改造普通灯变成 WiFi 控制灯

一、 项目目的

1. 改造普通 usb 灯为 WiFi 控制灯，通过手机浏览器传输指令，实现对普通灯的开关控制。

二、 项目环境

1. NodeMCU（ESP8266）开发板一块
2. 高低电平继电器模块一个
3. 普通 USB 灯一个
4. 11V 转 3.3V、5V 变压器
5. 手机一台（可开移动热点和浏览网页）
6. 移动电源
7. 杜邦线若干、USB 线一根、USB 转 DC 电源线一根
8. Arduino IDE 1.8.19
9. Visual Studio Code

三、 项目原理

1. ESP8266 是一款物联网 WiFi 芯片，基于 ESP8266 可以开发物联网串口 WiFi 模块，可将物理设备连接到 WiFi 无线网络上，进行互联网或局域网通信，实现联网功能。本次实现采用 ESP8266 的 STA（Station）模式，即 ESP8266 开发板作为网络服务器端，与同一局域网内的客户端，如手机，进行 HTTP 协议通信。
2. 用户在客户端开启移动热点，ESP8266 开发板连接该 WiFi 热点，就会生成一个 IP 地址，并通过 Arduino IDE 的串口监视器输出。考虑到便捷性，不用每次都控制 WiFi 灯都查看串口监视器，可以设置固定 IP 地址，只要该 IP 地址与手机的 IP 地址在同一局域网内，即可进行通信，从而实现一个较为完整的系统。
3. ESP8266 的数字输出作为高低电平继电器模块的数字输入。在本次实验中，任意边沿来临时高低电平继电器模块输出接通，任意边沿再次来临则高低电平继电器模块输出断开，以实现开关的效果。

4. 11V 转 3.3V、5V 变压器的作用是通过连接一个电源，可以同时给多个不同电压需求的设备供电。

四、 项目步骤与结果

1. 在 Arduino IDE 编写控制 WiFi 板的代码。

1.1 总体程序设计

程序流程如下图 1，控制 WiFi 板全部代码见附录代码连接。代码主要由 setup()函数和 loop()函数组成。setup()函数的内容用于初始化设置，如设置输出引脚和连接 WiFi，只运行一次；loop()函数检查是否有客户端设备通过网络向 ESP8266 网络服务器发送请求，一直循环运行。

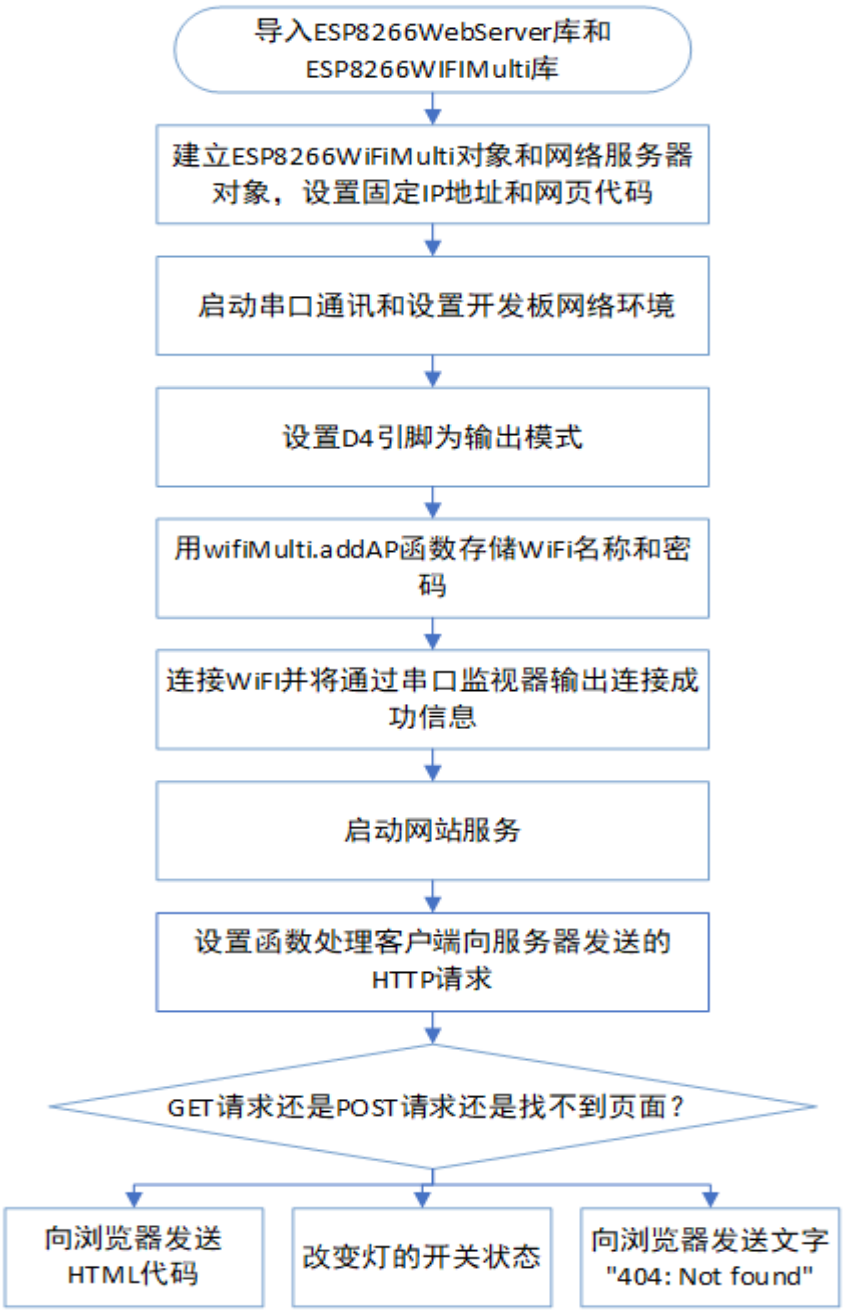

图 1

1.2 设置 WiFi 板的固定 IP 地址和网络环境。

调用 IPAddress 定义四个变量:local_IP、gateway、subnet 和 dns,分别为本地 ip、网关 ip、子网掩码和 dns 的 ip，如图 2。然后调用 WIFI.config 函数完成 IP 配置，用函数的返回结果做判断，如果设置成功返回 ture，设置失败，返回 false，如图 3。

```
//设置固定IP地址
IPAddress local_IP(192, 168, 121, 123); // 设置ESP8266-NodeMCU联网后的IP
IPAddress gateway(192, 168, 121, 1);    // 设置网关IP（通常网关IP是WiFI路由IP）
IPAddress subnet(255, 255, 255, 0);   // 设置子网掩码
IPAddress dns(192,168,121,1);           // 设置局域网DNS的IP
```

图 2

```
// 设置开发板网络环境
if (!WiFi.config(local_IP, gateway, subnet)) {
  Serial.println("Failed to Config ESP8266 IP");
}
```

图 3

1.3 ESP8266 网络服务器的主要工作代码

当 ESP8266 开发板利用 begin 函数开启网络服务以后，每当有客户端向服务器发送 HTTP 请求时，利用 on 函数设置 HTTP 请求回调函数。利用 onNotFound 函数处理页面找不到的情况，代码如下图 4。

```
esp8266_server.begin();                          // 启动网站服务
esp8266_server.on("/", HTTP_GET, handleRoot);    // 设置服务器根目录即'/'的函数'handleRoot'
esp8266_server.on("/LED", HTTP_POST, handleLED); // 设置处理LED控制请求的函数'handleLED'
esp8266_server.onNotFound(handleNotFound);       // 设置处理404情况的函数'handleNotFound'
```

图 4

1.4 处理请求开灯或关灯的 http 请求

每次 ESP8266 网络服务器接收到来自浏览器的开灯或关灯请求时，将当前 D4 引脚的电平输出状态改成相反的输出状态，核心代码为：

digitalWrite(D4,!digitalRead(D4));

即如果原来是高电平则改成低电平输出，原来低电平则改成高电平输出，从而控制灯的开关状态。

2. 在 Visual Studio Code 编写网页代码。

网页界面如图 5，网页详细代码见附录代码链接。核心代码为：

```
<form action="/LED" method="POST\">
LampController
<input type="submit" value="turn on/off">
</form>
```

用户在客户端点下灯开关按钮，提交表单，即向 ESP8266 网络服务器发送一个 POST 请求。

接下来是压缩 html 代码，将 html 代码以字符串的形式写入控制 WiFi 板的代码。压缩 html 代码时，在 Visual Studio Code 将双引号用单引号替换，否则 html 里的双引号会与两端表示字符串的双引号结合，提前结束 html 代码，引发程序错误。删除空格和换行。

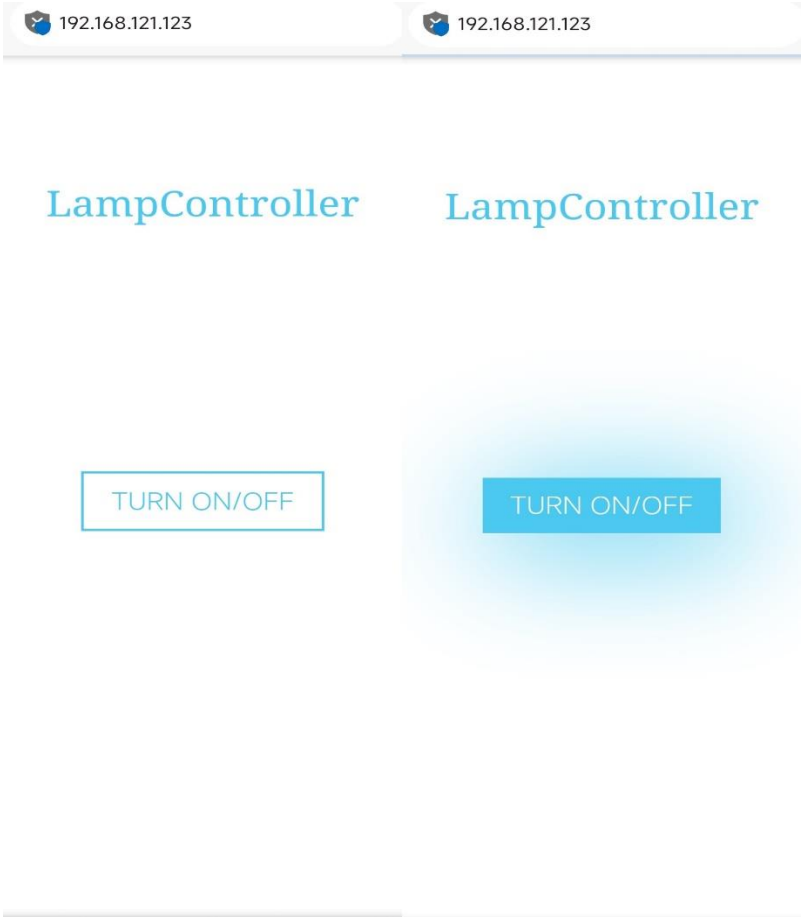

图 5 左：点击按钮前；右点击按钮后

3. 烧录。

烧录时，要注意保持 WiFi 板的端口连接正常。

4. 接线。总体接线如图 6。

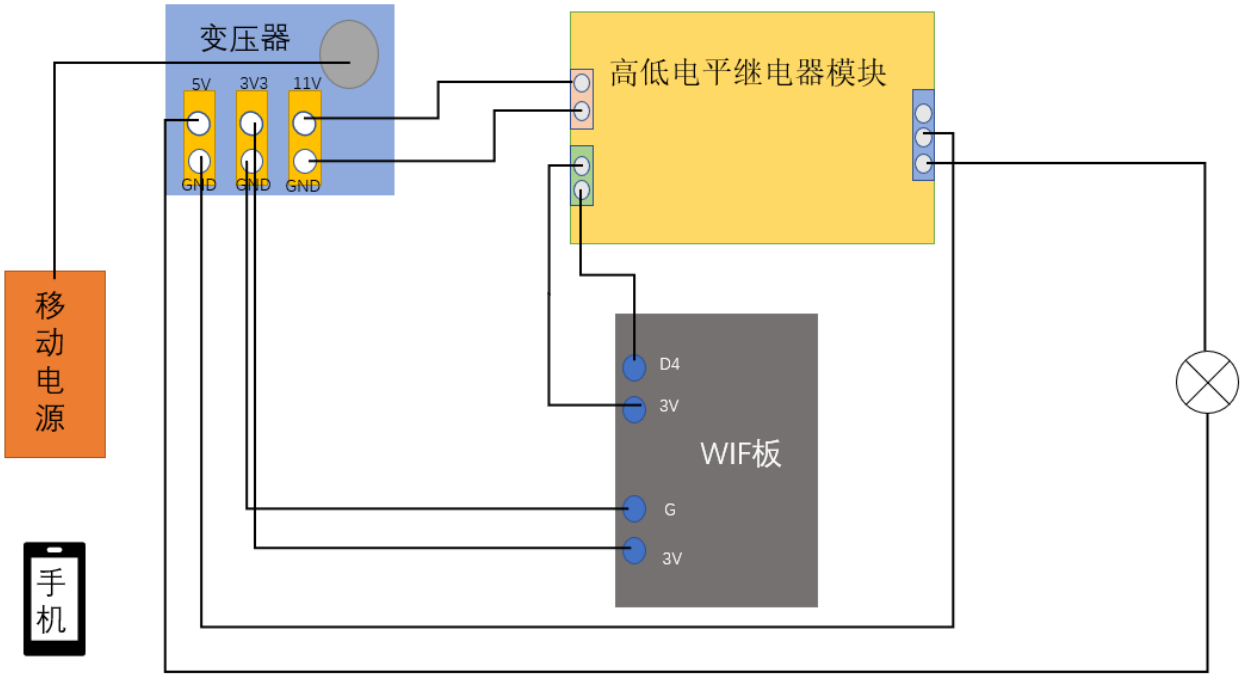

图 6

4.1 给变压器、继电器接通电源和选择模式。

将变压器连接上移动电源，用杜邦线将变压器 11V 引脚和继电器电源正极输入连接，GND 引脚和电源负极输入连接。继电器通电后，长按“M”按键来解锁设置，选择延时功能模式 F--1，再选择开关功能模式 0101，任意边沿自锁，即任意边沿来临时输出接通，任意边沿再次来临则输出断开，如图 7。

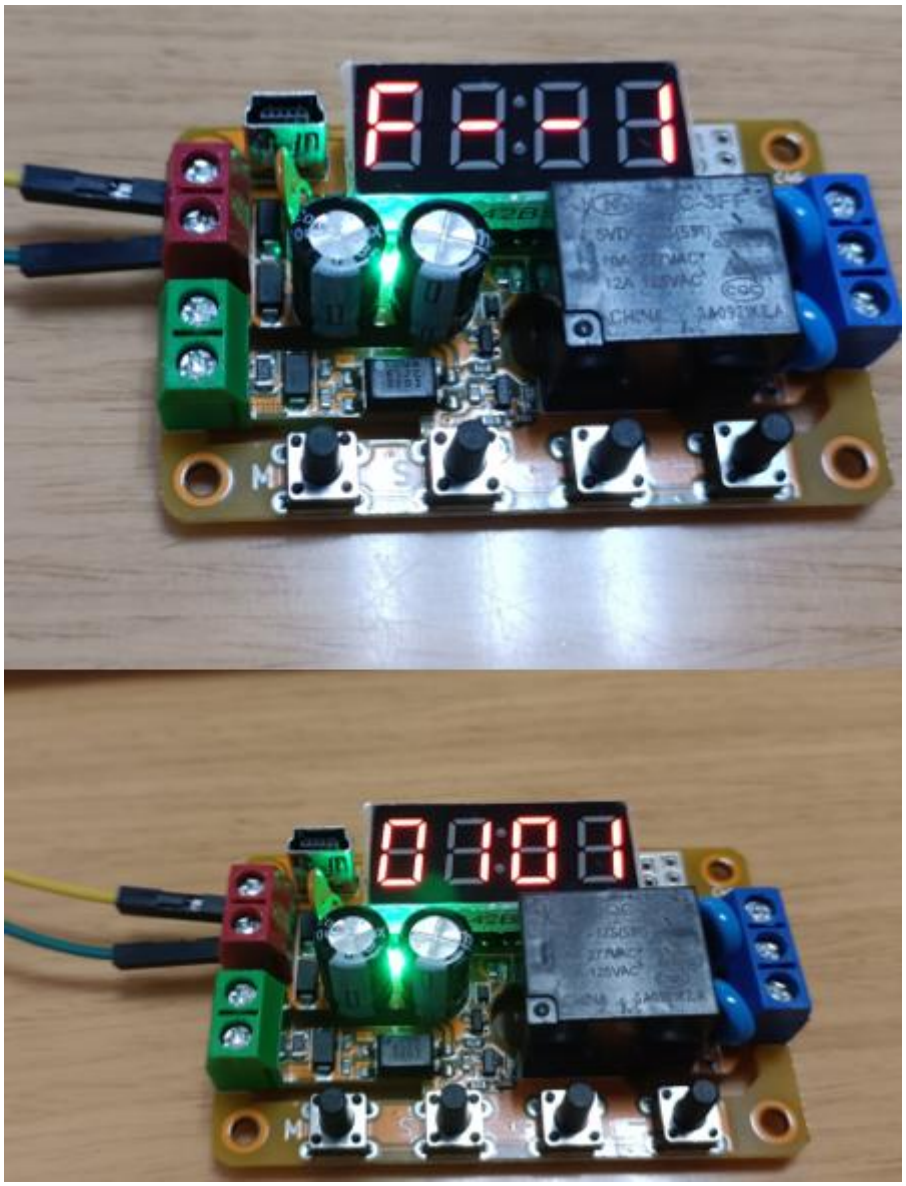

图 7

4.2　连接 WiFi 板和变压器、电器。

第一步，杜邦线将 WiFi 板的 G 引脚和变压器的 GND 引脚、WiFi 板的 3V 引脚和变压器的 3V 引脚连接。第二步，杜邦线连接高低电平继电器模块的数字输入正极和 WiFi 板 3V 引脚、数字输入负极和 D4 引脚，如图 8。

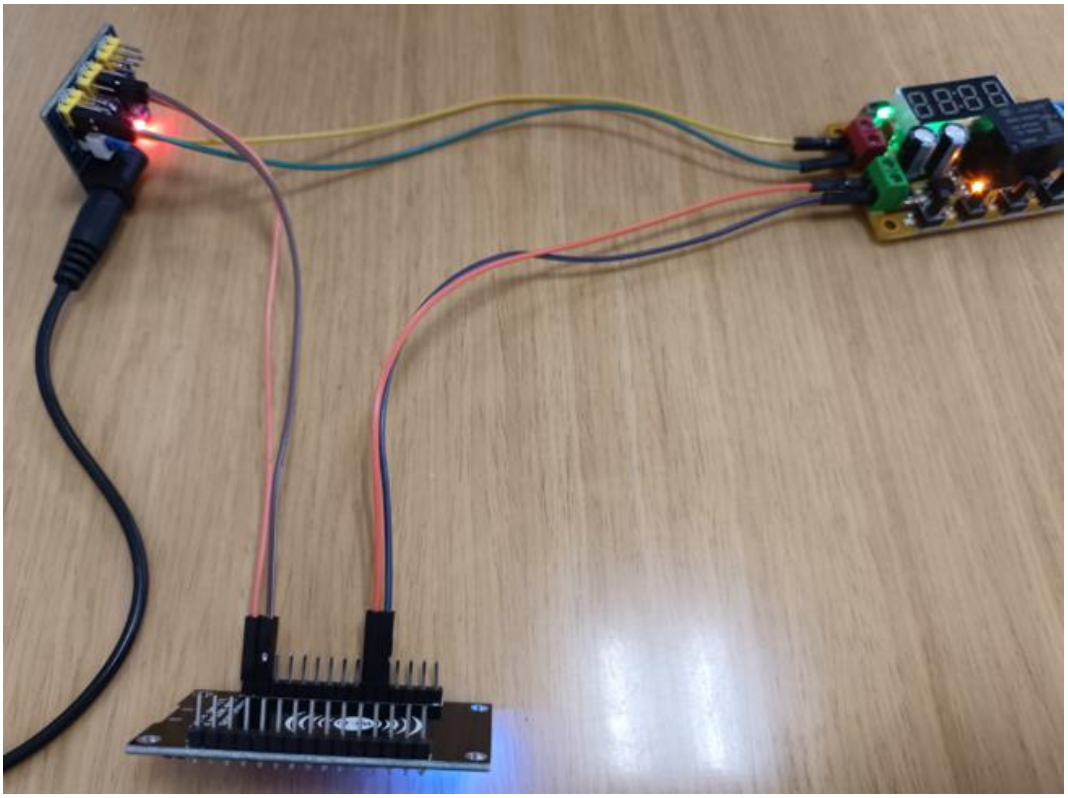

图 8

4.3　连接变压器、继电器和 USB 灯。

第一步，将 USB 灯的插头端剪开，并去除绝缘皮，使得电源+5V 导线（红色）裸露出来，地线（黑色）裸露出来。第二步，剪开两根杜邦线，并去除结缘皮，如图 9。第三步，将杜邦线和导线分别绞合在一起，拧紧，如图 10。将 USB 灯的正极（红色导线）连接的杜邦线接入变压器 5V 引脚，连接地线（黑色）的杜邦线接入继电器的常开触点。第四步，用杜邦线将 5V 引脚下的 GND 引脚与公共触点连接。至此，整个完成装置，效果图如图 11 所示。演示视频见附录视频链接。

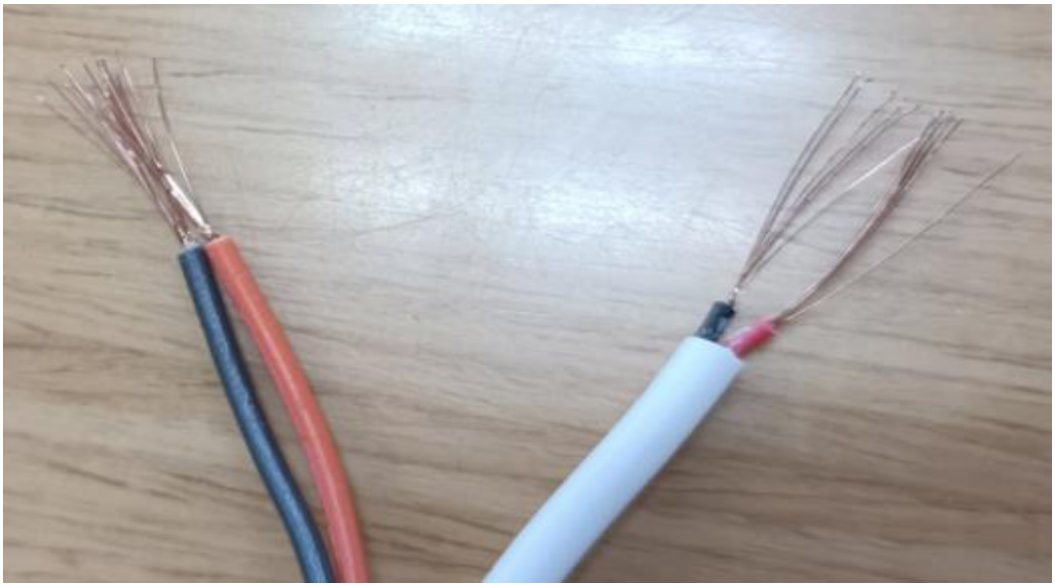

图 9

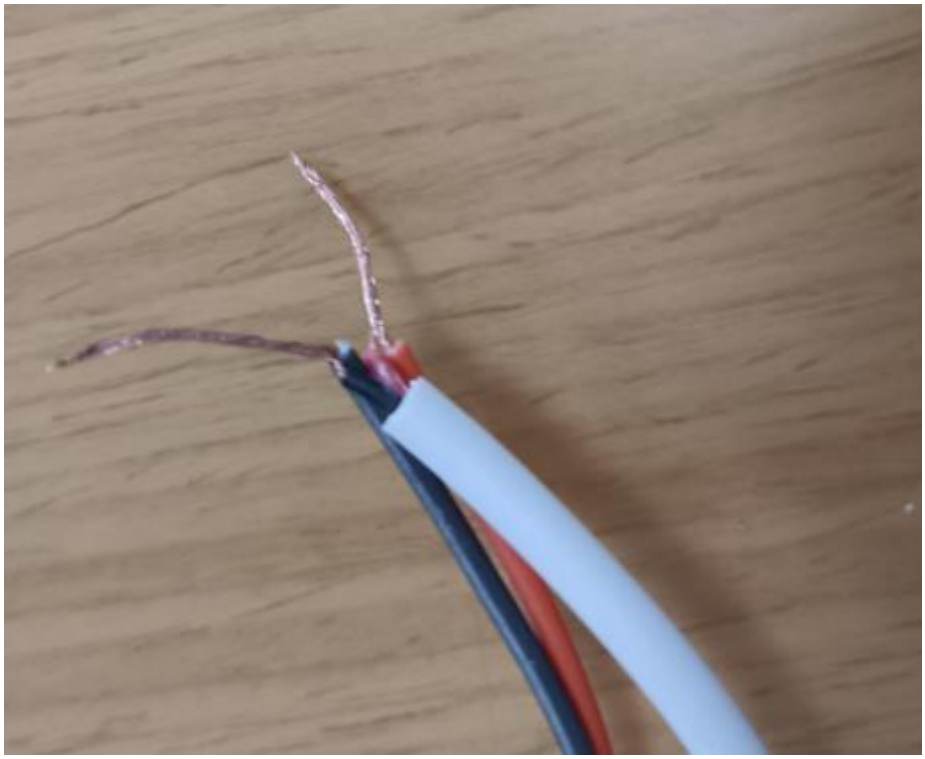

图 10

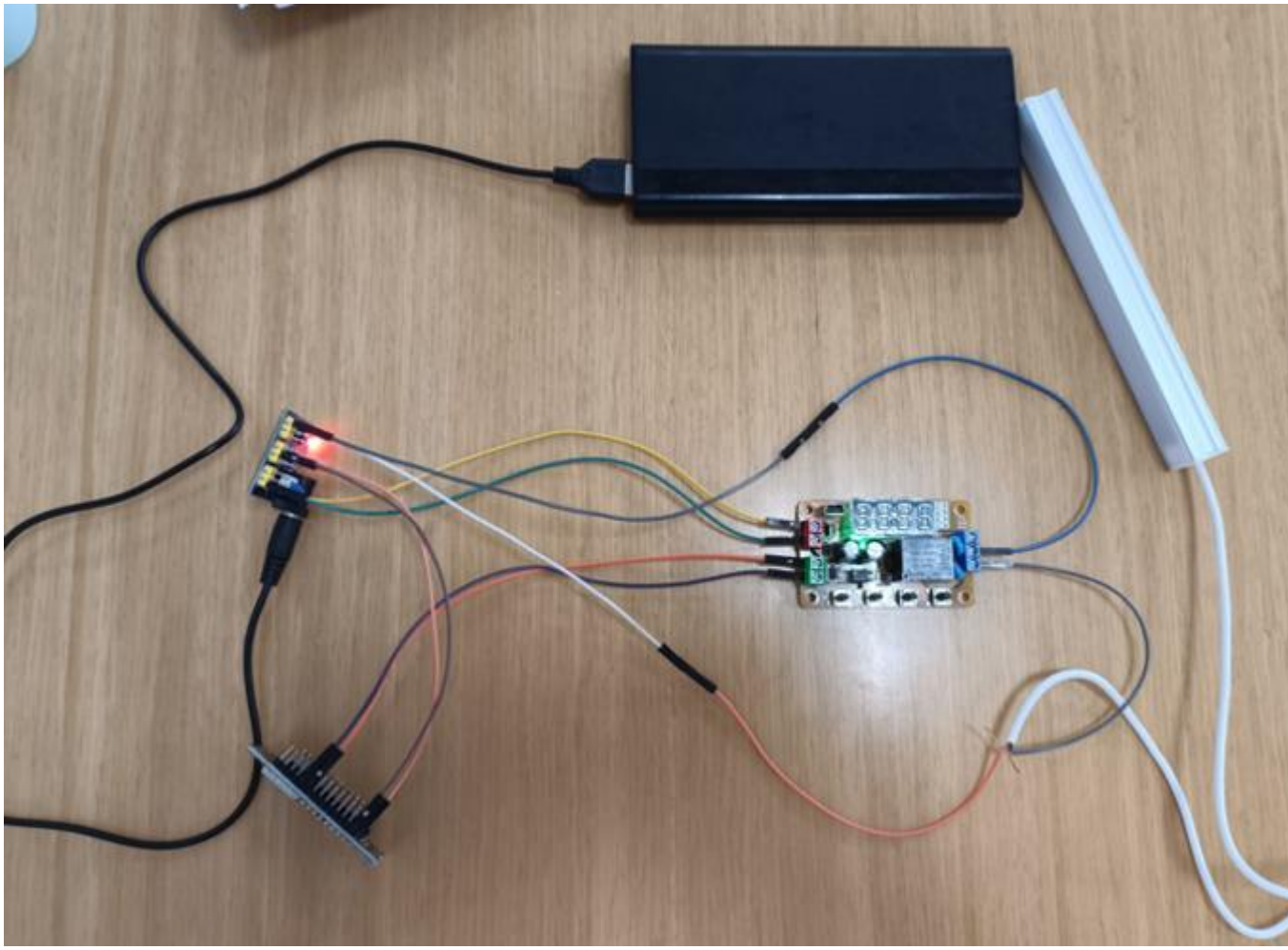

图 11

五、 项目总结

1. 实验结果评价。

本次实验填补了我许多知识的空白，实验结果我认为该系统是比较完善的，虽然仍存在一定的瑕疵。WiFi 板使用固定 IP 地址带来一定的便捷，但也存在一定的缺陷。如果是用手机流量开热点，由于手机 IP 地址是随着手机信号连接的基站的改变而改变得，有时候手机的 IP 地址和 WiFi 板的 IP 地址不在同一个局域网，使得两者之间无法通信，如果使用路由器连接可以解决这个缺点，WiFi 板的 IP 地址要设置在路由器分配 IP 地址的范围内。该系统可以进一步完善，与云平台配合，实现远程控制。

2. 实验中遇到的主要问题的分析与处理

问题 1：ESP8266 WiFi 板无法连接手机热点。

分析：手机热点频段有 2.4GHz 和 5GHz，ESP8266 不支持 5GHz 频段，如果手机热点开启时选择了 5GHz 频段，会导致 ESP8266 WiFi 板无法连接手机热点。

处理：手机热点开启时选择 2.4GHz 频段即可解决。

问题 2：如果使用手机热点连接的 ESP8266，每次都要通过串口监视器查看 WiFi 板的 IP 地址，能否实现一个较为独立的系统，不用每次控制灯都要查看 IP 地址？
分析：可以通过设置 WiFi 板连接网络后的 IP 地址，只有手机与该 WiFi 板处于同一个局域网即可互相通信。
处理：通过函数 WIFI.config 设置 WiFi 板的网络环境。

附录

1. 演示视频链接：
https://b23.tv/BA0CSyV?share_medium=android&share_source=qq&bbid=XX4B4504E7F2F8068BF467926084DEB7A888F&ts=1654171605571

2. 代码链接：
https://github.com/Avivi210/ESP8266-WiFi-LED.git