RTHmi 编程手册

Version1.2.0

版权说明

Zmotion®

为了更好的服务于广大受众,深圳市正运动技术有限公司,对所发布的信息(含文字、数据、图片等)作出以下声明:

本手册版权归深圳市正运动技术有限公司所有,严禁任何媒体、网站、个人或组织以任何 形式或出于任何目的在未经本公司书面授权的情况下抄袭、转载、摘编、修改本手册内容,或 链接、转贴或以其他方式复制用于商业目的或发行,或稍作修改后作为他用,前述行为均将构 成对本公司手册版权之侵犯,本司将依法追究其法律责任。

涉及运动控制器软件每个指令和函数的介绍和范例,请参阅正运动技术公司 RTBasic/RTPlc/RTHmi编程手册。

本手册中的信息资料仅供参考。如涉及产品升级,内容需要更改,恕不另行通知!正运动 技术公司保留对本资料的最终解释权!如需获取更多详情请登陆正运动技术公司网站。

, 调试机器要注意安全!请务必在机器中设计有效的安全保护装置,并在软件中加入出错处理程序,否则所造成的损失,正运动技术公司没有义务或责任对此负责。

修订记录

更新日期	版本号	版本(更改)说明		
2023/10/13	V1.0.0	RTHmi 编程手册初版发布	zxl	
		1. 文本库新增"代码页"功能,支持多种语言显示;		
		2. 补充 HMI_LASTWINDOW 指令说明;		
2024/1/17	V1.1.0	3. HMI_CONTROLATTR、VKEY_MODE、HMI_LISTTEXTS、	zxl	
		HMI_DEFAULTATTR、HMI_DEALINFO 指令新增参数及补充		
		示例;		
2024/2/29	V1.1.1	附录虚拟键值说明新增。	zxl	
2024/4/7	V112	1. SCROLLBAR_INIT 指令最多同时滚动条数增加至 32;	z v]	
2027/7/7	¥ 1.1.2	2. 其他错误内容修改。	ZAI	
		1. HMI DEALINFO 指令增加参数 EDITNEXT、EDITPREV 获取		
		下一个、上一个编辑控件;		
		2. HMICONTROLATTR 指令增加参数 FOCUS 设置、获取控件		
2024/6/19	V1.1.3	焦点状态;	zxl	
		3. 修改 1.1.6 章节内容;		
		4. 附录新增 hmi 版本功能介绍;		
		5. 附录 hmi 版本功能介绍新增: V1.02.04 版本内容。		
	1. 文本库新增"以语言方式显示"或"以状态方式显示"巧			
		2. Hmi 设置: 基本属性新增"不使用文本库格式文本"功能,		
		"水平分辨率"和"垂直分辨率"可编辑功能;		
		3. 窗口导入新增窗口对比功能;		
		4. 更新图片、完善内容;		
		5. 控件箱元件新增报表视图、文件浏览器、菜单、树形图, 新增		
		章节 4.3.26 - 4.3.29 介绍新增元件;		
2024/12/26	V1.2.0	6. 新增操作指令章节: 6.6.22 - 6.6.28;	hj	
		7. SETEX_ALPHA 指令优化;		
		 8. HMI_DEFAULTATTR 指令新增一个参数"RGB_GRAY", 灰阶颜色; 9. HMI_CONTROLATTR 指令优化:新增参数"MSELROWS", 		
		多选行数,正数往下多选,负数往上多选,0表示无多选。配合		
		"CURSORY"参数获取光标行可知道当前选中行区域;		
		10. 新增文件浏览器使用参考例程章节: 8.7。		



目录

第一章	章 R	ΓHmi 编	程入门	. 1
1	.1.	RTS	ys 编程特点	. 1
	1	1.1.1.	Hmi 组态程序开发	. 1
	1	1.1.2.	Hmi 任务	. 7
	1	1.1.3.	网络显示屏	. 8
	1	1.1.4.	Hmi 仿真运行	. 8
	1	1.1.5.	Xplcterm 运行	. 9
	1	1.1.6.	适用的控制器	10
	1	1.1.7.	适用的手持盒	10
1	.2.	常见	问题	10
第二章	章 H	MI 菜单	栏	11
2	2.1.	窗口	Ι	11
2	2.2.	资源	Í	14
	2	2.2.1.	控件箱	15
	2	2.2.2.	文本库	15
	2	2.2.3.	图片库	24
	2	2.2.4.	按键转换	34
2	2.3.	排列]	41
2	2.4.	编辑	ŧ	42
	2	2.4.1.	批量修改地址	42
	2	2.4.2.	Hmi 设置	45
2	2.5.	显示	设置	46
	2	2.5.1.	属性	46
	2	2.5.2.	快捷图片库	47
	2	2.5.3.	显示/隐藏图层	48
	2	2.5.4.	栅格和元件名称	48
2	2.6.	语言	/状态切换	49

第三章	组态窗口		51
3.1	. 窗口	1概述	51
	3.1.1.	窗口作用	51
3.2	. 窗口	1操作	51
	3.2.1.	窗口属性	51
	3.2.2.	窗口建立	53
	3.2.3.	窗口导入	54
	3.2.4.	窗口调用	56
	3.2.5.	窗口关闭	58
	3.2.6.	公共窗口	58
3.3	. 窗口	1类型	61
	3.3.1.	基本窗口	61
	3.3.2.	软键盘窗口	62
	3.3.3.	弹出窗口	63
	3.3.4.	菜单窗口	64
	3.3.5.	置顶窗口	64
第四章	组态元件		66
4.1	. 组态	5快捷工具	66
	4.1.1.	元件菜单	66
4.2	. 组态	5元件通用属性	68
	4.2.1.	寄存器	68
	4.2.2.	动作	69
	4.2.3.	基本属性	70
	4.2.4.	外观	71
	4.2.5.	位置和尺寸	72
	4.2.6.	格式文本	72
	4.2.7.	图片库和文本库	74
4.3	. 元件	+介绍及使用	74
	4.3.1.	线段/多线段/多边形	75

	4.3.2.	矩形	79
	4.3.3.	贝塞尔曲线	81
	4.3.4.	圆/圆弧/扇形	82
	4.3.5.	刻度	86
	4.3.6.	表格	88
	4.3.7.	导入	90
	4.3.8.	静态文本	92
	4.3.9.	静态图片	93
	4.3.10.	位状态显示	95
	4.3.11.	多状态显示	. 100
	4.3.12.	位状态设置	. 104
	4.3.13.	多状态设置	. 109
	4.3.14.	位状态切换开关	116
	4.3.15.	多状态切换开关	. 120
	4.3.16.	功能键	. 127
	4.3.17.	物理按键	. 132
	4.3.18.	列表	. 134
	4.3.19.	值	. 139
	4.3.20.	字符显示	. 143
	4.3.21.	滑块开关	. 147
	4.3.22.	定时器	. 152
	4.3.23.	自定义	. 155
	4.3.24.	CAD	. 157
	4.3.25.	三次文件编辑器	. 159
	4.3.26.	报表视图	. 161
	4.3.27.	文件浏览器	. 169
	4.3.28.	菜单	. 171
	4.3.29.	树形图	. 182
第五章	Hmi 调用	Basic 函数	. 189

5.1	I. H	mi 系统设置调用函数	189
5.2	2. 自	定义元件调用函数	190
5.3	3. 元	件调用函数	193
第六章	相关 Ba	sic 指令	194
6.1	1. 基	础指令	194
	6.1.1.	RUN 启动文件任务	194
	6.1.2.	SCAN_EVENT 数据状态变化扫描	194
	6.1.3.	SET_XPLCTERM 显示屏启动状态设置	195
	6.1.4.	SYSTIME 系统时间	195
6.2	2. 语	法指令	196
	6.2.1.	DMSET 数组区域赋值	196
	6.2.2.	ZINDEX_LABEL 建立索引指针	197
	6.2.3.	ZINDEX_ARRAY 访问数组	197
	6.2.4.	ZINDEX_CALL SUB 函数调用	198
	6.2.5.	ZINDEX_VAR 指针变量操作	198
	6.2.6.	ZINDEX_MARK 指针标号设置	198
	6.2.7.	ZINDEX_STRUCT 获取/访问结构变量	199
	6.2.8.	ZINDEX_ZVOBJ 获取对象索引数据	199
6.3	3. 显	示指令	200
	6.3.1.	LCD_FEATURE 读取显示器特征	200
	6.3.2.	LCD_LEDSTATE 控制 LED 灯状态	200
	6.3.3.	LCD_WDOGTIME 显示器掉线处理时间	200
	6.3.4.	DRAWNUM 自定义元件内显示数值	201
	6.3.5.	DRAWNUM2 指定位置显示数值	201
	6.3.6.	DRAWTEXT 绘制显示字符串	202
	6.3.7.	DRAWTEXT2 绘制显示字符串	203
	6.3.8.	DRAWLIBTEXT 显示文本库字符串	203
	6.3.9.	DRAWLIBTEXT2 显示文本库字符串	204
	6.3.10.	DRAWREVERSE 绘制方块	204

6.3.11.	DRAWRECT 绘制矩形	205
6.3.12.	DRAWLINE 绘制线段	205
6.3.13.	DRAWCLEAR 清除指定区域内容	206
6.3.14.	DRAWPIC 插入图片文件	206
6.3.15.	DRAWARC 绘制圆弧	206
6.3.16.	DRAWLIBPIC 插入图片库图片	208
6.3.17.	DRAWBEZIER 绘制贝塞尔曲线	208
6.3.18.	DRAWBSPLINE 绘制 B 样条曲线	209
6.3.19.	DRAWDTLIST 绘制图形	
6.3.20.	DRAWEX_LINE 绘制线段(带样式)	
6.3.21.	DRAWEX_ARC 绘制圆弧(带样式)	
6.3.22.	DRAWEX_BEZIER 绘制贝塞尔曲线(带样式)	
6.3.23.	DRAWEX_BSPLINE 绘制 B 样条曲线(带样式)	
6.3.24.	DRAWEX_RECT 绘制圆角矩形(带样式)	
6.3.25.	DRAWEX_ROTRECT 绘制旋转矩形(带样式)	
6.3.26.	DRAWEX_ELLIPSE 绘制椭圆(带样式)	219
6.3.27.	DRAWEX_SECTOR 绘制扇形(带样式)	220
6.3.28.	DRAWEX_POLYGON 绘制多边形(带样式)	221
6.3.29.	DRAWEX_POLYGON2 绘制多边形(table 存储)	222
6.3.30.	SETEX_LINE 设置线段属性	223
6.3.31.	SET_FONT 字体设置	223
6.3.32.	SET_COLOR 设置颜色	224
6.3.33.	SETEX_ALPHA 设置绘图透明度	225
6.3.34.	SET_REDRAW 重新绘图	225
6.3.35.	RGB 颜色属性	226
6.3.36.	HMI_LANG 文本库语言切换	226
6.3.37.	SCROLLBAR_FREE 释放滚动条	226
6.3.38.	SCROLLBAR_INIT 滚动条初始化	226
6.3.39.	SCROLLBAR_POS 获取/设置滚动值	227

	6.3.40.	SCROLLBAR_REFLASH 刷新滚动条	. 228
	6.3.41.	SCROLLBAR_DRAW 绘制滚动条	. 229
6.4.	触打	莫屏指令	. 229
	6.4.1.	TOUCH_ADJUST 触摸屏校正	. 229
	6.4.2.	TOUCH_SCAN 触摸动作扫描	. 230
	6.4.3.	TOUCH_STATE 获取触摸状态	. 231
	6.4.4.	MOUSE_SCAN 鼠标动作扫描	. 231
	6.4.5.	MOUSE_STATE 获取鼠标状态	. 232
6.5.	按银	建指令	. 233
	6.5.1.	MOUSE_WHEEL 获取鼠标滚轮值	. 233
	6.5.2.	KEY_STATE 获取物理按键状态	. 233
	6.5.3.	KEY_EVENT 获取物理按键状态变化	. 233
	6.5.4.	KEY_SCAN 获取物理按键编码	. 234
	6.5.5.	VKEY_MODE 开启虚拟键输入模式	. 234
	6.5.6.	VKEY_STATE 设置/获取虚拟键状态	. 235
	6.5.7.	VKEY_EVENT 获取虚拟键状态变化	. 236
	6.5.8.	VKEY_SCAN 获取虚拟键编码	. 236
	6.5.9.	VKEY_INPUT 输入虚拟键值内容到键盘窗口	. 237
	6.5.10.	VKEY_IME 设置/获取当前输入法	. 237
	6.5.11.	ZSIMU_KEY 仿真物理按键	. 237
	6.5.12.	ZSIMU_VKEY 仿真虚拟按键	. 238
6.6.	操作	乍指令	. 238
	6.6.1.	HMI_SHOWWINDOW 显示指定窗口	. 238
	6.6.2.	HMI_CLOSEWINDOW 关闭窗口	. 239
	6.6.3.	HMI_BASEWINDOW 切换基本窗口	. 239
	6.6.4.	HMI_FOCUSWINDOW 窗口焦点模式	. 239
	6.6.5.	HMI_LASTWINDOW 最后点击窗口	. 240
	6.6.6.	HMI_DEFAULTATTR 设置/获取 HMI 内置默认属性	. 240
	6.6.7.	HMI_DEALINFO 获取 HMI 处理信息	. 241

	6.6.8.	HMI_CONTROLSIZEX 获取元件宽度	
	6.6.9.	HMI_CONTROLSIZEY 获取元件高度	
	6.6.10.	HMI_CONTROLDATA 设置/获取自定义元件属性	
	6.6.11.	HMI_CONTROLBACK 设置/获取指定元件背景颜色	
	6.6.12.	HMI_CONTROLVALID 设置/获取元件使能	
	6.6.13.	HMI_CONTROLSTRING 获取字符串信息	
	6.6.14.	HMI_CONTROLATTR 设置/获取元件属性	
	6.6.15.	HMI_CONTROLTEXT 修改元件文本	
	6.6.16.	HMI_LISTTEXTS 修改列表元件文本	
	6.6.17.	HMI_LISTITEM 修改指定列表项文本	
	6.6.18.	HMI_STRAPPEND 元件文本追加	
	6.6.19.	HMI_IFMONO 获取窗口垄断状态	
	6.6.20.	HMI_WINDOWSTATE 获取窗口状态	
	6.6.21.	HMI_MOVEWINDOW 移动指定窗口	
	6.6.22.	HMI_TABLEVALUE 设置/获取表格数值	
	6.6.23.	HMI_TABLETEXT 设置/获取表格内容	错误!未定义书签。
	6.6.24.	HMI_TABLECURSOR 获取当前选中行列	
	6.6.25.	HMI_FILESE 获取当前选中文件	
	6.6.26.	HMI_FILEPATH 获取当前路径	
	6.6.27.	HMI_FILEFILTER 设置文件过滤器	
	6.6.28.	HMI_MENUITEM 菜单项属性获取/修改	
七章	DT 运动的	函数	错误!未定义书签。
7.1.	МС	WEDTSP/MOVEDTABSSP DT 直线运动	
7.2.	МС	VECIRCDTSP/MOVECIRCDTABSSP DT 圆弧运动	
7.3.	MC	WECIRC2DTSP/MOVECIRC2DTABSSP DT 三点画圆弧运动	
7.4.	MS	PHERICALDTSP/MSPHERICALDTABSSP DT 空间圆弧运动	
八章	参考例程		
8.1.	单车	由运动	
8.2.	物理	里键与虚拟键转换	

第

第

8.3.		动态	5列表	269
8.4.		视图	图缩放	272
8.5.		滚动	b条使用	277
8.6.		CAI	D 功能	281
	8.6.1	l .	CAD 导入矢量图片	281
	8.6.2	2.	CAD 结合三次文件使用	284
	8.6.3	3.	CAD 结合自定义元件使用	289
8.7.		文作	+浏览器使用	292
8.8.		例程	呈下载	303
附录				304
RTI	Hmi 肋	反本其	功能介绍	304
虚扎	以键值	说明	月	306
错误	吴码列	表		307

第一章 RTHmi 编程入门

RTHmi 是 ZMotion 运动控制器所使用的组态设计,使用前需要确认控制器是否支持 RTHmi 功能。

编写和调试 RTHmi 程序需要使用 RTSys 软件,支持 RTHmi 功能的运动控制器或仿真器 V5.20-20230706 以上版本固件,PC 在线命令发送需要 zmotion.dll 动态库。

RTSys 软件支持 Basic 程序、PLC 程序、Hmi 组态同时使用,可以使用程序在显示屏上动态绘图,建议 下载最新版本使用。(注:不推荐高版本创建的项目使用低版本的去打开及修改!可能导致项目异常且无法 复原。)

本文主要以 RTSys 平台介绍 RTHmi 编程。

1.1. RTSys 编程特点

1.1.1.Hmi 组态程序开发

在电脑里新建一个文件夹用来保存即将要建立的工程。打开 RTSys 编程软件,当前说明例程的 RTSys 软件版本为 V1.00.02,更新软件版本请前往正运动官方网站下载,官方网站上还有触摸屏例程提供下载,网址:www.zmotion.com.cn。

RTSys 编程开发流程:

1. 新建项目: 菜单栏"文件"-"新建项目"。

RTSys V1.00.02	2				- 🗆 ×
文件(F)		调试(D)			^ 😮
新建	最近的项目 1 lib ceshi zoi		🗠 🔁 🔁		
17开	2 动态列表.zpj	ROM 只读 向后导航 向前导航 🔨 🥕 编辑	示波器 寄存器 启动/停止调试 報 常用工具 调试	P助文档 ~ 帮助	STOP)紧急停止
天 河	<u>3</u> test.zpj			同州	
关闭所有	<u>4</u> PSO_HW.zpj			/声 土	* * ^
日 保存					
另存为					
保存所有					
iii 编译Lib					
新建项目					
打开项目					
关闭项目					
	×退出区				
文件视图 标题	签视图 组态视图			轴参数 帮助 属	生
命令与辅	俞出				
在线命令:				发送	捕获 清除
命令与輸出	查找结果				
				100%	



₹ 另存为					×
$\leftrightarrow \to \leftrightarrow \heartsuit$	▶ 此电脑 ▶ 本地磁盘	(E:) ト Hmi测	ìđ		i i -
> 🔜 桌面	文件名		大小 类型	修改日期	
	4				•
文件名(M):	Test		•	ZMC Project Files (*.zpj)	*
				保存	取消

点击"新建项目"后弹出"另存为"界面,选择一个文件夹打开,输入文件名后保存项目,后缀为".zpj"。

2. 新建文件:菜单栏"文件"-"新建文件"。

Test.zpj - RTSys V1.00.02 - E:\Hmi测试\			-	- D X
文件(F) 常用(O) 控制器(C) 编辑(E) 视图	B(V) 工具(T) 调试(D)			^ (
		□ → 示波器 告存器 启动/停止调试 帮助文档 常用工具 调试 帮助		5709 紧急停止
^素 文件视图 ▼ ×	· · · · · · · · · · · · · · · · · · ·		属性	→ 쿠 :
ž 文件名 自动运行				
	新建	×		
	新建文件类型:	文件名:		
	Basic Plc	Basic		
	Hmi	确定		
		取消		
·				
文件视图 标签视图 组态视图			轴参数 帮助 属性	
命令与输出				* ů
+484.4				5 (±++)
1133 印文: 会公与論中 春秋结里			友広 州3	和市
	错误: 0, 警告: 0, 消息: 0	没有连接	100%	+

点击"新建文件"后,出现下图弹窗,Hmi 一般和 Basic 混合编程,用 Basic 函数编写 Hmi 元件要实现的 功能,分别新建一个 Basic 文件和一个 Hmi 文件。

Basic/Plc/Hmi 分别针对 3 种不同类型的文件,表示 RTSys 支持的三种编程方式,基础连接使用步骤相

同,支持 Basic/Plc/Hmi 混合编程。

新建	×
新建文件类型:	文件名:
Basic	Basic
Hmi	福宁
	AMIAE
	取消

保存文件:确认后新建的文件会自动加入到项目"文件视图"中,如下图。在程序编辑窗口写好程序后,保存文件,新建的文件会自动保存到项目.zpj所在的文件夹下。

💦 Test.zpj - RTSys V1.00.02 - E:\Hmi测试\ - Hmi	i1.hmi						- 0 ×
文件(F) 常用(O) 控制器(C) 编辑(E)	视图(V) 工具(T) 调试(D)	HMI(P)					^ 🕜
	📑 🔳 💽 🚍			🔗 👼 📚	□ 栅格 📘	1 L2 L3 更多 0 ;	
新建窗口 导入窗口 背景预设 ③ 显示详细信息	控件箱 文本库 图片库 按键转换		止—型 二♀ 批量修改地址 Hmi设置	LI	☑ 元件名称 S₀ \$	S1 S2 S3 更多0 ;	🚥 紧急停止
窗口	资源	排列	编辑	显示		语言/状态	_
		·	· · ·		I	B ¹¹	
業又件砚图 ▼ ⁺ × ■ Basi	C1.bas Hmi1.hmi	×			÷	馮性	₩ 4 ×
文件名 自动运行 10:Starf	t						
Hmi 1.hmi							
						轴参数 帮助 国性	
命令与输出							
+HAA.						「「「「」」	捕菜 海除
11:55m文: 命令与输出 查找结果							
	位置 X- 038 V	- 21 供温- 0 慾告	:0 消息:0	沿台站	接	100%	

3. 设置文件自动运行:双击文件右边自动运行的位置,输入任务号"0"。

注: 1. 任务号可为任意数字,但不可设置超过控制器支持的最大任务数。

2. 设置了任务号的文件会自动同时运行,任务号数值不分优先级。

文件名称可重新自定义,先在该项目内关闭要重命名的文件。然后在文件处点击鼠标"右键"-"重命名文件"修改。



	×	文件视	图		×
自动运行		文件名		自动运行	
		Basic1.bas	E t	曾加到项目(A)	
0		Hmi 1.hmi		<u>分置(S)</u>	
				主成ZAR文件(E)	
			f.	删除文件(D)	
			Ī	重命名文件(R)	
	自动运行 0	× 自动运行 0	× 文件视 自动运行	× 文件祝客 自动运行 ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○	★ 文件视图 自动运行 文件名 自动运行 0 日本 日本 日本 日本

4. 组态程序编辑:在编辑组态程序之前,首先要打开"Hmi设置"窗口。

先切换到 Hmi 编程窗口,菜单栏"HMI"-"Hmi 设置"打开如下窗口(或者打开 Hmi 文件点击窗口外的空 白地方也可以显示该属性窗口),根据组态程序要应用的示教盒的尺寸,设置好水平分辨率和垂直分辨率, 分辨率需要提前确定,其他参数可以后续再设置。

根据需求选择是否需要设置初始化函数和周期函数,选择 Basic 里编写好的 GLOBAL 全局定义的 SUB 子函数。

Hmi属性窗口如下图,各属性含义请参考<u>第二章 Hmi设置</u>章节

属性	>
▲ 基本属性	
LCD编号	0
背光时间	0
屏保时间	0
起始基本窗口	10:Start
起始置顶窗口	无
初始化函数	
周期函数	
压缩图片	不压缩
图片质量	标准
文本自适应	False
不使用文本库格式文本	False
▲ 位置和尺寸	
水平分辨率	800
垂直分辨率	480 ,
轴参数 属性 帮助	

基础设置完成,新建组态窗口,添加组态元件,组态元件在组态窗口上显示。

Hmi 编程所需的窗口可在"Hmi"的菜单栏中进行新建和导入,组态元件则在菜单栏"HMI"→"控件箱"里

进行选择,建立 Hmi 文件后,自动新建三个软键盘窗口和一个起始基本窗口 10:Start。窗口和元件的使用说明参见<u>第三章和第四章</u>。

Rt Test.zpj	- RTSys V1.00.0	2 - E:\Hmi测试\ - Hm	i1.hmi											-	
 文件(F)	常用(O) 拍	:制器(C) 编辑(E)	视图(V)	工具(T)	调试(D)	HMI(P)									^ ()
		a .								-		_			
C	G	 显示缩略图 			1 in 1		● 皐川町王	L <u>a</u> Fà		▲ 📚	□ 栅格	Lo Li	L ₂ L ₃ 更多 0		
新建窗口	导入窗口 背景形	没 ● 显示详细信息	控件箱	文本库 图片周	₣ 按键转换		ġ 🔄 📌	批量修改地址 Hmi设置	居性 快掛	建図片库 显示/隐藏図层 、	☑ 元件名称	S0 S1	S ₂ S ₃ 更多 0		紧急停止
	छ □		1	资源		排列		编辑		显示			语言/状态		
文件视	₹ ~ ₽	× 控件箱		→ 쿠 ×	Basi	c1.bas 🖹 Hr	ni1.hmi	×				Ŧ	属性		* 4 ×
文件名	自动运	〒 ▲ 矢量图形			10:Start								▲ 基本属性		
Basic1.bas		○ (低段/多)	地段/多边形										窗口编号	10	
Hmi1.hmi	0	「短形											窗口名称	Start	
		「田寒尔曲	A48										窗口类型	基本窗口]
													垄断	False	
			188/12										⊿ 外观		
													背景颜色	F5F5	5F5
													绘制边框	False	
													图片来源	无	
		⊿ 控件											公共窗口1	无	
		▲	2										透明度	100	
		▲ 静态图片	t										▲ 位置和尺寸		
		② 位状态量	眎										水平位置	0	
		🏆 多状态显	眎										垂直位置	0	
			置										宽度	800	
		123 多状态说	置										局度	480	
			1换开关												
		· 多状态切	」换开关												
		555 功能键													
		♪ 物理按键	8												
4		▶ 三 列表													
文件视图 杨	透视图 组态视	图 123 值		-	4							×	轴参数 帮助 屈性		
命令与	俞出														
														4877	0000
在线命令:													发送	捕获	清除

在"控件箱"中选择元件后,将元件放置于组态窗口尺寸范围内,打开元件"属性"设置元件相关参数,如 下图。

直接拖拽元件选择放置的位置和元件大小,或在元件"属性"的"位置和尺寸"栏设置。

💦 Test.zpj - RTSy	s V1.00.02 - E	::\Hmi测试\ - Hmi1	1.hmi													- 0	×
文件(F) 常用(0	0) 控制器	ŧ(C) 编辑(E)	视图(V)	工具(T)	调试(D)	HMI(P)											^ 🕐
		○ 显示缩略图		Т 🗖	o 💶		Г <u>Б</u> (6 ф. ф. (1)			Au		٠	□ 栅格	Lo L	₁ L₂ L₃ 更多 0 ;		
新建窗口 导入窗口	1 背景预设	◉ 显示详细信息	控件箱 文	本库 图片	库 按键转换	再11日	. 💀 🖣 💌 メ	批量修	₩ Hmi设置	i 属性	快捷图片库	显示/隐藏图层	☑ 元件名称	So S	5 S₂ S₃ 更多 0 ;	j 💿 🕿	急停止
	窗口			资源			排列		编辑			显示			语言/状态		
文件视图	₩ 4 ×	控件箱		• 4 ×	🖻 Basi	c1.bas	Hmi1.hr	ni ×						-	属性		▼ # ×
文件名	自动运行	▲ 矢量图形		*	10:Start										▲ 基本属性		
Basic 1.bas		○ 結約/条件	韵/多边形												元件编号	1	
Hmi 1.hmi	0	1 16形		- 1											元件名称	Button1	
		< D要欠曲	(JE	- 1											显示层次	底层	
			-06 #+12%	- 1											有效显示	显示	
			88/12	- 1						-					采用有效控制	False	
		一刻是		- 1						But	ton				安全时间ms	0	
				- 1							_				绑定虚拟按键	No Key	
		1 号人		- 1						-		-			绑定物理按键	0	
		⊿ 控件		- 1											⊿ 外观		
		▼ 静态文本		- 1											图片来源	背景图片库	£
		💦 静态图片		- 1											背景图片库	0\按钮\9	
		② 位状态显示	示	- 1											绘制边框	False	
		🏪 多状态显	示	- 1											是否图片化	False	
		12 位状态设	置	- 1											⊿ 标签		
		123 多状态设计	망 												文本库		
															格式文本(0)		
		 · ·	10.717C												格式文本(1)		
		These	1940/19K												⊿ 动作		
		100 り月日秋世													动作	调用函数	
		小 物理按键													松开时动作	False	*
▲	● 组本和图	1 列表			4									Þ	抽象数 帮助 国性		
		123 伯		•											THE ALL LAND MALE		
命令与输出																	▼ 4 ×
在线命令:															发送	捕获	清除
命令与输出 查找结	课																
				6	立置 X: 618, Y:	109	错误: 0,	, 警告: 0, 消息:	: 0			没有	连接		100%		+

5. 连接控制器/仿真器。

编辑好 Basic 程序和 Hmi 程序,点击"常用"/"控制器"-"连接"控制器或仿真器。若选择连接控制器则将 弹出"连接到控制器"的窗口,连接方法参见下节。

File3Editor.zpj - RTSys V1.00.01 - D:\	系统文件与桌面\Desktop\HMI新功能\File3	iditor\ - main.hmi					-	- ×
文件(F) 常用(O) 控制器(C) 約	烏嶺(E) 视图(V) 工具(T) 调试(D) HMI(P)						^ 🕄
		 比較控制器 砂範定控制器 砂範定控制器 かいたがり器 かいたやり器 かいたやり器 かいたやり器 かいたやり器 かいたやり器 かいたやり器 かいたやり器 かいたやり器 かいたやり器	编译所有 增加到项目	この と で こ 本 で こ 本 こ こ 本 こ こ 本 こ こ 本 こ こ 本 こ こ 本 こ こ 本 こ こ 本 こ 、 本 こ 、 本 こ 、 本 、 、 、 、 、 、 、 、 、 、 、 、 、	R文件 注释 AI	LM		
	控制器			项目	指示	TX-		
	🖻 main.bas 🗎 main.hn	ni ×			~	属性		- # >
G:CharPad-Lower	13:Secing	36:String	64:WordStateString	13:String 12:	Stripping.	▲ 基本属性		
- A:CharPad-Upper - 8:NumKeypad	ADC	NDC	加速 ADC	ADC	ADU	LCD编号	0	
🖹 10:Start	^{41:StaticText} FILE3文件列表	52:StrinABC ABC	tring 496 tring	ABC tring 48:Strin	ABC	背光时间	0	
1:Rect1(Rect)	52-String 72-String	101 Valu121 String			6:Cas	展促时间	0	
- 3:Custom3(Custom)	ABC	BC 102 .Valu122.String				記始其木廠口		
5:Button1(Button)	54:String 71:String	123 ABC					Ŧ	
	ABC	123 ABC				起始直映園口	70	
- 10:Rect4(Rect)	55:String 70:String ABC	BC 104 Valu24 String				初始化的数		
11:Rect10(Rect)	56-String 69-String	105-Valr125-String				周期函数		
12: lext11(string)	ABC	BC 123 ABC				压缩图片	不压缩	
15:Rect15(Rect)	57:String 68:String	123 ABC				图片质量	标准	
- 16:StaticText16(StaticText)	ABC 2	107 Valu127 String				文本自适应	False	
- 35:Text35(String)	ABC 67:String	BC 100.V-1.120.Coming				▲ 位置和尺寸		
36:Text35(String)	59:String 66:String	123 ABC				水平分辨率	800	
40:Rect32(Rect)	ABC	BC 109 Valt129 String				垂直分辨率	480	
- 47:Timer47(Timer)	60:String 65:String	pc 110 Valua String						
48:Text48(String)	ABC	DC 123 ABC						
	61:Button 63:String 62:Butto	123 ABC						
51:Text48(String)	王一页 ABC ³ 下一页	18: Marx Tene7: String						
52:Text48(String)	60-D							
- 54:Text38(String)	8:Buttop1:Button 92:Button 93:B << 保存 昇存 +3	utton 94:Button 95:Button 主释 -注释 杏找	96:Button 97:Button 日 基地 指销	98:Button 99:Button 10 音価 洗择	0:Button 5:Button 新建			
				T IN NO IT	49122	10.00 M		
又1460回 你登祝回 组织代回						和参数等别。		
命令与输出								
are a provide								
							1.000	
在线命令:						发送	捕获	消除
命令与输出 查找结果								
	位置 X: 219 V: 10	(株)字・0 禁告・0 消白・		沿右边	= #2	100%		

连接是否成功输出窗口会打印出信息提示。

例如:成功连接到控制器 ZMC406:

命令与输出

Connected to Controller:ZMC406 Version:4.93-20230531.

注意: 使用 RTHmi 必须控制器固件支持! 若固件不支持则同时打印如下信息:

命令与输出

Connected to Controller:ZMC406 Version:4.93-20230307. Controller not support RTHmi function.

成功连接到仿真器:

命令与输出

Connected to Controller:VPLC5xx-Simu Version:5.20-20190529.

若连接失败,也会有相应的信息提示。

6. 下载运行程序: 下载程序到控制器/仿真器运行。



下载程序时可选择"下载到 RAM"或"下载到 ROM",下载成功"命令与输出"窗口打印下载成功提示,同时程序下载到控制器并自动运行。

RAM 下载掉电后程序不保存, ROM 下载掉电后程序保存。下载到 ROM 的程序下次连接上控制器之后 程序会自动按照任务号运行。

触摸屏程序必须下载到 ROM。

注意事项:

- 1. ZMC00x 系列控制器不支持下载到 RAM。
- 2. 未建立项目只打开文件则无法下载到控制器。
- 3. 自动运行的数字 0 表示任务编号 0,以任务 0 运行程序,任务编号不具备优先级。
- 4. 若整个工程项目内的文件都不设置任务编号,下载到控制器时,系统提示如下信息。

WARN: no program set autorun.

命令与输出

WARN: no program set autorun.

Down to Controller Ram Success, 2023-03-06 14:34:05, Elapsed time: 1094ms.

打开工程项目时,选择打开项目 zpj 文件,若只打开其中的 bas/plc/Hmi 文件,程序无法下载到控制器。如下图,RAM/ROM 下载图标均为灰色。

Rt	RTSys V1.00.01 - Hr	mi1.hmi									
文	牛(F) 常用(O)	控制器(C)	制辑(E)	视图(V)	工具(T)	调试(D)	HMI(P)			
[: 亲	ないます。 建 打开… 保存	保存所有	■ 注 注 注 注 注 ·	「 「 断 开 注 接	「 加 下裁到RAI	M 下载到ROM				→ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○	₽ 野助文档
	文件				控制器		编	構	常用工具	调试	帮助
工具箱	文件视图	-	- џ ×	Hr	mi1.hn	ni ×					
	文件名	自动运行	ī	10:Sta	art						
					1:Button						

1.1.2.Hmi 任务

要运行 Hmi 文件就要给 Hmi 文件设置自动运行任务号,每个显示屏最多允许一个 Hmi 文件运行,若两

个 Hmi 文件同时运行,会报错。

Hmi 文件需要占用一个自动运行任务号。Basic 文件可根据需求选择是否设置自动运行任务号。

文件视图		×
文件名	自动运行	
Basic1.bas		
Hmi 1.hmi	0	

Hmi 通常情况下要和 Basic 混合编程, Hmi 元件调用 Basic 的函数或寄存器, Hmi 也可以和 PLC 混合编程。

Hmi 任务不是实时的,需要实时性高的场合请使用独立的其他任务。

1.1.3.网络显示屏

RTHmi 支持通过以太网把电脑或其它触摸屏作为显示屏使用,也可以使用自身的显示屏(必须控制器带有显示屏)。

控制器支持多个显示屏时,通过设置组态文件属性,"Hmi系统设置"窗口选择使用的显示屏编号。

控制器支持的显示屏个数和最大分辨率在连接了控制器之后,在线命令发送?*max 打印信息,查看 max Hmi参数。

例如:某型号控制器打印 max Hmi 结果如下

max Hmi:2, x:1024 y:800 支持 2 个远端 Hmi, 支持的最大尺寸为 1024*800。

1.1.4.Hmi 仿真运行

程序下载到仿真器后,点击仿真器的"显示",即可运行 Hmi 界面,显示 xplc screen 组态界面进行仿真。 在不连接控制器和示教盒的时候,可以连接到仿真器之后调试。



Here Simulator							– 🗆 X
_ 輸入仿真	2	□ 3	4	5		□ 7	Power Led1 Led2
	10	11	12	13	14	15	
16 17	18	19	20	21	22	23	设置显示
	0			EN	ICODER3	0	
ENCODER1	0			EN	ICODER4	0	
ENCODER2	0			EN	ICODER5	0	
─ 输出监控 □ 0 □ 1	2	□3 [4	5	6	7 🗖 8	9 10 11
							退出

1.1.5.Xplcterm 运行

在没有触摸屏的时候,使用 Xplc Screen 仿真触摸屏的运行,可将 PC 作为一台人机界面,可以连接仿 真器或控制器。

程序下载到仿真器或控制器后。打开 Xplc Screen,点击菜单栏"工具"-"插件",设置连接的显示屏编号和 IP 地址,默认显示的是仿真器的 IP, IP 地址填入后点击 Connect 即可使用。

XPlc Screen		×
	Connect X	
	Led Num: 0 Reconnect RTSvs	
	IP Address: 127.0.0.1 Connect	

1.1.6.适用的控制器

支持 RTHmi 编程方式的控制器型号有:4 系列及 4 系列以上相应固件支持(4 系以下暂不支持 RTHmi, 型号带 H 的除外,例如: XPLC120H 等),部分控制器需要升级固件才能支持,详情请联系厂家。

(其他非上述支持的低系列型号,建议使用 Zdevelop 进行开发)

1.1.7.适用的手持盒

ZHD 系列手持盒后缀带 X 的表示支持 Hmi 组态编程方式,并且触摸屏程序可下载到控制器上(不需要单独下载到触摸屏),再将触摸屏连接到控制器即可通讯。

型号	ZHD300X	ZHD400X	ZHD500X
分辨率	480*272	800*480	1024*600
按键	47 个	18 个	16 个
USB 口	1 个	1 个	1个
急停开关	1个	1个	1个
触摸屏	支持	支持	支持

1.2. 常见问题

当程序运动出错后, RTSys 软件"命令与输出"窗口会显示出出错信息, 如果出错信息没有看到, 可以通过命令行输入?*task 或"故障诊断"窗口再次查看出错信息, 双击出错信息可以自动切换到程序出错位置。

问题	可能原因
无法正常显示 Hmi 界面	分辨率设置错误,请按照硬件要求设置分辨率
修改 Hmi 分辨率后页面显 示不正常	修改 Hmi 分辨率后选择"否",又将数值恢复选择了"是",导致页面分辨率异常。此时可以采用"ctrl + z"撤回。或关闭 RTSys 时选择"不保存"即可恢复。
自定义元件无法刷新显示	在刷新函数中,没有使用 SET_REDRAW 指令
元件调用函数失败	函数定义必须是 GLOBAL 类型
窗口操作失败	窗口操作类型与窗口属性类型不一致。
显示屏不亮, 或亮度不够	检查控制器的供电: USB 供电必须用质量很好的线并保证电脑的
显示屏不亮, 或亮度不够	USB 口供电足够, 否则请使用串口头的 24V 供电;
通讯不上	检查网线。

第二章 HMI 菜单栏

注: HMI 菜单栏仅在主窗口显示 Hmi 文件时才显示。

2.1. 窗口



新建窗口:在当前项目中新建一个或多个窗口。详情可查看第三章【窗口建立】章节。

导入窗口:在当前项目导入其他项目/文件中已创建的 Hmi 窗口。详情可查看第三章【窗口导入】章节。

背景预设:对 Hmi 窗口背景及部分元件设置默认的样式/颜色,保存设置后在新建窗口或元件时生效。 对已创建的窗口和元件不更改原样式。详细使用方法可参考本章【图片库】章节。

背景预设操作方法:选择需更改样式的元件/窗口,点击"修改",有两种样式方式选择。

方法一:若使用图片库样式则选择"图片库"可将图片导入,更改图片比例可调整元件显示大小;

方法二:若不使用图片库样式则在"状态 0"和"状态 1"自定义颜色。(注意:两种样式方式只能二选一 生效,使用了图片库样式则会覆盖自定义颜色样式)



窗口预设背景颜色则使用方法二设置"状态 0"自定义颜色,新建窗口后生效。

窗口对比:选择一个Hmi文件与当前Hmi文件进行各个窗口及窗口元件的对比。具体操作如下:

(1)点击菜单栏"HMI"-"窗口对比",点击","图标,弹出文件选择窗口,选择需要对比的窗口所在的项目文件路径,选择需要的.Hmi文件。

窗口对比		×	
当前文件:	📿 对比文件:		
₽1 ₽1 ₽1 ₽1 ₽1			



(2)选定待对比的 HMI 文件后,"窗口对比"界面将呈现两个文件的窗口信息:左侧列出当前文件的 各窗口名称,右侧则显示对比文件的相应窗口名称。系统将根据窗口号对左右两边的窗口进行逐一对 比。若窗口名称不一致,或窗口内的元件(包括类型、大小、位置等属性)存在差异,该窗口名称右侧 将标记(≠)。

窗口对比			×
当前文件:	C	对比文件:	C: \Users \Administrator \Desktop \Hm
当前文件: ▶ 6:CharPad-Lower ▷ 7:CharPad-Upper ▷ 8:NumKeypad ▶ 10:Start(≠) ▶ 11:报表视图 ▷ 12:文件选择(≠)		对比文件:	C:{Users\Administrator\Desktop\Hm ▲ ower pper :) 器 (≠)

(3)点击窗口名称前的"▶"图标,可以展开查看该窗口下的具体元件信息。对于元件不一致的情况, 其名称右侧同样会标记(≠)。



显示缩略图:在组态视图显示窗口缩略图、窗口号及窗口名称。(黄色背景表示当前已打开的窗口,鼠标单击可切换)参考以下左图。

显示详细信息:在组态视图显示窗口信息(窗口号、窗口名称)和元件信息(已创建的元件编号和元件 名称)。参考以下右图。



2.2.资源



2.2.1. 控件箱

控件箱主要是存放各种组态元件。HMI开发可在该窗口中添加组态元件。在菜单栏"HMI"→"控件箱"即可打开,一般默认将控件箱隐藏至界面最左侧。更多元件内容可参考【<u>第四章-组态元件</u>】介绍。

2.2.2.文本库

介绍: 采用标签形式一次性设置不同状态下的多种语言文本以及每个文本对应的字体并在对应元件调用它。(一个标签最多支持 256 种状态,一种状态最多支持编写 8 种语言文本)

窗口各功能介绍如下表:

功能		描述
状态选择	选择"以语言方式显示"时	寸显示,用于切换状态,设置不同状态的文本库内容
语言选择	选择"以状态方式显示"时	寸显示,用于切换状态,设置不同语言的文本库内容
以语言方式显示	标签设置栏以语言的方式显可设置 0-7种	显示,依次设置每种状态下的语言文本库内容,语言数
以状态方式显示	标签设置栏以状态的方式显可设置 0-255 个	显示,依次设置每种语言下的状态文本库内容,状态数
导入/导出	导入/导出文本库文件	
字体管理器	载入/删除字体文件	
今休迟罢	字体	选择各语言显示的字体
于评议直	代码页	输入各国语言编码(使用其他国家语言时须填写)
	编号	/
	标签名	文本库名字
	引用次数	引用该文本库的次数(不可设置)
	状态数	设置文本库的状态数
标签设置	语言数	设置使用的语言数量
	格式文本	设置语言文本格式
	语言 0-7	选择"以语言方式显示"时显示,写入文本内容
	状态 0-15	选择"以状态方式显示"时显示,写入文本内容 状态可选 0-15 / 16-31 / / 240-255,以 16 递增

注意[1]: 在中文操作系统下需使用其他国家语言显示时,需要在"代码页"处填入该国语言的编码,使用中国语言时可不填;若在非中文操作系统下,使用中国语言时则须填对应中文语言编码(一般为 936)。 注意[2]: 只要字体采用的是 ttf(TrueType Font)格式,用户即可在"代码页"设置中填入 65001(UTF-

8编码),可以有效避免乱码问题的出现。

态选择	≩:		以语言方式图	眎	○ 以状态:	方式显示				导入			导出		字体管理	32
体设置	ੀ: ਸ਼ਿਆ	2 27	<u>→</u> 1	(石合) 0	ंत्र	i à - 0	·五子		海车	-	किन्द्र व		(고수 2			
-44	日日 日日 日日 日日	1 1 1 1 1 1 1			다. 4명		日日	4	日日 1	0	0 百司 0					
ゆ	志ん iAA	志/	м	ж, IA 0	<u>مج</u>	64	#A 64		±1, 1/1,		жл ил 0		志ん いへ			
HJJQ																
S设置	<u>-</u> :														搜	索
号	标签名	引用次数	1 状态数	语言数	格式文本	语言 0	ì	语言 1		语言 2		语言:	3	语言	4	
	双击添加															
											_					
0 0 1/8.	除	青除											确知	È	<u>لي</u>	肖
mile ji库	\$	青除	11° 7 1 10							P)			确	Ì	取	肖 ····································
■ ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●		青除	以语言方式图	显示	● 以状态	方式显示	0-15	•]		导入			() 研究	È	取	肖器
■ 「「库 「「」 「」 「」 「」 「」 「」 「」 「」 「」 「	涂		以语言方式图 言 1	示 语言 2	 ● 以状态 语 	方式显示	0-15	-	语言!	导入	语言 6				取	肖器
■ ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	涂	青除 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	火语言方式団 言 1 ↓	ī示 语言 2 默认	 ● 以状态 语 	方式显示 言 3 认	0-15 百言 默认	•	语言。	导入 5	语言 6 默认		- 确3 导出 语言 7 默认		東注	肖器
● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	涂 译: 【: 【: 【: 】 【: 】 】 】 】 】 】 】 】 】 】 】 】	青除 	以语言方式型 言 1 认	記示 语言 2 默认 0	 ● 以状态 诺 默 0 	方式显示 言 3 认	0-15 语言 默认 0	•	· 语言 · 默认 0	导入 5	语言 6 默认 0		· 确: 导出 语言 7 默认 0		東 家 家 家 家 家 家 家 家 家 家 家 家 家 家 家 家 家 家 家	肖器
	涂 译: 	青除 ▼ ○ □ 译 男 0	以语言方式団 言 1 认	記示 语言 2 默认 0	 ● 以状态 语 数 0 	方式显示 言 3 认	0-15 语言 默认 0		· 语言, 默认 0	导入 5	语言 6 默认 0		· 确3 导出 语言 7 默认 0		取) 字体管理	肖
● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	徐 译: 	青除 	以语言方式団 言 1 认 切 状态数	- 記 - 記 - 記 - 記 - 記 - 記 - 記 - 記	 · 以状态 语 第 0 格式文本 	方式显示 言 3 认	0-15 语言 默认 0		语言 默认 0	导入 5	语言 6 默认 0		确 等出 语言 7 默认 0		取 字体管理 操 援 4	肖
● ₩W 库 這 本 体 码 设 号	除 ≩: □ □ □ □ □ □ □ □ □ □ □ □ □	青除 ▼ 0 译 思 0 引用次:	以语言方式图 言 1 认 状态数	記示 语言 2 默认 0 	 ・ 以状态 	方式显示 言 3 认 状态 0	0-15 语言 默认 0	4 状态 1	· 语言 · 默认 0	· 导入 5 状态 2	语言 6 默认 0	(确 等出 语言 7 默认 0		取) 字体管理 援羽 4	肖
■ m [™] 库 這 本 体 码 资 号	除 ≩: □	青除 ▼ ○ ほ 男類 0 3]用次3	以语言方式団 言 1 认 文 状态数	 記書 2 默认 0 语言数 	 · 以状态 语 默 0 格式文本 	方式显示 言 3 认 状态 0	0-15 语言 默认 0	• 4 状态 1	语言 默认 0	导入 5 状态 2	语言 6 默认 0	状态	· · · · · · · · · · · · · · · · · · ·	È 状态	取 字体管理 と と と 、 、 、 、 、 、 、 、 、 、 、 、 、	背
● ● ● ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■	除 ≩: □ □ □ □ □ □ □ □ □ □ □ □ □	青除 ▼ 0 译 思 0 3 日用次:	以语言方式图 言 1 认 状态数	2示 语言 2 默认 0	 ・ 以状态 	方式显示 言 3 认	0-15 语言 默认 0	4 状态 1	· 语言 · 默认 0	· 导入 5 状态 2	语言 6 默认 0	(- - - - - - - - - -		取) 字体管理 4	肖
■ ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	除 ≩: 遣: □ □ □ □ □ □ □ □ □ □ □ □ □	青除 ▼ ○ で ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・	以语言方式団 言 1 认 文 状态数	ま 正 ま に は に に ま ま に は に ま ま い に は い ま ま い に い ま ま い い ま ま い い ま ま い い ま ま い い ま ま い い ま ま い い ま ま い い ま ま い い ま ま い い ま ま ま い い ま ま ま い い ま ま ま い い ま	 以状态 语 默 0 	方式显示 言 3 认 状态 0	0-15 语言 默认 0	• 4 状态 1	· 语言 · 默认 · 0	- 与入 5 - 	语言 6 默认 0	【	· 确 · · · · · · · · · · · · · · · · · · ·		取 字体管理 と と と 、 、 、 、 、 、 、 、 、 、 、 、 、	肖 器 索
● ●●● 「「「」」「「」」「」」「」」「」」「」」「」」「」」「」」「」」「」」「」」	除 ≩: □ □ □ □ □ □ □ □ □ □ □ □ □	青除 ▼ 0 译 男 0 3 1 1 1 1 1 1 1 1 1 1 1 1 1	以语言方式 言 1 认 数 状态数	 法示 语言 2 默认 0 语言数 	 以状态 语 默 0 格式文本 	方式显示 言 3 认 状态 0	0-15 语言 默认 0	4 状态 1	· 语言 · 默认 · 0	- - - - - - - - - - - - - - - - - - -	语言 6 默认 0	(- - - - - - - - - -		取) 字体管理 4	肖 器
● ₩ 「」「」「」「」「」」「」」「」」「」」「」」「」」「」」「」」「」」「」」「	除 ≩: Ţ:	青除 ■ ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	以语言方式 言 1 认 文 初 状 态 数	ま こ ま は に は に は に は に は に は い に は い に は い い い い	 ・ 以状态 语 第 0 	方式显示 言 3 认 状态 0	0-15 语言 默认 0	4 状态 1	· 语言 · 默认 0	· 导入 5 / 状态 2	语言 6 默认 0	(- - - - - - - - - -		取 字体管理 と 投 2 4	肖 器 索
■ ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	除	青除 ▼ ○ ○ ○ ○ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	以语言方式団 言 1 い い な 状态数	 記書 記書 2 默认 0 () ()	 ● 以状态 语 数 0 	方式显示 言 3 认 状态 0	0-15 语言 默认 0	• • 【 状态 1	· 语言: 默认 0	- 导入 5 - - - - - - - - - - - - - - - - - -	语言 6 默认 0	· · · · · · · · · · · · · · · · · · ·	- - - - - - - - - -		取) 字体管理 】 4	消日の日本の日本の日本の日本の日本の日本の日本の日本の日本の日本の日本の日本の日本
「「」」「」「」」「」」「」」「」」「」」「」」「」」「」」「」」「」」「」」	除 ▲ … … … … … … … … … … … … …	青除 ■ ■ ■ ■ ■ ■ ■ ■ ■	以语言方式 言 1 认 文 数 次 次 次 次 数 3 3 3 5 5 5 5 5 5 5 5 5 5 5 5 5	 法示 语言 2 默认 0 	 ● 以状态 语 第 0 格式文本 	方式显示 言 3 认	0-15 语言 默认 0	4 状态 1	· 语言 · 默认 · 0	- 与入 5 - - - - - - - - - - - - - - - - - -	语言 6 默认 0	////////////////////////////////////	- - - - - - - - - -		取; 字体管理 4	

使用方法:

(一) 文本库设置("以语言方式显示"为例)

1. 选择"以语言方式显示",在[标签设置]板块双击添加"标签名"。("标签"用于区分元件调用哪个文本库)。

状态选排 字体设计	≩: 0 ≝:	•	● 以语	言方式显	示	○以状态;	方式显示				导入		Ę	寻出	1	字体管理器	p4/g
	语言	[0	语言	1	语言 2	语	言 3	语言	[4	语言!	5	语言 6		语言 7			
字体	默い		默认		默认	揌	і Л	默认		默认		默认		默认			
代码页	0		0		0	0		0		0		0		0			
示签设计	<u>و</u> :															搜索	
编号	标签名	31,F	用次数	状态数	语言数	格式文本	语言 0		语言 1		语言 2		语言 3		语言 4		T
1	双击添加																
		K															

2. 设置所需的状态数和语言数、文本的格式。

文本库																	×
状态选择	圣:	0	- ●以	语言方式显	示	○ 以状态	方式显示				导入		Ę	寻出		字体管理器	\$
字体设置	晋 : 立:																
		语言 0	语言	1	语言 2	걽	譶 3	语言	4	语言!	5	语言 6		语言 7			
字体		默认	默认		默认	鶢	iλ	默认		默认		默认		默认			
代码页		0	0		0	0		0		0		0		0			
标签设置	昱:															搜索	
编号	标签名	4	引用次数	状态数	语言数	格式文本	语言 0		语言 1		语言 2		语言 3	l	语言	4	语
1	lable0		0	2	4	点击设置											
2	双击添	加															
-														_			P
₩	除	清防	£											确	定	取消	

3. 对状态进行选择,在已选择的状态下给对应数量的语言添加需要显示的文本内容。

如下图所示: 在状态0下添加了对应的语言文本。

Zmotion[®] _{正运动技术}

本库																	×
犬态选择	¥: 0)	- •¢	语言方式显	示	○ 以状态	方式显示				导入			导出		字体管理器	p (a
字体设置	1:		\searrow														
	ì	语言 0	语	1	语言 2	语	信 3	语言	4	语言	5	语言 6		语言 7			
字体	Ę	默认	默礼		默认	鍡	і ,	默认		默认		默认		默认			
代码页	C)	0			0		0		0		0		0			
·签设3	1 :															搜索	1
编号	标签名		引用次数	状态数	语言数	格式文本	语言 0		语言 1		语言 2		语言:	3	语言	4	ĩ
	label0		0	2	4	点击设置	끄기		OFF		关闭		オフ				T
	双击添加	1															
刪	涂	清約	余											确定	Ē	取消	í

4. 在[状态选择]处切换为状态 1,可在语言处添加处于状态 1 时需显示的文本内容。(状态选择数与设置的状态数有关)

文本库																	×
状态选择 字体设置	¥: 1 5:		, 0以ii	999月19日	示	○以状态	方式显示				导入		:	导出		字体管理器	e b
	ì	吾言 0	语言	1	语言 2	语	信 3	语言	= 4	语言	5	语言 6		语言 7			
字体	黑	tίλ	默认	$\overline{}$	默认	鍡	પ્ર	默认	L. C.	默认		默认		默认			
代码页	0		0			0		0		0		0		0			
标签设置	Ē:															搜索	
编号	标签名		引用次数	状态数	语言数	格式文本	语言 0		语言 1		语言 2		语言:	}	语言	4	译
1	label0		0	2	4	点击设置	୍ୟ		ON		开启		オン				
2	双击添加																
4														_			•
刪	除	清院	ŝ											确	定	取洋	í

5. 设置好标签及语言文本后,若无需设置字体,点击[确定]即可保存。

若需对字体进行设置,则按以下步骤操作。

6. 首先需要在项目中添加字体文件。在文本库中右上方打开[字体管理器]。

字体管理器	×
当前字体:	
文件名 字体名	载入字体
	刪除字体
٠	

7. 点击[载入字体], 根据存放路径找到对应的字体文件, 点击[打开]即可载入字体(支持.ttf和.zft格式)

₹ 打开					×
\leftrightarrow \rightarrow \heartsuit	▶ 此电脑 ▶ 本地磁盘(E:))	• 文本	☞ ▶ 字体文('牛	•
> 🔜 桌面	文件名		大小	类型	修改日期
	default.zft		497 KB	ZFT 文件	2023/3/30 15:13:57
	📄 default2.zft		136 KB	ZFT 文件	2023/3/30 15:13:56
	📄 default3.zft		887 KB	ZFT 文件	2023/3/30 15:15:07
	▲ FZYTK.TTF		3,166 KB	TrueType 字体	2023/3/30 16:07:52
	🔊 puhui-medium.ttf		9,337 KB	TrueType 字体	2023/3/30 16:07:49
	🔊 puhui-regular.ttf		9,497 KB	TrueType 字体	2023/3/30 16:07:51
	🔊 simsun(1).ttf		10,254 KB	TrueType 字体	2023/3/914:45:13
	🔊 SourceHanSansCN-Bold.ttf		8,407 KB	TrueType 字体	2023/3/30 16:07:51
	STCAIYUN.TTF		5,541 KB	TrueType 字体	2022/10/10 14:02:20
	🔊 vssetup.ttf		140 KB	TrueType 字体	2023/3/30 15:15:24
	zzz.zft		1 KB	ZFT 文件	2023/3/30 15:15:15
	☐ 方括号.zft		242 KB	ZFT 文件	2023/3/30 14:47:02
	🔊 方正粗黑宋简体.ttf		2,710 KB	TrueType 字体	2023/3/30 15:14:20
		可	在此切换查	i找的字体格式	
	•				
文件名(N):			*	All files(*.zft;*ttf)	•
				打开(0)	取消



8. 成功载入字体后在列表中显示对应的文件名及字体名。(仅.ttf 格式支持显示字体名)

字体管理器		×
当前字体:		
文件名	字体名	载入字体
STSONG.TTF	华文宋体	
ariblk.ttf	Arial Black	
BOD_R.TTF	BOD_R	删除字体
calibrib.ttf	calibrib	
HarmonyOS_Sans_Medium_Italic.ttf	HarmonyOS_Sans_Medium_Ita	
4	•	

9. 导入字体后可在[字体设置]板块对每个语言进行字体设置,调用对应文本库后运行即可显示。

注意:由于语言 1,语言 4 分别是韩文字体和日文字体,所以需要在[代码页]处填写对应的语言编码。 (本例为中文操作系统,故中文字体可不填入代码页编码)

	1. (石会 o	海车		海会。	<u>تح</u> ا	≟ 0			海车	-	海舎で		(고수 - 7			
<u>~~~</u>	出言 リ	日言	1	出言 2	出	言 3	日言	4	日言	5	(出言 6)		間に			
7₩ 代码页	素和A 949	まれん 1252		款1A 0	また1 932	A 2	まれへ 0		жли 0		表い、 0		悲入1八 0			
·签设置	<u>-</u> :														搜索	
编号	标签名	引用次数	状态数	语言数	格式文本	语言 0		语言 1		语言 2		语言:	3	语言 4		
L	label0	0	2	4	点击设置	끄기		OFF		关闭		オフ				
2	双击添加															

(二) 文本库调用

1. 按上述操作设置好文本库后,可在元件的"属性"中调用文本库。



双击文本库标签名,可打开文本库并跳转到对应文本库位置。

文本库														×	属性	×
语言选择	e 0	J ON	语言方式员	示	◉以状态	方式显示	0-15	a f	島り			异出	字	体管理器	有效显示 显示	*
今休设署	. <u>e</u>				0 90000	57 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5	0 15								采用有效控制 False	
	·· (五本 o	:五		洒车 。	3	五 -	海主 4	洒车	-	(五十) c		(五十 7			绑定的设备编号 本地	
合体	0 百日 1	프 다 1 (4 명	1		14 E8		中 二 (49		5	16日 0					绑定的寄存器类型 DT	
子体	zh in	±A KA		#A 6A	*	50 KA	#A 64	za in		#A 64		, 14			绑定的寄存器编号 1000	
代码风	U	1252		932	9	50	U	U		U		U			选中调用函数 test111	
1- ** 10 -	с.													地安	⊿ 外观	
杯金饭面														12.77	绘制边框 True	
编号	标签名	引用次数	状态数	语言数	格式文本	、 状态 0	状态	1	状态 2		状态:	3	状态 4	扐	边框颜色 000000	
1	label 1	2	5	4	点击设置	向左	向右		向上		向下		不变		图片来源无	
2	label2	1	7	4	点击设置	向上	向上5	行	向上10行	1	向下		向下5行	Ó	角标类型 样式1	
3	label3	0	40	1	点击设置	1	2		3		4		5	6	角标颜色 🔲 000000	
4	双击添加														大道距 2	
															选中部) 颜色 99 CCFF	
															⊿ 标签	
															文本库 label1 •	
															列表文本 Item …	-
															文本库	
4						_								•	文本库的名称,如果为空,则表示使用文本体	<u>.</u>
															签。	
BBIR	余	除										确注	È	取消	轴参数 属性 帮助	

2. 下载运行后可通过指令: Hmi LANG=ilang(语言号)进行切换。

通过修改寄存器的数值可改变语言的状态。例如:寄存器值为0显示的是语言0的状态0,当寄存器的 值变为1时,显示的是语言0的状态1。

下方用例程演示。

设计思路:

- 用[多状态切换开关]元件对[多状态显示]元件实现状态切换。
- [多状态显示]元件通过调用子函数运行 Hmi_LANG 指令进行语言切换。
- (1) 组态设计

在组态窗口中添加以下元件:静态文本(仅用于标题显示)、多状态显示、多状态切换开关。



(2) basic 编程

```
GLOBAL dim runsub_sign = 0
                            '变量标志
                          '函数调用切换
GLOBAL SUB sub switch()
 IF runsub sign = 0 THEN
    langue0
  ELSEIF runsub_sign = 1 THEN
   langue1
 ELSE
   langue2
 ENDIF
 runsub\_sign=runsub\_sign+1
 IF runsub sign > 2 THEN
     runsub_sign = 0
 ENDIF
END SUB
```



GLOBAL SUB langue0() '使用语言 0

HMI_LANG=0

END SUB

GLOBAL SUB langue1() '使用语言 1

HMI_LANG=1

END SUB

GLOBAL SUB langue2() '使用语言 2

HMI_LANG=2

END SUB

(3) 调用过程

【多状态显示】元件

- ▶ 在"属性"窗口中的"文本库"选择需要使用的文本库内容(例子的文本库名称为 label0);
- ▶ 在"属性"窗口中的"动作"调用子函数 sub_switch();

【多状态切换开关】元件

- ▶ 在"属性"窗口中的"状态数量"设置为2,并对应状态添加文本;
- ▶ 在"属性"窗口中的"动作"选择[循环],"动作数据"设置为1;
- (4) 运行效果
 - 当前两个元件均处于状态 0,此时[多状态显示]元件显示文本库中状态 0 时的语言 0 文本 (即文本显示为:关闭);

xplc screen	×
文本库使用	
关闭 切换状态0	

▶ 点击[关闭]元件,可实现语言切换。如下左图,切换至语言1;右图则切换至语言2。



xplc screen	×	xplc screen	×
文本库使用		文本库使用	
OFF 切换状态0		國閉 切换 <mark>状态0</mark>	

点击[切换状态]元件,可实现状态切换。如下图两个元件切换至状态1。[多状态显示]元件显示文本库中状态1时的语言0文本(即文本显示为:开启);

xplc screen	×
文本	库使用
开启	切换状态1

▶ 点击[开启]元件,可实现语言切换。如下左图,切换至语言1;右图则切换至语言2。

xplc screen	×	xplc screen	×
文本库使用		文本库使用	
ON 切换状态1		開啟 切換状态1	

2.2.3.图片库

介绍: 汇总存放 Hmi 元件样式图片或自定义图片的库。包括系统图片库和用户图片库。该库中支持对图片样式颜色,不同状态及不同语言对应显示内容/颜色等进行修改。

系统图片库有多个类别,均为系统默认样式,不支持删除或增加以及修改图片属性。该库为用户提供了 丰富的元件样式选择,用户可直接在该处选择并应用。

用户图片库为用户创建自定义图片库,支持添加外部图片到该库中使用。

注:图片库最大数量为 512!



使用方法(以开关控制台灯亮灭为例):

【新建图片库】

(一) 台灯图片导入

1. 鼠标右键单击[用户图片库],选择"新增"。弹出窗口输入新建图片库名称点击确定即可。


新增		×
名称:	picture gallery	确定
		取消

2. 双击打开已新建的[picture gallery],点击"新增";在弹窗中填入名称,设置好对应语言数及状态数即可。

3. 新增完成后,单击选中该图片库(底色为黄色即为选中状态),在右侧选择对应的状态,给不同的状态添加对应的图片。



4. 选择状态 0, 鼠标双击上方空白处(或点击"修改"),选择目标图片存放路径,单击选择图片,点击"打开"即可。

₹ 打开			×
\leftrightarrow \rightarrow \heartsuit	- ▶ 台灯图片		in
> 🛄 桌面	文件名	▲ 大小 类型	修改日期
	►台灯1.jpg	77 KB JPEG 图像	2023/7/18 15:48:05
	■ 台 X.] 2.jpg	66 KB JPEG 图 惊	2023/7/18 15:48:07
	4		•
文件名(Ŋ):	台灯1.jpg	• AllF	PictureFiles +
		I	打开(0) 取消

5. 状态 0 添加图片后,按上述方法给状态 1 添加对应图片,单击选择状态 1 后双击上方空白处(或点击"修改")添加图片。添加完全部图片后点击"确定"即可保存。

图片库		×
图片库 □ 系统图片库 	ft	× 「 テ出修改 「 で の 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 0 0 0 0 0 0
		967×974 2828496 byte
		0/1 977×975 2858700 byte
		夏制
	新增 删除 导入 背景颜色: RGB(255,255,)	确定取消

(二) 开关按钮样式复制



1. 在新建的[picture gallery]图片库中,再新增[开关]的图片库,设置好对应的语言数和状态数后点击"确 定"即可完成创建。



2. 在[系统图片库]的分类中选择合适的样式,选择要复制的状态,点击"复制"即可将该状态下的样式复制。

注: 若不使用[系统图片库]已有样式,则可按上述新增方法自行添加图片。



3. 返回新建的[picture gallery]图片库,选择[开关]的图片库,单击选择状态 1,单击粘贴,即可将已复制的样式应用到该处。



4. 添加完各状态下的图片后,可根据需求修改图片显示颜色,点击"确定"即可。至此图片库创建完成。



【图片库应用】

(一) 元件直接应用

× ^ **?**

1. 新建 HMI 文件后,给 HMI 文件添加自动运行任务号,打开 HMI 文件。设置 Hmi 系统属性后,在 HMI 窗口中添加组态元件。

	💦 Hmi Test.zpj - RTSys V1.00.00	0 - D:\系统文件与	桌面\Desktop\HMI测试\test	t\ - Hmi1.hmi					- 0
	文件(F) 常用(O) 控制器((C) 编辑(E)	视图(V) 工具(T) i	周武(D) HMI(P)					
	c c	○ 显示缩略图	т 🖪 🚍	= 4 〒 1: + + 1: 三		- 樹格	L ₀ L ₁ L ₂ L ₃ 更多0	\$	
	新建窗口 导入窗口 背景预设	◉ 显示详细信息	文本库 图片库 按键转换	F I I I C C C 🖈	批量修改地址 Hmi设置	显示/隐藏图层	S ₀ S ₁ S ₂ S ₃ 更多 0	\$	
	窗口		资源	排列	编辑	显示	语言/状态		
	「具箱」	▼ -> × B	Basic1.bas 🖪 H	mi1.hmi ×			*	属性	
H	▲ 矢量图形	<u> </u> 10	Start					▲ 基本属性	
	○ 线段/多线段/多边形							窗口编号	10
	1 矩形			0. Deline and				窗口名称	Start

2. 从控件箱中选择组态元件"位状态显示"和"位状态切换开关"添加到窗口10中。

	🖲 Basic1.bas 🗟 Hmi1.hmi 🔺	▼ 属性	▼ ∓ ×
□ 本 1 日本 1	10:Start	▲ 基本	属性
○ 総約/条総約/条の形		窗口	編号 10
		窗口:	名称 Start
	2-BitSwitch	窗口	美型 基本窗口
		垄断	False
		▲ 外观	
		背果	颜色 F5F5F5
₩ 表格		绘制:	边框 False
日本		图片:	来源 无
控件		公共	窗口1 无
■ 静态文本		透明	度 100
● 💦 静态图片		∠ 位置	和尺寸
② 位状态显示		水平	位置 0
- - - - - - - - - - - - - -		垂直	位置 0
□ 位状态设置		宽度	800
123 多分本沿署		高度	480
			Ŧ
City AURCRE			
		轴参数 等	展助 層性
列表			▼ # ×
<u>Mec</u> 字符显示			
滑块开关			
◎ 定时器			
④ 自定义			
A CAD			发送捕获清除
🔒 三次文件编辑器			
	位置 X: 553, Y: 223 错误: 0, 谐意: 0, 消息: 0	VPLC5xx-Simu:127.0.0.1 - 运行	100%

4击选中元件 2"位状态切换开关",软件界面右侧弹出该元件"属性"窗口。在"外观"→"图片来源"
 选择"背景图片库",在下一项的"背景图片库"中单击"…"即可打开图片库。

	属	性		х
	4	基本属性		
2-BitSwitch		元件编号	2	
		元件名称	BitSwitch2	
		显示层次	底层	
		有效显示	显示	
		采用有效控制	False	
		安全时间ms	0	
		绑定的设备编号	本地	
		绑定的寄存器类型	Μ	
		绑定的寄存器编号	0	
		外观		
		图片来源	背景图片库	_
		背景图片库	0\开关\18 .	
		绘制边框	False	-
	轴	参数帮助属性		

4. 打开图片库后,在图片库中找到目标样式选中后并点击"确定"即可应用到该元件,根据需求对元件





大小进行调整。(元件1的样式应用与上述方法一致,绑定寄存器设置为M2)

图片库样式成功应用到元件后显示如下图:



(二) 背景预设应用至元件

1. 打开 HMI 文件,通过菜单栏"HMI"→"背景预设"打开如下窗口,可直接在该处修改元件的样式。





2. 选择需要修改样式的元件(以位状态切换开关为例),点击"修改"弹出如下窗口。



3. 点击图片库栏后的"选择"即可打开图片库窗口,选择需要应用的样式,点击"确定"即可。





4. 完成图片库样式修改之后,返回以下窗口,可在该窗口调整图片显示比例,以及状态背景色等,完成 设置后点击"确定"即可。

注: 状态背景色仅供预览参考, 实际应用时不生效。



【调用过程】

- (一) 位状态显示元件
- ▶ 单击该元件在"属性"窗口选择绑定寄存器为 M0。
- (二) 位状态切换开关
- ▶ 单击该元件在"属性"窗口选择绑定寄存器为 M0。
- ▶ 在"属性"窗口的动作选择"状态反转"。

Basic1.bas Hmi1.hmi ×			
10:Stag:StaticText 图片定估田	属性		×
国开杆使用	▲ 基本属性		î
	元件编号	1	
I:BitState	元件名称	BitState1	
B-Bi+Switch	显示层次	底层	
	有效显示	显示	
	采用有效控制	False	
	安全时间ms	0	
	绑定的设备编号	本地	
	绑定的寄存器类型	М	
· · · ·	绑定的寄存器编号	0	
	▲ 外观		
	图片来源	背景图片库	
	背景图片库	1\picture gall	
	绘制边框	False	
	轴参数 帮助 属性		

【运行效果】

- 1. 将程序下载到控制器/仿真器后,打开 xplc screen 即可显示组态窗口。
- 当两个元件处于状态 0 时,即台灯与开关均处于关闭状态,如下左图所示。当按下开关切换为状态1 时,台灯同时切换为状态1,即为开灯状态。如下右图所示。



2.2.4.按键转换

介绍:用于将物理按键与虚拟键的功能绑定起来,实现通过操作物理按键即可使用虚拟键功能的效果。 该工具已预设了 ZHD300X 和 ZHD400X 的按键功能。同时支持将已设置好的键值内容导出或导入。

使用方法:

(一)若使用己预设的 ZHD300X 和 ZHD400X 的按键功能:打开"按键转换"窗口→点击右上方的"下拉菜单"按钮,即可选择对应型号示教盒→点击"预设",列表即可显示对应按键功能→点击"确定"设置成功。
(若要更改预设内容,可点击"清除"即可全部清除;若只更改部分内容可直接双击修改)

(二)若需新建一套自定义按键功能:打开"按键转换"窗口→在对应空白格处双击即可填入物理键值和虚 拟键值→点击"确定"即可保存。

目态按键转	换表				×
			ZHD400X	•	预设
物理键	虚拟键	虚拟键描述		^	*=•
11	128	VKEY_F1			消际
12	129	VKEY_F2			导入
13	130	VKEY_F3			
14	131	VKEY_F4			导出
15	132	VKEY_F5			
16	133	VKEY_F6			
24	150	VKEY_1LEFT			72.
34	152	VKEY_2LEFT			明定
44	154	VKEY_3LEFT			取消
C/	102	WEV ALCET		•	

2.2.4.1. 物理键

物理键是指外部设备上的实际按键,每个按键都有独有的编码值,按下时会发送一条信息,这条信息 就是按键的编码值。

物理键的编码值由硬件决定,程序中无法修改。使用不同的外部设备,对应按键的编码值也不同。

ZHD400X 标准物理按键编码:

Global Const key_fl = 11 '功能键 1

Global Const key_f2 = 12 '功能键 2

Global Const key_f3 = 13 '功能键 3

Global Const key_f4 = 14 '功能键 4

Global Const key_f5 = 15 '功能键 5

Global Const key_f6 = 16 '功能键 6

Global Const key_X-=24 '轴移动按键



Global Const key_X+=25

Global Const key Y = 34

Global Const key Y + = 35

Global Const key_Z- = 44

Global Const key Z+=45

Global Const key_U-=54

Global Const key_U+ = 55

Global Const key_A-=64

Global Const key_A+=65

Global Const key B-=74

Global Const key_B+ = 75

或直接查看 RTSys 软件里的组态按键转换表:

组态按键转	换表				×
			ZHD400X	•	预设
物理键	虚拟键	虚拟键描述		A	×=•^
11	128	VKEY_F1			消际
12	129	VKEY_F2			导入
13	130	VKEY_F3			
14	131	VKEY_F4			导出
15	132	VKEY_F5			
16	133	VKEY_F6			
24	150	VKEY_1LEFT			72.
34	152	VKEY_2LEFT			佣疋
44	154	VKEY_3LEFT			取消
си	100	WEV ALCET		*	

ZHD300X 物理按键的编码按行列组合而成,键值=行号(1-10)×10+列号(1-5)。

ZHD300X标准物理按键编码:

Global Const key_fl = 11 '功能键 1

Global Const key_f2 = 12 '功能键 2

Global Const key_f3 = 13 '功能键 3

Global Const key_f4 = 14 '功能键 4



Global Const key_f5 = 15 '功能键 5

Global Const key 1=51 '数字键 1,同时字母按键切换

Global Const key_2 = 52

Global Const key_3 = 53

Global Const key_4 = 61

Global Const key_5 = 62

Global Const key_6 = 63

Global Const key_7 = 71

Global Const key_8 = 72

Global Const key 9 = 73

Global Const key 0 = 81 '数字键 0

Global Const key Add= 83 '加号

Global Const key_Point=82 '小数点

Global Const key_xUp=25 'JOG 第一轴

Global Const key_yUp=35 '第 2 轴

Global Const key_zUp=45 '第 3 轴

Global Const key rUp=55 '第 4 轴

Global Const key xDown =24 'JOG 第一轴

Global Const key_yDown =34

Global Const key_zDown =44

Global Const key_rDown =54

Global Const key_Jog5L=64

Global Const key Jog5R=65

Global Const key_Jog6L=74

Global Const key Jog7R=75

Global Const key_Left=21 '左移

Global Const key_Up=22

Global Const key_Right=23

Global Const key_Down=32

Global Const key_SpeedUp=41



Global Const key_SpeedDown=43

Global Const key Step=84

Global Const key Manual=85

Global Const key_Reset =91 '复位

Global Const key_Del =92 '删除

Global Const key_Inset =93 '插入

Global Const key_Switch=94 'SHIFT 切换

Global Const key_Save =95 '保存

Global Const key_Esc =101 '取消

Global Const key_Edit =102 '编辑监控

Global Const key_File =103 '文件管理

Global Const key_Set =104 '参数设置

Global Const key Ent =105 '输入确定

或直接查看 RTSys 软件里的组态按键转换表:

组态按键转	换表				×
			ZHD300X	•	预设
物理键	虚拟键	虚拟键描述		^	×=•4
11	128	VKEY_F1			消际
12	129	VKEY_F2			导入
13	130	VKEY_F3			
14	131	VKEY_F4			导出
15	132	VKEY_F5			
21	145	VKEY_LEFT			
22	147	VKEY_UP			
23	146	VKEY_RIGHT			棚定
32	148	VKEY_DOWN			取消
C 1	201	WEV 1 CTAD		Ŧ	

2.2.4.2. 虚拟键

在实际编程中,如果使用物理键编码编写程序,那么程序的可移植性很低,所以程序编写时希望有一 个编码可以用在所有外设上,所以虚拟编码就出现了,只要将外设的物理键编码与虚拟编码一一对应,程 序就可以用在不同的外设上。

由于虚拟编码的操作方式和物理键编码相似,所以就叫做虚拟键。

Hmi 中, 虚拟键编码由底层封装而成, 程序中无法修改。

虚拟键编码值 0-127 都对应 ASCII 码表, 128 往后支持自定义功能。部分虚拟键值已定义功能,详情可参见<u>附录-虚拟键值表</u>。

2.2.4.3. 按键转换表的编辑

使用方法:

选择菜单栏"HMI"-"按键转换"打开下方页面,组态按键转换表主要由列表区、功能区和选择菜单3部 分组成。

目态按键转换	與表			\times
		选择菜单	- · 预i	ž
物理键	虚拟键	虚拟键描述	清	余
				λ
				Ł
		列表区	功能区	ζ
			确知	È
			للله المراجع ال	肖

1. 选择菜单

该工具已内置 ZHD300X 和 ZHD400X 的转换表,可在下拉菜单中选择。

2. 列表区

用于编辑并显示各物理键值对应的虚拟键值及功能。双击首行空白格即可手动输入。

物理键:设置值为外部设备按键的编码值。

虚拟键:设置值为希望与外部物理键绑定的虚拟编码值,0-127都对应ASCII码表,128往后则自定义功能。详情可参见<u>附录-虚拟键值表</u>。

虚拟键描述:对当前虚拟键功能的说明。

示例:将物理键 10 与虚拟键 201 绑定,参考下图。



		_		
			•	预设
物理键	虚拟键	虚拟键描述		海险
10	201	VKEY_1_STAR		用柄
				导入
				8.11
				守出
				确定
				取消

3. 功能区

预设:结合选择菜单,调出已经编辑好的转换表,目前只有 ZHD300X 和 ZHD400X。

442	目态按键转换	ŧ				×
				ZHD400X	•	预设
	物理键	虚拟键	虚拟键描述			×≠₽∆
	11	128	VKEY_F1			消味
	12	129	VKEY_F2			导入
	13	130	VKEY_F3			
	14	131	VKEY_F4			导出
	15	132	VKEY_F5			
	16	133	VKEY_F6			
	24	150	VKEY_1LEFT			福宁
	34	152	VKEY_2LEFT			NHAE
	44	154	VKEY_3LEFT			取消
	си	100	WEV ALEET			

清除:删除上图所有的转换设置,清空列表。

导入:从外部文件调用编辑好的转换表,选择由本页面导出的保存转换表信息的文件,文件格式.ini。

导出:把当前编辑的转换表导出为.ini格式文件,用来保存当前转换表。

确定:编辑好后,要点击确定才可以应用转换表,否侧下次打开按键转换表时为空。

取消:取消对转换表的操作并退出。

2.2.4.4. 按键转换指令

虚拟键值与虚拟键功能支持自定义绑定,可通过相应指令及程序编写实现。

与按键转换相关的 basic 指令主要为以下 6 条。

KEY STATE: 物理按键状态

KEY_EVENT: 物理按键状态扫描

KEY SCAN: 读取物理按键编码

VKEY STATE: 虚拟按键状态

VKEY EVENT: 虚拟按键状态扫描

VKEY SCAN: 读取虚拟按键编码

程序中可以使用 VKEY_SCAN 来捕捉是哪个虚拟键按下,根据按键转换表就可以知道对应的是哪个物理键;也可以直接使用 KEY_SCAN 捕捉是哪个物理键按下。但一般情况下不建议使用 KEY_SCAN 及物理键相关指令,因为不同外设的物理键编码都不同,这么用程序的可移植性较低,建议使用 VKEY SCAN 及其他虚拟键相关指令。

由于这些指令只能在自定义元件的刷新函数中使用(Hmi的初始化函数也行,但是不建议这么做), 所以至少要有一个自定义元件存在。扫描到按键按下后,把返回值赋值到一个自定义变量,在自定义元件 的绘图函数中根据返回值的不同,来调用不同的函数,实现不同的功能。

参考例程参见"物理键与虚拟键转换"。

2.3. 排列



介绍: RTSys 提供多种对齐排列方式,将多个元件按一定规则整齐排列,使得整个 Hmi 界面更加美观 有序。如下表:

	左对齐		水平居中对齐	+	宽度相同	*	水平居中
	右对齐	-	垂直居中对齐		高度相同	Ļ	垂直居中
10	上对齐		水平方向相同间隔		水平和垂直方向 相同尺寸	*	居中显示
	下对齐		垂直方向相同间隔	*	锁定元件	/	/

使用方法:

1. 将所需排列的元件同时框选,选择适宜的排列方法即可。如下图所示,将三个元件"下对齐"。



注意:

- 1. HMI 使用排列功能时,需同时选中两个及以上的元件该功能才可使用。
- 2. 该功能的对齐目标是以框选元件中显示为红色框的为目标元件,以目标元件为标准进行排列。
- 3. 红色框选元件默认以最先添加的顺序为准,与元件编号顺序无关。
- 4. 若需自定义目标元件,则先选中目标元件,按住"crtl"键,鼠标单个点击其他跟随元件,元件选择 完毕后松开"ctrl"键,再选择排列方式。

2.4.编辑



2.4.1.批量修改地址

介绍: 对多个 HMI 组态元件的寄存器地址进行批量修改,可修改为统一的寄存器类型,以及设置地址间距。通过菜单栏"HMI"→"批量修改地址"可打开如下窗口。

【编辑界面】

控件:显示选中多个元件中支持绑定寄存器的元件编号及名称。(注:仅显示支持绑定寄存器的组态元件。)

寄存器类型:选择对应元件需要绑定的寄存器类型,包括 AUTO(自动)、X(输入口 IN)、Y(输出口 OUT)、M(MODBUS_BIT)、S(状态寄存器)、D(MODBUS_REG/MODBUS_LONG/MODBUS_IEEE)、D.DOT(按位读取 MOBUS_REG)、DT(TABLE)、T(定时器)、C(计数器)。寄存器更多介绍可参考

<u>寄存器</u>章节。

地址:设置寄存器的起始地址。

自动地址:根据已设置好的起始地址和地址间距,将自动给选择 AUTO 类型的寄存器分配好地址及对 应类型。

批量	修改地址			\times
	控件	寄存器类型	地址	
	1:BitState1	М	0	
	3:BitSwitch3	AUTO	0	
	4:WordModify4	AUTO	0	
	5:WordState5	AUTO	0	
地址	山间距: 1	自动地址	确定 取消	

使用方法:

1. 在 HMI 窗口中,选中需批量设置寄存器的多个元件; (建议选择可使用相同类型寄存器的元件)

2. 打开"批量修改地址"窗口,可看到选中的支持绑定寄存器的元件名称及可设置的寄存器类型和地址;

3. 对元件的"寄存器类型"进行选择,点击即可弹出下拉列表;并设置寄存器地址起始编号; (选择 AUTO 则表示自动跟随上一个寄存器类型,选择其他类型则不改变已选类型)



量修改地址			\times
控件	寄存器类型	地址	
1:BitState1	М	0	
3:BitSwitch3	AUTO	0	
4:WordModify4	D	30	
5:WordState5	AUTO	0	
5:WordState5	AUTO	0	

4. 在窗口下方根据需求设置"地址间距",默认间距为 1。确定间距后点击"自动地址";该窗口中寄存器类型为 AUTO 的将被统一,地址则按设置的地址间距顺序排列显示。点击"确定"即可。如下图所示。

批量	修改地址			×
	控件	寄存器类型	地址	
	1:BitState1	Μ	0	
	3:BitSwitch3	Μ	10	
	4:WordModify4	D	30	
	5:WordState5	D	40	
Luka L			72.5	
地力		目动地址		

注意:

- 1. 第一个元件不能设置为 AUTO 类型;
- 2. 设置地址时注意寄存器占用空间是否重复;
- 3. 元件显示顺序按元件添加顺序排列,建议将需要使用同类型寄存器的元件按顺序进行选择(选中某

个元件后按住"ctrl"键按顺序依次对元件进行选择)

4. 每次自动地址后,若需重新修改寄存器类型并自动地址,需重新手动对某个寄存器修改类型,并将 在该寄存器之后的寄存器设置为 AUTO 类型。

2.4.2.Hmi 设置

对 HMI 系统进行初始属性设置,可修改整体 HMI 窗口分辨率、起始基本窗口等属性,详情可参加下表。通过菜单栏"HMI"→"Hmi 设置"即可打开 Hmi 系统设置的属性窗口。

其它打开方式:单击 HMI 窗口画布外的空白地方也可打开 Hmi 系统设置属性窗口。

注: 使用 Hmi 组态时,一般需先在"Hmi 设置"中设置好各参数属性。

名称	功能	说明
LCD 编号	设置 LCD 编号	连接到触摸屏时可通过该编号进行选择显示 哪个HMI文件内容
背光时间	示教盒实际背光时间	/
屏保时间	设置屏保时间	/
起始基本窗口	设置 HMI 起始显示的基本窗口	默认显示 10 号窗口
起始置顶窗口	设置 HMI 起始置顶窗口	/
初始化函数	添加 HMI 初始化函数	上电后只调用一次的函数,在 Basic 文件中 定义,函数的定义必须是全局(GLOBAL)的 SUB
周期函数	添加 HMI 周期函数	上电后不断周期扫描的函数,在 Basic 文件 中定义,函数的定义必须是全局(GLOBAL) 的 SUB
压缩图片	选择是否对图片进行压缩	旧压缩:有损压缩且不可逆(即使用 ZDevelop 中的压缩方式) 新压缩:无损压缩,图片画质不变 不压缩:不压缩图片
图片质量	选择图片显示的质量 (注:选择不压缩时,图片质量决 定是否防失真缩放)	标准:图片显示质量较低,但HMI性能较高高:图片显示质量较高,但HMI性能较低
文本自适应	文本自适应元件大小	文本显示内容超出元件范围时自动缩小字体,最小不低于用户设置的 50%
不使用文本库格 式文本	True: 使用控件格式文本 False: 使用文本库格式文本	/



水平分辨率	窗口显示的分辨率	可编辑
垂直分辨率	窗口显示的分辨率	可编辑

用户能在"分辨率选择"界面中选择预设分辨率或自定义分辨率。在"修改内容"部分,可以选择是否 让"窗口和控件"及"文字大小"(随宽度或高度变化)与新的分辨率设置同步调整。

A 符篇 清誦 · · · · · · · · · · · · · · · · · ·	51990172007 00000000000000000000000000000000		JTD/CJ/04Mg/ EMA
分辨率选择:	800×480	*	
- 分辨率	480×272 800×480	【内容	
水平分辨率:	1024×600 1024×768	窗口和控件	
垂直分辨率:	1280×720 1920×1080	文字大小	随宽度变化 🔹
字体大小变化示例 : 厕	自定义 家大小为16,新大小为16。		

2.5. 显示设置



2.5.1.属性

用于显示和设置 HMI 文件中的窗口/元件属性。打开该窗口需先新建/打开 HMI 文件,再在菜单栏 "HMI"→"属性"即可打开,否则"属性"则呈灰色不可点击状态。

窗口和元件的属性窗口如下图所示:

属性		×	「	禹	性			×
▲ 基本属性		*		a,	基本属性			
元件编号	9				窗口编号	10		
元件名称	Button9				窗口名称	Sta	art	
显示层次	底层				窗口类型	基	本窗口	
有效显示	显示				垄断	Fa	lse	
采用有效控制	False			a.	外观			
安全时间ms	0				背景颜色		F5F5F5	
绑定虚拟按键	No Key				绘制边框	Fa	lse	
绑定物理按键	0				图片来源	无		
⊿ 外观					公共窗口1	无		
图片来源	背景图片库				透明度	10	0	
背景图片库	0\按钮\9			a,	位置和尺寸			
绘制边框	False				水平位置	0		
是否图片化	False				垂直位置	0		
⊿ 标签					宽度	80	0	
文本库					高度	48	0	
格式文本(0)								
格式文本(1)								
⊿ 动作								
动作	调用函数							
松开时动作	False							
动作函数名								
▲ 位置和尺寸		•						
轴参数 帮助 属性			4	抽象	参数 帮助 属性			

Zmotion

2.5.2.快捷图片库

用于给 HMI 元件快速应用或移除图片库中的样式。通过菜单栏"视图"→"快捷图片库"即可打开该窗口。 操作方式:

添加图片库:打开 HMI 文件,选中 HMI 窗口中的单个元件,打开快捷图片库,找到想要应用的样式图 片,双击该图片,即可将图片快速应用到元件上。(图片库中有多个分类,可于[快捷图片库]窗口顶部点击 则弹出下拉菜单进行选择)

移除图片库:单击选中元件,在[快捷图片库]窗口中点击"移除图片库"即可。



2.5.3.显示/隐藏图层

对设置在不同图层的元件进行选择显示或隐藏。(在菜单栏"HMI"→"显示/隐藏图层"进行设置)

操作方法:

1. 点击元件, 打开属性窗口, 在元件"显示层次"中进行选择, 共有顶层、中层、底层可选。

2. 点击"显示/隐藏图层"弹出下拉菜单,选择某个图层进行显示或隐藏。

2.5.4.栅格和元件名称

选择 HMI 窗口中是否显示栅格,元件和窗口是否显示名称。栅格的作用便于用户将文件参考对齐。全部勾选则如下图显示:



Basic1.bas ⊫Hmi1.hmi ×	
10:Start	

2.6. 语言/状态切换

L ₀ L ₁ L ₂ L ₃ 更多	D ‡					
So S1 S2 S3 更多	D ‡					
语言/状态						

语言切换:对已调用文本库的元件进行语言切换。需先在文本库设置好当前状态下每个语言对应的内容,在该元件进行文本库调用,选择语言进行文本内容切换,L₀即对应语言 0,L₁即对应语言 1,以此类推,超出可在更多中输入语言编号。语言最多可设置 8 个,即 L₇。

注: 需要提前制作多语言的文本库或者图片库才能看到效果。

状态切换:对功能键或位状态/多状态元件进行状态切换,选择状态 S 即可切换到不同状态。S₀ 对应状态 0, S₁ 对应状态 1,以此类推。状态最多可设置 256 个,即 S₂₅₅。

注: 当状态切换数值超过元件最大状态时,元件以最大状态值来显示。

Zmotion[®]

ctete L	会議での 最示/隠藏図层	 □ 栅格 ☑ 元件名称 	Lo L1 L2 L3 So S1 S2 S3	E多 0 ↓ E多 0 ↓	
	显示	7	语言/状	态	
B	Basic1.bas	🖻 Hmi1	.hmi ×		•
10:	Start	te t闭	(^{WordS}ritch 切换 状态	0	
4					•

切换语言 L1 后的效果如下:

10:Start	
OFF	切换 <mark>状态0</mark>

切换状态 S1 后的效果如下:

10:Start	
开启	如何 切换状态1

第三章 组态窗口

3.1. 窗口概述

3.1.1.窗口作用

窗口是构成触摸屏画面的基本元素,也是一个重要的元素。有了窗口后,画面上的各个元件、图形、文字等信息才可以显示在触摸屏上。一般的工程文件中,包含多个窗口,所以一个功能需要建立多个窗口。

由于基本窗口的尺寸大小(分辨率)必须与触摸屏显示屏幕的尺寸相同,所以其分辨率设置也必须与 所使用的触摸屏分辨率一致。(注: RTSys 新建窗口默认分辨率为 800*480。)

组态元件超出窗口尺寸时,该元件也能正常触发。

3.2. 窗口操作

3.2.1.窗口属性

窗口属性包括窗口的类型、外观、尺寸等,可通过"属性"窗口对所选窗口进行设置修改; 打开属性窗口的方法:

1. 打开需要查看的窗口, 鼠标左键单击窗口区域, 右侧属性栏即可显示;

2. 组态视图选中要打开的窗口,单击右键,选择"窗口属性"打开,如左图;

3. 在组态文件编辑窗口,单击右键,打开"属性",如右图;

组态视图	▼ ₽ ×				
		Hm	i1.hmi	×B	Basic1.bas
		10:Stai	rt		
⊡-11: 窗口属性			属性(P)		
删除窗口		•	撤销(U)	Ctrl+Z	
拷贝窗口			▶ 重做(R)	Ctrl+Y	

组态视图能显示全部窗口,窗口可设置的属性如下图所示:

Zmotion[®]

组态视图 ×	属性	×
	▲ 基本属性	
··· 7:CharPad-Upper ··· 8:NumKevpad	窗口编号 10	
国····9:修改标志	窗口名称 Start	
①··10:Start	窗口类型 基本窗口	
■ 12:IO界面-in	垄断 False	
由:13:参数设置	▲ 外观	
■ 1:2件管理 ■ 15:轴参数设置	背景颜色 408080	
由·16:机械参数设置	会制边框 False	
■ 18:IO映射	图片来源 无	
由 19:文件编辑	公共窗口1 54:错误信息	
■ 21:直线指令	公共窗口2 无	
由·22:三点圆弧指令	透明度 100	
■ 24:输出指令	▲ 位置和尺寸	
 由·25:输出复位指令 □.25:回心回顾指令 	水平位置 0	
■ 27:绝对模式	垂直位置 0	
▲ 28:相对模式	宽度 800	
■ 31:复位设置	高度 480	
由 32:文件选择		
■ 50.7 保存提示 ■ 51:保存成功提示		
亩··52:操作提示		
■··53:酮标症小 ■·54:错误信息		
■·55:u-参数设置		
■ 50:0-又曰首,刀 ■ 57:覆盖提示		
文件视图 标签视图 组态视图	轴参数 帮助 属性	

名称	功能	说明
窗口编号	当前窗口的编号	同一项目下窗口编号不能重复
窗口名称	当前窗口的名称	/
窗口类型	可选5种窗口类型	参见"窗口类型"说明
垄断	选择是否垄断	垄断后不能操作窗口下层的元件
背景颜色	选择窗口背景颜色	/
绘制边框	选择是否绘制边框	选择 TRUE 之后,可选择边框颜色
因止立酒	川北县网山东武北县网山山进区	先添加图片才能选择, <mark>图片名称不超</mark>
凶 万 木	<u> </u>	过 26 个字符
公开窗口 1	设置当前窗口的公开窗口 1	当前窗口可以显示公共窗口的控件,
	以且当时囱口的公共囱口 I	最多可设置3个公共窗口



透明度	背景透明度	预留,暂不支持使用
水平位置	窗口显示的左上角 X 坐标	不要超出水平分辨率
垂直位置	窗口显示的左上角 Y 坐标	不要超出垂直分辨率
宽度	当前窗口的显示宽度	/
高度	当前窗口的显示高度	/

3.2.2.窗口建立

组态显示必须以一个基本窗口为底窗口,作为其他窗口的背景画面,元件需要依附窗口显示,一个组态 文件下可新建多个不同类型的窗口。建立 Hmi 文件后自带 3 个软键盘窗口及 1 个基本窗口(窗口 10),可 直接在已有的窗口 10 中进行组态编辑,也支持自行添加新建窗口。

新建窗口:菜单栏"Hmi"-"新建窗口"打开如下窗口,输入窗口号和窗口名称后确认,注意窗口号不可重复。

新建窗口	×
新建窗口号:	12
窗口名:	
确定	取消

拷贝窗口:拷贝已创建的窗口及其内容,并新建一个窗口。即可直接将已设计好的组态界面直接复制 生成另一个新窗口。

操作方法:打开"组态视图"选中需拷贝的窗口,右键单击选择"拷贝窗口"即可弹出如下图窗口,根据 需求修改新建窗口号和窗口名即可。

新建窗口	×
新建窗口号:	29
窗口名:	手动操作
确定	取消

新建窗口后,若需对窗口进行设置相关属性或尺寸位置等信息则需要打开窗口"属性"修改。打开"属性" 窗口的方法有两种:

1. 在组态视图能显示全部窗口和各窗口下的元件,选中窗口单击右键选择"窗口属性",打开元件"属性" 窗口则双击组态视图中该元件名称即可。 2. 直接单击窗口画布或单击画布中的元件便能打开对应的属性窗口。

组态视图	×	属性	×	属	性		×
⊕ 6:CharPad-Lower		▲ 基本属性		- 4	基本属性		
		窗口编号	10		元件编号	1	
■ 9:修改标志		窗口名称	Start		元件名称	Button1	
田·· 10:Start 田·· 11:手动操作		窗口类型	基本窗口		显示层次	底层	
Ⅲ 12:IO界面-in		垄断	False		有效显示	显示	
■ 13:参数设置 ■ 14:文件管理		⊿ 外观			采用有效控制	False	
■ 15:轴参数设置		背景颜色	408080		安全时间ms	0	
II: 16:机械参数设置 II: 17:IO界面-op		绘制边框	False		绑定虚拟按键	No Key	
匣·· 18:IO映射		图片来源	无		绑定物理按键	0	
■ 19:又件编辑 ■ 20:运动类型选择		公共窗口1	54:错误信息	- 4	外观		
■ 21:直线指令		公共窗口2	无		图片来源	无	
■ 22:三点圆弧指令 ■ 23:延时指令		透明度	100		绘制边框	False	
■- 24:输出指令		▲ 位置和尺寸			是否图片化	False	
■ 25:輸出复位指令 ■ 26:圆心圆弧指令		水平位置	0	- 4	标签		
■ 27:绝对模式		垂直位置	0		文本库		
■ 28:相对模式 ■ 30:浏览文件		宽度	800		格式文本(0)	手动操作	
■ 31:复位设置		高度	480		格式文本(1)	手动操作	
■·· 32:文件选择 ■·· 50:未保存提示				- 4	动作		
■ 51:保存成功提示					动作	调用函数	
□ ± 52:操作提示 □ ± 53:删除提示					松开时动作	False	
 ➡							
文件视图 标签视图 组态视	2	轴参数帮助。	尾性	轴	参数 帮助 属性		

3.2.3.窗口导入

介绍: 在当前 Hmi 项目中导入其他项目中已创建的 Hmi 窗口(可选一个或多个同时导入)。 操作步骤:

1. 在菜单栏"Hmi"中点击"导入窗口";



2. 弹出文件选择窗口,选择需要导入的窗口所在的项目文件路径,选择需要的.Hmi文件;



飛 打开				×
\leftrightarrow \rightarrow \checkmark \uparrow \square \checkmark	此电脑 > 本地磁盘 (D:) > RTSys > 測试 > 菜单栏	~ C	在 菜单栏 中搜索	Q
组织 ▼ 新建文件夹			≣ ▾ □	?
■ 文档	★ 名称 ^	修改日期	类型 大小	
🔀 图片	A Hmi1.hmi	2024/9/19 15:48	HMI 文件 810 H	КВ
🕖 音乐	*			
🛂 视频	*			
 WPS云盘 二 共享文件夹 二 広用 一 此电脑 二 本地磁盘 (C:) 一 本地磁盘 (D:) 				
文件名(N)	Hmi1.hmi		✓ Hmi File (*.hmi)打开(O) 取消	✓ í

3. 打开需要的.Hmi 文件后,弹出"窗口导入"窗口。左侧显示窗口列表,右侧显示窗口对比图。

注: 若导入的 Hmi 项目中有窗口 ID 与窗口名称均不一致,则窗口名称显示为红色,并默认勾选当前项目中无相同窗口 ID 的窗口,用户可选择是否导入/替换原窗口。

名称	新増		
6:CharPad-Lower		ABC Ba	ackspace
7:CharPad-Upper		Sym 1 2 3 4 5 6 7 8 9 0 -	•
8:NumKeypad		Tab q w e r t y u i o p	?
10:Start		Caps a s d f g h j k 1 '	-
11:报表视图		EX/CE z x c v b n m , .	Enter
12:文件选择		Clear Del Space	
15.	v		
			ackspace
		Esc ABC Ba Sym 1 2 3 4 5 6 7 8 9 0 -	ackspace
		Esc ABC Ba 57m 1 2 3 4 5 6 7 8 9 0 - Tab q x e x t y u i o p	ackspace +
		Esc ABC Ba Sym 1 2 3 4 5 6 7 8 9 0 - Tab q * e r t y u i o p Caps a s d f g h j k 1	ackspace + ?
		Esc ABC Ba Sym 1 2 3 4 5 6 7 8 9 0 - Tab q w e x t y u i<0	ackspace + ? Enter



若点击"详细对比",则会弹出"窗口对比"窗口。

窗口对比 X X	$\langle \rangle$
当前文件: C:\Users\Administrator\Desktop\Hm]	
 ▶ 6:CharPad-Lower ▶ 7:CharPad-Upper ▶ 8:WumKeypad ▶ 10:Start ▶ 11:报表视图 (≠) ▶ 12:文件选择与浏览 (≠) ▶ 12:文件选择(≠) >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>	

 勾选需要导入的窗口,点击"确定",若选择的窗口中有与原文件窗口冲突的窗口号,则会弹出以下 窗口,选择"是"则替换,选择"否"则只导入未冲突的窗口。

RTSys			×
?	11, 12. 以	上窗口已存在,是否替换?	
	是(Y)	否(N) 取	消

3.2.4.窗口调用

窗口调用有如下两种方式。

方式一:功能键调用

1. 选择"控件箱"-"功能键",新建一个功能键按钮。



2. 选中功能键打开属性,找到"动作"下拉列表,可以选择打开3种窗口类型,其中菜单窗口属于弹出窗口类型。如下左图。

⊿ 动作		⊿ 动作	
动作 调用函数	-	动作	弹出窗口
松开时动作 无动作		松开时动作	False
动作函数名 打开基本窗口		动作操作窗口	无
▲ 位置和尺寸 打开置顶窗口		随父窗口一起	无
水平位置 弹出窗口		▲ 位置和尺寸	6:CharPad-Lowe
垂直位置 关闭当前窗口		水平位置	7:CharPad-Uppe
宽度 关闭指定窗口		垂直位置	8:NumKeypad
高度 返回上一个基	本窗	宽度	10:Start
调用函数		高度	11:top
按下和松开都	调用		
	•		•
动作 输入虚拟按键		动作操作窗口	
按键执行的动作。 输入字符		需要操作的窗口编号	o
轴参数 帮助 属性 软键盘切换		轴参数 帮助 属性	

3. 选择好要打开的窗口类型后,再找到"动作操作窗口",选择要打开的窗口编号即可,如上右图。

注意: 若要打开的窗口类型与选择的窗口类型不一致,则窗口类型会被强制转换。

方式二:程序指令调用

在元件属性"动作"中选择"调用函数",调用的函数通过在 Basic 中编写程序实现,主要使用到的指令是 HMI_SHOWWINDOW 和 HMI_BASEWINDOW,指令使用方法参见指令描述。

⊿ 动作	
动作	调用函数
松开时动作	False
动作函数名	openwin

```
GLOBAL sub openwin()
HHI_SHOWWINDOW(11,0) '打开窗口11
END SUB
```

3.2.5.窗口关闭

关闭窗口有两种方式,可参考以下内容。注意:基本窗口不支持关闭!

方式一: 功能键关闭

和调用窗口一样,需要先建立一个功能键,切换窗口操作一般把功能键放在基本窗口内;关闭弹出窗口和置顶窗口操作一般把功能键放在要关闭的窗口内。

窗口关闭的动作类型主要有两个:关闭当前窗口;关闭指定窗口(须在"动作操作窗口"设置窗口编号, 如右图)



方式二:程序指令关闭

程序指令关闭窗口主要通过在 Basic 中编写程序实现,主要使用到的指令是 Hmi_CLOSEWINDOW。请查看本帮助文件第五章。

```
global sub closewindow()
HTI_CLOSEWINDOW() '参数缺省关闭当前窗口
end sub
```

3.2.6.公共窗口

介绍:用于对多个窗口指定同一个公共窗口,有利于共用的内容可以显示在当前窗口里面。

使用方法:在"窗口属性"中为当前窗口选择公共窗口。一个窗口最多可以设置3个公共窗口。

注意:添加公共窗口后,公共窗口里的元件在当前窗口仅显示(窗口背景不显示),不可操作修改元件 属性。修改元件属性须打开元件所在窗口才可。对已设置好的元件动作下载运行后动作可执行生效。



属性		×
▲ 基本属性		*
窗口编号	13	
窗口名称	关闭	
窗口类型	基本窗口	
垄断	False	
⊿ 外观		
背景颜色	F5F5F5	
绘制边框	False	
图片来源	无	
公共窗口1	11:logo	
公共窗口2	12:企业文化	
公共窗口3	13:关闭	
透明度	100	
▲ 位置和尺寸		
水平位置	0	
垂直位置	0	
宽度	800	
高度	480	-
轴参数 帮助 属性		

示例:

1. 如下图,没有添加公共窗口的显示情况,每个窗口只显示当前窗口的元件。

组态视图 • [₽] ×	Basic1.bas ⊞Hmi1.hmi ×	属性	<i>≖</i> ‡ ×
⊕ 6:CharPad-Lower	10:Start	▲ 基本属性	
··· /:CharPad-Upper ··· 8:NumKeypad	1:StaticText 主界面	窗口编号	10
- 10:Start	工 97 四	窗口名称	Start
···· 1:Staticlext1(Staticlext)	Report to Report to the second se	窗口类型	基本窗口
	打开窗口11 打开窗口12	垄断	False
		▲ 外观	
□-12:企业文化		背景颜色	F5F5F5
Ⅲ·13:关闭		绘制边框	False
		图片来源	无
		公共窗口1	无
		透明度	100
		▲ 位置和尺寸	
		水平位置	0
		垂直位置	0
		宽度	800
		高度	480
文件视图 标签视图 组态视图		小世翻□1 轴参数 帮助 属性	

2. 给 10 号窗口添加一个公共窗口 13(最多可添加 3 个),这个窗口的元件就会显示在 10 号窗口里, 但不能通过 10 号窗口对元件进行修改。



组态视图 • म ×	Basic1.bas Hmi1.hmi ×	属性	▼ ₽ ×
	■ Basici.Das ● Hmil.hmi × 10:Start 単元前日11 単元前日12 単元前12 単元前日12 単一前日12 単一前12 単一前日12 単一前12 単一前12 単一前日12 単一前12 単一前日	周1 日 ▲ 基本属性 窗口場号 窗口名称 窗口之型 差断 ▲ 外观 背景懸色 绘制边框 图片来源 公共窗口1 公共窗口2 透明度 ▲ 位置和尺寸 水平位置 垂直立置 宽度 高度	 ▼ * × × 10 Start 基本館□ False 万5F5F5 False 万元 13:关闭 元 100 0 0 800 480
文件视图 标签视图 组态视图		轴参数 帮助 属性	

3. 运行下载后在当前窗口只显示公共窗口的元件,公共窗口的背景不显示。

xplc screen			×
		主界面	
打开窗口11	打开窗口	2	
			平河网上
			大团图万

通过操作主窗口(窗口10)的元件可打开对应的窗口,在窗口10中的公共窗口元件动作也可生效。

以下面图片为例:点击打开窗口 11 和 12 之后,主窗口上显示了窗口 11 和窗口 12,公共窗口 13 的元 件可对窗口 11 和窗口 12 进行关闭操作,但无法对窗口 10 和窗口 13 进行操作。

(注:本例对窗口11和12的大小及位置做了调整,因此打开不会覆盖主窗口)



xplc screen		×
	主界面	
打开窗口11	打开窗口12	主窗口元件
Ζ	mot	ion [®] ^{窗口11}
	做最好用的运动	」控制 _{窗口12}
		公共窗口13 关闭图片

3.3. 窗口类型

依照功能与使用方式不同,Hmi 支持的窗口类型分为五种:基本窗口(Base Window)、软键盘窗口 (Keyboard Window)、弹出窗口(Pop Window)、菜单窗口(Menu Window)、置顶窗口(Top Window)。

新建窗口默认是基本窗口类型,变更类型在窗口的"属性"窗口里修改。

属性	×
▲ 基本属性	A
窗口编号	10
窗口名称	Start
窗口类型	基本窗口 🔹
垄断	基本窗口
⊿ 外观	弹出窗口
背景颜色	菜单窗口
绘制边框	软键盘窗口
图片来源	置顶窗口
公共窗口1	无
透明度	100

3.3.1.基本窗口

介绍:组态显示必须以一个基本窗口为底窗口,作为其他窗口的背景画面。新建窗口默认为基本窗口 类型,且基本窗口可通过程序或元件操作进行切换。
注意:

- 1. 基本窗口不支持关闭。
- 2. 触摸屏同一时间只能显示一个基本窗口。
- 与触摸屏连接使用时, 仅须保证 HMI 分辨率与触摸屏分辨率一致即可。各个窗口大小可任意设置, 不超出屏幕尺寸即可。

3.3.2.软键盘窗口

介绍:用于"值"元件和"字符显示"元件等需要自定义输入数据的场合。

使用方法:新建的 Hmi 文件内置有三种软键盘窗口可供选择,无需新建。可在允许软键盘编辑的元件中直接调用。窗口号分别为 6、7、8。(即:打开"值"元件的属性窗口→"可编辑"选择 True→选择"软键盘窗口号")

注意:调用软键盘窗口时,只能在显示数据支持修改的元件中使用,比如"值"、"字符显示"元件。不能用于"功能键"等元件。各类元件用法请参考<u>第四章</u>。



6号软键盘窗口

7	1:Button 2:String			3:Button
	Esc	ABC		Backspace
4	4:Button5:Button6:Button7:Butto	n8:Button9:Button10:Butt	ofl1:Buttofl2:Buttofl3:But	ttol 4: Buttol 5: Buttol 6: Butto
	Sym 1 2 3	4 5 6	7 8 9	0 - +
	17:Button 18:Butto19:Butto20	:Butto£1:Butto£2:Butto£3:	Buttof4:Buttof25:Buttof	6:Buttof7:Buttof8:Button
	Tab Q W	E R T	Y U I	0 P ?
1	29:Button β0:Buttoβ1:Butt	φβ2:Buttφβ3:Buttφβ4:Butt	oß5:Buttoß6:Buttoß7:But	ttpβ8:Buttpβ9:Buttp 40 :Buttp
	Caps A S	D F G	H J K	L , "
Į	41:BitSwitch 42:Butto43:Bu	ttp44:Buttp45:Buttp46:But	tb47:Buttb48:Buttb49:B	utto50:Butto51:Button
	EN/CH Z X	C V B	N M ,	· Enter
	52:Button Clear Del	Spac	ce	55:Butto56:Butto ← →

7 号软键盘窗口



81:String										
	ABC									
2:Button	3:Button	4:Button	5:Button							
1	2	3	-							
6:Button	7:Button	8:Button	9:Button							
4	5	6	Clear							
10:Button	11:Button	12:Button	13:Button							
7	8	9	Esc							
14:Button	15:Button	16:Button								
	0	En	ter							

8号软键盘窗口

3.3.3.弹出窗口

介绍:弹出窗口类似是对话框一样的动态弹出窗口。

使用方法: 弹出窗口需要通过程序或元件操作打开,同时也需要通过程序指令或元件(位元件/字元件等)操作关闭/切换窗口。当建立并打开多个弹出窗口时,按照调用顺序,依次显示调用的弹出窗口。

提示:

1. 当多个弹出窗口重叠在一起时,弹出窗口中的功能按键仍然能正常触发。

2. 调用出弹出窗口后,仍支持对基本窗口等其他类型窗口进行操作。

xplc screen		×
弹出窗口1 弹出窗口2	弹出窗口3	
	菜单窗口1 菜单窗口2	
	置顶窗口1 置顶窗口2 置顶窗口3	
	通上路口1	
	· · · · · · · · · · · · · · · · · · ·	
	弹出窗口3	
	可按下拖动窗口	

3.3.4.菜单窗口

介绍:菜单窗口属于弹出型窗口,通过程序或元件操作调用后弹出。调用与关闭方法与弹出窗口一致。

注意:

1. 菜单窗口弹出后,**获得最大操作权限**,此时只能对菜单窗口进行操作。当点击到非菜单窗口区域时,菜单窗口自动关闭。

2. 菜单窗口打开后的**位置是跟随对应操作的元件位置进行显示**,与菜单窗口属性中设置的位置无关, 因此设置弹出菜单窗口时注意调整好操作元件位置。

xplc screen								×
弾出窗口1	弾出窗口2	弾出窗口3						
			菜单窗□1	茲单窗□2				
					菜单窗口1		置顶窗口3	
						关闭]	

3.3.5.置顶窗口

介绍:总是在最前端显示的窗口。

使用方法:调用/关闭置顶窗口须通过程序或元件操作。(一般建议设置为一个小窗口,可以用来实现 工具条等。)有多个置顶窗口时,按照调用顺序显示,后调用的窗口在之前调用窗口的上层。

置顶窗口有两种显示方式:初始化显示和手动调用显示。

显示方式	说明
	通过菜单栏"Hmi"-"Hmi 设置"-"属性"-"起始置顶窗口"来选择某个窗
初始化显示	口是否上电即显示置顶窗口(设置该属性后,置顶窗口将覆盖起始基
	本窗口)。



	通过程序或元件操作来显示,如下图,通过功能键的动作打开和关闭
手动调用目示	置顶窗口。
于幼师用亚尔	注意:同一窗口不可由两个元件设置"打开不同类型窗口"的动作。
	(即一个窗口只能选择一种窗口类型打开)

注意:将窗口设置为置顶窗口类型后,打开该窗口将一直处于置顶状态,但对其他窗口中的元件也可 进行操作,置顶窗口不会关闭,关闭置顶窗口须通过程序或元件操作关闭。

xplc screen									×
弹出窗口1	弾出窗口2	弾出窗口3							
			菜单窗口1	菜单窗口2	_		_		
		弾き	出窗口2		Ē	置顶窗口:	. 置顶窗口2	置顶窗口3	
			置顶窗	ii □ 1					
			可按下拖;	动窗口					
					关	;闭			

第四章 组态元件

4.1.组态快捷工具

4.1.1.元件菜单

Hmi 开发主要用到窗口和组态元件,先建立不同的显示窗口,再根据需求在各个窗口上添加不同的组态元件,并结合程序开发实现相应的功能。

组态元件在平台界面最左侧"控件箱"中选择并添加到窗口(支持搜索功能),或是从菜单栏中"HMI"-"控件箱"中打开。



控件箱元件菜单:

名称	图示	说明
		矢量图形
线段/多线段/多边形	\bigcirc	根据绘制点数画出相应的线段或多边形
矩形		绘制矩形



贝塞尔曲线	と 四点绘制三阶贝塞尔曲线		
圆/圆弧/扇形	0	拖动绘制整圆/椭圆/圆弧/扇形	
刻度 经		绘制等距的间隔刻度	
表格		绘制 3×3 表格,可自定义修改表格样式	
导入	9	导入矢量图形	
		控件	
静态文本	Т	在窗口添加静态文字,可自定义设置相关属性	
静态图片		从系统/背景图片库插入图片	
位状态显示	ଦ୍ର	根据绑定的位寄存器地址的值显示对应的状态	
多状态显示		根据绑定的字寄存器地址的值显示对应的状态	
位状态设置 根据元件		根据元件动作的状态设置位寄存器地址的值	
多状态设置 123 根据元件动作的状态设置字		根据元件动作的状态设置字寄存器地址的值	
位状态切换开关		根据元件动作设置位寄存器地址的值并显示对应状态 (位状态显示和位状态设置的功能结合)	
多状态切换开关	1	根据元件动作设置字寄存器地址的值并显示对应状态 (字状态显示和字状态设置的功能结合)	
功能键	BTN	根据元件动作实现状态切换/窗口切换/软键盘切换等,只有两种显示 状态并且无法绑定寄存器	
物理按键	Ð	用于与虚拟按键/实际按键绑定后通过实际按键动作	
列表		可显示多个列表项,通过绑定的寄存器的值切换对应选项	
值 123 编辑并显示数值以及修改对应绑定寄存器的值		编辑并显示数值以及修改对应绑定寄存器的值	
字符显示		编辑或显示字符串并更改字寄存器的值	
滑块开关		通过拖拽滑块更改字寄存器数值	
定时器	Ū	定时刷新进行重复动作	

自定义	\odot	在元件区域内通过调用 basic 函数实现动态绘图		
报表视图		以表格的形式呈视多组数据,用于显示和管理报表数据		
文件浏览器		显示当前目录、以表格形式显示文件内容		
三次文件编辑器		支持 Hmi 中开发三次程序的编辑元件		
CAD A		显示矢量图形		
菜単 :三		设置菜单项,通过点击菜单项触发调用 SUB 动作		
树形图	E	以树状图形式显示所有表项,单击树节点坐标的角图标(或双击树节 点内容)可进行子树的展开/缩起,单击树节点内容进行触发动作		

4.2.组态元件通用属性

不同类型的元件,具有一些共通的属性功能,例如调用寄存器、调用函数、位置和尺寸等。

4.2.1.寄存器

大部分的元件都包含"寄存器类型"这一属性,用来与各类寄存器建立数据联系。 可通过元件控制寄存器的值,或获取寄存器的值显示,可用寄存器类型如下表。

绑定的寄存器类型	Μ
绑定的寄存器编号	0

寄存器类型	对应控制器寄存器	说明
v	龄)口N	此寄存器对应通用输入,编号 0 对应
Λ		MODBUS_BIT(10000)
v	- 绘山口 OD	此寄存器对应通用输出, 编号 0 对应
1		MODBUS_BIT(20000)
м	MODBLIS BIT	不同型号控制器的寄存器个数有区别
IVI	MODBUS_BII	掉电保持: 2048-2175
	状态寄存器 S	编号 0-999 , 编号 0 对应
S		MODBUS_BIT(30000)
		掉电保持: 0-127
	MODBUS_4X 寄存器	
D	根据数据类型	不同型号控制器的寄存器个数有区别
	INT16: MODBUS_REG	



	INT22. MODDUS LONG		
	INT32: MODBUS_LONG		
	FLOAT32: MODBUS_IEEE		
DDOT	按位读取 MODBUS_REG	违使用位出本目三元件	
D.DOT	编号=reg 号*16+dot(0-15)	「「「「「」」「「」」「「」」「「」」「「」」「」」「「」」「」」「」」「」」	
DT	TABLE	32 位浮点型数据	
T	定时器	寄存器长度 32 位,当通过 16 位指令访问时	
1	编号 0-127	自动使用低 16 位	
C	计数器	寄存器长度 32 位,当通过 16 位指令访问时	
C	编号 0-127	自动使用低 16 位	
@	Basic 定义的变量、数组	必须是 GLOBAL 全局类型才能访问	

4.2.2.动作

通过下拉菜单选择,不同元件的可选动作不同,动作的功能参见各元件的例程。

⊿ 动作	
动作	调用函数
松开时动作	False
动作函数名	

动作名	功能	说明		
元件通用动作				
无动作	无	/		
调用函数	调用 Basic 定义的 SUB 函数	函数必须是 GLOBAL 全局类型。		
	功能键 But	ton		
打开基本窗口	以基本窗口类型打开窗口	窗口号通过"动作操作窗口"选择。		
打开置顶窗口	以置顶窗口类型打开窗口			
弹出窗口	以弹出窗口类型打开窗口			
关闭当前窗口	/	关闭当前功能键所在的窗口。		
关闭指定窗口	关闭所选择的窗口	窗口号通过"动作操作窗口"选择。		
返回上一个基本窗口	打开最后一个基本窗口	打开最近一次操作的基本窗口		
按下和松开都调用函	按下调用一个函数,松开调用	需要设置"调用函数名"和"松开调用函数"		
数	另一个函数			
输入物理按键	与物理按键绑定	需要物理按键对应表。		
输入虚拟按键	设为虚拟按键	通过"虚拟按键码"选择编号。		
输入字符	输入字符串	只能在软键盘窗口内使用		
软键盘切换	切换窗口	必须在非基本类型窗口内才可以使		
		用。且只能在同类窗口间切换。		

位状态切换开关 BitSwitch / 位状态设置 BitModify				
状态设置为1	按下时,置1	开关类型		
状态设置为0	按下时,置0			
状态反转	取反,为1时变成0,为0时			
	变成 1			
状态恢复	按下时置1,松开时置0			
多状态切换开关 WordSwitch / 多状态设置 WordModify				
修改数据	写入数据到寄存器	数据值通过"动作数据"设置。		
数据增加/减少	寄存器原来值增加/减少数据	寄存器通过"寄存器类型"和编号选择。		
循环	寄存器原来值加上数据,在设	寄存器通过"寄存器类型"和编号选择。		
	置的状态之间循环切换,实现	如果寄存器原来值大于"状态数量", 会先按"原		
	周期切换数据效果	来值"-"动作数据"递减至设定的状态数量范围		
		内,即开始循环。		

4.2.3.基本属性

▲ 基本属性	
元件编号	1
元件名称	Button1
显示层次	底层
有效显示	显示
采用有效控制	False
安全时间ms	0
绑定虚拟按键	No Key
绑定物理按键	0

基本属性	描述	
元件编号	在当前窗口按添加的顺序从1开始编号	
元件名称	名称+编号。支持自定义修改	
	在多个元件叠加时,可以设置元件的显示层次	
日一日次	顶层:显示在最外层,覆盖底下元件;	
业小层认	中层: 被顶层元件覆盖,覆盖底层元件;	
	底层: 被顶层和中间层元件覆盖	
	决定该元件是否显示于界面之中	
有效显示	显示:该元件显示于界面之中;	
	不显示: 该元件不显示于界面之中;	

	仅显示,不可用:该元件显示于界面之中但是点击该元件时不会产生任何	
	改变。	
亚田古油惊剧	采用设备的寄存器控制该元件是否显示,默认 False 若选择 True,须选择	
太用有双控制	寄存器类型和编号。当寄存器置0时该元件则隐藏,非0时则显示。	
安全时间 ms	最少按键时间(ms)	
绑定虚拟按键	选择要绑定的虚拟按键码	
绑定物理按键	绑定示教盒上面的物理按键	

4.2.4.外观

⊿ 外观		⊿ 外观			
图片来源 背	背景图片	图片	十来源	背景图片库	
背景图片		背易	图片库	0\按钮\13	
绘制边框 F	False	绘制	り边框	False	
是否图片化 F	False	是2	图片化	False	

外观	描述
	部分元件支持显示图片,可在此选择图片来源,选择图片来源后可在下一行选
	择目标图片添加。
图片来源	无:不添加图片
	背景图片库:从背景图片库选用图片
	背景图片:从项目中选用图片
背景图片库	从背景图片库中选择图片。图片先添加到图片库中,再在该处进行选择
北县团山	从项目中选择图片。图片先添加到项目的"文件视图"中,再在此处进行选择
月京四月	图片命名不可超过 26 个字符
绘制边框	是否绘制元件的边框
边框颜色	选择元件边框的颜色
目不同止化	将元件显示内容图片化。例如: 当显示的文本字体过大或包含生僻字时, 导致
定百凶斤化	字体不清晰,将显示内容图片化,设为 True 之后,显示效果能改善

4.2.5.位置和尺寸

▲ 位置和尺寸	
水平位置	478
垂直位置	340
宽度	109
高度	53

位置和尺寸	描述	
水亚位罢	元件在 Hmi 窗口中放置位置的水平距离。以 Hmi 显示窗口左上角的位置	
小丁位直	为(0,0)	
	元件在 Hmi 窗口中放置位置的垂直距离。以 Hmi 显示窗口左上角的位置	
王旦心且 	为(0,0)	
宽度	设置元件的显示宽度	
高度	设置元件的显示高度	
注意: 元件位置和尺寸最好不要超出窗口的范围		

4.2.6.格式文本

用于编辑元件各状态的文字显示内容。同时也支持在元件上直接编辑文字。

对于需要设置"状态数量"的元件,"状态数量"支持自定义修改(最多支持 256 种状态)。"格式文本" 的个数随状态数量修改而增减。

⊿ 标签		
文本库		
状态数量	4	
格式文本(0)		
格式文本(1)		
格式文本(2)		
格式文本(3)		

点击格式文本后的三个点弹出"格式文本"设置窗口。如下图所示。该窗口支持内容文本输入以及提供 多种文本样式进行设置。具体使用如下:

【内容】:自定义写入文本内容,支持换行。输入内容后点击"确定"即可保存。同时支持对该文本样 式进行设置,包括对齐方式、文本颜色、对齐颜色等,样式具体设置可参考下表。注意:若使用修改 元件背景颜色及样式,须先取消该元件已应用的图片库/背景图片样式才显示。

- 【拷贝到其他状态】:将该格式文本除了"内容"项的其他设置项拷贝到其他状态的格式文本中(即: 将当前状态的文本样式应用到其余状态下),单击其他状态的格式文本可以查看到参数与该文本参数 一致。注意:仅应用于当前元件的不同状态,无法拷贝至其他元件!
- 【将该格式文本保存为默认格式文本】:将当前元件设置的文本样式保存为全局默认格式,即将当前 文本样式应用至该项目中的元件样式。勾选即生效。注意:只能应用于新创建的元件,已创建的元件 样式不改变!
- ▶ 【创建时使用默认格式文本】:新创建的元件应用已设置的默认格式文本。勾选即生效。

格式文本					×
内容 					*
_ 对齐					
水平对齐:	0		垂直对齐:	0	
_ 文本					
颜色:	RGB(0,0,0)	•	尺寸:	16	
字体:		•			
颜色					
背景颜色:	RGB(192, 192, 192)	•	填充样式:	无	•
样式颜色:	RGB(0,0,0)	•			
闪烁					
闪烁时间:	Oms	•			
跑马灯					
方向:	不移动	•	移动速度:	1	•
□ 将该格式文本保存	存为默认格式文本			拷贝到其	他状态
□ 创建时使用默认	格式文本		I	确定	取消

操作	功能	说明
内容	输入要显示的文本内容	/
水平对齐	水平对齐选项	0: 居中对齐(默认) >0: 左边对齐,值表示距离左边的距离
		<0: 右边对齐,绝对值表示距离右边的距离
垂直对齐	垂直对齐选项	0: 居中对齐(默认) >0: 上边对齐,值表示距离上边的距离 <0: 下边对齐,绝对值表示距离下边的距离
颜色	选择字体颜色	/
尺寸	选择字体大小	/
字体	选择字体	将字体文件(支持后缀为.ttf/.zft)添加到项目中
背景颜色	选择背景颜色	元件图片来源选择"无"时背景颜色才应用至元件中
填充样式	填充到背景的样式	包括六种样式,元件图片来源选择"无"时填充样式才

		应用至元件中
样式颜色	选择样式颜色	元件图片来源选择"无"时样式颜色才应用至元件中
闪烁时间	选择闪烁时间	格式文本内容闪烁显示,闪烁间隔时间为用户设置的
		时间, 0ms 为不闪烁
方向	选择跑马灯的方向	格式文本内容按照用户选择的方向移动
移动速度	选择跑马灯的移动速度	格式文本内容按照用户选择的速度移动,若方向为不
		移动则该项不会作用

4.2.7.图片库和文本库

 外观		
图片来源	背景图片库	
背景图片库	0\按钮\13 .	
绘制边框	False	
是否图片化	False	
 标签		
文本库		
状态数量	2	
格式文本(0)		
格式文本(1)		

组态元件支持添加图片及文本,图片可从"背景图片库"、"背景图片"中添加;文本则支持从"文本库" 直接调用。

使用以上途径添加前均需先创建/添加图片或文本内容。插入背景图片需先将图片添加到项目中,再在 此处选择要加载的图片名称。

图片库和文本库的建立方法参见【图片库】、【文本库】章节。

4.3. 元件介绍及使用

组态元件的基本使用介绍:

- 添加组态元件:打开 Hmi 文件,在窗口 10 或新建窗口中,从控件箱(菜单栏"HMI"→"控件箱") 单击选择某个元件,移动至窗口中单击放置即可添加成功。(若想取消已选择但未放置的元件,将 鼠标置于窗口中单击右键即可)。
- 修改元件大小/位置:单击选中元件后,将鼠标放在下图红色方块上直接拖拽即可改变大小,或由 "属性"窗口的宽度、高度设置。修改位置则直接选中元件按住鼠标左键不放拖拽即可,或由"属性" 窗口的水平位置和垂直位置调整。



 设置元件功能属性:鼠标单击选中元件,当前元件的"属性"窗口自动被打开,即可编辑元件属性, 属性编辑完成后单击回车或者鼠标点到其他地方,自动保存编辑的属性信息。(若"属性"窗口不能 被自动打开,点击菜单栏"HMI"-"属性"先打开属性窗口后,再点击元件。)

注意: "属性"中部分功能需下载运行仿真后才可看到效果!

4.3.1.线段/多线段/多边形

4.3.1.1. 线段

介绍: "线段"为直线,直线的长度和斜率由用户自定义。可直接拖拽元件定义宽高,也可在元件属性里 定义。

使用方法:点击菜单栏"HMI"→"控件箱"→"矢量图形"→"线段/多线段/多边形",确定两点即可画出一 个线段,在即将开启第二段画线操作时点击右键,在弹出的小菜单中点击"完成"即可画出线段。具体操作方 式如下图所示。提示:绘制水平直线或垂直直线时,只需按住"Crtl"键移动鼠标确定方向和长度即可。



1. 属性窗口:

届性		×	起点形状	样式1	
			起点尺寸	中	
▲ 基本腐性			终点形状	样式1	
元件编号	1			173401	
			终点尺寸	中	
元件名称	Line1		4 位置和尺寸		
显示层次	底层		TELEVIC J		
			水平位置	205	
有效显示	显示			149	
采用有效控制	False			115	
			宽度	128	
<i>▲ 9</i> 下X92			宫府	21	
当前颜色	000000			51	
					Ť
线宽	1				
线段类型	默认		轴参数 帮助 属性		



2. 属性说明:

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会出现下方的三个参数
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型 (有效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号 (有效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
当前颜色	选择线段颜色	/
线宽	线段的宽度	默认宽度为1,最大为20
线段类型	线段的样式	下拉列表选择实线或虚线等类型
起点形状	多种形状下拉列表选择	/
起点尺寸	起点形状的尺寸大小	选了起点形状才有效
终点形状	多种形状下拉列表选择	/
终点尺寸	终点形状的尺寸大小	选了终点形状才有效
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

4.3.1.2. 多线段

介绍:由多个连续的线段组合而成的非封闭图形,即为多线段。

使用方法:点击菜单栏"HMI"→"控件箱"→"矢量图形"→"线段/多线段/多边形",连续在不同位置点击 多个点画出多个连续的线段,点击鼠标右键会弹出一个小窗口,点击"完成"即可画出多线段。画出的线段可 以拥有任意数目转角,可以画成任意图形。提示:绘制水平直线或垂直直线时,只需按住"Crtl"键移动鼠标 确定方向和长度即可。 注意:

- 1. 线段确定后各点的相对位置不能修改,只支持对整体形状进行缩放;
- 2. 若起始点和结束点重合在同一个坐标,但不能认为其是封闭图形,无法填充由线定义的区域。



1. 属性窗口:

属性	:	×	线宽	1
4 基本属性			线段类型	默认
			▲ 位置和尺寸	
元件编号	1		水亚位罢	112
元件名称	Polyline1	Ŀŀ	小十位且	112
目子巨次			垂直位置	141
业小坛人			宽度	159
有效显示	显示	Ŀŀ		F 4
采用有效控制	False	-	同股	•
▲ 外观				
当前颜色	000000		轴参数 帮助 属性	

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会出现下方的三个参数
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
当前颜色	选择线段颜色	/



线宽	线段的宽度	/
线段类型	线段的样式	/
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

4.3.1.3. 多边形

介绍:由多条连续线段组合成的封闭图形,支持填充背景颜色。"多边形"的形状需要用户手绘。

使用方法:点击菜单栏"HMI"→"控件箱"→"矢量图形"→"线段/多线段/多边形",单击左键确定第一个 点位置开始绘制,当绘制到 n 个点的时候,点击鼠标右键选择"闭合"图形自动封闭。(即:若画一个四边形, 则确定 4 个点后,最后一段线段无需手动绘制,选择"闭合"即可)。提示:绘制水平直线或垂直直线时,只 需按住"Crtl"键移动鼠标确定方向和长度即可。

注意:形状确定后各点的相对位置不能修改,只支持对整体形状进行缩放。



1. 属性窗口:

屋性		×	线宽	1	
			线段类型	默认	
▲ 基本腐性		-1	填充	False	
元件编号	1	- 11	▲ 位置和尺寸		
元件名称	Polygon1		水平位罟	87	_
显示层次	底层		新作位日	151	
有效显示	显示			110	
采用有效控制	False	- 11		77	_
⊿ 外观		- 11			
当前颜色	000000		轴参数 帮助 属性		

属性	功能	说明
元件编号	/	/

元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会出现下方的三个参数
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
当前颜色	选择线段颜色	/
线宽	线段的宽度	/
线段类型	线段的样式	/
填充	选择是否填充颜色	填充整个元件
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

4.3.2.矩形

介绍: 创建一个矩形, "矩形"是一个可以填充背景颜色的闭合对象。

使用方法:点击菜单栏"HMI"→"控件箱"→"矢量图形"→"矩形",确定单点位置后按住鼠标左键不放,拖拽至合适长度、宽度松开即可。注意:形状确定后各点的相对位置不能修改,只支持对整体形状进行缩放。

2:Rect		

1. 属性窗口:

属性	×	(组	設类型		默认	
▲ 基本属性		¥	俗		0	
元件编号	4	塡	铳		False	
	Rect4	_ ▲ 亿	置和尺	র্য	-	
显示层次	底层	7	、平位置		590	
有效显示	显示	Ŧ	直位置		327	
采用有效控制	False	贯	腹		158	
⊿ 外观		高	腹		117	
当前颜色	000000					
线宽	1	轴参数	女 帮助	属性		

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会出现下方的三个参数
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
当前颜色	选择线段颜色	/
线宽	线段的宽度	/
线段类型	线段的样式	/
半径	倒角半径	设置四个角是否要倒圆角
填充	选择是否填充颜色	填充整个元件
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

4.3.3.贝塞尔曲线

介绍:基于二维平面的矢量数学曲线。

使用方法:点击菜单栏"HMI"→"控件箱"→"矢量图形"→"贝塞尔曲线",绘制时确定四个点位置即可自动创建贝塞尔曲线。

注意:形状确定后各点的相对位置不能修改,只支持对整体形状进行缩放。

Rezier

1. 属性窗口:

属性		×	线宽	1
			线段类型	駅 代人
⊿ 基本属性		<u>*</u>		
二件位日	1		▲ 位置和尺寸	
フレコーー細ーー	1		水平位罟	69
元件名称	Bezier1			
			垂直位置	177
显示层次	底层		安府	102
有效显示	周示			105
HXXIII			高度	40
采用有效控制	False			•
▲ 外观				
当前颜色	000000		轴参数 帮助 属性	

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会出现下方的三个参数
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有	选择寄存器编号	寄存器值为0时不显示,非0时显示



效控制为 True)		
当前颜色	选择线段颜色	/
线宽	线段的宽度	/
线段类型	线段的样式	/
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

4.3.4.圆/圆弧/扇形

4.3.4.1. 圆

介绍: "圆"是一个可以填充背景颜色的闭合对象。支持对圆/椭圆设置线条类型、颜色及填充颜色和样 式等。

使用方法: 点击菜单栏"HMI"→"控件箱"→"矢量图形"→"圆/圆弧/扇形"。鼠标移动至窗口中单击确定 图形所在窗口中位置,再按如下方式进行绘制。

绘制椭圆只需拖动鼠标确认大小后单击鼠标右键点击"完成"(无需另外确定位置点)。

<u>绘制整圆</u>则需在鼠标拖动确认大小时持续按下 Ctrl 键,确认大小后松开鼠标和按键,接着单击鼠标右键点击"完成"(无需另外确定位置点)。

注意:形状确定后各点的相对位置不能修改,只支持对整体形状进行缩放。由于圆/椭圆的弧线分布着 无数点,此时调整宽高可改变形状(如:绘制整圆后调整大小即可变换成椭圆)



1. 属性窗口:



属性		×	贫	宽	1
			斜	 	默认
▲ 埜平腐壮			ţ	铳	False
元件编号	5		- K	置和尺寸	
元件名称	Ellipse5		7	(平位置	583
显示层次	底层		ヨ	E直位置	225
有效显示	显示		赁	問度	120
采用有效控制	False		릙	夏	120
▲ 外观					
当前颜色	000000		轴参数	牧 帮助 属性	

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会出现下方的三个参数
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
当前颜色	选择线段颜色	/
线宽	线段的宽度	/
线段类型	线段的样式	/
填充	选择是否填充颜色	填充整个元件
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

4.3.4.2. 圆弧

介绍:圆上任意两点间的曲线即圆弧。

使用方法:点击菜单栏"HMI"→"控件箱"→"矢量图形"→"圆/圆弧/扇形"。先拉一个矩形框确定大小,再确定圆弧的起点和终点,按起点到终点顺时针方向绘制圆弧,绘制完成后点击"完成"即可绘制成功。

注意:形状确定后各点的相对位置不能修改,只支持对整体形状进行缩放。



1. 属性窗口:

属性	×	:	线宽	1
▲ 基本属性			线段类型	默认
元件编号	1	4	位置和尺寸	
元件名称	Arc1		水平位置	531
显示层次	底层		垂直位置	266
有效显示	显示		宽度	18
采用有效控制	False		高度	78
▲ 外观				
当前颜色	000000	轴	参数帮助属性	

属性	功能	说明
元件编号	/	/
元件名称	/	/
		顶层:显示在最外层,覆盖底下元件;
显示层次	选择元件显示层次	中层: 被顶层元件覆盖,覆盖底层元件;
		底层:被顶层和中间层元件覆盖(默认)。
		显示:该元件显示于界面之中;
有效显示	进权一件具不且二	不显示: 该元件不显示于界面之中;
	远拜几件定省並小	仅显示,不可用:该元件显示于界面之中但
		是点击该元件时不会产生任何改变。
亚田右劫掠到	通过安友盟控制元件具不目二	默认 False,选择 True 才会出现下方的三个
木用有X1空刺	迪 过可付益控制几件走音亚小	参数
有效的设备编号	- 近夕 沪 旦	
(有效控制为 True)		款认 local
有效的寄存器类型(有	业权安方现米刑	夕孙安友盟下拉列主进权
效控制为 True)	远拜 讨 付	多件句什奋丫担刘衣远挥

有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
当前颜色	选择线段颜色	/
线宽	线段的宽度	/
线段类型	线段的样式	/
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

4.3.4.3. 扇形

介绍:圆弧与两条半径组成的闭合对象。

使用方法:点击菜单栏"HMI"→"控件箱"→"矢量图形"→"圆/圆弧/扇形"。先拉一个矩形框确定大小,再确定扇形的起点和终点,确定两点位置后右键点击"扇形"即可按顺时针方向生成扇形区域。

注意:形状确定后各点的相对位置不能修改,只支持对整体形状进行缩放。



1. 属性窗口:

属性		×	线段类型	E	默认	
▲ 基本属性		4	填充	F	False	
元件编号	1		▲ 位置和尺寸			
元件名称	Sector1		水平位置	1	171	
显示层次	底层		垂直位置	1	160	
有效显示	显示		宽度	5	57	
采用有效控制	False		高度	7	79	
⊿ 外观						•
当前颜色	000000					
线宽	1		1111 1111 1111 1111 1111 1111 1111	E		

属性	功能	说明
元件编号	/	/

元件名称	/	/	
		顶层:显示在最外层,覆盖底下元件;	
显示层次	选择元件显示层次	中层: 被顶层元件覆盖, 覆盖底层元件;	
		底层: 被顶层和中间层元件覆盖(默认)。	
		显示: 该元件显示于界面之中;	
有效显示	进择元件是不显示	不显示: 该元件不显示于界面之中;	
	龙叶九叶龙百亚小	仅显示,不可用:该元件显示于界面之中但	
		是点击该元件时不会产生任何改变。	
④ 平田右湖坊制	通过客方哭坊制元件具不且元	默认 False,选择 True 才会出现下方的三个	
术 用有双红刺	远 过可行确定时几日还有亚小	参数	
有效的设备编号	设久编号		
(有效控制为 True)	以宙狮与		
有效的寄存器类型(有		名种岑左哭下拉列表进择	
效控制为 True)	龙 汗可行研天主	夕竹可行奋于拉汋农选许	
有效的寄存器编号(有	选择客友哭编号	客存哭值为0时不显示。非0时显示	
效控制为 True)			
当前颜色	选择线段颜色	1	
线宽	线段的宽度	/	
线段类型	线段的样式	/	
填充	选择是否填充颜色	填充整个元件	
水平位置	元件的水平起始位置	不要超出水平分辨率	
垂直位置	元件的垂直起始位置	不要超出垂直分辨率	
宽度	元件的宽度	/	
高度	元件的高度	/	

4.3.5.刻度

介绍: 绘出多行的等高线条,用作元件对齐等辅助标记。

使用方法:点击菜单栏"HMI"→"控件箱"→"矢量图形"→"刻度"。鼠标移动至窗口合适位置单击放置即 可。支持调整整体大小及段落数,同时支持切换线条方向(可在"属性"窗口修改)。

注意:行间距不支持手动修改,可通过对该元件进行缩放或改变元件宽度和高度实现改变行间距的效果。

2:Scale		

1. 属性窗口:

属性	×	线段类型	默认
▲ 基本属性		刻度样式	水平方向
	1	内部段数	5
一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一	Scale1	▲ 位置和尺寸	
	 库巨	水平位置	26
五次見示		垂直位置	70
	False	宽度	201
木川有 X 近 利	raise	高度	151
· 717496			Ŧ
当前颜色	000000		
线宽	1	轴参数 帮助 属性	

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会出现下方的三个参数
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有	选择寄存器编号	寄存器值为0时不显示,非0时显示



效控制为 True)		
当前颜色	选择线段颜色	/
线宽	线段的宽度	/
线段类型	线段的样式	/
刻度样式	选择刻度纵向或横向显示	/
内部段数	设置内部的刻度段数	默认为5
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

例:将元件放置于刻度线条之间,起到对齐布局的作用。

2:Scale B:Value 123	
	4:Button
	5:String ABC

4.3.6.表格

介绍: 绘制一个网格型表项, 可将元件放置于表格中, 主要用作元件对齐布局。

使用方法:点击菜单栏"HMI"→"控件箱"→"矢量图形"→"表格"。鼠标移动至窗口合适位置单击放置即可。

[元件放置方法]:将元件拖到表格处自动吸附对齐,对齐方式可在属性中修改。

注意:

1. 一个表格只能设置一种对齐方式。

2. 需布局放置的元件大小必须小于表格内部单格的大小,才有自动对齐吸附效果。

3. 使用时需要注意元件的显示层次的设置。

2:Table	



1. 属性窗口:

属性	>	<		_	- 11
4 基本属性		<u>ــــــــــــــــــــــــــــــــــــ</u>	网格颜色	000000	
	1		网格线宽	1	
			网格线段类型	默认	
元件名称	Table1		埴充	False	11
显示层次	底层	11-		尾山辺せを	-11
有效显示	显示				-11
采用有效控制	False	14	位宜和尺寸		-11
4 夕下观		U	水平位置	72	
·····································	2		垂直位置	62	
	3		宽度	200	
列致	3		高度	150	-
外框颜色	000000				•
外框线宽	1				
外框线段类型	默认	轴	参数帮助属性		

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会出现下方的三个参数
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
行数	设置表格行数	/
列数	设置表格列数	/
外框颜色	设置表格外框颜色	默认黑色
外框线宽	设置表格外框线宽	/
外框线段类型	设置表格外框线段类型	/
网格颜色	设置表格内框线颜色	默认黑色
网格线宽	设置表格内框线线宽	/

网格线段类型	设置表格网格线段类型	/
填充	选择是否填充颜色	/
		9种对齐方式: 左上对齐/中上对齐/右上对齐/
对齐方式	设置表格内容的对齐方式	左中对齐/居中对齐/右中对齐/左下对齐/中下
		对齐/右下对齐
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

例:将元件调整至合适大小后,直接拖拽到表格内部中,会有自动吸附对齐的效果,通过设置对齐方式 使得元件在表格内部按对应方式摆放。

2:Table	3:Value 123	4:StaticText Text

4.3.7.导入

介绍:导入矢量图形。

使用方法:点击菜单栏"HMI"→"控件箱"→"矢量图形"→"导入"。在建立时先确定方框的位置,弹出文件选择窗口,打开系统盘的矢量文件填充到元件内显示。(目前支持导入的图形格式为:.dxf/.ai/.plt/.dst/)。 支持修改矢量图形的颜色、线宽、背景颜色等。



1. 属性窗口:



属性		×	线段类型	默认	
▲ 基本属性			填充	False	
元件编号	1		▲ 位置和尺寸		
元件名称	Vect1		水平位置	96	
显示层次	底层		垂直位置	139	
有效显示	显示	- 1	宽度	200	
采用有效控制	False	- 1	高度	150	
▲ 外观					-
当前颜色	000000				
线宽	1		轴参数 帮助 属性		

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会出现下方的三个参数
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
当前颜色	选择线段颜色	/
线宽	线段的宽度	默认宽度为1
线段类型	线段的样式	下拉列表选择实线或虚线等类型
填充	是否填充颜色	默认 False 不填充
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

4.3.8.静态文本

介绍: 输入并显示文本内容。支持显示单行或多行文字

使用方法:点击菜单栏"HMI"→"控件箱"→"静态文本"。选中后将矩形框放至合适位置,直接输入文本即可。

注意:

- 元件直接输入文本不支持直接换行。文本换行须在"属性"→"格式文本"的"内容"编辑处进行换行。
 格式文本使用具体可参见上述"<u>格式文本</u>"章节。
- 2. 文本较多的时候,注意调整元件尺寸,元件太小可能导致显示不全。

50510120F00F	
Tov	+
ICA	L

1. 属性窗口:

尾性		×	∡ 标签		
/两工		^	文本库		
▲ 基本属性			格式文本	Text	
元件编号	1		▲ 位置和尺寸		
元件名称	StaticText1		水平位置	171	
显示层次	底层		垂直位置	233	
有效显示	显示		宽度	100	
采用有效控制	False		高度	32	
▲ 外观					Ψ
是否图片化	False		轴参数 帮助 属性		

属性	功能	说明
元件编号	/	/
元件名称	/	/
		顶层:显示在最外层,覆盖底下元件;
显示层次	选择元件显示层次	中层: 被顶层元件覆盖, 覆盖底层元件;
		底层: 被顶层和中间层元件覆盖(默认)。
		显示:该元件显示于界面之中;
古社日二	<u>冲扫二件日不日二</u>	不显示: 该元件不显示于界面之中;
有双亚小	见拜儿针定百亚小	仅显示,不可用:该元件可正常显示,但点
		击该元件时动作不生效
或田子光检查。 这田子光检查。 圣社安士田拉西二位日本	通过安方盟掠制三册具不且三	默认 False,选择 True 才会出现下方的三个
木川 円 双 12 问	— 地 过可任奋狂刺儿什定百亚小	参数

有效的设备编号	迎友护旦		
(有效控制为 True)	以	苏氏 local	
有效的寄存器类型(有	<u> </u>	夕孙安左盟下拉列圭进权	
效控制为 True)	此件可行命失空	多們句仔奋下拉列衣选择 	
有效的寄存器编号(有	进权安方现护旦	家友照佐为0叶不且二,北0叶目二	
效控制为 True)	远伴 讨 仔 奋 绷 丂	奇仔	
是否图片化	元件变为图片的形式	默认 False	
文本库	文本库的名称	不设置文本库显示格式文本	
	二件立大日二	打开格式文本设置窗口设置元件要显示的文	
恰式义平	兀忤义 华並示	本	
水平位置	元件的水平起始位置	不要超出水平分辨率	
垂直位置	元件的垂直起始位置	不要超出垂直分辨率	
宽度	元件的宽度	/	
高度	元件的高度	/	

4.3.9.静态图片

介绍:导入并显示静态图片。

使用方法:点击菜单栏"HMI"→"控件箱"→"静态图片"。在窗口中添加"静态图片"元件,在元件"属性" 中添加要显示的图片,可选择图片库己有的图片,或选择增加到文件视图的背景图片。

背景图片库:图片先添加到图片库中,然后在背景图片库中选择。

背景图片:图片先添加到项目的"文件视图"中,再在背景图片中选择图片。



在使用视觉的时候,该元件支持显示视觉拍摄的图片,在"背景图片"中选择视觉通道,支持四个视觉通道: @ZV0、@ZV1、@ZV2、@ZV3,使用前先将图片存入视觉锁存通道才能获取显示。

1. 属性窗口:



属性		×	背景图片		
▲ 基本属性			使用图片原始尺寸	寸 False	
元件编号	1		▲ 位置和尺寸		
元件名称	StaticPic1	- 11	水平位置	79	_
显示层次	底层	- 11	垂直位置	124	
有效显示	显示	- 11	宽度	200	
采用有效控制	False		高度	150	
⊿ 外观					
图片来源	背景图片		轴参数 帮助 属性		

2. 属性说明:

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会出现下方的三个 参数
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
图片来源	无、背景图片库或背景图片	无图片、背景图片库或背景图片中选择
背景图片	背景图片选择	图片来源设置为选择背景图片显示
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示
使用图片原始尺寸	是否使用图片原始尺寸	False 图片自适应元件大小, True 元件适应 图片原始尺寸
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

例一:读取视觉通道图片

1. 先将图片存放至控制器或仿真器的 flash 文件夹中。

2. Basic 文件读取图片并保存到视觉通道0

GLOBAL ZVOBJECT Image

ZV_READIMAGE(Image,"test.bmp",0)

ZV LATCH(Image,0)

'从默认路径读取图片

age,0)

'图片存入视觉锁存通道0

3. 新建"静态图片"元件,属性的图片来源选择背景图片,再手动输入@ZV0即可读取该图片。

▲ 外观		
图片来源	背景图片	
背景图片	@ZV0	
使用图片原始尺寸	False	

显示效果:

图片元件显示图片,图片大小默认由元件大小决定,可选择使用图片原始尺寸。



4.3.10. 位状态显示

介绍:根据寄存器的位状态0或1来显示格式文本0或1。

使用方法:点击菜单栏"HMI"→"控件箱"→"位状态显示"。将元件放至合适位置,在该元件"属性"窗口选择需绑定的寄存器类型及地址。并在不同状态的格式文本输入显示内容。切换显示状态需通过切换寄存器状态实现。

切换寄存器状态的方式有三种:

- ▶ 点击菜单栏"HMI"→"语言/状态"→"S0/S1/S2…"即可实现状态切换;
- ▶ 在[命令与输出]窗口的"在线命令"行,输入绑定的寄存器地址及状态值,点击"发送"即可(如输入: MODBUS_BIT(0)=1);
- 通过"动作"设置调用子函数, Basic 子函数写入相关寄存器状态切换的程序, 按下元件调用子函数 程序可实现切换。

注意: 若无设置相关动作,则无法直接按下元件来切换显示状态。

提示:选择位寄存器时,寄存器值为0显示格式文本0,寄存器值为1显示格式文本1。若选择的寄存器不是位寄存器,那么寄存器值为0显示格式文本0,寄存器值不为1显示格式文本1,多个位的显示使用多状态。



1. 属性窗口:

属性		×	是否图片化	False	
▲ 基本属性			⊿ 标签		_
元件编号	3		文本库		
元件名称	BitState3		格式文本(0)		
显示层次	底层		格式文本(1)		
有效显示	显示		⊿ 动作		_
采用有效控制	False		点击调用函数		
安全时间ms	0		▲ 位置和尺寸		
绑定的设备编号	本地		水平位置	252	
绑定的寄存器类型	Μ		垂直位置	186	
绑定的寄存器编号	0		宽度	109	
⊿ 外观			高度	49	
图片来源	背景图片库				•
背景图片库	0\按钮\9				
绘制边框	False		轴参数 帮助 属性		

属性	功能	说明
元件编号	/	/
元件名称	/	/
		顶层:显示在最外层,覆盖底下元件;
显示层次	选择元件显示层次	中层:被顶层元件覆盖,覆盖底层元件;
		底层:被顶层和中间层元件覆盖(默认)。
		显示:该元件显示于界面之中;
古 故 日 二	<u>此权之供且不日二</u>	不显示: 该元件不显示于界面之中;
有双亚小	远拜几件定首並小	仅显示,不可用:该元件可正常显示,但点
		击该元件时动作不生效
亚田古动惊剧	通过安方限按制三件目不日三	默认 False,选择 True 才会出现下方的三个
米用有效控制	参数,通过寄存器控制元件是否显示	
有效的设备编号		₩4-21 11
(有效控制为 True)	以	款认 local

有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
安全时间 ms	最少按键时间	单位 ms
绑定的设备编号	指定设备	默认 local
绑定的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择
绑定的寄存器编号	设置寄存器的编号	通过获取寄存器不同数值,控制元件不同状 态
图片来源	无、背景图片库或背景图片	无图片、背景图片库或背景图片中选择
背景图片	背景图片选择	图片来源设置为选择背景图片显示
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示
绘制边框	选择是否绘制边框	/
是否图片化	元件变为图片的形式	默认 False
文本库	文本库的名称	不设置文本库显示格式文本
格式文本 0/1	打开格式文本设置窗口设置元 件要显示的文本	寄存器值为0显示文本0,寄存器值不为0 时显示文本1
点击调用函数	按键按下时调用函数	下拉框选择可以调用的函数名
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

例一:根据寄存器的不同状态来显示不同的文本

1. 选择绑定的寄存器类型和编号, M0 对应 MODBUS_BIT(0);

2. 在格式文本中填入不同状态对应的文本。
| Ŧ | 属性 | ▼ ₽ × |
|------------|-----------|------------------|
| | ▲ 基本属性 | |
| | 元件编号 | 1 |
| | 元件名称 | BitState1 |
| | 显示层次 | 底层 |
| | 有效显示 | 显示 |
| | 采用有效控制 | False |
| | 安全时间ms | 0 |
| | 绑定的设备编号 | 本地 |
| 1:BitState | 绑定的寄存器类型 | M |
| | 绑定的寄存器编号 | 0 |
| | ⊿ 外观 | |
| | 图片来源 | 背景图片库 |
| | 背景图片库 | 0\按钮\9 |
| | 绘制边框 | False |
| | 是否图片化 | False |
| | ⊿ 标签 | |
| | 文本库 | |
| | 格式文本(0) | 关闭 |
| | 格式文本(1) | 开启 |
| | ⊿ 动作 | |
| | 点击调用函数 | |
| | ▲ 位置和尺寸 | |
| | 水平位置 | 704 |
| | 垂直位置 | 162 |
| | 宽度 | 81 |
| | 高度 | 28 |
| | | |
| • | 轴参数 帮助 属性 | |

实现效果:

当 MODBUS_BIT(0)=0 时,元件显示"关闭",当 MODBUS_BIT(0)=1 时,元件显示"开启"。 提示:切换寄存器状态方法可参考本节上方内容。

例二:调用 SUB 函数

1. 在 Basic 里编辑好 Hmi 要调用的全局 SUB 子函数。

2. 在元件属性的"点击调用函数"处选择上一步编辑好的 SUB 子函数名称。

Hmi1.hmi × Basic1.bas × 属性 • 4 × Ŧ 1 2 3 **global sub sub1**() 10:Start ▲ 基本属性 print modbus_bit(0) end sub 1 元件编号 元件名称 BitState1 显示层次 底层 有效显示 显示 False 采用有效控制 安全时间ms 0 绑定的设备编号 本地 绑定的寄存器类型 М 绑定的寄存器编号 0 ▲ 外观 背景图片库 图片来源 0\按钮\9 背景图片库 False 绘制边框 是否图片化 False ⊿ 标签 文本库 格式文本(0) 函数调用 ... 格式文本(1) 动作 点击调用函数 sub1 位置和尺寸 水平位置 46 垂直位置 150 81 宽度 28 高度 4 Þ • 轴参数 帮助 属性

Zmotion

实现效果:

当元件被按下时,调用 Basic 的 SUB 子函数执行,每按下一次调用一次函数。

例三: 位切换功能

1. 寄存器选择 D.DOT, 寄存器编号=reg 编号*16+dot(0-15), 0 到 15 即是 reg 的 0 到 15 位选择(即可 设置 MODBUS REG 寄存器具体某 bit 位的状态值);

2. 在格式文本中填入不同状态对应的文本。



Ŧ	属性	▼ ₽ ×
	▲ 基本属性	^
	元件编号	1
	元件名称	BitState1
	显示层次	底层
	有效显示	显示
	采用有效控制	False
	安全时间ms	0
	绑定的设备编号	本地
	绑定的寄存器类型	D.DOT
	绑定的寄存器编号	162
f:BitState	▲ 外观	
■ D(10)的第2位为0 ■	图片来源	背景图片库
	背景图片库	0\按钮\9
	绘制边框	False
	是否图片化	False
	∡ 标签	
	文本库	
	格式文本(0)	D(10)的第2位为0
	格式文本(1)	D(10)的第2位为1
	⊿ 动作	
	点击调用函数	
	▲ 位置和尺寸	
	水平位置	631
	垂直位置	206
	宽度	153
	高度	46
		•
•	轴参数 帮助 属性	

实现效果:

162=10*16+2,即当D(10)第二位赋予1时,元件显示"D(10)的第2位为1"。

4.3.11. 多状态显示

介绍:根据寄存器的不同状态来显示不同的文本或是调用 SUB 函数。

使用方法:点击菜单栏"HMI"→"控件箱"→"多状态显示"。将元件放至合适位置,在该元件"属性"窗口 选择需绑定的寄存器类型及地址。并在不同状态的格式文本输入显示内容。切换显示状态需通过切换寄存 器状态实现。

切换寄存器状态的方式有三种:

▶ 点击菜单栏"HMI"→"语言/状态"→"S0/S1/S2…"即可实现状态切换;

- ▶ 在[命令与输出]窗口的"在线命令"行,输入绑定的寄存器地址及状态值,点击"发送"即可(如输入: MODBUS_REG(0)=1);
- ▶ 通过"动作"设置调用子函数, Basic 子函数写入相关寄存器状态切换的程序,按下元件调用子函数 程序可实现切换。

注意:

- 1. 若无设置相关动作,则无法直接按下元件来切换显示状态。
- 2. 可显示的文本数量根据"状态数量"自定义设置,数量范围 1-256。
- 建议选择字寄存器控制,默认 D0,对应寄存器值为 0 时,显示格式文本 0;值为 1 时,显示格式 文本 1;依此类推,一一对应显示。



1. 属性窗口:

属性		×	是否图片化 False	1
∡ 基本属性			⊿ 标签	
元件编号	4		文本库	
元件名称	WordState4		状态数量 2	
显示层次	底层		格式文本(0)	
有效显示	显示		格式文本(1)	
采用有效控制	False		⊿ 动作	
安全时间ms	0		点击调用函数	
绑定的设备编号	本地		▲ 位置和尺寸	
绑定的寄存器类型	D		水平位置 589	
绑定的寄存 <mark>器</mark> 编号	0		垂直位置 300	
⊿ 外观			宽度 109	
图片来源	背景图片库		高度 53	-
背景图片库	0\按钮\13			
绘制边框	False		轴参数 帮助 属性	

2. 属性说明:

属性	功能	说明
元件编号	/	/
元件名称	/	/
見一日次	选择元件员元目次	顶层:显示在最外层,覆盖底下元件;
亚小云闪	边拜几什亚小层伏	中层: 被顶层元件覆盖,覆盖底层元件;

		底层: 被顶层和中间层元件覆盖(默认)。
		显示: 该元件显示于界面之中;
古 故 日 二		不显示: 该元件不显示于界面之中;
有效亚示	远挥兀忤定省亚小	仅显示,不可用:该元件可正常显示,但点
		击该元件时动作不生效
可田古社校制	通过安方理论则二件目不日二	默认 False,选择 True 才会显示以下三个参
太用有效控制	— 通过苛仔奋控制几件定否並小	数,通过寄存器控制元件是否显示
有效的设备编号	设备行行	
(有效控制为 True)		A IN IOCAI
有效的寄存器类型(有		名种客方哭下拉列表进择
效控制为 True)	远许可行研关王	少件可行研门证则农选计
有效的寄存器编号(有	进择客在哭编号	客在哭值为0时不显示 非0时显示
效控制为 True)	2014可行带领 7	
安全时间 ms	最少按键时间	单位 ms
绑定的设备编号	指定设备	默认 local
绑定的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择
- - - - - - - - - - - - - - - - - - -	设置宪友婴的绾号	通过获取寄存器不同数值,控制元件不同状
外足们可行破病 了	以且可行研切拥 5	态
图片来源	无、背景图片库或背景图片	无图片、背景图片库或背景图片中选择
背景图片	背景图片选择	图片来源设置为选择背景图片显示
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示
绘制边框	选择是否绘制边框	/
是否图片化	元件变为图片的形式	默认 False
文本库	文本库的名称	不设置文本库显示格式文本
状态数量	元件的状态数量(1-256)	可显示多个状态
故书文本 0/1	打开格式文本设置窗口设置元	故式立大的粉是山北大粉是冲空
格式又本 0/1	件要显示的文本	俗式又平的数里田扒芯数里伏足
点击调用函数	按键按下时调用函数	下拉框选择可以调用的函数名
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

例一:根据寄存器的不同状态来显示不同的文本

1. 选择寄存器类型和编号,默认字寄存器 D0,即 MODBUS_REG(0);

2. 选择状态数量,在"格式文本"中填入不同状态对应的文本。



🖻 Hmi1.hmi 🛛 🔹	属性		┯ џ
	▲ 基本属性		
	元件编号	1	
	元件名称	WordState1	
	显示层次	底层	
	有效显示	显示	
	采用有效控制	False	
	安全时间ms	0	
	绑定的设备编号	本地	
	绑定的寄存器类型	D	
L:WordState	绑定的寄存器编号	0	
自动	⊿ 外观	I	
	图片来源	背景图片库	
	背景图片库	0\按钮\13	
	绘制边框	False	
	是否图片化	False	
	⊿ 标签		
	文本库		
	状态数量	3	
	格式文本(0)	自动	
	格式文本(1)	手动	
	格式文本(2)	待机	
	⊿ 动作		
	点击调用函数		
	▲ 位置和尺寸	I	
	水平位置	668	
	垂直位置	215	
	宽度	109	
	如会物 邦助 居姓		

实现效果:

- 当 MODBUS_REG(0)=0 时,元件显示"自动";
- 当 MODBUS_REG(0)=1 时,元件显示"手动";
- 当 MODBUS_REG(0)=2 时,元件显示"待机"。
- 提示: 切换寄存器状态方法可参考本节上方内容。

例二:调用 SUB 函数

- 1. 在 Basic 文件中,编写一个全局的 SUB 函数;
- 2. 在元件属性"动作"选择调用的 sub 函数。

文件视图	▼ ∓ ×	Basic1.bas ×	🖻 Hmi1.hmi
文件名	自动运行	1 2 □global sub s 3 print [#] 4 end sub 5	ub1 () 引用函数″
B Hmi1.hmi	× Ŧ	属性	<u>≁</u> џ ×
		▲ 基本属性	
		元件编号	1
		元件名称	WordState1
		显示层次	底层
		有效显示	显示
		采用有效控制	False
		安全时间ms	0
		绑定的设备编号	本地
		绑定的寄存器类型	D
L:Word	State	绑定的寄存器编号	0
- 1周)	用函数	▲ 外观	
		图片来源	背景图片库
		背景图片库	0\按钮\13
		绘制边框	False
		是否图片化	False
		⊿ 标签	
		文本库	
		状态数量	2
		格式文本(0)	调用函数
		格式文本(1)	调用函数
		⊿ 动作	
		点击调用函数	sub1
		▲ 位置和尺寸	
		水平位置	668
		垂直位置	215
		宽度	109
		高度	53
4		轴参数 帮助 属性	

实现效果:

下载到控制器/仿真器运行后,在触摸屏/仿真界面按下元件时,执行 SUB 函数内打印代码。

4.3.12. 位状态设置

介绍:根据元件动作的状态设置位寄存器地址的值,位状态元件只能显示两种状态。初始默认显示格式

文本 0, 按下时显示格式文本 1。

使用方法:点击菜单栏"HMI"→"控件箱"→"位状态设置"。将元件放至合适位置,在该元件"属性"窗口 选择需绑定的寄存器类型及地址,在不同状态的格式文本输入显示内容。并设置元件的"动作"类型。根据元件的动作类型对元件进行操作,实现对位寄存器值的设置。(支持调用函数,可选择元件按下或松开时调用。)

查看寄存器值的方法:

- ▶ 点击菜单栏"工具"→"寄存器",选择绑定的寄存器类型及地址,点击"读取"即可查看。
- ▶ 在[命令与输出]窗口的"在线命令"行,输入: print+绑定的寄存器地址及状态值,点击"发送"即可 (如输入: print MODBUS_BIT(0)/?MODBUS_BIT(0));

注意:

- 1. 不能通过寄存器的值来切换显示状态。
- 2. 元件显示状态不一定与寄存器值状态对应。



1. 属性窗口:

			_			 _
属性		×		是否图片化	False	
▲ 基本属性			-	标签		
元件编号	5			文本库		
元件名称	BitModify5			格式文本(0)		
显示层次	底层			格式文本(1)		
有效显示	显示		-	动作		
采用有效控制	False			动作	无动作	
安全时间ms	0			松开时动作	False	
绑定的设备编号	本地		-	位置和尺寸		
绑定的寄存器类型	Μ			水平位置	619	
绑定的寄存器编号	0			垂直位置	212	
⊿ 外观				宽度	109	
图片来源	背景图片库			高度	49	Ŧ
背景图片库	0\按钮\9					
绘制边框	False		轺	略数 帮助 属性		

2. 属性说明:

属性	功能	说明
元件编号	/	/

元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但 点击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 通过寄存器控制元件 是否显示
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	设置寄存器的编号	寄存器值为0时不显示,非0时显示
安全时间 ms	最少按键时间	单位 ms
绑定的设备编号	指定设备	默认 local
绑定的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择
绑定的寄存器编号	设置寄存器的编号	通过获取寄存器不同数值,控制元件不同 状态
图片来源	无、背景图片库或背景图片	无图片、背景图片库或背景图片中选择
背景图片	背景图片选择	图片来源设置为选择背景图片显示
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示
绘制边框	选择是否绘制边框	/
是否图片化	元件变为图片的形式	默认 False
文本库	文本库的名称	不设置文本库显示格式文本
格式文本 0/1	打开格式文本设置窗口设置元件 要显示的文本	默认显示文本 0, 按下时显示文本 1
动作	按键执行时的动作	参见" <u>动作</u> "章节描述
松开时动作	选择按下时或松开时执行动作	默认 False 为按下执行动作, True 为松开时动作
动作函数名	按键动作后要调用的 SUB 函数	下拉列表选择 Basic 已有全局 SUB 函数
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

例一: 给寄存器赋值1

1. 选择寄存器类型和编号;

2. "动作"选择"状态设置为1"。

🖻 Hmi1.hmi × 🔹 🔻	属性	₩ Д >
	▲ 基本属性	
	元件编号	1
	元件名称	BitModify1
	显示层次	底层
	有效显示	显示
	采用有效控制	False
	安全时间ms	0
	绑定的设备编号	本地
	绑定的寄存器类型	Μ
	绑定的寄存器编号	0
	▲ 外观	
I:BitModify	图片来源	背景图片库
按下置1	背景图片库	0\按钮\9
	绘制边框	False
	是否图片化	False
	▲ 标签	
	文本库	
	格式文本(0)	按下置1
	格式文本(1)	按下置1
	⊿ 动作	
	动作	状态设置为1
	松开时动作	False
	▲ 位置和尺寸	
	水平位置	234
	垂直位置	228
	宽度	83
	高度	33 ,
4 · · · ·	轴参数 帮助 属性	

实现效果:

按下元件后,将寄存器的值置为1(即 MODBUS_BIT(0)=1),此时元件显示状态为状态1。松开后元件显示状态为0,但寄存器值仍为1。

如果选择了松开时动作,即按下元件再松开后,MODBUS_BIT(0)=1,MODBUS_BIT(0)的值为保持为1。

动作"状态设置为0"则将寄存器置0,与"状态设置为1"相反。

例二:寄存器数值取反

1. 选择寄存器类型和编号;

2. "动作"选择"状态反转"。

🖻 Hmi1.hmi 🛛 🛛 属性 ₹ • ₽ × 采用有效控制 False ۰ 安全时间ms 0 绑定的设备编号 本地 绑定的寄存器类型 M 绑定的寄存器编号 0 ⊿ 外观 图片来源 背景图片库 背景图片库 0\按钮\9 绘制边框 False 是否图片化 False 反转 ⊿ 标签 文本库 格式文本(0) 反转 格式文本(1) 反转 ⊿ 动作 动作 状态反转 松开时动作 False ▲ 位置和尺寸 水平位置 607 垂直位置 211 宽度 109 高度 49 • • 轴参数 帮助 属性

Zmotion

实现效果:

如果 MODBUS_BIT(0)的初始值为 0,按下元件后取反,MODBUS_BIT(0)=1,再次按下元件,MODBUS_BIT(0)=0。

例三:按下时寄存器置1,松开时寄存器置0

1. 选择寄存器类型和编号;

2. "动作"选择"状态恢复"。

Zmotion[®] _{正运动技术}

🖻 Hmi1.hmi 🛛 🔹	属性	▼ ₽ :
	▲ 基本属性	
	元件编号	1
	元件名称	BitModify1
	显示层次	底层
	有效显示	显示
	采用有效控制	False
	安全时间ms	0
	绑定的设备编号	本地
	绑定的寄存器类型	M
	绑定的寄存器编号	0
	⊿ 外观	
E:BitModify	图片来源	背景图片库
	背景图片库	0\按钮\9
	绘制边框	False
	是否图片化	False
	∡ 标签	
	文本库	
	格式文本(0)	0
	格式文本(1)	1
	⊿ 动作	
	动作	状态恢复
	松开时动作	False
	▲ 位置和尺寸	
	水平位置	234
	垂直位置	228
	宽度	83
	高度	33
۰	轴参数 帮助 属性	

实现效果:

按下元件时,位寄存器 MODBUS_BIT(0)=1;松开元件后,位寄存器 MODBUS_BIT(0)=0。

例四:调用 SUB 函数

例程参见<u>功能键例一</u>。

4.3.13. 多状态设置

介绍:根据元件动作的状态设置字寄存器地址的值,同时支持调用函数。详细使用方法参见下方例子。 使用方法:点击菜单栏"HMI"→"控件箱"→"多状态设置"。将元件放至合适位置,在该元件"属性"窗口 选择需绑定的寄存器类型及地址,在不同状态的格式文本输入显示内容。并设置元件的"动作"类型。根据元件的动作类型对元件进行操作,实现对字寄存器值的设置。(支持调用函数,可选择元件按下或松开时调用。)

查看寄存器值的方法:

- ▶ 点击菜单栏"工具"→"寄存器",选择绑定的寄存器类型及地址,点击"读取"即可查看。
- ▶ 在[命令与输出]窗口的"在线命令"行,输入: print+绑定的寄存器地址及状态值,点击"发送"即可 (如输入: print MODBUS_BIT(0) / ?MODBUS_BIT(0));

注意:

- 1. 不能通过寄存器的值来切换显示状态。
- 2. 元件显示状态不一定与寄存器值状态对应。



1. 属性窗口:

属性		×				_
「「其木屋性」			⊿ 标签			
《 巫平周江	-		文本库			
	6	- 11	状态数量	ł	2	
元件名称	WordModify6	- 11	格式文本	(0)		
显示层次	底层	- 11	検式立本	(0)		
有效显示	显示		1810,X/4	(1)		
采用有效控制	False					
安全时间ms	0	- 11			元动作	_
御完的设备编号	本地	- 11	松升时动	11/F	False	
	- -	- 11	▲ 位置和尺	1		
	D	- 11	水平位置	2 L	663	
绑定的寄存器编号	0	- 11	垂直位置	ł	118	
▲ 外观		- 11	安度	•	100	
图片来源	背景图片库		一一一一一一		103	_
背景图片库	0\按钮\13		同皮		53	
绘制边框	False				1	
是否图片化	False		轴参数 帮助	属性		

2. 属性说明:

属性	功能	说明
元件编号	/	/

元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会显示以下三个参数,通过寄存器控制元件是否显示
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
安全时间 ms	最少按键时间	单位 ms
绑定的设备编号	指定设备	默认 local
绑定的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择
绑定的寄存器编号	设置寄存器的编号	通过获取寄存器不同数值,控制元件不同状 态
图片来源	无、背景图片库或背景图片	无图片、背景图片库或背景图片中选择
背景图片	背景图片选择	图片来源设置为选择背景图片显示
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示
绘制边框	选择是否绘制边框	默认 False
是否图片化	元件变为图片的形式	默认 False
文本库	文本库的名称	不设置文本库显示格式文本
状态数量	元件的状态数量(1-256)	可显示多个状态
格式文本 0/1	打开格式文本设置窗口设置元 件要显示的文本	格式文本的数量由状态数量决定
动作	按键执行时的动作	参见" <u>动作</u> "章节描述
松开时动作	选择按下时或松开时执行动作	默认 False 为按下执行动作, True 为松开时动作
动作数据	按键动作后给寄存器写入指定 值	/
动作函数名	按键动作后要调用的 SUB 函数	下拉列表选择 Basic 已有全局 SUB 函数
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

例一:调用 SUB 函数

1. 在 Basic 文件中,编写一个全局的 SUB 函数;

2. 在元件属性,动作选择"调用函数","动作函数名"选择对应的 SUB 函数名。



实现效果:按下元件时,执行 SUB 函数内打印代码。

例二: 写入数据到寄存器

1. 选择寄存器类型和编号

 选择动作"修改数据",设置"动作数据"写入寄存器的值,如例子设置"动作数据"为1; 状态支持显示两种,按下时显示"格式文本1",松开显示"格式文本0"。

🖻 Hmi1.hmi 🛛 💌	属性	▼ ₽	×
	▲ 基本属性		4
	元件编号	1	
	元件名称	WordModify1	
	显示层次	底层	
	有效显示	显示	
	采用有效控制	False	
L:WordModify D(0)=0	安全时间ms	0	
	绑定的设备编号	本地	
	绑定的寄存器类型	D	
	绑定的寄存器编号	0	
	⊿ 外观		
	图片来源	背景图片库	
· · · · · · · ·	背景图片库	0\按钮\13	
	绘制边框	False	
	是否图片化	False	
	⊿ 标签		
	文本库		
	状态数量	2	
	格式文本(0)	D(0)=0	
	格式文本(1)	D(0)=1	
	⊿ 动作		
	动作	修改数据	
	松开时动作	False	
	动作数据	1	
• •	轴参数 帮助 属性		

实现效果:

按下元件后,寄存器 MODBUS_REG(0)写入数据 1。若"动作数据"设置为其他数值,则寄存器写入对应数值。

例三:寄存器在原来的值上加上动作数据的值

- 1. 选择寄存器类型和编号;
- 2. 选择动作"数据增加/减少","动作数据"填入寄存器每次要增加/减少的数据。

🖻 Hmi1.hmi 🛛 🔻	属性	▼ 무
	▲ 基本属性	
	元件编号	1
	元件名称	WordModify1
	显示层次	底层
	有效显示	显示
	采用有效控制	False
	安全时间ms	0
	绑定的设备编号	本地
	绑定的寄存器类型	D
C:WordModify	绑定的寄存器编号	0
	⊿ 外观	
加一	图片来源	背景图片库
	背景图片库	0\按钮\13
	绘制边框	False
	是否图片化	False
	⊿ 标签	
	文本库	
	状态数量	2
	格式文本(0)	加— …
	格式文本(1)	加— …
	⊿ 动作	
	动作	数据增加/减少
	松开时动作	False
	动作数据	1
	格式文本(1)	
• • •	轴参数 帮助 屋性	

实现效果:

每按下元件一次后, MODBUS_REG(0)等于原来的值加1。

例四:寄存器原来值加上数据,在设置的状态之间循环切换

- 1. 选择寄存器类型和编号;
- 2. 选择动作"循环","动作数据"填入要增加的数据。

•	属性		→ ₽ ×
	▲ 基本属性		
	元件编号	1	
	元件名称	WordModify1	
	显示层次	底层	
	有效显示	显示	
	采用有效控制	False	
	安全时间ms	0	
	绑定的设备编号	本地	
л. «огамоату лп—	绑定的寄存器类型	D	
	绑定的寄存器编号	0	
	⊿ 外观		
	图片来源	背景图片库	
	背景图片库	0\按钮\13	
	绘制边框	False	
F:WordModity 加一	是否图片化	False	
	⊿ 标签		
	文本库		
	状态数量	3	
	格式文本(0)	ታበ	
	格式文本(1)	ታበ	
	格式文本(2)	ታበ	
	⊿ 动作	-	
	动作	循环	
	松开时动作	False	
	动作数据	1	-
	轴参数 帮助 属性		

实现效果:

按下元件后, MODBUS_REG(0)的值在 0、1、2 之间切换。

如果 MODBUS_REG(0)的初始值大于 2 时,按一下即可自动计算并递减至设定的状态数量范围内,再 开始在 0 到 2 之间切换。

寄存器的值根据状态的数量循环,例如:

- 状态数量为3,动作数据为1时,对应的寄存器值在0、1、2之间切换;
- ●状态数量为 5,动作数据为 2 时,对应的寄存器值在 0、2、4、1、3 之间切换。

4.3.14. 位状态切换开关

介绍:根据元件动作设置位寄存器地址的值并显示对应状态。根据动作设置寄存器的值使得显示状态 改变。寄存器值为0或1时则对应显示格式文本0或1。

使用方法:点击菜单栏"HMI"→"控件箱"→"位状态切换开关"。将元件放至合适位置,在该元件"属性" 窗口选择需绑定的寄存器类型及地址,在不同状态的格式文本输入显示内容。并设置元件的"动作"类型。根 据元件的动作类型对元件进行操作,实现对位寄存器值的设置及显示状态切换。(支持调用函数,可选择元 件按下或松开时调用。)

注意:

- 1. 当无设置动作时,无法直接通过按下元件来切换显示状态,需手动修改寄存器值进行切换。
- 选择位寄存器时,寄存器值为0时显示格式文本0;寄存器值为1时显示格式文本1。如果选择的 寄存器不是位寄存器,那么寄存器值为0时显示格式文本0;寄存器值不为1时显示格式文本1, 多个位的显示使用多状态元件。



1. 属性窗口:

		_				
属性		×		是否图片化	False	
▲ 基本属性				标签		
	7			文本库		
元件名称	BitSwitch7			格式文本(0)		
显示层次	底层			格式文本(1)		
有效显示	显示			动作		
采用有效控制	False			动作	无动作	
安全时间ms	0			松开时动作	False	
绑定的设备编号	本地			位置和尺寸		
绑定的寄存器类型	M			水平位置	702	
绑定的寄存器编号	0			垂直位置	351	
⊿ 外观				宽度	63	
图片来源	背景图片库			高度	109	-
背景图片库	0\开关\18					
绘制边框	False		轴翻	参数 帮助 属性		

2. 属性说明:

属性	功能	说明
元件编号	/	/

元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会显示以下三个参数,通过寄存器控制元件是否显示
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型 (有效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号 (有效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
安全时间 ms	最少按键时间	单位 ms
绑定的设备编号	指定设备	默认 local
绑定的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择
绑定的寄存器编号	设置寄存器的编号	通过获取寄存器不同数值,控制元件不同状 态
图片来源	无、背景图片库或背景图片	无图片、背景图片库或背景图片中选择
背景图片	背景图片选择	图片来源设置为选择背景图片显示
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示
绘制边框	选择是否绘制边框	默认 False
是否图片化	元件变为图片的形式	默认 False
文本库	文本库的名称	不设置文本库显示格式文本
格式文本 0/1	打开格式文本设置窗口设置元件 要显示的文本	默认显示文本 0,按下时显示文本 1
动作	按键执行时的动作	参见" <u>动作</u> "章节描述
松开时动作	选择按下时或松开时执行动作	默认 False 为按下执行动作, True 为松开时动作
动作函数名	按键动作后要调用的 SUB 函数	下拉列表选择 Basic 己有全局 SUB 函数
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

例一: 给寄存器赋值 0

1. 选择寄存器类型和编号;

2. "动作"选择"状态设置为0"。



🗄 Hmi1.hmi 🛛 🔹	属性	▼ ₽ ×	
	▲ 基本属性	^	
	元件编号	1	
	元件名称	BitSwitch1	
	显示层次	底层	
	有效显示	显示	
	采用有效控制	False	
	安全时间ms	0	
	绑定的设备编号	本地	
	绑定的寄存器类型	M	
1.BitSwitch	绑定的寄存器编号	0	
	⊿ 外观		
	图片来源	背景图片库	
技工20	背景图片库	0\开关\18	
	绘制边框	False	
	是否图片化	False	
	∡ 标签		
	文本库		
	格式文本(0)	按下置0	
	格式文本(1)	按下置0	
	⊿ 动作		
	动作	状态设置为0	
	松开时动作	False	
	▲ 位置和尺寸		
	水平位置	424	
	垂直位置	188	
	宽度	62	
	高度	109 🗸	
٠	轴参数 帮助 属性		

实现效果:

按下元件时,位寄存器 MODBUS_BIT(0)=0,元件显示状态为1 状态;松开后 MODBUS_BIT(0)仍为0,同时元件显示状态切换为0。

如果选择了松开时动作,即按下元件再松开后,MODBUS_BIT(0)=0,MODBUS_BIT(0)的值为保持为 0。

动作"状态设置为1"为将寄存器置1,与"状态设置为0"相反。

例二:寄存器数值取反

1. 选择寄存器类型和编号;

2. "动作"选择"状态反转"。

🖻 Hmi1.hmi 🛛 🔹	属性	▼ 부 >			
	⊿ 基本属性				
	元件编号	1			
	元件名称	BitSwitch1			
	显示层次	底层			
	有效显示	显示			
	采用有效控制	False			
	安全时间ms	0			
	绑定的设备编号	本地			
	绑定的寄存器类型	M			
1-BitSwitch	绑定的寄存器编号	0			
	⊿ 外观				
	图片来源	背景图片库			
- 収反 -	背景图片库	0\开关\18			
	绘制边框	False			
	是否图片化	False			
	⊿ 标签				
	文本库				
	格式文本(0)	取反			
	格式文本(1)	取反			
	⊿ 动作				
	动作	状态反转			
	松开时动作	False			
	▲ 位置和尺寸				
	水平位置	424			
	垂直位置	188			
	宽度	62			
	高度	109			
۰	轴参数 帮助 属性				

实现效果:

如果 MODBUS_BIT(0)的初始值为 0,按下元件后取反,MODBUS_BIT(0)=1;再次按下元件,MODBUS_BIT(0)=0。即可实现同时切换元件和寄存器的状态值。

例三:按下时寄存器置1,松开时寄存器置0

- 1. 选择寄存器类型和编号;
- 2. "动作"选择"状态恢复"。

Hmil.nmi ×	*	属性		▼ ₽
		▲ 基本属性		
		元件编号	1	
		元件名称	BitSwitch1	
		显示层次	底层	
		有效显示	显示	
		采用有效控制	False	
		安全时间ms	0	
		绑定的设备编号	本地	
		绑定的寄存器类型	М	
1.BitSwitch		绑定的寄存器编号	0	
		⊿ 外观		
		图片来源	背景图片库	
		背景图片库	0\开关\18	
		绘制边框	False	
		是否图片化	False	
		⊿ 标签		
		文本库		
		格式文本(0)	0	
		格式文本(1)	1	
		⊿ 动作		
		动作	状态恢复	
		松开时动作	False	
		▲ 位置和尺寸		
		水平位置	424	
		垂直位置	188	
		宽度	62	
		高度	109	

实现效果:

按下元件后, MODBUS_BIT(0)=1, 松开元件后, MODBUS_BIT(0)=0。

例四:调用 SUB 函数

例程参见<u>功能键例一</u>。

4.3.15. 多状态切换开关

介绍:根据元件动作设置字寄存器地址的值并显示对应状态。根据动作设置寄存器的值使得显示状态 改变。寄存器值为1或2时则对应显示格式文本1或2。

使用方法:点击菜单栏"HMI"→"控件箱"→"多状态切换开关"。将元件放至合适位置,在该元件"属性" 窗口选择需绑定的寄存器类型及地址,在不同状态的格式文本输入显示内容。并设置元件的"动作"类型。根 据元件的动作类型对元件进行操作,实现对字寄存器值的设置及显示状态切换。(支持调用函数,可选择元 件按下或松开时调用。)

注意:

- 1. 当无设置动作时,无法直接通过按下元件来切换显示状态,需手动修改寄存器值进行切换。
- 2. 可显示的文本数量根据"状态数量"自定义设置,数量范围 1-256。
- 3. 建议选择字寄存器控制,默认 D0,对应寄存器值为 0 时,显示格式文本 0;值为 1 时,显示格式 文本 1;依此类推,一一对应显示。



1. 属性窗口:

属性		×		标签			1
▲ 基本属性				文本库			1
元件编号	8			状态数量		2	1
元件名称	WordSwitch8			格式文本(0))		
显示层次	底层			格式文本(1))		
有效显示	显示		- 4	动作			
采用有效控制	False			动作		无动作	
安全时间ms	0			松开时动作		False	
绑定的设备编号	本地		- 4	位置和尺寸		,	
绑定的寄存器类型	D			水平位置		475	
绑定的寄存器编号	0			垂直位置		411	
⊿ 外观				宽度		109	
图片来源	背景图片库			高度		53	
背景图片库	0\按钮\13						Ŧ
绘制边框	False						
是否图片化	False		轴	参数帮助属	酣生		

2. 属性说明:

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件;

		中层: 被顶层元件覆盖, 覆盖底层元件;
		底层:被顶层和中间层元件覆盖(默认)。
		显示:该元件显示于界面之中;
有效显示	选择元件是否显示	不显示: 该元件不显示于界面之中;
		仪显示, 个可用: 该元件可止常显示, 但点
		击该元件时动作个生效 1000月1月1日(100月月1日)
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会显示以下三个参数,通过寄存器控制元件是否显示
有效的设备编号	设备编号	默认 local
(有效控制为 True)	《田洞 】	
有效的寄存器类型(有	选择寄存器类型	多种寄存器下拉列表选择
效控制为 True)		
有效的寄存器编号(有	选择寄存器编号	寄存器值为0时不显示,非0时使用
效控制为 True)		
安全时间 ms	最少按键时间	单位 ms
绑定的设备编号	指定设备	默认 local
绑定的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择
绑定的寄存器编号	设置寄存器的编号	通过获取寄存器不同数值,控制元件不同状 态
图片来源	无、背景图片库或背景图片	无图片、背景图片库或背景图片中选择
背景图片	背景图片选择	图片来源设置为选择背景图片显示
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示
绘制边框	选择是否绘制边框	/
是否图片化	元件变为图片的形式	默认 False
文本库	文本库的名称	不设置文本库显示格式文本
状态数量	元件的状态数量(1-256)	可显示多个状态
枚式 立木 1/0	打开格式文本设置窗口设置元	投武立大的粉景山北大粉景冲空
俗以又平 1/0	件要显示的文本	俗以文华的效里田扒芯效里伏足
动作	按键执行时的动作	参见" <u>动作</u> "章节描述
松开时动作	选择按下时或松开时执行动作	默认 False 为按下执行动作,True 为松开时 动作
	按键动作后给寄存器写入指定	
动作数据	值	/
动作函数名	按键动作后要调用的 SUB 函数	下拉列表选择 Basic 已有全局 SUB 函数
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

例一:调用 SUB 函数

1. 在 Basic 文件中,编写一个全局的 SUB 函数;

2. 在元件属性,动作选择"调用函数","动作函数名"选择对应的 SUB 函数名。

文件视图	▼ ₽ ×	Basic1.bas ×	🗄 Hmi1.hmi
文件名 自动 Basic1.bas Hmi1.hmi 0	运行	1 2 ⊟ global sub su 3 print ″调 4 end sub 5	ıb1 () 用函数″
🖻 Hmi1.hmi 🔺	. ₹	属性	* ₽ ×
[▲ 基本属性	
		元件编号	1
		元件名称	WordSwitch1
		显示层次	底层
		有效显示	显示
		采用有效控制	False
		安全时间ms	0
		绑定的设备编号	本地
		绑定的寄存器类型	D
		绑定的寄存器编号	0
		⊿ 外观	
		图片来源	背景图片库
		背景图片库	0\按钮\13
1:WordSwite	1 #/_	绘制边框	False
响用 图3	X.	是否图片化	False
	-	⊿ 标签	
		文本库	
		状态数量	1
		格式文本(0)	调用函数
		⊿ 动作	
		动作	调用函数
		松开时动作	False
		动作函数名	sub1 🗸
	•	轴参数 帮助 属性	

实现效果:

按下元件时,执行 SUB 函数内打印代码。

例二: 写入数据到寄存器

1. 选择寄存器类型和编号

2. 选择动作"修改数据",设置"动作数据"写入寄存器的值。

🖻 Hmi1.hmi 🛛 🔻	属性	▼ 부 >	×
	▲ 基本属性		+
	元件编号	1	
	元件名称	WordSwitch1	
	显示层次	底层	
	有效显示	显示	
	采用有效控制	False	
	安全时间ms	0	
	绑定的设备编号	本地	
	绑定的寄存器类型	D	
	绑定的寄存器编号	0	
	▲ 外观		
	图片来源	背景图片库	
	背景图片库	0\按钮\13	
L:WordSwitch	绘制边框	False	
	是否图片化	False	
	⊿ 标签		
	文本库		
	状态数量	2	
	格式文本(0)	0	
	格式文本(1)	1	
	⊿ 动作		
	动作	修改数据	
	松开时动作	False	
	动作数据	1	-
• •	轴参数 帮助 属性		

实现效果:

按下元件后,将动作数据值写入绑定寄存器,即 MODBUS_REG(0)=1,同时显示格式文本1。

例三:寄存器在原来的值上加上动作数据的值

- 1. 选择寄存器类型和编号
- 2. 选择动作"数据增加/减少",并在"动作数据"输入数值。

🗄 Hmi1.hmi 🛛 🔻	属性	▼ 1	ι×
	显示层次	底层	-
	有效显示	显示	
	采用有效控制	False	
	安全时间ms	0	
	绑定的设备编号	本地	
	绑定的寄存器类型	D	
	绑定的寄存器编号	0	
	⊿ 外观		
	图片来源	背景图片库	
	背景图片库	0\按钮\13	
	绘制边框	False	
	是否图片化	False	
	▲ 标签		
L:WordSwitch	文本库		
0	状态数量	5	
	格式文本(0)	0	
	格式文本(1)	1	
	格式文本(2)	2	
	格式文本(3)	3	
	格式文本(4)	4	
	⊿ 动作		
	动作	数据增加/减少	
	松开时动作	False	
	动作数据	1	-
• •	轴参数 帮助 属性		

实现效果:

每按下元件一次后, MODBUS_REG(0)等于原来的值加1。

注意: 当寄存器的值超过状态数量设置时,元件不显示,但触摸仍有作用效果,如上图,寄存器的值大于4时,元件不显示。

例四:寄存器原来值加上数据,在设置的状态之间循环切换

- 1. 选择寄存器类型和编号
- 2. 选择动作"循环", "动作数据"填入要增加的数据。

Hmi1.hmi × ₹	属性					
	显示层次	底层				
	有效显示	显示				
	采用有效控制	False				
	安全时间ms	0				
	绑定的设备编号	本地				
	绑定的寄存器类型	D				
	绑定的寄存器编号	0				
	✓ 外观					
	图片来源	背景图片库				
	背景图片库	0\按钮\13				
	绘制边框	False				
	是否图片化	False				
	∡ 标签					
L:WordSwitch	文本库					
	状态数量	5				
	格式文本(0)	0				
	格式文本(1)	1				
	格式文本(2)	2 .				
	格式文本(3)	3				
	格式文本(4)	4				
	⊿ 动作					
	动作	循环				
	松开时动作	False				
	动作数据	1				
4	轴会物 邦肋 屏树					

实现效果:

按下元件后, MODBUS_REG(0)的值在 0 到 4 之间切换。

如果 MODBUS_REG(0)的初始值大于 2 时,按一下即可自动计算并递减至设定的状态数量范围内,再 开始在 0 到 2 之间切换。

寄存器的值根据状态的数量循环,例如:

- 状态数量 3, 动作数据 1, 对应的寄存器值在 0、1、2 之间切换;
- 状态数量 5, 动作数据 2, 对应的寄存器值在 0、2、4、1、3 之间切换。

4.3.16. 功能键

介绍:根据元件动作实现状态切换/窗口切换/软键盘切换等。只有两种显示状态,初始默认显示格式文本 0,按下时显示格式文本 1,相当于一个"开关按钮"。

使用方法:点击菜单栏"HMI"→"控件箱"→"功能键"。将元件放至合适位置,在该元件"属性"窗口的格 式文本输入显示内容。并选择元件的"动作"类型。根据元件的动作类型对元件进行操作,实现相应的效果。

功能键的主要功能包括:调用 Basic 函数、打开/关闭窗口、按键/字符输入、切换软键盘。通过"属性" 窗口中的"动作"来选择功能。

注意:功能键不支持绑定寄存器。



1. 属性窗口:

属性		×	- 4	标签		
▲ 基本属性				文本库		
元件编号	9			格式文本(0)		
元件名称	Button9			格式文本(1)		
显示层次	底层		- 4	动作		
有效显示	显示			动作	调用函数	
采用有效控制	False			松开时动作	False	
安全时间ms	0			动作函数名		
绑定虚拟按键	No Key		- 4	位置和尺寸		
绑定物理按键	0			水平位置	246	
⊿ 外观				垂直位置	371	
图片来源	背景图片库			宽度	109	
背景图片库	0\按钮\9			高度	49	
绘制边框	False					
是否图片化	False		轴	参数 帮助 属性		

2. 属性说明:

属性	功能	说明
元件编号	/	/
元件名称	/	/
		顶层:显示在最外层,覆盖底下元件;
显示层次	选择元件显示层次	中层: 被顶层元件覆盖, 覆盖底层元件;
		底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示: 该元件显示于界面之中;

不显示: 该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点击 该元件时动作不生效 默认 False,选择 True 才会显示以下三个参 采用有效控制 通过寄存器控制元件是否显示 数,通过寄存器控制元件是否显示 有效的设备编号 设备编号 默认 local (有效控制为 True) 有效的寄存器类型 选择寄存器类型 多种寄存器下拉列表选择 (有效控制为 True) 有效的寄存器编号 寄存器值为0时不显示,非0时显示 设置寄存器的编号 (有效控制为 True) 安全时间 ms 最少按键时间 单位 ms 绑定虚拟按键 选择要绑定的虚拟按键码 默认不选择 绑定示教盒上面的物理按键 绑定物理按键 按键码值查看"虚拟键"章节 图片来源 无、背景图片库或背景图片 无图片、背景图片库或背景图片中选择 背景图片 背景图片选择 图片来源设置为选择背景图片显示 背景图片库 背景图片库选择 图片来源设置为选择背景图片库显示 绘制边框 选择是否绘制边框 是否图片化 元件变为图片的形式 默认 False 文本库 文本库的名称 不设置文本库显示格式文本 打开格式文本设置窗口设置元 默认显示文本 0, 按下时显示文本 1 格式文本 0/1 件要显示的文本 动作 按键执行时的动作 参见"动作"章节描述 默认 False 为按下执行动作, True 为松开时动 松开时动作 选择按下时或松开时执行动作 作 下拉列表选择 Basic 已有全局 SUB 函数 动作函数名 按键动作后要调用的 SUB 函数 动作操作窗口 指定动作的作用窗口 默认不选择 虚拟按键码 选择虚拟按键码 输入字符串 字符串 只能在软键盘窗口内使用 水平位置 元件的水平起始位置 不要超出水平分辨率 垂直位置 元件的垂直起始位置 不要超出垂直分辨率

Zmotion

例一:调用 SUB 函数

宽度

高度

1. 设置功能键的"动作"为"调用函数";

元件的宽度

元件的高度

2. 选择要调用的"动作函数名";

3. 选择按下时调用或松开时调用。

/

/



.	属性	₹ ₽ ×
	▲ 基本属性	A
	元件编号	1
	元件名称	Button1
	显示层次	底层
	有效显示	显示
	采用有效控制	False
	安全时间ms	0
	绑定虚拟按键	No Key
	绑定物理按键	0
t-Rutton	⊿ 外观	
调用函数	图片来源	背景图片库
	背景图片库	0\按钮\9
	绘制边框	False
	是否图片化	False
	∡ 标签	
	文本库	
	格式文本(0)	调用函数
	格式文本(1)	调用函数
	⊿ 动作	
	动作	调用函数
	松开时动作	False
	动作函数名	sub1
	▲ 位置和尺寸	
	水平位置	604
	垂直位置	193
	宽度	109
	高度	50
•	轴参数 帮助 属性	

被调用的 Basic 函数:

```
♥global sub sub1 ○
print "调用函数"
end sub
```

实现效果:

功能键按下时,调用 Basic 函数 sub1 执行打印命令。

例二:打开/关闭窗口

1. 在功能键的"动作"选择下图动作操作窗口;

⊿ 动作			⊿ 动作		
动作	弾出窗□ -	1	动作		关闭当前窗□
松开时动作	无动作	1	松开时动	作	无动作
动作操作窗口	打开基本窗口	1	▲ 位置和尺寸	ব	打开基本窗口
随父窗口一起关闭	打开置顶窗口		水平位置		打开置顶窗口
▲ 位置和尺寸	弹出窗口		垂直位置		弹出窗口
水平位置			宽度		关闭当前窗口
垂直位置			高度		关闭指定窗口
宽度		-			返回上一个基本窗口
动作	调用函数	E.			调用函数
按键执行的动作。	按下和松开都调用函数		动作		按下和松开都调用函数
10 M 20 M 2 M 2 M 2 M 2 M 2 M 2 M 2 M 2 M	输入物理按键		按键执行的动	作。	输入物理按键
	输入虚拟按键				输入虚拟按键
	输入字符				输入字符
轴参数 帮助 属性	如键内闭格		*** *** *** *** ***	₽ ML	软键盘切换

打开窗口

关闭窗口

Zmotion

2. 以打开窗口为例,在功能键的"动作"选择好要打开的窗口类型后,再找到"动作操作窗口",选择要 打开的窗口编号。

动作操作窗口	11:11弹出窗口 -	
随父窗口一起关闭	无	
▲ 位置和尺寸	6:CharPad-Lower	
水平位置	7:CharPad-Upper	
垂直位置	8:NumKeypad	
宽度	10:Start	-
动作操作窗口	11:11弾出窗口	

实现效果:

如若"动作"选择"弹出窗口","动作操作窗口"选择11号窗口,则按下功能键,立即弹出11号窗口。

例三: 自定义软键盘功能

- 新建一个基本窗口和一个软键盘窗口(新建窗口在窗口的"属性"将窗口类型修改为软键盘窗口即 可),调整软键盘窗口大小(建议调整至比基本窗口小);
- 在基本窗口中添加"字符显示"元件,设置该元件"属性"→"可编辑"为 True→"软键盘窗口号"选择上 述新建的软键盘窗口号;
- 在软键盘窗口中添加一个"字符显示"和多个"功能键"。设置"字符显示"的属性→"软键盘显示专用"
 为 True;设置"功能键"的属性→"动作"选择"输入字符"/"输入虚拟按键"/"输入物理按键"/"软键盘 切换",并对应绑定按键值/软键盘窗口或字符串内容。



Hmi1.hmi ×	属性	Ŧ	д >
	显示层次	底层	
	有效显示	显示	
	采用有效控制	False	
	安全时间ms	0	
	绑定虚拟按键	No Key	
	绑定物理按键	0	
da site No 4da bate sta	⊿ 外观		
目定义软键盘	图片来源	背景图片库	
ABC	背景图片库	0\按钮\9	
	绘制边框	False	
输入物理按键输入字符串	是否图片化	False	
	⊿ 标签		
■ 输入虚拟按键	文本库		
	格式文本(0)	输入虚拟按	
确认	格式文本(1)	输入虚拟按	
	⊿ 动作		
	动作	输入虚拟按键	
	松开时动作	False	
	虚拟按键码	VKEY_PLUS	
	4 位置和尺寸		
	水平位置	136	
	垂直位置	285	
	轴参数 帮助 尾性		

实现效果:

下载运行后,点击"功能键"按钮可触发对应的效果,如按下"输入字符串"则在"字符显示"元件上输入 "Zmotion";按下"软键盘切换"按钮则可切换其他软键盘窗口等。

xplc sci	reen	×
xplc sci	reen 功能键的使用 	×
	输入虚拟按键 软键盘切换 确认	

4.3.17. 物理按键

介绍:通过该元件与虚拟按键/示教盒物理按键绑定,实现自定义物理按键动作。

使用方法:点击菜单栏"HMI"→"控件箱"→"物理按键"。将该元件放至合适位置,在该元件"属性"窗口 绑定"虚拟按键"/"物理按键",只能二选一。在"动作"选择需要实现的动作效果,即可实现通过虚拟键/实际 硬件按钮进行对应动作。(如:该元件绑定了示教盒物理按键某个键,动作选择"调用函数",那么当按下示 教盒该按钮时,即可调用对应函数)

注意:

- 1. 该元件只在 Hmi 界面编辑时显示, 实际运行界面不显示。
- 2. 绑定物理按键需获取按键键值。键值需从对应型号的示教盒手册查看。

1:KeyButton

1. 属性窗口:

属性		×		
▲ 基本属性				
元件编号	1			
元件名称	KeyButton1			
显示层次	底层			
有效显示	显示			
采用有效控制	False			
安全时间ms	0			
绑定虚拟按键	No Key			
绑定物理按键	0			
⊿ 动作				
动作	调用函数			
松开时动作	False			
动作函数名				
▲ 位置和尺寸				
水平位置	74			
垂直位置	109			
宽度	100			
高度	32	-		
	1			
轴参数 帮助 属性				



2. 属性说明:

属性	功能	说明	
元件编号	/	/	
元件名称	/	/	
	选择元件显示层次	顶层:显示在最外层,覆盖底下元件;	
显示层次		中层: 被顶层元件覆盖,覆盖底层元件;	
		底层:被顶层和中间层元件覆盖(默认)。	
有效显示	选择元件是否显示	显示: 该元件显示于界面之中;	
		不显示: 该元件不显示于界面之中;	
		仅显示,不可用:该元件可正常显示,但点	
		击该元件时动作不生效	
亚田古动惊剧	通过寄存器控制元件是否显示	默认 False,选择 True 通过寄存器控制元件	
米用有效控制		是否显示	
有效的设备编号	设备结合		
(有效控制为 True)			
有效的寄存器类型(有	水均中方田米利	夕 动 宏 左 嬰 下 拉 列 丰 选 择	
效控制为 True)		多种可仔硷下拉列农选择	
有效的寄存器编号(有	<u> </u>	客存器值为0时不显示 非0时显示	
效控制为 True)	以且可什部的狮子	司行帝国/30时不驱亦,非0时驱亦	
安全时间 ms	最少按键时间	单位 ms	
绑定虚拟按键	选择要绑定的虚拟按键码	默认不选择	
绑定物理按键	绑定示教盒上面的物理按键	按键码值查看"虚拟键"章节	
动作	按键执行时的动作	参见" <u>动作</u> "章节描述	
松开时动作	选择按下时或松开时执行动作	默认 False 为按下执行动作,True 为松开时	
		动作	
动作函数名	按键动作后要调用的 SUB 函数	下拉列表选择 Basic 已有全局 SUB 函数	
动作操作窗口	指定动作的作用窗口	/	
虚拟按键码	选择虚拟按键码	默认不选择	
字符串	输入字符串	只能在软键盘窗口内使用	
水平位置	元件的水平起始位置	不要超出水平分辨率	
垂直位置	元件的垂直起始位置	不要超出垂直分辨率	
宽度	元件的宽度	/	
高度	元件的高度	/	

例:将 ZHD400X 示教盒外部物理按键 X-(对应的物理按键码值 24)绑定到虚拟按键,并在属性窗口 设置外部物理按键按下后要调用的函数,运行时,按下外部物理按键,立即执行调用的函数。
▲ 基本属性	
元件编号	1
元件名称	Х-
显示层次	底层
有效显示	显示
采用有效控制	False
安全时间ms	0
绑定虚拟按键	No Key
绑定物理按键	24
⊿ 动作	
动作	按下和松开都调用函数
动作函数名	deal_manul(0,-1)
松开调用函数	deal_manulstop(0)
▲ 位置和尺寸	
水平位置	140
垂直位置	270
宽度	100
高度	32

4.3.18.列表

介绍:元件通过列表的方式显示多个状态。列表行数通过"状态数量"设置,范围1-256。

使用方法:点击菜单栏"HMI"→"控件箱"→"列表"。将该元件放至合适位置,在"属性"窗口绑定寄存器 及对应编号,设置需显示的"状态数量",填入对应列表文本即可。该元件也可结合指令 HMI_LISTTEXTS 进行设置。

提示:

1. 当调整元件高度不足以显示全部列表项时,运行后会自动显示滚动条。

2. 选择某行文本时,绑定寄存器的值与文本对应的状态值对应。

1:List	
	-

1. 属性窗口:



属性	>	<	边框颜色		000000	
▲ 基本属性			选中部分颜色		99C9EF	
元件编号	10		行间距	2		
元件名称	List10		▲ 标签			
显示厚次	底层		文本库			
	显示		状态数量	2		
	False		格式文本	A	BC	
	本地		列表文本(0)			
	D		列表文本(1)			
	0		▲ 位置和尺寸			
选中调用函数			水平位置	94	ţ	
动态项	False		垂直位置	27	70	
▲ 外观			宽度	15	50	
列表方式	True		高度	60)	
弹出方式	向下弹出					Ŧ
图片来源	无					
绘制边框	True	4	轴参数 帮助 属性			

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会显示以下三个参数,通过寄存器控制元件是否显示
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
绑定的设备编号	指定设备	默认 local
绑定的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择
绑定的寄存器编号	设置寄存器的编号	通过获取寄存器不同数值,控制元件不同状 态

选中调用函数	选中操作调用的函数	/
动态项	动态读取并修改列表项	使用 HMI_LISTTEXTS 指令设置动态读取列 表时必须将动态项开启。
列表方式	设置列表元件为采样列表或下 拉式	默认 True,选择 False 时会弹出"下拉调用函数"
弹出方式	可选向下弹出或向上弹出	默认向下弹出
图片来源	无、背景图片库或背景图片	无图片、背景图片库或背景图片中选择
背景图片	背景图片选择	图片来源设置为选择背景图片显示
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示
绘制边框	选择是否绘制边框	/
选中部分颜色	设置列表元件选中部分的颜色	/
行间距	列表行间距,上下间距相等	默认为2
文本库	文本库的名称	不设置文本库显示格式文本
状态数量	元件的状态数量(0-255)	可显示多个状态
格式文本	设置空间标签的样式	/
列表文本 0/1	设置列表中的文本	/
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

示例**:**

1. 选择寄存器类型和编号;

2. 选择"状态数量",设置每个状态下列表对应显示的文本内容。

3. 切换列表选中行可使绑定寄存器的状态值变化。修改寄存器状态值也可切换选中行。

4. 列表文本内容也可通过 HMI_LISTTEXTS 列表指令设置,但需将属性中"动态项"设置为 True。指令 设置优先级大于元件属性设置。

5. 选择的状态显示颜色、行间距、弹出方式等可通过"属性"窗口进行设置。



Hmi1.hmi × =	属性	▼ ₽ ×
	▲ 基本属性	
	元件编号	1
	元件名称	List1
	显示层次	底层
	有效显示	显示
	采用有效控制	False
	绑定的设备编号	本地
the ten	绑定的寄存器类型	D
文本0 ▲	绑定的寄存器编号	0
文本2 -	选中调用函数	
	动态项	False
	⊿ 外观	
	列表方式	True
	弹出方式	向下弹出
	图片来源	背景图片
	背景图片	
	绘制边框	False
	选中部分颜色	E9E9FF
	行间距	2
	⊿ 标签	
	文本库	
	状态数量	3
	格式文本	ABC
	列表文本(0)	文本0
	列表文本(1)	文本1
	列表文本(2)	文本2
• •	轴参数 帮助 属性	

状态文本		×	状态文本	×
状态	5: 〕 本: 文本0		状态: 3 文本: 文本1	
确	龍	取消	确定 取消	



实现效果:

按下文本 0 行时, MODBUS_REG(0)=0 (为第一个状态 0 的值);按下文本 1 行时, MODBUS_REG(0)=3 (为第二个状态 1 的值);按下文本 2 行时, MODBUS_REG(0)=1 (为第三个状态 2 的值),选择的行显示颜色为所设置的颜色。反之,若使 MODBUS_REG(0)=0 时,文本 0 行即被选中; MODBUS_REG(0)=3 时,文本 1 行即被选中。

状态值若相同,相同状态之间无法切换。

详细使用可参见例程: 动态列表

特别注意:

设置多个列表时,注意每个列表绑定的寄存器变量是否相同。在选择列表选项时,若设置的寄存器变量 相同,各列表之间互相影响,会同步选定相同的选项;若设置的寄存器变量不同,则每个列表可独立选择选 项,互不影响。如下图。

当两个列表绑定的寄存器都为 D0 时,运行后选择"列表项 2"的同时,"选项 2"也会被同步选中。



当一个列表绑定的寄存器为 D0,另一个列表绑定的寄存器为 D1 时,运行后选择"列表项 1"时,另一列表可自由选择其他项,两者互不影响。





4.3.19. 值

介绍:支持编辑并显示数值,并将值存入对应绑定寄存器中。

使用方法:点击菜单栏"HMI"→"控件箱"→"值"。将该元件放至合适位置,在"属性"窗口中选择绑定的 寄存器及编号,以及选择是否使用"可编辑"功能调用软键盘。并对值的其他项进行设置,如:数据类型、字 符数、小数位数等。

值元件常用功能:

- ▶ 显示寄存器的内容,只能显示非字符类型数据
- ▶ 调用软键盘窗口直接设置绑定的寄存器的值
- ▶ 调用 SUB 函数

注意:可编辑设置为 True,选择调用软键盘窗口后,便不能调用函数。



1. 属性窗口:



属性	×	:	边框颜色	000000	
从 其木屋性			字符数	8	
	1		小数位数	0	
	i Value1		密码显示	False	
			最小值	0.000000	
			最大值	0.000000	
	Ealco		∡ 标签		
一 不 而 行 双 控 利	False		格式文本	123 .	
	INIT22		▲ 动作		
	11N152		点击调用函数		
新正的反首编写 	фии D		▲ 位置和尺寸		
	0		水平位置	130	
が正いおけ諸無ち	0		垂直位置	143	
安王时间们的	U Falsa		宽度	100	-
11%以时1进力1	Faise		高度	32	
▲ グ ГX光			113000		
图片来源	无	_			
绘制边框	True	\$	由参数帮助属性		

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会显示以下三个参数,通过寄存器控制元件是否显示
有效的设备编号(有效 控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
可编辑	数据元件是否启用输入	默认 False,选择 True 调用软键盘窗口输入
软键盘窗口号	数据元件编辑时使用的软键盘 窗口	可编辑选择 True 时存在的属性
数据类型	数值元件的数据类型设置	默认 INT32

绑定的设备编号	设备编号	默认 local
绑定的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择
绑定的寄存器编号	选择寄存器编号	通过获取寄存器不同数值,控制元件不同状 态
安全时间 ms	最少按键时间	单位 ms
修改时通知	修改后发出 bit 位通知(设 ON 或 OFF)	默认 False,选择 True 时选择通知的寄存器
通知的设备编号(修改 时通知为 True)	设备编号	默认 local
通知的寄存器类型(修 改时通知为 True)	选择寄存器类型	多种寄存器下拉列表选择
通知的寄存器编号(修 改时通知为 True)	选择寄存器编号	通过获取寄存器不同数值,控制元件不同状 态
图片来源	无、背景图片库或背景图片	无图片、背景图片库或背景图片中选择
背景图片	背景图片选择	图片来源设置为选择背景图片显示
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示
绘制边框	选择是否绘制边框	/
边框颜色	设置边框颜色	设置边框为 True 时显示该属性
字符数	设置字符元件操作的字符的长 度	默认 8
小数位数	小数点位数	默认 0
密码显示	设置为 True 时元件显示内容为 *号	默认为 False
最小值	输入下限	/
最大值	输入上限	/
格式文本	打开格式文本设置窗口设置元 件要显示的文本	格式文本的数量由状态数量决定
点击调用函数	调用 Basic 函数	/
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

示例:显示/更改寄存器的值

1. "可编辑"选择要调用的软件盘窗口

2. 设置寄存器的类型和编号

3. 根据需求设置数据类型,最大值,最小值等

▲ 基 本 周注	
元件编号	1
元件名称	Value1
显示层次	底层
有效显示	显示
采用有效控制	False
可编辑	True
软键盘窗口号	8:NumKeypad
数据类型	INT32
绑定的设备编号	本地
绑定的寄存器类型	D
绑定的寄存器编号	0
安全时间ms	0
修改时通知	False
▲ 外观	
图片来源	无
绘制边框	False
字符数	8
小数位数	0
密码显示	False
最小值	0.000000
最大值	999999.000000

实现效果:

元件初始的显示内容为绑定的寄存器的值,MODBUS_REG(0)=32,实时获取寄存器的值并刷新元件内容。



点击元件后打开软键盘窗口,此时可输入数据,确认后更改寄存器的值。

4.3.20. 字符显示

介绍:用于编辑并显示所有字符串类型数据。

使用方法:点击菜单栏"HMI"→"控件箱"→"字符显示"。将该元件放至合适位置,在"属性"窗口中选择 绑定的寄存器及编号,以及选择是否使用"可编辑"功能调用软键盘。并对其他项进行设置,如字符数、多行 显示、调用函数等。

注意:

- 1. 寄存器类型中的数据必须为字符串类型;
- 调用软键盘窗口可自定义输入数据并显示。当寄存器类型为自定义@且所设置的变量不为数组时, 软键盘不可调用编辑;
- 3. "可编辑"设置为 True,选择调用软键盘窗口后,便不能调用函数。
- 4. 开启"多行"和"可编辑"属性时,使用中不弹出软键盘窗口,可用物理键盘直接输入。(输入中文时 需用"Crtl+Shift"键进行切换到内置中文输入法)

1:String ABC

1. 属性窗口:

属性		×	5	字符数	16	
▲ 基本属性			B	图片来源	无	
元件编号	1		4	会制边框	True	
元件名称	String1		ì	力框颜色	000000	
显示层次	底层		ą	密码显示	False	
有效显示	显示		- 4	示签		
采用有效控制	False		ħ	各式文本	ABC	
软键盘显示专用	False		Z	动作		_
可编辑	False		5	点击调用函数		_
绑定的设备编号	本地		(立置和尺寸		_
绑定的寄存器类型	D		7.	水平位置	106	
绑定的寄存器编号	0		Ē	垂直位置	154	
安全时间ms	0		3	宽度	100	
修改时通知	False		Ē	高度	32	-
⊿ 外观						
多行	False		轴参	数帮助属性		

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 通过寄存器控制元件 是否显示
有效的设备编号(有效 控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
软键盘显示专用	软键盘的输入显示	一般用于键盘窗口
可编辑	字符显示元件是否启用输入	默认 False,选择 True 调用软键盘窗口输入
软键盘窗口号	字符显示元件编辑时使用的软 键盘窗口	可编辑选择 True 时存在的属性
绑定的设备编号	设备编号	默认 local
绑定的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择
绑定的寄存器编号	选择寄存器编号	通过获取寄存器不同数值,控制元件不同状 态
安全时间 ms	最少按键时间	单位 ms
修改时通知	修改后发出 bit 位通知(设 ON 或 OFF)	默认 False,选择 True 时选择通知的寄存器
通知的设备编号(修改 时通知为 True)	设备编号	默认 local
通知的寄存器类型(修 改时通知为 True)	选择寄存器类型	多种寄存器下拉列表选择
通知的寄存器编号(修 改时通知为 True)	选择寄存器编号	通过获取寄存器不同数值,控制元件不同状 态
多行	字符为多行字符时设置为 True	/
字符数	设置字符元件操作字符的长度	默认 16
图片来源	无、背景图片库或背景图片	无图片、背景图片库或背景图片中选择
背景图片	背景图片选择	图片来源设置为选择背景图片显示
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示
绘制边框	选择是否绘制边框	/



边框颜色	设置边框颜色	设置边框为 True 时显示该属性
密码显示	设置为 True 时元件显示内容为 *号	默认为 False
格式文本	打开格式文本设置窗口设置元 件要显示的文本	格式文本的数量由状态数量决定
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

例一:显示寄存器存储的字符串

1. 设置寄存器的类型和编号,显示字符串类型数据。



实现效果:

当 VRSTRING(0,8) = "abc", 元件显示寄存器保存字符串 abc。



例二: 自定义元件显示文本

- 1. "可编辑"选择要调用的软件盘窗口;
- 2. 设置寄存器的类型和编号。

	属性	×
	▲ 基本属性	<u>^</u>
	元件编号	1
	元件名称	String1
	显示层次	底层
1:String	有效显示	显示
ABC	采用有效控制	False
	软键盘显示专用	False
	可编辑	True
	软键盘窗口号	6:CharPad-Lower
	绑定的设备编号	本地
	绑定的寄存器类型	D
	绑定的寄存器编号	0
	安全时间ms	0
	修改时通知	False
	⊿ 外观	
	多行	False
	字符数	16
	图片来源	无
	绘制边框	False
	密码显示	False
	⊿ 标签	
	格式文本	ABC
	轴参数 帮助 属性	

实现效果:

点击字符显示元件,弹出软键盘窗口,设置元件要显示的内容后 Enter 确定。

plc screen		;
Zmotion		
Esc	Zmotion Backspace	
Sym 1	2 3 4 5 6 7 8 9 0 - +	
Tab q	wertyuiop?	
Caps	a s d f g h j k 1 , "	
EN/CH	z x c v b n m , . Enter	
Clear Del	Space + -	
L		

4.3.21. 滑块开关

介绍:在一定取值范围内,能进行拖动实现改变数值大小的元件。如:可通过滑块实现视图缩放、倍率 调节、充当滑块开关等,具体应用需根据实际需求。

使用方法:点击菜单栏"HMI"→"控件箱"→"滑块开关"。将该元件放至合适位置,在"属性"窗口中选择 绑定的寄存器及编号。通过寄存器变量控制数据变化从而实现效果。

注意:目前控制器仅4系列及以上控制器支持,示教盒ZHD500X已支持,ZHD300X和ZHD400X升级固件可支持。



1. 属性窗口

属性		×		刻度颜色	000000
▲ 基本属性				主刻度数	3
元件编号	1			副刻度数	4
	Slip1			是否显示符号	True
显示层次	底层			滑块选择	0\滑块\0
	显示			滑块颜色	00FFFF
	False			滑块水平位置	0
	本地			滑块垂直位置	0
	D			滑块宽度	16
	0			滑块高度	24
修改时通知	False			滑轨1颜色	404040
▲ 外观				滑轨2颜色	C0C0C0
	无		- 4	标签	
绘制边框	False		_	格式文本	符号
 填充	False		- 4	位置和尺寸	
	1		_	水平位置	32
	从左到右		_	垂直位置	136
是否显示刻度	True		_	宽度	200
最小刻度	1			高度	48 🗸
最小值	0.000000				
最大值	100.000000		轴	参数 帮助 属性	

属性	功能	说明
元件编号	/	/

元件名称	/	/	
		顶层:显示在最外层,覆盖底下元件;	
显示层次	选择元件显示层次	中层: 被顶层元件覆盖,覆盖底层元件;	
		底层: 被顶层和中间层元件覆盖(默认)。	
		显示: 该元件显示于界面之中;	
有效显示	选择元件是否显示	不显示: 该元件不显示于界面之中;	
14794-147		仅显示,不可用: 该元件可正常显示, 但点	
		击该元件时动作不生效	
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 才会显示以下三个参	
		数,通过寄存器控制元件是否显示	
有效的设备编号(有效	设备编号	默认 local	
控制为 True)			
有效的寄存器类型(有	选择寄存器类型	多种寄存器下拉列表选择	
效控制为 True)			
有效的寄存器编号(有	选择寄存器编号	寄存器值为0时不显示,非0时显示	
效控制为 True)			
绑定的设备编号	指定设备	默认 local	
绑定的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择	
绑定的寄存器编号	设置寄存器的编号	通过获取寄存器不同数值,控制元件不同状	
		态	
修改时通知	修改后友出 bit 位通知(设 ON	默认 False,选择 True 时选择通知的寄存器	
	<u> </u>		
图片米源 北月四川	尤、背景图片库或背景图片 北日四山地超	尤图片、背景图片库或背景图片甲选择 图4. 去还况罢为进程北县图4. 日二	
背景图片	育 京 图 片 広 连 按 超	图片米源设直为选择育景图片显示	
背景图片库	背景图片库选择	图片米源设直为选择肖景图片库显示	
绘制边框	选择是否绘制边框		
填充	选择是否填充颜色	填充整个元件	
边界	调节元件与该元件边框的距离	边界值越大,距离越大	
方向	选择滑块的方向	有 4 个方向选择:从左到右、从右到左、从	
		上到下、从下到上	
是合显示刻度	显示刻度时设置为True	设置为 True 时显示最小刻度属性	
最小刻度 	设置滑块每一次拖动的刻度	最小为1	
最小值	刻度的最小起始数值	正负数均可	
最大值	刻度的最大结束数值	正负数均可	
刻度颜色	选择刻度数的显示颜色	默认为黑色	
主刻度数	设置滑块长刻度的段数	/	
副刻度数	设置滑块短刻度的段数	/	
是否显示符号	是否显示刻度	/	
滑块选择	选择滑块的样式	/	
滑块颜色	设置滑块的颜色	/	
滑块水平位置	设置滑块水平位置	为0时滑块水平居中	



滑块垂直位置	设置滑块垂直位置	为0时滑块垂直居中
滑块宽度	设置滑块宽度	/
滑块高度	设置滑块高度	/
滑轨1颜色	滑块右边滑轨的颜色	/
滑轨2颜色	滑块左边滑轨的颜色	/
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

例一: 滑块刻度的基础设置

选择寄存器类型,用于存放滑块变化的数值参数,拖动滑块可根据实际调节大小。滑块显示的刻度数及 分度数可通过主刻度数和副刻度数设置。先确定最小值和最大值后根据实际情况设置主刻度数插入的线数 为5,再设置副刻度数插入单格中的线数为2,从而将单格分成3小格。根据喜好对进度条颜色,滑块颜色 和样式进行选择。

例程详情参见<u>视图缩放</u>。



例二:"滑块开关"绑定寄存器的方法

1. 添加"滑块开关"元件;

2. 在元件"属性"中的"寄存器类型"选择合适的寄存器类型,并填写寄存器编号;(例如:下图选择寄存器为 DT0)

注意:若选择"修改时通知"为 True 时,"通知的设备编号"寄存器编号不能与设备编号的寄存器重复。



实现效果:

初始值默认为 0, table(0)的打印值为 0, 同时由于未操作拖动滑块, 此时"修改时通知"的寄存器 table(10) 打印值也为 0。



xplc screen	×
命令与输出	
Down to Controller Ram Success, 2023-08-04 16:59:19, Elapsed time: 187ms. >>?table(0) 0 >>?table(10) 0	
在线命令: ?table(10)	
命令与输出 查找结果	

当滑块滑到 50 处, table(0)的打印值同步刷新为 50。而由于已拖动滑块做出修改,此时 table(10)的值变 为 1,表示该元件修改已生效。

xplc screen	×
命令与输出	
0 >>?table(0) 50 >>?table(10) 1	
在线命令: ?table(10)	
命令与输出 查找结果	

4.3.22. 定时器

介绍:利用定时器功能定时刷新进行重复动作。

使用方法:点击菜单栏"HMI"→"控件箱"→"定时器"。将该元件放至合适位置,在"属性"窗口中设置好 定时器启动间隔时间,并设置好"动作"内容,即可实现定时重复动作。"动作"支持调用 SUB 函数或对寄存 器进行赋值。

注意:Hmi界面编辑时显示,实际运行中不显示。

1:Timer	
	L

1. 属性窗口:

属性		×
∡ 基本属性		4
元件编号	1	
元件名称	Timer1	
显示层次	底层	
有效显示	显示	
采用有效控制	False	
时间ms	1000	
循环类型	False	
写入的设备编号	本地	
写入的寄存器类型		
写入的寄存器编号	0	
⊿ 动作		
动作	无动作	
松开时动作	False	
▲ 位置和尺寸		
水平位置	130	
垂直位置	117	
宽度	100	
高度	32	
	1	_
轴参数 帮助 属性		

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 通过寄存器控制元件 是否显示
有效的设备编号(有效 控制为 True)	设备编号	默认 local
有效的寄存器类型(有 效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号(有 效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时使用
时间 ms	最少按键时间	单位 ms(仅支持整数,不支持浮点数)
循环类型	选择定时器是否循环	默认 False
写入的设备编号	设备编号	默认 local
写入的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择
写入的寄存器编号	选择寄存器编号	通过获取寄存器不同数值,控制元件不同状 态
动作	按键执行时的动作	参见" <u>动作</u> "章节描述
松开时动作	选择按下时或松开时执行动作	默认 False 为按下执行动作, True 为松开时 动作
动作数据	按键动作后给寄存器写入指定 值	/
动作函数名	按键动作后要调用的 SUB 函数	下拉列表选择 Basic 已有全局 SUB 函数
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

例一:调用 SUB 函数

1. 在 Basic 文件中,编写一个全局的 SUB 函数;

2. 在元件属性,动作选择"调用函数","动作函数名"选择对应的 SUB 函数名;

3. 在"时间 ms"中填入调用函数的时间间隔。

	▲ 基本属性		
	元件编号	1	
	元件名称	Timer1	
	显示层次	底层	
	有效显示	显示	
	采用有效控制	False	
	时间ms	1000	
	循环类型	False	
imer	写入的设备编号	本地	
-	写入的寄存器类型		
	写入的寄存器编号	0	
	⊿ 动作		
	动作	调用函数	
	松开时动作	False	
	动作函数名	sub1	
	▲ 位置和尺寸		
	水平位置	205	
	垂直位置	183	
	宽度	100	
	高度	32	

Zmotion

实现效果:

Hmi运行后,元件不显示,每隔 1000ms 调用一次函数 SUB 执行。

例二: 给寄存器在原来的值上加上动作数据的值

- 1. 选择寄存器类型和编号;
- 2. 选择动作"数据增加/减少","动作数据"填入每次寄存器要增加的数据。
- 3. 在"时间 ms"中填入寄存器累加的时间间隔。

	▲ 基本属性		
	元件编号	1	
	元件名称	Timer1	
	显示层次	底层	
	有效显示	显示	
	采用有效控制	False	
	时间ms	1000	
	循环类型	False	
Timer	写入的设备编号	本地	
-	写入的寄存器类型	D	
	写入的寄存器编号	0	
	⊿ 动作		
	动作	数据增加/减少	
	松开时动作	False	
	动作数据	10	
	▲ 位置和尺寸		
	水平位置	205	
	垂直位置	183	
	宽度	100	
	高度	32	

Zmotion

实现效果:

每间隔 1000ms, MODBUS_REG(0)的值等于原来的值加 10。

查看寄存器值的方法:

- ▶ 点击菜单栏"工具"→"寄存器",选择绑定的寄存器类型及地址,点击"读取"即可查看。
- ▶ 在[命令与输出]窗口的"在线命令"行,输入: print+绑定的寄存器地址及状态值,点击"发送"即可 (如输入: print MODBUS_BIT(0) / ?MODBUS_BIT(0));

4.3.23. 自定义

介绍: 自定义元件是通过调用 Basic 程序来定义其行为的元件,与程序配合在显示屏上动态绘图,通过 元件大小确定绘图区域。 使用方法:点击菜单栏"HMI"→"控件箱"→"自定义"。将该元件放至合适位置,调整元件大小确定绘图 区域。在 Basic 文件中先定义全局的刷新函数和绘图函数,后在该元件"属性"窗口中"刷新函数"、"绘图函数"分别对应调用 Basic 的 SUB 函数。

注意: 绘图函数的坐标都是相对于元件的左上方为零点。

自定义元件的3个特别属性:

数据索引:一般用于 TABLE 编号,可以指明当前元件的数据位于的 TABLE 位置。

刷新函数、绘图函数:指明元件行为的 SUB 函数。

自定义元件的使用例程参见正运动官方网站 www.zmotion.com.cn 触摸屏例程。



1. 属性窗口:

届 性		×	刷新	函数		
/[冉] 上			绘图	函数		
▲ 基本属性			▲ 位置	和尺寸		
元件编号	1		水平	位置	93	
元件名称	Custom1		垂直	位置	136	
显示层次	底层		宽度		200	
有效显示	显示		高度		150	•
采用有效控制	False				1	
数据索引	0		轴参数	帮助 属性		

属性	功能	说明	
元件编号	/	/	
元件名称	/	/	
		顶层:显示在最外层,覆盖底下元件;	
显示层次	选择元件显示层次	中层: 被顶层元件覆盖,覆盖底层元件;	
		底层:被顶层和中间层元件覆盖(默认)。	
		显示: 该元件显示于界面之中;	
有效显示	选择元件是否显示	不显示: 该元件不显示于界面之中;	
		仅显示,不可用:该元件可正常显示,但点	



		击该元件时动作不生效	
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 通过寄存器控制元件	
		是否显示	
有效的设备编号(有效	设久编号	野けん local	
控制为 True)	这 面端 5	款 K IOCal	
有效的寄存器类型(有	<u> </u>	夕孙安方盟下台列丰进权	
效控制为 True)	远拜苛 <u></u> 行 奋矢空	多种句仔器卜拉列表选择 	
有效的寄存器编号(有	进权安方盟护马	寄存器值为0时不显示,非0时显示	
效控制为 True)	见非可什品细 5		
粉提志可	指明当前元件的数据位于的		
	TABLE 位置	7	
刷新函数	指明元件行为的 SUB 函数	周期调用来判断是否要重新绘图	
绘图函数	指明元件行为的 SUB 函数	需要绘图时自动被调用	
水平位置	元件的水平起始位置	不要超出水平分辨率	
垂直位置	元件的垂直起始位置	不要超出垂直分辨率	
宽度	元件的宽度	/	
高度	元件的高度	/	

4.3.24. CAD

介绍:用于显示矢量图形的元件。

使用方法:点击菜单栏"HMI"→"控件箱"→"CAD"。将该元件放至合适位置,选择三次文件导入的通道 号(仅 0、1 两个通道可设置),根据需求设置图层/轨迹/空移颜色等。导入/关闭图形等操作需配合 CAD 指 令进行使用。

注意:

- 1. 该矢量图片文件必须存放至 flash 文件中。
- 2. 目前支持导入的图形格式为: .dxf/.ai/.plt/.dst/.nc/.cnc。



1. 属性窗口:

属性		×
▲ 基本属性		4
元件编号	1	
元件名称	CAD1	
显示层次	底层	
有效显示	显示	
采用有效控制	False	
通道号	0	
▲ 外观		
绘制边框	False	
背景颜色	000000	
图层颜色	False	
轨迹颜色	00FFFF	
空移颜色	FF0000	
选中颜色	FFD700	
辅助颜色	00FF00	
⊿ 标签		
格式文本	Sign	
▲ 位置和尺寸		
水平位置	61	
垂直位置	84	
宽度	200	
高度	150	•
轴参数 帮助 属性		

属性	功能	说明
元件编号	/	/
元件名称	/	/
		顶层:显示在最外层,覆盖底下元件;
显示层次	选择元件显示层次	中层: 被顶层元件覆盖, 覆盖底层元件;
		底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示: 该元件显示于界面之中;
		不显示:该元件不显示于界面之中;
		仅显示,不可用:该元件可正常显示,但点
		击该元件时动作不生效
采用有效控制	·圣社安士田校时二件日本日二	默认 False,选择 True 通过寄存器控制元件是
	通过苛存奋控制儿件定省並小	否显示
有效的设备编号(有	设备编号	默认 local

效控制为 True)			
有效的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择	
(有效控制为True)			
有效的寄存器编号	<u> </u>	安方器估为 0 时不且子 北 0 时且子	
(有效控制为 True)	见件可任奋狮 5	苛什奋阻//0 的小並示,非0的並示	
		最多只支持两个通道: 0/1,最多能同时运行	
通道号	设置通道号	的文件个数	
绘制边框	选择是否绘制边框	默认 False,选择 True 弹出[边框颜色]选择	
背景颜色	选择背景颜色	默认黑色	
图层颜色	是否使用图层颜色	设置为 True 时,不同图层图形颜色不一致	
轨迹颜色	选择轨迹颜色	/	
空移颜色	选择空移颜色	不连贯的区域之间的位移为空移	
选中颜色	选择选中颜色	/	
辅助颜色	选择辅助颜色	/	
格式文本	设置元件文本的样式等参数	/	
水平位置	元件的水平起始位置	不要超出水平分辨率	
垂直位置	元件的垂直起始位置	不要超出垂直分辨率	
宽度	元件的宽度	/	
高度	元件的高度	/	

初级的 CAD 使用可以参照第八章【CAD 导入矢量图片】

4.3.25. 三次文件编辑器

介绍:在 Hmi 界面上进行三次程序开发的元件,支持编辑和显示程序等。需配合三次文件指令使用。 **使用方法:**点击菜单栏"HMI"→"控件箱"→"三次文件编辑器"。将该元件放至合适位置,设置通道号 后,配合其他元件和 Basic 程序,实现程序的导入和实时编辑(编辑需在"属性"窗口中将"可编辑"切换为 True)。三次文件编辑控件"选中多行"功能,可通过鼠标按下滑动或者通过指令进行操作。支持对选中 部分进行删除、编辑。

ABC.nc	Row 1	Co1 1	Lns 11111
1G01 X0.000	YO. 000 ZO. 000		
2G01 X0.000	YO. 000 ZO. 000		
3G01 X0.000	YO. 000 ZO. 000		
4G01 X0.000	YO. 000 ZO. 000		
5G01 X0.000	YO. 000 ZO. 000		
6G01 X0.000	YO. 000 ZO. 000		
7G01 X0.000	YO. 000 ZO. 000		
8G01 X0.000	YO. 000 ZO. 000		
9G01 X0.000	YO. 000 ZO. 000		

1. 属性窗口:



属性		×		行号宽度	30	
▲ 基本属性				文件信息	True	
	1			垂直滚动条	False	
元件名称	File3Edit1			水平滚动条	False	
	底层		- 4	标签		
	显示			格式文本(0)	Line number	
	False			格式文本(1)	Code	
	0			格式文本(2)	Select	
	False			格式文本(3)	File info	
安全时间ms	0		- 4	位置和尺寸		
	False			水平位置	33	
⊿ 外观	1			垂直位置	53	
会制边框	False			宽度	400	
 图片来源	无			高度	200	
行间距	2					
行号	True		轴	参数帮助属性		

属性	功能	说明
元件编号	/	/
元件名称	/	/
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但点 击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 通过寄存器控制元件 是否显示
有效的设备编号 (有效控制为 True)	设备编号	默认 local
有效的寄存器类型 (有效控制为 True)	选择寄存器类型	多种寄存器下拉列表选择
有效的寄存器编号 (有效控制为 True)	选择寄存器编号	寄存器值为0时不显示,非0时显示
通道号	设置三次文件通道号	最多只支持三个通道: 0/1/2
可编辑	是否启用元件的编辑功能	默认 False
软键盘窗口号	数据元件编辑时使用的软键盘窗口	可编辑选择 True 时存在的属性
安全时间 ms	最少按键时间	单位 ms
修改时通知	修改后发出 bit 位通知	默认 False,选择 True 时选择通知的寄存器



	(设 ON 或 OFF)	
通知的设备编号		
(修改时通知为 True)	[默认 local
通知的寄存器类型	业权安大田光利	发出安古明了达过去世权
(修改时通知为 True)	选择奇仔畚尖型	多种命仔盗下拉列衣选择
通知的寄存器编号	业权安方即位日	通过获取寄存器不同数值,控制元件不同状
(修改时通知为 True)	」 近洋	态
绘制边框	选择是否绘制边框	/
图片来源	无、背景图片库或背景图片	无图片、背景图片库或背景图片中选择
背景图片	背景图片选择	图片来源设置为选择背景图片显示
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示
行间距	设置行与行之间的距离	默认为2
行号	选择是否显示行序号	/
行号宽度	设置行序号位置宽度	行号设置为 True 时显示该属性
文件信息	设置是否显示标题行	/
垂直滚动条	设置是否使用垂直方向滚动条	/
水平滚动条	设置是否使用水平方向滚动条	/
格式文本 0	设置行号(Line number)样式	文本名称不可更改
格式文本 1	设置编程区域(Code)样式	文本名称不可更改
格式文本 2	设置编程区域选中状态(Select)样式	文本名称不可更改
格式文本 3	设置文件信息行(File info)样式	文本名称不可更改
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

初级的三次文件编辑器使用可以参照第八章【CAD结合三次文件使用】。

4.3.26. 报表视图

介绍: [报表]元件以表格的形式呈现多组数据,用于显示和管理报表数据。在[报表]元件中,用户可以 查看和编辑报表数据,支持字符和中文输入。

注意:使用该控件需同时确保 RTHmi 版本为 V1.3.0 版本及以上和 RTSys 版本为 V1.2.02 及以上版本!

使用方法:点击菜单栏"HMI"→"控件箱"→"报表视图"。将该元件放至合适位置,设置行列数、是否可编辑、是否显示表头行/列等属性,然后点击"单元格数据",对报表内容进行编辑,设置完成、保存下载后,可以查看(和编辑)报表数据。

1:R	eport Niew	В	С	
1				
2				
3				

1. 属性窗口:

			_				
属性		×		垂直滚动	条	False	
安全时间ms	0	*		水平滚动	条	False	
⊿ 外观			4	标签			
	3			文本库			
行高	20			格式文本	(0)	Cell	
列数	3			格式文本	(1)	Header row	
显示表头行	True			格式文本	(2)	Header colu	
	20			单元格数	据	Modify	
显示表头列	True		4	动作		-	
表头列宽	20			点击调用	函数		
选中部分颜色	99CCFF			修改调用	函数		
外框颜色	000000		4	位置和尺	ব		
外框线宽	1			水平位置		0	
外框线段类型	默认			垂直位置		205	
网格颜色	000000			宽度		400	_
网格线宽	1			高度		200	
网格线段类型	默认						-
固定列宽	False		_				
固定行高	False		轴	参数 属性	帮助		

属性	功能	说明
元件编号	/	/
元件名称	/	/
		顶层:显示在最外层,覆盖底下元件
显示层次	选择元件显示层次	中层: 被顶层元件覆盖, 覆盖底层元件
		底层: 被顶层和中间层元件覆盖(默认)
有效显示	选择元件是否显示	显示:该元件显示于界面之中

		不显示: 该元件不显示于界面之中		
		仅显示,不可用:该元件可正常显示,但		
		点击该元件时动作不生效		
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 通过寄存器控制元		
		件是否显示		
可编辑	Ⅰ 报表视图元件是否启用输入	默认 False,选择 True 调用软键盘窗口输		
		λ		
软键盘窗口号	数据元件编辑时使用的软键盘窗口	可编辑选择 True 时存在的属性		
安全时间 ms	最少按键时间	单位 ms		
行数	设置表格行数	不包含表头行,最多可设置 512 行		
		每一行的行高可以在下载后的 Hmi 界面拖		
		动修改		
行主	每一行的县 元 行享度(1 n 行)	每一列的列宽可以在"单元格数据"界面		
11 [□]		→"文件头设置"框→"宽度"修改,或		
		者直接拖动修改,再或者下载后的 Hmi 界		
		面拖动修改		
列数	设置表格列数(1~n行)	不包含表头列,最多可设置16列		
日二主礼行	True: 显示表头列	^略 とは とってmage		
並小衣头们	False: 不显示表头列	款认为True		
表示行高	设置表头行的高度	/		
見云書礼列	True: 显示表头列	野山 为 True		
业小农天勿	False: 不显示表头列	A K/Y Hue		
表头列宽	设置表头列的宽度	/		
选中部分颜色	设置列表元件选中部分的颜色	默认		
外框颜色	设置表格外框颜色	默认黑色		
外框线宽	设置表格外框线宽	/		
外框线段类型	设置表格外框线段类型	默认实线		
网格颜色	设置表格内框线颜色	默认黑色		
网格线宽	设置表格网格的线段宽度	默认宽度为1,最大为20		
网格线段类型	选择绘制网格框的线段的样式	默认实线		
田ウ利幸	True: 列宽固定			
凹 定 列	False: 可以拉伸列的宽度	款认为 Faise		
田內仁古	True: 行高固定			
」 回 疋 仃 尚	False: 可以拉伸行的高度	款认为 False		
垂直滚动条	设置是否使用垂直方向滚动条	/		
水平滚动条	设置是否使用水平方向滚动条	/		
格式文本 0 (Cell)	设置单元格显示文本格式	文本名称不可更改		
格式文本 1	·九里招主关气势投-2	· ナナタルズ ゴ 西 ル		
(Header row)	[ប <u></u> 直拔衣自仃的杆式	义		
格式文本 2	い四切まそれの投入	· ナナタルズ ゴ 西 ル		
(Header column)		又半名称个可史改		

	编辑单元枚内交	
	与′′′ 新开,前元枚昌士且元字符新	
	于竹致: 半儿俗取入亚小于竹数,	
	最大限制 255(使用字符或数值显示	
	时都会受字符数大小的限制)	
	数据类型: 限制输入类型,选择	
	"字符显示"或者"数值",	
	= 字符显示时为空;	
单元格数据	= 数值时以下属性有效:	/
	小数位数: 限制输入的有效小数位	
	数, 0~13	
	是否限制范围: 勾选限制,勾选时	
	下列属性有效: (最小值<最大值有	
	效)	
	最小值: 限制输入最小值	
	最大值: 限制输入最大值	
	按键按下时调用 Basic 定义的 SUB	函数必须是 GLOBAL 全局类型
点击调用函数	函数	
收立) 加田 <u>承</u> 粉	修改完成后触发调用的 Basic 定义的	函数必须是 GLOBAL 全局类型
修以响用函数	SUB 函数	
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

示例:

1. 在报表视图属性-外观栏设置行数和列数,以及行高和列高,不包括表头行和表头列。

2. 设置报表是否可编辑,是否显示表头行/列,行(列)的高(宽),格式文本以及选择外框颜色、线 宽等属性(在此例中设置为"可编辑",显示表头行/列)。

Basic1.bas 🗄 Hmi1.hmi × 🔹	属性	→ ₽ ×
	▲ 基本属性	
	元件编号	1
	元件名称	Report View1
	显示层次	底层
	有效显示	显示
	采用有效控制	False
1:Report Niew B C 🛶 表头行	可编辑	False 🔹
1	安全时间ms	0
2	⊿ 外观	
3	行数	3
	行高	20
表头列	列数	3
	显示表头行	True
	表头行高	20
	显示表头列	True
	表头列宽	20
	选中部分颜色	99CCFF
	外框颜色	000000
	外框线宽	1
	外框线段类型	默认
	网格颜色	000000
	网格线宽	1
	可编辑	
	轴参数 属性 帮助	

3. 点击"单元格数据",设置文件头(表头行)文本。

₿ Ba	asic1.b	as	∄Hn	ni1.hmi ×		*	属	性		*	џ	×
1:Repo	⊈rt Viegw	С	D		A			固定行高	False			*
1								垂直滚动条	True			
2								水平滚动条	True			
4							- 4	标签				
5								文本库				
6						•		格式文本(0)	Cell)	
		•		-				格式文本(1)	Header row			
								格式文本(2)	Header column			j
								单元格数据	Modify			
								动作				-
				•		•	单	元格数据				
							编	崔单元格内容。				
							轴	参数 属性 帮助]			



A	В	С	D			- 数据类型	
						 ● 字符显示	□ ○ 数值
						字符数:	8
						1.03%	Ľ
						─数值参数一	
						小数位数:	0
						□ 限制范围	Line and the second sec
						最大值:	0.0
						最小值:	0.0
					\mathbf{X}	- 文件头设置	
						选择:	0
						文本:	Α
						宽度:	80
						L	

4. 编写表格内容。数据类型选择"字符显示"时,只需设置字符数;数据类型选择"数值"时,还可 设置该表格数值的小数位数及限制范围等数值参数。如果设置了小数位数,但小数位没写或者写的是0,则 该参数的小数位只会在下载后的仿真界面显示。

抽数 抽类型 脉冲当量 速度 0 虚拟轴 - - - - - - - </th
0 虚拟轴 0 虚拟轴 0 虚拟轴 0 0
Image: Section of the section of t
<t< td=""></t<>
▲ ▲ ▲ ▲ 数值参数 ▲ ▲ ▲ ▲ ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●
小数位数: ● ● ● <td< td=""></td<>
□ 限制范围 最大值:
最大值:
最小值:
选择:
文本:
宽度:



5. Hmi 窗口报表视图效果图如下:

B	Basi	c1.bas 🛛 Hmi1.hm	ni ×		~	厚	属性		*	џ	×
1 : F	e軸数	View 轴类型	脉冲当量	速度	~		固定行高	False			٠
1	0	0(虛拟轴)	10	100			垂直滚动条	True			
2	1 1(脉冲方向方式的步进或伺服)		1000	500			水平滚动条	True			
3	2	3(正交编码器)	1	10			标签				
4	3	6(脉冲方向方式的编码器)				-	PJ-M				
5	4						文本库				
6	5				- E		格式文本(0)	Cell)	
							格式文本(1)	Header row)	
							格式文本(2)	Header column)	
							单元格数据	Modify)	
4						-	动作				-
						单	元格数据				
						编辑单元格内容。					
					轴参数 属性 帮助						

6. 连接控制器,将 Hmi 文件下载到控制器,可看到 xplc screen 窗口显示效果图。当元件属性的"固定 列宽(行高)"为 false 时,在下载后的仿真界面中,用户可以通过将鼠标悬停在表头行上来调整每列的宽度,或者将鼠标悬停在表头列上来调整每行的高度。

xplc sc	reen				×
	轴数	轴类型	脉冲当量	速度	
1	0	0(虚拟轴)	10	100.00	
2	1	1(脉冲方向方式的步进或伺服)	1000	500.00	
3	2	3(正交编码器)	1	10.00	
4	3	6(脉冲方向方式的编码器)			
5	4				
6	5				
4					

如下图所示,双击单元格,可对表格内容进行编辑。表头行和表头列的内容只能在属性"单元格数据" 中编辑,不支持在线编辑!

 \times

xplc sc	reen	_								_						×
	轴数	轴数 轴类型				月	永冲当	量	速度							
1	0	0 0 (虚拟轴)					10		100.00							
2	Esc	Esc 200								Ва	Backspace					
3 4	Sym	1	2	3	4	5		ò	7	8		9	0		+	
5	Tab	q	w	e		r	t	у		u	i		0	р	?	
6	Cap	s	a	s	d	f		ş	h	j		k	1	,	"	
	EN/CH z x c v b n m , .								Enter							
	C1ear	De1					2	Space						•		
•																

xpic sc	xpic screen							
	轴数	轴类型	脉冲当量	速度				
1	0	0(虚拟轴)	10	100.00				
2	1	1(脉冲方向方式的步进或伺服)	1000	500.00				
3	2	3(正交编码器)	1	10.00				
4	3	6(脉冲方向方式的编码器)	200					
5	4							
6	5							
4								

4.3.27. 文件浏览器

介绍:显示当前目录、以表格形式显示文件内容,需要指定显示格式(必选文件名称、可选文件大小、修改时间、文件类型)。鼠标双击文件夹进入,鼠标双击文件打开文件(调用指定 SUB)。

注意:使用该控件需同时确保 RTHmi 版本为 V1.3.0 版本及以上和 RTSys 版本为 V1.2.02 及以上版本!

使用方法: 点击菜单栏"HMI"→"控件箱"→"文件浏览器"。将该元件放至合适位置,调用指定 Basic 程序的全局 SUB 函数,实现文件的选择与打开。

Patthe Brower				
File name	Size	Туре	Modified	time
<mark>.</mark>		Folder	2024/9/4	8:00
- Folder1		Folder	2024/9/4	9:00
- Folder2		Folder	2024/9/4	10:00
🗋 Filel.z3p	36KB	File	2024/9/4	11:00
🗋 File2.nc	512B	File	2024/9/4	12:30

1. 属性窗口:

属性		×		显示文件类型	True	
▲ 基本属性				文件类型列宽度	60	
一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一	1			显示文件修改时间	True	
一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一	File Brower1	- 11		文件修改时间列宽度	150	
显示层次	库厚	- 11		行间距	2	
有效显示	見示	- 11		垂直滚动条	False	
	「日XX业小 业小 亚田右対抗制 Falce			水平滚动条	False	
双击调用函数	双丰调用函数			标签		
选中调用函数		- 11		文本库		
· 外观				格式文本(0)	File list	
	True			格式文本(1)	Title	
	000000			格式文本(2)	Select	
	无			位置和尺寸	, ,	
显示路径	True			水平位置	0	
	True			垂直位置	131	
	20	-		宽度	399	
文件名列宽度	140			高度	200	
显示文件大小	True	-	_			
文件大小列宽度	45	-	報	的多数 属性 帮助		

属性	功能	说明
元件编号	/	/
Zmotion[®]

元件名称	/	/
		顶层:显示在最外层,覆盖底下元件;
显示层次	选择元件显示层次	中层: 被顶层元件覆盖, 覆盖底层元件;
		底层: 被顶层和中间层元件覆盖(默认)。
		显示:该元件显示于界面之中;
本 港日二	<u> </u>	不显示: 该元件不显示于界面之中;
有双亚不	远挥兀忤定省並示	仅显示,不可用:该元件可正常显示,但
		点击该元件时动作不生效
亚田古动坛甸	通过安方器按制元件具不目二	默认 False,选择 True 通过寄存器控制元件
太 用有双控刺	通过可行福江前几件走百亚小	是否显示
<u>如</u> 土润田丞粉	双击时触发调用 Basic 定义的	函数必须是 GLOBAL 全局类型
X 击 响 用 函 致	SUB 函数	
送 山 田 瓜 粉	选中操作时触发调用的 Basic 定	函数必须是 GLOBAL 全局类型
见中,师用函数	义的 SUB 函数	
绘制边框	选择是否绘制边框	/
图片来源	无、背景图片库或背景图片	无图片、背景图片库或背景图片中选择
背景图片	背景图片选择	图片来源设置为选择背景图片显示
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示
显示路径	是否显示文件路径	默认为 True
显示标题	是否显示标题	默认为 True
标题高度	标题的高度	/
文件名列宽度	文件名列的宽度	/
显示文件大小	是否显示文件大小	默认为 True
文件大小列宽度	文件大小列的宽度	/
显示文件类型	是否显示文件类型	默认为 True
文件类型列宽度	文件类型列的宽度	/
显示文件修改时间	是否显示文件修改时间	默认为 True
文件修改时间列宽度	文件修改时间列的宽度	/
	显示每一行的上下间距,包括	
(行) (1) (1) (1) (1) (1) (1) (1) (1) (1) (1	文件列表及当前路径行	▶1.1.1.2. 可沿罢为 0, 100
1〕1町屹2	如字体大小=16,行间距=4,则	款认为2,可以直为0~100
	每一行高度=16+2*4=24	
垂直滚动条	设置是否使用垂直方向滚动条	/
水平滚动条	设置是否使用水平方向滚动条	/
枚式文本 0 (File list)	设置文件列表行的字体格式,	立木夕我不可再步
俗式文本 0(File list)	包括对当前路径行的设置	文平石林小可更以
	设置当前文件列表标题行的字	立木夕我不可再改
TTULX年1(IIIIC)	体格式	入平石你个时丈以
枚式文本 2(Salaat)	设置当前文件列表选中行的字	立木夕称不可再改
111111111111111111111111111111111111	体格式	X 平 石 柳 个 刊 史 以
水平位置	元件的水平起始位置	不要超出水平分辨率



垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

初级的三次文件编辑器使用可以参照第八章【<u>文件浏览器使用</u>】。

4.3.28. 菜单

介绍: 类似 window 风格的菜单栏,最多只能嵌套 5 层菜单。点击弹出菜单项,每一个菜单项可以设置是否禁用(不使能则灰阶显示),点击菜单项可触发调用 SUB 动作,每一个菜单项还可以设置是否选中状态。鼠标悬停处的菜单项将处于高亮状态。可在任意菜单项上面增加分隔符。

注意:使用该控件需同时确保 RTHmi 版本为 V1.3.0 版本及以上和 RTSys 版本为 V1.2.02 及以上版本!

使用方法:点击菜单栏"HMI"→"控件箱"→"菜单"。将该元件放至合适位置,打开菜单属性,设置列 表文本内容;然后根据各项的菜单编号在 Basic 文件中编写与之对应的全局 SUB 函数,以实现各菜单项的 具体操作;最后,在菜单属性中调用已经编写好的 SUB 函数,以完成菜单功能的绑定与实现。



1. 属性窗口:

属性		×		选中部分颜色		99CCFF	
▲ 基本属性			_	标签			
	1			文本库			
	Mopu1	1		格式文本(0)	Le	vel 0	
		- 11		格式文本(1)	Su	Ibmenu	
		- 11		列表文本	M	enu list	
有效显示	显示	- 11	-	位置和尺寸			
采用有效控制	False			水平位置	93		
下拉调用函数				垂直位置	14	0	_
选中调用函数			_	宽度	20	0	
▲ 外观			_	高度	20		
绘制边框	True			1-5/54			
边框颜色	787878						
行间距	4		轺	参数 属性 帮助			

2. 属性说明:

属性	功能	说明
元件编号	/	/
元件名称	/	/
		顶层:显示在最外层,覆盖底下元件;
显示层次	选择元件显示层次	中层: 被顶层元件覆盖, 覆盖底层元件;
		底层: 被顶层和中间层元件覆盖(默认)。

Zmotion[®]

有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但 点击该元件时动作不生效
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 通过寄存器控制元件 是否显示
下拉调用函数	进行下拉操作时触发调用 Basic 定义的 SUB 函数 注:可在该动作函数内对子菜 单项的灰阶状态、选中状态进 行管理	函数必须是 GLOBAL 全局类型
选中调用函数	弹出菜单项区域时,点击选中 任意一个有效的菜单项时触发 调用的函数 注:可在该动作函数判断最后 点击选中项并进行相应的动作 绑定动作 SUB_XXXX,但定义 动作时需要传入一个 ID 参数, 如 SUB_XXXX(ID), ID 表示最 后点击选中菜单项的编号。	函数必须是 GLOBAL 全局类型
绘制边框	选择是否绘制子菜单的边框	/
边框颜色	选择子菜单边框的颜色	默认灰色
行间距	设置子菜单行与行之间的距离	每一行菜单项上下的行距,例如字体大小 =16,行距=4,那么每一行菜单项的行高等 于 16+4*2=24 默认为 4,可设为 0~100
选中部分颜色	主菜单按钮和所有子菜单项的 选中高亮背景颜色 鼠标悬停在菜单按钮或菜单项 时,背景颜色进行高亮显示	默认蓝色
文本库	文本库的名称,如果为空,则 表示使用文本标签	/
格式文本 0(Level 0)	主菜单按钮的显示格式 在 RTSys 传入的控件大小即为 菜单按钮的大小,菜单项区域 大小是根据菜单项数量,即相 关属性自动计算并显示的	文本名称不可更改
格式文本1(Submenu)	子菜单项的显示格式,包括所 有级别的子菜单项 该格式文本的背景颜色作为整 个弹出菜单区域的背景颜色	文本名称不可更改



	限制只能左对齐,不支持右对	
	齐、居中对齐	
列表文本(Menu list)	设置各项菜单项的文本	/
水平位置	元件的水平起始位置	不要超出水平分辨率
垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

示例**:**

1. 在菜单属性 - 标签栏点击 "列表文本"打开。

	属性
	采用有效控制 False 4
	下拉调用函数
	选中调用函数
	⊿ 外观
	绘制边框 True
	边框颜色 787878
1:Menu 1 V O	行间距 4
Menu 1 Menu 2	选中部分颜色 99CCFF
	⊿ 标签
	文本库
	格式文本(0) Level 0
	格式文本(1) Submenu
	列表文本 Menu list
	▲ 位置和尺寸
	水平位置 66
	垂直位置 162 •
	轴参数 属性 帮助

2. 编辑菜单项内容,每项都对应唯一的菜单编号,菜单编号按其编写顺序编号。

莱单设计	×
Menu 1 Menu 2 请在此处键入	□禁用
请在此处键入	□勾选
	插入分隔符
	菜单编号:
	1000
	自动编号
	插入
	确定
	取消

注意:菜单编号支持手动设置和自动设置,但所用的编号不可重复!菜单编号可设置的最大值为:65535。 "自动编号"功能会自动寻找未使用的编号,可保证编号不重复,但手动设置则无法检测是否有冲突。

选中一项,点击"插入分隔符",分隔符将插入选中项的上方,其菜单编号为固定值:65535。

## ●## 第2 第4日 第2 第4日 第2 第4日 第4日 ●303 第4日 ●303 第4日 ●303 第4日 ●303 第4日 ●303 第4日 ●303 第4日 ●304 第4日 ●305 ●1005 ●104 ●11 ●10	报表 Menu2 请在此处键入 新建	□禁用
● 303 第五世汉道入 第五世汉道入 第五世汉道入 第五世汉道入 1005 自幼帰号 16人 1005 自幼帰号 1004 1004 1005 自幼帰号 1004 1004 1005 自幼帰号 1004 1004 1005 自幼帰号 1004 1004 1005 1005 1004 1004 1005 1004 1005 1004 1004 104 1005 1005 1005 1005 1005 1004 1005 1005 1005 1004 1004 1004 1005 1005 1005 1005 1006 1004 1007 1004 107 1004 108 1047 108 1047 1048 1048 1049 1044 1049 1044 1040 1044 104	新建	
保存 前点公用符 等出 前点公開符 第二 前点 節 前点 前点 前点 節 前点 前点 前 前< 前 前 日 前 日 前 日 前 日 前 日 前 日 前 日 前 日 前 日 前 日 前 日 前 日 前	打开	□勾选
第二 国営业党組入 第合社が強入 日初帰号 話入 開路 月 ● ○ → 中 ○ → + 第 ● 第 ● 第 ● 第 ● 第 ● ● → ● → ● → ● → ● → ● → ● → ● → ● → ● → ● → ● → ● → ● → ● → ● → ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	保存	插入分隔符
请在此处望入 005 自幼帰号 道入 一 ● ● ●		
自胡帰号 議入 一日 (1) (2) (2) (2) (2) (2) (2) (2) (2) (3) (3) (4) (5) (4) (4) (4) (4) (4) (5) (5) (4) (5) (5) (4) (5)	请在此处键入	1005
振為 第第 1		自动编号
第時 1		插入
● ●		冊版余
第二 第二 <th></th> <th></th>		
第単设计 確定 東海 取消 第単设计 X 「「「「」」」」 第二 「「」」」 「」」 「「」」」 「」」 「」」」 「」」 「」」」 「」」 「」」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」 「」」 「」 「」」 「」 「」」 「」 「」」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「 「」		
第定 取消 東牟公士 第年 第2 第4 第二 第二 <		
確定 取消 取消 取消 第単设计 X 「「「「「」」」」 「「」」」 「「」」」 「「」」」 「「」」」 「「」」」 「「」」」 「「」」」 「」 <th></th> <th></th>		
確定 取消 取消 第単设计 第単设计 第 前2 第2 第4 日本 第二		
確定 取消 取消 取消 業単设计 × 「振素 Menu2 」 请在此处違入 「耕用 「勾选 」「「」」」 「「「」」」 「「」」」 「「」」」 「「」」」 「「」」」 「「」」」		
東洋 東洋 東牟设计 × 東東 「百在此处键入」 「「百在此处键入」 「「二」 「「二」 「二」 「二」		确定
東牟设计 × 「振表」Memu2」 「斎在此处键入」 「新建 打刀 保存 导入 - 写出 - 语在此处键入 ●禁用 - 句法 - 插入 - 新路 - 插入 - 新島 「斎在此处键入 ● 「斎在此处键入 ●		取消
東美 Menu2 请在此处違入 二 「新建 二 二 「新建 二 二 「新建 二 二 「日本 二 二		
★年度日 振 Memu2 请在此处键入 ●禁用 新建 打开 句法 打开 保存 插入分隔符 导出 请在此处键入 ● 请在此处键入 ● ●		×
新建 新建 保存 句法 插入分隔符 専入 导出 请在此处键入 请在此处键入 自动编号 「 (本)	*+*** Wenn2 「请在此外違入」	□
打井 保存	新建	
	1 1 1 1 保存	括入分隔在
守山 東車瑞令: 请在此处键入 65535 自动编号 抵入 删除 ↑ ← ○ →	导入	
自动编号 插入 册除 ← ○ →	与出 「清在此处键入」	张里调写 : [65535
		自动编号
		冊修余
		$(\leftarrow \bigcirc \rightarrow$
		+
确定		

选中一项,点击"勾选",选中项前面会出现一个"√"。也可直接通过控件操作指令 <u>HMI_MENUITEM</u> 中的参数对其进行设置,"设置选中状态",下载实际操作时可对此项进行勾选。

报表 文件 编辑 帮助 请在此处键入 □	
日言 ✓ 次语 後用手册 ✓ 英语 检查更新 日语 关于 (後语) 反馈 请在此处键入 请在此处键入 第4	 禁用 (本) (本) (本) (本) (本) (本) (本) (本) (本) (本)

选中一项,点击"禁用",选中项将变为灰色,实际操作时无法选中。也可直接通过控件操作指令 <u>HMI_MENUITEM</u>中的参数对其进行设置,"设置灰阶状态"。

菜单设计	×
报表 文件 编辑 帮助 请在此处键入 语言 // 汉语 使用手册 // 英语 检查更新 日语 关于 微微 请在此处键入 请在此处键入	 ♥ 禁用 □ 勾迭 插入分隔符 菜单编号: 1034 自动编号 插入 ● 酬除 ↑ ↑ ↓



选中一项,点击"插入",插入项将插入选中项的上方。

报表 文件 编辑	帮助 请在此处键入	□禁用
	语言 使用手册 检查更新 反馈 请在此处键入	 □ 勾选 插入分隔符 菜单编号:
		1024
		自动编号
		插入
		删除
		确定
		取消
设计		

服表	文件	编辑	帮助 请在此处键入	□禁用
			语言 使用手册 1	□勾选
			检查更新	插入分隔符
			请在此处罐入 「请在此处罐入	菜单编号: 1034
				自动编号
				插入
				刪除
				确定
				面当

编辑完成后,若点击"自动编号",系统首先将会按照从左至右的顺序对主菜单进行编号,随后再依次 从左至右、从上至下对下一级子菜单进行编号,如此逐级类推,直至全部编号完成,图示如下:

^{菜单设计} ①	×
报表 文件 编辑 帮助 请在此处键入	□禁用
 新 第 3 4 1 1	 □ 勾选
	 ← ○ → → →<

3. 菜单各项内容编辑完成后,点击"确认",Hmi界面只会显示主菜单项。根据需求,在菜单基本属 性中选择对应 Basic 全局 SUB 子函数调用。相关指令可查看"<u>HMI MENUITEM</u>"。

				属	性			×
				- 4	基本属性			+
					元件编号	1		
					元件名称	Me	enu1	
					显示层次	底	킂	
					有效显示	显	T	
					采用有效控制	Fa	se	
		-			下拉调用函数			
^{1:1} 搬表	文件	编辑	帮助		选中调用函数			
-			•		外观			
					绘制边框	Tru	le	
					边框颜色		787878	
					行间距	4		
					选中部分颜色		99CCFF	
				- A	标签			
					文本库			
					格式文本(0)	Le	vel 0	 •
				轴参	数 属性 帮助			

Zmotion[®]

"选中调用函数" Basic 全局 SUB 函数举例:

工程视图	× Basic1.bas × Hmi1.hmi	*
▲ C 菜单栏 — B Basicl.bas — H Hmil.hmi[1] — 2.jpg — 04.jpg — 5.jpg — STSONG.TTF — eriblk.ttf — BOD_R.TTF — calibrib.ttf — 歐武文件	end ''结束主程序 '' 菜单项选中函数(部分功能演示) '' 传入一个参数(菜单编号) global sub Sub_MenuSelect(menuid) '' 小型菜单项点击事件 '' 处理菜单项点击事件 '' 检理菜单项点击事件 '' 告告: '' 估害: '' 告告: '' 告告: '' 告告: '' 语言只能选择一种,取消其他的,再选中当前的 '' 语言只能选择一种,取消其他的,再选中当前的 '' 语言只能选择一种,取消其他的,可选中当前的 '' 语言只能选择一种,取消其他的,可选中当前的 '' 语言只能选择一种, 0, 0) '' 语言只能达择一种, 0, 1) '' 语言只能达择一种, 0, 1) '' 语言只能UITEM(1033, 0, 0) '' HT_TENUITEM(1034, 0, 0) '' HT_TENUITEM(1034, 0, 0) '' endif	
工程视图 标签视图 组态视图	I 29 end sub	•

源代码:

end "结束主程序

```
"菜单项选中函数(部分功能演示)
"传入一个参数(菜单编号)
global sub Sub_MenuSelect(menuid)
 ?"点击了菜单项",menuid
 "处理菜单项点击事件
 if menuid = 1005 then
   HMI SHOWWINDOW(11) "打开报表
 elseif menuid = 1027 then
   HMI SHOWWINDOW(12) "打开文件浏览器
 elseif menuid >= 1031 AND menuid <= 1034 then "选择语言
   "语言只能选择一种,取消其他的,再选中当前的
   HMI MENUITEM(1031, 0, 0)
   HMI_MENUITEM(1032, 0, 0)
   HMI_MENUITEM(1033, 0, 0)
   HMI MENUITEM(1034, 0, 0)
   HMI MENUITEM(menuid, 0, 1)
 endif
end sub
```

4. 连接控制器,将 Hmi 文件下载到控制器,可看到 xplc screen 窗口显示效果图。



打开报表:

x	plc screen			
	报表	文件	编辑	帮助
	新建		****	
	打开			
	保存			
	导入			
	导出			

olc sci	reen						
	配方名	面粉/g	水/m1	鸡蛋/个	盐/g	蜂蜜/g	
1	甜面包	100.0	50.0	3	5.0	35.0	
2	无糖面包	100.0	65.0	2	3.5	0.0	
3	全麦面包	100.0	55.0	0	5.0	0.0	
4							
5							
6							
7							
8							

选择语言:

хр	lc screen						
	报表	文件	编辑	帮助			
				语言		>	汉语
				使用手册			英语
				检查更新			日语
				关于			俄语
				反馈			

程序中对语言选择进行了限制,即一次只能选择一种,选中了一种语言后,换成另一种语言时,会将当 前选中项置为"未选中"状态。被禁用的菜单项无法选中。

xplc screen					
报表 文件	编辑	帮助			
		语言	>	✔ 汉语	
		使用于	戶册	英语	
		日语			
		关于	俄语		
		反馈			

xpic screen						
报表	文件	编辑	帮助			
			语言		>	汉语
	使用手册					✓ 英语
			检查更	巨新	日语	
			关于		俄语	
			反馈			

5. 菜单控件的文本设置若使用文本库,则其内容将按照层次结构依次填充至各菜单项中。

文本库																		×
语言选	择:	0	• 0	○以语	言方式显	示	◉ 以状症	的方式显示	0-15	5 •		导入			导出		字体管理器	
字体设计	置:																	
		语言 0	ì	语言:	1	语言 2	:	语言 3	语言	4	语言	5	语言 6		语言 7			
字体		默认		默认		默认	E	默认	默认		默认		默认		默认			
代码页	ī	0	1	1252		932	9	50	0		0		0		0			
行 然识:	里·																搜索	
编号	山· 标签复	2	引用次	*浙	状态频	语言新	格式文オ	1. 壮态 በ		状态 1		状态 2		壮态 :	3	壮态	4	14
1	label 1	•	1	.82	5	4	占击设署	· 向左		向右		向上		向下		不变	-	10.
2	label2		1		7	4	点击设置	向上		向上5行		向上10行		向下		向下5	íī	向
3	label3		0		40	1	点击设置	1		2		3		4		5		6
4	双击添	:加																
4																		•
	-																	
	除	清除	Ŧ												确注	Ē	取消	





如下图所示,当使用文本库时,其状态内容将从主菜单的第一项开始,依次填充至该项下的所有子菜单中(分隔符也占一个文本库状态),逐层深入直到填满后,再填入主菜单的第二项,以此类推。



4.3.29. 树形图

介绍:树形图是一种以树状结构展示所有**表项**(树形图第一层的树节点)的控件,支持单击角图标展 开/缩起子树,单击树节点内容触发动作。当树节点数量超出显示范围时,自动启用垂直滚动条,便于用户 浏览和管理。

注意:使用该控件需同时确保 RTHmi 版本为 V1.3.0 版本及以上和 RTSys 版本为 V1.2.02 及以上版本!

当前版本的树形图控件不支持在下载后的仿真界面动态修改、添加、清空树节点,只能在 Hmi 文件 内设计好所有的树节点。

使用方法:点击菜单栏"HMI"→"控件箱"→"树形图"。将该元件放至合适位置,在列表文本里进行树 形图的结构设计,每一树节点都绑定一个唯一的 ID。选中树节点时,其关联的寄存器(状态)值将自动设 为该项 ID;修改寄存器值时,对应 ID 的树节点会被选中。通过树形图属性框"选中调用函数",调用相 应动作的全局 SUB 函数,实现单击树节点内容进行触发动作的功能。

1√TT€fem1	
Item1.1	
Item1.2	
Item2	

3. 属性窗口:

属性	:	×	角标类型		样式1		
▲ 基本属性			角标颜色		000000		
	1		行间距		2		
	Tree1		选中部分颜色	ļ	99CCFF		
	店 同		⊿ 标签				
	月一		文本库				
	SIE/JN Falso		格式文本		ltem)	
本川有双投制	rdise		列表文本		ltem		
			▲ 位置和尺寸				
	0	1	水平位置		180		
绑定的寄存器编号	0		垂直位罟		70		
选中调用函数		-	宽度		200		
▲ 外观		-	高度		250		
绘制边框	True	=	19/52		250		•
边框颜色	000000						
图片来源	无		轴参数 属性 帮助	b			



4. 属性说明:

属性	功能	说明			
元件编号	/	/			
元件名称	/	/			
显示层次	选择元件显示层次	顶层:显示在最外层,覆盖底下元件; 中层:被顶层元件覆盖,覆盖底层元件; 底层:被顶层和中间层元件覆盖(默认)。			
有效显示	选择元件是否显示	显示:该元件显示于界面之中; 不显示:该元件不显示于界面之中; 仅显示,不可用:该元件可正常显示,但 点击该元件时动作不生效			
采用有效控制	通过寄存器控制元件是否显示	默认 False,选择 True 通过寄存器控制元件 是否显示			
绑定的设备编号	指定设备	默认 local			
绑定的寄存器类型	选择寄存器类型	多种寄存器下拉列表选择			
绑定的寄存器编号	设置寄存器的编号	通过获取寄存器不同数值,控制元件不同 状态			
选中调用函数	指定点击选中选项时触发调用的 SUB函数	函数必须是 GLOBAL 全局类型			
绘制边框	选择是否绘制边框	为 TRUE 时有属性:边框颜色			
边框颜色	选择边框的颜色	默认灰色			
图片来源	背景图片或背景图片库	/			
背景图片	背景图片选择	图片来源设置为选择背景图片显示			
背景图片库	背景图片库选择	图片来源设置为选择背景图片库显示			
角标类型	父树角标的显示类型,支持样式 1~样式3	缩起时: > ▶ ▶ ▲ 展开时: ~ ▼ ▲			
角标颜色	显示角标的颜色				
行间距	设置行与行之间的距离	每一行菜单项上下的行距,例如字体大小 =16,行距=4,那么每一行菜单项的行高等 于 16+4*2=24 默认为 2,可设为 0~100			
选中部分颜色	鼠标悬浮在树表项上或树表项被 选中时,该项高亮显示的颜色	默认蓝色			
文本库	文本库的名称,如果为空,则表 示使用文本标签	/			
格式文本 (Item)	设置树形图标签的样式	文本名称不可更改			
列表文本(Item)	所有树(父树和子树)节点的显示文本及 ID,当前选中项会关 联树节点的 ID 编号	/			
水平位置	元件的水平起始位置	不要超出水平分辨率			



垂直位置	元件的垂直起始位置	不要超出垂直分辨率
宽度	元件的宽度	/
高度	元件的高度	/

示例:

1. 在树形图属性-基本属性栏选择寄存器类型和编号(在此例中将"绑定的寄存器类型"设为DT,"绑定的寄存器编号"设为1000,即绑定的寄存器地址为**TABLE(1000)**)。



2. 点击树形图属性框下的"列表文本",设计树节点结构并编写其文本内容。每一树节点都绑定一个唯一的 ID,树节点的 ID 按其编写顺序编号。编辑完成后,若点击"自动编号",则表项 ID 将从 上往下依次进行编号。



树形图设计	×
✓ Item1	选中顶编号: 1000 自动编号
└ \请在此处双击键入] ▲ Iten2 └ 「志在此かっ20+金)]	插入
[请在此处双击键入]	刪除
	确定
	取消

"插入"操作是指在选中项的上方添加与其同级别的一个新项。

树形图设计	×
 向上 小向上6行 小商主10行 小商主此处双击键入] 「请在此处双击键入] 「请在此处双击键入] 「请在此处双击键入] 	送中项编号: 1003 自动编号 插入 删除
	确定 取消



树形图设计	×
▲ 向上 ▲ - 向上5行 □ [清在此处双击键入] ▲ - 向上10行	选中项编号: 1004
□ [清在此处双击键入] □ [清在此处双击键入]	自动编号
▲ □□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□	插入
- 小晃 - 「清在此处双击键入」 「清在此处双击键入」	冊『除
	确定
	取消

3. 编辑完成后,点击"确定"。

形图设计	>
▲ 向上 → 向上5行 ↓ 「请在批外双击键入」	选中项编号: 1006
● 向上10行 □ 「请在此处双击能入」 「唐在此外羽击能入」	自动加高号
不向 〒677向	插入
□ 「请在此处汉击谑入」 「向下10行 □ 「请在此处汉击谑入」	
▲ ○ 「清在此处双击键入」 「清在此处双击键入」	
	确定

4. Hmi 窗口树形图效果图如下:



5. 连接控制器,将 Hmi 文件下载到控制器,可看到 xplc screen 窗口显示效果图,如左下图所示,只显示第一层的树节点;点击"▶"将显示该父树对应的子树项,如右下图所示。



6. 若使用文本库,则其内容将按照层次结构依次填充至树形图中,从顶层的第一个标题开始,逐级向下展开至第二个标题,乃至更深层次的子标题。

文本库																×
语言选择	0	- ○以语	言方式显	示	● 以状态;	方式显示	0-15	•		导入		Ę	寻出		字体管理器	
字体设置																
	语言 0	语言	1	语言 2	语	言 3	语言	4	语言:	5	语言 6		语言 7			
字体	默认	默认		默认	覞	认	默认		默认		默认		默认			
代码页	0	0		0	0		0		0		0		0			
标签设置															搜索	
编号	标签名	引用次数	状态数	语言数	格式文本	状态 0		状态 1		状态 2		状态:	3	状态	4	扐
1	树形图	1	7	1	点击设置	1		2		3		4		5		6
2	双击添加															
4																+
	余 清除												确	ŧ	取消	



- 7. 选择的状态显示颜色、行间距、弹出方式等可通过"属性"窗口进行设置。
- 8. 树形图调用 Basic 文件全局 SUB 函数的具体使用请参照第八章【<u>文件浏览器使用</u>】。

第五章 Hmi 调用 Basic 函数

本章主要介绍 Hmi 调用 Basic 函数的方法以及多种不同函数调用的介绍。

函数调用基本通用方法:

1. 在 Basic 文件中定义全局 (GLOBAL) SUB 子函数, 定义语法:

函数名和括号必须为英文字符。如: GLOBAL SUB redraw() END SUB

2. 在已定义好的 SUB 子函数中,编写该函数体的程序。

3. 打开/新建 Hmi 文件,添加对应元件,打开对应属性窗口,"动作"选择"调用函数",在动作函数名处选择需对应调用的函数。

4. 若是 Hmi 设置或自定义元件, 在属性窗口中"xx 函数"处直接调用即可。

5.1. Hmi 系统设置调用函数

菜单栏"Hmi系统设置"打开如下窗口,初始化函数和周期函数的调用可根据需求选择是否设置。

属性		×
▲ 基本属性		-
LCD编号	0	
背光时间	0	
屏保时间	0	
起始基本窗口	10:Start	
起始置顶窗口	无	
初始化函数		
周期函数		
压缩图片	不压缩	
图片质量	标准	
文本自适应	False	
▲ 位置和尺寸		
水平分辨率	800	
垂直分辨率	480	-
抽参数 帮助 屋性		

初始化函数: 上电后只调用一次的函数,一般将初始化的相关参数定义等内容写入定义好的初始化全局 SUB 子函数,并在 Hmi 设置中调用。

周期函数:上电后周期不断扫描的函数。

注意:初始化函数和周期函数选择 Basic 里编写好的 GLOBAL 全局定义的 SUB 子函数。



5.2. 自定义元件调用函数

自定义元件 CUSTOM 里可添加绘图函数和刷新函数是由 Basic 编写的全局 SUB 子函数。

绘图函数: 需要绘图时自动被调用,此函数里面通过调用 DRAW/DRAWEX 相关函数来自己绘图,绘 图函数的零点是自定义元件的左上角。

刷新函数:周期调用来判断是否要重新绘图(系统会自动周期性调用刷新函数),刷新绘图区域,通过调用 SET REDRAW 指令来指明哪部分区域要刷新。

属性		₹ Ф	×	▲ 位置和尺寸		
▲ 基本属性				水平位置	658	
元件编号	10			垂直位置	364	
元件名称	Custom10			宽度	200	
显示层次	底层			高度	150	
有效显示	显示					•
采用有效控制	False					
数据索引	0					
刷新函数						
绘图函数				轴参数 属性 執	助	

1. 绘图函数参考例程:

在 Hmi 窗口创建一个自定义元件 CUSTOM, 打开"属性"窗口,设置元件区域、绘图函数、刷新函数。



Basic 程序:

GLOBAL SUB reflash() '刷新	函数
SET_REDRAW	
END SUB	
GLOBAL SUB draw() '绘	图函数
SET_COLOR(RGB(255,0,255))	'设置颜色
DRAWRECT(0,0,300,300)	'自定义元件内绘制边框
DRAWTEXT(10,10, "运动控制器")	'自定义元件内显示字符串
DRAWARC(100,100,50, 0, PI*2)	'画一个整圆
END SUB	





2. 刷新函数参考例程:

虚拟键 20、21、22 分别绑定物理键 10、11、12。

态按键转	與表			×
			•	预设
物理键	虚拟键	虚拟键描述		海政会
10	20			
11	21			令人
12	22			
				「台田」
				确定
				取消

Basic 函数:

global sub runy() if num=20 then print 1	'绘图函数 '物理按键10按下 '命令行打印 1
elseif num=21 then table(10)=100	,物理按键11按下 'table10赋值为100
elseif num=22 then functionl() endif end sub	,物理按键12按下 ,调用自定义函数
global sub slt()	'刷新函数
if VKEY_SCAN<>0 then num=VKEY_SCAN endif SET_REDRAV end sub	'扫描虚拟按键,返回值给变量num

运行效果:

物理键 10 按下时,命令行打印 1。 物理键 11 按下时,TABLE(10)赋值为 100。 物理键 12 按下时,调用函数 function1,函数功能可以自定义。

5.3. 元件调用函数

Hmi 组态元件动作调用 Basic 自定义全局 SUB 子函数,很多元件都具有调用函数的功能,例如功能键 BUTTON,在"动作"选项中选择"调用函数","动作函数名"选择全局 SUB 子函数,在按下或松开功能键后 就能调用 SUB 子函数执行。

属性		×
▲ 基本属性		*
元件编号	1	
元件名称	Button1	
显示层次	底层	
有效显示	显示	
采用有效控制	False	
安全时间ms	0	
绑定虚拟按键	No Key	
绑定物理按键	0	
⊿ 外观		
图片来源	背景图片库	
背景图片库	0\按钮\9	
绘制边框	False	
是否图片化	False	
⊿ 标签		
文本库		
格式文本(0)	运动	
格式文本(1)	运动	
⊿ 动作		
动作	调用函数	
松开时动作	时动作 False	
动作函数名	动作函数名 onrun()	
▲ 位置和尺寸		
水平位置	70	
垂直位置	196	
宽度	109	
高度	48	-
轴参数 属性 帮助		

第六章 相关 Basic 指令

以下是 Hmi 增加的 Basic 指令。

6.1. 基础指令

6.1.1.RUN -- 启动文件任务

类型	任务指令	
描述	新建一个任务来执行控制器上的一个文件。	
	重复启动同一任务会报错。	
	多任务操作指令有:	
	END: 当前任务正常结束	
	STOP: 停止指定文件	
	STOPTASK: 停止指定任务	
	HALT:停止所有任务	
	RUN : 启动文件执行	
	RUNTASK: 启动任务在一个 SUB 或标记上执行	
语法	RUN "filename"[, tasknum]	
	filename: 程序文件名, BAS 文件可不加扩展名	
	tasknum: 任务号,缺省查找第一个有效的	
	RUN "file.Hmi", TASKID, [LCDNUM]	
	filename:程序文件名,BAS文件可不加扩展名	
	TASKID: 任务号	
适用控制器	通用	
例子	RUN "aaa", 1 /启动任务 1 运行 aaa.bas 文件	

6.1.2.SCAN_EVENT -- 数据状态变化扫描

类型	输入输出函数			
描述	数据变化扫描。			
		返回值	描述	
		1	数据0变为非0	
		-1	数据非0变为0	
		0	BOOL 数据没有变化	
语法	event = SCAN_EVENT (condition)			
	condition: 数据条件表达式			



适用控制器	通用
例子	WHILE 1
	IF SCAN_EVENT(IN(0))>0 THEN 'IN0 上升沿触发
	PRINT "IN0 ON"
	ELSEIF SCAN_EVENT(IN(0))<0 THEN 'IN0下降沿触发
	PRINT "IN0 Off"
	ENDIF
	WEND

6.1.3.SET_XPLCTERM -- 显示屏启动状态设置

类型	显示操作指令
描述	读取显示屏状态。
语法	SET_XPLCTERM=value
	value: 0-缺省, 不启动
	1-自动启动全屏的 XPLCTERM HDMI 版本
适用控制器	仅支持5系列控制器
例子	SET_XPLCTERM=1 '自动启动全屏 XPLCTERM

6.1.4.SYSTIME -- 系统时间

类型	系统时间		
描述	获取系统当前时间		
语法	string= SYSTIME(style) style: 时间格式, "%Y/%m/%d %H:%M:%S" => 2021/08/14/ 11:03		
	返回时间子付串		
	时间格式:		
	%a 星期几的简写		
	%A 星期几的全称		
	%b 月份的简写		
	%B 月份的全称		
	%c标准的日期的时间串		
	%C 年份的前两位数字		
	%d 十进制表示的每月的第几天		
	%D 月/天/年		
	%e 在两字符域中,十进制表示的每月的第几天		
	%F 年-月-日		
	%g 年份的后两位数字,使用基于周的年		
	%G 年份,使用基于周的年		
	%h 简写的月份名		



	%H 24 小时制的小时	
	%I 12 小时制的小时	
	%j 十进制表示的每年的第几天	
	%m 十进制表示的月份	
	%M 十进制表示的分钟数	
	%n 新行符	
	%p 本地的 AM 或 PM 的等价显示	
	%r 12 小时的时间	
	%R 显示 24 小时制的小时和分钟: hh:mm	
	%S 十进制的秒数	
	%t 水平制表符	
	%T 显示时分秒: hh:mm:ss (24 小时制)	
	%u 每周的第几天, 星期一为第一天 (值从1到7, 星期一为1)	
	%V 每年的第几周,使用基于周的年	
	%w 十进制表示的星期几(值从0到6,星期天为0)	
	%U 每年的第几周,把星期日做为第一天(值从0到53)	
	%W 每年的第几周,把星期一做为第一天(值从0到53)	
	%x 标准的日期串	
	%X 标准的时间串	
	%y 不带世纪的十进制年份(值从0到99)	
	%Y 带世纪部分的十制年份	
	%z, %Z 时区名称, 如果不能得到时区名称则返回空字符。	
	%% 百分号	
适用控制器	支持4系列和5系列以上控制器	
例子	例一: ?SYSTIME("%Y/%m/%d %H:%M:%S")	
	2021/11/12 11:30	
	例二: ?"日期: ", SYSTIME("%x")	
	?"每周第几天: ", SYSTIME("%u")	
	打印结果如下:	
	日期: 07/13/22	
	每周第几天: 3	

6.2. 语法指令

6.2.1.DMSET -- 数组区域赋值

类型	语法指令
描述	数组区域赋值。
语法	DMSET arrayname(pos, size, data)
	pos: 起始索引



	size: 长度
	data: 设置的数值
适用控制器	通用
例子	DMSET table(0,10,2) '数组区域赋值
	FOR i=0 TO 9
	PRINT "table",i, table(i) '打印数组
	NEXT
	DMSET table(0,10,3) 岁组区域赋值
	FOR i=0 TO 9
	PRINT "table",i, table(i)
	NEXT
相关指令	ZINDEX_LABEL,ZINDEX_ARRAY, ZINDEX_CALL

6.2.2.ZINDEX_LABEL -- 建立索引指针

类型	语法指令		
描述	可以为 SUB 函数或数组建立索引指针,后面通过索引指针来调用。		
语法	Pointer = ZINDEX_LABEL(subname)		
	subname: 数组或 SUB 名称		
适用控制器	通用		
例子	DIM arr1(100) '定义数组		
	Arr1(0,1) '对数组赋值		
	Pointer = ZINDEX_LABEL(arr1)		
	PRINT ZINDEX_ARRAY(Pointer)(0) '访问数组,打印数组第一位		
相关指令	ZINDEX_ARRAY, ZINDEX_CALL		

6.2.3.ZINDEX_ARRAY -- 访问数组

类型	语法指令
描述	通过索引指针来访问数组。
语法	var = ZINDEX_ARRAY (pointer)(index)
	subname: 数组或 SUB 名称
适用控制器	通用
例子	DIM arr1(100) '定义数组
	Arr1(0,1)'对数组赋值
	Pointer = ZINDEX_LABEL(arr1)
	PRINT ZINDEX_ARRAY(Pointer)(0) '访问数组,打印数组第一位
相关指令	ZINDEX_CALL, ZINDEX_LABEL

6.2.4.ZINDEX_CALL -- SUB 函数调用

类型	语法指令		
描述	通过索引指针来调用 SUB 函数。		
语法	ZINDEX_CALL(zidnex) (subpara,)		
	zidnex: 索引指针		
	subpara: sub 的参数调用		
适用控制器	通用		
例子	Pointer = ZINDEX_LABEL(sub1)'建立索引指针		
	ZINDEX_CALL(Pointer) (2) '调用函数		
	End		
	SUB sub1(a)		
	PRINT a		
	END SUB		
相关指令	ZINDEX_ARRAY, ZINDEX_LABEL		

6.2.5.ZINDEX_VAR -- 指针变量操作

类型	语法指令	
描述	通过索引指针来操作变量。	
语法	ZINDEX_VAR(zindex)	
	zindex:通过 ZINDEX_LABEL 生成的 z 指针	
适用控制器	通用	
例子	zindex= ZINDEX_LABEL(varname)	
	ZINDEX_VAR(zindex)=value	
	VAR2 = ZINDEX_VAR(zindex)	
相关指令	ZINDEX_ARRAY, ZINDEX_LABEL	

6.2.6.ZINDEX_MARK -- 指针标号设置

类型	语法指令	
描述	给 label 设置标号,从而可以把 label 的数组索引记录下来。	
语法	ZINDEX_MARK(labelname) = mark	
	varmark = ZINDEX_MARK(labelname)	
	labelname: 如果不同文件都有定义,注意不同文件里面调用的是不同的结果	
	mark: 标号	
适用控制器	支持 ZV 功能或者 5 系列以上的控制器	
例子	dim var1	
	dim arr1(10),MarkArr(10)	
	dim ArrIndex,VarIndex	

Zmotion[®]

	arr1(0,1,2,3,4,5,6,7,8,9,10)
	VarIndex = zindex_label(var1)
	zindex_mark("VarIndex") = 0 '设置变量指针、数组指针标号 zindex_mark("ArrIndex") = 1
	markarr(zindex_mark("VarIndex")) = VarIndex
	zindex_var(MarkArr(zindex_mark("VarIndex"))) = 777
	?zindex_array(MarkArr(zindex_mark("ArrIndex")))(9) ?zindex_array(MarkArr(1))(9)
	end ZINDEX ARRAY ZINDEX LAREL
相大指令	<u>LINDEA_AKKAY, LINDEA_LABEL</u>

6.2.7.ZINDEX_STRUCT -- 获取/访问结构变量

类型	语法指令	
描述	获取结构变量的指针后,通过指针来访问结构变量。	
语法	index = ZINDEX_LABEL(structvarname)	
	ZINDEX_STRUCT(structname,index).item = var	
	var = ZINDEX_STRUCT(structname,index).item	
适用控制器	支持 ZV 功能或者 5 系列以上的控制器	
例子	/	
相关指令	ZINDEX_ARRAY, ZINDEX_LABEL	

6.2.8.ZINDEX_ZVOBJ -- 获取对象索引数据

类型	语法指令
描述	获取对象索引的数据。
语法	ZINDEX_ZVOBJ(index)
适用控制器	支持 ZV 功能或者 5 系列以上的控制器
例子	ZVOBJECT img
	ZV_READIMAGE(img,"logo.png",0) '读取一张图像



	index = ZINDEX_LABEL(img)	'获取图像对象的索引	
	ZV_LATCH(ZINDEX_ZVOBJ(index), 0)	'获取图像索引的数据,	并将图像显示
	到锁存通道0上		
相关指令	ZINDEX_ARRAY, ZINDEX_LABEL		

6.3.显示指令

6.3.1.LCD_FEATURE -- 读取显示器特征

类型	显示指令	
描述	显示器的特征读取。必须控制器和显示器同时支持此功能才可以。	
语法	var = LCD_FEATURE(lcdnum, featurenum)	
	lcdnum: 0-显示器编号	
	featurenum: 特征号	
	0- type 0-控制器内置的显示器; 1-电脑 term; -1-未连接; -2-不存在 300-	
	ZHD300x 400-ZHD400X, 701-7 寸触摸屏.	
	1-width 显示器物理宽度, 0-可变的宽度范围	
	2- height 显示器物理高度,0-可变的高度范围	
适用控制器	支持 RTHmi	
例子	PRINT LCD_FEATURE(0,0)'打印显示器类型	
	PRINT LCD_FEATURE(0,1)'打印显示器物理宽度	
	PRINT LCD_FEATURE(0,2)'打印显示器物理高度	

6.3.2.LCD_LEDSTATE -- 控制 LED 灯状态

类型	系统屏参数
描述	设置示教盒上 LED 灯的状态。
	必须控制器和显示器同时支持此功能。
语法	LCD_LEDSTATE(lcdnum) = VAR
	lcdnum: 显示器编号
	按 Bit 位依次设置,缺省=1,亮运行灯。
适用控制器	支持 RTHmi, 固件版本 230801 以上
适用示教盒	ZHD500X
例子	LCD_LEDSTATE(0)=3 '亮运行灯和报警灯

6.3.3.LCD_WDOGTIME -- 显示器掉线处理时间

类型 系统屏参数	
-----------------	--

Zmotion[®]

描述	设置屏掉线处理时间, ms 单位, 当屏操作这个时间没有通讯时, 急停开关(物理按	
	键编号=5)自动按下,等于0时此功能不启用。	
	需要控制器 5 系列 20180404 以上固件版本支持, 4 系列标准固件 20170721 以上	
	版本支持。	
语法	LCD_WDOGTIME(lcdnum) = time	
	lcdnum: 0-显示器编号	
	time: ms 时间	
适用控制器	支持 RTHmi	
例子	$LCD_WDOGTIME(0) = 100$	

6.3.4.DRAWNUM -- 自定义元件内显示数值

类型	显示指令	
描述	显示一个数值。	
	此指令只能在自定义元件的绘图函数内使用,请查看自定义元件调用函数参考例	
	程。	
语法	DRAWNUM (x,y,value,n,dot)	
	x,y: 显示区域左上角的坐标位置	
	value: 缺省显示值	
	n: 总长度位数,包括小数点和符号位。当N设置负数时,表示右对齐	
	dot: 小数点后取几位	
适用控制器	支持 RTHmi	
例子	DRAWRECT(0,0,200,100) '自定义元件内绘制边框	
	DRAWNUM(10,10,0,4,2) '在自定义元件显示区域左上角(10,10)显示 0.00	
	0.00	
相关指令	DRAWNUM2	

6.3.5.DRAWNUM2 -- 指定位置显示数值

类型	显示指令	
描述	显示一个数值,指定一个方框内显示。	
	此指令只能在自定义元件的绘图函数内使用,请查看自定义元件调用函数参考例	
	程。	
语法	DRAWNUM2 (x1,y1, x2,y2, align,value,n,dot[,drawrect])	
	x1,y1: 显示区域左上角的坐标位置	
	x2, y2: 显示区域右下角的坐标位置	

Zmotion[®]

	align: 对齐选项
	0-居中
	>0, 左边对齐, 值表示距离左边的距离
	<0,右边对齐,绝对值表示距离右边的距离
	value: 缺省显示值
	n: 总长度位数,包括小数点和符号位。当n设置负数时,表示右对齐
	dot: 小数点后取几位
	drawrect: 可选, 0-不画边框(缺省), 1-画边框
适用控制器	支持 RTHmi
例子	DRAWRECT(0,0,200,100) '自定义元件内绘制边框
	DRAWNUM2(10,10,200,100,0,123,6,2) '显示在两个坐标位置的中间,保留两位小
	数,显示结果: 123.00
	123.00
相关指令	DRAWNUM

6.3.6.DRAWTEXT -- 绘制显示字符串

类型	显示指令
描述	显示一个字符串,支持中文,STRING 可以为字符串表达式。
	此指令只能在自定义元件的绘图函数内使用,请查看 <u>自定义元件调用函数</u> 参考例
	程。
语法	DRAWTEXT (x,y,STRING)
	X,Y: 显示区域左上角的坐标位置
	STRING: 显示的字符串
适用控制器	支持 RTHmi
例子	DRAWRECT(0,0,200,120) '自定义元件内绘制边框
	DRAWTEXT(10,10, "运动控制器") '在自定义元件上显示文本"运动控制器"
	运动控制器
相关指令	DRAWTEXT2

6.3.7.DRAWTEXT2 -- 绘制显示字符串

类型	显示指令
描述	显示一个字符串,指定一个方框内显示。
	此指令只能在自定义元件的绘图函数内使用。请查看自定义元件调用函数参考例
	程。
语法	DRAWTEXT (x1,y1, x2,y2, align, STRING[,drawrect])
	x1,y1: 显示区域左上角的坐标位置
	x2, y2: 显示区域右下角的坐标位置
	align: 对齐选项
	0-居中
	>0, 左边对齐, 值表示距离左边的距离
	<0,右边对齐,绝对值表示距离右边的距离
	STRING: 显示的字符串
	drawrect: 0-不画边框(缺省), 1-画边框
适用控制器	支持 RTHmi
例子	DRAWTEXT2(10,10,100,0,0,"运动控制器",1) '在自定义元件上显示文本,给指
	定的区域绘制边框
	运动控制器
相关指令	DRAWTEXT

6.3.8.DRAWLIBTEXT -- 显示文本库字符串

类型	显示指令
描述	显示文本库的字符串内容。
	此指令只能在自定义元件的绘图函数内使用,请查看自定义元件调用函数参考例
	程。
语法	DRAWLIBTEXT(x,y, state, "textname")
	x,y: 显示区域左上角的坐标位置
	state: 文本库的状态
	textname: 文本库的名称
适用控制器	支持 RTHmi
例子	DRAWLIBTEXT (10,10, 0, "text1")
相关指令	DRAWLIBTEXT2, DRAWTEXT, DRAWTEXT2

6.3.9.DRAWLIBTEXT2 -- 显示文本库字符串

类型	显示指令
描述	格式显示文本库的字符串内容。
	此指令只能在自定义元件的绘图函数内使用,请查看 <u>自定义元件调用函数</u> 参考例
	程。
语法	DRAWLIBTEXT2(x1,y1, x2,y2, align, state, "textname"[,drawrect])
	x1,y1: 显示区域左上角的坐标位置
	x2, y2: 显示区域右下角的坐标位置
	align: 对齐选项
	0-居中
	>0, 左边对齐, 值表示距离左边的距离
	<0,右边对齐,绝对值表示距离右边的距离
	state: 文本库的状态
	textname: 文本库的名称
	drawrect: 1-画边框, 0-不画边框.
适用控制器	支持 RTHmi
例子	DRAWLIBTEXT2(10,10,100,50,0,0,"label")
相关指令	DRAWLIBTEXT, DRAWTEXT, DRAWTEXT2

6.3.10. DRAWREVERSE -- 绘制方块

类型	显示指令
描述	画一个填充(黑色)的方框。
	此指令只能在自定义元件的绘图函数内使用,请查看自定义元件调用函数参考例
	程。
语法	DRAWREVERSE(x1,y1,x2,y2)
	x1,y1: 显示区域左上角的坐标位置
	x2,y2: 显示区域右下角的坐标位置
适用控制器	支持 RTHmi
例子	DRAWRECT(0,0,200,100) '自定义元件内绘制边框
	DRAWREVERSE(10,10,50,50) '黑色方框填充

6.3.11. DRAWRECT -- 绘制矩形

类型	显示指令
描述	画一个方框。
	此指令只能在自定义元件的绘图函数内使用,请查看 <u>自定义元件调用函数</u> 参考例
	程。
语法	DRAWRECT(x1,y1,x2,y2)
	x1,y1: 显示区域左上角的坐标位置
	x2,y2: 显示区域右下角的坐标位置
适用控制器	支持 RTHmi
例子	DRAWRECT(0,0,200,100) '自定义元件内绘制边框
相关指令	DRAWEX_RECT, DRAWEX_ELLIPSE

6.3.12. DRAWLINE -- 绘制线段

类型	显示指令
描述	画一条直线。
	此指令只能在自定义元件的绘图函数内使用,请查看自定义元件调用函数参考例
	程。
语法	DRAWLINE(x1,y1,x2,y2)
	x1,y1: 直线起始点的坐标位置
	x2,y2: 直线结束点的坐标位置
适用控制器	支持 RTHmi
例子	DRAWRECT(0,0,200,100) '自定义元件内绘制边框
	DRAWLINE(10,10,50,50) '画直线
相关指令	DRAWEX_LINE, DRAWARC
6.3.13. DRAWCLEAR -- 清除指定区域内容

类型	显示指令
描述	清除指定读取区域。
	此指令只能在自定义元件的绘图函数内使用,请查看 <u>自定义元件调用函数</u> 参考例
	程。
语法	DRAWCLEAR ([x1,y1,x2,y2])
	x1,y1: 清除区域左上角的坐标位置
	x2,y2: 清除区域右下角的坐标位置
	无参数时全部清除
适用控制器	支持 RTHmi
例子	DRAWCLEAR (10,10,50,50)

6.3.14. DRAWPIC -- 插入图片文件

类型	显示指令
描述	绘制图片,图片文件要先加入工程,在文件视图添加图片,注意图片比较占空间,
	不需要的图片不要加入工程。
	此指令只能在自定义元件的绘图函数内使用,请查看自定义元件调用函数参考例
	程。
语法	DRAWPIC (Picname, X1,Y1[,X2,Y2])
	picname: 图片文件名
	x1,y1: 显示区域左上角的坐标位置
	x2,y2: 显示区域右下角的坐标位置
适用控制器	支持 RTHmi
例子	DRAWRECT(0,0,200,100) '自定义元件内绘制边框
	DRAWPIC ("1.bmp",10,10,100,80) /加入图片
相关指令	DRAWLIBPIC

6.3.15. DRAWARC -- 绘制圆弧

类型	显示指令
描述	画圆弧。
	此指令只能在自定义元件的绘图函数内使用,请查看 <u>自定义元件调用函数</u> 参考例
	程。



	DRAWARC(centerx,centery,radius,0*PI/180+2*PI,90*PI/180)
相关指令	DRAWLINE, DRAWEX_ARC

6.3.16. DRAWLIBPIC -- 插入图片库图片

类型	显示指令
描述	绘制图片,图片文件要加入图片库,注意图片比较占空间,不需要的图片不要加
	λ .
	此指令只能在自定义元件的绘图函数内使用,请查看 <u>自定义元件调用函数</u> 参考例
	程。
语法	DRAWLIBPIC (piclibnamePath, state, x1,y1[,x2,y2])
	piclibnamePath: 要加载的图片库路径
	1、系统图片库下: System\二级目录名称\加载图片的名称
	2、用户图片库: User\二级目录名称\图片库名称
	state: 图片的状态编号
	x1,y1: 显示区域左上角的坐标位置
	x2,y2: 显示区域右下角的坐标位置
适用控制器	支持 RTHmi
例子	DRAWLIBPIC ("User\default\LOGO",0,100,100,300,300) '在 (100,100) 到
	(300,300)区域显示图片库中名称为 LOGO 的 0 状态
相关指令	DRAWPIC

6.3.17. DRAWBEZIER -- 绘制贝塞尔曲线

类型	显示指令
描述	画贝塞尔曲线。
	此指令只能在自定义元件的绘图函数内使用,请查看自定义元件调用函数参考例程。
语法	DRAWBEZIER(x1,y1,x2,y2,x3,y3,x4,y4)
	x1,y1: 第 1 个控制点
	x2,y2: 第 2 个控制点
	x3,y3: 第 3 个控制点
	x4,y4: 第 4 个控制点
适用控制器	支持 RTHmi



6.3.18. DRAWBSPLINE -- 绘制 B 样条曲线

类型	显示指令
描述	画 B 样条曲线。
	此指令只能在自定义元件的绘图函数内使用,请查看 <u>自定义元件调用函数</u> 参考例
	程。
语法	DRAWBSPLINE (x1,y1,x2,y2,x3,y3,x4,y4)
	x1,y1: 第 1 个控制点
	x2,y2: 第 2 个控制点
	x3,y3: 第 3 个控制点
	x4,y4: 第 4 个控制点
适用控制器	支持 RTHmi
例子	DRAWRECT(0,0,200,200) '自定义元件内绘制边框
	DRAWBSPLINE (0,0,200,0,150,200,0,200) '画 B 样条曲线

6.3.19. DRAWDTLIST -- 绘制图形

类型	显示指令			
描述	绘图指令,用于加	l快 TABLE 数排	居的绘制。	
	此指令只能在自定	主义元件的绘图	函数内使用,请查看 <u>自定义元件调用函数</u>	参考例
	程。			
语法	DRAWDTLIST (dtstart, imax, ispace, fxstart, fystart, fxlevel, fylevel, imode			mode,
	[drawtype], [TYPE	1, TYPE2, T	YPE3, TYPE4])	
	dtstart: table	起始位置,指向	句第一行的行类型	
	行类型:由用	户自行定义,	即可设置0为点或直线,1也可为点或直约	戋。
	imax: 行数 (设置的 table 的	1组数)	
	行数不一定和	设置的 table 组	1数相等,可根据实际情况填写。	
	ispace: 行间降	鬲(即每个 table	e 起始位置之差,若 imode 参数为1时,则	行间隔
	一般不能<2;若 ir	node 偏移了 N	个数据,则最小行间隔需相应增加N)	
	fxstart: 左上角	IX 坐标的偏移	3量,相对于 tbale 中设置的 X 坐标进行偏移	多(> 0 —
	向左偏移,<0—向]右偏移,=0不	《偏移》	h / -
	fystart: 左上角	FY 坐标的偏移	步重,相对于 tbale 中设置的 Y 坐标进行偏移	ğ(> 0 —
	回上偏移,<0—直	」 ト 偏 移 , =0 个		
	txlevel : tabl	e 屮 X 坐标的数	以但比例	
	tylevel: table	Ψ Y 坐标的数	1旦 ℃ 17月	
	imode: 仔储和	合 八 世 法]	
	11. 打	田处	5按书 此时 X 的位置上气光型位均 2 人	
		JSB	Ŋ恰式,	
		₩旧	abla 由的位置占行米刑伯投了 \r 人粉坦	
	1-9 I	4时11、衣A住じ V的粉捉竖丝	aur中的位直刁小天空俩份JN个数据,	
		▲ 町 奴1佰 糸 (矢) F	1	
	六丘祖 ノ drawtwrage 見	山 <u>风</u> 示方式		
	值	小 ノーム	描述	
	<u>国</u> 0(缺省)		直接绘制一个占	
	U、叭泪) N*256⊥ 子	N*256+ 1	<u> </u>	
	1 2.50 1	N*256+ 2	十///天/C 四/// 首占空心周弧	
	N=半谷可	N*256+ 3	→灬⊥・□凶/M 单占实心正方形	
	线宽	N*256+4	单占虚正方形	
	(N 的限	N*256+ 5	单点十字架	
	制范围:	N*256+ 6	单点叉	
	$1 \le N \le 16$)	N*256+ 9	单点显示,和上一占之间画一个虑线.	
			N 只能为1	
		N*256+10	前后两点之间画直线,必须都是同一种	
			类型	
		N*256+11	设置圆弧起始点、中间点和结束点,自	
			动计算圆弧并显示(该方式下不能通过	
			N 调节半径大小)	
			/ • • • • • • /	





矩形并非指令绘制,仅作提示)
例 ^一 · 绘制一段直线
TABLE(10,2,100,200)
TABLE(20,2,200,200) DRAWDTLIST(10.2,10.0.0.1.1.1.256+10.2)
例三:
TABLE(0,2,100,100)
TABLE(10,2,100,200) TABLE(20,2,200,200)
TABLE(30,2,200,100)
DRAWDTLIST(0,4,10,0,0,1,1,1,16*256+4,2) '绘制空心矩形
DRAWD1LIS1(0,4,10,40,0,1,1,1,16*256+3,2)







相关指令	DRAWLINE

6.3.20. DRAWEX_LINE -- 绘制线段(带样式)

类型	绘图指令
描述	画一条直线。与 DRAWLINE 的区别在于 DRAWEX_LINE 可以设置线条宽度和线 条类型,而 DRAWLINE 不支持。 DRAWEX 绘图指令也将对绘制区域进行限制。原则上是不合允许画出自定义按供
	之外的,DRAW 相关指令则无限制。
	此指令只能在自定义元件的绘图函数内使用。请查看 <u>自定义元件调用函数</u> 参考例 程。
语法	DRAWEX_LINE(x1,y1,x2,y2)
	x1,y1: 直线起始点的坐标位置
	x2,y2: 直线结束点的坐标位置
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	SETEX_LINE(3,0) '设置线条宽度
	SET_COLOR(RGB(0,0,255)) '设置线条颜色
	DRAWEX_LINE(100,100,200,50)
相关指令	DRAWLINE, <u>SETEX_LINE</u> , <u>SET_COLOR</u>

6.3.21. DRAWEX_ARC -- 绘制圆弧(带样式)

类型	绘图指令
描述	画圆弧/椭圆弧。与 DRAWARC 的区别在于 DRAWEX_ARC 可以设置线条宽度和
	线条类型,而 DRAWARC 不支持。
	DRAWEX 绘图指令也将对绘制区域进行限制,原则上是不会允许画出自定义控件





	SETEX_LINE(2,0) '设置线条宽度
	DRAWEX_ARC(200,200,50,50,90*PI/180,-135*PI/180) '逆时针画圆弧
	-135° 90°
相关指令	DRAWARC, DRAWEX_LINE, SETEX_LINE, SET_COLOR

6.3.22. DRAWEX_BEZIER -- 绘制贝塞尔曲线(带样式)

类型	绘图指令
描述	画贝塞尔曲线。与 DRAWBEZIER 的区别在于 DRAWEX_BEZIER 可以设置线条
	龙艮和线余尖空,间 DKAWBEZIEK 小又行。
	DRAWEX 绘图指令也将对绘制区域进行限制, 原则上是个会允许画出自定义控件
	之外的,DRAW 相关指令则尤限制。
	此指令只能在自定义元件的绘图函数内使用。请查看自定义元件调用函数参考例
	程。
语法	DRAWEX_BEZIER(x1, y1, x2, y2, x3, y3, x4, y4)
	x1,y1: 第 1 个控制点
	x2,y2: 第 2 个控制点
	x3,y3: 第 3 个控制点
	x4,y4: 第 4 个控制点
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	SETEX_LINE(1,1) '设置线条类型为虚线
	DRAWEX_BEZIER(200,0,350,100,100,300,200,400) '绘制贝塞尔曲线
相关指令	DRAWBEZIER, DRAWBSPLINE, SETEX_LINE, SET_COLOR

6.3.23. DRAWEX_BSPLINE -- 绘制 B 样条曲线(带样式)

类型 绘图指令

描述	画 B 样条曲线。与 DRAWBSPLINE 的区别在于 DRAWEX_BSPLINE 可以设置线 条宽度和线条类型,而 DRAWBSPLINE 不支持。 DRAWEX 绘图指令也将对绘制区域进行限制,原则上是不会允许画出自定义控件 之外的,DRAW 相关指令则无限制。 此指令只能在自定义元件的绘图函数内使用。请查看 <u>自定义元件调用函数</u> 参考例
	程。
语法	DRAWEX_BSPLINE(x1, y1, x2, y2, x3, y3, x4, y4)
	x1,y1: 第 1 个控制点
	x2,y2: 第 2 个控制点
	x3,y3: 第 3 个控制点
	x4,y4: 第 4 个控制点
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	SETEX_LINE(2,0) '设置线条宽度
	SET_COLOR(RGB(0,0,255)) '设置线条颜色
	DRAWEX_BSPLINE(200,0,50,100,100,300,200,400) '绘制 B 样条曲线
相关指令	DRAWBSPLINE, DRAWBEZIER, DRAWEX_BEZIER, SETEX_LINE,
	<u>SET_COLOR</u>

6.3.24. DRAWEX_RECT -- 绘制圆角矩形(带样式)

类型	绘图指令
描述	在指定区域画一个圆角矩形。与 DRAWRECT 的区别在于 DRAWEX_RECT 支持 设置线条宽度、线条类型和填充绘图,而 DRAWRECT 不支持。 DRAWEX 绘图指令也将对绘制区域进行限制,原则上是不会允许画出自定义控件 之外的,DRAW 相关指令则无限制。 此指令只能在自定义元件的绘图函数内使用。请查看 <u>自定义元件调用函数</u> 参考例 程。
语法	<pre>DRAWEX_RECT(x1,y1,x2,y2,round [, iffill]) x1,y1: 显示区域左上角的坐标位置 x2,y2: 显示区域右下角的坐标位置 round: 圆角半径 iffill: 是否填充,缺省为0(不填充)</pre>
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	例一:



6.3.25. DRAWEX_ROTRECT -- 绘制旋转矩形(带样式)

类型	绘图指令
描述	在指定区域画一个旋转矩形。 此指令支持设置线条宽度、线条类型和填充绘图,但不允许画出自定义控件之外。 此指令只能在自定义元件的绘图函数内使用。请查看 <u>自定义元件调用函数</u> 参考例 程。
语法	<pre>DRAWEX_ROTRECT(cx,cy,w,h,angle[, iffill]) cx, cy: 旋转矩形中心坐标位置 w, h: 旋转矩形宽高 angle: 矩形旋转角度, 弧度单位, 逆时针方向 iffill: 是否填充, 缺省为0(不填充) 0 - 不填充, 非0- 填充 (注意: 后绘制的可能会遮挡前绘制的图, 因此选择填充时建议先画填充, 后画 边框)</pre>
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	例: SET_COLOR(RGB(0,0,0),RGB(0,255,0)) DRAWEX_ROTRECT(150,150,200,100,45*PI/180, 1) '绘制逆时针旋转 45 弧度的 矩形 DRAWEX_ROTRECT(150,150,200,100, 45*PI/180, 0) '绘制逆时针旋转 45 弧度的 矩形边框

相关指令	DRAWEX_RECT, SETEX_LINE, SET_COLOR

6.3.26. DRAWEX_ELLIPSE -- 绘制椭圆(带样式)

类型	绘图指令
描述	在指定区域画一个椭圆。
	此指令支持设置线条宽度、线条类型和填充绘图,但不允许画出自定义控件之外。
	此指令只能在自定义元件的绘图函数内使用。请查看自定义元件调用函数参考例
语法	DRAWEX_ELLIPSE(centx, centy, radiusx, radiusy [, iffill])
	centx, centy: 圆心的位置
	radiusx: X方向的半径
	radiusx: Y方向的半径
	iffill: 是否填充,缺省为0(不填充)
	0- 不填充,非0- 填充
	(注意:后绘制的可能会遮挡前绘制的图,因此选择填充时建议先画填充,后画
	边框)
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	SETEX_LINE(2,0) '设置线条宽度为 2 的实线
	SET_COLOR(RGB(0,0,0),RGB(0,255,0)) '设置椭圆边框颜色为黑色,填充颜色
	为绿色
	DRAWEX_ELLIPSE(100,150,80,50,1)
	DRAWEX_ELLIPSE(100,150,80,50,0)
相关指令	DRAWEX_RECT, DRAWEX_SECTOR, SETEX_LINE, SET_COLOR

6.3.27. DRAWEX_SECTOR -- 绘制扇形(带样式)

绘图指令
在指定区域画一个扇形。
此指令支持设置线条宽度、线条类型和填充绘图,但不允许画出自定义控件之外。
此指令只能在自定义元件的绘图函数内使用。请查看自定义元件调用函数参考例
程。
DRAWEX_SECTOR(centx, centy, radiusx, radiusy, startangle, endangle [, iffill]) centx, centy: 圆心的位置 radiusx,radiusy: 半径大小 startangle: 起始角度, 弧度单位 endangle: 结束角度 iffill: 是否填充, 缺省为0(不填充) 0- 不填充, 非0- 填充 (注意: 后绘制的内容可能会遮挡前绘制的内容,因此选择填充时建议先画填充, 后画边框)
绘制的角度说明:
$ \begin{array}{c} -90^{\circ} \\ -135^{\circ} \\ 180^{\circ} \\ 135^{\circ} \\ 90^{\circ} \\ 45^{\circ} \\ 90^{\circ} \end{array} $
从起始角度向结束角度绘制扇形:
若起始角度<结束角度:顺时针绘制扇形
若起始角度>结束角度: 逆时针绘制扇形
文持 RIHmi (仅文持 4 系列及以上控制器)
例一: SETEX LINE(2.0) ·设置线条宽度
SET_COLOR(RGB(0,0,0),RGB(250,128,114)) '设置边框颜色和填充颜色
DRAWEX_SECTOR(200,150,100,100,PI,90*PI/180,1) '绘制扇形边框
DRAWEX_SECTOR(200,150,100,100,PI,90*PI/180,0) '逆时针绘制可填充扇形
SETEX LINE(2,0) ·设置线条宽度



	SET_COLOR(RGB(0,0,0),RGB(250,128,114)) '设置边框颜色和填充颜色 DRAWEX_SECTOR(200,150,50,50,PI/2+2*PI,4*PI,1) '绘制扇形边框 DRAWEX_SECTOR(200,150,50,50,PI/2+2*PI,4*PI,0) '顺时针绘制可填充扇形
相关指令	DRAWEX ELLIPSE, SETEX LINE, SET COLOR

6.3.28. DRAWEX_POLYGON -- 绘制多边形(带样式)

类型	绘图指令
描述	在指定区域画一个多边形。
	此指令支持设置线条宽度、线条类型和填充绘图,但不允许画出自定义控件之外。
	此指令只能在自定义元件的绘图函数内使用。请查看自定义元件调用函数参考例
	程。
语法	DRAWEX_POLYGON(points [, x1,y1,,xn,yn] [, iffill])
	points: 多边形点个数,决定后面有多少组 xy 数据(封闭图形起点和终点坐
	标需传两次,则总需传入点的坐标个数=多边形点个数+1)
	xn,yn: 第 n 个点的 x 和 y 的坐标值
	uffill: 是否填允, 缺省为0(个填允)
	0- 个項允,非 $0-$ 項允 (注意 后处制的可能人) 海拔 首处制的图 因此准权持去时建议生运持去 后运
	(注息: 后绘制的可能会巡扫削绘制的图, 因此远律填充可建议无画填充, 后画
	边框》
	特别说明。
	若绘制封闭的多边形, 加绘四边形需要传5个占, 日起占=终占, 反之则是善
	通的多线段(不能进行填充)。
	若绘制的填充多边形不显示,请先检查是否是封闭图形。
	多边形的边数是有限制的,当前版本最多支持32边形。(不同控制器支持参
	数个数不同,多边形的边数由控制器支持参数个数决定)
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	SETEX_LINE(1,2) '设置线条类型为虚线
	SET_COLOR(RGB(0,0,255),RGB(0,255,255)) '设置边框颜色和填充颜色
	DRAWEX_POLYGON(6,100,90,145,125,125,175,75,175,55,125,100,90,1)
	填充的五边形
	DRAWEX_POLYGON(6,100,90,145,125,125,175,75,175,55,125,100,90) '绘制五边
	形边框

相关指令	DRAWRECT, DRAWEX_RECT, SETEX_LINE, SET_COLOR

6.3.29. DRAWEX_POLYGON2 -- 绘制多边形(table 存储)

类型	绘图指令
描述	在指定区域画一个多边形
	此指令支持设置线条宽度、线条类型和填充绘图,但不允许画出自定义控件之外。
	此指令只能在自定义元件的绘图函数内使用。请查看自定义元件调用函数参考例
	程。
语法	 DRAWEX_POLYGON2(points, tableindex [, iffill]) points: 多边形点个数,决定后面有多少组 xy 数据(封闭图形起点和终点坐标需传两次,则总需传入点的坐标个数=多边形点个数+1) tableindex: 存储起始点数据的 table 下标(x1,y1,xn,yn) iffill: 是否填充,缺省为0(不填充) 0-不填充,非0-填充 (注意,后绘制的可能会遮挡前绘制的图,因此选择填充时建议先画填充,后画
	(在意: 后绘前的可能会巡扫前绘前的图, 因此远律填光时建议无画填光, 后画边框)
	特别说明:
	若绘制封闭的多边形,如绘四边形需要传5个点,且起点=终点,反之则是普
	通的多线段(不能进行填充)。
	若绘制的填充多边形不显示,请先检查是否是封闭图形。
	多边形的边数是有限制的,当前版本最多支持 32 边形。
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	SETEX_LINE(2,0) '设置线条宽度为 2,实线类型
	SET_COLOR(RGB(0,0,255),RGB(0,255,0)) '设置线条颜色和填充颜色
	TABLE(0)=100 'x1
	TABLE(1)=90 'y1
	TABLE(2)=55 'x2
	TABLE(3)=125 'y2
	TABLE(4)=75 'x3
	TABLE(5)= 175 'y ₃
	IABLE(6) = 125 X4
	IABLE(/)=1/5 Y4 TADLE(2)=145 $4x5$
	IABLE(0)=143 X3 $TABLE(0)=125 4x5$
	IABLE(9) - 123 y3 TABLE(10) - 100 y6
	IADLE(10)-100 X0

	TABLE(11)=90 'y6 DRAWEX_POLYGON2(6,0,1) '绘制可填充的五边形 DRAWEX_POLYGON2(6,0) '绘制五边形边框
相关指令	DRAWRECT, DRAWEX_RECT, SETEX_LINE, SET_COLOR

6.3.30. SETEX_LINE -- 设置线段属性

类型	显示指令
描述	绘图线宽度设置(该指令与 DRAWEX 类指令配套使用)
	所有的 DRAWEX 指令均支持线段属性
	此指令只能在自定义元件的绘图函数内使用,请查看 <u>自定义元件调用函数</u> 参考例
	程。
语法	SETEX_LINE (width [, type])
	width: 线段宽度(范围: 1-20)
	type: 线段类型(<mark>线宽>1时只能实线</mark>)
	0-实线(缺省)
	1-长虚线 DASH:
	2-虚线 DOT:
	3- DASHDOT:
	4- DASHDOTDOT:
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	SETEX_LINE(1,3) '设置线条宽度为 1,线条类型为 DASHDOT
	SET_COLOR(RGB(0,0,255))
	DRAWEX_LINE(100,100,400,400)
相关指令	<u>SET_COLOR</u>

6.3.31. SET_FONT -- 字体设置

类型	显示指令			
描述	字体设置,	缺省使用内置 16*16 的中文字体,	英文为16*8。	但尺寸与字库的尺寸



	不一致时,会出现缩放;如果需要自己的字体,请使用 zfontmaker 制作专门的字
	体文件。
语法	SET_FONT (width, height, [fontname])
	width: 字体宽度, 英文自动减半
	height: 字体高度
	fontname: 使用的字体文件名,不设置时不修改当前字体
适用控制器	支持 RTHmi
例子	SET_FONT(16,16, "16X16 字体文件小五.zft")
相关指令	<u>SET_COLOR</u>

6.3.32. SET_COLOR -- 设置颜色

水玉	日二七人		
尖 型	亚不指令		
描述	指定之后 draw 指令使用的颜色,不设置颜色默认黑色。		
	此指令只能在自定义元件的绘图函数内使用,请查看自定义元件调用函数参考例		
	程。		
语法	SET COLOR (cor[,backcor])		
	cor: 线段颜色		
	backcor: DRAWTEXT 时的背景色,不填的时候为透明(-1)		
适用控制器	支持 RTHmi		
例子	例一:		
	SET_COLOR(RGB(255,255,255)) '设置颜色为白色		
	例二:		
	SET COLOR(RGB(0,0,0),RGB(255,0,0)) '颜色黑色, DRAWTEXT 显示的字		
	符串背景为红色。		
	DRAWRECT(0,0,200,100) '自定义元件内绘制边框		
	DRAWTEXT(10,10, "运动控制器") '自定义元件内显示字符串		
	运动控制器		
	例三:		
	SET COLOR(RGB(0,150,255),RGB(255,255,0)) '蓝色线条,黄色填充背景		
	DRAWEX RECT(50,50,200,150,0,1) '自定义元件内绘制填充矩形		
	DRAWEX RECT(50,50,200,150,0) (自定义元件内绘制矩形边框		



相关指令	RGB

6.3.33. SETEX_ALPHA -- 设置绘图透明度

类型	显示指令
描述	绘图透明度设置。
	从 RTHMI_V1.3.0 版本开始,该指令对所有 DRAWEX 绘图指令、滚动条指令和绘
	图片指令(DRAWPIC、DRAWLIBPIC)生效。
语法	SETEX_ALPHA(alpha)
	alpha: 透明度值, 0-不透明, 100-完成透明
适用控制器	支持 RTHmi
例子	SETEX_ALPHA(50) '设置透明度 50
	DRAWEX_ELLIPSE(150, 150, 100, 100, 1)

6.3.34. SET_REDRAW -- 重新绘图

类型	显示指令	
描述	设置指定区域要重新绘制,自定义元件的刷新函数中使用。	
	此指令只能在自定义元件的刷新函数内使用,请查看自定义元件调用函数参考例	
	程。	
语法	SET_REDRAW ([x, y, width, height])	
	X,Y: 显示区域左上角的坐标位置	
	width, height: 区域宽和高	
	无参数时绘制全部区域	
适用控制器	支持 RTHmi	
例子	SET_REDRAW '重新绘制全部区域	



6.3.35. RGB -- 颜色属性

类型	显示指令	
描述	生成一个颜色。	
语法	$\overline{\text{COR} = \text{RGB}(\text{R},\text{G},\text{B})}$	
	RGB: 代表红、绿、蓝三个通道的颜色	
	R,G,B: 每个分量的颜色 0-255	
适用控制器	支持 RTHmi	
例子	RGB(255,255,0) '纯黄色	
相关指令	<u>SET_COLOR</u>	

6.3.36. HMI_LANG -- 文本库语言切换

类型	显示指令		
描述	选择语言版本,文本库可以自动切换语言,参见 <u>文本库</u> 例程		
语法	Hmi_LANG = ilang		
	语言编号 0-7		
适用控制器	支持 RTHmi		
例子	Hmi_LANG=0 '选择文本库语言编号 0 的内容		
相关指令	/		

6.3.37. SCROLLBAR_FREE -- 释放滚动条

类型	显示指令
描述	释放滚动条。
	删除释放已创建的滚动条,释放后的滚动条 ID 将在自定义控件中的刷新函数、绘
	图函数的调用失效。
语法	SCROLLBAR_FREE (id)
	id: 滚动条唯一编号
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	/

6.3.38. SCROLLBAR_INIT -- 滚动条初始化

类型	显示指令
描述	指定 ID 初始化一个滚动条。具体使用可参见 <u>滚动条使用</u> 例程。
	可用滚动条 ID(0~31)
语法	$SCROLLBAR_INIT(id,x1,y1,width,height,maxlines,showlines[\ forecor,backcor,pressc \ forecor,ba$
	or])



	id = SCORLLBAR_INIT (x1, y1, width, height, maxlines, showlines,
	[forecor,backcor,presscor])
	id:滚动条唯一编号,不传 id 参数会自动获取可用 ID 并返回
	作为返回值,返回-1 表示无空闲的 ID,自动初始化失败
	xl,yl: 滚动条左上角位置(以自定义元件框的左上角为原点)
	width,height: 滚动条宽高
	maxlines:最大显示长度
	showlines: 一页显示长度
	当 maxlines=20, showlines=10 时, 滚动条可滚动(20-10=10)个单位,
	滚动值(SCROLLBAR_POS)范围为 0~10
	forecor: 滚动条颜色,可缺省
	backcor: 滚动条育意颜色,可缺省
	presscor: 滚动余按压颜巴,可缺省
	特别说明
	行劢 仍切:
	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	SCROLLBAR INIT(0, 0, 0, 21, 330, 50, 10, RGB(255,0,0), RGB(0,255,0),
	RGB(0,0,255))
	滚动条显示效果:
	按住滚动条时显示效果:
相关指令	SCROLLBAR_POS, SCROLLBAR_REFLASH, SCROLLBAR_DRAW

6.3.39. SCROLLBAR_POS -- 获取/设置滚动值

类型	刷新指令
描述	设置/获取指定滚动条的当前滚动位置。
	处理滚动条刷新事件,控制滚动条上下拖动、滚动
	此指令只能在自定义元件的刷新函数内使用。具体使用可参见滚动条使用例程。
语法	自动刷新滚动条事件
	value = SCROLLBAR_POS(id [,winid,controlid])
	id: 滚动条唯一编号
	value: 返回当前滚动值,返回-1表示无刷新
	winid: 窗口 ID
	controlid: 控件 ID
	' 设置滚动值
	SCROLLBAR_POS (id, value [, winid, controlid])
	id: 滚动条唯一编号
	value: 强制设定当前滚动值

	winid: 窗口 ID
	controlid: 控件 ID
	注: 非自定义控件上使用时, id=0 表示操作垂直滚动条, id=1 表示水平滚动条。
	若控件只有一个滚动条,则 id 参数无效。
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	刷新函数
	value = SCROLLBAR_POS(0)
	?"当前滚动值: "value
相关指令	SCROLLBAR_INIT, SCROLLBAR_REFLASH, SCROLLBAR_DRAW

6.3.40. SCROLLBAR_REFLASH -- 刷新滚动条

类型	刷新指令
描述	刷新滚动条。
1HAL	处理滚动条刷新事件,控制滚动条上下拖动、滚动。
	此指令只能在自定义元件的刷新函数内使用。具体使用可参见滚动条使用例程。
语法	·自动刷新滚动条事件,自动触发重绘
	SCROLLBAR_RELASH (id)
	id: 滚动条唯一编号
	'滚动条事件触发时,用户可根据返回值来判断是否重绘
	bifupdate = SCROLLBAR_RELASH (id)
	id: 滚动条唯一编号
	bifupdate: 是否触发更新, 0-未触发 or 1-触发
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	例子 1:
	,刷新函数
	SCROLLBAR_RELASH (0)
	·绘图函数
	SCROLLBAR_DRAW (0)
	例子 2:
	'刷新函数(需通过自定义元件的刷新函数进行调用)
	global sub_reflash()
	bifredraw = SCROLLBAR_RELASH (0)
	if $bifredraw > 0$ then
	?"当前滚动值: "SCROLLBAR_POS (0)
	SET_REDRAW
	endif
	end sub

	'绘图函数(需通过自定义元件的绘图函数进行调用)
	global sub_redraw()
	SCROLLBAR_DRAW (0)
	end sub
	▲ ▶
相关指令	SCROLLBAR_INIT, SCROLLBAR_DRAW, SCROLLBAR_POS

6.3.41. SCROLLBAR_DRAW -- 绘制滚动条

类型	绘图指令
描述	滚动条绘图指令。此指令只能在自定义元件的绘图函数内使用。具体使用可参见
	<u>滚动条使用</u> 例程。
	滚动条显示并不限制在自定义控件内,请用户使用前计算好空间位置,避免使用
	异常。
语法	SCROLLBAR_DRAW (id)
	id: 滚动条唯一编号
适用控制器	支持 RTHmi (仅支持 4 系列及以上控制器)
例子	SCROLLBAR_DRAW (0)
	*
相关指令	SCROLLBAR_INIT, SCROLLBAR_POS, SCROLLBAR_REFLASH

6.4. 触摸屏指令

6.4.1.TOUCH_ADJUST -- 触摸屏校正

类型	触摸屏指令
描述	进行触摸屏校正,此时不要刷新屏幕,校正后参数会自动保存。
	注意:该指令连接 ZHD400X 示教盒、500X 示教盒或支持该指令的手持盒设备触
	摸屏上才起效!
	【校正方式】
	方式 1: 通过点击设置界面的 Touch Adjust 功能进入触摸校正界面
	(通过左上,右上,左下,右下,左上,右上,左下,右下的方式连续点击,可以
	弹出设置窗口,可以进行触摸校正。)
	方式 2: 连接控制器后,通过控制器的 TOUCH_ADJUST 指令来触发校正。
	方式 3:不连接控制器,按下 12(F2)按键不松开,再按下 11(F1)按键。
	不连接控制器,按下16(F6)按键,不松开时继续按下11(F1)按键
语法	TOUCH_ADJUST ()
适用控制器	支持 RTHmi
例子	用例 1:



	(1) 使用一个 Hmi 按钮控件 校正触摸屏
	(2) 在该控件的"属性"中调用子函数
	⊿ 标签
	文本库
	格式文本(0) 校正触摸屏
	格式文本(1)
	⊿ 动作
	动作 调用函数
	松开时动作 False
	动作函数名 correcting
	(3) Basic 子函数
	global sub correcting()
	TOUCH_ADJUST()
	end sub
	(4) 运行含有该指令的程序项目,点击该按钮
	(5) 直接进入触摸校正界面,后根据"+"的位置长按进行校正即可。
	用例 2: 直接放在桯序初始化函数里
	IOUCH_ADJUSI() 、进行肥捑仪止
关指令	/

6.4.2.TOUCH_SCAN -- 触摸动作扫描

类型	触摸屏指令
描述	扫描触摸按下动作,返回1表示扫描到按下,-1表示有松开,0没有变化,将触摸
	对应的位置的 X,Y 坐标保存到 table 表中。
	预留触摸屏模拟鼠标右键、中键功能。
语法	action = TOUCH_SCAN (num [, optkey])
	num: 存储触摸 XY 坐标的 table 编号, X,Y 坐标分别存储在 table(num),
	table(num+1)
	optkey:操作鼠标按键,可选,默认为0鼠标左键
	0 – 鼠标左键
	1 - 鼠标右键
	2 - 鼠标中键
	action: 返回鼠标动作, 1-按下, -1 松开, 0 无变化
适用控制器	支持 RTHmi
例子	例一:
	IF TOUCH_SCAN(0)=1 THEN '扫描按下操作,显示按下位置
	?"按下的位置为:" TABLE(0),TABLE(1)
	ENDIF



	IF TOUCH_SCAN(0) = -1 THEN '扫描松开操作
	?"松开"
	ENDIF
	例二:
	IF TOUCH_SCAN(0)=1 THEN '扫描按下操作
	TIMES = TIME
	ENDIF
	IF TOUCH_SCAN(0) = -1 THEN '扫描松开操作
	?"按下的时间为:" TIME-TIMES
	ENDIF
相关指令	MOUSE_SCAN

6.4.3.TOUCH_STATE -- 获取触摸状态

类型	触摸屏指令
描述	读取触摸状态,返回>0表示按下,返回0表示松开,将触摸对应的位置的X,Y坐
	标保存到 table 表中。
	预留触摸屏模拟鼠标右键、中键功能。
语法	<pre>state = TOUCH_STATE (num [, optkey])</pre>
	num: 触摸对应的位置 X,Y 坐标分别存储在 table(num), table(num+1)
	optkey:操作鼠标按键,可选,默认为0鼠标左键
	0 – 鼠标左键
	1 - 鼠标右键
	2 - 鼠标中键
	state: 返回鼠标状态, 1-按下, 0松开
适用控制器	支持 RTHmi
例子	WHILE 1
	IF SCAN_EVENT(TOUCH_STATE(0))>0 THEN
	?"按下的位置为: "TABLE(0),TABLE(1)
	ENDIF
	IF SCAN_EVENT(TOUCH_STATE(0))<0 THEN
	?"松开"
	ENDIF
	WEND
相关指令	MOUSE_STATE

6.4.4.MOUSE_SCAN -- 鼠标动作扫描

类型	触摸屏指令
描述	扫描鼠标按下动作,返回 1 表示扫描到按下,-1 表示有松开,0 没有变化,将触



	摸对应的位置的 X,Y 坐标保存到 table 表中。
	注:使用该指令需通过函数调用方式实现。
语法	action = MOUSE_SCAN (num [,optkey])
	num: 触摸对应的位置的 X,Y 坐标分别存储在 table(num), table(num+1)
	optkey:操作鼠标按键,可选,默认为0鼠标左键
	0- 鼠标左键
	1 - 鼠标右键
	2- 鼠标中键
	action:返回鼠标动作,1-按下,-1松开,0无变化
适用控制器	支持 RTHmi
例子	IF MOUSE_SCAN(0)=1 THEN '扫描按下操作,显示按下位置
	?"按下的位置为:" TABLE(0),TABLE(1)
	ENDIF
	IF MOUSE_SCAN(0) = -1 THEN '扫描松开操作
	?"松开"
	ENDIF
相关指令	TOUCH_SCAN

6.4.5.MOUSE_STATE -- 获取鼠标状态

类型	触摸屏指令						
描述	读取鼠标状态,返回>0表示按下,返回0表示松开,将触摸对应的位置的X,Y坐						
	标保存到 table 表中。						
语法	state = MOUSE_STATE (num [, optkey])						
	num: 触摸对应的位置的 X,Y 坐标分别存储在 table(num), table(num+1)						
	optkey:操作鼠标按键,可选,默认为0鼠标左键						
	0- 鼠标左键						
	1 - 鼠标右键						
	2 - 鼠标中键						
	state: 返回鼠标状态, 1-按下, 0松开						
适用控制器	支持 RTHmi						
例子	WHILE 1						
	IF SCAN_EVENT(MOUSE_STATE(0))>0 THEN						
	?"按下的位置为: "TABLE(0),TABLE(1)						
	ENDIF						
	IF SCAN_EVENT(MOUSE_STATE(0))<0 THEN						
	?"松开"						
	ENDIF						
	WEND						
相关指令	TOUCH_STATE						

6.5. 按键指令

6.5.1.MOUSE_WHEEL -- 获取鼠标滚轮值

类型	虚拟按键指令			
描述				
语法	vheel = MOUSE_WHEEL (tabid)			
	tabid: 鼠标位置的 X,Y 坐标分别存储在 table(tabid), table(tabid +1)			
	wheel: 返回鼠标滚轮值,-120表示向下滚动一格,120表示向上滚动一格			
适用控制器	支持 RTHmi (5 系列及以上, 4 系仅 ZMC432 最新固件支持)			
例子	wheelval = MOUSE_WHEEL(0)			
	if wheelval > 0 then			
	wheelval = wheelval/120 '表示滚动量, >0 向上滚动, <0 向下滚动			
	endif			

6.5.2.KEY_STATE -- 获取物理按键状态

类型	按键指令				
描述	读取物理按键状态,1-按下。				
	此指令只能在自定义元件的刷新函数内使用,请查看自定义元件调用函数参考例				
	程。				
语法	KEY_STATE (keynum)				
	keynum: 按键编号				
适用控制器	支持 RTHmi				
例子	num =KEY_SCAN()				
	IF KEY_STATE (num)=1 THEN '扫描按下操作,打印按键编号				
	?"按下" num				
	ENDIF				
相关指令	KEY_EVENT, KEY_SCAN				

6.5.3.KEY_EVENT -- 获取物理按键状态变化

类型	按键指令				
描述	读取物理按键状态变化,1-按下,-1-松开,0-不变。				
	此指令只能在自定义元件的刷新函数内使用,请查看自定义元件调用函数参考例				
	程。				
语法	KEY_EVENT (keynum) Keynum 按键编号				
适用控制器	支持 RTHmi				
例子	num =KEY_SCAN()				



		F KEY_EVENT (num)=1 THEN '扫描按下操作,打印按键编号 2"按下" num	
		ENDIF	
柎	目关指令	KEY_SCAN, KEY_STATE	

6.5.4.KEY_SCAN -- 获取物理按键编码

类型	按键指令							
描述	读取当前按下的物理按键编码,按下返回按键编码,当松开的时候按键编码的负							
	数,返回 0 表示没有按键状态变化。							
	物理按键编码值由硬件决定,程序中无法修改,但可以修改与物理键绑定的虚拟							
	键。							
	此指令只能在自定义元件的刷新函数内使用,请查看 <u>自定义元件调用函数</u> 参考例							
	程。							
语法	value=KEY_SCAN()							
适用控制器	支持 RTHmi							
例子	num =KEY_SCAN()							
	IF KEY_EVENT (num)=1 THEN '扫描按下操作,打印按键编号							
	?"按下" num							
	ENDIF							
相关指令	KEY_EVENT,KEY_STATE							

6.5.5.VKEY_MODE -- 开启虚拟键输入模式

类型	虚拟按键指令								
描述	开启/关闭虚拟键输入模式,开启虚拟键输入模式后,值显示、字符显示控件将会获得焦点(同一时刻有且只有一个控件获取焦点),获得焦点的控件可以直接接收虚拟键值,且获得高亮提示(如下第一行第二个控件)								
	GG. NC								
语法	VKEY_MODE (mode[, winid])								
	mode = VKEY_MODE()								
	mode: 0 - 关闭虚拟键输入模式								
	1- 开启虚拟键输入模式 1, 可编辑控件有效;								
	点击切换焦点,若当前处于焦点状态,再点击弹出键盘								
	2- 开启虚拟键输入模式 2, 可编辑控件有效, 屏蔽弹出键盘								
	3- 开启虚拟键输入模式 3, 可编辑控件有效								
	通过 TAB 键切换焦点,首次按下 ENTER 键后才能进入编辑状态								
	(未进入编辑状态不可输入,进入编辑状态后显示输入光标),再次								

	按 ENTER 则确认编辑并切换下一个控件												
	4- 开启虚拟键输入模式 4, 可编辑控件有效												
	跟模式 3 一样, 不同的是模式 4 进入编辑状态会默认清空原来的数												
		据											
	wi	winid: 指定窗口号, 缺省为当前(顶层)窗口											
	注:窗	口号街	央省时	,会自	司动扫	描最近	反层窗	口并切	J换当ī	前焦点	〔,只自	能在当	前最顶层
	窗口上	使用轴	俞入焦	点功食	匙,无 :	法在其	(他窗)	口上使	ē用。				
适用控制器	支持 R	THmi											
例子	VKEY_	MOD	PE(1)	'开启	自虚拟	键输)	、模式						
	VKEY_	_MOD	E(1, 1	0) '	指定窗	ā⊓ 10) 开启	虚拟镜	建输入材	漠式			
											0		
											0.000	0000	
					<-		->						
	1	2	3	4	5	6	7	8	9	0	-	Back	
	Q	Ŵ	E	R	Т	Y	U	I	0	Р	"	Clr	
	A	S	D	F	G	Н	J	K	L	En	ter	关闭	
	Z	х	С	v	В	N	M	•	?	Sp	ace	TAB	
相关也么	WEEV	SCAR		ZEV S	TATE								
伯大伯文	VNEY	SUAL	<u>v</u> , <u>vr</u>	$\underline{\mathbf{r}}$	ALE								

6.5.6.VKEY_STATE -- 设置/获取虚拟键状态

类型	虚拟按键指令
描述	读取虚拟按键状态,1-按下、0-松开。
	此指令只能在自定义元件的刷新函数内使用,请查看 <u>自定义元件调用函数</u> 参考例
	程。
语法	VKEY_STATE (keynum,state)
	state=VKEY_STATE(keynum)
	keynum: 虚拟按键编号
	state: 虚拟按键状态
适用控制器	支持 RTHmi
例子	例一:
	'设置虚拟按键状态,通过功能键选择动作"Call Sub Twice",分别指定按下时和松
	开时调用函数
	global sub VKeyPress_A()
	VKEY_STATE (65, 1)



	end sub
	global sub VKeyRelease_A()
	VKEY_STATE (65,0)
	end sub
	例二:
	'自定义控件刷新函数内获取虚拟按键状态
	num =VKEY_SCAN()
	IF VKEY_STATE (num)=1 THEN '扫描按下操作,打印按键编号
	?"按下" num
	ENDIF
相关指令	VKEY_SCAN, VKEY_EVENT

6.5.7.VKEY_EVENT -- 获取虚拟键状态变化

类型	虚拟按键指令
描述	读取虚拟按键状态变化, 1-按下, -1-松开, 0-不变。
	此指令只能在自定义元件的刷新函数内使用,请查看自定义元件调用函数参考例
	程。
语法	VKEY_EVENT (keynum)
	keynum: 按键编号
适用控制器	支持 RTHmi
例子	num =VKEY_SCAN()
	IF VKEY_EVENT (num)=1 THEN '扫描按下操作,打印按键编号
	?"按下" num
	ENDIF
相关指令	VKEY_SCAN,VKEY_STATE

6.5.8.VKEY_SCAN -- 获取虚拟键编码

类型	按键指令
描述	读取当前按下的虚拟按键编码,按下返回按键编码,当松开的时候按键编码的负
	数,返回0表示没有按键操作。
	虚拟按键编码内部已经定义,无法修改,但可以修改与虚拟键绑定的物理键。
	此指令只能在自定义元件的刷新函数内使用,请查看自定义元件调用函数参考例
	程。
语法	value=VKEY_SCAN()
适用控制器	支持 RTHmi



例子	Dim Curkey
	Curkey = VKEY_SCAN() '读取当前按键值(消息码)
相关指令	VKEY_STATE, VKEY_EVENT

6.5.9.VKEY_INPUT -- 输入虚拟键值内容到键盘窗口

类型	按键指令
描述	键盘窗口上控件模拟虚拟按键输入。
语法	VKEY_INPUT (vkeynum)
	vkeynum: 虚拟按键编号
适用控制器	支持 RTHmi
例子	/
相关指令	VKEY_STATE, VKEY_EVENT

6.5.10. VKEY_IME -- 设置/获取当前输入法

类型	虚拟按键指令
描述	设置/获取当前输入法类型
	中文输入仅对虚拟键 VKEY_a~VKEY_z 有效,对输入的大写字符无效。
语法	ime = VKEY_IME ()
	VKEY_IME (ime)
	ime: 当前输入法类型
	0: 默认值,英文输入法
	1: 中文拼音输入法
适用控制器	支持 RTHmi
例子	VKEY_IME (1)
	? VKEY_IME () ; 输出 1

6.5.11. ZSIMU_KEY -- 仿真物理按键

类型	按键指令
描述	仿真物理按键。
语法	ZSIMU_KEY(keycode, state)
	keycode: 物理按键编码
	state: 按下状态, 1-按下
适用控制器	支持 RTHmi
例子	ZSIMU_KEY(3,1) '仿真物理键 3 被按下
相关指令	ZSIMU_VKEY

6.5.12. ZSIMU_VKEY -- 仿真虚拟按键

类型	按键指令
描述	仿真虚拟按键。
语法	ZSIMU_VKEY(vkeycode, state)
	vkeycode: 虚拟按键编码
	state: 按下状态, 1-按下
适用控制器	支持 RTHmi
例子	ZSIMU_VKEY(4,1) '仿真虚拟键 4 被按下
相关指令	ZSIMU_KEY

6.6.操作指令

6.6.1.HMI_SHOWWINDOW -- 显示指定窗口

类型	窗口操作指令
描述	显示指定窗口。
	软键盘窗口要在编辑窗口的相关函数里面调用,否则无法确定是哪个 窗口的元件
	要编辑。
	不确定时默认选择最顶层窗口。
语法	HMI_SHOWWINDOW(winid [,showmode] [,modifycontrol] [,ifclosewithfather]
	[,fatherwindow])
	winid: 窗口号
	showmode:显示方式
	ZPLC_WIN_TYPE_AUTO=0,Hmi文件里面指定的窗口模式
	$ZPLC_WIN_TYPE_TOP = 1$
	$ZPLC_WIN_TYPE_BOTTOM = 2$
	ZPLC_WIN_TYPE_BASE=4,基本窗口 只能有一个的,一旦切换所有
	的子窗口都关闭.
	ZPLC_WIN_TYPE_KEYBOARD = 5, 软键盘必然弹出窗口
	$ZPLC_WIN_TYPE_POP = 6, 弹出窗口$
	ZPLC_WIN_TYPE_MENU=7,菜单窗口,自动关闭
	Modifycontrol: ZPLY_WIN_TYPE_KEBOARD 弹出软键盘窗口类型时,对应
	要编辑的本窗口元件 ID
	ifclosewithfather: 是否随父窗口一起关闭,缺省 1=true
	fatherwindow: 指定父窗口,缺省当前窗口
	注意:
	1.Modifycontrol 参数不可缺省,且元件 ID 必须是支持软键盘输入的元件。
	2.软键盘默认垄断,无法通过元件关闭,但可使用指令操作关闭,同时随所在
	窗口的关闭而关闭。
	3.从哪个窗口打开另一个窗口,即为父子窗口关系。
适用控制器	支持 RTHmi



例子	例一:在窗口10设置按键弹出13窗口,窗口10即为窗口13的父窗口。
	HMI_SHOWWINDOW(12,6) '弹出窗口 12
	HMI_SHOWWINDOW(13,6,0,1) '弹出窗口 13 并随父窗口关闭
	HMI_SHOWWINDOW(13,6,0,1,11) '弹出窗口 13 并指定随父窗口 11 关闭
	例二
	HMI_SHOWWINDOW(8,5,1) '弹出软键盘窗口 8,关联当前窗口 1 号元件
相关指令	HMI_CLOSEWINDOW

6.6.2.HMI_CLOSEWINDOW -- 关闭窗口

类型	窗口操作指令
描述	关闭指定窗口。
语法	HMI_CLOSEWINDOW([winid])
	winid: 窗口号, 缺省 0-当前函数调用的元件所在的窗口, 其它编号-Hmi 组
	态里面的窗口号
适用控制器	支持 RTHmi
例子	HMI_CLOSEWINDOW() '关闭当前窗口
相关指令	HMI_SHOWWINDOW

6.6.3.HMI_BASEWINDOW -- 切换基本窗口

类型	窗口操作指令
描述	切换基本窗口。
语法	HMI_BASEWINDOW = winid
	winid: Hmi 文件里面窗口编号
适用控制器	支持 RTHmi
例子	HMI_BASEWINDOW = 11 '切换到 11 号基本窗口
相关指令	HMI_SHOWWINDOW

6.6.4.HMI_FOCUSWINDOW -- 窗口焦点模式

类型	窗口操作指令
描述	设置/获取 HMI 窗口是否随鼠标点击焦点自动进行切换(被点中的窗口将自动在
	最上层显示)
	注: 弹出窗口即使切换到最顶层, 也始终在置顶窗口的下方。
语法	HMI_FOCUSWINDOW(mode)
	mode = HMI_FOCUSWINDOW()
	mode: 焦点窗口切换模式

	0: 默认不切换
	1: bit0=1, 弹出窗口(POP)随鼠标点击焦点切换
	2: bit1=1, 置顶窗口(TOP)随鼠标点击焦点切换
	注: 弹出窗口使用该指令焦点切换不能覆盖置顶窗口。
适用控制器	支持 RTHmi
例子	HMI_FOCUSWINDOW(1) '设置弹出窗口随鼠标焦点自动切换
	HMI_FOCUSWINDOW(2) '设置置顶窗口随鼠标焦点自动切换
	HMI_FOCUSWINDOW(1+2) '设置置顶和弹出窗口都随鼠标焦点自动切换
	以弹出窗口为例(其余模式等同),运行效果图如下:

6.6.5.HMI_LASTWINDOW -- 最后点击窗口

类型	窗口操作指令
描述	获取最后点击的窗口
语法	winid = HMI_LASTWINDOW()
适用控制器	支持 RTHmi
例子	?HMI_LASTWINDOW() '打印最后点击的窗口 ID

6.6.6.HMI_DEFAULTATTR -- 设置/获取 HMI 内置默认属性

类型	HMI 指令
描述	设置/获取 HMI 内置的默认属性,对整个 HMI 生效。
	注:需要在HMI初始化前进行设置才会生效。
语法	value = HMI_DEFAULTATTR(strAttr)
	HMI_DEFAULTATTR(strAttr, value)
	strAttr: 指定操作的控件属性
	"SBR_W": 滚动条宽度
	"SBR_F": 滚动条前景颜色
	"SBR_B":滚动条背景颜色
	"SBR_P": 滚动条按压颜色
	"BTN_RLM": 按钮松开模式
	0-默认
	1-鼠标移出控件取消触发

	"DRAW_M": 绘图模式
	0-默认点阵绘图
	1-抗锯齿绘图
	"MONO_M": 控件垄断模式(有弹窗的控件,如下拉列表、菜单栏等)
	0-不垄断
	1-垄断按键
	2-垄断鼠标
	3-同时垄断按键鼠标(默认)
	"RGB_GRAY": 灰阶颜色
	注:在 HMI 程序启动之前,必须预先配置好 SBR_W、SBR_F、SBR_B、SBR_P
	以及 RGB_GRAY 等参数。为此,可以在 Basic 程序的末尾 添加一个专门用于
	启动 HMI 的函数。这样做可以可以使必要参数都已正确设置后再启动 HMI,以确
	保其按照预期正常工作。
	value: 属性值
适用控制器	支持 RTHmi
例子	HMI_DEFAULTATTR("SBR_W", 25) '修改滚动条默认宽度为 25
	HMI_DEFAULTATTR("SBR_P", RGB(255,0,0)) '修改滚动条默认按压颜色

6.6.7.HMI_DEALINFO -- 获取 HMI 处理信息

类型	HMI 指令
描述	获取当前 HMI 处理的信息
语法	value = HMI_DEALINFO(strAttr)
	strAttr: 指定获取的 HMI 处理信息
	"CURWIN": 当前窗口号
	"CURCTRL": 当前控件号
	"TOPWIN": 最顶层显示的窗口号
	"EDITWIN":当前编辑输入窗口
	"EDITCTRL": 当前编辑输入控件
	"EDITNEXT":当前焦点的下一个编辑控件
	"EDITPREV":当前焦点的上一个编辑控件
	value: 数值
	注: "EDITWIN"、"EDITCTRL"、"EDITNEXT"、"EDITPREV"参数需要在虚拟
	键模式下才生效
适用控制器	支持 RTHmi
例子	例一:
	global dim aaa
	aaa=HMI_DEALINFO("CURWIN")
	?aaa
	例二:在HMI窗口中添加"可编辑"的控件
	VKEY_MODE(1) '开启虚拟键输入模式


?HMI_DEALINFO("EDITCTRL") '打印当前编辑的控件编号 打印结果为:当前编辑的控件编号

6.6.8.HMI_CONTROLSIZEX -- 获取元件宽度

类型	窗口操作指令
描述	获取元件宽度
语法	value= HMI _CONTROLSIZEX ([winid, controlid])
	winid: HMI 文件里面窗口编号
	controlid: 元件编号,缺省为当前自定义元件宽度
适用控制器	支持 RTHmi
例子	PRINT HMI_CONTROLSIZEX(10,11) '打印 10 号窗口 11 号元件宽度
相关指令	HMI_CONTROLSIZEY

6.6.9.HMI_CONTROLSIZEY -- 获取元件高度

类型	窗口操作指令
描述	获取元件高度
语法	value= HMI_CONTROLSIZEX ([winid, controlid])
	winid: HMI 文件里面窗口编号
	controlid: 元件编号,缺省为当前自定义元件高度
适用控制器	支持 RTHmi
例子	PRINT HMI_CONTROLSIZEX (10,11) '打印 10 号窗口 11 号元件高度
相关指令	HMI_CONTROLSIZEX

6.6.10. HMI_CONTROLDATA -- 设置/获取自定义元件属性

类型	窗口操作指令			
描述	获取或设置自定义元件的特殊属性,在Hmi里面指定,通过这个可以区分多个类			
	似的元件。			
语法	value= HMI_CONTROLDATA ([winid, controlid])			
	HMI_CONTROLDATA (winid, controlid) = value			
	winid: Hmi 文件里面窗口编号			
	controlid: 元件编号,缺省为当前自定义元件			
适用控制器	支持 RTHmi			
例子	HMI_CONTROLDATA(10,1)=5 '设置多个自定义元件为相同属性			
	HMI_CONTROLDATA(10,2)=5			
相关指令	HMI_CONTROLSIZEX, HMI_CONTROLSIZEY			

6.6.11. HMI_CONTROLBACK -- 设置/获取指定元件背景颜色

类型	窗口操作指令
描述	获取或设置值显示及字符显示元件的背景颜色
语法	value= HMI_CONTROLBACK ([winid, controlid])
	HMI_CONTROLBACK (winid, controlid) = value
	winid: HMI 文件里面窗口编号
	controlid: 元件编号, 缺省为当前自定义元件
适用控制器	支持 RTHmi
例子	HMI_CONTROLBACK(10,1)=RGB(255,255,0) '黄色
	HMI_CONTROLBACK(10,1)=RGB(255,0,0) ′红色
相关指令	RGB

6.6.12. HMI_CONTROLVALID -- 设置/获取元件使能

类型	窗口操作指令			
描述	获取或设置元件的使能,在 Hmi 里面可以指定。			
语法	value= HMI_CONTROLVALID ([winid, controlid])			
	HMI_CONTROLVALID (winid, controlid) = value			
	winid: HMI 文件里面窗口编号			
	controlid: 元件编号, 缺省为当前自定义元件			
	value: 为1时,元件触摸有作用为0时,元件触摸无作用效果			
适用控制器	支持 RTHmi			
例子	HMI_CONTROLVALID(10,5)=0 '10 号窗口第 5 个元件无作用效果			
相关指令	/			

6.6.13. HMI_CONTROLSTRING -- 获取字符串信息

类型	窗口操作指令			
描述	获取字符串控件或输入显示控件的字符串信息。			
	5 系列 20180405 以上固件支持。			
语法	string= HMI_CONTROLSTRING ([winid, controlid])			
	HMI_CONTROLSTRING (winid, controlid) = string			
	winid: HMI 文件里面窗口编号			
	controlid: 元件编号,缺省为当前自定义元件			
	String: 字符串			
适用控制器	支持 RTHmi			
例子	VRSTRING(0,10)= HMI_CONTROLSTRING (10, 2)	'10号窗口第2个元件的数		
	据读取到 VR			
相关指令	/			

6.6.14. HMI_CONTROLATTR -- 设置/获取元件属性

类型	控件操作指令					
描述	获取/设置控件的指定属性值					
	注:操作前需确保操作的控件具有该属性。					
语法	value=HMI_CONTROLATTR(strProperty[, winid, controlid])					
	HMI_CONTROLATTR(strProperty, value[, winid, controlid])					
	srtProperty:指定操作的控件属性					
	"POSX": 控件 X 位置					
	"POSY": 控件 Y 位置					
	"SIZEX": 控件宽度					
	"SIZEY": 控件高度					
	"MIN":最小值(包含该属性的单一状态控件才有效,下同)					
	"MAX":最大值					
	"FORE": 前景颜色					
	"BACK": 背景颜色					
	"PWD": 密码显示					
	"DECS": 小数位数					
	"STATE": 控件状态					
	"POPUP": 弹出状态(下拉列表)					
	"CURSORX": 招忤输入光标列					
	"CURSORY": 控件输入光标行					
	"FOCUS": 拴什焦点状态					
	"MSELROWS": 多选行数,止数往下多选,贝数往上多选,0 表示尤多选。 配合"CURSORV"会粉基取业长行可知道光觉选中行区域					
	远。配告"CURSORY"参数获取元称11 可知道当前远中11 区域。					
	value: 属性值					
	winid: HMI 文件里面窗口编号,缺省为当前窗口					
	controlid: 元件编号,缺省为当前自定义元件					
适用控制器	支持 RTHmi					
例子	例一:					
	HMI_CONTROLATTR ("FORE",RGB(255,0,0),10,1) '设置窗口 10 控件 1 的前景颜					
	色为红色					
	HMI_CONTROLATTR ("BACK",RGB(255,255,255),10,1) '设置窗口 10 控件 1 的背					
	景颜色为白色					
	▶ 设置前的样式:					
	※ 圳市正 云 寺 井 未					
	林州中正巡朔权不					

▶ 设置后的样式	×.								
深圳市	正运动技术	:							
例二: VRSTRING(0,8) HMI_CONTROI 示) = "123456" LATTR("PWD",1234	56,10,5	5) '	设置	窗口	10 控	这件 5 自	的字符为	密码显
***** 例三: HMI_CONTROI	LATTR("MIN",10,10	,6) 'ť	及置伯	直显	示控何	牛的占	最小值		
HMI_CONTROI 注:最小值数值 的最小值和最大	LATTR("MAX",100, 必须比最大值小才有 :值的传入值。	10,6) ī效,,	'设旨 大于'	置值! 等于:	显示哲 均无多	空件白 汝。┏	り最大 [↓] 同时应 [↓]	值 考虑"属	生"栏中
10:Start	xplc screen		10	0					
		1	2	3	-				
		7	5 8	9	Esc				
			0	En	ter				

6.6.15. HMI_CONTROLTEXT -- 修改元件文本

类型	控件操作指令			
描述	修改控件显示文本,支持所有带格式文本(可输入)的控件。			
	注: 修改文本长度不能超过当前控件文本,否则会被截断。			
语法	HMI_CONTROLTEXT (winid, controlid, state, strText)			
	winid: HMI 文件里面窗口编号			
	controlid: 元件编号			
	state: 指定修改哪个状态的控件文本			
	strText: 修改控件显示文本			
适用控制器	支持 RTHmi			



例子	HMI_CONTROLTEXT (10	0,1,0,"松开") '修改	按钮松开状态显	显示文本		
	HMI_CONTROLTEXT (10	0,1,1,"按下") '修改	按钮按下状态显	显示文本		
	· · · · · · · · · · · · · · · · · · ·	⊿ 标签				
	I:BitModify 按键	文本库				
		格式文本(0)	按键			
		格式文本(1)	按键			
	修改后元件文本显示情况 左图:0 状态时显示"松开 右图:1 状态时显示"按下	.: f"文本; [、] "文本;				
	松开	按下				

6.6.16. HMI_LISTTEXTS -- 修改列表元件文本

类型	控件操作指令
描述	强制修改列表控件的文本列表,具体应用例程参见 <u>动态列表</u> 。
	注意:使用该指令须先开启"列表"控件的"动态项"属性。
语法	HMI_LISTTEXTS (winid, controlid, strList,[,mode] [,lineheight]])
	winid: Hmi 文件里面窗口编号
	controlid: 元件编号
	strList: 文本列表, 以换行符隔开
	mode: 可选,修改模式,缺省0
	0-覆盖方式,清空原来的列表项重新写入新的列表项
	1 - 追加方式,在原来的列表项基础上末尾增加新的列表项,且滚动
	条自动跟随到追加位置
	lineheight: 可选,强制指定行高度,缺省-1 不指定
	-1- 自动计算行高度, 行高度将随列表项目数量变化
	=0- 使用当前行高度
	>0- 强制指定行高度,小于最小行高度时按最小行高度设置
	最小行高度 = 字高 + 2*行间距
	注意:
	1. 字符串最大字符数限制 2049。
	2. 使用该指令修改的列表项,其列表编号是逐行递增的,不能指定编号,追加操
	作的列表项也是。
	3. 非追加操作方式,列表控件的滚动条位置将会自动复位,追加操作方式可以选
	择保持滚动条在追加位置。
	4. 列表控件对于动态列表项缓冲区有大小限制, 最多 256 项, 当加满 256 项时,
	每一项列表项文本不得超过12个字符。
适用控制器	支持 RTHmi

例子	dim strList(100) = "选项 1\n 选项 2\n 选项 3"			
	HMI_LISTTEXTS (10, 3, strList, 1, 30) '在原有的列表项基础上追加新的列表项,			
	并指定行高度为 30			
	显示结果如下:			
	1 2 3 4 5 选项1 选项2			

6.6.17. HMI_LISTITEM -- 修改指定列表项文本

类型	控件操作指令								
描述	获取/强制修改列表控件的指定列表项文本								
	注意: 使用该指令须先开启"列表"控件的"动态项"属性。								
语法	IMI_LISTITEM (winid, controlid, id, strText)								
	strText = HMI_LISTITEM (winid, controlid, id)								
	winid: Hmi 文件里面窗口编号								
	controlid: 元件编号								
	id: 要修改的列表项 ID								
	strText: 修改的列表项文本								
适用控制器	支持 RTHmi								
例子	dim strList(100) = "选项 1\n 选项 2\n 选项 3"								
	Hmi_LISTTEXTS (10, 3, strList) '修改列表控件文本,自动更新状态数								
	Hmi_LISTITEM (10, 3, 1, "TWO") '修改元件 3 中第二个列表项为"TWO"								
	显示结果如下:								
	1 T₩O 3 4 5 选项1 选项2								

6.6.18. HMI_STRAPPEND -- 元件文本追加

类型	控件操作指令								
描述	在"字符显示"元件中追加字符串								
	注:使用该指令需在元件"属性"中设置为多行文本状态才生效。								
语法	HMI_STRAPPEND (strAppend, winid, controlid, mode [,ifNewLine])								
	strAppend: 追加的字符串								
	winid: HMI 文件里面窗口编号								
	controlid: 元件编号								
	mode:追加模式(bit),按 bit0 和 bit1 的组合值确定模式。(当字符串超过								
	最大字符数限制时,会对添加后的总字符串按模式清除溢出部分)								
	bit0: 溢出方向								
	0- 超限从末尾处清除溢出部分,即超限部分不追加								
	1- 从字符串开头处清除溢出部分								
	bit1: 追加方向								
	0 – 从末尾处追加,滚动条自动跟随到文本末尾处								
	1 – 从起始处追加,滚动条自动跟随到文本起始处								
	ifNewLine: 是否追加新行,缺省0(否)								
	注: 追加新行默认占用一个字符空间								
适用控制器	支持 RTHmi								
例子	例一: HMI_STRAPPEND ("zmotion",10, 2, 0, 0) '向元件文本末尾处追加一行字符								
	串								
	例二, 对元供文本限制字符数为16 对该元件追加文本								
	HMI STRAPPEND (" $zmotion$ " 10 2 2 1) '向元件文本起始处追加字符串,目追加								
	新行								
	运行效果如下: 仅显示不超过 16 个字符数的内容, 且从末尾处清除溢出部分								
	zmotion								
	正运动技								

6.6.19. HMI_IFMONO -- 获取窗口垄断状态

类型	窗口操作指令							
描述	自定义元件 reflash 刷新函数里使用,判断当前元件是否被其他窗口垄断,垄断							
	时不要响应鼠标和按键消息,-1-被垄断,0-没有被垄断							
语法	Value=HMI_IFMONO							
适用控制器	支持 RTHmi							
例子	GLOBAL SUB reflash()							
	IF SCAN_EVENT(Hmi_IFMONO)<0 THEN							
	?"返回自定义元件窗口"							
	ENDIF							
	IF SCAN_EVENT(Hmi_IFMONO)>0 THEN							
	?"离开自定义元件窗口"							
	ENDIF							
	END SUB							

6.6.20. HMI_WINDOWSTATE -- 获取窗口状态

类型	窗口操作指令								
描述	获取窗口状态。								
	20161112 以后固件版本支持。								
语法	value= HMI_WINDOWSTATE (winid [, tablenum])								
	winid: Hmi 文件里面窗口编号								
	tablenum:存储窗口的位置和大小,顺序存储 posx, posy, sizex, sizey								
	返回值对应的窗口类型:								
	ZPLC_WIN_TYPE_AUTO=0,没有显示								
	ZPLC_WIN_TYPE_TOP=1, 顶层窗口								
	ZPLC_WIN_TYPE_BOTTOM = 2, 底层窗口								
	ZPLC_WIN_TYPE_BASE=4, 基本窗口								
	ZPLC_WIN_TYPE_KEYBOARD=5, 软键盘.								
	ZPLC_WIN_TYPE_POP=6, 弹出窗口								
	ZPLC_WIN_TYPE_MENU=7,菜单窗口,自动关闭								
适用控制器	支持 RTHmi								
例子	命令与输出 >>?HMI_WINDOWSTATE(10) 4								
	读取结果: 4, 表示 10 号窗口为基本窗口								

6.6.21. HMI_MOVEWINDOW -- 移动指定窗口

类型	窗口操作指令							
描述	多动指定窗口。							
	20161112 以后固件版本支持。							
语法	HMI_MOVEWINDOW (winid, posx, posy [, sizex, sizey])							
	winid: 窗口号							
	posx: 水平坐标							
	posy: 垂直坐标							
	sizex: 水平尺寸							
	sizey: 垂直尺寸							
适用控制器	支持 RTHmi							
例子	HMI_MOVEWINDOW (11,100,100) '将 11 号窗口的显示位置改为(100,100)							

6.6.22. HMI_TABLEVALUE -- 设置/获取表格数值

类型	报表视图控件操作指令
描述	设置/获取表格指定项数值
	表格项指定为数值类型时调用该指令设置获取数据。
	获取数值时,多选只会返回选中的第一个单元格数值。
语法	设置/获取表格数值,只写入单个数据
	HMI_TABLEVALUE (winid, controlid, row, col, value)
	value = HMI_TABLEVALUE (winid, controlid, row, col)
	winid: HMI 文件里面窗口编号
	controlid: 元件编号
	row: 表格指定行, -1 表示选中所有行
	col: 表格指定列, -1 表示选中所有列
	value: 表格指定项数值
	多选时,选中的项目都写入相同 value 值。
	获取时,不支持多选。
	批量设置/获取表格数值,从 table 写入或读取
	HMI_TABLEVALUE (winid, controlid, row, col, tabid, ifset)
	tabid: 表格数值保存得 table 起始位置
	Ifset: 是否写入, 0-读取, 1-写入
	注:选中光标行和列同时等于-1时不表示为全选,表示不选中,该指令只能选中
	一个单元格、一行或者一列。行号、列号从0开始编码(表头行/列不计入行/列
	号)!
适用控制器	支持 RTHmi
例子	HMI_TABLEVALUE(16, 1, 2, 1, 500) '将 16 号窗口第 1 个控件的第 3 行 2 列单
	元格的数值设置为 500

Ĺ	运行效果如下:								
>	▶ 在"在线命令"中输入该指令,表格中第3行2列单元格的数值被设为500								
		轴数	速度	加速度	减速度				
	1	0	100	10	10				
	2	1							
	3	2							
	4	3							
		轴数	速度	加速度	减速度				
	1	0	10	10	10				
	2	1							
	3		500						
	4	3							

Zmotion

6.6.23. HMI_TABLETEXT -- 设置/获取表格内容

类型	报表视图	控件操	作指令	>						
描述	设置/获取	表格措	旨定内邻	容(字	符串)					
	表格项指	定为字	符串类	^{美型时}	周用该	指令设	置获明	又数据。		
语法	HMI_TAI	BLETE	XT (wi	inid, co	ontrolid	l, row, o	col, "st	ring")		
	string = H	MI_TA	BLET	EXT (v	winid,	control	id, row	r, col)		
	winid: HMI 文件里面窗口编号									
	contr	controlid: 元件编号								
	row:	表格打	省定行	,-1 表	ē示选 [,]	中所有	行			
	col:	表格指	定列,	-1 表	示选中	「所有多	利			
	string	g: 表格	指定リ		(字符	F串)	केंग्र का वि			
		多选	」时,快	日表種	各分隔	符","分	*割多′	个字符串,使用换行符"CHR(10)"		
	给子	付甲枳	:1丁。 			7/10.1	1 1 11			
	fff ~	火日: ~~ hhh	HMI_	IABLI ミニュと	1EXI +	.(10,1,- 王容口/	·1,-1,"a ∽へ	aa, bbb, ccc + CHR(10) + ddd, eee,		
	111, g 捉加	gg, nnn 下,亜	1, 111 月7 行の소	×小儿' 、畄 긒 ホ	〒104 タ粉捉	」 四日 二 一 空 —	────────────────────────────────────	招针的所有11种列,乃共以直数 为据 第二行6个数据 加下書		
	加如	고 고	11 2 1	-+- ノロか	百刻师	, Yi	11 5 1	数据,第二110 日数据,如于农		
	ТЦ 771	/1/0								
		aaa	bbb	ccc						
		ddd	eee	fff	ggg	hhh	iii			
	注:	指令只	会操作	F选中音	部分。	行号、	列号》	人 0 开始编码(表头行/列不计入		
	行/列号)	!								
适用控制器	支持 RTH	mi								
例子	HMI_TAE	BLETE	XT(16	, 1, 1,	1, "x/t	") '将	F16号	窗口第1个控件的第2行2列单		
	元格的内	容设置	为 x/t							
	运行效果	如下:								
	▶在"在	线命令	"中轴	俞入该	指令,	表格中	「第2彳	亍2列单元格的内容被设为 x/t		



	轴数	速度	加速度	减速度
1	0	100	10	10
2	1			
3	2			
4	3			
	十十 水石			
	40270	18世		
	THEAT	应及	NH KENC	吸应及
1	тшях 0	<u>死度</u> 100	10	10
1 2	о 0 1	100 x/t	10	10
1 2 3	0 1 2	100 x/t	10	10
1 2 3 4	тыях 0 1 2 3	100 x/t	10	10

6.6.24. HMI_TABLECURSOR -- 获取当前选中行列

类型	报表视图控件操作指令								
描述	获取指定报表控件当前选中行列编号,将其保存到 table 表中。								
语法	HMI_TABLECURSOR (winid, controlid, num)								
	winid: HMI 文件里面窗口编号								
	controlid: 元件编号								
	num:当前选中行号、列号分别存储在 table(num), table(num+1)								
	行=-1 表示选中所有行,列 =-1 表示选中所有列。								
	行列同时 =-1 表示未选中。								
	注:行号、列号从0开始编码(表头行/列不计入行/列号)!								
适用控制器	支持 RTHmi								
例子	HMI_TABLECURSOR (16,1,2) '获取 16 号窗口第 1 个控件当前选中的行列编								
	号,将其分别保存到 table(2)和 table(3)中								
	运行效果如下:								
	▶选中单元格后,在"在线命令"中输入"?table(2)"以及"?table(3)", "命令								
	与输出栏"打印当前选中单元格的行列号								
	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1								
	1 0 100 10 10								
	2 1 .								
	3 2 .								
	4 3								
	印マラ期出 × Derm to Controllon Rem Success 2024-11-22 16:20:57 Florest time: 427mg								
	>>?table(2)								
	0 \\9+ah1a(3)								
	1								
	★48会会、2table(3) 安洋 雄蕊 薄線会								

6.6.25. HMI FILESEL	获取当前选中文件/文件夹名称
---------------------	----------------

类型	文件浏览器控件操作指令									
描述	获取文件浏览器控件当前选中文件/文件夹名称(不带路径)									
语法	filename = HMI_FILESEL(winid, controlid [,tabid])									
	winid: HMI 文件里面窗口编号									
	controlid: 元件编号									
	tabid: table 编号, 可	可选, 输出是	否选中文件	夹						
	table(tabid)=0	为文件,=1	为文件夹,	=-1 为未选中						
	filename: 返回当前近	选中的文件/	文件夹名称,	未选中状态返回空						
适用控制器	支持 RTHmi									
例子	?HMI_FILESEL(10,1,0)	'获取并打印	110号窗口第	第1个控件当前选中文	件的名称					
	运行效果如下:									
	▶选中文件后,在"在线	命令"中输	入该指令,'	'命令与输出栏"打印:	选中文件					
	的名称									
	Path: C:/									
	File name	Size	Туре	Modified time						
	—		Folder	2024/11/18 15:36						
	Basic1.bas	2KB	File	2024/11/18 13:42						
	file1 - 副本.z3p	2KB	File	2024/11/18 15:36						
	file1.z3p	161B	File	2024/11/18 14:09						
	file2.nc	2KB	File	2024/11/18 14:11						
	main.dat	772KB 4908	File Fil-	2024/11/21 13:49						
	set.dat 480B File 2024/10/17 08:38 命令与输出 ×									
	Down to Controller Ram Succ	ess, 2024-11-2	28 16:34:45, E1	apsed time: 110ms. 🔺						
	file1.z3p			0						
			4228							
	在线前令: (HIMI_FILESEL(10,1,0)		友法	捕获						

6.6.26. HMI_FILEPATH - 设置/获取当前路径

类型	文件浏览器控件操作指令
描述	设置/获取当前路径(以'/'结尾)
语法	HMI_FILEPATH(winid, controlid, filepath)
	filepath = HMI_FILEPATH(winid, controlid)
	winid: HMI 文件里面窗口编号
	controlid: 元件编号
	filepath: 返回/设置当前路径
	接受盘符"C:/"和"A:/",不输入盘符时默认 C 盘
	C 盘为 flash 目录, A 盘为 U 盘目录
适用控制器	支持 RTHmi



 FILEPATH(10, 运行效果如下: ▶ 当前路径默认 C 盘 	1, "A:/") '设〕 註	<u> 10</u> 号窗口多	第1个控件的当前路径为A
Path: C:/			
File name	Size	Type	Modified time
		Folder	2024/11/18 15:36
Basic1. bas	2KB	File	2024/11/18 13:42
││file1 - 副本.z3p	2KB	File	2024/11/18 15:36
🗋 filel.z3p	161B	File	2024/11/18 14:09
🗋 file2.nc	2KB	File	2024/11/18 14:11
🗋 main. dat	772KB	File	2024/11/21 13:49
🗋 set. dat	480B	File	2024/10/17 08:38
▶在"在线命令"中	中输入该指令后	,当前路径被	支设为 A 盘
Path: A:/			
File name	Size	Type	Modified time
•		Folder	2024/10/11 09:04

6.6.27. HMI_FILEFILTER -- 设置文件过滤器

类型	文件浏览器控件操作指令
描述	设置文件过滤器,过滤显示当前路径所有文件
	注: 文件过滤器只过滤文件,不过滤文件夹
语法	HMI_FILEFILTER(winid, controlid, strfilter)
	strfilter = HMI_FILEFILTER (winid, controlid)
	winid: HMI 文件里面窗口编号
	controlid: 元件编号
	strfilter: 文件过滤器选项,支持通配符'*',以' '隔开多个过滤项
	输入"",表示取消文件过滤器,显示所有文件
	输入"*.*",表示显示所有带后缀的文件
	输入"*.nc *.cnc",表示显示所有 nc、cnc 文件
	注: 过滤器选项不区分大小写
适用控制器	支持 RTHmi
例子	HMI_FILEFILTER(10, 1, "*.bas") '显示 10 号窗口第 1 个控件的所有 bas 文件
	运行效果如下:
	▶ 文件浏览器默认显示 C 盘所有文件

Path: C:/			
File name	Size	Type	Modified time
—		Folder	2024/11/18 15:36
🗍 Basic1. bas	2KB	File	2024/11/18 13:42
││file1 - 副本.z3p	2KB	File	2024/11/18 15:36
🗋 🗍 filel.z3p	161B	File	2024/11/18 14:09
🗋 🗍 file2.nc	2KB	File	2024/11/18 14:11
🗍 main. dat	772KB	File	2024/11/21 13:49
🗋 set. dat	480B	File	2024/10/17 08:38
≻ 在"在线命令"中输 Path: C:/	入该指令后,	文件浏览器	器仅显示 C 盘所有 bas 文作
File name	Size	Type	Modified time
	2KB	Folder File	2024/11/18 15:36 2024/11/18 13:42

Zmotion

6.6.28. HMI_MENUITEM -- 菜单项状态获取/修改

类型	菜单控件操作指令		
描述	获取/修改菜单项状态		
语法	HMI_MENUITEM(id, option, value [,winid, controlid])		
	value = HMI_MENUITEM(id, option [,winid, controlid])		
	id: 菜单项编号(ID)		
	option:数据选项		
	0- 设置菜单项选中(勾选)状态		
	1 – 设置菜单项灰阶(禁用)状态		
	value: 选项值		
	option = 0 时,		
	value=0:将此菜单项设为非选中状态		
	value ≠ 0: 将此菜单项设为选中状态		
	option = 1 bf ,		
	value=0:将此菜单项设为正常状态		
	value ≠ 0: 将此菜单项设为灰阶状态		
	winid: HMI 文件里面窗口编号,缺省当前窗口		
	controlid: 元件编号,缺省当前元件		
适用控制器	支持 RTHmi		
例子	HMI_MENUITEM(1031, 0, 1, 10, 1) '将 10 号窗口第 1 个控件中菜单编号为 1031		
	的项设为选中状态		
	HMI_MENUITEM(1032, 1, 0, 10, 1) '将 10 号窗口第 1 个控件中菜单编号为 1032		
	的项设为灰阶状态		
	运行效果如下:		

▶ 在本菜单中,"汉语"的菜单编号为1031,"英语"的菜单编号为1032
报表 文件 编辑 帮助 语言 > 汉语 使用手册 英语 检查更新 日语 关于 俄语 反馈
▶ 在"在线命令"中输入第一行指令后,菜单栏中"汉语"被勾选
 报表 文件 编辑 帮助 语言 ◇ 汉语 使用手册 英语 注音 关于 反馈 上语 低语 人诺 人第二行指令后,菜单栏中"英语"被禁用
报表 文件 編辑 语言 > ✓ 汉语 使用手册 英语 检查更新 日语 关于 俄语

第七章 DT 运动函数

为了支持 G 代码的变参数个数,增加 DT 运动函数,指令调用 TABLE 表的参数运动。 没有 ABS 后缀的为相对运动指令,带 ABS 后缀的为绝对运动指令。

7.1. MOVEDTSP/MOVEDTABSSP -- DT 直线运动

类型	DT 运动函数
描述	将轴号和运动距离分别存放到 TABLE 表,通过 TABLE 列表进行直线运动。使
	用位存在选择 TABLE 表里的轴号。
语法	MOVEDTSP(最大轴数,位存在,轴 dt 列表起始编号,距离 dt 列表起始编号)
适用控制器	通用
例子	TABLE(10,4,5,6) 'TABLE 表 10 存放 456 三个轴
	TABLE(20,100,50,-10) 'TABLE 表 20 存放三个轴的运距离
	WHILE 1
	IF SCAN_EVENT(IN(0))>0 THEN
	MOVEDTSP(3,5,10,20) '每次运动 TABLE 表中的距离
	'MOVEDTSPABS(3,5,10,20) '运动到 TABLE 表中的位置
	'位存在为5,转换为二进制0101,只选择了轴4和轴6运动
	ENDIF
	WEND

7.2. MOVECIRCDTSP/MOVECIRCDTABSSP -- DT 圆弧运动

类型	DT 运动函数
描述	将轴号和运动距离分别存放到 TABLE 表,通过 TABLE 列表进行圆弧运动。使
	用位存在选择 TABLE 表里的轴号。
	根据当前点(起始点)和中点的位置,以及半径画圆,圆心自动计算。当终点与起
	始点的直线距离大于半径时,以这两点画半圆,半径为连线距离一半,圆心为连
	线中点。
语法	MOVECIRCDTSP(最大轴数,位存在,轴 dt 列表,终点 dt 列表,半径,顺时针
	0/逆时针1)
适用控制器	通用
例子	BASE(0,1,2)
	ATYPE=1,1,1
	DPOS=0,0,0
	TABLE(10,0,1)



7.3. MOVECIRC2DTSP/MOVECIRC2DTABSSP -- DT 三点画圆

弧运动

类型	DT 运动函数
描述	将轴号和运动距离分别存放到 TABLE 表,通过 TABLE 列表进行圆弧运动。使
	用位存在选择 TABLE 表里的轴号。 根据起始点、参考点、终点三点画圆弧。
语法	MOVECIRC2DTSP(最大轴数,位存在,轴dt列表,终点dt列表,参考点dt列
	表, mode)
	mode: <0 参考点在当前点的前面
	=0参考点在中间
	>0参考点在当前结束点的后面
适用控制器	通用

例子	BASE(0,1)		
	ATYPE=1,1		
	DPOS=0,0		
	TABLE(10,0,1) 'TABLE(10)存住	诸轴 0,1	
	TABLE(20,50,10) 'TABLE(20)	存储终点坐标	
	TABLE(30,25,25) 'TABLE(30)	存储参考点坐标	
	TRIGGER		
	WHILE 1		
	IF SCAN_EVENT(IN(0))>07	THEN	
	MOVECIRC2DTSP(3,3,	10,20,30,0)	
	'MODE=0 时,轨迹经过	1参考点,运行到参	学点+终点的坐标位置
	ENDIF		
	WEND		
	1 DPOS(0)	Min:0.00	Max:75.00
	2 DPOS(1)	Min:0.00	Max:35.85
			1

7.4. MSPHERICALDTSP/MSPHERICALDTABSSP -- DT 空间圆

弧运动

类型	DT 运	DT 运动函数		
描述	将轴	将轴号和运动距离分别存放到 TABLE 表,通过 TABLE 列表进行空间圆弧运动。		
	使用作	立存在选持	译 TABLE 表里的轴号。根据起始点、参考点、终点	三点画圆弧。
语法	MSPI	IERICAL	DTSP(轴数,位存在,轴 dt 列表,终点 dt 列表,参	考点 dt 列表,
	mode)		
	r	node:指注	定第二个点的位置	
		值	描述	
		-1	在当前点的前面,只是参考,不运行到参考点	
		0	中间,运行到参考点	
		1	在当前点的和结束点的后面	
适用控制器	通用			
例子	BASE(0,1,2)			
	ATYPE=1,1,1			
	DPOS	5=0,0,0		

TABLE(10,0,1,2)
TABLE(20,0,0,100) 'TABLE(20)终点位置
TABLE(30,30,40,50)
WHILE 1
IF SCAN_EVENT(IN(0))>0 THEN
MSPHERICALDTSP(3,7,10,20,30,0)
'MODE=0 时,轨迹从当前点开始,经过参考点,运行参考点+终点坐标
位置
ENDIF
WEND

第八章 参考例程

8.1. 单轴运动

此例程为单轴运动例程,包含两个文件,Basic 的程序由 Hmi 调用执行。该例程是各元件的综合使用。

【实现目标】:通过Hmi元件实现单轴运动的控制及状态读取。

【设计思路】:

▶ 通过"功能键"调用 Basic 函数控制轴的运动和停止。

▶ 通过多个"值"元件进行数据读取/写入:读取运动状态、写入轴参数。

▶ 通过"位状态切换开关"实现运动模式、运动方向等的切换。

【界面创建】:

创建 Basic 文件和 Hmi 文件,定义 Hmi 需调用的全局 SUB 子函数。根据下图中内容添加 Hmi 组态元件。

文件视图:工程下包含的文件;

过程视图: 各文件内包含的 SUB 子过程;

组态视图:组态窗口和组态元件。(在 Hmi 编程模式下,经常要使用组态视图切换组态编程窗口)

文件视图 ×		标签视图 ×		组态视图 ×		17:Value1(Value) 18:StaticText3(StaticText)		ł
文件视图 ^{文件名} single_move.bas single_move.hmi	× 自动运行 0	× 标签视图 × 组态视图 × 自动运行		 17:Value 1(Value) 18:StaticText3(StaticText) 19:Value 1(Value) 20:StaticText3(StaticText) 21:StaticText3(StaticText) 22:Value 1(Value) 23:StaticText3(StaticText) 24:StaticText24(StaticText) 25:BitSwitch25(BitSwitch) 26:StaticText24(StaticText) 27:BitSwitch25(BitSwitch) 28:Value 1(Value) 29:StaticText24(StaticText) 30:Button30(Button) 				
4						 31:Button30(Button) 32:Button30(Button) 34:StaticText34(StaticText) 35:StaticText34(StaticText) 36:StaticText34(StaticText) 37:StaticText34(StaticText) 40:Rect33(Rect) 41:Rect33(Rect) 42:Rect33(Rect) 43:Rect33(Rect) 	•	

【例程使用】:

(一) Basic 程序界面



Bsing	gle_move.bas × 🖻 single	_move.hmi	•
1 2 3	∃global sub main_int () global str(20) str="木选择"	'ⅢⅢ初始化函数 '轴状态,值显示元件11调用 '值显示元件11显示内容:未选择	
5	global state(20) state="停止"	'运行状态	
8	global axisnum axisnum=0	'轴选择 '1-X轴,2-Y轴	
10	intaxis ()	'初始化轴参数,默认值	
12 13 14 15 16 17 18 19	<pre>dpos=0 units = table(0) lspeed = table(1) speed = table(2) accel = table(3) decel = table(4) sramp = table(5)</pre>	'HMI界面手动设置值,保存在table	
20 21 22 23	table(10)=0 table(11)=0 table(15)=0	' 当前位置,值显示元件1调用 ' 当前速度,值显示元件2调用 ' 寸动距离,值显示元件28调用	
24	RAPIDSTOP (2)		
20	end sub		
20 29 30 31	⊟ global sub m ain_scan() slcaxis()	,HWI周期函数 ,选择轴	*

Basic 程序:

```
global sub main_int()
                   'HMI 初始化函数
 global str(20)
                 '轴状态, 值显示元件 11 调用
                          '值显示元件 11 显示内容:未选择
 str="未选择"
                 '运行状态
 global state(20)
 state="停止"
                 '轴选择
 global axisnum
 axisnum=0
                 '1-X 轴,2-Y 轴
         '初始化轴参数,默认值
 intaxis()
 dpos=0
                'HMI 界面手动设置值,保存在 table
 units = table(0)
 lspeed = table(1)
 speed = table(2)
 accel = table(3)
 decel = table(4)
 sramp = table(5)
```

```
table(10)=0
                   '当前位置, 值显示元件1调用
                   '当前速度,值显示元件2调用
 table(11)=0
                   '寸动距离, 值显示元件 28 调用
 table(15)=0
 RAPIDSTOP(2)
end sub
                     'HMI 周期函数
global sub main_scan()
                 '选择轴
 slcaxis()
 if idle=-1 then
                 '只有在停止状态, 轴参数才生效
   setaxis()
  endif
 table(10)=DPOS
                         '动态获取显示
 table(11)=MSPEED
 if idle=-1 then
   state="停止"
  endif
end sub
                 '轴参数初始化
sub intaxis()
                 'units 脉冲当量
 table(0)=10
                   'Ispeed 起始速度
 table(1)=10
                   'speed 运行速度
 table(2)=100
 table(3)=1000
                   'accel 加速度
                   'decel 减速度
 table(4)=1000
 table(5)=10
                 'sramp s 曲线时间
end sub
                   '轴参数设置
sub setaxis()
 units = table(0)
 lspeed = table(1)
  speed = table(2)
```

```
accel = table(3)
 decel = table(4)
 sramp = table(5)
end sub
global sub slcaxis()
                   '轴选择函数
 if MODBUS BIT(0)=1 then
                          'modbus bit(0)对应 hmi 界面的 X 轴选择按钮
                   '更换选择的轴时,停止Y轴 axis1 的运动
   cancel(2) axis(1)
   str="X轴"
                   '显示内容为:X轴
   axisnum=1
   base(0)
                 '选定 X 轴
 elseif MODBUS BIT(1)=1 then 'modbus bit(1)对应 hmi 界面的 Y 轴选择按钮
                   '更换选择的轴时,停止X轴Axis0的运动
   cancel(2) axis(0)
   str="Y 轴"
                   '显示内容为:Y轴
   axisnum=2
                 '选定Y轴
   base(1)
 endif
end sub
global sub onrun()
                     '运动功能键调用
 if axisnum=0 then
                     'axisnum=0 未选择轴号
   return
 elseif MODBUS_BIT(20)=0 then
                            'modbus bit(20)对应 hmi 界面的运动模式按钮,等于 0 为持续
   if MODBUS BIT(10)=0 then 'modbus bit(10)对应 hmi 界面的方向选择按钮
     vmove(1)
   elseif MODBUS_BIT(10)=1 then
     VMOVE(-1)
   endif
 elseif MODBUS_BIT(20)=1 then '运动模式,等于1为寸动
                       '寸动距离指定, 值显示元件 28
   move(table(15))
 endif
```



```
if idle=0 then
state="运动"
endif
end sub
global sub onstop() '停止功能键调用
state="停止"
RAPIDSTOP(2)
end sub
global sub clear_dpos() '位置清零功能键调用
dpos=0
end sub
```

(二) Hmi 组态界面

- 使用该例程时,先选择要运动的轴号,X轴或Y轴,否则无法运动。再根据需求选择运动方向和 运动模式,若选择运动模式为寸动,还需要设置寸动距离。
- 2. 基本轴参数可自定义设置或采用默认值,调用软键盘窗口自定义输入值,以上设置完成后,可点击运动让轴动起来。运行的速度 SPEED 和轴位置 DPOS 分别被读取到"值"元件1和2动态显示。
- 3. 按下停止按钮立即停止当前运动,按下位置清零按钮清零 DPOS。



8.2. 物理键与虚拟键转换

物理键与虚拟键相关指令只能在自定义元件的刷新函数中使用,参考例程如下。

【实现目标】:将示教盒 ZHD400X 的物理按键绑定自定义虚拟键功能。实现按下物理按键完成对应功能。

【设计思路】:

- ▶ 通过编写 Basic 程序定义虚拟键功能(以打印值功能为例)。
- ▶ 通过"按键转换"功能将物理按键键值与虚拟键键值绑定。
- ▶ HMI 使用"自定义"元件调用 Basic 程序中的刷新函数和绘图函数。

【操作流程】:

(一) Basic 程序定义变量及构建 SUB 子函数

新建项目.zpj 后,新建 Basic 文件,设置自动运行任务号,在程序编辑界面添加如下程序:



Basic 程序:

global dim num	
end	
global sub redraw() '绘图函数	故,自定义虚拟键功能
if num=20 THEN	
print 1	
elseif num=21 THEN	
table(10)=100	
elseif num=22 THEN	
function1()	



endif
end sub
global sub function1()
print num
end sub
global sub refresh()
if VKEY_SCAN<>0 THEN
num=VKEY_SCAN
SET_REDRAW
endif
end sub

(二) Hmi 组态界面

1. 新建 Hmi 文件,设置自动运行任务号,打开 Hmi 窗口。

2. 添加一个"自定义元件",调整其大小置于合适位置,在"属性"中的刷新函数(refresh)和绘图函数(redraw) 分别调用对应的 Basic 子函数。



3. 打开菜单栏"按键转换",按下图将虚拟键 20、21、22 分别绑定物理键 11、12、13,点击"确定"。

注:物理键键值需从《ZHD400X示教盒用户手册》获取。若使用第三方外部设备,则物理按键键值 需从对应设备说明书获取。



			- 预设
物理键	虚拟键	虚拟键描述	(注)(注)(注)(注)(注)(注)(注)(注)(注)(注)(注)(注)(注)(
11	20		用杯
12	21		= 一 号 入
13	22		
			「「「「「「」」「「」」「「」」「」」「「」」「」」「」」「」」「」」「」」「
			确定

【运行效果】:

连接好控制器和示教盒,将程序下载到控制器和示教盒中。

▶ 物理键 11 按下时, 命令行打印 1。

冷令与输出	×
onnected to Controller:VPLC516E Version:5.20-20230717. own to Controller Ram Success, 2023-07-26 17:10:18, Elapsed time: 140ms.	
线命令: 发送 捕获 清除	
命令与输出 查找结果	

▶ 物理键 12 按下时, TABLE(10)赋值为 100。

寄存器		×寄存器	
寄存器名	值	◆ 导入 导出 寄存器名 值	导入 导出
DT(0)	0.000	寄存器类型: DT(0) 0.000 寄存	7 器类型:
DT(1)	0.000	DT(TABLE) T DT(1) 0.000	
DT(2)	0.000	記始编号: DT(2) 0.000 記加	台编号·
DT(3)	0.000	DT(3) 0.000	ana -3 ·
DT(4)	0.000	DT(4) 0.000	(U
DT(5)	0.000	DT(5) 0.000	ų:
DT(6)	0.000	100 DT(6) 0.000	100
DT(7)	0.000	マ 白动場所 DT(7) 0.000 マ	白动刷新
DT(8)	0.000	DT(8) 0.000	
DT(9)	0.000	达m DT(9) 0.000	法即
DT(10)	0.000	DT(10) 100.000	19 4 4X
DT(11)	0.000	DT(11) 0.000	
DT(12)	0.000	DT(12) 0.000	
DT(13)	0.000	DT(13) 0.000	
DT(14)	0.000	DT(14) 0.000	
DT(15)	0.000	▼ DT(15) 0.000 ▼	

> 物理键 13 按下时,调用函数 function1,函数功能可以自定义(此处以打印 num 值为例)。



命令与转	输出
22	
在线命令:	发送 捕获 清除
命令与输出	查找结果

8.3. 动态列表

此例程为列表的使用示例,参考例程如下:

【实现目标】: 使用 HMI_LISTTEXTS 指令配合"列表"元件实现动态列表控制效果。即通过一个列表的选择控制另一个列表的表项显示,进而可省略由于选项众多而建立多个列表。

【设计思路】:

- ▶ 在 Basic 文件中使用 HMI_LISTTEXTS 指令编写程序并定义全局 SUB 子函数。
- 添加多个"列表"元件,并将其绑定各个寄存器。(建议不同列表绑定不同的寄存器,以免产生影响冲突)
- ▶ 将第一个"列表"元件调用 Basic 子函数,实现动态控制。

【操作流程】:

(一) Basic 程序定义变量及构建 SUB 子函数

新建项目.zpj 后,新建 Basic 文件,设置自动运行任务号,在程序编辑界面添加如下程序:

Basic 程序界面:

Basic1.bas × ⊮Hmi1.hmi ▼
1 global dim g.Curselltem '定义列表表成英型」の-列表页、1-数字、2-送页、3-英文 2 global dim g.Surselt1(128) '定义空和教理。
5 · · · · · 列表初始 0 · □gelobal sub sub_SwitchList0
7 - HIL_LISTTEXTS(10,1,"列表项(n数字(n选项(n选项(n英文")))) / 对控件1设置4个选项的列表文本
9
11 af g_CurSellten = 0 then 选择央型0-列表项 12
13 IIII_LISTTEXTS(10,2, *列表项1\n列表项2\n列表项3\n列表项3\n列表项5\n列表项5\n列表项6\n列表项7\n列表项7\n列表项7\n列表项1\n列表项1\n列表项11\n列表项12*) * 对理任设置列表文本 14
15 IEILLISTTEXTS(10,3, "列表项1\n列表项2\n列表项3\n列表项4\n列表项5\n列表项6\n列表项7\n列表项7\n列表项6\n列表项9\n列表项1\n列表项1\n列表项11\n列表项12">> ,对控件10设置列表文本 16
17 日 H■I_LISTTEXTS(10,4, "列表项1\n列表项2\n列表项3\n列表项4\n列表项5\n列表项6\n列表项7\n列表项7\n列表项6\n列表项9\n列表项6\n列表项1\n列表项11\n列表项12") ,对控件12设置列表文本 18 ト
19 elseif g_CurSelItem = 1 then '选择类型数字 20
21 HII_LISTTEXTS(10,2, *111111/n222222/n333333/n444444/n5555555/n6666666/n777777/n888888/n999999/n000000*)
33 HHI_LISTTEXTS(10,3, *111111\n222222\n333333\n444444\n555555\n6666666\n777777\n888888\n99999\n000000*)
25 HHI_LLSTTEXTS(10, 4, *111111\n222222\n333333\n444444\n555555\n6666666\n777777\n888888\n99999\n000000*)
27 elseif g_CurSelItem = 2 then '选择类型2-选项
800 HⅢI_LISTT#XTS(10,2,"选项1\n选项2\n选项3\n选项5\n选项6\n选项7\n选项8\n选项7)
00 31 H■I_LISTT#XTS(10,3, "选项1\n选项2\n选项3\n选项4\n选项5\n选项6\n选项7\n选项8\n选项9")
333 H∎I_LISTT#XTS(10,4,"选项1\n选项2\n选项3\n选项4\n选项5\n选项6\n选项7\n选项8\n选项9")
335 else / 选择类型3-英文
37 BULLISTEXTS(10,2, "one\n two\n three\n four\n five\n six\n seven\n eight\n nine")
38 BHILLISTEXTS(10,3, "one\ntwo\nthree\nfour\nfive\nsix\nseven\neight\nnine")
40 41 HHILLISTTEXTS(10, 4, "one\ntwo\nthree\nfour\nfive\nsix\nseven\neight\nnine")
42 endif 43
46 47
48 endsub 49

Basic 程序:

global dim g_CurSelItem '定义列表选项类型, 0-列表项、1-数字、2-选项、3-英文 global dim g_SHowStr1(128) '定义字符数组

"'列表切换

global sub sub_SwitchList()

if g CurSelItem = 0 then '选择类型 0-列表项

HMI_LISTTEXTS(10,2, "列表项 1\n 列表项 2\n 列表项 3\n 列表项 4\n 列表项 5\n 列表项 6\n 列表项 7\n 列表项 8\n 列表项 9\n 列表项 10\n 列表项 11\n 列表项 12") '对控件 2 设置列表文本

HMI_LISTTEXTS(10,3, "列表项 1\n 列表项 2\n 列表项 3\n 列表项 4\n 列表项 5\n 列表项 6\n 列表项 7\n 列表项 8\n 列表项 9\n 列表项 10\n 列表项 11\n 列表项 12") '对控件 3 设置列表文本

HMI_LISTTEXTS(10,4, "列表项 1\n 列表项 2\n 列表项 3\n 列表项 4\n 列表项 5\n 列表项 6\n 列表项 7\n 列表项 8\n 列表项 9\n 列表项 10\n 列表项 11\n 列表项 12") '对控件 4 设置列表文本

elseif g CurSelItem = 1 then '选择类型 1-数字

HMI LISTTEXTS(10,2,

 $"111111\n222222\n33333\n444444\n555555\n6666666\n777777\n8888888\n999999\n000000")$

HMI_LISTTEXTS(10,3,

 $"111111\n222222\n33333\n444444\n555555\n6666666\n777777\n888888\n999999\n000000")$

HMI LISTTEXTS(10,4,

 $"111111\n222222\n33333\n444444\n555555\n6666666\n777777\n8888888\n999999\n000000")$

elseif g_CurSelItem = 2 then '选择类型 2-选项

HMI_LISTTEXTS(10,2, "选项 1\n 选项 2\n 选项 3\n 选项 4\n 选项 5\n 选项 6\n 选项 7\n 选项 8\n 选项 9")

HMI_LISTTEXTS(10,3, "选项 1\n 选项 2\n 选项 3\n 选项 4\n 选项 5\n 选项 6\n 选项 7\n 选项 8\n

Zmotion[®] _{正运动技术}

选项 9")

HMI_LISTTEXTS(10,4, "选项 1\n 选项 2\n 选项 3\n 选项 4\n 选项 5\n 选项 6\n 选项 7\n 选项 8\n 选项 9")

else '选择类型 3-英文

HMI_LISTTEXTS(10,2, "one\ntwo\nthree\nfour\nfive\nsix\nseven\neight\nnine")

HMI_LISTTEXTS(10,3, "one\ntwo\nthree\nfour\nfive\nsix\nseven\neight\nnine")

HMI_LISTTEXTS(10,4, "one\ntwo\nthree\nfour\nfive\nsix\nseven\neight\nnine") endif

g_SHowStr1="触发选中调用函数,选中表项"g_CurSelItem+1 '控件5显示寄存器存储的字符 串

endsub

(二) Hmi 组态页面

1. 新建一个 Hmi 文件,设置自动运行任务号。

2. 打开 Hmi 窗口,选择"列表"元件,放置 4 个列表(列表数量视具体需求而定),对每个列表的属性进行设置,例如调整"列表方式": True——显示全部(注意调整元件高度),Flase——下拉/上拉列表。可通过"弹出方式"调整向上弹出或向下弹出。

3. 对第一个列表选择绑定的寄存器变量,并调用 Basic 子函数实现列表的动态控制,另外三个列表绑定 的寄存器变量编号保证不相同即可。

4. 最后可根据需求建立"文本"元件添加说明。

注意:使用 HMI LISTTEXTS 指令,"列表"元件需将"属性"中的"动态项"选择为 True。





Basic1.bas ⊡Hmi1.hmi ×	属性	▲ 1 ×
10:Start	▲ 基本属性	A
7 Staticlest T1 +	元件编号	1
切念列表	元件名称	List1
·	显示层次	底层
11.StatisText	有效显示	显示
根据左边列表所选内容,进行表项选择	采用有效控制	False
\$-Cenerations 50-Cenerations	绑定的设备编号	本地
点击表项,触发调用函数 显示所有列表项 · 拉列表	绑定的寄存器类型	@
I:List 2:List A:List	绑定的@寄存器	g_CurSelltem
	选中调用函数	sub SwitchList
	动态项	True
	⊿ 外观	
	列表方式	True
	弹出方式	向下弹出
	图片来源	无
	绘制边框	True
5:String_	边框颜色	000000
ABC		
	抽金粉 邦助 屏性	

【运行效果】:

xplc so	creen							×
				动态	态列表	R.		
			根据左	边列表所选口	内容,进行衰	長项选择		
点击	击表项,	触发调用函数	显示所有	列表项	ţ	动利表	上拉列表	
		列表项 数字	选项1 选项2 选项3		选项2	▼		
		选项	<mark>选项4</mark> 选项5					
		英文	选项6	-			选项8 ▲	
[触发选□	中调用函数,道	也中表项3				
						-		

8.4. 视图缩放

此例程为滑块的使用示例,参考例程如下:

【实现目标】:通过"滑块开关"元件实现"自定义"元件中图形的大小缩放效果。(以矩形为例) 【设计思路】:

▶ 通过编写 Basic 程序定义矩形变量,以及刷新函数和绘图函数。

(一) 定义参数变量,设置初始值。

- 1. 定义滑块的全局缩放变量(g_zoom),设置初始值。
- 2. 定义绘制图形的坐标数组,对各组 XY 坐标赋初始值。
- (二) 定义全局刷新函数

给"自定义元件"调用刷新函数(reflesh)。

(三) 定义全局绘图函数

该例程主要以倍数关系对图形实现缩小和放大。

- 1. 设置局部变量参数(zoom)和 g_zoom 得到其倍数关系。
- 设置绘制图形的局部变量 XY 坐标,根据缩放量与 XY 坐标数组的关系,得到最终的 XY 坐标。
- 3. 写入需要绘制的图形指令。
- ▶ 添加"自定义"元件,并调用 Basic 定义的刷新函数和绘图函数
- ▶ 添加"滑块"元件并绑定寄存器(g_zoom)

【操作流程】:

(一) Basic 程序定义变量及构建 SUB 子函数

新建项目.zpj 后,新建 Basic 文件,设置自动运行任务号,在程序编辑界面添加如下程序:

Basic 程序界面:

Basic1.bas × Hmi1.hmi	
1 2 GLOBAL DIN g_zoom '缩放 3 GLOBAL DIN g_pointx(2) 'x坐标集合 4 GLOBAL DIN g_pointy(2) 'y坐标集合 5 GLOBAL DIN g_round '倒角半径	
7 '参数初始化 8 g_zoom=100 9 g_pointx(0)=20 10 g_pointy(1)=100 12 g_pointy(1)=60 13 g_round=10	
14 15 end 16	
17 □GLOBAL sub reflesh() '刷新函数 18 · SET_REDRA¥ 19 end sub	
21 GLOBAL sub redraw() '绘图函数 22 LOCAL zoom, pointx, pointy 23 zoom=g_zoom/100.0 24 SETEX_LINE(2,0) '设置线条类型和线条宽度 25 SET_COLOR(RGB(255,0,0), RGB(255,255,0)) 26 pointx=g_pointx(0)+g_pointx(1)*zoom 27 pointy=g_pointy(0)+g_pointy(1)*zoom 28 DRAVEX_RECT(g_pointx(0), g_pointy(0), pointx, pointy, g_round*zoom, 1) 30 DRAVEX_RECT(g_pointx(0), g_pointy(0), pointx, pointy, g_round*zoom, 0) 31 end sub	

Basic 程序:

GLOBAL DIM g_zoom '缩放



GLOBAL DIM g pointx(2) 'x 坐标集合 GLOBAL DIM g pointy(2) 'y 坐标集合 '倒角半径 GLOBAL DIM g round '参数初始化 g_zoom=100 $g_pointx(0)=20$ g pointy(0)=20 g pointx(1)=100 g_pointy(1)=60 g round=10 end GLOBAL sub reflesh() '刷新函数 SET REDRAW end sub GLOBAL sub redraw() '绘图函数 LOCAL zoom,pointx,pointy $zoom=g_zoom/100.0$ SETEX_LINE(2,0) '设置线条类型和线条宽度 SET COLOR(RGB(255,0,0),RGB(255,255,0)) pointx=g pointx(0)+g pointx(1)*zoom pointy=g_pointy(0)+g_pointy(1)*zoom DRAWEX_RECT(g_pointx(0),g_pointy(0),pointx,pointy,g_round*zoom,1) DRAWEX_RECT(g_pointx(0),g_pointy(0),pointx,pointy,g_round*zoom,0) end sub

(二) Hmi 组态页面:

在自定义元件框中,采用指令进行矩形绘制,通过滑块可调节矩形放大或缩小。

1. 新建 Hmi 文件,设置自动运行任务号,打开 Hmi 窗口。

2. 添加一个"自定义元件",调整其大小置于合适位置,在"属性"中的刷新函数和绘图函数分别调用对应的 Basic 子函数(reflesh 和 redraw)。

3. 绘制一个"矩形"元件,大小与"自定义元件"一致,移动到和"自定义元件"位置重合。

4. 绘制"滑块"元件,在"属性"中调整样式、大小,显示数值范围等,刻度数可通过"属性"的主刻度数和 副刻度数进行设置(下图主刻度数为5,副刻度数为1),并放置于合适位置(建议放置在"自定义元件"外), 同时绑定寄存器变量。如下图所示。



【运行效果】:

运行后,由于设置的初始值为100,滑块默认处于中间位置,即100的刻度处,图形初始大小如图所示。



screen			

▶ 滑块拖动到 150 后,实现的放大效果如图所示。

xplc screen					>
0	50 	100	150	200	

▶ 滑块拖动到 50 后,实现的缩小效果如图所示。



8.5. 滚动条使用

此例程为滚动条的使用示例,参考例程如下:

【实现目标】: 使用滚动条指令结合"自定义"元件和"值"元件实现自定义滚动条及行号刷新。

【设计思路】:

▶ 在 Basic 文件中使用滚动条指令编写程序并定义全局 SUB 子函数。

滚动条的设置

- 1、滚动条初始化:利用 <u>SCROLLBAR_INIT</u>指令确定滚动条的编号和放置的位置,设置滚动条的 宽度和高度、颜色,以及总行数和显示行数。
- 2、滚动条的刷新:利用 <u>SCROLLBAR_RELASH</u>指令实现对应编号滚动条的刷新,并对滚动条变 换位置的时候实现刷新重绘。
- 3、滚动条的绘制:利用 SCROLLBAR DRAW 指令绘制滚动条。
- 添加"自定义"元件,并分别调用刷新函数和绘图函数。
- ▶ 添加多个"值"元件,绑定寄存器变量,每行绑定的寄存器变量数值需加一。
- ▶ 添加"定时器"元件,实现定时刷新效果。

【应用场景关联】

在文件管理、写多行文本或自定义绘图视图缩放等场景时,都可添加滚动条使得有更多的管理空间。 (本例子主要以多行文本的场景进行滚动条演示。可通过行号进行观察变化。)

【操作流程】:

(一) Basic 程序定义变量及构建 SUB 子函数

新建项目.zpj 后,新建 Basic 文件,设置自动运行任务号,在程序编辑界面添加如下程序:




Basic 程序:

global DIM Linenum	'首行行号
global DIM ScrollValue	'滚动条滚动值
global DIM maxlines	'滚动条总行数
global DIM showlines	'滚动条显示行数
maxlines=50 '总行数为 50	
showlines=10 '显示行数为	10
sub_ScrollBar_Init() '滚动条	初始化
end	
'定时器刷新数据,每秒刷新,	处理不需要频繁刷新的变量以及状态
global sub Timer_RefreshData()
dim timercnt	
Linenum = ScrollValue+ 1	'行号随滚动值变化
end sub	
global sub correct()	
TOUCH ADJUST()	
end sub	
'====滚动条刷新	Я
global sub sub_ScrollBar_reflas	sh() '在自定义元件 2 中的刷新函数进行调用
local value, bIfRedraw 5	定义变量用于判断是否重绘
bIfRedraw = SCROLLBAR_	REFLASH(0) '根据返回值手动判断是否重绘
if bIfRedraw > 0 then	
value = SCROLLBAR_PC	OS(0) '获取滚动条 0 的当前位置滚动值
if ScrollValue <> value the	m '判断当前的滚动值是否一致
ScrollValue = value	
endif	

Zmotion[®]

SET_REDRAW '重新绘制
endif
end sub
'======滚动条绘制====================================
global sub sub_ScrollBar_redraw() '在自定义元件 2 中的绘图函数进行调用
SCROLLBAR_DRAW(0) '绘制滚动条
end sub
'======滚动条初始化==========
global sub_ScrollBar_Init()
SCROLLBAR_INIT(0, 378, 0, 22, 346, maxlines, showlines) '设置滚动条 0 的宽度为 22, 高度为
350, 总行数为50, 显示行数为10。注意: 滚动条坐标位置不可超出自定义元件大小范围。
end sub
(二) Hmi 组态页面:

在自定义元件中绘制滚动条,并采用"值"元件来显示行号。使用"定时器"对行号进行刷新。

1. 新建 Hmi 文件和 Basic 文件,分别设置自动运行任务号,打开 Hmi 窗口;

2. 添加一个"自定义",调整其大小置于合适位置(本例程设置元件尺寸为:400*350),在"属性"中分别调用对应的刷新函数(例程中的:sub_ScrollBar_reflash())和绘图函数(例程中的:sub_ScrollBar_redraw())。

3. 添加一个"矩形"元件用作边框,调整其大小与"自定义元件"相同,放置与"自定义元件"重合的位置。 (目的是便于确定无边框自定义元件的位置)

4. 添加多个"值"元件,数量根据实际情况而定(本例程添加 10 个),调整其大小并放置于合适位置。 在"属性"中将各个"值"元件绑定到对应寄存器变量,各"值显示"元件绑定的寄存器变量逐个往下加1(如第 一个元件的寄存器变量+0,第二个元件的寄存器变量+1,第三个寄存器变量+2,以此类推),再根据需求 调整其他属性内容。如下图。





5. 添加一个"定时器"元件,放置于合适位置,适当调整"属性"中的"时间 ms",值越小定时器的触发周 期越短。在"属性"的"动作"中选择 call sub,在"动作函数名"处调用 Basic 定时器刷新子函数。



▲运行双禾】: 随着滚动条的滚动,行号会发生变化。





8.6. CAD 功能

本章节仅介绍元件的简单使用例程,CAD、File3 等完整功能 Demo 可联系正运动技术工程师获取。

8.6.1.CAD 导入矢量图片

此例程为使用 CAD 元件操控矢量图片的使用示例。

【实现目标】: 使用 CAD 指令结合"CAD"元件和"功能键"元件实现矢量图片插入和关闭的效果。

【设计思路】:

➢ 在 Basic 文件中使用 CAD_LOADFILE 和 CAD_CLOSE 指令等编写程序并定义全局 SUB 子函数。

(一) 定义导入 CAD 图像函数

"功能键"元件一调用动作函数,函数体中使用 CAD_LOADFILE 指令导入 CAD 图像。

(二) 定义关闭 CAD 图像函数

"功能键"元件二调用动作函数,函数体中使用 CAD CLOSE 指令关闭 CAD 图像。

- ▶ 添加"CAD"元件,设置通道号、图层颜色等。
- ▶ 添加"功能键"元件并在属性"动作"中调用对应函数。

【操作流程】:

(一) Basic 程序定义变量及构建 SUB 子函数

新建项目.zpj 后,新建 Basic 文件,设置自动运行任务号,在程序编辑界面添加如下程序:



语法	CAD_LOADFILE(dxfname [, refdis[, layers]])
	dxfname = CAD_LOADFILE ()
	dxfname: dxf文件名,带路径
	refdis: 导入复杂曲线分割精度,范围[0.01~0.5]
	该分割精度决定导入复杂曲线时是否自动分割为线段,refdis缺省或=0时,不自动分割曲线,反之则根据refdis精度进行曲线分割
	layers:选择加载图层,bit0~bit31位有效,支持最大32个图层
	缺省加载所有图层(加载G代码文件时无效)

Basic 程序:

'打开矢量图形	
global sub Btn_LoadCadFile()	
dim loadfile(128)	
loadfile = "CADTest/1.dxf"	'保存矢量图片的路径
?"加载图形",loadfile	
CAD_LOADFILE(loadfile,0)	
end sub	

'关闭矢量图形
global sub CloseCad()
CAD_CLOSE()
end sub

(二) Hmi 组态界面

1. 新建 Hmi 文件,设置 Hmi 文件自动运行任务号,打开 Hmi 窗口。

2. 在控制器或者仿真器的 flash 目录下,插入一个带有矢量图片的文件夹。

3. 添加一个"CAD"元件,调整其大小并置于合适位置。

4. 在 Basic 文件中使用 CAD_LOADFILE 编写插入矢量图片功能的代码,设置矢量图片带路径的文件 名、导入复杂曲线分割精度以及选择加载图层。

5. 在 Hmi 文件中添加一个"功能键"元件,调整其大小置于合适位置。格式文本设置为"导入图形",动 作设置为"调用函数",动作函数名选择第四步中设计的函数名。

6. 在 Basic 文件中使用 CAD CLOSE 编写关闭矢量图片功能的代码。

7. 在 Hmi 文件中添加一个"功能键"元件,调整其大小置于合适位置。格式文本设置为"关闭图形",动 作设置为"调用函数",动作函数名选择第六步中设计的函数名。



【运行效果】:





▶ 点击"导入图形"功能键后,CAD元件区域显示导入的CAD矢量图片。此时将鼠标停至CAD元件 区域使用滚轮该CAD图形会相应的缩放,点击CAD区域画面拖动时CAD图形位置会改变,此时 画面偏移。

xplc screen			×
	导入图形	关闭图形	
ſ			

8.6.2.CAD 结合三次文件使用

此例程为 CAD 元件结合三次文件元件的使用示例。

【实现目标】: 在上节例程的基础上将 CAD 图形文件的 Basic 代码和 G 代码导入三次文件编辑器元件中,并在界面上展示。

【设计思路】:

- ➢ 在 Basic 文件中使用 CAD_TOCODE 和 FILE3_BUFLOAD 指令等编写程序并定义全局 SUB 子函数。
 - (一) 定义导出 CAD 图像 Basic 代码的 SUB 子函数
 - 1. 定义数组,用于存储图片路径。
 - 2. 函数体中使用 CAD_TOCODE 指令, CAD_TOCODE 中 type=0 为导出 Basic 类型代码。
 - 3. 使用 FILE3_BUFLOAD 指令将三次文件加载出来。

(二) 定义导出 CAD 图像 G 代码的 SUB 子函数

- 1. 定义数组,用于存储图片路径。
- 2. 函数体中使用 CAD TOCODE 指令, CAD TOCODE 中 type=1 为导出 G 代码。
- 3. 使用 FILE3_BUFLOAD 指令将三次文件加载出来。
- ▶ 添加"三次文件编辑器"元件,设置通道号、可编辑等属性。
- ▶ 添加"功能键"元件并在属性"动作"中调用对应函数。

【操作流程】:

(一) Basic 程序定义变量及构建 SUB 子函数

```
新建项目.zpj 后,新建 Basic 文件,在程序编辑界面添加如下程序:
```

B Ba	asic1.l	oas × ⊫Hmi1.hmi		*
1 2 3	⊟gloł	oal sub Btn_LoadCadFile()		1
4 5 6 7 8		din loadfile(128) loadfile="CADTest/1.nc" ?"加载图片",loadfile C AD_LOADFILE (loadfile,0,1)		
9	end	sub		
11 12 13	ूCLOE end	BAL SUB closecad() CAD_CLOSE() sub		
14 15 16 17	'导出 □glot	46代码 al sub Btn_ExportGCode () dim FileName(128)		
18 19 20		''文件名称 FileName = ″CADTest/1.nc″		
◆22 23 24		CAD_TOCODE (FileName, 1, -1) ?"导出G代码 <i>"</i>	'导出代码文件(NC/Z3P)	
25		FILE3_BUFLOAD(0, FileName)		
20	ends	sub		
28 29 30	! · 导出	Basic pal sub Btn ExportBasic ()		
31		din FileName(128)		
33 34		''文件名称 FileName = ″CADTest/1.Z3P″	'导出代码文件(NC/Z3P)	Ŧ



语法	FILE3_BUFLOAD(mode, filename)
	mode:加载模式,0-仅加载已存在的文件,1-新建空白文件再加载
	2-清空文件再加载(若文件不存在,等同模式1)
	filename: 3 次文件名

语法	CAD_TOCODE(filename, type, layers)
	filename: 导出代码文件名
	xxx.Z3P输出basic的Z3P文件
	xxx.nc输出G代码的NC文件
	当filename= ""时,直接输出到当前3次文件加载区
	通过FILE3_CHANNEL指令指定输出到哪个三次文件通道
	type: 导出代码类型,0-Z3P(Basic),1-NC(G代码)
	layers: 导出图层, bit0~bit31位有效, -1表示导出所有图层
	如layers=1+2+4+8,表示导出图层0、1、2、3

Basic 程序:

```
'加载图形文件
global sub Btn_LoadCadFile()
dim loadfile(128)
loadfile = "CADTest/1.dxf"
?"加载图形",loadfile
CAD_LOADFILE(loadfile, 0,1) '从文件中载入图片
end sub
'关闭图形
global sub CloseCad()
CAD_CLOSE() '关闭 CAD 加载图形
end sub
global sub Btn_ExportGCode()
dim FileName(128)
```

Zmotion[®]

```
"文件名称
 FileName = "CADTest/1.nc"
 CAD TOCODE(FileName, 1, -1)
                          '导出代码文件(NC)
 ?"导出 G 代码 "
 FILE3 BUFLOAD(0, FileName)
                          '加载指定3次文件到内存中.0表示仅加载已存在的文件
endsub
'导出 Basic
global sub Btn ExportBasic()
 dim FileName(128)
 "文件名称
 FileName = "CADTest/1.Z3P"
 CAD TOCODE(FileName, 0, -1) '导出代码文件(Z3P)
 ?"导出 Basic"
 FILE3 BUFLOAD(0, FileName)
                           '加载指定3次文件到内存中,0表示仅加载已存在的文件
endsub
```

(二) Hmi 组态页面

1. 在上一例程——【CAD 导入矢量图片】程序的基础上添加一个"三次文件编辑器"元件,调整其大小并置于合适位置。"三次文件编辑器"元件属性中"垂直滚动条"和"水平滚动条"都需要设置为 True 才可滑动 显示完整的 Basic 代码和 G 代码。

2. 在 Basic 文件中使用 CAD_TOCODE 指令编写导出 Basic 类型代码功能的程序,设置导出代码文件 名、导出类型且 type=0 以及导出图层。

3. 在 Hmi 文件中添加一个"功能键"元件,调整其大小置于合适位置。格式文本设置为"导出 Basic",动作设置为"调用函数",动作函数名选择第二步中设计的函数名,此例程调用的是 Btn ExportBasic 函数。

4. 在 Basic 文件中使用 CAD_TOCODE 指令编写导出 G 代码功能的程序,设置导出代码文件名、导出 类型且 type=1 以及导出图层。

5. 在 Hmi 文件中添加一个"功能键"元件,调整其大小置于合适位置。格式文本设置为"导出 G 代码",动作设置为"调用函数",动作函数名选择第四步中设计的函数名,此例程调用的是 Btn_ExportGCode 函数。

【运行效果】:



xplc screen					×
导入图形	关闭图形	导出Bas	ric	导出	G代码
×		unloaded.	Row 1	Col 1	Lns 0
	-				
					-
		4			

▶ 点击"导入图形"后,"CAD"元件显示对应矢量图片。再点击"导出 Basic"或"导出 G 代码"元件时三次文件编辑器元件中会显示相应的 Basic 代码和 G 代码。

导出 Basic 效果图:

xplc screen		×
导入图形	导出Basic	出G代码
	CADTest/1.Z3P Row 1 Col 1 1BASE(0,1,2) 2BASE_MOVE=0 3MOVE_PARA(SPEED, 0, 500.0000) 4MOVE_PARA(ACCEL, 0, 1000.0000) 5MOVE_PARA(DECEL, 0, 1000.0000) 6MOVEABS(402.8137, 129.1925) 7MOVE_PARA(SPEED, 0, 1000.0000) 8MOVE_PARA(ACCEL, 0, 1000.0000) 9MOVE_PARA(DECEL, 0, 1000.0000) 10MOVEABS(1.0000) AXIS(2) 11MOVEABS(402.8137, 129.4925)	Lns 35



导出 G 代码效果图:

xplc screen						Х
导入图形	关闭图形	导出Ba	asic	导出	<mark>G代码</mark>	
		CADTest/1.nc	Row 1	Co1 1	Lns 16	
		1G17 G90				
		2G00 X0.000	YO. 000 Z10. 00	000		
		3GUU X402.8.	137 Y129.1925			
		5G01 X402 8	137 Y129 4925			
		6G00 X404.8	137 Y129. 1925			
		7G01 Z1.000	0 F1000.0000			
		8G01 X404.8	137 ¥129.4925			
		9G01 X402.8	137 Y129.4925			
		10G00 X404.8	137 Y129. 1925			
		11601 21.000	0 F1000.0000			-
		•			•	

8.6.3.CAD 结合自定义元件使用

此例程为 CAD 元件结合自定义元件使用的示例。

【实现目标】:使用 CAD_TOTABLE、CAD_LOADTABLE 和 CAD_DRAW 等指令结合"CAD 元件" 实现在"自定义"元件中绘制出 CAD 图形。

【设计思路】:

➤ 在 Basic 文件中使用 CAD_LOADTABLE 和 CAD_DRAW 指令等编写程序并定义全局 SUB 子函数。

使用 CAD_LOADFILE 指令<u>导入 CAD 图形</u>,使用 CAD_TOTABLE 指令<u>导出到 TABLE</u> 后<u>关闭</u> <u>CAD</u>,再使用 CAD_LOADTABLE 指令<u>从 TABLE 中加载图形</u>。

(一) 定义绘图函数

给"自定义"元件中的绘图函数调用。使用 CAD_DRAW 指令绘制 CAD 图形。

(二) 定义刷新函数

给"自定义"元件中的刷新函数调用。函数体中使用 CAD_SETCOLOR 等指令在自定义元件内刷新 CAD 图形颜色、显示方式等。

- ▶ 添加"自定义"元件,属性中"刷新函数"、"绘图函数"调用 Basic 对应的 SUB 函数。
- ▶ 添加"CAD"元件,并设置通道号、图层颜色等。

【操作流程】:

(一) Basic 程序定义变量及构建 SUB 子函数

新建项目.zpj 后,新建 Basic 文件,在程序编辑界面添加如下程序:

B Ba	sic1.bas × 🖻 Hmi1.hmi		•
1 2 3 4 5 6 7	CAD_LOADFILE("CADTest/3D-玫瑰花.dxf",0 CAD_TOTABLE(10000, -1) CAD_CLOSE() CAD_LOADTABLE(10000) end) '从文件中载入图片 '导出TABLE数据,导出到table(10000)之后 '关闭CAD加载图形 '从TABLE中载入图形,从TABLE(10000)开始加载图形	
9 10 11 12 13	''绘图函数 □GLOBAL SUB drawing1() CAD_DRAW(0, 0, 1) '绘制CAD图形		
14 15 16 17	END SUB ''刷新函数 GLOBAL SUB refresh1()		
18 19 20 21 22	CAD_TOUSE (2) CAD_SETCOLOR (0, RGB (255, 0, 255)) CAD_DI SPLAY (3)	'设置鼠标模式 '设置CAD绘制颜色 '设置CAD显示方式	
23	END SUB		-

语法	CAD_TOTABLE(datatabid, layers) datatabid: table地址,输出table数据 table存储数据格式如下:
	详看TABLE数据格式设计小节。 layers: 导出图层,bit0~bit31位有效,-1表示导出所有图层 如layers=1+2+4+8,表示导出图层0、1、2、3
语法	CAD_LOADTABLE (tabid)

tabid: table下标

语法	CAD_DRAW (offsetX, offsetY, fZoom [.plane, RotX, RotY, RotZ])
	offset1, offset2:显示图形偏移,=(0,0)时,图形中心处于自定义控件中心位置
	fZoom:显示缩放比例,1.0=自定义控件范围刚好完整显示加载图形
	以下属性为预留三维CAD图形:
	plane:显示平面,缺省0-XY,1-YZ,2-ZX
	RotX,RotY,RotZ: 绕XYZ轴旋转角度,0~360,缺省0。



语法	CAD_ SELECT (ix1, iy1, ix2, iy2, offsetX, offsetY, fZoom[,plane, RotX, RotY, RotZ])
	ix1,iy1,ix2,iy2:坐标的,选择框两个对顶点坐标(两点成矩形)
	offsetX, offsetY:显示图形偏移,=(0,0)时,图形中心处于自定义控件中心位置
	fZoom:显示缩放比例,1.0=自定义控件范围刚好完整显示加载图形
	以下属性为预留三维CAD图形:
	plane:显示平面,缺省0-XY,1-YZ,2-ZX
	RotX,RotY,RotZ:绕XYZ轴旋转角度,0~360,缺省0。

Basic 程序:

CAD_LOADFILE("CADTest/3	D-玫瑰花.dxf',0) '从文件中载入图片
CAD_TOTABLE(10000, -1)	'导出 TABLE 数据,导出到 table(10000)之后
CAD_CLOSE()	'关闭 CAD 加载图形
CAD_LOADTABLE(10000)	'从 TABLE 中载入图形,从 TABLE(10000)开始加载图形
end	
"绘图函数	
GLOBAL SUB drawing1()	
CAD_DRAW(0, 0, 1)	'绘制 CAD 图形
END SUB	
"刷新函数	
GLOBAL SUB refresh1()	
CAD_MOUSE(2)	'设置鼠标模式
CAD_SETCOLOR(0,RGB(2	55,0,255)) '设置 CAD 绘制颜色
CAD_DISPLAY(3)	'设置 CAD 显示方式
END SUB	

(二) Hmi 组态页面

1. 新建 Hmi 和 Basic 文件,设置 Hmi 文件自动运行任务号,打开 Hmi 窗口。

2. 在控制器或者仿真器的 flash 目录下,插入一个带有矢量图片的文件夹。

3. 添加一个"CAD"元件,调整其大小并置于合适的位置。添加一个"自定义"元件,调整其大小并置于合适的位置。

4. 在 Basic 文件中编写绘图函数和刷新函数。本例程使用的绘图函数为 drawing1, 使用的刷新函数为

refresh1 °

5. 在自定义元件中调用第四步中编写的绘图函数和刷新函数。



【运行效果】:



8.7. 文件浏览器使用

此例程为文件浏览器的使用示例,包含两个文件,Hmi 文件通过 Basic 文件打开,Basic 文件的其他函数由Hmi 调用执行。该例程是多个元件的综合使用。

【实现目标】:使用 HMI_FILEFILTER、HMI_FILESEL、HMI_FILEPATH、HMI_CONTROLATTR 等 指令结合"文件浏览器"元件、"列表"元件、"字符显示"元件、"功能键"元件、"三次文件编辑器" 元件以及"树形图"等元件实现各种类型文件的显示、打开、选中与修改。

【设计思路】:

- ➤ 在 Basic 文件中使用 HMI_FILEFILTER、HMI_FILESEL、HMI_FILEPATH 指令等编写程序并定 义全局 SUB 子函数。
- ▶ 使用"文件浏览器"元件调用 Basic 全局 SUB 函数实现文件类型的过滤与文件的选择。
- 使用"列表"元件绑定与文件类型选择相关的寄存器,设置"选中调用函数"以及需显示的"状态数量",填入对应列表文本。
- ▶ 使用"字符显示"元件绑定与文件名显示相关的寄存器。
- ▶ 使用"功能键"元件调用 Basic 全局 SUB 函数打开对应文件。
- ▶ 使用 FILE3_BUFLOAD 指令将选择的三次文件加载出来。
- ▶ 添加"三次文件编辑器"元件,设置通道号、可编辑等属性。
- ▶ 添加"树形图"元件,设置绑定的寄存器类型及编号,接着在"列表文本"内设计树节点结构并编写其文本内容。根据树节点对应的 ID 编号在 Basic 文件中结合"HMI_CONTROLATTR"指令编写全局 SUB 函数,然后在树形图元件属性框中调用,实现三次文件编辑器"选中多行"的功能。

【操作流程】:

(一) Basic 程序启动 Hmi 程序、定义变量及构建 SUB 子函数

新建项目.zpj 后,新建 Basic 文件,设置自动运行任务号,在程序编辑界面添加如下程序:

```
🖻 Basic1.bas 🛛 🗵 Hmi1.hmi
                                                                                                                                 *
          ''全局变量
                                                                                                                                 ۰
          global din g_iCurSelFileFilter
global din g_iCurSelFileName(128)
                                                                ''文件过滤器选择
''当前选择文件名
   2
3
   4
          ''启动HMI任务
   5
          RUN "Hmil.hmi", 1
   6
   7
          end
   8
   9
          ''文件浏览器 过滤器选择
 10

      □global sub Sub_FileFiterSel()

      □ if g_iCurSelFileFilter = 0 then

      ``m

      □ HTI_FILEFILTER(10, 1, "*.nc|*.cnc")

 11
                                                                       ''显示NC文件
 12
 13
                                                                            ,
显示Bas文件
                elseif g_iCurSelFileFilter = 1 then
 14
                elseif g_iCurSelFileFilter = 2 then
HII_FILEFILTER(10, 1, "*.bas")
elseif g_iCurSelFileFilter = 2 then
HII_FILEFILTER(10, 1, "*.z3p")
 15
                                                                             ''显示Z3P文件
 16
 17
        þ
                elseif g_iCurSelFileFilter = 3 then
    HHI_FILEFILTER(10, 1, "*.*")
                                                                             ''显示所有文件
 18
 19
                endif
 20
 21
          end sub
 22
          ''文件浏览器 单击选择文件
 23
 24
       □global sub Sub_FileSelect()
 25
                local strCurFileName(128)
```



```
      语法
      FILE3_BUFLOAD(mode, filename)

      mode:
      加载模式,0-仅加载已存在的文件,1-新建空白文件再加载

      2-清空文件再加载(若文件不存在,等同模式1)

      filename:
      3 次文件名
```

Basic 程序:

"全局变量
global dim g_iCurSelFileFilter "文件过滤器选择
global dim g_iCurSelFileName(128) "当前选择文件名
"启动 HMI 任务
RUN "Hmi1.hmi", 1
end
"文件浏览器 过滤器选择
global sub Sub_FileFiterSel()
if g_iCurSelFileFilter = 0 then "显示 NC 文件
HMI_FILEFILTER(10, 1, "*.nc *.cnc")
elseif g_iCurSelFileFilter = 1 then
HMI_FILEFILTER(10, 1, "*.bas")
elseif g_iCurSelFileFilter = 2 then
HMI_FILEFILTER(10, 1, "*.z3p")
elseif g_iCurSelFileFilter = 3 then
HMI_FILEFILTER(10, 1, "*.*")
endif
end sub
"文件浏览器 单击选择文件
global sub Sub_FileSelect()
local strCurFileName(128)
strCurFileName = HMI_FILESEL(10, 1, 0)
if table(0) = 0 then "选中了文件
g_iCurSelFileName = strCurFileName
endif
end sub
"文件浏览器 双击文件
global sub_FileDlbClick()



```
local strCurFileName(128)
 strCurFileName = HMI FILESEL(10, 1, 0)
 if table(0) = 0 then "选中了文件,直接打开
   g iCurSelFileName = strCurFileName
   Sub FileOpen()
 endif
end sub
"文件浏览器 打开文件
global sub Sub FileOpen()
 HMI SHOWWINDOW(11,6)
 local strFilePathName(512)
 "根目录为" C:/"时,不补"/"
 if strFilePathName(2) = 47 then
   strFilePathName = HMI FILEPATH(10, 1) + g iCurSelFileName
 else
   strFilePathName = HMI_FILEPATH(10, 1) + "/" + g_iCurSelFileName
 endif
 "打开三次文件
 FILE3 BUFLOAD(0, strFilePathName)
 ?"打开三次文件", strFilePathName
end sub
"树形图 结合"HMI CONTROLATTR"指令实现三次文件编辑器"选中多行"功能
global sub Sub Tree()
 if TABLE(1000) = 1001 THEN
   HMI CONTROLATTR("MSELROWS",-5,11,1) "往上选中5行
 elseif TABLE(1000) = 1002 THEN
   HMI CONTROLATTR("MSELROWS",-10,11,1) "往上选中 10 行
 elseif TABLE(1000) = 1004 THEN
   HMI CONTROLATTR("MSELROWS",5,11,1) "往下选中5行
 elseif TABLE(1000) = 1005 THEN
   HMI CONTROLATTR("MSELROWS",10,11,1) "往下选中 10 行
 elseif TABLE(1000) = 1006 THEN
   HMI CONTROLATTR("MSELROWS",0,11,1) "选中当前行不变
  endif
```

end sub

(二) Hmi 组态页面

1. 在控制器或者仿真器的 flash 目录下,插入一些 basic、z3p、nc 等类型的文件夹。

= flash		× +					- 0	×
\leftarrow \rightarrow	\uparrow	C 🖬 > …	RTSys >	RTSys	> zsimu > flash	在flas	h 中搜索	
+ 新建 ~	*	() (<u>)</u>	ē Ū	∩↓	排序 ◇ 🛛 📄 査看 ◇		🕕 详细信	恴
		名称	^		修改日期	类型	大小	
三 桌面	*	🗋 Basic1.bas			2024/11/18 13:42	BAS 文件	2 KB	
业 下载	I	📄 file1.z3p			2024/11/18 14:09	Z3P 文件	1 KB	
■ 文档	*	🗋 file2.nc			2024/11/18 14:11	NC 文件	2 KB	
■ 図片		🗋 main.dat			2024/11/18 14:07	媒体文件(.dat)	776 KB	
	*	🗋 set.dat			2024/10/17 8:38	媒体文件(.dat)	1 KB	
- 🕖 音乐	*							
5 个项目								

2. 新建 Hmi 文件, 打开 Hmi 基本窗口 10。

3. 在基本窗口 10 添加一个"文件浏览器"元件,选择"双击调用函数"和 "选中调用函数",合理按 需设置其他属性。

🖲 Basic1.bas 🖪 H	mi1.hmi	×		-	属性
Þatile Brower					▲ 基本属性
File name	Size	Туре	Modified time		元件编号 1
<mark></mark>		Folder	2024/9/4 8:00		元件名称 File Brower2
Folder1		Folder	2024/9/4 9:00		显示层次 底层
Folder2	26VD	Folder	2024/9/4 10:00		有效显示 显示
File2. nc	50KB 512B	File	2024/9/4 11:00		采用有效控制 False
	0122		1011, 0, 1 11100		双击调用函数 Sub_FileDlbClick
					选中调用函数 Sub_FileSelect
					▲ 外观
					绘制边框 False
					图片来源 无
					显示路径 True
					显示标题 True
▲			•		标题高度 30
					文件名列宽度 200
					显示文件大小 True
					文件大小列宽度 100
					显示文件类型True
					双击调用函数
					双击时调用的函数。
				•	轴参数 属性 帮助

4. 在"文件浏览器"元件下方添加一个"静态文本"元件、一个"字符显示"元件、一个"列表"元件 以及两个"功能键"元件,调整其大小并置于合适位置。

(1)将"静态文本"元件的格式文本设为"文本名:"。

(2) "字符显示"元件绑定"g_iCurSelFileName"寄存器,用于显示选中文件的名称。

(3)"列表"元件选中调用函数"Sub_FileFiterSel",弹出方式设置为"向下弹出",绑定 "g_iCurSelFileFilter"寄存器,状态数量设为4,每个状态对应的列表文本设置如下:



(4) 一个"功能键"元件调用"Sub_FileOpen"函数,实现打开三次文件的功能;另一个"功能键" 元件的动作选择"关闭指定窗口",实现关闭三次文件的功能。

5. 基本窗口 10 整体界面如下图:

Þáth: BC.7					
File name	Size	Туре	Modified time		-
<mark>-</mark>		Folder	2024/9/4 8:00		
📒 Folder1		Folder	2024/9/4 9:00		
Folder2		Folder	2024/9/4 10:00		
🗋 Filel.z3p	36KB	File	2024/9/4 11:00		
🗋 File2.nc	512B	File	2024/9/4 12:30		
					-
4					•
^{2:StatecLext} 文件名: ABC				NC ⁱ File(*.nc)	•
				5:Button 6:But	ton
				打开	天闭

6. 新建弹出窗口 11, 添加一个"三次文件编辑器"元件,设置其通道号、是否可编辑、外观等相关属性。使用 FILE3_BUFLOAD 指令加载指定三次文件到内存中显示。

🛚 Basic1.bas 🗎 Hm	i1.hmi ×			*	属性	Ŧ	ά×
ABC.nc	Row 1	Co1 1	Lns 11111		有效显示	显示	-
1G01 X0.000 Y0.000 Z0	0.000		-		采用有效控制	False	
2G01 X0.000 Y0.000 Z0	0.000				通道号	0	
3G01 X0.000 Y0.000 Z0	0.000				可编辑	True	
4G01 X0.000 Y0.000 Z0	0.000				安全时间ms	0	
6G01 X0 000 Y0 000 Z0	0.000				修改时通知	False	
7 GO1 X0. 000 Y0. 000 Z0	0.000				▲ 外观		
8G01 X0.000 Y0.000 Z0	0.000				绘制边框	False	
9G01 X0.000 Y0.000 Z0	0.000			k 1	图片来源	无	
10G01 X0.000 Y0.000 Z0	0.000				行间距	2	11
12601 X0.000 Y0.000 Z0	0.000				行号	True	
13G01 X0.000 Y0.000 Z0	0.000				行号宽度	30	
14G01 X0.000 Y0.000 Z0	0.000				文件信息	True	
15G01 X0.000 Y0.000 Z0	0.000				垂直滚动条	True	
16G01 X0.000 Y0.000 Z0	0.000				水平滚动条	True	
17G01 X0.000 Y0.000 Z0	0.000		-	-	▲ 标签		•
19			•				•
				•	轴参数 属性 帮助		

7. 在 11 窗口添加一个"树形图"元件,设置绑定的寄存器类型及编号,接着在 "列表文本"内设计树 节点结构并编写其文本内容。

Basic1.bas	🖻 Hmi1.hmi	×			• J	属性	
ABEide3Edit	Row 1	Col 1	Lns 11111	2 . Titæm1		绑定的设备编号	本地 🔺
1 GO1 X0.000 Y0.000 2 GO1 X0.000 Y0.000	Z0.000 Z0.000		<u>^</u>	Item1.1 Item1.2		绑定的寄存器类型	DT
3 G01 X0.000 Y0.000	20.000			Item2		绑定的寄存器编号	1000
5 G01 X0.000 Y0.000	20.000					选中调用函数	
6 G01 X0.000 Y0.000 7 G01 X0.000 Y0.000 :	Z0.000 Z0.000					▲ 外观	
8 G01 X0.000 Y0.000 3	20.000 20.000					绘制边框	True
10 GO1 X0. 000 Y0. 000	Z0. 000					边框颜色	000000
11 G01 X0. 000 Y0. 000 12 G01 X0. 000 Y0. 000 1	20.000 20.000					图片来源	无
13 G01 X0.000 Y0.000 14 G01 X0.000 Y0.000	20.000 20.000					角标类型	样式1
15 GO1 X0.000 Y0.000	Z0. 000					角标颜色	000000
17 G01 X0. 000 Y0. 000	20.000					行间距	2
18 G01 X0. 000 Y0. 000 19 G01 X0. 000 Y0. 000	20.000 20.000					选中部分颜色	99CCFF
20 GO1 X0.000 Y0.000	20.000					▲ 标签	
20			•			文本库	
						格式文本	Item
						列表文本	ltem
					1	限定的寄存器编号	
					\$	曲参数 属性 帮助	

根据树节点对应的 ID 编号在 Basic 文件中结合"HMI_CONTROLATTR"指令编写 Sub_Tree 函数, 然 后在树形图元件属性框"选中调用函数"中调用该函数,实现三次文件编辑器"选中多行"的功能。

树形图设计	×
▲ 向上 ▲ 向上5行 ↓ □ [请在此处双击键入]	选中顶编号: 1000
 ▲ - 向上10行 □ [请在此处双击键入] □ [请在此处双击键入] 	自动编号
	插入
└ Li青在此处汉无键入」 - 向下10行<br └ Li青在此处双击键入]	刪除
└ [请在此处双击键入] / 不变 └ [请在此处双击键入] [请在此处双击键入]	
	确定
	取消

代码段如下图所示:

''树形图 结合 "HMI_CONTROLATTR" 指令实现三次文件编辑器 "选中多行" 功能
□global sub Sub_Tree()
if TABLE (1000) = 1001 THEN
└── HⅢI_CONTROLATTR(″MSELROWS″,-5,11,1) ''往上选中5行
elseif TABLE(1000) = 1002 THEN
HⅢI_CONTROLATTR("MSELROWS",-10,11,1) '' 往上选中10行
elseif TABLE(1000) = 1004 THEN
└ HⅢI_CONTROLATTR(″MSELROWS″,5,11,1) ''往下选中5行
elseif TABLE(1000) = 1005 THEN
├ HⅢI_CONTROLATTR("MSELROWS",10,11,1) ''往下选中10行
elseif TABLE(1000) = 1006 THEN
└──
endif
end sub



8. 基本窗口 11 整体界面如下图:

ABC ¹ SEdi	t			Row 1	Col 1	Lns 11111	² √"向上
1G01	X0. 000	YO. 000	Z0. 000				向上5行
2G01	X0. 000	YO. 000	Z0. 000				向上10行
3G01	X0. 000	YO. 000	ZO. 000				▼ 向下
4G01	X0.000	YO. 000	ZO. 000				向下5行
5G01	X0. 000	YO. 000	ZO. 000				向下10行
6G01	X0. 000	YO. 000	Z0. 000				不变
7G01	X0.000	YO. 000	ZO. 000				
8G01	X0. 000	YO. 000	ZO. 000				
9G01	X0. 000	YO. 000	ZO. 000				
10G01	X0. 000	YO. 000	Z0. 000				
11G01	X0. 000	YO. 000	ZO. 000				
12G01	X0.000	YO. 000	ZO. 000				
13G01	X0. 000	YO. 000	ZO. 000				
14G01	X0. 000	YO. 000	ZO. 000				
15G01	X0. 000	YO. 000	ZO. 000				
16G01	X0. 000	YO. 000	ZO. 000				
17G01	X0.000	YO. 000	ZO. 000				
18G01	X0.000	YO. 000	ZO. 000				
19601	VO 000	V0 000	70 000				
20							

【运行效果】:

xplc screen					×
Path: C:/					
File name	Size	Туре	Modified time		-
<mark> </mark>		Folder	2024/11/18 14:11		
Basic1.bas	2KB	File	2024/11/18 13:42		
🗋 filel.z3p	161B	File	2024/11/18 14:09		
file2.nc	2KB	File	2024/11/18 14:11		
🗌 main. dat	776KB	File	2024/11/18 14:07		
set. dat	480B	File	2024/10/17 08:38		
4)
文件名:				A11 Files(*.*)	•
					۲J



▶ 点击"列表"元件,选择需要显示的文件类型后,"文件浏览器"元件内显示相应文件类型的文件。

xplc screen					×
Path: C:/					
File name	Size	Туре	Modified time		-
—		Folder	2024/11/18 14:11		
Basic1. bas	2KB	File	2024/11/18 13:42		
					-
<u>ا</u>					
文件名:				Bas File(*.bas)	-
				Bas File(*.bas)	
				Z3P File(*.z3p)	
				A11 Files(*.*)	

▶ 选中文件后,"字符显示"元件将显示该文件的文件名,点击"打开"或双击文件打开。

xplc screen					×
Path: C:/					
File name	Size	Type	Modified time		-
<mark>.</mark>		Folder	2024/11/18 14:11		
Basic1.bas	2KB	File	2024/11/18 13:42		
					-
•					•
文件名: Basic1.bas				Bas File(*.bas)	-
				打开 关	闭



▶ 弹出"三次文件编辑器"界面,显示文件内容,可以对文本内容进行编辑。



选中任意一行后,展开树形图的所有树节点。接着,点击"向上10行"按钮,如下二图所示,系 统将自动从选中的那一行开始,向上额外选中10行数据,共选中11行。





xplc screen					×
C://Basic1.bas	Row 16	Co1 20	Lns 58	▼ 向上	
1''全局变量				向上5行	
2global dim g iCurS	elFileFilter ''文件	+过滤器选择		向上10行	
3global dim g iCurS	e1Fi1eName(128) ''늴	前选择文件名		> 向下	
4				不变	
5''启动HMI任务					
6RUN "Hmi1.hmi", 1					
7 end					
8					
9					
10''文件浏览器 过滤器	8选择				
11global sub Sub_File	eFiterSel()				
12 if g_iCurSelFileF	ilter = O then ''5	l示NC文件			
13 HMI_FILEFILTER(1)	0, 2, "*.nc *.cnc")				
14 elseif g_iCurSelF	ileFilter = 1 then	′′显示Bas文件			
15 HMI_FILEFILTER(1	0, 2, "*.bas")				
16 elseif g_iCurSelF:	ileFilter = 2 then	~显示Z3P文件			
17 HMI_FILEFILTER(1)	0, 2, ~*.z3p~)				
4					
				17升 关闭	ſJ

▶ 点击"关闭"按钮,文件内容被关闭,界面回退到文件选择界面。

xplc screen						×
Path: C:/						
File name		Size	Туре	Modified time		-
—			Folder	2024/11/18 14:11		
Basic1. b	bas	2KB	File	2024/11/18 13:42		
						-
4						•
文件名:	Basic1.bas				Bas File(*.bas)	•
					打开	关闭

8.8. 例程下载

更多触摸屏应用例程请前往正运动官方网站下载,网址:<u>www.zmotion.com.cn</u>。 下载路径: 官网首页→支持与服务→下载中心→例程资料→触摸屏程序。

附录

RTHmi 版本功能介绍

不同的 RTHmi 版本可支持的 Hmi 功能有所不同,详见下表。

查询 RTHmi 版本的方法: RTSys 软件连接控制器→打开"控制器状态"→ "RTHmiVersion: 10300(20241008)"。数值表示 V1.3.0 版本, 括号中数值表示日期。

RTHmi 版本号	功能更新描述
V1.0.00	支持基础控件及功能、指令。
(ZDevelop 支持)	
	1. 新增滑块开关、表格、多行文本、三次文件编辑、中文输入 法、CAD 控件;
	2. 新增 DRAWEX 绘图指令集、滚动条绘图指令、CAD 基础指
V1.1.00	令集等;
	3. 升级所有绘图控件;
	4. 升级列表控件,支持滚动条显示,支持动态列表,支持选中
	函数调用、下拉函数调用,美化外观;
	5. 背景图片、图片库支持 32 位图片带透明度显示;
	1. 新增支持无损压缩图片、防失真缩放显示等;
V1 2 00	2. 新增 SDF 抗锯齿绘图,通过 HMI_DEFAULTATTR 指令开
(DTSug 吉持 ZDavalan 不吉持	关;
这版本及后续版本的新增功能)	3. DRAWEX 指令新增透明度显示,通过 SETEX_ALPHA 设
以版本及加续版本的新增功能/	置;
	4. 优化控件、zft字体等内容;
	1. 新增支持扫码枪等快速输入设备批量输入字符不丢失字符;
V1.2.01	2. 新增了 CAD 自定义扩展指令集和高级指令集;
	3. 支持多个刷新 dirty,刷新 Dirty 新增至 4个;
	1. 文本库新增支持语言代码页,优化 HMI 对外国语言的支持;
	2. 新增控件垄断功能,支持下拉列表仅垄断鼠标,不垄断按键;
V1.2.02	字符显示控件增加直接输入中文方式,无需弹出键盘;
	3.指 令 VKEY_MODE 、 HMI_LISTITEM 、
	HMI_CONTROLATTR、HMI_DEALINFO 新增参数;
	1. HMI_DEALINFO 指令增加参数 EDITNEXT、EDITPREV 获
V1 2 02	取下一个、上一个编辑控件;
v 1.2.05	2. HMI_CONTROLATTR 指令增加参数 FOCUS 设置、获取控
	件焦点状态;

Zmotion[®]

	3. 滑块控件新增支持小数滑动(最小刻度设置为0时生效);
	4. 下拉列表控件中心支持选中背景高亮;
	1. 支持文本库各国语言都可以直接使用 65001 的代码页。
V1.2.04	2. HMICONTROLATTR 指令的"FOCUS"参数支持设置 0 关闭
	指定控件编辑焦点状态。
	1. 文本库新增"以语言方式显示"或"以状态方式显示"功能;
	2. Hmi 设置: 基本属性新增"不使用文本库格式文本"功能,"水
	平分辨率"和"垂直分辨率"可编辑功能;
	3. 窗口导入新增窗口对比功能;
	4. 控件箱元件新增报表视图、文件浏览器、菜单、树形图;
V1 2 0	5. 新增操作指令: HMI_TABLEVALUE、HMI_TABLETEXT、
V1.5.0	HMI_TABLECURSOR、HMI_FILESEL、HMI_FILEPATH、
	HMI_FILEFILTER、HMI_MENUITEM;
	6. SETEX_ALPHA 指令优化;
	7. HMI_DEFAULTATTR、HMI_CONTROLATTR 指令新增一个
	参数;
	8. 新增文件浏览器使用参考例程章节。

虚拟键值说明

虚拟键	键值	功能描述
VKEY_0~VKEY9	48~57	输入数字或字符"0~9"
VKEY_PLUS	43	输入字符"+"
VKEY_POINT	46	输入字符"."或小数点
VKEY_MINUS	45	输入字符"-"或当前数值符号取反
VKEY_ENTER	10	输入确定,并切换编辑焦点
VKEY_CLR	13	清空当前输入
VKEY_SPACE	32	输入空格字符""
VKEY_TAB	9	切换当前编辑焦点
VKEY_BackSpace	8	退格,从光标处删除一个字符
VKEY_DEL	127	标准删除
VKEY_ESC	27	标准退出,并关闭虚拟键模式
VKEY_SHIFT	171	shift 按键
VKEY_MENU	172	暂不可用
VKEY_CONTROL	173	暂不可用
VKEY_PAGE	174	翻页,暂不可用
VKEY_SWITCH	175	暂不可用
VKEY_INS	176	插入
VKEY_CAPS	177	切换大小写, 输入复合按键时有效
VKEY_EN_CH	178	切换中英文输入法
VKEY_SYMBOL	179	符号(预留)
ZKEY_F1	128	快捷按键 F1
ZKEY_F2	129	快捷按键 F2
ZKEY_F3	130	快捷按键 F3
ZKEY_F4	131	快捷按键 F4
ZKEY_F5	132	快捷按键 F5
ZKEY_F6	133	快捷按键 F6
ZKEY_F7	134	快捷按键 F7
ZKEY_F8	135	快捷按键 F8
ZKEY_START	140	启动按键
ZKEY_STOP	141	停止按键
VKEY_LEFT	145	方向按键,向左移动

VKEY_RIGHT	146	方向按键,向右移动
VKEY_UP	147	方向按键,向上移动
VKEY_DOWN	148	方向按键,向下移动
ZKEY_1LEFT	150	JOG 按键
ZKEY_1RIGHT	151	JOG 按键
ZKEY_2LEFT	152	JOG 按键
ZKEY_2RIGHT	153	JOG 按键
ZKEY_3LEFT	154	JOG 按键
ZKEY_3RIGHT	155	JOG 按键
ZKEY_4LEFT	156	JOG 按键
ZKEY_4RIGHT	157	JOG 按键
ZKEY_5LEFT	158	JOG 按键
ZKEY_5RIGHT	159	JOG 按键
ZKEY_6LEFT	160	JOG 按键
ZKEY_6RIGHT	161	JOG 按键
VKEY_1_STAR	201	复合按键,1*
VKEY_2_ABC	202	复合按键, 2abc 或 2ABC, 通过 VKEY_CAPS 切换
VKEY_3_DEF	203	复合按键, 3def 或 3DEF
VKEY_4_GHI	204	复合按键,4ghi或4GHI
VKEY_5_JKL	205	复合按键, 5jkl 或 5JKL
VKEY_6_MNO	206	复合按键,6mno或6MNO
VKEY_7_PQRS	207	复合按键, 7pqrs 或 7PQRS
VKEY_8_TUV	208	复合按键, 8tuv 或 8TUV
VKEY_9_WXYZ	209	复合按键,9wxyz或9WXYZ
VKEY_POINT_HASH	210	复合按键,.#
VKEY_A~VKEY_Z	65~90	输入大写字母"A~Z"
VKEY_a~VKEY_z	97~122	输入小写字母"a~z"
其他	0~127	作为 ASCII 码输入

Zmotion[®]

错误码列表

错误码	意义	可能原因
5000	LCD 号错误	
5001	Hmi 文件错误	窗口号相同



5002	LCD 号冲突
5003	不支持对象
5004	内存不够
5005	控件层次错误
5006	窗口号超过
5007	无效窗口号
5008	HMI 文件内容错误
5009	窗口号重复
5010	对象属性丢失
5011	输入窗口有多个显示元件
5012	ACTION 类型错误
5013	事件过多
5014	返回上个窗口失败
5015	不能关闭基本窗口
5016	字体中找不到对应字符
5017	必须在 HMI 任务中使用
5018	HMI 中对应的控件类型不支持这个指令
5020	控件 ID 冲突
5021	LCD 号错误
5022	找不到可用 LCD
5023	LCD 没有打开
5024	LCD 无数据
5025	程序复位
5026	LCD 已经打开了
5027	不是网络 LCD
5028	不支持的压缩方式
5029	颜色深度不支持
5030	不支持的数据类型
5031	设备号错误
5032	LCD_SEL 不能使用
5033	设置 REDRAW 不能再 DRAW 阶段
5034	DRAW 函数只能在 DRAW 阶段
5035	操作不能再 DRAW 阶段调用



5036	内部 LCD 分辨率固定	
5037	LCD 分辨率超过	
5038	库文件名错误	
5039	字符过多	
5040	元件属性丢失	
5041	没有输入显示元件,无法输入	
5042	状态数过多	
5043	不支持的绘图属性	
5044	远程通讯设备名称错误	
5045	远程通讯数据没有更新	
5101	无效的日期格式	
5102	控件不存在	
5103	多边形点数太少或太多	
5104	无空闲可用的滚动条	
5105	无效的滚动条 ID	
5106	尚不支持功能	
5107	未加载图形	
5108	文件已损坏	
5109	菜单参数错误	
5110	导出 table 空间不足,溢出	
5111	不支持的数据类型	
5112	指令不支持的控件类型	
5113	数组溢出	传入数组下标超过最大值
5114	内部错误	
5115	通道溢出	传入通道号超过最大值