SIEMENS西门子广西省百色市（授权）电机一级代理商——西门子华南总代理

产品名称	SIEMENS西门子广西省百色市（授权）电机一级代理商——西门子华南总代理
公司名称	广东湘恒智能科技有限公司
价格	.00/件
规格参数	西门子总代理:PLC 西门子一级代:驱动 西门子代理商:伺服电机
公司地址	惠州大亚湾澳头石化大道中480号太东天地花园2栋二单元9层01号房
联系电话	15915421161 15903418770

产品详情

项目介绍

本项目是一个物料输送系统。控制系统核心是两台S7-1200系列PLC。人机界面是一台触摸式的工控一体机。点数大概有2000点左右。

本项目我们将以重构后的C#上位机项目实例作为模板进行修改，以一个实际的上位机项目开发过程来检验它的开发效率。

开发过程-驱动部分

作为一个上位机，最基础也是最重要的部分就是通信驱动了。在我们的C#上位机项目模板中，设备通信是作为后台独立运行的。前端页面通过变量名称和PLC进行数据交互。通信过程是封装好的，且开放的。我们在开发时不需要关注通信细节，只需要根据物理PLC数据进行实例化，然后创建变量即可。

具体修改过程非常简单，首先我们根据实际的物理PLC数量进行声明和初始化。

```
//根据物理PLC数量声明变量，该项目一共2台PLCstatic S7PLC PLC1 = null;static S7PLC PLC2 =
null;//////初始化PLC///public static void Initial(){   //往变量池中添加变量   AddTagsPLC1()
;   AddTagsPLC2();   //初始化1#PLC   PLC1 = newS7PLC("192.168.0.11",0,1);
//订阅事件   PLC1.StatusChanged += PLC1_StatusChanged;   PLC1.DataUpdate += PL
C1_DataUpdate;   //读写信息   ByteList1(PLC1);
     PLC1.WriteTagsList = TagsPLC1;   //通信启动   PLC1.Start();   //初始化
2#PLC   PLC2 = newS7PLC("192.168.0.12", 0, 1);   //订阅事件   PLC2.StatusChan
```

ged += PLC2_StatusChanged; PLC2.DataUpdate += PLC2_DataUpdate; //读写信息
 ByteList2(PLC2); PLC2.WriteTagsList = TagsPLC2; //通信启动 PLC2.Start
();}然后我们根据每台PLC需要访问的数据往PLC实例里面添加变量。

/// /// 添加变量/1#PLC/// private static void AddTagsPLC1(){ TagsPLC1.Add("A区注塑机
料位1", new TagModel() { Address = "D1.0.0" }); TagsPLC1.Add("A区注塑机料位2", ne
w TagModel() { Address = "D1.0.1" }); TagsPLC1.Add("A区注塑机料位3", new TagMod
el() { Address = "D1.0.2" }); TagsPLC1.Add("A区注塑机料位4", new TagModel() { Ad
dress = "D1.0.3" }); TagsPLC1.Add("A区注塑机料位5", new TagModel() { Address = "
D1.0.4" }); TagsPLC1.Add("A区注塑机料位6", new TagModel() { Address = "D1.0.5" });
  TagsPLC1.Add("A区注塑机料位7", new TagModel() { Address = "D1.0.6" }); Ta
gsPLC1.Add("A区注塑机料位8", new TagModel() { Address = "D1.0.7" }); //……
//变量比较多，这里就不全面贴出来了。}

同样的方法为2#PLC添加变量。再根据变量情况设置需要请求的报文信息。

private static void ByteList2(S7PLC mdb) { ReadMsg msg = new ReadMsg();
 msg.Id = "1"; msg.RgstArea = 0x84; msg.DBNumber = 1; msg.
StartAddress = 0; msg.Count = 9000; mdb.ReadMsgList.Add(msg); }

最后一步是更新数据，我们需要将驱动返回的报文解析并绑定到相应的变量。数据更新方法由PLC驱动的数据更新事件进行触发。

//////解析数据//////private static void AnalyzeDataPLC1(Dictionary Data){ Byte[] DataArray = Array.ConvertAll(Data["1"], val => checked((Byte)val)); TagsPLC1["A区注塑机料位1"].Value1 = Sharp7.S7.GetBitAt(DataArray, 0, 0); TagsPLC1["A区注塑机料位2"].Value1 = Sharp7.S7.GetBitAt(DataArray, 0, 1); TagsPLC1["A区注塑机料位3"].Value1 = Sharp7.S7.GetBitAt(DataArray, 0, 2); TagsPLC1["A区注塑机料位4"].Value1 = Sharp7.S7.GetBitAt(DataArray, 0, 3); TagsPLC1["A区注塑机料位5"].Value1 = Sharp7.S7.GetBitAt(DataArray, 0, 4); TagsPLC1["A区注塑机料位6"].Value1 = Sharp7.S7.GetBitAt(DataArray, 0, 5); TagsPLC1["A区注塑机料位7"].Value1 = Sharp7.S7.GetBitAt(DataArray, 0, 6); TagsPLC1["A区注塑机料位7"].Value1 = Sharp7.S7.GetBitAt(DataArray, 0, 7); //…… //变量比较多，这里就不全面贴出来了。 }

只需要上述简单的几步即可完成上位机软件和设备的数据交互。根据实测时间，一天不到的时间就完成了两台PLC共约2000点的变量配置。除了支持批量导入PLC变量的WinCC外，对于一些不支持变量导入的组态软件，在配置这么多变量的时间花费上估计也是相差无几。如果采用我们的支持变量自动解析的AdvScadaFrameworkgaoji上位机框架软件则可以在更短的时间内完成。

变量添加完成后，我们在需要展示的窗体定时刷新就可以了。这种方式比较简单方便。当然也可以采用基于变量名的绑定方式。这两种方式的工作量其实差不多。