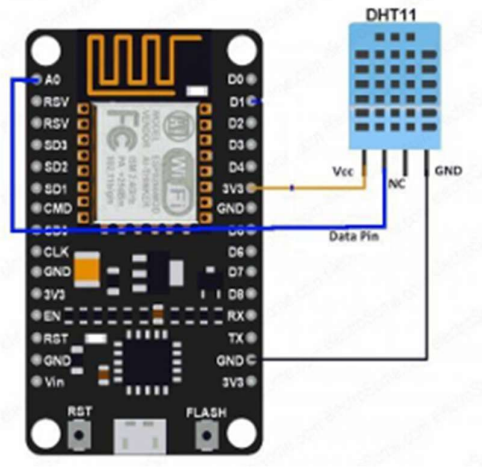
Ex.No 5

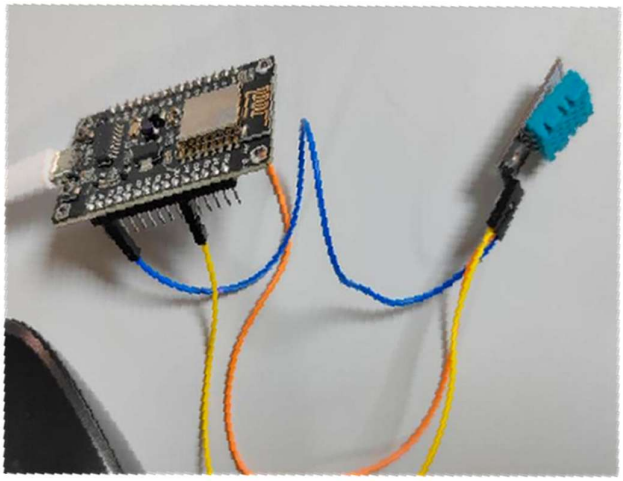
Temperature Warning Alarm

Akhil P S(24MCS1018)

Aim: To integrate ESP8266 microcontroller with MQTT protocol for temperature monitoring and LED control, utilizing the MQTT Explorer tool for visualization and interaction.

Circuit Diagram and Implementation:





Code:

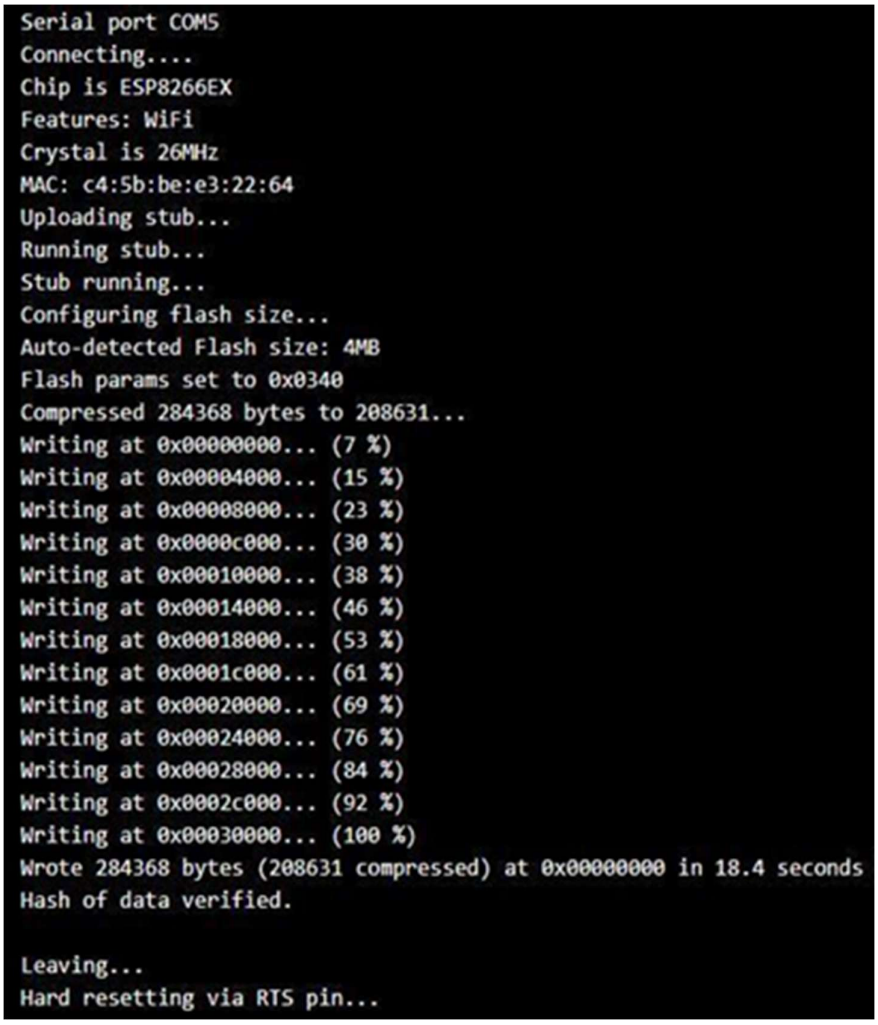
```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

#define MSG_BUFFER_SIZE 50
char msg[MSG_BUFFER_SIZE];

// WiFi Credentials
const char *ssid = "srujan";
const char *password = "srujan2003";

// MQTT Broker Settings
const char *mqtt_broker = "broker.emqx.io";
const char *mqtt_topic = "emqx/esp8266/led";
const char *mqtt_username = "emqx";
const char *mqtt_password = "public";
const int mqtt_port = 1883;

// Global Variables
unsigned long lastMsg = 0;
WiFiClient espClient;
PubSubClient mqtt_client(espClient);

// WiFi Connection
void connectToWiFi() {
    WiFi.begin(ssid, password);
    Serial.print("Connecting to WiFi");
    while (WiFi.status() != WL_CONNECTED) {
```

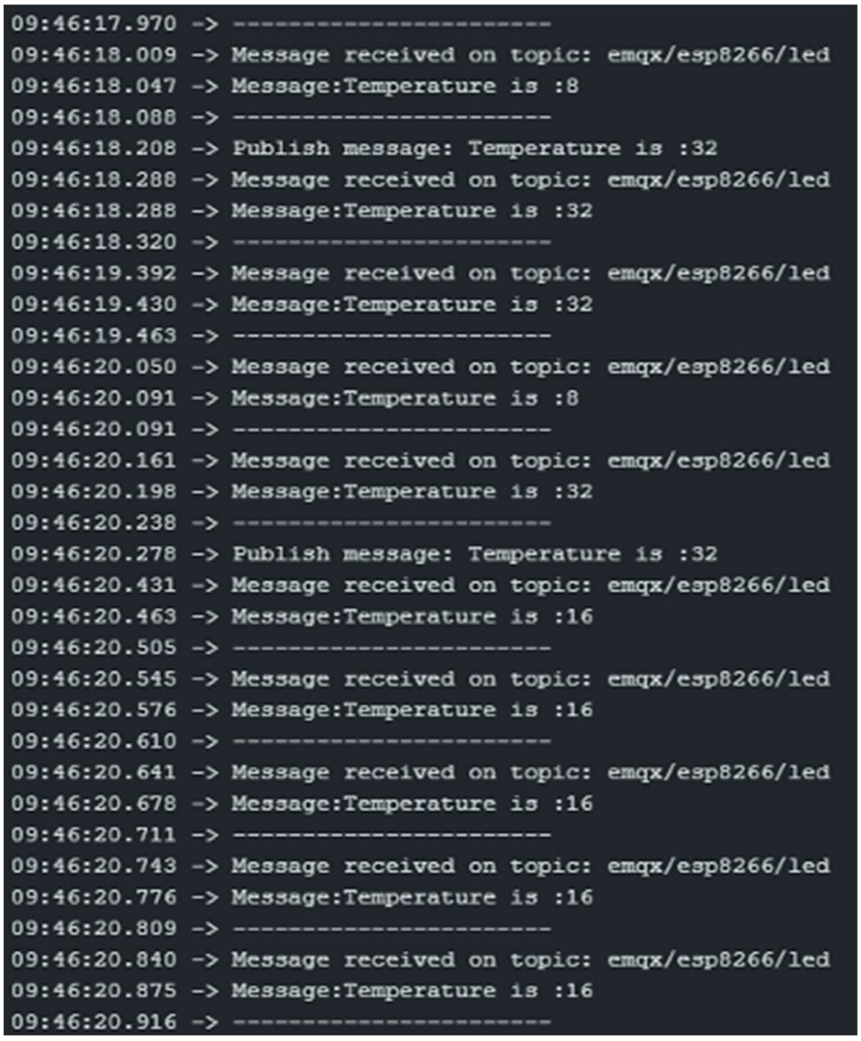
```
        delay(500);
        Serial.print(".");
    }
    Serial.println("\nConnected to WiFi");
}

// MQTT Connection
void connectToMQTTBroker() {
    while (!mqtt_client.connected()) {
        String client_id = "esp8266-client-" + String(WiFi.macAddress());
        Serial.printf("Connecting to MQTT Broker as %s.....\n", client_id.c_str());

        if (mqtt_client.connect(client_id.c_str(), mqtt_username, mqtt_password)) {
            Serial.println("Connected to MQTT broker");
            mqtt_client.subscribe(mqtt_topic);
            mqtt_client.publish(mqtt_topic, "Hi EMQX, I'm ESP8266 ^^");
        } else {
            Serial.printf("Failed to connect, rc=%d. Retrying in 5 seconds...\n",
mqtt_client.state());
            delay(5000);
        }
    }
}

// MQTT Callback Function
void mqttCallback(char *topic, byte *payload, unsigned int length) {
    Serial.printf("Message received on topic: %s\n", topic);
    Serial.print("Message: ");
    for (unsigned int i = 0; i < length; i++) {
```

```
        Serial.print((char)payload[i]);
    }
    Serial.println("\n-----------------------");
}

// Setup Function
void setup() {
    Serial.begin(9600);
    connectToWiFi();
    mqtt_client.setServer(mqtt_broker, mqtt_port);
    mqtt_client.setCallback(mqttCallback);
    connectToMQTTBroker();
}

// Main Loop
void loop() {
    if (!mqtt_client.connected()) {
        connectToMQTTBroker();
    }
    mqtt_client.loop();

    unsigned long now = millis();
    if (now - lastMsg > 2000) {
        lastMsg = now;

        int value = analogRead(A0) * 0.032; // Read temperature sensor on A0
        snprintf(msg, MSG_BUFFER_SIZE, "Temperature: %d", value);
        Serial.printf("Publishing message: %s\n", msg);
        mqtt_client.publish(mqtt_topic, msg);
```

```
  }
}
```

Output:


Serial port COM5
Connecting....
Chip is ESP8266EX
Features: WiFi
Crystal is 26MHz
MAC: c4:5b:be:e3:22:64
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Auto-detected Flash size: 4MB
Flash params set to 0x0340
Compressed 284368 bytes to 208631...
Writing at 0x00000000... (7 %)
Writing at 0x00004000... (15 %)
Writing at 0x00008000... (23 %)
Writing at 0x0000c000... (30 %)
Writing at 0x00010000... (38 %)
Writing at 0x00014000... (46 %)
Writing at 0x00018000... (53 %)
Writing at 0x0001c000... (61 %)
Writing at 0x00020000... (69 %)
Writing at 0x00024000... (76 %)
Writing at 0x00028000... (84 %)
Writing at 0x0002c000... (92 %)
Writing at 0x00030000... (100 %)
Wrote 284368 bytes (208631 compressed) at 0x00000000 in 18.4 seconds
Hash of data verified.
Leaving...
Hard resetting via RTS pin...



Conclusion:

The integration of the ESP8266 microcontroller with the MQTT protocol for temperature monitoring and LED control, using the MQTT Explorer tool for visualization and interaction, was successfully carried out and output is recorded.