แผนปฏิบัติการสอนสัปดาห์ที่ 10

วิชา 030143361 การโปรแกรมคอมพิวเตอร์สำหรับงานควบคุม **ระดับ:** ปริญญาตรี

เรื่อง เชื่อมต่อกับ Sensors และ Controller **เวลา:** บรรยาย 120 นาที
ปฏิบัติ 120 นาที

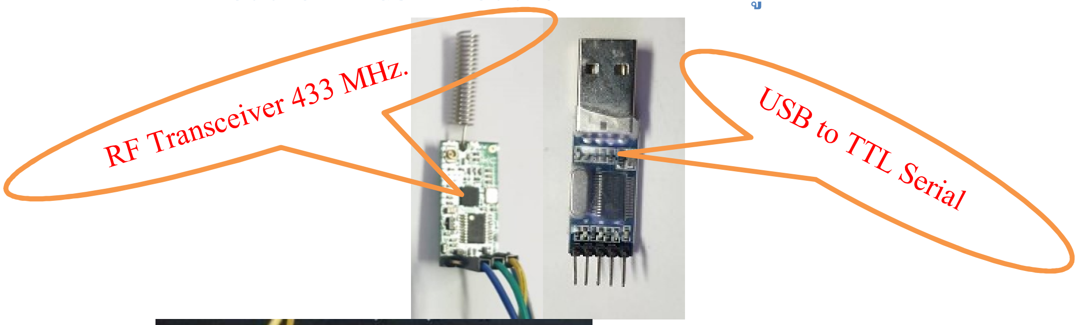
ก. วัตถุประสงค์การสอน	รายละเอียดตามที่ระบุไว้ใน
1. ใช้งาน RF Module และ GSM Module ในการสื่อสารข้อมูล 2. สื่อสารข้อมูล และควบคุม Microcontrollers 3. Data logger และ real-Time Control and Monitoring 4. เชื่อมต่อกับ PLC และ OPC 5. ตัวอย่างการใช้ Computer ควบคุม Hardware	หน้า 124-155

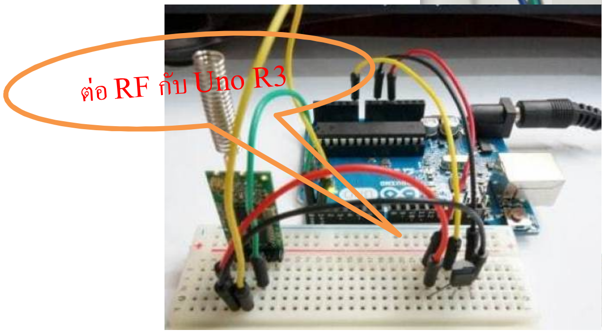
ข. การจัดการเรียนการสอน

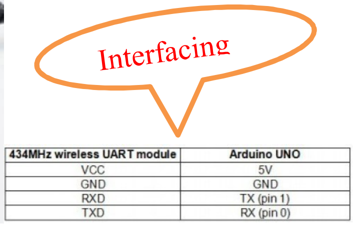
เวลา – นาที		0	60				120	180		240
วัตถุประสงค์		1, 2, 3, 4, 5								
การนำเข้าสู่บทเรียน		▨								
ให้เนื้อหา			▨							
สรุปเนื้อหา				▨	▨					
พัก						▨				
ทำแบบฝึกหัด							▨			
ให้เนื้อหา								▨		
สรุปเนื้อหา									▨	
ทดสอบและเก็บคะแนน										▨
ประเมินผล		พิจารณาจากผลการทดสอบในชั่วโมงสุดท้าย								
วิธีการสอน:	บรรยาย	▨	▨	▨						
	ถาม – ตอบ				▨					
	ทำแบบฝึกหัด					▨	▨			
	บรรยาย และแสดงให้ดู							▨	▨	
	ทดสอบ									▨
สื่อการสอน:	คอมพิวเตอร์	▨	▨	▨	▨	▨	▨	▨	▨	▨
	Presentations	▨	▨	▨	▨	▨	▨	▨	▨	▨

Content

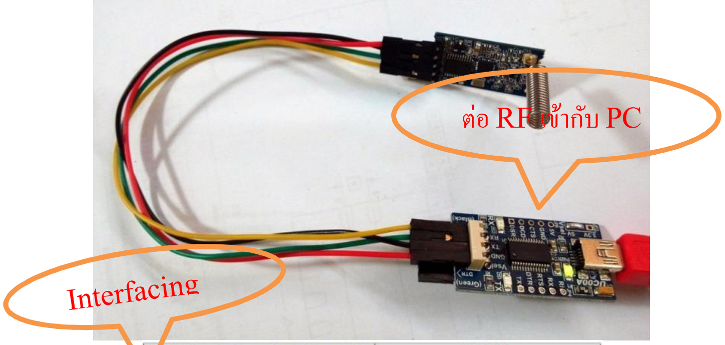
1. ใช้งาน RF Module และ GSM Module ในการสื่อสารข้อมูล

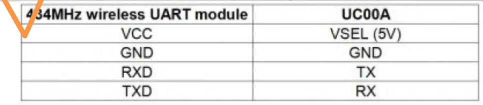





Interfacing

434MHz wireless UART module	Arduino UNO
VCC	5V
GND	GND
RXD	TX (pin 1)
TXD	RX (pin 0)





434MHz wireless UART module	UC00A
VCC	VSEL (5V)
GND	GND
RXD	TX
TXD	RX

CC1101 Module 433 Wireless to Serial

โมดูล RF ย่านความถี่ 433 Mhz สามารถตั้งค่าได้ทั้งโหมดรับ และโหมดส่งสัญญาณใช้พลังงานต่ำ ทำงานแบบ Serial Port Transparent

ข้อมูล CC1101 Module 433 Wireless to Serial

-Working frequency: 433MHz

-Operating supply voltage 3.3 V to 5.0 V

-Serial input and output via UART

-Serial Baud rate 1.2Kbps – 115.2Kbps (default 9600)

-Communication distance: 1m to 40m indoor

-Receiving sensitivity: -112dBm to -95dBm

-Transmit power: -30dBm to 10dBm

Introduction of the module:

1: FU1 mode:

The default delivery mode.

2: FU2 mode:

The IIDLE current is 80uA,but longer delay, transmission delay is close to 400ms.When use this mode,the serial port baud rate only can be 4800, 2400, 1200.Because of long time delay, when the continuous send serial data to the module, can only send 245 bytes of data by once.Low current consumption, suitable for the application of low power consumption.

3: FU3 mode:

Transmission delay has been reduced to less than 8ms, the corresponding power consumption also increased slightly.

4: FU4 mode:

Transmission distance is longer, the lower baud rate, the longer the transmission distance, the corresponding transmission delay is also a slight increase.

การเข้า Setup Mode

AT Instruct:

Enter into AT instruct mode:

1. Connect to 3.3V, give low level to the pin5 (keep low), then enter into AT instruct mode. It will exit AT instruct mode when give high level to pin 5.

2. Power off, pin 5 connect to GND, then power on, after 1 second, module will enter into AT instruct mode. It will exit AT instruct mode when give high level to pin 5.

Instructions as described below:

1. AT

Send this instruction, return OK.

2. AT+V

Return version information

3. AT + Bxxxx

Baud rate is xxxx, it can be 2400, 4800, 9600, 19200, 38400, 57600, 115200

4. AT+Cxxx

Set communication channels, form 001 to 127

5. AT+FUx

Switch serial pass through mode, from 1~4

https://www.arduinoall.com/product/741/cc1101-module-433-wireless-to-serial-hc-11 https://www.thaieasyelec.com/products/wireless-modules/rf-modules/cc1101-module-433-wireless-to-serial.html

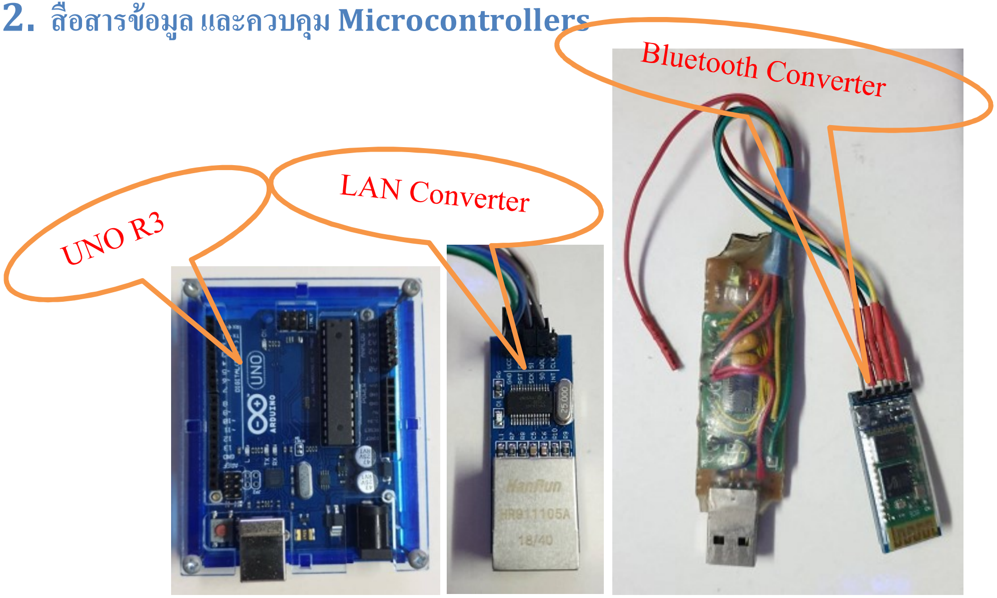
https://tutorial.cytron.io/2014/05/15/wireless-uart-arduino-433mhz-434mhz-module/

https://copperhilltech.com/content/GSM_GPRS_GNSS_HAT_User_Manual_EN.pdf

https://cdn-shop.adafruit.com/product-files/2637/SIM800+Series_TCPIP_Application+Note_V1.01.pdf

https://www.elecrow.com/wiki/images/2/20/SIM800_Series_AT_Command_Manual_V1.09.pdf

2. สื่อสารข้อมูล และควบคุม Microcontrollers



Example

Code

```
/*Arduino ENC28J60
  3V3   VCC
  GND   GND
  pin 13  SCK
  pin 12  SO
  pin 11  SI
  pin 10  CS

  LED 8
  SW 9
  Analog A0
*/

#include <UIPEthernet.h>

#define MY_ID "Min"
#define CMD_ON "on"
#define CMD_OFF "off"

#define LED_PIN 8
#define LED_ON digitalWrite(LED_PIN, HIGH)
#define LED_OFF digitalWrite(LED_PIN, LOW)

#define SW_PIN 9
#define IsSW_Push (digitalRead(SW_PIN)==0)

uint8_t mac[6] = {0x00, 0x01, 0x02, 0x03, 0x04, 0x25};
#define SERVER_NAME IPAddress(192,168,1,94)
#define SERVER_PORT 2000

#define SEND_TIME 1000  // mS.

signed long WaitCount;
EthernetClient client;
```

```
void SendToServer()
{
  if (((signed long)(millis() - WaitCount)) > 0)
  {
    WaitCount = millis() + SEND_TIME;
    if (client.connected())
    {
      int Temp = map(analogRead(A0), 0, 1023, 0, 100);
      client.print(MY_ID);
      client.print(",");
      if IsSW_Push
       client.print("1");
      else
       client.print("0");
      client.print(",");
      client.println(Temp);
      Serial.println("Client Sent!");
    }
    else
    {
      Serial.println("Client Disconnected Send Failed!");
    }
  }
}
int DoCmd(String Cmd)
{
  Serial.print("Cmd=");
  Serial.println(Cmd);
  if (Cmd == CMD_ON)
  {
    LED_ON;
    Serial.println(CMD_ON);
    client.print(CMD_ON);
    return (1);
  }
  else if (Cmd == CMD_OFF)
  {
    LED_OFF;
    Serial.println(CMD_OFF);
    client.print(CMD_OFF);
    return (1);
  }
  else
  {
    return (0);
  }
}

////********* Nwtwork Pulling ************
void ReadCmd()
{
  if (client.available())
  {
    String msg = client.readStringUntil('#');
    msg.trim();
    DoCmd(msg);
  }
}

void StartConnecting()
{
  client.stop();
  Serial.print("connecting to ");
  Serial.print(SERVER_NAME);
  Serial.println(":" + String(SERVER_PORT));
  if (client.connect(SERVER_NAME, SERVER_PORT))
    Serial.println("Connected successfully");
```

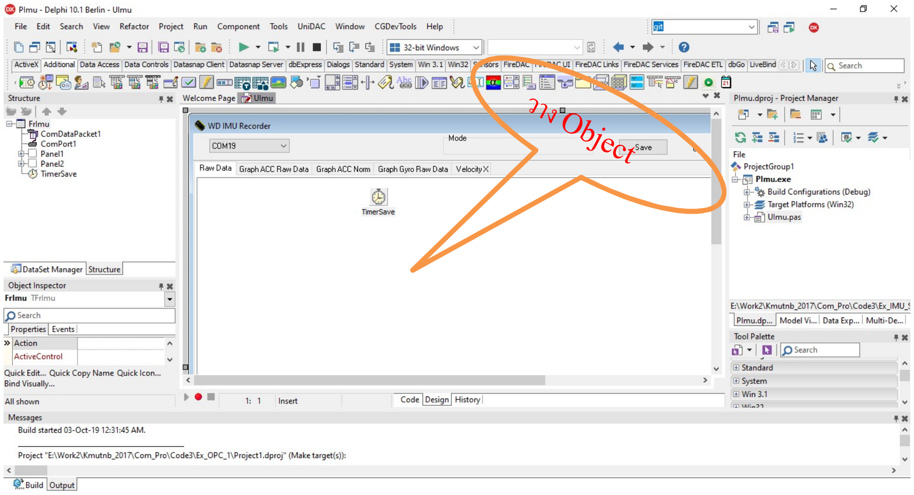
Code\week10\ Ex_LanControl_ino_1

```
  else
    Serial.println("Connection Failed!");
}

void DoNetwork()
{
  if (client.connected())
  {
    ReadCmd();
  }
  else
  {
    StartConnecting();
  }
}

void setup()
{
  pinMode(SW_PIN, INPUT_PULLUP);
  pinMode(LED_PIN, OUTPUT);
  LED_OFF;
  Ethernet.begin(mac);
  Serial.begin(9600);
  Serial.println("Lan Control!");
  WaitCount = 0;
}

void loop()
{
  DoNetwork();
  SendToServer();
}
```

3. Data logger และ real-Time Control and Monitoring

ตัวอย่างนี้แสดงการส่งข้อมูลดิบจาก Serial IMU ผ่าน USB to Serial TTL เข้าสู่คอมพิวเตอร์ แล้วเก็บข้อมูล แสดงผล และประมวลผลข้อมูล

Example



Messages

```
Build started 03-Oct-19 12:31:45 AM.

Project "E:\Work2\Kmutnb_2017\Com_Pro\Code3\Ex_OPC_1\Project1.dproj" (Make target(s)):
```

Code

```
unit Ulmu;


interface


uses

  Windows, Messages, SysUtils, Variants, Classes,

Graphics, Controls, Forms,

  Dialogs, ComCtrls, ExtCtrls, TeeProcs, TeEngine, Chart,

StdCtrls, CPort,

  CPortCtl, Series, Dateutils, GLWin32Viewer,

GLSpaceText, GLObjects,

  GLGeomObjects, GLScene, GLCoordinates, GLGraph,

GLCrossPlatform,

  BaseClasses, Math, jpeg;

const

 GRAPH_MAX_NUM = 200;

 LP_K = 0.3;

 HP_K = 0.01;
```

```
 SAMPLING_TIME = 0.01; // 10mS.

 MOVIVE_AVG_LEN = 10;   // 10mS. * 100 = 1000mS.


 ACC_NORM_VAL = 101;

 IMU_NUM = 1;


 BUFF_LEN = 6000;

 SAVE_FLODER = 'IMUData';


type

 TACC = record

  X:Integer;

  Y:Integer;

  Z:Integer;

  OldX:Integer;   // for Lp-Filter

  OldY:Integer;

  OldZ:Integer;

  OldMoveX:Integer;   // for Check Moving
  OldChangedX:Integer;
  OldChangedY:Integer;
  OldChangedZ:Integer;
  StationX:Integer;
  StationY:Integer;
  StationZ:Integer;
  VelocityX:Integer;
  OldVelocityX:Integer;
  PositionX:Integer;
  Norm:Single;
 end;

type
 TCalData = record
  OldData: Single;
  NewData: Single;
end;

type
 TImuAngle = record
  AngX: TCalData;
  AngY: TCalData;
  AngZ: TCalData;
end;

type
 TGyroForce = record
  FX: Single;
  FY: Single;
  FZ: Single;
  Fr:Single;
end;

type
 TGyro = record
  X:Integer;
```

```
  Y:Integer;
  Z:Integer;
  Norm:Single;
 end;

type
 TMovingAvgBuffer = record
  Data:array[1..MOVIVE_AVG_LEN] of Integer;
end;

type
 TACC_Data = record
  X:Single;
  Y:Single;
  Z:Single;
  Norm:Single;
 end;

type
 TGyro_Data = record
  X:Single;
  Y:Single;
  Z:Single;
 end;

Type
 TImuStatus = record
  DataReady:Boolean;
  BuffCount:Integer;
end;

Type
 TBuffData = record
  TimeStamp:TDateTime;
  Acc:TACC_Data;
  Gyro:TGyro_Data;
end;
```

```
type
 TImuBuffer = record
  ImuStatus:TImuStatus;
  BuffData:array[1..BUFF_LEN] of TBuffData;//
end;

 Type
 TSaveData = record
  AccX:String;
  AccY:String;
  AccZ:String;
  AccNorm:String;
  GyroX:String;
  GyroY:String;
  GyroZ:String;
end;

type
  TMyThread = class(TThread)
  private
    procedure SaveFile;
  protected
    procedure Execute; override;
  public
    constructor create;
    destructor Destroy; override;
  end;

type
  TFrImu = class(TForm)
    Panel1: TPanel;
    Panel2: TPanel;
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    TabSheet2: TTabSheet;
    TabSheet3: TTabSheet;
```

```
    ComPort1: TComPort;
    ComComboBox1: TComComboBox;
    ComDataPacket1: TComDataPacket;
    BtnOpen: TButton;
    BtnClose: TButton;
    Panel3: TPanel;
    Memo1: TMemo;
    Panel4: TPanel;
    Chart1: TChart;
    AccXRaw: TFastLineSeries;
    BtnMemClear: TButton;
    Chart2: TChart;
    AccYRaw: TFastLineSeries;
    Chart3: TChart;
    AccZRaw: TFastLineSeries;
    Button1: TButton;
    TabSheet4: TTabSheet;
    Panel5: TPanel;
    Chart4: TChart;
    AccNorm: TFastLineSeries;
    Panel6: TPanel;
    Button2: TButton;
    Chart5: TChart;
    GyroXRaw: TFastLineSeries;
    Chart6: TChart;
    GyroYRaw: TFastLineSeries;
    Chart7: TChart;
    GyroZRaw: TFastLineSeries;
    Label1: TLabel;
    GLScene1: TGLScene;
    GLXYZGrid1: TGLXYZGrid;
    Frame1: TGLCube;
    GLLightSource1: TGLLightSource;
    GLArrowLineX: TGLArrowLine;
    GLArrowLineY: TGLArrowLine;
    GLArrowLineZ: TGLArrowLine;
    GLSpaceTextX: TGLSpaceText;
    GLSpaceTextY: TGLSpaceText;
    GLSpaceTextZ: TGLSpaceText;
    Panel8: TPanel;
    Button3: TButton;
    LbStationX: TLabel;
    LbStationY: TLabel;
    LbStationZ: TLabel;
    Label2: TLabel;
    TabSheet7: TTabSheet;
    Chart9: TChart;
    GPVelocityX: TFastLineSeries;
    LbVelocityX: TLabel;
    ImgLeft: TImage;
    ImgRight: TImage;
    Rg1: TRadioGroup;
    TimerSave: TTimer;
    Button4: TButton;
    LbCount: TLabel;
    procedure BtnOpenClick(Sender: TObject);
    procedure BtnCloseClick(Sender: TObject);
    procedure ComDataPacket1Packet(Sender: TObject;
const Str: String);
    procedure BtnMemClearClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure TimerSaveTimer(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Rg1Click(Sender: TObject);
  private
   MyThread: TMyThread;
   SaveData:TSaveData;
   SaveFile:TextFile;
   SavePath: String;  // main path
   SaveSubPath: String;  // Sub path Seperate by date
   LastDay:TDateTime;
   Acc_Data:array[1..IMU_NUM] of TACC_Data;
   Gyro_Data:array[1..IMU_NUM] of TGyro_Data;
```

```
   ImuBuffer:array[1..IMU_NUM] of TImuBuffer;//
   ACC:TACC;
   Gyro:TGyro;
   FirstTrig:Boolean;
   FirstTrigTime:Cardinal;
   MovingAvgBuffX:TMovingAvgBuffer;
   GyroAng:TImuAngle;
   GyroForce:TGyroForce;
   AccNoMoveCount:Integer;
   procedure Init();
   procedure ChkRepFolder();
   procedure CreateThread_SvaeFlie;
   procedure DestroyThread_SvaeFlie;
   procedure AssignSaveFile(IMUNo, Index:Integer);
   procedure SaveDataFile(IMUNo:Integer);
   function AssignRawData(DataIn:String):Boolean;
   procedure PlotAccGraph();
   procedure PlotNormGraph();
   procedure PlotGyroGraph();
   procedure PlotVelocityXGraph();
   procedure PlotLineGraph(Graph: TFastLineSeries;
Data:Real);
   procedure CaptureAccStation();
   function LpFilter(k, OldVal, InputVal:Single):Single;
   function HpFilter(k, OldVal, InputVal:Single):Single;
   function CalMovingAvg(DataIn:Integer; var
DataBuff:TMovingAvgBuffer):Integer;
   function CallGyroForce(NewData:TGyro):TGyroForce;
   function ChkNoMove(OldVal, NewVal:Integer):Boolean;
   function ChkNoMove2(AccNorm:Single):Boolean;
   procedure PlotAllGraph();
   procedure SaveIMUFile();
   function FloatToStr2(Data:Single):String;
    { Private declarations }
  public
    { Public declarations }
  end;
```

```
var
  FrImu: TFrImu;

implementation

{$R *.dfm}

procedure TFrImu.BtnOpenClick(Sender: TObject);
begin
 if ComPort1.Connected then
  Comport1.Close;
 try
  ComDataPacket1.StopString:=#13#10;
  Comport1.Open;
  BtnOpen.Enabled:=False;
  BtnClose.Enabled:=True;
  FirstTrig:=True;
  Memo1.Clear;
  AccXRaw.Clear;
  AccYRaw.Clear;
  AccZRaw.Clear;
  AccNorm.Clear;
 except
  ComPort1.Close;
  ShowMessage('Comport Error!');
  end;
end;

procedure TFrImu.BtnCloseClick(Sender: TObject);
begin
 ComPort1.Close;
 BtnOpen.Enabled:=True;
 BtnClose.Enabled:=False;
end;
```

```
procedure TFrImu.ComDataPacket1Packet(Sender:
TObject; const Str: String);
begin
 if AssignRawData(Str) then
  begin
   if Rg1.ItemIndex = 0 then
    SaveImuFile
   else if Rg1.ItemIndex = 1 then
    PlotAllGraph;
  end
 else
  begin
   Memo1.Lines.Add('Error!');
  end;
end;

procedure TFrImu.BtnMemClearClick(Sender: TObject);
begin
  Memo1.Clear;
  AccXRaw.Clear;
  AccYRaw.Clear;
  AccZRaw.Clear;
  AccNorm.Clear;
  GyroXRaw.Clear;
  GyroYRaw.Clear;
  GyroZRaw.Clear;
end;

procedure TFrImu.FormCreate(Sender: TObject);
begin
 Memo1.Clear;
 ACC.X:=0;
 ACC.StationX:=0;
 ACC.VelocityX:=0;
 ACC.PositionX:=0;
 ACC.StationY:=0;
 ACC.StationZ:=100;
 Init();
 ChkRepFolder;
end;

function TFrImu.AssignRawData(DataIn: String):Boolean;
var
  List: TStrings;
  NumberErrorPos:Integer;
  i, IntVal:Integer;
  ConvertedOK:Boolean;
begin
  List:=TStringList.Create;
  try
    ExtractStrings([','], [], PChar(DataIn), List);
    if List.Count = 6 then
     begin
      ConvertedOK:=True;
      for i := 0 to 5 do
       begin
        val(List[i], IntVal, NumberErrorPos);
        if NumberErrorPos <> 0 then       // Get Error
         begin
          ConvertedOK:=False;
          break;
         end;
       end;

      if ConvertedOK then
       begin
        ACC.X:=StrToInt(List[0]);
        ACC.Y:=StrToInt(List[1]);
        ACC.Z:=StrToInt(List[2]);
Acc.Norm:=Sqrt((ACC.X*ACC.X)+(ACC.Y*ACC.Y)+(ACC.Z*AC
C.Z));
        Gyro.X:=StrToInt(List[3]);
        Gyro.Y:=StrToInt(List[4]);
        Gyro.Z:=StrToInt(List[5]);
```

```

Gyro.Norm:=Sqrt((Gyro.X*Gyro.X)+(Gyro.Y*Gyro.Y)+(Gyro.Z
*Gyro.Z));
        Result:=True;
       end
      else
       Result:=False;
     end
    else
     Result:=False;
  finally
    List.Free;
  end;
end;

procedure TFrImu.PlotAccGraph;
begin
 PlotLineGraph(AccXRaw, ACC.X);
 PlotLineGraph(AccYRaw, ACC.Y);
 PlotLineGraph(AccZRaw, ACC.Z);
end;

procedure TFrImu.PlotNormGraph;
begin
 PlotLineGraph(AccNorm, ACC.Norm-ACC_NORM_VAL);
end;

procedure TFrImu.PlotLineGraph(Graph: TFastLineSeries;
Data: Real);
var
 tmpX:Double;
begin
 with Graph do
  begin
    if XValues.Count<GRAPH_MAX_NUM then
     begin
      Add(Data);
     end
    else
     begin
      tmpX:=XValues[1]-XValues[0];
      Delete(0);
      AddXY( XValues.Last+tmpX, Data,'',clTeeColor);
     end;
  end;
end;

procedure TFrImu.PlotGyroGraph;
begin
 PlotLineGraph(GyroXRaw, Gyro.X);
 PlotLineGraph(GyroYRaw, Gyro.Y);
 PlotLineGraph(GyroZRaw, Gyro.Z);
end;

procedure TFrImu.Button3Click(Sender: TObject);
begin
 ACC.VelocityX:=0;
 ACC.PositionX:=0;
end;

procedure TFrImu.CaptureAccStation;
begin
 ACC.StationX:=ACC.X;
 ACC.StationY:=ACC.Y;
 ACC.StationZ:=ACC.Z;
 LbStationX.Caption:=IntToStr(ACC.StationX);
 LbStationY.Caption:=IntToStr(ACC.StationY);
 LbStationZ.Caption:=IntToStr(ACC.StationZ);
end;

function TFrImu.HpFilter(k, OldVal, InputVal: Single):
Single;
var
 Ans:Single;
```

```
begin
 Ans:=(k*OldVal)+((1-k)*InputVal);
 Result:=Ans;
end;

function TFrImu.LpFilter(k, OldVal, InputVal: Single):
Single;
var
 Ans:Single;
begin
 Ans:=(k*InputVal)+((1-k)*OldVal);
 Result:=Ans;
end;

function TFrImu.CalMovingAvg(DataIn:Integer; var
DataBuff:TMovingAvgBuffer): Integer;
var
 i, SumIn:Integer;
begin
 SumIn:=0;
 for i := 1 to MOVIVE_AVG_LEN do
  begin
   if i<MOVIVE_AVG_LEN then
    DataBuff.Data[i]:=DataBuff.Data[i+1]    // Shift Data
   else
    DataBuff.Data[i]:=DataIn; // Last Data  = Data input
   SumIn:=SumIn+DataBuff.Data[i];
  end;
 Result:=Round(SumIn/MOVIVE_AVG_LEN);
end;

function TFrImu.CallGyroForce(NewData: TGyro):
TGyroForce;
var
 NewAngle:TImuAngle;
 NewForce:TGyroForce;
begin
```

```
 NewAngle.AngX.NewData:=GyroAng.AngX.OldData+
   (NewData.X*SAMPLING_TIME );
 NewAngle.AngY.NewData:=GyroAng.AngY.OldData+
   (NewData.Y*SAMPLING_TIME);
 NewAngle.AngZ.NewData:=GyroAng.AngZ.OldData+
   (NewData.Z*SAMPLING_TIME );
NewAngle.AngX.NewData:=DegToRad(NewAngle.AngX.Ne
wData);
NewAngle.AngY.NewData:=DegToRad(NewAngle.AngY.Ne
wData);
NewAngle.AngZ.NewData:=DegToRad(NewAngle.AngZ.Ne
wData);
 NewForce.FX:=1/
(Sqrt(1+(Cot(NewAngle.AngX.NewData)*Cot(NewAngle.An
gX.NewData))*
(Sec(NewAngle.AngY.NewData)*Sec(NewAngle.AngY.NewD
ata))));
if NewAngle.AngX.NewData<0 then
NewForce.FX:=(-1)*NewForce.FX;
 NewForce.FY:=1/
Sqrt(1+(Cot(NewAngle.AngY.NewData)*Cot(NewAngle.Ang
Y.NewData))*
((Sec(NewAngle.AngX.NewData)*Sec(NewAngle.AngX.New
Data))));
if NewAngle.AngY.NewData<0 then
NewForce.FY:=(-1)*NewForce.FY;
 NewForce.FZ:=Sqrt(1-(NewForce.FX*NewForce.FX)-
(NewForce.FY*NewForce.FY));
 Result:=NewForce;
end;

function TFrImu.ChkNoMove2(AccNorm: Single): Boolean;
var
 AbsNorm:Single;
begin
 AbsNorm:=abs(AccNorm-ACC_NORM_VAL);
```

```
 if AbsNorm<3 then      // Stationary State threshold
  begin
   AccNoMoveCount:=AccNoMoveCount+1;    // Waiting
for long time
   if 20<AccNoMoveCount then      // wait for
20*Sampling time
    begin
     AccNoMoveCount:=0;
     Result:=True;
    end
   else
     Result:=False;
  end
 else
  begin
   AccNoMoveCount:=0;
   Result:=False;
  end;
end;

procedure TFrImu.PlotVelocityXGraph;
var
 SpeedThreshold:Integer;
begin
 SpeedThreshold:=10;
 PlotLineGraph(GPVelocityX, ACC.VelocityX/2);
 if ACC.VelocityX<-SpeedThreshold then
  begin
   ImgLeft.Visible:=True;
   ImgRight.Visible:=False;
  end
 else if SpeedThreshold<ACC.VelocityX then
  begin
   ImgLeft.Visible:=False;
   ImgRight.Visible:=True;
  end
  else
  begin
   ImgLeft.Visible:=False;
   ImgRight.Visible:=False;
  end;
  LbVelocityX.Caption:=IntToStr(ACC.VelocityX);
end;

function TFrImu.ChkNoMove(OldVal, NewVal:Integer):
Boolean;
var
 AbsDiff:Integer;
begin
 AbsDiff:=abs(NewVal-OldVal);
 if AbsDiff<=1 then      // Stationary State threshold
  begin
   AccNoMoveCount:=AccNoMoveCount+1;    // Waiting
for long time
   if 20<AccNoMoveCount then      // wait for
20*Sampling time
    begin
     AccNoMoveCount:=0;
     Result:=True;
    end
   else
     Result:=False;
  end
 else
  begin
   AccNoMoveCount:=0;
   Result:=False;
  end;
end;

procedure TFrImu.PlotAllGraph;
var
 AccChanged:Integer;
begin
```

```
   if FirstTrig then
    begin
     FirstTrigTime:=MillisecondOfTheDay(Now);
     FirstTrig:=False;
    end;
 Memo1.Lines.Add(//IntToStr(MillisecondOfTheDay(Now)-
FirstTrigTime)+','+
                  IntToStr(ACC.X)+','+
                  IntToStr(ACC.Y)+','+
                  IntToStr(ACC.Z)+','+
                  IntToStr(Gyro.X)+','+
                  IntToStr(Gyro.Y)+','+
                  IntToStr(Gyro.Z));
   ACC.StationX:=CalMovingAvg(ACC.X, MovingAvgBuffX);
 AccChanged:=ACC.X-ACC.StationX;
 ACC.VelocityX:=ACC.VelocityX + ACC.OldChangedX +
round((AccChanged - ACC.OldChangedX)/2);
 ACC.OldChangedX:=AccChanged;
 ACC.PositionX:=ACC.PositionX + ACC.OldVelocityX+
round((ACC.VelocityX - ACC.OldVelocityX)/2);
if ChkNoMove(ACC.OldMoveX, ACC.X) then
  begin
   ACC.OldChangedX:=0;
   ACC.VelocityX:=0;
  end
 else
  ACC.OldChangedX:=AccChanged;
ACC.OldMoveX:=ACC.X;
 GyroForce:=CallGyroForce(Gyro);  // Gyro Force Per Unit
GyroForce.FY:=GyroForce.FY*6000;
Label1.Caption:=Format('%0.2f', [ACC.Norm]);
Label2.Caption:=IntToStr(AccChanged);
  PlotAccGraph;
  PlotNormGraph;
  PlotGyroGraph;
  PlotVelocityXGraph;
end;
```

```
procedure TFrImu.SaveImuFile;
begin
  with ImuBuffer[1] do
    begin
     ImuStatus.DataReady:=True;
     if ImuStatus.BuffCount < BUFF_LEN then
      begin
       ImuStatus.BuffCount:=ImuStatus.BuffCount+1;
       LbCount.Caption:=IntToStr(ImuStatus.BuffCount);
       Acc_Data[1].X:=ACC.X;
       Acc_Data[1].Y:=ACC.Y;
       Acc_Data[1].Z:=ACC.Z;
       Acc_Data[1].Norm:=ACC.Norm;
       Gyro_Data[1].X:=Gyro.X;
       Gyro_Data[1].Y:=Gyro.Y;
       Gyro_Data[1].Z:=Gyro.Z;

ImuBuffer[1].BuffData[ImuStatus.BuffCount].TimeStamp:=
Now;
ImuBuffer[1].BuffData[ImuStatus.BuffCount].Acc:=Acc_Da
ta[1];
ImuBuffer[1].BuffData[ImuStatus.BuffCount].Gyro:=Gyro_
Data[1];
     end
    else if Rg1.ItemIndex = 0 then
     begin
      DestroyThread_SvaeFlie;
      CreateThread_SvaeFlie;
     end;

  end;
end;

procedure TFrImu.CreateThread_SvaeFlie;
begin
  if MyThread = nil then
```

```
  begin
    MyThread := TMyThread.create();
  end;
end;

procedure TFrImu.DestroyThread_SvaeFlie;
begin
  if MyThread <> nil then
  begin
    MyThread.Terminate;
    MyThread.WaitFor;
    FreeAndNil(MyThread);
  end;
end;

{ TMyThread }

constructor TMyThread.create;
begin
  inherited create(False);
  FreeOnTerminate := False; // Auto Free
end;
destructor TMyThread.Destroy;
begin
  inherited;
end;

procedure TMyThread.Execute;
begin
  try
    Application.ProcessMessages; // to get COM/DCOM
Callbacks
    Synchronize(SaveFile);
  except
    ShowMessage('Thread Error!');
  end;
end;
```

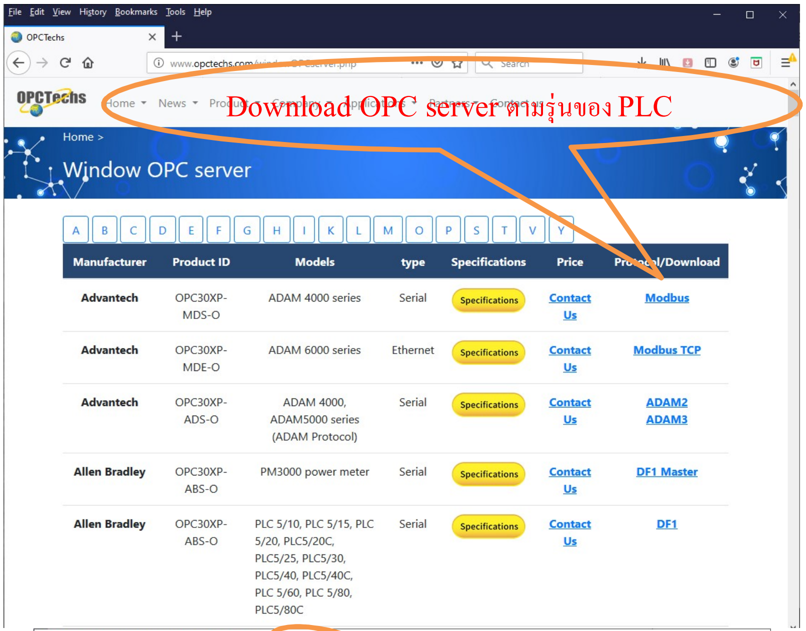
```
procedure TFrImu.TimerSaveTimer(Sender: TObject);
begin
 {if Rg1.ItemIndex = 0 then      //
  begin
   DestroyThread_SvaeFlie;
   CreateThread_SvaeFlie;
  end; }
end;

procedure TFrImu.AssignSaveFile(IMUNo, Index: Integer);
begin
 with ImuBuffer[IMUNo] do
  begin
   SaveData.AccX:=FloatToStr2(BuffData[Index].Acc.X);
   SaveData.AccY:=FloatToStr2(BuffData[Index].Acc.Y);
   SaveData.AccZ:=FloatToStr2(BuffData[Index].Acc.Z);
SaveData.AccNorm:=FloatToStr2(BuffData[Index].Acc.Nor
m);
   SaveData.GyroX:=FloatToStr2(BuffData[Index].Gyro.X);
   SaveData.GyroY:=FloatToStr2(BuffData[Index].Gyro.Y);
   SaveData.GyroZ:=FloatToStr2(BuffData[Index].Gyro.Z);
  end;
end;

procedure TFrImu.SaveDataFile(IMUNo: Integer);
var
 Buff:String;
 i:Integer;
begin
 if ImuBuffer[IMUNo].ImuStatus.BuffCount = 0 then
  exit;
 AssignFile(SaveFile, SaveSubPath+
'\Imu'+IntToStr(IMUNo)+'_'+
             FormatDateTime('hh_nn_ss', Now)+'.txt');
{Assigns the Filename}
 ReWrite(SaveFile); {Create a new file }
```

```
 Buff:='Name = IMU'+IntToStr(IMUNo);
 Writeln(SaveFile, Buff);
 Buff:='Date Time = '+FormatDateTime('dd-mm-yyyy
hh:nn:ss.zz', Now);
 Writeln(SaveFile, Buff);
 Buff:='Total =
'+IntToStr(ImuBuffer[IMUNo].ImuStatus.BuffCount)+#13#1
0;
 Writeln(SaveFile, Buff);

 Buff:='(No, Time, AccX, AccY, AccZ, AccNorm, GyroY,
GyroZ)'+#13#10;
 Writeln(SaveFile, Buff);
 for i := 1 to ImuBuffer[IMUNo].ImuStatus.BuffCount do
  begin
   AssignSaveFile(IMUNo, i);
   with ImuBuffer[IMUNo] do
    begin
     Buff:=IntToStr(i)+','+FormatDateTime('hh:nn:ss.zzz',
ImuBuffer[IMUNo].BuffData[i].TimeStamp)+
','+SaveData.AccX+','+SaveData.AccY+','+SaveData.AccZ
+','+SaveData.AccNorm+
','+SaveData.GyroX+','+SaveData.GyroY+','+SaveData.Gyr
oZ;
     Writeln(SaveFile, Buff);
    end;
  end;
 Closefile(SaveFile); {Closes file }
 ImuBuffer[IMUNo].ImuStatus.BuffCount:=0;
end;

procedure TMyThread.SaveFile;
begin
 FrImu.SaveDataFile(1);
end;

procedure TFrImu.ChkRepFolder;
```

```
var
 SaveFolder:String;
begin
  SavePath := ExtractFilePath(Application.ExeName) +
SAVE_FLODER;
  if not DirectoryExists(SavePath) then
  begin
    ForceDirectories(SavePath);
    ShowMessage('Create Directory => ' + SavePath);
  end;
  SaveFolder:='Imu_'+FormatDateTime('dd-mm-yyyy',
Now);
  SaveSubPath:=SavePath+'\'+SaveFolder;
  if not DirectoryExists(SavePath+'\'+SaveFolder) then
  begin
    ForceDirectories(SaveSubPath);
    ShowMessage('Create Subdirectory => ' +
SaveSubPath);
  end;
  LastDay:=Now;
end;

procedure TFrImu.Init;
var
 i:Integer;
begin
 for i := 1 to IMU_NUM do
  begin
   ImuBuffer[i].ImuStatus.DataReady:=False;
   ImuBuffer[i].ImuStatus.BuffCount:=0;
  end;
 MyThread := nil;
end;

procedure TFrImu.Button4Click(Sender: TObject);
begin
 DestroyThread_SvaeFlie;
 CreateThread_SvaeFlie;
end;

function TFrImu.FloatToStr2(Data: Single): String;
begin
 Result:= Format('%0.2f', [Data]);
end;

procedure TFrImu.Rg1Click(Sender: TObject);
begin
 if Rg1.ItemIndex = 0 then
  TimerSave.Enabled:=True
 else if Rg1.ItemIndex = 1 then
  TimerSave.Enabled:=False;
end;


end.
```

Code\week10\ Ex_IMU_STM32_1

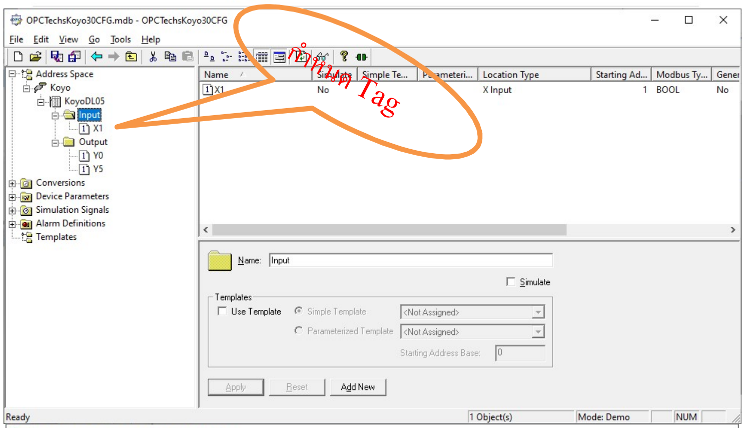
4. เชื่อมต่อกับ PLC และ OPC


File Edit View History Bookmarks Tools Help
OPCTechs
www.opctechs.com/window-OPCserver.php
Search
OPCTechs
Home News Product Company Applications Partners Contact us
Home >
A B C D E F G H I K L M O P S T V Y

Download OPC server ตามรุ่นของ PLC

Window OPC server

Manufacturer	Product ID	Models	type	Specifications	Price	Protocol/Download
Advantech	OPC30XP-MDS-O	ADAM 4000 series	Serial	Specifications	Contact Us	Modbus
Advantech	OPC30XP-MDE-O	ADAM 6000 series	Ethernet	Specifications	Contact Us	Modbus TCP
Advantech	OPC30XP-ADS-O	ADAM 4000, ADAM5000 series (ADAM Protocol)	Serial	Specifications	Contact Us	ADAM2 ADAM3
Allen Bradley	OPC30XP-ABS-O	PM3000 power meter	Serial	Specifications	Contact Us	DF1 Master
Allen Bradley	OPC30XP-ABS-O	PLC 5/10, PLC 5/15, PLC 5/20, PLC5/20C, PLC5/25, PLC5/30, PLC5/40, PLC5/40C, PLC 5/60, PLC 5/80, PLC5/80C	Serial	Specifications	Contact Us	DF1



กำหนด Tag

Name	Simulate	Simple Te...	Parameteri...	Location Type	Starting Ad...	Modbus Ty...	Gener
X1	No			X Input	1	BOOL	No

Name: Input

□ Simulate

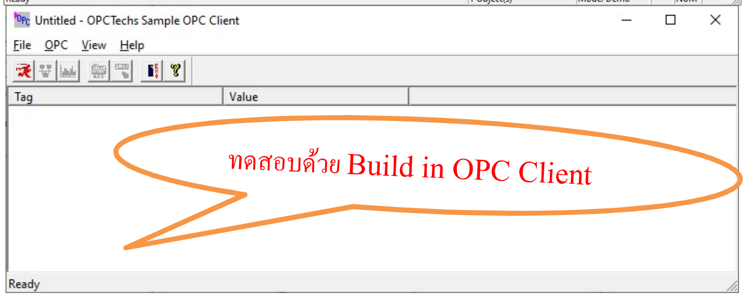
Templates

□ Use Template ● Simple Template <Not Assigned>

○ Parameterized Template <Not Assigned>

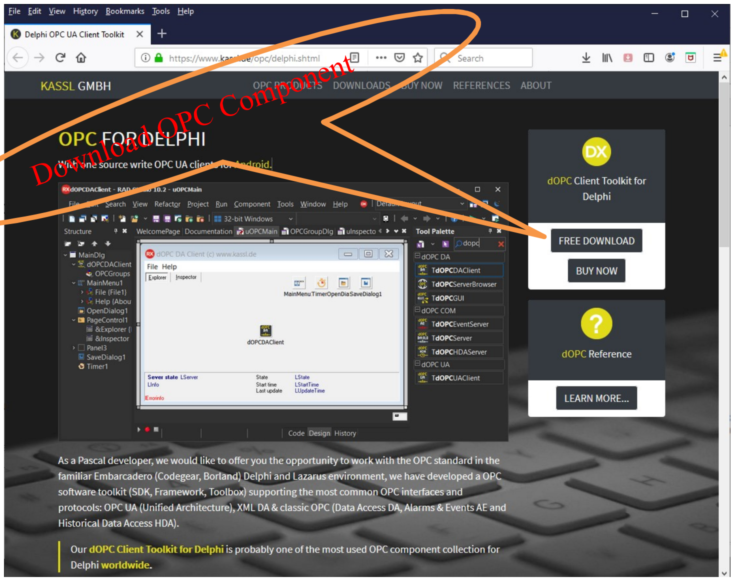
Starting Address Base: 0

Ready | 1 Object(s) | Mode: Demo | NUM



ทดสอบด้วย Build in OPC Client

http://www.opctechs.com/



OPC FOR DELPHI

With one source write OPC UA client for Android.

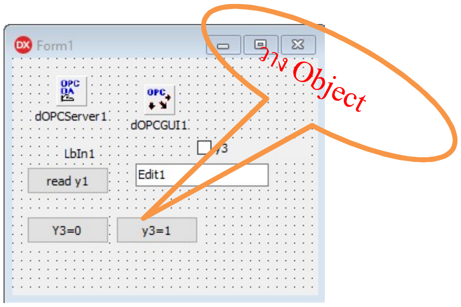
As a Pascal developer, we would like to offer you the opportunity to work with the OPC standard in the familiar Embarcadero (Codegear, Borland) Delphi and Lazarus environment, we have developed a OPC software toolkit (SDK, Framework, Toolbox) supporting the most common OPC interfaces and protocols: OPC UA (Unified Architecture), XML DA & classic OPC (Data Access DA, Alarms & Events AE and Historical Data Access HDA).

> Our dOPC Client Toolkit for Delphi is probably one of the most used OPC component collection for Delphi worldwide.

https://www.kassl.de/opc/delphi.shtml

ทดสอบเขียนโปรแกรมเชื่อมต่อ PLC ผ่าน OPC เพื่ออ่านค่า Input และควบคุม Output

Example



Code

```
unit Unit1;


interface


uses

  Winapi.Windows, Winapi.Messages, System.SysUtils,

System.Variants, System.Classes, Vcl.Graphics,

  Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls,

dOPCIntf,

  dOPCComn, dOPCDA, dOPC, dOPCGUI;
```

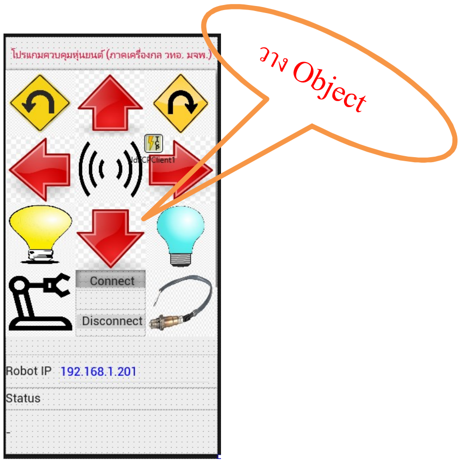
```
type

  TForm1 = class(TForm)

    dOPCServer1: TdOPCServer;

    Edit1: TEdit;

    Button2: TButton;

    LbIn1: TLabel;

    Button3: TButton;

    Button4: TButton;

    dOPCGUI1: TdOPCGUI;

    CheckBox1: TCheckBox;
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
  private
  function FindOPCItem(OpcServer: TdOPCServer; Item:
string)
      : TdOpcItem; overload;
  function FindOPCItem(OpcGroup: tdOPCGroup; Item:
string)
      : TdOpcItem; overload;
  function ReadFromPLC(Item: String): String;

  procedure WriteToPLC(Item: string; Value: OleVariant);
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

function TForm1.FindOPCItem(OpcGroup: tdOPCGroup;
Item: string): TdOpcItem;
var
  i: Integer;
  dopcItem: TdOpcItem;
begin
  Item := UpperCase(Item);
  Result := nil;
  for i := 0 to OpcGroup.OPCItems.Count - 1 do
  begin
    dopcItem := OpcGroup.OPCItems[i];
    if Item = UpperCase(dopcItem.ItemID) then
    begin
      Result := dopcItem;
      Exit;
    end;
  end;
end;

procedure TForm1.Button2Click(Sender: TObject);

var
 Status: String;
begin
   Status := ReadFromPLC('COM11.DL05.Output.Y1');
   Lbln1.Caption:=Status;
end;

procedure TForm1.Button3Click(Sender: TObject);
begin
 WriteToPLC('COM11.DL05.Output.Y3', '0');
end;

procedure TForm1.Button4Click(Sender: TObject);
begin
 WriteToPLC('COM11.DL05.Output.Y3', '1');
end;

function TForm1.FindOPCItem(OpcServer: TdOPCServer;
Item: string): TdOpcItem;
var
  i: Integer;
begin
  Result := nil;
  for i := 0 to OpcServer.OPCGroups.Count - 1 do
  begin
    Result := FindOPCItem(OpcServer.OPCGroups[i], Item);
    if Result <> nil then
      Exit;
  end;
end;

function TForm1.ReadFromPLC(Item: String): String;
var
  opcItem: TdOpcItem;
begin
  if dOPCServer1.Active then
  begin
    opcItem := FindOPCItem(dOPCServer1, Item);
    if opcItem = nil then
    begin
      raise Exception.create('Read From item =>' + Item +
' not found!');
      Result := '';
    end
    else
    begin
      Result := opcItem.ValueStr;
    end;
  end
  else
    Result := '';
end;

procedure TForm1.WriteToPLC(Item: string; Value:
OleVariant);
var
  opcItem: TdOpcItem;
begin
  opcItem := FindOPCItem(dOPCServer1, Item); // search
for item in OPC Groups
  if opcItem = nil then
    raise Exception.create('Write to item =>' + Item + '
not found!');
    opcItem.WriteSync(Value)
end;

end.
```

Code\week10\ Ex_OPC_1

5. ตัวอย่างการใช้ Computer ควบคุม Hardware

ตัวอย่างนี้เป็นการสร้าง Multi-Device-Application เพื่อส่งคำสั่งพร้อมพารามิเตอร์ไปควบคุมการเคลื่อนที่ของหุ่นยนต์ และอ่านข้อมูลจาก Analog Sensor กลับมาแสดงผล ผ่าน Wi-Fi

Example

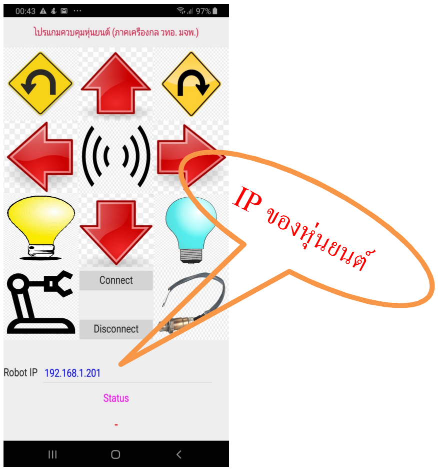


โปรแกมควบคุมหุ่นยนต์ (ภาคเครื่องกล วทอ. มจพ.)

Robot IP 192.168.1.201

Status

Result



ไปรแกมควบคุมหุ่นยนต์ (ภาคเครืองกล วทอ. มจพ.)

Robot IP 192.168.1.201

Status

-

IP ของหุ่นยนต์

Code

```
unit URobotWifi1;


interface


uses
  System.SysUtils, System.Types, System.UITypes,
System.Classes, System.Variants,
  FMX.Types, FMX.Controls, FMX.Forms, FMX.Graphics,
FMX.Dialogs, FMX.Layouts,
  System.ImageList, FMX.ImgList,
FMX.Controls.Presentation, FMX.StdCtrls,
  FMX.Effects, FMX.Objects, IdBaseComponent,
IdComponent, IdTCPConnection,
  IdTCPClient, FMX.Edit;
const
 CMD_TL = 'TL';
 CMD_TR = 'TR';
 CMD_GO = 'GO';
 CMD_BK = 'BK';
 CMD_RL = 'RL';
 CMD_RR  = 'RR';
 CMD_ST  = 'ST';
 CMD_BUZ_ON  = 'BO';
 CMD_BUZ_OFF  = 'BF';
```

```
 CMD_LED1_ON  = '1O';
 CMD_LED1_OFF  = '1F';
 CMD_LED2_ON  = '2O';
 CMD_LED2_OFF  = '2F';
 CMD_SER_ON  = 'SO';
 CMD_SER_OFF  = 'SF';
 CMD_GET_A0  = 'GA';


type
  TForm1 = class(TForm)
    ScaledLayout1: TScaledLayout;
    GridLayout1: TGridLayout;
    Layout3: TLayout;
    ShadowEffect1: TShadowEffect;
    Label1: TLabel;
    Image1: TImage;
    Image2: TImage;
    Image3: TImage;
    Image4: TImage;
    Image5: TImage;
    Image6: TImage;
    Image7: TImage;
    Image8: TImage;
    Image9: TImage;
    Image10: TImage;
    IdTCPClient1: TIdTCPClient;
    GridLayout2: TGridLayout;
    Image12: TImage;
    Button1: TButton;
    Layout1: TLayout;
    Button2: TButton;
    Layout2: TLayout;
    Layout4: TLayout;
    LbStatus: TLabel;
    Layout5: TLayout;
    LbAck: TLabel;
    Layout6: TLayout;
    Label2: TLabel;
    EdIp: TEdit;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Image2MouseDown(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure Image2MouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure IdTCPClient1Status(ASender: TObject;
const AStatus: TIdStatus;
      const AStatusText: string);
    procedure IdTCPClient1Disconnected(Sender:
TObject);
    procedure IdTCPClient1Connected(Sender: TObject);
    procedure Image8MouseDown(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure Image4MouseDown(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure Image6MouseDown(Sender: TObject;
Button: TMouseButton;
```

```
      Shift: TShiftState; X, Y: Single);
    procedure Image1MouseDown(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure Image3MouseDown(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure Image5MouseDown(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure Image5MouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure Image7MouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure Image7MouseDown(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure Image9MouseDown(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure Image9MouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure Image10MouseDown(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y: Single);
    procedure Image12MouseDown(Sender: TObject;
Button: TMouseButton;
      Shift: TShiftState; X, Y: Single);
  private
   ServoToggle:Boolean;
   procedure SendCmd(Cmd:String);
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.fmx}
{$R *.LgXhdpiPh.fmx ANDROID}

procedure TForm1.Button1Click(Sender: TObject);
begin
 try
   IdTCPClient1.Host:=EdIp.Text;
   IdTCPClient1.Connect;
 except
  LbStatus.Text:='Connect error!';
 end;
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
 IdTCPClient1.Disconnect;
end;

procedure TForm1.IdTCPClient1Connected(Sender:
TObject);
begin
 LbStatus.Text:='Connected!';
end;

procedure TForm1.IdTCPClient1Disconnected(Sender:
TObject);
begin
 LbStatus.Text:='Disconnected!';
end;
```

```
procedure TForm1.IdTCPClient1Status(ASender: TObject;
const AStatus: TIdStatus;
  const AStatusText: string);
begin
LbStatus.Text:='Status = '+AStatusText;
end;

procedure TForm1.Image10MouseDown(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 if ServoToggle then
  SendCmd(CMD_SER_ON)
 else
  SendCmd(CMD_SER_OFF);

 ServoToggle:=not ServoToggle;
end;

procedure TForm1.Image12MouseDown(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_GET_A0);
end;

procedure TForm1.Image1MouseDown(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_RL);
end;

procedure TForm1.Image2MouseDown(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
```

```
 SendCmd(CMD_GO);
end;

procedure TForm1.Image2MouseUp(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_ST);
end;

procedure TForm1.Image3MouseDown(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_RR);
end;

procedure TForm1.Image4MouseDown(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_TL);
end;

procedure TForm1.Image5MouseDown(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_BUZ_ON);
end;

procedure TForm1.Image5MouseUp(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_BUZ_OFF);
end;
```

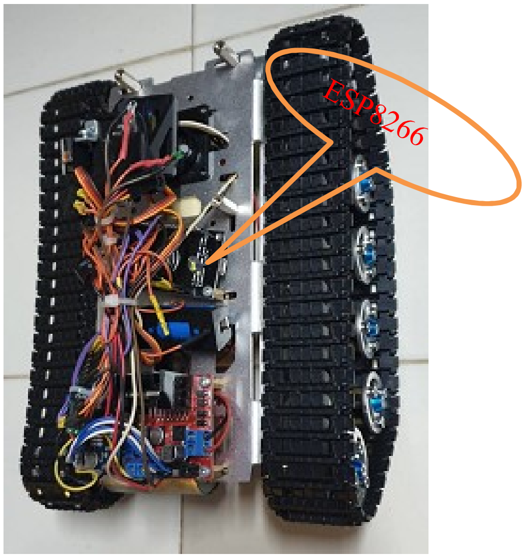
```

procedure TForm1.Image6MouseDown(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_TR);
end;

procedure TForm1.Image7MouseDown(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_LED1_ON);
end;

procedure TForm1.Image7MouseUp(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_LED1_OFF);
end;

procedure TForm1.Image8MouseDown(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_BK);
end;

procedure TForm1.Image9MouseDown(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_LED2_ON);
end;
```

```
procedure TForm1.Image9MouseUp(Sender: TObject;
Button: TMouseButton;
  Shift: TShiftState; X, Y: Single);
begin
 SendCmd(CMD_LED2_OFF);
end;

procedure TForm1.SendCmd(Cmd: String);
begin
 if IdTCPClient1.Connected then
  begin
   IdTCPClient1.IOHandler.InputBuffer.Clear;
   IdTCPClient1.Socket.WriteLn(Cmd);
   LbAck.Text:=IdTCPClient1.Socket.ReadLn;
  end
 else
  begin
   LbStatus.Text:='No Connection!';
  end;

end;

end.
```

Code\week10\ Ex_Robot_WiFi_1

ในส่วนควบคุมหุ่นยนต์ใช้ Microcontroller ESP8266 ซึ่งมี Build in Wi-Fi

Example



Code

```
#include <ESP8266WiFi.h>
#include <Servo.h>   // Include the library
// Default Server IP 192.168.4.1

// I/O Define
#define SW1 D0     // switch input Active Low
#define LED1 D5
#define LED2 D6
#define BUZZER D7
```

```
#define Servo1 D8
#define PWML1 D1
#define PWML2 D2
#define PWMR1 D3
#define PWMR2 D4
#define Analog0 A0

// Command
#define CMD_TL "TL"
#define CMD_TR "TR"
#define CMD_GO "GO"
#define CMD_BK "BK"
#define CMD_RL "RL"
#define CMD_RR "RR"
#define CMD_ST "ST"
#define CMD_BUZ_ON "BO"
#define CMD_BUZ_OFF "BF"
#define CMD_LED1_ON "1O"
#define CMD_LED1_OFF "1F"
#define CMD_LED2_ON "2O"
#define CMD_LED2_OFF "2F"
#define CMD_SER_ON "SO"
#define CMD_SER_OFF "SF"
#define CMD_GET_A0 "GA"

// Speed
#define GO_SPEED 100
#define BACK_SPEED 80
#define TURN_SPEED 70
#define ROTATE_SPEED 60

// Define Control
#define LED1_ON digitalWrite(LED1, HIGH)
#define LED1_OFF digitalWrite(LED1, LOW)
#define LED2_ON digitalWrite(LED2, HIGH)
#define LED2_OFF digitalWrite(LED2, LOW)
#define BUZZER_ON digitalWrite(BUZZER, HIGH)
```

```
#define BUZZER_OFF digitalWrite(BUZZER, LOW)
#define SERVO_ON MyServo.write(180); // Servo Pulse = 50Hz.
#define SERVO_OFF MyServo.write(0); // Servo Pulse = 50Hz.
// Wifi Configuration
#define PORT 2000 // Server Port
#define AP_NAME "ESP_Robot_Min"//SSID Name

WiFiServer server(PORT); //Server on port 2000
WiFiClient client;

Servo MyServo;        // Create the Servo and name it "servo1"
int AnalogVal0;

void GetAnalog0()
{
  AnalogVal0 = analogRead(Analog0);    // read the Analog input pin
}

void RobotGo(int Speed)
{
  analogWrite(PWML1, Speed);
  analogWrite(PWML2, 0);
  analogWrite(PWMR1, Speed);
  analogWrite(PWMR2, 0);
}

void RobotBack(int Speed)
{
  analogWrite(PWML1, 0);
  analogWrite(PWML2, Speed);
  analogWrite(PWMR1, 0);
  analogWrite(PWMR2, Speed);
}
```

```
void RobotTurnL(int Speed)
{
  analogWrite(PWML1, 0);
  analogWrite(PWML2, 0);
  analogWrite(PWMR1, Speed);
  analogWrite(PWMR2, 0);
}

void RobotTurnR(int Speed)
{
  analogWrite(PWML1, Speed);
  analogWrite(PWML2, 0);
  analogWrite(PWMR1, 0);
  analogWrite(PWMR2, 0);
}

void RobotRotateL(int Speed)
{
  analogWrite(PWML1, 0);
  analogWrite(PWML2, Speed);
  analogWrite(PWMR1, Speed);
  analogWrite(PWMR2, 0);
}

void RobotRotateR(int Speed)
{
  analogWrite(PWML1, Speed);
  analogWrite(PWML2, 0);
  analogWrite(PWMR1, 0);
  analogWrite(PWMR2, Speed);
}

void RobotStop()
{
  analogWrite(PWML1, 0);
  analogWrite(PWML2, 0);
```

```
  analogWrite(PWMR1, 0);
  analogWrite(PWMR2, 0);
}

bool ChkCmd(String Cmd)
{
  if (Cmd == CMD_TL)
  {
    RobotTurnL(TURN_SPEED);
    Serial.println(CMD_TL);
    client.println(CMD_TL);
    return (1);
  }
  else if (Cmd == CMD_TR)
  {
    RobotTurnR(TURN_SPEED);
    Serial.println(CMD_TR);
    client.println(CMD_TR);
    return (1);
  }
  else if (Cmd == CMD_RL)
  {
    RobotRotateL(ROTATE_SPEED);
    Serial.println(CMD_RL);
    client.println(CMD_RL);
    return (1);
  }
  else if (Cmd == CMD_RR)
  {
    RobotRotateR(ROTATE_SPEED);
    Serial.println(CMD_RR);
    client.println(CMD_RR);
    return (1);
  }
  else if (Cmd == CMD_GO)
  {
    RobotGo(GO_SPEED);
  Serial.println(CMD_GO);
  client.println(CMD_GO);
  return (1);
}
else if (Cmd == CMD_BK)
{
  RobotBack(BACK_SPEED);
  Serial.println(CMD_BK);
  client.println(CMD_BK);
  return (1);
}
else if (Cmd == CMD_ST)
{
  RobotStop();
  Serial.println(CMD_ST);
  client.println(CMD_ST);
  return (1);
}

else if (Cmd == CMD_BUZ_ON)
{
  BUZZER_ON;
  Serial.println(CMD_BUZ_ON);
  client.println(CMD_BUZ_ON);
  return (1);
}
else if (Cmd == CMD_BUZ_OFF)
{
  BUZZER_OFF;
  Serial.println(CMD_BUZ_OFF);
  client.println(CMD_BUZ_OFF);
  return (1);
}
else if (Cmd == CMD_LED1_ON)
{
  LED1_ON;
  Serial.println(CMD_LED1_ON);
```

```
  client.println(CMD_LED1_ON);
  return (1);
}
else if (Cmd == CMD_LED1_OFF)
{
  LED1_OFF;
  Serial.println(CMD_LED1_OFF);
  client.println(CMD_LED1_OFF);
  return (1);
}
else if (Cmd == CMD_LED2_ON)
{
  LED2_ON;
  Serial.println(CMD_LED2_ON);
  client.println(CMD_LED2_ON);
  return (1);
}
else if (Cmd == CMD_LED2_OFF)
{
  LED2_OFF;
  Serial.println(CMD_LED2_OFF);
  client.println(CMD_LED2_OFF);
  return (1);
}
else if (Cmd == CMD_SER_ON)
{
  SERVO_ON;
  Serial.println(CMD_SER_ON);
  client.println(CMD_SER_ON);
  return (1);
}
else if (Cmd == CMD_SER_OFF)
{
  SERVO_OFF;
  Serial.println(CMD_SER_OFF);
  client.println(CMD_SER_OFF);
  return (1);
  }
  else if (Cmd == CMD_GET_A0)
  {
    GetAnalog0();
    Serial.print(CMD_GET_A0);
    Serial.print("=");
    Serial.println(AnalogVal0);
    client.print(CMD_GET_A0);
    client.print("=");
    client.println(AnalogVal0);
    return (1);
  }
  else
  {
    return (0);
  }
}

void setup()
{
  Serial.begin(115200); // On board USB Serial
  Serial.println("SSID:" + String(AP_NAME));
  Serial.print("Server IP: 92.168.4.1");
  Serial.println(" Port:" + String(PORT));;
  WiFi.mode(WIFI_AP); // Ap Mode
  WiFi.softAP(AP_NAME); // Set SSID Name
  server.begin(); // Start Server
  Serial.println("Server Begin!");
  pinMode(LED1, OUTPUT);
  LED1_OFF;
  pinMode(LED2, OUTPUT);
  LED2_OFF;
  pinMode(BUZZER, OUTPUT);
```

Code\week10\ EX_Robot1_AP_ino_1

```
  BUZZER_OFF;
  pinMode(SW1, INPUT_PULLUP);
  delay(10);
  MyServo.attach(Servo1);
  SERVO_OFF;
  analogWriteRange(100);  // 1-1023 (100 = 0-100%) PWM
  analogWriteFreq(10000); // 1 = 1Hz. , 10000 = 10kHz.
  RobotStop();
}

void loop()
{
  client = server.available();
  if (!client) // Check Cliect
    return; // Skip if no client
  Serial.println("New client:"); // There is a client
  Serial.println(client.remoteIP()); // There is a client

  while (client.connected())
  {
    if (client.available()) // There are some data in
    {
      String Cmd = client.readStringUntil('\r');
      Cmd.trim();
      if (ChkCmd(Cmd) == 0)
        client.println("ERROR");
    }
  }
  delay(1);
  client.stop(); // close connection
  Serial.println("Client disconnect");
}
```

Exercise

- *อธิบายหลักการทำงานของ RF Module, GSM Module, Wi-Fi, LAN และ OPC*
- *เขียนโปรแกรมควบคุมหุ่นยนต์ผ่าน Wi-Fi ให้เคลื่อนที่ไปข้างหน้า ถอยหลัง และหยุด ด้วยการกดปุ่ม*

Assignment

- *เขียนโปรแกรมอ่าน Input0 และเขียน Output0 ไปยัง PLC โดยใช้ OPC เป็นตัวกลาง*
- *เขียนโปรแกรม plot graph และแสดงผลข้อมูลในตาราง ของค่า Analog 0 ที่ส่งจาก Arduino UNO R3 มาทาง Bluetooth*

Answer Sheet

Code\week10\ Ans_WiFiControl_1

Code\week10\ Ans_OPC_1

Code\week10\ Ans_Bluetooth_1