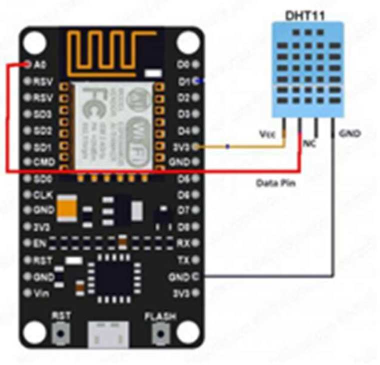
Ex.No 7

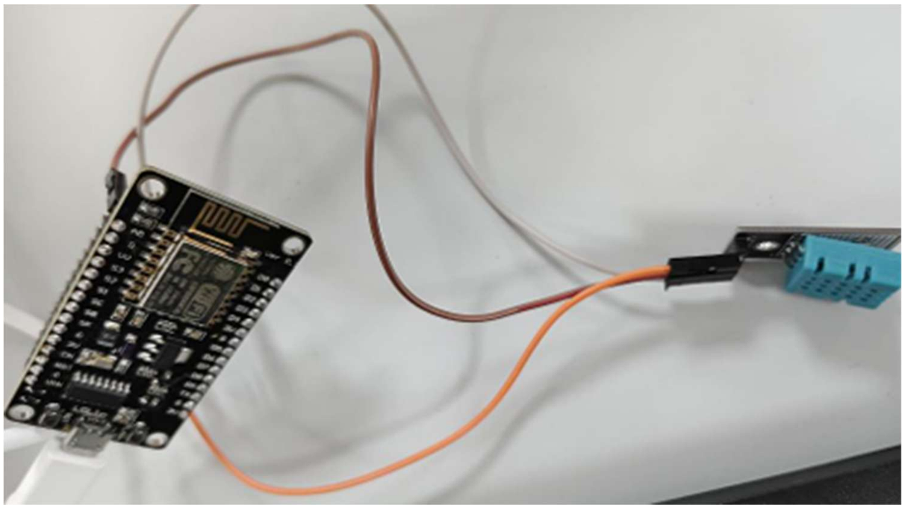
Study and implement COAP Protocol using Arduino IDE

Akhil P S(24MCS1018)

Aim: To integrate ESP8266 microcontroller with COAP protocol for temperature monitoring utilizing node-red to listen to COAP messages

Circuit Diagram and Implementation:





Components:

● ESP8266 Wi-Fi Module.

- DHT11 Temperature and Humidity Sensor.

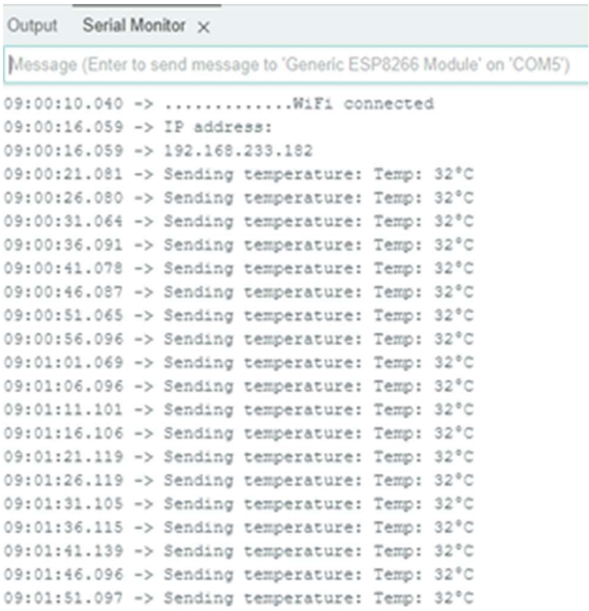
Code:

```
#include <ESP8266WiFi.h>
#include <WiFiUdp.h>
#include <coap-simple.h>

// WiFi Credentials
const char* ssid = "svt"; // Replace with WiFi SSID
const char* password = "123456789"; // Replace with WiFi Password

// Node-RED Server IP (Replace with actual PC IP)
const IPAddress nodeRedServerIp(192, 168, 1, 5);

// UDP & CoAP Setup
WiFiUDP udp;
Coap coap(udp);

// Function Declarations
void callback_temperature(CoapPacket Cpacket, IPAddress ip, int port);
void send_temperature(IPAddress ip, int port);

// WiFi Connection Setup
void connectToWiFi() {
  Serial.print("Connecting to WiFi");
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
```

```
        Serial.print(".");
    }
    Serial.println("\nWiFi Connected");
    Serial.print("IP Address: ");
    Serial.println(WiFi.localIP());
}

// Setup Function
void setup() {
    Serial.begin(115200);
    connectToWiFi();
    pinMode(A0, INPUT); // Setup the temperature sensor pin
    coap.server(callback_temperature, "temp");
    coap.start();
}

// Main Loop
void loop() {
    delay(1000);
    coap.loop();
    send_temperature(nodeRedServerIp, 5683); // Send temperature regularly
}

// CoAP Callback Function
void callback_temperature(CoapPacket Cpacket, IPAddress ip, int port) {
    send_temperature(ip, port); // Send temperature upon request
}

// Send Temperature Data
```

```
void send_temperature(IPAddress ip, int port) {
  int value = analogRead(A0) * 0.032; // Example conversion, adjust as needed
  char tempStr[16];
  snprintf(tempStr, sizeof(tempStr), "Temp: %d°C", value);

  Serial.print("Sending Temperature: ");
  Serial.println(tempStr);
  coap.sendResponse(ip, port, 0, tempStr); // Send response
}
```

Output:


Output
Serial Monitor ×
Message (Enter to send message to 'Generic ESP8266 Module' on 'COM5')

```
09:00:10.040 -> .............WiFi connected
09:00:16.059 -> IP address:
09:00:16.059 -> 192.168.233.182
09:00:21.081 -> Sending temperature: Temp: 32°C
09:00:26.080 -> Sending temperature: Temp: 32°C
09:00:31.064 -> Sending temperature: Temp: 32°C
09:00:36.091 -> Sending temperature: Temp: 32°C
09:00:41.078 -> Sending temperature: Temp: 32°C
09:00:46.087 -> Sending temperature: Temp: 32°C
09:00:51.065 -> Sending temperature: Temp: 32°C
09:00:56.096 -> Sending temperature: Temp: 32°C
09:01:01.069 -> Sending temperature: Temp: 32°C
09:01:06.096 -> Sending temperature: Temp: 32°C
09:01:11.101 -> Sending temperature: Temp: 32°C
09:01:16.106 -> Sending temperature: Temp: 32°C
09:01:21.119 -> Sending temperature: Temp: 32°C
09:01:26.119 -> Sending temperature: Temp: 32°C
09:01:31.105 -> Sending temperature: Temp: 32°C
09:01:36.115 -> Sending temperature: Temp: 32°C
09:01:41.139 -> Sending temperature: Temp: 32°C
09:01:46.096 -> Sending temperature: Temp: 32°C
09:01:51.097 -> Sending temperature: Temp: 32°C
```

Conclusion:

The integration of the ESP8266 microcontroller with the CoAP protocol for temperature monitoring, utilizing Node-RED to receive and process CoAP messages, was successfully implemented. and output is recorded.