目录

第一章简介

1.1 简介

Zcm968 主要提供了 IO 控件、轴控件、参数控件等控件及通讯和运动相关方法

1.2 功能介绍

& IO 控件

& 轴控件

& 参数控件

& 其他控件

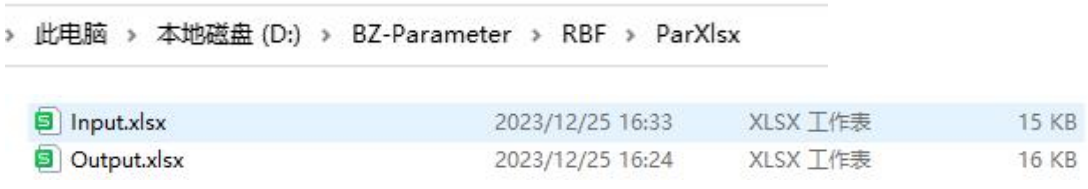
& 通讯相关

& 运动相关

第二章 控件说明

2.1 IO 控件

使用 IO 相关控件需要注意在 D 盘下 BZ-Parameter 文件夹下具体机型的 ParXlsx 文件夹下的 Input.xlsx 和 Output.xlsx



Input.xlsx 文件

output.xlsx 文件

配置 IO 相关控件时需注意左边编号与控件上编号一致

```
#Region "I/O 枚举"
    99+ references
    Public Enum InNo
        急停按钮信号
        启动按钮信号
        停止按钮信号
        复位按钮信号
        手自动切换
        总正气压信号
        左前安全门信号
        右前安全门信号
        左后安全门信号
        右后安全门信号
        备用10
        主流线_下游要料信号
        主流线_上游有料信号
        主流线_上游生料信号
        主流线_上游熟料OK信号
        主流线_上游熟料NG信号
        右机械手准备好
        右机械手运动中
        右机械手暂停中
        右机械手报警中
        备用20
        备用21
        备用22
        备用23
```

输入枚举

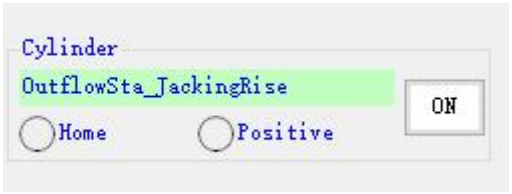
```
Public Enum OutNo
    三色灯_红色
    三色灯_黄色
    三色灯_绿色
    三色灯_蜂鸣
    启动按钮指示灯
    停止按钮指示灯
    复位按钮指示灯
    前安全门锁控制
    后安全门锁控制
    日光灯
    备用10
    主流线_给上游要料信号
    主流线_给下游有料信号
    主流线_给下游生料信号
    主流线_给下游熟料OK信号
    主流线_给下游熟料NG信号
    右机械手启动
    右机械手停止
    右机械手暂停
    右机械手继续
    右机械手复位
    右机械手压力到达
    备用22
    右废料盒吸气
    左机械手启动
    左机械手停止
    左机械手暂停
```

输出枚举

枚举索引应与文件编号一致
具体讲解如下

2.1.1 Zcm.Cylinder

使用 Cylinder 控件更方便控制双控的气缸控件会对两个 IO 输出同时操作（关断一个并开启另一个），并可以根据 IO 输入状态判断气缸状态



需要在控件的属性页面配置 mInput1Index1，mInput2Index1，mOutIndex1。mInput1Index1，mInput2Index1 为对应 IO 输入的编号，mOutIndex1 为对应输出的编号，气缸双控需配置两个 IO，点击控件上的按钮会对应输出编号开启其中一个关断另外一个，也可查看对应输入查看气缸是否正常运动

配置完成后点击控件上的按钮，判断配置是否正确

2.1.2 Zcm.mBtn_Out

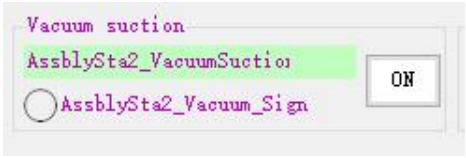
mBtn_Out 控件可以对单个 IO 输出进行使能和失能

需在控件的属性页面配置 mOutIndex1，mOutIndex1 为 Io 输出对应的编号，根据需要也可以改变属性中的 PlusWidth1 配置输出脉冲宽度，将输出方式从电平输出改为脉冲输出

配置完成后点击控件上的按钮，判断输出是否正确

2.1.3 Zcm.Vacuum

使用 Vacuum 控件可以配置设备真空吸并查看真空状态

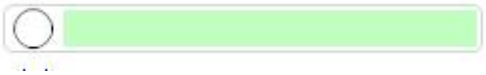


需在控件的属性页面配置 Input1 和 Output1，Onput1 为触发真空吸 IO 对应的编号，Input1 为是否检测到真空信号对应的 IO 输入的编号

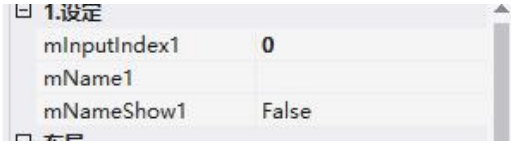
配置完成后点击控件上的按钮，判断是否触发真空吸，堵住气孔后判断是否有真空信号

2.1.4 Zcm.mShp_Input

使用 mShp_Input 可以监测单个 IO 输入状态



需在控件的属性页面配置好 mInputIndex1，mInputIndex1 为对应的 IO 输入的编号

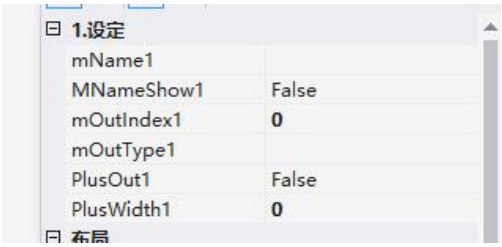


配置完成即可使用

2.1.5 Zcm.mButton

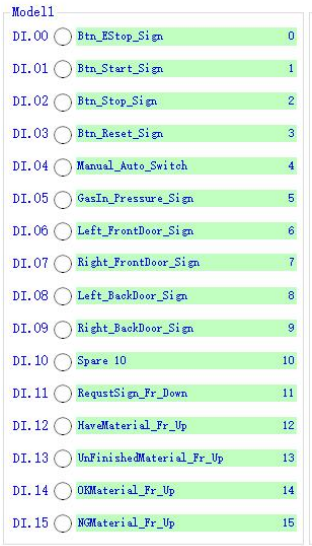
使用 mButton 控件快速使能和失能单个 IO 输出

需在控件的属性页面配置 mOutIndex1，mOutIndex1 为 Io 输出对应的编号，根据需要也可以改变属性中的 PlusWidth1 配置输出脉冲宽度，将输出方式从电平输出改为脉冲输出



2.1.6 Zcm.InputClass

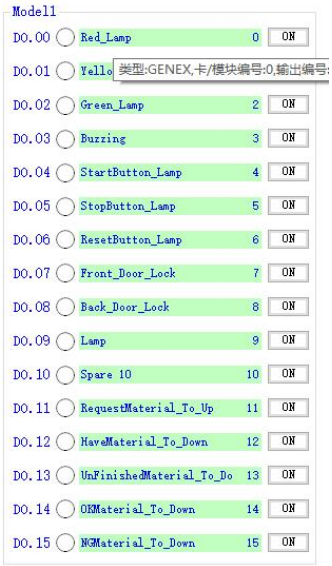
使用 Zcm.InputClass 可以一次性显示多个 IO 输入的状态



需要配置 mCardIndex1 和 mCtlIndex1，mCardIndex1 为对应的卡编号，mCtlIndex1 为对应的模块编号，一个控件显示 16 个 IO 状态，根据卡编号和模块编号自动索引到相关的 IO 项

2.1.7 Zcm.OutputClass

使用 Zcm.OutputClass 可以一次性显示多个 IO 输出的状态并提供使能和失能的方法



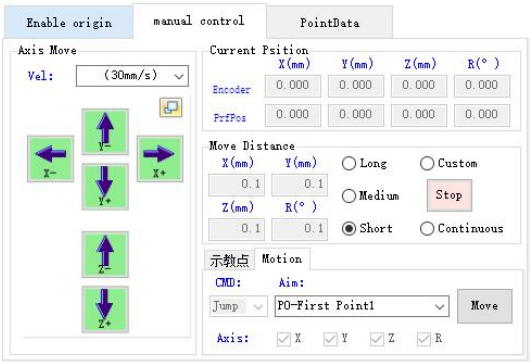
需要在属性页面配置 mCardIndex1 和 mCtlIndex1,mCtlIndex1 为对应的模块编号，一个控件显示 16 个 IO 状态，根据卡编号和模块编号自动索引到相关的 IO 项

配置完成后点击按钮验证配置是否正常

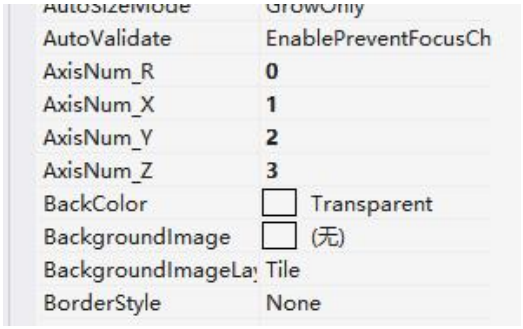
2.2 轴相关控件

2.2.1 Zcm.Teach

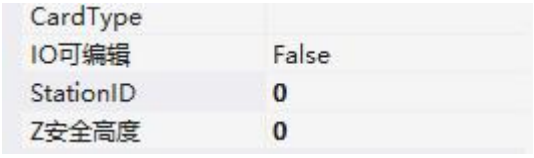
使用 Teach 控件可以更便利的操作运动轴



在控件的配置页面配置 AxisNum_X、AxisNum_Y、AxisNum_Z 和 AxisNum_R 没有对应的轴则配置为 0，有对应的值则配置对应的轴编号



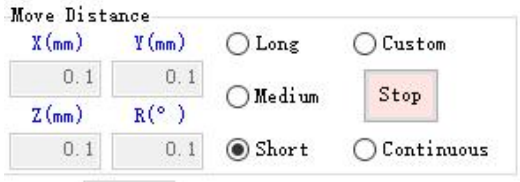
还可以通过改变属性的 Z 安全高度没模组 Z 轴设置安全高度



在 Enable origin 页面可以也可以查看各个轴的状态，也可以对每个轴进行单独的使能或回原点

使能且回原点后可以在 manual control 页面自由移动模组

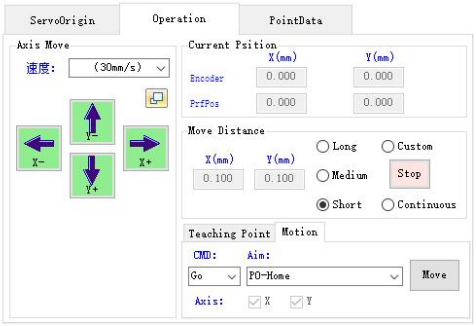
自由移动方式分为长距（Long），中距离（Medium）和短距离（Short），也可在 TextBox 自定义(Custom)设置移动距离，选择连续(Continuous)则可以长按持续移动，还可以通过 stop 打断动作



在 PointData 页面可以添加对点位进行增删改查

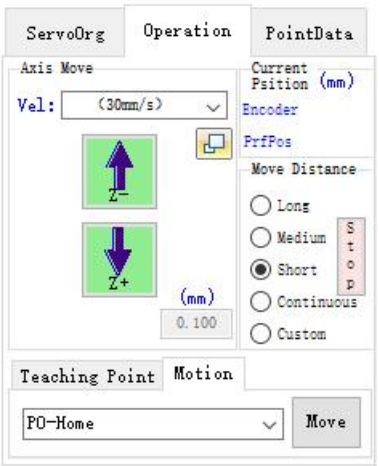
在 manual control 页面切到执行运动页面，通过下拉菜单选择对应的点位，点击运动并确认后即可使模组移动到对应的点位

2.2.2 Zcm.TeachXY



具体使用方法与 Teach 类比 Zcm.Teach 控件，适用于只有两个轴的模组，较为节约空间

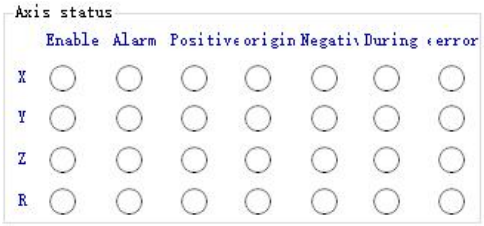
2.2.3 Zcm.TeachS



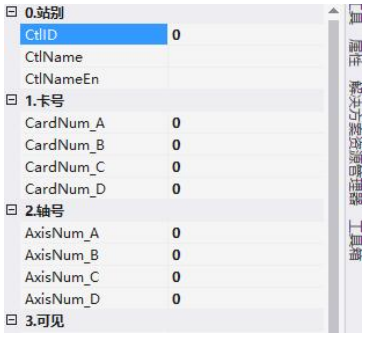
具体使用方法与 Teach 类比 Zcm.Teach 控件，适用于只有 1 个轴的模组，较为节约空间

2.2.4 Zcm.motorStatus

使用 Zcm.motorStatus 控件监控多个轴的状态

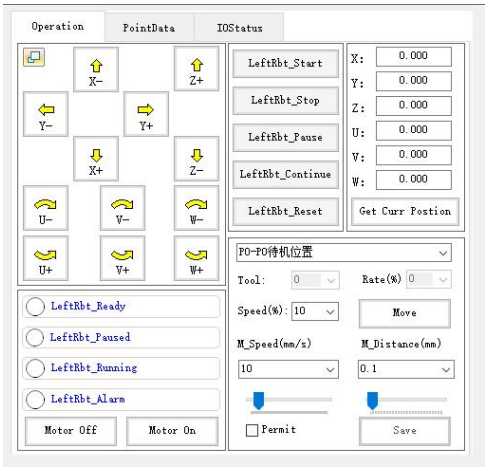


需要在控件的属性页面配置卡号还有轴号，配置完成后可以在此页面监控各个轴的状态



2.3 Robot 相关控件

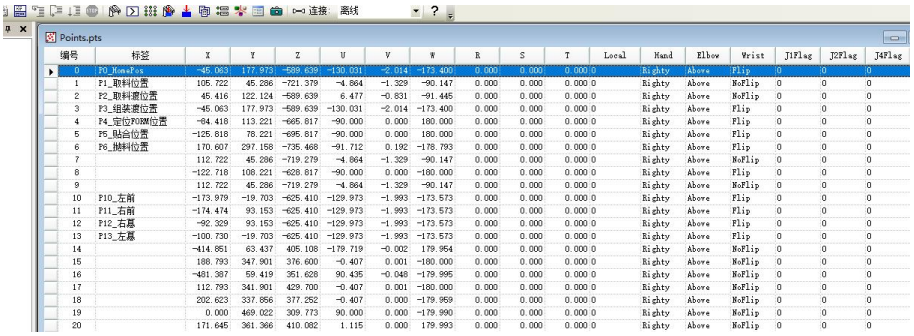
2.3.1 Zcm.Robot



在控件的属性页面需配置好 Robot 的输入设定相关 IO，以便监测机械手运行状态；
输入设定配置完成后还需配置机械手的输出设定，通过对应的 IO 输出可以控制机械手的启停复位相关；
配置属性中的点位数量可以控制控件中点位数据页面显示的数据条数；
在换位相关的属性中可以交换轴政府移动按钮的位置；
最后还需选择属性中的 RobotSelect 确定机械手四六轴选择；
配置完成后启动机械手即可在手动控制页面控制机械手的移动。

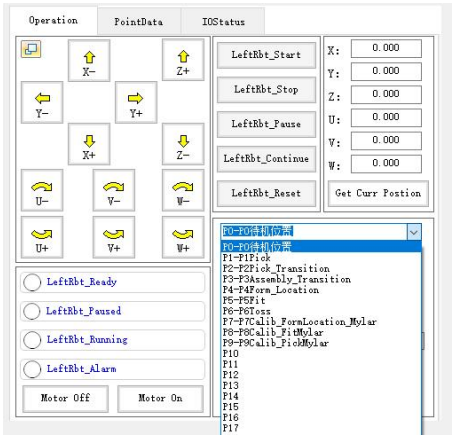
在控件的 PointData 界面可以查看机械手中存取的点位并对点位进行增删改查

此处点位和机械手处的点位文件对应

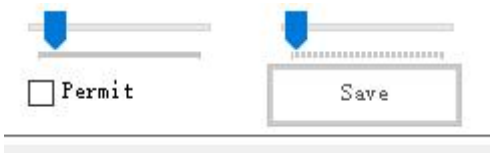


在 IOStatus 页面可以查看 Robot 的各种状态，也可以对机械手进行启停复位等控制

下拉点位菜单项可以选择对应的点位，选择后点击运动按钮并确认弹出的弹窗后，Robot 便会移动到对应的点位



在 Operation 页面可以通过手动控制将机械手移动到需要的位置，将点位菜单选择到需要保存的点位项，勾选 Permit 的 CheckBox 点击 Save 按钮，点击确认后将当前点位保存到对应的点位中，如果保存的点位与之前的点位偏差过大则需要输入密码才可以保存（密码一般为 1234）



在程序中使用需要在对应工站的 Initialize 方法中初始化一些机械手相关配置，为机械手 mtnClick 添加防呆方法，TaskID 为工站任务编号 TcpName.RobotA 为机械手对应的 TCPIP 控件编号，1 为机械手控件编号"robot1"为机械手名称

```
private EpsonRobot robot1;
#endregion 定义
#region Override
2 个引用
public override void Initialize()
{
    TaskID = 1;  //唯一编号，不同工位不可重复
    WkManager.BindStation(TaskID, "机械手工站", this);
    mSend.mAction += Err;
    OnGetDiAction += Err;
    InitPos();
    //TaskDeviceInit();
    Frm_Engineering.Instance.robot1.mBtnClick += SafeCheckRobotManual;  //机械手外部事件防呆
    robot1 = new EpsonRobot(mTaskId.Task2Robot, mTcpIpId.robot1);       //实例化机械手构造函数
    RobotManager.Instance.BindRobot(robot1, 1, "robot1");               //绑定机械手参数，加入字典
}
```

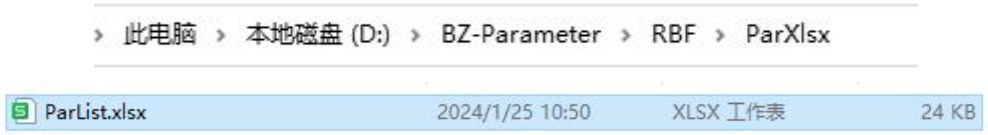
机械手防呆会方法在手动走点位时触发，在进入方法时将 mFunction.mStaData.StepStop 置为 false，当感应到异常时将 mFunction.mStaData.StepStop 置为 true，机械手便会停止当前动作并跳出动作以拦截弹窗

```
/// <summary> 防呆专用-Robot
2 个引用
public void SafeCheckRobotManual()
{
    mFunction.mStaData.StepStop = false;
    int px = Convert.ToInt16(Frm_Engineering.Instance.RobotA.StaData.Px);
    string PointiIndex= Frm_Engineering.Instance.RobotA.StaData.PointIndex;
    //前面加一些安全信号，防呆
    string cmd = Frm_Engineering.Instance.RobotA.StaData.CMD;

    switch (cmd)
    {
        case "PalletRun":
        case "RobotRun":
            if (px == 5)
            {
                if (PointiIndex=="0")
                {
                    MessageBox.Show("矩阵点位不存在0点!!!");
                    mFunction.mStaData.StepStop = true;//不安全就停
                }
            }
            break;
        default:
            break;
    }
    //mFunction.mStaData.StepStop = true;//不安全就停
}
```

2.4 参数相关控件

使用 IO 相关控件需要注意在 D 盘下 BZ-Parameter 文件夹下具体机型的 ParXlsx 文件夹下的 ParList.xlsx



	A	B	C	D	E	F	G	H	I	J	K	L
1	序号	参数名称Cn	参数名称En	分类	上限	下限	参数单位	序号	功能名称Cn	功能名称En	分类	
2	0	Robot速度	Robot speed	Robot	100	1		0			D	
3	1	Robot速度百分比	Robot speed percentage	Robot	100	1		1			D	
4	2	Rbt_贴合压力	Rbt_FitPressure	Pressure	10	0	kgf	2			D	
5	3	Rbt_贴合压力上限	Rbt_FittingPressureUpperLimit	Pressure	10	0	kgf	3			D	
6	4	Rbt_贴合压力下限	Rbt_FittingPressureLowerLimit	Pressure	10	0	kgf	4			D	
7	5	Rbt_贴合延时	Rbt_AdhesiveDelay	Delay	10	0	S	5			D	
8	6	Rbt_取料延时	Rbt_MaterialRetrievalDelay	Delay	10	0	S	6			D	
9	7	拍照前延时	Delay before taking photos		10	0		7			D	
10	8	单个取料拍照失败次数	SingleReclaiming_PhotographFailNum	Count	5	1		8			I	
11	9	连续取料拍照失败次数	ContinuousReclaiming_PhotographFailNum	Count	5	1		9			I	
12	10	单个贴合拍照失败次数	SinglePaste_PhotographFailNum	Count	5	1		10			I	
13	11	连续贴合拍照失败次数	ContinuousPaste_PhotographFailNum	Count	5	1		11			I	
14	12	单个取料真空吸异常次数	SingleReclaiming_VacuumAbnormalNum	Count	5	1		12			I	
15	13	连续取料真空吸异常次数	ContinuousReclaiming_VacuumAbnormalNum	Count	5	1		13			I	
16	14	备用14	Spare 14		5	0		14			I	
17	15	备用15	Spare 15		5	0		15			I	
18	16	扫码次数	Scanning frequency	Scan	6	0		16			I	
19	17	条码长度上限	Upper limit of barcode length	Scan	50	0		17			I	
20	18	条码长度下限	Lower limit of barcode length	Scan	50	0		18			I	
21	19	扫码连续失败次数	ConsecutiveScanCode_FailNum	Scan	5	0		19			I	
22	20	备用20	Spare 20		5	0		20			I	
23												
24	110				1000	-1000		110	启用PDCA传图	PDCA img transmission	B	
25	111				1000	-1000		111	启用HIVE	Enable HIVE	B	
26	112				1000	-1000		112	启用MES	Enable MES	B	
27	113				1000	-1000		113	启用路由检查	Enable routing check	B	
28	114				1000	-1000		114	启用VISION功能	Enable VISION	B	
29	115				1000	-1000		115	启用GROUP存图	Spare 15	B	
30	116				1000	-1000		116	屏蔽左机械手	Shield left robot	B	
31	117				1000	-1000		117	屏蔽右机械手	Shield right robot	B	
32	118				1000	-1000		118	启用提前取料	Advance material retrieval	B	
33	119				1000	-1000		119	CPK模式获取本地WIP	CPK obtains local WIP	B	
34	120				1000	-1000		120	扫码异常弹框	Scan Exception Pop Up	B	
35	121				1000	-1000		121	UOP异常弹框	UOP abnormal pop-up	B	
36	122				1000	-1000		122	启用压力绘制	Pressure drawing	B	
37	123				1000	-1000		123	启用旋转吹料气缸	Rotary blowing cylinder	B	

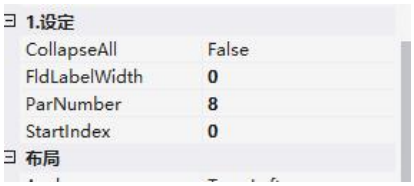
ParList.xlsx 文件内容

配置参数相关控件时需注意左边序号与控件上编号一致
具体讲解如下

2.4.1 Zcm.PropertyChk

使用 Zcm.PropertyChk 控件可以对多个参数进行查改

需要配置控件的 StartIndex 项和 ParNumber 两项 StartIndex 为参数列表的起始索引，ParNumber 为显示的参数量



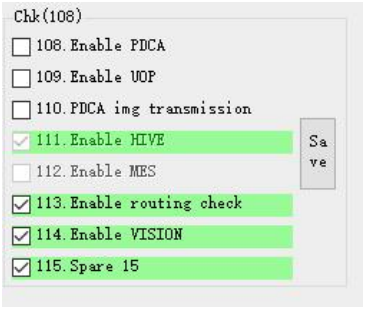
配置完成后会根据参数列表自己在对应的位置读取参数项，在控件上修改参数后点击 Save 按钮即可将修改后的数据进行保存

2.4.2 Zcm.PropertyChkS

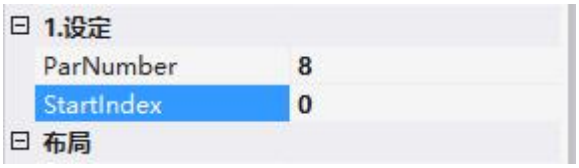
Zcm.PropertyChkS 的使用与 Zcm.PropertyChk 使用方法相同

2.4.3 使用 Zcm.UserChk

使用 Zcm.UserChk 控件可以对多个 bool 类型参数进行查改



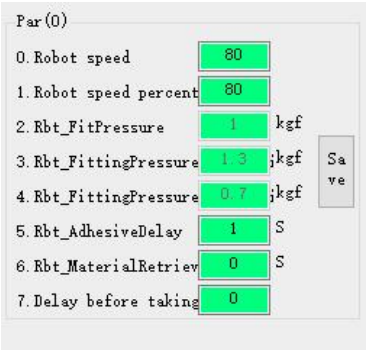
需要配置控件的 StartIndex 项和 ParNumber 两项 StartIndex 为参数列表的起始索引，ParNumber 为显示的参数量



配置完成后会根据参数列表自己在对应的位置读取参数项

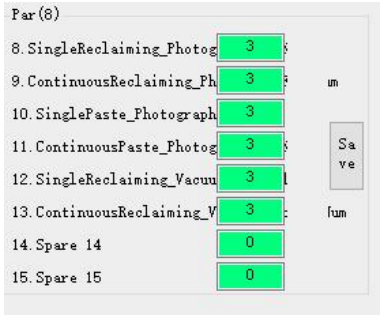
2.4.5 Zcm.UserPar

使用 Zcm.UserPar 控件可以对多个参数进行查改

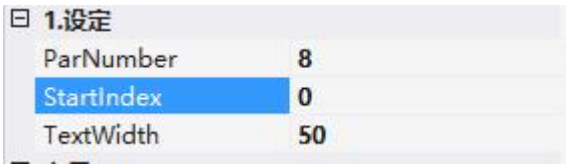


配置完成后会根据参数列表自己在对应的位置读取参数项，在控件上修改参数后点击 save 按钮即可将修改后的数据进行保存

2.4.5 Zcm.UserParS

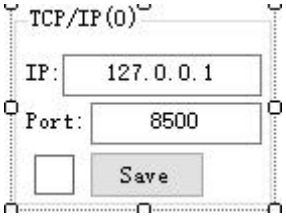


Zcm.UserParS 的使用与 Zcm.UserPar 使用方法相同，增加了 TextWidth 可以调整字体大小

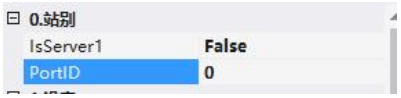


2.4.6 Zcm.Tcp_IP

使用 TCP_IP 控件



需要在控件的属性页面修改属性中的 PortID



PortID 对应如下枚举项

```
public enum mTcpName : short
{
    右Robot组装1 = 1,//192.168.210.100  ; 2000
    左Robot组装2,    //192.168.220.100  ; 2000
    Scanner,         //192.168.0.101~105  ; 904    与两个摄像头，工控机，触摸屏，PLC
    右取料CCD,       //127.0.0.1 ; 8001
    右HSG定位CCD,     //127.0.0.1 ; 8003
    右Form定位CCD,     //127.0.0.1 ; 8002
    左取料CCD,       //127.0.0.1 ; 8004
    左HSG定位CCD,     //127.0.0.1 ; 8006
    左Form定位CCD,     //127.0.0.1 ; 8005

    PDCA = 11,//IP：169.254.1.10  端口号：1111
    UOP = 12,//IP：169.254.1.10  端口号：1111
    外流线PLC = 13,//IP：192.168.101.20:  端口号：2
}
```

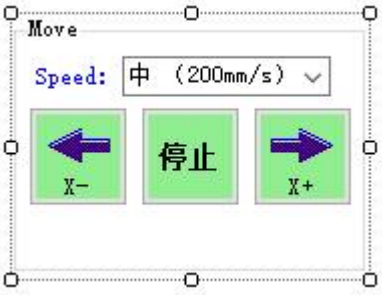
在控件上输好具体的 IP 和 Port 后点击 Save 按钮后对应的 IP 和端口会存储到本地，程序加载时会读入本体存储的文件可通过如下方法访问保存的 IP 和端口

```
string ip = mFunction.TcpIP[(int)mTcpName.外流线PLC].TcpClient.ServerIp;
int port = mFunction.TcpIP[(int)mTcpName.外流线PLC].TcpClient.ServerPort;
```

2.5 其他控件

2.5.1 Zcm.Converoy

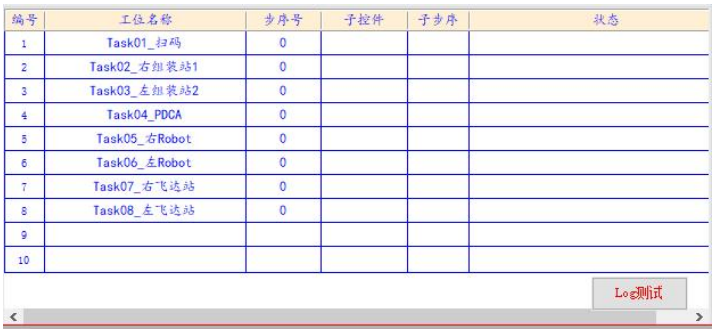
使用 Zcm.Converoy



需在属性页面配置 StationID，StationID 为对应的流水线编号，配置完成后通过此控件即可控制流线的启停相关

2.5.2 Zcm.Stasts

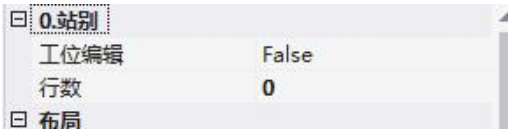
使用 Zcm.Stats 控件



编号	工位名称	步序号	子控件	子步序	状态
1	Task01_扫码	0			
2	Task02_右组装站1	0			
3	Task03_左组装站2	0			
4	Task04_PDCA	0			
5	Task05_右Robot	0			
6	Task06_左Robot	0			
7	Task07_右飞达站	0			
8	Task08_左飞达站	0			
9					
10					

Log测试

在控件属性页面可以对显示行数进行编辑调整需要的显示行数



第三章 类属性及方法说明

3.1 Zcm.DataSend

Zcm.DataSend 提供了 TCPIP 通讯的各类方法

3.1.1 GetData

```
public string GetData
```

功能:通讯接收到的值暂存在此属性中

例:

数据发送后

```
mSend.WaitDone((int)mTcpName.PDCA, 1, BaliTempStr, 0, "", 15000, true, false);//发送模式1为TCPIP,2为RS232
```

接收收到的结果

```
PDCAReceiveStr = mSend.GetData;
```

3.1.2 RunSts

```
public bool RunSts
```

```
mSend.WaitDone((int)mTcpName.PDCA, 1, BaliTempStr, 0, "", 15000, true, false);//发送模式1为TCPIP,2为RS232
```

功能:接收异常是此项会置为 true

例:

数据发送后

```
mSend.WaitDone((int)mTcpName.PDCA, 1, BaliTempStr, 0, "", 15000, true, false);//发送模式1为TCPIP,2为RS232
```

判断接收是否异常

```
if (!mSend.RunSts)//异常就进入里面
{
    _logs.AddLog("<==:数据返回值异常:", PDCAReceiveStr, LogsType._PDCA, PDCAStep, true, false);
    PDCAStep = 1000;
    break;
}
```

3.1.3 WaitDone()

```
public bool WaitDone(ValueType 通讯控件 ID, ValueType 发送模式选择, string 发送的数据, short 数据发送及返回要求, string 被检查数据, int 超时时间, bool 是否清除接收变量, bool 超时弹窗提示重试 = false, short SleepTime = 5)
```

功能:

向指定的 IP 发送收据并等待返回数据

形式参数:

通讯控件:mTcpName 枚举对应的索引

```
public enum mTcpName : short
{
    扫码枪 = 1,          //192.168.200.40 : 9102
    PDCA,//IP: 169.254.1.10  端口号: 1111
    UOP,//IP: 169.254.1.10  端口号: 1111

    左Robot组装2,      //192.168.210.10 : 2000
    左上取料定位CCD,   //192.168.20.30 : 8004
    左下定位CCD,       //192.168.20.30 : 8005
    左HSG定位复检CCD,  //192.168.20.30 : 8006

    右Robot组装1,      //192.168.220.10 : 2000
    右上取料定位CCD,   //192.168.20.30 : 8001
    右下定位CCD,       //192.168.20.30 : 8002
    右HSG定位复检CCD,  //192.168.20.30 : 8003
    PLC,
    MES,
    CARRIER
}
```

发送模式选择:设置发送模式
发送的数据:向对应 IP 发送的数据
数据发送及返回要求:数据发送及返回要求
被检查数据:被检查数据
超时时间:设置超时时间，超时返回 false
是否清除接收变量:true-每次接收清除上一次数据，false-每次接收不清除上一次数据
超时弹窗提示重试:超时后弹窗提醒，可以选择重试
SleepTime:循环读取间隔
返回值:
bool:超时返回 false,没超时返回 true
例：

```
mSend.WaitDone((int)mTcpName.扫码枪, 1, "T\r\n", 0, "", 5000, true, false);
```

3.2 Zcm.Dialog

Zcm.Dialog 提供了阻塞弹窗的实现方法

3.2.1 ReturnData

功能:根据弹窗的按钮后 OKStep 或 NGStep 的值会赋给此属性
例：
弹窗触发后

```
AutoConv.nDialog[mStaNum].DialogShow("流水线" + Convert.ToString(mStaNum) + "阻挡气缸缩回超时，是否继续等待？", "等待", "忽略", 100, 200);
```

判定返回结果

```
if (AutoConv.nDialog[mStaNum].ReturnData != -1)
{
    if (AutoConv.nDialog[mStaNum].ReturnData == 100)
    {
        SetStepC(ref mFunction.流水线[mStaNum], AutoConv.StaStep.阻挡缩回, true);
    }
    else if (AutoConv.nDialog[mStaNum].ReturnData == 200)
    {
        if (mFunction.流水线[mStaNum].允许同步流入)
        {
            SetStepC(ref mFunction.流水线[mStaNum], AutoConv.StaStep.即将流出, true);
        }
        else
        {
            SetStepC(ref mFunction.流水线[mStaNum], AutoConv.StaStep.流出结束, true);
        }
        if (mFunction.流水线[mStaNum].上一段流线编号 != -1)
        {
            AutoConv.启动计时[mFunction.流水线[mStaNum].上一段流线编号] = (long)(GetTickCount());
        }
        ConveyorStart(mStaNum, Convert.ToDouble(mFunction.流水线[mStaNum].流出速度), "+");
    }
}
```

3.2.2 DialogShow()

A.

```
public bool DialogShow(string MsgStr, string ContinuBtnLabel, string StopBtnLabel, int OKStep, int NGStep, int 输出编号 = -1, int 输出时间 = -1)
```

功能:

弹窗并阻塞进程

形式参数:

MsgStr:弹窗显示的文本

ContinuBtnLabel:继续按钮的文本显示

StopBtnLabel:停止按钮的文本显示

OKStep:点击继续按钮返回此数据

NGStep:点击停止按钮返回此数据

返回值:

bool:点击继续按钮 true,点击停止按钮返回 false

例:

```
AutoConv.nDialog[mStaNum].DialogShow("流水线" + Convert.ToString(mStaNum) + "阻挡气缸缩回超时，是否继续等待？", "等待", "忽略", 100, 200);
```

B.

```
public bool DialogShow(string MsgStr, string OKBtnLabel, int OKStep, int 输出编号 = -1, int 输出时间 = -1)
```

功能:

弹窗并阻塞进程

形式参数:

MsgStr:弹窗显示的文本

ContinuBtnLabel:继续按钮的文本显示

OKStep:点击继续按钮返回此数据

返回值:

bool:点击继续按钮 true

例:

```
AutoConv.nDialog[mStaNum].DialogShow("流水线 " + Convert.ToString(mStaNum) + "载具已流出,仍然有载具感应信号!" + "\r\n" + "流出异常，是否继续等待？", "等待", 100);
```

3.3 Zcm.DoAndDi

Zcm.DoAndDi 提供了操作 IO 的各种方法

3.3.1 WaitDi()

A.

```
public bool WaitDi(ValueType 输入编号, ValueType 需求状态, int 延时时间 = 0, int 超时时间 = 3000)
```

功能:

阻塞并等待输入状态变成需求状态

形式参数:

输入编号:IO 输入对应的索引

需求状态:需要的 IO 输入的状态

延时时间:延时判断时间

超时时间:等待时间

返回值:

bool:在超时时间内被置为需求状态则返回 true，反之则返回 false

B.

```
public bool WaitDi(ValueType 输入编号, ValueType 需求状态, int 延时时间, int 超时时间, bool 弹窗提示重试 = false)
```

功能:

阻塞并等待输入状态变成需求状态

形式参数:

输入编号:IO 输入对应的索引

需求状态:需要的 IO 输入的状态

延时时间:延时判断时间

超时时间:等待时间

弹窗提示重试:超时是否弹窗提示，false 不提示，true 提示，弹窗后点击重试再次判定

返回值:

bool:在超时时间内被置为需求状态则返回 true，反之则返回 false

3.3.2 WaitDone()

A.

```
public bool WaitDone(ValueType 输出编号, ValueType 输出状态, ValueType 输入编号, ValueType 输入状态, int 延时时间, int 超时时间, bool 弹窗提示重试 = false);
```

功能:

将对应的输出置为需要的状态并阻塞线程等待输入状态变成需求状态

形式参数:

输出编号:IO 输出对应的索引

输出状态:需要的 IO 输出的状态

输入编号:IO 输入对应的索引

需求状态:需要的 IO 输入的状态

延时时间:延时判断时间

超时时间:等待时间

弹窗提示重试:超时是否弹窗提示，false 不提示，true 提示，弹窗后点击重试再次判定

返回值:

bool:在超时时间内被置为需求状态则返回 true，反之则返回 false

B.

```
public bool WaitDone(ValueType 输入编号, ValueType 需求状态, int 延时时间, int 超时时间, bool 弹窗提示重试 = false)
```

功能:

阻塞并等待输入状态变成需求状态

形式参数:

输入编号:IO 输入对应的索引

需求状态:需要的 IO 输入的状态

延时时间:延时判断时间

超时时间:等待时间

弹窗提示重试:超时是否弹窗提示，false 不提示，true 提示，弹窗后点击重试再次判定

返回值:

bool:在超时时间内被置为需求状态则返回 true，反之则返回 false

3.4 Zcm.LanguageHelper

Zcm.LanguageHelper 提供了切换语言的方法

3.4.1 SetLanguage()

```
public static bool SetLanguage(string language, object form)
```

功能:

设置系统语言选择

形式参数:

language:需要的语言

form:设置的窗体

返回值:

bool:设置成功返回 true，失败则返回 false

例:

```
LanguageHelper.SetLanguage("中文", this);
```

3.5 Zcm.Moving

Zcm.Moving 提供了轴移动的种方法

3.5.1 AbsMove()

A.

```
public bool AbsMove(ValueType Card, ValueType Axis, double Position, double AxisVel = -1)
```

功能:

单轴绝对位置运动

形式参数:

Card:指定运动卡号

Axis:指定运动轴号

Position:目标绝对位置（mm）

AxisVel:轴运控速度（mm/s）

返回值:

bool:运动成功返回 true，运动失返回 false

B.

```
public bool AbsMove(ValueType AxisID, double Position, double AxisVel = -1)
```

功能:

单轴绝对位置运动

形式参数:

AxisID:对应轴的索引

Position:目标绝对位置（mm）

AxisVel:轴运控速度（mm/s）

返回值:

bool:运动成功返回 true，运动失返回 false

例:

```
pMove.AbsMove(Axis.取料Z轴, Pos.Z[(int)StationID.Task01_取料工位, (short)点位.取料位置], 30);
```

3.5.2 WaitDone()

A.

public bool WaitDone(ValueType 站号, ValueType 点位编号, bool 多轴同动, double Z 轴安全抬升高度, int 到位延时时间, int 最大等待时间, double 低速接近行程 = 0, double 低速接近速度 = -1, double 低速抬升行程 = 0, double 低速抬升速度 = -1)

功能:

单轴绝对位置运动（根据轴控件的 StationID 属性去锁定对应的轴）

形式参数:

站号:对应轴控件的 StationID
点位编号:对应点位编号（与控件上添加的点位有关）
多轴同动:是否多个轴一起动作
Z 轴安全抬升高度:设置 Z 轴安全高度，方便走模组门型
到位延时时间:到位后延时时间
最大等待时间:等待动作执行完成时间，超时直接返回 false
低速接近行程:到目标点低速接近的行程
低速接近速度:低速接近速度
低速抬升行程:初始动作低速抬升的行程
低速抬升速度:低速抬升速度

返回值:

bool:运动成功返回 true，运动失返回 false

例:

```
pMove.WaitDone(mStationID, (short)点位.临时位置, false, Pos.Z[mStationID, (short)点位.取料过渡位置], 0, 10000);
```

B.

public bool WaitDone(ValueType 卡号, ValueType 轴号, double 目标位置, double 速度, int 到位延时时间, int 最大等待时间, double 偏移量)

功能:

单轴绝对位置运动

形式参数:

卡号:指定运动卡号
轴号:指定运动轴号
目标位置:目标绝对位置（mm）
速度:轴运控速度（mm/s）
到位延时时间:到位后延时时间

最大等待时间:等待动作执行完成时间，超时直接返回 false
偏移量:设置此变量可以相较于目标位置进行固定偏移
返回值:
bool:运动成功返回 true，运动失返回 false

C.

public bool WaitDone(ValueType 轴编号, double 目标位置, double 速度, int 到位延时时间, int 最大等待时间)
功能:
单轴绝对位置运动
形式参数:
轴编号:对应轴的索引
目标位置:目标绝对位置（mm）
速度:轴运控速度（mm/s）
到位延时时间:到位后延时时间
最大等待时间:等待动作执行完成时间，超时直接返回 false
返回值:
bool:运动成功返回 true，运动失返回 false

3.6 Zcm.MutiAxis

Zcm.MutiAxis 提供多轴运动的相关方法

3.6.1 OverTime

功能:运动执行是否超时，超时置为 true

3.6.2 AbsMove()

public bool AbsMove(short[] AxisID, double[] Position, double[] AxisVel)
功能:多轴不同速绝对运动（数组索引关系一一对应）
形式参数:
AxisID:轴对应索引的数组
Position:目标位置的数组
AxisVel:轴移动速度的数组
返回值:
bool:运动成功则返回 true，反之则返回 false

3.6.3 WaitDone()

public bool WaitDone(short[] AxisID, double[] Position, double[] AxisVel, short WaitTime, int MaxWaitTime = 60000)
功能:多轴不同速绝对运动（数组索引关系一一对应）
形式参数:
AxisID:轴对应索引的数组
Position:目标位置的数组
AxisVel:轴移动速度的数组
WaitTime:到位延时时间
MaxWaitTime:超时时间

返回值:
bool:运动成功则返回 true，反之则返回 false

3.7 Zcm.OneAxis

Zcm.MutiAxis 提供单轴运动的相关方法

3.7.1 OverTime

功能:运动执行是否超时，超时置为 true

3.7.2 AbsMove()

```
public bool AbsMove(ValueType AxisID, double Position, double AxisVel = -1)
```

功能:单轴绝对运动
形式参数:
AxisID:轴对应的索引
Position:目标位置
AxisVel:轴移动速度
返回值:
bool:运动成功则返回 true，反之则返回 false

3.7.3 WaitDone()

A.

```
public bool WaitDone(ValueType AxisID, double Position, double AxisVel, int WaitTime, int MaxWaitTime = 60000, double 低速接近行程 = 0, double 低速接近速度 = -1, double 低速脱离行程 = 0, double 低速脱离速度 = -1)
```

功能:单轴绝对运动
形式参数:
AxisID:轴对应的索引
Position:目标位置
AxisVel:轴移动速度
WaitTime:到位延时时间
MaxWaitTime:超时时间
低速接近行程:到目标点低速接近的行程
低速接近速度:低速接近速度
低速抬升行程:初始动作低速抬升的行程
低速抬升速度:低速抬升速度
返回值:
bool:运动成功则返回 true，反之则返回 false

B.

```
public bool WaitDone(ValueType CardNum, ValueType AxisNum, double Position, double AxisVel, int WaitTime, int MaxWaitTime = 60000, double 低速接近行程 = 0, double 低速接近速度 = -1, double 低速脱离行程 = 0, double 低速脱离速度 = -1)
```

功能:单轴绝对运动
形式参数:

CardNum:指定运动卡号
AxisNum:指定运动轴号
Position:目标位置
AxisVel:轴移动速度
WaitTime:到位延时时间
MaxWaitTime:超时时间
低速接近行程:到目标点低速接近的行程
低速接近速度:低速接近速度
低速抬升行程:初始动作低速抬升的行程
低速抬升速度:低速抬升速度

返回值:

bool:运动成功则返回 true，反之则返回 false