

1. Software for the device

The main device in the project is Arduino based and due to this fact are used Arduino IDE and the Arduino C to be programmed. The Arduino C is language developed especially for Atmel microcontroller and in the case of this project Arduino UNO. The language is mixture between C, C++ and some special libraries only for Arduino.

The black box below is representing the final version of the embedded software for the devices. In the code at the beginning are initialized all of weight sensors (LOADCELL-s) and also the SHT85 sensors. Also it is defining the calibration factor for the load cells that later on this will be define the accuracy of the device. The first rows are for implementing the libraries for the sensors.

However, in the loop part of the code firstly is collecting the data of the SHT85 sensors and if there is not data (NAN) that the code is defining the value as 0 and this means that the sensor is not responding. This can be caused from damage or disconnection from the main device. After that is collecting the data from the load cells and putting all of the data in one big sting that will be transfer via the COM port to the PC (to the Firmware).

In the end of the code are visible all of the functions for communicating and collecting data from the sensors. Also the function for using the I2C expander that is giving quince addresses to all of the SHT85 sensors. Otherwise all of them will have same address and they won't work or if they work they will mix their data and won't be possible to identify which data from which sensor it is.

```
#include <Wire.h>
#include "HX711.h"
#include "SHTSensor.h"
extern "C" {
    #include "utility/twi.h"}
#define TCAADDR 0x70
#define LOADCELL_DOUT_PIN_1 3
#define LOADCELL_SCK_PIN_1 2
#define LOADCELL_DOUT_PIN_2 5
#define LOADCELL_SCK_PIN_2 4
#define LOADCELL_DOUT_PIN_3 7
#define LOADCELL_SCK_PIN_3 6
#define LOADCELL_DOUT_PIN_4 9
#define LOADCELL_SCK_PIN_4 8
#define LOADCELL_DOUT_PIN_5 11
#define LOADCELL_SCK_PIN_5 10
#define LOADCELL_DOUT_PIN_6 13
#define LOADCELL_SCK_PIN_6 12
```

```

HX711 scale[6];
SHTSensor sht[7];
void tcasselect(uint8_t i);
int reading_loadcell(int number);
void setupsht85(int index);
int temp(int index);
int hum(int index);
void setupscales(int index,int out, int sck);
float calibration_factor = -7050; //-7050 worked for my 440lb max scale setup
long reading_cell[6];
float temp_s[7];
float hum_s[7];
long zero_factor[6];
void setup()
{
    Serial.begin(9600);
    Wire.begin();
    setupscales(0,LOADCELL_DOUT_PIN_1, LOADCELL_SCK_PIN_1);
    setupscales(1,LOADCELL_DOUT_PIN_2, LOADCELL_SCK_PIN_2);
    setupscales(2,LOADCELL_DOUT_PIN_3, LOADCELL_SCK_PIN_3);
    setupscales(3,LOADCELL_DOUT_PIN_4, LOADCELL_SCK_PIN_4);
    setupscales(4,LOADCELL_DOUT_PIN_5, LOADCELL_SCK_PIN_5);
    setupscales(5,LOADCELL_DOUT_PIN_6, LOADCELL_SCK_PIN_6);
    for (int i = 0; i <= 6; i++)
    {
        setupsht85(i);
    }
}
void loop() {
    for (int j = 0; j <= 6; j++)
    {
        hum_s[j] = hum(j);
        temp_s[j]= temp(j);
    }
    for (int k = 0; k <= 5; k++)
    {
        reading_cell[k] = reading_loadcell(k);
    }
    if(isnan(sht[0].getHumidity())){hum_s[0]=0;temp_s[0]=0;}
    else{hum_s[0]=sht[0].getHumidity();temp_s[0]=sht[0].getTemperature();}
    if(isnan(sht[1].getHumidity())){hum_s[1]=0;temp_s[1]=0;}
    else{hum_s[1]=sht[1].getHumidity();temp_s[1]=sht[1].getTemperature();}
    if(isnan(sht[2].getHumidity())){hum_s[2]=0;temp_s[2]=0;}
    else{hum_s[2]=sht[2].getHumidity();temp_s[2]=sht[2].getTemperature();}
    if(isnan(sht[3].getHumidity())){hum_s[3]=0;temp_s[3]=0;}
    else{hum_s[3]=sht[3].getHumidity();temp_s[3]=sht[3].getTemperature();}
    if(isnan(sht[4].getHumidity())){hum_s[4]=0;temp_s[4]=0;}
    else{hum_s[4]=sht[4].getHumidity();temp_s[4]=sht[4].getTemperature();}
    if(isnan(sht[5].getHumidity())){hum_s[5]=0;temp_s[5]=0;}
}

```

```

    else{hum_s[5]=sht[5].getHumidity();temp_s[5]=sht[5].getTemperature();}
    if(isnan(sht[6].getHumidity())){hum_s[6]=0;temp_s[6]=0;}
    else{hum_s[6]=sht[6].getHumidity();temp_s[6]=sht[6].getTemperature();}
    String H1 = String(hum_s[0]); String W1 = String(reading_cell[0]);String H
2 = String(hum_s[1]);
    String W2 = String(reading_cell[1]);String H3 = String(hum_s[2]);String W3
= String(reading_cell[2]);
    String H4 = String(hum_s[3]);String W4 = String(reading_cell[3]);String H5
= String(hum_s[4]);
    String W5 = String(reading_cell[4]);String H6 = String(hum_s[5]);String W6
= String(reading_cell[5]);
    String T0 = String(temp_s[6]);
    String T1 = String(temp_s[0]); String T2 = String(temp_s[1]); String T3 =
String(temp_s[2]);
    String T4 = String(temp_s[3]); String T5 = String(temp_s[4]);String T6 =
String(temp_s[5]);
    Serial.println(H1+", "+W1+", "+H2+", "+W2+", "+H3+", "+W3+", "+H4+", "+W4+", "+H5+",
"+W5+", "+H6+", "+W6+", "+T0+", "+T1+", "+T2+", "+T3+", "+T4+", "+T5+", "+T6");
    delay(300);
}
void tcselect(uint8_t i)
{
    if (i > 7) return;
    Wire.beginTransmission(TCAADDR);
    Wire.write(1 << i);
    Wire.endTransmission();
}
int reading_loadcell(int number)
{
    float value = 0;
    scale[number].set_scale(calibration_factor);
    value = (((scale[number].get_units()*4.5359237)/2));
    return value;
}
void setupscales(int index,int out, int sck)
{
    scale[index].begin(out, sck);
    scale[index].set_scale();
    scale[index].tare();
    zero_factor[index] = scale[index].read_average();
}
void setupsht85(int index)
{
    tcselect(index);
    sht[index].setAccuracy(SHTSensor::SHT_ACCURACY_MEDIUM);
    if (sht[index].init())
    { //Serial.println(F("Yes it is connected"));
    }
}

```

```

else
{ //Serial.println(F("No it is not connected"));
}
}
int temp(int index)
{
    long value_temp;
    tcselect(index);
    if (sht[index].readSample())
    {
        value_temp = sht[index].getTemperature();
    }
    return value_temp;
}
int hum(int index)
{
    long value_hum;
    tcselect(index);
    if (sht[index].readSample())
    {
        value_hum = sht[index].getHumidity();
    }
    return value_hum;
}

```

2. Software for firmware

The picture below is the visual representation of the code for the firmware. The client in the end will see only this representation. There are 12 graphs for the 6 cups (6 for weight sensors and 6 for the humidity of each of the cups). Also in right there is a column with temperature for each of the cups and also for the temperature of the oven.



There is a second window that the client will establish the connection between the computer and the device. Also there is a feature that can log the data in a txt file. After that this txt file really easy can be transferred to excel or any other file format.

REV'IT! LAB

DARK MODE Off ☐

Graphs

Connection

PORT

BAUD

PARITY

DATA BITS



STOP BITS

DATA LOGGER

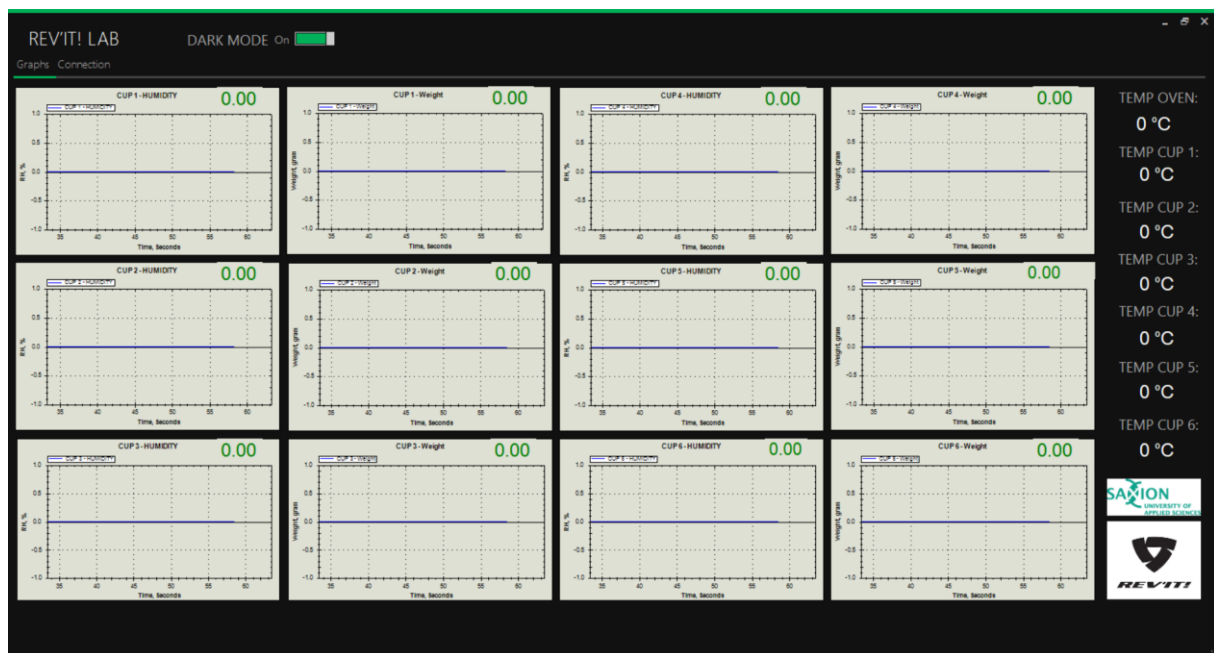
☐ ENABLE DATA LOGGER

CONNECT

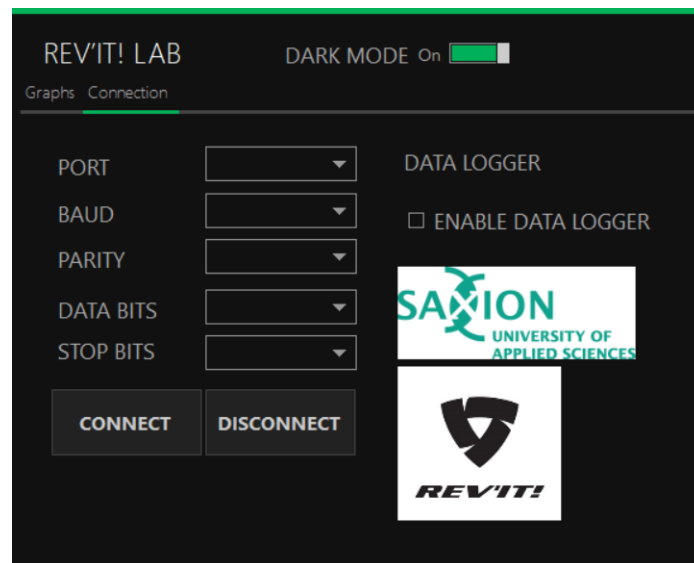
DISCONNECT

Also there is a dark mode for the graphs visualization. All of the components are going white and meanwhile the background is going black.



Also the dark mode is working in the connection windows. All of the components are change in black. Only the logos keep being white.



All of the functions described above are in the code below. The main logic of the code is following: the code is reading a string and separate it by special symbol in that case comma. And after that is representing the values in the graphs and the column for the temperature. Also there is a part that is establishing the connection between the computer and firmware.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO.Ports;
using System.Diagnostics;
using System.Threading;
using System.Windows.Forms.DataVisualization.Charting;
using ZedGraph;

namespace FFD_GUI
{
    public partial class Main : MetroFramework.Forms.MetroForm
    {
        string[] arrList = new string[19];
        System.IO.StreamWriter out_file;
```

```

class Settings
{
    public string[] returndata = new string[65];
    public string[] Yaxis = new string[13];
    public string[] Xaxis = new string[13];
    public string[] Danger1 = new string[15];
    public string[] Danger2 = new string[15];
    public string[] Danger3 = new string[15];
    public string[] sensor_data = new string[13];
    public string[] calib = new string[13];
    public string[] calibbuff = new string[13];
}
class Datacon
{
    public string[] rdata = new string[28];
    public string[] dataport_sensor = new string[28];
    public double[] data = new double[13];
}
Datacon d = new Datacon();
Settings u = new Settings();
//double temp = 0;
//double humidity = 0;
int TickStart1;
int TickStart2;
int TickStart3;
int TickStart4;
int TickStart5;
int TickStart6;
int TickStart7;
int TickStart8;
int TickStart9;
int TickStart10;
int TickStart11;
int TickStart12;

public Main()
{
    InitializeComponent();
    openFileDialog1.Filter = "Text|*.txt";
    cbBaud.Items.Add(9600);
    cbBaud.Items.Add(14400);
    cbBaud.Items.Add(19200);
    cbBaud.Items.Add(38400);
    cbBaud.Items.Add(57600);
    cbBaud.Items.Add(74880);
    cbBaud.Items.Add(115200);
    cbBaud.Items.Add(230400);
    cbBaud.Items.Add(256000);
    cbBaud.Items.Add(460800);
}

```

```

        cbBaud.Items.Add(921600);

        cbDatabits.Items.Add(8);
        cbDatabits.Items.Add(7);
        cbDatabits.Items.Add(6);
        cbDatabits.Items.Add(5);

        cbStopbits.Items.Add("One");
        cbStopbits.Items.Add("OnePointFive");
        cbStopbits.Items.Add("Two");

        cbParity.Items.Add("None");
        cbParity.Items.Add("Even");
        cbParity.Items.Add("Odd");
        cbParity.Items.Add("Mark");
        cbParity.Items.Add("Space");
        lblsensor13.ForeColor = Color.Green;
        lblsensor14.ForeColor = Color.Green;
        lblsensor15.ForeColor = Color.Green;
        lblsensor16.ForeColor = Color.Green;
        lblsensor17.ForeColor = Color.Green;
        lblsensor18.ForeColor = Color.Green;
        lblsensor19.ForeColor = Color.Green;
        timer16.Tick += new EventHandler(DoUpdate); // Everytime timer ticks, timer_Tick will be called
        timer16.Enabled = true;
        timer15.Tick += new EventHandler(DoUpdate); // Everytime timer ticks, timer_Tick will be called
        timer15.Enabled = true;
        timer2.Tick += new EventHandler(DoUpdate); // Everytime timer ticks, timer_Tick will be called
        timer2.Enabled = true; // Enable the timer
        timer2.Start(); // Start the timer
        timer1.Tick += new EventHandler(DoUpdate); // Everytime timer ticks, timer_Tick will be called
        timer1.Enabled = true; // Enable the timer
        timer1.Start();
        timer3.Tick += new EventHandler(DoUpdate); // Everytime timer ticks, timer_Tick will be called
        timer3.Enabled = true; // Enable the timer
        timer3.Start(); // Start the timer
        timer4.Tick += new EventHandler(DoUpdate); // Everytime timer ticks, timer_Tick will be called
        timer4.Enabled = true; // Enable the timer
        timer4.Start(); // Start the timer
        timer5.Tick += new EventHandler(DoUpdate); // Everytime timer ticks, timer_Tick will be called
        timer5.Enabled = true; // Enable the timer
        timer5.Start(); // Start the timer

```



```

        timer6.Tick += new EventHandler(DoUpdate); // Everytime timer tick
s, timer_Tick will be called
        timer6.Enabled = true; // Enable the timer
        timer6.Start(); // Start the timer
        timer7.Tick += new EventHandler(DoUpdate); // Everytime timer tick
s, timer_Tick will be called
        timer7.Enabled = true; // Enable the timer
        timer7.Start(); // Start the timer
        timer8.Tick += new EventHandler(DoUpdate); // Everytime timer tick
s, timer_Tick will be called
        timer8.Enabled = true; // Enable the timer
        timer8.Start(); // Start the timer
        timer9.Tick += new EventHandler(DoUpdate); // Everytime timer tick
s, timer_Tick will be called
        timer9.Enabled = true; // Enable the timer
        timer9.Start(); // Start the timer
        timer10.Tick += new EventHandler(DoUpdate); // Everytime timer tic
ks, timer_Tick will be called
        timer10.Enabled = true; // Enable the timer
        timer10.Start(); // Start the timer
        timer11.Tick += new EventHandler(DoUpdate); // Everytime timer tic
ks, timer_Tick will be called
        timer11.Enabled = true; // Enable the timer
        timer11.Start(); // Start the timer
        timer11.Tick += new EventHandler(DoUpdate); // Everytime timer tic
ks, timer_Tick will be called
        timer11.Enabled = true; // Enable the timer
        timer11.Start(); // Start the timer
        timer12.Tick += new EventHandler(DoUpdate); // Everytime timer tic
ks, timer_Tick will be called
        timer12.Enabled = true; // Enable the timer
        timer12.Start(); // Start the timer
        timer13.Tick += new EventHandler(DoUpdate); // Everytime timer tic
ks, timer_Tick will be called
        timer13.Enabled = true; // Enable the timer
        timer13.Start(); // Start the timer
        timer14.Tick += new EventHandler(DoUpdate); // Everytime timer tic
ks, timer_Tick will be called
        timer14.Enabled = true; // Enable the timer
        timer14.Start();
    }

    public void DoUpdate(object sender, EventArgs e)
    {
    }
    private void connectionToolStripMenuItem_Click(object sender, EventArgs e)
    {

```

```

    }

    private void optionsToolStripMenuItem_Click(object sender, EventArgs e
)
    {
    }

    private void aboutToolStripMenuItem_Click(object sender, EventArgs e)
    {
    }
    //
    private void Main_Load(object sender, EventArgs e)
    {
        this.StyleManager = metroStyleManager1;
        GraphPane myPane1 = zedGraphControl1.GraphPane;
        myPane1.Title.Text = "CUP 1 - HUMIDITY";
        myPane1.XAxis.Title.Text = "Time, Seconds";
        myPane1.YAxis.Title.Text = "RH, %";

        RollingPointPairList list1 = new RollingPointPairList(60000);
        LineItem Curve1 = myPane1.AddCurve("CUP 1 -
HUMIDITY", list1, Color.Blue, SymbolType.None);
        myPane1.YAxis.Scale.MaxAuto = true;
        myPane1.YAxis.Scale.MinAuto = true;
        myPane1.XAxis.Scale.MaxAuto = true;
        myPane1.XAxis.Scale.MinAuto = true;
        myPane1.XAxis.MajorGrid.IsVisible = true;
        myPane1.YAxis.MajorGrid.IsVisible = true;
        zedGraphControl1.AxisChange();
        TickStart1 = Environment.TickCount;

        GraphPane myPane2 = zedGraphControl2.GraphPane;
        myPane2.Title.Text = "CUP 1 - Weight";
        myPane2.XAxis.Title.Text = "Time, Seconds";
        myPane2.YAxis.Title.Text = "Weight, gram";
        myPane2.XAxis.MajorGrid.IsVisible = true;
        myPane2.YAxis.MajorGrid.IsVisible = true;
        RollingPointPairList list2 = new RollingPointPairList(60000);
        LineItem Curve2 = myPane2.AddCurve("CUP 1 -
Weight", list2, Color.Blue, SymbolType.None);
        //
        myPane2.YAxis.Scale.MaxAuto = true;
        myPane2.YAxis.Scale.MinAuto = true;
        myPane2.XAxis.Scale.MaxAuto = true;
        myPane2.XAxis.Scale.MinAuto = true;

        zedGraphControl2.AxisChange();
        TickStart2 = Environment.TickCount;
    }

```

```

/*Display the graph 3 of contents*/
//Theta graph//
GraphPane myPane3 = zedGraphControl3.GraphPane;
myPane3.Title.Text = "CUP 2 - HUMIDITY";
myPane3.XAxis.Title.Text = "Time, Seconds";
myPane3.YAxis.Title.Text = "RH, %";
myPane3.XAxis.MajorGrid.IsVisible = true;
myPane3.YAxis.MajorGrid.IsVisible = true;
RollingPointPairList list3 = new RollingPointPairList(60000);
LineItem Curve3 = myPane3.AddCurve("CUP 2 -
HUMIDITY", list3, Color.Blue, SymbolType.None);

myPane3.YAxis.Scale.MaxAuto = true;
myPane3.YAxis.Scale.MinAuto = true;
myPane3.XAxis.Scale.MaxAuto = true;
myPane3.XAxis.Scale.MinAuto = true;

zedGraphControl3.AxisChange();
TickStart3 = Environment.TickCount;

GraphPane myPane4 = zedGraphControl4.GraphPane;
myPane4.Title.Text = "CUP 2 - Weight";
myPane4.XAxis.Title.Text = "Time, Seconds";
myPane4.YAxis.Title.Text = "Weight, gram";
myPane4.XAxis.MajorGrid.IsVisible = true;
myPane4.YAxis.MajorGrid.IsVisible = true;
RollingPointPairList list4 = new RollingPointPairList(60000);
LineItem Curve4 = myPane4.AddCurve("CUP 2 -
Weight", list4, Color.Blue, SymbolType.None);

myPane4.YAxis.Scale.MaxAuto = true;
myPane4.YAxis.Scale.MinAuto = true;
myPane4.XAxis.Scale.MaxAuto = true;
myPane4.XAxis.Scale.MinAuto = true;

zedGraphControl4.AxisChange();
TickStart4 = Environment.TickCount;

GraphPane myPane5 = zedGraphControl5.GraphPane;
myPane5.Title.Text = "CUP 3 - HUMIDITY";
myPane5.XAxis.Title.Text = "Time, Seconds";
myPane5.YAxis.Title.Text = "RH, %";
myPane5.XAxis.MajorGrid.IsVisible = true;
myPane5.YAxis.MajorGrid.IsVisible = true;
RollingPointPairList list5 = new RollingPointPairList(60000);
LineItem Curve5 = myPane5.AddCurve("CUP 3 -
HUMIDITY", list5, Color.Blue, SymbolType.None);

```

```

myPane5.YAxis.Scale.MaxAuto = true;
myPane5.YAxis.Scale.MinAuto = true;
myPane5.XAxis.Scale.MaxAuto = true;
myPane5.XAxis.Scale.MinAuto = true;

zedGraphControl5.AxisChange();
TickStart5 = Environment.TickCount;

GraphPane myPane6 = zedGraphControl6.GraphPane;
myPane6.Title.Text = "CUP 3 - Weight";
myPane6.XAxis.Title.Text = "Time, Seconds";
myPane6.YAxis.Title.Text = "Weight, gram";
myPane6.XAxis.MajorGrid.IsVisible = true;
myPane6.YAxis.MajorGrid.IsVisible = true;
RollingPointPairList list6 = new RollingPointPairList(60000);
LineItem Curve6 = myPane6.AddCurve("CUP 3 -
Weight", list6, Color.Blue, SymbolType.None);

myPane6.YAxis.Scale.MaxAuto = true;
myPane6.YAxis.Scale.MinAuto = true;
myPane6.XAxis.Scale.MaxAuto = true;
myPane6.XAxis.Scale.MinAuto = true;

zedGraphControl6.AxisChange();
TickStart6 = Environment.TickCount;

GraphPane myPane7 = zedGraphControl7.GraphPane;
myPane7.Title.Text = "CUP 4 - HUMIDITY";
myPane7.XAxis.Title.Text = "Time, Seconds";
myPane7.YAxis.Title.Text = "RH, %";
myPane7.XAxis.MajorGrid.IsVisible = true;
myPane7.YAxis.MajorGrid.IsVisible = true;
RollingPointPairList list7 = new RollingPointPairList(60000);
LineItem Curve7 = myPane7.AddCurve("CUP 4 -
HUMIDITY", list7, Color.Blue, SymbolType.None);

myPane7.YAxis.Scale.MaxAuto = true;
myPane7.YAxis.Scale.MinAuto = true;
myPane7.XAxis.Scale.MaxAuto = true;
myPane7.XAxis.Scale.MinAuto = true;

zedGraphControl7.AxisChange();
TickStart7 = Environment.TickCount;

GraphPane myPane8 = zedGraphControl8.GraphPane;
myPane8.Title.Text = "CUP 4 - Weight";
myPane8.XAxis.Title.Text = "Time, Seconds";

```

```

        myPane8.YAxis.Title.Text = "Weight, gram";
        myPane8.XAxis.MajorGrid.IsVisible = true;
        myPane8.YAxis.MajorGrid.IsVisible = true;
        RollingPointPairList list8 = new RollingPointPairList(60000);
        LineItem Curve8 = myPane8.AddCurve("CUP 4 -
Weight", list8, Color.Blue, SymbolType.None);

        myPane8.YAxis.Scale.MaxAuto = true;
        myPane8.YAxis.Scale.MinAuto = true;
        myPane8.XAxis.Scale.MaxAuto = true;
        myPane8.XAxis.Scale.MinAuto = true;

        zedGraphControl8.AxisChange();
        TickStart8 = Environment.TickCount;

        GraphPane myPane9 = zedGraphControl9.GraphPane;
        myPane9.Title.Text = "CUP 5 - HUMIDITY";
        myPane9.XAxis.Title.Text = "Time, Seconds";
        myPane9.YAxis.Title.Text = "RH, %";
        myPane9.XAxis.MajorGrid.IsVisible = true;
        myPane9.YAxis.MajorGrid.IsVisible = true;
        RollingPointPairList list9 = new RollingPointPairList(60000);
        LineItem Curve9 = myPane9.AddCurve("CUP 5 -
HUMIDITY", list9, Color.Blue, SymbolType.None);

        myPane9.YAxis.Scale.MaxAuto = true;
        myPane9.YAxis.Scale.MinAuto = true;
        myPane9.XAxis.Scale.MaxAuto = true;
        myPane9.XAxis.Scale.MinAuto = true;

        zedGraphControl9.AxisChange();
        TickStart9 = Environment.TickCount;

        GraphPane myPane10 = zedGraphControl10.GraphPane;
        myPane10.Title.Text = "CUP 5 - Weight";
        myPane10.XAxis.Title.Text = "Time, Seconds";
        myPane10.YAxis.Title.Text = "Weight, gram";
        myPane10.XAxis.MajorGrid.IsVisible = true;
        myPane10.YAxis.MajorGrid.IsVisible = true;
        RollingPointPairList list10 = new RollingPointPairList(60000);
        LineItem Curve10 = myPane10.AddCurve("CUP 5 -
Weight", list10, Color.Blue, SymbolType.None);

        myPane10.YAxis.Scale.MaxAuto = true;
        myPane10.YAxis.Scale.MinAuto = true;

```

```

myPane10.XAxis.Scale.MaxAuto = true;
myPane10.XAxis.Scale.MinAuto = true;

zedGraphControl10.AxisChange();
TickStart10 = Environment.TickCount;

GraphPane myPane11 = zedGraphControl11.GraphPane;
myPane11.Title.Text = "CUP 6 - HUMIDITY";
myPane11.XAxis.Title.Text = "Time, Seconds";
myPane11.YAxis.Title.Text = "RH, %";
myPane11.XAxis.MajorGrid.IsVisible = true;
myPane11.YAxis.MajorGrid.IsVisible = true;
RollingPointPairList list11 = new RollingPointPairList(60000);
LineItem Curve11 = myPane11.AddCurve("CUP 6 -
HUMIDITY", list11, Color.Blue, SymbolType.None);

myPane11.YAxis.Scale.MaxAuto = true;
myPane11.YAxis.Scale.MinAuto = true;
myPane11.XAxis.Scale.MaxAuto = true;
myPane11.XAxis.Scale.MinAuto = true;

zedGraphControl11.AxisChange();
TickStart11 = Environment.TickCount;

GraphPane myPane12 = zedGraphControl12.GraphPane;
myPane12.Title.Text = "CUP 6 - Weight";
myPane12.XAxis.Title.Text = "Time, Seconds";
myPane12.YAxis.Title.Text = "Weight, gram";
myPane12.XAxis.MajorGrid.IsVisible = true;
myPane12.YAxis.MajorGrid.IsVisible = true;
RollingPointPairList list12 = new RollingPointPairList(60000);
LineItem Curve12 = myPane12.AddCurve("CUP 6 -
Weight", list12, Color.Blue, SymbolType.None);

myPane12.YAxis.Scale.MaxAuto = true;
myPane12.YAxis.Scale.MinAuto = true;
myPane12.XAxis.Scale.MaxAuto = true;
myPane12.XAxis.Scale.MinAuto = true;
zedGraphControl12.AxisChange();
TickStart12 = Environment.TickCount;
}
private void Connection_Load(object sender, EventArgs e)
{
    cbBaud.Enabled = true;
    cbPorts.Enabled = true;
    cbDatabits.Enabled = true;

```

```

        cbStopbits.Enabled = true;
        cbParity.Enabled = true;
        //Disable button control

        //Load value//
        cbBaud.Items.Add(9600);
        cbBaud.Items.Add(14400);
        cbBaud.Items.Add(19200);
        cbBaud.Items.Add(38400);
        cbBaud.Items.Add(57600);
        cbBaud.Items.Add(74880);
        cbBaud.Items.Add(115200);
        cbBaud.Items.Add(230400);
        cbBaud.Items.Add(256000);
        cbBaud.Items.Add(460800);
        cbBaud.Items.Add(921600);

        cbDatabits.Items.Add(8);
        cbDatabits.Items.Add(7);
        cbDatabits.Items.Add(6);
        cbDatabits.Items.Add(5);

        cbStopbits.Items.Add("One");
        cbStopbits.Items.Add("OnePointFive");
        cbStopbits.Items.Add("Two");

        cbParity.Items.Add("None");
        cbParity.Items.Add("Even");
        cbParity.Items.Add("Odd");
        cbParity.Items.Add("Mark");
        cbParity.Items.Add("Space");
        btnDisConn.Enabled = false;
    }
    private void serialPort1_DataReceived(object sender, SerialDataReceive
dEventArgs e)
    {
        try
        {
            //split data receive from serialport
            arrList = serialPort1.ReadLine().Split(',');

        }
        catch
        {
            return;
        }
    }
    private void BtnConn_Click(object sender, EventArgs e)
    {

```

```

    }
    private void BtnDisConn_Click(object sender, EventArgs e)
    {

    }
    private void Datalogger_checkbox_CheckedChanged(object sender, EventArgs e)
    {
    }
    public void logger_saveinfo()
    {
        try
        {
            using (System.IO.StreamWriter file =
                new System.IO.StreamWriter(@datalogger_checkbox.Text, true))
            {
                string year = DateTime.Now.Year.ToString("0000");
                string month = DateTime.Now.Month.ToString("00");
                string date = DateTime.Now.Day.ToString("00");
                string hour = DateTime.Now.Hour.ToString("00");
                string minute = DateTime.Now.Minute.ToString("00");
                string second = DateTime.Now.Second.ToString("00");
                string CurrentDate = date + "/" + month + "/" + year + " || " + hour + ":" + minute + ":" + second;
                file.Write(CurrentDate + "," + arrList[0] + "," + arrList[1]
                    + "," + arrList[2] + "," + arrList[3] + "," + arrList[4] + "," + arrList[5] +
                    "," + arrList[6] + "," + arrList[7] + "," + arrList[8] + "," + arrList[9] +
                    "," + arrList[10] + "," + arrList[11] + "," + arrList[12] + "," + arrList[13] +
                    "," + arrList[14] + "," + arrList[15] + "," + arrList[16] + "," + arrList[17]
                    + "," + arrList[18] + "\n");
            }
        }
        catch
        {
            return;
        }
    }

    int intlen = 0;
    private void Timer15_Tick(object sender, EventArgs e)
    {
        logger_saveinfo();
    }
    private void Timer16_Tick(object sender, EventArgs e)
    {

    }
    private void Chart1_Click(object sender, EventArgs e)

```



```

{

}

private void Chart14_Click(object sender, EventArgs e)
{

}

private void Lbl_Click(object sender, EventArgs e)
{

}

private void Button2_Click(object sender, EventArgs e)
{

}

private void ToolStripMenuItem2_Click(object sender, EventArgs e)
{
}

private void MenuStrip2_ItemClicked(object sender, ToolStripItemClickedEventArgs e)
{

}

private void CpuChart_Click(object sender, EventArgs e)
{

}

private void Timer1_Tick(object sender, EventArgs e)
{

    Draw();
    lblsensor1.ForeColor = Color.Green;
    lblsensor1.Text = String.Format("{0:0.00}", Convert.ToDouble(arrList[0]));

    lblsensor2.ForeColor = Color.Green;
    lblsensor2.Text = String.Format("{0:0.00}", Convert.ToDouble(arrList[1]));

    lblsensor3.ForeColor = Color.Green;
    lblsensor3.Text = String.Format("{0:0.00}", Convert.ToDouble(arrList[7]));

    lblsensor4.ForeColor = Color.Green;
    lblsensor4.Text = String.Format("{0:0.00}", Convert.ToDouble(arrList[3]));
}

```

```

        lblsensor5.ForeColor = Color.Green;
        lblsensor5.Text = String.Format("{0:0.00}", Convert.ToDouble(arrLi
st[4]));

        lblsensor6.ForeColor = Color.Green;
        lblsensor6.Text = String.Format("{0:0.00}", Convert.ToDouble(arrLi
st[5]));

        lblsensor7.ForeColor = Color.Green;
        lblsensor7.Text = String.Format("{0:0.00}", Convert.ToDouble(arrLi
st[6]));

        lblsensor8.ForeColor = Color.Green;
        lblsensor8.Text = String.Format("{0:0.00}", Convert.ToDouble(arrLi
st[2]));

        lblsensor9.ForeColor = Color.Green;
        lblsensor9.Text = String.Format("{0:0.00}", Convert.ToDouble(arrLi
st[8]));

        lblsensor10.ForeColor = Color.Green;
        lblsensor10.Text = String.Format("{0:0.00}", Convert.ToDouble(arrL
ist[9]));

        lblsensor11.ForeColor = Color.Green;
        lblsensor11.Text = String.Format("{0:0.00}", Convert.ToDouble(arrL
ist[10]));

        lblsensor12.ForeColor = Color.Green;
        lblsensor12.Text = String.Format("{0:0.00}", Convert.ToDouble(arrL
ist[11]));

        lblsensor13.Text = String.Format("{0} °C", Convert.ToDouble(arrLis
t[12]));

        lblsensor14.Text = String.Format("{0} °C", Convert.ToDouble(arrLis
t[13]));

        lblsensor15.Text = String.Format("{0} °C", Convert.ToDouble(arrLis
t[14]));

        lblsensor16.Text = String.Format("{0} °C", Convert.ToDouble(arrLis
t[15]));

        lblsensor17.Text = String.Format("{0} °C", Convert.ToDouble(arrLis
t[16]));

        lblsensor18.Text = String.Format("{0} °C", Convert.ToDouble(arrLis
t[17]));

        lblsensor19.Text = String.Format("{0} °C", Convert.ToDouble(arrLis
t[18]));

    }

    private void ChartTest_Click(object sender, EventArgs e)
    {
    }

    private void Draw()
    {
        if (zedGraphControl11.GraphPane.CurveList.Count <= 0)
            return;

```

```

        if (zedGraphControl12.GraphPane.CurveList.Count <= 0)
            return;
        if (zedGraphControl13.GraphPane.CurveList.Count <= 0)
            return;
        if (zedGraphControl14.GraphPane.CurveList.Count <= 0)
            return;
        if (zedGraphControl15.GraphPane.CurveList.Count <= 0)
            return;
        if (zedGraphControl16.GraphPane.CurveList.Count <= 0)
            return;
        if (zedGraphControl17.GraphPane.CurveList.Count <= 0)
            return;
        if (zedGraphControl18.GraphPane.CurveList.Count <= 0)
            return;
        if (zedGraphControl19.GraphPane.CurveList.Count <= 0)
            return;
        if (zedGraphControl110.GraphPane.CurveList.Count <= 0)
            return;
        if (zedGraphControl111.GraphPane.CurveList.Count <= 0)
            return;
        if (zedGraphControl112.GraphPane.CurveList.Count <= 0)
            return;

        LineItem curve1 = zedGraphControl11.GraphPane.CurveList[0] as LineI
tem;
        LineItem curve2 = zedGraphControl12.GraphPane.CurveList[0] as LineI
tem;
        LineItem curve3 = zedGraphControl13.GraphPane.CurveList[0] as LineI
tem;
        LineItem curve4 = zedGraphControl14.GraphPane.CurveList[0] as LineI
tem;
        LineItem curve5 = zedGraphControl15.GraphPane.CurveList[0] as LineI
tem;
        LineItem curve6 = zedGraphControl16.GraphPane.CurveList[0] as LineI
tem;
        LineItem curve7 = zedGraphControl17.GraphPane.CurveList[0] as LineI
tem;
        LineItem curve8 = zedGraphControl18.GraphPane.CurveList[0] as LineI
tem;
        LineItem curve9 = zedGraphControl19.GraphPane.CurveList[0] as LineI
tem;
        LineItem curve10 = zedGraphControl110.GraphPane.CurveList[0] as Lin
eItem;
        LineItem curve11 = zedGraphControl111.GraphPane.CurveList[0] as Lin
eItem;
        LineItem curve12 = zedGraphControl112.GraphPane.CurveList[0] as Lin
eItem;

```

```
if (curve1 == null)
    return;
if (curve2 == null)
    return;
if (curve3 == null)
    return;
if (curve4 == null)
    return;
if (curve5 == null)
    return;
if (curve6 == null)
    return;
if (curve7 == null)
    return;
if (curve8 == null)
    return;
if (curve9 == null)
    return;
if (curve10 == null)
    return;
if (curve11 == null)
    return;
if (curve12 == null)
    return;
//
IPointListEdit list1 = curve1.Points as IPointListEdit;
IPointListEdit list2 = curve2.Points as IPointListEdit;
IPointListEdit list3 = curve3.Points as IPointListEdit;
IPointListEdit list4 = curve4.Points as IPointListEdit;
IPointListEdit list5 = curve5.Points as IPointListEdit;
IPointListEdit list6 = curve6.Points as IPointListEdit;
IPointListEdit list7 = curve7.Points as IPointListEdit;
IPointListEdit list8 = curve8.Points as IPointListEdit;
IPointListEdit list9 = curve9.Points as IPointListEdit;
IPointListEdit list10 = curve10.Points as IPointListEdit;
IPointListEdit list11 = curve11.Points as IPointListEdit;
IPointListEdit list12 = curve12.Points as IPointListEdit;
//
if (list1 == null)
    return;
if (list2 == null)
    return;
if (list3 == null)
    return;
if (list4 == null)
    return;
if (list5 == null)
    return;
if (list6 == null)
```

```

        return;
    if (list7 == null)
        return;
    if (list8 == null)
        return;
    if (list9 == null)
        return;
    if (list10 == null)
        return;
    if (list11 == null)
        return;
    if (list12 == null)
        return;
    //
    double time1 = (Environment.TickCount - TickStart1) / 1000.0;
    double time2 = (Environment.TickCount - TickStart2) / 1000.0;
    double time3 = (Environment.TickCount - TickStart3) / 1000.0;
    double time4 = (Environment.TickCount - TickStart4) / 1000.0;
    double time5 = (Environment.TickCount - TickStart5) / 1000.0;
    double time6 = (Environment.TickCount - TickStart6) / 1000.0;
    double time7 = (Environment.TickCount - TickStart7) / 1000.0;
    double time8 = (Environment.TickCount - TickStart8) / 1000.0;
    double time9 = (Environment.TickCount - TickStart9) / 1000.0;
    double time10 = (Environment.TickCount - TickStart10) / 1000.0;
    double time11 = (Environment.TickCount - TickStart11) / 1000.0;
    double time12 = (Environment.TickCount - TickStart12) / 1000.0;
    //
    list1.Add(time1, Convert.ToDouble(arrList[0]));
    list2.Add(time2, Convert.ToDouble(arrList[1]));
    list3.Add(time3, Convert.ToDouble(arrList[2]));
    list4.Add(time4, Convert.ToDouble(arrList[3]));
    list5.Add(time5, Convert.ToDouble(arrList[4]));
    list6.Add(time6, Convert.ToDouble(arrList[5]));
    list7.Add(time7, Convert.ToDouble(arrList[6]));
    list8.Add(time8, Convert.ToDouble(arrList[7]));
    list9.Add(time9, Convert.ToDouble(arrList[8]));
    list10.Add(time10, Convert.ToDouble(arrList[9]));
    list11.Add(time11, Convert.ToDouble(arrList[10]));
    list12.Add(time12, Convert.ToDouble(arrList[11]));

    Scale xScale1 = zedGraphControl1.GraphPane.XAxis.Scale;
    Scale xScale2 = zedGraphControl2.GraphPane.XAxis.Scale;
    Scale xScale3 = zedGraphControl3.GraphPane.XAxis.Scale;
    Scale xScale4 = zedGraphControl4.GraphPane.XAxis.Scale;
    Scale xScale5 = zedGraphControl5.GraphPane.XAxis.Scale;
    Scale xScale6 = zedGraphControl6.GraphPane.XAxis.Scale;
    Scale xScale7 = zedGraphControl7.GraphPane.XAxis.Scale;
    Scale xScale8 = zedGraphControl8.GraphPane.XAxis.Scale;

```

```

Scale xScale9 = zedGraphControl9.GraphPane.XAxis.Scale;
Scale xScale10 = zedGraphControl10.GraphPane.XAxis.Scale;
Scale xScale11 = zedGraphControl11.GraphPane.XAxis.Scale;
Scale xScale12 = zedGraphControl12.GraphPane.XAxis.Scale;
//
Scale yScale1 = zedGraphControl11.GraphPane.YAxis.Scale;
Scale yScale2 = zedGraphControl12.GraphPane.YAxis.Scale;
Scale yScale3 = zedGraphControl13.GraphPane.YAxis.Scale;
Scale yScale4 = zedGraphControl14.GraphPane.YAxis.Scale;
Scale yScale5 = zedGraphControl15.GraphPane.YAxis.Scale;
Scale yScale6 = zedGraphControl16.GraphPane.YAxis.Scale;
Scale yScale7 = zedGraphControl17.GraphPane.YAxis.Scale;
Scale yScale8 = zedGraphControl18.GraphPane.YAxis.Scale;
Scale yScale9 = zedGraphControl9.GraphPane.YAxis.Scale;
Scale yScale10 = zedGraphControl10.GraphPane.YAxis.Scale;
Scale yScale11 = zedGraphControl11.GraphPane.YAxis.Scale;
Scale yScale12 = zedGraphControl12.GraphPane.YAxis.Scale;

//
if (time1 > xScale1.Max - xScale1.MajorStep)
{
    xScale1.Max = time1 + xScale1.MajorStep;
    xScale1.Min = xScale1.Max -
30; //Auto scale x axis in limit tim
}
if (time2 > xScale2.Max - xScale2.MajorStep)
{
    xScale2.Max = time2 + xScale2.MajorStep;
    xScale2.Min = xScale2.Max - 30;
}
if (time3 > xScale3.Max - xScale3.MajorStep)
{
    xScale3.Max = time3 + xScale3.MajorStep;
    xScale3.Min = xScale3.Max - 30;
}
if (time4 > xScale4.Max - xScale4.MajorStep)
{
    xScale4.Max = time4 + xScale4.MajorStep;
    xScale4.Min = xScale4.Max - 30;
}
if (time5 > xScale5.Max - xScale5.MajorStep)
{
    xScale5.Max = time5 + xScale5.MajorStep;
    xScale5.Min = xScale5.Max - 30;
}
if (time6 > xScale6.Max - xScale6.MajorStep)
{
    xScale6.Max = time6 + xScale6.MajorStep;
    xScale6.Min = xScale6.Max - 30;
}

```

```

    }
    if (time7 > xScale7.Max - xScale7.MajorStep)
    {
        xScale7.Max = time7 + xScale7.MajorStep;
        xScale7.Min = xScale7.Max - 30;
    }
    if (time8 > xScale8.Max - xScale8.MajorStep)
    {
        xScale8.Max = time8 + xScale8.MajorStep;
        xScale8.Min = xScale8.Max - 30;
    }
    if (time9 > xScale9.Max - xScale9.MajorStep)
    {
        xScale9.Max = time9 + xScale9.MajorStep;
        xScale9.Min = xScale9.Max - 30;
    }
    if (time10 > xScale10.Max - xScale10.MajorStep)
    {
        xScale10.Max = time10 + xScale10.MajorStep;
        xScale10.Min = xScale10.Max - 30;
    }
    if (time11 > xScale11.Max - xScale11.MajorStep)
    {
        xScale11.Max = time11 + xScale11.MajorStep;
        xScale11.Min = xScale11.Max - 30;
    }
    if (time12 > xScale12.Max - xScale12.MajorStep)
    {
        xScale12.Max = time12 + xScale12.MajorStep;
        xScale12.Min = xScale12.Max - 30;
    }

    //
    zedGraphControl11.AxisChange();
    zedGraphControl12.AxisChange();
    zedGraphControl13.AxisChange();
    zedGraphControl14.AxisChange();
    zedGraphControl15.AxisChange();
    zedGraphControl16.AxisChange();
    zedGraphControl17.AxisChange();
    zedGraphControl18.AxisChange();
    zedGraphControl19.AxisChange();
    zedGraphControl10.AxisChange();
    zedGraphControl11.AxisChange();
    zedGraphControl12.AxisChange();
    //
    zedGraphControl11.Invalidate();
    zedGraphControl12.Invalidate();
    zedGraphControl13.Invalidate();

```

```
zedGraphControl14.Invalidate();
zedGraphControl15.Invalidate();
zedGraphControl16.Invalidate();
zedGraphControl17.Invalidate();
zedGraphControl18.Invalidate();
zedGraphControl19.Invalidate();
zedGraphControl110.Invalidate();
zedGraphControl111.Invalidate();
zedGraphControl112.Invalidate();

}

private void ZedGraphControl1_Load(object sender, EventArgs e)
{

}

private void Lblsensor13_Click(object sender, EventArgs e)
{

}

private void Timer2_Tick(object sender, EventArgs e)
{

}

private void PictureBox3_Click(object sender, EventArgs e)
{
    System.Diagnostics.Process.Start("https://www.saxion.nl/onderzoek/
smart-industry/advanced-forensic-technology");
}
private void Label63_Click(object sender, EventArgs e)
{

}

private void Label1_Click(object sender, EventArgs e)
{

}

private void Lblsensor15_Click(object sender, EventArgs e)
{

}

private void Timer3_Tick(object sender, EventArgs e)
{
    GraphPane myPane1 = zedGraphControl11.GraphPane;
```



```
}

private void Timer4_Tick(object sender, EventArgs e)
{
    GraphPane myPane2 = zedGraphControl12.GraphPane;
}

private void Timer5_Tick(object sender, EventArgs e)
{
    GraphPane myPane3 = zedGraphControl13.GraphPane;
}

private void Timer6_Tick(object sender, EventArgs e)
{
    GraphPane myPane4 = zedGraphControl14.GraphPane;
}

private void Timer7_Tick(object sender, EventArgs e)
{
    GraphPane myPane5 = zedGraphControl15.GraphPane;
}

private void Timer8_Tick(object sender, EventArgs e)
{
    GraphPane myPane6 = zedGraphControl16.GraphPane;
}

private void Timer9_Tick(object sender, EventArgs e)
{
    GraphPane myPane7 = zedGraphControl17.GraphPane;
}

private void Timer10_Tick(object sender, EventArgs e)
{
    GraphPane myPane8 = zedGraphControl18.GraphPane;
}

private void Timer11_Tick(object sender, EventArgs e)
{
    GraphPane myPane9 = zedGraphControl19.GraphPane;
}

private void Timer12_Tick(object sender, EventArgs e)
{
    GraphPane myPane10 = zedGraphControl110.GraphPane;
}

private void Timer13_Tick(object sender, EventArgs e)
{

```

```

        GraphPane myPane11 = zedGraphControl11.GraphPane;
    }

    private void Timer14_Tick(object sender, EventArgs e)
    {
        GraphPane myPane12 = zedGraphControl12.GraphPane;
    }

    private void ZedGraphControl5_Load(object sender, EventArgs e)
    {

    }

    private void MetroProgressBar1_Click(object sender, EventArgs e)
    {

    }
    bool drum;
    private void MetroToggle1_CheckedChanged(object sender, EventArgs e)
    {
        if(drum == true)
        {
            metroStyleManager1.Theme = MetroFramework.MetroThemeStyle.Ligh
t;
            zedGraphControl1.GraphPane.Fill = new Fill(Color.FromArgb(255,
255,255));
            zedGraphControl1.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(255,255,255));
            zedGraphControl2.GraphPane.Fill = new Fill(Color.FromArgb(255,
255, 255));
            zedGraphControl2.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(255, 255, 255));
            zedGraphControl3.GraphPane.Fill = new Fill(Color.FromArgb(255,
255, 255));
            zedGraphControl3.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(255, 255, 255));
            zedGraphControl4.GraphPane.Fill = new Fill(Color.FromArgb(255,
255, 255));
            zedGraphControl4.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(255, 255, 255));
            zedGraphControl5.GraphPane.Fill = new Fill(Color.FromArgb(255,
255, 255));
            zedGraphControl5.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(255, 255, 255));
            zedGraphControl6.GraphPane.Fill = new Fill(Color.FromArgb(255,
255, 255));
            zedGraphControl6.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(255, 255, 255));

```

```

        zedGraphControl17.GraphPane.Fill = new Fill(Color.FromArgb(255,
255, 255));
        zedGraphControl17.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(255, 255, 255));
        zedGraphControl18.GraphPane.Fill = new Fill(Color.FromArgb(255,
255, 255));
        zedGraphControl18.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(255, 255, 255));
        zedGraphControl19.GraphPane.Fill = new Fill(Color.FromArgb(255,
255, 255));
        zedGraphControl19.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(255, 255, 255));
        zedGraphControl10.GraphPane.Fill = new Fill(Color.FromArgb(255
, 255, 255));
        zedGraphControl10.GraphPane.Chart.Fill = new Fill(Color.FromAr
gb(255, 255, 255));
        zedGraphControl11.GraphPane.Fill = new Fill(Color.FromArgb(255
, 255, 255));
        zedGraphControl11.GraphPane.Chart.Fill = new Fill(Color.FromAr
gb(255, 255, 255));
        zedGraphControl12.GraphPane.Fill = new Fill(Color.FromArgb(255
, 255, 255));
        zedGraphControl12.GraphPane.Chart.Fill = new Fill(Color.FromAr
gb(255, 255, 255));
        lblsensor1.BackColor = Color.FromArgb(255, 255, 255);
        lblsensor2.BackColor = Color.FromArgb(255, 255, 255);
        lblsensor3.BackColor = Color.FromArgb(255, 255, 255);
        lblsensor4.BackColor = Color.FromArgb(255, 255, 255);
        lblsensor5.BackColor = Color.FromArgb(255, 255, 255);
        lblsensor6.BackColor = Color.FromArgb(255, 255, 255);
        lblsensor7.BackColor = Color.FromArgb(255, 255, 255);
        lblsensor8.BackColor = Color.FromArgb(255, 255, 255);
        lblsensor9.BackColor = Color.FromArgb(255, 255, 255);
        lblsensor10.BackColor = Color.FromArgb(255, 255, 255);
        lblsensor11.BackColor = Color.FromArgb(255, 255, 255);
        lblsensor12.BackColor = Color.FromArgb(255, 255, 255);
        lblsensor13.ForeColor = Color.Green;
        lblsensor14.ForeColor = Color.Green;
        lblsensor15.ForeColor = Color.Green;
        lblsensor16.ForeColor = Color.Green;
        lblsensor17.ForeColor = Color.Green;
        lblsensor18.ForeColor = Color.Green;
        lblsensor19.ForeColor = Color.Green;
        drum = false;
    }
    else if(drum ==false)
    {
        metroStyleManager1.Theme = MetroFramework.MetroThemeStyle.Dark
;

```

```
zedGraphControl11.GraphPane.Fill = new Fill(Color.FromArgb(222,
224, 212));
zedGraphControl11.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(222, 224, 212));
zedGraphControl12.GraphPane.Fill = new Fill(Color.FromArgb(222,
224, 212));
zedGraphControl12.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(222, 224, 212));
zedGraphControl13.GraphPane.Fill = new Fill(Color.FromArgb(222,
224, 212));
zedGraphControl13.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(222, 224, 212));
zedGraphControl14.GraphPane.Fill = new Fill(Color.FromArgb(222,
224, 212));
zedGraphControl14.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(222, 224, 212));
zedGraphControl15.GraphPane.Fill = new Fill(Color.FromArgb(222,
224, 212));
zedGraphControl15.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(222, 224, 212));
zedGraphControl16.GraphPane.Fill = new Fill(Color.FromArgb(222,
224, 212));
zedGraphControl16.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(222, 224, 212));
zedGraphControl17.GraphPane.Fill = new Fill(Color.FromArgb(222,
224, 212));
zedGraphControl17.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(222, 224, 212));
zedGraphControl18.GraphPane.Fill = new Fill(Color.FromArgb(222,
224, 212));
zedGraphControl18.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(222, 224, 212));
zedGraphControl19.GraphPane.Fill = new Fill(Color.FromArgb(222,
224, 212));
zedGraphControl19.GraphPane.Chart.Fill = new Fill(Color.FromArg
b(222, 224, 212));
zedGraphControl10.GraphPane.Fill = new Fill(Color.FromArgb(222
, 224, 212));
zedGraphControl10.GraphPane.Chart.Fill = new Fill(Color.FromAr
gb(222, 224, 212));
zedGraphControl11.GraphPane.Fill = new Fill(Color.FromArgb(222
, 224, 212));
zedGraphControl11.GraphPane.Chart.Fill = new Fill(Color.FromAr
gb(222, 224, 212));
zedGraphControl12.GraphPane.Fill = new Fill(Color.FromArgb(222
, 224, 212));
zedGraphControl12.GraphPane.Chart.Fill = new Fill(Color.FromAr
gb(222, 224, 212));
lblsensor1.BackColor = Color.FromArgb(222, 224, 212);
```

```

        lblsensor2.BackColor = Color.FromArgb(222, 224, 212);
        lblsensor3.BackColor = Color.FromArgb(222, 224, 212);
        lblsensor4.BackColor = Color.FromArgb(222, 224, 212);
        lblsensor5.BackColor = Color.FromArgb(222, 224, 212);
        lblsensor6.BackColor = Color.FromArgb(222, 224, 212);
        lblsensor7.BackColor = Color.FromArgb(222, 224, 212);
        lblsensor8.BackColor = Color.FromArgb(222, 224, 212);
        lblsensor9.BackColor = Color.FromArgb(222, 224, 212);
        lblsensor10.BackColor = Color.FromArgb(222, 224, 212);
        lblsensor11.BackColor = Color.FromArgb(222, 224, 212);
        lblsensor12.BackColor = Color.FromArgb(222, 224, 212);
        lblsensor13.ForeColor = Color.White;
        lblsensor14.ForeColor = Color.White;
        lblsensor15.ForeColor = Color.White;
        lblsensor16.ForeColor = Color.White;
        lblsensor17.ForeColor = Color.White;
        lblsensor18.ForeColor = Color.White;
        lblsensor19.ForeColor = Color.White;
        drum = true;
    }
}

private void MetroTabPage1_Click(object sender, EventArgs e)
{

}

private void MetroTabPage2_Click(object sender, EventArgs e)
{

}

private void HtmlPanel1_Click(object sender, EventArgs e)
{

}

private void MetroLabel17_Click(object sender, EventArgs e)
{

}

private void CbDatabits_SelectedIndexChanged(object sender, EventArgs
e)
{

}

private void BtnConn_Click_1(object sender, EventArgs e)

```

```

{
    try
    {
        if (cbPorts.Text != "")
        {
            if (cbBaud.Text != "")
            {
                serialPort1.PortName = cbPorts.Text;
                serialPort1.BaudRate = Convert.ToInt32(cbBaud.Text);
                serialPort1.Parity = (Parity)Enum.Parse(typeof(Parity)
, cbParity.Text);

                serialPort1.StopBits = (StopBits)Enum.Parse(typeof(Sto
pBits), cbStopbits.Text);
                serialPort1.DataBits = Convert.ToInt32(cbDatabits.Text
);

                serialPort1.Handshake = Handshake.None;
                serialPort1.RtsEnable = true;
                serialPort1.DataReceived += new SerialDataReceivedEven
tHandler(serialPort1_DataReceived);
                if (serialPort1.IsOpen) return;
                serialPort1.Open();
                btnConn.Enabled = false;
                btnDisConn.Enabled = true;
                //
                cbBaud.Enabled = false;
                cbPorts.Enabled = false;
                cbDatabits.Enabled = false;
                cbStopbits.Enabled = false;
                cbParity.Enabled = false;
                if (datalogger_checkbox.Checked)
                    try { out_file.Dispose(); }
                    catch { /*ignore*/ }
            }
            else
                return;
        }
        else
            return;
    }
    catch
    {
        return;
    }
}

private void BtnDisConn_Click_1(object sender, EventArgs e)
{
    try
    {

```

```

        if (serialPort1.IsOpen == false) return;
        serialPort1.Close();
        btnConn.Enabled = true;
        btnDisConn.Enabled = false;
        //
        cbBaud.Enabled = true;
        cbPorts.Enabled = true;
        cbDatabits.Enabled = true;
        cbStopbits.Enabled = true;
        cbParity.Enabled = true;
    }
    catch
    {
        return;
    }
}

private void Datalogger_checkbox_CheckedChanged_1(object sender, EventArgs e)
{
    if (datalogger_checkbox.Checked)
    {
        if (openFileDialog1.ShowDialog() == DialogResult.OK)
        {
            datalogger_checkbox.Text = openFileDialog1.FileName;
            string text = "Time,H1,W1,H2,W2,H3,W3,H4,W4,H5,W5,H6,W6,T0
,T1,T2,T3,T4,T5,T6";
            System.IO.File.WriteAllText(@datalogger_checkbox.Text, text);
        }
        else
        {
            datalogger_checkbox.Checked = false;
        }
    }
    else
    {
        datalogger_checkbox.Text = "Enable Data logger";
    }
}

private void Lblsensor3_Click(object sender, EventArgs e)
{
}

private void Timer16_Tick_1(object sender, EventArgs e)
{
    string[] ports = SerialPort.GetPortNames();

```

```
        if (intlen != ports.Length)
        {
            intlen = ports.Length;
            cbPorts.Items.Clear();
            for (int j = 0; j < intlen; j++)
            {
                cbPorts.Items.Add(ports[j]);
            }
        }
    }
}
```