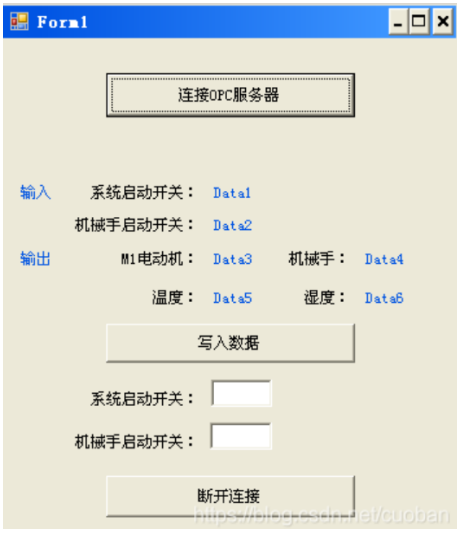
OPC 通讯实例(C#通过 OPC 连接 PLC 读写数据)（目前最解决我问题的文章之一，特此收藏)

https://blog.csdn.net/cuoban/article/details/106130764



using

```
using System.Collections.Generic;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using OPCAutomation;
using System.Diagnostics;
```

Global variable

```
OPCServer objServer;
OPCGroups objGroups;
OPCGroup objGroup;
OPCItems objItems;
Array strItemIDs;
Array lClientHandles;
Array lserverhandles;
Array lErrors;
//  int ltransID_Rd &#61; 1;
// int lCancelID_Rd;
object RequestedDataTypes &#61; null;
object AccessPaths &#61; null;
//   Array lerrors_Rd;
Array lErrors_Wt;
int lTransID_Wt &#61; 2;
int lCancelID_Wt;
```

Button1 按鈕: 连接 opc server

```
private void button1_Click(object sender, EventArgs e)
{
     //(1)创建opc server对象
    objServer &#61; new OPCServer();
    //连接opc server
    objServer.Connect(&#34;KEPware.KEPServerEx.V4&#34;, null);
    //(2)建立一个opc组集合
    objGroups &#61; objServer.OPCGroups;
    //(3)建立一个opc组
    objGroup &#61; objGroups.Add(null); //Group组名字可有可无
    //(4)添加opc标签
    objGroup.IsActive &#61; true; //设置该组为活动状态&#xff0c;连接PLC时&#xff0c;设置为非活动状态也一样
    objGroup.IsSubscribed &#61; true; //设置异步通知
    objGroup.UpdateRate &#61; 250;
    objServer.OPCGroups.DefaultGroupDeadband &#61; 0;
    objGroup.DataChange  &#61; new DIOPCGroupEvent_DataChangeEventHandler(KepGroup_DataChange);
    // objGroup.AsyncReadComplete  &#61; new DIOPCGroupEvent_AsyncReadCompleteEventHandler(AsyncReadComplete);
    objGroup.AsyncWriteComplete  &#61; new DIOPCGroupEvent_AsyncWriteCompleteEventHandler(AsyncWriteComplete);
    objItems &#61; objGroup.OPCItems; //建立opc标签集合
    string[] tmpIDs &#61; new string[7];
    int[] tmpCHandles &#61; new int[7];
    for (int i &#61; 1; i &lt; 7; i  )
    {
        tmpCHandles[i] &#61; i;
    }
    tmpIDs[1] &#61; &#34;西门子S7-300.PLC.系统启动开关&#34;;
    tmpIDs[2] &#61; &#34;西门子S7-300.PLC.机械手启动开关&#34;;
    tmpIDs[3] &#61; &#34;西门子S7-300.PLC.M1电动机&#34;;
    tmpIDs[4] &#61; &#34;西门子S7-300.PLC.机械手&#34;;
    tmpIDs[5] &#61; &#34;西门子S7-300.PLC.温度&#34;;
    tmpIDs[6] &#61; &#34;西门子S7-300.PLC.湿度&#34;;
    strItemIDs &#61; (Array)tmpIDs;//必须转成Array型&#xff0c;否则不能调用AddItems方法
    lClientHandles &#61; (Array)tmpCHandles;
    // 添加opc标签
    objItems.AddItems(6, ref strItemIDs, ref lClientHandles, out lserverhandles, out lErrors, RequestedDataTypes,
AccessPaths);
}
```

Button4 按鈕: 结束并断开 opc server

```
private void button4_Click(object sender, EventArgs e)
{
    objServer.Disconnect();
    //关闭kepserver进程&#xff0c;这个跟OPC操作无关
    /*
    foreach ( Process oneProcess in Process.GetProcesses())
    {
    if (oneProcess.ProcessName &#61;&#61; &#34;ServerMain&#34;)
    oneProcess.Kill();
    }
    */
}
```

Button3 按鈕: 发送异步写数据指令 (寫入數據)

```
private void button3_Click(object sender, EventArgs e)
{
    Array AsyncValue_Wt;
    Array SerHandles;
    object[] tmpWtData &#61; new object[3];//写入的数据必须是object型的&#xff0c;否则会报错
    int[] tmpSerHdles &#61; new int[3];
    //将输入数据赋给数组&#xff0c;然后再转成Array型送给AsyncValue_Wt
```

```
    tmpWtData[1] &#61; (object)textBox1.Text;
    tmpWtData[2] &#61; (object)textBox2.Text;
    AsyncValue_Wt &#61; (Array)tmpWtData;
    //将输入数据送给的Item对应服务器句柄赋给数组&#xff0c;然后再转成Array型送给SerHandles
    tmpSerHdles[1] &#61; Convert.ToInt32(lserverhandles.GetValue(1));
    tmpSerHdles[2] &#61; Convert.ToInt32(lserverhandles.GetValue(2));
    SerHandles &#61; (Array)tmpSerHdles;
    objGroup.AsyncWrite(2, ref SerHandles, ref AsyncValue_Wt, out lErrors_Wt, lTransID_Wt, out
lCancelID_Wt);

}

//异步写入成功
private void AsyncWriteComplete(int TransactionID, int NumItems, ref Array ClientHandles, ref Array
Errors)
{
    MessageBox.Show(&#34;数据写入成功&#xff01;&#34;);
}
```

//每当项数据有变化时执行的事件,采用订阅方式

```
void KepGroup_DataChange(int TransactionID, int NumItems, ref Array ClientHandles, ref Array
ItemValues, ref Array Qualities, ref Array TimeStamps)
{
    //为了测试&#xff0c;所以加了控制台的输出&#xff0c;来查看事物ID号
    //Console.WriteLine(&#34;********&#34; TransactionID.ToString() &#34;*********&#34;);
    for (int i &#61; 0; i &lt; NumItems; i  )
    {
        if (Convert.ToInt32(ClientHandles.GetValue(i   1)) &#61;&#61; 1)
        {
            if (ItemValues.GetValue(i   1) !&#61; null)
            {
                this.Data1.Text &#61; ItemValues.GetValue(i   1).ToString();
            }
        }
        if (Convert.ToInt32(ClientHandles.GetValue(i   1)) &#61;&#61; 2)
        {
            if (ItemValues.GetValue(i   1) !&#61; null)
            {
                this.Data2.Text &#61; ItemValues.GetValue(i   1).ToString();
            }
        }
        if (Convert.ToInt32(ClientHandles.GetValue(i   1)) &#61;&#61; 3)
        {
            if (ItemValues.GetValue(i   1) !&#61; null)
            {
                this.Data3.Text &#61; ItemValues.GetValue(i   1).ToString();
            }
        }
        if (Convert.ToInt32(ClientHandles.GetValue(i   1)) &#61;&#61; 4)
        {
            if (ItemValues.GetValue(i   1) !&#61; null)
            {
                this.Data4.Text &#61; ItemValues.GetValue(i   1).ToString();
            }
        }
        if (Convert.ToInt32(ClientHandles.GetValue(i   1)) &#61;&#61; 5)
        {
            if (ItemValues.GetValue(i   1) !&#61; null)
            {
                this.Data5.Text &#61; ItemValues.GetValue(i   1).ToString();
            }
```

```
        }
        if (Convert.ToInt32(ClientHandles.GetValue(i   1)) &#61;&#61; 6)
        {
            if (ItemValues.GetValue(i   1) !&#61; null)
            {
                this.Data6.Text &#61; ItemValues.GetValue(i   1).ToString();
            }
        }
    }
}
```