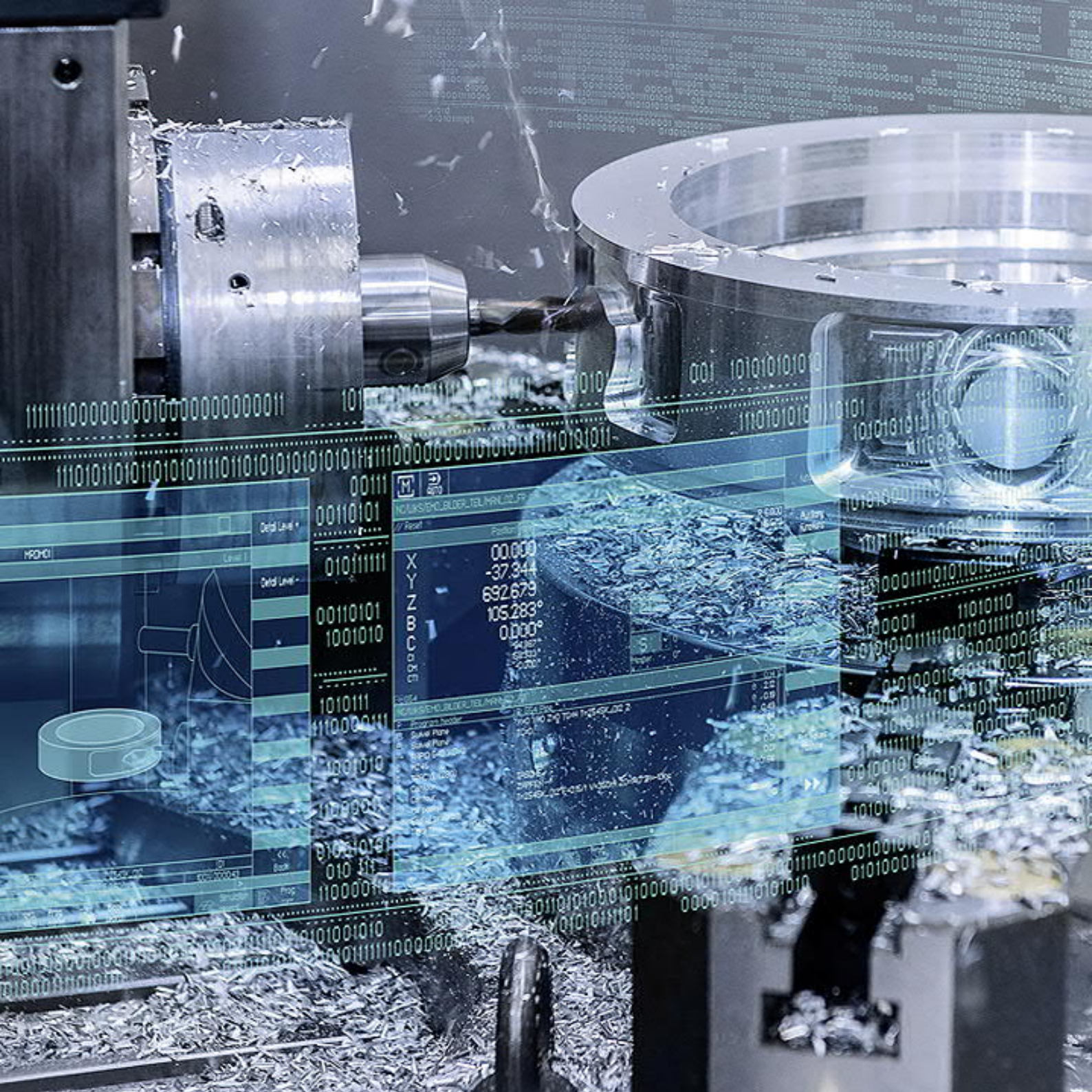




V1.0

SLC DI MC MTS APC

RUN MYSCREENS 应用手册

SINUMERIK 828D & 840DSL

HMI OA 应用样例包

目录

1	免责声明	0
2	概述	1
3	要求	2
3.1	软硬件及选项要求	2
3.2	其他要求	2
3.3	句法说明	2
3.4	设计文档的结构	2
3.4.1	文档的优先级	3
3.4.2	文档的保存路径	4
4	界面结构	5
4.1	区域划分	5
4.2	操作树	5
4.3	界面组成	5
5	对话框	7
5.1	概述	7
5.2	句法	7
5.3	参数说明	8
5.4	可编辑的对话框属性	9
6	变量	10
6.1	概述	10
6.2	句法	10
6.3	参数说明	10
6.4	其他说明	13
6.4.1	变量值 (val)	13
6.4.2	变量状态 (vld)	14

目录

6.4.3	辅助变量	14
6.4.4	变量计算	14
6.4.5	系统变量间接编译地址	14
6.4.6	更改变量属性	14
7	按键	15
7.1	水平/竖直软键	15
7.1.1	定义软键栏	15
7.1.2	软键句法	15
7.1.3	可更改的软键属性	17
7.1.4	系统预定义按键（水平/竖直软键栏）	17
8	触摸屏-按钮	19
8.1	概述	19
8.2	要求	19
8.3	按钮定义	19
8.4	按钮属性	20
8.4.1	示例	21
8.4.2	读按钮属性	22
8.4.3	写按钮属性	22
8.5	按钮动作	22
8.5.1	点击按钮（click）	23
8.5.2	转换按钮（checked）	23
8.5.3	不可操作按钮（clickedDisabled）	24
8.6	示例	24
8.6.1	示例 1：按下按钮跳转至其他界面	24
8.6.2	示例 2：显示文本来自语言文件	24
9	初始化文件	25
9.1	easyscreen.ini	25

目录

9.1.1	标签[STARTFILES]	25
9.1.2	使用模板文件	25
9.1.3	入口关联的.com 文件	26
9.1.4	操作树	26
9.1.5	空余按键	27
10	方法	28
10.1	ACCESSLEVEL (访问等级)	28
10.2	CHANGE (变量值改变)	28
10.3	CHANNEL (通道切换)	29
10.4	CONTROL (多个 NCU 切换时)	29
10.5	FOCUS (焦点定位)	29
10.6	LANGUAGE (语言切换)	30
10.7	LOAD (装载对话框)	30
10.8	UNLOAD (卸载对话框)	30
10.9	OUTPUT (输出加工程序代码)	31
10.10	PRESS (按下软键)	31
10.11	RESOLUTION (更改分辨率)	32
10.12	SUSPEND (切换区域中断时)	32
10.13	RESUME (切换区域恢复时)	32
11	功能	33
11.1	读写驱动,NC,PLC	33
11.2	块,数组,表格定义	33
11.3	程序文件	34
11.4	文件读写	34
11.5	列表框操作	34
11.6	口令功能	35
11.7	装载功能	35

目录

11.8 字符串功能	35
11.9 逻辑功能	36
11.10 绘图功能	37
11.11 其他功能	37
11.12 参考手册	37
12 附录	38
12.1 附录 1：easyscreen.ini 模板	38
12.2 附录 2：颜色表	43
12.3 附录 3：运算符	44
12.4 附录 4：预定义按键	46
13 作者/联系人	47

1 免责声明

本使用手册及样例包目录内所包含文档、PLC 程序、机床可执行程序（MPF、SPF、...）、电气图，可能与用户实际使用不同，用户可能需要先对例子程序做修改和调整，才能将其用于测试。本例程的作者和拥有者对于该例程的功能性和兼容性不负任何责任，使用该例程的风险完全由用户自行承担。由于它是免费的，所以不提供任何担保，错误纠正和热线支持，用户不必为此联系西门子技术支持与服务部门。

对于在使用中发生的人员、财产损失本公司不承担任何责任，由使用者自行承担风险。

以上声明内容的最终解释权归西门子（中国）有限公司所有，后续内容更新不做另行通知。

2 概述

“Run MyScreens”界面设计开发功能在早期的 SINUMERIK 软件版本中称为 “EasyScreen” 界面开发功能，此二者在句法上相同，只是随着软件版本的升级，名称也有更新。本书所涉略到 “EasyScreen” 的名词将统一由 “Run MyScreens” 替代。

Run MyScreens 界面开发具有如下特点：

- 代码量小，编程简单，门槛低，上手快
- 基于系统定义好的框架和固定句法进行编程
- 可以运用界面底层的代码和算法实现用户工艺应用的需求
- 通过 .com 编译器和 .ini 的初始化文件来实现界面开发
- 支持嵌入系统界面按键下的界面开发
- 支持结合用户循环，生成加工程序的界面开发
- 可自定义界面窗口的大小

开发工具：UTF8 编辑器，常用软件有

- notepad++
- ultraedit
- notepad2
- 记事本



3.1 软硬件及选项要求

	828D (车/铣/磨)			840Dsl		
软硬件	PPU24x	PPU26x	PPU28x	NCU710	NCU720	NCU730
<=5 幅界面	√/√/√	√/√/√	√/√/√	√	√	√
>5 幅界面	O/O/√	O/O/√	O/O/√	O	O	O
选项	6FC5800-0AP64-0YB0					
说明	√: 标配, O: 选项, -: 不支持					

只允许在一个操作区域内切换对话框

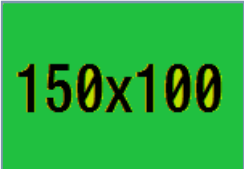
3.3 句法说明

传统句法

扩展句法 (Operate V4.7 起)

3.4 设计文档的结构

.ini 文件	定义界面使用的 com 界面文件，ts 语言文件等其他界面属性
<pre>[STARTFILES] StartFile06 = area := AreaMachine, dialog := SIMachine, menu := SIMaJogCircularGrindMenuHU, startfile := StartFile30 = area := Custom, dialog := SIEsCustomDialog, startfile := custom.com</pre>	
.com 文件	设计开发界面元素，界面方法
<pre>//M[Mask13.hd="变量展示：列表",mc="#696969"] DEF VAR_LISTBOX_Text1 = (U"1="列表条目1",2="列表条目2",3="列表条目3",4="列表条目4""列表条目1",,,"请从列表 DEF Var1 = (S///,"你选择了""\$R[1]"//,,), Var2 = (S///,"号列表""wr0/"//,,), Var3 = (S///,"wr0/"//,,), DEF Var4 = (S///,"wr0/"//,,), DEF VAR_LISTBOX_Text2 = (S/"1="列表条目1",,,"列表条目2",,,"列表条目3",,,"列表条目4""列表条目1",,,"请从列表中选择" DEF Var5 = (S///,"你选择了""PPV[V[0]"//,,), Var6 = (S///,"wr0/"//,,), Var7 = (S///,"wr0/"//,,), HS1=("用户名密码") MC2=（列表）--3-->2）</pre>	
.ts 文件	界面语言文本列表，主要用于系统语言切换时界面语言跟随改变

<pre> 124 </message> 125 <message> 126 <source>65530/NCK</source> 127 <translation>无效的调用参数 %4</translation> 128 </message> 129 <message> 130 <source>65531/NCK</source> 131 <translation>超出公差范围 %4</translation> 132 </message> </pre>	
.png 文件	界面显示的图形源文件，需对应屏幕分辨率放置在不同的文件夹下
	
.html 文件	在线帮助源码，按下系统帮助按键时弹出的帮助说明文档
<pre> <html><head><meta http-equiv="Content-Type" content="text/html; charset="UTF-8"/><title></title></head> <body> <table> <tr><td width="15%">65000</td> <td width="85%">Anwender-Alarm 65000</td> </tr> <tr><td valign="top" width="15%">Parameter:</td> <td width="85%"></td> </tr> <tr><td valign="top" width="15%">Eri&auml;uterung:</td> <td width="85%">Eri&auml;uterung zu Anwender-Alarm 65000</td> </pre>	

3.4.1 文档的优先级

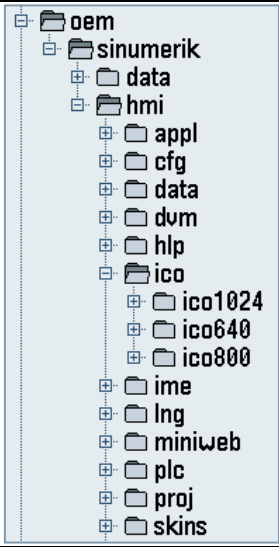
界面文件按 siemens , oem , user 三个文件夹进行优先级划分，user 目录下的文件优先级最高，其次为 oem , siemens。当在相同路径下遇到同名文件时，优先级最高的文件有效。

文件目录优先级		Siemens < oem < user （低 -> 高）
西门子目录 (siemens)	Linux	/card/ siemens /sinumerik/hmi
	Windows7	C:\ProgramData\Siemens\MotionControl\ siemens \sinumerik\hmi
制造商目录 (oem)	Linux	/card/ oem /sinumerik/hmi
	Windows7	C:\ProgramData\Siemens\MotionControl\ oem \sinumerik\hmi
用户目录 (user)	Linux	/card/ user /sinumerik/hmi
	Windows7	C:\ProgramData\Siemens\MotionControl\ user \sinumerik\hmi



3.4.2 文档的保存路径

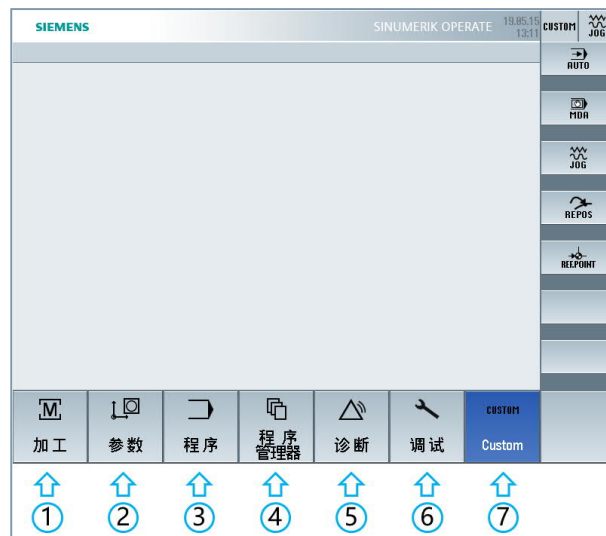
配置，文本，图片，帮助等文件的保存路径遵循如下规定。(hmi 详细路径见上文)

以 oem 目录为例	文件类型	扩展名	保存路径
	配置文件	*.ini	.../hmi/cfg
	界面文件	*.com	.../hmi/proj
	语言文件	*.ts	.../hmi/lng
	图形文件	*.png/* .jpg..	.../hmi/ico/ico640
			-ico640 (分辨率 640x..的屏)
			-ico800 (分辨率 800x..的屏)
			-ico1024 (分辨率 1024x..的屏)
			-ico1280 (分辨率 1280x..的屏)
	在线帮助	*.html	.../hmi/hlp
	...	...	...

4 界面结构

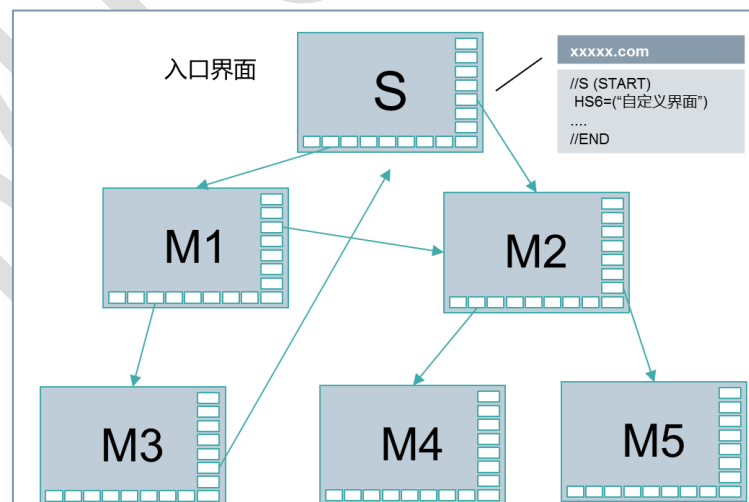
4.1 区域划分

系统界面分为 7 个操作区，操作区 1-6 为系统界面区，其中预留了一些空余的按键位置，作为用户自定义界面的入口；操作区 7【Custom】为系统专门为用户预留的独立操作区，此操作区不包含系统界面，用户可在此操作区整体设计规划一整套用户自定义界面。

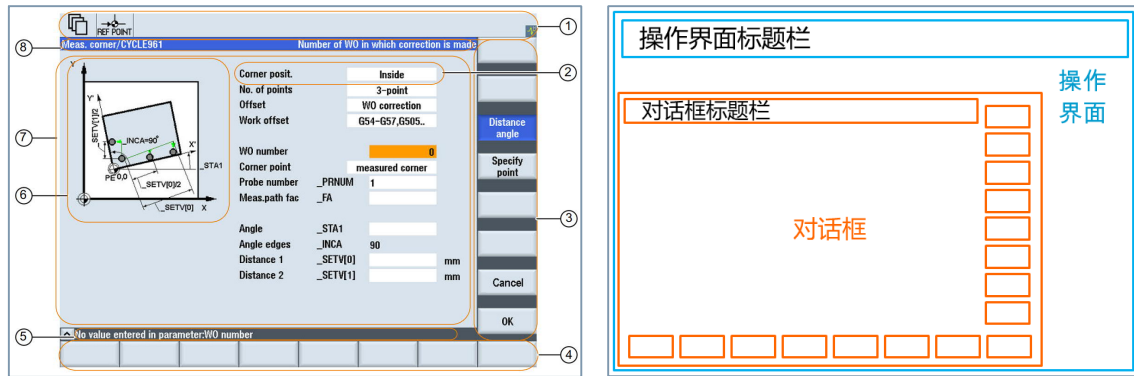


4.2 操作树

- 每个操作区（加工，参数...）都有固定的软键入口（S）
- 从入口界面可跳转到其他子界面（M）
- 子界面之间亦可灵活跳转



4.3 界面组成



- | | |
|--------------------|-------------------|
| ① 操作界面标题栏：机床状态显示 | ⑤ 对话框输出状态栏 |
| ② 界面变量：输入框，文本显示等元素 | ⑥ 界面变量：图形元素 |
| ③ 垂直按键：8个 | ⑦ 对话框 |
| ④ 水平按键：8个 | ⑧ 对话框标题栏：显示标题和长文本 |

示例

```
//M{Mask8,HD="Mask8,HD_AL=1,CB=0,TA=2,PG=1",HD_AL=1,CB=0,TA=2,PG=1}

DEF VAR02={TXT_X=020,TXT_Y=035,TXT_W=100,X=120,Y=035,W=110,WR=4,TYP="I",ST="VAR2",
TT="VAR02"}
DEF VAR03={TXT_X=020,TXT_Y=055,TXT_W=100,X=120,Y=055,W=110,WR=4,TYP="I",ST="VAR3",
TT="VAR03", VAL=0}
DEF VAR04={TXT_X=020,TXT_Y=075,TXT_W=100,X=120,Y=075,W=110,WR=4,TYP="I",ST="VAR4",}
DEF VAR05={TXT_X=020,TXT_Y=095,TXT_W=100,X=120,Y=095,W=110,WR=4,TYP="S",ST="VAR5",}

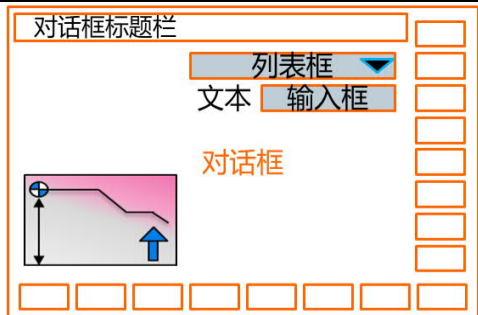
HS8=("M8",,se3)
VS2=("")
VS8=("HOME")

PRESS(VS8)
EXIT
END_PRESS

//END
```

5 对话框

5.1 概述

对话框的效果	对话框的结构
	<pre>//M (Mask1/"界面1") 变量定义 DEF VAR1=(R////) DEF VAR2=(R//) HS1=("HS1") VS1=("VS1") 方法动作 LOAD VAR1=0 VAR2=1.2 END_LOAD PRESS(HS1) LM("Mask2") END_PRESS 结束标记 //END</pre>

对话框包括	
- 名称 - 标题 - 尺寸 - 变量（文本，输入框...）	- 图片 - 属性 - 系统或用户变量 - 颜色

注意事项
- 对话框中应首先定义变量（DEF ...），然后定义水平/垂直软键（HS1,VS1），最后定义方法和动作（LOAD,PRESS,FOCUS..）。如顺序错误，将会导致界面出错，无法正常显示。 - 标题采用右对齐（HD_AL=2）时，会与变量长文本显示重叠，不支持长文本显示。

5.2 句法

传统句法	
//M(屏幕名称/标题/图片/尺寸/系统或用户变量/图片位置/属性/屏幕背景色，专用语言文件列表)	
扩展句法	
//M {<屏幕名称> ,HD=<标题> ,HLP=<图片> ,X=<X 位置> ,Y=<Y 位置> ,W=<宽度> , H=<高度> ,VAR=<系统或用户变量> ,HLP_X=<图片 X 位置> , HLP_Y=<图片 Y 位置> ,CM=<列对齐> ,CB=<对话框打开方式> , XG=<将辅助画面编译为 X3d 图片> ,PANEL=<已链接 FormPanels 的名称> , MC=<屏幕背景色> ,HD_AL=<屏幕标题对齐> ,LANGFILELIST=<屏幕专用语言文件列表>}	
示例	
//M(VariantTest/"My Mask"/////3) ... //END	//M{VariantTest, HD="My Mask", MC=3} ... //END

句法不区分大小写。

5.3 参数说明

	名称	说明	示例
0	屏幕名称	对话框的名称，.com 文件内对话框的专属 ID	Mask1
1	标题（HD）	对话框的文本标题，支持调用语言文本\$85000...	"Mask 标题"
2	图片（HLP）	对话框的帮助图片（背景图片）	HLP ="\\pic1.png"
3	尺寸（X,Y,W,H）	- 对话框的起始坐标、宽和高（X,Y,W,H）。 - 数值为整数，单位：像素。 - 默认值：全屏显示。	0,0,120,100 或 w=120
4	系统或用户变量 （VAR）	- 指定当前光标位置的系统和用户变量。 - 可通过系统或用户变量将光标位置传送给 NC 或 PLC - 第一个变量的光标位置为 1，顺序按变量设计顺序	"\$R1"
5	图片位置 （HLP_X,HLP_Y）	图片的位置，起始坐标（HLP_X, HLP_Y）。	30,30 或 HLP_X=30, HLP_Y=30
6	属性	CM(0,1)= 列对齐 ,CB(0,1)= 处理 CHANGE 方法 ,XG(0,1)= X3D 动画 , AL(0,2)= 标题文字对齐 ,KM(0,1)= 触摸屏滚动条 ,PG(0,1)= 带图片标题 , MA(0,2)= 触摸屏适配栏高 ,PA(0,1)= 分辨率像素优化 ,FA(0,2)= 行高适配 , NT(0,2)= 光标导航 ,NR(0,4)= enter 导航 ,TA(0,8)= 提示框对齐 ,	
	CM	列对齐	CM1 或 CM=1
	缺省	CM0	每行单独分列
		CM1	以包含最多列的行为标准分列
	CB	对话框 CHANGE 方法的处理顺序	
	缺省	CB0	打开对话框时处理所有 带预设值变量的 CHANGE 部分，
		CB1	只有在 CHANGE()附属值改变后才处理 CHANGE 部分
	XG	是否启用 X3D 动画作为帮助画面，HLP 中也应进行设置，该数值运行期间无法更改	
	缺省	XG0	不启用
		XG1	启用。
	AL	屏幕标题的对齐方式，扩展语法中使用 HD_AL=	
	缺省	AL0	左对齐
		AL1	右对齐
		AL2	居中
	KM	【触摸屏】虚拟键盘显示时，自由滚动条是否有效；优先级高于 easyscreen.ini 中对虚拟键盘的全局设置。 [GENERAL] DefaultVirtualKeyboardMode=1	
		KM0	禁用
	缺省	KM1	激活
	PG	对齐并预览 PNG 图片的窗口标题（只在编辑器中有效！） 示例： //M(MyPg1SampleMask"\"My PG1 Sample Mask\""/PG1)	
	缺省	PG0	不使用
		PG1	路径说明（最左），屏幕名称（右），长文本失效。

7		MA		【触摸屏】是否自适应匹配栏高，优先级高于在 easyscreen.ini 中的全局设置 [GENERAL] DefaultMultiTouchAdjustmentLevel=1		
		缺省	MA0	未设置匹配		
			MA1	只匹配未设置栏高/上间距的栏		
			MA2	匹配所有栏		
		PA		提高分辨率时精确按像素优化扩展各栏，优先级高于在 easyscreen.ini 中的全局设置 [GENERAL] DefaultPixelFineAdjustment=1		
		缺省	PA1	基于 640x480 计算位置进行拉伸		
			PA2	按像素优化拉伸		
		FA		与字体成比例设置栏高及行间距，优先级高于在 easyscreen.ini 中的全局设置 [GENERAL] DefaultFontAdjustment=1		
		缺省	FA0	按当前方式拉伸		
			FA1	与字体成比例		
			FA2	同 FA1,但会额外设置 X 坐标和列宽（需属性 XG=1）		
		NT		光标导航方向		
		缺省	NT0	按行导航，标准模式，与其他操作区相似		
			NT1	几何方向导航（上下左右）直到屏幕边界		
			NT2	按行导航，行中向右，直到行末		
		NR		输入框按 Enter 键导航方向		
		缺省	NR0	按表格顺序		
			NR1-4	NR1/NR2/NR3/NR4	向上/向下/向左/向右	
		TA		提示框对齐方式		
		缺省	TA0	自动		
			TA1-4	TA1/ TA2/ TA3/ TA4	左/右/上/下	
			TA5-8	TA5/ TA6/ TA7/ TA8	左上/左下/右上/右下	
		MC		屏幕背景色，例：//M(MyMask/"MyMask"/////6)或 MC=6		MC="#00FFFF"
		, 语言文件 (LANGFILELIST)		屏幕专用语言文件列表		

5.4 可编辑的对话框属性

如下属性可以在 PRESS 等方法中读写。

可读写属性	只读属性
- HD=标题("MMM") - HLP=帮助画面("\\123.png") - VAR=系统变量("\$R1") - MC=背景色(6,或"#00ffff")	- CM=列对齐 (0-1) - CB=处理 change 方法 (0-1) - XG=连接 X3D (0-1) - AL=标题对齐方式 (0-2)

6 变量

6.1 概述

对话框中所显示的文本，列表框，输入框，图片等元素都是由定义好的界面变量实现的。

6.2 句法

定义变量	以 DEF 关键字起始书写 “<变量名>= ” 来定义变量。具体语法见下表
传统句法	以 “()” 和 “/” “,” 组成，属性之间以 “/” 隔开，位置有顺序关系。意味着第 n 个 “/” 后代表了什么属性就应该填写什么代码。代码编写上形式固定，容易写错，不易解读
扩展句法	主要以 “{}” 和 “关键字” 赋值的形式组成，每个属性有固定的关键字，位置不分先后，系统只编译已编写属性，代码更易读，系统编译速度更快

传统句法	
DEF 变量名称=(变量类型/极限值或转换栏/预设值/文本/属性/帮助画面/系统或用户变量/短文本的位置/输入-输出栏的位置/颜色/在线帮助文件/单元选择框)	
扩展句法	
DEF <变量名称> = {TYP=<类型> ,MIN=<最小值> ,MAX=<最大值> ,TGL=<转换值> , VAL=<预设值> ,LT=<长文本> ,ST=<短文本> ,GT=<图片文本> ,UT=<单位文本> , TT=<提示框文本> ,TG=<转换选项> ,WR=<输入模式> ,AC=<访问级> ,AL=<文本对齐> , FS=<字体大小> ,LI=<极限值处理> ,UR=<刷新率> ,CB=<对话框打开方式> , HLP=<辅助画面> ,VAR=<系统变量和用户变量> ,TXT_X=<短文本 X 位置> , TXT_Y=<短文本 Y 位置> ,TXT_W=<短文本宽度> ,TXT_H=<短文本高度> , X=<输入/输出栏 X 位置> ,Y=<输入/输出栏 Y 位置> ,W=<输入/输出栏宽度> , H=<输入/输出栏高度> ,UT_DX=<输入/输出栏之间的间距和单位文本> , UT_W=<单位文本宽度> ,BC=<输入/输出栏背景色> ,FC=<输入/输出栏前景色> , BC_ST=<短文本背景色> ,FC_ST=<短文本前景色> ,BC_GT=<图片文本背景色> , FC_GT=<图片文本前景色> ,BC_UT=<单位文本背景色> ,FC_UT=<单位文本前景色> , SC1=<进度条信号色 1> ,SC2=<进度条信号色 2> ,SVAL1=<进度条阈值 1> , SVAL2=<进度条阈值 2> ,DT=<显示类型> ,DO=<显示对齐> ,OHLP=<在线帮助> , LINK_TGL=<单元选择框的名称>}	
示例	
DEF Var1 = (R///,"实际值",,"mm"//"/Var1.png"////8,2)	DEF Var1={TYP="R",ST="实际值",UT="mm", HLP="Var1.png",FC_ST=8,BC_ST=2}

句法不区分大小写。

6.3 参数说明

下表第一列的序号代表了传统句法以 “/” 分割变量属性的位置。

例如预设值在第 2 个 “/” 后面，下表中预设值前面为 2。方便查看使用。

	名称	说明	示例
0	变量类型 (TYP="")	R[x] Real 实数型 (x代表小数点位数, 正整数)	R1,R3
		I Integer 整数型	
		S[x] String 字符串 (x代表字符串长度, 正整数)	
		C Char 单字符	
		B Bool 布尔型	
		V Variant 变量	
		DEF VAR1=(R///,"实数",) DEF VAR1={TYP="R",ST="实数"}	
1	极限值 (MIN=,MAX=, SVAL1=, SVAL2=)	极小值, 极大值。传统句法中极限值之间通过逗号隔开。扩展句法中 MIN=极小值, MAX=极大值。 当变量定义为可变色进度条时, 极限值增加 2 个变色阈值 SVAL1 和 SVAL2。阈值在极小值和极大值之间。	
	转换栏 (TGL="* ")	输入框为列表格式, 列表通过 * 开始, 各输入项用逗号隔开。 如果只输入一个 *, 则建立一个可变的转换栏。	
2	预设值 (VAL=)	如果没有定义任何预设值并且没有分配系统或者用户变量给变量, 则分配转换栏的第一个单元。如果没有定义转换栏, 则不进行预设。	
3	文本	可以显示一个图形代替文本。传统句法: 顺序以预先规定: 长文本, 短文本, 图形文本, 单位文本, 提示框。均支持语言文本。	
		长文本属性 在对话框标题栏显示的单行文本	
		短文本属性 在对话框内以文本或图形显示 (图形语法"*.png", 双引号中书写\和图形完整名称, 图形文件放在 ico 中对应分辨率文件夹下)	
		图形文本属性 在对话框内以文本形式显示, 描述图形的说明文本	
		单位文本属性 在输入框后以单位形式显示	
		提示文本属性 当光标定位在输入框后显示的提示文本。	
4	属性	DT=<显示类型>, DO=<显示方向>, UR=<刷新速度>, TG=<转换符号开关>, WR=<读写权限>, AC=<访问等级>, AL=<短文本对齐方式>, FS=<字体大小>, LI=<极限值检查方式>, CB=<触发 CHANGE 方法>, EI=<空字符输入框响应>, OHLP=<在线帮助>, VAR=<系统或用户变量>, TXT_X=<短文本 X 位置>, TXT_Y=<短文本 Y 位置>, TXT_W=<短文本宽度>, TXT_H=<短文本高度>, X=<输入/输出栏 X 位置>, Y=<输入/输出栏 Y 位置>, W=<输入/输出栏宽度>, H=<输入/输出栏高度>, UT_DX=<输入/输出栏之间的间距和单位文本>, UT_W=<单位文本宽度>, BC=<输入/输出栏背景色>, FC=<输入/输出栏前景色>, BC_ST=<短文本背景色>, FC_ST=<短文本前景色>, BC_GT=<图片文本背景色>, FC_GT=<图片文本前景色>, BC_UT=<单位文本背景色>, FC_UT=<单位文本前景色>, SC1=<进度条信号色 1>, SC2=<进度条信号色 2>, SVAL1=<进度条阈值 1> ,	
		DT 显示类型	
		缺省 DT0 标准输入/输出框, 转换栏	
		DT1 变色进度条	
		DT2 双色进度条	
		DT3	
		DT4 列表框	
		DT5 密码框 (星号)	
		DT6 多行输入/输出框	
		DO 进度条显示方向	
		缺省 DO0 从左往右	

	DO1	从右往左	
	DO2	从下往上	
	DO3	从上往下	
UR		控制显示的刷新速度，根据配置不同，CPU 负载可能被大幅降低。	
缺省	UR0	SI::standardUpdateRate()，当前=200ms	
	UR1	50ms	
	UR2	100ms	
	UR3	200ms	
	UR4	500ms	
	UR5	1000ms	
	UR6	2000ms	
	UR7	5000ms	
	UR8	10000ms	
TG		转换符号	
缺省	TG0	转换符号关闭	
	TG1	转换符号打开	
WR		读写权限与输入中心	
	WR0	短文本可见，输入框不可见	
	WR1	短文本可见，输入框只读，无输入中心	
缺省	WR2	短文本可见，输入框可读写，	
	WR3	短文本可见，输入框只读，带输入中心	
	WR4	所有单元不可见，无输入中心	
	WR5	与 WR2 相同，但是立即保存输入值。WR2--退出栏或返回键才开始保存值	
AC		如果等级未达到，行显示为灰色	
	AC0	系统级	
	AC1	制造商	
	AC2	服务	
	AC3	用户	
	AC4	钥匙开关 3	
	AC5	钥匙开关 2	
	AC6	钥匙开关 1	
缺省	AC7	钥匙开关 0	
AL		短文本对齐方式	
缺省	AL0	左对齐	
	AL1	右对齐	
	AL2	居中	
FS		短文本，输入框，图形文本，单位文本字体大小	
缺省	FS1	标准字体大小 (8Pt)	
	FS2	双倍大小	
LI		检查变量极限值方式	
缺省	LI0	没有检查	
	LI1	检查极小值	
	LI2	检查极大值	
	LI3	检查极小值和极大值	
CB		对话框打开时 CHANGE 属性，优先于对话框总定义	
	CB0	当变量有预设值时，打开时触发 CHANGE 方法	
缺省	CB1	通过显示屏幕无法触发 CHANGE 方法	
EI		输入空字符串时输入栏的反应	
缺省	EI0	接受空白输入	

		E11	空白输入会导致输入栏出错，并自动撤销为之前的数值	
5	帮助画面		帮助画面文件的名称在双引号中 ("*.png")。如果光标移至该变量，则自动显示画面 (替代目前的图形)。	
6	系统或用户变量		可以将一个来自 NC/PLC 的系统数据或用户数据指定给变量。系统或者用户变量用双引号括起。(文档：参数手册 系统变量，/PGAsI/)	
7	短文本位置		和左边缘/上边缘的间距、宽度，单位为像素，数据之间用逗号隔开。(X 坐标，Y 坐标，宽度，高度)	
8	输入/输出栏位置		(和左边缘/上边缘的间距、宽度、高度) 单位为像素，数据之间用逗号隔开。(X 坐标，Y 坐标，宽度，高度)	
9	颜色		输入/输出栏、短文本、图形文本、单位文本的前景色和背景色及进度条的信号色。颜色通过逗号隔开。数据参见附录颜色表。当变量为变色进度条时 (DT=1) ,可选择配置两个信号色 SC1,SC2.	
	缺省	输入栏	前景色：黑色，背景色：白色	
	缺省	短文本	前景色：黑色，背景色：透明	
			变量定义中要求的颜色顺序如下，颜色之间通过逗号隔开，无设定值时可设置为空，逗号不可省略： 1. 输入/输出栏的前景色：FC 2. 输入/输出栏的背景色：BC 3. 短文本的前景色：FC_ST 4. 短文本的背景色：BC_ST 5. 图形文本的前景色：FC_GT 6. 图形文本的背景色：BC_GT 7. 单位文本的前景色：FC_UT 8. 单位文本的背景色：BC_UT 9. 信号色 1：SC1 10 信号色 2：SC2	
10	在线帮助文件		在线帮助文件的名称在双引号中。("*.html")	
11	单位选择框		使用集成了单位选择的输入/输出栏可在不同的单位之间进行切换。此外会显示一个带有转换符号的提示框，其中为该功能的相应说明信息。按 " select " 按键可转换单位。 示例：DEF VarEdit=(R////////200,,100//I"VarTgl") , VarTgl=(S*0="mm",1="inch"/O//WR2///302,,40) 或者 DEF VarEdit_2={TYP="R", VAL=1.234, X=200, W=100, LINK_TGL="vartgl_2"}, VARTgl_2={TYP="S", TGL="* 0=""mm"", 1=""inch"", WR=2, X=302, W=40}	

6.4 其他说明

6.4.1 变量值 (val)

变量可通过以下三种方式赋值：

- 预设值：定义变量时给出预设值
- 系统或用户变量：变量关联了系统或用户变量
- 直接赋值：在界面方法中直接为变量赋值，如 LOAD,CHANGE 等

示例	VAR3 = VAR4 + SIN(VAR5) VAR3.VAL = VAR4 + SIN(VAR5)
----	--

6.4.2 变量状态 (vld)

通过变量状态的属性可以在运行时查询，变量是否包含有一个有效值。

当变量具有有效值时，vld 值为 TRUE，否则为 FALSE

示例	IF VAR1.VLD == FALSE VAR1 = 84 ENDIF
----	--

6.4.3 辅助变量

DEF OTTO；定义一个辅助变量

辅助变量是内部计算变量，只有变量值和状态两个属性，在对话框中不可见，是 VARIANT 类型。

示例	LOAD OTTO = " TEST " END_LOAD LOAD OTTO = REG[9].VAL END_LOAD
----	--

6.4.4 变量计算

在每次退出输入/输出栏（通过 ENTER 或者 SELECT 键）后计算变量。

6.4.5 系统变量间接编译地址

字符和变量之间使用 "<<" 连接符进行连接。

示例	DEF ACHSE={TYP="C",VAL="X"},WEG={TYP="R3"} PRESS(HS1) WEG.VAR="\$AA_DTBW["<<ACHSE<<"]"；通过变量编译轴地址 END_PRESS
----	---

6.4.6 更改变量属性

通过“变量.属性”的方式可在界面程序中更改对变量的赋值。

示例	HS3.st = " 新文本 "；更改软键标签
----	-------------------------

7 按键

7.1 水平/竖直软键

7.1.1 定义软键栏

界面开发中除了定义起始软键栏“//S (START)”，对话框“//M (Mask)”外，还可以单独定义一个软键栏。

当调用软键栏时，界面只改变软键栏，对话框的界面显示不发生变化。

在软键栏“//S (_SK_BAR1)”，起始软键栏“//S (START)”，对话框“//M (Mask)”中均可定义软键。

软键栏句法

//S (软键栏名称)	软键栏开始标记
HSx=...	定义软键
PRESS(HSx)	方法的开始标记
LM...	动作
END_PRESS	方法的结束标记
//END	软键栏结束标记

7.1.2 软键句法

传统句法				
纯文本	SK=(“文本”，访问等级，状态，图标对齐方式，文本与图标对齐方式)			
文本+图标	SK=(["\\图标名称","文本"],访问等级，状态，图标对齐方式，文本与图标对齐方式)			
纯图标	SK=(("\\图标名称",访问等级，状态，图标对齐方式，文本与图标对齐方式)			
扩展句法				
纯文本	SK={ST=""文本"", AC=.., SE=.., PA=..,TP=..}			
文本+图标	SK={ST=[""文本"", ""\\图形名称""], AC=.., SE=.., PA=..,TP=..}			
纯图标	SK={ST=""\\图形名称"", AC=.., SE=.., PA=..,TP=..}			
示例				
HS1=(["\\dg_axis_ok.png","text"],ac7,se1)				
VS8=(["\\dg_axis_ok.png",\$83533],,se1)				
HS1={st= ["\\dg_axis_ok.png",\$83533],ac=7,se=1,pa=3,tp=0}				
注意事项				
已经定义的软键，才可以更改属性或使用 PRESS 等方法				
属性				
ST	软键文本，包含纯文字（支持中文）和图标，支持语言文本。 软键文本，用 %n 进行换行，最多 2 行，每行各 9 个字符。 软键图标应遵循如下像素规则，否则无法完整显示。			
	图标分辨率	640 x 480 mm → 25 x 25 像素 640 x 480 mm → 25 x 25 像素 800 x 600 mm → 30 x 30 像素	参考显示屏	OP 08: OP 010: OP 012:

		1024 x 768 mm → 40 x 40 像素 1280 x 1024 mm → 72 x 72 像素		OP 015: OP 019:
	纯图片按键 分辨率	640 x 480 mm → 25 x 25 像素 640 x 480 mm → 25 x 25 像素 800 x 600 mm → 76 x 28 像素 1024 x 768 mm → 40 x 40 像素 1280 x 1024 mm → 72 x 72 像素		
AC	软键访问等级			
	ac0 到 ac7 (ac7 : 缺省设置)			
SE	软键状态			
	se1:可见 (预设置 , 可作为常态) se2:不可操作 (灰色标签 , 可作为禁止操作的状态) se3:高亮显示 (可作为软键按下的状态)			
PA	软键图标对齐方式			
	0:左 1:右 2:居中 3:上 (默认) 4:下			
TP	以软键图为准的文本对齐			
	0:文本和图形未对齐 1:文本和图形已对齐 (默认)			

示例	<pre>//M{Mask1,hd="软键"} VS1=("[\ldg_axis_ok.png","_SK"],,se1,PA0) VS2=("[\ldg_axis_ok.png","_SK"],,se1,PA1) VS3=("[\ldg_axis_ok.png","_SK"],,se1,PA2) VS4=("[\ldg_axis_ok.png","_SK"],,se1,PA3) VS5=("[\ldg_axis_ok.png","_SK"],,se1,PA4) VS6=("") VS7=("[\lsk/sk_next.png") VS8=(SOFTKEY_NAV_BACK,,se1) HS1=("[\ldg_axis_ok.png","_SK"],,se1,PA0) HS2=("[\ldg_axis_ok.png","_SK"],,se1,PA1) HS3=("[\ldg_axis_ok.png","_SK"],,se1,PA2) HS4=("[\ldg_axis_ok.png","_SK"],,se1,PA3) HS5=("[\ldg_axis_ok.png","_SK"],,se1,PA4) HS6=("") HS7=("[\lsk_sim_stop.png",,se2,) HS8=("[\lsk_sim_start.png",,se3,) PRESS(VS8) EXIT END_PRESS PRESS(VS7) LS("Mask2",,1) END_PRESS</pre>			
----	--	--	--	--

	<pre>//END //S{Mask2} HS1=("") HS2=("") VS4=("") VS7=("") VS8=("\\sk/sk_before.png",,se1) PRESS(VS8) LM("Mask1") END_PRESS //END</pre>
--	--



7.1.3 可更改的软键属性

软键的文本、访问等级和状态的属性可以在运行期间在方法中改变：

SK.st = "文本"	；软键标签
SK.ac = 访问等级	；软键访问等级
SK.se = 状态	；软键状态
SK.pa = “ 软键图对齐 ”	；带图形的软键
SK.tp = “ 以软键图为基准的文本对齐 ”	；带图形和标签的软键

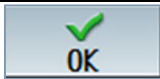
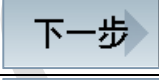
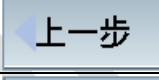
7.1.4 系统预定义按键（水平/竖直软键栏）

为方便客户开发，简化客户开发代码，系统预定义按键提供了界面文字和图标，无需用户编程和排版，方便用户使用，而且预定义的按键与系统界面的风格保持一致，支持语言切换。

预定义按键背后的 PRESS 方法由用户自行设计。

预定义按键用于水平和竖直软键栏，不用于触摸屏按键。

名称	按键（英文）	按键（中文）
----	--------	--------

SOFTKEY_OK		
SOFTKEY_CANCEL		
SOFTKEY_APPLY		
SOFTKEY_MORE		
SOFTKEY_BACK		
SOFTKEY_ASSISTANT_NEXT		
SOFTKEY_ASSISTANT_PREVIOUS		
SOFTKEY_NAV_BACK		

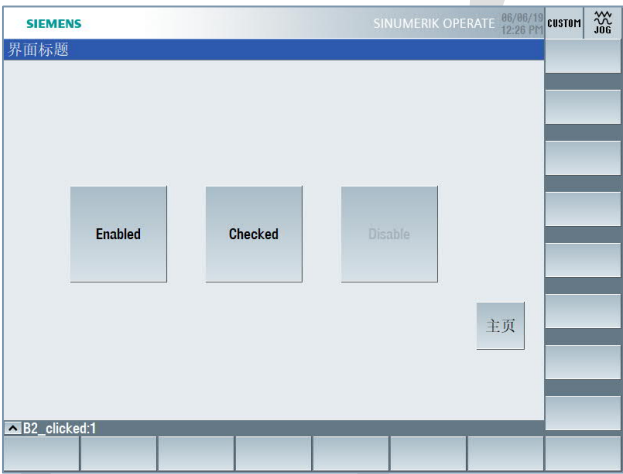
示例	vs1={st="softkey_ok",se=1} vs1=(softkey_ok,ac7,se1)
----	---

8 触摸屏-按钮

8.1 概述

在触控操作上，借助 Run MyScreens 可以简单的方式，配置任意放置的按钮（ TouchButton ）。 TouchButton 的配置有以下特性

- 可控制按键 “ 已点击 ” 和 “ 已检查 ” 两种状态的更改并触发相应的操作
- 可通过单触控或多触控、鼠标和键盘操作
- 支持文本、图片显示，根据当前屏幕分辨率进行显示
- 有两种显示方式：“ Softkey Look&Feel ”（ 同操作软键 ）和 “ 用户专用 ”

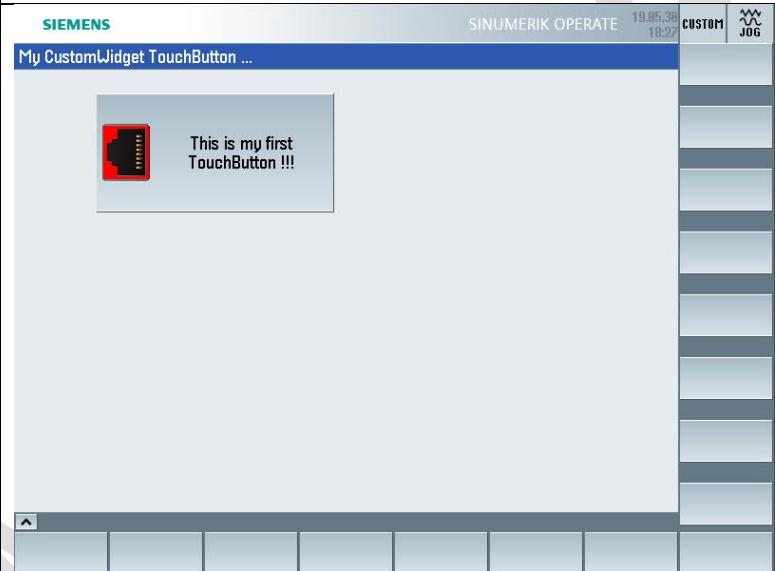


8.2 要求

硬件要求	828D PPU290.4
软键要求	828D SW04.08 SP4 及以上
选项要求	无

8.3 按钮定义

句法	
DEF 变量名=(W///,"slesstdcw.SIEsTouchButton"/////X,Y,TB_W,TB_H/0,0,0,0)	
属性	
W	变量类型为 Widget 函数类型
slesstdcw.SIEsTouchButton	CustomWidget 函数名，系统预定义的触摸屏按钮函数
X	按钮距对话框左边缘的距离
Y	按钮距对话框上边缘的距离
TB_W	按钮宽度
TB_H	按钮高度
示例	
DEF MyTB1 = (W///,"slesstdcw.SIEsTouchButton"/////70,20,200,100/0,0,0,0)	

示例
<pre>//M(MyTBMask/"My CustomWidget TouchButton ...") DEF MyTB1 = (W///,"slesstdcw.SIEsTouchButton"/////70,20,200,100/0,0,0,0) LOAD WRITECWPROPERTY("MyTB1", "text", "This is my first TouchButton !!!") WRITECWPROPERTY("MyTB1", "textPressed", "This is my first TouchButton (pressed)!!!") WRITECWPROPERTY("MyTB1", "picture", "dsm_remove_trashcan_red.png") WRITECWPROPERTY("MyTB1", "pictureAlignment", "left") WRITECWPROPERTY("MyTB1", "scalePicture", FALSE) WRITECWPROPERTY("MyTB1", "picturePressed", "slsu_topology_empty_round_slot.png") WRITECWPROPERTY("MyTB1", "picture", "slsu_topology_empty_slot_left_error.png") END_LOAD</pre>
说明
• 首先定义一个按钮 • 通过 WRITECWPROPERTY 函数更改按钮变量的属性，从而丰富按钮显示 • WRITECWPROPERTY 函数可以在 load,press 等其他方法中调用 

8.4 按钮属性

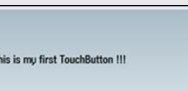
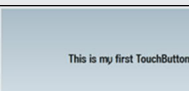
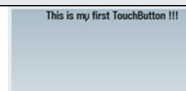
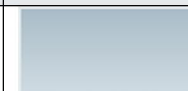
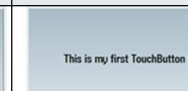
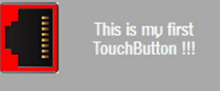
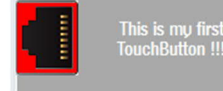
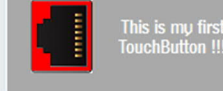
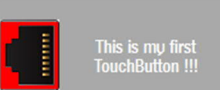
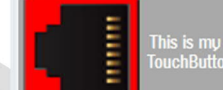
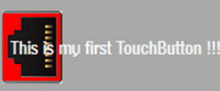
属性&数值	说明
ButtonStyle	显示风格：0=系统风格；1=用户专用
Enabled	可操作性：TRUE=可操作；FALSE=不可操作
Checkable	按钮是否具有自锁功能：TRUE=是；FALSE=否
ShowFocusRect	输入焦点是否显示焦点矩形：TRUE=是；FALSE=否
Picture	未按压状态时的图标显示
PicturePressed	按压状态时的图标显示
PictureAlignment	图片对齐：1=左，2=右，32=上，64=下，128=居中
PictureAlignmentString	图片对齐：left=左，right=右，top=上，bottom=下，center=居中
ScalePicture	缩放按钮图片：TRUE=缩放；FALSE=不缩放
PictureKeepAspectRatio	保持页面比例缩放按钮图片：TRUE=保持比例；FALSE=不保持比例

Text	未按压时的显示文本
TextPressed	按压时的显示文本，disable 的按钮无此属性
TextAlignment	文本对齐：1=左，2=右，32=上，64=下，128=居中
TextAlignmentString	文本对齐：left=左，right=右，top=上，bottom=下，center=居中
TextAlignedToPicture	显示文本是否相对于图标定位：TRUE=是；FALSE=否
BackgroundPicture	按钮的背景画面
BackgroundPictureAlignme nt	背景画面对齐：1=左，2=右，32=上，64=下，128=居中
BackgroundPictureAlignme ntString	背景画面对齐：left=左，right=右，top=上，bottom=下，center=居中
ScaleBackgroundPicture	缩放背景画面：TRUE=缩放；FALSE=不缩放
BackgroundPictureKeepAs pectRatio	按照页面比例缩放背景画面：TRUE=保持比例；FALSE=不保持比例
以下属性，在 buttonStyle=1（用户专用风格）时有效	
BackColor	未按下时的背景色
BackColorChecked	Checked 状态下的背景色
BackColorPressed	按压状态时的背景色
BackColorDisabled	未激活状态下的背景色
TextColor	未按下时的文本颜色
TextColorChecked	Checked 状态下的文本颜色
TextColorPressed	按压状态时的文本颜色
TextColorDisabled	未激活状态下的文本颜色
Flat	显示方式：TRUE - 平面方式；FALSE - 3D 方式。

说明
- 所提供的矩形框中的文本达到字数限值时自动转换。只有待显示的文本来自语言文件，才可进行强制换行。为此必须在文本中相应的位置插入“ %n ” - 图片文件自动从当前分辨率文件夹读取；在状态为未激活的按钮上自动显示为灰度梯度 - PicturePressed 如未指定文件名，将显示属性“ Picture ”确定的图片 - 自锁按钮（ Checkable=true ）:按下时，转变为按下状态，再次按下按钮时恢复为常态 - 按钮不自锁（ Checkable=false ）:按下时，转变为按下状态，松开时恢复为常规状态

8.4.1 示例

系统风格	用户专用	平面方式	3D 方式
可操作	不可操作	未按下时的显示	按下的显示

 This is my first TouchButton !!!		 This is my first TouchButton !!!		 This is my first TouchButton !!!		 This is my first TouchButton !!! (Pressed!)					
有焦点矩形		无焦点矩形		Picture		PicturePressed					
 This is my first TouchButton !!!		 This is my first TouchButton !!!		 This is my first TouchButton !!!		 This is my first TouchButton !!! (Pressed!)					
文本对齐-左		文本对齐-右		文本对齐-上		文本对齐-下		文本对齐-居中			
 This is my first TouchButton !!!		 This is my first TouchButton !!!		 This is my first TouchButton !!!		 This is my first TouchButton !!!		 This is my first TouchButton !!!			
setMargins(-1, -1, -1, -1, -1)				setMargins(0, 0, 0, 0, 0)				setMargins(20, 20, 20, 20, 20)			
 This is my first TouchButton !!!				 This is my first TouchButton !!!				 This is my first TouchButton !!!			
"scalePicture" : FALSE				"scalePicture" : TRUE				"pictureKeepAspectRatio" : TRUE			
 This is my first TouchButton !!!				 This is my first TouchButton !!!				 This is my first TouchButton !!!			
"textAlignedToPicture":FALSE						"textAlignedToPicture":TRUE					
 This is my first TouchButton !!!						 This is my first TouchButton !!!					

8.4.2 读按钮属性

返回值 = ReadCWProperty("按钮名称", "按钮属性")

示例：

REG[0] = ReadCWProperty("MyTouchButton", "Text")

8.4.3 写按钮属性

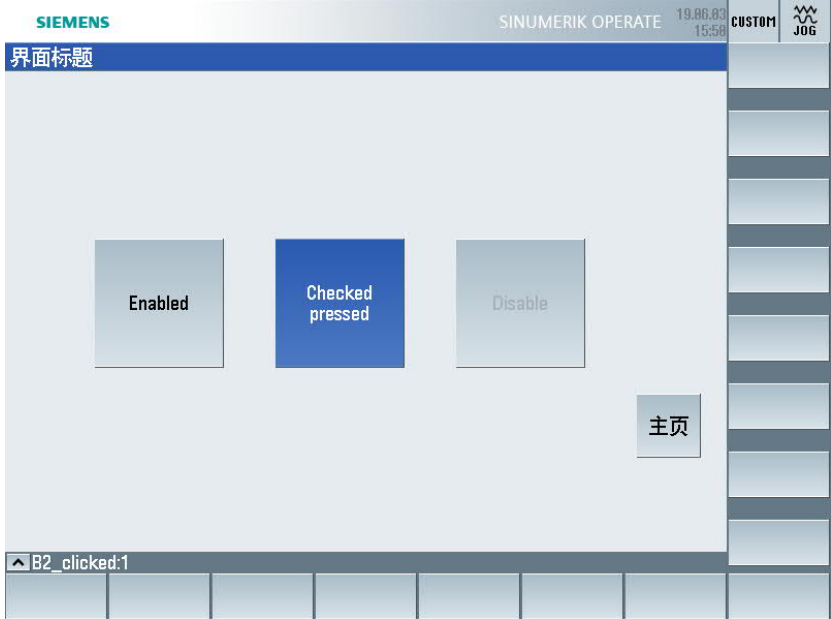
WriteCWProperty("按钮名称", "按钮属性", 写入值)

示例：

WriteCWProperty("MyTouchButton", "Picture", "sk_ok.png")

8.5 按钮动作

TouchButton 类别	TouchButton 属性	操作按钮时发出动作
可操作，不可转换	Enabled=1；Checkable=0	Clicked；
可操作，可转换	Enabled=1；Checkable=1	Checked -> clicked；
不可操作	Enabled=0；	clickedDisabled；

句法
通过 SUB 子程序实现按钮的动作操作，句法如下 SUB(on_按钮名_动作名称) ... END_SUB
说明
• 上述信号总是在进行了操作后才发送（即只有松开了鼠标或空格键或在多点触控点击之后） • SIEsTouchButton 仅有纯粹的开关功能，无法实现按键功能 • 多点触控操作时，当完全识别到点击手势后（在 0.7 秒内按住并松开），才会发送 Click 信息 • 通过系统界面变量 SIGARG[0]可读取 TouchButton 的转换状态 (bool)
示例


8.5.1 点击按钮 (click)

句法	SUB(on_按钮名_clicked) ... END_SUB
示例	<pre> DEF MyTB1 = (W////,"slesstdcw.SIEsTouchButton"/////070,130,100,100/0,0,0,0) load WRITECWPROPERTY("MyTB1", "Enabled", true) end_load SUB(on_MyTB1_clicked) DLGL("B1_clicked:"<<SIGARG[0]) END_SUB </pre>

8.5.2 转换按钮 (checked)

句法	SUB(on_按钮名_checked) ... END_SUB
----	--

示例	<pre> DEF MyTB2 = (W///,"slesstdcw.SlEsTouchButton"/////210,130,100,100/0,0,0,0) load WRITECWPROPERTY("MyTB2", "Checkable", true) end_load SUB(on_MyTB2_clicked) DLGL("B2_clicked:"<<SIGARG[0]) END_SUB SUB(on_MyTB2_checked) DLGL("B2_checked:"<<SIGARG[0]) END_SUB </pre>
----	--

8.5.3 不可操作按钮 (clickedDisabled)

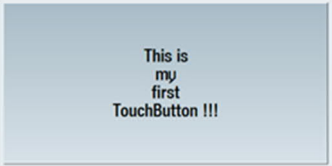
句法	<pre> SUB(on_按钮名_clickedDisabled) ... END_SUB </pre>
示例	<pre> DEF MyTB3 = (W///,"slesstdcw.SlEsTouchButton"/////350,130,100,100/0,0,0,0) load WRITECWPROPERTY("MyTB3", "Enabled", false) end_load SUB(on_MyTB3_clickedDisabled) DLGL("B3_clickedDisabled:"<<SIGARG[0]) END_SUB </pre>

8.6 示例

8.6.1 示例 1：按下按钮跳转至其他界面

示例	<pre> DEF MyTB1 = (W///,"slesstdcw.SlEsTouchButton"/////070,130,100,100/0,0,0,0) load WRITECWPROPERTY("MyTB1", "text", "跳转") end_load SUB(on_MyTB1_clicked) LM("Mask2") ;调用 LM 方法跳转到 Mask2 界面 END_SUB </pre>
----	--

8.6.2 示例 2：显示文本来自语言文件

easyscreen.ini
[LANGUAGEFILES]LngFile03 = user.txt
user_eng.txt
85001 0 0 "This is%nmymy%nfirst%nTouchButton !!!"
custom_tb.com
WRITECWPROPERTY("MyTB1", "text", \$85001)
结果


9 初始化文件

9.1 easyscreen.ini

easyscreen.ini 用于声明界面入口文件，语言文件等。

在 [STARTFILES] 标签下定义界面所属位置及文件名

在 [LANGUAGEFILES] 标签下定义语言文件的文件名

行注释：行首加 “ ; ” 或 “ # ” 注释整行，暂不支持区块注释。

用户可用入口及配置句法可参考 easyscreen.ini 模板文件(见附录 1)，文件路径如下

linux：

</card/siemens/sinumerik/hmi/cfg/easyscreen.ini>

windows7：

<C:\ProgramData\Siemens\MotionControl\siemens\sinumerik\hmi\cfg\leasyscreen.ini>

9.1.1 标签[STARTFILES]

不同区域的入口用不同的 StartFile xx 来定义， xx 索引号不能相同

✓

```
[STARTFILES]
StartFile01 = area := AreaStartup, dialog := SISuDialog, startfile := startup.com
StartFile02 = area := Custom, dialog := SIEsCustomDialog, startfile := custom.com
StartFile03 = area := AreaProgramEdit, dialog := SIProgramEdit, startfile := aeditor.com
```

用户可更改 StartFile 索引号和界面文件名称

✓

```
[STARTFILES]
StartFile01 = area := AreaProgramEdit, dialog := SIProgramEdit, startfile := aeditor.com
StartFile12 = area := Custom, dialog := SIEsCustomDialog, startfile := customuser1.com
StartFiles101 = area := AreaStartup, dialog := SISuDialog, startfile := custom_test1.com
```

StartFile 为固定句法， xx 为数字索引号，不可重复，

✗

```
[STARTFILES]
StartFile01 = area := AreaProgramEdit, dialog := SIProgramEdit, startfile := aeditor.com
StartFile01 = area := Custom, dialog := SIEsCustomDialog, startfile := custom1.com
```

StartFile 索引号不可有字母

✗

```
StartFiles1 = area := AreaStartup, dialog := SISuDialog, startfile := custom2.com
```

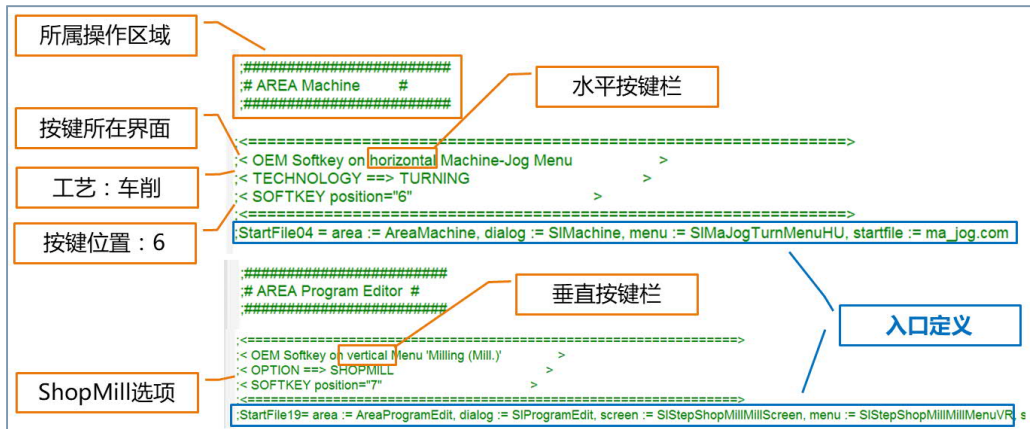
同一个区域的界面入口只能定义一次，不允许重复定义

✗

```
[STARTFILES]
StartFile01 = area := AreaProgramEdit, dialog := SIProgramEdit, startfile := aeditor.com
StartFile02 = area := Custom, dialog := SIEsCustomDialog, startfile := custom1.com
StartFile03 = area := Custom, dialog := SIEsCustomDialog, startfile := custom2.com
```

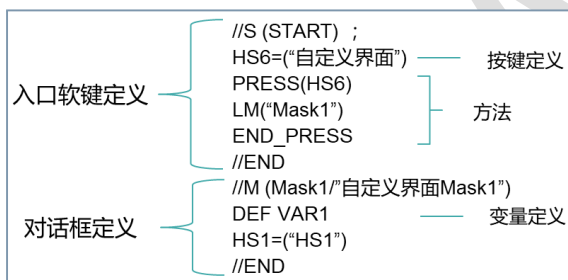
9.1.2 使用模板文件

- 按“界面区域”“机床工艺”在模板文件中找到对应的“入口定义”
- 拷贝（或更新）至系统 oem 或 user 文件夹下的 easyscreen.ini 中
- 注意修改 StartFile 索引号和.com 文件名，不要出现上文所列重名等句法错误
- horizontal/vertical，给出了软键栏的类别：水平/竖直
- SOFTKEY position=，给出了空余按键的位置



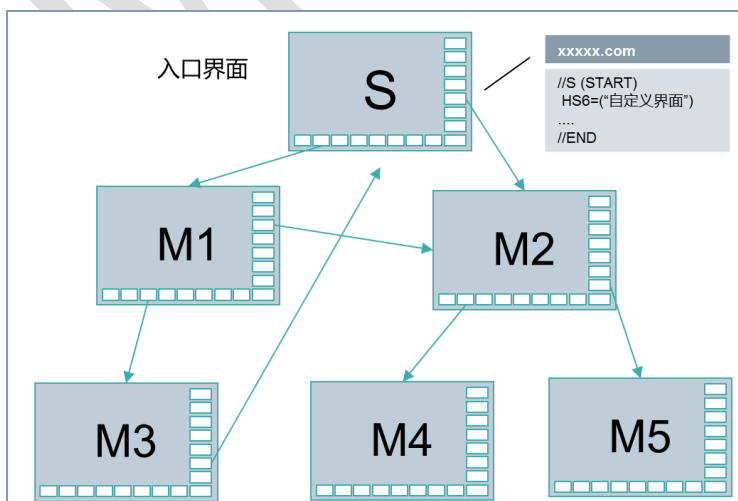
9.1.3 入口关联的.com 文件

- 入口关联的.com 文件中有且仅有一个//S(START)起始软键栏
- 仅在//S(START)中定义入口的按键位置，及按下按键的方法
- 按键位置可参考 easyscreen.ini 模板所示空余软键位置
- 对话框 M 可在入口关联的.com 文件中编写，也可在另外的.com 文件中编写



9.1.4 操作树

- 每个操作区（加工，参数...）都有固定的软键入口（.com，//S（START））
- 由此入口通过软键及方法定义灵活跳转到其他子界面（Mxx）
- 其他子界面同样可实现到其他界面的灵活跳转



9.1.5 空余按键

在操作区中“Run MyScreens”登入软键的允许位置有：

加工	参数	程序	程序管理器	诊断	调试
HS6	HS7	HS6 和 HS15	HS2-8,12-16 (空余位置)	HS7	HS7

10 方法

概述

在对话框和与对话框相关的软键栏中（软键栏由新设计的对话框调用），可以通过不同的事件（退出输入栏，按下软键）触发某些特定的动作。这些动作设计在方法中。

方法的基本编程按如下方式进行：

方法的开始标记	PRESS(HS1)
功能...	LM...
改变变量属性...	VAR1.ST=...
计算变量...	VAR2=VAR3+VAR4
方法的结束标记	END_PRESS

10.1 ACCESSLEVEL（访问等级）

当界面更改当前访问等级时，就会运行 ACCESSLEVEL 方法。

编程	ACCESSLEVEL ... END_ACCESSLEVEL
变量	S_ALEVEL（int 型，记录当前的访问等级 1-7。1=制造商，7=钥匙开关 3）
举例	ACCESSLEVEL SWITCH(S_ALEVEL) CASE 1 VARA0.ST="制造商" CASE 2 VARA0.ST="服务" CASE 3 VARA0.ST="用户" END_ACCESSLEVEL

10.2 CHANGE（变量值改变）

当变量值已改变时运行 CHANGE 方法。

编程	CHANGE ... END_CHANGE
系统变量	无
举例	Library/001

分为指定变量的 CHANGE 方法和全局的 CHANGE 方法。

指定变量	当指定变量值改变时，运行 CHANGE 方法
示例	CHANGE(VAR1) ... END_CHANGE
全局	界面中改变任意变量值时，运行全局 CHANGE 方法
示例	CHANGE(LIST1) IF LIST1.VAL==0 VARA1.WR=2

	ELSE VARA1.WR=4 ENDIF
--	-----------------------------

10.3 CHANNEL (通道切换)

当界面通道切换时，就会运行 CHANNEL 方法。

编程	CHANNEL ... END_CHANNEL
系统变量	S_CHAN (int 型，记录当前的通道号)
举例	CHANNEL IF S_CHAN == 1 MX1.BC=5 MX1.FC=2 MX1.BC_ST=5 MX1.FC_ST=2 ELSE MX1.BC=10 MX1.FC=1 MX1.BC_ST=10 MX1.FC_ST=1 ENDIF END_CHANNEL

10.4 CONTROL (多个NCU切换时)

更改当前控制系统 (NCU) 时，即通常在 1:n 切换时，就会运行 CONTROL 方法。

编程	CONTROL ... END_CONTROL
系统变量	S_CONTROL (int 型，记录当前的 NCU 号)
举例	CONTROL DLGL("你当前的控制系统为 NCU"<< S_CONTROL) END_CONTROL

10.5 FOCUS (焦点定位)

当对话框中聚焦 (光标) 定位在变量栏上时，运行 FOCUS 方法。

编程	FOCUS ... END_FOCUS
系统变量	FOC (string 型，当前变量名)
举例	FOCUS(FOC) SWITCH(FOC) CASE "AA" VARA0.ST="AA = "<<AA.VAL CASE "BB" VARA0.ST="BB = "<<BB.VAL CASE "CC" VARA0.ST="CC = "<<CC.VAL CASE "DD" VARA0.ST="DD = "<<DD.VAL CASE "EE"

	<pre> VARA0.ST="EE = "<<EE.VAL CASE "FF" VARA0.ST="FF = "<<FF.VAL CASE "GG" VARA0.ST="GG = "<<GG.VAL END_SWITCH END_FOCUS </pre>
--	---

注意

- 带有输入模式 wr = 0 和 wr = 4 的变量以及辅助变量无焦点

10.6 LANGUAGE (语言切换)

更改当前语言时，就会运行 LANGUAGE 方法。

编程	<pre> LANGUAGE ... END_LANGUAGE </pre>
系统变量	S_LANG (int 型，记录当前的 NCU 号)
举例	<pre> LANGUAGE SWITCH(S_LANG) CASE "chs" LIST1.st="颜色" CASE "eng" LIST1.st="COLOR" END_SWITCH VARA0="S_LANG = "<<S_LANG END_LANGUAGE </pre>

10.7 LOAD (装载对话框)

在已编译变量定义和软键定义 (DEF Var1= ..., HS1= ...) 后，对话框显示前，运行 LOAD 方法。

编程	<pre> LOAD ... END_LOAD </pre>
系统变量	无
举例	<pre> LOAD VAR1=10 VAR3=REG[0] END_LOAD </pre>

可装载变量初始值，软键文本，线，矩形等图形。

10.8 UNLOAD (卸载对话框)

在卸载对话框前 (退出界面时)，运行 UNLOAD 方法。

编程	<pre> UNLOAD ... END_UNLOAD </pre>
系统变量	无
举例	<pre> UNLOAD VAR2=0 REG[0]=VAR3 </pre>

	END_UNLOAD
--	------------

10.9 OUTPUT（输出加工程序代码）

当调用功能"GC"时，运行OUTPUT方法。

编程	OUTPUT(名称) ... END_OUTPUT
系统变量	无

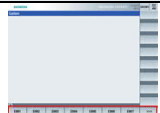
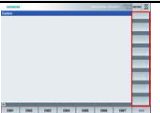
示例	完整示例	OUTPUT(NC1.1) VARA1","VARA2","VARA3 """"VARA3"""" "cyc("VARA1","VARA2","VARA3")" "cyc("VARA1","VARA2","""VARA3""")" END_OUTPUT	CHAN1 NC/MPF/TEST 0, 0, ss "ss" cyc(0, 0, ss) cyc(0, 0, "ss")
变量值输出	无引号	VARA1","VARA2","VARA3 或 ""VARA1","VARA2.VAL","VARA3""	0, 0, ss
常量字符串	1*双引号 "cyc("	"cyc("VARA1","VARA2","""VARA3""")"	cyc(0, 0, "ss")
变量字符串	3*双引号 """"VARA3"""" """"VARA3""""	""""VARA3"""" "cyc("VARA1","VARA2","""VARA3""")"	"ss" cyc(0, 0, "ss")

注意
- OUTPUT 方法不允许包含行号和隐藏标记 - 在零件程序中通过编辑器更改会影响用户循环界面反编译

10.10 PRESS（按下软键）

当已按下相应的软键时，运行PRESS方法。

编程	PRESS(软键名称) ... END_PRESS
系统变量	无
举例	PRESS(HS1) LM("Mask1") END_PRESS

软键名称	软键含义	对应图标	软键名称	软键含义	对应图标
HS1-HS8	水平软键栏		VS1-VS8	竖直软键栏	
RECALL	返回键		SL	光标向左	
ENTER	回车键		SR	光标向右	
TOGGLE	选择键		SU	光标向上	

PU	向上翻页			SD	光标向下	
PD	向下翻页					

注意						
- 对输入模式为 WR3 或 WR5 的变量，当按下回车(Enter)键时，会调用 PRESS(ENTER)方法 - 与当前处于焦点下的变量无关，只要按下转换键就会调用 PRESS(TOGGLE)方法						

10.11 RESOLUTION (更改分辨率)

当更改当前分辨率时，即通常在 TCU 切换时，就会运行 RESOLUTION 方法。

编程	RESOLUTION ... END_RESOLUTION
系统变量	S_RESX S_RESY
举例	RESOLUTION DLGL("当前分辨率为:"<< S_RESX<<"x"<< S_RESY END_RESOLUTION

10.12 SUSPEND (切换区域中断时)

当区域切换中断了屏幕（未卸载）时，就会调用 SUSPEND 方法。

编程	SUSPEND ... END_SUSPEND
系统变量	无
举例	SUSPEND WNP("\$R[1]",100) END_SUSPEND

10.13 RESUME (切换区域恢复时)

当返回中断界面时，就会调用 RESUME 方法。

编程	RESUME ... END_RESUME
系统变量	无
举例	RESUME WNP("\$R[1]",1) END_RESUME

11 功能

如下句法描述中，[]代表可选项，在需要时使用。

11.1 读写驱动,NC,PLC

功能名称	功能说明	示例	句法
RDOP	读驱动参数	MyVar=RDOP("SERVO_3.3:2","35")	RDOP ("驱动对象的名称","参数号")
WDOP	写驱动参数	WDOP("SERVO_3.3:2","105",1)	WDOP ("驱动对象的名称","参数号","值")
MRDOP	读一个驱动对象的多个驱动参数	MRDOP("SERVO_3.3:2","35*80",10)	MRDOP ("驱动对象的名称","参数号"*"参数号"*...,寄存器索引)
MRNP	多次读取 NC PLC	MRNP("\$R[0]*\$R[1]*\$R[2]*\$R[3]",1)	MRNP("变量名称 1*变量名称 2[*...]", 寄存器索引)
RNP	读 NC PLC 变量	VAR2=RNP("\$AA_IN[2]")	RNP ("系统或者用户变量")
WNP	写 NC PLC 变量	WNP("DB20.DBB1",1)	WNP ("系统或者用户变量",值)

11.2 块，数组，表格定义

块，数组，表格定义			
功能名称	功能说明	示例	句法
CALL	调用子程序	CALL("MY_UP1")	CALL("子程序名称")
SUB	定义子程序	SUB(MY_UP1) ... END_SUB	SUB(子程序名称) 子程序内容 END_SUB
//B	定义块	//B(PROG1) SUB(MY_UP1) ... END_SUB //END	//B(块名称) SUB(子程序名称) ... END_SUB //END
//A	定义数组	//A(螺纹)；高度/螺距/底直径 (0.3 / 0.075 / 0.202) (0.4 / 0.1 / 0.270) (0.5 / 0.125 / 0.338) (0.6 / 0.15 / 0.406) (0.8 / 0.2 / 0.540) //END	数组通过标记 //A 定义并通过 //END 结束 //A(名称) (a/b...) (c/d...) ... //END
//G	定义表格	DEF MyGridVar=(R/% MyGrid1//WR2////100,,351,100) LOAD LG("MyGrid1","MyGridVar","mygrids.com") END_LOAD //G(MyGrid1/0/5) (I///,"MyGrid1"/wr1//1"/80/1) (R3///"LongText1","R1-R4"/wr2//"\$R[1]"/80/1) (IBB///"LongText2","M2.2-M2.5"/wr2//"\$M2.2"/80/1) (R3///"LongText3","R9,R11,R13,R15"/wr2//"\$R[9]"/110/2) //END	//G (表格名称/表格类型/行数 / [固定行属性],[固定列属性]) (类型/极限值/空/长文本,列标题/属性/帮助图形/系统或用户变量/列宽/偏移 1、偏移 2、偏移 3) //END
()	表格内-定义列	(R3///"LongText1","R1-R4"/wr2//"\$R[1]"/80/1)	(类型/极限值/空/长文本,列标题/属性/帮助图形/系统或用户变量/列宽/偏移 1、偏移 2、偏移 3)
CVAR	检查变量是否有有效值	IF CVAR == TRUE ... ENDIF	CVAR;检查所有变量

1/TRUE=有有效值 0/FALSE=无有效值	IF CVAR("VAR1", "VAR2") == TRUE ... ENDIF	CVAR(变量名称,变量名称...); 检查指定变量,最多29 个
-----------------------------	---	-----------------------------------

11.3 程序文件

程序文件			
功能名称	功能说明	示例	句法
CP	复制/粘贴程序	CP("//NC/MPF.DIR/HOHO.MPF", "//NC/MPF.DIR/ASLAN.MPF", VAR1)	CP("源文件", "目标文件", 返回值); 路径: NC,HMI
DP	删除程序	DP("//NC/CMA.DIR/MYPROG.SPF", VAR1)	DP("文件", 返回值); 路径: NC,HMI
EP	检查程序是否存在	EP("//NC/WKS.DIR/TEST.WPD/XYZ.MPF", VAR1)	EP("文件", 返回值); 路径: NC,HMI
MP	移动程序	MP("CF_CARD:/mpf.dir/myprog.mpf", "//NC/MPF.DIR/123.MPF", VAR1)	MP("源文件", "目标文件", 返回值); 路径: NC,HMI
SP	选择程序	SP("//NC/MPF.DIR/MYPROG.MPF", VAR1)	SP("文件", 返回值); 路径: NC
示例	DEF VAR2="//NC/MPF.DIR/HOHO.MPF" CP(VAR2, "CF_CARD:/wks.dir/WST1.WPD/MESS.MPF", VAR1)		路径为变量形式,从零件程序复制到用户 CF 卡

11.4 文件读写

文件读写			
处理的文件不得位于 NC 文件系统中, 只能位于 HMI 中			
功能名称	功能说明	示例	句法
RDFILE	从文件中读关键字数值	MyVar = RDFILE("C:/tmp/myfile.ini", "MyData", "MyName")	RDFILE("文件名 + 路径", "段", "关键字")
WRFILE	将关键字数值写入文件	WRFILE(VARS, "C:/tmp/myfile.ini", "MySession", "NrOfSessions")	WRFILE(数值, "文件名 + 路径", "段", "关键字")
RDLINEFILE	从文件中读取一行	MyVar = RDLINEFILE("C:/tmp/myfile.mpf", 4)	RDFILE("文件名 + 路径", 行编号); 行编号从 0 开始
WRLINEFILE	在文件末尾写一行	WRLINEFILE("F100 X" << VARX << " Y" << VARY, "C:/tmp/mypp.mpf")	WRLINEFILE(内容, "文件名 + 路径")

11.5 列表框操作

列表框操作			
	DEF VAR_AC = (I/* 0="Off", 1="On"/1, "Switch"/WR2)		定义列表框 VAR_AC: 有 0,1 两个列表
功能名称	功能说明	示例	句法
LISTADDITEM	在列表末尾添加一个元素	LISTADDITEM("VAR_AC", -1, """"Undefined""")	LISTADDITEM("变量名", ItemValue[, ItemDispValue])
LISTINSERTITEM	在指定位置添加列表元素	LISTINSERTITEM("VAR_AC", 2, 99, """"Maybe""")	LISTINSERTITEM("变量名", 位置, ItemValue[, itemDispValue])
LISTDELETEITEM	删除某个列表元素	LISTDELETEITEM("VAR_AC", 1)	LISTDELETEITEM("变量名", 位置)
LISTCOUNT	获取当前列表元素总数	REG[10]=LISTCOUNT("VAR_AC")	LISTCOUNT("变量名")

LISTCLEAR	删除整个列表	LISTCLEAR("VAR_AC")	LISTCLEAR("变量名")
DLGL	对话框行文本显示	DLGL("值过大！")	DLGL("字符串")
DEBUG	调试日志	DEBUG("Value of ""Var1"": " << Var1)	DEBUG("字符串")
EXIT	退出对话框	EXIT EXIT(5,,7)	EXIT EXIT(返回值 1, 返回值 2, ...)
EXITLS	退出软键栏	EXITLS("SK_BAR_1", "AEDITOR.COM")	EXITLS("软键栏"[, "路径名"])
EVAL	获取变量的值	REG[7] = EVAL("VAR"<<REG[5])	EVAL("字符串")
GC	生成 NC 代码	GC("CODE1", "\MPF.DIR\MESSEN. MPF") GC("CODE1") GC("CODE1", "\MPF.DIR\MESSEN. MPF",0,0) GC("CODE1", "\MPF.DIR\MESSEN. MPF",1,1)	GC("名称"[, "目标文件"][, Opt], [Append]) ;功能 GC 仅在 NC 中可用。请使用功能 CP 或 MP 来移动文件 ;Opt: 0=带反编译注释; 1=无反编译注释 ;Append: 0=删除旧内容, 1=在开始处写入, 2=在结束处写入

11.6 口令功能

口令功能			
功能名称	功能说明	示例	句法
HMI_LOGIN	口令登入	REG[0] = HMI_LOGIN("CUSTOMER")	HMI_LOGIN("口令")
HMI_LOGOFF	删除当前口令	REG[0] = HMI_LOGOFF	HMI_LOGOFF
HMI_SETPASSWD	更改口令	REG[0] = HMI_SETPASSWD(3, "MYPWD")	HMI_SETPASSWD(访问等级数值, "口令")

11.7 装载功能

装载功能			
功能名称	功能说明	示例	句法
LA	装载数组	LA("ARR5", "arrayext.com")	LA("名称" [, "文件"])
LB	装载块	LB("PROG2", "XY.COM")	LB("块名称" [, "文件"])
LM	装载屏幕窗口	LM("Mask2")	LM("名称" [, "文件"] [, MSx [, VARx]])
		LM("Mask2", "CFI.COM", 1)	MSx: 0=退出并装载新对话框; 1=中断并装载新对话框;
		LM("Mask2", "CFI.COM", 1, POSX, POSY, DIAM)	VARx: 传送变量到新对话框, 最多 20 个, 逗号隔开
LS	装载软键栏	LS("SK_BAR_1")	LS("名称" [, "文件"] [, 合并])
		LS("SK_BAR_1", "AEDITOR.COM", 1)	合并: 0=清除所有软键, 载入新软键; 1=仅将新软键覆盖现有软键
LG	装载表格	LG("MyGrid1", "MyGridVar", "mygrids.com")	LG ("栅格名称", "变量名称" [, "文件名称"])

11.8 字符串功能

字符串功能			
功能名称	功能说明	示例	句法
LEN	获取字符串长度	VAR02=LEN("HELLO"),	LEN("字符串"或变量名称)

		VAR02=LEN(VAR01)	
INSTR	查找字符串中的一个字符	VAR02=INSTR(1,"/", "HALLO/WELT")	INSTR(开始, "字符串 1", "字符串 2" [,方向])
LEFT	由左提取部分字符串	VAR02=LEFT("HELLO/WELT" ,5)	LEFT("字符串", 长度)
RIGHT	由右提取部分字符串	VAR02=RIGHT("HELLO/WELT" ,4)	RIGHT("字符串", 长度)
MIDS	由字符串中间提取部分字符串	VAR02=MIDS("HELLO/WELT" ,4)	MIDS("字符串", 开始 [, 长度])
REPLACE	替换部分字符串	VAR02=REPLACE("HELLO/WELT", "E", "A", 0,10)	REPLACE ("字符串", "查找字符串", "替换字符串" [, 开始 [, 计数]])
STRCMP	字符串的比较	REG[0]=STRCMP("Hugo", "HUGO")	STRCMP("字符串 1", "字符串 2" [, FALSE]); 区分大小写
		REG[0]=STRCMP("Hugo", "HUGO", TRUE)	STRCMP("字符串 1", "字符串 2" [, TRUE]); 不区分大小写
STRINSERT	将字符串插入到另一个字符串中	REG[0]=STRINSERT("Hello!", "world", 5)	STRINSERT("字符串 1", "字符串 2", 插入位置)
STRREMOVE	从字符串中删除一个字符串	REG[0]=STRREMOVE("Hello world!", 5, 6)	STRREMOVE("字符串 1", 删除位置, 计数)
TRIMLEFT	从左删除首字符前的空格符	REG[0]=TRIMLEFT(" Hello!")	TRIMLEFT("字符串 1")
TRIMRIGHT	从右删除首字符前的空格符	REG[0]=TRIMRIGHT("Hello! ")	TRIMRIGHT("字符串 1")
FORMAT	插入带格式化标识的值或字符串	VAR2 = FORMAT("Hello %08b %.2f %s!", VAR1 +1, 87.654, "world")	FORMAT("带格式化标识的文本"[值 1] ... [,值 28])
		结果 = "Hello 01111100 87.65 world!"	%08b(8 位长度二进制显示), VAR1+1 %.2f(浮点数保留 2 位小数点), 87.654 %s(字符串格式显示), "world"

11.9 逻辑功能

逻辑功能			
功能名称	功能说明	示例	句法
SWITCH	多结果选择	SWITCH (VAR1) CASE "VarF" ... DEFAULT ... END_SWITCH	SWITCH (变量名) CASE 比较值 1 ... DEFAULT ... END_SWITCH
WHILE	WHILE 循环	DO ... LOOP_WHILE REG[0] < 10	DO ... LOOP_WHILE <继续执行的条件>
		DO_WHILE 10 > REG[0] ... LOOP	DO_WHILE <继续执行的条件> ... LOOP
UNTIL	UNTIL 循环	DO ... LOOP_UNTIL 0 <= REG[1]	DO ... LOOP_UNTIL <循环结束的条件>
		DO_UNTIL 0 <= REG[1] ... LOOP	DO_UNTIL <循环结束的条件> ... LOOP
IF	IF 循环(支持嵌套)	IF FOC == "Var1"	IF

		REG[1] = Var1 ENDIF	... ENDIF
		IF ERR == TRUE ; VAR5 = "数组存取出错" ELSE VAR5 = "一切正常" ; ENDIF	IF ... ELSE ... ENDIF

11.10 绘图功能

绘图功能			
功能名称	功能说明	示例	句法
LINE	绘制线段	LINE(220,80,230,100,3,3)	LINE(起点 x 坐标,起点 y 坐标,终点 x 坐标,终点 y 坐标,颜色,样式)
ELLIPSE	绘制圆/椭圆	ELLIPSE(200,100,100,50,3,4,1)	ELLIPSE(起点 x 坐标,起点 y 坐标,宽,高,边框颜色,填充颜色,边框样式)
RECT	绘制矩形	RECT(210,170,80,120,3,4,2)	RECT(起点 x 坐标,起点 y 坐标,宽,高,边框颜色,填充颜色,边框样式)
V_SEPARATOR	绘制垂直分割线	V_SEPARATOR(15,2,3,1)	V_SEPARATOR(x 位置,线条粗细,颜色,样式)
H_SEPARATOR	绘制水平分割线	H_SEPARATOR(15,2,3,1)	H_SEPARATOR(y 位置,线条粗细,颜色,样式)
CLEAR_BACKGROUND	清除图形元素	PRESS(HS1) CLEAR_BACKGROUND END_PRESS	CLEAR_BACKGROUND ;删除上述图形元素

11.11 其他功能

其他功能			
功能名称	功能说明	示例	句法
RESIZE_VAR_IO	改变变量的输入栏位置及尺寸	RESIZE_VAR_IO("MyVar1", 200, , 100)	RESIZE_VAR_IO(变量名,[X],[Y],[宽度],[高度])
RESIZE_VAR_TXT	改变变量的短文本位置及尺寸	RESIZE_VAR_TXT("MyVar1", 200, , 100)	RESIZE_VAR_TXT(变量名,[X],[Y],[宽度],[高度])
RETURN	子程序中中断并返回	SUB(MY_UP1) ... RETURN ... END_SUB	RETURN; 返回主程序调用位置
START_TIMER	计时器开始	START_TIMER("MyTimerSub", 1000)	START_TIMER("SUB 名称", 间隔) ; 间隔以 ms 为单位, 最小 100ms
STOP_TIMER	计时器结束	STOP_TIMER("MyTimerSub")	STOP_TIMER("SUB 名称")

11.12 参考手册

更多说明参见如下手册

编程 SINUMERIK 828D SINUMERIK Integrate Run MyScreens

<https://support.industry.siemens.com/cs/document/109760830/sinumerik-828d-sinumerik-integrate-run-myscreens?dti=0&lc=en-WW>

SINUMERIK 840D sl SINUMERIK Integrate Run MyScreens

<https://support.industry.siemens.com/cs/document/109760871/sinumerik-840d-sl-sinumerik-integrate-run-myscreens?dti=0&lc=en-WW>

12 附录

12.1 附录1 : easyscreen.ini 模板

```
#####
;# OEM / Easyscreen Start Softkeys #
#####
;#                                     #
;# (please copy the required line into section [STARTFILES] and #
;# delete the semicolon at the beginning of the line)      #
;#                                     #
#####

#####
;# AREA Machine      #
#####

;=====>
;< OEM Softkey on horizontal Machine-Automatic Menu      >
;< SOFTKEY position="6"                                   >
;=====>
;StartFile03 = area := AreaMachine, dialog := SIMachine, menu := SIMaAutoMenuHU, startfile := ma_auto.com

;=====>
;< OEM Softkey on horizontal Machine-Jog Menu            >
;< TECHNOLOGY ==> TURNING                                >
;< SOFTKEY position="6"                                   >
;=====>
;StartFile04 = area := AreaMachine, dialog := SIMachine, menu := SIMaJogTurnMenuHU, startfile := ma_jog.com

;=====>
;< OEM Softkey on horizontal Machine-Jog Menu            >
;< TECHNOLOGY ==> MILLING                                >
;< SOFTKEY position="6"                                   >
;=====>
;StartFile05 = area := AreaMachine, dialog := SIMachine, menu := SIMaJogMillMenuHU, startfile := ma_jog.com

;=====>
;< OEM Softkey on horizontal Machine-Jog Menu            >
;< TECHNOLOGY ==> UNIVERSAL                              >
;< SOFTKEY position="6"                                   >
;=====>
;StartFile06 = area := AreaMachine, dialog := SIMachine, menu := SIMaJogUniversalMenuHU, startfile := ma_jog.com

;=====>
;< OEM Softkey on horizontal Machine-MDI Menu            >
;< SOFTKEY position="6"                                   >
;=====>
;StartFile07 = area := AreaMachine, dialog := SIMachine, menu := SIMaMdaMenuHU, startfile := ma_mda.com

;=====>
;< OEM Softkey on horizontal Manual Machine Menu >
;< TECHNOLOGY ==> TURNING >
;< SOFTKEY position="3" (extended horizontal softkey bar)>
;=====>
;StartFile08 = area := AreaMachine, dialog := SIMachine, menu := SIMaJogManualTurnMenuHU, startfile := xxxx.com

;=====>
;< OEM Softkey on horizontal Manual Machine Menu >
;< TECHNOLOGY ==> MILLING>
;< SOFTKEY position="3" (extended horizontal softkey bar)>
;=====>
;StartFile09 = area := AreaMachine, dialog := SIMachine, menu := SIMaJogManualMillMenuHU, startfile := xxxxx.com

;=====>
;< OEM Softkey on vertical menu workpiece zero >
```

```

;< TECHNOLOGY ==> CIRCULAR GRINDING >
;< SOFTKEY position="2-7"
>=====
;StartFile10 = area := AreaMachine, dialog := SIMachine, screen := SIMaJogMeTurnEdgeEasyScreen, menu :=
SIMaJogMeTurnEdgeEasyScreenMenuVR, startfile := xxx.com

;<=====
;< OEM Softkey on vertical menu measure tool >
;< TECHNOLOGY ==> CIRCULAR GRINDING >
;< SOFTKEY position="7"
>=====
;StartFile11 = area := AreaMachine, dialog := SIMachine, screen := SIMaJogMeToolCircularGrindEasyScreen, menu :=
SIMaJogMeToolCircularGrindEasyScreenMenuVR, startfile := xxx.com

;<=====
;< OEM Softkey on horizontal machine-Jog Menu >
;< TECHNOLOGY ==> CIRCULAR GRINDING >
;< SOFTKEY position="6" >
>=====
;StartFile12 = area := AreaMachine, dialog := SIMachine, menu := SIMaJogCircularGrindMenuHU, startfile := xxx.com

;<=====
;< OEM Softkey on vertical menu workpiece zero >
;< TECHNOLOGY ==> SURFACE GRINDING >
;< SOFTKEY position="2-7"
>=====
;StartFile13 = area := AreaMachine, dialog := SIMachine, menu := SIMaJogMeSurfGrindEdge1EasyScreenMenuVR,
startfile := xxx.com

;<=====
;< OEM Softkey on vertical menu measure tool >
;< TECHNOLOGY ==> SURFACE GRINDING >
;< SOFTKEY position="2-7"
>=====
;StartFile14 = area := AreaMachine, dialog := SIMachine, screen := SIMaJogMeToolSurfaceGrindEasyScreen, menu :=
SIMaJogMeToolSurfaceGrindEasyScreenMenuVR, startfile := xxx.com

;<=====
;< OEM Softkey on horizontal machine-Jog Menu >
;< TECHNOLOGY ==> SURFACE GRINDING >
;< SOFTKEY position="6" >
>=====
;StartFile15 = area := AreaMachine, dialog := SIMachine, menu := SIMaJogSurfaceGrindMenuHU, startfile := xxx.com

#####
;# AREA Parameter #
#####

;<=====
;< OEM Softkey on horizontal Parameter Menu >
;< SOFTKEY position="7" >
>=====
;StartFile10 = area := AreaParameter, dialog := SIParameter, menu := SIPaMenuHU, startfile := param.com

;<=====
;< OEM Softkey on vertical Menu 'Ctrl-Energy' >
;< 'Energy analysis' and 'Energy-sav. profile' >
;< SOFTKEY position="4" >
>=====
;StartFile11 = area := AreaParameter, dialog := SIParameter, menu := SINrgDisplayVr, startfile := param.com
;StartFile12 = area := AreaParameter, dialog := SIParameter, menu := SINrgControlVr, startfile := param.com

#####
;# AREA Program Editor #
#####

```

```

;<=====>
;< OEM Softkey on horizontal Main Menu          >
;< SOFTKEY position="3,4,5"                      >
;<=====>
;StartFile13 = area := AreaProgramEdit, dialog := SIProgramEdit, menu := SIStepEditorMenuHU, startfile :=
aeditor.com

;<=====>
;< OEM Softkey on horizontal Main Menu          >
;< TECHNOLOGY ==> MILLING                      >
;< SOFTKEY position="15"                        >
;<=====>
;StartFile14 = area := AreaProgramEdit, dialog := SIProgramEdit, menu := SIStepStdMillMenuHU, startfile :=
aeditor.com

;<=====>
;< OEM Softkey on horizontal Main Menu          >
;< TECHNOLOGY ==> TURNING                      >
;< SOFTKEY position="15"                        >
;<=====>
;StartFile15 = area := AreaProgramEdit, dialog := SIProgramEdit, menu := SIStepStdTurnMenuHU, startfile :=
aeditor.com

;<=====>
;< OEM Softkey on horizontal Main Menu          >
;< OPTION ==> SHOPMILL                        >
;< SOFTKEY position="15"                        >
;<=====>
;StartFile16 = area := AreaProgramEdit, dialog := SIProgramEdit, menu := SIStepShopMillMenuHU, startfile :=
aeditor.com

;<=====>
;< OEM Softkey on horizontal Main Menu          >
;< OPTION ==> SHOPTURN                        >
;< SOFTKEY position="15"                        >
;<=====>
;StartFile17 = area := AreaProgramEdit, dialog := SIProgramEdit, menu := SIStepShopTurnMenuHU, startfile :=
aeditor.com

;<=====>
;< OEM Softkey on vertical Menu 'Drilling (Drill.)' >
;< OPTION ==> SHOPMILL                        >
;< SOFTKEY position="6"                        >
;<=====>
;StartFile18 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepShopMillDrillScreen, menu :=
SIStepShopMillDrillMenuVR, startfile := aeditor.com

;<=====>
;< OEM Softkey on vertical Menu 'Milling (Mill.)' >
;< OPTION ==> SHOPMILL                        >
;< SOFTKEY position="7"                        >
;<=====>
;StartFile19 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepShopMillMillScreen, menu :=
SIStepShopMillMillMenuVR, startfile := aeditor.com

;<=====>
;< OEM Softkey on vertical Menu 'Contour Milling (Cont. Mill.)' >
;< OPTION ==> SHOPMILL                        >
;< SOFTKEY position="15" (2. vertical menu)    >
;<=====>
;StartFile20 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepShopMillFciMillScreen, menu :=
SIStepShopMillFciMillMenuVR, startfile := aeditor.com

```

```

;<=====>
;< OEM Softkey on vertical Menu 'Miscellaneous (Misc.)'      >
;< OPTION ==> SHOPMILL                                     >
;< SOFTKEY position="15" (2. vertical menu)                >
;<=====>
;StartFile21 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepShopMillMiscScreen, menu :=
SIStepShopMillMiscMenuVR, startfile := aeditor.com

;<=====>
;< OEM Softkey on vertical Menu 'Drilling (Drill.)'          >
;< OPTION ==> SHOPTURN                                     >
;< SOFTKEY position="6"                                     >
;<=====>
;StartFile22 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepShopTurnDrillScreen, menu :=
SIStepShopTurnDrillMenuVR, startfile := aeditor.com

;<=====>
;< OEM Softkey on vertical Menu 'Turning (Turn.)'            >
;< OPTION ==> SHOPTURN                                     >
;< SOFTKEY position="7"                                     >
;<=====>
;StartFile23 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepShopTurnTurnScreen, menu :=
SIStepShopTurnTurnMenuVR, startfile := aeditor.com

;<=====>
;< OEM Softkey on vertical Menu 'Countour Turning (Cont. Turn.)' >
;< OPTION ==> SHOPTURN                                     >
;< SOFTKEY position="15" (2. vertical menu)                >
;<=====>
;StartFile24 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepShopTurnFciTurnScreen, menu :=
SIStepShopTurnFciTurnMenuVR, startfile := aeditor.com

;<=====>
;< OEM Softkey on vertical Menu 'Milling (Mill.)'            >
;< OPTION ==> SHOPTURN                                     >
;< SOFTKEY position="7"                                     >
;<=====>
;StartFile25 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepShopTurnMillScreen, menu :=
SIStepShopTurnMillMenuVR, startfile := aeditor.com

;<=====>
;< OEM Softkey on vertical Menu 'Miscellaneous (Misc.)'      >
;< OPTION ==> SHOPTURN                                     >
;< SOFTKEY position="15" (2. vertical menu)                >
;<=====>
;StartFile26 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepShopTurnMiscScreen, menu :=
SIStepShopTurnMiscMenuVR, startfile := aeditor.com

;<=====>
;< OEM Softkey on vertical Menu 'Miscellaneous (Misc.)'      >
;< TECHNOLOGY ==> CIRCULAR GRINDING                        >
;<=====>
;StartFile04 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepCircularGrindingMiscScreen,
menu := SIStepCircularGrindingMiscMenuVR, startfile := xxx.com

;<=====>
;< OEM Softkey on vertical Menu 'Contour '                  >
;< TECHNOLOGY ==> CIRCULAR GRINDING                        >
;<=====>
;StartFile05 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepCircularGrindingFciScreen, menu :=
SIStepCircularGrindingFciMenuVR, startfile := xxx.com

```

```

;<=====>
;< OEM Softkey on horizontal Main Menu          >
;< TECHNOLOGY ==> CIRCULAR GRINDING              >
;<=====>
;StartFile14 = area := AreaProgramEdit, dialog := SIProgramEdit, menu := SIStepCircularGrindingMenuHU, startfile :=
xxx.com

;<=====>
;< OEM Softkey on vertical Menu 'Miscellaneous (Misc.)' >
;< TECHNOLOGY ==> SURFACE GRINDING              >
;<=====>
;StartFile04 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepSurfaceGrindingMiscScreen,
menu := SIStepSurfaceGrindingMiscMenuVR, startfile := xxx.com

;<=====>
;< OEM Softkey on vertical Menu 'Contour '          >
;< TECHNOLOGY ==> SURFACE GRINDING              >
;<=====>
;StartFile05 = area := AreaProgramEdit, dialog := SIProgramEdit, screen := SIStepSurfaceGrindingFciScreen, menu :=
SIStepSurfaceGrindingFciMenuVR, startfile := xxx.com

;<=====>
;< OEM Softkey on horizontal Main Menu          >
;< TECHNOLOGY ==> SURFACE GRINDING              >
;<=====>
;StartFile14 = area := AreaProgramEdit, dialog := SIProgramEdit, menu := SIStepSurfaceGrindingMenuHU, startfile :=
tail_man.com

#####
;# AREA ProgramManager #
#####

;<=====>
;< OEM Softkey on first horizontal Main Menu      >
;< SOFTKEY position="7"                          >
;<=====>
;StartFile27 = area := AreaProgramManager, dialog := SIPmDialog, menu := hu_global, startfile := progman.com

#####
;# AREA Diagnosis #
#####

;<=====>
;< OEM Softkey on first horizontal Main Menu      >
;< SOFTKEY position="7"                          >
;<=====>
;StartFile28 = area := AreaDiagnosis, dialog:= SIdgDialog, menu := DgGlobalHu, startfile := diag.com

#####
;# AREA StartUp #
#####

;<=====>
;< OEM Softkey on first horizontal Main Menu      >
;< SOFTKEY position="7"                          >
;<=====>
;StartFile29 = area := AreaStartup, dialog := SIsuDialog, menu := SIsuMainScreenMenuHu, startfile := startup.com

#####

```

```

;# AREA Custom      #
;#####

;=====
;< All SOFTKEY position is free      >
;=====
;StartFile30 = area := Custom, dialog := SIEsCustomDialog, startfile := custom.com

```

12.2 附录2：颜色表

对于单元（文本、输入栏、背景等等）可以从 0 到 133 号的颜色中快速的选择一种。

索引	图标	颜色	描述
1		黑色	
2		桔黄色	
3		深绿色	
4		浅灰	
5		深灰色	
6		蓝色	
7		红色	
8		棕色	
9		黄色	
10		白色	
126		黑色	当前处于焦点下的输入/输出栏的文本颜色
127		浅橙色	当前处于焦点下的输入/输出栏的背景色
128		桔黄色	聚焦系统颜色
129		浅灰	背景颜色
130		蓝色	标题颜色（激活）
131		黑色	标题字体颜色（激活）
132		蓝绿色	转换栏的背景色
133		淡蓝色	列表框的背景色

除了以上预定义的颜色外，还可以使用 RGB 值(“ #RRGGBB ”)来给定颜色。



颜色值: #123456

实色效果	英文名称	R.G.B	16色	实色效果	英文名称	R.G.B	16色
	Snow	255 250 250	#FFFAFA		PaleTurquoise1	187 255 255	#BBFFFF
	GhostWhite	248 248 255	#F8F8FF		PaleTurquoise2	174 238 238	#AEEEEE
	WhiteSmoke	245 245 245	#F5F5F5		PaleTurquoise3	150 205 205	#96CDCD
	Gainsboro	220 220 220	#DCDCDC		PaleTurquoise4	102 139 139	#66B8B8
	FloralWhite	255 250 240	#FFFAF0		CadetBlue1	152 245 255	#98F5FF
	OldLace	253 245 230	#FDF5E6		CadetBlue2	142 229 238	#8EE5EE
	Linen	250 240 230	#FAF0E6		CadetBlue3	122 197 205	#7AC5CD
	AntiqueWhite	250 235 215	#FAEBD7		CadetBlue4	83 134 139	#53868B
	PapayaWhip	255 239 213	#FFEFD5		Turquoise1	0 245 255	#00F5FF

更多颜色值可查询 RGB 值相关文档。

<http://tool.oschina.net/commons?type=3>

常用颜色

实色效果	颜色	#RRGGBB	实色效果	颜色	#RRGGBB
	黑色	#000000		蓝色	#0000FF
	白色	#FFFFFF		绿色	#00FF00
	红色	#FF0000		黄色	#FFFF00
	青蓝色	#00FFFF			
	主题 0 背景颜色	#D7E1EB			
	主题 1 背景颜色	#E5ECF0			
界面主题					
显示机床数据 9112 操作界面设计 (主题) 设定 ; 9112=1 , 主题 1。9112=0 , 主题 0。					

12.3 附录3：运算符

数学运算符	说明	三角函数	说明
+	加法	SIN(x)	正弦 x
-	减法	COS(x)	余弦 x
*	乘法	TAN(x)	正切 x
/	除法	ATAN(x, y)	反正切 x/y
MOD	模数运算	SQRT(x)	平方根 x
()	括号	ABS(x)	绝对值 x
AND	与运算	SDEG(x)	换算为度数

OR	或运算
NOT	非运算
ROUND	带小数点的四舍五入

SRAD(x)	换算为弧度
CALC_ASIN(x)	反正弦 x
CALC_ACOS(x)	反余弦 x
SIN(x)	正弦 x
COS(x)	余弦 x

示例：

VAR1.VAL = 45 * (4 + 3) VAR2.VAL = SQRT(2)	VAR1 = 5,2328543 VAR2 = ROUND(VAR1, 4) 结果：VAR2 = 5.2339
---	---

数学函数	说明
CALC_CEIL(x)	计算下一个比 x 大的整数（向上取整）
CALC_FLOOR(x)	计算下一个比 x 小的整数（向下取整）
CALC_LOG(x)	计算以 e 为底的 x 的（自然）对数
CALC_LOG10(x)	计算以 10 为底的 x 的对数
CALC_POW(x, y)	计算 x 的 y 次方（x 的 y 次乘方）
CALC_MIN(x, y)	计算 x 与 y 哪个小（取最小值）
CALC_MAX(x, y)	计算 x 与 y 哪个大（取最大值）
CALC_CEIL(x)	计算下一个比 x 大的整数（向上取整）
CALC_FLOOR(x)	计算下一个比 x 小的整数（向下取整）

位运算符	说明
BOR	位方式 OR
BXOR	位方式 XOR
BAND	位方式 AND
BNOT	位方式 NOT
SHL	向左移位
SHR	向右移位

示例：

VAR01 = 16 SHL 2；结果 = 64

VAR03 = 16 SHR 2；结果 = 4

VAR02 = VAR02 SHL VAR04；VAR02 向左移位 VAR04

比较运算符	说明	比较运算符	说明
==	相等	<	小于
<>	不等	>=	大于等于
>	大于	<=	小于等于

示例：

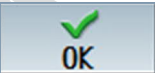
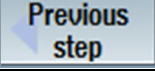
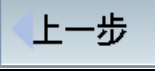
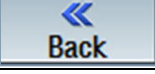
```
IF VAR1.VAL == 1
  VAR2.VAL = TRUE
ENDIF
```

其他代码	说明
RANDOM (下限值, 上限值)	取给定范围内的随机数值
PI	3.14159265358979323846 (圆周率)
FALSE	0
TRUE	1

示例：

```
REG[0] = RANDOM(-10,10) ; 可能结果 = -3
VAR1.VAL = PI
```

12.4 附录4：预定义按键

名称	按键 (英文)	按键 (中文)
SOFTKEY_OK		
SOFTKEY_CANCEL		
SOFTKEY_APPLY		
SOFTKEY_MORE		
SOFTKEY_BACK		
SOFTKEY_ASSISTANT_NEXT		
SOFTKEY_ASSISTANT_PREVIOUS		
SOFTKEY_NAV_BACK		

13 作者/联系人

DI MC MTS APC Chengfei

版本信息

版本	日期	修改内容
V1.0	6/19/2019	第一版

MTSAPC