BANGLADESH DIGITAL UNIVERSITY

Lab Report

Course Title: IoT Architecture and Technologies Sessional

Course Code:IRE 212

Submitted By:

Name:Takia Yasmin

Id:2201049

Department Of IRE

Session:2022-2023

Submitted to:

Md. Saurav Chandra Das

(Lecturer)

Department of IRE

Gazipur Digital University.

Submission Date:18 march 2025

MQTT Lab problem solution

Abstract

This experiment demonstrates how an ESP32 microcontroller communicates with an MQTT broker using the Wokwi simulator. The ESP32 connects to a WiFi network, publishes messages to an MQTT topic, and subscribes to receive messages. The experiment was conducted using the Wokwi simulator and MQTTX as the MQTT client. The results show successful communication between the ESP32 and the MQTT broker, with messages being sent and received as expected.

Introduction

The purpose of this experiment is to explore MQTT (Message Queuing Telemetry Transport) communication using an ESP32 microcontroller in the Wokwi simulator. MQTT is a lightweight messaging protocol widely used in IoT applications for device communication. The ESP32 is a popular microcontroller with built-in WiFi capabilities, making it ideal for IoT projects. Wokwi is an online simulator that allows testing and debugging of microcontroller code without physical hardware.

Materials and Methods

Materials:

- Wokwi simulator (https://wokwi.com/)
- MQTTX (MQTT client software)
- Arduino IDE (for code editing)

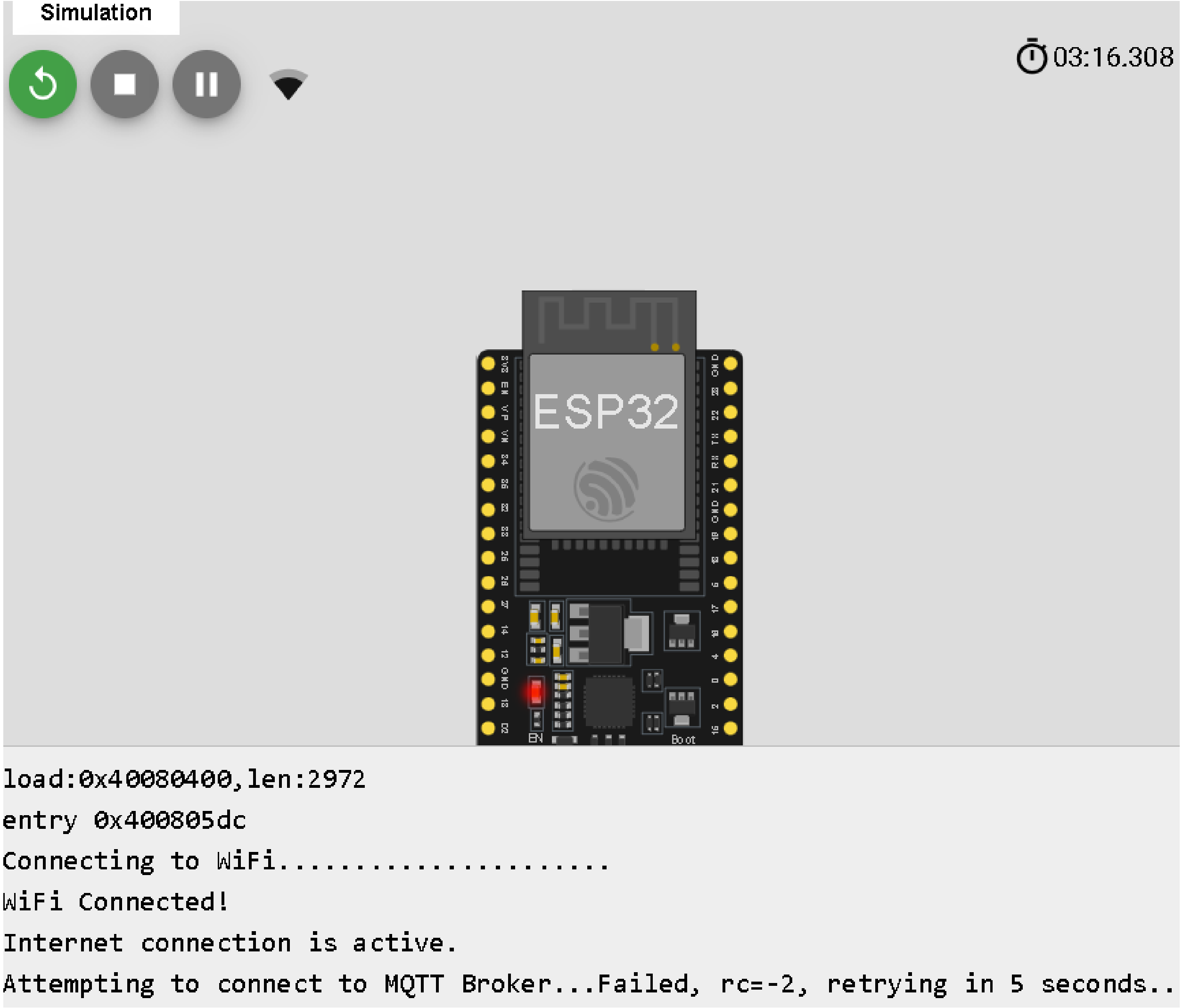
Procedure:

1. Open the Wokwi simulator and create a new ESP32 project.
2. Write the code to connect the ESP32 to the Wokwi-GUEST WiFi network.
3. Configure the ESP32 to connect to the MQTT broker (test.mosquitto.org on port 1883).
4. Set up the ESP32 to publish a message (Hello from ESP32 in Wokwi!) to the topic wokwi/esp32/demo every 5 seconds.
5. Use MQTTX to subscribe to the topic and send test messages.
6. Observe the Serial Monitor in Wokwi for outputs.

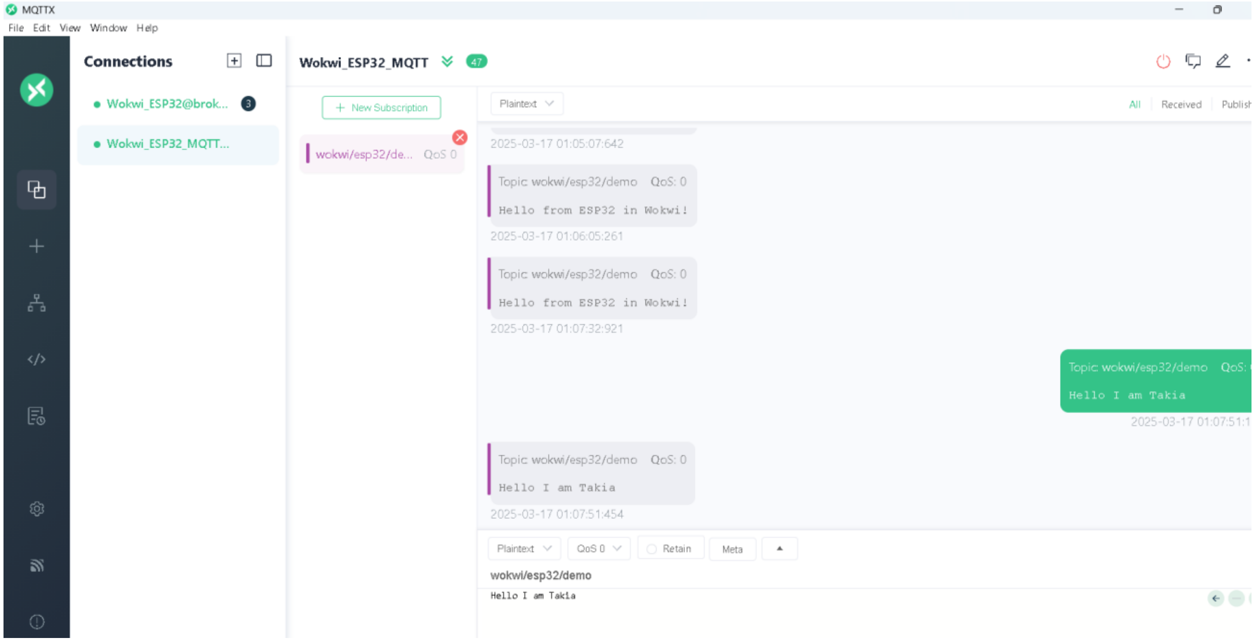
CODE:

```
#include <WiFi.h>
```

```
#include <PubSubClient.h>

const char* ssid = "Wokwi-GUEST";
const char* password = "";

const char* mqtt_broker = "test.mosquitto.org";
const int mqtt_port = 1883;
const char* topic = "wokwi/esp32/demo";

const char* mqtt_username = "Takia";
const char* mqtt_password = "2532467";

WiFiClient espClient;
PubSubClient client(espClient);

void connectToMqtt() {
  while (!client.connected()) {
    Serial.print("Attempting to connect to MQTT Broker...");
    String clientId = "ESP32_Client-" + String(random(0xffff), HEX);
    if (client.connect(clientId.c_str(), mqtt_username, mqtt_password)) {
      Serial.println("Connected!");
      client.subscribe(topic);
    } else {
      Serial.print("Failed, rc=");
      Serial.print(client.state());
      Serial.println(", retrying in 5 seconds...");
      delay(5000);
    }
  }
}

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

void setup() {
  Serial.begin(115200);

  WiFi.begin(ssid, password);
  Serial.print("Connecting to WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }
  Serial.println("\nWiFi Connected!");

  if (WiFi.status() == WL_CONNECTED) {
    Serial.println("Internet connection is active.");
  } else {
    Serial.println("No internet connection.");
  }

  client.setServer(mqtt_broker, mqtt_port);
  client.setCallback(callback);
  connectToMqtt();
}

void loop() {
  if (!client.connected()) {
    connectToMqtt();
  }
  client.loop();

  static unsigned long lastTime = 0;
  if (millis() - lastTime > 5000) {
    lastTime = millis();
    client.publish(topic, "Hello from ESP32 in Wokwi!");
    Serial.println("Message Published!");
  }
}
```



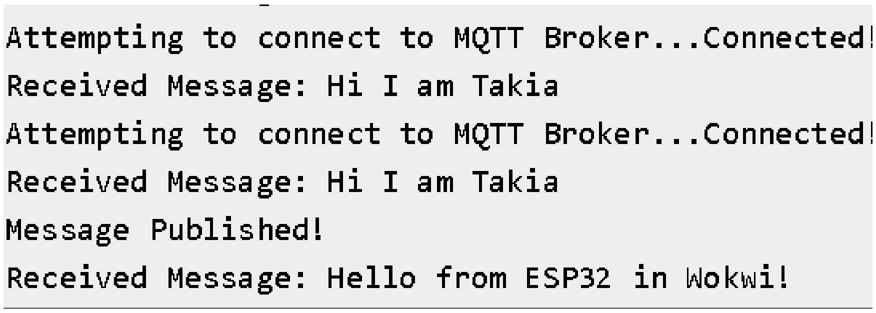
```
load:0x40080400,len:2972
entry 0x400805dc
Connecting to WiFi.....................
WiFi Connected!
Internet connection is active.
Attempting to connect to MQTT Broker...Failed, rc=-2, retrying in 5 seconds..
```



```
2025-03-17 01:05:07:642

Topic: wokwi/esp32/demo   QoS: 0
Hello from ESP32 in Wokwi!
2025-03-17 01:06:05:261

Topic: wokwi/esp32/demo   QoS: 0
Hello from ESP32 in Wokwi!
2025-03-17 01:07:32:921

Topic: wokwi/esp32/demo   QoS:
Hello I am Takia
2025-03-17 01:07:51:1

Topic: wokwi/esp32/demo   QoS: 0
Hello I am Takia
2025-03-17 01:07:51:454
```

wokwi/esp32/demo

```
Hello I am Takia
```



```
Attempting to connect to MQTT Broker...Connected!
Received Message: Hi I am Takia
Attempting to connect to MQTT Broker...Connected!
Received Message: Hi I am Takia
Message Published!
Received Message: Hello from ESP32 in Wokwi!
```

Explanation of How the Message is Sent

1. **WiFi Connection**:

   ➢ The ESP32 first connects to the WiFi network using the provided SSID (Wokwi-GUEST) and password (blank for Wokwi).

   ➢ Once connected, it prints WiFi Connected! in the Serial Monitor.

2. **MQTT Broker Connection**:

   ➢ After connecting to WiFi, the ESP32 attempts to connect to the MQTT broker (test.mosquitto.org on port 1883).

   ➢ If the connection is successful, it prints Connected! and subscribes to the topic wokwi/esp32/demo.

3. **Message Publishing**:

    - Every 5 seconds, the ESP32 publishes a message (Hello from ESP32 in Wokwi!) to the topic wokwi/esp32/demo.
    - It also prints Message Published! in the Serial Monitor.

4. **Message Receiving**:
    - If the ESP32 receives a message on the subscribed topic, it prints the message in the Serial Monitor.

Results

The ESP32 successfully connected to the Wokwi-GUEST WiFi network and the MQTT broker. The following outputs were observed in the Serial Monitor:

- ✓ WiFi Connected!
- ✓ Connected! (to MQTT broker)
- ✓ Message Published! (every 5 seconds)

When a message (Hello from MQTTX!) was sent from MQTTX, the ESP32 printed:

- Received Message: Hello from MQTTX!

Short Video:

2025-03-17 00-48-29.mkv