2023-2 IoT System Design

HW #03

웹상에서 MQTT를 이용한

LED제어

Due 10.07(Sat) 22:00

전산전자공학부

22100579 이진주

1. **과제 설명**

이 과제에서는 라즈베리파이에서 python과 flask를 이용해 서버를 만들고, Mosquitto를 통해 nodeMCU와 통신하여 웹 서버에서의 버튼 신호에 따라 nodeMCU의 LED가 제어되도록 하는 것을 목표로 한다.

요구 기능/동작

* 웹서버의 버튼 조작에 따라 Mosuqitto의 “iot/[학번]” topic으로 정해진 keyword가 publishing된다.
  + Led toggle -> led
  + Led on -> ledon
  + Led off -> ledoff
* nodeMCU에서는 Mosquitto의 “iot/[학번]” topic을 subscribe 하고 있다가 특정 keyword가 payload로 들어오면 그에 따라 LED를 적절히 제어한다.
  + led -> ON if LED was OFF, OFF if LED was ON
  + ledon -> ON
  + ledoff -> OFF

1. **nodeMCU code**
2. Global scope

|  |
| --- |
| /\*lab5e\_EspMQTTClientTest.ino\*/  #include "EspMQTTClient.h"  #define D0 16  // LED(5파이)  #define LED\_PIN D0  #define LED\_ON HIGH //active HIGH  #define LED\_OFF LOW  const char \*WifiSSID = "leeejjju";  const char \*WifiPassword = "18639158";  #define mqtt\_broker "192.168.137.140"  #define mqtt\_clienname "JJLee"  int LED\_state;  EspMQTTClient mqtt\_client(    WifiSSID, //wifi information    WifiPassword,    mqtt\_broker,  // MQTT Broker server ip    // MQTTUsername,   // Can be omitted if not needed    // MQTTPassword,   // Can be omitted if not needed    mqtt\_clienname,     // Client name that uniquely identify your device    1883              // The MQTT port, default to 1883. this line can be omitted  ); |

Mosquitto로 통신하기 위하여 EspMQTTClient.h 을 include해준다.

가독성을 위해 LED제어에 사용될 pin 정보와 조정값 등을 미리 define해준다.

Mosquitto 연결에 사용될 정보들을 지정해두고 그를 사용해 mqtt\_client 인스턴스를 선언한다.

이후 LED 상태 제어에 사용할 LED\_state 변수를 선언해준다.

1. setup

|  |
| --- |
| Serial.begin(115200);  LED\_state = LED\_OFF;    pinMode(LED\_PIN, OUTPUT);  // LED 연결단자 |

메시지 출력을 위한 Serial을 셋업 하고 LED\_state의 초기값을 LED\_OFF로 설정해준다.

pinMode 함수를 사용해 LED와 연결될 output을 미리 define해둔 정보를 이용해 셋업 하였다.

1. onConnectionEstablished

|  |
| --- |
| mqtt\_client.subscribe("iot/22100579", [](const String & payload) {      Serial.print(payload);        if(payload.indexOf("ledon") != -1){        LED\_state = LED\_ON;        Serial.println(" :LED turnd on");      }else if(payload.indexOf("ledoff") != -1){        LED\_state = LED\_OFF;        Serial.println(" :LED turnd off");      }else if(payload.indexOf("led") != -1){        if(LED\_state == LED\_ON) LED\_state = LED\_OFF;        else LED\_state = LED\_ON;        Serial.println(" :LED toggled");      }else{        Serial.println(" :invalid command");      }      if(LED\_state == LED\_ON) digitalWrite(LED\_PIN, LED\_ON);      else digitalWrite(LED\_PIN, LED\_OFF);    }); |

위에서 만들어둔 mqtt\_client 를 사용하여 토픽 iot/22100579로부터 subscribe된 메시지를 payload에 담아 사용할 수 있게 한다.

Pyaload가 지정된 각 키워드에 일치할 시 LED\_state를 적절한 값으로 변화시키고 LED\_state의 값을 이용해 실제 LED를 제어한다.

1. loop

|  |
| --- |
| mqtt\_client.loop(); |

Setup 실행 후 무한히 반복되는 loop함수의 내용이다.

1. **Python 및 HTML 코드**

|  |
| --- |
| # lab4\_rpi\_flask\_nodeMCU\_LED\_MQTT\_v2.py   from flask import Flask, render\_template   from flask\_mqtt import Mqtt     app = Flask(\_\_name\_\_)   app.config['MQTT\_BROKER\_URL'] = '192.168.137.140'   app.config['MQTT\_BROKER\_PORT'] = 1883     mqtt = Mqtt(app)     @app.route('/led')   def main():           return render\_template('index.html')     @app.route('/led/toggle')   def led\_toggle():           mqtt.publish('iot/22100579', 'led')           return render\_template('index.html')       @app.route('/led/on')   def led\_on():           mqtt.publish('iot/22100579', 'ledon')           return render\_template('index.html')       @app.route('/led/off')   def led\_off():           mqtt.publish('iot/22100579', 'ledoff')           return render\_template('index.html')     if \_\_name\_\_ == '\_\_main\_\_':           app.run(host = '0.0.0.0', debug=False) |

Python 코드에서는 Flask를 이용해 웹서버 특정 주소로 이동될 때의 동작을 설정한다.

Mosquitto를 사용할 수 있도록 import해주고, 적절한 broker 정보를 넣어 세팅해주었다.

버튼의 동작 결과로 이동될 각 주소에서 적절한 키워드가 mosquitto의 토픽 'iot/22100579'으로 publishing되도록 하였다.

|  |
| --- |
| <!DOCTYPE HTML>   <html>   <head>   <title>NodeMCU Control</title>   </head>   <body>   <div style='width: 300px; margin: auto; text-align: center;'>   <h1>Welcome to Handong Global University</h1>   <h2>NodeMCU Web Server with Mosquitto</h2>   <p></p>   <a href="/led/toggle"><button>LED toggle </button></a>   <a href="/led/on"><button>LED On </button></a>   <a href="/led/on"><button>LED Off </button></a><br/>   </div>   </body>   </html> |

서버의 메인 페이지에 띄워질 html 코드이다.

적절한 GUI를 구성하며, 각 버튼들이 적절한 주소로 이동하도록 설정해 주었다.

1. **결과 및 느낀점**
2. 결과

Flask를 통해 만들어진 웹 서버에 접속시 다음과 같은 GUI를 이용할 수 있다.

텍스트, 스크린샷, 소프트웨어, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

라즈베리파이의 터미널에서 topic을 subscribe하여 버튼이 눌릴 때 적절한 keyword가 publishing됨을 확인하였다.

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

동일한 토픽을 subscribe 하고 있는 nodeMCU 또한 해당 keyword를 받아 payload에 저장하며, Serial monitor에서 다음과 같이 확인해볼 수 있다.

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

nodeMCU가 mqtt subscribe를 통해 입력된 payload와 각 keyword의 일치 여부를 판단하여 LED를 적절히 조작함을 확인하였다.

|  |  |
| --- | --- |
| 전자 공학, 회로, 회로 구성요소, 전자 부품이(가) 표시된 사진  자동 생성된 설명 | 전자 공학, 전기 배선, 회로 구성요소, 전자제품이(가) 표시된 사진  자동 생성된 설명 |
| MQTT로 ledoff || ON상태에서 led 받았을 때 | MQTT로 ledon || OFF상태에서 led 받았을 때 |

1. 배운 점 및 느낀 점

Flask를 통해 간단하게 웹 서버를 만들 수 있다는 점이 신기했다. GUI를 사용해 눈에 보이고 익숙한 인터페이스로 하드웨어를 제어할 수 있다는 점이 흥미로운 과제였다. 또한 라즈베리파이, nodeMCU라는 별개의 기계가 mosquitto를 통해 유기적으로 동작하는 점이 재미있었다.

코드를 작성할 때는 이전에 배웠듯 state의 설정과 제어를 분리하는 방법을 자연스럽게 실습해볼 수 있어 좋았다. 보고서에서 코드의 간단한 설명을 덧붙이며 불필요한 코드를 쳐내고 작성된 코드들을 확실히 이해할 수 있었던 것 같다.