文章编号：2096-1472(2020)-08-38-03
DOI:10.19644/j.cnki.issn2096-1472.2020.08.011

基于ESP32平台和MQTT协议的远程控制系统设计

王 浩

(苏州健雄职业技术学院人工智能学院，江苏 太仓 215411)
✉769623995@qq.com

摘 要：随着工业互联网的快速发展，智能化远程控制成为现代工业发展的必然趋势，目前主流的基于TCP/IP网络连接方式是一种MQTT通信协议，它可以通过发布和订阅方式进行数据双向通信，是面向物联网远程通信的轻量级连接协议。本文设计一种基于MQTT通信协议在ESP32硬件平台上的远程控制设计方案，利用Python语言编程实现远程控制功能，并通过MQTT通信协议方式实现远程控制风扇。实验结果表明：该系统数据通信稳定和可靠性强，具有一定的应用前景。

关键词：ESP32；MQTT；Python
中图分类号：TP323 **文献标识码**：A

Design of Remote Control System based on ESP32 Platform and MQTT Protocol

WANG Hao

(*Institute of Artificial Intelligence*, *Suzhou Chien-shiung Institute of Technology*, *Taicang* 215411, *China*)
✉769623995@qq.com

Abstract: With the fast development of the industrial Internet, intelligent remote control has become the inevitable trend of the modern industry development. At present, the main TCP/ IP-based network connection mode is an MQTT (Message Queuing Telemetry Transport) communication protocol. As a lightweight connection protocol for Internet of Things telecommunication, it allows two-way data communication via publishing and subscription. This paper designs a remote control plan on the ESP32 hardware platform based on MQTT communication protocol. It uses Python as the programming language to realize remote control and remote control fans through MQTT communication protocol. The experiment result shows that the data communication of this system is stable with higher reliability, and has a certain application prospect.

Keywords: ESP32; MQTT; Python

1 引言(Introduction)

随着工业互联网技术和无线网络通信技术的迅速发展，智能制造产业对远程智能化控制工业设备的开发和应用不断加大力度，使得对工业嵌入式设备和PC端之间相互通信提出了更高的要求，如果采用原始的socket网络通信，并不能保障数据通信可以准确到达接收方，同时数据的可靠性和实时性也会有一定的影响[1]。为了保障数据信息传递的服务质量，本文提出一种在ESP32硬件平台上基于MQTT通信协议方式，使用Python语言实现远程通信控制风扇设计方案。首先启动PC端MQTT云服务器，然后利用ESP32硬件模块的WIFI功能连接MQTT云服务器，并作为MQTT通信客户端，向MQTT云服务器进行订阅消息，一旦有PC端向MQTT云服务器进行发布消息，就可以实现双方消息的相互推送[2]，并达到远程控制硬件设备。

2 总体设计(Overall design)

为了提高PC终端对智能制造产业中的执行机构实现远程控制的灵活性和可扩展性，本系统以ESP32硬件平台为载体，一方面利用Python语言编程构建MQTT的客户端，通过订阅的通信方式与MQTT云服务器进行数据通信，另一方面PC终端也通过MQTT云服务器进行发布相关信息，这样双方就可以通过云服务器作为中间桥梁，实现远程控制硬件设备，如图1所示系统的整体架构。

基金项目：2020年度江苏省现代教育技术研究课题“基于OBD车联网应用系统在技术技能型人才培养中的应用研究”(2019-R-76578)成果.

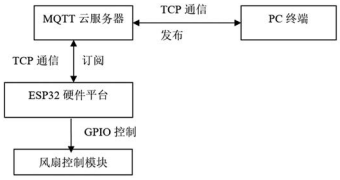

图1 系统整体架构图

Fig.1 Overall architecture of the system

3 系统的硬件设计(Hardware design of the system)

3.1 ESP32平台的硬件设计

ESP32硬件平台中的MCU芯片是一款可作为独立运行应用程序的设备模块，其主要载体可以通过SPI/SDIO或I2C/UART接口提供WiFi和蓝牙功能[3]。另外ESP32模块只需极少的外围器件，即可实现安全可靠数据通信处理功能。本文主要使用GPIO输出功能和WIFI通信的STA客户端模式，通过ESP32模块提供的WiFi功能连接至MQTT云服务器端实现对风扇的远程控制，ESP32硬件平台电路如图2所示。

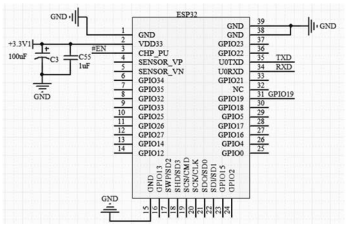

图2 ESP32硬件平台电路连接

Fig.2 Circuit connection of ESP32 hardware platform

3.2 风扇控制硬件设计

为了能够通过ESP32硬件模块驱动大功率的风扇设备运行，需要将ESP32硬件模块的GPIO19引脚连接L9110直流电机驱动芯片，这里L9110芯片是为控制和驱动电机设计的两通道推挽式功率放大专用集成电路器件[4]，该芯片有两个TTL/CMOS兼容电平的输入，具有较大的电流驱动能力，每通道能通过750—800mA的持续电流，它的两个输出端能直接驱动直流风扇电机的运行和停止，风扇控制硬件电路如图3所示。

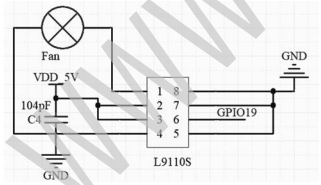

图3 风扇控制硬件电路

Fig.3 Hardware circuit of fan control

4 系统的软件设计(Software design of the system)

4.1 无线连接WiFi功能程序设计

ESP32设备端系统软件部分主要是利用Python语言在VSCode开发平台上进行功能代码编写，实现MQTT通信控制功能[5]，这里包括无线连接WiFi功能、MQTT客户端与MQTT云服务器端数据通信功能。

为了能够让ESP32硬件平台连接WIFI的AP热点，需要将WIFI启动为STA模式，这里首先从JSON配置文件中读取热点名称和密码，如果没有产生配置文件，用户需要从终端输入热点和密码，然后保存，接着连接当前环境AP热点，在连接网络成功之后，显示ESP32设备平台IP、子网掩码、网关和DNS信息，如图4所示ESP32硬件平台连接WiFi功能流程。

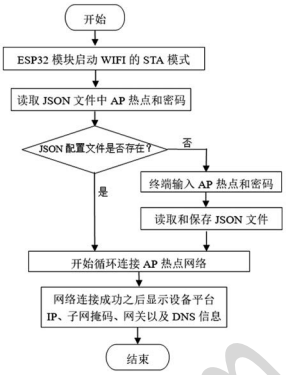

图4 ESP32硬件平台连接WiFi功能流程图

Fig.4 Function flow chart of ESP32 hardware platform connecting WiFi

无线连接WiFi主要功能代码如下：

```
def do_connect():
    import json
    import network
    # 尝试读取配置文件wifi_confi.json,这里以json的方式来存储WIFI配置
    # wifi_config.json在根目录下
    # 若不是初次运行,则将文件中的内容读取并加载到字典变量config
    try:
        with open('wifi_config.json','r') as f:
            config = json.loads(f.read())
    # 若初次运行,则将进入except执行配置文件的创建
    except:
        essid = input('wifi name:') # 输入essid
        password = input('wifi passwrod:') # 输入password
        config = dict(essid=essid, password=password) # 创建字典
        with open('wifi_config.json','w') as f:
            f.write(json.dumps(config)) # 将字典序列化为json字符串,存入wifi_config.json
    #以下为正常的WIFI连接流程
    wifi = network.WLAN(network.STA_IF)
    if not wifi.isconnected():
        print('connecting to network...')
        wifi.active(True)
        wifi.connect(config['essid'], config['password'])
        while not wifi.isconnected():
            pass
    print('network config:', wifi.ifconfig())
if __name__ == '__main__':
    do_connect()
```

4.2 MQTT客户端连接MQTT云服务器功能程序设计

4.2.1 MQTT协议通信流程

MQTT通信协议是一种基于发布和订阅模型的轻量级消息传输网络协议，这个轻量级协议可在设备硬件资源受限、高延迟以及带宽有限的网络上实现。它可以为物联网设备的多样化应用场景提供适当的资源平衡和灵活性服务。另外在基于MQTT协议的IOT网络里面有发布者Publisher负责发布消息，订阅者Subscriber订阅消息，以及MQTT云服务器中转站负责将信息从发布者传递到订阅者[6]。这里在进行MQTT异步消息通信前，需要建立可靠的TCP网络通信连接，整个通信过程大致分成三个部分。

(1)PC端MQTT云服务器开启Server模式

MQTT云服务器是整个网络通信的核心，这里采用Mosquitto作为整个系统MQTT云服务端，当开启Server模式之后，所有MQTT报文都是通过Mosquitto进行管理和转发的，首先ESP32硬件平台利用PC端MQTT云服务器IP地址和端口号作为参数，创建一个订阅信息的MQTT客户端，然后PC端再创建一个可以发布消息的MQTT客户端，这样通过MQTT云服务器的中转站可以实现ESP32设备端和PC端之间信息的订阅和发布。

(2)ESP32设备平台订阅主题过程

当MQTT云服务器开启Server模式之后，ESP32设备平台通过设置PC端MQTT云服务器IP地址和端口号，启动与MQTT云服务端进行长连接，然后向MQTT云服务端订阅TOPIC_ID的主题为Fan_Control，以实现一个MQTT客户端创建。

(3)PC端发布主题

首先PC端上通过MQTT云服务器IP地址和端口号与MQTT云服务端进行长连接，实现PC端的MQTT客户端创建，然后发送数据帧TOPIC_ID+消息指令，这里TOPIC_ID：Fan_Control，MESSAGE：Fan_on或者Fan_off。这样就实现了向MQTT云服务端发布主题。

一旦MQTT云服务端收到发送过来数据帧之后，发现ESP32设备平台订阅了Fan_Control这个主题，立即将数据帧转发至ESP32设备平台，最后根据消息指令Fan_on或者Fan_off实现对风扇的开启和关闭操作，如图5所示系统订阅与发布流程。

图5 系统订阅与发布流程图

Fig.5 Flow chart of system subscription and publishing

4.2.2 ESP32设备平台的MQTT客户端实现

为了使ESP32硬件平台能够创建MQTT客户端对象，并连接MQTT云服务器，实现订阅控制风扇的功能。这里使用Python编程语言在VSCode开发平台上，调用针对ESP32平台的MQTT通信功能库，实现MQTT客户端连接MQTT云服务器订阅功能[7]，主要功能代码如下：

```
from umqtt.simple import MQTTClient
import time
from machine import Pin
fan = Pin(19, Pin.OUT)  #设置控制风扇的GPIO19引脚
SERVER = '云服务器IP地址'
TOPIC = b'Fan_Control'
def mqtt_callback(topic, msg):
    if msg==b"Fan_on":
        fan.value(1) #风扇运行
if msg==b"Fan_off":
    fan.value(0) #风扇停止
def connmqtt():
    client = MQTTClient(CLIENT_ID, SERVER)
    client.set_callback(mqtt_callback)
    client.connect()
    client.subscribe(TOPIC)#订阅主题
```

5 系统测试(System testing)

为了验证ESP32硬件平台和PC端之间通过MQTT云服务器作为中转站，实现双方数据信息的订阅和发布，这里采用第三方Mosquitto作为MQTT云服务器，当双方连接MQTT云服务器成功之后，一旦ESP32硬件平台向MQTT云服务器订阅主题为Fan_Control，如果PC端向MQTT云服务器发布主题信息和控制风扇的命令消息之后，ESP32硬件平台通过订阅主题的信息获得PC端发来的数据帧，最后通过数据帧中包含Fan_on或者Fan_off命令信息实现对风扇设备的远程控制操作，如图6所示基于MQTT客户端运行界面。

图6 基于MQTT客户端运行界面

Fig.6 MQTT-based client running interface

6 结论(Conclusion)

文中采用ESP32硬件平台和PC端通过MQTT通信协议连接MQTT云服务器之后，实现数据双方的相互推送，并利用Python语言编程实现ESP32硬件平台的MQTT数据订阅通信功能，PC端设备通过MQTT发布数据信息实现对ESP硬件平台的风扇设备进行远程控制。实验结果表明：该系统操作方便和通信可靠，具有一定的应用前景。

参考文献(References)

[1] 张玉杰,张海涛,张婷婷.基于MQTT的物联网系统消息发布/订阅方法研究[J]电视技术,2017(Z3):23-25.

[2] 崔自赏,陈冰,艾武,等.基于MQTT协议的物联网电梯监控系统设计[J].电子测量技术,2018(07):15-17.

[3] 范兴隆.ESP8266在智能家居监控系统中的应用[J].单片机与嵌入式系统应用,2016,16(9):47-50.

[4] 王浩.基于Esp8266WIFI平台和MQTT协议的远程设备数据

采集与控制设计[J].泰山学院学报,2017,39(06):86-91.
[5] 姚丹谢.雪松.基于MQTT协议的物联网通信系统的研究与实训[J].信息通信,2016,3(20):33-35.
[6] 蒋鹏,袁嵩.基于MQTT协议的综合消息推送[J].现代计算机,2015,4(16):20-22.
[7]吴俊辉,吴桂初,陈冲,等.基于MQTT协议的物联网网关设计[J].温州大学学报(自然科学版),2019,40(04):54-61.

作者简介：
王　浩(1971-)，男，硕士，副教授.研究领域：物联网工程应用.

(上接第47页)

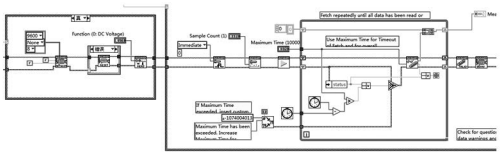

图13 万用表通信协议

Fig.13 Multi-meter communication protocol

4 动力电池测试系统检测(Power battery test system detection)

本文设计的动力电池测试系统中，电池控制部分设计了：连接按钮、串口的选择及连接指示灯。用GPIB配套线缆将上位机与硬件连接起来，在上位机上选择合适的串口通道，点击连接按钮，若是连接成功指示灯被点亮。若是没被点亮，就要检查线缆是否有松动或者损坏，以及调试通信协议。测试参数设置部分设计了：模式选择控件、充/放电截止电压输入控件、额定容量输入控件、充/放电电流输入控件、通信指示灯、是否进行数据保存选项，以及测试按钮。测试开始前，需要将参数设置完毕，再点击测试按钮，在测试过程中，通信指示灯会不断闪烁。监控电池状态部分设计了：电池电压显示控件、电池电流显示控件、倍率显示控件、测试时间显示控件、电池状态显示控件及波形图表，以便于工作人员能实时监测电池状态。其中波形图表曲线1代表电池电压与测试时间的关系，曲线2代表容量与测试时间之间的关系。

测试系统充电模式界面如图14所示，该界面中，电源、万用表、通信指示灯被点亮用来指示连接状态，仪器连接的按钮变为停止输出按钮，开始测试按钮变为停止测试，电池电压随着测试时间的变长而逐渐增加，容量与测试时间呈线性关系的上升。

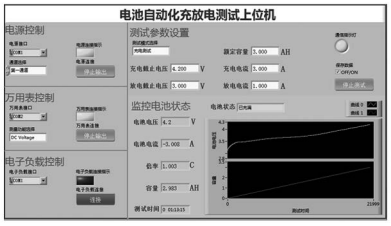

图14 充电模式测试界面

Fig.14 Charging mode test interface

测试系统放电模式界面如图15所示，电子负载与万用表连接完毕，通信指示灯也点亮，电池电压随测试时间的加长而逐渐下降，容量与测试时间表现为线性关系。

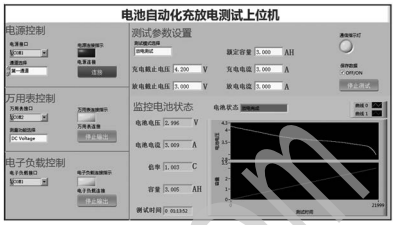

图15 放电模式界面

Fig.15 Discharge mode interface

5 结论(Conclusion)

为了测试电动汽车动力电池特性，本文设计了基于LabVIEW的电动汽车动力电池测试系统，通过人机交互界面实现控制电池的充放电，对电池的电池倍率特性、功率特性和脉冲功率特性等的测量。实验结果表明，该系统能够实现电池重要特性的测试，具有重要的推广价值。

参考文献(References)

[1] Balasingam Balakumar, Pattipati Bharath, Sankavaram Chaitanya, et al. An EM approach for dynamic battery management systems[J]. Information Fusion(FUSION), 2012 15th International Conference on. IEEE, 2012(07): 2110-2117.
[2] 李国洪,资新运,刘鲁源.动力电池综合性能测试系统的开发研制[C].中国汽车工程学会学术年会,2003:1046-1050.
[3] 王宏志,武俊峰.基于LabVIEW的锂离子动力电池内阻测试系统[J].自动化技术与应用,2009(4):80-82.
[4] 王健,陈磊,温小明.基于LabVIEW的锂电池组温度状态的在线测试系统研究[J].电子世界,2018(18):8-10.
[5] 王昕灿,郑燕萍,张林峰,等.基于LabVIEW的动力电池SOC实时估算系统研发[J].电源技术,2017,41(3):374-376.
[6] 李桂娟,张持健,施志刚,等.基于LabVIEW的锂电池SOC预估与参数监测系统[J].传感器与微系统,2018,37(10):69-71.
[7] 陈雨飞,李志扬,朱建新,等.基于LabVIEW的电池管理系统测试平台设计[J].电源技术,2019,43(7):1205-1207.

作者简介：
郁俊伟(1997-)，男，本科生.研究领域：车辆工程.
李旭东(1998-)，男，本科生.研究领域：车辆工程.
赵奉奎(1986-)，男，博士，讲师.研究领域：智能汽车环境感知.本文通讯作者.