به نام خدا

درس : آزمایشگاه سخت افزار

استاد : دکتر اجلالی

گزارش پیشرفت

اعضای گروه :

محمدمعین صمدی آزاد ۴۰۰۱۰۵۰۹۳

امیرحسین عزیزی ۴۰۰۱۰۵۱۲۲

محمدشایان شعبانی ۴۰۰۱۰۵۰۶۹

تابستان ۱۴۰۳

مقدمه:

در قسمت قبل، با اتصال سنسور ها و عملگر ها و توانستیم اطلاعات سنسور ها را خوانده و عملگر ها را کنترل کنیم. در این قسمت یک بروکر بالا می‌اوریم، و بورد ها را به آن متصل کرده و از طریق یک سرور جنگو که آن هم به بروکر متصل است آن ها را کنترل می‌کنیم.

پیشنیاز ها:

علاوه بر بورد ها، این بار نیاز به یک شبکه وای فای، یک دستگاه با بروکر mqtt و یک سرور جنگو داریم.

شرح پیشرفت:

ابتدا با دانلود بروکر mosquito و نصب و کانفیگ آن، با یک یوزرنیم و پسورد می توانیم به آن کانکت بشویم و پیام ها را انتقال دهیم، بعد از نصب آن مطابق سایت رسمی، تمامی دستگاه ها را به یک شبکه وای فای متصل می‌کنیم.

در مرحله بعد، کد هر کدام از سنسور ها و عملگر ها را برای اتصال به شبکه و دریافت یا ارسال پیام تغییر خواهیم داد.

قسمت اتصال به وایفای:

```
// wifi
const char* ssid = "connect";
const char* password =  "ramzeSade";

// mqtt
const char *mqtt_broker = "192.168.136.198";
const char *topic = "buzzer";
const char *mqtt_username = "uname";
const char *mqtt_password = "upass";
const int mqtt_port = 1883;

WiFiClient espClient;
PubSubClient client(espClient);


void setup() {

  Serial.begin(9600);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.println("Connecting to WiFi..");
  }

  Serial.println("Connected to the WiFi network");
  // Print local IP address and start web server
  Serial.println("");
  Serial.println("WiFi connected.");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP()); //show ip address when connected on serial monitor.
```

قسمت اتصال به بروکر:

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

}
```

همچنین کد هایی که در قسمت قبل زده بودیم، برای عملگر ها وارد تابع کال‌بک و برای سنسور ها در قسمت لوپ قرار می دهیم.

برای مثال عملگر بازر:

```
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]);
    }

    if (strcmp((char *) payload, "on")){
      digitalWrite(RELAY_PIN, HIGH);
      delay(1000);
      digitalWrite(RELAY_PIN, LOW);
    }


    Serial.println();
    Serial.println("-----------------------");
}
```

برای سنسور التراسونیک:

```
void loop() {
  client.loop();
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);

  // measure duration of pulse from ECHO pin
  duration_us = pulseIn(ECHO_PIN, HIGH);
  // calculate the distance
  distance_cm = 0.017 * duration_us;

  // print the value to Serial Monitor
  Serial.print("distance: ");
  Serial.print(distance_cm);
  Serial.println(" cm");
  char distance[64];
  sprintf(distance, "%f", distance_cm);
  client.publish(topic, distance);
  delay(5000);

}
```

برای سایر سنسور ها و عملگر ها هم کد ها را مشابه قبل تغییر دادیم.

در ادامه تابع هایی با جنگو در سرور نوشتیم و آن را هم به بروکر متصل کردیم، این تابع ها وظیفه کنترل هر کدام از عملگر ها ، و همچنین دریافت اطلاعات هر کدام از سنسور ها را دارند.

ما یک پروژه به وسیله جنگو بالا آوردیم که همزمان دو کار را میتواند انجام دهد. از یک سو می تواند به وسیله URL هایی که برای آن تعریف شده است، سیگنال های on یا off را برای Buzzer و LED و همچنین سیگنال position را برای servo motor از کاربر پشت سیستم دریافت کند و برای ESP32 بفرستد. از سوی دیگر، این سرور ما همواره روی تاپیک های Sound, Light, Ultrasound دارد به ESP32 گوش می کند و منتظر است تا اطلاعاتی را دریافت کند و سپس به کاربر نشان دهد.

در ادامه کار با پیاده سازی یک UI، می توانیم با قرار دادن تعدادی دکمه، کار اول را به طریق بهتری انجام دهیم و همچنین به صورت Live، تعدادی Chart را که مربوط به وضعیت سنسور های ما می باشد را تغییر دهیم.

نمونه کد های جنگو برای روشن کردن led و بازر:

```python
def buzzer_on(request):
    client = mqtt.Client()
    client.username_pw_set(MQTT_USER, MQTT_PASSWORD)
    client.connect(MQTT_BROKER, MQTT_PORT, MQTT_KEEPALIVE)

    client.publish(MQTT_TOPIC_BUZZER, "on")
    client.disconnect()

    return HttpResponse("Buzzer turned on", status=200)


def led_on(request):
    client = mqtt.Client()
    client.username_pw_set(MQTT_USER, MQTT_PASSWORD)
    client.connect(MQTT_BROKER, MQTT_PORT, MQTT_KEEPALIVE)

    client.publish(MQTT_TOPIC_LED, "on")
    client.disconnect()

    return HttpResponse("LED turned on", status=200)
```

کد دریافت اطلاعات سنسور ها:

```python
def on_connect(client, userdata, flags, rc):
    print(f"Connected with result code {rc}")
    for topic, qos in MQTT_TOPICS:
        client.subscribe(topic, qos)


def on_message(client, userdata, msg):
    print(f"Topic: {msg.topic}\nMessage: {msg.payload.decode()}")


def start_mqtt_client():
    client = mqtt.Client()
    client.on_connect = on_connect
    client.on_message = on_message
    client.username_pw_set(MQTT_USER, MQTT_PASSWORD)
    client.connect(MQTT_BROKER, MQTT_PORT, MQTT_KEEPALIVE)
    client.loop_forever()


def run_mqtt_client_in_thread():
    mqtt_thread = threading.Thread(target=start_mqtt_client)
    mqtt_thread.daemon = True
    mqtt_thread.start()
```

نمونه پیام های ارسال شده به وسیله بروکر **mosquito** :