课 程 设 计

设计题目	基于 esp8266 的远程开关灯控制
学　　号	206001829 206000990
学生姓名	罗宇涛欧阳宁
专业班级	计算机科学与技术 1 班
指导教师	江先亮

2022 年 12 月 24 日

目录

一、 项目背景

在寒冷的冬日，钻进了温暖的被窝却不料家中有盏灯还没关，这时我们不免想如果能够在床上就能实现开关灯的控制多好。

具体的，人在楼上，不必跑到楼下去关灯等; 说必要的，对上年纪、腿脚不方便的老人，可以通过场景开关，进行开关灯等; 说实际的，远程开关也进一步解放了体力，生活变得更简单。有些情况经常遇到，如出门之后，心头突然想到是不是忘记关灯了，非常纠结却很无奈。远程开关的出现，可以说完全解决了这种忧虑，完全可以随时随地通过手机查看家、办公室等地方照明状态，并进行合理的开关。

因此我们设计了基于 esp8266 的远程开关灯控制

二、 设计目的与目标

2.1 设计目的

1. 掌握 51 单片机 ESP8266 的使用
2. 掌握使用在 EMQX 平台上搭建 MQTT 协议
3. 理解和掌握 docker 搭建网站

2.2 设计目标

可在手机端、pc 端显示房间内所有灯的状态，并通过舵机实现实时远程灯光控制

三、 课程设计器材

3.1 硬件平台

1. Esp8266 nodeMCU 开发板
2. SG90 舵机
3. 腾讯云服务器

3.2 软件平台

1. ubantu 操作系统
2. EMQX 平台
3. pytharm 编译器

四、 逻辑设计总体方案

4.1 系统总体概述

我们采用舵机，在开关外部实现驱动开关，会影响一定美观，但家庭实际上完全可以使用继电器或者可控硅，修改开关内部电路代替舵机。同时价格便宜，成本低，大部分家庭能够接受，让自己的家更智能！这项技术更能横向扩展到，控制电机棒，您在外面也能物理的按下家中的按钮。利用不同的电机来代替简单的物理动作。结合一些物理材料，也可以实现宠物饲料的控制投放等，具有强大的可扩展性。

1. 相互控制：房间里所有的灯都可以在手机上控制
2. 照明显示：房间里所有电灯的状态会在手机上显示出来。
3. 全关功能：可一键关闭房间里所有的电灯或关闭任何一个房间的灯。
4. 自动化控制：设定每天 11 点自动执行关灯指令。
5. 语音控制：“嘿，Siri，关闭/打开寝室灯”，即可控制寝室灯关闭/打开

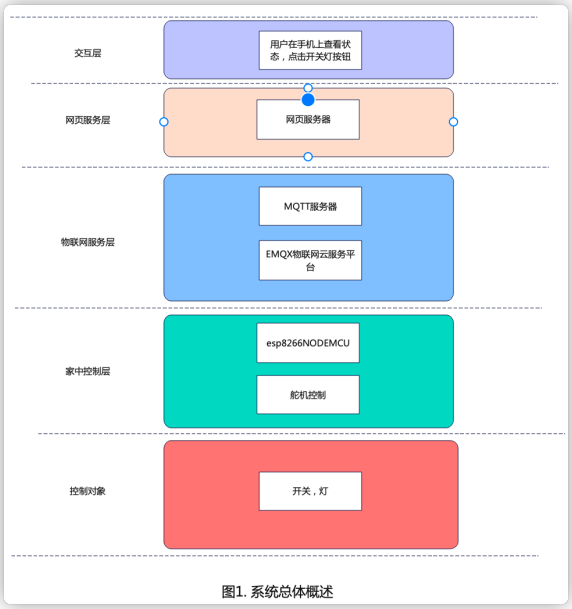

图 4.1 系统总体概述

4.2 mqtt 概念

MQTT 是一个基于客户端-服务器的消息发布/订阅传输协议。MQTT 协议是轻量、简单、开放和易于实现的，这些特点使它适用范围非常广泛。在很多情况下，包括受限

的环境中，如：机器与机器（M2M）通信和物联网（IoT）。其在，通过卫星链路通信传感器、偶尔拨号的医疗设备、智能家居、及一些小型化设备中已广泛使用。

4.3 homeassit 平台

Home Assistant 是一个成熟完整的基于 Python 的智能家居系统，设备支持度高，支持自动化（Automation）、群组化（Group）、UI 客制化（Theme）等等高度定制化设置。背后又有庞大的社群基础，且不断在更新。最重要的是我们可以通过 Homeassistant-homebridge 插件打通两个平台，同样实现设备的 Siri 控制。并且解决了对于个人使用者，买多个品牌商的智能产品，却每个厂商使用协议不同，提供管理的软件平台不同等待，无法联合使用的巨大问题。Home Assistant 作为一个桥梁，每个品牌的产品都可以接入到这个平台通一管理，在每个产品基础上，实现网页的统一控制，自动化联动，真正意义上在家庭中实现万物互联。

图 4.2 homeassit 平台

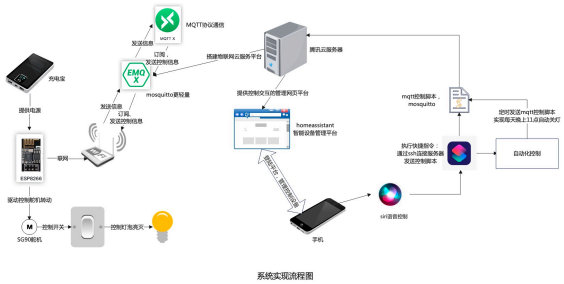

图 4.3 系统实现流程图

1. 在手机打开 homeassistant 提供的控制台网页（云服务器 ip:8123），按下控制开关灯按钮。

2.homeassistant 向控制开关灯的 mqtt 消息给 mosquitto 平台 (搭建在 1883 端口)，发布订阅主题 luo/home / pubupdate，内容为控制消息 on or off。

3.mosquitto 平台向所有订阅了 luo/home / pubupdate 的设备发送该消息 4.Esp8266NodeMCU 连接了家庭的 WIFI，并对该平台（云服务器 ip:1883）订阅了 luo/home / pubupdate 主题，接收到开关灯的控制消息。执行消息的处理。

图 4.4　展示图

图 4.5　**homeassistant** 连接的 **mqtt** 服务地址

```
// wifi info
const char *SSID = "macbook"; // wifi 名字
const char *PASSWORD ="12345678"; // wifi密码

// MQTT info https://www.emqx.com/zh/mqtt/public-mqtt5-broker
const char* MQTT_SERVER = "test.ranye-iot.net"; // MQTT 服务器地址
const int MQTT_PROT = 1883;                     // MQTT 端口

// mqtt 主题
const char *MQTT_TOPIC_PUB_ONLINE = "luo/home/pub_online";
const char *MQTT_TOPIC_PUB_UPDATE = "luo/home/pub_update"; // 订阅更新
const char *MQTT_TOPIC_SUB_DATA = "luo/home/sub_data";     // 发布
const char *MQTT_TOPIC_HEARTBEAT = "luo/home/heartbeat";   // 心跳
const char *CLIENT_ID = "esp8266-457065e5-f063-474e-2saf-cacf5cb52af5";
```

图 4.6　**WI-FI** 和 **mqtt** 主题

5.Esp8266NodeMCU 根据控制消息，控制 SG90 舵机，执行开关灯的旋转，物理的开关被按下开/关。

4.4 Siri 语音控制流程：

利用 iphone 自带的快捷指令功能，对 Siri 说特定语句会执行对应快捷指令。快捷指令里编写脚本，实现与云服务器的 ssh 远程连接并执行 shell 脚本，通过 mosquitto 的指令发布主题消息。

4.5 自动化控制流程：

自动化控制在 homeassistant 和 iphone 上均可实现，我们采用了 ipone 的快捷指令，增加定时任务，23:00 执行连接服务器和执行 mqtt 消息的控制脚本，向 luo/home / pubupdate 发布 off 的控制命令。

若在 homeassistant 实现，则还可以与其他智能设备联动。（图 4.7 至图 4.8 ）

4.6 esp8266 NodeMCU 烧录代码

代码展示（图 4.9 至图 4.14 ）

五、 总结展望

基于 esp8266 的远程开关灯控制的设计与实现涉及多方面的理论、方法和技术，本系统还有许多新的问题需要解决，需要在实际应用中不断积累和完善，在以下几个方面，还需要做进一步的研究和开发。

1. 论文并未在智能家居等方面进行深入发掘。使得系统只考虑了一些简单情况，如何应付复杂的应用场景还值得加强。可以进一步提升用户体验。

2. 该系统只考虑了寝室开关灯应用的一些基本情况，未对其他应用场景进行更为深入的研究，使得实际使用时效果不尽如人意。

3. 系统在测试上还存在一些问题, 不过都是预期可以解决的。

mqtt 等相关网络技术在该系统的开发过程中应用范围还不够，还有提高的余地。

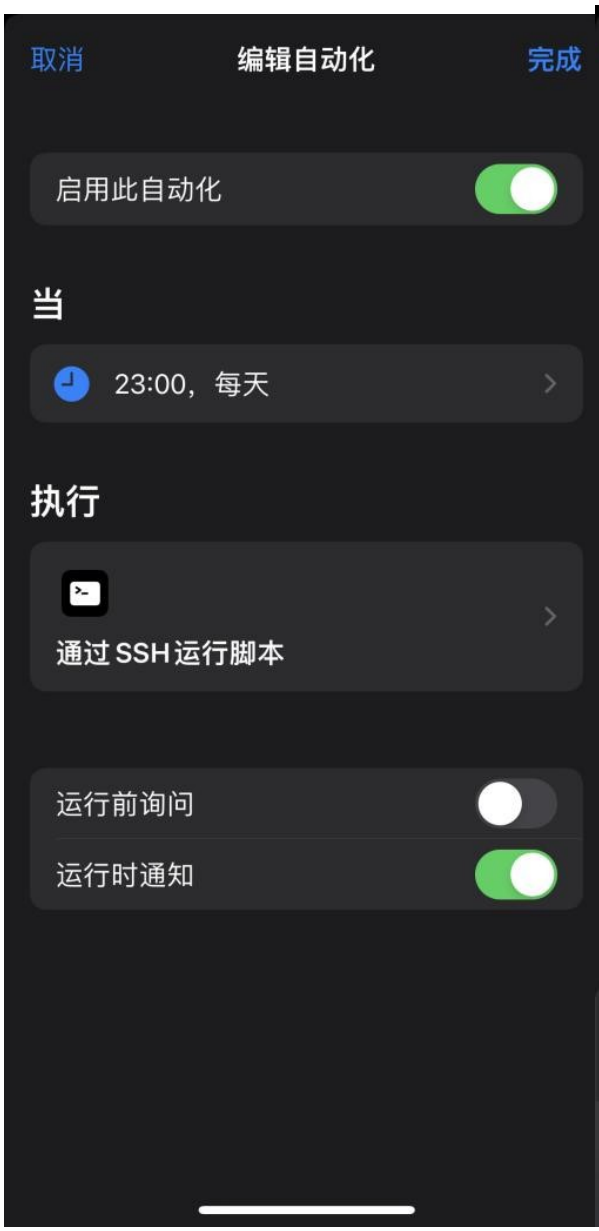

图 4.7　ssh 脚本 1

图 4.8　ssh 脚本 2

```
//#include <Arduino.h>
#include <Ticker.h>
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <Servo.h>

#define SERVO_PIN  14

// wifi info
const char *SSID = "macbook"; // wifi 名字
const char *PASSWORD ="12345678"; // wifi密码

// MQTT info https://www.emqx.com/zh/mqtt/public-mqtt5-broker
const char* MQTT_SERVER = "test.ranye-iot.net"; // MQTT 服务器地址
const int MQTT_PROT = 1883;                  // MQTT 端口

// mqtt 主题
const char *MQTT_TOPIC_PUB_ONLINE = "luo/home/pub_online";
const char *MQTT_TOPIC_PUB_UPDATE = "luo/home/pub_update"; // 订阅更新
const char *MQTT_TOPIC_SUB_DATA = "luo/home/sub_data";     // 发布
const char *MQTT_TOPIC_HEARTBEAT = "luo/home/heartbeat";   // 心跳
const char *CLIENT_ID = "esp8266-457065e5-f063-474e-2saf-cacf5cb52af5";

// SG90
int SG90_i;
#define default_angle 90   //舵机复位角度
#define rotation_angle 40  //旋转角度，复位角度为90时最大为90，复位角度为0时最大180
#define delay_time 1000   //归位延迟时间

// ticker.attach(s秒数，函数名)
Ticker ticker; // 定时调用某一个函数
WiFiClient espClient;
PubSubClient client(espClient);
Servo servo;//初始化舵机对象
```

图 **4.9**　连接 **wifi** 信息 **mqtt** 主题 **SG90** 控制变量

```
//#include <Arduino.h>
#include <Ticker.h>
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <Servo.h>

#define SERVO_PIN  14

// wifi info
const char *SSID = "macbook"; // wifi 名字
const char *PASSWORD ="12345678"; // wifi密码

// MQTT info https://www.emqx.com/zh/mqtt/public-mqtt5-broker
const char* MQTT_SERVER = "test.ranye-iot.net"; // MQTT 服务器地址
const int MQTT_PROT = 1883;                  // MQTT 端口

// mqtt 主题
const char *MQTT_TOPIC_PUB_ONLINE = "luo/home/pub_online";
const char *MQTT_TOPIC_PUB_UPDATE = "luo/home/pub_update"; // 订阅更新
const char *MQTT_TOPIC_SUB_DATA = "luo/home/sub_data";     // 发布
const char *MQTT_TOPIC_HEARTBEAT = "luo/home/heartbeat";   // 心跳
const char *CLIENT_ID = "esp8266-457065e5-f063-474e-2saf-cacf5cb52af5";

// SG90
int SG90_i;
#define default_angle 90   //舵机复位角度
#define rotation_angle 40  //旋转角度，复位角度为90时最大为90，复位角度为0时最大180
#define delay_time 1000   //归位延迟时间

// ticker.attach(s秒数，函数名)
Ticker ticker; // 定时调用某一个函数
WiFiClient espClient;
PubSubClient client(espClient);
Servo servo;//初始化舵机对象
```

图 **4.10**　主程序

```
void init_wifi() //初始化连接wifi函数
{
  Serial.println("Connecting to");
  Serial.println(SSID);

  WiFi.begin(SSID, PASSWORD);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
}
```

图 4.11　主程序用到的函数 1

```
void mqtt_reconnect()  //连接mqtt函数
{
  while (!client.connected())
  {
    Serial.print("Attempting MQTT connection...");

    // 第一步：创建连接
    if (client.connect(CLIENT_ID))
    {
      Serial.println("connected");
      client.publish(MQTT_TOPIC_PUB_ONLINE, "online"); // 发布
      client.subscribe(MQTT_TOPIC_PUB_UPDATE);         // 监听
    }
    else
    {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      delay(5000);
    }
  }
}
```

图 4.12　主程序用到的函数 2

```
void mqtt_msg_callback(char *topic, byte *payload, unsigned int length) //接收控制消息后执行函数
{
  Serial.print("Message arrived [");
  Serial.print(topic); // 打印主题信息
  Serial.print("] ");
  String data = "";
  for (unsigned int i = 0; i < length; i++)
  {
    Serial.print((char)payload[i]); // 打印主题内容
    data += (char)payload[i];//存mqtt数据
  }
  Serial.println();

  if (data == "on") {

        SG90_update(default_angle + rotation_angle);
        Serial.printf("on");
        delay(delay_time);//延时1秒
        SG90_update(default_angle);//舵机归零，回到垂直状态


    }
    else if (data =="off") {
         SG90_update(default_angle - rotation_angle);
         Serial.printf("off");
         delay(delay_time);
         SG90_update(default_angle);
    }

}
```

图 4.13　主程序用到的函数 3

```
void mqtt_heartbeat() //心跳发布函数
{
  if (client.connected())
  {
    client.publish(MQTT_TOPIC_HEARTBEAT, "1");
  }
}

void SG90_update(int data){ //SG90舵机控制函数
  servo.write(data);//data为设置转动的角度
  delay(300);
}

void SG90_reset()  //SG90舵机复位函数
{
  servo.write(default_angle);//上电时舵机归零垂直
}
```

图 4.14　主程序用到的函数 2