|  |
| --- |
| **แนวทางการใช้งานอินเทอร์เน็ตของสรรพสิ่งในระบบการผลิต**  **IoT Approaches to Manufacturing System** |
| **ขื่อ-สกุล : วราสิริ ลิ้มประเสริฐ B6214005** |

**5/5. คำถามท้ายบทเพื่อทดสอบความเข้าใจ**

**Quiz\_401 – Ubidots: Monitor DHT22, Monitor Digital Switch and Control 4 LED**

|  |
| --- |
|  |
| #include <WiFi.h>  #include <PubSubClient.h>  #include "DHTesp.h"  const char \*My\_SSID = "V2036";  const char \*My\_Pass = "fnafchica";  const char \*MQTT\_Server = "things.ubidots.com";  const char \*MQTT\_User = "BBFF-gvcR0u8y0BegX9muo6Vfs4mKvsItpI";  const char \*MQTT\_Pass = "BBFF-gvcR0u8y0BegX9muo6Vfs4mKvsItpI";  const char \*PTopic1 = "/v2.0/devices/bearish";  const char \*STopic1 = "/v2.0/devices/bearish/humid";  const char \*STopic2 = "/v2.0/devices/bearish/tempp";  const char \*STopic3 = "/v2.0/devices/bearish/led1";  const char \*STopic4 = "/v2.0/devices/bearish/led2";  const char \*STopic5 = "/v2.0/devices/bearish/led3";  const char \*STopic6 = "/v2.0/devices/bearish/led4";  const char \*STopic7 = "/v2.0/devices/bearish/sw1";  const char \*STopic8 = "/v2.0/devices/bearish/sw2";  #define MQTT\_Port 1883  #define Test\_LED1 18  #define Test\_LED2 19  #define Test\_LED3 22  #define Test\_LED4 23  #define Test\_SW1 4  #define Test\_SW2 21  #define Pin\_DHT22 15  DHTesp dht;  WiFiClient espClient;  PubSubClient client(espClient);  long lastMsg = 0;  char msg[50];  int value = 0;  void Setup\_Wifi() {  delay(10);  Serial.println();  Serial.print("Connecting to ");  Serial.println(My\_SSID);  WiFi.begin(My\_SSID, My\_Pass);  while (WiFi.status() != WL\_CONNECTED) {  delay(500); Serial.print(".");  }  randomSeed(micros());  Serial.println("");  Serial.println("WiFi connected");  Serial.println("IP address: ");  Serial.println(WiFi.localIP());  }  void reconnect()  { while (!client.connected()) // Loop until we're reconnected  { Serial.print("Attempting MQTT connection...");  String clientId = "ESP32 Client-";  clientId += String(random(0xffff), HEX); // Create a random client ID  if (client.connect(clientId.c\_str(), MQTT\_User, MQTT\_Pass)) // Attempt to connect  { Serial.println("connected"); // Once connected, publish an announcement...  client.subscribe(STopic1);  client.subscribe(STopic2);  client.subscribe(STopic3);  client.subscribe(STopic4);  client.subscribe(STopic5);  client.subscribe(STopic6);  client.subscribe(STopic7);  client.subscribe(STopic8);  } else  { Serial.print("failed, rc=");  Serial.print(client.state());  Serial.println(" try again in 5 seconds");  delay(5000);  }  }  }  void callback(char \*topic, byte \*payload, unsigned int length)  { Serial.print("Message arrived [");  Serial.print(topic);  Serial.print("] ");  for (int i = 0; i < length; i++)  { Serial.print((char)payload[i]);  }  if (topic[24] == STopic3[24]) {  Serial.print(" -LED1->> ");  Serial.print((char)payload[10]);  if (payload[10] == '1')  digitalWrite(Test\_LED1, HIGH);  else  digitalWrite(Test\_LED1, LOW);  }  if (topic[24] == STopic4[24]) {  Serial.print(" -LED2->> ");  Serial.print((char)payload[10]);  if (payload[10] == '1')  digitalWrite(Test\_LED2, HIGH);  else  digitalWrite(Test\_LED2, LOW);  }  if (topic[24] == STopic5[24]) {  Serial.print(" -LED3->> ");  Serial.print((char)payload[10]);  if (payload[10] == '1')  digitalWrite(Test\_LED3, HIGH);  else  digitalWrite(Test\_LED3, LOW);  }  if (topic[24] == STopic6[24]) {  Serial.print(" -LED4->> ");  Serial.print((char)payload[10]);  if (payload[10] == '1')  digitalWrite(Test\_LED4, HIGH);  else  digitalWrite(Test\_LED4, LOW);  }  Serial.println();  }  void setup()  { pinMode(Test\_LED1, OUTPUT);  pinMode(Test\_LED2, OUTPUT);  pinMode(Test\_LED3, OUTPUT);  pinMode(Test\_LED4, OUTPUT);  pinMode(Test\_SW1, INPUT\_PULLDOWN);  pinMode(Test\_SW2, INPUT\_PULLDOWN);  dht.setup(Pin\_DHT22, DHTesp::DHT22);  Serial.begin(115200);  Setup\_Wifi();  client.setServer(MQTT\_Server, MQTT\_Port);  client.setCallback(callback);  }  void loop()  { if (!client.connected()) reconnect();  client.loop();  long now = millis();  if (now - lastMsg > 5000)  { lastMsg = now;  float humidity = dht.getHumidity();  float temperature = dht.getTemperature();  int sw1 = 0;  int sw2 = 0;  if (digitalRead(Test\_SW1) == HIGH) sw1 = 1;  else sw1 = 0;  if (digitalRead(Test\_SW2) == LOW) sw2 = 1;  else sw2 = 0;  snprintf (msg, 75, "{ \"humid\" : %.2f, \"tempp\": %.2f, \"sw1\": %d, \"sw2\": %d }",  humidity, temperature, sw1, sw2);  Serial.print("Publish message: ");  Serial.println(msg);  client.publish(PTopic1, msg);  }  } |
|  |
|  |
|  |

**Quiz\_402 – Ubidots: Monitor DHT22 with TM1638 Display and LINE Alert**

* + ส่งข้อมูลอุณหภูมิไปยัง Ubidots
  + หากอุณหภูมิที่อ่านได้เกิน 28’C ให้แจ้งเตือนผ่าน LINE และบอกด้วยว่าอุณหภูมิเท่าใด
* แสดงอุณหภูมิที่ 7\_Segment Display TM1638 Board

|  |
| --- |
|  |
| #include <WiFi.h>  #include <PubSubClient.h>  #include <HTTPClient.h>  #include <TM1638plus.h>  #include "DHTesp.h"  const char \*My\_SSID = "V2036";  const char \*My\_Pass = "fnafchica";  const char \*MQTT\_Server = "things.ubidots.com";  const char \*MQTT\_User = "BBFF-gvcR0u8y0BegX9muo6Vfs4mKvsItpI";  const char \*MQTT\_Pass = "BBFF-gvcR0u8y0BegX9muo6Vfs4mKvsItpI";  #define WebHooksKey "oXSQX-hS7mc2o1blAA3UlubXBXN2WIrMlIheoCkvYQI"  #define WebHooksEventName "test\_GSheet"  #define WebHooksEventName\_line "Test\_Key"  const char \*PTopic1 = "/v2.0/devices/bearish";  const char \*STopic1 = "/v2.0/devices/bearish/humid";  const char \*STopic2 = "/v2.0/devices/bearish/tempp";  #define Brd\_STB 18 // strobe = GPIO connected to strobe line of module  #define Brd\_CLK 19 // clock = GPIO connected to clock line of module  #define Brd\_DIO 21 // data = GPIO connected to data line of module  bool high\_freq = true; //default false,, If using a high freq CPU > ~100 MHZ set to true.  TM1638plus tm(Brd\_STB, Brd\_CLK , Brd\_DIO, high\_freq);  #define MQTT\_Port 1883  #define Pin\_DHT22 15  #define My\_NAME "B6214005 Varasiri Limprasert"  DHTesp dht;  WiFiClient espClient;  PubSubClient client(espClient);  long lastMsg = 0;  char msg[50];  int value = 0;  void Setup\_Wifi() {  delay(10);  Serial.println();  Serial.print("Connecting to ");  Serial.println(My\_SSID);  WiFi.begin(My\_SSID, My\_Pass);  while (WiFi.status() != WL\_CONNECTED) {  delay(500); Serial.print(".");  }  randomSeed(micros());  Serial.println("");  Serial.println("WiFi connected");  Serial.println("IP address: ");  Serial.println(WiFi.localIP());  }  void reconnect()  { while (!client.connected()) // Loop until we're reconnected  { Serial.print("Attempting MQTT connection...");  String clientId = "ESP32 Client-";  clientId += String(random(0xffff), HEX); // Create a random client ID  if (client.connect(clientId.c\_str(), MQTT\_User, MQTT\_Pass)) // Attempt to connect  { Serial.println("connected"); // Once connected, publish an announcement...  client.subscribe(STopic1);  client.subscribe(STopic2);  } else  { Serial.print("failed, rc=");  Serial.print(client.state());  Serial.println(" try again in 5 seconds");  delay(5000);  }  }  }  void setup()  {  tm.displayBegin();  dht.setup(Pin\_DHT22, DHTesp::DHT22);  Serial.begin(115200);  Setup\_Wifi();  client.setServer(MQTT\_Server, MQTT\_Port);  }  void loop()  { if (!client.connected()) reconnect();  client.loop();  long now = millis();  if (now - lastMsg > 5000)  { lastMsg = now;  float humidity = dht.getHumidity();  float temperature = dht.getTemperature();  snprintf (msg, 75, "{ \"humid\" : %.2f, \"tempp\": %.2f}", humidity, temperature);  Serial.print("Publish message: ");  Serial.println(msg);  client.publish(PTopic1, msg);  Serial.println();  Serial.print("\nTemperature('C) = ");  Serial.print(temperature, 1);  Serial.print("\tHumidity(%) = ");  Serial.print(humidity, 1);  String serverName = "http://maker.ifttt.com/trigger/" +  String(WebHooksEventName) + "/with/key/" + String(WebHooksKey);  String httpRequestData = "value1=" + String(My\_NAME) + "&value2=" +  String(temperature) + "&value3=" +  String(humidity);  Serial.println();  Serial.println("Server Name >> " + serverName);  Serial.println("json httpRequestData >> " + httpRequestData);  if (WiFi.status() == WL\_CONNECTED) {  HTTPClient http;  http.begin(serverName);  http.addHeader("Content-Type", "application/x-www-form-urlencoded");  int httpResponseCode = http.POST(httpRequestData);  Serial.print("HTTP Response code: ");  Serial.println(httpResponseCode);  http.end();  if (httpResponseCode == 200)  Serial.println("[Google sheet] --> Successfully sent");  else  Serial.println("[Google sheet] --> Failed!");  }  else {  Serial.println("WiFi Disconnected");  }  /// if temp > 28 C send notifications >> line  if (temperature > 28) {  String serverName = "http://maker.ifttt.com/trigger/" +  String(WebHooksEventName\_line) + "/with/key/" + String(WebHooksKey);  String httpRequestData = "value1=" + String(temperature);  Serial.println();  Serial.println("Server Name >> " + serverName);  Serial.println("json httpRequestData >> " + httpRequestData);  if (WiFi.status() == WL\_CONNECTED) {  HTTPClient http;  http.begin(serverName);  http.addHeader("Content-Type", "application/x-www-form-urlencoded");  int httpResponseCode = http.POST(httpRequestData);  Serial.print("HTTP Response code: ");  Serial.println(httpResponseCode);  http.end();  if (httpResponseCode == 200)  Serial.println("[Line] --> Successfully sent");  else  Serial.println("[Line] --> Failed!");  }  else {  Serial.println("WiFi Disconnected");  }  }  /\*Display \*/  int t = int(temperature \* 100);  int Tempp2 = (int)temperature / 10; int Tempp1 = (int)temperature % 10; int Tempp0 =  (int)(temperature \* 10) % 10;  int Humi2 = (int)humidity / 10; int Humi1 = (int)humidity % 10; int Humi0 =  (int)(humidity \* 10) % 10;  tm.displayHex(0, Tempp2);  tm.displayASCIIwDot(1, Tempp1 + '0'); // turn on dot  tm.displayHex(2, Tempp0);  tm.display7Seg(3, B01011000); // Code=tgfedcba  tm.displayHex(4, Humi2);  tm.displayASCIIwDot(5, Humi1 + '0'); // turn on dot  tm.displayHex(6, Humi0);  tm.display7Seg(7, B01110100); // Code=tgfedcba  delay(2000);  int WaitTime = 10;  Serial.print(" >> Wait for next time --> ");  for (int i = WaitTime; i >= 0; i -= 5) {  Serial.print(",");  Serial.print(i);  delay(5000);  }  }  } |

|  |  |
| --- | --- |
|  |  |
|  |
|  | |