《物联网与嵌入式技术》

大作业课题报告

Arduino 平台实现 ESP32 云通讯

学 生 姓 名：张 伟

学科、 专业：机械工程

学　　号：22304142

完 成 日 期：2024.05.05

大连理工大学

Dalian University of Technology

目　　录

1 介绍

ESP32 是一款 Wi-Fi 和蓝牙系统级芯片（SoC），具有行业领先的射频性能、低功耗和高度集成的优势。ESP32 集成了完整的发射/接收射频功能，包括天线开关，射频 balun，功率放大器，低噪放大器，过滤器，电源管理模块和先进的自校准电路。自校准电路实现了动态自动调整以消除外部电路的缺陷。

1.1 ESP32

ESP32 是 ESP8266 的升级版本，是一款低成本、低功耗的片上系统微控制器。除了 Wi-Fi 模块外，ESP32 还包含蓝牙 4.0 模块。双核 CPU 的运行频率为 80 至 240MHz。它包含两个 Wi-Fi 和蓝牙模块以及各种输入和输出引脚。ESP32 是物联网项目的理想选择。

1.2 MQTT

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。MQTT 协议当前版本为，2014 年发布的 MQTT v3.1.1。除标准版外，还有一个简化版 MQTT-SN，该协议主要针对嵌入式设备，这些设备一般工作于 TCP/IP 网络，如：ZigBee。

MQTT 与 HTTP 一样，MQTT 运行在传输控制协议/互联网协议(TCP/IP)堆栈之上。

1.3 ESP32 与 MQTT 通讯特点

MQTT 是一种轻量级消息传递协议，针对 ESP32 和 Wi-Fi 等受限设备和网络进行了优化，因此对功耗和带宽的影响最小。MQTT 支持不同级别的可靠性和服务质量，以匹配 ESP32 的功能。这种灵活性使其即使在网络不稳定的情况下也适合使用。

ESP32 和 MQTT 在物联网应用中广泛使用，使其能够很好地集成到物联网解决方案中。MQTT 协议还旨在简化与云平台的集成，以实现跨网络的设备控制和数据监控。

总体而言，ESP32 和 MQTT 的组合非常适合需要在许多设备之间进行无线连接和高效消息传递的物联网应用。

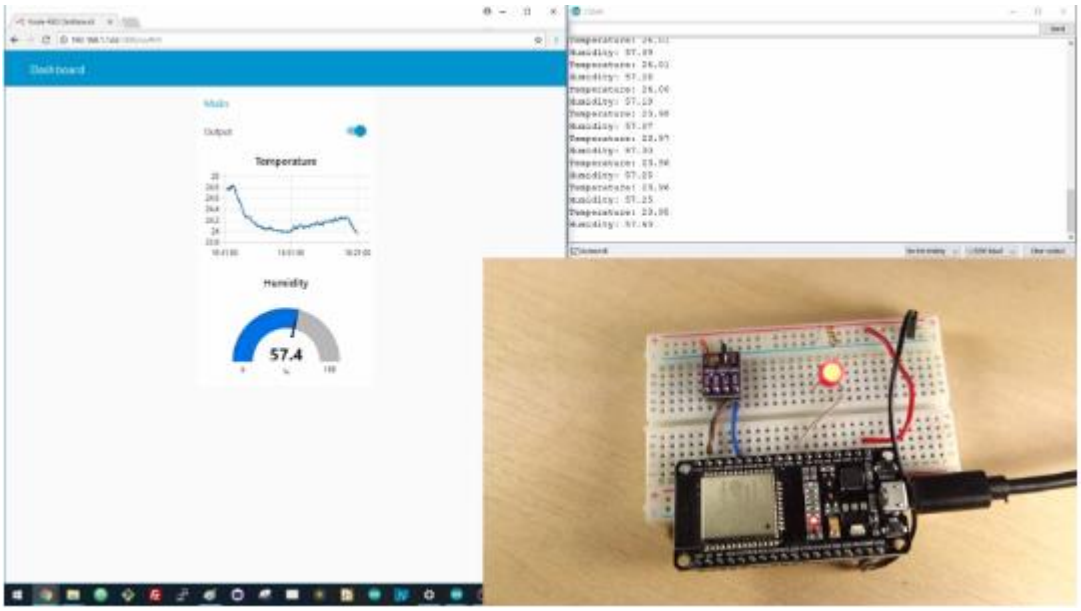

图 1.1　使用 esp32 与 MQTT 通讯上传数据

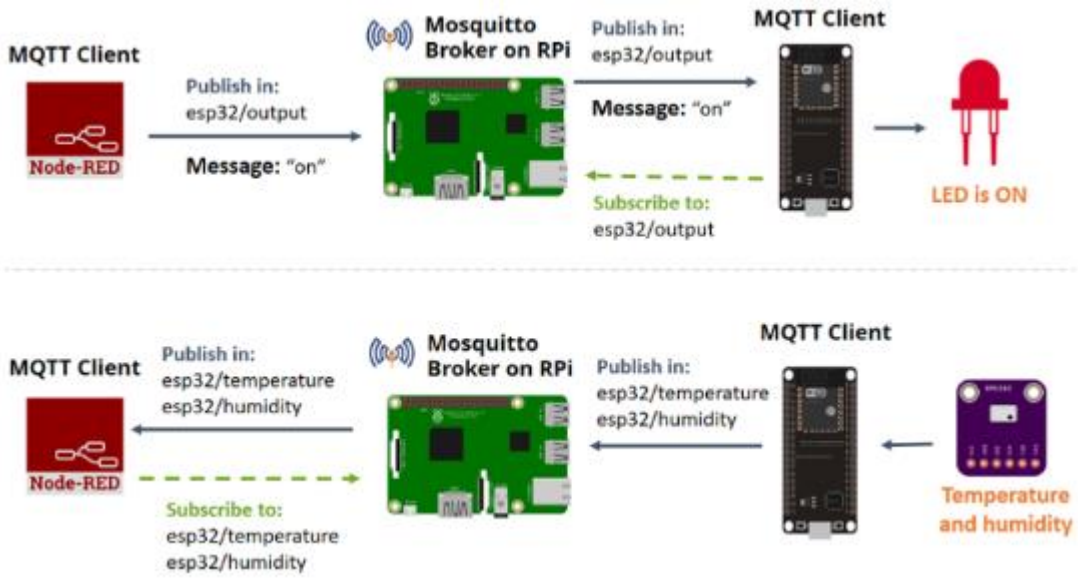

图 1.1　使用 esp32 与 MQTT 通讯流程

2 基于 Arduino 实现 ESP32 与 MQTT 通讯

2.1 Arduino 配置

Arduino 是一个易于使用的硬件和软件的开源电子平台。它适用于任何制作交互式项目的人。Arduino 板可以读取输入（传感器上的灯、按钮信号）并将其转换为输出（激活电机、点亮 LED 或在线发布内容）。

在本项目中，我们将 ESP32 模块连接到笔记本电脑。Arduino 将处理向 ESP32 上传代码，并在 ESP32 和我们的笔记本电脑之间提供串行连接。

2.1.1 安装 ESP32 开发板

ESP32 凭借其集成的 Wi-Fi 和蓝牙功能、用于与外部组件连接的 GPIO 引脚以及与 Arduino IDE 的兼容性，ESP32 开发板可实现基于 MQTT 的物联网应用的无缝连接、原型设计和测试。

ESP32 开发板需要在 Arduino IDE 的开发板管理中寻找并安装

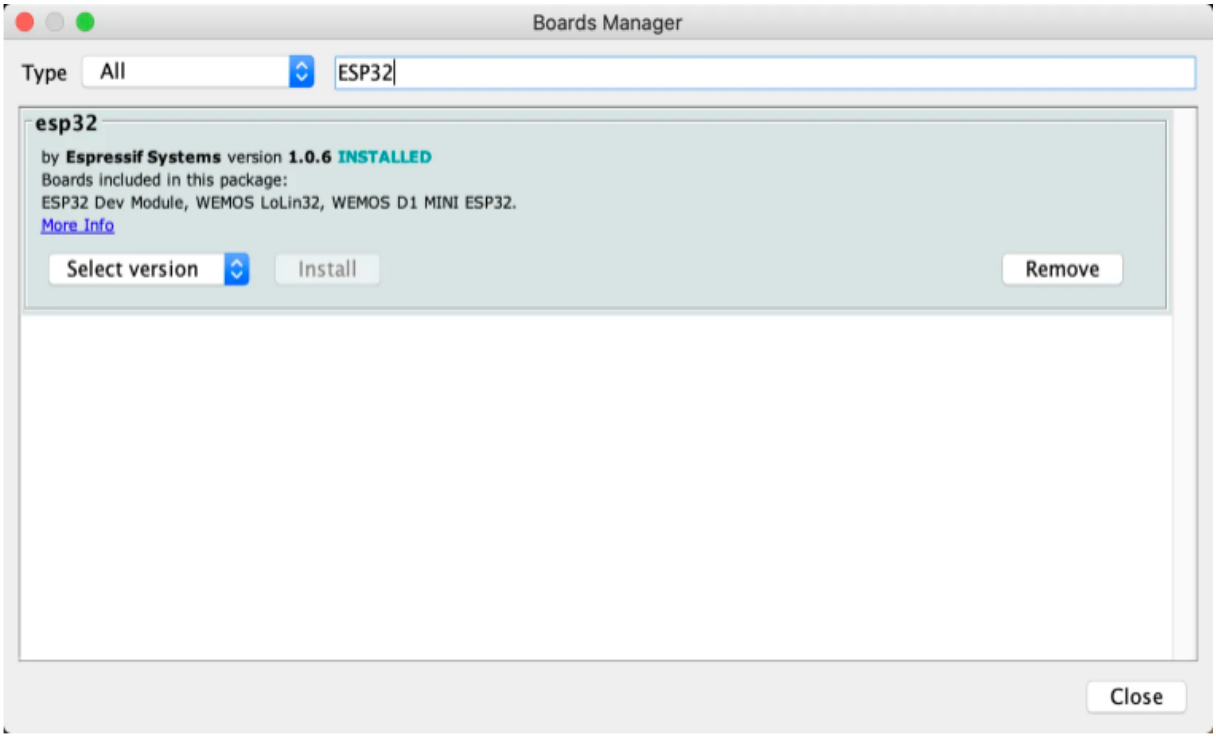

图 2.1 Arduino IDE 中安装 ESP32 开发板

2.1.2 安装 PubSubClient

PubSubClient 由 Nick O'Leary 开发，是一个轻量级 MQTT 客户端库，专为基于 Arduino 的项目而设计。它提供了一个用于简单发布/订阅消息传递的客户端以及支持 MQTT 的服务器。该库简化了 MQTT 通信，并在基于 Arduino 的 IoT 应用程序中实现高效的数据交换。

在 Arduino IDE 中的库管理器寻找 PubSubClient 库进行安装。

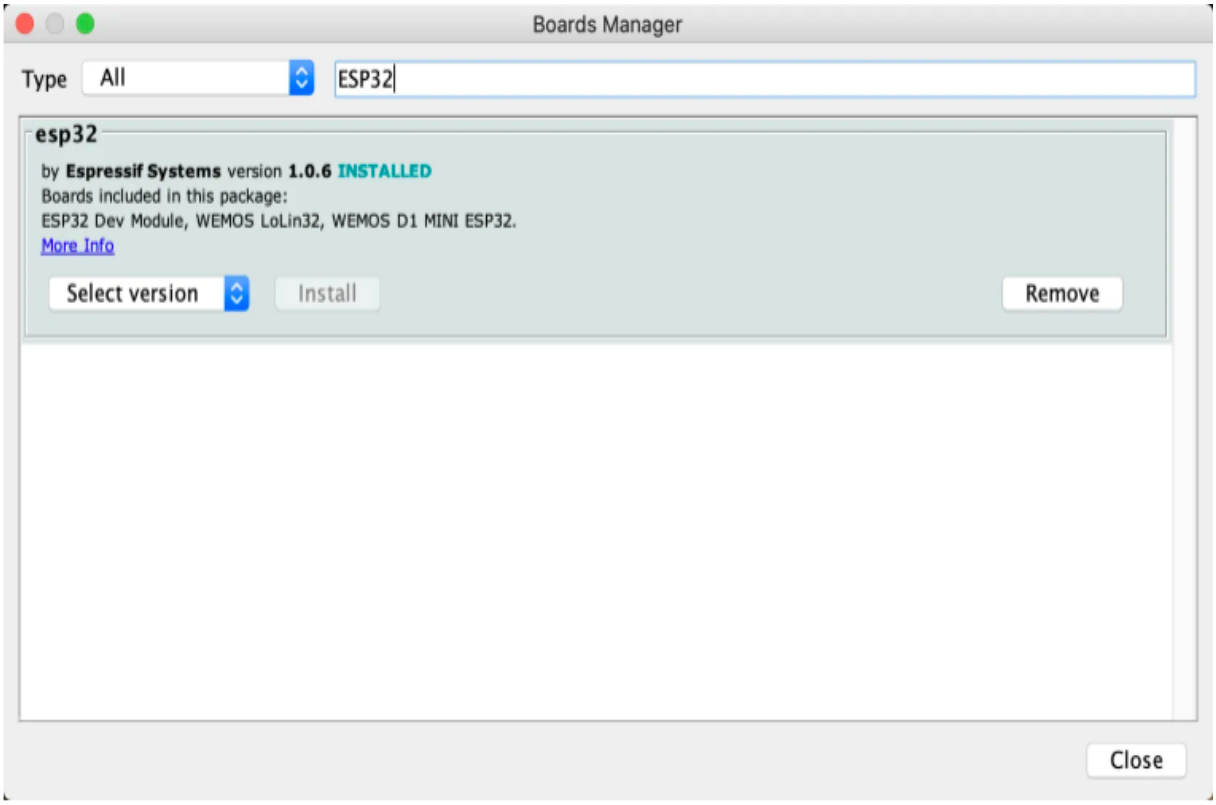

图 2.1　Arduino IDE 中安装 PubSubClient 库

2.2　创建 MQTT 连接

2.2.1　TCP 连接

在 Arduino IDE，导入 WiFi 和 PubSubClient 库。WiFi 库允许 ESP32 与 Wi-Fi 网络建立连接，而 PubSubClient 库允许 ESP32 连接到 MQTT 代理以发布消息和订阅主题。

首先配置 Wi-Fi 网络名称和密码、MQTT 代理地址和端口以及主题。打开串行连接以显示程序结果并建立与 Wi-Fi 网络的连接。然后利用 PubSubClient 与 MQTT 代理建立连接。代码如下：

```
#include <WiFi.h>
#include <PubSubClient.h>
// WiFi
const char *ssid = "xxxxx"; // Enter your WiFi name
const char *password = "xxxxx";  // Enter WiFi password

// MQTT Broker
const char *mqtt_broker = "broker.emqx.io";
const char *topic = "emqx/esp32";
const char *mqtt_username = "emqx";
const char *mqtt_password = "public";
const int mqtt_port = 1883;
// Set software serial baud to 115200;
Serial.begin(115200);
// Connecting to a Wi-Fi network
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.println("Connecting to WiFi..");
}
client.setServer(mqtt_broker, mqtt_port);
client.setCallback(callback);
while (!client.connected()) {
    String client_id = "esp32-client-";
    client_id += String(WiFi.macAddress());
    Serial.printf("The client %s connects to the public MQTT broker\n", client_id.c_str());
    if (client.connect(client_id.c_str(), mqtt_username, mqtt_password)) {
        Serial.println("Public EMQX MQTT broker connected");
    } else {
        Serial.print("failed with state ");
        Serial.print(client.state());
        delay(2000);
    }
}
```

2.2.2 传输层安全/SSL

在 MQTT 中需要使用 TLS 来保证信息的机密性和完整性，防止信息泄露和篡改。

使用服务器根 CA 证书建立安全的 Wi-Fi 连接。ca_cert 变量包含 PEM 格式的根 CA 证书。使用该函数使用服务器根 CA 证书配置 espClient 对象 setCACert()。此设置使 ESP32 客户端能够建立安全的 Wi-Fi 连接并确保传输数据的机密性和完整性。

2.2.3 发布消息并订阅

一旦与 MQTT 服务器的连接成功建立，ESP32 就会向该主题发布消息，然后订阅该主题。代码如下：

```
// publish and subscribe
client.publish(topic, "Hi, I'm ESP32 ^^");
client.subscribe(topic);
```

2.2.4 接收 MQTT 消息

设置回调函数将主题名称打印到串口，并打印从主题接收到的消息，代码如下：

```
void callback(char *topic, byte *payload, unsigned int length) {
    Serial.print("Message arrived in topic: ");
    Serial.println(topic);
    Serial.print("Message:");
    for (int i = 0; i < length; i++) {
        Serial.print((char) payload[i]);
    }
    Serial.println();
    Serial.println("-----------------------");
}
```

3　运行和测试

使用 Arduino 上传完整代码并给 ESP32 上电，打开串口监视器，将波特率设置为 115200，然后通过监视串口监视器的输出来检查 ESP32 的连接状态。使用 MQTTX 客户端与 MQTT Broker 建立连接并向 Hi, I'm MQTTX ESP32 等发布消息。

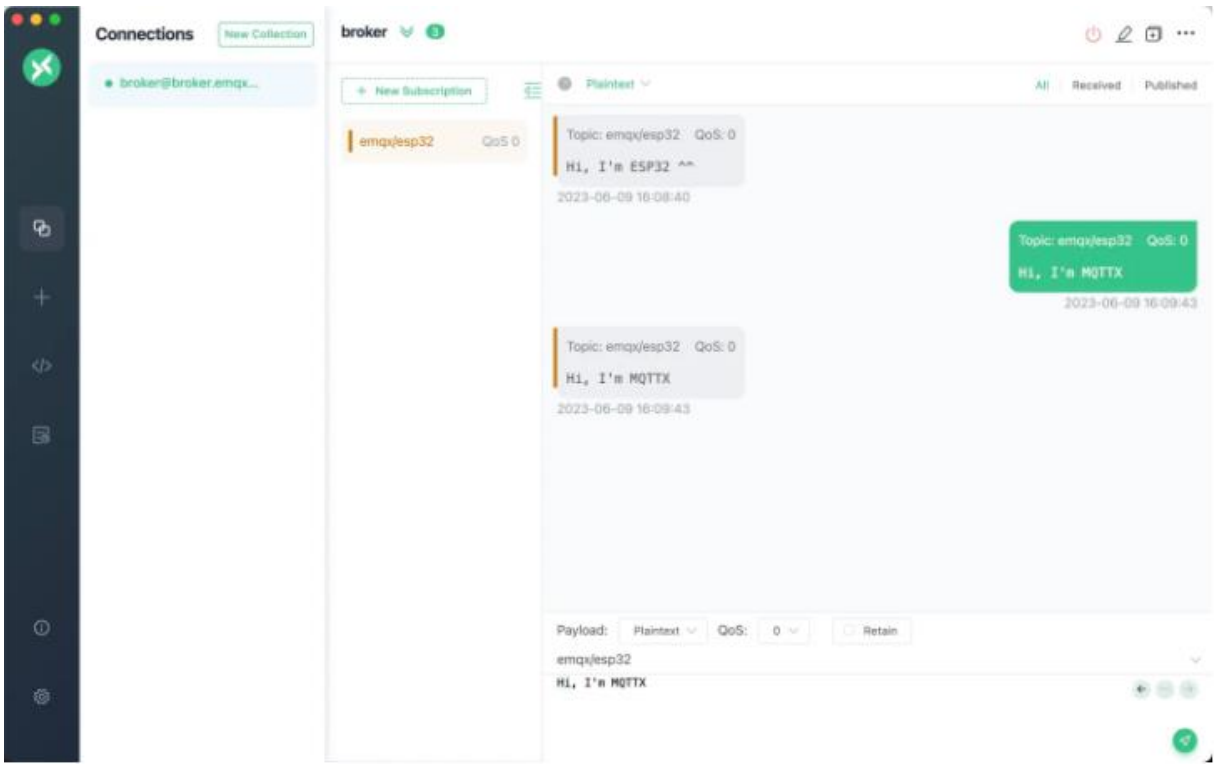

图 3.1　通讯实现