การควบคุมเครื่องจักรอัจฉริยะโดยใช้การสื่อสารระหว่างเครื่องจักรกับเครื่องจักร

M2M - Intelligence Machine Control

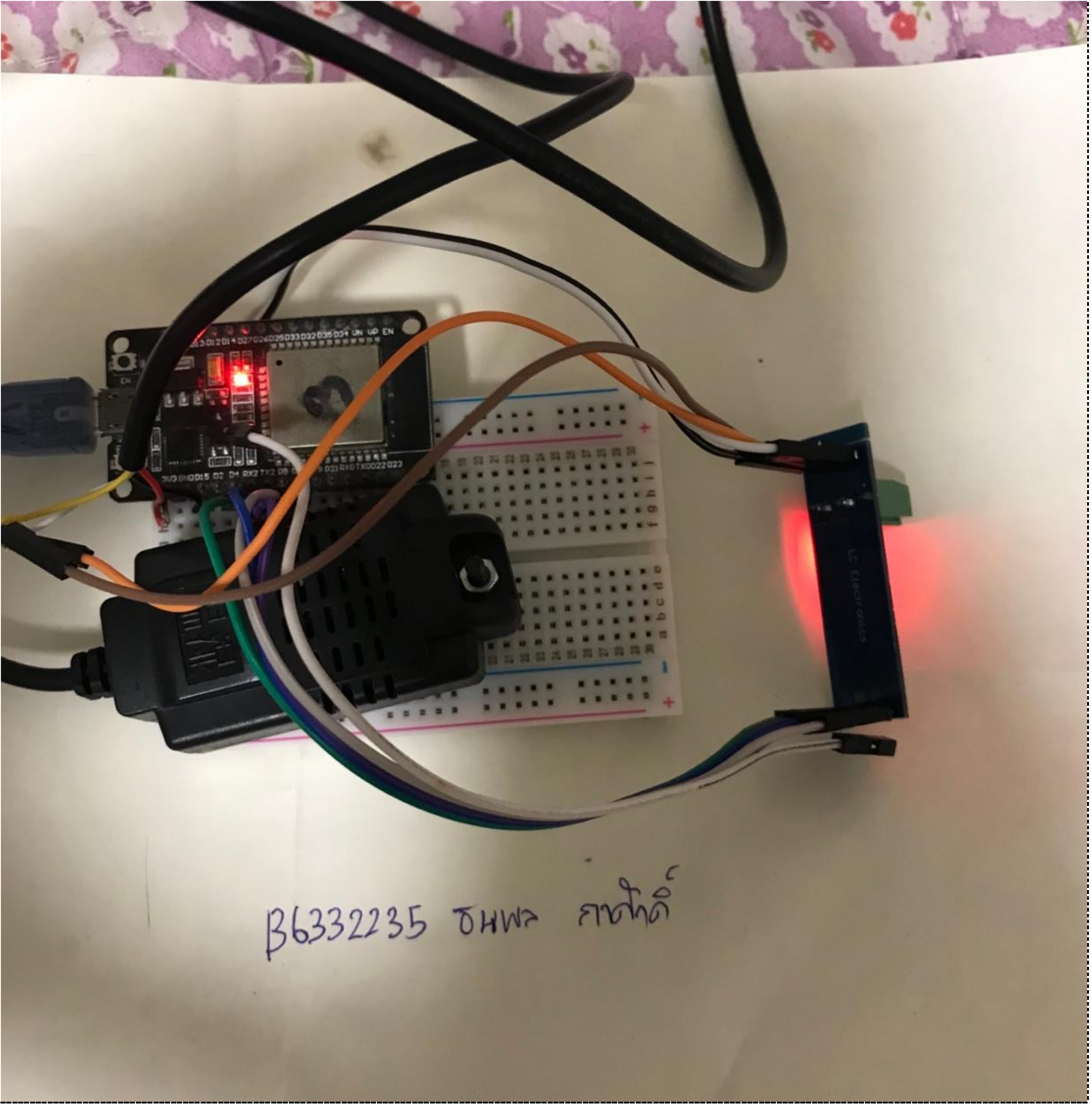
ขื่อ-สกุล : นายธนพล กาศักดิ์

4/4: -- คำถามท้ายบทเพื่อทดสอบความเข้าใจ

Quiz_201 – Read Modbus RTU

< รูปอุปกรณ์ที่ใช้ทดสอบ ขณะทำการทดสอบ >

< รูปอุปกรณ์ที่ใช้ทดสอบ ขณะทำการทดสอบ >



< โปรแกรมทดสอบ >

```
#define RS485Transmit HIGH
#define RS485Receive LOW
#define RS485Control 4 //RS485 Direction control
#define Pin_LEDMonitor 2int Wr_Index, StepConut = 0;
byte Request[] = {0x01, 0x03, 0x00, 0x00, 0x00, 0x02, 0xC4, 0x0B};
byte Echo[20];void setup() {
  pinMode(Pin_LEDMonitor, OUTPUT);
  pinMode(RS485Control, OUTPUT);
  Serial.begin(9600);
  Serial2.begin(9600);
  digitalWrite(RS485Control, RS485Receive);
  Serial.println("\nStart Test MODBUS RTU");
}
void loop() {
  Serial.print("\nTest(");
  Serial.print(++StepConut);
  Serial.print(") >>");
  digitalWrite(Pin_LEDMonitor, HIGH);
  digitalWrite(RS485Control, RS485Transmit);
  delay(10);
  for (int i = 0; i < sizeof(Request); i++) {
    Serial2.write(Request[i]);}
  delay(10);
  digitalWrite(RS485Control, RS485Receive);
  digitalWrite(Pin_LEDMonitor, LOW);
  Wr_Index = 0;
  for (long int i = 0; i < 300000; i++) {
    if (Serial2.available())
      Echo[Wr_Index++] = Serial2.read();
      if (Wr_Index > 12) i = 999999;
    delayMicroseconds(10);
  }

  for (int i = 0; i < Wr_Index - 1; i++) {
    Serial.print(" ");
    if (Echo[i] < 0x10) Serial.print("0");
    Serial.print(Echo[i], HEX);
  }

  float Tempp = (Echo[3] * 256 + Echo[4]) / 10.0;
  Serial.print(" > Tempp('C)="); Serial.print(Tempp, 2);
  float Humid = (Echo[5] * 256 + Echo[6]) / 10.0;
  Serial.print(" > Humid(%)="); Serial.print(Humid, 2);

  delay(5000);
}
```

< ผลการทดสอบ >

Quiz_202 – Write Modbus RTU

< รูปอุปกรณ์ที่ใช้ทดสอบ ขณะทำการทดสอบ >

< รูปอุปกรณ์ที่ใช้ทดสอบ ขณะทำการทดสอบ >

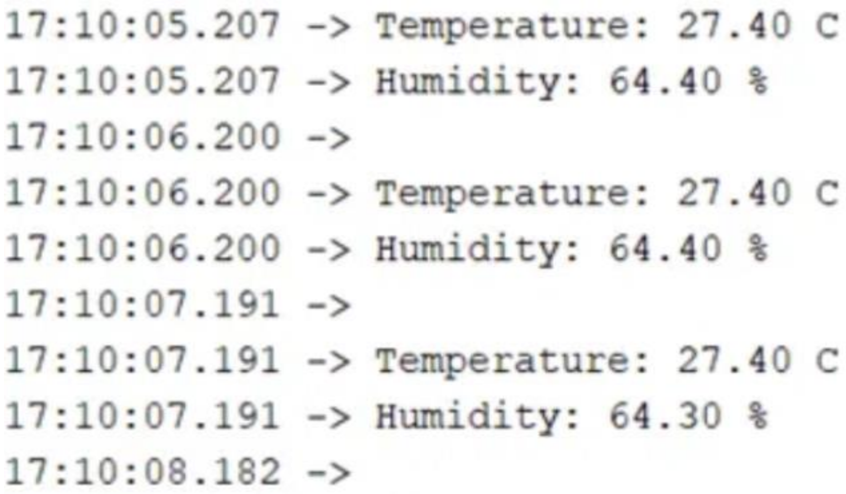
< โปรแกรมทดสอบ >

```
#define RS485Transmit HIGH
#define RS485Receive LOW
```

```
#define RS485Control 4 //RS485 Direction control
#define Pin_LEDMonitor 2byte Board_ID = 0x03;
byte Mdbs_Cmd = 0x06;
byte H_RelayID = 0x00;
byte L_RelayID = 0x03;
byte Relay_On = 0x01;
byte Relay_Off = 0x02;
byte OnOff_Dly = 0x00;
byte HByte_CRC = 00;
byte LByte_CRC = 00;
int StepConut = 0;
byte Echo[20];void setup() {
 pinMode(Pin_LEDMonitor, OUTPUT);
 pinMode(RS485Control, OUTPUT);
 Serial.begin(9600);
 Serial2.begin(9600);
 digitalWrite(RS485Control, RS485Receive);
 Serial.println("Start Test MODBUS RTU");
}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 loop() {
  for (int relay = 1; relay <= 8; relay++) {
    RTU_RelayCtrl(relay, Relay_On);
    delay(3000);
  }
  for (int relay = 1; relay <= 8; relay++) {
    RTU_RelayCtrl(relay, Relay_Off);
    delay(3000);
  }
}
```

< ผลการทดสอบ >

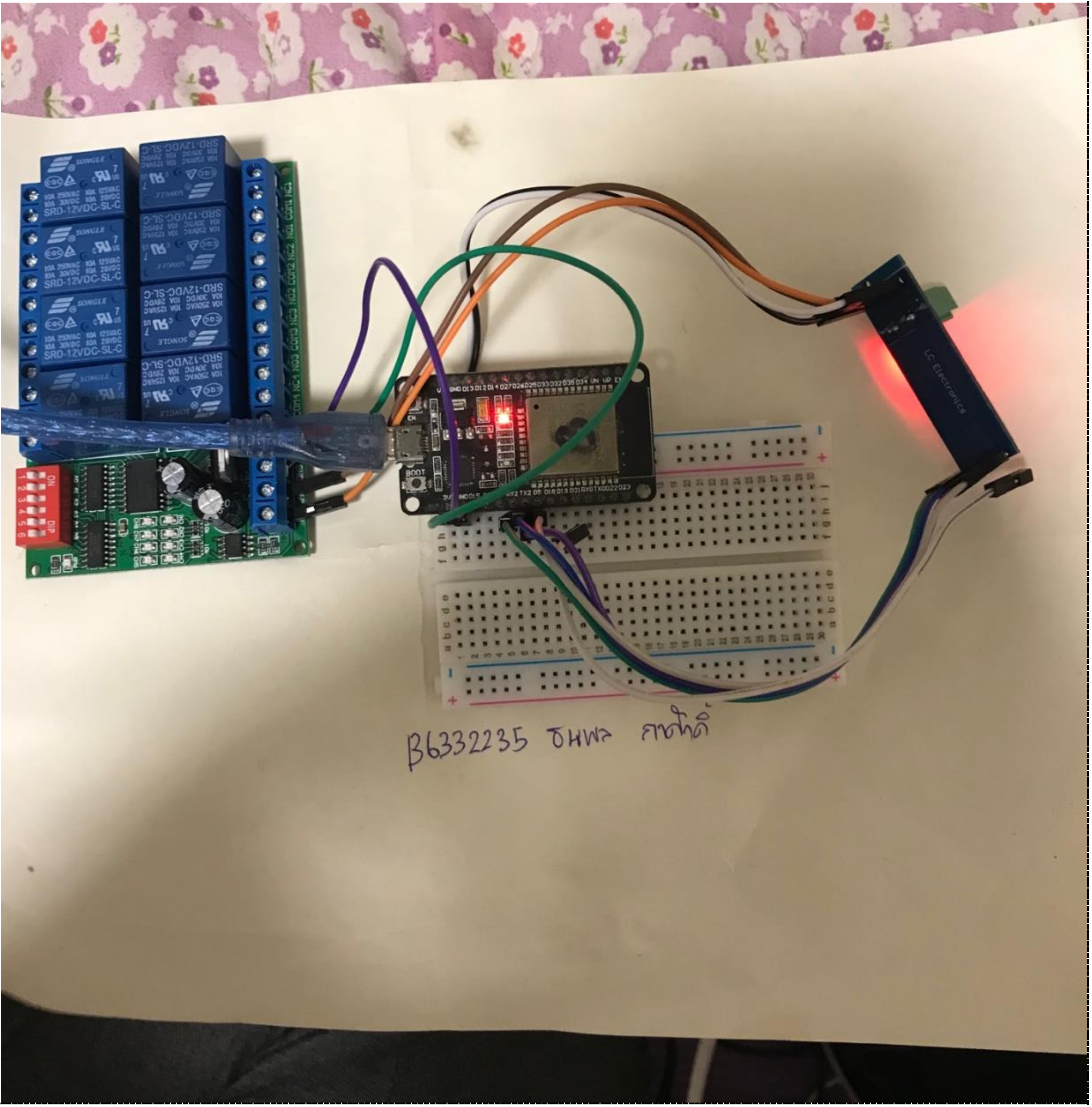


```
17:10:05.207 -> Temperature: 27.40 C
17:10:05.207 -> Humidity: 64.40 %
17:10:06.200 ->
17:10:06.200 -> Temperature: 27.40 C
17:10:06.200 -> Humidity: 64.40 %
17:10:07.191 ->
17:10:07.191 -> Temperature: 27.40 C
17:10:07.191 -> Humidity: 64.30 %
17:10:08.182 ->
```

Quiz_203 – Read/Write Modbus RTU

< รูปอุปกรณ์ที่ใช้ทดสอบ ขณะทำการทดสอบ >

< รูปอุปกรณ์ที่ใช้ทดสอบ ขณะทำการทดสอบ >

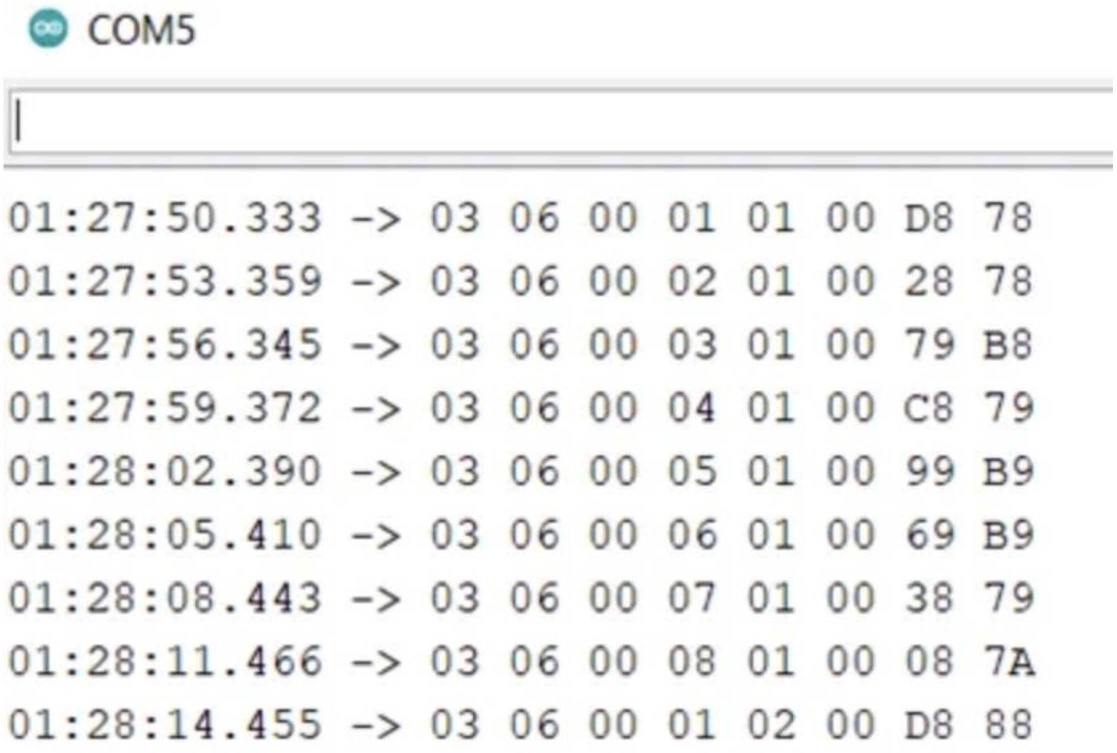


< โปรแกรมทดสอบ >

```
#include <ModbusMaster.h>

#define RS485Transmit HIGH

#define RS485Receive LOW

#define RS485Control 4 //RS485 Direction control

#define Pin_LEDMonitor 2

#define Slave_Sensor_ID 1
```

```
#define Slave_Relay8_ID 3
#define Slave_Ry4In4_ID 5
int state = 0;
float CTempp, Hudmid;
bool DgInput0, DgInput1, DgInput2, DgInput3;
ModbusMaster node_Sensor;
ModbusMaster node_Relay8;
ModbusMaster node_Ry4In4;
void preTransmission() {
  digitalWrite(RS485Control, RS485Transmit);
}
void postTransmission() {
 digitalWrite(RS485Control, RS485Receive);
}void setup() {
  pinMode(RS485Control, OUTPUT);
  pinMode(Pin_LEDMonitor, OUTPUT);
  Serial.begin(115200);
  Serial2.begin(9600);
  postTransmission();
  node_Sensor.begin(Slave_Sensor_ID, Serial2); // Modbus slave ID=1
  node_Sensor.preTransmission(preTransmission);
  node_Sensor.postTransmission(postTransmission);
  node_Relay8.begin(Slave_Relay8_ID, Serial2); // Modbus slave ID=3
  node_Relay8.preTransmission(preTransmission);
  node_Relay8.postTransmission(postTransmission);
  node_Ry4In4.begin(Slave_Ry4In4_ID, Serial2); // Modbus slave ID=5
  node_Ry4In4.preTransmission(preTransmission);
  node_Ry4In4.postTransmission(postTransmission);
}void ReadTemperature(void) {
  uint8_t result;
  // Toggle the coil at address (Manual Load Control)
  result = node_Sensor.writeSingleCoil(Slave_Sensor_ID, state);
  state = !state;
  // Read 2 registers starting at 0x0000)
  result = node_Sensor.readInputRegisters(0x0000, 2); // From=0, nByte=2
  if (result == node_Sensor.ku8MBSuccess) {
    CTempp = node_Sensor.getResponseBuffer(0x00) / 10.0f;
    Hudmid = node_Sensor.getResponseBuffer(0x01) / 10.0f;
  }
}void ReadDigitalInput(void) {
  uint8_t result;
  // Toggle the coil at address (Manual Load Control)
  result = node_Ry4In4.writeSingleCoil(Slave_Sensor_ID, state);
  state = !state;
  // Read 4 registers starting at 0x0000)
  result = node_Ry4In4.readDiscreteInputs(0, 4); // Start=0, nByte=4
  if (result == node_Ry4In4.ku8MBSuccess) {
    int DgTemp = node_Ry4In4.getResponseBuffer(0x00);
    DgInput3 = (DgTemp >> 3) & 1;
    DgInput2 = (DgTemp >> 2) & 1;
    DgInput1 = (DgTemp >> 1) & 1;
    DgInput0 = (DgTemp >> 0) & 1;
  }
}void RelayControl(int inputCase) {
  int rnMode = inputCase / 10;
  int nRelay = inputCase % 10;
  if (rnMode == 81) node_Relay8.writeSingleRegister(nRelay, 0x0100); // On RelayX
  if (rnMode == 80) node_Relay8.writeSingleRegister(nRelay, 0x0200); // Off RelayX
  if (rnMode == 41) node_Ry4In4.writeSingleCoil(nRelay, 0x01); // On RelayX
  if (rnMode == 40) node_Ry4In4.writeSingleCoil(nRelay, 0x00); // Off RelayX
}void loop() {
  ReadTemperature();
  ReadDigitalInput();
  Serial.print("\n Tempp('C): "); Serial.print(CTempp, 2);
  Serial.print(", Humid(%): "); Serial.print(Hudmid, 2);
  Serial.print(", Sensor[0:3]: "); Serial.print(DgInput3);
  Serial.print("-"); Serial.print(DgInput2);
  Serial.print("-"); Serial.print(DgInput1);
  Serial.print("-"); Serial.print(DgInput0);
  if (Serial.available() > 0) {
    int DataInput = Serial.parseInt();
    Serial.print("\n >> XYZ > X={8,4}Board Name, Y={1,0}On,Off, Z={0-8}RlyID >> ");
    Serial.println(DataInput);
    RelayControl(DataInput);
  }
  delay(2000);
}
```

< ผลการทดสอบ >



```
01:27:50.333 -> 03 06 00 01 01 00 D8 78
01:27:53.359 -> 03 06 00 02 01 00 28 78
01:27:56.345 -> 03 06 00 03 01 00 79 B8
01:27:59.372 -> 03 06 00 04 01 00 C8 79
01:28:02.390 -> 03 06 00 05 01 00 99 B9
01:28:05.410 -> 03 06 00 06 01 00 69 B9
01:28:08.443 -> 03 06 00 07 01 00 38 79
01:28:11.466 -> 03 06 00 08 01 00 08 7A
01:28:14.455 -> 03 06 00 01 02 00 D8 88
```

Quiz_204 – PLC Test

< รูปอุปกรณ์ที่ใช้ทดสอบ ขณะทำการทดสอบ >

< รูปอุปกรณ์ที่ใช้ทดสอบ ขณะทำการทดสอบ >

< โปรแกรมทดสอบ >

< ผลการทดสอบ >