微算機系統實習

第 13 組專案報告

LAB 03

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：2024.04.02

一、實驗內容：

1. lab3_1

   使用 Qt Creator 寫一個 GUI 程式控制 4 個 LED（led 圖示）

   (1) 項目一：控制 LED 開關事件

   (2) 項目二：依據指定閃爍次數控制 LED 開關

2. lab3_2

   使用 Qt Creator 程式控制 GPIO 上的 4 個 LED

   (1) 項目一：控制指定 LED 開關事件

   預設按鈕名稱為 LED Shining

   勾選需要點亮的燈號

   再點擊 LED Shining 按鈕，示意燈泡圖片出現，並點亮 GPIO 對應的 LED 燈，反之，未被勾選的 LED 燈必須熄滅

   (2) 項目二：控制多顆 LED 同時閃爍

   預設閃爍啟動按鈕名稱為 Switching on

   點擊後，間隔閃爍 2 組 LED 燈，且示意燈泡圖片出現

   預設閃爍停止按鈕名稱為 Switching off

   點擊後 LED 熄滅，且示意圖片消失

   (3) 項目三：閃爍速度控制

   根據 spin box 或 slider 控制閃爍速度

   調整區間為 1% - 100%

   製作 slider, spin box, progress bar 且能夠彼此連動

   (4) 項目四：快捷鍵控制

   可利用鍵盤快捷鍵操作以下事件

   可任意選取 LED1 至 LED4

   LED Shining (燈亮)

   Switching on (LED 閃爍)

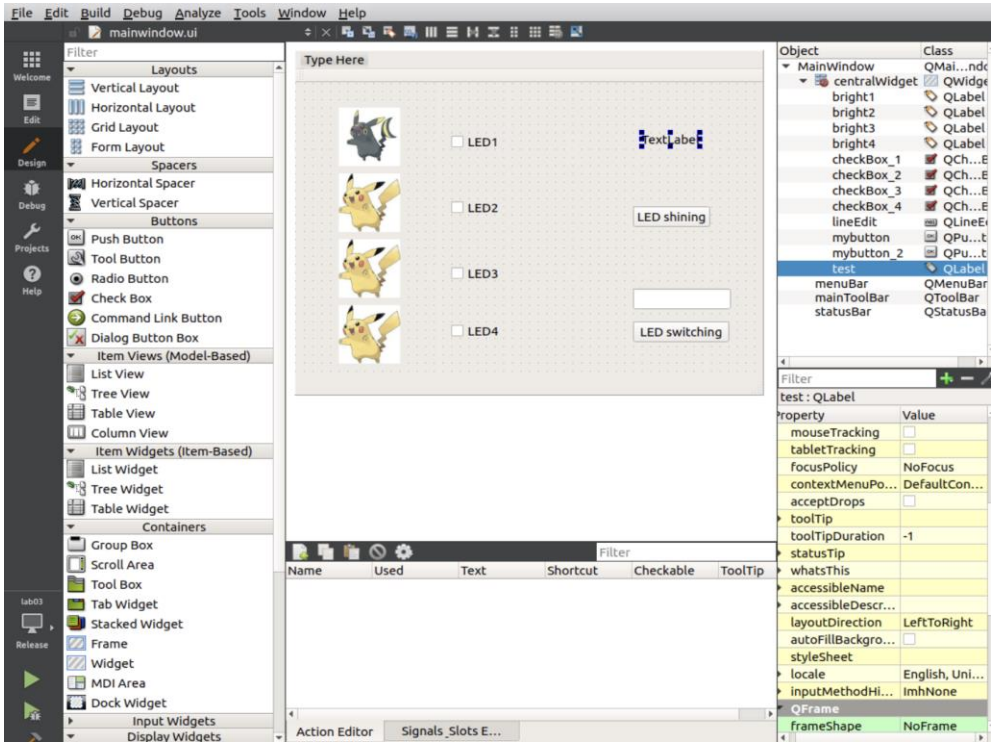
   Switching off (LED 熄滅)

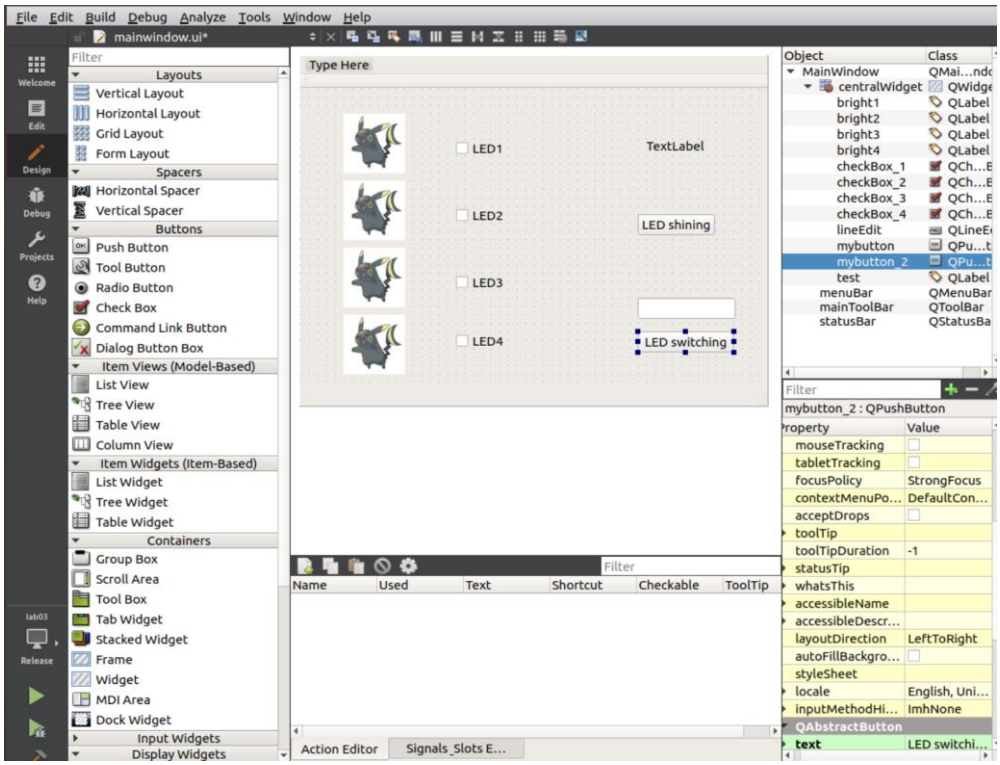
二、實驗過程及結果：

1. 實驗過程

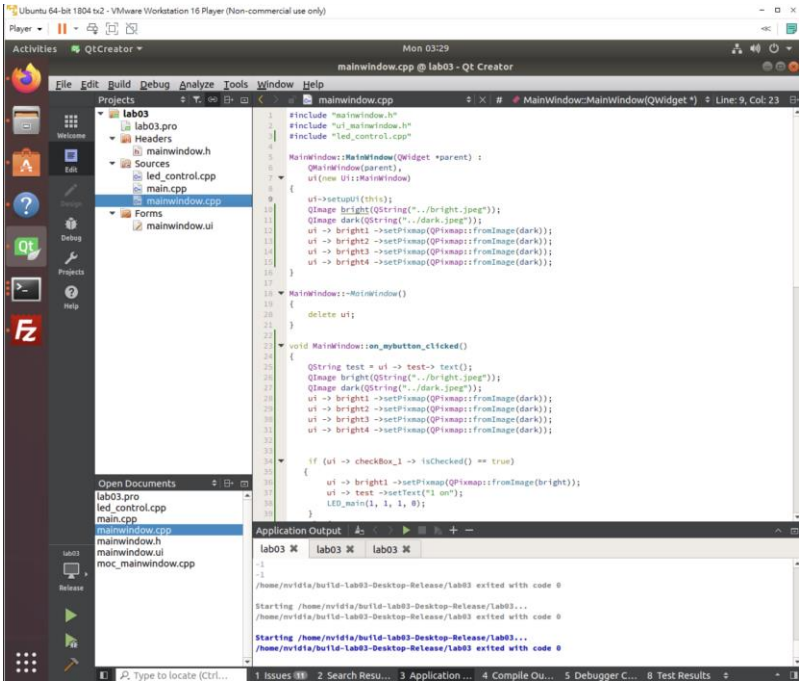
(1) lab3_1

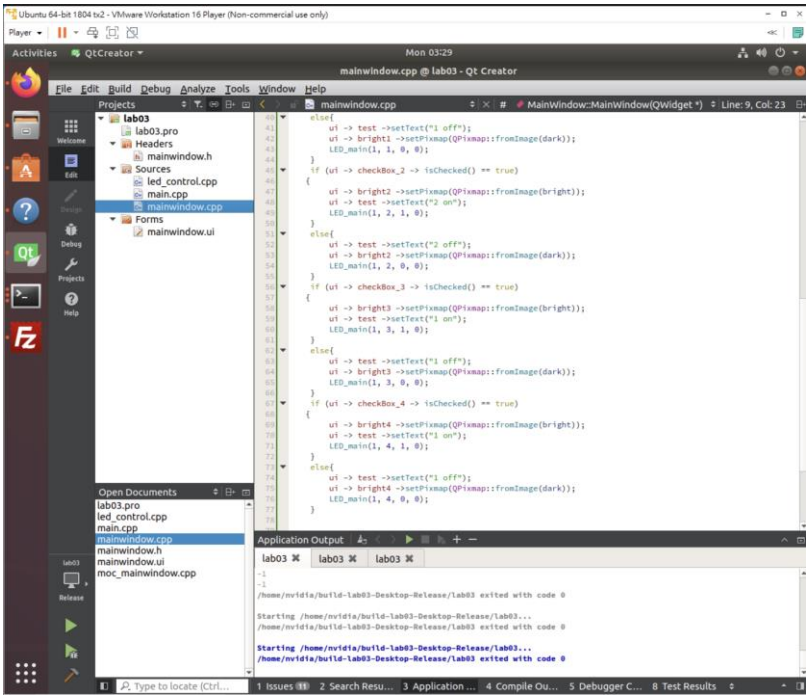
(a) 設計 UI

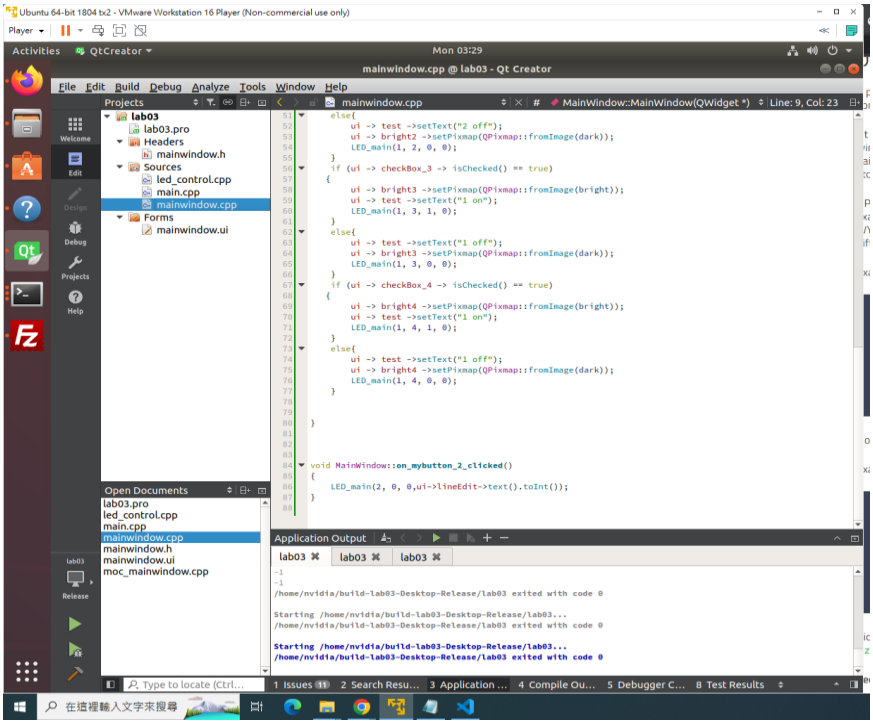




(b) 撰寫程式碼


Ubuntu 64-bit 1804 tx2 - VMware Workstation 16 Player (Non-commercial use only)
Activities
QtCreator
Mon 03:29
mainwindow.cpp @ lab03 - Qt Creator
File Edit Build Debug Analyze Tools Window Help
Projects
lab03
lab03.pro
Headers
mainwindow.h
Sources
led_control.cpp
main.cpp
mainwindow.cpp
Forms
mainwindow.ui
Open Documents
lab03.pro
led_control.cpp
main.cpp
mainwindow.cpp
mainwindow.h
mainwindow.ui
moc_mainwindow.cpp
Application Output
lab03
Starting /home/nvidia/build-lab03-Desktop-Release/lab03...
/home/nvidia/build-lab03-Desktop-Release/lab03 exited with code 0

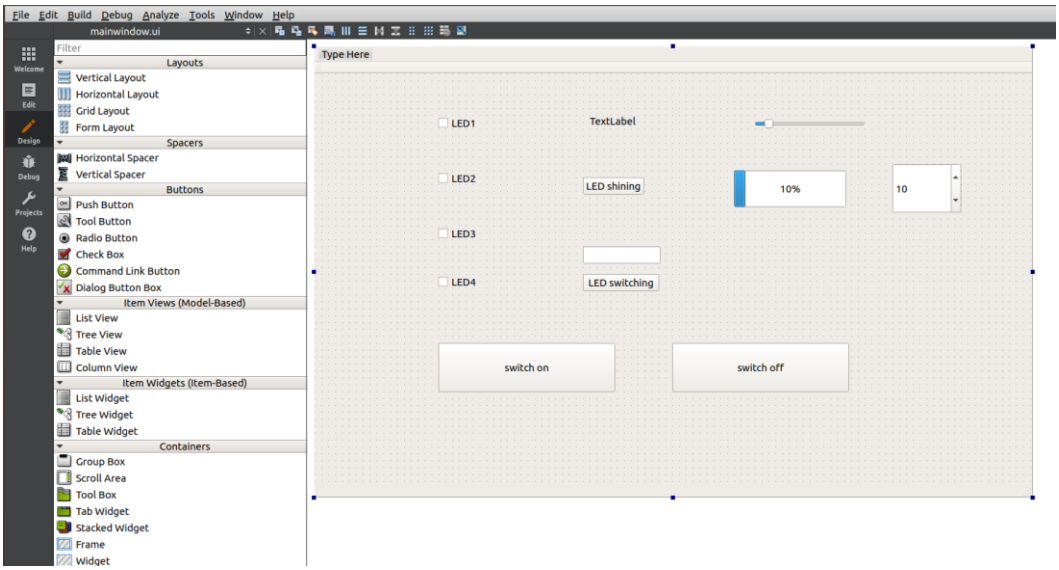

Ubuntu 64-bit 1804 tx2 - VMware Workstation 16 Player (Non-commercial use only)
Activities
QtCreator
Mon 03:29
mainwindow.cpp @ lab03 - Qt Creator
File Edit Build Debug Analyze Tools Window Help
Projects
lab03
lab03.pro
Headers
mainwindow.h
Sources
led_control.cpp
main.cpp
mainwindow.cpp
Forms
mainwindow.ui
Open Documents
lab03.pro
led_control.cpp
main.cpp
mainwindow.cpp
mainwindow.h
mainwindow.ui
moc_mainwindow.cpp
Application Output
lab03
Starting /home/nvidia/build-lab03-Desktop-Release/lab03...
/home/nvidia/build-lab03-Desktop-Release/lab03 exited with code 0



(c) 丟到 TX2 編譯執行

(2) lab3_2

(a) 設計 UI(+閃爍速度控制)



(b) 撰寫程式碼

mainwindow.h*

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <QCheckBox>
#include <QPixmap>
#include <QTimer>



namespace Ui {
class MainWindow;
}

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    explicit MainWindow(QWidget *parent = 0);
    ~MainWindow();

private slots:

    void on_mybutton_clicked();

    void on_mybutton_2_clicked();

    void on_switch_on_clicked();

    void on_switch_off_clicked();

private:
    Ui::MainWindow *ui;
    QTimer *timer;
    QTimer *timer2;
    int counter = 0;
};

#endif // MAINWINDOW_H

```

mainwindow.cpp

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include "led_control.cpp"
#include <iostream>


MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    timer = new QTimer(this);
    timer2 = new QTimer(this);
    connect(timer, SIGNAL(timeout()), this, SLOT(on_mybutton_2_clicked()));
    connect(timer2, SIGNAL(timeout()), this, SLOT(on_switch_on_clicked()));

    ui->setupUi(this);
    QImage bright(QString("../bright.jpeg"));
    QImage dark(QString("../dark.jpeg"));
    ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::on_mybutton_clicked()
{
    QString test = ui -> test-> text();
    QImage bright(QString("../bright.jpeg"));
    QImage dark(QString("../dark.jpeg"));
    ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));


    if (ui -> checkBox_1 -> isChecked() == true)
    {
        ui -> bright1 ->setPixmap(QPixmap::fromImage(bright));
        ui -> test ->setText("1 on");
        LED_main(1, 1);
        LED_main(1, 1);
        ui -> test ->setText(re());
    }
    else{
        ui -> test ->setText("1 off");
        ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
        LED_main(1, 0);
    }
    if (ui -> checkBox_2 -> isChecked() == true)
    {
        ui -> bright2 ->setPixmap(QPixmap::fromImage(bright));
        LED_main(2, 1);
        LED_main(2, 1);
    }
    else{
        ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
        LED_main(2, 0);
    }
    if (ui -> checkBox_3 -> isChecked() == true)
    {
        ui -> bright3 ->setPixmap(QPixmap::fromImage(bright));
        LED_main(3, 1);
        LED_main(3, 1);
    }
    else{
        ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
        LED_main(3, 0);
    }
    if (ui -> checkBox_4 -> isChecked() == true)
    {
        ui -> bright4 ->setPixmap(QPixmap::fromImage(bright));
        LED_main(4, 1);
        LED_main(4, 1);
    }
    else{
        ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
        LED_main(4, 0);
    }


}

void MainWindow::on_mybutton_2_clicked()
{
    QImage bright(QString("../bright.jpeg"));
    QImage dark(QString("../dark.jpeg"));
    ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
    //QString userInput = ui -> lineEdit->text();
    int n = ui->lineEdit->text().toInt();
    QString nn = QString::number(n);
    std::cout << n << std::endl;
    ui -> test ->setText(nn);

    int spinBoxValue = ui->spinBox->value();

    timer->start(1000000 / (spinBoxValue * 100));
        if(counter % 2 == 0){
            std::cout << counter << std::endl;
            //std::cout << re() << std::endl;
            ui -> bright1 ->setPixmap(QPixmap::fromImage(bright));
            ui -> bright2 ->setPixmap(QPixmap::fromImage(bright));
            ui -> checkBox_1 -> setChecked(true);
            ui -> checkBox_2 -> setChecked(true);
            LED_main(1, 1);
            LED_main(2, 1);
            LED_main(1, 1);
            LED_main(2, 1);
            ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
            ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
            ui -> checkBox_3 -> setChecked(false);
            ui -> checkBox_4 -> setChecked(false);
            LED_main(3, 0);
            LED_main(4, 0);
        }
        else{
            std::cout << counter << std::endl;
            ui -> bright3 ->setPixmap(QPixmap::fromImage(bright));
            ui -> bright4 ->setPixmap(QPixmap::fromImage(bright));
            ui -> checkBox_3 -> setChecked(true);
            ui -> checkBox_4 -> setChecked(true);
            LED_main(3, 1);
            LED_main(4, 1);
            LED_main(3, 1);
            LED_main(4, 1);


            LED_main(1, 0);
            LED_main(2, 0);
            ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
            ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
            ui -> checkBox_1 -> setChecked(false);
            ui -> checkBox_2 -> setChecked(false);
        }
        counter++;
        if( counter >= 2*n+1){
            timer->stop();
            counter = 0;
            ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
            ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
            ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
            ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
            ui -> checkBox_1 -> setChecked(false);
            ui -> checkBox_2 -> setChecked(false);
            ui -> checkBox_3 -> setChecked(false);
            ui -> checkBox_4 -> setChecked(false);
            LED_main(1, 0);
            LED_main(2, 0);
            LED_main(3, 0);
            LED_main(4, 0);
        }

    std::cout << "end" << std::endl;
}
```

```
void MainWindow::on_switch_on_clicked()
{
    QImage bright(QString("../bright.jpeg"));
    QImage dark(QString("../dark.jpeg"));
    ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));

    int spinBoxValue = ui->spinBox->value();
    timer2->start(1000000 / (spinBoxValue * 100));

        if(counter % 2 == 0){
            std::cout << counter << std::endl;

            ui -> bright1 ->setPixmap(QPixmap::fromImage(bright));
            ui -> bright2 ->setPixmap(QPixmap::fromImage(bright));
            ui -> checkBox_1 -> setChecked(true);
            ui -> checkBox_2 -> setChecked(true);
            LED_main(1, 1);
            LED_main(2, 1);
            LED_main(1, 1);
            LED_main(2, 1);
            ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
            ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
            ui -> checkBox_3 -> setChecked(false);
            ui -> checkBox_4 -> setChecked(false);
            LED_main(3, 0);
            LED_main(4, 0);
        }
        else{
            std::cout << counter << std::endl;
            ui -> bright3 ->setPixmap(QPixmap::fromImage(bright));
            ui -> bright4 ->setPixmap(QPixmap::fromImage(bright));
            ui -> checkBox_3 -> setChecked(true);
            ui -> checkBox_4 -> setChecked(true);
            LED_main(3, 1);
            LED_main(4, 1);
            LED_main(3, 1);
            LED_main(4, 1);

            ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
            ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
            ui -> checkBox_1 -> setChecked(false);
            ui -> checkBox_2 -> setChecked(false);
            LED_main(1, 0);
            LED_main(2, 0);
       }
       counter++;
}

void MainWindow::on_switch_off_clicked()
{
    counter=0;
    timer2->stop();

    QImage dark(QString("../dark.jpeg"));
    ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
    ui -> checkBox_1 -> setChecked(false);
    ui -> checkBox_2 -> setChecked(false);
    ui -> checkBox_3 -> setChecked(false);
    ui -> checkBox_4 -> setChecked(false);
    LED_main(1, 0);
    LED_main(2, 0);
    LED_main(3, 0);
    LED_main(4, 0);
}
```

(c) 快捷鍵設定

功能	鍵位
LED1 select	1
LED2 select	2
LED3 select	3
LED4 select	4
LED shining	etr
LED switching	w
LED switch on	q
LED switch off	e

(d) 丟到 TX2 編譯執行

2. 預期實驗結果

(1) lab3_1

(a) 項目一：控制 LED 開關事件

透過 GUI 控制特定 LED 圖片為亮暗的狀態，勾選特定幾顆 LED，按下 LED shining 按鍵後，對應的 LED 將由暗的圖片轉為亮的圖片。沒勾選到的 LED 則維持暗的圖片

(b) 項目二：依據指定閃爍次數控制 LED 開關

在輸入框輸入要閃爍的次數後按下 LED switching 按鍵，則會以 LED1、LED2 為第一組，LED3、LED4 為第二組依序交替閃爍(轉換成亮的狀態)，各組分別閃爍一次視為一次，達到輸入的次數後將停止閃爍(恢復暗的圖片狀態)

(2) lab3_2

(a) 項目一：控制指定 LED 開關事件

勾選需要點亮的燈號，再點擊 LED Shining 按鈕，示意燈泡圖片出現，並點亮 GPIO 對應的 LED 燈，反之，未被勾選的 LED 燈必須熄滅

(b) 項目二：控制多顆 LED 同時閃爍

點擊 Switch on 按鈕後，間隔閃爍 2 組 LED 燈，且示意燈泡圖片出現。按下閃爍停止按鈕 Switch off 後 LED 熄滅，且示意圖片消失

(c) 項目三：閃爍速度控制

根據 spin box 或 slider 控制閃爍速度，調整區間為 1% - 100%，slider, spin box, progress bar 之間能彼此連動

(d) 項目四：快捷鍵控制

可利用鍵盤快捷鍵操作以下事件

可透過 1, 2, 3, 4 鍵任意選取 LED1 至 LED4 的勾選

enter 鍵視為 LED shining (燈亮)

W 鍵控制輸入次數之 LED switching (固定次數閃爍)

以 Q 鍵控制 Switch on 啟動(開始閃爍)

以 E 鍵控制 Switch off (LED 熄滅)

3. 實際上的結果

led shining UI 控制

https://drive.google.com/file/d/12K2wPg9sr4qwnDNEYr5GQhWICvm3ddzd/view?usp=drive_link

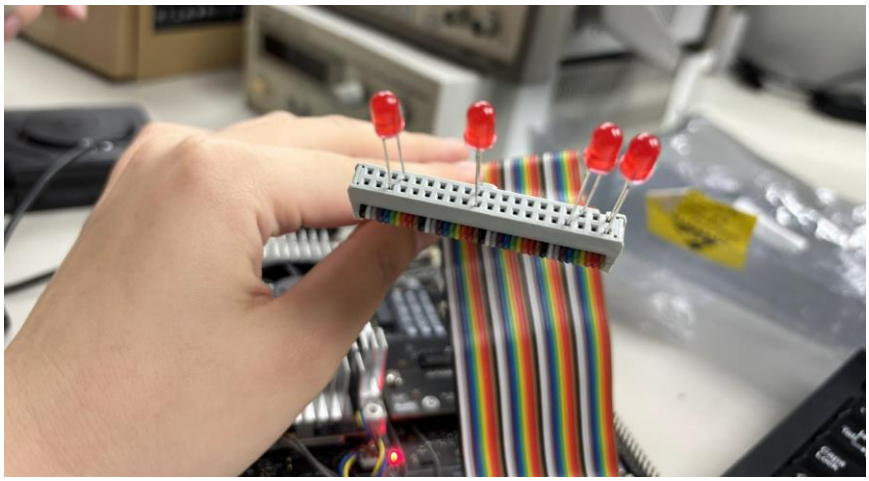
led shining & switching 結合 LED 控制

https://drive.google.com/file/d/14vx8nIvl5D3y4yYuYVCeeEHHpUx7HF9w/view?usp=drive_link

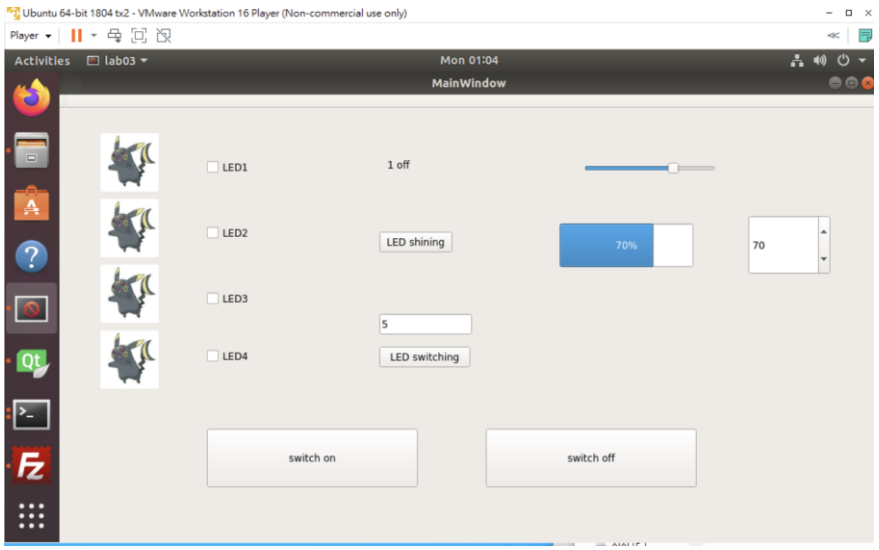
switch on/off & 閃爍速度控制 & 快捷鍵控制

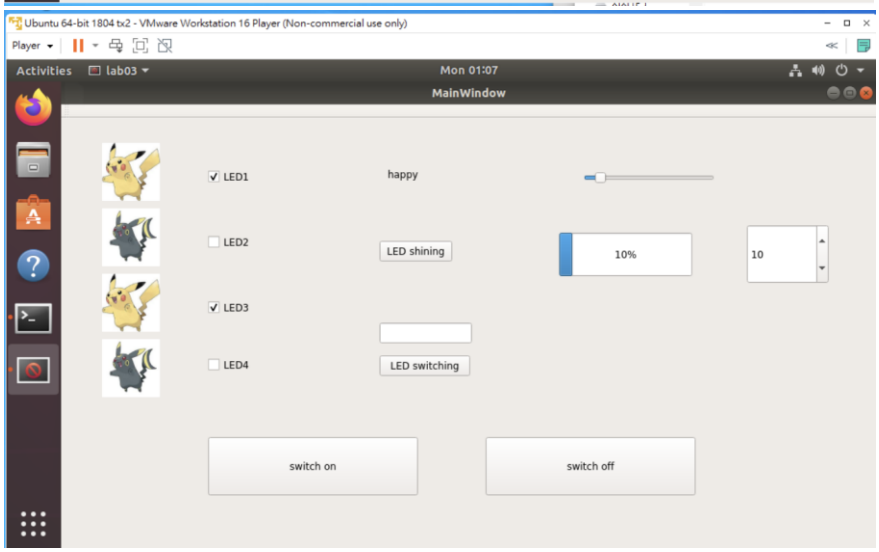
https://drive.google.com/file/d/1SleEQRsAx9ZZkinqtgqUul5l00o8fvdI/view?usp=drive_link

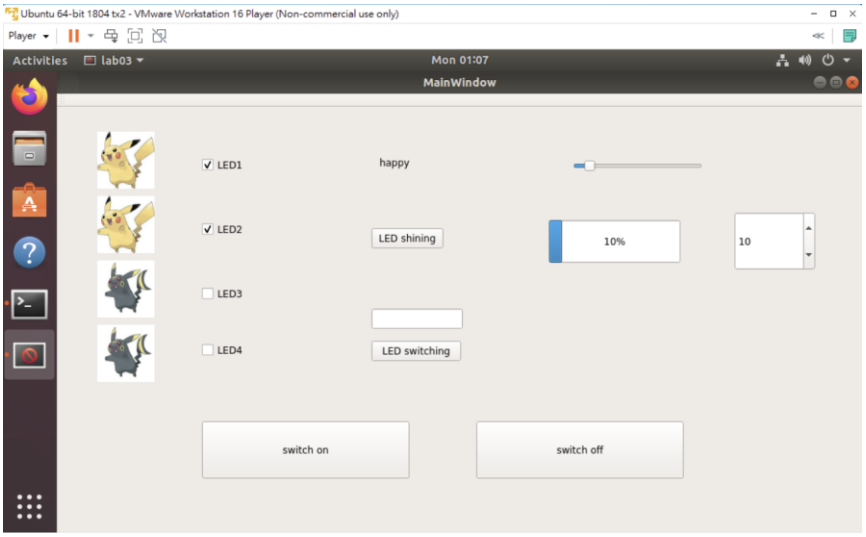
(1) 彩虹排線接腳畫面

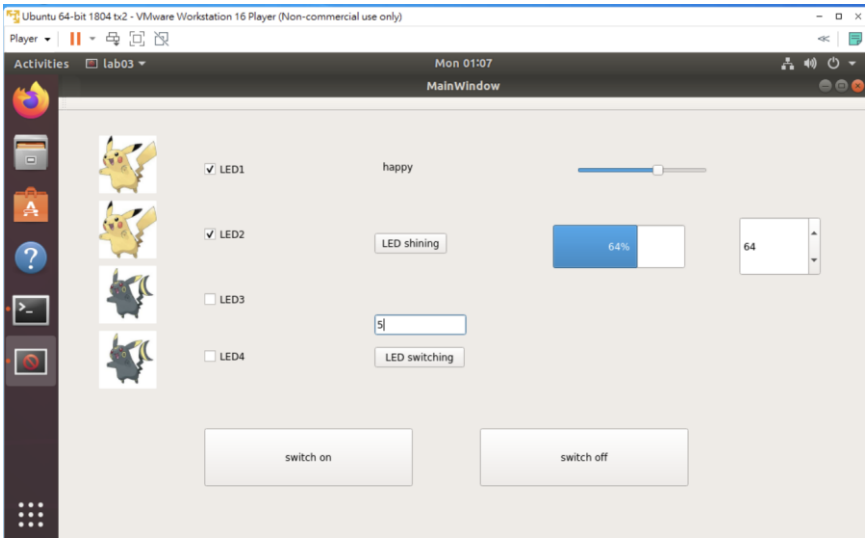


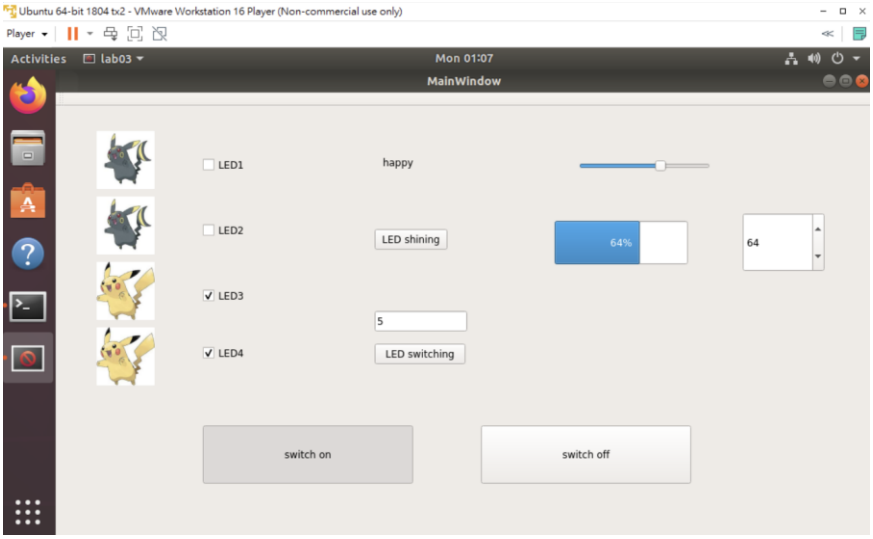
(2) 實際操作頁面變化

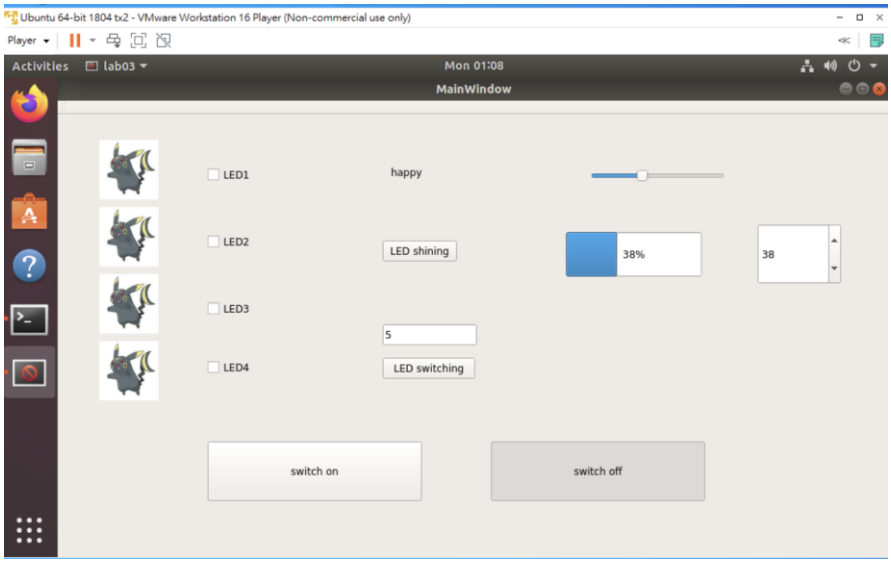





Ubuntu 64-bit 1804 tx2 - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
lab03
Mon 01:07
MainWindow
LED1
LED2
LED3
LED4
happy
LED shining
LED switching
10%
10
switch on
switch off


Ubuntu 64-bit 1804 tx2 - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
lab03
Mon 01:07
MainWindow
LED1
LED2
LED3
LED4
happy
LED shining
LED switching
64%
64
5
switch on
switch off


Ubuntu 64-bit 1804 tx2 - VMware Workstation 16 Player (Non-commercial use only)
Player
Activities
lab03
Mon 01:07
MainWindow
LED1
LED2
LED3
LED4
happy
LED shining
LED switching
64%
64
5
switch on
switch off



4. 遇到的問題&問題怎麼解決

(1) 圖片無法輸入顯示:

需要放在專案資料夾外面，並在程式中用路徑的方式輸入

(2) 圖片無法在閃爍時無法顯示改變:

在 switching 按鈕的 click 函數也得先初始化圖片

(3) led 一直不能亮:

權限問題，使用 sudo 就可以了

(4) timer 控制失敗:

用了兩個 timer，卻忘記分別宣告變數。另外 timer 的 start 要寫在 function 外

(5) 呼叫 LED 控制 function 卻沒有執行:

程式碼的資料型態傳遞錯誤，導致他無法判別 switch 條件

三、程式碼

mainwindow.h

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <QCheckBox>
#include <QPixmap>
#include <QTimer>

namespace Ui {
class MainWindow;
}
```

```
class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    explicit MainWindow(QWidget *parent = 0);
    ~MainWindow();

private slots:

    void on_mybutton_clicked();

    void on_mybutton_2_clicked();

    void on_switch_on_clicked();

    void on_switch_off_clicked();

private:
    Ui::MainWindow *ui;
    QTimer *timer;
    int counter = 0;
};

#endif // MAINWINDOW_H
```

mainwindow.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>1043</width>
    <height>652</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>MainWindow</string>
  </property>
  <widget class="QWidget" name="centralWidget">
   <widget class="QPushButton" name="mybutton">
    <property name="geometry">
     <rect>
```

```
   <x>390</x>
   <y>150</y>
   <width>89</width>
   <height>25</height>
  </rect>
 </property>
 <property name="text">
  <string>LED shining</string>
 </property>
</widget>
<widget class="QLabel" name="bright1">
 <property name="geometry">
  <rect>
   <x>50</x>
   <y>30</y>
   <width>71</width>
   <height>71</height>
  </rect>
 </property>
 <property name="text">
  <string/>
 </property>
 <property name="pixmap">
  <pixmap>../Downloads/dark.jpeg</pixmap>
 </property>
 <property name="scaledContents">
  <bool>true</bool>
 </property>
</widget>
<widget class="QCheckBox" name="checkBox_2">
 <property name="geometry">
  <rect>
   <x>180</x>
   <y>140</y>
   <width>158</width>
   <height>23</height>
  </rect>
 </property>
 <property name="text">
  <string>LED2</string>
 </property>
</widget>
<widget class="QCheckBox" name="checkBox_1">
 <property name="geometry">
  <rect>
   <x>180</x>
   <y>60</y>
   <width>158</width>
   <height>23</height>
  </rect>
```

```
 </property>
 <property name="text">
  <string>LED1</string>
 </property>
</widget>
<widget class="QCheckBox" name="checkBox_3">
 <property name="geometry">
  <rect>
   <x>180</x>
   <y>220</y>
   <width>158</width>
   <height>23</height>
  </rect>
 </property>
 <property name="text">
  <string>LED3</string>
 </property>
</widget>
<widget class="QCheckBox" name="checkBox_4">
 <property name="geometry">
  <rect>
   <x>180</x>
   <y>290</y>
   <width>158</width>
   <height>23</height>
  </rect>
 </property>
 <property name="text">
  <string>LED4</string>
 </property>
</widget>
<widget class="QLabel" name="bright2">
 <property name="geometry">
  <rect>
   <x>50</x>
   <y>110</y>
   <width>71</width>
   <height>71</height>
  </rect>
 </property>
 <property name="text">
  <string/>
 </property>
 <property name="pixmap">
  <pixmap>../Downloads/bright.jpeg</pixmap>
 </property>
 <property name="scaledContents">
  <bool>true</bool>
 </property>
</widget>
```

```
<widget class="QLabel" name="bright3">
 <property name="geometry">
  <rect>
   <x>50</x>
   <y>190</y>
   <width>71</width>
   <height>71</height>
  </rect>
 </property>
 <property name="text">
  <string/>
 </property>
 <property name="pixmap">
  <pixmap>../Downloads/bright.jpeg</pixmap>
 </property>
 <property name="scaledContents">
  <bool>true</bool>
 </property>
</widget>
<widget class="QLabel" name="bright4">
 <property name="geometry">
  <rect>
   <x>50</x>
   <y>270</y>
   <width>71</width>
   <height>71</height>
  </rect>
 </property>
 <property name="text">
  <string/>
 </property>
 <property name="pixmap">
  <pixmap>../Downloads/bright.jpeg</pixmap>
 </property>
 <property name="scaledContents">
  <bool>true</bool>
 </property>
</widget>
<widget class="QLabel" name="test">
 <property name="geometry">
  <rect>
   <x>400</x>
   <y>60</y>
   <width>67</width>
   <height>17</height>
  </rect>
 </property>
 <property name="text">
  <string>TextLabel</string>
 </property>
```

```
 </widget>
 <widget class="QPushButton" name="mybutton_2">
  <property name="geometry">
   <rect>
    <x>390</x>
    <y>290</y>
    <width>111</width>
    <height>25</height>
   </rect>
  </property>
  <property name="text">
   <string>LED switching</string>
  </property>
 </widget>
 <widget class="QLineEdit" name="lineEdit">
  <property name="geometry">
   <rect>
    <x>390</x>
    <y>250</y>
    <width>113</width>
    <height>25</height>
   </rect>
  </property>
 </widget>
 <widget class="QSlider" name="horizontalSlider">
  <property name="geometry">
   <rect>
    <x>640</x>
    <y>50</y>
    <width>160</width>
    <height>45</height>
   </rect>
  </property>
  <property name="minimum">
   <number>1</number>
  </property>
  <property name="maximum">
   <number>100</number>
  </property>
  <property name="value">
   <number>10</number>
  </property>
  <property name="orientation">
   <enum>Qt::Horizontal</enum>
  </property>
 </widget>
 <widget class="QProgressBar" name="progressBar">
  <property name="geometry">
   <rect>
    <x>610</x>
```

```
    <y>140</y>
    <width>163</width>
    <height>53</height>
   </rect>
  </property>
  <property name="minimum">
   <number>0</number>
  </property>
  <property name="value">
   <number>10</number>
  </property>
 </widget>
 <widget class="QSpinBox" name="spinBox">
  <property name="geometry">
   <rect>
    <x>840</x>
    <y>130</y>
    <width>100</width>
    <height>72</height>
   </rect>
  </property>
  <property name="minimum">
   <number>1</number>
  </property>
  <property name="maximum">
   <number>100</number>
  </property>
  <property name="value">
   <number>10</number>
  </property>
 </widget>
 <widget class="QPushButton" name="switch_on">
  <property name="geometry">
   <rect>
    <x>180</x>
    <y>390</y>
    <width>257</width>
    <height>71</height>
   </rect>
  </property>
  <property name="text">
   <string>switch on</string>
  </property>
 </widget>
 <widget class="QPushButton" name="switch_off">
  <property name="geometry">
   <rect>
    <x>520</x>
    <y>390</y>
    <width>257</width>
```

```
     <height>71</height>
    </rect>
   </property>
   <property name="text">
    <string>switch off</string>
   </property>
  </widget>
 </widget>
 <widget class="QMenuBar" name="menuBar">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>1043</width>
    <height>56</height>
   </rect>
  </property>
 </widget>
 <widget class="QToolBar" name="mainToolBar">
  <attribute name="toolBarArea">
   <enum>TopToolBarArea</enum>
  </attribute>
  <attribute name="toolBarBreak">
   <bool>false</bool>
  </attribute>
 </widget>
 <widget class="QStatusBar" name="statusBar"/>
</widget>
<layoutdefault spacing="6" margin="11"/>
<resources/>
<connections>
 <connection>
  <sender>horizontalSlider</sender>
  <signal>valueChanged(int)</signal>
  <receiver>progressBar</receiver>
  <slot>setValue(int)</slot>
  <hints>
   <hint type="sourcelabel">
    <x>739</x>
    <y>176</y>
   </hint>
   <hint type="destinationlabel">
    <x>717</x>
    <y>284</y>
   </hint>
  </hints>
 </connection>
 <connection>
  <sender>spinBox</sender>
  <signal>valueChanged(int)</signal>
```

```
   <receiver>horizontalSlider</receiver>
   <slot>setValue(int)</slot>
   <hints>
    <hint type="sourcelabel">
     <x>876</x>
     <y>271</y>
    </hint>
    <hint type="destinationlabel">
     <x>739</x>
     <y>178</y>
    </hint>
   </hints>
  </connection>
  <connection>
   <sender>progressBar</sender>
   <signal>valueChanged(int)</signal>
   <receiver>spinBox</receiver>
   <slot>setValue(int)</slot>
   <hints>
    <hint type="sourcelabel">
     <x>646</x>
     <y>275</y>
    </hint>
    <hint type="destinationlabel">
     <x>910</x>
     <y>299</y>
    </hint>
   </hints>
  </connection>
 </connections>
</ui>
```

led_control.cpp

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

using namespace std;

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

```
QString re();

QString re(){
   return "happy";
}


void LED_main(int LED, int state){
   if(state == 1){
      switch(LED){
      case 1:
         //cout << "1 bright" << endl;
         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(LED){
      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;
}
    }

}


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;
}
```

main.cpp

```
#include "mainwindow.h"
#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;
    w.show();

    return a.exec();
}
```

mainwindow.cpp

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
```

```
#include "led_control.cpp"
#include <iostream>


MainWindow::MainWindow(QWidget *parent) :
   QMainWindow(parent),
   ui(new Ui::MainWindow)
{
   timer = new QTimer(this);
   //connect(timer, SIGNAL(timeout()), this, SLOT(on_mybutton_2_clicked()));
   connect(timer, SIGNAL(timeout()), this, SLOT(on_switch_on_clicked()));

   ui->setupUi(this);
   QImage bright(QString("../bright.jpeg"));
   QImage dark(QString("../dark.jpeg"));
   ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
   ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
   ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
   ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
}

MainWindow::~MainWindow()
{
   delete ui;
}

void MainWindow::on_mybutton_clicked()
{
   QString test = ui -> test-> text();
   QImage bright(QString("../bright.jpeg"));
   QImage dark(QString("../dark.jpeg"));
   ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
   ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
   ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
   ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));


   if (ui -> checkBox_1 -> isChecked() == true)
  {
      ui -> bright1 ->setPixmap(QPixmap::fromImage(bright));
      ui -> test ->setText("1 on");
      LED_main(1, 1);
      LED_main(1, 1);
      ui -> test ->setText(re());
   }
   else{
      ui -> test ->setText("1 off");
      ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
      LED_main(1, 0);
   }
```

```
    if (ui -> checkBox_2 -> isChecked() == true)
    {
        ui -> bright2 ->setPixmap(QPixmap::fromImage(bright));
        LED_main(2, 1);
        LED_main(2, 1);
    }
    else{
        ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
        LED_main(2, 0);
    }
    if (ui -> checkBox_3 -> isChecked() == true)
    {
        ui -> bright3 ->setPixmap(QPixmap::fromImage(bright));
        LED_main(3, 1);
        LED_main(3, 1);
    }
    else{
        ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
        LED_main(3, 0);
    }
    if (ui -> checkBox_4 -> isChecked() == true)
    {
        ui -> bright4 ->setPixmap(QPixmap::fromImage(bright));
        LED_main(4, 1);
        LED_main(4, 1);
    }
    else{
        ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
        LED_main(4, 0);
    }


}



void MainWindow::on_mybutton_2_clicked()
{
    QImage bright(QString("../bright.jpeg"));
    QImage dark(QString("../dark.jpeg"));
    ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
    ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
    //QString userInput = ui -> lineEdit->text();
    int n = ui->lineEdit->text().toInt();
    QString nn = QString::number(n);
    std::cout << n << std::endl;
    ui -> test ->setText(nn);
```

```
int spinBoxValue = ui->spinBox->value();

timer->start(1000000 / (spinBoxValue * 100));
   if(counter % 2 == 0){
      std::cout << counter << std::endl;
      //std::cout << re() << std::endl;
      ui -> bright1 ->setPixmap(QPixmap::fromImage(bright));
      ui -> bright2 ->setPixmap(QPixmap::fromImage(bright));
      ui -> checkBox_1 -> setChecked(true);
      ui -> checkBox_2 -> setChecked(true);
      LED_main(1, 1);
      LED_main(2, 1);
      LED_main(1, 1);
      LED_main(2, 1);
      ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
      ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
      ui -> checkBox_3 -> setChecked(false);
      ui -> checkBox_4 -> setChecked(false);
      LED_main(3, 0);
      LED_main(4, 0);
   }
   else{
      std::cout << counter << std::endl;
      ui -> bright3 ->setPixmap(QPixmap::fromImage(bright));
      ui -> bright4 ->setPixmap(QPixmap::fromImage(bright));
      ui -> checkBox_3 -> setChecked(true);
      ui -> checkBox_4 -> setChecked(true);
      LED_main(3, 1);
      LED_main(4, 1);
      LED_main(3, 1);
      LED_main(4, 1);


      LED_main(1, 0);
      LED_main(2, 0);
      ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
      ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
      ui -> checkBox_1 -> setChecked(false);
      ui -> checkBox_2 -> setChecked(false);
   }
   counter++;
   if( counter >= 2*n+1){
      timer->stop();
      counter = 0;
      ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
      ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
      ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
      ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
      ui -> checkBox_1 -> setChecked(false);
      ui -> checkBox_2 -> setChecked(false);
```

```
        ui -> checkBox_3 -> setChecked(false);
        ui -> checkBox_4 -> setChecked(false);
        LED_main(1, 0);
        LED_main(2, 0);
        LED_main(3, 0);
        LED_main(4, 0);
      }

   std::cout << "end" << std::endl;
}


void MainWindow::on_switch_on_clicked()
{
   QImage bright(QString("../bright.jpeg"));
   QImage dark(QString("../dark.jpeg"));
   ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
   ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
   ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
   ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));

   int spinBoxValue = ui->spinBox->value();
   timer->start(1000000 / (spinBoxValue * 100));

      if(counter % 2 == 0){
         std::cout << counter << std::endl;

         ui -> bright1 ->setPixmap(QPixmap::fromImage(bright));
         ui -> bright2 ->setPixmap(QPixmap::fromImage(bright));
         ui -> checkBox_1 -> setChecked(true);
         ui -> checkBox_2 -> setChecked(true);
         LED_main(1, 1);
         LED_main(2, 1);
         LED_main(1, 1);
         LED_main(2, 1);
         ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
         ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
         ui -> checkBox_3 -> setChecked(false);
         ui -> checkBox_4 -> setChecked(false);
         LED_main(3, 0);
         LED_main(4, 0);
      }
      else{
         std::cout << counter << std::endl;
         ui -> bright3 ->setPixmap(QPixmap::fromImage(bright));
         ui -> bright4 ->setPixmap(QPixmap::fromImage(bright));
         ui -> checkBox_3 -> setChecked(true);
         ui -> checkBox_4 -> setChecked(true);
         LED_main(3, 1);
         LED_main(4, 1);
```

```
        LED_main(3, 1);
        LED_main(4, 1);

        ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
        ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
        ui -> checkBox_1 -> setChecked(false);
        ui -> checkBox_2 -> setChecked(false);
        LED_main(1, 0);
        LED_main(2, 0);
    }
    counter++;
}

void MainWindow::on_switch_off_clicked()
{
  counter=0;
  timer->stop();

  QImage dark(QString("../dark.jpeg"));
  ui -> bright1 ->setPixmap(QPixmap::fromImage(dark));
  ui -> bright2 ->setPixmap(QPixmap::fromImage(dark));
  ui -> bright3 ->setPixmap(QPixmap::fromImage(dark));
  ui -> bright4 ->setPixmap(QPixmap::fromImage(dark));
  ui -> checkBox_1 -> setChecked(false);
  ui -> checkBox_2 -> setChecked(false);
  ui -> checkBox_3 -> setChecked(false);
  ui -> checkBox_4 -> setChecked(false);
  LED_main(1, 0);
  LED_main(2, 0);
  LED_main(3, 0);
  LED_main(4, 0);
}
```

四、本次實驗過程說明與解決方法:

1. 實驗過程

   設計 UI → 撰寫並依功能改進 c++程式碼→ 傳送到 TX2 編譯成執行檔 → 執行 → 測試功能是否滿足實驗要求

   最終成功達成根據不同操作，使 LED 燈完成指定動作

2. 解決方法

   遇到的問題涵蓋了圖片輸入顯示、閃爍時的圖片變化、LED 亮度控制、計時器失效以及功能呼叫未執行等方面。解決方法包括適當管理圖片路徑、確保圖片初始化正確、解決 LED 控制的權限問題、修正計時器的使用方式以及檢查資料型態傳遞以確保功能正確執行。

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