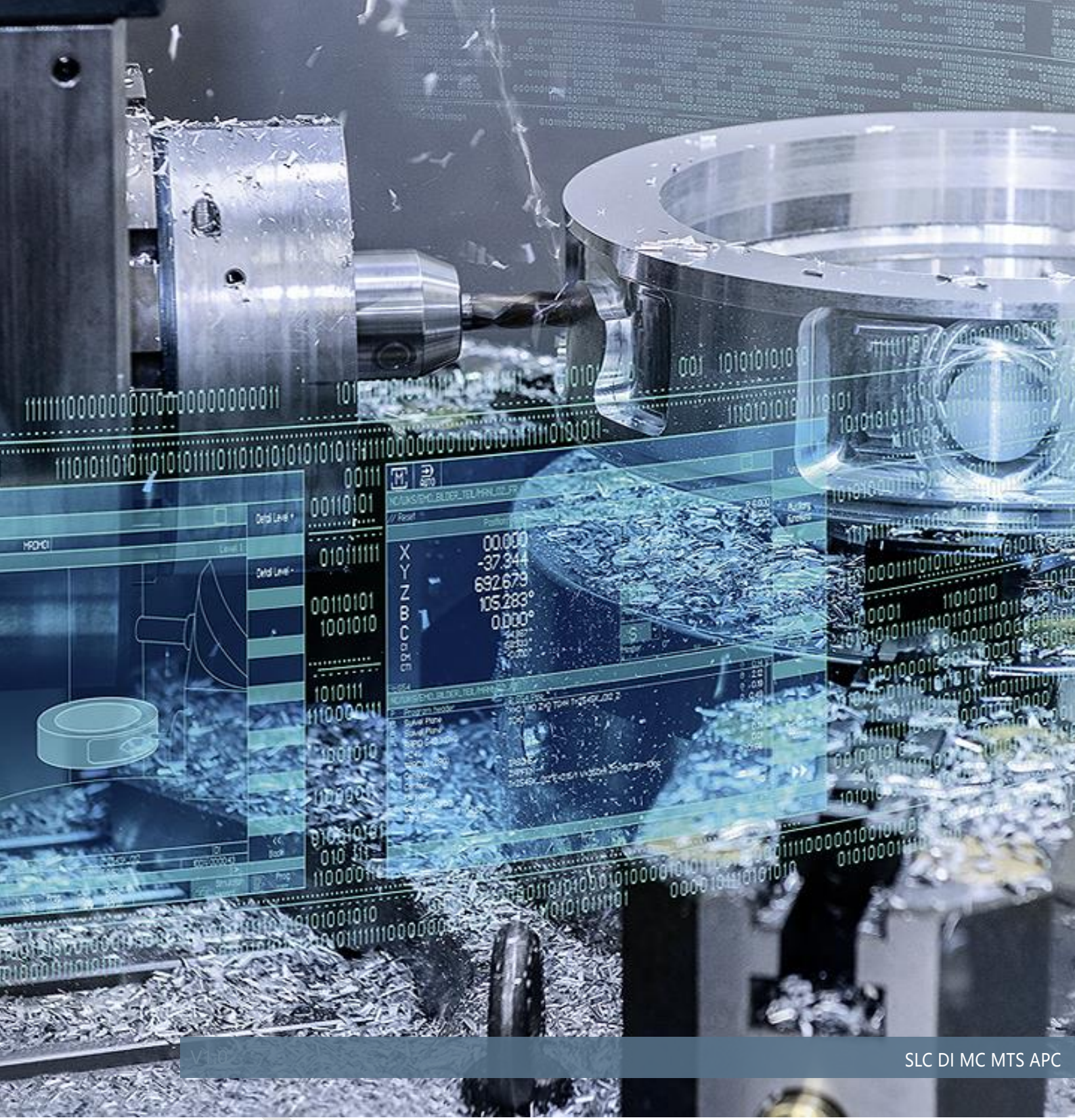


SINUMERIK OPERATE DISPLAY MANAGER 应用

SINUMERIK840D sl

目录

目录

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

借助 Display Manager 和 Display Configuration，可以缩小 SINUMERIK Operate，即标准操作界面不再全屏显示，以腾出空间显示其他应用程序。这些应用程序可以是标准的 Windows 应用程序（.EXE）或是属于 Display Manager 的小程序。这两种应用程序也可以由第三方借助 SINUMERIK Operate 的 OA Toolkit 套件开发而成，然后集成到一个 Display Configuration 中（下文称为显示配置）。

屏幕被划分为多个“框架”（frame）来显示各个应用程序。每个框架包含一个应用程序或者一个控制菜单，该菜单可控制在另一个框架中显示的一个应用程序。因此屏幕上会产生多种划分情况，构成不同的“画面”（display），程序运行时可在各画面之间来回切换。这些画面可以组成一种显示配置。可以创建多种显示配置，以便适应不同的屏幕分辨率条件。

一种显示配置只适用于一种屏幕分辨率或一种操作单位。该显示配置必须在系统配置文件 systemconfguration.ini 的段落 [displayConfigurations] 中完成。

2.1 激活 SINUMERIK Operate Display Manager

MD9114 $MM_SIDESCREEN

0 = Sidescreen 禁用

1 = Sidescreen 激活

2 = SINUMERIK Operate Display Manager 激活

2.2 分辨率

不同尺寸显示屏对应的分辨率，

19" 显示屏，分辨率：1366 x 768 像素

22'' 显示屏，分辨率：1920 x 1080 像素

24" 显示屏，分辨率：1920 x 1080 像素

2.3 SINUMERIK Operate 可显示的尺寸

SINUMERIK Operate 只能以特定尺寸显示。以下为支持的显示尺寸，

单位：像素

640x480

760x505

800x480

800x600

1024x768

1024x1014

1080x810

1280x768

1280x800

1280x1024

1300x768

1366x768

1440x1080
1454x1080
1854x1014
1854x1080
1920x1080

3 系统配置

文件名： systemconfguration.ini

3.1 存储路径

\oem\sinumerik\hmi\cfg，或
\user\sinumerik\hmi\cfg

3.2 [displayConfigurations]项

[displayConfigurations]
DC001 = res:=1366x768, file:=sidescreen_1366x768
DC002 = tcu:=TCU1, file:=tcu1_layout
DC003 = panelType:=OEM, res:=1366x768, file:=oem_layout
在本例中有三种显示配置。配置条目的名称始终以字符串 DC 开头，后面跟一个连续编号的数字。每个条目都有一个“file”属性，表示引用包含了显示配置的文件。文件名随意选择。和其他配置文件一样，系统也会在 siemens、addon、 oem 和 user 下的 cfg 目录下查找该文件。文件后缀名“.ini”无需输入。参数“res”、“tcu”和“panelType”描述了每一种显示配置的分辨率、操作单元和制造商版式。

- 说明

● 制造商版式的大小必须和参数化过的分辨率相符合，不然可能导致用户界面显示错误。
● 如果一个用户界面设置了多个显示配置，则优先的显示配置为“tcu” → “panelType” → “res”
● 如果没有任何一个显示配置合适，则用户界面全屏显示。

3.3 [dialogs]项

[dialogs]
定义调用的 A 型应用程序，即在 SlGfwSideScreenDialog 类基础之上开发的应用程序。

3.4 [processes]项

[processes]
定义调用的 B 型应用程序，即标准 Windows 应用程序（.EXE）。

3.5 [displayConfigurations]项

定义显示分辨率；
指定调用的画面配置文件，路径\oem\sinumerik\hmi\cfg，或\user\sinumerik\hmi\cfg 下的 ini 文件，如 dm_portrait.ini。

3.6 示例

；systemconfguration.ini
[dialogs]

DLG500=name:=PdfApp, implementation:=sldmpdfviewerapp.SlDmPdfViewerApp, process:=SlHmiHost1, preload:=false cmdline:="-pdfFile gestenbedienung.pdf -matteColor #3b4c58"
DLG502=name:=McpApp, implementation:=sldmsidescreenapp.SlSideScreenDialog, process:=SlHmiHost1, preload:=false, cmdline:="-sidescreen1 sldmmcppage.ini"
DLG503= name:=SlWidgetsApp, implementation:=sldmsidescreenapp.SlSideScreenDialog, process:=SlHmiHost1, preload:=false, cmdline:="-sidescreen1 sldmwidgets1.ini -sidescreen2 sldmwidgets2.ini -spacing 3"
DLG504=name:=KeyboardApp, implementation:=sldmvirtualkeyboardapp.SlDmVirtualKeyboardApp, process:=SlHmiHost1, preload:=false, cmdline:="-settingsFile sldm_keyboard.ini"

[processes]
PROC500=process:=ExeApp, cmdline:="C:\\Windows\\System32\\notepad.exe", oemframe:=true, windowname:="Untitled - Notepad", classname:="Notepad", deferred:=true

[displayConfigurations]
DC001=res:=1080x1920, file:=dm_portrait
；指定调用的画面配置文件，路径\oem\sinumerik\hmi\cfg，或\user\sinumerik\hmi\cfg 下的 ini 文件，如 dm_portrait.ini。

4 自定义显示配置

画面配置文件扩展名为 ini，如 dm_portrait.ini 等。

4.1 Displays（画面）

对于每种显示配置，可以在对应的配置文件下定义一幅或多幅画面。并为每幅画面命名并为它指定一系列框架（“frame”）。框架是一幅画面上划分出的、显示不同应用程序或菜单的矩形区域。

属性	含义
name	画面的名称，比如在切换画面时需要使用该名称。
frames	包含了多个需要显示在该画面上、用逗号隔开的框架，也就是说，该属性定义了画面的分隔情况。
enableMirrorMode (true|false)	启用（true）或关闭（false）画面的镜像模式 启用镜像模式时，setMirrorMode（true）指令会使该画面的所有框架水平镜像显示。关闭镜像模式时，setMirrorMode（true）指令不会影响框架的显示。 默认为"true"。
onStartup	包含了多个用分号隔开、在第一次显示画面时需要执行的动作。 允许的动作在下文说明。
onShow	包含了多个用分号隔开、在每次显示画面时都需要执行的动作。 允许的动作在下文说明。

4.1.1 示例

; dm_portrait.ini

[displays]

DISPLAY001=name:=d_main, frames:="Header, Menu_Left, Frame_Operate, Frame_Center, Frame_MCP, Menu_Bottom"

4.2 Frames（框架）

框架“frame”相当于一个显示应用程序或菜单的“容器”，可用于定义应用程序和菜单的显示位置和尺寸，单位为像素。另外，它还可以用于指定 Display Manager 启动后在其中显示的应用程序或菜单。各框架中要显示的应用程序可以在文件 systemconfiguration.ini 中规定。

如果希望在一个框架中显示 SINUMRIK Operate，作为应用程序可以输入“OPERATE”。

- 注意：SINUMERIK Operate 只能以特定尺寸显示

参考 1.3 SINUMERIK Operate 可显示的尺寸。

属性	含义
name	框架的名称，比如：该名称用于输入到画面的框架列表中。
x, y, width, height	框架的绝对位置 (x, y)、宽度和高度，单位：像素。
App	在首次显示该框架时，在其中显示的应用程序。
runableApps	在应用程序互换显示时可以在该框架中显示的一系列应用程序。
menu	在该框架中显示的菜单。 “app”中有输入时，该条目被忽略，无效。

4.2.1 示例

; dm_portrait.ini

[frames]

;Header

FRAME010=name:=Header, x:=0, y:=0, width:=1080, height:=86, menu:=mHeader

;Navigation links

FRAME020=name:=Menu_Left, x:=0, y:=86, width:=56, height:=768, menu:=mNavleft

;Frame mit Operate

FRAME030=name:=Frame_Operate, x:=56, y:=86, width:=1024, height:=768, app:=OPERATE

;Frame für versch. Apps

FRAME040=name:=Frame_Center, x:=0, y:=854, width:=1080, height:=768, app:=PdfApp

;Navigation unten

FRAME060=name:=Menu_Bottom, x:=0, y:=1622, width:=1080, height:=64, menu:=mNavbottom

;MCP

FRAME050=name:=Frame_MCP, x:=0, y:=1686, width:=1080, height:=234, app:=McpApp

;Keyboard-PopUp

FRAME055= name:=fKeyboardPopUp, x:=56, y:=1000, width:=824, height:=230

4.3 菜单

除了应用程序外，在一幅画面的框架中还可以显示菜单。这些菜单主要用于使某个应用程序在另一个框架中显示或者切换画面。菜单配置由四个段落组成：

- 菜单项定义（段落[menuitems]）
- 菜单项的外观和风格定义（段落 [menuitemstyle]）
- 菜单结构定义（段落[menus]）
- 菜单的外观和风格定义（段落[menustyle]）

4.3.1 段落[menuitems]

[menuitems]用于定义菜单项（即按键/按钮）。一个条目对应一个按键或按钮。[menuitems]的各个属性都可以定义。

[menuitems]有以下属性：

属性	含义
name	菜单项的唯一名称。
onClicked	操作该菜单项后触发的动作。允许的动作在“菜单动作”中说明。
Text	菜单项上显示的文字。此处可以用\n 或 %n 换行。
textID Text-ID	用于引用和语言相关的文本。如果系统读到该文本，该文本会取代“text”属性下输入的文本。
textContext	Text-ID 的上下文。该上下文也可以在设计菜单时指定，或在文件层级全局指定。但具体菜单项中的上下文指定有最高优先级，会覆写上两种方法中的指定，而菜单设计中的上下文指定会覆写文件层级的指定。
image	显示在菜单项上的图标的文件名。 显示按钮（Display Button）特例： Display Manager 会自动为用于切换画面的按钮生成一个图标。此处输入文件名时，输入需要切换到的目标画面的名称再加上后缀名“.auto”。随后画面的简图会作为图标显示。简图使用的线条宽度和颜色可以在属性 strokeWidth 和 strokeColor 下确定。 Display Button 既上没有文字，也没有 imagePressed 图标。
strokeWidth[1)]	Display Button 上显示的简图的线条宽度 默认属性：1
strokeColor[1)]	Display Button 上显示的简图的线条颜色，参见属性 image。 默认属性：Qt::lightGray
imagePressed	菜单项被按下后显示的图符的文件名。如果此处不作设置，系统会自动使用名为“<image>_activated”的一份文件，其中的 <image>是属性 image 中输入的名称。
accessLevel	显示菜单项需要具有的访问权限： System、Manufacturer、Service、Customer、Key_3、 Key_2、Key_1、Key_0 或者对应的数字（0-7） 默认值：Key_0 (7)
menuItemStyle	菜单项的外观和样式。 提示 所有下文说明的属性既可以设计具体菜单项时指定，也可以在统一设计菜单项样式时指定，但前者的优先级更高，会覆写后者。
textSize[2)]	字体大小

textAlignment[2)]	菜单项文字的对齐方式： Left、Right、Top、Bottom、TopLeft、TopRight、BottomLeft、BottomRight（类似于 Qt::Alignment）。 此处也可以十六进制（0x00）或十进制方式输入 Qt 标记字段的数值。 默认值：Qt::AlignCenter
imageAlignment[2)]	菜单项图标的对齐方式： Left、Right、Top、Bottom、TopLeft、TopRight、BottomLeft、BottomRight（类似于 Qt::Alignment）。 此处也可以十六进制（0x00）或十进制方式输入 Qt 标记字段的数值。 默认值：Qt::AlignCenter
width[2)]	菜单项的宽度，单位：像素。 默认值： 在排成一排的菜单布局中（row 型），总宽度减去菜单项之间的间隔后，所有菜单项均匀分布；在排成一列的菜单布局中（column 型），所有菜单项一样宽，等于总宽度渐去边距。 在网格型菜单布局中（grid 型），菜单项在高度和宽度上都均匀分布。
height[2)]	菜单项的高度，单位：像素。 默认值： 在排成一列的菜单布局中（column 型），总高度减去菜单项之间的间隔后，所有菜单项均匀分布；在排成一排的菜单布局中（row 型），所有菜单项一样高，等于总高度渐去边距。 在网格型菜单布局中（grid 型），菜单项在高度和宽度上都均匀分布。
color[2)]	菜单项的颜色。 参见"颜色定义"一节。 默认值：Qt::lightGray
pressedColor[2)]	菜单项被按下后的颜色。 参见"颜色定义"一节。 默认值：Qt::blue
textColor[2)]	菜单项文字的颜色。 参见"颜色定义"一节。 默认值：Qt::black
pressedTextColor[2)]	菜单项被按下后文字的颜色。 参见"颜色定义"一节。 默认值：Qt::white
borderWidth[2)]	菜单项边框的宽度，单位：像素。 默认值：1
borderColor[2)]	菜单项边框的颜色。 参见"颜色定义"一节。 默认值：Qt::darkGray
textRect[2)]	用于菜单项定位的矩形，即小窗口部件“Widget”坐标系，也就

	是说：菜单项的左上角为坐标 0,0，文字按该原点对齐。 输入格式为： x, y, width, height。比如： textRect:="0,10,40,30"或 textRect:=0/10/40/30 或 textRect:=0 10 40 30 注意，如果该属性中用逗号来分隔各元素，便需要加引号，因为逗号同时也是属性之间的分隔符。 默认值：矩形位于边框内部，当 borderWidth=0 时该矩形占据整个菜单项。
imageRect[2)]	用于菜单项定位的矩形，即小窗口部件“Widget”坐标系，也就是说：菜单项的左上角为坐标 0,0，图标按该原点对齐。 输入格式为： x, y, width, height。比如 imageRect:="0,10,40,30" 或 imageRect:=0/10/40/30 或 imageRect:=0 10 40 30 注意，如果该属性中用逗号来分隔各元素，便需要加引号，因为逗号同时也是属性之间的分隔符。 默认值：矩形位于边框内部，当 borderWidth=0 时该矩形占据整个菜单项。

- 说明：

1) 只针对 Display Button

2) 针对 menuItem 和 menuItemStyle

4.3.1.1 示例

```
；dm_portrait.ini
[menuitems]
MENUITEM001=name:=shortcut1, menuItemStyle:=misMenuOperate,
onClicked:="hidePopup(); sendCmd(OPERATE, AreaMachine); showApp(defaultFRAME,
OPERATE)", image:=dm_maschine.png, text:="Maschine"
MENUITEM002=name:=shortcut2, menuItemStyle:=misMenuOperate,
onClicked:="hidePopup(); sendCmd(OPERATE, AreaParameter, SlParameter, ToolList);
showApp(defaultFRAME, OPERATE)", image:=dm_tool_list_m.png, text:="Para-%nmeter"
MENUITEM003=name:=shortcut3, menuItemStyle:=misMenuOperate,
onClicked:="hidePopup(); sendCmd(OPERATE, AreaProgramEdit); showApp(defaultFRAME,
OPERATE)", image:=dm_programm.png, text:="Paramm%nEdit"
```

4.3.2 段落[menuitemstyles]

[menuitemstyles]可以统一定义菜单项的样式，即所有外观属性。各个外观属性可以在定义具体菜单项时在[menuItems]中再次修改。所有外观属性整合在一个段落中可以方便您的菜单项设计，因为通常菜单项很多，但外观都应保持一致。但如果您的确需要某个菜单项与众不同，也可以在[menuItems]中单独修改。

属性 menuItemStyle 的说明：见上文。

4.3.2.1 示例

```
；dm_portrait.ini
```

[menuitemstyles]
MENUITEMSTYLE001=name:=misMenuOperate, textSize:=13, textAlignment:=bottom, color:=37/53/63, textColor:=153/173/185, pressedColor:=52/87/140, pressedTextColor:=White, borderColor:=153/173/185, pressedBorderColor:=52/87/140, borderWidth:=1, spacing:=0, height:=80, width:=56
MENUITEMSTYLE002=name:=misMenuCenter, textSize:=15, textAlignment:=bottom, color:=37/53/63, textColor:=153/173/185, pressedColor:=52/87/140, pressedTextColor:=White, borderColor:=153/173/185, pressedBorderColor:=52/87/140, borderWidth:=1
MENUITEMSTYLE003= name:=misHeader, textSize:=16, color:=White, pressedColor:=White, borderWidth:=0

4.3.3 菜单[menus]

在段落[menus]中，您可以将一个或多个菜单项 menuItems 组合成一个菜单。具体是通过该段落中的属性 menuItems 实现的，在该属性下可以一一列出需要在一个菜单中显示的菜单项（按键/按钮）。该段落中提供的所有属性详见下表：

属性	含义
menuItems	列明菜单项。 各个菜单项之间用逗号隔开。 采用 grid 菜单布局时，需要指定菜单项在菜单中所处的位置： menuItem(row, col, rowSpan=1, colSpan=1) row 和 col 从 0 开始编号。 rowSpan 和 colSpan 可选择性指定，默认值是 1，类似于 QGridLayout::addWidget()。 采用 row 或 col 菜单布局时，还需要指定间距 spacing(pixel) 和 stretch(factor=0)： ● spacing 指定菜单项之间的间距，单位为像素，类似于 QBoxLayout::addSpacing()。 ● stretch 指定各菜单项之间可拉伸的间距，还可以为它选择性设置一个系数，类似于 QBoxLayout::addStretch()。
DefaultFrame	该菜单的菜单项控制的框架的名称，在 showApp 指令中该框架为目标，因此一个菜单项可以在多个菜单中使用。
textContext	菜单项文本的上下文，详见上文。
itemAccessLevel / accessLevel	显示菜单项需要具有的访问权限： System、Manufacturer、Service、Customer、Key_3、 Key_2、Key_1、Key_0 或者对应的数字（0-7） 默认值：Key_0 (7)
menuStyle	菜单样式。 指定所有外观属性。
	提示

	所有下文说明的属性既可以在[menus]中指定，也可以在[menustyle]中指定。但前者的优先级更高，会覆写后者。
layout[1)]	菜单项的对齐方式： row、col 或 grid。row 或 col 表示菜单项排成一行或排成一列。 grid 表示菜单项网格式排列。 默认属性：row
margin[1)]	表示菜单项和菜单外边缘之间的距离。 单位为像素，类似于 QBoxLayout::setMargin() 或 QGridLayout::setContentsMargin()；适用于所有 4 个方向 默认值：1
spacing[1)]	菜单项之间的间距。 单位为像素，类似于 QBoxLayout::setSpacing() 或 QGridLayout::setSpacing()。 默认值：1
color/ backgroundColor1)	菜单的颜色/背景色。 参见“颜色定义”一节。 默认值：Qt::gray

- 说明

1) 针对 menu 和 menuStyle

4.3.3.1 示例

```
；dm_portrait.ini
[menus]
MENU001=name:=mNavleft, menuStyle:=msNavOperate, items:="shortcut1, shortcut2, shortcut3, shortcut4, shortcut5, spacing(3)"
MENU002=name:=mNavbottom, menuStyle:=msNavCenter, items:="pdfbutton, widgetsbutton, exebutton"
MENU003=name:=mHeader, menuStyle:=msHeader, items:="miHeader, miKeyboard, miMirror"
```

4.3.4 段落[menustyles]

段落[menustyles]可以统一定义菜单的样式，即所有外观属性。各个外观属性可以在定义具体菜单时再次在[menus]中修改。

属性 menuStyle 的说明：见上文。

4.3.4.1 示例

```
；dm_portrait.ini
[menustyles]
MENUSTYLE001= name:=msNavOperate, color:=37/53/63, layout:=col, margin:=0, spacing:=0
MENUSTYLE002= name:=msNavCenter, color:=37/53/63, layout:=row, margin:=0, spacing:=0
MENUSTYLE003= name:=msHeader, color:=White, layout:=row, margin:=0, spacing:=0
```

4.4 菜单动作

在菜单项的属性 onClicked 中可以定义按下该菜单项后执行的动作。该属性可包含一条指令或包含多条用分号隔开、需要连续执行的指令。

可以执行以下指令：

指令	含义
showDisplay(displayname)	切换到名为 displayname 的画面上。 目标画面必须已在同一份显示配置文件中定义完毕。
showApp(framename, appname)	在名为 framename 的框架中显示名为 appname 的应用程序。 如果该应用程序已在同一画面的另一个框架中显示，则该程序从该框架移动到指令中的框架中。 可能的话，从 framename 中挤出的应用程序会转移到“变空”的框架中显示。这种情况就是所谓的“应用程序交换”。在该框架的属性 runableApps 中可以指定哪些应用程序可以转移到空置框架中显示。 在框架名称前还可以加一个画面名称，语法为 displayname.framename，以便修改不可见的画面中的框架占用情况。 除了 appname 外，也可以使用 framename.currentApp 或 displayname.framename.currentApp。 showApp() 会将显示在框架（framename 或 displayname.framename）中的应用程序移动到指定的框架中。
showPopup(framename, appname)	在名为 framename 的框架中以提示框的形式显示名为 appname 的应用程序。 框架 framename 不得属于某个画面，即不能在某个现有画面的属性 Frames 中指定。 showPopup() 会自动打开目标框架，将它叠加在当前的活动画面上。 **此处的“应用程序”只允许是在 SlGfwSideScreenDialog 类基础之上开发的应用程序。SINUMERIK Operate 和 OEM 框架应用程序可能无法作为弹出式程序显示。要将在 SlGfwSideScreenDialog 类基础之上开发的应用程序设计为弹出式程序时，需要将它的窗口指定为窗口，也就是说：在**

	windowFlags() 中置位位 **“Qt::WindowStaysOnTopHint”。** **一个时间点上始终只能打开一个弹出式窗口。**
hidePopup(appname)	再次隐藏用 showPopup() 或 togglePopup() 显示的应用程序。
togglePopup(framename, appname)	在名为 framename 的框架中以提示框的形式显示名为 appname 的应用程序。如果应用程序 appname 已经作为提示框显示，则会被隐藏。
showMenu(framename, menuname)	在名为 framename 的框架中显示名为 menuname 的菜单。
showPopupMenu(framename, menuname)	在名为 framename 的框架中以弹出菜单的形式 显示名为 menuname 的菜单
hidePopupMenu(menuname)	再次隐藏用 showPopupMenu togglePopupMenu() 显示的弹出菜单
togglePopupMenu(framename, menuname)	在名为 framename 的框架中以弹出菜单的形式 显示名为 menuname 的菜单 menuname 已经作为弹出菜单在 架中显示，则会被隐藏。
setMirrorMode(true|false)	启用（true）或关闭（false）镜像模式 在此处启用镜像模式时，在所有没有 enableMirrorMode(false) 关闭镜像模式的画面中，框架都呈水平镜像显示。也就是说：之 前显示在左边的框架现在都显示在右边，之前显示在右边的框架现在都显示在左边。框架的尺寸此时保持不变。
toggleMirrorMode()	切换镜像模式。 关闭启用的镜像模式或启用之前关闭的镜像模式。
sendCmd(appname, cmd1, ...)	向名为 appname 的应用程序发送一条指令。如果目标应用程序是 SINUMERIK Operate （appname="OPERATE"），便可以用这种方式选择 SINUMERIK Operate 的某个操作区域。 比如：sendCmd(OPERATE, AreaMachine) 操作区域（AreaMaschine）的名称位于文件 systemconfiguration.ini 的段落[areas]中。如果目标应用程序是一个在 SlGfwSideScreenDialog 类基础上开发的应用程序，则该应用程序的 onMessage()条目中会包含一条类型为 SL_GFW_MSG_DISPCONFIG_CMD 的指令。 指令 cmd1,..., cmd10 作为消息数据指定，也就是字符串格式，各个参数 cmd1 ... cmd10 之间用符号“|”分隔。 sendCmd() 无法将指令传给 Windows 应用程序（.EXE）!

4.4.1.1 示例

[menuitems]
MENUITEM001= name:=item1, onClicked:=
"showApp(frame1,OPERATE);showApp(frame2,myDialog)", text:="HMI"
；表示在当前画面的“frame1”中显示 SINUMERIK Operate，在“frame2”中显示应用程序“myDialog”。

MENUITEM003= name:=item3,
onClicked:="showApp(full.frame1,frame2.currentApp);showDisplay(fu
ll)", text:="Vollbild"
表示将当前显示的应用程序（currentApp）从“frame2”转移到画面“full”中的“frame1”显示，然后切换到画面“full”。

MENUITEM004= name:=item4, onClicked:="sendCmd(OPERATE,'AreaMachine')",
text:="Maschine"
表示选中 SINUMERIK Operate 中的操作区域“Maschine”。

MENUITEM005= name:=item5,
onClicked:="sendCmd(myDialog,'doFun','42')", text:="Maschine"
表示将包含数据“doFun|42”的消息 SL_GFW_MSG_DISPCONFIG_CMD 发送给应用程序“myDialog”，此程序要求使用 CreateMyHMI/3GL 开发。

4.5 应用程序

在显示配置中使用的应用程序必须在文件 systemconfiguration.ini 中指定。
应用程序分两种：

- A 型：在 SlGfwSideScreenDialog 类基础之上开发的应用程序。
- B 型：标准 Windows 应用程序（.EXE）

在系统启动时，Display Manager 会自动启动所有应用程序；在系统关闭时，它也会自动关闭所有应用程序。

A 型应用程序在文件 systemconfiguration.ini 的区段 [dialogs] 中的数字范围 500 - 999（OEM 范围）中声明。
B 型应用程序如同 OEM 框架应用程序一样处理，在段落 [processes] 中定义。为使用该类应用程序，另外还须在文件 C:\Program Files
(x86)\Siemens\MotionControl\compat\user\OEMFRAME.INI
中将参数 nPlacementMode 设为 3。

系统标配有以下应用程序

- PDF Viewer
- VNC Viewer
- 虚拟键盘

● 虚拟 MCP
● 显示窗口小部件的应用程序
● SINUMERIK Operate

4.5.1 PDF Viewer

在显示配置中进行各项指定时候应使用程序名称：SlPdfApp

默认定义：
DLG500=name:=PdfApp, implementation:=sldmpdfviewerapp.SlDmPdfViewerApp, process:=SlHmiHost1, preload:=false cmdline:="-pdfFile gestenbedienung.pdf -matteColor #3b4c58"

在参数“cmdline”中指定了需要显示的文件。两个参数的含义为：
● pdfFile:Pdf Viewer 中待显示的文件。该文件可以位于 appl 或 hlp 目录中。
● matteColor:Pdf Viewer 的背景色，颜色以十六进制代码指定。

在一段显示配置中可以定义 Pdf Viewer 的多个实例。其中的参数“name”包含了更多实例的名称；参数“cmdline”指定由该实例显示的文件，比如：
DLGxxx= name:=anotherPdf, implementation:=sldmpdfviewerapp.SlDmPdfViewerApp, process:=SlHmiHost1, preload:=false cmdline:="-pdfFile mydocument.pdf -matteColor #3b4c58"

4.5.1.1 示例

；systemconfiguration.ini
；调用应用程序 SlPdfApp，显示名为‘gestenbedienung.pdf’的文档
DLG500=name:=PdfApp, implementation:=sldmpdfviewerapp.SlDmPdfViewerApp, process:=SlHmiHost1, preload:=false cmdline:="-pdfFile gestenbedienung.pdf -matteColor #3b4c58"

4.5.2 VNC Viewer

在显示配置中进行各项指定时候应使用程序名称：SlVncApp

默认定义：
[dialogs]
DLG109= name:=SlVncApp, implementation:=sldmvncviewerapp.SlDmVncViewerApp, process:=SlHmiHost1, preload:=false, cmdline:="-configuration MyPC"

VNC Viewer 在文件 slvncconfig.ini 中设计。在参数“cmdline”中可以输入定义具体 VNC Viewer 实例所在段落的名称，作为变量“-configuration”的值。上面的例子为“MyPC”。

每个 VNC Viewer 实例都需要输入下列参数：
● host:VNC Server 所在主机的 IP 地址或 DNS 名称。
● port:Server 使用的 TCP 端口。

● Password （可选）:保证 VNC Server 访问安全的口令

4.5.2.1 示例

文件 slvncconfig.ini 中的定义：

[MyPC]
host=172.218.152.43
port=5900
password=geheim

4.5.3 虚拟键盘

在显示配置中进行各项指定时候应使用程序名称：SlKeyboardApp

默认定义：

[dialogs]
DLG112= name:=SlKeyboardApp,
implementation:=sldmvirtualkeyboardapp.SlDmVirtualKeyboardApp,
process:=SlHmiHost1, preload:=false, cmdline:="-settingsFile
sldm_keyboard.ini"

在一种显示配置内，该应用程序只允许一个实例中使用。

在标准配置中，为虚拟键盘预配置了三种尺寸（参见文件 sldm_keyboard.ini）：

[760x505]
Num= KeyboardLayout:="numpadlayout", x:=430, y:=270
Alpha= KeyboardLayout:="nclayout", x:=92, y:=50
KeyHeight=47
IME= x:=92, y:=270

[1024x242]
Num= KeyboardLayout:="numpadlayout", x:=784, y:=25
Alpha= KeyboardLayout:="nclayout", x:=206, y:=25
KeyHeight=47
IME= x:=5, y:=25

[824x210]
Num= KeyboardLayout:="numpadlayout", x:=584, y:=22
Alpha= KeyboardLayout:="nclayout", x:=5, y:=22
KeyHeight=47

区段名称符合相应的框架尺寸（以像素定义）。属性的含义如下：

属性	含义
KeyboardLayout	虚拟键盘的布局。有两种可用的布局：

	● numpadlayout:仅包含数字输入按键 ● nclayout:完整 NC 键盘
x, y	虚拟键盘左上角的位置，以框架的左上角为基准来显示键盘。
KeyHeight	虚拟键盘的按键高度（和宽度）。

如果在一个显示配置中还配置了其他尺寸的框架，用于在其中显示虚拟键盘，则应在文件 sldm_keyboard.ini 中增加相应的区段并将文件保存在 oem 目录下。

4.5.4 示例

```
; systemconfiguration.ini
; 调用应用程序 SlKeyboardApp，键盘框架尺寸在文件 sldmmcppage.ini 定义。
DLG504=name:=KeyboardApp,
implementation:=sldmvirtualkeyboardapp.SlDmVirtualKeyboardApp, process:=SlHmiHost1,
preload:=false, cmdline:="-settingsFile sldm_keyboard.ini"
```

4.5.5 虚拟 MCP

在显示配置中进行各项指定时候应使用程序名称：SlMcpApp

默认定义：

```
[dialogs]
DLG111= name:=SlMcpApp, implementation:=sldmsidescreenapp.SlSideScreenDialog,
process:=SlHmiHost1, preload:=false, cmdline:="-sidescreen1 sldmmcppage.ini"
```

在一种显示配置内，该应用程序只允许一个实例中使用。

4.5.6 示例

4.5.6.1 systemconfiguration.ini

```
; systemconfiguration.ini
; 调用应用程序 SlMcpApp，sldmmcppage.ini 文件定义
DLG502=name:=McpApp, implementation:=sldmsidescreenapp.SlSideScreenDialog,
process:=SlHmiHost1, preload:=false, cmdline:="-sidescreen1 sldmmcppage.ini"
```

4.5.6.2 sldmmcppage.ini

```
; sldmmcppage.ini
[Sidescreen]
PAGE001= name:=McpPage, implementation:=slsidescreenmcppage.SlSideScreenMcpPage
```

4.5.6.3 slsidescreenmcppageconfig.xml

```
; slsidescreenmcppageconfig.xml
; 虚拟 MCP 软键配置
<sideScreenControlPanel>
  <mainBlock id="sideScreenMcpBlock">
    <!-- block 1 -->
    <block position="1">
      <row position="1">
```

```
      <buttonGroup spacing="true" title="COOLANT">
       <button position="1" text="ON" icon="slsidescreenmcp_cooling.png">
        <onPressed address="HmiUserKey1"/>
        <active address="HmiUserLed1"/>
       </button>
       <button position="2" text="OFF" icon="slsidescreenmcp_cooling_off.png">
        <onPressed address="HmiUserKey2"/>
        <active address="HmiUserLed2"/>
       </button>
      </buttonGroup>
     </row>
          <row position="2">
      <buttonGroup spacing="true" title="EXTRA">
       <button position="1" icon="change.png">
        <onPressed address="HmiUserKey3"/>
        <active address="HmiUserLed3"/>
       </button>
       <button position="2" icon="chuck.png">
        <onPressed address="HmiUserKey4"/>
        <active address="HmiUserLed4"/>
       </button>
      </buttonGroup>
     </row>
    </block>

  </mainBlock>
</sideScreenControlPanel>
```

4.5.7 Widget 显示窗口小部件的应用程序

在显示配置中进行各项指定时候应使用程序名称：SlWidgetsApp

默认定义：

```
[dialogs]
DLG110= name:=SlWidgetsApp,
implementation:=sldmsidescreenapp.SlSideScreenDialog,
process:=SlHmiHost1, preload:=false, cmdline:="-sidescreen1
sldmwidgets1.ini -sidescreen2 sldmwidgets2.ini -spacing 3"
```

在一种显示配置内，该应用程序只允许一个实例中使用。

4.5.8 示例

4.5.8.1 systemconfiguration.ini

```
; systemconfiguration.ini
DLG503= name:=SlWidgetsApp, implementation:=sldmsidescreenapp.SlSideScreenDialog,
process:=SlHmiHost1, preload:=false, cmdline:="-sidescreen1 sldmwidgets1.ini -sidescreen2
sldmwidgets2.ini -spacing 3"
```

4.5.8.2 sldmwidgets1.ini

```
; sldmwidgets1.ini
[Sidescreen]
PAGE001= name:=WidgetsPage, implementation:=SlSideScreenPage

[Page_WidgetsPage]
ELEMENT001= name:=AxesPosition, implementation:=SlSideScreenElement
ELEMENT002= name:=WorkOffset, implementation:=SlSideScreenElement
ELEMENT003= name:=ActTool, implementation:=SlSideScreenElement
ELEMENT004= name:=VariableView, implementation:=SlSideScreenElement

[Element_AxesPosition]
TextId=AXES_POSITION_CAPTION
TextFile=slmaforms
TextContext=SlMaSideScreenAxesPositionWidget
WIDGET001= name:=AxesPositionWidget,
implementation:=slmaforms.SlMaSideScreenAxesPositionWidget

[Element_WorkOffset]
TextId=ZERO_POINT
TextFile=slpaforms
TextContext=SlPaSideScreenWorkOffsetWidget
WIDGET001= name:=WorkOffsetDetailsWidget,
implementation:=slpaforms.SlPaSideScreenWorkOffsetDetailsWidget

[Element_ActTool]
TextId=TM_SIDE_TOOL
TextFile=sltmlistdialog
TextContext=SlTmSideScreenActToolWidget
WIDGET001= name:=ActToolWidget,
implementation:=sltmlistdialog.SlTmSideScreenActToolWidget

[Element_VariableView]
TextId=Caption
```

TextFile=sldgforms
TextContext=SlDgVarViewCtx
WIDGET001= name:=VariableViewWidget,
implementation:=sldmvarviewapp.SlDgVarViewSideScreenWidget

4.5.8.3 sldmwidgets2.ini

; sldmwidgets1.ini
[Sidescreen]
PAGE001= name:=WidgetsPage, implementation:=SlSideScreenPage

[Page_WidgetsPage]
ELEMENT001= name:=AxisPerformance, implementation:=SlSideScreenElement
ELEMENT002= name:=Alarms, implementation:=SlSideScreenElement
ELEMENT003= name:=TimeCounter, implementation:=SlSideScreenElement
ELEMENT004= name:=ToolMonitoring, implementation:=SlSideScreenElement

[Element_AxisPerformance]
TextId=AXIS_PERFORMANCE_CAPTION
TextFile=slmaforms
TextContext=SlMaSideScreenAxisPerformanceWidget
WIDGET001= name:=AxisPerformanceWidget,
implementation:=slmaforms.SlMaSideScreenAxisPerformanceWidget

[Element_Alarms]
TextId=CaptionUppercase
TextFile=sldgforms
TextContext=SlDgAlarmsCtx
WIDGET001= name:=AlarmsWidget,
implementation:=sldgforms.SlDgAeSideScreenAlarmsWidget
WIDGET002= name:=MessagesWidget,
implementation:=sldgforms.SlDgAeSideScreenMsgsWidget

[Element_TimeCounter]
TextId=PROG_RUN_TIME
TextFile=slmaforms
TextContext=SlMaSideScreenRunTimeWidget
WIDGET001= name:=TimeCounterWidget,
implementation:=slmaforms.SlMaSideScreenTimesCounterWidget

[Element_ToolMonitoring]
TextId=TM_SIDE_MONITORING

TextFile=sltmlistdialog
TextContext=SlTmSideScreenMonitoringWidget
WIDGET001= name:=ToolTimeMonitoringWidget,
implementation:=sltmlistdialog.SlTmSideScreenMonitoringWidget
WIDGET002= name:=ToolCounterMonitoringWidget,
implementation:=sltmlistdialog.SlTmSideScreenMonitoringWidget
WIDGET003= name:=ToolWearMonitoringWidget,
implementation:=sltmlistdialog.SlTmSideScreenMonitoringWidget
[Widget_ToolTimeMonitoringWidget]
PROPERTY001= name:=tmToolMonitoringMode, type:=QString, value:="ToolMonTime"
[Widget_ToolCounterMonitoringWidget]
PROPERTY001= name:=tmToolMonitoringMode, type:=QString, value:="ToolMonPieces"
[Widget_ToolWearMonitoringWidget]
PROPERTY001= name:=tmToolMonitoringMode, type:=QString, value:="ToolMonWear"

4.5.9 SINUEMRIK Operate

在显示配置中进行各项指定时候应使用程序名称：OPERATE

在一种显示配置内，SINUMERIK Operate 只允许一个实例中使用。
另外还要注意，SINUMERIK Operate 只能以特定尺寸显示或执行。

4.6 B 型应用程序示例

以应用程序 notepad.exe 为例，在文件 systemconfiguration.ini 中进行程序设计：

```
[processes]
PROC500=process:=Notepad, image:=notepad, cmdline:="C:\Windows
\system32\notepad.exe", oemframe:=true, windowname:="Unbenannt - Editor",
classname:="Notepad", processaffinitymask:=0xFFFFFFFF, deferred:=true
```

在显示配置中进行各项指定时候应使用程序名称：Notepad

在一种显示配置内，该应用程序只允许一个实例中使用。
B 型应用程序只能在一个实例中执行。

4.6.1 示例

```
；oemframe.ini
[notepad]
nPlacementMode=3
WindowStyle_Off=15597568

x=56
y=1432
Width=1024
Height=488
```

hOEMFrameWnd=26060C
hOEMFrameTask=0950
hOEMAppWnd=300634
hOEMAppTask=0
hOEMAppWndRelatedOEMAppTask=1650
hOEMAppThread=0001

5 全局设置

除了定义画面及其分布的段落外，还有一个段落[miscellaneous]。其中可以对一种画面配置（配置文件）下的所有画面进行统一设置。

5.1.1 示例

```
；dm_portrait.ini
; ------------------------------------------------------------------------------------------
; miscellaneous settings for the whole configuration
; ------------------------------------------------------------------------------------------
[miscellaneous]
backgroundcolor=100/120/135
startupdisplay=d3
hmihostofmenus=SlHmiHost1
textfile=slam
textcontext=SlAmAreaMenu
```

6 应用例子

6.1 配置文件

6.1.1 appl, cfg, ico ,proj 等

\oem\sinumerik\hmi，或
\user\sinumerik\hmi

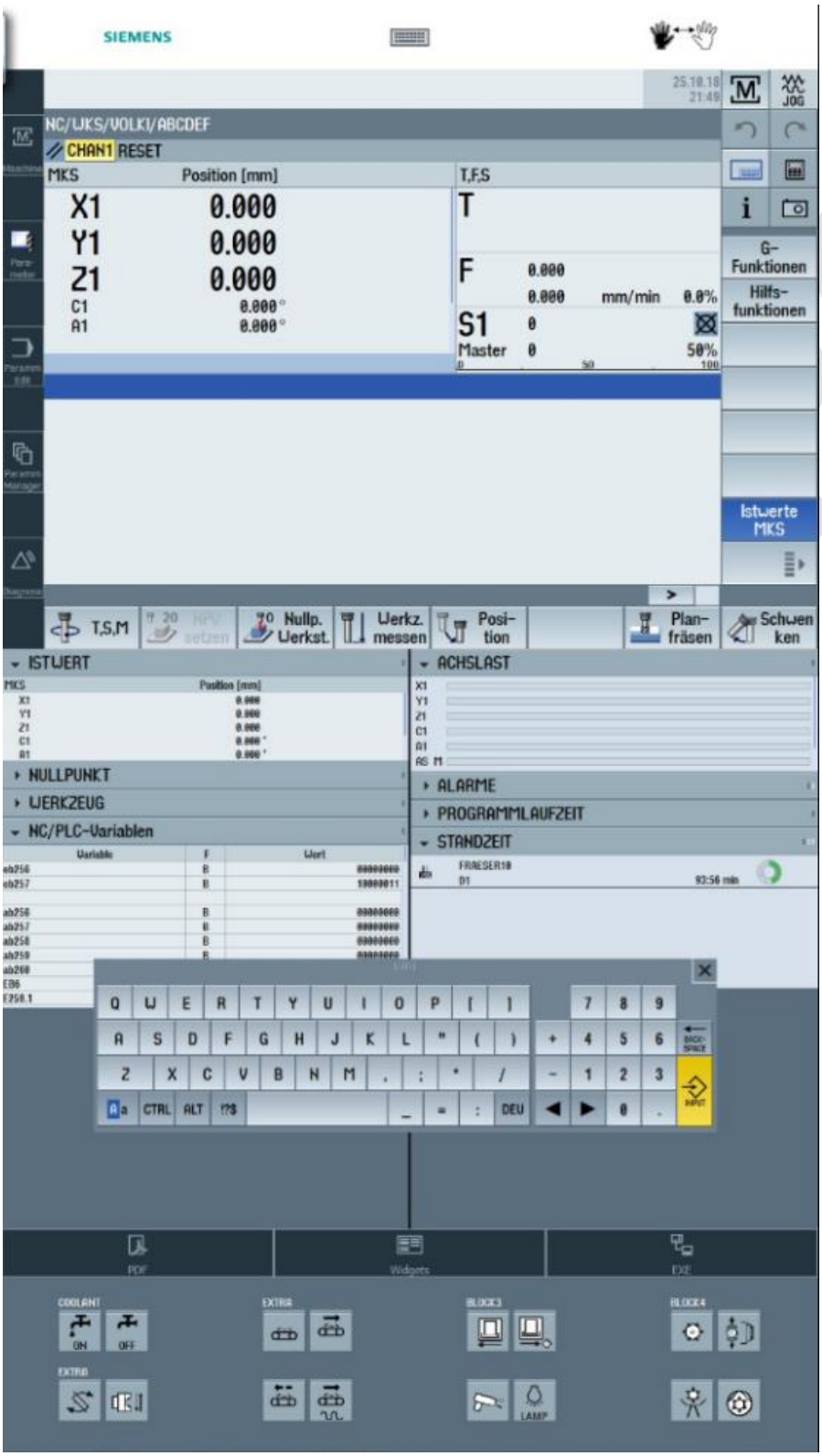

oem.7z

6.1.2 OEMFRAME.INI

\compat\user\OEMFRAME.INI

oemframe.ini

6.2 效果图



7 参考文档

SINUMERIK Operate (IM9)开机调试手册，12/2018

8 作者/联系人

顾向清

2019-04-01

9 版本信息（Option）

版本	日期	修改内容
V1.0	2019.04.01	