《物联网与嵌入式技术》

大作业课题报告

基于乐高小车的扫地机器人设计

学 生 姓 名：柏广庆

学科、 专业：计算机科学与技术

学　　号：32309259

完 成 日 期：2024.04.15

大连理工大学

Dalian University of Technology

目　　录

1 任务概览

本人在此次项目中承担嵌入式嵌入式软件的编写以及物联网的具体实现。

本项目的预期功能是通过连网终端通过云端服务器作为中转远程控制小车以实现物联网远程清洁功能，为用户提供方便且实用的服务。

要完成以上效果，技术实现过程可总体分为四步：

① 、在单片机硬件平台上编写嵌入式程序实现本地控制电机以及伺服电机；

② 、建立物联网云平台，实现消息的中转和互联；

③ 、开发用户端控制软件；

④ 、功能整合、联调联试。

下面将就这四点分别进行详细介绍。

2 实现细节

本部分首先对本项目的硬件方案进行简单介绍，后面的工作分为四个点进行叙述，分别为嵌入式软件开发、物联网平台开发、用户端软件开发以及整合调试。

2.1 硬件方案选型

为了平衡项目成本与功能实现，经调研，决定使用 ESP8266 NodeMCU 作为控制芯片，L298N 进行电机驱动，动力部分采用两个 LEGO 原装的电机及伺服电机。

ESP8266 NodeMCU 集成了 ESP12-E WIFI 模块，支撑了小车的物联网需求，并且其体积较小，功率较低，适用于终端设备的长时间续航；L298N 双 H 桥直流电机驱动芯片是常见的电机驱动模块，稳定耐用是其优点。

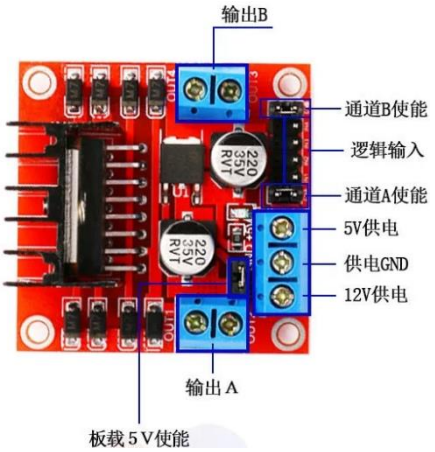


图 1　L298N 示意图

如图 1 所示，L298N 支持两个电机的驱动和控制。在本项目中，由于前后双电机的控制逻辑是相同的，于是在此我们使用一端输出信号同时控制两个电机，另一端输出用于驱动伺服电机。

L298N 支持两种供电方式，5V 供电和 12V 供电。由于在 12V 供电下，板载的 5V 使能引脚可以作为 ESP8266 的电压源，于是本次采用 12V 供电模式。

在使能端 ENA、ENB 均为高电平时，L298N 的控制方式为，通过对输入端 IN1-IN4 进行电平设置，L298N 驱动会产生相应的 PWM 信号来作为下游驱动信号。IN1\IN2 用于 A 端的输出，IN3\IN4 用于 B 端的输出。控制逻辑如下：

IN1	IN2	电机动作
0	0	制动
0	1	正转
1	0	反转
1	1	制动

表 1　L298N 的控制逻辑

此外，L298N 还支持 PWM 调速，本次我们设置为高电平，即默认最大速度运行。

对于伺服电机，控制逻辑类似，如下所示：

IN3	IN4	电机动作
0	0	复位
0	1	左转
1	0	右转

表 2　LEGO 伺服电机的控制逻辑

2.2　嵌入式软件开发

本次项目中使用 Arduino IDE 进行开发。

2.2.1　动力系统嵌入式软件设计

为了实现小车的基本运动控制，首先构建了基于串口控制的嵌入式软件开发。通过键盘输入对应的按键，实现小车的前后左右四个方向的运动。拓扑结构为：PC 通过 USB 连接 ESP8266 NodeMCU，ESP8266 通过 D5-D8 四个 IO 输出端口输出控制信号连接到 L298N 驱动芯片，L298N 驱动连接两个电机以及伺服电机，剩余为电源连线等，如图 2 所示。控制逻辑如表 1 和表 2 所示。

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>
#include <U8g2lib.h>
#include <Wire.h>
#include <stdlib.h>

#define IN1 14          // (D5)   --> IN1
#define IN2 12          // (D6)   --> IN2
#define IN3 13          // (D7)   --> IN3
#define IN4 15          // (D8)   --> IN4

void both_stop(){
   digitalWrite(IN1, LOW);
   digitalWrite(IN2, LOW);

   digitalWrite(IN3, LOW);
   digitalWrite(IN4, LOW);
   delay(100);
}

void both_forward(){
   digitalWrite(IN1, LOW);
   digitalWrite(IN2, HIGH);

   delay(1000);
   both_stop();
}

void both_backward(){
   digitalWrite(IN1, HIGH);
   digitalWrite(IN2, LOW);

   delay(1000);
   both_stop();
```

```
}

void co_left(){
   digitalWrite(IN3, LOW);
   digitalWrite(IN4, HIGH);
   delay(1000);
}

void co_right(){
   digitalWrite(IN3, HIGH);
   digitalWrite(IN4, LOW);
   delay(1000);
}

void setup(){
      pinMode(LED_BUILTIN, OUTPUT);

      pinMode(IN1, OUTPUT);
      pinMode(IN2, OUTPUT);
      pinMode(IN3, OUTPUT);
      pinMode(IN4, OUTPUT);

      /* initialize serial for debugging */
      Serial.begin(115200);
      Serial.println("Demo Start");
}

// the loop function runs over and over again forever
void loop(){
   while (Serial.available()) {
      char c = Serial.read();
      Serial.println(c);
      if(c=='1'){
         both_forward();
         Serial.println("go");
      }else if(c=='2'){
```

```
        both_backward();
        Serial.println("back");
      }else if(c=='3'){
        co_left();
      Serial.println("left");
      }else if(c=='4'){
        co_right();
        Serial.println("right");
      }else if(c=='5'){
        both_stop();
        Serial.println("stop");
      }else{
        Serial.println("default");
      }
   }
}
```

代码的总体思路就是通过控制 IO 输出电平控制驱动芯片，进而达到控制电机的目标。通过实际测试，实现了通过串口发送键盘字符控制相应的小车动作，较好地达到了预期目标。

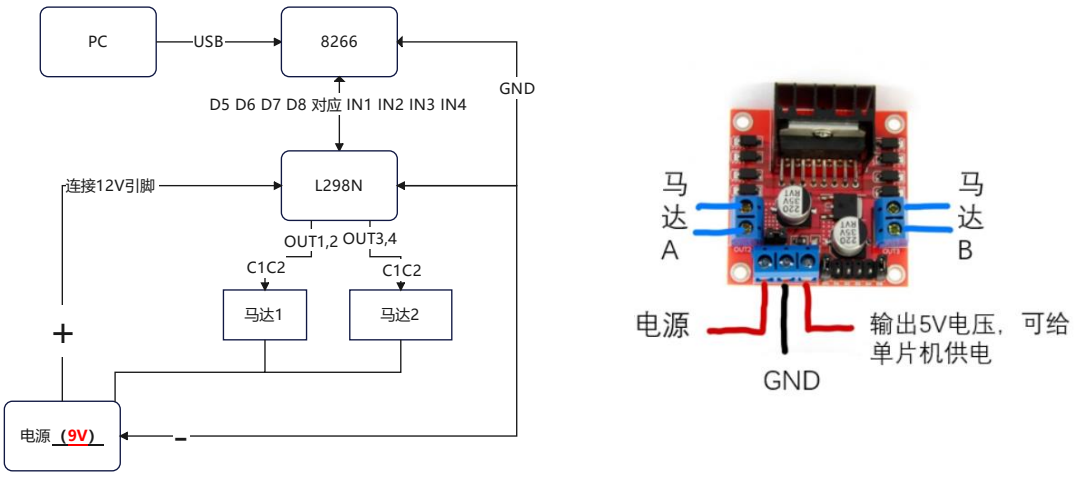


图 2 动力系统嵌入式软件测试连线图

2.2.2 物联网嵌入式软件设计

为了测试 ESP8266 与云平台的信息传输有效性，设计了如下测试代码，实现远程控制 ESP8266 的 LED 的亮灭。代码的整体思路，首先将 ESP8266 与 WIFI 连接，实现

联网；之后编写回调函数，以实现接收到服务端信息后进行信息的解码以及后续控制；之后就是不断地测试连接连通性。具体实现代码如下：

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>

#include "aliyun_mqtt.h"

#define WIFI_SSID "YOUR_WIFI_NAME"
#define WIFI_PASSWD "YOUR_WIFI_PASSWORD"

#define PRODUCT_KEY "YOUR_PRODUCT_KEY"
#define DEVICE_NAME "esp8266"
#define DEVICE_SECRET "YOUR_DEVICE_SECRET"

#define ALINK_BODY_FORMAT
"{\"id\":\"666\",\"version\":\"1.0\",\"method\":\"%s\",\"params\":%s}"
#define TOPIC_PUB "/" PRODUCT_KEY "/" DEVICE_NAME "/user/pub"
#define TOPIC_SUB "/" PRODUCT_KEY "/" DEVICE_NAME "/user/sub"

unsigned long lastMs = 0;

WiFiClient espClient;
PubSubClient mqttClient(espClient);

void initWifi(const char *ssid, const char *password){
    WiFi.mode(WIFI_STA);
    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED){
        Serial.println("WiFi does not connect, try again ...");
        delay(3000);
    }

    Serial.println("Wifi is connected.");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
```

```
}

void callback(char *topic, byte *payload, unsigned int length){
    Serial.print("Message arrived [");
    Serial.print(topic);
    Serial.print("] ");
    payload[length] = '\0';
    Serial.println((char *)payload);

    if (strstr(topic, TOPIC_SUB)){
        StaticJsonBuffer<100> jsonBuffer;
        JsonObject &root = jsonBuffer.parseObject(payload);
        /*----------------------CUSTOM FUNC--------------------------------*/
        int val = root["data"];
        if (val == 1){
           Serial.println("LED ON...");
           digitalWrite(LED_BUILTIN, LOW);
        }else if (val == 0){
           Serial.println("LED OFF...");
           digitalWrite(LED_BUILTIN, HIGH);
        }else{
           Serial.println("Undefied Command!");
        }
        /*----------------------CUSTOM FUNC--------------------------------*/
        if (!root.success()){
            Serial.println("parseObject() failed");
            return;
        }
    }
}

void mqttReconnect(){
    // while(!connectAliyunMQTT(mqttClient, PRODUCT_KEY, DEVICE_NAME,
DEVICE_SECRET)){
    // };
    while(!mqttClient.connected()){
```

```
        connectAliyunMQTT(mqttClient, PRODUCT_KEY, DEVICE_NAME,
DEVICE_SECRET);
    }
    mqttClient.subscribe(TOPIC_SUB); //----------------------------SUBSCRIBE----------------
------------
    // Serial.print(TOPIC_SUB);
    // Serial.println(" subscribe done");
}

void mqttIntervalPost(unsigned long lastMs){
    char param[32];
    char jsonBuf[128];

    sprintf(param, "{\"LAST_MS\":%lu}", lastMs);
    sprintf(jsonBuf, ALINK_BODY_FORMAT, TOPIC_PUB, param);
    // Serial.println(jsonBuf);
    mqttClient.publish(TOPIC_PUB, jsonBuf); //----------------------------INTERVAL
PUBLISH--------------------------
}

void setup(){
    pinMode(LED_BUILTIN, OUTPUT);
    /* initialize serial for debugging */
    Serial.begin(115200);

    Serial.println("Demo Start");

    initWifi(WIFI_SSID, WIFI_PASSWD);

    mqttClient.setCallback(callback);
}

// the loop function runs over and over again forever
void loop(){
    if (millis() - lastMs >= 5000){
        lastMs = millis();
```

```
            mqttReconnect();
            mqttIntervalPost(lastMs);
        }
        mqttClient.loop();
}
```

通过测试程序，成功实现了远程控制 ESP8266 上的 LED 灯的亮灭，验证了云服务平台以及程序的正确性。

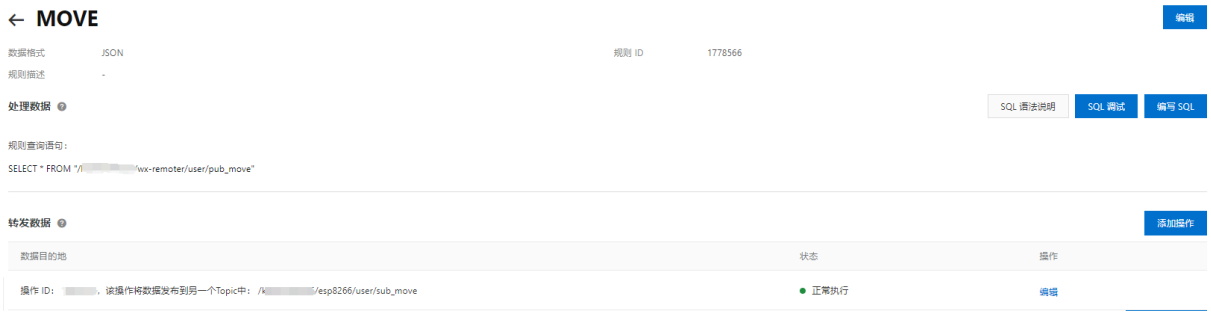
2.3 物联网平台开发

本项目借助阿里云物联网平台实现小车远程控制功能，并使用 MQTT 协议进行通信。大致构建思路如下：

① 、建立设备模型（产品）

② 添加新设备，此处添加小车和控制端，分别命名为 esp8266 和 wx-remoter

③ 定义消息流转规则



2.4 用户端软件开发

鉴于微信在日常生活中的高渗透率以及使用率，本项目拟基于微信小程序实现用户端对清洁小车的远程控制。

用户端的逻辑概述如下：客户端连接到云平台，并且在初始化方法中对相应主题进行订阅；当监听到用户的按钮事件后，调用相应的函数，发送信息到云平台进行转发，实现远程控制。

鉴于篇幅，此处只给出主功能的 JavaScript 脚本：

```
import mqtt from'../../utils/mqtt.js';
const aliyunOpt = require('../../utils/aliyun/aliyun_connect.js');
let that = null;
Page({
    data:{
      client:null,//记录重连的次数
      reconnectCounts:0,//MQTT 连接的配置
      options:{
        protocolVersion: 4, //MQTT 连接协议版本
        clean: false,
        reconnectPeriod: 1000, //1000 毫秒，两次重新连接之间的间隔
        connectTimeout: 30 * 1000, //1000 毫秒，两次重新连接之间的间隔
        resubscribe: true, //如果连接断开并重新连接，则会再次自动订阅已订阅的主题（默认 true）
        clientId: '',
        password: '',
        username: '',
      },
      aliyunInfo: {
        productKey: 'XXX', //阿里云连接的三元组 ，请自己替代为自己的产品信息!!
        deviceName: 'wx-remoter', //阿里云连接的三元组 ，请自己替代为自己的产品信息!!
        deviceSecret: 'XXX', //阿里云连接的三元组 ，请自己替代为自己的产品信息!!
        regionId: 'cn-shanghai', //阿里云连接的三元组 ，请自己替代为自己的产品信息!!
        pubTopicDict: {led:'/XXX/wx-remoter/user/pub',
                       msg:'/XXX/wx-remoter/user/pub_msg',
                       move:'/XXX/wx-remoter/user/pub_move'}, //发布消息的主题
        subTopic: '/XXX/wx-remoter/user/sub', //订阅消息的主题
      },
```

```
        msg_to_edge:'test',
      },
onLoad:function(){
  that = this;
        let clientOpt = aliyunOpt.getAliyunIotMqttClient({
          productKey: that.data.aliyunInfo.productKey,
          deviceName: that.data.aliyunInfo.deviceName,
          deviceSecret: that.data.aliyunInfo.deviceSecret,
          regionId: that.data.aliyunInfo.regionId,
          port: that.data.aliyunInfo.port,
        });

        console.log("get data:" + JSON.stringify(clientOpt));
        let host = 'wxs://' + clientOpt.host;
        console.log("get data:" + JSON.stringify(clientOpt));
        this.setData({
          'options.clientId': clientOpt.clientId,
          'options.password': clientOpt.password,
          'options.username': clientOpt.username,
        })
        console.log("this.data.options host:" + host);
        console.log("this.data.options data:" + JSON.stringify(this.data.options));

        this.data.client = mqtt.connect(host, this.data.options);
        this.data.client.on('connect', ()=>{
          wx.showToast({
            title: '连接成功'
          })
        })
        this.data.client.subscribe(this.data.aliyunInfo.subTopic, function (err) {
          if (!err) {
            console.log("SUBSCRIBE DONE.")
          }else{
            console.log(err)
          }
        })
```

```
        that.data.client.on("message", function (topic, payload) {
          console.log("  收到  topic:" + topic + " , payload :" + payload)
          wx.showModal({
            content: "  收到 topic:[" + topic + "], payload :[" + payload + "]",
            showCancel: false,
          });
        })
        //服务器连接异常的回调
        that.data.client.on("error", function (error) {
          console.log("  服务器  error  的回调" + error)

        })
        //服务器重连连接异常的回调
        that.data.client.on("reconnect", function () {
          console.log("  服务器  reconnect 的回调")

        })
        //服务器连接异常的回调
        that.data.client.on("offline", function () {
          console.log("  服务器 offline 的回调")
        })


    },
onClickOpen() {
    that.sendCommond(1,'led');
},
onClickOff() {
    that.sendCommond(0,'led');
},
msg_local_save(e){
    const val = e.detail.value
    this.setData({
          msg_to_edge: val
```

```
    })
  },
  msg_send(){
    that.sendCommond(this.data.msg_to_edge, 'msg');
  },
  CLICK_FORWARD(){
    that.sendCommond(1,'move');
  },
  CLICK_BACKWARD(){
    that.sendCommond(2,'move');
  },
  CLICK_LEFT(){
    that.sendCommond(3,'move');
  },
  CLICK_RIGHT(){
    that.sendCommond(4,'move');
  },
  sendCommond(data,topic_index) {
    console.log('sending:'+data);
    let sendData = {
      data: data,
    };
    if (this.data.client && this.data.client.connected) {
      this.data.client.publish(this.data.aliyunInfo.pubTopicDict[topic_index],
JSON.stringify(sendData));
    } else {
      wx.showToast({
        title: '请先连接服务器',
        icon: 'none',
        duration: 2000
      })
    }
  }

})
```

通过测试，用户可以通过小程序端的四个方向按钮实现对小车的远程操作，验证了程序的正确性。

2.5 功能整合

将嵌入式软件中的小车控制以及物联网逻辑结合，并且与云平台和微信小程序端联调，最终实现预期功能。除了基础的远程控制小车，还添加了新功能，支持接收用户端下发的消息，并在 LED 显示屏上显示，下面是 ESP8266 烧录的嵌入式软件最终版本：

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>
#include <U8g2lib.h>
#include <Wire.h>
#include <stdlib.h>

#include "aliyun_mqtt.h"

#define SCL 5            // (D1)
#define SDA 4            // (D2)

#define IN1 14          // (D5)
#define IN2 12          // (D6)
#define IN3 13          // (D7)
#define IN4 15          // (D8)

#define WIFI_SSID "XXX" // -------------------------------CHANGE TO YOUR WIFI NAME-
-------------------------------------------
#define WIFI_PASSWD "XXX"// ----------------------------CHANGE TO YOUR WIFI
PASSWORD-----------------------------------------

#define PRODUCT_KEY "XXX"
#define DEVICE_NAME "esp8266"
#define DEVICE_SECRET "XXX"
```

```
#define ALINK_BODY_FORMAT
"{\"id\":\"1024\",\"version\":\"1.0\",\"method\":\"%s\",\"params\":%s}"
#define TOPIC_PUB "/" PRODUCT_KEY "/" DEVICE_NAME "/user/pub"
#define TOPIC_SUB "/" PRODUCT_KEY "/" DEVICE_NAME "/user/sub"
#define TOPIC_SUB_MSG "/" PRODUCT_KEY "/" DEVICE_NAME "/user/sub_msg"
#define TOPIC_SUB_MOVE "/" PRODUCT_KEY "/" DEVICE_NAME "/user/sub_move"

unsigned long lastMs = 0;
char wx_msg[16];
char led_status[4];
char move_status[6];

WiFiClient espClient;
PubSubClient mqttClient(espClient);
U8G2_SSD1306_128X64_NONAME_F_SW_I2C u8g2(U8G2_R0, /*clock=*/SCL,
/*data=*/SDA, /*reset=*/U8X8_PIN_NONE);

void motor_reset(){
   digitalWrite(IN1, LOW);
   digitalWrite(IN2, LOW);
}

void servo_reset(){
   digitalWrite(IN3, LOW);
   digitalWrite(IN4, LOW);
}

void both_forward(){
   digitalWrite(IN1, LOW);
   digitalWrite(IN2, HIGH);
   delay(100);
   motor_reset();
}

void both_backward(){
   digitalWrite(IN1, HIGH);
```

```
    digitalWrite(IN2, LOW);
    delay(100);
    motor_reset();
}

void co_left(){
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, HIGH);
    delay(100);
}

void co_right(){
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);
    delay(100);
}

void initWifi(const char *ssid, const char *password){
        WiFi.mode(WIFI_STA);
        WiFi.begin(ssid, password);
        while (WiFi.status() != WL_CONNECTED){
                Serial.println("WiFi does not connect, try again ...");
                delay(3000);
        }

        Serial.println("Wifi is connected.");
        Serial.println("IP address: ");
        Serial.println(WiFi.localIP());
}

void callback(char *topic, byte *payload, unsigned int length){
        Serial.print("Message arrived [");
        Serial.print(topic);
        Serial.print("] ");
        payload[length] = '\0';
        Serial.println((char *)payload);
```

```
StaticJsonBuffer<100> jsonBuffer;
JsonObject &root = jsonBuffer.parseObject(payload);
if (!root.success()){
      Serial.println("parseObject() failed");
      return;
}
/*----------------------CUSTOM FUNC---------------------------------*/
if (!strcmp(topic, TOPIC_SUB)){
      int val = root["data"];
      if (val == 1){
         // Serial.println("LED ON...");
         digitalWrite(LED_BUILTIN, LOW);
         strcpy(led_status, "ON");
      }else if (val == 0){
         // Serial.println("LED OFF...");
         digitalWrite(LED_BUILTIN, HIGH);
         strcpy(led_status, "OFF");
      }else{
         Serial.println("Undefied LED Command!");
      }
} else if (!strcmp(topic, TOPIC_SUB_MSG)) {
   const char* tmp = root["data"];
   // Serial.println(tmp);
   strcpy(wx_msg,tmp);
   // Serial.println(wx_msg);
} else if (!strcmp(topic, TOPIC_SUB_MOVE)) {
   int move = root["data"];
   if (move == 1){
      both_forward();
      strcpy(move_status, "GO");
   } else if (move == 2){
      both_backward();
      strcpy(move_status, "BACK");
   } else if (move == 3){
      co_left();
```

```
            strcpy(move_status, "LEFT");
        } else if (move == 4){
            co_right();
            strcpy(move_status, "RIGHT");
        } else {
            Serial.println("Undefied Move Command!");
        }
    } else {
        Serial.println("Undefied Topic!");
    }
    /*----------------------CUSTOM FUNC--------------------------------*/
}

void mqttReconnect(){
    // while(!connectAliyunMQTT(mqttClient, PRODUCT_KEY, DEVICE_NAME,
DEVICE_SECRET)){
    // };
    while(!mqttClient.connected()){
        connectAliyunMQTT(mqttClient, PRODUCT_KEY, DEVICE_NAME,
DEVICE_SECRET);
    }
    mqttClient.subscribe(TOPIC_SUB); //----------------------------SUBSCRIBE----------
----------
    mqttClient.subscribe(TOPIC_SUB_MSG);
    mqttClient.subscribe(TOPIC_SUB_MOVE);
    // Serial.print(TOPIC_SUB);
    // Serial.println(" subscribe done");
}

void mqttIntervalPost(unsigned long lastMs){
    char param[32];
    char jsonBuf[128];

    sprintf(param, "{\"LAST_MS\":%lu}", lastMs);
    sprintf(jsonBuf, ALINK_BODY_FORMAT, TOPIC_PUB, param);
    // Serial.println(jsonBuf);
```

```
    mqttClient.publish(TOPIC_PUB, jsonBuf); //----------------------------INTERVAL
PUBLISH--------------------------
}

void setup(){
    u8g2.begin();
    u8g2.enableUTF8Print();

    memset(wx_msg, 0 ,sizeof(wx_msg));
    memset(led_status, 0 ,sizeof(led_status));
    memset(move_status, 0 ,sizeof(move_status));

    pinMode(LED_BUILTIN, OUTPUT);

    // myServo.attach(SERVO_OUT);

    pinMode(IN1, OUTPUT);
    pinMode(IN2, OUTPUT);
    pinMode(IN3, OUTPUT);
    pinMode(IN4, OUTPUT);

    /* initialize serial for debugging */
    Serial.begin(115200);
    Serial.println("Demo Start");

    initWifi(WIFI_SSID, WIFI_PASSWD);

    mqttClient.setCallback(callback);
}

// the loop function runs over and over again forever
void loop(){
    if (millis() - lastMs >= 5000){
        lastMs = millis();
        mqttReconnect();
        // mqttIntervalPost(lastMs);
```

```
    }
    u8g2.setFont(u8g2_font_unifont_t_symbols);
    u8g2.firstPage();
    do {
    u8g2.setCursor(0, 15);
    u8g2.print("LED:");
    u8g2.setCursor(64, 15);
    u8g2.print(led_status);

    u8g2.setCursor(0, 30);
    u8g2.print("MOVE:");
    u8g2.setCursor(64, 30);
    u8g2.print(move_status);

    u8g2.setCursor(0, 45);
    u8g2.print("MSG FROM WX:");
    u8g2.setCursor(0, 60);
    u8g2.print(wx_msg);
    } while (u8g2.nextPage());
    mqttClient.loop();
}
```

3 总结

通过本项目的动手实践，我对物联网与嵌入式的相关技术有了新的认识。尽管自己之前对相关技术有所了解，但是当真正上手做项目的时候发现还是存在一些短板。当亲手通过编写的指令和代码实现了预期功能的时候，那种感觉与在书本上学会如何操作是完全不同的。正如古话所说，“纸上得来终觉浅，绝知此事要躬行”。