Arduino 课题研究记录——红外摇控灯

17307110259　黄文卓　物理学系

一、课题设计

Arduino 是一款便捷灵活、方便上手的开源电子原型平台，它构建于开放原始码 simple I/O 介面版，并且具有使用类似 Java、C 语言的 Processing/Wiring 开发环境。Arduino 能通过各种各样的传感器来感知环境，通过控制灯光、马达和其他的装置来反馈、影响环境[1]。本课题采用红外摇控模块，实现对三色 LED 灯的各种控制功能，包括开关、颜色选择、亮度调节、三色闪烁等功能，同时还加入光敏电阻模块，实现 LED 亮度随外界亮度变化的功能；加入液晶模块，使操作状态实时呈现。

二、实验过程

1.实验器材：

ArduinoUNO R3 开发板

MB-102 面包板

LCD1602A 液晶显示器

KY-018 光敏电阻模块

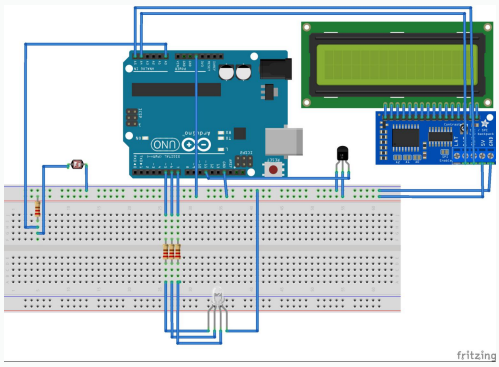
KY-005 红外发射传感器模块

KY-016 全彩 LED 灯

KY-022 红外传感器接收模块

摇控器（右图所示）、电阻、面包线若干、杜邦线若干

2.实验电路接线图（电源采用笔记本电脑电源，图中未画出）



3.代码思路（具体代码见附录）

1）引入红外模块和液晶模块用到的库：IRremote.h、Wire.h、

LiquidCrystal_I2C.h

2）定义各针脚，初始化红外接收器、液晶，关闭所有灯。红外信号接收的实现分两步，一是用库函数 irrecv.decode(&results)接收信号，将信号的编码储存在 results 中，然后跟据 results 的具体编码执行相应操作；二是用 irrecv.resume()函数重置 results，以接收下一个信号。代码的结构像树形图一样，每个节点处的选择都会进入不同的分支。

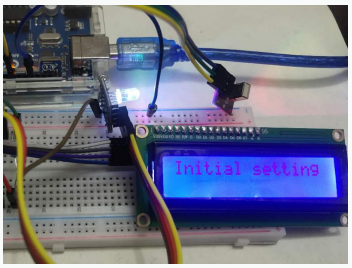
4）初始化设置后第一个选择：开灯或关灯，分别用摇控器上的向左和向右键对应。

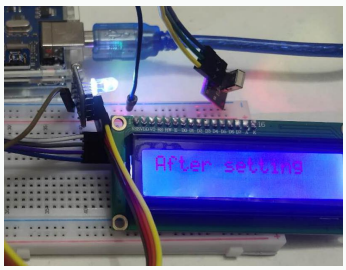
5）开灯操作后进入 5 个选择，这 5 个功能和对应摇控器的按键分别为：总亮度调节（0）、纯红色亮度调节（1）、纯绿色亮度调节（2）、纯蓝色亮度调节（3）、三色闪烁（*）、光敏电阻调节（#）

6）进入亮度调节后可用向上和向下键分别调节亮度

7）每个功能的代码放在 While（1）死循环中，方便随时接收信号。在完成对应功能的操作后，按 OK 键跳出循环，进入上一层的分支选择。

三、实验结果

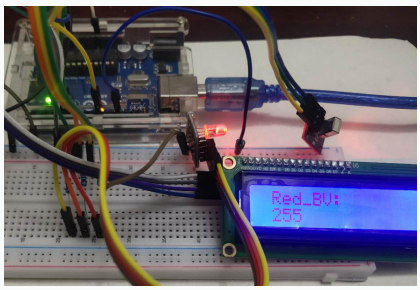


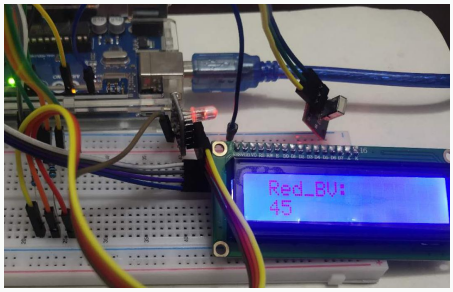


按向左键进入初始设置（如上左图），此时按向右键可关灯。再按一次进入开灯后操作，接着可能选择相应功能进行操作。

1.亮度调节（以纯红色为例）

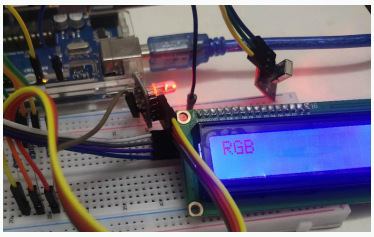
按 1 号键进入纯红色亮度调节，在此功能中按向上向下键可分别提高、降低灯的亮度：

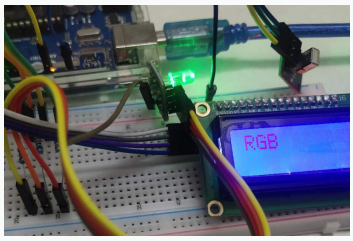


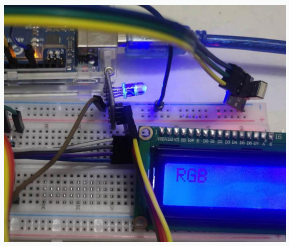


2. RGB 三色闪烁

按*号键进入三色闪烁功能，LED 的颜色在红绿蓝之间变换，闪烁间隔设为 1s。

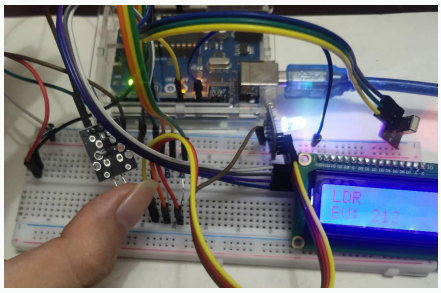


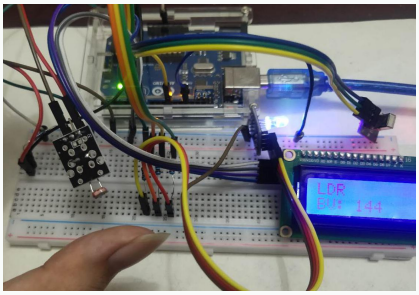




3. 光敏电阻亮度调节

按#号键进入光敏电阻亮度调节功能，此时改变光敏电阻受到的光照强度，LED 的亮度也会随之改变。变化规律设为光照越强亮度越低，光照越弱亮度越高。

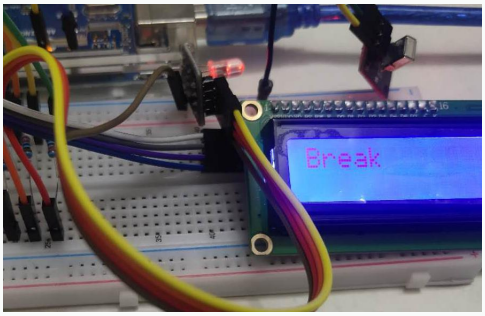


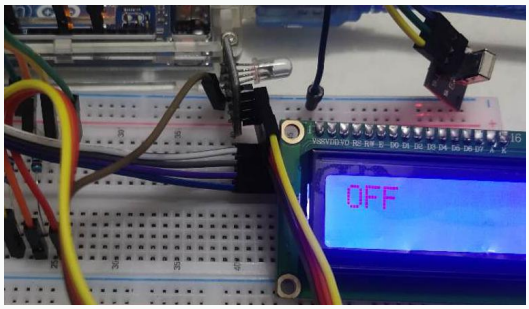


可以看到遮光时的 LED 明显更亮一些，而且亮度的数值也比不遮光时高。

4. 跳出和关灯

按 OK 键依次跳出循环，到初始设置部分时按向右键关灯





四、简要分析

本实验通过各种模块的组合，实现了红外摇控 LED 的各种功能，但也存在一些小问题。首先是器材上的问题，摇控器的灵敏度和按键质量都不够好，经常出现无法读取的情况，按下按键的过程要非常短促才不会得到多个编码。其次是代

码编写中出现的问题，死循环中放置相应功能的代码，代码的前后分别是接收信号和重置信号的库函数。如果在代码执行中按下按键就会被重置函数清除信号，无法进到下一循环激活功能。解决办法是在重置函数的代码后面加入 1s 的延迟，在这 1s 的延迟中收到的信号就能顺利进入下一循环，而代码执行的过程远小于 1s，这也使得接收的信号总能成功激活功能。

五、参考文献

[1]arduino，百度百科

附录——代码一览

```
#include <IRremote.h>
#include "Wire.h"
#include "LiquidCrystal_I2C.h"

//定义11针脚接收信息
int RECV_PIN = 11;
//定义5,6,7数字针脚为LED各颜色输出
int led_red = 5;
int led_green = 6;
int led_blue = 7;
//定义光敏电阻接口为A0号接口
int potpin = A0;

IRrecv irrecv(RECV_PIN);

decode_results results;
//默认LED不亮
int val = 0;

//设置LCD
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
  Serial.begin(9600);
  //初始化接收器
  irrecv.enableIRIn();
  //设定数字IO口的模式，OUTPUT 为输出
  pinMode(led_red, OUTPUT);
  pinMode(led_green, OUTPUT);
  pinMode(led_blue, OUTPUT);

  //关闭所有灯
  analogWrite(led_red, 0);
  analogWrite(led_green, 0);
  analogWrite(led_blue, 0);
  // 延时等待系统初始化
  delay(1000);
  // 初始化 LCD
  lcd.init();
  // 打开屏幕背光
  lcd.backlight();
}
```

```
void loop() {

  if (irrecv.decode(&results)) {
    Serial.println("初始设置: ");
    Serial.println(results.value, HEX);
    //按"向左"键进入开灯
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Initial setting");
    if (results.value == 0xFF10EF) {
      digitalWrite(led_red, HIGH);
      digitalWrite(led_green, HIGH);
      digitalWrite(led_blue, HIGH);
      val = 255;
      Serial.println("开灯");
      irrecv.resume();
      delay(1000);
      while (1) {
        //开灯之后的操作
        if (irrecv.decode(&results)) {
          Serial.println("开灯后操作: ");
          Serial.println(results.value, HEX);
          lcd.clear();
          lcd.setCursor(0, 0);
          lcd.print("After setting");
          //一、亮度调节
          if (results.value == 0xFF9867) { //按下0号键进入总的亮度调节
            brightAll();
          } else if (results.value == 0xFFA25D) { //按下1号键进入纯红色的亮度调节
            brightRed();
          } else if (results.value == 0xFF629D) { //按下2号键进行纯绿色的亮度调节
            brightGre();
          } else if (results.value == 0xFFE21D) { //按下3号键进行纯蓝色的亮度调节
            brightBlu();
          } else if (results.value == 0xFF6897) { //按*键进入三色交替闪烁
            rgb();
          } else if (results.value == 0xFFB04F) { //按#号键进入光敏电阻功能
            lightR();
          } else if (results.value == 0xFF38C7) {
            Serial.println("跳出开灯操作");
            lcd.clear();
```

```
            lcd.setCursor(0, 0);
            lcd.print("After break");
            break;
          }//跳出开灯操作的循环
        } else {};
        irrecv.resume();
        delay(1000);
      }
    }

    //按下"向右"号键关灯
    else if (results.value == 0xFF5AA5) {
      digitalWrite(led_red, LOW);
      digitalWrite(led_green, LOW);
      digitalWrite(led_blue, LOW);
      Serial.println("关灯");
      lcd.clear();
      lcd.setCursor(0, 0);
      lcd.print("OFF");
    }

    //接收下一个值
    irrecv.resume();
  }
}

//functions
void brightAll() {
  while (1) {
    if (irrecv.decode(&results)) {
      Serial.println("总的亮度调节: ");
      Serial.println(results.value, HEX);
      lcd.clear();
      lcd.print("BV:mixed mode");
      if (results.value == 0xFF18E7 && val < 255) {
        val = val + 30;
      } else if (results.value == 0xFF4AB5 && val > 0) {
        val = val - 30;
      } else if (results.value == 0xFF38C7) {
        Serial.println("跳出");
        lcd.clear();
        lcd.print("Break");
```

```
        break;
      }
      analogWrite(led_red, val);
      analogWrite(led_green, val);
      analogWrite(led_blue, val);
      Serial.println(val);
      lcd.clear();
      lcd.setCursor(1,0);
      lcd.print("Mix_BV:");
      lcd.setCursor(1,8);
      lcd.print(val);
      irrecv.resume();
      delay(1000);
    }
  }
}
void brightRed() {
  digitalWrite(led_red, HIGH);
  digitalWrite(led_green, LOW);
  digitalWrite(led_blue, LOW);
  while (1) {
    if (irrecv.decode(&results)) {
      Serial.println("纯红色的亮度调节: ");
      Serial.println(results.value, HEX);
      lcd.clear();
      lcd.print("BV:red mode");
      if (results.value == 0xFF18E7 && val < 255) {
        val = val + 30;
      } else if (results.value == 0xFF4AB5 && val > 0) {
        val = val - 30;
      } else if (results.value == 0xFF38C7) {
        Serial.println("跳出");
        lcd.clear();
        lcd.print("Break");
        break;
      }
      analogWrite(led_red, val);
      Serial.println(val);
      lcd.clear();
      lcd.setCursor(1,0);
      lcd.print("Red_BV:");
      lcd.setCursor(1,8);
```

```
      lcd.print(val);
      irrecv.resume();
      delay(1000);
    }
  }
}
void brightGre() {
  digitalWrite(led_red, LOW);
  digitalWrite(led_green, HIGH);
  digitalWrite(led_blue, LOW);
  while (1) {
    if (irrecv.decode(&results)) {
      Serial.println("纯绿色的亮度调节: ");
      Serial.println(results.value, HEX);
      lcd.clear();
      lcd.print("BV:green mode");
      if (results.value == 0xFF18E7 && val < 255) {
        val = val + 30;
      } else if (results.value == 0xFF4AB5 && val > 0) {
        val = val - 30;
      } else if (results.value == 0xFF38C7) {
        Serial.println("跳出");
        lcd.clear();
        lcd.print("Break");
        break;
      }
      analogWrite(led_green, val);
      Serial.println(val);
      lcd.clear();
      lcd.setCursor(1,0);
      lcd.print("Gre_BV:");
      lcd.setCursor(1,8);
      lcd.print(val);
      irrecv.resume();
      delay(1000);
    }
  }
}
void brightBlu() {
  digitalWrite(led_red, LOW);
  digitalWrite(led_green, LOW);
  digitalWrite(led_blue, HIGH);
```

```
  while (1) {
    if (irrecv.decode(&results)) {
      Serial.println("纯蓝色的亮度调节: ");
      Serial.println(results.value, HEX);
      lcd.clear();
      lcd.print("BV:blue mode");
      if (results.value == 0xFF18E7 && val < 255) {
        val = val + 30;
      } else if (results.value == 0xFF4AB5 && val > 0) {
        val = val - 30;
      } else if (results.value == 0xFF38C7) {
        Serial.println("跳出");
        lcd.clear();
        lcd.print("Break");
        break;
      }
      analogWrite(led_blue, val);
      Serial.println(val);
      lcd.clear();
      lcd.setCursor(1,0);
      lcd.print("Blu_BV:");
      lcd.setCursor(1,8);
      lcd.print(val);
      irrecv.resume();
      delay(1000);
    }
  }
}
void rgb() {
  digitalWrite(led_red, LOW);
  digitalWrite(led_green, LOW);
  digitalWrite(led_blue, LOW);
  Serial.println("三色闪烁");
  lcd.clear();
  lcd.print("RGB");
  while (1) {
    digitalWrite(led_red, HIGH);
    delay(1000);
    digitalWrite(led_red, LOW);
    digitalWrite(led_green, HIGH);
    delay(1000);
    digitalWrite(led_green, LOW);
    delay(1000);
    digitalWrite(led_blue, LOW);
    if (irrecv.decode(&results)) {
      if (results.value == 0xFF38C7) {
        Serial.println("跳出");
        lcd.clear();
        lcd.print("Break");
        break;
      }
      irrecv.resume();
      delay(1000);
    }
  }
}
void lightR() {
  int lig;
  while (1) {
    Serial.println("光敏电阻亮度调节");
    lcd.clear();
    lcd.print("LDR");
    lcd.setCursor(0,1);
    lcd.print("BV:");
    lig = analogRead(potpin) / 4;
    Serial.println(lig);
    lcd.setCursor(4,1);
    lcd.print(lig);
    analogWrite(led_red, lig);
    analogWrite(led_green, lig);
    analogWrite(led_blue, lig);
    delay(300);
    if (irrecv.decode(&results)) {
      if (results.value == 0xFF38C7) {
        Serial.println("跳出");
        lcd.clear();
        lcd.print("Break");
        break;
      }
      irrecv.resume();
      delay(1000);
    }
  }
}
```