วันที่ 07 เดือน มกราคม พ.ศ 2565

ชื่อ-นามสกุล นายวรเชษฐ์ เปไธสง รหัสนักศึกษา B6214197

ปฏิบัติการที่ 2

JSON and Pubsubclient

วัตถุประสงค์

1. เข้าใจวิธีการเก็บข้อมูลด้วยรูปแบบ JSON และใช้งานได้อย่างถูกต้อง
2. สามารถใช้งานไมโครคอนโทรลเลอร์เพื่อรับส่งข้อมูลผ่านทาง MQTT protocol ได้
3. สามารถประยุกต์ใช้งานการรับส่งข้อมูล ด้วย MQTT สำหรับการอ่านค่าหรือควบคุมอุปกรณ์ปลายทางได้

อุปกรณ์การทดลอง

- คอมพิวเตอร์
- MQTT broker
- ESP32 microcontroller
- IO board

ปฏิบัติการที่ 2.1 สามารถเก็บข้อมูลและอ่านข้อมูลที่อยู่ในรูปแบบ JSON ได้

จงเขียนข้อมูลต่อไปนี้ให้อยู่ในรูปแบบ JSON

2.1.1 Name = Minny
Age = 25
Car = Ford

{“Name”:”Minny”,”Age”:25,”Car”:”Ford”}

2.1.2 employee
Name = John
Position=Manager
Salary = 85,000
Married=True

{“employee”:{“Name”:”John”,”Position”:”Manager”,”Salary”:85,000,”Married”:True}

2.1.3 Company = Energy corp
Employee = 2500
Working_day= Monday, Tuesday, Wednesday, Thursday, Friday

{“Company”:”Energy corp”,”Employee”:2500,”Working_day”,[“Monday”, “Tuesday”, “Wednesday”, “Thursday”, “Friday”]}

ปฏิบัติการที่ 2.2 การใช้งานไมโครคอนโทรลเลอร์เพื่อรับส่งข้อมูลผ่านทาง MQTT protocol

2.2.1 จง publish ข้อมูลด้วยไมโครคอนโทรลเลอร์โดยมีรายละเอียดดังนี้

Topic1: “Bxxxxxx/lab02/ext2_2_1” (ให้ใช้รหัสนักศึกษาของตนเอง)
Payload: {“emeter”:”meter01”,”energy”:250,”freq”:60}
ส่งข้อมูลทุกๆ 1 วินาที

ให้ทำการ subscribe โดยใช้ MQTTbox

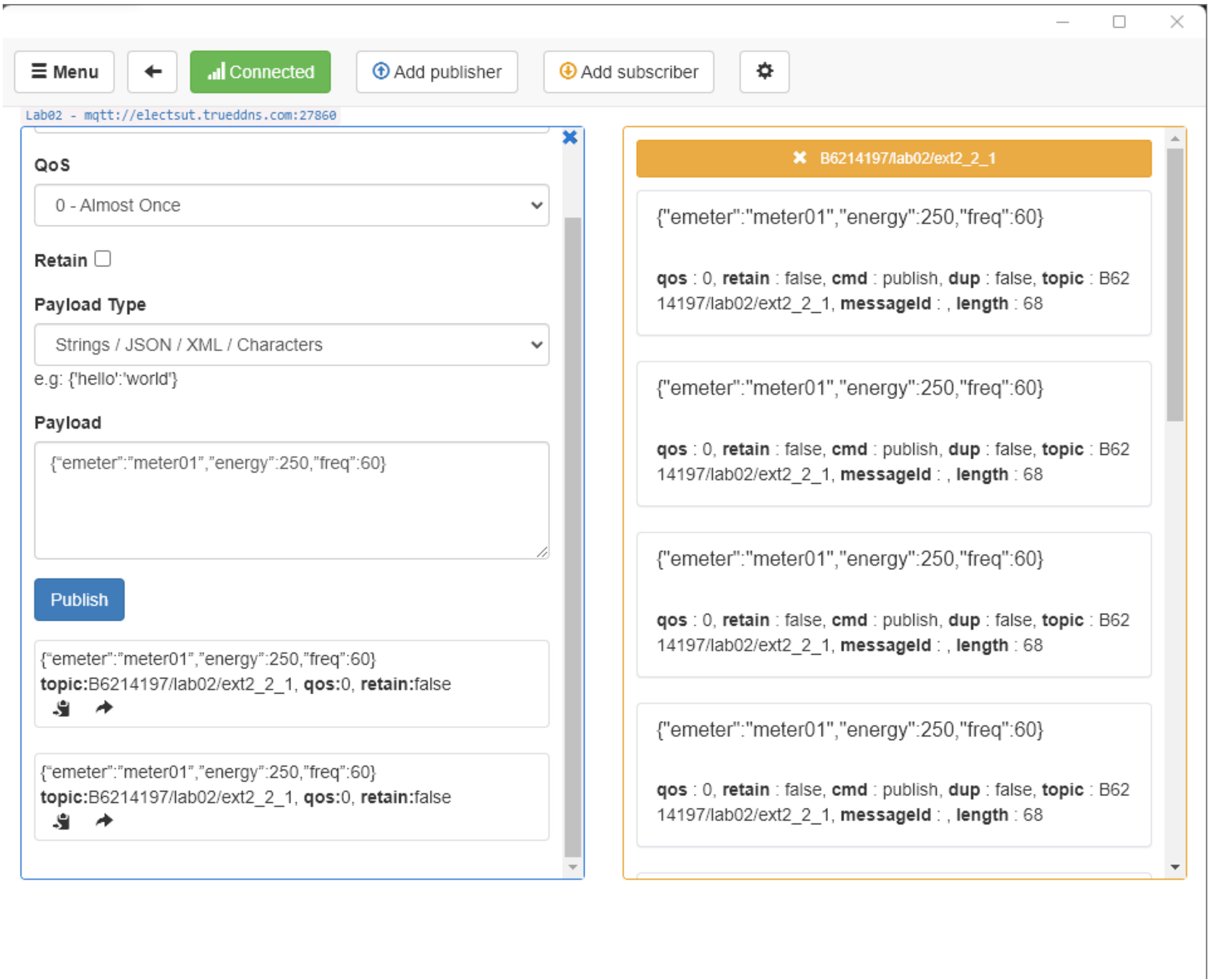
Code:

```
#include <WiFi.h>
#include <PubSubClient.h>
WiFiClient client;
PubSubClient mqtt(client);
#define WIFI_STA_NAME "Rungtiwa"
#define WIFI_STA_PASS "0818774251"
#define MQTT_SERVER   "electsut.trueddns.com"
#define MQTT_PORT     27860
#define MQTT_USERNAME ""
#define MQTT_PASSWORD ""
#define MQTT_NAME     "B6214197"
#define LED_PIN 23
#define sw 5

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

void setup() {
  Serial.begin(115200);
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(WIFI_STA_NAME);
```

```
  WiFi.mode(WIFI_STA);
  WiFi.begin(WIFI_STA_NAME, WIFI_STA_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
    digitalWrite(LED_BUILTIN, !digitalRead(LED_BUILTIN));
  }
  digitalWrite(LED_BUILTIN, HIGH);
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  mqtt.setServer(MQTT_SERVER, MQTT_PORT);
  mqtt.setCallback(callback);
}
void loop() {
  if (mqtt.connected() == false) {
    Serial.print("MQTT connecting... ");
    if (mqtt.connect(MQTT_NAME, MQTT_USERNAME, MQTT_PASSWORD))
    {
      Serial.println("connected");
    }
    else {
      Serial.println("failed");
      delay(1000);
         }
                              }
  else {
         mqtt.loop();

mqtt.publish("B6214197/lab02/ext2_2_1","{\"emeter\":\"meter01\",\"energy\":250,\"freq\":60}");
         delay(1000);
     }
          }
```

ชื่อ-นามสกุล นายวรเชษฐ์ เปไธสง รหัสนักศึกษา B6214197

ภาพของ MQTTbox ที่ได้รับข้อมูลดังกล่าว



คำถาม

ข้อมูลที่ได้รับมีความยาวเท่าไหร

length : 68

2.2.2 ให้ทำการเพิ่ม topic และข้อมูล ดังต่อไปนี้

Topic2: “Bxxxxxx/lab02/ext2_2_2”

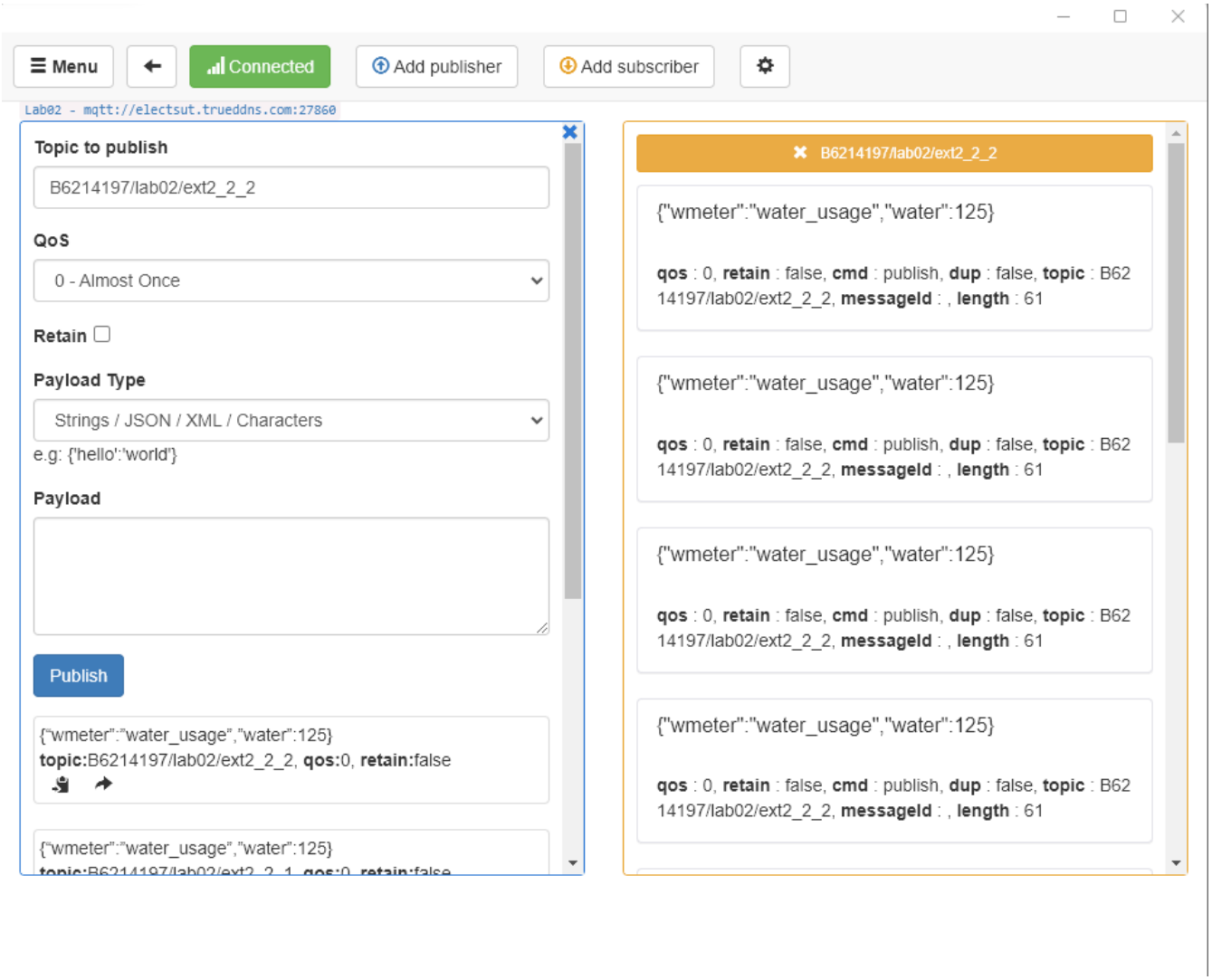
Payload: {“wmeter”:”water_usage”,”water”:125}

ส่งข้อมูลทุกๆ 1 วินาที

Code: เฉพาะในส่วน mqtt.publish

```
mqtt.publish("B6214197/lab02/ext2_2_1","{\"emeter\":\"meter01\",\"energy\":250,\"freq\":60}");
mqtt.publish("B6214197/lab02/ext2_2_2","{\"wmeter\":\"water_usage\",\"water\":125}");
```

ภาพของ MQTTbox ที่ได้รับข้อมูลดังกล่าว



2.2.3 ให้ทำการบรรจุข้อมูลดังกล่าวโดยใช้ Arduino Json ในการ Serialize ข้อมูล จากนั้นให้ publish ข้อมูล และ subscribe ด้วย MQTTbox

meter= “meter01”
energy=250
freq=60
wmeter=”water_usage”
water=125

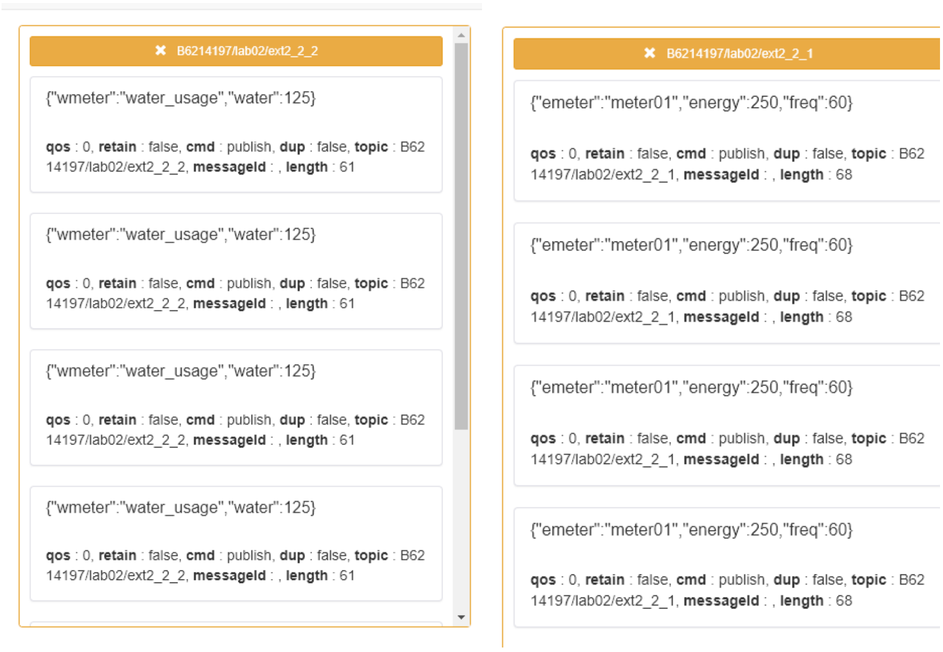
Code:

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

```
DynamicJsonDocument docd(1024);
DynamicJsonDocument docs(1024);
DynamicJsonDocument docs1(1024);

WiFiClient client;
PubSubClient mqtt(client);
#define WIFI_STA_NAME "Rungtiwa"
#define WIFI_STA_PASS "0818774251"
#define MQTT_SERVER   "electsut.trueddns.com"
#define MQTT_PORT     27860
#define MQTT_USERNAME ""
#define MQTT_PASSWORD ""
#define MQTT_NAME     "B6214197"
#define LED_PIN 23
#define sw 5
char out1[256];
char out2[256];
void callback(char* topic, byte* payload, unsigned int length){
  Serial.print("Topic = ");
  Serial.print(topic);
  for(int i=0;i< length;i++){
    Serial.print((char)payload[i]);
                        }
  Serial.println();
                                                }

void setup() {
  Serial.begin(115200);
  pinMode(LED_BUILTIN, OUTPUT);
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(WIFI_STA_NAME);
  WiFi.mode(WIFI_STA);
  WiFi.begin(WIFI_STA_NAME, WIFI_STA_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
    digitalWrite(LED_BUILTIN, !digitalRead(LED_BUILTIN));
  }
  digitalWrite(LED_BUILTIN, HIGH);
```

```
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  mqtt.setServer(MQTT_SERVER, MQTT_PORT);
  mqtt.setCallback(callback);
}
void loop() {
  if (mqtt.connected() == false) {
    Serial.print("MQTT connecting... ");
    if (mqtt.connect(MQTT_NAME, MQTT_USERNAME, MQTT_PASSWORD))
    {
      Serial.println("connected");
    }
    else {
      Serial.println("failed");
      delay(1000);
         }
                              }
  else {
         mqtt.loop();
         docs["emeter"]="meter01";
         docs["energy"]=250;
         docs["freq"]=60;
         serializeJson(docs,out1);
         mqtt.publish("B6214197/lab02/ext2_2_1",out1);
         docs1["wmeter"]="water_usage";
         docs1["water"]=125;
         serializeJson(docs1,out2);
         mqtt.publish("B6214197/lab02/ext2_2_2",out2);
//
mqtt.publish("B6214197/lab02/ext2_2_1","{\"emeter\":\"meter01\",\"energy\":250,\"freq\":60}");
//         mqtt.publish("B6214197/lab02/ext2_2_2","{\"wmeter\":\"water_usage\",\"water\":125}");
         delay(1000);
    }
          }
```

ภาพของ MQTTbox ที่ได้รับข้อมูลดังกล่าว



คำถาม
ข้อมูลที่ได้รับทาง MQTTbox เหมือนหรือต่างจากข้อที่ผ่านมาอย่างไรจงอธิบาย

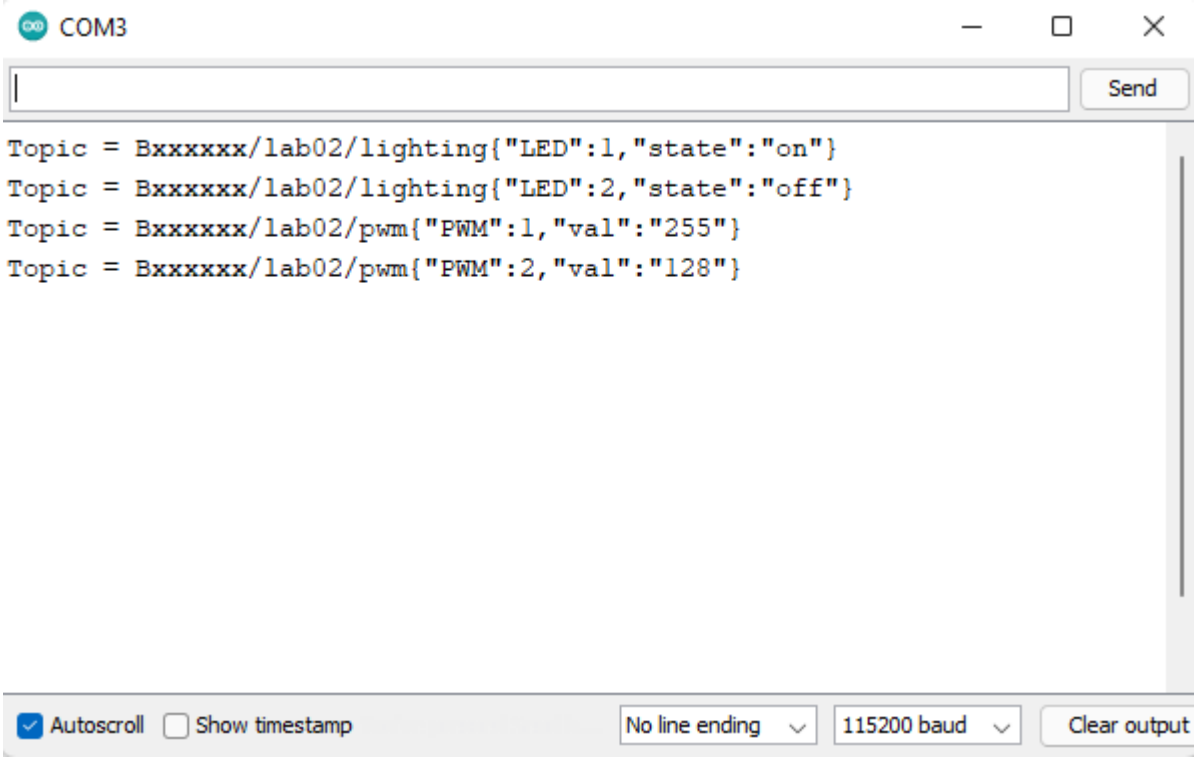
รับข้อมูลเหมือนกัน แต่ใช้ส่งข้อมูลต่างกัน จากข้อที่ผ่านมาจะใช้ส่งข้อมูลแบบ library pubsubclient และข้อ 2.2.3 ใช้ส่งข้อมูลแบบ include library เข้ามาช่วยในการจัดสรรข้อมูล

2.2.4 ให้ทำการเขียนโค้ดไมโครคอนโทรลเลอร์เพื่อทำการ subscribe ข้อมูล โดยให้ publish ข้อมูลผ่านทาง MQTT box โดยมีรายละเอียดดังต่อไปนี้

Topic 3: “Bxxxxxx/lab02/lighting”
MQTTbox publish = {"LED":1,"state":"on"}
MQTTbox publish = {“LED”:2,”state”:”off”}

Topic 4: “Bxxxxxx/lab02/pwm”
MQTTbox publish = {“PWM”:1,”val”:255}
MQTTbox publish = {“PWM”:2,”val”:128}

ให้พิมพ์ topic และข้อมูลที่ได้รับออกมาทาง Serial Port ดังตัวอย่างต่อไปนี้



Code: เฉพาะในส่วนของ callback

```
void callback(char* topic, byte* payload, unsigned int length){
  Serial.print("Topic= ");
  Serial.print(topic);
  deserializeJson(docd,payload,length);
  serializeJson(docd,out3);
  Serial.print();
  }
```

Code: เฉพาะในส่วนของ subscribe

```
mqtt.subscribe("B6214197/lab02/lighting");
mqtt.subscribe("B6214197/lab02/pwm");
```

ปฏิบัติการที่ 2.3 การประยุกต์ใช้งานการรับส่งข้อมูลด้วย MQTT สำหรับการอ่านค่าหรือควบคุมอุปกรณ์ปลายทาง

2.3.1 จากปฏิบัติการที่ 2.2 ให้ทำการแก้ไขโค้ดไมโครคอนโทรลเลอร์ เพื่อทำการแปลงข้อมูลที่ได้รับจาก MQTTbox เพื่อสั่งการอุปกรณ์โดยมีรายละเอียดดังต่อไปนี้

หากได้รับข้อความ {"LED":1,"state":"on"} ให้ทำการ on LED ที่ต่ออยู่กับขา 18

หากได้รับข้อความ {"LED":1,"state":"off"} ให้ทำการ off LED ที่ต่ออยู่กับขา 18

หากได้รับข้อความ {"LED":2,"state":"on"} ให้ทำการ on LED ที่ต่ออยู่กับขา 19
หากได้รับข้อความ {"LED":2,"state":"off"} ให้ทำการ off LED ที่ต่ออยู่กับขา 19

Code: เฉพาะในส่วนของ callback

```
Serial.print("Topic = ");
 Serial.print(topic);
 deserializeJson(docd,payload,length);
 serializeJson(docd,out3);
 Serial.print(out3);
 Serial.println();
 int LED = docd ["LED"];
 const char* state = docd["state"];
 Serial.println(LED);
 Serial.println(state);
 std::string s = state;
 if (LED == 1){
   if(s == "on"){digitalWrite(18,1);}
   else if (s == "off"){digitalWrite(18,0);}
                    }
 else if (LED == 2){
   if(s == "on"){digitalWrite(19,1);}
   else if (s == "off"){digitalWrite(19,0);}
                          }
```

2.3.2 จากปฏิบัติการที่ 2.2 ให้ทำการแก้ไขโค้ดไมโครคอนโทรลเลอร์ เพื่อทำการแปลงข้อมูลที่ได้รับจาก MQTTbox เพื่อสั่งการอุปกรณ์โดยมีรายละเอียดดังต่อไปนี้
ให้กำหนด PWM ของไมโครคอนโทรลเลอร์ให้มีความละเอียด 8 bit

หากได้รับข้อความ {“PWM”:1,”val”:255} ให้ทำการปรับความสว่างของหลอด LED ที่ต่ออยู่กับขา 12 ให้เท่ากับค่าที่ส่งไปซึ่งอยู่ในช่วง 0- 255

หากได้รับข้อความ {“PWM”:1,”val”:255} ให้ทำการปรับความสว่างของหลอด LED ที่ต่ออยู่กับขา 14 ให้เท่ากับค่าที่ส่งไป ซึ่งอยู่ในช่วง 0-255

Code: เฉพาะในส่วนของ callback

```
 Serial.print("Topic = ");
  Serial.print(topic);
  deserializeJson(docd,payload,length);
  serializeJson(docd,out3);
  Serial.print(out3);
  Serial.println();
  int pwm = docd ["PWM"];
  int val = docd ["val"];
  Serial.println(pwm);
  Serial.println(val);
  if (pwm == 1){
    ledcWrite(1,val);
                    }
  else if(pwm == 2){
    ledcWrite(2,val);
                        }
```

2.3.3 ให้ทำการต่อสวิตซ์เข้ากับไมโครคอนโทรลเลอร์ขาที่ 4 จากนั้นให้เขียนโปรแกรมเพื่อเรียกดูสถานะของสวิตซ์ โดยมีรายละเอียดดังต่อไปนี้

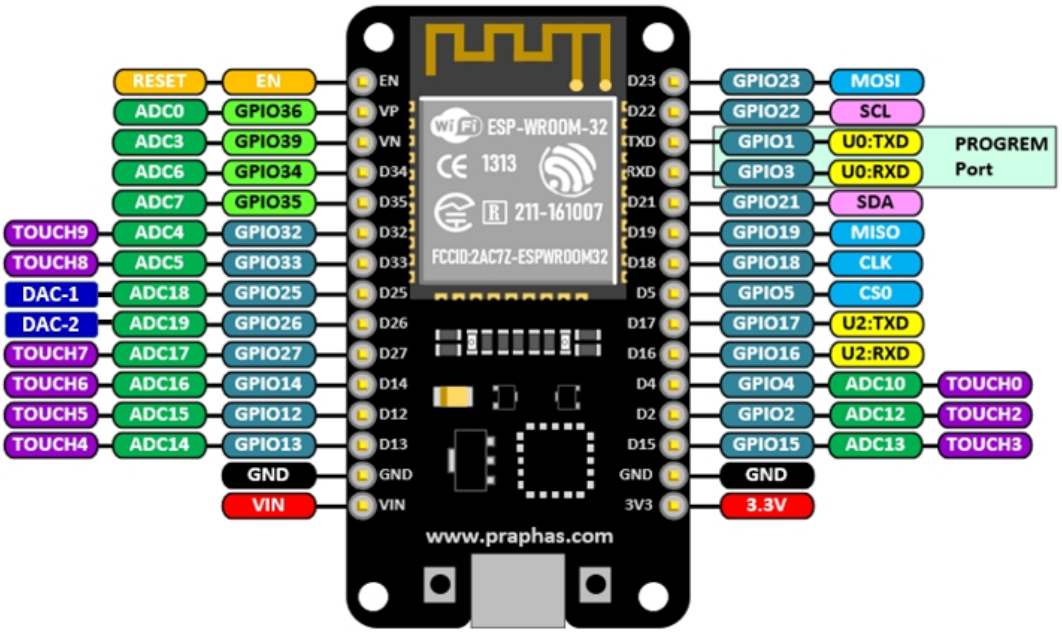
ให้ publish ข้อมูลผ่านทาง MQTTbox เพื่อเรียกดูสถานะของสวิตซ์1 {“sw_status”:”sw1”}
โดยใช้ **Topic 5:** “Bxxxxxx/lab02/sw”

จากนั้นให้ ไมโครคอนโทรลเลอร์ตอบกลับสถานะของสวิตช์โดย publish ข้อมูลกลับไปทาง **Topic5:** “Bxxxxxx/lab02/swst โดยpublish ข้อความกลับไปว่า {“sw1”:”on”} ถ้าสวิตซ์ถูกกด หรือ {“sw1”:”off”} ถ้าสวิตซ์ไม่ถูกกด

Code: เฉพาะในส่วนของ callback

```
 Serial.print("Topic = ");
  Serial.print(topic);
  deserializeJson(docd,payload,length);
  serializeJson(docd,out3);
  Serial.print(out3);
  Serial.println();
  const char* sw_status = docd["sw_status"];
//  Serial.println(sw_status);
   int state = !digitalRead(4);
   Serial.println(state);
   if(state == 1){
     docs3["sw1"] = "on";
     serializeJson(docs3,out4);
```

```
  mqtt.publish("B6214197/lab02/swst",out4);

            }
else if (state == 0){
  docs3["sw1"] = "off";
  serializeJson(docs3,out4);
  mqtt.publish("B6214197/lab02/swst",out4);
                }
```



GPIO	Input	Output	Notes	ADC	Touch	PWM
0	pulled up	OK	outputs PWM signal at boot			OK
1	TX pin	OK	debug output at boot			OK
2	OK	OK	connected to on-board LED	ADC12	2	OK
3	OK	RX pin	HIGH at boot			OK
4	OK	OK		ADC10	0	OK
5	OK	OK	outputs PWM signal at boot			OK
6	x	x	connected to SPI flash			x
7	x	x	connected to SPI flash			x
8	x	x	connected to SPI flash			x
9	x	x	connected to SPI flash			x
10	x	x	connected to SPI flash			x
11	x	x	connected to SPI flash			x
12	OK	OK	boot fail if pulled high	ADC15	5	OK
13	OK	OK		ADC14	4	OK
14	OK	OK	outputs PWM signal at boot	ADC16	6	OK
15	OK	OK	outputs PWM signal at boot	ADC13	3	OK
16	OK	OK				OK
17	OK	OK				OK
18	OK	OK				OK
19	OK	OK				OK
21	OK	OK				OK
22	OK	OK				OK
23	OK	OK				OK
25	OK	OK		ADC18		OK
26	OK	OK		ADC19		OK
27	OK	OK		ADC17	7	OK
32	OK	OK		ADC4	9	OK
33	OK	OK		ADC5	8	OK
34	OK	x	input only	ADC6		x
35	OK	x	input only	ADC7		x
36	OK	x	input only	ADC0		x
39	OK	x	input only	ADC3		x