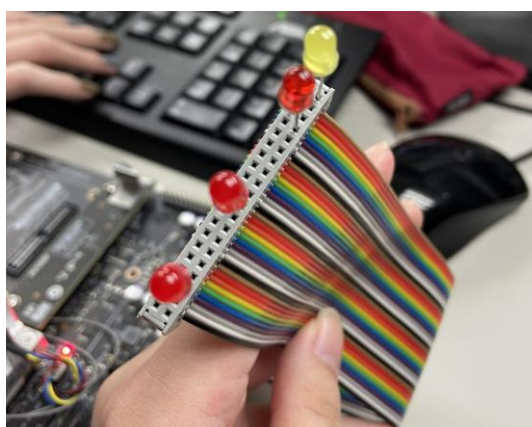
3. 實際上的結果

- (1)基本題 (下方影片也有實際操作畫面)

(a)LED3 亮燈狀態



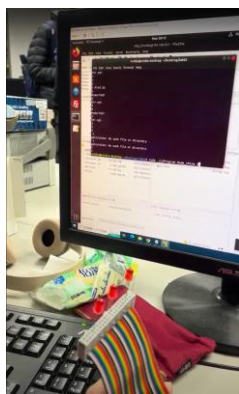
(b)LED3 滅燈狀態



(2)進階題

影片：

<https://drive.google.com/file/d/1dVbzqEne1se8vahRZe82TsEx-b9JPB7v/view?usp=sharing>



4. 遇到的問題&問題怎麼解決

- (1) O_WRONLY 是 0 不是 0，所以一開始一直報錯，後來放大加上網查才發現問題並修改
- (2) main 中抓到的參數 argv 一開始不太知道怎麼切分開來，因為他抓到的是一個字元的陣列。後來發現要用二維陣列去抓到每一個字元的值。還有 switch 不能用字串，只能選擇一個字元
- (3) 輸入進來的是一個字元型態，但我們要計數的必須是整數型態，所以我們把他轉成了整數。可是，直接轉換會轉成字元對應的 ASCII 碼，而不是整數，後來有在 -48 來獲得他的對應整數值
- (4) LED 燈一開始執行時一直不亮，以為是程式問題，但換了其他顆測試發現是能正常亮燈的，後來換了好幾顆才換到沒壞掉的

三、程式碼

目標 1: 基礎題

```
#include <vector>
#include <iostream>
#include <string>
#include <string.h>
#include <stdlib.h>
#include <errno.h>
#include <unistd.h>
#include <fcntl.h>

using namespace std;

int gpio_export(unsigned int gpio);
int gpio_unexport(unsigned int gpio);
int gpio_set_dir(unsigned int gpio, string dirStatus);
int gpio_set_value(unsigned int gpio, int value);

int main(int argc, char *argv[]){
    cout << argv[1][3] << argv[2] << endl;
    if(argv[2][1] == 'n'){
        switch(argv[1][3]){
            case '1':
                gpio_export(396);
                gpio_set_dir(396, "out");
                gpio_set_value(396,1);
                break;
            case '2':
                gpio_export(429);
                gpio_set_dir(429, "out");
                gpio_set_value(429,1);
                break;
            case '3':
```

```

        gpio_export(395);
        gpio_set_dir(395, "out");
        gpio_set_value(395,1);
        break;
    case '4':
        gpio_export(393);
        gpio_set_dir(393, "out");
        gpio_set_value(393,1);
        break;
    }}
    else{
        switch(argv[1][3]){
            case '1':
                gpio_set_value(396, 0);
                gpio_unexport(396);
                break;
            case '2':
                gpio_set_value(429, 0);
                gpio_unexport(429);
                break;
            case '3':
                gpio_set_value(395, 0);
                gpio_unexport(395);
                break;
            case '4':
                gpio_set_value(393, 0);
                gpio_unexport(393);
                break;
        }
    }
    /*
    if(input == 1){
        gpio_export(255);
        gpio_set_dir(255, "out");
        gpio_set_value(255,1);
    }
    else if(input ==2){
        gpio_set_value(255, 0);
        gpio_unexport(255);
    }*/
    return 0;
}

int gpio_set_value(unsigned int gpio, int value){
    int fd;
    char buf[64];

    snprintf(buf, sizeof(buf), "/sys/class/gpio/gpio%d/value",

```

```

gpio);

    fd = open(buf, O_WRONLY);
    cout << fd << endl;
    if(fd < 0){
        perror("gpio/value");
        return fd;
    }

    if(value == 0)
        write(fd, "0", 2);
    else
        write(fd, "1", 2);

    close(fd);
    return 0;
}

int gpio_set_dir(unsigned int gpio, string dirStatus){
    int fd;
    char buf[64];

    snprintf(buf, sizeof(buf),
"/sys/class/gpio/gpio%d/direction", gpio);

    fd = open(buf, O_WRONLY);
    cout << fd << endl;
    if(fd < 0){
        perror("gpio/direction");
        return fd;
    }

    if(dirStatus == "out")
        write(fd, "out", 4);
    else
        write(fd, "in", 3);

    close(fd);
    cout << "dir-ed!"<< endl;
    return 0;
}

int gpio_unexport(unsigned int gpio){
    int fd, len;
    char buf[64];

    fd = open("/sys/class/gpio/unexport", O_WRONLY);
    if(fd < 0){
        perror("gpio/export");
        return fd;
    }

    len = snprintf(buf, sizeof(buf), "%d", gpio);

```



```

        write(fd, buf, len);
        close(fd);
        return 0;
    }

    int gpio_export(unsigned int gpio){
        int fd, len;
        char buf[64];

        fd = open("/sys/class/gpio/export", O_WRONLY);
        cout << fd << endl;
        if(fd <0){
            perror("gpio/export");
            return fd;
        }
        len = snprintf(buf, sizeof(buf), "%d", gpio);
        write(fd, buf, len);
        close(fd);
        cout << "exported!" << endl;
        return 0;
    }
}

```

目標 2: 進階題

```

#include <vector>
#include <iostream>
#include <string>
#include <string.h>
#include <stdlib.h>
#include <errno.h>
#include <unistd.h>
#include <fcntl.h>

using namespace std;

int gpio_export(unsigned int gpio);
int gpio_unexport(unsigned int gpio);
int gpio_set_dir(unsigned int gpio, string dirStatus);
int gpio_set_value(unsigned int gpio, int value);

int main(int argc, char *argv[]){
    cout << argv[1][3] << argv[2] << endl;
    if(argv[1][0] == 'L'){
        if(argv[2][1] == 'n'){
            switch(argv[1][3]){
                case '1':
                    gpio_export(396);
                    gpio_set_dir(396, "out");
                    gpio_set_value(396,1);
                    break;

```

```

        case '2':
            gpio_export(429);
            gpio_set_dir(429, "out");
            gpio_set_value(429,1);
            break;
        case '3':

            gpio_export(395);
            gpio_set_dir(395, "out");
            gpio_set_value(395,1);
            break;
        case '4':
            gpio_export(393);
            gpio_set_dir(393, "out");
            gpio_set_value(393,1);
            break;
    }}
    else{
        switch(argv[1][3]){
            case '1':
                gpio_set_value(396, 0);
                gpio_unexport(396);
                break;
            case '2':
                gpio_set_value(429, 0);
                gpio_unexport(429);
                break;
            case '3':
                gpio_set_value(395, 0);
                gpio_unexport(395);
                break;
            case '4':
                gpio_set_value(393, 0);
                gpio_unexport(393);
                break;
        }
    }

    else{
        for (int i = 0; i < (int)argv[2][0] - 48; i++){
            cout << "i:" << i << "int:" << (int)argv[2][0] <<
endl;

            if(i % 2 == 0){
                gpio_export(396);
                gpio_set_dir(396, "out");
                gpio_set_value(396,1);
                gpio_export(429);
                gpio_set_dir(429, "out");
                gpio_set_value(429,1);

                gpio_set_value(395, 0);

```

```

        gpio_unexport(395);
        gpio_set_value(393, 0);
        gpio_unexport(393);
        sleep(1);
    }
    else{
        gpio_export(395);
        gpio_set_dir(395, "out");
        gpio_set_value(395,1);
        gpio_export(393);
        gpio_set_dir(393, "out");
        gpio_set_value(393,1);

        gpio_set_value(396, 0);
        gpio_unexport(396);
        gpio_set_value(429, 0);
        gpio_unexport(429);
        sleep(1);
    }
}

gpio_set_value(395, 0);
        gpio_unexport(395);
        gpio_set_value(393, 0);
        gpio_unexport(393);
    gpio_set_value(396, 0);
        gpio_unexport(396);
        gpio_set_value(429, 0);
        gpio_unexport(429);
}

    return 0;
}

int gpio_set_value(unsigned int gpio, int value){
    int fd;
    char buf[64];

    snprintf(buf, sizeof(buf), "/sys/class/gpio/gpio%d/value",
gpio);

    fd = open(buf, O_WRONLY);
    cout << fd << endl;
    if(fd < 0){
        perror("gpio/value");
        return fd;
    }

    if(value == 0)
        write(fd, "0", 2);
    else

```

```

        write(fd, "1", 2);

        close(fd);
        return 0;
    }

int gpio_set_dir(unsigned int gpio, string dirStatus){
    int fd;
    char buf[64];

    snprintf(buf, sizeof(buf),
"/sys/class/gpio/gpio%d/direction", gpio);

    fd = open(buf, O_WRONLY);
    cout << fd << endl;
    if(fd < 0){
        perror("gpio/direction");
        return fd;
    }

    if(dirStatus == "out")
        write(fd, "out", 4);
    else
        write(fd, "in", 3);

    close(fd);
    cout << "dir-ed!" << endl;
    return 0;
}

int gpio_unexport(unsigned int gpio){
    int fd, len;
    char buf[64];

    fd = open("/sys/class/gpio/unexport", O_WRONLY);
    if(fd < 0){
        perror("gpio/export");
        return fd;
    }
    len = snprintf(buf, sizeof(buf), "%d", gpio);
    write(fd, buf, len);
    close(fd);
    return 0;
}

int gpio_export(unsigned int gpio){
    int fd, len;
    char buf[64];

    fd = open("/sys/class/gpio/export", O_WRONLY);

```

```

    cout << fd << endl;
    if(fd <0){
        perror("gpio/export");
        return fd;
    }
    len = snprintf(buf, sizeof(buf), "%d", gpio);
    write(fd, buf, len);
    close(fd);
    cout << "exported!" << endl;
    return 0;
}

```

四、本次實驗過程說明與解決方法：

1. 實驗過程

將彩虹排線插入 TX2 腳位 → 將 LED 插入對應 pin 腳 → 撰寫 c++程式碼(基本題&進階題) → 編譯成執行檔 → 傳送到 TX2 並執行 → 輸入指令查看結果

最終成功達成根據不同指令，對 LED 燈進行不同操作

2. 解決方法

在開發過程中，透過修正初始錯誤、適當處理參數、轉換資料型態並解決硬體問題，完成了程式的開發與測試

五、分工：

學號、組員	貢獻比例	工作內容
B812110004 葉芸茜	50%	文書處理、實驗設計與實作、程式規劃、測試與除錯
B812110011 湯青秀	50%	文書處理、實驗設計與實作、程式規劃、測試與除錯