การควบคุมเครื่องจักรอัจฉริยะโดยใช้การสื่อสารระหว่างเครื่องจักรกับเครื่องจักร
M2M – Intelligence Machine Control

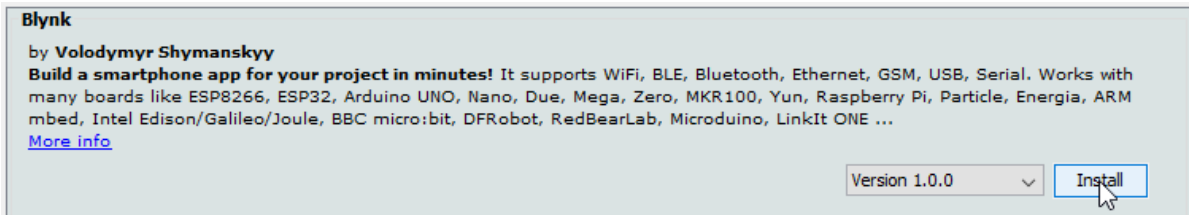
4/4 – Control and Monitor Modbus Device via The IoTs Platform

- การโปรแกรมใช้งาน Blynk เพื่อตรวจสอบ/สั่งงาน อุปกรณ์ Modbus
- การโปรแกรมใช้งาน Ubidots เพื่อตรวจสอบ/สั่งงาน อุปกรณ์ Modbus
- คำถามท้ายบทเพื่อทดสอบความเข้าใจ

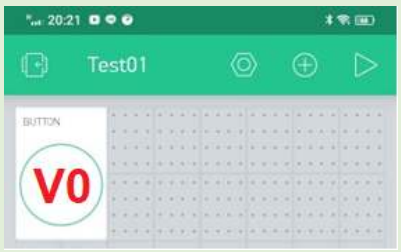
การโปรแกรมใช้งาน Blynk เพื่อตรวจสอบ/สั่งงาน อุปกรณ์ Modbus

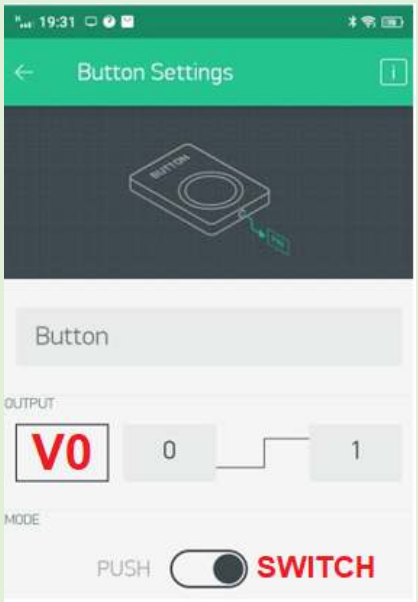
Test 1/7. Remote Control ESP-32 via Blynk

1. ตรวจสอบให้แน่ใจว่ามี Blynk Library เรียบร้อยแล้ว



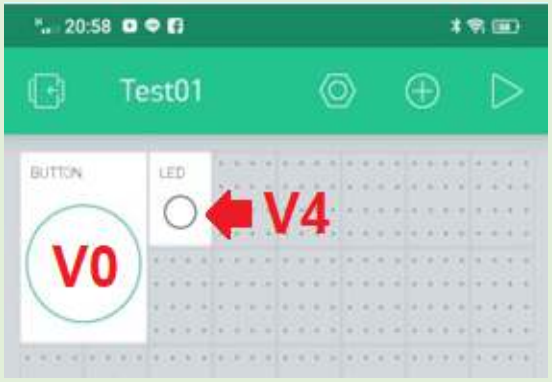
2. ทดสอบการควบคุมผ่าน Blynk ไปยัง Build in LED

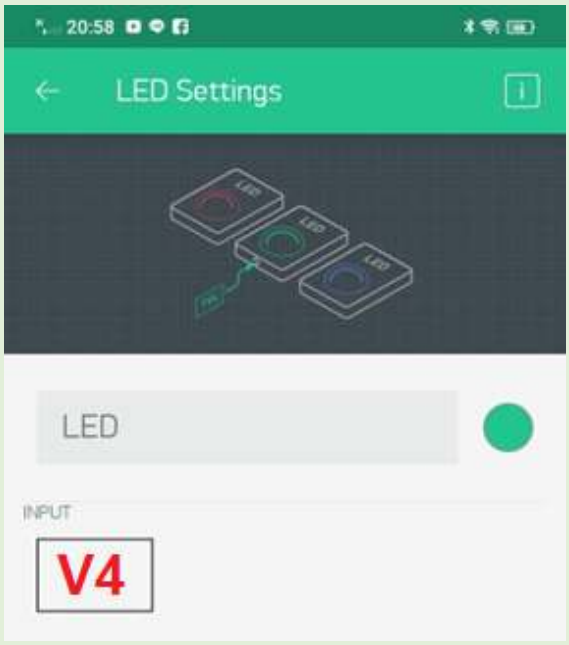





```
#define BLYNK_PRINT Serial

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

#define testLED 2
#define testSW  0

char auth[] = "GzUREJbWPpqdufO2I4WFA4kT4FjTh_AT"; // Token Key
char ssid[] = "Test1234";                         // AP Name
char pass[] = "0816601929";                       // Wifi-Password

BLYNK_WRITE (V0) {
  int pinValue = param.asInt();
  digitalWrite(testLED, pinValue);
  if (pinValue == 1) {
    Serial.println("Force: On");
  } else {
    Serial.println("Force: Off");
  }
}

void setup() {
  Serial.begin(115200);
  pinMode(testLED, OUTPUT);
  Blynk.begin(auth, ssid, pass);
}

void loop() {
  Blynk.run();
}
```

3. ทดสอบอ่านค่า SW0 และควบคุม D2 พร้อมกันผ่าน Blynk

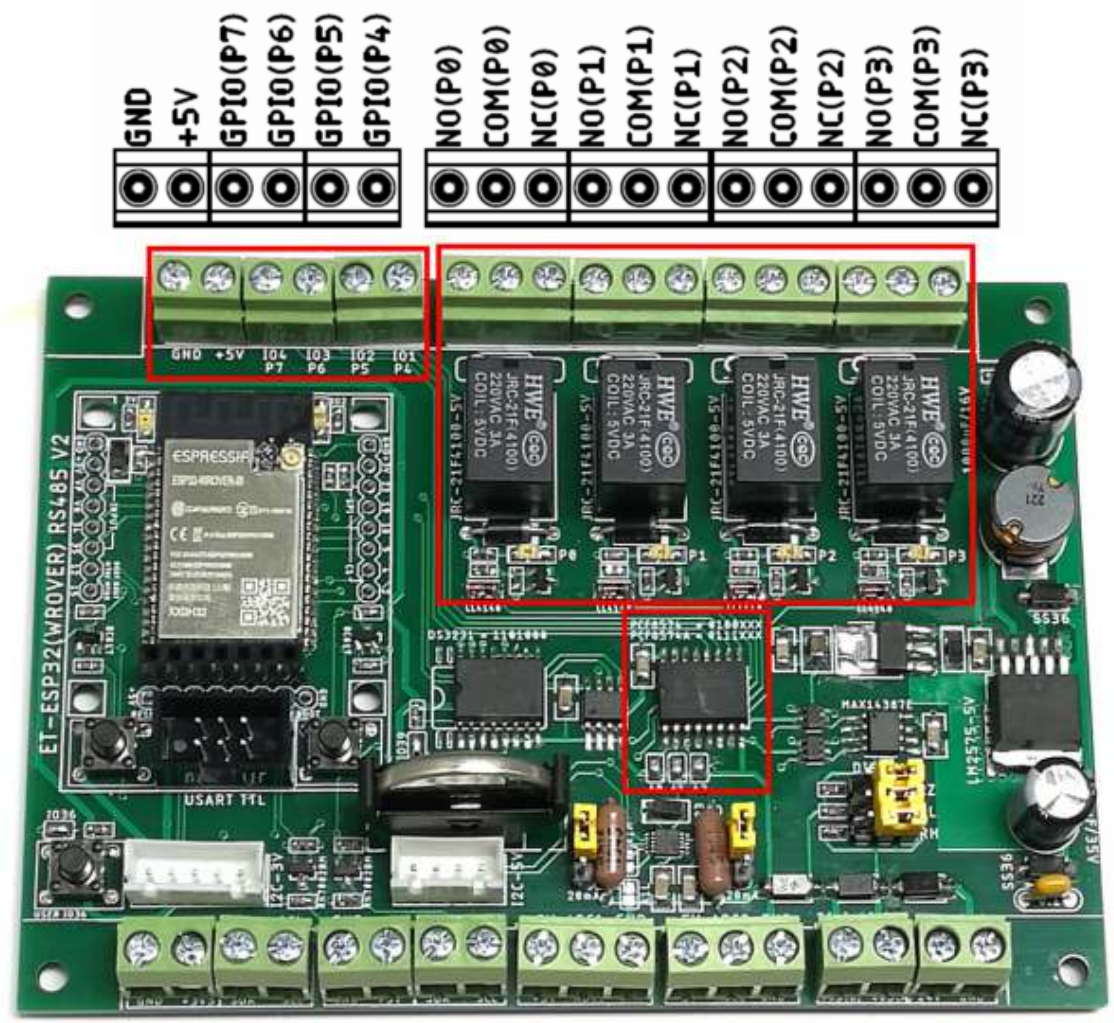





```
#define BLYNK_PRINT Serial

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>

#define testLED 2
#define testSW  0

char auth[] = "GzUREJbWPpqdufO2I4WFA4kT4FjTh_AT";  // Token Key
char ssid[] = "Test1234";                          // AP Name
char pass[] = "0816601929";                        // Wifi-Password

WidgetLED Switch0(V4);
BlynkTimer timer;

BLYNK_WRITE (V0) {
  int pinValue = param.asInt();
  digitalWrite(testLED, pinValue);
  if (pinValue == 1) {
    Serial.println("LED On");
  } else {
    Serial.println("LED Off");
  }
}

void loopReadInput()
{ bool stsRead = digitalRead(testSW);
  if (stsRead == 0) {
    Switch0.on();  Serial.println("Sw0-Press");
  } else {
    Switch0.off(); Serial.println("Sw0-Release");
  }
}


void setup() {
  Serial.begin(115200);
  pinMode(testLED, OUTPUT);
  Blynk.begin(auth, ssid, pass);
  timer.setInterval(1000L, loopReadInput);
}

void loop() {
  Blynk.run();
  timer.run();
}
```

Test 2/7. Remote Control and Monitor ET-ESP32-RS485 V2 Board via Blynk

4. บอร์ด ET-ESP32-RS485 V2 เป็นบอร์ดที่เหมาะในการแปลงการสื่อสารระหว่าง Modbus กับ IoT เนื่องจากมีทั้งช่องทางสื่อสารทำให้เรียกใช้งานได้ทันที



- P0-P3 ของ PCF8574/A ควบคมุ การทำงานของ Relay(P0) - Relay(P3)
- P4-P7 ต่อกับ GPIO(P4)- GPIO(P7) สามารถกำหนดเป็น Input/Output ได้อิสระจากโปรแกรม
- PCF8574 Address=0x20, I2C_SCL_Pin D22, I2C_SDA_Pin D21

5. Add WROVER Chip ให้ Arduino IDE (หากกำหนดแล้วให้ข้าม)
   - File → Preference … กำหนด https://dl.espressif.com/dl/package_esp32_index.json
   - Tools → Board → Board Manager … ทำการเพิ่ม esp32 บอร์ด
   - Add WROVER Chip ให้ Arduino IDE (หากกำหนดแล้วให้ข้าม)

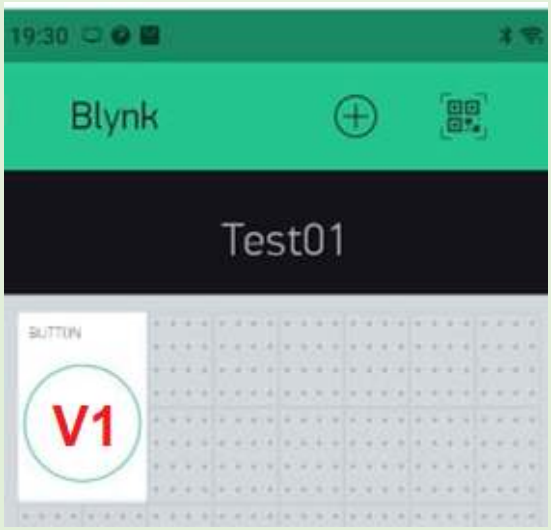
6. เลือกบอร์ดเป็น ESP32 Wrover Module ทดสอบการทำงานด้วย Blink Example Code

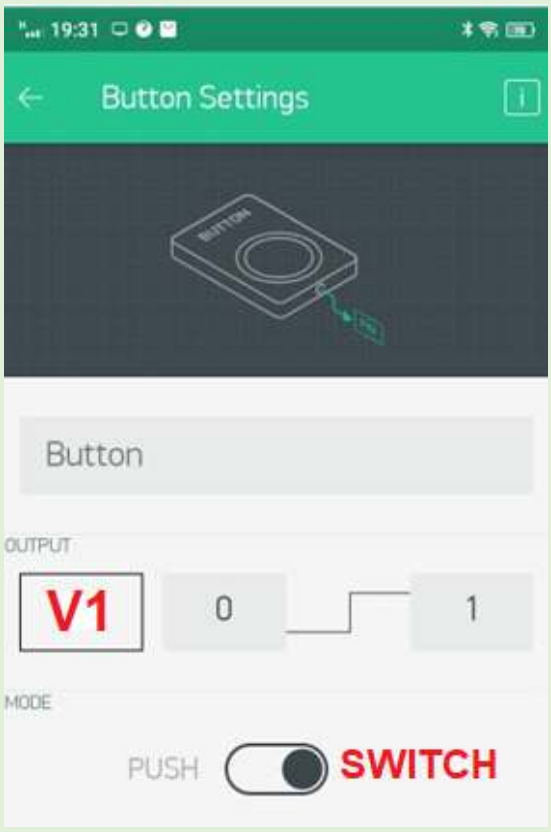
```
#define testLED 2
void setup() {
  pinMode(testLED, OUTPUT);
}

void loop() {
  digitalWrite(testLED, HIGH);  delay(1000);
  digitalWrite(testLED, LOW);   delay(1000);
}
```

7. ควบคุมการทำงานของ Relay

```
#include <Wire.h>

#define PCF8574_Addr 0x20
#define I2C_SCL_Pin 22
#define I2C_SDA_Pin 21
#define testLED      2

void setup() {
  pinMode(testLED, OUTPUT);
  Wire.begin(I2C_SDA_Pin, I2C_SCL_Pin);
}

void loop() {
  digitalWrite(testLED, HIGH);
  Wire.beginTransmission(PCF8574_Addr);
  Wire.write(0xAA);
  Wire.endTransmission();
  delay(1000);

  digitalWrite(testLED, LOW);
  Wire.beginTransmission(PCF8574_Addr);
  Wire.write(0x55);
  Wire.endTransmission();
  delay(1000);
}
```

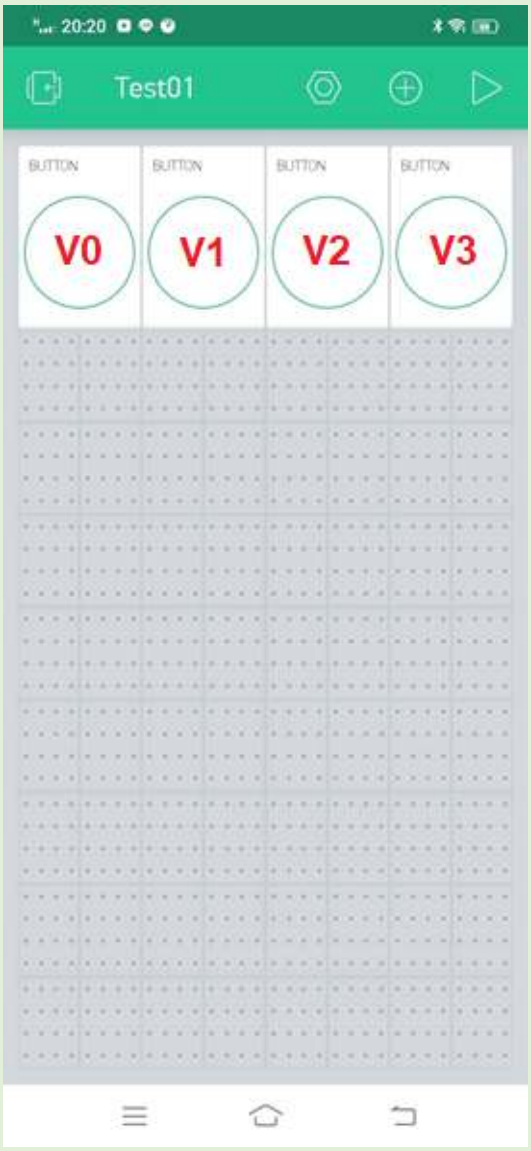
8. ทดสอบการควบคุมผ่าน Blynk ไปยัง PCF8517 Relay_0





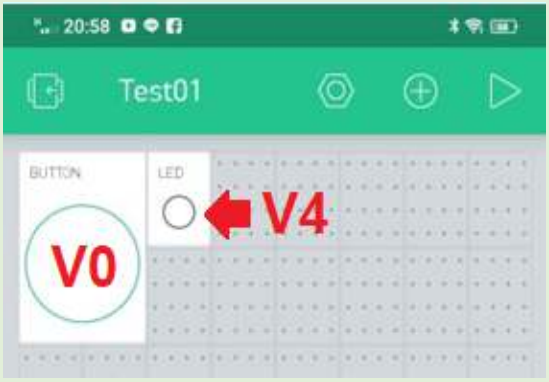

```
#define BLYNK_PRINT Serial

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <Wire.h>

#define PCF8574_Addr 0x20
#define I2C_SCL_Pin 22
#define I2C_SDA_Pin 21
#define testLED 2

char auth[] = "GzWPpqdufO2I4WFA4kT4FjTh_AT";          // Token Key
char ssid[] = "Test1234";                             // AP Name
char pass[] = "0816601234";                           // Wifi-Password

byte statusRelay = 0x0ff; // All Off

void RelayUpdate(int idRelay, int stsRelay) {
  byte stsRelayTemp;
  Serial.print("Relay_");
  Serial.print(idRelay);
  if (stsRelay == 1) {
    stsRelayTemp = ~(1 << idRelay);
    statusRelay &= stsRelayTemp;
    Serial.println("Force: On");
  }
  else {
    stsRelayTemp = 1 << idRelay;
    statusRelay |= stsRelayTemp;
    Serial.println("Force: Off");
  }
  Wire.beginTransmission(PCF8574_Addr);
  Wire.write(statusRelay);
  Wire.endTransmission();
}

BLYNK_WRITE (V0) {
  int pinValue = param.asInt();
  RelayUpdate(0, pinValue);
  digitalWrite(testLED, pinValue);
}

void setup() {
  Serial.begin(115200);
  pinMode(testLED, OUTPUT);
  Blynk.begin(auth, ssid, pass);
  Wire.begin(I2C_SDA_Pin, I2C_SCL_Pin);
}

void loop() {
  Blynk.run();
}
```

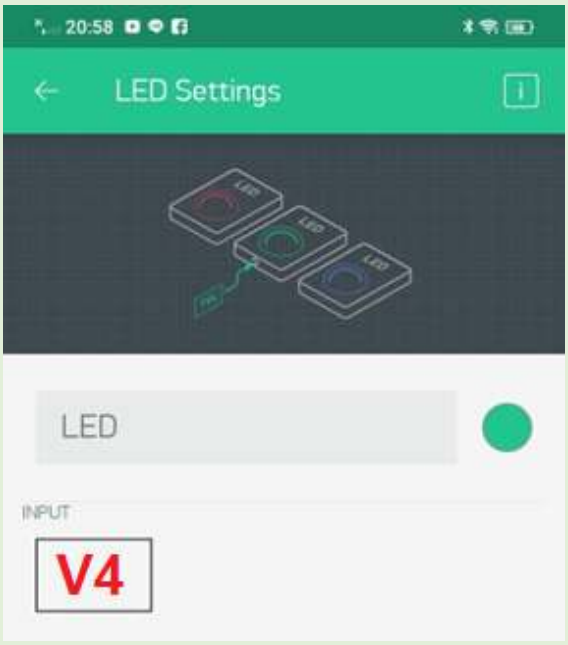
9. ปรับแก้เป็นควบคุม 4-Switch สำหรับ 4-Relay




```
#define BLYNK_PRINT Serial

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <Wire.h>

#define PCF8574_Addr 0x20
#define I2C_SCL_Pin 22
#define I2C_SDA_Pin 21
#define testLED 2

char auth[] = "GzWPpqdufO2I4WFA4kT4FjTh_AT";          // Token Key
char ssid[] = "Test1234";                             // AP Name
char pass[] = "0816601234";                           // Wifi-Password

byte statusRelay = 0x0ff; // All Off

void RelayUpdate(int idRelay, int stsRelay) {
  byte stsRelayTemp;
  Serial.print("Relay_");
  Serial.print(idRelay);
  if (stsRelay == 1) {
    stsRelayTemp = ~(1 << idRelay);
    statusRelay &= stsRelayTemp;
    Serial.println("Force: On");
  }
  else {
    stsRelayTemp = 1 << idRelay;
    statusRelay |= stsRelayTemp;
    Serial.println("Force: Off");
  }
  Wire.beginTransmission(PCF8574_Addr);
  Wire.write(statusRelay);
  Wire.endTransmission();
}

BLYNK_WRITE (V0) {
  int pinValue = param.asInt();
  RelayUpdate(0, pinValue);
  digitalWrite(testLED, pinValue);
}

BLYNK_WRITE (V1) {
  int pinValue = param.asInt();
  RelayUpdate(1, pinValue);
}

BLYNK_WRITE (V2) {
  int pinValue = param.asInt();
  RelayUpdate(2, pinValue);
}

BLYNK_WRITE (V3) {
  int pinValue = param.asInt();
  RelayUpdate(3, pinValue);
}


void setup() {
  Serial.begin(115200);
  pinMode(testLED, OUTPUT);
  Blynk.begin(auth, ssid, pass);
  Wire.begin(I2C_SDA_Pin, I2C_SCL_Pin);
}

void loop() {
  Blynk.run();
}
```

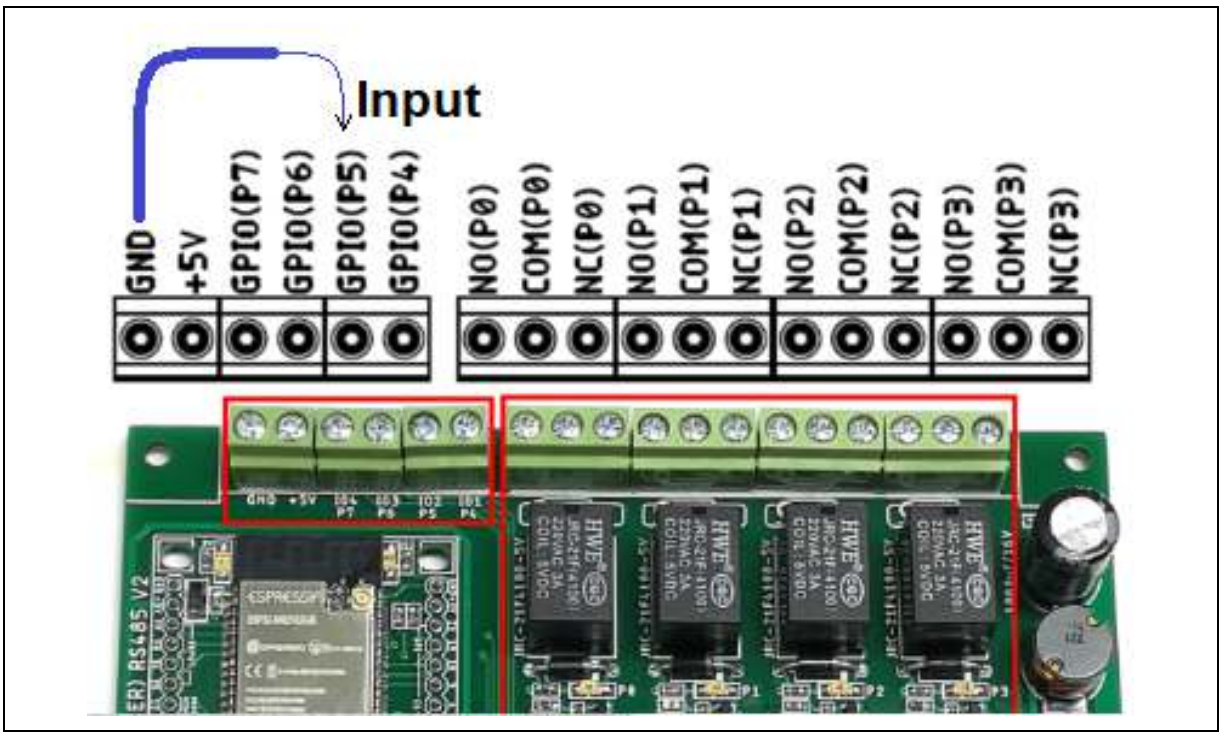
10. ทดสอบอ่านค่า GPIO(P4) และควบคุม Relay-0 พร้อมกันผ่าน Blynk





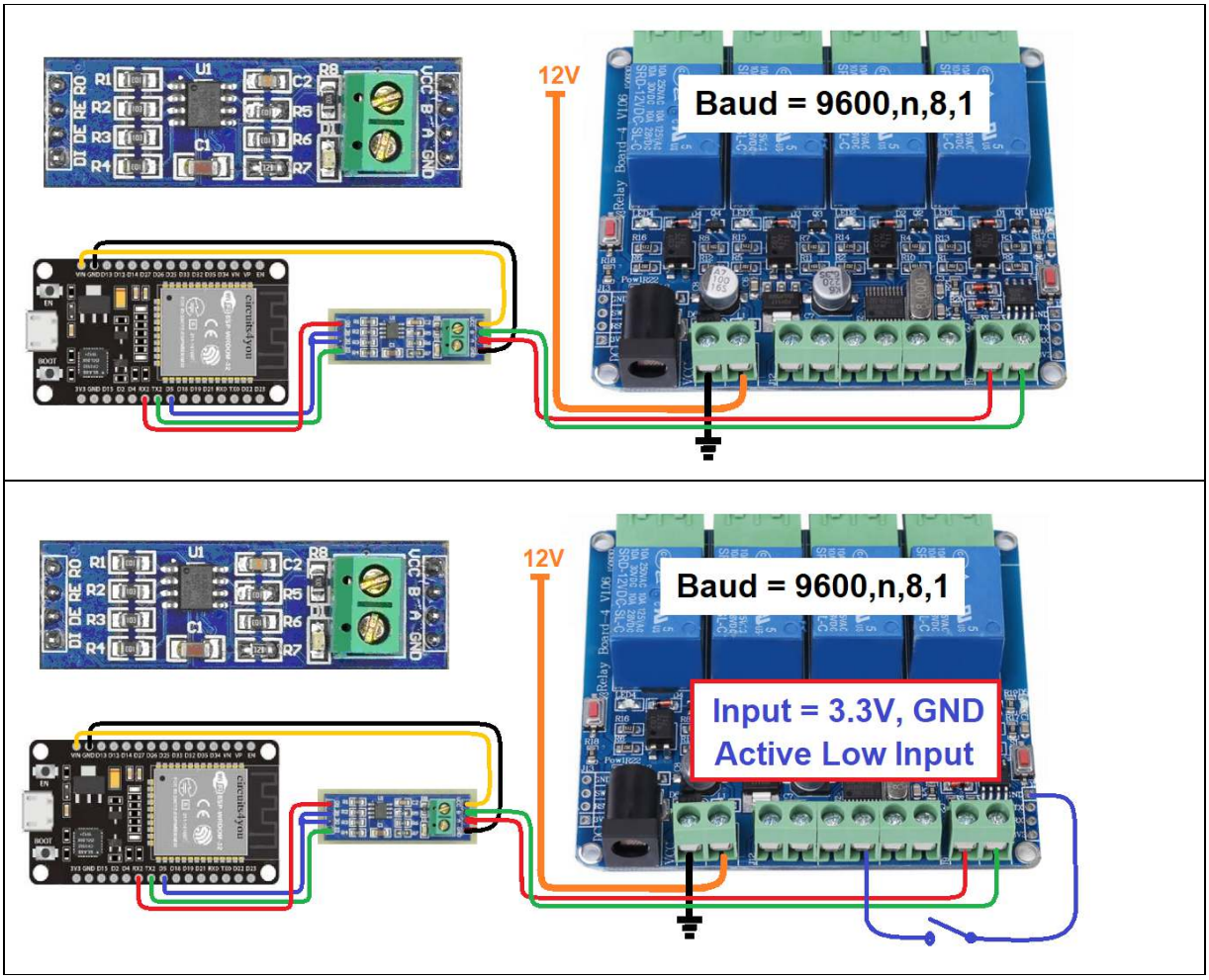

```
#define BLYNK_PRINT Serial

#include <WiFi.h>
#include <WiFiClient.h>
#include <BlynkSimpleEsp32.h>
#include <Wire.h>

#define PCF8574_Addr 0x20
#define I2C_SCL_Pin 22
#define I2C_SDA_Pin 21
#define testLED 2

char auth[] = "GzWPpqdufO2I4WFA4kT4FjTh_AT";          // Token Key
char ssid[] = "Test1234";                             // AP Name
char pass[] = "0816601234";                           // Wifi-Password

byte statusRelay = 0x0ff; // All Off

WidgetLED Input4(V4);
BlynkTimer timer;

//-----------------------------------------
void expanderWrite(byte _data ) {
 Wire.beginTransmission(PCF8574_Addr);
 Wire.write(_data);
 Wire.endTransmission();
}
//-----------------------------------------
byte expanderRead() {
 byte _data;
 Wire.requestFrom(PCF8574_Addr, 1);
 if (Wire.available()) {
  _data = Wire.read();
 }
 return _data;
}
//-----------------------------------------
void RelayUpdate(int idRelay, int stsRelay) {
 byte stsRelayTemp;
 Serial.print("Relay_");
 Serial.print(idRelay);
 if (stsRelay == 1) {
  stsRelayTemp = ~(1 << idRelay);
  statusRelay &= stsRelayTemp;
  Serial.println("Force: On");
 }
 else {
  stsRelayTemp = 1 << idRelay;
  statusRelay |= stsRelayTemp;
  Serial.println("Force: Off");
 }
 expanderWrite(statusRelay);
}
//-----------------------------------------
BLYNK_WRITE (V0) {
 int pinValue = param.asInt();
 RelayUpdate(0, pinValue);
 digitalWrite(testLED, pinValue);
}
//-----------------------------------------
void loopReadInput()
{ byte stsRead = expanderRead();
 if ((stsRead >> 4) & 1) {
  Input4.off();   Serial.println("Input V4: off");
 } else {
  Input4.on();   Serial.println("Input V4: on");
 }
}
//=========================================
void setup() {
 Serial.begin(115200);
 pinMode(testLED, OUTPUT);
 Blynk.begin(auth, ssid, pass);
 Wire.begin(I2C_SDA_Pin, I2C_SCL_Pin);
 timer.setInterval(1000L, loopReadInput);
}

void loop() {
 Blynk.run();
 timer.run();
}
```

11. การอ่านค่าจาก Input Connector

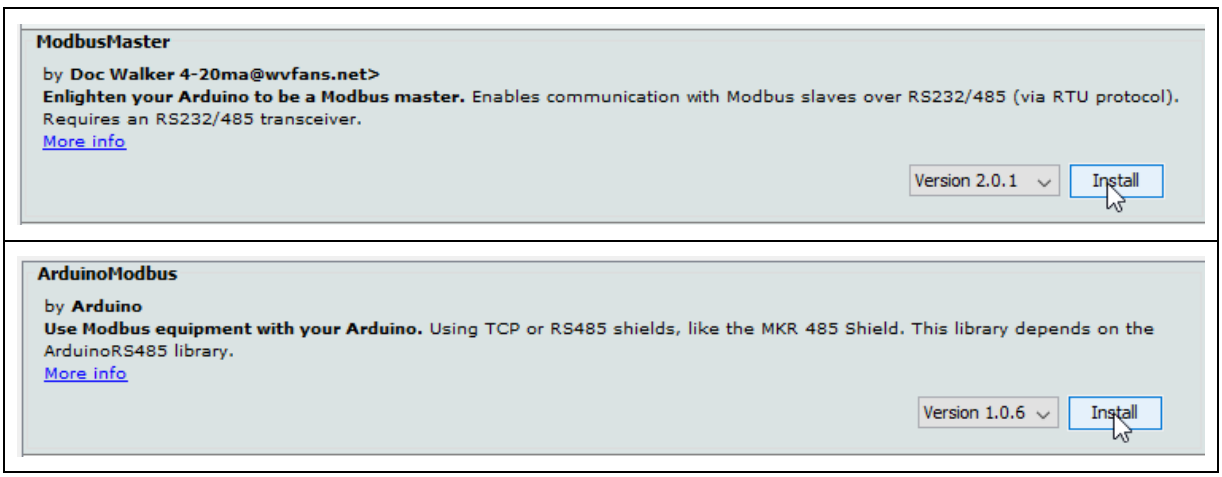


Test 3/7. Remote monitor and control Modbus RTU device via Blynk

12. วงจรทดสอบการควบคุม และวงจรการอ่านค่า กับ Modbus Device



13. การเพิ่ม Library >> Arduino Modbus by Arduino V 1.0.6
14. การเพิ่ม Library >> Modbus Master by Doc Walker V 2.0.1

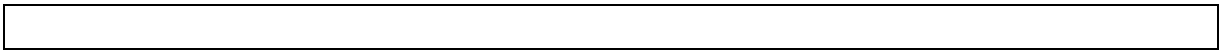


15. ทดสอบการควบคุม Modbus Device แบบ Direct Code ไม่ใช้ Library และแบบใช้ Modbus Master Lib.

```
// Code without Library
#define RS485Transmit   HIGH
#define RS485Receive    LOW
#define RS485Control    5   //RS485 Direction control
#define Pin_LEDMonitor  2

byte Board_ID = 0x05; // ID = 5
byte Mdbs_Cmd = 0x05; // Command 05
byte H_RelayID = 0x00;
byte L_RelayID = 0x00;
byte Relay_On  = 0x01;  // On = 0100
byte Relay_Off = 0x00;  // Off = 0000
byte OnOff_Dly = 0x00;
byte HByte_CRC = 00;
byte LByte_CRC = 00;

int StepConut = 0;
byte Echo[20];

uint16_t CRC16_Update(uint16_t tempCRC, uint8_t inData) {
 tempCRC ^= inData;
 for (int i = 0; i < 8; ++i)
  if (tempCRC & 1)  tempCRC = (tempCRC >> 1) ^ 0xA001;
  else          tempCRC = (tempCRC >> 1);
 return tempCRC;
}

uint16_t SendByte_CRCUpdate(uint16_t tempCRC, uint8_t inData) {
 Serial2.write(inData);
 if (inData < 0x10) Serial.print("0");
 Serial.print(inData, HEX);
 Serial.print(" ");
 tempCRC = CRC16_Update(tempCRC, inData);
 return tempCRC;
}

void RTU_RelayCtrl(int rly_ID, byte rly_Cmd) {
 uint16_t Calc_CRC = 0xffff; // the initial value
 H_RelayID = highByte(rly_ID);
 L_RelayID = lowByte(rly_ID);
 digitalWrite(Pin_LEDMonitor, HIGH);
 digitalWrite(RS485Control, RS485Transmit);  delay(10);
 Calc_CRC = SendByte_CRCUpdate(Calc_CRC, Board_ID);
 Calc_CRC = SendByte_CRCUpdate(Calc_CRC, Mdbs_Cmd);
 Calc_CRC = SendByte_CRCUpdate(Calc_CRC, H_RelayID);
 Calc_CRC = SendByte_CRCUpdate(Calc_CRC, L_RelayID);
 Calc_CRC = SendByte_CRCUpdate(Calc_CRC, rly_Cmd);
 Calc_CRC = SendByte_CRCUpdate(Calc_CRC, OnOff_Dly);
 HByte_CRC = highByte(Calc_CRC);
 LByte_CRC = lowByte(Calc_CRC);
 Calc_CRC = SendByte_CRCUpdate(Calc_CRC, LByte_CRC);
 Calc_CRC = SendByte_CRCUpdate(Calc_CRC, HByte_CRC);
 delay(10);
 digitalWrite(RS485Control, RS485Receive);
 digitalWrite(Pin_LEDMonitor, LOW);
 Serial.println();
}

void setup()  {
 pinMode(Pin_LEDMonitor, OUTPUT);
 pinMode(RS485Control, OUTPUT);
 Serial.begin(9600);
 Serial2.begin(9600);
 digitalWrite(RS485Control, RS485Receive);
 Serial.println("Start Test MODBUS RTU");
}

void loop()  {
 RTU_RelayCtrl(0, Relay_On);    delay(3000);
 RTU_RelayCtrl(1, Relay_On);    delay(3000);
 RTU_RelayCtrl(2, Relay_On);    delay(3000);
 RTU_RelayCtrl(3, Relay_On);    delay(3000);
 RTU_RelayCtrl(0, Relay_Off);   delay(3000);
 RTU_RelayCtrl(1, Relay_Off);   delay(3000);
 RTU_RelayCtrl(2, Relay_Off);   delay(3000);
 RTU_RelayCtrl(3, Relay_Off);   delay(3000);
}
```

```
// Code with ModbusMaster Library
#include <ModbusMaster.h>

#define MAX485_Monitor 2
#define MAX485_Ctrl 5 // Pin Ctrl 1=Tx and 0=Rx_NEG
#define MAX485_Rx 16  // Pin RXD2 16
#define MAX485_Tx 17  // Pin TXD2 17
#define Slave_ID 5     // Slave ID
ModbusMaster node;  // instantiate ModbusMaster object

void preTransmission() {
 digitalWrite(MAX485_Monitor, 1);
 digitalWrite(MAX485_Ctrl, 1);
}

void postTransmission() {
 digitalWrite(MAX485_Monitor, 0);
 digitalWrite(MAX485_Ctrl, 0);
}

void setup() {
 pinMode(MAX485_Monitor, OUTPUT);
 pinMode(MAX485_Ctrl, OUTPUT);
 postTransmission(); // Init in receive mode
 Serial.begin(115200);
 Serial2.begin(9600, SERIAL_8N1, MAX485_Rx, MAX485_Tx);
 node.begin(Slave_ID, Serial2); // Modbus slave ID Setting
 // Callbacks allow us to configure the RS485 transceiver correctly
 node.preTransmission(preTransmission);
 node.postTransmission(postTransmission);
}
int state = 1;

void loop() {
 for (int i = 0; i <= 3; i++) {
  Serial.println("Addr-" + (String)(i) + " >> " + (String)(state));
  node.writeSingleCoil(i, state);
  delay(2000);
 }
 state = 1 - state;
}
```

16. วงจรทดสอบ การควบคุมและวงจรการอ่านค่า กับ Modbus Device



Test 4/7 Remote monitor and control Modbus RTU/ASCII/TCP device via Blynk

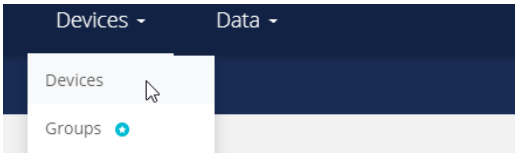
การโปรแกรมใช้งาน Ubidots เพื่อตรวจสอบ/สั่งงาน อุปกรณ์ Modbus

Test 5/7. Remote Control and Monitor via Ubidots IoTs Platform

1. Ubidots -- https://ubidots.com/

2. Signed Up (or Signed In) -- https://industrial.ubidots.com/accounts/signin/

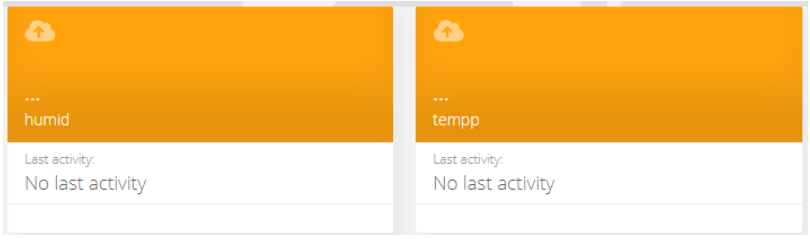
3. Create Device and Variable



Device -→ Device

Create Device
Blank Device

Device Name = pk007test
Device Label = pk007test
<all lowercase alphabet>

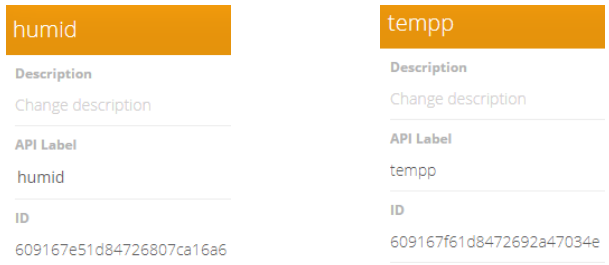


Click pk007test Device
Add Variable → raw

humid

tempp

<all lowercase alphabet>



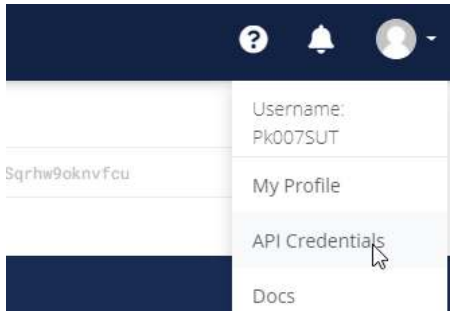
Click …

Set API Label = humid

Set API Label = tempp

<all lowercase alphabet>

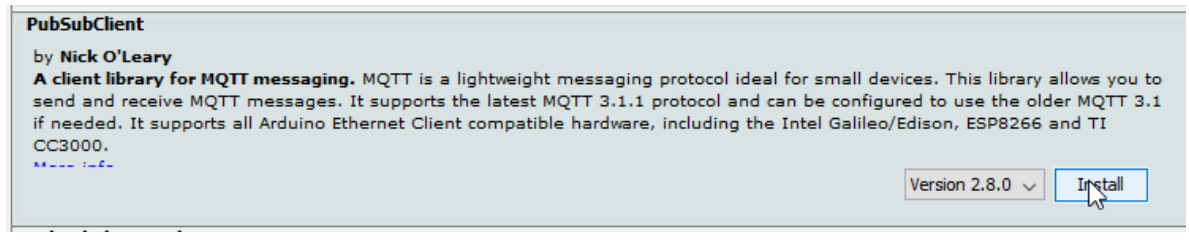
4. Get Tokens Key



API Credentials

Get Tokens Key

5. ตรวจสอบว่าติดตั้ง PubSubClient by Nick O'Leary – V2.8.0

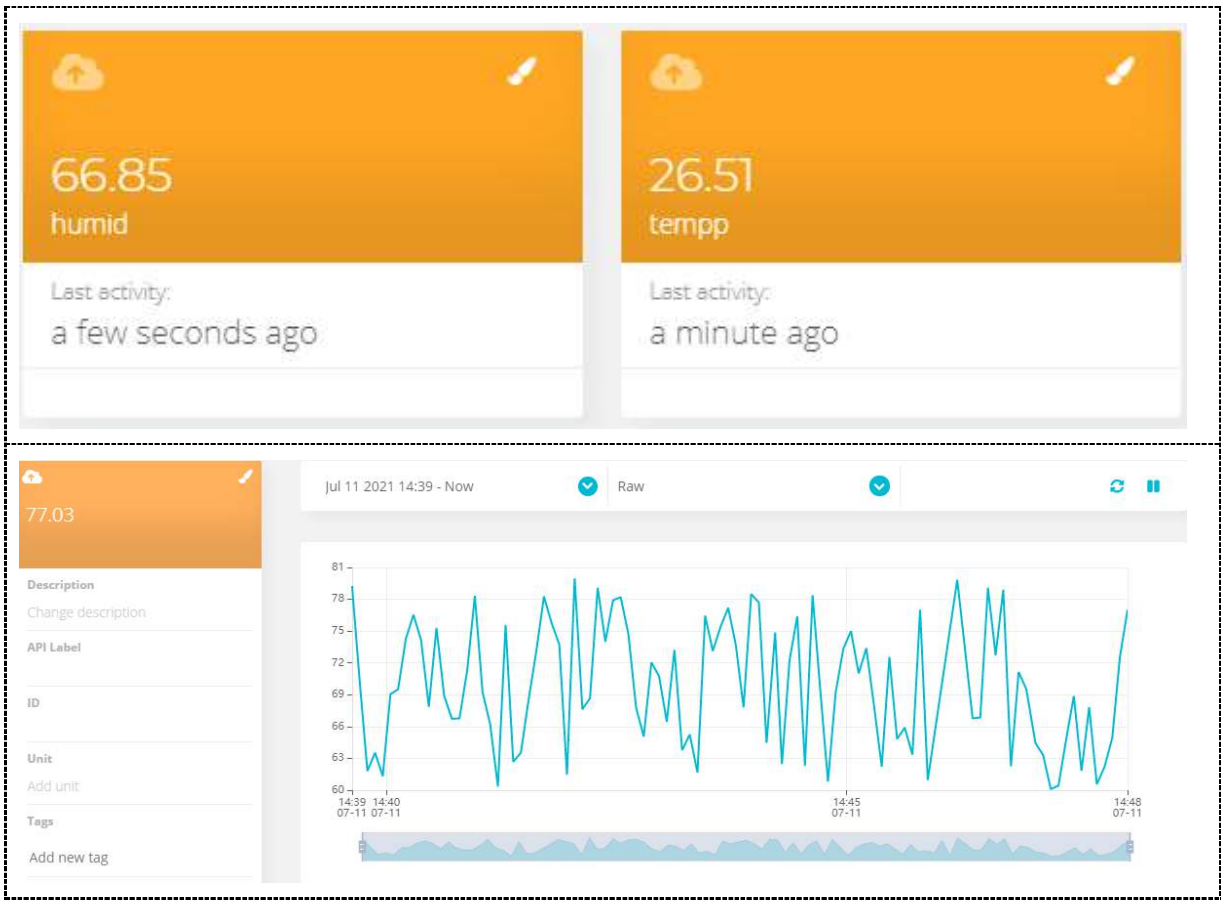


6. Coding Send Random Data to Ubidots

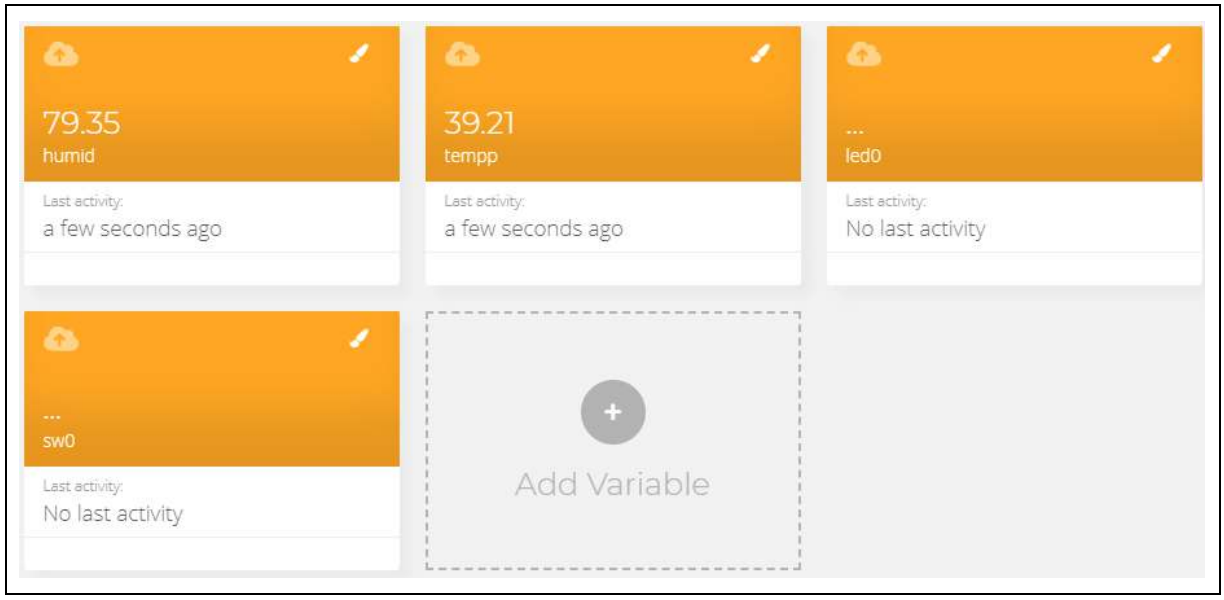
```
#include <WiFi.h>
#include <PubSubClient.h>

const char  *My_SSID = "Test1234";
const char  *My_Pass = "0816601929";
const char  *MQTT_Server = "things.ubidots.com";
const char  *MQTT_User = "BBFF-YgcQcuqoAiO9g46ehWh12TxVwyTjCx";
const char  *MQTT_Pass = "BBFF-YgcQcuqoAiO9g46ehWh12TxVwyTjCx";
const char  *PTopic1 = "/v2.0/devices/pk007test";
const char  *STopic1 = "/v2.0/devices/pk007test/humid";
const char  *STopic2 = "/v2.0/devices/pk007test/tempp";

#define MQTT_Port 1883

WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];

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 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]);
 }
 Serial.println();
}

void setup()
{ 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 xTempp = random(2000, 4000) / 100.0;
   float xHumid = random(6000, 8000) / 100.0;
   snprintf (msg, 75, "{ \"humid\" : %5.2f, \"tempp\": %5.2f}", xHumid, xTempp);
   Serial.print("Publish message: ");
   Serial.println(msg);
   client.publish(PTopic1, msg);
 }
}
```

```
Connecting to Test1234
.....
WiFi connected
IP address:
192.168.1.22
Attempting MQTT connection...connected
Message arrived [/v2.0/devices/pk007test/humid] {"value": 69.53, "timestamp": 1625989
Message arrived [/v2.0/devices/pk007test/tempp] {"value": 32.73, "timestamp": 1625989
Publish message: { "humid" : 64.44, "tempp": 30.80}
Message arrived [/v2.0/devices/pk007test/humid] {"value": 64.44, "timestamp": 1625989
Message arrived [/v2.0/devices/pk007test/tempp] {"value": 30.8, "timestamp": 16259896
Publish message: { "humid" : 63.31, "tempp": 34.42}
```



7. Update Code for monitor and control via Ubidots
8. Add 2 Variable → ledo and swo {ตัวเล็กเท่านั้น}

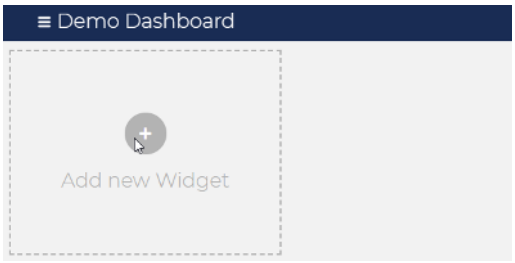
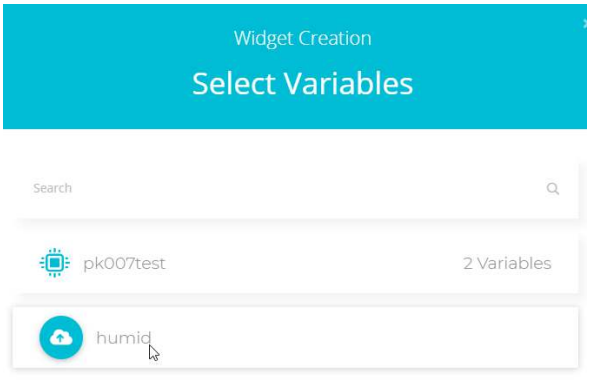


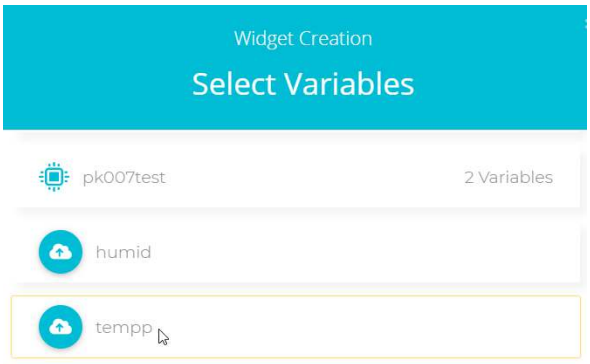
```
Message arrived [/v2.0/devices/pk007test/tempp] {"value": 32.01, "timestamp": 162599041340
Message arrived [/v2.0/devices/pk007test/sw0] {"value": 1.0, "timestamp": 1625990413404, '
Publish message: { "humid" : 72.10, "tempp": 24.22, "sw0": 0}
Message arrived [/v2.0/devices/pk007test/sw0] {"value": 0.0, "timestamp": 1625990418408, '
Message arrived [/v2.0/devices/pk007test/tempp] {"value": 24.22, "timestamp": 162599041840
Message arrived [/v2.0/devices/pk007test/humid] {"value": 72.1, "timestamp": 162599041840
Publish message: { "humid" : 74.82, "tempp": 31.04, "sw0": 1}
Message arrived [/v2.0/devices/pk007test/sw0] {"value": 1.0, "timestamp": 1625990423400, '
Message arrived [/v2.0/devices/pk007test/humid] {"value": 74.82, "timestamp": 162599042340
Message arrived [/v2.0/devices/pk007test/tempp] {"value": 31.04, "timestamp": 162599042340
```

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

const char  *My_SSID = "Test1234";
const char  *My_Pass = "0816601929";
const char  *MQTT_Server = "things.ubidots.com";
const char  *MQTT_User = "BBFF-YgcQcuqoAiO9g46ehWh12TxVwyTjCx";
const char  *MQTT_Pass = "BBFF-YgcQcuqoAiO9g46ehWh12TxVwyTjCx";
const char  *PTopic1 = "/v2.0/devices/pk007test";
const char  *STopic1 = "/v2.0/devices/pk007test/humid";
const char  *STopic2 = "/v2.0/devices/pk007test/tempp";
const char  *STopic3 = "/v2.0/devices/pk007test/led0";
const char  *STopic4 = "/v2.0/devices/pk007test/sw0";

#define MQTT_Port 1883

#define Test_LED0 2
#define Test_SW00 0

WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];

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);
    } 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_LED0, HIGH);
    else
```

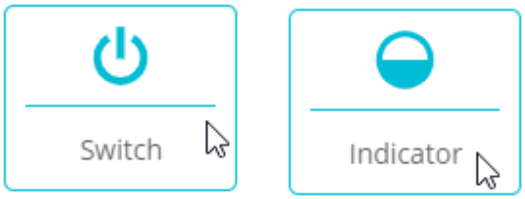
```
    digitalWrite(Test_LED0, LOW);
  }
  Serial.println();
}


void setup()
{ pinMode(Test_LED0,OUTPUT);
  pinMode(Test_SW00,INPUT_PULLUP);
  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 xTempp = random(2000, 4000) / 100.0;
    float xHumid = random(6000, 8000) / 100.0;
    int stsSW = digitalRead(Test_SW00);
    snprintf (msg, 75, "{ \"humid\" : %5.2f, \"tempp\": %5.2f, \"sw0\": %d}", xHumid, xTempp, stsSW);
    Serial.print("Publish message: ");
    Serial.println(msg);
    client.publish(PTopic1, msg);
  }
}
```

8. Create Dashboard

	Data Dashboard Add New Dashboard Add new Widget
	Line chart
	Add Value → humid



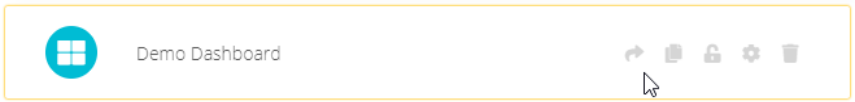
Line chart
Add Value → tempp



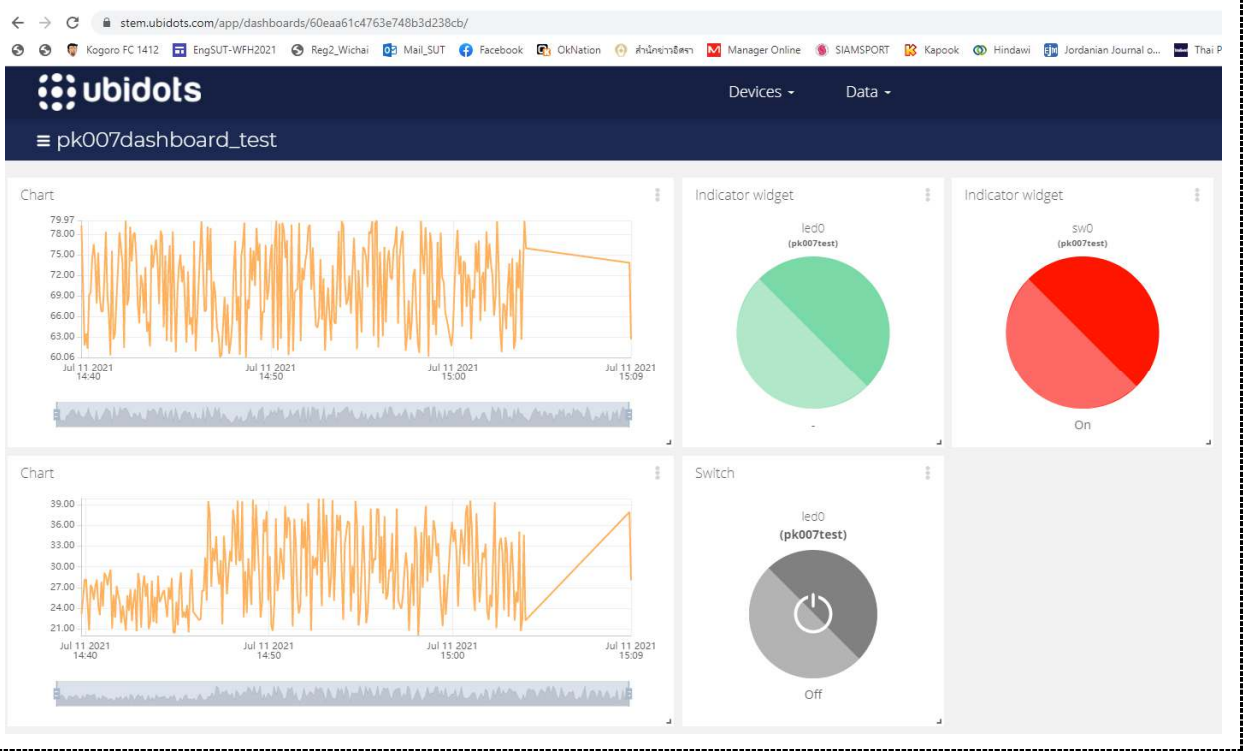
Switch
Add Value → led0

Indicator
Add Value → led0
Add Value → sw0

9. Test Dashboard and Create Share Dashboard Link

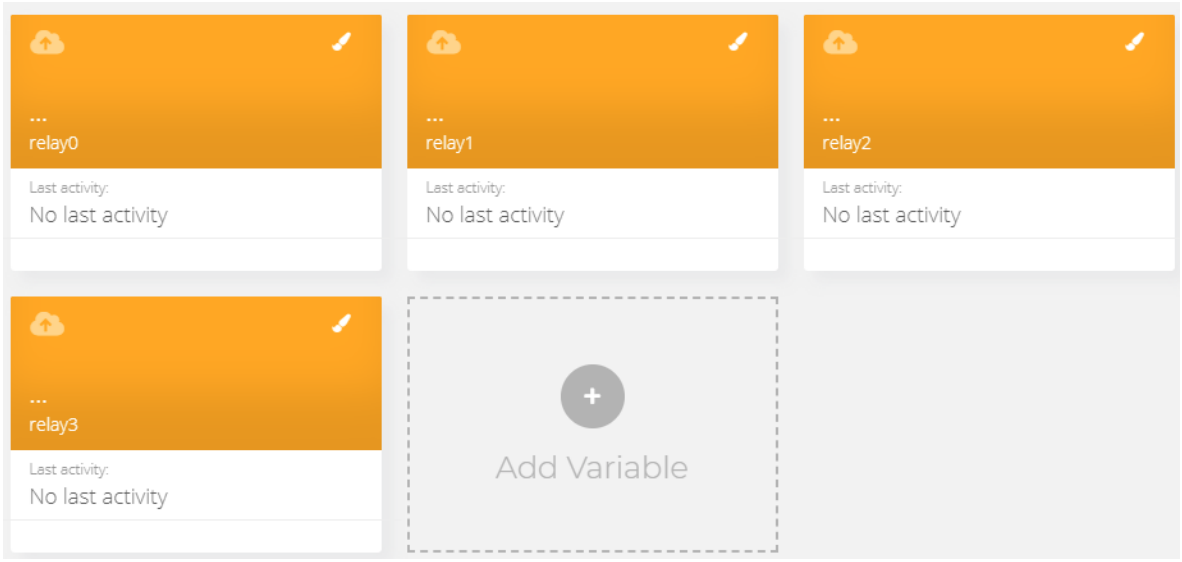
	Click Demo Dashboard
	Click Share Create Link

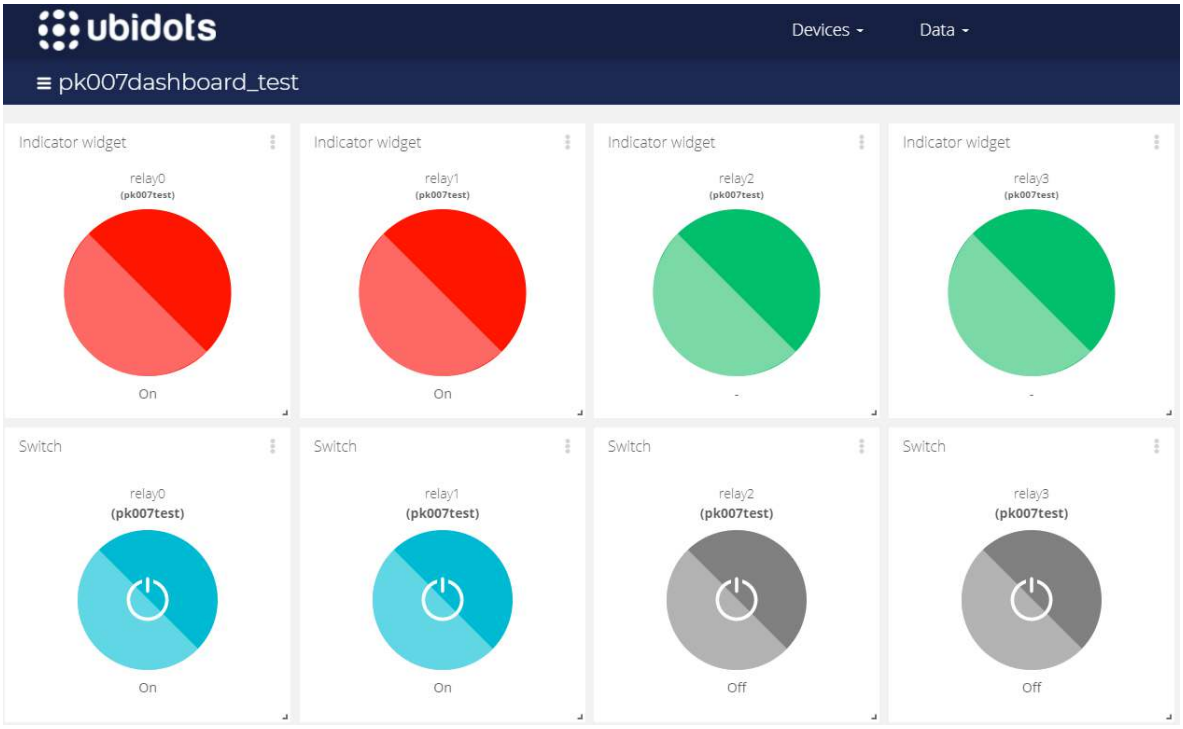
10. Test with Browser



Test 6/7. Remote monitor and control Modbus RTU device via Ubidots

9. Create Device → Variable {relay0, relay1, relay2, relay3}→ dashboard on Ubidots





10. Test This Code for Relay Control

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

#define MAX485_Monitor 2
#define MAX485_Ctrl 5 // Pin Ctrl 1=Tx and 0=Rx_NEG
#define MAX485_Rx 16  // Pin RXD2 16
#define MAX485_Tx 17  // Pin TXD2 17
#define Slave_ID 5     // Slave ID
ModbusMaster node;  // instantiate ModbusMaster object

const char  *My_SSID = "Test1234";
const char  *My_Pass = "0816601929";
const char  *MQTT_Server = "things.ubidots.com";
const char  *MQTT_User = "BBFF-YgcQcuqoAiO9g46ehWh12TxVwyTjCx";
const char  *MQTT_Pass = "BBFF-YgcQcuqoAiO9g46ehWh12TxVwyTjCx";
const char  *PTopic1 = "/v2.0/devices/pk007test";
const char  *STopic1 = "/v2.0/devices/pk007test/relay0";
const char  *STopic2 = "/v2.0/devices/pk007test/relay1";
const char  *STopic3 = "/v2.0/devices/pk007test/relay2";
const char  *STopic4 = "/v2.0/devices/pk007test/relay3";
#define MQTT_Port 1883
#define testLED   2
int stsLED = 0;

WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];

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);
   } 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]);
 }
 int RlyID  = (int)topic[29] - 0x30;   // '0'
 int RlySts = (int)payload[10] - 0x30; // '0'
 Serial.println("\nRlyID-" + (String)(RlyID) + " >> RlyStatus-" + (String)(RlySts));
 node.writeSingleCoil(RlyID, RlySts);
}

void preTransmission() {
 digitalWrite(MAX485_Monitor, 1);
 digitalWrite(MAX485_Ctrl, 1);
}

void postTransmission() {
 digitalWrite(MAX485_Monitor, 0);
 digitalWrite(MAX485_Ctrl, 0);
}
```

```
void setup()
{ pinMode(testLED, OUTPUT);

 pinMode(MAX485_Monitor, OUTPUT);
 pinMode(MAX485_Ctrl, OUTPUT);
 postTransmission(); // Init in receive mode
 Serial.begin(115200);
 Serial2.begin(9600, SERIAL_8N1, MAX485_Rx, MAX485_Tx);
 node.begin(Slave_ID, Serial2); // Modbus slave ID Setting
 // Callbacks allow us to configure the RS485 transceiver correctly
 node.preTransmission(preTransmission);
 node.postTransmission(postTransmission);

 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;
   digitalWrite(testLED, stsLED);
   stsLED = 1 - stsLED;
 }
}
```

```
Connecting to Test1234
.....
WiFi connected
IP address:
192.168.1.22
Attempting MQTT connection...connected
Message arrived [/v2.0/devices/pk007test/relay0] {"value": 1.0, "timestamp": 162599406757
RlyID-0 >> RlyStatus-1
Message arrived [/v2.0/devices/pk007test/relay2] {"value": 1.0, "timestamp": 162599407163
RlyID-2 >> RlyStatus-1
Message arrived [/v2.0/devices/pk007test/relay1] {"value": 1.0, "timestamp": 162599406918
RlyID-1 >> RlyStatus-1
Message arrived [/v2.0/devices/pk007test/relay3] {"value": 1.0, "timestamp": 162599407315
RlyID-3 >> RlyStatus-1
```

29 ↓ 0123456789..

01234...

↑10

11. Test Control and Monitor Modbus Device

Test 7/7. Remote monitor and control Modbus RTU/ASCII/TCP device via Ubidots

การควบคุมเครื่องจักรอัจฉริยะโดยใช้การสื่อสารระหว่างเครื่องจักรกับเครื่องจักร M2M - Intelligence Machine Control
ชื่อ-สกุล :

4/4: -- คำถามท้ายบทเพื่อทดสอบความเข้าใจ

Quiz_201 – xxxxxxx

- จาก Proximity Sensor นำเข้า PLC ส่งจำนวนที่นับได้เก็บที่ GSheet

Quiz_202 – xxxxxxx

- จาก Proximity Sensor นำเข้า PLC ส่งจำนวนที่นับได้เก็บที่ GSheet
- จาก PLC หากมีการกดปุ่ม Stop ให้ LINE Alert

Quiz_203 – xxxxxxx

Quiz_204 – xxxxxxx