SIEMENS

SINUMERIK

SINUMERIK 840D sl
基本软件和操作软件

开机调试手册

前言

基本安全说明 1

SINUMERIK Operate 2

Easy XML 3

SINUMERIK Integrate Run MyScreens 4

NCU 操作系统 5

附录 A

适用于
控制系统：
SINUMERIK 840 D sl / 840 DE sl
软件：
CNC 软件 V4.94
SINUMERIK Operate V4.94

10/2020
6FC5397-1DP40-6RA4

法律资讯

警告提示系统

为了您的人身安全以及避免财产损失，必须注意本手册中的提示。人身安全的提示用一个警告三角表示，仅与财产损失有关的提示不带警告三角。警告提示根据危险等级由高到低如下表示。

⚠ 危险

表示如果不采取相应的小心措施，**将会**导致死亡或者严重的人身伤害。

⚠ 警告

表示如果不采取相应的小心措施，**可能**导致死亡或者严重的人身伤害。

⚠ 小心

表示如果不采取相应的小心措施，可能导致轻微的人身伤害。

注意

表示如果不采取相应的小心措施，可能导致财产损失。

当出现多个危险等级的情况下，每次总是使用最高等级的警告提示。如果在某个警告提示中带有警告可能导致人身伤害的警告三角，则可能在该警告提示中另外还附带有可能导致财产损失的警告。

合格的专业人员

本文件所属的产品/系统只允许由符合各项工作要求的**合格人员**进行操作。其操作必须遵照各自附带的文件说明，特别是其中的安全及警告提示。 由于具备相关培训及经验，合格人员可以察觉本产品/系统的风险，并避免可能的危险。

按规定使用 Siemens 产品

请注意下列说明：

⚠ 警告

Siemens 产品只允许用于目录和相关技术文件中规定的使用情况。如果要使用其他公司的产品和组件，必须得到 Siemens 推荐和允许。正确的运输、储存、组装、装配、安装、调试、操作和维护是产品安全、正常运行的前提。必须保证允许的环境条件。必须注意相关文件中的提示。

商标

所有带有标记符号 ® 的都是 Siemens AG 的注册商标。本印刷品中的其他符号可能是一些其他商标。若第三方出于自身目的使用这些商标，将侵害其所有者的权利。

责任免除

我们已对印刷品中所述内容与硬件和软件的一致性作过检查。然而不排除存在偏差的可能性，因此我们不保证印刷品中所述内容与硬件和软件完全一致。印刷品中的数据都按规定经过检测，必要的修正值包含在下一版本中。

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
德国

文件订购号: 6FC5397-1DP40-6RA4
Ⓟ 10/2020 本公司保留更改的权利

Copyright © Siemens AG 2006 - 2020.
保留所有权利

前言

SINUMERIK 文献

SINUMERIK 文档分为以下几个类别：

- 通用文档/产品样本
- 用户文档
- 制造商/服务文档

其它信息

访问下面的网址 (https://support.industry.siemens.com/cs/de/en/view/108464614)获取有关该主题的信息：

- 订购文档/查看印刷品一览
- 进入下载文档的链接
- 使用在线文档（查找搜索手册/信息）

如果您对技术文档有疑问（例如建议、修改），请发送一份电子邮件到以下地址 (mailto:docu.motioncontrol@siemens.com)。

mySupport/文档

您可以访问下面的网址 (https://support.industry.siemens.com/My/cn/zh/documentation)，了解如何随意组合西门子文档内容，再结合机器，创建自己的机器文档。

培训

通过以下地址 (http://www.siemens.com/sitrain)可获取有关 SITRAIN 的信息 - 西门子为驱动和自动化产品、系统和解决方案制定的培训。

常见问题

常见问题（FAQ）请参见产品支持 (https://support.industry.siemens.com/cs/de/en/ps/faq)下的服务&支持页面。

SINUMERIK

有关 SINUMERIK 的信息请访问以下网址 (http://www.siemens.com/sinumerik)。

目标读者

该手册供调试人员使用。

设备/系统已经安装连接完毕，随时可以使用。 本调试手册包含了后续步骤，
如：单个组件的配置和设计所需的所有信息，
或者必要的说明。

使用

阅读本调试手册后，
相关人员可以专业、安全地检测并调试系统或者设备。

适用阶段：安装和调试阶段

标准功能范畴

本手册描述了标准功能范畴。机床制造商增添或者更改的功能，由机床制造商资料进行说明。

控制系统有可能执行本文档中未描述的某些功能。但是这并不意味着在提供系统时必须带有这些功能，或者为其提供有关的维修服务。

同样，因为只是概要，所以该文档不包括全部类型产品的所有详细信息，也无法考虑到安装、运行和维修中可能出现的所有情况。

有关数据保护准则的说明

西门子遵守数据保护准则，特别是数据最小化原则（privacy by design）。对于该产品的具体含义是：

产品不会处理/存储个人相关数据，技术功能数据除外（例如时间戳）。用户如果将此类数据与其他数据（例如排班表）关联或者将个人相关数据存储在同一介质（例如硬盘）上而产生个人相关性，则应由用户自行确保遵循数据安全法规。

技术支持

访问网址 (https://support.industry.siemens.com/cs/cn/zh/sc/2090)下的“联系方式”，
您便可以获取各个国家技术支持的电话号码。

如需咨询技术疑问，请使用“支持请求”一栏下的在线表格。

CF 卡

- 存储卡读写数据时，不要将卡拔出， 否则可能损坏存储卡和 SINUMERIK 系统以及存储卡上的数据。
- 请小心地将存储卡按照正确的方向插入卡槽中（注意标记如箭头等）， 以避免对存储卡或设备造成机械性损伤。
- 请只使用西门子允许用于 SINUMERIK 系统的存储卡。 即使 SINUMERIK 在存储卡兼容方面遵循了一般通用的工业标准，但仍有可能出现一些制造商的存储卡在设备上无法正常工作或者不能完全兼容的情况（有关兼容性的信息请咨询存储卡制造商或供应商）。
- 有关使用 CF 卡的其他说明请参见 NCU 设备手册。

目录

前言 3

1 基本安全说明 9

1.1 一般安全说明 9

1.2 静电场或静电放电可导致设备损坏 12

1.3 应用示例的质保规定 13

1.4 安全性信息 14

1.5 驱动系统（电气传动系统）的遗留风险 15

2 SINUMERIK Operate 17

3 Easy XML 689

4 SINUMERIK Integrate Run MyScreens 1005

5 NCU 操作系统 1397

A 附录 1509

A.1 文档一览 1509

基本安全说明 1

1.1 一般安全说明



⚠ 警告

其他能源可导致电击危险和生命危险

接触带电部件可能会造成人员重伤，甚至是死亡。

- 只有专业人员才允许在电气设备上作业。
- 在所有作业中必须遵守本国的安全规定。

通常有以下安全步骤：

1. 准备断电。通知会受断电影响的组员。
2. 给驱动系统断电并确保不会再次接通。
3. 请等待至警告牌上说明的放电时间届满。
4. 确认功率接口和安全接地连接无电压。
5. 确认辅助电压回路已断电。
6. 确认电机无法运动。
7. 检查其他所有危险的能源供给，例如：压缩空气、液压、水。将能源供给置于安全状态。
8. 确保正确的驱动系统已经完全闭锁。

结束作业后以相反的顺序恢复设备的就绪状态。



⚠ 警告

连接不合适的电源可导致电击危险

连接不合适的电源会导致可接触部件携带危险电压，从而导致人员重伤，甚至是死亡。

- 所有的连接和端子只允许使用可以提供 SELV(Safety Extra Low Voltage：安全低压) 或 PELV(Protective Extra Low Voltage：保护低压) 输出电压的电源。



⚠ 警告

设备损坏可导致电击危险

未按规定操作会导致设备损坏。设备损坏后，其外壳或裸露部件可能会带有危险电压，接触外壳或这些裸露部件可能会导致重伤或死亡。

- 在运输、存放和运行设备时应遵循技术数据中给定的限值。
- 不要使用已损坏的设备。



⚠ 警告

电缆屏蔽层未接地可导致电击危险

电缆屏蔽层未接地时，电容超临界耦合可能会出现致命的接触电压。

- 电缆屏蔽层和未使用的电缆芯线至少有一侧通过接地的外壳接地。



⚠ 警告

缺少接地可导致电击危险

防护等级 I 的设备缺少安全接地连接或连接出错时，在其裸露的部件上会留有高压，接触该部件会导致重伤或死亡。

- 按照规定对设备进行接地。

注意

使用不合适的螺丝刀可损坏设备

使用不合适的螺丝刀或者采用不恰当的拧紧操作都可能损坏设备上的螺钉。

- 请使用与螺钉头完全匹配的螺丝刀。
- 请使用技术文档中规定的扭矩拧紧螺钉。
- 请使用扭力扳手或者带动态扭矩传感器和转速限制功能的机械式高精度螺丝刀。

⚠ 警告

内置型设备内可引起火灾

发生火灾时，内置型设备的外壳无法避免火苗和烟雾冒出。这可能导致人员重伤或财产损失。

- 将内置型设备安装在合适的金属控制柜中，从而保护人员免受火苗和烟雾伤害，或者对人员采取其他合适的防护措施。
- 确保烟雾只能经所设安全通道排出。

⚠ 警告

无线电设备或移动电话可导致机器意外运动

在设备的无屏蔽范围内使用无线电设备或移动电话，会干扰设备功能。功能异常会对设备功能安全产生影响并能导致人员伤亡或财产损失。

- 大约距离组件 20 cm 时，请关闭无线电设备或移动电话。
- 仅在已关闭的设备上使用“SIEMENS Industry Online Support App”。

⚠ 警告

通风空间不足可引起火灾

通风空间不足会导致过热，产生烟雾，引发火灾，从而造成人身伤害。这可能就是导致重伤或死亡的原因。此外，设备/系统故障率可能会因此升高，使用寿命缩短。

- 组件之间应保持规定的最小间距，以便通风。

注意

安装位置错误可导致过热

安装位置错误时，设备可能会过热并因此损坏。

- 只允许在规定的安装位置上运行设备。

⚠ 警告

安全功能失效可导致机器意外运动

无效的或不适合的安全功能可引起机器意外运动，可能导致重伤或死亡。

- 调试前请注意相关产品文档中的信息。
- 对整个系统和所有安全相关的组件进行安全监控，以确保安全功能。
- 进行适当设置，以确保所使用的安全功能是与驱动任务和自动化任务相匹配并激活的。
- 执行功能测试。
- 在确保了机器的的安全功能能正常工作后，才开始投入生产。

说明

Safety Integrated 功能的重要安全说明

使用 Safety Integrated 功能时务必要注意 Safety Integrated 手册中的安全说明。

⚠ 警告

因参数设置错误或修改参数设置引起机器故障

参数设置错误可导致机器出现故障，从而导致人员重伤或死亡。

- 采取保护措施，防止未经授权的参数设置。
- 采取适当措施（如驻停或急停）处理可能出现的故障。

1.2 静电场或静电放电可导致设备损坏

静电敏感元器件 (ESD) 是可被静电场或静电放电损坏的元器件、集成电路、电路板或设备。



注意

静电场或静电放电可导致设备损坏

电场或静电放电可能会损坏单个元件、集成电路、模块或设备，从而导致功能故障。

- 仅允许使用原始产品包装或其他合适的包装材料（例如：导电的泡沫橡胶或铝箔）包装、存储、运输和发运电子元件、模块和设备。
- 只有采取了以下接地措施之一，才允许接触元件、模块和设备：
  - 佩戴防静电腕带
  - 在带有导电地板的防静电区域中穿着防静电鞋或配带防静电接地带
- 电子元件、模块或设备只能放置在导电性的垫板上（带防静电垫板的工作台、导电的防静电泡沫材料、防静电包装袋、防静电运输容器）。

1.3 应用示例的质保规定

应用示例在组态和配置以及各种突发事件方面对设备没有强制约束力，无需一一遵循。应用示例不会提供客户专用的解决方案，仅在典型任务设置中提供保护。

用户自行负责上述产品的规范运行事宜。应用示例并没有解除您在应用、安装、运行和维护时确保安全环境的责任。

1.4 安全性信息

Siemens 为其产品及解决方案提供了工业信息安全功能，以支持工厂、系统、机器和网络的安全运行。

为了防止工厂、系统、机器和网络受到网络攻击，需要实施并持续维护先进且全面的工业信息安全保护机制。Siemens 的产品和解决方案构成此类概念的其中一个要素。

客户负责防止其工厂、系统、机器和网络受到未经授权的访问。只有在有必要连接时并仅在采取适当安全措施（例如，防火墙和/或网络分段）的情况下，才能将该等系统、机器和组件连接到企业网络或 Internet。

关于可采取的工业信息安全措施的更多信息，请访问 https://www.siemens.com/industrialsecurity (https://www.siemens.com/industrialsecurity)。

Siemens 不断对产品和解决方案进行开发和完善以提高安全性。Siemens 强烈建议您及时更新产品并始终使用最新产品版本。如果使用的产品版本不再受支持，或者未能应用最新的更新程序，客户遭受网络攻击的风险会增加。

要及时了解有关产品更新的信息，请订阅 Siemens 工业信息安全 RSS 源，网址为 https://www.siemens.com/industrialsecurity (https://new.siemens.com/global/en/products/services/cert.html#Subscriptions)。

其他信息请上网查找：

工业安全功能选型手册 (https://support.industry.siemens.com/cs/cn/zh/view/108862708/en)

⚠ 警告

篡改软件会引起不安全的驱动状态

篡改软件（如：病毒、木马、蠕虫等）可使设备处于不安全的运行状态，从而可能导致死亡、重伤和财产损失。

- 总是使用最新版本的软件。
- 将自动化和驱动组件集成到设备或机器上的整套先进工业信息安全方案中。
- 全面考虑整套工业信息安全方案中使用的所有产品。
- 采取相应的保护措施（如：使用杀毒软件）防止移动存储设备中的文件受到恶意软件的破坏。
- 在调试结束后，检查所有和安全相关的设置。

1.5 驱动系统（电气传动系统）的遗留风险

机器或设备制造商在依据相应的本地指令（比如欧盟机械指令）对机器或设备进行风险评估时，必须注意驱动系统的控制组件和驱动组件会产生以下遗留风险：

1. 调试、运行、维护和维修时机器或设备部件意外运行，原因（举例）：
   - 编码器、控制器、执行器和连接器中出现了硬件故障和/或软件故障
   - 控制器和传动设备的响应时间
   - 运行和/或环境条件不符合规定
   - 凝露/导电杂质
   - 参数设置、编程、布线和安装出错
   - 在电子器件附近使用无线电装置/移动电话
   - 外部影响/损坏
   - X 射线辐射、电离辐射和宇宙辐射
2. 在出现故障时，组件内/外部出现异常温度、明火以及异常亮光、噪音、杂质、气体等，原因可能有：
   - 零件失灵
   - 软件故障
   - 运行和/或环境条件不符合规定
   - 外部影响/损坏
3. 危险的接触电压，原因（举例）：
   - 零件失灵
   - 静电充电感应
   - 旋转电机的感应电压
   - 运行和/或环境条件不符合规定
   - 凝露/导电杂质
   - 外部影响/损坏
4. 设备运行中产生的电场、磁场和电磁场可能会损坏近距离的心脏起搏器支架、医疗植入体或其它金属物。
5. 当不按照规定操作以及/或违规处理废弃组件时，会释放破坏环境的物质并且产生辐射。
6. 影响通讯系统，如中央控制发送器或通过电网进行的数据通讯

其它有关驱动系统组件产生的遗留风险的信息见用户技术文档的相关章节。

SIEMENS

SINUMERIK

SINUMERIK 840D sl
SINUMERIK Operate

开机调试手册

适用于
控制系统：
SINUMERIK 840 D sl / 840 DE sl
软件：
CNC 软件 V4.94
SINUMERIK Operate V4.94

10/2020
6FC5397-1DP40-6RA4

前言

基本安全说明 1

系统概述 2

开机调试的基础 3

一般设置 4

许可 5

创建驱动器连接 6

通道菜单 7

机床数据和设定数据 8

匹配操作区域“加工” 9

模拟和记录 10

主轴功能 11

驱动系统 12

配置报警 13

数据备份 14

配置网络 15

转下页

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
德国

文件订购号: 6FC5397-1DP40-6RA4
Ⓟ 10/2020 本公司保留更改的权利

Copyright © Siemens AG 2006 - 2020.
保留所有权利

SINUMERIK 840D sl
SINUMERIK Operate

开机调试手册

续

诊断与服务 16

OEM 专用在线帮助 17

SINUMERIK Integrate - AMB、AMC、AMM 18

工艺和循环 19

语言扩展 20

PCU/IPC 上的 SINUMERIK Operate 21

手持终端 22

缩略语列表 A

法律资讯

警告提示系统

为了您的人身安全以及避免财产损失，必须注意本手册中的提示。人身安全的提示用一个警告三角表示，仅与财产损失有关的提示不带警告三角。警告提示根据危险等级由高到低如下表示。

⚠ 危险

表示如果不采取相应的小心措施，**将会**导致死亡或者严重的人身伤害。

⚠ 警告

表示如果不采取相应的小心措施，**可能**导致死亡或者严重的人身伤害。

⚠ 小心

表示如果不采取相应的小心措施，可能导致轻微的人身伤害。

注意

表示如果不采取相应的小心措施，可能导致财产损失。

当出现多个危险等级的情况下，每次总是使用最高等级的警告提示。如果在某个警告提示中带有警告可能导致人身伤害的警告三角，则可能在该警告提示中另外还附带有可能导致财产损失的警告。

合格的专业人员

本文件所属的产品/系统只允许由符合各项工作要求的**合格人员**进行操作。其操作必须遵照各自附带的文件说明，特别是其中的安全及警告提示。 由于具备相关培训及经验，合格人员可以察觉本产品/系统的风险，并避免可能的危险。

按规定使用 Siemens 产品

请注意下列说明：

⚠ 警告

Siemens 产品只允许用于目录和相关技术文件中规定的使用情况。如果要使用其他公司的产品和组件，必须得到 Siemens 推荐和允许。正确的运输、储存、组装、装配、安装、调试、操作和维护是产品安全、正常运行的前提。必须保证允许的环境条件。必须注意相关文件中的提示。

商标

所有带有标记符号 ® 的都是 Siemens AG 的注册商标。本印刷品中的其他符号可能是一些其他商标。若第三方出于自身目的使用这些商标，将侵害其所有者的权利。

责任免除

我们已对印刷品中所述内容与硬件和软件的一致性作过检查。然而不排除存在偏差的可能性，因此我们不保证印刷品中所述内容与硬件和软件完全一致。印刷品中的数据都按规定经过检测，必要的修正值包含在下一版本中。

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
德国

文件订购号: 6FC5397-1DP40-6RA4
Ⓟ 10/2020 本公司保留更改的权利

Copyright © Siemens AG 2006 - 2020.
保留所有权利

前言

SINUMERIK 文献

SINUMERIK 文档分为以下几个类别：

- 通用文档/产品样本
- 用户文档
- 制造商/服务文档

其它信息

访问下面的网址 (https://support.industry.siemens.com/cs/de/en/view/108464614)获取有关该主题的信息：

- 订购文档/查看印刷品一览
- 进入下载文档的链接
- 使用在线文档（查找搜索手册/信息）

如果您对技术文档有疑问（例如建议、修改），请发送一份电子邮件到以下地址 (mailto:docu.motioncontrol@siemens.com)。

mySupport/文档

您可以访问下面的网址 (https://support.industry.siemens.com/My/cn/zh/documentation)，了解如何随意组合西门子文档内容，再结合机器，创建自己的机器文档。

培训

通过以下地址 (http://www.siemens.com/sitrain)可获取有关 SITRAIN 的信息 - 西门子为驱动和自动化产品、系统和解决方案制定的培训。

常见问题

常见问题（FAQ）请参见产品支持 (https://support.industry.siemens.com/cs/de/en/ps/faq)下的服务&支持页面。

SINUMERIK

有关 SINUMERIK 的信息请访问以下网址 (http://www.siemens.com/sinumerik)。

目标读者

该手册供调试人员使用。

设备/系统已经安装连接完毕，随时可以使用。 本调试手册包含了后续步骤，
如：单个组件的配置和设计所需的所有信息，
或者必要的说明。

使用

阅读本调试手册后，
相关人员可以专业、安全地检测并调试系统或者设备。

适用阶段：安装和调试阶段

标准功能范畴

本手册描述了标准功能范畴。机床制造商增添或者更改的功能，由机床制造商资料进行说明。

控制系统有可能执行本文档中未描述的某些功能。但是这并不意味着在提供系统时必须带有这些功能，或者为其提供有关的维修服务。

同样，因为只是概要，所以该文档不包括全部类型产品的所有详细信息，也无法考虑到安装、运行和维修中可能出现的所有情况。

有关数据保护准则的说明

西门子遵守数据保护准则，特别是数据最小化原则（privacy by design）。对于该产品的具体含义是：

产品不会处理/存储个人相关数据，技术功能数据除外（例如时间戳）。用户如果将此类数据与其他数据（例如排班表）关联或者将个人相关数据存储在同一介质（例如硬盘）上而产生个人相关性，则应由用户自行确保遵循数据安全法规。

技术支持

访问网址 (https://support.industry.siemens.com/cs/cn/zh/sc/2090)下的“联系方式”， 您便可以获取各个国家技术支持的电话号码。

如需咨询技术疑问，请使用“支持请求”一栏下的在线表格。

CF 卡

- 存储卡读写数据时，不要将卡拔出， 否则可能损坏存储卡和 SINUMERIK 系统以及存储卡上的数据。
- 请小心地将存储卡按照正确的方向插入卡槽中（注意标记如箭头等）， 以避免对存储卡或设备造成机械性损伤。
- 请只使用西门子允许用于 SINUMERIK 系统的存储卡。 即使 SINUMERIK 在存储卡兼容方面遵循了一般通用的工业标准，但仍有可能出现一些制造商的存储卡在设备上无法正常工作或者不能完全兼容的情况（有关兼容性的信息请咨询存储卡制造商或供应商）。
- 有关使用 CF 卡的其他说明请参见 NCU 设备手册。

目录

前言 **5**

1 基本安全说明 **21**

1.1 一般安全说明 21

1.2 静电场或静电放电可导致设备损坏 24

1.3 应用示例的质保规定 24

1.4 安全性信息 24

1.5 驱动系统（电气传动系统）的遗留风险 25

2 系统概述 **27**

2.1 系统概述 27

2.2 系统条件 28

3 开机调试的基础 **31**

3.1 SINUMERIK Operate 的数据结构 31

3.2 编辑文件 35

3.2.1 在控制系统上编辑文件 35

3.2.2 外部编辑文件 37

4 一般设置 **41**

4.1 切换语言 41

4.2 设置日期/时间 42

4.3 使用时间服务器 45

4.4 使用时区 46

4.5 设备注册 48

4.6 启用对主轴负载剩余时间的显示 48

4.7 设计键盘特性 48

4.8 激活/取消虚拟键盘 49

4.9 确定软键"HMI Reboot"（重新启动 HMI）的访问权限 49

4.10 设置软键"HMI 重启"和"退出"的 OEM 功能 50

4.11 调整操作界面的设计 52

4.12 设置屏幕保护 52

4.13 设置操作面板的清洁模式 53

4.14 添加用户自定义引导启动画面 53
4.15 并行使用操作界面 54
4.16 在存档时设置登录数据处理 56
4.17 VNC 远程连接 56
4.18 设置 WEB 浏览器 60
4.19 访问等级 62
4.19.1 定义存取权限 62
4.19.2 编辑存取权限的口令 64
4.19.3 工作场所安全 65
4.19.4 程序的访问等级 66
4.19.5 文件的访问权限 68
4.19.6 软键的访问等级 69
4.19.7 确定软键的访问等级的步骤 70
4.19.8 通过保护等级设置 MDA 程序访问 72
4.20 程序选择 72
4.20.1 显示软键 72
4.21 激活并配置 Sidescreen 窗口 74
4.21.1 激活 Sidescreen 74
4.21.2 显示 Sidescreen ABC 键盘 78
4.21.3 显示机床控制面板 Sidescreen 78
4.21.3.1 设置机床控制面板的自定义按键 79
4.21.3.2 机床控制面板文本的自定义翻译 83
4.21.4 在 Sidescreen 中显示摄像头 84
4.22 SINUMERIK Operate Display Manager 85
4.22.1 Display Configuration 85
4.22.2 自定义显示配置 88
4.22.2.1 Displays（画面） 88
4.22.2.2 Frames（框架） 89
4.22.2.3 菜单 91
4.22.2.4 菜单动作 98
4.22.2.5 应用程序 101
4.22.2.6 全局设置 108
5 许可 111
5.1 确定许可证需求 112
5.2 激活测试许可证 114
5.3 输出许可证需求 115
5.4 读入许可证密钥 116
5.5 新 CF 卡的许可证备份 116

6 创建驱动器连接 119
6.1 设置驱动器 119
6.2 设置 EXTCALL 120
6.3 设置全局零件程序存储器 121
7 通道菜单 123
7.1 设计通道菜单 123
7.2 限制 124
7.3 通道菜单的一般结构 124
7.4 在 “netnames.ini" 文件中定义通道组 125
7.5 在 “config.ini" 文件中设计操作站 126
7.6 示例： 1:N 时的配置 127
7.7 1:N 配置中工作表的划分 129
8 机床数据和设定数据 133
8.1 机床数据和设定数据 133
8.2 显示/更改机床数据 134
8.3 显示/修改显示机床数据 136
8.4 显示/更改设定数据 137
8.5 显示/修改驱动参数 139
8.6 机床数据显示选项 140
8.7 编辑机床数据和驱动参数 142
8.7.1 编辑十六进制的数值 142
8.7.2 编辑 BICO 数值 143
8.7.3 编辑 Enum 值 144
8.8 查找数据 145
8.9 管理数据 146
8.10 用户视图 148
8.10.1 创建用户视图 149
8.10.2 加工用户视图 151
8.10.3 删除用户视图 152
8.11 机床数据和设定数据的明码文本 153
9 匹配操作区域“加工” 155
9.1 设置实际值显示的字体大小 155
9.2 插入用户专用商标 156

9.3 设置 G 代码组的显示 157
9.4 配置通道运行信息 159
9.5 取消程序测试 161
9.6 显示铰接位置 STAT 和回转轴位置 TU 162
9.7 激活“示教”功能 162
9.8 程序段搜索 165
9.8.1 激活程序段搜索模式 165
9.8.2 由外部处理的加速程序段搜索过程 166
9.9 多通道支持 167
9.10 手动加工 169
9.11 用户状态显示（OEM） 172
9.12 配置叠加显示 173
9.13 激活加工时间记录 174
10 模拟和记录 177
10.1 模拟概述 177
10.2 设置用于模拟的坐标系 179
10.3 同步记录概述 182
10.4 毛坯夹具 184
11 主轴功能 189
11.1 主轴控制 189
11.2 模拟主轴 190
11.3 主轴诊断 191
11.3.1 主轴诊断 191
11.3.2 温度 193
11.3.3 电机温度传感器 195
11.3.4 附加温度传感器 196
11.3.5 温度柱形图 196
11.3.6 转速/转矩 197
11.3.7 夹紧系统 197
11.3.8 夹紧系统：转速限值 199
11.3.9 夹紧系统：诊断统计 200
11.3.10 夹紧系统：夹紧时间统计 201
11.3.11 查看主轴的物流数据 202
12 驱动系统 203
12.1 驱动调试 203

13 配置报警 205
13.1 通过操作界面创建报警文本和信息文本 205
13.2 通过报警文本文件创建报警文本和信息文本 207
13.2.1 创建自身报警文本 208
13.2.2 创建指示报警参数的文本 210
13.2.3 创建零件程序信息文本 213
13.2.4 修改报警属性 216
13.2.5 替换标准报警文本 221
13.2.6 报警序号范围 226
13.2.7 报警文本的参数说明 228
13.2.8 打开故障文件 229
13.3 配置报警记录 230
13.3.1 通过操作面板设置报警记录 231
13.3.2 通过配置文件设置报警记录 231
13.4 含参数的 PLC 报警 236
13.4.1 定义数据类型为"Octet-String"的参数 236
13.4.2 定义和语种相关的格式指令 239
13.5 取消激活警告提示 240
14 数据备份 243
14.1 概览 243
14.2 创建调试存档 245
14.3 导入调试存档 248
14.4 备份硬件设计 249
14.5 创建原始数据存档 250
14.6 读入原始数据存档 253
14.7 生成完整存档 253
14.8 串行接口(V24 / RS232) 254
14.8.1 通过串行接口读入/读出存档 254
14.8.2 设置接口参数 256
14.9 保存装调数据 258
14.10 网络设置 259
15 配置网络 261
15.1 显示网络一览 261
15.2 系统网络的设置 262
15.3 公司网络设置 266

15.4 保存网络设置 267
15.5 配置操作面板 268
15.5.1 显示可用的操作面板 268
15.5.2 配置 VNC 连接 270
15.5.3 T:M:N 配置 271
15.6 基于子站的网络诊断 272
15.6.1 显示网络适配器 273
15.6.2 故障分析 275
15.6.2.1 开展故障诊断 275
15.6.2.2 无法访问 MCP/EKS 网络节点 275
15.6.2.3 无法建立 NC 和 PCU 上的 HMI 之间的网络连接 276
15.6.2.4 无法建立 TCU 和 HMI 之间的网络连接 281
15.6.2.5 无法建立 TCU 和指定 PCU 的网络连接 283
15.6.3 配置 TCP/IP 诊断 285
15.6.4 以太网诊断 288
15.6.4.1 网络诊断和 DHCP 服务器 288
15.6.4.2 信息 289
15.6.4.3 网络跟踪 289
15.6.4.4 可访问节点 293
16 诊断与服务 297
16.1 NC/PLC 变量 297
16.1.1 显示和编辑 PLC 变量和 NC 变量 297
16.1.2 保存和加载屏幕 301
16.2 显示维修概览 302
16.2.1 选择轴和驱动 303
16.2.2 轴诊断 305
16.2.3 驱动信息 311
16.3 系统负载 319
16.4 生成屏幕拷贝 320
16.5 机床识别符 321
16.5.1 输入机床专用信息 322
16.5.2 创建模板 323
16.5.3 导入模板 325
16.5.4 保存信息 327
16.5.5 添加硬件组件 328
16.5.6 配置数据 330
16.6 日志 330
16.6.1 1. 确定开机运转 331
16.6.2 2. 确定开机运转 331
16.6.3 创建日志条目 332

16.7 操作日志 333
16.7.1 设置操作日志 333
16.7.2 插入变量 335
16.7.3 显示记录文件 337
16.7.4 在记录文件中搜索 337
16.7.5 保存记录 338
16.7.6 日志文件的结构 339
16.7.7 扩展设置 342
16.8 HMI 跟踪 343
16.9 PROFIBUS 诊断 343
16.9.1 显示 DP 从站的详细信息 346
16.10 驱动系统 347
16.10.1 显示驱动状态 347
16.10.2 显示驱动对象的详细信息 348
16.11 远程诊断 350
16.11.1 匹配远程诊断 350
16.11.2 通过连接在 X127 上的远程服务适配器 IE 进行远程诊断 351
16.11.3 远程访问的 PLC 控制 353
16.12 跟踪 354
16.12.1 一般步骤 355
16.12.2 跟踪会话 356
16.12.2.1 创建跟踪会话文件 356
16.12.2.2 保存跟踪会话文件 356
16.12.2.3 加载跟踪会话文件 358
16.12.3 用于跟踪的变量 359
16.12.3.1 筛选/搜索变量 359
16.12.3.2 选择变量的属性 361
16.12.3.3 显示变量的详细信息 364
16.12.4 跟踪设置 365
16.12.4.1 跟踪设置（PLC、NC、伺服） 365
16.12.4.2 跟踪设置（驱动） 366
16.12.4.3 跟踪的选项 368
16.12.4.4 启动跟踪 368
16.12.5 分析跟踪 370
16.12.5.1 设置跟踪的视图 370
16.12.5.2 选择变量 371
16.12.5.3 设置曲线图的比例 371
16.12.5.4 缩放曲线图 372
16.12.5.5 定位光标 374
16.12.5.6 获取测量值 374
16.13 PROFIBUS/PROFINET 诊断和 AS-i 总线诊断 375
16.13.1 PROFIBUS/PROFINET 375

16.13.2 显示 PROFIBUS/PROFINET 诊断 376
16.13.3 配置 AS-i 总线 377
16.13.4 显示 AS-i 总线诊断信息 380
17 OEM 专用在线帮助 383
17.1 概览 383
17.2 生成 HTML 文件 384
17.3 创建帮助手册 388
17.4 将在线帮助集成至 SINUMERIK Operate 中 390
17.5 保存帮助文件 392
17.6 创建针对用户报警和机床数据的在线帮助 393
17.7 示例： 创建针对 NC /PLC 变量的在线帮助的步骤 397
17.8 示例： 创建编程在线帮助的步骤 399
17.9 PDF 格式的帮助文件 401
17.10 PDF 文件的语言支持 402
18 SINUMERIK Integrate - AMB、AMC、AMM 405
19 工艺和循环 407
19.1 激活车削/铣削/钻削/磨削工艺 407
19.2 用于钻削的工艺循环 414
19.3 制造商循环 416
19.3.1 制造商循环 416
19.3.2 模拟运行的前提条件 418
19.3.3 扩展标准循环 PROG_EVENT 418
19.3.4 用于换刀的制造商循环 CUST_T 和 CUST_M6 419
19.3.5 制造商循环 CUST_TECHCYC.SPF 420
19.3.6 用户循环 CUST_MULTICHAN 424
19.4 铣削 425
19.4.1 常规 425
19.4.2 用于铣削的工艺循环 425
19.4.3 设置用于铣削的 ShopMill 循环 426
19.4.4 柱面转换（TRACYL） 430
19.4.5 示例： 带有轴配置 XYZ-AC 的铣床 431
19.4.6 用于多工件加工的 ShopMill 循环 435
19.5 车削 437
19.5.1 常规 437
19.5.2 设置用于车削的循环 438
19.5.3 设置用于车削的 ShopTurn 循环 443
19.5.4 设置副主轴 445

19.5.5 设置 ShopTurn 中的副主轴 448
19.5.6 用于车削的工艺循环 450
19.5.7 车床的轴配置 454
19.5.8 柱面转换（TRACYL） 456
19.5.9 端面加工（TRANSMIT） 460
19.5.10 斜置 Y 轴 （TRAANG） 464
19.5.11 当前程序的转速限值 467
19.6 磨削 468
19.7 回转 471
19.7.1 用于回转的工艺循环 471
19.7.2 用于机床运动识别的检查列表 CYCLE800 483
19.7.3 调试 Toolcarrier/回转数据组 483
19.7.4 调试基于运动链的 Toolcarrier 490
19.7.5 调试传统的 Toolcarrier 回转数据 493
19.7.6 示例：调试传统 Toolcarrier 498
19.7.7 Herstellerzyklus CUST_800.SPF 527
19.7.8 间接编程回转数据组的名称 534
19.7.9 CYCLE9960 工件 - 运动测量 535
19.8 铣床上的车削 536
19.8.1 设置机床数据和设定数据 537
19.8.2 设置运动转换 543
19.8.3 复位和重新上电后的特性 545
19.8.4 设置刀具夹紧角度 546
19.8.5 设置切端面齿 547
19.8.6 铣削和车削不同的坐标系 548
19.8.7 CUST_800 调整 549
19.8.8 程序段搜索后的回转轴定位 552
19.8.9 程序段搜索后回转轴定位时的跟踪 552
19.9 装料适配控制器设置 553
19.9.1 装料适配 (CYCLE782) 553
19.9.2 CYCLE782:帮助图形 554
19.9.3 制造商循环 CUST_782.SPF 556
19.10 高速加工自由形状表面 558
19.10.1 配置高速设定 CYCLE832 558
19.10.2 制造商循环 CUST_832.SPF 562
19.11 测量循环和测量功能 565
19.11.1 测量循环和测量功能，通用 565
19.11.2 制造商与用户循环 CUST_MEACYC.SPF 569
19.11.3 在 JOG 运行方式下测量。 571
19.11.3.1 在铣床上测量工件 575
19.11.3.2 在铣床上测量刀具 578
19.11.3.3 在车床上测量刀具 584

19.11.3.4 在 JOG 方式中不用电子探头进行测量 585
19.11.4 在 AUTOMATIK 运行方式下测量。 586
19.11.4.1 测量工件，概述 588
19.11.4.2 在铣床上测量工件 591
19.11.4.3 在铣床上测量刀具 592
19.11.4.4 在车床上测量工件 599
19.11.4.5 在车床上测量刀具 600
19.11.4.6 双主轴上的同时测量 602
19.11.5 记录 603
19.11.5.1 测量记录，通用 603
19.11.5.2 记录 JOG 模式下的测量 604
19.11.5.3 记录 AUTOMATIK 模式下的测量 604
19.11.5.4 制造商与用户循环 CUST_MEAPROT.SPF 605
19.12 比较循环版本 607
19.12.1 显示循环版本 607
19.12.2 指定循环版本 609
20 语言扩展 613
20.1 安装其他语言 613
20.2 卸载语言 614
20.3 支持的语言 614
21 PCU/IPC 上的 SINUMERIK Operate 617
21.1 将 OEM 框架（OEMFrame）应用程序集成到系统中 617
21.2 OEMFrame 应用程序的参数设置 626
21.3 编制 OEM 子目录 636
21.4 FindWindow 程序应用 636
21.5 激活/取消窗口模式 637
21.6 设置边缘平滑的文本可读性 638
21.7 使用互动安装模式或静默（Silent）安装模式 639
21.8 设置 NCU 的 IP 地址 641
21.9 退出 SINUMERIK Operate 642
22 手持终端 643
22.1 设置 HT 8 运行键 643
22.2 HT 10 运行键 649
22.3 激活/取消虚拟键盘 649
22.4 设计用户专用按键说明 650
22.5 配置用户专用键（U 按键）上的功能显示 652

A 缩略语列表 657
索引 661

基本安全说明 1

1.1 一般安全说明



⚠ 警告

其他能源可导致电击危险和生命危险

接触带电部件可能会造成人员重伤，甚至是死亡。

- 只有专业人员才允许在电气设备上作业。
- 在所有作业中必须遵守本国的安全规定。

通常有以下安全步骤：

1. 准备断电。通知会受断电影响的组员。
2. 给驱动系统断电并确保不会再次接通。
3. 请等待至警告牌上说明的放电时间届满。
4. 确认功率接口和安全接地连接无电压。
5. 确认辅助电压回路已断电。
6. 确认电机无法运动。
7. 检查其他所有危险的能源供给，例如：压缩空气、液压、水。将能源供给置于安全状态。
8. 确保正确的驱动系统已经完全闭锁。

结束作业后以相反的顺序恢复设备的就绪状态。



⚠ 警告

连接不合适的电源可导致电击危险

连接不合适的电源会导致可接触部件携带危险电压，从而导致人员重伤，甚至是死亡。

- 所有的连接和端子只允许使用可以提供 SELV(Safety Extra Low Voltage：安全低压) 或 PELV(Protective Extra Low Voltage：保护低压) 输出电压的电源。



⚠ 警告

设备损坏可导致电击危险

未按规定操作会导致设备损坏。设备损坏后，其外壳或裸露部件可能会带有危险电压，接触外壳或这些裸露部件可能会导致重伤或死亡。

- 在运输、存放和运行设备时应遵循技术数据中给定的限值。
- 不要使用已损坏的设备。



⚠ 警告

电缆屏蔽层未接地可导致电击危险

电缆屏蔽层未接地时，电容超临界耦合可能会出现致命的接触电压。

- 电缆屏蔽层和未使用的电缆芯线至少有一侧通过接地的外壳接地。



⚠ 警告

缺少接地可导致电击危险

防护等级 I 的设备缺少安全接地连接或连接出错时，在其裸露的部件上会留有高压，接触该部件会导致重伤或死亡。

- 按照规定对设备进行接地。

注意

使用不合适的螺丝刀可损坏设备

使用不合适的螺丝刀或者采用不恰当的拧紧操作都可能损坏设备上的螺钉。

- 请使用与螺钉头完全匹配的螺丝刀。
- 请使用技术文档中规定的扭矩拧紧螺钉。
- 请使用扭力扳手或者带动态扭矩传感器和转速限制功能的机械式高精度螺丝刀。

⚠ 警告

内置型设备内可引起火灾

发生火灾时，内置型设备的外壳无法避免火苗和烟雾冒出。这可能导致人员重伤或财产损失。

- 将内置型设备安装在合适的金属控制柜中，从而保护人员免受火苗和烟雾伤害，或者对人员采取其他合适的防护措施。
- 确保烟雾只能经所设安全通道排出。

⚠ 警告

无线电设备或移动电话可导致机器意外运动

在设备的无屏蔽范围内使用无线电设备或移动电话，会干扰设备功能。功能异常会对设备功能安全产生影响并能导致人员伤亡或财产损失。

- 大约距离组件 20 cm 时，请关闭无线电设备或移动电话。
- 仅在已关闭的设备上使用“SIEMENS Industry Online Support App”。

⚠ 警告

通风空间不足可引起火灾

通风空间不足会导致过热，产生烟雾，引发火灾，从而造成人身伤害。这可能就是导致重伤或死亡的原因。此外，设备/系统故障率可能会因此升高，使用寿命缩短。

- 组件之间应保持规定的最小间距，以便通风。

注意

安装位置错误可导致过热

安装位置错误时，设备可能会过热并因此损坏。

- 只允许在规定的安装位置上运行设备。

⚠ 警告

安全功能失效可导致机器意外运动

无效的或不适合的安全功能可引起机器意外运动，可能导致重伤或死亡。

- 调试前请注意相关产品文档中的信息。
- 对整个系统和所有安全相关的组件进行安全监控，以确保安全功能。
- 进行适当设置，以确保所使用的安全功能是与驱动任务和自动化任务相匹配并激活的。
- 执行功能测试。
- 在确保了机器的的安全功能能正常工作后，才开始投入生产。

说明

Safety Integrated 功能的重要安全说明

使用 Safety Integrated 功能时务必要注意 Safety Integrated 手册中的安全说明。

⚠ 警告

因参数设置错误或修改参数设置引起机器故障

参数设置错误可导致机器出现故障，从而导致人员重伤或死亡。

- 采取保护措施，防止未经授权的参数设置。
- 采取适当措施（如驻停或急停）处理可能出现的故障。

1.2 静电场或静电放电可导致设备损坏

静电敏感元器件 (ESD) 是可被静电场或静电放电损坏的元器件、集成电路、电路板或设备。



注意

静电场或静电放电可导致设备损坏

电场或静电放电可能会损坏单个元件、集成电路、模块或设备，从而导致功能故障。

- 仅允许使用原始产品包装或其他合适的包装材料（例如：导电的泡沫橡胶或铝箔）包装、存储、运输和发运电子元件、模块和设备。
- 只有采取了以下接地措施之一，才允许接触元件、模块和设备：
  - 佩戴防静电腕带
  - 在带有导电地板的防静电区域中穿着防静电鞋或配带防静电接地带
- 电子元件、模块或设备只能放置在导电性的垫板上（带防静电垫板的工作台、导电的防静电泡沫材料、防静电包装袋、防静电运输容器）。

1.3 应用示例的质保规定

应用示例在组态和配置以及各种突发事件方面对设备没有强制约束力，无需一一遵循。应用示例不会提供客户专用的解决方案，仅在典型任务设置中提供保护。

用户自行负责上述产品的规范运行事宜。应用示例并没有解除您在应用、安装、运行和维护时确保安全环境的责任。

1.4 安全性信息

Siemens 为其产品及解决方案提供了工业信息安全功能，以支持工厂、系统、机器和网络的安全运行。

为了防止工厂、系统、机器和网络受到网络攻击，需要实施并持续维护先进且全面的工业信息安全保护机制。Siemens 的产品和解决方案构成此类概念的其中一个要素。

客户负责防止其工厂、系统、机器和网络受到未经授权的访问。只有在有必要连接时并仅在采取适当安全措施（例如，防火墙和/或网络分段）的情况下，才能将该等系统、机器和组件连接到企业网络或 Internet。

关于可采取的工业信息安全措施的更多信息，请访问 https://www.siemens.com/industrialsecurity (https://www.siemens.com/industrialsecurity)。

Siemens 不断对产品和解决方案进行开发和完善以提高安全性。Siemens 强烈建议您及时更新产品并始终使用最新产品版本。如果使用的产品版本不再受支持，或者未能应用最新的更新程序，客户遭受网络攻击的风险会增加。

要及时了解有关产品更新的信息，请订阅 Siemens 工业信息安全 RSS 源，网址为 https://www.siemens.com/industrialsecurity (https://new.siemens.com/global/en/products/services/cert.html#Subscriptions)。

其他信息请上网查找：

工业安全功能选型手册 (https://support.industry.siemens.com/cs/cn/zh/view/108862708/en)

⚠ 警告

篡改软件会引起不安全的驱动状态

篡改软件（如：病毒、木马、蠕虫等）可使设备处于不安全的运行状态，从而可能导致死亡、重伤和财产损失。

- 总是使用最新版本的软件。
- 将自动化和驱动组件集成到设备或机器上的整套先进工业信息安全方案中。
- 全面考虑整套工业信息安全方案中使用的所有产品。
- 采取相应的保护措施（如：使用杀毒软件）防止移动存储设备中的文件受到恶意软件的破坏。
- 在调试结束后，检查所有和安全相关的设置。

1.5 驱动系统（电气传动系统）的遗留风险

机器或设备制造商在依据相应的本地指令（比如欧盟机械指令）对机器或设备进行风险评估时，必须注意驱动系统的控制组件和驱动组件会产生以下遗留风险：

1. 调试、运行、维护和维修时机器或设备部件意外运行，原因（举例）：
   - 编码器、控制器、执行器和连接器中出现了硬件故障和/或软件故障
   - 控制器和传动设备的响应时间
   - 运行和/或环境条件不符合规定
   - 凝露/导电杂质
   - 参数设置、编程、布线和安装出错
   - 在电子器件附近使用无线电装置/移动电话
   - 外部影响/损坏
   - X 射线辐射、电离辐射和宇宙辐射
2. 在出现故障时，组件内/外部出现异常温度、明火以及异常亮光、噪音、杂质、气体等，原因可能有：
   - 零件失灵
   - 软件故障
   - 运行和/或环境条件不符合规定
   - 外部影响/损坏

3. 危险的接触电压，原因（举例）：
   – 零件失灵
   – 静电充电感应
   – 旋转电机的感应电压
   – 运行和/或环境条件不符合规定
   – 凝露/导电杂质
   – 外部影响/损坏
4. 设备运行中产生的电场、磁场和电磁场可能会损坏近距离的心脏起搏器支架、医疗植入体或其它金属物。
5. 当不按照规定操作以及/或违规处理废弃组件时，会释放破坏环境的物质并且产生辐射。
6. 影响通讯系统，如中央控制发送器或通过电网进行的数据通讯

其它有关驱动系统组件产生的遗留风险的信息见用户技术文档的相关章节。

系统概述 2

2.1 系统概述

说明

PCU / IPC

本文档说明了如何进行 PCU 和 IPC 中的文件配置。

NCU

操作软件 SINUMERIK Operate V4.x 集成在数控软件 V4.x 内，可以在 NCU 7x0.3 上运行。出厂时，数控软件保存在 NCU 的 CF 卡上。

操作软件 SINUMERIK Operate 随 NCU 软件启动而自动启动，提供出厂时标配的功能，但标配功能也受当前 NC 配置和 PLC 配置的影响。

PCU/IPC

操作软件 SINUMERIK Operate 是以 DVD 光盘的形式发货的，必须随后安装到 PCU/IPC 的硬盘上。操作软件随着控制系统的启动而启动。

PC

操作软件 SINUMERIK Operate 是以 DVD 光盘的形式发货的，可以安装到任意一个目录下。

下文将说明机床厂商如何对操作软件 SINUMERIK Operate 进行扩展和调整。

更多信息

关于 SINUMERIK 840D sl 开机调试的详细说明请参见以下文档：

CNC 调试手册：NCK、PLC、驱动

关于将 SINUMERIK Operate 集成到 WinCC / TIAP 的说明，请参见以下文档：

SINUMERIK Integrate Create MyHMI /WinCC 选型手册

2.2 系统条件

概述

所有系统组件完整装配、安装和配置是软件运行的前提条件。NC 和 PLC 必须已经完成调试，并且 PLC 处于“运行”状态。

- 网络连接和/或存储媒介，例如：USB 设备，使控制系统和外部 PC/PG 之间可以交换数据。

说明

USB 键盘

您可以使用 USB 键盘输入包含大小写字母的文本。

说明

使用 USB 设备存储用户数据时，数据不可超过 4 GB!

Windows 操作系统下的 SINUMERIK Operate

在 Windows 操作系统下使用 SINUMERIK Operate 时，必须满足以下条件：

- PCU 50.5：自 V1.2 起的 PCU 基础软件
- PCU 50.5：自 V11.4 起的 PCU 基础软件 Win7
- IPC:自 V11.4 起的 PCU 基础软件 Win7
- IPC:PCU Base 适用于 IPC Win 10 自 V13 起
- PC：Windows XP SP3 / Windows 7 / Windows 10 操作系统

附加程序

- 文本编辑器 TextPad:用于在 Windows 操作系统下外部创建和编辑 XML 和 ini 文件。请访问以下网址获得程序：
  http://www.textpad.com
- WinSCP:用于多个计算机(NCU - PC)之间的安全数据传送。请访问以下网址获得程序：
  http://winscp.net/eng/docs/lang:de
- SINUMERIK Integrate Access MyMachine /P2P（之前的 RCS Commander）：一种远程诊断工具，使服务人员在一台标准 Windows PC 上通过操作软件 SINUMERIK Operate 即可远程诊断机床。

- SINUMERIK Integrate Create MyConfig:基于模块化方案，可以对同一批次的不同机床进行批量调试和升级。
- Access MyMachine 提供了一种特殊的转化功能，将 HMI 高级版或 HMI 内置版的报警文字转换成 SINUMERIK Operate 支持的格式。

开机调试的基础 3

3.1 SINUMERIK Operate 的数据结构

SINUMERIK Operate 的所有文件保存在 CF 卡上（NCU）；或保存在硬盘上（PCU/IPC）；或保存在 DVD 光盘上（PC）。

文件系统位于最高级别的目录中：

- 在 NCU 上，位于目录：”/System CF-Card/”中
- 在 PCU/IPC 上，位于目录“C:\Program Files (x86)\Siemens\MotionControl”中
- 在 PC 上可自由选择驱动器，例如“C:\Programme\siemens\sinumerik”

目录由以下子目录构成：

- addon
- compat（仅限 PC/PCU/IPC）
- oem
- siemens
- user

上述子目录的结构基本上相同。

说明

目录“siemens”内的文件表示出厂状态，无法修改！

可添加或修改内容的文件在以下目录中作为模板（样本文件）提供：

文件	目录
配置文件	/siemens/sinumerik/hmi/template/cfg
文本文件	/siemens/sinumerik/hmi/template/lng

在对这些文件进行修改/添加前，必须将其复制到目录 /**oem**/sinumerik/hmi/、/**addon**/sinumerik/hmi/ 或者 /**user**/sinumerik/hmi/ 中的 /cfg 或 /lng 中。

结构

下表列出了目录结构中和 SINUMERIK Operate 相关的部分：

siemens 目录			
/siemens/sinumerik			
	/hmi		
		/appl	// 应用（操作区域）
		/base	// 基础系统组件
		/cfg	// 所有配置文件
		/data	// 版本数据
		/hlp	// 在线帮助文件
		/hlps	// 在线帮助文件压缩包和版本文件
		/ico	// 图标文件
		/Ico640	// 分辨率 640x480 的图标
		/ico800	// 分辨率 800x600 的图标
		/ico1024	// 分辨率 1024x768 的图标
		/ico1280	// 分辨率 1280x1024 的图标
		/ico1600	// 分辨率 1600x1240 的图标
		/lng	// 文本文件
		/lngs	// 文本文件压缩包和版本文件
		/osal	
		/ace	// ACE/TAO
		/qt	// Qt
		/proj	// EasyScreen 配置
		/template	// 各种模板
		/cfg	// 配置文件的模板
		/ing	// 文本文件的模板
		/tmpp	// 临时数据存档
	/sys_cache/hmi		// 各种记录文件

compat 目录		
/compat		// 用于并入 OEMFrame 应用程序的文件
	/add_on	
	/hmi_adv	
	/mmc2	
	/oem	
	/user	

addon 目录			
/addon/sinumerik			
	/hmi		
		/appl	// 应用（操作区域）
		/cfg	// 配置文件
		/data	// 版本数据
		/hlp	// 在线帮助文件压缩包和版本文件
		/ico	// 图标文件
		/Ico640	// 分辨率 640x480 的图标
		/ico800	// 分辨率 800x600 的图标
		/ico1024	// 分辨率 1024x768 的图标
		/ico1280	// 分辨率 1280x1024 的图标
		/ico1600	// 分辨率 1600x1240 的图标
		/lng	// 文本文件
		/lngs	// 文本文件压缩包和版本文件
		/proj	// EasyScreen 配置
		/template	// 各种模板

oem 目录		
/oem/sinumerik		
	/data	// 版本数据
	/archive	// 制造商存档
	/hmi	

oem 目录			
	/appl		// 应用（操作区域）
	/cfg		// 配置文件
	/data		// 版本数据
	/hlp		// 在线帮助文件
	/hlps		// 在线帮助文件压缩包和版本文件
	/ico		// 图标文件
		/Ico640	// 分辨率 640x480 的图标
		/ico800	// 分辨率 800x600 的图标
		/ico1024	// 分辨率 1024x768 的图标
		/ico1280	// 分辨率 1280x1024 的图标
		/ico1600	// 分辨率 1600x1240 的图标
	/lng		// 文本文件
	/lngs		// 文本文件压缩包和版本文件
	/proj		// EasyScreen 配置
	/template		// 各种模板

user 目录			
/user/sinumerik			
	/data		// 版本数据
		/archive	// 用户专用存档
		/prog	// 用户专用程序
	/hmi		
		/cfg	// 配置文件
		/data	// 版本数据
		/hlp	// 在线帮助文件
		/ico	// 图标文件
		/Ico640	// 分辨率 640x480 的图标
		/ico800	// 分辨率 800x600 的图标
		/ico1024	// 分辨率 1024x768 的图标
		/ico1280	// 分辨率 1280x1024 的图标
		/ico1600	// 分辨率 1600x1240 的图标

user 目录		
	/lng	// 文本文件
	/proj	// EasyScreen 配置
	/log	// 记录文件
	/md	// 机床数据视图
	/proj	// EasyScreen 配置

3.2 编辑文件

3.2.1 在控制系统上编辑文件

您可以复制模板文件，执行所需修改。模板文件位于如下目录中：

/**siemens**/sinumerik/hmi/template/cfg 和 /**siemens**/sinumerik/hmi/template/lng

将模板文件保存在目录 “User” 或 “OEM” 中相应的文件夹下。

说明

“User”目录中文件的优先级始终高于“Siemens”目录中的文件，只有当“User”文件缺少某个条目时，才会采用“Siemens”文件中的条目。
文件中的设置只有在重启操作软件后才生效。

前提条件

- 复制文件要求访问级别 1（制造商）。
- 给文件命名时应注意，文件名最长只能为 49 个字符。

复制/粘贴/打开文件



1. 选择操作区域“调试”。



2. 按下软键“系统数据”。
   显示树形图。

3. 以 NCU 为例，打开“Siemens/System CF-Card”下的所需目录，例如 /sinumerik/hmi/template/cfg。
4. 将光标移到所需文件上。



5. 按下软键“复制”。
6. 以 NCU 为例，打开“oem”或者“user”中“System CF-Card”下希望存放拷贝的目录，例如：/sinumerik/hmi/cfg。



7. 按下软键“粘贴”。

   如果已经存在一个同名的文件，会弹出提示，您可以覆盖或重命名文件。



8. 按下软键“确认”。



9. 按下软键“打开”，在编辑器中打开所需文件。

   -或者-



   按下 <INPUT> 键。

   -或者-

   双击选中的文件。

重命名文件

1. 选中所需的文件。



2. 按下软键“>>”和“属性”。



   窗口“...属性”打开。

   显示下列数据：

   - 路径：
   - 名称：
   - 创建日期和时间：
   - 修改日期和时间：

3. 需要修改文件名时，将光标移动到输入栏“名称”，重新写入名称。修改日期和时间会一同更新。



4. 按下软键“确认”，保存新的文件名。

剪切/删除文件

1. 选中所需的文件。



2. 按下软键“剪切”。

文件被剪切到剪贴板中，同时从原有目录删除。

文件位于剪贴板中，也可以粘贴到另一个目录中。

-或者-



2. 按下软键“>>”和“删除”。

随即弹出提示信息，可按下软键“确认”删除文件。

按下软键“取消”，中断删除。

在预览窗口显示文件

1. 选中所需的文件。



2. 按下软键“>>”和“预览窗口”。

在窗口下方会弹出窗口“预览”，并显示文件内容。





再次按下软键“预览窗口”，关闭窗口。

3.2.2 外部编辑文件

如果需要在外部 PC 的 Windows 操作系统下创建或者编辑 XML 文件，请使用支持 UTF-8 编码的文本编辑器，比如 “TextPad”。

从而可通过键盘输入所有 Unicode 系统的字符或者直接粘贴，而无须转换。

若文件以其它编码保存，则其特殊字符不能在编辑器中正确显示。SINUMERIK Operate 中不能自动转换至 UTF 8 编码！

以 UTF-8 编码保存 XML 文件

1. 选择对话框“另存为”。
2. 将编码方式设为“UTF-8”。

在 XML 文件内输入注释

在输入注释来说明程序时，必须注意以下几点：

- 注释始终以字符串开始：<!--
- 注释始终以字符串结束：-->

示例

<!-- 零点偏移: -->

说明

然而，在注释中不能连续使用两个负号！

特殊字符

若要在文本中使用对于 XML 句法比较重要的字符，必须先对其进行专门的格式转换。

特殊字符	输入方法
&	&
'	'
<	<
>	>
"	"

ini 文件中的注释

在 ini 文件中输入注释时，注释行以一个分号开始。

复制文件

使用程序"WinSCP"或"Access MyMachine"通过网络连接将文件从外部计算机复制到 NCU，或者从 NCU 复制到外部 PC。

更多信息

关于备份和恢复数据、服务工具 WinSCP 和 PuTTY 的详细信息，请参见

NCU 操作系统调试手册。

参见

通过报警文本文件创建报警文本和信息文本 (页 207)

一般设置

4

4.1 切换语言

标准配置中，操作软件有多种语言。

选择语言

可通过如下显示机床数据确定语言选择模式。

MD9100 $MM_CHANGE_LANGUAGE_MODE	语言选择模式
= 1	通过窗口“语言选择”确定界面语言（缺省值）。



1. 选择操作区域“调试”。



2. 按下软键“Change language”。
   “语言选择”窗口打开。上次设置的语言被选中。
3. 请将光标定位在需要的语言上。



4. 按下 <INPUT> 键。

-或者-



按下软键“确认”。

操作软件将切换到所选语言。

确定两种语言

MD9100 $MM_CHANGE_LANGUAGE_MODE	语言选择模式
= 2	在两种语言之间进行选择。在运行期间可以通过软键“Change language”切换语言。



1. 选择操作区域“调试”。



2. 按下软键“HMI”。



3. 按下软键“>>”。



4. 按下软键“语言选择”。
“语言选择”窗口打开。
在“第一种语言”和“第二种语言”栏中将显示全部已安装的语言。

5. 分别选择一种语言。



6. 按下 <INPUT> 键。

-或者-



按下软键“确认”。



在运行中可以在两种语言间切换，通过按下软键“Change language”。

4.2 设置日期/时间

您可以手动设置 NCU（PLC 和 Linux）或 PCU/PC（Windows）的日期和时间。 其中您还可以选择不同的显示格式。

日期显示的格式

格式	示例
d.M.yy	3.1.09
dd.MM.yy	03.01.09
dd.MM.yyyy	03.01.2009
d/M/yy	3/01/09
dd/MM/yy	03/01/09
dd/MM/yyyy	03/01/2009
M/d/yy	1/3/09
MM/dd/yy	01/03/09

格式	示例
MM/dd/yyyy	01/03/2009
yyyy/MM/dd	2009/01/03

这里：

日期	输入说明
d	每个月的一位或者二位数日期 (1 - 31)。
dd	每个月的两位数日期，第一位默认为零 (01 - 31)。
M	每年的一位或者两位数月份 (1 - 12)。
MM	每年的两位数月份，第一位默认为零 (01 - 12)。
yy	两位数年份 (00 - 99)。
yyyy	四位数年份（比如 2009）。

时间显示的格式

格式	示例	小时制
H:m:s AP	1:2:4 PM	24 小时制，显示“AM”/“PM”
h:m:s ap	1:2:8 pm	12 小时制，显示“am”/“pm”
HH:mm:ss AP	13:02:08 PM	24 小时制，显示“AM”/“PM”
hh:mm:ss ap	01:02:09 pm	12 小时制，显示“am”/“pm”。
H:mm:ss AP	1:02:09 PM	24 小时制，显示“AM”/“PM”
h:mm:ss ap	1:02:09 pm	12 小时制，显示“am”/“pm”
h:mm:ss	13:02:09	24 小时制
hh:mm:ss	13:12:04	24 小时制

这里：

时间符号：	输入说明
h	用一位或者两位数表示小时（0 - 23 或者 1 - 12 am/pm）。
hh	用两位数表示小时，第一位默认为零（00 - 23 或者 01 - 12 am/pm）。
H	用一位或者两位数表示小时（0 - 23 AM/PM）。
HH	用两位数表示小时，第一位默认为零（00 - 23 AM/PM）。

时间符号：	输入说明	
m	用一位或者两位数表示分钟 (0 - 59)。	
mm	用两位数表示分钟，第一位默认为零 (00 - 59)。	
s	用一位或者两位数表示秒数 (0 - 59)。	
ss	用两位数表示秒，第一位默认为零 (00 - 59)。	
AP	白天时间的显示：	AM 或者 am = 上午 PM 或者 pm = 下午

步骤



1. 选择操作区域“调试”。



2. 按下软键“HMI”。



3. 按下软键“日期时间”。

   窗口“设置日期和时间”打开。



4. 按下<SELECT>键，在“格式”栏选择需要的日期和时间的显示格式。

5. NCU 中的 SINUMERIK Operate：在输入栏中在“新建”后输入当前日期以及时间。

   PCU/PC 上的 SINUMERIK Operate：在输入栏中在“新 NCU + PCU”后输入当前日期以及时间。



6. 按下软键“确认”结束输入。

   新的日期或时间被接收，分别在“当前 NCU”栏和“当前 PCU”栏中显示。

说明

设置 PCU/PC 上的日期和时间需要管理员权限。

4.3 使用时间服务器

可将最多三个时间服务器与操作软件连接，这样一来控制系统总能自动获得正确的时间信息。在窗口“设置日期和时间”中进行设置。

说明

“使用时间服务器”设置在 PCU/PC/IPC 上的 SINUMERIK Operate 中不提供。

注意
时间同步错误 不要在两个不同的系统上设置时间同步；这表示： 要么在 PLC 项目中通过 PROFINET IO (X150) 选择 NTP 服务器，要么在文件 basesys.ini 的 NCU 上选择 [LinuxBase] - 但两者不能同时选中，因为在 NCU 上还可指定一个时区，而在 PLC 项目中不可以。

更多信息

有关配置文件 “basesys.ini” 和外部时间服务器使用的更多信息请参见

NCU 操作系统调试手册。

前提条件

访问级别至少为 3（用户）时，才可使用时间服务器。

检查连接状态

“名称/IP”输入栏右边的图标显示服务器连接的状态。

图标	含义
	连接状态未知。设置还未处理或确定。
	与时间服务器之间存在连接。系统成功与时间说明同步。
	系统仍尝试与时间服务器同步。

说明

系统循环检查时间服务器连接。如果进行了修改，则在设置生效前会持续几秒钟。

步骤



1. 选择操作区域“调试”。



2. 按下“HMI”软键。



3. 按下“日期时间”软键。

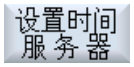
   窗口“设置日期和时间”打开。

4. 勾选“使用时间服务器”栏。

   显示输入栏“名称 / IP”。

5. 在“名称/IP”栏中输入 DNS 名称或时间服务器的 IP 地址。

   最多可输入三个时间服务器，用空格隔开。



6. 按下软键“设置时区”。

   如果时间服务器上存在连接，则设置与系统即刻进行同步。然后系统会循环检查设置是否生效。窗口“设置日期和时间”保持打开。



7. 按下软键“确认”结束输入。

   时间服务器设置和当前的时间说明由控制系统传输。

说明

如果取消勾选“使用时间服务器”栏，则系统再次使用标准时区“UTC”。

4.4 使用时区

您可以选择所在时区，使控制系统了解所在时区的当地时间。

系统会使用此处设置的时区，将 UTC 时间（协调世界时）换算成所在时区的当地时间。

说明

UTC 时间是默认设置。

步骤



1. 选择操作区域“调试”。



2. 按下“HMI”软键。



3. 按下“日期时间”软键。
   窗口“设置日期和时间”打开。
4. 勾选“使用时间服务器”栏。
   输入栏“时区”激活。
5. 在“时区”栏中选择合适的时区。



6. 按下软键“确认”结束输入。
   当前时区传送给系统。

说明

时区的重新设置可能要持续几秒。

说明

无夏令时/冬令时的时区

在设置时区时请注意您所在国家是否使用夏令时/冬令时。比如：中国或日本等国家没有夏令时/冬令时，而欧洲和美国有。

示例

以小时为单位将 UTC 时间往前调或往后调。比如：您可以设置以下时区：

欧洲	通用 Etc 时间（UTC）
	欧洲阿姆斯特丹时间（UTC-1:00）
	欧洲柏林时间（UTC-1:00）
美国	美国中部时间（UTC+6:00）
	美国密歇根时间（UTC+5:00）
	美国太平洋时间（UTC+8:00）
中国	亚洲香港时间（UTC-8:00）
日本	东京时间（UTC-9）

4.5 设备注册

您的机床必须完成注册，才能获得西门子的服务支持。注册与质保关联。

如果机床未注册，则会出现信息 150412：

无西门子质保相关信息。

说明

信息 150412 无法应答。

请在“诊断”操作区域中注册您的机床：

“诊断” → “版本” → “注册机床”

更多信息

有关如何注册机床的更多信息，请参见在线文档
调试功能及服务。

4.6 启用对主轴负载剩余时间的显示

为了避免主轴过载，在“T,F,S”窗口中的主轴负载率显示旁，还会显示主轴的最长剩余使用时间，单位：秒。当主轴在当前负载下的剩余时间短于 120 秒时，便会显示时间和主轴负载图标。

轴机床数据

设置以下机床数据：

MD53035 $MAS_SHOW_SPINDLE_UTIL_TIME	显示主轴使用时长
= 0	不显示主轴使用时长
= 1	显示主轴使用时长

4.7 设计键盘特性

通过 CAPSLOCK （大写锁定）可以锁定外部键盘的小写输入，一直采用大写输入。

按键属性通过显示机床数据设定。

设置

MD9009 $MM_KEYBOARD_STATE	
= 0	CAPSLOCK 关闭（缺省值）
= 2	CAPSLOCK 打开

在更改机床数据后必须重新启动系统。

4.8 激活/取消虚拟键盘

在文件 "slguiconfig.ini" 中配置虚拟键盘。

步骤

1. 从目录
   /**siemens**/sinumerik/hmi/template/cfg 中复制文件 "slguiconfig.ini"。
2. 将复制件加入目录中：
   /**oem**/sinumerik/hmi/cfg 或者/**user**/sinumerik/hmi/cfg。
3. 在编辑器中打开文件。
4. 为激活或者取消键盘进行如下设置：
   - 激活虚拟键盘：
     段落 [TouchPanel], EnableTouch = true
     段落 [Keyboard], EnableVirtualKeyBoard = true
   - 取消虚拟键盘：
     段落 [TouchPanel], EnableTouch = true
     段落 [Keyboard], EnableVirtualKeyBoard = false
5. 双击输入框，以显示键盘。

4.9 确定软键"HMI Reboot"（重新启动 HMI）的访问权限

机床数据显示

可通过如下显示机床数据设置从哪个访问级别起可以显示软键"HMI 重启"

PC 或 PCU 上的 SINUMERIK Operate 上会显示软键“退出”。

MD9110_$MM_ACCESS_HMI_EXIT	软键"EXIT"的访问级别
= 1	访问级别 1（制造商），缺省值

重新启动 HMI

通过以下步骤重新启动 SINUMERIK Operate：

打开操作区域，例如使用按键 <MENU SELECT> → "菜单扩展键" → "HMI 重启"

4.10 设置软键"HMI 重启"和"退出"的 OEM 功能

可以修改软键"HMI 重启"或"退出"的标准生效方式。配置在文件"slamconfig.ini"中设置。

创建配置文件"slamconfig.ini"

在以下某个目录下插入"slamconfig.ini"文件的副本：

<安装路径>/**user**/sinumerik/hmi/cfg

<安装路径>/**oem**/sinumerik/hmi/cfg

配置文件"slamconfig.ini"

通过文件"slamconfig.ini"中的以下段落来配置数值：

段落	说明
[ExitSoftkey]	针对"退出"软键的段落 在 NCU 的 SINUMERIK Operate 上，该配置会显示"HMI 重启"。
[RebootSoftkey]	针对"HMI 重启"软键的段落 该段落只针对 PCU/IPC 上的 SINUMERIK Operate。

值	说明
Visible	true 或 false 显示或隐藏操作区域内的软键
SoftkeyPosition	区域软键的固定位置 其中，软键位置 1 到 8 位于第 1 个水平软键栏，软键位置 9 到 16 位于第 2 个水平软键栏，以此类推。

值	说明
AccessLevel	0 到 7 显示软键的最低访问级别。 该值是为不支持 MD9110 的操作软件提供的兼容值。
QueryUser	true 或 false 启用或关闭确认对话框

说明

访问级别的兼容值

注意，在该版本的 SINUMERIK Operate 中，使用 AccessLevel 的值定义访问权限没有任何作用。访问权限通过 MD9110 设置。

更多说明参见"确定软键"HMI Reboot"（重新启动 HMI）的访问权限 (页 49)"一章。

示例

下文的示例是"slamconfig.ini"的标准配置：

```
[ExitSoftkey]
Visible=true
SoftkeyPosition=16
AccessLevel=1
QueryUser=false

[RebootSoftkey]
Visible=false
SoftkeyPosition=15
AccessLevel=1
QueryUser=false
```

更多信息

关于配置文件"slamconfig.ini"的更多说明参见"将 OEM 框架（OEMFrame）应用程序集成到系统中 (页 617)"一章。

4.11 调整操作界面的设计

机床数据显示

可通过以下显示机床数据调整操作界面的设计。

MD9112 $MM_HMI_SKIN	操作界面的设计（皮肤） 给定皮肤的编号
= 0	皮肤 0
= 1	皮肤 1（标准）
= 2	皮肤 2

在更改机床数据后必须重启操作软件。

4.12 设置屏幕保护

确定关闭显示的时间（屏幕保护）

可通过显示机床数据 MD9006 $MM_DISPLAY_SWITCH_OFF_INTERVAL 确定无任何键盘操作后多长时间则关闭显示器（分钟）。

注意
关闭屏幕对使用寿命的影响 建议的屏幕关闭时间为 60 分钟，以延长使用寿命。 如果大幅缩短了屏幕关闭时间（即：频繁关闭屏幕）会缩短屏幕使用寿命。

接口信号

通过设置接口信号关闭屏幕显示。

DB 19 字节	**发送到操作面板的信号(PLC → HMI)**	
DBB0	位 1	位 0
	关闭屏幕显示	打开屏幕显示

通过接口信号激活显示关闭时：

- 不能再通过按键重新恢复显示。
- 第一次在操作面板前端的按键操作已经触发了操作处理。

说明

为了避免通过接口信号关闭屏幕显示时误操作，我们推荐也同时锁定键盘。

4.13 设置操作面板的清洁模式

使用清洁模式可在不改变屏幕上显示的数据的情况下清洁操作面板。激活功能“面板清洁模式”后，操作面板的多点触碰和按键功能会在数秒内锁定。进度条为您显示剩余时间。

通过操作区域“调试”中的一个特殊软键可以激活操作面板清洁模式：

“调试” → “面板清洁模式”

说明

清洁模式只用于有触摸屏的操作面板。

机床数据显示

通过下列显示的机床数据可确定功能锁定的时长。

MD9116 $MM_PANEL_CLEAN_MODE_TIME	清洁模式持续时间
= 30	请确定清洁模式激活时的持续时间，单位秒。 设置范围 10 到 60 秒，标准时长为 30 秒

4.14 添加用户自定义引导启动画面

OEM 自定义的引导启动画面

西门子引导启动画面可以替换成自定义的引导启动画面。

步骤：

1. 命名自定义引导启动画面“logo.png”。

   说明

   文件名只能由小写字母构成。

2. 将自定义引导启动画面保存在下面其中一个目录下：
   /**user**/sinumerik/hmi/ico/ico640 或
   /**oem**/sinumerik/hmi/ico/ico640 或
   /**addon**/sinumerik/hmi/ico/ico640

   说明

   画面的大小没有影响，分辨率始终设为 640x480 像素
   。

4.15 并行使用操作界面

SINUMERIK Operate 能够实现在一个 NCU 上同时使用多个操作界面。但仅可将其中一个操作界面用作主界面，所有其他界面不是活动界面，只提供部分操作。为了避免各个操作界面与控制系统之间发生功能冲突，请调整以下配置文件：

文件“systemconfiguration.ini”

请在下面其中一个目录下新建配置文件“systemconfiguration.ini”：

<安装路径>/**user**/sinumerik/hmi/cfg

<安装路径>/**oem**/sinumerik/hmi/cfg

请在配置文件的段落**[miscellaneous]**中写入参数 HMIFunctionMode 并设置合适的值。可以设定以下值：

值	说明
PanelMode 或 <empty>	操作软件和系统的默认设置。 如果 SINUMERIK Operate 同时在另外的操作界面上运行，则可导致系统出错。
InterfaceMode	SINUMERIK Operate 在系统中切换为非活动状态并不作为正式操作界面工作。操作软件： • 不对 PLC 接口进行写访问 • 不监控 PLC 硬键 • 不监控 MMC 指令 • 不通过 PLC 监控语言切换 • 不监控 PLC 指令 • 不监控 Ctrl-Energy • 不通过 PLC 监控亮度调整 • 不在 PLC 中设置 HMI-Ready
TerminalMode	SINUMERIK Operate 在系统中作为装刀站使用。操作界面处于非活动状态。

示例

在一台数控机床上，两个操作界面在同一系统上使用一个 NCU 运行。

主操作界面的设置：

```
[miscellaneous]
HMIFunctionMode=PanelMode
```

副操作界面的设置：

```
[miscellaneous]
HMIFunctionMode=InterfaceMode
```

更多信息

关于操作界面上接口信号（DB19）和指令的功能和设置的更多说明请参见：

- PLC 功能手册
- NC 编程手册
- Easy XML 编程手册

4.16 在存档时设置登录数据处理

如果您生成一个调试文档，SINUMERIK Operate 会弹出一个对话框，询问是否对已设置的驱动器和 OPC UA 的全部登录数据都进行存档保存。您可以在以下的机床数据显示中为登录数据处理进行参数设置：

机床数据显示

MD9112 $MM_SAVE_CREDENTIALS	在存档的同时备份网络驱动器的访问权限
= 0	对话框高亮（标准）
= 1	无对话框，访问权限未备份
= 2	无对话框，访问权限始终备份

说明

使用按键组合 CTRL+ALT+S 无法将登录数据接收至存档。

4.17 VNC 远程连接

可在 SINUMERIK Operate 中通过 VNC 操作外部系统：

- 确定远程访问并通过一个已配置的软键将此选择接入操作区域。
- 确定按键转发，以便通过按键组合灵活使用 VNC 操作区域操作面板的操作功能。

远程连接由操作区域菜单的一个指定软键启动。激活了 Sidescreen 时，还可以通过 Sidescreen 的导航条来调用 VNC 操作区域。

说明

VNC 操作区域仅支持 16 或 32 位颜色深度的 Hextile 编码。

前提条件

在工厂或公司网络中，至少有一个系统带有设置的 VNC 服务器。

示例

可以是下列使用情形：

- 与其他控制系统的连接
- 与 CAD 系统的连接
- 与刀具测量站点的连接
- 与其他 PC 的连接

操作步骤

在窗口“操作区域设置”中进行远程连接的设置：

操作区“调试” → “HMI” → “操作区域菜单”

如需要设置额外的按键转发或设置下拉列表“显示 VNC 源”，按下软键“详细”。

可以为“选择菜单”键确定其他方式的快捷键。

说明

预留软键

系统预留软键已被标出并不可用于自由设置。

一般说明

条目		含义
操作区域 1 - 24		
类型	VNC	对外部控制器或 PC 的远程访问
	空	已配置的 VNC 操作区域被删除
计算机名称		外部系统的名称或 IP 地址
密码		VNC 服务器的登录密码 密码通过“*”加密显示。
状态栏	隐藏 上对齐 底部	VNC 访问信息的显示位置： • 系统名称或 IP 地址 • 端口 • 连接状态（比如：错误连接）
端口		到 VNC 服务器的接口 标准端口为 5900。

条目		含义
显示 VNC 源 （按下软键“详细信息”后可见）	截图 左对齐 宽度 上对齐 高度 全屏	确定所显示的区域 VNC Viewer 中显示的是整个源图。 原则上，必须延伸或压缩源图（必要时改变页面比例），才能在 VNC Viewer 中完整的显示。 通过配置截图的大小和位置您可以确定您想在 VNC Viewer 中看到的 VNC 源图的区域。 同样，也须延伸或压缩截图（必要时改变页面比例），才能在 VNC Viewer 中完整的显示。
Operate 标题	显示 隐藏	操作区域的显示栏、“有效/无效”的消息栏

针对所配置软键的说明

条目		含义
访问级别	钥匙开关位置 0 ... 制造商	设置访问各实时连接的权限：从访问级别 7（钥匙位置 0）到 1（制造商）。
软键文本		两行式软键文本 %n 为分行符。 若第一行太长，文本会自动换行。 若存在空格，空格被视为分行符。 对于与语言相关的软键文本，可以输入文本 ID 在文本文件中搜索。 如未在输入栏中输入任何内容，则会将自动给定的操作区域符号名称作为软键文本 （比如 AreaVNC_500）。
软键图标	无图标	该软键没有图标。
	hd_vnc.png 	软键图标的文件名。

条目		含义
文本文件		某个语种的软键文本文件以及语境。若未在输入栏中输入任何信息，则软键上显示的文本和在“软键文本”输入栏中给出的一样。
文本语境		

连至外部控制器的按键转发

按下软键“详细”，显示“按键转发”区域，然后勾选对应的复选框。

按下“Menu Select”后，会额外打开一个窗口，其中您可以选择所需快捷键。

说明

请注意，取决于具体使用的操作面板类型，下表列明的所有按键不一定都提供。

名称	快捷键	含义
M	Shift+F10	调用“加工”操作区域 （M 键）
Alarm Cancel	Esc	删除报警和显示信息
Machine	Shift+F11	调用“加工”操作区域（硬键）
Program Manager	Home（数字区）	调用“程序管理器”操作区域
Channel	F11	多通道时继续切换
Program	End（数字区）	调用“程序”操作区域
Alarm	Page Up （数字区）	调用“诊断”操作区域
Menu Select	F10	打开操作区域菜单 通过输入域更改按键组合。对话框“Menu Select 的备选按键组合”在尝试编辑时自行打开并准备进行以下的参数设置： A - Z, F1 - F12, Home, End, Page Up, Page Down 可与 SHIFT, CTRL, ALT 组合
Help	F12	调用所选窗口中和上下文相关的在线帮助
Offset	Page Down （数字区）	调用“参数”操作区域
Custom	Shift+F12	调用“Custom”操作区域
Next Window	Home	窗口切换键

自动按键转发

选中了 VNC 操作区域后，以下按键及快捷键便会始终直接转发给 VNC 服务器：

操作面板的按键	快捷键
水平软键（上边栏，HU）	F1 - F8
垂直软键（右边栏，VR）	Shift+F1 - Shift+F8
水平软键（下边栏，HL）	Ctrl+F1 - Ctrl+F8
垂直软键（左边栏，VL）	Shift+Ctrl+F1 - Shift+Ctrl+F8
Recall	F9
ETC 切换	Shift+F9
扩展水平软键（上边栏，HU）	F13 - F20
扩展垂直软键（右边栏，VR）和右侧直接按键 HT8	Shift+F13 - Shift+F20
扩展水平软键（下边栏，HL）	Ctrl+F13 - Ctrl+F20
扩展垂直软键（左边栏，VL）和左侧直接按键 HT8	Shift+Ctrl+F13 - Shift+Ctrl +F20
底边及左侧软键栏（HL，VL）	(Shift)+Ctrl+F1 - (Shift)+Ctrl +F8

4.18 设置 WEB 浏览器

SINUMERIK Operate 中可以设置额外的操作区域“WEB 浏览器”。

WEB 浏览器可直接在控制系统上显示网页。

根据具体需求完成相应的 WEB 浏览器配置。

WEB 浏览器通过操作区域菜单中的一个指定软键启动。激活了 Sidescreen 时，还可以通过 Sidescreen 的导航条来调用 WEB 浏览器。

前提条件

控制系统接入了互联网、内网或者和一个网络服务器相连。

示例

如需获取关于操作的更多信息，请访问互联网或内网网页。

操作步骤

在窗口“设置操作区域”中设置 WEB 浏览器：

操作区“调试” → “HMI” → “操作区域菜单”

一般说明

条目		含义
操作区域 1 - 24		
类型	WEB Browser	用于在控制系统上显示网页
启动 URL		启动 WEB 浏览器时显示的 URL。
地址输入	显示 隐藏	WEB Browser 中 URL/网址的输入栏
状态显示	显示 隐藏	WEB 浏览器中的状态显示，例如：显示 URL 的加载进度
搜索	显示 隐藏	WEB 浏览器中的搜索栏
主页 URL		设置了主页 URL 时，在 WEB 浏览器中便会显示对应的图标。 按下主页-URL 后，便会调用主页-URL。
Operate 标题	显示 隐藏	操作区域显示栏和消息栏
导航	显示 隐藏	WEB Browser 中的导航箭头

针对所配置软键的说明

条目		含义
访问级别	钥匙开关位置 0 ... 制造商	所配置软键的操作性 存取级别 7 - 入门操作员 ... 存取级别 1 - 制造商
软键文本		不同语言文本的文本 ID 或两行式软键文本 %n 为分行符。 若第一行太长，文本会自动换行。若存在空格，空格被视为分行符。 对于与语言相关的软键文本，可以输入文本 ID 在文本文件中搜索。
软键图标	无图标	该软键没有图标。
	sk_web.png 	软键图标的文件名。
文本文件		某个语种的软键文本文件以及语境。如未在输入栏中输入任何内容，则此处使用“软键文本”中输入的、所有语言适用的文本。
文本语境		

4.19 访问等级

4.19.1 定义存取权限

针对不同用户对程序、数据和功能的访问需求，系统将访问权限分为 7 个等级。它们分为

- 3 个口令，供制造商、服务人员和用户使用
- 4 个钥匙开关位置，用于用户

下表列出了访问等级 1 到 7，其中

- 1 表示最高等级，
- 7 表示最低等级。

表格 4-1　访问等级概念

访问等级	保护方式	范围
1	口令：SUNRISE（缺省值）	制造商
2	口令：EVENING（缺省值）	服务人员
3	口令：CUSTOMER（缺省值）	用户
4	钥匙开关位置 3	编程员，调试员
5	钥匙开关位置 2	合格的操作员
6	钥匙开关位置 1	受过培训的操作员
7	钥匙开关位置 0	刚刚入门的操作员

口令安全

说明

设定安全口令

在设定新口令时请遵循以下规则：

- 在设定新口令时，请勿设定易于猜测的口令，例如简单的词、键盘上的易于猜测的按键序列等。
- 口令必须包含大小写字母、数字和特殊字符。口令最少须包含 8 个字符。PINS 必须是无规律的数字序列。
- 在可行且 IT 系统和软件支持的情况下，在选择口令时必须采用尽可能复杂的字符。

在采用口令的情况下，可以使用用于口令管理的程序来加以协助。借助该程序能够对口令和密码进行加密保存、管理以及生成安全口令。

说明

SINUMERIK Operate 与 NC 之间的口令更改

SINUMERIK Operate 和 NC 的访问级别是合在一起的，在修改 SINUMERIK Operate 的口令的同时也会将 NC 的相关口令修改，反之亦然。必须注意下列事项：

- 在执行 NC 清零时，口令不会恢复为标准口令。
- 在软件升级后，SINUMERIK Operate 口令适用于 NC。
- 口令一经修改，便无法恢复为标准口令，即使通过西门子服务也无法恢复。恢复口令会违反信息安全的要求。
  因此请务必妥善保管修改过的口令！
  如果口令遗失，则只能通过备份文件重新安装整个 SINUMERIK 系统（还原）。
  有关备份的**更多信息**，请参见 NCU 操作系统调试手册。

4.19.2 编辑存取权限的口令

概述

您可以在操作界面上激活受口令保护的访问等级。

可以选择下列方式编辑口令：

- 设置新口令
- 修改口令
- 删除口令

设定口令



1. 选择操作区域“调试”。



2. 按下软键“口令”。



3. 按下软键“设定口令”。
   “设定口令”窗口打开。



4. 输入一个允许的默认口令，按下软键“确认”确认输入。
   允许的口令会成功设置，屏幕上显示有效的访问等级，而无效的口令会被拒绝。

修改口令

必须修改默认口令，提高访问等级安全性。



1. 按下软键“修改口令”。
   “修改口令”窗口打开。
2. 选中需要设置新口令的范围（如用户）。
3. 在输入栏“新口令”和“重复口令”中输入新口令。
   请始终设置一个安全口令。



4. 按下软键“确认”，确认输入。
   只有当两次的输入一致时，新的有效口令才会被接收。

说明

无效的口令会被操作软件拒绝。请注意设置安全口令的规则，例如：

- 最少包含 8 个字符
- 最少包含一个大写字母
- 最少包含一个小写字母
- 最少包含一个数字
- 最少包含一个特殊字符（ASCII 0x20 - 0x7E）

删除口令



按下软键“删除口令”。访问权限再次被取消。

说明

上电不会自动删除访问权限。

4.19.3 工作场所安全

为防止意外操作机床并避免人身伤害事故，在离开工作场所时需要执行以下操作：

1. 将钥匙开关置于 0 位并拔下钥匙。



2. 按下软键“删除口令”。
   访问权限复位。

4.19.4 程序的访问等级

目标用途

设置访问权限一方面可以确保信息安全，另一方面可以防止控制系统或机床被非法操作或被蓄意损坏。

在文件系统的某个文件和目录上设置访问权限，便可以控制各种用户对该文件或目录的操作。这里的操作指：

- 读取文件内容(READ)，复制文件或生成文件存档。
- 修改文件内容(WRITE)，或删除文件。
- 执行文件(EXECUTE)。
- 列出并显示某个目录中的某个文件(SHOW)。
- 在某个目录内新建文件。
- 显示或读取某个目录的内容。
- 新建或删除某个目录。

功能说明

上述单项操作的访问权限的设置 针对 NC 文件系统中的以下目录生效：

- 零件程序
- 子程序
- 制造商循环目录
- 用户循环目录
- 工件目录
- 工件目录

访问权限的设置还针对其中的主程序文件“*.mpf”、子程序文件或循环文件“*.spf”生效。

另外，访问权限还可以扩大到本地驱动器上的以下目录：

- 零件程序
- 子程序

- 工件目录
- 工件目录

说明

远程访问时的访问权限

在远程访问控制系统时，例如通过 WinSCP 或者 Access MyMachine 访问，在新建或者编辑文件和目录方面享有和直接在控制系统上访问一样的权限。

更多信息

对 WinSCP 或 Access MyMachine 访问权限的说明参见以下手册：

- NCU 操作系统调试手册
- SINUMERIK Integrate Access MyMachine /OPC UA

通过机床数据设置访问权限

可以用下面的机床数据来**统一 设置访问权限**：MD51050 $MNS_ACCESS_WRITE_PROGRAM

默认设置为 4（访问等级：用户）

说明

设置冲突时访问权限的效力

如果不仅设置了单项操作的访问权限，还通过"MD51050 $MNS_ACCESS_WRITE_PROGRAM"设置了统一访问权限，一旦两者冲突，则其中限制性更强的设置生效。

文件 "slfsfileattributes.ini" 中的设置

文件 "slfsfileattributes.ini" 作为模板，位于以下目录中：

../siemens/sinumerik/hmi/template/cfg

值 1 至 7 对应访问等级，跟它们在有口令时的设置一样。4 位数的数字表示以下权限的顺序：

从左→右的含义：R W X S		
R	READ	读取文件和目录
W	WRITE	修改、删除或者创建文件和目录
X	EXECUTE	执行程序：文件；如果是目录，则始终为 7
S	SHOW	输出文件和目录的列表

将文件 “slfsfileattributes.ini” 复制到以下一个目录中：

../**oem**/sinumerik/hmi/cfg 或 ../**user**/sinumerik/hmi/cfg

重新启动操作软件后，该文件修改才能生效。

示例

路径由数据类型命名符组成。在下面的例子中，“\wks.dir*.wpd*.mpf”是位于工件目录 “wks.dir”下工件“*.wpd”的零件程序 “*.mpf”。

所有工件的新建零件程序的默认访问权限为 7777。

所有条目都位于段落[ACCESSMASKS]中，格式如下：

```
[ACCESSMASKS]

\wks.dir\*.wpd\*.mpf = 7777
\mpf.dir\*.mpf = 6577
\cus.dir\*.spf = 6577
```

参见

定义存取权限 (页 62)

4.19.5 文件的访问权限

修改访问权限

在“属性”窗口中显示了执行、写入、列明和读取文件的权限。

- 执行：选择文件加以执行。
- 写入：控制文件或目录的修改和删除操作。

可为每个 NC 文件单独设置访问权限：从钥匙开关 0 到当前的访问等级。

如果文件/目录所需的访问等级高于您当前的访问等级，则无法进行设置。

对于外部文件而言，比如本地驱动器上的文件，此处只向您显示机床制造商为这些文件设置的访问权限。您无法在属性窗口中进行任何修改。

说明

设置目录和文件的访问权限

通过配置文件和 MD51050 您可以对 NC 存储器和用户存储器（本地驱动器）上的目录和文件的访问权限进行修改和预设。

配置文件 “slfsfileattributes.ini”

在 NC 中创建文件时，配置文件“slfsfileattributes.ini”和 MD51050 中设置的访问权限开始起作用，您可以随后在程序管理器的功能“属性”中单独修改。在 NC 中，访问权限是每个文件的单独属性。

在访问 NCU CF 卡中的外部数据或者访问 PCU/PC 的硬盘时，配置文件“slfsfileattributes.ini”和 MD51050 中设置的访问权限对所有指定目录和文件起作用，访问权限通过操作无法加以修改。外部数据的访问权限只能统一设置，不能单独设置。

4.19.6 软键的访问等级

使用

软键的显示和操作不仅可以由制造商隐藏，也可以由用户隐藏，以便使操作软件更加符合实际功能范围的需要，使操作界面更加清晰简单。

为此需要对控制系统的功能范围加以限制，以取消对操作软件某些功能的访问，降低误操作几率。

访问权限的等级

只有当前访问等级能够查看并操作的软键才可以被重新设置访问等级。

示例：一个软键只能由访问等级“制造商”访问，因此它的访问等级不能被“钥匙开关 3”修改。

前提条件

为了确保系统的稳定运行和可操作性，有些软键的访问等级是禁止被修改的，例如：垂直软键条的软键“<<”，或软键“调整软键”本身，以便可以再次恢复软键显示/操作。

另外，以下软键也禁止被隐藏或修改访问等级：

- 所有“确认”软键
- 所有“取消”软键
- 所有“接收”软键
- 所有“**<<**”和“**>>**”软键

循环窗口中软键的访问等级也可以被修改，但“Easy Screen”窗口中的软键不可以。

说明

软件升级

由于软键访问等级的所有改动都通过软键 ID 加以指明，因此即使在可能引起软键更新的软件升级后，也仍然保持有效。

对于在访问时由机床数据如：MD51045 $MNS_ACCESS_TEACH_IN 加以保护的软键，在设置新的访问等级后，就只由新的访问等级保护。系统不会再考虑该机床数据。

所有通过这种方法隐藏的软键都可以暂时通过系统默认的访问权限显示，按下软键“显示所有软键”即可。只要当前还有软键保持旧访问等级，该软键就一直有效。重启系统后，软键访问等级的设置才生效。

说明

已更改的软键访问等级的可用性

在 PCU 上设置软键访问等级仅作用于各个 PCU 软键本身。为切换 NCU 上的访问等级，制造商和用户必须使用合适的装置及对应的权限，参见章节“程序的访问等级 (页 66)”。

4.19.7 确定软键的访问等级的步骤

一般操作顺序

按如下步骤为软键分配新的访问等级：

- 激活软键调整模式。
- 修改软键的访问等级。
- 接收并确认修改。

前提条件

对软键“调整软键”的访问受到通用机床数据 MD51073 $MNS_ACCESS_SET_SOFTKEY_ACCESS 的保护，该机床数据的默认访问等级是 3，也就是说：从口令“服务人员”起才可以使用该功能。将该数据设为大于 3 的值，访问等级为“用户”的操作人员便可使用该功能。

调整软键的访问等级时，需要使用一个外部鼠标。

修改软键的访问等级

步骤

1. 按下操作区域“调试”中的软键“HMI”。
2. 按下“**>>**”，按下“调整软键”。
   接着会弹出一个对话框，给出详细的操作提示。
3. 按下“确认”，激活“调整软键访问等级”模式。随即显示一张列表。
4. 选择一个操作区域。
5. 将鼠标光标移到一个软键上，点击鼠标右键打开右键菜单“确定访问等级”：
   - 当前显示的访问等级是默认等级。
   - 点击“始终隐藏”，使该软键隐藏。
   - 点击“恢复设置”，再次清除额外指定的访问权限。该选项只有在已经设置了单个访问等级后才会出现。
6. 返回“调试”操作区域，按下软键“结束调整”，接收修改。现在，便退出了“调整软键访问等级”模式。
   按下“确认”确认弹出的询问，接收修改，修改在系统重启后生效。
7. 如果希望取消软键的调整，可在“调整软键访问等级”窗口中选择一列，然后按下软键“恢复设置”。在“新访问等级”一列中输入标识“恢复设置”。软键调整结束和系统重启后，恢复软键的原厂设置并删除窗口“调整软键访问等级”中的行条目。

示例

通过上述步骤修改了以下访问等级：

软键	新访问等级
SISuMainScreenMenuHu::0::3	制造商
SIMaJogMillMenuHU::jogHuMore::6	用户
SISuPaUserDataMenu::::4	服务人员

在“软键”列中记录了选择软键时软键的标识和位置。

结果

一旦接收了软键调整，系统便会生成一份名为“slsoftkeyaccess.xml”的配置文件，它位于文件系统的以下目录下：../**user**/sinumerik/hmi/cfg。在**/oem** 下也可以保存一些在系统启动时参与计算的文件。此时的优先级如下：oem < user。

说明

编辑文件“slsoftkeyaccess.xml”的注意事项

为了避免 xml 句法错误和软键标识错误，只允许通过操作软件按照上文说明的步骤来编辑文件“slsoftkeyaccess.xml”，而不允许直接在文件系统的“系统数据”中编辑文件。否则可能会导致无法编译整个文件或个别软键条目。

4.19.8 通过保护等级设置 MDA 程序访问

通过以下通用机床数据设置，自哪个保护等级起可在 MDA 运行方式下编辑程序。该设置也适用于自动运行方式下的程序覆盖。访问级别可以为 0 至 7。

设置保护等级

MD51075 $MNS_ACCESS_WRITE_MDI	写入 MDA 程序的保护等级
= 7	无联锁，允许编辑（默认）

说明

上电后激活保护等级变更。

如果当前访问级别不够高，则在 MDA 运行方式下既不能编辑程序，也不能通过软键“加载 MDA”和“删除程序段”来更改。

4.20 程序选择

4.20.1 显示软键

操作界面上的软键可以由通用的配置机床数据激活，这些软键可以调用程序列表。

设计软键

MD51041 $MNS_ENABLE_PROGLIST_USER	
0	无软键
1	显示软键"Prog.-liste"（程序列表）

MD51043 $MNS_ENABLE_PROGLIST_MANUFACT	
0	无软键
1	显示软键"MANUFACTURER"（制造商）

前提条件

创建程序列表需要下列权限：

- 操作区域“程序管理器”：访问等级 3（口令：用户）或者
- 操作区域“调试”：访问等级 1（口令：制造商）

程序列表

您可以创建一个程序列表，列表包含了从 NC 数据中提取的所需程序，例如：工件、零件程序、子程序。 然后通过 PLC 可选择该程序进行处理。

程序列表	目录
plc_proglist_user.ppl	/**user**/sinumerik/hmi/plc/programlist
plc_proglist_manufacturer.ppl	/**oem**/sinumerik/hmi/plc/programlist

步骤



1. 如果访问权限等级是 1（口令：制造商），则选择操作区域“调试”。

-或者-



如果访问权限等级是 3（口令：用户），则选择操作区域“程序管理器”。



2. 按下菜单扩展键和软键“程序列表”。
   窗口“程序列表”打开。



3. 如果要创建一个程序列表，该列表可用于特定系列的所有机床，按下软键“Manufacturer”。
4. 将光标定位到所需的行中（程序编号）。



5. 按下软键“选择程序”。
   窗口“程序”打开。会显示出带有“工件目录”、“零件程序目录”和“子程序目录”的 NC 存储器的文件树形图。



6. 将光标定位到所需的程序上并按下软键“确认”。
   所选定的程序及其路径信息被置于列表的第一行。
   -或者-
   直接在列表中输入程序名称。
   在手动输入时要注意准确的路径信息（例如 //NC/WKS.DIR/MEINPROGRAMM.WPD/MEINPROGRAMM.MPF）。
   有时需要加上后缀名（.MPF）。
   提示：
   系统不检查句法。
   对于工件请注意，在工件中主程序或工作列表的名称相同。



7. 要从列表中删除程序，可以将光标定位到相应的行上并按下软键“删除”。
   -或者-



如要从程序列表中删除所有的程序，可以按下软键“全部删除”。

4.21 激活并配置 Sidescreen 窗口

4.21.1 激活 Sidescreen

宽屏格式的多点触控面板可在 SINUMERIK Operate 两侧提供额外的屏幕区域，用于显示更多的屏幕元素。默认显示小部件，可允许您快速且持续地查看重要信息。除了页面外，虚拟键盘和虚拟 MCP 的功能也在准备之中。此外还可配置自定义页面。

更多信息

有关 Sidescreen 的更多信息和示例，请参见以下文档：

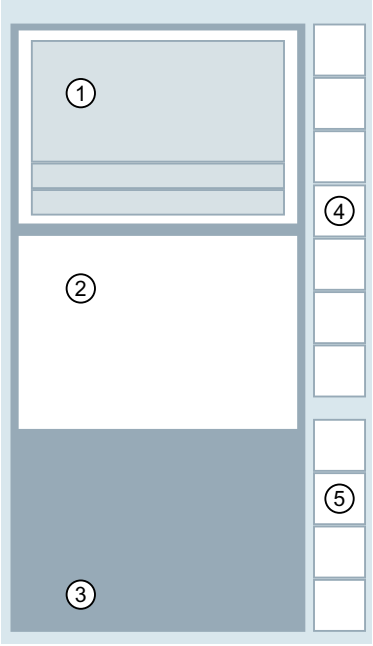
- Easy XML 编程手册
- Run MyScreens 编程手册
- 操作手册，通用版/车削版/铣削版/磨削版

前提条件

该功能需要使用宽屏格式的多点触控面板（分辨率：1366x768、1920x1080）。

Sidescreen 结构

在 Sidescreen 中可以显示多个页面。其中始终只有一个页面生效。通过页面选择键，可以在页面之间进行切换。
默认页面包含用于显示小部件的默认元素。您可在该页面上设置附加元素。



① 西门子默认元素中的小部件
② 默认页面上的元素 (OEM)
③ 页面
④ SINUMERIK Operate 导航栏
⑤ 页面选择键

激活

⚠ 警告

错误操作可导致机器意外运动

错误操作有触摸屏的设备可能导致机器意外运动。这可能导致人员受伤或财产损失。

- 通过适当的界面布局来避免无意造成的错误操作，比如可以在机器组件的启动按钮和停止按钮之间留足够的距离。
- 注意在设置时不要让与安全相关的功能可通过触摸屏设定。

通过以下显示机床数据来激活 Sidescreen 功能：

MD9114 $MM_SIDESCREEN	SINUMERIK Operate Sidescreen
= 1	启用多点触控面板的额外 Sidescreen 显示界面，准高清或全高清。
= 2	启用 Sidescreen 显示界面，准高清。 启用 Display Manager，全高清。

说明

自动隐藏 Windows 任务栏

SINUMERIK Operate 在 PCU/IPC 上有时会被 Windows 任务栏盖住。为了不影响的可操作性，您可设置任务栏自动隐藏：

用鼠标右键点击任务栏并在菜单中选择“属性”（Windows 7）或“任务栏设置”（Windows 10）。然后在选项卡“任务栏”下激活选件“任务栏自动隐藏”（Windows 7）或“桌面模式下任务栏自动隐藏”（Windows 10）。

通过机床数据可以定义小部件的棒图显示和数值显示：

轴负载显示

软件选件

使用该功能需要下列软件选件：

“内部驱动变量的计算”

通过以下轴机床数据为 Sidescreen 中的轴负载显示激活对额外的驱动实际值的采集

MD36730 $MA_DRIVE_SIGNAL_TRACKING	额外的驱动实际值的采集
= 1	$AA_LOAD:驱动负载率 $AA_POWER:驱动有功功率 $AA_TORQUE:驱动力矩设定值 $AA_CURR:驱动的经平滑的电流实际值（贯穿电流）

说明

仅当驱动提供实际值时，才能实现针对轴负载显示的采集。

轴和主轴功率的柱状图显示

通过以下轴机床数据对轴功率的柱状图显示进行参数设置：

MD53030 $MAS_AXIS_MAX_POWER	轴功率显示的最大值
= 100	标准 允许的轴功率的最大值，以百分比为单位 在 0 与 AXIS_MAX_POWER 的值之间的区域内，柱状图显示采用绿色。

MD53031 $MAS_AXIS_POWER_RANGE	轴功率显示的显示范围
= 100	标准 轴功率显示的标度值，以百分比为单位 这个值必须大于等于 AXIS_MAX_POWER。在 AXIS_MAX_POWER 与 AXIS_POWER_RANGE 的值之间的区域内， 柱状图显示采用红色。

通过以下通用机床数据对主轴功率的柱状图显示进行参数设置：

MD51030 $MNS_SPIND_MAX_POWER	主轴功率显示的最大值
= 100	标准 允许的主轴功率的最大值，以百分比为单位 在 0 与 SPIND_MAX_POWER 的值之间的区域内，柱状图显示采用绿色。

MD51031 $MNS_SPIND_POWER_RANGE	主轴功率显示的显示范围
= 100	标准 主轴功率显示的标度值，以百分比为单位 这个值必须大于等于 SPIND_MAX_POWER。在 SPIND_MAX_POWER 与 SPIND_POWER_RANGE 的值之间的区域内，柱状图显示采用红色。

说明

不需要使用标准小部件时，可以在模板文件“slsidescreen.ini”中的对应关键字前添加注释符“;”。

4.21.2 显示 Sidescreen ABC 键盘

在多点触控面板的 Sidescreen 显示区域中，除了可显示标准窗口，还可显示 ABC 键盘窗口。

步骤

1. 从目录 /siemens/sinumerik/hmi/template/sidescreen 下
   复制样板文件“slsidescreen.ini”
2. 将副本保存到目录：
   /**oem**/sinumerik/hmi/cfg 或
   /**user**/sinumerik/hmi/cfg
3. 打开副本并删除关键字 `PAGE002` 前的注释符“;”。

   说明

   属性 `name` 和 `implementation` 的值不得修改。

4. 保存设置。
   ABC 键盘可在启动后选择。

4.21.3 显示机床控制面板 Sidescreen

在多点触控面板的 Sidescreen 显示区域中，除了可显示标准窗口，还可显示机床控制面板功能的窗口。

还可以选择显示多个机床控制面板窗口。

按照以下步骤可显示机床控制面板的样板：

步骤

1. 从目录 /siemens/sinumerik/hmi/template/sidescreen/mcp/ 下
   复制样板文件“slsidescreenmcppageconfig.xml", "slsidescreenmcppage_deu.ts" 以及图标文件的子目录。
   图标文件位于以下子目录中：
   /ico1024，适用于准高清显示
   /ico1920，适用于全高清显示
2. 将"slsidescreenmcppageconfig.xml"的副本保存到目录：
   /**oem**/sinumerik/hmi/proj 或
   /**user**/sinumerik/hmi/proj
3. 将"slsidescreenmcppage_deu.ts"的副本保存到目录：
   /**oem**/sinumerik/hmi/lng 或
   /**user**/sinumerik/hmi/lng

   说明

   TS 文件在启动时自动被系统集成。

4. 将图标文件保存到以下路径的子目录中：
   /**oem**/sinumerik/hmi/ico 或
   /**user**/sinumerik/hmi/ico
   屏幕分辨率不同，子目录也不同：
   /ico1024，适用于准高清显示
   /ico1920，适用于全高清显示
5. 从目录 /siemens/sinumerik/hmi/template/sidescreen 下
   复制样板文件“slsidescreen.ini”
6. 将"slsidescreen.ini"的副本保存到目录：
   /**oem**/sinumerik/hmi/cfg 或
   /**user**/sinumerik/hmi/cfg
7. 打开文件"slsidescreen.ini"的副本并删除关键字 `PAGE003` 前的注释符“;”。
   提供多个机床控制面板作为选择：
   `PAGE004`、`PAGE005` 和 `PAGE006`

   说明

   属性 `name` 和 `implementation` 的值不得修改。

8. 保存设置。
   机床控制面板可在启动后选择。

4.21.3.1 设置机床控制面板的自定义按键

在 Sidescreen “机床控制面板” 上也可显示自定义按键。为此应在文件“slsidescreenmcppageconfig.xml”中配置这些按键。在此可对属性、按键位置和标题加以确定。

示例文件 “slsidescreenmcppageconfig.xml”

```
<sideScreenControlPanel>
 <mainBlock id="sideScreenMcpBlock1">
  <block position="1" title="SPINDLE">
   <row position="1">
    <buttonGroup spacing="false" title="COOLANT">
     <button position="1" text="ON" icon="slsidescreenmcp_cooling.png"
style="redStyle" title="Spindle">
      <onPressed address="HmiUserKey1"/>
      <active address="HmiUserLed1"/>
      <disabled address="DisableHmiUserKey1"/>
     </button>
    <button position="2" text="OFF" icon="slsidescreenmcp_cooling_off.png">
      <onPressed address="HmiUserKey2"/>
      <active address="HmiUserLed2"/>
     </button>
    </buttonGroup>
   </row>
  </block>
 </mainBlock>
 ************************************************************************
 *** optional                                                         ***
 ************************************************************************
 <mainBlock id="sideScreenMcpBlock2">
  <.../>
 </mainBlock>
 <mainBlock id="sideScreenMcpBlock3">
  <.../>
 </mainBlock>
 <mainBlock id="sideScreenMcpBlock4">
  <.../>
 </mainBlock>
 <mainBlock id="sideScreenMcpBlockBelow1">
  <.../>
 </mainBlock>
 <mainBlock id="sideScreenMcpBlockBelow2">
  <.../>
 </mainBlock>
 <mainBlock id="sideScreenMcpBlockBelow3">
  <.../>
 </mainBlock>
 <mainBlock id="sideScreenMcpBlockBelow4">
  <.../>
 </mainBlock>
</sideScreenControlPanel>
```

文件 “slsidescreenmcppageconfig.xml” 的匹配项

在标签<sideScreenControlPanel>中可对自定义按键进行设置：

标签	描述	
<mainBlock id="sideScreenMcpBlock1">	机床控制面板窗口的内部名称	
<block position="1" title="SPINDLE">	主段落可有任意多个区块。	
	position	区块的布局
	title	区块的标题
<row position="1">	每个区块可有任意多行。	
	position	行的定位
<buttonGroup spacing="false" title="COOLANT">	一个按键组可在一个标题下包含两个按键。	
	spacing	通过该属性可将按键的显示方式设为相邻显示"false"或间隔显示"true"。默认为间隔显示。
	title	一个按键组的所有按键适用一个共同的标题。
<button position="1" text="ON" icon="slsidescreenmcp_cooling.png" style="redStyle" title="Spindle">	定义按键属性	
	position	按键的布局
	text	按键上显示的文本
	icon	按键的图标 此符号显示在 text 标记的上方。如果无标记，那么此符号要额外占据标记位置。
	style	更改按键的背景色方案 允许的值有： "redStyle" 用于红色按键 "greenStyle" 用于绿色按键 默认为 "defaultStyle"
	title	不归属任何组的按键可显示自己的标题。

标签	描述	
<onPressed address="HmiUserKeyX"/>	如果按下按键，PLC 位被置 1。 只要按键被一直按住，该值就保持为 1。如果是短时间按下，该值至少会保持 300 ms 为 1。	
	address	用于定址的 PLC 位的 Alias 名称 "HmiUserKeyX" 对于 X 而言允许的值为 1 – 64。 HmiUserKey1 至 HmiUserKey64 对应 DB19.DBB404.0 至 DB19.DBB411.7。
<active address="HmiUserLedX"/>	按键显示为生效（位于按键上边沿的 LED 动画）。	
	address	推导出生效标记的 PLC 位的 Alias 名称 对于 X 而言允许的值为 1 – 64。 HmiUserLed1 至 HmiUserLed64 对应 DB19.DBB412.0 至 DB19.DBB419.7。
<disabled address= "DisableHmiUserKeyX"/>	按键显示为无效（图标和文本变灰）。	
	address	推导出无效标记的 PLC 位的 Alias 名称 对于 X 而言允许的值为 1 – 64。 DisableHmiUserKey1 至 DisableHmiUserKey64 对应 DB19.DB420.0 至 DB19.DB427.7。

保存自定义图标

您可在以下路径下创建自定义图标文件：

/**oem**/sinumerik/hmi/ico 或

/**user**/sinumerik/hmi/ico

根据屏幕分辨率提供以下子目录：

/ico1024，适用于准高清显示

/ico1920，适用于全高清显示

说明

图标文件的名称必须与其在 XML 文件中使用的名称一致，例如 `icon="slsidescreenmcp_cooling.png"`。

4.21.3.2 机床控制面板文本的自定义翻译

机床控制面板的所有文本都可显示为不同的语言。翻译保存在文件“slsidescreenmcppage_xxx.ts”中。

“xxx” 为语言标识，例如德语为 “deu”，英语为 “eng” 等。支持的语种请见 “支持的语言 (页 614)”

步骤

1. 将文件“slsidescreenmcppage_xxx.ts”及需要的语言标识保存在以下目录：
   /**oem**/sinumerik/hmi/lng 或
   /**user**/sinumerik/hmi/lng
2. 打开语言文件并添加相应的名称。
   以下文本可显示翻译：
   - 所有的标题，例如 `title="COOLANT"`
   - 所有的按键名称，例如 `text="ON"`
3. 保存语言文件。
   翻译在启动和语言切换后显示。

文件 “slsidescreenmcppage_xxx.ts” 的匹配项

在标签`<message>`中可添加翻译。

标签	说明
`<source>ON</source>`	此为源语言的文本标识。
`<translation>EIN</translation>`	此为翻译。
`<chars>8</chars>`	此为翻译的最大长度。 此标签仅作为对翻译人员的提示。

文件 "slsidescreenmcppage_deu.ts" 的摘录（示例）

```
...
<message>
 <source>ON</source>
 <translation>EIN</translation>
 <chars>8</chars>
</message>
...
```

4.21.4 在 Sidescreen 中显示摄像头

在多点触控面板的 Sidescreen 显示区域中，除了可显示标准窗口，还可显示最多两个摄像头传输窗口。

说明

兼容性

NCU 710 不支持该功能。

步骤

1. 从目录 /siemens/sinumerik/hmi/template/sidescreen 下
   复制样板文件"slsidescreen.ini"
2. 将副本保存到目录：
   /**oem**/sinumerik/hmi/cfg 或
   /**user**/sinumerik/hmi/cfg
3. 打开副本并删除摄像头 1 的关键字 `ELEMENT009` 和摄像头 2 的关键字 `ELEMENT010` 之前的注释符";"。

   说明

   属性 `name` 和 `implementation` 的值不得修改。
4. 保存设置。
   摄像头可在启动后选择。

更多信息

有关设置摄像头的更多信息，请参见在线帮助

调试功能及服务。

4.22 SINUMERIK Operate Display Manager

4.22.1 Display Configuration

借助 Display Manager 和 Display Configuration，您可以缩小 SINUMERIK Operate，即操作界面不再全屏显示，以便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。您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。关于显示配置本身的定义则在单独的一份配置文件中进行。

系统条件

以下系统支持 Display Manager：

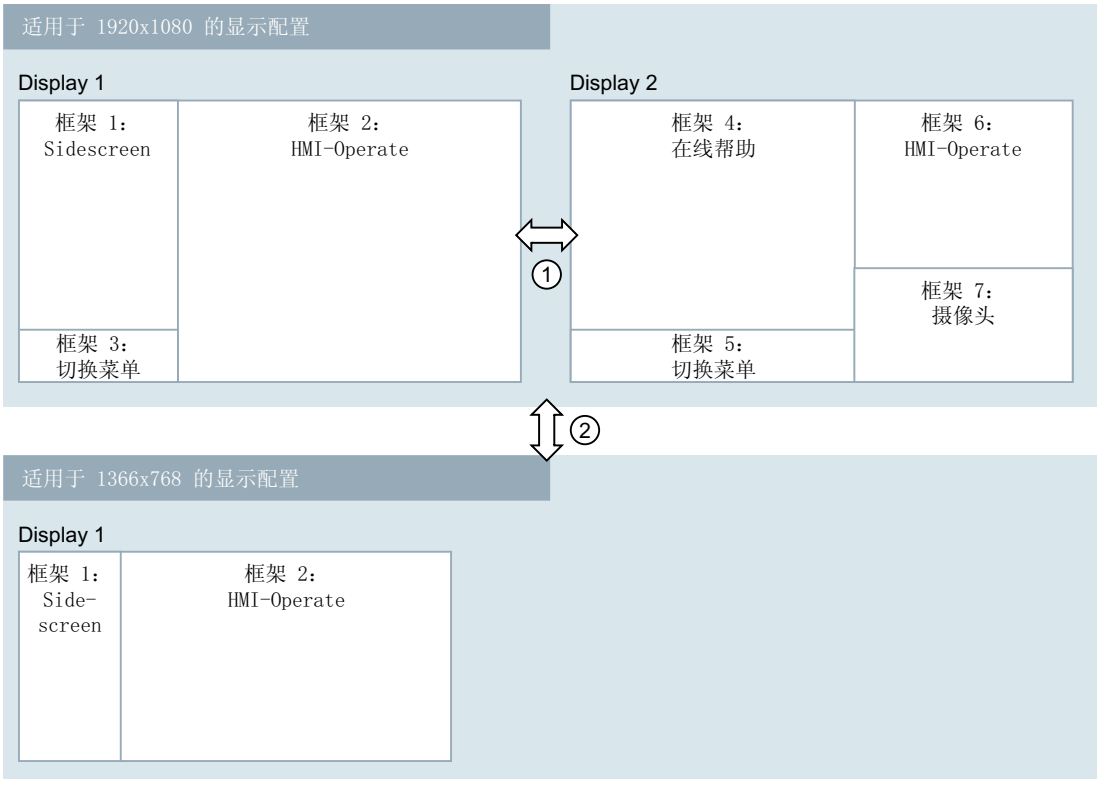
- PC, PCU/IPC
- NCU 720 和 NCU 730

更多信息

有关 Display Manager 的更多信息和示例，请参见以下文档：

- Run MyScreens 编程手册
- 操作手册，通用版/车削版/铣削版/磨削版

显示配置（示例）



① 使用切换菜单进行切换

② 使用 TCU 或分辨率切换进行切换

示例文件“systemconfiguration.ini”

```
[displayConfigurations]
DC001 = res:=1366x768, file:=sldm_1366x768
DC002 = res:=1920x1080, file:=sldm_1920x1080
DC003 = tcu:=TCU1, file:=tcu1_layout
```

示例中包含三项显示配置。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。属性"res"表示该显示配置适用的屏幕分辨率。通过属性"tcu"可以在不同的面板上显示不同的 Display Manager 配置。

说明

特殊性

- 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" → "res"
- 如果没有任何一个显示配置合适，则用户界面全屏显示。

激活

⚠ 警告

错误操作可导致机器意外运动

错误操作有触摸屏的设备可能导致机器意外运动。这可能导致人员受伤或财产损失。

- 通过适当的界面布局来避免无意造成的错误操作，比如可以在机器组件的启动按钮和停止按钮之间留足够的距离。
- 注意在设置时不要让与安全相关的功能可通过触摸屏设定。

设置以下显示机床数据，SINUMERIK Operate Display Manager 才可以激活。

MD9114 $MM_SIDESCREEN	SINUMERIK Operate Sidescreen
= 2	启用 Sidescreen 显示界面，准高清。 启用 Display Manager，全高清。

Display Manager 中提供了适用于分辨率 1920x1080 和 1366x768 的标准显示配置，分别位于以下文件中：

- sldm_1920x1080.ini
- sldm_1366x768.ini

这两份文件可用作模板，方便您自定义显示配置。本章我们将介绍如何自定义显示配置。

说明

自动隐藏 Windows 任务栏

SINUMERIK Operate 在 PCU/IPC 上有时会被 Windows 任务栏盖住。为了不影响的可操作性，您可设置任务栏自动隐藏：

用鼠标右键点击任务栏并在菜单中选择“属性”（Windows 7）或“任务栏设置”（Windows 10）。然后在选项卡“任务栏”下激活选件“任务栏自动隐藏”（Windows 7）或“桌面模式下任务栏自动隐藏”（Windows 10）。

4.22.2 自定义显示配置

4.22.2.1 Displays（画面）

对于每种显示配置，您都可以在对应的配置文件下定义一幅或多幅画面。您可以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。

说明

显示顺序可任意设置。

示例

```
[displays]
DISPLAY001= name:=display_1, frames:="frame1, frame2, frame3"
DISPLAY002= name:=display_2, frames:="frame4, frame5, frame6,
frame7"
…
```

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。

属性	含义
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。

4.22.2.2 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。

示例

```
[frames]
FRAME001= name:=frame1, x:=0, y:=0, width:=460, height:=970,
app:=mydialog
FRAME002= name:=frame2, x:=460, y:=0, width:=1460, height:=1080,
app:=OPERATE
FRAME003= name:=frame3, x:=0, y:=970, width:=460, height:=110,
menu:=mymenu
FRAME004= name:=frame4, x:=0, y:=0, width:=1280, height:=970,
app:=mydialog2, runableApps:="mydialog, mydialog2"
FRAME005= name:=frame5, x:=0, y:=970, width:=1280, height:=110,
menu:=menu2
FRAME006= name:=frame6, x:=1280, y:=0, width:=640, height:=480,
app:=OPERATE
FRAME007= name:=frame7, x:=1280, y:=480, width:=640, height:=600,
app:=mydialog3
…
```

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。以下为当前支持的显示尺寸，单位：像素：

- 640x480
- 760x505
- 800x480

- 800x600
- 1024x768
- 1024x1014
- 1080x810
- 1280x768
- 1280x800
- 1280x1024
- 1300x768
- 1366x768
- 1440x1080
- 1454x1080
- 1854x1014
- 1854x1080
- 1920x1080

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。

4.22.2.3 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]）

段落[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，参见属性 image，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

属性	含义
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。

属性	含义
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

属性	含义
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。

[1)] 只针对 Display Button。

[2)] 针对 menuItem 和 menuItemStyle。

示例 1

```
[menuitems]
MENUITEM001= name:=menuitem1, onClicked:=showDisplay(display1),
text:="Display1"
MENUITEM002= name:=menuitem2, onClicked:=showDisplay(display2),
text:="Display2", accessLevel:=Customer, width:=100
MENUITEM003= name:=miPdf, menuItemStyle:=misMenu,
onClicked:="showApp(defaultFrame, appPdf)", image:=dm_pdf.png,
text:="PDF"
MENUITEM004= name:=miBrowser, menuItemStyle:=misMenu,
onClicked:="showApp(defaultFrame, appBrowser)",
image:=dm_browser.png, textID:=TEXT_ID_BROWSER
```

段落[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。

示例 2

```
[menuitemstyles]
MENUITEMSTYLE001= name:=misMenu, textSize:=10,
textAlignment:=bottom, imageAlignment:=top, color:=37/53/63,
textColor:=153/173/185, pressedColor:=52/87/140,
pressedTextColor:=White, borderColor:=37/53/63,
pressedBorderColor:=52/87/140, borderWidth:=5, height:=66,
width:=66
```

凡是没有指定的属性，系统会采用默认属性，详见上文。

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)

属性	含义
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/ backgroundColor[1)]	菜单的颜色/背景色。 参见“颜色定义”一节。 默认值：Qt::gray

[1)] 针对 menu 和 menuStyle

示例 3

```
[menus]
MENU001= name:=menu1, menuItems:="menuitem1, menuitem2,
menuitem3"
MENU002= name:=menu2, orientation:=vertical, margin:=2,
spacing:=2, items:="menuitem4, menuitem5, spacing(50), menuitem6,
menuitem7, stretch()", itemHeight:=100
MENU003= name:=mMenuOben, menuStyle:=msMenu,
defaultFrame:=fOben, items:="miPdf, miBrowser, spacing(10),
miOperate, stretch(), miMaximize"
MENU004= name:=mGridMenu, menuStyle:=msGrid,
defaultFrame:=fUnten, items:="miMachine(0,0), miProgram(0,1),
miProgramManager(0,2), miDiagnosis(0,3), miPdf(1,0),
miBrowser(1,1), mOperate(1,2,1,2), miMirror(0,3,2,1)"
```

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。

示例 4

```
[menustyles]
MENUSTYLE001= name:=msVerticalMenu, color:=59/76/88,
layout:=column, margin:=0, spacing:=1
MENUSTYLE023= name:=msGrid, color:=59/76/88, layout:=grid,
margin:=10, spacing:=1
```

最后还必须为菜单指定框架，可使用框架的属性 menu：

```
[frames]
FRAME003= name:=frame3, x:=0, y:=970, width:=460, height:=110,
menu:=menu2
```

4.22.2.4 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。

指令	含义
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() 或 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。如果菜单 menuname 已经作为弹出菜单在 framename 框架中显示，则会被隐藏。

指令	含义
setMirrorMode(true|false)	启用（true）或关闭（false）镜像模式 在此处启用镜像模式时，在所有没有用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）！

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。

4.22.2.5 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

- 摄像头显示
- WEB Browser
- SINUMERIK Operate

PDF Viewer

默认设置下，会在文件 systemconfiguration.ini 中进行以下声明：

```
[dialogs]
DLG107= name:=SlPdfApp,
implementation:=sldmpdfviewerapp.SlDmPdfViewerApp,
process:=SlHmiHost1, preload:=false cmdline:="-pdfFile
gestenbedienung.pdf -matteColor #3b4c58"
```

在显示配置中进行各项指定时候应使用程序名称：

```
SlPdfApp
```

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"
```

PDF-Viewer 支持语言切换。

有关如何关联 PDF 文件用于语言识别的**更多信息**，请参见“PDF 文件的语言支持 (页 402)”一章。

关于“PDF Viewer”中提供功能的**更多说明**，请参见“操作手册 - 通用版/车削版/铣削版/磨削版”。

VNC Viewer

默认设置下，会在文件 systemconfiguration.ini 中进行以下声明：

```
[dialogs]
DLG108= name:=SlVncApp,
implementation:=sldmvncviewerapp.SlDmVncViewerApp,
process:=SlHmiHost1, preload:=false, cmdline:="-configuration
MyPC"
```

在显示配置中进行各项指定时候应使用程序名称：
SlVncApp

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

示例

文件 slvncconfig.ini 中的定义：

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

虚拟键盘

默认设置下，会在文件 systemconfiguration.ini 中进行以下声明：

```
[dialogs]
DLG111= name:=SlKeyboardApp,
implementation:=sldmvirtualkeyboardapp.SlDmVirtualKeyboardApp,
process:=SlHmiHost1, preload:=false, cmdline:="-settingsFile
sldm_keyboard.ini"
```

在显示配置中进行各项指定时候应使用程序名称：
SlKeyboardApp

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

[1024x242]
```

```
Num= KeyboardLayout:="numpadlayout", x:=784, y:=25
Alpha= KeyboardLayout:="nclayout", x:=206, y:=25
KeyHeight=47

[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。

虚拟 MCP

在 Display Manager 中可使用两种不同尺寸的 MCP：

- 760x505
- 1024x242 - 在下文中标识为"below"

如果 Display Manager 有四个窗口并且操作界面显示在右上方，则 MCP 会将一个窗口以 [760x505] 的大小显示在左下，将其他窗口以 [1024x242] 的大小显示在操作界面下方。

更多信息还可参见章节 “显示机床控制面板 Sidescreen (页 78)”。

操作步骤

1. 从以下目录中复制样板文件 sldm_1920x1080.ini, sldmmcppage1.ini, sldmmcppagebelow1.ini, systemconfiguration.ini, slsidescreenmcppageconfig.xml, slsidescreenmcppage_deu.ts 以及子目录 ico640 和 ico1920：
   /siemens/sinumerik/hmi/template/displaymanager/mcp/
2. 将 sldm_1920x1080.ini, sldmmcppage1.ini, sldmmcppagebelow1.ini 和 systemconfiguration.ini 的副本保存到以下目录中：
   /**oem**/sinumerik/hmi/cfg 或
   /**user**/sinumerik/hmi/cfg

3. 将 slsidescreenmcppageconfig.xml 的副本保存到目录：
   /**oem**/sinumerik/hmi/proj 或
   /**user**/sinumerik/hmi/proj
4. 将 slsidescreenmcppage_deu.ts 的副本保存到目录：
   /**oem**/sinumerik/hmi/lng 或
   /**user**/sinumerik/hmi/lng

   说明

   TS 文件在启动时自动被系统集成。

5. 将目录 ico640 和 ico1920 保存到以下路径中：
   /**oem**/sinumerik/hmi/ico 或
   /**user**/sinumerik/hmi/ico
6. 机床控制面板可在启动后选择。

也可以选择激活其他机床控制面板窗口（本例中有另外四个）：

1. 请在文件 systemconfiguration.ini 中进行以下调整：
   删除关键字 DLG501, DLG502, DLG503 和 DLG504 之前的符号 ";"。
2. 请在文件 sldm_1920x1080.ini 中进行以下调整：
   删除关键字 MENUITEM500, MENUITEM501, MENUITEM502, MENUITEM503, MENUITEM550, MENU550 和 FRAME500 之前的符号 ";"。
   以下关键字首次出现时请在前面添加一个符号 ";"，以后再出现时请将 ";" 删除：
   MENUITEM124, MENUITEM125, MENU022, MENU032, MENU033, FRAME022 和 FRAME028。
3. 复制样板文件 sldmmcppage2.ini, sldmmcppagebelow2.ini, sldmmcppagebelow3.ini 和 sldmmcppagebelow4.ini 到以下目录：
   /**oem**/sinumerik/hmi/cfg 或
   /**user**/sinumerik/hmi/cfg

显示窗口小部件的应用程序

默认设置下，会在文件 systemconfiguration.ini 中进行以下声明：

```
[dialogs]
DLG109= name:=SlWidgetsApp,
implementation:=sldmsidescreenapp.SlSideScreenDialog,
process:=SlHmiHost1, preload:=false, cmdline:="-sidescreen1
sldmwidgets1.ini -sidescreen2 sldmwidgets2.ini -spacing 3"
```

在显示配置中进行各项指定时候应使用程序名称：

```
SlWidgetsApp
```

在一种显示配置内，该应用程序只允许一个实例中使用。

摄像头显示

在文件 sldmwidgets1.ini 和 sldmwidgets2.ini 中使用了下面这个用于应用设置的关键字：
ELEMENT005

如果通过 SINUMERIK Operate 设置摄像头，则在 Display Manager 中最多可显示两个传输画面。

WEB Browser

文件 sldm_1920x1080.ini 中使用了下面这个用于应用设置的关键字：
MENUITEM129

关于如何设置"WEB 浏览器"的**更多说明**，参见"设置 WEB 浏览器 (页 60)"一章。

SINUEMRIK Operate

在显示配置中进行各项指定时候应使用程序名称：
OPERATE

默认文件 sldm_1920x1080.ini：
```
FRAME040= name:=fMax, x:=66, y:=66, width:=1854, height:=1014,
app:=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。以下为当前支持的显示尺寸（单位：像素）：

- 640x480
- 760x505
- 800x480
- 800x600
- 1024x768
- 1024x1014
- 1080x810

- 1280x768
- 1280x800
- 1280x1024
- 1300x768
- 1366x768
- 1440x1080
- 1454x1080
- 1854x1014
- 1854x1080
- 1920x1080

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。

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

```
[notepad]
nPlacementMode=3
WindowStyle_Off = 15597568
```

B 型应用程序只能在一个实例中执行。

说明

可能的故障

所连接的 OEM 应用有可能不兼容并导致故障。故障时，请检查设置。例如，请通过章节“将 OEM 框架（OEMFrame）应用程序集成到系统中 (页 617)”中介绍设置方式来添加 OEM 应用。

4.22.2.6 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。

条目	含义
startupdisplay	Display Manager 启动后选用的画面的名称。如果此处不作设置，系统将最先设计的画面用作 startupdisplay。
hmihostofmenus	指定执行属于一种画面配置的菜单所在的主机进程。默认为 SlHmiHost1。
backgroundcolor	桌面背景色，在一些框架设计中，当框架和框架之间有间隙时便会露出背景。
textfile	和语言相关的文本，使用方式和 SINUMERIK Operate 上一样。此处可以指定一个或者多个文本文件。 slam 应指定为第 1 个文件。自定义文本文件在逗号后添加。
textcontext	除了在设计菜单或菜单项时指定文本上下文外，还可以在此处进行全局设置，它将统一适用于所有菜单项和语言相关的文本。

示例

```
[miscellaneous]
backgroundcolor=100/120/135
startupdisplay=d3
hmihostofmenus=SlHmiHost1
textfile=slam
textcontext=SlAmAreaMenu
```

说明

名称 slam 应始终保持不变。自定义语言文件在逗号后添加，例如：

```
textfile=slam, <anothertextfile>
```

颜色指定

颜色可以用十进制的 RGB 值来指定，比如：

color:=255/255/255 或 color:="255, 255, 255" 或 color:=255 255 255

说明

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。

另一种颜色指定的方法是使用 HTML 上常用的十六进制值：

```
color:=#FFFFFF
```

许可

5

使用激活的选件需要相应的授权。在购买了许可证后您可以获得“许可证密码”，它包含了所有要求许可证的选件并且只适用于您的 CF 卡。

前提条件

勾选或撤销机床选件，至少要求访问级别为 3（用户）。

只有起码具有该级别的访问权限，您才可以给机床选件授权。

一般说明

CF 卡的序列号	固定的 CF 卡序列号
硬件类型	控制系统的信息
机床名称/编号	显示机床数据 MD17400 $MN_OEM_GLOBAL_INFO 中定义的数据。 只有具有访问级别 1（制造商），显示栏才会变成编辑栏。
许可证密码	需要获得许可证选件的信息

步骤



1. 选择操作区域“调试”。



2. 按下菜单扩展键。



3. 按下软键“许可证”。

   “授权”窗口打开。

   显示控制系统和许可证的一般数据。

许可证数据库

授权通过由西门子工业管理的许可证数据库 Web License Manager（网络许可证管理器）在线进行。

更多信息

关于许可证管理的详细信息请参见

CNC 调试手册：NCK、PLC、驱动。

取消勾选选项

在读入其他 HMI 系统的旧存档时，其中可能包含了当前系统不再需要的选件，但这些选件仍需要获得许可授权：

产品编号	选件
6FC5800-0AP04-0YB0	工步编程
6FC5800-0AP11-0YB0	Manual Machine
6FC5800-0AP15-0YB0	ShopTurn HMI/ShopMill HMI，适用于含 HMI-Advanced 的 SINUMERIK 840Di sl
6FC5800-0AP20-0YB0	成品件 3D 模拟
6FC5800-0AP21-0YB0	铣削模拟（2D 动态；3D 静态）
6FC5800-0AP23-0YB0	ShopMill 同步记录（实时模拟）
6FC5800-0AP24-0YB0	ShopTurn 同步记录（实时模拟）

更多信息：全部选件一览，请见（例如）SIEMENS 工业制系统服务与支持门户网站 (https://support.industry.siemens.com/cs/document/104466659/sinumerik-840d-sl-glossary-functions-and-terms?dti=0&dl=en&lc=de-WW)。

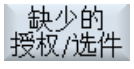
5.1 确定许可证需求

步骤



1. 按下软键“所有选件”，列出控制系统的全部选件。

-或者-



按下软键“缺少的许可证/选件”。

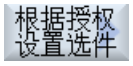
一张选件列表随即显示，其中包含了：

- 没有授权的选件
- 具有授权，但是没有设置的选件

2. 在“已设置”栏中激活或取消激活选件：
   - 勾选复选框或者
   - 输入选件的数量

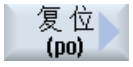
   红色显示的选件已激活，其他的为尚未激活或者许可证数量不足。

   -或者-



按下软键“根据许可证设置选件”，激活许可证密码涵盖的所有选件。

此时会弹出一条安全询问，请按下软键“确认”。



3. 按下软键“复位(po)”，使得刚刚激活的选件生效。出现一条安全询问。

   某些选件还须重启 SINUMERIK Operate 才能生效，在对话框中会弹出重启提示信息。



4. 按下软键“确认”，执行热启动。

   -或者-



按下软键“取消”，中断操作过程。

说明

如果激活了“根据许可证设置选件”功能，则可能会超出可激活轴的总数。因此，请减少相关选件中实际使用的轴数。

查找选件

您可以通过搜索功能找出特定选件。

1. 窗口“许可：所有选件”或“许可：未许可选件”打开。



2. 按下软键“搜索”，在查找框中输入所需选件。

   光标将移动到和关键字吻合的第一个条目上。

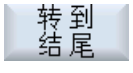


3. 如果找到的选件不符合您的需要，按下软键“继续搜索”。

其他搜索方法



按下软键“转到开头”，从第一个选件开始搜索。



按下软键“转到结尾”，从最后一个选件开始搜索。

说明

使用没有授权的选件

即使没有授权（许可证密码），您也可以暂时激活选件，进行测试。此时控制系统会循环显示报警 8080/8081，指出该选件尚未获得授权。此外在这种状况下，无法进行 NC 启动。

5.2 激活测试许可证

如果希望不购买授权而自由使用选件，可以使用有一定时限的试用授权。一共有 6 个试用授权期。第一个试用授权期持续时间相对较长，根据使用的 NCU 不同而不同。后续五个试用授权期通常每个为 150 个系统工作小时。

说明

试用授权用于方便非生产性质的“短期使用”，例如用于测试和评估。

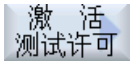
第一个试用授权期

NCU	运行时间
710.3	1000 个小时
720.3	2000 个小时
730.3	3000 个小时

前提条件

激活试用授权需要 2 级访问权限（服务级）。

步骤



1. 按下软键“激活试用授权”。
   出现关于试用授权期剩余数量的提示信息。



2. 按下软键“确认”。
   试用授权期现在激活。
   当前的试用授权期和剩余时间现在都会显示。

说明

当前试用授权期的剩余时间会每小时在控制器的信息行中更新一次。

说明

试用授权中断

试用授权期在以下情况下会被中断：

- 当没有轴真正投入运行时（例如：模拟运行中）。
- 重新输入新的有效许可证密钥时。
- 复位所有未授权的选件时。
  重新设置未授权的选件后，试用授权便开始恢复生效，使用时间继续计算。

说明

试用授权到期

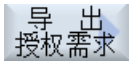
某个试用授权到期后，该授权便失效。

所有试用授权到期后，NCU 上便不再可以激活后续试用授权。

5.3 输出许可证需求

如果您已确定了许可证需求，可以通过设定的外部保存地点输出数据。

步骤



1. 按下“导出许可需求”软键。
   打开一个窗口并提供设定的存储位置。
2. 例如，当您需要将许可证导出到 USB 闪存驱动器上，您可以选择“USB”。



3. 按下“确认”软键。
   已设置的选件、未设置但是获得许可的选件、许可证需求和许可证密码被保存在一个 Alm 文件中。
   此外会建立与 Web License Manager 的互联网连接。

更多信息

关于许可证管理的详细信息请参见

CNC 调试手册：NCK、PLC、驱动。

5.4 读入许可证密钥

有以下方法获取控制系统的许可证密码：

- 手动输入
- 读入电子密码

步骤



“授权”窗口打开。

手动输入许可证密码

1. 如果您通过网络许可证管理器获得许可证密码，在“请在此处输入新的许可证密码”栏中手动输入许可证密码。



2. 按下 <INPUT> 键。

   许可证密码有效时会出现提升信息“已设置许可证密码”。

读入电子许可证密码



如果您获得了具有相应许可证密码的 alm 文件，请按下软键“读入许可证密码”。

…



按下“确认”打开相应的保存地点（例如：USB）。

系统将自动识别匹配的 .alm 文件并读入许可证密码。

提示：

如果目录中有多个合适的 .alm 文件，则会出现一个选择对话框。

选择所需文件。

5.5 新 CF 卡的许可证备份

如果因卡故障而需更换控制系统中的卡并将已有的存档传输至新的空 CF 卡上，必须在恢复前备份新 CF 卡的许可证。不备份就恢复会覆写 CF 卡的许可证密钥。

前提条件

为了能进行许可证密钥备份，PG/PC 上需要一个带 USB 接口的多卡读卡器。

步骤

1. CF 卡插在多卡读卡器中并与 PG/PC 连接。
2. 打开目录 cards/keys/sinumerik/。
   此处找到 txt 文件格式的许可证密钥。
3. 将许可证密钥复制到备份目录中。
4. 将 CF 卡插入控制系统并执行恢复操作。
5. 再次将 CF 卡插入多卡读卡器中。
6. 将许可证密钥重命名为 keys.txt 重新复制到目录 cards/keys/sinumerik 下。
7. 再次将 CF 卡插入控制系统中。
   控制系统重启后，文件 keys.txt 中的许可证密钥安装至系统中并生效。系统会以安装日期重命名 txt 文件。

丢失许可证密钥时的操作

为获取新的空 CF 卡的有效许可证密钥，请联系 SINUMERIK 热线。他们会立即向您发送新的许可证密钥。

创建驱动器连接 6

在 HMI 中最多可设置 21 个与所谓的逻辑驱动器（数据存储器）的连接。在操作区域“程序管理器”和“调试”中可以访问该驱动器。

可配置以下逻辑驱动器：

- USB 接口
- NCU 的 CF 卡，仅限 NCU 中的 SINUMERIK Operate
- 网络驱动器
- PCU 的本地硬盘，仅限 PCU 中的 SINUMERIK Operate



软件选件

如需将 CF 卡用作数据存储器，您需要选件“NCU 附加 HMI CF 卡用户存储器”。

说明

NCU 上的 USB 接口不用于 SINUMERIK Operate，因此无法配置。

6.1 设置驱动器

操作区域“调试”中有一个“设置驱动器”窗口，用于设置程序管理器中的软键。

更多信息

有关操作步骤和配置的更多信息，请参见

操作手册，通用版/车削版/铣削版/磨削版。

文件

生成的设置数据保存在文件“logdrive.ini”中。文件保存在目录 /user/sinumerik/hmi/cfg 下。

一般说明

说明

当系统中有多个本地 USB 前置接口时（多个 TCU、使用 HT8、等），不建议使用 USB 前置接口进行 EES 零件程序执行。通过切换 TCU，可以断开与 U 盘的连接。该设置可以通过复选框“使用该驱动进行 EES 程序执行”实现。

说明

如要通过 PCU 使用全局 USB 前置接口，那么必须通过 VNC 连接将 PCU 设为服务器，参见章节“配置 VNC 连接 (页 270)”。

6.2 设置 EXTCALL

使用指令 EXTCALL 可实现从零件程序中调用外部程序存储器上的程序。

在特殊配置中将两个 SINUMERIK Operate 与一个 NCU 关联时，需要取消 EXTCALL。

打开功能

可以通过以下显示机床数据激活或取消 EXTCALL 调用命令的执行。

MD9106 $MM_SERVE_EXTCALL_PROGRAMS	
= 0	HMI 忽略 EXTCALL 指令并通过 PLC 选择
= 1	HMI 执行 EXTCALL 指令并通过 PLC 选择（缺省值）
= 2	HMI 执行 EXTCALL 指令并忽略通过 PLC 的选择
= 3	HMI 忽略 EXTCALL 指令并执行通过 PLC 的选择

说明

通过外部程序存储器选择主程序的情形下，若上电时该程序存储器继续可用，且 MD9106 中激活了通过 EXTCALL 调用加工，则系统会自动重新选择该程序。

6.3 设置全局零件程序存储器

通过全局零件程序存储器(GDIR)提供一个类似 NC 被动文件系统的结构。它集成在 NC 搜索路径中，用于子程序调用。如设置了全局零件程序存储器(GDIR)，则系统的全部节点都可访问同一个逻辑驱动。各节点可直接从共享的程序存储器中执行零件程序。GRID 替代或扩展了 NC 零件程序存储器。

说明

GDIR 只用于 EES 功能。建立 GDIR 对于 EES 操作而言并不是必须的。

可能的步骤

1. 在“设置驱动器”窗口中设定全局零件程序存储器。
2. 将本地程序存储器中 SPF、MPF 和 WKS 目录下的内容移动到外部驱动器上，如 U 盘。
3. 将备份的目录复制到全局零件程序存储器中。

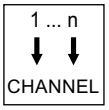
更多信息

有关程序存储器的更多信息，请参见

NC 编程手册。

通道菜单 7

7.1 设计通道菜单



通道菜单的显示通过通道切换键激活。

它用于在 SINUMERIK Operate 上（确切地说是操作站）显示的各个 NC 通道之间来回切换。配置通道菜单后，按下通道切换键后将显示通道菜单。若没有配置任何通道菜单，按下通道切换键后，HMI 将切换到下一个 NC 通道。

配置

操作软件在 “netnames.ini” 文件中进行配置，单个操作站在相应的 “config.ini” 文件中进行配置。

“netnames.ini” 文件只能通过编辑器创建。

存储位置

SINUMERIK Operate 的 “netnames.ini” 文件保存于以下目录中：
/oem/sinumerik/hmi/cfg oder /user/sinumerik/hmi/cfg。

“config.ini”文件的保存目录：

- NCU 上：
  /user/common/tcu/<*TCU-Name*>/common/tcu/config.ini
- PCU 上：
  F:\user_base\common\tcu\<*TCU-Name*>\common/tcu/config.ini

其中 <*TCU-Name*> 是操作站的 TCU 名称，是 TCU 第一次在设备上启动时的命名。

通道菜单的应用场合

- 设备复杂，由多个工作站和多个 NCU 组成（在操作单元管理的范畴内），一个操作站需要操作多个 NCU 组成的 NC 通道。
- 设备简单，由多个 NC 通道组成，不希望采用标准通道切换方法：通道切换键。

通道菜单的基本设计思路是采用一个下拉菜单，向操作人员展示可以用通道菜单切换的所有 NCU 的 NC 通道。为此引入了“通道组”这一概念，以实现通道的逻辑分组。现在，通过下拉菜单便可以直接选中所需的 NC 通道。没有配置通道菜单时，操作人员必须按下通道切换键，浏览每个 NC 通道，直到找到所需的 NC 通道。只有使用通道菜单才能越过 NCU 对操作站进行切换。

7.2 限制

- PCU/ PC 上的 SINUMERIK Operate 只能固定与一个 NCK 连接。 不能切换至其它的 NCK。
- NCU 上的 SINUMERIK Operate 只能和自己的 NCK 一起工作，不能切换至其他的 NCU。

7.3 通道菜单的一般结构

简介

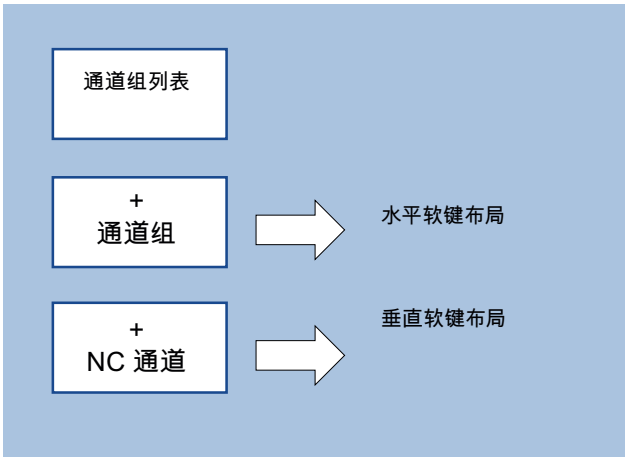
在通道菜单中实现了通道的逻辑分组：即所有 NCK 的所有通道被分为一个个通道组，各个通道组中包含了具体的 NC 通道。

因此首先要为通道菜单定义一个通道组列表。 通道组列表由一个或多个通道组组成。 通道组由一个或多个 NC 通道组成。

在通道菜单中通过水平软键可选择通道组。

所选通道组的通道用垂直软键可选择。

此时，始终只有一个 NCU 的一个 NC 通道可供选择。 在通过通道菜单切换到另一个通道时，有可能已经自动切换到另一个 NCU 上。



说明

比方说，在一台设备中，一个加工站相当于一个通道组，而加工站中的各个加工单元相当于一个个通道。

在一个通道菜单中可以最多配置 32 个通道组，每个通道组最多 8 个通道。

配置整体结构

设备通道菜单的整体结构在 “netnames.ini” 文件中进行配置。 其中可以配置设备上的所有切换目标。 该配置必须统一保存或发送给所有提供通道菜单功能的 SINUMERIK Operate 系统。

而每个操作站本身的通道菜单可以在文件“config.ini”中配置。

两步操作

1. 第一步：在文件 “netnames.ini” 中定义设备上所有操作站使用的所有通道组。
2. 第二步：在操作站自己的 “config.ini” 文件中定义本操作站的通道菜单。

7.4 在 “netnames.ini" 文件中定义通道组

定义通道组

在“netnames.ini” 文件中定义设备上使用的所有通道组， 这一定义是操作站的统一定义。

每个通道组要用单独的语句“[CH_GRP_x] (1 <= x <= 32)”来命名。

要为每个通道组定义以下项目：

- 在通道菜单中显示的通道组垂直软键上的文字。
- 本通道组包含的所有切换目标：比如：各个 NC 通道。 这些切换目标要加以定义，并连续编号，组成一张列表：
  - Membery.Target: 切换目标
  - Membery.Text: 通道菜单中显示的软键文本 (1 <= y <= 8)。

文件 “netnames.ini”

下面以通道组 1 为例来说明如何配置：

```
[CH_GRP_1]                          ;  通道组名称；命名必须符合该格式。 最后一位是通道组
                                    ;  的编号，要连续编号。
                                    ;
                                    ;
Text = Bearbeitung1                 ;  在通道菜单中显示的通道组软键文字（水平软键），通道
                                    ;  组的切换目标 1
                                    ;
                                    ;
Member1.Target =                    ;  运行 HMI 的计算机的名称（主机名称）或 IP 地址。
[<HMI>.]<NC>.#<通道号>              ;  <HMI> = <HMI 主机(PC/PCU)的名称或 IP 地址>
                                    ;  如果 HMI 在 NCU 上本地运行，也可以不设置<HMI>
                                    ;  <NC> = <NC 的名称或地址>
                                    ;  此处也可以输入 mmc.ini 中的机床名称。
                                    ;  可使用“ #通道编号”指定参与切换的 NC 通道。
                                    ;
                                    ;
                                    ;
                                    ;
                                    ;
                                    ;
                                    ;
Member<NR>.Text = NCU730_K1         ;  在通道菜单中显示的软键文字（垂直软键）。 若在
                                    ;  “Target”指定了通道号，此处可以不使用固定文字
                                    ;  “$MC_CHAN_NAME”，而是确定需要显示的软键文字是否
                                    ;  是 NCK 机床数据中确定的通道名称。
                                    ;
                                    ;
                                    ;
                                    ;
                                    ;
Member2.Target =
Member2.Text                        ;  垂直软键 2 上的文字
```

7.5 在 “config.ini" 文件中设计操作站

在操作站自己的“config.ini” 文件中为本操作站定义通道菜单的属性。此处会使用到文件 “netnames.ini”中的输入。其中，通道组由通道菜单中的一个软键调用。

文件 “config.ini”

- 段落[T2M2N]用于配置通道菜单。
- 软键参考标记“CH_GRP_xx”指的是“netnames.ini”中定义的通道组。标记是绑定的。
- 其中，SKx 指的是通道菜单中第 x 个水平软键。

示例

```
...
[T2M2N]
SK1=CH_GRP_1
SK2=CH_GRP_3
SK8=CH_GRP_15
```

7.6 示例： 1:N 时的配置

配置 1:N

1:N 配置要买满足以下前提条件：

- 一个安装了 SINUMERIK Operate 的 PCU 最多和 4 个 NCU 相连。
- 所有 NCU 上的 SINUMERIK Operate 已经关闭。
- 需要修改以下配置文件：
  siemens/sinumerik/hmi/template/cfg/**netnames.ini**
  siemens/sinumerik/hmi/template/cfg/**mmc.ini**

下文以 1 个 PCU 和 2 个 NCU 为例说明如何进行配置。

mmc.ini 示例

```
[Global]
NcddeMachineNames=net, NCU840D_1, NCU840D_2
NcddeDefaultMachineName=NCU840D_1

[NCU840D_1]
ADDRESS0=192.168.214.1,LINE=14,NAME=/NC,SAP=040d,PROFILE=CLT1__CP_L4_INT
ADDRESS1=192.168.214.1,LINE=14,NAME=/PLC,SAP=0202,PROFILE=CLT1__CP_L4_INT
```

```
ADDRESS2=192.168.214.1,LINE=14,NAME=/CP,SAP=0502,PROFILE=CLT1__CP_L4_INT
ADDRESS10=192.168.214.1,LINE=14,NAME=/DRIVE_00_000,SAP=0201,
SUBNET=0046-00000000:000,PROFILE=CLT1__CP_L4_INT
ADDRESS11=192.168.214.1,LINE=14,NAME=/DRIVE_03_003,SAP=0900,
PROFILE=CLT1__CP_L4_INT

[NCU840D_2]
ADDRESS0=192.168.214.2,LINE=14,NAME=/NC,SAP=040d,
PROFILE=CLT1__CP_L4_INT
ADDRESS1=192.168.214.2,LINE=14,NAME=/PLC,SAP=0202,
PROFILE=CLT1__CP_L4_INT
ADDRESS2=192.168.214.2,LINE=14,NAME=/CP,SAP=0502,
PROFILE=CLT1__CP_L4_INT
ADDRESS10=192.168.214.2,LINE=14,NAME=/DRIVE_00_000,SAP=0201,
SUBNET=0000-00000000:000,PROFILE=CLT1__CP_L4_INT
ADDRESS11=192.168.214.2,LINE=14,NAME=/DRIVE_03_003,SAP=0900,
PROFILE=CLT1__CP_L4_INT
```

netnames.ini 示例

```
;*****************************************************************
**
;     netnames.ini      - EXAMPLE -
;
; This is an example of a netnames.ini for 1:N NCU-switching in
; a configuration of 3 NCU and a PCU50.
;*****************************************************************
***
[own]
owner = HMI

;***************************
; ChanMenu
;***************************
[chan HMI]
ShowChanMenu = true

;*****************************************************************
***
;Channel-groups
;Correlation from channel-group to horizontal softkeys
; -> see config.ini of the TCU
;The section [T2M2N] can also be created in the netnames.ini file
;instead of config.ini file. This configuration applies to all
TCU,
;which are not configured by their config.ini.
;*****************************************************************
***
[T2M2N]
```

```
SK1 = CH_GRP_1
SK2 = CH_GRP_2

[CH_GRP_1]

Text = 1. NCU
Member1.Target = 192.168.214.241.192.168.214.1.#1
Member1.Text = NCU1%nKanal 1

Member2.Target = 192.168.214.241.192.168.214.1.#2
Member2.Text = NCU1%nKanal 2

[CH_GRP_2]
Text = 2. NCU
Member1.Target = 192.168.214.241.192.168.214.2.#1
Member1.Text = NCU2%nKanal 1

Member2.Target = 192.168.214.241.192.168.214.2.#2
Member2.Text = NCU2%nKanal 2
```

说明

切换带操作软件的 MCP 时，不允许使用 PLC 功能块 FB9（即 FB9 不允许被激活）。必要时必须在 PLC 用户程序内实现 MCP 切换。

解决办法

在 OB100（调用 FB1）中将 MCP1BusAdr 设为 255，MCP1Stop 设为 true。

在 OB1 中查看信号“MMCBTSSready”(DB10.DBX108.3)。如果该信号置位，将 DB19.DBB123 (MCP 索引) 中的 MCP 地址写入到 DB7.DBB27 中并随后撤销 MCP 停止位 DB7.DBX62.1。 MCP 现在激活。

如果现在再次撤销了信号“MMCBTSSready”(DB10.DBX108.3) ，即操作软件切换到另一个 NCU 上，则必须重新置位 MCP 停止位 DB7.DBX62.1。

7.7 1:N 配置中工作表的划分

简单的程序选择方式

该功能只用于 “一个 PCU 对多个 NCU”的配置，必须在配置文件“systemconfiguration.ini”中激活：

```
[miscellaneous]

distributeJoblist=true
```

原始程序位于 PCU 或 PC 的本地驱动器上。

程序的传送是通过选择“工件”目录中的工作表实现的。 工作表可以包含 COPY 和 SELECT 指令。

```
;复制 A.MPF 到 NCU1 上：

COPY A.MPF NC=NCU1

;复制 B.MPF 到 NCU2 上：

COPY B.MPF NC=NCU2

;复制所有子程序到所有 NCU 上：

COPY *.SPF *

;选择通道 1 中 NCU1 上的 A.MPF。

SELECT A.MPF ch=NCU1,1

;选择通道 2 中 NCU2 上的 B.MPF。

SELECT B.MPF ch=NCU2,2
```

工作流程：

1. 如果在选择程序时已经有一个正在活动的工作表，则传送到 NCU 上的所有程序无需确认直接被删除。 如果 NCU 上的“工件”目录随后为空，该目录也被删除。
2. 程序被复制到指定 NCU 的“工件”目录中。
3. 该目录中的已有程序不经提示直接被覆写。
4. 通过编辑器修改本地驱动器上的原始程序后，新的原始程序被复制到指定 NCU 上。

结构示例

/user/sinumerik/data/prog/JOBA.WPD	NCU1	NCU2
JOBA.JOB	WKS.DIR	WKS.DIR
A.MPF	WKS.DIR/JOBA.WPD	WKS.DIR/JOBA.WPD
B.MPF	WKS.DIR/JOBA.WPD/B.MPF	WKS.DIR/JOBA.WPD/B.MPF
UP1.SPF	JOBA.WPD/UP1.SPF	JOBA.WPD/UP1.SPF
UP2.SPF	WKS.DIR/JOBA.WPD/UP2.SPF	WKS.DIR/JOBA.WPD/UP2.SPF

/user/sinumerik/data/prog/wks.dir/JOBA.WPD	NCU1	NCU2
JOBA.JOB	WKS.DIR	WKS.DIR
A.MPF	WKS.DIR/JOBA.WPD	WKS.DIR/JOBA.WPD
B.MPF	WKS.DIR/JOBA.WPD/B.MPF	WKS.DIR/JOBA.WPD/B.MPF
UP1.SPF	JOBA.WPD/UP1.SPF	JOBA.WPD/UP1.SPF
UP2.SPF	WKS.DIR/JOBA.WPD/UP2.SPF	WKS.DIR/JOBA.WPD/UP2.SPF

前提条件

特殊性：

- 在操作软件重启后，传送给 NCU 的程序得以保留，不会丢失。
- 既不能删除、重命名本地驱动器上的原始程序，也不能通过复制/移动覆写原始程序。
- 对传送给 NCU 的程序所作的修改如果不手动保存会丢失。
- “加工”区内的“程序修改”功能失效。
- 在建立调试存档时，本地驱动器的所有数据一同自动备份。

限制：

- 通过编辑器修改本地驱动器上的原始程序时要注意，只有当前相连 NCU 的通道状态才受到监控。 如果该程序在另一个 NCU 上处于暂停状态或活动状态，修改后可能无法正确传送。
- 当前程序段的显示、程序段预处理的标记和编辑器中当前打开程序的只读处理仅限当前相连 NCU 上当前活动的程序。
- 在编辑器中，打开的始终是当前相连 NCU 上的程序。 对于传送的程序来说，这可能会导致错误的进度显示。
- 在操作区“加工”的当前程序段显示画面中，只显示当前相连 NCU 上当前活动的/传送的程序。
- 跨越多台机床的加工模拟无法实现。
- 如果 systemconfiguration.ini 文件中已经配置了 `distributeJoblist=true`，则无法通过操作区“程序管理器”选择零件程序或子程序。

机床数据和设定数据

8

8.1 机床数据和设定数据

一览

前缀	编号范围	机床数据和设定数据	保存区域（软键）
$MM	9000 - 9999	显示机床数据	显示机床数据
$MNS	51000 - 51299	通用配置机床数据	通用机床数据
	51300 - 51999	通用循环机床数据	
$MCS	52000 - 52299	通道专用配置机床数据	通道机床数据
	52300 - 52999	通用专用循环机床数据	
$MAS	53000 - 53299	轴专用配置机床数据	轴机床数据
	53300 - 53999	轴专用循环机床数据	
$SNS	54000 - 54299	通用配置设定数据	通用设定数据
	54300 - 54999	通用循环设定数据	
$SCS	55000 - 55299	通道专用配置设定数据	通道设定数据
	55300 - 55999	通道专用循环设定数据	
$SAS	56000 - 56299	轴专用配置设定数据	轴设定数据
	56300 - 56999	轴专用循环设定数据	

更多信息

关于机床数据和设定数据的详细信息，请参见参数手册的“机床数据和参数”一章。

8.2 显示/更改机床数据

可在操作区域“调试”中读取如下机床数据。

- 通用机床数据 ($MN) 和通用配置机床数据 ($MNS)
- 通道专用机床数据 ($MC) 和通道专用配置机床数据 ($MCS)
- 轴专用机床数据 ($MA) 和轴专用配置机床数据 ($MAS)
- 驱动专用机床数据($M_)
- 控制单元机床数据: 驱动参数
- 供电机床数据: 驱动参数
- I/O 组件参数： 驱动参数
- 通讯参数 （CULINK, DMC）： 驱动参数

说明

机床数据或驱动参数的显示取决于控制系统的配置。
机床数据区的访问权限可以通过钥匙开关位置或者口令锁定。
自访问级别 4（钥匙开关位置 3）起可以读取数据。
自访问级别 1（口令：制造商）起可以修改机床数据。

⚠ 警告

错误的参数设置

机床数据更改对机床会有重大影响。 错误的参数设置可能危及人员生命，并导致机床损坏。

关于机床数据的信息

以下信息由左向右依次显示：

- 机床数据编号，可能带有数组索引
- 机床数据名称
- 机床数据值

- 机床数据单位
- 有效性

说明

对于没有单位的机床数据，其相应栏为空。 如果数据不可用，则用“#”代替数值进行显示。
如果值以“H”结尾，则为十六进制值。

机床数据物理单位显示在输入栏旁的右侧。

从每个机床数据右侧的一列中读取其激活方式。

so	立即： 无要求动作
cf	配置： 软键“机床数据有效(cf)”
re	复位： 机床控制面板上的 <Reset> 键
po	POWER ON: 软键“复位(po)”

附加信息

在页脚中显示所选机床数据或设定数据的简要说明。

步骤



1. 选择操作区域“调试”。



2. 按下软键“机床数据”。 会显示其他用于单个机床数据区的软键。

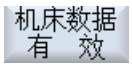






3. 按下软键“通用机床数据”、“通道机床数据”或“轴机床数据”。
   打开一个窗口并显示所选择的机床数据。

4. 将光标置于所需的机床数据上。
5. 将光标定位在需要更改的条目上，并输入需要的数值。
6. 根据激活方式的不同，有两种方式可以使设置生效：
   按下软键“机床数据有效(cf)”。 该值被接收。



-或者-



按下软键“复位(po)”。

出现一条安全询问。



7. 按下软键“确认”，执行热启动。

-或者-



如果不希望接收设置，可以按下软键“取消”。

轴选择

如果所选区域存在多个组件（例如“轴机床数据”），则有以下选项：





1. 按下软键“轴 +”或“轴 -”。

显示下一个（+）或上一个（-）轴的值。

-或者-



按下软键“直接选择”。

“轴直接选择”窗口 打开。

2. 从下拉菜单中直接选出所需的轴。



3. 按下软键“确认”，保存设置。

-或者-



按下软键“取消”，放弃设置。

8.3 显示/修改显示机床数据

您可以在操作界面的操作区域“调试”内访问“显示机床数据”。

步骤



1. 选择操作区域“调试”。



2. 按下软键“机床数据”。



3. 按下菜单扩展键。
   显示其他软键。



4. 按下软键“显示机床数据”。
   显示出显示机床数据。
5. 将光标定位在需要更改的条目上，并输入需要的数值。



6. 按下软键“复位(po)”。
   出现一条安全询问。



7. 按下软键“确认”，执行热启动。

   -或者-



   如果不希望接收设置，可以按下软键“取消”。

8.4 显示/更改设定数据

您可以在操作界面的操作区域“调试”内访问以下设定数据。

- 通用设定数据
- 通道专用设定数据
- 轴专用设定数据

步骤



1. 选择操作区域“调试”。



2. 按下软键“机床数据”。



3. 按下菜单扩展键。
   会显示出软键“通用设定数据”、“通道设定数据”和“轴设定数据”。



4. 按下相应的软键可以显示所需的设定数据区。
   打开一个窗口并显示所选择的设定数据。

5. 将光标移至需要的设定数据上，然后定位到待修改的条目上。 输入所需的值。
6. 根据激活方式的不同，有两种方式可以使设置生效：



按下软键“机床数据有效(cf)”。

-或者-



按下软键“复位(po)”。

出现一条安全询问。



7. 按下软键“确认”，执行热启动。

-或者-



如果不希望接收设置，可以按下软键“取消”。

轴选择/通道选择

如果所选区域存在多个组件（例如“轴设定数据”或者“通道设定数据”），则有以下选择方式：









1. 按下软键“轴 +”或“轴 -”或“通道 +”或“通道 -”。

显示下一个（+）或上一个（-）轴/通道的值。

-或者-



按下软键“直接选择”。

窗口“直接选择”打开。

2. 通过选择列表直接从提供的轴中选择所需的轴。



3. 按下软键“确认”，保存设置。

-或者-



按下软键“取消”，放弃设置。

8.5 显示/修改驱动参数

对于驱动参数的显示和编辑可使用下列软键：“控制单元机床数据”、“供电机床数据”和“驱动机床数据”以及菜单扩展条上的软键“I/O 组件机床数据”和“通讯机床数据”。

每个区域都有单独的列表画面，在该画面中可以显示和修改驱动参数。

前提条件

驱动参数的设置受访问级别 2（服务）的保护。
拥有访问级别 4（钥匙开关 3）的人员可以查看驱动参数列表。

更多信息

关于驱动调试的详细信息，请参见

CNC 调试手册：NCK、PLC、驱动。

步骤



1. 选择操作区域“调试”。



2. 按下软键“机床数据”。
   出现软键“控制单元参数”、“供电机床数据”和“驱动机床数据”。
3. 按下所需范围的软键进而显示相关的参数。
4. 将光标定位到所需的参数上。
5. 将光标定位在需要更改的条目上，并输入需要的数值。



6. 可执行保存或复位操作使设置生效。
   按下软键“保存/复位”。



7. 按下软键“保存”。
   会询问“您需要保存什么？”
   可通过软键进行如下选择：
   - 当前驱动对象
   - 驱动设备
   - 驱动系统

   -或者-



按下软键“复位(po)”。

会询问是否为 NCK 和整个驱动系统执行热启动。



按下软键“是”确认询问。



按下软键“否”拒绝热启动。

扩展复位

若仅需要对特定对象或者特定区域执行热启动，可使用其它的软键。



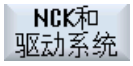
按下软键“扩展复位”。

会询问热启动执行对象。

可通过如下软键进行选择：



只重新启动 NCK。



重新启动系统中所有的控制单元和 NCK。若通过“是”进行了确认，则此处的软键与软键“复位(po)”的功能相同。



重新启动系统中所有的控制单元、NCK 和操作软件。

隐藏扩展软键栏

在文件“slsuconfig.ini”中定义软键“扩展复位”。若未显示软键，则执行如下操作：

1. 从目录 /**siemens**/sinumerik/hmi/template/cfg/ 中复制文件“slsuconfig.ini”。
2. 并将其粘贴至如下目录中：
   /**user**/sinumerik/hmi/cfg/。
3. 打开此文件，并在段
   [Softkeys]
   中输入 AdvancedReset = **0**。

8.6 机床数据显示选项

使用机床数据显示筛选可以根据需要缩小每个区域机床数据的显示数量，例如通用机床数据。

机床数据区域

显示过滤器可用于以下机床数据区域：

- 通用机床数据
- 通道专用机床数据
- 轴专用机床数据
- 驱动机床数据
- 显示机床数据

筛选选项

有以下筛选选项，限制机床数据的显示：

- 仅显示使用的轴
- 按照索引过滤
- 按照显示组过滤
- 显示专家参数

仅显示使用的轴

如果勾选该复选框，则在翻页时会隐藏所有未使用的轴。

索引从...到

索引筛选针对的是机床数据字段。在窗口中，这些机床数据由附加在机床数据字符串前面的字段索引加以区分。

示例：10000[索引]AXCONF_MACHAX_NAME_TAB

如激活了索引过滤器，只显示指定索引范围中的机床数据字段。

显示分组

一个显示分组包含了一个机床数据区域中同属于一类的机床数据。

说明

一个机床数据属于哪个显示分组，请参见各个机床数据描述中的参数“显示过滤器”。

更多信息：参数手册的“机床数据和参数”一章

通过选择或取消选择显示分组来增加或减少当前机床数据区域中显示的机床数据的数量。

显示专家参数

如不勾选“显示专家参数”，则只显示在调试 NC 基本功能时必需的机床数据。

步骤



1. 选择操作区域“调试”。



2. 按下软键“机床数据”。



3. 按下所需数据区的软键，例如“通用机床数据”。
显示出通用机床数据的列表。



4. 按下软键“>>”和“显示选项”。
所选机床数据的“显示选项...”窗口打开。

5. 勾选复选框，来选择需要的筛选选项（按显示分组、按索引、专家参数）。



6. 如要激活所有的显示分组，按下软键“全部选择”。

-或者-



按下软键“全部删除”。
所有的小勾被清除，可以有目的的进行显示分组的选择。



按下软键“确认”，保存设置。

8.7 编辑机床数据和驱动参数

8.7.1 编辑十六进制的数值

对于十六进制的机床数据，您可通过“位编辑器”来输入需要的值。

步骤



1. 选择操作区域“调试”。





2. 按下软键“机床数据”，选择需要的机床数据区域（例如通用机床数据）。
3. 请将光标定位在要更改的输入上。





4. 按下<SELECT>或<INSERT>键。
   窗口“位编辑器”打开。
   显示机床数据的所有位及描述文本。
5. 如勾选了复选框“显示所有位”，将根据数据类型（8 位、16 位、或 32 位）显示所有位。 勾选所需的位。



6. 按下软键“确认”。 系统会再次检查设置，必要时输出故障信息。
   关闭窗口并返回到机床数据概览。
   新输入的值将显示为十六进制。

8.7.2 编辑 BICO 数值

对于驱动参数，您可以通过“BICO 编辑器”输入 BICO 数值。

说明

也可通过复制、粘贴或者直接在输入栏中输入 BICO 数值。

步骤



1. 选择操作区域“调试”。



2. 按下软键“机床数据”。



3. 按下软键“控制单元机床数据”，并从中选择一个 BICO 参数，如 "p738"。
4. 将光标定位至需更改的条目，比如 BICO 63.2091.0





5. 按下<SELECT>或<INSERT>键。
   窗口“BICO 编辑器”打开。
6. 输入“驱动对象编号”，“参数编号”“位/索引”和“参数值（十六进制）”的数据。

说明

当待修改的值不输入为 0 或 1 时，BICO 编辑器也会打开。

8.7.3 编辑 Enum 值

对于驱动参数，可通过选择列表编辑 Enum 项（比如：r2、 p3、 p4 和 p97 等）。

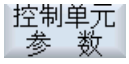
步骤



1. 选择操作区域“调试”。



2. 按下软键“机床数据”。



3. 选择所需的驱动参数区域，比如“控制单元机床数据”。
4. 选择所需的参数，比如“p3”并将光标定位至待更改的条目，比如“[1] 显示, Si...”。





5. 按下<SELECT>或<INSERT>键。
   打开窗口，出现含有全部已定义 Enum 值的列表，及其相关描述。
6. 可从列表中选择一个数值。
   - 或者
   可直接在括号 [ ] 中输入十进制数值。

8.8 查找数据

您可以根据需要搜索机床数据、设定数据和显示机床数据。

搜索方案

- 输入数字或带索引的数字时，能精确搜索到该参数。
  示例：输入“9” > 找到 p9（而不是 p99）
- 输入文本时将执行全文本搜索，即不仅在相应的表格中进行搜索，也在页脚的解释性文本中搜索该关键字。

说明

使用占位符进行搜索

在搜索机床数据时，您可以使用占位符：

- "*": 代表任意字符串
- "?": 代表任意字符

搜索的字段

如果未设置选项“同时在解释性文本中搜索”，则会根据输入的字符串搜索以下机床数据字段：

- 机床数据编号
- 符号名称
- 单位

如果设置了选项“同时在解释性文本中搜索”，则还会搜索以下字段：

- 说明文本
- 位文本
- Enum 文本（仅针对驱动）

步骤



1. 选择操作区域“调试”。



2. 按下软键“机床数据”。
3. 按下相应软键来选取所需的范围。

搜索

4. 按下软键“搜索...”并在搜索框输入所需文本或数字。

5. 如果只想在显示的表格中进行搜索，请取消勾选“同时在解释性文本中搜索”。

SELECT

6. 在“方向”栏中选择所需的搜索方向（向前、向后）。

光标将移动到和关键字吻合的第一个条目上。

继续搜索

7. 如果找到的机床数据、设定数据或显示机床数据不是搜索项，按下软键“继续搜索”。

其他搜索方法

转到开始 按下软键“转至开头”，从第一个数据开始搜索。

转到结尾 按下软键“转至结尾”，从最后一个数据开始搜索。

8.9 管理数据

一览

“管理数据”功能用于简化调试，并可以传输、保存、加载和比较机床数据文件、设定数据文件、补偿数据文件和驱动数据文件。文件保存为 ASCII 格式 .tea。

软键“管理数据”从访问级别“服务”(= 2)开始可见。

以下选项栏在“传输机床数据”窗口中可用：

- 在控制系统内传输数据
  这里只列出当前控制系统所有的控制对象。
- 将数据保存至文件
  对于 SINAMICS 参数还另外创建了后缀名为 .acx 和 .log 的文件：
  – 后缀名为 .acx 的文件包含了所选驱动对象的参数描述。
  – Log 文件包含将 .acx 文件转换为 .tea 文件时出现的警告和错误。
- 从文件加载数据

说明

驱动对象加载后只有 NCK 和驱动会自动复位，其他所有控制对象都必须手动复位。

- 比较数据
  可同时对多个文件 (.tea) 和/或控制对象（轴、通道、伺服系统、供电系统等）进行比较。
  但是无法对经常变化的数据进行在线比较，因为比较时的值只是暂存的值。
  比较数据列表最多可包含 8 个元素。
- 导出/导入 SINAMICS 列表
  比如在系统中使用了液压组件时，则可导出或导入一个阀门列表（液压驱动）。

更多信息

关于阀门列表应用的详细信息，请参见

SINAMICS S120 液压驱动系统手册

步骤



1. 选择操作区域“调试”。



2. 按下软键“机床数据”。
   在任何机床数据区都有软键“管理数据”，除了“用户视图”区。
   -或者-
   按下软键“系统数据”并选择目录 HMI 数据/数据备份/IB 数据。





3. 按下软键“管理数据”。
   窗口“传输机床数据”打开。



4. 激活所需的选项栏并按下软键“确认”。

在控制系统内传输数据



5. 将光标移至控制对象并点击“确认”。



6. 从打开的窗口选择目标控制对象并点击“确认”。

将数据保存至文件



5. 将光标移至控制对象并点击“确认”。

6. 窗口“另存为：选择存储位置”打开，系统已预选择匹配的存储位置。



按下软键“新建目录”，创建新目录。在窗口“新建目录”输入所需名称并按下软键“确认”。



7. 在窗口“新建文件”输入需备份文件的名称并按下软键“确认”。

保存的文件在目录 HMI 数据/数据备份/IB 数据的“系统数据”树形图中。

从文件加载数据

5. 窗口“加载数据：选择文件”打开。



将光标移至 IB 数据目录下或 USB 闪存驱动器内的 .tea 文件，并按下软键“确认”。

注意：在加载驱动数据时，如果数据源和目标驱动对象编号不同，则必须调整单个参数。按下软键“调整”并在随后弹出的窗口内编辑参数值。

比较数据



5. 在窗口“比较数据 - 控制系统”中，从控制对象的树形结构选择要比较的元素。

-或者-



按下软键“文件”，从文件的树形结构选择要比较的 .tea 文件。



6. 按下软键“导入到清单中”。

在窗口的下部将显示要比较的数据列表，并添加选定的元素。



7. 按下软键“从清单中删除”，将单个元素从比较列表中删除。

-或者-

从待比较数据列表中删除所选元素复选框的选勾。



8. 只要至少列出 2 个元素，就可以使用软键“比较”。按下该软键进行比较。

8.10 用户视图

功能的用途

在“用户视图”窗口中，您可以将某个功能的所有相关机床数据综合在一起，并加以调整，这些自定义的机床数据组合可以大大简化您的工作。

您还可以在创建或编辑用户视图时，添加注释。

说明

导入用户视图（兼容性）

如果已经在高级版 HMI 中创建了用户视图，可以将文件 *.klb 复制到 /**user**/sinumerik/hmi/template/user_views 中，以便在 SINUMERIK Operate 中继续使用。

在“参数”操作区域中显示用户视图

按照以下步骤，同时在“参数”操作区域中显示用户视图：

1. 选择操作区域“调试”。
2. 按下软键“机床数据”。
3. 按下软键“用户视图”。
4. 按下软键“管理视图”。
5. 按下软键“新视图”。
6. 若需要在“参数”→“设定数据”→“数据列表”中一并显示用户视图，则应勾选复选框“在参数/设定数据下也进行显示”。
7. 命名用户视图。
8. 按下“确认”。

8.10.1 创建用户视图

步骤



1. 选择操作区域“调试”，按下软键“机床数据”。



2. 按下软键“用户视图”，
   “管理视图”和
   “新视图”。
   窗口“新视图”打开。

3. 命名用户视图。



按下软键“确认”。

若与已有的用户视图同名，会出现安全消息询问是否覆盖已有的视图。



4. 按下软键“插入数据”。

“插入数据”选择窗口打开。



5. 按下<INSERT>键打开下拉菜单，选择所需的机床数据区并按下<INPUT>键。







通过光标键或软键“搜索”，选择需要的机床数据。







6. 按下软键“行前插入”或“行后插入”，将所选择的机床数据加入用户视图中所需的位置上。

提示：

在打开的用户视图中，您可以用鼠标浏览，选择所需的数据行，无需关闭选择窗口。



7. 按下软键“返回”关闭选择窗口，返回到待编辑的用户视图中。

输入值会被自动储存。



8. 按下软键“插入文本”，可以向打开的用户视图中插入任意文本。

输入窗口“插入文本”打开。





9. 输入所需的文本以及与插入文本相关的注释说明，按下软键“行前插入”或“行后插入”，插入文本。



10. 按下软键“返回”，保存输入内容并返回到待编辑的用户视图中。

选择特定的机床数据



借助搜索对话框可以明确查找到目标数据。

8.10.2 加工用户视图

按下软键“属性”，您可以查看并修改注释、机床数据的来源（通道、轴、驱动设备）以及数据说明，具体情况取决于所选的数据行。

- 机床数据
  可以为机床数据选择固定或可变的数据源，视数据类型而定。
  选项“可变（*）”表示使用垂直软键 1 - 3 （+、- 和 直接选择）可以切换相应的数据源（通道、轴、驱动设备），并且显示值会随即变为刚刚选取的数据源。
  也可以选择固定的数据源，这样在切换到其他数据源时显示值不会发生变化。
- 注释文本
  文本和说明

步骤



1. 选择操作区域“调试”，按下软键“机床数据”。





2. 按下软键“用户视图”。
   “用户视图”窗口打开。



3. 按下软键“编辑视图”。 编辑用户视图时，还可使用其它的软键。



4. 如果需要添加其他数据，按下软键“插入数据”。





5. 借助光标键选择所需的机床数据。





6. 按下软键“向下”或“向上”来移动所选中的行。

-或者-



按下软键“删除行”，从视图中删除所选中的行。 此处不会弹出安全询问，而是直接删除数据。

-或者-



按下软键“属性”，以读取或更改注释文本。

“属性”窗口打开。 取决于设置，机床数据的说明或机床数据可能显示在窗口左下方。

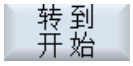


7. 按下软键“返回”保存修改。

选择特定的机床数据



借助搜索对话框可以明确查找到目标数据。



按下软键“转到开始”，从第一个数据开始搜索。



按下软键“转到结尾”，从最后一个数据开始搜索。

如果找到的数据不符合您的需要，按下软键“继续搜索”。

8.10.3 删除用户视图

步骤



1. 选择操作区域“调试”。



2. 按下软键“机床数据”。



3. 按下软键“用户视图”。



4. 按下软键“管理视图”。

5. 在选择列表中选择所需视图。



6. 按下软键“删除视图”。



随即弹出一条安全询问，要求按下“是”确认，或按下“否”取消操作。

或者



之后会弹出对话框“用户视图”。

还有一种方法是，在以下路径的“系统数据”下修改用户视图：/user/sinumerik/hmi/template/user_views。

说明

属于系统的用户视图无法删除。

8.11 机床数据和设定数据的明码文本

对于机床数据和设定数据，可以按照特征[索引]来配置用户专用的、与语言相关的明码文本。

明码文本的显示

可通过以下显示机床数据设置明码文本的显示：

MD9900 $MM_MD_TEXT_SWITCH	替代机床数据名称的明码文本
= 1	明码文本在表格中显示
= 0	明码文本在对话栏中显示

文件

文件	机床数据/设定数据
nctea_xxx.txt	通用机床数据/设定数据
chtea_xxx.txt	通道专用机床数据/设定数据
axtea_xxx.txt	轴专用机床数据/设定数据
hmitea_xxx.txt	显示机床数据

对于 “xxx” 必须使用规定的语言标识，例如：德语为 “deu”，英语为 “eng” 等。
所支持的语言请见 “支持的语言 (页 614)”。

请确保写入文件名时使用的是小写字母。

步骤

1. 根据需要修改哪种机床数据或设定数据的明码文本，创建相应的文件，例如“nctea_xxx.txt”。
2. 采用 UTF-8 编码保存文件，从而使变音字符和特殊字母能在 SINUMERIK Operate 中正确显示。
3. 在 SINUMERIK Operate 中将文件保存在以下目录：
   System CF-Card/**user**/sinumerik/hmi/lng 或
   System CF-Card/**oem**/sinumerik/hmi/lng 或
   System CF-Card/**addon**/sinumerik/hmi/lng

说明

搜索文件时，会依次搜索“user”、“oem”和“addon”目录。然后对最先搜索到的文件进行分析。例如文本文件位于“user”和“oem”目录中时，会对“user”目录中的文件进行分析。

说明

如果一个机床数据有不同的索引，而明码文本中未指定索引，则对机床数据的所有索引显示相同的明码文本。

参数和索引必须为升序排列。

使用德语文本的文件示例

```
14510 Mein Integer-Text für PLC_MD 14510

14510[1] Mein Integer-Text für MD 14510[1]

14512[1] Mein Text für HEX-Datum 14512[1]
```

显示的结果：

在操作区域“调试”、“通用 MD”中，明码文本在表格中或在对话栏中显示。

机床数据	文本
14510	Mein Integer-Text für PLC_MD 14510
14510[1]	Mein Integer-Text für MD 14510[1]
14512[1]	Mein Text in HEX-Datum 14512[1]

匹配操作区域“加工” 9

自动切换

通过“MD51040 $MNS_SWITCH_TO_MACHINE_MASK”可以设置，在哪种条件下系统可以自动切换到操作区域“加工”。

MD51040 $MNS_SWITCH_TO_MACHINE_MASK	**0（缺省设置）**
位 0 = 1	在“程序管理器”操作区中选择程序时不会自动切换到“加工”操作区。
位 1 = 1	在“程序管理器”操作区中选择程序时不会自动切换到“加工”操作区。 在通过机床操作面板切换操作方式时，不会自动切换到“加工”操作区。
位 2 = 1	在“程序”操作区中选择程序时不会自动切换到“加工”操作区。
位 3 = 1	在“程序”操作区中选择程序或执行程序时，不会自动启动程序段搜索。

9.1 设置实际值显示的字体大小

设置字体大小

在操作区“加工”中，MCS 或 WCS 实际值窗口可以通过两种不同的字体大小显示。

通过通道机床数据可以确定用大字体显示的轴数。

MD52010 $MCS_DISP_NUM_AXIS_BIG_FONT	大字体的实际值数量
= 3	大字体的实际值数量（缺省设置）

如果轴数在转换中改变，那么以下控制字体大小的显示模式的通道专用机床数据是相关的：

MD52011 $MCS_ADJUST_NUM_AXIS_BIG_FONT	将大字体的实际值数量与几何轴数量动态配合
= 0	只有 MD52010 是有效的。如此就规定了数量。
= 1	只有几何轴用大字体显示。MD52010 被忽略。
= 2	几何轴的数量和内容 MD52010 用大字体显示。

9.2 插入用户专用商标

在老一代的 SINUMERIK Operate 操作界面上，西门子商标会显示在操作区域“加工”的标题行中。

在新一代的 SINUMERIK Operate 操作界面上，西门子商标会一直显示在标题行中。

您可以用自己公司的商标来替代西门子商标。

在老一代操作界面中加入 OEM 商标

步骤：

1. 命名自己的商标“logo.png”。

   说明

   文件名只能由小写字母构成。

2. 根据位图大小，在如下目录中保存自己的商标文件： /**oem**/sinumerik/hmi/ico oder /**user**/sinumerik/hmi/ico

目录	位图大小
/Ico640	122 x 19
/Ico800	160 x 24
/Ico1024	199 x 30

皮肤 1 和皮肤 2 显示的 OEM 商标

步骤：

1. 根据屏幕分辨率的大小来命名一份商标文件，格式为：company_label_XXXXxYYYY.png。XXXXxYYYY 是占位符，表示屏幕分辨率，比如：company_label_1024x768.png。
2. 在以下其中一个目录中保存自己的商标：
   /**oem**/sinumerik/hmi/skins/**skin1**/ico/ico640
   /**user**/sinumerik/hmi/skins/**skin1**/ico/ico640
   或
   /**oem**/sinumerik/hmi/skins/**skin2**/ico/ico640
   /**user**/sinumerik/hmi/skins/**skin2**/ico/ico640

说明

显示错误

在使用新一代操作软件时，必须从目录 /oem 或 /user 中删除自定义商标文件“logo.png”。

自定义商标的大小应适合使用的屏幕分辨率，不能超出标题行的高度。

屏幕分辨率和位图大小的关系：

屏幕分辨率	最大的位图高度
640 x 480	32
760 x 505	32
800 x 480	32
800 x 600	40
1080 x 810	53
1024 x 768	51
1024 x 1014	51
1280 x 768	51
1280 x 800	51
1280 x 1024	53
1300 x 768	51
1366 x 768	51
1440 x 1080	72
1454 x 1080	70
1854 x 1014	51
1854 x 1080	70
1920 x 1080	70

9.3 设置 G 代码组的显示

在操作区域“加工”中可以看到“G 功能”窗口。

窗口中最多可以显示 16 个 G 代码组。 在文件“slmagcodeconfig.ini”中设置，哪个 G 代码组在哪个位置显示。

设置 G 代码组

段落	含义
Selected G-Groups	该区域用于在西门子模式下运行控制系统。
Selected G-Groups ISO-Mode	该区域用于在 ISO 模式下运行控制系统。

按以下步骤输入 G 代码组和位置：

SelectedGGroupN = M	
N	G 代码组 M 的显示位置。 如果位置需要为空，则不输入任何编号 (M)，例如“SelectedGGroup1 = ”
M	在 N 位置显示的 G 代码组。

文件结构

```
[Selected G-Groups]
SelectedGGroup1 = 1
SelectedGGroup2 = 2
SelectedGGroup3 = 3
SelectedGGroup4 = 4
SelectedGGroup5 = 5
SelectedGGroup6 = 6
SelectedGGroup7 = 7
SelectedGGroup8 = 8
SelectedGGroup9 = 9
SelectedGGroup10 = 11
SelectedGGroup12 = 12
SelectedGGroup13 = 13
SelectedGGroup14 = 14
SelectedGGroup15 = 15
SelectedGGroup16 = 16
[Selected G-Groups ISO-Mode]
... (与西门子区相同)
```

步骤

1. 从目录 /**siemens**/sinumerik/hmi/template/cfg 中复制文件“slmagcodeconfig.ini”。
2. 将文件保存在以下目录：/**oem**/sinumerik/hmi/cfg 或者 /**user**/sinumerik/hmi/cfg。
   只要该文件位于用户特定的目录中，则该文件的条目优先于西门子的文件。 如在用户专用文件中缺少注册信息，则使用西门子文件中的。
3. 打开文件并将相应 G 代码的编号输入到需要的位置“`SelectedGGroup1 = 5`”。
   即在第 1 个位置显示第 5 个 G 代码组。

9.4 配置通道运行信息

概述

在操作区“加工”中，通道运行信息显示在标题的第三行中。具备两种通道运行信息：

1. 要求操作以继续处理程序的通道运行信息，例如：缺少进给使能时，M01 生效。
   
   该通道运行信息通过感叹号加以标注。
2. 要求不进行操作以继续处理程序的通道运行信息。停止条件长时间（>1s）生效，由程序处理中断，例如：停留时间生效，等待换刀。
   
   该通道运行信息通过钟表符号加以标注，特定时间届满后不再显示。

文件

相应的设置在文件“slmahdconfig.ini”中确定。

配置通道运行信息

段落	含义
CondStopIcons	确定和特定通道运行信息一同显示的符号。 如果没有为特定的通道运行信息规定符号，则显示“DefaultIcon”中确定的符号。
CondDelayTime	确定特定信息显示前应持续的时间段（毫秒为单位）。 • 如果没有为特定的通道运行信息规定时间，则使用“DefaultDelayTime”下给定的时间。 • 如果需要立即显示通道运行信息，则必须将该时间设为“0”。

步骤

1. 将自定义的符号文件保存到以下路径下的预设子目录中：
   /**oem**/sinumerik/hmi/ico 或
   /**user**/sinumerik/hmi/ico
   根据屏幕分辨率提供以下子目录：

目录	PNG 图像大小（像素）
/Ico640	16 x 16
/Ico800	17 x 17
/Ico1024	27 x 27

2. 从以下目录复制样板文件“slmahdconfig.ini”
   /**siemens**/sinumerik/hmi/template/cfg.
3. 将文件保存在以下目录：
   /**oem**/sinumerik/hmi/cfg 或
   /**user**/sinumerik/hmi/cfg
4. 打开该文件并执行所需的设置：
   – 确定自定义符号，其中修改用于通道运行信息的符号的文件名。
   – 确定另一时间段，其中修改通道运行信息的时间值。
5. 保存设置。
   启动后这些配置才会生效。

示例文件“slmahdconfig.ini”

配置

以下标准信息设计用于技术故障：

```
[CondStopIcons]
DefaultIcon = condwait.png
1 =  condstop.png   ; 无 NC 就绪
2 =  condstop.png   ; 无 BAG 就绪
3 =  condstop.png   ; 急停生效
4 =  condstop.png   ; 带停止的报警激活
...
26 = condwait.png   ; 等待位置调节
...
30 = condwait.png   ; 等待安全运行
31 = condwait.png   ; 无通道就绪
```

```
...
46 = condwait.png    ; 快速后退已触发
...


[CondDelayTime]
DefaultDelayTime=1000
1 = 0            ; 无 NC 就绪
2 = 0            ; 无 BAG 就绪
3 = 0            ; 急停生效
4 = 0            ; 带停止的报警激活
...
```

9.5 取消程序测试

有多种程序测试的方法可用于测试和试运行新的零件程序，这些方法可以使用“程序测试”功能来运行。

标准设置下，此功能已激活。可在“加工”操作区域中的“程序控制”窗口通过复选框“PRT”来选择程序测试。

取消程序测试

可使用如下配置机床数据取消程序测试。取消后，状态栏中不再显示“PRT”。

MD51039 $MNS_PROGRAM_CONTROL_MODE_MASK	机床程序控制的选项
位 0 = 0	程序测试功能不可用

更多信息

有关测试的更多信息，请参见

功能手册之基本功能。

9.6 显示铰接位置 STAT 和回转轴位置 TU

在窗口 “MCS” 中显示 STAT 和 TU

对于笛卡尔 PTP (Point-to-Point)运行，如果想要在“机床”操作区中用 STAT 值显示机床的支轴位置以及用 TU 值显示回转轴位置，应将以下通道专用配置机床数据设为 ≠ 0：

MD52032 $MCS_STAT_DISPLAY_BASE	数字基准显示：支轴位置 STAT
= 0	无显示，MD51032 中的值生效（默认）
= 2	显示为二进制值
= 10	显示为十进制值
= 16	显示为十六进制值

MD52033 $MCS_TU_DISPLAY_BASE	数字基准显示：回转轴位置 TU
= 0	无显示，MD51033 中的值生效（默认）
= 2	显示为二进制值
= 10	显示为十进制值
= 16	显示为十六进制值

MD52032 和 MD52033 中的设置也可作用于示教。

更多信息

有关 STAT 和 TU 的更多信息，请参见以下信息：

- 章节“激活“示教”功能 (页 162)”
- 功能手册之坐标转换

9.7 激活“示教”功能

打开示教功能

此外，可通过通用配置机床数据打开示教功能。

设置

MD51034 $MNS_TEACH_MODE	
位 0 = 1	显示软键“程序示教”。按下软键“Accept”（接收），将运行位置接收到程序中。（缺省）
位 1 = 1	示教程序段的接收可以由 PLC 禁止。 DB19.DBX13.0 = 0 接收程序段 DB19.DBX13.0 = 1 不接收程序段

此外，该功能的访问还受存取权限影响。

MD51045 $MNS_ACCESS_TEACH_IN	示教保护等级
= 7	确定示教时将运行过程写入 MDA 缓冲器/零件程序的保护等级。（缺省）

MD51050 $MNS_ACCESS_WRITE_PROGRAM	写入零件程序保护等级
= 7	确定写入零件程序的保护等级。（缺省）

设置通道专用机床数据：

MD20700 $MC_REFP_NC_START_LOCK	不带参考点的 NC 启动禁用
= 0	

MD52207 $MCS_AXIS_USAGE_ATTRIB	轴的属性 [n] 通道轴号
位 14	使能进给轴模式下的主轴用于示教
=1	处于进给轴模式下的主轴也可进行示教

激活示教功能

1. 设置机床数据并执行复位。
2. 切换到 “AUTO” 和 “MDA” 运行方式。
3. 选择机床功能 TEACHIN。
4. 按下软键“程序示教”。

示教未激活

若软键“程序示教”处于未激活状态，则原因可能如下：

- 该程序不处于“复位”、“停止”或“中断”状态。
- 不是所有必须回参考点的轴都已回参考点。

接收 STAT 和 TU

对于笛卡尔 PTP 运行，如需使用 STAT 值将机床的支轴位置以及使用 TU 值将回转轴的位置导入至运行程序段中用于示教，则须勾选“程序示教”下“设置”窗口中的复选框“STAT”和“TU”。

如果以下配置机床数据 ≠ 0，则只显示复选框：

MD52032 $MNS_STAT_DISPLAY_BASE	数字基准显示：支轴位置 STAT
= 0	没有显示
= 2	显示为二进制值（缺省）
= 10	显示为十进制值
= 16	显示为十六进制值

MD52033 $MNS_TU_DISPLAY_BASE	数字基准显示：回转轴位置 TU
= 0	没有显示
= 2	显示为二进制值（缺省）
= 10	显示为十进制值
= 16	显示为十六进制值

9.8 程序段搜索

9.8.1 激活程序段搜索模式

设置程序段查找方式

通过下列机床数据您可以确定 G 代码程序中程序段的查找方式：

MD51028 $MNS_BLOCK_SEARCH_MODE_MASK	设置可用查找方式的位标记
位 0 = 1	程序段查找，带计算，不逼近断点
位 1 = 1	程序段查找，带计算，逼近断点
位 4 = 1	程序段查找，不带计算
位 5 = 1	程序段查找，带试车

通过下列机床数据您可以确定 JobShop 程序(ShopMill, ShopTurn)中程序段的查找方式：

MD51024 $MNS_BLOCK_SEARCH_MODE_MASK_JS	设置可用查找方式的位标记 （ShopMill, ShopTurn 为单通道查找）
位 0 = 1	程序段查找，带计算、不逼近断点（缺省设置）
位 1 = 1	程序段查找，带计算，逼近断点 说明： ShopMill、ShopTurn 程序上只能通过 G 代码进行程序段搜索。
位 3 = 1	跳过 EXTCALL 程序
位 5 = 1	程序段查找，带试车

MD52212 $MCS_FUNCTION_MASK_TECH	工艺通用的功能屏幕
位 3	程序段搜索 - 用于 ShopMill/ShopTurn 的循环
= 0	在程序段搜索循环 PROG_EVENT.SPF 中，不调用循环 E__ASUP 或 F_S_ASUP 。
= 1	在程序段搜索循环 PROG_EVENT.SPF 中，调用循环 E__ASUP （在 Shop-Mill 中）或 F_S_ASUP （在 ShopTurn 中）。
位 5	调用程序段搜索功能 SERUPRO
= 0	在程序段搜索循环 PROG_EVENT.SPF 中不调用循环 CYCLE207。

= 1	在程序段搜索循环 PROG_EVENT.SPF 中（通过程序测试进行程序段搜索 - SERUPRO）调用用于 SERUPRO（CYCLE207）的循环。循环 CYCLE207 在通过程序测试进行程序段搜索（SERUPRO）后可输出收集到的加工指令。
位 15	$MC_TOOL_CARRIER_RESET_VALUE 在程序段搜索后未置位
= 0	在程序段搜索循环 PROG_EVENT.SPF 中，当前 Toolcarrier 会被写入机床数据 $MC_TOOL_CARRIER_RESET_VALUE 中。
=1	在程序段搜索循环 PROG_EVENT.SPF 中，当前 Toolcarrier 不会被写入机床数据 $MC_TOOL_CARRIER_RESET_VALUE 中。
位 16	程序段搜索后不会自动执行平面回转
= 0	在程序段搜索循环 PROG_EVENT.SPF 中，执行标准循环程序段：回转平面。
=1	在程序段搜索循环 PROG_EVENT.SPF 中，不执行标准循环程序段：回转平面。
位 17	程序段搜索后不会自动执行车刀校准
= 0	在程序段搜索循环 PROG_EVENT.SPF 中执行车刀校准程序段。
=1	在程序段搜索循环 PROG_EVENT.SPF 中不执行车刀校准程序段。

说明

只有在机床数据 MD51024 和 MD51028 中定义了不止一种程序段查找方式时（例如：“程序段查找，带计算，不逼近断点”和“程序段查找，带计算，不逼近断点”）才会显示“查找方式”软键。

注意，“查找方式”窗口中有效设置的查找方式也是基于程序的。

9.8.2 由外部处理的加速程序段搜索过程

可通过加速程序段搜索运行零件程序，直至到达程序的所需位置。

设置

若已激活了“跳过 EXCALL”功能，则必须设置下列通用配置机床数据：

MD51028 $MNS_BLOCK_SEARCH_MODE_MASK	可使用搜索模式的位掩码
位 3 = 1 (33H 至 3BH)	

对于外部主程序而言，不包含计算的程序段搜索速度总是较快的。 只会加载自搜索位置起的程序段。

9.9 多通道支持

在“机床”操作区中最多可以看到运行方式组中相互四个任意通道。通道数量取决于所用操作面板的显示尺寸和分辨率。

通过使用多通道编辑器，可在多通道机床上同时打开、修改或创建多个时间上相互独立的程序。此时相关程序会并排显示在编辑器中。

在实际加工前还可在控制系统上进行程序模拟。

此外可根据同步点校正程序。

操作面板的视图方法

操作面板支持以下视图：

操作面板	最大通道数
OP12	2 个通道
OP15	3 个通道
OP15 black	4 个通道
OP19	4 个通道
TOP 1500	4 个通道
TOP 1900	4 个通道
TOP 2200	4 个通道

前提条件

- 多通道数据仅与第一工艺为车削的机床相关。
- 对于多通道机床，以下数据必须在所有通道中都相同：
  - 尺寸单位
  - 零点偏移（例如 G54）
  - 零点偏移 Z 值（可选）
  - 毛坯
  - 转速限制
- 此外下列 JobShop 数据必须在所有通道中相同：
  - 回退平面（分为主主轴和副主轴）
  - 尾座（分为主主轴和副主轴）
  - 换刀点（分为主主轴和副主轴）
  - 安全距离
  - 顺铣和逆铣

创建多通道程序

进行多通道加工时，需要包含以下数据的工作列表：

- 程序名称
- 程序类型
- G 代码
- ShopTurn 程序
- 通道分配



软件选件

需要选件“ShopMill/ShopTurn”用于创建和编辑 ShopTurn/ShopMill 程序。

在创建了一个工作列表后，可在“多通道数据”参数屏幕中将 G 代码和 JobShop 数据进行汇总，之后再将其作为循环调用保存在工作列表中。只要工作列表中至少有一个 ShopTurn 程序，就会显示“多通道数据”参数屏幕。

仅编程 G 代码程序

如果在工作列表中只有 G 代码程序，可以取消“多通道数据”功能。为此设置以下机床数据：

MD51228 $MNS_FUNCTION_MASK_TECH	工艺通用的功能屏幕
位 0	G 代码编程，无多通道数据
= 1	在只有 G 代码程序的工作列表中不提供多通道数据。不显示软键“多通道数据”。

9.10 手动加工

概述

“Manual Machine”功能使人员可以利用丰富多样的循环进行手动加工，而无需编程。



软件选件

使用该功能需要下列软件选件：

“ShopMill/ShopTurn”

激活

MD52212 $MCS_FUNCTION_MASK_TECH	工艺通用的功能屏幕
位 8	Manual Machine(ShopMill/ShopTurn)

使能

MD52214 $MCS_FUNCTION_MASK_MILL	铣削功能屏幕
位 5	使能刀具主轴的控制
= 0	通过 NC/PLC 接口
= 1	通过操作界面
位 6	使能旋转主轴的控制
= 0	通过 NC/PLC 接口
= 1	通过操作界面

MD52218 $MCS_FUNCTION_MASK_TURN	车削功能屏幕
位 3	使能主主轴的控制
= 0	通过 NC/PLC 接口
= 1	通过操作界面
位 4	使能副主轴的控制
= 0	通过 NC/PLC 接口
= 1	通过操作界面
位 5	使能刀具主轴的控制
= 0	通过 NC/PLC 接口
= 1	通过操作界面

SD43200 $SA_SPIND_S	通过 NC/PLC 接口信号给出主轴启动转速；通过操作软件输出数值
SD43202 $SA_SPIND_CONSTCUT_S	通过 NC/PLC 接口信号给出主轴启动切削速度；通过操作软件输出数值
SD43206 $SA_SPIND_SPEED_TYPE	通过 NC/PLC 接口信号给出主轴启动转速类型；通过操作软件输出数值

更多设置

推荐以下更多设置：

手轮	
MD11346 $MN_HANDWH_TRUE_DISTANCE = 3	手轮设定行程还是设定速度
MD11602 $MN_ASUP_START_MASK 位 0，3 =1	忽略 ASUP 的停止原因
MD20150 $MC_G_CODE_RESET_VALUE [7] = 2	G 功能组的初始化设置；G54 引导启动有效
MD20624 $MC_HANDWH_CHAN_STOP_COND = 0xFFFF	确定通道专用的手轮运行性能

手轮	
MD32084 HANDWH_STOP_COND = 0x7FF	手轮运行性能
MD52212 FUNCTION_MASK_TECH 位 9 = 1	工艺通用的功能屏幕；通过软键设置零点偏移

说明

如果机床数据 MD52212 FUNCTION_MASK_TECH 的位 9 被置位，则会在窗口“零点偏移 – G54 ... G599”中显示软键“NPV 选择”。

工作区域限制	
MD28600 $MC_MM_NUM_WORKAREA_CS_GROUPS > 0	坐标系专用的工作区域限制数量
MD20150 $MC_GCODE_RESET_VALUES[59] = 2	G 功能组的初始化设置；激活功能组的工作区域限制

说明

如果机床数据 MD20150 $MC_GCODE_RESET_VALUES[59] = 2 的位被置位，则会在窗口 “Manual Machine” 中显示软键“挡块”。前提是 MD28600 $MC_MM_NUM_WORKAREA_CS_GROUPS 的值也被置位为 1。

进给率	
SD42600 $SC_JOG_FEED_PER_REV_SOURCE = -3	JOG 方式下的控制系统转速进给
SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE [X 轴] = -3	定位轴 X 的旋转进给率
SD43300 $SA_ASSIGN_FEED_PER_REV_SOURCE [Z 轴] = -3	定位轴 Z 的旋转进给率

说明

要注意轴专用的设置数据 SD43300 SA_ASSIGN_FEED_PER_REV_SOURCE 必须相同。

更多信息

有关加工区域限制和使用 Manual Machine 加工的更多信息，请参见以下文档：

- 操作手册，车削版/铣削版
- NC 编程手册

9.11 用户状态显示（OEM）

功能

在“加工”操作区域中，可以在标题的第二行显示用户自定义图标。此时，程序名称显示在标题第三行的右侧。有停止条件激活时，它会覆盖程序名称。

用户图标的显示是由 PLC 位来控制的，这些图标以 PNG 格式的文件提供，按照各自的分辨率保存在文件系统中的目录：`/user/sinumerik/hmi/ico/ico640` 下。用户图标可以最多占据**标题内的 16 个位置**。

选型

用户图标的设计在文件 `slmahdconfig.ini` 中的段落[UserIcons]中进行。在该段落中，需要指定：

- PLC 双字的基准地址（包含了控制用户图标显示的位）。
- 图标的文件名称和图标需要显示的位置。

您也可以选择使用下面的变量进行定义：

- **USER_ICON_BASE**
  一共有 32 个位可使用，从 UI_0（最高位）到 UI_31（最低位）。
  示例：USER_ICON_BASE=DB80.DBB0
  ⇒ UI_0 = DB80DBX3.0 和 UI_31 = DB80.DBX0.7
- **USER_ICON_BASE256**
  一共有 256 个位可使用，从 UI_0（最高位）到 UI_255（最低位）。
  示例：USER_ICON_BASE256=DB19.DBB80
  ⇒ UI_0 = DB19.DBX111.0 而 UI_255 = DB19.DBX80.7

示例

[UserIcons]

USER_ICON_BASE=DB19.DBB80

;设置 plc 双字的起始字节(DB19.DBD80)来显示图标

UI_0 = icon1.png,4

;位 0(DB19.DBX83.0)置位后在位置 4 上显示 icon1.png

UI_1 = icon2.png,6

;位 1(DB19.DBX83.1)置位后在位置 6 上显示 icon2.png

叠加

多项选择中（多个图标需要在同一个位置上显示）：

在同一位置上，高标识符编号的图标覆盖低编号的图标。即使使用了少于 16 个有效的选择位，也可以进行多项选择。PLC 可利用该叠加，使高标识符编号的重要图标覆盖低编号的次要图标。

说明

如果给一个 NC/PLC 单元分配了 2 个 HMI 应用（M:N），则 PLC 必须首先根据接口确定已激活哪些 HMI 接口。只有当 HMI 应用程序激活时，才能向 PLC 请求图标。

应用

单项选择中，4 个选择字节中的最多 16 个位可以置位，用于各个不同的位置。

- 一个位置有 32 个可选图标，每个图标由一个位选择（在同一时间只有一个激活的选择位）
- 允许的 16 个位置中的每个位置有 2 个可选图标，每个位置需要使用 32 个选择位中的 2 个位（在同一时间内最多有 16 个有效的选择位）
- 综合起来就是：最多有 16 个显示位置和 32 个图标（在同一时间最多有 16 个有效的选择位）
- 多于 16 个选择位。

9.12 配置叠加显示

配置屏幕显示

手轮偏移(DRF)或工具长度补偿造成的叠加可由系统变量$AA_TOFF 的值直接显示在屏幕上。

用于叠加的屏幕显示可通过以下通道专用的机床数据配置屏幕显示：

MD52210 $MCS_FUNCTION_MASK_DISP	显示功能屏幕
位 14	显示机床下的叠加 DRF
位 15	显示机床下的叠加刀具（$AA_TOFF）

9.13 激活加工时间记录

如果要在自动运行中计算并保存加工时间，可以通过操作面板进行设置。

更多信息

有关设置的更多信息，请参见

操作手册，通用版/车削版/铣削版/磨削版。

启动时间记录

为了使用该功能，必须将全部通道中的下列通道机床数据按以下方式进行设置，即使并未激活所有通道的加工时间记录：

MD18370 $MN_MM_PROTOC_NUM_FILES[1]	日志文件的最大数量
= 2	针对 OEM 应用程序

MD18371 $MN_MM_PROTOC_NUM_ETPD_STD_LIST[1]	标准数据列表 ETPD 的数量
= 6	跟踪系统功能所需的 BTSS 模块中的标准数据列表 ETPD 的数量

MD28300 $MC_MM_PROTOC_USER_ACTIVE[1]	为用户激活记录
= 1	针对系统功能

MD28302 $MC_MM_PROTOC_NUM_ETP_STD_TYP[1]	标准事件类型 ETP 的数量
= 5	所需的 BTSS 模块中的标准事件类型 ETP 的数量

说明

加工时间记录时，机床数据 MD18370、MD18371 和 MD28302 中较大的值也生效。

模拟和记录 10

10.1 模拟概述



软件选件

3D 模拟显示功能需要使用选件
“3D 模拟 1（成品件）”。

SINUMERIK Operate 中集成了一个单独的程序编译器(SNCK)和一个单独的模拟数据环境，为加工模拟提供支持。SNCK 编译器可编译 SINUMERIK 系列产品的所有句法，并通过和 NCK 数据环境的对比纳入了机床上安装的特殊用户选件。在必要时，操作软件会一直比较模拟数据和 NCK 数据环境（初始化数据、宏指令、用户数据、刀具数据和加工循环），或只在刀具数据或加工循环有所变化时进行数据对比。

在具有特定运动结构的机床上，可以在系统界面上在工件坐标系中模拟加工工艺的整个过程，重点是钻削、铣削、车削和磨削工艺。

- 成品的模拟是用真实 NC 数据进行的。
- 在每次更换零件程序时系统会自动对比 NC 数据。
- 模拟功能可以动态展示整个加工过程，甚至支持 5 轴转换(TRAORI) 和回转功能。
- 铣床上的车削
- 传统铣床加工通道模拟。
- 在带 B 轴的车床上，可最多模拟 4 个加工通道。
- 以智能的方式确定程序段处理时间和程序处理时间。
- 连续不断地绘制工件，模拟画面更新更加迅速
- 最佳分辨率，无论截取哪个部分的画面。
- 可模拟任何切削。
- 从 NCU 720.x 起，在加工一个工件的同时可模拟另一个工件的加工过程。
- 模拟 3D 图（可选）。

模拟存储器配置

- 模拟画面更新非常迅速，但这也意味着动态分辨率比较低。
- 在复杂的工件上，绘图程序过慢，在绘制某些非常复杂的工件时甚至无法工作。
- “Quickview”，用于模具制造零件程序的快速查看。
- 在使用某些带特殊运动结构的编译循环时无法使用模拟功能。
- 仅部分支持编译循环。
- 无法读出毛坯的 CAD 数据，比如在铸造加工中。
- 无法模拟轴的耦合过程。（联动、轴的主值耦合、电子齿轮箱）：
  - 联动：TRAILON(<从动轴>, <主动轴>, <耦合系数>)
  - 轴的主值耦合：LEADON(从动轴, 主动轴, 曲线表 n)，实际值耦合和设定值耦合
  - 电子齿轮箱：EGDEF(FA, LA1, 耦合 1, LA2, 耦合 2, ... LA5, 耦合 5), EGON, EGONSYN, EGONSYNE
- 在模拟中参考 G74 无效。
- //不支持
  - 轴容器
  - 对零件程序和 PLC 的依赖性
  - 多工位加工
  - CNC 功能同步动作，用于 NC 的安全可编程逻辑（SPL）
- 仅部分支持不同通道的不同技术和不同的轴坐标系统。
- 在磨削工艺中不可使用 3D 模拟。
- 不评估软件限位开关：
  在一个刀具切换的生产周期里，机床数据“软件限位开关+Z 轴”作为切换位置使用：
  G0 Z=$MA_POS_LIMIT_PLUS[Z]-0.1
  在模拟中该位置不可用，而是机床数据的预定设置 = 1e8 = 100.000.000。这些预定的位置在模拟中不处理。因此，模拟很慢，加工时间激增，例如计算的时间为 555 小时，而不是 2 分钟。

- 模拟中的程序启动
  在一个通道中设置了 MD22622 $MCS_DISABLE_PLC_START 时，此通道在模拟中不会自动启动。只能通过零件程序指令“START”从另一个通道启动。该功能不适用于 SINUMERIK 828D。
- 模拟中的报警显示
  在模拟功能中报警的输出方式和正常程序处理过程基本一样，只是它最多显示五条报警。报警位于图形上方，以便和机床报警加以区分。
  如果同步记录和模拟具有不同的故障报告动作（例如：句法错误），则可能是在编程中定义了不同的硬件访问方式：模拟不访问硬件，只是内部读取输入 0，然后结束。而同步记录则出现报警。

10.2 设置用于模拟的坐标系

车削工艺

例如对于机床运动学：

- 使用两个几何轴的传统车削
- 三个主轴：主主轴，副主轴，变量
- 副主轴滑块，尾座为 NC 轴。
- B 轴：较准刀具主轴上的车刀
- 使用几何轴的铣削：TRANSMIT, TRACYL, TRAANG

用于车削工艺：

MD52200 $MCS_TECHNOLOGY = 1	车削工艺
MD52000 $MCS_DISP_COORDINATE_SYSTEM = 34（举例）	坐标系的位置（可变）

铣削工艺

例如对于机床运动学：

- 5 轴铣削：回转/TRAORI
- 回转头切换

用于铣削工艺：

MD52200 $MCS_TECHNOLOGY = 2	铣削工艺
MD52000 $MCS_DISP_COORDINATE_SYSTEM = 0	坐标系的位置（可变）

磨削工艺

适用于磨削

MD52200 $MCS_TECHNOLOGY = 3	外圆磨削工艺
MD52200 $MCS_TECHNOLOGY = 4	平面磨削工艺
MD52000 $MCS_DISP_COORDINATE_SYSTEM = 0	使用操作台的外圆磨削和平面磨削
MD52000 $MCS_DISP_COORDINATE_SYSTEM = 6	使用工作台的平面磨削

轴的含义

MD52206 $MCS_AXIS_USAGE[n]	通道中轴的含义 [n]为通道轴号
= 0	没有特定含义
= 1	刀具主轴（动力刀具）
= 2	辅助主轴（动力刀具）
= 3	主主轴（车削）
= 4	主主轴的 C 轴（车削）
= 5	副主轴（车削）
= 6	副主轴的 C 轴（车削）
= 7	副主轴的直线轴（车削）
= 8	尾座（车削）
= 9	中心架（车削）

说明

为了不出错，只能将同一轴定义为所有通道的主主轴或副主轴。

请通过以下通道专用的机床数据输入旋转轴的旋转方向，该旋转轴没有在 ToolCarrier 或 5 轴转换中配置。

MD52207 $MCS_AXIS_USAGE_ATTRIB[n]	轴的属性 [n]通道轴号
位 0	围绕第 1 几何轴旋转（旋转轴）
位 1	围绕第 2 几何轴旋转（旋转轴）
位 2	围绕第 3 几何轴旋转（旋转轴）
位 3	旋转方向逆转（适用于旋转轴）

机床数据 MD52290 $MCS_SIM_DISPLAY_CONFIG 仅在 OP019 上起作用。对于 SINUMERIK 828D 没有任何意义。

MD52290 $MCS_SIM_DISPLAY_CONFIG	通道的状态栏在模拟窗口中的位置。
位 0	左上角
位 1	右上角
位 2	左下角
位 3	右下角

在以下通道专用机床数据中，在每个通道中至少为一根几何轴设定一个不为 0.0 的值，例如 0.001。当值为 0.0 时，系统会认为此参数尚未设置。

MD53230 $MCS_SIM_START_POSITION	启动模拟时的轴位置
只有在为至少一根几何轴设置了不为 0 的值时，才能进行模拟。	

取消模拟

设置以下机床数据关闭模拟功能：

MD51226 $MNS_FUNCTION_MASK_SIM	模拟的功能标记
位 1 = 1	取消模拟
位 10 = 1	隐藏零点符号

运动链模拟

如要使用运动链模拟，请设置以下机床数据：

MD51226 $MNS_FUNCTION_MASK_SIM	模拟的功能标记
位 22	用于运动链模拟的机床模型（也无碰撞监测）

换刀时间

以下机床数据确定了模拟时执行一次换刀需要多长时间：

MD10190 $MN_TOOL_CHANGE_TIME	模拟时的换刀时间
= 0	秒（默认值：0）

此值仅代替了原则上在被模拟的换刀过程中不累积的时间，如等待 PLC 指令的时间（M206）。

如果运行指令包含在换刀循环中，那么这些指令也被算在预估的模拟过程的时长之内。

请通过以下示例补充换刀循环中的编程，使得 MD10190 中的值作为换刀总时长算入模拟过程的时长评估中：

```
IF $MN_SIM_ENVIRONMENT B_AND 'B100'
 M206 ; vgl. MD 22560 $MC_TOOL_CHANGE_M_CODE
 RET
ENDIF
```

10.3 同步记录概述



软件选件

而“同步记录”功能需要使用选件“同步记录（实时模拟）”。



软件选件

显示尾座需要选件“ShopMill/ShopTurn”。

软件选件

同步记录功能 3D 图需要使用选件
“3D 模拟 1（成品件）”。

在机床开展加工的同时，系统可在屏幕上同步绘制出刀具的运动轨迹，以 3 面视图或 3D 视图显示。工件的外观、视图与图形化的“模拟”功能一致。

说明

为避免遗漏某些显示，在启动系统前必须提前激活同步记录功能。

示例：激活主轴卡盘数据中的尾座图

步骤：

1. 通过在程序中选择使尾座可见。
2. 给出尾座尺寸：
   在操作区域内选择“参数”→“设置数据”→“主轴卡盘数据”并输入尾座长度和直径参数“ZR”和“XR”。
   在下列机床数据中写入尾座的尺寸：

MD52218 $MCS_FUNCTION_MASK_TURN	车削功能屏幕
位 2 = 1	激活尾座： 在程序中选择“尾座”后，尾座即可见。

MD53240 $MAS_ SPINDLE_PARAMETER[i]	主轴卡盘数据，单位：mm
[0]:卡盘尺寸 [1]:挡块尺寸 [2]:卡爪尺寸	

MD53241 $MAS_ SPINDLE_CHUCK_TYPE	主轴卡盘类型
0 = 从外面夹紧 1 = 从里面夹紧	

MD53242 $MAS_TAILSTOCK_PARAMETER[i]	尾座尺寸，单位：mm
[0]:尾座直径 [1]:尾座长度	

激活处理时间的显示

在功能“程序运行时间”下提供了计时器，它相当于系统变量。NCK 专用的计时器会一直保持工作，以测量上次控制系统启动后经过的时间，而该通道专用的计时器必须通过下述通道专用的机床数据启动：

MD27860 $MC_PROCESSTIMER_MODE	程序运行时间计算使能。
位 0 = 1	计算所有程序的总运行时间有效 ($AC_OPERATING_TIME)。
位 1 = 1	计算当前程序运行时间有效 ($AC_CYCLE_TIME)。
位 4 = 1	空运行进给有效时计算。
位 5 = 1	程序测试时计算。

10.4 毛坯夹具

毛坯夹具功能可用来选择并确定毛坯的夹具位置。

实行该功能要求机床能满足以下前提条件之一或多个：

前提条件

- 机床具有多种夹具方法可供选择。
- 机床具有一种不会默认显示在 Z 方向上（例如：钻头）的夹具方法。

以下情况中可以在机床数据 MD53220 和 MD52207 中注明回转轴的位置和方向：

- 回转轴的位置和方向朝着 MCS 零点相反的方向移动，但不能通过 Toolcarrier 或 TRAORI 确定。
- 车床 X 轴或 Z 轴上的主轴或副主轴的基准点朝着 MCS 零点相反的方向移动。
- 副主轴不能借助于滑块运动，而是位于一个固定的位置上。

必须通过以下机床数据指定机床坐标轴中的位置，才能在模拟/同步记录时将旋转轴显示在正确的位置：

MD53220 $MAS_AXIS_MCS_POSITION[n]	MCS 中轴的位置 [n] 0 = X, 1 = Y, 2 = Z

有关回转轴转换的**更多信息**，请参见章节 “工艺和循环 (页 407)”。

功能

设定以下通道专用的机床数据：

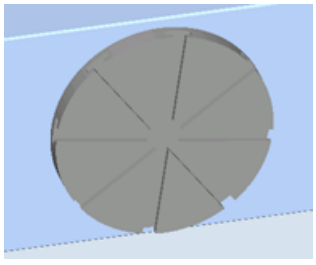
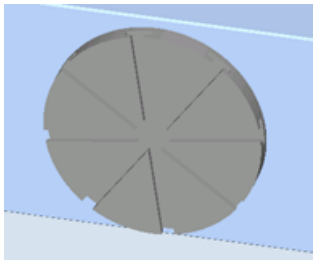
MD52207 $MCS_AXIS_USAGE_ATTRIB[n]	通道中轴的属性 [n] 通道轴号
位 0	围绕第 1 几何轴旋转（旋转轴）
位 1	围绕第 2 几何轴旋转（旋转轴）
位 2	围绕第 3 几何轴旋转（旋转轴）
位 8	提供用于毛坯夹具的旋转轴（铣削机床上）
位 13	选择毛坯时从激活的零点偏移中选择轴位置（旋转轴）
= 0	启动模拟或记录时可设立的旋转轴会移动至 MD53220 定义的位置。
= 1	在毛坯命令中可设立的旋转轴会移动至当前生效的零点偏移的零点上。

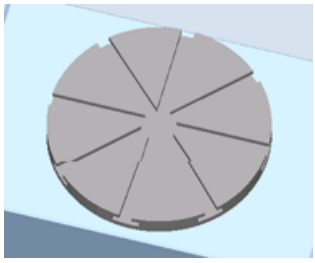
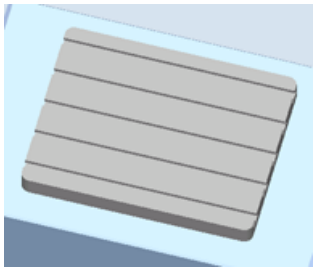
MD52214 $MCS_FUNCTION_MASK_MILL	铣削功能屏幕
位 1	提供用于毛坯夹具的固定工作台（铣床上）

从示例中获取 MD20080 的机床配置。

功能可能性

程序头中可选择毛坯夹具 A、B、C 并设置工作台：

夹具	视图	机床数据
A 轴		MD52207 $MCS_AXIS_USAGE_ATTRIB 位 0 和 位 8 = 1
B 轴		MD52207 $MCS_AXIS_USAGE_ATTRIB 位 1 和 位 8 = 1

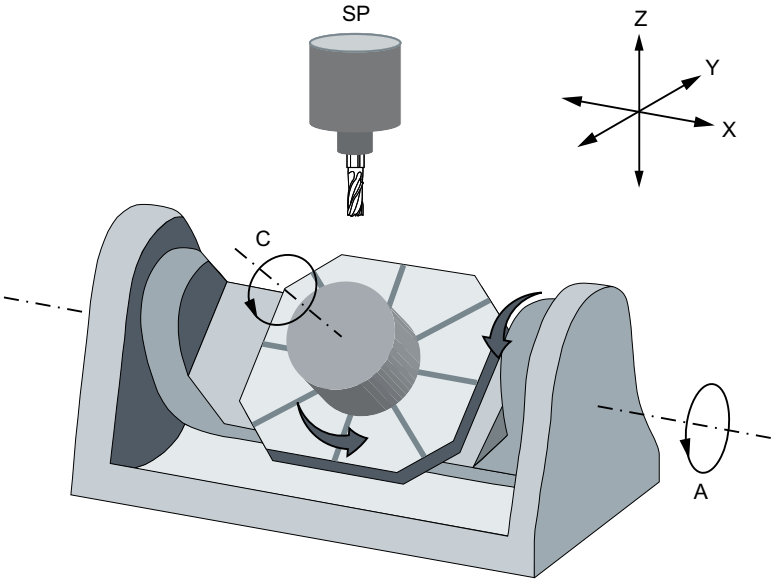
夹具	视图	机床数据
C 轴		MD52207 $MCS_AXIS_USAGE_ATTRIB 位 2 和 位 8 = 1
工作台		MD52214 $MCS_FUNCTION_MASK_MILL 位 1 = 1

说明

为了在程序头上保留选择可能性，必须为毛坯件夹紧装置定义两个旋转轴。

机床配置示例

A 轴围绕 X 轴旋转的机床配置示例：



MD20080 $MC_AXCONF_CHANAX_NAME_TAB[ ]	通道内的通道轴名称
[0]	X
[1]	Y
[2]	Z
[3]	主轴
[4]	A
[5]	C

MD52207 $MCS_AXIS_USAGE_ATTRIB[4]	旋转轴 A 的属性
= 181H	
位 0 = 1	围绕第 1 几何轴旋转（旋转轴）
位 8 = 1	提供用于毛坯夹具的旋转轴（铣削机床上）

更多信息

有关 ShopMill 设置的更多信息，请参见

操作手册，铣削版。

主轴功能 11

11.1 主轴控制

主轴控制可用以下设置：

程序结束

下列 M 功能用于程序结束：

MD10714 $MN_M_NO_FCT_EOP 中的 M 功能	主程序结束	主轴继续运行
M2，M30	主程序结束并返回程序开头	主轴停止
M17	子程序结束并返回主程序	主轴继续运行

如果机床数据 MD10714 $MN_M_NO_FCT_EOP 不等于零，则 M2 / M30（NC 程序结束）和 MD10714 $MN_M_NO_FCT_EOP 中设置的 M 功能之间会有区别。

示例

MD10714 $MN_M_NO_FCT_EOP = 32 表示“M32”，即运行方式“JOG”或“MDA”下创建的程序的结尾。

此外还需要此功能，用于在手动运行中持续启动主轴（例如用于对刀）。

配置按键

通过机床操作面板的按键手动控制主轴时，需在主轴数据块中使用下列接口信号：

DB3x.DBX30.0	主轴停止
DB3x.DBX30.1	主轴顺转
DB3x.DBX30.2	主轴逆转

可在以下状态中启动和停止主轴：

DB21.DBX35.7= 1	通道状态: 复位
DB21.DBX35.6 = 1	通道状态：中断
DB21.DBX35.3 = 1	程序状态：中断

说明

需要在程序运行的情况下停止主轴时，请在用户 PLC 中设置接口信号，DB3x.DBX4.3，“进给轴停止/主轴停止”。

其他设置：

MD11450 $MN_SEARCH_RUN_MODE		编程搜索模式
位 1 = 1	输出动作程序段后自动启动 ASUB（参见 MD11620 PROG_EVENT_NAME）。 当 ASUB 结束后才输出报警 10208。	

更多信息

主轴配置的详细描述请参见

功能手册之进给轴和主轴。

旋转方向（仅适用于 ShopTurn）

在 ShopTurn 界面中，确保主轴和 C 轴的旋转方向显示正确，并且在编程 ShopTurn 功能时执行了正确的旋转方向。此处的设置必须与机床上主轴/ C 轴的实际旋转方向一致。

- 使用柱面转换和端面加工功能前，必须确保已正确调试了功能。
  参见柱面转换（TRACYL） (页 456)
- 此外可以通过机床数据定义，在 ShopTurn 界面中使用 M3 时显示哪些主轴/ C 轴旋转方向：
  参见：定义旋转方向的章节：设置用于车削的 ShopTurn 循环 (页 443)
- 通过接口信号 DB31, ... DBX17.6 可以指定 C 轴正向旋转相当于主轴哪个转向 (M3/M4)。该位用来确定，M3 和 C+ 是否为同一方向（= 0）或相反方向 （= 1）。相应的设置方法参见下列章节：
  - 副主轴旋转方向设置 ShopTurn 中的副主轴 (页 448)
  - 主主轴旋转方向设置用于车削的 ShopTurn 循环 (页 443)

11.2 模拟主轴

表示模拟主轴的机床轴索引

没有指定驱动（VSA、HSA）的主轴的负载显示可由 PLC 控制。

您需要在以下机床数据中输入模拟主轴的轴索引，以便操作软件将主轴判定为模拟主轴。

MD51068 $MNS_SPIND_DRIVELOAD_FROM_PLC 1	主轴 1 的机床轴索引 负载显示来自 PLC
	主轴（模拟）的机床轴索引，T,F,S 窗口中该主轴的负载显示来自 PLC (DB19.DBB6) 。

MD51069 $MNS_SPIND_DRIVELOAD_FROM_PLC 2	主轴 2 的机床轴索引 负载显示来自 PLC
	主轴（模拟）的机床轴索引，T,F,S 窗口中该主轴的负载显示来自 PLC (DB19.DBB7)。

11.3 主轴诊断

11.3.1 主轴诊断

软件选件

“主轴诊断”功能需要选件“S-Monitor”（主轴监测器）。

利用各种状态信号可对主主轴的驱动器进行监控。系统通过主轴上的 DRIVE CliQ 接口读入驱动器内的这些状态信号，系统会对状态信号进行检测，使主轴运行状态的诊断更加高效：借助软件选件“S-Monitor（主轴监示器）”，操作界面上会显示以下信息：

- 运行小时
- 温度
- 转速/转矩
- 夹紧系统
- 物流数据

前提条件

- 带 DQI 编码器的主轴：r0459，位 14 = 1。
- 为主轴配置了驱动报文 139。
- 机床轴具有主轴功能，即机床数据 MD35000 $MA_SPIND_ASSIGN_TO_MACHAX > 0
  该数据的值即主轴号。

更多信息

关于配备 SMI 24 的主轴（Weiss 主轴）的详细信息，请参见功能手册的“轴和主轴”一章。

数据一览

在窗口“主轴诊断 S1”中显示以下信息：

参数	值
运行小时 • 主轴处于闭环控制中 • 主轴旋转	h
夹紧次数（即换刀次数） 夹紧循环定义：夹紧状态：已松开 [3] → 夹紧状态：已夹紧 [7、8 或 10] → 夹紧状态：已松开 [3]。	
电机温度	°C
夹紧状态（传感器 1）	V
活塞松开（传感器 4）	
例如：定义的轴旋转角位（传感器 5）	是/否
最大转速	rpm

步骤

1. 选择操作区域“诊断”。
2. 按下菜单扩展键。
3. 按下软键“主轴诊断”，显示数据一览表。
   窗口“主轴诊断 S1”打开。

说明

如果有多个主轴在工作，可通过软键“主轴+”或“主轴-”选择所需主轴。

参见

使用 Weiss 公司的混合主轴换刀 (https://support.industry.siemens.com/cs/ww/en/view/109475707)

11.3.2 温度

检测和显示的温度取决于安装在主轴中的传感器的数量和安装位置（例如：传感器 S6 用于测量前轴承温度）。 如果温度传感器类型的参数 p0601 = 0 或 1，则不显示电机温度。

说明

温度限值

此处所显示的限值针对的是主轴监控，不会影响在驱动中设置的适用于主轴停止响应的驱动参数值。 温度限值在 SMI24 中为缺省设置，无法在该窗口中修改。 必要时还须调整驱动的关闭限值。

最多可以检测 3 个温度：

信号	传感器	列/含义	说明
电机温度 (页 195)	KTY	实际值	当前电机温度。 如果超温保护存在且响应，则显示值 250 °C。
电机温度 报警		限值	保存的电机温度阈值，超出该阈值会因 KTY 电机温度过高而出现报警。
		超限时间	显示累积的超限时间（单位：小时）。
		最后一次超限	显示最后一次超限时的时间戳[日期]和[时间]。
		超限 次数	累积的超限次数。

信号	传感器	列/含义	说明
电机温度 故障		限值	保存的电机温度阈值，超出该阈值会因 KTY 电机温度过高而出现故障。
		超限时间	显示累积的超限时间（单位：小时）。
		最后一次超限	显示最后一次超限时的时间戳[日期]和[时间]。
		超限 次数	累积的超限次数
超温保护	PTC	实际值	如果超温保护响应，则显示“激活”（红色）。“实际值”一列中显示的电机温度值为 250 °C。
		限值	保存的电机温度阈值，超出该阈值会因 PTC 绕组电机温度过高而出现故障。
		最后一次超限	显示最后一次超限时的时间戳[日期]和[时间]。
		超限 次数	累积的超限次数。
附加温度 (页 196) （例如：前轴承温度）	S6 (KTY)	实际值	测量点上的当前温度。
附加温度 报警		限值	保存的温度阈值，超出该阈值会因轴承温度过高而出现报警。
		超限时间	显示累积的超限时间（单位：小时）。
		最后一次超限	显示最后一次超限时的时间戳[日期]和[时间]。
		超限 次数	累积的超限次数。

信号	传感器	列/含义	说明
附加温度 故障		限值	保存的温度阈值，超出该阈值会因轴承温度过高而出现故障。
		超限时间	显示累积的超限时间（单位：小时）。
		最后一次超限	显示最后一次超限时的时间戳[日期]和[时间]。
		超限 次数	累积的超限次数。

11.3.3 电机温度传感器

停止响应

SMI24 中存有电机温度传感器的报警和故障阈值，在操作界面上显示为限值。 这两个值既可用于 SMI24 的统计分析，也可用于驱动的停止响应。 超出保存在 SMI24 中的报警阈值直到触发故障的延时设置不影响显示的值。

注意：p0604、p0605 的电机参数值（报警/故障阈值）和延迟时间 p0606 是预先定义的，不可修改。

p0606 中的延迟时间不影响操作界面上所显示的值。 也就是说：超出报警阈值以及 p0606 中设置的延迟时间届满时，信号“电机温度故障”中所显示的“最后一次超限”值不变，“超限次数”也不会增加。

p0604、p0605 的参数值（报警/故障阈值）和 p0603（延迟时间）会影响主轴的停止响应。超出故障阈值或者超出报警阈值及 p0603 中设置的延迟时间届满后，主轴停止。

当前，延迟时间不由 SMI24 给定。 p0603 的参数值显示为出厂设置 (240 s)。

报警：

- 207015 = 电机温度传感器报警
- 207016 = 电机温度传感器故障

11.3.4 附加温度传感器

停止响应

SMI24 中存有附加温度传感器的报警和故障阈值，在操作界面上显示为限值。 这两个值仅用于 SMI24 的统计分析。 超出保存在 SMI24 中的报警阈值直到触发故障的延时设置不影响显示的值。

注意：p4102[0..1] 的参数值（报警/故障阈值）和 p4103（延迟时间）是预先定义的，不可修改。

修改 p4102[0..1] 的参数值（报警/故障阈值）和 p4103（延迟时间）不影响操作界面上所显示的值。

p4102[0..1] 的参数值（报警/故障阈值）和 p4103（延迟时间）会影响主轴的停止响应。 超出故障阈值或者超出报警阈值及 p4103 中设置的延迟时间届满后，主轴停止。 为防止主轴意外停止，可对参数值进行相应的调整。

当前，延迟时间不由 SMI24 给定。 p4103 的参数值显示为出厂设置 (0 s)。 也就是说：延时段被取消。

报警：

- 207017 = 超出了附加温度的报警阈值
- 207018 = 超出了附加温度的故障阈值

11.3.5 温度柱形图

柱形图中会显示电机温度和可选的附加温度（例如：轴承温度或机壳温度）。 通过颜色来区分以下温度范围：

颜色	含义
	第一限值以内的温度范围。
	第一和第二限值之间的温度范围。
	第二限值以外的温度范围。

单个条形格的高度对应的是设置的温度测量点的温度范围。

指定给单个测量点（传感器）温度范围的时间即主轴旋转时间。

条形格下方会显示当前测量的温度。

11.3.6 转速/转矩

系统可在一条柱形图中显示特定工作小时后主轴的转速和转矩。 每种颜色指出了一种状态：

- 淡绿色：轻载
- 绿色： 额度负载
- 黄色： 过载
- 橙色：重载

单个条形格的高度对应的是基于主轴旋转时间的时间百分比（停留时间）。 在各个条形格的内部或上方是相应转速或转矩范围的累积主轴旋转时间次数。 条形格的宽度对应的是评估的转速或转矩范围。

柱形图分析

- 转速柱形图可用于分析转速对轴承和回转接头产生的负载。
- 转矩柱形图可用于分析总切削力对轴承产生的负载，但是转矩和径向力之比受到刀具直径（力臂）、铣削弧（各个切削刃产生的主切削力的叠加）和切削力系数（总切削力由主切削力、进给力和背向力组成 - 但是系统的转矩图中只显示主切削力）的影响。

11.3.7 夹紧系统

利用“夹紧系统诊断”功能，您可以：

- 快速可靠地查看主轴的夹紧状态
- 监测主轴的夹紧过程和夹紧系统
- 主轴中的传感器 S1 和 S4 用于提供夹具的信息。
- 传感器 S5 用于识别位置，例如：定义的轴旋转角位。

夹紧状态

上半部分窗口中会显示针对三个夹紧状态所设置的限值。

夹紧状态是针对给定的传感器 S1 模拟输出信号的电压范围而定义的。

“最小电压”和“最大电压”两列中的值显示了相应夹紧状态下传感器 S1 的模拟信号所对应的电压范围。

缺省转速限值“最大转速”指定了夹紧状态所允许的最大转速并由驱动对此加以限制。

信号一览

信号一览中会显示传感器、刀具夹头、设置的设定转速和有效转速限值的当前状态值。

信号	传感器	说明	值	含义
状态	S1	夹紧状态	3	已松开
			7 或 8	已夹紧，带刀具
			10	已夹紧，无刀具
活塞松开	S4	活塞询问	是	释放活塞未与轴接触。
			否	释放活塞与轴接触。
轴进入切换位置	S5	位置询问	是	例如：轴位于定义的旋转角位中。
			否	例如：轴不位于定义的旋转角位中。
设定转速		设定转速		由用户设定转速。
状态		夹紧状态	0	传感器 S1 不存在或状态值无效。
			1	状态初始化中。
			2	已松开，发出消息（故障状态）
			3	已松开
			4	夹紧，带刀具
			5	松开，带刀具
			6	松开，无刀具
			7	已夹紧，带刀具且 S4 = 0（否）
			8	已夹紧，带刀具且 S4 = 1（是）
			9	夹紧，无刀具
			10	已夹紧，无刀具
			11	已夹紧，发出消息（故障状态）

状态条形图

夹紧状态以状态条形图显示。绿色区域是已夹紧“带或无刀具”和“刀具已松开”的状态区域。箭头标记显示的是传感器 S1 的当前模拟电压。

更多信息

关于对应接口信号 (NC → PLC) 的详细信息，请参见

参数手册的“NC 变量和接口信号”一章。

步骤

1. 窗口“主轴诊断 S1”打开。
   在没有软件选件“S-Monitor（主轴监示器）”的情况下，窗口“夹紧系统 S1”打开。
2. 按下软键“夹紧系统”。
   “夹紧系统 S1”窗口打开，显示系统获得的数据。
3. 按下软键“统计”，查看主轴的夹紧时间并对夹紧系统进行诊断。
   “夹紧系统统计 S1”窗口打开，显示系统获取的数据。
   -或者-
   按下软键“返回”，回到数据一览表。

11.3.8 夹紧系统：转速限值

出于安全因素和保护主轴的目的，某些夹紧状态下只允许使用最大转速。驱动会将各个夹紧状态下的主轴转速限定为输入的转速。

夹紧状态	含义
已松开	刀具夹紧系统与释放活塞接触。注意：释放活塞会对夹紧系统施加释放力。
夹紧	刀具夹紧系统从“已松开”状态过渡到“已夹紧，带刀具”状态。
松开，从“已夹紧，带刀具”状态过渡而来	刀具夹紧系统从“已夹紧，带刀具”状态过渡到“已松开”状态。
松开，从“已夹紧，无刀具”状态过渡而来	刀具夹紧系统从“已夹紧，无刀具”状态过渡到“已松开”状态。
已夹紧，带刀具	刀具夹紧系统处于“已夹紧，带刀具”状态。
夹紧，无刀具	刀具夹紧系统从“已夹紧，带刀具”状态过渡到“已夹紧，无刀具”状态。
已夹紧，无刀具	刀具夹紧系统处于“已夹紧，无刀具”状态。

可通过软键“修改”按照您的要求修改缺省转速限值。

注意
转速过高可损坏主轴 修改缺省转速限值可导致运行状态发生变化，进而损坏主轴。

步骤

1. 窗口“主轴诊断 S1”打开。
   在没有软件选件“S-Monitor（主轴监示器）”的情况下，窗口“夹紧系统 S1”打开。
2. 按下软键“夹紧系统”。
   “夹紧系统 S1”窗口打开，显示系统获得的数据。
3. 按下软键“转速限值”，根据夹紧状态调整转速限值。
   窗口“转速限值 S1”打开。
4. 按下软键“修改”，根据夹紧状态输入所需的转速限值。
   按下“OK”。

11.3.9 夹紧系统：诊断统计

该窗口中会显示夹紧过程和转速超限的统计数据。

统计数据	含义
正确的夹紧循环次数	累积的正确夹紧循环（带或无刀具）次数。 正确的夹紧循环是：夹紧状态“已松开” [3] → 夹紧状态“已夹紧” [7、8 或 10] → 夹紧状态“已松开” [3]。 正确的夹紧循环是指夹紧过程和之后的松开过程是通过相应的夹紧状态（无“状态回退”）进行的。 从一个夹紧状态开始，下一个可达到的夹紧状态不可以是前一个夹紧状态。 此时不存在时间关系。
夹紧时间（无刀具）	夹紧状态“刀具已松开”和“已夹紧，无刀具”之间的夹紧过程内的时间。
• 在公差范围内	设定时间内累积的夹紧过程（无刀具）次数。
• 在公差范围外	设定时间外累积的夹紧过程（无刀具）次数。
• 设定	夹紧过程（无刀具）时间的出厂设定。
• 最新	测出的上一次执行的夹紧过程（无刀具）的时间。
已夹紧（无刀具）	夹紧状态“已夹紧，无刀具”期间的诊断
• 转速超限	累积的主轴转速大于出厂时设置的限值的事件次数。

统计数据	含义
• 转速阈值	夹紧状态“已夹紧，无刀具”下的主轴转速限值（出厂值）。 通过诊断一览“夹紧系统 S1” → “转速限值” → “修改”修改该转速限值不会影响此处给定的值。
夹紧/松开过程错误（整体）	累积的非连续运行夹紧状态：夹紧状态“已松开” [3] → 夹紧状态“已夹紧” [7、8 或 10] → 夹紧状态“已松开” [3] 的夹紧过程次数。 错误的夹紧循环是指：从一个夹紧状态开始，下一个达到的夹紧状态又是前一个夹紧状态（“状态回退”）。

11.3.10 夹紧系统：夹紧时间统计

该窗口中会显示测出的夹紧时间的统计数据。

统计数据	含义
上一次测出的夹紧时间	测出的上一次执行的夹紧过程的时间。 夹紧过程定义：夹紧状态“已松开” [3] → 夹紧状态“已夹紧，带刀具” [7 或 8]。夹紧状态 3 = 已松开；夹紧状态 7 = 已夹紧，带刀具及 S4 = 0（否）；夹紧状态 8 = 已夹紧，带刀具及 S4 = 1（是）
参考夹紧时间（带刀具）	在主轴调试阶段，前 100 个夹紧过程由 SMI24 测定并确定最小值、最大值和平均值。
• 最小值	从前 100 个夹紧过程中测出的最小夹紧时间。
• 最大值	从前 100 个夹紧过程中测出的最大夹紧时间。
• 平均值	从前 100 个夹紧过程中测出的平均夹紧时间。
平均夹紧时间（带刀具）	系统会以前 100 个测出的夹紧时间为参考计算其余测出的夹紧时间的平均值。
• 24 小时	最后 24 小时内测出的夹紧循环的平均夹紧时间。
• 10 天	最后 10 天内测出的夹紧循环的平均夹紧时间。
• 100 天	最后 100 天内测出的夹紧循环的平均夹紧时间。

如果在调试阶段重启了 SMI24，系统会删除迄今为止所测出的值。 前 100 个夹紧过程的分子被置零，可以重新测定夹紧过程（最多 100 个）的参考时间。 调试阶段（前 100 个夹紧过程）结束之后，参考夹紧时间值被“冻结”。 重启 SMI24 后，值仍被保留且不可修改。

11.3.11 查看主轴的物流数据

前提条件

只有“服务”访问级别才可查看主轴数据。

主轴数据

通过 DRIVE-CLiQ 可以查看主轴制造商的下列主轴数据：

- 厂商名称
- 图纸号
- 序列号
- 生产日期
- 服务数据和服务信息(1...4)：
  服务信息即 WEISS 主轴技术有限责任公司服务报告的过程编号。

驱动系统 12

12.1 驱动调试

步骤

在完成 PLC 调试后，可通过 SINUMERIK Operate 操作界面对 SINAMICS 驱动进行调试。



1. 选择操作区域“调试”。



2. 按下软键“驱动系统”。
   窗口“驱动系统一览”打开。
   此时会弹出一条消息，提示是否已经调试了驱动系统。

更多信息

SINAMICS 驱动的调试步骤请参见

CNC 调试手册：NCK、PLC、驱动。

配置报警

13

13.1 通过操作界面创建报警文本和信息文本

您可以在 HMI 的操作面板上，从零件程序创建和编辑自定义的报警文本和消息文本。报警和消息文本分别按照其编号保存在不同的文本文件中。

存储位置

文本文件在以下目录中创建：
/**oem**/sinumerik/hmi/lng

步骤



1. 选择操作区域“调试”。



2. 按下软键“HMI”。



3. 按下软键“报警文本”。
   窗口“选择文件”打开，向您提供用户专用的文本文件。



4. 选中所需文件并按下软键“确认”。

5. 在“编号”列中输入所需的报警编号。
   在“报警文本”列中输入所需的报警文本。
   在“颜色”列中选择所需的字体颜色。
   在“弹出窗口”列中选择“是”或“否”，确定是否在需要确认的窗口中显示报警文本。
   - 或 / 与 -



按下软键“插入行”。
将在光标的上方插入一个新行。
- 或者



按下软键“删除行”，删除选定的行。



6. 按下软键“搜索”。

窗口“搜索”打开。

在“文本”栏中输入所需的报警文本或者报警编号。

如果您需要区分输入文本的大小写，请勾选复选框“区分大小写”。

- 或者



按下软键“搜索+替换”。

“搜索与替换”窗口打开。

在“文本”栏中输入待查找的关键字。在“替换为”栏中输入待替换的关键字。



7. 将光标定位到“方向”栏中。通过按键 <SELECT> 选择搜索方向（向前，向后）。



8. 按下软键“确认”，开始进行搜索。



可按下软键“取消”中断操作。



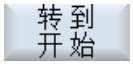
9. 为报警输入帮助文本后，按下软键“确认”。

会出现消息“已经保存并转换了报警文本。”。

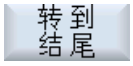
重新启动 SINUMERIK Operate

重启 SINUMERIK Operate 后，属性“颜色”和“弹出窗口”才生效。但修改的报警文本立即生效，无需重启系统。

其他搜索方法



光标跳至所选报警文本文件的第一个条目上。



光标跳至所选报警文本文件的最后一个条目上。

创建搜索目录

为了在现有目录 /lng 以外的其他子目录中搜索报警文件，您可以

1. 将文件“systemconfiguration.ini”保存在以下目录：
   /**oem**/sinumerik/hmi/cfg
2. 在 /hmi, z. B. /oem_dir 下任意创建一个子目录。
3. 在配置文件 “systemconfiguration.ini” 中创建以下段：
   ```
   [oem_dirs]
   OEM_3=oem_dir
   ```
4. 重新启动 SINUMERIK Operate。
   现在系统在搜索时会考虑子目录。

外语报警和消息文本

若要创建其它语种的报警和消息文本，必须切换至所需的语种。所有文本文件均自动包含一个与所设的界面语言相对应的语种标识。比如，在英语界面中创建的报警文本则含有语种标识 "eng"。

参见

报警序号范围 (页 226)

编制 OEM 子目录 (页 636)

13.2 通过报警文本文件创建报警文本和信息文本

您可以通过报警文本文件创建和修改用户专用的报警文本或消息文本。

报警文本以 SINUMERIK Operate 的标准格式（“.ts”格式）创建。“.ts” 是基于 XML 的格式。

不仅可以通过 SINUMERIK Operate，还可以通过外部 PC 创建和编辑这些文件。

说明

如果在 PC 上编辑报警文本文件，要使用支持 UTF-8 编码的编辑器。

说明

建议中文的报警文本仅能使用符号 GB2312 转换。
韩语中不支持中文音节符号。

创建和编辑报警文本

- 创建自定义的报警文本
- 创建带有索引的报警参数的文本
- 创建零件程序消息文本
- 修改报警属性
- 替换缺省报警文本
- 转换报警文本

13.2.1 创建自身报警文本

创建报警文本文件

1. 可从如下目录中复制模板文件“oem_alarms_deu.ts”： /**siemens**/sinumerik/hmi/template/lng.
2. 在目录/**oem**/sinumerik/hmi/lng 或者 /**user**/sinumerik/hmi/lng 中保存或创建文件。且分别为支持的各国语言均创建一个单独的文件。
3. 为文件命名，例如 "my_alarms_deu.ts"。 除语种标识和后缀名外，可自由选择文件名称。文件名的结尾始终是相应语种的语种标识，后缀名始终为“.ts”。 例如：“my_alarms_deu.ts”为德语文件；“my_alarms_eng.ts”为英语文件。

创建报警文本

1. 打开文件“my_alarms_deu.ts”。
2. 必须在此文件中为每个报警文本插入一个单独区域，区域由标签<message>标出。
3. 标签 <name> 包含有语境名称。 一般情况下最好不要修改默认名称“slaeconv”。
4. 标签 <source> 里包含有报警编号和报警源(SourceURL)的名称。 示例：“700000”是报警编号，“/PLC/PMC”为报警源的名称。 潜在的报警源参见章节报警序号范围 (页 226)中的表格。
5. 标签 <translation> 包含有报警文本。

示例：

```
<!DOCTYPE TS>
<TS>
    <context>
        <name>slaeconv</name>
```

```
        <message>
            <source>700000/PLC/PMC</source>
            <translation>第一个 OEM 报警文本</translation>
        </message>
        <message>
            <source>700001/PLC/PMC</source>
            <translation>第二个 OEM 报警文本</translation>
        </message>
    </context>
</TS>
```

注册报警文本文件

为了在程序运行时系统能识别报警文本，还必须注册报警文本文件：

1. 可从如下目录中复制模板配置文件"oem_slaesvcadapconf.xml"： /**siemens**/sinumerik/hmi/template/cfg.
2. 将文件保存到目录 /**oem**/sinumerik/hmi/cfg 或 /**user**/sinumerik/hmi/cfg 中。
3. 将此文件命名为"slaesvcadapconf.xml"。
4. 打开文件，在标签<BaseNames>中输入报警文本文件的名称（不含语种标识和后缀名），例如：输入"my_alarms"，而不是"value"。
   比如：您可以以名称"OEM_BaseName_01"注册报警文本文件。

   说明

   原则上您可以在标签<BaseNames>中自由命名。但在系统中该名称应是唯一的，不能和西门子使用的名称冲突。

   名称"Siemens_BaseName_01"预留给西门子使用。

   还可注册多个报警文本文件。您可以使用诸如"OEM_BaseName_02"或"OEM_BaseName_03"的名称注册。

示例：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- Solutionline 报警&事件适配器的配置 -->
<CONFIGURATION>
    <AlarmTexts>
        <BaseNames>
```

```
            <OEM_BaseName_01 type="QString" value="my_alarms"/>
        </BaseNames>
    </AlarmTexts>
    ...
</CONFIGURATION>
```

重新启动 SINUMERIK Operate

为了在程序运行时能显示报警文本，必须先将其转换至二进制格式。 此转换仅在启动时进行。

为此重新启动 SINUMERIK Operate。 会在自有目录（其中包含 .ts 文件）中创建一个后缀名为“.qm”的同名文件。

转换的结果将写入到文件“alarmtext_conversion.log”或“oem_text_conversion.log”中。 此外在转换中出现的错误，例如参数文件中的句法错误，也被写入文件。

此文件保存在目录“/user/sinumerik/hmi/log”中。

说明

只有当 .ts 文件早于所属 .qm 文件时，才进行转换。

13.2.2 创建指示报警参数的文本

如果报警文本中报警参数占位符后的尖括号中给出了一个字符串，即所谓的“索引标识符”（例如：“%1<OEM>”），则不会将参数本身插入报警文本中，而是从另一个报警文本文件（即：索引文本文件）中插入一个文本。其中，报警参数数值作为在索引文件中选择此文本的索引值。 索引文本也可包含参数，甚至包含索引参数。

说明

字符串“索引标识符”的解释

- "<" ≙ 左尖括号"<"
- "<" ≙ 右尖括号"<"

创建索引文本文件

1. 可从如下目录中复制模板索引文本文件“oem_indexparams_deu.ts”： /**siemens**/sinumerik/hmi/template/lng.
2. 在目录/**oem**/sinumerik/hmi/lng 或者 /**user**/sinumerik/hmi/lng 中保存或创建文件。且分别为支持的各国语言均创建一个单独的文件。
3. 为文件命名，例如“my_indextexts_deu.ts”。 除语种标识和后缀名外，可自由选择文件名称。文件名的结尾始终是相应语种的语种标识，后缀名始终为“.ts”。例如：“my_indextexts_deu.ts”为德语文件；“my_indextexts_eng.ts”为英语文件。

创建索引文本

1. 打开文件 “my_indextexts_deu.ts”。
2. 在标签 <name> 中输入名称，例如：“my_context”。
3. 对于每个报警文本都必须插入一个单独的区域，区域通过标签 <message> 和 </message> 标出。
4. 标签 <source> 包含报警参数数值，比如数值“1”和“2”。
5. 标签 <translation> 包含索引文本，当所涉报警参数拥有 <source> 和 </source> 之间的数值时，显示此文本。

示例

```
<!DOCTYPE TS>
<TS>
    <context>
       <name>my_context</name>
       <message>
            <source>1</source>
            <translation>第一个 OEM 参数文本</translation
       </message>
       <message>
            <source>2</source>
            <translation>第二个 OEM 参数文本</translation
       </message>
    </context>
</TS>
```

注册索引文本文件

为了在程序运行时系统能识别索引文本，必须注册索引文本文件。 在文件“slaesvcadapconf.xml”中完成注册。

1. 打开目录 /**oem**/sinumerik/hmi/cfg 或者 /**user**/sinumerik/hmi/cfg 下为自身报警文本而创建的文件 "slaesvcadapconf.xml"。
2. 删除行“<!-- 注释开头”和“注释结尾 -->”。
3. 输入所谓的索引标识符，例如：<Identifier type="QString" value="OEM"/>。 索引标识符始终在报警文本参数说明之后的尖括号内，例如： "%1<OEM>"。
4. 输入索引文本文件的 BaseName， 例如：
<BaseName type="QString" value="my_indextexts"/>。
5. 输入所选择的语境名称， 例如：
<ContextName type="QString" value="my_context"/>。
6. 还可使用多个不同的索引。 然后必须在标签 <IndexTexts> 和 </IndexTexts> 之间为每个索引创建单独的段落。 这些段落的标签为诸如
<OEM_IndexText_01>、<OEM_IndexText_02> 和 <OEM_IndexText_03> 等标签。
标签 <IndexText_01> 至 <IndexText_99> 预留给西门子使用。

示例

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!- Configuration of the Solutionline Alarm & Event Service
Adapter ->
<CONFIGURATION>
...
    <!-- The following narrated part of the configuration is used
    only when 'Indexparameters' (e.g. '%1&lt;OEM&gt;') are used
    within OEM alarm texts. In this case the value of the
    parameter is used as an index into an additional text list to
    reference another text which is placed into the alarm text
    instead of the original parameter value. -->
    <IndexTexts>
       <OEM_IndexText_01>
          <Identifier type="QString" value="OEM"/>
          <BaseName type="QString" value="my_indextexts"/>
          <ContextName type="QString" value="my_context"/>
          <MetaTextID type="QString" value="%ParamValue%"/>
       </OEM_IndexText_01>
```

```
    </IndexTexts>
</CONFIGURATION>
```

重新启动 SINUMERIK Operate

为了在程序运行时能显示报警文本，必须先将其转换至二进制格式。 此转换仅在启动时进行。

为此重新启动 SINUMERIK Operate。 会在自有目录（其中包含 .ts 文件）中创建一个后缀名为“.qm”的同名文件。

转换的结果将写入到文件“alarmtext_conversion.log”或“oem_text_conversion.log”中。此外在转换中出现的错误，例如参数文件中的句法错误，也被写入文件。

此文件保存在目录 /user/sinumerik/hmi/log 中。

说明

只有当 .ts 文件早于所属 .qm 文件时，才进行转换。

13.2.3 创建零件程序信息文本

来自零件程序的消息文本

您需要在零件程序中编写指令 MSG，以便从中输出消息文本。 消息文本通过“$”字符后的编号标记，比如：MSG("$4711")。 您可以为每个 NC 通道配置单独的消息文本。

创建消息文本

步骤：

1. 复制如下目录中的消息文本模板文件“oem_pp_messages_deu.ts”： /**siemens**/sinumerik/hmi/template/lng.
2. 在目录/**oem**/sinumerik/hmi/lng 或者 /**user**/sinumerik/hmi/lng 中保存或创建文件。且分别为支持的各国语言均创建一个单独的文件。
3. 为文件命名，例如“my_msgs_deu.ts”。 除语种标识和后缀名外，可自由选择文件名称。 文件名的结尾始终是相应语种的语种标识，后缀名始终为“.ts”。 例如：“my_msgs_deu.ts”为德语文件；“my_msgs_eng.ts”为英语文件。
4. 打开文件“my_msgs_deu.ts”，在标签<name>中输入字符串“partprogmsg01”。 这是所有通道的零件程序消息文本的默认设置。
5. 每条消息文本必须占用单独的区域，区域通过标签 <message> 和 </message> 标出。

6. 标签 <source> 中包含了零件程序中 MSG 指令的编号。
7. 标签 <translation> 包含有消息文本。

示例

```
<!DOCTYPE TS>
<TS>
   <context>
     <name>partprogmsg01</name>
     <message>
        <source>4711</source>
        <translation> 零件程序消息编号 4711</translation
     </message>
   </context>
</TS>
```

注册消息文本文件

为了在程序运行时系统能识别消息文本，您必须首先注册消息文本文件：

1. 可从如下目录中复制模板配置文件"oem_slaesvcadapconf.xml"： /**siemens**/sinumerik/hmi/template/cfg.
2. 将文件保存到目录 /**oem**/sinumerik/hmi/cfg 或 /**user**/sinumerik/hmi/cfg 中。
3. 修改"slaesvcadapconf.xml"中的名称。
4. 打开文件，在标签<BaseNames>中输入消息文本文件的名称（不含语种标识和后缀名），例如：输入"my_msgs"，而不是"value"。 比如：您可以以名称"OEM_BaseName_01"注册消息文本文件。

   说明

   原则上您可以在标签<BaseNames>中自由命名。 但在系统中该名称应是唯一的，不能和西门子使用的名称冲突。

   名称"Siemens_BaseName_01"预留给西门子使用。

   还可注册多个消息文本文件。 您可以使用诸如"OEM_BaseName_02"或"OEM_BaseName_03"的名称注册。

示例

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- Solutionline 报警&事件适配器的配置 -->
<CONFIGURATION>
    <AlarmTexts>
       <BaseNames>
          <OEM_BaseName_01 type="QString" value="my_msgs"/>
       </BaseNames>
    </AlarmTexts>
    ...
</CONFIGURATION>
```

多个通道的消息文本

您可以在不同的 NC 通道中为同一个消息号（比如“消息 4711）指定不同的消息文本，即：系统会根据零件程序所处的通道来输出不同的消息文本。

1. 在编辑器中打开文件“my_msgs_deu.ts”。
2. 更改“partprogmsgXY”中“partprogmsg01”的标签 <name> 和 </name> 之间的文本，其中“XY”可以用相应的通道编号替代，比如通道 2 的“partprogmsg02”。
3. 仅打开配置文件“slaesvcconf.xml”。
4. 在标签<Connections>中输入相应的 NC 通道。 说明 NC 通道时，只能使用如下表格中的 XML 标签：

NC 通道	XML 标签	ContextName
1	PartprogramMessageChannel_01	partprogmsg01
2	PartprogramMessageChannel_02	partprogmsg02
3	PartprogramMessageChannel_03	partprogmsg03
4	PartprogramMessageChannel_04	partprogmsg04
5	PartprogramMessageChannel_05	partprogmsg05
6	PartprogramMessageChannel_06	partprogmsg06
7	PartprogramMessageChannel_07	partprogmsg07
8	PartprogramMessageChannel_08	partprogmsg08
9	PartprogramMessageChannel_09	partprogmsg09
10	PartprogramMessageChannel_10	Partprogmsg10

示例

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- Solutionline 报警&事件适配器的配置 -->
<CONFIGURATION>
    <Connections>
       <PartprogramMessageChannel_02>
          <ContextName type="QString" value="partprogmsg02"/>
       </PartprogramMessageChannel_02>
    </Connections>
</CONFIGURATION>
```

重新启动 SINUMERIK Operate

为了在程序运行时能显示报警文本，必须先将其转换至二进制格式。此转换仅在启动时进行。

为此重新启动 SINUMERIK Operate。会在自有目录（其中包含 .ts 文件）中创建一个后缀名为 ".qm"的同名文件。

转换的结果将写入到文件"alarmtext_conversion.log"或"oem_text_conversion.log"中。此外在转换中出现的错误，例如参数文件中的句法错误，也被写入文件。

此文件保存在目录 /user/sinumerik/hmi/log 中。

说明

只有当 .ts 文件早于所属 .qm 文件时，才进行转换。

13.2.4 修改报警属性

报警的颜色类型

可以分别修改消息栏中显示的彩色报警和消息。

每个报警号和报警源可以更改成下列颜色类型：

- 报警/消息文本的字体颜色
- 报警/消息文本的背景颜色

- 报警号的字体颜色
- 报警号的背景颜色

创建报警属性文件

步骤：

1. 可从如下目录中复制模板报警配置文件“oem_slaedatabase.xml”： /**siemens**/sinumerik/hmi/template/cfg/。
2. 将文件保存到目录 /**oem**/sinumerik/hmi/cfg/ 或 /**user**/sinumerik/hmi/cfg/ 中。
3. 为文件命名，例如“muster_slaedatabase.xml”。
   可以自由选择文件名，但只能为小写字母。

定义报警颜色

步骤：

1. 打开文件“muster_slaedatabase.xml”。
2. 首先确定您要更改的报警颜色。
   在区域 <Attributes> 中为需要更改的各种颜色都分别创建一个段落 <Attribute>。
3. 在标签 <Attribute AttrName=...> 中输入要更改报警颜色的属性名称。 可以更改下列颜色：
   - 报警/消息文本的字体颜色 = TEXTCOLOR
   - 报警/消息文本的背景颜色 = TEXTBACKGROUNDCOLOR
   - 报警编号的字体颜色 = NUMBERCOLOR
   - 报警编号的背景颜色 = NUMBERBACKGROUNDCOLOR
4. 区域<Sources> 中针对要更改报警颜色的报警定义了报警源，比如报警源“/HMI”和“/PLC/PMC”。

   说明

   如果想继续添加报警源，需要注意将报警号分配给正确的报警源。

   允许的 SourceID 和 SourceURL 参见章节报警序号范围 (页 226)中的表格。

5. 在标签 <Alarms> 中为每个独立的报警或每个报警号范围设定一个自己特有的子域。

6. 在标签 <Alarm AlarmID= "..." > 中输入报警号，或者在标签 <Range FromAlarmID="..." ToAlarmID= "..."> 中输入报警号范围。
7. 在下列标签中输入所需的颜色数值：
   <TEXTCOLOR>
   <TEXTBACKGROUNDCOLOR>
   <NUMBERCOCLOR>
   <NUMBERBACKCOLOR>
   通过 RGB 值来定义颜色值的属性：
   - RGB 值始终使用符号“#”开头。
   - 单个颜色值 R、G 和 B 用两位数的十六进制值表示。句法： “RRGGBB”，比如： “FF9A00”。

示例

在指定各个报警号范围的颜色时，要注意 xml 配置文件中的顺序：

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE SlAeAlarmAttributes>
<SlAeAlarmAttributes Version="01.00.00.00">
<Types>
  <Type TypeName="Condition" TypeID="32">
   <Category Version="1.0" CatID="1">
      <CatDescr>840D sl 的报警。</CatDescr>
      <Attributes>
       <Attribute AttrName="TEXTCOLOR" AttrID="5003"
       AttrDataType="10">
         <AttrDescr>
           在面板标题栏显示报警所使用的文本颜色。
         </AttrDescr>
       </Attribute>
       <Attribute AttrName="TEXTBACKGROUNDCOLOR" AttrID="5004"
       AttrDataType="10">
         <AttrDescr>
           在面板标题栏显示报警所使用的背景颜色。
         </AttrDescr>
       </Attribute>
```

```
        <Attribute AttrName="NUMBERCOLOR" AttrID="5005"
        AttrDataType="10">
          <AttrDescr>
            在面板标题栏显示报警所使用的文本颜色。
          </AttrDescr>
        </Attribute>
        <Attribute AttrName=" NUMBERBACKGROUNDCOLOR "
        AttrID="5006" AttrDataType="10">
          <AttrDescr> 在面板标题栏显示报警所使用的背景颜色。
          </AttrDescr>
        </Attribute>
       </Attributes>
     </Category>
   </Type>
</Types>
<Sources>
   <Source> CatLink="1" SourceID="10000" SourceURL="/HMI">
     <Alarms>
       <Alarm AlarmID="130000">
         <TEXTCOLOR>#000000</TEXTCOLOR>
         <TEXTBACKGROUNDCOLOR>#FFFFFF</TEXTBACKGROUNDCOLOR>
         <NUMBERCOLOR>#FFFFFF</NUMBERCOLOR>
         <NUMBERBACKGROUNDCOLOR>#000000</NUMBERBACKGROUNDCOLOR
       </Alarm>
     </Alarms>
   </Source>
<Source> CatLink="1" SourceID="51" SourceURL="/PLC/PMC">
     <Alarms>
       <Range FromAlarmID="700100" ToAlarmID="700199">
         <TEXTCOLOR>#000000</TEXTCOLOR>
         <NUMBERCOLOR>#00FF00</NUMBERCOLOR>
       </Range>
       <Alarm AlarmID="700000">
```

```
            <TEXTCOLOR>#000000</TEXTCOLOR>
            <TEXTBACKGROUNDCOLOR>#FFFFFF</TEXTBACKGROUNDCOLOR>
            <NUMBERCOLOR>#FFFFFF</NUMBERCOLOR>
            <NUMBERBACKGROUNDCOLOR>#000000</NUMBERBACKGROUNDCOLOR
          </Alarm>
        </Alarms>
    </Source>
</Sources>
</SlAeAlarmAttributes>
```

注册报警属性文件

如需指定程序运行期间发出的报警的颜色，需要首先注册含颜色定义的报警属性文件：

1. 可从如下目录中复制模板配置文件“oem_slaesvcconf.xml”： /**siemens**/sinumerik/hmi/template/cfg.
2. 将文件保存到目录 /**oem**/sinumerik/hmi/cfg 或 /**user**/sinumerik/hmi/cfg 中。
3. 命名此文件为“slaesvcconf.xml”
4. 打开文件，在标签<DataBases>中输入报警属性文件的名称（不含语种标识和后缀名），例如：输入“muster_slaedatabase”，而不是“value”。 比如：您可以以名称“OEM_DataBase_01”注册报警属性文件。

   说明

   原则上您可以在标签<DataBases>中自由命名。 但在系统中该名称应是唯一的，不能和西门子使用的名称冲突。

   名称“Siemens_DataBase_01”预留给西门子使用。

   若需要注册更多的报警属性文件，即将它们将加入配置文件中，可使用诸如“OEM_DataBase_02”和“OEM_DataBase_03”等名称。

重新启动 SINUMERIK Operate

为了在程序运行时报警颜色能生效，必须先将其转换至二进制格式。 此转换仅在启动时进行。

为此重新启动 SINUMERIK Operate。 这时会在同一个目录中（此目录中含有报警属性文本）创建一个同名文件、其文件后缀名为“.hmi”，例如“muster_slaedatabase.hmi”。

转换的结果将写入到文件“oem_ae_database_conversion.log”中。 此文件保存在目录“/user/sinumerik/hmi/log”中。

说明

只有当 XML 文件早于所属 hmi 文件时，才进行转换。

13.2.5 替换标准报警文本

您可以将 SINUMERIK Operate 的缺省报警文本替换为自定义的报警文本。

创建报警文本文件

1. 可从如下目录中复制模板报警文本文件“oem_alarms_deu.ts”： /**siemens**/sinumerik/hmi/template/lng.
2. 在目录/**oem**/sinumerik/hmi/lng 或者 /**user**/sinumerik/hmi/lng 中保存或创建文件。且分别为支持的各国语言均创建一个单独的文件。
3. 为该文件指定一个用户专用的名称，例如：“my_nck_alarms_deu.ts”。 除语种标识和后缀名外，可自由选择文件名称。
   文件名的结尾始终是相应语种的语种标识，后缀名始终为“.ts”。 例如：
   “my_nck_alarms_deu.ts”为德语文件；“my_nck_alarms_eng.ts”为英语文件。

创建报警文本

1. 打开文件“my_nck_alarms_deu.ts”。
2. 每条报警文本必须占用单独的区域，区域通过标签 <message> 标出。
3. 标签 <name> 包含有语境名称。 修改名称，注意名称必须是系统中唯一的名称，例如：myNckAlarms。

   说明

   标签<name> 的名称原则上可以自由选择。 但是这个名称不能和西门子使用的名称“slaeconv”相冲突。

4. 标签 <source> 包含了缺省报警的报警号，比如：10000。
5. 标签<translation>包含了自定义的报警文本，比如“NCK 报警 10.000 的 OEM 示例报警文本”。

示例

```
<!DOCTYPE TS>
<TS>
```

```
    <context>
      <name>myNckAlarms</name>
      <message>
        <source>10000/NCK</source>
        <translation> NCK 报警 10.000 的 OEM 示例报警文本 </translation>
      </message>
    </context>
  <TS>
```

注册报警文本文件

为了在程序运行时系统能识别报警文本，还必须注册报警文本文件：

1. 可从如下目录中复制模板配置文件"oem_slaesvcadapconf.xml"： /**siemens**/sinumerik/hmi/template/cfg.
2. 将文件保存到目录 /**oem**/sinumerik/hmi/cfg 或 /**user**/sinumerik/hmi/cfg 中。
3. 将此文件命名为"slaesvcadapconf.xml"。
4. 打开文件，在标签<BaseNames>中输入报警文本文件的名称（不含语种标识和后缀名），例如：输入"my_nck_alarms"，而不是"value"。 比如：您可以以名称"Siemens_BaseName_01"注册报警文本文件。

   说明

   原则上您可以在标签<BaseNames>中自由命名。 但在系统中该名称应是唯一的，不能和西门子使用的名称冲突。

   名称"Siemens_BaseName_01"预留给西门子使用。

   还可注册多个报警文本文件。 您可以使用诸如"OEM_BaseName_02"或"OEM_BaseName_03"的名称注册。

示例：

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- Solutionline 报警&事件适配器的配置 -->
<CONFIGURATION>
  <AlarmTexts>
    <BaseNames>
      <OEM_BaseName_01 type="QString" value="my_nck_alarms"/>
  </BaseNames>
```

```
  </AlarmTexts>
</CONFIGURATION>
```

替换缺省报警文本

为了在程序运行时您的报警文本能够替换原始报警文本，必须事先确定您的报警文本将替换哪个缺省报警文本。

1. 在目录/**oem**/sinumerik/hmi/cfg 或者/**user**/sinumerik/hmi/cfg 中创建新的报警属性，见修改报警属性 (页 216)。 可以自由命名此文件，比如将其命名为“my_nck_alarms_db.xml”。
2. 打开文件。
3. 此文件始终由两个部分组成： <Types> 和 <Sources>。
4. 标签 <Sources> 中包含了报警源如 “/NCK”和报警文本的参照关系。
5. 将标签<context> <name> 的内容与报警文本文件中标签 <message> <source> 的内容关联起来。
   - <context> <name> = 您规定的语境名称
   - <message> <source> = 其文本应被替换的报警编号。
     例如：在报警属性文件中输入关联： <MSGTEXT>myNckAlarms I 10000</MSGTEXT>
     注意：应用分隔符“I”分开语境名称和报警编号。
6. 需要建立另一个 NCK 报警时，请复制 <Alarms>到 </Alarms>之间的内容，然后修改报警的编号。
7. 如果需要建立其他编号范围，例如 PLC 报警内的另一个报警，请复制<Sources>到</Sources>之间的内容，然后修改 SourceID、SourceURL 以及报警号。
   对应的 SourceID 和 SourceURL 请参见章节报警序号范围 (页 226)中的表格。

说明

必须为每个需要替换文本的缺省报警建立单独的关联。

示例

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE SlAeAlarmAttributes>
<SlAeAlarmAttributes Version="01.00.00.00">
<Types>
  <Type TypeName="Condition" TypeID="32">
    <Category Version="1.0" CatID="1">
      <CatDescr>Alarms of the Sinumerik 810/840 D(i).</CatDescr>
```

```
      <Attributes>
      </Attributes>
    </Category>
  </Type>
</Types>
<Sources>
  <Source CatLink="1" SourceID="0" SourceURL="/NCK">
    <Alarms>
      <Alarm AlarmID="10000">
        <MSGTEXT>myNckAlarms|10000/NCK</MSGTEXT>
      </Alarm>
    </Alarms>
  </Source>
</Sources>
</SlAeAlarmAttributes>
```

注册报警属性文件

为了在程序运行时替换报警文本替换能生效，必须先注册此报警属性文件：

1. 可从如下目录中复制模板配置文件"oem_slaesvcconf.xml"： /**siemens**/sinumerik/hmi/template/cfg.
2. 将文件保存到目录 /**oem**/sinumerik/hmi/cfg 或 /**user**/sinumerik/hmi/cfg 中。
3. 修改"slaesvcconf.xml"中的名称。
4. 打开文件，并在标签 <DataBases> 中输入报警属性文件的名称，比如：<OEM_DataBase_01 type="QString" value="my_nck_alarms_db"/>。

说明

原则上您可以在标签<DataBases>中自由命名。 但在系统中该名称应是唯一的，不能和西门子使用的名称冲突。

名称<Siemens_DataBase_01>预留给西门子使用。

若需要注册更多的报警属性文件，即将它们将加入配置文件中，可使用诸如<OEM_DataBase_02>和<OEM_DataBase_03>等名称。

重新启动 SINUMERIK Operate

为了在程序运行时能显示报警文本，必须先将含有报警文本的文件和属性文件转换至二进制格式。 此转换仅在启动时进行。

为此重新启动 SINUMERIK Operate。 会在自有目录（其中包含 .ts 文件或者报警属性文件）中创建一个带后缀名".qm"和".hmi"的同名文件。

转换的结果将写入到文件"alarmtext_conversion.log"或"oem_text_conversion.log"中。此外在转换中出现的错误，例如参数文件中的句法错误，也被写入文件。

此文件保存在目录"/user/sinumerik/hmi/log"中。

说明

只有当 .ts 文件早于所属 .qm 文件时，才进行转换。

13.2.6 报警序号范围

报警编号范围

编号范围	说明		SourceID	SourceURL
000.000 – 009.999	一般报警		0 (缺省 NCU) 100（第 1 个 NCU） ... 9999（第 99 个 NCU）	/NCK //<配置专用名称>//NCK
010.000 – 019.999	通道报警			
020.000 – 029.999	进给轴/主轴报警			
030.000 – 039.999	功能报警	常规		
040.000 – 059.999		预留		
060.000 – 064.999		西门子循环报警		
065.000 – 069.999		用户循环报警		
070.000 – 079.999		编译循环 制造商和 OEM		
080.000 – 084.999		西门子循环的消息文本		
085.000 – 089.999		用户循环消息文本		
090.000 – 099.999		预留		
100.000 – 129.999	系统		10.000	/HMI
130.000 – 139.999	OEM			
140.000 – 199.999	预留			

编号范围	说明		SourceID	SourceURL
200.000 – 299.999	SINAMICS 驱动		0	/NCK
300.000 – 399.999	驱动报警和 I/O 设备报警			
400.000 – 499.999	一般报警		51	/PLC/PMC
500.000 – 599.999	通道报警			
600.000 – 699.000	进给轴/主轴报警			
700.000 – 799.999	用户范围			
800.000 – 899.999	流程 / 图表			
810.000 – 810.009	系统故障显示消息		50 150（第 1 个 NCU）	/PLC/DiagBuffer //<配置专用名称>/PLC/DiagBuffer
900.001 – 965.999	HMI PRO sl Runtime			
966.000 – 999.999	预留			

SourceIDs 1 ... 10

SourceIDs 1 ... 10 有以下关联：

SourceID	SourceURL
1	/NCK/Channel#1/Partprogram
2	/NCK/Channel#2/Partprogram
3	/NCK/Channel#3/Partprogram
4	/NCK/Channel#4/Partprogram
5	/NCK/Channel#5/Partprogram
6	/NCK/Channel#6/Partprogram

SourceID	SourceURL
7	/NCK/Channel#7/Partprogram
8	/NCK/Channel#8/Partprogram
9	/NCK/Channel#9/Partprogram
10	/NCK/Channel#10/Partprogram

13.2.7 报警文本的参数说明

报警文本可以包含报警参数（伴随值），它们通常能够更详细地说明报警原因。 这些参数一般是数值，在发出报警信号时，和其他报警数据一起同时由报警源发出。

通过语言专用报警文本中的占位符（参数说明）来规定参数在报警文本中的位置，例如：“通道 **%1** 轴 **%2** 在机床数据 **%3** 中对多个通道进行了定义”。

当出现报警时，相应的参数将会替换文本，例如：“通道 **5** 轴 **A3** 在机床数据 **4711** 中对多个通道进行了定义。

标准参数说明

下表列出了标准参数说明：

表格 13-1　标准参数说明

参数分隔符	说明
%1	报警源数据的第一个参数。
%2	报警源数据的第二个参数。
%3	报警源数据的第三个参数。
%4	报警源数据的第四个参数。
%5	报警源数据的第五个参数。 仅 NCK 报警： 报警源数据的第四个参数的第一部分（直至分隔符）。
%7	报警源数据的第七个参数 仅 NCK 报警： 报警源数据的第四个参数的第三部分（第 2 和第 3 分隔符之间）。

参数分隔符	说明
%8	报警源数据的第八个参数 仅 NCK 报警： 报警源数据的第四个参数的第四部分（第 3 和第 4 分隔符之间）
%9	报警源数据的第九个参数。
%0	报警源数据的第十个参数。
%Z	同 %1，在 S7-HiGraph 报警时显示为图表的步长号。
%K	专用于 PLC 报警： 十进制报警 ID： 1**2**3456 的第二位，表示通道号，0 表示通道 10。
%A	专用于 PLC 报警： 十进制报警 ID 12**34**56 的第三位和第四位，表示轴号。
%N	专用于 PLC 报警： 十进制报警 ID 1234**56** 的第五位和第六位，表示信号号。

13.2.8 打开故障文件

转换过程中出现的错误将写入文件“alarmtext_conversion.log”或“oem_text_conversion.log”中。

保存目录： /**user**/sinumerik/hmi/log

打开故障文件



1. 选择操作区域“调试”。



2. 按下软键“系统数据”。
3. 打开文件夹“System CF-Card”。
4. 打开保存目录，选中所需文件。



5. 按下软键“打开”，打开错误记录文件。

-或者-



按下 <INPUT> 键。

13.3 配置报警记录

缺省设置

标准情况下，报警日志中包含了自最后一次启动的所有报警和消息，且顺序与其来、去的时间戳相吻合。但 NC 零件程序的消息除外（msg 指令）。

与“报警列表”或者“消息”窗口中不同，显示日志时同样会显示此时间点时无效的报警和消息（历史报警事件）。

确定事件数量

报警日志的结构为环形缓冲器：当超过最大容量时，会用新的报警事件来覆盖最旧的记录项。每个事件（进入、退出、确认等）都显示为一个单独的条目。报警日志的最大长度不以包含的报警数量而定，而是以条目数量而定。

持久备份（持续备份）

需要时可以持久设置报警日志，使其中也包含有上一次开机之前的报警日志。日志可随后按时间或按每个报警事件触发，以内部二进制格式保存到文件系统的目录“card/**user**/sinumerik/hmi/log/alarm_log”中。

根据所使用的硬件不同，报警日志将保存在 NCU 的 CF 卡中或者 PC/ PCU 的硬盘上。

注意
缩短 CF 卡的使用寿命 若需永久保存报警日志而将其写入 CF 卡上时，卡只能保存一定限制数量的写循环。因此要确保仅在切实需要时才进行备份！如果不再需要保存报警日志，请再次取消勾选“对每一事件”。 删除目录“/alarm_log”中的报警日志“slaepp_<nr>.hmi”，以便在 CF 卡上腾出存储空间。 如果平均每天写入的数据量在 60 MB 左右，那么 CF 卡的使用寿命可达 10 年以上。 在预设的配置中不备份报警日志。

配置报警日志

- 通过操作区域“诊断”中的操作软件
- 通过配置文件“slaesvcconf.xml”

13.3.1 通过操作面板设置报警记录

步骤



1. 选择操作区“诊断”。





2. 按下软键“报警日志”和“设置”。

3. 在“项数”栏中输入所需的数字，可以修改所管理的来、去事件的最大数量。

   报警日志的结构为环形缓冲器：当超过最大容量时，会用新的报警事件来覆盖最旧的记录项。每个来、去时间戳均表示一个独立的事件。

   您可以给定 500 到 32000 之间的一个值。标准数值为 500。

4. 在如下条目的“文件写入模式”栏中选择：
   - “关”，不会记录变更（标准设置）
   - “对每一事件”，每次报警事件发生时都会保存报警日志（永久保存）。
   - “时间控制的”，间隔一段时间后保存报警日志。另外在输入栏“写间隔”中可以以秒为单位设定时间。

   若仅保存在 CF 卡上，会收到警告提示。

重新启动 SINUMERIK Operate

文件中的设置只有在重启操作软件后才生效。为此重新启动 SINUMERIK Operate。

13.3.2 通过配置文件设置报警记录

创建配置文件

在文件“slaesvcconf.xml”中进行配置。复制以下目录中的配置文件模板“oem_alarmprot_slaesvcconf.xml”：/**siemens**/sinumerik/hmi/template/cfg。

步骤：

1. 将文件保存到目录 /**oem**/sinumerik/hmi/cfg 或 /**user**/sinumerik/hmi/cfg 中。
2. 命名此文件为“slaesvcconf.xml”。

3. 打开文件，在标签 <Records type .../> 中输入需要输出事件的数量。缺省值为 500。最大事件数量取决于存储介质（硬盘或 CF 卡）。
4. 在标签 <DiskCare type="int" value="-1"/> 中输入备份模式。允许值为：
   -1:不进行报警记录的备份（缺省值）。
   0：每个报警事件会触发即时的报警记录备份（永久备份）。
   >0：记录备份周期，单位秒：只要有修改，就备份所有设置了按时触发（n > 0）的记录（永久备份）。
5. 在标签<Filter>中根据条目类型来设置过滤器。此时需注意下列内容:
   – 只有在满足过滤标准时，报警事件才会进入记录中。
   – 连续设置了多个过滤器时，这些过滤器进行逻辑或运算。
   – 如需多个过滤器进行逻辑与运算，必须用指令 AND 连接这些过滤器。

说明

每条报警或每条消息的出现事件和消失事件都需要占用单独的条目，即使这些事件属于一条报警或一条消息。

此外报警记录中还包含有应答事件。应答时间也需要占用一个条目，即使当前在记录显示中还无法看出。

过滤器属性

过滤器由下列三个单元组成：<标识> <关系> <值>。

条目<标识>	说明
AlarmID	报警号
SourceID	允许的 SourceID 和 SourceURL 的默认值参见章节报警序号范围 (页 226)中的表格。
SourceURL	
<属性名称>	文件“slaedatabase.xml”中任意的报警属性，比如“SEVERITY”或“CLEARINFO”。

条目<关系>	说明
EQUAL	等于
NOT	不等于
LOWER	小于
HIGHER	大于

条目<值>	说明
数字	-
字符串	-

清除标准

清除标准 (ClearInfo)	报警源 (Source)	说明
0	/HMI	由 HMI 删除的报警
1	/NCK	由 NCU 重新上电清除的报警
2		由 NCU 的硬件复位清除的条件。
3		由向 NCU 发送“取消”命令而清除的条件。
4		由 NCK 本身清除的条件。
5		由向 NCU 发送“NC 启动”命令而清除的条件。
6		由操作方式复位而清除的条件。
7		由向 NCU 发送“NC 复位”命令而清除的条件。
8	/PLC	FB15 的 PLC 消息（基本程序）
9		FB15 的 PLC 报警（基本程序）
10		由“[^]”键清除的 HMI 诊断报警。
11		预留
12		PLC (SFC17/18)带有状态“未应答”的 S7-PDiag、S7-Graph、S7-HiGraph 报警或其他_S(Q)报警。
13		PLC (SFC17/18)带有状态“已应答”的 S7-PDiag、S7-Graph、S7-HiGraph 报警或其他_S(Q)报警。
14	/NCK	驱动通过 NCK 发出的报警
15		零件程序消息
16		SINAMICS Safety Integrated 信息

示例

记录所有 ClearInfo 不等于 15 的报警，也就是说：不记录零件程序消息。

```
<CONFIGURATION>
  <Protocol>
   <Filters>
      <Siemens_Filter_01 type="QString" value="CLEARINFO NOT
      15" />
   </Filters>
  </Protocol>
</CONFIGURATION>
```

记录所有 SourceURL 为"/NCK"或"/HMI"的报警：

```
<CONFIGURATION>
  <Protocol>
   <Filters>
      <Filter_01 type="QString" value="SourceURL EQUAL /NCK" />
      <Filter_02 type="QString" value="SourceURL EQUAL /HMI" />
   </Filters>
  </Protocol>
</CONFIGURATION>
```

在标签<FilePath>中设置永久保存报警记录的文件的名称和路径。

```
<CONFIGURATION>
  <Protocol>
   <Filters>
     <FilePath type="QString" value="$(HMI_INSTALL_DIR)user/
     sinumerik/hmi/ log/alarm_log/slaepp_" />
   </Filters>
  </Protocol>
</CONFIGURATION>
```

- 路径
  在路径中也可以使用环境变量，例如用于安装目录：$(HMI_INSTALL_DIR)。
- 文件名
  在运行时会默认自动将一个 4 位的取值范围从 0000 至 0009 的编号以及文件后缀"hmi"添加在给定的文件名之后。编号自动提高，当：
  - 文件达到最大容量(DiskCare = 0)或者
  - 给定的时间段届满(DiskCare > 0)。
    此时将旧的文件删除。
- 记录的数量
  - 所保存记录的数量可通过 MaxFileNumbers 在 2 至 9999 之间（默认：10）进行调整。

示例：关闭现有的过滤器

执行以下步骤，可关闭现有的西门子过滤器，而无需覆写该过滤器：

1. 将过滤器清空可以关闭所有迄今为止设置的**所有**过滤器。
   此操作无需了解过滤器名称。
2. 过滤器的效力取决于设置过滤器的顺序：
   siemens → addon → oem → user

以筛选出报警 700000 到 700010、700500 到 700599 及零件程序信息 (CLEARINFO = 15) 为例：

```
<CONFIGURATION>
  <Protocol>
    <Filters>
      <Filter_00 value="" type="QString"/>
      <Filter_01 value="CLEARINFO NOT 15 AND AlarmID LOWER
      700000" type="QString"/>
      <Filter_02 value="CLEARINFO NOT 15 AND AlarmID HIGHER
      700010 AND AlarmID LOWER 700500" type="QString"/>
      <Filter_03 value="CLEARINFO NOT 15 AND AlarmID HIGHER
      700599" type="QString"/>
    </Filters>
  </Protocol>
</CONFIGURATION>
```

结果

重启 SINUMERIK Operate，激活记录设置的修改。

13.4 含参数的 PLC 报警

简介

在由 PLC 功能块 SFC17 或 SFC18 触发的报警中，在每次调用功能块时可传递一个参数。如果此时选择的是数据类型"Octet-String"，则最多可传送 12 个字节。

适当地配置操作软件后，这 12 个字节可以是由字节组成的数组或只是由一些简单类型的数据按照任意顺序组成。利用这些字节报警文本中也可以显示多个参数。

为此必须在操作软件中为每条报警定义一个参数说明。定义该参数说明时必须考虑两点：

- 如何正确编译参数（数据类型和长度）？
- 如何正确处理参数，以便在屏幕上显示参数（字符串或是数字、十进制、十六进制还是二进制）？

第 1 点和屏幕上选择的语言无关，第 2 点则和语言有关。

因此参数说明分成两部分：一部分为和语种无关的参数指令，一部分为和语种有关的格式指令，保存目录为：

- 参数指令：/oem/sinumerik/hmi/cfg/ 或者 /user/sinumerik/hmi/cfg/
- 格式指令：/oem/sinumerik/hmi/lng 或者 /user/sinumerik/hmi/lng

文件的名称可以自由选择。参数指令以报警属性"HMIPROPARAMDESCR2"的形式插入到数据库中对应报警的配置数据中。格式指令因为和语种相关，所以它是报警文本的一部分。

13.4.1 定义数据类型为"Octet-String"的参数

参数指令

参数指令指出了一个或多个参数如何理解由报警 PDU 传送的、最长 12 个字节的 Octet-String。

句法

参数指令的句法为：

<参数指令 Ext> ::= [% <参数说明>]

<参数说明 Ext> ::= <伴随值号><元素类型><偏移>

<伴随值号> ::= 始终为 1（为将来的应用预留）

<偏移> ::= 参数在伴随值内的偏移，和数据类型相关：

- 布尔型(B)：
  在伴随值内按位偏移
- 布尔型以外的所有数据类型(Y, W, X, I, D, C, R)：
  在伴随值内按字节偏移

元素类型	数据类型
Y	字节型（8 位，无符号）
W	单字型（16 位，无符号）
X	双字型（32 位，无符号）
I	整型（16 位，带符号）
D	整型（32 位，带符号）
B	布尔型（1 位）
C	字符型（8 位）
R	实型（32 位）

示例

%1W0%1W2 → 参数 1： 伴随值 1，单字型，字节偏移为 0
参数 2： 伴随值 1，单字型，字节偏移为 2

%1Y0%1Y1%1W2 → 参数 1： 伴随值 1，字节型，字节偏移为 0
参数 2： 伴随值 1，字节型，字节偏移为 1
参数 3： 伴随值 1，单字型，字节偏移为 2

%1B0%1B1%1B2%1B3%1I1 → 参数 1： 伴随值 1，布尔型，位偏移为 0
参数 2： 伴随值 1，布尔型，位偏移为 1
参数 3： 伴随值 1，布尔型，位偏移为 2

参数 4： 伴随值 1，布尔型，位偏移为 3

参数 5：伴随值 1，整型（16 位，带符号），字节偏移 1

以报警属性文件为例（创建该文件的步骤见修改报警属性 (页 216)）：

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SlAeAlarmAttributes Version="01.00.00.00">
<Types>
  <Type TypeID="32" TypeName="Condition">
  <Category CatID="1" Version="1.0">
    <CatDescr>Sinumerik 840D sl 的报警。</CatDescr>
    <Attributes>
      <Attribute AttrName="MSGTEXT" AttrID="-1" AttrDataType="10"/>
      <Attribute AttrName="HMIPROPARAMDESCR2" AttrID="5012" AttrDataType="10"/>
    </Attributes>
  </Category>
  </Type>
</Types>
<Sources>
  <Source CatLink="1" SourceID="51" SourceURL="/PLC/PMC">
    <Alarms>
      <!-- 报警 700000，含 3 个字节型参数 -->
      <Alarm AlarmID="700000">
        <MSGTEXT>my_oem_plc_alarms_context|700000/PLC/PMC</MSGTEXT>
        <HMIPROPARAMDESCR2>%1Y0%1Y1%1Y2</HMIPROPARAMDESCR2>
      </Alarm>
      <!-- 报警 700001，含 3 个单字型参数 -->
      <Alarm AlarmID="700001">
        <MSGTEXT>my_oem_plc_alarms_context|700001/PLC/PMC</MSGTEXT>
        <HMIPROPARAMDESCR2>%1W0%1W2%1W4</HMIPROPARAMDESCR2>
      </Alarm>
      <!-- 报警 700002，含 3 个 16 位整型参数 -->
      <Alarm AlarmID="700002">
        <MSGTEXT>my_oem_plc_alarms_context|700002/PLC/PMC</MSGTEXT>
        <HMIPROPARAMDESCR2>%1I0%1I2%1Y4</HMIPROPARAMDESCR2>
      </Alarm>
      <!-- 报警 700003，含 4 个参数： 字节型 -> 单字型-> 32 位整型-> 字节型-->
      <Alarm AlarmID="700003">
        <MSGTEXT>my_oem_plc_alarms_context|700003/PLC/PMC</MSGTEXT>
        <HMIPROPARAMDESCR2>%1Y0%1W1%1D3%1Y7</HMIPROPARAMDESCR2>
      </Alarm>
```

```
    </Alarms>
  </Source>
</Sources>
</SlAeAlarmAttributes>
```

13.4.2 定义和语种相关的格式指令

格式指令

格式指令是报警文本的一部分，是需要显示的参数的占位符。

句法

格式指令的句法为：

<格式指令>	::=	@ <参数说明号><格式数据> @
%[i]x	::=	十六进制值，有 i 位
%[i]u	::=	十进制值，有 i 位，无符号
%[i]d	::=	十进制值，有符号，有 i 位（含符号）
%[i]b	::=	二进制值，有 i 位
%[i][.y]f	::=	定点数： 带符号，格式为[-]xxx.yyyy，其中 i 表示含符号、小数点的位数，y 表示小数点后的位数。 小数点前后缺少的数位一律用 0 填满。 如果小数点后的位数超过 y，系统会舍去多余的小数位。

没有指定位数 i 时，系统会只采用完整表达一个数字所需的位数。 同样如果指定的位数 i 不够完整表达一个数字及其符号，系统会采用完整表达所需的位数而忽略 i。

在格式指令中包含的号不是报警 PDU 中伴随值的号，而是参数指令中参数说明的号。 一条报警中可以包含任意多的格式指令。

示例

@1%b@ → 第一个报警说明中的值用二进制表示，比如：1011011。

@2%8X@ → 第二个报警说明中的值用 8 位数的十六进制表示，缺少的位用 0 填满，比如：00AF37FE

@1%7.2f@ → 第一个报警说明中的值用 7 位数（含符号和小数点）的定点数表示，小数点后有 2 位，缺少的位用 0 填满，比如：-012.34

以报警文本文件为例（创建该文件的步骤见创建自身报警文本 (页 208)）：

```
<!DOCTYPE TS>
<TS>
<context>
  <name>my_oem_plc_alarms_context</name>
  <message>
    <source>700000/PLC/PMC</source>
    <translation>报警文本含 3 个字节型参数： @1%X@, @2%u@, @3%b@</translation>
  </message>
  <message>
    <source>700001/PLC/PMC</source>
    <translation>报警文本含 3 个单字型参数： @1%X@, @2%u@, @3%b@</translation>
  </message>
  <message>
    <source>700002/PLC/PMC</source>
    <translation>报警文本含 3 个 16 位整型参数： @1%X@, @2%u@, @3%b@</translation>
  </message>
  <message>
    <source>700003/PLC/PMC</source>
    <translation>报警文本含 4 个参数： @1%X@, @2%u@, @3%r@, @4%b@</translation>
  </message>
</context>
</TS>
```

13.5 取消激活警告提示

通常报警/消息都包含了报警文本。 如果没有报警文本，则输出警告提示“no text available”（无文本）。

在配置文件“slaesvcadapconf.xml”中添加区域 <ControlFlags>，可以取消报警提示。

您可以从原始文件中复制该区域。 文件“slaesvcadapconf.xml”位于如下目录中： **/siemens/** sinumerik/hmi/base

设置

在标签 <MissingTextWarning type="bool" value="FALSE"/> 中您可以激活/取消警告提示。

条目	含义
TRUE	显示警告提示。
FALSE	取消警告提示。

示例

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CONFIGURATION>
        <ControlFlags>
            <MissingTextWarning type="bool" value="FALSE"/>
        </ControlFlags>
</CONFIGURATION>
```

数据备份 14

14.1 概览

数据备份

我们建议在以下情况中执行数据备份：

- 开机调试后
- 更改了机床专用设置后
- 更换硬件组件后
- 在软件升级中
- 激活存储器配置机床数据之前

说明

调试存档将以".arc"（存档）格式保存。

通过 SINUMERIK Operate 操作界面您可以采取不同的方法创建存档并再次读入存档。

- 正常存档： 通过软键“系统数据”您可以根据需要从文件树形图中选择并备份数据。
- 调试存档： 按下软键“调试存档”提供以下选项：
  - 创建和读入调试存档
  - 创建 PLC 硬件升级存档（仅 SDB）
  - 创建并读入初始状态存档

保存地点

存档可以保存在以下目录中：

- CF 卡中的： System CF-Card/**user**/sinumerik/data/archive 或者 System CF-Card/**oem**/sinumerik/data/archive
- PCU: F:hmisl**user**sinumerik\data\archive 或者 **oem**sinumerik\data\archive
- 所有配置的逻辑驱动器（USB 驱动器、网络驱动器）

说明

USB 设备

USB 设备不适合作为永久保存媒介。

设置行校验和

读取数据存档至少需要一个能够顺利执行存档中现有机床数据指令的权限。通常是“制造商”。为了也能够通过“服务”权限读取数据存档，应事先通过行校验和生成存档。数据存档中现有的机床数据指令包含有效的行校验和且在缺少初始权限的青虾也能执行。

说明

如果在数据存档中通过机床数据指令修改行校验和，行校验和则无效且读取存档时会输出报警 12600。

通过以下机床数据在存档中激活行校验和生成（缺省）：

MD11230 $MN_MD_FILE_STYLE	机床数据备份文件的结构
位 0 = 1	生成行校验和
位 1 = 1	生成机床数据编号

14.2 创建调试存档

一览

在控制系统调试结束后，您可以创建一份调试存档，此时您可以选择一种数据单独存档或多种数据一起存档。存档文件可以传送到另一个控制系统，使该系统达到相同状态。

说明

在创建调试存档时，不会涉及存储在外部（网络驱动）及可能进行了设置的全局零件程序存储器的程序。只对 NC 存储器和本地驱动器中的程序进行存档。

前提条件

您至少需要访问级别 3（用户）。

控制系统的数据

数据分类		数据
NC 数据		机床数据 设定数据 选项数据 全局（GUD）和本地（LUD）用户数据 刀具和刀库数据 保护区域数据 R 参数 零点偏移 工件、零件程序和子程序 标准循环和用户循环 定义和宏指令
	带有补偿数据	• QEC - 象限误差补偿 • CEC- 直线/悬垂度补偿 • EEC - 丝杠螺距/编码器误差补偿 只有当调试文件需要再次导入同一个控制系统时，才建议存档机床专用的补偿数据。
	带编译循环	如果存在编译循环，则显示编译循环（*.elf）。

数据分类		**数据**
PLC 数据（包含 CP）		OB（组织模块） FB（功能模块） SFB（系统功能模块） FC（功能） SFC（系统功能） DB（数据模块） SDB（系统数据模块）
驱动数据		驱动数据的存档有二进制或 ASCII 格式供选择。
所有 HMI 数据		
部分 HMI 数据		如果存在数据，必须打勾激活组件。
	文本	用户文本、报警文本
	模板	单个模板、工件模板
	应用程序	HMI 应用程序、OEM 应用程序
	设计	设计
	配置	配置，包括显示机床数据
	帮助	帮助文件
	版本数据	版本数据
	记录	错误日志
	程序列表	程序列表
	字典	字典
	数据备份	在目录 HMI 数据/数据备份中“系统数据”树形图里的文件。
	本地驱动器/ NC Extend 上的程序	保存在用户存储区的程序。 **提示：**注意，用户数据可能很大，因此由此生成的存档文件可能会需要占用更大存储器。
注释		调试存档的注释。
创建人		调试存档的创建人员和日期。

步骤



1. 选择操作区域“调试”。



2. 按下菜单扩展键。



3. 按下软键“调试存档”。

   “调试”窗口打开。



4. 激活“创建调试存档”并按下软键“确认”。

   “创建调试存档”窗口打开。

5. 选中所需的数据类别。

6. 必要时，可以输入注释、创建人名称和创建日期。



7. 按下软键“确认”。

   窗口“创建存档：选择存储位置”打开。



8. 如果您想搜索某个目录或子目录，可以将光标移到对应的存储位置，按下软键“搜索”，然后在搜索对话框中输入关键字。

   提示：用星号“*”替代字符串，用问号“？”替代字符可以使搜索更简单。



   -或者-



   选择所需存储器，按下软键“新目录”、在“新目录”窗口中输入名称并按下软键“确认”，创建一个新的目录。





9. 按下软键“确认”。

   窗口“创建存档：名称”打开。



10. 输入所需名称并按下软键“确认”。

    现在，一份 ARC 格式的存档保存在所选目录中。

14.3 导入调试存档

注意
软件升级时扩展语言丢失 软件升级时会覆盖系统中的所有语言。如果系统中的语言比升级存档文件中的多，则系统读取存档时这些扩展语言会丢失。软件升级之前无法进行语言备份。

说明

读入含驱动器配置的调试存档时会导致覆盖 logdrive.ini 文件中的驱动器配置。系统会创建一个名为 logdrive_ini.sav 或 logdrive.oldstyle 的备份文件。读入存档前，系统会先激活新的驱动器配置。

说明

为了防止 NCU 意外关闭，必须在读入调试存档之前禁用所有节能曲线。

更多信息：操作手册，通用版/车削版/铣削版/磨削版

前提条件

您至少需要访问级别 2（服务）。

步骤



1. 选择操作区域“调试”。



按下菜单扩展键。



2. 按下软键“调试存档”。

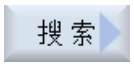


3. 激活“读入调试存档”并按下软键“确认”。

“选择调试存档”窗口打开。

4. 选择存档的存储位置，并将光标移到所需调试存档(ARC)上。

-或者-





如果想查找指定的目录、子目录或调试存档，可以选择存储器，按下软键“搜索”，输入关键字并按下软键“确认”。



5. 按下软键“确认”。
“读入调试存档”窗口打开。
显示路径、版本信息、存档名称等概览以及存档组件列表。
取消复选框勾选，以排除组件的读入过程。
6. 按下软键“确认”，读入所选存档文件。
窗口“读入存档”打开，用进度条显示导入进度。



7. 如果要覆盖已存在的文件，按下软键“全部覆盖”。

-或者-



如果不要覆盖已存在的文件，按下软键“不覆盖”。

-或者-



若只需覆盖个别特定文件，可按下软键“跳过”。
接着会弹出一张“‘读存档”故障记录”，其中会列出已跳过的文件或覆盖的文件。



8. 按下软键“取消”，中断导入。

14.4 备份硬件设计

硬件配置是通过系统数据块 SDB 完成的，因此，只有系统数据会进入存档。

步骤



1. 选择操作区域“调试”。



按下菜单扩展键。



2. 按下软键“调试存档”。
“调试”窗口打开。



3. 勾选“PLC 硬件升级存档（仅 SDB）”，按下软键“确认”。
窗口“创建 PLC 硬件升级存档（仅 SDB）”：选择存储位置”打开。

4. 在显示的文件树形图中选择需要的保存地点。

-或者-



按下软键“新建目录”，创建自有目录。

窗口“新建目录”打开。



5. 输入所需名称并按下软键“确认”。

现在在该文件夹下成功新建了一个目录。



6. 按下软键“确认”。

窗口“创建存档：名称”打开。



一份 ARC 格式的存档显示在窗口中。现在输入所需的名称并按下软键“确认”。

存档成功创建，保存在所选的目录中。

14.5 创建原始数据存档

您可以创建一份原始数据存档，以便恢复控制系统的出厂状态，存档名称为“original.arc”。

该文件位于 CF 卡的目录“存档/制造商” 下。

如果在该目录中没有文件“original.arc”，我们推荐从提供的控制系统原始数据中创建存档。

正如创建调试存档一样，您可以选择一种数据单独存档或多种数据一起存档。

前提条件

NC 数据的存档需要访问级别 2（服务）。

备份程序/工件需要访问级别 6（钥匙开关位置 1）。

控制系统的数据

数据分类		数据
NC 数据		机床数据 设定数据 选项数据 全局（GUD）和本地（LUD）用户数据 刀具和刀库数据 保护区域数据 R 参数 零点偏移 补偿数据 工件、零件程序和子程序 标准循环和用户循环 定义和宏指令
	带有补偿数据	• QEC - 象限误差补偿 • CEC- 直线/悬垂度补偿 • EEC - 丝杠螺距/编码器误差补偿 • 只有当调试文件需要再次导入同一个控制系统时，才建议存档机床专用的补偿数据。
	带编译循环	如果存在编译循环，则显示编译循环（*.elf）。
PLC 数据（包含 CP）		OB（组织模块） FB（功能模块） SFB（系统功能模块） FC（功能） SFC（系统功能） DB（数据模块） SDB（系统数据模块）
所有 HMI 数据		
部分 HMI 数据		
	循环存档	定义和循环
	文本	用户文本、报警文本
	模板	单个模板、工件模板
	应用程序	HMI 应用程序、OEM 应用程序

数据分类		数据
	设计	设计
	配置	配置，包括显示机床数据
	帮助	帮助文件
	版本数据	版本数据
	记录	错误日志
	用户视图	用户自定义表格，由自选的机床数据和设定数据组成。
	字典	字典
	本地驱动器/ NC Extend 上的程序	保存在用户存储区的程序。 **提示：**注意，用户数据可能很大，因此由此生成的存档文件可能会需要占用更大存储器。

步骤



1. 选择操作区域“调试”。



2. 按下菜单扩展键。



3. 按下软键“调试存档”。
   “调试”窗口打开。



4. 勾选复选框“创建原始数据存档”，按下软键“确认”。
   “创建原始数据存档”窗口打开。
5. 选中所需的数据类别。



6. 必要时，可以输入注释、创建人名称和日期，并按下软键“确认”。
   如果存档“original.arc”已经存在，则输出一条警告提示。



7. 按下软键“确认”覆盖该文件。
   现在，新的存档成功创建，保存在目录“存档/制造商”下。
   -或者-



按下软键“取消”，不覆盖已有文件。

14.6 读入原始数据存档

控制系统的原始数据保存在文件“original.arc”中。 如果您需要将控制系统恢复到出厂状态，可以导入原始数据存档。

前提条件

您需要访问级别 2（服务）。

步骤



1. 选择操作区域“调试”。



2. 按下菜单扩展键。



3. 按下软键“调试存档”。
   “调试”窗口打开。



4. 勾选复选框“读取原始存档”并按下软键“确认”。
   SINUMERIK Operate 自动访问文件“original.arc”。
   此时会弹出一条信息，询问您是否要执行调试。



5. 按下软键“确认”。
   存档开始导入系统，一条进度条会显示导入进度。
   接着会弹出一张“‘写存档”故障记录。

14.7 生成完整存档

您可以在外部存储器上保存系统的所有数据，以便为技术支持人员开展诊断提供便利。

步骤

1. 通过 USB 接口插入一个存储器。
2. 按下组合键 <Ctrl> + <Alt> + S。
   诊断所需的全部数据将集中保存在一个文档中。
   提示：
   存档期间不能进行任何操作。
3. 存档结束后，会有消息提示。
   存档名称由系统给出："CompletArchiv<日期>_<时间>.arc.
   对话框关闭，可重新操作控制系统。

14.8 串行接口(V24 / RS232)

14.8.1 通过串行接口读入/读出存档

可通过 V24 串行接口在操作区“程序管理器”和“启动”中读入读出存档。

串行接口 V24 的可用性

如要更改接口 V24 的可用性，可在文件“slpmconfig.ini”中设置以下参数：

参数	说明	
[V24]	说明与设置相关的参数所在的段落。	
useV24	串行接口 V24 可用性的设置	
	= true	接口和软键可用（默认）
	= false	接口和软键不可用

文件“slpmconfig.ini”的保存

用于 SINUMERIK Operate 的文件模板“slpmconfig.ini”保存在以下目录：

<安装路径>/siemens/sinumerik/hmi/template/cfg

将文件复制到以下一个目录中：

<安装路径>/**user**/sinumerik/hmi/cfg

<安装路径>/**oem**/sinumerik/hmi/cfg

说明

如要通过自定义更改对概览进行改进，则只需从文件副本“slpmconfig.ini”中删除未更改的参数。

读出存档

需要发送的文件（目录、单个文件）被打包成一个存档文件（*.arc）。存档文件 (*.arc) 可以直接发送，无需另外打包。如果同时选择了一个存档文件 (*.arc) 和另外一个文件（例如目录），则将其打包到一个新的存档中，然后发送。

读入存档

如果要读取存档，可以使用接口 V24。传输存档，然后解包。

说明

导入调试存档

通过 V24 接口读入调试存档时，该存档会立即激活。

外部编辑穿孔带格式

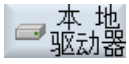
需要外部编辑存档时，首先创建一份穿孔带格式的文件。

步骤





...



1. 选择操作区域“程序管理器”，按下软键“NC”或“本地驱动器”。

-或者-



选择操作区“启动”，按下软键“系统数据”。



读出存档

2. 选中您需要发送到 V24 上的目录或文件。



3. 按下软键“>>”和“存档”。





4. 按下软键“V24 发送”。

-或者-

读入存档



需要通过 V24 读入文件时，按下软键“V24 接收”。

14.8.2 设置接口参数

步骤



1. 选择操作区域“调试”。



2. 按下软键“系统数据”。



3. 按下软键“>>”和“存档”。





4. 按下软键“RS232C 设置”。
窗口“接口：V24”打开，并显示接口设置。



5. 按下软键“详细”，查看或编辑接口的其他设置。

V24 的设置

参数	含义
协议	RS232C 接口支持以下协议： • RTS/CTS
传输	另外它还支持断点传输协议（ZMODEM 协议）。 • 标准传输（缺省设置） • 断点传输 断点传输和握手协议 RTS/CTS 一同设置用于选中的接口。
波特率	传输速率：可使用的波特率取决于连接的设备，导线长度和电气环境条件。 • 19200（缺省设置，NCU (Linux) 的最大波特率） • 115200（PC/PCU (Windows) 的最大波特率）
存档格式	• 穿孔带 • 二进制格式（PC 格式）
V24 的设置（详细）	
接口	• COM1 • COM2（仅用于 PC 机上的 SINUMERIK Operate）
奇偶校验	奇偶校验位用于错误识别：奇偶校验位添加在编码的字符上，目的是使"1"位上的数字变为一个奇数（奇校验）或者为一个偶数（偶校验）。 • 无（缺省设置） • 奇校验 • 偶校验
停止位	异步数据传输时的停止位数目。 • 1（缺省设置）
数据位	异步传输时的数据位数目。 • 8 位（缺省设置）
XON (Hex)	仅针对穿孔带格式
XOFF (Hex)	仅针对穿孔带格式

参数	含义
传输结束 (hex)	仅针对穿孔带格式 通过传输结束符号停止 传输结束字符的缺省设置是 (HEX) 1A。
时间监控秒数	时间监控 在出现传输故障或传输结束但是没有传输结束字符时，传输在设定的秒数后中断。 时间监控是由一个计时器控制的，它随第一个字符启动，每传送一个字符便复位一次。时间监控秒数是可以设置的。

14.9 保存装调数据

“保存装调数据”的设置

仅在机床数据“MD11280 $MN_WPD_INI_MODE”为缺省值 1 时，才能使用“保存装调数据”功能。 否则在选择零件程序后的第一次 NC 启动时会自动执行后缀为“.ini”的同名文件。

机床数据：

MD11280 $MN_WPD_INI_MODE		
工件目录中配置文件（ini 文件）的编辑模式		
= 1	在首次启动 NC 选择工件后，便执行所选零件程序名称对应的 ini 文件和带以下扩展名的 ini 文件。	
	CEC	垂度补偿(Cross Error Compensation)
	GUD	用户数据(Global User Data)
	PRO	保护区
	RPA	R 参数
	SEA	赋值(Setting Data Active)
	TMA	刀库数据(Tool Magazine Active)
	TOA	刀具补偿(Tool Offset Active)
	UFR	零点偏移(User Frame)

14.10 网络设置

激活

缺省设置中系统不备份网络设置。您必须手动使能该项备份。

1. 从目录 /siemens/sinumerik/hmi/template/cfg 中复制文件“slpmconfig.ini”。
2. 将文件保存到目录/oem/sinumerik/hmi/cfg 或/user/sinumerik/hmi/cfg 中。
3. 打开文件，在[SeriesSetup]中输入：
   SupportNetworkSettings=true
4. 重新启动 SINUMERIK Operate。

说明

该设置也作用于利用 CTRL+ALT+S 或 CTRL+ALT+C 创建完整标准存档这一操作。

更多信息

有关按键组合的说明，请参见

操作手册，通用版/车削版/铣削版/磨削版。

限制

- 租赁文件不会一同备份：
  - /system/etc/udhcpd-eth0.leases
  - /system/etc/udhcpd-ibn0.leases
- 在配置设备网络时，必须依据 DNS 名称进行配置，而不是依据 IP 地址。

基本步骤

- 备份网络设置
  勾选该选项，您可以在创建调试存档时一同备份网络设置，见创建调试存档 (页 245)。
- 读取网络设置
  网络设置备份可以如同调试存档一样读取，见导入调试存档 (页 248)。

说明

有备件时读取网络设置

在读取网络设置备份前，首先要断开设备网络中的备件。

NCU

通过 NCU 上的 SINUMERIK Operate 可对以下 NCU 设置进行备份：

- /user/system/etc/basesys.ini
- /user/common/tcu（整个目录）

说明

NCU 的网络设置只能通过 PC / PG 上的 SINUMERIK Operate 备份和读取。
PC / PG 上 mmc.ini 中的 NC 地址是 192.168.215.1。

PCU

使用 PCU 上的 SINUMERIK Operate 可以备份以下 PCU 和 NCU 的设置：

- PCU
  - /user/system/etc/basesys.ini
  - SINUMERIK Operate 的 mmc.ini
- NCU（当作为主站工作时）
  - /user/system/etc/basesys.ini
  - /user/common/tcu（整个目录）

备份产生以下存档：

- <Archivname>_pcu.arc
- <Archivname>_ncu.arc

在“NCU 主站(DHCP)+PCU (on_low)”的配置中读取存档：

1. 利用 SINUMERIK Operate 读取 PCU 上的存档 “存档名称_pcu.arc”。
2. 在换入 NCU 备件时注意以下事项：
   - 没有更换 NCU 时，设备重新启动。
   - 如要更换 NCU，则必须通过接口 X127 在单独的操作界面上读取 <Archivname>_ncu.arc。

配置网络 15

概述

若需在 SINUMERIK Operate 中进行网络设置，可采用以下方案：

- 通过操作区“调试”可进行针对设备网络和公司网络的所有基本设置（参见章节“显示网络一览 (页 261)”）。
- 通过操作区“诊断”能够获取所有可用的网络适配器的概览，并可指定和检查例如针对公司网络或者 DHCP 服务器的设置（参见章节“配置 TCP/IP 诊断 (页 285)”）。

15.1 显示网络一览

在“网络设置一览”窗口中列出了所有可用的接口及其网络设置。此处会显示公司网络和系统网络的 IP 地址、子网掩码和 MAC 地址。可以对该设置进行编辑。

步骤



1. 选择操作区域“调试”。



2. 按下菜单扩展键。



3. 按下软键“网络”。



“网络设置一览”窗口打开。



4. 按下软键“更改”，对设置进行修改。

参见

系统网络的设置 (页 262)

公司网络设置 (页 266)

保存网络设置 (页 267)

15.2 系统网络的设置

窗口“系统网络的设置”中列出了 DHCP 服务器、路由转发、生效的协议以及 DNS 服务器和其他系统网络属性的配置。

设置	说明
DHCP 激活	激活或取消激活系统网络的 DHCP 模式，这样，IP 地址可以自动分配给系统网络中的节点。
DHCP 服务器同步模式	激活或取消激活 DHCP 服务器同步模式。如果该设置激活，系统网络(X120)中的 DHCP 服务器会相互同步，进而只有其中一个服务器分配有效地址。可以通过这种方法同时运行多个 NCU 或 PCU，而无需修改网络设置。 此时没有激活的 DHCP 服务器进入“待机”模式，它会定期从激活的服务器上读取当前地址数据和 TCU 数据，以便在激活服务器发生故障时待机服务器可以迅速激活，而不丢失数据。 • 通过设置“主站优先级”可让带有“主站”设置的服务器始终作为有效服务器工作（为此，该服务器在网络中必须是激活的）。这样，通常情况下就可以始终由同一个 DHCP 服务器控制，并在该服务器中找到当前的地址和 TCU 数据。“主站优先级”只允许用于系统网络中的唯一一个 DHCP 服务器。 • 通过设置“低优先级”和“高优先级”可以确定优先将哪个 DHCP 服务器用作有效服务器。如果网络中没有带高优先级的服务器，则只能将带低优先级的 DHCP 服务器用作有效服务器。 缺省设置：“高优先级”
DHCP 地址范围	确定 DHCP 地址的范围。该范围从开始中确定的地址开始，到结束中输入的地址结束。由此产生的号段得出以下缺省设置： 起始地址 = 系统网络中的第一个地址 结束地址 = 起始地址 + 10 或 + 2（当地址数少于 16 个时）

设置	说明
等待主站超时	确定应答主站服务器的最大等待时间（单位为秒）（参见 DHCP 服务器同步模式）。时间届满后，该服务器本身变为有效服务器。 通过这种额外的暂停可以使指定的 DHCP 主站顺利变为有效 DHCP 服务器，即使该主机的启动比其它控制器稍晚或占用更长时间。 缺省设置：120 秒
主机名	为 DHCP 服务器指定一个固定的主机名。这样就既可以通过 IP 地址，也可以通过主机名来访问该服务器。 由于主机名也用作 DNS 名称，因此它必须满足相应的 RFC 要求： • ASCII 字母(a-z, A-Z)、数字(0-9)和"-" • 最多 63 个字符
DNS 域	确定系统网络中使用的 Top Level Domain (TLD)，即：顶级域名。NCU 的 DNS 服务器会将该区域中的名称指定给工程网络中的各个设备。而针对其他所有区域的名称查询将会转送给公司网络上的外部名称服务器。 为避免和全局定义的域名冲突，在缺省设置采用了由 RFC1035 建议的"local"，用于本地网络。通常该缺省设置足够。
PN 站名称	出于 ProfiNet 目的，此处可定义一个单独的机床名称。该名称用在 ERTEC 接口上（如果有的话），分配给 ERTEC 系统程序，也用在 DCP 的网络接口上。
站名称	该字符串用于 SINUMERIK 专用的 OID mcSinumerikMIB.mcSinumerikMiscStatio 。该值是一个设备所属的站的名称，因此，可以整体识别出带有相同站名称的设备。该说明仅用于提供信息。
刀位	该字符串用于标准 OID SNMPv2-MIB::sysLocation。如果有需要，还可以在此处输入一个地点，该地点随后可通过 SNMP 客户端调用。该说明仅用于提供信息。

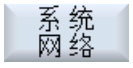
设置	说明
联系人	该字符串用于标准 OID SNMPv2-MIB::sysContact。如果有需要，还可以在此处输入一个联系地址，该地址随后可通过 SNMP 客户端调用。该说明仅用于提供信息。
功能	该字符串用于西门子工业专用的 OID automationSystemIdent.automationFunctionTag。如果有需要，还可以在此处输入一个功能名称，该名称随后可通过 SNMP 客户端调用。该说明仅用于提供信息。
固定域	定义一个额外的 DNS 域名（DNS 请求的后缀）。这特别适用于：当 X120 上的 DHCP 服务器为关闭状态时，因为该服务器的域无法设置为“本地”，但需要用于某些 DNS 请求（如机床控制面板）。
固定的 DNS 服务器	最多确定三个应由 DHCP 服务器使用的 DNS 服务器。这特别适用于：当 X120 上的 DHCP 服务器为关闭状态时，因为该服务器也可以是一个 DNS 服务器或网络中的 DNS 服务器。
路由转发	
• 路由转发 X120/X127 -> X130	确定是否可以从系统网络(X120)转发数据包和/或从服务接口(X127_向公司网络 (X130) 转发数据包： • 使能：激活 X120 和 X127 的转发 • 禁用：取消激活 X120 和 X127 的转发 • X127：只激活 X127 到公司网络的转发 • X120：只激活系统网络向公司网络的转发
• 路由转发 X127 -> X120	激活或关闭从服务接口(X127)向系统网络(X120)进行的数据包 NAT 转发

设置	说明
• 路由转发 X120 -> X127	激活或取消激活从系统网络（X120）向服务接口 X127 转发数据包。这通常会被防火墙禁止。 需要注意的是，此时不进行任何 NAT，发送方必须自己负责以下任务：将包发送到 X127（地址为 192.168.215.x）上编程装置或服务 PC 的数据包首先完整地发送到连接设备的 NCU。
生效的协议	确定系统网络中应使用哪条协议，有“DCP” (Discovery and Basic Configuration Protocol) 和“LLDP”(Link Layer Discovery Protocol) 两种协议可选。在 NCU 上，服务接口 X127 也采用相应的生效协议。

步骤



1. 打开网络一览。



2. 按下软键“系统网络”。
   “系统网络设置”窗口打开。



3. 按下软键“更改”，对设置进行修改。



4. 按下软键“复位 DHCP 服务器”，将 DHCP 服务器的当前设置恢复到初始状态。



5. 按下软键“分配 DHCP 数据”，将设置与设备网络中的所有节点进行调整。

说明

软键“复位 DHCP 服务器”和“分配 DHCP 数据”只能在与激活的 DHCP 服务器连接时使用。

15.3 公司网络设置

窗口“公司网络设置”中列出了公司网络接口的配置，当前生效的协议以及防火墙特例。

设置	说明
网关	如果该值不为空，则其中给出的主机作为缺省网关来使用，也就是说，所有不能直接递送的 IP 包都会发送到那里用来转发。
DNS 服务器	如果在此处给定了 DNS 名称服务器（最多 3 个），则该服务器用于分辨主机符号名称，即：在大多数需要 IP 地址的位置上也可以使用主机名称。 名称服务器设置也会通过 NCU 的 DHCP 服务器传递到 DHCP 用户 (TCU, PG)，从而使得这些用户也能够用符号名称来工作。
时间服务器	在此处最多可以输入 3 个 NTP 服务器 (UDP/123)，NCU 上的 NTPD 通过此服务器进行时间同步。
主机名	此处可以确定一个本地主机的名称。该手动给出的名称相对于其他所有名称有优先权。 主机名依照以下顺序来决定： • 该条目（与文件 basesys.ini 中的主机名一致） • 一个由 DHCP 服务器获取的名称（反向进行 DNS 查找，即哪个名称属于哪个获取的 IP 地址） • 缺省名称 ("NONAME_...") 由于主机名也用作 DNS 名称，因此它必须满足相应的 RFC 要求： • ASCII 字母(a-z, A-Z)、数字(0-9)和“-” • 最多 63 个字符
生效的协议	确定公司网络中应使用哪条协议，有 DCP (Discovery and Basic Configuration Protocol) 和 LLDP (Link Layer Discovery Protocol) 两种协议可选。
防火墙例外	此处可以输入在外部网络接口的防火墙中使能的端口。

设置	说明
• S7 通讯(TCP/102)	S7 通讯端口
• VNC 访问(TCP/5900)	VNC 访问端口
• 更多端口	此处可以使能更多端口。此时，先输入协议名称（TCP 或 UDP），然后输入斜杠 '/' 和待使能的服务的数字端口号。多个端口数据用空格隔开。 示例（使能 VNC 服务器）：TCP/5900 TCP/5904 TCP/5905

注意
安全风险 请注意，打开防火墙的端口潜在有安全隐患。请只有在确实需要的情况下才打开防火墙端口。 请注意，每个可访问的服务也可能有安全漏洞。

步骤



1. 打开网络一览。



2. 按下软键“公司网络”。
   “公司网络设置”窗口打开。



3. 按下软键“更改”，对设置进行修改。

15.4 保存网络设置

可以将 NCU 预设置保存为文件。

在窗口“保存网络设置：选择保存位置”中选择一个目标文件夹，将网络配置保存在该文件夹中，或在该文件夹中选中一个已有文件，用新设置覆盖该文件。

步骤



1. 网络一览已打开，您已完成了网络设置。



2. 按下软键“保存 NCU 设置”。
   窗口“保存网络设置：选择保存位置”打开。
3. 选择用于保存网络配置的文件夹。如果已经有一个文件，则将其覆盖掉。
   -或者-
   按下软键“新建目录”，创建新目录。
   -或者-



   按下软键“搜索”，找到网络配置文件的保存位置。



4. 如果找到的文件不是所需的配置文件，则按下软键“继续搜索”。



5. 按下软键“取消”，放弃设置。
   -或者-



   按下软键“确认”，保存网络配置。

15.5 配置操作面板

15.5.1 显示可用的操作面板

在窗口**...的可用操作面板**的顶部会显示一张列明可使用的所有操作面板的列表。一行对应一个操作面板且包含以下信息：

列	说明
	用于识别操作面板的流水号
状态	操作面板状态：  ：该操作面板当前可在总线上使用。 ：该操作面板当前不可在总线上使用。
硬件	操作面板硬件的信息 允许的值有：“未知”（无可用信息），NCU、PCU、PPU、TCU（包括 HT8 等在内的所有 TCU）。 默认值：未知

列	说明
名称	操作面板名
MCP	机床控制面板 (Machine Control Panel) 的 DIP 开关位置 可在 1 到 254 之间任意选择值（0 表示保留最后一个生效的机床控制面板，255 表示不使用任何机床控制面板）。 默认值：192
DCK	直接键 (Direct Control Keys) 的可用性 生效：TCU 具有直接键 无效：TCU 不具有直接键
TCU	TCU (Thin Client Unit) 的序号 该值可在 0 到 255 之间任意配置并用于在 PLC 中识别 TCU。
EKS	访问等级芯片阅读器 (Electronic Key System) 的 DIP 开关位置 允许值在 1 到 255 之间。值 0 表示没有使用任何 EKS。
虚拟键盘	使用虚拟键盘 允许的值为：“从未”、“始终”和“自动”，其中“自动”设为默认值。

说明

操作面板的信息只可读，无法在**操作面板**窗口中修改。

更新一览

在切换到**操作面板**窗口时，系统会自动搜索可用的操作面板。该搜索也可在日后通过软键“显示新面板”或“重新加载 OP 数据”进行。后者会重新进行一次完整的扫描，将现有的操作面板从本地缓存中删除。加载时，系统只会搜索新的操作面板和数据。

生效的（操作面板数据所在的）DHCP 服务器可以是另一个 NCU，也可以是本地 NCU。不管是哪个，系统都会在本地 NCU 上创建一个数据副本并使用该副本。

步骤



1. 选择操作区域“调试”。



2. 按下软键“操作面板”。



“操作面板”窗口打开。

系统显示所有当前可用的操作面板。



3. 按下“刷新”软键，刷新一览。

此时，系统会搜索新的操作面板和数据。



4. 按下软键“重载操作面板数据”，以重新进行完整的搜索。

此时，系统会将当前的操作面板从缓冲存储器/缓冲器中删除。



5. 按下软键“分配 TCU 数据”，将可用的操作面板与设备网络中的所有节点进行调整。

说明

软键“分配 TCU 数据”只能在与激活的 DHCP 服务器连接时使用。

15.5.2 配置 VNC 连接

在“...的可用操作面板”窗口的底部中会显示一个 VNC 连接的列表，其中包含以下信息：

列	说明
	用于识别 VNC 连接的流水号
VNC 服务器	IP 地址或 NC 名
替换	确定是否允许其他的节点替换 TCU。
/Prio	替换优先权：确定影响是否可以用其他的节点替换 TCU 的优先权。 允许值位于 0 到 10 之间。值越小，就越可能替换 TCU。如果替换被禁用，系统会忽略该优先权。
启动	确定 TCU 是否生效，即是否作为操作站启动。
/Prio	启动优先权：确定优先权，它会影响以怎样的顺序使用 TCU 连接。 允许值从 0 开始。值越大，系统就越可能会复位连接顺序。
屏幕	确定在激活 TCU 时系统会显示哪块 HMI 屏幕。
通道	确定在激活 TCU 时系统会显示哪个通道。

编辑 VNC 连接列表



1. “...的可用操作面板”窗口打开。



2. 按下软键“更改”。
   现在可以编辑 VNC 连接的信息。

-或者-



3. 按下软键“添加”，以创建一个新的 VNC 连接。

-或者-



4. 按下软键“删除”，从列表中删除一个选中的 VNC 连接。

-或者-





5. 按下软键“上移”或“下移”，在列表内移动 VNC 连接。



6. 进行所需的修改并按下软键“确认”。

-或者-



按下软键“取消”，放弃修改。

参见

显示可用的操作面板 (页 268)

15.5.3 T:M:N 配置

T:M:N 配置指的是由 t 个操作站 (TCU 带 MCP)，m 个 HMI 系统和 n 个 NCK 系统组成的一个组合，其中操作站在运行模式下可以自由切换到任何一个 HMI 系统上，以便进行操作。 对于各个操作站的切换目标，针对每个操作站单独以一个二级步骤在通道菜单中设计切换目标。

配置了一个通道菜单后，后，从 HMI 角度度而言，T:M:N 运行便已存在。 切换目标可直接通过 HMI 地址和可选的一个 NC 通道（该通道在切换 HMI 时加入）规定。 切换目标可以逻辑方式，即所谓的通道组合并在一起。通道组在通道菜单中的水平软键上显示，真正的切换目标 (HMI, NC 通道) 显示在通道组的垂直软键上。

软键和通道组的互联

系统会在底部窗口中显示一个 HMI 软键和所设计的通道组的互联列表。通道组可以再次包含执行操作站功能的切换目标。一个通道组（如 CH_GRP_0）最多可以和 32 个软键（SK_1 到 SK_32）互联。

步骤



1. “...的可用操作面板”窗口打开。



2. 选中所需的操作站并按下软键“T:M:N”。
   系统会在底部窗口中显示一个 HMI 软键和所设计的通道组的互联列表。



3. 按下软键“更改”。
   现在可编辑列表。



4. 进行所需的修改并按下软键“确认”。

   -或者-



   按下软键“取消”，放弃修改。

参见

显示可用的操作面板 (页 268)

15.6 基于子站的网络诊断

借助基于子站的网络诊断功能，您可以找到失灵、设置错误或没有接入的组件。

系统会向您显示设备网络组件的一览：

- NCU
- PCU
- TCU
- MCP
- EKS

在一览中，失灵、根本不存在或者无法访问的组件会被标出。

故障分析

您可以开展针对出错组件的分析和诊断。 诊断结果以一条故障信息（报告）的形式提供给您。 这条信息中描述了可能出现的故障状态、原因，并提供故障应对措施。

15.6.1 显示网络适配器

在窗口“TCP/IP 诊断”中，以树形图的形式显示了当前的网络适配器及其可用性（设备网络 X120、公司网络 X130 或者设备网络 ETH2 和公司网络 ETH1）。

组件的显示

窗口中显示的组件有：

- 控制系统，位于最上面一级
- 设备网络(X120 / ETH2)，及其操作站、IP 地址
  - 操作面板及其 IP 地址
  - 机床控制面板
  - EKS
- 公司网络 (X130 / ETH1)，其中有：
  - 操作面板及其 IP 地址
  - 机床控制面板
  - EKS

无法访问的组件



无法访问的组件会标有该符号。

输入权限

具有输入权限的操作面板显示为绿色。

详细信息显示

可显示所选组件的以下信息。

TCU

- IP 地址
- 软件版本
- 配置中的 MCP 索引
- 配置中的 TCU 索引

- DNS 名称
- 分辨率

MCP

- IP 地址
- DNS 名称
- 配置中的 MCP 索引
- HMI 请求的 MCP 索引
- PLC 请求的 MCP 索引

步骤



1. 选择操作区域“诊断”。



2. 按下菜单扩展按键和软键“TCP/IP 总线”。





3. 按下软键“ TCP/IP 诊断”。
   窗口“TCP/IP 诊断”打开，显示网络连接的当前可用性。
4. 需要了解组件的详细信息时，将光标移动到该组件上。



5. 按下软键“详细”，显示配置的网络连接的所有可用参数。



6. 按下软键“详细”，退出详细信息画面。



7. 按下软键“返回”，退出窗口“TCP/IP 诊断”，回到窗口“PROFIBUS 诊断”。

15.6.2 故障分析

15.6.2.1 开展故障诊断

前提条件

故障分析、诊断只针对一个 PCU。

步骤





1. 打开“TCP/IP 诊断”窗口。
2. 将光标移动到标为故障的组件上（NCU）。
3. 按下软键“故障分析”。
   对应的故障信息弹出。

15.6.2.2 无法访问 MCP/EKS 网络节点

错误	可能的故障原因和排除方法
无法访问 MCP/EKS 网络节点。	
	故障原因 1
	没有物理网络连接。 **注意：** 这条信息针对的是 MCP/EKS 和下一个节点的连接失败的情况，也就是： • MCP/EKS 的以太网电缆没有插入， • 下一个节点上的以太网电缆没有插入，或该节点不可用，如 MCP， • MCP/EKS 的以太网电缆损坏。
	排除方法
	请建立物理网络连接（检查插接情况、更换电缆、检查连接节点）。
	故障原因 2
	没有下一个节点到设备网络的物理网络连接。 MCP/EKS 和具有有效 DHCP 服务器的计算机之间没有物理连接。 可能是电缆损坏、没有插入，或者没有接通开关。 开关可以位于 MCP 或配电盒上。
	排除方法

错误	可能的故障原因和排除方法
	请建立物理网络连接（检查插接情况、更换电缆、检查连接节点）。 连接成功后，TCU 继续启动。
无法访问 MCP/EKS 网络节点。 存在到设备网络的物理网络连接。	
	故障原因
	在设备网络中，没有找到有效的 DHCP 服务器。 可能的原因： • 在设备网络中，只配置了一台计算器具有激活的 DHCP 服务器，而这台计算器不可用。 设备网络无法和正在同步的 DHCP 服务器运行。 • 在设备网络中，没有配置具有激活的 DHCP 服务器的计算机。 在所有的潜在计算机(NCU, PCU)上，DHCP 模式被关闭。 这是一个配置错误。 • 在设备网络中，没有 NCU 或 PCU 类型的计算机可用，该计算机具有激活的 DHCP 模式，并可能成为服务器。
	排除方法
	在设备网络中配置一台可以作为 DHCP 服务器运行的计算机。

15.6.2.3 无法建立 NC 和 PCU 上的 HMI 之间的网络连接

错误	故障原因 1	排除方法
HMI 弹出报警 120202“等待和 NC/PLC 建立连接”，或者根本不显示数值，只显示井号。	在 PCU 上设置的 目标 NCK/PLC 的 IP 地址错误。	请在 HMI 上修改 NCK/PLC 的目标地址，并重新启动 PCU。

错误	故障原因 2.1	排除方法
HMI 弹出报警 120202“等待和 NC/PLC 建立连接”，或者根本不显示数值，只显示井号。	PCU 的 IP 地址无效或者无法被网络识别。 在 PCU 上取消了“TCU 支持”，也就是取消了 DHCP，并且 PCU 采用无法由网络识别的固定 IP 地址运行，或者和标准的 Windows DHCP 客户端一起运行，动态分配地址。 这是一个配置错误。	在 Windows 的服务模式中正确设置系统网络：为设备网络设置固定 IP 地址和子网掩码，地址分配模式采用固定 IP 地址。 借助“System Network Center”程序可以： • 将 DHCP 服务连接到系统网络接口上， • 设置 DHCP 模式（TCU 支持、DHCP 同步模式、可分配地址的范围）， • 开始“DHCP-Clear”。

错误	故障原因 2.2	排除方法
HMI 弹出报警 120202"等待和 NC/PLC 建立连接"，或者根本不显示数值，只显示井号。	PCU 的 IP 地址无效或者无法被网络识别 在 PCU 上激活了"TCU 支持"，也就是激活了 DHCP；并且 PCU 是 DHCP 客户端；或者 PCU 是激活的 DHCP-/DNS 服务器（当 DHCP 同步模式为"ON_MASTER"，或者它的同步等级和其他备选服务器相同或者偶然选中时）。 由于软件错误或当前的网络配置，PCU 无法获得有效的 IP 地址，例如："0 地址"。	在 Windows 的服务模式中正确设置系统网络：为设备网络设置固定 IP 地址和子网掩码，地址分配模式采用固定 IP 地址。 借助"System Network Center"程序可以： • 将 DHCP 服务连接到系统网络接口上， • 设置 DHCP 模式（TCU 支持、DHCP 同步模式、可分配地址的范围）， • 开始"DHCP-Clear"。 另外，还可以在具有 DHCP 主站的工作站上开始"DHCP-Clear"（DHCP 同步模式 = ON_MASTER）。

错误	故障原因 2.3	排除方法
HMI 弹出报警 120202"等待和 NC/PLC 建立连接"，或者根本不显示数值，只显示井号。	PCU 的 IP 地址无效或者无法被网络识别 在 PCU 上激活了"TCU 支持"，也就是激活了 DHCP， PCU 是激活的 DHCP/DNS 服务器（当 DHCP 同步模式为"ON_MASTER"，或者它的同步等级和其他备选服务器相同或者偶然选中时）。 PCU 的 IP 地址错误，并且设置了一个错误的动态分配 IP 地址范围。 由于具有 DHCP-MASTER 的计算机失灵或关闭，PCU 可能从 DHCP 待机状态切换为激活的服务器。	在 Windows 的服务模式中正确设置系统网络：为设备网络设置固定 IP 地址和子网掩码，地址分配模式采用固定 IP 地址。 借助"System Network Center"程序可以： • 将 DHCP 服务连接到系统网络接口上， • 设置 DHCP 模式（TCU 支持、DHCP 同步模式、可分配地址的范围）， • 开始"DHCP-Clear"。 另外，还可以在具有 DHCP 主站的工作站上开始"DHCP-Clear"（DHCP 同步模式 = ON_MASTER）。

错误	故障原因 2.4	排除方法
HMI 弹出报警 120202“等待和 NC/PLC 建立连接”，或者根本不显示数值，只显示井号。	PCU 的 IP 地址无效或者无法被网络识别 在 PCU 上激活了“TCU 引导支持”，也就是激活了 DHCP，PCU 是“DHCP/DNS 待机服务器”，也就是 DHCP 客户端。 设备网络接入了另一个处于运行状态的激活 DHCP 服务器，而它的地址范围设置错误。 这是一个配置错误。	在 Windows 的服务模式中正确设置系统网络：为设备网络设置固定 IP 地址和子网掩码，地址分配模式采用固定 IP 地址。 借助“System Network Center”程序可以： • 将 DHCP 服务连接到系统网络接口上， • 设置 DHCP 模式（TCU 支持、DHCP 同步模式、可分配地址的范围）， • 开始“DHCP-Clear”。 另外，还可以在具有 DHCP 主站的工作站上开始“DHCP-Clear”（DHCP 同步模式 = ON_MASTER）。 重新启动整个设备。

错误	可能的故障原因和排除方法
HMI 弹出报警 120202“等待和 NC/PLC 建立连接”，或者根本不显示数值，只显示井号。	
	故障原因 3.1
	NCU 的 IP 地址无效或者和网络不匹配。 NCU 不参与 DHCP 模式，而是以不和设备网络配套、错误的固定 IP 地址运行。
	排除方法 3.1

错误	可能的故障原因和排除方法
	将服务 PG 通过 X127 连到 NCU。 通过 WinSCP 或 Putty 登录到 Linux。 通过服务指令“sc show ip -eth0”显示当前设置，然后检查设置。 确认故障原因，重新设置 NCU。 执行以下服务指令和设置： • “sc enable DHCPSvr –X120”或者相同作用的“sc enable DHCPSvr –eth0”，激活 DHCP 模式 • “sc enable DHCPSync –X120 -LOW|HIGH|MASTER”或者相同作用的“sc enable DHCPSync –eth0 -LOW|HIGH|MASTER”，设置该 NCU 的 DHCP 同步模式 在文件“/user/system/etc/basesys.ini”中正确输入以下条目： [InternalInterface] ; InternalIP 和 InternalNetMask 必须一起设置， ; 您可以修改内部/TCU/自动化/系统网络上的地址。 InternalIP=192.168.214.1 InternalNetMask=255.255.255.0 ... ;动态 IP 地址范围，由 DHCP 服务器指定 ; 缺省值是合理值 InternalDynRangeStart=192.168.214.10 InternalDynRangeEnd=192.168.214.239 应按照整个设备来设置绿色数值。 然后重新启动 NCU。
	故障原因 3.2
	NCU 的 IP 地址无效或者和网络不匹配。 NCU 不参与 DHCP 模式，NCU 是激活的 DHCP-/DNS 服务器（当 DHCP 同步模式为“ON_MASTER”，或者它的同步等级和其他备选服务器相同或者偶然选中时）。 NCU 的 IP 地址错误，并且设置了一个错误的动态分配 IP 地址范围。 由于具有 DHCP-MASTER 的计算机失灵或关闭，NCU 可能从 DHCP 待机状态切换为激活的服务器。
	排除方法 3.2 和排除方法 3.1 相同
	故障原因 3.3
	NCU 的 IP 地址无效或者和网络不匹配。 NCU 参与 DHCP 模式，以错误的 IP 地址运行。 在 NCU 启动后，另一个处于运行、具有激活 DHCP 的服务器的组件（NCU 或 PCU）连接到网络中。 设备网络接入了另一个处于运行状态的激活 DHCP 服务器，而它的地址范围设置错误。 这是一个配置错误。

错误	可能的故障原因和排除方法
	排除方法 3.3 和排除方法 3.1 相同
HMI 弹出报警 120202“等待和 NC/PLC 建立连接”，或者根本不显示数值，只显示井号。	
	故障原因 4.1
	NCU 的 IP 地址在设备网络内有效，但是不是期望的 IP 地址。 NCU 不参与 DHCP 模式，它的 IP 地址设置错误、和 MMC.INI 中定义的目标地址不同，但是在设备网络内却是有效地址。 这是一个配置错误。
	排除方法 4.1
	将服务 PG 通过 X127 连到 NCU。 通过 WinSCP 或 Putty 登录到 Linux。 需要保持设置的 NCU 模式时（无 DHCP），必须修改该 NCU 的固定 IP 地址。 在文件“/user/system/etc/basesys.ini”中正确输入以下条目： [InternalInterface] ; With InternalIP and InternalNetMask (both must be set together), ; you can change the address on the internal/TCU/automation/ ystem net. InternalIP=192.168.214.1 InternalNetMask=255.255.255.0 然后重新启动 NCU。
	故障原因 4.2
	NCU 的 IP 地址在设备网络内有效，但是不是期望的 IP 地址。 NCU 不参与 DHCP 模式，它的 IP 地址设置错误、和 MMC.INI 中定义的目标地址不同，但是在设备网络内却是有效地址。 这是一个配置错误。
	排除方法 4.2 和排除方法 4.1 相同
	故障原因 4.3
	NCU 的 IP 地址在设备网络内有效，但是不是期望的 IP 地址。 NCU 参与 DHCP 模式，但是不能保持和 MMC.INI 中定义的目标地址相符的期望 IP 地址，而是动态分配到一个地址。 原因可能是：期望的 IP 地址在设备网络内重复配置，或者不位于预留的地址范围，而是在动态地址范围内。 这是一个配置错误。
	排除方法 4.3 和排除方法 4.1 相同
	故障原因 4.4

错误	可能的故障原因和排除方法
	NCU 的 IP 地址在设备网络内有效，但是不是期望的 IP 地址。 NCU 作为 DHCP 客户端参与 DHCP 模式，设置的期望 IP 地址不和设备网络匹配。例如，在设备网络内修改了缺省网络地址，但是忘记了将 NCU 的期望 IP 地址修改到该地址范围中，便会出现该情况。 NCU 动态分配到一个和设备网络匹配的 IP 地址，但是和它原先未经修改的旧期望 IP 地址不同。 对于低于等于 2.4.1 版本的软件来说，在这种情况下，设备网络中的 NCU 不会发出响应，因为它内部已经建立了一道防火墙针对输入的不符合设备网络的期望IP地址，因此不会在X120上应答。
	排除方法 4.4 和排除方法 4.1 相同
HMI 弹出报警 120202“等待和 NC/PLC 建立连接”，或者根本不显示数值，只显示井号。	
	故障原因 5.1
	没有物理连接。 可能的原因有： • 开关没有接通或者失灵 ，这个开关也可能是 MCP 的开关 • 电缆松脱（接触不良） • 电缆错误（交叉型/不交叉型） • 一段电缆损坏
	排除方法 5.1
	建立物理连接。

15.6.2.4 无法建立 TCU 和 HMI 之间的网络连接

错误	可能的故障原因和排除方法
TCU 在启动时报告“ERROR: Network connection not established”。	
	故障原因
	没有物理连接。 注意： 这条信息针对的是 TCU 和下一个节点的连接失败的情况，也就是： • TCU 的以太网电缆没有插入， • 下一个节点上的以太网电缆没有插入，或该节点不可用，如 MCP， • TCU 的以太网电缆损坏。
	排除方法

错误	可能的故障原因和排除方法
	请建立物理网络连接（检查插接情况、更换电缆、检查连接节点）。 连接成功后，TCU 继续启动。
TCU 在启动时报告“onnection established – no DHCPServer available”。	
	故障原因
	没有下一个节点到设备网络的物理网络连接。 TCU 和具有有效 DHCP 服务器的计算机之间没有物理连接。 可能是电缆损坏、没有插入，或者没有接通开关。 开关可以位于 MCP 或配电盒上。
	排除方法
	请建立物理网络连接（检查插接情况、更换电缆、检查连接节点）。 连接成功后，TCU 继续启动。
TCU 在启动时报告“onnection established – no DHCPServer available”。 存在到设备网络的物理网络连接。	
	故障原因
	在设备网络中，没有找到有效的 DHCP 服务器。 可能的原因有： • 在设备网络中，只配置了一台计算器具有激活的 DHCP 服务器，而这台计算器不可用。 设备网络无法和正在同步的 DHCP 服务器运行。 • 在设备网络中，没有配置具有激活的 DHCP 服务器的计算机。 在所有的潜在计算机 (NCU, PCU)上，DHCP 模式被关闭。 这是一个配置错误。 • 在设备网络中，没有激活了 DHCP 模式且可作为服务器的 NCU 或 PCU 类型的计算机可用。
	排除方法
	在设备网络中配置一台可以作为 DHCP 服务器运行的计算机。
TCU 在启动后显示“Waiting for HMI”（即 TCU 无法建立与 HMI 的连接）。	
	故障原因 1
	和 HMI 的连接，无论是在“config.ini”中定义的连接还是缺省连接，都没有成功。 在文件“config.ini”中设置了错误的启动后连接到 HMI 的地址。
	排除方法
	借助程序“System Network Center”正确设置配置。 TCU 自动重启。
	故障原因 2
	和 HMI 的连接，无论是在“config.ini”中定义的连接还是缺省连接，都没有成功。 “config.ini”中定义的、TCU 启动后应该接入的计算机无法访问，因为该计算机没有接通或不可用。
	排除方法

错误	可能的故障原因和排除方法
	重新启动计算机（NCU 或 PCU ）。
	故障原因 3
	和 HMI 的连接，无论是在“config.ini”中定义的连接还是缺省连接，都没有成功。“config.ini”中定义的、TCU 在启动后应该接入的计算机上的 HMI 无法访问，因为该 HMI 被锁定。
	排除方法
	在 NCU 上通过服务指令“sc enable HMI”激活 HMI。
	故障原因 4
	和 HMI 的连接，无论是在“config.ini”中定义的连接还是缺省连接，都没有成功。“config.ini”中定义的、TCU 在启动后应该接入的计算机上的 HMI 无法访问，因为软件错误导致该 HMI 无法启动或崩溃。
	排除方法
	重新启动计算机（NCU 或 PCU ）。

15.6.2.5 无法建立 TCU 和指定 PCU 的网络连接

错误	可能的故障原因和排除方法
PCU 的 IP 地址不是由配置定义，而是在设备网络内动态分配。 在这种错误下，TCU 无法访问定义的、用于显示的 PCU。	
	故障原因 1
	IP 地址重复分配 PCU 具有缺省的 IP 地址，但它也分给了另一个 NCU/PCU。 因此出现重复分配的 IP 地址，出现设备配置错误。 PCU 不是激活的 DHCP 服务器（否则它自己有一个期望 IP），但是却以正在同步的 DHCP 模式运行。 由于 PCU 的目标地址已被分配或占用，因此 PCU 通过 DHCP 动态分配到一个地址。
	排除方法

错误	可能的故障原因和排除方法
	请确定哪一个计算机的 IP 地址被重复分配。 如果是 PCU，便 • 在 PCU 的“网络设置”下，重新将该地址设为设备网络内的固定 IP， • 在以 DHCP 同步模式“ ON_MASTER”运行的 NCU/PCU 上，执行“Clear DHCP”；NCU 上采用 sc 指令“sc clear dhcp –X120”，而 PCU 上采用程序“System Network Center”。 • 在重复分配 IP 地址的第二个计算机上，重新配置 IP 地址。 注意： 如果这台计算机是 PCU，则通过程序“System Network Center”将它连接到新地址（程序中的第一幅画面）。 如果需要修改 PCU 的地址，请在 PCU 上 • 在“网络设置”下将新地址设为设备网络内的固定 IP， • 通过程序“System Network Center”连接到新地址（程序中的第一幅画面）。 • 在以 DHCP 同步模式“ ON_MASTER”运行的 NCU/PCU 上，执行“Clear DHCP”；NCU 上采用 sc 指令，而 PCU 上采用程序“System Network Center”。 重新启动设备。
	故障原因 2
	PCU 硬件被更换。 由于更换了硬件，因此虽然新 PCU 的设置相同，但由于系统原因，所需的 IP 地址在内部仍然预留用于旧硬件。 PCU 是激活或未激活的 DHCP 服务器，但是以同步的 DHCP 模式运行。 由于 PCU 的目标地址仍然预留用于旧硬件，因此 PCU 通过 DHCP 动态分配到一个地址。
	排除方法
	执行以下步骤： • 在 PCU 的“网络设置”下，重新将目标地址设为设备网络内的固定 IP。 • 在以 DHCP 同步模式“ ON_MASTER”运行、或者是激活的 DHCP 服务器的 NCU/PCU 上，执行“Clear DHCP”；NCU 上采用 sc 指令“sc clear dhcp –X120”，而 PCU 上采用程序“System Network Center”。 • 重新启动设备。
	故障原因 3
	PCU 配置了一个无法由网络识别的地址。 PCU 的 IP 地址和设备网络不匹配，因此它通过 DHCP 从激活的服务器处动态分配到一个 IP 地址。 PCU 不是激活的 DHCP 服务器，仍然以正在同步的 DHCP 模式运行。

错误	可能的故障原因和排除方法
	排除方法
	执行以下步骤： • 在 PCU 的“网络设置”下，重新将目标地址设为设备网络内的固定 IP。 • 请通过 PCU 上的程序“System Network Center”将 DHCP 客户端连接到这个新地址上（程序“System Network Center”中的第一幅画面），必要时还需要修改 DHCP 的其他设置（动态地址范围）。 • 在以 DHCP 同步模式“ ON_MASTER”运行、或者是激活的 DHCP 服务器的 NCU/PCU 上，执行“Clear DHCP”；NCU 上采用 sc 指令“sc clear dhcp –X120”，而 PCU 上采用程序“System Network Center”。 • 重新启动设备。

15.6.3 配置 TCP/IP 诊断

窗口“TCP/IP 配置”中显示了网络适配器的配置。

设备网络

在设备网络中，由各组件通过操作软件与显示单元（TCU）进行过程数据通讯和画面传输。

TCU 和 NCU 通过以太网接口 X120 连接到设备网络中。

PCU 则通过以太网适配器 ETH2 接入。

公司网络

通过公司网络可以访问网络驱动器。

NCU 通过以太网接口 X130 连接到公司网络中。

PCU 则通过以太网适配器 ETH1 接入。

网络连接的可用性

网络适配器接口



白色　　网线已插入



红色　　网线未插入

可用性

可用性指在所有已发送和接收的数据包中，错误数据包所占的百分比。

公司网络中出现的故障，例如：无法访问逻辑驱动器，IP 地址重复等，以及启动中的起振时间可能会影响可用性的稳定性。

	绿色	大于 95 %
	黄色	50 - 95 %
	红色	小于 50 %

网络连接的详细信息

- **计算机名称**
- **DNS 名称**
  控制系统的计算机名称，正如保存在 DNS 服务器(**D**omain **N**ame **S**ervice) 上的名称。
- **MAC 地址**
  网络适配器的物理地址
- **地址类型**
  配置文件 basesys.ini 中网络适配器配置的相关信息：
  - DHCP:DHCP 对该网络适配器生效。
    连字符"-"后的数据说明了 DHCP 的工作方式：
    关 - DHCP 不对该适配器生效
    客户端 接口上运行的 DHCP 客户端，它由服务器获取 IP 地址和其它数据。
    服务器 - 在该接口上，NCU 提供 DHCP 服务器，它为该网络上的客户端提供 IP。
    同步服务器 多个 NCU 同步运行的协议生效。它可以确保只有其中的一个 NCU 作为 DHCP 服务器运行并且不会分配重复的 IP
    地址。
    如果一个网络适配器作为"同步服务器"运行，则在"同步 DHCP 服务器状态"栏中还显示其他信息。
    如果在配置文件中没有对网络适配器进行任何修改，则还输出信息"默认"。
  - 手动
    IP 地址、子网掩码、DNS 服务器 1、DNS 服务器 2、网关的设置都在"basesys.ini"中进行。
    提示：
    在"修改"模式中可以切换"手动"和"DHCP"（仅适用于公司网络：X130）。
- **分配的 IP 地址**
  网络适配器的当前 IP 地址
- **分配的子网掩码**
- **DHCP 服务器**
  网络适配器的 IP 地址（地址类型"DHCP"）。
- **DHCP 服务器状态**
  同步 DHCP 服务器的状态：
  - 有效（在所有的 NCU 中，当前 NCU 处于模式"同步服务器"，分配 IP 地址）；
  - 待机（NCU 不处于当前有效状态，但如果有效的服务器失灵，该 NCU 会成为服务器）。
- **DHCP 同步方式**
  同步 DHCP 服务器的优先级：
  低、高、主机

- **DNS 服务器 1, DNS 服务器 2**
  域名服务器
- **网关**
- 提示
  所有不存在的信息由表格项中的连字符"-"表明。

可修改的参数

通过选择"地址类型"可以设定公司网络适配器(X130)的以下参数：

- 地址类型
- IP 地址
- 子网掩码
- DNS 服务器 1 和 2
- 网关（适用于设备网络和公司网络）

地址类型

- DHCP
  DHCP 服务器根据"basesys.ini"中的条目自动设定 IP 地址、子网掩码、DNS 服务器 1、DNS 服务器 2 和网关的可设定参数值。
- 手动
  可以手动设定 IP 地址、子网掩码、DNS 服务器 1、DNS 服务器 2 和网关的可设定参数值，这些值随后保存到"basesys.ini"中。

步骤



1. 选择操作区域"诊断"。



2. 按下菜单扩展按键和软键"TCP/IP 总线"。





3. 按下软键" TCP/IP 诊断"和"TCP/IP 配置"。
   "TCP/IP 诊断"窗口打开。





4. 按下“更改”软键，修改特定参数。
可被编辑的栏显示为白色背景。



5. 输入所需的修改并按下软键“确认”，结束输入。



6. 按下“刷新”软键，刷新显示。

15.6.4 以太网诊断

15.6.4.1 网络诊断和 DHCP 服务器

按下软键“网络诊断”，屏幕上即可显示 Ethernet 上每个节点的数据和可用性，使调试工作更加高效。 诊断结果中包含了以下信息：

- 硬件 ID 号
- TCP/IP 地址/ MAC 地址
- DHCP 服务器 ID
- 节点状态

您可以检查实际拓扑结构和设定拓扑结构是否一致。

步骤



1. 选择操作区域“诊断”。



2. 按下菜单扩展按键和软键“TCP/IP 总线”。





3. 按下软键“ TCP/IP 诊断”。
窗口“TCP/IP 诊断”打开，显示网络连接的当前可用性。



4. 按下软键“网络诊断”，扫描整个网络。
“网络状态”窗口打开，显示设备网络 HMI(X120)、设备网络 PN (X150)和公司网络(X130)的地址。
网络无法访问时，窗口中对应的一栏显示为空。



5. 按下“刷新”软键，刷新显示。

-或者-



按下软键“DHCP 服务器”，显示 DHCP 服务器的信息和状态。

“DHCP 服务器”窗口打开。

窗口上方以列表形式显示了每个激活的 DHCP 服务器的状态、角色（优先级）、MAC 地址、TCU 版本和租赁版本。 窗口下方显示了目标 IP 及其地址范围。

和目标不符的数据标为红色。



6. 按下软键“地址列表”，查询当前 DHCP 服务器的所有信息。

窗口“地址列表”打开。

屏幕上以表格形式显示了每个节点的 IP 地址、客户端 ID（MAC 地址或 CF 卡号）、租赁版本、设备编号、DNS 名称及附加信息。

未激活的节点用灰色表示。

15.6.4.2 信息

通过软键“信息”可以显示日志文件“syslog”中所有和系统相关的状态信息和故障信息。

步骤



1. “网络状态”窗口打开。



2. 按下软键“信息”。

“信息”窗口打开。



3. 按下软键“全部显示”，“DHCP”，“MCP”或“TCU”以过滤显示。







15.6.4.3 网络跟踪

发生网络问题时，可以将数据传输通过“网络追踪”窗口作为日志文件记录下来，以便后续分析和评估。

一般说明

- 该功能设有口令保护，默认口令为 NETWORK。

  说明
  请修改该默认口令，以防滥用。

- 记录过程由程序 tcpdump (Linux) 或 windump (Windows) 完成。
  更多信息请参见 www.tcpdump.org。

前提条件

修改口令要求访问级别 2（服务）。

步骤



1. “网络状态”窗口打开。



2. 按下软键“网络追踪”。
   打开窗口“设定口令”。



3. 输入口令并按下软键“确认”，结束输入。
   “网络跟踪”窗口打开。



4. 如果参数保存在文件中，按下软键“从文件载入”，载入 USB 等存储器中的文件。
   “选择文件”窗口打开。



5. 从目录中选择文件并按下“确认”。

-或者-

4. 如果需要，可在输入栏“参数(tcpdump)”中设置记录条件。



5. 按下软键“开始”，进行记录。
   提示：记录在 20 分钟后自动停止。



6. 按下软键“停止”，结束记录。
   “导出跟踪图”窗口打开。



7. 选择存储位置，输入文件名称并按下“确认”。
   跟踪文件被移至存储位置中。按下软键“取消”，记录则仍保存在以下目录下：
   HMI 数据/记录/网络跟踪/tcpdump.pcap bzw. windump.pcap



8. 如果记录已经存在，按下软键“导出跟踪图”，将当前记录文件存储到 USB 等存储器上。

   “选择存储位置”窗口打开。



9. 选择存储位置，输入文件名称并按下“确认”。

修改口令

1. “网络跟踪”窗口打开。



2. 按下软键“修改口令”。

   打开窗口“修改口令”。



输入新口令并按下“确认”键确定。

跟踪参数

通过网络跟踪窗口中的输入栏“参数(tcpdump)”可自行设置记录条件。

句法

<开关><选项>

参数<开关>

值	含义
-D	显示所有网络接口的列表。 忽略其他所有<开关>参数。
-i<接口>	定义哪些网络接口需检查。
-n	主机名称未分解。
-c <数量>	在指定数量的数据包后结束记录。
-C <文件大小>	文件大小受限。 如果达到文件大小上限，系统会创建一个含名称和连续数字（从 1 开始）的新文件。 文件大小的单位为 MB。
-e	输出以太网标题。
-q	生成记录摘要。
-vv	生成详细的数据包信息。
-S	数据包序号从绝对号变为相对号。
-A	输出 ASCII 格式的数据包内容。
-xx	输出十六进制格式的数据包内容。

值	含义
-XX	同时输出十六进制和 ASCII 格式的数据包内容。
-s <字节>	指定必须记录的每个数据包的大小（字节数）。 标准为 68 字节。

说明

调用中始终隐含值-w，用于生成标准记录文件：

HMI 数据/记录/网络追踪/tcpdump.pcap bzw. windump.pcap

参数<选项>

选项	值	含义
host	IP 地址或主机名称	只记录特定数据包，该数据包作为数据源或目标包含指定 IP 地址或主机名。
net	网络范围	只记录特定数据包，该数据包作为数据源或目标包含指定网络区域内的 IP 地址或主机名。
端口	0 到 65535 间的一个数字	记录特定数据包，该数据包作为数据源或目标包含指定端口，例如 5900。
portrange	0 到 65535 之间的一个范围	记录特定数据包，该数据包作为数据源或目标包含指定端口范围内的一个端口。
src	host, net, port 或 portrange	只记录特定数据包，该数据包作为数据源包含指定值。
dst	host, net, port 或 portrange	只记录特定数据包，该数据包作为数据目标包含指定值。
proto	ether, fddi, tr, wlan, ip, ip6, arp, rarp, decnet, tcp 和 udp	可记录数据包的网络协议。

您可以通过以下表达式将多个选项进行逻辑运算：

and, or, not

可通过下列字符限定扩展表达式：

( ) 和 '

参考

其他参数设置请参见 www.tcpdump.org。

示例

`-n port 22`	；记录数据包是否到达本地系统的端口 22。
`-i eth0 port 4080`	；记录所有到达接口 eth0，用于端口 4080 的数据。
`-i 2`	；记录所有到达带适配器号 2 的接口的数据。
`-n host 10.113.20.0`	；记录系统 10.113.20.0 的网络传输，在本传输中 IP 地址没有转换为主机名称。
`192.168.214.4 and not net 10.113.20.0`	；记录系统 192.168.214.4 的全部网络传输，该传输不会经过系统 10.113.20.0。
`-i 10.113.20.0 -q '(tcp port 80) or (tcp port 443)'`	；一同记录系统 10.113.20.0 网络接口的 HTTP 数据包和 HTTPS 数据包。

15.6.4.4 可访问节点

按下软键“可访问节点”，系统会生成一张列出网络中 X120 和 X150 所有可访问节点的列表并将它显示在屏幕上。该列表可作为实际拓扑结构和设定拓扑结构相比。

说明

系统没有将手持单元和 EKS 识别系统列入“可访问节点”列表中，而是将它们作为“更多节点”插入到拓扑结构中。

步骤

网络诊断

1. “网络状态”窗口打开。

可访问节点

2. 按下软键“可访问节点”。

   窗口“可访问节点”打开。

   屏幕上以表格形式显示了每个节点的 IP 地址、硬件 ID /型号、状态和地点。

排序

3. 按下软键“排序”。

根据名称

根据IP

根据类型

根据地点

4. 按下软键“根据名称”、“根据 IP”、“根据类型”或“根据地点”来排序。

-或者-

拓扑

3. 如果需要查看所有节点的联网情况（如检查连接问题），请按下软键“拓扑”。

   窗口“网络拓扑结构”打开。

   以树形图形式显示每个节点的接口、端口及当前连接。

   提示：为确保拓扑结构显示正确，须为所有的节点定义名称。

全部展开

全部关闭

4. 按下软键“全部展开”或“全部收起”，以最大化或最小化树形图。

转至

5. 按下软键“转至”，通过结构树中选中的行访问节点。

-或者-

详细信息

3. 如果选择的节点有进一步信息，可按下软键“详细”调用网络界面。

   提示：系统上安装的操作软件必须和需要查看的设备位于同一网络中。必要时可通过指定第 2 个 IP 地址（别名）。

   窗口“详细信息”打开。

缩放+

缩放-

4. 按软键“缩放 +”或“缩放 -”，直至达到所需的显示大小为止。

更多节点



3. 按下软键“更多节点”，查找没有在“可访问节点”中显示的节点。
   窗口“查找更多节点”打开。必要时您可以修改地址范围，按下“确认”。



   此时系统找到的更多节点被添加到表格中，输入栏“硬件 ID/型号”、“地点”、“联系人”、“工作站”和“说明”变为可编辑状态。您可勾选第 1 列的复选框，然后点击“保存”，将节点插入到设定拓扑结构中。

对比



3. 按下软键“对比”，您可以对当前拓扑结构与之前保存的厂商拓扑结构、设定拓扑结构或 xml 格式的备份拓扑结构中的某个结构进行对比。
   窗口“对比对象”打开。将光标移动到所需选项上，按下“确认”。



两个拓扑结构之间的不同以一定颜色突出显示：

- 红色：和设定拓扑结构相比，实际拓扑结构中缺少组件或有组件不可访问。
- 粉色：和设定拓扑结构相比，实际拓扑结构中某些组件的属性不同
- 蓝色：和设定拓扑结构相比，实际拓扑结构中有新组件

保存



3. 按下软键“保存”，保存实际拓扑结构。
   窗口“另存为”打开。



4. 将光标移动到所需选项上，输入文件名称和必要的注释，按下“确认”。

说明

只有至少具有访问级别 3（用户）才能创建厂商拓扑结构和设定拓扑结构，这两份文件始终保存在有效的 DHCP 服务器上。

缺省保存目录：

- 实际拓扑结构 (.xml 或.csv)：
  根据配置的不同提供有下列保存目录：
  – 网络拓扑结构（保存目录：HMI 数据/数据备份）
  – 本地驱动器
  – 网络驱动器
  – USB
  在 csv 格式中，分号";"用作分隔符，可以在 sldgconfig.xml 中加以修改。
- 设定拓扑结构：
  /user(_base)/common/tcu/$soll
- 厂商拓扑结构：
  /oem(_base)/common/tcu/$soll

诊断与服务 16

16.1 NC/PLC 变量

16.1.1 显示和编辑 PLC 变量和 NC 变量

只有具有口令，才能更改 NC/PLC 变量。

⚠ 警告
错误的参数设置 NC/PLC 变量的状态更改对机床有重大影响，错误的参数设置可能危及人员生命，并导致机床损坏。

在窗口“NC/PLC 变量”的列表中输入您想要查看或更改的 NC 系统变量和 PLC 变量：

- 变量
  NC/PLC 变量的地址
  错误的变量背景色为红色，且在“数值”列中显示 #。
- 注释
  可随意添加变量注释。
  该列可以显示和隐藏。
- 格式
  指定变量显示的格式。
  该格式可以固定设置，例如：浮点值。
- 值
  NC/PLC 变量的当前数值显示

PLC 变量	
输入端	• 输入位（Ex），输入字节（EBx），输入字(EWx)，输入双字(EDx) • 输入位 (Ix)，输入字节 (IBx)，输入字 (Iwx)，输入双字 (IDx)
输出端	• 输出位 (Ax)，输出字节 (ABx)，输出字 (AWx)，输出双字 (ADx) • 输出位 (Qx)，输出字节 (Qbx)，输出字 (Qwx)，输出双字 (QDx)
标志	标志位（Mx），标志字节（MBx），标志字（MWx），标志双字（MDx）

PLC 变量	
时间	时间（Tx）
计数器	• 计数器（Zx） • 计数器（Cx）
数据	• 数据模块（DBx）：数据字（DBXx），数据字节（DBBx），数据字（DBWx），数据双字（DBDx） • 数据模块（VBx）：数据字（VBXx），数据字节（VBBx），数据字（VBWx），数据双字（VBDx）

格式	
B	二进制
H	十六进制
D	十进制，无符号
+/-D	十进制，带符号
F	浮点（对于双字）
A	ASCII 字符

表示方法示例

允许的变量表示方法：

- PLC 变量：EB2, A1.2, DB2.DBW2, VB32000002
- NC 变量：
  - NC 系统变量：表示方法 $AA_IM[1]
  - 用户变量/GUD：表示方法 GUD/MyVariable[1,3]
  - BTSS - 表示方法： /CHANNEL/PARAMETER/R[u1,2]

说明

若通过 PLC 用户程序将字符串写入 NC/PLC 变量，则仅当在 NC 侧将该变量参数设置为“A”（ASCII）类型的数组变量时，才能正确显示该字符串。

数组变量的示例

变量	格式
DBx.DBBy[<数量>]	A

插入变量

按照不同条件进行“筛选/查找”时，初始值有较大差别。例如要插入变量 $R[0] 时，请输入以下起始值：

- 如果按照“系统变量”筛选，则初始值为 0。
- 如果按照“全部（无过滤）”筛选，则初始值为 1,此时会用 BTSS 输入方式显示所有信号。

如激活了相应的定义文件，则在选择变量时的查找窗口中只会显示机床数据中的 GUD。另一种方法是，手动输入查找的变量，例如 GUD/SYG_RM[1]。

下面的机床数据代表着所有的变量类型（INT、BOOL、AXIS、CHAR、STRING）：MD18660 $MN_MM_NUM_SYNACT_GUD_REAL[1]。

说明

- 系统变量受通道影响，切换通道时，会显示所选通道的数值。
  可以针对特定通道显示变量，例如 $R1:CHAN1 和 $R1:CHAN2。在此情形下会显示通道 1 和通道 2 的值，不论当前处于哪个通道中。
- 用户变量(GUD)无须指定为全局 GUD 或通道专用 GUD。GUD 数组的首个元素从下标 0 开始，如 NC 变量。
- 窗口中弹出的提示条会说明 NC 变量的 BTSS 表示方法（GUD 除外）。

伺服变量

伺服变量只能在“诊断” → “跟踪”下选择和显示。

修改和删除变量的数值



1. 选择操作区域“诊断”。



2. 按下软键“NC/PLC 变量”。

-或者-



1. 选择操作区域“调试”。



2. 按下软键“PLC”和“NC/PLC 变量”。



“NC/PLC 变量”窗口打开。

3. 将光标移到“变量”列中，并输入所需的变量。



4. 按下 <INPUT> 键。

现在，操作数及其数值一同显示。



5. 按下软键“详细信息”。

窗口“NC/PLC 变量：详细信息”打开。“变量”、“注释”和“值”的数据完整显示在窗口中。



6. 将光标移到“格式”栏，按下 <SELECT> 键选择所需格式。



7. 按下软键“显示注释”。

“注释”列随即显示。现在，您可添加注释或者编辑已有注释。



再次按下软键“显示注释”，隐藏注释列。



8. 如果要编辑数值，按下软键“更改”。

“值”列切换到可编辑状态。



9. 若需从所有已有变量中选择一个变量并插入该变量，按下软键“插入变量”。

“选择变量”窗口打开。



10. 按下软键“过滤/搜索”，以通过下拉菜单“过滤”筛选变量的显示（比如，按照运行方式组变量筛选），或通过输入栏“搜索”选择所需的变量。



11. 如果要删除所有的操作数条目，按下软键“全部删除”。



12. 按下软键“确认”，确认完成的修改或删除。

-或者-



按下软键“取消”，放弃修改。

编辑变量列表

使用软键“添加行”和“删除行”可对变量列表进行编辑。



如果按下软键，可在光标所在的行之前新增一行。

只有当变量列表末尾至少有一个空行时，软键“添加行”才可使用。

如果没有空行，则软键无效。



如果按下软键“删除行”，可将光标所在的行删除。

在变量列表末尾添加一个空行。

修改操作数

按下软键“操作数＋”和“操作数-”，地址或地址索引（由操作数的类型决定）会加或减 1。

说明

下标用作轴名称

下标用作轴名称时，软键“操作数+”和“操作数-”无效，比如在 $AA_IM[X1] 中。

	示例
操作数+	DB97.DBX2.5
	结果：DB97.DBX2.6
	$AA_IM[1]
	结果：$AA_IM[2]
操作数-	MB201
	结果：MB200
	/Channel/Parameter/R[u1.3]
	结果：/Channel/Parameter/R[u1,2]

16.1.2 保存和加载屏幕

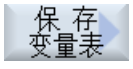
可将“NC/PLC 变量”窗口中进行的变量配置保存在屏幕中，并根据需要加载。

编辑屏幕

更改已加载的屏幕时，屏幕后会有 * 进行标识。

显示关闭后屏幕的名称保存不变。

步骤

1. 已在“NC/PLC 变量”窗口中为所需的变量输入了数值。
2. 
   按下软键“>>”。
3. 

   按下软键“保存屏幕”。
   “保存屏幕：选择存档位置”窗口打开。



4. 将光标放置在您希望保存当前窗口的模板文件夹上，然后按下“确认”。
“保存屏幕：名称”窗口打开。



5. 输入文件名称并按下软键“确定”。
状态行中会出现消息提示：屏幕已保存至指定的文件夹。
若与已有文件同名，会发出询问。



6. 按下软键“载入屏幕”。
窗口“载入屏幕”打开并显示变量屏幕的模板文件夹。
7. 选中所需文件并按下软键“确认”。
重新返回变量视图。 显示已确定的所有 NC 变量和 PLC 变量列表。

16.2 显示维修概览

在窗口“服务概览”中显示机床轴和驱动的运行状态。

可以通过软键“扩展选择>”显示或直接通过选择列表根据以下标准配置：

- 所有 NC 轴和驱动（=预设）
- NC 轴
- 未指定 NC 轴的驱动

显示使能

- 开/关 1
- 关 2
- 关 3
- 来自驱动：运行已使能
- 来自电源：运行已使能
- 脉冲使能
- 转速调节器使能 NC
- 脉冲已使能
- 驱动就绪
- 散热器温度
- 功率部件受 i^2t 限制

- 电机温度
- **测量系统 1 生效/测量系统 2 生效**
  测量系统 1/2 的状态有以下含义：

符号	含义
	位置测量系统已生效。
	位置测量系统已驻停。 -或者：- 未配置位置测量系统。
	位置测量系统未生效。

步骤



1. 选择操作区域“诊断”。



2. 按下菜单扩展键和软键“轴诊断”。



3. 在“选择”栏中选择概览。
   -或者-
   按下软键“转至选择”，选择概览。



16.2.1 选择轴和驱动

要显示某个使能和机床轴状态，可以从窗口“扩展选择”中按任意顺序编排的所有可用轴和驱动中进行选择。

说明

以下选择配置是预先定义的，不可更改或删除：

- 所有 NC 轴和驱动
- NC 轴
- 未指定 NC 轴的驱动

选择概览

您可在选择概览中查看以下信息：

列	说明
总线编号	未指定 NC 轴的驱动的总线编号。 无 NC 轴选项。
从站地址	未指定 NC 轴的驱动的从站地址。 无 NC 轴选项。
设备编号	未指定 NC 轴的驱动的设备编号。 无 NC 轴选项。
驱动对象编号	未指定 NC 轴的驱动的驱动对象编号。 无 NC 轴选项。
名称	对于 NC 轴，其中包含了标准机床轴名称和用户自定义的机床轴名称。 对于未指定 NC 轴的驱动，其为驱动对象名称。
分配	内部对象分配时显示 NC。 外部对象分配时显示 PLC。

说明

在选择概览中，PROFIBUS 和 PROFINET 配置之间有关从站地址和设备编号的信息数据会有所不同。

步骤



1. 选择操作区域“诊断”。





2. 按下菜单扩展键和软键“轴诊断”。
   窗口“服务概览”打开。



3. 按下软键“扩展选择”。
   窗口“扩展选择：轴和驱动”打开。



4. 按下软键“新选择”。
   对话框“新选择”打开。
5. 在输入栏中输入一个显示名和一个文件名。



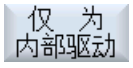
6. 按下软键“确认”结束输入。
   窗口“编辑选择”打开，用于编辑新的选择。
7. 在复选框中勾选您要显示的轴和驱动。
   -或者-
   按下以下某个软键，以进行某项选择：



选择所有轴/驱动。



未选择任何轴/驱动。



只选择内部驱动。



只选择外部驱动。



选择指定了真实驱动的轴。



选择所有至少在一个通道中定义的轴。



8. 按下软键“确认”，确认选择。



9. 按下软键“删除选择”，删除当前选择。



第 10 个 按下软键“编辑选择”，对当前选择进行编辑。
窗口“编辑选择”打开。



11. 按下软键“返回”，进入“服务概览”初始画面。

16.2.2 轴诊断

使用

在“轴/主轴信息”窗口会显示以下信息：

- 检测给定值部分（例如：编程的位置给定值、转速给定值、写入的主轴转速给定值）
- 检测实际值部分（例如：位置实际值，测量系统½，转速实际值）
- 优化轴的位置控制回路（例如：跟随误差、控制偏差、Kv 系数）

- 检测轴的全部控制回路（例如：通过比较位置给定值和位置实际值，转速给定值和转速实际值）
- 检测硬件错误（例如：编码器的控制：如果轴开始机械运动，必然会产生一个位置实际值变化）
- 轴监控设置和检查

步骤



1. 选择操作区域“诊断”。





2. 按下菜单扩展键和软键“轴诊断”。
   窗口“服务概览”打开。



3. 按下软键“轴信息”。
   “轴/主轴信息”窗口打开。





4. 按下软键“轴 +”或“轴 -”，向前或向后选择轴。

   -或者-



   按下软键“轴选择”。
   窗口“轴直接选择：”打开。
   通过选择列表直接从提供的轴中选择所需的轴。



6. 按下软键“确认”，确认选择。
   显示轴的数值。

显示数据

显示数据	含义
跟随误差	位置给定值和有效测量系统 1 或 2 的位置实际值之间的差值 单位：毫米，英寸或者度
控制偏差	位置控制器输入端的位置给定值和有效测量系统 1 或 2 位置实际值之间的差值 单位：毫米，英寸或者度

显示数据	含义
轮廓偏差（轴向）	该值显示当前的轮廓偏差（通过调速器的调节过程基于负载变化导致跟随误差的波动）。 轮廓偏差可以通过从位置给定值预先计算出来的位置实际值和有效测量系统 1 或 2 的位置实际值之间的差值得出。 单位：毫米，英寸或者度
伺服增益系数（计算出的）	所显示的伺服增益系数将由 NC 根据以下公式计算： K_v - 系数 = 速度给定值 / 跟随误差 ； 单位（标准设置）：[m/min] / [毫米] ； 速度给定值 = 当前输出至轴/主轴的给定值 **更多信息：**功能手册之进给轴和主轴
已激活的测量系统	**0：**没有测量系统激活。 **1：**测量系统 1 激活 **2：**测量系统 2 激活
测量系统 1 的状态 测量系统 2 的状态	输出测量系统 1/2 的状态： 生效 - 驻停 - 未生效
测量系统 1 位置实际值 测量系统 2 位置实际值	通过测量系统 1/2 测量轴的实际位置 在机床坐标系中显示位置（不考虑零点偏移和刀具补偿）。 单位：毫米，英寸或者度
位置设定值	从插补器到位置控制输出的设定位置 单位：毫米，英寸或者度
绝对补偿值 测量系统 1 绝对补偿值 测量系统 2	显示测量系统 1/2 的绝对补偿值 补偿值是当前轴位置的间隙补偿和丝杠螺距误差补偿的和。 单位：毫米，英寸或者度
补偿 垂度 + 温度	显示由当前轴位置的垂度补偿和温度补偿总和得出的补偿值 单位：毫米，英寸或者度
有效编码器的转速实际值	NC 会分析并显示编码器输出的脉冲。 单位：% 100% 指的是最大转速。

显示数据	含义
驱动转速给定值	传输至驱动的转速给定值（= 位置控制器和预控制的转速给定值） 单位：% 100% 指的是最大转速给定值。
主轴转速给定值已编程	由用户编程的转速给定值 单位：转/分钟 例如：输入：S1000；显示：1000 rpm 显示仅适用主轴。
当前主轴转速给定值	符号正确、当前有效的转速给定值，带计算的补偿值和有效的转速限制（通过设定数据或机床数据来规定） 单位：转/分钟 显示仅适用主轴。
主动轴/主轴实际值的位置偏置	如果在同步主轴功能内编程一个位置偏置（跟随主轴和引导主轴之间的角度偏差），则在这里显示基于实际值的当前有效值。 单位：毫米、英寸、度 有关同步主轴的**更多信息**：功能手册之进给轴和主轴
主动轴/主轴给定值的位置偏置	如果在同步主轴功能内编程一个位置偏置（跟随主轴和引导主轴之间的角度偏差），则在这里显示基于给定值的当前有效值。 单位：毫米、英寸、度
倍率	显示进给率倍率开关和主轴倍率开关的有效倍率。 单位：%
当前传动级	显示当前实际传动级 当主轴分配到轴时才会显示该轴的数据。显示和 NC/PLC 接口信号相对应： DB31, ... DBX16.0-2（实际传动级） 有关主轴的**更多信息**：功能手册之进给轴和主轴
参数组（轴）	显示位置控制器的 6 个参数组中哪些是有效的。 **更多信息**：功能手册之进给轴和主轴

显示数据	含义
控制器模式	显示当前控制器状态： • 位置控制 • 转速控制 • 停止 • 驻留 • 跟踪 • 制动 **更多信息**：功能手册之基本功能
预控制模式	显示轴的动态预控制模式是否有效，如果有效哪个模式有效： • 无效 • 速度 和速度相关的转速预控制有效。 • 转矩 和加速度相关的转矩预控制有效（结合转速预控制） 有关补偿的**更多信息**：功能手册之监控与补偿
状态“回参考点”	回参考点的状态显示（轴）： • 有效测量系统无需回参考点 （有效测量系统无需回参考点） • 有效测量系统已回参考点 • 有效测量系统需要回参考点 （有效测量系统需要回参考点） 该显示取决于机床数据中的设置： • MD34110 $MA_REFP_CYCLE_NR • MD20700 $MC_REFP_NC_START_LOCK 显示和 NC/PLC 接口信号相对应： DB31, ... DBX60.4 和 60.5（已回参考点/已同步 1 或 2） 有关回参考点的**更多信息**：功能手册之进给轴和主轴

显示数据	含义
QEC 状态	显示轴的象限误差补偿 (QEC) 方法是否有效，如果有效哪些方法有效： • 无效 • 学习系统 QEC 有效 • 常规 QEC 有效 • 带补偿值调整的常规 QEC 有效 （带补偿值调整的常规 QEC 有效） • 系统 QEC 有效 • 带测量时间调整的系统 QEC 有效 （带测量时间调整的系统 QEC 有效） • 带补偿值衰变时间调整的系统 QEC 有效 • （带补偿值衰变时间调整的系统 QEC 有效） • 带补偿值测量时间和衰变时间调整的系统 QEC 有效 （带补偿值测量时间和衰变时间调整的系统 QEC 有效） 有关补偿的**更多信息**：功能手册之监控与补偿
状态“运行至固定挡块”	显示轴在激活功能“运行至固定挡块”时是否满足“到达固定挡块”的条件 (NST DB31，... DBX62.5)： • 正常控制 （功能“运行至固定挡块”未激活） • 到达固定挡块 • 失败 有关运行至固定挡块的**更多信息**：功能手册之监控与补偿
转矩限制值	给出“运行至固定挡块”时通过 FXST[x] 或 SD43510 $SA_FIXED_STOP_TORQUE 编程的值以及通过 MD37010 $MA_FIXED_STOP_TORQUE_DEF 定义的夹紧转矩值。 单位：最大转矩的 %
匹配	显示轴的动态参数和控制参数匹配是否生效： • 无效 • 生效 **更多信息**：功能手册之进给轴和主轴

16.2.3 驱动信息

显示驱动数据和电机数据

在窗口“驱动信息”中显示电机和驱动模块状态的重要信息，例如：电机温度和直流母线电压。

步骤



1. 选择操作区域“诊断”。





2. 按下菜单扩展键和软键“轴诊断”。
   窗口“服务概览”打开。



3. 按下软键“驱动信息”。
   窗口“驱动信息”打开。





4. 按下软键“驱动 +”或“驱动 -”，向前或向后选择驱动。

-或者-



按下软键“驱动选择”。
“驱动直接选择：”窗口打开。
从下拉菜单中直接选择所需驱动。



5. 按下软键“确认”，确认选择。
   显示驱动数据。

一览

下文将介绍窗口“驱动信息”内显示的各个状态、报警和消息等内容。

更多信息

有关接口信号的更多信息，请参见以下文档：

- 功能手册之基本功能
- 参数手册之 NC 变量和接口信号

有关编码器数据组切换的详细信息，请参见

功能手册之监控与补偿。

有关驱动参数（驱动对象 SERVO）的更多信息，请参见

SINAMICS S120/S150 参数手册。

PLC 脉冲使能

窗口中显示的 PLC 是否给出了驱动所需的脉冲使能取决于接口信号：
DB31, ... DBX21.7“脉冲使能”的状态。

状态	含义	显示
1	PLC 给出了该驱动需要的脉冲使能。	是
0	PLC 封锁了该驱动需要的脉冲使能。	否

NC 转速控制器使能

窗口中显示的 NC 是否给出了驱动所需的转速控制器使能取决于接口信号：
DB31 ... DBX61.6“转速控制器激活”的状态。

状态	含义	显示
1	NC 给出了转速控制器使能。	是
0	NC 未给出转速控制器使能。	否

快速停机斜坡函数发生器

窗口中显示的快速停机斜坡函数发生器的状态取决于接口信号：
DB31, ... DBX92.1 “斜坡函数发生器封锁激活”的状态。

状态	含义	显示
1	快速停机斜坡函数发生器激活。驱动因此不利用下降斜坡制动，而是利用转速设定值 0 制动，并且不封锁脉冲。	是
0	快速停机斜坡函数发生器未激活。	否

脉冲已使能

窗口中显示的关于驱动是否获得了脉冲使能的消息取决于接口信号：
DB31, ... DBX93.7“脉冲已使能”的状态。

状态	含义	显示
1	驱动模块所需的脉冲已经使能。进给轴/主轴可以移动。	是
0	驱动模块所需的脉冲已被封锁。进给轴/主轴无法移动。	否

驱动就绪

窗口中显示的所选驱动的当前状态取决于接口信号：
DB31, ... DBX93.5“驱动就绪”的状态。

状态	含义	显示
1	驱动已经准备好运行。	是
0	驱动没有准备好运行。	否

启动阶段

窗口中显示的所选驱动的当前启动状态取决于驱动参数：
r0002 “驱动运行显示”。

出错生命符号的数量

显示利用硬件检测出的 NC 和驱动之间的通讯故障。

说明

如果窗口中的显示不为 0，请联系当地的西门子办事处！

直流母线 1 驱动报警消息

显示是否有 1 类状态的消息。1 类状态的消息指具有以下特性的报警：

- 报警会引发内部响应，比如：再生式制动、立即封锁脉冲等。
- 报警是无需应答的。

此类报警是总报警。关于实际可能出现的驱动报警请查看窗口“驱动系统诊断”，见显示驱动状态 (页 347)的说明。

经过滤波的直流母线电压

窗口中显示的所选驱动经过滤波的直流母线电压取决于驱动参数：
r0026“经过滤波的直流母线电压”。

单位：V

经过滤波的转速设定值

窗口中显示的经过滤波的转速设定值（在针对转速控制器比例元件的设定值限幅后测出）取决于驱动参数：
r1438“转速控制器转速设定值”。

单位：rpm

转速实际值

窗口中显示的经过滤波的实际电机转速取决于驱动参数：
r0021“经过滤波的转速实际值”。

单位：rpm

经过滤波的电流实际值

窗口中显示的经过滤波的电流实际值取决于驱动参数：
r0078[1]“转矩电流实际值”。

单位：A

电机温度

窗口中显示的实际电机温度取决于驱动参数：
r0035“电机温度”。
单位：°C

积分元件封锁

窗口中显示的关于转速控制器的积分元件是否激活的状态取决于接口信号：
DB31, ... DBX93.6“转速控制器积分元件已封锁”的状态。

状态	含义	显示
1	驱动中转速控制器的积分元件已被关闭，因此转速控制器从比例积分控制器转换成比例控制器。	是
0	转速控制器中的积分元件已使能，位置控制器作为比例积分控制器工作。	否

驻留轴

显示是否有被驻留的进给轴或主轴。

说明

进给轴/主轴一旦被驻留后，所有和编码器相关的监控和计算功能都将关闭。因此可以卸下编码器，而不会触发任何报警。

设定驱动数据组

窗口中显示的 PLC 应激活 8 个驱动数据组中的哪个数据组取决于接口信号：
DB31, ... DBX21.0 到 21.2“驱动数据组选择 A、B、C”，缺省值是 DDS0。

关于如何处理驱动数据组和电机数据组的详细说明请参见：
数控系统调试手册：NCK、PLC、驱动

实际驱动数据组

窗口中显示的 8 个驱动数据组中当前激活的驱动数据组取决于接口信号：
DB31, ... DBX93.0 到 93.2“激活的驱动数据组 A、B、C”，缺省值是 DDS0。

生效编码器的设定/实际编码器数据组

显示请求的和生效的编码器数据组。

设定电机数据组

窗口中显示的 PLC 应激活哪个电机数据组取决于接口信号：
DB31, ... DBX21.3 到 21.4“电机据组选择 ABC”，显示值范围为：MDS0...3。

具体可以分为：

电机数据组	编码	
MDS0	0	0
MDS1	0	1
MDS2	1	0
MDS3	1	1

实际电机数据组

窗口中显示的当前激活的电机数据组取决于接口信号：
DB31, ... DBX93.3 到 93.4 “激活的电机数据组 AB”，显示值范围为：MDS0...3。

运行方式

窗口中显示的驱动的控制方式取决于驱动参数：
p1300[0...n] “开环控制/闭环控制方式”。

不同的下标 n 中显示值有所不同：

n	显示
20	无编码器的转速控制
21	带编码器的转速控制
23	带编码器的转矩控制

测量系统 1/2 提供的位置实际值

测量系统 1/2 测出的实际位置在机床坐标系中显示位置（不考虑零点偏移和刀具补偿）。

单位：毫米，英寸或者度

散热器温度

窗口中显示的关于散热器温度是否正常的消息取决于接口信号：
DB31, ... DBX94.1“散热器温度预警”的状态。

状态	含义	显示
1	驱动向 PLC 发出散热器温度预警。	过热
0	驱动模块的散热器温度监控没有发出响应。	OK

电机温度

窗口中显示的关于电机温度是否正常的消息取决于接口信号：
DB31, ... DBX94.0“电机温度预警”的状态。

状态	含义	显示
1	电机温度超出了驱动中定义的报警阈值。	过热
0	电机温度低于报警阈值。	OK

加速过程结束

窗口中显示的驱动是否结束加速取决于接口信号：
DB31, ... DBX94.2“加速过程结束”的状态。

状态	含义	显示
1	在给出新的转速设定值后，转速实际值达到了 p2164 定义的转速公差带，并且在 p2166 时间期满后仍在公差带内。	是
0	修改转速设定值后，驱动仍进行加速。	否

低于转矩阈值

窗口中显示的驱动是否低于转矩阈值取决于接口信号：
DB31, ... DBX94.3 “$|M_d| < M_{dx}$”的状态。

状态	含义	显示
1	当前转矩利用率低于 p2194 中设置的转矩利用率阈值。驱动向 PLC 报告，转矩设定值$|M_d|$ 没有超过转矩阈值 M_{dx}。	是
0	转矩设定值$|M_d|$超过了转矩阈值 M_{dx}。借助该信号可判断电机是否过载。	否

低于最小转速

窗口中显示的驱动是否低于最小转速取决于接口信号：
DB31, ... DBX94.4 “$|n_{实际}| < n_{最小}$”的状态。

状态	含义	显示
1	转速实际值$|n_{实际}|$低于 p2161 中设置的最小转速 $n_{最小}$。	是
0	转速实际值超过了最小转速。	否

低于转速阈值

窗口中显示的驱动是否低于转速阈值取决于接口信号：
DB31, ... DBX94.5 “$|n_{实际}| < n_x$”的状态。

状态	含义	显示
1	转速实际值$|n_{实际}|$低于 p2155 中设置的最小转速 n_x。	是
0	转速实际值超过了设置的转速阈值。	否

转速实际值 = 转速设定值

窗口中显示的驱动转速实际值是否等于转速设定值取决于接口信号：
DB31, ... DBX94.6 “$|n_{实际}| < n_{设定}$”的状态。

状态	含义	显示
1	转速实际值与转速设定值之差在 p2163 设置的公差内。	是
0	转速实际值与转速设定值之差在设置的公差外。	否

出现报警时开展诊断

在出现报警时这些信息也可以为诊断提供便利：

- 驱动故障信息：
  ⇒ **直流母线 1 驱动报警**已设置。
  ⇒ 检查**设定驱动数据组**、**设定电机数据组**和**直流母母线电压**。
- 报警 25040"静态监控"
  报警 25050"轮廓监控"
  报警 25060"转速设定值限制"
  报警 25080"定位监控"
  ⇒有可能是因为缺少驱动使能（不存在 **PLC 脉冲使能**）；从而导致屏幕上显示"**脉冲已使能** = 否"。
- **电机温度**过热
  ⇒ 检查实际电机温度。

更多信息

出现报警时数控系统的响应方式及解决办法请参见

诊断手册。

16.3 系统负载

您可以在 NC 区域显示当前所用的系统资源（负载显示）。

- 总 NC 负载
- 位置控制器和插补器占用的 NC 负载
- 位置控制器、插补器和预处理的运行时间
- 同步动作占用的时间
- 插补缓冲区的占用率

说明

如果 MD11510 $MN_IPO_MAX_LOAD 中的值 ≠ 0，则只显示同步动作的负荷计算。

步骤



1. 选择“诊断”操作区域。



2. 按下菜单扩展键。
   显示其他水平软键。



3. 按下软键“系统负载”。
   “系统负载”窗口打开。
   可以跟踪当前最新的负载显示。



4. 按下软键“停止”，停止显示更新。



5. 按下“开始”软键再次更新值。

16.4 生成屏幕拷贝

可通过当前操作界面生成截屏。

每个截屏将保存为文件，并存放在如下文件夹中：

/**user**/sinumerik/hmi/log/screenshot

步骤

Ctrl + P　按下快捷键 <Ctrl + P>，
从当前操作界面创建格式为 .png 的截屏。
文件名由系统按升序给定，从“SCR_SAVE_0001.png”到“SCR_SAVE_9999.png”。
最多可以创建 9999 幅图。

复制文件



1. 选择操作区域“调试”。



2. 按下软键“系统数据”。

3. 打开上述文件夹并选择必要的截屏。

复制

4. 按下软键“复制”。

-或者-

剪切

按下软键“剪切”。

粘贴

5. 打开需要的存储位置目录，例如在 U 盘上，并按下软键“粘贴”。

说明

您还可以将截屏通过“WinSCP”复制到 Windows-PC 上。

说明

如要查看截屏，可在 SINUMERIK Operate 中将文件打开。 在 Windows-PC 上可使用图片编辑程序将文件打开，例如“Office Picture Manager”。

16.5 机床识别符

可以将机床的目前仅以纸质形式提供的重要信息以电子形式存储，例如用于终端用户提示。在进行维修时，可借助远程诊断查看这些记录，或者为了注册机床而将这些记录传输至西门子工业服务平台。

可为每个机床保存下列数据：

- 机床类型
- 机床编号
- 制造商信息
- 销售商信息
- 最终用户信息
- 所用控制技术的组件列表
- 机床修改的日志条目

可在“identSNAPSHOT”窗口中保存机床信息和地址信息。您可以手动采集数据或从模板中导入。

机床信息

条目	含义
机床编号	在头行中显示有唯一的机床编号（ CF 卡的编号）。该编号仅用于提供信息并且不能修改。
机床名称	在该栏中由制造商给定一个特有的机床编号并保存在机床数据 MD17400 $MN_OEM_GLOBAL_INFO 中。 提示：机床名称是必填栏。只有在填写了该栏后才可以保存数据。
机床类型	机床的类型

地址信息

在地址处可以获得制造商/制造商办事处、销售商和最终用户的下列数据：

- 用户编号
- 制造商名称和办事处名称/销售商名称/最终用户名称
- 街道、邮政编码/邮递区号、地点、 省（可通过选择列表选择）、地区/国家
- 联系人的地址：名称、电话、传真、电子邮件、网址

16.5.1 输入机床专用信息

前提条件

若需输入和更改机床专用信息，必须拥有如下访问权限。

制造商 访问级别 1（制造商）

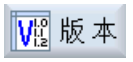
经销商 访问级别 2（服务）

最终用户 访问级别 3（用户）

步骤



第 1 选择操作区域“诊断”。



2. 按下软键“版本”。
调用版本显示需等待一段时间。对话框中会显示一个进度条和说明性文字。



3. 按下软键“日志”。



第 4 步 按下软键“更改”。
窗口“IdentSNAPSHOT”打开。

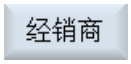
5. 采集机床名称和机床类型。

-或者-

导入模板：导入模板 (页 325)









6. 选择想要采集的地址信息。为此请按下以下其中一个软键：
- “制造商”
如果再次按下软键“制造商”，则软键变为“制造商办事处”。
- “制造商办事处”
再次按下该软键，则软键恢复为“制造商”。
- “经销商”
- “最终用户”

7. 在相应文本栏中输入地址数据。

-或者-

导入模板：导入模板 (页 325)



按下软键“确认”保存修改。

参见

1. 确定开机运转 (页 331)

2. 确定开机运转 (页 331)

创建日志条目 (页 332)

16.5.2 创建模板

借助模板准备机床信息和地址信息并传输至控制系统中。这样就不需要手动输入了。

或者您也可以使用程序“PridaNet identSNAPSHOT”从 工业服务平台数据库中导出机床信息和地址信息并用作模板。如此可保证信息是唯一的。

模板

可在目录“HMI 数据/模板/示例/机床识别”下选择模板。

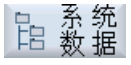
控制系统上存在以下模板：

- “dealer.xml”，针对经销商数据
- “ma_types.xml”，针对机床类型
- “oem.xml”，针对制造商数据
- “oemsubs.xml”，针对制造商办事处数据
- “user.xml”，针对最终用户数据

步骤



1. 选择操作区域“调试”。



2. 按下软键“系统数据”。



3. 从目录
“/HMI 数据/模板/示例/机床信息”中将文件拷贝至数据存储器中。
4. 将编辑文件拷贝至 PC。
5. 编辑完之后将数据拷贝至数据存储器并传输至控制系统。



6. 将文件复制粘贴至文件夹 HMI 数据/模板/制造商/机床信息或使用软键“导入数据”。
地址或机床类型会在显示在选择列表中。在选择了所需的地址后会自动在输入栏中输入与之相关的保存数据。

获取手轮数据

在文件“dealer.xml”中可以获得任意多个经销商的地址数据。需要输入多个经销商时，拷贝每个经销商的 <Dealer> 到 </Dealer>。

```
<?xml version="1.0" encoding="utf-8"?>
<Addresses>
   <销售商>
```

```
    <Customer-Id></Customer-Id>
    <Name></Name>
    <Street></Street>
    <ZIP-Code></ZIP-Code>
    <Location></Location>
    <Country></Country>
    <State></State>
    <Contact></Contact>
    <Phone></Phone>
    <Fax></Fax>
    <E-mail></E-mail>
    <URL></URL>
  </Dealer>
</Addresses>
```

在文件"oem.xml"中可以获得任意多个制造商的地址数据。需要输入多个制造商时，拷贝每个制造商的 <Manufacturer> 到 </Manufacturer>。

在文件"oemsubs.xml"中可以获得任意多个制造商办事处的地址数据。需要输入多个制造商办事处时，分别拷贝 <Manufacturer-Subsidiary> 到 </Manufacturer-Subsidiary>。

在文件"user.xml"中可以获得任意多个用户的地址数据。需要输入多个用户时，分别拷贝 <User> 到 </User>。

示例：文件"ma_types.xml"

在文件"ma_types.xml"中可以获得可用的机床类型。

```
<?xml version="1.0" encoding="utf-8"?>
<Machinetypes>
   <Type>LC 80</Type>
   <Type>LC 82</Type>
   <Type>LC 120</Type>
   <Type>LC 122</Type>
   <Type>LC 150</Type
</Machinetypes>
```

16.5.3 导入模板

您可以直接将模板读入到控制系统中。取决于配置，这些文件可以来自本地驱动器、网络驱动器或 USB 设备，导入成功后，文件会自动复制到文件夹：/**oem**/sinumerik/hmi/template/identity 或 HMI 数据/模板/制造商/机床信息。

模板

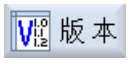
您可以导入任意名称的文件。文件的内容会被检测。根据所含信息生成新文件名。

文件	说明
ma_types.xml	机床类型
dealer.xml	经销商数据
oem.xml	制造商数据
oemsubs.xml	办事处的制造商数据
user.xml	最终用户数据

步骤



1. 选择操作区域“诊断”。



2. 按下软键“版本”。
   获取版本信息的过程较慢，在对话框中会显示一个信息获取进度条。





3 按下软键“日志”和“更改”。



4. 按下软键“导入数据”。
5. 选择相应的驱动器和所需的文件。



6. 按下“确认”软键。
   文件会自动复制到系统中正确的位置。
   如果模板只含有一个数据组，那么数据会直接被接收到输入栏中并保存。



7. 如果输入栏中已存在数据，请按下软键“是”，覆盖数据。
8. 如果模板含有多个数据组，请从列表中选择一个数据组。
   相应数据会被接收到输入栏中。



9. 按下软键“确认”。
   确认并保存选择或手动修改。

参见

创建模板 (页 323)

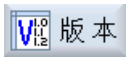
16.5.4 保存信息

控制系统中机床专用的所有信息都通过操作界面汇集到一个配置文件中。通过所设置的驱动器可对机床专用数据进行保存。

步骤



1. 选择操作区域“诊断”。



2. 按下软键“版本”。
   调用版本显示需等待一段时间。对话框中会显示一个进度条和说明性文字。



3. 按下软键“保存”。
   窗口“保存版本信息：选择存储地点”打开。根据配置的不同提供有下列存储位置：
   - 本地驱动器
   - 网络驱动器
   - USB
   - 版本数据（存储位置：“HMI 数据”目录下的树形图）



4. 若需创建单独的目录，按下软键“新建目录”。



5. 按下软键“确认”。目录创建完毕。



6. 再次按下软键“确认”，确认存储位置。
   窗口“保存版本信息：名称”打开。

7. 确定所需预设值。
   - 输入栏“名称：”
     文件名默认为 <机床名称/编号>+<CF 卡编号>。文件名称后会自动添加“_config.xml”或“_version.txt”。
   - 输入栏“注释：”
     在本栏中可以输入注释，注释与配置数据一同保存。

- 版本数据（.TXT）
  如需输出纯文本格式的版本数据，请勾选复选框。
- 配置数据（.XML）
  如需输出 XML 格式的配置数据，请勾选复选框。
  配置文件中包含在机床信息下所输入的数据、许可证要求、版本信息和日志条目。



8. 按下软键“确认”，启动数据传输。

16.5.5 添加硬件组件

西门子提供的硬件大多数具有电子数据，您可以在系统中添加硬件。

您可通过操作界面查看并添加硬件。

所有硬件信息均包含在版本显示和配置数据中。

前提条件

要添加其他硬件组件需要访问级别 2（服务）。

输入添加的硬件

添加硬件时，必须为硬件指定类别，窗口“添加硬件组件”中的下拉菜单提供以下类别选项：

类别

- NCU/PLC
- 操作面板
- PLC 外设
- 驱动器/电机
- 电缆
- 附件/其他

窗口“输入附加组件（配置数据）”包含以下数据：

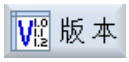
条目	含义
名称	硬件名称
版本	版本信息
MLFB	订货号

条目	含义
序列号	序列号
数量	组件的数量

步骤



1. 选择操作区域“诊断”。



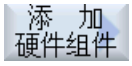
2. 按下软键“版本”。

   获取版本信息的过程较慢，对话框中会显示一个进度条和说明性文字。



3. 选中“硬件”区并按下软键“详细”。窗口“版本数据/硬件”打开。

   软键“添加硬件组件”显示在屏幕上。



4. 按下软键“添加硬件组件”。

   下拉菜单“选择组件类型”打开。

5. 选择需要保存其信息的硬件的类别。

   “输入额外的硬件组件”窗口打开。

6. 通过键盘输入添加硬件的信息。

   -或者-

   若使用 CSV 文件（CSV = 逗号分隔值），例如来自工业服务平台的 BOM 零件清单，则也可以读入这些数据。

   选择 CSV 文件的保存目录。



   选中该文件并按下“确认”。

   现在，CSV 文件中的数据会传送到组件列表中。根据 BOM 清单的 CSV 格式从工业服务平台读入这些数据。其中包括每一个组件的数量、MLFB、名称和序列号。版本不在 CSV 文件中，未被导入。

   示例：

   STUECK;MLFB;Z_OPTION;SERIAL

   12;6AV7812-0BB11-2AC0;;

   1;6FC52030AB110AA2;;T-0815



7. 按下软键“确认”。

   数据随即导入到配置文件“versions.xml”中，电子输入完成。

   所有硬件信息随即显示在窗口“版本数据/硬件”中。

   手动添加的组件前标有“+”符号，例如：电缆+。

16.5.6 配置数据

机床专用信息储存在配置数据中。这些以 XML 格式保存的文件是工业服务平台的进一步处理的基础。

- 可以借助远程诊断检查这些数据。
- 也可以将数据直接传输至工业服务平台。
  更多信息请访问网址 (https://www.siemens.com/sinumerik/register)。

配置数据

配置数据包括：

- 机床专用数据，保存在对话框“机床信息”中
- 硬件/软件版本，保存在对话框“版本”中
- 需授权选件，保存在对话框“许可”中
- 日志，其条目保存在对话框“日志”中

存放路径

根据驱动器配置的不同，可以将配置数据保存在可用的目录下。

参见

机床识别符 (页 321)
添加硬件组件 (页 328)
许可 (页 111)
日志 (页 330)

16.6 日志

日志就是机床的电子历史记录。

机床的调试时间记录在日志中，并且在对机床进行维护时，也可将其作为电子记录存储。从而对维护进行优化。

存储的日志记录不可更改或删除。

16.6.1 1. 确定开机运转

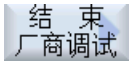
前提条件

机床标识信息中至少应输入机床名称/编号、客户编号和制造商所属的省。

步骤



1. 获取机床专用信息。
2. 调试机床。



3. 按下软键“调试 1 退出”。
   “机床日志”窗口中显示条目“首次设置”及其日期和时间。

说明

执行二次调试

首次调试结束后，系统会提示进行第二次调试。
如未进行第二次调试，每隔一定的时间系统便会要求完成第二次调试。

参见

输入机床专用信息 (页 322)

16.6.2 2. 确定开机运转

前提条件

必须在机床标识信息中至少输入最终用户所属的省。

步骤



获取机床专用信息。

2. 调试机床。



3. 按下软键“调试 2 退出”。
   “机床日志”窗口中显示条目“第二次设置”及其日期和时间。

说明

在工业服务平台中注册机床信息

在第 2 次调试结束后，需要通过互联网在工业服务平台数据库中注册机床信息。

参见

输入机床专用信息 (页 322)

16.6.3 创建日志条目

可通过窗口“新日志条目”在日志中创建新条目。

您可以输入名称、公司和机构，简短地描述所采取的措施或者发生的故障。

说明

设置换行

若需在“故障诊断/措施”栏中换行，可使用快捷键<ALT> + <INPUT>。

会自动添加日期和条目编号。

条目的排序

日志条目经编号后显示在窗口“机床日志”中。

显示时，最新的条目总是位于上部。

步骤

1. 日志已打开。
2. 

按下软键“新项”。
“新日志条目”窗口打开。
3. 

执行所需输入，按下软键“确认”。
返回到窗口“机床日志”，新建条目显示在机床标识数据的下方。

说明

删除日志条目

在进行二次调试前您都可以使用软键“清除”删除到首次调试为止输入的日志条目。

查找日志条目

您可以通过查找功能找出特定条目。

1. “机床日志”窗口已打开。
2. 按下软键“搜索”。

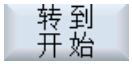


3. 在查找框中输入所需关键字。您可以按照日期/时间、公司名称/单位或者按照故障诊断/措施等条件查找。

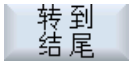
   光标将定位在与查找关键字吻合的第一个条目上。
4. 如果找到的条目不符合您的需要，按下软键“继续搜索”。



其他搜索方法



按下软键“转至开头”，从最新的条目开始搜索。



按下软键“转至结尾”，从最老的条目开始搜索。

16.7 操作日志

16.7.1 设置操作日志

在窗口“操作日志设置”中可以确定，是否记录整个操作过程。

在此您可以选择记录哪些控制系统的动作，以便将来分析操作过程。

说明

免责说明

西门子不保证运行记录仪数据的完整性，也无法阻止数据日后被篡改、

激活了操作日志后，您可以通过下面的文件查看系统数据日志：

- “actual_actionlog.com”，当前操作日志
- “actual_crashlog.com”，系统崩溃日志备份

可记录的数据

记录开	打开或关闭记录。
报警状态改变	记录报警的来、去信息。
按键操作	记录所有操作面板和外部键盘上的所有操作。
通道状态更改	NC/PLC 状态通过通道状态信息进行记录。在时间充裕时，通过这些状态至少可以部分验证机床操作面板上的操作。
窗口切换	打开和关闭窗口时记录格式名称和对话框名称（由编程人员命名的名称）。
写入 NCK/PLC 数据	记录 NCK 和 PLC 变量的写入过程。
文件存取	记录向 NC 进行的复制过程。
NCK 中的功能调用（PI 服务）	记录特定的程序调用，例如 ASUP。
激活程序状态	为特定事件激活辅助信息。在出现需要使用 NC 停止、NC 启动或 NC 复位的重要报警时，会记录例如实际值和当前程序段。 可以确定辅助信息要使用哪个通道或哪根主轴。
文件写间隔	文件备份的预设： • “自动”：操作日志会在内部缓冲器中备份信息。缓冲器写满后将记录储存到 CF 卡上。 控制系统关闭时记录可能会丢失。 • “在每个事件后”：直接储存所有的记录项，断电或其他情况时也不会丢失。 注意：因为 CF 卡只允许使用限制数量的写存取，因此不建议正常运行时使用该设置。 • “时间控制的”：按照特点的时间储存新的记录项。另外在输入栏中可以以秒为单位设定时间。
出现报警时将日志保存为文件	给出生成“Crashlog”的报警号。输入报警时用逗号隔开。

创建系统崩溃日志文件

出现紧急事件时，例如急停报警，“actual_crashlog.com”相当于当前日志的一个备份文件。

因此文件不再处于操作日志的环形缓冲器内，并且也不会被新的记录项覆盖。

只有当出现其他紧急事件时，系统崩溃日志文件中的记录项才会被覆盖。

文件一开始为空文件，只有出现以下事件后才被写入：

- DB19.DBX0.6 接口信号“备份操作日志”从 0 切换到 1。
- “出现报警时将日志保存为文件”栏中输入的报警出现。

接口信号

DB19 (PLC → HMI)								
	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB00		备份操作日志						

步骤



1. 选择操作区域“调试”。



2. 按下软键“HMI”、“诊断”和“操作日志”。

   窗口“操作日志设置”打开。

   默认情况下，“数据选择”区域是激活的。

3. 选中复选框“记录开”，激活操作日志。
4. 如果希望操作日志中包含特定的事件，请选中对应的复选框。

16.7.2 插入变量

在“操作日志设置”窗口中您可以插入最多 50 个您想记录或监控的变量。

从选择列表中选择可用变量或手动添加至列表。

从变量列表中插入变量

1. 进入窗口“操作日志设置”。
2. 按下软键“变量选择”。
   “用于持续监控的变量”窗口打开。
3. 按下软键“插入变量”，从包含所有已有变量的列表中插入一个变量。
   “选择变量”窗口打开。
4. 从列表中选择所需变量。
5. 按下软键“插入”。
   “变量属性”窗口打开。
6. 根据需要接收或更改所显示的属性，例如：轴分配。
7. 若要限制变量显示，在标题区选择各自的分类（例如：系统变量）即可。
   -或者-
   按下软键“筛选/搜索”，通过下拉菜单“筛选”限制变量的显示，或通过输入栏“搜索”选择所需的变量。
8. 若要清除搜索条目，按下软键“清除搜索”。
9. 按下软键“确认”，确认修改。
   -或者-
   按下软键“取消”，放弃修改。

手动插入变量

1. 进入窗口“操作日志设置”。
2. 按下软键“变量选择”。
   “用于持续监控的变量”窗口打开。
3. 将光标移到“变量”列中，并输入所需的变量。
4. 从选择列表的“格式”一列中选择所支持的变量格式。
5. 按下软键“确认”，确认完成的修改或删除。
   -或者-
   按下软键“取消”，放弃修改。

16.7.3 显示记录文件

在窗口“操作日志 日志文件”中会显示所有在日志功能激活时可记录的数据。

日志文件

可以通过以下文件显示当前数据：

- "actual_actionlog.com"
- actual_crashlog.com

步骤



1. 选择操作区域“调试”。



2. 按下软键“系统数据”。
   显示树形图。
   在文件夹“HMI 数据”的路径：/日志/操作日志下可以查看保存的文件。
3. 选择所需的 com 文件，并按下软键“打开”或者双击此文件。



4. 按下软键“重新显示”，可以刷新日志（actual_actionlog.com）中的事件显示。
   会显示上一次调用日志文件后的记录项。

16.7.4 在记录文件中搜索

您可以在日志中根据需要查找事件。

前提条件

打开所需的日志文件。

步骤



1. 按下软键“搜索”。
   显示新的垂直软键栏。 同时打开“搜索”窗口。
2. 在“文本”栏中输入搜索词。



3. 将光标定位至“方向”栏中并通过 <SELECT> 键选择搜索方向（向前、向后）。



4. 按下软键“确认”，开始进行搜索。

如果找到要搜索的文本，则标记出相应的行。



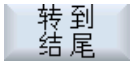
如果需要继续查找，按下“继续搜索”软键。

-或者-

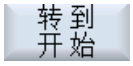


如果需要中断搜索，则按下软键“取消”。

其他搜索方法



1. 按下软键“转到结尾”，可以比较长的日志中一直翻页至结尾。 跳转到日志中最旧的条目。



2. 按下软键“转到开始”，跳转到日志中最新的条目。

16.7.5 保存记录

所显示的记录可保存在“hmi 数据/记录/报警记录”目录中。 它可以作为二进制文件或 ASCII 文件加以保存。 可以使用任意编辑器阅读 ASCII 文件。

可以生成以下日志：

- action.com（二进制文件）
- action.log（ASCII 文件）
- crash.com（二进制文件）
- crash.log（ASCII 文件）

前提条件

打开所需的日志文件。

步骤



1. 按下软键“保存日志”。

记录保存在“hmi 数据/记录/报警记录”目录中。

说明

软键“保存日志”在文件尚未保存时才能使用。

16.7.6 日志文件的结构

日志文件的内容

对于记录的每个事件，都会记录以下数据：

HMI 版本　　NC 版本

系统　　记录关键字　　日期/时间

记录文本

详细信息

条目		说明
HMI 和 NCK 版本		
系统：		
	HMI	按键操作，窗口切换
	NC	写入变量，PI 通讯
	MSG	报警记录
	USR	由机床制造商创建的用户记录
	ERR	操作日志错误，例如未找到日志文件
记录关键字：		
	HMI_START	“HMI 启动”记录
	HMI_EXIT	“HMI 关闭”记录
	PLC_CRASH	“PLC 崩溃”记录
	KEY_PRESSED	“键按下”记录
	KEY_HOLD	“键按住”记录
	KEY_RELEASED	“键松开”记录

条目		说明	
	ALARM_ON	“报警显示”事件的记录	见下“报警显示”
	ALARM_OFF	“报警消失”事件的记录	
	ALARM_ACK	“报警应答”事件的记录	
	OPEN_WINDOW	“窗口打开”的记录	
	CHN_STATE_CHANGED	“通道状态变更”的记录	
	OPMODE_CHANGED	“运行方式切换”的记录	
	TOOL_CHANGED	“换刀”的记录	
	OVERRIDE_CHANGED	“倍率变更”的记录	
	DOM_CMD	“下载至 NC”的记录	
	PI_CMD	“PI 通讯”的记录	
	WRITE_VAR	“写入 NCK/PLC 变量”的记录	
	AREA_CHANGED	“区域切换”的记录	
	NC_CONNECTION	“NC 连接建立/断开”的记录	
	USER	OEM 接口的用户记录	
	ACTIVATED	操作日志已激活	
	DEACTIVATED	操作日志已取消激活	
	INTERNAL	内部操作日志记录	
日期/时间		事件的日期和时间 在 HMI_Start、PLC_Crash 或日期变更中从最后一条记录起会在关键字和日期之间标记一个“-”。	
记录文本		事件描述为明码文本	
详细信息		记录的更多详细信息，例如记录的 IPO 跟踪。	

示例

说明

接口信号 DB19.DBW24

示例中所用的接口信号 DB19.DBW24 仅适用 SINUMERIK 840D sl。 适用 SINUMERIK 828D 的是 DB1900.DBW4。

```
NCK    WRITE_VAR                                    18.02.2009  09:40:12
       ncu1.local wrote Variable: DB19.DBW24 = 0
```

```
HMI    KEY_PRESSED                                    18.02.2009  09:40:12
       Key up: Key up: "Return" (1000004/ 0)
HMI    HMI_START ----------------------------         18.02.2009  09:34:15
       HMI started.
```

报警显示

```
MSG      ALARM                                                    日期/时间
         编号          删除标准                  文本
         详细信息
```

条目		说明
MSG		
ALARM		报警类型： ALARM_ON, ALARM_OFF, ALARM_ACK
日期/时间		事件的日期和时间
编号		报警号
删除标准：		报警被应答时的设定
	AUTOMATIC	自动应答
	POWER-ON	通过 PowerOn 应答
	RESET	通过 NCK 复位应答
	NC-START	通过 NC 启动应答
	NC-RESET	通过 NC 复位应答
	ALARM-CANCEL	通过 Alarm-Cancel 键应答
	RECALL	通过回调键应答
	HMI	通过 HMI 应答
	PLC	通过 PLC 应答
文本		显示英语报警文本
详细信息		记录的更多详细信息。

示例

```
MSG    ALARM_ON                                         20.02.2009   14:25:37
       8020   POWER-ON   : Option 'activation of more than 1 channels' not
       set
```

```
DETAILS:
Mode : JOG Program: canceled Channel: interrupted
Program-Level information:
Level Program running:                          Invoc         偏移
1      /_N_MPF0                                 0             0
```

16.7.7 扩展设置

操作日志的日志文件的大小在配置文件"actlog.ini"中定义。

步骤

1. 可从以下目录中复制模板配置文件"actlog.ini"：/siemens/sinumerik/hmi/template/cfg。
2. 在目录 /oem/sinumerik/hmi/cfg 或
   /user/sinumerik/hmi/cfg 中创建和保存文件。
3. 打开文件，然后在"; ActionLogSize="后输入文件的大小（单位：字节）。

配置文件"actlog.ini"

操作日志的模板配置文件

```
;
; 要激活设置时，删除
; 行首的注释符';'
[ActionLogSettings]
;使用以下设置修改内部操作日志的路径
;ActionLogPath=/user/sinumerik/hmi/ac_log/action.com
;使用以下设置修改内部操作日志的大小
;ActionLogSize=5000000
;使用以下设置修改内部崩溃日志的路径
;CrashLogPath=/user/sinumerik/hmi/ac_log/crash.com
```

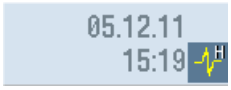
16.8 HMI 跟踪

创建 HMI 跟踪

“HMI 跟踪”功能用于进一步诊断操作软件。前提条件是您至少拥有访问级别 3（用户）。

创建 HMI 跟踪文件的步骤：

1. 从服务与支持中心的专家处获取文件 sltrc_keybitset.ini。将文件 sltrc_keybitset.ini 复制到 USB 存储器上，以便将文件载入控制系统或操作面板。
2. 选择操作区“调试” → “HMI” → “诊断” → “HMI 跟踪”。
3. 按下软键“载入配置”，将 ini 文件从 USB 存储器中载入系统。
4. 选中对应的复选框来启动跟踪功能，或者从操作界面的任意一个位置起利用以下快捷键启动跟踪功能：
   在操作面板上：<ALT> + <T>
   在外部键盘上：<ALT> + <Shift> + <T>
   屏幕标题栏的右上方会显示以下图标，表示正在进行跟踪。

5. 在操作面板上重复需要进行诊断的操作。
6. 撤销对应的复选框来停止跟踪功能，或者从操作界面的任意一个位置起利用以下快捷键停止跟踪功能：
   在操作面板上：<CTRL> + <T>
   在外部键盘上：<CTRL> + <Shift> + <T>
7. 现在生成一份输出文件 sltrc.out，按下软键“导出数据”将文件导出到 USB 存储器上。
8. 将文件 sltrc.out 发回给服务与支持中心进行后续分析。

说明

在结束分析后我们建议删除../user/sinumerik/hmi/log/sltrc 下的所有文件。

说明

软键“高级”

该对话框中有更多选项，有助于专业的服务人员分析系统和评估 HMI 跟踪结果。

16.9 PROFIBUS 诊断

在配置时或出现故障时，系统会显示 PROFIBUS 的状态，以便进行诊断。该诊断窗口只用于提供信息，而不能作任何修改。

PROFIBUS 接口

- DP1 X126
- DP2 X136
- 集成的 DP

显示	含义/信息
状态	
总线状态	POWER_ON (0):控制系统开机后的状态 OFFLINE (1):已进行初始化 STOP (2):根据硬件配置(SDB)启动 CLEAR (3):PROFIBUS 从站已根据硬件设计(SDB)设定参数和配置，并以零输出数据参与到循环数据交换中 OPERATE (4):和 PROFIBUS 从站之间的循环数据交换正在运行 ERROR (9):检测到严重故障（例如：无效或错误的 SDB） (<n>) 与总线状态的 BTSS 变量值相同。
总线配置	
S7 子网 ID	PROFIBUS 子网的 S7 子网 ID
波特率	传输速度，单位 MBd
周期时间	配置的总线循环时间，单位：ms；它同时定义了位置控制器周期
同步分量 (TDX)	配置的一个 PROFIBUS-DP 循环内周期性数据交换的时间段，单位：ms
PROFIBUS 诊断/从站	
从站地址	DP 从站的配置编号
分配	DP 从站是否分配给 NC 或 PLC 的信息。 NC 例如：一个或多个由 NC 控制的驱动 PLC：例如：I/O 外设或由 PLC 控制的轴 NC/PLC（集成的 DP）
通讯状态	通讯状态信息，是否能识别总线上的 DP 从站 绿色：在 PROFIBUS-DP 上已检测到 DP 从站，并且和相应组件（NC 和/或 PLC）的数据交换正常 红色：故障/没有通讯

显示	含义/信息
和 NC 的同步	总线上的 DP 从站是否和 NC 同步运行。 绿色：PROFIBUS-DP 上，DP 从站和 NC 同步运行，即：等时的数据交换 红色：故障/没有通讯 灰色：DP 从站没有分配给 NC，而是分配给 PLC
槽数量	DP 从站内设计的槽数量

更多信息

有关配置 PROFIBUS 网络接口属性的更多信息参见

CNC 调试手册：NCK、PLC、驱动。

步骤



1. 选择操作区域“诊断”。





2. 按下菜单扩展按键和软键“TCP/IP 总线”。

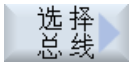
   “PROFIBUS 诊断...”窗口打开。





3. 如果配置了多个 PROFIBUS 接口，请按下软键“总线 +”或“总线 -”，选择所需的配置。

   -或者-



   按下软键“选择总线”，从列表中选择所需配置。



   按下软键“确认”。

16.9.1 显示 DP 从站的详细信息

此处可以显示所选 DP 从站插槽的详细信息。

显示	含义/信息
从站	
地址	在子窗口“PROFIBUS 诊断/从站”下选择的 DP 从站
通讯状态	DP 从站的 NC 或 PLC 分配信息
和 NC 的同步	DP 从站的 NC 同步过程信息 如不存在 DP 从站的同步值，则不进行显示。
槽	
编号	DP 从站的槽编号
I/O 地址	PLC I/O 地址范围内、分配给该槽的 I/O 地址 对于 NC 轴，设定值和实际值必须始终定义相同的 I/O 地址。
类型	指出插槽是输入、输出还是诊断插口 如果插槽分配给了一个 NC 轴，则输出始终视为设定值，输入始终视为实际值。
长度（字节）	STEP7 I/O 地址范围内预留给该插槽的 I/O 长度
通讯状态	插槽的当前通讯状态 如不存在 NC 轴的值，则不进行显示。 绿色： NC 正在使用插槽，通讯激活 红色： NC 正在使用插槽，当前通讯未激活 灰色： 无 NC 轴
机床轴	机床数据中定义的该槽的名称 如槽未分配给 NC 轴，则不进行显示。
报文类型(NC)	显示分配的报文类型 如未在 NC 机床数据中分配报文类型，则不进行显示。

步骤



1. 选择“诊断”操作区域。





2. 按下菜单扩展按键和软键“TCP/IP 总线”。
   “PROFIBUS 诊断...”窗口打开。

3. 选择您需要了解详细信息的 PROFIBUS 配置。



4. 按下软键“详细”。
   “PROFIBUS 诊断...详细信息”窗口打开。

16.10 驱动系统

16.10.1 显示驱动状态

状态显示

可在窗口“驱动系统诊断”中获得驱动设备及其所属驱动对象的状态信息。 状态符号的含义如下：

显示图标		含义
	绿色	驱动对象处于循环运行中，没有识别出故障。
	黄色	驱动对象识别出略微严重的问题，例如：出现警告或缺少使能。
	红色	驱动对象识别出严重问题，例如：出现报警。
	灰色	不能确定此驱动对象的驱动状态。
#	特殊字符	读取数据时出错。

步骤



1. 选择操作区域“诊断”。



2. 按下菜单扩展键和软键“驱动系统”。



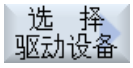
3. “驱动系统诊断”窗口打开。
   在窗口标题中给出了所选驱动设备的名称。



   按下软键“驱动对象 +”或者“驱动对象 -”。
   将选定下一 (+) 或者前一 (-) 驱动对象..



   -或者-



   按下软键“选择驱动设备”。
   窗口“选择驱动设备”打开。

4. 通过选择栏选择需要的驱动设备，并按下软键“确认”，确认选择。

参见

显示驱动对象的详细信息 (页 348)

16.10.2 显示驱动对象的详细信息

驱动系统诊断 - 详细信息

显示驱动对象的下列信息：

- 运行显示(r0002)
  - 显示驱动对象的运行显示（参数 r0002）。
- 缺少使能 (r0046)
  - 除了没有使能信号的驱动对象，例如：控制单元，所有使能都被诊断。 缺少的使能将显示在画面的子窗口中。

- 调试参数过滤器
  - (p0009) – 控制单元
    显示控制单元参数“设备调试参数过滤器”(p0009)的数值。
  - (p0010) – 其他驱动对象
    显示参数“驱动对象调试参数过滤器”(p0010)的数值。
- 当前故障
  显示驱动对象的当前故障编号；如果驱动对象上不存在故障，则显示“当前无故障”。
  可在另一个窗口中概览待处理的故障，其中包含如下信息：
  - 报警号
  - 时间点： 日期和时间
  - 报警文本
- 当前警告
  显示驱动对象的当前警告号；如果驱动对象上不存在警告，则显示“当前无警告”。
  可在另一个窗口中概览待处理的报警，其中包含如下信息：
  - 报警号
  - 时间点： 日期和时间
  - 报警文本

标记的含义

显示图标		含义
	绿色	驱动对象的相关值表明没有识别出故障。
	黄色	驱动对象的相关值表明识别出略微严重的问题，例如：出现警告或缺少使能。
	红色	驱动对象的相关值表明出现严重的问题，例如：出现报警。
	灰色	不能确定此驱动对象的驱动状态。
#	特殊字符	读取数据时出错。

步骤



1. 选择操作区域“诊断”。



2. 按下菜单扩展键和软键“驱动系统”。



“驱动系统诊断”窗口打开。

3. 选择您需要了解详细信息的驱动对象。



4. 按下软键“详细信息”。

“驱动系统诊断-详细信息”窗口打开。

在窗口标题中显示驱动对象的名称。



按下软键“故障”。

窗口“驱动设备故障”打开，显示一张已输出故障的清单。



按下软键“警告”。

窗口“驱动设备警告”打开，显示一张已输出警告的清单。



若您的控制系统中连接了 Safety Integrated，则可显示待处理的 SI 消息。

16.11 远程诊断

16.11.1 匹配远程诊断

根据运行模式不同，远程诊断可由服务提供者或者控制系统发起。

若由控制系统发起，则需使用软键“Require remote diagnosis”（要求远程诊断）。

在文件“tcu.ini”的 [PINGSERVICE] 段中确定软键“Require remote diagnosis”（要求远程诊断）的显示。

- 变量 PingServiceMode
  0 = 禁用
  1 = 使能

本段的其它变量用于设置此服务要求电文的相关数据。

- 变量 PingServerIP
  服务提供商的 IP 地址，向该地址发送电文，比如 PingServerIP = 128.45.234.19
- 变量 PingTransmissionData
  已发送要求电文中的数据内容文本，
  比如 PingTransmissionData = 帮助机床 35
- 变量 PingTransmissionPeriod
  电文发送的持续时间，单位为分钟，
  比如 PingTransmissionPeriod = 5

- 变量 PingTransmissionInterval
  重复发送电文的间隔时间，单位为秒，
  比如 PingTransmissionInterval = 15
- 变量 PingServerPor
  制造商 Ping 服务的端口号，
  比如 PingServerPort=6201

16.11.2 通过连接在 X127 上的远程服务适配器 IE 进行远程诊断

通过调制解调器使用连接在 X127 上的远程服务适配器 IE，即 TS 适配器，进行远程诊断时，须完成以下设置：

TS 适配器中的必要设置

- 适配器的 IP 地址：192.168.215.30，子网掩码 255.255.255.224
- 呼叫方的 IP 地址：192.168.215.29
  这两个地址也保存在“tcu.ini”文件中，为标准设置。若需使用其它的地址，需在“tcu.ini”文件中对条目“X127ModemIP”和“X127RemoteAccessIP”作相应修改。
- 在防火墙中打开 S7 通讯、http、HTTPS、SSH 以及 VNC（端口 5900）的端口。

远程诊断的调制解调器接口

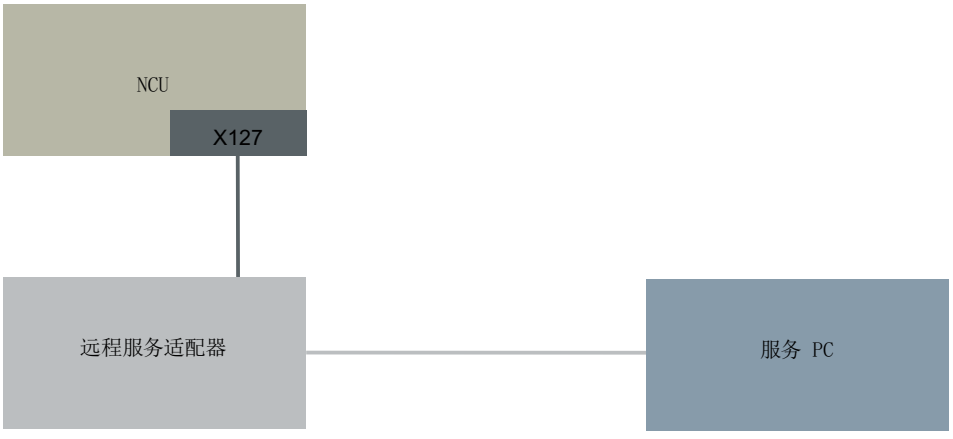


图 16-1 NCU 上含 SINUMERIK Operate 的简单装置（以及带 TCU 的操作站）

完成以上设置和预设数值后，即可使用本配置通过 TS 适配器实现远程诊断。

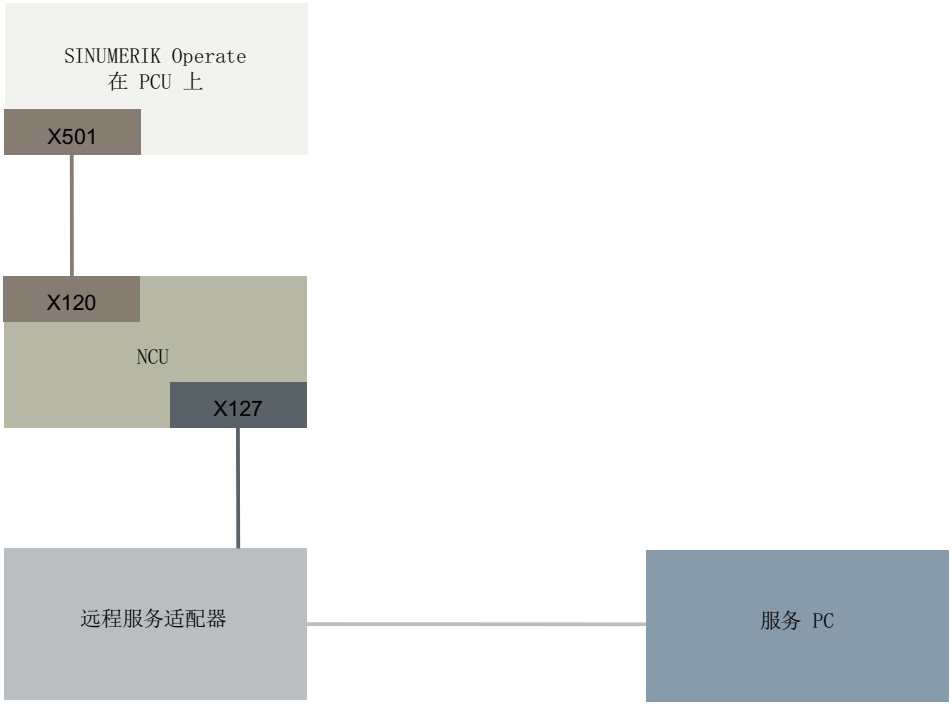


图 16-2　PCU 上含 SINUMERIK Operate 的简单装置

完成以上设置和预设数值后，即可使用本配置通过 TS 适配器实现远程诊断。在 PCU 上运行 SINUMERIK Operate 时，必须关闭 NCU 上的 SINUMERIK Operate。

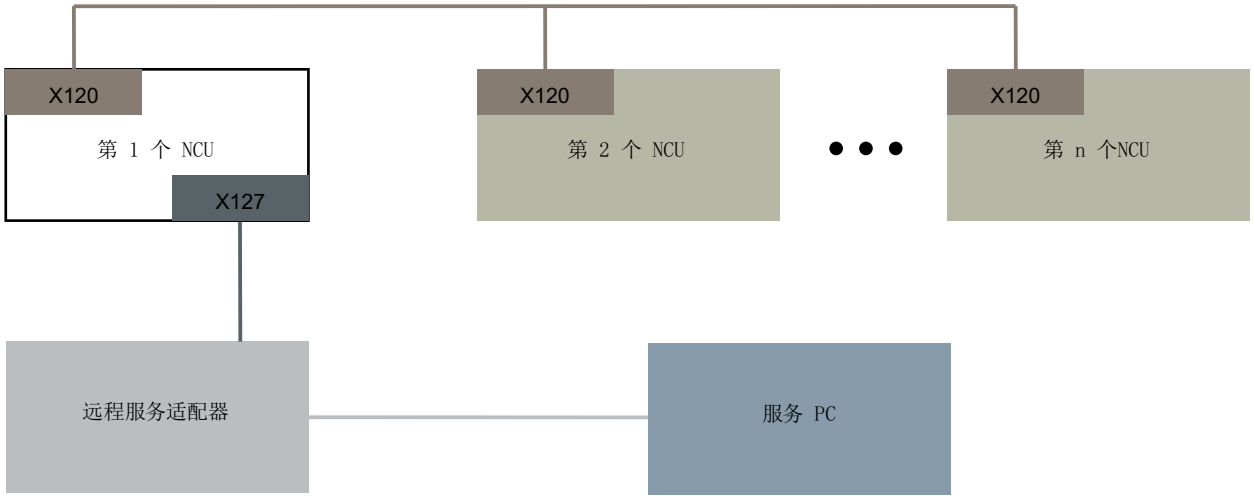


图 16-3　含多个 NCU 和一个固定调制解调器的装置

只须选中 X127 接口上接有 TS 适配器的 NCU 选件"Access MyMachine /P2P"。只有在该 NCU 中的 SINUMERIK Operate 上才可通过操作允许调制解调器访问，并执行"请求远程操作"功能。允许调制解调器访问后，服务 PC 便可访问所有的 NCU 及其操作软件。为了使各站

（NCU 上的 SINUMERIK Operate）与相应本地 PLC 协调合作，可在每个站上单独操作和设置用于控制远程访问的操作画面。

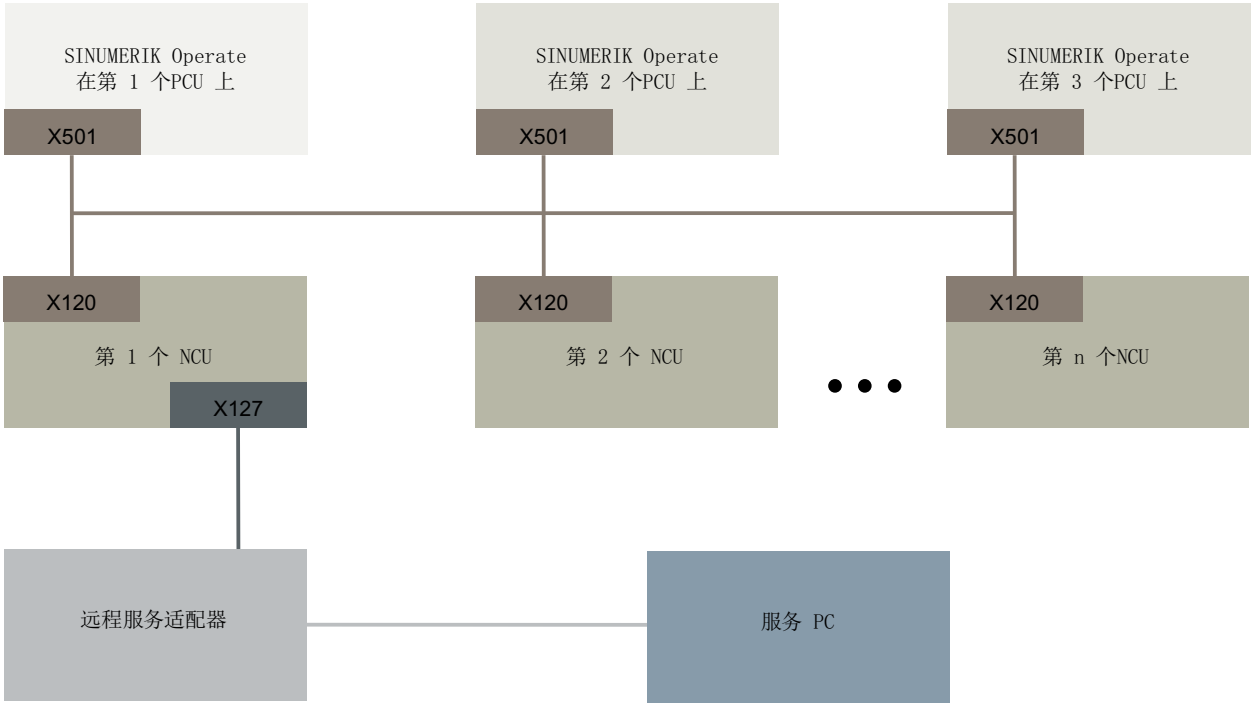


图 16-4　含多个 NCU、多个 PCU 和一个固定调制解调器的装置

只须选中 X127 接口上接有 TS 适配器的 NCU 选件“Access MyMachine /P2P”。只有在指定给该 NCU 的 PCU 上的 SINUMERIK Operate 上，才可以通过操作允许调制解调器访问，并执行“请求远程操作”功能。允许调制解调器访问后，服务 PC 便可通过设备网络访问其他所有 NCU 及其操作软件。为了使各站（NCU 上的 SINUMERIK Operate）与相应本地 PLC 协调合作，可在每个站上单独操作和设置用于控制远程访问的操作画面。

16.11.3　远程访问的 PLC 控制

可在 PLC 用户程序中通过如下设置控制外部察看者。

DB19.DBX1.0	PLCExtViewerReject		
值	= 0	= 1	
含义	允许远程操作	不允许远程操作	
DB19.DBX1.1	PLCExtViewerMode		
值	= 0	= 1	已忽略

含义	远程操作的操作权限	远程操作中仅察看	-
结果	远程操作的操作权限	远程操作中仅察看	不允许远程操作

16.12 跟踪

“跟踪”（Trace ）为您提供了示波器功能，能够用形象的图形显示以下数据（信号）：

- NC 的通用数据
- PLC 数据

跟踪会话

该功能用于故障搜索和消除、机床效率分析以及过程效率分析。 会在会话中记录事件发生前或发生后的数据（信号）。

图形显示

记录下来的数据以曲线图的形式显示。

您可以自定义记录类型。

保存和载入数据

可将记录下来的数据保存在一个 XML 文件中。 保存下来的数据可以再次载入，以便之后开展分析。

记录的基础： 信号（数据）

信号（数据）决定该记录的内容：

- NC 变量（轴数据、驱动数据和通道数据等）
- PLC 数据

周期性事件


插补周期 (IPO 周期)

PLC 周期 OB1


位置控制周期（伺服周期）

非周期性事件



启动会话

- 通过触发事件记录
- 手动启动和停止记录

分析结果

对记录结果进行分析时可以使用如下功能:

- 设置曲线图的比例
- 缩放曲线图
- 通过标识定义一段曲线

16.12.1 一般步骤

步骤

- 按以下步骤建立跟踪会话文件：
  - 选择变量，然后确定它们的显示方式
  - 调节图形窗口，以及保存模态
  - 设置触发事件的属性，确定记录开始或结束的时间
- 保存本次记录，以及设置的属性
- 分析结果，若需要进一步的细节，可以设置显示画面，例如：缩放，滚动，剪切等。

参见

跟踪的选项 (页 368)

保存跟踪会话文件 (页 356)

用于跟踪的变量 (页 359)

跟踪设置 (页 365)

分析跟踪 (页 370)

16.12.2 跟踪会话

16.12.2.1 创建跟踪会话文件

步骤



1. 选择操作区域“诊断”。



2. 按下菜单扩展键和软键“跟踪”。







3. 按下软键“载入跟踪”和“新建”。
   窗口“新建会话”打开，其名称默认为“会话”。文件格式自动设为“.xml”。
4. 添加文件名或输入一个新文件名。



5. 选择一个变量类型并按下软键“确认”。
   此时跳回窗口“选择用于跟踪的变量：....”。
   在标题行会显示新建会话的名称。

参见

筛选/搜索变量 (页 359)

16.12.2.2 保存跟踪会话文件

可使用两种方法保存跟踪会话：

- 变量、设置：只保存变量和设置。
- 变量、设置和已记录的数值：会保存变量、设置以及所记录的数值。

说明

保存选项

如果已在“跟踪”→“选项 (页 368)”下激活了复选框“以两种格式保存文件”，会话文件则会以 xml 格式和 csv 文件的格式保存。

步骤



1. 已编辑了一个有效的跟踪会话。



2. 按下软键“保存跟踪”。

   窗口“保存跟踪：会话...”打开，提供保存选项。



3. 激活相应的复选框后按下软键“确认”。

   窗口“保存跟踪”打开，在“跟踪文件名称”栏中输入文件名。文件格式自动设为“.xml”，无法更改。



4. 按下软键“确认”，保存会话文件。

   会话文件已经存在时，会跳出一条安全询问。

5. 希望覆盖已有会话文件时，再次按下软键“确认”。

   -或者-



   不希望覆盖已有会话文件时，按下软键“另存为”。

   您可以采用建议的名称或自己命名。

   -或者-



   若需要将会话文件保存至标准目录“跟踪”之外的其它目录，按下软键“更改目录”。

   文件树形图打开，您可以在已配置的驱动器之间选择。



   若需创建新的目录，按下软键“新建目录”。

   窗口“新建目录”打开，你可以为新建目录命名。



   按下软键“确认”。

   窗口“保存跟踪”打开，显示目标目录的名称和会话文件。

   可以更改会话文件的名称。



6. 再次按下软键“确认”，保存会话文件。

   -或者-

   具有该名称的会话文件已经存在时，会跳出一条安全询问。

16.12.2.3 加载跟踪会话文件

您可以加载已保存的会话文件，以重新启动此文件或者显示跟踪记录。

设置	含义
变量、设置	可以重新启动一个已创建的跟踪会话。
变量、设置和已记录的数值	可以查看跟踪会话的记录、修改变量以及在必要时重新启动。

前提条件

跟踪记录的会话文件已经存在。

步骤

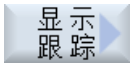


1. 已选择了功能“跟踪”。



2. 按下软键“载入跟踪”。

   窗口“载入会话文件：选择文件”打开。



3. 从相应目录中选择所需的会话文件（比如“跟踪”）后按下软键“显示跟踪”。

   窗口“载入跟踪：会话...”打开。

4. 若需要同时启动跟踪会话，则选定复选框“变量、设置”。



5. 按下软键“确认”和“启动跟踪”，以重新启动记录。



-或者-

若需要显示跟踪会话，则选定复选框“变量、设置和已记录的数值”。



按下软键“确认”。

所选择会话的数据将载入并显示在窗口“跟踪”中。

16.12.3 用于跟踪的变量

16.12.3.1 筛选/搜索变量

查找 PLC/NC/伺服变量

为了限制 PLC/NC/伺服变量的选择，可以设置筛选和/或输入一个查找初始值。

筛选：	显示下列变量：
系统变量	系统变量
NC	NC 变量
轴	轴变量
通道	通道变量
运行方式组	BTSS 格式的运行方式组变量
伺服	伺服跟踪变量
全局用户数据	GUD，例如：SGUD, MGUD, UGUD, GUD4, ...
PLC	PLC 变量
PLC 报警	制造商专用的 PLC 报警信号
所有（无筛选）	按字母顺序排列的所有变量

插入变量

按照不同条件进行“筛选/查找”时，初始值有较大差别。例如：输入以下初始值，以插入变量 $R[0]：

- 如果按照“系统变量”筛选，则初始值为 0。
- 如果按照“全部（无过滤）”筛选，则初始值为 1,此时会用 BTSS 输入方式显示所有信号。

在选择变量时，仅当相关定义文件已激活时，机床数据中的 GUD 才会显示在查找窗口中。否则必须手动输入查找的变量，例如：GUD/SYG_RM[1]

以下机床数据适用于所有变量类型 (INT、BOOL、AXIS、CHAR、STRING)：
MD18660 $MN_MM_NUM_SYNACT_GUD_REAL

查找驱动变量

此处通过所选驱动对象限制查找。在查找窗口中可通过输入字母数字值限制选择。还可通过互联“全部”或“推荐的”限制选择。

步骤

在开展一个跟踪会话前，首先在窗口“选择用于跟踪的变量：会话...”中选择需要记录其信号的变量。



1. 可在窗口“选择用于跟踪的变量：会话...”中显示跟踪会话。
2. 将光标移动到“变量”栏的一个空行中。

   -或者-

   将光标移动到“变量”栏中的一个需要被替代的变量上。

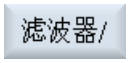


3. 按下软键“插入变量”。

   “选择变量”窗口打开。

   提示：

   需要选择一个驱动参数时，首先选择一个驱动对象（DO），然后再从列表中选择所需参数。



4. 按下软键“过滤/搜索”，开始查找特定对象。

   在下拉菜单“过滤”中选择一个变量范围：

   -或者-

   直接在输入框“搜索”中输入所需的变量。

   光标会跳转到找到的目标上。



   按下软键“删除搜索”，结束查找。



5. 按下软键“插入”，将所需变量插入到跟踪会话中。

   窗口“变量属性”打开，可从下拉框中选择所需数值，例如：通道号、驱动总线号等。

   -或者-



   按下软键“替换”，替换所选变量。



6. 按下软键“确认”，插入或替换变量。

   每个变量都会自动获得一个参考号，它显示在窗口上方。这个编号自动分配。一个变量被删除时，它后面的变量自动重新编号。

7. 现在选择显示所需的设置，如颜色、线形、曲线图和显示等。

编辑跟踪会话中的变量

快捷键	作用
Ctrl + X 或者右击鼠标，弹出右键菜单“剪切”	按下快捷键或打开右键菜单，剪切变量。
Ctrl + C 或者右击鼠标，弹出右键菜单“复制”	按下快捷键或打开右键菜单，复制变量。
Ctrl + V 或者右击鼠标，弹出右键菜单“粘贴”	按下快捷键或打开右键菜单，粘贴变量。

16.12.3.2 选择变量的属性

属性

需要开展跟踪会话时，选择一个变量，并指定它的属性：

列	含义
变量	变量的地址

属性	含义
注释	• 所选变量的名称 • 指定变量的一般性说明。
颜色	曲线图的颜色选择 在创建变量时会自动从调色板中选择下一个颜色。

属性	含义
线形	曲线图的线形选择 • 无线 • 实线 • 虚线 • 点划线 • 双点划线 • 点线 • 阶梯线 XY • 阶梯线 XY • 星线 • + 线 • X 线
显示	确定是否在窗口“跟踪”中显示变量的曲线图。 不选中这个复选框时，不显示曲线图。
事件	选择触发信号记录的事件：
	• 插补周期 (IPO 周期) • 插补周期(IPO2) • 位置控制周期（伺服周期） • PLC 周期 OB1 • 驱动跟踪速率 • 几何轴启动/方向转换 • 几何轴停止 • 机床轴启动/方向转换 • 机床轴停止 • NC 启动（程序运行中） • NC 停止（程序已运行） • 数据记录启动

属性	含义
	• 1 类程序段开始、程序段结束（所有程序级，无中间程序段） • 2 类程序段开始、程序段结束（所有程序级，有中间程序段） • 3 类程序段开始、程序段结束（所有主程序，无中间程序段） • 程序段开始、程序段结束、程序段搜索（所有程序级，无中间程序段） • 几何轴启动/切换方向（第 2 事件） • 几何轴停止（第 2 事件） • 2 类程序段开始（第 2 事件：所有程序级，无中间程序段） • 2 类程序段结束（第 2 事件：所有程序级，有中间程序段） • 1 类程序段开始，程序段预处理（第 2 事件） • 1 类 OEM 测试事件（非周期式、程序段预处理） • 2 类 OEM 测试事件（非周期式、程序段主处理）
	• 激活/取消激活同步动作 • 同步动作已被触发（满足条件） • 报告的报警 • 软键“取消”被按下 • 程序级切换 • 程序段结束（编译器） • 跟踪结束（最后一个事件！） • 跟踪启动
	• WRTPR 零件程序指令 • WRTPR 零件程序指令（程序段搜索） • 换刀 • 刀沿切换 • 换刀（程序段搜索） • 刀沿（程序段搜索） • 启动触发事件已发生 • 停止触发事件已发生 • 程序段结束（程序段预处理） • 程序段结束（第 2 事件：程序段预处理） • WRTPR 零件程序指令（程序段预处理） • 换刀（程序段预处理） • 刀沿切换（程序段预处理）
事件通道	此处可以确定 SINUMERIK 上特定事件所在的通道。 对于通道通用的事件，此栏为空。

属性	含义
位标记	如果是整数变量，可在此栏中指定位标记。 位标记会首先和对应的信号值逻辑“与”运算，然后显示。 设置标记后，所选择位将向右移动，所以看起来该位或者这些位似乎都是从零位开始的。 即：若除位 7 外，所有的位都设置了标记，则产生的整数数值只能为 0 或者 1，而不可能为 0 或者 128。 若除了位 7 和位 0 外所有的位都设置了标记，则所得整数结果为 0、1、2 或 3，而不会是 0、1、128 或 129。
小数位	指定轴名称上显示的小数位位数。
坐标轴	坐标轴显示在图形窗口的左边或右边或者根本不显示。
Y 轴显示	输入的数值或 0
比例系数	确定比例尺
单位	显示尺寸单位，例如：mm/min。尺寸由系统给定，无法更改。

16.12.3.3 显示变量的详细信息

详细信息窗口中会显示重要信息和设置一览：

- 变量地址
- 含有变量描述的注释
- 事件
- 通道、轴、访问级别、运行方式组等
- 缩小的图形窗口，含有触发曲线图记录的事件及曲线图设置，如颜色、线形等。

步骤



1. 您位于窗口“选择用于跟踪的变量：...”。



2. 将光标定位在所需变量上并按下“详细”软键。
   在窗口下方显示了局部画面，其中包含了最重要的信息以及图形。



3. 再次按下软键“详细信息”，退出局部画面。

16.12.4 跟踪设置

16.12.4.1 跟踪设置（PLC、NC、伺服）

在窗口“跟踪-设置（PLC、NC、伺服）”中确定跟踪会话的启停条件。

启动跟踪的条件

通过软键启动跟踪	按下软键“启动跟踪”后，立即开始记录。
有变量时	会话由变量激活。通过软键“插入变量”或“快捷列表”选择变量。
• 条件和值	确定激活会话后启动跟踪记录的值。 提供以下值：= =, ≠, >, >=, <, <=, 升序、降序、更改。
数据采集 [s]	
确定所记录数据的存储位置： • 采用“NCK”时，这些数据会一直保存在 NC 工作存储器中，直到记录停止。然后会话文件才保存在 CF 卡上。 • 采用“CF 卡”或“硬盘”时，记录期间数据仍会保存到 CF 卡或硬盘（如 PCU）上。这样便可以处理更多的信号。	

停止跟踪的条件

通过软键停止跟踪	按下软键“停止跟踪”后，立即开始记录。
时间届满后	确定跟踪记录运行至结束的时间周期。
存储器存满时	达到规定的数据量 [kByte] 后，跟踪会话结束。
有变量时	记录由变量停止。通过软键“插入变量”或“快捷列表”选择变量。
• 条件和值	确定会话激活后结束跟踪记录的数值。 提供以下值：= =, ≠, >, >=, <, <=, 升序、降序、更改。
• 后续运行时间 [s]	确定会话结束后记录仍运行的时间长度。
• 超出存储极限 [kByte]	确定覆盖数据的最低存储极限（环形缓冲器）。

说明

超出最长持续时间时记录的数据可能丢失

使用参数设置“停止跟踪 - 如有变量 - 达到存储极限时覆盖”进行跟踪时，不得超出记录的最长起止持续时间。

如果超出最长持续时间，停止事件无法生效并且记录也无法正常结束。

最长持续时间可通过以下方式测定：

通过临时设置“停止跟踪 - 当时间到期”可以测定，设定的记录每分钟需要多少存储空间，例如 602 kB/分钟。

那么，记录起止期间的最长时间（分钟）则可按以下方式计算：

2^{31} (2147483648) 字节 / 设定记录所需存储空间（字节/分钟）

602 kB = 602 * 1024 字节 = 616448 字节/分钟，也就是 2147483648 字节 / 每分钟
616448 字节 = 3483 分钟 = 58 小时

说明

后续运行时间和存储极限无效的停止条件

如果选择了结果为“PLC 周期 OB1”的 PLC 信号用于跟踪，那么后续运行时间和存储极限的设置均无效。

跟踪前至少要将一个 PLC 信号结果设为“插补周期（IPO 周期）”，如此才能记录后续运行时间和存储极限的数据。

16.12.4.2 跟踪设置（驱动）

在窗口“跟踪-设置（驱动）”中确定跟踪会话的启停条件。

启动跟踪的条件

<table>
<tr><td>通过软键启动跟踪</td><td colspan="3">按下软键“启动跟踪”后，立即开始记录。</td></tr>
<tr><td>有变量时</td><td colspan="3">如果所选变量符合条件和给定的值，按下软键“启动跟踪”后，记录激活。
提供以下条件：
• 上升沿
• 下降沿
• 位掩码
选择“位掩码”条件时，系统会显示软键“位掩码”，用于选择所需的位。可以输入一个用于启动跟踪记录的值（十进制），与通过位掩码可以设置的最大值（十六进制）无关。
示例：
设置了以下位时，跟踪记录启动：位 0 = 0；位 1 = 1；位 2 = 0；位 3 = 1
<table>
<tr><th>位掩码 = fH</th><th>条件</th><th>值（十进制）</th></tr>
<tr><td>变量：CU/r722/3.3.1</td><td>位掩码
☑ 位 0
☑ 位 1
☑ 位 2
☑ 位 3</td><td>10</td></tr>
</table></td></tr>
<tr><td>预触发时间 [s]：</td><td colspan="3">输入预触发时间在功能上等同于参数 p4722“触发延时跟踪”。
• 如果预触发时间为正值，则在触发事件前启动记录。
• 如果预触发时间为负值，则在触发事件后延长记录。
输入预触发时间时须考虑整个记录时间：
• 如果输入的预触发时间大于整个记录时间，系统会取消该输入。
• 如果输入的预触发时间过大，系统可能不会记录触发事件。</td></tr>
<tr><td colspan="4">将数据保存到控制单元上</td></tr>
<tr><td>驱动跟踪速率 [µs]：</td><td colspan="3">通过软键“采样速率”输入一个系数，设置有效的采样速率。采样速率越高，记录时间也越长。</td></tr>
</table>

停止跟踪的条件

时间届满后	确定记录运行至结束的时间周期。
存储器存满时	达到规定的数据量 [kByte] 后，记录结束。

16.12.4.3 跟踪的选项

在窗口“跟踪 - 选项”中您可以确定一个跟踪会话的属性。

- 调整图形窗口
  - 显示水平栅格线：确定是否显示水平栅格以及栅格的颜色。
  - 显示垂直栅格线：确定是否显示垂直栅格以及栅格的颜色。
  - 背景：选择图形窗口的背景色。
- 确定保存选项
  - 若需在覆盖跟踪会话前总是弹出安全询问，则激活复选框“确认文件覆盖”。若取消此复选框，则覆盖跟踪会话文件时不会有安全询问。
  - 激活复选框“以两种格式保存文件”，除 xml 会话文件外，再将数据保存为 csv 文件 (页 356)。但无法再次读取 csv 格式的文件。
  - 勾选复选框“文件另存为 HMI-Advanced Servo-Trace-Format”，可另外将导出数据保存为 HMI-Advanced Servo-Trace-Format 格式。可参考以下文件格式：
    svt 文件（跟踪会话的设置）
    st1 文件（第一个和第二个跟踪会话的变量）
    st2 文件（第三个和第四个跟踪会话的变量）

步骤



1. 您位于窗口“选择用于跟踪的变量：...”中，并选择了一个跟踪会话。



2. 按下软键“选项”。
   “跟踪 - 选项”窗口打开。



3. 选择所需的设置并按下软键“确认”，采用该设置。

16.12.4.4 启动跟踪

在窗口“跟踪”中展示了一张曲线图，此图含有为跟踪会话选择的变量数据。

该图由两个直角坐标轴构成：

- 时间轴
  即水平 X 轴，提供时间值。
- 数值轴
  即垂直 Y 轴，提供精度（振幅）。

说明

PLC 变量的真实值显示

若跟踪中的值格式显示错误，则在窗口“为跟踪选择变量”中通过参数设置 :REAL 对变量指示进行扩展，例如 DB21.DBD76:REAL。这样便在跟踪中显示变量的真实值。

记录跟踪会话

- 手动启动和停止记录
- 使用触发事件启动和停止记录

步骤



1. 已经将跟踪会话所需的变量整合在一起。



2. 按下软键“显示跟踪”。
   “跟踪”窗口打开。记录被载入，载入进度会显示在一个进度窗口中。



3. 需要立即开始记录过程时，按下软键“启动跟踪”。

   -或者-

   通过特定触发条件启动记录。



4. 需要手动结束记录时，按下软键“停止跟踪”。

   -或者-

   通过特定触发条件停止记录。

16.12.5 分析跟踪

16.12.5.1 设置跟踪的视图

为了更加精确地分析记录下来的信号，您可以编辑曲线图的视图：

- 选择/图形
  - 显示图形：除了图形外，变量数据的显示还附加了注释、编号和颜色。
  - 选择：确定在图形窗口中显示哪些变量。
- 比例
  - 确定最大、最小 X/Y 值的比例系数
- 缩放
  - 缩放视图
- 光标：光标视图能够精确测量一条或两条曲线。
  - 光标 A
  - 光标 B
  - 两个光标
  - 沿曲线：光标一直沿着曲线移动。
  - 点模式：光标直接跳转到不同值。
  - 峰值模式：光标从一个峰值跳转到下一个峰值。
  - 谷值模式：光标跳转到最低值。

前提条件

图形窗口“跟踪”已打开且跟踪记录已完成。

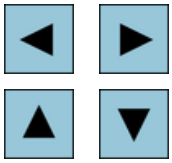
编辑曲线



可使用按键 <TAB> 切换至下一个变量。



使用 <SHIFT> + <TAB> 返回。



通过光标键可以设置标记，移动到某个值，并定义缩放区域。

16.12.5.2 选择变量

您可以将通过复选框选中的变量显示为图形。

图形
变量的编号
轴名称
变量的注释

显示图形和选择变量



1. 在图形窗口“跟踪”中，一个记录打开。



2. 按下软键“图形/选择”。
   图形窗口缩小一半。 在窗口左侧中显示了变量（注释）和复选框。
3. 通过这些复选框勾选希望作为曲线显示的变量，或取消曲线显示。



4. 再次按下软键，退出曲线区域。

16.12.5.3 设置曲线图的比例

通过“比例”，您可以更加精确地测量 X 时间轴和 Y 数值轴上选中的变量。 您可以任意缩放曲线图，并保存修改。

设置	含义	
X 轴最小值或 X 轴最大值	X 时间轴的最大值和最小值 在重新开始“跟踪”后，会使用这些值进行记录。	
	X 轴固定比例	保留 X 时间轴的值。
	穿孔带记录仪模式	在记录期间采用 X 时间轴的值。
Y 轴最小值或 Y 轴最大值	Y 数值轴（幅度）的最大值和最小值。	
偏移或 分辨率	确定幅度的有效值（平均值）。 比例按照系数 10 变化。	

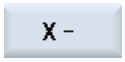
调整时间轴和数值轴的比例



1. 在图形窗口“跟踪”中，一个记录打开。



2. 按下软键“比例”，调节曲线图的比例。



输入窗口“选择 X-Y 比例系数”打开，显示 X 时间轴的值。



3. 需要隐藏该值时，按下软键“X 时间轴”。

-或者-



按下软键“曲线 Y 轴”显示 Y 数值轴的值。



需要隐藏该值时，再次按下软键。

-或者-

您选择了两根轴。





4. 按软键“比例 +”或者“比例 -”，直至达到所需的比例为止。

-或者-

直接在输入窗口中输入比例值，选择所需属性。



5. 按下软键“比例”，保存设置。

-或者-



按下软键“多图全窗口”，全屏显示所选的多个曲线图。



按下软键“单图全窗口”，全屏显示所选的曲线图。

16.12.5.4 缩放曲线图

通过“缩放”，您可以查看所选曲线段的细节。 确定了曲线段后，您便可以开始缩放。

步骤



1. 在图形窗口“跟踪”中，一个记录打开。



2. 若要放大或者缩小曲线图，按下软键“缩放”。





3. 选择软键"X 时间轴"或者"Y 数值轴"。



4. 若需要每个曲线图在图形窗口中依次单独显示，可按下软键"纵向排列"或"平铺显示"。

或



-或者-



若需调整图形窗口的所有曲线图，可使用软键"层叠显示"。
由于现在所有的曲线占满了整个窗口，因此曲线以层叠的方式显示。

-或者-



若只需要使 X 时间轴占满图形窗口，可按软键"X 轴最大"。

-或者-



若只需要使 Y 数值轴占满图形窗口，可按软键"Y 轴等分"或"平铺显示"。

或



-或者-



若需使所有曲线图的数值轴占满图形窗口，可使用软键"Y 轴最大"。
所有曲线都会占满层叠窗口，以层叠方式显示。



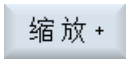
5. 按软键"缩放 +"或"缩放 -"，直至达到所需的显示大小为止。



缩放区域



按下软键"缩放区域"，确定特定区域。
显示一个边框。



按下软键"缩放 +"或者"缩放 -"，逐步调整大小。





按下软键"确认"，显示缩放区域。

16.12.5.5 定位光标

您可以借助光标获得定义的曲线段的精确数值。 此时，光标直接定位在曲线上。

步骤



1. 在图形窗口“跟踪”中，一个记录打开。



2. 按下软键“光标 A”。
   在该软键上显示变量的参考号和颜色，以及坐标轴的颜色。





3. 将“(A)”移动到您需要开始测量的曲线位置上。
   在提示条和状态条中会提示坐标位置。



4. 按下软键“光标 B”。
   在该软键上也显示变量的参考号和颜色，以及坐标轴的颜色。
   将“(B)”移动到您需要结束测量的曲线位置上。







5. 按下软键“两个光标”，显示 A 和 B 之间的范围。
   状态栏中会显示 A 和 B 的坐标，以及两者的差值“Δ”。



再次按下已经激活的软键“光标 A”和“光标 B”，取消选中的位置。
再次选中时，位置再次复位。





按下软键“截屏”，将“光标 A”和“光标 B”选中的区域占满整个图形窗口。



按下软键“曲线模式”，使光标沿曲线逐步移动。

16.12.5.6 获取测量值

您可以直接在曲线上获取以下值：

- 峰峰值
- 峰谷值

步骤

	在图形窗口“跟踪”中，一个记录打开。
	按下软键“点模式”，使光标一直沿曲线移动。
	按下软键“峰值模式”，使光标跳转到最高值。
	按下软键“谷值模式”，使光标跳转到最低值。
	按下软键“峰峰值模式”，只检查峰峰值。
	按下软键“峰谷值模式”，只检查峰谷值。

16.13 PROFIBUS/PROFINET 诊断和 AS-i 总线诊断

16.13.1 PROFIBUS/PROFINET

使用 PROFIBUS/PROFINET 诊断可获取快速概览，通过此概览可根据需要调用外部 DP 主站系统和 PROFINET IO 系统的状态信息和详细信息。

详细信息诊断的前提条件

使用了 HMI PRO RT 时，诊断会提供更多信息，例如：组件名称和订货号等。 这些信息都是离线数据，源自所属的 Step7 项目。

- 通过程序“HMIPRO CS”可建立离线数据。
- 通过程序“HMIPRO”可将数据传送到控制系统。



软件选件

如果除了状态信息，还需要显示出错模块的详细信息，则需要选件：“HMI PRO sl RT 操作软件”。

为了显示 PROFINET-IO 系统的详细信息，必须使用 HMI PRO 模块库中的 PLC 模块 FB_SL_COM。 将 DB449 作为相应的实例数据块。

说明

而 DP 主站系统的详细信息在未使用 FB_SL_COM 的情况下也可用。

DP 主站系统的诊断信息

可以配置的类型有：

1. 没有软件选件，没有 HMIPRO 离线数据
2. 有软件选件，没有 HMIPRO 离线数据
3. 有软件选件，有 HMIPRO 离线数据

16.13.2 显示 PROFIBUS/PROFINET 诊断

DP 主站系统的诊断信息

您可以在操作界面上显示由 HMIPRO 鉴别的现场总线诊断。 组件的状态是通过诊断符号表明的。

取决于您配置的类型，诊断信息有：

针对所有类型的信息		
状态		模块出现故障
		模块断开
		模块正常
		模块被禁用（只有 HMI PRO RT 才可以）
PROFIBUS 地址		PROFIBUS 地址信息

针对 2 类组件的其他信息	
PROFIBUS/PROFINET 模块诊断或通道诊断	
模块诊断	针对西门子的 DP/DP 耦合器和诊断重复器

针对 3 类组件的其他诊断信息	
标识	HMI 离线数据中模块的标识
名称	HMI 离线数据中模块的名称
订货号	HMI 离线数据中模块的订货号
IP 地址	HMI 离线数据中模块的 IP 地址（仅针对 PROFINET）
在单独的窗口中以图形显示整个硬件配置	

步骤



1. 选择操作区域“诊断”。

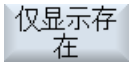




2. 按下菜单扩展键 2 次和软键“PB/PN 诊断”。
“PROFIBUS/PROFINET 诊断”窗口打开。

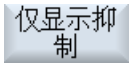
3. 选择所需的 DP 主站系统。
随即显示所有的诊断信息。





4. 按下软键“现有设备”，只显示现有模块的诊断数据。

-或者-



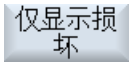
按下软键“显示禁用设备”，只显示被禁用模块的诊断数据。

-或者-



按下软键“显示完全故障设备”，只显示完全故障的模块的诊断数据。

-或者-



按下软键“显示局部故障设备”，只显示出现局部故障的模块的诊断数据。

16.13.3 配置 AS-i 总线

通过“AS 接口(AS-i)诊断”您可以迅速了解现有 AS 接口网络的概况。 由于持续不断的监控和故障查找，系统可以为您提供以下信息：

- 符合 AS-i 规范的状态旗标
- AS-i 从站的状态

前提条件

- 必须使用 HMI PRO 模块库中的 PLC 模块 FB_SL_COM 用于 AS-i 诊断。
- 同时还必须有配置文件“sltlprodiaghmi.ini”。

AS-i 组件

支持的 AS-i 组件有：

CP142-2	（型号 ID 1）
CP343-2	（型号 ID 2）
DP-AS-i-Link	（型号 ID 3）
DP-AS-i-Link-20E	（型号 ID 4）
DP-AS-i-Link-Advanced	（型号 ID 6）

诊断信息

在配置文件“sltlprodiaghmi.ini”中，必须定义所有需要诊断的 AS-i 组件。

条目		说明
[SKO]		
	ComboCount=x	下拉表中条目的数量(x)
[SKO_Index0]		1. 条目
	DiagType= 6	AS-i 组件的类型 ID（见上文）
	TypeSection=DP-AS-i-Link-Advanced 1	在下拉表中输出的 AS-i 组件的标识
	Logische_Adresse=43	硬件配置 AS-i 组件的输入地址
	AuftragsDB=449	采用功能块“FB_SL_COM”中的实例数据块

示例：sltlprodiaghmi.ini，具有 5 个组件

```
[SK0]
ComboCount=5

[SK0_Index0]
DiagType= 1
TypeSection=CP142-2
Logische_Adresse=43
AuftragsDB=449
```

```
[SK0_Index1]
DiagType= 2
TypeSection=CP343-2
Logische_Adresse=75
AuftragsDB=449

[SK0_Index2]
DiagType= 3
TypeSection=DP-AS-i-Link
Logische_Adresse=75
AuftragsDB=449

[SK0_Index3]
DiagType= 4
TypeSection=DP-AS-i-Link-20E
Logische_Adresse=75
AuftragsDB=449

[SK0_Index4]
DiagType= 6
TypeSection=DP-AS-i-Link-Advanced
Logische_Adresse=75
AuftragsDB=449
```

步骤

1. 从目录 /**siemens**/sinumerik/hmi/template/cfg/ 中复制文件“sltlprodiaghmi.ini”。
2. 将文件保存到目录：/**oem**/sinumerik/hmi/cfg 或 /**user**/sinumerik/hmi/cfg。
   如果该目录中已经存在该文件，只需在文件中添加一些条目。

3. 打开文件，然后输入 AS-i 组件的数据。
4. 保存并关闭文件。
   现在诊断信息显示在操作界面上。

16.13.4 显示 AS-i 总线诊断信息

AS-i 组件的诊断信息

您可以在操作界面上显示关于二进制传感器和执行器(AS-i) 的联网信息。



软件选件

勾选了选件“ HMI PRO sl RT 操作软件 ”时，还会为“DP-AS-i-Link-Advanced”显示额外的故障计数器。

诊断信息

符合 AS-i 规范的旗标	说明
离线	运行状态：离线
内部	运行状态：内部
EEPROM 正常	内部 EEPROM 正常
自动定址程序已被使能	自动定址已使能
外设故障	存在外设故障
主站上电 - 启动	主站上电后启动
离线完成	离线阶段完成
AS-i 电压过低	AS-i 电缆电压过低
CP 进入正常模式	CP 进入正常模式
运行模式	运行模式状态
自动定址程序执行完毕	自动定址执行完毕
自动定址程序允许	可进行自动定址
存在地址为 0 的 AS-i 从站	存在地址为 0 的从站
设定配置 = 实际配置	设定配置等于实际配置

状态	标识
从站正常	绿色
从站故障	红色
有从站，但没有配置	黄色

步骤



1. 选择操作区域“诊断”。





2. 按下菜单扩展键 2 次和软键“AS-i 诊断”。
   “AS-Interface 诊断”窗口打开。

3. 选择所需的 DP AS-Interface。
   随即显示所有的诊断信息。
   斜体数值表示故障计数大于零。

OEM 专用在线帮助 17

17.1 概览

除已有的各种在线帮助外，还可创建制造商专用在线帮助，并将其集成至 SINUMERIK Operate 中。

此种在线帮助使用 HTML 格式，即：由彼此链接的 HTML 文档组成。 可在特定的窗口中通过内容或关键词目录调用所搜索的主题。 与文档浏览器（比如 Windows 浏览器）相似，在窗口的左边部分显示选择概览，点击所需的主题后，会在右边窗口部分显示相应的说明。

不能相关上下文选择在线帮助页。

步骤

1. 创建 HTML 文件
2. 创建帮助手册
3. 将在线帮助集成到 SINUMERIK Operate 中
4. 保存帮助文件

其他使用情况

可以建立以下 OEM 专有扩展功能的在线帮助，以补充 SINUMERIK Operate 的在线帮助系统：

- 机床制造商**循环与/或 M 功能**的在线帮助，可扩展 SINUMERIK 控制系统的编程方式。 该在线帮助通过 SINUMERIK Operate 在线帮助“编程”来调用。
- 机床制造商 OEM 专用**变量**的在线帮助。 此在线帮助从 SINUMERIK Operate 的变量视图调用。

编程在线帮助

对于在线帮助的其它设计可使用“SINUMERIK HMI 编程包 sl”。 可使用此编程包用程序语言 C++ 为 NCU 7x0 上的 SINUMERIK Operate 开发标准语言应用程序。

说明

“SINUMERIK HMI 编程语言包 sl”是一个需要单独订购的软件选件。 对应的说明文档随编程语言包一起发货。

17.2 生成 HTML 文件

创建 HTML 格式的帮助文件。您可以将所有信息放在一份 HTML 文件中或者放在几份 HTML 文件中。

可以自己命名文件，但需遵守以下事项：

- 必须为 HTML 文件中的参照关系规定相关的路径。只有这样能保证引用不仅在开发计算机上有效，而且在载入系统后同样有效。
- 若需要在 HTML 文件中的特定点上通过链接进行跳转，必须定义所谓的锚点。
  HTML 锚点示例：
  `<a name="myAnchor">This is an anchor</a>`
- HTML 文档的内容必须保存为 UTF-8 编码格式。只有这样才能确保 HTML 文档在所有 SINUMERIK Operate 所支持的语言中都可正确显示。
- 可支持如下 HTML 函数集的子集：

HTML 标签

标签	说明	注释
a	锚点或链接	所支持的属性：href 和名称
address	地址	
big	大字体	
blockquote	首行缩进段落	
body	正文	所支持的属性：bgcolor (#RRGGBB)
br	换行	
center	居中段落	
cite	内嵌引用	与标签 i 的效果相同
code	代码	与标签 tt 的效果相同
dd	定义数据	
dfn	定义	与标签 i 的效果相同
div	文档分隔	支持标准程序段属性
dl	定义列表	支持标准程序段属性
dt	定义术语	支持标准程序段属性
em	强调	与标签 i 的效果相同
font	字体大小，语系，颜色	所支持的属性：大小，字符体和颜色 (#RRGGBB)

标签	说明	注释
h1	1 级标题	支持标准程序段属性
h2	2 级标题	支持标准程序段属性
h3	3 级标题	支持标准程序段属性
h4	4 级标题	支持标准程序段属性
h5	5 级标题	支持标准程序段属性
h6	6 级标题	支持标准程序段属性
head	文档页眉	
hr	水平线	所支持的属性：宽度（可以是绝对数值或者相对数值）
html	HTML 文档	
i	斜体	
img	图像	所支持的属性：src，宽度，高度
kbd	用户输入文本	
meta	元信息	
li	列表条目	
nobr	不易破碎文本	
ol	有序列表	支持列表的标准属性
p	段落	支持标准程序段属性（默认设置：左对齐）
pre	预先设好格式的文本	
s	删除线	
samp	示例代码	与标签 tt 的效果相同
small	小字体	
span	分组元素	
strong	加粗	连续文本加粗显示，替代标签 b
sub	下标	
sup	上标	
table	表格	所支持的属性：边框，bgcolor (#RRGGBB)，单元格间距，单元格填充，宽度（绝对或者相对），高度
tbody	表格正文	无效
td	表格数据单元	支持表格单元格的标准属性
tfoot	表格页脚	无效
th	表格页眉单元	支持表格单元格的标准属性

标签	说明	注释
thead	表格页眉	打印表格时使用，可扩展至多页
title	文档标题	
tr	表格行	所支持的属性：bgcolor (#RRGGBB)
tt	打字字体	
u	下划线	
ul	无序列表	支持列表的标准属性
var	变量	与标签 tt 的效果相同

程序段属性

标签 div、dl、dt、h1、h2、h3、h4、h5、h6 和 p 支持如下属性：

- 对齐（左对齐，右对齐，居中，垂直居中）
- dir (ltr, rtl)

列表的标准属性

标签 ol 和 ul 支持如下属性：

- 类型 （1, a, A, 方形，圆形，环形）

表格的标准属性

标签 td 和 th 支持如下属性：

- 宽度（绝对，相对，无数值）
- bgcolor (#RRGGBB)
- 列间距
- 行间距
- 对齐（左对齐，右对齐，居中，垂直居中）
- 对齐（上对齐，居中，下对齐）

CSS 属性

如下表格中为所支持的 CSS 函数集：

属性	值	说明
background-color	<color>	元素的背景颜色
background-image	<uri>	元素的背景图像
color	<color>	文本的颜色
text-indent	<length>px	段落首行缩进长度，单位：像素
white-space	normal | pre | nowrap | pre-wrap	确定如何处理 HTML 文档中的空格
margin-top	<length>px	边框上边距的宽度，单位：像素
margin-bottom	<length>px	边框下边距的宽度，单位：像素
margin-left	<length>px	边框左边距的宽度，单位：像素
margin-right	<length>px	边框右边距的宽度，单位：像素
vertical-align	baseline | sub | super | middle | top | bottom	文本垂直对齐，在表格中只支持数值“居中”、“上对齐”、“下对齐”。
border-color	<color>	文本表格的边框颜色
border-style	none | dotted | dashed | dot-dash | dot-dot-dash | solid | double | groove | ridge | inset | outset	文本表格的边框样式
background	[<'background-color'> || <'background-image'>]	背景属性的缩写
page-break-before	[auto | always]	在段落/表格前分页
page-break-after	[auto | always]	在段落/表格后分页
background-image	<uri>	元素的背景图像

所支持的 CSS 选择器

除所谓的虚拟选择器等级（如 :first-child， :visited 和 :hover）外，支持所有的 CSS 2.1 选择器等级。

17.3 创建帮助手册

帮助手册是一个 XML 文件，可在其中决定在线帮助的结构。 可在此文件中定义：

- HTML 文档
- 内容和关键词目录

帮助手册的句法

标签	数量	含义	
HMI_SL_HELP	1	XML 文档的根元素	
I-BOOK I I I I I I I I I I	+	帮助手册命名。手册名称可自由选择，只要不是手册中预定义的名称即可，例如：sinumerik_alarm_plc_pmc。 本例中帮助手册的名称为“hmi_myhelp”。 属性：	
		ref	命名 HTML 文档，其将作为帮助手册的主页显示。
		titel	显示在内容目录中的帮助手册的标题。
		helpdir	包含帮助手册的在线帮助的目录。
I-ENTRY II II II II II II II	*	在线帮助的章节 属性：	
		ref	命名 HTML 文档，其将作为章节的主页显示。
		titel	显示在内容目录中的章节标题。
II-INDEX_ENTRY II II II II II II II	*	待显示的关键词 属性：	
		ref	命名 HTML 文档，将根据关键词条目跳至此文档。
		titel	在关键词目录中显示的关键词的标题。

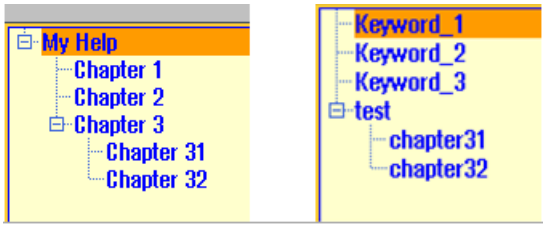
“数量”列：
* 表示 0 或者更多
+ 表示 1 或者更多

帮助手册示例

如下示例将描述名称为“My Help”的帮助手册的结构。 此外，其还是内容目录和关键词目录的基础。

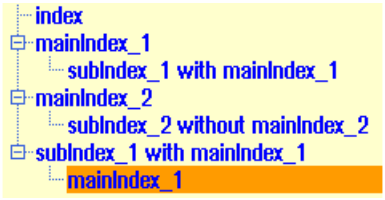
```
<?xml version="1.0" encoding="utf-8"?>
<HMI_SL_HELP language="en-US">
  <BOOK ref="index.html" title="My Help" helpdir="hmi_myhelp">
    <ENTRY ref="chapter_1.html" title="Chapter 1">
      <INDEX_ENTRY ref="chapter_1html#Keyword_1" title="Keyword_1"/>
      <INDEX_ENTRY ref="chapter_1.html#Keyword_2" title="Keyword_2"/>
    </ENTRY>
    <ENTRY ref="chapter_2.html" title="Chapter 2">
      <INDEX_ENTRY ref="chapter_2.html#Keyword_3" title="Keyword_3"/>
    </ENTRY>
    <ENTRY ref="chapter_3.html" title="Chapter 3">
      <ENTRY ref="chapter_31.html" title="Chapter 31">
        INDEX_ENTRY ref="chapter_31.html#test" title="test;chapter31"/>
      </ENTRY>
      <ENTRY ref="chapter_32.html" title="Chapter 32">
        INDEX_ENTRY ref="chapter_32.html#test" title="test;chapter32"/>
      </ENTRY>
    </ENTRY>
  </BOOK>
</HMI_SL_HELP>
```

整个手册由三章组成，其中第三章包括两个子章节。 在各章中均定义了不同的关键词。



可采用如下三种方法编排关键词目录的格式：

1. 单个条目：
   `<INDEX_ENTRY ...title="index"/>`
2. 两个二级条目，其中每个标题拥有一个主条目和子条目。 用逗号将不同的条目隔开：
   `<INDEX_ENTRY ...title="mainIndex_1,subIndex_1 with mainIndex_1"/>`
3. 二级条目，其中为主条目的第一个标题和子条目的第二个标题。 使用分号隔开各条目。
   `<INDEX_ENTRY ...title="mainIndex_2;subIndex_2 without mainIndex_1"/>`



17.4 将在线帮助集成至 SINUMERIK Operate 中

若需将创建的帮助手册集成至 SINUMERIK Operate 的在线帮助系统中，需使用文件“slhlp.xml”。

“slhlp.xml”文件的格式描述

标签	数量	含义	
CONFIGURATION	1	XML 文档的根元素。 用于标识配置文件。	
I-OnlineHelpFiles	1	引入在线帮助手册的段落。	
II-<help_book>	*	引入帮助手册的段落。	
III-EntriesFile III III III III III	1	含内容条目和关键词条目的帮助手册的文件名。 属性：	
		值	XML 文件的名称
		类型	数值的数据类型 (QString)

标签	数量	含义	
III-Technology III III III III III III III III III IIII	0, 1	规定帮助手册所适用的工艺。 “All”（全部）表示适用于全部工艺。 若一个帮助手册适用于多个工艺，则工艺之间用逗号隔开。 可能的数值： 全部，通用，铣床，车床，磨床，冲床 属性：	
		值	工艺说明
		类型	数值的数据类型 (QString)
III -DisableSearch III III III III III	0, 1	关闭帮助手册的关键词搜索。 属性：	
		值	true, false
		类型	数值的数据类型 (bool)
III- DisableFullTextSearch III III III III	0, 1	关闭帮助手册的全部文本搜索功能。 属性：	
		值	true, false
		类型	数值的数据类型 (bool)
III-DisableIndex III III III III	0, 1	关闭帮助手册的关键词目录。 属性：	
		值	true, false
		类型	数值的数据类型 (bool)
III-DisableContent III III III III	0, 1	关闭帮助手册的内容目录。 属性：	
		值	true, false
		类型	数值的数据类型 (bool)

标签	数量	含义	
III-DefaultLanguage III III III III III	0, 1	若帮助手册中有当前国家的语言，则会显示此语言的缩写。 属性：	
		值	chs, deu, eng, esp, fra, ita, ...
		类型	数值的数据类型 (QString)

“数量”列：
* 表示 0 或者更多

文件“slhlp.xml”示例

以下示例将讲述如何使 SINUMERIK Operate 识别帮助手册“hmi_myhelp.xml”。

没有激活此帮助手册的关键词目录。

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE CONFIGURATION>
<CONFIGURATION>
   <OnlineHelpFiles>
     <hmi_myHelp>
       <EntriesFile value="hmi_myhelp.xml" type="QString"/>
       <DisableIndex value="frue" type="bool"/>
     </hmi_myHelp>
   </OnlineHelpFiles>
</CONFIGURATION>
```

17.5 保存帮助文件

在目标系统中保存帮助文件

1. 打开目录 /**oem**/sinumerik/hmi/hlp 并在其中为所需的语种创建一个文件夹。请使用规定的语种标识。
   文件夹名称只能为小写。
   例如：若目标语言为德语和英语，则创建文件夹“deu”和“eng”。
2. 将各自的帮助手册（如“hmi_myhelp.xml”）分别保存至“deu”和“eng”文件夹中。
3. 将帮助文件复制至如下目录中，比如：德语为：/**oem**/sinumerik/hmi/hlp/deu/hmi_myhelp；英语为：/**oem**/sinumerik/hmi/hlp/eng/hmi_myhelp。

4. 将配置文件“slhlp.xml”保存至目录/**oem**/sinumerik/hmi/cfg 中。
5. 重新启动 HMI。

说明

在显示帮助手册的内容和关键词目录时，为了进行快速处理，可在目录/siemens/sinumerik/sys_cache/hmi/hlp 下将帮助文件保存为二进制格式 (slhlp_<Hilfe-Buch_*.hmi)。修改帮助手册时，必须删除此文件。

参见

支持的语言 (页 614)

17.6 创建针对用户报警和机床数据的在线帮助

您可以为用户 PLC 报警自定义在线帮助。出现报警时，可以从报警列表中打开针对此条报警的帮助。

用户专用 PLC 报警的帮助文本位于 HTML 文件 "sinumerik_alarm_oem_plc_pmc.html" 中。

创建 HTML 文件

1. 请在以下目录中创建 HTML 文件：/**oem**/sinumerik/hmi/hlp/<lng>/sinumerik_alarm_plc_pmc/ 或 /**user**/sinumerik/hmi/hlp/<lng>/sinumerik_alarm_plc_pmc/。
   <lng> 表示语言代码，例如 deu、eng 等。如果要创建语言目录，请使用章节“支持的语言 (页 614)”中表格内的语言代码。
2. 将此文件命名为“sinumerik_alarm_oem_plc_pmc.html”。若要为多个语种创建帮助文本，必须严格使用此名称相应地创建多个 HTML 文件。此文件名不能更改，适用于所有的语种。
3. 请将文件保存在相应的语言目录下。例如，将德语的帮助文本文件放入以下目录：/**oem**/sinumerik/hmi/hlp/deu/sinumerik_alarm_plc_pmc/sinumerik_alarm_oem_plc_pmc.html
4. 删除目录/**siemens**/sinumerik/sys_cache/hmi//hlp 中的文件“slhlp_sinumerik_alarm_plc_pmc_*.hmi”，然后重新启动操作软件。

文件“sinumerik_alarm_oem_plc_pmc.html”

条目	含义
AlarmNr	报警编号超级链接
.....	相应报警的帮助文本
<td width="85%">.....</td>	显示在“注释”或“消除方法”栏后面的文本。

示例

报警编号作为 HTML 锚点使用。

```
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE html PUBLIC "-//W3C//DTD
HTML 4.0 Transitional//EN" >
- <html>
   <head><title></title></head>
-  <body>
-    <table>
-      <tr>
-       <td width="15%">
          <b><a name="510000">510000</a></b>
        </td>
-       <td width="85%">
          <b>Dies ist die Hilfe für den Anwender-Alarm 510000</b>
        </td>
      </tr>
-     <tr>
-       <td valign="top" width="15%">
          <b>Erl&auml;uterung</b>
        </td>
        <td width="85%"> Dies ist die Erläuterung zum Anwender-Alarm
        510000.</td>
      </tr>
-     <tr>
        <td valign="top" width="15%"><b>Abhilfe:</b></td>
        <td width="85%">Alarm-Grund beheben. </td>
      </tr>
    </table>

  </body>
 </html>
```

将用户报警的在线帮助分散到多个 HTML 文件中

您可以将用户报警的在线帮助分散到多个 HTML 文件中。

1. 请在以下目录中创建一个名为"sinumerik_alarm_plc_pmc.xml"的 XML 文件：/**oem**/sinumerik/hmi/hlp/<lng>/ 或 /**user**/sinumerik/hmi/hlp/<lng>/
   <lng> 表示语言代码，例如 deu、eng 等。如果要创建语言目录，请使用章节 "支持的语言 (页 614)"中表格内的语言代码。
2. 打开这个 XML 文件，在标签 <NUM_AREAS> 中输入报警的编号范围和对应的 HTML 文件。
3. 在下面的目录中创建 XML 文件中指定的 HTML 文件：
   /**oem**/sinumerik/hmi/hlp/<lng>/sinumerik_alarm_plc_pmc 或
   /**user**/sinumerik/hmi/hlp/<lng>/sinumerik_alarm_plc_pmc

示例：创建 XML 文件

条目	含义
<BOOK>	帮助手册
<NUM_AREAS>	报警的编号范围和与相应 HTML 文件的参照关系

下面的在线帮助示例中：
报警 500000 ~ 599999 对应的是 HTML 文件"sinumerik_alarm_oem_1.html"、
报警 600000 ~ 699999 对应的是 HTML 文件"sinumerik_alarm_oem_2.html"、
报警 700000 ~ 799999 对应的是 HTML 文件"sinumerik_alarm_oem_3.html" 、
报警 800000 ~ 899999 对应的是 HTML 文件"sinumerik_alarm_oem_4.html"。

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
 <!DOCTYPE HMI_SL_HELP>
 <HMI_SL_HELP>
  <BOOK>
    <NUM_AREAS>
      <NUM_AREA from="500000" to="599999"
      ref="sinumerik_alarm_oem_1.html" />
      <NUM_AREA from="600000" to="699999"
      ref="sinumerik_alarm_oem_2.html" />
      <NUM_AREA from="700000" to="799999"
      ref="sinumerik_alarm_oem_3.html" />
      <NUM_AREA from="800000" to="899999"
      ref="sinumerik_alarm_oem_4.html" />
    </NUM_AREAS>
  </BOOK>
 </HMI_SL_HELP>
```

替换缺省的在线帮助

您可以为 HMI 报警、NC 报警、PLC 报警以及机床数据创建一个自定义的在线帮助，替换原有的缺省在线帮助。

自定义的在线帮助直接通过控制器上"INFO"按键即可调用。随后便显示自定义的在线帮助，而不是缺省在线帮助。

HTML 文件的目录

区域	目录
PLC 报警	/**oem**/sinumerik/hmi/hlp/<lng>/sinumerik_alarm_plc_pmc/ 或 /**user**/sinumerik/hmi/hlp/<lng>/sinumerik_alarm_plc_pmc/
HMI 报警	/**oem**/sinumerik/hmi/hlp/<lng>/sinumerik_alarm_hmi/ 或 /**user**/sinumerik/hmi/hlp/<lng>/sinumerik_alarm_hmi/
NC 报警	/**oem**/sinumerik/hmi/hlp/<lng>/sinumerik_alarm_nck/ 或 /**user**/sinumerik/hmi/hlp/<lng>/sinumerik_alarm_nck/
NC 机床数据	/**oem**/sinumerik/hmi/hlp/<lng>/sinumerik_md_nck/
通道机床数据	/**oem**/sinumerik/hmi/hlp/<lng>/sinumerik_md_chan/
轴机床数据	/**oem**/sinumerik/hmi/hlp/<lng>/sinumerik_md_axis/

<lng> 表示语言代码，例如 deu、eng 等。请只使用章节 "支持的语言 (页 614)"中表格内的语言代码。

步骤：

1. 建立对应的目录，参见表格"HTML 文件的目录"。
2. 创建帮助文件，文件名采用报警号或机床数据号，并在 HTML 帮助文件后加上后缀名".html"，PDF 帮助文件后加上后缀名".pdf"。若要为多个语种创建帮助文本，必须严格使用此名称相应地创建多个帮助文件，然后将这些文件放入对应的语种目录中。

示例：

- 针对 PLC 报警 510000 的自定义德语在线帮助：
  /oem/sinumerik/hmi/hlp/deu/sinumerik_alarm_plc_pmc/510000.html
  /oem/sinumerik/hmi/hlp/deu/sinumerik_alarm_plc_pmc/510000.pdf
- 针对 NC 机床数据 14510 $MN_USER_DATA_INT 的自定义德语在线帮助：
  /oem/sinumerik/hmi/hlp/deu/sinumerik_md_nck/14510.html

17.7 示例： 创建针对 NC /PLC 变量的在线帮助的步骤

概述

为能像下面示例一样，为 NC /PLC 变量或系统变量创建一个上下文敏感的在线帮助，须在和语言相关的 html 文件中管理描述文本。

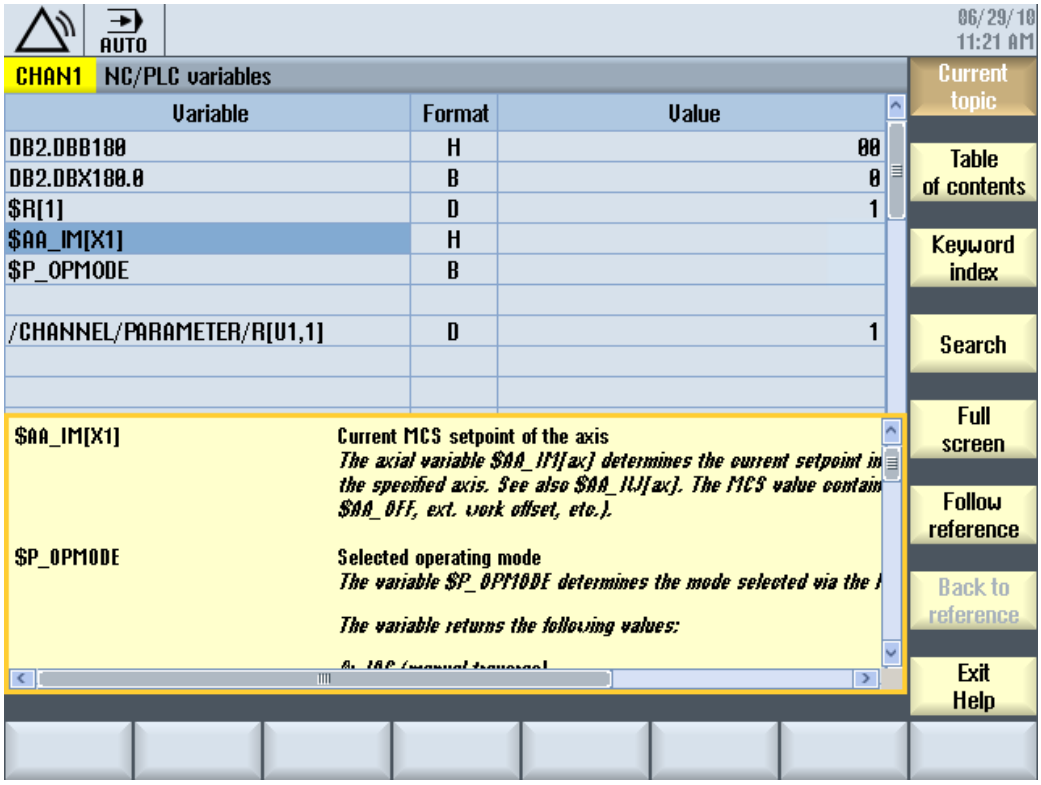


图 17-1 示例： 用户变量的在线帮助

在线帮助的建立

对于在线帮助，需要以下文件：

文件	含义
sldgvarviewhelp.ini	配置文件，用于管理 html 文件或多个 html 文件
<lng>/<name>1.html	在线帮助 html 文件的内容是与语言相关的，并且储存在各个语言目录 <lng>中。
<lng>/<name>2.html	
. . .	
<lng>/<name>n.html	

配置文件的结构

该文件**不分语言**，结构为：

sldgvarviewhelp.ini

```
[HelpBindings]
/BAG/STATE/OPMODE = var1_help.html#var1
$AA_IM[X1] = var1_help.html
$R[1] = var1_help.html#var2
/Channel/Parameter/R[u1,1] = var2_help.html#var2
DB2.DBX180.0 = var2_help.html#var1
GUD/MyVar[2] = var2_help.html
```

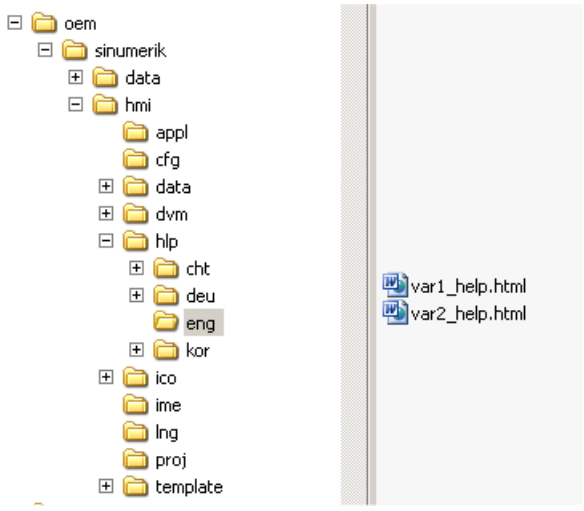
说明

html 文件可以通过任意的 html 编辑器来创建。 在配置文件中定义，哪些 html 文件属于在线帮助。

这个描述可以由一个或多个 html 文件构成： 例如，每个变量有一个 html 文件或多个同类型的变量在一个文件中。

操作步骤：

1. 将配置文件复制到以下目录中：
   /oem/sinumerik/hmi/cfg/sldgvarviewhelp.ini
2. 将 html 文件复制到以下任何一个目录中：
   /oem/sinumerik/hmi/hlp/<lng>/
   /user/sinumerik/hmi/hlp/<lng>/
   <lng> 代表语种标识。



3. 删除目录下的所有文件：
   /siemens/sinumerik/sys_cache/hmi//hlp

这些设置只有在系统重新启动后才会生效。

17.8 示例： 创建编程在线帮助的步骤

前提条件

创建以下文件：

- 配置文件：“prog_help.ini”

```
[milling]
CYCLE1=cycle1_help.html
CYCLE2=cycle2_help.html#TextAnchor1
CYCLE3=cycle3_help.html
CYCLE4=cycle4_help.html
```

```
[turning]
CYCLE3=cycle2_help.html
CYCLE4=cycle3_help.html
```

- 帮助手册"slhlp.xml"的配件文件（可选）

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE CONFIGURATION>
<CONFIGURATION>
  <OnlineHelpFiles>
    <hmi_prog_help>
      <EntriesFile value="hmi_prog_help.xml" type="QString"/>
      <DisableIndex value="true" type="bool"/>
    </hmi_prog_help>
  </OnlineHelpFiles>
</CONFIGURATION>
```

- 帮助手册"hmi_prog_help.xml"的配件文件（可选）

```
<?xml version="1.0" encoding="UTF-8"?>
<HMI_SL_HELP language="en-US">
  <BOOK ref="index.html" title="OEM_CYCLES" helpdir="hmi_prog_help">
    <ENTRY ref="cycle1_help.html" title="Cycle1"></ENTRY>
    <ENTRY ref="cycle2_help.html" title="Cycle2"></ENTRY>
    <ENTRY ref="cycle3_help.html" title="Cycle3"></ENTRY>
    <ENTRY ref="cycle4_help.html" title="Cycle4"></ENTRY>
    <ENTRY ref="cycle_help.html" title="OEM_Cycles"></ENTRY>
  </BOOK>
</HMI_SL_HELP>
```

- 与语言相关的文件"<prog_help_eng>.ts"：此文件名是固定的。

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!DOCTYPE TS>
<TS>
    <context>
      <name>oem_cycles</name>
        <message>
       <source>CYCLE1</source>
          <translation>short description for cycle 1</translation>
          <chars>*</chars>
        </message>
```

```
        <message>
       <source>CYCLE2</source>
          <translation>short description for cycle 2</translation>
          <chars>*</chars>
        </message>
        <message>
          <source>CYCLE3</source>
          <translation>short description for cycle 3</translation>
          <chars>*</chars>
        </message>
    </context>
</TS>
```

将帮助文件保存到目标系统中

操作步骤：

1. 将配置文件"prog.ini"复制到目录：
   /oem/sinumerik/hmi/cfg
2. 将帮助手册的文件“slhlp.xml”拷贝到以下目录中：
   /oem/sinumerik/hmi/cfg
3. 在路径下创建一个目录用于所需语言的在线帮助：/oem/sinumerik/hmi/hlp/<lng> 并把它拷贝到文件 hmi_prog_help.xml 中去
   对此，使用文件名语言识别表中规定的语言符号。目录的名称必须为小写。
4. 把用于简短描述的、和语言有关的文件 prog_help_<lng>.ts 拷贝到以下路径：/oem/sinumerik/hmi/<lng>/prog_help_<lng>.ts
5. 把对 OEM 循环描述的 html 文件拷贝到以下目录中：
   /oem/sinumerik/hmi/hlp/<lng>/hmi_prog_help/cycle<n>_help.html

这些设置只有在系统重新启动后才会生效。

17.9 PDF 格式的帮助文件

除了 HTML 格式的帮助文件以外，还可以在操作软件中插入 PDF 格式的信息。通过目录或索引中的链接可以打开单个 PDF 帮助文件，或直接打开 HTML 文件。

PDF 帮助文件存档

将 PDF 帮助文件复制到以下任何一个目录中：

/oem/sinumerik/hmi/hlp/<lng>/<hmi_myhelp>

/user/sinumerik/hmi/hlp/<lng>/<hmi_myhelp>

插入 PDF 帮助文件

在对话项目或目录和索引项目中插入后缀为"PDF"的文件，插入方法和插入"HTML"文件一样：

```
<ENTRY ref="myFile.pdf" title="Help 1">
```

从 HTML 文件链接到 PDF 帮助文件：

```
<a href="myFile.pdf">My Help File</a>
```

说明

在 PDF 帮助文件中无法选择上下文相关的跳转标签或跳转到其他 HTML 或 PDF 文件。
搜索功能仅适用一个 PDF 文件内。不支持在多个 PDF 帮助文件中进行高级搜索。

17.10 PDF 文件的语言支持

SINUMERIK Operate 支持在默认语言切换时对已打开的 PDF 文件进行显示切换。如果运行中所选择的 PDF 文件语言存在，则在每次切换操作语言时会自动加载相应语言版本的 PDF 文件。

说明

如果相应语言版本的 PDF 文件不存在，则会打开作为默认语言版本的英文版 PDF 文件。如果此时默认语言版本不存在或损坏，则上一次打开的 PDF 文件仍然生效。

有关默认语言和语言识别的更多信息，参见章节 "支持的语言 (页 614)"。

可通过以下方式保存所支持语言的 PDF 文件：

总目录下带语言标识的 PDF 文件

将带有语言标识的 PDF 文件复制到一个目录下，例如：

```
/card/<home>/<myFolder>/<abc>_eng.pdf
/card/<home>/<myFolder>/<abc>_deu.pdf
/card/<home>/<myFolder>/<abc>_fra.pdf
...
```

语言目录下的 PDF 文件

将 PDF 文件复制到一个语言目录下，例如：

```
/card/<home>/<myFolder>/eng/<abc>.pdf
/card/<home>/<myFolder>/deu/<abc>.pdf
/card/<home>/<myFolder>/fra/<abc>.pdf
...
```

自定义语言目录下的 PDF 帮助文件

将 PDF 帮助文件复制到以下其中一个目录中，例如：
/**oem**/sinumerik/hmi/hlp/**eng**/<abc>.pdf
/**oem**/sinumerik/hmi/hlp/**deu**/<abc>.pdf

或
/**user**/sinumerik/hmi/hlp/**eng**/<abc>.pdf
/**user**/sinumerik/hmi/hlp/**deu**/<abc>.pdf
...

SINUMERIK Integrate - AMB、AMC、AMM 18

前提条件

您需要通过显示机床数据设计一个软键，来启动“SINUMERIK Integrate - AMB、AMC、AMM”。软键的设计至少需要访问级别 3 （用户）。

设计软键

MD9108 $MM_SINUMERIK_INTEGRATE	
= 1	软键“SINUMERIK Integrate”显示在扩展的操作区域栏中。

更多信息

软件信息请见

SINUMERIK Integrate - AMB、AMC、AMM 功能手册

工艺和循环 19

19.1 激活车削/铣削/钻削/磨削工艺

在控制系统启动（上电）时会自动载入循环。

循环是由循环机床数据和循环设定数据配置的，具体而言，这些数据有：

- 通用循环机床数据
- 通用专用循环机床数据
- 轴专用循环机床数据
- 通用循环设定数据
- 通道专用循环设定数据
- 轴专用循环设定数据

存储器配置前提条件

为确保循环正确运行，需要检测以下缺省设置：

MD28082 $MC_MM_SYSTEM_FRAME_MASK	系统框架（SRAM）
位 5	循环系统框架
= 1	缺省设置

MD52201 $MCS_TECHNOLOGY_EXTENSION	带有多种工艺的联合机床上的工艺扩展
= 1	扩展的车削工艺的设置
= 2	扩展的铣削工艺的设置， 例如带铣削工艺的车床 MD52200 $MCS_TECHNOLOGY = 1 MD52201 $MCS_TECHNOLOGY_EXTENSION = 2

说明

SINUMERIK Operate 不支持外圆磨削和平面磨削作为工艺扩展。

调整操作界面

借助下列通道专用的配置机床数据您可以设置软键显示，该软键用于在程序编辑器中选择循环支持:

MD52200 $MCS_TECHNOLOGY	工艺
= 1	车削工艺的设置
= 2	铣削工艺的设置
= 3	外圆磨削工艺的设置
= 4	平面磨削工艺的设置

更多信息

关于磨削工艺的信息参见下列文档：

- 章节“磨削 (页 468)”
- 操作手册，磨削版
- NC 编程手册
- 功能手册之监控与补偿

配置屏幕显示

可通过以下通道专用的机床数据配置屏幕显示：

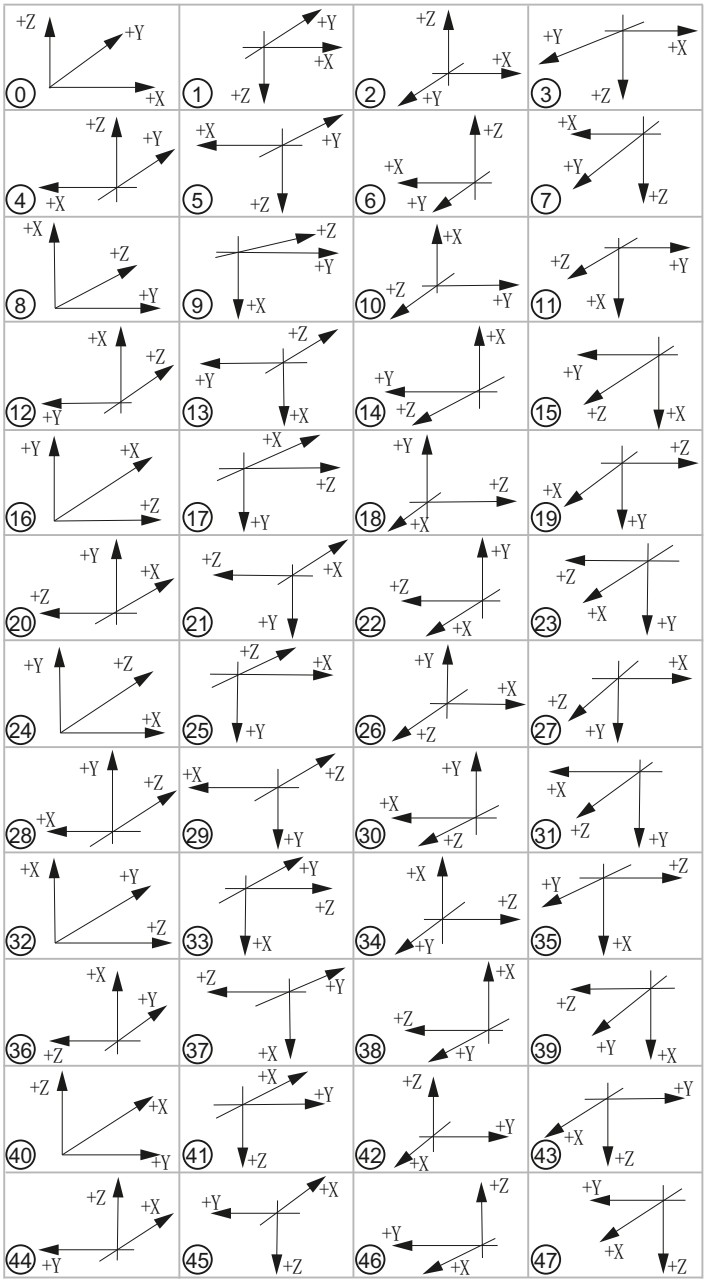
MD52210 $MCS_FUNCTION_MASK_DISP	显示功能屏幕
位 0	用于程序的单位制始终在基准系统中
位 1	示教坐标系中车削时的端面视图
位 2	隐藏 JOG 范围内的软键“T、S、M”
位 3	在 MDA 中自动生成程序尾（通过软键“删除程序段”）
位 4	在 T、F、S 窗口中显示下一把刀具
位 5	隐藏软键“MCS 实际值”
位 6	隐藏 T、F、S 窗口中的刀具半径/直径
位 7	隐藏 T、F、S 窗口中的刀具长度
位 8	隐藏 T、F、S 窗口中的刀具图标
位 9	显示“简单”输入选择，参见章节“在循环支持中的“简单”输入”
位 10	任务列表中未提供通道

位 11	WCS 上激活逻辑主轴显示
位 12	隐藏 G 代码模具制造视图

设置坐标系

MD52000 $MCS_DISP_COORDINATE_SYSTEM	坐标系的位置
= 0	缺省设置
使用该机床数据可进行操作界面坐标系位置与机床坐标系的匹配。在操作界面上会自动根据所选位置更新所有的辅助图、过程图、模拟和与旋转方向有关的输入栏。	

坐标系有以下设置选项：



轴的含义

MD52206 $MCS_AXIS_USAGE[n]	通道中轴的含义 [n]为通道轴号
= 0	没有特定含义
= 1	刀具主轴（动力刀具）

= 2	辅助主轴（动力刀具）
= 3	主主轴（车削）
= 4	主主轴的独立 C 轴（车削）
= 5	副主轴（车削）
= 6	副主轴的独立 C 轴（车削）
= 7	副主轴的直线轴（车削）
= 8	尾座（车削）
= 9	中心架（车削）
= 10	B 轴（车削）。
= 12	副主轴上的 B 轴（车削）
= 13	副主轴的横轴 X 轴（车削）

确定旋转方向

对于没有在 ToolCarrier 或 5 轴转换中设计的回转轴，通过以下通道专用机床数据输入其旋转方向。

MD52207 $MCS_USAGE_ATTRIB[n]	通道中轴的属性 [n]为通道轴号
位 0	围绕第 1 几何轴旋转（旋转轴）
位 1	围绕第 2 几何轴旋转（旋转轴）
位 2	围绕第 3 几何轴旋转（旋转轴）
位 3	旋转方向为逆时针方向（适用于旋转轴/ C 轴）
位 4	显示的主轴/ C 轴旋转方向 M3 为逆转
位 5	M3/M4 逆转（在主轴时） 该位必须按 PLC 位 DB31, ... DBX17.6 进行设置！
位 6	显示旋转轴作为测量的补偿目标
位 7	提供位置模式中的旋转轴
位 8	提供用于毛坯接收的旋转轴（铣床上）
位 9	主轴没有定位能力
位 10	回转轴围绕第 1 几何轴旋转（仅用于位置模式）
位 11	回转轴围绕第 2 几何轴旋转（仅用于位置模式）
位 12	回转轴围绕第 3 几何轴旋转（仅用于位置模式）

循环支持中的“简单”输入

MD52210 $MCS_FUNCTION_MASK_DISP 位 9 的设置会影响可见参数的数量：

- 缺省设置位 9 = 0 （“完全”）时，显示所有输入参数。
- 位 9 = 1 时，可在循环支持的输入参数“完全”或“简单”之间进行选择。

如果选择了“简单”输入屏幕来输入参数，系统会将以下设定数据用于不可见输入参数（机床专用适配的）：

SD55300 $SCS_EASY_SAFETY_CLEARANCE	简单输入：安全距离
= 1	mm

SD55301 $SCS_EASY_DWELL_TIME	简单输入：暂停时间
= 0.6	s

SD55305 $SCS_EASY_DRILL_DEEP_FD1	简单输入：深孔钻削第 1 进给率的百分比值
= 90	%

SD55306 $SCS_EASY_DRILL_DEEP_DF	简单输入：深孔钻削进刀的百分比值
= 90	%

SD55307 $SCS_EASY_DRILL_DEEP_V1	简单输入：深孔钻削的最小切深
= 1.2	mm

SD55308 $SCS_EASY_DRILL_DEEP_V2	简单输入：深孔钻削的回退量
= 1.4	mm

SD55309 $SCS_EASY_THREAD_RETURN_DIST	简单输入：螺纹车削的回退距离
= 2	mm

说明

指定的标准值适用于公制系统机床，在英制机床上自动由 NC 进行换算。

其他设置

MD52005 $MCS_DISP_PLANE_MILL	平面选择 G17, G18, G19
= 0	铣削：编程时，在“programGUIDE G-Code”下进行循环辅助中的平面选择
= 17	G17 平面（缺省值）
= 18	G18 平面
= 19	G19 平面

MD52006 $MCS_DISP_PLANE_TURN	平面选择 G17, G18, G19
= 18	G18 平面（缺省值）

MD52212 $MCS_FUNCTION_MASK_TECH	工艺通用的功能屏幕
位 0	使能回转
= 0	不使能平面回转、刀具回转
= 1	使能平面回转、刀具回转
位 1	没有沿软件限位开关的优化运行
= 0	没有沿软件限位开关的优化运行
= 1	沿软件限位开关的优化运行

MD52240 $MCS_NAME_TOOL_CHANGE_PROG	用于 G 代码步骤的换刀程序
= 程序名	为换刀调用相应的程序。

参见

用于回转的工艺循环 (页 471)

19.2 用于钻削的工艺循环

钻削工艺

使用下列通道专用的配置机床数据和循环设定数据，您可以对钻削工艺进行设置。

MD52216 $MCS_FUNCTION_MASK_DRILL	钻削功能屏幕
位 0	攻丝 CYCLE84，工艺输入栏
= 0	隐藏输入栏
= 1	显示输入栏
位 1	攻丝 CYCLE840，工艺输入栏
= 0	隐藏输入栏
= 1	显示输入栏

SD55216 $SCS_FUNCTION_MASK_DRILL_SET	钻削功能屏幕
位 1	镗孔 CYCLE86：主轴定位时考虑刀具平面旋转 注： 必须根据 DIN 设置主轴旋转方向 M3/ M4 和回转轴旋转方向。 SPOS=0 时，显示平面的第 1 轴 + 方向的刀沿（G17 中为 X+）。
= 0	主轴定位（SPOS）时不考虑刀具平面旋转。
= 1	主轴定位（SPOS）时考虑刀具平面旋转。
位 2	镗孔 CYCLE86：进行主轴定位时要考虑回转台系统（Toolcarrier）
= 0	进行主轴定位时，不考虑回转台运动系统的分量（通过 CYCLE800，Toolcarrier 回转）。 注： 仅考虑旋转工件的机床运动系统的回转轴。 在运动系统的初始位置中，此回转轴必须平行于刀具轴定向。 例如绕 Z 轴（G17）旋转的回转轴 C，回转数据组的回转轴矢量 V2xyz = 0 , 0 , -1。
= 1	进行主轴定位时，考虑回转台运动系统的分量（通过 CYCLE800，Toolcarrier 回转）。
位 3	攻丝 CYCLE84：主轴机床数据 MD31050 $MA_DRIVE_AX_RATIO_DENOM 和 MD31060 $MA_DRIVE_AX_RATIO_NUMERA 的监控
位 4	攻丝 CYCLE84， 设置主轴机床数据 MD31050 $MA_DRIVE_AX_RATIO_DENOM 和 MD31060 $MA_DRIVE_AX_RATIO_NUMERA 的监控

= 0	无监控
= 1	监控
位 6	镗孔 CYCLE86：调整主轴位置，使之与刀具定向一致 可在通过 TOOLCARR 或 TRAORI 调用循环之前进行刀具定向。
= 1	位 1 和 位 2 不再生效。
位 7	镗孔 CYCLE86：根据激活的镜像指令调整平面上的退刀方向
= 0	平面上的退刀按照编程进行。
= 1	镜像指令激活时，平面上的退刀方向在循环内部自动根据该指令调整。

攻丝 （CYCLE84 和 CYCLE840）

当通过通道专用配置机床数据 MD52216 $MCS_FUNCTION_MASK_DRILL 隐藏工艺屏幕时，下列通道专用循环设定数据中的设置生效：

SD55481 $SCS_DRILL_TAPPING_SET_GG12[0]	设置攻丝 G 代码组 12： 精准停时程序段切换的属性
= 0	准停特性，与循环调用之前相同（缺省值）。
= 1	G601
= 2	G602
= 3	G603

SD55482 $SCS_DRILL_TAPPING_SET_GG21[0]	设置攻丝 G 代码组 21: 加速属性
= 0	加速特性，与循环调用之前相同（缺省值）。
= 1	SOFT
= 2	BRISK
= 3	DRIVE

SD55483 $SCS_DRILL_TAPPING_SET_GG24[0]	设置攻丝 G 代码组 24: 前馈控制
= 0	前馈控制，与循环调用之前相同（缺省值）。
= 1	FFWON
= 2	FFWOF

攻丝（CYCLE84）

SD55484 $SCS_DRILL_TAPPING_SET_MC[0]	MCALL 时的主轴运行
= 0	MCALL 时重新激活主轴运行（缺省值）。
= 1	MCALL 时保持位置控制的主轴运行。

19.3 制造商循环

19.3.1 制造商循环

说明

保护工艺知识

如需保护专有技术，可以为循环自定义访问权限，并设置文件加密。通过以下方法可实现循环保护：

- 使用另一个西门子应用程序"SINUCOM Protector"为循环数据加密。
  关于"SINUCOM Protector"的**更多说明**请访问互联网 (https://support.industry.siemens.com/cs/document/109474775/sinumerik-828d-840d-sl%3A-cycle-protection-with-protector-for-sinumerik-(oem-system-integrator)?dti=0&lc=en-WW)。
- 为循环数据设置访问权限，并针对用户调整该权限。
  关于自定义访问权限的**更多说明**请参见 "程序的访问等级 (页 66)"一章。

在循环软件包中，我们还为您提供了如下自定义循环：

CUST_TECHCYC.SPF	用来调整工艺循环的制造商循环。
CUST_782.SPF	测定装料适配的制造商循环(CYCLE782)，参见章节 "制造商循环 CUST_782.SPF (页 556)"。
CUST_800.SPF	用来调整平面回转与刀具回转功能（CYCLE800）的制造商循环，参见章节 "Herstellerzyklus CUST_800.SPF (页 527)"。
CUST_832.SPF	用来调整高速设定功能（CYCLE832）的制造商循环，参见章节 "制造商循环 CUST_832.SPF (页 562)"。
CUST_MEACYC.SPF	用来调整测量功能的制造商循环，参见章节 "制造商与用户循环 CUST_MEACYC.SPF (页 569)"。
CUST_MEAPROT.SPF	用于调整记录格式和内容的制造商和用户循环参见章节 "制造商与用户循环 CUST_MEAPROT.SPF (页 605)"。

PROG_EVENT.SPF	标准循环支持以下功能： • 在铣削或车削被激活时进行程序段搜索 • 程序段搜索和平面回转 • 程序段搜索和定位/调整刀具 注意 MD52212 $MCS_FUNCTION_MASK_TECH 位 3, 5 和 15 – 17。
CUST_T	该循环用于在 SERUPRO 后跟踪 T 就绪状态。
CUST_M6	该循环用于 SERUPRO 后的换刀状态。
CUST_MULTICHAN	用于多通道车床的制造商循环
CUST_CLAMP	夹具的制造商循环

复制制造商循环

执行修改前，将循环从
目录 NC 数据/循环/标准循环中复制到目录 NC 数据/循环/
用户循环中。只有在该目录下循环修改才
生效。



1. 选择操作区域“调试”。



2. 按下软键“系统数据”。

3. 打开目录“NC 数据/ 循环/标准循环”，选定制造商循环。



4. 按下软键“复制”，打开目录“NC 数据/ 循环/制造商循环”。



5. 按下软键“粘贴”。
   该循环可供个性化匹配。

说明

软件升级时要注意升级说明。

19.3.2 模拟运行的前提条件

配置

如需获得有关机床状态的反馈，例如在 CYCPE1MA、换刀循环、测量循环或 CUST_800 中，则必须在 NC 程序中关联以下系统变量，以避免 PLC 存取。 否则，PLC 存取会阻止数据比较、导致报警或中断模拟：

系统变量	含义
$P_SIM	仿真 NCK 生效

示例

在以下示例中，PROG_EVENT 或 CYCPE1MA.SPF 中的模拟被 PLC 存取中断：

```
N100 REPEAT
N110 UNTIL PLC_VAR <> 0
```

当使用系统变量$P_SIM 时，模拟运行中的 PLC 存取被阻止：

```
N099 IF NOT $P_SIM
N100 REPEAT
N110 UNTIL PLC_VAR <> 0
N111 ENDIF
```

19.3.3 扩展标准循环 PROG_EVENT

如果想要扩展 PROG_EVENT 的功能，一定不能将其复制到制造商循环目录，而要创建两个新的制造商循环。 为循环选用名称 CYCPE1MA.SPF 或 CYCPE_MA.SPF。

对于制造商循环 CYCPE1MA.SPF 或 CYCPE_MA.SPF ，在开始以及在

PROG_EVENT.SPF 结束时要加入相应的跳转标记。

如果制造商循环 CYCPE1MA.SPF 或 CYCPE_MA.SPF 建立在 NC 中，则 PROG_EVENT 会相应地转入这些制造商循环。

步骤



1. 选择操作区域“调试”。



2. 按下软键“系统数据”。
3. 选定目录“NC 数据/ 循环/制造商循环”



4. 按下软键“新建”。
   窗口“新建 G 代码程序”随即打开。
5. 将它命名为 CYCPE1MA 或 CYCPE_MA。
   只能使用该名称。



6 按下“确认”软键。
   循环已创建，可以对其进行调整

19.3.4 用于换刀的制造商循环 CUST_T 和 CUST_M6

功能

两个循环用于跟踪刀具准备（CUST_T）和 SERUPRO 后的换刀（CUST_M6）。此时针对特定刀套输出以下信息：

- 最后编程的换刀
- 最后编程的准备

前提条件

仅在刀具管理生效时才可使用这些循环。

步骤

两个循环在“SERUPRO-ENDE-ASUP”中调用，其还提供两个传输参数。

参数	含义
_THnr	已经为换刀或准备而设置过的刀套号或主主轴号。
_MTHnr	编程时生效的刀套或主主轴的编号。

循环自身不包含“机械逻辑”。只进行单纯的数据运算。

生效的刀套或主主轴会被保存。

通过传输参数测定 T 编程或 M6 编程时的情况。此情况决定，输出 T 和/或 M06，或者进入制造商循环。在输出 T 和 M06 后，或者从制造商循环返回后，会重新激活最初保存的刀套。

当存在相关询问（例如程序测试、程序段搜索等）时，可不受限制地调用换刀循环。

换刀循环

使用换刀循环时，必须注意以下提示：

说明

循环 CUST_T 使用语言指令 TCA（ToolChangeAbsolut）调用刀具，因此可换入已被禁用的刀具。但是不可替换此语言指令。

在机床上使用换刀操作时，必须注意以下事项：

- 必须明确地在规定位置上调用刀具准备循环。
- 如果通过循环 CUST_T 进行调用，则会跳过参数分析（$C_TS_PROG, ...）。
  或者可重新配置原始的语言指令 TCA。

更多信息

关于“SERUPRO-ENDE-ASUP”的更多信息，请参见

功能手册之基本功能。

19.3.5 制造商循环 CUST_TECHCYC.SPF

功能

当机床制造商需要根据具体机床进行调整时（例如激活特定的 M 指令），可从工艺循环以及 ShopMill 循环和 Shop Turn 循环中调用 CUST_TECHCYC 循环。该循环比如可用于，在车床上进行切断加工时，驶入/驶出接料箱；对主轴进行定位或夹紧（仅限加工车间）。

因此在编程或执行必要的进程时可以使用 CUST_TECHCYC.SPF 循环。

调整 OEM 循环“CUST_TECHCYC.SPF”

利用参数_MODE 可以执行以下操作：

标记	操作	
_M1	主主轴：	切换到 C 轴模式
_M2		切换到主轴模式
_M3		夹紧 C 轴
_M4		松开 C 轴 阻尼制动关闭
_M5		冲洗卡盘
_M6		闭合卡盘
_M7		主轴静止时打开卡盘
_M8		主轴旋转时打开卡盘
_M9		关闭冲洗卡盘
_M11	刀具主轴：	切换到 C 轴模式
_M12		切换到主轴模式
_M13		夹紧 C 轴
_M14		松开 C 轴
_M21	副主轴：	切换到 C 轴模式
_M22		切换到主轴模式
_M23		夹紧 C 轴
_M24		松开 C 轴 阻尼制动关闭
_M25		冲洗卡盘
_M26		闭合卡盘
_M27		主轴静止时打开卡盘
_M28		主轴旋转时打开卡盘
_M29		关闭冲洗卡盘
_M30	在程序段查找结束后进行第 4 根轴的定位（比如：副主轴、尾座和中心架）	

标记	操作	
_M41	拖动动力刀具	
_M42	脱开动力刀具	
_M61	切换到旋转加工	
_M62	切换到 C 柱面加工	
_M63	切换到 C 端面加工	
_M64	切换到 Y 柱面加工	
_M65	切换到 Y 端面加工	
_M66	切换到 B 端面加工（回转）	
_M67	撤销 B 端面加工（回转）	
_M68	在程序段查找结束后跳转到回转前的 ST 程序段	
_M80	插补车削：	初始化
_M81		在选择坐标转换前
_M82		在选择坐标转换后
_M85		取消
_M100	接料箱：	在切断工件前就位
_M101		在切断工件时移出
_M102		在切断工件后归位
_M103	结束切断工件	
_M110	换刀	
_M111	在换刀周期结束后（有换刀）	
_M112	在换刀前回到返回平面	
_M113	在重新选择刀沿前	
_M114	在换刀周期结束后（无换刀）	
_M115	在预选刀具后（ShopTurn）	
_M116	在换刀点上（ShopTurn）	
_M117	换刀 T 指令前	
_M118	在换刀后输出 M 功能和 H 功能（ShopMill、ShopTurn）	

标记	操作	
_M120	定义:主主轴与副主轴的耦合	
_M121	在 COUPOF 后关闭位置控制	
_M122	定义:主主轴/副主轴耦合	
_M123	在 COUPOF 后关闭位置控制	
_M125	切换到主主轴	
_M126	切换到副主轴	
_M131	程序头（ShopTurn 程序）	
_M132	程序头结束（ShopTurn 程序）	
_M135	程序循环末尾（ShopTurn 程序）	
_M136	程序尾（ShopTurn 程序）	
_M140	程序段查找	换刀前
_M141		换刀后
_M142		换刀后（ShopTurn 的新刀具）
_M150	2 通道切削：	主通道在“Balance Cutting”前
_M151		主通道在“Balance Cutting”后
_M152		副通道在“Balance Cutting”前
_M153		副通道在“Balance Cutting”后
_M154		在主通道中开始加工
_M155		在主通道中结束加工
_M156		在副通道中开始加工
_M157		在副通道中开始加工
_M160	多通道编程开始	
_M170	带主轴的开始程序块	
_M171	带主轴的结束程序块	
_M231	程序头（ShopMill 程序）	

标记	操作	
_M232	程序头结束（ShopMill 程序）	
_M235	程序循环末尾（ShopMill 程序）	
_M236	程序尾（ShopMill 程序）	
_M300	镗孔 CYCLE86 根据刀具位置调整主轴（SD55216 $SCS_FUNCTION_MASK_DRILL_SET 位 6 = 1） 某些机床坐标转换可能会要求对补偿角度进行附加调整，用于主轴定位（例如：主轴上的附加齿轮箱）。在计算补偿角度后使用标记 _M300。必要时可修改补偿角度 _MEA_CORR_ANGLE[0]（用于制造上）和 _MEA_CORR_ANGLE[1]（系统）。在 CYCLE86 的后续主轴定位中，两个角度都作为“增量值”生效。	
_M403	主主轴：	阻尼制动启用
_M423	副主轴：	阻尼制动启用
_M900	在输出毛坯前（仅限 ShopMill + ShopTurn）	

19.3.6 用户循环 CUST_MULTICHAN

功能

循环 CUST_MULTICHAN 用于车床上的多通道编程。 在含程序块的编程中，总是在程序块开始处调用此循环。

在循环中例如会将主主轴设置为程序块开始处设定的主轴。 通过返回值（= 1）可设置跳过整个程序块。

CUST_MULTICHAN

参数	含义
_S_NR	主轴号为为主主轴设置的编号。
_RET	返回值
= 0	根据试运行模式执行程序块。
= 1	跳过整个程序块。

19.4 铣削

19.4.1 常规

标准循环的机床数据和设定数据的所有设置均适用于 ShopMill 铣削和 ShopTurn 车削。

19.4.2 用于铣削的工艺循环

铣削功能标记

SD52214 $SCS_Function_MASK_MILL	铣削功能标记
位 0	使能圆柱体表面转换（ShopMill）
位 1	提供用于夹装毛坯的工作台（铣床上） （只有机床上存在用于夹装毛坯的旋转台和固定工作台时，才允许置位该位。）
位 2	预留
位 3	使能内外加工
位 4	使能主轴夹紧（C 轴）
位 5	使能界面上刀具主轴的控制
位 6	使能界面上旋转主轴的控制

SD55214 $SCS_FUNCTION_MASK_MILL_SET	铣削功能屏幕
位 0	同向运行铣削的基本设置。
位 2	有和没有安全距离的铣削循环深度计算。
= 0	在参考平面 + 安全距离与深度之间进行铣削循环的深度计算。
= 1	进行深度计算时不包括安全距离 位 2 在下列铣削循环中有效：CYCLE61，CYCLE71，CYCLE76，CYCLE77，CYCLE79，CYCLE899，LONGHOLE，SLOT1，SLOT2，POCKET3，POCKET4。

轮廓铣削（CYCLE63、CYCLE64）

SD55460 $SCS_MILL_CONT_INITIAL_RAD_FIN	精加工逼近圆半径 精加工轮廓腔时控制逼近圆的半径。
= 0	选择半径，使得起始点与精加工余量保持安全距离（缺省值）。
> 0	选择半径，使得起始点与精加工余量保持由该通道专用设定数据所确定的值。

SD55212 $SCS_FUNCTION_MASK_TECH_SET	工艺通用的功能屏幕
= 6	
位 3	删除由轮廓循环（CYCLE63、CYCLE64、CYCLE952）生成的程序
= 0	生成的程序未删除（兼容性，和目前一致）
= 1	编辑完调用循环后立即再次删除生成的程序

雕刻（CYCLE60）

SD55400 $SCS_MILL_ENGRAVE_POINT_RAD	雕刻循环 CYCLE60：创建符号“点”的圆弧轨迹半径
= 0	
= 0	通过简单地刀具线性插入运动形成符号“点”（常规雕刻刀具的缺省设置，兼容性）。
> 0	使用该设置日期中的半径值作为圆弧执行符号“点”操作（特定特殊刀具的设置）。

多边形 （CYCLE79），圆位置模式（HOLES2），圆弧槽（SLOT2）

SD55230 $SCS_CIRCLE_RAPID_FEED	以毫米/分钟为单位的快速进给率，用于圆弧槽之间或轮廓元素之间圆弧轨迹上的定位。
= 10000	

19.4.3 设置用于铣削的 ShopMill 循环

读入用于铣削的 ShopMill 循环

控制系统启动时，ShopMill 循环会自动读入以下目录：../siemens/sinumerik/cycles/sm/prog

设置坐标系

MD52000 $MCS_DISP_COORDINATE_SYSTEM	坐标系的位置
= 0	

示例：

=0 用于立式铣床

=16 用于卧式铣床，卧式镗床

MD52000 的详细说明请参见章节：

激活车削/铣削/钻削/磨削工艺 (页 407)

请根据以下机床数据设置铣削用的平面：

MD52005 $MCS_DISP_PLANE_MILL	铣削平面选择
= 17	
= 0	在循环支持中
= 17	G17 （缺省值）
= 18	G18
= 19	G19

模拟和同步记录

为了正确显示模拟和同步记录，可以按照下面章节中的说明设置机床数据：

设置用于模拟的坐标系 (页 179)

确定旋转方向

为了能够在 ShopMill 界面中正确显示旋转方向并且在编程 ShopMill 功能时执行正确的旋转方向，必须进行一些彼此协调相关的设置。此处的设置必须与机床上轴的实际旋转方向一致。

MD52207 $MCS_USAGE_ATTRIB[n]	通道中轴的属性 [n]为通道轴号
位 4	显示的旋转方向 M3 为逆时针旋转（主轴）
位 5	M3 相当于回转轴负向旋转（主轴） 必须按照 PLC 位 DB31, ... DBX17.6 设置该位！

定义其他用于位置模式的旋转方向

可为刀具轴定义位置模式。例如为了对应的回转位置处理回转轴 C 上的位置模式，必须在 MD52207 中激活位 12，而非位 10。

MD52207 $MCS_USAGE_ATTRIB[n]	通道中轴的属性 [n] 为通道轴号
位 2	围绕第 3 几何轴旋转（针对回转轴）
位 4	显示的旋转方向 M3 为逆时针旋转（主轴） 可以选择视向。或是自内腔出发，或是自外部至主轴。 但必须为所有主轴作出相同的选择。
位 5	旋转方向 M3 为回转轴负向（主轴）
位 6	显示回转轴作为测量的补偿目标
位 7	提供位置模式中的回转轴 通过位 0 至 2 定义轴的定向。 若位 10 至 12 中的一个置位，则这些位定义用于位置模式的轴定向。
位 8	提供用于毛坯夹具的回转轴（铣床上） 通过位 0 至 2 定义轴的定向。
位 10	回转轴围绕第 1 几何轴旋转（仅用于位置模式）

其他通道专用的机床数据

MD52216 $MCS_FUNCTION_MASK_DRILL	钻削功能屏幕
位 2	攻丝，带/无编码器 设置取决于机床上是否存在用于刀具主轴的编码器。
= 0	只用于带编码器的攻丝
= 1	攻丝，带/无编码器
位 3	基于中心孔直径的恒定切削速度
= 0	基于刀具直径的恒定切削速度
= 1	基于中心孔直径的恒定切削速度

MD52229 $MCS_ENABLE_QUICK_M_CODES	使能 M 指令
= 0	
位 0	冷却液关闭 （缺省值）

位 1	冷却液 1 开
位 2	冷却液 2 开
位 3	冷却液 1 和 2 开

MD52230 $MCS_M_CODE_ALL_COOLANTS_OFF	用于全部冷却液关闭的 M 代码
= 9	确定用于关闭冷却液的 M 功能，在换刀时输出。

MD52231 $MCS_M_CODE_COOLANT_1_ON	用于冷却液 1 打开的 M 代码
= 8	确定用于冷却液 1 的 M 功能，在换刀时输出。

MD52232 $MCS_M_CODE_COOLANT_2_ON	用于冷却液 2 打开的 M 代码
= 7	确定用于冷却液 2 的 M 功能，在换刀时输出。

MD52233 $MCS_M_CODE_COOLANT_1_AND_2_ON	用于两个冷却液打开的 M 代码
= -1	确定用于冷却液 1 和 2 的 M 功能，在换刀时输出。

MD52281 $MCS_TOOL_MCODE_FUNC_ON[ ]	刀具专用功能打开的 M 代码
= -1	不输出 M 功能。 功能的两个 M 指令"= -1"，则在界面中不显示相应的栏。
[0]	用于激活刀具专用功能 1 的 M 代码
[1]	用于激活刀具专用功能 2 的 M 代码
[2]	用于激活刀具专用功能 3 的 M 代码
[3]	用于激活刀具专用功能 4 的 M 代码

MD52282 $MCS_TOOL_MCODE_FUNC_OFF[ ]	用于关闭刀具专用功能的 M 代码
= -1	输出 M 功能。 功能的两个 M 指令"= -1"，则在界面中显示相应的栏。
[0]	用于关闭刀具专用功能 1 的 M 代码
[1]	用于关闭刀具专用功能 2 的 M 代码

[2]	用于关闭刀具专用功能 3 的 M 代码
[3]	用于关闭刀具专用功能 4 的 M 代码

通道专用循环设定数据

SD55212 $SCS_FUNCTION_MASK_TECH_SET	工艺通用的功能标记
= 6	
位 0	刀具预选有效。 在换刀后立即准备下一个刀具。 提示：使用转塔刀架时必须将设定数据的值设置为“0”。
位 1	在公制螺纹时自动计算螺纹深度。
位 2	从列表接收螺纹直径和螺纹深度。

19.4.4 柱面转换（TRACYL）



软件选件

使用该功能需要下列软件选件：

“Transmit 与外表面转换”

前提条件

- 机床上至少要有一根回转轴。
- 铣刀必须在待加工圆柱体上径向定向。

功能

使用柱面转换功能可以进行下列槽加工：

- 圆柱体上的纵向槽
- 圆柱体上的横向槽
- 圆柱体上任意形状的槽

槽的形状要在展开的平坦柱面上编程。 可以通过直线/圆弧，钻削或铣削循环或者通过轮廓铣削（自由轮廓编程）来进行编程。

圆柱体表面转换有两种表现方式：

1. 有槽壁补偿（打开）
2. 没有槽壁补偿（关闭）

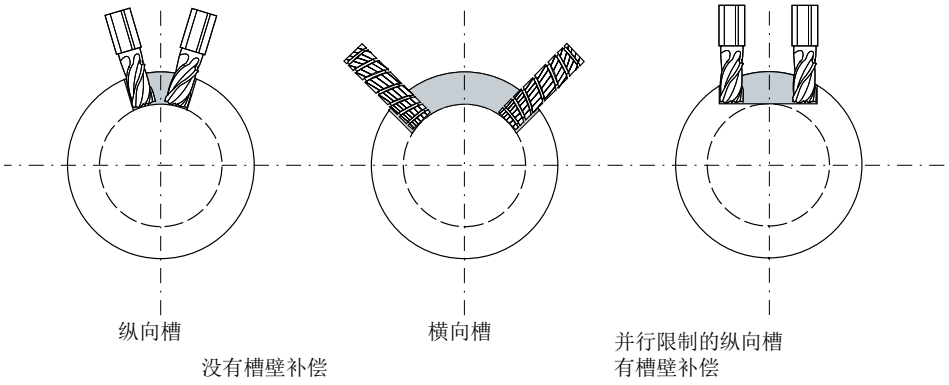


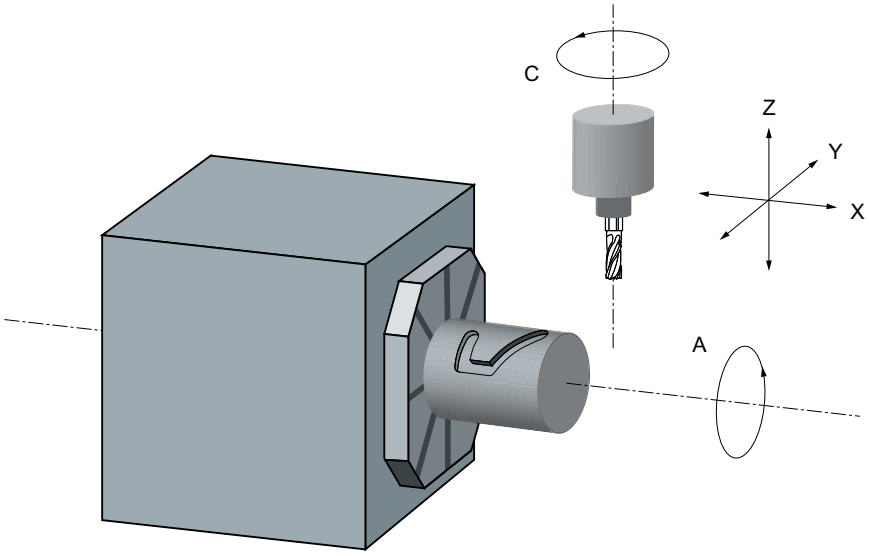
图 19-1　有和没有槽壁补偿的键槽

19.4.5　示例： 带有轴配置 XYZ-AC 的铣床

示例：带有轴配置 XYZ-AC 的铣床

以下述机床为例说明轴的配置方式：

轴配置



X　1. 加工平面的轴平行于回转轴

Y　2. 加工平面的轴

Z　进给轴（刀具轴），垂直于（径向）回转轴

A　回转轴

C　工作主轴

图 19-2　使用 X-A-Z 运动在圆柱面上进行槽加工

上述机床必须使用下列机床数据对两个数据组进行配置：

MD20070 $MC_AXCONF_MACHAX_USED[4]	通道内有效的机床轴号
= 5	通道轴的数量

MD20080 $MC_AXCONF_CHANAX_NAME_TAB[ ]	通道内的通道轴名称
[0] = XC	通道轴 XC， 相当于通道轴 1
[1] = YC	通道轴 YC， 相当于通道轴 2
[2] = ZC	通道轴 ZC，相当于通道轴 3
[3] = A	通道轴 A，相当于通道轴 4
[4] = C	通道轴 C，相当于通道轴 5

MD28082 $MC_MM_SYSTEMFRAME_MASK	系统框架（SRAM）
= 21H	缺省设置
位 0 = 1	系统框架用于实际值设置和对刀
位 5 = 1	循环系统框架
位 6 = 1	系统框架用于转换

用于转换的通用设置

MD10602 $MN_FRAME_GEOAX_CHANGE_MODE	切换几何尺寸标记时的框架
= 1	在切换几何尺寸标记时（选定、取消 TRACYL）重新计算当前有效的总框架（零点偏移）。

通道内转换 1 的轴数据组

MD24100 $MC_TRAFO_TYPE_1	定义通道内的转换 1 。
= 512	转换类型 （512 = 没有槽壁补偿时的柱面转换）

MD24110 $MC_TRAFO_AXES_IN_1	通道内转换 1 的轴分配	
[0] = 3	通道轴： 进给轴（刀具轴），垂直于（径向）回转轴	Z
[1] = 4	通道轴： 回转轴	A
[2] = 1	通道轴： 加工平面内的轴 1，平行于回转轴	X
[3] = 2	通道轴： 加工平面的轴 2	Y

MD24120 $MC_TRAFO_GEOAX_ASSIGN_TAB_1	转换 1 时几何轴与通道轴的分配。
[0] = 1	通道轴： 第 1 几何轴	X
[1] = 4	通道轴： 第 2 几何轴	A
[2] = 3	通道轴： 第 3 几何轴	Z

MD24800 $MC_TRACYL_ROT_AX_OFFSET_1	
= 0	用于 TRACYL 转换 1 的回转轴偏移。

MD24805 $MC_TRACYL_ROT_AX_FRAME_1	
= 1	用于 TRACYL 转换 1 的回转轴轴向偏移。

MD24810 $MC_TRACYL_ROT_SIGN_IS_PLUS_1	
= 1	用于 TRACYL 转换 1 的回转轴符号。

MD24820 $MC_TRACYL_BASE_TOOL_1[ ]	
[0] = 0	用于 XYZ 中 TRACYL 转换 1 的基本刀具矢量
[1] = 0	
[2] = 0	

通道内转换 2 的轴数据组

MD24200 $MC_TRAFO_TYPE_2	
= 513	转换类型 （513 = 有槽壁补偿时的柱面转换）

MD24210 $MC_TRAFO_AXES_IN_2		
[0] = 3	通道轴： 横向进给轴垂直于（径向）回转轴	Z
[1] = 4	通道轴：回转轴	A
[2] = 1	通道轴： 加工平面内的轴 1，平行于回转轴	X
[3] = 2	通道轴： 加工平面的轴 2	Y

MD24220 $MC_TRAFO_GEOAX_ASSIGN_TAB_2		
[0] = 1	通道轴：第 1 几何轴	X
[1] = 4	通道轴：第 2 几何轴	A
[2] = 3	通道轴：第 3 几何轴	Z

MD24850 $MC_TRACYL_ROT_AX_OFFSET_2	
= 0	用于 TRACYL 转换 2 的回转轴偏移

MD24855 $MC_TRACYL_ROT_SIGN_IS_FRAME_2	
= 1	用于 TRACYL 转换 2 的轴向偏移

MD24860 $MC_TRACYL_ROT_SIGN_IS_PLUS_2	
= 1	用于 TRACYL 转换 2 的回转轴符号

MD24870 $MC_TRACYL_BASE_TOOL[ ]	
[0] = 0	用于 XYZ 中 TRACYL 转换 2 的基本刀具矢量
[1] = 0	
[2] = 0	

说明

转换功能所需的数据组

对于这两个数据组可以使用所提供转换（MD24100 $MC_TRAFO_TYPE_1，MD24200 $MC_TRAFO_TYPE_2 等）中的任意两个转换。这两个数据组不一定要前后相随。

数据组 1 必须始终用于“**没有**槽壁补偿的柱面转换”（= 512），而数据组 2 用于“**有**槽壁补偿的柱面转换”（= 513）。

19.4.6 用于多工件加工的 ShopMill 循环



软件选件

使用该功能需要下列软件选件：“ShopMill/ShopTurn”

功能

多工件加工功能可以连续加工多个工件，优化换刀过程。首先它可以节省加工准备时间。其次可以缩短换刀时间，因为只有在完成所有工位上工件的加工后，才会换入下一把刀具。

不同工位上可以运行同一个程序，也可以运行不同的程序。如果机床上使用的是平面夹具，则不需要进行其他设置。

如果是旋转式夹具，您需要根据夹具条件调整循环，使得在加工完一个工件后夹具可以将下一个工件旋转到加工位置，或者在多夹具机床上加工一个工件的同时使另一个夹具将下一个工件旋转到加工位置。

调试步骤

1. 修改目录 cycle/sc/prog/templates_deu 或 cycle/sc/prog/templates_eng 中的循环 CLAMP.SPF。
2. 将循环复制到用户或制造商目录中。

示例

此处使用包含 4 个工位的旋转式夹具（交换工作台）。 回转轴 A 每旋转一次，便加工一个工件：

工件 1：	A = 0°
工件 2：	A = 90°
工件 3：	A = 180°
工件 4：	A = 270°

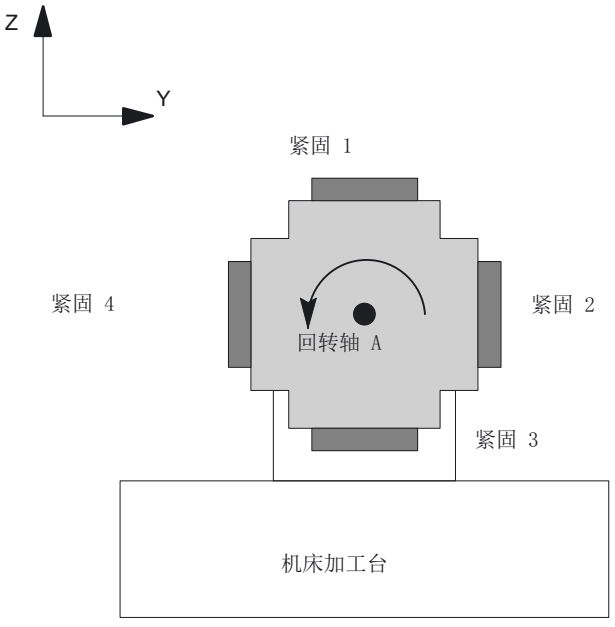


图 19-3　旋转式夹具（交换工作台）

接着必须按如下方式调整循环 CLAMP.SPF：

```
...
DEF INT _NV ;  帮助变量
```

```
;-----------------------------------
; 调整
;
IF _ACT==1
G0 A=DC(0)
ENDIF
;
IF _ACT==2
G0 A=DC(90)
ENDIF
;
IF _ACT==3
G0 A=DC(180)
ENDIF
;
IF _ACT==4
G0 A=DC(270)
ENDIF
;
;-----------------------------------
_NV=_NPV+_ACT ;  计算当前零点偏移
N10 G[8]=_NV ;  此外不许进行计算
RET
```

19.5 车削

19.5.1 常规

标准循环的机床数据和设定数据的所有设置均适用于 ShopMill 铣削和 ShopTurn 车削。

19.5.2 设置用于车削的循环

设置坐标系

MD52000 $MCS_DISP_COORDINATE_SYSTEM	坐标系的位置
= 0	缺省设置

示例：

0：立式车床（旋转车床）
19:卧式车床，在旋转中心前方进行加工
34:卧式车床，在旋转中心后方进行加工（倾斜式床身车床）

MD52000 的详细说明请参见章节：“激活车削/铣削/钻削/磨削工艺 (页 407)”。

车削功能屏幕

通过下列通道专用的配置机床数据，您可以在车削功能屏幕中激活各种功能。

MD52218 $MCS_FUNCTION_MASK_TURN	车削功能屏幕
位 0	手动测量刀具时激活放大镜
位 1	激活切断工件时的工件托盘
位 2	激活尾座
位 3	激活界面上主主轴的控制
位 4	激活界面上副主轴的控制
位 5	使能界面上刀具主轴的控制
位 6	使能双通道式切削的“Balance Cutting”
位 7	使用 G1 进行轮廓切削时的退刀
位 8	程序中主轴卡盘数据输入
位 9	程序中附加的尾座数据输入
位 10	使能球螺纹
位 12	关闭螺纹同步
位 13	使用 CYCLE95 进行轮廓切削（828D programGUIDE，不带 Advanced Technology

轴的含义

在机床数据 MD52206 $MCS_AXIS_USAGE[n]中输入轴的含义。

MD52206 $MCS_AXIS_USAGE[n]	通道中轴的含义 [n]为通道轴号

MD52206 的详细说明请参见章节：“激活车削/铣削/钻削/磨削工艺 (页 407)”

运行方向

进行以下设置，使操作界面和循环了解主轴和回转轴的旋转方向。

通过 MD32100 $MA_AX_MOTION_DIR 设置 NC 回转轴的旋转方向。

MD32100 $MA_AX_MOTION_DIR	运行方向
-1	方向反转
0, 1	无方向反转

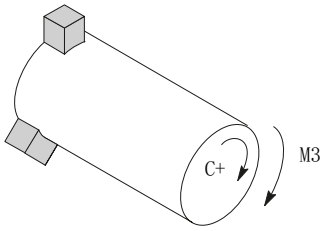
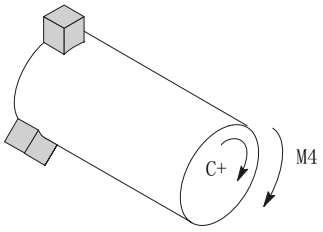
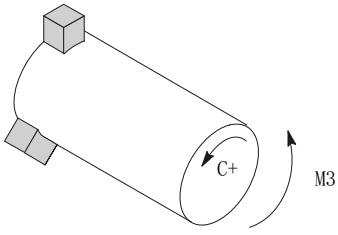
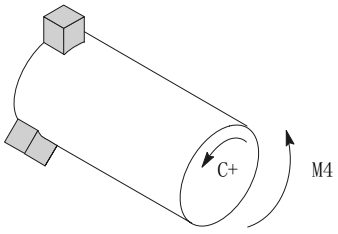
确定旋转方向

为了使主轴和 C 轴的旋转方向正确显示在操作界面中，并且在编程时会执行正确的旋转方向，一些设置必须相互协调，此处的设置必须与机床上主轴/ C 轴的实际旋转方向一致。

在下列机床数据中进行设置：

MD52207 $MCS_AXIS_USAGE_ATTRIB[n]	轴的属性 [n]通道轴号
位 3	显示的正旋转方向是逆时针旋转（回转轴） 旋转方向是从机床内部观察回转轴得出的方向（相当于操作员视角）。
位 4	显示的旋转方向 M3 为逆时针旋转（主轴） 该位可以从操作员的视角出发设置或者根据 DIN66025 设置。
位 5	旋转方向 M3 为回转轴负向（主轴）。 该位必须按 PLC 位 DB31, ... DBX17.6 进行设置！

由此可以得出下列用于主主轴的设置选项：

主主轴旋转方向	52207[n] 位 3 =	52207[n] 位 4 = *)	52207[n] 位 5 =	DB31, ... DBX17.6
	0	0	0	0
	0	1	1	1
	1	1	0	0
	1	0	1	1

说明

MD52207[n]位 4 = *)

位 4 可以从操作员视角出发设置（值如上），也可以依据 DIN（0 和 1 互调）设置。

尺寸

为了确定副主轴运行时的参考点，必须知道副主轴的尺寸。既可以在下列轴专用循环机床数据中，也可以在菜单“参数”→“设定数据”→“主轴卡盘数据”中输入尺寸。机床数据中的修改会被自动接收到菜单中，反之亦然。

MD53240 $MAS_SPINDLE_PARAMETER[ ]	主轴卡盘数据
[0]	卡盘尺寸
[1]	挡块尺寸
[2]	卡爪尺寸

MD53241 $MAS_SPINDLE_CHUCK_TYPE[ ]	主轴卡盘类型
= 0	外部夹紧
= 1	内部夹紧

其他设置

在下列机床数据中 M 代码例如 M34 或 M1 = 34 用来定义主轴卡盘。制造商循环 CUST_TECHCYC.SPF 也使用下列机床数据中的 M 功能：

MD52250 $MCS_M_CODE_CHUCK_OPEN[ ]	主轴静止时用于打开卡盘的 M 代码
[0]	主主轴
[1]	副主轴

MD52251 $MCS_M_CODE_CHUCK_CLOSE_OPEN_ROT[ ]	主轴旋转时用于打开卡盘的 M 代码
[0]	主主轴
[1]	副主轴

MD52252 $MCS_M_CODE_CHUCK_CLOSE[ ]	用于闭合卡盘的 M 代码
[0]	主主轴
[1]	副主轴

MD52253 $MCS_M_CODE_TAILSTOCK_FORWARD[ ]	用于前面套筒的 M 代码
[0]	与主主轴相对的套筒
[1]	

MD52254 $MCS_M_TAILSTOCK_BACKWARD[ ]	用于后部套筒的 M 代码
[0]	与主主轴相对的套筒
[1]	与主主轴相对的套筒

轮廓车削（CYCLE952）

SD55212 $SCS_FUNCTION_MASK_TECH_SET	工艺通用的功能屏幕
= 6	
位 3	删除由轮廓循环（CYCLE63、CYCLE64、CYCLE952）生成的程序
= 0	生成的程序未删除（兼容性，和目前一致）
= 1	编辑完调用循环后立即再次删除生成的程序

收紧轮廓

SD55582 $SCS_TURN_CONT_TRACE_ANGLE	车削轮廓：指定收紧轮廓的最小角度
= 5	指定刀沿与轮廓的夹角值，在车削轮廓时从此开始收紧轮廓，将剩余的材料清除（缺省值）。

SD 55218 $SCCS_FUNCTION_MASK_TURN_SET	车削功能屏幕
位 3	作为 CYCLE930 中倒角的长度 (CHF)
位 4	从内部加工中退回 (CYCLE951)
= 0	兼容性，和目前一致 如果在循环中识别到拐角收紧，则会执行从内部加工中退回，否则不执行。
= 1	同时将切削轴从内部加工中退回。

19.5.3 设置用于车削的 ShopTurn 循环

MD52210 $MCS_FUNCTION_MASK_DISP	显示功能屏幕
位 0 = 1	程序始终采用系统的单位制（缺省）
位 1 = 1	车削时的端面视图（缺省设置）
位 4 = 1	在 T、F、S 窗口中显示下一把刀具

MD52212 $MCS_FUNCTION_MASK_TECH	工艺通用的功能屏幕
位 2	阶梯钻头的逼近逻辑
位 3	程序段搜索 - 用于 ShopMill/ShopTurn 的循环
= 0	在程序段搜索循环 PROG_EVENT.SPF 中，不调用循环 E__ASUP 或 F_S_ASUP 。
= 1	在程序段搜索循环 PROG_EVENT.SPF 中，调用循环 E__ASUP （在 ShopMill 中）或 F_S_ASUP （在 ShopTurn 中）。
位 4	通过循环的逼近逻辑
位 5	在程序段搜索循环 PROG_EVENT.SPF 中调用用于 SERUPRO （CYCLE207）的循环。 设置了位 5 时，位 3 不会生效。
位 6	零点偏移值不可输入
位 7	在程序中监测刀具寿命（ShopMill / ShopTurn）
位 8	手动加工(ShopMill/ShopTurn)
位 9	通过软键选定/取消零点偏移补偿
位 11	切断钻削刀具和铣削刀具位置检查

MD52214 $MCS_FUNCTION_MASK_MILL	铣削功能屏幕
位 3	在 ShopTurn 屏幕中激活了“内部/后部”加工，它们可以自行确定加工平面。
位 4	如果想通过制造商循环 CUST_TECHCYC.SPF 实现“夹紧/松开主轴”功能，可以使用该机床数据在钻削与铣削屏幕中激活参数“夹紧/松开主轴”。
= 0	参数“夹紧/松开主轴”不显示在钻削与铣削屏幕中。 当前加工需要时，ShopTurn 会自动夹紧主轴。
= 1	参数“夹紧/松开主轴”不显示在钻削与铣削屏幕中。 可确定在哪种加工时夹紧主轴。
位 8	启用阻尼制动（C 轴）

= 0	参数“阻尼制动开/关”不显示在铣削屏幕中。
= 1	参数“阻尼制动开/关”显示在铣削屏幕中。 可确定在哪种加工方式（端面 C/外表面 C）中使 C 轴制动。 通过机床制造商循环“CUST_TECHCYC.SPF ”实现了“阻尼制动”功能后，便可以使用该机床数据在铣削屏幕中激活参数“阻尼制动开/关”。

MD52216 $MCS_FUNCTION_MASK_DRILL	钻削功能屏幕
位 3	基于中心孔直径的恒定切削速度
= 0	基于刀具直径的恒定切削速度
= 1	基于中心孔直径的恒定切削速度

MD52218 $MCS_FUNCTION_MASK_TURN	车削功能屏幕
位 2	激活尾座
位 5	使能界面上刀具主轴的控制
位 6	使能双通道式切削的“Balance Cutting”

MD52229 $MCS_ENABLE_QUICK_M_CODES	使能 M 指令
= 0	
位 0	冷却液 关
位 1	冷却液 1 开
位 2	冷却液 2 开
位 3	冷却液 1 和 2 开

MD52230 $MCS_M_CODE_ALL_COOLANTS_OFF	用于全部冷却液关闭的 M 代码
= 9	使用该机床数据可以定义用于冷却液关闭的 M 功能，在换刀时输出该功能。

MD52231 $MCS_M_CODE_COOLANT_1_ON	用于冷却液 1 打开的 M 代码
= 8	使用该机床数据可以定义用于冷却液 1 的 M 功能，在换刀时输出该功能。

MD52232 $MCS_M_CODE_COOLANT_2_ON	用于冷却液 2 打开的 M 代码
= 7	使用该机床数据可以定义用于冷却液 2 的 M 功能，在换刀时输出该功能。

MD52233 $MCS_M_CODE_COOLANT_1_AND_2_ON	用于两个冷却液打开的 M 代码
= -1	使用该机床数据可以定义用于冷却液 1 和 2 的 M 功能，在换刀时输出该功能。

SD55505 $SCS_TURN_ROUGH_O_RELEASE_DIST	指定轮廓切削中加工外部时的回退距离
= 1	指定距离值，刀具在切削外角时按该值从轮廓退回。这不适用于轮廓切削（缺省值）。
= -1	距离为内部定义。

SD55506 $SCS_TURN_ROUGH_'I_RELEASE_DIST	指定轮廓切削内部加工时的回退距离。
= 0.5	指定距离值，刀具在切削内角时按该值从轮廓退回。这不适用于轮廓切削（缺省值）。
= -1	距离为内部定义。

19.5.4 设置副主轴



软件选件

为了在机床上使用副主轴，需要使用下列软件选件：

“通用耦合 CP-STATIC”或更高级耦合

功能

如果车床有副主轴，不必手动重新卡持工件，即可使用车削、钻削和铣削功能加工工件的正面和背面。

开始在背面加工之前，副主轴必须夹紧工件，将其从主主轴上卸下，并移动到新的加工位置。

轴的含义

MD52206 $MCS_AXIS_USAGE[n]	通道中轴的含义 [n]为通道轴号

MD52006 的详细说明请参见章节：“激活车削/铣削/钻削/磨削工艺 (页 407)”。

运行方向

进行以下设置，使操作界面和循环了解主轴和回转轴的旋转方向。

通过 MD32100 $MA_AX_MOTION_DIR 设置 NC 回转轴的旋转方向。

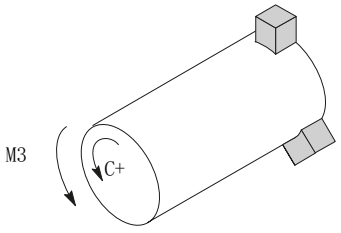
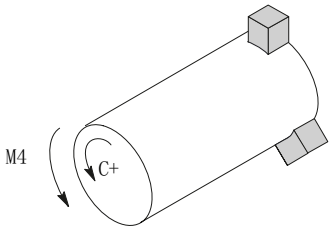
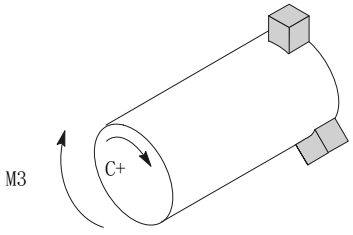
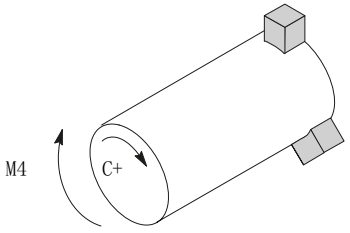
MD32100 $MA_AX_MOTION_DIR	运行方向（非调节方向）
-1	方向反转
0, 1	无方向反转

确定旋转方向

在以下机床数据中设置平面的旋转方向：

MD52207 $MCS_AXIS_USAGE_ATTRIB[n]	通道中轴的属性 [n] 通道轴号
位 3	显示的正旋转方向是逆时针旋转（回转轴） 旋转方向是从机床内部观察回转轴得出的方向（相当于操作员视角）。
位 4	显示的旋转方向 M3 为逆时针旋转（主轴） 该位可以从操作员的视角出发设置或者根据 DIN 设置。
位 5	旋转方向 M3 为回转轴负向（主轴）。 该位必须按 PLC 位 DB31, ... DBX17.6 进行设置！

由此可以得出下列用于副主轴的设置选项：

副主轴旋转方向	52207[n] 位 3 =	52207[n] 位 4 = *)	52207[n] 位 5 =	DB31, ... DBX17.6
	1	1	0	0
	1	0	1	1
	0	0	0	0
	0	1	1	1

说明

MD52207[n]位 4 = *)

位 4 可以从操作员视角出发设置（值如上），也可以依据 DIN（0 和 1 互调）设置。

主主轴和副主轴的旋转方向设置必须相同，要么都从操作员视角出发设置，要么都依据 DIN。

尺寸

为了确定副主轴运行时的参考点，必须知道副主轴的尺寸。既可以在下列轴专用循环机床数据中，也可以在菜单“参数”→“设定数据”→“主轴卡盘数据”中输入尺寸。机床数据中的修改会被自动接收到菜单中，反之亦然。

MD53240 $MAS_SPINDLE_PARAMETER[ ]	主轴卡盘数据
[0]	卡盘尺寸
[1]	挡块尺寸
[2]	卡爪尺寸

MD52241 $MCS_SPINDLE_CHUCK_TYPE	主轴卡盘类型
= 0	外部夹紧
= 1	内部夹紧

19.5.5 设置 ShopTurn 中的副主轴



软件选件

为了在机床上使用副主轴，需要使用下列软件选件：

“通用耦合 CP-STATIC”或更高级耦合

副主轴在程序开始时运行到的目标位置，在通道专用的循环设定数据中确定：

SD55232 $SCS_SUB_SPINDLE_REL_POS	副主轴的 Z 轴退回位置

当固定挡块运行被激活时，以下各通道的设置数据才会起作用。

SD55550 $SCS_TURN_FIXED_STOP_DIST	运行至固定挡块的路径
SD55551 $SCS_TURN_FIXED_STOP_FEED	运行至固定挡块的进给率
SD55552 $SCS_TURN_FIXED_STOP_FORCE	运行至固定挡块的力，以 % 为单位。

在运行至固定挡块与夹紧之间副主轴可以回退一小段，用来抵消工件中的压缩应力。

SD55553 $SCS_TURN_FIXED_STOP_RETRACTION	夹紧前至固定点的回退行程。

夹紧后可以切断工件。 副主轴可以事先带着工件回退一小段，用来加大工件张应力。 借此在切断时去除刀具负荷。

SD55543 $SCS_TURN_PART_OFF_RETRACTION	切断之前的回退。

切断之后可以执行出料检查并在车削时使用“运行至固定挡块”功能。 可以通过下列通道专用循环设定数据设置或激活出料检查：

SD55540 $SCS_TURN_PART_OFF_CTRL_DIST	用于出料检查的行程。
SD55541 $SCS_TURN_PART_OFF_CTRL_FEED	用于出料检查的进给率
SD55542 $SCS_TURN_PART_OFF_CTRL_FORCE	用于出料检查的力，以 % 为单位。

如果运行至固定挡块失败，则需要进行切断。 会输出下列报警：

报警	报警文本
20091	轴 % 1 未到达固定点。
20094	轴 % 1 功能已被中断。

可以通过下列机床数据关闭报警显示：

MD37050 $MA_FIXED_STOP_ALARM_MASK	激活到达固定点报警。
= 2	抑制报警 20091 和 20094。

可以在“机床数据”窗口的操作区“刀具零点”中设置这个轴专用的机床数据。

如在出料检查时未达到给定的力（即运行至固定挡块成功），则发出报警 61255“切断故障：刀具受损？”。

说明

主轴夹紧时也可以使用“运行至固定挡块”功能（参见上面）。 如果在这种情况下运行至固定挡块失败，当然也会输出报警。 但不是报警 20091 和 20094，而会显示报警 61254“运行至固定挡块时故障”。

说明

MD37050 也可以包含不等于 2 的值，例如：10 = 2 + 8。 确保包含逻辑值 2，这样才会隐藏运行至固定挡块监控。 如果 MD37050 不包含值 2，则目标检查不会生效。

机床制造商循环

如要执行下列动作之一，必须对机床制造商循环 CUST_TECHCYC.SPF 进行匹配。

- 主轴和 C 轴运行时在主主轴和副主轴间切换。
- 打开，关闭，清洗卡盘（主主轴/副主轴）。
- 修改用于主主轴与副主轴耦合的预设置。

相关的具体步骤请查阅下列章节：

设置用于车削的 ShopTurn 循环 (页 443)

19.5.6 用于车削的工艺循环

螺纹切削（CYCLE99）

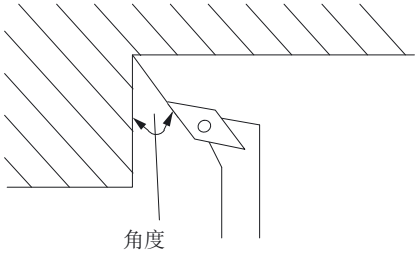
螺纹的正确加工需要设置机床数据 MD52207 $MCS_AXIS_USAGE_ATTRIB[n] 位 3，参见章节：设置副主轴 (页 445)。

轮廓槽式车削 CYCLE930、轮廓车削 CYCLE950、拐角车削 CYCLE951、轮廓车削 CYCLE952

SD55500 $SCS_TURN_FIN_FEED_PERCENT		在进行粗加工与精加工的综合加工时输入精加工进给率。 百分比值等于参数 F（进给率）中的输入值。
= 100	100 % 精加工进给率	

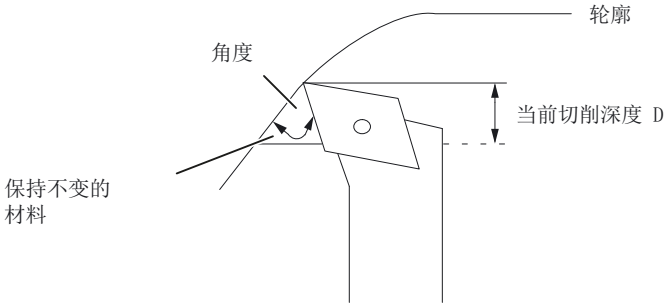
SD55510 $SCS_TURN_GROOVE_DWELL_TIME		工艺“槽式车削”中刀具切入与撤回之间的暂停时间， 在“切槽”中，底部的退回时间
= > 0	暂停时间，单位秒	
= < 0	暂停时间，单位：主轴转数	

SD55580 $SCS_TURN_CONT_RELEASE_ANGLE		“轮廓车削”中粗加工时，刀具按该角度退刀。
= 45	退回角度为 45 度	



SD55581 $SCS_TURN_CONT_RELEASE_DIST	粗加工轮廓时，采用退刀角度 SD55580 $SCS_TURN_CONT_RELEASE_ANGLE 的退刀量。 这些数据都适用于轮廓车削、槽式车削和往复车削。
= 1	退刀量为 1 毫米或 1 英寸

SD55582 $SCS_TURN_CONT_TRACE_ANGLE	刀沿与轮廓的夹角，在车削轮廓时从此开始收紧轮廓将剩余材料清除。 如果该角度大于设定数据中所确定的角度，则刀具开始紧跟轮廓。
= 5	角度为 5 度



SD55583 $SCS_TURN_CONT_VARIABLE_DEPTH	轮廓车削中可变切削深度百分比值。 在“轮廓车削”和“轮廓车削余料”中，您可以选择可变的切削深度。
= 20	20 % 可变切削深度

SD55584 $SCS_TURN_CONT_BLANK_OFFSET	绕行障碍物时的安全距离，以免在走刀和退刀时出现碰撞。 这些数据都适用于轮廓车削、槽式车削和往复车削。
= 1	毛坯余量为 1 毫米或 1 英寸

SD55585 $SCS_TURN_CONT_INTERRUPT_TIME	车削轮廓时的进给中断时间。 这些数据都适用于轮廓车削、槽式车削和往复车削。
= > 0	进给中断时间，以秒为单位
= < 0	进给中断时间，以转为单位
= 0	无中断

说明

只有当 SD55586 $SCS_TURN_CONT_INTER_RETRACTION = 0 时，才分析通道专用循环设定数据 SD55585 。

SD55586 $SCS_TURN_CONT_INTER_RETRACTION	车削轮廓时进给中断的回退行程。 这些数据都适用于轮廓车削、槽式车削和往复车削。
= > 0	进给中断的回退行程 SD55585 $SCS_TURN_CONT_INTERRUPT_TIME 无效。
= 0	无回退行程

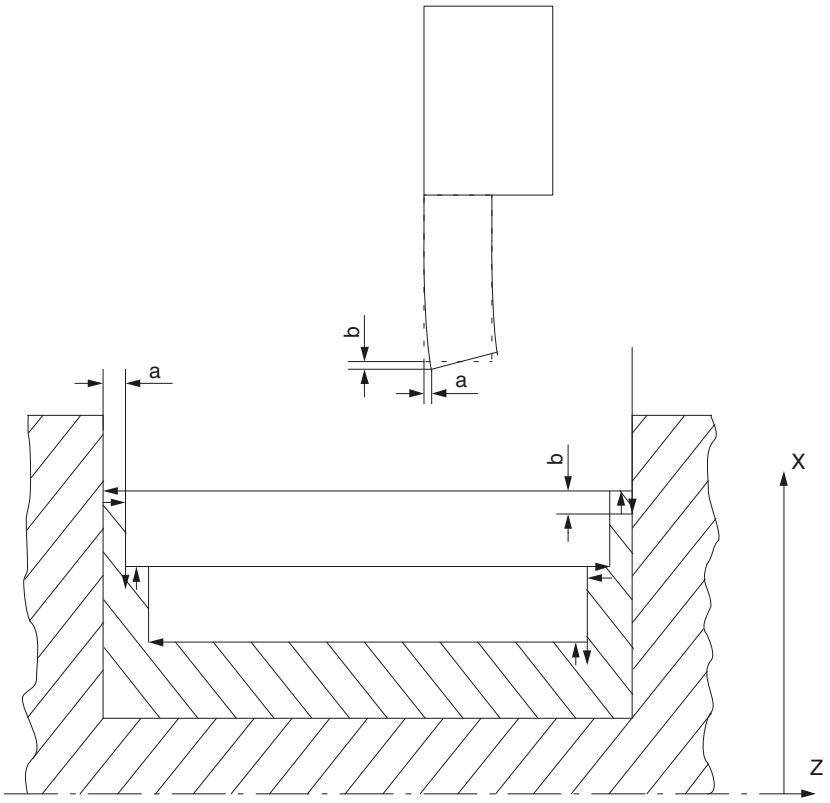
SD55587 $SCS_TURN_CONT_MIN_REST_MAT_AX1	轮廓车削中轴 1（G18 为加工平面时为 Z 轴）方向上清理余料的极限值。 这些数据都适用于轮廓车削余料、槽式车削和往复车削。
50	如果待切削材料少于精加工余量的 50%，则不进行余料加工的粗加工步骤。 在精加工时才开始切去材料。

示例： 如果设定数据的设置为 50% 并且精加工余量为 0.5 毫米，则小于 0.25 毫米的余量在余料加工时不会被去除，而是在精加工时进行切除。 如果某个加工步骤中的余料比设定数据中规定的尺寸小，系统会报告故障“没有材料”。 也就是说，这一余料粗切步骤被跳过，因为没有余料可被切除。

SD55588 $SCS_TURN_CONT_MIN_REST_MAT_AX2	轮廓车削中轴 2（G18 为加工平面时为 X 轴）方向上清理余料的极限值。 这些数据都适用于轮廓车削余料、槽式车削和往复车削。
50	如果待切削材料少于精加工余量的 50%，则余料加工时不进行粗加工步骤。 在精加工时才开始切去材料。

由于“槽式车削”中的刀具弯曲，可能会在“轮廓车削”中无法完全逼近轮廓， 下一次切削中需要缩短的侧面间距，在以下通道专用的循环设定数据中定义。

SD55595 $SCS_TURN_CONT_TOOL_BEND_RETR	刀具弯曲产生的退刀距离
0.1	退刀距离为 0.1 毫米或 0.1 英寸



a 和上一次切削相隔的间距， SD55595

b “槽式车削”和“轮廓车削”之间的退刀， SD55596

由于“槽式车削”中的刀具弯曲，可能会在“轮廓车削”中形成更深的切削。“槽式车削”和“轮廓车削”之间的退刀距离可以在下列通道专用循环设定数据中定义。

SD55596 $SCS_TURN_CONT_TURN_RETRACTION	车削前的退刀深度
= 0.1	退刀深度为 0.1 毫米或 0.1 英寸

19.5.7 车床的轴配置

通用配置

当车床上提供的是传动铣刀时，可以在该机床上另外设置下列功能：

- 柱面转换（TRACYL） (页 456)
- 端面加工（TRANSMIT） (页 460)

带 X 和 Z 轴、主主轴与刀具主轴的车床

对于带 X 和 Z 轴以及主主轴（C1）与刀具主轴（WZ）的车床可以象下面这样配置通道专用机床数据：

MD20080 $MC_AXCONF_CHANAX_NAME_TAB[ ]	通道内的通道轴名称
[0] = XC	通道轴 XC
[1] = ZC	通道轴 ZC
[2] = C1	主主轴 C1
[3] = WZ	刀具主轴 WZ

带 X 和 Z 轴、主主轴、刀具主轴与副主轴的车床

对于带 X 和 Z 轴以及主主轴（C1），刀具主轴（WZ）与主主轴（C2）的车床可以象下面这样配置机床数据：

MD20080 $MC_AXCONF_CHANAX_NAME_TAB[ ]	通道内的通道轴名称
[0] = XC	通道轴 XC
[1] = ZC	通道轴 ZC
[2] = C1	主主轴 C1
[3] = WZ	刀具主轴 WZ
[5] = C2	副主轴 C2

带 X 和 Z 轴、主主轴、刀具主轴与 Y 轴的车床

对于带 X、Z 和 Y 轴以及主主轴（C1）与刀具主轴（WZ）的车床可以象下面这样配置机床数据：

MD20080 $MC_AXCONF_CHANAX_NAME_TAB[ ]	通道内的通道轴名称
[0] = XC	通道轴 XC
[1] = ZC	通道轴 ZC
[2] = C1	主主轴 C1
[3] = WZ	刀具主轴 WZ
[5] = YC	通道轴 YC

19.5.8 柱面转换（TRACYL）



软件选件

使用该功能需要下列软件选件：

“Transmit 与外表面转换”

功能

使用柱面转换（TRACYL）功能可以加工车削件的外表面。

调整

用于柱面转换的通用设置：

MD10602 $MN_FRAME_GEOAX_CHANGE_MODE	几何轴切换时的框架。
= 1	在切换几何轴时（选定/取消 TRACYL）会重新计算当前有效的总框架（零点偏移）。

MD24040 $MC_FRAME_ADAPT_MODE	调整有效的框架。
位 0 = 1	如果有效框架中坐标轴的旋转没有可用的几何轴，该旋转会从生效框架中删除。
位 1 = 1	生效框架中的剪切角变为垂直角。
位 2 = 1	将生效框架中所有几何轴的比例缩放设为 1。

MD28082 $MC_MM_SYSTEM_FRAME_MASK	
位 6 = 1	用于补偿的转换框架配置。

在设置功能时可以考虑下列通道专用机床数据：

MD24300 $MC_TRAFO_TYPE_3	主主轴柱面转换：转换 3
= 512	没有槽壁补偿（无 Y 轴）
= 513	有槽壁补偿（带 Y 轴）
= 514	有或无槽壁补偿与 Y 补偿

MD24400 $MC_TRAFO_TYPE_4	副主轴柱面转换：坐标转换 4
= 512	没有槽壁补偿（无 Y 轴）
= 513	有槽壁补偿（带 Y 轴）
= 514	有或无槽壁补偿与 Y 补偿

说明

对于单个转换还必须分别设置其他机床数据。

没有槽壁补偿的柱面转换

MD24300 $MC_TRAFO_TYPE_3	在通道中定义第 3 个转换。
= 512	TRACYL 主主轴。

MD24310 $MC_TRAFO_AXES_IN_3[ ]	用于第 3 个转换的轴分配。
[0] = 1	垂直于回转轴 XC
[1] = 3	回转轴（主主轴） C1
[2] = 2	平行于回转轴 ZC

MD24320 $MC_TRAFO_GEOAX_ASSIGN_TAB_3[ ]	用于转换 3 的几何轴与通道轴的分配。
[0] = 1	第 1 通道轴 X 轴
[1] = 3	第 2 通道轴 Y 轴
[2] = 2	第 3 通道轴 Z 轴

MD24800 $MC_TRACYL_ROT_AX_OFFSET_1	用于第 1 个 TRACYL 转换的回转轴偏移。
= 0	

MD24805 $MC_TRACYL_ROT_AX_FRAME_1	在 TRACYL 时考虑回转轴的轴向偏移。
= 2	

MD24810 $MC_TRACYL_ROT_SIGN_IS_PLUS_1	用于 TRACYL 转换 1 的回转轴符号。
= 1	

MD24820 $MC_TRACYL_BASE_TOOL_1[ ]	用于第 1 个 TRACYL 转换的基本刀具矢量。
[0] = 0	
[1] = 0	
[2] = 0	

设置 ShopTurn 中的柱面转换

另外要设置下列通道专用配置机床数据：

MD52214 $MCS_FUNCTION_MASK_MILL	铣削功能屏幕。
位 3	在 ShopTurn 屏幕中激活“内部/后部”加工，它们可以自行确定加工平面。
位 4	如果想通过机床制造商循环 CUST_TECHCYC.SPF 实现“夹紧/松开主轴”功能，可以使用该机床数据在钻削与铣削屏幕中激活参数“夹紧/松开主轴”。
= 0	参数“夹紧/松开主轴”不显示在钻削与铣削屏幕中。 当前加工需要时，ShopTurn 会自动夹紧主轴。
= 1	参数“夹紧/松开主轴”显示在钻削与铣削屏幕中。 操作员可以决定，在何种加工时夹紧主轴。
位 8	启用阻尼制动（C 轴）
= 0	参数“阻尼制动开/关”不显示在铣削屏幕中。
= 1	参数“阻尼制动开/关”显示在铣削屏幕中。 可确定在哪种加工方式（端面 C/外表面 C）中使 C 轴制动。 通过机床制造商循环“CUST_TECHCYC.SPF ”实现了“阻尼制动”功能后，便可以使用该机床数据在铣削屏幕中激活参数“阻尼制动开/关”。

有槽壁补偿的柱面转换

MD24300 $MC_TRAFO_TYPE_3	在通道中定义第 3 个转换。
= 513	TRACYL 主主轴

MD24310 $MC_TRAFO_AXES_IN_3[ ]	用于第 3 个转换的轴分配。
[0] = 1	垂直于回转轴 XC
[1] = 3	回转轴（主主轴） C1
[2] = 2	平行于回转轴 ZC
[3] = 6	平行于圆柱体外表面并垂直于回转轴 ZC

MD24320 $MC_TRAFO_GEOAX_ASSIGN_TAB_3[ ]	用于转换 3 的几何轴与通道轴的分配。
[0] = 1	第 1 通道轴 X 轴
[1] = 3	第 2 通道轴 Y 轴
[2] = 2	第 3 通道轴 Z 轴

MD24800 $MC_TRACYL_ROT_AX_OFFSET_1	用于第 1 个 TRACYL 转换的回转轴偏移。
= 0	

MD24805 $MC_TRACYL_ROT_AX_FRAME_1	在 TRACYL 时考虑回转轴的轴向偏移。
= 2	

MD24810 $MC_TRACYL_ROT_SIGN_IS_PLUS_1	用于 TRACYL 转换 1 的回转轴符号。
= 1	

MD24820 $MC_TRACYL_BASE_TOOL_1[ ]	用于第 1 个 TRACYL 转换的基本刀具矢量。
[0] = 0	
[1] = 0	
[2] = 0	

19.5.9 端面加工（TRANSMIT）



软件选件

使用该功能需要下列软件选件：

“Transmit 与外表面转换”

功能

使用端面加工（TRANSMIT）功能可以加工车削件的端面。

设置

用于转换的通用设置可以参见章节：“柱面转换（TRACYL） (页 456)”。

其他设置可以阅读下列通道专用机床数据：

MD24100 $MC_TRAFO_TYPE_1	主主轴端面加工：转换 1
= 256	无 Y 轴的加工
= 257	带 Y 轴的加工

MD24200 $MC_TRAFO_TYPE_2	副主轴端面加工：转换 2
= 256	无 Y 轴的加工
= 257	带 Y 轴的加工

MD24110 $MC_TRAFO_AXES_IN_1[ ]	通道内转换 1 的轴分配
[0] = 1	垂直于回转轴 XC
[1] = 3	回转轴（主主轴） C1
[2] = 2	平行于回转轴 ZC

MD24120 $MC_TRAFO_GEOAX_ASSIGN_TAB_1[ ]	转换 1 时几何轴与通道轴的分配
[0] = 1	1. 通道轴 X
[1] = 3	2. 通道轴 Y
[2] = 2	3. 通道轴 Z

MD24900 $MC_TRANSMIT_ROT_AX_OFFSET_1	
= 0	用于第 1 个 TRANSMIT 转换的回转轴偏移。

MD24905 $MC_TRANSMIT_ROT_AX_FRAME_1	
= 2	在 TRANSMIT 1 时考虑回转轴的轴向偏移。

MD24910 $MC_TRANSMIT_ROT_SIGN_IS_PLUS_1	
= 0	用于第 1 个 TRANSMIT 转换的回转轴符号。

MD24911 $MC_TRANSMIT_POLE_SIDE_FIX_1	
= 1	极点前/后的工作区极限，第 1 个 TRANSMIT 转换。

MD24920 $MC_TRANSMIT_BASE_TOOL_1[ ]	用于第 1 个 TRANSMIT 转换的基本刀具矢量。
[0] = 0	
[1] = 0	
[2] = 0	

带真正 Y 轴的 TRANSMIT 转换

MD24100 $MC_TRAFO_TYPE_1	
= 257	定义通道内的转换 1：主主轴 TRANSMIT 转换。

MD24110 $MC_TRAFO_AXES_IN_1[ ]	通道内转换 1 的轴分配。
[0] = 1	垂直于回转轴 XC
[1] = 3	回转轴 C1
[2] = 2	平行于回转轴 ZC

MD24120 $MC_TRAFO_GEOAX_ASSIGN_TAB_1[ ]	转换 1 时几何轴与通道轴的分配。
[0] = 1	1. 通道轴 X
[1] = 3	2. 通道轴 Y
[2] = 2	3. 通道轴 Z

MD24900 $MC_TRANSMIT_ROT_AX_OFFSET_1	
= 0	用于第 1 个 TRANSMIT 转换的回转轴偏移。

MD24905 $MC_TRANSMIT_ROT_AX_FRAME_1	
= 2	在 TRANSMIT 1 时考虑回转轴的轴向偏移。

MD24910 $MC_TRANSMIT_ROT_SIGN_IS_PLUS_1	
= 0	用于第 1 个 TRANSMIT 转换的回转轴符号。

MD24911 $MC_TRANSMIT_POLE_SIDE_FIX_1	
= 1	极点前/后的工作区极限，第 1 个 TRANSMIT 转换。

MD24920 $MC_TRANSMIT_BASE_TOOL_1[ ]	用于第 1 个 TRANSMIT 转换的基本刀具矢量。
[0] = 0	
[1] = 0	
[2] = 0	

回转轴其他设置

如果要整合回转轴，除了以下通道专用的机床数据外还需要带位 6 的 MD28082，参见章节 “柱面转换（TRACYL） (页 456)”。

MD24906 $MC_TRANSMIT_BASE_TOOL_COMP_1	第 1 转换框架中的基本偏移补偿。
位 0	第 1 几何轴

位 1	第 2 几何轴
位 2	第 3 几何轴

MD24956 $MC_TRANSMIT_BASE_TOOL_COMP_2	第 2 转换框架中的基本偏移补偿。
位 0	第 1 几何轴
位 1	第 2 几何轴
位 2	第 3 几何轴

更多信息还可参见“毛坯夹具 (页 184)”一章。

ShopTurn 中的端面加工

如要使用 ShopTurn 中的端面加工功能，还要另外设置下列通道专用配置机床数据：

MD52214 $MCS_FUNCTION_MASK_MILL	ShopTurn 功能屏幕。
位 3	在 ShopTurn 屏幕中激活“内部/后部”加工，它们可以自行确定加工平面。
位 4	如果想通过机床制造商循环 CUST_TECHCYC.SPF 实现“夹紧/松开主轴”功能，可以使用该机床数据在钻削与铣削屏幕中激活参数“夹紧/松开主轴”。
= 0	参数“夹紧/松开主轴”不显示在钻削与铣削屏幕中。当对加工有意义时，ShopTurn 会自动夹紧主轴。
= 1	参数“夹紧/松开主轴”显示在钻削与铣削屏幕中。操作员可以决定，在何种加工时夹紧主轴。

说明

端面加工自动集成在循环中，直线或圆弧除外。

对于这两个循环可以在“程序”操作区的“直线”或“圆弧”中选择刀具功能。

更多信息

关于端面加工的更多信息，请参见

功能手册之坐标转换。

19.5.10 斜置 Y 轴 （TRAANG）



软件选件

为了在机床上使用“斜置 Y 轴”功能，需要使用下列软件选件：

“斜置轴”

功能

如果车床具有“斜置 Y 轴”功能（即该轴不垂直于 X 轴和 Z 轴），就可以继续在直角坐标系中对加工进行编程。控制器借助斜置轴（TRAANG）功能将直角坐标转换为斜置轴的运行动作。

设置

此外必须通过机床数据对斜置轴（TRAANG）功能进行设置。

ShopTurn 中的斜置轴

另外，需要注意：

说明

在操作界面中斜置轴在设置后会自动集成到循环中。您可在斜置轴加工时在屏幕中选择加工平面“端面 Y”或“柱面 Y”并输入直角坐标。

更多信息

关于斜置 Y 轴功能的更多信息，请参见

- 功能手册之坐标转换
- NC 编程手册

示例

对于带 X、Z 和斜置 Y 轴以及主主轴（C）与刀具主轴（WZ）的车床可以象下面这样配置机床数据：

MD20050 $MC_AXCONF_GEOAX_ASSIGN_TAB[ ]	分配几何轴给通道轴。
[0] = 1	第 1 实际几何轴 X 轴。

[1] = 0	第 2 实际几何轴 Y 轴不存在。
[2] = 2	第 3 实际几何轴 Z 轴。

MD20110 $MC_RESET_MODE_MASK	
位 0 = 1 位 7 = 0	TRAANG 在引导启动后保持不变。

MD20112 $MC_START_MODE_MASK	
位 7 = 1	TRAANG 在“循环启动”后保持不变。

MD20118 $MC_GEOAX_CHANGE_RESET	
= 1	允许自动更换几何轴。

MD20140 $MC_TRAFO_RESET_VALUE	
= 5	TRAANG 在复位时始终有效。

MD20144 $MC_TRAFO_MODE_MASK	
位 0 = 1	TRAANG 在后台运行（持续）并且不显示在操作界面中。

MD20070 $MC_AXCONF_MACHAX_USED[4]	
= 5	通道轴 YC = 第 5 机床轴

MD20080 $MC_AXCONF_CHANAX_NAME_TAB[ ]	通道内的通道轴名称。
[0] = XC	第 1 通道轴 XC
[1] = ZC	第 2 通道轴 ZC
[2] = C	第 3 通道轴 C
[3] = WZ	第 4 通道轴 WZ
[4] = YC	第 5 通道轴 YC

用于斜置轴的数据程序段：

MD24430 $MC_TRAFO_TYPE_5	
= 1024	转换 5 ：TRAANG

MD24432 $MC_TRAFO_AXES_IN_5[ ]	用于转换 5 的轴分配。
[0] = 5	第 1 转换轴 = 通道轴， YC
[1] = 1	第 2 转换轴 = 通道轴 XC
[2] = 2	第 3 转换轴 = 通道轴 ZC

MD24434 $MC_TRAFO_GEOAX_ASSIGN_TAB_5[ ]	用于转换 5 的几何轴与通道轴的分配。
[0] = 1	第 1 轴 = 通道轴 XC
[1] = 5	第 2 轴 = 通道轴 YC
[2] = 2	第 3 轴 = 通道轴 ZC

MD24436 $MC_TRAFO_INCLUDES_TOOL_5	
= 0	第 5 转换生效时的刀具处理

24700 $MC_TRAANG_ANGLE_1	
= 55	第 1 和第 2 转换轴之间的夹角用于链接（TRACON）主主轴端面加工（TRANSMIT）和斜置轴（TRAANG）的数据组。

用于链接（TRACON）主主轴端面加工（TRANSMIT）和斜置轴（TRAANG）的数据组：

MD24440 $MC_TRAFO_TYPE_6	
= 8192	通道中第六个提供的转换类型

MD24444 $MC_TRAFO_GEOAX_ASSIGN_TAB_6[ ]	用于转换 6 的几何轴与通道轴的分配。
[0] = 1	第 1 轴 = 通道轴 XC
[1] = 3	第 2 轴 = 通道轴 YC
[2] = 2	第 3 轴 = 通道轴 ZC

MD24995 $MC_TRACON_CHAIN_1[ ]	转换链接
[0] = 1	用于链接的转换编号 TRANSMIT （主主轴）。
[1] = 5	用于链接的转换编号 TRAANG 用于链接（TRACON）主主轴柱面转换（TRACYL）与斜置轴（TRAANG）的数据组。

用于链接（TRACON）主主轴柱面转换（TRACYL）与斜置轴（TRAANG）的数据组：

MD24450 $MC_TRAFO_TYPE_7	
= 8192	通道 TRACON 中转换 7 的类型。

MD24454 $MC_TRAFO_GEOAX_ASSIGN_TAB_7[ ]	用于转换 7 的几何轴与通道轴的分配。
[0] = 1	第 1 轴 = 通道轴 XC
[1] = 3	第 2 轴 = 通道轴 YC
[2] = 2	第 3 轴 = 通道轴 ZC

MD24996 $MC_TRACON_CHAIN_2[ ]	转换链接
[0] = 3	用于链接的转换编号 TRACYL （主主轴）。
[1] = 5	用于链接的转换编号 TRAANG。

19.5.11 当前程序的转速限值

在零件程序中编程转速限值。使用了零件程序时，转速限值会纳入设定数据并在程序结束后仍然保留。

如果程序结束后所修改的转速限值不再适用，则可通过机床制造商 GUD 模块中的以下指令修改特性：

仅能修改当前运行的程序的转速限值

```
REDEF $SA_SPIND_MAX_VELO_LIMS PRLOC    ; V 常量的最大转速限值 (G96)
REDEF $SA_SPIND_MAX_VELO_G26 PRLOC     ; rpm 的最大转速限值 (G26)
```

更多信息

有关转速限制的更多信息，请参见

NC 编程手册。

19.6 磨削

常规

磨具的刀具类型为 400、410，校正刀具的刀具类型为 490。“新刀具”功能提供了这些类型选项。刀具列表中列出了刀具类型“砂轮”和“校正刀具”，按刀具类型编排。为生成砂轮补偿，设置人员会创建一个校正程序。根据复杂程度，可访问外部或内部代码发生器。之后，该程序被分配给一个砂轮。

往复循环

这些磨削循环可以通过磨削机床上的两根几何轴加工工件。

由于通过循环只能生成在每台机床上都可任意使用的摆动运动，因此，不设定机床类型。

前提是有适用于程序加工的 SINUMERIK 控制系统及快速输入和输出。

循环软件包包含以下循环：

- CYCLE4071:反向点处带进给的纵向磨削
- CYCLE4072:反向点处带进给以及测量控制器的纵向磨削
- CYCLE4073:带连续进给的纵向磨削
- CYCLE4074:带连续进给以及测量控制器的纵向磨削
- CYCLE4075:反向点处带进给的平面磨削
- CYCLE4077:平面磨削：带测量控制器的切槽磨削
- CYCLE4078:带连续进给的平面磨削
- CYCLE4079:带间歇进给的平面磨削

往复循环：激活同步动作 GUD 变量

MD18661 $MN_MM_NUM_SYNACT_GUD_INT[n]	可定义的 INT 型全局用户数据变量的数量
> 0	使用该机床数据可使各个 GUD 数据块额外增加整数型通道专用参数区。 [n] 对应于数据块访问权限

更多信息

关于摆动循环或同步动作 GUD 变量的更多信息参见下列文档：

- NC 编程手册
- 功能手册之同步动作

磨削功能屏幕

MD51840 $MNS_GRIND_FUNCTION_MASK	磨削循环功能屏幕
位 0	修整器/砂轮采集时的参考点为零点偏移
= 0	撤销作为参考点的零点偏移
= 1	选择作为参考点的零点偏移

MD52840 $MCS_FUNCTION_MASK	磨削循环功能屏幕
位 7	成型磨削使用 G1 退回
= 0	成型磨削使用快速移动速度（G0）退回
= 1	成型磨削使用进给率（G1）退回

激活和显示底座相关补偿

激活底座相关补偿：

MD18603 $MN_MM_NUM_GLOBAL_G_FRAMES	全局磨削框架的数量 (SRAM)
该值为预定义数组 $P_GFR[] 中数组元素的数量。 如果该数据的值 > 0，则所有可设定的框架都为全局框架。MD28079 则可忽略。	

MD28079 $MC_MM_NUM_G_FRAMES	磨削框架的数量 (SRAM)
确定预定义磨削框架的数量。	

在操作区域“参数”→“零点偏移”窗口中显示底座相关补偿：

MD52211 $MCS_FUNCTION_MASK_DISP_ZOA		功能屏幕“显示零点偏移一览”
位 22	显示底座相关补偿 ($P_GFRAME)	

给定砂轮尺寸

MD52842 $MCS_GRIND_DIAMETER_LENGTH	砂轮直径的刀长编号
= 1	长度 1 是砂轮直径
= 2	长度 2 是砂轮直径

MD52843 $MCS_GRIND_WIDTH_LENGTH	砂轮宽度的刀长编号
= 1	长度 1 是砂轮宽度
= 2	长度 2 是砂轮宽度

成型磨削（CYCLE495）

SD55880 $SCS_GRIND_CONT_RELEASE_ANGLE	回退角度
	该设定数据确定了成型磨削中砂轮从轮廓退出的角度。

SD55881 $SCS_GRIND_CONT_RELEASE_DIST	回退量
	该设定数据确定了成型磨削中砂轮在两根轴上的退回量。

SD55884 $SCS_GRIND_CONT_BLANK_OFFSET	毛坯余量
	该设定数据确定了成型磨削中砂轮从 G0 变为 G1 的位置点（间距）。

19.7 回转

19.7.1 用于回转的工艺循环

前提条件

为了确保回转功能 (CYCLE800) 正确运作，必须进行机床运动链的调试。机床运动矢量在刀具参数 $TC_CARR1 到 $TC_CARR65 中定义。机床运动在运动链中定义，通过刀具参数 $TC_CARR_KIN... 访问。也可以不定义运动链，而是在刀具参数 $TC_CARR1 到 $TC_CARR65 中定义机床运动矢量。

说明

机床运动矢量可以通过测量功能“测量运动”（CYCLE9960）加以补偿。

要激活回转功能，至少要在 NCK 中激活一个可定向刀架（回转数据组）以及以工件、刀具和旋转工作台为参考的系统框架：

MD18088 $MN_MM_NUM_TOOL_CARRIER	最大可定义刀架数量
> 0	

MD28082 $MC_MM_SYSTEM_FRAME_MASK	系统框架（SRAM）
= 7DH	
位 2 = 1	TCARR 和 PAROT
位 3 = 1	TOROT 和 TOFRAME
位 4 = 1	工件参考点
位 5 = 1	循环系统框架
位 6 = 1	转换

说明

修改机床数据 MD18088 和 MD28082 会导致缓冲存储器重组。
在修改机床数据后必须创建批量调试文件并读入，否则许算出现数据丢失。

修改机床数据需要下列权限：访问级别 1（制造商）。

更多信息

更多信息请见：

- 功能手册之刀具
- 编程手册之测量循环

激活回转功能

在操作界面上可以通过下列通道专用机床数据使能回转功能：

MD52212 $MCS_FUNCTION_MASK_TECH	工艺通用的功能屏幕
位 0 = 1	使能回转

配置输入屏幕

可以通过下列通道专用循环设定数据来配置回转的输入屏幕。设定数据对所有相关的回效数据组有效。

SD55221 $SCS_FUNKTION_MASK_SWIVEL_SET	回转功能屏幕 CYCLE800
位 0	输入栏“回转否”
= 0	隐藏
= 1	显示
位 1	回退时的选择文本“Z, ZY”或“固定位置 1/2”
= 0	显示文本 Z = “Z”， 显示文本 Z, XY = “Z,XY”
= 1	显示文本 Z = “固定点 1”， 显示文本 Z, XY = “固定点 2”。 如果通过制造商循环“CUST_800.SPF”修改了退回方案“Z”或“Z, XY”，则会显示中性文本“固定点 1”和“固定点 2”。
位 2	允许勾选“撤销”回转数据
= 0	如果不允许取消选择，在输入屏幕“回转”中不会显示选择栏“回转数据组” (TC) 。
= 1	允许取消 另参见回转数据组 $TC_CARR37 亿位
位 3	在“JOG 中回转”下显示当前有效的回转面。回转功能标记中的设置会针对所有的回转数据组生效。
位 4	运动初始设置（极位置）中的定位

= 0	机床运动极位置中输入值的分析
= 1	兼容性
位 5	绝对刀具定位或通过框架计算进行刀具定位
= 0	通过框架计算（G18 上为 TCOFRY）进行刀具定位
= 1	绝对刀具定位（TCOABS） 在车床的 B 轴运动链上，运动链初始位置中的刀具主轴可平行于 Z 或 X。因此，车/铣床上的“铣刀定位”功能需要位 5 = 1。
位 6	JOG 中回转下不提供回转模式“直接”
位 7	回转平面：运动初始设置中的方向选择
= 0	方向为 + 或 - 时，运行回转轴的两个方案
= 1	方向为 + 或 - 时，只运行回转轴的两个方案之一
位 8	作为偏移运行回转数据组中回转轴的零点偏移
= 0	只要注意 WCS 中回转数据组中回转轴的零点偏移（兼容性）
= 1	作为偏移运行回转数据组中回转轴的零点偏移 位 8 与 MD21186 一同设置。
位 9	回转数据组 TC 固定分配至刀具
= 0	显示回转数据组 TC 的编号
= 1	未显示回转数据组 TC 的编号 设置了回转磨削的工艺 MD52200 = 3 时，通过参数 SGUD _TC_GNO 将回转数据组分配至刀具。 示例： 机床制造商可在刀具切换程序中 L6 通过当前刀具 _TC_GNO = $P_TOOLNO 写入参数 _TC_GNO。这样一来，换刀时刀架也一同更新。
位 10	回转平面：显示输入栏“定位方向”
= 0	不显示输入栏“定位方向”
= 1	显示旋转工作台和“摆头+旋转工作台”组合上的输入栏“定位方向” 所选定位方向保存在 GUD _TC_POS_DIR 中且可在 CUST_800 中检测： _TC_POS_DIR = 0：自动（最短行程） _TC_POS_DIR = -1：方向 - _TC_POS_DIR = +1：方向 + 该功能必须由机床制造商执行。
位 11	AB 运动上定位两个方案
= 0	确定的回转角度上仅定位一个方案（兼容性）

= 1	定位两个由 NC 计算的方案 运动时应设置位 11 = 1，在运动的初始设置中，两个回转轴均不会围绕刀具轴旋转（Z 方向上带刀具轴的 AB 运动）
位 12	平面回转，对刀：参考回转轴位置的复位属性
= 0	参照回转数据组的回转轴终点位置的复位状态（兼容性） 在参数 $TC_CARR21/22 中写入线性轴名称。复位（TCOABS）时，安装带回转轴终点位置（$TC_CARR13/14）的刀架。
= 1	参照回转数据组的回转轴当前位置的复位状态 在参数 $TC_CARR21/22 中写入回转数据组的回转轴名称。复位（TCOABS）时，安装带回转轴实际值的刀架。
位 13	按极位置逐轴回转时扩展旋转检测激活
= 0	按极位置回转（兼容性）时无旋转检测，适用位 4 和 11 的设置
= 1	按极位置回转时进行扩展旋转检测

通过以下通道专用设定数据可以设置，当按下软键“初始设置”时输入屏幕“回转平面”中会处于哪种状态：

SD55410 $SCS_MILL_SWIVEL_ALARM_MASK	隐藏和显示循环报警
	显示循环报警 CYCLE800、CYCLE996、CYCLE9960
位 0	显示报警 62186：有效零点偏移 G54 ff. 和基准（基本参考）包括旋转
位 1	显示报警 62187：有效基准和基本参考（G500）包括旋转
位 2	显示报警 61148：当前车刀无法进行平面回转

SD55420 $SCS_MILL_SWIVEL_RESET_RETRACT	回转的初始设置：回退
	该设定数据可设置按下软键“基本设置”后屏幕“回转平面”中切换栏“回退”的状态：
	无变化
= 1	否
= 2	Z
= 3	Z XY
= 4	刀具方向最大回退量
= 5	刀具方向增量回退量

SD55421 $SCS_MILL_SWIVEL_RESET_TRACK	回转的初始设置：跟踪刀具
	该设定数据用于设置在按下软键“基本设置”后屏幕“平面回转”中“跟踪刀具”切换栏的状态：
= 0	无变化
= 1	不跟踪
= 2	跟踪

SD55422 $SCS_MILL_SWIVEL_RESET_MODE	回转的初始设置：回转模式
	该设定数据用于设置在按下软键“基本设置”后屏幕“平面回转”中“回转模式”切换栏的状态：
= 0	逐轴
= 1	直接

SD55423 $SCS_MILL_SWIVEL_RESET_SEQ_AXIS	回转的初始设置：轴顺序
	该设定数据用于设置在按下软键“基本设置”后屏幕“平面回转”中“轴顺序”切换栏的状态：
= 0	XYZ
= 1	XZY
= 2	YXZ
= 3	YZX
= 4	ZXY
= 5	ZYX

其他设置

如要使用回转功能，至少要像下面一样设置下列机床数据，使它不同于缺省值：

MD10602 $MN_FRAME_GEOAX_CHANGE_MODE	
= 1	在切换几何轴时（选定/取消 TRAORI）会重新计算当前有效的总框架（零点偏移）。

MD10760 $MN_G53_TOOLCORR	G53、G153 和 SUPA 的生效方式
位 0 = 1	回转前何时使能机床轴，请参见章节“Herstellerzyklus CUST_800.SPF (页 527)”

MD11450 $MN_SEARCH_RUN_MODE	程序段搜索的设置
位 1 = 1	在程序段搜索之后激活 PROG_EVENT.SPF。这样在程序段搜索时可以对当前有效回转数据组的回转轴进行预定位。

定义回转数据组

为使用“回转”功能，必须在 NCK 中创建回转数据组，并在机床数据 MD18088 $MN_MM_NUM_TOOL_CARRIER 中加以定义。

如果在 NCU 中有多个通道，回转数据组的数量会按照 MD28085 $MC_MM_LINK_TOA_UNIT 划分。

MD28085 $MC_MM_LINK_TOA_UNIT	TO 单元至通道的分配 (SRAM)
设置了多个通道时，可在机床数据中进行以下设置： TO 单元（刀具和可定向刀架）分配至 NC 通道。可以将多个通道分配给一个 TO 单元。 -或者- 每个通道分配给一个 TO 单元。	

示例 1：

机床具有两个带不同 TO 单元的通道。

通道 1：MD28085 $MC_MM_LINK_TOA_UNIT= 1

通道 2：MD28085 $MC_MM_LINK_TOA_UNIT= 2

每个 TO 单元需要三个回转数据组（SDS）。

MD18088 = (TO 单元数) x (每个 TO 上的 SDS 数) = 2 x 3 = 6

示例 2：

机床具有三个带两种不同 TO 单元的通道。

通道 1：MD28085 $MC_MM_LINK_TOA_UNIT= 1

通道 2：MD28085 $MC_MM_LINK_TOA_UNIT= 2

通道 3：MD28085 $MC_MM_LINK_TOA_UNIT= 1

每个 TO 单元需要一个回转数据组（SDS）。

MD18088 = (TO 单元数) x (每个 TO 上的 SDS 数) = 2 x 1 = 2

使用角度刀具

在 NC 上使用刀具类型 130 来对角度刀具创建并管理角度刀具。在刀具类型 130 刀具中记录了刀具长度。如果需要在回转的加工平面上（CYCLE800）使用刀具，则必须通过 MD18114 $MM_ENABLE_TOOL_ORIENT = 2 激活刀具基本定向。这样便可以向刀具参数 $TC_DPV3[n] ~ $TC_DPV5[n] 传输一个方向矢量。在换刀时应编程指令 TOROT (G17)、TOROTY (G18) 和 TOROTX (G19)，用来更新刀具基本定向。

n = 内部刀具编号

MD18114 $MM_ENABLE_TOOL_ORIENT	指定刀具刀沿定向。 刀具基本定向
= 0	无有效的刀具基本定向
= 2	有效的刀具基本定向 在角度刀具和回转时使用

MD20110 $MC_RESET_MODE_MASK	复位与零件程序结束时的控制系统初始设置
位 14 = 1	计算基准框架与系统框架，示例：4041H。

MD20112 $MC_START_MODE_MASK	启动与零件程序的控制初始设置。
400H	

MD20126 $MC_TOOL_CARRIER_RESET_VALUE	复位时的有效刀架。
= 0	复位时无刀架生效。
= > 0 (n)	复位时编号为 n 的刀架生效。 在 CYCLE800 中描述 MD20126 CYCLE800() 与取消刀架选择相符 （MD20126 = 0）。

MD20150 $MC_GCODE_RESET_VALUES[ ]	G 功能组初始化设置
[41] = 1	TCOABS 上 G 功能组 42 的初始化设置
[51] = 2	PAROT 上 G 功能组 52 的初始化设置
[52] = 1 [52] = > 1	TOROTOF 上 G 功能组 53 的初始化设置 TOROT、 TOROTY 或 TOROTX 上 G 功能组 53 的初始化设置 机床运动类型为"T"和"M"时的应用。 参见参数 $TC_CARR34

说明

如需在 NC 复位后计算刀具方向的框架，可以将 MD20150 $MC_GCODE_RESET_VALUES[52] 的值设置为 > 1。

应用：

- 带端面齿的机床运动
- 带刀具基本方向的角度刀具

对带端面齿运动（摆头/"摆头+旋转工作台"混合运动）的说明：

以有效的平面（G17，G18，G19）为基准，在 NCU 中为了计算 CYCLE800 端面齿时补偿框架，编程指令 TOROT（或者 TOROTX、TOROTY）（G 功能组 53）。如果编程的旋转由于端面齿而与回转轴可能的位置相偏离，在摆头运动中和混合运动中会产生一个补偿框架 $P_TOOLFRAME（参见 HMI 有效的零偏/刀具基准详细信息）。

如果在复位或零件程序结束后需要保留补偿框架，则必须在通道专用机床数据中输入下列数值：

MD20150 $MC_GCODE_RESET_VALUES[52]	G 功能组的复位属性
= 2	在 G17 时 （TOROT）
= 3	
= 4	在 G18 时 （TOROTY）
= 5	在 G19 时（TOROTX）

MD20196 $MC_TOCARR_ROTAX_MODE	可定向刀架的回转轴模式
位 0 = 1	带一根回转轴的回转数据组 应用于带 C 轴的旋转工作台

位 1 = 1	带两根回转轴的回转数据组 应用于标准回转
位 2	
= 1	在旋转轴和坐标轴相互垂直的回转运动中，只有唯一的解法，即奇点位置，此时变量 $P_TCSOL 会输出 1 作为解法数量。
= 0	此类型是出于与旧软件版本的兼容性考虑的。输出两个解法，绕回转轴的角度相差 180 度。

MD20360 $MC_TOOL_PARAMETER_DEF_MASK	刀具参数的设置
位 10 = 1	在 T0 或 D0 时（无刀具），可定向刀架中的刀具仍保留。 机床运动类型为“T”和“M”时的应用。 参见参数 $TC_CARR34
位 19 = 1	在 T0 或 D0 时（无刀具），可定向刀架引起刀具定向变化仍保留。 位 19 仅在位 10 = 1 时有效。

MD21186 $MC_TOCARR_ROT_OFFSET_FROM_FR	由回转轴零偏得出的可定向刀架回转轴偏移。
= 0	在 CYCLE800 中为回转轴零偏（NPV）中的数值重新计算工件坐标系 WCS。 如果在当前激活的回转数据组中输入了回转轴的零点偏移，回转数据组的初始设置可能会发生变化，工件坐标系的初始设置也可能因此变化。
= 1	回转轴零偏中的数值被用作可定向刀架的偏移。工件坐标系 WCS 保持不变。

不允许在程序中通过调用 CYCLE800 改写 MD21186 。

MD21186 = 0 只适用于一些现有回转轴能够精确地围绕刀具轴旋转的回转数据组。因此，当在回转轴的零点偏移中输入了数值时，也能在 CYCLE800 中计算出有效的 WCS 旋转量。该设置适用于例如：运动初始设置为 G17、具有绕 X 轴旋转的回转轴 A 和绕 Z 轴旋转的回转轴 C 的旋转工作台。

如果每个通道有几个回转数据组，并且在更换摆头或者旋转工作台时要求触发一些机床功能，则可以在 PLC 程序中切换回转数据组时输出一个 M 指令。

MD22530 $MC_TOCARR_CHANGE_M_CODE	用于回转数据组切换的 M 代码
= 0	无回转数据组切换
< 0	M 代码 + 用于回转数据组切换的回转数据组的编号

示例

通道 1 中回转数据组的数量	= 2
MD22530 $MC_TOCARR_CHANGE_M_CODE	= -800
编程回转数据组 1 （TCARR=1）	= M801
编程回转数据组 2 （TCARR=2）	= M802

通过输出 M 指令，PLC 可以限制或逆转主轴转速或者夹紧或松开主轴。

设置工件、刀具和旋转工作台参考

使用下列机床数据可以设置系统框架的工件、刀具和旋转工作台参考或者可以控制系统框架的特性。

应用：复位或上电后系统框架仍有效，这样可以将例如钻头在不发生碰撞的情况下从回转位置退回。

MD24006 $MC_CHSFRAME_RESET_MASK	复位后有效的系统框架。
位 4	系统框架工件参考
= 0	无效
= 1	仍有效

MD24007 $MC_CHSFRAME_RESET_CLEAR_MASK	在复位后删除系统框架
位 4	系统框架工件参考
= 0	不删除
= 1	删除

在测量或 JOG 手动方式回转时，工件参考在复位时必须有效而不能删除（级联测量）。

MD24006 $MC_CHSFRAME_RESET_MASK	复位后有效的系统框架。
位 4 = 1	系统框架工件参考在复位后仍有效

MD24007 $MC_CHSFRAME_RESET_CLEAR_MASK	在复位后删除系统框架
位 2 = 0	预留，不删除 TCARR 和 PAROT
位 3 = 0	预留，不删除 TOROT 和 TOFRAME
位 4 = 0	系统框架工件参考在复位后不删除

MD24008 $MC_CHSFRAME_POWERON_MASK	系统框架在上电后重置。
位 2	系统框架旋转工作台参考（PAROT）
= 0	不重置
= 1	重置
位 3	系统框架刀具参考 （TOROT，..）
= 0	不重置
= 1	重置
位 4	系统框架工件参考
= 0	不重置
= 1	重置

MD24080 $MC_USER_FRAME_POWERON_MASK	用于可设定框架的设置。
位 0	
= 0	可设定零点偏移上电后无效。
=1	最后有效的可设定零点偏移在上电后仍有效，当 MD20152 $MC_GCODE_RESET_MODE[7] = 1 时。

应用：如果零点偏移为 G5xx，包括所有旋转在上电后都仍有效。

MD28082 $MC_MM_SYSTEM_FRAME_MASK	设置系统框架（SRAM）。
位 2 = 1	旋转台参考（$P_PARTFRAME）
位 3 = 1	刀具参考（$P_TOOLFRAME）
位 4 = 1	工件参考（$P_WPFRAME）

MD28083 $MC_MM_SYSTEM_DATAFRAME_MASK	设置系统框架（SRAM）数据管理
位 2 = 1	旋转台参考（$P_PARTFRAME）
位 3 = 1	刀具参考（$P_TOOLFRAME）
位 4 = 1	工件参考（$P_WPFRAME）

用于回转数据组模数回转轴的轴向机床数据

MD30455 $MA_MISC_FUNCTION_MASK	轴功能
位 0	模数回转轴编程

= 0	编程的位置必须位于模数范围内，否则会发出报警（如 0 到 359.999 度）。
= 1	在模数范围外编程位置不会发出报警。位置在内部转换为模数 （例如 -180 到 + 180 度）。
位 2	回转轴定位
= 0	和编程一样
= 1	按最短行程 应用：例如在 G90 带 DC 时回转轴 C 使用设置“位 2 = 1”按最短行程运行。附加信息请参见章节“Herstellerzyklus CUST_800.SPF (页 527)”。

MD32010 $MA_JOG_VELO_RAPID[AX] AX = 轴名称	JOG 方式下快速移动， 回转轴与加工轴，在 JOG 方式下回转时运行。
= 10000	用于 JOG 回转的 JOG 运行方式下快速移动

SD42980 $SC_TOFRAME_MODE	TOROT、PAROT 时的框架定义设置
= 2000	回转（缺省值）

SD42974 $SC_TOCARR_FINE_CORRECTION	精偏移 TCARR（回转数据组）。
= 0	回转数据组的矢量没有精偏移。
= 1	回转数据组的矢量有精偏移。 回转数据组的参数从 $TC_CARR41[n] 起生效，n...回转数据组的编号。

在 JOG 运行方式下回转

通过以下循环机床数据可以显示或隐藏循环报警 62186 和 62187：

MD55410 $MC_MILL_SWIVEL_ALARM_MASK	激活故障分析 CYCLE800
位 0	激活故障 61186
= 0	隐藏故障 61186“有效零点偏移 G%4 和基准（基本参考）包括旋转”（缺省设置）。
= 1	显示故障 61186
位 1	激活故障 61187

= 0	隐藏故障 61187 “有效基准和基本参考（G500）包括旋转”（缺省设置）。
= 1	显示故障 61187

19.7.2 用于机床运动识别的检查列表 CYCLE800

说明

机床运动的命名符合 DIN 66217 或 ISO 841-2001。
下面的检查列表只是示例，并没有考虑所有情况。

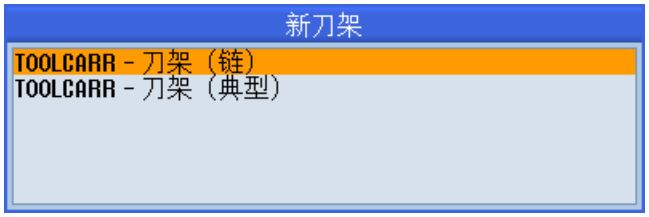
- **坐标转换生效的机床上，3 根线性轴是否构成一个直角坐标系？**几何轴 X 轴、Y 轴、Z 轴
- **机床有多少回转运动？**
  始终由 2 根（或 1 根）回转轴与 3 根线性轴构成。
- **涉及哪些机床组件？**
  摆头、旋转工作台，或者“摆头+旋转工作台”的组合。
- **运动的回转轴叫什么名称？**
  手动回转轴不需要也不一定要在 NC 中定义。
- **什么是回转数据组的第 1 或第 2 回转轴？**
- **线性轴和回转轴的运行方向正确吗？**右手规则
  规则：如线性轴或回转轴使工件移动，那么轴的运动方向与回转轴矢量都要变化。
- **运动的初始位置是什么样的？**
  从初始位置得出刀具方向和平面 G17，G18， G19。
- **哪些回转轴绕着哪些坐标轴或机床轴旋转？**
  可以由此得出运动的回转轴矢量。
  示例 1：
  摆头运动，回转轴 2 绕 Y 轴旋转 → 回转轴矢量 V2xyz = 0,1,0
  示例 2：
  旋转工作台运动，回转轴 1 绕 X 轴旋转 → 回转轴矢量 V1xyz = -1,0,0

19.7.3 调试 Toolcarrier/回转数据组

定义 Toolcarrier/回转数据组

必须给每个摆头、旋转工作台或者每个“摆头+旋转工作台”的组合创建一个 Toolcarrier。

Toolcarrier 可以在多个通道中定义。Toolcarrier 可按照传统方式定义或通过运动链定义。



一个传统的 Toolcarrier 由参数 $TC_CARR1[n] 到 $TC_CARR65[n] 构成，
其中 n = Toolcarrier 的编号。

基于运动链的 Toolcarrier 的几何数据来自于运动链。另外还有一些参数 $TC_CARR_KIN...[n] 和$TC_CARRxx[n] 来定义 Toolcarrier。

当满足“用于回转的工艺循环 (页 471)”一章中指出的所有前提条件时，便可以在操作区“调试”→“NC”→“坐标转换”
的输入窗口中设置 Toolcarrier。

在该窗口中可以确定，当创建新的 Toolcarrier 时是以传统方式设置它，还是通过运动链设置它。

除了上述方式外，也可以通过 NC 编程的方式来确定。程序启动后 Toolcarrier 的参数立即生效。

操作区域“程序”→“其它”中的以下软键用于回转功能：

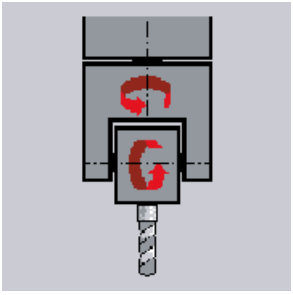
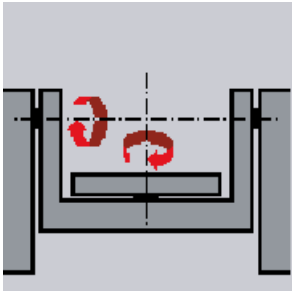
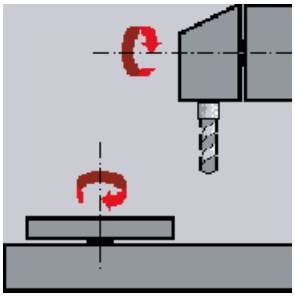
铣削工艺	**车削工艺**	
“平面回转”	“平面回转”	
“铣刀回转”	“刀具回转”	→“车刀定位”
		→“铣刀定位”
		→“铣刀回转”

仅在通过 $TC_CARR37[n] 激活“B 轴运动”时，才会显示“铣刀定位”和“车刀定位”软键。

更多信息

有关设置运动链的详细信息，
请见“功能手册之基本功能”。

运动类型 $TC_CARR23[n]

摆头（T 型）	旋转工作台（P 型）	摆头+ 旋转工作台（M 型）
		
偏移矢量 I1	偏移矢量 I2	偏移矢量 I1
回转轴矢量 V1	回转轴矢量 V1	回转轴矢量 V1
偏移矢量 I2	偏移矢量 I3	偏移矢量 I2
回转轴矢量 V2	回转轴矢量 V2	偏移矢量 I3
偏移矢量 I3	偏移矢量 I4	回转轴矢量 V2
		偏移矢量 I4

回转数据组名称

如果每个 NC 通道定义有多个回转数据组，则为每个回转数据组分配一个名称：$TC_CARR34[n]

回转数据组名称中仅可以使用 NC 编程所允许的字符：A...Z， 0...9 和 _ ！

设置了多个 NC 通道时，要注意 MD28085 中的设置。如果一个 TO 单元分配有多个 NC 通道，则与 TO 单元对话框中的通道显示相符。

更多说明参见 “用于回转的工艺循环 (页 471)”一章。

系统变量 $TC_CARR37[n]

回转对话框的输入栏/下拉栏提供以下几种数值显示选项：

小数位的含义

$TC_CARR37[n]	CYCLE800 输入屏幕显示方案	
小数位	含义	
个位	选择回转模式	
	0 =	逐轴
	1 =	逐轴 + 投影角度
	2 =	逐轴 + 投影角度 + 立体角度
	3 =	逐轴 + 直接
	4 =	逐轴 + 投影角度 + 直接
	5 =	逐轴 + 投影角度 + 立体角度 + 直接
十位	回转轴 1	
	0 =	自动
	1 =	手动
	2 =	半自动
百位	回转轴 2	
	0 =	自动
	1 =	手动
	2 =	半自动
千位	方向选择栏：回转轴的方向参考	
	0 =	不显示仅有一个方案的运动的方向参考。方向选择（_DIR）“负”在循环调用 CYCLE800 中生成。
	3 =	回转轴 1 方向参考，方向选择“负”在运动的初始设置中。
	4 =	回转轴 2 方向参考，方向选择“负”在运动的初始设置中。
	5 =	不显示仅有一个方案的运动的方向参考。方向选择（_DIR）“正”在循环调用 CYCLE800 中生成。
	8 =	回转轴 1 方向参考，方向选择“正”在运动的初始设置中。
	9 =	回转轴 2 方向参考，方向选择“正”在运动的初始设置中。
		不允许值 1, 2, 6 和 7。
万位	跟踪刀尖或 B 轴运动的选择栏	
	0 =	没有用于跟踪刀尖的输入栏显示。
	1 =	借助 TRAORI 跟踪刀尖。
	2 =	没有跟踪刀尖 + B 轴运动车削工艺。

$TC_CARR37[n]	**CYCLE800 输入屏幕显示方案**		
小数位	**含义**		
十万位	运动初始设置 [1)]		
	0 =	与工作平面 G17、G18 或 G19 相符（兼容性）	
	1 =	刀具轴在"-Z"方向上	
	3 =	刀具轴在"-X"方向上	
百万 千万	回退选择栏		
	00 =	无回退	
	01 =	沿 Z 轴回退	
	02 =	沿 Z 轴回退，然后沿 X 轴 Y 轴回退	
	03 =	沿 Z 轴回退或沿 Z 轴回退，然后沿 X 轴 Y 轴回退	
	04 =	刀具方向最大回退量	
	...		
	08 =	刀具方向增量回退量	
	...		
	15 =	沿 Z 轴回退或沿 Z 轴回退，然后沿 X 轴 Y 轴回退或 刀具方向最大回退量或者刀具方向增量回退量	
		$TC_CARR38[n]	X 轴回退位置
		$TC_CARR39[n]	Y 轴回退位置
		$TC_CARR40[n]	Z 轴回退位置
亿	使能回转数据组 设置回转数据组的切换 只有在 ShopMill 或 ShopTurn 中需要手动或自动切换刀具（→ 另见：CUST_800.spf，标志_M2 到_M13）。 无论如何都必须使能回转数据组，即设为大于等于 4 的值。		
	0 =	不激活 Toolcarrier	
	4 =	激活 Toolcarrier 自动执行回转数据组切换，自动换刀	
	5 =	激活 Toolcarrier 自动执行回转数据组切换，手动换刀	

$TC_CARR37[n]	CYCLE800 输入屏幕显示方案	
小数位	含义	
	6 =	激活 Toolcarrier 手动执行回转数据组切换，自动换刀
	7 =	激活 Toolcarrier 手动执行回转数据组切换，手动换刀

运动初始位置的设置说明 [1)]

在带有 B 轴的车床（第 1 工艺为车削工艺）上，可以确定在机床运动的初始位置中刀具是否已在"-Z"或"-X"方向上。在"刀具定位"和"平面回转"中要注意该设置。

刀具定位

- 刀具定位时，Beta 和 Gamma 编程与机床无关。
- 初始位置为"工作平面"或"-Z"时，Beta = 0 也就相当于该初始位置（兼容性）。
- 初始位置为"-X"时，Beta = 0 时，B 轴旋转 90°。

平面回转（新的回转平面）

- 初始位置为"工作平面"或"-Z"和当前工作平面 G17 时，在"平面回转"初始位置中不会产生新的刀具定向（兼容性）。
- 初始位置为"-X"和当前工作平面 G17 时，在"平面回转"初始位置中刀具会围绕 Y 轴旋转 90°。

因此，当初始位置为"-X"时，位于回转后平面上的铣刀就可以在不同的工作平面（G17 或 G19）上。

不允许进行以下设置和编程，否则会导致错误信息：

- 平面回转初始位置 -Z 和 G18 或 G19 有效
- 平面回转初始位置 -X 和 G18 有效

回转之前几何轴的回退



图 19-4　对话框：选择传统 Toolcarrier 的回退方式

在输入栏中选择回退方式。该方式在系统变量 $TC_CARR37[n] 的百万位和千万位中确定：

- 沿 Z 轴回退
- 先沿 Z 轴回退，然后沿 X 轴、Y 轴回退
- 刀具方向最大回退量或刀具方向增量回退量

注意
碰撞监测 在运行刀具轴时要注意，避开回转时刀具和工件碰撞。

“沿 Z 轴回退”或者“沿 Z 轴回退，然后沿 X 轴 Y 轴回退”是回退到参数 $TC_CARR38[n] 到 $TC_CARR40[n] 中定义的数值（绝对机床位置）。

$TC_CARR38[n]	X 轴回退位置
$TC_CARR39[n]	Y 轴回退位置
$TC_CARR40[n]	Z 轴回退位置

在 Herstellerzyklus CUST_800.SPF (页 527) 中修改回退类型。

说明

在一些刀具可以摆动的机床上，比如：带摆头或“摆头+工作台”混合运动结构的机床上，需要在回转数据组中考虑刀具方向上的回退，特别是带 B 轴的车床。选择“刀具方向最大回退量”时，刀具会一直回退，直到其中一根线性轴到达软限位。

19.7.4 调试基于运动链的 Toolcarrier

必须给每个摆头、旋转工作台或者每个“摆头+旋转工作”的组合创建一个 Toolcarrier。

前提条件

机床上已经有一个运动链或者创建一个运动链。

更多信息

更多信息参见“
功能手册之刀具”。

创建 Toolcarrier

在操作区“调试”→ “NC” → “坐标转换”的输入窗口
中创建 Toolcarrier。

在“新建 Toolcarrier”对话框中可以选择基于运动链的 Toolcarrier 或传统 Toolcarrier。

关于选择的**更多说明**参见“调试 Toolcarrier/回转数据组 (页 483)”一章。



图 19-5　对话框：基于运动链的 Toolcarrier

使用 Toolcarrier 调试屏幕可以设置以下刀具数据：

变量名	含义
$TC_CARR_KIN_CNTRL[n]	确定是否采用来自运动链单元的 Toolcarrier 的几何数据 位 0：Toolcarrier 几何数据通过运动链获得 位 1：闭合工件链（计算 I4） 位 2：闭合刀具链（计算 I1）
$TC_CARR_KIN_TOOL_START[n]	运动链元素的名称，它构成刀具链的起点
$TC_CARR_KIN_TOOL_END[n]	运动链元素的名称，它构成刀具链的终点
$TC_CARR_KIN_PART_START[n]	运动链元素的名称，它构成刀具链的起点
$TC_CARR_KIN_PART_END[n]	运动链元素的名称，它构成工件链的终点
$TC_CARR_CORRELEM[n, m] n = Toolcarrier 数据组的编号 0 ≤ m ≤ 3	运动链元素的名称，CYCLE9960 对它进行补偿
$TC_CARR_KIN_ROTAX_NAME[n, m]	运动链元素的字符串格式名称，它包含了回转轴

变量名	含义
$TC_CARR23[n]	运动形式：摆头（T）、旋转工作台（P）、混合式（M）
$TC_CARR26[n] $TC_CARR27[n]	回转轴的端面齿偏移
$TC_CARR28[n] $TC_CARR29[n]	回转轴的端面齿增量
$TC_CARR30[n] $TC_CARR31[n]	回转轴最小位置软限位
$TC_CARR32[n] $TC_CARR33[n]	回转轴最大位置软限位
$TC_CARR34[n]	Toolcarrier 名称
$TC_CARR37[n]	
$TC_CARR38[n] $TC_CARR39[n] $TC_CARR40[n]	回退位置

回转轴（旋转轴）

按下软键“选择元素”，打开机床画面的输入屏幕。在其中选择包含回转轴的运动链元素。

包含回转轴的运动链元素的名称位于变量 $TC_CARR_KIN_ROTAX_NAME[n, m] 中。

补偿元素

在通过 CYCLE9960 测量并补偿机床运动时，变量 $TC_CARR_CORRELEM[n, m] 可参照包含运动补偿的运动链元素。

19.7.5 调试传统的 Toolcarrier 回转数据

回转数据组名称

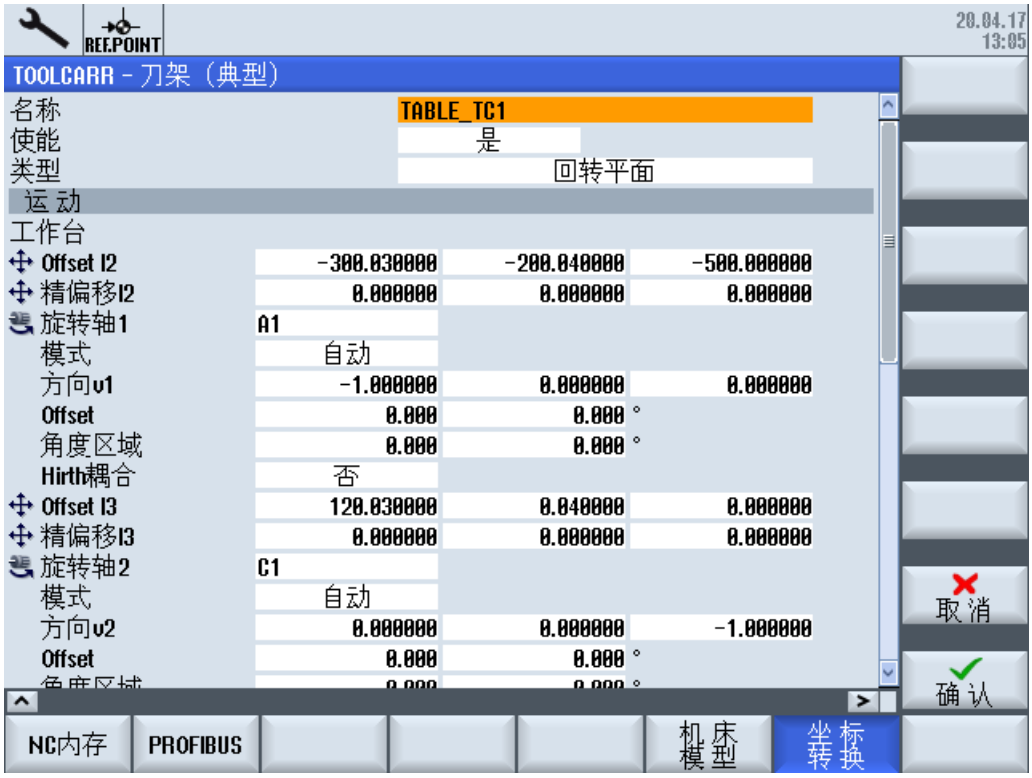


图 19-6 用于输入回转数据组参数的对话框

偏移 I1 到 I4 $TC_CARR1[n] ... $TC_CARR20[n]

这些矢量始终包含三个部分，它们是机床轴（X, Y, Z）的基准。运动链的位置由机床制造商测量，它们始终和摆头或旋转工作台（Toolcarrier）相关。

偏移 I1 到 I4 针对的是回转轴尚未回转的状态（机床运动初始位置有效）。必须注意的是，回转平面中的运行区域可能受限。如果机床运动只有一根回转轴，则必须将它定义为第 1 回转轴。

可以手调的回转轴（手动模式）可以带/不带测量系统，在“简单机床”中可以使用。

通过附加机械定位的回转轴能在“半自动”模式中进行定义。

$TC_CARR1[n]	$TC_CARR2[n]	$TC_CARR3[n]	偏移 I1 xyz
$TC_CARR4[n]	$TC_CARR5[n]	$TC_CARR6[n]	偏移 I2 xyz

$TC_CARR15[n]	$TC_CARR16[n]	$TC_CARR17[n]	偏移 I3 xyz
$TC_CARR18[n]	$TC_CARR19[n]	$TC_CARR20[n]	偏移 I4 xyz

示例

钻头上的摆头通过主轴耦合进行定位。该情况下，摆头的回转轴必须指定为“半自动”模式。在 CUST_800.SPF 中调用相应的机床专用的循环。

机床制造商

请注意机床制造商的说明。

摆头

- I3：从刀柄到第 2 个回转轴的旋转中心/交点的距离
- I2 从回转轴 2 的旋转中心/交点到回转轴 1 的旋转中心/交点的距离
- I1 如果摆头不可更换，闭合矢量链 I1=-(I2+I3)

旋转工作台

- I2 从机床参考点到第 1 回转轴的旋转中心/交点
- I3 从回转轴 1 的旋转中心/交点到回转轴 2 的旋转中心/交点的距离（或者到刀具安装的参考点）
- I4 如果摆头不可更换，闭合矢量链 I4=-(I2+I3)

摆头+旋转工作台（混合运动）

- I2：从刀柄到第 1 个回转轴的旋转中心/交点的距离
- I1 如果摆头不可更换，闭合矢量链 I1=-I2
- I3 从机床参考点到第 2 个回转轴的旋转中心/交点的距离（或者到刀具安装参考点）
- I4 如果摆头不可更换，闭合矢量链 I4=-I3。

不必强制这些偏移矢量指向回转轴的旋转中心。重要的是，这些矢量指向旋转方向的一个点（交点）。

偏移矢量(I1...I4)和回转轴矢量(V1, V2)的符号按照 **ISO 841-2001** 或者 **DIN 66217**（右手规则）规定的轴方向定义得出。在使工件运动（回转工作台）的机床运动中，轴的方向相反。

方向 V1 和 V2

$TC_CARR7[n]	$TC_CARR8[n]	$TC_CARR9[n]	第 1 回转轴的方向矢量
$TC_CARR10[n]	$TC_CARR11[n]	$TC_CARR12[n]	第 2 回转轴的方向矢量

更多信息

有关刀具补偿以及多次坐标转换的更多信息参见：

- 功能手册之刀具
- 功能手册之坐标转换

设置偏移

说明

TOOLCARRIER ⇔ 5 轴转换 (转换类型 24, 40, 56)的联系:

在 5 轴转换中可以使用“MD24100: $MC_TRAFO_TYPE_1”中定义的转换类型 72。

转换类型 72 使用“MD24582: $MC_TRAFO5_TCARR_NO_1”中定义的刀架矢量（TOOLCARRIER）。

摆头（可回转刀具）			
$TC_CARR23[n]="**T**"			MD24100: $MC_TRAFO_TYPE_1=**24**
I1	$TC_CARR1...3[n]		MD24500: $MC_TRAFO5_PART_OFFSET_1[0...2]
I2	$TC_CARR4...6[n]		MD24560: $MC_TRAFO5_JOINT_OFFSET_1[0...2]
I3	$TC_CARR15...17[n]		MD24550: $MC_TRAFO5_BASE_TOOL_1 [0...2]
闭合矢量链 **I1=-(I2+I3)**; 摆头固定安装时			

旋转工作台（可回转工件）			
$TC_CARR23[n]=**"P"**			MD24100: $MC_TRAFO_TYPE_1=**40**
I2	$TC_CARR4...6[n]		MD24550: $MC_TRAFO5_BASE_TOOL_1 [0..2]
I3	$TC_CARR15...17[n]		MD24558: $MC_TRAFO5_JOINT_OFFSET_1 [0...2]
I4	$TC_CARR18...20[n]		MD24500: $MC_TRAFO5_PART_OFFSET_1[0...2]
闭合矢量链 **I4=-(I2+I3)**; 旋转工作台固定安装时			

摆头＋旋转工作台（可回转刀具＋工件）			
$TC_CARR23[n]="**M**"			MD24100: $MC_TRAFO_TYPE_1=**56**
I1	$TC_CARR1...3[n]		MD24560: $MC_TRAFO5_JOINT_OFFSET_1[0...2]
I2	$TC_CARR4...6[n]	⟷	MD24550: $MC_TRAFO5_BASE_TOOL_1 [0...2]
I3	$TC_CARR15...17[n]		MD24558: $MC_TRAFO5_JOINT_OFFSET_PART_1[0...2]
I4	$TC_CARR18...20[n]		MD24500: $MC_TRAFO5_PART_OFFSET_1[0...2]
闭合矢量链 **I1=-I2 I4=-I3**; 摆头＋旋转工作台固定安装时			

回转轴矢量 V1, V2			
V1	$TC_CARR7..9[n]	⟷	MD24570: $MC_TRAFO5_AXIS1_1[0..2]
V2	$TC_CARR10..12[n]		MD24572: $MC_TRAFO5_AXIS2_1[0..2]

偏移矢量的精偏移

偏移矢量的精偏移在参数 $TC_CARR41[n] 到 $TC_CARR60[n] 中定义。

分配基本矢量到精偏移矢量：

I1 $TC_CARR1..3[n]	到	$TC_CARR41..43[n]
I2 $TC_CARR4..6[n]	到	$TC_CARR44..46[n]
I3 $TC_CARR15..17[n]	到	$TC_CARR55..57[n]
I4 $TC_CARR18..20[n]	到	$TC_CARR58..60[n]

激活精偏移要通过设定数据：

SD42974 $SC_TOCARR_FINE_CORRECTION = 1	打开/关闭精偏移 TCARR
= 0	在激活可定位的刀架时不会考虑到精偏移值。
= 1	在激活可定位的刀架时会考虑到精偏移值。

在调用回转功能 CYCLE800 或 NC 功能 TCARR=n 时，精偏移会对相应的基本矢量起到附加作用。

旋转轴（回转轴）的名称

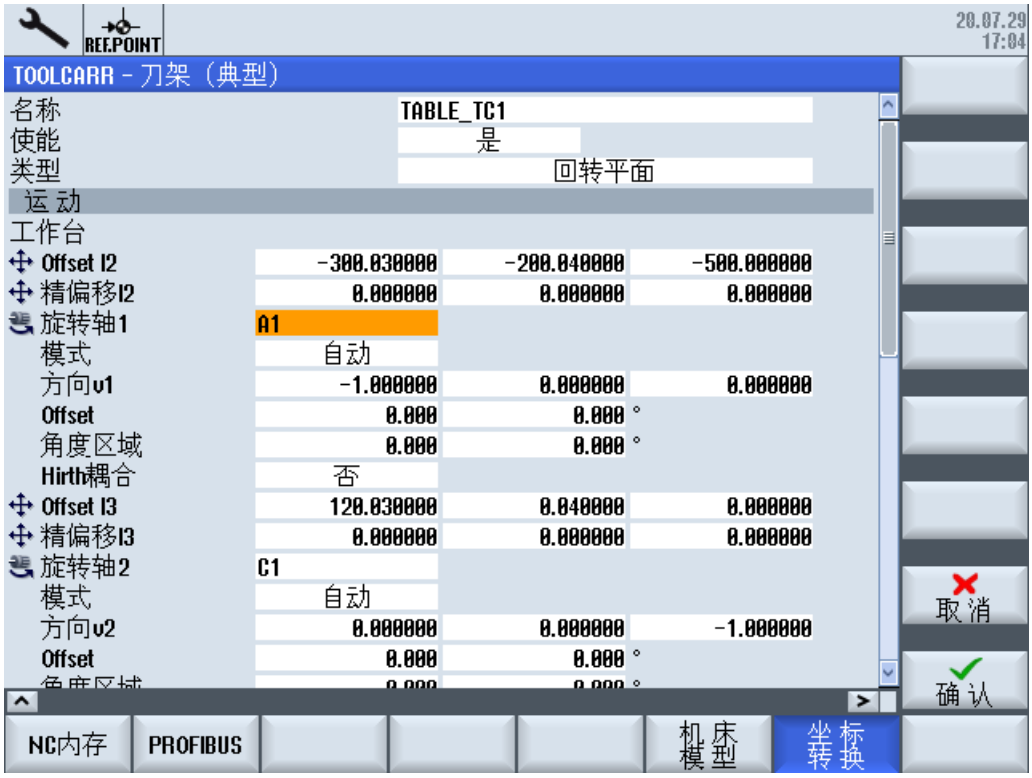


图 19-7　用于输入回转轴参数的对话框

优先选择以下的名称作为回转轴名称：

- 围绕机床轴 X 轴旋转的回转轴 → A 轴
- 围绕机床轴 Y 轴旋转的回转轴 → B 轴
- 围绕机床轴 Z 轴旋转的回转轴 → C 轴

自动回转轴上必须记录通道轴名称与相应的 NC 回转轴（参见 $TC_CARR37[n] 十位和百位：自动模式）。对于手动（手动调节）和半自动回转轴，可以使用任意轴名称（最多 6 个字符或数字）。

NC 中未定义手动和半自动回转轴。手动回转轴中，在对话框中输入角度范围值。半自动回转轴在机床上进行机械运动（通过耦合的主轴）。可以在制造商循环 CUST_800 中进行调整。无需给出角度范围值，回转轴就能进行自动定位。

机床制造商

请注意机床制造商的说明。

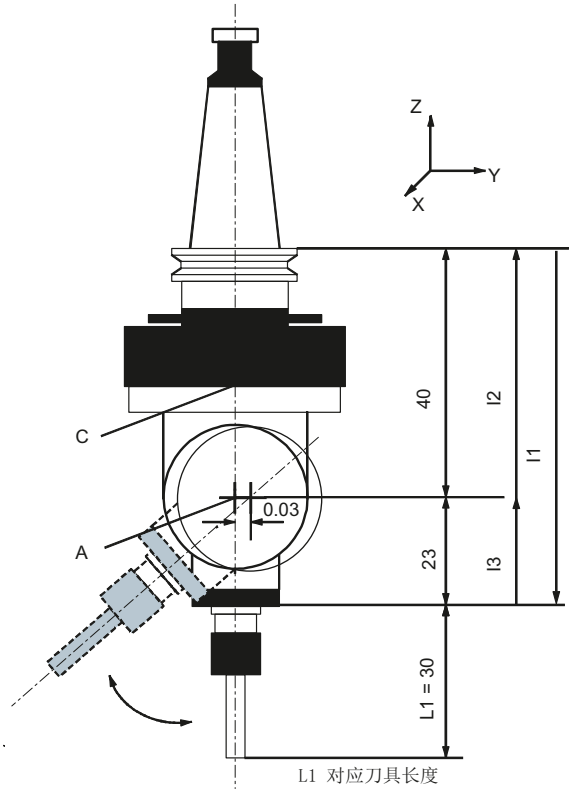
$TC_CARR35[n]	回转轴 1 名称
$TC_CARR36[n]	回转轴 2 名称

19.7.6 示例：调试传统 Toolcarrier

下面的示例适用于传统 Toolcarrier。对于基于运动链的 Toolcarrier 而言，必须相应地调试运动链。

示例 1：摆头 1“HEAD_1”

图纸中的矢量适用于机床运动的初始位置。图纸不按照比例。摆头可更换，可手调。



C 回转轴 1（手动）围绕 Z 轴旋转。

A 回转轴 2（手动）围绕 X 轴旋转。

图 19-8 可更换的锥形摆头，用于安装主轴



图 19-9　运动类型选择

名称	HEAD_1		
使能	是		
类型	回转平面		
偏移 I1	0.000000	0.030000	-63.000000
精偏移	0.000000	0.000000	0.000000
旋转轴 1	C		
模式	手动		
方向 v1	0.000000	0.000000	0.000000
偏移	0.000°		
角度范围	0.000°	360.000°	
端面齿	否		
偏移 I2	0.000000	0.000000	40.000000
精偏移	0.000000	0.000000	0.000000
旋转轴 2	A		
模式	手动		
方向 v2	1.000000	0.000000	0.000000
偏移	0.000°		
角度范围	-15.000°	100.000°	
端面齿	否		
偏移 I3	0.000000	-0.030000	-23.000000
精偏移	0.000000	0.000000	0.000000
属性			
回退选择			
• 机床轴 Z 轴			是
• 回退位置			200

选择回转模式			
• 逐轴			是
首选方向参考轴			旋转轴 2
跟踪选择			否

示例 2a：摆头 2"HEAD_2"：基于运动链的 Toolcarrier

我们建议在使用运动链时根据机床的上实际运动情况来图示机床运动结构。如左图所示，矢量的标注和运动结构图示符合机床实际情况。其中，Toolcarrier 的起点位于 B 轴旋转中心；然后它按"C_OFFSET"偏移运动，到达 C 轴的实际旋转中心，位置是"0,1,1"；最后它运动一段偏移，到达刀具参考点。但正如传统 Toolcarrier 上常见的，Toolcarrier 的起点也可以位于 C 轴旋转中心。当摆头固定安装在机床上，运动结构闭合时，两种图示方式的效果一样。当机床上有多个摆头，运动结构开放时，建议采用第一种运动图示方式。

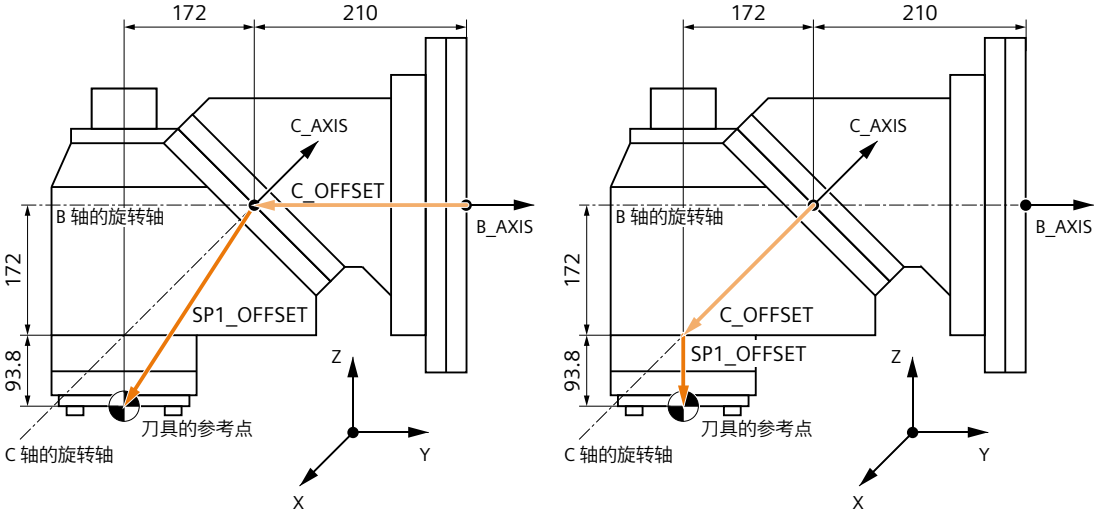
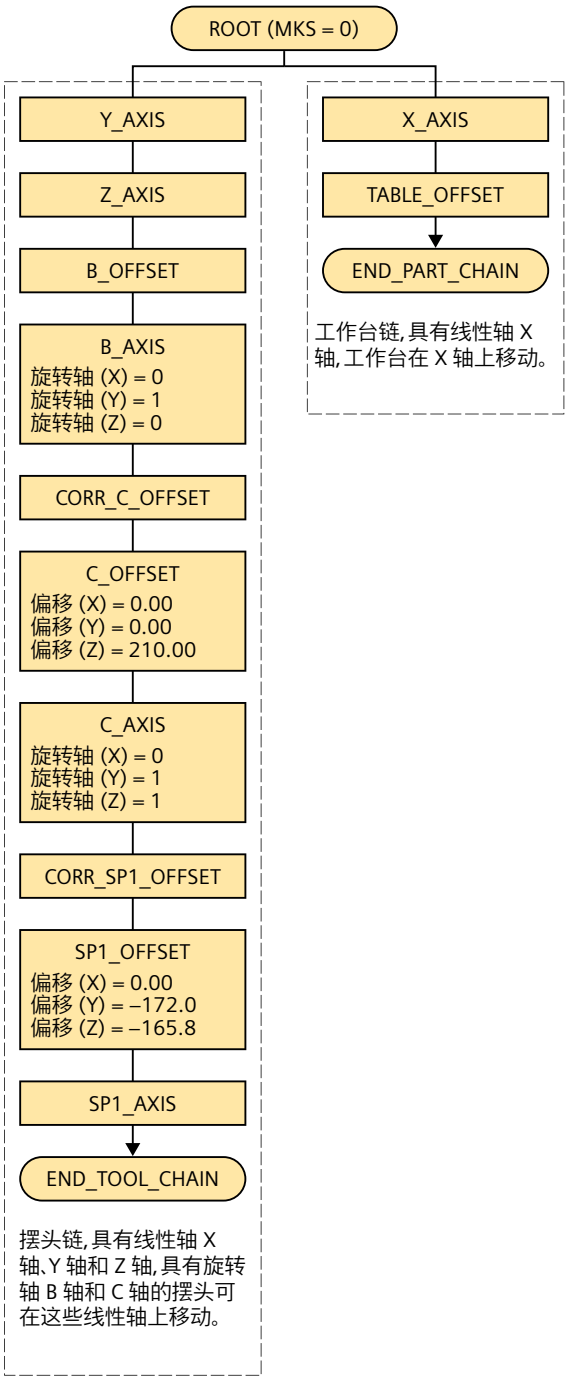


图 19-10　摆头图示方式：实际运动图示和偏移后的运动图示

运动链示意图：

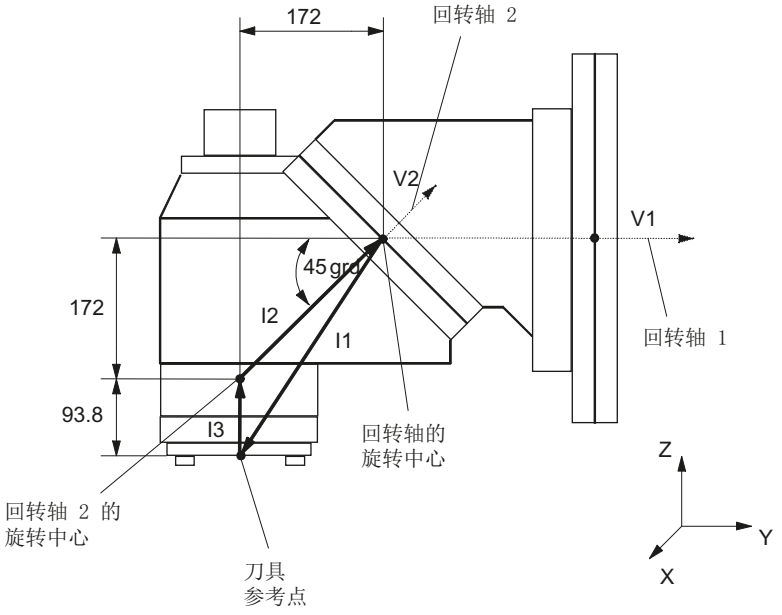


名称	HEAD_2		
使能	是		
类型	回转平面		
运动			

刀具链			
刀具链开端	Z_AXIS		
旋转轴	B_AXIS		
角度范围		0.000°	360.000°
端面齿	否		
补偿元素	CORR_C_OFFSET		
旋转轴	C_AXIS		
角度范围		-15.000°	100.000°
端面齿	否		
补偿元素	CORR_SP1_OFFSET		
刀具链终端	SP1_OFFSET		
闭合刀具链	是		
属性			
回退选择			
• 机床轴 Z 轴			是
• 回退位置			200
选择回转模式			
• 逐轴			是
首选方向参考轴			旋转轴 2
跟踪选择			否

示例 2b：摆头 2“HEAD_2”：传统 Toolcarrier

图纸中的矢量适用于机床运动的初始位置。摆头固定安装，表示矢量链闭合。回转轴 1 和 2 的旋转中心参考点可以在旋转中心线上移动，不必与机械旋转中心一致。



方向 v1　　回转轴 1 围绕 Y 轴旋转。

方向 v2　　回转轴 2 围绕 Y 轴和 Z 轴旋转。

偏移 I1　　摆头固定安装、I1 = - (I2 + I3) 时闭合矢量链。

偏移 I2　　回转轴 1 的旋转中心和回转轴 2 的旋转中心之间的距离。

偏移 I3　　刀具参考点和回转轴 2 的旋转中心之间的距离。

图 19-11　带端面齿的万向摆头，可手调

名称	HEAD_2		
使能	是		
类型	回转平面		
偏移 I1	0.000000	-172.000000	265.800000
精偏移 I1	0.000000	0.000000	0.000000
旋转轴 1	B		
模式	自动		
方向 v1	0.000000	1.000000	0.000000
偏移	0.000°		
角度范围	0.000°	360.000°	

端面齿	否		
偏移 I2	0.000000	172.000000	172.000000
精偏移	0.000000	0.000000	0.000000
旋转轴 2	C		
模式	自动		
方向 v2	0.000000	-1.000000 1)	1.000000 1)
偏移	0.000°		
角度范围	0.000°	180.000°	
端面齿	是	1°	
偏移 I3	0.000000	-0.030000	93.800000
精偏移	0.000000	0.000000	0.000000
属性			
回退选择			
• 刀具方向增量回退量			是
• 刀具方向最大回退量			是
• 机床轴 Z 轴			
• Z 轴回退位置			200
选择回转模式			
• 逐轴			是
首选方向参考轴			旋转轴 2
跟踪选择			否

1）计算回转轴矢量：

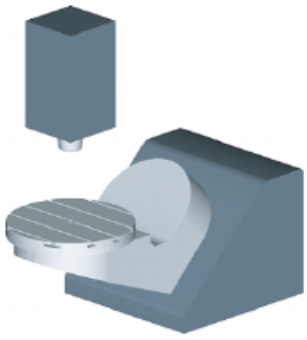
V2:45 度角

V2Y = sin(-45) = -0.7071

V2Z = cos(-45) = 0.7071

V2Y 和 V2Z 可以定标为 1。

示例 3a：万向工作台“TABLE_45”：基于运动链的 TCARR



图纸中的矢量适用于机床运动的初始位置。矢量 A_OFFSET 可在 A 轴的旋转轴上随意移动。

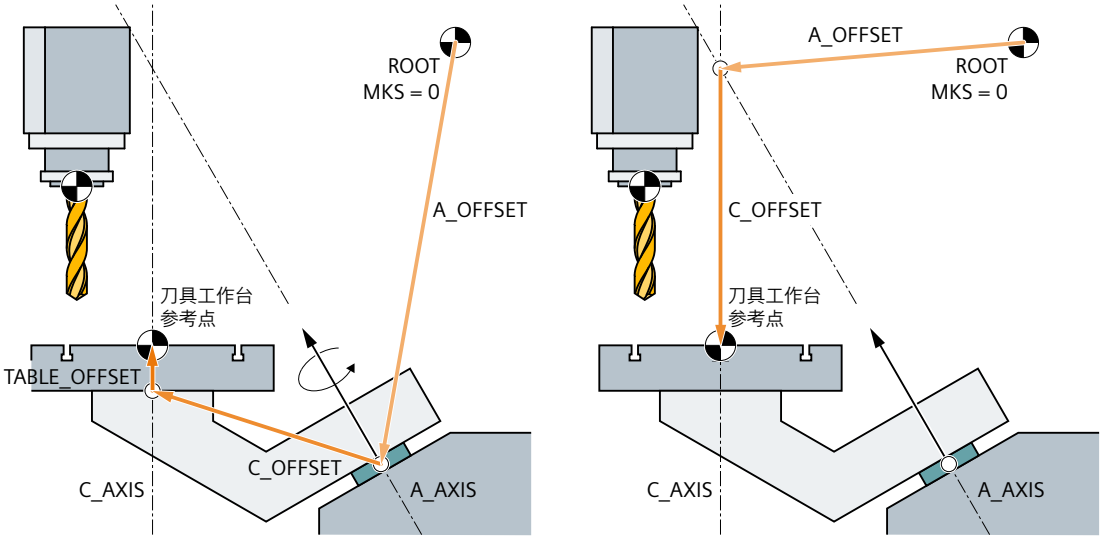


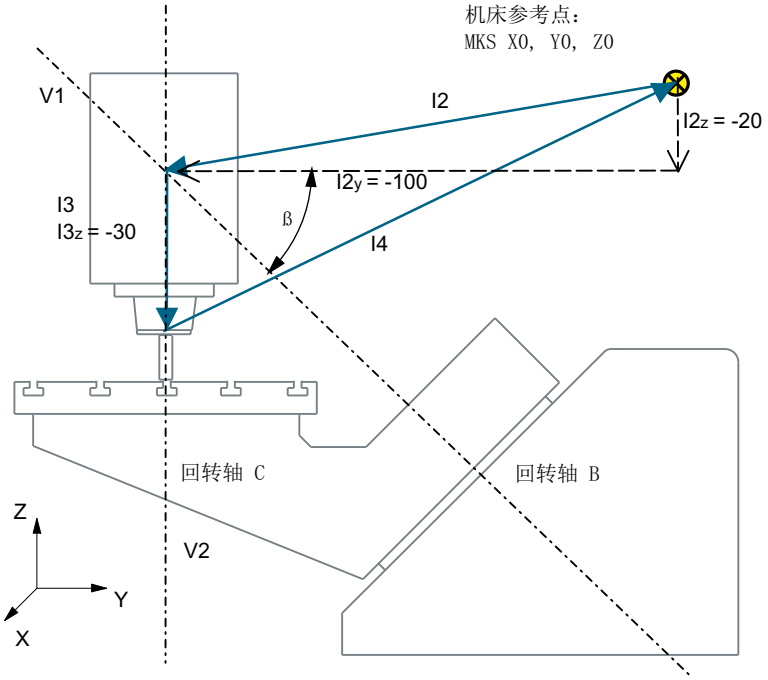
图 19-12　矢量偏移 A_OFFSET

名称	TABLE_45		
使能	是		
类型	回转平面		
运动			
刀具链			
刀具链开端	Z_AXIS		
	CORR_A_OFFSET		
旋转轴	A_AXIS		
角度范围		0.000°	360.000°
端面齿	否		

补偿元素	CORR_C_OFFSET		
旋转轴	C_AXIS		
角度范围		0.000°	360.000°
端面齿	否		
刀具链终端	OFFSET_TABLE		
闭合刀具链	是		
属性			
回退选择			
• 机床轴 Z 轴			是
• 回退位置			200
选择回转模式			
• 逐轴			是
首选方向参考轴			旋转轴 1
跟踪选择			否

示例 3b：万向工作台“TABLE_45”：传统 TCARR

图纸中的矢量适用于机床运动的初始位置。主轴（刀柄/刀架）已定位到工作台边缘上方或者工作台中心上方（回转轴 C）。旋转工作台 C 轴的旋转中心利用主轴上的测头确定。



方向 v1	回转轴 B 围绕 Y 轴和 Z 轴旋转。
方向 v2	回转轴 C 围绕 Z 轴旋转。
偏移 I2	从机床参考点到回转轴 1 的旋转中心/交点的距离。
偏移 I3	回转轴 1 的旋转中心/交点和回转轴 2 的旋转中心/交点之间的距离。
偏移 I4	闭合矢量链，I4 = - (I2 + I3)。

图 19-13　机床侧视图

名称	TABLE_45		
使能	是		
类型	回转平面		
运动			
工作台			
偏移 I2	0.000000	-100.000000	-20.000000
精偏移 I2	0.000000	0.000000	0.000000
旋转轴 1	B		

模式	自动		
方向 v1	0.000000	-1.000000 1)	1.000000 1)
偏移	0.000°		
角度范围	0.000°	180.000°	
端面齿	否		
偏移 I3	0.000000	0.000000	-30.000000
精偏 I3	0.000000	0.000000	0.000000
旋转轴 2	C		
模式	自动		
方向 v2	0.000000	0.000000	-1.000000
偏移	0.000°		
角度范围	0.000°	360.000°	
端面齿	否		
偏移 I4	-300.000000	100.000000	50.000000
精偏 I4	0.000000	0.000000	0.000000
属性			
选择回转模式			
• 逐轴			是
首选方向参考轴			旋转轴 1
跟踪选择			否

1）计算回转轴矢量：

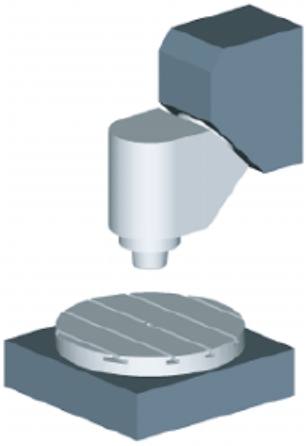
V1: β = -45 度

V1Y= sin(-45)= -0.7071

V1Z= cos(-45)= 0.7071

V1Y 和 V1Z 可以定标为 -1 和 1。

示例 4a：摆头+旋转工作台“MIXED_45”，基于运动链的 Toolcarrier



图纸中的矢量适用于机床运动的初始位置。B 轴和刀具参考点之间的矢量 SP1_OFFSET 可以在 B 轴的旋转中心轴上任意移动。

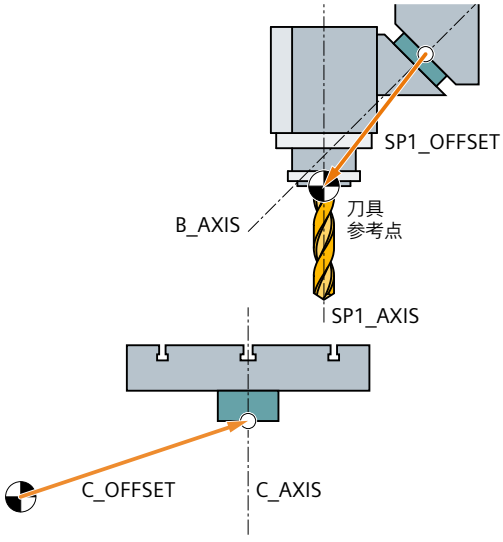


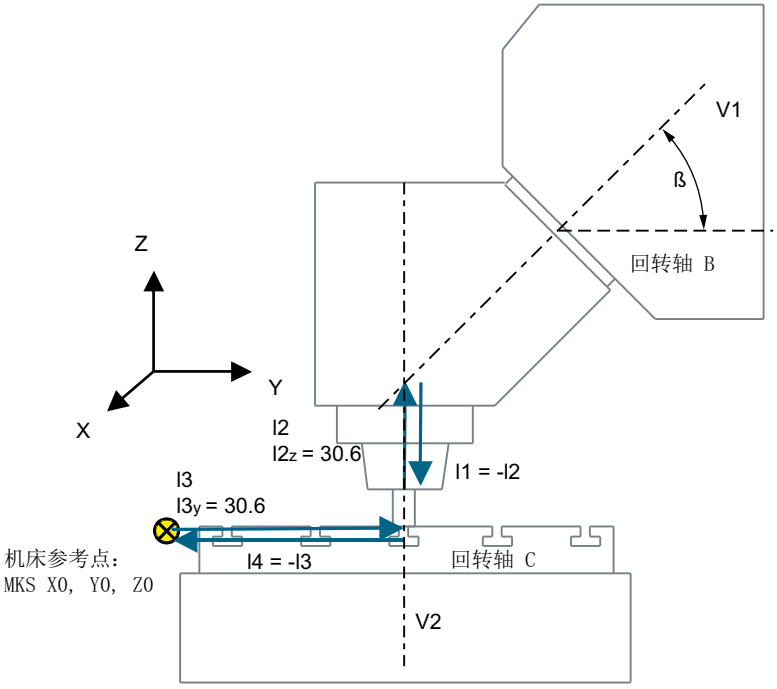
图 19-14　矢量偏移 SP1_OFFSET

名称	MIXED_45		
使能	是		
类型	回转平面		
运动			
刀具链			
刀具链开端	Z_AXIS		
旋转轴	B_AXIS		

角度范围		0.000°	180.000°
端面齿	否		
补偿元素	CORR_SP1_OFFSET		
刀具链终端	SP1_AXIS		
闭合刀具链	是		
工件链			
工件链开头			
补偿元素	CORR_C1_OFFSET		
旋转轴	C_AXIS		
角度范围		0.000°	360.000°
端面齿	否		
工件链终端	C_AXIS		
闭合刀具链	是		
属性			
回退选择			
• 机床轴 Z 轴			否
选择回转模式			
• 逐轴			是
首选方向选择			是，默认=+
首选方向参考轴			旋转轴 1
跟踪选择			否

示例 4b：摆头 + 旋转工作台“MIXED_45”，传统 Toolcarrier

图纸中的矢量适用于机床运动的初始位置。主轴（刀柄/刀架）已定位到工作台边缘上方或者工作台中心上方（回转轴 C）。回转台 C 轴的旋转中心利用主轴上的测头确定。



方向 v1	回转轴 B 围绕 Y 轴和 Z 轴旋转。
方向 v2	回转轴 C 围绕 Z 轴旋转。
偏移 I2	从刀柄参考点到回转轴 1 的旋转中心/交点的距离。
偏移 I1	闭合矢量链，I1 = - I2。
偏移 I3	从机床参考点到回转轴 2 的旋转中心/交点的距离。
偏移 I4	闭合矢量链，I4 = - I3。

图 19-15　机床侧视图

名称	MIXED_45		
使能	是		
类型	回转平面		
运动			
摆头			
偏移 I1	0.000000	0.000000	-30.000000
精偏移 I1	0.000000	0.000000	0.000000

旋转轴 1	B		
模式	自动		
方向 v1	0.000000	1.000000 1)	1.000000 1)
偏移	0.000°		
角度范围	0.000°	180.000°	
端面齿	否		
偏移 I2	0.000000	0.000000	30.000000
精偏移 I2	0.000000	0.000000	0.000000
工作台			
偏移 I3	300.000000	150.000000	0.000000
精偏 I3	0.000000	0.000000	0.000000
旋转轴 2	C		
模式	自动		
方向 v2	0.000000	0.000000	-1.000000
偏移	0.000°		
角度范围	0.000°	360.000°	
端面齿	否		
偏移 I4	-300.000000	-150.000000	0.000000
精偏 I4	0.000000	0.000000	0.000000

1）计算回转轴矢量：

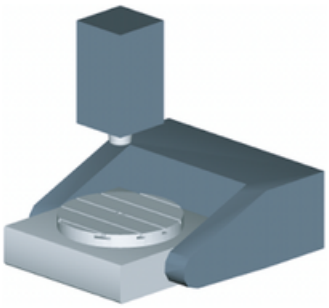
V1: β = 45 度

V1Y= sin(-45)= -0.7071

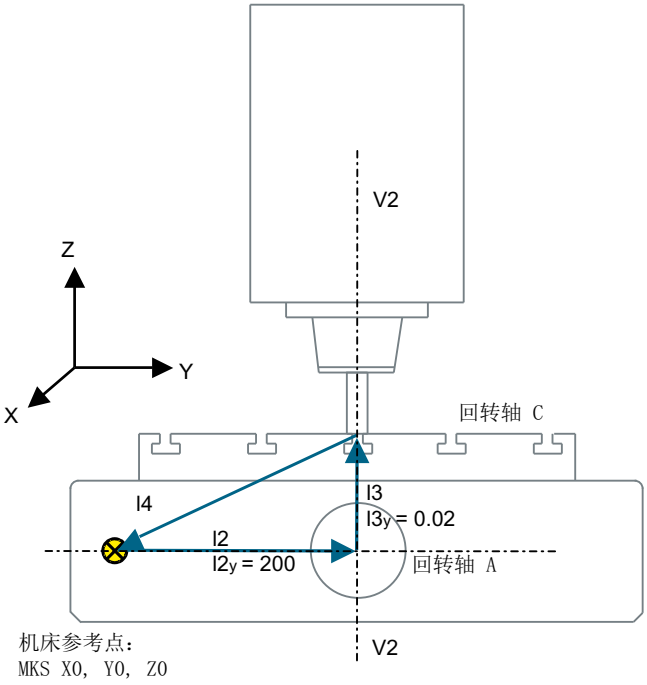
V1Z= cos(-45)= 0.7071

V1Y 和 V1Z 可以定标为 1。

示例 5：旋转工作台“TABLE_5”，传统 Toolkarrier

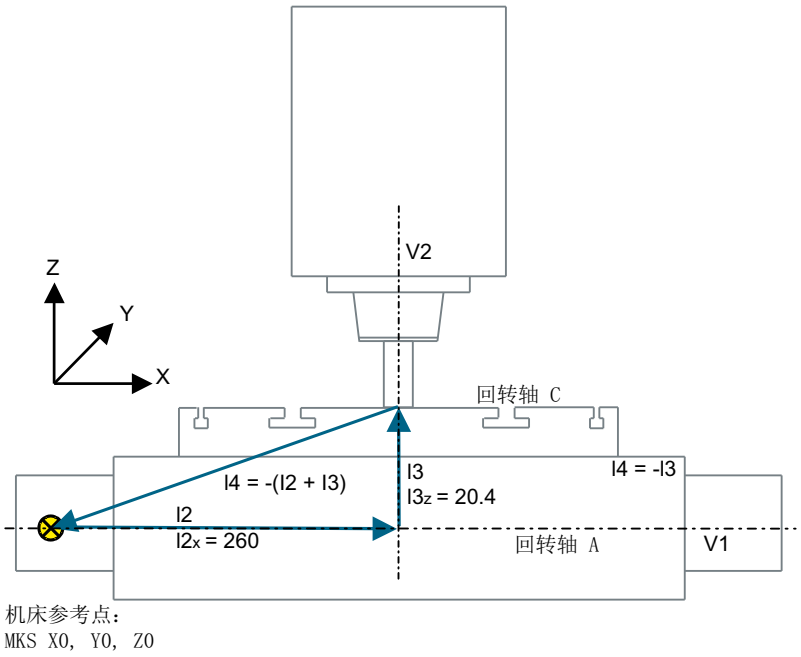


图纸中的矢量适用于机床运动的初始位置。主轴（刀柄/刀架）已定位到工作台边缘上方或者工作台中心上方（回转轴 C）。回转台 C 轴的旋转中心利用主轴上的测头确定。



方向 v2	回转轴 C 围绕 Z 轴旋转。
偏移 I2	从机床参考点到回转轴 1 的旋转中心/交点的距离。
偏移 I3	回转轴 1 的旋转中心和回转轴 2 的旋转中心/交点之间的距离。
偏移 I4	闭合矢量链，I4 = - (I2 + I3)。

图 19-16　自 X 方向的机床侧视图



方向 v1	回转轴 A 围绕 X 轴旋转。
方向 v2	回转轴 C 围绕 Z 轴旋转。
偏移 I2	从机床参考点到回转轴 1 的旋转中心/交点的距离。
偏移 I3	回转轴 1 的旋转中心和回转轴 2 的旋转中心/交点之间的距离。
偏移 I4	闭合矢量链，I4 = - (I2 + I3)。

图 19-17　自 Y 方向的机床前视图

名称	TABLE_5		
使能	是		
类型	回转平面		
运动			
工作台			
偏移 I2	260.000000	200.000000	0.000000
精偏移 I2	0.000000	0.000000	0.000000
旋转轴 1	A		
模式	自动		
方向 v1	-1.000000	0.000000	0.000000
偏移	0.000°		
角度范围	-90.000°	-90.000°	

端面齿	否		
偏移 I3	0.000000	0.020000	20.400000
精偏 I3	0.000000	0.000000	0.000000
旋转轴 2	C		
模式	自动		
方向 v2	0.000000	0.000000	-1.000000
偏移	0.000°		
角度范围	0.000°	360.000°	
端面齿	否		
偏移 I4	-260.000000	-200.020000	-20.400000
精偏 I4	0.000000	0.000000	0.000000

示例 6a：设置带 B 轴的车床（运动链）

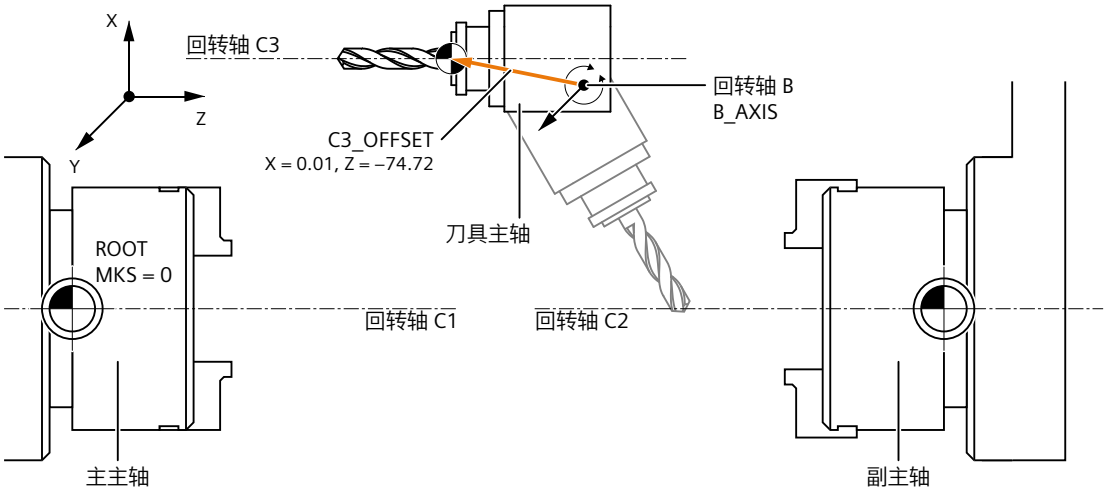


图 19-18　带 B 轴的车床

下面以一台带主主轴、副主轴以及 B 轴摆头的车床为例。

首先必须为该机床创建一条运动链。

运动链的示意图：

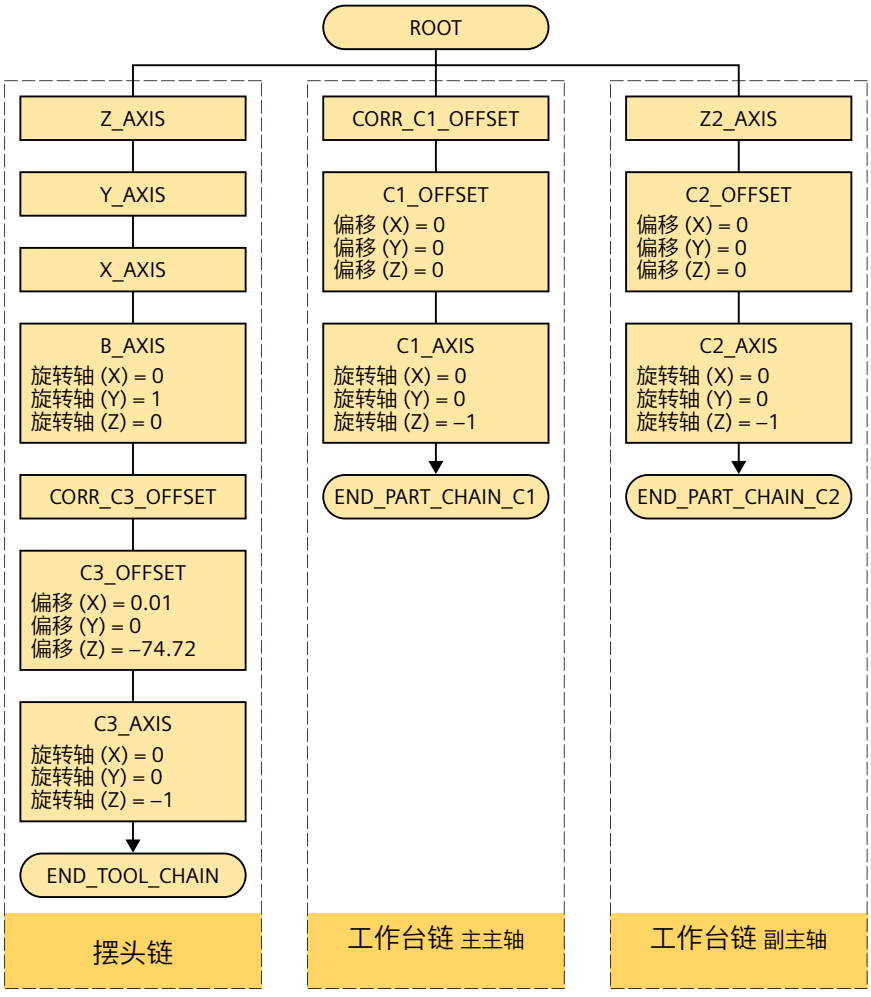


图 19-19　运动链的示意图

创建了机床的运动链后，便可以在它的基础上创建各个 Toolcarrier。

在带有附加 B 回转轴的车床上，刀具可以在 X/Z 平面上定向或回转。如果机床有副主轴，则可以使用一把刀具在主主轴与副主轴上交替进行加工。

使用“刀具定位”和“平面回转”功能需要创建下列 Toolcarrier：

Toolcarrier 1

用于进行刀具定位并使其回转到主主轴和副主轴上（车削、端面 Y/C、外表面 Y/C）：

名称			
使能	是		
类型	车刀定位		
运动			
刀具链			

刀具链开端	Z_AXIS		
旋转轴	B_AXIS		
角度范围		-15.000°	195.000°
端面齿	否		
补偿元素			
旋转轴	C3_AXIS		
角度范围		0.000°	360.000°
端面齿	否		
补偿元素			
刀具链终端	SP1_OFFSET		
闭合刀具链	是		
属性			
回退选择			
• 机床轴 Z 轴			否
选择回转模式			
• 逐轴			是
首选方向选择			是，默认=+
首选方向参考轴			旋转轴 1
跟踪选择			否

在调试“车刀定向”时，必须在回转数据组 1 中将参数 $TC_CARR37[n] 的万位设为 2 或 3。之后在程序编辑器中就会提供用于 Toolcarrier 1 的“刀具回转 - 车刀或铣刀定位”。

更多信息参见“调试 Toolcarrier/回转数据组 (页 483)”一章。

Toolcarrier 2

用于主主轴上（端面 B）的斜面铣削加工：

名称	TC_MAIN_SPINDEL		
使能	是		
类型	回转平面		
运动			
刀具链			
刀具链开端	Z_AXIS		
旋转轴	B_AXIS		

角度范围		-15.000°	195.000°
端面齿	否		
补偿元素	CORR_C3_OFFSET		
刀具链终端	C3_AXIS		
闭合刀具链	是		
工件链			
工件链开头	CORR_C1_OFFSET		
补偿元素	CORR_C1_OFFSET		
旋转轴	C1_AXIS		
角度范围		0.000°	360.000°
端面齿	否		
工件链末端	C1_AXIS		
闭合工件链	是		
属性			
回退选择			
• 机床轴 Z 轴			否
选择回转模式			
• 逐轴			是
首选方向选择			是，默认=+
首选方向参考轴			旋转轴 1
跟踪选择			否

Toolcarrier 3

用于副主轴上（端面 B）斜面铣削加工：

名称	TC_SUB_SPINDEL		
使能	是		
类型	回转平面		
运动			
刀具链			
刀具链开端	Z_AXIS		
旋转轴	B_AXIS		
角度范围		-15.000°	195.000°

端面齿	否		
补偿元素	CORR_C3_OFFSET		
刀具链终端	C3_AXIS		
闭合刀具链	是		
工件链			
工件链开头	CORR_C2_OFFSET		
补偿元素	CORR_C2_OFFSET		
旋转轴	C2_AXIS		
角度范围		0.000°	360.000°
端面齿	否		
工件链末端	C2_AXIS		
闭合工件链	是		
属性			
回退选择			
• 机床轴 Z 轴			否
选择回转模式			
• 逐轴			是
首选方向选择			是，默认=+
首选方向参考轴			旋转轴 1
跟踪选择			否

运动链一次性创建，所有需要的 Toolcarrier 都参照对应的运动链元素。必须修改运动链中一个描述运动的数值时，该值立即对所有 Toolcarrier 生效。

示例 6b：设置带 B 轴的车床（传统）

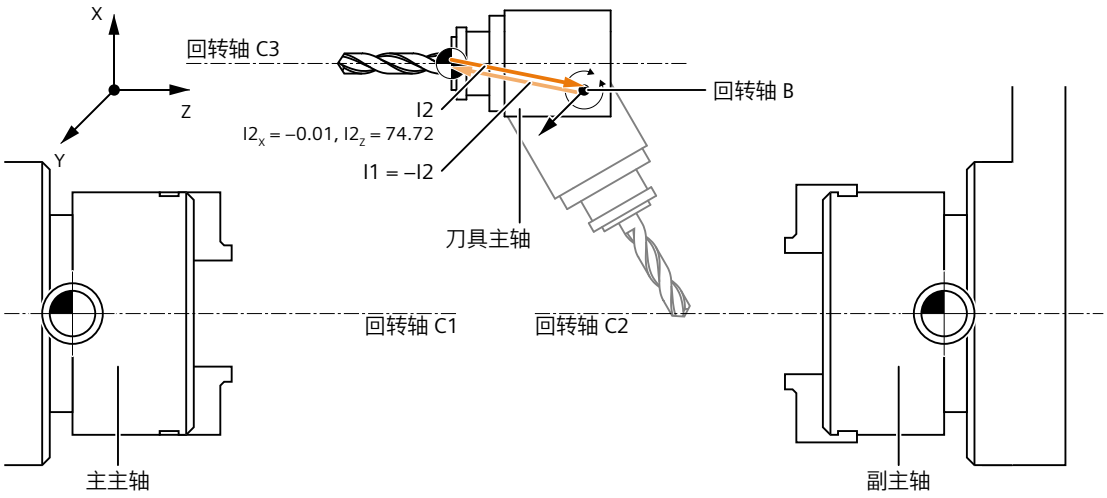


图 19-20　带 B 轴的车床

使用“刀具定位”和“平面回转”功能时，也可以创建下列传统 Toolcarrier：

Toolcarrier 1

用于进行刀具定位并使其回转到主主轴和副主轴上（车削、端面 Y/C、外表面 Y/C）：

名称	TC_1		
使能	是		
类型	车刀定位		
摆头			
偏移 I1	-0.010000	0.000000	-75.070000
精偏移 I1	0.000000	0.000000	0.000000
旋转轴 1	B		
模式	自动		
方向 v1	0.000000	1.000000	0.000000
偏移	0.000°		
角度范围	-15.000°	195.000°	
端面齿	否		
偏移 I2	0.010000	0.000000	75.070000
精偏移 I2	0.000000	0.000000	0.000000
旋转轴 2	C3		
模式	自动		

方向 v2	0.000000	0.000000	1.000000
偏移	0.000°		
角度范围	0.000°	360.000°	
端面齿	否		
偏移 I3	0.000000	0.000000	0.000000
精偏 I3	0.000000	0.000000	0.000000
刀具方向初始位置			- Z
属性			
回退选择			
• 刀具方向增量回退量			是
• 刀具方向最大回退量			是
• 机床轴 Z 轴			
• Z 轴回退位置			200
选择回转模式			
• 逐轴			是
首选方向参考轴			旋转轴 2
跟踪选择			否

在调试“车削工艺的 B 轴运动”时，必须在 Toolcarrier 1 中将参数 $TC_CARR37[n] 的万位设为 2 或 3。之后在程序编辑器中就会提供用于 Toolcarrier 1 的“刀具回转 - 车刀或铣刀定位”。

更多信息参见章节“调试 Toolcarrier/回转数据组 (页 483)”

Toolcarrier 2

用于主主轴上（端面 B）的斜面铣削加工：

名称	TC_2		
使能	是		
类型	回转平面		
运动			
摆头			
精偏移 I1	0.000000	0.000000	0.000000
旋转轴 1	B		
模式	自动		

方向 v1	0.000000	1.000000	1.000000
偏移	0.000°		
角度范围	-15.000°	195.000°	
端面齿	否		
偏移 I2	0.010000	0.000000	75.070000
精偏移 I2	0.000000	0.000000	0.000000
工作台			
偏移 I3	0.000000	0.000000	0.000000
精偏 I3	0.000000	0.000000	0.000000
旋转轴 2	C1		
模式	自动		
方向 v2	0.000000	0.000000	-1.000000
偏移	0.000°		
角度范围	0.000°	360.000°	
端面齿	否		
偏移 I4	0.000000	0.000000	0.000000
精偏 I4	0.000000	0.000000	0.000000
刀具方向初始位置			- Z

Toolcarrier 3

用于副主轴上（端面 B）斜面铣削加工：

名称	TC_3		
使能	是		
类型	回转平面		
运动			
摆头			
偏移 I1	-0.010000	0.000000	-75.070000
精偏移 I1	0.000000	0.000000	0.000000
旋转轴 1	B		
模式	自动		
方向 v1	0.000000	1.000000	1.000000
偏移	0.000°		

角度范围	-15.000°	195.000°	
端面齿	否		
偏移 I2	0.010000	0.000000	75.070000
精偏移 I2	0.000000	0.000000	0.000000
工作台			
偏移 I3	0.000000	0.000000	0.000000
精偏 I3	0.000000	0.000000	0.000000
旋转轴 2	C2		
模式	自动		
方向 v2	0.000000	0.000000	-1.000000
偏移	0.000°		
角度范围	0.000°	360.000°	
端面齿	否		
偏移 I4	0.000000	0.000000	0.000000
精偏 I4	0.000000	0.000000	0.000000
刀具方向初始位置			- Z

示例 7：配置带 B 轴铣床的车削加工

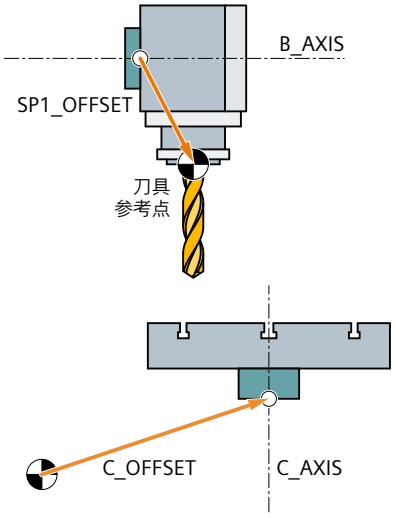


图 19-21　带 B 轴铣床的车削加工

可以配置带 B 轴（绕 Y 旋转）的铣床，使其能够车削加工。可以将车削工艺设置为“扩展工艺”。

更多信息参见章节“铣床上的车削 (页 536)”

下面例子中的铣床带有绕 Y 旋转的回转轴（B）、 绕 Z 旋转的回转轴（C）和刀具方向 Z（G17）上的主主轴（SP）。

使用“刀具定位”和“平面回转”功能需要创建静态坐标转换。

在操作区域“坐标转换”中，创建一个新的坐标转换，它包含摆头上的 2 根旋转轴和工作台上的 1 根旋转轴。

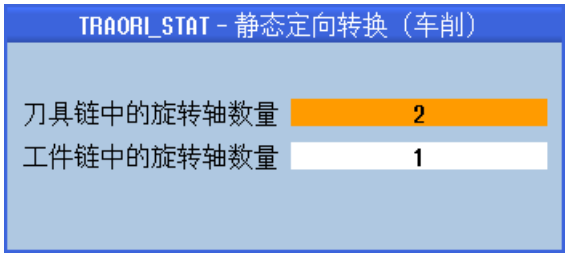


图 19-22　对话框 TRAORI_STAT

名称	TURN_TRAFO		
运动链的指定			
刀具链			
旋转轴	B_AXIS		
旋转轴偏移			0.000°
补偿元素	CORR_SP1_OFFSET		
旋转轴	SP_AXIS		
旋转轴偏移			0.000°
刀具链终端	SP_AXIS		
闭合刀具链	自动		
工件链			
补偿元素	CORR_C2_OFFSET		
旋转轴	C_AXIS		
旋转轴偏移			0.000°
端面齿	否		
工件链末端	C2_AXIS		
闭合工件链	自动		
线性轴			
机床轴 X	X_AXIS_LIN		

机床轴 Y	Y_AXIS_LIN		
机床轴 Z	Z_AXIS_LIN		
特性（输入屏幕）			
回退选择			
• 机床轴 Z 轴			否
选择回转模式			
• 逐轴			是
首选方向选择			是，默认=+
首选方向参考轴			旋转轴 1
跟踪选择			否
特性（坐标转换）			
刀具初始方向			
方向矢量	0.000	0.000	1.000
法向量	0.000	1.000	0.000

在创建的运动链的基础上，可以按如下方式为“平面回转”创建 Toolcarrier：

名称	MIX_BC		
使能	是		
类型	回转平面		
运动			
刀具链			
刀具链开端	Z_AXIS		
旋转轴	B_AXIS		
角度范围		-90.000°	90.000°
端面齿	否		
补偿元素	CORR_SP_OFFSET		
刀具链终端	SP_AXIS		
闭合刀具链	是		
工件链			
工件链开头	CORR_C_OFFSET		
补偿元素	CORR_C_OFFSET		

旋转轴	C_AXIS		
角度范围		0.000°	360.000°
端面齿	否		
工件链末端	C_AXIS		
闭合工件链	是		
属性			
回退选择			
• 机床轴 Z 轴			否
选择回转模式			
• 逐轴			是
首选方向选择			是，默认=+
首选方向参考轴			旋转轴 1
跟踪选择			是

CYCLE800 中的参数 CUTMOD

调用区别

- 在 NC 程序（CYCLE800 车刀定位）中调用数据组 2 时要激活 NC 功能 CUTMOD，以便使刀沿位置或刀具参考点根据刀具定向更新：
- 在 NC 程序中调用回转模式用回转数据组（CYCLE800 刀具定位）时要激活 NC 功能 CUTMOD，以便按照“刀具定位”后当前有效的刀具方向来计算刀沿位置、刀具角度、切削方向和刀具参考点。

更多信息

参数 CUTMOD 的详细说明请见

功能手册之刀具

19.7.7 Herstellerzyklus CUST_800.SPF

功能

回转中的所有轴位置是通过循环 CUST_800.SPF 定义的。这些位置只能从回转循环 CYCLE800 中调用或者从循环
E_TCARR（ShopMill）或 F_TCARR（ShopTurn）中调用。

在循环 CUST_800.SPF 中提供并记录了功能标记（_M2：到 _M59）。也可参见后面的“结构程序 CYCLE800”。

CUST_800.SPF 循环中也有对于铣床上的车削功能的扩展，参见章节“CUST_800 调整 (页 549)”。

说明

修改 CUST_800 时要注意，在 CUST_800 的末尾要再次加入所用 G 功能组的 G 指令（型号 S_G_1 到 S_G_5!）。

参数

CUST_800 (INT _MODE, INT _TC1, REAL _A1, REAL _A2, INT _TC2, REAL _T_POS)

_MODE	跳过标记 _M2 到 _M59
_TC1	“摆头+旋转工作台”编号
_A1	回转轴 1 的角度
_A2	回转轴 2 的角度
_TC2	JOG 模式中回转的进给率系数(%)以及 ShopMill 中新换入的“摆头+旋转工作台”的编号
_T_POS	沿刀具方向退回时的增量位置（参见标记 _M44， _M45）

回转前的回退

如果循环 CUST_800.SPF 没有被修改，在进行回转运动前 Z 轴（标记_M41）首先退回 MCS 中的某个位置，然后是 X 轴和 Y 轴（标记_M42）退回。在系统变量 $TC_CARR38[n] 到 $TC_CARR40[n] 中确定了这些位置值。在轴退回时，之前选中的刀沿(D0) 被撤销，在退回后再次被选中。

在确定了沿刀具方向回退时，刀具会一直回退，直到达到软限位或者从工件出发回退一定的增量行程。此时系统会适当地考虑刀具长度。

回转循环的大致结构

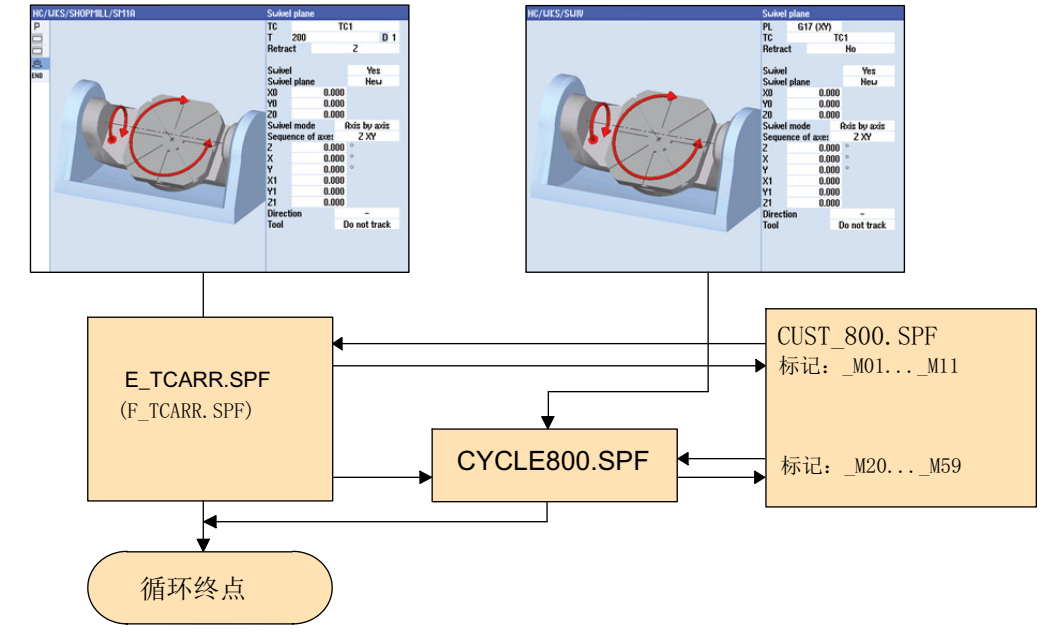


图 19-23　回转循环的大致结构

结构程序 CYCLE800

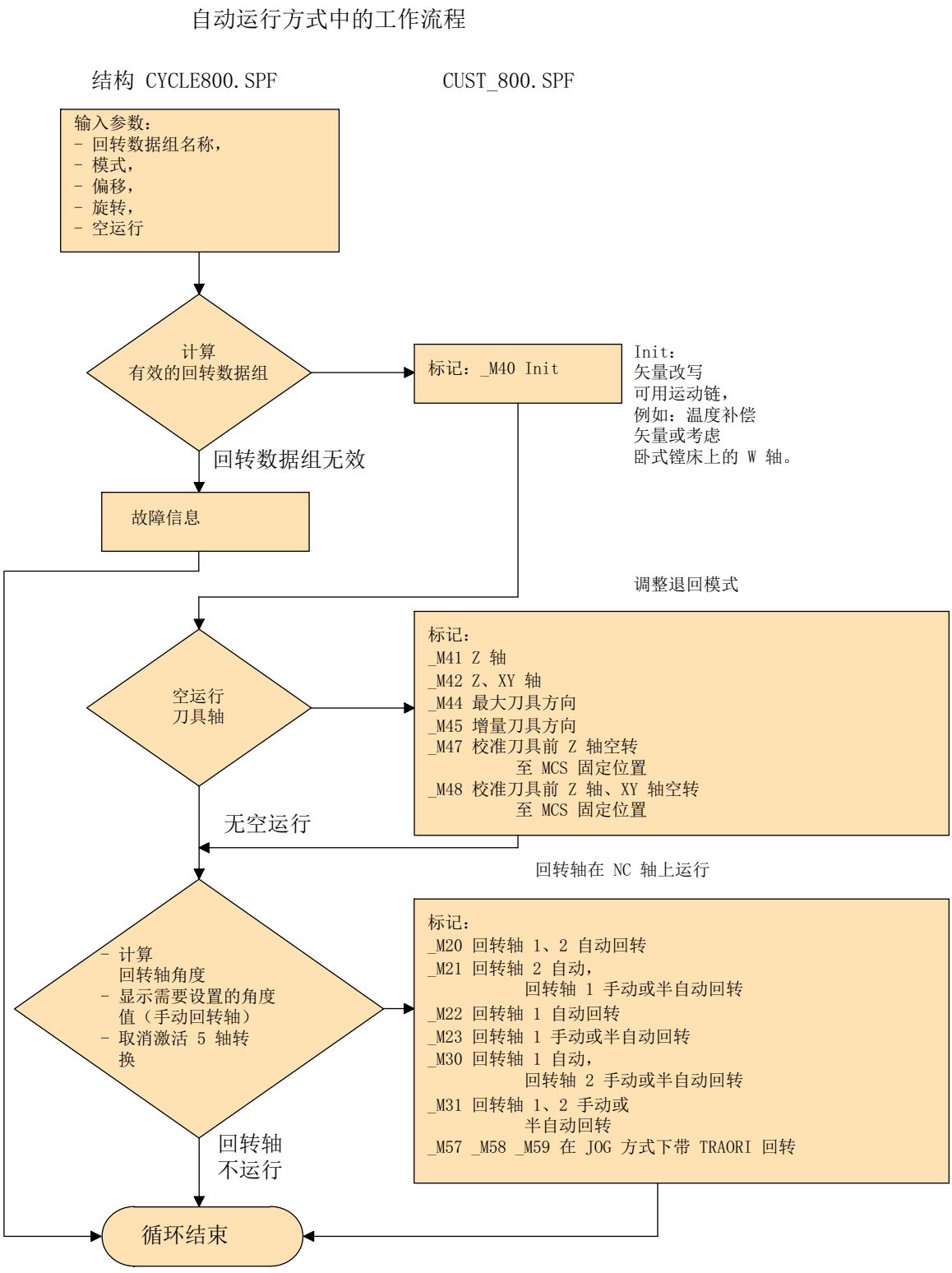


图 19-24　结构：CYCLE800.SPF / CUST_800.SPF

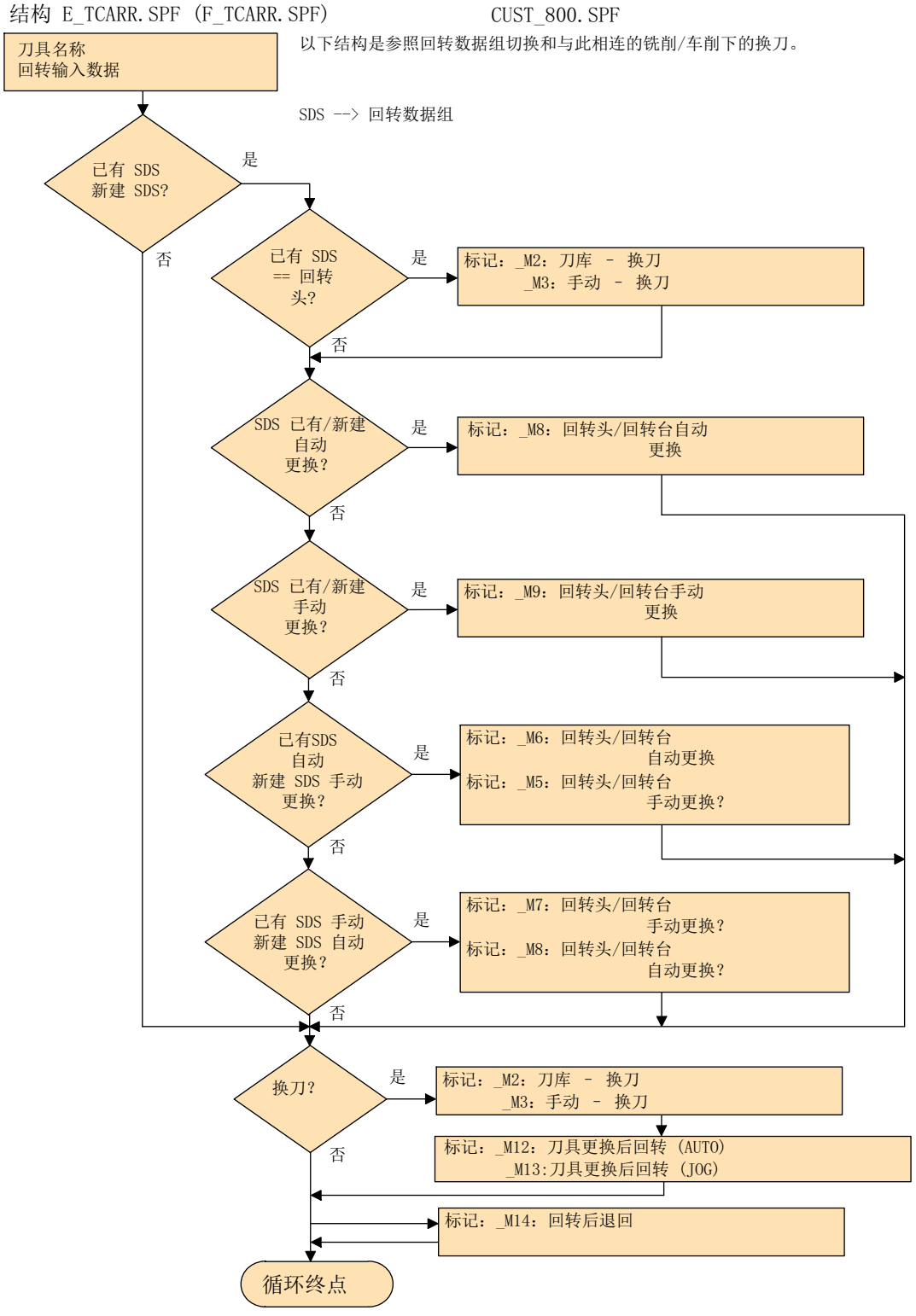


图 19-25 结构：E_TCARR.SPF (F_TCARR.SPF) / CUST_800.SPF

对于标记 _M2 到 _M13 的说明

在切换回转数据组(SDS)或换刀时，线性轴以之前的退回模式（模态有效）退回。

如果在车削/铣削时不需要此特性，则用分号 （；）标注相关的调用。在制造商循环 CUST_800.SPF 中，车削/铣削（参见标记 _M2 到 _M9）调用的是循环 E_SWIV_H 或者 F_SWIV_H。

参数 E_SWIV_H（参数 1、参数 2、参数 3）

- 参数 1：回转数据组号（_TC1）
- 参数 2：角度 1 旋转中心/交点的距离
- 参数 3：角度 2 旋转中心/交点的距离

修改示例

如果在切换回转数据/换刀时不要求定位回转轴（摆头+旋转工作台），可以在对应的标记上对循环 E_SWIV_H 进行注释。如果要求回转轴运行到指定的位置，则可以传送一个角度值到参数 2 和参数 3 中。

对于标记 _M14，_M15 的说明

根据退回平面和程序中的回转平面，程序段搜索结束后，线性轴开始移动，从当前位置移动到一同被回转的退回平面内的软限位上。为了避免该问题，在回转结束后将调用 CUST_800.SPF 中的标记 _M14。其中缺省的循环 E_SP_RP(30)会使轴移动到铣削退回平面上，必要时移动到软限位。可以在标记 _M15 上设置一段程序段结束后的退回动作。

对标记 _M16 的说明

希望在切换回转数据组或撤销回转数据时回转轴运动到零点，可以相应地修改标记_M16 的句法，例如：写入程序段 N8006 E_SWIV_H(_TC1,0,0)。

对标记 _M20 到 _M31 的说明

标记 _M20 到 _M31 的区别在于机床运动带有两个或一个回转轴。此外，区别还在于自动回转轴（已为 NCU 所知）和手动（半自动）回转轴。带有有效回转数据组的回转运动只有一个标记有效。

对标记 _M35 的说明

在程序段搜索和带手动回转轴的回转数据组中使用标记 _M35。

对标记 _M40 的说明

该标记作为用户 Init 在调用和激活功能 Toollcarrier 前激活。在一些特殊的运动过程中，借助该标记可以更新或改写有效回转数据组中的矢量。

钻头上的应用示例：在平行轴上，Z 方向上的矢量随 W 轴的实际值更新。

在该标记处安装了一个 STOPRE。
制造商可以测试 STOPRE 在 CUST_800.spf 中的标记 M40 是否满足其功能要求。此时要在中断条件(RESET)的速度（无 STOPRE）和功能之间进行权衡。该功能也与机床运动类型有关。

针对标记 _M41 到 _M45 和 _M47、_M48 的说明

在标记 _M41 到 _M45 上，可以修改回转轴在执行回转前的回退模式，在标记 _M47 和 _M48 上，可以修改刀具摆动前的回退模式。

刀具沿 Z 轴或先沿 Z 轴，然后沿 X 轴、Y 轴回退到 MCS 位置，在此期间系统会考虑刀具长度分量，并且不撤销刀沿选择。

对于标记 _M41 和 _M42 的说明

在“摆头-旋转工作台”混合机床运动形式中，当选择回退方式“沿 Z 轴”或“先沿 Z 轴，然后沿 X 轴 Y 轴”时，NC 会考虑运动参考点。因此，摆头摆动后所在的位置可能和回退位置 $TC_CARR38 到 $TC_CARR40 的值不同。

不需要该特性时，可以在计算刀具分量时关闭 Toolcarrierer 的标记 M41 和 M42。

说明

可以在结束时再次启用标记 M41 和 M42。

示例：

```
IF (_MODE==41)OR(_MODE==42)OR(_MODE==47)OR(_MODE==48)
 TCARR=0 ; 关闭 Toolcarrier
 TRAFOOF
 ...
ENDIF
...

_M41: ; 沿 Z 轴回退到 MCS 固定位置
...
TCARR=TC1 ; 再次激活 Toolcarrier
GOTOF _MEND

_M42: ; 沿 Z 轴回退，然后沿 X 轴 Y 轴回退到 MCS 固定位置
...
TCARR=TC1 ; 再次激活 Toolcarrier
```

```
GOTOF _MEND
```

对标记 _M46 的说明

标记 _M46 可以设置程序段搜索后、回转前的退回模式。如果是铣削工艺程序，则变量 _E_VER 为 1。

对于标记 _M57 到 _M59 的说明

标记 _M57 到 _M59 用于 JOG 回转和 5 轴转换（TRAORI）。

对“跟踪刀具”的说明

跟踪刀具的前提是，已对与相应的回转数据组相对应的 5 轴转换进行调试。用于跟踪刀具的程序部分集成在标记 _M20， _M21， _M22 和 _M30 中。首个 5 轴转换用 TRAORI(1)调用。

关于换刀 + 回转的说明

总得来说，一台机床上回转 (CYCLE800) 功能和换刀功能是相互独立的功能。因此在一个需要使用多个刀具进行加工的工艺流程中（例如：先定中心、再钻削和攻丝），工作平面在回转后可以保持不变。

在换刀的机械过程中，有效回转数据组的回转轴是否参与或者必须要退回，都需要在换刀程序中加以考虑。换刀后回转轴退回到换刀前的位置。如果有线性轴（几何轴）参与了换刀过程，则不能删除 NC 中的旋转（回转框架），而是要用指令 G153 或 SUPA 将线性轴作为机床轴进行定位。

关于不带有效刀具补偿的回转的说明

如果在没有有效刀沿(D0)条件下回转轴无法回转，您可以修改循环 CUST_800.SPF：

```
_M40:
IF ((NOT $P_TOOL) AND _TC1)
 LOOP
 MSG（“无刀沿有效”）
 M0
 STOPRE
 ENDLOOP
ENDIF
GOTOF_MEND
```

19.7.8 间接编程回转数据组的名称

功能

通道中定义的 GUD 变量 **_TC_CHANGE** 是长度为 32 的字符串。借助这个变量，能够以与调用自身中的编程无关的方式，在 CYCLE800 调用前定义在 CYCLE800 中调用哪些回转数据组。

每次复位都会将 GUD 变量 _TC_CHANGE 重新初始化。在 CYCLE800 的末尾同样会将此变量删除。这表示，在每个新的 CYCLE800 调用前必须重新对这个变量进行赋值。

在 CYCLE800 中，在 CUST_800 跳跃至标记 _M40 后进行回转数据组的切换。故也可以在 CUST_800 中对这个参数进行赋值。

编程示例

```
N10 _TC_CHANGE="HEAD_2"
N20 CYCLE800(0,"HEAD_1",100000,57,0,0,0,0,0,0,0,0,0,-1,100,1)
N30 M0 ;* HEAD_2 生效

N40 _TC_CHANGE="HEAD_3"
N50 CYCLE800(0,"HEAD_1",100000,57,0,0,0,0,0,0,0,0,0,-1,100,1)
N60 M0 ;* HEAD_3 生效

N70 CYCLE800(0,"HEAD_1",100000,57,0,0,0,0,0,0,0,0,0,-1,100,1)
N80 M0 ;* HEAD_1 生效

N90 _TC_CHANGE="TABLE_2"
N100 CYCLE800(0,"TABLE_1",100000,57,0,0,0,0,0,0,0,0,0,-1,100,1)
N110 M0 ;* TABLE_2 生效
N120 M2
```

说明

在每个新的 CYCLE800 调用前必须重新对变量 _TC_CHANGE 进行赋值。

CYCLE800 中编程的边界条件

就下列特性对在 _TC_CHANGE 中述及的回转数据组进行检查：

- 待激活的回转数据组必须已使能。
- 下列参数必须与在屏幕中选择的回转数据组一致：
  - $TC_CARR23（回转数据组类型）
  - $TC_CARR37（回转模式，个位）
  - $TC_CARR37（跟踪选择，万位）
  - 屏幕中设置的模式（位 6，7）
    （轴方式/空间角/投影角/直接），针对 TC_CARR37

19.7.9 CYCLE9960 工件 - 运动测量

功能

使用功能运动测量可以通过测量空间内的曲面位置计算几何矢量，这些矢量用于定义 5 轴转换（TRAORI 和 TCARR）。

通常，工件测头在每个回转轴上接触球体的三到十二个位置以进行测量。球体位置可以由用户根据机床的几何比例确定。每次重新定位待测回转轴，球体位置就会重新设置。

使用范围

功能“运动测量”(CYCLE9960) 可以确定回转轴运动转换 (TRAORI 和 TCARR) 功能相关的数据。

可能的情况：

- 重新确定回转数据组或运动链
  - 机床的调试
  - 使用可回转的夹具作为 TCARR
- 检查回转数据组或运动链
  - 碰撞后的维修
  - 检查加工过程中的坐标转换

手动控制轴（可手动调节的回转台、可回转的夹具等）的坐标转换矢量测量和 NC 控制的回转轴一样。

启动 CYCLE9960 时必须为回转数据配置基础数据（运动类型）。

如果在设定数据 SD55740 $MNS_MEA_FUNCTION_MASK 中置位了位 7，则根据设置，运动测量基于计算出的回转轴矢量，或基于以保存在回转数据组中的回转轴矢量。

前提条件

CYCLE9960（坐标转换测量）的使用必须满足以下前提条件：

- 安装了西门子测量包
- 校准了工件探头
- 安装了标准球
- 可定向刀架 (MD18088 $MN_MM_NUM_TOOL_CARRIER > 0) 或运动链已设置。
- 机床已经回参考点，处于直角坐标系（X, Y, Z）
  垂直针对刀具主轴，可通过测量杆进行控制。
- 参与坐标转换的回转轴的位置已定义
- 参与坐标转换的轴的运行方向符合标准 ISO 841-2001 或 DIN 66217（右手准则）
- 定义的日志文件大小（MD11420 $MN_LEN_PROTOCOL_FILE ≥ 20）

更多信息

详细的测量与 CYCLE9960 编程步骤，包括示例，请参见

编程手册之测量循环。

19.8 铣床上的车削

可以将车削工艺设置为铣床上的扩展工艺。因此，操作软件上包含铣床和车床的所有设置方法。进行工件车削加工时，可以将回转台调为高转速。此时车刀夹紧在铣削主轴位置上并且可以在不同位置上使用。



机床制造商

请注意机床制造商的说明。

19.8.1 设置机床数据和设定数据

设置参数

机床数据

设置以下机床数据以设置该工艺：

MD10602 $MN_FRAME_GEOAX_CHANGE_MODE	几何轴切换时的框架
= 1	几何轴切换时重新计算当前总框架，同时新几何轴的偏移、比例缩放和镜像生效。原有几何轴的旋转保留不变。

MD18114 $MN_MM_ENABLE_TOOL_ORIENT	指定刀具刀沿定向
= 0	刀具定位功能不生效。
	-或者-
= 3	刀具 T = n 的每个刀沿 D = m 分配至以下系统参数： $TC_DPV[n, m] 和 $TC_DPV3 - $TC_DPV5 $TC_DPVN3[n, m], $TC_DPVN4[n, m] 和 $TC_DPVN5[n, m]

说明

如果要通过可变刀具定向设置角度刀具（类型 130），则需要设置 MD18114 = 3。

MD51226 $MNS_FUNCTION_MASK_SIM	模拟的功能标记
位 5 = 1	刀具主轴中的车刀可旋转（用于模拟）

MD51781 $MNS_MEA_T_PROBE_THICKNESS	刀具测头的厚度，用于车刀自动逼近铣床/车床的测头
= 0	缺省值

MD20100 $MC_DIAMETER_AX_DEF	带端面轴功能的几何轴
= X	通过以下机床数据将几何轴定义为断面轴。

MD20110 $MC_RESET_MODE_MASK	复位/零件程序结束后控制系统初始设置的定义
位 0 = 1	系统缺省值

MD20112 $MC_START_MODE_MASK	确定零件程序启动后控制器的初始设置
位 7 = 0	已生效转换的当前设置保持生效。

MD20124 $MC_TOOL_MANAGEMENT_TOOLHOLDER	刀套号
= 1	可变

MD20125 $MC_CUTMOD_ERR	CUTMOD 故障时的报警显示
= CFFFH	显示 CUTMOD 故障时的报警

MD20150 $MC_GCODE_RESET_VALUES[21]	G 功能组的初始设置
= 2	CUT2DF 上 G 功能组 22 的初始化设置

MD20150 $MC_GCODE_RESET_VALUES[55]	G 功能组的初始设置
= 3	G 功能组的初始设置 56 TOWWCS

MD20152 $MC_GCODE_RESET_MODE[21]	G 功能组的复位属性
= 0	G 功能组 22 的复位属性

MD20152 $MC_GCODE_RESET_MODE[55]	G 功能组的复位属性
= 3	G 功能组的复位属性 56 TOWWCS

MD20360 $MC_TOOL_PARAMETER_DEF_MASK	刀具参数定义
位 17 = 0	车刀和铣刀 CUTMOD 中，刀沿用于计算补偿值，而不是映射在加工平面上。
位 18 = 1	车刀和铣刀 CUTMOD 中一直使用车刀和铣刀有效的平面。
位 22 = 1	运动关系转换生效时会实时参考刀具磨损。

MD21203 $MC_LIFTFAST_MODE_MASK	快速退刀设置
位 0 = 1	快速提升时该位激活了方向监控。

MD24030 $MC_FRAME_ACS_SET	SZS 坐标系的设置
= 1	SZS 通过 $P_CYCFRAME 由 WCS 转换而成

MD35040 $MA_SPIND_ACTIVE_AFTER_RESET[ ]	主轴的复位属性
[刀具主轴] = 2	刀具主轴在 JOG 中不停止 → 提供在 T、S、M 屏幕中
[车削主轴] = 2	车削主轴在 JOG 中不停止 → 提供在 T、S、M 屏幕中

MD52001 $MCS_DISP_COORDINATE_SYSTEM_2	车削加工时,在操作界面上确定所显示的方向（可变）
= 0	

MD52200 $MCS_TECHNOLOGY	工艺
= 2	铣削

MD52201 $MCS_TECHNOLOGY_EXTENSION	扩展工艺
= 1	车削

MD52206 $MCS_AXIS_USAGE[ ]	通道中轴的含义
[刀具主轴] = 1	刀具主轴
[车削主轴] = 3	车削主轴

MD52207 $MCS_AXIS_USAGE_ATTRIB[车削主轴]	轴的属性
位 8 = 1	提供用于毛坯夹具的回转台

MD52212 $MCS_FUNCTION_MASK_TECH	工艺通用的功能屏幕
位 10 = 1	通过运动转换进行车刀校准

MD52214 $MCS_FUNCTION_MASK_MILL	铣削功能屏幕
位 1 = 1	提供用于毛坯夹具的固定工作台

说明

只有在除了旋转工作台之外还有固定工作台时才设置 MD52214 中的位 1。

MD52780 $MCS_J_MEA_T_PROBE_APPR_MODE	JOG 中“测量刀具”时刀具的逼近方式
= 0	随工艺变化的逼近刀具测头的方式（缺省值） 车削工艺：手动逼近 铣削工艺：自动逼近
= 1	随工艺变化的逼近刀具测头的方式 手动逼近刀具测头
= 2	随工艺变化的逼近刀具测头的方式 自动逼近刀具测头

MD52242 $MCS_TURN_TOOL_FIXING	车刀的刀架
= 0	车刀在刀具主轴中

= 1	车刀固定到刀具主轴箱上
= 2	车刀在刀具主轴中或固定到刀具主轴箱上

说明

根据 MD52242 中所设置的安装校准，“车刀校准”对话框中提供多种不同的参数。

设定数据

推荐使用下列设定数据。设置设定数据以设置该工艺：

注意
碰撞危险 在程序运行期间，如果 SD42940、SD42942 和 SD42950 的值发生修改，则会导致碰撞。仅在调试时确定设定数据。

SD42935 $SC_WEAR_TRANSFORM	用于刀具分量的转换
= 1	铣刀定向：缺省 G-组 56:注意 TOWWCS 并设置

SD42940 $SC_TOOL_LENGTH_CONST	平面切换时切换刀具长度分量
= 17	L1 = Z, L2 = Y, L3 = X

SD42942 $SC_TOOL_LENGTH_CONST_T	平面切换时切换车刀刀具长度分量
= 19	L1 = Z, L2 = Y, L3 = X

说明

SD42942 首先通过设置 SD42950 进行计算。

SD42950 $SC_TOOL_LENGTH_TYPE	刀具长度补偿的分配与刀具类型无关
= 3	刀具长度分量的分配分为针对车刀和磨刀的（刀具类型 400 到 599）以及针对其他所有刀具（铣刀）的。

SD42954 $SC_TOOL_ORI_CONST_M	平面切换时切换铣刀刀具长度分量
= 0	铣刀定向：缺省

SD42956 $SC_TOOL_ORI_CONST_T	平面切换时切换车刀刀具长度分量
= 18	车刀定向：G18

SD42998 $SC_CUTMOD_PLANE_TOL	刀刃/加工平面偏差
= 5	Gamma 最大允许的偏差为 0° 或 180°。

SD42999 $SC_ORISOLH_INCLINE_TOL	ORISOLH 上刀刃/加工平面偏差
= 1	“刀具直接校准”时刀刃和 G18 平面之间允许的最大倾斜度

在下面的设定数据中，索引 [k] 代表测头当前数据组的编号（即“测头编号 -1”）。

SD54633 $SNS_MEA_TP_TYPE[k]	指定测头类型：立方体/圆片
= 0	兼容性（测量循环：立方体，车削表面表示测头，铣削表面表示砂轮）
= 3	立方体

SD54648 $SNS_MEA_TPW_TYPE[k]	指定探头的规格
= 0	兼容性（测量循环：立方体，车削表面表示测头，铣削表面表示砂轮）
= 3	立方体

其他设置

- 通过可设置的零点偏移将当前零点放置在车刀工件的中心。
- 在调用“车刀校准”前切换车刀。
- 必须在零件程序中编程刀具以进行模拟。

  说明

  模拟开始后没有刀具生效。

- 在调用“车刀校准”后，以 Y = 0 运行刀具。

- 通过刀具切换循环来限制刀具主轴的最大转速，以避免车刀转速过高。
- 刀具管理中的车刀由系统自动进行设置。可通过文件 sltmmillturnlistconfig.xml 自行调整，例如：用于 OEM 刀具列表。

19.8.2 设置运动转换

必须为车削加工设置至少一个运动转换，才能使用铣床上的车削工艺。

前提条件

机床上已经有一个运动链。

更多信息

有关运动元素的详细信息请参见

操作手册，通用版/车削版/铣削版/磨削版。

设置参数

设置以下机床数据以设置运动转换：

MD18866 $MN_MM_NUM_KIN_TRAFOS	NC 中转换对象的最大数量
≥ 1	该机床数据指定了 NC 中转换的最大数量，该数量是可通过运动链进行定义的（可变）。

设置系统变量

使用以下系统变量来设置运动转换：

系统变量	含义
$NT_NAME[n]	在操作界面上显示的转换的名称
$NT_TRAFO_TYPE[n]	铣床上车削的转换类型：TRAORI_STAT
$NT_T_CHAIN_LAST_ELEM[n]	运动链末尾描述刀具方向的运动链元素的名称
$NT_P_CHAIN_LAST_ELEM[n]	运动链末尾描述工件方向的运动链元素的名称
$NT_GEO_AX_NAME[n, 0 - 2]	描述几何轴 1 到 3（X, Y, Z）的链元素名称

系统变量	含义
$NT_ROT_AX_NAME[n, 0 - 2]	描述回转轴 1 到 3（旋转工作台、回转轴、刀具主轴）的链元素名称
$NT_IDENT[n, 0 - 2]	用来调整屏幕的设置
$NT_AUX_POS[n, 0 - 2]	在几何轴 1 到 3（X、Y、Z）方向上空运行的值 [1)]
$NT_CNTRL[n]	用来调整机床的设置

n:转换数据组编号，其中 n = 1, 2, 3, ...(MD18866 $MN_MM_NUM_KIN_TRAFOS)

[1)] 参见制造商循环 CUST_800.SPF 屏幕 _M47 和 _M48

$NT_IDENT[n, 0 - 2] 小数位含义

可以设置以下用于铣床上车削的输入屏幕显示变量：

小数位	含义	
个位	校准选择	
	0 =	Beta/Gamma
	1 =	Beta/Gamma + 直接校准 (B/SP)
千位	选择栏：回转轴的 +/- 方向参考	
	0 =	界面上无选择，一直为负
	4 =	通过界面进行 +/- 选择，缺省值：负
	5 =	界面上无选择，一直为正
	9 =	通过界面进行 +/- 选择，缺省值：正
万位	选择栏：回转时跟踪刀尖	
	2 =	不提供切换
	3 =	提供切换
百万位	选择栏：空运行模式 0X 和	
千万位	选择栏：空运行模式 X0（所允许的空运行模式由单个值的总和得出：0 ... 15）	
	0 =	无空运行
	1 =	空运行 Z [2)]
	2 =	空运行 Z、XY [2)]
	4 =	刀具方向上空运行，最大
	8 =	刀具方向上空运行，增量

小数位	含义	
亿	使能数据组	
	4 =	运动生效(固定值)

2) 参见系统变量 $NT_AUX_POS[n,0] 到 $NT_AUX_POS[n,2]

$NT_CNTRL[n] 位的含义

可进行以下设置用于机床调整：

位 0 = 1	顺序轴固定 为了更简单地处理数据要确定好顺序轴固定： 1. 旋转工作台 2. 回转轴 3. 刀具主轴
位 1 = 1	旋转工作台为主轴
位 2 = 0	回转轴不是主轴
位 3 = 0	刀具主轴不是主轴（车削时）
位 4 = 0	1. 回转轴没有切端面齿（旋转工作台）3)
位 5 = 0	2. 回转轴没有切端面齿（回转轴）3)
位 6 = 0	3. 回转轴没有切端面齿（刀具主轴）3)

3) 只有在第 2 或第 3 回转轴有切端面齿时才允许。

19.8.3 复位和重新上电后的特性

复位后的特性

进行以下设置，使得复位后也能保持运动转换并在 JOG 模式下使用车刀进行加工：

MD20110 $MC_RESET_MODE_MASK	确定复位/零件程序结束后控制系统的初始设置
位 7 = 1	复位/零件程序结束后已生效转换的当前设置保持生效。
位 16 = 1	主站主轴的当前设置生效。

MD20112 $MC_START_MODE_MASK	确定零件程序启动后控制器的初始设置
位 7 = 0	“有效运动转换”的起始特性

MD20152 $MC_GCODE_RESET_MODE[ ]	G 功能组的复位属性
[5] = 1	复位/零件程序结束后第 6 G 功能组的当前设置保持生效（G17...）。
[14] = 1	复位/零件程序结束后第 15 G 功能组的当前设置保持生效（G94...）。
[28] = 1	复位/零件程序结束后第 29 G 功能组的当前设置保持生效（DIAMxxx）。

重新上电后的特性

为了在重新上电后也能保持运动转换生效，必须进行以下附加设置：

MD20144 $MC_TRAFO_MODE_MASK	坐标转换功能的选择
位 1 = 1	引导启动后转换的重新选择

MD24008 $MC_CHSFRAME_POWERON_MASK	系统框架在上电后重置
位 4 = 0	工件参考点在上电后不会删除

说明

CUTMODK 功能在重新上电后不会保持生效。

19.8.4 设置刀具夹紧角度

当主轴零点位置中已夹紧的车刀不在 G18 平面上时，必须设置夹紧角度。

当刀具位于所需的机械位置上时，应将此时的刀具主轴位置作为夹紧角输入。

步骤



1. 选择操作区域“参数”。



2. 按下软键“刀具表”。

   “刀具表”窗口打开。

3. 在列表中选择一个相应的车刀。



4. 按下软键“其他数据”。

   打开窗口“其他数据- ...”。

   当“其它数据”窗口为选定的车刀进行了配置时，软键“其它数据”才有效。

5. 在“角度”输入栏中输入所需的值。

说明

夹紧角度保存在刀沿参数 $TC_DPROT 中。

19.8.5 设置切端面齿

如果刀具主轴或回转轴具有一个单位为 2.5° （360°/2.5°= 144 分度）的切端面齿时，可进行以下设置：

设置

MD30505 $MA_HIRTH_IS_ACTIVE	轴是带切端面齿的分度轴
= 0	刀具主轴/回转轴不是真正的切端面齿

MD30330 $MA MODULO RANGE	模数区域的大小
= 360°	在模数轴上生效

MD30340 $MA_MODULO_RANGE_START	模数区域的起始位置
= 0°	

MD30501 $MA_INDEX_AX_NUMERATOR	分度轴等分位置分子
= 360°	在非模数轴上生效

MD30502 $MA_INDEX_AX_DENOMINATOR	分度轴等分位置分母
= 144	该值用于确定分母，以计算等分位置上两个位置的间距。 模数轴上用该值指定分度位置的数量。

回转轴具有切端面齿

$NT_CNTRL[n]	
位 5 = 1	回转轴具有切端面齿 [n] 转换名称

刀具主轴具有切端面齿

$NT_CNTRL[n]	
位 6 = 1	刀具主轴具有切端面齿 [n] 转换名称

19.8.6 铣削和车削不同的坐标系

钻头上的旋转工作台通常是围绕 Y 轴（B 轴）旋转的。 车削加工时，G18 平面上的工件必须围绕 Z 轴（C 轴）旋转。 因此，在铣削和车削时，机床上必须使用不同的坐标系。

设置

通过以下通道专用的机床数据可以设置对应的坐标系（例如：用于钻头的值）：

MD52000 $MCS_DISP_COORDINATE_SYSTEM	坐标系的位置（可变）
= 16	Y 轴向上，Z 轴向右

MD52001 DISP_COORDINATE_SYSTEM_2	铣床上车削的坐标系（可变）
= 0	Z 轴向上，X 轴向右

根据以下 CUST_800.SPF 循环中的屏幕 _M70 所示描述坐标系旋转：

```
_M70:            ；用于车削的坐标系旋转
N800703 $P_WPFRAME=$P_WPFRAME:CROT(Y,-90):CROT(X,-90)
```

19.8.7 CUST_800 调整

调整 CUST_800 循环中的这些屏幕用于铣床上的车削：

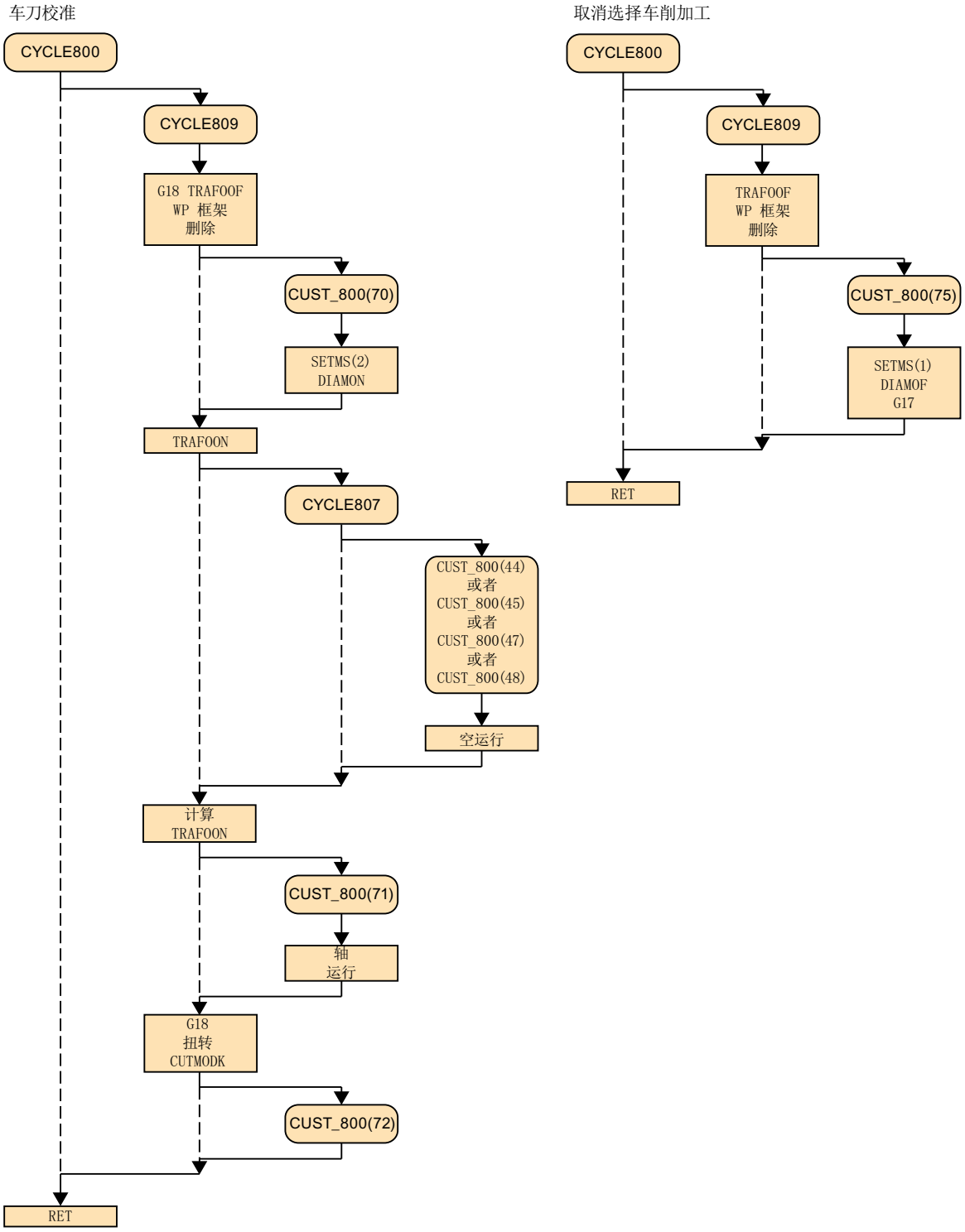
功能屏幕

```
_M40:                                ；用户 Init
N800400 SETMS(1)                     ；设置刀具主轴作为主主轴
N800401 DIAMOF                       ；铣削时未确定作为直径编程的端面轴
GOTOF _MEND

_M47:                                ；刀具校准前，Z 轴空运行至 MCS 固定位置
GOTOF _MEND

_M48:                                ；刀具校准前，Z 轴先空运行至 MCS 固定位置，然后是 XY 轴
GOTOF _MEND

_M70:                                ；校准刀具：准备
IF($P_SMODE[1]==1)                   ；刀具主轴主轴模式
N800700 SPOS[1]=0                    ；作为回转轴 C 的刀具主轴
ENDIF
N800701 SETMS(2)                     ；作为主主轴设置工作台
N800702 DIAM90                       ；车削时确定作为直径编程的端面轴
GOTO _MEND

_M71:                                ；校准刀具：回转轴
_AX1=$PC_TRAFO_ROT_CHAN_AX_IN[1]     ；回转轴 B
_AX2=$PC_TRAFO_ROT_CHAN_AX_IN[2]     ；刀具主轴
```

```
IF(_TC_N_WZ==0)                          ; 无工具提示框跟踪
N800710 TRAFOOF
ENDIF
SBLON
N800711 G0 G40 G60 G90 AX[_AX1]=_A1 AX[_AX2]=_A2
SBLOF
GOTOF _MEND

_M72:                                    ; 校准刀具：结束
GOTOF _MEND

_M75:                                    ; 校准刀具：撤销
N800750 G17                              ; 铣削标准平面
N800751 SETMS(1)                         ; 设置刀具主轴作为主主轴
N800752 DIAMOF                           ; 铣削时未确定作为直径编程的端面轴
GOTOF _MEND
```

简化的流程图

刀具空运行时，在回转轴定位之前根据空运行方式执行屏幕 _M44 到 _M48。



带退回到 MCS 固定位置的回转请参见章节 “Herstellerzyklus CUST_800.SPF (页 527)”。

19.8.8 程序段搜索后的回转轴定位

程序段搜索后必须如下进行回转轴定位（参照回转平面）。通常涉及到两个回转轴（回转轴和刀具主轴）：

步骤

通过以下系统变量读出回转轴的位置值：

系统变量	含义
$AC_RETPOINT[轴]	ASUB 中的再逼近点

在用户循环 CYCPE_MA.SPF 中定位回转轴。

说明

在车刀校准程序段后启动 G 代码程序中的程序段搜索，这样原有的回转轴位置就不会改变。

有关 CYCPE_MA.SPF 的**更多信息**参见章节“扩展标准循环 PROG_EVENT（页 418)”。

如果关闭了用于刀具切换的运动转换，可通过以下程序扩展重新打开转换：

```
DEF STRING[32] U_TRAFO_NAME
U_TRAFO_NAME=$AC_TRAFO_NAME
TRAFOOF
...
TRAFOON(U_TRAFO_NAME)
```

19.8.9 程序段搜索后回转轴定位时的跟踪

程序段搜索后必须进行回转轴定位。如果定位时进行跟踪，直线轴则可能会到达软件限位开关。为避免此过程，回转轴定位时通常不进行跟踪。

CYCPE1MA.SPF 的补充

如果要在定位回转轴时进行跟踪，可在用户循环 CYCPE1MA.SPF 中补充以下内容：

```
; “带跟踪”的回转轴定位：
IF($P_PROG_EVENT==5) ; 程序段搜索
  IF($PC_TRAFO_TYPE_NAME=="TRAORI_STAT")
```

```
    IF(($NT_IDENT[($P_TRAFO_PARSET MOD 1000),0] _DEC5)>=2)
      _TC_N_WZ=1 ; 跟踪刀具
      CUST_800(71,,$AC_RETPOINT[$PC_TRAFO_ROT_CHAN_AX_IN[1]],
      $AC_RETPOINT[$PC_TRAFO_ROT_CHAN_AX_IN[2]])
    ENDIF
  ENDIF
ENDIF
```

19.9 装料适配控制器设置

19.9.1 装料适配 (CYCLE782)

功能

工件和夹具的转动惯量会影响机床的动态特性。在轴运动时，可能在工件加工过程中产生误差。使用 CYCLE782 循环可以根据具体情况自动调整驱动的控制器设置和轴的动态响应设置，从而可以更快、更精确地加工工件。



软件选件

使用该功能需要下列软件选件：

“智能负载适配”

说明

设置了选件“智能负载适配”时，机床数据 MD19321 $ON_TECHNO_FUNCTION_MASK_1 的位 20 会激活，而且输入屏幕上会显示软键“根据负载适配”和“适配”。

配置输入屏幕

通过以下通道专用的机床数据可以设置输入屏幕打开时的负载属性：

MD52212 $MCS_FUNCTION_MASK_TECH	工艺通用的功能屏幕
位 18	激活确定总负载：空轴 + 工件

= 0	在循环 CYCLE782 中激活选择“单独确定负载”。确定的负载值则只针对当前工件。（缺省）
= 1	在循环 CYCLE782 中确定总负载。确定的负载值为二者的总和：空载轴 + 工件。

其他设置

MD53350 $MAS_ILC_BASE_VALUE	轴的基本转动惯量
= 0	（缺省） 在此机床数据中保存了不含工件的空轴的惯量。 该值应在机床调试时一次性确定。 该值在使用循环 CYCLE782 单独确定负载的功能时需要用到。

线性轴上的值输出

无论是线性轴还是回转轴，都可以指定或设定负载。在测量结果画面上，线性轴通常显示质量；而回转轴通常显示惯量。但线性轴的数值显示方式会可能会随着驱动方式变化，比如：

线性轴的驱动	值	单位
主轴或回转轴	惯量	kgm^2
通过滚珠丝杠、直线电机或齿轮箱旋转	质量	kg

更多信息

有关 CYCLE782 的操作和程序结构的更多信息请参见：

- 操作手册，通用版/车削版/铣削版/磨削版
- NC 编程手册

19.9.2 CYCLE782:帮助图形

帮助图像集成在循环 CYCLE782 输入窗口中用于支持。

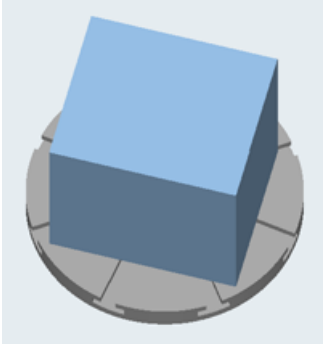
当已根据给定的机器配置设定了下列通道专用的机床数据之后，帮助图形会正确地显示在输入窗口中：

MD52206 $MCS_AXIS_USAGE[n]	通道中轴的含义
MD52207 $MCS_AXIS_USAGE_ATTRIB[n]	轴的属性

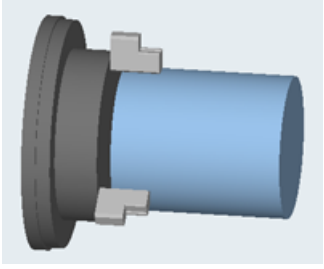
对 MD52206 和 MD52207 的详细描述参见“设置用于模拟的坐标系 (页 179)”和“激活车削/铣削/钻削/磨削工艺 (页 407)”章节。

进给轴辅助画面

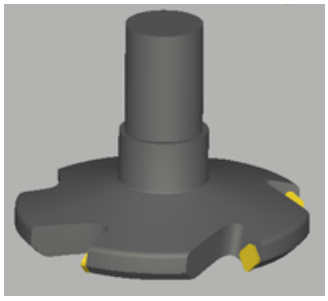
根据轴类型，会显示如下帮助画面：



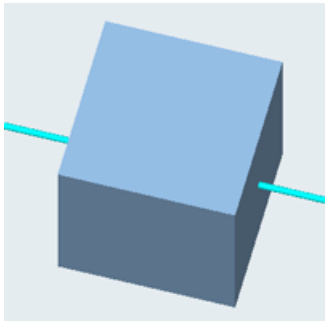
C 轴



主主轴



刀具主轴 Z



线性轴

19.9.3 制造商循环 CUST_782.SPF

功能

循环 CUST_782 由 CYCLE782（测定装料）从内部调用。此循环用于实现所需的操作路径和操作条件来测定装料。这依机器而定，且必须由机器制造商进行相应的指定。

调整制造商循环 CUST_782.SPF

利用参数_MODE 可以执行以下操作：

标记	操作
_M1	在 CYCLE782 的开始处调用
_M10	回到起始位置
_M11	测定装料的运行运动
_M12	测定摩擦的运行运动
_M19	到达终点
_M99	在 CYCLE782 的终点处调用

复位转动惯量的数值

在装料测定之前，可通过将以下变量值改写为所定义的值来复位上次在 CUST_782 运行曲线上测定的转动惯量。

$VA_RESET_INERTIA_TOTAL[ax]	转动惯量评估器的复位
= -1	复位转动惯量和负载
= 0	无操作（读入时默认设置）
= 1	复位转动惯量
ax = 系统中的轴	

为了不使旧的转动惯量对新的结果产生影响，请在每次重新运行前通过参数设置 = 1 将上次估算的转动惯量复位。这样既可重新生成评估结果。

如要同时复位估算的负载力矩（例如摩擦力），请使用参数设置 = -1。

说明

系统变量 $VA_RESET_INERTIA_TOTAL 仅在 SINAMICS 驱动上可用。它描述了 SINAMICS 参数 p1565。

更多信息

有关所使用的参数、调试及轴功能边界条件的更多信息，请见以下文档：

- SINAMICS S120 功能手册驱动功能
- SINAMICS S120/S150 参数手册

19.10 高速加工自由形状表面

19.10.1 配置高速设定 CYCLE832

功能

使用高速设定功能（CYCLE832）可以预先指定加工自由形状的表面所需的最佳数据。CYCLE832 的调用包含三个参数：

- 公差
- 加工方式（工艺）
- 输入定向公差
  第 3 个参数只用于带多轴方向转换的机床(TRAORI) 。

快速设定功能（CYCLE832）除了提供“精优曲面”外，还提供扩展功能“臻优曲面”。通过“臻优曲面”可以改进自由形状表面的表面质量。设置激活时，轨迹控制在轮廓公差内进行优化。调用 CYCLE832 包含额外的参数平滑。



软件选件

使用该功能需要下列软件选件：

“臻优曲面”

设置参数

设置以下机床和设定数据以设置臻优曲面扩展：

SD55220 $SCS_FUNCTION_MASK_MILL_TOL_SET	高速设定功能设定 CYCLE832
位 2	在高速设定屏幕中有模具制造功能可选
= 0	自动使用提供的最佳模具制造功能： • 臻优曲面未激活 → 精优曲面 • 臻优曲面激活 → 臻优曲面
= 1	可在输入屏幕中选择模具制造（精优曲面或臻优曲面）。
位 4	选择显示“平滑”
= 0	选择在屏幕中不显示“平滑”（对应平滑 = 支持）
= 1	选择在屏幕中显示“平滑”

MD52441 $MCS_LIMIT_RES_PATH_SPEED	轨迹运动中计算出的轨迹速度的限值
mm/min	

通过 MD52441 $MCS_LIMIT_RES_PATH_SPEED，系统会在一些关键程序段中自动限制或下调从各轴速度限值计算出的轨迹速度。

计算出的轨迹速度会被下调到该机床数据指定的限值以下。输入 0 则限值无效。

另外，在撤销 CYCLE832、执行复位或零件程序结束时，该限值也会失效。

轴/轮廓公差

功能“精优曲面”与“臻优曲面”之间的区别如下：

“精优曲面”功能下，线性轴（几何轴）的公差通过 NC 指令 CTOL 传送给数控系统。
CTOL = √3 * 轴公差

“臻优曲面”功能下，轮廓公差通过 NC 指令 CTOL 传输给数控系统。

公差为 0：公差值（CTOL）从以下设定数据中加载：

示例：CYCLE832(0,_ROUGH,1) → 公差来自 SD55446。

SD55446 $SCS_MILL_TOL_VALUE_ROUGH	粗加工 CYCLE832 的公差值
SD55447 $SCS_MILL_TOL_VALUE_SEMIFIN	初精整 CYCLE832 的公差值
SD55448 $SCS_MILL_TOL_VALUE_FINISH	精加工 CYCLE832 的公差值
SD55449 $SCS_MILL_TOL_VALUE_PRECISION	精整 CYCLE832 的公差值



软件选件

如要使用轮廓公差，则需要软件选件：
“臻优曲面”

定向公差

如果回转轴参与加工（多轴转换），则定向公差值通过 NC 指令 OTOL 传送给数控系统。

为此有 2 个选项：

1. 直接输入定向公差 OTOL
   加工方式：_ORI_ROUGH、_ORI_SEMIFIN、_ORI_FINISH、_ORI_PRECISION
   如要在 CYCLE832 的输入屏幕中输入定向公差，
   则必须设置设定数据 SD55220 $SCS_FUNCTION_MASK_MILL_TOL_SET 位 1 = 1。
   定向公差的默认值保存在以下设定数据中：

SD55451 $SCS_MILL_ORI_TOL_ROUGH	粗加工 CYCLE832 的定向公差值
SD55452 $SCS_MILL_ORI_TOL_SEMIFIN	初精整 CYCLE832 的定向公差值
SD55453 $SCS_MILL_ORI_TOL_FINISH	精加工 CYCLE832 的定向公差值
SD55454 $SCS_MILL_ORI_TOL_PRECISION	精整 CYCLE832 的定向公差值

2. 公差和某个系数相乘后传送给 NC 指令 OTOL。
   OTOL = 公差 * 系数 * √2
   加工方式：_ROUGH、_SEMIFIN、_FINISH、_PRECISION
   在下列通道专用设定数据中可以为每种加工方式设定此系数：

SD55441 $SCS_MILL_TOL_FACTOR_ROUGH	G 功能组 59 用于粗加工的回转轴公差系数。
SD55442 $SCS_MILL_TOL_FACTOR_SEMIFIN	G 功能组 59 用于初精整的回转轴公差系数。
SD55443 $SCS_MILL_TOL_FACTOR_FINISH	G 功能组 59 用于精加工的回转轴公差系数。
SD55444 $SCS_MILL_TOL_FACTOR_PRECISION	G 功能组 59 用于精整的回转轴公差系数。
= 10	

示例：多轴转换，SCS_MILL_TOL_FACTOR...=10

	CTOL	**OTOL**
CYCLE832(0.1,_ROUGH,1)	0.1 * √3 = 0.173	0.1 * √2 * 系数 = 1.414
CYCLE832(0.05,TOP_SURFACE_SMOOTH_ON +_SEMIFIN,1)	0.05	0.05 * √2 * 系数 = 1.414
CYCLE832(0.05,TOP_SURFACE_SMOOTH_ON +_ORI_SEMIFIN,0.3)	0.05	0.3

平滑

使用功能“臻优曲面”，平滑会被 CYCLE832 关闭。为此必须设置设定数据 SD55220 $SCS_FUNCTION_MASK_MILL_TOL_SET 位 4 = 1。

模具制造功能	简明文本
臻优曲面，不带平滑	TOP_SURFACE_SMOOTH_OFF+
臻优曲面，带平滑	TOP_SURFACE_SMOOTH_ON+

加工方式和工艺 G 功能组 59

在 CYCLE832 中或在 CUST_832.SPF 中为 G 功能组 59 固定分配工艺加工方式：

加工方式	简明文本	工艺 G 功能组 59	数组下标
取消	_OFF	DYNNORM	0
粗加工	_ROUGH	DYNROUGH	2
初精整	_SEMIFIN	DYNSEMIFIN	3
精加工/速度	_FINISH	DYNFINISH	4
精整/精度	_PRECISION	DYNPREC	5
粗加工*)	_ORI_ROUGH	DYNROUGH	2
初精整*)	_ORI_SEMIFIN	DYNSEMIFIN	3
精加工/速度 *)	_ORI_FINISH	DYNFINISH	4
精整/精度 *)	_ORI_PRECISION	DYNPREC	5

*) 含定向公差输入

使用工艺 G 功能组可以根据各加工过程调整动态参数。使用工艺 G 功能组 59 指令可以激活下列带有相应**数组下标**的通道专用与轴专用轨迹动态机床数据。

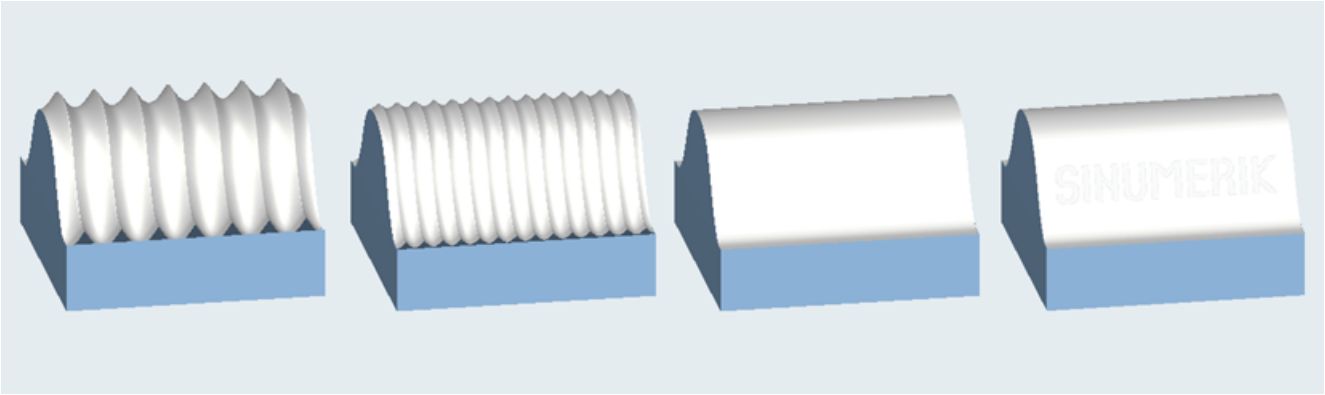


图 19-26　粗加工、初精整、精加工或精整

更多信息

轨迹插补的动态模式的更多信息，请参见

功能手册之基本功能。

Advanced Surface/Top Surface 的设置规定可访问以下网址 (https://support.industry.siemens.com/cs/document/109738423/sinumerik-828d-840d-sl-check-programs-for-advanced-surface-top-surface-settings?dti=0&dl=en&lc=zh-WW)。

取消加工方式选择

在取消 CYCLE832 后，G 功能组会在程序运行期间设为机床数据 MD20150 $MC_GCODE_RESET_VALUES[ ] 中定义的值。涉及到的 G 功能组有：1、 4、 10、 12、 21、 24、 30、 45、 59、 61。

轮廓公差与定向公差关闭（CTOL=-1， OTOL=-1）。

19.10.2　制造商循环 CUST_832.SPF

功能

如果因 CYCLE832.SPF 而与设置（G 功能）有所不同，您可以在制造商循环 CUST_832.SPF 中对该设置进行修改。

在 CUST_832.SPF 中定义了以下标记：

_M_INIT:	初始化 _FGREF
_M_NORM:	用于撤销 _OFF CYCLE832

_M_TOP_SURFACE_PRESICION:	精整 _PRECISION 和 _ORI_PRECISION 带 COMPSURF
_M_TOP_SURFACE_FINISH:	精加工 _FINISH 和 _ORI_FINISH 带 COMPSURF
_M_TOP_SURFACE_SEMIFINISH:	预精整 _SEMIFIN 和 _ORI_SEMIFIN 带 COMPSURF
_M_TOP_SURFACE_ROUGH:	粗加工 _ROUGH 和 _ORI_ROUGH 带 COMPSURF
_M_FINISH:	精加工 _FINISH 和 _ORI_FINISH 带 COMPCAD
_M_SEMIFINISH:	预精整 _SEMIFIN 和 _ORI_SEMIFIN 带 COMPCAD
_M_ROUGH:	粗加工 _ROUGH 和 _ORI_ROUGH 带 COMPCAD

在 CUST_832.SPF 中对下列 G 功能进行编程：

```
SOFT
COMPCAD / COMPSURF
G645
FIFOCTRL
UPATH
;FFWON
```

使用指令 CALCFIR 可显式激活 FIR 低通加加速度滤波器的计算：

```
;CALCFIR
```

根据加工方式编程以下 G 功能组 59 的 G 指令：

```
DYNNORM
DYNPREC
DYNFINISH
DYNSEMIFIN
DYNROUGH
```

在采用诸如 TRAORI 多轴转换的加工中，我们建议编写 FGREF()。此时在 CUST_832.SPF 中变量 _FGREF 的缺省值为 10。同样可以修改该数值。在循环 CYCLE832.SPF 中利用指令 FGREF（回转轴）将变量 _FGREF 写入参与 5 轴转换的回转轴上。在 G70/G700 有效时 _FGREF 的值在写入指令 FGREF 之前会被换算为英寸。

更多信息

有关指令 CALCFIR 的更多信息，请参见

- NC 编程手册
- 功能手册之进给轴和主轴

在 CYCLE832 中线性轴（几何轴）少于三根的情况

在精优曲面下，CYCLE832 内部将轴公差换算成了轮廓公差，因为 NC 只以轮廓公差 CTOL 工作，如：

CYCLE832(0.01,_Rough,1) → CTOL = 0.01 * √3 = 0.0173

NC 一直被设置为以 CTOL 和三个几何轴产生一个轮廓。若在通道中设置了少于三个几何轴，则必须调整 CUST_832 CTOL，以便在只有两个几何轴时也生成一个正确的轮廓。操作界面上会显示数值：

在 CYCLE832 中回转轴（定向轴）多于三根的情况

在 CYCLE832 中编程了 FGREF 时，定向转换（TRAORI）中最多允许两根回转轴。如果在每个通道中为定向转换定义了不止两根回转轴，可以按照以下句法在 CUST_832 中写入 FGREF 的值：

```
FGREF[AA]=$AA_FGREF[C]
```

C = 回转轴 1（轴由 CYCLE832 考虑）
AA = 回转轴 4

制造商循环 CUST_832.SPF 中提供用于分析“多轴编程是/否”的参数_MOTOL。

`_MOTOL=0`	无多轴编程
`_MOTOL=1`	多轴编程，带定向公差输入

通过以下设定数据可在应用多轴编程时在 CUST_832.SPF 中编程其他 G 功能：

SD55220 $SCS_FUNCTION_MASK_MILL_TOL_SET	高速设定功能设定 CYCLE832
位 1	输入定向公差（OTOL）为公差系数（CTOL）或直接输入

示例

CUST_832.SPF 中精加工标记的_MOTOL 应用：

```
_M_FINISH:
IF _MOTOL==1                ; 多轴编程
    ORISON
ELSE                        ; 无多轴编程
    ORISOF
ENDIF
...
```

在 CYCLE832 中多于一个坐标转换的情况

CYCLE832 可实现下列机床数据和系统变量的求值，以便由类型 TRAORI 和其回转轴来确定使用的坐标转换：

- $MC_TRAFO_TYPE 无坐标转换配置，
  $NT_NAME 无坐标转换配置 ⇒ 唯一
- $MC_TRAFO_TYPE 一个坐标转换配置，
  $NT_NAME 0 个、1 个或多个坐标转换配置 ⇒ $MC_TRAFO_TYPE 的坐标转换
- $MC_TRAFO_TYPE 两个坐标转换配置，
  $NT_NAME 0 个、1 个或多个坐标转换配置 ⇒ $MC_TRAFO_TYPE 的第二个坐标转换
- $MC_TRAFO_TYPE 无坐标转换配置，
  $NT_NAME 一个坐标转换配置 ⇒ 唯一
- $MC_TRAFO_TYPE 无坐标转换配置，
  $NT_NAME 多个坐标转换配置 ⇒ $NT_NAME 最后一个（最大下标）坐标转换

说明

若需要在模具制造时进行一个生效的坐标转换，此坐标转换必须在 CYCLE832 前被激活。

19.11 测量循环和测量功能

19.11.1 测量循环和测量功能，通用

前提条件

测量基本上需要使用两类电子测头：

- 测量工件的测头
- 测量刀具的测头

电子测头必须连接到 NCU 7x0 上。

更多信息

有关测头的数字量输入和输出的更多信息，请参见

NCU 设备手册。

说明

在 NCU 设备手册中电子测头简称为测头。

设置开关动作

可以通过下列通用机床数据设置所连接测头的极性：

MD13200 $MN_MEAS_PROBE_LOW_ACTIVE	测头极性反转
= 0	测头非偏转状态为 0 V，缺省值 测头偏转状态为 24 V
= 1	测头非偏转状态为 24 V 测头偏转状态为 0 V

MD13210 $MN_MEAS_TYPE	分布式驱动上的测量方式
= 0	缺省值

检查测头功能

可以通过手动偏转测头与控制下列 PLC 接口信号来检查测头的开关功能：

DB 10								
	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
DBB107							操作测头	
							测头 2	测头 1

例如可以使用 NC 检查程序借助下列 NC 命令来检查测头的开关动作与测量值的传输：

MEAS	测量，带剩余行程删除
$AC_MEA[n]	检查开关动作， n = 测量输入编号
$AA_MW[轴名称]	在工件坐标中轴的测量值
$AA_MM[轴名称]	在机床坐标中轴的测量值

检查程序示例

程序代码	注释
%_N_PRUEF_MESSTASTER_MPF	
$PATH=/_N_MPF_DIR	; 测头连接的检查程序
N00 DEF INT MTSIGNAL	; 开关状态检查标志
N05 G17 G54 T="3D_Taster" D1	; 选择用于测头的刀具几何值
N10 M06	; 激活刀具
N15 G0 G90 X0 F150	; 起始位置和测量速度
N20 MEAS=1 G1 X100	; X 中的测量输入端 1 处的测量
N30 MTSIGNAL=$AC_MEA[1]	; 测量输入 1 上进行了开关动作，是/否
N35 IF MTSIGNAL == 0 GOTOF _FEHL1	; 信号分析
N40 R1=$AA_MM[X]	; 将机床坐标的测量值保存到 R1
N45 R2=$AA_MW[X]	; 将工件坐标的测量值保存到 R2
N50 M0	; 检查 R1/R2 中的测量值
N55 M30	
N60 _FEHL1: MSG（“测头未开关！”）	
N65 M0	
N70 M30	

测量循环的通用功能设置

MD51740 $MNS_MEA_FUNCTION_MASK	测量循环的功能屏幕
位 0	工件，校准状态的监控
位 1	工件，测头在进给轴中基于球体范围的长度参考
= 0	刀具长度 L1，取决于测头球体中心的位置
= 1	刀具长度 L1，取决于测头球体范围
位 2	工件，对刀具的补偿，考虑刀架
位 4	工件，测头固定位于机床上（不在主轴中）
位 5	工件，在未设置精零偏的情况下，粗零偏作为补偿
位 6	测量循环中的测量程序段采用进给倍率 100%
位 7	工件，在未生效的 NPV 中激活接收，仅运行方式 JOG，设置边沿
位 14	工件，同时测量功能激活
位 15	工件，测量程序段以轨迹特性 G60 准停运行
位 16	刀具，车削工艺，考虑刀具

说明

该机床数据修改后，必须重新补偿（标定）测头。

MD52740 $MCS_MEA_FUNCTION_MASK	测量循环的功能屏幕
位 0	工件测头连接在测头输入 2 上
= 0	CNC 测量输入 1 上的工件测头，激活 - 缺省设置
= 1	CNC 测量输入 2 上的工件测头，激活
位 1	工件，车削工艺，在第三几何轴（Y）中的测量
位 2	计算校正角度，带/不带激活的定向转变
位 16	刀具测头连接在测头输入 2 上
= 0	CNC 测量输入 1 上的刀具测头，激活
= 1	CNC 测量输入 2 上的工件测头，激活 - 缺省设置
位 17	车削刀具测量，刀沿位置的重新编码

SD54740 $SNS_MEA_FUNCTION_MASK	测量循环的功能屏幕
位 0	工件，在超出 TSA、TDIF 时重复测量
位 1	工件，在超出 TSA、TDIF 时以 M0 重复测量
位 2	工件，在超出 TUL、TLL 时停止在 M0
位 3	工件，将经校准的测头测球半径接收至刀具数据
位 4	工件，校准与测量之间的平面可以不同
位 16	刀具，在超出 TDIF、TSA 时重复测量
位 17	刀具，在超出 TDIF、TSA 时通过停止在 M0 重复测量
位 19	刀具，铣削，在最后一次探测时降低主轴转速

SD55740 $SCS_MEA_FUNCTION_MASK	测量循环的功能屏幕
位 0	工件，中间定位时的碰撞监控
位 1	工件，主轴位置与围绕进给轴的坐标旋转的耦合
位 2	工件，在主轴位置耦合中将定位方向反转
位 3	工件，在测头未接通时不重复
位 4	工件，在每次测量的第 1 次探测时采用快速测量进给率

位 5	工件，测量后快速回退
位 6	工件，在主轴定位期间将测头取消激活
位 7	运动系统，定标以回转轴矢量的输入值为基础
位 8	运动系统，通过生效的回转（TCARR）或生效的 TRAORI 进行测量
位 9	运动系统，回转轴矢量的用户专用的定标
位 10	借助参考完整测量运动系统
位 11	运动系统，测量校准球时的测量点数量
位 15	工件，以环中心为起点校准半径，仅运行方式 JOG
位 16	刀具，中间定位时的碰撞监控
位 17	刀具，在测头未接通时不重复
位 19	刀具，测量后快速回退
位 28	JOG 测量的标准日志持续增加（添加）
位 29	JOG 测量的标准日志采用表格形式

19.11.2 制造商与用户循环 CUST_MEACYC.SPF

功能

循环“CUST_MEACYC.SPF”为测量循环功能的一部分，在每个测量循环中都会在执行测量任务前后调用该文件。它对于 JOG 方式下的测量和 AUTO 方式下的测量同样有效。

借助 CUST_MEACYC.SPF，可以编程并执行测量前后必要的步骤（例如激活/禁用测头）。

说明

执行修改前，将 CUST_MEACYC 循环从目录“NC 数据/循环/标准循环”中复制到目录“NC 数据/循环/制造商循环”或
目录“NC 数据/循环/用户循环”中。只有在这些目录中，CUST_MEACYC 的修改才能生效。

CUST_MEACYC 结构程序

AUTO 方式下的步骤，以钻孔测量为例，带测头类型 712 的 CYCLE977。

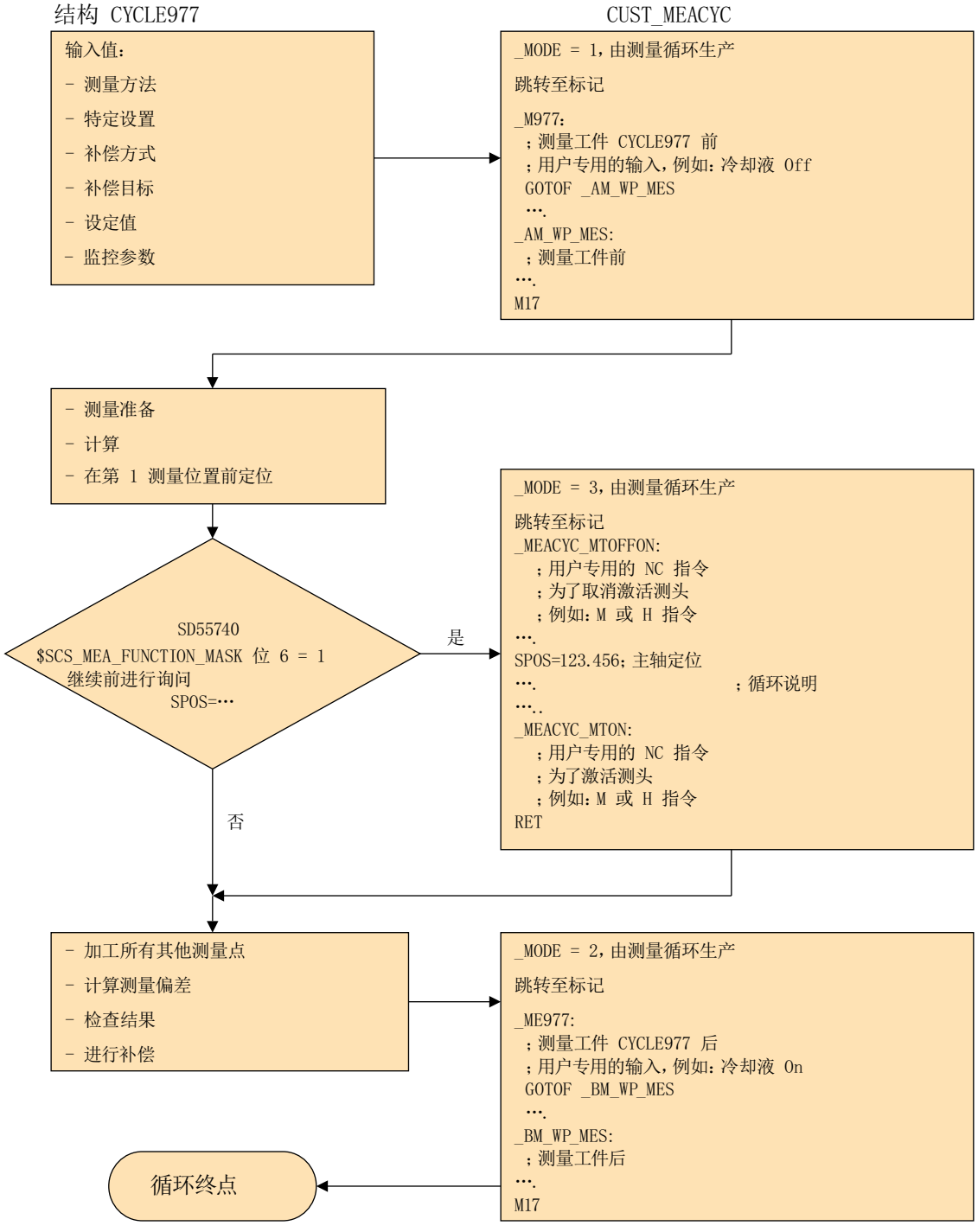


图 19-27　结构：CYCLE977 / CUST_MEACYC

说明

标注主轴 SPOS 指令

如需在循环 CUST_MEACYC.SPF 中标注主轴 SPOS 指令，则应事先通过位 1 = 1 将 SD55740 $SCS_MEA_FUNCTION_MASK 中的循环设置设为有效。

如何在循环 CUST_MEACYC.SPF 中标注一个主轴 SPOS 指令，请参考以下示例：

示例

```
;
;*#1 主轴预定位的用户示例，无耦合
;*##1 在主轴和坐标系之间生效!
;*#1 该示例可根据需要删除!
;*#2 User example for spindel positioning, if you do not have a coupling
;*##2 between the spindel and the coordinate system!
;*#2 If this lines are not necessary, the user can delete this example!
;
; cycle is calling from the AUTOMATIC-mode AND the spindel is positionable
IF NOT(_OVI[14]) AND NOT(_SPOS_MOD)
  ; no coupling between the spindel and the coordinate system!
  ; SD55740, Bit 1 = 0
  IF ($SCS_MEA_FUNCTION_MASK B_AND 'B10')==0
  ; SPOS=0 ; <===== 标注
  ENDIF
ENDIF
M17
;
```

19.11.3 在 JOG 运行方式下测量。

前提条件

已执行了之前章节“测量循环和测量功能，通用 (页 565)”中描述的设置。

测量工件

用于铣削工艺：

- 将测头换入刀具主轴。
- 在刀具列表中选择类型 7xx（测头）。
- 将测头作为当前 NC 通道中的刀具激活。

刀具测量

进行刀具测量时必须在机床空间内加装相应的测头，确保它能够到达主轴上的刀具处。

刀具测量支持下列刀具类型：

- 铣削工艺：刀具类型 1xx 和 2xx
- 车削工艺：刀具类型 5xx, 1xx, 2xx

可以测量上述刀具类型的刀具长度和刀具半径。

设置通用数据

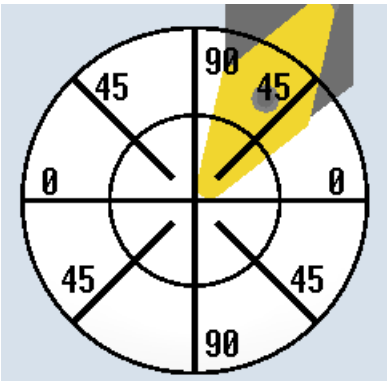
SD54782 $SNS_J_MEA_FUNCTION_MASK_TOOL	输入屏幕“在 JOG 运行方式下测量，刀具测量”的设置。
位 2 = 1	激活功能“自动刀具测量使能”。
位 3 = 1	选择刀具测头标定数据组，使能
位 10 = 1	单片刀沿选项
位 11 = 1	选择在平面内标定测头时的主轴是否旋转

MD51784 $MNS_J_MEA_T_PROBE_APPR_AX_DIR [i]	
在 JOG 方式下测量：选择在平面内逼近刀具测头的方向	
-2	负向第 2 测量轴
-1	负向第 1 测量轴（缺省设置）
0	正向第 1 测量轴
1	
2	正向第 2 测量轴

MD52750 $MCS_J_MEA_FIXPOINT	相对固定点测量的 Z 值
毫米	

MD52750 可以保存“测量：固定点”Z 轴位置。固定点是在铣削工艺的 JOG 方式中开展手动刀具测量的参考点。该值可以通过输入屏幕“标定：固定点”设置。

MD52751 $MCS_J_MEA_MAGN_GLAS_POS	刀具测量的放大位置
毫米	
[0]	第 1 轴的定位
[1]	第 2 轴的定位



机床上有放大镜时，便可以利用它来确定刀具几何尺寸。此时，SINUMERIK Operate 通过已知刀架参考点的位置以及放大镜的十字光标计算出刀具补偿数据。

检查并设置通用机床数据

MD11450 $MN_SEARCH_RUN_MODE	编程搜索模式。
位 1 = 1	输出动作程序段后自动启动 ASUB（参见 MD11620 $MN_PROG_EVENT_NAME）。当 ASUB 结束后才输出报警 10208。

MD11604 $MN_ASUP_START_PRIO_LEVEL	优先级从 ASUP_START_MASK 起有效。
= 1 - 64	用于 ASUP_START_MASK 的优先级。

设置通道专用机床数据

MD20050 $MC_AXCONF_GEOAX_ASIGN_TAB[ ]	几何轴和对应的通道轴
[0]	前提条件是，在 JOG 运行方式下测量时所有几何轴可用。特别是 XYZ。
[1]	
[2]	

MD20110 $MC_RESET_MODE_MASK	确定复位/零件程序结束后控制器的初始设置。
= 4045H	最小值
位 0 = 1	确定上电和复位后控制器的初始设置。
位 2 = 1	
位 6 = 1	
位 14 = 1	

MD20112 $MC_START_MODE_MASK	确定零件程序启动后控制器的初始设置。
= 400H	最小值
位 6 = 0	确定 NC 启动后控制器的初始设置。

MD20310 $MC_TOOL_MANAGEMENT_MASK	激活刀具管理功能。
= 4002H	最小值
位 1 = 1	刀具管理和监控功能有效。
位 14 = 1	在复位和启动时自动换刀。

MD28082 $MC_SYSTEM_FRAME_MASK	系统框架（SRAM）。
位 0 = 0	不设置用于实际值设定和对刀的系统框架。
位 0 = 1	设置用于实际值设定和对刀的系统框架。

只有将通道专用的机床数据 MD28082 $MC_SYSTEM_FRAME_MASK 的位 0 设为 0 后，即设置用于实际值设定和对刀的系统框架，下列机床数据才生效。

MD24006 $MC_CHSFRAME_RESET_MASK	复位后有效的系统框架。
位 0 = 1	实际值设定时系统框架的复位特性。

MD24007 $MC_CHSFRAME_RESET_CLEAR_MASK	复位时删除系统框架。
位 0 = 0	复位时系统框架的删除特性。

说明

如果满足了本章中所述的前提条件，并设置或控制了机床设定数据，就可以在 JOG 运行方式下使用工件测头在铣床上测量工件！

在 JOG 运行方式下可以使用刀具测头在铣床或车床上测量刀具！

在后续章节中将说明，可以开展哪些设置，使测量过程符合机床上的特殊要求。

19.11.3.1 在铣床上测量工件

您可以通过以下通道专用的通用机床数据、循环设定数据，来根据特定要求调整“加工”区中的测量。

通用配置机床数据

SD55630 $SCS_MEA_FEED_MEASURE	JOG 和 AUTO 方式中“工件测量”用进给率
300	缺省值

说明

“测量工件”时的测量进给率

所有的测量循环都使用补偿（标定刀具测头）之后保存在 SD54611 中的值作为测量进给率。每一个标定字段[n]可以有不同的测量进给率。

在标定测头时，要么使用来自“SD55630 $SCS_MEA_FEED_MEASURE”设置的测量进给率，要么使用在校准时输入屏幕窗口中的测量进给率。为此必须在 SD54760 $SNS_MEA_FUNCTION_MASK_PIECE 中将位 4 置 1。

MD52207 $MCS_AXIS_USAGE_ATTRIB[n]	轴的属性 [n]通道轴号
位 6	请求旋转轴作为测量的补偿目标
= 0	缺省值
= 1	在窗口"JOG 测量"中将旋转轴作为补偿目标显示
位 9	结合测量循环影响主主轴的 SPOS 特性
= 0	不影响刀具主轴的定位特性
= 1	测量循环不执行刀具主轴的定位（无 SPOS）

说明

在刀具主轴无 SPOS 能力的情况下，必须设置位 9 = 1。

MD51740 $MS_MEA_FUNCTION_MASK	测量循环、工件测量和刀具测量的功能屏幕
位 7	工件，在未生效的 NPV 中激活接收，仅运行方式 JOG，设置边沿
= 0	在运行方式 JOG 下，使用功能"设置边沿"且会将测量结果补偿到未生效的零点偏移（目标框架）中，对于目标框架中的几何轴，只有测量的轴会被写入该测量结果。 对于未测量的几何轴，不会从生效的零点偏移（源框架）中接收值。（兼容性） 对于所有非几何轴（回转轴和辅助轴），镜像、旋转和转换会从生效的零点偏移（源框架）接收到所选择的未生效的零点偏移（目标框架）中。（兼容性）
= 1	在运行方式 JOG 下，使用功能"设置边沿"且会将测量结果补偿到未生效的零点偏移（目标框架）中，目标框架中的测量轴会被写入该测量结果。 对于其他所有轴（几何轴、回转轴和辅助轴），镜像、旋转和转换会从生效的零点偏移（源框架）接收到所选择的未生效的零点偏移（目标框架）中。

该描述适用于使用电子测头的自动测量以及使用加工刀具的"对刀"功能，参见章节"在 JOG 方式中不用电子探头进行测量 (页 585)"。

MD51751 $MNS_J_MEA_M_DIST_MANUELL	测量点前后的测量路径，以毫米为单位
= 10	缺省值

MD51757 $MNS_J_MEA_COLL_MONIT_FEED	碰撞监控生效时工作平面中的进给率，以毫米/分钟为单位
= 1000	缺省值

MD51758 $MNS_J_MEA_COLL_MONIT_POS_FEED	碰撞监控生效时进给轴中的进给率，以毫米/分钟为单位
= 1000	缺省值

MD51770 $MNS_J_MEA_CAL_RING_DIAM[n]	校准环直径的预设，单位毫米，专用于校准数据组
= -1	缺省值

MD51772 $MNS_J_MEA_CAL_HEIGHT_FEEDAX[n]	进给轴中的校准高度的预设，单位毫米，专用于校准数据组
= -99999	缺省值

通用循环设定数据

SD54780 $SNS_J_MEA_FUNCTION_MASK_PIECE	用于在 JOG 运行方式下操作区“加工”中的测量的输入屏幕的配置
= 512	缺省值
位 2 = 1	激活采用电子测头的测量
位 3 = 1	使能测头校准数据的选项
位 6 = 1	使能基准参考（SETFRAME）中的零偏补偿选项
位 7 = 1	使能通道专用基准框架中的零偏补偿选项
位 8 = 1	使能全局基准框架中的零偏补偿选项
位 9 = 1	使能可设置框架中的零偏补偿选项
位 16 = 1	带主轴变换的工件测量选项
位 17 = 1	在切换方向上校准工件测头选项
位 28 = 1	使能刀具长度调整选项

19.11.3.2 在铣床上测量刀具

JOG 和 AUTO 方式中“刀具测量”用进给率

SD55628 $SCS_MEA_TP_FEED_MEASURE	校准刀具测量头或主轴静止时测量刀具采用的进给率。
= 300	缺省值

说明

测量刀具时的进给率

所有的测量循环都使用补偿（标定）刀具测头之后保存在 SD54636 或 SD54651 中的进给率。每一个标定字段[n]可以有不同的测量进给率。

在标定测头时，要么使用来自“SD55628 SCS_MEA_TP_FEED_MEASURE ”设置的测量进给率，要么使用在“标定”屏幕窗口中输入的测量进给率。为此必须将“SD54762 $SNS_MEA_FUNCTION_MASK_TOOL”的位 4 设为 1。

MD51757 $MNS_J_MEA_COLL_MONIT_FEED	碰撞监控生效时工作平面中的进给率，以毫米/分钟为单位
= 1000	缺省值

MD51758 $MNS_J_MEA_COLL_MONIT_POS_FEED	碰撞监控生效时进给轴中的进给率，以毫米/分钟为单位
= 1000	缺省值

在下面的设定数据中，下标 [k] 代表测头当前数据组的编号（即“测头编号 -1”）。

SD54633 $SNS_MEA_TP_TYPE[k]	指定测头类型：方形/盘形
= 0	兼容性（测量循环：方形，在车削表面上显示球形，铣削表面上显示盘形）
= 101	盘形测头，位于 XY 平面，工件平面 G17
= 201	盘形测头，位于 ZX 平面，工件平面 G18
= 301	盘形测头，位于 YZ 平面，工件平面 G19
= 2	球形
= 3	方形

使用下列设定数据能够确定，可以在哪根轴的哪个方向上对刀具测头进行标定。

SD54632 $SNS_MEA_T_PROBE_ALLOW_AX_DIR[k]	指定参与标定的轴和方向。
= 133	缺省值

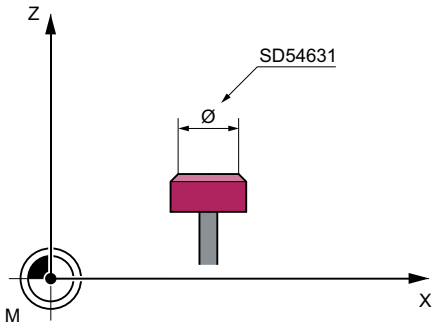
小数位		
个位	第 1 轴	
	= 0	该轴无标定
	= 1	仅有负方向
	= 2	仅有正方向
	= 3	两个方向
十位	第 2 轴	
	= 0	该轴无标定
	= 1	仅有负方向
	= 2	仅有正方向
	= 3	两个方向
百位	第 3 轴	
	= 0	该轴无标定
	= 1	仅有负方向
	= 2	仅有正方向
	= 3	两个方向

示例

如果通用设定数据 SD54632 $SNS_MEA_T_PROBE_ALLOW_AX_DIR[k] 的值为 123，则在 G17 平面中按如下方式标定刀具测头：

- X 轴两个方向
- 仅 Y 轴仅正方向
- 仅 Z 轴负方向

SD54631 $SNS_MEA_TP_EDGE_DISK_SIZE[k]	指定刀具测头的有效直径，进行长度测量。
= 0	缺省值



刀具测头的通用循环机床数据/设定数据

MD51780 $MNS_J_MEA_T_PROBE_DIAM_RAD[k]	指定刀具测头的有效直径，用于半径测量。
= 0	缺省值

SD54634 $SNS_MEA_TP_CAL_MEASURE_DEPTH[n]	刀具测头上边沿和铣刀下边沿之间的距离
= 2	缺省值

MD51752 $MNS_J_MEA_M_DIST_TOOL_LENGTH	指定在 JOG 方式中用旋转主轴进行刀具长度测量时的测量行程。
= 2	缺省值

MD51753 $MNS_J_MEA_M_DIST_TOOL_RADIUS	指定在 JOG 方式中用旋转主轴进行刀具半径测量时的测量行程。
= 1	缺省值

MD51786 $MNS_J_MEA_T_PROBE_MEASURE_DIST	指定在 JOG 方式中“标定刀具测头”和“用静止主轴测量刀具”时的测量行程。
= 10	缺省值

用旋转主轴进行测量时的通用循环设定数据

SD54670 $SNS_MEA_CM_MAX_PERI_SPEED[0]	指定允许的待测刀具的最大圆周速度。
= 100	缺省值

SD54671 $SNS_MEA_CM_MAX_REVOLUTIONS[0]	指定允许的待测刀具的最大转速。一旦超过该转速，则转速自动减少。
= 1000	缺省值

SD54672 $SNS_MEA_CM_MAX_FEEDRATE[0]	指定允许的待测刀具接触测头时的最大进给率。
= 20	缺省值

SD54673 SNS_MEA_CM_MIN_FEEDRATE[0]	指定待测刀具首次接触测头的最小进给率。这样可以避免刀具半径较大时首次接触测头的进给率太小。
= 1	缺省值

SD54674 $SNS_MEA_CM_SPIND_ROT_DIR[0]	指定测量刀具时主轴的旋转方向。
= 4	主轴旋转方向和 M4 一样（缺省值）

说明

如果调用测量循环时主轴已经旋转，则会保持该旋转方向不变，不管上述数据的设置如何。

SD54675 $SNS_MEA_CM_FEEDFACTOR_1[0]	指定进给系数 1
= 10	缺省值

= 0	仅使用由循环计算出的进给率进行一次接触。但进给率至少为 “SD54673[0] $SNS_MEA_CM_MIN_FEEDRATE”指定的最小进给率。
>= 1	使用进给率第一次接触。但进给率起码是 “SD54673[0] $SNS_MEA_CM_MIN_FEEDRATE”x “SD54675[0] $SNS_MEA_CM_FEEDFACTOR_1”的值。

SD54676 $SNS_MEA_CM_FEEDFACTOR_2[0]	指定进给系数 2
= 0	使用由测量循环计算出的进给率进行二次接触。仅在 SD54673[0] $SNS_MEA_CM_FEEDFACTOR_1 > 0 时有效，缺省值
>= 1	使用“SD54673[0] $SNS_MEA_CM_MIN_FEEDRATE”x“进给系数 2”计算出的进给率进行第二次接触。 第三次接触使用计算出的进给率。

说明

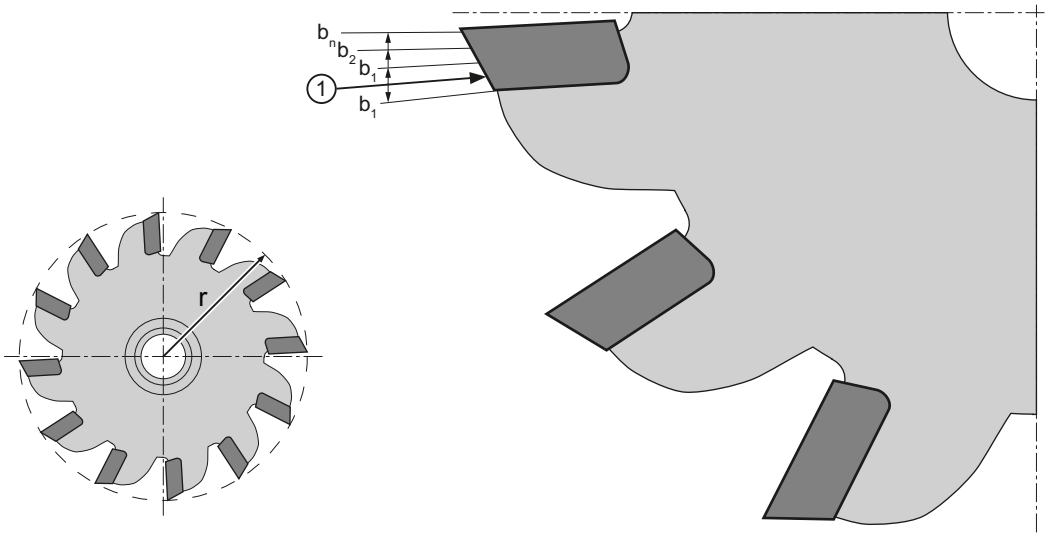
进给系数 2 应该小于进给系数 1。

SD54677 $SNS_MEA_CM_MEASURING_ACCURACY[0]	指定要求的测量精度。 测量精度指刀具最后一次接触测头时的精度。
= 0,005	缺省值

SD54692 $SNS_MEA_T_CIRCULAR_ARC_DIST	测量单片刀沿的半径时的弧长
= 0.25 *)	缺省值

SD54693 $SNS_MEA_T_MAX_STEPS	测量单片刀沿的半径时的接触次数
= 10 *)	缺省值

*) 值为 0 时，系统不查找最长刀沿上的最高点，而是采用接触旋转主轴得出的测量值。



r 有效刀具半径

b 弧长 = SD54692

n 最大接触次数 = SD54693

① 第 1 测量点

图 19-28 单片刀沿测量

使用旋转主轴进行测量时借助补偿列表进行补偿的通用循环设定数据

SD54691 $SNS_MEA_T_PROBE_OFFSET	激活测量结果补偿
= 0	不指定，缺省值
= 1	指定循环内部补偿。仅当 SD54689 $SNS_MEA_T_PROBE_MANUFACTURER>0 时有效。
= 2	指定通过用户自定义补偿表进行补偿。

SD54689 $SNS_MEA_T_PROBE_MANUFACTURER	激活一些刀具测头型号预先制订的补偿表（用户专用）
= 0	不指定，缺省值
= 1	TT130 (Heidenhain)
= 2	TS27R (Renishaw)

19.11.3.3 在车床上测量刀具

车削工艺中刀具测量用进给率

SD55628 $SCS_MEA_TP_FEED_MEASURE	校准刀具测量头或主轴静止时测量刀具采用的进给率。
= 300	缺省值

说明

测量刀具时的进给率

所有的测量循环都使用补偿（标定）刀具测头之后保存在 SD54636 或 SD54651 中的进给率。每一个标定字段[n]可以有不同的测量进给率。

在标定测头时，要么使用 SD55628 设置的测量进给率，要么使用在“标定”屏幕窗口中输入的测量进给率。为此必须将“SD54762 $SNS_MEA_FUNCTION_MASK_TOOL”的位 4 设为 1。

通用循环机床数据/设定数据，用于指定测量行程/测量进给率

SD42950 $SC_TOOL_LENGTH_TYPE	指定刀具长度补偿，和刀具类型无关。
= 2	刀具长度分量的分配与实际刀具类型无关，始终和车刀/磨具上的分配的相同。 该设置推荐用于铣刀和钻头的测量。

MD51786 $MNS_J_MEA_T_PROBE_MEASURE_DIST	指定“标定刀具测头”和“用静止主轴测量刀具”时的测量行程。
= 10	缺省值

SD55628 $SCS_MEA_TP_FEED_MEASURE	指定主轴静止时标定刀具测头的进给率。
= 300	缺省值

标定刀具测头

可通过以下通用循环机床数据设置显示软键“标定测头”的最低访问级别。前提条件是已通过设定数据 SD54782（位 2=1）使能了自动刀具测量。

MD51070 $MNS_ACCESS_CAL_TOOL_PROBE		指定访问“标定测头”的访问级别。
= 7	访问权限：访问级别 7（钥匙开关 0）	
= 4	读取访问权限：访问级别 4（钥匙开关 3），缺省值	
= 3	写访问权限：访问级别 3（用户）	

19.11.3.4 在 JOG 方式中不用电子探头进行测量

测量工件

操作方式 JOG 中的手动测量功能要求主轴运转，同时：

- 在接收测量值前通道必须处于复位状态。
- 主轴可以通过 T,S,M 菜单或者 DB3x.DBB30 中的位移动或定位。
- 在固定转速时使用工艺功能 DB3x.DBB30 需要满足以下前提条件：

机床数据的设置		
MD10709 $MN_PROG_SD_POWERON_INIT_TAB [0] = 0		指定需要初始化的设定数据
MD35035 $MA_SPIND_FUNCTION_MASK	Bit 4 = 0 Bit 5 = 1	主轴功能
SD43200 $SA_SPIND_S		指定转速

说明

上电后，“SD43200 $SA_SPIND_S”中的设置保持不变。

该设置不会再影响通过“SD41200 $SN_JOG_SPIND_SET_VELO”设置的手动模式主轴速度。

19.11.4 在 AUTOMATIK 运行方式下测量。

前提条件

软件选件

“在自动运行方式下测量”的功能需要下列软件选件：“测量循环”

已执行了之前章节“测量循环和测量功能，通用 (页 565)”中描述的设置。

测量工件

铣削工艺中的顺序：

1. 在刀具列表中选择类型 7xx（测头）。
2. 将测头换入刀具主轴。
3. 在当前的 NC 通道中激活探头。

车削工艺中的顺序：

1. 选择刀具“类型 580 （3D 车削测头）”。
2. 在当前的 NC 通道中激活刀具。

刀具测量

进行刀具测量时必须在机床空间内加装相应的测头，确保它能够到达主轴上的刀具处。

刀具测量支持下列刀具类型：

- 铣削工艺：刀具类型 1xx 和 2xx
- 车削工艺：刀具类型 5xx, 1xx, 2xx

可以测量上述刀具类型的刀具长度和刀具半径。

通道专用循环设定数据

使用以下通道专用循环设定数据，可以在“程序”操作区中按特殊要求调整**工件测量和刀具测量**，但通常这两个测量不需要进行任何更改。

SD55613 $SCS_MEA_RESULT_DISPLAY	指定测量结果的图形画面
= 0	没有测量结果的图形画面（缺省值）
= 1	有测量结果的图形画面，持续 8 秒

= 3	测量循环在内部机床数据上停止时，测量结果的图形画面保持不变！ 按下“NC 启动”继续运行时，图形画面被撤销。
= 4	仅在循环报警 61303、61304、61305、61306 时出现测量结果的图形画面。 按下“NC 启动”继续运行时，图形画面被撤销。

SD55614 $SCS_MEA_RESULT_MRD	测量结果图形显示 MRD 的程序控制的设置
= 0	关闭测量结果图形显示
= 1	打开测量结果图形显示

SD55623 $SCS_MEA_EMPIRIC_VALUE[n]	经验值
= 0	缺省值

SD55618 $SCS_MEA_SIM_ENABLE	测量循环模拟
= 0	在测量循环和“SINUMERIK Operate”组合使用、开展 ShopMill 模拟或 ShopTurn 模拟时，会跳过测量循环（缺省值）。
= 1	默认设置： 测量循环和“SINUMERIK Operate”组合使用、开展 ShopMill 模拟或 ShopTurn 模拟时，会执行测量循环，但不进行补偿，也不进行记录。测量结果没有图形画面。

更多信息

有关 SD55618 的更多信息，请参见

编程手册之测量循环。

配置程序编辑器中测量循环的输入屏幕

通过以下设定数据可以配置自动方式下刀具/工件测量的显示画面和补偿方式：

- SD54760 $SNS_MEA_FUNCTION_MASK_PIECE 测量工件：测量循环位于铣削区域，参见章节“测量工件，概述 (页 588)”
- SD54762 $SNS_MEA_FUNCTION_MASK_TOOL 测量刀具：测量循环位于铣削区域，参见章节“在铣床上测量刀具 (页 592)”

说明

如果满足了本章中所述的前提条件，设置并检查了机床设定数据，就可以在自动方式下使用工件/刀具测头在机床上进行测量了！

如果希望调整车床上的工件测量过程，则需要进一步地进行设置。相关设置请参见章节“在车床上测量工件 (页 599)”。

如果希望调整刀具测量过程，同样需要进一步设置：相关设置请参见章节“在车床上测量刀具 (页 600)”或“在铣床上测量刀具 (页 592)”。

在后续章节中将说明，可以开展哪些设置，使测量过程符合机床上的特殊要求。

19.11.4.1 测量工件，概述

使用下面的通用和通道专用循环设定数据，可以在“程序”操作区中调整工件测量过程，使其符合特殊要求。

前提条件

已执行章节 “测量循环和测量功能，通用 (页 565)” 中的设置。

设置

通用循环设定数据

SD54760 $SNS_MEA_FUNCTION_MASK_PIECE	输入屏幕的设置，在自动运行方式下测量， 工件测量。
位 1	显示选择软键“3D 测量”
位 3	选择“使能测头校准数据组”
位 4	选择“校准测量进给率输入” 1)
位 6	激活选项“零偏补偿计入基准参考（SETFRAME）”
位 7	激活选项“零偏补偿计入通道专用的基准框架”
位 8	激活选项“零偏补偿计入全局基准框架”
位 9	激活选项“零偏补偿计入可设定的框架”
位 10	激活选项“零偏补偿：精偏和粗偏”
位 11	几何和磨损刀具补偿选项

位 12	刀具补偿未取反和取反选项
位 13	刀具补偿 L1、R 或 L1、L2、L3 R 选项
位 14	刀具补偿零点补偿（_TZL）选项
位 15	刀具补偿尺寸差值检查（_TDIF）选项
位 16	带主轴旋转的工件测量选项
位 17	在切换方向上标定工件测头选项
位 18	测量次数（_NMSP）选项
位 19	选项“带平均值计算的补偿”(_TMV) [1)]
位 20	经验值（_EVNUM）选项
位 21	设置补偿和选项
位 22	在未知中心点和已知中心点上标定的选项
位 24	带/不带位置偏移标定选项
位 25	使能主轴角度测量时的零点补偿选项
位 26	不使能刀具补偿选项
位 27	不使能线性矢量在运动系统测量中的公差选项
位 28	激活选项“刀具补偿，调整长度”

通道专用循环设定数据

SD55630 $SCS_MEA_FEED_MEASURE	指定测量进给率 [毫米/分钟]
= 300	指定补偿（标定）工件测头时的测量进给率

说明

测量工件时的测量进给率

所有的测量循环都使用补偿（标定）刀具测头之后保存在“SD54611 $SNS_MEA_WP_FEED[n]”中的值作为测量进给率。

每一个标定字段[n]可以有不同的测量进给率。

在标定测头时，要么使用来自“SD55630 $SCS_MEA_FEED_MEASURE”的测量进给率，要么使用“标定”屏幕窗口中输入的测量进给率。为此必须将“SD54760 MEA_FUNCTION_MASK_PIECE”的位 4 设为 1。

SD55632 $SCS_MEA_FEED_RAPID_IN_PERCENT	指定减速百分比。
= 50	在不带碰撞监控的循环内部中间定位时，快速移动速度的减速百分比（缺省值）。 通道专用循环设定数据“SD55600 SCS_MEA_COLLISION_MONITORING”必须设为0。

SD55634 $SCS_MEA_FEED_PLANE_VALUE	指定碰撞监控有效时工作平面中的中间定位进给率。
= 1000	缺省值

SD55636 $SCS_MEA_FEED_FEEDAX_VALUE	指定碰撞监控有效时进给轴中的中间定位进给率。
= 1000	缺省值

SD55638 $SCS_MEA_FEED_FAST_MEASURE	指定快速测量进给率。
= 900	缺省值

SD55640 $SCS_MEA_FEED_CIRCLE	指定沿圆弧轨迹的进给率。
= 1000	缺省值

SD55642 $SCS_MEA_EDGE_SAVE_ANG	指定“测量拐角”(CYCLE961)时的安全角度
= 0	度

该角度与工件参考边和正方向上激活的坐标系中的第 1 轴之间的角度用户说明共同生效并涉及到自动运行中的角度测量（CYCLE961）。

用户设定的角度“拐角测量”设置屏幕中的符号为“α0”。兼容程序中参见符号_STA1。

提示：

兼容程序中需要设置 $SCS_MEA_EDGE_SAVE_ANG=10。

19.11.4.2 在铣床上测量工件

在“程序”操作区中，您可以通过通道专用的循环设定数据来自定义测量，使其符合您的要求，

设置

SD55625 $SCS_MEA_AVERAGE_VALUE[n]	平均值的数量。
= 0	缺省值

通过带有“测量头补偿”（校准）功能的测量循环描述下列循环设定数据。 在此无需设置用户参数。 当然可以在测量头校准过程后检查这些数值并计算测量头的质量，例如在位置偏差时没有数值 > 0.1 毫米。 另外可以重新机械校准测量头。

校准时的测量进给率会被应用于后续的测量循环。

说明

请注意测量头制造商的提示说明。

SD54600 $SNS_MEA_WP_BALL_DIAM[n]	工件测量头球的有效直径。
SD54601 $SNS_MEA_WP_TRIG_MINUS_DIR_AX1[n]	负方向触发点， 平面中的测量轴 1 。
SD54602 $SNS_MEA_WP_TRIG_PLUS_DIR_AX1[n]	正方向触发点， 平面中的测量轴 1 。
SD54603 $SNS_MEA_WP_TRIG_MINUS_DIR_AX2[n]	负方向触发点， 平面中的测量轴 2 。
SD54604 $SNS_MEA_WP_TRIG_PLUS_DIR_AX2[n]	正方向触发点， 平面中的测量轴 2 。
SD54605 $SNS_MEA_WP_TRIG_MINUS_DIR_AX3[n]	负方向触发点， 刀具方向上的测量轴 3。
SD54606 $SNS_MEA_WP_TRIG_PLUS_DIR_AX3[n]	正方向触发点， 刀具反方向上的测量轴 3 。 缺省情况下 = 0。
SD54607 $SNS_MEA_WP_POS_DEV_AX1[n]	位置偏差， 平面中的测量轴 1 。

SD54608 $SNS_MEA_WP_POS_DEV_AX2[n]	位置偏差， 平面中的测量轴 2 。
SD54609 $SNS_MEA_WP_STATUS_RT	轴定位的标定状态
SD54610 $SNS_MEA_WP_STATUS_GEN[n]	校准状态
SD54611 $SNS_MEA_WP_FEED[n]	校准时的测量进给率

19.11.4.3 在铣床上测量刀具

设置

SD54762 $SNS_MEA_FUNCTION_MASK_TOOL	输入屏幕的设置，在自动运行方式下测量， 刀具测量
位 3	输入屏幕的设置，在自动运行方式下测量， 刀具测量；使能刀具测量头标定数据组选项
位 4	校准测量进给率输入 [1]选项
位 5	接触时的进给输入和主轴转速选项
位 7	MCS 和 WCS 中测量选项
位 8	绝对测量和增量测量选项
位 9	几何和磨损刀具补偿选项
位 10	单片刀沿选项
位 11	选择在平面内标定测头时的主轴是否旋转
位 12	测量次数（_NMSP）选项
位 13	经验值（_EVNUM）选项

[1] 测量进给输入适用于 AUTO 和 JOG 模式

SD55634 $SCS_MEA_FEED_PLANE_VALUE	指定碰撞监控有效时工作平面中的中间定位进给率。
= 1000	缺省值

SD55636 $SCS_MEA_FEED_FEEDAX_VALUE	指定碰撞监控有效时进给轴中的中间定位进给率。
= 1000	缺省值

刀具探头在机床坐标系中的校准数据

在开始标定前，必须将刀具探头在机床坐标系（MCS）中的位置输入到下列通用循环设定数据中。此时的参考点为主轴上有效刀具的外直径或刀具长度。如果主轴上没有刀具，则参考点为主轴中心点和主轴上的刀具参考点。

说明

校准探头

如果在 JOG 运行方式下已经校准了刀具探头，则校准数据会记录到以下设定数据中：
SD54632 $SNS_MEA_TP_AX_DIR_AUTO_CAL[k]
在 AUTOMATIC 运行方式下不用再次校准刀具测头。
索引[k] 代表当前数据组的号码（即探头编号 -1）。

SD54625 $SNS_MEA_TP_TRIG_MINUS_DIR_AX1[k]	指定测量轴 1 负方向上的触发点。
SD54626 $SNS_MEA_TP_TRIG_PLUS_DIR_AX1[k]	指定测量轴 1 正方向上的触发点。
SD54627 $SNS_MEA_TP_TRIG_MINUS_DIR_AX2[k]	指定测量轴 2 负方向上的触发点。
SD54628 $SNS_MEA_TP_TRIG_PLUS_DIR_AX2[k]	指定测量轴 2 正方向上的触发点。
SD54629 $SNS_MEA_TP_TRIG_MINUS_DIR_AX3[k]	指定测量轴 3 负方向上的触发点。
SD54630 $SNS_MEA_TP_TRIG_PLUS_DIR_AX3[k]	指定测量轴 3 正方向上的触发点。
SD54631 $SNS_MEA_TP_EDGE_DISK_SIZE[k]	指定刀具探头的边沿长度/圆片直径。
SD54632 $SNS_MEA_TP_AX_DIR_AUTO_CAL[k]	指定在自动方式下参与探头校准的轴和方向
SD54634 $SNS_MEA_TP_CAL_MEASURE_DEPTH[k]	指定刀具探头上沿与刀具下沿之间的距离（测量铣刀半径时指定校准深度和测量深度）。
SD54635 $SNS_MEA_TPW_STATUS_GEN[k]	校准状态
SD54636 $SNS_MEA_TPW_FEED[k]	指定校准时的测量进给率

在通用循环设定数据 SD54632 $SNS_MEA_TP_AX_DIR_AUTO_CAL 中能够确定，可以在哪根轴的哪个方向上对刀具探头进行校准。

小数位		
个位	第 1 轴	
	= 0	该轴无标定
	= 1	仅有负方向
	= 2	仅有正方向
	= 3	两个方向
十位	第 2 轴	
	= 0	该轴无标定
	= 1	仅有负方向
	= 2	仅有正方向
	= 3	两个方向
百位	第 3 轴	
	= 0	该轴无标定
	= 1	仅有负方向
	= 2	仅有正方向
	= 3	两个方向

示例

如果通用循环设定数据 SD54632 $SNS_MEA_TP_AX_DIR_AUTO_CAL 的值为 123，则在 G17 平面中按如下方式校准刀具探头：

- X 轴两个方向
- 仅 Y 轴仅正方向
- 仅 Z 轴负方向

SD54633 $SNS_MEA_TP_TYPE[k]	指定探头的规格
= 0	兼容性（测量循环：立方体，车削表面表示测头，铣削表面表示砂轮）
= 101	XY 中的圆片，工件平面 G17。
= 201	ZX 中的圆片，工件平面 G18。
= 301	YZ 中的圆片，工件平面 G19。

= 2	测头
= 3	立方体

刀具探头在工件坐标系中的校准数据

开始校准探头前，必须将刀具探头在工件坐标系（WCS）中的粗略位置输入到下列通用循环设定数据中。此时的参考点为主轴上有效刀具的外直径或刀具长度。如果主轴上没有刀具，则参考点为主轴中心点和主轴上的刀具参考点。

说明

在测量刀具时要注意，可设定零点偏移或基本参考的数据总是与校准时的数据一致（在工件坐标系 WCS 中测量！）。

始终使用相同的可设定零点偏移进行测量或校准。

SD54640 $SNS_MEA_TPW_TRIG_MINUS_DIR_AX1[k]	指定测量轴 1 负方向上的触发点。
SD54641 $SNS_MEA_TPW_TRIG_PLUS_DIR_AX1[k]	指定测量轴 1 正方向上的触发点。
SD54642 $SNS_MEA_TPW_TRIG_MINUS_DIR_AX2[k]	指定测量轴 2 负方向上的触发点。
SD54643 $SNS_MEA_TPW_TRIG_PLUS_DIR_AX2[k]	指定测量轴 2 正方向上的触发点。
SD54644 $SNS_MEA_TPW_TRIG_MINUS_DIR_AX3[k]	指定测量轴 3 负方向上的触发点。
SD54645 $SNS_MEA_TPW_TRIG_PLUS_DIR_AX3[k]	指定测量轴 3 正方向上的触发点。
SD54646 $SNS_MEA_TPW_EDGE_DISK_SIZE[k]	指定刀具探头的边沿长度/圆片直径。
SD54647 $SNS_MEA_TPW_AX_DIR_AUTO_CAL[k]	指定自动校准刀具探头，自动使能轴/方向。
SD54648 $SNS_MEA_TPW_TYPE[k]	指定探头的规格
= 0	兼容性（测量循环：立方体，车削表面表示测头，铣削表面表示砂轮）
= 101	XY 中的圆片，工件平面 G17。
= 201	ZX 中的圆片，工件平面 G18。
= 301	YZ 中的圆片，工件平面 G19。
= 2	测头
= 3	立方体
SD54649 $SNS_MEA_TPW_CAL_MEASURE_DEPTH[k]	指定刀具探头上沿与刀具下沿之间的距离（测量铣刀半径时指定校准深度和测量深度）。

SD54650 $SNS_MEA_TPW_STATUS_GEN[k]	校准状态
SD54651 $SNS_MEA_TPW_FEED[k]	指定校准时的测量进给率

使用下列通用的循环设定数据 SD54647 $SNS_MEA_TPW_AX_DIR_AUTO_CAL 能够确定，可以在哪根轴的哪个方向上对刀具探头进行校准。

小数位		
个位	第 1 轴	
	= 0	该轴无标定
	= 1	仅有负方向
	= 2	仅有正方向
	= 3	两个方向
十位	第 2 轴	
	= 0	该轴无标定
	= 1	仅有负方向
	= 2	仅有正方向
	= 3	两个方向
百位	第 3 轴	
	= 0	该轴无标定
	= 1	仅有负方向
	= 2	仅有正方向
	= 3	两个方向

示例

如果通用循环设定数据 SD54647 $SNS_MEA_TPW_AX_DIR_AUTO_CAL 的值为 123，则在 G17 平面中按如下方式校准刀具探头：

- X 轴两个方向
- 仅 Y 轴仅正方向
- 仅 Z 轴负方向

用旋转主轴测量时的监控

SD54670 $SNS_MEA_CM_MAX_PERI_SPEED[0]	指定允许的待测刀具的最大圆周速度。
= 100	缺省值

SD54671 $SNS_MEA_CM_MAX_REVOLUTIONS[0]	指定允许的待测刀具的最大转速。一旦超过该转速，则转速自动减少。
= 1000	缺省值

SD54672 $SNS_MEA_CM_MAX_FEEDRATE[0]	指定允许的待测刀具接触测头时的最大进给率。
= 20	缺省值

SD54673 $SNS_MEA_CM_MIN_FEEDRATE[0]	指定待测刀具首次接触测头的最小进给率。这样可以避免刀具半径较大时首次接触测头的进给率太小。
= 1	缺省值

SD54674 $SNS_MEA_CM_SPIND_ROT_DIR[0]	指定测量刀具时主轴的旋转方向。
4 = M4	缺省值

说明

如果调用测量循环时主轴已经旋转，则会保持该旋转方向不变，不管上述数据的设置如何。

SD54675 $SNS_MEA_CM_FEEDFACTOR_1[0]	指定进给系数 1
= 10	缺省值

= 0	仅使用由循环计算出的进给率进行一次接触。但进给率至少为 “SD54673[0] $SNS_MEA_CM_MIN_FEEDRATE”指定的最小进给率。
= ≥ 1	使用进给率第一次接触。但进给率起码是 “SD54673[0] $SNS_MEA_CM_MIN_FEEDRATE”x “SD54675[0] $SNS_MEA_CM_FEEDFACTOR_1”的值。

SD54676 $SNS_MEA_CM_FEEDFACTOR_2[0]	指定进给系数 2
= 0	使用由循环计算出的进给率进行二次接触。仅在 “SD54673 $SNS_MEA_CM_FEEDFACTOR_1[0] > 0”时有效，缺省值
= ≥ 1	第二次碰触使用由 SD54673 $SNS_MEA_CM_MIN_FEEDRATE[0] 计算出的进给率，进给系数 2。 第三次接触使用计算出的进给率。

说明

进给系数 2 应该小于进给系数 1。

SD54677 $SNS_MEA_CM_MEASURING_ACCURACY[0]	指定要求的测量精度。 测量精度指刀具最后一次接触探头时的精度。
= 0,005	缺省值

用旋转主轴测量：借助补偿表进行测量值补偿

SD54691 $SNS_MEA_T_PROBE_OFFSET	激活测量结果补偿。
= 0	不指定，缺省值
= 1	指定循环内部补偿。仅当 SD54690 $SNS_MEA_T_PROBE_MANUFACTURER>0 时有效。
= 2	指定通过用户自定义补偿表进行补偿。

SD54689 $SNS_MEA_T_PROBE_MANUFACTURER	激活一些刀具探头型号预先制订的补偿表（用户专用）。
= 0	不指定，缺省值

= 1	TT130 (Heidenhain)
= 2	TS27R (Renishaw)

用户补偿值

当通用循环设定数据 SD54691 $SNS_MEA_T_PROBE_OFFSET= 2 时，下列设置有效：

SD54695 到 SD54700	补偿值用于半径测量。	参见后面的通用循环设定数据。
SD54705 到 SD54710	补偿值用于长度测量。	参见后面的通用循环设定数据。

SD54695 $SNS_MEA_RESULT_OFFSET_TAB_RAD1[n] SD54705 $SNS_MEA_RESULT_OFFSET_TAB_LEN1[n]	半径测量 长度测量
= 0	0
= 1	第 1 半径
= 2	第 2 半径
= 3	第 3 半径
= 4	第 4 半径

19.11.4.4 在车床上测量工件

车削用通用设置

MD52740 $MNS_MEA_FUNCTION_MASK	测量循环的功能屏幕
位 1	工件测量，第三几何轴（Y，车削上）上的测量
= 0	测量循环不支持已有的第三几何轴（Y 应用）！
=1	设定值说明和参数（SETVAL, _TUL, _TLL, SZO）涉及第三几何轴（Y 轴） 然而刀具长度或零点偏移的补偿发生在第二几何轴（X 轴，纵坐标）激活的部分（即：Y 轴上测量；X 轴上补偿）。 补偿目标会受参数_KNUM 影响！
位 16	刀具测头测量输入

刀具测头在机床坐标系（MCS）中的标定数据

在开始标定前，必须将刀具测头在机床坐标系（MCS）中的位置输入到下列通用循环设定数据中。

SD54615 $SNS_MEA_CAL_EDGE_BASE_AX1[n]	指定标准槽底部在测量轴 1 上的位置。
SD54617 $SNS_MEA_CAL_EDGE_PLUS_DIR_AX1[n]	指定标准槽边缘在测量轴 1 正方向上的位置。
SD54618 $SNS_MEA_CAL_EDGE_MINUS_DIR_AX1[n]	指定标准槽边缘在测量轴 1 负方向上的位置。
SD54619 $SNS_MEA_CAL_EDGE_BASE_AX2[n]	指定标准槽底部在测量轴 2 上的位置。
SD54620 $SNS_MEA_CAL_EDGE_UPPERE_AX2[n]	指定标准槽上边缘在测量轴 2 上的位置。
SD54621 $SNS_MEA_CAL_EDGE_PLUS_DIR_AX2[n]	指定标准槽边缘在测量轴 2 正方向上的位置。
SD54622 $SNS_MEA_CAL_EDGE_MINUS_DIR_AX2[n]	指定标准槽边缘在测量轴 2 负方向上的位置。

说明

在带有轴 X 轴和 Z 轴（G18）的标准车床上，测量轴 1 为 Z 轴而测量轴 2 为 X 轴。

19.11.4.5 在车床上测量刀具

刀具探头在机床坐标系（WCS）中的校准数据

如果要在机床坐标系（MCS）中校准刀具探头，就必须在下列通用循环设定数据中输入刀具探头在 MCS 中的位置。

SD54625 $SNS_MEA_TP_TRIG_MINUS_DIR_AX1[k]	指定测量轴 1（G18 平面内为 Z 轴）负方向上的触发点。
SD54626 $SNS_MEA_TP_TRIG_PLUS_DIR_AX1[k]	指定测量轴 1（G18 平面内为 Z 轴）正方向上的触发点。

SD54627 $SNS_MEA_TP_TRIG_MINUS_DIR_AX2[k]	指定测量轴 2（G18 平面内为 X 轴）负方向上的触发点。
SD54628 $SNS_MEA_TP_TRIG_PLUS_DIR_AX2[k]	指定测量轴 2（G18 平面内为 X 轴）正方向上的触发点。

刀具探头在工件坐标系中的校准数据

如果要在工件坐标系（WCS）中校准刀具探头，就必须在下列通用循环设定数据中输入刀具探头在 WCS 中的大致位置。 此时的参考点为主轴上有效刀具的外直径或刀具长度。

索引[k] 代表当前数据组的号码（即探头编号 -1）。

SD54640 $SNS_MEA_TPW_TRIG_MINUS_DIR_AX1[k]	测量轴 1 （G18 上为 Z 轴）负向上的触发点。
SD54641 $SNS_MEA_TPW_TRIG_PLUS_DIR_AX1[k]	测量轴 1 （G18 上为 Z 轴）正向上的触发点。
SD54642 $SNS_MEA_TPW_TRIG_MINUS_DIR_AX2[k]	测量轴 2（G18 上为 X 轴）负向上的触发点。
SD54643 $SNS_MEA_TPW_TRIG_PLUS_DIR_AX2[k]	测量轴 2（G18 上为 X 轴）正向上的触发点。

使用“可定向刀架”或“回转刀具”功能测量刀具

当通用循环机床数据“MD52740 $MNS_MEA_FUNCTION_MASK”的位 16 为 1 时，下列设置生效：

MD51618 $MNS_MEA_CM_ROT_AX_POS_TOL	回转轴设置的公差参数
= 0,5	缺省值。

回转轴实际的角度位置可能和编程位置不同（精准停窗口）， 该偏差取决于轴的位置环特性， 轴上可能出现的最大偏差应输入到该参数中。 一旦超过公差，会发出报警 61442“刀架与几何轴不平行”。

19.11.4.6 双主轴上的同时测量

前提条件



软件选件

如需使用功能“同时测量”，则需要软件选件：“第 2 级测量”

检查并设置通用机床数据

MD51740 $MNS_MEA_FUNCTION_MASK	测量循环的功能屏幕
位 14	工件，同时测量功能激活
= 0	双主轴同时测量功能未激活
= 1	双主轴同时测量功能激活

MD18660 $MN_MM_NUM_SYNACT_GUD_REAL[0]	可定义的 Real 型全局用户数据变量的数量
= 2	最小值

通用循环设定数据

SD54740 $SNS_MEA_FUNCTION_MASK	测量循环的功能屏幕
位 3 = 0	标定工件测头半径未接收至刀具数据中

SD54760 $SNS_MEA_FUNCTION_MASK_PIECE	输入屏幕的设置，在自动运行方式下测量， 工件测量
位 3 = 1	选择“使能测头校准数据组”
位 6 = 0	选择“使能基准参考中的零偏补偿”
位 7 = 0	选择使能通道专用基准零偏中的零偏补偿
位 8 = 0	选择“零偏使能全球基准中的零偏补偿”
位 9 = 0	选择“使能零点偏移中的零偏补偿”
位 26 = 0	不使能刀具补偿选项
位 28 = 1	使能刀具长度调整选项

SD54780 $SNS_J_MEA_FUNCTION_MASK_PIECE	输入窗口"JOG 工件测量"的设置
位 3 = 1	选择"使能测头校准数据组"
位 28 = 1	使能刀具长度调整选项

19.11.5 记录

19.11.5.1 测量记录，通用

无特殊设置可以进行常规记录：

- 在 NC 被动文件系统中
- 在 CF 卡的本地驱动器上

如果使用了一个逻辑驱动器进行记录，则需要"从外部存储执行"（EES）功能。

更多信息

有关功能 EES 的详细描述，请参见

功能手册之基本功能。

通用机床数据

记录会受以下机床数据设置影响：

MD11420 $MN_LEN_PROTOCOL_FILE	记录文件的文件大小（kB）
≥ 20	推荐值

当必须在 NC 文件系统中保存较大量的记录文件时，必要时要提高机床数据 MD11420 的值。

MD11422 $MN_PROTOCOL_FILE_MODE	WRITE 指令的特性设置
位 0	永久保存
= 0	延时的 WRITE 指令（立刻永久保存）
= 1	快速 WRITE 指令（延时永久保存）

说明

设置 MD11422 位 0 = 1，以便不会显著延长程序运行时间。

19.11.5.2 记录 JOG 模式下的测量

可以在 JOG 模式的“测量记录设置”窗口中设置以下设定数据。

更多信息

有关设置过程的更多信息，请参见

操作手册，车削版/铣削版/磨削版。

循环－通道设定数据

SD55774 $SCS_J_MEA_PROTOCOL_FILE	JOG 测量时记录文件的名称和路径
//NC:/WKS.DIR/TEMP.WPD/J_MEAPROT.TXT	默认设置

SD55740 $SCS_MEA_FUNCTION_MASK	测量循环、工件/刀具测量功能屏幕
位 28	在 JOG 中测量时新建标准记录或将它添加到已有记录中
= 0	新建记录（缺省值）
= 1	添加，即：总是添加到已有记录中
位 29	在 JOG 中测量时选择标准记录的文件格式
= 0	文本格式（缺省值）
= 1	表格格式

19.11.5.3 记录 AUTOMATIK 模式下的测量

循环－通道设定数据

SD55730 $SCS_MEA_PROTOCOL_USER_EXT	用户记录的文件后缀
TXT	默认设置

SD55614 $SCS_MEA_RESULT_MRD	测量结果图形显示 MRD 的程序控制的设置
= 0	关闭测量结果图形显示
= 1	打开测量结果图形显示（缺省值）

定义文件：用户 GUD 功能块

用户记录数据默认是从 GUD 字段 S_PROTTXT[10] 中记录的（PGUD）。

如果该数据字段不够用，可在一个自定义的 GUD 功能块（MGUD 或 UGUD）中创建第二个字段，预定义名称为 S_USERTXT[n]：

```
DEF NCK STRING[200] S_USERTXT[n] ; n = 字段元素数量
```

说明

记录过程中系统会检测是否存在 S_USERTXT 字段。 如果存在 S_USERTXT，则系统记录该字段内容；如果不存在 S_USERTXT，则系统记录 S_PROTTXT 内容。

19.11.5.4 制造商与用户循环 CUST_MEAPROT.SPF

功能

在 CUST_MEAPROT 循环中可对记录格式和记录内容进行个性化调整。

步骤

1. 打开循环文件 CUST_MEAPROT.SPF。
2. 在 Block _MARK1 中标记程序行。
3. 从 Block _MARK1 中将内容复制到 Block _MARK2。
4. 个性化调整 Block _MARK2 中的参数。
5. 保存已修改的循环文件。
   初始缺省值予以保存，可随时激活/撤销（注释符设置）。

参数

CUST_MEAPROT 循环中，在 Block _MARK2_1 下对所有可能的参数修改进行了更详细的说明。

可以调整以下参数：

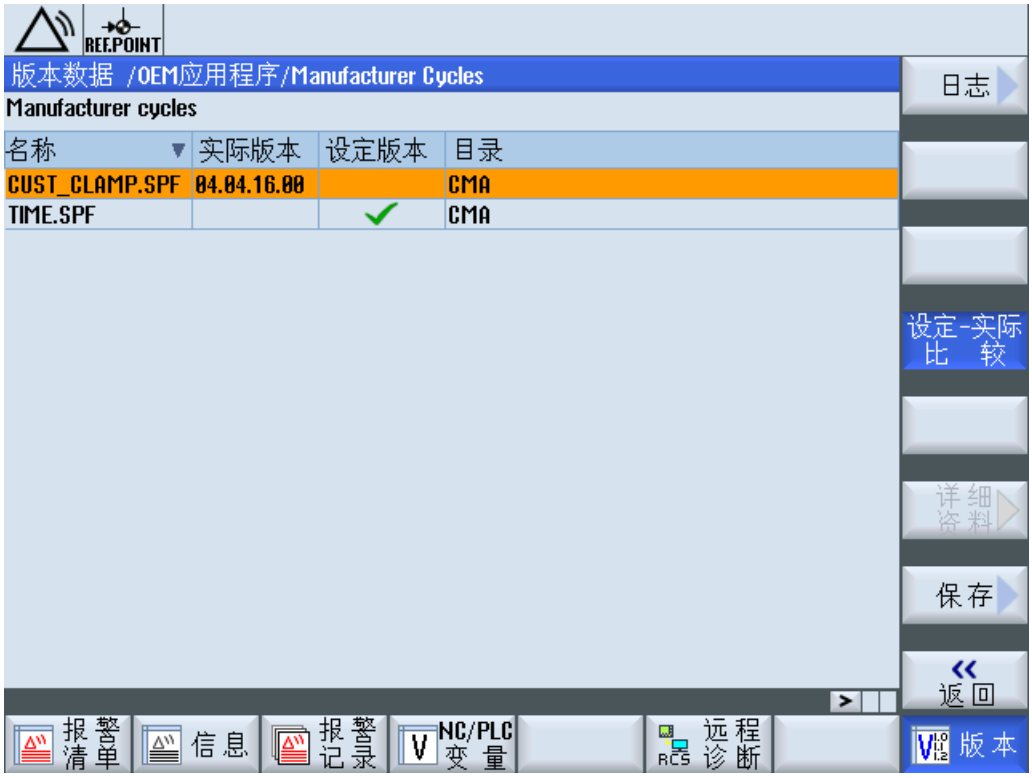
参数	缺省值	输入范围
每一行的字符	66	可大于 66 (1
每 2 到 4 列的字符	14	最小 12，最大 14
工件计数器	""	名称
列间距	" "	空字符串（≥ 2 * 空格）
公差说明标记	"#"	1 个字符
公差超出标记	"!"	1 个字符
行 5 到 7 的记录头	""	任意文本

1) 缺省设置中，根据屏幕宽度，每一行可有 66 个字符。 记录行完全显示在屏幕上。 主要记录在 NC 中的文件系统中时，出于更好的可读性原因，应当保留该设置。 主要记录在外部文件中时，可以相应地提高行宽。 该情况下会自动增大列 1 的宽度（与列 2、3、4 的宽度相同，列间距也相同）。

19.12 比较循环版本

19.12.1 显示循环版本

如果循环中配有版本说明，可以在版本视图中显示该版本。



前提条件

版本说明通过以下方式录入循环文件：

```
;VERSION: <Version> ;DATE:<YYYY-MM-DD>
```

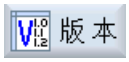
示例：

```
;VERSION: 05.05.05.00 ;DATE:2012-11-30
```

步骤



1. 选择操作区域“诊断”。



2. 按下软键“版本”。
   获取版本信息的过程较慢，对话框中会显示一个进度条和说明性文字。



3. 选中“OEM 应用/制造商循环”区并按下软键“详细”。
   “制造商循环”窗口打开。
   软键“设定-实际比较”默认选中。

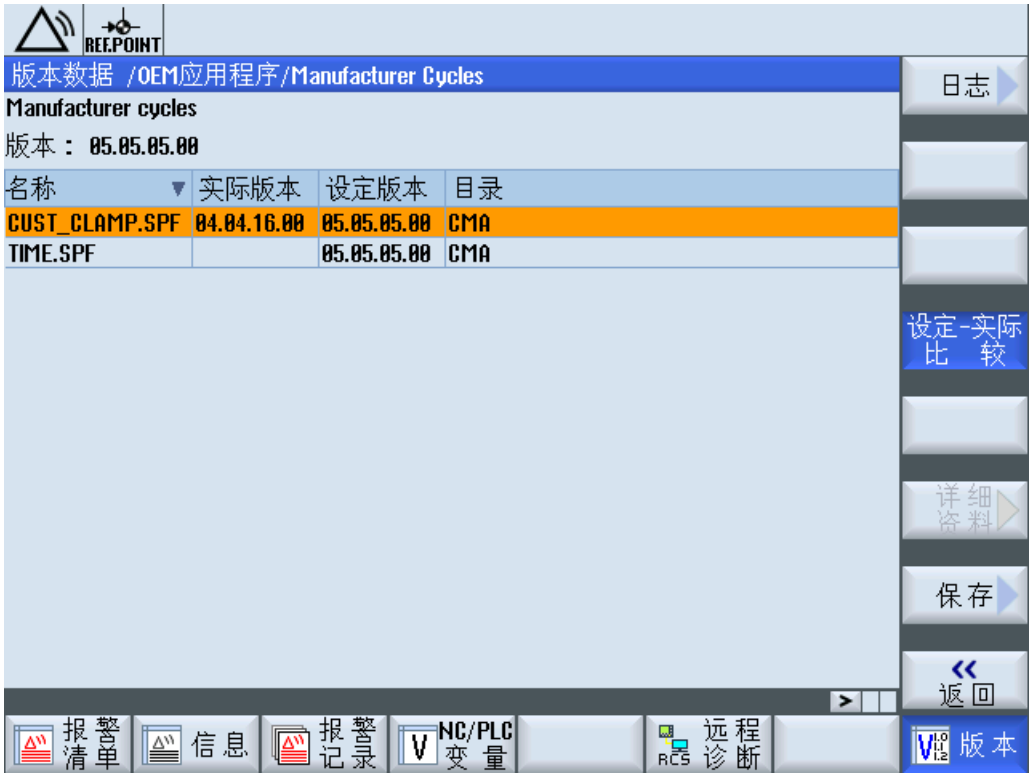
参见

保存信息 (页 327)
日志 (页 330)

19.12.2 指定循环版本

简介

可在版本视图中查看当前循环版本和目标版本的对比表。



文件概览

需要以下文件：

名称	路径	含义
versions.xml	/siemens/sinumerik/hmi/data/version/oem/cma	版本文件
versions.xml	/oem/	版本路径文件

设置版本文件

循环版本说明中需要手动设置版本文件。

1. 从如下目录中复制模板文件"versions.xml"：
   /**siemens**/sinumerik/hmi/data/version/oem/cma
2. 将模板文件保存/oem 下的自定义目录下。
   此处的默认目录为 /oem/sinumerik/hmi/data/version/oem/cma。
3. 打开文件。
4. 在标签<info>"defaultFileVersion"中输入循环文件的目标版本，在标签<Version>中输入总版本说明。
5. 关闭文件，保存修改。

示例：

```
<info defaultFileType=" *.spf *.cpf" defaultFileVersion="05.05.00.00"
linkname="CMA" linkpath="//NC/CMA.DIR" EffDirOrder="CUS CMA">
<Name>Manufacturer cycles</Name>
<Version>5.5.0.0</Version>
<Link>
<Name>CUS</Name>
<Path>//NC/CUS.DIR</Path>
</Link>
</info>
```

设置版本路径文件

如果版本文件在默认目录下且当前为 oem\versions.xml，则无需进行任何更改。 在所有其他情况下可以手动输入路径：

1. 打开 /oem 下的文件 versions.xml。
2. 使用自己的版本文件路径替换标签<Path>"Manufacturers Cycles"中的默认路径。
3. 关闭文件，保存修改。

标准：

```
<Component>
<Name>Manufacturer Cycles</Name>
<Path>/siemens/sinumerik/hmi/data/version/oem/cma</Path>
</Component>
```

示例：

```
<Component>
<Name>Manufacturer Cycles</Name>
```

```
<Path>/oem/sinumerik/hmi/data/version/oem/cma</Path>
</Component>
```

语言扩展 20

20.1 安装其他语言

语言扩展

除了现有的标准语言外，在单独的 DVD 上还提供其他界面语言。

集成的在线帮助既有各个国家的语言显示，也有英文显示。

所提供的操作界面语言请参见章节“支持的语言 (页 614)”。

前提条件



软件选件

使用该功能需要下列软件选件：

“附加语言”

- 安装后 CF 卡上必须还有约 256 MB 的存储空间。
- 语言扩展软件“hmi_sl_language_xxx_02.06.00.00.00y.arc”。
  xxx = 语言标识，y = 内部版本标识
  在语言扩展 DVD 中随附

安装语言

按照调试存档一样（ARC）安装语言。相关步骤请参见章节“导入调试存档 (页 248)”。

说明

早前版本的语言扩展软件都与当前操作软件兼容。

20.2 卸载语言

如不再需要一种语言界面，可将其卸载。

说明

标准语言和操作软件当前使用的语言无法卸载。

步骤



1. 选择操作区域“调试”。



2. 按下“更改语言”软键。
   “语言选择”窗口打开。上次设置的语言被选中。
3. 请将光标定位在需要的语言上。



4. 按下软键“删除语言”。
   窗口“删除语言”打开。



5. 按下软键“确认”。
   操作软件卸载掉所选语言。

20.3 支持的语言

表格 20-1 支持的语言

语言	语言标识	标准语言
简体中文	chs	X
繁体中文	cht	
丹麦语	dan	
德语	deu	X
英语	eng	X
芬兰语	fin	
法语	fra	X
印度尼西亚语	ind	
意大利语	ita	X

语言	语言标识	标准语言
日语	jpn	
韩语	kor	
马来西亚语	msl	
荷兰语	nld	
波兰语	plk	
葡萄牙语	ptb	
罗马尼亚语	rom	
俄语	rus	
瑞典语	sve	
斯洛文尼亚语	slv	
斯洛伐克语	sky	
西班牙语	esp	X
泰语	tha	
捷克语	csy	
土耳其语	trk	
匈牙利语	hun	
越南语	vit	

PCU/IPC 上的 SINUMERIK Operate 21

21.1 将 OEM 框架（OEMFrame）应用程序集成到系统中

集成 OEM 应用程序

需要创建并调整以下文件，将 OEMFrame 应用程序集成到操作软件中：

- 配置文件 “systemconfiguration.ini”
  系统管理程序启动并控制操作软件，因此也负责控制 OEMFrame 应用程序。可通过配置文件“systemconfiguration.ini”配置系统管理程序。
- 配置启动软键
  需要在操作区域的软键扩展条上配置一个对应软键，以便通过操作软件启动 OEMFrame 应用程序。
- 配置文件：“slamconfig.ini”
  首先创建文件“slamconfig.ini”，以配置用于启动 OEMFrame 应用程序的软键位置/文字和或者图标。
- 在文件“mytext_<lng>.ts”中加入某种语言的软键标签。

文件“systemconfiguration.ini”

首先创建配置文件“systemconfiguration.ini”，将它放在以下某个目录中，以便将 OEMFrame 应用程序集成到操作软件中：

<安装路径>/**user**/sinumerik/hmi/cfg

<安装路径>/**oem**/sinumerik/hmi/cfg

[processes] 部分为所有由系统管理程序管理的进程，包括作为 OEMFrame 应用程序集成在里面的程序。

值	含义
process	OEMFrame 应用程序的符号名称。设计操作区域中会用到。
cmdline	命令行，启动时会转至“oemframe.exe”进程。
oemframe	对于 OEMFrame 应用程序，始终设置为“true”。
windowname	OEMFrame 应用程序的 Windows 名称，可使用 “findwindow.exe” 确定。
classname	OEMFrame 应用程序的 Class 名称，可使用 “findwindow.exe” 确定。

值	含义
deferred	true：OEMFrame 应用程序在 SINUMERIK Operate 启动时不启动，而是在首次选择后才启动。
startupTime	相关进程按以下方式启动： immediately：立即（缺省） afterServices：在所有服务都引导完成后 afterGuis：在所有 GUI 组件都引导完成后 如果关闭了 SINUMERIK Operate，过程则反过来： immediately：立即（缺省） afterServices：在所有服务都关闭完成后 afterGuis：在所有 GUI 组件都关闭完成后
gimmekeys	由 OEMFrame 应用程序操控的系统配置按键的使能掩码。参数设置以位掩码 (bitmask) 的形式进行。
disablekeys	键盘过滤器属性的设置。参数设置以位掩码 (bitmask) 的形式进行。
menuselectkey	用于修改操作区域菜单要显示的按键（默认：F10）。此时的值为组合键 Shift、Alt、Ctrl 与虚拟按键代码（同 Microsoft 的定义）的或 OR 组合。
timeout	查找 OEMFrame 应用程序的最大持续时间，单位为毫秒。如果在该时间届满后未找到 OEMFrame 应用程序，则无法由系统管理器进行管理。 文件 “systemconfiguration.ini” 中的缺省设置： [miscellaneous] startTimeoutDefault
shutdowntime	OEMFrame 应用程序关闭的最大持续时间，单位为毫秒。如果在该时间届满后未找到 OEMFrame 应用程序，则该过程终止。 文件 “systemconfiguration.ini” 中的缺省设置： [miscellaneous] shutdownTimeoutDefault

位掩码 “gimmekeys”

通常，一个 OEMFrame 应用程序的位掩码都设为二进制值 0xF。F1 到 F8 的所有按键组合都提供给了 OEMFrame 应用程序。通过设置其他的位，相应的键/组合键可由 OEMFrame 应用程序本身来操控。否则系统配置将接管求值计算并且 OEMFrame 应用程序会获得完全未被提供的键/组合键。

位掩码 “gimmekeys” 可如下进行设置：

位	按键	含义
0	F1 - F8	水平软键（上边栏，HU）
1	Shift+F1 - Shift+F8	垂直软键（右边栏，VR）
2	Ctrl+F1 - Ctrl+F8	水平软键（下边栏，HL）
3	Shift+Ctrl+F1 - Shift+Ctrl+F8	垂直软键（左边栏，VL）
4	F9	Recall
5	Shift+F9	ETC 切换
6	F10	操作区域菜单
7	Shift+F10	M 键
8	F11	通道切换键
9	Shift+F11	M 键（硬键）
10	F12	信息/帮助
11	Shift+F12	自定义键（硬键）
12	ESC	报警取消
13	HOME	窗口切换键
14	END	PROGRAM（硬键）
15	PAGE UP	ALARM（硬键）
16	PAGE DOWN	TOOL OFFSET（硬键）
17	HOME（数字区）	PROGRAM MANAGER（硬键）
18	F13 - F20	扩展水平软键（上边栏，HU）
19	Shift+F13 - Shift+F20	扩展垂直软键（右边栏，VR）和右侧直接按键 HT8
20	Ctrl+F13 - Ctrl+F20	扩展水平软键（下边栏，HL）
21	Shift+Ctrl+F13 - Shift+Ctrl +F20	扩展垂直软键（左边栏，VL）和左侧直接按键 HT8

位掩码 “disablekeys”

通常，一个 OEMFrame 应用程序的位掩码都设为二进制值 0x3FFFF。这样便可筛选出所有键盘顺序且不提供给 OEMFrame 应用程序。如果将某个位设为 0，则会取消相应按键组合的键盘过滤器并且 OEMFrame 应用程序会接收该设置。

如果 OEMFrame 应用程序需要接收底边及左侧软键条上的全部软键，则将 “disablekeys” 位掩码设为二进制值 0x300FF。

位掩码 “disablekeys” 可如下进行设置：

位	按键	含义
0 - 7	预留	
8	(Shift)+Ctrl+F1	底边及左侧软键栏（HL，VL）
9	(Shift)+Ctrl+F2	底边及左侧软键栏（HL，VL）
10	(Shift)+Ctrl+F3	底边及左侧软键栏（HL，VL）
11	(Shift)+Ctrl+F4	底边及左侧软键栏（HL，VL）
12	(Shift)+Ctrl+F5	底边及左侧软键栏（HL，VL）
13	(Shift)+Ctrl+F6	底边及左侧软键栏（HL，VL）
14	(Shift)+Ctrl+F7	底边及左侧软键栏（HL，VL）
15	(Shift)+Ctrl+F8	底边及左侧软键栏（HL，VL）
16	预留	
17	预留	

位掩码 “disablekeyshigh”

快捷键的映射可能必须进行，因为在某些情况下操作系统已经对 CTRL-F4 和 CTRL-F6 作出响应。

位掩码 “disablekeyshigh” 可如下进行设置：

位	含义
0 - 28	预留
29	快捷键 CTRL-F1 到 CTRL-F8 映射给快捷键 CTRL-F13 到 CTRL-F20。
30 - 31	预留

说明

位掩码 “gimmekeys”、“disablekeys” 和 “disablekeyshigh” 可指定为十进制（如 31）或十六进制（如 0x1F）。

示例

说明

写入错误

避免写入错误。只可通过程序 FindWindow 确定 “processes” 和 “areas” 段的条目。

“notepad.exe” 和 “calc.exe”

如下示例中将两个 Windows 应用程序 “notepad.exe” 和 “calc.exe” 配置为 OEMFrame 应用程序。

```
[processes]

PROC500=process:=notepadOEM, cmdline:="C:\\WINDOWS\\system32\
\notepad.exe", oemframe:=true, deferred:=true,
windowname:="Untitled - Notepad", classname:="Notepad"


PROC501=process:=calcOEM, cmdline:="C:\\WINDOWS\\system32\
\calc.exe", oemframe:=true, deferred:=true,
windowname:="Calculator", classname:="SciCalc"


[areas]

AREA500=name:=AreaNote, process:=notepadOEM

AREA501=name:=AreaCalc, process:=calcOEM
```

“keycatcher.exe”

在以下示例中将 Windows 应用程序 “keycatcher.exe” 集成了进来。此时将全部四个软键条和回调键都提供给了 Windows 应用程序。底边和左侧软键条的键盘过滤器被取消。

```
[processes]

PROC500= process:=keycatcherOEM, cmdline:="keycatcher.exe",
oemframe:=true, deferred:=true, windowname:="keycatcher",
classname:="QWidget", gimmekeys:=0x1F, disablekeys:=0x300FF


[areas]

AREA500=name:=AreaKeyCatcher, process:= keycatcherOEM
```

在以下示例中将 Windows 应用程序 “keycatcher.exe” 集成了进来。此时将全部四个软键条和 F10 键都提供给了 Windows 应用程序。欲在 Windows 应用程序中显示操作区域菜单（F10 不再由系统配置处理），可按下 Ctrl+F12。

```
[processes]

PROC500= process:=keycatcherOEM, cmdline:="keycatcher.exe",
oemframe:=true, deferred:=true, windowname:="keycatcher",
classname:="QWidget", gimmekeys:=0x4F, disablekeys:=0x300FF,
menuselectkey:=Key_Control|0x7B


[areas]

AREA500=name:=AreaKeyCatcher, process:= keycatcherOEM
```

在以下示例中将 Windows 应用程序 “keycatcher.exe” 集成了进来。此时将全部四个软键条都提供给了 Windows 应用程序。快捷键 CTRL-F1 到 CTRL-F8 映射给快捷键 CTRL-F13 到 CTRL-F20。

```
[processes]

PROC500= process:=keycatcherOEM, cmdline:="keycatcher.exe",
oemframe:=true, deferred:=true, windowname:="keycatcher",
classname:="QWidget", gimmekeys:=0xF, disablekeys:=0x300FF


[areas]

AREA500=name:=AreaKeyCatcher, process:= keycatcherOEM
```

[areas] 段

在此部分配置 SINUMERIK Operate 的操作区域。

值	含义
name	操作区域的符号名称
process	[processes] 段中 OEMFrame 应用程序的名称。
panel	要使用的面板 (Header) 的名称。 当前只有 “SlHdStdHeaderPanel” 可用于 OEMFrame 应用程序。
plcid	ID，用于标识 SINUMERIK Operate 显示屏的操作区 只允许 150 到 199 范围内的值。

注意
覆盖基本组件 若使用的编号小于 500，则可能会覆盖西门子的基本组件。在 “processes” 和 “areas” 段中，编号范围只允许在 500 - 999 之间。

示例

```
[areas]

AREA600= name:=AreaOEM, process:=notepadOEM

AREA601= name:=AreaCalc, process:=calcOEM,
panel:=SlHdStdHeaderPanel
```

说明

只支持使用 SINUMERIK Operate 程序接口的 OEMFrame 应用程序。

[miscellaneous] 段

可在此部分完成多种设置。通常情况下，会更改启动操作区域。

密码	值
startuparea	启动操作区域的名称

示例

```
[miscellaneous]

startuparea = AreaOEM
```

配置操作区域菜单

操作区域菜单用于切换配置文件“systemconfiguration.ini”中已配置的操作区域。在每个已配置的操作区域的水平软键栏中均有一个软键，可通过此软键选择相应的区域。

操作区域菜单在操作区域软键上以文本形式显示，菜单显示的名称是在配置文件“systemconfiguration.ini”中配置的操作区域名称。为此，系统会自动在水平软键栏上为每个操作区域设置一个空闲的软键。

配置其它设置

配置其它设置时，需要配置文件“slamconfig.ini”：

- 指定软键在某个操作区域内的位置。
- 创建某个语言的软键标签。
- 在软键上显示操作区域的图标。

创建配置文件“slamconfig.ini”

复制文件“slamconfig.ini”至保存文件“systemconfiguration.ini”的目录：

<安装路径>/**user**/sinumerik/hmi/cfg

<安装路径>/**oem**/sinumerik/hmi/cfg

文件“slamconfig.ini”

可在配置文件“slamconfig.ini”中为已在文件“systemconfiguration.ini”中配置过的操作区域创建一个段落。此段落的名称必须包含有已配置的操作区域的名称，比如 AreaOEM。

值	含义
TextId	外语文本的文本，该文本会显示在软键上。
TextContext	外语文本的语境
TextFile	语境和外语文本所属的文本文件的名称。
Picture	软键图标文件的名称。
SoftkeyPosition	区域软键的固定软键位置。其中，软键位置 1 至 8 位于第 1 个水平软键栏，软键位置 9 至 16 位于第 2 个水平软键栏，等.....
AccessLevel	显示软键的最低访问权限。没有给定该值时，设置访问级别 7（钥匙开关位置 0）。

示例

在下面的示例中，展示了如何配置具有以下特性的操作区域“AreaOEM”软键：

- 软键显示在文本文件“mytext_<lng>.ts”的“mycontext”中 TextID“MY_AREA”定义的文字。
- 软键所显示的图标为“mypicture.png”。
- 该软键位于操作区域菜单中的位置 7。
- 具有访问级别 5（钥匙开关位置 2）时显示该软键。

```
[AreaOEM]
; 和语言相关的文本 ID
TextId = MY_AREA
; 包含文本 ID 的文本文件名称
TextFile = mytext
; 文本 ID 指定文本文件的语境
TextContext = mycontext
; 软键上显示的图标的文件名
Picture = mypicture.png
; 区域菜单上的区域软键位置，
; 没有指定位置时，会查找空位置
SoftkeyPosition = 7
; 软键的访问等级
AccessLevel = 5
```

说明

操作区域位置 7 已预留给 OEM 客户。

软键的标签

保存路径：

<安装路径>/**user**/sinumerik/hmi/lng

<安装路径>/**oem**/sinumerik/hmi/lng

XML 标签的含义如下：

属性	说明
context	文件文件中的“语境”段。 每个文件都必须至少有一个“语境”。
name	语境的名称。
message	文本翻译。 每个语境必须有一条消息。

属性	说明
source	文本 ID。
translation	翻译后的文本。
remark	文本的注释（可选）。
chars	指定允许的最大文本字符数量。如果没有指定，则文本可以为任意字符数（可选）。
lines	指定最多可用于显示的行数。如果没有指定，则为任意行数（可选）。

和语言相关的 ts 文件的结构，包含了软键的标签：

mytext_<lng>.ts

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!DOCTYPE TS>
<TS>
  <context>
  <name>mycontext</name>
    <message>
      <source>MY_AREA</source>
        <translation>软键上显示的文本。</translation>
        <remark>注释（可选）</remark>
        <chars>20</chars>
        <lines>2</lines>
    </message>
  </context>
</TS>
```

<lng>代表语种标识。

21.2 OEMFrame 应用程序的参数设置

文件 “oemframe.ini”

通过文件 “oemframe.ini” 可继续对 OEMFrame 应用程序进行参数设置。将文件保存在以下目录：

<安装路径>/compat/oem

针对每个 OEMFrame 应用程序创建一个带所需参数的固有段落。根据相应的程序文件为其命名，不带文件名扩展。将名称加入方括号内。

示例

```
[notepad]
```

参数一览

以下参数可用于 OEM 应用程序：

参数	含义	缺省值
WindowStyle_On	表示要分配给窗口的属性。	0
WindowStyte_Off	表示窗口所不具有的属性。	0
x	OEMFrame 应用程序的水平起始坐标（单位：像素）	0
y	OEMFrame 应用程序的垂直起始坐标（单位：像素）	0
Width	OEMFrame 应用程序的宽度（单位：像素）	桌面宽度
Height	OEMFrame 应用程序的高度（单位：像素）	桌面高度
nDelayInitComplete	延迟向系统管理器反馈	0
fSearchOnlyForTaskWindow	说明在 “systemconfiguration.ini” 中指定的窗口是否属于同样在此处指定的任务。	1
fRestoreTaskWindow	确定退出从 OEMFrame 应用程序中调用的应用程序时的特性。	0
fKeepPlacement	取消大小调整	0
fForceTaskFocus fSearchForPopUps	该参数确定启动 OEMFrame 应用程序时显示哪个窗口。	0 1
nInitShowMode	启动 OEMFrame 应用程序时窗口显示的状态。	SW_SHOWMINNOACTIVE
nShowMode	OEMFrame 应用程序的窗口显示状态。	SW_SHOWNORMAL
nUnShowMode	OEMFrame 应用程序的窗口隐藏状态。	SW_SHOWMINNOACTIVE

参数	含义	缺省值
fWinForms	在涉及 "Windows Forms Application" 时进行设置。	0
nSwitchToTaskAfterTermination	控制结束 OEMFrame 应用程序时的特性	-1
fFindWindowWithWildcards	在属性 Windowname 中使用占位符 (?, *)	0

WindowStyle_On/WindowStyle_Off

Windows 应用程序的外观主要借助 Windows-API 函数 SetWindowLong 来确定。调用函数 SetWindowLong 时，应用程序的外观通过长度为 8 个字节的字来控制。使用参数 WindowStyle_On 和 WindowStyle_Off 可修改其中 2 个字节。

参数 WindowStyle_On 表示要分配给窗口的属性。参数 WindowStyle_Off 表示窗口所不具有的属性。

以下控制方法可采用 WindowsStyle 参数（二进制特征值）：

0000	0000	xxxx	xxxx	0000	0000	0000	0000
		1010		标题			
		1000		边界			
		0100		对话框的窗口边框样式			
		0010		垂直滚动条			
		0001		水平滚动条			
			1000	系统菜单			
			0100	边框 (Thickframe)			
			0010	Minimize-Box（最小化按钮）			
			0001	Maximize-Box（最大化按钮）			

将二进制特征值以十进制数形式分配给 WindowStyle 参数。换算可通过计算器在 Windows 中进行。

换算示例

必须对系统菜单以及水平和垂直滚动条的属性进行定义。根据表格应为：

0000 0000 0011 1000 0000 0000 0000 0000 二进制或

0038 0000 十六进制

1. 在计算器上确认选择十六进制。
2. 输入数字 00380000（可以省略前面的零）。
3. 确认选择十进制。
   结果：3670016
4. 将结果以特征值形式分配给参数。

应用示例

为 Windows 应用程序 Notepad 显示系统菜单以及水平和垂直滚动条：

```
[notepad]
WindowStyle_On = 3670016
```

不在 Windows 应用程序 Notepad 显示最小化按钮和最大化按钮：

```
[notepad]
WindowStyle_Off = 196608
```

x/y

参数 x 和 y 表示所集成的 Windows 应用程序的窗口的起始坐标，将屏幕的左上角作为测量原点。X 是水平坐标，Y 是垂直向下的坐标。尺寸单位为像素 (pixel)。

可用的工作区域取决于使用的屏幕布局。

Width

该参数表示 Windows 应用程序的窗口宽度，从窗口原点开始测量，根据参数 x，单位像素。

Height

该参数表示 Windows 应用程序的窗口高度，从窗口原点开始测量，根据参数 y，单位像素。

nDelayInitComplete

一旦初始化 Windows 应用程序的窗口，就会向系统管理器发送一条消息。之后即可通过系统管理器来选择 Windows 应用程序。使用参数 nDelayInitComplete 可将此类消息的发送延迟。指定的单位为毫秒。

如果 Windows 应用程序在生成其窗口期间还须执行其他操作，则有必要延时。如果窗口过早由系统管理器激活，则可能导致显示故障。

示例

一个 Windows 应用程序 “app.exe” 在生成其窗口后还要继续从数据库中读取状态数据。Windows 应用程序的窗口在所有的状态数据都读取完毕后才可显示。该读取操作应持续约一秒。可进行以下参数设置：

```
[app]
nDelayInitComplete = 2000
```

fSearchOnlyForTaskWindow

使用该参数指定在文件 “systemconfiguration.ini” 中通过 `ClassName`/`WindowName` 定义的窗口是否属于同样在此处指定的任务。

可使用以下值：

fSearchOnlyForTaskWindow	
= 0	该窗口不属于给定的任务。
= 1	该窗口属于给定的任务。

在查找时，不只局限于在 “systemconfiguration.ini” 中所设置的任务的窗口，而是针对查找时存在于系统中的全部窗口。

示例

Windows 应用程序由多个进程组成，例如 “startup.exe” 和 “user.exe”。在文件 “systemconfiguration.ini” 中（只）记录了 “startup.exe”，之后会从该进程中启动 “user.exe”。应用程序窗口属于 “user.exe”，因此在单独查找 “startup.exe” 的窗口时不会被找到。

可进行以下参数设置：

```
[startup]
fSearchOnlyForTaskWindow = 0
```

fRestoreTaskWindow

该参数确定退出从 OEMFrame 应用程序中调用的 Windows 应用程序时的特性，作为第二任务级。

一般在退出 OEMFrame 应用程序时，会对最后激活的窗口 `ForegroundWindow` 进行保存。在重新选择此 OEMFrame 应用程序时，会再次激活该窗口。

如果从 OEMFrame 应用程序中又启动了其他的应用程序，激活的窗口是指新的 Windows 应用程序。

说明

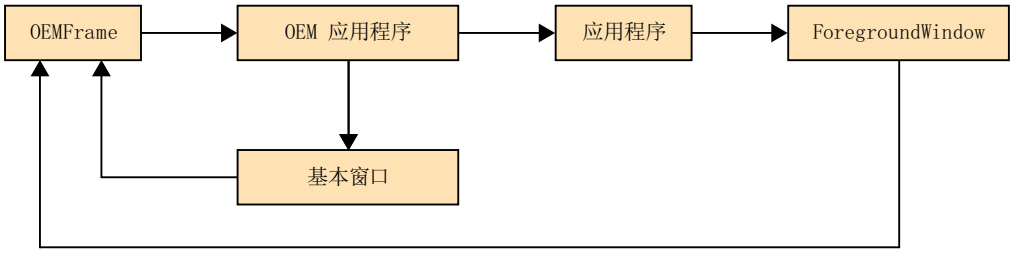
错误显示

某些情况下，退出新 Windows 应用程序无法识别代理应用程序 “oemframe.exe”。在显示第二任务级时可能会出错。激活 OEMFrame 应用程序的基本窗口。

通过以下值可修改特性：

fRestoreTaskWindow	
= 0	在选择 OEMFrame 应用程序或从 OEMFrame 应用程序中启动 Windows 应用程序时会激活窗口 ForegroundWindow。在退出二任务级时会显示 OEMFrame 应用程序。
= 1	在选择 OEMFrame 应用程序或从 OEMFrame 应用程序中启动 Windows 应用程序时，会预先激活窗口 ForegroundWindow 的一个基本窗口。在退出第二任务级时始终显示 OEMFrame 应用程序的基本窗口。

示意图



fForceTaskFocus/fSearchForPopUps

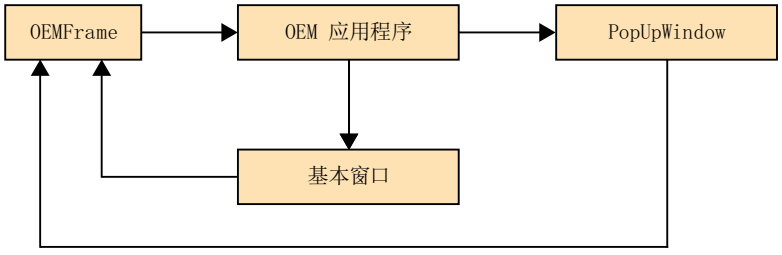
使用这两个参数确定，OEMFrame 应用程序在退出和重新选择后激活哪个窗口。

一般在区域切换时，会对最后激活的 OEMFrame 应用程序的窗口 ForegroundWindow 进行保存。ForegroundWindow 窗口在重新选择应用程序时会被再次激活。

通过以下值可修改此特性：

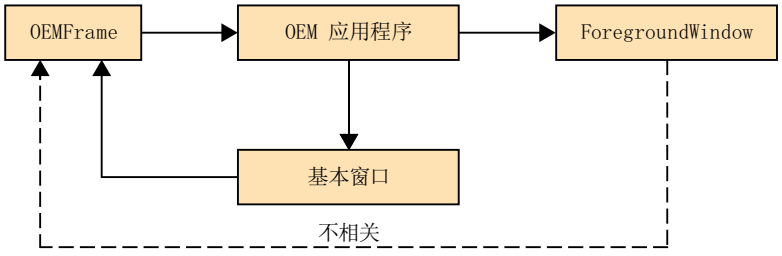
fForceTaskFocus 和 fSearchForPopUps	
= 1	在撤销 OEMFrame 应用程序后，不会根据窗口 ForegroundWindow 查找，而是根据生效的 Popup 窗口查找。 查找方式： • 如果找到了 Popup 窗口，则在重新选择 OEMFrame 应用程序时会显示该窗口。 • 如果不存在或未找到 Popup 窗口，则在重新选择时会显示应用程序的基本窗口。

示意图：带值 1 的 fForceTaskFocus/fSearchForPopUps



修改以下值后，重新选择应用程序时还会查找 OEMFrame 应用程序的基本窗口。基本窗口通过 ClassName/WindowName 在文件 “systemconfiguration.ini” 中指定：

示意图：带值 1 的 fForceTaskFocus 和带值 0 的 fSearchForPopUps



fKeepPlacement

该参数取消 OEMFrame 应用程序基本窗口的大小调整（缩放）。通常应用程序在显示前会对其屏幕大小进行缩放。对不允许进行窗口缩放的应用程序进行大小调整时会引起显示问题。此时应取消缩放。

示例

应用程序 “fixres.exe” 应以所设置的窗口大小显示。需要进行以下参数设置：

```
[fixres]
```

```
fKeepPlacement = 1
```

fSearchForPopUps	
= 0	未找到生效的 Popup 窗口。

nInitShowMode/nShowMode/nUnShowMode

这三个参数确定在启动时以及在显示和隐藏应用程序时以何种形式显示应用程序窗口。

参数 nInitShowMode 用于启动应用程序。参数 nShowMode 用于显示应用程序（区域被激活）。参数 nUnShowMode 用于隐藏应用程序。

参数 nShowMode 和 nUnShowMode 的取值范围如下：

nShowMode 和 nUnShowMode	
= 0	应用程序窗口不可见 (SW_HIDE)。
= 1	应用程序窗口以其原始形式（位置，大小）显示并获得输入焦点 (SW_SHOWNORMAL、SW_NORMAL)。
= 2	应用程序窗口被最小化并获得输入焦点 (SW_SHOWMINIMIZED)。
= 3	应用程序窗口被最大化 (SW_SHOWMAXIMIZED)。
= 4	显示应用程序窗口，但不获得输入焦点 (SW_SHOWNOACTIVATE)。
= 5	显示应用程序窗口并获得输入焦点 (SW_SHOW)。
= 6	应用程序窗口被最小化并失去输入焦点 (SW_MINIMIZE)。
= 7	应用程序窗口被最小化，但不获得输入焦点 (SW_SHOWMINNOACTIVE)。
= 8	显示应用程序窗口，但不获得输入焦点 (SW_SHOWNA)。
= 9	应用程序窗口以其原始形式（位置，大小）显示 (SW_RESTORE)。
= 10	应用程序窗口以应用程序启动时的相同形式显示 (SW_SHOWDEFAULT)。

说明

显示问题

缺省设置适用于大多数使用情况。

例外情况是，采用 Borland-Delphi 显示的应用程序可能会出现显示问题，如失相窗口等。

此时应使用以下参数：

```
nUnShowMode = 0
fKeepPlacement = 1
```

或

```
nInitShowMode = 1
```

fWinForms

Windows Forms 参数为应用程序提供了同样使用 Windows 标准应用的控制元素，如对话框、菜单和按钮。

如果 OEMFrame 应用程序涉及 “Windows Forms Application”，则必须进行以下的参数设置：

```
[<app-name>]
fWinForms = 1
```

说明

取消大小调整

如果参数 `fWinForms` 未置位，OEMFrame 应用程序不会以最大化形式打开。指定的大小调整（`x`、`y`、`Width` 和 `Height`）被取消。

nSwitchToTaskAfterTermination

该参数用于控制结束 OEMFrame 应用程序时的特性。在该情况下，系统管理器通常会显示操作区域切换。

通过参数 `nSwitchToTaskAfterTermination` 可实现自动切换回之前生效的区域。

`nSwitchToTaskAfterTermination`	
`= -1`	显示操作区域切换（缺省）
`= -2`	切换至之前的有效区域

示例

结束 Windows 应用程序 “closeapp.exe” 时，应切换回之前的区域。需要进行以下参数设置：

```
[closeapp]
nSwitchToTaskAfterTermination = -2
```

fFindWindowWithWildcards

通过该参数可为属性 Windowname 激活一个占位符查找。这种方法可以区分在每次启动相同的属性 Windowname 时可能会变化的应用程序，如：访问等级、Windowname 中的时间等：

fFindWindowWithWildcards		
= 0	占位符查找已关闭	
= 1	占位符查找已开启 占位符 (Wildcard)：	
	?	一个任意字符
	*	任意字符数量

示例

现在需要集成一个应用程序，其启动时间包含在属性 Windowname 中。

oemframe.ini

```
[MyQtTest]
fFindWindowWithWildcards = 1
```

systemconfiguration.ini

```
[processes]
PROC500=process:=ProcessOEM, cmdline:="C:\\Program Files(x86)\\Siemens\\
MotionControl\\oem\\sinumerik\\hmi\\appl\\MyQtTest.exe",oemframe:=true,
windowname:="MyQtTest - ??:??:??", classname:="QWidget",deferred:=true

[areas]
AREA500=name:=Test, process:=ProcessOEM
```

21.3 编制 OEM 子目录

如果要集成多个 OEM 应用程序（用户界面对话框或 OEMFrame 应用程序），则可将其分别分配到各自的 OEM 子目录中。这样做的好处是可将文件与 OEM 应用程序清晰的分类。这尤其适用于文件 "systemconfiguration.ini"，因为在复制时总是存在将其误覆盖并因此中断其他 OEM 应用程序的风险。

OEM 子目录可使用任意目录名称在 "/oem/sinumerik/hmi" 路径下进行创建。该子目录在目录/oem/sinumerik/hmi/**cfg** 下的 "systemconfiguration.ini" 文件中进行声明。在**[oem_dirs]**段中使用以下句法进行子目录的创建：

```
OEM_<数目>=<OEM 子目录>
```

示例

```
[oem_dirs]
OEM_1=oem_example
OEM_2=oem_new
```

在这两个 OEM 子目录（"/oem/sinumerik/hmi/oem_example" 和 "/oem/sinumerik/hmi/oem_new"）下可根据需要继续创建目录 appl、cfg、lng、hlp 等。

21.4 FindWindow 程序应用

通过程序 FindWindow 可以查询配置文件 "systemconfiguration.ini" 的所有相关参数，这些参数用于将一个 OEMFrame 应用程序集成到 SINUMERIK Operate 中。

前提条件

为使通过 FindWindow 程序进行的更改在 SINUMERIK Operate 中生效，需要访问级别 2（服务）。

安装

FindWindow 可在任何 PCU 上使用且无需安装。

FindWindow (findwindow.exe) 位于以下目录中：
C:\Program Files (x86)\Siemens\MotionControl\Common\FindWindow\

应用

1. 启动 PCU。
2. 安装需要通过 OEMFrame 集成的应用程序。
3. 打开应用程序。
4. 打开程序 FindWindow。
5. 通过点击鼠标左键将程序 FindWindow 中的红色的叉拖至应用程序的标题栏。
6. 检查应用程序参数 “FileName”、“ClassName” 等。
7. 根据需要，通过复选框选择参数 “Gimmekey”、“Disablekeys” 等。

   如果存在配置文件 “systemconfiguration.ini” 且需对其进行扩展：
8. 点击按键 “Copy to Clipboard” 将条目复制到剪贴板中。
9. 打开现有配置文件并添加剪贴板中的参数。在插入参数的位置上点击鼠标右键，然后点击“添加”。
10 保存已修改的配置文件。

.

-或者-

如果不存在配置文件 “systemconfiguration.ini” 且需新建：

8. 点击按键 “Create systemconfiguration.ini”。

   在 FindWindow 的目录中同时创建一个新配置文件。覆盖已有的配置文件。
9. 将新配置文件的备份添加到指定目录中。

说明

如果连接一个 64 位应用程序，“FileName” 下不会显示任何参数数据。此时会打开一个提示对话框，可通过该对话框手动选择应用程序。

说明

只有在不进行此参数设置会出现问题的情况下才可以使用 8.3 文件名标记。

21.5 激活/取消窗口模式

SINUMERIK Operate 在 PCU/IPC 上默认不以窗口模式 (Window Mode) 显示。

在文件“slguiconfig.ini”中激活操作软件的窗口模式。

步骤

1. 从目录
   /**siemens**/sinumerik/hmi/template/cfg 中复制文件 "slguiconfig.ini"。
2. 把复件粘贴到目录:
   /**oem**/sinumerik/hmi/cfg 或 /**user**/sinumerik/hmi/cfg 下。
3. 在编辑器中打开文件。
4. 为激活或者取消窗口模式进行如下设置：
   - 激活窗口模式：
     在段落 [WindowMode] 中，ActivateWindowMode = true
   - 取消窗口模式：
     在段落 [WindowMode] 中，ActivateWindowMode = false
5. 重新启动 SINUMERIK Operate。
   启动后，操作软件以窗口模式显示。

21.6 设置边缘平滑的文本可读性

为了能够更好地读取极小的 TrueType 字体，Windows 下的文本显示已经过优化。此设置能够获得更好的字体边缘平滑效果和屏幕显示效果（Windows ClearType 工艺）。

如需修改 SINUMERIK Operate 的文本可读性，可通过文件 “slstartup.ini” 调整以下参数。

调整文件 “slstartup.ini”

参数	说明	
`[FontSmoothing]`	说明与文本可读性设置相关的所有参数所在的段落	
`UseSystemSettings`	确定边缘平滑和显示优化的系统设置方式	
	`= true`	Windows 设置
	`= false`	SINUMERIK Operate 设置（缺省）
`UseFontSmoothing`	激活 SINUMERIK Operate 中所用字体的边缘平滑（只可通过已关闭的参数 `UseSystemSettings`）	
	`= true`	接通边缘平滑
	`= false`	关闭边缘平滑（缺省）

参数	说明	
UseFontSmoothingType	激活 Windows 显示优化 ClearType（只可通过已关闭的参数 UseSystemSettings 和已接通的参数 UseFontSmoothing）	
	= 1	只使用边缘平滑 FE_FONTSMOOTHINGSTANDARD
	= 2	使用 ClearType 工艺 FE_FONTSMOOTHINGCLEARTYPE
	= 0	关闭功能（缺省）

存储位置

用于 SINUMERIK Operate 的文件模板 “slstartup.ini” 保存在以下目录：

<安装路径>/siemens/sinumerik/hmi/cfg/slstartup.ini

将文件复制到以下一个目录中：

<安装路径>/**user**/sinumerik/hmi/cfg

<安装路径>/**oem**/sinumerik/hmi/cfg

说明

如要通过自定义更改对概览进行改进，则只需从文件副本 “slstartup.ini” 中删除未更改的参数。缺少的参数继续由目录 /siemens/sinumerik/hmi/cfg 从模板中读取。

21.7 使用互动安装模式或静默（Silent）安装模式

简介

如果通过 PC/PCU 安装 SINUMERIK Operate，则通过 MSI 安装程序进行。 此时有以下安装模式可供使用：通过 Windows 指令行进行的互动安装或静默安装

- SINUMERIK Operate 完全安装
- SINUMERIK Operate 基础安装（带帮助文件）
- SINUMERIK Operate 基础安装（不带帮助文件）

执行

通过指令行控制安装程序的执行。 可通过互动模式/v 将参数字符串传输至 setup.exe。 如果想不通过用户互动执行安装，可使用静默模式/qn 扩展参数字符串。

将参数字符串置于引号中。 如果要在一列字符串中使用多个参数，中间需用空格隔开。

如果调用 setup.exe 时没有给出参数，则会在互动模式中执行完全安装。

说明

在/v 和参数字符串之间无需空格。

说明

如果静默模式中如要一个进度条，则可以指定/qn 或/qb!。

互动安装模式

setup.exe	互动模式中的完全安装
setup.exe /v"BASEONLY=1"	互动模式中的基础安装，带帮助文件
setup.exe /v"BASEONLY=1 HELPFILES=0"	互动模式中的基础安装，不带帮助文件

静默（Silent）安装

setup.exe /v"/qn" setup.exe /s /v"/qn"	静默模式中的完全安装 (/s 隐藏所有安装信息）
setup.exe /v"/qn BASEONLY=1" setup.exe /s /v"/qn BASEONLY=1"	静默模式中的基础安装，带帮助文件 (/s 隐藏所有安装信息）
setup.exe /v"/ qn BASEONLY=1 HELPFILES=0" setup.exe /s /v"/ qn BASEONLY=1 HELPFILES=0"	静默模式中的基础安装，不带帮助文件 (/s 隐藏所有安装信息）

其他设置

安装路径

如果要安装 SINUMERIK Operate PC 型，则需额外指定安装路径：
setup.exe /s /v"/qn INSTALLDIR=C:\Programme\testdir"

路径中的空格：

```
setup.exe /s /v"/qn INSTALLDIR=\"C:\Program Files\testdir\""
```

退出代码

从 setup.exe 调用退出代码上可以判断安装是成功还是失败：

退出代码 == 0	无故障
退出代码 <> 0	出错

日志文件

安装过程中可通过/L 创建日志文件：

```
setup.exe /s /v"/qb! /L*vx log.txt"
```

21.8 设置 NCU 的 IP 地址

可通过 SINUMERIK Operate 的操作面板设置相连的 NCU 的 IP 地址。

步骤



1. 选择操作区域“调试”。





2. 按下软键“HMI”和“>>”。



按下垂直软键“NCU 连接”。

“NCU 连接”窗口打开。

在输入栏中输入 NCU 的 IP 地址。



3. 按下软键“确认”，确定输入。

21.9 退出 SINUMERIK Operate

步骤



1. 按下操作面板上的<MENU SELECT>键。

   或者是：

   激活触摸屏上的<M>图标。



   外部键盘： 按下<F10>键。



2. 按下菜单扩展键。

   外部键盘： 按下按键<Shift> + <F9>。



3. 按下软键“退出”。

   外部键盘： 按下<F8>。

   - PCU 上的 SINUMERIK Operate 和 Windows 系统退出。
   - 如需在服务模式中启动了 PCU 上的操作软件 SINUMERIK Operate，则只退出操作软件。
   - 在 PC 上仅退出 SINUMERIK Operate。

手持终端 22

移动式手持终端 HT 8 和 HT 10 在一台设备中集合了操作面板和机床控制面板的功能。它能实现完整的机床操作和监控功能。

更多信息

有关手持终端调试和操作的更多信息，请参见：

- 操作手册，通用版/车削版/铣削版/磨削版
- HT 8 手持终端设备手册
- HT 10 手持终端设备手册

22.1 设置 HT 8 运行键

HT 8 的运行键上没有印制名字，所以它的作用没有固定规定。运行键的名称应该根据按键的作用灵活调整。按键的名称将显示在屏幕内部垂直软键栏内，以表明运行键的作用。

运行键的菜单由两列垂直键组成，每列 8 个键，所以可以最多设计 16 个文本。最上方和下方的运行键为空，可以设定其它功能。

可以显示以下数据：

- 机床轴名称
- 机床轴假符号名
- 任意特定语言的文本
- 符号

需要以下用户特定的文件。可将样本文件作为模板使用：

文件	含义
配置文件“sljkconfig.ini”	配置运行键的文件。
文本文件 "oem_sljk_xxx.ts"	特定语言的运行键名称文件，xxx = 语种标识

配置文件“sljkconfig.ini”

条目	含义	
[State_1]	运行键名称类型 - 通过 PLC 切换。	
ParamText_x_y	由参数组成的运行键文本。 允许 2 行名称，其中每行最多包含 5 个字符。 x:按键列内按键的位置（2 到 7） y:按键列（1 或 2）。	
	%m1	第 1 轴的机床轴名用作参考，显示为文本。从数据模块 DB10 读取当前生效的轴编号。通过该索引确定机床数据的名称。
	%n	定义文本中换行的位置。
	%a1	第 1 轴的假符号名用作参考，显示为文本。从数据模块 DB10 读取当前生效的轴编号。通过该索引确定文本文件“oem_sljk_deu.ts”的名称。
TextId_x_y	运行键的文本，从文本文件中读取（外语文本）	
Picture_x_y	待显示符号的文件名。 带符号的文件必须位于以下目录中：/**oem**/sinumerik/hmi/ico 以及 根据分辨率选择以下子目录： /ico640 /ico800 /ico1024 /ico1280	

步骤

1. 从目录 /**siemens**/sinumerik/hmi/template/cfg 中复制配置文件 "sljkconfig.ini"。
2. 将复制件保存到目录/**oem**/sinumerik/hmi/cfg 或/**user**/sinumerik/hmi/cfg 中。
3. 在编辑器中打开拷贝，确定运行键的名称。

配置文件 "sljkconfig.ini" 示例

在示例中确定了 12 个轴，软键 1 和软键 2 包含文本，软键 3 和软键 4 分别有一幅图片。

```
[Settings]

文件类型 = INI
```

```
；其它配置选项的几个示例
；加工轴的假符号名（%a1、%a2 等），来自 oem_sljk_deu.ts（示例）

[State_1]
TextId_1_1 = OEM_JK_TEXT_1
TextId_1_2 = OEM_JK_TEXT_2
ParamText_2_1 = %a1%n-
ParamText_2_2 = %a1%n+
ParamText_3_1 = %a2%n-
ParamText_3_2 = %a2%n+
ParamText_4_1 = %a3%n-
ParamText_4_2 = %a3%n+
ParamText_5_1 = %a4%n-
ParamText_5_2 = %a4%n+
ParamText_6_1 = %a5%n-
ParamText_6_2 = %a5%n+
ParamText_7_1 = %a6%n-
ParamText_7_2 = %a6%n+
Picture_8_1 = AlarmCancel.png
Picture_8_2 = AlarmNCReset.png
```

文本文件“oem_sljk_deu.ts”

条目	含义
name	可自由选择的文本名称。 在文本文件的模板中，文本名称为“SIJkLabels”，代表运行键名称（solution line jog key labels）。该名称已经保存在配置文件中。
source	各个轴的运行键名称。在配置文件“sljkconfig.ini”中，文本 ID 参考“TextId_2_1”。 不允许修改假符号名（JK_AXISNAME_2 到 JK_AXISNAME_7）的文本 ID。
translation	输入 <source> 中轴的外语文本。

文件名称“xxx”中的语言标识

语言	语言标识
德语	deu
英语	eng
法语	fra
西班牙语	esp
意大利语	ita
汉语	chs

步骤

1. 可从以下目录中复制模板文件“oem_sljk_deu.ts”：/**siemens**/sinumerik/hmi/template/lng。
2. 在目录 /**oem**/sinumerik/hmi/lng 或者 /**user**/sinumerik/hmi/lng 中保存或创建文件。
3. 为文件命名，比如对于德语文本可命名为：“sljk_deu.ts”。
   若需创建其它语种的说明，必须为此语种创建一个自身文件。在文件名称中使用相应的语种标识并保存此文件。使用如上所规定的语种标识。
4. 打开此文件，并在区域 <message> 和 </message> （消息）中确定说明。
5. 重新启动 HMI。
   为了在程序运行时能显示按键名称，必须将文件转换至二进制格式。此转换过程只能在启动 HMI 时完成。

文本文件“sljk_deu.ts”示例

示例中确定了 12 个轴，软键 1 (SF1) 和 2 (SF2) 有文本：

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE TS>
<TS>
    <context>
       <name>SlJkLabels</name>
       <!-- 机床轴的假符号名 (%a1, %a2, 等) -->
       <!-- 不要更改文本 ID (JK_AXISNAME_1, 等) -->
       <message>
            <source>JK_AXISNAME_1</source>
            <translation>X</translation>
       </message>
       <message>
            <source>JK_AXISNAME_2</source>
            <translation>Y</translation>
       </message>
       <message>
            <source>JK_AXISNAME_3</source>
            <translation>Z</translation>
       </message>
       <message>
            <source>JK_AXISNAME_4</source>
            <translation>A</translation>
       </message>
       <message>
            <source>JK_AXISNAME_5</source>
            <translation>B</translation>
       </message>
       <message>
            <source>JK_AXISNAME_6</source>
            <translation>C</translation>
       </message>
```

```
<message>
    <source>JK_AXISNAME_7</source>
    <translation>U</translation>
</message>
<message>
    <source>JK_AXISNAME_8</source>
    <translation>V</translation>
</message>
<message>
    <source>JK_AXISNAME_9</source>
    <translation>W</translation>
</message>
<message>
    <source>JK_AXISNAME_10</source>
    <translation>UV1</translation>
</message>
<message>
    <source>JK_AXISNAME_11</source>
    <translation>UV2</translation>
</message>
<message>
    <source>JK_AXISNAME_12</source>
    <translation>UV3</translation>
</message>
<!-- 用户定义的、特定语言的文本（示例） -->
<message>
    <source>OEM_JK_TEXT_1</source>
    <translation>SF1</translation>
</message>
<message>
    <source>OEM_JK_TEXT_2</source>
    <translation>SF2</translation>
```

```
        </message>
</TS>
```

22.2 HT 10 运行键

如果 HT 10 已与系统相连，则可自动通过操作软件中的复选框选择轴并使用按键 +/- 或手轮来运行。

通过复选框选择轴

为了通过面板的触控功能勾选复选框来选择轴，请设置以下通道专用机床数据：

MD52210 $MCS_FUNCTION_MASK_DISP	显示功能屏幕
位 16	通过 HMI 选择 JOG 轴

说明

如果像在机床控制系统上一样通过进行轴选择来实现在 MCS 和 WCS 之间进行切换，那么在 HT 10 上的选择将会失效。

22.3 激活/取消虚拟键盘

在文件 "slguiconfig.ini" 中配置虚拟键盘。

步骤

1. 从目录 /**siemens**/sinumerik/hmi/template/cfg 中复制文件 "slguiconfig.ini"。
2. 把复件粘贴到目录：/**oem**/sinumerik/hmi/cfg 或 /**user**/sinumerik/hmi/cfg 下。
3. 在编辑器中打开文件。

4. 为激活或者取消键盘进行如下设置：
   – 激活虚拟键盘：
     段落 [TouchPanel], EnableTouch = true
     段落 [Keyboard], EnableVirtualKeyBoard = true
   – 取消虚拟键盘：
     段落 [TouchPanel], EnableTouch = true
     段落 [Keyboard], EnableVirtualKeyBoard = false
5. 双击输入框，以显示键盘。

22.4 设计用户专用按键说明

手持终端上的按键名称

HT 8 或 HT 10 中 CPF 菜单的文本（CPF：控制面板功能）可以使用各个国家语言的自有文本进行说明。

文本将创建在文件“slck_xxx.ts”中。
“xxx”= 按键名称的语种标识。

不仅可以通过 SINUMERIK Operate，还可以通过外部 PC 创建和编辑该文件。

说明

如果要在 PC 上编辑这些文件，要使用支持 UTF-8 编码的编辑器。

文件名称“xxx”中的语言标识

语言	语言标识
德语	deu
英语	eng
法语	fra
西班牙语	esp
意大利语	ita
汉语	chs

创建按键名称

天	含义
source	用户软键的名称。可从“SK_USERKEY1”至“SK_USERKEY16”，其中的名称不能更改。
comment	键盘布局的用户专用说明。
translation	应显示在按键上的文本。 • 每行最多可显示 10 个字符。 • 也可输入 2 行名称，通过“%n”换行。
remark	关于键盘布局的说明。
chars	字符数。每行最多可输入 10 个字符。
lines	行规定。可使用 2 行。

步骤

1. 可从如下目录中复制模板文件“oem_slck_deu.ts”：/**siemens**/sinumerik/hmi/template/lng。
2. 在目录 /**oem**/sinumerik/hmi/lng 或者 /**user**/sinumerik/hmi/lng 中保存或创建文件。
3. 为文件命名，比如对于德语文本可命名为：“slck_deu.ts”。
   若需创建其它语种的按键名称，必须为此语种创建一个自有文件。在文件名称中使用相应的语种标识并保存此文件。使用如上所规定的语种标识。
4. 打开此文件，并在区域 <message> 和 </message>（消息）中确定按键名称。
5. 重新启动操作界面。
   为了在程序运行时能显示按键名称，必须将文件转换至二进制格式。此转换仅在操作界面启动时进行。

按键名称举例

```
<!DOCTYPE TS><TS>
<context>
   <name>SlCkDialog</name
   <message>
     <source>SK_USERKEY1</source>
     <comment></comment>
     <translation>U1</translation>
     <remark>Usertaste 1</remark>
     <chars>10</chars>
```

```
    <lines>2</lines>
    <languageIndependent>true</languageIndependent>
  </message>
  <message>
    <source>SK_USERKEY2</source>
    <comment></comment>
    <translation>U2</translation>
    <remark>Usertaste 2</remark>
    <chars>10</chars>
    <lines>2</lines>
    <languageIndependent>true</languageIndependent>
  </message>
  <message>
  .....
  </message>
</context>
</TS>
```

22.5 配置用户专用键（U 按键）上的功能显示

功能

通过 PLC 可以显示可定义的用户键上的功能是否当前有效。为此，LED 可以表明软键的状态。

功能在文件“slckcpf.ini”中配置。

接口信号

PLC 位位于 PLC-HT 8/HT 10 接口的输出映像区内，和输入映像区的位类似。

MCP1（或 MCP2）上的信号 接口 PLC → HT 8/HT 10								
字节	位 7	位 6	位 5	位 4	位 3	位 2	位 1	位 0
AB n + 1		U4	U3		U2	U1		

MCP1（或 MCP2）上的信号 接口 PLC → HT 8/HT 10								
AB n + 4	U9	U10	U11	U12	U13	U14	U15	U16
AB n + 5		U8	U7	U6	U5			

在文件“slckcpf.ini”中调节显示

段	说明	
UserKeyLEDIcon	Icon 文件的名称。 标准条目：led_green.png	
	OFF	取消图标的显示。
	PRESSED	通过软键的按下状态表示功能当前有效。
UserKeyLEDIconAlignment	指定图标的位置。 标准位置：AlignLeft | AlignTop 可以组合水平和垂直排列。必须用符号"|" 加以分隔。 可以采用以下排列方式：	
	AlignLeft	左对齐
	AlignRight	右对齐
	AlignHCenter	水平居中
	AlignTop	上对齐
	AlignBottom	下对齐
	AlignVCenter	垂直居中
UserKeyLEDMap	指定输出映像区的起始地址。 条目的格式为："DBx.DBBy", "ABx", "MBx"。 缺省设置：起始地址由 DB7 MCP1Out（或 MCP2Out）确定。	
U1LED ... U16LED VarIncLED SBLLED WCSLED	通过这个条目可以定义和输出映像区不同的状态位地址。 **提示：**此处不考虑 HT 8 或 HT 10 输出映像区产生的、一个或多个软键的偏移。	
showVarIncLED	true	在增量模式激活时，在软键“[VAR]”上也显示图标。
	false	不显示图标。

段	说明	
showSBLLED	true	在单程序段模式激活时，在软键“单程序段”上也显示图标。
	false	不显示图标。
showWCSLED	true	在单程序段模式激活时，在软键“单程序段”上也显示图标。
	false	不显示图标。

1. 可从如下目录中复制模板文件“slckcpf.ini”：/**siemens**/sinumerik/hmi/template/cfg
2. 将文件保存到目录/**oem**/sinumerik/hmi/cfg 或/**user**/sinumerik/hmi/cfg。
3. 希望使用自己的图标时，请将它以适合 HT 8 或 HT 10 的分辨率放置在以下目录中：例如 /**oem**/sinumerik/hmi/ico/ico640 bzw. /**user**/sinumerik/hmi/ico/ico640。
4. 打开文件，修改设置。

模板文件“slckcpf.ini”

配置模板文件

```
; HT8/HT10 控制面板功能菜单
;
; 要激活设置时，删除
; 行首的注释符';'
;; 用户软键的 LED 显示设置
[UserKeyLED]
; LED 图标的文件名称
;UserKeyLEDIcon = led_green.png
; LED 图标的排列
;UserKeyLEDIconAlignment= AlignLeft | AlignTop

; 采用下一个 LED 映射区起始地址，而不是计算 DB7.MCP1Out
;UserKeyLEDMap = AB0
; 作以下设置，采用状态位，而不是特定软键的 LED 映射
;U1LED=/channel/parameter/R[U1,1]
```

```
;....
;U16LED=/channel/parameter/R[U1,16]
;VarIncLED = DB11.DBX8.5
;SBLLED = DB21.DBX0.4
;WCSLED = DB19.DBX0.7

; 显示增量软键的 LED
;showVarIncLED = true
; 显示单程序段软键的 LED
;showSBLLED = true
; 显示 wcs/mcs 软键的 LED
;showWCSLED = true
```

缩略语列表

A

缩写	含义
ASCII	American Standard Code for Information Interchange: 美国信息互换标准码
BAG	操作方式组
BTSS	操作面板接口
CNC	Computerized Numerical Control: 计算机数控系统
CEC	Cross Error Compensation:垂度补偿
CP	Communication Processor
DCP	Discovery and basic Configuration Protocol: ProfiNet 协议
DHCP	Dynamic Host Configuration Protocol
DIN	Deutsche Industrie Norm
DNS	Domain Name System
DIR	Directory: 目录
DRAM	Dynamic Random Access Memory
DRF	Differential Resolver Function:微分旋转变压器功能（手轮）
EES	Execution from External Storage
ERTEC	Enhanced Real Time Ethernet Controller
ESR	扩展的停止和退回
ETC	ETC 键">"；相同菜单中的扩展软键栏
FRAME	数据段（框架）
FIFO	First In - First Out: 数据进出存储器的方式。
GDIR	Global Directory: 全局零件程序存储器
GUD	Global User Data: 全局用户数据
GUI	Graphical User Interface
HSA	主主轴驱动
HSC	High Speed Cutting: 高速加工
ILC	Intelligent Load Control: 智能负载适配
INC	Increment: 增量尺寸

缩写	含义
INI	Initializing Data: 初始化数据
IP	Internet Protocol
IPO	插补器
ISO	International Standard Organization
JOG	Jogging: 点动模式
LED	Light Emitting Diode: 发光二极管
LLDP	Link Layer Discovery Protocol
LUD	Local User Data: 局部用户数据
MAC	Media Access Control
MB	Megabyte
MD	机床数据
MDA	Manual Data Automatic: 手动输入，自动运行
MCS	机床坐标系
MLFB	Maschinenlesbare Fabrikatbezeichnung
MPF	Main Program File: NC 零件程序（主程序）
MCP	Machine Control Panel: 机床控制面板
NC	Numerical Control: 数控系统
NCK	Numerical Control Kernel: 带有程序段处理、运行范围等等的数控内核
NCU	Numerical Control Unit: NCK 硬件单元
NFS	Network File System
NTP	Network Time Protocol
NV	零点偏移
OEM	Original Equipment Manufacturer
OID	Object Identifier: 信息标签
OP	Operator Panel
PCU	Programmable Control Unit
PI 服务	“程序-实例”服务
PG	编程器
PLC	Programmable Logic Control
RCS	Remote Control System
RFC	Request for Comments

缩写	含义
REF	返回参考点功能
REPOS	再定位功能
ROV	Rapid Override: 快进补偿
RPA	R-Parameter Active: NCK 中用于 R 参数的存储器区域
SBL	Single Block: 单程序段
SD	设定数据
SDB	系统数据块
SEA	Setting Data Active: 设定数据符号（文件类型）
SK	Softkey
SMB	Server Message Block
SNMP	Simple Network Management Protocol
SPF	Sub Program File: 子程序
SRAM	Static Random Access Memory: 静态存储器（缓存）
SSH	Secure Shell: 网络协议
SW	软件
SYF	System Files: 系统文件
TCU	Thin Client Unit
TEA	Testing Data Active: 机床数据标识
TMA	Tool Magazine Active: 刀库数据
TO	Tool Offset: 刀具补偿
TOA	Tool Offset Active: 刀具补偿符号（文件类型）
UDP	User Datagram Protocol: 网络协议
UFR	User Frame
VNC	Virtual Network Computing
VSA	进给驱动
WCS	工件坐标系
WZV	刀具管理

索引

3

3D 模拟, 177, 183

A

Access MyMachine, 28, 38
action.com, 338
action.log, 338
actlog.ini, 342
actual_actionlog.com, 334
actual_crashlog.com, 334
alarmtext_conversion.log, 210, 213, 216, 225, 229
AS-i
    配置诊断, 379
    诊断, 377
    组件, 378

C

config.ini, 127
crash.com, 338
crash.log, 338
CSV 文件, 329
CUST_782.SPF, 416, 556
CUST_800.SPF, 416
CUST_832.SPF, 416, 562
CUST_CLAMP, 417
CUST_M6, 417, 419
CUST_MEACYC
    结构程序, 570
CUST_MEACYC.SPF, 416
CUST_MEAPROT.SPF, 416, 605
CUST_MULTICHAN, 417, 424
CUST_T, 417, 419
CUST_TECHCYC.SPF, 416
    调整, 421
CYCLE495, 470
CYCLE60, 426
CYCLE63, 426
CYCLE64, 426
CYCLE782, 553
CYCLE79, 426
CYCLE800
    结构程序, 529
CYCLE832
    臻优曲面（Top Surface）, 558
CYCLE84, 415
CYCLE840, 415
CYCLE930, 450
CYCLE950, 450
CYCLE951, 450
CYCLE952, 450
CYCLE99, 450
CYCLE9960, 535

D

dealer.xml, 324
DHCP 服务器 ID, 288
Display Manager
    Display Configuration, 86
    激活, 87

E

Ethernet / PROFINET
    跟踪参数, 291
    网络跟踪, 290
    显示信息, 289
    诊断, 288
EXIT
    显示软键, 49
EXTCALL
    关闭/打开, 120

G

G 代码组
    设置显示, 157
G 功能组 59, 561

H

Handheld Terminal
    slguiconfig.ini, 649
    按键名称, 650
    激活、取消虚拟键盘, 649
    用户键上的 LED 灯, 652
HMI PRO RT, 375
HMI 跟踪, 343
HMI 重启
    显示软键, 49
    执行, 50

HOLES2, 426
HT 10
    运行键, 649
HT 8
    slguiconfig.ini, 49
    激活、取消虚拟键盘, 49
HT 8
    oem_sljk_deu.ts, 646
    sljkconfig.ini, 644
    设计运行键, 643

I

identSNAPSHOT, 321

M

ma_types.xml, 325
Manual Machine
    激活功能, 169
    设置, 169

N

NC/PLC 变量
    显示, 297
    修改, 299
NCU 连接
    设置，在 PCU 中, 641
    设置，在 Windows 中, 641
netnames.ini, 126

O

OEM 子目录, 636
oem.xml, 325
oem_ae_database_conversion.log, 220
oem_alarms_deu.ts, 208
oem_indextexts_deu.ts, 211
oem_slaedatabase.xml, 217
oem_slaesvcadapconf.xml, 209
oem_slck_deu.ts, 651
oem_sljk_deu.ts, 646
oem_text_conversion.log, 210, 229
OEMFrame 应用程序
    FindWindow 程序应用, 637
    集成, 617
    扩展参数设置, 626
    配置, 617
oemsubs.xml, 325

P

PDF 帮助文件
    语言切换, 402
PDF 文件
    语言切换, 402
PLC 硬件升级存档
    创建, 249
PROFIBUS
    DP 从站, 346
    连接, 344
PROFIBUS/PROFINET 诊断, 375, 376
PROG_EVENT.SPF, 344, 417
    扩展, 419

R

RCS Commander, 28

S

ShopMill
    用于铣削的设置, 427
ShopTurn
    端面加工, 463
    设置副主轴, 445
    斜置 Y 轴, 464
    用于车削的设置, 439
    柱面转换, 458
Sidescreen
    固有设计, 81
    激活, 76
    显示 ABC 键盘, 78
    显示翻译, 83
    显示机床控制面板, 79
    显示摄像头, 84
slaesvcadapconf.xml, 209, 212, 214, 222, 241
slaesvcconf.xml, 215, 224
slckcpf.ini, 654
slhlp.xml, 390
slmagcodeconfig.ini, 157
SLOT2, 426
sltlprodiaghmi.ini, 378
Source-ID, 227, 228
Source-URL, 227, 228
STAT, 162

T

TextPad, 28
TRAANG, 466
    在 ShopTurn 中, 464
    在车削中, 464
TRACON, 466
TRACYL, 467
    在 ShopTurn 中, 458
    在车削中, 456
TRANSMIT
    带 Y 轴, 461
    在 ShopTurn 中, 463
    在车削中, 460
TU, 162

U

USB 键盘, 28
USB 设备, 28
user.xml, 325

V

VNC 操作区域
    设置, 57

W

WEB 浏览器操作区域
    设置, 61
Window Mode, 638
WinSCP, 28, 38

安

安装
    互动, 640
    静默, 640

保

保存装调数据, 258

报

报警编号范围, 226
报警故障文件, 229
报警记录
    过滤器属性, 232
    设置, 231
报警索引文本文件
    创建文本, 211
报警文本
    标准参数说明, 228
    创建, 205, 208
    创建外语文本, 207
    创建文件, 208
    更改颜色, 216
    取消激活警告提示, 241
    替换, 221
    替换缺省报警文本, 221
    选择字体颜色, 205
    支持的语言, 614
    注册文件, 209
报警文本的标准参数说明, 228
报警用索引文本文件
    创建文件, 211
    注册, 212

边

边缘平滑, 638

编

编程 G 代码程序, 169
编辑 Enum 值, 144
编辑 xml 文件, 37

变

变量屏幕, 301

并

并行操作界面, 54

操

操作面板
    配置, 268
操作区域 “加工”, 155
操作日志
    actlog.ini, 342
    保存日志文件, 338
    打开日志文件, 337
    日志文件, 339

设置, 333
设置日志大小, 342
搜索事件, 337

测

测量工件
在车削中, 599
在铣削中, 575, 591
测头
测量刀具, 565
测量工件, 565
检查程序示例, 567
检查功能, 566

查

查找
日志条目, 333
查找方式
配置, 165

车

车削
ShopTurn 中的设置, 439
测量工件, 599
刀具测量, 584
端面加工, 460
工艺循环, 450
示例，设置车床, 455
显示软键, 408
斜置 Y 轴, 464
柱面转换, 456
转速限值, 467
车削轮廓, 450

成

成型磨削, 470

程

程序测试
激活, 161
取消, 161
程序访问
通过保护等级设置, 72
程序列表
创建, 73
设计软键, 72

创

创建系统崩溃日志文件, 334
创建在线帮助
创建 HTML 文件, 393
创建 XML 文件, 395
目录创建, 396
替换缺省的在线帮助, 396

存

存档
创建, 247
创建用于热线咨询, 253
登录数据处理, 56
读入, 248
行校验和, 244

带

带有图标的状态栏, 172

刀

刀具测量
在车削中, 584
在铣削中, 578
刀具夹紧角度, 547

雕

雕刻, 426

叠

叠加
配置, 173

定

定义通道组, 125

端

端面齿, 478
端面加工
    在 ShopTurn 中, 463
    在车削中, 460

对

对驱动系统进行调试, 203

多

多边形, 426
多个不同工件的加工（选件）, 435

二

二维码, 48

访

访问等级, 63
访问权限
    单独, 66
    全局, 67
    生效, 67
    远程访问, 67

服

服务概览, 302
    轴和驱动, 304

复

复位
    扩展, 140

高

高速设定, 558

跟

跟踪
    PLC/NC/伺服变量, 359
    保存会话, 357
    保存选项, 368
    编辑比例, 372
    编辑会话, 361
    编辑图形视图, 370
    变量属性, 361
    步骤, 355
    插入变量, 360
    查找变量, 359
    创建会话, 356
    定位光标 A/B, 374
    定义缩放区域, 373
    会话, 360
    将数据保存到控制单元上, 367
    启动会话, 358
    启动记录, 369
    驱动变量, 359
    筛选变量, 359
    设置（PLC、NC、伺服）, 365
    设置（驱动）, 366
    设置栅格线, 368
    数据采集, 365
    缩放曲线图, 372
    替换变量, 360
    停止记录, 369
    显示变量的详细信息, 364
    显示峰峰值, 374
    显示峰谷值, 374
    选择变量, 360, 371
    一览, 354
    载入会话, 358

更

更快速的程序段搜索, 166

工

工业服务平台, 321
工艺
    车削, 450
    回转, 471
    平面磨削, 469
    外圆磨削, 469
    铣削, 425
    钻削, 414
工作场所安全, 65

攻

攻丝, 415

故

故障分析, 275

拐

拐角车削, 450

管

管理数据, 146

回

回转
传统 Toolcarrier, 493
创建角度刀具, 477
定义数据组, 476
回退, 489
回转轴, 497
基于运动链的 Toolcarrier, 490
激活, 472
配置输入屏幕, 472
示例 HEAD_1, 498
示例 HEAD_2（传统）, 503
示例 HEAD_2（运动链）, 500
示例 MIXED_45（传统）, 511
示例 MIXED_45（运动链）, 509
示例 TABLE_45（传统）, 507
示例 TABLE_45（运动链）, 505
示例 TABLE_5（传统）, 513
示例：带 B 轴的车床（传统）, 520
示例：带 B 轴的车床（运动链）, 515
示例：带 B 轴铣床的车削加工, 523
调试机床运动, 483
显示方式, 485

机

机床数据
编辑 Enum 值, 144
编辑十六进制的数值, 142
配置明码文本, 153
输入 BICO 数值, 143
说明, 134
搜索, 145
显示, 134
显示过滤器, 140
一览, 133
轴选择, 136
机床信息, 321
机床专用信息, 327
创建模板, 323
扫描, 322

基

基于子站的网络诊断, 272

激

激活、取消窗口模式, 638
激活键盘, 49, 650

记

记录测量
AUTOMATIK 时的设置, 604
JOG 时的设置, 604

加

加工时间记录
激活, 174

夹

夹具
功能设定, 185

键

键盘
设置, 48

截

截屏
创建, 320
打开, 321
复制, 320

可

可定向刀架, 601

口

口令
删除, 65
设置, 64
修改, 65

轮

轮廓槽式车削, 450
轮廓铣削, 426

螺

螺纹切削, 450

面

面板清洁模式, 53

模

模板文件, 35
模拟
车削工艺, 179
功能的基本条件, 178
换刀时间, 182
磨削工艺, 180
取消, 181
铣削工艺, 179
循环应用, 418
运动链, 182
模拟主轴, 190

目

目录结构, 31

配

配置操作站, 126

平

平面磨削, 469
显示软键, 408

屏

屏幕
保护, 52
关闭显示, 52
关闭显示的时间, 52

启

启动 SINUMERIK Operate
IPC, 27
NCU, 27
PCU, 27
Windows 操作系统下, 27

切

切端面齿, 547

清

清洁模式, 53

驱

驱动参数
机床数据, 134
显示/更改, 139
驱动器
逻辑驱动器, 119
全局零件程序存储器, 121
设置, 119
设置 EXTCALL, 120
驱动系统
诊断, 347
诊断 - 详细信息, 348
驱动信息
参数描述, 311
显示, 311
驱动专用机床数据, 134

取

取消激活键盘, 49, 650

日

日期
格式选择, 42
设置, 44
日志, 330
查找日志, 333
创建条目, 332
导入模板, 326
机床信息, 321
确定调试, 331
输出, 327
日志文件
操作日志, 339

软

软键
访问等级, 71
恢复设置, 71

设

设备注册, 48
设定数据
搜索, 145
通道选择, 138
显示, 137
一览, 133
轴选择, 138
设置 NCU 的 IP 地址, 641
设置带铣刀的车床, 454
设置副主轴
ShopTurn, 445

时

时间
格式选择, 43
设置, 44
时间服务器, 46
时区, 47

实

实际值窗口
设置字体大小, 155

使

使用角度刀具, 477

示

示教, 163

试

试用授权, 114

输

输入 BICO 数值, 143
输入注释
Ini 文件, 38
xml 文件, 38

数

数据备份, 243

搜

搜索
机床数据, 145
设定数据, 145
显示机床数据, 145

替

替换缺省报警文本
创建文本, 221
创建文件, 221
注册报警文本文件, 222
替换商标, 156

添

添加引导启动画面, 53

调

调试
导入原始数据存档。, 253

调试存档
创建, 245, 259
读入, 248
调整 JOG 运行方式下的测量, 572
调整刀具测量
车削, 600
铣削时, 593
铣削示例, 594
调整工件测量, 588
调整设计, 52

通

通道菜单
定义通道组, 125
配置, 123
配置操作站, 126
配置整体结构, 125
通道运行信息, 159
标准设计, 160
通道专用机床数据, 134
通道专用设定数据, 137
通用机床数据, 134
通用设定数据, 137

同

同步记录, 183
激活处理时间, 184
选件, 182
同时测量, 602

退

退出, 642

拓

拓扑结构, 293

外

外圆磨削, 469
显示软键, 408

网

网络
配置, 261
网络适配器, 273
配置, 285
网络诊断, 272

尾

尾座
选件, 182

文

文本可读性, 638
文件
打开, 35
复制, 35
剪切, 37
删除, 37
使用 WinSCP 复制, 38
外部编辑, 37
在预览窗口显示, 37
粘贴, 35
重命名, 36

铣

铣床上的车削
CUST_800, 549
程序段搜索, 552
刀具夹紧角度, 547
复位后的特性, 545
切端面齿, 547
设置, 537
设置坐标系, 548
运动转换, 543
重新上电后的特性, 546
铣削
ShopMill 中的设置, 427
测量工件, 575, 591
刀具测量, 578
示例，铣床设置, 432
铣床上的车削, 536
显示软键, 408
用于位置模式的旋转方向, 428
柱面转换, 430

系

系统负载, 319

显

显示 AS-i, 380
显示分组, 141
显示过滤器
    机床数据, 140
显示机床数据, 136
    搜索, 145

消

消息文本
    创建, 213
    零件程序, 213
    为多个通道创建, 215

斜

斜置 Y 轴
    在 ShopTurn 中, 464
    在车削中, 464

许

许可证, 111
许可证密码
    读入, 116
    输入, 116
许可证密钥
    从 CF 卡上备份, 117
许可证需求
    输出, 115

旋

旋转方向
    副主轴, 446
    主主轴, 439

循

循环版本
    显示, 608
    指定, 609
循环报警, 482

硬

硬件组件
    扫描, 328
    添加, 329

用

用户视图
    编辑, 151
    创建, 149
    导入, 148
    删除, 152

语

语言
    确定选择模式, 41
语言标识, 614, 615
语言扩展
    安装, 613
    卸载, 614

圆

圆弧槽, 426
圆位置模式, 426

远

远程诊断
    调整, 350

运

运动
    测量, 535
    带端面齿, 478
    检查列表, 483
    调试示例, 498
运行状态, 302

诊

诊断
    AS-i, 380
    Ethernet / PROFINET, 288
    HMI 跟踪, 343

智

智能负载适配, 553

轴

轴诊断, 305
轴专用机床数据, 134
轴专用设定数据, 137

主

主轴负载
    显示剩余时间, 48
主轴控制, 189
主轴诊断, 191
    电机温度传感器, 195
    附加温度传感器, 196
    夹紧时间统计, 201
    夹紧状态, 198
    温度, 193
    温度柱形图, 196
    物流数据, 202
    诊断统计, 200
    主轴负载率, 197
    转速限值, 199

柱

柱面转换
    无槽壁补偿, 457
    铣削，示例, 432
    有槽壁补偿, 458
    在 ShopTurn 中, 458
    在车削中, 456
    在铣削中, 430

专

专家参数, 142

转

转换
    用于铣削的设置, 433

装

装料适配, 553

状

状态显示
    插入商标, 156

钻

钻削, 414

坐

坐标系
    MD52000, 409

SIEMENS

SINUMERIK

SINUMERIK 840D sl
Easy XML

编程手册

前言

基本安全说明 1

创建用户对话框 2

创建调试对话框 3

适用于：

CNC 软件　　版本 4.94

10/2020
6FC5397-1DP40-6RA4

法律资讯

警告提示系统

为了您的人身安全以及避免财产损失，必须注意本手册中的提示。人身安全的提示用一个警告三角表示，仅与财产损失有关的提示不带警告三角。警告提示根据危险等级由高到低如下表示。

⚠ 危险

表示如果不采取相应的小心措施，**将会**导致死亡或者严重的人身伤害。

⚠ 警告

表示如果不采取相应的小心措施，**可能**导致死亡或者严重的人身伤害。

⚠ 小心

表示如果不采取相应的小心措施，可能导致轻微的人身伤害。

注意

表示如果不采取相应的小心措施，可能导致财产损失。

当出现多个危险等级的情况下，每次总是使用最高等级的警告提示。如果在某个警告提示中带有警告可能导致人身伤害的警告三角，则可能在该警告提示中另外还附带有可能导致财产损失的警告。

合格的专业人员

本文件所属的产品/系统只允许由符合各项工作要求的**合格人员**进行操作。其操作必须遵照各自附带的文件说明，特别是其中的安全及警告提示。 由于具备相关培训及经验，合格人员可以察觉本产品/系统的风险，并避免可能的危险。

按规定使用 Siemens 产品

请注意下列说明：

⚠ 警告

Siemens 产品只允许用于目录和相关技术文件中规定的使用情况。如果要使用其他公司的产品和组件，必须得到 Siemens 推荐和允许。正确的运输、储存、组装、装配、安装、调试、操作和维护是产品安全、正常运行的前提。必须保证允许的环境条件。必须注意相关文件中的提示。

商标

所有带有标记符号 ® 的都是 Siemens AG 的注册商标。本印刷品中的其他符号可能是一些其他商标。若第三方出于自身目的使用这些商标，将侵害其所有者的权利。

责任免除

我们已对印刷品中所述内容与硬件和软件的一致性作过检查。然而不排除存在偏差的可能性，因此我们不保证印刷品中所述内容与硬件和软件完全一致。印刷品中的数据都按规定经过检测，必要的修正值包含在下一版本中。

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
德国

文件订购号: 6FC5397-1DP40-6RA4
Ⓟ 10/2020 本公司保留更改的权利

Copyright © Siemens AG 2014 - 2020.
保留所有权利

前言

SINUMERIK 文献

SINUMERIK 文档分为以下几个类别：

- 通用文档/产品样本
- 用户文档
- 制造商/服务文档

其它信息

访问下面的网址 (https://support.industry.siemens.com/cs/de/en/view/108464614)获取有关该主题的信息：

- 订购文档/查看印刷品一览
- 进入下载文档的链接
- 使用在线文档（查找搜索手册/信息）

如果您对技术文档有疑问（例如建议、修改），请发送一份电子邮件到以下地址 (mailto:docu.motioncontrol@siemens.com)。

mySupport/文档

您可以访问下面的网址 (https://support.industry.siemens.com/My/cn/zh/documentation)，了解如何随意组合西门子文档内容，再结合机器，创建自己的机器文档。

培训

通过以下地址 (https://www.siemens.com/sitrain)可获取有关 SITRAIN 的信息 - 西门子为驱动和自动化产品、系统和解决方案制定的培训。

常见问题

常见问题（FAQ）请参见产品支持 (https://support.industry.siemens.com/cs/de/en/ps/faq) 下的服务&支持页面。

SINUMERIK

有关 SINUMERIK 的信息请访问以下网址 (https://www.siemens.com/sinumerik)。

目标用户

该手册供以下人员使用：

- 编程人员
- 设计人员

优点

此编程手册可协助目标用户，使用基于 XML 的脚本元素并根据客户和应用的需要进行操作界面的自定义设计。

标准功能范畴

在该编程手册中描述了标准的功能范畴。机床制造商增添或者更改的功能，由机床制造商资料进行说明。

控制系统有可能执行本文档中未描述的某些功能。但这并不表示在交付系统时必须提供这些功能以及相关的维修服务。

同样，因为只是概要，所以该文献不包括全部类型产品的所有详细信息，也无法考虑到安装、运行和维修中可能出现的各种情况。

有关数据保护准则的说明

西门子遵守数据保护准则，特别是数据最小化原则（privacy by design）。对于该产品的具体含义是：

产品不会处理/存储个人相关数据，技术功能数据除外（例如时间戳）。用户如果将此类数据与其他数据（例如排班表）关联或者将个人相关数据存储在同一介质（例如硬盘）上而产生个人相关性，则应由用户自行确保遵循数据安全法规。

技术支持

访问网址 (https://support.industry.siemens.com/cs/cn/zh/sc/2090)下的“联系方式”， 您便可以获取各个国家技术支持的电话号码。

如需咨询技术疑问，请使用“支持请求”一栏下的在线表格。

目录

前言 **3**

1 基本安全说明 **9**

1.1 一般安全说明 9

1.2 静电场或静电放电可导致设备损坏 12

1.3 应用示例的质保规定 13

1.4 安全性信息 14

1.5 驱动系统（电气传动系统）的遗留风险 15

2 创建用户对话框 **17**

2.1 功能范围 17

2.2 配置基础 19

2.3 设计文件 24

2.4 设计文件的结构 28

2.5 语言相关性 34

2.6 XML 诊断 35

2.7 XML 标签 37

2.7.1 通用结构 37

2.7.2 指令/标签说明 38

2.7.3 颜色代码 70

2.7.4 专用 XML 句法 70

2.7.5 XML 特殊字符 70

2.7.6 运算符 72

2.7.7 系统变量 73

2.7.8 创建软键菜单和对话框窗口 74

2.8 创建用户菜单 138

2.8.1 创建加工循环窗口 138

2.8.2 替代符号 141

2.9 创建 struct_def.xml 文件 142

2.10 元素定址 144

2.10.1 PLC 定址 144

2.10.2 NC 变量定址 146

2.10.3 通道专用定址 146

2.10.4 在运行时生成 NC/PLC 地址 146

2.10.5 驱动组件的定址 147
2.10.6 示例：测定电机模块的 DO 号 149
2.10.7 机床数据和设定数据的定址 155
2.10.8 通道专用机床数据 156
2.10.9 用户数据的定址 157

2.11 预定义功能 158

2.12 多点触控操作 225
2.12.1 多点触控功能 225
2.12.2 手指手势编程 227
2.12.3 图形的手势控制 228
2.12.4 手势处理 231

2.13 自定义按钮 233
2.13.1 下压按钮 233
2.13.2 按钮的功能 235
2.13.2.1 下压按钮的子标签 235
2.13.2.2 下压按钮的属性 237
2.13.2.3 下压按钮的控制项变量 239
2.13.3 开关 on/off 244
2.13.4 开关的功能 245
2.13.4.1 开关的属性 245
2.13.4.2 开关的控制项变量 246
2.13.5 单选按钮 248
2.13.6 复选框 249
2.13.7 分组框 249
2.13.8 滚动区域 250

2.14 Sidescreen 的使用 253
2.14.1 Sidescreen 中的 Easy XML 253
2.14.2 集成 Sidescreen 对话框 254
2.14.3 语言和文本管理 255
2.14.4 Sidescreen 部件 256
2.14.4.1 Sidescreen 元素 256
2.14.4.2 Sidescreen 微体 257
2.14.4.3 Sidescreen 页面 259

2.15 Display Manager 应用 261
2.15.1 Display Manager 中的 Easy XML 261
2.15.2 将 Customer Area 整合至 Display Manager 菜单 261
2.15.3 Display Manager 小部件 263
2.15.4 Display Manager Sidescreen 页面 265

2.16 通过 PLC 硬键选择对话框 267

2.17 将用户窗口分配给预留软键 271
2.17.1 确定无需区分语种的软键文本 272
2.17.2 Easy XML 区域分配 273

2.17.3 标签描述 279
2.18 创建无需区分语种的文本 282
2.19 特定项目的文本文件 283
2.19.1 创建特定项目的文本文件 283
2.19.2 给定文本上下文 286
2.19.3 给定文本文件列表 289
2.20 恢复会话 291
3 创建调试对话框 293
3.1 功能概述 293
3.2 PLC 用户程序中的配置 296
3.3 操作界面上的显示画面 299
3.4 创建和语言相关的文本 300
3.5 功率部件应用示例 301
3.6 脚本语言 303
3.6.1 CONTROL_RESET 306
3.6.2 FILE 306
3.6.3 OPTION_MD 307
3.6.4 PLC_INTERFACE 308
3.6.5 POWER_OFF 309
3.6.6 WAITING 309
3.6.7 用于窗口的 XML 标签 310
3.6.8 SOFTKEY_OK, SOFTKEY_CANCEL 310
索引 313

基本安全说明 1

1.1 一般安全说明



⚠ 警告

其他能源可导致电击危险和生命危险

接触带电部件可能会造成人员重伤，甚至是死亡。

- 只有专业人员才允许在电气设备上作业。
- 在所有作业中必须遵守本国的安全规定。

通常有以下安全步骤：

1. 准备断电。通知会受断电影响的组员。
2. 给驱动系统断电并确保不会再次接通。
3. 请等待至警告牌上说明的放电时间届满。
4. 确认功率接口和安全接地连接无电压。
5. 确认辅助电压回路已断电。
6. 确认电机无法运动。
7. 检查其他所有危险的能源供给，例如：压缩空气、液压、水。将能源供给置于安全状态。
8. 确保正确的驱动系统已经完全闭锁。

结束作业后以相反的顺序恢复设备的就绪状态。



⚠ 警告

连接不合适的电源可导致电击危险

连接不合适的电源会导致可接触部件携带危险电压，从而导致人员重伤，甚至是死亡。

- 所有的连接和端子只允许使用可以提供 SELV(Safety Extra Low Voltage：安全低压) 或 PELV(Protective Extra Low Voltage：保护低压) 输出电压的电源。



⚠ 警告

设备损坏可导致电击危险

未按规定操作会导致设备损坏。设备损坏后，其外壳或裸露部件可能会带有危险电压，接触外壳或这些裸露部件可能会导致重伤或死亡。

- 在运输、存放和运行设备时应遵循技术数据中给定的限值。
- 不要使用已损坏的设备。



⚠ 警告

电缆屏蔽层未接地可导致电击危险

电缆屏蔽层未接地时，电容超临界耦合可能会出现致命的接触电压。

- 电缆屏蔽层和未使用的电缆芯线至少有一侧通过接地的外壳接地。



⚠ 警告

缺少接地可导致电击危险

防护等级 I 的设备缺少安全接地连接或连接出错时，在其裸露的部件上会留有高压，接触该部件会导致重伤或死亡。

- 按照规定对设备进行接地。

注意

使用不合适的螺丝刀可损坏设备

使用不合适的螺丝刀或者采用不恰当的拧紧操作都可能损坏设备上的螺钉。

- 请使用与螺钉头完全匹配的螺丝刀。
- 请使用技术文档中规定的扭矩拧紧螺钉。
- 请使用扭力扳手或者带动态扭矩传感器和转速限制功能的机械式高精度螺丝刀。

⚠ 警告

内置型设备内可引起火灾

发生火灾时，内置型设备的外壳无法避免火苗和烟雾冒出。这可能导致人员重伤或财产损失。

- 将内置型设备安装在合适的金属控制柜中，从而保护人员免受火苗和烟雾伤害，或者对人员采取其他合适的防护措施。
- 确保烟雾只能经所设安全通道排出。

⚠ 警告

无线电设备或移动电话可导致机器意外运动

在设备的无屏蔽范围内使用无线电设备或移动电话，会干扰设备功能。功能异常会对设备功能安全产生影响并能导致人员伤亡或财产损失。

- 大约距离组件 20 cm 时，请关闭无线电设备或移动电话。
- 仅在已关闭的设备上使用“SIEMENS Industry Online Support App”。

⚠ 警告

通风空间不足可引起火灾

通风空间不足会导致过热，产生烟雾，引发火灾，从而造成人身伤害。这可能就是导致重伤或死亡的原因。此外，设备/系统故障率可能会因此升高，使用寿命缩短。

- 组件之间应保持规定的最小间距，以便通风。

注意

安装位置错误可导致过热

安装位置错误时，设备可能会过热并因此损坏。

- 只允许在规定的安装位置上运行设备。

⚠ 警告

安全功能失效可导致机器意外运动

无效的或不适合的安全功能可引起机器意外运动，可能导致重伤或死亡。

- 调试前请注意相关产品文档中的信息。
- 对整个系统和所有安全相关的组件进行安全监控，以确保安全功能。
- 进行适当设置，以确保所使用的安全功能是与驱动任务和自动化任务相匹配并激活的。
- 执行功能测试。
- 在确保了机器的的安全功能能正常工作后，才开始投入生产。

说明

Safety Integrated 功能的重要安全说明

使用 Safety Integrated 功能时务必要注意 Safety Integrated 手册中的安全说明。

⚠ 警告

因参数设置错误或修改参数设置引起机器故障

参数设置错误可导致机器出现故障，从而导致人员重伤或死亡。

- 采取保护措施，防止未经授权的参数设置。
- 采取适当措施（如驻停或急停）处理可能出现的故障。

1.2 静电场或静电放电可导致设备损坏

静电敏感元器件 (ESD) 是可被静电场或静电放电损坏的元器件、集成电路、电路板或设备。



注意

静电场或静电放电可导致设备损坏

电场或静电放电可能会损坏单个元件、集成电路、模块或设备，从而导致功能故障。

- 仅允许使用原始产品包装或其他合适的包装材料（例如：导电的泡沫橡胶或铝箔）包装、存储、运输和发运电子元件、模块和设备。
- 只有采取了以下接地措施之一，才允许接触元件、模块和设备：
  - 佩戴防静电腕带
  - 在带有导电地板的防静电区域中穿着防静电鞋或配带防静电接地带
- 电子元件、模块或设备只能放置在导电性的垫板上（带防静电垫板的工作台、导电的防静电泡沫材料、防静电包装袋、防静电运输容器）。

1.3 应用示例的质保规定

应用示例在组态和配置以及各种突发事件方面对设备没有强制约束力，无需一一遵循。应用示例不会提供客户专用的解决方案，仅在典型任务设置中提供保护。

用户自行负责上述产品的规范运行事宜。应用示例并没有解除您在应用、安装、运行和维护时确保安全环境的责任。

1.4 安全性信息

Siemens 为其产品及解决方案提供了工业信息安全功能，以支持工厂、系统、机器和网络的安全运行。

为了防止工厂、系统、机器和网络受到网络攻击，需要实施并持续维护先进且全面的工业信息安全保护机制。Siemens 的产品和解决方案构成此类概念的其中一个要素。

客户负责防止其工厂、系统、机器和网络受到未经授权的访问。只有在有必要连接时并仅在采取适当安全措施（例如，防火墙和/或网络分段）的情况下，才能将该等系统、机器和组件连接到企业网络或 Internet。

关于可采取的工业信息安全措施的更多信息，请访问 https://www.siemens.com/industrialsecurity (https://www.siemens.com/industrialsecurity)。

Siemens 不断对产品和解决方案进行开发和完善以提高安全性。Siemens 强烈建议您及时更新产品并始终使用最新产品版本。如果使用的产品版本不再受支持，或者未能应用最新的更新程序，客户遭受网络攻击的风险会增加。

要及时了解有关产品更新的信息，请订阅 Siemens 工业信息安全 RSS 源，网址为 https://www.siemens.com/industrialsecurity (https://new.siemens.com/global/en/products/services/cert.html#Subscriptions)。

其他信息请上网查找：

工业安全功能选型手册 (https://support.industry.siemens.com/cs/cn/zh/view/108862708/en)

⚠ 警告

篡改软件会引起不安全的驱动状态

篡改软件（如：病毒、木马、蠕虫等）可使设备处于不安全的运行状态，从而可能导致死亡、重伤和财产损失。

- 总是使用最新版本的软件。
- 将自动化和驱动组件集成到设备或机器上的整套先进工业信息安全方案中。
- 全面考虑整套工业信息安全方案中使用的所有产品。
- 采取相应的保护措施（如：使用杀毒软件）防止移动存储设备中的文件受到恶意软件的破坏。
- 在调试结束后，检查所有和安全相关的设置。

1.5 驱动系统（电气传动系统）的遗留风险

机器或设备制造商在依据相应的本地指令（比如欧盟机械指令）对机器或设备进行风险评估时，必须注意驱动系统的控制组件和驱动组件会产生以下遗留风险：

1. 调试、运行、维护和维修时机器或设备部件意外运行，原因（举例）：
   - 编码器、控制器、执行器和连接器中出现了硬件故障和/或软件故障
   - 控制器和传动设备的响应时间
   - 运行和/或环境条件不符合规定
   - 凝露/导电杂质
   - 参数设置、编程、布线和安装出错
   - 在电子器件附近使用无线电装置/移动电话
   - 外部影响/损坏
   - X 射线辐射、电离辐射和宇宙辐射
2. 在出现故障时，组件内/外部出现异常温度、明火以及异常亮光、噪音、杂质、气体等，原因可能有：
   - 零件失灵
   - 软件故障
   - 运行和/或环境条件不符合规定
   - 外部影响/损坏
3. 危险的接触电压，原因（举例）：
   - 零件失灵
   - 静电充电感应
   - 旋转电机的感应电压
   - 运行和/或环境条件不符合规定
   - 凝露/导电杂质
   - 外部影响/损坏
4. 设备运行中产生的电场、磁场和电磁场可能会损坏近距离的心脏起搏器支架、医疗植入体或其它金属物。
5. 当不按照规定操作以及/或违规处理废弃组件时，会释放破坏环境的物质并且产生辐射。
6. 影响通讯系统，如中央控制发送器或通过电网进行的数据通讯

其它有关驱动系统组件产生的遗留风险的信息见用户技术文档的相关章节。

创建用户对话框 2

2.1 功能范围

概述

使用功能“创建用户对话框”可以保证开放性，它能够让用户设计出客户专用和应用专用的 SINUMERIK Operate 界面。

控制系统提供基于 XML 的脚本语言用于创建用户对话框。

该脚本语言可以在 SINUMERIK Operate 上的操作区 <CUSTOM> 中显示机床专用菜单和对话框窗口。

所有对话框窗口的构成都与语言无关。这种情况下系统会从随附的语言数据库中读取要显示的文本。

使用

已定义的 XML 指令可以实现下列特性：

1. 显示对话框并提供：
   - 软键
   - 变量
   - 文本和帮助文本
   - 图形和帮助画面
2. 通过以下方法调用对话框：
   - 按下（登入）软键
3. 动态重组对话框
   - 修改、删除软键
   - 定义并设计变量栏
   - 显示、更换、删除显示文本（和语言相关或无关）
   - 显示、更换、删除图形
   - 动态创建可执行脚本并集成到脚本处理过程中

4. 在进行以下操作时触发动作：
   – 显示对话框
   – 输入数值（变量）
   – 按下软键
   – 关闭对话框
5. 对话框间的数据交换
6. 变量
   – 读取（NC 变量、PLC 变量、用户变量）
   – 写入（NC 变量、PLC 变量、用户变量）
   – 和数学、比较或者逻辑运算符相连
7. 执行下列功能：
   – 子程序
   – 文件功能
   – PI 服务
8. 根据用户组考虑保护等级

关于脚本语言的有效单元（标签）的描述可参见章节 “XML 标签” (页 37)。

说明

以下章节“基础配置”对 XML（Extensible Markup Language 可扩展标记语言）的描述并不完整。更多的信息可查阅相应的专业文献。

2.2 配置基础

配置文件

新操作界面的说明存储在配置文件中。这些文件自动编译并显示屏幕上的结果。在供货时并不提供配置文件，因此必须首先创建或下载此文件。

可以使用 XML 编辑器或其他的文本编辑器来创建配置文件。

说明

文件名只能由小写字母构成。

菜单树的原理

多个相互关联的对话框构成了一个菜单树。如果能从一个对话框切换入另一个对话框，则表示这两个对话框间存在关联。通过此对话框内重新定义的水平或者垂直软键可以返回上级对话框或者进入任意一个对话框。

可以在登入菜单后面通过配置好的登入软键生成更多的菜单树：

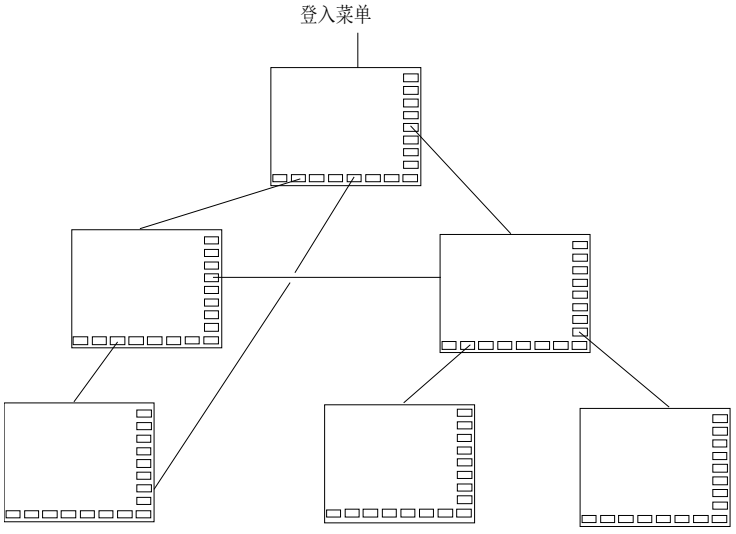


图 2-1 用户对话框菜单树

登入菜单

在文件“xmldial.xml”中使用名称“main”来定义登入菜单。登入菜单是操作流程自身的输出点。

使用“主”菜单可以与加载自身对话框或其他软键条连接在一起，而使用它们又可以执行其他动作。

下图显示了控制系统上的制造商目录“System CF-Card/oem/sinumerik/hm”。

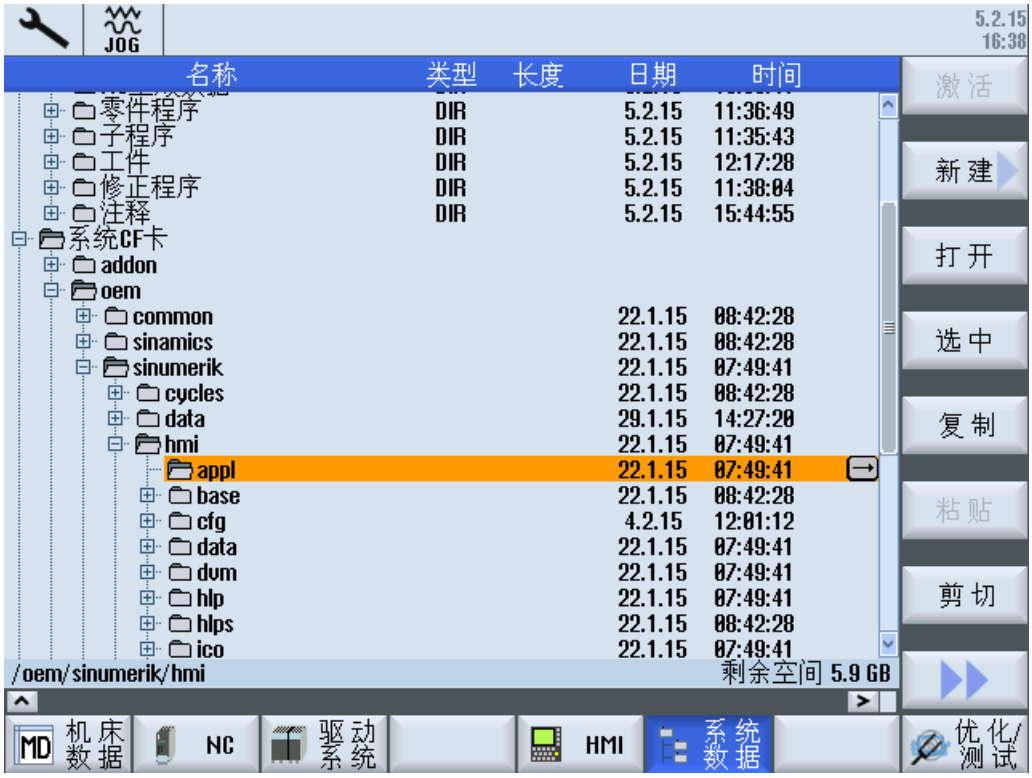


图 2-2　制造商目录

为了进行用户对话框的配置，需要使用控制系统中制造商目录“System CF-Card/oem/sinumerik/hmi”下的以下文件：

表格 2-1　用于配置的文件

文件类型	文件名称	含义	操作区调试 > 系统数据中的存储位置
脚本文件	“xmldial.xml”	标准文本文件 该脚本文件通过 XML 标签控制已配置软键菜单的映像以及 SINUMERIK Operate 上操作区 <CUSTOM> 中的对话框屏幕。	制造商目录 > 在用于应用的子目录“appl”中
用户专用脚本文件	“xxx.xml”	可自由选择的文件名称。这些名称的引用通过相应的 INI 文件创建。	

文件类型	文件名称	含义	操作区调试 > 系统数据中的存储位置
文本文件	“oem_xml_screens_xxx.ts”	用于用户操作界面的标准文本文件 该文本文件包含有用于单个语言菜单和对话框屏幕的文本。	制造商目录 > 在用于所需语言的子目录“lng”中
用户专用文本文件	“yyy_xxx.ts”	可自由选择的文件名称。这些名称的引用通过 DialogGUI 标签、菜单或窗体的相应属性创建。相关描述请参见章节 “特定项目的文本文件 (页 283)”。 Sidescreen 或 Diaplay Manager 应用中的文本文件名称由页面或小部件名称衍生而来。示例请参见章节 “语言和文本管理 (页 255)”。	
位图		控制系统支持 BMP 格式和 PNG 格式。	制造商目录 > 在子目录“ico”中
XML 文件，在控制文件 “xmldial.xml”中插入了 XML 标签 “INCLUDE” 。	例如 “machine_settings.xml”	该文件同样包含用来在 SINUMERIK Operate 上显示对话框窗口和参数的编程指令。	制造商目录 > 在用于应用的子目录 “appl”中

Easy XML 连接点

以下连接点可用于 Easy XML 脚本：

硬键/软键	连接点

软键“操作菜单”	标准脚本文件名“xmldial.xml”
	在文件“slamconfig.ini”中激活
	
	示例：
	`[CustomXML]`
	`TextId=SL_AM_CUSTOM`
	`SidescreenTextId=SL_SIDESCREEN_CUSTOM`
	`TextFile=slam`
	`TextContext=SlAmAreaMenu`
	`SoftkeyPosition=12`
	`Visible=true`

用户菜单	标准脚本文件名“menu_user.xml”
功能菜单	标准脚本文件名“menu_function.xml”

PLC 硬键	脚本文件名基于按键说明。
	
	示例：
	`KEY50.0 = area:=CustomXML, dialog:=SlEECustomDialog, cmdline:="-conf slagmdialog.hmi -mainModule restore.xml -entry cmc2"`
	`KEY51.0 = area:=CustomXML, dialog:=SlEECustomDialog, cmdline:="-conf slagmdialog.hmi -mainModule activate.xml -entry cmc1"`

MMC 指令	脚本文件名基于 NC 指令说明。 **示例：** `MMC("XML, XML_ON, TEST.XML, cmd1","A")`

硬键/软键	**连接点**

操作区预留软键	欲将 OEM 应用程序链接至操作区，目录 siemens\sinumerik\hmi\template\cfg\ (sl<操作区>_oem.xml)下提供相应的模板。必须将属于该操作区的 XML 文件复制到目录 oem\sinumerik\hmi\cfg 下并进行如下编辑：
	必须编程函数 **switchToDialog**，作为软键响应。必须在关键词 **area** 中输入 **CustomXML** 并在 **dialog** 中输入 **SlEECustomDialog** 作为自变量。
	
	语法：
	`<FUNCTION args="-area CustomXML -dialog SlEECustomDialog -mainModule <name of the main module> -entry <menu name>" name="switchToDialog"/>`
	
	示例：
	将 Easy XML 应用链接至诊断区：
	使用“dg.xml”作为脚本文件名称。在该文件中需要调用名称为 **main** 的菜单。
	`<MENU name="DgGlobalHu">`
	`<ETCLEVEL id="0">`
	`<SOFTKEYGROUP name="GroupEtc">`
	`<SOFTKEY position="7">`
	`<PROPERTY name="textID" type="QString">DG_SK</PROPERTY>`
	`<PROPERTY name="translationContext" type="QString">EASY_XML</PROPERTY>`
	`<FUNCTION args="-area CustomXML -dialog SlEECustomDialog -mainModuledg.xml -entry main" name="switchToDialog"/>`
	`</SOFTKEY>`
	`</SOFTKEYGROUP>`
	`</ETCLEVEL>`
	`</MENU>`

Sidescreen	脚本文件名基于“slsidescreen.ini”文件的 Sidescreen 说明。
Easy Extend 软键	标准脚本文件名“agm.xml”

2.3 设计文件

前言

下图显示了控制系统上的制造商目录“System CF-Card/oem/sinumerik/hmi”。

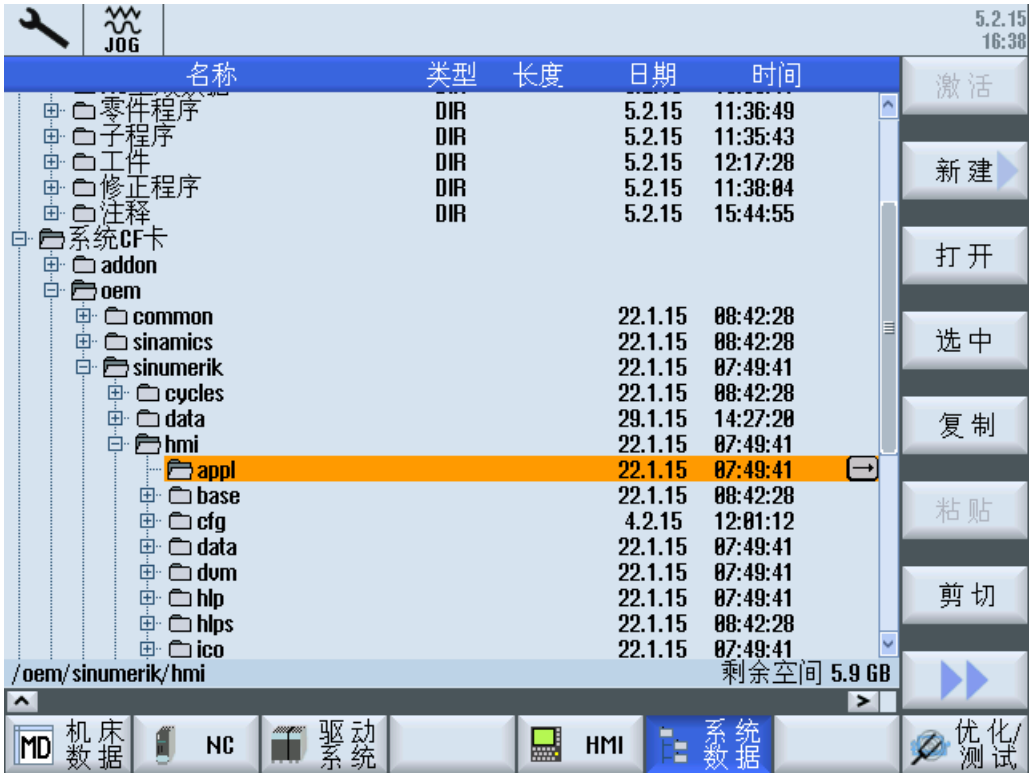


图 2-3 制造商目录

为了进行用户对话框的配置，需要使用控制系统中制造商目录“System CF-Card/oem/sinumerik/hmi”下的以下文件：

表格 2-2　用于配置的文件

文件类型	文件名称	含义	操作区调试 > 系统数据中的存储位置
脚本文件	“xmldial.xml”	该脚本文件通过 XML 标签控制已配置软键菜单的映像以及 SINUMERIK Operate 上操作区 <CUSTOM> 中的对话框屏幕。	制造商目录 > 在用于应用的子目录“appl”中
文本文件	“oem_xml_screens_xxx.ts”	该文本文件包含有用于单个语言菜单和对话框屏幕的文本。	制造商目录 > 在用于语言的子目录“lng”中
位图		控制系统支持 BMP 格式和 PNG 格式。	制造商目录 > 在子目录“ico”中 位图保存在控制系统所属的屏幕分辨率的子目录中。 Bem.：如给定了位图文件的路径，则可以将文件直接保存至该目录中。
XML 文件，在控制文件“xmldial.xml”中插入了 XML 标签 “INCLUDE” 。	例如 “machine_settings.xml”	该文件同样包含用来在 SINUMERIK Operate 上显示对话框窗口和参数的编程指令。	制造商目录 > 在用于应用的子目录“appl”中

文件与用户对话框配置的相关性

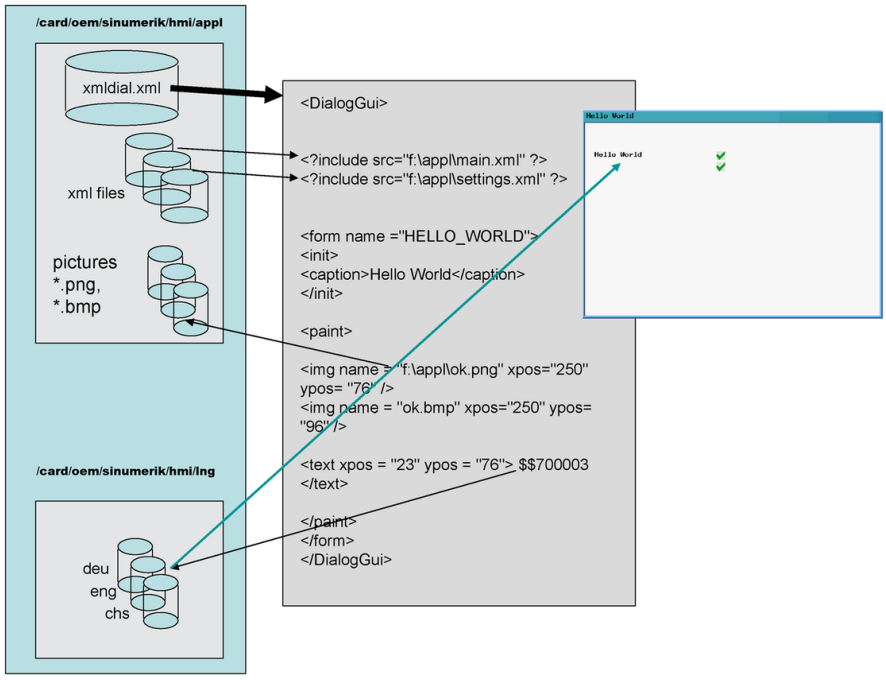


图 2-4　相关性

加载配置

与前面表格“用于配置的文件”的“操作区中的存储位置”一列的说明相同，所创建的文件被复制到制造商目录相应的子目录下。

说明

一旦子目录中有了用于应用的脚本文件“xmldial.xml”，用户就可以在操作区 <CUSTOM> 中启动用户对话框。

第一次复制后必须通过“正常上电启动”对 HMI 进行复位。

SINUMERIK Operate 中的用户对话框示例

在调用操作区 <CUSTOM> 时显示配置好的软键菜单。用户可以通过配置好的各个对话框窗口进行操作。

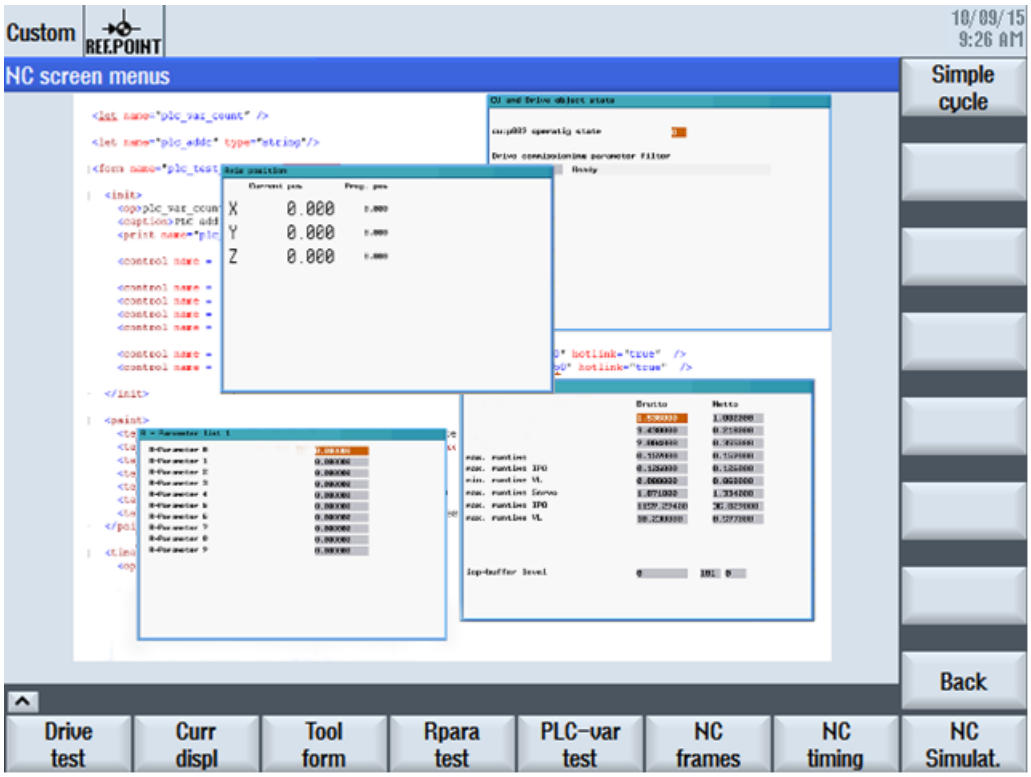


图 2-5　操作区 <CUSTOM> 中用户对话框示例

更多示例参见工具箱

参见

预定义功能 (页 158)

2.4 设计文件的结构

概述

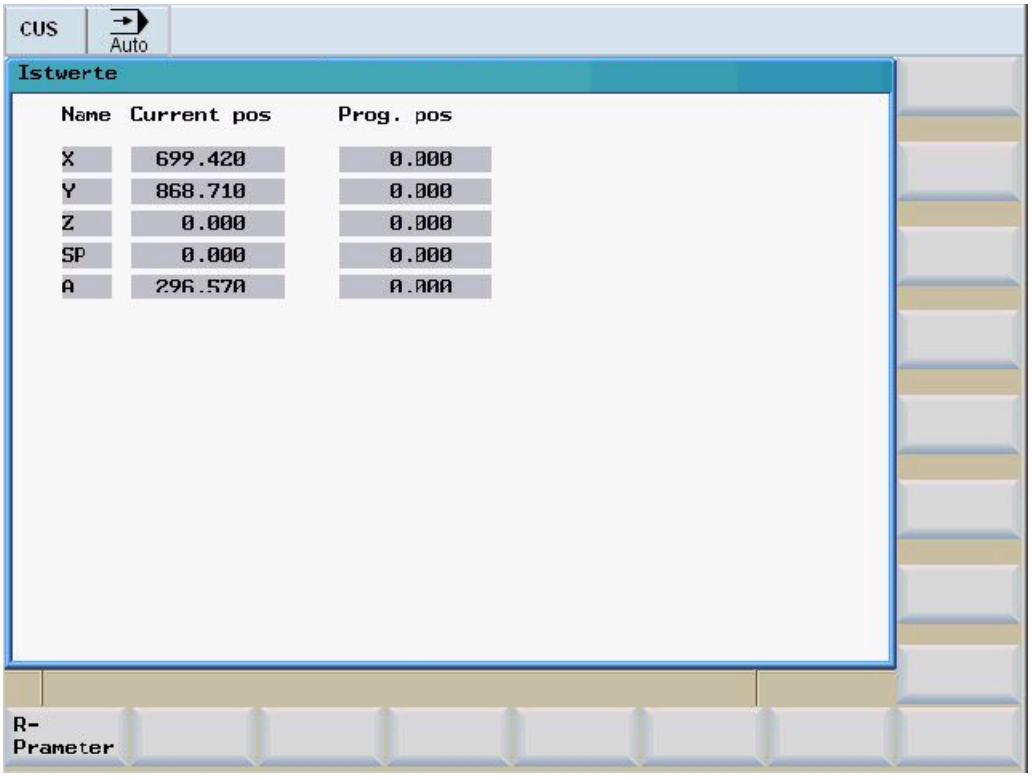
配置文件由以下单元组成：

- 对带有登入软键的登入菜单“main”的说明
- 对话框定义
- 变量定义
- 块说明
- 定义软键条

下列示例说明了文件“xmldial.xml”的 XML 脚本以及相应的截屏。

脚本包含有用于显示实际值和剩余行程的对话框，以及一张 R 参数表。

实际值窗口截图



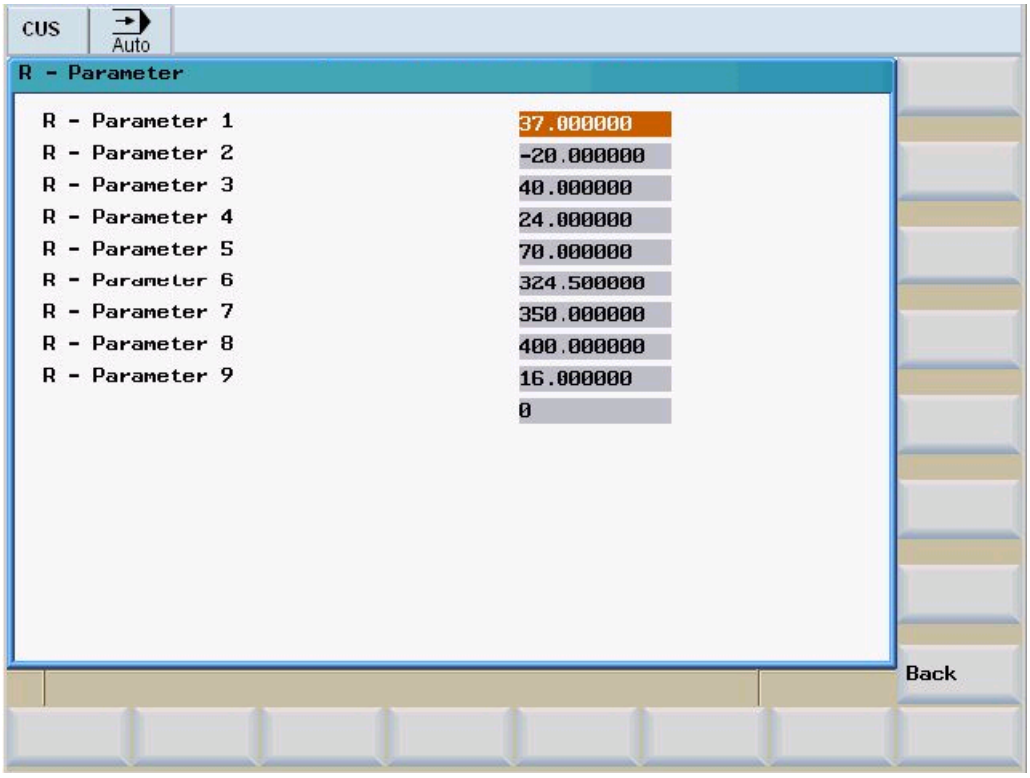
xmldial.xml

```
<DialogGui>

<!--
main menu
It is called by the system software. It starts the application.
The menu tag manages the soft key reactions. One input form can be assigned
to a menu tag.
-->
<menu name = "MAIN">
<OPEN_FORM name = "CURRENT_DISPLAY" />

<softkey POSITION="1">
<caption>R-%nPrameter</caption>
<navigation>MENU_R_PARAMETER</navigation> <!-- opens the menu R parameter --
>
</softkey>


</menu>


<form name = "CURRENT_DISPLAY">

<init>
<caption>Istwerte</caption>

<control name = "label1" xpos = "36" ypos = "56" width = "32"
fieldtype="readonly" refvar="nck/Channel/GeometricAxis/name[0]" />
<control name = "label2" xpos = "36" ypos = "76" width = "32"
fieldtype="readonly" refvar="nck/Channel/GeometricAxis/name[1]" />
<control name = "label3" xpos = "36" ypos = "96" width = "32"
fieldtype="readonly" refvar="nck/Channel/GeometricAxis/name[2]" />
<control name = "label4" xpos = "36" ypos = "116" width = "32"
fieldtype="readonly" refvar="nck/Channel/GeometricAxis/name[3]" />
<control name = "label5" xpos = "36" ypos = "136" width = "32"
fieldtype="readonly" refvar="nck/Channel/GeometricAxis/name[4]" />

<control name = "edit1" xpos = "80" ypos = "56" refvar="nck/Channel/
GeometricAxis/actProgPos[0]" hotlink="true" fieldtype="readonly"
format="%9.3f" time="super fast"/>
<control name = "edit2" xpos = "80" ypos = "76" refvar="nck/Channel/
GeometricAxis/actProgPos[1]" hotlink="true" fieldtype="readonly"
format="%9.3f" time="super fast"/>
<control name = "edit3" xpos = "80" ypos = "96" refvar="nck/Channel/
GeometricAxis/actProgPos[2]" hotlink="true" fieldtype="readonly"
format="%9.3f" time="super fast"/>
<control name = "edit4" xpos = "80" ypos = "116" refvar="nck/Channel/
GeometricAxis/actProgPos[3]" hotlink="true" fieldtype="readonly"
format="%9.3f" time="super fast"/>
```

xmldial.xml

```
<control name = "edit5" xpos = "80" ypos = "136" refvar="nck/Channel/
GeometricAxis/actProgPos[4]" hotlink="true" fieldtype="readonly"
format="%9.3f" time="super fast"/>

<control name = "edit11" xpos = "210" ypos = "56" refvar="nck/Channel/
GeometricAxis/progDistToGo[0]" hotlink="true" fieldtype="readonly"
format="%9.3f" time="super fast"/>
<control name = "edit12" xpos = "210" ypos = "76" refvar="nck/Channel/
GeometricAxis/progDistToGo[1]" hotlink="true" fieldtype="readonly"
format="%9.3f" time="super fast"/>
<control name = "edit13" xpos = "210" ypos = "96" refvar="nck/Channel/
GeometricAxis/progDistToGo[2]" hotlink="true" fieldtype="readonly"
format="%9.3f" time="super fast"/>
<control name = "edit14" xpos = "210" ypos = "116" refvar="nck/Channel/
GeometricAxis/progDistToGo[3]" hotlink="true" fieldtype="readonly"
format="%9.3f" time="super fast"/>
<control name = "edit15" xpos = "210" ypos = "136" refvar="nck/Channel/
GeometricAxis/progDistToGo[4]" hotlink="true" fieldtype="readonly"
format="%9.3f" time="super fast"/>

</init>

<paint>
<text xpos= "36" ypos="30">Name</text>
<text xpos= "80" ypos="30">Current pos</text>
<text xpos= "210" ypos="30">Prog. pos</text>

</paint>
</form>

.....
```

R 参数窗口截图


CUS
Auto
R - Parameter
R - Parameter 1
R - Parameter 2
R - Parameter 3
R - Parameter 4
R - Parameter 5
R - Parameter 6
R - Parameter 7
R - Parameter 8
R - Parameter 9
37.000000
-20.000000
40.000000
24.000000
70.000000
324.500000
350.000000
400.000000
16.000000
0
Back

xmldial.xml

```
.....

<!-- ************************************************************
Menu R- Parameter
-->

<menu name ="MENU_R_PARAMETER">
<OPEN_FORM name = "R-Parameter" />

<softkey POSITION="16">
<caption>Back</caption>
<navigation>MAIN</navigation>
</softkey>

</menu>

<form name = "R-Parameter">

<init>
<DATA_ACCESS type="true" />
<caption>R - Parameter</caption>

<control name = "edit1" xpos = "322" ypos = "34" refvar="nck/Channel/
Parameter/R[1]" />
<control name = "edit2" xpos = "322" ypos = "54" refvar="nck/Channel/
Parameter/R[2]" />
<control name = "edit3" xpos = "322" ypos = "74" refvar="nck/Channel/
Parameter/R[3]" />
<control name = "edit4" xpos = "322" ypos = "94" refvar="nck/Channel/
Parameter/R[4]" />
<control name = "edit5" xpos = "322" ypos = "114" refvar="nck/Channel/
Parameter/R[5]" />
<control name = "edit6" xpos = "322" ypos = "134" refvar="nck/Channel/
Parameter/R[6]" />
<control name = "edit7" xpos = "322" ypos = "154" refvar="nck/Channel/
Parameter/R[7]" />
<control name = "edit8" xpos = "322" ypos = "174" refvar="nck/Channel/
Parameter/R[8]" />
<control name = "edit9" xpos = "322" ypos = "194" refvar="nck/Channel/
Parameter/R[9]" />
<control name = "edit10" xpos = "322" ypos = "214" refvar="nck/Channel/
Parameter/R[10]" />
<control name = "edit10" xpos = "322" ypos = "214" refvar="plc/mb170" />
</init>

<paint>
<text xpos = "23" ypos = "34">R - Parameter 1</text>
<text xpos = "23" ypos = "54">R - Parameter 2</text>
<text xpos = "23" ypos = "74">R - Parameter 3</text>
```

xmldial.xml

```
<text xpos = "23" ypos = "94">R - Parameter 4</text>
<text xpos = "23" ypos = "114">R - Parameter 5</text>
<text xpos = "23" ypos = "134">R - Parameter 6</text>
<text xpos = "23" ypos = "154">R - Parameter 7</text>
<text xpos = "23" ypos = "174">R - Parameter 8</text>
<text xpos = "23" ypos = "194">R - Parameter 9</text>
</paint>
</form>

</DialogGui>
```

2.5 语言相关性

使用和语言相关的文本：

- 软键标记
- 标题
- 辅助文本
- 其它任意文本

与语言相关的文本保存在文本文件中。

说明

使用该文本文件时需要进行下列操作：

- 提供需要使用的语言。
- 传输到控制系统相应的语言目录中。

2.6 XML 诊断

为了诊断和找出脚本错误，系统提供了"Easy XML 诊断"功能。

诊断功能可检查 XML 语法并运行属于项目的所有脚本。为此还会检查功能、菜单、格式和变量的存在/有效性。发现的错误会罗列出来。

"Easy XML 诊断"功能即可使用 **DialogGui** 标签中的属性，也可通过显示机床数据来激活。

激活 Easy XML 诊断

示例：

```
<DialogGui diagnose="true" >
…
…
</DialogGui >
```

或

显示机床数据

MD9113 $MM_EASY_XML_DIAGNOSE		Easy XML 脚本的诊断与更正辅助支持
= 0	诊断无效	
= 1	语法检查生效	

通过跟踪输出诊断

可通过标签 **debug** 将跟踪输出集成至脚本。只有在标签 **DialogGui** 中的 **debug_msg** 属性值为 **on** 时，保存才能成功。

软键功能一览

对话框	软键	功能
"EasyXML 诊断"主菜单	启动脚本	直接启动加载的项目。
	启动检查	启动语法检查。生成的所有信息都可加以查看。可重新进行检查。

对话框	软键	功能
“故障”和“报警”诊断	故障	结果会根据故障和报警进行分类显示。
	报警	
	转到故障	如果发现了故障或报警，则显示该软键。 该软键可从故障列表中打开选中的 XML 文件以便进行编辑。光标被自动置于故障行。
	检查结果	生成的所有信息都可加以查看。
对话框“文档”	保存	保存 XML 文件中的更改。
	取消	XML 文件不更改，然后关闭。检查结果会再次显示。

2.7 XML 标签

2.7.1 通用结构

用于对话框配置的脚本文件结构与指令

所有的对话框配置都保存在 **DialogGui** 标签中。

```
<DialogGui>
…
</DialogGui>
```

示例：

```
<?xml version="1.0" encoding="utf-8"?>
<DialogGui>
…
<FORM name ="Hello_World">
<INIT>
<CAPTION>Hello World</CAPTION>
</INIT>
...
</FORM>

</DialogGui>
```

指令

语言为完成有条件指令与完成程序循环提供有下列指令：

- FOR 循环
- While 循环
- Do-While 循环
- 有条件处理
- Switch 指令和 Case 指令
- 对话框窗口中的操作单元
- 软键说明
- 定义变量

指令的详细说明，请参见"指令/标签说明 (页 38)"一章。

2.7.2 指令/标签说明

为了创建对话框和菜单以及处理程序顺序需要定义下列 **XML 标签**：

说明

引号"<...>"表示属性值，可替换为实际使用的表达式。

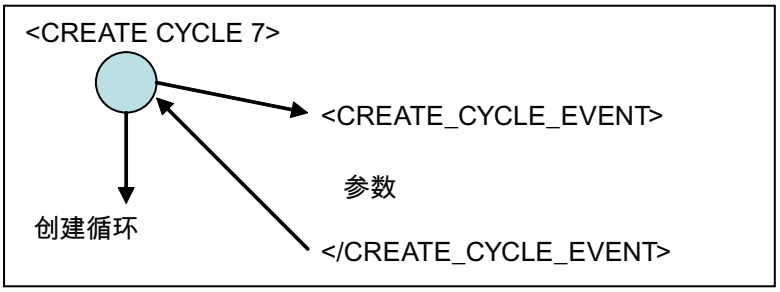
示例：

<DATA_LIST action="read/write/append" id="<list name>">
会进行如下的参数设置：

<DATA_LIST action="read/write/append" id="my datalist">

在复制多行 XML 标签和示例时要注意，不要使引号因不必要的换行而断开。

标签名称	含义
BREAK	有条件中断一个循环。
CONTROL	该标签用来创建控件。详细说明请见章节"创建软键菜单和对话框窗口 (页 74)"。
CONTROL_RESET	该标签可以重新启动一个或多个控制系统组件。 **语法：** `<CONTROL_RESET resetnc="TRUE" />` **属性：** • **RESETNC = "TRUE"** 重新启动 NC 组件 • **RESETDRIVE = "TRUE"** 重新启动驱动组件。

标签名称	含义
CREATE_CYCLE_EVENT	解析器处理标签 CREATE_CYCLE 时，会首先将信息<CREATE_CYCLE_EVENT>发送到当前 FORM 标签中。在解析器根据参数表和创建规则生成 NC 指令之前，该信息会被用于循环参数的创建准备。  **语法：** <CREATE_CYCLE_EVENT> … … </CREATE_CYCLE_EVENT> **示例：** <SOFTKEY_OK> … <CREATE_CYCLE /> … … <SOFTKEY_OK> … … <FORM> <NC_INSTRUCTION>MY_CYCLE($P1, $P2)</ NC_INSTRUCTION > … … <CREATE_CYCLE_EVENT> <type_cast name="P1" type="int"/> <OP> P1 = P1 * 150 </OP> </CREATE_CYCLE_EVENT> … … </FORM>

标签名称	含义
DATA	该标签用于写入 NC、PLC、GUD 和驱动数据。 地址构成可以查阅章节 “组件地址分配 (页 144)”。 **属性：** • **name** 变量地址 **标签值：** 标签值可以为常量、**变量或 term**。 **语法：** <DATA name="<variable name>"> value </DATA> <DATA name="<variable name>"> <term> </DATA> **示例：** <DATA name = "plc/mb170"> 1 </DATA> … <LET name = "tempVar"> 7 </LET> <!– 局部变量"tempVar”的内容会写入到存储器字节 170 中 → <DATA name = "plc/mb170">$tempVar</DATA>
DATA 续	**示例：** Term 的计算。计算结果会分配给指定的变量。 <let name="a" >#f</let> <let name="b" >7</let> <data name = "b"> a & b</data> <let name="c" type="string"></let> <data name = "c"> _T" test" + _T" and test"</data>

标签名称	含义

DATA_LIST	该标签可以保存或者恢复所列出的驱动数据和机床数据。
	按行方式列出地址。地址构成可以查阅章节 “组件地址分配 (页 144)”。
	可以创建最多 20 个临时数据列表。
	
	属性：
	• **action**
	read - 将列出变量的值存储到一个临时存储器中
	append - 将列出变量的值添加到已有列表中
	write - 将保存的值复制到相应的机床数据中
	• **id**
	标签用于识别临时存储器
	语法：
	`<DATA_LIST action="<read/write/append>"  id="<list name>">`
	`NC/PLC 地址汇编`
	`</DATA_LIST>`
	
	示例：
	`<DATA_LIST action ="read" id="<name>">`
	`nck/channel/parameter/r[2]`
	`nck/channel/parameter/r[3]`
	`nck/channel/parameter/r[4]`
	`$MN_USER_DATA_INT[0]`
	`…`
	`</ DATA_LIST>`
	`<DATA_LIST action ="write" id="<name>" />`

DIALOGGUI	该标签显示了 Easy XML 的 Root 标签。可通过 Easy XML 解析器再编辑所有已完成的指令。 **属性：** 可以选择性输入提及的属性，使能集体函数： • **diagnose** - 值为 **true** 表示激活 XML 诊断 • **debug_msg** - 值为 “on”表示保存跟踪记录 **更多信息**参见章节 “XML 诊断 (页 35)” • **textfile** - 待使用的文本文件名称 • **textcontext** - 只有与属性文本文件一同使用时待使用的文本才有效 • **textfilelist** - 可以加载含不同文本文件的列表 **更多信息**参见章节 “特定项目的文本文件 (页 283)”

标签名称	含义
DIALOGGUI 续	• **restoresession** - **真值** 如果再次选择 Easy XML 项目，解析器会打开最后选择的菜单以及最后选择的窗体。该属性会持续至重新启动 HMI。 **更多信息**参见章节“恢复会话 (页 291)”
DEBUG_MSG	此属性控制跟踪输出数据的存储。描述参见 DEBUG 标签下的章节“创建软键菜单和对话框窗口 (页 74)”。 若值为**打开**，则 Parser 将所有输出保存在一个文件中。这样有可能导致脚本执行变慢。缺省设置为**关闭**。
DO_WHILE	Do-While 循环 DO 指令 WHILE （测试） **语法：** <DO_WHILE> 指令 … <CONDITION>…</CONDITION> </DO_WHILE> Do-While 循环由一个指令块和一个条件构成。首先执行指令块中的代码，然后计算条件。如果条件为真（true），则重新执行代码部分。一直重复，直至条件为假（false）。 **示例：** <DO_WHILE> <DATA name = "PLC/qb11"> 15 </DATA> <CONDITIION> "plc/ib9" == 0 </CONDITION> </DO_WHILE>

标签名称	含义
DYNAMIC_INCLUDE	标签包含 XML 脚本文件。 与标签 INCLUDE 相反，在执行相应指令时才进行读取。 在大型项目中该标签的使用能缩短用户使用或循环支持的载入时间。此外，对系统资源的平均需求也会下降，因为在会话过程中不需要一直调用所有的对话框。 **语法：** <DYNAMIC_INCLUDE src="path name"/> **示例：** <SOFTKEY POSITION="3"> <CAPTION>MY_MENU</CAPTION> <DYNAMIC_INCLUDE src/> <NAVIGATION>MY_MENU</NAVIGATION> </SOFTKEY>
ELSE	满足条件时的指令（**IF（如果）**、**THEN（则）**、**ELSE（否则）**）

标签名称	含义
FOR	FOR 循环 for （初始化；测试；继续）指令 **语法：** <FOR> <INIT>…</INIT> <CONDITION>…</CONDITION> <INCREMENT>…</INCREMENT> 指令 … </FOR> 按下列方式执行 For 循环： 1. 运用表达式 **初始化** （INIT）。 2. 运用表达式 **文本** （CONDITION） 作为布尔型表达式。 如果值为假（false），则结束 For 循环。 3. 执行后续的指令。 4. 运用表达式 **继续** （INCREMENT）。 5. 继续使用 2。 所有在 INIT、CONDITION 和 INCREMENT 分支中使用的变量都要在 For 循环的外部创建。 **示例：** <LET name = "count">0</LET> <FOR> <INIT> <OP> count = 0</OP> </INIT> <CONDITION> count <= 7 </CONDITION> <INCREMENT> <OP> count = count + 1 </OP> </INCREMENT> <OP> "plc/qb10" = 1+ count </OP> </FOR>
FORM	该标签包含对用户对话框的说明。详细说明请见章节“创建软键菜单和对话框窗口 (页 74)”。
HMI_RESET	该标签会触发 HMI 重新启动。 编译在该指令后会被中断。

标签名称	含义
IF	有条件的指令（**IF, THEN, ELSE**） 标签 THEN 和 ELSE 包含在标签 IF 中。 标签 IF 后跟有条件，而条件在标签 CONDITION 中执行。需要通过其他的指令处理来确定运算结果。如果函数结果为真，则执行 THEN 分支而跳过 ELSE 分支。如果函数结果为假，则解析器执行 ELSE 跳转。 **语法：** <IF> <CONDITION> 条件 != 7 </CONDITION> <THEN> 情况说明：条件满足 </THEN> <ELSE> 情况说明：条件不满足 </ELSE> </IF> **示例：** <IF> <CONDITION> "plc/mb170" != 7 </CONDITION> <THEN> <OP> "plc/mb170" = 7 </OP> … </THEN> <ELSE> <OP> "plc/mb170" = 2 </OP> … </ELSE> </IF>

标签名称	含义

IF 续	如果条件中仅使用了局部变量，则可将该条件作为属性输入 IF 标签。
	
	示例：
	`<let name="var1">1</let>`
	`<if condition="var == 1">`
	`<then>`
	`...`
	`...`
	`</then>`
	`</if>`
	
	提示：
	如果条件中使用了 Control 变量，则需为 Control 变量分配 Integer 数据类型。
	特例：
	imagebox 和 **multiline** 类型的 Control 变量分配的数据类型为 **string**。

INCLUDE	指令包含有 XML 说明。 （另见该表格中的 DYNAMIC_INCLUDE ） **属性：** • **src** 包含路径名。 **语法：** `<?INCLUDE src="<Path name>" ?>`

标签名称	含义
LET	指令以给出的名称创建一个局部变量。 **数组：** 通过属性 **dim** （维数）可创建一维或二维的数组。各个数组元素的寻址通过数组下标进行。 对于二维数组，先指定行下标，再指定列下标。 • 一维数组： 下标 0 至 4 0 1 2 3 4 • 二维数组： 行下标 0 至 3，列下标 0 至 5 0,0 0,1 0,2 0,3 0,4 0,5 1,0 1,1 1,2 1,3 1,4 1,5 2,0 2,1 2,2 2,3 2,4 2,5 3,0 3,1 3,2 3,3 3,4 3,5 **属性：** • **name** 变量名称 • **type** 变量类型可以为整数型（INT）、无符号整数型（UINT）、双精度型（DOUBLE）、浮点型（FLOAT）、字符串型（STRING）或 STRUCT。此外，可以将通过 **typedef** 创建的结构用作变量类型（参见标签名称 TYPEDEF）。如果未给出类型指令，则系统会创建一个整数型变量。 `<LET name = "VAR1" type = "INT" />` • **permanent** 如果属性为 **true**，则永久保存变量值。该属性仅对全局变量有效。 • **poweroff** 如果属性的值为 **true**，则数值会一直保留至控制系统关闭。 **更多信息**参见章节 “恢复会话 (页 291)” • **dim** 用来指定数组元素的数量。对于二维数组，用逗号分隔第一维和第二维。对数组元素的存取通过数组下标进行，写在变量名称后的方括号内。 name[index] 或 name[row,column] – 一维数组：dim="<元素数量>"

标签名称	含义
	– 二维数组：dim="<行数>,<列数>" 未赋初值的数组元素用“0”占用。

标签名称	含义

LET 续	**示例：**
	一维数组：
	`<let name="array" dim="10"></let>`
	二维数组：
	`<let name="list_string" dim="10,3" type="string"></let>`
	
	预设值：
	可以对变量进行初始化赋值。
	`<LET name = "VAR1" type = "INT"> 10 </LET>`
	
	如果将 NC 或 PLC 变量的值保存在一个局部变量中，则赋值运算会自动将格式与所读入变量的格式进行匹配。
	• 字符串变量的预设值：
	只要将格式化的文本作为值进行传送，就可以为一个字符串变量赋值多行文本。如果一行要用 line feed <LF>（换行）结尾，应在行尾添加字符 "\\n"。
	`<LET name = "text" type = "string"> F4000 G94\\n`
	`G1 X20\\n`
	`Z50\\n`
	`M2\\n`
	`</LET>>`
	
	数组 (Arrays)：
	`<let name="list" dim="10,3">`
	`  {1,2,3},`
	`  {1,20}`
	`</let>`
	
	`<let name="list_string" dim="10,3" type="string">`
	`  {"text 10","text 11"},`
	`  {"text 20","text 21"}`
	`</let>`
	
	赋值：
	可以使用赋值运算"="将机床数据或子程序中的值赋值给一个变量。
	变量的有效性会一直延伸到最上一级的 XML 数据块。
	需要全局使用的变量应直接在标签 **DialogGUi** 后创建。
	对于对话框须注意以下事项：
	• 处理信息会打开相应的标签。
	• 在信息执行后会关闭标签。
	• 标签中的所有变量会随着关闭而被清零。

标签名称	含义
LET 续	**变量类型 struct：** 该变量类型包含在使用结构名称时所能响应的变量汇总。一个结构能包含所有变量类型以及结构。 结构中的变量通过标签“element”进行说明。标签的属性和初始化与 let 说明的属性和初始化相符。 <let name=“name” type=“struct”> <element name=“<名称>” type=“<变量类型>” /> … … … </let>

标签名称	含义
LET 续	通过结构和变量名称访问结构变量。两个名称由点隔开。 <op> 结构_名称.变量_名称 = 值; </op> **示例：** <let name="info" type="struct"> <element name="id" type="int" /> <element name="name" type="string" /> <element name="phone" type="string" /> </let> <op> info.id = 1; info.name = _T"my name"; info.phone = _T"0034 45634"; </op> **结构初始化：** 可以通过在结构元素中输入初始值，在创建变量时对结构进行初始化。在结构数组中，每个结构间使用尖括号隔开。

标签名称	含义
LOCK_OPERATING_AREA	禁止操作区域切换。 使用该标签 **UNLOCK_OPERATING_AREA** 可取消禁止操作区域切换。 **语法：** `<LOCK_OPERATING_AREA />`
MSG	操作组件显示标签中所给出的信息。 如果使用报警编号，则对话框会显示为该编号所保存的文本。 **示例：** `<MSG text ="my message" />`

MSGBOX	指令会打开一个信息盒，其给分支的返回值可供使用。
	
	语法：
	`<MSGBOX text="<Message>" caption="<caption>" retvalue="<variable name>" type="<button type>" />`
	
	属性：
	• **text**
	文本
	• **caption**
	标题
	• **retvalue**
	其中复制有返回值的变量名：
	1 – OK
	0 – CANCEL
	• **type**
	应答可能：
	"BTN_OK"
	"BTN_CANCEL"
	"BTN_OKCANCEL"
	如果对属性 **"text"** 或 **"caption"** 使用报警编号，则信息盒会显示为该编号所保存的文本。
	
	示例：
	`<MSGBOX text="测试消息" caption="信息" retvalue="result" type="BTN_OK" />`

标签名称	含义

OP	该标签会执行给定的操作。
	可以执行章节“运算符 (页 72)”中所列出的各种运算。
	对于 NC/PLC 上的存取和驱动数据要在 **引用字符** 中设置完整的变量名称。地址构成可以查阅章节 “组件地址分配” (页 144) 。
	`PLC: "PLC/MB170"`
	`NC: "NC/Channel/…"`
	
	示例：
	`<LET name = "tmpVar" type="INT"> </LET>`
	`<OP> tmpVar = "plc/mb170" </OP>`
	`<OP> tmpVar = tmpVar *2 </OP>`
	`<OP> "plc/mb170" = tmpVar </OP>`
	
	一个运算标签内可以使用多个等式。用分号标记指令结束。
	
	示例：
	`<op>`
	`x = x+1;`
	`y = y+1;`
	`</op>`
	
	字符串处理：
	运算指令可以对字符串进行处理并将结果字符串赋值给等式中所给出的字符串变量。
	为了标记文本表达式需要在文本前放置标志 _T 。此外还能对变量值进行格式化。格式规定以标志 _F 开始，后面跟有格式指令。接着给定变量地址。
	
	示例：
	`<LET name="buffer" type="string"></LET>`
	`…`
	`…`
	`…`
	`<OP> buffer = _T"unformatted value R0= " + "nck/Channel/Parameter/R[0]" + _T" and " + _T"$$85051" + _T" formatted value R1 " + _F %9.3f"nck/Channel/Parameter/R[1]" </OP>`

标签名称	含义

OPERATION	**Operation**
	一个等式内可以使用一个移位操作。
	向左移动运算符 "<<"
	通过函数 << 向左移动位。可直接或通过一个变量确定待移动值和移动数。如果达到数据格式极限，位会超出极限移动，此时，系统不显示错误信息。
	运算法则
	从左往右算且 '+' 和 '-' 位于 '<<' 或 '>>' 前
	
	示例：
	`<op> idx = idx << 2 </op>`
	`<op> idx = 3 + idx << 2 </op>`
	
	向右移动运算符 ">>"
	通过函数 >> 向右移动位。可直接或通过一个变量确定待移动值和移动数。如果达到数据格式极限，位会超出极限移动，此时，系统不显示错误信息。
	运算法则
	从左往右算且 '+' 和 '-' 位于 '<<' 或 '>>' 前
	
	示例：
	`<op> idx = idx >> 2 </op>`
	`<op> idx = 3+idx >> 2</op>`

标签名称	含义

PASSWORD	在“EasyExtend”功能中，该标签用于启用附加设备。
	此处输入的字符串会写入指定的参考变量中，在 PLC 用户程序中处理。
	
	语法：
	`<PASSWORD refVar ="<variable name>" />`
	
	属性：
	• **refVar**
	参考变量的名称
	• **text**
	属性说明替换标准文本（可选）
	
	示例：
	`<PASSWORD refvar="plc/mw107" />`
	
	可选示例：
	`<password refVar = "plc/MD108" text="Password" />`

POWER_OFF	显示信息要求操作者关闭机床。显示文本固定保存在系统中。

标签名称	含义

PRINT	标签会在对话行中输出文本或者将文本复制到给定的变量中。
	如果文本包含有格式化标志，则会将变量值插入到相应的位置上。
	
	语法：
	`<PRINT name="变量名称" text="text %格式化"> Variable, … </PRINT>`
	`<PRINT text="text %格式化"> Variable, … </PRINT>`
	属性：
	• **name**
	保存文本的变量的名称（可选）
	• **text**
	文本
	
	格式化：
	字符"%"会对作为值给定的变量进行格式化。
	%[标记] [宽度] [.小数点后面的位置] 类型
	• 标记：
	用于确定输出格式的可选字符：
	– 右对齐或左对齐（"-" 用于左对齐）
	– 前面加零（"0"）
	– 使用空格符填充
	• 宽度：
	确定一个非负数最小输出宽度的依据。如果待输出值占用的位置少于依据所确定的位置，则使用空格符填充空缺的位置。
	• 小数点后面的位置：
	使用浮点数时优化参数用来确定小数点后面的位置数。
	• 类型：
	类型字符用来确定打印指令要输出哪些数据格式。该字符必须给定。
	– d:整数值
	– f：浮点数
	– s:字符串

标签名称	含义

PRINT 续	**值：**
	其值应当插入到文本中的变量数量。
	变量类型必须与格式规定的相应类型标识一致并用逗号加以间隔。
	
	示例：
	在信息行中输出文本
	`<PRINT text="信息文本" />`
	使用变量格式输出文本
	`<LET name="trun_dir"></LET>`
	`<PRINT text="M%d">trun_dir</PRINT>`
	在字符串变量中使用变量格式输出文本
	`<LET name="trun_dir"></LET>`
	`<LET name="str" type="string" ></LET>`
	`<print name="str" text="M%d ">trun_dir</print>`

PROGRESS_BAR	标签打开或关闭一根进度条。进度条显示在应用窗口的下方。
	
	语法：
	`<PROGRESS_BAR type="<true/false>"> value </ PROGRESS_BAR>`
	
	属性：
	• **type =** “TRUE” - 打开进度条
	• **type** = “FALSE” - 关闭进度条
	• **min**
	（可选）- 最小值
	• **max**
	（可选）- 最大值
	
	值：
	• **Value**
	进度条的百分比位置
	示例：
	`<PROGRESS_BAR type="true" min="0" max="101" >20< /`
	`PROGRESS_BAR>......<PROGRESS_BAR >50< /`
	`PROGRESS_BAR>......<PROGRESS_BAR type="false" >100< /PROGRESS_BAR>`

标签名称	含义

SEND_MESSAGE	该标签会向有效的 FORM 标签发送含两个参数的信息，该信息在标签“Message”中进行处理。
	
	语法：
	`<SEND_MESSAGE>p1, p2</SEND_MESSAGE>`
	
	示例：
	`<SOFTKEY POSITION="3">`
	`  <caption>Set%nParameter</caption>`
	`  <send_message>1, 0</send_message>`
	`</SOFTKEY>`
	
	`<FORM>`
	…
	…
	`<MESSAGE>`
	`<SWITCH>`
	`<CONDITION>$message_par1</CONDITION>`
	`<CASE value="1">`
	…
	…
	`</CASE>`
	`</SWITCH>`
	`</MESSAGE>`
	…
	…
	
	`</FORM>`

标签名称	含义

SHOW_CONTROL	借助该标签可以指定是否显示控件。
	
	语法：
	`<SHOW_CONTROL name="<name>" type="<type>" />`
	
	属性：
	• **name**
	控件的名称
	• **type =** “TRUE” - 控件可见
	• **type**= “FALSE” - 控件不可见（隐藏）
	
	示例：
	`<SHOW_CONTROL name="myEditfield" type="false" />`
	…
	…
	`<SHOW_CONTROL name="myEditfield" type="true" />`

SLEEP	该标签会在指定的时间区间内中断脚本的执行。中断时间为返回值乘以 50ms 的基准时间。
	
	语法：
	`<SLEEP value="中断时间" />`
	
	示例：
	等待时间 1.5 s。
	`<SLEEP value="30" />`

STOP	解释在该位置中断。

标签名称	含义
SWITCH	指令 **SWITCH** 表示多项选择。表达式运用一次并与常数进行比较。如果表达式与常数一致，则执行 **CASE** 指令中的指令。 如果常数与表达式不一致，则执行 **DEFAULT** 指令。 **语法：** <SWITCH> <CONDITION> 值 </CONDITION> <CASE value="<常量 1>"> 指令 … </CASE> <CASE value="<常量 2>"> 指令 … </CASE> <DEFAULT> 指令 … </DEFAULT> </SWITCH>

标签名称	含义

SWITCHTOAREA	SWITCHTOAREA 标签从用户操作区切换至指定的操作区。
	参数指定为属性值。
	
	语法：
	`<switchToArea name="area" args="argument "/>`
	
	属性：
	name
	以下操作区名称与操作区一致：
	• **AreaMachine** - 机床
	• **AreaParameter** - 参数
	• **AreaProgramEdit** - 编辑器
	• **AreaProgramManager** - 程序管理器
	• **AreaDiagnosis** - 诊断
	• **AreaStartup** - 调试
	args（预留）
	激活与对话框相应的运行时间自变量时派定一个操作区。
	
	示例：
	`<switchToArea name="AreaMachine" />`

SWITCHTODYNAMICTARGET	如果将之前的对话框定义为动态跳转目标，则 SWITCHTODYNAMICTARGET 标签会激活该对话框并结束脚本编辑。 **语法：** `<switchToDynamicTarget />`
THEN	当条件满足时应执行的指令（**IF, THEN, ELSE**）

标签名称	含义
TYPE_CAST	该标签用于控制局部变量的数据类型转换。 **语法：** <type_cast name="variable name" type=" new type" /> **属性：** • **name** 变量名称 • **type** 给变量赋值新的数据类型。 • **convert** 给变量赋值新的数据类型。此外，变量值转换为新的数据类型。

标签名称	含义

TYPEDEF	通过标签可为数据类型确定一个新的名称。这对于结构定义的优势在于只要定义一次数据类型，然后便能在 LET 指令中作为数据类型使用。
	作为属性需要具有名称和类型。
	解析器只支持轮廓定义的确定。
	
	类型定义中的变量通过标签“element”进行声明。标签的属性与 let 命令的属性一致。
	
	`<typedef name="<名称>" type="struct">`
	`  <element name="<名称>" type="<变量类型>" />`
	…
	…
	…
	`</typedef>`
	
	定义之后，名称可作为 LET 命令的数据类型使用。
	
	`<let name="<变量名称>" type="<名称>"></let>`
	
	示例:
	`<typedef name="my_struct" type="struct">`
	`  <element name="id" type="int" />`
	`  <element name="name" type="string" />`
	`  <element name="phone" type="string" />`
	`</typedef>`
	
	`<let name="info" type="my_struct"></let>`
	…
	…
	`<op>`
	`  info.id = 1;`
	`  info.name = _T"my name";`
	`  info.phone= _T"0034 45634";`
	`</op>`

标签名称	含义
TYPEDEF 续	作为调用参数，某些预定义的函数需要使用“结构型”的变量 RECT、POINT 或 SIZE。该结构仅在文件 struct_def.xml 中定义。 **更多信息**参见章节“创建 struct_def.xml 文件 (页 142)” **RECT:** `<typedef name="StructRect" type="struct" > <element name="left" type="int">0</element> <element name="top" type="int">0</element> <element name="right" type="int">0</element> <element name="bottom" type="int">0</element> </typedef>` **POINT:** `<typedef name="StructPoint" type="struct" > <element name="x" type="int">0</element> <element name="y" type="int">0</element> </typedef>` **SIZE:** `<typedef name="StructSize" type="struct" > <element name="width" type="int">0</element> <element name="height" type="int">0</element> </typedef>`
UNLOCK_OPERATING_AREA	取消禁止切换操作区域 **语法：** `<UNLOCK_OPERATING_AREA />`

标签名称	含义

WAITING	标签在 NC 或者驱动复位后等待组件重新启动。
	
	属性：
	• **WAITINGFORNC** = "TRUE" - 等待 NC 重启
	• **WAITINGFORDRIVE** = "TRUE" - 等待驱动重启
	
	语法：
	`<WAITING WAITINGFORNC = "TRUE"/>`
	
	示例：
	…
	`<CONTROL_RESET resetnc = "true" resetdrive = "true"/>`
	`<WAITING waitingfornc = "true" waitingfordrive = "true" />`
	…

WHILE	While 循环
	`WHILE (Test)`
	`  指令`
	
	语法：
	`<WHILE>`
	`<CONDITION>…</CONDITION>`
	指令
	…
	`</WHILE>`
	While 循环用于按顺序多次执行指令，只要满足条件。在处理指令序列前对条件进行检查。
	
	示例：
	`<WHILE>`
	`<CONDITION> "plc/ib9" == 0 </CONDITION>`
	`<DATA name = "PLC/qb11"> 15 </DATA>`
	`</WHILE>`

标签名称	含义
XML_PARSER	标签 “XML_PARSER” 可用于解析 XML 文件。 解析器编译 XML 文件并调用定义的回调函数。每个回调函数都属于一个预定义事件。编程人员可在该函数内部处理 XML 文件。 **预定义事件：** • start document 解析器打开文档并开始解析。 • end document 解析器关闭文档。 • start element 解析器找到了一个元素并创建了一个含所有属性和属性值的列表。 这些列表被转送给回调函数。 • end element 找到元素末尾。 • characters 解析器转送元素的所有字符。 • error 解析器发现了一个语法错误。 出现一个事件时，解析器会调用回调函数并检查函数的返回值。函数返回值“true”时，解析器会继续进程。

标签名称	含义

XML_PARSER 续	**接口**
	属性值 **name** 包含 XML 文件的路径。
	为给回调函数分配事件，须确定以下属性：
	标准
	startElementHandler
	endElementHandler
	charactersHandler
	可选
	errorHandler
	documentHandler
	属性值用于定义回调函数的名称。
	
	示例：
	`<XML_PARSER name="f:\appl\xml_test.xml">`
	`<!-- standard handler -->`
	`<property startElementHandler="startElementHandler" />`
	`<property endElementHandler="endElementHandler" />`
	`<property charactersHandler="charactersHandler" />`
	`<!-- optional handler -->`
	`<property errorHandler="errorHandler" />`
	`<property documentHandler="documentHandler" />`
	`</XML_PARSER>`

标签名称	含义

XML_PARSER 续	此外，解析器还会提供可使回调函数访问事件数据的变量。
	
	startElementHandler:
	函数参数
	tag_name - 标签名称
	num - 找到的属性数
	系统变量
	$xmlAttribute 具有一定数目的通过 num 给定的单元的字符串数组。
	$xmlValue 属性值范围为 0-num 的字符串数组。
	
	示例：
	`<function_body name="startElementHandler" return="true" parameter="tag_name, num">`
	`  <print text="attribute name %s"> $xmlAttribute[0]</print>`
	`  <print text="attribute value %s"> $xmlValue[0]</print>`
	`</function_body>`
	
	endElementHandler:
	函数参数
	tag_name - 标签名称
	
	示例：
	`<function_body name="endElementHandler" parameter="tag_name">`
	`  <print text="name %s"> tag_name </print>`
	`</function_body>`

标签名称	含义

XML_PARSER 续	**charactersHandler:**
	系统变量
	$xmlCharacters 含数据的字符串
	$xmlCharactersStart 始终为 0
	$xmlCharactersLength 字节数
	
	示例：
	`<function_body name="charactersHandler" return="true" >`
	`  <print text="chars %s"> $xmlCharacters </print>`
	`</function_body>`
	
	documentHandler:
	函数参数
	state 1 start document，2 end document
	
	errorHandler:
	系统变量
	$xmlErrorString 包含无效行（系统变量）
	
	示例：
	`<function_body name="errorHandler" >`
	`  <print text="error %s">$xmlErrorString</print>`
	`</function_body>`
	
	回调结果：
	$return 如果为 1 (true)，解析器会继续解析文件

2.7.3 颜色代码

颜色属性使用 HTML 语言的颜色代码搭配。

颜色数据通过句法由字符"#"（菱形）和六个十六进制系统数字共同组成，而每种颜色通过两个数字代表。

R – 红色

G – 绿色

B – 蓝色

#RRGGBB

示例：

color= "#ff0011"

示例颜色：

红色	绿色	蓝色	黄色	白色	黑色
#FF0000	#00FF00	#0000FF	#ffff00	#FFFFFF	#000000

2.7.4 专用 XML 句法

如果要在通用 XML 编辑器中正确显示在 XML 句法中有特殊含义的字符，则必须进行写法转换。

关系到下列字符：

字符	XML 中的表示法
<	<
>	>
&	&
"	"
'	'

2.7.5 XML 特殊字符

为了显示字符或使用控制符，提供了对 HTML 特殊字符的识别。

可以使用代码页"Latin 1"的十进制和十六进制编码。

示例

字符	说明	十进制符号	十六进制符号
"	上引号	"	"
LF	换行（Line Feed）

诊断

```
<MENU name="DgGlobalHu">
 <ETCLEVEL id="0">
  <SOFTKEYGROUP name="GroupEtc">
   <SOFTKEY position="7">
    <PROPERTY name="textID" type="QString">DG_SK</PROPERTY>
    <PROPERTY name="translationContext" type="QString">EASY_XML</
PROPERTY>
    <FUNCTION args="-area CustomXML -dialog SlEECustomDialog -
mainModule<space ?>dg.xml -entry main" name="switchToDialog"/>
   </SOFTKEY>
  </SOFTKEYGROUP>
 </ETCLEVEL>
</MENU>
```

<space ?> - 此处空格

2.7.6 运算符

运算指令可以处理下列运算：

运算符	含义
=	赋值
==	等于
<, <	小于
>, >	大于
<=, <=	小于等于
>=, >=	大于等于
|	位模式或（OR）连接
||	逻辑或（OR）连接
&, &	位模式与（AND）连接
&&, &&	逻辑与（AND）连接
+	加法
-	减法
*	乘法
/	除法
!	否
!=	不等

运算由左向右进行。 括号可能对于表达式也有意义，可以确定部分表达式的处理优先级。

2.7.7 系统变量

系统变量是在各个脚本中使用的变量，用于解析器和脚本执行过程之间的数据交换。

下表为自动生成的变量与标签对应关系一览：

变量名称	含义	适用标签
$actionresult	解析器报告，是否要通过解析器处理事件	KEY_EVENT
$focus_name	包含当前所要输入的数组的名称	FOCUS_IN INDEX_CHANGED EDIT_CHANGED
$focus_item_data	包含已分配给数组的数字值 item_data	
$gestureinfo	针对手势操作：解析器会在变量 $gestureinfo 中提供手势信息，并执行标签 gesture_event。	GESTURE_EVENT
$return	变量用于传输间接函数的返回值	FUNCTION_BODY
$message_par1 $message_par2	包含函数 SEND_MESSAGE 的调用参数	MESSAGE
$xmlAttribute	包含已找到的标签属性的列表	XML_PARSER startElementHandler
$xmlValue	包含已找到的标签值的列表	
$xmlCharacters	包含数据流	XML_PARSER charactersHandler
$xmlCharactersStart	包含起始下标	
$xmlCharactersLength	包含数据流中保存的字符的数量	
$mouse_event.type $mouse_event.x $mouse_event.y $mouse_event.id $mouse_event.buttons $mouse_event.button	鼠标事件参数的传输结构	MOUSE_EVENT

2.7.8 创建软键菜单和对话框窗口

欲链接对话框窗口，在 XML 说明中应含有名称为"main"的菜单标签。在激活操作区 <CUSTOM> 后由系统调用该标签。在标签之中可以定义其他的对话框窗口以及激活一个对话框。

```
<menu name= "MAIN">
<OPEN_FORM name= "main dialogue">
<softkey POSITION="1">
<caption>sub menu 1</caption>
<navigation>sub menu 1</navigation>
</softkey>
<softkey POSITION="8">
<caption>sub menu 8</caption>
<navigation>sub menu 8</navigation>
</softkey>
</menu>

 <menu name= "sub menu 1">
 <OPEN_FORM name= "dialogue 1">
</menu>

<menu name= "sub menu 8">
<OPEN_FORM name= "dialogue 8">
</menu>
```

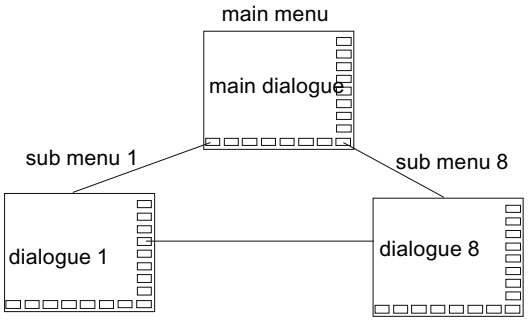


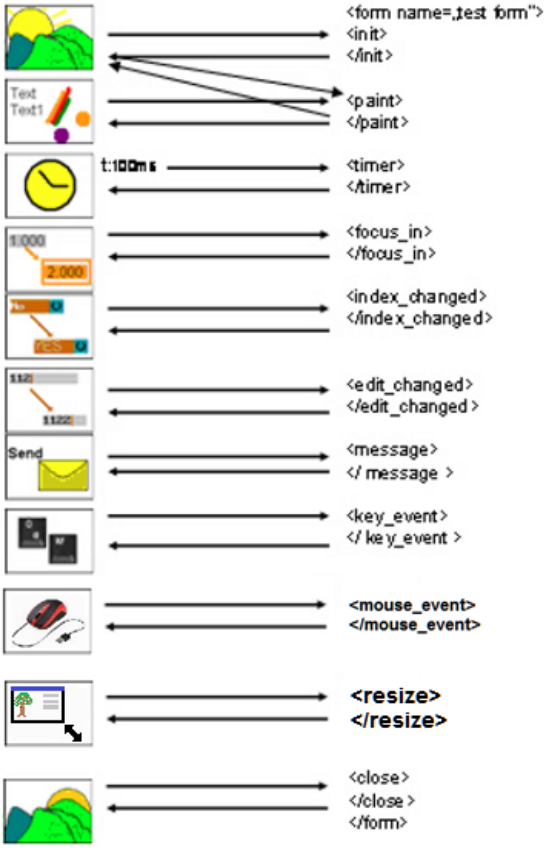
图 2-6　菜单结构

标签名称	含义
FORM	该标签包含对用户对话框的说明。其他相关的标签见章节“创建菜单与对话框窗口”中的说明。 **语法：** `<FORM name="<dialog name>" color="#ff0000">` **属性：** • **color** 对话框背景色 **更多信息**参见章节“颜色代码 (页 70)” – 了解缺省值 • **name** 格式的名称 • **type** 允许的值是 *cycle* ，表示一个用户循环窗口 • **xpos** 对话框左上角的 X 坐标（可选） • **ypos** 左上角的 Y 坐标（可选） • **width** X 方向上的长度（以像素为单位）（可选） • **height** Y 方向上的长度（以像素为单位）（可选）

标签名称	含义
FORM 续	通过 AUTOSCALE_CONTENT 属性可以确定不同屏幕分辨率时 Form 控件的性能。默认情况下，控件自动适应现有的屏幕分辨率。如果想自行控制位置和尺寸，则需将 Form 标签的属性值设为 OFF。 **属性：** **autoscale_content** **值：** • **on** 控件的坐标自动与屏幕分辨率相适应（默认） • **off** 控件的坐标使用时不会变化 **示例：** <form name="main_form" autoscale_content="off"> … … … </form>
FORM 续	属性： • **textfile** 该属性可以指定格式相关的文本文件。系统使用 EASY_XML 名称作为标准文本。 • **textcontext** 该属性可以指定待使用文本的名称。此属性仅在与属性 **textfile** 结合时才生效。

标签名称	含义

FORM 续	要使用文件 easyscreen.ini 中保存的标准版式，需要使用属性 LAYOUT。此处要给出的值是要使用的版式的名称。
	提示：
	默认设置的更改只对于弹出窗口生效，因为这里调用的窗口未隐去。
	
	属性：
	layout
	
	语法：
	`<form name="窗体名称" layout="版式名称" >`
	…
	…
	`</form>`
	
	示例：
	`<form name="form0" layout="slstandardscreenlayout.`
	`SlStandardScreenLayout.LowerForm" >`
	…
	…
	`</form>`
	
	除了预定义的窗口位置和尺寸之外，还可以在文件 easyscreen.ini 中保存自定义。
	easyscreen.ini 中的记录：
	`[640x480]`
	`MyPanel = x:=0, y:=220, width:=340, height:=174`
	`[800x480]`
	`MyPanel = x:=0, y:=220, width:=420, height:=174`
	
	示例：
	`<form name="form0" layout="MyPanel" >`
	…
	…
	`</form>`

标签名称	含义
FORM 续	**对话框信息：** • **INIT** • **PAINT** • **TIMER** • **CLOSE** • **FOCUS_IN** • **INDEX_CHANGED** • **EDIT_CHANGED** • **GESTURE_EVENT** • **KEY_EVENT** • **MESSAGE** • **MOUSE_EVENT** • **RESIZE** • **CHANNEL_CHANGED** • **LANGUAGE_CHANGED**
FORM 续	

标签名称	含义

FORM 续	**语法：**
	`<FORM name = "<dialog name>" color = "#ff0000">`
	示例：
	`<FORM name = "R-Parameter">`
	` <INIT>`
	` <DATA_ACCESS type = "true" />`
	` <CAPTION>R - Parameter</CAPTION>`
	` <CONTROL name = "edit1" xpos = "322" ypos = "34" refvar = "nck/Channel/Parameter/R[1]" />`
	` <CONTROL name = "edit2" xpos = "322" ypos = "54" refvar = "nck/Channel/Parameter/R[2]" />`
	` <CONTROL name = "edit3" xpos = "322" ypos = "74"`
	` </INIT>`
	
	`  <PAINT>`
	` <TEXT xpos = "23" ypos = "34">R - Parameter 1</TEXT>`
	` <TEXT xpos = "23" ypos = "54">R - Parameter 2</TEXT>`
	` <TEXT xpos = "23" ypos = "74">R - Parameter 3</TEXT>`
	
	` </PAINT>`
	
	`</FORM>`

INIT	**对话框信息** 在对话框创建之后立即处理该标签。在此创建所有的输入单元以及对话框窗口的“热连接”。

标签名称	含义

KEY_EVENT	对话框信息
	标签 KEY_EVENT 可将对键盘事件的评估链接到标签 FORM 中。如果在 FORM 中存在该标签，则系统会将 MF2 键盘代码发送给有效的 FORM。如果变量 **$actionresult** 未置零，系统会接着处理键盘事件。
	键盘代码会作为整型值在变量 **$keycode** 中提供。
	
	示例：
	在变量 **exclude_key** 中输入的信号应从输入电流中剔除。
	
	`<LET name="stream" type="string"/>`
	`<LET name="exclude_key" type="string"/>`
	
	`<FORM name = "keytest_form">`
	`  <INIT>`
	`  <CONTROL name = "p1" xpos = "120" ypos = "84" width ="200" refvar="stream" hotlink="true" />`
	`  <CONTROL name = "p2" xpos = "160" ypos = "104" width ="8" refvar="exclude_key" hotlink="true" />`
	`  </INIT>`
	
	`  <PAINT>`
	`  <text xpos = "8" ypos = "84">data stream</text>`
	`  <text xpos = "8" ypos = "104">exclude key</text>`
	`  </PAINT>`
	
	`  <KEY_EVENT>`
	`  <LET name="excl_keycode" type="string"/>`
	`  <OP>excl_keycode = exclude_key</OP>`
	`  <type_cast name="excl_keycode" type="int" />`
	`  <PRINT text="%d %d">$keycode, excl_keycode</PRINT>`
	
	`  <IF>`
	`  <CONDITION>$keycode == excl_keycode</CONDITION>`
	`  <THEN>`
	`  <op> $actionresult = 0</op>`
	`  </THEN>`
	`  </IF>`
	
	`  </KEY_EVENT>`
	
	`</FORM>`

标签名称	含义

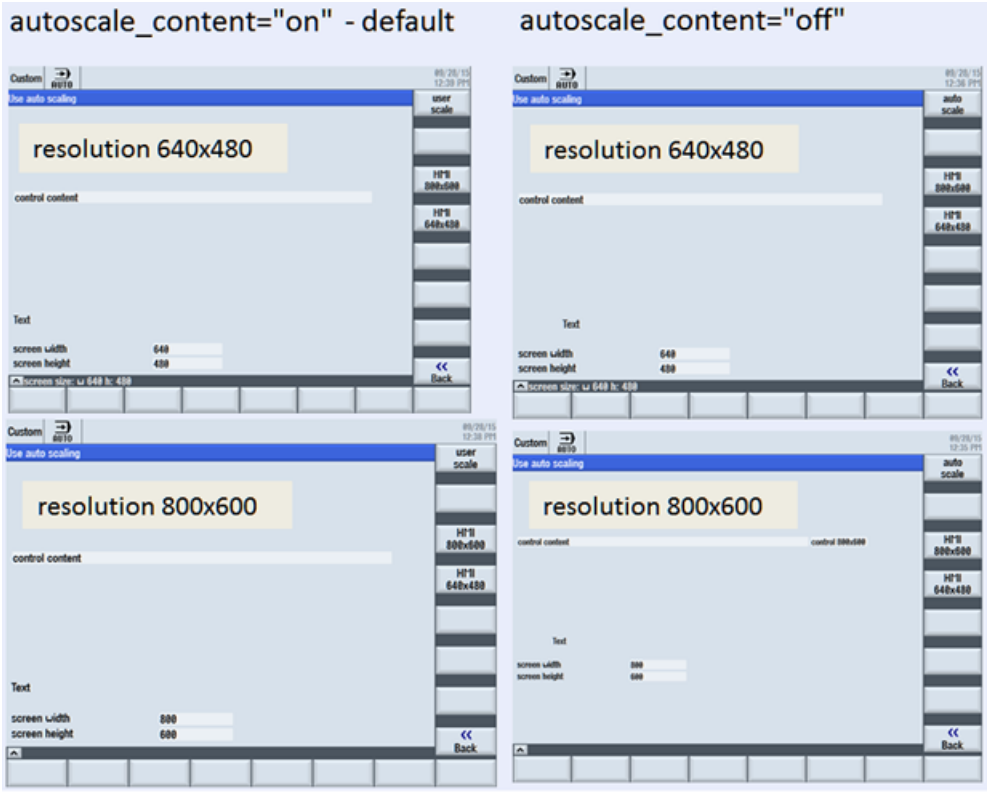
MOUSE_EVENT	该标签可将对鼠标事件的编辑链接到脚本中。具有以下鼠标操作时执行：
	• 某个按键被按下
	• 某个按键被松开
	• 鼠标被移动
	解析器在结构中提供信息并创建带有以下元素的结构变量**$mouse_event**：
	
	结构元素：
	• **type**
	操作的编码
	– 2 - 某个按键被按下
	– 3 - 某个按键被松开
	– 5 - 鼠标被移动
	• **x**
	基于当前屏幕分辨率的光标的 X 位置，单位：像素
	• **y**
	基于当前屏幕分辨率的光标的 Y 位置，单位：像素
	• **id**
	名称
	– -1，该位置没有分配控件时
	– != -1，鼠标光标位于控件内时，提供属性 **idemdata**
	• **button**
	包含事件时间点时按键的状态
	– 0 - 无按键
	– 1 - 左侧按键
	– 2 - 右侧按键
	– 4 - 中间按键
	按键可通过逐位 ODER 运算连接起来。
	
	示例：
	`<MOUSE_EVENT>`
	`  <print text="button %d type %d x %d y %d ">$mouse_event.button, $mouse_event.type, $mouse_event.x, $mouse_event.y</print>`
	`</MOUSE_EVENT>`

标签名称	含义

MESSAGE	**对话框信息**
	如果在脚本中执行指令 Send_message，解析器会执行标签 MESSAGE。值 p1 和 p2 会在变量 $message_par1 和 $message_par2 中提供（参见标签 “SEND_MESSAGE”）。
	
	语法：
	`<MESSAGE>`
	`</MESSAGE>`
	
	示例：
	`<LET name="user_selection" />`
	
	`<SOFTKEY POSITION="3">`
	`<CAPTION>Set%nParameter</CAPTION>`
	`<SEND_MESSAGE>1, 10</SEND_MESSAGE>`
	`</SOFTKEY>`
	…
	…
	
	`<FORM>`
	…
	…
	`<MESSAGE>`
	`<SWITCH>`
	`<CONDITION>$message_par1</CONDITION>`
	`<CASE value="1">`
	`<OP> user_selection = $message_par2 </OP>`
	…
	…
	`</CASE>`
	`</SWITCH>`
	`</MESSAGE>`
	…
	…
	`</FORM>`

标签名称	含义

SEND_MESSAGE	该标签会向有效的 FORM 标签发送含两个参数的信息，该信息在标签“Message”中进行处理（另见 MESSAGE）。
	
	语法：
	`<SEND_MESSAGE>p1, p2</SEND_MESSAGE>`
	
	示例：
	`<LET name="user_selection" />`
	
	`<SOFTKEY POSITION="3">`
	`  <CAPTION>Set%nParameter</CAPTION>`
	`  <SEND_MESSAGE>1, 10</SEND_MESSAGE>`
	`</SOFTKEY>`
	…
	…
	`<FORM>`
	…
	…
	`<MESSAGE>`
	`<SWITCH>`
	`<CONDITION>$message_par1</CONDITION>`
	`<CASE value="1">`
	`<OP> user_selection = $message_par2 </OP>`
	…
	`</CASE>`
	`</SWITCH>`
	`</MESSAGE>`
	…
	…
	`</FORM>`

标签名称	含义
RESIZE	**对话框信息** 该标签可将对 RESIZE 事件的编辑链接到脚本中。结果由动态分辨率切换生成。 

标签名称	含义

RESIZE 续	**示例：**
	`<let name="screen_size" type="StructSize" />`
	`<form name="menu_userscale_form" autoscale_content="off">`
	` <init>`
	` <caption>Use auto scaling</caption>`
	
	` <function name="hmi.get_hmi_resolution">screen_size</function>`
	` <print text="screen size: w %d h: %d">screen_size.width, screen_size.height</print>`
	
	` <data_access type="true" />`
	` <control name="s_c" xpos="8" ypos="140" fieldtype="readonly" width="500" refvar="string_var" hotlink="true"/>`
	` <if>`
	` <condition>screen_size.width == 800</condition>`
	` <then>`
	` <control name="s_1" xpos="508" ypos="140" fieldtype="readonly" width="100" refvar="string_var1" hotlink="true"/>`
	` </then>`
	` </if>`
	` <control name="s_w" xpos="200" ypos="350" fieldtype="readonly" refvar="screen_size.width" hotlink="true"/>`
	` <control name="s_h" xpos="200" ypos="370" fieldtype="readonly" refvar="screen_size.height" hotlink="true"/>`
	` </init>`
	` <paint>`
	` <text xpos ="68" ypos="310">Text</text>`
	` <text xpos ="8" ypos="350">screen width</text>`
	` <text xpos ="8" ypos="370">screen height</text>`
	` </paint>`
	
	` <resize>`
	` <function name="hmi.get_hmi_resolution">screen_size</function>`
	` <print text="resize_event screen size: w %d h: %d">screen_size.width, screen_size.height</print>`
	` <switch>`
	` <condition>screen_size.width</condition>`
	` <case value="640">`
	` <function name="control.delete">_T"s_1"</function>`
	` </case>`
	` <case value="800">`
	` <control name="s_1" xpos="508" ypos="140" fieldtype="readonly" width="100" refvar="string_var1" hotlink="true"/>`
	` </case>`
	` </switch>`
	` </resize>`
	`</form>`

标签名称	含义

CHANNEL_CHANGED	**对话框信息**
	标签使得可以对通道转换事件进行处理。在用户按下通道接通按钮时会启动进程。
	
	语法：
	`<channel_changed>`
	`</channel_changed>`
	
	示例：
	通道更换时，应关闭并再次打开 FORM 来显示已选择的通道的数据。
	`<channel_changed>`
	`  <open_form name = "form1" reopen="true"/>`
	`</channel_changed>`
	
	或
	
	当通道更换时，会激活另一个菜单。
	`<channel_changed>`
	`  <navigation>menu2</navigation>`
	`</channel_changed>`

LANGUAGE_CHANGED	**对话框信息**
	标签使得可以对语言转换要求进行处理。用户在打开对话框时更换语言会启动进程。
	
	语法：
	`<language_changed>`
	`</language_changed>`
	
	示例：
	语言更换时，应关闭并再次打开 FORM 来实现对生效语言的文本元素的控制。
	`<language_changed>`
	`  <open_form name = "form1" reopen="true"/>`
	`</language_changed>`

标签名称	含义

FOCUS_IN	**对话框信息**
	当系统将焦点设置到控件时，调用该标签。为了识别控件，系统将控件的名称复制到变量 **$focus_name** 中并将 **item_data** 的属性值复制到变量 **$focus_item_data** 中。
	例如可以使用该信息，来输出与焦点位置有关的图像。
	示例：
	`<focus_in>`
	`<PRINT text="focus on filed:%s, %d">$focus_name,`
	`$focus_item_data </PRINT>`
	`</focus_in>`

PAINT	**对话框信息** 该标签在显示对话框时执行。在此要给定需要在对话框中显示的所有文本与图像。 当系统确定对话框的一些地方需要重新显示时，会继续执行该标签。这可以通过，例如，关闭上一级窗口来实现。
TIMER	**对话框信息** 循环处理该标签。 每个 FORM 都分配了一个 TIMER，大约每 100 ms 就会触发一次 TIMER 标签的执行。

标签名称	含义

CAPTION	该标签包含有对话框的标题。
	在 INIT （初始化）标签内使用该标签。
	标题行可划分为多列。
	为了划分标题行，在输出文本前应使用属性“define_section”对标签“caption”进行编程。
	属性“define_section”确定列数。
	
	通过“property”属性定义每一列的长度、起始位置和文本方向。位置和长度说明是基于表格宽度的百分数值。
	属性值“value”表示列编号（从零开始）
	
	`<caption define_sections="3">`
	`<property value="0" length="20" position="2" alignment="left" />`
	`<property value="1" length="20" position="79"`
	`alignment="right" />`
	`</caption>`
	
	然后可以借助列索引指定属性 index 来指定列文本。
	`<caption index=“0”>Spalte1</caption>`
	`<caption index=“1”>Spalte2</caption>`
	
	语法：
	`<CAPTION>Titel</CAPTION>`
	
	示例：
	`<CAPTION>my first dialogue</CAPTION>`

CLOSE	**对话框信息** 在关闭对话框之前处理该标签。

标签名称	含义

CLOSE_FORM	该标签会关闭激活的对话框。
	只有通过 MMC 指令打开对话框且为操作者提供了关闭对话框的软键功能时，该指令才是必须的。一般而言，会自动管理对话框，而无需另外加以关闭。
	
	语法：
	`<CLOSE_FORM/>`
	
	示例：
	`<softkey_ok>`
	` <caption>OK</caption>`
	` <CLOSE_FORM />`
	` <navigation>main_menu</navigation>`
	`</softkey_ok>`

标签名称	含义
CONTROL	该标签用来创建控件。 **语法：** `<CONTROL name = "<control name>" xpos = "<X 坐标>" ypos = "<Y 坐标>" refvar = "<NC 变量>" hotlink = "true" format = "<格式>" />` **提示：** 必须将用于设置属性值的变量声明为全局变量。 **属性：** • **name** 字段名称 该名称会同时显示一个局部变量，并且在 FORM 中不允许重复使用。 • **xpos** 左上角的 X 坐标 • **ypos** 左上角的 Y 坐标 • **fieldtype** 字段类型 如未给定类型，则字段被自动设置为 Edit 型字段。 – **edit** 数据可以修改。 – **readonly** 数据无法修改。 **combobox** 字段显示相应的名称来代替数值。 如果选择了字段类型“**combobox**”，还要为字段设置需要显示的表达式。为此要使用标签 <ITEM> 。 组合框（Combo Box）将当前选定文本的下标保存在控件所属的变量中（参见属性 **refvar**）。 在创建了控件后，其他元素可通过函数 **addItem** 或 **insertItem** 来添加。 – **progressbar** 在值域 0 到 100 内显示进度条。 值域与待显示数据的最小值和最大值相匹配。

标签名称	含义

CONTROL 续	• **fieldtype**
	– **graphicbox**
	该字段类型会生成一个 2 维图形控件。通过该标签 <ITEM> 会向控件中添加一个图形元素。参数 **width** 和 **height** 用来指定图形框的宽度和高度。
	在创建了控件后，其他元素可通过函数 **addItem** 或 **insertItem** 来添加。参数 **itemdata** 在该控件时不进行处理。
	使用控件可分配一个参考变量，在 100 ms 栅格内记录值。最多记录 10000 个值。属性为 **max** 时可以无穷记录数值，达到最大记录时间时，删除最早记录的值。记录时间单位为毫秒。
	在非连续的时间内保存的值以相互连接的直线的形式表示控件。属性 **channel** 使得能记录最多三个其他的参考变量。索引从一开始。索引零表示设置控件标签中分配的变量的属性。
	示例：
	`<control name= "c_gbox1" xpos = "250" ypos="24" width="240" height="356" fieldtype="graphicbox" refvar="val" hotlink="true" COLOR_BK="#ffffff">`
	`  <property max="60000" /> <!-- steps of ms -->`
	`  <property min="0" />`
	`  <property channel="0" color="#00ffff" penwidth="3"/>`
	`  <property ORDINATE="Y sinus" />`
	`  <property ABSCISSA="time *100 ms" />`
	`  <property channel="1" refvar="val2" color="#000000" />`
	`</control>`

标签名称	含义
CONTROL 续	**图形框示例：** `<CONTROL name= "graphic" xpos = "8" ypos="23" width="300" height="352" fieldtype="graphicbox" />` • 添加元素： 可使用函数 **additem** 或 **loaditem** 添加元素。 可使用以下 2 维元素： – 直线 - l(inc) – 圆弧 - c(ircle) – 点 - p(oint) **元素结构：** <Elementtyp>；坐标 • 直线： l; xs; ys; xe, ye l - 直线符号 Xs - X 起始位置 Ys - Y 起始位置 Xe - X 结束位置 Ye - Y 结束位置 • 圆： C, xs, ys, xe, ye, cc_x, cc_y, r C - 圆弧符号 Xs - X 起始位置 Ys - Y 起始位置 Xe - X 结束位置 Ye - Y 结束位置 Cc_x – 圆心的 X 坐标 Cc_y – 圆心的 Y 坐标 • 半径： R • 点： P, x, y P - 点的符号 X - X 坐标 Y - Y 坐标 • 清除图形： 可使用函数 **empty** 清空内容。

标签名称	含义
CONTROL 续	以下字段类型在“自定义按钮 (页 233)”一章中说明。 • **fieldtype** – **pushbutton** 该字段类型是“按钮”或者“开关”。 – **radiobutton** 该字段类型用于单选或多选。 – **checkbox** 该字段类型用于多选。 – **groupbox** 该字段类型将一组控件框在一起，并设置一个标题。 – **switch** 该字段类型是一种图形元件，借助一个图标指示两个状态中的一个。 – **scrollarea** 该字段类型用于在指定区域内显示控件。

标签名称	含义
CONTROL 续	• **fieldtype** – **listbox** 该字段类型会生成一个空的列表框控件。 通过标签 <ITEM> 会向列表框中添加一个列表框元素。 通过属性 ITEM **value** 为该元素赋唯一的一个值。 该值比如可用于明确标记该元素。 参数 **width** 和 **height** 用来指定列表框的宽度和高度。 在创建了控件后，其他的列表框元素可通过函数 **addItem**、**insertItem** 或 **loadItem** 来添加。 在创建了控件后，其他的列表框元素可通过函数 **addItem**、或 **insertItem** 来添加。参数 **itemdata** 在该控件时不进行处理。 – **itemlist** 该字段类型会生成一个静态控件，取代数值而显示相应的名称。 标签 <ITEM> 可为字段指定一个名称。 • **item_data** 可以使用用户专用的整数值为该属性赋值。该值被提供给 FOCUS_IN 信息，用来识别焦点字段。 • **refvar** 可以与字段互连的参考变量的名称（可选）。 • **hotlink** = "TRUE" 如果参考变量的值发生了变化，该字段会自动进行更新（可选）。 • **format** 该属性定义指定变量的显示格式。 格式定义参见"print-Tag "（可选）。 属性 **format** 使得整型值可以显示在其他进制中，也就是作为数组类型**编辑**和**只读**的布尔值。不支持字符数量和排列的格式化。 可使用下列显示： • **%o** - 八进制的 • **%x** - 十六进制的 • **%n** - 二进制的 • **%b** - 布尔

CONTROL 续	**列的创建示例，属性 property：**
	`<control name="lb1" xpos = "10" ypos = "94" width="200" height="250" fieldtype="listbox" refvar="tmp" hotlink="true">`
	`  <property columns="4" />`
	`  <property column="0" type="width">190</property>`
	`  <property column="1" type="width">100</property>`
	`  <property column="2" type="width">190</property>`
	`  <property column="3" type="width">16</property>`
	`</control>`

标签名称	含义

CONTROL 续	**属性 format 示例：**
	`<let name="val_test">255</let>`
	`<form name="main_form">`
	`<init>`
	`<caption>bool hex octal bin display</caption>`
	`<control name="test_oVal" xpos="250" ypos="60" refvar="val_test" hotlink = "true" />`
	`<control name="test_bool" xpos="250" ypos="90" refvar="val_test" hotlink = "true" format="%b"/>`
	`<control name="test_hex" xpos="250" ypos="120" refvar="val_test" hotlink = "true" format="%x"/>`
	`<control name="test" xpos="250" ypos="150" refvar="val_test" hotlink = "true" format="%o"/>`
	`<control name="test_b" xpos="200" ypos="180" width="300" refvar="val_test" hotlink = "true" format="%n"/>`
	`</init>`
	`<paint>`
	`<text xpos="16" ypos="60">integer value</text>`
	`<text xpos="16" ypos="90">boolean display</text>`
	`<text xpos="16" ypos="120">hexadecimal display</text>`
	`<text xpos="16" ypos="150">octal display</text>`
	`<text xpos="16" ypos="180">binary display</text>`
	`</paint>`
	`</form>`

标签名称	含义
CONTROL 续	**属性：** • **color_bk** 该属性设置控件的背景颜色。 • **color_fg** 该属性设置控件的前景颜色。 **更多信息**参见章节“颜色代码 (页 70)” • **display_format** 该属性定义了指定变量的处理格式。在存取 PLC 浮点型变量时必须使用该属性，因为要通过双字读取来进行存取。 允许使用下列数据格式： – FLOAT – INT – DOUBLE – STRING 向列表框、图形框或组合框分配表达内容（例如要显示的文本或图形）： **语法：** <ITEM>表达</ITEM> <ITEM value ="<值>">表达</ITEM>

标签名称	含义

CONTROL 续	**示例：**
	`<CONTROL name = "button1" xpos = "10" ypos = "10" fieldtype = " combobox ">`
	`<ITEM>text1</ITEM>`
	`<ITEM>text2</ITEM>`
	`<ITEM>text3</ITEM>`
	`<ITEM>text4</ITEM>`
	`</CONTROL>`
	如果要为表达式分配任意一个整数值，则要为标签添加属性 **value = "数值"** 。
	控制变量含有已赋值的对象值，用来代替连续的编号。
	示例：
	`<CONTROL name = "button1" xpos = "10" ypos = "10" fieldtype = " combobox ">`
	`<ITEM value = "10">text1</ITEM>`
	`<ITEM value = "20">text2</ITEM>`
	`<ITEM value = "12">text3</ITEM>`
	`<ITEM value = "1">text4</ITEM>`
	</CONTROL>
	进度条示例
	`<CONTROL name = "progress1" xpos = "10" ypos = "10" width = "100" fieldtype = "progressbar" hotlink = "true" refvar = "nck/Channel/GeometricAxis/actProgPos[1]">`
	`<PROPERTY min = "0" />`
	`<PROPERTY max = "1000" />`
	`</CONTROL>`
	列表框示例：
	`<let name="item_string" type="string"></let>`
	`<let name="item_data" ></let>`
	`<CONTROL name="listbox1" xpos = "360" ypos="150" width="200" height="200" fieldtype="listbox" />`
	• 添加元素：
	可使用函数 **additem** 或 **loaditem** 添加元素。
	• 清除内容：
	可使用函数 **empty** 清空内容。
	`<op> item_string = _T"text1\\n" </op>`
	`<function name="control.additem">_T"listbox1", item_string, item_data </function>`
	`<op> item_string = _T"text2\\n" </op>`
	`<function name="control.additem">_T"listbox1", item_string, item_data </function>`

标签名称	含义

CONTROL 续	**示例 itemlist:**
	`<CONTROL name = "itemlist1" xpos = "10" ypos = "10" fieldtype = "itemlist">`
	`<ITEM value = "10">text1</ITEM>`
	`<ITEM value = "20">text2</ITEM>`
	`<ITEM value = "12">text3</ITEM>`
	`<ITEM value = "1">text4</ITEM>`
	`</CONTROL>`

CONTROL 续	控件 Imagebox 控制位图格式或 GIF 格式的图形。如果图形大于所显示的区域，则出现滚动条控件。 系统提供函数 CONTROL.IMAGEBOXSET，用于控制可见区域。 **更多信息**参见章节“预定义功能 (页 158)” **语法：** `<function name="control.imageboxget"> ... </function>`
CONTROL 续	**属性：** • **disable** 该属性禁止/允许在 EDIT 控件中输入。 • **tooltip** 当光标移动到控件上时，显示提示文本。 • **factor** 换算系数 • **font** 指定字体 • **fontpixelsize** 字体大小 - 以像素为单位指定需要的字符高度。

标签名称	含义
CONTROL 续	**创建后修改控件** 可使用控件标签在创建后修改已有控件的控件属性。必须同时给定待修改控件的名称及新属性。该操作可在 "Form" 标签内部进行。例如可修改以下属性： • name • xpos • ypos • width • height • color_bk • color_fg • access level • fieldtype • itemdata • min • max • default • disable • tooltip • font • factor 无法修改参考变量。如果要通过软键“事件”触发来修改属性，则须使用标签"send message"，以上下文格式传送该要求。使用标签 "message" 采集信息。

标签名称	含义

CONTROL 续	**运行命令中的控件更改**
	在运行时更改控件属性的另一种方式是在运行命令中进行更改。为此需要指明控件的名称以及要分配新值的属性。属性与控件名称通过符号“点”隔开。
	语法：
	<Name>.<Eigenschaft>
	示例：
	`...`
	`...`
	`<let name="value" />`
	`<let name="w" />`
	`<let name="h" />`
	`...`
	`...`
	`<control name="c_move" xpos="$xpos" ypos="124" />`
	`...`
	`...`
	`<op>`
	`c_move.xpos = 300;`
	`value = c_move.xpos;`
	`h = c_move.height;`
	`w = c_move.width;`
	`</op>`

标签名称	含义

HELP_CONTEXT	该标签用来确定所要调用的帮助主题。在 INIT 初始化数据块中进行编程。
	系统 808/802D sl
	在属性中给定的名称会加上前缀 XmlUserDlg_ 并转发到帮助系统中。这样可以从在线帮助主题中提取出帮助文件的关联结构。
	激活帮助系统的过程：
	1. 按下“信息”键。
	2. 对话画面会显示 "my_dlg_help"。
	3. 解析器会将表达式转换为 "XmlUserDlg_my_dlg_help" 。
	4. 帮助系统激活。
	5. 传递查找关键字 "XmlUserDlg_my_dlg_help"。
	系统 840D sl/828D
	有关帮助手册的结构和生成方式的**更多信息**，请参见
	SINUMERIK Operate 调试手册
	
	属性：
	• **name**
	在属性中给定的名称会转发到帮助系统中。需要给出帮助手册中的标签 **entry** 中使用的文件名。
	• **anchor**
	使用此属性可给定关键字。
	
	语法：
	`<HELP_CONTEXT name="<file name>" anchor="key word"/>`
	
	示例：
	`<form name = "form1">`
	` <init>`
	`  <help_context name="chapter_1.html" anchor="Keyword_1" />`
	`...`
	`...`
	`  <control name = "cpwd" xpos = "322" ypos = "64" >`
	`   <property password="true" />`
	`   <property help_context="chapter_1.html" anchor="Keyword_2" />`
	`  </control>`
	`</form>`

标签名称	含义
DATA_ACCESS	该标签在储存用户输入数据时用来控制对话框窗口的特性。 在 INIT 标签中确定其特性。 如未使用标签，则总是使用输入数据中间存储器。 例外：在 **hotlink** 设置为 **true** 的控制器中总是直接写入和读取。 **属性：** • **type** = "TRUE" – 不对输入值进行缓存。对话框窗口直接将输入值复制到参考变量中。 • **type** = "FALSE" – 值只通过标签 **UPDATE_CONTROLS** type = "FALSE" 复制到参考变量中。 **示例：** `<DATA_ACCESS type = "true" />`

标签名称	含义

DEBUG	标签用于保存可在脚本中被编程的踪迹输出。为描述要被保存的文本，请使用属性 **text**。此属性和其值符合 **text** 属性和打印指令的值。
	一般来说不会运行调试标签，因为这会使系统运行变慢。若要激活调试输出，需要将标签 **DialogGui** 或 **AGM** 中的属性 **debug_msg** 的值编程为 **on**。
	调试输出和 Parser 报警信息一同存储在下列文件中：
	card/oem/sinumerik/hmi/dvm/log/dvm.log
	
	语法：
	`<debug text="<文本参见打印指令>">变量参见打印指令</debug>`
	
	Easy XML 示例：
	`<DialogGui debug_msg="on">`
	`...`
	`...`
	`<control name="test_bool" xpos="250" ypos="90"`
	`refvar="val_test" hotlink = "true" format="%b"/>`
	`<debug text="main_form control test_bool created`
	`value:%d">test_bool</debug>`
	
	文件 dvm.log 中的条目：
	`date: 10/22/ 2018 time: 09:38:44:351`
	`Debug xmldial.xml 17: main_form control test_bool`
	`created value:255`
	
	Easy Extend 示例 ：
	`<AGM debug_msg="on">`
	`...`
	`...`
	`<control name="test_bool" xpos="250" ypos="90"`
	`refvar="val_test" hotlink = "true" format="%b"/>`
	`<debug text="main_form control test_bool created`
	`value:%d">test_bool</debug>`
	
	文件 dvm.log 中的条目：
	`date: 10/22/ 2018 time: 09:38:44:351`
	`Debug xmldial.xml 17: main_form control test_bool`
	`created value:255`

标签名称	含义

EDIT_CHANGED	**对话框信息**
	当 EDIT 控件的内容发生了变化时，调用该标签。
	为了识别控件，系统将控件的名称复制到变量 **$focus_name** 中并将 **item_data** 的属性值复制到变量 **$focus_item_data** 中。
	
	示例：
	`<EDIT_CHANGED>`
	`<print text="index changed filed:%s, %d"> $focus_name, $focus_item_data </print>`
	…
	…
	`</EDIT_CHANGED>`

GESTURE_EVENT	该标签用于执行多点触控操作中的手势。 该标签在“多点触控操作 (页 225)”一章中说明。

INDEX_CHANGED	**对话框信息**
	当操作员修改了复合框的选择时，调用该标签。
	为了识别控件，系统将控件的名称复制到变量
	$focus_name 中，将属性 **item_data** 的值复制到变量**$focus_item_data** 中。
	提示：
	分配给控件的参考变量此时还未与控件变量进行匹配，所含的仍是上一次复合框选择的下标。
	
	示例：
	`<INDEX_CHANGED>`
	`<print text="index changed filed:%s, %d"> $focus_name, $focus_item_data </print>`
	…
	…
	`</INDEX_CHANGED>`

标签名称	含义

MENU	该标签定义一个菜单，其中包含有软键说明和要打开的对话框。
	
	属性：
	• **name**
	菜单名称
	
	语法：
	`<MENU name = "<menu name>">`
	…
	`<open_form …>`
	…
	`<SOFTKEY …>`
	`</SOFTKEY>`
	`</MENU>`

NAVIGATION	该标签用来确定所要调用的菜单。可以在软键程序块、菜单程序块和 Form 中设置。如果给标签分配了一个变量名称作为值，则解析器会激活变量中的菜单。
	菜单程序块中，导航至说明位置。后续说明将不再执行。
	
	提示：
	在需要通过变量的内容定义导航目标的情况下，必须将此变量声明为全局变量。
	
	语法：
	`<NAVIGATION>menu name</NAVIGATION>`
	
	示例：
	`<menu name = "main">`
	`<softkey POSITION="1">`
	`<caption>sec. form</caption>`
	`<navigation>sec_menu</navigation>`
	`</softkey>`
	`</menu>`
	
	`<menu name = "sec_menu">`
	`<open_form name = "sec_form" />`
	`<softkey_back>`
	`<navigation>main</navigation>`
	`</softkey_back>`
	`</menu>`

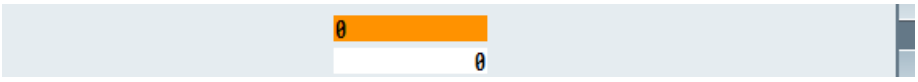
标签名称	含义

OPEN_FORM	该标签可以打开名称下指令的对话框窗口。
	若给定的对话框已经激活，则不会运行此标签。
	对属性 **reopen** 的说明可以强制重复打开对话框。当系统状态改变且预设了窗口内容的重组时，这可能是必要的。
	
	属性：
	• **name**
	对话框窗口的名称
	• **reopen**
	若属性值为 **true**，则在重复调用标签 **open_form** 时会检验这里是否为一个已激活的对话框。如果是，则 Parser 会关闭并再次打开已激活的 Form。
	
	语法：
	`<OPEN_FORM name = "<form name>" />`
	
	示例：
	`<menu name = "main">`
	`  <open_form name = "main_form" />`
	`  <softkey POSITION="1">`
	`  <caption>main form</caption>`
	`  <navigation>main</navigation>`
	`  </softkey>`
	`</menu>`
	
	`<form name="main_form">`
	`  <init>`
	`  </init>`
	
	`  <paint>`
	`  </paint>`
	
	`</form>`

标签名称	含义

PROPERTY	使用该标签可以确定操作单元的附加特性。该标签嵌入在控件标签中。
	
	属性：
	• **max** = “<最大值>”
	• **min** = “<最小值>”
	• **default** = “<预设值>”
	• **factor** = “换算系数”
	• **color_bk** = “<背景颜色代码>”
	• **color_fg** = “<字体颜色代码>”
	• **password** = “<true>” - 所输入的字符显示为“*”
	• **multiline** = “<true>” - 在 EDIT 控件中允许多行输入
	• **disable** = “”<true/false>” - 在 EDIT 控件禁止/允许输入
	• **transparent** = “位图的透明色”
	更多信息参见章节“颜色代码 (页 70)”
	• **tooltip** = 当光标移动到控件上时，显示提示文本。
	语法为： <property tooltip="提示文本" />
	• **abscissa** = “第一个坐标轴的名称”（仅允许用于图形框）
	• **ordinate** = “第二个坐标轴的名称”（仅允许用于图形框）
	示例：
	`<CONTROL name = "progress1" xpos = "10" ypos = "10" width = "100" fieldtype = "progressbar" hotlink = "true" refvar = "nck/Channel/GeometricAxis/actProgPos[1]">`
	`<PROPERTY min = "0" />`
	`<PROPERTY max = "1000" />`
	`</CONTROL>`
	`<CONTROL name = "edit1" xpos = "10" ypos = "10">`
	`<PROPERTY min = "20" />`
	`<PROPERTY max = "40" />`
	`<PROPERTY default = "25" />`
	`</CONTROL>`

标签名称	含义

PROPERTY 续	**属性：**
	alignment - 使文本左对齐或右对齐。默认设置为文本左对齐。
	
	值：
	• **left**
	• **right**
	
	语法：
	`alignment="<right/left>"`
	
	示例：
	`<control name="edit2" xpos="208" ypos="52" >`
	`<property alignment="right"/>`
	`</control>`
	
	

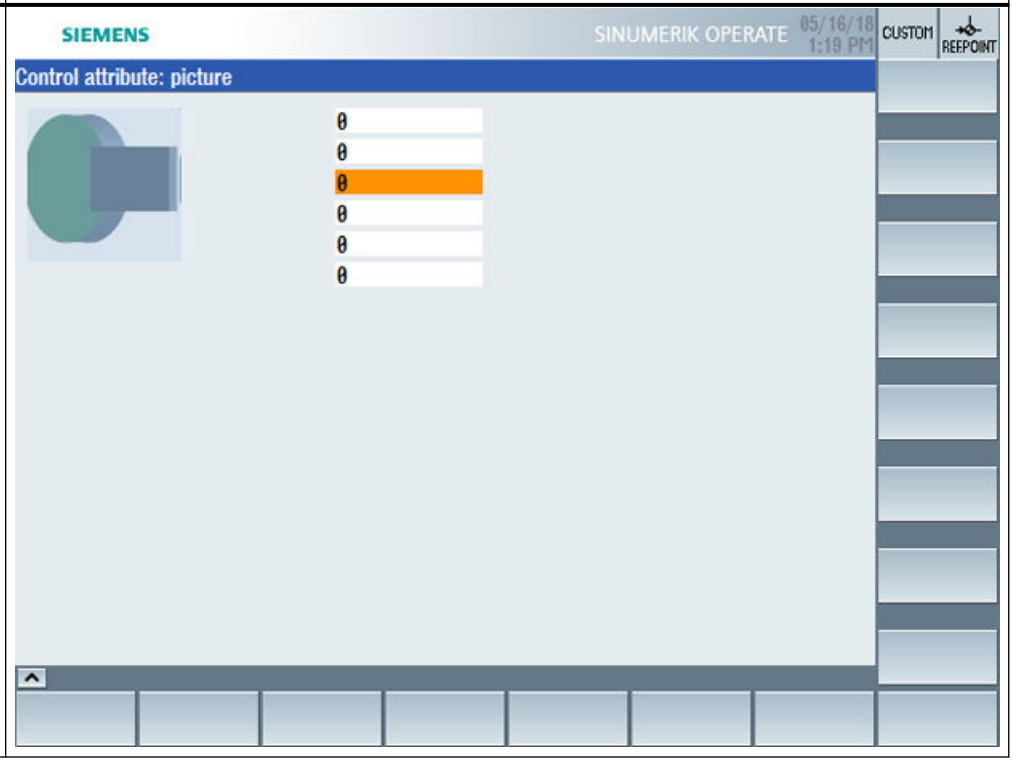
标签名称	含义

PROPERTY 续	**属性：**
	parent - 确定控件的所属关系。默认设置为控件属于窗体。如要将控件分配给滚动视图，则要将其指定为父窗体。
	
	值：
	父窗体名称
	
	示例：
	`...`
	`<control name="area" xpos="8" ypos="72" width="554" height="320" fieldtype="scrollarea" >`
	`<control name="test" xpos="20" ypos="40" width="20" fieldtype="readonly" />`
	`</control >`
	`<control name="test1" xpos="20" ypos="80" width="20" fieldtype="readonly" parent="area"/>`
	
	test
	
	test1
	
	`<init>`
	`<control name="area" xpos="8" ypos="72" width="554" height="120" fieldtype="scrollarea">`
	`<control name="test" xpos="20" ypos="40" width="80" fieldtype="readonly" type="string"/>`
	`</control >`
	
	`<control name="test1" xpos="20" ypos="80" width="80" fieldtype="readonly" type="string" parent="area"/>`
	`<op>test = _T"test"</op>`
	`<op>test1 = _T"test1"</op>`
	`<update_controls type="true" />`
	`...`

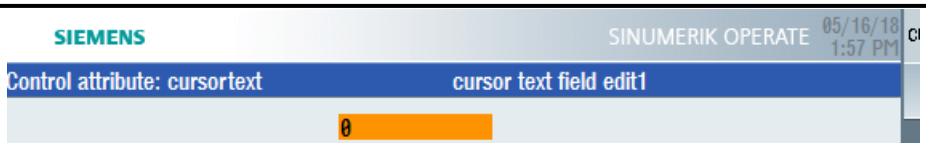
标签名称	含义
PROPERTY 续	**属性：** **picture** - 确定要显示的图片 如果此区域处于输入状态，则显示给定的图片或者将图片名保存在一个变量中。此属性应与 **refvar** 属性合用，以便确定数据宿。 为显示图片或动画，需要给定 **imagebox** 类型的控件名称。此控件可管理图片。 给定一个局部变量名称时，此变量的数据类型必须是 **String**。 图片将一直显示到一个新名称分配到参考变量中为止。 **语法：** `<property picture="<Name der Grafik>" refvar="<Datensenke>"/>`

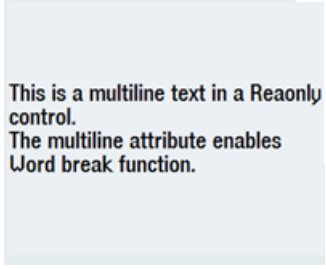
标签名称	含义

PROPERTY 续	**示例：**
	`<let name="cursor_icon" type="string" />`
	` <form name="main_form1">`
	`  <init>`
	`  <caption>Control attribute: picture</caption>`
	`  <control name="cursor_image" xpos="8" ypos="32" width="100" height="100" fieldtype="imagebox" />`
	`  <control name="edit1" xpos="208" ypos="32" >`
	`  <property picture="red_led_off.bmp" refvar="cursor_image"/>`
	`  </control>`
	`  <control name="edit2" xpos="208" ypos="52" >`
	`  <property picture="red_led_on.bmp" refvar="cursor_image"/>`
	`  </control>`
	`  <control name="edit3" xpos="208" ypos="72" >`
	`  <property picture="test.gif" refvar="cursor_image"/>`
	`  </control>`
	`  <control name="edit4" xpos="208" ypos="92" >`
	`  <property picture="red_led_off.bmp" refvar="cursor_image"/>`
	`  </control>`
	`  <control name="edit5" xpos="208" ypos="112" >`
	`  <property picture="red_led_off.bmp" refvar="cursor_icon"/>`
	`  </control>`
	`  <control name="edit6" xpos="208" ypos="132" >`
	`  <property picture="red_led_on.bmp" refvar="cursor_icon"/>`
	`  </control>`
	`  </init>`
	`  <timer><print text="%s">cursor_icon</print></timer>`
	` </form>`

标签名称	含义
	SIEMENS SINUMERIK OPERATE 05/16/18 1:19 PM CUSTOM REEPOINT Control attribute: picture 0 0 0 0 0 0

标签名称	含义
PROPERTY 续	**属性：** **cursortext** - 此属性确定要显示的光标文本。 当此区域为输入状态时，文本会在标题行输出。 对于此属性，还可以给定两个属性来确定文本和光标文本区域的对齐方式。默认文本左侧对齐。光标文本区域默认占据标题行宽度的 50%。 alignment="<right/left>" - 文本对齐 右对齐/左对齐 length="<宽度>" - 光标文本应当在标题行中占有的百分比 **语法：** <property cursortext="<text>" /> 或 <property cursortext="<text>" alignment="<right/left>" /> 或 <property cursortext="text2" alignment="r="<text>" alignment="<right/left>" length="<比率值>"/> **示例：** <form name="main_form2"> <init> <caption>Control attribute: cursortext</caption> <control name="edit1" xpos="208" ypos="32" > <property cursortext="cursor text field edit1" /> </control> <control name="edit2" xpos="208" ypos="52" > <property cursortext="cursor text field edit2" alignment="right"/> </control> <control name="edit3" xpos="208" ypos="72" > <property cursortext="cursor text field edit3" alignment="right" length="40"/> </control> <control name="edit4" xpos="208" ypos="92" > <property cursortext="cursor text field edit4" /> </control> </init> </form>

标签名称	含义
	

PROPERTY 续	**属性：**
	multiline - 实现 EDIT 或 Readonly 控件中文本的多行输出
	
	语法：
	`<property multiline="true" />`
	
	示例：
	`...`
	`...`
	`<op>`
	`textlist[2] = _T"This is a multiline text in a Reaonly control.\`
	`\nThe multiline attribute enables Word break function.";`
	`</op>`
	`<control name="lstring" xpos="8" ypos="120" width="200" height="160"`
	`fieldtype="readonly" refvar="textlist[2]" >`
	`<property multiline="true" />`
	`</control>`
	
	

标签名称	含义

PROPERTY 续	**属性：**
	• **help_context**
	此属性给一个控件分配一个帮助主题。需要指定帮助手册中标签 **entry** 中使用的文件名。
	• **anchor**
	使用此属性可给定关键字。此属性仅在与属性 **help_context** 结合时才生效。
	
	语法：
	`<property help_context="<file name>" anchor="<key word>"/>`
	
	示例：
	`<control name = "cpwd" xpos = "322" ypos = "64" >`
	`<property password="true" />`
	`<property help_context="chapter_1.html" anchor="Keyword_2" />`
	`</control>`

SOFTKEY	该标签定义了软键的特性与反应。 **属性：** • **position** 软键的编号。1-8 水平软键， 9-16 垂直软键 以下属性生效于： • **type** 定义软键的属性。 user_controled - 软键的显示由脚本确定 toggle_softkey - 软键在交替按下时显示或不按时显示 • **refvar** 只在连接时使用类型 toggle_softkey 。 参考变量，将当前的软键属性复制到其中。 定义了类型为"String"的变量，含有属性“按下”、“不按”或“禁用”（参见标签 state）。 • **picture** 借助该属性，位图会按左对齐方式输出在软键上。必须给出完整的路径。 可显示的文本字符的数量受位图宽度的限制。

标签名称	含义
SOFTKEY 续	可以在软键程序块中确定下列其他操作： • **picturealignment** 通过该属性指定图片的校准。 图片默认位于软键的左侧。 可以设置下列值，用于校准： – **top** 上边沿 – **bottom** 下边沿 – **left** 左边沿 – **right** 右边沿 – **center** 居中 • **caption** 软键文本 • **state** 只在连接时使用类型 user_controlled 。 该标签向系统分配所需的软键显示。 **语法：** <state type="<state>" /> 可以指定以下字符串： – **notpressed** 软键在不按时显示。 – **pressed** 软键在按下时显示。 – **disabled** 软键禁用并显示为灰色。 • **navigation** • **update_controls** • **function**

标签名称	含义
SOFTKEY 续	**语法：** 标准软键： <state type="<softkey state>" /> <softkey position = "<1>"> … … </softkey> 或 受脚本控制的软键： <softkey position = "<1>" type="<user_defined>" > <state type="<softkey state>" /> … … </softkey> 或 切换软键： <softkey position = "<1>" type="<toggle_softkey>" refvar="<variable name>" > … … </softkey> **示例：** <let name="define_sk_type" type="string">PRESSED</let> <let name="sk_type">1</let> <softkey POSITION="1" type="user_controled" > <caption>Toggle%nSK</caption> <if> <condition>sk_type == 0 </condition> <then> <op> sk_type = 1 </op> <op> define_sk_type = _T"PRESSED" </op> </then> <else> <op> define_sk_type = _T"NOTPRESSED" </op> <op> sk_type = 0 </op> </else> </if> <state type="$$$define_sk_type" /> … … </softkey>

标签名称	含义
SOFTKEY 续	**示例：** 或 <let name="curr_softkey_state" type="string">PRESSED</let> </softkey> <softkey POSITION="3" type="toggle_softkey" refvar="curr_softkey_state"> <caption>Toggle%nSK</caption> … </softkey>  
SOFTKEY_OK	该标签定义软键“确认”的反应。  可以在软键程序块中确定下列其他操作： • **navigation** • **update_controls** • **function** **语法：** <SOFTKEY_OK> … … </SOFTKEY_OK>

标签名称	含义

SOFTKEY_CANCEL	该标签定义软键“取消”的反应。
	
	
	可以在软键程序块中确定下列其他操作：
	• **navigation**
	• **update_controls**
	• **function**
	语法：
	`<SOFTKEY_CANCEL>`
	…
	…
	`</SOFTKEY_CANCEL>`

SOFTKEY_BACK	该标签定义软键“返回”的反应。
	
	
	可以在软键程序块中确定下列其他操作：
	• **navigation**
	• **update_controls**
	• **function**
	语法：
	`<SOFTKEY_BACK>`
	…
	…
	`</SOFTKEY_BACK>`

标签名称	含义

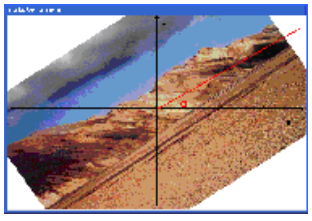
SOFTKEY_ACCEPT	该标签定义软键“接收”的反应。
	
	
	可以在软键程序块中确定下列其他操作：
	• **navigation**
	• **update_controls**
	• **function**
	语法：
	`<SOFTKEY_ACCEPT>`
	…
	…
	`</SOFTKEY_ACCEPT>`

TEXT	该标签用来在指定的位置上显示文本。 如果使用报警编号，则对话框会显示为该编号所保存的文本。 **语法：** `<TEXT xpos = "<X 坐标>" ypos = "<Y 坐标>"> Text </TEXT>` **属性：** • **xpos** 左上角的 X 坐标 • **ypos** 左上角的 Y 坐标 • **color** 文本颜色 **更多信息**参见章节“颜色代码 (页 70)” **值：** 待显示的文本

标签名称	含义

IMG	该标签用来在指定的位置上显示图像。支持 BMP 和 PNG 图像格式。
	
	语法：
	`<IMG xpos = "<X 坐标>" ypos = "<Y 坐标>" name = "<名称>" />`
	`<IMG … xrot="X 轴角度"yrot="Y 轴角度" zrot="Z 轴角度"/>`
	
	属性：
	• **xpos**
	左上角的 X 坐标
	• **ypos**
	左上角的 Y 坐标
	• **name**
	完整的路径名称。路径名称只能用小写字母。
	• **transparent**
	位图的透明色
	更多信息参见章节“颜色代码 (页 70)”

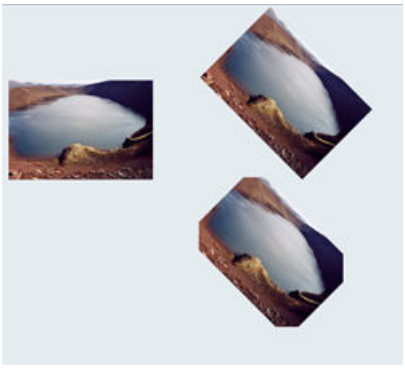
标签名称	含义

IMG 续	**示例：**
	图片绕 Z 轴旋转 34°。
	`<img xpos = "5" ypos = "23" name = "f:/appl/pic3.bmp" height="355" width="550" zrot="34" color_bk="#ffffff" ></img>`
	
	提示：
	驱动器名称根据系统而有所不同。
	
	
	可选：
	希望修改图片的原始大小时，可以使用属性 **width** 和 **height** 来确定尺寸。
	• **width**
	宽度，单位像素
	• **height**
	高度，单位像素
	
	示例：
	`<IMG xpos = "20" ypos = "40" name = "f:/appl/test.bmp" />`
	`<IMG xpos = "5" ypos = "23" name = "f:/appl/test.bmp" height = "355" width = "550"/>`

标签名称	含义

IMG 续	**属性：**
	AspectRatioMode
	此属性在非比例性的缩放中用于控制图形长宽比。
	
	值：
	• **Ignore**
	位图的长宽比按照给定的高度和宽度进行调整。
	• **Keep**（标准）
	长宽比保留并确定图片缩放为最大限度地填充给定高度和宽度的矩形。
	• **KeepByExpanding**
	长宽比保留并确定图片缩放为超出给定的高度和宽度最少的矩形。
	
	示例：
	`<img xpos = "8" ypos = "30" name = "test.bmp" height="155" width="150" AspectRatioMode="KeepbyExpanding">`
	`  <property transparent="#00ff00" />`
	`</img>`
	`<img xpos = "8" ypos = "200" name = "test.bmp" height="155" width="150" AspectRatioMode="Ignore">`
	`  <property transparent="#00ff00" />`
	`</img>`
	`<img xpos = "200" ypos = "200" name = "test.bmp" height="155" width="150" AspectRatioMode="Keep">`
	`  <property transparent="#00ff00" />`
	`</img>`
	
	• 上方 - 已设置 **KeepbyExpanding** 属性
	• 左下方 - 已设置 **Ignore** 属性
	• 右下方 - 已设置 **Keep** 属性

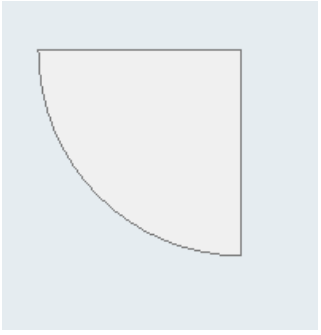
标签名称	含义

IMG 续	**属性：**
	ExpandingForRotation
	长宽比保留。图片缩放为与已旋转和缩放的图形的边框相对应的矩形。
	
	示例：
	`<img xpos = "8" ypos = "100" name = "pic1.bmp" height="155" width="150" >`
	`  <property transparent="#ffffff" />`
	`</img>`
	`<op>`
	`  zrot = 45.0;`
	`</op>`
	`<img xpos = "205" ypos = "23" name = "pic1.bmp" height="155" width="150" zrot="$zrot" AspectRatioMode="ExpandingForRotation">`
	`  <property transparent="#ffffff" />`
	`</img>`
	`<img xpos = "205" ypos = "200" name = "pic1.bmp" height="155" width="150" zrot="$zrot" >`
	`  <property transparent="#ffffff" />`
	`</img>`
	
	• 左侧 - 图片缩放为 150x155 像素
	• 右上方 - 图片缩放为 150x155 像素且用 **ExpandingForRotation** 属性旋转 45 度
	• 右下方 - 图片缩放为 150x155 像素且旋转 45 度

标签名称	含义
ARC	此标签用于在窗体中绘制圆弧。 **属性：** 以像素为单位指定在窗体中的位置： • **xpos1** - 圆弧起点的 X 坐标 • **ypos1** - 圆弧起点的 Y 坐标 • **xpos2** - 圆弧终点的 X 坐标 • **ypos2** - 圆弧终点的 Y 坐标 以像素为单位指定在窗体中的位置： • **ccx** - 圆弧圆心的 X 坐标 • **ccy** - 圆弧圆心的 Y 坐标 • **radius** - 圆弧半径，像素值 **提示：** **圆心或半径** 指定这两个参数时使用圆弧圆心。
ARC 续	• **penwidth** 以像素为单位设置线条的粗细。 • **color** 线条颜色 • **color_bk** 圆弧的填充色 **更多信息**参见章节“颜色代码 (页 70)” • **style** 设置线条样式： – **"SolidLine"** - 实线（默认设置） – **"DashLine"** - 虚线 – **"DotLine"** - 点线 – **"DashDotLine"** - 点划线 – **"DashDotDotLine"** - 双点划线

标签名称	含义
ARC 续	• **type** – **cw** - 顺时针 – **ccw** - 逆时针 • **arrow** 在圆弧两端显示箭头： – **"P1"** - 箭头位于线条起点 – **"P2"** - 箭头位于线条起点 – **"P1P2"** - 箭头分别位于线条起点和终点 – **"P1_FILLED"** - 实心箭头位于线条起点 – **"P2_FILLED"** - 实心箭头位于线条起点 – **"P1P2_FILLED"** - 实心箭头分别位于线条起点和终点 • **arrow_size** 如果使用属性 arrow，则可以像素为单位指定箭头的长度。

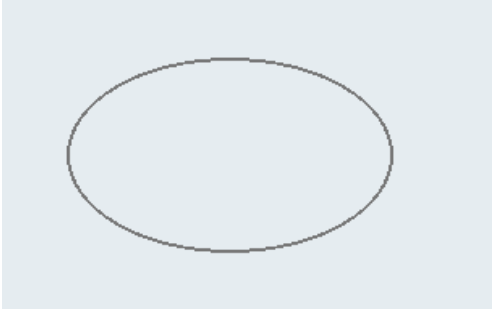
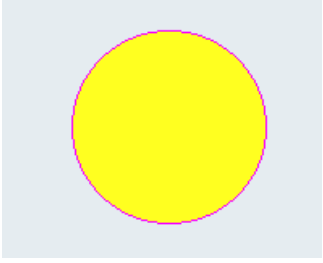
标签名称	含义

ARC 续	**示例：**
	
	`<arc xpos1="20" ypos1="200" xpos2="180" ypos2="200" ccx="100" ccy="200" type="ccw" PENWIDTH="1" color="#777777" arrow="p1p2_filled"/>`
	
	或
	
	`<arc xpos1="20" ypos1="200" xpos2="180" ypos2="200" radius="80" type="ccw" PENWIDTH="1" color="#777777" arrow="p1p2_filled"/>`
	
	
	`<arc xpos1="20" ypos1="200" xpos2="100" ypos2="280" radius="80" type="ccw" PENWIDTH="1" color="#777777" color_bk="#f0f0f0"/>`

标签名称	含义
BOX	该标签可以在指定的位置以指定的颜色绘制一个填满的矩形。 **语法：** `<BOX xpos = "<X 坐标>" ypos = "<Y 坐标>" width = "<X 长度>" height = "<Y 长度>" color = "<颜色代码>" />` **属性：** • **xpos** 左上角的 X 坐标 • **ypos** 左上角的 Y 坐标 • **width** X 方向上的宽度（以像素为单位） • **height** Y 方向上的高度（以像素为单位） • **color** 颜色代码 **更多信息**参见章节“颜色代码 (页 70)”

标签名称	含义
ELLIPSE	此标签用于在窗体中绘制椭圆。 **属性：** 以像素为单位指定在窗体中的位置： • **xpos** - 线条起点的 X 坐标 • **ypos** - 线条起点的 Y 坐标 • **width** - 外切矩形的宽度 (bounding box) • **height** - 外切矩形的高度 (bounding box) • **penwidth** 以像素为单位设置线条的粗细。 • **color** 线条颜色 • **color_bk** 椭圆的填充色 **更多信息**参见章节“颜色代码 (页 70)” • **style** 设置线条样式： – **"SolidLine"** - 实线（默认设置） – **"DashLine"** - 虚线 – **"DotLine"** - 点线 – **"DashDotLine"** - 点划线 – **"DashDotDotLine"** - 双点划线

标签名称	含义

ELLIPSE 续	**示例：**
	
	`<ellipse xpos="302" ypos="160" width="100" height="60" PENWIDTH="2" color="#777777" />`
	
	`<ellipse xpos="402" ypos="360" width="60" height="60" PENWIDTH="1" color="#f800ff"  color_bk="#ffff20"/>`

标签名称	含义

FUNCTION	**函数调用**
	该标签可以执行属性“**name**”中定义的
	函数体。
	
	属性：
	• **name**=“函数体的名称”
	• **return** =“用于储存函数结果的变量名称”
	
	值：
	需要将其传输到函数体的变量列表。变量相互之间以逗号分隔。最多可以传输 10 个参数。
	此外，还可以将常数或文本表达式指定为调用参数。为了标记文本表达式需要在文本前放置标志 _T 。
	
	语法：
	`<FUNCTION name = "<function name>" />`
	待调用的函数等待一个返回值
	`<FUNCTION name = "<function name>" return = "<Variablenname>" />`
	参数传递
	`<FUNCTION name = "<function name>"> var1, var2, var3 </FUNCTION>`
	`<FUNCTION name = "<function name>"> _T"Text", 1.0, 1 </FUNCTION>`
	
	示例：
	参见“FUNCTION_BODY”

标签名称	含义

FUNCTION_BODY	**函数体**
	该标签包含有一个子函数的函数体。要在 DialogGui 标签中对函数体进行编程。
	
	属性：
	• **name**= “函数体的名称”
	• **parameter** = “参数列表” （可选）
	属性列出了需要的传输参数。参数相互之间以逗号分隔。
	调用函数体时，函数调用中给定参数的值会复制到所列出的传递参数中。
	• **return** = “true” （可选）
	如果属性设为 true，则创建局部变量 $return。将函数的返回值复制到其中，在退出该函数时该值会继续传输给主调函数。
	
	语法：
	
	无参数的函数体
	`<FUNCTION_BODY name = "<function name>">`
	…
	…
	…
	`</ FUNCTION_BODY>`
	
	带参数的函数体
	`<FUNCTION_BODY name = "<function_name>" parameter = "<p1, p2, p3>">`
	…
	`<LET name = "tmp></LET>`
	`<OP> tmp = p1 </OP>`
	…
	`</FUNCTION_BODY>`
	
	带返回值的函数体
	`<FUNCTION_BODY name = "<function_name>" parameter = "<p1, p2, p3>" return = "true">`
	…
	`<LET name = "tmp></LET>`
	`<OP> tmp = p1 </OP>`
	…
	`<OP> $return = tmp </OP>`
	`</FUNCTION_BODY>`

标签名称	含义

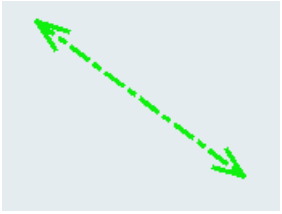
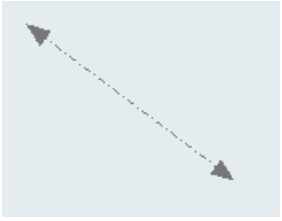
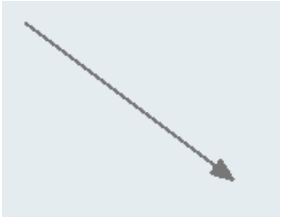
FUNCTION_BODY 续	**示例：**
	`<function_body name = "test" parameter = "c1,c2,c3" return = "true">`
	`<LET name = "tmp">0</LET>`
	`<OP> tmp = c1+c2+c3 </OP>`
	`<OP> $return = tmp </OP>`
	`</function_body>`
	…
	…
	…
	`<LET name = "my_var"> 4 </LET>`
	`<function name = "test" return = " my_var "> 2, 3,4</function>`
	`<print text = "result = %d"> my_var </print>`
	…
	…

LINE	此标签用于在窗体中绘制直线。 **属性：** 以像素为单位指定在窗体中的位置： • **xpos1** - 线条起点的 X 坐标 • **ypos1** - 线条起点的 Y 坐标 • **xpos2** - 线条终点的 X 坐标 • **ypos2** - 线条终点的 Y 坐标 • **penwidth** 以像素为单位设置线条的粗细。 • **color** 线条颜色 **更多信息**参见章节“颜色代码 (页 70)” • **style** 设置线条样式： – **"SolidLine"** - 实线（默认设置） – **"DashLine"** - 虚线 – **"DotLine"** - 点线 – **"DashDotLine"** - 点划线 – **"DashDotDotLine"** - 双点划线

标签名称	含义

LINE 续	• **arrow**
	在直线两端显示箭头：
	– **"P1"** - 箭头位于线条起点
	– **"P2"** - 箭头位于线条起点
	– **"P1P2"** - 箭头分别位于线条起点和终点
	– **"P1_FILLED"** - 实心箭头位于线条起点
	– **"P2_FILLED"** - 实心箭头位于线条起点
	– **"P1P2_FILLED"** - 实心箭头分别位于线条起点和终点
	• **arrow_size**
	如果使用属性 arrow，则可以像素为单位指定箭头的长度。
	
	语法：
	`<line xpos1="<起点 X 坐标>" ypos1="<起点 Y 坐标>" xpos2="<终点 X 坐标>" ypos2="<终点 Y 坐标>"`
	`PENWIDTH="<线条宽度>" color="<线条颜色>" style="<线条样式>" arrow="<箭头样式>" arrow_size="<箭头大小>"/>`

标签名称	含义

LINE 续	**示例：**
	`<line xpos1="20" ypos1="40" xpos2="100" ypos2="100" PENWIDTH="3" color="#0ff00f" style="DashDotLine" arrow="p1p2" arrow_size="12"/>`
	
	`<line xpos1="20" ypos1="40" xpos2="100" ypos2="100" PENWIDTH="1" color="#777777" style="DashDotLine" arrow="p1p2_filled" arrow_size="9"/>`
	
	`<line xpos1="20" ypos1="40" xpos2="100" ypos2="100" PENWIDTH="2" color="#777777" arrow="p2_filled" arrow_size="9"/>`
	

标签名称	含义

REQUEST	借助该标签可以在周期性读取服务（hotlink 热连接）中使用一个变量。这样可以缩短对未与控件相连的变量的存取时间。
	如果在值变化时自动调用一个函数，则应将函数的名称作为其它属性加以定义。
	该标签仅在 INIT 初始化指令中进行处理。
	
	属性：
	• **name**
	地址名称
	• **function**
	函数名称
	
	语法：
	`<REQUEST name = "<NC-Variable>" />`
	或
	`<REQUEST name = "<NC-Variable>" function="<function name>"/>`
	
	示例：
	`<request name ="plc/mb10" />`
	
	或
	
	`<function_body name="my_function" >`
	`<print text="value changed" />`
	`</function_body>`
	…
	…
	…
	`<request name ="plc/mb10" function="my_function"/>`

标签名称	含义

RECALL	如要通过 Recall 键执行导航，则可在菜单中使用该标签。如对该标签编程，则会显示 Recall 符号并允许通过解析器对 Recall 键进行处理。
	
	语法：
	`<recall>`
	…
	…
	`</recall>`

UPDATE_CONTROLS	该标签用于比较操作单元和参考变量。
	
	属性：
	• **type**
	该属性可以确定数据比较的方向。
	= TRUE – 从参考变量读取数据并复制到操作单元中。
	= FALSE – 从操作单元读取数据并复制到参考变量中。
	
	语法：
	`<UPDATE_CONTROLS type = "<方向>"/>`
	
	示例：
	`<SOFTKEY_OK>`
	`< UPDATE_CONTROLS type="false"/>`
	`</SOFTKEY_OK>`

2.8 创建用户菜单

2.8.1 创建加工循环窗口

循环支持功能可以通过格式对话框自动创建和反编译循环调用。

提供以下标签实施该功能：

- **NC_INSTRUCTION**
- **CREATE_CYCLE**

为了标记循环窗口，应在标签 **FORM** 中将属性 **type** 的值定义为 **cycle**。该标记可实现指令 **NC_INSTRUCTION** 的执行。

示例

```
<FORM name = "cycle100_form" type= "CYCLE">
…
…
</FORM>
```

标签 **NC_INSTRUCTION** 含有要执行的循环调用。所有循环参数都通过占位符进行了预留。

示例

```
<FORM name = "cycle100_form" type= "CYCLE">
<NC_INSTRUCTION refvar= "cyc_string" >Cycle100 ($p1, $p2, $p3)</
NC_INSTRUCTION>
…
…
…
</FORM
```

标签 **CREATE_CYCLE** 提供保存在占位符变量中的值并生成 NC 指令。

然后复制到指定的变量中。

标签名称	含义
NC_INSTRUCTION	使用该标签可定义要创建的 NC 指令。 列出的所有循环参数都会自动作为 **FORM** 字符串变量被创建并可用于 **FORM**。 前提条件：FORM 的属性 **type** 设置为 **CYCLE**。 **属性：** • **refvar** 如果向标签分配了一个参考变量，则所有参数都会预设为参考变量中保存的 NC 程序段中的值。 **句法：** <NC_INSTRUCTION> 带占位符的 NC 指令</ NC_INSTRUCTION> **示例：** <let name="cyc_string" type="string"> Cycle100(0, 1000, 5)</ let> … … … <FORM name = "cycle100_form" type= "CYCLE"> <NC_INSTRUCTION refvar= "cyc_string" >Cycle100($p1, $p2, $p3)</ NC_INSTRUCTION> … … … </FORM>

标签名称	含义

CREATE_CYCLE	该标签会生成一个 NC 程序段，其句法由标签 **NC_INSTRUCTION** 确定。
	在生成 NC 指令前，分析程序会调用标签 FORM 的 CYCLE_CREATE_EVENT。该标签可用于计算循环参数。
	句法：
	`<CREATE_CYCLE/>`
	
	选项：
	指定了参考变量时，指令会在该变量中复制创建的调用。
	`<create_cycle refvar="name" />`
	
	属性：
	• **refvar**
	如果向该标签分配了一个参考变量，则 NC 指令会复制到该变量中。
	
	示例：
	`<LET name="cyc_string" type="string"> Cycle100(0, 1000, 5)</LET>`
	
	`<SOFTKEY_OK>`
	`<caption>OK</caption>`
	`<CREATE_CYCLE />`
	`<close_form />`
	`<navigation>main_menu</navigation>`
	`</SOFTKEY_OK>`
	
	或
	
	`<SOFTKEY_OK>`
	`<caption>OK</caption>`
	`<CREATE_CYCLE refvar= "cyc_string" />`
	`<close_form />`
	`<navigation>main_menu</navigation>`
	`</SOFTKEY_OK>`

2.8.2 替代符号

系统可以确定有关运行时间的控制特性（属性值）。为了能够使用该功能，需要在一个局部变量中提供所需的特性并使用前置的 **$ 符号** 将变量名称作为属性值传输给标签。

如果标签要以字符串作为属性值或值，则应在变量名称前加上符号 $$$。

示例：

```
<let name="my_ypos">100</let>
<let name="field_name" type="string"></let>

<control name = "edit1" xpos = "322" ypos = "$my_ypos"
refvar="nck/Channel/Parameter/R[1]" />

<op>my_ypos = my_ypos +20 </op>

<control name = "edit2" xpos = "322" ypos = "$my_ypos"
refvar="nck/Channel/Parameter/R[2]" />

<print name ="field_name" text="edit%d">3</print>
<op>my_ypos = my_ypos +20 </op>

<control name = "$field_name" xpos = "322" ypos = "$my_ypos"
refvar="nck/Channel/Parameter/R3]" />

<caption>$$$field_name</caption>
```

2.9 创建 struct_def.xml 文件

步骤

在一些函数中会使用文件 **struct_def.xml** 进行类型定义。如果缺少此 XML 文件，则可以使用下面的示例脚本创建一个文件副本并将其集成到函数项目中。

说明

以 UTF8 格式创建 XML 文件。

脚本

struct_def.xml

```
<!--
Copyright 2015 by Siemens AG and Wolfram Kuhnert
(wolfram.kuhnert@siemens.com)
Permission to use, copy, modify, distribute, and sell this softwareand its
documentation for any purpose is hereby granted without fee,provided that
the above copyright notice appear in all copies and that both that copyright
notice and this permission notice appearin supporting documentation, and
that the names of Siemens AG and Wolfram Kuhnertnot be used in advertising
or publicity pertaining todistribution of the software without specific,
written priorpermission.
Siemens AG and Wolfram Kuhnert make no representations about the
suitability of this software for any purpose. It isprovided "as is" without
express or implied warranty.
Required software version: Operate 4.7 Sp1 HF1
-->

<!--
    typedef struct tagRECT {
     LONG left;
     LONG top;
     LONG right;
     LONG bottom;
    } RECT;
-->

<typedef name="StructRect" type="struct" >
  <element name="left" type="int">0</element>
  <element name="top" type="int">0</element>
  <element name="right" type="int">0</element>
  <element name="bottom" type="int">0</element>
</typedef>

<typedef name="StructPoint" type="struct" >
  <element name="x" type="int">0</element>
  <element name="y" type="int">0</element>
</typedef>

<typedef name="StructSize" type="struct" >
  <element name="width" type="int">0</element>
  <element name="height" type="int">0</element>
</typedef>

<typedef name="StructMDLimits" type="struct" >
  <element name="lowerLimit" type="double">0</element>
  <element name="upperLimit" type="double">0</element>
  <element name="arrayDim1" >0</element>
  <element name="arrayDim2" >0</element>
</typedef>
```

2.10 元素定址

为了给 NC 变量、PLC 组件或驱动数据分配地址，地址名称由所需要的数据构成。 一个地址由部分路径 **组件名称** 和 **变量地址**组成。 使用斜线作为分隔符。

2.10.1 PLC 定址

PLC 地址分配从路径 **plc** 开始。

表格 2-3　允许使用下列地址：

DBx.DB(f)	数据块
I(f)x	输入端
Q(f)x	输出端
M(f)x	标志
V(f)x	变量

DBx.DBXx.b	数据块
Ix.b	输入端
Qx.b	输出端
Mx.b	标志
Vx.b	变量

表格 2-4　数据格式 **f**：

B	字节
W	字
D	双字

在位地址分配时取消数据格式标识。

地址 **x**：

有效的 S7 200 地址名称

位地址分配：

b – 位号

示例：

```
<data name = "plc/mb170">1</data>
<data name = "i0.1"> 1 </data>
<op> "m19.2" = 1 </op>
refvar="plc/db9062.dbb0:string"
```

2.10.2 NC 变量定址

NC 变量的定址从路径 **nck** 开始。

在该部分之后跟有数据地址，其结构可以查阅 NC 变量和接口信号的参数手册。

示例：

```
<LET name = "tempStatus"></LET>
<OP> tempStatus ="nck/channel/state/chanstatus" </OP>
```

2.10.3 通道专用定址

如果未在地址记号中给定通道编号，则会始终访问操作软件的通道 1。

如果需要读取某一特定通道的数据，则可在地址中添加标识 **u** (Unit) 以及所需通道号。

示例：

```
nck/Channel/MachineAxis/actFeedRate[3]
nck/Channel/MachineAxis/actFeedRate[u1, 3]
```

2.10.4 在运行时生成 NC/PLC 地址

支持在运行时生成地址名称。

此时可将字符串变量的内容作为地址用在操作指令以及功能 nc.cap.read 和 nc.cap.write 中。

对于该定址模式应注意以下事项：

- 变量名称写在引号内。
- 在变量名称前标记三个 ‚$' 符号。

句法：

```
"$$$variable name"
```

示例：

```
<PRINT name="var_adr" text="DB9000.DBW%d"> 2000</PRINT>
<OP> "$$$var_adr" = 1 </OP>
```

2.10.5 驱动组件的定址

驱动组件的定址从路径 **drive** 开始。

后面跟有驱动设备的详细规格：

CU – Control Unit（控制单元）

DC – Drive Control（电机模块）

CULNK – 扩展模块 (HUBs)

TM – Terminal-Module（端子模块）

LM – Line-Module（电源模块）

该部分要添加上待设置参数。

示例：

```
<LET name="r0002_content"></LET>
<LET name="p107_content"></LET>

<!– 读取 CU 上 r0002 的值 ->

<OP> r0002_content = "drive/cu/r0002" </OP>
<OP> r0002_content = "drive/cu/r0002[CU1]" </OP>

<!– 读取 NX1 上 r0002 的值 ->
<OP> r0002_content = "drive/cu/r0002[CU2]" </OP>

<!– 读取 CU 上 p107[0] 的值 ->
<OP> p107_content = "drive/cu/p107[0]" </OP>
<PRINT text="%d"> p107_content </PRINT>

<!– 读取 CU 上 p107[0] 的值 ->

<OP> p107_content = "drive/cu/p107[0, CU1]" </OP>
<PRINT text="%d"> p107_content </PRINT>

<!– 读取 NX1 上 p107[0] 的值 ->

<OP> p107_content = "drive/cu/p107[0, CU2]" </OP>
<PRINT text="%d"> p107_content </PRINT>
```

驱动对象的地址分配：

为了给单个对象分配地址，要在参数之后的方括号中给出所需的对象。

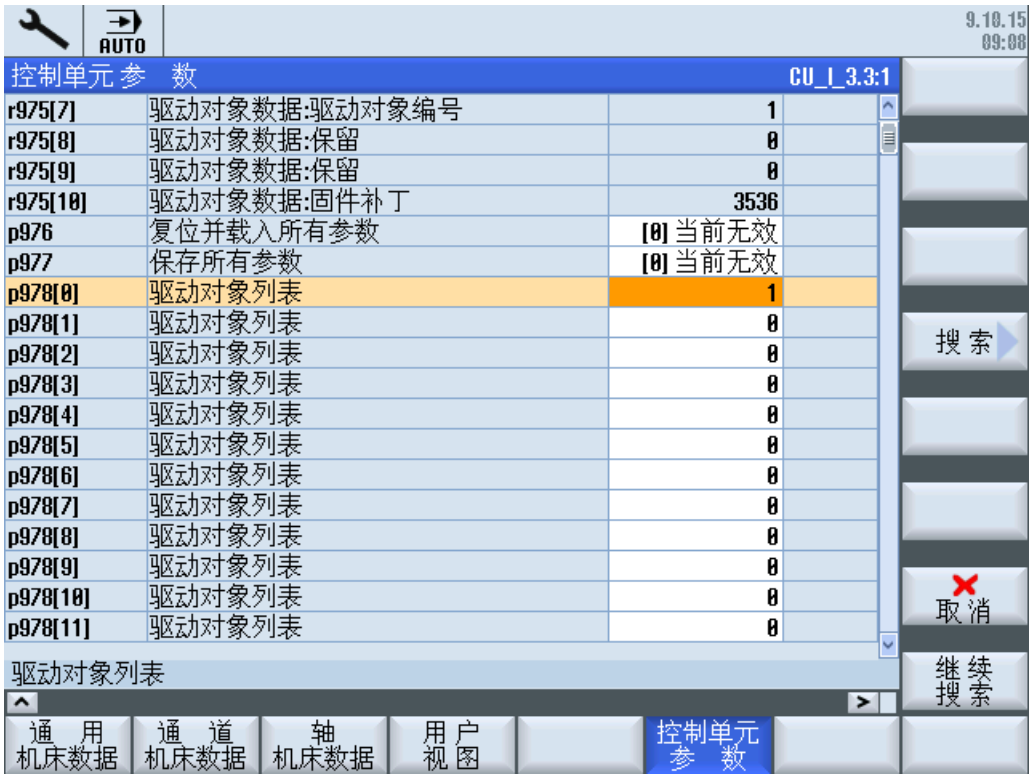


图 2-7 控制系统上的驱动参数 p0978 [...]

参数号[do<DO-index>]

示例：

```
p0092[do1]
```

可以选择利用“替代符号”**$<变量名称>** 从一个局部变量中读取驱动下标。
z.B. DO$局部变量

示例：

```
<DATA name ="drive/cu/p0092">1</DATA>
<DATA name ="drive/dc/p0092[do1] ">1</DATA>
```

间接地址分配：

```
<LET name = "driveIndex> 0 </LET>
<OP> driveIndex = $ctrlout_module_nr[0, AX1] </OP>
<DATA name ="drive/dc[do$driveIndex]/p0092">1</DATA>
```

2.10.6 示例：测定电机模块的 DO 号

11 型电机模块的 DO 号可通过以下方式测定：

所有已连接的驱动对象都根据其接口编号列在相应 CU 的数组 p978 中。同时会在数组 p101 中列出组件编号并在数组 p107 中列出组件类型。

为下列组件类型分别使用一个自己的下标：

CU – Control Units（控制单元）

DC – Drive Controls （电机模块）

CULNK – 扩展模块 (HUBs)

TM – Terminal-Module（端子模块）

LM – Line-Module（电源模块）

定址下标按这样的方式测定，将每个所连接的 CU 的数组 p0107 按升序通过，根据所找到类型的出现次序将类型下标增加一。基准值为一。如果在该数组中找到了 NX 组件，则在当前数组完全通过后，才会对 NX 组件继续进行计数。对 NX 组件和 CU 组件的处理按照发现的顺序进行。

下标测定：CUI 与 NX

CUI		NX1		定址下标	
				DO	CU
p107[0]	[3]SINAMICS				1
p107[2]	[11]SERVO			1	
p107[3]	[11]SERVO			2	
p107[4]	[11]SERVO			3	
p107[5]	[254]CU-LINK				2
p107[6]	[11]SERVO			4	
		p107[1]	[11]SERVO	5	
		p107[2]	[11]SERVO	6	
		p107[3]	[11]SERVO	7	

在该拓扑中含有七个电机模块。下标一至四指向分配给 CU 的电机模块。下标五至七指向 NX 的电机模块。访问 CU 使用下标一。NX 使用下标二来定址。

脚本示例：xmldial.xml 和 drv_sys_hlpfunct.xml

xmldial.xml

```
<DialogGui>

 <?include src="f:\appl\drv_sys_hlpfunct.xml" ?>

 <menu name = "main">
  <open_form name = "main_form" />
  <softkey POSITION="1">
   <caption></caption>
   <navigation>main</navigation>
  </softkey>
 </menu>

 <form name="main_form">
  <init>
   <caption>Component arrangement</caption>
    <let name="count" />
    <let name="str" type="string" />
    <let name="do_name" type="string" />
    <let name="cu_name" type="string" />
    <let name="cui_idx" />

    <op>
     cui_idx = 0;
    </op>

    <function name="load_component" />
    <print text="%d CUs found">num_cus</print>

    <function name="calculate_do_index" />

    <control name="list_comp_no_do_idx" xpos="8" ypos="80"
fieldtype="listbox" width="360" height="500" >
     <property item_data="100" />
    </control>

    <op>
     count = 0;
    </op>

    <while>
    <condition>count &lt; 32 && address_idx_map[$count].comp_no != 0</
condition>

    <op>
     do_name= _T"";
    </op>

    <function name="read_do_name_fast" return="do_name">count</function>
```

xmldial.xml

```
    <function name="read_cu_name_fast"
return="cu_name">address_idx_map[$count].cu_idx</function>

    <print name="str" text="%3d %3d %3d %s
%s">address_idx_map[$count].cu_idx, address_idx_map[$count].comp_no,
address_idx_map[$count].do_idx, do_name, cu_name</print>
    <function name="control.additem">_T"list_comp_no_do_idx", str, count</
function>

    <op>
     count = count +1;
    </op>
    </while>

   <paint>
    <text xpos="8" ypos="30">Motor moduls</text>
    <text xpos="8" ypos="60">CU</text>
    <text xpos="40" ypos="60">Comp.</text>
    <text xpos="92" ypos="60">DO Index</text>
    <text xpos="172" ypos="60">Name</text>
  </paint>

 </form>
</DialogGui>
```

drv_sys_hlpfunct.xml

```
<typedef name="components" type="struct">
 <element name="cu_p978" dim="25" />
 <element name="do_p0101" dim="25" />
 <element name="do_p0107" dim="25" />
</typedef>

<typedef name="comp_no_do_idx_map" type="struct">
 <!-- cu index -->
 <element name="cu_idx" />
 <!-- component number -->
 <element name="comp_no" />
 <!-- address index -->
 <element name="do_idx" />
</typedef>

<let name="componentsList" dim="5" type="components" />
<let name="address_idx_map" dim="32" type="comp_no_do_idx_map" />
<let name="num_cus" />
<let name="_drv_sys_comp_array_size">23</let>

<!-- ------------------------------------------------------------

 function: load_components_description
 This function loads the parameter arrays p978, p101 and p107 into a local
memory

 input:
 cuno: CU index 0 based (index 0 == CUI)

 output:
 componentsList structure

 ------------------------------------------------------------ -->

<function_body name="load_components_description" parameter="cuno">
 <let name="count" />
 <let name="next_nx" >0</let>
 <let name="cuidx"></let>
 <let name="error" />

 <op>
  count = _drv_sys_comp_array_size;
  next_nx = cuno;
  cuidx = cuno+1;
 </op>

 <print text="gather data" />
```

drv_sys_hlpfunct.xml

```
 <function name="ncfunc.cap.read" return="error"
rows="$count">componentsList[$cuno].cu_p978, "drive/cu/p0978[0, cu
$cuidx]"</function>
 <function name="ncfunc.cap.read" return="error"
rows="$count">componentsList[$cuno].do_p0101, "drive/cu/p0101[0, cu
$cuidx]"</function>
 <function name="ncfunc.cap.read" return="error"
rows="$count">componentsList[$cuno].do_p0107, "drive/cu/p0107[0, cu
$cuidx]"</function>

 <print text="gather data finished" />
 <sleep value="20" />

 <op>
  count = 0;
 </op>

 <while>
  <condition>count < _drv_sys_comp_array_size</condition>
  <if>
   <condition>componentsList[$cuno].cu_p978[$count] == 60</condition>
  <then>
   <op>
    next_nx= next_nx +1;
   </op>
  <print text="next nx %d">next_nx</print>
  <function name="load_components_description">next_nx</function>
  </then>
  </if>

 <op>
  count = count+1;
 </op>
 </while>
 <op>
  num_cus = next_nx+1;
 </op>
</function_body>

<!-- ------------------------------------------------------------

 function: calculate_do_index
 This function is looking for components of type 11 (SERVO) and lists these
in the componentsList array.

 input:
 -

 output:
 componentsList array filled
```

drv_sys_hlpfunct.xml

```
 ------------------------------------------------------------------ -->

<function_body name="calculate_do_index">
 <let name="cuno" />
 <let name="count" />
 <let name="do_index" >1</let>
 <let name="map_index" />
 <while>
  <condition>
   cuno < num_cus</condition>
   <op>
    count = 0;
   </op>
  <while>
   <condition>count < _drv_sys_comp_array_size</condition>
   <if>
    <condition> componentsList[$cuno].do_p0107[$count] == 11</condition>
    <then>
     <op>
      address_idx_map[$map_index].cu_idx = cuno+1;
      address_idx_map[$map_index].comp_no =
componentsList[$cuno].do_p0101[$count];
      address_idx_map[$map_index].do_idx = do_index;
      do_index = do_index+1;
      map_index = map_index +1;
     </op>
    </then>
   </if>
   <op>
    count = count +1;
   </op>
  </while>
  <op>
   cuno = cuno +1;
  </op>
 </while>
</function_body>
```

2.10.7 机床数据和设定数据的定址

可以使用 $ 符号标记驱动和设定数据，后面跟有数据名称。

机床数据：

$Mx_<名称[**索引, AX<轴编号>**]>

设定数据：

$Sx_<名称[索引, AX<轴编号>]>

x：

N – 通用的机床或设定数据

C – 通道专用的机床或设定数据

A – 轴专用的机床或设定数据

索引：

参数在该数组中给出数据索引。

AX<轴编号>：

在轴专用数据时对所需要的轴（**<轴编号>**）进行详细说明。

可以选择利用“替代符号”**$<变量名称>** 从一个局部变量中读取轴索引。

例如 AX$ 局部变量

示例：

```
<DATA name ="$MN_AXCONF_MACHAX_NAME_TAB[0] ">X1</DATA>
```

轴的直接地址分配：

```
<DATA name ="$MA_CTRLOUT_MODULE_NR[0, AX1] ">1</DATA>
…
…
```

轴的间接地址分配：

```
<LET name ="axisIndex"> 1 </LET>
<DATA name ="$MA_CTRLOUT_MODULE_NR[0, AX$axisIndex] ">1</DATA>
```

2.10.8 通道专用机床数据

如果未在地址记号中给定通道编号，则会始终访问当前设置的操作软件通道。

如果需要读取某一特定通道的数据，则可在地址中使用方括号添加标识 **u** (Unit) 以及所需通道号。使用数组定址时，通道数据是方括号中的最后一个自变量。

示例：

```
$MC_RESET_MODE_MASK
```

或者

```
$MC_AXCONF_GEOAX_ASSIGN_TAB[0]
```

```
$MC_RESET_MODE_MASK[u1]
```

或者

```
$MC_AXCONF_GEOAX_ASSIGN_TAB[0, u1]
$MC_RESET_MODE_MASK
```

2.10.9 用户数据的定址

用户数据的地址表示以路径 **gud** 开头，后面跟有 GUD 名称。

对于数组型的数据而言，还需要在 GUD 名称后的方括号中指定数组序号。

示例：

```
<DATA name ="gud/syg_rm[0]"
<OP>"gud/syg_rm[0]" = 10 </op>
```

全局用户数据的地址表达方式

地址从路径 gud 开头，后面跟着 CHANNEL，该部分后面跟着具体的 GUD 范围：

GUD 范围	分配
sgud	西门子的 GUD
mgud	机床制造商的 GUD
ugud	用户的 GUD

GUD 范围后面跟着 GUD 名称。对于数组型数据而言，还需要在 GUD 名称后面的的方括号中指定数组序号。

示例：

```
<data name =“gud/channel/mgud/syg_rm[0]”>1</data>
<op>“gud/channel/mgud/syg_rm[0]” = 5*2 </op>
```

多维数组的地址表示

在表示多维数组的地址时，行号后面要跟上列号。两个编号用一个点隔开。

对于通道专用的 GUD 而言：

- 通道号的表示方式为：字母 **u**（表示 unit）后加数字，前面或后面加上行号/列号。
- 参数之间以逗号分隔。
- 没有指定通道号时，系统访问第一条通道。

示例：

```
"gud/Channel/sgud/_WP[u2, 2.0]"
```

或者

```
"gud/Channel/sgud/_WP[2.0, u2]"
```

2.11 预定义功能

脚本语言提供有各种字符串运算和标准数学函数。下文列出的函数名称是预留的并且不能改写。

函数名称	含义

Ncfunc cap read	该函数会将指定地址中的值复制到局部变量中。如果读取成功，返回变量中的值为零。
	
	与运算指令相反，该函数在出错时不会终止脚本指令的执行。
	
	属性：
	• **return** - 执行状态
	– 值 = 0 - 正常
	– 值 = 1 - 变量的读取无法进行
	• **rows** - 数组中额外读取的行数（可选）
	如果给定了数组下标的参考变量，该函数可将读取的值从该下标复制到目标变量中。
	
	语法：
	`<function name="ncfunc.cap.read" return="error">`
	`lokale variable, "address"</function>`
	
	示例：
	`<let name="error"></let>`
	`<function name="ncfunc.cap.read" return="error"> 3,`
	`"drive/cu/p0009"</function>`
	`<if>`
	`<condition>error != 0</condition>`
	`<then>`
	`<break />`
	`</then>`
	`</if>`
	
	或
	
	`<let name="cu_p978" dim="25" ></let>`
	`...`
	`...`
	`<function name="ncfunc.cap.read" return="error"`
	`rows="23">cu_p978, "drive/cu/p0978[0, 1]"`
	`</function>`

函数名称	含义

Ncfunc cap write	该函数会向指定变量写入一个值。如果写入成功，返回变量中的值为零。
	
	与运算指令相反，该函数在出错时不会终止脚本指令的执行。
	
	属性：
	• **return** - 执行状态
	– 值 = 0 - 正常
	– 值 = 1 - 变量的读取无法进行
	• **rows** - 数组中额外写入的行数（可选）
	如果给定了数组下标的参考变量，该函数可将值从该下标复制到目标变量中。
	
	语法：
	`<function name="ncfunc.cap.read" return="error">`
	`local variable or constant, "address"</function>`
	
	示例：
	`<let name="error"></let>`
	`<function name="ncfunc.cap.write" return="error">`
	`0, "drive/cu/p0009"</function>`
	`<if>`
	`<condition>error != 0</condition>`
	`<then>`
	`<break />`
	`</then>`
	`</if>`
	
	或
	
	`<let name="cu_p978" dim="25" ></let>`
	`...`
	`...`
	`<function name="ncfunc.cap.write" return="error"`
	`rows="23">cu_p978, "drive/cu/p0978[0, 1]"`
	`</function>`

函数名称	含义
Ncfunc PI-Service	通过程序实例（PI）服务可将任务传输给 NCK。 如果服务执行成功，该函数会将值 1 返回到返回变量中。 **刀具表的操作：** _N_CREATO - 创建刀具 _N_DELETO - 删除刀具 _N_CREACE - 创建刀沿 _N_DELECE - 删除刀沿 **零点偏移的激活：** _N_SETUFR - 将当前的用户框架激活 _N_SETUDT - 将当前的用户数据激活 **程序段查找：** _N_FINDBL - 激活查找 _N_FINDAB - 取消程序段查找 **机床数据的重新配置：** _N_CONFIG 机床数据和 NEW_CONF 生效级（后续由操作员输入的）会被一同激活。 **语法：** `<function name="ncfunc.pi_service" return="return var"> pi name, var1, var2, var3, var4, var5 </function>` **属性：** name - 函数名称 return - 保存执行结果的变量名称： • 值 == 1 – 成功执行任务 • 值 == 0 – 任务失败 **标签值：**

函数名称	含义
	pi name - PI 服务的名称（字符串） var1 到 var5 - PI 专用自变量

函数名称	含义
Ncfunc PI-Service 续	**自变量：** • _N_CREATO var1 - 刀具号 • _N_DELETO var1 - 刀具号 • _N_CREACE var1 - 刀具号 var2 - 刀沿号 • _N_DELECE var1 - 刀具号 var2 - 刀沿号 • _N_SETUFR 无自变量 • _N_SETUDT var1 - 待激活的用户数据区域 – 1 - 刀具补偿数据 – 2 - 有效基本框架 – 3 - 有效可设置框架 • _N_FINDBL var1 - 查找模式 – 2 - 带轮廓计算的查找 – 4 - 查找到程序段末尾 – 1 - 不带计算的查找 • _N_FINDAB 无自变量 • _N_CONFIG

函数名称	含义

	无自变量
	
	示例：
	创建刀具 - 刀具号 3
	
	`<function name="ncfunc.pi_service">_T"_N_CREATO", 3</function>`
	
	删除刀具 5 的刀沿 1
	
	`<function name="ncfunc.pi_service">_T"_N_DELECE", 5, 1 </function>`

ncfunc chan PI-Service	该函数会执行基于通道的 PI 服务。在 PI 服务名称后传输通道编号。然后是所有其他调用参数。
	
	参数：
	channel - 通道编号
	
	语法：
	`<function name="ncfunc.chan_pi_service" return="error"> _T"_N_SETUFR", channel, …</function>`
	
	示例：
	`<let name="chan" >1</let>`
	`<function name="ncfunc.chan_pi_service" return="error"> _T"_N_SETUFR", chan</function>`
	
	`<function name="ncfunc.chan_pi_service" return="error"> _T"_N_SETUDT", chan, _T"016", _T"00000", _T"00000"</function>`

函数名称	含义

Ncfunc display resolution	该函数提供由控制系统确定的浮点数转换规则。提供了字符串变量。
	
	另见显示机床数据 MD203 DISPLAY_RESOLUTION 和 MD204 DISPLAY_RESOLUTION_INCH
	
	语法：
	`<function name="ncfunc.displayresolution" return="dislay_res" />`
	
	示例：
	`<let name="dislay_res" type="string"></let>`
	`...`
	`<function name="ncfunc.displayresolution" return="dislay_res" />`
	
	`<control name = "cdistToGo" xpos = "210" ypos = "156" refvar="nck/Channel/GeometricAxis/ progDistToGo[2]" hotlink="true" height="34" fieldtype="readonly" format="$$$dislay_res" time="superfast" color_bk="#ffffff"/>`

函数名称	含义
Ncfunc Get drive by axis name	该函数通过给定所属的轴名称来给出电机模块的寻址下标。 如未识别到相应的分配，函数返回数值 -1。如果未进行轴/驱动的分配，就会出现此情况。 **参数：** **str** - 轴名称 **返回值：** 电机模块下标 - 变量名称，此变量中会写入已获得的下标。 **语法：** <function name="NCFUNC.GETDRVBYAX_NAME" return="<index>"><axis name></function> **示例：** <let name="drv_idx" /> <function name="NCFUNC.GETDRVBYAX_NAME" return="drv_idx">_T"X1"</function> <control name = "c_do2" xpos = "140" ypos = "114" width = "320" fieldtype="itemlist" refvar="drive/dc/ p10[**do$ drv_idx**]" hotlink="true" color_bk="#f1f1f1"> <item value="0">Ready</item> <item value="1">Quick commissioning</item> <item value="2">Power unit commissioning</item> <item value="3">Motor commissioning</item> <item value="4">Encoder commissioning</item> <item value="5">Technological application/units </item> <item value="15">Data sets</item> <item value="17">Basic positioning commissioning </item> <item value="25">Commissioning the position control</item> <item value="29">Download</item> <item value="30">Parameter reset</item> <item value="95">Safety Integrated commissioning </item> </control> </init>

函数名称	含义

Ncfunc Get drive by axis index	该函数通过给定所属的轴下标来给出电机模块的寻址下标。
	如未识别到相应的分配，函数返回数值 -1。如果未进行轴/驱动的分配，就会出现此情况。
	
	参数：
	index - 轴下标
	
	返回值：
	电机模块下标 - 变量名称，此变量中会写入已获得的下标。
	
	语法：
	`<function name="NCFUNC.GETDRVBYAX_IDX" return="<index>"><axis index></function>`
	
	示例：
	`<let name="drv_idx" />`
	`<function name="NCFUNC.GETDRVBYAX_IDX" return="drv_idx">1</function>`
	`<control name = "c_do2" xpos = "140" ypos = "114" width = "320" fieldtype="itemlist" refvar="drive/dc/p10[`**`do$ drv_idx`**`]" hotlink="true" color_bk="#f1f1f1">`
	`<item value="0">Ready</item>`
	`<item value="1">Quick commissioning</item>`
	`<item value="2">Power unit commissioning</item>`
	`<item value="3">Motor commissioning</item>`
	`<item value="4">Encoder commissioning</item>`
	`<item value="5">Technological application/units</item>`
	`<item value="15">Data sets</item>`
	`<item value="17">Basic positioning commissioning</item>`
	`<item value="25">Commissioning the position control</item>`
	`<item value="29">Download</item>`
	`<item value="30">Parameter reset</item>`
	`<item value="95">Safety Integrated commissioning</item>`
	`</control>`
	`</init>`

函数名称	含义

Ncfunc Get drive by drive name	该函数通过给定电机模块名称来给出电机模块的寻址下标。
	如未识别到相应的分配，函数返回数值 -1。如果未进行轴/驱动的分配，就会出现此情况。
	
	参数：
	string - 电机模块名称
	
	返回值：
	电机模块下标 - 变量名称，此变量中会写入已获得的下标。
	
	语法：
	`<function name="NCFUNC.GETDRVBYDRV_NAME" return="<index>"><drive name></function>`
	
	示例：
	`<let name="drv_idx" />`
	`<function name="NCFUNC.GETDRVBYDRV_NAME" return="drv_idx">_T"SERVO_3.13:4"</function>`
	`<control name = "c_do2" xpos = "140" ypos = "114" width = "320" fieldtype="itemlist" refvar="drive/dc/p10[`**`do$ drv_idx`**`]" hotlink="true" color_bk="#f1f1f1">`
	`<item value="0">Ready</item>`
	`<item value="1">Quick commissioning</item>`
	`<item value="2">Power unit commissioning</item>`
	`<item value="3">Motor commissioning</item>`
	`<item value="4">Encoder commissioning</item>`
	`<item value="5">Technological application/units</item>`
	`<item value="15">Data sets</item>`
	`<item value="17">Basic positioning commissioning</item>`
	`<item value="25">Commissioning the position control</item>`
	`<item value="29">Download</item>`
	`<item value="30">Parameter reset</item>`
	`<item value="95">Safety Integrated commissioning</item>`
	`</control>`
	`</init>`

函数名称	含义
Ncfunc Get drive by bud address	该函数通过给定电机模块总线地址来给出电机模块的寻址下标。 如未识别到相应的分配，函数返回数值 -1。如果未进行轴/驱动的分配，就会出现此情况。 **参数：** **bus** - 总线编号 **slave** - 从站编号 **component** - 组件编号 **返回值：** 电机模块下标 - 变量名称，此变量中会写入已获得的下标。 **语法：** `<function name="NCFUNC.GETDRVBYBUS_ADDR" return="<index>"><bus>,<slave>,<component></ function>` **示例：** `<let name="drv_idx" /> <function name="NCFUNC.GETDRVBYBUS_ADDR" return="drv_idx"> 3,13,4 </function> <control name = "c_do2" xpos = "140" ypos = "114" width = "320" fieldtype="itemlist" refvar="drive/dc/p10[`**`do$ drv_idx`**`]" hotlink="true" color_bk="#f1f1f1"> <item value="0">Ready</item> <item value="1">Quick commissioning</item> <item value="2">Power unit commissioning</item> <item value="3">Motor commissioning</item> <item value="4">Encoder commissioning</item> <item value="5">Technological application/units</item> <item value="15">Data sets</item> <item value="17">Basic positioning commissioning</item> <item value="25">Commissioning the position control</item> <item value="29">Download</item> <item value="30">Parameter reset</item> <item value="95">Safety Integrated commissioning</item> </control> </init>`

函数名称	含义

Ncfunc Get MD limits	该函数将机床数据限值复制到数据类型为 **StructMDLimits** 的指定变量中。结构定义包含在文件 struct_def.xml 中。
	更多信息参见章节“创建 struct_def.xml 文件 (页 142)”
	
	参数：
	range - 必须输入一个字符串来指定机床数据范围：
	• **displayMD** - 显示机床数据
	• **generalMD** - NC 全局机床数据
	• **channelMD** - NC 通道专用机床数据
	• **axisMD** - NC 轴专用机床数据
	• **generalSettingMD** - NC 全局设定数据
	• **channelSettingMD** - NC 通道专用设定数据
	• **axisSettingMD** - NC 轴专用设定数据
	• **channelGUD** - NC 通道专用用户数据
	• **globalGUD** - NC 全局用户数据
	MD-number - 范围的机床数据编号
	variabel name - 调用该函数之后，该变量包含 Limits 的值
	range/unit - 可选项（如：通道编号）
	
	语法：
	`<function name="NCFUNC.GETMD_LIMITS"><range>, <MD-number, <variabel name>, <range/unit>></function>`
	
	示例：
	`<let name="limits" type="StructMDLimits" />`
	`<function name="NCFUNC.GETMD_LIMITS">_T"generalMD", 19220, _T"limits"</function>`
	`<op>`
	`upper_BAGS = limits.upperLimit;`
	`</op>`

函数名称	含义

Ncfunc bico to int	该函数将指定的 BICO 格式的字符串转换为整型值。（参见 SINAMICS）。
	
	语法：
	`<function name="ncfunc.bicotoint" return="integer variable">bico-string</function>`
	
	示例：
	`<let name="s_np0480_0" type="string"></let>`
	`<let name="i_p0480_0">0</let>`
	
	`<function name="ncfunc.bicotoint" return="i_p0480_0">s_np0480_0</function>`

Ncfunc int to bico	该函数将整型值转换为BICO 格式的字符串（参见 SINAMICS）。
	
	语法：
	`<function name="ncfunc.inttobico" return="string variable">integer variable</function>`
	
	示例：
	`<function name="ncfunc.inttobico" return="s_p0480_0">"drive/dc/p0480[0, DO2]"</function>`

函数名称	含义

Ncfunc is bico str valid	当结果是以 BICO 格式给定的字符串时，该函数会提供零值。（参见 SINAMICS）
	
	语法：
	`<function name="ncfunc.isbicostrvalid" return="integer variable">string varaible</function>`
	
	示例：
	`…`
	`<let name="s_np0480_0" type="string"></let>`
	`…`
	`…`
	`<control name = "cp0480_0" xpos = "402" ypos = "76" hotlink="true" refvar="s_np0480_0" >`
	`  <property item_data="4001" />`
	`</control>`
	`…`
	`…`
	`<function name="ncfunc.isbicostrvalid" return="valid">cp0480_0</function>`

Ncfunc password	该函数用来设置或取消 NC 口令等级。
	• 设置口令：
	所需口令等级的口令将作为参数给定。
	• 删除口令：
	空字符串会取消口令等级。
	
	语法：
	`<function name="ncfunc.password">口令 </function>`
	
	示例：
	`<let name="口令"type="string"></let>`
	
	`<function name="ncfunc.password" > 口令 </function>`
	`<function name="ncfunc.password" > _T"CUSTOMER" </function>`
	
	删除口令：
	`<function name="ncfunc.password" > _T"" </function>`

函数名称	含义

Control form color	该函数以字符串形式提供对话框的文本颜色或背景颜色。
	更多信息参见章节“颜色代码 (页 70)”
	
	区域：
	• BACKGROUND – 请求背景颜色值
	• TEXT – 请求文本（前景）颜色值
	
	语法：
	`<function name="control.formcolor" return="variable">_T"区域"</function>`
	
	示例：
	`<let name="bk_color" type="string"></let>`
	
	`<function name="control.formcolor" return="bk_color">_T"BACKGROUND"</function>`

函数名称	含义

Control local time	该函数将本地时间复制到包含 7 个元素的数组中。
	变量名称将会用于调用参数。
	数组元素中保存以下内容：
	• 下标 0 - 年份
	• 下标 1 - 月份
	• 下标 2 - 星期
	• 下标 3 - 日
	• 下标 4 - 小时
	• 下标 5 - 分钟
	• 下标 6 - 秒
	
	语法：
	`<function name ="control.localtime">_T"time_array" </function>`
	
	示例：
	`<!-- index`
	`0 = Year`
	`1 = Month`
	`2 = Day of week`
	`3 = Day`
	`4 = Hour`
	`5 = Minute`
	`6 = Second`
	`-->`
	`<let name="time_array" dim="7" />`
	
	`<function name ="control.localtime">_T"time_array" </function>`

Control exist	如果指定控件存在，该函数返回数值 1。
	
	语法：
	`<function name="CONTROL.EXIST" return="<return variable>"><control name></function>`
	
	示例：
	`<let name="ret" /> <function name="CONTROL.EXIST" return="ret">_T"edit"</function>`

函数名称	含义

Control is modified	如果指定的 EDIT 控件的内容已更改，该函数返回数值 1。
	
	语法：
	`<function name="CONTROL.ISMODIFIED" return="<return variable>"><control name></function>`
	
	示例：
	`<let name="ret" /> <function name="CONTROL.ISMODIFIED" return="ret">_T"edit"</function>`

Control is visible	如果指定控件可见，该函数返回数值 1。
	
	语法：
	`<function name="CONTROL.ISVISIBLE" return="<return variable>"><control name></function>`
	
	示例：
	`<let name="ret" /><function name="CONTROL.ISVISIBLE" return="ret">_T"edit"</function>`

Control set modified	该函数会更改指定 EDIT 控件的“更改”标记。
	
	参数：
	control name
	• 值 = 1 - 字段标记为已更改。
	• 值 = 0 - 字段标记为未更改。
	
	语法：
	`<function name="CONTROL.SETMODIFIED" return="<return variable>"><control name>, <Flag></function>`
	
	示例：
	`<let name="b" >1</let>`
	`<let name="ret" />`
	`<control name="edit" xpos="10" ypos="80" width="30" refvar="b" hotlink = "true" />`
	`<function name="CONTROL.SETMODIFIED" return="ret">_T"edit", 1</function>`

函数名称	含义

Control set working area	如果更改了**滚动视图**的内容，例如：删除或添加控件，则要重新计算滚动视图的可见区域。
	调用此函数执行计算。
	
	值：
	滚动视图名称
	
	示例：
	`...`
	`<control name="area" xpos="8" ypos="72" width="554" height="320" fieldtype="scrollarea" >`
	`  <control name="test" xpos="20" ypos="40" width="20" fieldtype="readonly" />`
	`</control >`
	
	`<control name="test1" xpos="20" ypos="80" width="20" fieldtype="readonly" parent="area"/>`
	`<control name="test2" xpos="20" ypos="100" width="20" fieldtype="readonly" parent="area"/>`
	
	`<function name="CONTROL.SETWORKINGAREA">_T"area"</function>`
	`...`

函数名称	含义

String to compare	对两个字符串从书写方式上进行比较。
	当字符串相同时该函数提供返回值零，当第一个字符串小于第二个时返回值小于零或者当第二个字符串小于第一个时返回值大于零。
	参数：
	str1 - 字符串
	str2 - 比较字符串
	
	语法：
	`<function name="string.cmp" return ="<int var>" > str1, str2 </function>`
	
	示例：
	`<let name="rval">0</let>`
	`<let name="str1" type="string">A brown bear hunts a brown dog.</let>`
	`<let name="str2" type="string">A brown bear hunts a brown dog.</let>`
	
	`<function name="string.cmp" return="rval"> str1, str2 </function>`
	结果：
	rval= 0

函数名称	含义

不区分大/小写的字符串比较	在不区分大/小写的情况下对两个字符串从书写方式上进行比较。
	当字符串相同时该函数提供返回值零，当第一个字符串小于第二个时返回值小于零或者当第二个字符串小于第一个时返回值大于零。
	
	参数：
	str1 - 字符串
	str2 - 比较字符串
	
	语法：
	`<function name="string.icmp" return ="<int var>" > str1, str2 </function>`
	
	示例：
	`<let name="rval">0</let>`
	
	`<let name="str1" type="string">A brown bear hunts a brown dog.</let>`
	`<let name="str2" type="string">A brown Bear hunts a brown Dog.</let>`
	`<function name="string.icmp" return="rval"> str1, str2 </function>`
	结果：
	rval= 0

函数名称	含义

String left	该函数提取字符串 1 中第一个的 nCount 字符并将其复制到返回变量中。
	
	参数：
	str1 - 字符串
	nCount - 字符数量
	
	语法：
	`<function name="string.left" return="<result string>"> str1,  nCount </function>`
	
	示例：
	`<let name="str1" type="string">A brown bear hunts a brown dog.</let>`
	`<let name="str2" type="string"></let>`
	
	`<function name="string.left" return="str2"> str1, 12 </function>`
	结果：
	str2="A brown bear"

函数名称	含义

String right	该函数提取字符串 1 中最后一个 nCount 字符并将其复制到返回变量中。
	
	参数：
	str1 - 字符串
	nCount - 字符数量
	
	语法：
	`<function name="string.right" return="<result string>"> str1, nCount </function>`
	
	示例：
	`<let name="str1" type="string">A brown bear hunts a brown dog.</let>`
	`<let name="str2" type="string"></let`
	
	`<function name="string.right" return="str2"> str1, 10 </function>`
	结果：
	str2="brown dog."

函数名称	含义

String middle	该函数从下标 iFirst 开始提取字符串 1 中指定数量的字符并将其复制到返回变量中。
	
	参数：
	str1 - 字符串
	iFirst - 起始下标
	nCount - 字符数量
	
	语法：
	`<function name="string.middle" return="<result string>"> str1, iFirst, nCount </function>`
	
	示例：
	`<let name="str1" type="string">A brown bear hunts a brown dog.</let>`
	`<let name="str2" type="string"></let`
	
	`<function name="string.middle" return="str2"> str1, 2, 5 </function>`
	结果：
	str2="brown"

String length	该函数可以提供字符串的字符数量。
	
	参数：
	str1 - 字符串
	
	语法：
	`<function name="string.length" return="<int var>"> str1 </function>`
	
	示例：
	`<let name="length">0</let>`
	
	`<let name="str1" type="string">A brown bear hunts a brown dog.</let>`
	`<function name="string.length" return="length"> str1 </function>`
	结果：
	length = 31

函数名称	含义

Strings to replace	该函数可以使用新的字符串替代所有查找到的字符串部分。
	
	参数：
	string - 字符串变量
	find string - 要替代的字符串
	new string - 新字符串
	
	语法：
	`<function name="string.replace"> string, find string, new string </function>`
	
	示例：
	`<let name="str1" type="string">A brown bear hunts a brown dog. </let>`
	
	`<function name="string.replace" > str1, _T"a brown dog" , _T"a big salmon"</function>`
	结果：
	str1 = "A brown bear hunts a big salmon!"

Strings to remove	该函数可以删除所有查找到的字符串部分。
	
	参数：
	string - 字符串变量
	remove string - 要删除的字符串部分
	
	语法：
	`<function name="string.remove"> string, remove string </function>`
	
	示例：
	`<let name="index">0</let>`
	
	`<let name="str1" type="string">A brown bear hunts a brown dog. </let>`
	`<function name="string.remove" > str1, _T"a brown dog" </function>`
	结果：
	str1 = "A brown bear hunts"

函数名称	含义

Strings to insert	该函数可以在指定的下标中插入一个字符串。
	
	参数：
	string - 字符串变量
	index - 下标（从零开始）
	insert string - 要插入的字符串
	
	语法：
	`<function name="string.insert"> string, index, insert string </function>`
	
	示例：
	`<let name="str1" type="string">A brown bear hunts. </let>`
	`<let name="str2" type="string">a brown dog</let>`
	
	`<function name="string.insert" > str1, 19, str2 </function>`
	结果：
	str1 = "A brown bear hunts a brown dog"

函数名称	含义

String delete	该函数可以从指定的启始位置开始删除确定数量的字符。
	
	参数：
	string - 字符串变量
	start index - 起始下标（从零开始）
	nCount - 要删除的字符数量
	
	语法：
	`<function name="string.delete"> string, start index , nCount </function>`
	
	示例：
	`<let name="str1" type="string">A brown bear hunts. </let>`
	
	`<function name="string.delete" > str1, 2, 5 </function>`
	结果：
	str1 = "A bear hunts"

函数名称	含义

String find	该函数可以在所传输的字符串中查找与部分字符串一致的第一处地方。
	如果找到了部分字符串，则该函数会提供到第一个字符的下标（从零开始），否则为 -1。
	
	参数：
	string - 字符串变量
	findstring - 要查找的字符串
	startindex - 起始索引 （可选）
	
	语法：
	`<function name="string.find" return="<int val>"> str1, find string </function>`
	
	示例：
	`<let name="index">0</let> <let name="str1" type="string">A brown bear hunts a brown dog. </let> <function name="string.find" return="index"> str1, _T"brown" </function>`
	
	结果：
	索引 = 2
	
	或
	`<function name="string.find" return="index"> str1, _T"brown", 1 </function>`

函数名称	含义

String reverse find	该函数可以在所传输的字符串中查找与部分字符串一致的最后一处地方。
	如果找到了部分字符串，则该函数会提供到第一个字符的下标（从零开始），否则为 -1。
	
	参数：
	string - 字符串变量
	find string - 要查找的字符串
	startindex - 起始索引 （可选）
	
	语法：
	`<function name="string.reversefind" return="<int val>"> str1, find string </function>`
	
	示例：
	`<let name="index">0</let>`
	`<let name="str1" type="string">A brown bear hunts a brown dog. </let>`
	`<function name="string.reversefind" return="index"> str1, _T"brown" </function>`
	
	结果：
	索引 = 21
	
	或
	`<function name="string.reversefind" return="index"> str1, _T"brown" 10 </function>`
	
	结果：
	索引 = 2

函数名称	含义

String GetAt	该函数可读取特定位置的一个字符。
	参数：
	str - 字符串
	index - 待读取字符的下标，从零开始编号
	
	返回值：
	给出一个字符串变量名，字符应当储存在此变量中。
	
	语法：
	`<function name="string.getat" return="<result string>"><string>, <index></function>`
	
	示例：
	`<let name="resStr" type="string" />`
	`<function name="string.getat" return="resStr">_T"brown", 2</function>`

函数名称	含义

String pixel height	以像素为单位计算字符串的高度。在指定控件或窗体的当前字体格式设置下进行计算。控件名称作为字符串提供。如果指定的是空字符串，计算则基于窗体进行。
	
	参数：
	control name
	字符串
	字符串可直接指定为文本或指定为字符串变量。
	
	返回值：
	名称为整型变量，其中写入高度值。
	
	语法：
	`<function name="STRING.PIXELHEIGHT" return="<return variable>"><control name>,<varaible or text></ function>`
	
	示例：
	根据窗体的字体计算高度
	`<let name="pixelsh" />`
	`...`
	`<function name="STRING.PIXELHEIGHT" return="pixelsh">_T"", cedit1</function>`
	
	根据控件的字体计算高度
	`<function name="STRING. PIXELHEIGHT" return="pixelsh"> cedit1, cedit1</function>`

函数名称	含义

String pixel width	以像素为单位计算字符串的宽度。在指定控件或窗体的当前字体格式设置下进行计算。控件名称作为字符串提供。如果指定的是空字符串，计算则基于窗体进行。
	
	参数：
	control name
	字符串
	字符串可直接指定为文本或指定为字符串变量。
	
	返回值：
	名称为整型变量，其中写入文本宽度值。
	
	语法：
	`<function name="STRING.PIXELWIDTH" return="<return variable>"><control name>,<varaible or text></ function>`
	
	示例：
	根据窗体的字体计算宽度
	`<let name="pixels" />`
	`...`
	`<function name="STRING. PIXELWIDTH" return="pixels">_T"", cedit1</function>`
	
	根据控件的字体计算宽度
	`<function name="STRING.PIXELWIDTH" return="pixels"> cedit1, cedit1</function>`

函数名称	含义
String SetAt	该函数用于在特定位置写一个字符。此下标必须小于文本最大长度。 **参数：** **str** - 字符串 **char** - 应在特定位置写入的字符 **index** - 目标变量字符串的下标，从零开始编号 **语法：** <function name="string.setat" ><destination string>, <character string>, <index></function> **示例：** <let name="str" type="string" >br_wn</string> <function name="string.setat">str, ">_T"o", 2 </function>

函数名称	含义

String Split	该函数将一个字符串分隔为多个子字符串并将其复制到给定的字符串数组中。此分隔在给定的分隔符处发生。分隔符不会存储在子字符串中。
	如果找到的分隔符数量超过既定的数组大小，该函数会自动扩展数组。
	
	参数：
	str - 字符串
	char - 包含分隔符的变量名称
	number - 包含该函数执行完毕后产生的子字符串数量的变量名称。
	
	返回值：
	给出包含该函数执行完毕后的子字符串的字符串数组的名称。
	
	语法：
	`<function name=" string.split" return="<result string array>"><string>, <char>, <number></function>`
	
	示例：
	`<let name="strlist" type="string" dim="2"/>`
	`<let name="str_num" />`
	`<function name="string.split" return="strlist">`
	`_T"brown;green;blue;red", _T";", str_num </function>`

函数名称	含义

String trim left	该函数可以从字符串中删除打头的空格。
	
	参数：
	str1 - 字符串变量
	
	语法：
	`<function name="string.trimleft" > str1 </function>`
	
	示例：
	`<let name="str1" type="string">test trim left</let>`
	`<function name="string.trimleft" > str1 </function>`
	结果：
	str1 = "test trim left"

String trim right	该函数可以从字符串中删除后续的空格。
	
	参数：
	str1 - 字符串变量
	
	语法：
	`<function name="string.trimright" > str1 </function>`
	
	示例：
	`<let name="str1" type="string"> test trim right`
	`</let>`
	`<function name="string.trimright" > str1`
	`</function>`
	结果：
	str1 = "test trim right"

函数名称	含义
Sinus	该函数可以计算所传输值（单位度）的正弦值。 **参数：** **double** - 角度 **语法：** <function name="sin" return="<double val>"> double </function> **示例：** <let name= "sin_val" type="double"></let> <function name="sin" return="sin_val"> 20.0 </function>
Cosinus	该函数可以计算所传输值（单位度）的余弦值。 **参数：** **double** - 角度 **语法：** <function name="cos" return="<double val>"> double </function> **示例：** <let name= "cos_val" type="double"></let> <function name="cos" return="cos_val"> 20.0 </function>

函数名称	含义
Tangens	该函数可以计算所传输值（单位度）的正切值。 **参数：** **double** - 角度 **语法：** <function name="tan" return="<double val>"> double </function> **示例：** <let name= "tan_val" type="double"></let> <function name="tan" return="tan_val"> 20.0 </function>
ARCSIN	该函数可以计算所传输值（单位度）的反正弦值。 **参数：** **double** - x 取值范围 -PI/2 至 +PI/2 **语法：** <function name="arcsin" return="<double val>"> double </function> **示例：** <let name= "arcsin_val" type="double"></let> <function name="arcsin" return="arcsin_val"> 20.0 </function>

函数名称	含义
ARCOS	该函数可以计算所传输值（单位度）的反余弦值。 **参数：** **double -** x 取值范围 -PI/2 至 +PI/2 **语法：** <function name="arcos" return="<double val>"> double </function> **示例：** <let name= "arccos_val" type="double"></let> <function name="arccos" return="arccos_val"> 20.0 </function>
ARCTAN	该函数可以计算所传输值（单位度）的反正切值。 **参数：** **double** - 反正切 y/x **语法：** <function name="arctan" return="<double val>"> double </function> **示例：** <let name= "arctan_val" type="double"></let> <function name="arctan" return="arctan_val"> 20.0 </function>
文件处理	

函数名称	含义

读取文件	该函数可以将指定文件的内容读到一个字符串变量中。
	或者可以将待读取的字符数量指定为第二参数。
	
	属性：
	name - 文件名只能用小写字母。
	需要访问另一个目录中的文件时，可采用相对路径，即目录从 appl 或 dvm 开始。
	return - 局部变量的名称
	
	参数：
	progname - 文件名
	number of characters - 待读取的字符数量，字节（可选）
	
	语法：
	`<function name="doc.readfromfile" return="<string var>"> progname, number of characters </function>`
	
	示例：
	`<let name = "my_var" type="string" ></let>`
	
	NC 文件系统
	`<function name="doc.readfromfile" return="my_var"> _T"n:\mpf\test.mpf" </function>`
	
	CF 卡
	`<function name="doc.readfromfile" return="my_var"> _T"f:\appl\test.mpf" </function>`
	
	或
	
	`<function name="doc.readfromfile" return="my_var"> _T".\test.mpf" </function>`

函数名称	含义

写文件	该函数可以将一个字符串变量的内容写入指定文件中。
	文件名只能用小写字母。
	需要访问另一个目录中的文件时，可采用相对路径，即目录从 appl 或 dvm 开始。
	
	参数：
	progname - 文件名
	str1 - 字符串
	
	语法：
	`<function name="doc.writetofile" > progname, str1 </function>`
	
	示例：
	`<let name = "my_var" type="string" > file content </let>`
	
	NC 文件系统
	`<function name="doc.writetofile">_T"n:\mpf\test.mpf", my_var </function>`
	
	CF 卡
	`<function name="doc.writetofile">_T"f:\appl\test.mpf", my_var </function>`
	
	或
	
	`<function name="doc.writetofile">_T".\test.mpf", my_var </function>`

函数名称	含义

删除文件	该函数可以从目录中删除指定的文件。
	文件名只能用小写字母。
	需要访问另一个目录中的文件时，可采用相对路径，即目录从 appl 或 dvm 开始。
	
	参数：
	progname - 文件名
	
	语法：
	`<function name="doc.remove" > progname </function>`
	
	示例：
	NC 文件系统
	`<function name="doc.remove">_T"n:\mpf\test.mpf" </function>`
	
	CF 卡
	`<function name="doc.remove">_T"f:\appl\test.mpf" </function>`
	
	或
	
	`<function name="doc.remove">_T".\test.mpf" </function>`

函数名称	含义

提取脚本文件部分	该函数将零件程序中包含的对话框说明复制到指定的局部变量中。
	
	指定用于保存主菜单名称的对话框名称和参数的程序名称用作调用参数。如果在零件程序中找到了对话框说明名称，则 Return 变量中会包含该说明。如果变量内容保存在文件中，则可以通过间接调用执行脚本文件。
	
	系统提供脚本文件，可从激活的零件程序中提取对话框说明并激活对话框。可通过 MMC 指令调用脚本文件，从而激活属于零件程序的对话框。
	
	语法：
	`<function name="doc.loadscript" return="<name of script variable>">progname, _T"dialog part name", main menu </function>`
	
	属性：
	return - 保存有所提取的脚本的变量
	
	参数：
	progname - 程序中的完整路径（该路径名称可传输至 DOS 记录函数中。）
	main menu - 找到的菜单名称复制到该变量中
	dialog part name - 包含对话框说明的标签名称
	
	示例：
	`<function name="doc.loadscript" return="contents">prog_name, _T"main_dialog", entry</function>`

函数名称	含义

Exist	如文件存在，则函数提供值 1。
	文件名只能用小写字母。
	需要访问另一个目录中的文件时，可采用相对路径，即目录从 appl 或 dvm 开始。
	
	参数：
	progname - 文件名
	
	语法：
	`<function name="doc.exist" return="<int_var>" > progname </function>`
	
	示例：
	`<let name ="exist">0</let>`
	
	NC 文件系统
	`<function name="doc.exist" return="exist">_T"n:\mpf \test.mpf" </function>`
	
	CF 卡
	`<function name="doc.exist" return="exist">_T"f:\appl \test.mpf" </function>`
	
	或
	
	`<function name="doc.exist" return="exist">_T".\test.mpf" </function>`

函数名称	含义

NC 程序选择	该函数可以选择将要处理的程序。程序必须保存在 NC 文件系统中。
	
	参数：
	progname - 文件名
	
	语法：
	`<function name="ncfunc.select"> progname </function>`
	
	示例：
	NC 文件系统
	`<function name="ncfunc.select"> _T"n:\mpf\test.mpf" </function>`

设置位	该函数用于设置给定变量的各个位。
	可以置位或复位。
	
	语法：
	`<function name="ncfunc.bitset" refvar="address" value="set/reset" > bit0, bit1, … bit9 </function>`
	
	属性：
	refvar - 指定要写入的位组合所在变量的名称
	value – 位的取值为 0 和 1
	
	值：
	作为函数值，位号从 0 开始给定。
	每次调用最多可以修改 10 个位。
	
	示例：
	`<function name="ncfunc.bitset" refvar="nck/Channel/Parameter/R[1]" value="1" > 0, 2, 3, 7 </function>`
	
	`<function name="ncfunc.bitset" refvar="nck/Channel/Parameter/R[1]" value="0" > 1, 4 </function>`

函数名称	含义

删除控件	该函数可删除指定的控件。
	
	语法：
	`<function name="control.delete"> control name </function>`
	
	属性：
	name – 函数名称
	
	值：
	control name – 控件的名称
	
	示例：
	`<function name="control.delete"> _T"my_editfield" </function>`

函数名称	含义

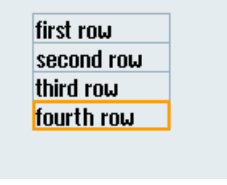
Add Item	该函数可在列表末尾添加一个新元素。
	提示：
	该函数只能用于控件类型“listbox”和“graphicbox”。
	
	语法：
	`<function name="control.additem"> control name, item </function>`
	
	属性：
	name – 函数名称
	
	值：
	control name – 控件名称
	item - 要添加的表达式
	itemdata - 整数值；由用户确定
	
	示例：
	`<let name ="itemdata">1</let>`
	…
	…
	…
	`<op> item_string = _T"text1" </op>`
	`<function name="control.additem">_T"listbox1", item_string, itemdata`
	`</function>`

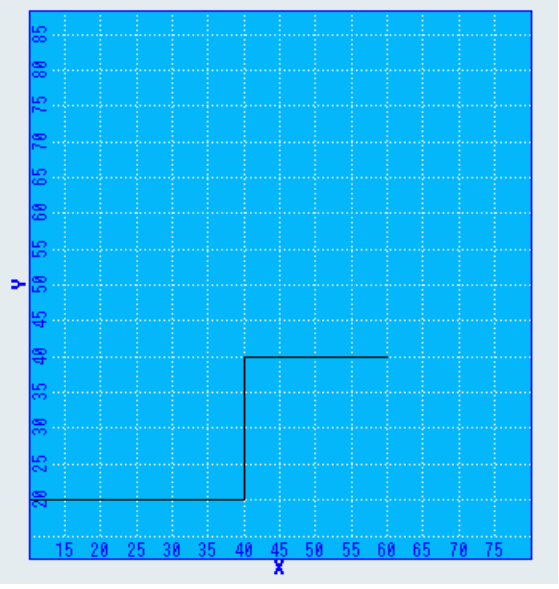
函数名称	含义
Insert Item	该函数可在指定位置添加一个新元素。 提示： 该函数只能用于控件类型“listbox”。 **语法：** <function name="control.insertitem"> control name, index, item, itemdata </function> **属性：** **name** – 函数名称 **值：** **control name** – 控件名称 **index** – 位置从零开始 **item** - 要添加的表达式 **itemdata** - 整数值；由用户确定 **示例：** <let name ="itemdata">1</let> … … … <op> item_string = _T"text2" </op> <function name="control.insertitem">_T"listbox1", 1, item_string, itemdata </function>

函数名称	含义

Delete Item	该函数可删除指定位置上的元素。
	提示：
	该函数只能用于控件类型“listbox”。
	
	语法：
	`<function name="control.deleteitem"> control name, index </function>`
	
	属性：
	name - 函数名称
	
	值：
	control name – 控件名称
	index- 下标从 0 开始
	
	示例：
	`<function name="control.deleteitem">_T"listbox1", 1</function>`

函数名称	含义

Load Item	该函数可向控件中添加一个表达式列表。
	该函数只能用于控件类型"listbox"和"graphicbox"。
	
	语法：
	`<function name="control.loaditem"> control name, list </function>`
	
	属性：
	name – 函数名称
	
	值：
	control name – 控件名称
	list 字符串变量
	在变量中包含的字符串必须符合下面描述的结构。
	
	列表结构：
	列表包含一定数量的表达式，它们之间用 \\n 分隔。
	
	示例：
	在本例中，内容通过函数 control.loaditem 分配给了列表框。
	控制符\\n 分开了可列为不同的行的表达式。
	`<CONTROL name = "list" xpos = "160" ypos = "32" width ="100" height="200" fieldtype="listbox"/>`
	`<let name="lb_list" type="string" />`
	`<op>`
	`lb_list = _T"first row\\n";`
	`lb_list = lb_list + _T"second row\\n";`
	`lb_list = lb_list + _T"third row\\n";`
	`lb_list = lb_list +_T"fourth row\\n";`
	`</op>`
	`<function name="control.loaditem">_T"list", lb_list</function>`
	

函数名称	含义
Load Item 续	**示例：** 在本例中，内容通过函数 control.loaditem 分配给了图形框。 控制符\\n 分开了不同的图表元素。 `<control name= "c_gbox" xpos = "6" ypos="32" width="328" height="356" fieldtype="graphicbox" /> <let name="plot_list" type="string" /> <op> plot_list = _T"l;10;20;40;20\\n"; plot_list = plot_list + _T"l;40;20;40;40\\n"; plot_list = plot_list + _T"l;40;40;60;40\\n"; plot_list = plot_list +_T"l;80;0;80;100\\n"; </op> <function name="control.loaditem">_T"c_gbox", plot_list</function>` 

函数名称	含义

Empty	该函数可清除指定列表框或图形框的内容。
	
	语法：
	`<function name="control.empty"> control name, </function>`
	
	属性：
	name – 函数名称
	
	值：
	control name – 控件名称
	
	示例：
	`<function name="control.empty">_T"listbox1" </function>`

Get focus	该函数可提供当前正处在输入状态的控件的名称。
	
	语法：
	`<function name="control.getfocus" return="focus_name" />`
	
	属性：
	name – 函数名称
	return – 给定字符串变量，将控件的名称复制到其中。
	
	示例：
	`<let name>="focus_field" type="string"></let>`
	`<function name="control.getfocus" return="focus_field"/>`

函数名称	含义

Set focus	该函数可将指定的控件设置到输入状态。
	控件的名称将以函数的文本表达方式传输。
	
	语法：
	`<function name="control.setfocus"> control name </function>`
	
	属性：
	name – 函数名称
	
	值：
	control name – 控件的名称
	
	示例：
	`<function name="control.setfocus" ">_T"listbox1" </function>`

Get cursor selection	该函数可提供列表框中光标所在的下标。
	控件的名称将以函数的文本表达方式传输。
	
	语法：
	`<function name="control.getcurssel" return="var">control name </function>`
	
	示例：
	`<let name>="index"></let>`
	`<function name="control.getcurssel" return="index">_T"listbox1"</function>`

函数名称	含义

Set cursor selection	该函数可在列表框中将光标放置在相应的行中。
	控件的名称将以函数的文本表达方式传输。
	
	语法：
	`<function name="control.setcurssel" > control name, index</function>`
	
	示例：
	`<let name>="index">2</let>`
	`<function name="control.setcurssel" ">_T"listbox1",index</function>`

Get Item	该函数可将列表框中所选择行的内容复制到指定的变量中。
	提供了字符串变量作为参考变量。
	控件的名称将以函数的文本表达方式传输。
	
	语法：
	`<function name="control.getitem" return="var"> control name, index </function>`
	
	示例：
	`<let name>="index">2</let>`
	`<let name>="item" type="string"></let>`
	
	`<function name="control.getitem" return="item" ">_T"listbox1",index</function>`

函数名称	含义

Get Item Data	该函数可将列表框中用户自定义的元素值复制到指定的变量中。
	在 EDIT 控件中，该函数可将用户自定义的值（item_data）复制到指定的变量中。
	提供了整型变量作为参考变量。
	控件的名称将以函数的文本表达方式传输。
	
	语法：
	`<function name="control.getitemdata" return="var">`
	`control name, index </function>`
	
	示例：
	`<let name>="index">2</let>`
	`<let name>="itemdata"></let>`
	
	`<function name="control.getitemdata"`
	`return="itemdata" ">_T"listbox1",index</function>`

函数名称	含义
Image Box Set	该函数用于 **Imagebox** 和 **Graphicbox** 控件可见区域的控制。可以指定控件名称、控制指令和相应的值作为调用参数。 **语法：** `<function name="control.imageboxset">control name, command, command parameter</function>` **调用参数：** • **x_offs** 该函数将图像区移至指定的 X 位置。指定左边角度的坐标为其他参数。 • **y_offs** 该函数将图像区移至指定的 Y 位置。指定上边角度的坐标为其他参数。 • **xy_offs** 该函数将图像区移至指定的 X/Y 位置。指定左边和上边角度的坐标为其他参数。 • **followCursor** 图像区遵循设置的光标位置。 • **zoomplus** 图像以 0.12 的系数放大。 • **zoomminus** 图像以 0.12 的系数缩小。 • **autozoom** 图像根据显示区域自动缩放。 • **Update** 控件重新标记。 • **SetBkColor** 该指令用于设置指定的背景颜色。指定颜色代码为其他参数。 • **SetCursorRect** 指定为矩形的图像区移至可见区域。 • **SetAnimationState** （仅适用于 Imagebox 控件） – **start** - 指令启动动画。 – **stop** - 指令停止动画。

函数名称	含义

Image Box Set 续	**示例：**
	`...`
	`...`
	`<softkey POSITION="1">`
	`<caption>zoom%nin</caption>`
	`<function name="control.imageboxset">_T"c_gbox",`
	`_T"zoomplus"</function>`
	`</softkey>`
	`...`
	`...`
	`<form name = "main_form">`
	`<init>`
	`<caption> graphicbox</caption>`
	`<control name= "c_gbox" xpos = "6" ypos="32"`
	`width="328" height="356" fieldtype="graphicbox" />`
	`<let name="plot_list" type="string" />`
	`<op>`
	`plot_list = _T"1;10;20;40;20\\n";`
	`plot_list = plot_list + _T"1;40;20;40;40\\n";`
	`plot_list = plot_list + _T"1;40;40;60;40\\n";`
	`plot_list = plot_list +_T"1;80;0;80;100\\n";`
	`</op>`
	`<function name="control.loaditem">_T"c_gbox",`
	`plot_list</function>`
	`</init>`
	`</form>`

Image Box Get	该函数获取 Imagebox 控件属性。可以指定控制指令和相应的值作为调用参数。
	
	语法：
	`<function name="control.imageboxget">control name, command, command parameter</function>`
	
	调用参数：
	GetViewSize
	提供显示区尺寸
	指定 SIZE 结构类型的变量作为其他参数。
	
	示例：
	`<let name="view_size" type="size" />`
	`<function name="control.imageboxget">_T"topoview",`
	`_T"`**`GetViewSize`**`", view_size</function>`

函数名称	含义

Bitmap Dim	该函数将位图尺寸重新复制到 SIZE 结构类型的变量中。将 struct_def.xml in 文件整合至项目中进行类型定义。
	更多信息参见章节“创建 struct_def.xml 文件 (页 142)”
	
	语法：
	`<function name="bitmap.dim" >name, variable type </function>`
	
	参数：
	name - 文件路径
	variable type - SIZE 类型变量的名称
	
	示例：
	`<let name="bmp_size" type="size" />`
	
	`<function name="bitmap.dim" >_T"test.bmp", bmp_size </function>`

Screen Resolution	该函数将系统的绝对屏幕分辨率重新复制到 SIZE 结构类型的变量中。将 struct_def.xml in 文件整合至项目中进行类型定义。 **更多信息**参见章节“创建 struct_def.xml 文件 (页 142)” **语法：** `<function name="hmi.screen_resolution" >varialble type </function>`
Get HMI Resolution	该函数将 SINUMERIK Operate 所用的屏幕分辨率重新复制到 SIZE 结构类型的变量中。将 struct_def.xml in 文件整合至项目中进行类型定义。 **更多信息**参见章节“创建 struct_def.xml 文件 (页 142)” **语法：** `<function name="hmi.get_hmi_resolution" >variable type </function>`

函数名称	含义
Get Caption Height	该函数提供标题栏高度，单位：像素。 **语法：** `<function name="hmi.get_caption_height" return="<return var>" />` **属性：** **return** - 整形变量
Abs	该函数提供给定数值的绝对值。 **语法：** `<function name="abs" return="var"> value </function>`
SDEG	该函数可将给定值换算为度。 **语法：** `<function name="sdeg" return="var"> value </function>`
SRAD	该函数可将给定值换算为弧度。 **语法：** `<function name="srad" return="var"> value </function>`
SQRT	该函数可以计算给定值的根。 **语法：** `<function name="sqrt" return="var"> value </function>`
ROUND	该函数可以将所传输的数值的小数位保留到指定的位数。如果未指定要保留的小数点后的位数，则该函数会保留到小数点后第一位。 **语法：** `<function name="round" return="var"> value, nDecimalPlaces </function>`

函数名称	含义
FLOOR	该函数提供小于等于所传输值的最大整数值。 **语法：** `<function name="floor" return="var"> value </function>`
CEIL	该函数提供大于等于所传输值的最小的整数值。 **语法：** `<function name="ceil" return="var"> value </function>`
LOG	该函数可以计算给定值的对数。 **语法：** `<function name="log" return="var"> value </function>`
LOG10	该函数可以计算给定值的十进制对数。 **语法：** `<function name="log10" return="var"> value </function>`
POW	该函数计算“a^b”的值。 **语法：** `<function name="pow" return="var"> a, b </function>`
MIN	该函数会比较所传输的值并将较小的值返回。 **语法：** `<function name="min" return="var"> value1, value2 </function>`
MAX	该函数会比较所传输的值并将较大的值返回。 **语法：** `<function name="max" return="var"> value1, value2 </function>`

函数名称	含义
RANDOM	该函数提供伪随机数。 **语法：** `<function name="random" return="var" </function>`

函数名称	含义
MMC	**函数：** 通过 MMC 指令可以在 SINUMERIK Operate 上从零件程序中显示用户自定义对话框窗口。对话框窗口的式样由纯文本配置确定（制造商目录中的 XML 文件），系统软件保持不变。 Operate-Base 系统中发生变化后，参数如下： XML → CYCLES 或 POPUPDLG XML_ON → PICTURE_ON XML_OFF → PICTURE_OFF **语法：** MMC（“操作区、指令、XML 脚本文件、菜单名称、预留、预留、显示时间或者应答变量、预留”，“应答模式”） **含义：** • MMC 在 SINUMERIK Operate 上从零件程序中调用交互式对话框窗口。 • EasyXml 应从零件程序中开始的用户对话框的标识。 • XML_ON 或 XML_OFF 指令：对话框选择或对话框取消选择 • XML 文件 XML 脚本文件的名称 • 菜单 管理待显示对话框的菜单标签的名称 • 预留 预留用于 SINUMERIK Operate • 预留 预留用于 SINUMERIK Operate • 时间 应答模式“N”中的对话框显示时间 • 预留 预留用于 SINUMERIK Operate • 应答模式 “S” 同步，通过软键“OK”应答 “N” 异步，在规定的时间后关闭对话框

函数名称	含义

MMC 续	**同步调用示例：**
	Operate-Base 系统中发生变化后，参数如下：
	XML → CYCLES 或 POPUPDLG
	XML_ON → PICTURE_ON
	XML_OFF → PICTURE_OFF
	NC 指令
	`MMC("EASYXML oder CYCLES”,XML_ON,mmc_cmd.xml,cmd1,,,,,","S")`
	文件： mmc_cmd.xml
	`<menu name = "cmd1">`
	`<open_form name = "cmd1_form" />`
	`<softkey_ok>`
	`<close_form />`
	`</softkey_ok>`
	`</menu>`
	`<form name = "cmd1_form" xpos ="12" ypos="100" width="500" height="240">`
	`<init>`
	`</init>`
	`<paint>`
	`</paint>`
	`</form>`
	异步调用示例（无应答）：
	Operate-Base 系统中发生变化后，参数如下：
	XML → CYCLES 或 POPUPDLG
	XML_ON → PICTURE_ON
	XML_OFF → PICTURE_OFF
	NC 指令
	`MMC("EASYXML oder CYCLES”,PICTURE_ON,mmc_cmd.xml,cmd1,,,10,,","N")`
	文件： mmc_cmd.xml `<menu name = "cmd1">`
	`<open_form name = "cmd1_form" />` `<softkey_ok>`
	`<close_form />` `</softkey_ok>`
	`</menu>`
	`<form name = "cmd1_form" xpos ="12" ypos="100" width="500" height="240">`
	`<init>`

函数名称	含义
	</init> <paint> </paint> </form>

函数名称	含义

MMC 续	**从零件程序中提取脚本文件部分示例**
	Operate-Base 系统中发生变化后，参数如下：
	XML → CYCLES 或 POPUPDLG
	XML_ON → PICTURE_ON
	XML_OFF → PICTURE_OFF
	NC 指令
	`MMC("XML,XML_ON,XMLDIAL_EMB.XML,main","S")`
	文件：xmldial_emb.xml
	
	`<DialogGui>`
	`<let name="menu_name" type="string">main</let>`
	`<let name="script_loaded">0</let>`
	`<menu name = "main">`
	`<if>`
	`<condition>script_loaded == 0</condition>`
	`<then>`
	`<function name ="load_current_program" />`
	`<dynamic_include src="__tmp.xml" />`
	`<op>`
	`script_loaded = 1;`
	`</op>`
	`<navigation>$$$menu_name</navigation>`
	`</then>`
	`</if>`
	`<softkey_back>`
	`<close_form />`
	`</softkey_back>`
	`</menu>`
	`<function_body name ="load_current_program" >`
	`<let name="prog_name" type= "string"/>`
	`<let name="contents" type= "string"/>`
	`<let name="entry" type= "string"/>`
	`<let name="len" />`
	`<op> prog_name = "nck/Channel/ProgramInfo/workPandProgName"</op>`
	`<function name="DOC.LOADSCRIPT" return="contents">prog_name, _T"main_dialog", entry</function>`
	`<function name="string.length" return="len">contents</function>`
	`<if>`
	`<condition>len > 0</condition>`
	`<then>`
	`<op>`
	`menu_name = entry;`
	`</op>`

函数名称	含义
	<function name="doc.writetofile" >_T"__tmp.xml", contents</function> </then> </if> </function_body> </DialogGui>

函数名称	含义
MMC 续	程序示例 ;<main_dialog entry="rpara_main"> ; <let name="xpos" /> ; <let name="ypos" /> ; <let name="field_name" type="string" /> ; <let name="num" /> ; <menu name="rpara_main"> ; <open_form name="rpara_form"/> ; <softkey_back> ; <close_form /> ; </softkey_back> ;</menu> ; ;<form name="rpara_form"> ; <init> ; <caption>test mask</caption> ; <let name="count" >0</let> ; <op> ; xpos = 120; ; ypos = 34; ; ; "nck/Channel/Parameter/R[10]" = 10; ; </op> ; <!-- load the number of controls --> ; <op> ; num = "nck/Channel/Parameter/R[10]"; ; </op> ; ; <while> ; <condition>count < num</condition> ; <print name="field_name" text="edit%d">count</print> ; ; <control name = "$field_name" xpos = "$xpos" ypos = "$ypos" refvar="nck/Channel/Parameter/R[$count]" hotlink="true" /> ; <op> ; ypos = ypos +24; ; count = count +1; ; </op> ; </while> ; ; </init> ; ; <paint> ; <op> ; xpos = 8; ; ypos = 36; ; count = 0;

函数名称	含义
	; </op> ; <while> ; <condition>count < num</condition> ; <print name="field_name" text="R-Parameter %d">count </print> ; ; <text xpos = "$xpos" ypos = "$ypos" >$$ $field_name</text> ; <op> ; ypos = ypos +24; ; count = count +1; ; </op> ; </while> ; ; </paint> ; ;</form> ; ;</main_dialog> … G94 F100 MMC("XML,XML_ON,XMLDIAL_EMB.XML,main","A") … … … MMC("XML,XML_OFF,XMLDIAL_EMB.XML,main","A") G4 F2 M2

函数名称	含义
ISNUMERIC	该函数用于检查字符串内容是否可作为数字分析。空格和表格均被忽略。此外，符号和小数点均被视为有效字符。如果满足条件，该函数返回数值 1。 **参数：** **string** - 字符串 **语法：** `<function name="isnumeric" return="result"><string></function>`
ROOT	该函数用于从数字中调用第 n 位根。 **参数：** **value** - 数字 **n** - 根说明 **语法：** `<function name="ROOT" return="result"><value>, <n></function>`

2.12 多点触控操作

2.12.1 多点触控功能

概述

随着多点触控显示屏的投入使用，屏幕操作也需要根据相应的功能扩展来加以调整。例如取消了位置固定的软键，由可自由定位的按钮取代，或作为对软键的补充。鉴于按钮设计方案的多样性，编程时仅使用面向操作软件外观设计的控件已无法再满足需要。比如：仅识别两个状态的切换控件可通过开关替代，该开关借助图标显示相应的状态并对轻击（Tap）或滑动（Flick）手势作出响应。

显示的图形能够对拖拽（Pan）手势作出响应。为此对 **imagebox** 控件进行相应扩展。

说明

以 Paint 标签输出的图形不会对手势作出响应。

除了解析器所提供的控件外，还可在脚本中对手指手势进行处理，以便在对话框中提供新的导航策略。手指手势可用于放大/缩小对话框内容。

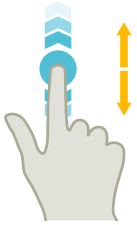
Easy XML 解析器支持下列手指手势：

手指手势



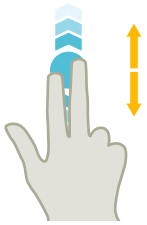
轻击 (Tap)

- 选择窗口
- 选择对象（例如 NC 程序段）
- 激活输入栏
  - 输入或覆盖值
  - 重新轻击以修改值



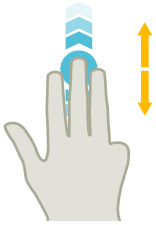
用 1 根手指垂直滑动 (Flick)

- 在列表中滚动（例如：程序、刀具、零点）
- 在文件中滚动（例如：NC 程序）



用 2 根手指垂直滑动 (Flick)

- 按页滚动列表（例如：NPV）
- 按页滚动文件（例如：NC 程序）



用 3 根手指垂直滑动 (Flick)

- 在列表开头或结尾滚动
- 在文件开头或结尾滚动



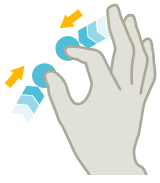
用 1 根手指水平滑动 (Flick)

- 在多栏列表中滚动



放大 (Spread)

- 放大图形内容（例如：模拟、模具制造视图）



缩小 (Pinch)

- 缩小图形内容（例如：模拟、模具制造视图）



用 1 根手指移动 (Pan)

- 移动图形内容（例如：模拟、模具制造视图）
- 移动列表内容

使用标签 **gesture_event** 和变量 **$gestureinfo** 来操纵手指手势。其实现针对经解码之手指手势的程序动作。

2.12.2 手指手势编程

标签 gesture_event

说明
仅当操作面板支持多点触控时，才执行此标签。

若操作软件识别出手指手势，则解析器会在变量 **$gestureinfo** 中提供手势信息，并执行标签 **gesture_event**。该变量具有数据类型 **GestureInfoStruct**，并包含以下属性：

属性	值	含义
type	3	移动手势
	4	缩小手势
	5	轻击手势
flag	1	缩放系数已改变
	2	旋转角已改变
state	1	已启动
	2	已更新
	3	已结束
item_data		生效控件的识别
	-1	手势未分配给控件
	!= -1	手势已在控件中执行，创建时已将该值分配给此控件。
point		手势的位置
	point.x	手势的 X 位置
	point.y	手势的 Y 位置
delta		手势的开始与当前事件之间的差
	<缩放差>	在缩放系数改变时
	<角度差>	在旋转角改变时

这些属性在文件 **struct_def.xml** 中预定义。

更多信息参见章节“创建 struct_def.xml 文件 (页 142)”

2.12.3 图形的手势控制

控制变量 imagebox

针对图形的手势控制，在控制变量 **Imagebox** 中有三个扩展：

属性	含义/特性
rotationangle	此属性值给出显示图形时采用的角度。
setrotationmode	属性值 **true** 允许对捏合（Pinch）手势的处理
setzoommode	属性值 **true** 允许在显示区域中放大/缩小以及移动图形。缺省状态下，此属性被设置为 **false**。

句法

```
<property rotationangle="角度" />
<property setrotationmode="true/false" />
<property setzoommode="true/false" />
```

示例：旋转位图

将位图顺时针旋转 30.5 度：

```
<let name="image_box_pict_name" type="string" >pic1.bmp</let>
…
…
…
 <control name="image_view" xpos="100" ypos="123" width="300"
height="200" fieldtype="imagebox" refvar="image_box_pict_name"
hotlink="true" >
 <property item_data="1000" />
 <property rotationangle="30.5" />
 <property setrotationmode ="true" />
```

```
</control>
```



示例：放大位图

允许位图的放大：

```
<control name="image_view" xpos="100" ypos="123" width="300"
height="200" fieldtype="imagebox" refvar="image_box_pict_name"
hotlink="true" >
 <property item_data="1000" />
 <property setzoommode="true" />
 </control>
```



控制功能 imageboxset

还可通过 **control.imageboxset** 功能设置特性。

可供使用的指令如下：

指令	含义/特性
SetRotationAngle	使图形旋转在 **lparam1** 中给定的角度。
SetRotationMode	**lparam1** 的值定义对捏合（Pinch）手势的与旋转有关的分析。 值等于 1 - 手势由控制项处理
SetZoomMode	若 **lparam1** 的值等于 1，则对于用于放大/缩小以及移动的捏合（Pinch）手势进行分析。

2.12.4 手势处理

标签 message

若缩放系数大于 1.0，则在显示区域中移动图片。在系数为 1.0 的情况下，例如可使用这个手指手势来在图片的列表中翻页。在此情形下，分析程序将一个可在标签 **message** 中分析的消息发送至表格。

消息参数 **$message_par1** 包含控制项的 Item_Data 值。消息参数 **$message_par2** 提示手势的运动方向。

$message_par2 可取下列值：

- -1，向左滚动
- 1，向右滚动
- -2，向上滚动
- 2，向下滚动

示例

```
…
…
<let name="image_box_pict_name" type="string" ></let>
<let name="pictindex" />
<let name="image_box_list" type="string" dim="4">
 "pic1.bmp",
 "pic2.bmp",
 "pic3.bmp",
 "pic4.bmp",
</let>

…
…
<control name="image_view" xpos="100" ypos="123" width="300"
height="200" fieldtype="imagebox" refvar="image_box_pict_name"
hotlink="true" >
 <property item_data="1000" />
 <property setzoommode="false" />
</control>
…
…
…
<!--
 -1，向左滚动
1，向右滚动
-2，向上滚动
2，向下滚动
```

```
 -->
<message>
 <if condition="$message_par1 == 1000">
  <then>

   <switch>
    <condition>$message_par2</condition>
    <case value="1">
     <op>
      pictindex = pictindex+1;
     </op>
     <if condition="pictindex > 3">
      <then>
       <op>
        pictindex = 0;
       </op>
      </then>
     </if>
    </case>
    <case value="-1">
     <op>
      pictindex = pictindex-1;
     </op>
     <if condition="pictindex < 0">
      <then>
       <op>
        pictindex = 3;
       </op>
      </then>
     </if>
    </case>
   </switch>
    <op>
     image_box_pict_name = image_box_list[$pictindex];
    </op>
   </then>
  </if>
 </message>
…
…
```

2.13 自定义按钮

按钮作为动作按钮或选择按钮嵌入表格。

2.13.1 下压按钮

标签 property

下压按钮是一个可用作按键或开关（带锁定功能的按键）的控制元件。可通过属性定义以“软键外观”风格以及以用户专用风格来设计按钮。相关的属性需要通过标签 **property** 定义。

可通过属性 **color_fg**、**color_bk**、**color_fg_pressed** 和 **color_bk_pressed** 修改前景色和背景色。

状态**按下/锁定**或者**松开**通过值一或零表达。为对按键的状态变化进行分析，需要指定处理程序功能。在 Control 标签中通过属性 **function** 指定处理程序功能。

可通过读取控制项变量或者分配的参考变量来确定开关的状态。

在触控操作中，当手指手势“松开”结束后，才提交手指手势“按下”和“松开”。

句法

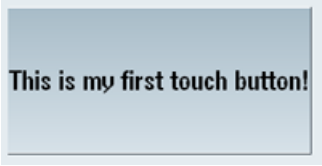
```
<control name="name" xpos="x position" ypos="y position"
fieldtype="pushbutton" >
 <caption></caption>
</control>
```

或者采用处理程序功能

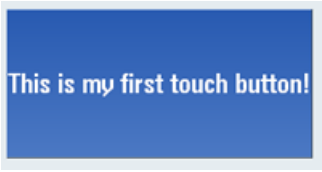
```
<control name="name" xpos="x position" ypos="y position"
fieldtype="pushbutton" function="button_handler" hotlink="true" >
 <caption></caption>
</control>
…
…
…
 <function_body name="button_handler">
…
…
…
 </function_body>
```

或

```
<control name="name" xpos="x position" ypos="y position"
fieldtype="pushbutton" refvar="button_state" hotlink="true" >
 <caption></caption>

 <property checkable="true" />
</control>
```



关闭



接通，锁定

示例

```
<control name="name" xpos="x position" ypos="y position"
fieldtype="pushbutton" refvar="button_state" hotlink="true"
color_fg="#ff00ff" color_bk="#000eee">
 <property checkable="true" />
  <caption></caption>
 <property picture="sk_circ_grind_cg.png" alignment="left"
TextAlignedToPicture="true" scaled="true"/>
</control>
```



2.13.2 按钮的功能

2.13.2.1 下压按钮的子标签

标签 caption

借助标签 **caption**，编程人员定义待为未按下状态显示的按钮文本。在缺省设置下，文本居中显示。可通过属性 **alignment** 修改文本的对齐。可给定下列属性值：

属性值	对齐
left	左对齐
right	右对齐
top	上
bottom	下
center	居中

若需为按下状态显示另一文本，则需要编写具有属性 **pressed** 和值 **true** 的第二 Caption 指令。

句法

居中显示

```
<control name="name" xpos="x position" ypos="y position"
fieldtype="pushbutton" >
 <caption>Button text</caption>
</control>
```



左对齐显示

```
<control name="name" xpos="x position" ypos="y position"
fieldtype="pushbutton" >
 <caption alignment= "left">Button text</caption>
</control>
```



按下显示

```
<control name="name" xpos="x position" ypos="y position"
fieldtype="pushbutton" >
 <caption pressed="true">Button text pressed</caption>

</control>
```

2.13.2.2 下压按钮的属性

通过标签 **property** 为控制项分配附加属性。

定义按键属性

借助属性 **checkable** 能够将按钮用作开关。为此将该属性的值设置为 **true**。

句法

```
<control name="name" xpos="x position" ypos="y position"
fieldtype="pushbutton" >
 <property checkable="true/false" />
</control>
```

禁用按键

属性 **disabled** 控制按钮的可操作性。若值为 **true**，则不对操作进行处理。

因分配的参考变量所引起的状态变化会导致按钮的更新。

句法

```
<property disabled="true/false" />
```

示例

```
<control name="name" xpos="x position" ypos="y position"
fieldtype="pushbutton" >
 <caption alignment="left">Button text</caption>
 <property disabled="true " />
</control>
```

分配图标

通过指定专有的图标或按键颜色，能够对预定义的设计进行修改。针对状态“未按下”、“按下”和“禁用”，可分别为按钮分配一个图标。若未针对状态“按下”或“禁用”进行分配，则会显示针对“未按下”状态的图标。

其他属性对各图标的对齐、缩放以及透明度加以控制。

属性	含义/特性
Picture	前景中的图标 针对“未按下”状态的图标的名称
PicturePressed	前景中的图标 针对“按下”状态的图标的名称

属性	含义/特性
PictureDisabled	前景中的图标 针对“禁用”状态的图标的名称
BackgroundPicture	背景中的图标 针对“未按下”状态的图标的名称
BackgroundPicturePressed	背景中的图标 针对“按下”状态的图标的名称
BackgroundPictureDisabled	背景中的图标 针对“禁用”状态的图标的名称

属性 alignment

在标签中可指定其他属性，其值基于列举的图标分配的 **alignment**。

属性值	对齐
left	左对齐
right	右对齐
top	上
bottom	下
center	居中
stretch	背景图标： 若使用值 **stretch**，分析程序会将图标缩放至按钮的矩形范围。 借助通过属性 **transparent** 指定透明色，能够产生按钮的任意轮廓。

2.13.2.3 下压按钮的控制项变量

通过为下面列出的属性变量赋予新的值，能够追加改变按钮的属性。在运算指令中通过给定控制项名称以及跟随于其后控制项变量名称来进行赋值。两个名称必须通过点相互分隔。

句法

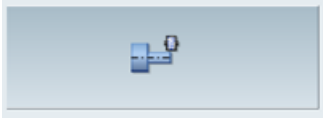
<控制项名称>.<控制项变量>

控制项变量	含义/特性
backgroundpicture	变量类型：字符串 背景图标的名称
backgroundpicturedisabled	变量类型：字符串 针对禁用状态的背景图标的名称
backgroundpicturepressed	变量类型：字符串 针对按下状态的背景图标的名称
picture	变量类型：字符串 前景图标的名称
picturedisabled	变量类型：字符串 针对禁用状态的前景图标的名称
picturepressed	变量类型：字符串 针对按下状态的前景图标的名称
picture_textalignedtopicture	变量类型：Bool 值： 1 = true 0 = false 在图标上将按钮文本对齐
picturedisabled_textalignedtopicture	变量类型：Bool 值： 1 = true 0 = false 在“禁用状态”图标上将按钮文本对齐

控制项变量	含义/特性
picturepressed_textalignedtopicture	变量类型：Bool 值： 1 = true 0 = false 在“按下状态”图标上将按钮文本对齐
backgroundpicture_keepaspectratio	变量类型：Bool 值： 1 = true 0 = false 保持或忽略背景图标的长宽比
backgroundpicturedisabled_keepaspectratio	变量类型：Bool 值： 1 = true 0 = false 保持或忽略“禁用状态”背景图标的长宽比
backgroundpicturepressed_keepaspectratio	变量类型：Bool 值： 1 = true 0 = false 保持或忽略“按下状态”背景图标的长宽比
picture_keepaspectratio	变量类型：Bool 值： 1 = true 0 = false 保持或忽略图标的长宽比
picturedisabled_keepaspectratio	变量类型：Bool 值： 1 = true 0 = false 保持或忽略“禁用状态”图标的长宽比

控制项变量	含义/特性
picturepressed_keepaspectratio	变量类型：Bool 值： 1 = true 0 = false 保持或忽略“按下状态”图标的长宽比
backgroundpicture_scaled	变量类型：Bool 值： 1 = true 0 = false 根据按钮的尺寸调整背景图标
backgroundpicturedisabled_scaled	变量类型：Bool 值： 1 = true 0 = false 根据按钮的尺寸调整“禁用状态”背景图标
backgroundpicturepressed_scaled	变量类型：Bool 值： 1 = true 0 = false 根据按钮的尺寸调整“按下状态”背景图标
picture_scaled	变量类型：Bool 值： 1 = true 0 = false 根据按钮的尺寸调整图标
picturedisabled_scaled	变量类型：Bool 值： 1 = true 0 = false 根据按钮的尺寸调整“禁用状态”图标
picturepressed_scaled	变量类型：Bool 根据按钮的尺寸调整“按下状态”图标

控制项变量	含义/特性
backpicture_alignment	变量类型：字符串 参见属性 alignment
backgroundpicturedisabled_alignment	
backgroundpicturepressed_alignment	
picture_alignment	
picturedisabled_alignment	
picturepressed_alignment	
caption	变量类型：字符串 “未按下状态”按钮文本
captionpressed	变量类型：字符串 “按下状态”按钮文本
caption_alignment	变量类型：字符串 参见属性 alignment
captionpressed_alignment	
captiondisabled_alignment	
disabled	变量类型：Bool 值： 1 = true - 下压按钮被禁用 0 = false - 下压按钮可操作



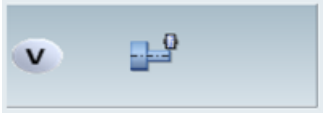
Picture



PicturePressed



PictureDisabled



BackgroundPicture + Picture

属性 TextAlignedToPicture

若此属性的值为 **true**，则基于图标将文本对齐。

```
<property picture="sk_circ_grind_cg.png" alignment="left"
TextAlignedToPicture="true" />
```



属性 scaled

若此属性的值为 **true**，则根据表格中按钮的尺寸调整图片的尺寸，使得按钮的高度或宽度的最多 50 % 可供图形使用。

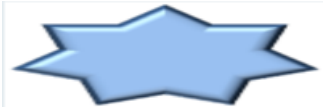
```
<property picture="sk_circ_grind_cg.png" alignment="left"
TextAlignedToPicture="true" scaled="true"/>
```



属性 stretch（仅对背景图标生效）

若此属性的值为 **true**，则根据按钮的尺寸调整图片的尺寸。

```
<property backgroundpicture="f:\appl\pbutton.png"
alignment="stretch" />
```



```
<property backgroundpicture="f:\appl\pbutton.png"
alignment="stretch" transparent="#ffffff"/>
```



```
<caption alignment="left" >This is my first touch button!</
caption>
 <property picture="sk_circ_grind_cg.png" alignment="left"
TextAlignedToPicture="true" scaled="true"/>
 <property backgroundpicture="pbutton.png" alignment="stretch"
transparent="#ffffff"/>
```



2.13.3 开关 on/off

开关控制项是一种图形元件，其借助一个图标指示两个状态中的一个。此控制项可以通过触控或鼠标来进行操作。

进入的状态可以存储在参考变量中，或者，可以在分配给该控制项的功能中确定状态切换。在 Control 标签中通过属性 **function** 进行分配。

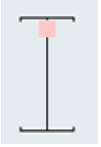
属性 **alignment** 实现按钮的垂直或水平对齐。

此控制项不具有缺省设计。若未为该控制项分配图标，则仅显示边界框和开关位置（通过点显示）。

句法

```
<control name="name" xpos = "x position" ypos="y position"
width="width" height="height" fieldtype="switch" alignment="hr/
vr">
 <property left_position="value" />
 <property right_position="value" />
</control>
```

属性 alignment



开关垂直：值 **vr**



开关水平：值 **hr**

2.13.4 开关的功能

2.13.4.1 开关的属性

标签 property

可通过标签 **property** 为该控制项分配下列开关位置：

属性	含义/特性
left_position	在开关按钮向左运动的情况下，为控制项变量分配这个属性的值。
right_position	在开关按钮向右运动的情况下，为控制项变量分配这个属性的值。
upper_position	在开关按钮向上运动的情况下，为控制项变量分配这个属性的值。
lower_position	在开关按钮向下运动的情况下，为控制项变量分配这个属性的值。
disabled	此属性控制开关的可操作性。 若值为 **true**，则不对操作进行分析。 因分配的参考变量所引起的状态变化会导致开关状态的更新。

需要通过给定其他属性来定义最终的开关设计。这些属性必须与属性 **left_position**、**right_position**、**upper_position** 或 **lower_position** 一起定义。

借助通过属性 **transparent** 指定透明色，能够产生开关的任意轮廓。

属性	含义/特性
picture	前景中的图标，针对“未按下”状态的图标的名称
pictureDisabled	前景中的图标，针对“禁用”状态的图标的名称
transparent	该属性值包含透明色的颜色值。 此颜色值被用于所有与开关对应的图标。
caption	该属性值包含归属于开关位置的文本。此文本会显示在元件上，并受开关尺寸限制。

2.13.4.2 开关的控制项变量

通过为下面列出的控制项变量赋予新的值，能够追加改变开关的属性。在运算指令中通过给定控制项名称以及跟随于其后的控制项变量名称来进行赋值。两个名称必须通过点相互分隔。

句法

<控制项名称>.<控制项变量>

控制项变量	含义/特性
Left_position	变量类型：Int 在开关按钮向左运动的情况下，为控制项变量分配这个属性的值。
Right_position	变量类型：Int 在开关按钮向右运动的情况下，为控制项变量分配这个属性的值。
Lower_position	变量类型：Int 在开关按钮向下运动的情况下，为控制项变量分配这个属性的值。
Upper_position	变量类型：Int 在开关按钮向上运动的情况下，为控制项变量分配这个属性的值。
Picture_left_position	变量类型：字符串 针对左侧位置的图标的名称
Picture_right_position	变量类型：字符串 针对右侧位置的图标的名称
Picture_upper_position	变量类型：字符串 针对上部位置的图标的名称
Picture_lower_position	变量类型：字符串 针对下部位置的图标的名称
Picturedisabled_left_position	变量类型：字符串 针对左侧位置“禁用状态”的图标的名称
Picturedisabled_right_position	变量类型：针对右侧位置“禁用状态”的图标的名称

控制项变量	含义/特性
Picturedisabled_upper_position	变量类型：字符串 针对上部位置“禁用状态”的图标的名称
Picturedisabled_lower_position	变量类型：字符串 针对下部位置“禁用状态”的图标的名称
Caption_left_position	变量类型：字符串 左侧位置上的文本
Caption_right_position	变量类型：字符串 右侧位置上的文本
Caption_upper_position	变量类型：字符串 上部位置上的文本
Caption_lower_position	变量类型：字符串 下部位置上的文本
disabled	变量类型：Bool 值： 1 = true - 开关被禁用 0 = false - 开关可操作

水平显示

```
<control name="name" xpos="x position" ypos="y position"
width="width" height="height" fieldtype="switch" alignment="hr">
 <property left_position="value" picture="name" />
 <property right_position="value" picture="name" />
</control>
```

垂直显示

```
<control name="name" xpos="x position" ypos="y position"
width="width" height="height" fieldtype="switch" alignment ="vr">
 <property upper_position="value" picture="name" />
 <property lower_position="value" picture="name" />
</control>
```

或者分配处理程序功能

```
<control name="name" xpos="x position" ypos="y position"
width="width" height="height" fieldtype="switch"
function="switch_handler" hotlink="true" alignment ="vr">
</control>
```

示例



icon_left.bmp



icon_right.bmp

```
<let name="switch_state" />
<control name="main_switch" xpos="100" ypos="53" width="40"
height="30" fieldtype="switch" refvar="switch_state"
hotlink="true" alignment="hr">
 <property left_position="10" picture="icon_left.bmp" />
 <property right_position="11" picture="icon_right.bmp" />
</control>
```

通过控制项变量分配属性

```
<control name="main_switch" xpos = "450" ypos="340" width="20"
height="46" fieldtype="switch" refvar="switch_statebf"
hotlink="true" alignment="vr">
 <property upper_position="1" />
 <property lower_position="0" />
</control>
…
…
…
<op>
  main_switch.transparent = #ffffff;
  main_switch.picture_upper_position = _T"switch_up.png";
  main_switch.picture_lower_position = _T"switch_bottom.png";
  main_switch.disable= 1;
</op>
```

2.13.5 单选按钮

借助单选按钮能够从多个选项中选择一个。必须为每个选项创建一个按钮。所有归属于表格、分组框或者滚动区域的单选按钮相互交互作用，故仅一个按钮被标记为选中。按钮状态被传输至控制项变量。

句法

```
<control name="name" xpos="x position" ypos="y position"
width="width" height="height" fieldtype="radiobutton"  >
```

```
 <caption>button text</caption>
</control>
```

2.13.6 复选框

借助复选框能够选择多个选项。必须为每个选项创建一个按钮。复选框的状态被传输至控制项变量。

句法

```
<control name="name" xpos="x position" ypos="y position"
width="width" height="height" fieldtype="checkbox"  >
 <caption>button text</caption>
</control>
```

2.13.7 分组框

就外观而言，分组框将一组控制项包围，具有框架和标题。其用于将逻辑关联的控制项分组，这些控制项仅需要在该组内进行交互。嵌入的控制项的坐标以标题行下方的左上角为基准。

所有归属于组的控制项被作为子控制项嵌入 Control 标签。

在给定嵌入的控制项名称和值 **Item_Data** 时需要注意的是，就所有归属于表格的控制项而言，这些设定必须具有唯一性。

通过标签 **caption** 定义分组框的标题。

句法

```
<control name="name" xpos="x position" ypos="y position"
width="width" height="height" fieldtype="groupbox">
 <caption>caption text</caption>
  <control>…</control>
…
…
…
</control>
```

2.13.8 滚动区域

滚动区域用于显示定义的区域内的控制项。嵌入的控制项的坐标以滚动区域的左上角为基准。若在可见区域外创建控制项，则实现可见区域的移动。所有归属于区域的控制项必须作为子控制项嵌入 Control 标签。

在给定嵌入的控制项名称和值 **Item_Data** 时需要注意的是，就所有归属于表格的控制项而言，这些设定必须具有唯一性。

句法

```
<control name="name" xpos="x position" ypos="y position"
width="width" height="height" fieldtype="scrollarea">
  <control>…</control>
…
…
…
</control>
```

触控操作

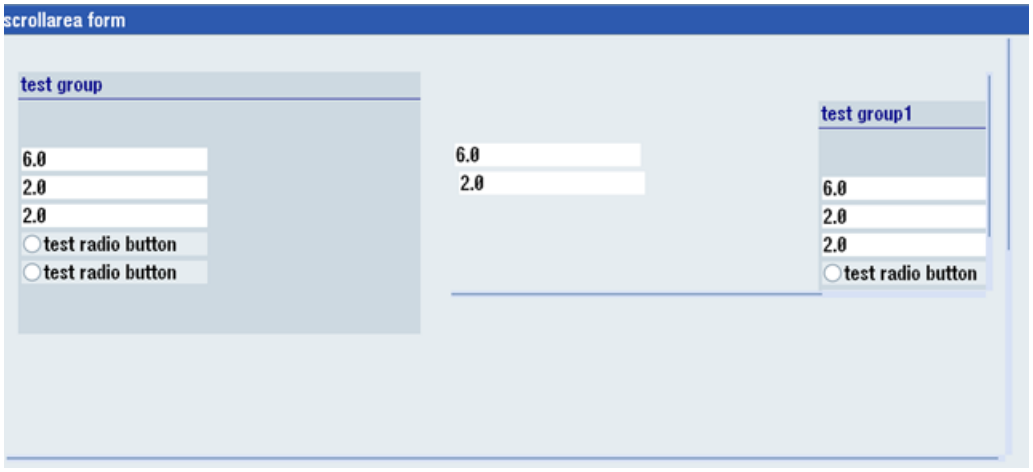
若将焦点置于当前不完全可见的控制项上，则使可见区域移动，直至能够完全显示该控制项。

轻击手势

在无法将此手势分配给嵌入的控制项的情况下，将此手势传输至脚本。

示例

嵌套的滚动区域



```
<let name="CB1" >1</let>
<let name="RB1" />

<form name="scrollarea_form">
 <init>
  <caption>scrollarea form</caption>
   <data_access type="true" />
    <control name= "c_scroll" xpos = "2" ypos="24" width="520"
height="300" fieldtype="scrollarea" >
    <control name= "c_group" xpos = "6" ypos="24" width="208"
height="186" fieldtype="groupbox" >
    <caption>test group</caption>
    <control name = "c18" xpos = "2" ypos = "54" refvar="nck/
Channel/Parameter/R[1]" hotlink="true" />
    <control name = "c19" xpos = "2" ypos = "74" refvar="nck/
Channel/Parameter/R[2]" hotlink="true" />
    <control name = "c20" xpos = "2" ypos = "94" refvar="nck/
Channel/Parameter/R[3]" hotlink="true" />
    <control name="radiobg" xpos = "2" ypos="114"
fieldtype="radiobutton" >
    <caption>test radio button</caption>
    <property backgroundpicture="f:\appl\cycle_my1.png" />
   </control>
   <control name="radiobg1" xpos = "2" ypos="134"
fieldtype="radiobutton" >
    <caption>test radio button</caption>
    <property backgroundpicture="f:\appl\cycle_my1.png" />
   </control>
  </control>
 <control name= "c_scroll_emb" xpos = "230" ypos="24" width="280"
height="160" fieldtype="scrollarea" >

 <control name = "c18emb" xpos = "2" ypos = "54" refvar="nck/
Channel/Parameter/R[1]" hotlink="true" />
 <control name = "c19emb" xpos = "4" ypos = "74" refvar="nck/
Channel/Parameter/R[2]" hotlink="true" />
 <control name = "c20emb" xpos = "3" ypos = "194" refvar="nck/
Channel/Parameter/R[3]" hotlink="true" />
 <control name= "c_group1" xpos = "190" ypos="24" width="208"
height="186" fieldtype="groupbox" >
 <caption>test group1</caption>
 <control name = "c18gemb" xpos = "2" ypos = "54" refvar="nck/
Channel/Parameter/R[1]" hotlink="true" />
 <control name = "c19gemb" xpos = "2" ypos = "74" refvar="nck/
Channel/Parameter/R[2]" hotlink="true" />
 <control name = "c20gemb" xpos = "2" ypos = "94" refvar="nck/
Channel/Parameter/R[3]" hotlink="true" />
 <control name="radiobggemb" xpos = "2" ypos="114"
fieldtype="radiobutton" >
  <caption>test radio button</caption>
  <property backgroundpicture="f:\appl\cycle_my1.png" />
</control>
```

```
<control name="radiobg1emb" xpos = "2" ypos="134"
fieldtype="radiobutton" >
 <caption>test radio button</caption>
  <property backgroundpicture="f:\appl\cycle_my1.png" />
 </control>
 </control>
 </control>
<control name = "c22" xpos = "2" ypos = "400" refvar="nck/Channel/
Parameter/R[12]" hotlink="true" />
 <control name="ChB1" xpos="208" ypos="430" width="50"
height="30" fieldtype="checkbox" refvar="PLC/M100.7" hotlink =
"true" color_bk="#00ffff">
  <caption>Check1</caption>
 </control>
  <control name="ChB2" xpos="208" ypos="460" width="58"
height="30" fieldtype="checkbox" refvar="PLC/M100.6" hotlink =
"true" color_bk="#00ffff">
  <caption>Check2</caption>
 </control>
  <control name="Radio1" xpos="208" ypos="490" width="40"
height="30" fieldtype="radiobutton" refvar="PLC/M100.0" hotlink =
"true" color_bk="#00ffff">
   <caption>Radio1</caption>
  </control>
   <control name="Radio2" xpos="208" ypos="520" width="45"
height="30" fieldtype="radiobutton" refvar="PLC/M100.1" hotlink =
"true" color_bk="#00ffff">
    <caption>Radio2</caption>
  </control>
   <control name="Radio3" xpos="208" ypos="550" width="50"
height="30" fieldtype="radiobutton" refvar="PLC/M100.2" hotlink =
"true" >
    <caption>Radio3</caption>
   </control>
    <control name="VChB1" xpos="8" ypos="430" width="50"
height="30" refvar="PLC/MB100" hotlink = "true"
color_bk="#00ffff"/>
    <control name="VChB2" xpos="8" ypos="465" width="53"
height="30" refvar="PLC/MB100" hotlink = "true"
color_bk="#00ffff"/>
   </control>
   <control name="ChB3" xpos="208" ypos="340" width="60"
height="30" fieldtype="checkbox" refvar="CB1" >
    <caption>Check3</caption>
   </control>
   </init>
  <timer>
  </timer>
</form>
```

2.14 Sidescreen 的使用

2.14.1 Sidescreen 中的 Easy XML

Easy XML 脚本语言同样可以用于在 Sidescreen 区域中创建对话框。对话框的显示由 Sidescreen 部件“页面”（**Page**）和“微件”（**Widget**）实现。对话框的集成由文件 **slsidescreen.ini** 实现。Sidescreen 部件会随着操作软件的启动、关闭而自动启动和关闭，也就是说，脚本解析器会随着 Sidescreen 部件的首次激活一次性加载对应的脚本。

Sidescreen 区域的尺寸由文件 **slsidescreen_<屏幕分辨率>.ini** 中的布局参数（layout）来确定。Easy XML 脚本不受该尺寸的影响，可以使用的宽度为 276 个像素，Easy XML 对话框可以使用的高度取决于使用的 Sidescreen 部件：页面型部件可以使用 768 个像素；微件型部件可以使用的高度由 Sidescreen 管理项决定，可以查看 **GETHMIResolution** 得知。

脚本解析器会查找一个可以激活某个“窗体”（form）或切换到另一个窗体的菜单。主菜单必须始终包含名称 **main**。Sidescreen 不支持软键标记，因此必须通过窗体的操作单元才能切换到另一个窗体。

脚本解析器不限制 Sidescreen 页面或 Sidescreen 微件的数量。

2.14.2 集成 Sidescreen 对话框

页面或微件的集成通过文件 **slsidescreen.ini** 实现。

页面应在段落 **[Sidescreen]** 中创建，每个页面的定义从关键字 **PAGE** 开始，后面跟一个连续编号的数字以及必要的一些属性。属性 **name** 确定页面的对象名称，属性 **implementation** 确定设计库和类名称，这两个属性要用逗号隔开。类型为 **SlSideScreenPage** 的页面可以集成 Sidescreen 元素，具体是通过 **ELEMENT** 条目实现的。创建一条元素行 **ELEMENTxxx** 需要遵循的规定和创建页面一样。如果希望页面上只包含由 Easy XML 设计的对话框，必须在属性 **implementation** 中输入 **slagmforms.SlEESideScreenPage** 。

微体可以添加到段落 **[Element_<name>]** 中的一个 Sidescreen 元素中。

创建一条微体行 **WIDGETxxx** 需要遵循的规定和创建页面一样。

需要激活由 Easy XML 设计的 Sidescreen 微体时，必须在属性 **implementation** 中输入 **slagmforms.SlEESideScreenWidget**。

语法

```
ELEMENT002= name:=<name>, implementation:=SlSideScreenElement

…
…

[Element_<name>]
WIDGET001= name:=<Widget Name>, implementation:=
slagmforms.SlEESideScreenWidget
```

默认设置中，微体的名称会用于组成主模块的名称。

示例

```
PAGE004= name:=sidescreen_proginfluence,
implementation:=slagmforms.SlEESideScreenPage
```

主模块的名称：sidescreen_proginfluence.xml

其他属性

在文件 **slsidescreen.ini** 的段落 **PAGE_<pagename>**、**ELEMENT_<elmentname>** 或 **WIDGET_<widgetname>** 中有更多属性用于页面或微体的设计。

以下属性可用于设计页面、元素或微体：

属性	含义/特性
TextId	文本标识符，和语言无关
TextFile	文本文件名称
TextContext	文本上下文 EASY_XML
Icon	图标名称

属性	含义/特性
maskPath	该属性确定要使用的主模块名称。
maskName	该属性确定要使用的主菜单名称。
focusable	该属性确定是否可以在该元素上放置输入焦点。

示例

```
[Page_sidescreen_proginfluence]
PROPERTY001= name:=focusable, type:=bool, value:="true"
Icon=sidescreen_proginfluence.png

[PAGE_sidescreen_proginfluence]
Icon= sidescreen_proginfluence.png
PROPERTY001= name:=focusable, type:=bool, value:="true"
PROPERTY002= name:=maskPath, type:=QString, value:="
sidescreen_proginfluence.xml"
PROPERTY003= name:=maskName, type:=QString, value:="main"
```

2.14.3 语言和文本管理

您可以为每个部件指定一份文本文件来管理和语言无关的文本。该文本文件的名称由部件名称、语言版本和后缀名".ts"构成。

文本上下文请使用 **EASY_XML**。

语法

<name>_<language>.ts

示例

```
PAGE004= name:=sidescreen_proginfluence,
implementation:=slagmforms.SlEESideScreenPage
```

文本文件名称：sidescreen_proginfluence**_eng.ts**

没有指定文本文件时，解析器会在标准文本文件 **oem_xml_screens_xxx.ts** 中查找文本标识符。

2.14.4 Sidescreen 部件

2.14.4.1 Sidescreen 元素

页面关联了标准设计时，便可以为该页面指定元素。为此要创建段落 **[Page_<page_name>]**，然后列出所有页面下的元素。

示例

```
[Sidescreen]
PAGE001= name:=<page_name>, implementation:=SlSideScreenPage

[Page_<page_name>]

ELEMENT<nummer>= name:=<Element name>,
implementation:=SlSideScreenElement
```

每个元素都需要用一个单独的段落 **[Element_<element name>]** 来定义，在该段落中列出属性以及对应的微体。

```
[Element_<element_name>]
WIDGET001= name:=<widget_name>, implementation:=
<Implementierung>
```

2.14.4.2 Sidescreen 微体

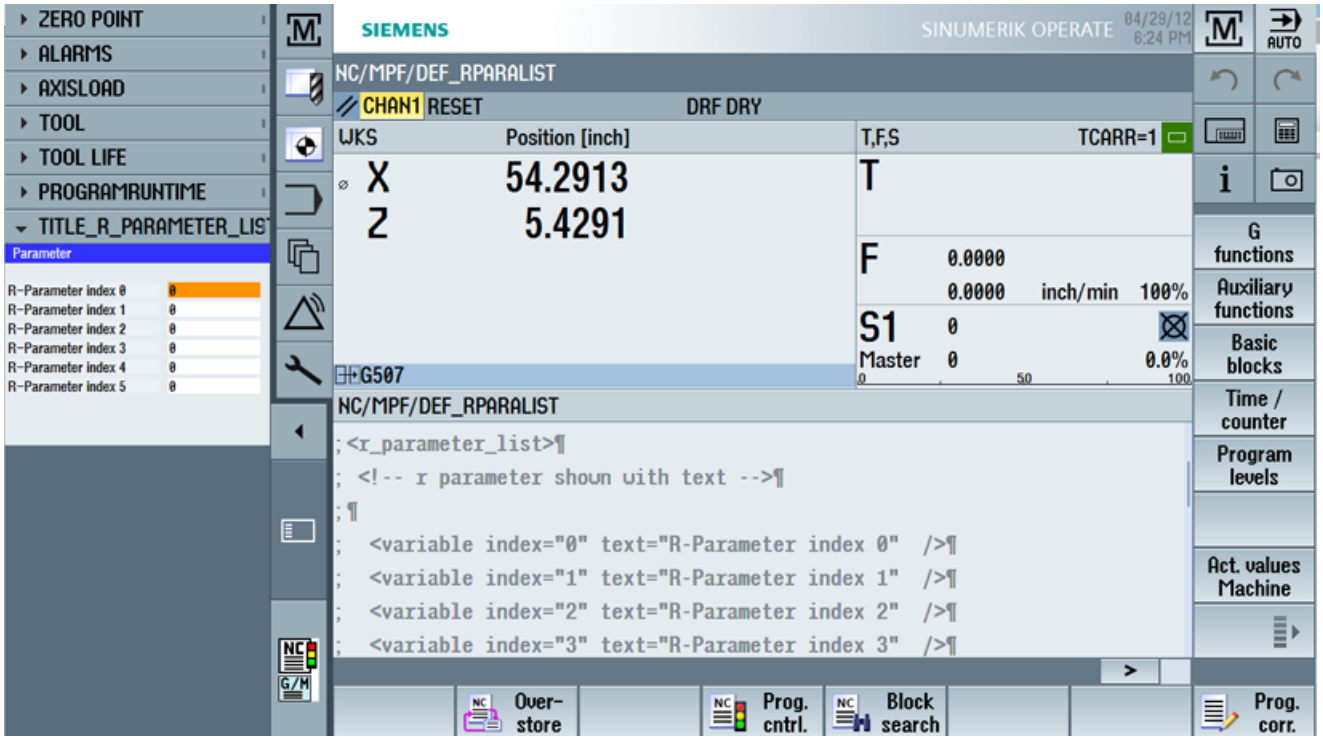
“微体”（Widget）是 Sidescreen 管理项指定的、用于显示设计的窗体的某个区域。默认设置中，微体没有输入焦点，因此无法编辑各字段。该情况可以通过属性 **focusable** 修改。随着微体的打开，解析器会一同打开属于主菜单的窗体。在该窗体内，可以通过导航指令转入其他菜单。

示例

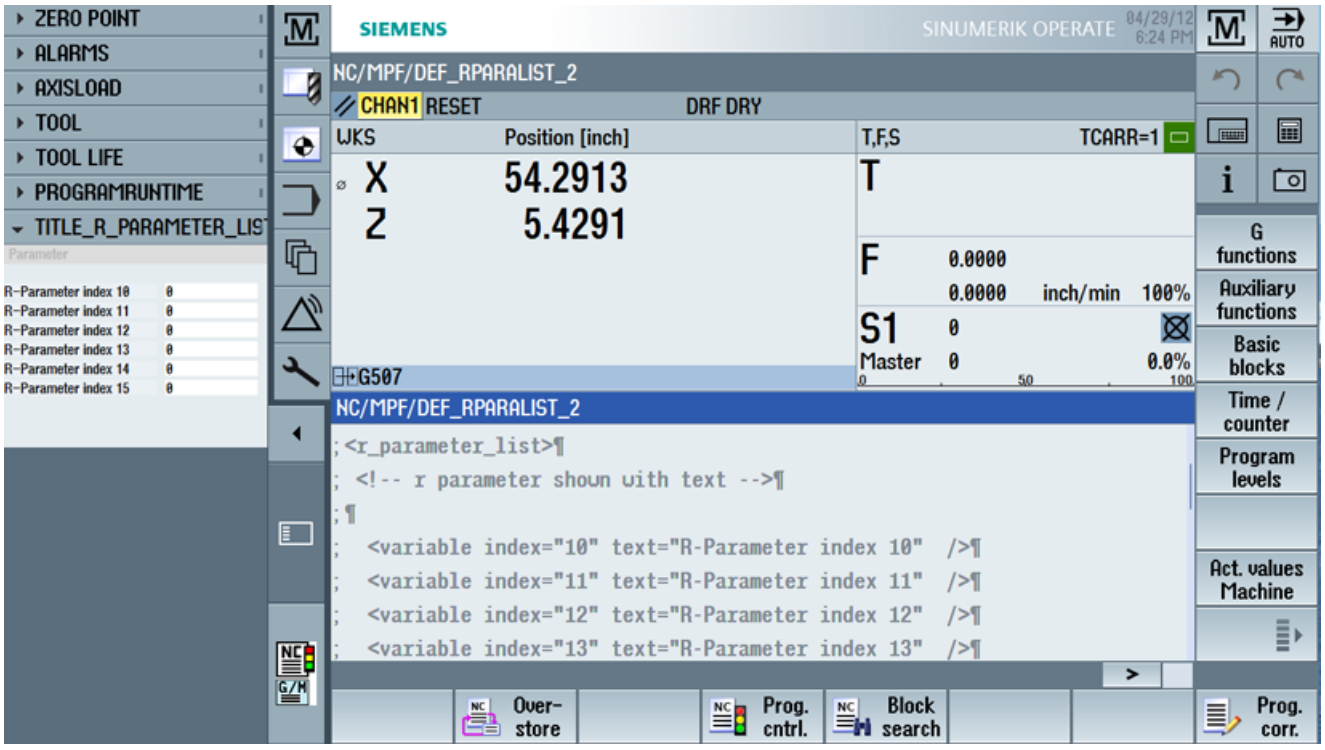
Easy XML 脚本在当前的零件程序内查找对于微体数据内容的说明。具体的表现形式有：脚本查找需要显示的、R 参数列表的说明。每个参数都可以有一条说明文本。

Toolbox 示例

Custom Screen Sample\
side_screen_examples\sidescreenwidget\displayRparameter\rparameter_widget.xml



选择了其他一个零件程序时，微体会自动更新。


ZERO POINT
ALARMS
AXISLOAD
TOOL
TOOL LIFE
PROGRAMRUNTIME
TITLE_R_PARAMETER_LIS
Parameter
R-Parameter index 10 0
R-Parameter index 11 0
R-Parameter index 12 0
R-Parameter index 13 0
R-Parameter index 14 0
R-Parameter index 15 0
SIEMENS
SINUMERIK OPERATE
04/29/12
6:24 PM
AUTO
NC/MPF/DEF_RPARALIST_2
CHAN1 RESET
DRF DRY
WKS
Position [inch]
T,F,S
TCARR=1
X 54.2913
Z 5.4291
T
F 0.0000
0.0000 inch/min 100%
S1 0
Master 0 0.0%
G507
G functions
Auxiliary functions
Basic blocks
Time / counter
Program levels
Act. values Machine
NC/MPF/DEF_RPARALIST_2
;<r_parameter_list>¶
; <!-- r parameter shown with text -->¶
;¶
; <variable index="10" text="R-Parameter index 10" />¶
; <variable index="11" text="R-Parameter index 11" />¶
; <variable index="12" text="R-Parameter index 12" />¶
; <variable index="13" text="R-Parameter index 13" />¶
Over-store
Prog. cntrl.
Block search
Prog. corr.

2.14.4.3 Sidescreen 页面

通过设计 **slagmforms.SIEESideScreenPage** 启动的“页面”（page）指供窗体使用的整个 Sidescreen 区域。它没有标题行，因此无法用指令 Caption 来设计标题。默认设置中，页面没有输入焦点，因此无法编辑各字段。该情况可以通过属性 **focusable** 修改。随着页面的打开，解析器会一同打开属于主菜单的窗体。在该窗体内，可以通过导航指令转入其他菜单。

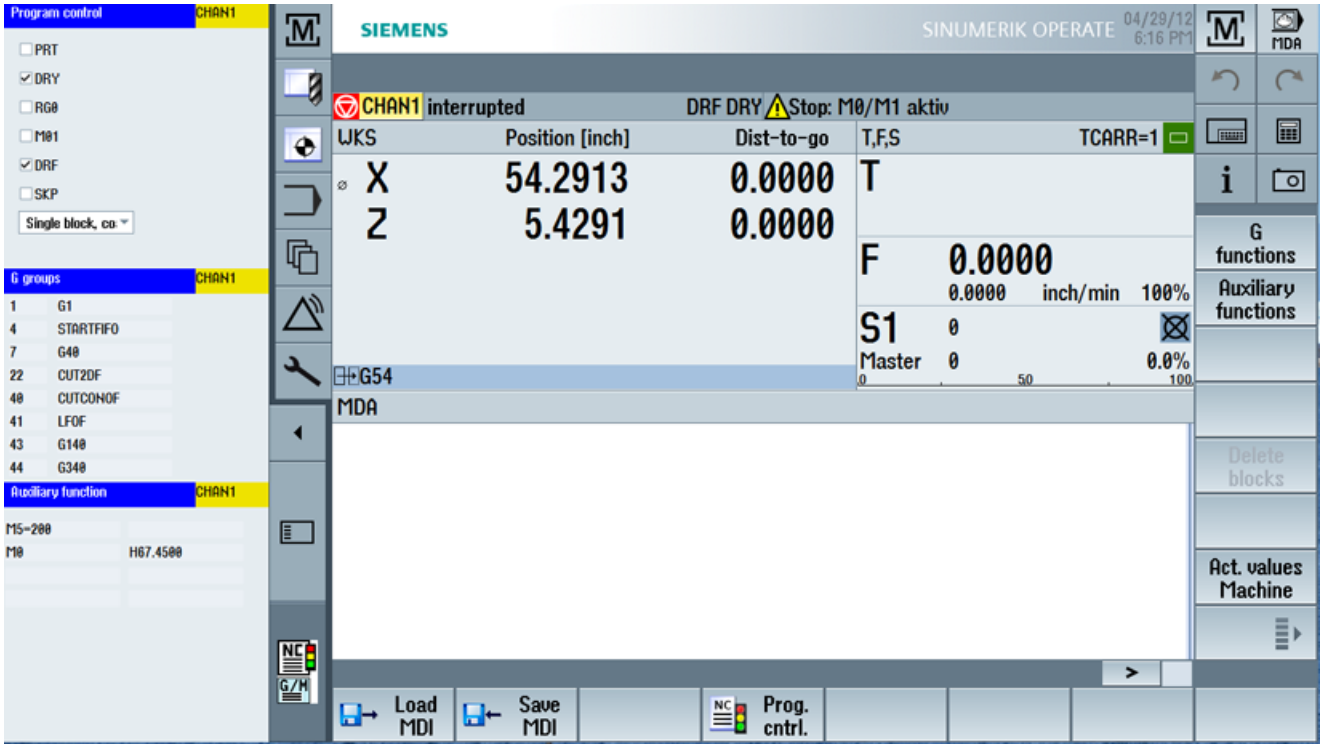
示例

Sidescreen 页面要设计为一幅只显示一个 Easy XML 窗体的页面。

本例中，Sidescreen 区域中显示的是附加信息。

Toolbox 示例

Custom Screen Sample\side_screen_examples\sidecreenpage
\sidescreen_proginfluence.xml




```
[Sidescreen]
PROPERTY001= name:=minimizable, type:=bool, value:="true"
PROPERTY002= name:=buttonBarVisible, type:=bool, value:="true"
```

```
PAGE001= name:=WidgetsPage, implementation:=SlSideScreenPage
PAGE004= name:=sidescreen_proginfluence,
implementation:=slagmforms.SlEESideScreenPage

[Page_sidescreen_proginfluence]
PROPERTY001= name:=focusable, type:=bool, value:="true"
Icon=sidescreen_proginfluence.png
```

文本文件：sidescreen_proginfluence_deu.ts

2.15 Display Manager 应用

2.15.1 Display Manager 中的 Easy XML

Easy XML 函数可以对话框 **SISideScreenPage** 为基础接收至 Display Manager 配置中。该对话框可以根据空余的空间显示一个或多个 Sidescreen 页面。多个页面会以分列形式并排显示。每一列占据相同的宽度。您需要配置待显示页面的数量及其内容。每个页面都有自己的一份配置文件。这些配置文件的名称在文件 **systemconfiguration.ini** 对话框配置段落中的指令行参数 **cmdline** 中确定。

第一列页面的配置文件的名称跟在参数名称**–sidescreen1** 后面，第二列页面的配置文件的名称跟在参数名称**–sidescreen2** 后面，以此类推。

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。也就是说，只有在 HMI 重启后，脚本更改才会生效。

2.15.2 将 Customer Area 整合至 Display Manager 菜单

在 Customer 区域调用 Easy XML 解析器的前提条件是已有一个 Area 已经链接至 Easy XML 对话框。默认情况下在文件 **systemconfiguration.ini** 中指定 Area 定义 **AREA111**。

句法

```
AREA0111= name:=CustomXML, dialog:=SlEECustomDialog,
panel:=SlHdStdHeaderPanel
```

出厂时该区域是不可见的。可将文件 **slamconfig.ini** 中的可视化标签 **Visible** 值由 **false** 改为 **true**，便可显示该区域。

```
[CustomXML]
TextId=SL_AM_CUSTOM
SidescreenTextId=SL_SIDESCREEN_CUSTOM
TextFile=slam
TextContext=SlAmAreaMenu
SoftkeyPosition=12
Visible=false -> true
```

示例

通过与 Display Manager 分辨率相关的 INI 文件成功将 Customer 区域集成至 Display Manager 菜单。



Operate buttons 下的每一个文件中都需要创建一个菜单项，包含至 Area **CustomXML** 的引用信息：

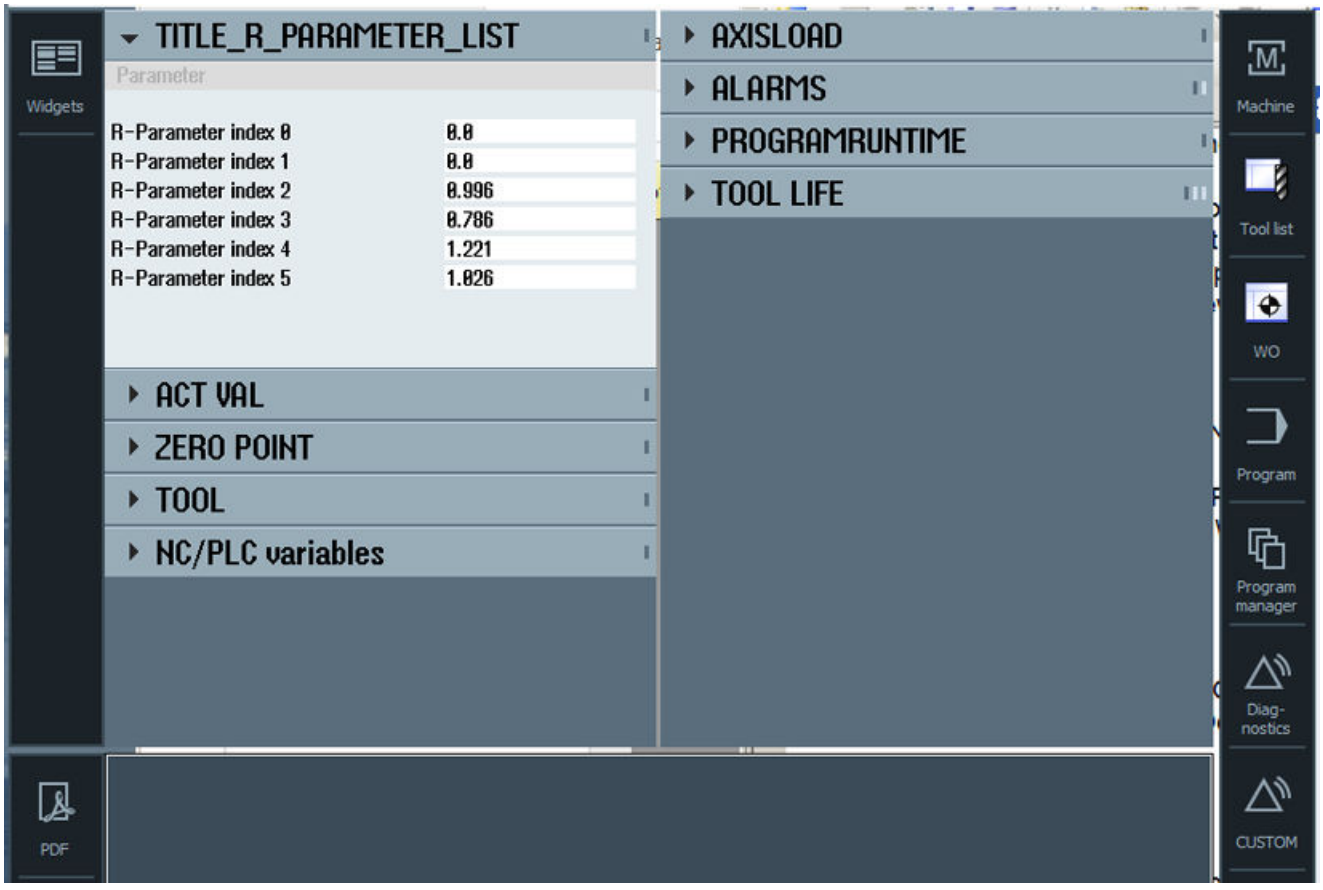
```
MENUITEM107= name:=mieasyXML, menuItemStyle:=misMenu,
onClicked:="hidePopup(); sendCmd(OPERATE, CustomXML);
showApp(defaultFrame, OPERATE)", image:=<bild.png>,
textID:= <textid>
```

再次将该菜单元素分配至 Display Manager 的菜单中：

```
MENU020= name:=mOperate3, menuStyle:=msMenu,
defaultFrame:=fOperate3, items:="miMachine, miToolList,
miWorkOffset, miProgramEdit, miProgramManager, miDiagnosis,
mieasyXML, miStretch, miMaximizeOperate3"
```

2.15.3 Display Manager 小部件

Display Manager 中的小部件可按照 Sidescreen 小部件进行创建和管理。但是，“Sidescreen 小部件”的说明仍在 Display Manager 的配置文件中完成，例如：INI 文件 **sldmwidgets*.ini**。通过 **systemconfiguration.ini** 创建至该文件的引用。



包含文件 **rparameter_widget.xml** 的示例说明请见章节“Sidescreen 微体 (页 257)”。

示例

systemconfiguration.ini

```
DLG109= name:=SlWidgetsApp,
implementation:=sldmsidescreenapp.SlSideScreenDialog,
process:=SlHmiHost1, preload:=false, cmdline:="-sidescreen1
sldmwidgets1.ini -sidescreen2 sldmwidgets2.ini -spacing 3"
```

将对话框名称分配至框架：

```
FRAME020= name:=fTop, x:=66, y:=66, width:=760, height:=505,
app:=SlWidgetsApp
```

Display 中定义的框架：

```
DISPLAY001= name:=d3, frames:="fHeader, fOperate3, fMenuOperate3,
fTop, fMenuTop, fBottom, fMenuBottom"
```

sldmwidgets1.ini

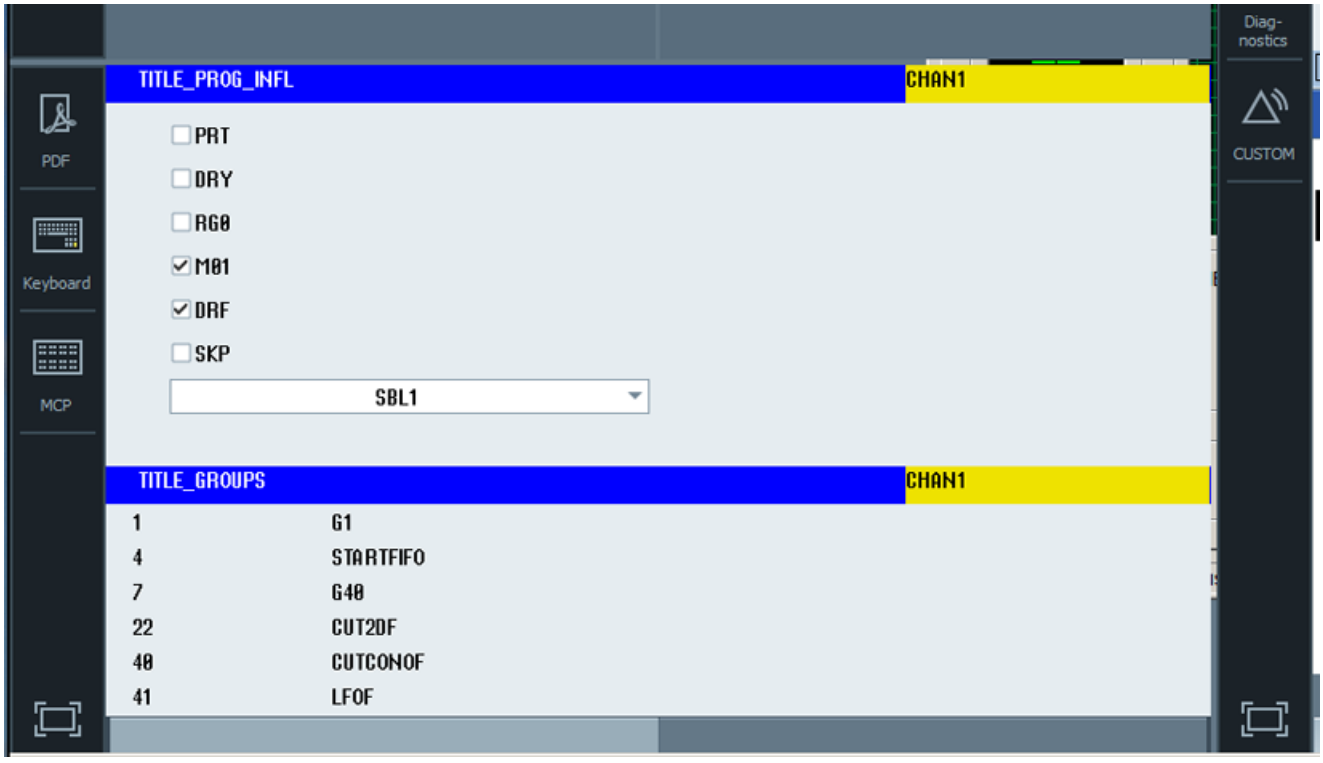
开关及元件的创建及含义在章节“Sidescreen 的使用 (页 253)”中予以说明。

```
[Sidescreen]
PAGE001= name:=WidgetsPage, implementation:=SlSideScreenPage
[Page_WidgetsPage]
ELEMENT001= name:=EE, implementation:=SlSideScreenElement
ELEMENT002= name:=AxesPosition,
implementation:=SlSideScreenElement
ELEMENT003= name:=WorkOffset, implementation:=SlSideScreenElement
ELEMENT004= name:=ActTool, implementation:=SlSideScreenElement
ELEMENT005= name:=VariableView,
implementation:=SlSideScreenElement

[元素_EE]
TextId=TITLE_R_PARAMETER_LIST
TextFile=rparameter_widget
TextContext=EASY_XML
WIDGET001= name:=rparameter_widget,
implementation:=slagmforms.SlEESideScreenWidget
```

待编辑的脚本（不带文件扩展）以及执行 **slagmforms.SlEESideScreenWidget** 分配至 **WIDGET001** 。

2.15.4 Display Manager Sidescreen 页面

Display Manager 中的 Sidescreen 页面可按照 Sidescreen 页面进行创建和管理。但是，“Sidescreen 页面”的说明仍在 INI 文件 **sldmxxxpage*.ini** 中完成。通过 **systemconfiguration.ini** 创建至该文件的引用。



包含文件 **sidescreen_proginfluence.xml** 的示例说明请见章节“Sidescreen 页面 (页 259)”。

示例

sldmmcppage.ini

```
DLG110= name:=SlMcpApp,
implementation:=sldmsidescreenapp.SlSideScreenDialog,
process:=SlHmiHost1, preload:=false, cmdline:="-sidescreen1
sldmmcppage.ini"
```

将对话框名称分配至框架：

```
FRAME028= name:=fBelowOperate, x:=896, y:=838, width:=1024,
height:=242, app:=SlKeyboardApp, runnableApps:="SlKeyboardApp,
SlMcpApp"
```

Display 中定义的框架：

```
DISPLAY002= name:=d4, frames:="fHeader, fOperate4, fMenuOperate4,
fBelowOperate, fMenuBelowOperate, fTop, fMenuTop, fBottom,
fMenuBottom"
MENUITEM124= name:=miMcp, menuItemStyle:=misMenu,
onClicked:="hidePopup(); showApp(defaultFrame, SlMcpApp)",
image:=dm_mcp.png, textID:=SL_DM_MCP
```

sldmmcppage.ini

```
[Sidescreen]
PAGE001= name:=McpPage,
implementation:=slsidescreenmcppage.SlSideScreenMcpPage
PAGE002= name:=sidescreen_proginfluence,
implementation:=slagmforms.SlEESideScreenPage
```

待编辑的脚本（不带文件扩展）以及执行 **slagmforms.SlEESideScreenPage** 分配至 **PAGE002** 。

2.16 通过 PLC 硬键选择对话框

使用范围

通过 PLC 能够在操作软件中启动以下功能：

- 选择操作区域，
- 选择操作区内的某个屏幕
- 执行软键上标注的功能

硬键

所有按键，包括 PLC 按键在下文都称为“硬键”。可以最多定义 254 个硬键，具体可以分为：

按键号	使用
按键 1 - 9	操作面板上的按键
按键 10 - 49	保留
按键 50 - 254 按键 50 - 81	PLC 按键： 预留给 OEM 使用
按键 255	由控制信息预定义

硬键 1 - 9 的预定义如下：

按键标识		动作/作用
HK1	MACHINE	选择操作区“加工”，回到上一个对话框
HK2	PROGRAM	选择操作区“程序”，回到上一个对话框或上一个程序
HK3	OFFSET	选择操作区“参数”，回到上一个对话框
HK4	PROGRAM MANAGER	选择操作区“程序”，初始画面“程序管理器” 无功能
HK5	ALARM	选择操作区“诊断”，对话框“报警列表”
HK6	CUSTOM	选择操作区“Custom”

配置

按键的定义在配置文件"systemconfiguration.ini"的章节[keyconfiguration]中进行。每一行定义了一个“硬键事件”。“硬键事件”指第几次按下某个硬键。比方说第二次按下某个硬键和第三次按下该硬键产生的动作就不一样。

配置文件"systemconfiguration.ini"中的条目可以载入用户自定义的设置，可以使用目录 [系统用户目录]/cfg 和 [系统 oem 目录]/cfg。

配置文件中用于定义硬键事件的行的结构为：

```
KEYx.n = area:=area, dialog:=dialog, screen:=screen,
forms:=form, menus:=menu, action:=menu.action, cmdline:=cmdline
KEYx.n = area:=area, dialog:=dialog, cmdline:=cmdline,
action:= action
```

x:硬键的编号，值域：1 – 254

n:事件号 - 相当于第几次按下硬键，值域：0 – 9

前提条件

PLC 用户程序必须首先符合以下前提条件：

每次总是只执行一个硬键功能。因此只有当操作软件应答了前面的一个请求后，才可以处理新的请求。当 PLC 用户程序将 MCP 按键转成硬键时，必须有足够大的中间缓存，这样即使在快速操作中也不会丢失某个按键事件。

PLC 接口

在 PLC 接口上对用于选择硬键的范围进行了规定。该范围定义在 DB19.DBB10 中。在该范围中，PLC 可以直接指定 50 到 254 之间的按键值。

操作软件分两步应答硬键事件。这样操作软件便可以将两个连续的相同按键代码看成两个单独的事件进行处理。在第一步中，控制信息 255 被写入到字节 DB19.DBB10 中。借助该虚拟的按键操作可以明确地看出 PLC 的每个按键序列。控制信息对于 PLC 程序没有作用，不允许被修改。在第二步中，操作软件删除 DB19.DBB10，真正地向 PLC 发出应答。从此刻起，PLC 用户程序便成功指定了一个新硬键。与此同时，操作软件开始处理新的硬键请求。

示例

配置文件：

```
; configuration of OP hardkeys (KEY1-KEY9) and
; PLC hardkeys (KEY50-KEY254)
[keyconfiguration]
; MACHINE key (hardkey block)
KEY1.0 = area:=AreaMachine, dialog:=SlMachine
; PROGRAM key (hardkey block)
KEY2.0 = area:=AreaProgramEdit
; OFFSET key (hardkey block)
KEY3.0 = area:=AreaParameter
; PROGRAM MANAGER key (hardkey block)
KEY4.0 = area:=AreaProgramManager
; ALARM key (hardkey block)
KEY5.0 = area:=AreaDiagnosis, dialog:=SlDgDialog,
cmdline:="-slGfwHmiScreen SlDgAeAlarmsScreen"
; CUSTOM (hardkey block)
KEY6.0 = area:=Custom
; MACHINE key (optional, Shift + F10)
KEY7.0 = area:=AreaMachine, dialog:=SlMachine,
cmdline:="-MKey"
; MENU USER (hardkey block, Ctrl + F10)
KEY10.0 = area:=MenuUser
; MENU FUNCTION (hardkey block, Ctrl + Shift + F10)
KEY11.0 = area:=MenuFunction
KEY50.0 = area:=CustomXML, dialog:=SlEECustomDialog
```

区域和对话框名称请从 [系统西门子目录]/cfg 目录下的 systemconfiguration.ini 文件中获取。

若仅适用对话框设定 **SlEECustomDialog**，则分析程序等待应用目录中的文件 **xmldial.xml** 以及名称为 **main** 的菜单标签。可通过添加密钥 **cmdline** 公布其他文件名或主菜单名称。参数 **-mainModule** 定义待载入的 XML 说明。在此脚本中，分析程序等待主菜单。参数 **-entry** 定义主菜单的名称。若缺少此参数，则分析程序会使用名称 **main** 来启动功能。必须总是包含值为 **slagmdialog.hmi** 的参数 **-conf**。

示例：

```
KEY50.0 = area:=CustomXML, dialog:=SlEECustomDialog,
cmdline:="-conf slagmdialog.hmi -mainModule restore.xml -entry
cmc2"
KEY51.0 = area:=CustomXML, dialog:=SlEECustomDialog,
cmdline:="-conf slagmdialog.hmi -mainModule activate.xml
-entry cmc1"
```

操作界面

借助文件 systemconfiguration.ini 中的下列条目，能够将 Easy XML 用于上述功能。

```
AREA0111= name:=CustomXML, dialog:=SlEECustomDialog,
panel:=SlHdStdHeaderPanel
DLG056= name:=SlEECustomDialog,
implementation:=slagmdialog.SlEECustomDialog,
process:=SlHmiHost1, preload:=false, terminate:=true,
cmdline:="-conf slagmdialog.hmi"
```

2.17 将用户窗口分配给预留软键

在所有的标准操作区域中都为扩展当前功能而预留了一个软键。为激活和连接带有用户项目的 OEM 软键，需要在以下系统目录中保存相应的文件：

/siemens/sinumerik/hmi/template/cfg

这些文件包含了特定操作区域的 XML 描述，这些描述在操作软件启动时会被编译并集成到操作区域的菜单树中。

为了集成自定义应用程序，需要将模板目录中与相应操作区域相关的文件复制到以下目录中并进行修改：

/oem/sinumerik/hmi/template/cfg

2.17.1 确定无需区分语种的软键文本

标签 TEXTFILE 包含了属于 OEM 区域的文本文件的文件名。文件名不需包括所使用语种的缩写和文件扩展名。

语法

```
<TEXTFILE>oemtextfile</TEXTFILE>
```

示例

oem_xml_screens_deu.ts

```
<TEXTFILE>oem_xml_screens</TEXTFILE>
```

在标签 **property** 中 **softkey** 中管理要显示的软键文本。OEM 值可用希望使用的文本 ID 来替代。属性 **translationContext** 指向文本文件的一个区域，文本分配在此区域中。在 easyXML 中，需要给定 EASY_XML 上下文。

语法

```
<PROPERTY name="textID" type="QString">OEM</PROPERTY>
<PROPERTY name="translationContext" type="QString">OEMContext</
PROPERTY>
```

示例

```
<PROPERTY name="textID" type="QString">MY_TEXT1</PROPERTY>
<PROPERTY name="translationContext" type="QString">EASY_XML</
PROPERTY>
```

2.17.2 Easy XML 区域分配

在标签 **softkey** 中可进一步确定导航目标。为此，需要在标签 **area** 中将导航目标 **CustomXML** 输入属性 **name** 中。需要删除属性 **dialog** 和 **screen**，因为这些参数是自动确定的。

语法

```
<NAVIGATION target="area">
<AREA name="OEMArea" dialog="OEMDialog" screen="OEMScreen"/>
</NAVIGATION>
```

示例

```
<NAVIGATION target="area">
<AREA name="CustomXML" />
</NAVIGATION>
```

如果使用导航目标 **area**，那么 Parser 会以文件 xmldial.xml 作为启动模块，以菜单 **main** 作为菜单管理的进入点。为了同时提供多个应用，可使用 **switchToDialog** 功能。为此，用 FUNCTION 标签取代 NAVIGATION 标签。**CustomXML** 区域、**SlEECustomDialog** 对话框以及主菜单和启动模块的名称都通过此功能来调用。

通过 **switchToArea** 标签退出 easyXML 区域，返回标准操作区域。

语法

```
<switchToArea name="<Area>"/>
...
<FUNCTION args="-area CustomXML -dialog SlEECustomDialog -
mainModule <Filename> -entry <Menuname>" name="switchToDialog"/>
```

项目示例

sldiagnose_oem.xml

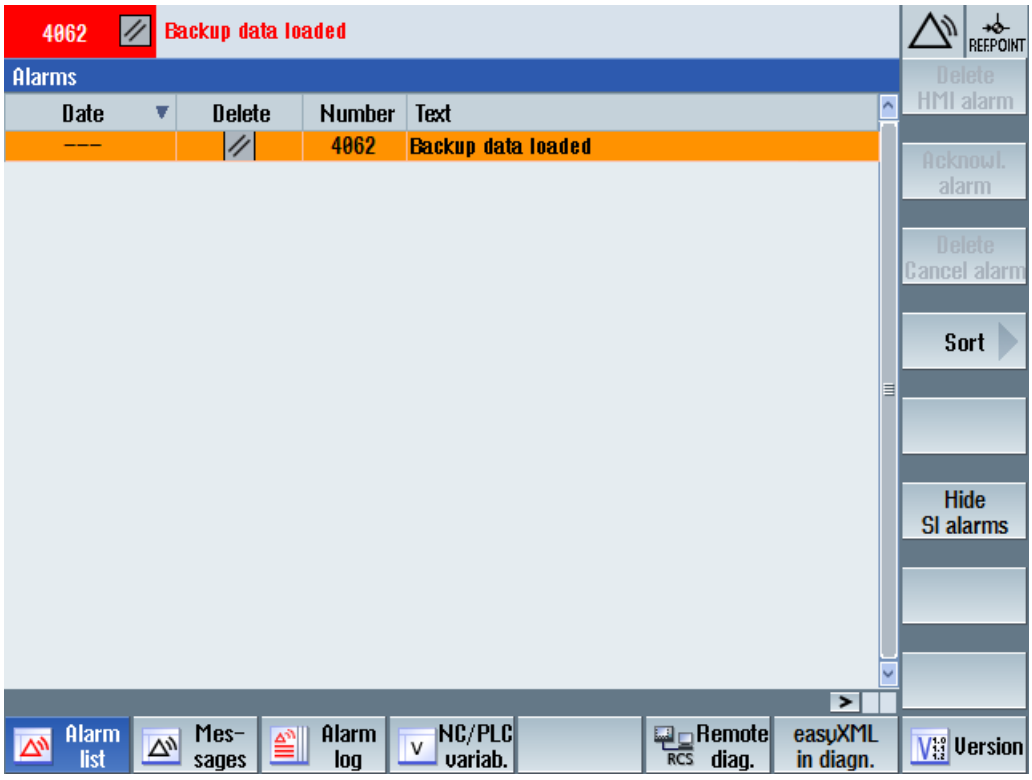
```
<!DOCTYPE DIALOG>
<DIALOG>
<TEXTFILE>oem_xml_screens</TEXTFILE>
<!-- ========================================================================
-->
<!-- OEM softkey number 7 on first horizontal main menu in area diagnosis --
>
<!-- ========================================================================
-->
<MENU name="DgGlobalHu">
 <ETCLEVEL id="0">
 <SOFTKEYGROUP name="GroupEtc">
 <SOFTKEY position="7">
  <PROPERTY name="textID" type="QString">DG_SK</PROPERTY>
  <PROPERTY name="translationContext" type="QString">EASY_XML</PROPERTY>
  <NAVIGATION target="area">
   <AREA name="CustomXML" />
  </NAVIGATION>
 </SOFTKEY>
</SOFTKEYGROUP>
</ETCLEVEL>
</MENU>
</DIALOG>
```

或

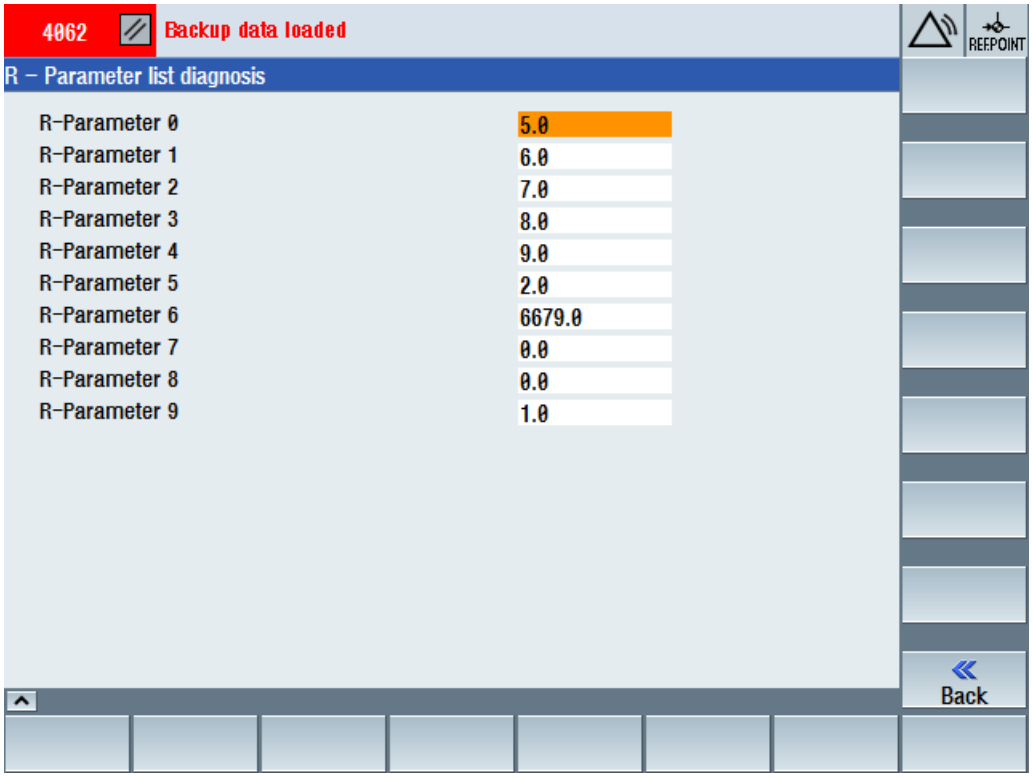
```
<!DOCTYPE DIALOG>
<DIALOG
<!-- ========================================================================
-->
<!-- OEM softkey number 7 on first horizontal main menu in area diagnosis --
>
<!-- ========================================================================
-->
<TEXTFILE>oem_xml_screens</TEXTFILE>
<MENU name="DgGlobalHu">
 <ETCLEVEL id="0">
 <SOFTKEYGROUP name="GroupEtc">
 <SOFTKEY position="7">
  <PROPERTY name="textID" type="QString">DG_SK</PROPERTY>
  <PROPERTY name="translationContext" type="QString">EASY_XML</PROPERTY>
  <FUNCTION args="-area CustomXML -dialog SlEECustomDialog -mainModule
dg.xml -entry main" name="switchToDialog"/>
 </SOFTKEY>
</SOFTKEYGROUP>
</ETCLEVEL>
</MENU>
</DIALOG>
```

dg.xml

```
<DialogGui>
<menu name = "main">
 <open_form name = "R_PARAMETER_LIST" />
 <softkey_back>
  <switchToArea name="AreaDiagnosis" dialog="SlDgDialog"/>
 </softkey_back>
</menu>
<!-- This form shows a R - parameter list -->
<form name = "R_PARAMETER_LIST">
 <init>
 <caption>R - Parameter list diagnosis</caption>
 <data_access type="true" />
  <control name = "c1" xpos = "322" ypos = "34" refvar="nck/Channel/
Parameter/R[0]" hotlink="true" />
  <control name = "c2" xpos = "322" ypos = "54" refvar="nck/Channel/
Parameter/R[1]" hotlink="true" />
  <control name = "c3" xpos = "322" ypos = "74" refvar="nck/Channel/
Parameter/R[2]" hotlink="true" />
  <control name = "c4" xpos = "322" ypos = "94" refvar="nck/Channel/
Parameter/R[3]" hotlink="true" />
  <control name = "c5" xpos = "322" ypos = "114" refvar="nck/Channel/
Parameter/R[4]" hotlink="true" />
  <control name = "c6" xpos = "322" ypos = "134" refvar="nck/Channel/
Parameter/R[5]" hotlink="true" />
  <control name = "c7" xpos = "322" ypos = "154" refvar="nck/Channel/
Parameter/R[6]" hotlink="true" />
  <control name = "c8" xpos = "322" ypos = "174" refvar="nck/Channel/
Parameter/R[7]" hotlink="true" />
  <control name = "c9" xpos = "322" ypos = "194" refvar="nck/Channel/
Parameter/R[8]" hotlink="true" />
  <control name = "c10" xpos = "322" ypos = "214" refvar="nck/Channel/
Parameter/R[9]" hotlink="true" />
 </init>
 <paint>
  <text xpos = "23" ypos = "34">R-Parameter 0</text>
  <text xpos = "23" ypos = "54">R-Parameter 1</text>
  <text xpos = "23" ypos = "74">R-Parameter 2</text>
  <text xpos = "23" ypos = "94">R-Parameter 3</text>
  <text xpos = "23" ypos = "114">R-Parameter 4</text>
  <text xpos = "23" ypos = "134">R-Parameter 5</text>
  <text xpos = "23" ypos = "154">R-Parameter 6</text>
  <text xpos = "23" ypos = "174">R-Parameter 7</text>
  <text xpos = "23" ypos = "194">R-Parameter 8</text>
  <text xpos = "23" ypos = "214">R-Parameter 9</text>
 </paint>
</form>
</DialogGui>
```


4062
Backup data loaded
REFPOINT
Alarms
Date
Delete
Number
Text

4062
Backup data loaded
Delete HMI alarm
Acknowl. alarm
Delete Cancel alarm
Sort
Hide SI alarms
Alarm list
Mes-sages
Alarm log
NC/PLC variab.
Remote diag.
RCS
easyXML in diagn.
Version


4062
Backup data loaded
REFPOINT
R – Parameter list diagnosis
R-Parameter 0
5.0
R-Parameter 1
6.0
R-Parameter 2
7.0
R-Parameter 3
8.0
R-Parameter 4
9.0
R-Parameter 5
2.0
R-Parameter 6
6679.0
R-Parameter 7
0.0
R-Parameter 8
0.0
R-Parameter 9
1.0
Back

2.17.3 标签描述

软键标签

除了 **softkey** 标签，软键状态也可在已使用 POSITION 属性时通过 **state** 属性进行设置。给出软键号作为属性值，以设定软键的状态。

示例

```
<menu name = "menu_sktest">

 <state type="$$$define_sk_type" POSITION="2"/>

<softkey POSITION="2" type="user_controlled" picture="$$
$bmp_name">
 <caption>Toggle%nSK</caption>
 <if>
 <condition>sk_type == 0 </condition>
 <then>
  <op> sk_type = 1 </op>
  <op> define_sk_type = _T"PRESSED" </op>
  <op>bmp_name = _T"f:\appl\red_led_on.bmp"</op>
  <state type="$$$define_sk_type" />
 </then>
 <else>
  <op> define_sk_type = _T"NOTPRESSED" </op>
  <op>bmp_name = _T"f:\appl\red_led_off.bmp"</op>
  <op> sk_type = 0 </op>
  <state type="$$$define_sk_type" />
 </else>
 </if>
  <print text="%s">sk_type_name</print>
  <navigation>menu_sktest</navigation>
</softkey>
...
...
```

typedef 标签

说明

示例中的元素预设值应当删除。

在类型定义中，使用 **element** 标签声明一个变量。标签的属性对应于 **let** 指令的属性。元素预设置可在创建变量时进行。

示例

RECT:

```
<typedef name="StructRect" type="struct" >
<element name="left" type="int" />
<element name="top" type="int" />
<element name="right" type="int" />
<element name="bottom" type="int" />
</typedef>
```

POINT:

```
<typedef name="StructPoint" type="struct" >
<element name="x" type="int" />
<element name="y" type="int" />
</typedef>
```

SIZE:

```
<typedef name="StructSize" type="struct" >
<element name="width" type="int" />
<element name="height" type="int" />
</typedef>
```

let 标签

用属性 **dim** 可给定数组元素的数量。对于二维数组，用逗号分隔第一维和第二维。对数组元素的存取通过数组下标进行，写在变量名称后的方括号内。字符串的维度可直接定义或通过一个变量的内容来定义。

示例

```
 <let name="d1" >3</let>

<let name="list_string" dim="3" type="string" />
```

或

```
<let name="list_string" dim="$d1" type="string" />
...
...
<op>
  list_string[2] = _T"test";
</op>
```

或

```
<let name="d1" >3</let>
<let name="d2" >6</let>

<let name="list_string3" dim="3, $d2" type="string" />
```

```
<op>
  list_string3[2, 5] = _T"test";
</op>
```

2.18 创建无需区分语种的文本

通过在对话框中使用文本标签，所有在对话框中使用的文本都会被替代为目前所选语种的语言，因而管理对话框中使用的文本时可不用考虑语种。此文本会和已提供的每个语种的文本标签一同保存在一个 UTF8 格式的文本文件中。

oem_xml_screens_<文件默认会作为文本文件归入 Easy XML 功能的语种缩写>.ts 中。给定 EASY_XML 标签作为文本上下文。

为辨识文本标签，在脚本中文本标签前会添加两个美元符号。

结构

oem_xml_screens_deu.ts

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE TS>
<TS>
<context>
    <name>EASY_XML</name>
    <message>
    <source>MY_TEXT1</source>
    <translation>text1 from textfile</translation>
    </message>
    <message>
    <source>MY_TEXT2</source>
    <translation>text2 from textfile</translation>
    </message>
</context>
</TS>
```

示例

```
<text xpos ="120" ypos="158">$$MY_TEXT1</text>
```

2.19 特定项目的文本文件

2.19.1 创建特定项目的文本文件

为分开管理各个项目，可为项目、窗体和菜单各自分别使用一个单独的文本文件。通过 Attribut 属性在相对应的标签中进行发布。

给出不含语言标签和文件扩展名的文本文件名作为属性值。在窗体或菜单关闭前，一直可以访问文本。如果加载带有 **DialogGui** 标签文本文件，其内容在退出项目之前可以一直访问。对文本标签的搜索从最近加载的文件开始。

示例

main_form_screens_deu.ts

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE TS>
<TS>
<context>
    <name>EASY_XML</name>
    <message>
    <source>MAIN_FORM_TITLE</source>
    <translation>user text file title</translation>
    </message>
</context>
</TS>
```

激活

```
<DialogGui textfile="main_form_screens">
...
...
</DialogGui>
```

或

```
<form name="main_form" textfile="main_form_screens">
...
...
</form>
```

或

```
<menu name="main" textfile="main_form_screens">
...
...
</menu>
```

对话框示例

main_form_screens_deu.ts

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE TS>
<TS>
<context>
    <name>EASY_XML</name>
    <message>
    <source>MAIN_FORM_TITLE</source>
    <translation>user text file title</translation>
    </message>
    <message>
    <source>MAIN_FORM_TEXT</source>
    <translation>text from text file</translation>
    </message>
</context>
</TS>
```

main2_sktext_screens_deu.ts

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE TS>
<TS>
<context>
    <name>EASY_XML</name>
    <message>
    <source>SK1</source>
    <translation>Softkey1</translation>
    </message>
</context>
</TS>
```

form2_screens_deu.ts

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE TS>
<TS>
<context>
    <name>EASY_XML</name>
    <message>
    <source>FORM2_TITLE</source>
    <translation>Form2 Title</translation>
    </message>
    <message>
    <source>FORM2_TEXT</source>
    <translation>Form2 text from text file</translation>
    </message>
</context>
</TS>
```

对话框脚本

xmldial.xml

```
<DialogGui textfile="main_form_screens">
<menu name = "main">
    <open_form name = "main_form" />
    <softkey POSITION="1">
    <caption>Menu 2</caption>
    <navigation>menu_2</navigation>
    </softkey>
</menu>
<menu name = "menu_2" textfile="main2_sktext_screens">
    <open_form name = "form2" />
    <softkey POSITION="1">
    <caption>$$SK1</caption>
    </softkey>
    <softkey_back>
    <navigation>main</navigation>
    </softkey_back>
</menu>
<form name="main_form" >
 <init>
    <data_access type = "true" />
    <caption>$$MAIN_FORM_TITLE</caption>
 </init>
 <paint>
    <text xpos="5" ypos="30">$$MAIN_FORM_TEXT</text>
 </paint>
</form>
<form name="form2" textfile="form2_screens">
 <init>
    <data_access type = "true" />
    <caption>$$FORM2_TITLE</caption>
 </init>
 <paint>
    <text xpos="5" ypos="30">$$FORM2_TEXT</text>
 </paint>
</form>
</DialogGui>
```

2.19.2 给定文本上下文

在一个文本文件内，可通过自行确定的区域名将文本进行分组。在标签 **context** 中，通过标签 **name** 确定区域标签。

语法

```
<context>
```

```
 <name>name</name>
</context>
```

示例

```
<context>
 <name>my context 1</name>
...
...
</context>
<context>
 <name>my context 2</name>
...
...
</context>
```

如要使用自定义的上下文，应通过 TEXTCONTEXT 属性为项目、窗体或菜单给定上下文名称以及文本文件名称。在设定新的上下文之前，在一个上下文之中的文本可以一直访问。也就是说，一个设置在项目上的上下文会替换为一个设置在窗体或菜单上的上下文。

项目示例

form2_screens_deu.ts

```
?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE TS>
<TS>
<context>
 <name>EASY_XML</name>
  <message>
   <source>FORM2_TITLE</source>
   <translation>Form2 Title</translation>
  </message>
  <message>
   <source>FORM2_TEXT</source>
   <translation>Form2 text from text file</translation>
  </message>
</context>
<context>
 <name>MY_FORM_CONTEXT</name>
  <message>
   <source>FORM3_TITLE</source>
   <translation>Title from the text context MY_FORM_CONTEXT</translation>
  </message>
  <message>
   <source>FORM3_TEXT</source>
   <translation>Form3 text from text file</translation>
  </message>
</context>
<context>
 <name>MY_SOFTKEY_CONTEXT</name>
  <message>
   <source>MENU3_SK1</source>
   <translation>SK%n1</translation>
  </message>
</context>
</TS>
```

using_textcontext.xml

```
...
<menu name = "menu3" textfile="form2_screens"
textcontext="MY_SOFTKEY_CONTEXT">
 <open_form name = "menu3_form" />
 <softkey POSITION="1">
  <caption>$$SK1</caption>
 </softkey>
 <softkey_back>
  <navigation>main</navigation>
 </softkey_back>
</menu>
<form name="menu3_form" textfile="form2_screens"
textcontext="MY_FORM_CONTEXT">
 <init>
  <data_access type = "true" />
  <caption>$$FORM3_TITLE</caption>
 </init>

 <paint>
  <text xpos="5" ypos="30">$$FORM3_TEXT</text>
 </paint>
</form>
...
...
```

2.19.3 给定文本文件列表

通过属性值 TEXTFILELIST 可在一个待加载文本文件的列表中给定文件名称。

文本文件名称的标记逐行定义，并通过换行符加以结束。作为文本文件的名称，不带有语言扩展名和文件扩展名。

示例

文件 form2_screens_deu.ts 应列在带 form2_screens 的列表中。

textfilelist.ini

```
form2_screens
...
...
```

激活

```
<DialogGui textfilelist="txtfilelist.ini">
...
...
</DialogGui>
```

2.20 恢复会话

关闭 Easy XML 项目时，所有本地变量都会被释放，脚本指令会被卸载。如果想要在关闭控制系统后仍保留变量数据，则需要预先将这些变量的属性设为 **permanent**。只需保留至控制系统关闭时的数据应将属性设为 **poweroff**。

如果再次选择 Easy XML 应用程序，则应在主菜单中设置重新选择应用程序后要激活哪个菜单或哪个窗体。编程人员必须确保所有必需数据均可用。

如果在标签 **DialogGUI** 中设置了属性 **restoresession**，则分析程序会重新选择上次打开的菜单和窗体。编程人员负责准备必需的数据。

创建调试对话框

3

3.1 功能概述

使用目的

使用"Easy Extend"能够非常简单地调试、激活、取消激活或测试附加设备。控制系统中有一张显示所有可用设备和设备状态的清单。系统可以最多管理 64 个设备。

设备的激活或取消由软键操作实现。

功能"Easy Extend"位于"参数"操作区域中。

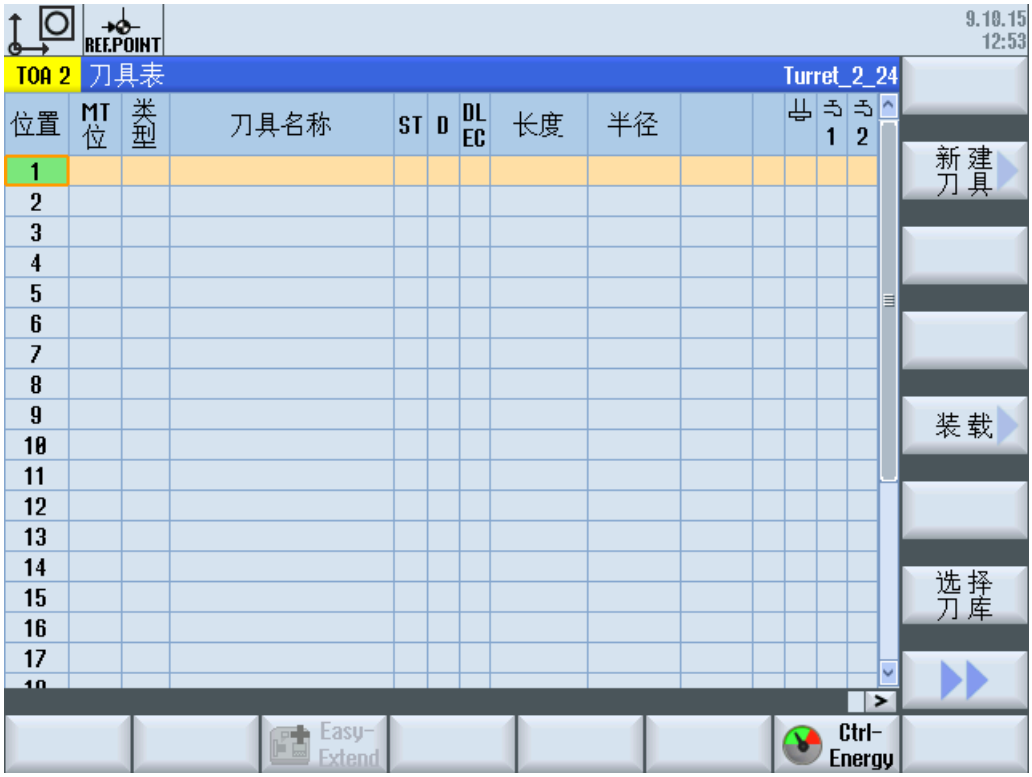


图 3-1 基本画面"参数"

配置

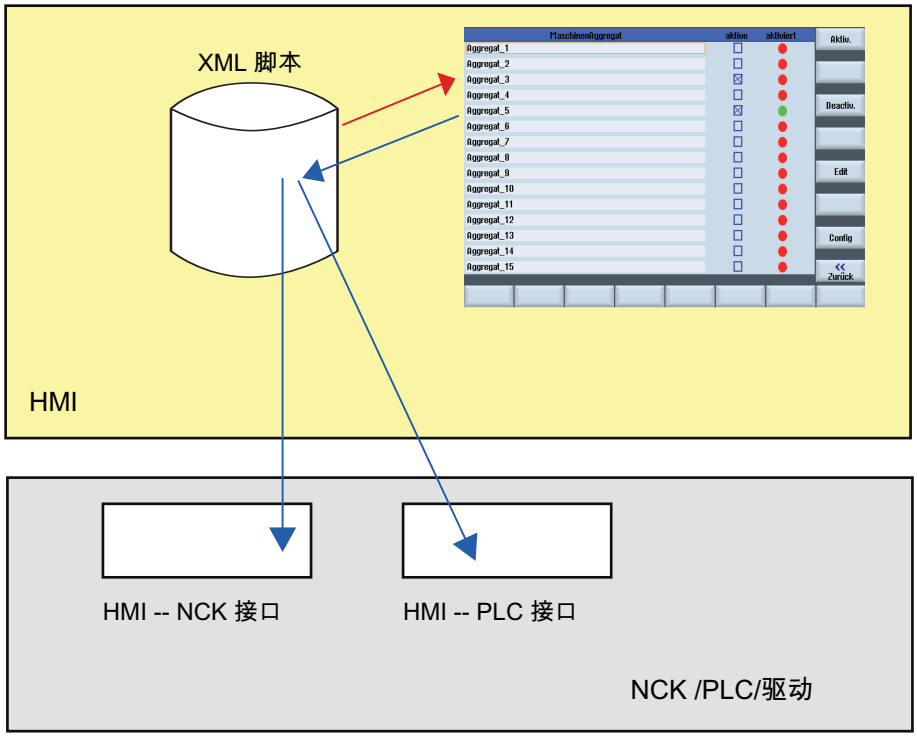


图 3-2 “Easy Extend”的功能

机床制造商首先定义以下功能，才可以使用“Easy Extend”功能：

- **PLC ↔ HMI 的接口**
  可选设备是通过操作界面和 PLC 之间的接口来管理的。
- **脚本编辑**
  机床制造商应在一个指令脚本中定义如何测安装、激活、取消和测试设备的流程。
- **参数窗口（可选）**
  参数窗口用于显示在脚本文件中定义的设备信息。

文件保存

和“Easy Extend”相关的文件保存在系统 CF 卡的目录“dvm”（机床制造商）中。

文件	名称	目标目录
文本文件	oem_aggregate_xxx.ts	\oem\sinumerik\hmi\lng
脚本文件	agm.xml	\oem\sinumerik\hmi\dvm

文件	名称	目标目录
存档文件	任意	\oem\sinumerik\hmi\dvm\archives
PLC 用户程序	任意	PLC

3.2 PLC 用户程序中的配置

载入配置

首先将制造商创建的配置和脚本文件、文本文件一起传送到控制系统的制造商目录中。另外，还需要载入对应的 PLC 用户程序。

设备的编程

操作组件和 PLC 之间是通过 PLC 用户程序中由编程人员确定的数据块进行通讯的，该数据块中有 128 个字预留用于设备的管理。有关数据块标识符的说明请见章节“PLC_INTERFACE (页 308)”。

数据块必须作为用户块创建并加载。脚本中 "agm.xml"，每个程序块由 EasyExtend 功能的 **plc_interface** 标签进行标识。

说明

兼容性

建议将数据块 DB9905 定义为 PLC 接口，这样脚本也能与 SINUMERIK 828D 兼容。

示例：

```
<plc_interface name = "plc/db1000.dbb0" />
```

创建程序块时可以采用下表中的数据格式以及符号名称：

数据格式/符号名称	说明
DBX0.0 Enable_1 BOOL Bit OFF OFF HMI → PLC	设备已经启动
DBX0.1 Activate_1 BOOL Bit OFF OFF HMI → PLC	请激活设备
DBX0.2 Deactivate_1 BOOL Bit OFF OFF HMI → PLC	请取消激活设备
DBB1 Res_1 BYTE 不带符号 0 0	保留作为将来用
DBX2.0 IsActive_1 BOOL Bit OFF OFF PLC → HMI	设备已激活
DBX2.1 Error_1 BOOL Bit OFF OFF PLC → HMI	设备出错
DBB3 DeviceId_1 BYTE 不带符号 0 0	唯一设备编号

首先从设备 1 开始，指定 PLC 数据字：

数据块	设备名称
DB9905.DBB0	设备 1
DB9905.DBB4	设备 2
...	...
DB9905.DBB192	设备 49
DB9905.DBB196	设备 50

每个设备可以占用四个字节，含义如下：

字节	位	说明	
0	0	== 1	设备已经启动（HMI 反馈信息）
	1	== 1	请激活设备（HMI 请求）
	2	== 1	请取消激活设备（HMI 请求）
	3-7	预留	
1	0-7	预留	
2	0	== 1	设备已激活（PLC 反馈）
	1	== 1	设备出错
	2-7	预留	
3	0-7	设备的唯一标识	

文件保存

文件保存在系统 CF 的目录“oem”（机床制造商）和“oem_i”（个人）下。

说明

文件名只能由小写字母构成。

文件	名称	目标目录
文本文件	oem_aggregate_xxx.ts	/oem/sinumerik/hmi/lng/ /oem_i/sinumerik/hmi/lng/
脚本文件	agm.xml	/oem/sinumerik/hmi/dvm /oem_i/sinumerik/hmi/dvm

文件	名称	目标目录
存档文件	任意	/oem/sinumerik/hmi/dvm/archives /oem_i/sinumerik/hmi/dvm/archives
PLC 用户程序	任意	PLC

一般操作顺序

机床制造商首先执行以下操作，提供所需数据：

1. 创建 PLC 用户程序，该程序中包含了由 PLC 控制的设备。
2. 调试“标准机床”，接着将数据备份到批量调试文档中。
3. 安装、调试设备，将数据作为差别批量调试文档读出。

说明

修改机床配置

需要修改驱动机床数据时，首先在控制系统中执行修改。然后在所有设置和装置上再次修改。

添加轴

在需要为机床增加机床数据时，请按照固定顺序添加驱动对象(DO)，因为批量调试文档包含了机床制造商参考机床的结构，一旦顺序被改动，便无法使用。

我们推荐为控制系统组件选择以下设置：

- NC 数据
- PLC 数据
- 驱动数据
  – ACX　格式（二进制）

3.3 操作界面上的显示画面

操作界面上的窗口

以下窗口用于“Easy Extend” 功能：

- 控制系统提供了一个**自定义窗口**，其中可显示可用设备。
- 如果还没有开展首次调试，系统会打开**调试窗口**。

如果已经为设备编写了一个调试流程（XML 指令：“START_UP”），而设备还没有经过调试，系统会启动调试流程。

此时，在读取脚本文件中保存的批量调试文档前，会完整备份数据。

出错时调试人员可以决定，是否取消调试，或手动排除可能出现在机床配置中的错误。

- 按下“取消”，可以中断调试。系统随后会将取消调试数据的备份。

调试成功结束后需要断开机床时，可以编写 XML 指令“POWER_OFF”，控制系统上便会输出对应的信息。

3.4 创建和语言相关的文本

文本文件结构

包含和语言相关的文本的 xml 文件应以 UTF8 格式创建：

示例

oem_aggregate_eng.ts

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE TS>
<TS>
<context>
    <name>EASY_EXTEND</name>
    <message>
    <source>DEVICE_ONE</source>
    <translation>Device one</translation>
    </message>
    <message>
    <source>DEVICE_TWO</source>
    <translation>Device two</translation>
    </message>
</context>
</TS>
```

oem_aggregate_deu.ts

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE TS>
<TS>
<context>
    <name>EASY_EXTEND</name>
    <message>
    <source>DEVICE_ONE</source>
    <translation>Erstes Gerät</translation>
    </message>
    <source>DEVICE_TWO</source>
    <translation>Zweites Gerät</translation>
    </message>
</context>
</TS>
```

3.5 功率部件应用示例

激活驱动对象

需要激活的驱动对象已经过调试，并由机床制造商重新取消激活，用于将轴作为选件投入市场。

必须执行以下步骤激活轴：

- 通过 p0105 激活驱动对象。
- 在通道机床数据中激活第 2 根轴。
- 通过 p0971 备份驱动机床数据。
- 等待数据写入。
- 重启 NCK 和驱动。

编程：

```
<DEVICE>
  <list_id>1</list_id>
  <name> "激活驱动"</name>

  <SET_ACTIVE>
    <data name = "drive/dc/p105[DO5]">1</data>
    <data name = "$MC_AXCONF_MACHAX_USED[4]">5</data>
    <data name = "drive/dc/p971[DO5]">1</data>
    <while>
      <condition> "drive/dc/p971[DO5]" !=0 </condition>
    </while>
    <control_reset resetnc ="true" resetdrive = "true"/>
  </SET_ACTIVE>

  <SET_INACTIVE>
    <data name = "drive/dc/p105[DO5]">0</data>
    <data name = "$MC_AXCONF_MACHAX_USED[4]">0</data>
    <data name = "drive/dc/p971[DO5]">1</data>
    <while>
      <condition> "drive/dc/p971[DO5]" !=0 </condition>
    </while>
    <control_reset resetnc ="true" resetdrive = "true"/>
  </SET_INACTIVE>

</DEVICE>
```

激活由 PLC 控制的设备

设备通过输出字节 10 应答，并且通过输出字节 9 向 PLC 发送运行就绪反馈信息。

输出字节被设置为固定的编码用于激活设备。接着，While 循环等待设备运行准备就绪。

编程：

```
<SET_ACTIVE>
  <DATA name = "plc/qb10"> 8 </DATA>
  <while>
    <condition> "plc/ib9" !=1 </condition>
  </while>
</SET_ACTIVE>
```

3.6 脚本语言

说明

在功能 创建用户对话框 (页 17) 中描述的所有脚本元素形成了“Easy Extend”功能的基础。为管理附加装置，还定义了其它的脚本元素。

脚本的程序段落

脚本由以下段落组成：

- “Easy Extend” - 标签
- 用于确定设备可执行操作的框架
- 用于设备的标签
- 用于设备调试的标签
- 用于设备激活的标签
- 用于设备取消的标签
- 用于设备测试的标签
- 参数窗口的标签

下面的章节详细说明各个标签。

说明

标签 <tag>	含义
AGM	“调试向导”功能的标签
DEVICE	用于描述设备的标签。 **属性：** • **option_bit** 设备指定有一个固定的位编号，用于管理选件。
NAME	该标签定义在窗口中显示的设备名称。 如果使用参考文本，窗口中会显示该标签下定义的文本。
START_UP	该标签包含调试设备所需流程的说明。

标签 <tag>	含义
SET_ACTIVE	该标签包含激活设备所需流程的说明。 **属性：** • **timeout** 该属性指定超时时间，单位秒。如果脚本在该时间后仍未结束，系统会中断处理。
SET_INACTIVE	该标签包含取消设备所需流程的说明。 **属性：** • **timeout** 该属性指定超时时间，单位秒。如果脚本在该时间后仍未结束，系统会中断处理。
TEST	该标签包含设备测试所需的指令。 **属性：** • **timeout** 该属性指定超时时间，单位秒。如果脚本在该时间后仍未结束，系统会中断处理。
UID	用于区分 PLC ↔ HMI 接口中设备的唯一数字标识码。
VERSION	用于版本的标签

应答功能执行（negative）

通过自动提供的变量"$actionresult"系统可以向 XML-Parser 传递一个"negative"的执行结果。该值为零时，Parser 会中断执行功能。

示例

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE AGM>
<AGM>                                  “调试向导”的标签
<DEVICE>
  <NAME> 设备 1 </NAME>                用于设备的标签
  <START_UP>                           用于设备调试的标签
  …
  </START_UP>
  <SET_ACTIVE>                         用于设备激活的标签
  …
  </SET_ACTIVE>
```

```
    <SET_INACTIVE>                    用于设备取消的标签
    …
    </SET_INACTIVE>
    <TEST>                            用于设备测试的标签
    …
    </TEST>
  </DEVICE>
  …
  </AGM>
```

3.6.1 CONTROL_RESET

说明

该标签可以重新启动一个或多个控制系统组件。 只有当控制系统恢复周期运行时，才可以继续执行脚本。

编程

标签：	**CONTROL_RESET**	
句法：	<CONTROL_RESET resetnc="TRUE" />	
属性：	resetnc="true"	重新启动 NC 组件。
	resetdrive="true"	重新启动驱动组件。

3.6.2 FILE

说明

该标签可以读入或创建标准文档。

- 读入文档：
  请输入文档名称。
- 创建文档：
  定义了属性 create= "true" 后，该功能会以指定名称创建一个标准文档(*.arc)，并保存在目录 .../dvm/archives 中。

编程

标签：	**FILE**	
句法：	<file name ="<文档名称>" />	
	<file name =“<文档名称>” create=“true” class=“<数据类别>”	
	group=“<范围>” />	
属性：	name	用于文件名称的标签

create	以指定的名称在目录 …/dvm/archives/ 下创建一个调试文档。
group	指定文档包含的数据组。 如果需要备份多个数据组，请用空格隔开。 文档可以包含以下数据组： • NC • PLC • HMI • DRIVES

示例

```
<!-- 创建数据级文档 -->
<file name="user.arc" create="true"
group="nc plc hmi" />

<!–将文档读入控制系统->
<file name="user.arc" />
```

3.6.3 OPTION_MD

说明

该标签允许重新定义选件机床数据。 在出厂时，系统使用的是 MD14510 $MN_USER_DATA_INT[0] ~ $MN_USER_DATA_INT[3]。

如果由 PLC 用户程序管理选件，请在一个数据块或 GUD 中提供对应的数据字。

数据按位管理。 从位 0 开始，各个位按照设备清单上的顺序固定指定给各个设备，即：位 0 指定给设备 1；位 1 指定给设备 2，如此类推。如果需要管理多于 16 个设备，请通过区域索引将地址标识指定给设备组 1-3。

说明

换算取值范围

MD14510 $MN_USER_DATA_INT[i] 的取值范围是 -32768 ~ +32767。为了在机床数据窗口中按位使能各个设备，需要将位组换算成十进制格式。

编程

标签：	**OPTION_MD**	
句法：	区域 0：	
	<option_md name = "数据的地址标识" />	
	或者是	
	<option_md name = "数据的地址标识" index= "0"/>	
	区域 1~3：	
	<option_md name = "数据的地址标识" index= "区域索引"/>	
属性：	name	地址标识，例如：$MN_USER_DATA_INT[0]
	index	用于区域索引的标签：
		0 （缺省设置）： 设备 1 ~ 16
		1: 设备 17 ~ 32
		2: 设备 33 ~ 48
		3: 设备 49 ~ 64

3.6.4 PLC_INTERFACE

说明

该标签允许重新定义 PLC ↔ HMI 接口。系统提供 128 个可编址的数值。

说明
标签未预设，应由编程人员自行定义。

编程

标签：	**PLC_INTERFACE**	
句法：	<plc_interface name = "数据的地址标识" />	
属性：	name	地址标识，例如“plc/mb170”

示例：`plc/mb170`

3.6.5 POWER_OFF

说明

该标签用于设定一条要求操作员关闭机床的信息。 显示文本固定保存在系统中。

编程

标签：	**POWER_OFF**
句法：	<power_off />
属性：	--

3.6.6 WAITING

说明

该标签用于指定：在 NC 或驱动复位后，等待各个组件重新启动。

编程

标签：	**WAITING**
句法：	<WAITING WAITINGFORNC ="TRUE" />
属性：	waitingfornc="true" 等待 NC 重新启动。
	waitingfordrive="true" 等待驱动重新启动。

3.6.7 用于窗口的 XML 标签

用于设置参数的窗口

可以为每个设备设计一个窗口，用于设置或输出运行时间所需的附加参数。按下软键“附加参数”后，会显示该窗口。

在创建对话框时可以使用章节 创建用户对话框 (页 17) 中描述的所有脚本元素。

示例

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE AGM>
<AGM>
<DEVICE>
  <NAME> 设备 1 </NAME>
  <START_UP>
  …
  </START_UP>
  <SET_ACTIVE>
  …
  </SET_ACTIVE>
…
  <FORM name="<对话框名称>">                用于用户窗口的标签
    <INIT>
      <CONTROL name = "编辑 1" .../>       用于输入栏的标签
    </INIT>
    <PAINT>                                 用于文本显示或图片显示的标签
      <TEXT>大家好! </TEXT>
    </PAINT>
  </FORM>

</DEVICE>
…
</AGM>
```

3.6.8 SOFTKEY_OK, SOFTKEY_CANCEL

说明

标签 SOFTKEY_OK 的用途在于，在按下软键”OK”关闭窗口时覆盖标准属性。 标签 SOFTKEY_CANCEL 的用途在于，在按下软键"CANCEL"关闭窗口时覆盖标准属性。

在该标签内，可以执行以下功能：

- 数据复制
- 有条件处理
- 循环处理

编程

标签：	**SOFTKEY_OK**
句法：	<SOFTKEY_OK>
	...
	</SOFTKEY_OK>
标签：	**SOFTKEY_CANCEL**
句法：	<SOFTKEY_CANCEL>
	...
	</SOFTKEY_CANCEL>

索引

E

Easy Extend, 293
Easy XML 诊断, 35

X

XML
    XML 特殊字符, 71
    句法, 70
    系统变量, 73
    运算符, 72
XML 标签
    AGM, 303
    ARC, 125
    AUTOSCALE_CONTENT, 76
    BOX, 128
    BREAK, 38
    CAPTION, 88, 235, 249
    CHANNEL_CHANGED, 86
    CLOSE, 88
    CLOSE_FORM, 89
    CONTROL, 38, 90, 239, 246
    CONTROL_RESET, 38, 306
    CREATE_CYCLE, 140
    CREATE_CYCLE_EVENT, 39
    DATA, 40
    DATA_ACCESS, 102
    DATA_LIST, 41
    DEBUG, 103
    DEBUG_MSG, 42
    DEVICE, 303
    DIALOGGUI, 41
    DO_WHILE, 42
    DYNAMIC_INCLUDE, 43
    EDIT_CHANGED, 104
    ELLIPSE, 129
    ELSE, 43
    FILE, 306
    FOCUS_IN, 87
    FOR, 44
    FORM, 44, 75
    FUNCTION, 131, 244
    FUNCTION_BODY, 132
    GESTURE_EVENT, 104, 227
    HEPL_CONTEXT, 101
    HMI_RESET, 44
    IF, 45
    IMG, 121
    INCLUDE, 46
    INDEX_CHANGED, 104
    INIT, 79
    KEY_EVENT, 80
    LANGUAGE_CHANGED, 86
    LAYOUT, 77
    LET, 47, 280
    LINE, 133
    LOCK_OPERATING_AREA, 52
    MENU, 105
    MESSAGE, 82, 231
    MOUSE_EVENT, 81
    MSG, 52
    MSGBOX, 52
    NAME, 303
    NAVIGATION, 105
    NC_INSTRUCTION, 139
    OP, 53
    OPEN_FORM, 106
    OPERATION, 54
    OPTION_MD, 308
    PAINT, 87
    PASSWORD, 55
    PLC_INTERFACE, 308
    POWER_OFF, 55, 309
    PRINT, 56
    PROGRESS_BAR, 57
    PROPERTY, 107, 233, 237, 245
    RECALL, 137
    REQUEST, 136
    RESIZE, 84
    SEND_MESSAGE, 58, 83
    SET_ACTIVE, 304
    SET_INACTIVE, 304
    SHOW_CONTROL, 59
    SLEEP, 59
    SOFTKEY, 115, 279
    SOFTKEY_ACCEPT, 120
    SOFTKEY_BACK, 119
    SOFTKEY_CANCEL, 119, 311
    SOFTKEY_OK, 118, 311
    START_UP, 303
    STOP, 59
    SWITCH, 60
    SWITCHTOAREA, 61
    SWITCHTODYNAMICTARGET, 61
    TEST, 304

TEXT, 120
TEXTCONTEXT, 76
TEXTFILE, 76
THEN, 61
TIMER, 87
TYPE_CAST, 62
TYPEDEF, 63, 279
UID, 304
UNLOCK_OPERATING_AREA, 64
UPDATE_CONTROLS, 137
VERSION, 304
WAITING, 65, 309
WHILE, 65
XML_PARSER, 66
XML 功能
imagebox, 228
ImageBoxSet, 229
XML 函数
Abs, 214
AddItem, 202
ARCOS, 194
ARCSIN, 193
ARCTAN, 194
CEIL, 215
Cosinus, 192
DeleteItem, 204
Empty, 207
Exist, 199
FLOOR, 215
Get cursor selection, 208
getfocus, 207
GetItem, 209
GetItemData, 210
ImageBoxGet, 212
ImageBoxSet, 98, 211, 212
ISNUMERIC, 224
LoadItem, 205, 206
LOG, 215
LOG10, 215
MAX, 215
MIN, 215
MMC, 217
NC 程序选择, 200
Ncfunc Get MD limits, 169
PI 服务, 160, 162
POW, 215
RANDOM, 216
ROOT, 224
ROUND, 214
SDEG, 214
Set cursor selection, 209
Sinus, 192
SQRT, 214
SRAD, 214
String find, 184
String GetAt, 186
String pixel height, 187
String pixel width, 188
String reverse find, 185
String SetAt, 189
String Split, 190
Tangens, 193
本地时间控件, 173
标题栏高度, 214
不区分大/小写的字符串比较, 177
插入项, 203
插入字符串, 182
窗体颜色控件, 172
从 bico 转换为 int 的数控函数, 170
读取文件, 195
复制和读取值, 158
复制和写入值, 159
基于通道的 PI 服务, 163
将 int 转换为 bico 的数控函数, 170
控件存在, 173
控件可见, 174
控件设为工作区域, 175
控件设为已更改, 174
控件已更改, 174
口令函数, 171
屏幕分辨率, 213
屏幕分辨率 HMI, 213
删除控件, 201
删除文件, 197
删除字符串, 181
删除字符串一定数量的字符, 183
设置活动项, 208
设置位, 200
提取脚本文件部分, 198
替代字符串, 181
通过驱动器名称调用数控功能, 167
通过轴名称调用数控功能, 165
通过轴下标调用数控功能, 166
通过总线地址调用数控功能, 168
位图尺寸, 213
显示器分辨率, 164
写文件, 196
有效的 bico 字符串, 171
字符串比较, 176, 177
字符串右侧, 179
字符串长度, 180
字符串中间, 180
字符串左侧, 178
XML 诊断, 35

操

操作树, 19

创

创建调试对话框, 293

登

登入软键, 19

配

配置文件, 19

手

手指手势, 225

文

文件
    struct_def.xml, 142

SIEMENS

SINUMERIK 840D sl
SINUMERIK Integrate Run
MyScreens

编程手册

前言

基本安全说明 1

前言 2

基本操作 3

基本知识 4

对话框 5

变量 6

编程指令 7

图形单元和逻辑单元 8

操作区“Custom（定制）” 9

对话框选择 10

循环屏幕示例 11

在 Sidescreen 中设计 12

在 Display Manager 中设计 13

Referenz - Listen A

提示和技巧 B

动画元素 C

文档一览 D

适用于

CNC 软件　版本 4.94

10/2020
A5E40869575F AD

法律资讯

警告提示系统

为了您的人身安全以及避免财产损失，必须注意本手册中的提示。人身安全的提示用一个警告三角表示，仅与财产损失有关的提示不带警告三角。警告提示根据危险等级由高到低如下表示。

⚠ 危险
表示如果不采取相应的小心措施，**将会**导致死亡或者严重的人身伤害。

⚠ 警告
表示如果不采取相应的小心措施，**可能**导致死亡或者严重的人身伤害。

⚠ 小心
表示如果不采取相应的小心措施，可能导致轻微的人身伤害。

注意
表示如果不采取相应的小心措施，可能导致财产损失。

当出现多个危险等级的情况下，每次总是使用最高等级的警告提示。如果在某个警告提示中带有警告可能导致人身伤害的警告三角，则可能在该警告提示中另外还附带有可能导致财产损失的警告。

合格的专业人员

本文件所属的产品/系统只允许由符合各项工作要求的**合格人员**进行操作。其操作必须遵照各自附带的文件说明，特别是其中的安全及警告提示。 由于具备相关培训及经验，合格人员可以察觉本产品/系统的风险，并避免可能的危险。

按规定使用 Siemens 产品

请注意下列说明：

⚠ 警告
Siemens 产品只允许用于目录和相关技术文件中规定的使用情况。如果要使用其他公司的产品和组件，必须得到 Siemens 推荐和允许。正确的运输、储存、组装、装配、安装、调试、操作和维护是产品安全、正常运行的前提。必须保证允许的环境条件。必须注意相关文件中的提示。

商标

所有带有标记符号 ® 的都是 Siemens AG 的注册商标。本印刷品中的其他符号可能是一些其他商标。若第三方出于自身目的使用这些商标，将侵害其所有者的权利。

责任免除

我们已对印刷品中所述内容与硬件和软件的一致性作过检查。然而不排除存在偏差的可能性，因此我们不保证印刷品中所述内容与硬件和软件完全一致。印刷品中的数据都按规定经过检测，必要的修正值包含在下一版本中。

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
德国

A5E40869575F AD
Ⓟ 10/2020 本公司保留更改的权利

Copyright © Siemens AG 2020.
保留所有权利

前言

SINUMERIK 文献

SINUMERIK 文档分为以下几个类别：

- 通用文档/产品样本
- 用户文档
- 制造商/服务文档

其它信息

访问下面的网址 (https://support.industry.siemens.com/cs/de/en/view/108464614)获取有关该主题的信息：

- 订购文档/查看印刷品一览
- 进入下载文档的链接
- 使用在线文档（查找搜索手册/信息）

如果您对技术文档有疑问（例如建议、修改），请发送一份电子邮件到以下地址 (mailto:docu.motioncontrol@siemens.com)。

mySupport/文档

您可以访问下面的网址 (https://support.industry.siemens.com/My/cn/zh/documentation)，了解如何随意组合西门子文档内容，再结合机器，创建自己的机器文档。

培训

通过以下地址 (https://www.siemens.com/sitrain)可获取有关 SITRAIN 的信息 - 西门子为驱动和自动化产品、系统和解决方案制定的培训。

常见问题

常见问题（FAQ）请参见产品支持 (https://support.industry.siemens.com/cs/de/en/ps/faq) 下的服务&支持页面。

SINUMERIK

有关 SINUMERIK 的信息请访问以下网址 (https://www.siemens.com/sinumerik)。

目标使用人群

本手册供编程人员和设计人员使用。

优点

利用该编程手册目标用户可以设计、编写、测试程序和软件界面以及排除故障。

标准功能范畴

本文档描述了标准功能范畴。机床制造商增添或者更改的功能，由机床制造商资料进行说明。

控制系统有可能执行本文档中未描述的某些功能。但是这并不意味着在提供系统时必须带有这些功能，或者为其提供有关的维修服务。

同样，因为只是概要，所以本文档不包括全部类型产品的所有详细信息，也无法考虑到安装、运行和维修中可能出现的所有情况。

有关数据保护准则的说明

西门子遵守数据保护准则，特别是数据最小化原则（privacy by design）。对于该产品的具体含义是：

产品不会处理/存储个人相关数据，技术功能数据除外（例如时间戳）。用户如果将此类数据与其他数据（例如排班表）关联或者将个人相关数据存储在同一介质（例如硬盘）上而产生个人相关性，则应由用户自行确保遵循数据安全法规。

技术支持

访问网址 (https://support.industry.siemens.com/cs/cn/zh/sc/2090)下的“联系方式”， 您便可以获取各个国家技术支持的电话号码。

如需咨询技术疑问，请使用“支持请求”一栏下的在线表格。

目录

前言 **3**

1 基本安全说明 **13**

1.1 一般安全说明 13

1.2 应用示例的质保规定 14

1.3 安全性信息 15

2 前言 **17**

3 基本操作 **21**

3.1 简介 21

3.2 项目示例 22

3.2.1 任务说明 22

3.2.2 创建配置文件（示例） 26

3.2.3 将配置文件保存到 OEM 目录下（示例） 29

3.2.4 创建在线帮助（示例） 30

3.2.5 集成在线帮助并将文件保存到 OEM 目录下（示例） 33

3.2.6 将 “easyscreen.ini” 复制到 OEM 目录中（示例） 34

3.2.7 申明 “easyscreen.ini” 中的 COM 文件（示例） 34

3.2.8 测试项目（示例） 35

4 基本知识 **37**

4.1 设计文件的结构 37

4.2 操作树的结构 39

4.3 登入软键的定义及功能 41

4.3.1 定义登入软键 41

4.3.2 登入软键的功能 42

4.4 错误处理（日志） 44

4.5 “easyscreen.ini” 的说明 46

4.6 针对改用 “Run MyScreens” 人员的说明 49

4.7 扩展配置句法 51

4.8 SmartOperation 及 MutliTouch 操作 53

5 对话框 **55**

5.1 对话框的结构和组成单元 55

5.1.1 定义对话框 55

5.1.2 定义对话框属性 58

5.1.3 定义对话框单元 65
5.1.4 定义多列对话框 67
5.1.5 口令对话框 69
5.1.6 使用图/图形 71
5.2 定义软键栏 72
5.2.1 运行时改变软键属性 74
5.2.2 和语言相关的文本 77
5.3 设计在线帮助 81
5.3.1 概览 81
5.3.2 生成 HTML 文件 82
5.3.3 创建帮助手册 86
5.3.4 将在线帮助集成至 SINUMERIK Operate 中 88
5.3.5 保存帮助文件 90
5.3.6 PDF 格式的帮助文件 91
6 变量 93
6.1 定义变量 93
6.2 应用举例 95
6.3 示例 1： 分配变量类型、文本、帮助画面、颜色、提示框 97
6.4 举例 2： 定义变量类型，极限值，属性，短文本位置 99
6.5 举例 3： 定义变量类型、预设、系统或者用户变量、输入/输出栏位置 100
6.6 示例 4： 转换栏和列表栏 101
6.7 示例 5： 图片显示 102
6.8 示例 6： 进度条 103
6.9 示例 7： 口令输入模式（星号） 106
6.10 变量参数 107
6.11 各个变量类型的详细说明 115
6.12 转换栏的详细说明 119
6.13 预设值的详细说明 121
6.14 短文本位置、输入输出栏位置的详细说明 122
6.15 使用字符串 124
6.16 变量 CURPOS 126
6.17 变量 CURVER 127
6.18 变量 ENTRY 128
6.19 变量 ERR 129
6.20 变量 FILE_ERR 130

6.21 变量 FOC 132
6.22 变量 S_ALEVEL 133
6.23 变量 S_CHAN 134
6.24 变量 S_CONTROL 135
6.25 变量 S_LANG 136
6.26 变量 S_NCCODEREADONLY 137
6.27 变量 S_RESX 和 S_RESY 138

7 编程指令 139

7.1 运算符 139
7.1.1 数学运算符 139
7.1.2 位运算符 143

7.2 方法 145
7.2.1 ACCESSLEVEL 145
7.2.2 CHANGE 146
7.2.3 CHANNEL 147
7.2.4 CONTROL 148
7.2.5 FOCUS 148
7.2.6 LANGUAGE 149
7.2.7 LOAD 150
7.2.8 UNLOAD 150
7.2.9 OUTPUT 151
7.2.10 PRESS 152
7.2.11 PRESS(ENTER) 153
7.2.12 PRESS(TOGGLE) 154
7.2.13 RESOLUTION 155
7.2.14 RESUME 155
7.2.15 SUSPEND 156
7.2.16 示例： 带 OUTPUT 方法的版本管理 157

7.3 功能 159
7.3.1 读写驱动参数： RDOP, WDOP, MRDOP 159
7.3.2 子程序调用(CALL) 162
7.3.3 定义块(//B) 163
7.3.4 检查变量（CVAR） 164
7.3.5 CLEAR_BACKGROUND 165
7.3.6 文件功能“Copy Program (CP，复制程序)” 165
7.3.7 文件功能“Delete Program (DP，删除程序)” 166
7.3.8 文件功能“Exist Program (EP，退出程序)” 167
7.3.9 文件功能“Move Program (MP，移动程序)” 168
7.3.10 文件功能“Select Program (SP，选择程序)” 169
7.3.11 文件存取： RDFILE、WRFILE、RDLINEFILE、WRLINEFILE 170

7.3.12 对话框行(DLGL) 173
7.3.13 DEBUG 174
7.3.14 退出对话框(EXIT) 175
7.3.15 转换栏或列表框栏的动态列表操作 176
7.3.16 评估(EVAL) 179
7.3.17 退出装载软键(EXITLS) 180
7.3.18 功能 (FCT) 181
7.3.19 生成代码(GC) 183
7.3.20 口令功能 185
7.3.21 装载数组(LA) 187
7.3.22 装载块（LB） 188
7.3.23 装载屏幕窗口 (LM) 189
7.3.24 装载软键(LS) 191
7.3.25 Load Grid (LG) 192
7.3.26 多次选择 SWITCH 193
7.3.27 多次读取 NC PLC (MRNP) 194
7.3.28 PI 服务 196
7.3.29 读取 (RNP) 和写入 (WNP) 系统或用户变量 197
7.3.30 RESIZE_VAR_IO 和 RESIZE_VAR_TXT 198
7.3.31 寄存器(REG) 199
7.3.32 RETURN 201
7.3.33 反编译 202
7.3.34 无注释反编译 203
7.3.35 向前/后查找(SF, SB) 207
7.3.36 字符串功能 208
7.3.37 WHILE/UNTIL 循环 215
7.3.38 循环执行脚本： START_TIMER，STOP_TIMER 217

8 图形单元和逻辑单元 219

8.1 线、分割线、矩形、圆形和椭圆形 219

8.2 定义数组 223
8.2.1 存取数组单元的值 224
8.2.2 举例： 存取数组单元 226
8.2.3 查询数组单元的状态 228

8.3 表格描述(栅格) 230
8.3.1 定义表格(栅格) 231
8.3.2 定义列 232
8.3.3 表格中的焦点控制（栅格） 234

8.4 自定义小部件 235
8.4.1 自定义小部件 235
8.4.2 自定义小部件库的结构 236
8.4.3 自定义小部件接口的结构 236
8.4.4 自定义小部件与对话框的互动 - 自动数据交换 238
8.4.5 自定义小部件与对话框的互动 - 手动数据交换 239

8.4.5.1 读取和写入属性 240
8.4.5.2 执行自定义小部件的方法 241
8.4.5.3 响应一个自定义小部件信号 245

8.5 SlEsGraphCustomWidget 248
8.5.1 SlEsGraphCustomWidget 248
8.5.2 性能说明 250
8.5.3 读取和写入属性 250
8.5.4 属性 251
8.5.5 功能 264
8.5.6 信号 282

8.6 SlEsTouchButton 284
8.6.1 SlEsTouchButton 284
8.6.2 读取和写入属性 285
8.6.3 属性 286
8.6.4 功能 307
8.6.5 信号 308
8.6.6 图片和文本的定位和对齐 311
8.6.7 和语言相关的文本 313

9 操作区"Custom（定制）" 315

9.1 这样激活操作区“自定义” 315

9.2 这样设计“自定义”软键 316

9.3 这样设计操作区“自定义” 317

9.4 操作区“自定义”的编程示例 318

10 对话框选择 323

10.1 通过 PLC 软键选择对话框 323
10.1.1 im 在程序编辑器中调用 Run MyScreens-循环窗口 324

10.2 通过 PLC 硬键选择对话框 326

10.3 通过 NC 选择对话框 329

11 循环屏幕示例 331

11.1 循环屏幕示例 331

12 在 Sidescreen 中设计 333

12.1 在一个单独的 Sidescreen 页面中显示一个 Run MyScreens 设计窗口 334

12.2 在一个单独的 Sidescreen 页面中显示多个 Run MyScreens 设计窗口 336

12.3 在一个现有 Sidescreen 页面中添加 Run MyScreens 设计窗口 338

12.4 在一个 Sidescreen 元素中添加 Run MyScreens 设计窗口 340

12.5 在 Sidescreen 中执行 Run MyScreens 设计窗口时的注意事项 343

13 在 Display Manager 中设计 345
13.1 在 Display Manager 中显示 Run MyScreens 设计窗口 345
13.2 集成到 SISideScreenDialog 中 346
13.3 集成到 SIWidgetsApp 中 347
13.4 在 Display Manager 中执行 Run MyScreens 设计窗口时的注意事项 348
A Referenz - Listen 349
A.1 登入软键表 349
A.1.1 车床登入软键表 349
A.1.2 铣床登入软键表 350
A.2 预定义软键列表 353
A.3 访问等级列表 354
A.4 颜色表 355
A.5 文件名中的语种缩写表 356
A.6 可用的系统变量列表 358
A.7 对话框打开方式（属性 CB） 359
B 提示和技巧 361
B.1 一般提示 361
B.2 DEBUG 的提示 363
B.3 CHANGE 方法的提示 364
B.4 DO-LOOP 循环的提示 366
C 动画元素 367
C.1 简介 367
C.2 建模 369
C.2.1 前提条件 369
C.2.2 建模规则 369
C.2.3 导入图形（模型） 371
C.2.4 建模模板 373
C.3 XML 指令 375
C.3.1 一览 375
C.3.2 场景说明文件的结构 375
C.3.3 镜像和旋转 378
C.3.4 视图类型 379
C.4 转换为 hmi 文件 380
C.5 在 Create MyHMI /3GL 中显示 381

C.5.1 X3D 浏览器 381
C.5.2 类 SlX3dViewerWidget 381
C.5.3 公共方法 381
C.5.4 公共槽 382
C.5.5 库 383
C.5.6 执行示例 383
C.5.7 机床数据 383
C.5.8 使用说明 383
C.6 在 Run MyScreens 中显示 384
D 文档一览 385
索引 387

基本安全说明 1

1.1 一般安全说明

⚠ 警告

未遵循安全说明和遗留风险可引发生命危险

忽视随附硬件文档中的安全说明和遗留风险会导致重伤或死亡。

- 遵守硬件文档中的安全说明。
- 进行风险评估时应考虑到遗留风险。

⚠ 警告

因参数设置错误或修改参数设置引起机器故障

参数设置错误可导致机器出现故障，从而导致人员重伤或死亡。

- 采取保护措施，防止未经授权的参数设置。
- 采取适当措施（如驻停或急停）处理可能出现的故障。

1.2 应用示例的质保规定

应用示例在组态和配置以及各种突发事件方面对设备没有强制约束力，无需一一遵循。应用示例不会提供客户专用的解决方案，仅在典型任务设置中提供保护。

用户自行负责上述产品的规范运行事宜。应用示例并没有解除您在应用、安装、运行和维护时确保安全环境的责任。

1.3 安全性信息

Siemens 为其产品及解决方案提供了工业信息安全功能，以支持工厂、系统、机器和网络的安全运行。

为了防止工厂、系统、机器和网络受到网络攻击，需要实施并持续维护先进且全面的工业信息安全保护机制。Siemens 的产品和解决方案构成此类概念的其中一个要素。

客户负责防止其工厂、系统、机器和网络受到未经授权的访问。只有在有必要连接时并仅在采取适当安全措施（例如，防火墙和/或网络分段）的情况下，才能将该等系统、机器和组件连接到企业网络或 Internet。

关于可采取的工业信息安全措施的更多信息，请访问 https://www.siemens.com/industrialsecurity (https://www.siemens.com/industrialsecurity)。

Siemens 不断对产品和解决方案进行开发和完善以提高安全性。Siemens 强烈建议您及时更新产品并始终使用最新产品版本。如果使用的产品版本不再受支持，或者未能应用最新的更新程序，客户遭受网络攻击的风险会增加。

要及时了解有关产品更新的信息，请订阅 Siemens 工业信息安全 RSS 源，网址为 https://www.siemens.com/industrialsecurity (https://new.siemens.com/global/en/products/services/cert.html#Subscriptions)。

其他信息请上网查找：

工业安全功能选型手册 (https://support.industry.siemens.com/cs/cn/zh/view/108862708/en)

⚠ 警告

篡改软件会引起不安全的驱动状态

篡改软件（如：病毒、木马、蠕虫等）可使设备处于不安全的运行状态，从而可能导致死亡、重伤和财产损失。

- 总是使用最新版本的软件。
- 将自动化和驱动组件集成到设备或机器上的整套先进工业信息安全方案中。
- 全面考虑整套工业信息安全方案中使用的所有产品。
- 采取相应的保护措施（如：使用杀毒软件）防止移动存储设备中的文件受到恶意软件的破坏。
- 在调试结束后，检查所有和安全相关的设置。

前言 2

概述

通过“Run MyScreens”机床制造商或最终用户可以自行设计扩展功能的操作界面，或者设计操作界面的布局。

还可以在其他循环调用下生成自定义的操作界面。或直接在控制系统上创建对话框。

“Run MyScreens”通过编译器和包含操作界面说明的配置文件实现。

通过 ASCII 文件配置“Run MyScreens”：该配置文件包含了关于操作界面的说明。创建文件所需的句法参见下列章节。

基本功能

通过“Run MyScreens”机床制造商可自定义对话框。基本功能支持在操作菜单树中设计 5 个对话框，或者用于设计用户自定义的循环对话框。



软件选件

扩展对话框数量，需要下列软件选件：

- SINUMERIK 828D/840D sl, SINUMERIK Integrate Run MyScreens (6FC5800-0AP64-0YB0)
- SINUMERIK 840D sl, SINUMERIK Integrate Run MyHMI / 3GL (6FC5800-0AP60-0YB0)
- SINUMERIK 840D sl, SINUMERIK Integrate Run MyHMI / WinCC (6FC5800-0AP61-0YB0)

边界条件

必须符合以下边界条件：

- 只允许在一个操作区内切换对话框。
- 用户变量不允许与系统变量或者 PLC 变量有相同的名称（另请参见参数手册系统变量/PGAsl/）
- 由 PLC 激活的对话框构成一个单独的操作区（和测量循环屏幕类似）。

工具

- UTF8 编辑器（例如：操作界面的集成编辑器或记事本）
- 创建图形/屏幕所需的图形程序。

文件名和编码

说明

使用 NCU 中的 HMI Operate 时要注意 CF 卡上的所有文件名都是以小写字母保存的（如：com、png、txt）。这是 Linux 操作系统的要求。

在 PCU 上您可以任意以大小写字母保存文件名。为便于之后传送到 Linux，此处仍建议只使用小写字母。

说明

保存配置文件和语言文件时请注意，应将您所使用的编辑器中的编码设为 UTF 8。

保存路径

保存配置文件、帮助文件等时应遵循以下规定：

[系统西门子目录]	
Linux：	/card/siemens/sinumerik/hmi
Windows 7：	C:\ProgramData\Siemens\MotionControl\siemens
[系统 oem 目录]	
Linux：	/card/oem/sinumerik/hmi
Windows 7：	C:\ProgramData\Siemens\MotionControl\oem
[系统用户目录]	
Linux：	/card/user/sinumerik/hmi
Windows 7：	C:\ProgramData\Siemens\MotionControl\user
[系统插件目录]	
Linux：	/card/addon/sinumerik/hmi

	Windows 7：	C:\ProgramData\Siemens\MotionControl\addon
保存位置		
	“Run MyScreens” 配置：	[系统 oem 目录]/proj
	配置文件：	[系统 oem 目录]/cfg
	语言文件：	[系统 oem 目录]/lng
	图形文件：	[系统 oem 目录]Ico/ico[分辨率] 示例： [系统 oem 目录]/Ico/ico640
	在线帮助：	[系统 oem 目录]/hlp/[语言] 示例： [系统 oem 目录]/hlp/eng

使用

可以实现以下功能：

- 显示对话框并提供：
  - 软键
  - 变量
  - 文本和帮助文本
  - 图形和帮助画面
- 通过以下方法调用对话框：
  - 按下（登入）软键
  - 选择 PLC/NC
- 动态重组对话框
  - 修改、删除软键
  - 定义并设计变量栏
  - 显示、更换、删除显示文本（和语言相关或无关）
  - 显示、更换、删除图形

- 在进行以下操作时触发动作：
  - 显示对话框
  - 输入数值（变量）
  - 按下软键
  - 关闭对话框
  - 系统或用户变量的值变化
- 对话框间的数据交换
- 变量：
  - 读取（NC 变量、PLC 变量、用户变量）
  - 写入（NC 变量、PLC 变量、用户变量）
  - 和数学、比较或者逻辑运算符相连
- 执行下列功能：
  - 子程序
  - 文件功能
  - PI 服务
- 根据用户组考虑保护等级

基本操作 3

3.1 简介

根据以下示例可了解使用 Run MyScreen 在 SINUMERIK Operate 操作界面中添加自定义对话框的必要操作步骤。 此外还可了解如何创建自定义对话框、如何添加上下文有关的帮助图形和帮助调用框、如何定义软键以及如何在对话框之间进行导航。

3.2 项目示例

3.2.1 任务说明

在项目示例中创建下述对话框，包含上下文相关的帮助信息。

对话框 1

第一个对话框中显示可写 R 参数（0 和 1）和几何轴名称（输入栏）。两个 R 参数均链接有相应的帮助图形。几何轴名称未链接上下文有关的帮助信息。此外对话框中还包含诸如分割线（水平和垂直）、转换栏、集成了单位选择框的输入/输出栏和进度条（变色或不变色）等元素。

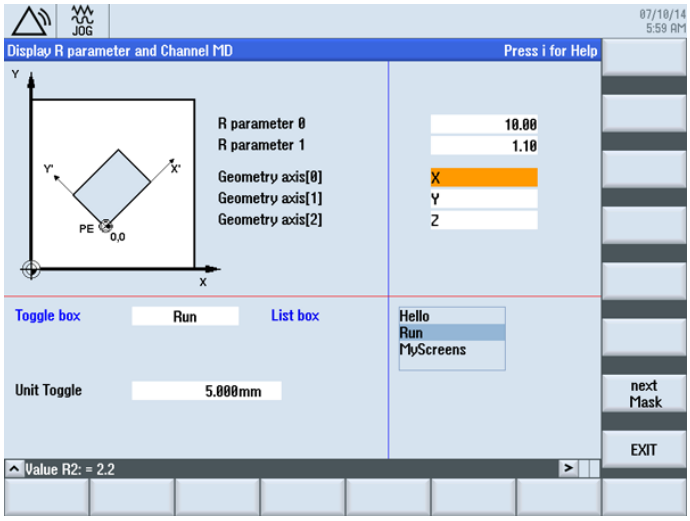


图 3-1 带辅助图的 R 参数

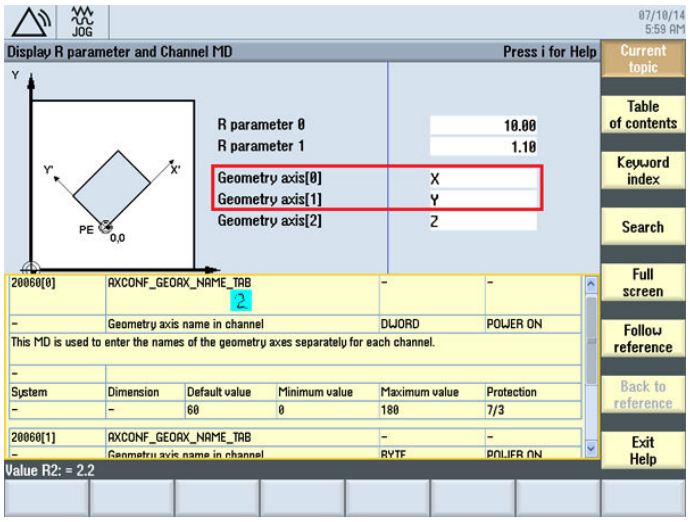


图 3-2　带上下文相关帮助信息的几何轴

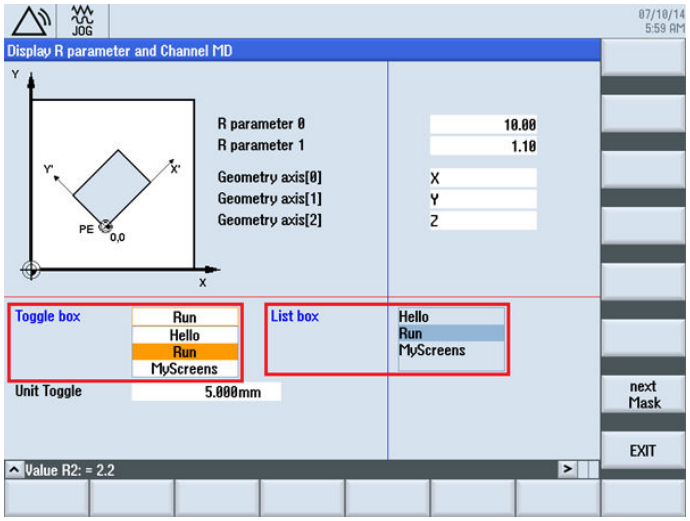


图 3-3　转换栏：列表和选项框

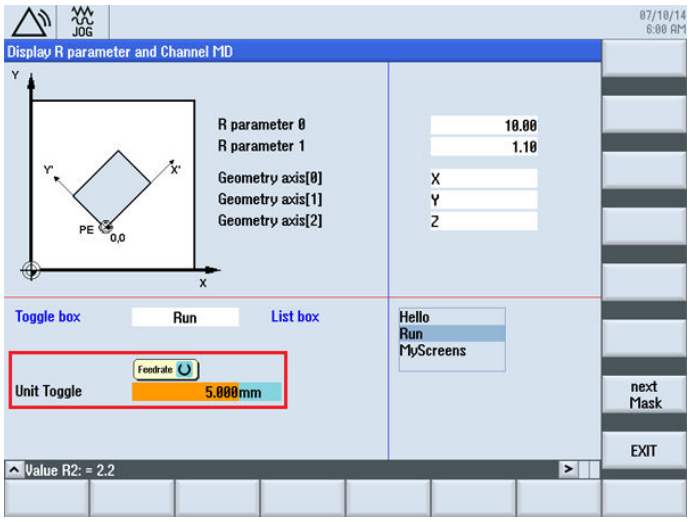


图 3-4　集成了单位选择框的输入/输出栏

对话框 2

第二个对话框中显示 MCS 值和 WCS 值。此外对话框中还含有变色或不变色的进度显示。

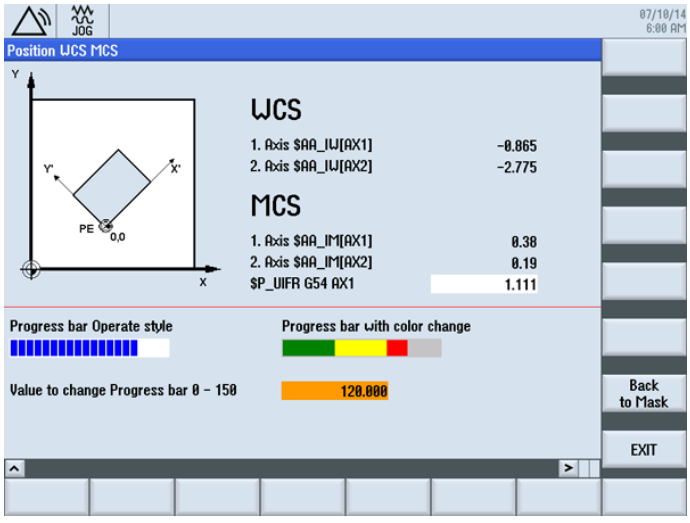


图 3-5　变色（右）和不变色（左）的进度条

浏览

通过“诊断”操作区中的 “START” 登入软键调用第一个对话框。使用水平软键 SK7。

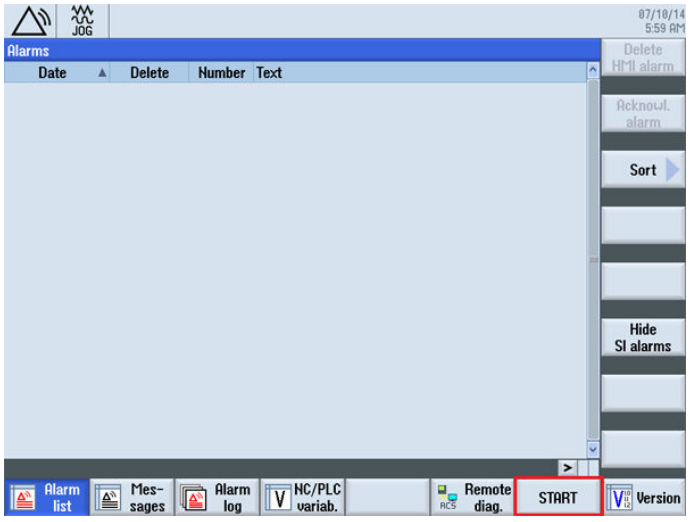


图 3-6　AUTO 运行模式下“加工”操作区的登入软键„START“

在第一个对话框中使用软键 “next Mask” 可调用第二个对话框，使用软键 “EXIT” 返回操作区的基本画面（见上图）。

在第二个对话框中使用软键 “EXIT” 也同样能返回操作区的基本画面（见上图）。使用软键 “Back to Mask” 可以返回第一个对话框。

步骤

以下章节中说明了必要的操作步骤：

1. 创建配置文件（COM 文件） (页 26)
2. 将配置文件保存到 OEM 目录下 (页 29)
3. 创建在线帮助 (页 30)
4. 集成在线帮助并将文件保存到 OEM 目录下 (页 33)
5. 将 easyscreen.ini 复制到 OEM 目录中 (页 34)
6. 申明 easyscreen.ini 中的 COM 文件 (页 34)
7. 测试项目 (页 35)

3.2.2 创建配置文件（示例）

配置文件的内容

在 UTF8 编辑器中为两个对话框创建配置文件 "diag.com" 。

```
; 登入软键开始标记
//S(START)
   ; 纯文本登入软键
   HS7=("START")
   ; 登入软键，带和语言相关的文本和 png
   ;HS7=([$80792,"\\sk_ok.png"])

   ; PRESS 方法
   PRESS(HS7)
      ; 功能 LM 或者 LS
      LM("MASK1")
      ; 带 com 文件说明的 LM
      LM("MASK1","TEST.COM")
   END_PRESS
; 登入软键结束标记
//END

; 带标题和图像的对话框 1 的定义
//M(MASK1/"Display R parameter and Channel MD"/"mz961_01.png")
   ; 变量定义
   DEF VAR1 = (R2///,"R parameter 0"///"$R[0]"/200,50,150/400,50,100,)
   ; 带帮助图形
   DEF VAR2 = (R2///,"R parameter 1"//"mz961_02.png"/"$R[1]"/200,70,150/400,70,100)
   ; 带在线帮助
   DEF ACHS_NAM1 = (S///"Press i for Help","Geometry axis[0]"////
200,100,150/400,100,100//"sinumerik_md_1.html","20060[0]")
   ; 带在线帮助
   DEF ACHS_NAM2 = (S///"Press i for Help","Geometry axis[1]"////
200,120,150/400,120,100//"sinumerik_md_1.html","20060[1]")
   DEF ACHS_NAM3 = (S///,"Geometry axis[2]"////200,140,150/400,140,100)
   ; 转换栏和单位转换栏的定义
   DEF VAR_TGL = (S/* "Hello", "Run", "MyScreens"/"Run"/,"Toggle box"////10,230,100/120,230,100/,,6)

   DEF VAR_TGB = (S/* "Hello", "Run", "MyScreens"/"Run"/,"List box"/
dt4///250,230,100/370,230,100,60/,,"#0602ee")
   ; BC, FC, BC_ST, FC_ST, BC_GT, FC_GT, BC_UT, FC_UT, SC1, SC2
   DEF VarEdit = (R///,"Unit Toggle",,,"Feedrate"///"$R[11]"/5,300,100/120,300,100///"VarTgl"),
VarTgl = (S/*0="mm",1="inch"/0//WR2////220,300,40)
```

```
; 对话框中的软键定义
HS1=("")
HS2=("")
HS3=("")
HS4=("")
HS5=("")
HS6=("")
HS7=("")
HS8=("")
VS1=("")
VS2=("")
VS3=("")
VS4=("")
VS5=("")
VS6=("")
VS7=("next Mask")
VS8=("EXIT")

; LOAD 块定义
LOAD
   ; 通过 RNP 读取值
   ACHS_NAM1 = RNP("$MC_AXCONF_GEOAX_NAME_TAB[0]")
   ACHS_NAM2 = RNP("$MC_AXCONF_GEOAX_NAME_TAB[1]")
   ACHS_NAM3 = RNP("$MC_AXCONF_GEOAX_NAME_TAB[2]")
   ; 对话框行输出
   DLGL("Value R2: = " << RNP("$R[2]"))
   ; 通过 WNP 写入值
   WNP("$R[3]",VAR0)

  ; 垂直分割线
  V_SEPARATOR(360,1,6,1)
  ; 水平分割线
  H_SEPARATOR(220,1,7,1)
END_LOAD

; PRESS 方法
PRESS(VS7)
   ; 载入其他对话框
   LM("MASK2")
; PRESS 方法的结束标记
END_PRESS

; PRESS 方法
PRESS(VS8)
```

```
      ；退出对话框
      EXIT
   ；PRESS 方法的结束标记
   END_PRESS
；对话框 1 的结束标记
//END

；带标题和图像的对话框 2 的定义
//M(MASK2/"Position WCS MCS"/"mz961_01.png")
   ；变量定义
   DEF TEXT1 = (I///,"WCS"/WR0,fs2///230,30,120/,,1)
   DEF VAR1 = (R3///,"1. Axis $AA_IW[AX1]"/WR1//"$AA_IW[AX1]"/230,70,150/400,70,100)
   DEF VAR2 = (R3///,"2. Axis $AA_IW[AX2]"/WR1//"$AA_IW[AX2]"/230,90,150/400,90,100)

   DEF TEXT2 = (I///,"MCS"/WR0,fs2///230,120,120/,,1)
   DEF VAR3 = (R2///,"1. Axis $AA_IM[AX1]"/WR1//"$AA_IM[AX1]"/230,160,150/400,160,100)
   DEF VAR4 = (R2///,"2. Axis $AA_IM[AX2]"/WR1//"$AA_IM[AX2]"/230,180,150/400,180,100)
   DEF VAR5 = (R3///,"$P_UIFR G54 AX1"///"$P_UIFR[1,AX1,TR]"/230,200,150/400,200,100)

   ；进度条的定义
   DEF PROGGY0 = (R/0,150//,"Progress bar Operate style"/DT2,DO0//"$R[10]"/5,240,190/5,260,150/6,10)
   DEF PROGGY4 = (R/0,150,50,100/120/,"Progress bar with color change"/DT1,DO0//"$R[10]"/
260,240,190/260,260,150/3,4,,,9,7)

   ；变量定义
   DEF PROGVAL = (R/0,150//,"Value to change Progress bar 0 - 150"///"$R[10]"/5,300,230/260,300,100)

  ；对话框中的软键定义
  HS1=("")
  HS2=("")
  HS3=("")
  HS4=("")
  HS5=("")
  HS6=("")
  HS7=("")
  HS8=("")
  VS1=("")
  VS2=("")
  VS3=("")
  VS4=("")
  VS5=("")
  VS6=("")
  VS7=("Back to Mask")
  VS8=("EXIT")
```

```
   ; Load方法
   LOAD
      H_SEPARATOR(230,1,7,1)
   END_LOAD

   ; 更改方法
   CHANGE(VAR5)
      ; PI_START 功能
      PI_START("/NC,201,_N_SETUFR")
   ; 更改方法的结束标记
   END_CHANGE

   ; PRESS 方法
   PRESS(RECALL)
      ; 返回主调对话框
      LM("MASK1")
   ; PRESS 方法的结束标记
   END_PRESS

   ; PRESS 方法
   PRESS(VS7)
      ; 返回主调对话框
      LM("MASK1")
   ; PRESS 方法的结束标记
   END_PRESS

   ; PRESS 方法
   PRESS(VS8)
      ; 退出对话框，返回到标准应用程序
      EXIT
   ; PRESS 方法的结束标记
   END_PRESS
; 对话框的结束标记
//END
```

3.2.3 将配置文件保存到 OEM 目录下（示例）

存放路径

将配置文件“diag.com”保存到以下路径中：

[系统 oem 目录]/proj

3.2.4 创建在线帮助（示例）

创建 HTML 文件“sinumerik_md_1.html”。“在线帮助的内容”中的示例通过 **name="20060[0]"** 和 **name="20060[1]"** 显示上下文相关的帮助调用框。

提示

创建 HTML 文件的说明参见章节生成 HTML 文件 (页 82)。

如何加入在线帮助的说明参见章节集成在线帮助并将文件保存到 OEM 目录下（示例） (页 33)。

在线帮助的内容

示例无法添加注释。

```
<html><head><meta http-equiv="Content-Type" content="text/html; charset="UTF-8"/><title></
title></head>
   <body>
      <table border="1" cellspacing="0" cellpadding="2" width="100%">
         <tr>
            <td><a name="20060[0]"><b>20060[0]</b></a></td>
            <td colspan="3">AXCONF_GEOAX_NAME_TAB</td>
            <center>
               <img src="../picture/bitmap3.png">
            </center>
            <td>-</td>
            <td>-</td>
         </tr>
         <tr>
            <td>-</td>
            <td colspan="3">Geometry axis name in channel</td>
            <td>DWORD</td>
            <td>POWER ON</td>
         </tr>
         <tr>
            <td colspan="6">This MD is used to enter the names of the geometry axes
separately for each channel.<br />
         <tr>
            <td>-</td>
            <td colspan="5"> </td>
         </tr>
         <tr>
          <td width="16*">System</td><td width="16*">Dimension</td><td width="16*">Default
value</td>
            <td width="16*">Minimum value</td><td width="16*">Maximum value</td>
            <td width="16*">Protection</td>
         </tr>
         <tr>
            <td>-</td>
            <td>-</td>
            <td>60</td>
            <td>0</td>
            <td>180</td>
            <td>7/3</td>
         </tr>
      </table>
      <p></p>
      <table border="1" cellspacing="0" cellpadding="2" width="100%">
         <tr>
            <td><a name="20060[1]"><b>20060[1]</b></a></td>
            <td colspan="3">AXCONF_GEOAX_NAME_TAB</td>
            <td>-</td>
            <td>-</td>
         </tr>
         <tr>
```

```
            <td>-</td>
            <td colspan="3">Geometry axis name in channel</td><td>BYTE</td>
            <td>POWER ON</td>
         </tr>
         <tr>
            <td colspan="6">This MD is used to enter the names of the geometry axes
separately for each channel.<br />
         <tr>
            <td>-</td>
            <td colspan="5"> </td>
         </tr>
         <tr>
            <td width="16*">System</td>
            <td width="16*">Dimension</td>
            <td width="16*">Default value</td>
            <td width="16*">Minimum value</td>
            <td width="16*">Maximum value</td>
            <td width="16*">Protection</td>
         </tr>
         <tr>
            <td>-</td>
            <td>-</td>
            <td>0</td>
            <td>0</td>
            <td>2</td>
            <td>7/3</td>
         </tr>
      </table>
      <p></p>
   </body>
</html>
```

3.2.5 集成在线帮助并将文件保存到 OEM 目录下（示例）

集成在线帮助

集成在线帮助的句法与 SINUMERIK Operate 相似：

DEF RFP=(R//1/,"RFP","RFP"///////"sinumerik_md_1.html","9006")

说明

LINUX 系统要求 HTML 文件名称必须以小写字母保存！

存放路径

将 HTML 文件 "sinumerik_md_1.html" 的德语帮助信息保存到以下路径下：

[系统 oem 目录]/hlp/deu

必要时，需要保存其他语言文件夹（例如：chs、eng、esp、fra、ita ...）。

语种缩写列表参见附录“参考列表”。

其它信息

有关创建 OEM 特定的在线帮助的详细信息请参考设计在线帮助 (页 81)一章。

3.2.6 将 “easyscreen.ini” 复制到 OEM 目录中（示例）

存放路径

从以下目录将文件 "easyscreen.ini"

[系统西门子目录]/cfg

复制到

[系统 oem 目录]/cfg。

3.2.7 申明 “easyscreen.ini” 中的 COM 文件（示例）

“easyscreen.ini” 中的调整

在 OEM 目录下的 “easyscreen.ini” 中进行如下修改，以便申明配置文件 “diag.com”。

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

3.2.8 测试项目（示例）

测试对话框调用

转换至“诊断”操作区。按下水平软键 “START”。

如果 “Run MyScreens” 编译配置文件时出现错误，则该错误会保存在文本文件 easyscreen_log.txt 中。更多信息参见错误处理（日志） (页 44)一章。

测试上下文相关的帮助图形和在线帮助

测试上下文相关的帮助图形和在线帮助。如果出现错误请检查保存路径。

基本知识 4

4.1 设计文件的结构

简介

新操作界面的描述保存在配置文件中。这些文件自动编译并显示屏幕上的结果。在供货时并不提供配置文件，因此必须首先创建此文件。

创建配置文件和语言文件时使用 UTF 8 编辑器（例如：操作界面的集成编辑器或记事本）。描述可以通过注释来进一步说明。注释字符前会添加“；”。

说明

保存配置文件和语言文件时请注意，应将您所使用的编辑器中的编码设为 UTF 8。

然而，关键字只允许由 ASCII 字符集中的字符组成（UTF 8 编译的 COM 文件中也一样）。否则就无法保证编译和屏幕/对话框的显示。

配置

每个 HMI 操作区都包含固定的登入软键，以进入新创建的对话框。

调用配置文件中的“载入屏幕” (LM) 或者“载入软键栏” (LS)可以重新命名被调对象所在的文件。采用这种方式可以对配置文件进行分类，例如：同一操作级的所有功能在一个单独的配置文件中。

配置文件的结构

配置文件由以下单元组成：

1. 登入软键说明
2. 对话框定义
3. 变量定义
4. 方法说明
5. 软键栏定义

说明

顺序

必须遵循上文列出的各单元在配置文件中的顺序。

示例：

```
//S (START)                    ; 登入软键定义（可选）
....
//END
//M (.....)                    ; 对话框定义
DEF .....                      ; 变量定义
LOAD                           ; 块描述
...
END_LOAD
UNLOAD
...
END_UNLOAD
...
//END
//S (...)                      ; 软键条定义
//END
```

配置文件的保存

配置文件保存在目录 [系统用户目录]/proj 下，也相应的保存在目录 [系统插件目录] 和 [系统 oem 目录] 下。

转换其他 HMI 应用程序中的文本

从 Codepage 编码的文本文件向 UTF-8 编码文本转换的步骤：

1. 在 PG/PC 上的文本编辑器中打开文本文件。
2. 保存时选择 UTF 8 编码。

使用 Codepage 编码的读取机制还继续被支持。

4.2 操作树的结构

操作树的原理

多个相互关联的对话框构成了一个操作树。如果能从一个对话框切换入另一个对话框，则表示这两个对话框间存在关联。点击在这种对话框内重新定义的水平或者垂直软键，可以返回上级对话框或者进入任意一个对话框。

在每个登入软键下都可以生成一个操作树：

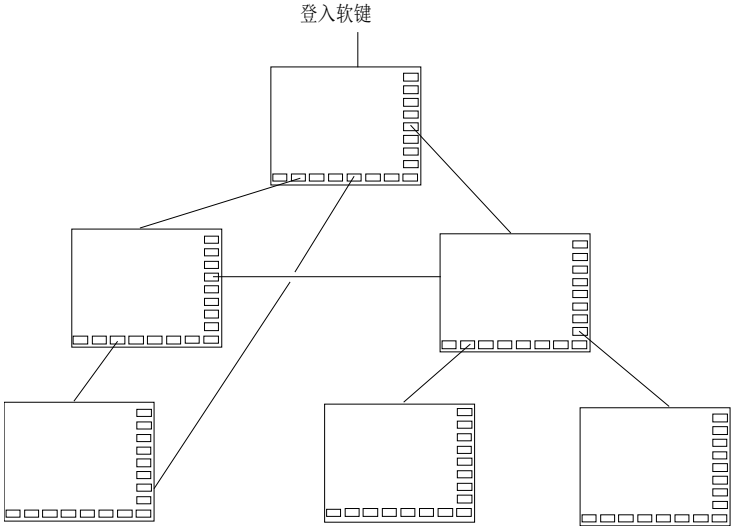


图 4-1 操作树

登入软键

在一个 “easyscreen.ini” 中指明的配置文件中，定义了一个或者多个软键（登入软键），这些软键可以作为操作的出发点。

软键的定义决定执行下一步动作的自定义对话框或者其他软键栏。

按下登入软键则载入所属的对话框。同时，属于对话框的相应软键激活。如果没有设计特定位置，则在标准位置上给出变量。

返回到标准应用程序

可以退出新定义的对话框并返回到标准操作区。

通过<RECALL>（回调）键可以退出新定义的操作界面，如果这个按键还未被设计用于其它用途。

说明

在 PLC 用户程序中调用对话框

除了软键，也可以由 PLC 选择对话框。DB19.DBB10 中的接口信号用于 PLC → HMI 的信号交换。

4.3 登入软键的定义及功能

4.3.1 定义登入软键

和对话框无关的软键

登入软键是和对话框无关的软键，它不由对话框调用，而是在第一个新对话框**之前**设计。为访问一个登入对话框或者一个登入软键栏，必须对登入软键进行定义。

编程

登入软键的描述块如下构建：

```
//S(Start)                    ; 登入软键开始标记
HS1=(...)                     ; 定义登入软键：水平软键 SK 1
PRESS(HS1)                    ; 方法
  LM...                       ; 功能 LM 或者 LS
END_PRESS                     ; 方法结束
//END                         ; 登入软键结束标记
```

登入软键的允许位置

在操作区中“Run MyScreens”登入软键的允许位置有：

操作区	位置
机床	HSK6
参数	HSK7
程序	HSK6 和 HSK15 测量循环：HSK13 和 HSK14
程序管理器	HSK2-8 和 HSK12-16，如未被驱动器占用。
诊断	HSK7
调试	HSK7

登入软键在专门的文件中设计。文件的名称在文件 "easyscreen.ini" 中声明。通常为一个操作区的名称（例如：调试区为“startup.com”）。加工区是特例。在加工区中，针对不同运行方式有不同文件（“ma_jog.com”，“ma_auto.com”）。

带有登入软键的软键栏叫“Start”。现有的登入软键设计仍可使用。登入软键与登入软键菜单中各个 HMI 应用（操作区）中软键的汇总功能（Merge）还不被支持。

直到首次对话框调用时，即从全部功能（例如，执行 PRESS 块）都可用的时刻起，才能完全替换菜单或软键栏。

登入软键的配置模板

有关登入软键的全部允许位置的详细描述及其设计见以下目录中的文件“easyscreen.ini”：

[系统西门子目录]/cfg

该文件用作登入软键的自定义配置模板。

另见

登入软键表

4.3.2 登入软键的功能

和对话框无关的软键的功能

通过登入软键只可以触发特定的功能。

允许下列功能：

- 通过**功能 LM** 可以装载另一个对话框。**LM**("名称"[,"文件"])
- 通过**功能 LS** 可以显示另一个软键栏。 **LS**("名称"[, "文件"][, 合并])
- 通过**功能 "EXIT"**可以离开新配置的操作界面并返回标准应用程序。
- 通过**功能 "EXITLS"**可以离开当前的操作界面并装载一个定义的软键栏。

PRESS 方法

在描述块内定义软键并在方法 PRESS 内分配功能 "LM" 或者 "LS"。

如果登入软键定义标记为注释（在行开始处用分号;）或者已删除配置文件，则登入软键无效。

```
//S(Start)                          ; 开始标记
HS6=(“第 1 屏幕”)                    ; 水平软键 6，文字为“第 1 屏幕”
```

```
PRESS(HS6)                          ; 水平软键 6 的 PRESS 方法
  LM("屏幕 1")                      ; 装载功能屏幕 1，此时必须在同一文件内定义了屏幕 1。
END_PRESS                           ; PRESS 方法结束标记
HS7=("第 2 屏幕")                   ; 水平软键 7，文字为“第 2 屏幕”
PRESS(HS7)                          ; 水平软键 7 的 PRESS 方法
  LM("屏幕 2")                      ; 装载功能屏幕 2，此时必须在同一文件内定义屏幕 2。
END_PRESS                           ; PRESS 方法结束标记
//END                               ; 登入块的结束标记
```

示例

```
HS1 = ("新的软键栏")
HS2 = ("没有功能")
PRESS (HS1)
  LS("软键栏 1")                    ; 载入新的软键栏
END_PRESS
PRESS (HS2)                         ; 空的 PRESS 方法
END_PRESS
```

各种登入软键设计

各个登入软键的设计会汇总在一起。其中，首先从“easyscreen.ini”中读取需要编译的文件的名称。在以下目录中查找 *.com 文件：

- [系统用户目录]/proj
- [系统 oem 目录]/proj
- [系统插件目录]/proj
- [系统西门子目录]/proj

此时所包含的登入软键设计会被汇总到一个设计中，即会对单个软键进行比较。如果对于一个软键存在两个或多个设计，则总是将较高的值接收到汇总版本。

可能包含的软键栏或对话框被忽略。如果一个软键包含一个无文件设定的指令，例如 LM(“test”)，因为所需的软键栏或者对话框包含在相同的文件中，则在内部汇总版本中会添加相应的文件名，这样就不需再进行调整。接下来显示得到的汇总设计。

4.4 错误处理（日志）

概述

如果“Run MyScreens”编译配置文件时出现错误，则该错误会保存在文本文件“easyscreen_log.txt”中。系统只记录当前选中的对话框的错误。之前选中的对话框中的错误记录则被删除。

文件包含以下信息：

- 在执行哪一操作时出错。
- 第一个错误字符的行编号和列编号。
- 配置文件中所有的出错行。
- DEBUG 功能生成的记录。

说明

在每个记录前会附加一个带有方括号的当前时间戳。这可为（例如）分析对时间要求苛刻的配置提供帮助。

“easyscreen-log.txt”的保存

文件“easyscreen_log.txt”保存在以下目录中：

[系统用户目录]/log

句法

只有在定义了登入软键并且设计了带开始和结束标记的对话框和定义行后，才开始编译句法。

```
//S(Start)
HS6=("第 1 屏幕")
PRESS(HS6)
  LM("Maske1")
END_PRESS
//END

//M(Maske1)
  DEF Var1=(R)
  DEF VAR2 = (R)
  LOAD
```

```
  VAR1 = VAR2 + 1                ；日志中的错误报告，因为 VAR2 没有数值
...
//END

；正确，比如：
//M(Maske1)
   DEF Var1=(R)
   DEF VAR2 = (R)

   LOAD
      VAR2 = 7
      VAR1 = VAR2 + 1 ;
...
```

4.5 “easyscreen.ini” 的说明

自 SINUMERIK Operate V4.7 起，“easycreen.ini”还扩展了本章节中描述的记录。
“easyscreen.ini”位于目录 [系统西门子目录]/cfg 下。

帮助画面起始位置

“easyscreen.ini”中的记录：

```
[GENERAL]
HlpPicFixPos=true
```

提示：

帮助画面的起始位置会定位在所设置的像素位置，与分辨率无关（默认=true）。

拉伸特性，默认行高和行间距

easyscreen.ini 中的记录：

```
[GENERAL]
SymmetricalAspectRatio=false
DefaultLineHeight=18
DefaultLineSpacing=3
```

说明：

- “SymmetricalAspectRatio”来确定，根据 X 和 Y 方向上确定的屏幕分辨率来调整配置时是否要使用相同的拉伸系数。
  - “false”（默认）：使用宽屏分辨率时，各栏和图片会在 Y 方向上被压缩（基于 640x480 的非对称拉伸）。例如，一个以 640x480 设置的正方形在宽屏面板上会被垂直压缩，而成为一个长方形。
  - “true”：在 X 和 Y 方向上以相同的拉伸系数进行拉伸，此时各栏和图片会以 640x480 为基础并保持其原先设置的比例。例如，一个以 640x480 设置的正方形在宽屏面板上仍然会显示为正方形。
- “DefaultLineHeight” 和 “DefaultLineSpacing” 可在 640x480 的基础上确定默认行高（默认值：18 像素）和行间距（默认值：3 像素）。只有在短文本或输入/输出栏的位置设置中未给定 Y 位置或高度，此项设置才会一直有效。

机床基本画面中的循环

- 在机床运行方式 JOG 下，通过 HS6 生成循环调用进入机床基本画面的 [JOBSHOPINTEGRATION] 区域：

```
[JOBSHOPINTEGRATION]
Integration = true
```

或者通过配置文件中的初始程序段

```
LM("Maske1",,1)
PRESS(VS8)
   GC("MOVE_RIDE") ; 生成程序调用
   EXIT
END_PRESS
```

- 保留 [Integration] 区域的 Operate 菜单：
  如果对话框仅包含垂直软键，则在显示对话框时会保留包含登入软键在内的标准水平软键条。如果对话框包含水平和垂直软键，则对话框的水平软键条会被登入软键替代。

```
[Integration]
OperateMenusEnabled = true
```

与分辨率相关的屏幕位置：窗体面板

“easyscreen.ini”中的记录：

```
[640x480]
MyPanel = x:=0, y:=220, width:=340, height:=174
[800x480]
MyPanel = x:=0, y:=220, width:=420, height:=174
...
```

说明：

屏幕位置可通过以下方式定义：

- 屏幕位置可以“基于 640x480 的像素”给定。
  示例：
  //M(MyMask/"MyCaption"/"\\myhelp.png"/**0,219,335,174**)
- 屏幕位置可从 Operate 默认屏幕布局（Operate Standard-Screen-Layout）中，例如来自“机床”区域的屏幕布局“slmastandardscreenlayout.SlMaStandardScreenLayout”中的“FormPanel4”（“辅助功能”）
  //M(MyMask/"MyCaption"/"\
  \myhelp.png"/**"slmastandardscreenlayout.SlMaStandardScreenLayout.FormPanel4"**)
  连接到与分辨率相关的位置。
  这样可更为方便地使 Operate 的默认窗体交叉淡入淡出，或者可将 Easyscreen 屏幕精确地放置到位。
  示例：
  //
  M(Mask/"Mask"//**"slstandardscreenlayout.SlStandardScreenLayout.LowerForm"**)
  但也可以使用其他任意的屏幕布局。
- 屏幕位置可以连接到“easyscreen.ini”文件中自行定义的、与分辨率相关的窗体面板的定义。
  示例：
  //M(MyMask/"MyCaption"/"\\myhelp.png"/**"MyPanel"**)

4.6 针对改用 “Run MyScreens” 人员的说明

说明

使用 NCU 中的 HMI Operate 时要注意 CF 卡上的所有文件名都是以小写字母保存的（如：com、png、txt）。

说明

保存配置文件和语言文件时请注意，应将您所使用的编辑器中的编码设为 UTF 8。

图形文件

始终以 PNG 格式“*.png”保存图形文件。

数据（例如：用于用于 OEM 自定义的数据）必须保存在

[系统 oem 目录]ico/[分辨率]

目录中。

更多信息参见章节 “使用图/图形 (页 71)”。

调整配置文件

根据以下步骤检查配置文件：

- 比较登入软键和当前允许的软键，必要时进行调整。
- 根据上述“图形文件”重命名相连的图形文件。

数据（例如：用于 OEM 自定义的数据）保存在以下目录中：

[系统 oem 目录]/proj

[系统用户目录]/proj A

[系统插件目录]/proj

调整帮助文件

所有帮助文件必须以 UTF 8 格式保存。检查已有文件并使用合适的编辑器重新保存。

HTML 文件保存在以下目录中（例如：德语）：

[系统 oem 目录]/hlp/deu

[系统用户目录]/hlp/deu

[系统插件目录]/hlp/deu

其他语言的目录必须根据相应的语种缩写创建。

检查 “Run MyScreens” 授权

检查所添加的对话框数目是否超出了最大 5 个对话框的基本范围。

扩展对话框数量，需要下列软件选件：

- SINUMERIK 828D/840D sl, SINUMERIK Integrate Run MyScreens (6FC5800-0AP64-0YB0)
- SINUMERIK 840D sl, SINUMERIK Integrate Run MyScreens + Run MyHMI (6FC5800-0AP65-0YB0)
- SINUMERIK 840D sl, SINUMERIK Integrate Run MyHMI / 3GL (6FC5800-0AP60-0YB0)
- SINUMERIK 840D sl, SINUMERIK Integrate Run MyHMI / WinCC (6FC5800-0AP61-0YB0)

4.7 扩展配置句法

自 SINUMERIK Operate V4.7 起，提供了用于屏幕定义、变量定义、软键定义以及表格列定义的简化句法。该句法改善了可读性并缩短了等待时间。特性和属性可以任意顺序给定，空白记录被取消。相比于以前的句法，特性和属性列表不再使用圆括号 "**(**" 和 "**)**"，而是使用圆弧括号 "**{**" 和 "**}**"。

特性和属性按以下方式给定：
{<名称> = <值>, <名称> = <值>, ...}

以前的句法仍然兼容。

屏幕定义的扩展句法

//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**=<>系统变量或用户变量] [,**HLP_X**=<辅助画面 X 位置>] [,**HLP_Y**=<辅助画面 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, HD="My Mask"}
```

变量定义的扩展句法

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**=<已链接转换变量的名称>]}

示例：

```
DEF MyVar5={TYP="R2", ST="MyVar5", VAL=123.4567, OHLP="myhelp.html", MIN=100.1,
MAX=200.9}
DEF MyVar2={TYP="I", TGL="*1,2,3", VAL=1}
DEF MyVar3={TYP="R2", TGL="*0=""Aus"", 1=$80000", VAL=1}
DEF MyVar4={TYP="R2", TGL="*MyArray",VAL=1}
DEF MyVar1={TYP="R2", TGL="%grid99", X = 0, W=300, H=200}
DEF MyVar6={TYP="R2", TGL="+$80000", VAR="$R[10]", ST="Textoffset"}
```

软键定义的扩展句法

SK = {[**ST**=<名称>] [,**AC**=<访问级>] [,**SE**=<状态>]}

示例：

```
HS1={ST="""MySk""", AC=6, SE=1}
HS3={ST="SOFTKEY_CANCEL"}
HS5={ST="[$81251,""\\sk_ok.png""]"}
HS8={ST="[""Test"",""\\sk_ok.png""]"}
```

表格列定义的扩展句法

{[**TYP**=<类型>] [,**MIN**=<最小值>] [,**MAX**=<最大值>] [,**LT**=<长文本>] [,**ST**=<短文本>] [,**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>] [,**HLP**=<辅助画面>] [,**VAR**=<系统变量或用户变量>] >] [,**W**=<列宽度>] [,**OF1**=<偏移 1>] [,**OF2**=<偏移 2>] [,**OF3**=<偏移 3>]}

示例：

```
DEF MyGridVar={TYP="R", TGL="%MyGrid1", X=10, W=550, H=100}
//G(MyGrid1/0/5)
   {TYP="I", ST="Index", WR=1, VAR="1", W=80, OF1=1}
   {TYP="S", LT="LongText2", ST="Text", WR=1, VAR="$80000", AL=2, W=330, OF1=1}
   {TYP="R3", LT="LongText1", ST="R9,R11,R13,R15", WR=2, VAR="$R[1]", W=110, OF1=2}
//END
```

4.8 SmartOperation 及 MutliTouch 操作

以下设置可专门用于调整 SmartOperation 及 MutliTouch 操作：

- 自动将栏高及栏宽调节至最小可操作的栏尺寸及行间距（屏幕属性 MA）。
- 提高分辨率时精确按像素优化拉伸各栏（屏幕属性 PA）。
- 与字体成比例设置栏高及行间距（屏幕属性 FA）。
- 实现各栏自由滚动，以免虚拟键盘覆盖输入栏（屏幕属性 KM）。

详细信息，请参见定义对话框属性 (页 58)一章的“编程”一节。

对话框

5

5.1 对话框的结构和组成单元

5.1.1 定义对话框

定义

对话框是操作界面上的一个组成部分，操作界面包含标题行、对话框单元和/或图形、显示消息的输出行以及 8 个水平软键和 8 个垂直软键。

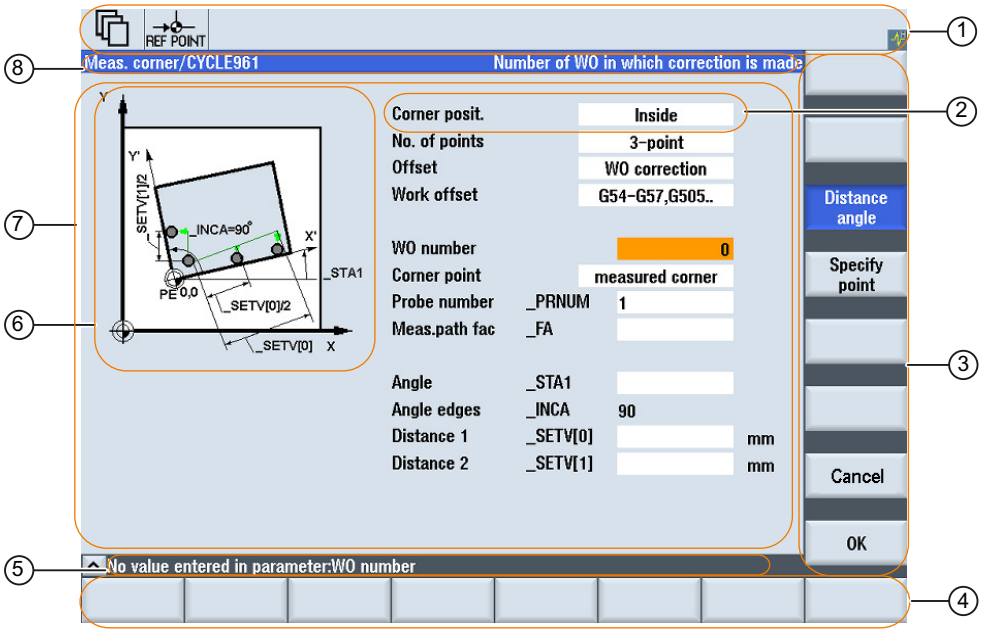
对话框单元包含：

- 变量
  - 极限值/转换栏
  - 变量预设值
- 帮助图形
- 文本
- 属性
- 系统或者用户变量
- 短文本的位置
- 输入/输出栏的位置
- 颜色

对话框的属性：

- 标题
- 图形
- 尺寸
- 系统或者用户变量

- 图形的位置
- 属性



① 机床状态显示（“标题”）
② 对话框单元
③ 8 个垂直软键
④ 8 个水平软键
⑤ 诊断信息的输出
⑥ 图形
⑦ 对话框
⑧ 对话框的标题栏，包含标题和长文本

图 5-1 对话框结构

概述

基本上对话框说明（说明块）的结构如下：

说明块	注释	参考章节
//M...	；对话框的开始标记	
DEF Var1=... ...	；变量	变量 (页 93)

说明块	注释	参考章节
HS1=(...) ...	; 软键	定义软键栏 (页 72)
PRESS(HS1) LM... END_PRESS	; 方法的开始标记 ; 动作 ; 方法的结束标记	方法 (页 145)
//END	; 对话框的结束标记	

在对话框的说明块中，首先定义在对话框的对话框单元中显示的不同变量，然后定义水平和垂直软键。然后在方法中设计不同的动作。

说明

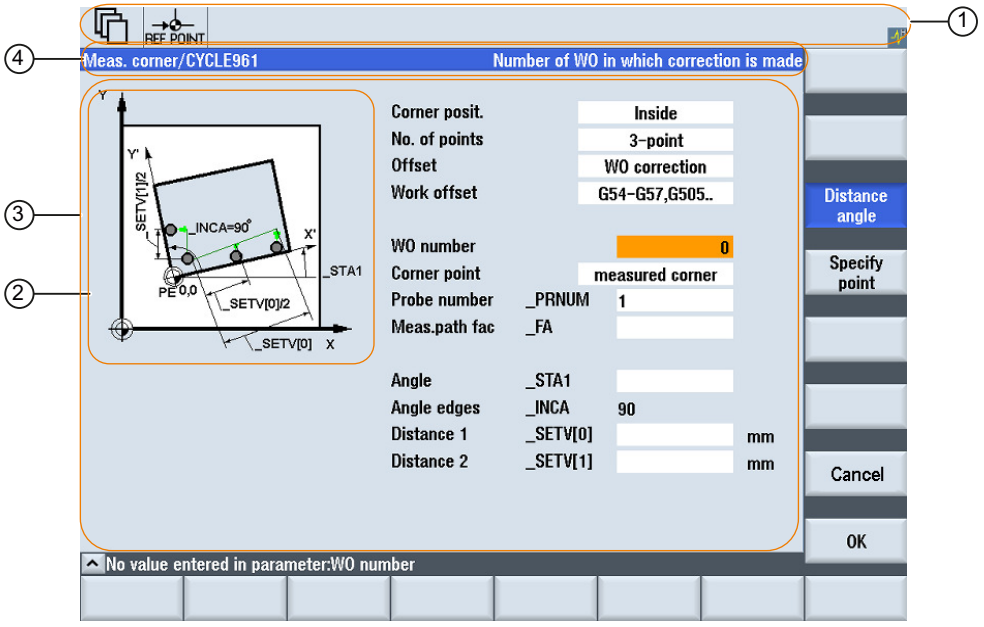
在“机床”操作区切换对话框时需要注意以下顺序：

在后续屏幕的尺寸小于先前屏幕的尺寸或后续屏幕的位置与先前屏幕的位置不同的前提条件下，当返回第一个屏幕之前关闭了后续屏幕并重新加载了第一个屏幕时，对话框切换功能才生效。

5.1.2 定义对话框属性

说明

使用对话框的开始标记可以同时定义对话框的属性。



① 机床状态显示（“标题”）

② 图形

③ 对话框

④ 对话框的标题栏，包含标题和长文本

图 5-2 对话框属性

编程

句法：	**//M**(名称/[标题]/[图形]/[尺寸]/[系统或用户变量]/[图形位置]/[属性]/屏幕专用语言文件列表) 另见章节 扩展配置句法 (页 51)。
说明：	定义对话框

参数：	名称			对话框的名称
	标题			对话框的文本标题或者从和语言相关的文本文件中调用文本（例如：$85011）
	图形			图形文件，路径在双引号内
	尺寸			对话框的位置和大小，单位像素（和左/右边缘的间距、宽度、高度），以屏幕的左上角为基准。数据值用逗号相隔。
	系统或者用户变量			指定当前光标位置的系统和用户变量。可以通过系统或者用户变量将光标位置传送给 NC 或 PLC。第一个变量为索引 1。顺序按照变量的设计顺序。
	图形的位置			图形的位置，单位像素（和左/上边缘的间距），以对话框的左上角为基准。数据值用逗号相隔。
	属性			给定的属性用逗号相隔。 允许的属性有：
		CM		Column Mode:列对齐
			CM0	预设置：每行单独分列。
			CM1	以包含最多列的行为标准分列。
		CB		CHANGE 块:打开对话框时的属性：定义变量时规定的 cb 属性，优先于定义对话框时的总定义 另见 AUTOHOTSPOT。
			CB0	预设置：在打开对话框时处理所有 CHANGE 部分。
			CB1	只有在附属的值改变后才处理 CHANGE 部分。
		XG		连接 X3D 动画作为辅助画面（只用在工步链编程中）。 示例： `//M(Meas/$85605/"myx3dhelpfile.hmi,,Z_Animation,,G17"///30,10/XG1)`
			XG0	缺省值 = 0
			XG1	屏幕属性 XG 必须已在屏幕定义中设为 1，运行期间无法进行更改。 说明： 屏幕属性 HLP 中的数据也应进行相应的设置。
		AL		使用屏幕属性 AL 可以控制屏幕标题的对齐。 示例： 窗口标题“Set password”居中

			//M(MY_PWD_SET/”Set password”//"EasyPwdModalLayout"//AL2)
		AL0	左对齐，缺省设置
		AL1	右对齐
		AL2	居中
	KM		当 MultiTouch 上的虚拟键盘显示时，屏幕上的自由滚动条（SlGfwScrollArea）可能会受以下因素影响： • 屏幕属性为“KM”的屏幕定义行中（KeyboardMode） 示例： //M(MyMTMask/"MultiTouch Mask"////KM1) • 在 “easyscreen.ini” 中作为所有 Run MyScreens 屏幕的全局设置 示例： [GENERAL] DefaultVirtualKeyboardMode=1 说明： 如果在屏幕设置中精确设置了屏幕属性 KM，则它的优先级高于 “easyscreen.ini” 中的全局设置。 （另见章节 SmartOperation 及 MutliTouch 操作 (页 53)）
		KM1	自由滚动条激活（缺省）
		KM0	自由滚动条禁用
	PG		对齐并预览带 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)
		PG0	(缺省)
		PG1	对齐并预览带 PNG 图片的窗口标题 路径说明（最左），屏幕名称（右） 在此模式下无法显示长文本。您可另行选择使用提示框。
	MA		使用 MultiTouch 操作(MutliTouch Adjustment)时自动匹配栏高 MultiTouch 操作面板的匹配设置包括将各栏（进度条、GIF 动图、CustomWidget 除外）扩大至最小可操作尺寸（宽、高）以及行间距。 MultiTouch 匹配可 • 在 “easyscreen.ini” 中作为所有 Run MyScreens 屏幕的全局设置，例如

			[GENERAL] DefaultMultiTouchAdjustmentLevel=1 • 或者可针对具体屏幕在屏幕定义行中使用屏幕属性“MA”将“easyscreen.ini”中的全局设置覆盖掉。例如 //M(MyMTMask/"MultiTouch Mask"/////MA1) 如果在屏幕设置中明确设置了屏幕属性 MA，则它的优先级高于“easyscreen.ini” 中的全局设置。 （另见章节 SmartOperation 及 MutliTouch 操作 (页 53)）
		MA0	未进行任何匹配
		MA1	只匹配未设置上间距和/或栏高/栏宽的栏 – 也包括使用缺省位置并且由 Run MyScreens 来计算上间距和/或栏高/栏宽的栏。(缺省)
		MA2	匹配所有栏，不考虑所设置的栏大小。
	PA		提高分辨率时精确按像素优化扩展各栏（Pixel Fine Adjustment）。 此时会首先基于 640x480 计算所有的栏位置，然后根据相应的水平或垂直拉伸系数进行缩放。但是该属性也有缺点，如若“拉伸系数不合适”则会导致取整误差。比如栏高或行与行之间的间距可能会“跳过”一个像素。为避免出现这种情况，而提供了“Pixel Fine Adjustment”模式，在此模式下栏位置会直接根据当前分辨率来确定。 //M(MyMask/"My Mask"/////PA1) 该设置也可在 “easyscreen.ini” 中作为所有 Run MyScreens 屏幕的全局设置，例如 [GENERAL] DefaultPixelFineAdjustment=1 如果在屏幕设置中明确设置了屏幕属性 PA，那么在“easyscreen.ini” 中则具有比全局设置更高的优先级、 如果屏幕通过 Font Adjustment = FA1 来显示，那么对于该屏幕 Pixel Fine Adjustment 模式也可自动生效。 （另见章节 SmartOperation 及 MutliTouch 操作 (页 53)）
		PA0	拉伸属性保持当前设置，即先基于 640x480 计算位置，然后拉伸。 （由于兼容性的原因：缺省）
		PA1	提高分辨率时精确按像素优化拉伸各栏
	FA		与字体成比例设置栏高及行间距（Font Adjustment） 该设置也可在 “easyscreen.ini” 中作为所有 Run MyScreens 屏幕的全局设置，例如

		[GENERAL] DefaultFontAdjustment=1 如果在屏幕设置中明确设置了屏幕属性 FA，则它的优先级高于 “easyscreen.ini” 中的全局设置。 DefaultLineHeight 和 DefaultLineSpacing 的设置在 Font Adjustment 模式生效时无用。 （另见章节 SmartOperation 及 MutliTouch 操作 (页 53)）
	FA0	栏高和行间距按当前方式拉伸（由于兼容性的原因：缺省）
	FA1	与字体成比例设置栏高及行间距（自动设置屏幕属性 PA=1）
	FA2	与 FA1 相同，但会额外与字体成比例设置 X 坐标和栏宽。 （自动设置屏幕属性 PA=1） （只在同时使用属性 XG=1 时可能！）
NT		导航类型（Navigation Type）
	NT0	NT0 = 标准模式 导航与集成了 Run MyScreens 屏幕的操作区中的屏幕相似。也就是说，在编辑器的循环屏幕中是按行导航的（参见 NT2），在其他所有“常规”屏幕中是按几何方向导航的（NT1）。
	NT1	几何方向导航（上、下、左、右），直到到达屏幕的边界（常规 Run MyScreens 环境中（例如“Custom”区域）的标准方式）
	NT2	按行导航，即：在行中向右，直到到达行末尾（编辑器循环屏幕中的标准方式）
NR		按着 Enter 键时的导航方向（Return Navigate Direction）
	NR0	NR0 = off（缺省），即：按着 Enter 键时，屏幕中的导航是按照表格顺序进行的（常规 Run MyScreens 环境中的标准方式）
	NR1	按着 Enter 键时，导航与按着向上箭头按键时一样
	NR2	与 NR1 相同，只不过是向下
	NR3	与 NR1 相同，只不过是向左
	NR4	与 NR1 相同，只不过是向右
TA		提示框对齐
	TA0	自动（缺省）
	TA1	左
	TA2	右
	TA3	上
	TA4	下

			TA5	左上
			TA6	左下
			TA7	右上
			TA8	右下

	MC			屏幕背景色(Mask Color)
				示例：
				设置屏幕背景色为蓝色(= 6)
				`//M(MyMask/"MyMask"//////6)`
				或者
				`PRESS(HS1)`
				`  MC=6`
				`END_PRESS`

	屏幕专用语言文件列表			用逗号隔开

存取对话框属性

在方法（例如：PRESS 块）的范围内可以读取和写对话框的以下属性：

- HD = 标题 (Header)
- HLP = 辅助画面
- VAR = 系统或用户变量
- MC = 屏幕背景色
- CM = 列对齐（只读）
- CB = 打开方式（只读）
- XG = 连接 X3d（只读）
- AL = 屏幕标题对齐（只读）

示例

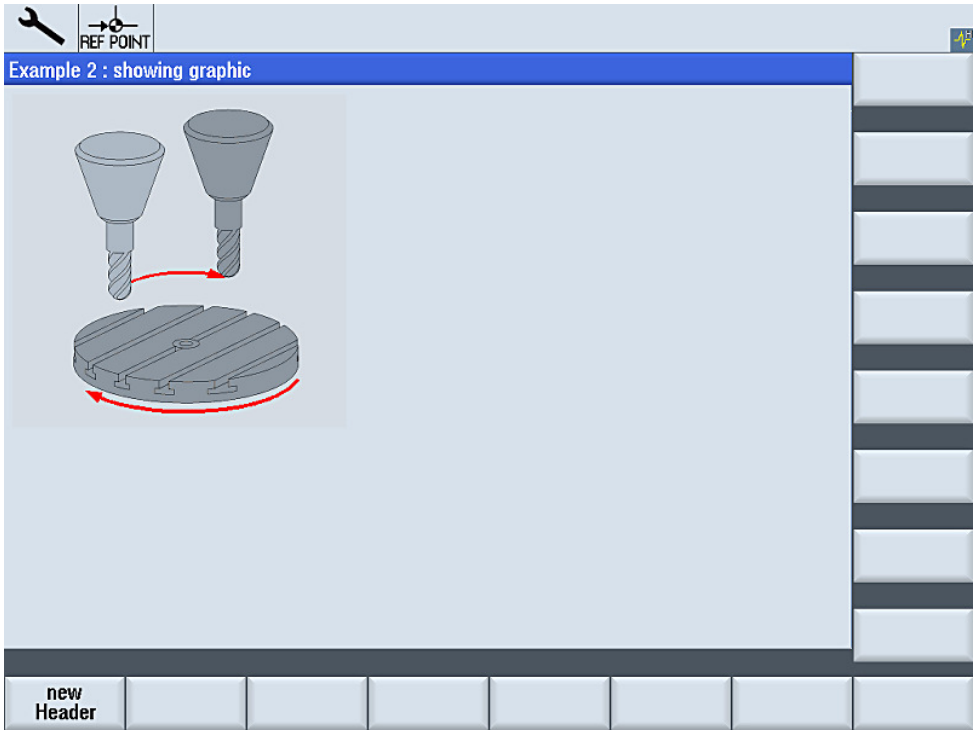


图 5-3 "Example 2: showing graphic"

```
//S(Start)
HS7=("Example", se1, ac7)

PRESS(HS7)
  LM("Mask2")
END_PRESS

//END
//M(Mask2/"Example 2 : showing graphic"/"example.png")
HS1=("new%nHeader")
HS2=("")
HS3=("")
HS4=("")
HS5=("")
HS6=("")
HS7=("")
HS8=("")
VS1=("")
VS2=("")
```

```
VS3=("")
VS4=("")
VS5=("")
VS6=("")
VS7=("")
VS8=("")

PRESS(HS1)
  Hd= "new Header"
END_PRESS
...
//END
```

参见

操作区“自定义”的编程示例 (页 318)

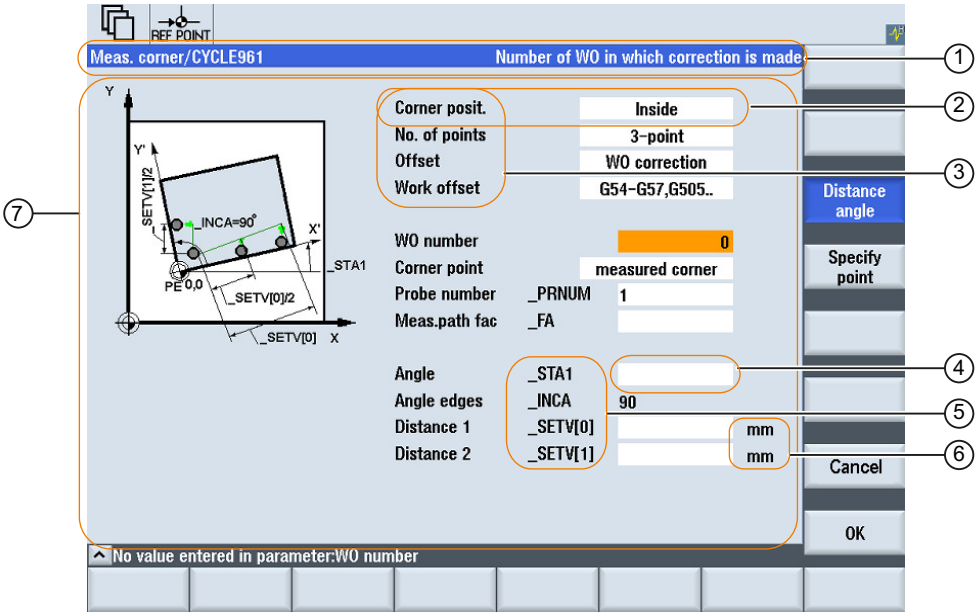
5.1.3 定义对话框单元

对话框单元

对话框单元是变量的可见部分，即短文本、图形文本、输入/输出栏、单位文本和提示框。对话框单元位于对话框主体的行中。每行可以定义一个或者多个对话框单元。

变量属性

所有变量仅在激活的对话框中有效。通过定义变量指定其属性。在方法（例如：PRESS 方法）的范围内可以存取对话框属性值。



① 对话框的标题栏，包含标题和长文本
② 对话框单元
③ 短文本
④ 输入/输出栏
⑤ 图形文本
⑥ 单位文本
⑦ 对话框主体

图 5-4 对话框单元

编程一览

在圆括号中通过逗号隔开各个参数。

DEF [命名符] =	**命名符 = 变量名称**
	变量类型
	/[极限值或转换栏]
	/[预设值]
	/[文本（长文本，短文本|图，图形文本，单位文本）]

	/[属性]
	/[帮助画面]
	/[系统变量或者用户变量]
	/[短文本位置]
	/[输入栏/输出栏位置（左、上、宽度、高度）]
	/[颜色]
	/[在线帮助] (页 81)

另见

变量参数 (页 107)

5.1.4 定义多列对话框

概述

在对话框中，一行可以显示多个变量。 在这种情况下，设计文件中的所有变量都定义在一个定义行内。

```
DEF VAR11 = (S///"Var11"), VAR12 = (I///"Var12")
```

为了可以更清除地显示设计文件中的各个变量，在每个变量定义和其紧随的逗号后可以换行。

关键字“DEF”总是表示新的一行的开始：

```
DEF Tnr1=(I//1/"","T ",""/wr1///,,10/20,,50),
   TOP1=(I///,"Typ="/WR2//"$TC_DP1[1,1]"/80,,30/120,,50),
   TOP2=(R3///,"L1="/WR2//"$TC_DP3[1,1]"/170,,30/210,,70),
   TOP3=(R3///,"L2="/WR2//"$TC_DP4[1,1]"/280,,30/320,,70),
   TOP4=(R3///,"L3="/WR2//"$TC_DP5[1,1]"/390,,30/420,,70)
DEF Tnr2=(I//2/"","T ",""/wr1///,,10/20,,50),
   TOP21=(I///,"Typ="/WR2//"$TC_DP1[2,1]"/80,,30/120,,50),
   TOP22=(R3///,"L1="/WR2//"$TC_DP3[2,1]"/170,,30/210,,70),
```

```
TOP23=(R3///,"L2="/WR2//"$TC_DP4[2,1]"/280,,30/320,,70),
TOP24=(R3///,"L3="/WR2//"$TC_DP5[2,1]"/390,,30/420,,70)
```

说明

在设计多列对话框时注意，列过多可能会导致系统变慢！

5.1.5 口令对话框

使用标准口令对话框

您可在屏幕配置中连接以下预定义的口令对话框：

- 设置口令

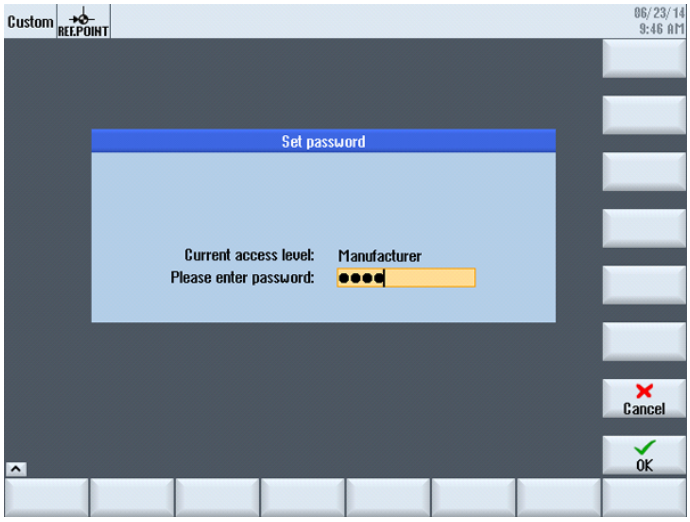


图 5-5 设置口令对话框

- 更改口令

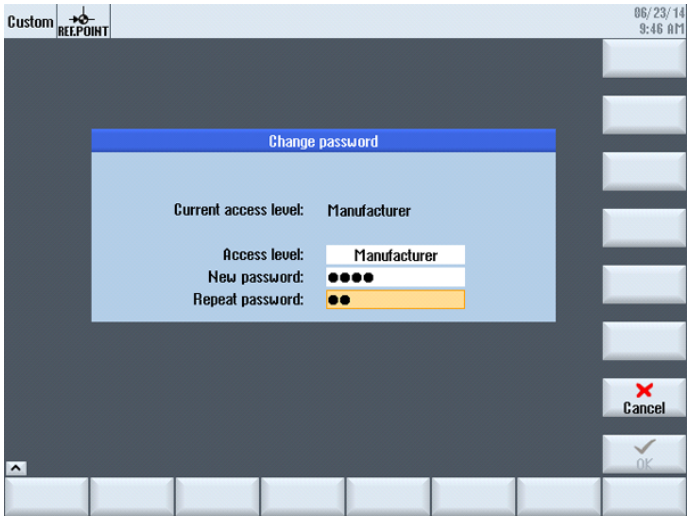


图 5-6 更改口令对话框

- 删除口令

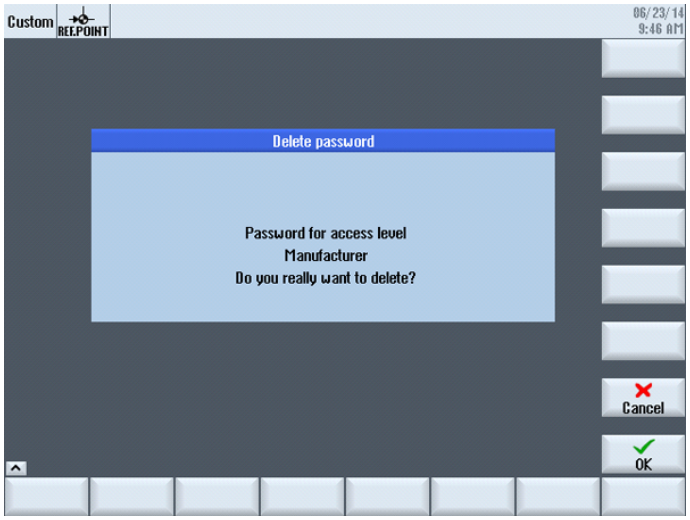


图 5-7　删除口令对话框

说明

使用这些对话框可为您提供有关口令的各项功能。这些对话框与 SINUMERIK Operate 的对话框并不一样。

调用对话框既可以通过功能 Load Softkey (LS)进行，也可以通过 Load Mask (LM)进行：

1. 通过功能 Load Softkey (LS)调用：
   `LS("Passwd","slespasswd.com",1)`
   垂直软键扩展：
   - 软键 1：设置口令
   - 软键 2：更改口令
   - 软键 3：删除口令
2. 或者也可以直接通过调用功能 Load Mask (LM)跳转到这三个屏幕：
   - 设置口令：`LM("PWD_SET", "slespasswd.com",1)`
   - 更改口令：`LM("PWD_CHG", "slespasswd.com",1)`
   - 删除口令：`LM("PWD_CLEAR", "slespasswd.com",1)`

5.1.6 使用图/图形

使用图形

分为：

- 使用图/图形
- 帮助图形，例如：图示说明各个变量并突出显示在图形区中。
- 可以设计其它帮助图形替代可任意定位的短文本或者输入/输出栏。

保存位置

与相连监视器的分辨率相匹配的图片，请先从相应的分辨率目录下按顺序查找：

[系统用户目录]/ico/ico<分辨率>
[系统 oem 目录]/ico/ico<分辨率>
[系统插件目录]/ico/ico<分辨率>

如果未显示或未找到图片，将其复制到分辨率为 640 x 480 像素的目录中：

[系统用户目录]/ico/ico640
[系统 oem 目录]/ico/ico640
[系统插件目录]/ico/ico640

说明

在面板分辨率不同时，图片会按比例定位。

5.2 定义软键栏

定义

所有水平软键和所有垂直软键分别布置在一起称作软键栏。另外针对已有的软键栏，还可以定义其它软键栏，可以部分或者完全覆盖已有的软键栏。

确定软键名称。不必占用所有软键。

HSx x 1 - 8, 水平软键 1 至 8

VSy y 1 - 8, 垂直软键 1 至 8

原则上软键栏描述（描述块）如下构建：

说明块	注释	参考章节
`//S...`	; 软键栏开始标记	
`HSx=...`	; 定义软键	

`PRESS(HSx)`
`  LM...`
`END_PRESS`
	; 方法的开始标记
	; 动作
	; 方法的结束标记
		参见章节 方法 (页 145)

`//END`	; 软键栏结束标记	

说明

定义软键栏也同时分配软键属性。

编程

句法：	*//S(名称)* ... //END	;软键栏开始标记 ;软键栏结束标记
	另见章节 扩展配置句法 (页 51)。	
说明：	定义软键栏	
参数：	名称	软键栏名称
	文本或图形文件名称	
句法：	**SK** = (文本[，访问等级][，状态][，软键图对齐][，以软键图为基准的文本对齐])	

说明：	定义软键		
参数：	SK	软键，例如： HS1 到 HS8, VS1 到 VS8	
	文本	给定文本	
	图文件名称	"\\my_pic.png" 或者通过单独的文本文件 $85199，例如：（和语言相关的）文本文件中的下列文本：85100 0 0 "\\my_pic.png"。 所能显示的软键图片的大小取决于所使用的 OP:	
		OP 08:	640 x 480 mm → 25 x 25 像素
		OP 010:	640 x 480 mm → 25 x 25 像素
		OP 012:	800 x 600 mm → 30 x 30 像素
		OP 015:	1024 x 768 mm → 40 x 40 像素
		OP 019:	1280 x 1024 mm → 72 x 72 像素
	访问等级	ac0 到 ac7（ac7：缺省设置）	
	状态	se1:可见（预设置） se2:不可操作（灰色标签） se3:高亮显示（最后操作的软键）	
	软键图对齐	PA (PictureAlignment)	
		有效值： 0:左 1:右 2:居中 3:上（默认） 4:下	
	以软键图为基准的文本对齐	TP (TextAlignedToPicture)	
		有效值： 0:文本和图形未对齐 1:文本和图形已对齐（默认）	

说明

对于软键标签，用 %n 进行换行。

最多 2 行，每行各 9 个字符。

分配访问等级

操作员只能访问符合其访问等级和各个低于其访问等级的信息（另见章节 AUTOHOTSPOT）。

示例

```
//S(软键栏 1)                                  ;软键栏开始标记
HS1=("新建", ac6, se2)                         ; 定义软键 HS1：软键标签为 "新建"，保护等级为 6，状态为"不可操作"。
HS2=("\\bild1.png")                            ; 指定软键图片
HS3=("退出")
HS4=(["Confirm","\\sk_ok.png"],PA0,TP1)        ; 带文本和图形的软键，文本="Confirm"，图形="sk_ok.png"，软键图对齐：左，文本 和图形对齐

VS1=("子窗口")
VS2=($85011, ac7, se2)                         ; 定义软键 VS2：软键标签来自语言文件，保护等级为 1，状态为"不可操作"
VS3=("取消", ac1, se3)                         ; 定义软键 VS3：软键标签为"取消"，保护等级为 1，状态为"高亮"
VS4=("确认", ac6, se1)                         ; 定义软键 VS4：软键标签为"确认"，保护等级为 6，状态为"可见"
VS5=(SOFTKEY_CANCEL,,se1)                      ; 定义标准软键 VS5：软键标签为"取消"，状态为"可见"
VS6=(SOFTKEY_OK,,se1)                          ; 定义标准软键 VS6：软键标签为"确认"，状态为"可见
VS7=(["\\bild1.png","OEM 文本"],,se1)          ; 定义软键 VS7：指定图片，软键标签为"OEM 文本"，状态为"可见"
VS8=(["\\bild1.png",$83533],,se1)              ; 定义软键 VS8：指定图片，软键标签来自语言文件，状态为"可见"

PRESS(HS1)                                     ; 方法开始标记
  HS1.st="计算"                                ; 指定软键标签
...
END_PRESS                                      ; 方式结束标记

PRESS(RECALL)                                  ; 方法开始标记
  LM("屏幕 21")                                ; 载入对话框
END_PRESS                                      ; 方式结束标记
//END                                          ; 软键栏结束标记
```

5.2.1 运行时改变软键属性

说明

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：

编程

句法：	SK.st = "文本" SK.ac = 访问等级 SK.se = 状态 SK.pa = “软键图对齐” SK.tp = “以软键图为基准的文本对齐”		；软键标签 ；软键访问等级 ;软键状态 ；带图形的软键 ；带图形和标签的软键
说明：	分配属性		
参数：	文本		引号中的标签文本
	访问等级		值范围： 0 ... 7
	状态	1: 2: 3:	可见并可操作 不可操作（灰色标签） 高亮显示（最后操作的软键）
	软键图对齐	0: 1: 2: 3: 4:	左 右 居中 上（默认） 下
	以软键图为基准的文本对齐	0: 1:	文本和图形未对齐 文本和图形已对齐（默认）

示例

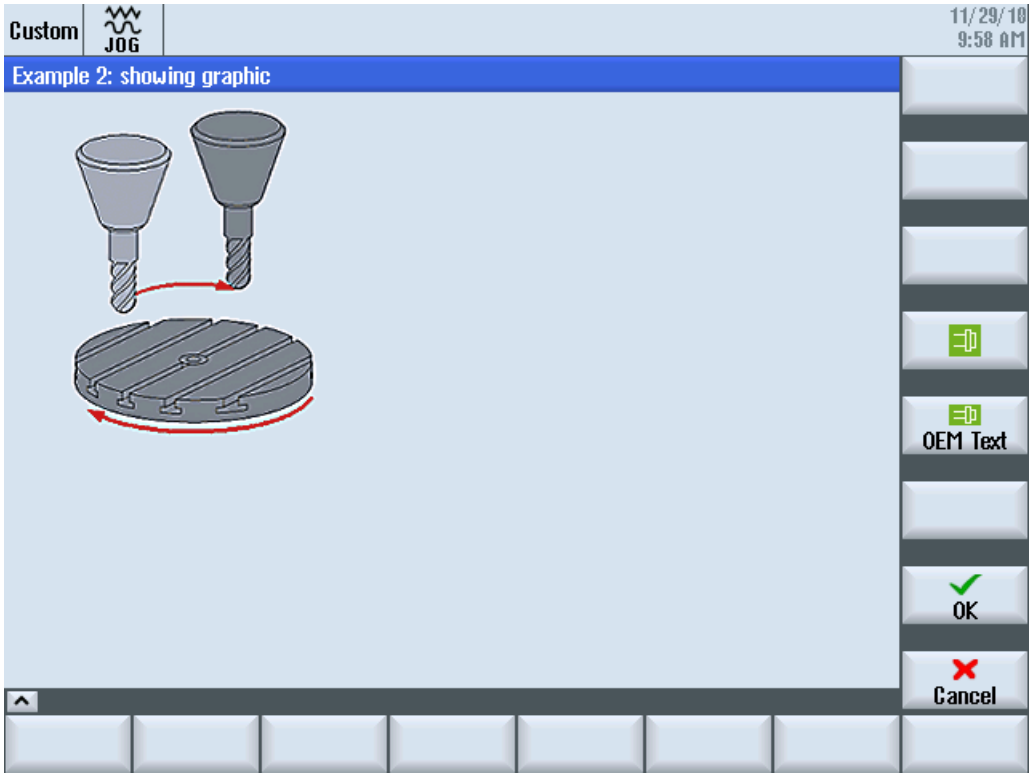


图 5-8　示例 3：图形和软键

```
//S(Start)
HS7=("Example", ac7, se1)

PRESS(HS7)
  LM("Maske3")
END_PRESS

//END

//M(Maske3/"Example 2: showing graphic"/"example.png")
HS1=("")
HS2=("")
HS3=("")
HS4=("")
HS5=("")
HS6=("")
HS7=("")
HS8=("")
```

```
VS1=("")
VS2=("")
VS3=("")
VS4=("\\sp_ok.png",,SE1)
VS5=(["\\sp_ok_small.png","OEM Text"],,SE1)
VS6=("")
VS7=(SOFTKEY_OK,,SE1)
VS8=(SOFTKEY_CANCEL,,SE1)
PRESS(VS4)
  EXIT
END_PRESS
PRESS(VS5)
  EXIT
END_PRESS
PRESS(VS7)
  EXIT
END_PRESS
PRESS(VS8)
  EXIT
END_PRESS
//END
```

5.2.2 和语言相关的文本

概述

使用和语言相关的文本：

- 软键标签
- 标题
- 辅助文本
- 其它任意文本

对话框中与语言相关的文本保存在文本文件中。

文本文件位于以下目录中：

- [系统用户目录]/lng
- [系统 oem 目录]/lng
- [系统插件目录]//lng

说明

文本文件必须如同项目文件一样进行保存。
例如：
[系统用户目录]/lng/[文本文件]
[系统用户目录]/proj/[配置文件 i]

alsc.txt　　用于西门子标准循环、和语言相关的文本
almc.txt　　用于西门子测量循环、和语言相关的文本

程序运行中所使用的文本文件在“easyscreen.ini”文件中给定：

```
[LANGUAGEFILES]
LngFile03 = user.txt      ;->user<_xxx>.txt (例如：user_eng.txt)
```

文件“user.txt”在这里是作为文本文件的示例。原则上可自由选择名称。根据文件中文本的不同语种，还必须按照下面的句法附加上各语言的缩写。在名称后加上一个下划线以及相应的语种缩写，例如“user_eng.txt”。

说明

由于 LngFile01 和 LngFile02 已经指定给标准“easyscreen.ini”，因而不可用于用户文本。

另见

AUTOHOTSPOT

屏幕专用语言文件

只能在屏幕上使用确定的语言文件，以避免语言文件中所用的编号带重叠。

因此，在屏幕定义行中使用逗号隔开所用的语言文件。最后位置上的文件具有最高的优先级（与“easyscreen.ini”中的逻辑类似）。

所有在屏幕中使用的语言相关的文本提前都在语言文件中查找过了。如果没有找到文本，则接着在“easyscreen.ini”配置的文本文件中查找。

如果屏幕中没有定义语言文件，则只需在“easyscreen.ini”指定的语言文件中查找。

说明

如果当前所选的语言中没有语言文件，则使用英文语言文件（缺省）。

示例：

```
//M(MyMask/$85597///// //"mytexts.txt, general.txt, mask.txt")
DEF ….
```

该操作也能用于登入软键：

```
//S(Start, "mytexts.txt, general.txt, mask.txt")
VS1=($85597)
PRESS(VS1)
   LM("MyMask","masks.com")
END_PRESS
```

文本文件的格式

文本文件必须保存为 UTF-8 编码格式。

说明

保存配置文件和语言文件时请注意，应将您所使用的编辑器中的编码设为 UTF 8。

文本输入形式

句法：	8xxxx 0 0 “文本”		
说明：	文件中文本号和文本的排列		
参数：	xxxx	85000 到 89999 900.000 至 999.999	预留于用户的文本识别号范围。号码必须是唯一的。
	“文本”		显示在对话框中的文本
	%n		文本中用于分行的控制符

两个通过空格分开的参数 2 和 3 是用于报警文本输出的控制符。鉴于与报警文本的格式一致性，在任何情况下控制符都必须置零。

和语言相关的文本示例：

```
85000 0 0 “退回平面”
85001 0 0 “钻削深度”
85002 0 0 “螺距”
85003 0 0 “凹槽半径”
```

5.3 设计在线帮助

5.3.1 概览

除已有的各种在线帮助外，还可创建制造商专用在线帮助，并将其集成至 SINUMERIK Operate 中。

此种在线帮助使用 HTML 格式，即：由彼此链接的 HTML 文档组成。 可在特定的窗口中通过内容或关键词目录调用所搜索的主题。 与文档浏览器（比如 Windows 浏览器）相似，在窗口的左边部分显示选择概览，点击所需的主题后，会在右边窗口部分显示相应的说明。

不能相关上下文选择在线帮助页。

步骤

1. 创建 HTML 文件
2. 创建帮助手册
3. 将在线帮助集成到 SINUMERIK Operate 中
4. 保存帮助文件

其他使用情况

可以建立以下 OEM 专有扩展功能的在线帮助，以补充 SINUMERIK Operate 的在线帮助系统：

- 机床制造商**循环与/或 M 功能**的在线帮助，可扩展 SINUMERIK 控制系统的编程方式。 该在线帮助通过 SINUMERIK Operate 在线帮助“编程”来调用。
- 机床制造商 OEM 专用**变量**的在线帮助。 此在线帮助从 SINUMERIK Operate 的变量视图调用。

编程在线帮助

对于在线帮助的其它设计可使用“SINUMERIK HMI 编程包 sl”。 可使用此编程包用程序语言 C++ 为 NCU 7x0 上的 SINUMERIK Operate 开发标准语言应用程序。

说明

“SINUMERIK HMI 编程语言包 sl”是一个需要单独订购的软件选件。 对应的说明文档随编程语言包一起发货。

5.3.2 生成 HTML 文件

创建 HTML 格式的帮助文件。您可以将所有信息放在一份 HTML 文件中或者放在几份 HTML 文件中。

可以自己命名文件，但需遵守以下事项：

- 必须为 HTML 文件中的参照关系规定相关的路径。只有这样能保证引用不仅在开发计算机上有效，而且在载入系统后同样有效。
- 若需要在 HTML 文件中的特定点上通过链接进行跳转，必须定义所谓的锚点。
  HTML 锚点示例：
  ```
  <a name="myAnchor">This is an anchor</a>
  ```
- HTML 文档的内容必须保存为 UTF-8 编码格式。只有这样才能确保 HTML 文档在所有 SINUMERIK Operate 所支持的语言中都可正确显示。
- 可支持如下 HTML 函数集的子集：

HTML 标签

标签	说明	注释
a	锚点或链接	所支持的属性：href 和名称
address	地址	
big	大字体	
blockquote	首行缩进段落	
body	正文	所支持的属性：bgcolor (#RRGGBB)
br	换行	
center	居中段落	
cite	内嵌引用	与标签 i 的效果相同
code	代码	与标签 tt 的效果相同
dd	定义数据	
dfn	定义	与标签 i 的效果相同
div	文档分隔	支持标准程序段属性
dl	定义列表	支持标准程序段属性
dt	定义术语	支持标准程序段属性
em	强调	与标签 i 的效果相同
font	字体大小，语系，颜色	所支持的属性：大小，字符体和颜色 (#RRGGBB)

标签	说明	注释
h1	1 级标题	支持标准程序段属性
h2	2 级标题	支持标准程序段属性
h3	3 级标题	支持标准程序段属性
h4	4 级标题	支持标准程序段属性
h5	5 级标题	支持标准程序段属性
h6	6 级标题	支持标准程序段属性
head	文档页眉	
hr	水平线	所支持的属性：宽度（可以是绝对数值或者相对数值）
html	HTML 文档	
i	斜体	
img	图像	所支持的属性：src，宽度，高度
kbd	用户输入文本	
meta	元信息	
li	列表条目	
nobr	不易破碎文本	
ol	有序列表	支持列表的标准属性
p	段落	支持标准程序段属性（默认设置：左对齐）
pre	预先设好格式的文本	
s	删除线	
samp	示例代码	与标签 tt 的效果相同
small	小字体	
span	分组元素	
strong	加粗	连续文本加粗显示，替代标签 b
sub	下标	
sup	上标	
table	表格	所支持的属性：边框，bgcolor (#RRGGBB)，单元格间距，单元格填充，宽度（绝对或者相对），高度
tbody	表格正文	无效
td	表格数据单元	支持表格单元格的标准属性
tfoot	表格页脚	无效
th	表格页眉单元	支持表格单元格的标准属性

标签	说明	注释
thead	表格页眉	打印表格时使用，可扩展至多页
title	文档标题	
tr	表格行	所支持的属性：bgcolor (#RRGGBB)
tt	打字字体	
u	下划线	
ul	无序列表	支持列表的标准属性
var	变量	与标签 tt 的效果相同

程序段属性

标签 div、dl、dt、h1、h2、h3、h4、h5、h6 和 p 支持如下属性：

- 对齐（左对齐，右对齐，居中，垂直居中）
- dir (ltr, rtl)

列表的标准属性

标签 ol 和 ul 支持如下属性：

- 类型 （1, a, A, 方形，圆形，环形）

表格的标准属性

标签 td 和 th 支持如下属性：

- 宽度（绝对，相对，无数值）
- bgcolor (#RRGGBB)
- 列间距
- 行间距
- 对齐（左对齐，右对齐，居中，垂直居中）
- 对齐（上对齐，居中，下对齐）

CSS 属性

如下表格中为所支持的 CSS 函数集：

属性	值	说明
background-color	<color>	元素的背景颜色
background-image	<uri>	元素的背景图像
color	<color>	文本的颜色
text-indent	<length>px	段落首行缩进长度，单位：像素
white-space	normal | pre | nowrap | pre-wrap	确定如何处理 HTML 文档中的空格
margin-top	<length>px	边框上边距的宽度，单位：像素
margin-bottom	<length>px	边框下边距的宽度，单位：像素
margin-left	<length>px	边框左边距的宽度，单位：像素
margin-right	<length>px	边框右边距的宽度，单位：像素
vertical-align	baseline | sub | super | middle | top | bottom	文本垂直对齐，在表格中只支持数值“居中”、“上对齐”、“下对齐”。
border-color	<color>	文本表格的边框颜色
border-style	none | dotted | dashed | dot-dash | dot-dot-dash | solid | double | groove | ridge | inset | outset	文本表格的边框样式
background	[<'background-color'> || <'background-image'>]	背景属性的缩写
page-break-before	[auto | always]	在段落/表格前分页
page-break-after	[auto | always]	在段落/表格后分页
background-image	<uri>	元素的背景图像

所支持的 CSS 选择器

除所谓的虚拟选择器等级（如 :first-child， :visited 和 :hover）外，支持所有的 CSS 2.1 选择器等级。

5.3.3 创建帮助手册

帮助手册是一个 XML 文件，可在其中决定在线帮助的结构。 可在此文件中定义：

- HTML 文档
- 内容和关键词目录

帮助手册的句法

标签	数量	含义	
HMI_SL_HELP	1	XML 文档的根元素	
I-BOOK I I I I I I I I I I	+	帮助手册命名。手册名称可自由选择，只要不是手册中预定义的名称即可，例如：sinumerik_alarm_plc_pmc。 本例中帮助手册的名称为“hmi_myhelp”。 属性：	
		ref	命名 HTML 文档，其将作为帮助手册的主页显示。
		titel	显示在内容目录中的帮助手册的标题。
		helpdir	包含帮助手册的在线帮助的目录。
I-ENTRY II II II II II II II II	*	在线帮助的章节 属性：	
		ref	命名 HTML 文档，其将作为章节的主页显示。
		titel	显示在内容目录中的章节标题。
II-INDEX_ENTRY II II II II II II II II	*	待显示的关键词 属性：	
		ref	命名 HTML 文档，将根据关键词条目跳至此文档。
		titel	在关键词目录中显示的关键词的标题。

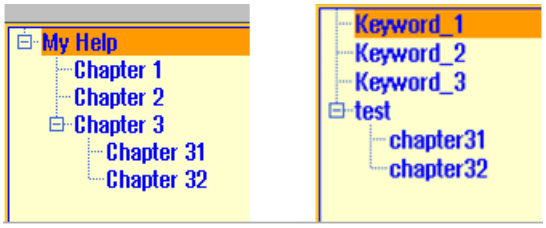
“数量”列：
* 表示 0 或者更多
+ 表示 1 或者更多

帮助手册示例

如下示例将描述名称为“My Help”的帮助手册的结构。 此外，其还是内容目录和关键词目录的基础。

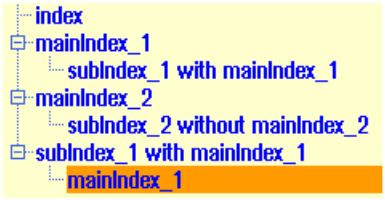
```
<?xml version="1.0" encoding="utf-8"?>
<HMI_SL_HELP language="en-US">
  <BOOK ref="index.html" title="My Help" helpdir="hmi_myhelp">
    <ENTRY ref="chapter_1.html" title="Chapter 1">
      <INDEX_ENTRY ref="chapter_1html#Keyword_1" title="Keyword_1"/>
      <INDEX_ENTRY ref="chapter_1.html#Keyword_2" title="Keyword_2"/>
    </ENTRY>
    <ENTRY ref="chapter_2.html" title="Chapter 2">
      <INDEX_ENTRY ref="chapter_2.html#Keyword_3" title="Keyword_3"/>
    </ENTRY>
    <ENTRY ref="chapter_3.html" title="Chapter 3">
      <ENTRY ref="chapter_31.html" title="Chapter 31">
        INDEX_ENTRY ref="chapter_31.html#test" title="test;chapter31"/>
      </ENTRY>
      <ENTRY ref="chapter_32.html" title="Chapter 32">
        INDEX_ENTRY ref="chapter_32.html#test" title="test;chapter32"/>
      </ENTRY>
    </ENTRY>
  </BOOK>
</HMI_SL_HELP>
```

整个手册由三章组成，其中第三章包括两个子章节。 在各章中均定义了不同的关键词。



可采用如下三种方法编排关键词目录的格式：

1. 单个条目：
   <INDEX_ENTRY ...title="index"/>
2. 两个二级条目，其中每个标题拥有一个主条目和子条目。 用逗号将不同的条目隔开：
   <INDEX_ENTRY ...title="mainIndex_1,subIndex_1 with mainIndex_1"/>
3. 二级条目，其中为主条目的第一个标题和子条目的第二个标题。 使用分号隔开各条目。
   <INDEX_ENTRY ...title="mainIndex_2;subIndex_2 without mainIndex_1"/>



5.3.4 将在线帮助集成至 SINUMERIK Operate 中

若需将创建的帮助手册集成至 SINUMERIK Operate 的在线帮助系统中，需使用文件“slhlp.xml”。

“slhlp.xml”文件的格式描述

标签	数量	含义	
CONFIGURATION	1	XML 文档的根元素。 用于标识配置文件。	
I-OnlineHelpFiles	1	引入在线帮助手册的段落。	
II-<help_book>	*	引入帮助手册的段落。	
III-EntriesFile III III III III III	1	含内容条目和关键词条目的帮助手册的文件名。 属性：	
		值	XML 文件的名称
		类型	数值的数据类型 (QString)

标签	数量	含义	
III-Technology III III III III III III III III III IIII	0, 1	规定帮助手册所适用的工艺。 “All”（全部）表示适用于全部工艺。 若一个帮助手册适用于多个工艺，则工艺之间用逗号隔开。 可能的数值： 全部，通用，铣床，车床，磨床，冲床 属性：	
		值	工艺说明
		类型	数值的数据类型 (QString)
III -DisableSearch III III III III III	0, 1	关闭帮助手册的关键词搜索。 属性：	
		值	true, false
		类型	数值的数据类型 (bool)
III- DisableFullTextSearch III III III III	0, 1	关闭帮助手册的全部文本搜索功能。 属性：	
		值	true, false
		类型	数值的数据类型 (bool)
III-DisableIndex III III III III	0, 1	关闭帮助手册的关键词目录。 属性：	
		值	true, false
		类型	数值的数据类型 (bool)
III-DisableContent III III III III	0, 1	关闭帮助手册的内容目录。 属性：	
		值	true, false
		类型	数值的数据类型 (bool)

标签	数量	含义	
III-DefaultLanguage III III III III III	0, 1	若帮助手册中有当前国家的语言，则会显示此语言的缩写。 属性：	
		值	chs, deu, eng, esp, fra, ita, ...
		类型	数值的数据类型 (QString)

“数量”列：
* 表示 0 或者更多

文件“slhlp.xml”示例

以下示例将讲述如何使 SINUMERIK Operate 识别帮助手册“hmi_myhelp.xml”。

没有激活此帮助手册的关键词目录。

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE CONFIGURATION>
<CONFIGURATION>
   <OnlineHelpFiles>
     <hmi_myHelp>
       <EntriesFile value="hmi_myhelp.xml" type="QString"/>
       <DisableIndex value="frue" type="bool"/>
     </hmi_myHelp>
   </OnlineHelpFiles>
</CONFIGURATION>
```

5.3.5 保存帮助文件

在目标系统中保存帮助文件

1. 打开目录 /**oem**/sinumerik/hmi/hlp 并在其中为所需的语种创建一个文件夹。请使用规定的语种标识。
   文件夹名称只能为小写。
   例如：若目标语言为德语和英语，则创建文件夹“deu”和“eng”。
2. 将各自的帮助手册（如“hmi_myhelp.xml”）分别保存至“deu”和“eng”文件夹中。
3. 将帮助文件复制至如下目录中，比如：德语为：/**oem**/sinumerik/hmi/hlp/deu/hmi_myhelp；英语为：/**oem**/sinumerik/hmi/hlp/eng/hmi_myhelp。

4. 将配置文件"slhlp.xml"保存至目录/**oem**/sinumerik/hmi/cfg 中。
5. 重新启动 HMI。

说明

在显示帮助手册的内容和关键词目录时，为了进行快速处理，可在目录/siemens/sinumerik/sys_cache/hmi/hlp 下将帮助文件保存为二进制格式 (slhlp_<Hilfe-Buch_*.hmi)。修改帮助手册时，必须删除此文件。

5.3.6 PDF 格式的帮助文件

除了 HTML 格式的帮助文件以外，还可以在操作软件中插入 PDF 格式的信息。通过目录或索引中的链接可以打开单个 PDF 帮助文件，或直接打开 HTML 文件。

PDF 帮助文件存档

将 PDF 帮助文件复制到以下任何一个目录中：

```
/oem/sinumerik/hmi/hlp/<lng>/<hmi_myhelp>
```

```
/user/sinumerik/hmi/hlp/<lng>/<hmi_myhelp>
```

插入 PDF 帮助文件

在对话项目或目录和索引项目中插入后缀为"PDF"的文件，插入方法和插入"HTML"文件一样：

```
<ENTRY ref="myFile.pdf" title="Help 1">
```

从 HTML 文件链接到 PDF 帮助文件：

```
<a href="myFile.pdf">My Help File</a>
```

说明

在 PDF 帮助文件中无法选择上下文相关的跳转标签或跳转到其他 HTML 或 PDF 文件。
搜索功能仅适用一个 PDF 文件内。不支持在多个 PDF 帮助文件中进行高级搜索。

变量 6

6.1 定义变量

变量值

变量的重要属性即变量值。

可以通过如下方式赋值：

- 定义变量时预设
- 分配系统或者用户变量
- 采取方法

编程

句法：	名称 **val** = 变量值 名称 = 变量值	
说明：	变量值 val (value)	
参数：	名称：	变量名称
	变量值	变量值
示例：	`VAR3 = VAR4 + SIN(VAR5)` `VAR3.VAL = VAR4 + SIN(VAR5)`	

变量状态

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。 该属性可通过值 FALSE = 0 读写。

编程

句法：	名称.**vld**
说明：	变量状态 vld (validation)

参数：	名称：	变量名称
		查询结果可能是：
	FALSE =	无效值
	TRUE =	有效值
示例：	IF VAR1.VLD == FALSE VAR1 = 84 ENDIF	

6.2 应用举例

辅助变量

辅助变量是内部计算变量。计算变量如同变量一样定义，但是另外除了变量值和状态之外没有属性，即辅助变量在对话框中不可见。辅助变量是 VARIANT 类型。

编程

句法：	DEF [命名符]	
说明：	VARIANT 类型的内部计算变量	
参数：	标签：	辅助变量名称
示例：	DEF OTTO；定义一个辅助变量	

句法：	名称 val = 辅助变量值 名称 = 辅助变量值	
说明：	辅助变量值在方式中指定。	
参数：	标签：	辅助变量名称
	辅助变量值：	辅助变量内容

示例：

```
LOAD
  OTTO = "Test"              ; 为辅助变量 Otto 赋值 "Test"
END_LOAD
LOAD
  OTTO = REG[9].VAL          ; 为辅助变量 Otto 赋寄存器的值
END_LOAD
```

通过变量计算

在每次退出输入/输出栏（通过 ENTER 或者转换键）后计算变量。计算在 CHANGE 方式中设计并在每次更改值时运算。

通过变量状态可以查询，变量是否包含有效值，例如：

```
IF VAR1.VLD == FALSE
   VAR1 = 84
ENDIF
```

系统变量间接编译地址

系统变量也可以间接编译地址，即和另一个变量相关。

```
PRESS(HS1)
  ACHSE=ACHSE+1
  WEG.VAR="$AA_DTBW["<<ACHSE<<"]"        ；通过变量编译轴地址
END_PRESS
```

更改软键标签

示例：

```
HS3.st = “新文本”                          ；更改软键标签
```

6.3 示例 1： 分配变量类型、文本、帮助画面、颜色、提示框

示例 1a

下面的示例中会定义一个变量，并对它的变量类型、文本、辅助画面和颜色属性进行设置。

DEF Var1 = (R///,"实际值",,"mm"//"Var1.png"////8,2)			
	变量类型：		REAL
	文本：		
		短文本：	实际值
		单位文本：	mm
	帮助图形：		Var1.png
	颜色：		
		前景色：	8（棕色）
		背景色：	2（橙色）

示例 1b

下面的示例中会定义一个变量，并对它的变量类型、预设值、文本、提示框、输入模式和短文本位置属性进行设置。

DEF Var2 = (I//5/"","值","","","提示框文本"/wr2///20,250,50)			
	变量类型：		INTEGER
	预设值：		5
	文本：		
		短文本：	值（可能的语言文本 ID）
		提示框：	提示框文本
	属性：		
		输入模式	读和写
	短文本位置：		
		到左边缘的间距	20
		到上边缘的间距	250
		宽度：	50

参见

变量参数 (页 107)

6.4 举例 2： 定义变量类型，极限值，属性，短文本位置

示例 2

下面的示例中会定义一个变量，并对它的变量类型、极限值、输入模式、对齐和位置属性进行设置。

DEF Var2 = (I/0,10///wr1,al1/// , ,300)			
	变量类型：		INTEGER
	极限值或者转换栏输入项：		MIN： 0 MAX： 10
	属性：		
		输入模式	只读
		短文本的文本对齐	右对齐
	短文本位置：		
		宽度：	300

参见

变量参数 (页 107)

6.5 举例 3： 定义变量类型、预设、系统或者用户变量、输入/输出栏位置

示例 3

下面的示例中会定义一个变量，并对它的变量类型、预设值、系统变量或用户变量和位置属性进行设置。

DEF Var3 = (R//10////"$R[1]"//300,10,200//)			
	变量类型：		REAL
	预设值：		10
	系统变量或者用户变量：		$R[1] （R 参数 1）
	短文本位置：		相对于输入/输出栏的标准位置
	输入/输出栏位置：		
		到左边缘的间距	300
		到上边缘的间距	10
		宽度：	200

参见

变量参数 (页 107)

6.6 示例 4： 转换栏和列表栏

示例 4a

转换栏中不同的条目：

```
DEF Var1 = (I/* 0,1,2,3)                    ; 用于数值转换的简单转换栏
DEF Var2 = (S/* "On", "Off")                ; 用于字符串转换的简单转换栏
DEF Var3 = (B/* 1="On", 0="Off")            ; 用于数值转换的扩展转换栏，为每个数值分配了一个显示文本
DEF Var4 = (R/* ARR1)                       ; 转换栏中要转换的值基于一个数组
```

示例 4b

列表栏与转换栏的配置一样，但要额外设置列表栏的显示类型（变量属性 DT = 4）。

DEF VAR_LISTBOX_Text = (S/*$80000,$80001,$80002,$80003,$80004/$80001// DT4////200,,340,60)		
变量类型：		字符串
极限值/转换栏：		* $80000,$80001,$80002,$80003,$80004 （要显示的与语言相关的文本列表）
预设值：		$80001
属性：		
	显示类型：	4（列表栏）
输入/输出栏位置：		
	左边距：	200
	宽度：	340
	高度：	60
颜色：		
	前景色：	6（蓝色）
	背景色：	10（白色）

6.7 示例 5： 图片显示

示例 5

显示图片，而不是短文本： 图片的大小和位置在“输入/输出栏位置（左侧、上部、宽度、高度）”中规定。

DEF VAR6 = (V///,"\\bild1.png" ////160,40,50,50)			
	变量类型：		VARIANT
	文本：		
		短文本：	bild1.png
	输入输出栏的位置：		
		左边距：	160
		上边距：	40
		宽度：	50
		高度：	50

6.8 示例 6： 进度条

进度条是一种特殊的输入/输出栏显示类型并且只用于显示而不能输入。

原则上有两种进度条类型：

1. 最多可进行两次变色的进度条，例如温度或负载显示（见示例 6a）。
2. Operate 风格的进度显示（不变色）条（见示例 6a）

示例 6a

两次变色的进度条：



图 6-1 两次变色的进度条

DEF PROGGY0 = (R/0,150,50,100///DT1,DO0//"$R[10]"//,,150/3,4,,,,,,9,7)		
变量类型：		REAL
极限值/转换栏：		
	MIN：	0
	MAX：	150
	信号值 SVAL1：	50
	信号值 SVAL2：	100
属性：		
	显示模式 DT：	1（进度条）
	显示选项 DO：	0（从左往右（默认））
系统变量或者用户变量：		$R[10]
输入/输出栏位置：		
	宽度：	150
颜色：		
	前景色：	3（深绿）
	背景色：	4（浅灰）
	信号色 SC1：	9（黄色）
	信号色 SC2：	7（红色）

如要使用可变色的进度条应将显示模式 DT (DisplayType)设为 1。

进度显示的方向通过属性“显示选项 DO (DisplayOption)”来确定：

0:从左往右（默认）

1:从右往左

2:从下往上

3:从上往下

进行进度条的显示时，应给定最小值和最大值（示例中：MIN：0，MAX：150）。

在此设置下，取决于变量 PROGGY0 的当前值，进度条将显示前景色 3（= 深绿）和背景色 4（= 浅灰）。

另外还可以选择定义一或两个信号值 SVAL1 和 SVAL2（参数限值）（示例中：SVAL1: 50 以及 SVAL2: 100）。达到这两个信号值时，进度显示的前景色会变化。相应的信号色通过参数 SC1 和 SC2 确定（上例中 SC1: 9（= 黄色）以及 SC2: 7（= 红色））。

进度显示的限值给定适用如下规则：

MIN < SVAL1 < SVAL2 < MAX。

示例 6b

不变色进度条：



图 6-2 进度条

DEF PROGGY0 = (R/0,150///DT2,DO0//"$R[10]"//,,150/6,10)		
变量类型：		REAL
极限值/转换栏：		
	MIN：	0
	MAX：	150
属性：		
	显示类型 DT：	2（进度条）
	显示选项 DO：	0（从左往右（默认））
系统变量或者用户变量：		$R[10]
输入/输出栏位置：		
	宽度：	150
颜色：		
	前景色：	6（蓝色）
	背景色：	10（白色）

如要使用不变色的进度条应将显示模式 DT (DisplayType)设为 2。

进度显示的方向通过属性“显示选项 DO (DisplayOption)”来确定（参见示例 6a 的说明）：

进行进度条的显示时，应给定最小值和最大值（示例中：MIN：0，MAX：150）。

在此设置下，取决于变量 PROGGY0 的当前值，进度条将显示前景色 6（= 蓝色）和背景色 10（= 白色）。

6.9 示例 7： 口令输入模式（星号）

示例 7

如要对输入栏中的输入内容实现隐藏，例如在输入口令时，必须将显示模式 DT 设为 5。所输入的字符会显示为星号。



图 6-3 口令输入模式（星号）

DEF VAR_SET_PWD=(S/""//DT5)			
	变量类型：		STRING
	预设值：		空字符串
	属性：		
		显示模式 DT：	5（口令输入模式）

6.10 变量参数

参数 - 一览

在以下概述中简要说明了变量参数。详细的说明请参见后续章节。

参数	说明	
变量类型 (页 115)	必须规定变量类型。	
	R[x]:	REAL（小数点位数为正数 +）
	I:	INTEGER
	S[x]:	字符串（字符串长度为正数 +）
	C:	字符（单字符）
	B：	BOOL
	V:	VARIANT
极限值 (页 99)	最小极限值，最大极限值 预设置：空 极限值通过逗号隔开。类型 I、C 和 R 的极限值可以使用十进制的格式或者字符"A"、 "F"定义。	
	对于可变色的进度条显示（变量属性 DT = 1），可选择配置两个信号色 SC1 和 SC2 (Signal Color)，在超出信号值 SVAL1 或 SVAL2 (Signal Value)时会使用相应的进度条前景色。信号值为整型值。 参见进度条 (页 103)应用示例。	
预设值 (页 121)	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，即：变量处于状态“未计算”。 预设置：未预设	
转换栏 (页 119)	输入/输出栏中带有预设输入项的列表：列表通过 * 开始，各输入项用逗号隔开。可以赋值输入项。 极限值的输入项视为转换栏的列表。如果只输入一个 *，则建立一个可变的转换栏。 预设置：无	

参数	**说明**	
文本 (页 97)	顺序已预先规定。也可以显示一个图形代替短文本。 预设置：空	
	长文本：	显示行的文本
	短文本：	对话框单元的名称
	图形文本：	文本参考图形名称。
	单位文本：	对话框单元的单位
	提示框	在设计窗口时，用于为各个显示栏和切换栏提供简短的说明信息。该信息通过纯文本和语言文本 ID 设计。

参数	说明	
属性 (页 99)	属性影响下列特性： • 显示模式 • 显示选项 • 刷新率 • 转换符号 • 提示框 • 输入模式 • 访问等级 • 短文本的文本对齐 • 字体大小 • 极限值 属性通过逗号隔开，顺序任意每个组件都可以进行定义。	
	显示模式	dt0：标准（输入/输出栏或转换栏）（默认） dt1：可变色的进度条： dt2：Operate 风格的不变色进度条 dt4：列表栏 dt5：口令输入模式（星号）
	显示选项	特别是对于显示模式 dt1 和 dt2（进度条），还需要同时设置显示选项。 do0：从左往右（默认） do1：从右向左 do2：从下往上 do3：从上往下
	刷新速度	使用属性 UR (update rate)可以控制显示的刷新以及相应配置的变量或 Grid 列的 CHANGE 块的处理。根据配置的不同，CPU 负载可能被大幅降低并且可达到更短的用户接口响应时间。 ur0：SlCap::standardUpdateRate()（当前 = 200 ms，默认值） ur1：50 ms ur2：100 ms ur3：200 ms ur4：500 ms ur5：1000 ms ur6：2000 ms

参数	说明	
		ur7：5000 ms ur8：10000 ms
	转换符号	tg0：转换符号关闭（默认） tg1：转换符号打开 如果将属性 TG 设为 1，则在输入栏提示框中还会额外显示转换符号。 示例： `DEF OFFS = (R//123.456/,,,,"My ToolTip"/TG1)`

	提示框	提示框文本在运行期间可通过变量属性 “TT” 来更改。
		示例：
		`PRESS(VS1)`
		`MyVar.TT = "My new ToolTip"`
		`END_PRESS`
		或
		`DEF MyVar=(R///,,,,"My ToolTip-text")`

	输入模式	wr0:输入/输出栏不可见，短文本可见 wr1:读取（没有输入中心） wr2:读取和写入（行以白色显示） wr3:wr1 带输入中心 wr4:所有变量单元不可见，没有输入中心 wr5:按下任何键立即保存输入的值（和 wr2 相反 - 该模式下，在退出栏或者按下返回键后才开始保存值）。 预设置：wr2
	访问等级	空：总是可以写入 ac0...ac7:保护等级 如果访问等级未达到，行显示为灰色，标准设置：ac7
	短文本的文本对齐	al0:左对齐 al1:右对齐 al2:居中 预设置：al0
	字体大小	fs1:标准字体大小（8 Pt） fs2:双倍字体大小

参数	说明	
		预设置：fs1 确定行间距。标准字体大小为每对话框 16 行。图形和单位文本只能为标准字体大小。
	极限值	通过极限值可以检查变量值是否在规定的最小极限值和最大极限值之内。 预设置：取决于规定的极限值 li0:没有检查 li1:检查最小极限值 li2:检查最大极限值 li3:检查最大极限值和最小极限值
	打开时的属性	定义变量时规定的 cb 属性，优先于定义对话框时的总定义。多个属性通过逗号隔开记录。（参见 AUTOHOTSPOT）
	CHANGE 方式调用	CB0:如果变量当前有一个有效值（例如：通过预设值或 NC/PLC 变量），则通过显示屏幕触发 CHANGE 方法。 CB1：通过显示屏幕无法显式触发 CHANGE 方法。如果变量有一个配置的 NC/PLC 变量，则一定会调用 CHANGE 方法。
	Empty Input (EI)	通过变量属性 "EI" (= Empty Input) 可确定，当输入空字符串时，输入栏应如何反应： EI0:默认设置，即接受输入栏中的空白输入。 EI1:空白输入导致输入栏出错并会再次设定之前的生效值（Undo/撤销）。
帮助画面 (页 97)	帮助画面文件：	png 文件的名称 预设置：空
	帮助画面文件的名称在双引号中。如果光标移至该变量，则自动显示画面（替代目前的图形）。	
系统或者用户变量 (页 100)	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，/PGAsl/	
短文本的位置 (页 122)	短文本位置（和左边缘/上边缘的间距、宽度） 位置数据的单位为像素，并且以对话框主体的左上角为基准。数据总是用逗号隔开。	

参数	说明
输入/输出栏的位置 (页 122)	输入/输出栏位置（和左边缘/上边缘的间距、宽度、高度） 位置数据的单位为像素，并且以对话框主体的左上角为基准。数据总是用逗号隔开。如果改变此位置，则短文本、图形文本和单位文本的位置也同时改变。
颜色 (页 97)	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。 颜色数据参见章节 AUTOHOTSPOT。 输入/输出栏的缺省设置：前景色：黑色，背景色：白色 输入/输出栏的标准颜色取决于写入模式 “wr”： 短文本、图形文本、单位文本的缺省设置：前景色：黑色，背景色：透明
	对于可变色的进度条显示（变量属性 DT = 1），可选择配置两个信号色 SC1 和 SC2 (Signal Color)，在超出信号值 SVAL1 或 SVAL2 (Signal Value)时会使用相应的进度条前景色。 参见进度条 (页 103)应用示例。
	变量定义中要求的颜色顺序如下： 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 10.信号色 2

参数	说明
在线帮助文件	在线帮助文件的名称在双引号中。

集成了单位选择框的输入/输出栏	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。如果将光标置于输入栏中，集成的单位选择框会被高亮显示（焦点无需显式设置在其上）。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。
	示例：
	`DEF VarEdit=(R////////200,,100///"VarTgl")`
	`VarTgl=(S/*0="mm",1="inch"/0//WR2////302,,40)`
	或
	`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}`

变量：更改属性

以格式“名称.属性 = 值”重新赋值。系统会分析等号右边的表达式并为变量赋值或者分配变量属性。

名称.**ac** = 存取级	(ac:access level)
名称.**al** = 文本对齐	(al:alignment)
名称.**bc** = 输入/输出栏的背景色	(bc:back color)
名称 **bc_gt** = 图形文本的背景色	(bc:back color) (gt:graphic text)
名称.**bc_st** = 短文本的背景色	(bc:back color) (st:short text)
名称.**bc_ut** = 单位文本的背景色	(bc:back color) (ut:unit text)
名称 **do** = 显示选项	(do:display option)
名称.**dt** = 显示模式	(dt:display type)
名称 **.fc** = 输入/输出栏的前景色	(fc:front color)
名称.**fc_gt** = 图形文本的前景色	(fc:front color) (gt:graphic text)

名称.**fc_st** = 短文本的前景色	(fc:front color) (st:short text)
名称.**fc_ut** = 单位文本的前景色	(fc:front color) (ut:unit text)
名称.**al** = 字体大小	(fs:font size)
名称.**gt** = 图形文本	(gt:graphic text)
名称.**hlp** = 帮助图形	(hlp:help)
名称.**li** = 极限值	(li:limit)
名称.**lt** = 长文本	(lt:long text)
名称.**max** = 最大极限值	(max:maximum)
名称.**min** = 最小极限值	(min:minimum)
名称.**sc** = 信号色	(sc:signal color)
名称.**st** = 简要说明	(st:short text)
名称.**tg** = 转换符号	(tg:toggle)
名称.**tt** = 提示框	(tt:提示框
名称.**typ** = 变量类型	(typ:type)
名称.**ur** = 刷新速度	(ur:update rate)
名称.**ut** = 单位文本	(ut:unit text)
名称 **val** = 变量值	(val:value)
名称.**var** = 系统或者用户变量	(var:variable)
名称.**vld** = 变量状态	(vld:validation)
名称.**wr** = 输入模式	(wr:write)

另见

扩展配置句法 (页 51)

6.11 各个变量类型的详细说明

INTEGER 型变量

“ INTEGER”整型变量可以使用下列扩展符号显示在输入/输出栏中并保存在存储器中。

扩展数据类型中的**第 2 个字符**

表示格式	
B	二进制
D	十进制，有符号
H	十六进制
没有数据	十进制，有符号

扩展数据类型中的**第 3 个和第 4 个字符**

存储器保存	
B	字节
W	单字
D	双字
BU	不带正负号的字节
WU	字，无符号
DU	双字，无符号

INTEGER 型数据的字符顺序

1. “I” 主标识符，表示 INTEGER（整型）
2. 表示格式
3. 存储器保存
4. “U”，无符号

有效的整型定义：	
IB	整数变量 32 位二进制表示
IBD	整数变量 32 位二进制表示
IBW	整数变量 16 位二进制表示

有效的整型定义：	
IBB	整数变量 8 位二进制表示
I	整数变量 32 位十进制，有符号
IDD	整数变量 32 位十进制，有符号
IDW	整数变量 16 位十进制，有符号
IDB	整数变量 8 位十进制，有符号
IDDU	整数变量 32 位十进制，无符号
IDWU	整数变量 16 位十进制，无符号
IDBU	整数变量 8 位十进制，无符号
IH	整数变量 32 位十六进制
IHDU	整数变量 32 位十六进制
IHWU	整数变量 16 位十六进制
IHBU	整数变量 8 位十六进制

VARIANT 型变量

VARIANT 型变量通过最后赋值的数据类型确定。如果分配或输入的值是以 '-'、'+'、'.' 或数字 ('0'-'9') 开头的，该值则被编译为数值。其它情况下作为字符串。

它可以通过功能 ISNUM 或者 ISSTR 查询。VARIANT 型主要用于向 NC 码写入变量名或者数值。

编程

可以检查变量的数据类型：

句法：	**ISNUM** (VAR)	
参数：	VAR	待检查其数据类型的变量名称。
		查询结果可能是：
	FALSE =	没有数值（数据类型 = STRING）
	TRUE =	数值（数据类型 = REAL）

句法：	**ISSTR** (VAR)	
参数：	VAR	待检查其数据类型的变量名称。
	FALSE = TRUE =	查询结果可能是： 数值（数据类型 = REAL） 没有数值（数据类型 = STRING）
示例：	IF ISNUM(VAR1) == TRUE IF ISSTR(REG[4]+2) == TRUE	

可以改变变量的显示模式：

- 对于 INTEGER 型变量可以改变显示方式。

B	二进制
D	十进制，有符号
H	十六进制
不带正负号	
另加一个 U，表示 unsigned（无符号）	

- 对于 REAL 型变量，只能更改小数点的位数。
  不允许更改基本类型，若更改，则导致文件“easyscreen_log.txt”出现错误报告。
  示例：
  ```
  Var1.typ = "IBW"
  Var2.typ = "R3"
  ```

数字格式

数字可以以二进制、十进制、十六进制或者指数方式描述。

二进制	B01110110
十进制	123.45
十六进制	HF1A9
指数	-1.23EX-3
示例：	
	VAR1 = HF1A9

	`REG[0]= B01110110`
	`DEF VAR7 = (R//-1.23EX-3)`

说明

在通过功能 GC 生成代码时计算值仅考虑十进制或者指数描述方式，而不使用二进制和十六进制描述方式。

参见

变量参数 (页 107)

6.12 转换栏的详细说明

说明

通过转换栏扩展可以显示与 NC/PLC 变量有关的文本（在转换栏中的输入项）。只能读取使用转换栏扩展的变量。按下 INSERT 按键，打开转换栏列表。

编程

句法：	**DEF VAR1=(IB/+ $85000/15////"DB90.DBB5")** 或者 **DEF VAR_TGL = (S/* "Hello", "Run", "MyScreens"/"Run")**	
说明：	显示对话框时，在输入/输出栏中显示文本号码 $85015 的内容。在系统变量 DB90.DBB5 输入预设值 15。如果系统变量 DB90.DBB5 中的值改变，在每次改变时显示的文本号码重新生成 $（85000 + <DB90.DBB5>）。	
参数：	变量类型	系统或者用户变量中指定的变量类型
	文本号码	与语言相关的文本号码（基本），该号码作为基本号码使用
	系统或者用户变量	通过最终的文本号码（基本＋补偿）形成的系统或者用户变量（补偿）。

图形与转换栏有关

转换栏用替换的图像覆盖。如果标记字节值为 1，则显示“bild1.png”；如果标记字节值为 2，则显示 “bild2.png”。

```
DEF VAR1=(IDB/*1="\\bild1.png", 2="\\bild2.png"//,$85000/wr1//"MB[130]"//160,40,50,50)
```

图形的大小和位置在“输入/输出栏位置（左侧、上部、宽度、高度）”中规定。

虚拟转换键

未配置列表的转换栏，例如 `DEF NoTglList=(R/*)`，不带有任何列表。下个元素在按下转换键或运行完相应变量的相应 CHANGE() 方法后才会生成。

此时，当操作触摸屏时，一个虚拟的小键盘会显示在可变转换栏的右侧，该转换栏中只有转换键。

通过配置文件 “slguiconfig.ini” 中的以下条目也可以强制取消虚拟转换键的显示与触摸屏的关联。

```
[VirtualKeyboard]
; Forces easyscreen virtual toggle keyboard function
ForceEasyscreenVirtualToggleKey = true
```

参见

变量参数 (页 107)

6.13 预设值的详细说明

概述

根据变量栏（输入/输出栏或者转换栏）是否分配了一个预设值，一个系统或者用户变量或者两者都分配，得到不同的变量状态。 只有当变量分配了一个有效值时，转换才可行。

预设值生效

条件			栏类型反应
栏类型	预设值	系统或者用户变量	
输入/输出栏	是	是	在系统或者用户变量中写预设值
	否	是	使用系统或者用户变量作为预设值
	错误	是	未计算，系统或者用户变量未描述/未使用
	是	否	预设值
	否	否	未计算
	错误	否	未计算
	是	错误	未计算
	否	错误	未计算
	错误	错误	未计算
转换	是	是	在系统或者用户变量中写预设值
	否	是	使用系统或者用户变量作为预设值
	错误	是	未计算， 系统或者用户变量未描述/未使用
	是	否	预设值
	否	否	预设值 =转换栏的第一个单元
	错误	否	未计算
	是	错误	未计算
	否	错误	未计算
	错误	错误	未计算

参见

变量参数 (页 107)

6.14 短文本位置、输入输出栏位置的详细说明

概述

短文本和图形文本以及输入/输出栏和单位文本总是形成一个单元。即短文本的位置数据也对图形文本和输入/输出栏数据以及单位文本上的数据有效。

编程

设计的位置数据覆盖标准值，即也仅能改变一个单独的值。如果下列对话框单元没有设计位置数据，则采用上一对话框单元的数据。

如果对话框单元没有规定位置，则使用预设置。短文本和输入/输出栏的栏宽度在标准情况下各行由栏数和最大栏宽度确定，即：栏宽度=最大行宽/列数。

图像和单位文本宽度是固定的，根据编程支持的请求优化。如果已设计图像和单位文本宽度，则短文本或者输入/输出栏的宽度相应缩短。

短文本和输入/输出栏的顺序可以通过位置数据互换。

输入/输出栏和单位栏之间的间距和单位栏宽度

可以配置输入/输出栏和单位栏之间的间距和单位栏宽度。

在定义行中输入输入/输出位置章节中用逗号隔开的输入/输出栏与单位栏之间的间距（例如 7 像素）和/或单位栏的宽度（例如 60 像素）：

```
DEF VarDT=(R3//0.000/,"DT",,"s"////0,,24/39,,71,,7,60)
```

或者：

```
DEF VarDT={TYP="R3", VAL="0.000", ST="DT", UT="s", TXT_X=0,
TXT_W=24, X=39, W=71, UT_DX=7, UT_W=60}
```

在配置了输入/输出栏/单位栏之间的间距和/或单位栏宽度时，需要注意以下几点：

- 配置的输入/输出区不包含单位栏的固定宽度（固定 50 像素）。即直接配置输入/输出栏宽度。
- 如果没有配置单位栏宽度，则宽度默认为 50 像素。
- 如果没有配置输入/输出栏/单位栏之间的间距，则间距默认为 0 像素。

组合转换栏的特性

前提条件：

- 一个变量配置了一个组合转换栏，
- 变量的输入/输出栏和单位栏之间的间距和/或单位栏宽度已经配置且
- 变量无单位文本。

该情况下，组合转换栏定位在变量单位栏的配置位置处。

忽略可能存在的、为组合转换栏的输入/输出部分配置的位置。

示例

以下示例中组合转换栏 **F_Unit** 自动以与变量 **VarF** 的输入/输出栏 **7 像素**的间距定位并设置宽度 **59 像素**。

```
DEF VarF=(R//0.0/,"F",,,////0,,24/39,,85,,7,59///"F_Unit"),
F_Unit = (I/*3="mm/min", 1="mm/U"/3// ////181,,155)
```

参见

变量参数 (页 107)

6.15 使用字符串

字符串链

设计时也可以使用字符串，以动态配置文本显示或者合并代码生成的不同文本。

规则

使用字符串变量时注意以下规定：

- 链接由左向右处理。
- 层叠的表达式由内向外运算。
- 忽略大小写。
- 字符串变量通常左对齐显示。

字符串可以通过一个简单的空字符串指令删除。

字符串在等号右边以运算符 "<<"开始。 字符串中双引号（"）通过两个连续的双引号标记。 字符串可以在 IF 指令中检查相等性。

示例

下列示例预设：

```
VAR1.VAL =“这是一个”
VAR8.VAL = 4
VAR14.VAL = 15
VAR2.VAL = “错误”
$85001 = “这是一个”
$85002 = “报警文本”
```

编辑字符串：

- 合并字符串：
  ```
  VAR12.VAL = VAR1 << “错误”    ;结果： “这是一个错误”
  ```
- 删除一个变量：
  ```
  VAR10.VAL = ""    ;结果： 空字符串
  ```
- 设定一个带有文本变量的变量：
  ```
  VAR11.VAL = VAR1.VAL    ;结果： “这是一个”
  ```

- 数据类型匹配：

```
VAR13.VAL ="这是 " << (VAR14 - VAR8) << ". 错误"
            ;结果： “这是第 11 个错误”
```

- 处理数字值：

```
VAR13.VAL = “错误” << VAR14.VAL << ": " << $85001 << $85002
            ;结果： “错误 15： “这是一个报警文本”
IF VAR15 == “错误”  ; IF 指令中的字符串
  VAR16 = 18.1234
            ;结果： VAR16 等于 18.1234，
     ; 当 VAR15 等于“错误”时
ENDIF
```

- 字符串中的双引号：

```
VAR2=“你好，这是一个“测试”“
            ;结果： 你好，这是一个“测试”
```

- 和变量内容有关的系统或者用户变量的字符串：

```
VAR2.Var = "$R[" << VAR8 << "]"   ; 结果： $R[4]
```

另见

字符串函数 (页 208)

6.16 变量 CURPOS

说明

通过变量 CURPOS 可以在当前对话框的激活的输入栏中调出或者操纵光标位置。 变量显示光标前有多少个字符。 如果光标位于输入栏开始处，则 CURPOS 接受值为 0。 如果更改 CURPOS 值，则光标停留在输入栏中相应的位置上。

为了可以在变量值更改情况下反应，可以借助于一个 CHANGE 方法监控改变情况。 如果 CURPOS 值改变，则跳转 CHANGE 方法并执行包含的指令。

6.17 变量 CURVER

说明

属性 CURVER（当前版本） 允许匹配编程用于处理不同的版本。 变量 CURVER 仅可读。

说明

在代码生成时自动以最新的版本生成，即使之前已用老的版本反编译。 命令 "GC" 总是生成最新的版本。 在生成代码中，在使用注释当版本 > 0 时插入一个生成的代码的附加标记。

规则

总是显示带有所有变量的最新的对话框。

- 以前的变量不允许改变。
- 新的变量以任意顺序插入在以前（循环）的编程中。
- 不允许从对话框由一个版本到下一个版本去除变量。
- 对话框必须包含所有版本的变量。

举例

```
(IF CURVER==1 ...)                    ; CURVER 在反编译时自动通过反编译代码的版本设置。
```

6.18 变量 ENTRY

说明

通过变量 ENTRY 可以检查如何调用对话框。

编程

句法：	**ENTRY**	
说明：	变量 ENTRY 仅可读。	
返回值：		查询结果可能是：
	0 =	没有编程支持
	1 =	通过软键启动屏幕；还未生成任何代码（预设值，同配置）
	2 =	通过软键启动屏幕；已生成代码 （预设值来自最后由该屏幕生成的代码）
	3 =	重新编译，带使用注释（#行）
	4 =	Code_typ = 0：带使用注释的 NC 代码（#行）
	5 =	Code_typ = 1：无使用注释的 NC 代码（#行）

示例

```
IF ENTRY == 0
  DLGL（“在编程下不调用对话框”）
ELSE
  DLGL（“在编程下调用对话框”）
ENDIF
```

6.19 变量 ERR

说明

通过变量 ERR 可以检查是否已正确执行先前行。

编程

句法：	**ERR**	
说明：	变量 ERR 仅可读。	
返回值：		查询结果可能是：
	FALSE =	已正确执行先前行
	TRUE =	未正确执行先前行

示例

```
VAR4 = 螺纹[VAR1,"KDM",3]                ；作为数组给出值
IF ERR == TRUE                           ；查询是否在数组中找到值
VAR5 = “数组存取出错”
                                         ；如果未在数组中找到值，则赋值“数组存取出错”给变量。
ELSE
VAR5 = “一切正常”                        ；如果在数组中找到值，则赋值“一切正常”给变量。
ENDIF
```

6.20 变量 FILE_ERR

说明

用变量 FILE_ERR 可以检查前面的 GC 或者 CP 指令是否正确执行。

编程

句法：	**FILE_ERR**	
说明：	变量 FILE_ERR 仅可读。	
返回值：		可能的结果是：
	0 =	排列中的操作
	1 =	驱动器/路径不存在
	2 =	路径/文件存取故障
	3 =	驱动器未就绪
	4 =	错误的文件名
	5 =	文件已经打开
	6 =	存取失败
	7 =	目标路径不存在或不允许
	8 =	复制源符合目标
	10 =	内部错误： FILE_ERR = 10 时，与一个未分类到其他类别故障有关。

示例

```
CP("D:\source.mpf","E:\target.mpf")
                                                    ; 从 source.mpf 向 E:\target.mpf 复制
IF FILE_ERR > 0                                     ; 询问是否出现故障
IF FILE_ERR == 1                                    ; 询问特定的错误号并输出所属的故障文本
VAR5 = "驱动器/路径不存在"
ELSE
IF FILE_ERR == 2
VAR5 = "路径一/文件存取故障"
ELSE
IF FILE_ERR == 3
```

```
CP("D:\source.mpf","E:\target.mpf")
VAR5 = "错误文件名"
ENDIF
ENDIF
ENDIF
ELSE
VAR5 = "一切正常"                                    ; 如果 CP (或者 GC)中没有出现错误
                                                     , 则输出“一切正常”
ENDIF
```

6.21 变量 FOC

说明

使用变量 FOC 可以控制对话框内的输入中心（当前有效的输入/输出栏）。 已预先固定定义光标左右键、前后键以及 PGUP、PGDN 的反应。

说明

FOC 不允许通过一个导航事件触发。 光标位置只允许在软键 PRESS 方法、CHANGE 方法、... 中改变。

带有输入模式 wr = 0 和 wr = 4 的变量以及辅助变量无法实现聚焦。

编程

句法：	**FOC**	
说明：	可以读取和写入该变量。	
返回值：	读取	聚焦变量的名称作为结果输出。
	写入	可以写入一个字符串或者数值。 字符串视为变量名称，数值视为变量索引。

示例

```
IF FOC == "Var1"                ; 读取输入中心
  REG[1] = Var1
ELSE
  REG[1] = Var2
ENDIF

FOC = "Var1"                    ; 分配变量 1 给输入中心。
FOC = 3                         ; 分配第 3 对话框单元， WR ≥ 2 给输入中心。
```

参见

FOCUS (页 148)

6.22 变量 S_ALEVEL

说明

在配置中会通过屏幕属性 S_ALEVEL 询问当前访问级。

编程

句法：	**S_ALEVEL**
说明：	查询当前访问级
返回值：	0: 系统
	1: 制造商
	2: 维修
	3: 用户
	4: 钥匙开关位置 3
	5: 钥匙开关位置 2
	6: 钥匙开关位置 1
	7: 钥匙开关位置 0

示例

```
REG[0] = S_ALEVEL
```

参见

ACCESSLEVEL (页 145)

6.23 变量 S_CHAN

说明

通过变量 S_CHAN 可以确定用于显示或者某个评估的当前通道号码。

编程

句法：	**S_CHAN**
说明：	查询当前通道号
返回值：	通道号

示例

```
REG[0] = S_CHAN
```

参见

CHANNEL (页 147)

6.24 变量 S_CONTROL

说明

在配置中会通过屏幕属性 S_CONTROL 询问当前控制系统名称。

编程

句法：	**S_CONTROL**
说明：	查询当前控制系统名称
返回值：	”控制系统名称为 “mmc.ini" 中的段落名。

示例

```
REG[0] = S_CONTROL
```

参见

CONTROL (页 148)

6.25 变量 S_LANG

说明

在配置中会通过屏幕属性 S_LANG 询问当前语言。

编程

句法：	**S_LANG**
说明：	查询当前语言
返回值：	resources.xml 中的语言缩写，例如：“deu”、“eng” 等

示例

```
REG[0] = S_LANG
```

参见

LANGUAGE (页 149)

6.26 变量 S_NCCODEREADONLY

说明

只有在编辑器中通过对话框 “Run MyScreens” 反编译循环时，屏幕属性 S_NCCODEREADONLY 才有意义。通过 S_NCCODEREADONLY 确定是否可以修改编辑器中的反编译 NC 代码。

返回值适用条件：

- TRUE：可以修改编辑器中的反编译 NC 代码
- FALSE：由于编辑器中的反编译 NC 代码已经处于预处理状态 (= TRUE)，因此，无法进行修改（只读）。

6.27 变量 S_RESX 和 S_RESY

说明

在配置中会通过屏幕属性 S_RESX 和 S_RESY 询问当前分辨率或其 X 或 Y 分量。

示例

```
REG[0] = S_RESY
```

参见

RESOLUTION (页 155)

编程指令 7

7.1 运算符

概述

在编程时可以使用以下几种运算符：

- 数学运算符
- 比较运算符
- 逻辑（布尔）运算符
- 位运算符
- 三角函数功能

7.1.1 数学运算符

概述

数学运算符	名称
+	加法
-	减法
*	乘法
/	除法
MOD	模数运算
( )	括号
AND	和运算符
OR	或运算符
NOT	非运算符
ROUND	带有小数位的数四舍五入

示例： `VAR1.VAL = 45 * (4 + 3)`

ROUND

在执行对话框设计时，用 ROUND 指令把数值四舍五入直至小数点后面的 12 位。小数点后位置在变量栏不能显示。

使用

ROUND 通过用户的两个参数来控制:

```
VAR1 = 5,2328543
VAR2 = ROUND( VAR1, 4 )
```

结果： `VAR2 = 5,2339`

VAR1 包含在四舍五入的数中。参数"4"给出了结果中的小数点后的位数，保存在 VAR2 中。

三角函数功能

三角函数功能	名称
SIN(x)	正弦 x
COS(x)	余弦 x
TAN(x)	正切 x
ATAN(x, y)	反正切 x/y
SQRT(x)	平方根 x
ABS(x)	绝对值 x
SDEG(x)	换算为度数
SRAD(x)	换算为弧度
CALC_ASIN(x)	反正弦 x
CALC_ACOS(x)	反余弦 x

说明

这些函数处理弧度。为了换算，可以使用函数 SDEG() 和 SRAD()。

示例： `VAR1.VAL = SQRT(2)`

数学函数

数学函数	名称
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
CALC_MAX(x, y)	计算 x 与 y 哪个大

随机功能：随机数

句法：	**RANDOM**（下限值，上限值）	
说明：	**RANDOM** 功能为提供预先给定范围内的伪随机数。	
参数：	下限值	下限值 >= -32767, 下限值 < 上限值
	上限值	上限值 <= 32767

示例

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

常量

常量	
PI	3.14159265358979323846
FALSE	0
TRUE	1

示例： `VAR1.VAL = PI`

比较运算符

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

示例：

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

条件

层叠深度没有限制。

带有一个命令的条件：

```
IF
...
ENDIF
```

带有两个命令的条件：

```
IF
...
ELSE
...
ENDIF
```

7.1.2 位运算符

概述

位运算符	名称
BOR	位方式 OR
BXOR	位方式 XOR
BAND	位方式 AND
BNOT	位方式 NOT
SHL	位向左移动
SHR	位向右移动

运算符 SHL

通过运算符 SHL(SHIFT LEFT)向左移动位。此时要移动的值或者移动步数可以直接规定或者作为变量规定。如果达到数据格式极限，则已超出位，没有出错信息。

使用

句法：	变量 =值 **SHL** 步数	
说明：	向左移动	
参数：	值	要移动的值
	步数	移动步数

示例

```
PRESS(VS1)
  VAR01 = 16 SHL 2            ; 结果 = 64
  VAR02 = VAR02 SHL VAR04     ; VAR02 内容转换为 32 位格式，无符号
                              ;并且位向左移动 VAR04 内容。接着
                               32 位值重新转换回
                              变量 VAR02 的格式。
END_PRESS
```

运算符 SHR

通过运算符 SHR (SHIFT RIGHT)向右移动位。此时要移动的值或者移动步数可以直接规定或者作为变量规定。如果达到数据格式极限，则已超出位，没有出错信息。

使用

句法：	变量 =值 **SHR** 步数	
说明：	向右移动	
参数：	值	要移动的值
	步数	移动步数

示例

```
PRESS(VS1)
  VAR01 = 16 SHR 2                ; 结果 = 4
  VAR02 = VAR02 SHR VAR04         ; VAR02 内容转换为 32 位格式，无符号
                                  ;并且位向右移动 VAR04 内容。接着
                                   32 位值重新转换回
                                  变量 VAR02 的格式。
END_PRESS
```

7.2 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：

说明块	注释	参考章节
PRESS(HS1)	；方法的开始标记	
LM... LS...	；功能	参见章节 功能 (页 159)
Var1.st = ...	；改变属性	参见章节定义软键栏 (页 72) 和定义对话框属性 (页 58)
Var2 = Var3 + Var4 ... EXIT	；通过变量计算	参见章节 定义变量 (页 93)
END_PRESS	；方法的结束标记	

7.2.1 ACCESSLEVEL

说明

当在屏幕打开时更改当前访问等级时，就会运行 ACCESSLEVEL 方法。

编程

句法：	ACCESSLEVEL <指令> END_ACCESSLEVEL
说明：	访问等级
参数：	- 无 -

参见

变量 S_ALEVEL (页 133)

7.2.2 CHANGE

说明

当变量值已改变时运行 CHANGE 方法。即在 CHANGE 方法中设计变量改变时立即运行的变量计算。

单元特定的 CHANGE 方法和全局的 CHANGE 方法有所不同：

- 当特定变量值已改变时运行**单元特定的 CHANGE 方法**。如果系统或者用户变量已分配一个变量，则可以在 CHANGE 方法中循环更新变量值。
- 当改变任意一个变量值且没有设计单元特定的 CHANGE 方法时，运行**全局 CHANGE 方法**。

编程“单元特定”

句法：	CHANGE(命名符) ... END_CHANGE	
说明：	修改指定变量的值	
参数：	名称	变量名称

示例

```
DEF VAR1=(I//////"DB20.DBB1")          ; Var1 分配一个系统变量
CHANGE(VAR1)
  IF VAR1.Val <> 1
  VAR1.st="工具正确！"                  ; 如果系统变量的值 ≠ 1，则变量的简要说明为：工具正确！
    otto=1
  ELSE
    VAR1.st="注意错误！"                ; 如果系统变量的值 = 1，则变量的简要说明为：注意 错误！
    otto=2
  ENDIF
  VAR2.Var=2
```

```
END_CHANGE
```

编程“全局特定”

句法：	CHANGE() ... END_CHANGE
说明：	改变任意变量值
参数：	- 无 -

示例

```
CHANGE()
  EXIT                                    ；如果任何一个变量值改变，则退出对话框。
END_CHANGE
```

7.2.3 CHANNEL

说明

当在屏幕打开时更改当前通道时，即通道切换时，就会运行 CHANNEL 方法。

编程

句法：	CHANNEL <指令> END_CHANNEL
说明：	通道切换
参数：	- 无 -

参见

变量 S_CHAN (页 134)

7.2.4 CONTROL

说明

当在屏幕打开时更改当前控制系统时，即通常在 1:n 切换时，就会运行 CONTROL 方法。

编程

句法：	CONTROL <指令> END_CONTROL
说明：	系统切换
参数：	- 无 -

参见

变量 S_CONTROL (页 135)

7.2.5 FOCUS

说明

当对话框中聚焦（光标）定位在另一个栏上时，运行 FOCUS 方法。

方法 FOCUS 不允许通过一个导航事件触发。 光标位置只允许在软键 PRESS 方法、CHANGE 方法、...中改变。 光标移动的反应预先固定定义。

说明

在 FOCUS 方法中不允许定位在另一个变量上并且也不允许装载新的对话框。

编程

句法：	FOCUS ... END_FOCUS
说明：	光标：定位
参数：	- 无 -

示例

```
FOCUS
DLGL("聚焦已设定在变量"<< FOC <<"上。")
END_FOCUS
```

参见

变量 FOC (页 132)

7.2.6 LANGUAGE

说明

当在屏幕打开时更改当前语言时，就会运行 LANGUAGE 方法。

编程

句法：	LANGUAGE <指令> END_LANGUAGE
说明：	语言
参数：	- 无 -

参见

变量 S_LANG (页 136)

7.2.7 LOAD

说明

在已编译变量定义和软键定义（DEF Var1= ..., HS1= ...）后运行 LOAD 方法。 此时，对话框还未显示。

编程

句法：	LOAD ... END_LOAD
说明：	装载
参数：	- 无 -

示例

```
LOAD                          ；开始标记
 屏幕窗口 1.Hd =  $85111      ；对话框标题来自文本文件
  VAR1.Min = 0                ；指定变量的最小极限值
  VAR1.Max = 1000             ；指定变量的最大极限值
END_LOAD                      ；结束标记
```

参见

线、分割线、矩形、圆形和椭圆形 (页 219)

7.2.8 UNLOAD

说明

在卸载对话框之前，运行 UNLOAD 方法。

编程

句法：	UNLOAD ... END_UNLOAD
说明：	卸载
参数：	- 无 -

示例

```
UNLOAD
  REG[1] = VAR1              ; 保存寄存器的变量
END_UNLOAD
```

7.2.9 OUTPUT

说明

当调用功能 "GC" 时，运行 OUTPUT 方法。在 OUTPUT 方法中变量和辅助变量作为 NC 代码设计。代码行各个单元的链接用一个空格符实现。

说明

NC 代码可以用文件功能在一个额外的文件中生成并移向 NC。

编程

句法：	OUTPUT(命名符) ... END_OUTPUT	
说明：	输出 NC 程序中的变量	
参数：	名称	OUTPUT 方法的名称

程序段号码和隐藏标记

当要继续保留编程支持激活时在零件程序中直接设置的行号和反编译时的隐藏标记时，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影响下列行为：

条件	行为
程序段数目保持不变。	程序段号码继续保留。
程序段数目变小。	删除最大的程序段号码。
程序段数目变大。	新的程序段没有程序段号码。

示例

```
OUTPUT(CODE1)
  "CYCLE82(" Var1.val "," Var2.val "," Var3.val ","Var4.val "," Var5.val "," Var6.val ")"
END_OUTPUT
```

7.2.10 PRESS

说明

当已按下相应的软键时，运行 PRESS 方法。

编程

句法：	PRESS(软键) ... END_PRESS
名称：	按下软键

参数：	软键	软键名称：HS1 - HS8 和 VS1 - VS8	
	回调	按键<RECALL>（回调）	
	ENTER	按键 <ENTER>，参见 PRESS(ENTER) (页 153)	
	转换	按键 <TOGGLE>，参见 PRESS(TOGGLE) (页 154)	
	PU	Page Up	向前翻页
	PD	Page Down	向后翻页
	SL	Scroll Left	光标向左
	SR	Scroll Right	光标向右
	SU	Scroll Up	光标向上
	SD	Scroll Down	光标向下

示例

```
HS1 = ("其他软键栏")
HS2 = ("没有功能")
PRESS (HS1)
  LS("软键栏 1")                    ；载入其他软键栏
  Var2 = Var3 + Var1
END_PRESS
PRESS (HS2)
END_PRESS
PRESS(PU)
  INDEX = INDEX -7
  CALL("UP1")
END_PRESS
```

7.2.11 PRESS(ENTER)

说明

对于带有输入/输出栏且输入模式为 WR3 或 WR5 的变量，当按下回车(Enter)键时，就会调用 PRESS(ENTER)方法：

- WR3： 定位到栏的位置并按下回车键
- WR5： 在输入模式下，使用回车键接收值

编程

句法：	PRESS(ENTER) <指令> END_PRESS
说明：	回车键已按下
参数：	- 无 -

7.2.12 PRESS(TOGGLE)

说明

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。

需要时，可借助屏幕属性 FOC 来检测哪个变量当前处于焦点下。

编程

句法：	PRESS(TOGGLE) <指令> END_PRESS
说明：	转换键已按下
参数：	- 无 -

示例

```
PRESS(TOGGLE)
   DLGL("Toggle key pressed at variable " << FOC)        ; 借助屏幕属性 FOC 检测哪个变量当前处于焦点下
END_PRESS
```

7.2.13 RESOLUTION

说明

当在屏幕打开时更改当前分辨率时，即通常在 TCU 切换时，就会运行 RESOLUTION 方法。

编程

句法：	RESOLUTION <指令> END_RESOLUTION
说明：	分辨率
参数：	- 无 -

参见

变量 S_RESX 和 S_RESY (页 138)

7.2.14 RESUME

说明

当屏幕（例如）被区域切换中断并要重新显示时，就会调用 RESUME 方法。此时会重新处理数值变化，必要时启动计时器并执行 RESUME 块。

说明

在 SUSPEND 或 RESUME 方法中不允许配置 LS、LM、DLGL、EXIT 和 EXITLS 功能。如果在这些方法中使用这些功能，则会对功能的执行造成阻碍。

编程

句法：	RESUME <指令> END_RESUME
说明：	屏幕重新激活
参数：	- 无 -

7.2.15 SUSPEND

说明

当中断屏幕且不卸载时，就会调用 SUSPEND 方法。例如，当因为简单的区域切换而离开了屏幕，但不显式卸载该屏幕时。屏幕仍保留在后台，但此时不处理数值变化和可能存在的计时器。屏幕暂时中止。

说明

在 SUSPEND 或 RESUME 方法中不允许配置 LS、LM、DLGL、EXIT 和 EXITLS 功能。如果在这些方法中使用这些功能，则会对功能的执行造成阻碍。

编程

句法：	SUSPEND <指令> END_SUSPEND
说明：	屏幕中断
参数：	- 无 -

示例

```
SUSPEND
   MyVar1 = MyVar1 + 1
END_SUSPEND
```

7.2.16 示例： 带 OUTPUT 方法的版本管理

概述

已有的对话框可以通过扩展补充附加的变量。 附加的变量在定义中变量名后的圆括号内有一个版本识别号： （0 = 原始，未写入），1 = 版本 1，
2 = 版本 2， ...

举例：

```
DEF var100=(R//1)                    ; 原始，相当于版本 0
DEF var101(1)=(S//"Hallo")           ; 补充，从版本 1 起
```

在写入 OUTPUT 方法时可以参考某个版本状态，与定义的总体性有关。

举例：

```
OUTPUT(NC1)                          ; 在 OUTPUT 方法中仅提供原始变量。
OUTPUT(NC1,1)                        ; 在 OUTPUT 方法中提供原始变量和带有版本标识符 1 的补充变量。
```

原始版的 OUTPUT 方法不需要版本标识符，然而也可以记为 0。 OUTPUT(NC1) 相当于 OUTPUT(NC1,0)。 OUTPUT 方法中的版本标识符 n 包括所有变量，从原始 0、1、2、... 直至 n。

编程版本标识

```
//M(XXX)                             版本 0 (默认)
DEF var100=(R//1)
DEF var101=(S//"Hallo")
DEF TMP
VS8=("GC")
PRESS(VS8)
GC("NC1")
END_PRESS

OUTPUT(NC1)
var100",,"var101
END_OUTPUT

; ************ 版本 1，补充的定义 ***************
//M(XXX)
```

```
DEF var100=(R//1)
DEF var101=(S//"Hallo")
DEF var102(1)=(V//"HUGO")
DEF TMP
VS8=("GC")
PRESS(VS8)
GC("NC1")
END_PRESS
...

OUTPUT(NC1)                          ; 原始和其它新的版本
var100","var101
END_OUTPUT
...

OUTPUT(NC1,1)                        版本 1
var100","var101"," var102
END_OUTPUT
```

7.3 功能

概述

在对话框和与对话框相关的软键栏中提供不同的功能，这些功能通过事件（例如：退出输入栏，按下软键）触发并在方法中设计。

子程序

重复的或者其它的设计指令，这些指令总结为一个特定的过程，可以在子程序中设计。 子程序可以随时装载到主程序或者其它子程序中并随时进行编辑，即指令不必多次重复设计。作为主程序适用于对话框描述块或者软键栏。

外部功能

通过外部功能可以引入其它一些针对用户的功能。 外部功能存放在一个 DLL 文件中并通过设计文件定义行中的条目识别。

PI 服务

通过功能 PI_START 可以在由 PLC 在 NC 区中启动 PI 服务（程序实例服务）。

另见

功能 (FCT) (页 181)

PI 服务 (页 196)

7.3.1 读写驱动参数： RDOP, WDOP, MRDOP

说明

使用功能 RDOP、WDOP 和 MRDOP 可以读写驱动参数。

说明

读取驱动参数时不要短于 1 秒的周期，时间长些更佳。
原因：否则与驱动的通讯可能会受到很大干扰，甚至导致故障。

说明

如果在读写驱动参数时出错，则会对屏幕属性 ERR 进行相应的设置。

编程

句法：	**RDOP**（“驱动对象的名称”，“参数号”）	
说明：	读驱动参数(Drive Object Parameter)	
参数：	驱动对象的名称	驱动对象的名称可参考“诊断”操作区（> ETC > HSK 8：驱动系统）“驱动系统诊断”中的“DO Name”一列（见下图）。
	参数号	

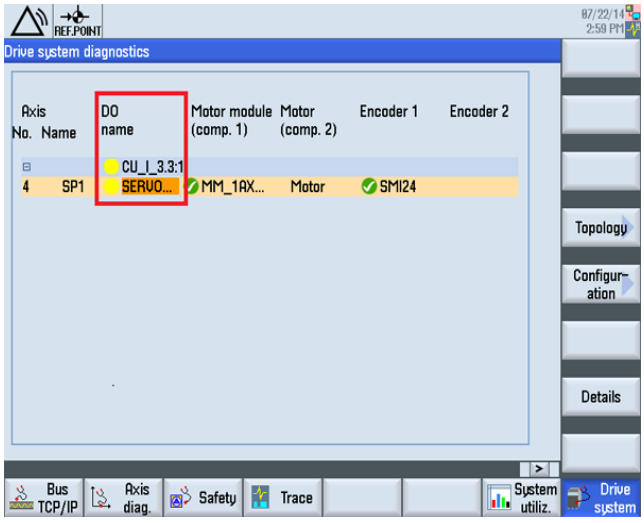


图 7-1　驱动对象的名称

句法：	**MRDOP**（“驱动对象的名称”，“参数号 1”*“参数号” [*...]，寄存器索引）	
说明：	多次读取驱动参数。 使用命令 MRDOP 可以通过一次存取向寄存器中传输一个驱动对象的多个驱动参数。这种存取比通过单个存取读取速度明显加快。	
参数：	驱动对象的名称	驱动对象的名称可从（例如）“调试”操作区域的基本画面中获取
	参数号 1 ... n	变量名称用“*”作为分隔符。按照命令中变量名称的顺序将值接收到寄存器 REG[寄存器索引]中。
	寄存器索引	第一个变量的值在 REG[寄存器索引] 中。 第二个变量的值在 REG[寄存器索引 + 1] 中。

句法：	**WDOP**（“驱动对象的名称”，“参数号”，“值”）	
说明：	写驱动参数(Drive Object Parameter)	
参数：	驱动对象的名称	驱动对象的名称可从（例如）“调试”操作区域的基本画面中获取
	参数号	参数号
	值	待写入的值

示例

读取驱动对象“SERVO_3.3:2”的电机温度 r0035：

```
MyVar=RDOP("SERVO_3.3:2","35") ;
```

读取驱动对象“SERVO_3.3:2”的电机温度 r0035 和转矩实际值 r0080 并从寄存器索引 10 开始保存各个结果：

```
MRDOP("SERVO_3.3:2","35*80",10)
```

7.3.2 子程序调用(CALL)

说明

通过 CALL 功能可以从方法的任意一个位置调用一个装载的子程序。允许叠加，即由一个子程序调用另一个子程序。

编程

句法：	**CALL**("名称")	
说明：	调用子程序	
参数：	名称	子程序名称

示例

//M(屏幕窗口 1)

```
DEF VAR1 = ...
DEF VAR2 = ...
CHANGE(VAR1)
  ...
  CALL("MY_UP1")              ; 调用并编辑子程序
  ...
END_CHANGE

CHANGE(VAR2)
  ...
  CALL("MY_UP1")              ; 调用并编辑子程序
  ...
END_CHANGE

SUB(MY_UP1)
                              ;do something
END_SUB
//END
```

7.3.3 定义块(//B)

说明

子程序在程序文件中用块标记 //B 标记并通过 //END 结束。每个块标记可以定义多个子程序。

说明

必须在调用子程序的对话框中定义子程序所使用的变量。

编程

一个块有下列结构：

句法：	**//B**(块名称) **SUB**(名称) END_SUB [SUB(名称) ... END_SUB] ... //END	
说明：	定义子程序	
参数：	块名称	块标记名称
	名称	子程序名称

示例

```
//B(PROG1)                          ; 块开始
SUB(UP1)                            ; 子程序开始
  ...
  REG[0] = 5                        ; 寄存器 0 赋值 5
  ...
END_SUB                             ; 子程序结束
SUB(UP2)                            ; 子程序开始
  IF VAR1.val=="Otto"
    VAR1.val="Hans"
```

```
    RETURN
  ENDIF
  VAR1.val="Otto"
END_SUB                              ; 子程序结束
//END                                ; 块结束
```

7.3.4 检查变量（CVAR）

说明

借助于功能 CVAR（检查变量）可以查询对话框的所有变量或者仅特定变量或者辅助变量是否正确。

在用功能 GC 产生一个 NC 代码之前，查询变量是否包含一个有效值是必要的。

如果变量状态命名符.vld = 1，则变量正确，

编程

句法：	**CVAR**(VarN)	
说明：	检查变量的有效内容	
参数：	VarN	要检查的变量列表。 可以最多有 29 个变量，各自之间通过逗号隔开。最大字符长度为 500。 查询结果可能是：
	1 =	TRUE（真）（所有变量都有有效内容）
	0 =	FALSE（假）（至少一个变量没有有效内容）

示例

```
IF CVAR == TRUE                      ; 检查所有变量
  VS8.SE = 1                         ; 如果所有变量正确，则软键 VS8 可见
ELSE
  VS8.SE = 2                         ; 如果变量包含错误值，则软键 VS8 不可操作
ENDIF

IF CVAR("VAR1", "VAR2") == TRUE
```

```
                                                    ; 检查变量 VAR1 和 VAR2
  DLGL ("VAR1 和 VAR2 正确")
                                                    ; 如果 VAR1 和 VAR2 无错误，则显示对话框行"VAR1
                                                    和 VAR2 正确"
ELSE
    DLGL ("VAR1 和 VAR2 不正确")
                                                    ; 如果 VAR1 和 VAR2 错误，则显示对话框行"VAR1
                                                    和 VAR2 不正确"
ENDIF
```

7.3.5 CLEAR_BACKGROUND

说明

使用功能 CLEAR_BACKGROUND 可以删除图形元素 LINE、RECT、ELLIPSE、V_SEPARATOR 和 H_SEPARATOR。

参见

线、分割线、矩形、圆形和椭圆形 (页 219)

7.3.6 文件功能"Copy Program (CP，复制程序)"

说明

功能 CP（复制程序）用于在 HMI 文件系统或者 NC 文件系统中复制文件。

编程

句法：	**CP**("源文件", "目标文件")	
说明：	文件：复制	
参数：	源文件	源文件完整的路径数据
	目标文件	目标文件完整的路径数据

可通过返回值（VAR1 被定义为辅助变量）查询功能是否成功执行：

```
CP("//NC/MPF.DIR/HOHO.MPF","//NC/MPF.DIR/ASLAN.MPF",VAR1)
```

示例

带返回值的应用情况：

```
CP("//NC/MPF.DIR/HOHO.MPF","//NC/MPF.DIR/ASLAN.MPF",VAR1)
CP("CF_CARD:/wks.dir/MESS_BILD.WPD/MESS_BILD.MPF","//NC/WKS.DIR/AAA.WPD/HOHO2.MPF",VAR1)
CP("//NC/MPF.DIR/HOHO.MPF","CF_CARD:/wks.dir/WST1.WPD/MESS.MPF",VAR1)        ; WPD 必须存在
```

无返回值的应用情况：

```
CP("//NC/MPF.DIR/HOHO.MPF","//NC/MPF.DIR/ASLAN.MPF")
CP("CF_CARD:/mpf.dir/MYPROG.MPF","//NC/MPF.DIR/HOHO.MPF")
CP("//NC/MPF.DIR/HOHO.MPF","CF_CARD:/XYZ/MYPROG.MPF")                          ; XYZ 必须存在
```

另见

FILE_ERR 的支持：变量 FILE_ERR (页 130)

7.3.7 文件功能“Delete Program (DP，删除程序)”

说明

功能 DP（删除程序）删除一个被动的 HMI 文件系统或者主动的 NC 文件系统的文件。

编程

句法：	**DP**("文件")	
说明：	文件：删除	
参数：	文件	要删除文件的完整路径数据

示例

此功能使用以下句法进行数据管理：

- 带返回值

  ```
  DP("//NC/MPF.DIR/MYPROG.MPF",VAR1)
  DP("//NC/WKS.DIR/TEST.WPD/MYPROG.MPF",VAR1)
  ```

```
DP("//NC/CMA.DIR/MYPROG.SPF",VAR1)
```

VAR1 = 0　　文件已删除。

VAR1 = 1　　文件尚未删除。

- 无返回值：

```
DP("//NC/MPF.DIR/MYPROG.MPF")
DP("//NC/WKS.DIR/TEST.WPD/MYPROG.MPF")
DP("//NC/CMA.DIR/MYPROG.SPF")
```

7.3.8 文件功能“Exist Program (EP，退出程序)”

说明

功能 EP（存在程序）检查 NC 文件系统的特定 NC 程序或者 HMI 文件系统中在规定的路径下面是否存在某个文件。

编程

句法：	**EP**("文件")	
说明：	检查 NC 程序的存在	
参数：	文件	NC 文件系统或者 HMI 文件系统内文件的完整路径数据
返回值：	待分配查询结果的变量的名称。	

功能 EP 使用新的句法和旧的逻辑（调整句法）。

使用符合要求的名称直接响应文件：

```
//NC/MPF.DIR/XYZ.MPF
```

或

`CF_CARD: /MPF.DIR/XYZ.MPF`（指示 /user/sinumerik/data/prog）

或

`LOC:`（对应 CF_CARD）

新句法示例：

```
EP("//NC/WKS.DIR/TEST.WPD/XYZ.MPF",VAR1)
EP("CF_CARD:/mpf.dir/XYZ.MPF",VAR1)
EP("LOC:/mpf.dir/XYZ.MPF",VAR1)
; 带返回值：
; VAR1 = 0 文件存在。
; VAR1 = 1 文件不存在。
```

旧句法示例：

```
EP("/MPF.DIR/CFI.MPF", VAR1)
; 带返回值：
; VAR1 = M 文件位于 HMI 文件系统中。
; VAR1 = N 文件位于 NC 文件系统中。
; VAR1 = B 文件位于 HMI 和 NC 文件系统中。
```

示例

```
EP("/MPF.DIR/GROB.MPF",VAR1)                                    ; 旧句法 - 路径现在采用 /，而不是 \

IF VAR1 == "M"
   DLGL（“文件位于 HMI 文件系统中”）
ELSE
   IF VAR1 == "N"
      DLGL（“文件位于 NC 文件目录中”）
   ELSE
     DLGL（“文件既不在 HMI 文件系统中，也不在 NC 文件系统中”）
   ENDIF
ENDIF
```

7.3.9 文件功能“Move Program (MP，移动程序)”

说明

功能 MP（移动程序）用于在 HMI 文件系统或者 NC 文件系统中复制文件。

编程

句法：	**MP**("源", "目标")	
	MP("CF_CARD:/MPF.DIR/MYPROG.MPF","//NC/MPF.DIR")	
说明：	移动文件	
参数：	源文件	完整路径
	目标文件	完整路径
返回值	查询结果	

示例

带返回值：

```
MP("//NC/MPF.DIR/123.MPF","//NC/MPF.DIR/ASLAN.MPF",VAR1)          ; 位于 NC 内部
MP("//NC/MPF.DIR/123.MPF",VAR0,VAR1)                              ; 通过变量设定目标
MP(VAR4,VAR0,VAR1)                                                ; 通过变量设定源和目标
MP("CF_CARD:/mpf.dir/myprog.mpf","//NC/MPF.DIR/123.MPF",VAR1)     ; 从 CF 卡至 NC
MP("//NC/MPF.DIR/HOHO.MPF","CF_CARD:/XYZ/123.mpf",VAR1)           ; 从 NC 至 CF 卡

; 带返回值：
; 执行 VAR1 = 0
; 未执行 VAR1 = 1
```

7.3.10 文件功能“Select Program (SP，选择程序)”

说明

功能 SP（选择程序）选择一个主动的 NC 文件系统文件，以对其进行处理。即该文件之前必须就已装载在 NC 中。

编程

句法：	**SP**("文件")	
名称：	选择程序	
参数：	“文件”	NC 文件的完整路径数据

示例

此功能使用以下句法进行数据管理：

- 带返回值

  ```
  SP("//NC/MPF.DIR/MYPROG.MPF", VAR1)
  ```

  VAR1 = 0　　文件已装载。

  VAR1 = 1　　文件未装载。
- 无返回值：

  ```
  SP("//NC/MPF.DIR/MYPROG.MPF")
  ```

7.3.11 文件存取： RDFILE、WRFILE、RDLINEFILE、WRLINEFILE

说明

采用 INI 句法对文件进行读写访问时可使用功能 RDFILE 和 WRFILE。

对文件中的各行进行读写访问时可使用功能 RDLINEFILE 和 WRLINEFILE。

编程

句法：	**RDFILE**(“文件名 + 路径”，“段”，“关键字”)	
说明：	从文件中读	
参数：	文件名 + 路径	路径和文件名
	段	INI 文件中的段落
	关键字	INI 文件中的关键字

句法：	**WRFILE**(数值，“文件名 + 路径”，“段”，“关键字”)	
说明：	写入文件	
参数：	Value	待写入的值
	文件名 + 路径	路径和文件名
	段	INI 文件中的段落
	关键字	INI 文件中的关键字

句法：	**RDFILE**(“文件名 + 路径”，行编号)	
说明：	从文件中读取一行	
参数：	文件名 + 路径	路径和文件名
	行号	行号 第一行从 0 开始编号。

句法：	**WRLINEFILE**(数值，“文件名 + 路径”)	
说明：	在文件末尾写一行	
参数：	值	待写入的值
	文件名 + 路径	路径和文件名

说明

- 文件不得位于 NC 文件系统（数据维护）中。
- 如果文件不存在或者到达了文件末尾又或者是出现了其他错误，都会相应地设置变量 FILE_ERR 和 ERR。您可事先使用文件功能 Exist Program (EP)检查一个文件是否存在。
- 处理过的文件会以“UTF-8”编码（无 BOM（Byte Order Mask））。读取的文件要求以“UTF-8”编码。
- 通过文件功能 Delete Program (DP)可在需要时将文件显式删除。

示例

从 INI 文件中读：

前提条件/假设：

文件 “C:/tmp/myfile.ini” 的内容：

```
<...>
[MyData]

MyName=Daniel

<...>


MyVar = RDFILE("C:/tmp/myfile.ini", "MyData", "MyName")
```

结果：

MyVar 现在包含值 “Daniel”。

写入 INI 文件：

前提条件/假设：

VARS=12

```
WRFILE(VARS, "C:/tmp/myfile.ini", "MySession", "NrOfSessions")
```

结果：

文件 “C:/tmp/myfile.ini” 的内容：

<...>
[MySession]

NrOfSessions=12

<...>

读取文件的第 4 行：

前提条件/假设：

c:/tmp/myfile.mpf:
R[0]=0 的内容

```
F3500 G1

MYLOOPIDX1:X0 y-50 Z0

  X150 Y50 Z10

  R[0]=R[0]+1

GOTOB MYLOOPIDX1

M30
```

```
MyVar = RDLINEFILE("C:/tmp/myfile.mpf", 4)
```

结果：

MyVar 现在包含值 “R[0]=R[0]+1”

在文件末尾写入：

前提条件/假设：

```
VARX=123
VARY=456

WRLINEFILE("F100 X" << VARX << " Y" << VARY, "C:/tmp/mypp.mpf")
```

结果：

c:/tmp/mypp.mpf 的内容：

```
<...>
F100 X123 Y456
```

7.3.12 对话框行(DLGL)

说明

在对话框的对话框行中可以根据确定的情况给出短文本（信息或者输入帮助）。

标准字体大小时允许的字符数量：约 50 个字符

说明

在 SUSPEND 或 RESUME 方法中不允许配置 LS、LM、DLGL、EXIT 和 EXITLS 功能。如果在这些方法中使用这些功能，则会对功能的执行造成阻碍。

编程

句法：	**DLGL**("字符串")	
说明：	显示对话框行的文本	
参数：	字符串	显示在对话框中的文本

示例

```
IF Var1 > Var2
```

```
                                    ；如果变量 1>变量 2，则对话框行中显示文本“值过大！”。
  DLGL("值过大！")
ENDIF
```

7.3.13 DEBUG

说明

通过功能 DEBUG 可在 Run MyScreen 用户屏幕的设计阶段为您提供分析帮助。采用功能 DEBUG 可对括号中返回的运行时间的表述进行检测。结果会在日志文件“easyscreen_log.txt”中作为独立条目加以记录。

在每条记录前会附加一个带有方括号的当前时间戳（参见下例）。

建议在对时间要求苛刻的部件上根据情况将 DEBUG 输出文件删除。建议在设计阶段结束后为 DEBUG 输出文件加上注释。对不断增多的日志文件 “easyscreen_log.txt” 的写访问可能会导致速度变慢。

编程

句法：	**DEBUG**(表达式)
说明：	在日志文件“easyscreen_log.txt”中进行记录
参数：	待检测的表述，由此在日志文件中生成一条记录

示例

```
IF Var1 > Var2
  DEBUG("Value of ""Var1"": " << Var1)
                                        ; “easyscreen_log_txt” 中的记录：
                                        [10:22:40.445] DEBUG:Value of "Var1":123.456
ENDIF
```

7.3.14 退出对话框(EXIT)

说明

通过功能 EXIT 可以退出对话框并返回主对话框。如果不存在主对话框，则退出新配置的操作界面并返回标准应用程序。

说明

在 SUSPEND 或 RESUME 方法中不允许配置 LS、LM、DLGL、EXIT 和 EXITLS 功能。如果在这些方法中使用这些功能，则会对功能的执行造成阻碍。

编程（没有参数）

句法：	**EXIT**
说明：	退出对话框
参数：	- 无 -

示例

```
PRESS(HS1)
  EXIT
END_PRESS
```

说明

如果调用带传输变量的当前对话框，则变量值改变并且返回初始对话框。

变量值分别分配给通过功能"LM"从初始对话框传输到后续对话框的变量。可以最多传输 20 个变量值，各自之间通过逗号隔开。

说明

变量/变量值的顺序必须根据 LM 功能的传递变量顺序进行，以此达到分配明确。如果一些变量值没有规定，则这些变量不改变。改变的传输变量在功能 LM 后立即在初始对话框中生效。

使用传输变量编程

句法：	**EXIT**[(VARx)]	
说明：	退出对话框，传输一个或者多个变量	
参数：	VARx	计算变量

示例

```
//M(屏幕窗口 1)
...
PRESS(HS1)
  LM("屏幕 2","CFI.COM",1, POSX, POSY,
DURCHMESSER)
                                        ；中断屏幕 1 并显示屏幕 2。传输变量 POSX、POSY
                                        和直径。
  DLGL(“退出屏幕 2”)                    ；从屏幕 2 返回后在屏幕 1 的对话框行中显示文本：退
                                        出屏幕 2。
END_PRESS
...
//END

//M(屏幕 2)
...
PRESS(HS1)
  EXIT(5, , 计算_直径)
                                        ；退出屏幕 2 并返回到屏幕 1 的 LM 后的行中。此时，
                                        赋值 5 给变量 POSX 并分配变量“计算_直径”的值给
                                        变量“直径”。变量 POSY 包含当前值。
END_PRESS
...
//END
```

7.3.15 转换栏或列表框栏的动态列表操作

说明

功能 LISTADDITEM、LISTINSERTITEM、LISTDELETEITEM 和 LISTCLEAR 用于对转换栏或列表框栏进行动态列表操作。

这些功能只对本身具有列表的变量有效，例如

- “基本”列表
  `DEF VAR_AC1 = (I/* 0,1,2,3,4,5,6,7,8)` 或
- “扩展”列表
  `DEF VAR_AC2 = (I/* 0="AC0", 1="AC1", 2="AC2", 3="AC3", 4="AC4", 5="AC5", 6="AC6", 7="AC7", 8="AC8").`

当变量指向一个数组时，例如 `DEF VAR_AC3 = (I/* MYARRAY)`，该功能无法使用，否则全局数组会发生变化。

一个变量应至少在 DEF 行中定义了一个值。这样可确定为“基本”或“扩展”型列表。

此后仍允许将列表完全删除以及全部重建。但类型“基本”或“扩展”必须保留或无法动态更改。

编程

句法：	**LISTINSERTITEM**(变量名，位置，ItemValue[, ItemDispValue])	
说明：	在某个特定位置添加元素	
参数：	变量名	
	位置	列表中需要添加元素的位置
	ItemValue	列表条目的值
	ItemDispValue	应在列表中显示的值

句法：	**LISTADDITEM**(变量名，ItemValue[, ItemDispValue])	
说明：	在列表末尾附加一个元素	
参数：	变量名	
	ItemValue	列表条目的值
	ItemDispValue	应在列表中显示的值

句法：	**LISTDELETEITEM**(变量名，位置)
说明：	删除某个元素

参数：	变量名	
	位置	列表中要删除元素的位置

句法：	**LISTCOUNT**（变量名）	
说明：	提供当前列表元素数	
参数：	变量名	

句法：	**LISTCLEAR**(变量名)	
说明：	删除整个列表	
参数：	变量名	

示例

说明

以下几个示例彼此互为基础，它们的顺序对各个结果的理解起着重要作用。

前提条件/假设：

```
DEF VAR_AC = (I/* 0="Off",1="On"/1/,"Switch"/WR2)
```

在列表末尾附加一个元素 -1="Undefined"：

```
LISTADDITEM("VAR_AC", -1, """Undefined""")
```

结果：0="Off", 1="On", -1="Undefined"

在位置 2 上添加一个元素 99="Maybe"：

```
LISTINSERTITEM("VAR_AC", 2, 99, """Maybe""")
```

结果：0="Off", 1="On", 99="Maybe", -1="Undefined"

确定当前列表元素数：

```
REG[10]=LISTCOUNT("VAR_AC")
```

结果：REG[10] = 4

删除位置 1 上的元素：

```
LISTDELETEITEM("VAR_AC", 1)
```

结果：0="Off", 99="Maybe", -1="Undefined"

删除整个列表：

```
LISTCLEAR("VAR_AC")
```

结果：列表为空

7.3.16 评估(EVAL)

说明

功能 EVAL 评估作出的输出结果，然后执行。为此可以首先在运行期间建立表达式。例如用于变量上的显示存取。

编程

句法：	**EVAL**(exp)	
说明：	评估表达式	
参数：	exp	逻辑表达式

示例

```
VAR1=(S)
VAR2=(S)
VAR3=(S)
VAR4=(S)
```

```
CHANGE()
  REG[7] = EVAL("VAR"<<REG[5])  ；如果 REG[5] 的值为 3， 则括号中的表达式为 VAR3。然后，分配
                                REG[7] VAR3 的值。
  IF REG[5] == 1
    REG[7] = VAR1
  ELSE
    IF REG[5] == 2
      REG[7] = VAR2
    ELSE
      IF REG[5] == 3
        REG[7] = VAR3
      ELSE
        IF REG[5] == 4
          REG[7] = VAR4
        ENDIF
      ENDIF
    ENDIF
  ENDIF
END_CHANGE
```

7.3.17 退出装载软键(EXITLS)

说明

通过功能 EXITLS 可以离开当前的操作界面并装载一个定义的软键栏。

说明

在 SUSPEND 或 RESUME 方法中不允许配置 LS、LM、DLGL、EXIT 和 EXITLS 功能。如果在这些方法中使用这些功能，则会对功能的执行造成阻碍。

编程

句法：	**EXITLS**("软键栏"[, "路径名"])	
说明：	退出时装载软键栏	
参数：	软键栏	待装载的软键栏名称
	路径名	要装载的软键栏目录路径

示例

```
PRESS(HS1)
    EXITLS( "软键栏 1", "AEDITOR.COM" )
END_PRESS
```

7.3.18 功能 (FCT)

说明

外部功能存放在一个 DLL 文件中，并通过配置文件定义行中的条目识别。

说明

一个外部功能必须至少有一个返回参数。

编程

句法：	**FCT** 功能名称 =（“文件”/返回类型/固定参数的类型/可变参数的类型）	
	FCT InitConnection = ("c:\tmp\xyz.dll"/I/R,I,S/I,S)	
说明：	例如可从 LOAD 方法中或在 PRESS 方法中执行外部功能的调用。	
参数：	功能名称	外部功能的名称
	文件	DLL 文件的完整路径数据
	返回类型	返回值的数据类型
	固定参数类型	值参数
	可变参数类型	参考参数
	数据类型用逗号隔开。	

例如可从 LOAD 方法中或在 PRESS 方法中执行外部功能的调用。

示例：

```
press(vs4)
RET = InitConnection(VAR1,13,"Servus",VAR2,VAR17)
end_press
```

外部功能的结构

外部功能必须遵循特定的符号：

句法：	extern "C" dllexport void InitConnection (ExtFctStructPtr FctRet, ExtFctStructPtr FctPar, char cNrFctPar)	
说明：	仅在 Windows 环境下执行 DLL 导出 区分符和传输参数为固定设定。通过传输的结构传送原先的调用参数。	
参数：	cNrFctPar	调用参数的数量 = FctPar 中结构单元的数量
	FctPar	指示某结构单元栏，其包含相应的调用参数（含数据类型）。
	FctRet	指示某结构用于返回功能值（含数据类型）。

传输结构定义

```
union CFI_VARIANT
    (
    char                                b;
    short int                           i;
    double                              r;
    char*                               s;
    )
typedef struct ExtFctStructTag
    (
    char                                cTyp;
    union CFI_VARIANT                   value;
    )ExtFctStruct;
typedef struct ExtFct* ExtFctStructPtr;
```

如需独立于操作平台（Windows、Linux）开发外部功能，则不可使用关键字 __declspec(dllexport)。只在 Windows 下需要此关键字。例如在 Qt 下可使用以下宏：

```
#ifdef Q_WS_WIN
       #define MY_EXPORT __declspec(dllexport)
#else
       #define MY_EXPORT
#endif
```

函数声明如下：

```
extern "C" MY_EXPORT void InitConnection
(ExtFctStructPtr FctRet, ExtFctStructPtr FctPar,char cNrFctPar)
```

如需将通过 Run MyScreens 设计的画面用于 NCU 和 PCU/PC，则必须略去二进制文件的扩展名：

```
FCT InitConnection = ("xyz"/I/R,I,S/I,S)
```

在省略了绝对路径信息时，Run MyScreens 会首先在项目索引中搜索二进制文件。

7.3.19 生成代码(GC)

说明

功能 GC（Generate Code）从 OUTPUT 方法中生成 NC 代码。

编程

句法：	**GC**("名称"[,"目标文件"][,Opt],[Append])	
说明：	生成 NC 代码（功能 GC 仅在 NC 中可用）	
参数：	名称	OUTPUT 方法名称，作为代码生成的基础
	目标文件	HMI 或者 NC 文件系统的目标文件路径数据。 如果未规定目标文件（只能在编程支持中），则在该位置上写入代码，在该位置处光标停留在当前打开的文件中。
	可选	可选注释生成
	0:	（缺省设置）生成带有反编译注释的代码（另见 反编译 (页 202)）。
	1:	在生成的代码中不生成注释。 **提示：**该代码不能反编译（另见反编译 (页 202)）。
	Append	只有当规定目标文件情况下，该参数才有意义。
	0:	（预设）当文件已存在情况下，删除旧的内容。
	1:	当文件已存在情况下，在文件开始处写入新的代码。
	2:	在文件已存在的情况下，在结束处加入新的代码。

示例

```
//M(测试 GC/"代码生成:")
DEF VAR1 = (R//1)
DEF VAR2 = (R//2)
DEF D_NAME
LOAD
  VAR1 = 123
  VAR2 = -6
END_LOAD
OUTPUT(CODE1)
  "Cycle123(" VAR1 "," VAR2 ")"
  "M30"
END_OUTPUT

PRESS(VS1)
  D_NAME = "\MPF.DIR\MESSEN.MPF"
  GC("CODE1",D_NAME)                  ; 将 OUTPUT 方法的 NC 代码写入文件 \MPF.DIR
                                      \MESSEN.MPF 中:
                                      Cycle123(123, -6)
                                      M30
END_PRESS
```

保存目标文件

功能 GC 仅在 NC 中可用。请使用功能 CP 或 MP 来移动文件。为此可参见以下示例：

```
; Windows 目录
  GC("Code","//NC/MPF.DIR/NC1.MPF")
  MP("//NC/MPF.DIR/NC1.MPF","d:/tmp/WIN1.MPF")

; LOCAL_DRIVE
  GC("Code","//NC/MPF.DIR/NC1.MPF")
  MP("//NC/MPF.DIR/NC1.MPF","LOCAL_DRIVE:/WIN_LD1.MPF")
```

反编译

- **目标文件没有数据：**
  功能 GC 只能在编程支持中使用并待当前编辑器中打开的文件内写入 NC 代码。NC 代码可以反编译。如果功能 GC 在没有目标文件数据的情况下在"Run MyScreens"下设计，则此时发出故障信息。
- **目标文件数据：**
  从 OUTPUT 方法中生成的代码输入到目标文件中。如果不存在目标文件，则在 NC 文件系统中设立。如果目标文件在 HMI 文件系统中，则该文件存放在硬盘上。不设立使用注释行（反编译需要的信息），即无法进行反编译。

反编译时的特殊情况

在子对话框中无法调用 GC 功能，因为在子对话框中可以使用源自主对话框的变量，然而在直接调用时并不存在。

在通过编辑器手动介入生成的代码时不允许改变由代码生成产生的值的符号数目。这些改变会妨碍到反编译。

解决方法：

1. 反编译
2. 通过设计的对话框输入修改（例如：99 → 101）
3. GC

7.3.20 口令功能

概述

提供以下可用的口令功能：

- 设置口令
- 删除口令
- 更改口令

HMI_LOGIN 功能：设置新口令

句法：	**HMI_LOGIN**(Passwd)	
说明：	通过功能 HMI_LOGIN() 可发送一个口令到 NCK，通过该口令设置当前的存取等级。	
参数：	Passwd	口令

示例

```
REG[0] = HMI_LOGIN("CUSTOMER")          ; 口令成功设置时，结果 = TRUE，否则为 FALSE
```

HMI_LOGOFF 功能：删除口令

句法：	**HMI_LOGOFF**
说明：	通过功能 HMI_LOGOFF 可复位当前的存取等级。
参数：	- 无 -

示例

```
REG[0] = HMI_LOGOFF                     ; 口令成功删除时，结果 = TRUE，否则为 FALSE
```

HMI_SETPASSWD 功能：更改口令

句法：	**HMI_SETPASSWD**(AC, Passwd)	
说明：	通过功能 HMI_SETPASSWD 可对当前口令级或更低口令级的口令进行更改。	
参数：	AC	访问等级
	Passwd	口令

示例

```
REG[0] = HMI_SETPASSWD(4,"MYPWD")       ; 口令成功更改时，结果 = TRUE，否则为 FALSE
```

7.3.21 装载数组(LA)

说明

通过功能 LA（装载数组）可以从另一个文件装载一个数组。

编程

句法：	**LA**(名称 [, 文件])	
说明：	从文件装载数组	
参数：	名称	待装载数组的名称
	文件	在该文件中定义数组

说明

如果在当前的配置文件中，一个数组要由另一个配置文件中的数组替代，则数组名称必须相同。

示例

```
                                          ; 文件 maske.com 部分摘录
DEF VAR2 = (S/*ARR5/"关闭", "转换栏")
PRESS(HS5)
  LA("ARR5","arrayext.com")               ; 从文件 arrayext.com 中
                                          装载数组 ARR5
  VAR2 = ARR5[0]                          ; 代替 "Aus"/"Ein"（"关"/"开"）显示在 VAR2 的
                                          转换栏
                                          "上"/"下"/"右"/"左"
END_PRESS
//A(ARR5)
("关"/"开")
//END
                                          ; 文件 arrayext.com 部分摘录
//A(ARR5)
("上"/"下"/"右"/"左")
//END
```

说明

请注意，在通过 LA 功能分配给变量转换栏另一个数组后，必须赋予一个有效值给变量。

7.3.22 装载块（LB）

说明

通过功能 LB（装载块）可以在运行期间内装载带有子程序的块。首先 LB 要在 LOAD（装载）方法中设计，由此可以随时调用装载的子程序。

说明

子程序也可以直接在对话框中定义，如果那样则不必装载。

编程

句法：	**LB**("块名称"[,"文件"])	
说明：	运行时装载子程序	
参数：	块名称	块标记名称
	文件	配置文件的路径数据 预设 = 当前的配置文件

示例

```
LOAD
  LB("PROG1")              ；在当前的配置文件中查找块“PROG1”，接着装载块。
  LB("PROG2","XY.COM")     ；在配置文件 XY.COM 中查找块“PROG2”，接着装载块。
END_LOAD
```

7.3.23 装载屏幕窗口 (LM)

说明

通过功能 LM 可以载入一个新的对话框。取决于对话框切换的模式（见下表），还可采用该功能实现信息对话框。

说明

在 SUSPEND 或 RESUME 方法中不允许配置 LS、LM、DLGL、EXIT 和 EXITLS 功能。如果在这些方法中使用这些功能，则会对功能的执行造成阻碍。

主对话框/子对话框

能够调用其他对话框并且不能自行结束的对话框称为主对话框。由主对话框调用的对话框是子对话框。

编程

句法：	**LM**("名称"[,"文件"] [,MSx [, VARx]])
说明：	载入对话框

参数：	名称	待载入对话框的名称
	文件	配置文件的路径数据（HMI 文件系统或者 NC 文件系统）；缺省设置：当前的配置文件
	MSx	对话框切换模式
	0:	（缺省设置）退出当前对话框，装载并显示新的对话框。在 EXIT 时（参见“退出”功能）返回到标准应用程序。通过参数 MSx 可以确定在对话框切换时是否退出当前对话框。保留当前的对话框，可以将变量传递到新的对话框中。 使用参数 MSx 的优点在于，在切换时无需总是重新初始化对话框，而是保当前对话框的数据和设计，简化数据传输
	1:	自功能 LM 起中断当前对话框，装载并显示新的对话框（例如用于实现信息对话框）。退出时关闭子对话框并且返回到主对话框的中断位置。 中断时，不处理主对话框中的 UNLOAD（卸载）块。
	VARx	前提条件：MS1 列出可以从主对话框传送到子对话框的变量。可以最多传输 20 个变量，各自之间通过逗号隔开。

说明

参数 VARx 总是只传递变量值，也就是说变量可以在子对话框中读写，但是不可见。由子对话框向主对话框返回传递变量可以通过功能 EXIT（退出）进行。

示例

屏幕 1（主对话框）：跳至子对话框屏幕 2（带变量传输）

```
PRESS(HS1)

  LM("屏幕 2","CFI.COM",1, POSX, POSY, DURCHMESSER)
                                    ；中断屏幕 1 并显示屏幕 2：传输变量 POSX、POSY 和直径。
  DLGL(“退出屏幕 2”)                ；从屏幕 2 返回后在屏幕 1 的对话框行中显示文本：退出屏幕
                                    2。
END_PRESS
```

屏幕 2（子对话框）：返回主对话框屏幕 1（带变量传输）

```
PRESS(VS8)

  EXIT(POSX, POSY, DURCHMESSER)
                                        ；隐藏屏幕 2 并继续显示屏幕 1：传输修改的变量 POSX、
                                        POSY 和直径。
END_PRESS
```

7.3.24 装载软键(LS)

说明

通过功能 LS 可以显示另一个软键栏。

说明

在 SUSPEND 或 RESUME 方法中不允许配置 LS、LM、DLGL、EXIT 和 EXITLS 功能。如果在这些方法中使用这些功能，则会对功能的执行造成阻碍。

编程

句法：	**LS**("名称"[, "文件"][, 合并])	
说明：	显示软键栏	
参数：	名称	软键栏名称
	文件	配置文件的路径数据（HMI 文件系统或者 NC 文件系统） 预设置：当前的配置文件
	Merge	
	0:	删除所有存在的软键，输入重新设计的软键。
	1:	预设置 仅重新设计的软键覆盖现有的软键，其他软键（＝HMI 应用程序的软键）的功能和文本保持不变。

示例

```
PRESS(HS4)
  LS("软键栏 2",,0)     ；软键栏 2 覆盖现有的软键栏，删除显示的软键。
END_PRESS
```

说明

只要编译器还没有显示出对话框，即还没有处理 LM 功能，则在登入软键描述块和软键栏描述块的 PRESS（按压）方法中总是仅设计一个 LS 命令或者 LM 命令，并且没有其它动作。

功能 LS 和 LM 仅允许在软键 PRESS 方法中调用，然而不作为导航键（PU、PD、SL、SR、SU、SD）上的反应。

7.3.25 Load Grid (LG)

说明

表格描述(Grid)可以通过 LG 方法在 LOAD 方法内动态提供。

为此可以通过 LG 方法指定一个表格，必须已将变量定义为 Grid 变量并引用已存在的有效表格。

编程

句法：	**LG**（栅格名称，变量名称 [,文件名称]）	
说明：	载入表格	
参数：	栅格名称	表格（栅格）名称，用双引号括起
	变量名	应指定表格的变量名称，用双引号括起
	文件名	文件名称，在该文件中定义表格（栅格），用双引号括起如果表格未在文件中定义，在该文件中也定义变量，则必须指定。

示例

```
//M(MyGridSample/"My Grid Sample")
DEF MyGridVar=(R/% MyGrid1///WR2////100,,351,100)
```

```
LOAD
   LG("MyGrid1","MyGridVar","mygrids.com")
END_LOAD
```

mygrids.com 的内容：

```
//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
```

7.3.26 多次选择 SWITCH

说明

使用 SWITCH 命令可以对变量的不同数值进行检验。

在 SWITCH 命令中设置的表达式会与在接下来的 CASE 命令中设置的所有数值进行比较。

如不相同，则会继续和之后的 CASE 命令进行比较。 如果之后有 DEFAULT 命令并且之前一直没有与在 SWITCH 命令中设置的表达式相同的 CASE 命令，则会执行 DEFAULT 命令之后的指令。

如果相同，则会执行接下来的指令，直到出现 CASE、DEFAULT 或 END_SWITCH 命令为止。

示例

```
FOCUS
   SWITCH (FOC)
      CASE "VarF"
         DLGL("Variable ""VarF"" has the input focus.")
      CASE "VarZ"
         DLGL("Variable ""VarZ"" has the input focus.")
      DEFAULT
         DLGL("Any other variable has the input focus.")
   END_SWITCH
```

```
END_FOCUS
```

7.3.27 多次读取 NC PLC (MRNP)

说明

用命令 MRNP 可以通过在寄存器中的一次存取输入多个 NC/PLC 变量。这种存取比通过单个存取读取速度明显加快。NC/PLC 变量必须在相同区域的 MRNP 命令中。

NC/PLC 变量区域分为如下几种：

- 一般 NC 数据 ($MN..., $SN.., /nck/...)
- 通道专用的 NC 数据 ($MC..., $SC.., /channel/...)
- PLC 数据 (DB..., MB.., /plc/...)
- 相同轴专用的 NC 数据 ($MA..., $SA..)

编程

句法：	**MRNP**(变量名称 1*变量名称 2[* ...], 寄存器索引)
说明：	读取多个变量
参数：	变量名称用"*"作为分隔符。按照命令中变量名称的顺序，采用寄存器 REG[寄存器索引] 中的值和下列值。 关系到： 第一个变量的值在 REG[寄存器索引] 中。 第二个变量的值在 REG[寄存器索引 + 1] 中 ，以次类推

说明

请注意，变量列表最多为 500 个字符，寄存器的数量也有一定限制。

示例

```
MRNP("$R[0]*$R[1]*$R[2]*$R[3]",1)      ; REG[1] 至 REG[4] 以变量值 $R[0] 至 $R[3] 描述。
```

NC 变量

提供所有的机床和设置数据以及 R 参数，但是仅特定的系统变量可用（另见：AUTOHOTSPOT）。

可访问所有全局的和通道专用的用户变量（GUD）。本地和程序全局的用户变量无法编辑。

机床数据	
全局机床数据	$MN_...
轴专用的机床数据	$MA_...
通道专用的机床数据	$MC_...

设定数据	
全局设定数据	$SN_...
轴专用的设定数据	$SA_...
通道专用的设定数据	$SC_...

系统变量	
R 参数 1	$R[1]

PLC 变量

所有 PLC 数据可用。

PLC 数据	
数据模块 x 的字节 y 位 z	DBx.DBXy.z
数据模块 x 的字节 y	DBx.DBBy
数据模块 x 的字 y	DBx.DBWy
数据模块 x 的双字 y	DBx.DBDy
数据模块 x 的实数 y	DBx.DBRy
标记字节 x 位 y	Mx.y
标记字节 x	MBx
标记字 x	MWx
标记双字 x	MDx
输入字节 x 位 y	Ix.y 或者 Ex.y

PLC 数据	
输入字节 x	IBx 或者 EBx
输入字 x	IWx 或者 EWx
输入双字 x	IDx 或者 EDx
输出字节 x 位 y	Qx.y 或者 Ax.y
输出字节 x	QBx 或者 ABx
输出字 x	QWx 或者 AWx
输出双字 x	QDx 或者 ADx
数据模块 x 中长度 z 的字符串 y	DBx.DBSy.z

7.3.28 PI 服务

说明

通过功能 PI_START 可以在由 PLC 在 NC 区中启动 PI 服务（程序实例服务）。

说明

可用的 PI 服务列表请见功能手册之基本功能。

编程

句法：	**PI_START**（“传输字符串”）	
说明：	执行 PI 服务	
参数：	“传输字符串”	传输的字符串与 OEM 文献相反，应用双引号括起。

示例

```
PI_START("/NC,001,_N_LOGOUT")
```

说明

通道相关的 PI 服务总是与当前的通道有关。
刀具功能（TO 区）的 PI 服务总是以分配到当前通道的 TO 区为参考。

7.3.29 读取 (RNP) 和写入 (WNP) 系统或用户变量

说明

通过指令 RNP (读取 NC PLC) 可以读取 NC 或者 PLC 变量或者机床数据。

编程

句法：	**RNP** ("系统或者用户变量",值)	
说明：	读取 PLC 或者 NC 变量或者机床数据	
参数：	系统或者用户变量	NC 或者 PLC 变量名称
	值	要写入系统或者用户变量中的值。 如果值类型为字符串，则必须以双引号括起。

示例

```
VAR2=RNP("$AA_IN[2]")          ; 读取 NC 变量
```

说明

通过指令 WNP (写入 NC PLC) 可以写入 NC 或者 PLC 变量或者机床数据。

在每次处理功能 WNP 都重新存取 NC 变量或 PLC 变量，即：总是在 CHANGE 方法下存取 NC 变量或 PLC 变量。这在系统或者用户变量经常改变值情况下有意义。如果只进行一次 NC/PLC 存取，则必须在 LOAD（装载）或者 UNLOAD（卸载）方法下设计。

编程

句法：	**WNP**("系统或者用户变量", 值)	
说明：	写入 PLC 或者 NC 变量或者机床数据	
参数：	系统或者用户变量	NC 或者 PLC 变量名称
	值	要写入系统或者用户变量中的值。 如果值类型为字符串，则必须以双引号括起。

示例

```
WNP("DB20.DBB1",1)          ; 写入 PLC 变量
```

7.3.30 RESIZE_VAR_IO 和 RESIZE_VAR_TXT

说明

使用命令 RESIZE_VAR_IO() 和 RESIZE_VAR_TXT() 可以更改变量的输入/输出分量或文本分量的几何尺寸。

设置了新的几何尺寸后，屏幕的所有栏都会调整位置，就像屏幕从一开始就是按这样的位置配置的。从而根据配置，各栏会相互重新对齐。

编程

句法：	**RESIZE_VAR_IO**(变量名，[X]，[Y]，[宽度]，[高度]) **RESIZE_VAR_TXT**(变量名，[X]，[Y]，[宽度]，[高度])	
说明：	更改变量的输入/输出分量或文本分量的几何尺寸。	
参数：	变量名	要更改输入/输出分量或文本分量的几何尺寸的变量的名称。
	X	左上 X 坐标
	Y	左上 Y 坐标
	宽度	宽度
	高度	高度

说明

如未给定 X、Y、宽度或高度的值，则保留当前值。如将其中一个值给定为 -1，则会设为由 “Run MyScreens” 预设的标准位置分量。

示例

```
RESIZE_VAR_IO("MyVar1", 200, , 100)    ；变量“MyVar1”的 IO 分量偏移至 X 位置 200 像素，宽度设为 100 像素。之前的 Y 位置和高度值被保留。
```

7.3.31 寄存器(REG)

寄存器说明

寄存器用于在两个不同的对话框之间切换数据。寄存器分配给每个对话框并且在载入第一个对话框时生成，以 0 或空字符串预占。

说明

寄存器不允许直接在 OUTPUT 方法中被用于 NC 代码的生成。

编程

句法：	**REG**[x]	
说明：	定义寄存器	
参数：	x	寄存器编号，以 x = 0...19 类型：REAL 或者 STRING = VARIANT x ≥ 20 的寄存器由西门子使用。

寄存器值说明

寄存器值的分配在方法中设计。

说明

如果由一个对话框通过功能 LM 生成另一个对话框，则寄存器的内容自动接受入新的对话框中并在第二个对话框中用于其它计算。

编程

句法：	名称.val = 寄存器值 或 名称 = 寄存器值	
说明：		
参数：	名称	寄存器名称
	寄存器值	寄存器的值

示例

```
UNLOAD
  REG[0] = VAR1           ; 寄存器 0 赋值为变量 1 的值
END_UNLOAD

UNLOAD
  REG[9].VAL = 84         ; 寄存器 9 赋值为变量 84 的值
END_UNLOAD
                          ; 在下列对话框中该寄存器可以在方法中重新分配本地变量。
LOAD
  VAR2 = REG[0]
END_LOAD
```

寄存器状态说明

通过状态特性可以在设计中查询寄存器是否包含一个有效值。

此外，当一个对话框作为主对话框使用时，可以使用寄存器的状态查询向一个寄存器中仅写入一个值。

编程

句法：	命名符.vld	
说明：	该属性仅可读。	
参数：	名称	寄存器名称
返回值：		查询结果可能是：
	FALSE =	无效值
	TRUE =	有效值

示例

```
IF REG[15].VLD == FALSE     ; 查询寄存器值的有效性
  REG[15] = 84
ENDIF
VAR1 = REG[9].VLD           ; 向 Var1 分配 REG[9] 状态查询的值。
```

7.3.32 RETURN

说明

通过 RETURN 功能可以提前取消当前的子程序处理并返回到最后一次 CALL 命令的跳转位置。

如果子程序中没有设计 RETURN，则执行子程序，直至结束然后返回到跳转位置。

编程

句法：	**RETURN**	
说明：	返回跳转位置	
参数：	- 无 -	

示例

```
//B(PROG1)                  ; 块开始
```

```
SUB(UP2)                    ; 子程序开始
  IF VAR1.val=="Otto"
    VAR1.val="Hans"
RETURN                      ; 如果变量值 = Otto，则赋值 "Hans" 给变量， 子程序在此位置结束。
ENDIF
  VAR1.val="Otto"           ; 如果变量值 ≠ Otto，则赋值  "Otto" 给变量。
END_SUB                     ; 子程序结束
//END                       ; 块结束
```

7.3.33 反编译

说明

在编程支持中可以**反编译**功能 GC 生成的 NC 代码并在所属输入对话框中的输入/输出栏中再次显示变量值。

编程

来自 NC 代码的变量接受至对话框中。 此时对来自 NC 代码的变量值与来自配置文件的计算的变量值加以比较。 如果存在不一致，则在日志文件中给出错误信息，因为在生成的 NC 代码中数值已改变。

如果一个变量在 NC 代码中多次存在，则在反编译时总是分析该变量最后一次出现的值。 另外在日志文件中给出警告。

代码生成时，不在 NC 代码中使用的变量作为使用注释存储。 通过使用注释标记所有反编译时需要的信息。 使用注释不得更改。

说明

NC 代码块和使用注释如果在一行的启始处开始，则只能反编译。

示例：

在程序中有下列 NC 代码：

```
DEF VAR1 = (I//101)
OUTPUT(CODE1)
  "X" VAR1 " Y200"
  "X" VAR1 " Y0"
END_OUTPUT
```

在零件程序中存放下列代码：

```
;NCG#TestGC#\cus.dir\aeditor.com#CODE1#1#3#
X101 Y200
X101 Y0
;#END#
```

在反编译时读取编辑器：

```
X101 Y200
X222 Y0          ; 在零件程序中更改 X 的值 (X101 → X222)
```

在输入对话框中给出下列 VAR1 值： `VAR1 = 222`

参见

生成代码(GC) (页 183)

7.3.34 无注释反编译

说明

在编程支持中，可以对通过 GC 功能生成的 NC 代码执行**无注释反编译**，并在对应输入对话框的输入输出栏中显示变量值。

编程

可以按照以下方式执行 GC 指令，忽略在常规的代码生成中产生的注释行：

```
GC("CODE1",D_NAME,1)
```

产生的代码通常是无法反编译的，但是执行以下操作，便可以反编译生成的循环调用代码：

- **扩展文件 “easyscreen.ini”**
  在文件“easyscreen.ini”中加入段落[RECOMPILE_INFO_FILES]。在该段落中列出所有 “ini” 文件，这些文件说明了哪些循环可以进行无注释反编译：
  ```
  [RECOMPILE_INFO_FILES]
  IniFile01 = cycles1.ini
  IniFile02 = cycles2.ini
  ```
  可以指定多个“INI”文件，文件名称可自由选择。
- **新建包含循环说明的“INI”文件**
  将含有循环说明的 ini 文件保存到以下路径中：
  [系统用户目录]/cfg
  [系统 oem 目录]/cfg
  [系统插件目录]/cfg
  在该文件中，每个循环需要占用一个单独的说明段落。段落的名称就是循环的名称：
  ```
  [Cycle123]
  Mname = TestGC
  Dname = testgc.com
  OUTPUT = Code1
  Anzp = 3
  Version = 0
  Code_typ = 1
  Icon = cycle123.png
  Desc_Text = This is describing text
  ```

Mname	窗口名称
Dname	对窗口进行定义的文件的名称
OUTPUT	所涉及 OUTPUT 方法的名称
Anzp	需要反编译的窗口的参数数量（指所有用 DEF 创建的变量，也包含辅助变量）
Version	循环的版本，可选

Icon	在工步链程序中显示的图符，格式为“png” 不同图片大小对应的图符分辨率： 640 x 480 mm → 16 x 16 像素 800 x 600 mm → 20 x 20 像素 1024 x 768 mm → 26 x 26 像素 1280 x 1024 mm → 26 x 26 像素 1280 x 768 mm → 26 x 26 像素 保存：[系统用户目录]/ico/ico<分辨率> **说明：** • 分辨率为“1280 x xx mm”的图片保存在用于“1024 x 768 mm”的文件夹下（仅针对工步链程序）。 • 图片大小不仅与分辨率有关，也与编辑器设置的字体大小有关。
Desc_Text	工步链程序中显示画面上的说明性文本，可选，字符串最多 17 个字符（仅针对工步链程序）。
Param_Text	（可选）参数列表文本 T=%1 D=%2

说明

Icon、Desc_Text 和 Param_Text 的说明：

1. 用户图标始终在 G 代码编辑器中显示。在 ShopMill/ShopTurn 编辑器中始终显示 G 代码图标。这可更加清楚地区别 ShopMill/ShopTurn 工步与非 ShopMill/ShopTurn 工步。
2. Run MyScreens 工步取决于编辑器设置“循环显示为工步”。这适用于 G 代码以及 JobShop 编辑器。
   即 “循环显示为工步” = “是” 时，使用 Run MyScreens 配置的 “Desc_Text”，“否” 时显示循环。
3. Desc_Text 和 Param_Text
   – 可通过符号 $xxxxx 指定语言
   – 通过 %1, %2, ... 可在生成的循环调用中访问参数。此时占位符会被替换为所生成的参数列表中位于'%'之后的参数。

示例

```
//M(测试 GC/"代码生成:")
DEF VAR1 = (R//1)
DEF VAR2 = (R//2)
```

```
DEF D_NAME
LOAD
  VAR1 = 123
  VAR2 = -6
END_LOAD
OUTPUT(CODE1)
  "Cycle123(" VAR1 "," VAR2 ")"
  "M30"
END_OUTPUT

PRESS(VS1)
  D_NAME = "\MPF.DIR\MESSEN.MPF"
  GC("CODE1",D_NAME)                          ; 将 OUTPUT 方法的 NC 代码写入文件 \MPF.DIR
                                              \MESSEN.MPF 中：
                                              Cycle123(123, -6)
                                              M30
END_PRESS
```

边界条件

- “无注释反编译”功能不具备“带注释反编译”的全部功能。
  “无注释反编译”支持典型的循环调用，如 MYCYCLE(PAR1, PAR2, PAR3, ...)。但在函数调用行中不允许存在任何注释。但是，不在函数调用时传递的以及字符串 S 型的可选参数至少应带有空的引号，如 **“”**。否则，“Run MyScreens” 会尝试使用逗号替代该参数，接着将其反编译为“被替代的循环调用”。
- 字符串型参数不允许待传递的字符串中含有逗号和分号。
- “无注释反编译”时，OUTPUT 方法中包含的所有变量都应位于括号内，这样功能“使用逗号替代缺少的循环参数”才能生效。

示例：

允许：

```
OUTPUT
    "MYCYCLE(" MYPAR1 "," MYPAR2 "," MYPAR3 ")"
END_OUTPUT
```

不允许（变量 MYCOMMENT 位于右括号后面）：

```
OUTPUT
    "MYCYCLE(" MYPAR1 "," MYPAR2 "," MYPAR3 ")"
MYCOMMENT
END_OUTPUT
```

7.3.35 向前/后查找(SF, SB)

说明

通过功能 **SF（向前查找）**、**SB（向后查找）**可以在编辑器当前的 NC 程序中从当前光标位置开始查找某个字符串并输出其值。

编程

句法：	**SF**("字符串")	
名称：	**S**earch **F**orward:从当前光标位置向前查找	
句法：	**SB**("字符串")	
名称：	**S**earch **B**ackward:从当前光标位置向后查找	
参数：	字符串	要查找的文本

查找规则：

- 要查找的字符串和数值的单元在 NC 程序中前后必须有空格。
- 查找对象无法在注释中和字符串内查找。
- 给出的值必须是一个数字表达式，表达式形式“X1=4+5”无法识别。
- 可识别十六进制常数的形式 X1=‘HFFFF'、二进制常数的形式 X1=‘B10010' 和指数常数的形式 X1=‘-.5EX-4’。
- 在字符串和数值之间有下列符号，可以给出字符串值：
  - 无
  - 空格键
  - 等号

示例

允许以下的写入方式：

```
X100 Y200                ; 变量 Abc 包含数值 200
Abc = SB("Y")
X100 Y 200               ; 变量 Abc 包含数值 200
Abc = SB("Y")
X100 Y=200               ; 变量 Abc 包含数值 200
Abc = SB("Y")
```

7.3.36 字符串功能

概述

以下的功能允许进行字符串处理：

- 确定字符串长度
- 查找字符串中的一个字符
- 由左提取部分字符串
- 由右提取部分字符串
- 由字符串中间提取部分字符串
- 替换部分字符串
- 字符串的比较
- 将字符串插入到另一个字符串中
- 从字符串中删除一个字符串
- 删除空格符（从左或从右）
- 插入带格式化标识的值或字符串

LEN 功能：字符串长度

句法：	**LEN**(字符串 | 变量名称)
说明：	确定一个字符串的字符数目。

参数：	字符串	每个有效的字符串表达式。对于一个空字符串，会返回零。
	变量名称	每个有效的和表示的变量名称
	仅允许两个可行参数中的一个。	

示例

```
DEF VAR01
DEF VAR02

LOAD
  VAR01="HALLO"
  VAR02=LEN(VAR01)           ; 结果 = 5
END_LOAD
```

NSTR 功能：查找字符串中的字符

句法：	**INSTR**(开始, 字符串 1, 字符串 2 [,方向])	
说明：	查找字符	
参数：	开始	从字符串 1 向字符串 2 查找的开始位置。如果从字符串 2 的开头开始查找，则指定为 0。
	字符串 1	要查找的字符。
	字符串 2	在该字符链中查找
	方向（可选）	查找的方向 0:从左向右（预设） 1:从右向左
	如果字符串 2 中没有包含字符串 1，则返回数值 0。	

示例

```
DEF VAR01
DEF VAR02

LOAD
  VAR01="HALLO/WELT"
```

```
  VAR02=INST(1,"/",VAR01)        ; 结果 = 6
END_LOAD
```

LEFT 功能：左边字符串

句法：	**LEFT**(字符串, 长度)	
说明：	LEFT 返回一个字符串，该字符串从字符串左侧开始包含指定的字符数。	
参数：	字符串	字符串或者带有要处理的字符串的变量
	长度	要读取的字符数目

示例

```
DEF VAR01
DEF VAR02
LOAD
  VAR01="HALLO/WELT"
  VAR02=LEFT(VAR01,5)           ; 结果 = “HALLO”
END_LOAD
```

RIGHT 功能：右边字符串

句法：	**RIGHT**(字符串, 长度)	
说明：	RIGHT 返回一个字符链，该字符链从字符串右侧开始包含指定的字符数。	
参数：	字符串	字符串或者带有要处理的字符串的变量
	长度	要读取的字符数目

示例

```
DEF VAR01
DEF VAR02
LOAD
  VAR01="HALLO/WELT"
  VAR02=LEFT(VAR01,4)         ; 结果 = “WELT”
END_LOAD
```

MIDS 功能：中间字符串

句法：	**MIDS**(字符串, 开始 [, 长度])	
说明：	MIDS 返回一个字符串，该字符串从字符串的指定位置开始包含指定的字符数。	
参数：	字符串	字符串或者带有要处理的字符串的变量
	开始	开始，从字符链中该处开始读取
	长度	要读取的字符数目

示例

```
DEF VAR01
DEF VAR02
LOAD
  VAR01="HALLO/WELT"
  VAR02=LEFT(VAR01,4,4)            ; 结果 = "LO/W"
END_LOAD
```

REPLACE 功能：替换字符

句法：	**REPLACE** (字符串, 查找字符串, 替换字符串 [, 开始 [, 计数]])	
说明：	功能 REPLACE 用另一个字符/字符链替代字符串中的一个字符/字符链。	
参数：	字符串	待通过替换字符串替换查找字符串的字符串。
	查找字符串	需被替代的字符串
	替代字符串	替代字符串（位于查找字符串位置）
	开始	查找和替换的开始位置
	计数	从开始位置起要开始查找的查找字符串的字符数量
返回值：		
	字符串 = 空字符串	复制字符串
	查找字符串 = 空字符串	复制字符串
	替换字符串 = 空字符串	复制字符串，在该字符串中删除所有出现的查找字符串
	开始 > Len(长度)	空字符串
	计数 = 0	复制字符串

STRCMP 功能：比较字符串

句法：	**STRCMP**(字符串 1，字符串 2 [, CaseInsensitive])	
说明：	STRCMP 将一个字符链与另一个字符链进行比较。	
参数：	字符串 1	要与第二个字符链进行比较的字符链
	字符串 2	要与第一个字符链进行比较的字符链
	CaseInsensitive	比较时区分/不区分大/小写： 未说明或 FALSE = 区分大/小写 TRUE = 不区分大/小写

示例

```
REG[0]=STRCMP("Hugo", "HUGO")              ; 结果 = 32
                                           ; (<> 0，字符串不相同)

REG[0]=STRCMP("Hugo", "HUGO", TRUE)        ; 结果 = 0
                                           ; (== 0，字符串相同
```

STRINSERT 功能：插入字符串

句法：	**STRINSERT**(字符串 1，字符串 2，插入位置)	
说明：	STRINSERT 可在一个字符链的某个特定位置上插入另一个字符链。	
参数：	字符串 1	要向其中插入一个字符链的字符链
	字符串 2	要插入的字符链
	插入位置	插入字符链的位置

示例

```
REG[0]=STRINSERT("Hello!", " world", 5)    ; 结果 = "Hello world!")
```

STRREMOVE 功能：删除字符串

句法：	**STRREMOVE**(字符串 1，删除位置，计数)	
说明：	STRREMOVE 可从字符链的某一位置上删除一定数量的字符。	
参数：	字符串 1	要从其中删除字符的字符链
	删除位置	从字符链的该位置起删除字符
	计数	要删除的字符数量

示例

```
REG[0]=STRREMOVE("Hello world!", 5, 6)     ; 结果 = "Hello!")
```

TRIMLEFT 功能：从字符串左侧删除空格符

句法：	**TRIMLEFT**(字符串 1)	
说明：	TRIMLEFT 可从字符链的左侧删除空格符。	
参数：	字符串 1	要从其左侧开始删除空格符的字符链

示例

```
REG[0]=TRIMLEFT("    Hello!")              ; 结果 = "Hello!")
```

TRIMRIGHT 功能：从字符串右侧删除空格符

句法：	**TRIMRIGHT**(字符串 1)	
说明：	TRIMRIGHT 可从字符链的右侧删除空格符。	
参数：	字符串 1	要从其右侧开始删除空格符的字符链

示例

```
REG[0]=TRIMRIGHT("Hello!    ")             ; 结果 = "Hello!")
```

FORMAT 功能：插入带格式化标识的值或字符串

句法：	**FORMAT**(带格式化标识的文本[,值 1] ... [,值 28])	
说明：	FORMAT 功能可通过使用格式化标识在预设文本的某个位置插入最多 28 个值或字符串。	
格式化标识：	句法	%[标记] [宽度] [.小数点后面的位置] 类型
	标记	用于确定输出格式的可选字符： • 右对齐或左对齐（"-" 用于左对齐） • 前面加零（"0"） • 缺省值：当要输出的值的数位少于[宽度]定义时，使用空格符补齐
	宽度	确定一个非负数最小输出宽度的依据。如果值的数位少于依据所确定的，则使用空格符补齐。
	小数点后的数位	使用浮点数时，该可选参数用来确定小数点后面的位数。
	类型	类型说明符用来确定 PRINT 指令会输出哪些数据格式。该说明符必须进行给定。 • d:整型值 • f：浮点数 • s:字符串 • o:八进制 • x:十六进制 • b:二进制

示例

```
DEF VAR1
DEF VAR2
LOAD
VAR1 = 123
VAR2 = FORMAT("Hello %08b %.2f %s!", VAR1 + 1, 987.654321, "world")
                                    ; 结果 = "Hello 01111100 987.65 world!"
END_LOAD
```

参见

使用字符串 (页 124)

7.3.37 WHILE/UNTIL 循环

说明

使用 DO-LOOP 命令可实现一个循环。取决于配置，这个循环会一直执行，直到满足一个条件(WHILE)或一个条件适用(UNTIL)。

取决于配置，由于循环会影响系统的性能，因此请您慎重使用这些循环并且放弃使用循环中费时的动作。

例如，建议使用寄存器(REG[])作为运行变量，因为一般显示变量（尤其是那些使用系统或用户变量连接的）会极其频繁地刷新或写入，这同样会影响系统性能。

借助功能 DEBUG（参见章节 DEBUG (页 174)）可对 “Run MyScreens” 方法的运行时间进行检测。这样还可对循环可能导致的问题（高 CPU 负载、响应性能降低）进行识别。

说明

由于 FOR 循环可被 WHILE 循环替代，因此在 EasyScreen 中不支持 FOR 循环的表达句法。

编程

```
DO
   <指令>
LOOP_WHILE <循环继续执行的条件>

DO
   <指令>
LOOP_UNTIL <循环结束的条件>

DO_WHILE <循环继续执行的条件>
   <指令>
LOOP

DO_UNTIL <循环结束的条件>
   <指令>
LOOP
```

示例

```
REG[0] = 5
DO
   DEBUG("OUTER: " << REG[0])
   REG[0] = REG[0] + 1
   REG[1] = -5

   DO
      DEBUG("INNER: " << REG[1])
      REG[1] = REG[1] + 1
   LOOP_WHILE REG[1] < 0

LOOP_WHILE REG[0] < 10
```

```
REG[0] = 5
DO
   DEBUG("OUTER: " << REG[0])
   REG[0] = REG[0] + 1
   REG[1] = -5

   DO
      DEBUG("INNER: " << REG[1])
      REG[1] = REG[1] + 1
   LOOP_UNTIL 0 <= REG[1]

LOOP_WHILE 10 > REG[0]
```

```
REG[0] = 5
DO_WHILE 10 > REG[0]
   DEBUG("OUTER: " << REG[0])
   REG[0] = REG[0] + 1
   REG[1] = -5

   DO_UNTIL 0 <= REG[1]
     DEBUG("INNER: " << REG[1])
     REG[1] = REG[1] + 1
   LOOP
```

```
LOOP
```

```
REG[0] = 5
DO_WHILE 10 > REG[0]
   DEBUG("OUTER: " << REG[0])
   REG[0] = REG[0] + 1
   REG[1] = -5

   DO
     DEBUG("INNER: " << REG[1])
     REG[1] = REG[1] + 1
   LOOP_UNTIL 0 <= REG[1]

LOOP
```

7.3.38 循环执行脚本： START_TIMER，STOP_TIMER

说明

借助计时器可循环调用 SUB 方法。为此提供了功能 START_TIMER()和 STOP_TIMER()。

说明

只可为每个 SUB 方法配置一个计时器。

编程

句法：	**START_TIMER**(“SUB 名称”，间隔)	
说明：	开始 SUB 方法的循环执行	
参数：	SUB 名称：	要循环调用的 SUB 方法的名称
	间隔：	以 ms 为单位的时间间隔

句法：	**STOP_TIMER**("SUB 名称")	
说明：	停止 SUB 方法的循环执行	
参数：	SUB 名称：	要停止其计时器的 SUB 方法的名称

示例

```
//M(TimerSample/"My timer")
DEF MyVariable=(I//0/,"Number of cyclic calls:"/WR1)
VS1=("Start%ntimer")
VS2=("Stop%ntimer")

SUB(MyTimerSub)
   MyVariable = MyVariable + 1
END_SUB

PRESS(VS1)
   ; 每 1000 ms 调用一次 SUB "MyTimerSub"
   START_TIMER("MyTimerSub", 1000)
END_PRESS

PRESS(VS2)
   STOP_TIMER("MyTimerSub")
END_PRESS
```

如果为已经分配给一个 SUB 方法的计时器重新调用 START_TIMER，则在有差异时会接收新的间隔。否则这两次调用都会被忽略。

由于系统的限制，最短间隔为 100 ms。

如果为一个 SUB 方法调用 STOP_TIMER，而当时没有正在运行的计时器，则会忽略此次调用。

图形单元和逻辑单元

8

8.1 线、分割线、矩形、圆形和椭圆形

说明

线、分割线、矩形和椭圆形/圆形在 LOAD 方法中配置：

- 可将填充色设置为系统背景色，以得到透明的矩形。
- 只要高度和宽度设为相同，元素 ELLIPSE 上就可生成圆形。
- 水平和垂直分割线始终与窗口宽度或窗口高度完全一致。分割线不会产生滚动条。
  如果此前一直将元素 RECT 用于显示分割线，例如 RECT(305,0,1,370,0,0,1)，这会被编程支持识别到并自动转换为 V_SEPARATOR(305,1,"#87a5cd", 1)。这同时适用于垂直和水平分割线。

LINE 元素

编程：

句法：	LINE (x1,y1,x2,y2,f,s)	
说明：	定义线	
参数：	x1	起点的 x 坐标
	y1	起点的 y 坐标
	x2	终点的 x 坐标
	y2	终点的 y 坐标
	f	线的颜色
	s	线的样式： 1 = 实线 2 = 虚线 3 = 点线 4 = 点虚线

RECT 元素

编程：

句法：	RECT (x,y,w,h,f1,f2,s)	
说明：	定义矩形	
参数：	x	左上 x 坐标
	y	左上 y 坐标
	w	宽度
	h	高度
	f1	边框颜色
	f2	填充色
	s	边框样式： 1 = 实线 2 = 虚线 3 = 点线 4 = 点虚线

ELLIPSE 元素

编程：

句法：	ELLIPSE(x,y,w,h,f1,f2,s)	
说明：	定义椭圆或圆形	
参数：	x	左上 x 坐标
	y	左上 y 坐标
	w	宽度
	h	高度
	f1	边框颜色
	f2	填充色
	s	边框样式： 1 = 实线 2 = 虚线 3 = 点线 4 = 点虚线

高度与宽度值相同时为圆形。

V_SEPARATOR 元素

编程：

句法：	V_SEPARATOR (x,w,color,pen)	
说明：	定义垂直分割线	
参数：	x	x 位置
	w	线条粗细
	color	颜色
	pen	线的样式： 1 = 实线 2 = 虚线 3 = 点线 4 = 点虚线

H_SEPARATOR 元素

编程：

句法：	H_SEPARATOR (y,h,color,pen)	
说明：	定义水平分割线	
参数：	y	y 位置
	h	线条粗细
	color	颜色
	pen	线的样式： 1 = 实线 2 = 虚线 3 = 点线 4 = 点虚线

另见

LOAD (页 150)

参见

CLEAR_BACKGROUND (页 165)

8.2 定义数组

定义

通过数组可以排列或者保存同一数据类型的数据，从而可以通过索引存取数据。

说明

数组可以是一维或者二维。一个一维数组可视为带有一行或者一列的一个二维数组。

数组通过标记 //A 定义并通过 //END 结束。行和列数目任意。一个数组有下列结构：

编程

句法：	**//A**(名称) (a/b...) (c/d...) ... //END	
说明：	定义数组	
参数：	名称	数组名称
	a, b, c, d	数组值 STRING 类型的值必须用双引号括起。

示例

```
//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)
(1.0 / 0.25  / 0.676)
(1.2 / 0.25  / 0.676)
(1.4 / 0.3   / 1.010)
(1.7 / 0.35  / 1.246)
//END
```

8.2.1 存取数组单元的值

说明

通过属性值（名称.val）可以继续传送一个数组存取值。

行索引（数组的行编号）和列索引（数组的列编号）各自从 0 开始。如果显示数组外的行索引或者列索引，则输出值 0 或者空字符串并且变量 ERR 的值为 TRUE。 当未找到查找关键字时，变量 ERR 同样为 TRUE。

编程

句法：	名称 [Z,[M[,C]]].val 或者 名称 [Z,[M[,C]]]
说明：	存取仅带有一列的一维数组：
句法：	名称 [S,[M[,C]]].val] 或者 名称 [S,[M[,C]]] 或者
说明：	存取仅带有一行的单维数组
句法：	名称 [Z,S,[M[,C]]].val 或者 名称 [Z,S,[M[,C]]]
说明：	存取二维数组

参数：	名称：	数组名称	
	Z:	行值（行索引或者查找关键字）	
	S:	列值（列索引或者查找关键字）	
	M:	存取模式	
		0	直接
		1	按行查找，列直接
		2	按列查找，行直接
		3	查找
		4	查找行索引
		5	查找列索引
	C:	比较模式	
		0	查找关键字必须位于行或者列的值范围内。
		1	查找关键字必须准确找到

示例：	`VAR1 = MET_G[REG[3],1,0].VAL`	；分配 Var1 数组中的一个值 MET_G

存取模式

- **存取模式“直接”**
  在存取模式“直接”（M = 0）情况下，数组上的存取通过行索引（以 Z 表示）和列索引（以 S 表示）实现。不评估比较模式 C。
- **存取模式“查找”**
  在存取模式 M = 1、2 或者 3 情况下，查找总是在行 0 或者列 0 中实现。

模式 M	行值 Z	列值 S	输出值
0	行索引	栏索引	行 Z 和列 S 中的值
1	查找关键字： 在列 0 中查找	列索引，从该列中读取值	查找的行和列 S 中的值
2	行索引，从该行中读取返回值	查找关键字： 在行 0 中查找	行 Z 和查找的列中的值

模式 M	行值 Z	列值 S	输出值
3	查找关键字： 在列 0 中查找	查找关键字： 在行 0 中查找	查找的行和查找的列中的值
4	查找关键字： 在列 S 中查找	行索引，在该行中查找	行索引
5	行索引，在该行中查找	查找关键字： 在行 Z 中查找	栏索引

比较模式

在使用比较模式 C = 0 时，查找行或者查找列的内容以升序分类。 如果查找关键字小于第一个元素或者大于最后一个元素，则给出值 0 或者一个空字符并且错误变量 ERR 为真。

在使用比较模式 C = 1 时，查找关键字必须可在查找行或者查找列中找到。 如果没有找到查找关键字，则给出值 0 或者一个空字符并且错误变量 ERR 为真。

8.2.2 举例： 存取数组单元

前提条件

下列定义两个数组是下面例子的前提条件。

```
//A(螺纹)
                (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)
                (1.0 / 0.25  / 0.676)
                (1.2 / 0.25  / 0.676)
                (1.4 / 0.3   / 1.010)
                (1.7 / 0.35  / 1.246)
//END
```

```
//A(Array2)
                ("BEZ" /     "STG" /     "KDM" )
                (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 )
                (1.0 /       0.25 /      0.676 )
                (1.2 /       0.25 /      0.676 )
                (1.4 /       0.3 /       1.010 )
                (1.7 /       0.35 /      1.246 )
//END
```

示例

- **存取模式 1 示例：**
  在 Z 中有查找关键字。 该关键字总是在列 0 中查找。 通过查找的关键字的行索引给出列 S 中的值：
  `VAR1 = 螺纹[0.5,1,1] ; VAR1 值为 0.125`
  说明:
  数组“螺纹”的列 0 中查找值 0.5 并给出列 1 中查找的相同行的值。
- **存取模式 2 示例：**
  在 S 中有查找关键字。 该关键字总是在行 0 中查找。 通过查找的关键字的列索引给出行 Z 中的值：
  `VVAR1 = ARRAY2[3,"STG",2] ; VAR1 值为 0.125`
  说明:
  在数组“Array2”的行 0 中查找带有内容“STG”的列。 得到查找的列中的值和带有索引 3 的行。
- **存取模式 3 示例：**
  在 Z 和 S 中总是有一个查找关键字。 通过 Z 中的关键字在列 0 中找到行索引和通过 S 中的关键字在行 0 中找到列索引。 通过查找的行索引和列索引输出数组中的值：
  `VAR1 = ARRAY2[0.6,"STG",3]  ;VAR1 值为 0.15`
  说明:
  在数组“Array2”的列 0 中查找带有内容 0.6 的行，在数组“Array2”的行 0 中查找带有内容“STG”的列。 根据 VAR1 给出查找的行和列中的值。

- **存取模式 4 示例：**
  在 Z 中有查找关键字。 在 S 中有要查找的列索引。 给出查找的关键字的行索引：
  VAR1 = 螺纹[0.125,1,4] VAR1 值为 2
  说明:
  在数组“ 螺纹的列 1 中查找值 0.125 并根据 VAR1 给出查找的值的行索引。
- **存取模式 5 示例：**
  在 Z 中有要查找行的行索引。 查找关键字在 S 中。给出查找的关键字的列索引：
  VAR1 = 螺纹[4,0.2,5,1] VAR1 值为 1
  说明:
  在数组“ 螺纹” 的行 4 中查找值 0.2 并根据 VAR1 给出查找的值的列索引。 已选择比较模式 1，因为行 4 的值不是按升序分类的。

8.2.3 查询数组单元的状态

说明

通过属性状态可以查询数组存取是否提供一个有效值。

编程

句法：	名称 [Z, S, [M[,C]]]**.vld**	
说明：	该属性仅可读。	
参数：	名称	数组名称
返回值：	FALSE =	无效值
	TRUE =	有效值

示例

```
DEF MPIT = (R///"MPIT",,"MPIT",""/wr3)
DEF PIT  = (R///"PIT",,"PIT",""/wr3)
PRESS(VS1)
  MPIT = 0.6
  IF MET_G[MPIT,0,4,1].VLD == TRUE
    PIT    = MET_G[MPIT,1,0].VAL
    REG[4] = PIT
    REG[1] = "OK"
```

```
  ELSE
    REG[1] = "ERROR"
  ENDIF
END_PRESS
```

8.3 表格描述(栅格)

定义

与数组相反，表格(栅格)的值会持续更新。 这会涉及（例如）来自 NC 或 PLC 的数值的表格式显示。 表格是按列组织排列的，因此一列中始终为同一类型的变量。

分配

表格描述参考内容收录在变量描述中：

- 变量定义确定显示的值，表格单元定义确定屏幕的外观和布置。 来自变量定义行的输入/输出栏属性将会接收到表格中。
- 表格的可见区域通过输入/输出栏的宽度和高度确定。 如果比可见区域位置存在更多的行和列，则可以通过水平和垂直滚动条进行查看。

表格名称

NCK/PLC 相同类型值的表格名称，它可以通过通道模块编译地址。 表格名称通过一个前置的极限值或者转换栏的 % 符号进行区分。 表格命名符可以通过逗号分隔，还可以跟随一个文件名，该文件名指定定义表格描述的文件。

同类型值的表格命名符，例如来自 NCK 或 PLC 的值。 表格命名符在配置极限值或转换栏的地方给定。 表格命名符前会带有 % 符号。 之后可用逗号隔开并加上一个文件名。 这用于给定在哪个文件中定义表格描述。 表格默认会在配置文件中查找屏幕本身的配置位置。

表格定义也可以通过功能 Load Grid (LG)进行动态加载，例如在 LOAD 方法中。

系统或者用户变量

这些参数保留为空以用于表格，因为详细信息中要显示的变量会在列定义行中进行给定。 表格描述可以动态提供。

示例

变量 MyGridVar 的定义，该变量在您的输入/输出栏（左边距：100，上边距：标准，宽度：350，高度：100）中显示为一个栅格“MyGrid1”。

```
DEF MyGridVar=(V/% MyGrid1///////100,,350,100)
```

另见

变量参数 (页 107)

Load Grid (LG) (页 192)

8.3.1 定义表格(栅格)

说明

表格块由以下部分组成：

- 标题描述
- 1 至 n 列描述

编程

句法：	***//G***（表格名称*/*表格类型*/*行数*/* [固定行属性],[固定列属性]）		
说明：	定义表格(栅格)		
参数：	表格名称	这里表格名称不使用前缀 % 符号。表格名称只能在一个对话框中使用一次。	
	表格类型	0（预设置）	PLC 或者用户数据（NCK 和通道专用的数据）表
		1	备用
	行数	行数包括标题行	
		固定行或者固定列无法滚动显示。列数由设计的列的数目给定。	
	列标题行的显示	-1:	不显示列标题行
		0:	显示列标题行（默认）
	固定列的数量	0:	无固定列
		1 ... n:	列数应在各列都可见的范围内，即不需水平滚动

示例

`//G(grid1/0/5/-1,2)`	列标题行隐藏并有 2 个固定列
`//G(grid1/0/5/,1)`	列标题行显示并有 1 个固定列
`//G(grid1/0/5)`	列标题行显示并且无固定列

8.3.2 定义列

说明

为表格(栅格)使用带下标的变量非常重要。索引号码对于带有一个或者多个索引的 PLC 或者 NC 变量比较重要。

您可以对表格中显示的值在由属性确定的权限范围和可能已设定的极限范围内直接加以编辑。

编程

句法：	（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） 另见 扩展配置句法 (页 51)。
说明：	定义列

参数：	和变量类似	
	类型	数据类型
	极限值	最小极限值，最大极限值
	长文本，列标题	
	属性	
	帮助画面	
	系统或者用户变量	作为变量在双引号内给出 PLC 或者 NC 变量。
	列宽	参数，单位像素
	偏移	在分配的偏移参数内规定步宽（在该步宽中各个索引应指数运算），以填写该列。 • 偏移 1：第 1 个索引的步宽 • 偏移 2：第 2 个索引的步宽 • 偏移 3：第 3 个索引的步宽

文本文件的列标题

列标题可以规定为文本或者文本号码（$8xxxx），并且同样无法滚动显示。

改变列属性

可动态改变的（可写）的列属性称为：

- 极限值（最大、最小）
- 列标题（st）
- 属性（wr、ac 和 li）
- 帮助图形（hlp）

通过定义行中的变量命名符和列索引（以 1 开始）改变列属性。

示例：`VAR1[1].st="Spalte 1"`

对于列定义，可以规定属性 wr、ac 和 li。

另见

Load Grid (LG) (页 192)

8.3.3 表格中的焦点控制（栅格）

说明

通过列和行属性可以在表格中设置和确定聚焦：

- 名称 **Row**
- 名称 **Col**

编程

表格的每行都具有属性 Val 和 Vld。

对于行属性写入和读出，除了定义行中的变量命名符之外，还规定一个行索引和列索引。

句法：	名称[行索引，列索引].Vld 或者 名称[行索引，列索引]
说明：	Val 属性
句法：	名称[行索引，列索引].Vld
说明：	Vld 属性

示例

```
Var1[2,3].val=1.203
```

如果没有规定行索引和列索引，则适用于聚焦行的索引，即：

```
Var1.Row =2
Var1.Col=3
Var1.val=1.203
```

8.4 自定义小部件

8.4.1 自定义小部件

说明

通过自定义小部件，可以在对话框内设计用户专用的显示部件。



软件选件

使用自定义小部件功能额外需要下列软件选件：

“SINUMERIK Integrate Run MyHMI /3GL” (6FC5800-0AP60-0YB0)

编程

定义:	**DEF** （名称）	
句法：	**(W///"","**（库名称）**.**（类别名称）**"/////**a,b,c,d**);**	
说明：	W	自定义小部件
参数：	名称	自定义小部件的名称，可自由选择
	库名称	可自由选择，dll (Windows)的名称或 so (Linux) 库文件的名称
	类别名称	可自由选择，是前面指出的库的类别功能的名称
	a, b, c, d	小部件的位置和大小

示例

在对话框配置文件中，可以按照以下方式自定义小部件：

```
DEF Cus =
(W///"","slestestcustomwidget.SlEsTestCustomWidget"/////
20,20,250,100);
```

8.4.2 自定义小部件库的结构

说明

自定义小部件库基本上都包含了一个定义的类别，在对话框配置文件中，应在指定库名称后指定该类别的名称。Run MyScreens 根据库名称访问同名 dll 文件，例如：

```
slestestcustomwidget.dll
```

编程

dll 文件的类别定义应为：

```
#define SLESTESTCUSTOMWIDGET_EXPORT Q_DECL_EXPORT

class SLESTESTCUSTOMWIDGET_EXPORT SlEsTestCustomWidget  : public QWidget
{
   Q_OBJECT
....
public slots:
   bool serialize(const QString& szFilePath, bool bIsStoring);
....
}
```

8.4.3 自定义小部件接口的结构

说明

自定义小部件的显示需要在库中增加一个接口。 该接口包含了 Run MyScreens 初始化自定义小部件的宏定义。 接口在 cpp 文件中定义。 文件名称可以自由定义，例如：
`sleswidgetfactory.cpp`

编程

可以按照下面的方式定义接口：

```
#include "slestestcustomwidget.h"      ; 将相关自定义小部件的标题文件添加到文件开头
....
//Makros                               ; 保持宏定义
```

```
....
WIDGET_CLASS_EXPORT(SlEsTestCustomWi   ；在文件末尾申明相关的自定义小部件
dget)
```

示例

自定义小部件文件“sleswidgetfactory.cpp”的内容，类别名称为“SlEsTestCustomWidget”：

```
#include <Qt/qglobal.h>
#include "slestestcustomwidget.h"

////////////////////////////////////////////////////////////////////////
// MAKROS FOR PLUGIN DLL-EXPORT - DO NOT CHANGE
////////////////////////////////////////////////////////////////////////

#ifndef Q_EXTERN_C
#ifdef __cplusplus
#define Q_EXTERN_C    extern "C"
#else
#define Q_EXTERN_C    extern
#endif
#endif

#define SL_ES_FCT_NAME(PLUGIN) sl_es_create_ ##PLUGIN
#define SL_ES_CUSTOM_WIDGET_PLUGIN_INSTANTIATE( IMPLEMENTATION , PARAM) \
    { \
    IMPLEMENTATION *i = new PARAM; \
    return i; \
    }

#ifdef Q_WS_WIN
#   ifdef Q_CC_BOR
#       define EXPORT_SL_ES_CUSTOM_WIDGET_PLUGIN(PLUGIN,PARAM) \
    Q_EXTERN_C __declspec(dllexport) void* \
    __stdcall SL_ES_FCT_NAME(PLUGIN) (QWidget* pParent) \
    SL_ES_CUSTOM_WIDGET_PLUGIN_INSTANTIATE( PLUGIN,PARAM )
#   else
#       define EXPORT_SL_ES_CUSTOM_WIDGET_PLUGIN(PLUGIN,PARAM) \
    Q_EXTERN_C __declspec(dllexport) void* SL_ES_FCT_NAME(PLUGIN) \
    (QWidget* pParent) \
    SL_ES_CUSTOM_WIDGET_PLUGIN_INSTANTIATE( PLUGIN,PARAM )
#   endif
#else
```

```
#   define EXPORT_SL_ES_CUSTOM_WIDGET_PLUGIN(PLUGIN,PARAM) \
    Q_EXTERN_C void* SL_ES_FCT_NAME(PLUGIN) (QWidget* pParent) \
    SL_ES_CUSTOM_WIDGET_PLUGIN_INSTANTIATE( PLUGIN,PARAM )
#endif

#define WIDGET_CLASS_EXPORT(CLASSNAME)  \
    EXPORT_SL_ES_CUSTOM_WIDGET_PLUGIN(CLASSNAME,CLASSNAME(pParent))

////////////////////////////////////////////////////////////////////////
// FOR OEM USER - please declare here your widget classes for export
////////////////////////////////////////////////////////////////////////

WIDGET_CLASS_EXPORT(SlEsTestCustomWidget)
```

8.4.4 自定义小部件与对话框的互动 - 自动数据交换

自定义小部件与对话框相互影响并能显示或处理值。

条件

自动数据交换会在以下条件下进行：

条件	方向
打开或反编译对话框时	对话框 → 自定义小部件
执行用于生成循环调用的 GC 指令时	自定义小部件 → 对话框

编程

针对互动，以下定义都是必须的：

对话框设计扩展

定义:	**DEF** (变量)	
句法：	**((**类型**)//5/"","("**变量**)","/wr2/)**	
变量类型：	类型	带任意数据类型（无 W）的标准输入区（无栅格或转换）
参数：	变量	用于数据交换的变量的任意名称
输入模式：	wr2	读和写

示例

```
DEF CUSVAR1 = (R//5/"","CUSVAR1",""/wr2/)
```

类定义扩展

在自定义小部件的类定义中必须创建 QProperty，其名称应与在对话框设计中选择的变量名称一致，例如：

```
Q_PROPERTY(double CUSVAR1 READ cusVar1 WRITE setCusVar1);
```

示例

dll 文件的类定义应为：

```
#define SLESTESTCUSTOMWIDGET_EXPORT Q_DECL_EXPORT

class SLESTESTCUSTOMWIDGET_EXPORT SlEsTestCustomWidget  : public QWidget
{
 Q_OBJECT
 Q_PROPERTY(double CUSVAR1  READ cusVar1    WRITE setCusVar1);
....
....
}
```

8.4.5 自定义小部件与对话框的互动 - 手动数据交换

除自动数据交换外，也可进行手动数据交换。该交换是动态进行的，即在对话框运行期间进行。可执行以下操作：

- 可读取和写入自定义小部件的属性。
- 可从 Run MyScreens 配置中调用自定义小部件的方法。
- 可以响应一个特定的自定义小部件信号并借此在 Run MyScreens 配置中调用子程序 (SUB)。

8.4.5.1 读取和写入属性

说明

Run MyScreens 配置中提供了 ReadCWProperties 和 WriteCWProperties 函数用于读取和写入自定义小部件的属性。

编程

句法：	**ReadCWProperty**（“变量名称”，“属性名称”）	
说明：	读取自定义小部件的属性	
参数：	变量名	分配到一个自定义小部件的对话框变量的名称
	属性名称	待读取的自定义小部件属性的名称
返回值：	自定义小部件属性的当前值	

句法：	**WriteCWProperty**（“变量名称”，“属性名称”，“值”）	
说明：	写入 CustomWidget 的属性	
参数：	变量名	分配到一个自定义小部件的对话框变量的名称
	属性名称	待写入的自定义小部件属性的名称
	值	应在 CustomWidget 属性中写入的值

示例

示例 1：

读取包含对话框变量 “MyCWVar1” 的自定义小部件属性 “MyStringVar” 并在寄存器 7 中分配值。

CustomWidget 类声明：

```
class SLESTESTCUSTOMWIDGET_EXPORT SlEsTestCustomWidget  :public QWidget
{
   Q_OBJECT
   Q_PROPERTY(QString MyStringVar
              READ myStringVar
              WRITE setMyStringVar);
   ...
```

对话框配置：

```
DEF MyCWVar1 = (W///,"slestestcustomwidget.SlEsTestCustomWidget")
PRESS(VS1)
   REG[7]=ReadCWProperty("MyCWVar1", "MyStringVar")
END_PRESS
```

示例 2：

在包含对话框变量 “MyCWVar1” 的 CustomWidget 属性 “MyRealVar” 中写入计算结果 “3 + sin(123.456)”。

CustomWidget 类声明：

```
class SLESTESTCUSTOMWIDGET_EXPORT SlEsTestCustomWidget  :public QWidget
{
   Q_OBJECT
   Q_PROPERTY(double MyRealVar
              READ myRealVar
              WRITE setMyRealVar);
   ...
```

对话框配置：

```
DEF MyCWVar1 = (W///,"slestestcustomwidget.SlEsTestCustomWidget")
PRESS(VS1)
   WriteCWProperty("MyCWVar1", "MyRealVar", 3 + sin(123.456))
END_PRESS
```

8.4.5.2 执行自定义小部件的方法

说明

Run MyScreens 配置中提供了 CallCWMethod 函数用于执行自定义小部件的方法。

待调用的自定义小部件方法最多可以有 10 个传输参数。

支持以下传输参数数据格式：

- bool
- uint
- int
- double
- QString
- QByteArray

编程

句法：	**CallCWMethod**（“变量名称”，“方法名称[，自变量 0][，自变量 1 ... [，自变量 9]”）	
说明：	调用 CustomWidget 方法	
参数：	变量名	分配到一个自定义小部件的对话框变量的名称
	方法名称	待调用的自定义小部件方法的名称
	自变量 0 - 9	CustomWidget 方法的传输参数 不支持的数据格式：参见前面的 **说明：**传输参数将始终传输 “ByVal”，即始终只传输值，不传输变量。
返回值：	自定义小部件方法的返回值 支持以下传输参数数据格式： • void • bool • uint • int • double • QString • QByteArray **提示：**即使自定义小部件方法的返回值的数据格式为“void”，也必须将该值正式分配给一个变量。	

示例

CustomWidget 类声明：

```
class SLESTESTCUSTOMWIDGET_EXPORT SlEsTestCustomWidget  :public QWidget
{
   Q_OBJECT
   public slots:
       void myFunc1(int nValue, const QString& szString, double dValue);
...
```

对话框配置：

```
DEF MyCWVar1 = (W///,"slestestcustomwidget.SlEsTestCustomWidget")
DEF MyStringVar1 = (S)
DEF MyRealVar = (R)

PRESS(VS3)
   REG[9] = CallCWMethod("MyCWVar1", "myFunc1", 1+7, MyStringVar1, sin(MyRealVar) -
8)
END_PRESS
```

提示

自定义小部件必须执行“serialize”方法。采用该方法可以将自定义小部件的内部数据写入指定的文件中或重新创建。当打开的“ ”屏幕切换至其他操作区域时然后再次返回至该屏幕时，该方法非常有必要。否则的话，内部数据在重新显示时会丢失。

句法：	public slots: bool **serialize**(const QString& szFilePath, bool bIsStoring);	
说明：	内部数据和状态从文件中读出或写入文件中	
参数：	szFilePath	从其中读出或写入自定义小部件的内部数据和状态、带有完整路径说明的文件名称。 视情况而定，自定义小部件需要自行创建该文件。
	bIsStoring	TRUE = 写入 FALSE = 读出

示例

```
bool SlEsTestCustomWidget::serialize(const QString& szFilePath, bool bIsStoring)
{
   QFileInfo fi(szFilePath);
   bool bReturn = false;
```

```
bool SlEsTestCustomWidget::serialize(const QString& szFilePath, bool bIsStoring)
   QDir dir;
   if (dir.mkpath(fi.canonicalPath()))
   {
      QFile fileData(szFilePath);
      QIODevice::OpenMode mode;

      if (bIsStoring)
      {
         mode = QIODevice::WriteOnly;
      }
      else
      {
         mode = QIODevice::ReadOnly;
      }

      if (fileData.open(mode))
      {
         QDataStream streamData;
         streamData.setDevice(&fileData);

         if (bIsStoring)
         {
            streamData << m_nDataCount << m_dValueX;
         }
         else
         {
            streamData >> m_nDataCount >> m_dValueX;
         }

         streamData.setDevice(0);
            fileData.flush();
            fileData.close();
         bReturn = true;
      }
   }
   return bReturn;
}
```

8.4.5.3 响应一个自定义小部件信号

说明

在 Run MyScreens 中，可以响应一个特定的自定义小部件信号 (invokeSub()) 并借此调用一个子程序 (SUB)。

传输值（自定义小部件信号 -> SUB）时，有 10 个在配置中可与寄存器 (REG) 进行比较的全局变量（即所谓的 SIGARG）可用。系统会保存通过自定义小部件信号传输的值。

支持以下传输参数数据格式：

- bool
- uint
- int
- double
- QString
- QByteArray

编程

调用子程序：

句法：	`void invokeSub(const QString& rszSignalName, const QVariantList& rvntList);`	
说明：	自定义小部件信号，通过其调用一个 Run MyScreens 子程序。	
参数：	rszSignalName	待调用的 Run MyScreens 子程序名称
	rvntList	QVariantList Array，用于传输保存在全局参数 SIGARG 中的参数，可在配置中使用。 最大数量：10 个元素 不支持的数据格式：见上 **提示：**传输参数将始终传输 “ByVal”，即始终只传输值，不传输变量。

待调用的子程序：

句法：	SUB(on_<变量名称>_<信号名称>) … END_SUB	
说明：	响应一个自定义小部件信号	
参数：	变量名称	分配到一个自定义小部件的对话框变量的名称。
	信号名称	自定义小部件信号的名称
	SIGARG 0 - 9	自定义小部件方法的传输参数。 不支持的数据格式：见上 **说明：**传输参数将始终传输 “ByVal”，即始终只传输值，不传输变量。

示例

CustomWidget 类声明：

```
class SLESTESTCUSTOMWIDGET_EXPORT SlEsTestCustomWidget :public QWidget
{
   Q_OBJECT
signals:
   void invokeSub(const QString& szSubName, const QVariantList& vntList);
…
}
```

CustomWidget 类：

```

QVariantList vntList;
vntList << 123.456;
emit invokeSub("MySub", vntList);
```

对话框配置：

```
DEF MyCWVar1 = (W///,"slestestcustomwidget.SlEsTestCustomWidget")
SUB(on_MyCWVar1_MySub)
   DEBUG("SUB(on_MyCWVar1_MySub) was called with parameter: """ << SIGARG[0] <<
""""")
END_SUB
```

结果 “easyscreen_log.txt”:

```
[10:22:40.445] DEBUG:SUB(on_MyCWVar1_MySub) was called with parameter:"123.456"
```

8.5 SlEsGraphCustomWidget

8.5.1 SlEsGraphCustomWidget

通用说明

借助于 SlEsGraphCustomWidget 可以显示支持点（例如：测量值，过程）上的几何对象（点、线、矩形、圆角矩形、椭圆、圆弧、文本）和曲线。

对象建立在一个或多个所谓的轮廓中。这些对象可以单独或组合显示或清空、选择或删除。

可以借助当前所选轮廓的功能添加对象并以该顺序显示。如果要显示特定颜色的对象，则添加前必须设置相应的字符颜色。所有以后添加的对象均显示该字符颜色。除了字符颜色外还能影响字符宽度和字符样式。在封闭式矩形、圆角矩形和椭圆中在添加对象前可以设置填充色。

每种轮廓都能采用不同的程序：

- 所有对象类型正常显示。
- 点可以连接至所谓的多线段上并以曲线/图表的格式显示。
- 点、线或圆弧到 X 轴之间的平面可使用相应的填充色填色。例如可以使用该方法显示车削时的剩余材料。
  如果在该模式下的点显示为多线段，则曲线以下至 X 轴的整个平面填满填充色。点、线和圆弧可以处于任意四象限中。

采用特殊的光标可以“顺序控制”一个或多个轮廓。此时设置光标位于轮廓的首个或最后一个点对象上或从当前光标位置延伸至轮廓内部，向前或向后运动点对象。

需要时可以显示带有单独比例缩放的第二个 Y 轴（右）。该轴通过偏移和系数与首个 Y 轴（左）进行耦合。

通过查找功能可以根据指定的 X 坐标查找先前添加至轮廓中的点，需要时也可以在此设置光标。

也可以通过可设定的尺寸将轮廓设置为环形缓冲器。

借助串行化可将 SlEsGraphCustomWidget 的当前状态以二进制格式保存在文件中，也可重新创建。

SlEsGraphCustomWidget 可通过“Pan”（视图移动）和“Pinch”/“Spread”（视图放大/缩小）进行操作。

通过鼠标操作可以移动（左键+移动键）和缩放（鼠标滚轮）视图。

由于性能原因，无法自动更新显示。根据应用情况可通过调用相应的功能触发更新。

示例

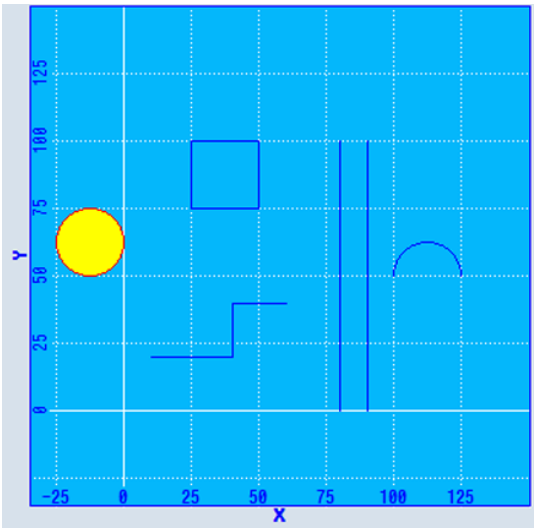


图 8-1 SlEsGraphCustomWidget - 示例

```
//M(MyGraphSampleMask/"SlEsGraphCustomWidget Sample")
DEF MyGraphVar = (W///,"slesgraphcustomwidget.SlEsGraphCustomWidget"/////10,10,340,340/0,0,0,0)
VS1=("Add objects",,se1)
LOAD
   WRITECWPROPERTY("MyGraphVar", "AxisNameX", "X")
   WRITECWPROPERTY("MyGraphVar", "AxisNameY", "Y")
   WRITECWPROPERTY("MyGraphVar", "ScaleTextOrientationYAxis", 2)
   WRITECWPROPERTY("MyGraphVar", "KeepAspectRatio", TRUE)

   REG[0]= CALLCWMETHOD("MyGraphVar", "addContour", "MyContour", TRUE)
   REG[0]= CALLCWMETHOD("MyGraphVar", "showContour", "MyContour")
   REG[0]= CALLCWMETHOD("MyGraphVar", "setView", -35, -35, 150, 150)
END_LOAD

PRESS(VS1)
   REG[0]= CALLCWMETHOD("MyGraphVar", "addLine", 10, 20, 40, 20)
   REG[0]= CALLCWMETHOD("MyGraphVar", "addLine", 40, 20, 40, 40)
   REG[0]= CALLCWMETHOD("MyGraphVar", "addLine", 40, 40, 60, 40)
   REG[0]= CALLCWMETHOD("MyGraphVar", "addLine", 80, 0, 80, 100)
   REG[0]= CALLCWMETHOD("MyGraphVar", "addLine", 90, 0, 90, 100)
   REG[0]= CALLCWMETHOD("MyGraphVar", "addRect", 25, 100, 50, 75)
   REG[0]= CALLCWMETHOD("MyGraphVar", "setPenColor", "#ff0000");red
   REG[0]= CALLCWMETHOD("MyGraphVar", "setFillColor", "#ffff00");yellow
   REG[0]= CALLCWMETHOD("MyGraphVar", "addCircle", -12.5, 62.5, 12.5)
   REG[0]= CALLCWMETHOD("MyGraphVar", "setFillColor");off/default
   REG[0]= CALLCWMETHOD("MyGraphVar", "setPenColor");off/default
```

```
//M(MyGraphSampleMask/"SlEsGraphCustomWidget Sample")
   REG[0]= CALLCWMETHOD("MyGraphVar", "addArc", 100, 37.5, 125, 62.5, 0, 180)
   REG[0]= CALLCWMETHOD("MyGraphVar", "update")
END_PRESS
```

8.5.2 性能说明

根据所用系统的硬件和基本负载情况，在使用 SIEsGraphCustomWidget 时必须注意以下参考值。除了所用硬件和基本负载外，值会根据 SIEsGraphCustomWidget 的配置而变化。

要注意该参考值不能影响到整个系统的响应能力和稳定性。

- 同一时间点能一同配置的 SIEsGraphCustomWidget 数量（例如：最大 1）
- 配置的轮廓数量（例如：最大 6）
- 每个轮廓中可配置的图形数量（例如：最大 1000）
- 要求更新的频率（例如：最大 500 ms）

说明

显示没有实时功能。

8.5.3 读取和写入属性

说明

以下章节中列出的属性可以通过 ReadCWProperty() 功能读取，通过 WriteCWProperty() 功能写入。

示例

读取通过显示变量“MyGraphVar”所连的 SlEsGraphCustomWidget 的属性“CursorX”。结果写入寄存器 0 中。

```
REG[0] = ReadCWProperty("MyGraphVar", "CursorX")
```

将值“MyFirstContour”写入通过显示变量“MyGraphVar”所连的 SlEsGraphCustomWidget 的属性“SelectedContour”中。

```
WriteCWProperty("MyGraphVar ", "SelectedContour",
"MyFirstContour")
```

8.5.4 属性

概述

属性	说明
AxisNameX	轴名称 X 轴
AxisNameY	轴名称 Y 轴
AxisNameY2	轴名称第二 Y 轴（右）
AxisY2Visible	显示/隐藏第二 Y 轴（右）
AxisY2Offset	第二 Y 轴（右）偏移
AxisY2Factor	第二 Y 轴（右）系数
ScaleTextEmbedded	比例缩放文本的定位
ScaleTextOrientationYAxis	Y 轴比例缩放文本校准
KeepAspectRatio	保持页面比例
SelectedContour	当前所选轮廓的名称
BackColor	背景色
ChartBackColor	字符平面的背景色
ForeColor	文本的前景色和缺省字符颜色
ForeColorY2	第二 Y 轴（右）的文本前景色
GridColor	栅格线颜色
GridColorY2	第二 Y 轴（右）水平栅格线的颜色
CursorColor	光标十字线的颜色
ShowCursor	显示/隐藏光标
CursorX	光标 X 坐标
CursorY	光标 Y 坐标
CursorY2	基于第二 Y 轴（右）的光标 Y 位置
CursorStyle	光标显示方式
ViewMoveZoomMode	缩放和移动时的特性

AxisNameX – 轴名称 X 轴

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**AxisNameX**") WriteCWProperty(GraphVarName, "**AxisNameX**", Value)	
说明：	读取或设置 X 轴的名称。如果没有给出名称，则需充分利用可用的字符平面。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（QString）
	Value	待设置的值（QString）

AxisNameY – 轴名称 Y 轴

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**AxisNameY**") WriteCWProperty(GraphVarName, "**AxisNameY**", Value)	
说明：	读取或设置 Y 轴的名称。如果没有给出名称，则需充分利用可用的字符平面。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（QString）
	Value	待设置的值（QString）

AxisY2Visible - 显示/隐藏第二 Y 轴（右）

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**AxisY2Visible**") WriteCWProperty(GraphVarName, "**AxisY2Visible**", Value)	
说明：	显示/隐藏第二 Y 轴（右）	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE 或 FALSE

AxisY2Offset - 第二 Y 轴（右）偏移

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**AxisY2Offset**") WriteCWProperty(GraphVarName, "**AxisY2Offset**", Value)	
说明：	第二 Y 轴（右）基于第一 Y 轴（左）的偏移。 参见 Property AxisY2Factor 示例。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（double）
	Value	待设置的值（double）

AxisY2Factor – 第二 Y 轴（右）系数

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**AxisY2Factor**") WriteCWProperty(GraphVarName, "**AxisY2Factor**", Value)	
说明：	第二 Y 轴（右）基于第一 Y 轴（左）的系数。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（double）
	Value	待设置的值（double）

与“AxisY2Offset”属性一起时可以以单独的缩放比例显示第二 Y 轴（右）。比例通过偏移和系数与首个 Y 轴（左）进行耦合。

Y2 转换为 Y 的公式：

Y = Y2 / 系数 - 偏移

Y 转换为 Y2 的公式：

Y2 = (Y + 偏移) * 系数

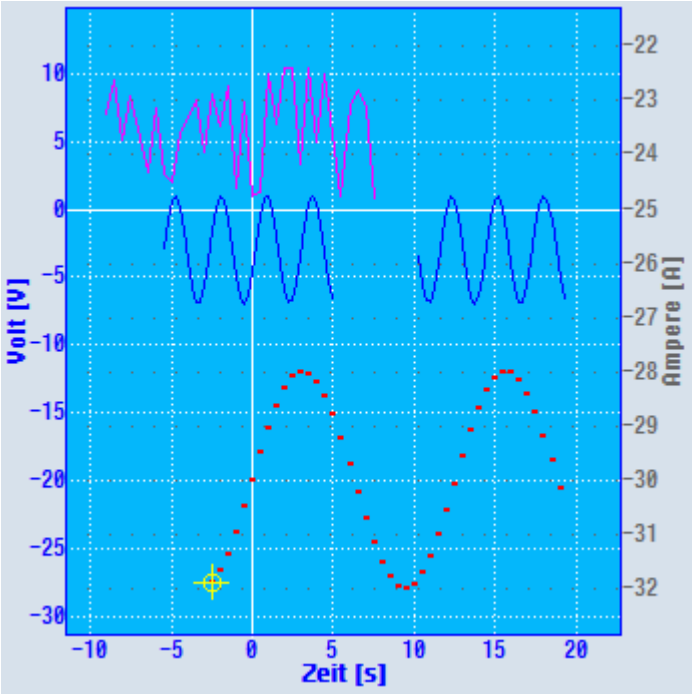


图 8-2　具有单独比例缩放的第二 Y 轴示例

示例

Y：0 到 200 时，Y2 相应为 -25 到 25。

- 偏移：-100
- 系数：0.25

计算示例：

Y2 = -30

Y = -30 / 0.25 – (-100) = -20

计算示例：

Y = 0

Y2 = (0 + (-100)) * 0.25 = -25

两个坐标系统上的 X 轴都是一样的。

可通过 setForeColorY2() 和 setGridColorY2() 设置颜色（参见颜色）。

通过 setAxisNameY2() 设置轴的标记（参见轴名称）。

ScaleTextEmbedded - 比例缩放文本的定位

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**ScaleTextEmbedded**") WriteCWProperty(GraphVarName, "**ScaleTextEmbedded**", Value)	
说明：	通过该属性可以确定比例缩放文本是定位于字符平面内部还是外部。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE 或 FALSE

示例

字符区外的比例缩放文本：

ScaleTextEmbedded = FALSE

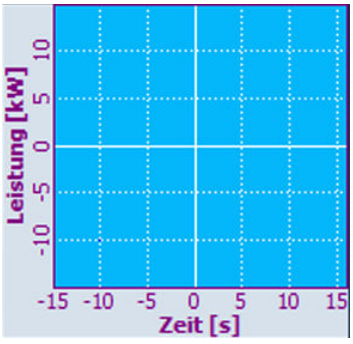


图 8-3　字符区外的比例缩放文本

字符区内的比例缩放文本：

ScaleTextEmbedded = TRUE

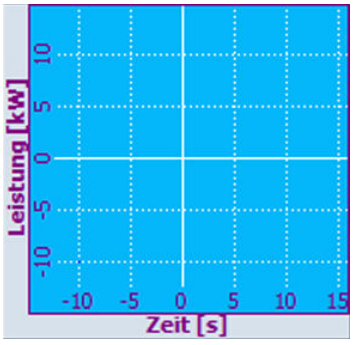


图 8-4　字符区内的比例缩放文本

ScaleTextOrientationYAxis - Y 轴比例缩放文本校准

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**ScaleTextOrientationYAxis**") WriteCWProperty(GraphVarName, "**ScaleTextOrientationYAxis**", Value)	
说明：	通过该功能垂直校准 Y 轴的比例缩放文本。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（int）
	Value	待设置的值（int）：1 (= 水平) 或 2 (= 垂直)

示例

字符区内的比例缩放文本：

- ScaleTextEmbedded = TRUE

水平文本校准：

- ScaleTextOrientationYAxis = 1

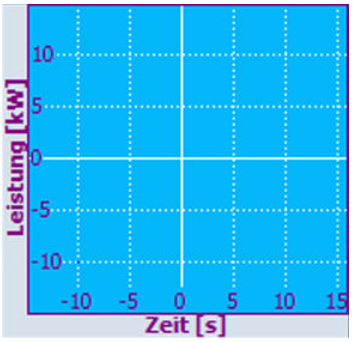


图 8-5　字符区内的比例缩放文本，水平文本校准

垂直文本校准：

- ScaleTextOrientationYAxis = 2

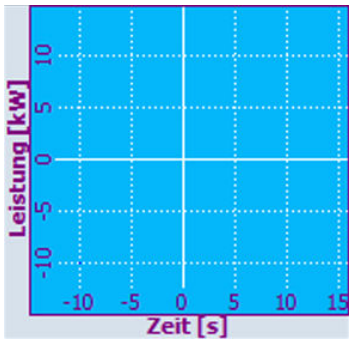


图 8-6　字符区内的比例缩放文本，垂直文本校准

字符区外的比例缩放文本：

- ScaleTextEmbedded = FALSE

水平文本校准：

- ScaleTextOrientationYAxis = 1

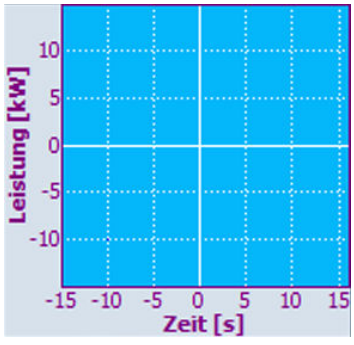


图 8-7　字符区外的比例缩放文本，水平文本校准

垂直文本校准：

- ScaleTextOrientationYAxis = 2

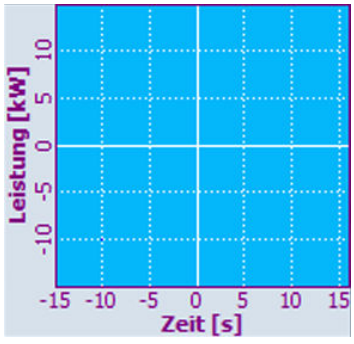


图 8-8　字符区外的比例缩放文本，垂直文本校准

KeepAspectRatio – 保持页面比例

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**KeepAspectRatio**") WriteCWProperty(GraphVarName, "**KeepAspectRatio**", Value)	
说明：	通过该属性确定，由 setView() 确定的视图是否自动校准、调整或扩展，X 轴与 Y 轴的页面比例是否一直相同。该特性在显示几何数据时尤其重要。这样一来，圆弧、正方形以及其他和椭圆或矩形便不会扭曲显示。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Return Value	待设置的值（bool）：TRUE 或 FALSE

示例

```
setView(-15, -15, 15, 15)
setFillColor("#A0A0A4")
addCircle(0, 0, 5)
KeepAspectRatio = TRUE
```

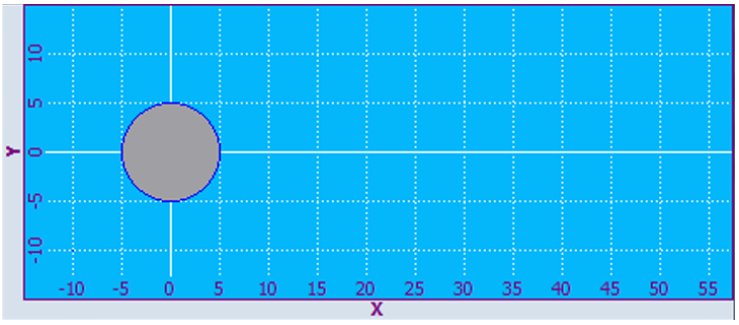


图 8-9　X 轴与 Y 轴的页面比例保持相同

```
KeepAspectRatio = FALSE
```

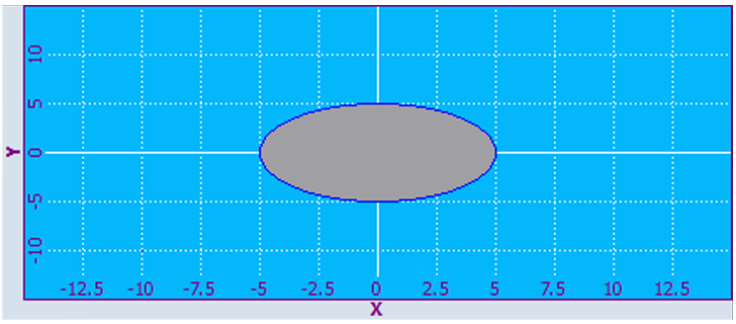


图 8-10　X 轴与 Y 轴的页面比例扭曲

SelectedContour – 当前所选轮廓的名称

句法：	ReturnValue = ReadCWProperty(GraphVarName, " **SelectedContour**") WriteCWProperty(GraphVarName, " **SelectedContour**", Value)	
说明：	读取或设置当前轮廓的选择。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（QString）
	Value	待设置的值（QString）

说明

为了能在轮廓上进行操作，例如：添加图形对象，必须提前选择。

BackColor – 背景色

句法：	ReturnValue = ReadCWProperty(GraphVarName, " **BackColor**") WriteCWProperty(GraphVarName, " **BackColor**", Value)	
说明：	轴标记的背景颜色，取决于属性 ScaleTextInsideChartArea 和比例缩放文本。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性颜色（QString）
	Value	待设置的值（QString），作为“#RRGGBB”格式的 RGB 值，例如“#04B7FB”

ChartBackColor – 字符平面的背景色

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**ChartBackColor**") WriteCWProperty(GraphVarName, "**ChartBackColor**", Value)	
说明：	字符区的背景颜色。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性颜色（QString）
	Value	待设置的值（QString），作为“#RRGGBB”格式的 RGB 值，例如“#04B7FB”

ForeColor - 标记的前景颜色和缺省字符颜色

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**ForeColor**") WriteCWProperty(GraphVarName, "**ForeColor**", Value)	
说明：	文本的前景颜色和缺省字符颜色	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性颜色（QString）
	Value	待设置的值（QString），作为“#RRGGBB”格式的 RGB 值，例如“#04B7FB”

ForeColorY2 - 第二 Y 轴（右）的标记前景色

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**ForeColorY2**") WriteCWProperty(GraphVarName, "**ForeColorY2**", Value)	
说明：	第二 Y 轴（右）的文本前景色。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性颜色（QString）
	Value	待设置的值（QString），作为“#RRGGBB”格式的 RGB 值，例如“#04B7FB”

GridColor - 栅格线颜色

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**GridColor**") WriteCWProperty(GraphVarName, "**GridColor**", Value)	
说明：	栅格线颜色。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性颜色（QString）
	Value	待设置的值（QString），作为“#RRGGBB”格式的 RGB 值，例如“#04B7FB”

GridColorY2 - 第二 Y 轴（右）水平栅格线的颜色

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**GridColorY2**") WriteCWProperty(GraphVarName, "**GridColorY2**", Value)	
说明：	第二 Y 轴（右）水平栅格线的颜色	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性颜色（QString）
	Value	待设置的值（QString），作为“#RRGGBB”格式的 RGB 值，例如“#04B7FB”

CursorColor - 光标颜色

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**CursorColor**") WriteCWProperty(GraphVarName, "**CursorColor**", Value)	
说明：	光标颜色。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性颜色（QString）
	Value	待设置的值（QString），作为“#RRGGBB”格式的 RGB 值，例如“#04B7FB”

ShowCursor – 显示/隐藏光标

句法：	ReturnValue = ReadCWProperty(GraphVarName, " **ShowCursor**") WriteCWProperty(GraphVarName, " **ShowCursor**", Value)	
说明：	显示/隐藏光标。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE 或 FALSE

说明

该功能会自动更新显示。

CursorX – 光标 X 坐标

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**CursorX**") WriteCWProperty(GraphVarName, "**CursorX**", Value)	
说明：	光标的 X 坐标。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（double）
	Value	待设置的值（double）

说明

该功能会自动更新显示。

CursorY – 光标 Y 坐标

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**CursorY**") WriteCWProperty(GraphVarName, "**CursorY**", Value)	
说明：	光标的 Y 位置。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（double）
	Value	待设置的值（double）

说明

该功能会自动更新显示。

CursorY2 – 基于第二 Y 轴（右）的光标 Y 位置

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**CursorY2**") WriteCWProperty(GraphVarName, "**CursorY2**", Value)	
说明：	基于第二 Y 轴（右）的光标的 Y 位置。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（double）
	Value	待设置的值（double）

说明

该功能会自动更新显示。

CursorStyle – 光标显示方式

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**CursorStyle**") WriteCWProperty(GraphVarName, "**CursorStyle**", Value)	
说明：	光标显示方式。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（int）
	Value	待设置的值（int）： 0 (= 十字) 1 (= 十字线) 2 (= 垂直线) 3 (= 水平线) 4 (= 水平和垂直线))

ViewMoveZoomMode – 用手势缩放和移动时的特性

句法：	ReturnValue = ReadCWProperty(GraphVarName, "**ViewMoveZoomMode**") WriteCWProperty(GraphVarName, "**ViewMoveZoomMode**", Value)	
说明：	可通过手势修改所显示的 SlEsGraphCustomWidget 视图。通过该属性确定允许哪种方式。例如可在特定应用情况下确定仅允许水平移动或缩放视图（测量值过程显示时）。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	已读取的属性值（int）
	Value	待设置的值（int）： 0 (= 关闭) 1 (= 仅水平) 2 (= 仅垂直) 3 (= 水平和垂直)

8.5.5 功能

可通过 CallCWMethod() 调用以下功能。

示例

设置通过显示变量"MyGraphVar"连接的 SlEsGraphCustomWidget 的视图。

结果写入寄存器 0 中。

```
REG[0] = CallCWMethod("MyGraphVar", "setView", -8, -5, 11- 20)
```

概述

功能	说明
setView	设置坐标系
setMaxContourObjects	设置轮廓的最大对象数量（环形缓冲器）
addContour	添加轮廓
showContour	显示轮廓
hideContour	隐藏轮廓

功能	说明
hideAllContours	隐藏所有轮廓
removeContour	删除轮廓
clearContour	删除轮廓中的图形对象
fitViewToContours	视图自动适配
fitViewToContour	视图自动适配
findX	在轮廓中查找
setPolylineMode	显示轮廓的点为多线段/曲线
setIntegralFillMode	显示轮廓的点为多线段/曲线
repaint, update	更新视图
addPoint	添加轮廓中的点
addLine	添加轮廓中的线
addRect	添加轮廓中的矩形
addRoundedRect	添加轮廓中的圆角矩形
addEllipse	添加轮廓中的椭圆
addCircle	添加轮廓中的圆
addArc	添加轮廓中的圆弧
addText	添加轮廓中的文本
setPenWidth	设置字符宽度
setPenStyle	设置字符样式
setPenColor	设置字符颜色
setFillColor	设置填充色
setCursorPosition	光标定位
setCursorPositionY2	基于第二 Y 轴（右）定位光标
setCursorOnContour	定位轮廓上的光标
moveCursorOnContourBegin	定位轮廓上第一个图形对象的光标
moveCursorOnContourEnd	定位轮廓上最后一个图形对象的光标
moveCursorOnContourNext	定位轮廓上下一个图形对象的光标
serialize	保存/恢复当前状态

setView – 设置坐标系

句法：	ReturnValue = CallCWMethod(GraphVarName, "**setView**", x1, y1, x2, y2)	
说明：	通过该功能确定坐标系的大小，在该坐标系要在 SlEsGraphCustomWidget 内显示。另见 KeepAspectRatio 属性。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	x1	左边沿（double）
	y1	上边沿（double）
	x2	右边沿（double）
	y2	下边沿（double）

说明

该功能会自动更新显示。

示例

```
setView(-8, -5, 11, 20)
AxisNameX = "X"
AxisNameY = "Y"
ScaleTextOrientationYAxis = 1
ScaleTextEmbedded = false
KeepAspectRatio = false
update
```

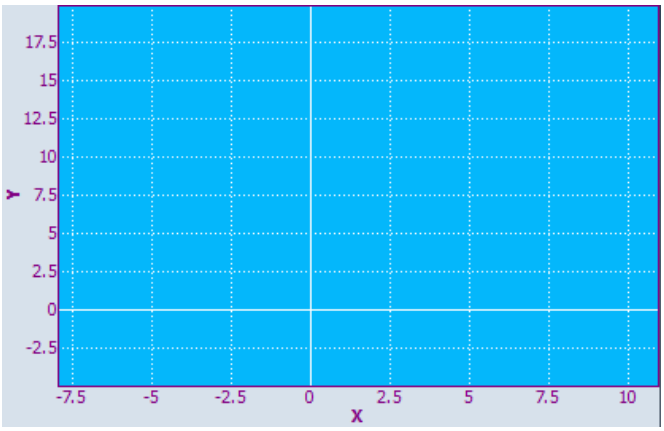


图 8-11　示例 - setView

setMaxContourObjects - 设置轮廓的最大对象数量（环形缓冲器）

句法：	ReturnValue = CallCWMethod(GraphVarName, "**setMaxContourObjects**", Value) ReturnValue = CallCWMethod(GraphVarName, "**setMaxContourObjects**", Value, ContourName)	
说明：	通过该功能确定轮廓环形缓冲器的大小。如果轮廓添加的对象超出了该限值，则会删除最旧的对象。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	Value	图形对象的最大数目（unit）
	ContourName	轮廓名称（QString）。 如果没有指定轮廓，则自动调用当前所选的轮廓。

说明

该功能会自动更新显示。

addContour – 添加轮廓

句法：	ReturnValue = CallCWMethod(GraphVarName, "**addContour**", ContourName) ReturnValue = CallCWMethod(GraphVarName, "**addContour**", ContourName, Selected) ReturnValue = CallCWMethod(GraphVarName, "**addContour**", ContourName, Selected, AssignedY2)	
说明：	使用该功能创建新轮廓。 或者也可以分配第二 Y 轴（右）的坐标系轮廓。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	ContourName	轮廓名称（QString）。 如果没有指定轮廓，则自动调用当前所选的轮廓。
	Selected	选择轮廓（bool）：TRUE 或 FALSE
	AssignedY2	应分配至第二 Y 轴（右）的轮廓 （bool）：TRUE 或 FALSE

showContour – 显示轮廓

句法：	ReturnValue = CallCWMethod(GraphVarName, "**showContour**") ReturnValue = CallCWMethod(GraphVarName, "**showContour**", ContourName)	
说明：	该功能用于显示轮廓。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	ContourName	轮廓名称（QString）。 如果没有指定轮廓，则自动调用当前所选的轮廓。

hideContour – 隐藏轮廓

句法：	ReturnValue = CallCWMethod(GraphVarName, "**hideContour**") ReturnValue = CallCWMethod(GraphVarName, "**hideContour**", ContourName)	
说明：	该功能用于隐藏轮廓。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	ContourName	轮廓名称（QString）。 如果没有指定轮廓，则自动调用当前所选的轮廓。

showAllContours - 显示所有轮廓

句法：	ReturnValue = CallCWMethod(GraphVarName, "**showAllContours**")	
说明：	该功能用于显示所有轮廓。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功

hideAllContours – 隐藏所有轮廓

句法：	ReturnValue = CallCWMethod(GraphVarName, "**hideAllContours**")	
说明：	该功能用于隐藏所有轮廓。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功

removeContour – 隐藏轮廓

句法：	ReturnValue = CallCWMethod(GraphVarName, "**removeContour**") ReturnValue = CallCWMethod(GraphVarName, "**removeContour**", ContourName)
说明：	删除轮廓。

参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	ContourName	轮廓名称（QString）。 如果没有指定轮廓，则自动调用当前所选的轮廓。

clearContour – 删除轮廓中的图形对象

句法：	ReturnValue = CallCWMethod(GraphVarName, "**clearContour**") ReturnValue = CallCWMethod(GraphVarName, "**clearContour**", ContourName)	
说明：	删除轮廓中包含的所有图形对象。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	ContourName	轮廓名称（QString）。 如果没有指定轮廓，则自动调用当前所选的轮廓。

fitViewToContours – 视图自动适配

句法：	ReturnValue = CallCWMethod(GraphVarName, "**fitViewToContours**") ReturnValue = CallCWMethod(GraphVarName, "**fitViewToContours**", OnlyVisible)	
说明：	通过该功能视图自动与轮廓进行适配。根据传输参数确定，是否只显示可显示的轮廓还是所有轮廓。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	OnlyVisible	待设置的值（bool）：TRUE 或 FALSE

fitViewToContour – 视图自动适配

句法：	ReturnValue = CallCWMethod(GraphVarName, "**fitViewToContour**", ContourName)
说明：	通过该功能视图自动适配，仅显示一个特定的轮廓。

参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	ContourName	轮廓名称（QString）

findX – 在轮廓中查找

句法：	ReturnValue = CallCWMethod(GraphVarName, "**findX**", x) ReturnValue = CallCWMethod(GraphVarName, "**findX**", x, ContourName) ReturnValue = CallCWMethod(GraphVarName, "**findX**", x, ContourName, SetCursor)	
说明：	通过该功能可在轮廓中通过确定的 X 坐标查找之前定义的点。获得 Y 坐标作为结果。或者也可以在该点上同时定位光标。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	Errorcode (QString):成功时提供相应的 Y 值
	x	待查找的 X 值（double）
	ContourName	轮廓名称（QString）。 如果没有指定轮廓，则自动调用当前所选的轮廓。
	SetCursor	设置光标（bool）：TRUE 或 FALSE

说明

通过该功能设置光标时，显示自动更新。

setPolylineMode – 显示轮廓的点为多线段/曲线

句法：	ReturnValue = CallCWMethod(GraphVarName, "**setPolylineMode**", SetPoly)	
说明：	当前轮廓上所有在该功能调用后添加的点连接至多线段/曲线。该功能是轮廓专用的可逐段打开和关闭。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	SetPoly	PolylineMode (bool):TRUE = 开

示例

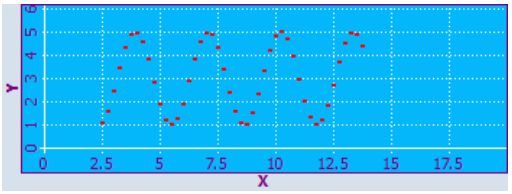


图 8-12　示例 - PolylineMode:关

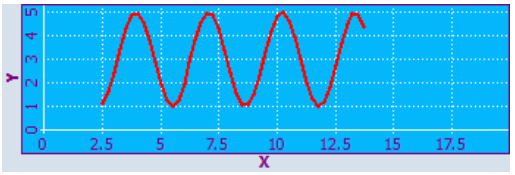


图 8-13　示例 - PolylineMode:开

setIntegralFillMode – 给点、线和圆弧及 X 轴之间的平面填色

句法：	ReturnValue = CallCWMethod(GraphVarName, "**setIntegralFillMode**", SetIntFill)	
说明：	点、线和圆弧及 X 轴间的平面用当前填充色进行填充（不管点是 +Y 还是 –Y）。 该功能是轮廓专用的可逐段打开和关闭。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	SetIntFill	IntegralFillMode (bool):TRUE = 开

示例

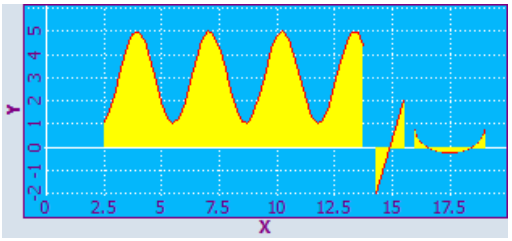


图 8-14　示例 - setIntegralFillMode:开

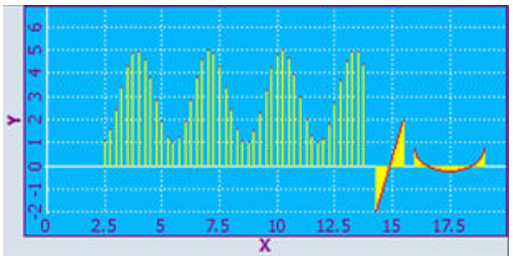


图 8-15　示例 - setIntegralFillMode:关

repaint, update – 更新视图

句法：	ReturnValue = CallCWMethod(GraphVarName, "**repaint**") ReturnValue = CallCWMethod(GraphVarName, "**update**")	
说明：	通过该功能可以手动更新显示。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功

- update()
  该功能可以更新视图，前提条件是未取消小部件的更新且小部件未隐藏。优化该功能，确保保留应用的性能且视图不会因频繁更新而闪烁。
- repaint()
  调用该功能后直接更新视图。因此只有在迫切需要立即更新时（例如：动画）才使用 repaint 功能。

建议始终与 update() 功能一同处理。SlEsGraphCustomWidget 内部始终与 update() 功能一同处理，例如：setView() 上。

addPoint – 添加轮廓中的点

句法：	ReturnValue = CallCWMethod(GraphVarName, "**addPoint**", x, y) ReturnValue = CallCWMethod(GraphVarName, "**addPoint**", x, y, DummyPoint)
说明：	添加当前所选轮廓上的点。此外还可以设置假点，该点虽然不会显示，但仍可通过光标定位。

参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	x	x 坐标（double）
	y	y 坐标（double）
	DummyPoint	不可见的点（bool）:TRUE = 是

addLine – 添加轮廓上的线

句法：	ReturnValue = CallCWMethod(GraphVarName, "**addLine**", x1, y1, x2, y2)	
说明：	添加当前所选轮廓上的线	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	x1	起点的 x 坐标（double）
	y1	起点的 y 坐标（double）
	x2	终点的 x 坐标（double）
	y2	终点的 y 坐标（double）

addRect – 添加轮廓上的矩形

句法：	ReturnValue = CallCWMethod(GraphVarName, "**addRect**", x1, y1, x2, y2)	
说明：	添加当前所选轮廓上的矩形	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	x1	左边沿（double）
	y1	左边沿（double）
	x2	右边沿（double）
	y2	下边沿（double）

addRoundedRect – 添加轮廓中的圆角矩形

句法：	ReturnValue = CallCWMethod(GraphVarName, "**addRoundedRect**", x1, y1, x2, y2, r)	
说明：	添加当前所选轮廓上的圆角矩形	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	x1	左边沿（double）
	y1	上边沿（double）
	x2	上边沿（double）
	y2	上边沿（double）
	r	半径（double）

addEllipse – 添加轮廓中的椭圆

句法：	ReturnValue = CallCWMethod(GraphVarName, "**addEllipse**", x1, y1, x2, y2, 半径)	
说明：	添加当前所选轮廓上的椭圆。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	x1	左边沿（double）
	y1	上边沿（double）
	x2	右边沿（double）
	y2	下边沿（double）
	半径	半径（double）

addCircle – 添加轮廓中的圆

句法：	ReturnValue = CallCWMethod(GraphVarName, "**addCircle**", x, y, 半径)
说明：	添加当前所选轮廓上的圆。

参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	x	左边沿（double）
	y	上边沿（double）
	半径	半径（double）

addArc – 添加轮廓中的圆弧

句法：	ReturnValue = CallCWMethod(GraphVarName, "**addArc**", x1, y1, x2, y2, StartAngle, SpanAngle	
说明：	添加当前所选轮廓上的圆弧。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	x1	左边沿（double）
	y1	上边沿（double）
	x2	右边沿（double）
	y2	下边沿（double）
	StartAngle	起始角，单位：度（double）
	SpanAngle	张角，单位：度（double）

addText –添加轮廓中的文本

句法：	ReturnValue = CallCWMethod(GraphVarName, "**addText**", x, y, Text)	
说明：	添加当前所选轮廓上的文本。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	x	左边沿（double）
	y	上边沿（double）
	Text	文本（QString）

setPenWidth – 设置字符宽度

句法：	ReturnValue = CallCWMethod(GraphVarName, "**setPenWidth**") ReturnValue = CallCWMethod(GraphVarName, "**setPenWidth**", width)	
说明：	确定从功能调用时间点开始随后在当前轮廓中添加的对象的字符宽度。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	width	字符宽度（double）。 如果没有指定字符宽度，则设置缺省字符宽度。

setPenStyle – 设置字符样式

句法：	ReturnValue = CallCWMethod(GraphVarName, "**setPenStyle**") ReturnValue = CallCWMethod(GraphVarName, "**setPenStyle**", style)	
说明：	确定从功能调用时间点开始随后在当前轮廓中添加的对象的字符式样。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	style	1 :实线 (___) 2 :虚线 (_ _) 3 :点线 (...) 4 :线点线 (_._) 5 :线点点 (_..)

setPenColor – 设置字符颜色

句法：	ReturnValue = CallCWMethod(GraphVarName, "**setPenColor**") ReturnValue = CallCWMethod(GraphVarName, "**setPenColor**", Color)	
说明：	确定从功能调用时间点开始随后在当前轮廓中添加的对象的字符颜色。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	Color	待设置的值（QString），作为“#RRGGBB”格式的 RGB 值，例如“#04B7FB”

setFillColor – 设置填充色

句法：	ReturnValue = CallCWMethod(GraphVarName, "**setFillColor**") ReturnValue = CallCWMethod(GraphVarName, "**setFillColor**", Color)	
说明：	确定从功能调用时间点开始随后在当前轮廓中添加的对象的填充色。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	Color	待设置的值（QString），作为“#RRGGBB”格式的 RGB 值，例如“#04B7FB”

setCursorPosition – 光标定位

句法：	ReturnValue = CallCWMethod(GraphVarName, "**setCursorPosition**", x, y)	
说明：	在规定的位置上设置光标。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	x	x 坐标（double）
	y	y 坐标（double）

说明

该功能会自动更新显示。

setCursorPositionY2Cursor – 基于第二 Y 轴（右）定位光标

句法：	ReturnValue = CallCWMethod(GraphVarName, "**setCursorPositionY2**", x, y2)	
说明：	基于第二 Y 轴（右），在规定的位置上设置光标。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	x	x 坐标（double）
	y2	基于第二 Y 轴（右）的 Y 坐标（double）

说明

该功能会自动更新显示。

setCursorOnContour – 定位轮廓上的光标

句法：	ReturnValue = CallCWMethod(GraphVarName, "**setCursorOnContour**") ReturnValue = CallCWMethod(GraphVarName, "**setCursorOnContour**", ContourName)	
说明：	在轮廓内设置最后光标位置上的光标。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	ContourName	轮廓名称（QString）。 如果没有指定轮廓，则自动调用当前所选的轮廓。

说明

该功能会自动更新显示。

示例

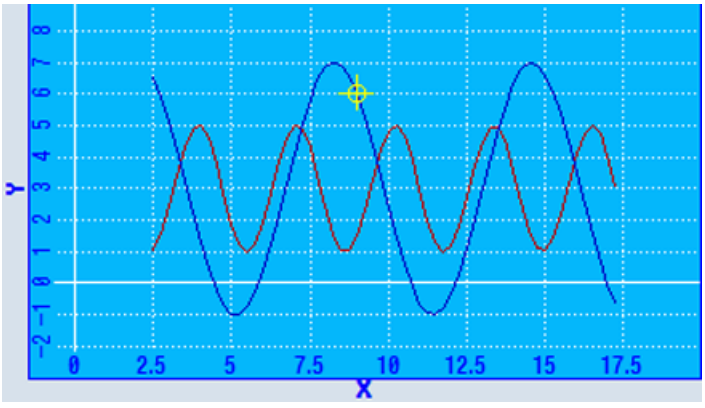


图 8-16　示例 - setCursorOnContour

moveCursorOnContourBegin – 定位轮廓上第一个图形对象的光标

句法：	ReturnValue = CallCWMethod(GraphVarName, "**moveCursorOnContourBegin**") ReturnValue = CallCWMethod(GraphVarName, "**moveCursorOnContourBegin**", ContourName)	
说明：	从轮廓内的当前光标位置开始，可以通过该功能将光标导航至光标的第一个点对象。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	ContourName	轮廓名称（QString）。 如果没有指定轮廓，则自动调用当前所选的轮廓。

说明

该功能会自动更新显示。

moveCursorOnContourEnd – 定位轮廓上最后一个图形对象的光标

句法：	ReturnValue = CallCWMethod(GraphVarName, "**moveCursorOnContourEnd**") ReturnValue = CallCWMethod(GraphVarName, "**moveCursorOnContourEnd**", ContourName)	
说明：	从轮廓内的当前光标位置开始，可以通过该功能将光标导航至光标的最后一个点对象。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	ContourName	轮廓名称（QString）。 如果没有指定轮廓，则自动调用当前所选的轮廓。

说明

该功能会自动更新显示。

moveCursorOnContourNext – 定位轮廓上下一个图形对象的光标

句法：	ReturnValue = CallCWMethod(GraphVarName, "**moveCursorOnContourNext**") ReturnValue = CallCWMethod(GraphVarName, "**moveCursorOnContourNext**", ContourName)	
说明：	从轮廓内的当前光标位置开始，可以通过该功能将光标导航至光标的下一个点对象。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	ContourName	轮廓名称（QString）。 如果没有指定轮廓，则自动调用当前所选的轮廓。

说明

该功能会自动更新显示。

moveCursorOnContourBack – 定位轮廓上前一个图形对象的光标

句法：	ReturnValue = CallCWMethod(GraphVarName, "**moveCursorOnContourBack**") ReturnValue = CallCWMethod(GraphVarName, "**moveCursorOnContourBack**", ContourName)	
说明：	从轮廓内的当前光标位置开始，可以通过该功能将光标导航至光标的前一个点对象。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	ContourName	轮廓名称（QString）。 如果没有指定轮廓，则自动调用当前所选的轮廓。

说明

该功能会自动更新显示。

serialize – 保存/恢复当前状态

句法：	ReturnValue = CallCWMethod(GraphVarName, "**serialize**", FilePath, IsStoring)	
说明：	通过该功能可以在需要时将 SlEsGraphWidget 的当前状态和内容以二进制格式写入文件或数据流中或恢复。	
参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	FilePath	包含文件名称的完整路径（QString）
	IsStoring	保存/恢复（bool）：TRUE = 保存

说明

该功能会自动更新显示。

8.5.6 信号

以下描述的信号 ViewChanged 可在配置时拦截并作出相应的响应。

概述

功能	说明
ViewChanged	视图变化

ViewChanged – 视图变化

句法：	SUB(on_<GraphVarName>**_ViewChanged**) … END_SUB
说明：	如果视图发生变化，则会发送信号“ViewChanged”。可在配置中以特定的 SUB 方法对此进行响应。相应地设置 SIGARG 参数。

参数：	GraphVarName	包含 SlEsGraphCustomWidget 的显示变量的名称
	SIGARG[0]	左边沿（double）
	SIGARG[1]	上边沿（double）
	SIGARG[2]	右边沿（double）
	SIGARG[3]	下边沿（double）

示例

```
DEF MyGraphVar = (W///,"slesgraphcustomwidget.SlEsGraphCustomWidget"/////
10,10,340,340/0,0,0,0)
```

```
SUB(on_MyGraphVar_ViewChanged
     DLGL("Current view: " << SIGARG[0] << ", " << SIGARG[1] << ", " <<   SIGARG[2]
<< ", " << SIGARG[3]
END_SUB
```

8.6 SlEsTouchButton

8.6.1 SlEsTouchButton

常规

借助 Run MyScreens 可以简单的方式创建功能性高要求应用。尤其是在触控操作上，可以配置任意放置的按键（TouchButton）。TouchButton 的配置有以下特性：

- 可以在配置时控制按键“已点击”和“已检查”的两种可能的状态更改并触发相应的操作。
- TouchButton 可通过单触控或多触控、鼠标和键盘操作。
- TouchButton 根据当前分辨率进行显示。此外，这也适用于字体和显示图片。
- TouchButton 有两种显示方式：“Softkey Look&Feel”和“用户专用”。在“Softkey Look&Feel”显示方式中，TouchButton 按照操作软键进行显示。两种显示方式都具有相同的功能（例如：缩放功能）。
- TouchButton 作为 CustomWidget 使用。

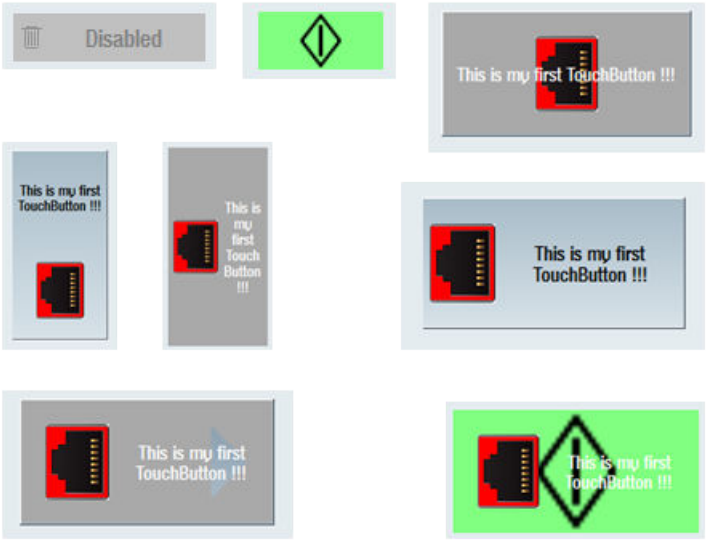


图 8-17 TouchButton 示例

示例

```
//M(MyTBMask/"My CustomWidget TouchButton ...")
DEF MyTB1 = (W///,"slesstdcw.SlEsTouchButton"/////70,20,200,100/0,0,0,0)
LOAD
```

```
//M(MyTBMask/"My CustomWidget TouchButton ...")
   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
```

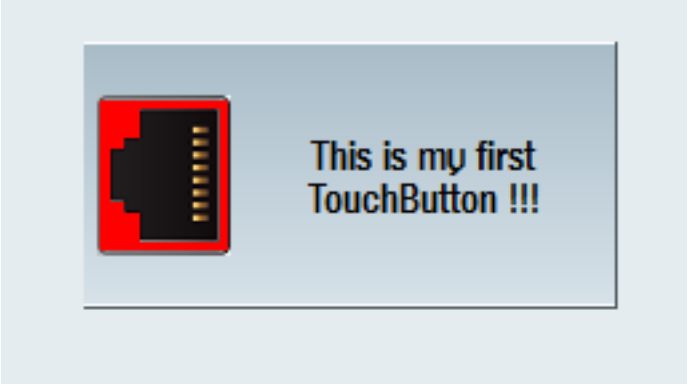


图 8-18 "This is my first TouchButton !!!"示例

8.6.2 读取和写入属性

说明

以下章节中列出的属性可以通过 ReadCWProperty() 功能读取，通过 WriteCWProperty() 功能写入。

示例

读取通过显示变量“MyTouchButton”所连的 SlEsTouchButton 的属性“Text”。结果写入寄存器 0 中。

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

将值“sk_ok.png”写入通过显示变量“MyTouchButton”所连的 SlEsTouchButton 的属性“Picture”中。

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

8.6.3 属性

概述

属性	说明
ButtonStyle	TouchButton 的显示方式
Flat	平面显示或 3D 显示
Enabled	操作性激活/关闭
Checkable	转换功能激活/关闭
Checked	TouchButton 刚好啮合（已检查）
ShowFocusRect	当 TouchButton 具有输入焦点时，显示焦点矩形
Picture	TouchButton 处于未按压、常规状态时的待显示图片
PicturePressed	TouchButton 已按压或已啮合（已检查）显示时的待显示图片
PictureAlignment PictureAlignmentString	图片对齐
ScalePicture	图片已缩放（放大或缩小）
PictureKeepAspectRatio	图片的放大特性
Text	常规显示文本
TextPressed	TouchButton 按压时的显示文本
textAlignment textAlignmentString	文本对齐
TextAlignedToPicture	文本相对于图片对齐 - 激活/关闭
BackgroundPicture	待显示的背景画面
BackgroundPictureAlignment BackgroundPictureAlignmentString	背景画面对齐
ScaleBackgroundPicture	背景画面已缩放（放大或缩小）
BackgroundPictureKeepAspectRatio	图片的放大特性

属性	说明
BackColor	TouchButton 处于未按压、常规状态时的背景色
BackColorChecked	TouchButton 啮合时（已检查）的背景色
BackColorPressed	暂时按压 TouchButton 时的背景色
BackColorDisabled	TouchButton 不可操作时的背景色
TextColor	TouchButton 处于未按压、常规状态时的文本色
TextColorChecked	TouchButton 啮合时（已检查）的文本色
TextColorPressed	暂时按压 TouchButton 时的文本色
TextColorDisabled	TouchButton 不可操作时的文本色

ButtonStyle – 显示方式

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**ButtonStyle**") WriteCWProperty(TouchButtonVarName, "**ButtonStyle**", Value)	
说明：	读取/设置 TouchButton 的显示方式。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（int）
	Value	待设置的值（int） 0 = Softkey Look&Feel（缺省设置） 1 = 用户专用

采用“0 = Softkey Look&Feel”设置时，TouchButton 显示、表现与软键一样。此时也会考虑当前所设的皮肤 - 参见显示机床数据 9112 = $MM_HMI_SKIN。

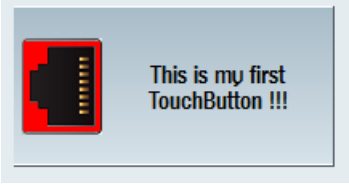


图 8-19　ButtonStyle - Softkey Look&Feel

采用“1 = 用户专用”设置时，可根据 TouchButton 的状态定义文本色和背景色。由此产生 3D 效应，可自动缩放图片并影响边框之间的间距。

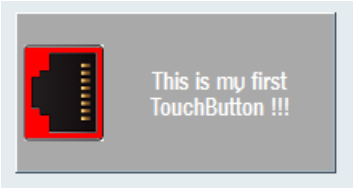


图 8-20　ButtonStyle - 用户专用

Flat – 显示方式

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**Flat**") WriteCWProperty(TouchButtonVarName, "**Flat**", Value)	
说明：	读取/设置 TouchButton 是平面显示（缺省设置）还是 3D 显示 **提示：**该属性只在显示方式（ButtonStyle）“1 = 用户专用”激活时才可用。	
	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
参数：	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE（缺省）或 FALSE

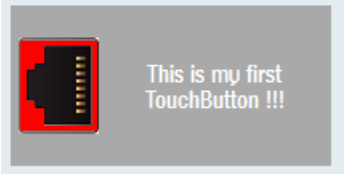


图 8-21　平面显示方式

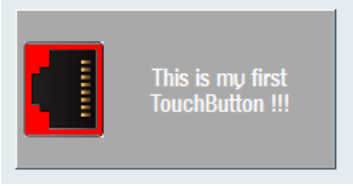


图 8-22　3D 显示方式

Enabled - TouchButton 的可操作性

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**Enabled**") WriteCWProperty(TouchButtonVarName, "**Enabled**", Value)	
说明：	读取/设置 TouchButton 是可以操作（= TRUE）还是不可以操作（FALSE）。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE（缺省）或 FALSE

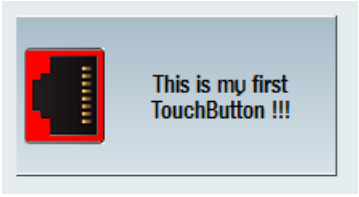


图 8-23　TouchButton 可操作

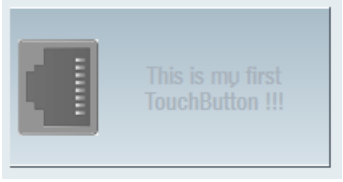


图 8-24　TouchButton 不可操作

说明

如果单击不可操作的 TouchButton，系统则会发送信号“clickedDisabled”。系统可以分析该信号并输出消息（为什么此时此刻 TouchButton 及其所连功能无法操作）。在未激活的状态下，显示图或背景图自动显示为灰度梯度。

Checkable – 转换功能激活/关闭

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**Checkable**") WriteCWProperty(TouchButtonVarName, "**Checkable**", Value)
说明：	读取/设置 TouchButton 的转换功能。

参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE 或 FALSE（缺省）

说明

正常情况下，松开后，TouchButton 会再次切换至正常状态（参看按键）。反之，如果 TouchButton 的转换功能激活（TRUE），则按下按键后，TouchButton 保持在所按位置。如果重新确认，则其切换至正常状态（参看开关）。

另见属性“Checked”。

Checked – 当前转换状态

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**Checked**") WriteCWProperty(TouchButtonVarName, "**Checked**", Value)	
说明：	读取/设置 TouchButton 的当前转换状态。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE 或 FALSE（缺省）

如果通过“Checkable”属性将转换功能设为“TRUE”，则可借助属性“Checked”读取或设置当前装换状态。

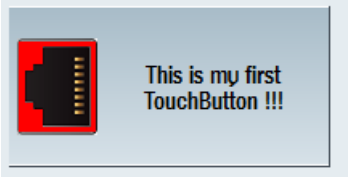


图 8-25　Unchecked

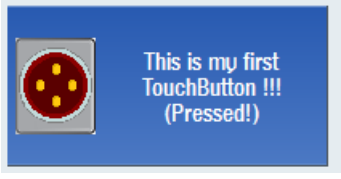


图 8-26　Checked

两种状态下均可以特殊显示文本和图片。

说明

另见属性“Checkable”。

ShowFocusRect – 焦点矩形显示

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**ShowFocusRect**") WriteCWProperty(TouchButtonVarName, "**ShowFocusRect**", Value)	
说明：	读取/设置 TouchButton 包含输入焦点时是否应显示焦点矩形。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE 或 FALSE（缺省）

焦点框架的颜色自动由 Operate 颜色设置设置，以下示例中为“橙色”。

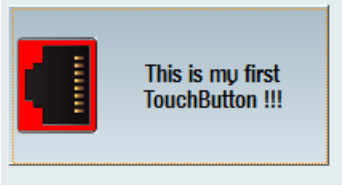


图 8-27　ShowFocusRect

Picture – 显示图

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**Picture**") WriteCWProperty(TouchButtonVarName, "**Picture**", Value)	
说明：	读取/设置 TouchButton 处于原始位置（未按压）时的待显示图片。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（String）
	Value	待设置的值（String）

指定文件名作为值，例如：“sk_ok.png”。TouchButton 自动从当前分辨率目录中确定正确的图片文件（参见章节使用图/图形 (页 71)）。

在未激活的状态下，显示图自动显示为灰度梯度。

说明

另见属性 "Text", "TextAlignedToPicture", "PictureAlignment", "ScalePicture", "PictureKeepAspectRatio"。

PicturePressed – 显示图处于按压状态

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**PicturePressed**") WriteCWProperty(TouchButtonVarName, "**PicturePressed**", Value)	
说明：	读取/设置 TouchButton 被按住或处于啮合状态时的待显示的图片。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（String）
	Value	待设置的值（String）

如果没有指定值（空字符串“”），TouchButton 在此状态下显示由属性“Picture”确定的图片。

指定文件名作为值，例如：“sk_ok.png”。TouchButton 自动从当前分辨率目录中确定正确的图片文件（参见章节使用图/图形 (页 71)）。

在未激活的状态下，显示图自动显示为灰度梯度。

说明

另见属性 "Text", "TextAlignedToPicture", "PictureAlignment", "ScalePicture", "PictureKeepAspectRatio"。

PictureAlignment – 图片对齐

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**PictureAlignment**") WriteCWProperty(TouchButtonVarName, "**PictureAlignment**", Value)	
说明：	读取/设置 TouchButton 上的显示图片对齐。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（int）
	Value	待设置的值（int）： 1 = 左（缺省） 2 = 右 32 = 上 64 = 下 128 = 居中

说明

参见章节图片和文本的定位和对齐 (页 311)。
另见属性 "Text", "TextAlignedToPicture", "PictureAlignmentString", "ScalePicture", "PictureKeepAspectRatio"。

PictureAlignmentString – 图片对齐

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**PictureAlignmentString**") WriteCWProperty(TouchButtonVarName, "**PictureAlignmentString**", Value)
说明：	读取/设置显示图的对齐。该属性含义等同于属性“PictureAlignment”。然而此处的值仍不能作为数值（int），而是字符串传输。

参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（String）
	Value	待设置的值（String）： "Left" = 左（缺省） "Right" = 右 "Top" = 上 "Bottom" = 下 "Center" = 居中

说明

参见章节 图片和文本的定位和对齐 (页 311)。

另见属性 "Text", "TextAlignedToPicture", "PictureAlignment", "ScalePicture", "PictureKeepAspectRatio"。

ScalePicture – 缩放显示图

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**ScalePicture**") WriteCWProperty(TouchButtonVarName, "**ScalePicture**", Value)	
说明：	读取/设置是否根据待显示的图缩放 TouchButton。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE 或 FALSE（缺省）

说明

参见章节 图片和文本的定位和对齐 (页 311)。

PictureKeepAspectRatio – 按页面比例缩放显示图

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**PictureKeepAspectRatio**") WriteCWProperty(TouchButtonVarName, "**PictureKeepAspectRatio**", Value)	
说明：	读取/设置是否应在缩放时（属性 "ScalePicture" = "TRUE"）保持显示图的页面比例。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE（缺省）或 FALSE

说明

参见章节 图片和文本的定位和对齐 (页 311)。

Text – 显示文本

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**Text**") WriteCWProperty(TouchButtonVarName, "**Text**", Value)	
说明：	读取/设置 TouchButton 处于原始位置（未按压）时的显示文本。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（String）
	Value	待设置的值（String）

所提供的矩形框中的文本达到字数限值时自动转换。

当显示文本来自于语言文件时，只能强制进行换行（受制约的）。

说明

参见章节 和语言相关的文本 (页 313)。

TextPressed – 显示文本处于按压状态

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**TextPressed**") WriteCWProperty(TouchButtonVarName, "**TextPressed**", Value)	
说明：	读取/设置 TouchButton 被按住或处于啮合状态时的待显示的文本。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（String）
	Value	待设置的值（String）

所提供的矩形框中的文本达到字数限值时自动转换。

当显示文本来自于语言文件时，只能强制进行换行（受制约的）。

说明

参见章节 和语言相关的文本 (页 313)。

TextAlignment – 文本对齐

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**TextAlignment**") WriteCWProperty(TouchButtonVarName, "**TextAlignment**", Value)	
说明：	读取/设置 TouchButton 上的文本对齐。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（int）
	Value	待设置的值（int）： 1 = 左 2 = 右 32 = 上 64 = 下 128 = 居中（缺省） （参见以下示例）



图 8-28　TextAlignment - 左



图 8-29　TextAlignment - 右

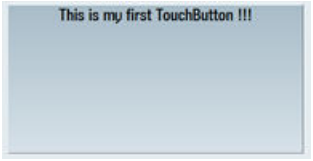


图 8-30　TextAlignment - 上

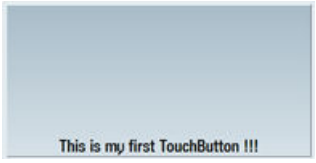


图 8-31　TextAlignment - 下



图 8-32　TextAlignment - 居中

说明

参见章节 图片和文本的定位和对齐 (页 311)。

另见属性 "Text", "TextAlignmentString", "TextAlignedToPicture"。

TextAlignmentString – 文本对齐

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**TextAlignmentString**") WriteCWProperty(TouchButtonVarName, "**TextAlignmentString**", Value)	
说明：	读取/设置文本的对齐。该属性含义等同于属性"TextAlignment"。此处的值仍不能作为数值（int），而是字符串传输。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（String）
	Value	待设置的值（String）： "Left" = 左 "Right" = 右 "Top" = 上 "Bottom" = 下 "Center" = 居中（缺省）

说明

参见章节 图片和文本的定位和对齐 (页 311)。
另见属性 "Text", "TextAlignment", "TextAlignedToPicture"。

TextAlignedToPicture – 相对于图片的文本对齐

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "TextAlignedToPicture") WriteCWProperty(TouchButtonVarName, "**TextAlignedToPicture**", Value)	
说明：	读取/设置显示文本是否应相对于图片定位。如果此处设为 FALSE，则文本在 TouchButton 上居中显示。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE（缺省）或 FALSE

说明

参见章节图片和文本的定位和对齐 (页 311)。
另见属性 "Text", "TextPressed", "Picture", "PicturePressed"。

BackColor – 背景色

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, " **BackColor**") WriteCWProperty(TouchButtonVarName, " **BackColor**", Value)	
说明：	读取/设置 TouchButton 处于原始位置（未按压）时的背景色。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性颜色（String）
	Value	待设置的值（String），作为“#RRGGBB”格式的 RGB 值， 例如“#04B7FB”

说明

该属性只在显示方式 “1 - 用户专用” 激活时才可用。

BackColorChecked – 啮合状态下的背景色

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**BackColorChecked**") WriteCWProperty(TouchButtonVarName, " **BackColorChecked**", Value)
说明：	读取/设置 TouchButton 处于啮合状态（转换功能）时的背景色。参见属性“Checked”、“Checkable”

参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性颜色（String）
	Value	待设置的值（String），作为“#RRGGBB”格式的 RGB 值， 例如“#04B7FB”

说明

该属性只在显示方式“1 - 用户专用”激活时才可用。

说明

另见属性“Checked”、“Checkable”

BackColorPressed – 按压状态时的背景色

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**BackColorPressed**") WriteCWProperty(TouchButtonVarName, " **BackColorPressed**", Value)	
说明：	读取/设置 TouchButton 处于按压状态时的背景色。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性颜色（String）
	Value	待设置的值（String），作为“#RRGGBB”格式的 RGB 值， 例如“#04B7FB”

说明

该属性只在显示方式“1 - 用户专用”激活时才可用。

BackColorDisabled – 未激活状态下的背景色

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**BackColorDisabled**") WriteCWProperty(TouchButtonVarName, " **BackColorDisabled**", Value)	
说明：	读取/设置 TouchButton 处于未激活状态时的背景色。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性颜色（String）
	Value	待设置的值（String），作为“#RRGGBB”格式的 RGB 值， 例如“#04B7FB”

说明

该属性只在显示方式“1 - 用户专用”激活时才可用。

说明

另见属性“Enabled”。

TextColor – 文本颜色

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**TextColor**") WriteCWProperty(TouchButtonVarName, " **TextColor**", Value)	
说明：	读取/设置 TouchButton 处于原始位置（未按压）时的文本颜色。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性颜色（String）
	Value	待设置的值（String），作为“#RRGGBB”格式的 RGB 值， 例如“#04B7FB”

说明

该属性只在显示方式“1 - 用户专用”激活时才可用。

TextColorChecked – 啮合状态下的文本颜色

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**TextColorChecked**") WriteCWProperty(TouchButtonVarName, " **TextColorChecked**", Value)	
说明：	读取/设置 TouchButton 处于啮合状态（转换功能）时的文本颜色。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性颜色（String）
	Value	待设置的值（String），作为“#RRGGBB”格式的 RGB 值， 例如“#04B7FB”

说明

该属性只在显示方式“1 - 用户专用”激活时才可用。

说明

另见属性“Checked”、“Checkable”

TextColorPressed – 按压状态时的文本颜色

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**TextColorPressed**") WriteCWProperty(TouchButtonVarName, "**TextColorPressed**", Value)	
说明：	读取/设置 TouchButton 处于按压状态时的文本颜色。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性颜色（String）
	Value	待设置的值（String），作为“#RRGGBB”格式的 RGB 值， 例如“#04B7FB”

说明

该属性只在显示方式“1 - 用户专用”激活时才可用。

TextColorDisabled – 未激活状态下的文本颜色

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**TextColorDisabled**") WriteCWProperty(TouchButtonVarName, "**TextColorDisabled**", Value)	
说明：	读取/设置 TouchButton 处于未激活状态时的文本颜色。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性颜色（String）
	Value	待设置的值（String），作为“#RRGGBB”格式的 RGB 值， 例如“#04B7FB”

说明

该属性只在显示方式“1 - 用户专用”激活时才可用。

说明

另见属性“Enabled”。

BackgroundPicture – 背景画面

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**BackgroundPicture**") WriteCWProperty(TouchButtonVarName, "**BackgroundPicture**", Value)
说明：	读取/设置待显示的、持续性（即：取决于状态的）背景画面。

参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（String）
	Value	待设置的值（String）

指定文件名作为值，例如：“sk_ok.png”。TouchButton 自动从当前分辨率目录中确定正确的图片文件（参见章节使用图/图形 (页 71)）。

在未激活的状态下，背景画面自动显示为灰度梯度。

说明

参见章节 使用图/图形 (页 71)。

另见属性 "BackgroundPictureAlignment", "ScaleBackgroundPicture", "BackgroundPictureKeepAspectRatio"。

BackgroundPictureAlignment – 背景画面对齐

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**BackgroundPictureAlignment**") WriteCWProperty(TouchButtonVarName, "**BackgroundPictureAlignment**", Value)	
说明：	读取/设置背景画面的对齐。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（int）
	Value	待设置的值（int）： 1 = 左 2 = 右 32 = 上 64 = 下 128 = 居中（缺省）

说明

参见章节 图片和文本的定位和对齐 (页 311)。

另见属性 "BackgroundPictureAlignment", "ScaleBackgroundPicture", "BackgroundPictureKeepAspectRatio"。

BackgroundPictureAlignmentString – 背景画面对齐

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**BackgroundPictureAlignmentString**") WriteCWProperty(TouchButtonVarName, "**BackgroundPictureAlignmentString**", Value)	
说明：	读取/设置背景画面的对齐。该属性含义等同于属性 “BackgroundPictureAlignment”。然而此处的值仍不能作为数值（int），而是字符串传输。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（int）
	Value	待设置的值（int）： "Left" = 左 "Right" = 右 "Top" = 上 "Bottom" = 下 "Center" = 居中（缺省）

说明

参见章节 图片和文本的定位和对齐 (页 311)。

另见属性 "BackgroundPictureAlignment", "ScaleBackgroundPicture", "BackgroundPictureKeepAspectRatio"。

ScaleBackgroundPicture – 缩放背景画面

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**ScalePicture**") WriteCWProperty(TouchButtonVarName, "**ScalePicture**", Value)	
说明：	读取/设置是否根据待显示的图缩放 TouchButton。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE 或 FALSE（缺省）

说明

参见章节 图片和文本的定位和对齐 (页 311)。

另见属性 "BackgroundPictureAlignment", "ScaleBackgroundPicture", "BackgroundPictureKeepAspectRatio"。

BackgroundPictureKeepAspectRatio – 按照页面比例缩放背景画面

句法：	ReturnValue = ReadCWProperty(TouchButtonVarName, "**BackgroundPictureKeepAspectRatio**") WriteCWProperty(TouchButtonVarName, "**BackgroundPictureKeepAspectRatio**", Value)	
说明：	读取/设置是否应在缩放时（属性 "ScaleBackgroundPicture" = "TRUE"）保持背景画面的页面比例。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	已读取的属性值（bool）
	Value	待设置的值（bool）：TRUE（缺省）或 FALSE

说明

参见章节 图片和文本的定位和对齐 (页 311)。

8.6.4 功能

可通过 CallCWMethod() 调用以下功能。

示例

设置通过显示变量“MyTouchButton”所连的 SlEsTouchButton 的间距。

结果写入寄存器 0 中。

```
REG[0] = CallCWMethod("MyTouchButton", "setMargins", 20, 20, 20,
20, 20)
```

概述

功能	说明
setMargins	设置间距
serialize	保存/恢复当前状态

setMargins – 设置间距

句法：	ReturnValue = CallCWMethod(TouchButtonVarName, "**setMargins**", left, top, right, bottom, center) ReturnValue = CallCWMethod(TouchButtonVarName, "**setMargins**", left, top, right, bottom, center, marginAlignment)
说明：	借助此功能可以定义边框间距和图片与文本之间的间距。 如果一个值为“-1”，则该值对应系统缺省间距。如果该值大于或等于“0”，则该值设为边框间距。

参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	左	左侧边框（int）
	上	上边框（int）
	右	右侧边框（int）
	下	下边框（int）
	center	不是“居中”对齐时，图片与文本之间的间距（int）
	marginAlignment	可选：确定页边距除了作用于显示图外是否也作用于显示文本的定位（bool），TRUE（缺省）

serialize – 保存/恢复当前状态

句法：	ReturnValue = CallCWMethod(TouchButtonVarName, "**serialize**", FilePath, IsStoring)	
说明：	通过该功能可以在需要时将 SlEsTouchButton 的当前状态和内容以二进制格式写入文件或数据流中并恢复。该功能会自动更新显示。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	Return Value	错误代码（bool）：TRUE = 成功
	FilePath	包含文件名称的完整路径（QString）
	IsStoring	保存/恢复（bool）：TRUE = 保存

说明

设计者不能直接使用该功能！

8.6.5 信号

以下描述的信号可在配置时拦截并作出相应的响应。

概述

功能	说明
clickedDisabled	操作未激活的 TouchButton
clicked	已点击
checked	已转换

- 如果 TouchButton 可操作而不可转换，则操作时系统会发送信息“clicked”。
- 如果 TouchButton 可操作且可转换，则操作时系统会发送以下顺序的信息：
  "checked" → "clicked"
- 如果 TouchButton 不可操作，则操作时系统会发送信息“clickedDisabled”。

所有上述信号总是在进行了操作之后才发送，即：只有松开了鼠标或空格键时或在多点触控上点击之后。

如此一来，SlEsTouchButton 仅有纯粹的开关功能，即：无法实现按键功能。

说明

在多点触控操作时要注意，当完全识别到点击手势（按住并松开在 0.7 秒内）后，才会发送 Click 信息。如果在简单按压时已经执行了操作，则其后的 ScrollArea 无法滚动/移动，或者说很容易导致意外的错误操作。

clicked – 在 TouchButton 上执行了 Click 操作

句法：	SUB(on_<TouchButtonVarName>_clicked) … END_SUB	
说明：	可操作 TouchButton 的按压和松开构成的完整顺序会产生信号“clicked”。建议主要以该信号进行操作。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	SIGARG[0]	提供 TouchButton 的转换状态（bool）

示例

```
DEF MyTouchButton = (W///,"slesstdcw.SlEsTouchButton"/////70,20,200,100/0,0,0,0)
SUB(on_MyTouchButton_clicked)
   DLGL("checked: " << SIGARG[0])
END_SUB
```

checked – 转换了 TouchButton

句法：	SUB(on_<TouchButtonVarName>_checked) … END_SUB	
说明：	按压可转换的 TouchButton，从而改变其状态。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	SIGARG[0]	提供 TouchButton 的转换状态（bool）

示例

```
DEF MyTouchButton = (W///,"slesstdcw.SlEsTouchButton"/////70,20,200,100/0,0,0,0)
SUB(on_MyTouchButton_toggled)
   DLGL("toggled: " << SIGARG[0])
END_SUB
```

clickedDisabled – 在不可操作的 TouchButton 上执行了 Click 操作

句法：	SUB(on_<TouchButtonVarName>_clickedDisabled) … END_SUB	
说明：	不可操作 TouchButton 的按压和松开构成的完整顺序会产生信号 “clickedDisabled”。	
参数：	TouchButtonVarName	包含 SlEsTouchButton 的显示变量的名称
	SIGARG[0]	提供 TouchButton 的转换状态（bool）

示例

```
DEF MyTouchButton = (W///,"slesstdcw.SlEsTouchButton"/////70,20,200,100/0,0,0,0)
SUB(on_MyTouchButton_clickedDisabled)
   DLGL("checkedDisabled: " << SIGARG[0])
END_SUB
```

8.6.6 图片和文本的定位和对齐

图片的定位

使用规定的对齐方式（Alignment）将图片进行如下定位：

- 第一步从所属分辨率目录中确定与当前分辨率相适应的图片。
- 然后，表面矩形（ClientArea）缩小与规定的边框距离差不多大小的区域（MarginArea）。
  MarginArea 会受“setMargins”的影响：
  - setMargins(-1, -1, -1, -1, -1)

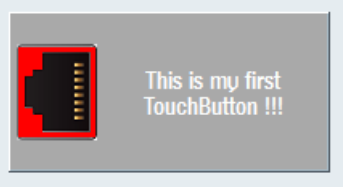


图 8-33 setMargins(-1, -1, -1, -1, -1)

  - setMargins(0, 0, 0, 0, 0)

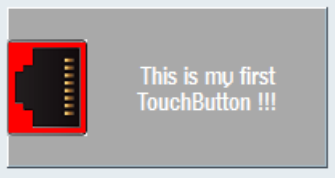


图 8-34 setMargins(0, 0, 0, 0, 0)

  - setMargins(20, 20, 20, 20, 20)

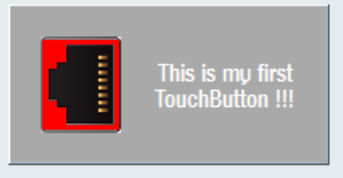


图 8-35 setMargins(20, 20, 20, 20, 20)

“左”对齐示例

平面进行水平对分，使得待显示的图片最大占用半个平面的大小。

此时，图片显示如下：

- 属性 "scalePicture"：FALSE
  图片在此位置显示，无任何缩放。此时要注意，作为基础的图片不要太大且它能在 TouchButton 内完整地显示出来。

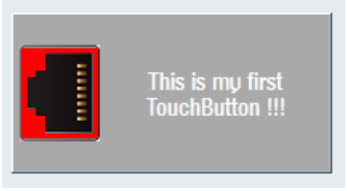


图 8-36　scalePicture FALSE

- 属性 "scalePicture"：TRUE
  图片进行缩放（放大或缩小），使其适应 TouchButton 的左半边。此时需要考虑到属性“pictureKeepAspectRatio”：
  - 属性 "pictureKeepAspectRatio"：FALSE
    图片进行水平和垂直缩放，使其刚好适应 TouchButton 的左半边。此时无需再关注原始图片的页面比例。

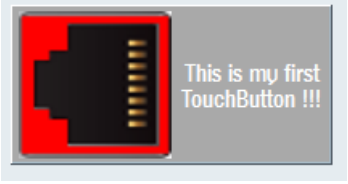


图 8-37　scalePicture - pictureKeepAspectRatio FALSE

  - 属性 "pictureKeepAspectRatio"：TRUE
    在考虑到原始图片的页面比例的情况下将图片尽可能大地缩放到 TouchButton 的左半边大小。

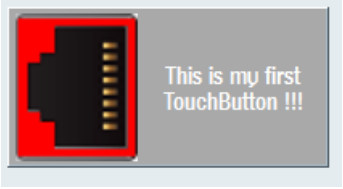


图 8-38　scalePicture - pictureKeepAspectRatio TRUE

说明

右对齐、上对齐、下对齐时的原则与此相同。

居中对齐时，试着根据属性“scalePicture”和“pictureKeepAspectRatio”让图片适应整个 MarginArea 区域。

文本的定位

根据属性“textAlignedToPicture”，对文本进行如下定位：

- 属性 "textAlignedToPicture":FALSE
  文本在 MarginArea 垂直水平居中显示。

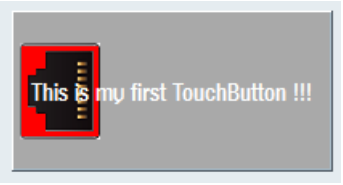


图 8-39　textAlignedToPicture FALSE

- 属性 "textAlignedToPicture":TRUE
  除了图片所显示的区域之外，文本在 MarginArea 其他区域中居中水平垂直输出。

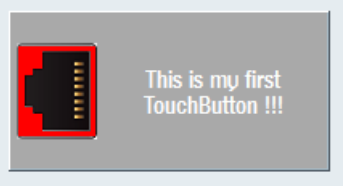


图 8-40　textAlignedToPicture TRUE

8.6.7　和语言相关的文本

SlEsTouchButton 中并没有独立的第三方语言支持。然而以下示例中显示了如何实现语言相关性：

示例

easyscreen.ini：

```
[LANGUAGEFILES]LngFile03 = user.txt
```

user_eng.txt：

```
85000 0 0 "This is my first Touchbutton !!!"
```

user_deu.txt：

```
85000 0 0 "Das ist mein erster Touchbutton !!!"
```

配置文件：

```
//M(MyTBMask/"My CustomWidget TouchButton ...")
DEF MyTB1 = (W///,"slesstdcw.SlEsTouchButton"/////70,20,200,100/0,0,0,0)
LOAD
   WRITECWPROPERTY("MyTB1", "text", $85000)
END_LOAD
LANGUAGE
   WRITECWPROPERTY("MyTB1", "text", $85000)
END_LANGUAGE
```

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”（参见以下示例）。

user_eng.txt：

```
85001 0 0 "This is%nmy%nfirst%nTouchButton !!!"
```

配置文件：

```
WRITECWPROPERTY("MyTB1", "text", $85001)
```

结果



图 8-41　和语言相关的文本中的换行示例

操作区“Custom（定制）” 9

9.1 这样激活操作区“自定义”

激活操作区“Custom”

“自定义”操作区在交付时未激活。

1. 首先将文件“slamconfig.ini”从 [系统西门子目录]/cfg 目录下复制到 [系统 oem 目录]/cfg 下或者复制到相应的 [系统插件目录]/cfg 或 [系统用户目录]/cfg 下。
2. 为激活操作区“Custom”，必须进行以下输入：

```
[Custom]
Visible=True
```

结果

激活后，操作区“Custom”的软键位于 HSK4 (= 预设置) 扩展菜单栏的主菜单(F10)中。

操作区“Custom”显示为带有可设置标题的空窗口。所有的水平和垂直软键都是可设置的。

9.2 这样设计“自定义”软键

设计用于操作区"Custom"的软键

在文件 "slamconfig.ini" 中设计操作区"Custom"软键的标签和位置。

登入软键的设计如下：

1. 为了使用**语言相关的文本**替代软键上的标签，则在[Custom]段中必须有以下输入项：
   ```
   TextId=MY_TEXT_ID
   TextFile=mytextfile
   TextContext=mycontext
   ```
   在该示例中软键显示语言相关的文本，文本 ID 为"MY_TEXT_ID"保存在文本文件 mytextfile_xxx.qm 中的"MyContext"下（xxx 代表语种缩写）。
2. 为了使用**与语言无关的文本**替代软键上的标签，则在[Custom]段中必须有以下输入项：
   ```
   TextId=HELLO
   TextFile=<empty>
   TextContext=<empty>
   ```
   在该示例中操作区"Custom"软键在每种语言中都显示为"HELLO"。
3. 除了文本外，软键上还可以显示**图标**。
   为此在[Custom]段中必须有以下输入项：
   ```
   Picture=mypicture.png
   ```
   这样软键就会显示 mypicture.png 文件中的图标。图形和位图保存在以下路径下：[系统 oem 目录]/ico/ico<分辨率>。根据显示器的分辨率来使用相应目录下的图片。
4. 此外还可以设置软键的**位置**。为此在[Custom]段中必须有以下输入项：
   ```
   SoftkeyPosition=12
   ```
   位置 12 是预设置。这是操作区菜单扩展栏上的 HSK4。位置 1-8 是菜单栏上的 HSK1 到 HSK8，位置 9-16 是菜单扩展栏上的 HSK1 到 HSK8。

9.3 这样设计操作区“自定义”

设计用于操作区"Custom"的软键

进行操作区设计需要文件 "easyscreen.ini" 和 "custom.ini"。两个文件的模板位于 [系统西门子目录]/templates/cfg 目录下。

1. 首先将这些文件复制到目录 [系统 oem 目录]/cfg 下，再在这里进行修改。
2. 在文件 "easyscreen.ini" 中已经包含了操作区“自定义”的定义行：
   ```
   ;StartFile02 = area := Custom, dialog := SlEsCustomDialog,
   startfile := custom.com
   ```
   一行开头的“；”代表是注释内容。这一行即是注释并因此无效。要变其状态必须删除“；”。
   通过该行中的属性"startfile"来定义，在选择操作区“自定义”时输入项参考设计文件 "custom.com"。
3. 在 [系统 oem 目录]/proj 下创建**项目文件“custom.com”**。其中包括模拟操作区“程序”的 aeditor.com 文件的设计。所设计的登入软键于是显示在操作区“自定义”中。
4. 在文件 "custom.ini" 中设计用于对话框标题行的 **与语言无关的文本**。
   为此在模板中必须有以下输入项：
   ```
   [Header]
   Text=Custom
   ```
   这些文本可以替换成用户自定义的文本。
5. 为了设计操作区“自定义”的**开始画面**，模板中必须有以下输入项：
   ```
   [Picture]
   Picture=logo.png
   ```
   Logo.png 是在操作区“自定义”的开始对话框中显示的开始画面的名称。此处可以显示公司徽标或其他图片。文件保存在相应分辨率的文件夹下：[系统 oem 目录]/ico/ ...
6. 如要在首次显示“自定义”操作区时直接显示某个 "Run MyScreens" 屏幕，则在模板中应存在以下记录：
   ```
   ; Mask shown with startup of area "custom"
   [Startup]
   ;Startup = Mask:=MyCustomStartupMask, File:=mycustommasks.com
   ```
   典型的应用场合，例如在启动 SINUMERIK Operate 时直接以某个 "Run MyScreens" 屏幕启动。
   为此，在配置文件 "systemconfiguration.ini" 中应按以下方式设置段"[miscellaneous]"关键字 "startuparea"：
   ```
   [miscellaneous]
   ;name of the area to be shown at system startup
   startuparea = Custom
   ```

9.4 操作区“自定义”的编程示例

示例



图 9-1 软键" Start example"示例

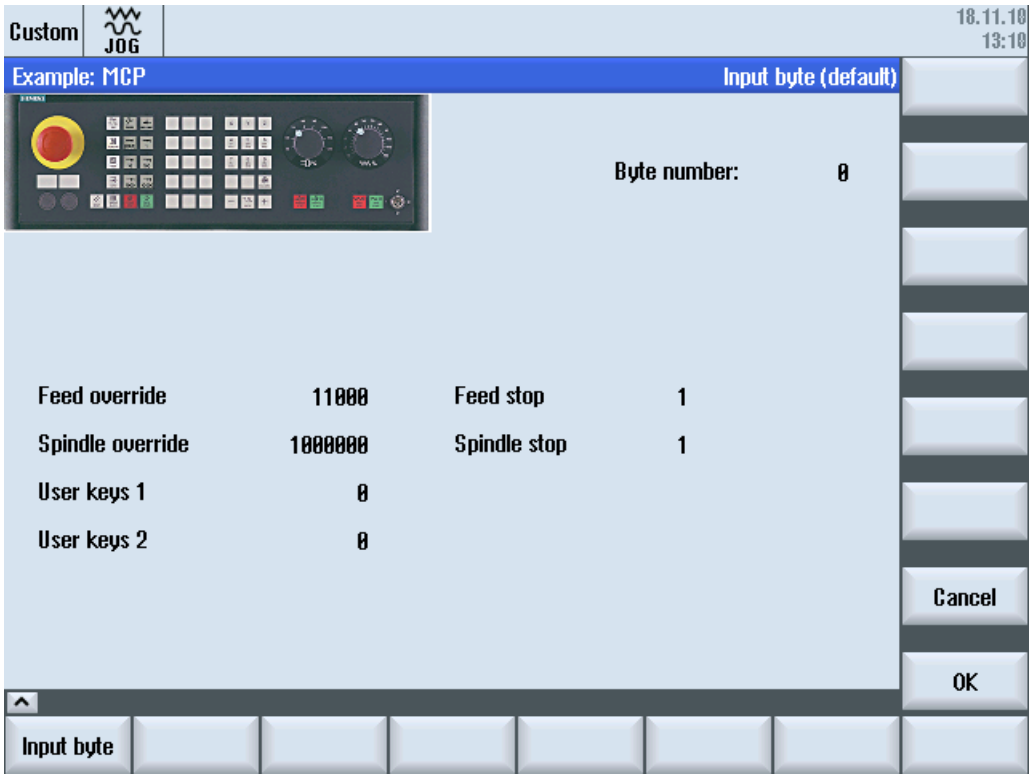


图 9-2　位图和文本栏示例

文件概览

需要以下文件：

- "custom.com"
- "easyscreen.ini"

编程

文件 “custom.com” 内容：

说明

本例中链接的图形文件 mcp.png 仅用于示范。如果希望改编本示例，可以使用自己的图形文件。

```
//S(Start)
   HS7=("Start example", se1, ac7)
```

```
   PRESS(HS7)
      LM("Maske4")
   END_PRESS
//END
//M(Maske4/"Example: MCP"/"mcp.png")
   DEF byte=(I/0/0/"Input byte=0 (default)","Byte number:",""/wr1,li1///380,40,100/480,40,50)
   DEF Feed=(IBB//0/"","Feed override",""/wr1//"EB3"/20,180,100/130,180,100), Axistop=(B//0/"","Feed
stop",""/wr1//"E2.2"/280,180,100/380,180,50/100)
   DEF Spin=(IBB//0/"","Spindle override",""/wr1//"EB0"/20,210,100/130,210,100), spinstop=(B//
0/"","Spindle stop",""/wr1//"E2.4"/280,210,100/380,210,50/100)
   DEF custom1=(IBB//0/"," User keys 1",""/wr1//"EB7.7"/20,240,100/130,240,100)
   DEF custom2=(IBB//0/"","User keys 2",""/wr1//"EB7.5"/20,270,100/130,270,100)
   DEF By1
   DEF By2
   DEF By3
   DEF By6
   DEF By7

   HS1=("Input byte", SE1, AC4)
   HS2=("")
   HS3=("")
   HS4=("")
   HS5=("")
   HS6=("")
   HS7=("")
   HS8=("")
   VS1=("")
   VS2=("")
   VS3=("")
   VS4=("")
   VS5=("")
   VS6=("")
   VS7=("Cancel", SE1, AC7)
   VS8=("OK", SE1, AC7)

   PRESS(VS7)
      EXIT
   END_PRESS

   PRESS(VS8)
      EXIT
   END_PRESS

   LOAD
      By1=1
```

```
      By2=2
      By3=3
      By6=6
      By7=7
   END_LOAD

   PRESS(HS1)
      Byte.wr=2
   END_PRESS

   CHANGE(Byte)
      By1=byte+1
      By2=byte+2
      By3=byte+3
      By6=byte+6
      By7=byte+7
      Feed.VAR="EB"<<By3
      Spin.VAR="EB"<<Byte
      Custom1.VAR="EB"<<By6
      Custom2.VAR="EB"<<By7
      Axisstop.VAR="E"<<By2<<".2"
      Spinstop.VAR="E"<<By2<<".4"
      Byte.wr=1
   END_CHANGE

   CHANGE(Axis stop)
      IF Axistop==0
         Axistop.BC=9
      ELSE
         Axistop.BC=11
      ENDIF
   END_CHANGE

   CHANGE(Spin stop)
      IF Spinstop==0
         Spinstop.BC=9
      ELSE
         Spinstop.BC=11
      ENDIF
   END_CHANGE
//END
```

对话框选择 10

10.1 通过 PLC 软键选择对话框

设计

步骤说明：

- 在 systemconfiguration.ini 中有 [keyconfiguration] 一段。此记录定义了指定 PLC 软键的动作。
- 设定动作编号，如其大于等于 100 则为 Run MyScreens 调用。
- 在文件“easyscreen.ini”中必须创建一个段用于定义需要执行的动作，其名称由操作区域名称和对话框名称组成（参见 [keyconfiguration] 下的记录 → Area:=..., Dialog:=...）→ [<Area>_<Dialog>] → 例如 [AreaParameter_SlPaDialog]
- 在此段中定义动作编号（已在 systemconfiguration.ini 中设定 → 参见 Action:=...）。此处涉及两个指令：
  1. LS("Sofkeyleiste1","param.com") ... 装载软键栏
  2. LM("Maske1","param.com") ... 装载屏幕

通过 PLC 软键选择软键栏

在 Run MyScreens 中可通过 PLC 软键选择 Run MyScreens 软键栏和 Run MyScreens 对话框。为此在设计相关 PLC 软键时候，必须为属性“action”设定大于等于 100 的赋值。

PLC 软键的设计在文件 systemconfiguration.ini 中的 [keyconfiguration] 段中进行：

```
[keyconfiguration]
KEY75.1 = Area:=area, Dialog:=dialog, Screen:=screen, Action:=
100,
```

对于相应 PLC 软键，需要执行的 LM 和 LS 指令在文件 “easyscreen.ini” 中的段中进行，段的名称按照以下规定创建。

`[areaname_dialogname]`	名称的第一部分“areaname”指示操作区域，第二部分“dialogname”指示对话框，段中设计的指令适用于此对话框。

`[AreaParameter_SlPaDialog]`
`100.screen1 = LS("Softkey1","param.com")`
`101.screen3 = LM("Maske1","param.com")`
	必须使用在文件 systemconfiguration.ini 中为操作区域和对话框设定的名称。对话框的设定是可选项。
	特别是对于通过单个对话框执行的操作区域，可省略此设定，见旁边的例子。
	如果在通过对话框 SlPaDialog 执行的操作区域 AreaParameter 中显示“screen1”，则在出现值为 100 的“action”时执行指令 "LS("Softkey1","param.com")"。

`action.screen=指令`	通过两个属性“action”和“screen”将执行指定指令的时间设定为唯一值。
	“screen”的设定是可选项。
	允许的指令为：
	`LM (LoadMask)`
	`LS (LoadSoftkeys)`

10.1.1 im 在程序编辑器中调用 Run MyScreens-循环窗口

使用范围

通过 PLC 硬键可以在程序编辑器中直接打开 Run MyScreens 循环窗口。

边界条件

工艺循环只能在程序编辑器打开时进行选择。在窗口中设置过的工艺循环可插入到程序编辑器中光标的当前位置处。

步骤

1. 通过 PLC 硬键 (页 326) 创建 Run MyScreens 循环。
2. 为了能将 Run MyScreens 窗口正常集成到程序编辑器中，必须如在以下示例中描述的那样对 PLC 硬键配置进行扩展设置：
   完整的硬键配置示例：
   systemconfiguration.ini
   ```
   [keyconfiguration]
   KEY101.0 = action:=209, runmyscreenmode:=HandledByDialog
   ```
   easyscreen.ini
   ```
   [AreaProgramEdit_SlProgramEdit]
   209 = LM("MASK_E_DR_O1_MAIN","e_dr_o1.com")
   ```
3. 所打开的 Run MyScreens 循环窗口的反馈可以通过 PLC 接口"DB19.DBW24"来检查。
   示例：
   在本示例中，当 Run MyScreens 窗口打开时，值"9876"被写入至 PLC 接口：
   配置，PLC-ID=9876：
   //M(MASK_E_DR_O1_MAIN/$85305/////XG1,FA2,TA7//"drilling.txt"/**9876**)
   或
   //M{ MASK_E_DR_O1_MAIN, HD=$85305, LANGFILELIST=" drilling.txt", **PLC_ID=9876**}
   另见：SINUMERIK 840D sl 功能手册之基本功能

OA 接口上的程序编辑器状态

程序编辑器的状态可通过 CAP 本地变量"/Hmi/StepEditorInfo"来询问。

变量名称：	/Hmi/StepEditorInfo
变量描述：	当前选中程序的编辑器信息。
字符串包含下列信息	
[fileName]	程序的路径信息
[channel]	通道编号
[technlogy]	工艺
[appearance]	程序标识（ShopMill/ShopTurn/G 代码）
[state]	StepEditor 的状态： • EditorClosed：编辑器关闭（例如 HMI 未在程序操作区域中） • EditEnabled：允许编辑 • EditDisabled：不允许编辑（例如只读或光标位置无效）

10.2 通过 PLC 硬键选择对话框

使用范围

PLC 可以启动操作软件中的以下功能：

- 选择操作区域，
- 选择操作区内的某个屏幕
- 执行软键上标注的功能

硬键

所有按键，包括 PLC 按键在下文都称为“硬键”。可以最多定义 254 个硬键，具体可以分为：

按键号	使用
按键 1 - 9	操作面板上的按键
按键 10 - 49	保留
按键 50 - 254 按键 50 - 81	PLC 按键： 预留给 OEM 使用
按键 255	由控制信息预定义。

硬键 1 - 9 的预定义如下：

按键标识		动作/作用
HK1	MACHINE	选择操作区“加工”，回到上一个对话框
HK2	PROGRAM	选择操作区“程序”，回到上一个对话框或上一个程序
HK3	OFFSET	选择操作区“参数”，回到上一个对话框
HK4	PROGRAM MANAGER	选择操作区“程序”，初始画面“程序管理器”
HK5	ALARM	选择操作区“诊断”，对话框“报警列表”
HK6	CUSTOM	操作区“自定义”
HK7 1)	MENU SELECT	选择“初始菜单”
HK8 1)	MENU FUNCTION	选择“功能”操作区
HK9 1)	MENU USER	选择“用户”操作区

1) 只用于 828D

配置

按键的定义在配置文件“systemconfiguration.ini”的章节[keyconfiguration]中进行。每一行定义了一个“硬键事件”。“硬键事件”指第几次按下某个硬键。比方说第二次按下某个硬键和第三次按下该硬键产生的动作就不一样。

配置文件“systemconfiguration.ini”中的条目可以载入用户自定义的设置，为此可以使用目录 [系统用户目录]/cfg 和 [系统 oem 目录]/cfg。

配置文件中用于定义硬键事件的行的结构为：

```
KEYx.n = area:=area, dialog:=dialog, screen:=screen,
forms:=form, menus:=menu,
action:=menu.action, cmdline:=cmdline
KEYx.n = area:=area, dialog:=dialog, cmdline:=cmdline, action:=
action
```

x:硬键的编号，值域：1 – 254

n:事件号 - 相当于第几次按下硬键，值域：0 – 9

前提条件

PLC 用户程序必须首先符合以下前提条件：

每次总是只执行一个硬键功能。因此只有当操作软件应答了前面的一个请求后，才可以处理新的请求。当 PLC 用户程序将 MCP 按键转成硬键时，必须有足够大的中间缓存，这样即使在快速操作中也不会丢失某个按键事件。

PLC 接口

在 PLC 接口上对用于选择硬键的范围进行了规定。该范围定义在 DB19.DBB10 中。在该范围中，PLC 可以直接指定 50 到 254 之间的按键值。

操作软件分两步应答硬键事件。这样操作软件便可以将两个连续的相同按键代码看成两个单独的事件进行处理。在第一步中，控制信息 255 被写入到字节 DB19.DBB10 中。借助该虚拟的按键操作可以明确地看出 PLC 的每个按键序列。控制信息对于 PLC 程序没有作用，不允许被修改。在第二步中，操作软件删除 DB19.DBB10，真正地向 PLC 发出应答。从此刻起，PLC 用户程序便成功指定了一个新硬键。与此同时，操作软件开始处理新的硬键请求。

示例

配置文件：

```
; PLC 硬键(KEY50-KEY254)
[keyconfiguration]
KEY50.0 = name := AreaMachine, dialog := SlMachine
KEY51.0 = name := AreaParameter, dialog := SlParameter
KEY52.0 = name := AreaProgramEdit, dialog := SlProgramEdit
KEY53.0 = name := AreaProgramManager, dialog := SlPmDialog
KEY54.0 = name := AreaDiagnosis, dialog := SlDgDialog
KEY55.0 = name := AreaStartup, dialog := SlSuDialog
KEY56.0 = name := Custom, dialog := SlEsCustomDialog
```

区域和对话框名称请从 [系统西门子目录]/cfg 目录下的“systemconfiguration.ini”文件中获取。

10.3 通过 NC 选择对话框

HMI Operate 中的 MMC 指令

可按下述方法使用 MMC 指令：

1. **MMC 指令的定义**
   标准文件 "systemconfiguration.ini" 中含有以下组合：
   地址:=MCYCLES --> 指令:=LM
   地址:=CYCLES --> 指令:=PICTURE_ON
   该区别用于区分测量循环和其他循环。即：
   - LM 只用于测量循环，PICTURE_ON 只用于非测量循环
   - 新定义的 MMC 指令不允许称为 PICTURE_ON 和 LM
2. **"Run MyScreens" 授权**
   所有 "Run MyScreen" 打开的对话框均需获得 "Run MyScreens" 授权，若无授权，只能使用有限数量的对话框。
   使用 test.com (Run MyScreens) 的调用示例：

```
g0 f50
MMC("CYCLES,PICTURE_ON,test.com,maske1","A")
m0
MMC("CYCLES,PICTURE_OFF","N")
M30
```

循环屏幕示例 11

11.1 循环屏幕示例

安装 SINUMERIK Operate 后也可以保存以 Run MyScreens 创建的循环屏幕示例。

在以下保存路径中查找 Run MyScreens 示例：

[系统西门子目录]		
...\siemens\sinumerik\hmi\template\easyscreen\		
...		
	standard\	帮助画面，PNG
	x3d\	帮助画面，动画
...		
	Milling\	铣削加工
	Turning\	车削加工
...		
	deu\	德语说明
	eng\	英语说明

在 HMI 界面上，可在调试操作区中找到相关示例：

水平软键“系统数据 → HMI 数据 → 模板 → 示例 → Easyscreen → ...”（见上）。

示例简介

- 铣削加工
  - 钻削示例：此处可添加一个位置
  - 测量工件示例：此处可在对话框打开时通过 NC 启动生成循环调用并执行
  - 工件计数器示例：在本例中以 GUD 的使用为例
- 车削加工
  - 设备：包含尾座、棒料装载器、零件抓取器
  - 测量工件示例：此处可在对话框打开时通过 NC 启动生成循环调用并执行
  - 钻削示例：此处可添加一个位置

11.1 循环屏幕示例

在 Sidescreen 中设计

12

需要在 Sidescreen 中显示 Run MyScreens 设计窗口时，您有以下几种操作：

1. 在一个单独的 Sidescreen 页面中显示一个 Run MyScreens 设计窗口：
   Run MyScreens 设计窗口此时会占据整个 Sidescreen 页面。
2. 在一个单独的 Sidescreen 页面中显示多个 Run MyScreens 设计窗口：
   多个 Run MyScreens 设计窗口以所谓的“Sidescreen 元素”在一个 Sidescreen 页面内堆叠显示。这些 Sidescreen 元素，即 Run MyScreens 设计窗口此时可以垂直滚动。
3. 在一个现有 Sidescreen 页面中添加一个或多个 Run MyScreens 设计窗口：
   即将 Run MyScreens 设计窗口添加到出厂时西门子交付的 Sidescreen 页面中。该操作和第 2 种操作基本相同，区别仅仅是涉及的 Sidescreen 页面有所不同。
4. 在一个 Sidescreen 元素中添加一个或多个 Run MyScreens 设计窗口：
   此时，各个 Run MyScreens 设计窗口以所谓的“Sidescreen 窗体”并排显示，可以水平滚动。

12.1 在一个单独的 Sidescreen 页面中显示一个 Run MyScreens 设计窗口

您必须在现有的 Sidescreen 配置基础（“slsidescreen.ini”）上创建一个新的 Sidescreen 页面，才可以一个单独的 Sidescreen 页面中显示一个 Run MyScreens 设计窗口。随后该 Run MyScreens 设计窗口便可以显示在该 Sidescreen 页面中，或在其中执行。

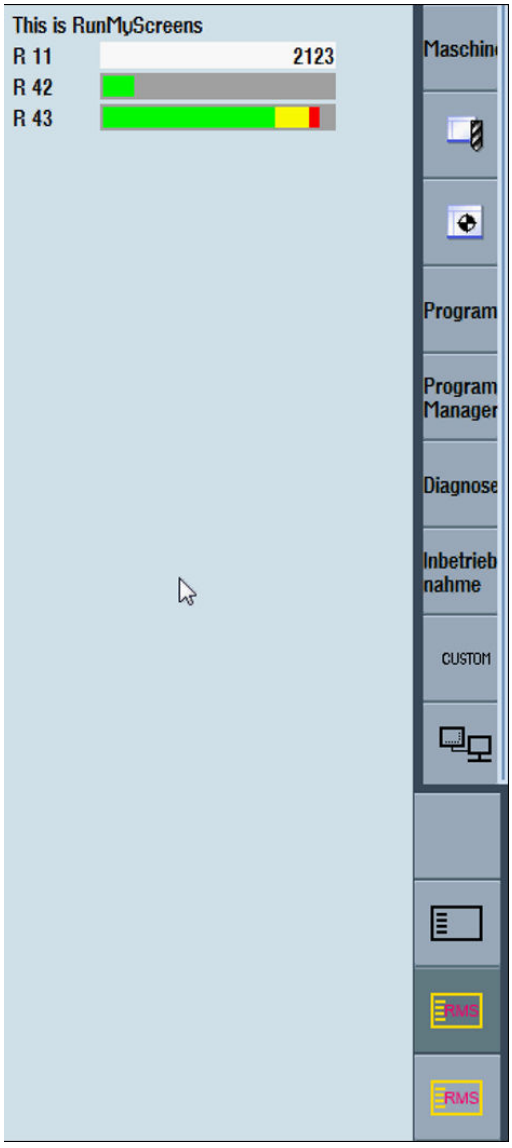


图 12-1 **在一个单独的 Sidescreen 页面中显示一个 Run MyScreens 设计窗口**

步骤

在文件 “slsidescreen.ini” 中增加以下条目：

```
[Sidescreen]
PAGE100= name:=RMSPage,
implementation:=sleseasyscreen.SlEsSideScreenPage
```

- 您必须根据现有的 Sidescreen 页面数量为新的页面编号，此处为 100。页面编号必须大于等于 100，以避免和西门子页面冲突。
- 页面名称可自由选择，此处为 RMSPage。
- 页面设计不允许更改，此处为 sleseasyscreen.SlEsSideScreenPage。

下一步便需要配置待执行的 Run MyScreens 设计窗口。新建一个段落，其名称包含了新页面的名称；

```
[Page_RMSPage]
Icon=rmspage.png
PROPERTY001= name:=maskPath, type:=QString,
value:="sidescreenmask.com"
PROPERTY002= name:=maskName, type:=QString, value:="Mask"
PROPERTY003= name:=focusable, type:=bool, value:="true"
```

- 段落名称的格式为：前缀“Page_”，后面跟了页面名称，此处为 RMSPage。
- 在“Icon”属性中输入图标文件的名称，此处为 rmspage.png。该文件包含了页面选择按钮的图标。将文件保存在 /user/sinumerik/hmi/ico/ico640 或者 /oem/sinumerik/hmi/ico/ico640 目录中。
- 在“maskPath”属性中输入包含了 RunMyScreens 设计窗口文件的名称，此处为 sidescreenmask.com。
- 在“maskName”属性中输入需要显示的 RunMyScreens 设计窗口的名称，此处为 Mask。
- 在“focusable”属性中确定输入焦点。只有当此属性为“真”时，页面才可获得输入焦点。具有输入元素的页面需要具备此属性。

将“slsidescreen.ini”保存在 /oem/sinumerik/hmi/cfg 或 /user/sinumerik/hmi/cfg 目录中。

重启 HMI 后，屏幕上便会显示 Run MyScreens 设计窗口。

示例

示例：“slsidescreen.ini” 中的一段完整配置：

```
[Sidescreen]
PAGE005= name:=RMSPage,
implementation:=sleseasyscreen.SlEsSideScreenPage
[Page_RMSPage]
PROPERTY001= name:=maskPath, type:=QString,
value:="sidescreenmask.com"
PROPERTY002= name:=maskName, type:=QString, value:="Mask"
```

12.2 在一个单独的 Sidescreen 页面中显示多个 Run MyScreens 设计窗口

除了创建一个新 Sidescreen 页面外，您还必须配置所谓的“Sidescreen 元素”，才可以一个单独的 Sidescreen 页面中显示多个 Run MyScreens 设计窗口。Sidescreen 元素用于显示 Run MyScreens 设计窗口。

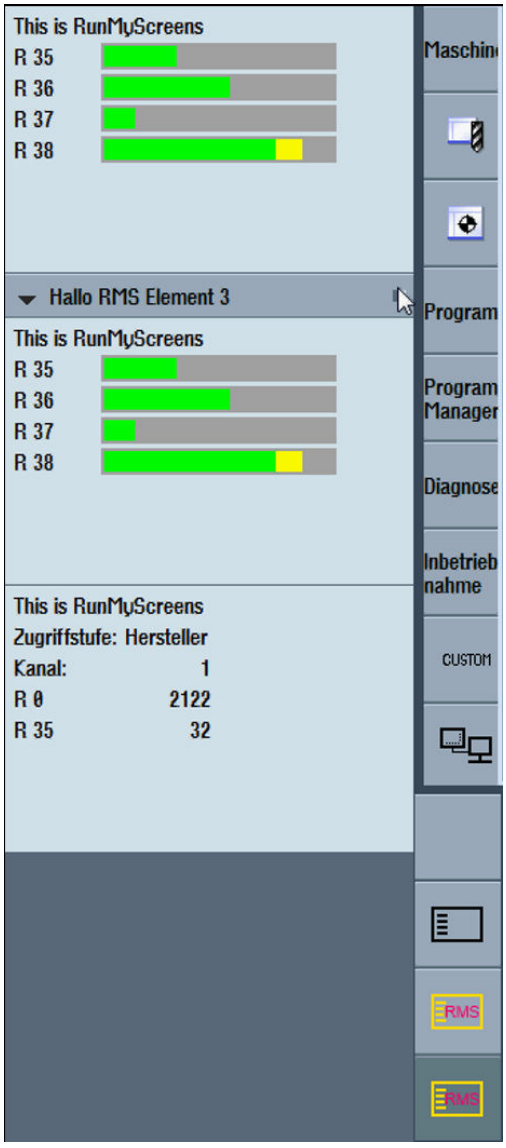


图 12-2 **在一个单独的 Sidescreen 页面中显示多个 Run MyScreens 设计窗口**

步骤

在文件 “slsidescreen.ini” 中增加以下条目：

```
[Sidescreen]
PAGE100= name:=RMSPage, implementation:=SlSideScreenPage
```

- 您必须根据现有的 Sidescreen 页面数量为新的页面编号，此处为 100。页面编号必须大于等于 100，以避免和西门子页面冲突。
- 页面名称可自由选择，此处为 RMSPage。
- 页面设计不允许更改，此处为 SlSideScreenPage。

```
[Page_RMSPage]
ELEMENT001= name:=RMSElement001,
implementation:=sleseasyscreen.SlEsSideScreenElement
ELEMENT002= name:=RMSElement002,
implementation:=sleseasyscreen.SlEsSideScreenElement
```

- 该页面内有两个 Sidescreen 元素：RMSElement001 和 RMSElement002。元素名称可以自由选择。
- 元素设计不允许更改，此处为 sleseasyscreen.SlEsSideScreenElement。

最后需要指定在这两个 Sidescreen 元素内显示的 Run MyScreens 定义窗口：

```
[Element_RMSElement001]
PROPERTY001= name:=maskPath, type:=QString,
value:="sidescreenmask.com"
PROPERTY002= name:=maskName, type:=QString, value:="Mask001"
…
[Element_RMSElement002]
PROPERTY001= name:=maskPath, type:=QString,
value:="sidescreenmask.com"
PROPERTY002= name:=maskName, type:=QString, value:="Mask002"
```

将“slsidescreen.ini”保存在 /oem/sinumerik/hmi/cfg 或 /user/sinumerik/hmi/cfg 目录中。

重启 HMI 后，屏幕上便会显示 Run MyScreens 设计窗口。

12.3 在一个现有 Sidescreen 页面中添加 Run MyScreens 设计窗口

在一个现有 Sidescreen 页面中添加一个或多个 Run MyScreens 设计窗口时的步骤和 “在一个单独的 Sidescreen 页面中显示多个 Run MyScreens 设计窗口 (页 336)”一章说明的步骤一样。该操作的唯一区别在于，Sidescreen 元素现在需要添加到一个现有的 Sidescreen 页面上，以便显示 Run MyScreens 设计窗口。

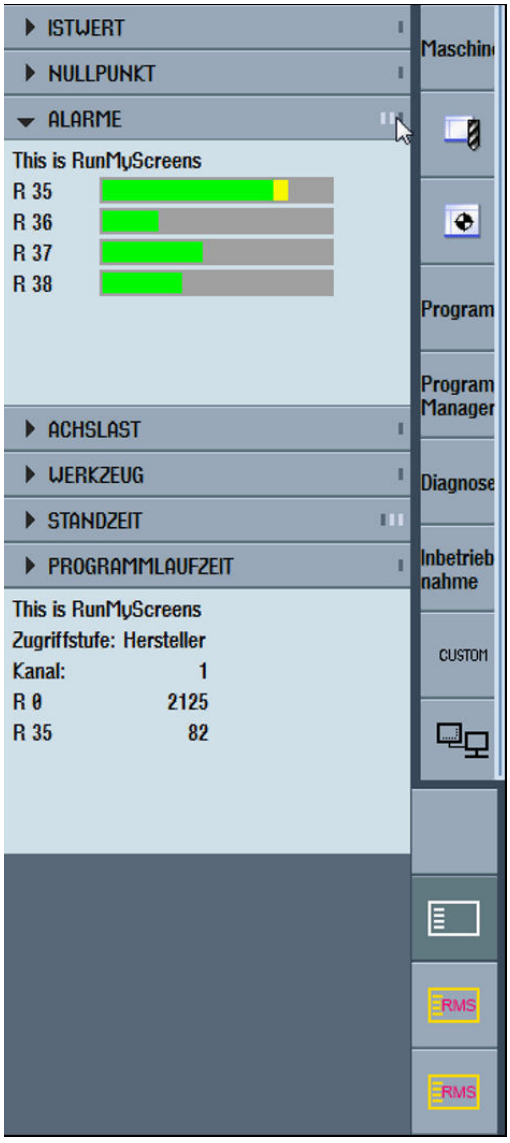


图 12-3 **在一个现有 Sidescreen 页面中添加 Run MyScreens 设计窗口**

说明

在西门子出厂时候提供的 Sidescreen 页面中，只有 WidgetsPage（参见 slsidescreen.ini 文件）可以添加 Run MyScreens 设计窗口。

步骤

在文件 “slsidescreen.ini” 中增加以下条目：

在 [Page_WidgetsPage] 一段中添加一个对应的元素，用于容纳 Run MyScreen 设计窗口。

```
[Page_WidgetsPage]
ELEMENT100= name:=RMSElement,
implementation:=sleseasyscreen.SlEsSideScreenElement
```

- 新元素的编号必须大于等于 100，以避免和西门子提供的显示元素冲突。RMSElement 便会根据该编号添加到 WidgetsPage 中，位于西门子元素下方。

```
[Element_RMSElement]
PROPERTY001= name:=maskPath, type:=QString,
value:="sidescreenmask.com"
PROPERTY002= name:=maskName, type:=QString, value:="Mask001"
```

12.4 在一个 Sidescreen 元素中添加 Run MyScreens 设计窗口

下面会举例说明详细步骤，在该示例中，Sidescreen 页面需要添加一个 Sidescreen 元素。而在该 Sidescreen 元素中需要并排显示两个 Run MyScreens 设计窗口。

按下文为一个 Sidescreen 页面设计一个 Sidescreen 元素。然后为该 Sidescreen 元素指定两个 Sidescreen 窗体，各包含一个 Run MyScreens 设计窗口。

步骤

在文件 “slsidescreen.ini” 中增加以下条目：

```
[Sidescreen]
PAGE100= name:=RMSPage, implementation:=SlSideScreenPage
```

- 页面设计不允许更改，此处为 SlSideScreenPage。

```
[Page_RMSPage]
ELEMENT001= name:=RMSElement, implementation:=
SlSideScreenElement
```

- 元素设计不允许更改，此处为 SlSideScreenElement。

```
[Element_RMSElement]
TextId=RMS_ELEMENT_TITLE
TextFile=rmstexts
TextContext=rmstexts
TextContext=rmstexts
WIDGET001= name:=RMSWidget001,
implementation:=sleseasyscreen.SlEsSideScreenWidget
WIDGET002= name:=RMSWidget002,
implementation:=sleseasyscreen.SlEsSideScreenWidget
```

- 在 Sidescreen 元素的 SlSideScreenElement 设计中，元素会包含一个标题栏，其中可以显示一条文本（见 [Page_RMSPage] 段落）。
  该文本可通过条目 TextId、TextFile 和 TextContext 设计：
  – TextId:待显示文本的 ID
  – TextFile:保存文本的文件
  – TextContext:文本在文件中的上下文
  每种语言可以有一份单独的文件。
  该文件的名称格式为：
  <TextFile>_<语言缩写>.ts，比如：针对上面的例子，名称为 rmstexts_deu.ts。
  没有定义文件和上下文时，即 TextFile=<empty>且 TextContext=<empty>，系统以 Text-Id 显示文本。Text-Id 可以是任意字符串。
  文本文件（比如：rmstexts_deu.ts）由以下部分组成：

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE TS>
<TS>
 <context>
 <name>rmstexts</name>
 <message>
  <source>我的应用</source>
 <translation>My Application</translation>
 </message>
 </context>
</TS>
```

  将文件保存在目录 /oem/sinumerik/hmi/lng 或 /user/sinumerik/hmi/lng 中。HMI 重启后，便会将该文件转换为运行所需的格式。
- 元素包含的窗体的名称可以自由选择，此处为 RMSWidget001 和 RMSWidget002。
- 窗体设计不允许更改，此处为 sleseasysscreen.SlEsSideScreenWidget。

```
[Widget_RMSWidget001]
PROPERTY001= name:=maskPath, type:=QString,
value:="mymask001.com"
PROPERTY002= name:=maskName, type:=QString, value:="MyMask001"

[Widget_RMSWidget002]
PROPERTY001= name:=maskPath, type:=QString,
value:="mymask002.com"
PROPERTY002= name:=maskName, type:=QString, value:="MyMask002"
```

- Sidescreen 窗体配置段落的名称格式为：前缀“Widget_”加目标窗体的名称。
- 在“maskPath”属性中输入包含了 RunMyScreens 设计窗口文件的名称，此处为 mymask001.com 或 mymask002.com。
- 在“maskName”属性中输入需要显示的 RunMyScreens 设计窗口的名称，此处 MyMask001 或 MyMask002。

12.5 在 Sidescreen 中执行 Run MyScreens 设计窗口时的注意事项

请注意如下提示：

- 在 Sidescreen 中执行 Run MyScreens 设计窗口时，函数 LS、LM、DLGL、EXIT 以及 EXITLS 不可使用。
- 无法在 LOAD 功能快块中查询窗口大小。
- 文本始终采用 SINUMERIK Operate 在屏幕分辨率 640x480 像素下使用的字体。
- 各操作单元的定义位置不变，比如：不会拉伸。
- 对话框大小会自动调整。
- 在一个 Sidescreen 页面中执行 Run MyScreens 设计窗口时，该设计窗口会占据整个页面。在 Sidescreen 元素或窗体中执行设计窗口时，该设计窗口占据的页面大小由系统视具体条件而定。
- 标题和软键不可设置。
- 所有调试和错误消息都按顺序依次写入文本文件 easyscreen_log.txt 中。不提供特别的标记来指出哪条消息来自哪个设计窗口。
- 在隐藏了设计窗口后，Sidescreen 中显示的 Run MyScreens 设计窗口的状态便被忽视。

说明

将多个 Run MyScreens 设计窗口集成到 Sidescreen 中后，有时多个 Run MyScreens 设计窗口会同时执行。请注意，该情况会增加系统负载，降低整个系统的性能，具体程度视使用的硬件有所不同。

在 Display Manager 中设计 13

13.1 在 Display Manager 中显示 Run MyScreens 设计窗口

需要在 Display Manager 中显示 Run MyScreens 设计窗口时，您可以使用 SlSideScreenDialog 对话框。该对话框可以根据空余的空间显示一个或多个 Sidescreen 页面，参见 IM9 章节 3.13。多个页面会以分列形式并排显示。每一列占据相同的宽度。您需要配置待显示页面的数量及其内容。每个页面都有自己的一份配置文件。这些配置文件的名称在文件 “systemconfiguration.ini” 对话框配置段落中的指令行参数“cmdline”中确定。第一列页面的配置文件的名称跟在参数名称“–sidescreen1”后面，第二列页面的配置文件的名称跟在参数名称“–sidescreen2”后面，以此类推。

示例

```
[dialogs]
DLG500= name:=RMSApp,
implementation:=sldmsidescreenapp.SlSideScreenDialog,
process:=SlHmiHost1, preload:=false, cmdline:="-sidescreen1
rmspage1.ini -sidescreen2 rmspage2.ini -spacing 3"
```

在本例中创建的对话框 SlSideScreenDialog 的实例名称为 RMSApp。该对话框有两个页面/列。这两页面的配置文件分别为：“rmspage1.ini” 和 “rmspage2.ini”。两列之间间隔 3 个像素。

说明

页面配置文件的名称可以自由选择，此处为 rmspage1.ini 和 rmspage2.ini。

在 Display Manager 中，Run MyScreens 设计窗口的使用方式和在 Sidescreen 中设计 (页 333)一章中介绍的一样。唯一的区别在于，此处设计窗口不是通过文件 “slsidescreen.ini” 集成，而是通过为现有页面配置文件进行。

需要在 SINUMERIK Operate 的 Display Manager 中集成 Run MyScreens 设计窗口时，您有两种方法：相关内容将在后面的章节进行说明。

13.2 集成到 SISideScreenDialog 中

需要在该功能中集成 Run MyScreens 设计窗口时，您需要在文件 “systemconfiguration.ini” 中创建一个对话框实例 SlSideScreenDialog，然后创建一份用于集成 Run MyScreens 设计窗口的文件。

下面我们举例说明 Run MyScreens 设计窗口的集成步骤。在本例中，Run MyScreens 设计窗口会占据整个对话框 SlSideScreenDialog，即：只有一个页面或一列。

步骤

首先在文件 “systemconfiguration.ini” 的段落 [dialogs] 中创建一个对话框实例 SlSideScreenDialog。该实例名称为 RMSApp。

```
[dialogs]
DLG500= name:=RMSApp,
implementation:=sldmsidescreenapp.SlSideScreenDialog,
process:=SlHmiHost1, preload:=false,
cmdline:="-sidescreen1 rmsapp.ini"
```

接着在文件 “rmsapp.ini” 中配置或者集成 RunMyScreens 设计。

```
[Sidescreen]
PAGE001= name:=RMSPage, implementation:=SlSideScreenPage
```

- 创建名为“RMSPage”的新页面。

```
[Page_RMSPage]
ELEMENT001= name:=RMSElement,
implementation:=sleseasyscreen.SlEsSideScreenElement
```

- 在页面 RMSPage 上创建一个名为 RMSElement 的元素。该元素用于显示 Run MyScreens 设计窗口。

```
[Element_RMSElement]
PROPERTY001= name:=maskPath, type:=QString,
value:="sidescreenmask.com"
PROPERTY002= name:=maskName, type:=QString, value:="Mask"
```

- 在“maskPath”属性中输入包含了 RunMyScreens 设计窗口文件的名称，此处为 sidescreenmask.com。
- 在“maskName”属性中输入需要显示的 RunMyScreens 设计窗口的名称，此处为 Mask。

将文件“rmsapp.ini”和“systemconfiguration.ini”（见上文）保存在目录 /oem/sinumerik/hmi/cfg 或 /user/sinumerik/hmi/cfg 中。

重启 HMI 后，屏幕上便会显示 Run MyScreens 设计窗口。

13.3 集成到 SIWidgetsApp 中

下面介绍如何将一个 Run MyScreens 设计窗口集成到西门子出厂时提供的 SIWidgetsApp 应用程序中。

根据 Run MyScreens 设计窗口集成到的 SIWidgetsApp 目标页面/列，选择合适的文件：“sldmwidgets1.ini” 或 “sldmwidgets2.ini”。

下面我们举例说明何将 Run MyScreens 设计窗口集成到应用程序 SIWidgetsApp 的左侧页面 / 列。

步骤

在文件 “sldmwidgets1.ini” 中增加以下条目：

```
[Sidescreen]
PAGE001= name:=WidgetsPage, implementation:=SlSideScreenPage
[Page_WidgetsPage]
ELEMENT100= name:=RMSElement,
implementation:=sleseasyscreen.SlEsSideScreenElement
```

- 创建一个名为 RMSElement 的元素，以显示设计 Run MyScreens 窗口。

说明

新元素的编号必须大于等于 100，以避免和西门子提供的显示元素冲突。RMSElement 便会根据该编号添加到 WidgetsPage 中，位于西门子元素下方。

```
[Element_RMSElement]
PROPERTY001= name:=maskPath, type:=QString,
value:="sidescreenmask.com"
PROPERTY002= name:=maskName, type:=QString, value:="Mask"
```

- 在“maskPath”属性中输入包含了 RunMyScreens 设计窗口文件的名称，此处为 sidescreenmask.com。
- 在“maskName”属性中输入需要显示的 RunMyScreens 设计窗口的名称，此处为 Mask。

将文件“sldmwidgets1.ini”保存在目录 /oem/sinumerik/hmi/cfg 或 /user/sinumerik/hmi/cfg 中。

重启 HMI 后，屏幕上便会显示 Run MyScreens 设计窗口。

13.4 在 Display Manager 中执行 Run MyScreens 设计窗口时的注意事项

请注意如下提示：

- 在 Display Manager 中执行 Run MyScreens 设计窗口时，函数 LS、LM、DLGL、EXIT 以及 EXITLS 不可使用。
- 无法在 LOAD 功能快块中查询窗口大小。
- 文本始终采用 SINUMERIK Operate 在屏幕分辨率 640x480 像素下使用的字体。
- 各操作单元的定义位置不变，比如：不会拉伸。
- 在一个 Sidescreen 页面中执行 Run MyScreens 设计窗口时，该设计窗口会占据整个页面。在 Sidescreen 元素或窗体中执行设计窗口时，该设计窗口占据的页面大小由系统视具体条件而定。
- 所有调试和错误消息都按顺序依次写入文本文件 easyscreen_log.txt 中，不提供特别的标记来指出哪条消息来自哪个设计窗口。
- 在隐藏了设计窗口后，Display Manager 中显示的 Run MyScreens 设计窗口的状态便被忽视。

说明

将多个 Run MyScreens 设计窗口集成到 Display Manager 中后，有时多个 Run MyScreens 设计窗口会同时执行。请注意，该情况会增加系统负载，降低整个系统的性能，具体程度视使用的硬件有所不同。

Referenz - Listen

A

A.1 登入软键表

A.1.1 车床登入软键表

车床的程序操作区

HSK1	HSK2	HSK3	HSK4	HSK5	HSK6	HSK7	HSK8
编辑	**钻削**	**车削**	**车削轮廓**	**铣削**	**其它**	**模拟**	**NC 选择**
HSK9	HSK10	HSK11	HSK12	HSK13	HSK14	HSK15	HSK16
--	直线 圆弧 （仅限 ShopTurn 程序）	--	--	测量工件	测量刀具	**OEM**	--

车削

在下表中列出了车削工艺中可能的登入软键。 各登入软键的分配根据不同的系统可能会有所不同。 给定的 OEM 软键允许用于 Run MyScreens。

programGUIDE（G 代码）登入软键：

	钻削	**车削**	**车削轮廓**		**铣削**		**其它**	
	HSK2	**HSK3**	**HSK4**		**HSK5**		**HSK6**	
VSK1	钻中心孔	轮廓车削	轮廓	--	端面铣削	轮廓	设置	高速设定
VSK2	钻削铰孔	凹槽	轮廓车削	--	型腔	轨迹	平面回转	平行轴
VSK3	深孔钻削	退刀槽	余料车削	--	多边形凸台	预钻削	刀具回转	--
VSK4	镗孔	螺纹	槽式车削	--	槽	型腔	--	--

VSK5	螺纹	切断	余料槽式车削	--	螺纹铣削	型腔余料	--	--
VSK6	**OEM**	--	往复车削	--	雕刻	凸台	子程序	--
VSK7	位置	**OEM**	余料往复车削	**OEM**	**OEM**	凸台余料	--	**OEM**
VSK8	重复位置	--	>>	<<	轮廓铣削	<<	>>	<<

ShopTurn 登入软键:

	钻削	**车削**	**车削轮廓**		**铣削**		**其它**		
	HSK2	**HSK3**	**HSK4**		**HSK5**		**HSK6**		**HSK10**
VSK1	中心钻孔	轮廓车削	新建轮廓	--	端面铣削	新建轮廓	设置	高速设定	刀具
VSK2	钻中心孔	凹槽	轮廓车削	--	型腔	轨迹	平面回转	平行轴	直线
VSK3	钻削铰孔	退刀槽	余料车削	--	多边形凸台	预钻削	刀具回转	重复程序	圆心
VSK4	深孔钻削	螺纹	槽式车削	--	槽	型腔	副主轴	--	圆半径
VSK5	螺纹	切断	余料槽式车削	--	螺纹铣削	型腔余料	转换	--	极坐标
VSK6	**OEM**	--	往复车削	--	雕刻	凸台	子程序	--	逼近/返回
VSK7	位置	**OEM**	余料往复车削	**OEM**	**OEM**	凸台余料	--	**OEM**	--
VSK8	重复位置	--	>>	<<	轮廓铣削	<<	>>	<<	--

A.1.2 铣床登入软键表

铣床的程序操作区

HSK1	HSK2	HSK3	HSK4	HSK5	HSK6	HSK7	HSK8
编辑	**钻削**	**铣削**	**轮廓铣削**	**车削** （仅限 G 代码程序）	**其它**	**模拟**	**NC 选择**

HSK9	HSK10	HSK11	HSK12	HSK13	HSK14	HSK15	HSK16
--	直线 圆弧 （仅限 ShopMill 程序）	--	--	测量工件	测量刀具	**OEM**	--

铣削

在下表中列出了铣削工艺中可能的登入软键。 各登入软键的分配根据不同的系统可能会有所不同。 给定的 OEM 软键允许用于 Run MyScreens。

programGUIDE（G 代码）登入软键：

	钻削	**铣削**	**轮廓铣削**		**车削**		**其它**	
	HSK2	**HSK3**	**HSK4**		**HSK5**		**HSK6**	
VSK1	钻中心孔	端面铣削	轮廓	--	轮廓车削	轮廓	设置	--
VSK2	钻削铰孔	型腔	轨迹	--	凹槽	轮廓车削	平面回转	平行轴
VSK3	深孔钻削	多边形凸台	预钻削	--	退刀槽	余料车削	刀具回转	--
VSK4	镗孔	槽	型腔	--	螺纹	槽式车削	高速设定	--
VSK5	螺纹	螺纹铣削	型腔余料	--	切断	余料槽式车削	--	--
VSK6	**OEM**	雕刻	凸台	--	--	往复车削	子程序	--
VSK7	位置	**OEM**	凸台余料	**OEM**	**OEM**	余料往复车削	--	**OEM**
VSK8	重复位置	--	>>	<<	车削轮廓	<<	>>	<<

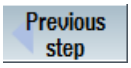
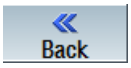
ShopMill 登入软键:

	钻削	**铣削**	**轮廓铣削**		**其它**		**直线圆弧**
	HSK2	**HSK3**	**HSK4**		**HSK6**		**HSK10**
VSK1	钻中心孔	端面铣削	新建轮廓	--	设置	--	刀具
VSK2	钻削铰孔	型腔	轨迹	--	平面回转	平行轴	直线
VSK3	深孔钻削	多边形凸台	预钻削	--	刀具回转	重复程序	圆心
VSK4	镗孔	槽	型腔	--	高速设定	--	圆半径
VSK5	螺纹	螺纹铣削	型腔余料	--	转换	--	螺旋线

VSK6	**OEM**	雕刻	凸台	--	子程序	--	极坐标
VSK7	位置	**OEM**	凸台余料	**OEM**	--	**OEM**	--
VSK8	重复位置	--	>>	<<	>>	<<	--

A.2 预定义软键列表

表格 A-1 预定义软键

名称	软键
SOFTKEY_OK	
SOFTKEY_CANCEL	
SOFTKEY_APPLY	
SOFTKEY_MORE	
SOFTKEY_BACK	
SOFTKEY_ASSISTANT_NEXT	
SOFTKEY_ASSISTANT_PREVIOUS	
SOFTKEY_NAV_BACK	

A.3 访问等级列表

不同的访问等级有以下不同的含义：

保护等级	禁用密码	区域
ac0	预留用于西门子	
ac1	密码	机床制造商
ac2	密码	维修
ac3	密码	用户
ac4	钥匙开关位置 3	编程员，调试员
ac5	钥匙开关位置 2	合格的操作员
ac6	钥匙开关位置 1	受过培训的操作员
ac7	钥匙开关位置 0	学过相关内容的操作员

A.4 颜色表

系统颜色

提供统一的颜色表用于对话框设计（各个标准颜色的色块）。对于单元（文本、输入栏、背景等等）可以从 0 到 133 号的颜色中选择一种。

除了预定义的颜色，您也可以使用 RGB 值(“#RRGGBB”)来给定颜色。

索引	图标	颜色	颜色描述
1		黑色	
2		桔黄色	
3		深绿色	
4		浅灰	
5		深蓝色	
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

A.5 文件名中的语种缩写表

支持的语言

标准语言：

语言	文件名中的缩写
简体中文	chs
德语	deu
英语	eng
法语	fra
意大利语	ita
西班牙语	esp

其他语言：

语言	文件名中的缩写
繁体中文	cht
丹麦语	dan
芬兰语	fin
印度尼西亚语	ind
日语	jpn
韩语	kor
马来西亚语	msl
荷兰语	nld
波兰语	plk
葡萄牙语（巴西）	ptb
罗马尼亚语	rom
俄语	rus
瑞典语	sve
斯洛伐克语	sky
斯洛文尼亚语	slv
泰语	tha
捷克语	csy

语言	文件名中的缩写
土耳其语	trk
匈牙利语	hun
越南语	vit

A.6 可用的系统变量列表

文档

参数手册 系统变量 /PGAsl/

A.7 对话框打开方式（属性 CB）

下表列出了在哪些条件下调用 CHANGE 方法。

属性 CB 的适用条件：

CB0

如果变量当前有一个有效值（例如：通过预设值或 NC/PLC 变量），则通过显示屏幕触发 CHANGE 方法。

CB1（默认）

通过显示屏幕无法显式触发 CHANGE 方法。如果变量有一个配置的 NC/PLC 变量，则一定会调用 CHANGE 方法。

条件				响应
类型	系统或者用户变量	预设值	编辑 CHANGE 方法	
输入/输出栏	是	是	是	有了配置的 NC/PLC 变量，系统至少会通过当前 NC/PLC 变量值自动调用一次 CHANGE 方法。此时，CB 的预设值没有用。
			否	
		否	是	
			否	
	否	是	是	通过预设值调用 CHANGE 方法。
			否	不调用 CHANGE 方法。
		否	是	由于当前没有可以用来调用 CHANGE 方法的有效值，因此不调用。
			否	
转换	是	是	是	有了配置的 NC/PLC 变量，系统至少会通过当前 NC/PLC 变量值自动调用一次 CHANGE 方法。此时，CB 的预设值没有用。
			否	
		否	是	
			否	
	否	是	是	通过预设值调用 CHANGE 方法。
			否	不调用 CHANGE 方法。
		否（转换列表中的第一个值默认为预设值）	是	通过转换列表中的第一个值（预设值）来调用 CHANGE 方法。
			否	不调用 CHANGE 方法。

提示和技巧

B

B.1 一般提示

- 可视情况将 BTSS 变量用作系统变量($变量)。
  原因：
  这样可避免可能较为费时的名称分析。
  示例：
  最好使用"/通道/参数/R[u1,10]"替代"$R[10]"。
- 避免循环（同类）设置，例如屏幕和变量属性 HLP。 比较之前要设置的值与当前值，不相同时再真正设置该值。
  原因：
  避免在查找与分辨率相关的辅助画面上费时。
  示例：

```
DEF MyVar=(R3///,"X1",,"mm"/WR1//"$AA_IM[0]")
CHANGE(MyVar)
   IF MyVar.VAL < 100
      HLP="mypic1.png"
   ELSE
      HLP="mypic2.png"
   ENDIF
END_CHANGE
```

建议：

```
CHANGE(MyVar)
   IF MyVar.VAL < 100
      IF HLP <> "mypic1.png"
         HLP="mypic1.png"
      ENDIF
   ELSE
      IF HLP <> "smypic2.png"
         HLP="mypic2.png"
```

```
      ENDIF
   ENDIF
END_CHANGE
```

- 使用一个 MRNP()函数替代多个连续的 RNP()函数。
  使用一个 MRDOP()函数替代多个连续的 RDOP()函数。
  原因：
  降低通讯负载并提升性能。
- 读取驱动参数时不要短于•1•秒的周期，时间长些更佳。
  原因： 否则与驱动的通讯可能会受到很大干扰，甚至导致故障。
- 避免随之发生的与系统或用户变量的计算。 为此请使用，例如寄存器(REG[x])或(不可见)辅助变量。
  原因： 每次赋值都会导致向所连接的系统或用户变量中写入。
- 出于对清晰、可维护和性能（显示屏幕时）的考虑，不同块中所使用的同类代码应放在一个 SUB 块中。 这可在相应位置上使用函数 CALL()进行调用。
- 通过对对话栏（在 slguiconfig.xml 中设置）中 CPU 负载的查看可以确定屏幕上的哪些更改会在多大程度上影响性能。

B.2 DEBUG 的提示

- 函数 DEBUG()用于进行诊断。调用函数 DEBUG()时返回的字符串会写入 “easyscreen_log.txt” 文件中。函数 DLGL()对向对话栏输出信息也会有帮助。屏幕开发阶段结束后，出于对性能的考虑应将该函数删除或取消。
- 日志文件 “easyscreen_log.txt” 在屏幕开发完成后始终为空。

B.3 CHANGE 方法的提示

- CHANGE 方法总是只持续很短的时间，特别是那些与系统或用户变量相连并且频繁变化的变量。
  原因：
  提升屏幕性能。
- 视情况请不要在 CHANGE 方法中设置 RNP()函数。 而是最好与要读取的系统或用户变量一起创建一个不可见的变量并使用。
  原因：
  每次调用都会强制取消 RNP()函数。 否则会直接存取已存在的当前值。

示例：

此处会在轴运行每次变化时通过 RNP()进行名称分析，以读取通道专用的机床数据：

```
DEF AXIS_POSITION_X =  (R///,""///"$AA_IM[X]")

CHANGE(AXIS_POSITION_X)
      DLGL("Axis """ << RNP("$MC_AXCONF_GEOAX_NAME_TAB[0]") <<
""" has moved: "
<< AXIS_POSITION_X)
END_CHANGE
```

借助不可见变量保持通道专用机床数据的最新状态，每个值变化都会复制到临时变量中，例如寄存器。

该临时变量可以在轴位置变化的 CHANGE 方法中使用，无需每次都进行机床数据的名称分析以及相应的读访问：

```
DEF AXIS_POSITION_X =  (R///,""///"$AA_IM[X]")
DEF AXIS_NAME_X = (S///,""/WR0//"$MC_AXCONF_GEOAX_NAME_TAB[0]")

CHANGE(AXIS_NAME_X)
      REG[0] = AXIS_NAME_X
END_CHANGE

CHANGE(AXIS_POSITION_X1)
```

```
   DLGL("Axis """ << REG[0] << """ has moved " <<
AXIS_POSITION_X)
END_CHANGE
```

- 刷新率以及与频繁变化的系统或用户变量相连接的变量的 CHANGE 方法的执行速度会由于变量属性 UR 的使用而降低，例如与轴实际值耦合的变量。
  原因：
  由此，值变化时相应的 CHANGE 方法将在固定设置的光栅中执行。

B.4 DO-LOOP 循环的提示

取决于配置，由于循环会影响 SINUMERIK Operate 的性能，因此请您慎重使用这些循环并且视情况放弃使用循环中费时的动作。

同时应使用寄存器(REG[])作为运行变量，因为一般显示变量（特别是那些使用 BTSS 连接的）会极其频繁地刷新或写入，这同样会影响系统性能

请注意，程序块中执行的脚本行的数量目前限制为 10000 行。如果达到该数量，文件 “easyscreen_log.txt” 中的相应记录会中断脚本执行。

原因：

- 防止无限循环
- “Run MyScreens” 必须保持可操作

动画元素

C

C.1 简介

简介

本手册介绍的是通过 X3D 浏览器进行作业，借助该浏览器可将动态和静态图形场景（动画元素）- 以下称作辅助画面 - 集成到 SINUMERIK Operate（V4.7 SP1 起）的图形界面中。

您可在上下文关联的辅助画面中显示目标运动过程和参数。这样可以提升应用的可操作性并设计相应的用户界面。

从实施理念到完成辅助画面共包含以下几步：

- 创建用于最后可显示在 X3D 浏览器中的辅助画面的图形元素和 3D 模型（参见章节 建模 (页 369)）。
- 创建场景说明文件，在该文件中图形文件的模型数据被分配给待显示的场景和动画（参见章节 场景说明文件的结构 (页 375)）。
- 将 X3D 和 XML 文件转换为 HMI 文件（参见章节 转换为 hmi 文件 (页 380)）。
- 将 X3D 浏览器以 C++ 语言格式集成到自身应用中（参见章节 执行示例 (页 383)）。
- 在 Run MyScreens 中使用（参见章节 在 Run MyScreens 中显示 (页 384)）。

文件格式一览

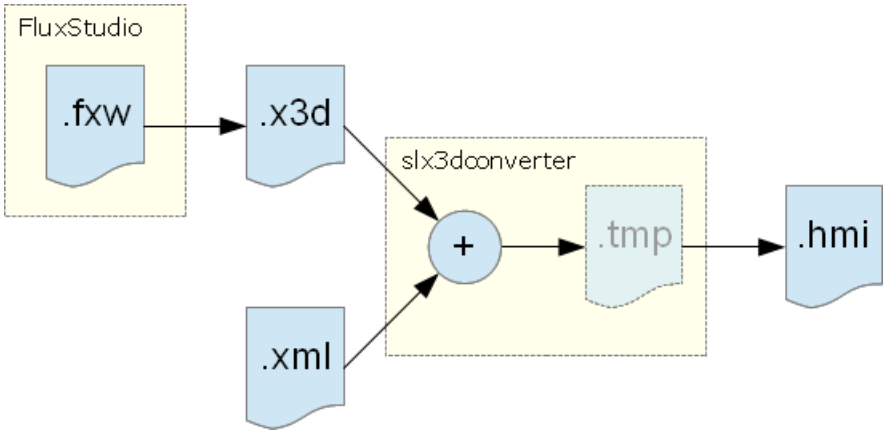


图 C-1 文件格式

文件格式	说明
.fxw	Vivaty Studio 的图形文件
.x3d	交换格式的模型文件
.xml	动画和场景的定义文件
.hmi	X3D 浏览器的输出文件

C.2 建模

C.2.1 前提条件

建模的目的在于创建一个含生成的 3D 模型（文件格式为 .x3d）和一个 3D 内容官方标准的文件。

建模在 **Vivaty Studio** 中进行。Vivaty Studio 是一个可以提供多种导出和导入方式的免费 3D 建模工具。

前提条件

- 已安装 Vivaty Studio，版本 1.0。
- 知道如何进行建模和使用 Vivaty Studio。
- 本说明中没有对文件格式 .x3d 作进一步说明，需提前作相应的了解。

说明

Vivaty Studio 可从以下链接 (https://www.web3d.org/projects/vivaty-studio)下载。

C.2.2 建模规则

建模时请注意以下规则。之后处理辅助画面文件时需遵循这些规则。

建模规则

1. TimeSensor 的名称必须是“动画”。
2. 所有配套元素都必须被分配到一个组中。
3. 所有组都必须设为以下位置：
   x = 0，y = 0，z = 0
4. 如需隐藏组，则须按如下方式设置翻译值：
   x = 1000000，y = 1000000，z = 1000000
5. 下列元素在 Vivaty Studio 图形模型和 XML 场景定义（参阅 一览 (页 375) 一章）中的名称必须相同：
   - 刀具
   - 摄像机

6. 如需创建一个 .x3d 文件，则须在对话框“导出选项”中进行以下设置：

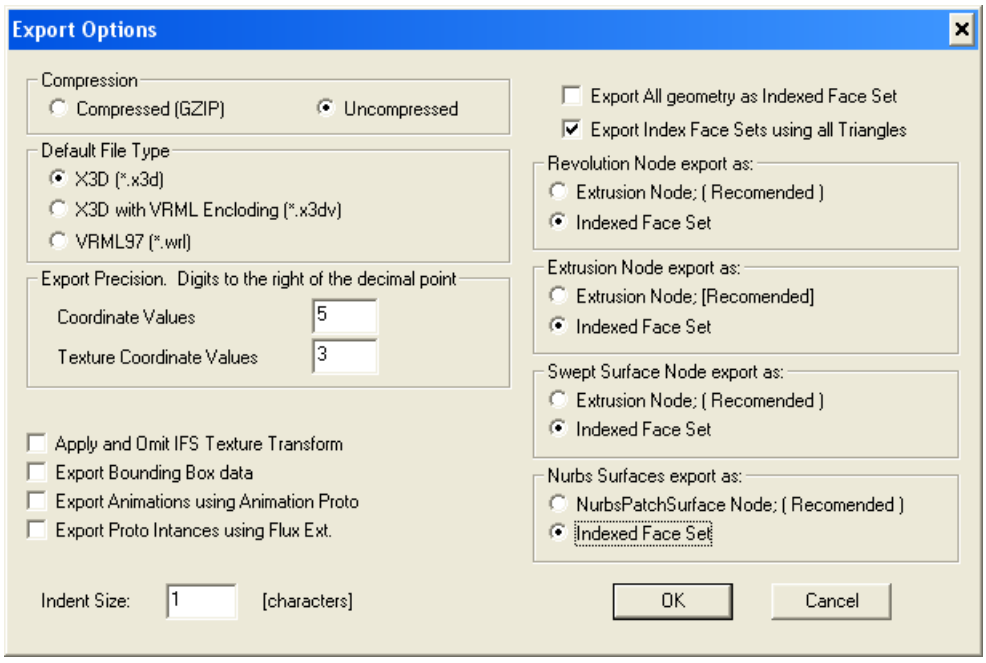


图 C-2　对话框“导出选项”中的设置

7. 对于文本建议采用以下设置（另见下列对话框）：
   – 文本必须始终中心对齐。
   – 文本大小应设为 0.2。便于文本进行精准的定位。文本输出是通过 X3D 浏览器进行的，与操作界面中的字体大小无关。

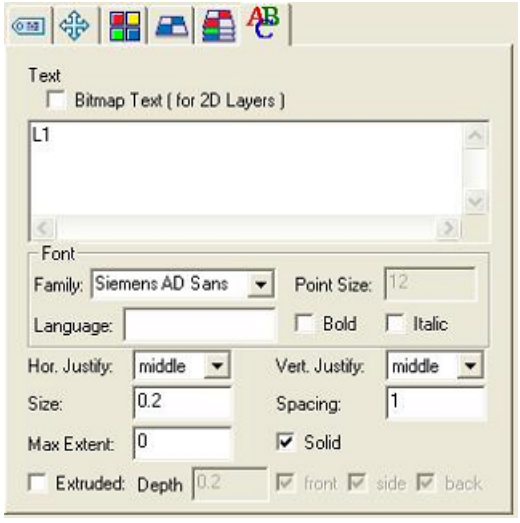


图 C-3　文本设置

8. 如需创建一个阴影线，应将文件 schnitt.png 用作结构。阴影线的走线方式始终是左下到右上，成 45°度角。两个尺寸中的缩放比例都应设为 3。旋转取决于结构形式且必须手动调整。

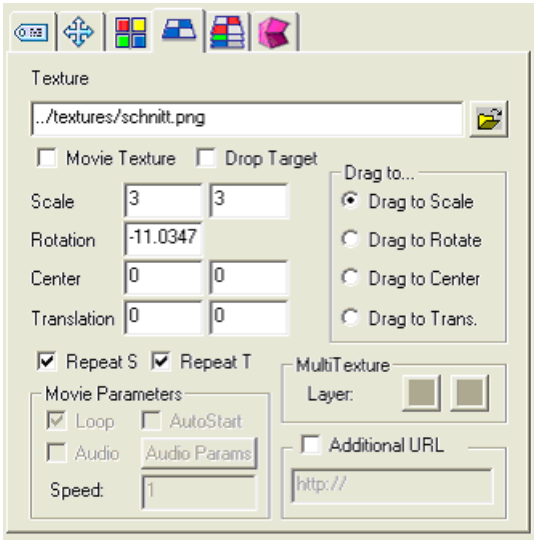


图 C-4　阴影线设置

9. 毛坯的大小/尺寸建议按如下方式设置：
   – 用于车削加工的圆柱体：
     长度 = 5.5；半径 = 1.4（参阅模板 turning_blank.fxw）
   – 用于铣削加工的长方六面体：
     x = 4.75；y = 3；z = 3（参阅模板 milling_blank.fxw）

参见

XML 指令 (页 375)

C.2.3　导入图形（模型）

在 Flux Studio 中可以导入外部图形（模型）。 常用模型（如刀具）可作为 .x3d 或 .hmi 文件集中保存并作为完成的元素在其他辅助画面中重复使用。 随附的建模模板列表请见 建模模板 (页 373) 一章。

复杂的对象也可通过其他建模工具（如 Cinema4D）创建和导入。

下文介绍了如何重复使用这些模型。

作为 “Inline” 导入

Flux Studio 提供了对象 “Inline” 用于导入 .x3d 文件。 这样可以关联外部元素，而不增加这些模型的 3D 数据。

通过菜单项 **Create** -> **Create Inline** 插入对象。 之后在对象属性中输入文件名。

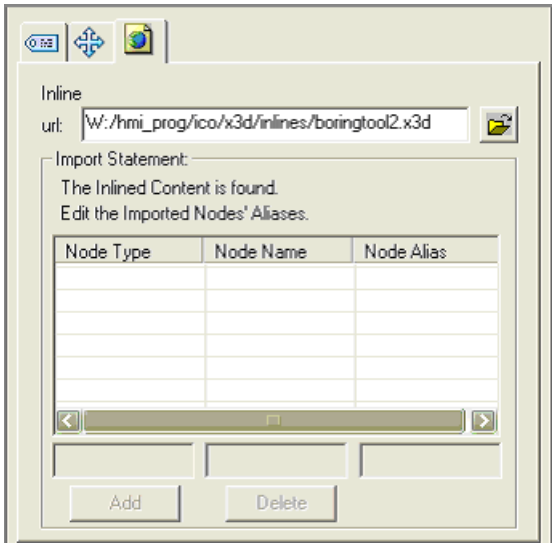


图 C-5　通过对象 “Inline” 导入

导入 3D 模型数据

按如下步骤导入一个文件中的模型数据。

1. 通过菜单项 **File** -> **Import Other Format** 打开对话框 “Flux Studio Accutrans3D Translation Utility”。
2. 通过下拉菜单 “File Types” 选择相应的格式。 例如：如需导入一个 Cinema4D 文件，则须选择文件类型 “3D Studio”。
3. 然后通过 **Select Files** 选择所需的文件并开始导入。

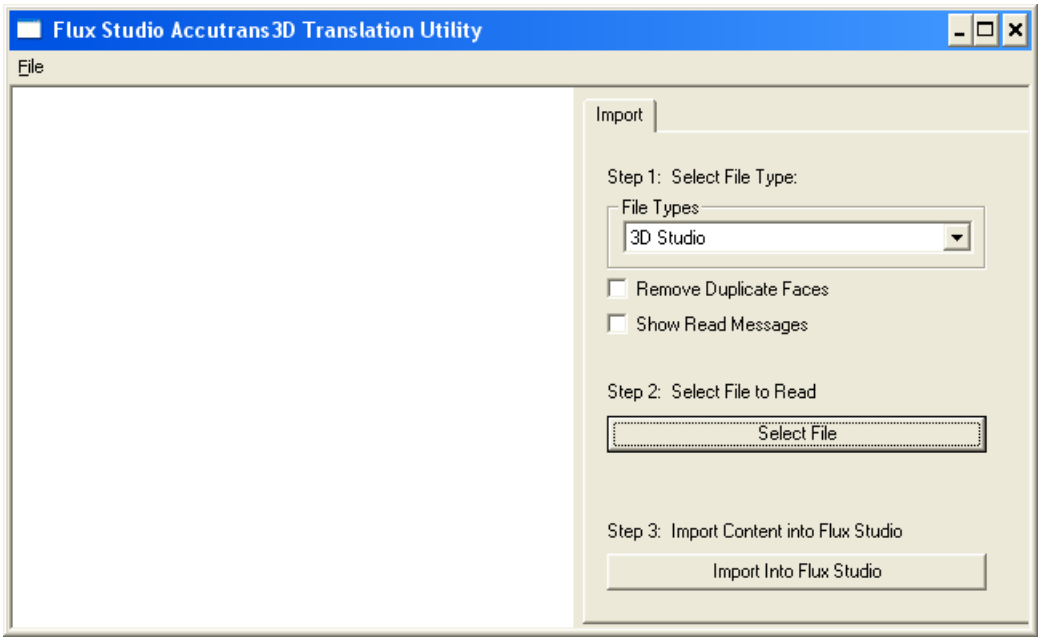


图 C-6　导入 3D 模型数据

将 Cinema4D 文件中的模型数据作为整个组插入。 删除一并导入的所有摄像机和其他信息。只需要 Cinema4D 文件的图形元素。

C.2.4 建模模板

本章节中列举了一个作为外形、颜色和尺寸基础的图形元素列表。 为确保操作界面中的“外观”一致，建议使用模板格式或模板创建自有图形。

概述

模板	说明
intersection_texture.png	截面结构
rapid_traverse_line_hori.fxw	用于阐明刀具运行路径的水平线（快速模式）

模板	说明
rapid_traverse_line_vert.fxw	用于阐明刀具运行路径的垂直线（快速模式）
feed_traverse_line_hori.fxw	用于阐明刀具路径的水平线（进给模式）
feed_traverse_line_vert.fxw	用于阐明刀具路径的垂直线（进给模式）
dimensioning_lines_hori.fxw	水平尺寸辅助线
dimensioning_lines_vert.fxw	垂直尺寸辅助线
z1_inc.fxw	水平尺寸线 Z1
x1_inc.fxw	垂直尺寸线 X1
3d_coordinate_origin.fxw	3D 坐标原点
3d_zero_point.fxw	3D 零点

车削

模板	说明
turning_blank.fxw	用于车削工艺的毛坯：一个简易圆柱体
turning_centerline.fxw	中心线
turning_centerpoint.fxw	用于显示 WCS 中零点的坐标原点
turning_refpoint.fxw	加工参考点
turning_machining_area.fxw	加工面的边界线

铣削

模板	说明
milling_blank.fxw	用于车削工艺的毛坯：一个简易长方六面体
milling_centerline.fxw	中心线
milling_refpoint.fxw	加工参考点

C.3 XML 指令

C.3.1 一览

为了使 X3D 浏览器能够访问图形，需要场景说明文件（*.xml）。在场景说明文件中分配从配置中调用的场景名称及图形所在的*.x3d 文件 (TimeSensor) 中的时间。

C.3.2 场景说明文件的结构

下文显示了场景说明文件的结构并对各个 XML 指令进行了说明：

结构和说明

<!-- 在车削加工时处理平面相关的文本 -->

```
<TextPlane plane="G18_ZXY" />
```

说明

<TextPlane plane="G18_ZXY" />	在车削加工时处理平面相关的文本（轴名称）。 如果使用一个标记为“Z”的文本（G18 第一根几何轴），相应的控制系统几何轴名称则会显示在辅助画面中（例如：“Z”）。

<!--文本对齐 -->

```
<TextPosition center='true' />
```

说明

<TextPosition center='true' />	文本对中的标记。 这有助于在旋转画面中显示文本。 推荐使用此项设置。

<!--调整刀具速度 -->

```
<ToolSpeedFactors planeSpeedFactor="0.4" rapidSpeedFactor="0.7"
reducedSpeedFac¬tor="1.0"/>
```

说明

<ToolSpeedFactors	如需调整刀具速度，可添加该项。
planeSpeedFactor="0.4"	进给速度系数 (0.4 = 40%)
rapidSpeedFactor="0.7"	快速速度系数 (0.7 = 70%)
reducedSpeedFactor="0.1" />	第三方速度系数 (0.1 = 10%)

<!--动画定义-->

```
<SceneKey name='BoringAnimation' masterRotationSpeed="-64.0"
maxRotAngle="45.0" begin-Time='0.110 endTime='0.118 view='camiso'
speedMaster='boringtool' Type="VIEW_3D_TURN_CYL">
```

说明

<SceneKey name='BoringAnimation'	动画名称
masterRotationSpeed="-64.0"	刀具的旋转方向和旋转速度。 该设置为可选项。
maxRotAngle="45.0"	避免混淆效应（刀具似乎按错误的方向旋转）。 该值通常是四分之一的刀具对称。 该设置为可选项。
beginTime='0.110	Timesensor 上的动画起始时间点
endTime='0.118'	Timesensor 上的动画结束时间点
view='camiso'	用于动画的摄像机
speedMaster='boringtool'	动画刀具名称
type="VIEW_3D_TURN_CYL">	视图类型确定视图（参见章节 视图类型 (页 379)）

<!-- 元素 -->

```
<Element name='1' time='0.110' feed='rapid' dwell='2'/>
<Element name='2' time='0.112' feed='rapid'/>
<Element name='3' time='0.114' feed='rapid'/>
<Element name='4' time='0.116' feed='plane'/>
<Element name='5' time='0.118' feed='plane'/>
```

说明

<Element name='1'	表示针对的是第一个动画元素。
time='0.110'	Timesensor 上第一个元素的时间点
feed='rapid'	元素的运行速度 plane = 加工进给 rapid = 快速 reduced = 减速，速度低于 "plane"
dwell='2'/>	动画运行暂停 旋转的刀具保持旋转状态。

<!-- 动画定义结束 -->

```
</SceneKey>
```

说明

</SceneKey>	动画定义结束

<!-- 现有动画作为新动画的来源 -->

```
<SceneKey source=''BoringAnimation' name=''BoringAnimationRear' mirror='
MirrorScreenX ' />
```

说明

<SceneKey source="BoringAnimation'	用作另一个动画来源的动画的名称。
name="BoringAnimationRear'	基于来源的新动画的名称
mirror=' MirrorScreenX ' />	按 X 轴方向镜像，在新动画中输出。

<!-- 静态场景（画面）定义（4 个示例） -->

```
<SceneKey name='Default' time='0.026' view='camiso'
type="VIEW_3D_DRILL_CUT"/>
<SceneKey name='Cut' time='0.046' view='camside' type="VIEW_SIDE"/>
<SceneKey name='Z1ink' time='0.056' view='camside' type="VIEW_SIDE"
 highLightedGroup='dad_Z1ink'/>
<SceneKey name='Z1abs' time='0.066' view='camside' type="VIEW_SIDE"
 highLightedGroup='dad_Z1abs'/>
```

说明

<SceneKeyname='Z1abs'	画面名称
time='0.066'	Timesensor 上的画面时间点
view='camside'	用于画面的摄像机
type="VIEW_SIDE"	视图类型确定视图（参见章节 视图类型 (页 379)）
highLightedGroup='dad_Z1abs'/>	需要加亮的组的名称。 组在 FluxStudio 中已被命名为 "Z1abs"。 前缀 "dad_" 由 Flux 自动添加。

<!-- end of xml file –!>

C.3.3 镜像和旋转

通过指令 **mirror** 确定画面/模型的镜像或旋转。

说明

mirror="RotateScreenX"	画面围绕 X 轴旋转
mirror="RotateScreenY"	画面围绕 Y 轴旋转
mirror="RotateScreenZ"	画面围绕 Z 轴旋转
mirror="MirrorScreenX"	画面按 X 轴方向镜像
mirror="MirrorScreenY"	画面按 Y 轴方向镜像
mirror="MirrorScreenZ"	画面按 Z 轴方向镜像

mirror="RotatePieceX"	模型围绕 X 轴旋转
mirror="RotatePieceY"	模型围绕 Y 轴旋转
mirror="RotatePieceZ"	模型围绕 Z 轴旋转
mirror="MirrorPieceX"	模型按 X 轴方向镜像
mirror="MirrorPieceY"	模型按 Y 轴方向镜像
mirror="MirrorPieceZ"	模型按 Z 轴方向镜像

所有旋转和镜像方式都可组合使用。 旋转需要确定旋转角度。

示例

mirror="RotatePieceZ=180"

mirror="RotatePieceZ=-90 MirrorScreenX"

mirror="RotateScreenY=90"

mirror="MirrorPieceX MirrorPieceY"

mirror="MirrorPieceZ RotatePieceX=-90"

C.3.4 视图类型

通过视图类型确定视图。 视图以适用于显示对象的经验值和规定为基础。
借此确保对象的显示与操作界面坐标系（MD 52000 $MCS_DISP_COORDINATE_SYSTEM 或 MD 52001 $MCS_DISP_COORDINATE_SYSTEM_2）的设置相匹配。

说明

"VIEW_STATIC"	无需转换的直接视图
"VIEW_3D_TURN_CYL"	车削加工的 3D 视图（圆柱体）
"VIEW_3D_MILL_CUBE"	铣削加工的 3D 视图（长方六面体）
"VIEW_3D_DRILL_CUT"	钻削加工的 3D 视图（剖视图）
"VIEW_SIDE"	钻削加工的 2D 视图（剖视图）
"VIEW_TOP_GEO_AX_1"	第 1 根几何轴方向上的 2D 视图
"VIEW_TOP_GEO_AX_2"	第 2 根几何轴方向上的 2D 视图
"VIEW_TOP_GEO_AX_3"	第 3 根几何轴方向上的 2D 视图

C.4 转换为 hmi 文件

说明

在 HMI 启动时将含有配套 XML 文件的 X3D 文件转换为 HMI 文件。

必须为每个 X3D 文件创建一个相应（同名）的 XML 文件。因此，必须将 X3D 文件和 XML 文件保存到目录 `Suchpfad des HMI\ico\x3d\turning` 或 `milling` 中。步骤与 .ts 文件类似。

C.5 在 Create MyHMI /3GL 中显示

C.5.1 X3D 浏览器

如需在自有 OA 应用中显示目标辅助画面，则须将 X3D 浏览器小部件集成到自有 OA 应用中。

X3D 浏览器小部件提供了可在 HMI 内部显示 X3D 内容的接口。

类 SlX3dViewerWidget 用于显示图形场景。 该类的定义是在全局 GUI Include 目录 `\hmi_prog\gui\include` 下的标题文件 slx3dviewerwidget.h 中进行的。

C.5.2 类 SlX3dViewerWidget

该类提供了一个可灵活使用的小部件，该部件可以独立显示指定模型文件中的内容，必要时还可运行动画。

类接口由构造函数、析构函数及两种图形输出控制方法组成。

提供了功能全面的接口用于直接推导 Qt 类 QWidget（例如：show(), hide()和 resize(...)，但此处不进行进一步说明，详细信息请参考 Qt 文档）。

C.5.3 公共方法

SlX3dViewerWidget (QWidget* pParent = 0)

X3D 浏览器小部件的构造函数。

参数	含义
pParent	将参数传送给 Qwidget 的构造函数。

~SlX3dViewerWidget ( )

X3D 浏览器小部件的析构函数。

参数	含义
-	-

C.5.4 公共槽

void viewSceneSlot (const QString& rsFileName, const QString& rsScene, const QString& rsAnimationScene, int nChannel, int nPlane, SlStepTechnology nTechnology)

借助 viewSceneSlot 方法，X3D 浏览器可指定载入和显示 rsFileName 文件中的静态场景 rsScene 和动画场景 rsAnimationSzene。

正确的显示方式是先显示静态场景，再显示动画场景（按固定时间循环）。

如果未指定静态场景，则立即显示动画；相应的也可能未指定动画场景。

使用通道号、平面和工艺使场景转动到正确的位置上（取决于设置的机床坐标系）。

参数	含义
rsFileName	含待显示场景的文件的名称。
rsScene	静态场景名称。
rsAnimationScene	动画场景名称。
nChannel	通道号
nPlane	平面
nTechnology	工艺 针对计数类型 SlStepTechnology 定义了以下常量： • SL_STEP_NO_TECHNOLOGY • SL_STEP_MILLING • SL_STEP_TURNING • SL_STEP_SURFACE_GRINDING • SL_STEP_CIRCULAR_GRINDING

void viewSceneSlot (const QString& rsFileName, const QString& rsScene, const QString& rsAnimationScene)

一种简易化 viewSceneSlot 方法，X3D 浏览器可借助其指定载入和显示 rsFileName 文件中的静态场景 rsScene 和动画场景 rsAnimationSzene。

参数	含义
rsFileName	含待显示场景的文件的名称。
rsScene	静态场景名称。
rsAnimationScene	动画场景名称。

C.5.5 库

如需在自有项目中使用 X3D 浏览器，则须为库相关性列表扩展项 'slx3dviewer.lib'。

C.5.6 执行示例

执行示例请参考 Create MyHMI/3GL 程序包中的 **\examples\GUIFrameWork\SlExGuiX3D**。

C.5.7 机床数据

显示 MD 9104 $MM_ANIMATION_TIME_DELAY

至使用辅助画面中的动画的延时，单位：秒。 该设置对只进行了动画处理的辅助画面无效。

该设置对 SINUMERIK Operate 的所有动画都有效。

C.5.8 使用说明

- 如果 X3D 浏览器小部件消失，动画则会中断。 此时无需选中无动画场景选择。
- 考虑到性能和存储容量，应避免经常或多次实例化 X3D 浏览器小部件。 此类应用场景下推荐使用（执行）X3D 浏览器小部件。
- X3D 浏览器也可在信号槽方案中使用。
- 如果出错（例如：文件未找到或场景名称不可识别），X3D 浏览器小部件则会显示一个信息区。 一旦操控另一个辅助画面文件，该信息区则会再次自动消失。

C.6 在 Run MyScreens 中显示

本章节面向有经验的 “Run MyScreens” 开发人员。 可在对应的文档中查阅相关背景知识。

除了使用 .bmp 或 .png 格式的图片外，也可通过 X3D 浏览器显示动画辅助画面。

通常使用用于辅助画面的 Run MyScreens 接口进行集成。

如需输出 .bmp 或 .png 格式的图片，则应在定义对话框时在属性中输入数据 XG0 或清空参数。 如需将 X3D 辅助画面集成到对话框中，则须在属性中输入数据 **XG1**。

```
//M(MASK_F_DR_O1_MAIN/$85407////52,80/XG1)
```

操控单个辅助画面需要以下参数，其含义参见章节 5.3 中的公开槽：

1. 文件名
2. 静态场景（可选）
3. 动画场景（可选）
4. 工艺（可选）
5. 平面（可选）

参数按此顺序以字符串的形式排列汇总，用逗号隔开。

```
Hlp = "Dateiname,StaticScene,AnimationScene,Technologie,Ebene"
```

示例

- 默认辅助画面可通过指定变量 Hlp 设置。
  下面的示例是从文件 “MyDlgHelp.hmi” 中输出动画 “MyAnimation”。 未指定静态场景，即不会输出静态场景。 也没有指定工艺和平面数据，即使用默认值。
  ```
  Hlp = "MyDlgHelp.hmi,,MyAnimation"
  ```
- 对于输入窗口的单个参数，可将对应变量上的属性 hlp 设为一个特定的辅助画面。
  下面的示例是从文件 “MyDlgHelp.hmi” 中输出平面 G17 中的静态场景 “MyParam” 和动画 “MyAnimation”。 未指定工艺数据，即使用默认值。
  ```
  VarMyParam.hlp = "MyDlgHelp.hmi,MyParam,MyAnimation,,G17"
  ```

文档一览

D

有关 SINUMERIK 840D sl 功能的详细文档，自版本 4.8 SP4 起，请参见 840D sl 文档一览 (https://support.industry.siemens.com/cs/ww/en/view/109766213)。



您可以直接打开文档或者下载 PDF 和 HTML5 格式。

文档分为以下几个类别：

- 用户：操作
- 用户：编程
- 制造商/服务：功能
- 制造商/服务：硬件
- 制造商/服务：配置/调试
- 制造商/服务：Safety Integrated
- 制造商/服务：SINUMERIK Integrate / MindApp
- 介绍和培训
- 制造商/服务：SINAMICS

索引

D

DLL 文件, 181

E

ELLIPSE（定义椭圆形）, 220
ELLIPSE（定义圆形）, 220

H

H_SEPARATOR（定义水平分割线）, 221

L

LINE（定义线）, 219

N

NC 变量
　读取, 197
　写入, 197

P

PI 服务, 159
PLC 变量
　读取, 197
　写入, 197

R

RECT（定义矩形）, 220

S

SIEsButton
　属性概述, 286
SIEsGraphCustomWidget
　addArc, 276
　addCircle, 275
　addContour, 268
　addEllipse, 275
　addLine, 274
　addPoint, 273
　addRect, 274
　addRoundedRect, 275
　addText, 276
　AxisNameX, 252
　AxisNameY, 252
　AxisY2Factor, 253
　AxisY2Offset, 253
　AxisY2Visible, 252
　BackColor, 259
　ChartBackColor, 260
　clearContour, 270
　CursorColor, 261
　CursorStyle, 263
　CursorX, 262
　CursorY, 262
　CursorY2, 263
　findX, 271
　fitViewToContour, 270
　fitViewToContours, 270
　ForeColor, 260
　ForeColorY2, 260
　GridColor, 261
　GridColorY2, 261
　hideAllContours, 269
　hideContour, 269
　KeepAspectRatio, 258
　moveCursorOnContourBack, 281
　moveCursorOnContourBegin, 280
　moveCursorOnContourEnd, 280
　moveCursorOnContourNext, 281
　removeContour, 269
　repaint, update, 273
　ScaleTextEmbedded, 255
　ScaleTextOrientationYAxis, 256
　SelectedContour, 259
　serialize, 282, 308
　setCursorOnContour, 279
　setCursorPosition, 278
　setCursorPositionY2Cursor, 278
　setFillColor, 278
　setIntegralFillMode, 272
　setMargins, 307
　setMaxContourObjects, 267
　setPenColor, 277
　setPenStyle, 277
　setPenWidth, 277
　setPolylineMode, 271
　setView, 266

showAllContours, 269
showContour, 268
ShowCursor, 262
ViewChanged, 282
ViewMoveZoomMode, 264
功能概述, 264
属性概述, 251
信号概述, 282
SIEsTouchButton, (和语言相关的文本)
BackColor, 299
BackColorChecked, 299
BackColorDisabled, 301
BackColorPressed, 300
BackgroundPicture, 303
BackgroundPictureAlignment, 304
BackgroundPictureAlignmentString, 305
BackgroundPictureKeepAspectRatio, 306
ButtonStyle, 287
Checkable, 289
checked, 310
Checked, 290
clicked, 309
clickedDisabled, 310
Enabled, 289
Flat, 288
Picture, 291
PictureAlignment, 293
PictureAlignmentString, 293
PictureKeepAspectRatio, 295
PicturePressed, 292
ScaleBackgroundPicture, 306
ScalePicture, 294
ShowFocusRect, 291
Text, 295
TextAlignedToPicture, 298
TextAlignment, 296
TextAlignmentString, 298
TextColor, 301
TextColorChecked, 302
TextColorDisabled, 303
TextColorPressed, 302
TextPressed, 296
功能概述, 307
信号概述, 309
SlEsGraphCustomWidget
读取和写入属性, 250
SlEsTouchButton
读取和写入属性, 285
slhlp.xml, 88
SlX3dViewerWidget, 381

V

V_SEPARATOR（定义垂直分割线）, 221
Vivaty Studio, 369

帮

帮助画面, 111
帮助文件, 113

保

保护等级, 354

报

报警
语种缩写, 356

背

背景色, 112

比

比较运算符, 142

变

变量
CURPOS, 126
CURVER, 127
ENTRY, 128
ERR, 129
FILE_ERR, 130
FOC, 132
S_ALEVEL, 133
S_CHAN, 134
S_CONTROL, 135
S_LANG, 136
S_NCCODEREADONLY, 137
S_RESX, 138
S_RESY, 138
参数, 107
传输, 175
更改属性, 113
计算, 95
检查, 164

变量类型, 107
INTEGER, 115
VARIANT, 116
变量值, 93
变量状态, 93

表

表格
编程, 231
定义, 230
定义列, 232

不

不变色进度条, 104

操

操作树, 39

常

常量, 141

触

触控操作, 284

单

单位文本, 108

导

导入
图形（模型）, 371

登

登入软键, 39, 41

定

定义软键栏, 72

短

短文本, 108

对

对话框
定义, 55
多列, 67
属性, 58
说明块, 56
对话框单元, 65
对话框切换模式, 190

多

多点触控操作, 284

方

方法
ACCESSLEVEL, 145
CHANGE, 146
CHANNEL, 147
CONTROL, 148
LANGUAGE, 149
LOAD, 150
LOAD GRID, 192
OUTPUT, 151
PRESS, 152
PRESS(ENTER), 153
PRESS(TOGGLE), 154
RESOLUTION, 155
RESUME, 155
SUSPEND, 156
UNLOAD, 150
概述, 145

访

访问等级, 74

辅

辅助变量, 95

工

工步链支持, 203

公

公共槽, 382

功

功能
- CALL （子程序调用）, 162
- CLEAR_BACKGROUND), 165
- CP (复制程序), 165
- CVAR (检查变量), 164
- DEBUG, 174
- DLGL (对话框行), 173
- DO-LOOP), 215
- DP (删除程序), 166
- EP (存在程序), 167
- EVAL (评估), 179
- EXIT, 175
- EXITLS （退出装载软键）, 180
- FCT, 181
- FORMAT(字符串), 214
- GC （生成代码）, 183
- HMI_LOGIN, 186
- HMI_LOGOFF, 186
- HMI_SETPASSWD, 186
- INSTR (字符串), 209
- LA （装载数组）, 187
- LB （装载块）, 188
- LEFT(字符串), 210
- LEN (字符串), 208
- LISTADDITEM), 177
- LISTCLEAR), 177
- LISTCOUNT), 177
- LISTDELETEITEM, 177
- LISTINSERTITEM, 177
- LM （装载屏幕）, 189
- LS （装载软键）, 191
- MIDS(字符串), 211
- MP（移动程序）, 168
- MRDOP), 160
- MRNP（多次读取 NC PLC）, 194
- PI_START, 196
- RDFILE), 170
- RDLINEFILE), 170
- RDOP), 160
- REPLACE(字符串), 211
- RESIZE_VAR_IO, 198
- RESIZE_VAR_TXT, 198
- RETURN （返回）, 201
- RIGHT(字符串), 210
- RNP（读取 NC PLC 变量）, 197
- SP（选择程序）, 169
- START_TIMER), 217
- STOP_TIMER, 217
- STRCMP(字符串), 212
- STRINSERT(字符串), 212
- STRREMOVE(字符串), 213
- SWITCH), 193
- TRIMLEFT(字符串), 213
- TRIMRIGHT(字符串), 213
- UNTIL, 215
- WDOP, 160
- WHILE, 215
- WNP (写入 NC PLC 变量), 197
- WRFILE, 170
- WRLINEFILE), 170
- 读写驱动参数, 160
- 反编译 NC 代码, 202
- 概述, 159
- 忽略注释的反编译, 203
- 文件访问, 170
- 向后查找(SB), 207
- 向前查找(SF), 207
- 循环执行脚本, 217

和

和语言相关的文本, (SIEsTouchButton)

机

机床数据, 383

极

极限值, 107

集

集成了单位选择框的输入/输出栏, 113

寄

寄存器
- 交换数据, 199

值, 200
状态, 200

聚

聚焦控制, 234

口

口令对话框, 69
口令输入模式（星号）, 106

两

两次变色的进度条, 103

列

列表栏, 101

配

配置句法“；”变量, 51
配置句法“；”表格列, 52
配置句法“；”扩展句法, 51
配置句法“；”屏幕, 51
配置句法“；”软键, 52
配置文件, 37

前

前景色, 112

软

软键
分配属性, 72
属性, 74

三

三角函数功能, 140

设

设计 PLC 软键, 323

生

生成 NC 代码, 183

输

输入模式, 110

属

属性, 109

数

数学函数, 141
数字格式, 117
数组
比较模式, 225
存取模式, 225
单元, 224
定义, 223
行索引, 225
栏索引, 225
状态, 228

刷

刷新速度, 109

提

提示框, 108, 110

条

条件, 142

图

图片取代短文本, 102
图形文本, 108

位

位置
短文本, 111, 122
输入/输出栏, 112, 122

文

文本, 108
文件
    复制, 165
    删除, 166
    移动, 168
文件格式
    fxw, 367
    hmi, 367
    x3d, 367
    xml, 367

系

系统变量, 96, 111
系统颜色, 355

显

显示模式, 109
显示选项, 109

写

写入模式, 112

信

信号色“；”进度条, 112
信号值“；”进度条, 107

颜

颜色, 112

用

用户变量, 111

语

语种缩写, 356

预

预设值, 107

运

运算符
    数学, 139
    位, 143

栅

栅格 → 表格, 230

长

长文本, 108

主

主对话框, 189

转

转换符号, 110
转换栏, 101, 107, 119

子

子程序, 159
    变量, 163
    块标记, 163
    调用, 162
    中断, 201
子对话框, 189

自

自定义小部件
    定义, 235
    读取和写入属性, 240
    接口, 236
    库, 236
    手动数据交换, 239
    响应自定义小部件信号, 245
    执行方法, 241
    自动数据交换, 238

字

字符串链, 124

SIEMENS

系统设置 1

配置系统 2

诊断与维修 3

附录 A

SINUMERIK

SINUMERIK 840D sl NCU 操作系统

开机调试手册

适用于：

840D sl/ 840DE sl 的 CNC 软件　V4.94
Linux 基础软件 V5.10

10/2020
A5E48312736F AB

法律资讯

警告提示系统

为了您的人身安全以及避免财产损失，必须注意本手册中的提示。人身安全的提示用一个警告三角表示，仅与财产损失有关的提示不带警告三角。警告提示根据危险等级由高到低如下表示。

⚠ 危险

表示如果不采取相应的小心措施，**将会**导致死亡或者严重的人身伤害。

⚠ 警告

表示如果不采取相应的小心措施，**可能**导致死亡或者严重的人身伤害。

⚠ 小心

表示如果不采取相应的小心措施，可能导致轻微的人身伤害。

注意

表示如果不采取相应的小心措施，可能导致财产损失。

当出现多个危险等级的情况下，每次总是使用最高等级的警告提示。如果在某个警告提示中带有警告可能导致人身伤害的警告三角，则可能在该警告提示中另外还附带有可能导致财产损失的警告。

合格的专业人员

本文件所属的产品/系统只允许由符合各项工作要求的**合格人员**进行操作。其操作必须遵照各自附带的文件说明，特别是其中的安全及警告提示。 由于具备相关培训及经验，合格人员可以察觉本产品/系统的风险，并避免可能的危险。

按规定使用 Siemens 产品

请注意下列说明：

⚠ 警告

Siemens 产品只允许用于目录和相关技术文件中规定的使用情况。如果要使用其他公司的产品和组件，必须得到 Siemens 推荐和允许。正确的运输、储存、组装、装配、安装、调试、操作和维护是产品安全、正常运行的前提。必须保证允许的环境条件。必须注意相关文件中的提示。

商标

所有带有标记符号 ® 的都是 Siemens AG 的注册商标。本印刷品中的其他符号可能是一些其他商标。若第三方出于自身目的使用这些商标，将侵害其所有者的权利。

责任免除

我们已对印刷品中所述内容与硬件和软件的一致性作过检查。然而不排除存在偏差的可能性，因此我们不保证印刷品中所述内容与硬件和软件完全一致。印刷品中的数据都按规定经过检测，必要的修正值包含在下一版本中。

Siemens AG
Digital Industries
Postfach 48 48
90026 NÜRNBERG
德国

A5E48312736F AB
Ⓟ 10/2020 本公司保留更改的权利

Copyright © Siemens AG 2006 - 2020.
保留所有权利

目录

1 系统设置 5

1.1 系统供货状态 5

1.2 开关位置的含义 8

1.3 系统引导启动 9

1.4 系统引导启动时的显示 10

2 配置系统 13

2.1 NCU 的 Ethernet 接口 13

2.2 确定 NCU IP 地址的方法 14

2.3 配置文件"basesys.ini" 15

2.3.1 段落 [ExternalInterface] 16

2.3.2 段落 [InternalInterface] 19

2.3.3 段落 [IBNInterface] 22

2.3.4 段落 [SNMP] 23

2.3.5 段落 [DCP] 25

2.3.6 段落 [LLDP] 25

2.3.7 段落 [LinuxBase] 27

2.3.8 示例： 配置文件"basesys.ini" 32

2.4 服务命令 42

2.4.1 使用服务命令 42

2.4.2 动作的句法 43

2.4.3 类单元 44

2.5 动作描述 47

2.5.1 Help 47

2.5.2 Check-cf 48

2.5.3 Clear 48

2.5.4 Closeport 49

2.5.5 Connect 50

2.5.6 Disable 52

2.5.7 Disconnect 53

2.5.8 Distribute 54

2.5.9 Enable 55

2.5.10 ntpdiag（诊断） 56

2.5.11 Openport 57

2.5.12 Portstatus 58

2.5.13 Restart 58

2.5.14 Restore 59
2.5.15 Save 60
2.5.16 save-nvram（备份数据） 62
2.5.17 set-svc-route（网关） 62
2.5.18 Show 63
2.5.19 Start, Stop 67
2.5.20 usb（连接设备） 67
2.5.21 vncpwd（密码） 69

3 诊断与维修 71

3.1 生成服务系统 71
3.1.1 这样创建 NCU 的服务系统 71
3.1.2 这样操作维修系统 73
3.1.3 诊断功能 74
3.1.4 这样连接网络驱动器 75

3.2 备份授权 77
3.2.1 这样备份授权密钥 78

3.3 通过维修菜单备份数据 80
3.3.1 这样备份用户数据 80
3.3.2 这样恢复用户数据 83
3.3.3 这样创建 CF 卡的完整备份 84
3.3.4 这样安装完整的系统备份 86

3.4 通过 Service Shell 备份数据 88
3.4.1 这样将数据备份至服务系统 88
3.4.2 这样从服务系统恢复数据 89
3.4.3 这样将数据备份至网络驱动器 89
3.4.4 这样从网络驱动器恢复数据 90

3.5 执行软件更新 92
3.5.1 这样执行固件更新 92
3.5.2 这样执行软件更新 94

3.6 系统诊断 96
3.6.1 执行系统诊断 96
3.6.2 这样修改系统设置 98
3.6.3 这样检查 CF 卡 100
3.6.4 调用 VNC 浏览器 102
3.6.5 WinSCP 和 PuTTY 104

A 附录 105

A.1 缩写 105

术语表 107

索引 109

系统设置 1

1.1 系统供货状态

适用范围：

该文献适用于所有 CF 卡上为 Linux 操作系统的系统，例如：类型为 NCU 7x0.3B PN 的所有 NCU。

CF 卡的目录

在 CF 卡上有 Linux 分区和 FAT 分区。其中 Linux-EXT3 分区占主要部分，它包含了系统软件和用户数据。只剩下 2 MB 大小的 FAT 分区用于内部管理。

系统运行时，CF 卡上的 EXT3 分区有如下目录（节选，没有全部列出）：

目录	使用
/siemens	为保存西门子系统软件预留
/addon	为保存西门子插件预留
/oem	保存机床制造商的插件和配置
/user	• 保存用户数据 • 保存 HMI 应用程序的配置 • 保存调试时 HMI 应用程序产生的数据
/system	Linux 操作系统
/user/system/etc	basesys.ini 文件（可以更改）
/user/common/tcu	TCU 的配置文件
/var/log/messages	系统日志文件（和 Windows 操作系统下的 event.log 类似）

说明

如果文件同名，目录 /user 下文件的优先级比目录 /oem → /addon → /siemens 下的文件要高。

存储器划分

CF 卡上存储器的划分是纯逻辑式的，也就是说：原则上用户可以任意占用存储器，直到达到最大物理限制。如果可用存储容量小于 4 MB，LINUX 会发出报警。

目录	1 GB 存储卡	8 GB 存储卡
/oem /addon	100 MB	1 GB
/user	256 MB	2 GB

预设用户

系统设定了以下几种用户：

用户	存取等级	目标客户
Operator	钥匙开关位置 0	用户，操作人员
Operator1	钥匙开关位置 1	用户，操作人员
Operator2	钥匙开关位置 2	用户，操作人员
Operator3	钥匙开关位置 3	用户，操作人员
user	密码：CUSTOMER	用户，操作人员
service	口令： EVENING	服务人员
manufact	密码：SUNRISE	机床制造商

注意
更改密码 您只能在 NCU 基础系统中自行修改 Linux 用户的密码 - 而不是在已引导启动的服务系统上。 如要提高系统安全性，则应像 NC 访问级的口令一样，将 Linux 系统中的密码更改为用户“manufact”、“service”和“user”。

如要进入 NCU 基础系统，例如可通过程序 WinSCP/Putty (页 104)、服务控制台或者在 TCU 上。

在 TCU 上使用菜单回调键及按键<MENU SELECT>调用服务控制台：



密码的设置规则

在设置新密码时应注意以下规则：

- 允许（和要求）的字符为小写字母、大写字母、数字和特殊字符。
- 密码至少应包含 6 个字符。每缺少一种字符类别（小写字母、大写字母、数字和特殊字符），最短密码长度延长两个字符。
  示例：如果密码只含大写字母，则长度应为 12 个字符。
- 密码不得含有过多的重复字符。
- 密码不得与用户名、计算机名和旧密码相同（例如倒序书写、大/小写调换）。

如若违反这些规则，会提示“too weak”并且旧密码不会发生变化。

更改密码

您只能更改所登录用户的密码，就是说如您要更改所有预设用户的密码，则需要分别登录这些用户。

```
$ passwd
Changing password for service
Old password:
New password:
Bad password: too weak
passwd: password for service is unchanged
$ passwd
Changing password for service
Old password:
New password:
Retype password:
```

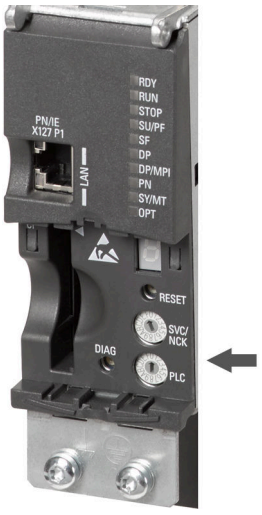
步骤：

1. 在 NCU 基础系统 (Linux) 下使用用户名和密码登录。
2. 输入命令“passwd”。
3. 根据指示，输入截止目前所使用的旧密码。
4. 输入新密码，并重复输入一次，加以确认。
5. 如果所有输入都正确，则新密码生效。任何差异都会导致退出对话框，并且旧密码依旧生效。

1.2 开关位置的含义

NCU 的诊断显示

NCU 面板的下方有两个旋转开关：



旋转开关 SVC/NCK 上的位置含义：

开关位置	NC 的运行方式
0	NC 正常启动
1	NC 采用预设值启动（= 清零）
2	NC 采用预设值启动；启动后 PLC 处于 RUN 状态。
7	调试模式（NC 不启动。）
8	NCU 的 IP 地址以 7 段方式输出。
所有其他位置	不相关

PLC 旋转开关的设置和使用一个 SIMATIC S7-CPU 时的含义相同：

开关位置	PLC 的运行方式
0	RUN
1	RUN （保护模式）
2	STOP
3	清零(MRES)
所有其他位置	不相关

1.3 系统引导启动

操作步骤

为了确保 NCU 顺利启动，必须插入 CF 卡。

NCU 启动时，通过下列显示装置输出当前运行状态的可视信息：

- 在存取 CF 卡时，RDY-LED 橙色灯慢速闪烁。
- 7 段显示器在启动时会输出不同的代码，例如会在启动 BIOS 或存取 CF 卡等情况下发出信号。

当启动成功结束时，可以看见下列显示：

- PLC-LED 绿灯亮起。
- 7 段显示器用闪烁点显示"6."。
- RDY-LED 绿灯亮起，其他所有的 LED 都关闭。

执行复位

复位键位于 NCU 盖板的后面。

复位会使整个系统复位并要求重启系统，和“上电复位”相比无需关闭 24 V 电源。

维护情况下启动

出于维修或者诊断目的，可以在维修系统（Emergency Boot System：紧急启动系统）上执行 NCU 的引导启动。

另见

这样创建 NCU 的服务系统 (页 71)

1.4 系统引导启动时的显示

启动时 LED 的特性

在 NCU 前面的 LED 中，只有 RDY-LED 及其状态对启动 NCU 有用。

LED RDY	其他 LED	状态/阶段	7 段显示
橙色	橙色	BIOS 启动	位置代码 参见表格：装载 BIOS
橙灯缓慢闪烁（0.5 Hz）	由 PLC 或选件模块设置	内核启动 阶段 2：驱动器初始化	1
橙灯/绿灯缓慢闪烁（0.5 Hz）	由 PLC 或选件模块设置	基础系统启动 阶段 3：基础系统初始化	参见表格：装载基础系统
OFF	由 PLC 或选件模块设置	NRK/NCK 输出	参见表格：NRK/NCK 输出
绿灯/红灯闪烁（0.5 Hz）	由 PLC 或选件模块设置	基础系统关闭	OFF
红色	由 PLC 或选件模块设置	内核关闭	OFF
红色（5 Hz）	由 PLC 或选件模块设置	SINAMICS 故障	---

SINAMICS 故障和报警，另见：SINAMICS S120/S150 参数手册 (LH1)

装载 BIOS

7 段显示	含义
00	在装载 CF 卡的主引导记录区（MBR：Master Boot Record）前
01	在启动主引导记录（MBR：Master Boot Record）代码前
02	主引导记录（MBR：Master Boot Record）代码已启动
03	在读取配置文件前，启动 Boot Loader 的第二级
04	在装载内核前
05	在启动内核前

基础系统启动

见图：开关位置的含义 (页 8)

7 段显示	含义
1 (begin)	基本初始化
2 (prepfs)	文件系统的清理和准备
3 (hostname, loopback)	设置名称和配置环回接口(Loopback-Interface)。
4 (syslog)	启动系统日志 启动 CFS 等级 1，准备网络接口。
5 (network)	网络接口和防火墙初始化
6 (NCU:rtai, TCU:tcuconfig)	启动时间服务器（ntpd）
7 (lsh)	启动 SSH 服务器（若无 Host 键，可创建）。
8 (NCU:ftpd, TCU:mtouch)	启动 FTP 服务器
9 (NCU:tcuservices, TCU:usbexport)	启动 TCU 通讯（TFTP，VNC）
U r	Update required: NCU 的 CF 卡上存在的软件版本既不能在 3B 类型的模块上运行，也不能通过 Update 进行更新。 **解决方法：** • 检查该 CF 卡是否在其他 NCU 上启动。 • 借助维修系统检查该 CF 卡是否损坏：sc check-cf (页 48) 以下情况中可以额外输出代码"U r"： • 未插入 CF 卡。 • 无法读取 CF 卡。 • CF 的分区表有错误。

NRK/NCK 输出

见图：开关位置的含义 (页 8)

7 段显示	含义
1	不相关。
2	不相关。
3	Debugger 已启动。
4	NRK 操作系统已成功激活。
5	NRK 操作系统已启动，初始化任务进行中（→ 应用程序启动）。
6 6.	初始化已成功进行；控制系统处于循环运行中。 带闪烁点：控制系统处于循环运行中，且周期有效。
F	内部故障：仅可通过附加诊断读出。
1 或 2	CF 卡和 SRAM 数据不匹配。需要通过含开关位置 1 或 2 的引导启动进行清零来排除故障。
Lxx=yyy	NCK 启动时的内部故障，其中“xx”为 mcsystem.ini 中的行编号，“yyy”为代码位的唯一故障代码。

系统故障

见图：开关位置的含义 (页 8)

LED RDY	其他 LED	含义	7 段显示	含义
红灯快速闪烁 (2 Hz)	由 PLC 或选件模块设置	**永久故障：** 系统停止运行。	C	“crash”（崩溃）：操作系统崩溃（和 Windows 中的蓝屏相似）；详细信息请参见系统日志文件。
			P	“partition”（分区）：重新分区时出错
红灯/橙灯快速闪烁（2 Hz）	由 PLC 或选件模块设置	**暂时故障：** 系统继续运行，但可能某些功能会被限制。	E	“error”（错误）：在 CF 卡上发生写入或读取出错（对于写入错误还会显示额外的点）。可能是 CF 卡损坏并且需要更换。
			F	“full”（过满）：CF 卡存储过满，因此可能不能成功启动所有通讯。 此故障仅在启动时显示 1 分钟；之后系统会继续启动，尽管可能会存在问题。

配置系统 2

2.1 NCU 的 Ethernet 接口

前提条件

为运行 NCU：

- 在工程网络中最多允许一个 NCU 作为 DHCP 服务器运行。
- 建议使用一个外部鼠标来操作系统网络中心。

接口的配置

通过以太网接口可以建立下列连接：

接口	标记	内部名称	接口设置
Ethernet IE1/OP	X120	(Eth 2)	连到工程网络上的接口，使用默认 IP 地址 192.168.214.1、子网掩码 255.255.255.0 和有效的 SINUMERIK DHCP 服务器。
以太网 IE2/NET	X130	(Eth 1)	作为标准 DHCP 客户端接到公司网络上的接口
Ethernet	X127	(Ibn 0)	服务接口，使用固定 IP 地址 192.168.215.1 和固定子网掩码 255.255.255.224 以及有效的 DHCP 服务器

网络接口

网络接口是实现网络通讯的接口。 在 NCU 上，网络接口是以太网接口。

VNC（virtual network computing：虚拟网络计算）

虚拟网络处理是一种软件，它可以在本地计算机（VNC 浏览器运行中）上显示远程计算机（VNC 服务器运行中）的屏幕内容并且可以通过本地计算机的键盘和鼠标来操作远程计算机。

2.2 确定 NCU IP 地址的方法

工程网络(X120)中的 NCU

NCU X120 的缺省地址已经适用，不需要进行任何设置。

公司网络(X130)中的 NCU

NCU X130 上已经设置了 DHCP 分配的地址。如公司网络中已具有一台 DHCP 服务器，此时不需要进行其他的设置。

示例：测定 IP 地址

有三种方法可以确定 NCU 在公司网络中获取的 IP 地址：

- 如果 NCU 从开关位置 8 启动，X130 的 IP 地址会显示在 7 段指示器上。
- 在 NCU 成功启动后，您可以在 TCU 上打开一个 Service Shell 程序并执行下列命令 (页 63)，用来获取所需信息：
  ```
  SC SHOW IP
  ```

如果在公司网络中未安装或者没有使用 DHCP 服务器，可以为 NCU X130 设定一个固定地址（与公司网络中已使用的地址一致）。

以下指令可以查看 X130 的 IP 地址：

```
SC SHOW IP -X130
```

说明

也可直接通过 SINUMERIK Operate 操作界面上的操作区“调试”→“网络”设置 IP 地址。

2.3 配置文件"basesys.ini"

存放路径

在基于 Linux 的系统中，文件模板以以下路径保存在系统 CF 卡上：
`siemens/system/etc/template-basesys.ini`

将该模板以 basesys.ini 名复制到以下其中一个目录中：
`oem/system/etc/basesys.ini`
`user/system/etc/basesys.ini`

使用

在"basesys.ini"文件中可以设置系统在网络中的属性。

编辑该文件时应注意以下几点：

- 注释通过';'或'#'在行首引入，并占据一整行。空行也视为一条注释。
- 由一个 "[NAME]" 行引入的段落会被基础系统自身忽略，但是 HMI 会部分地使用该段落。
- 变量定义的格式为 "名称=数值"。' ='前后可以有空格。此外，数值也可以选择括在双引号内。

说明

适用于 Linux 的编辑器

大多数 Linux 系统文件中仅允许用 LF 来换行，而不允许用 Windows 操作系统中的 CRLF。在选择编辑器时要注意这一点。"调试" 下的 HMI 应用程序编辑器是合适的编辑器。

在 Linux 操作系统中，有 UNIX 自带的编辑器 vi 可供使用。

使用 Linux 操作系统时请区分大小写。

文档

为了在文件 basesys.ini 中进行设置，可使用 SINUMERIK Operate "调试" 操作区中的对话框来显示或修改各个变量的值："SINUMERIK Operate"调试手册 (IM9)

2.3.1 段落 [ExternalInterface]

说明

在该段中会定义外部以太网接口的参数和设置。使用 NCU7x0 时此处是 X130 (Eth 1)。如果外部接口处于 DHCP 模式下，也就是说 ExternalIP （外部 IP）的值为空或者未定义，那么只要该服务器为此提供一个值，DHCP 服务器便会接收所有此处列出的参数（除了 'DHCPClientID'）。

DisableCompanyNet

该变量置 1 后，公司网络接口 X130 便不再初始化，不再可以使用。

值： 空、0 或者 1
缺省值： 空

ExternalIP

如果 ExternalIP 不为空，则使用其中给定的、公司网络接口上的固定 IP 地址。此外，必须设置 ExternalNetMask（外部网络掩码），并且根据需要也要设置 Gateway （网关）、Nameservers（名称服务器）、Hostname （主机名称）和 Domain（域名）。

如果 ExternalIP 未设置或者为空，则会启动接口上的一个 DHCP 用户。

值： IP 地址
缺省值： 空

ExternalNetMask

ExternalNetMask 必须连同 ExternalIP 一起设置，以便定义网络的大小。

值： 网络掩码
缺省值： 空

Gateway

如果 Gateway 的值不为空，则其中给出的 Host （主机）作为 Default-Gateway（缺省网关）来使用，也就是说，所有不能直接递送的 IP 包都会发送到那里用来转寄。

如果没有规定网关，则只能到达直接连接的网络。

值： IP 地址
缺省值： 空

Nameservers

如果在此处给定了 DNS 名称服务器，则该服务器用于分辨符号型的主机名称，即：在大多数需要 IP 地址的位置上也可以使用计算机名称。

名称服务器设置也会通过 NCU 的 DHCP 服务器传递到 DHCP 用户 (TCU, PG)，从而使得这些用户也能够用符号名称来工作。

值： IP 地址列表（用空格隔开）
缺省值： 空

Timeservers

在此处可以输入 NTP 服务器(Network Time Protocol)的列表，NCU 上的 NTPD 通过此服务器进行时间同步。NTPD 的状态输出在系统日志文件 /var/log/messages 中。

也可通过"ntpq -p localhost"查看系统日志文件是否有一个用 '*' 标记的时间服务器。通过该服务器同步时间。

值： IP 地址列表（用空格隔开）
缺省值： 空

说明

设置时区和时间服务器

也可直接通过 SINUMERIK Operate 操作界面上的操作区“调试” → 软键“日期时间”进行设置。

Hostname

此处可以确定一个本地主机的名称。该手动给出的名称相对于其他所有名称有优先权。

该主机名称依照以下顺序来决定：

- "basesys.ini"文件中的主机名称，如果设置了该名称。
- 由 DHCP 服务器获得的名称（如果提供 DHCP 客户模式和名称）：反向 DNS 查找的结果，即哪个名称属于获得的 IP （如果定义了名称服务器）。
- 缺省名称 ("NONAME_...")

由于主机名称也用作 DNS 名称，因此它必须满足相应的 RFC 要求：

- ASCII 字母(a-z, A-Z)、数字(0-9)和"-"
- 最多 63 个字符

值： 名称
缺省值： 空

Domain

使用该变量可以配置 DNS 域名。其作用在于，域名无需验证即可被识别出，例如：如果域名为"test.local"，则可以选择给定名称为"rechner1.test.local"或"rechner1"。

值： 名称
缺省值： 空

DHCPClientID

使用该变量可以控制 DHCP 用户将哪个用户 ID 显示在它的服务器上。服务器可以使用该 ID，以便能够给用户分配确定的参数，例如一个静态 IP 地址。

通常以太网接口的 MAC 地址具有该用途，这也是系统的缺省设置。此外还可以使用主机名称 ("@NAME")，当然该名称稍后必须在 basesys.ini 文件中定义，以便在 DHCP 要求之前能够识别。除此之外也可以使用任意一个字符串作为 ClientID。

值： @MAC, @CFID, @NAME 或者任意一个字符串
缺省值： @MAC

2.3.2 段落 [InternalInterface]

说明

在段落[InternalInterface]中可以定义用于工程网络中 DHCP 服务器同步的数据。

DisableSysNet

该变量置 1 后，工程网络接口 X120 便不再初始化，不再可以使用。

值： 空、0 或者 1
缺省值： 空

InternalIP

使用该变量可以设置工程网络中的固定 IP 地址。InternalIP 必须始终和 InternalNetMask 一起使用。

值： IP 地址
缺省值： 192.168.214.1

InternalNetMask

该变量指定了 InternalIP 的子网掩码，两者必须始终一起使用。

值： 子网掩码
缺省值： 255.255.255.0

InternalIP_Alias

使用该变量可以为工程网络接口(X120)定义一个附加的假 IP 地址，在特定应用情况下会需要使用该地址。

值： IP 地址
缺省值： 空

InternalIP_Alias 必须始终和 InternalNetMask_Alias 一起使用。

InternalNetMask_Alias

该变量指定了 InternalIP_Alias 的子网掩码，两者必须始终一起使用。

值： 子网掩码
缺省值： 空

EnableDHCPD_SysNet

如果该变量设为 0，则会阻止工程网络 X120 上的 DHCP 服务器启动。

值： 空、0 或者 1
缺省值： 空

SyncModeDHCPD_SysNet

如果没有将该变量设为"OFF"，工程网络(X120)中的 DHCP 服务器会相互同步，进而只有其中一个服务器分配有效地址。可以通过这种方法同时运行多个 NCU 或 PCU，而无需修改网络设置。

值： ON_MASTER, ON_HIGH (= ON), ON_LOW, 或 OFF
缺省值： ON_HIGH

此时没有激活的 DHCP 服务器进入“待机”模式，它会定期从激活的服务器上读取当前地址数据和 TCU 数据，以便在激活服务器发生故障时待机服务器可以迅速激活，而不丢失数据。

- 如果设置为 **ON_MASTER**，而不是 **ON_HIGH**，则在同步中具有"Master"设置的服务器始终是激活服务器（为此它必须始终联网）。通过这种方法可以确保，正常情况下始终是同一个控制系统用作 DHCP 服务器，其中包含了当前地址数据(/var/etc/udhcp-eth0.leases)和 TCU 数据(/user/common/tcu/...)。只有当工程网络只具有一个 DHCP 服务器时，才可以设为 ON_MASTER；存在多个主机时不允许使用该变量。
- **ON_LOW** 设置可以为 DHCP 服务器指定一个比普通等级更低的优先级。只有在网络中通过"ON_HIGH"或"ON_MASTER"没有找到服务器时，该服务器才会被激活。

说明

和之前版本的兼容性

如果 DHCP 服务器关闭，则会自动使用优先级"ON_LOW"。因此，虽然此 NCU 拥有一个（待机）DHCP 服务器以及一个 Lease 数据备份，但在正常情况下它不会被激活。

InternalDynRangeStart

若有需要，可以通过"InternalDynRangeStart "和"InternalDynRangeEnd "这两个变量精确定义 DHCP 服务器指定的 IP 地址的范围。通常默认设置中的编号范围已经足够使用。

值： IP 地址

缺省值： 工程网络中的第一个地址 + 10 或 +2（≤ 16 个地址时）

InternalDynRangeEnd

值： IP 地址

缺省值： 工程网络中的最后一个地址 - 15 或 -1（≤ 16 个地址时）

DHCPDNoMasterWait

如果一个正在同步的 DHCP 服务器之前有一次曾被定义为主机服务器，在同步中它会一直等待，直至此处给定的时间届满，才变为激活的服务器。

通过这种额外的暂停可以使规定的 DHCP 主机顺利变为激活的 DHCP 服务器，即使该主机的启动比其它控制器稍晚或占用更长时间。

值： 时间，单位为秒

缺省值： 120

InternalDNSDomain

通过该变量可以确定工程网络中使用的 Top Level Domain (TLD)，即：顶级域名。NCU 的 DNS 服务器会将该区域中的名称指定给工程网络中的各个设备。而针对其他所有区域的名称查询将会转送给公司网络上的外部名称服务器。

为避免和全局定义的域名冲突，在缺省设置采用了由 RFC1035 建议的"local"，用于本地网络。

推荐:应保持缺省设置。

值： 域名（字母、数字、-、_；最多 63 个字符）

缺省值： local

DisableNATRouting

预设置允许使用 NAT 路由 (= "Network Address Translation")使机床通过控制系统从工程网络到达公司网络。此时 NCU 将包的源地址转换为其自身的地址并相应地反向返回应答包。

如果该变量被设为零，则会禁止从内部接口 eth0=X120 和 ibn0=X127 到外部网络 eth1=X130 的 NCU 的 NAT 路由选择。为了不会同时关闭两个内部网络的 NAT 路由，可使用值"X120"或"X127"选择关闭工程网络或调试网络。

值： 空、0、1、X120 或者 X127
缺省值： 空

2.3.3 段落 [IBNInterface]

说明

本段落的设置针对 X127 接口。

EnableDHCP_IBNNet

该变量置 1 后，调试接口 X127 上的 DHCP 服务器不会再启动。

值： 空、0 或者 1
缺省值： 空

DisableIBNNet

该变量置 0 后，调试接口 X127 便不再初始化，不再可以使用。

值： 空、0 或者 1
缺省值： 空

EnableSysNetToIBNForwarding

该变量置为 1 时，允许从工程网络 (X120)转送数据包到 X127，通常该过程会被防火墙禁止。

需要注意的是，此时不进行任何 NAT，发送方必须通过相应的路由自己负责以下任务：将发到 X127 （地址为 192.168.215.x）上编程装置或服务 PC 的数据包首先完整地发送到连接设备的 NCU。

值： 空、0 或者 1

缺省值： 空

注意
安全风险 请注意，打开防火墙可能会有安全隐患。仅针对实际所需时间使能转送。请注意，每个可访问的服务也可能有安全漏洞。

DisableIBNForwarding

该变量置为 1 时，数据包从服务接口 X127 到工程网络(X120) 的 NAT 转送功能被取消。

值： 空、0 或者 1

缺省值： 空

2.3.4 段落 [SNMP]

说明

此处可以设置由 SNMP (Simple Network Management Protocol) 提供的字符串，以输出信息。

SNMPLocation

该字符串用于标准 OID SNMPv2-MIB::sysLocation。此处可以输入一个地点，该地点随后可通过 SNMP 客户端调用。

值： 任意字符串

缺省值： 空

SNMPContact

该字符串用于标准 OID SNMPv2-MIB::sysContact。此处可以输入一个联系地址，该地址随后可通过 SNMP 客户端调用。

值： 任意字符串

缺省值： 空

SNMPAutLocation

该字符串用于西门子专用的 OID automationSystemIdent.automationLocationTag。此处可以输入其他一个地点，该地点随后可通过 SNMP 客户端调用。该点和 SNMPLocation 类似，但根据 MIB 说明该变量是出于自动化的角度而给定的。

值： 任意字符串

缺省值： 空

SNMPFunction

该字符串用于西门子专用的 OID automationSystemIdent.automationFunctionTag。此处可以输入一个功能名称，该名称随后可通过 SNMP 客户端调用。

值： 任意字符串

缺省值： 空

SNMPStation

该字符串用于 SINUMERIK 专用的 OID mcSinumerikMIB.mcSinumerikMiscStation。该值是设备所属工作站的名称，具有相同工作站名称的设备因此被视为一组设备。

值： 任意字符串

缺省值： 空

2.3.5 段落 [DCP]

说明

本段中的参数定义了 DCP 协议（Discovery and Basic Configuration Protocol：发现与基本配置协议）的属性，例如此协议在菜单项“可达节点”中由 STEP 7 使用。

InternalDcpEnabled

这样可在设备网络（X120）中激活和取消 DCP，在 NCU 中值还用于 X127。

值： 0 或 1

缺省值： 1

ExternalDcpEnabled

这样可在公司网络（X130）中激活和取消 DCP。

值： 0 或 1

缺省值： 0

2.3.6 段落 [LLDP]

说明

本段中的参数定义了 LLDP 协议（Link Layer Discovery Protocol:链路层发现协议）的属性，此协议被用于一些网络诊断应用。

InternalLldpEnabled

这样可在设备网络（X120）中激活和取消 LLDP，在 NCU 中值还用于 X127。

值： 0 或 1

缺省值： 1

ExternalLldpEnabled

这样可在公司网络（X130）中激活和取消 LLDP。

值： 0 或 1
缺省值： 0

InternalLldpTLVsTxEnabled

使用此参数释放 X120/X127 上 LLDP 包中的更多信息，这些信息通常不包含在内。

值： 数值 0 ... 15
缺省值： 0

值为一个位域，即在需要发送相关信息时，总数值为设定的数值的总和：

1： 端口描述
2： 系统名称
4： 系统描述
8： 权能

ExternalLldpTLVsTxEnabled

使用此参数释放 X130 上 LLDP 包中的更多信息，这些信息通常不包含在内。

值： 数值 0 ... 15
缺省值： 0

值对应参数 InternalLldpTLVsTxEnabled。

2.3.7 段落 [LinuxBase]

说明

在本段落中综合了 Linux 系统的其他设置方法。

说明

设置时区和时间服务器

也可直接通过 SINUMERIK Operate 操作界面上的操作区“调试” → 软键“日期时间”进行设置。

使时间同步

时间同步分为以下情形：

- 如果存在一个外部 NTP 服务器作为时间服务器（记录在文件 basesys.ini 中或者通过 DHCP），则使 PLC 时间和 Linux 时间同步。
- 如果不存在外部时间服务器，则使 Linux 时间和 PLC 时间同步。
  在“清零”时 PLC 时间会保留，不会被复位。而在开关位置 3 “MRES”上接通 PLC 时，时间会被复位。此时采用 Linux 系统的有效时间，无需重新设定。

可通过以下指令获取 NTP 服务器的信息：sc ntpdiag (页 56)

注意
时间同步的边界条件 不要在两个不同的系统上设置时间同步；这表示： 要么在 PLC 项目中通过 PROFINET IO (X150) 选择 NTP 服务器，要么在 basesys.ini 的 NCU 上选择 [LinuxBase] - 但两者不能同时选中，因为在 NCU 上还可指定一个时区，而在 PLC 项目中不可以。

Timezone

系统使用此处设置的时区，以便将 UTC 时间(Universal Time Coordinated)换算成本地时间。该时区可以通过环境变量“TZ”转送给所有的应用程序，并且由 libc 功能 localtime() 观察。

在基础系统中时区对所有时间数据生效，主要是指令“date”、“ls -l”以及系统日志文件 /var/log/messages。

值： 时区说明（参见句法和示例）
缺省值： UTC

示例：

如要将控制系统与所列出的某个时区进行同步，则可直接复制表格中的示例。

区分夏令时/冬令时的时区：

欧洲： WET0WEST,M3.5.0,M10.5.0/3
CET-1CEST,M3.5.0,M10.5.0/3
EET-2EEST,M3.5.0,M10.5.0/3
美国： EST5EDT,M4.1.0,M10.5.0
CST6CDT,M4.1.0,M10.5.0
MST7MDT,M4.1.0,M10.5.0
PST8PDT,M4.1.0,M10.5.0

无夏令时/冬令时的时区：

中国： CST-8
日本： JST-9

时区说明的句法

如要与另一个时区同步，适用以下句法：

tz offset dst,Mmwd,Mmwd

句法（必选）：	
tz offset	时区以及与 UTC 的时差： 零度经线以西时差增加；零度经线以东时差减少。
句法（可选，只在有夏令时时）：	
dst,Mmwd,Mmwd	夏令时区，含起止日期
m (1 ≤ m ≤ 12)	m:月
w (1 ≤ w ≤ 5)	w:一个月中的星期，即 w = 5 时为一个月的最后一星期
d (0 ≤ d ≤ 6)	d:星期几（从 0 = 星期日到 6 = 星期六）

PLCTimeType

该变量指定了所设的 PLC 时间的类型：

- `local-nodst` 适用于**不带**夏令时的本地时间，即使时区中定义了一个夏令时。在夏天，所有时间慢 1 个小时，而不是时间跳跃。如果 PLC 时间设为夏令时，Linux 时间则快 1 个小时。
  此特性在导入该变量时已使用，因此，出于兼容性原因，也可默认为缺省设置。
- 如果相应的时区中有夏令时，`local` b 表示**带**夏令时/冬令时切换的本地时间。如果 Linux 的时间是由外部服务器指定的，则通过此设置每年切换两次 PLC 时间。

注意
PLC 时间报警缺失或翻倍 如果 Linux 的时间是由 PLC 指定的，则必须手动或从外部自动在夏令时和冬令时之间切换。如未执行，在夏天 Linux 时间慢 1 个小时。

- `utc` 表示 PLC 时间被设为世界时间。由于在 UTC 内部也可管理 Linux 时间，使得时间调整更为简单。由于 UTC 从没有夏令时，因此在该设置下，不会有时间跳跃。但 SINUMERIK Operate 和 PLC 的配置工具都无法识别该模式或支持从 UTC 时间切换为本地时间，所以多数情况下显示的是一个与日常时间存在偏差的时间。

值： local-nodst, local 或 utc

缺省值： local-nodst

LogFileLimit

借助该变量可以限制系统日志文件 `/var/log/messages` 的大小。"0"值表示日志文件的大小不受限制。

为提高效率，此处给定的极限值不应视为硬限制，而应视为目标值。如果日志文件达到目标值的 1.5 倍，文件大约会缩小为目标值的 75%。系统会每 2 分钟（最长间隔）检查文件是否太大。

值： 大小，单位为字节

缺省值： 100 kB

FirewallOpenPorts

此处您可以给定一个端口列表，在外部网络接口的防火墙中会打开这些端口。日志为"TCP"或"UDP"、端口、需要释放通讯的端口的编号。多个端口数据用空格隔开。

值： 日志/端口对的列表

缺省值： 空

注意
安全风险 请注意，打开防火墙的端口潜在有安全隐患。请只有在确实需要的情况下才打开防火墙端口。 请注意，每个可访问的服务也可能有安全漏洞。

DisableSubsystems

借助该变量可以阻止特定子系统（CFS 系统）的安装和启动。多个子系统名称可以用空格隔开。

值： CFS 名称列表（用空格分隔）

缺省值： 空

名称可以就是 CFS 基本名称，如 nck，这样所有具有该名称的 CFS 系统都会被忽略。也可以是一个绝对路径，如`/siemens/sinumerik/nck`，这样只有该路径下的 CFS 系统被忽略。

也可以设置"DisableSubsystems=all"，这样所有的 CFS 系统都无法安装或启动。

DefaultCIFSSecurity

Linux 指令 sc connect 为 CIFS 网络驱动器提供了一个选件 "-sec=TYPE"，借助该选件在连接至服务器时可以设置熔断器类型。如果在调用 sc connect (页 50) 时未指定该选件，则可使用 basesys.ini 文件中的变量 DefaultCIFSSecurity 的值：

none	无认证，仅可作为客户机连接 - 服务器允许时。
ntlm	NTLMv1 哈希密码（兼容性原因）
ntlmi	NTLMv1 带强制包-发送信号（兼容性原因）
ntlmv2	NTLMv2 哈希密码
ntlmv2i	NTLMv2 带强制包-发送信号

ntlmssp　　NTLMv2 通过 NTLMSSP 包装

ntlmsspi　　NTLMSSP 带强制包-发送信号

（krb5 和 krb5i 会因缺少 Kerberos 基础设施而无法在 NCU 上使用。）

值：　　none、ntlm、ntlmi、ntlmv2、ntlmv2i、ntlmssp、ntlmsspi

缺省值：　　ntlmssp

DisableNTPTimeSync

如果设置此变量为 1， 则不启动时间同步 NTPD 服务器(Network Time Protocol Daemon)也不进行任何时间同步。

值：　　0 或 1

缺省值：　　0

DisablePLCTimeSync

如果设置此变量为 1，则不进行任何系统时间和 PLC 时间的同步（两个方向）。

值：　　0 或 1

缺省值：　　0

EnableCoreDumps

如果将该变量设为 1，则会创建一个此过程的日志文件，并保存在`/var/tmp` 路径下。在日志文件中包含了异常中断或者自动结束过程的内存转储。

值：　　空、0 或者 1

缺省值：　0

CheckTCUforUpdatePackage

如果将该变量设为 1，则会在引导启动过程中检查 TCU 正面 USB 接口上的存储介质。如果检查发现了升级包，例如使用 Create MyConfig 创建的类型为 *.usz 的文件，则会启动该升级包的安装。

预设为零时会跳过该项检查，节省时间。

值： 0 或 1
缺省值： 0

EnableUsbShares

当该变量设为 1 时，USB 存储器会作为共享网络驱动连接。

当该变量设为 0 时，USB 存储器会作为独立 USB 存储设备连接。

值： 0 或 1
缺省值： 1

2.3.8 示例： 配置文件"basesys.ini"

配置文件 basesys.ini

目录 /siemens/system/etc 下提供的文件 template_basesys.ini 的缺省设置如下：

```
;
------------------------------------------------------------------
-
; Default Linux basesystem configuration
;
------------------------------------------------------------------
-
;
; section ID is for Windows compatibility and is ignored ;)


[ExternalInterface]
; If DisableCompanyNet is set to 1, the whole interface will be
disabled.
;DisableCompanyNet=1
```

```
; If ExternalIP is set, you can force the external Ethernet
; interface to use a fixed IP address etc. instead of using DHCP
; If a Hostname is set, it even overrides one received by DHCP
; Please note that only letters, digits and '-' are allowed in
Hostname;
; specifically '_' or '.' are forbidden!
;ExternalIP=210.210.210.210
;ExternalNetMask=255.255.255.0
;Gateway=210.210.210.1
;Nameservers=210.210.210.1 210.210.210.2
;Timeservers=210.210.210.3
;Hostname=somename
;Domain=example.com

; If ExternalIP is empty (default), DHCP is used with the
following
; ClientID the default is "@MAC" to use the MAC address,
; alternatives are "@NAME" to use the hostname (Hostname above),
; "@CFID" to use the ID of the CF card if one is available
; (like on X120), or any other arbitrary string
;DHCPClientID=@MAC

[InternalInterface]
; If DisableSysNet is set to 1, the whole interface will
; be disabled
;DisableSysNet=1

; With InternalIP and InternalNetMask (both must be set
together),
```

```
; you can change the address on the internal/TCU/automation/
system net.
;InternalIP=192.168.214.1
;InternalNetMask=255.255.255.0

; These two can define an alias IP for X120.
;InternalIP_Alias=192.168.216.1
;InternalNetMask_Alias=255.255.255.0

; This defines a "name of station" for ProfiNet (ERTEC interface)
; for use with DCP, default is the normal hostname.
; Please be careful if a PN-Name can also be set by other means,
; for example by NCU-Link. Usually it is not needed to define
; a name here.
;PN_StationName="PN_IO"

; Setting EnableDHCPD_SysNet to 0 suppresses that a DHCP server is
; started on that interface
;EnableDHCPD_SysNet=0

; Set synchronization of all DHCP servers in the
; system/TCU network (X120). Possible values are:
; OFF,
; or ON_LOW (low priority),
; or ON or ON_HIGH (normal priority),
; or ON_MASTER (highest priority),
; or ON_CLIENT_SYNC (starts DHCP client only, synchronization
active),
```

```
; or ON_CLIENT_NO_SYNC (starts DHCP client only, synchronization
inactive).
; ON_MASTER is used to make this machine deterministically the
; DHCP master server, but should be used only for one server
; in the network.
;SyncModeDHCPD_SysNet=ON

; This is the range of dynamic IPs given out by the DHCP server
; Defaults should be sensible
InternalDynRangeStart=192.168.214.10
InternalDynRangeEnd=192.168.214.239

; If once a DHCP master was seen, the DHCP server waits this many
; seconds for the master to show up before trying to become the
; active server
;DHCPDNoMasterWait=120

; DNS domain name used for names in system/TCU network
; (default is "local" to conform with RFC)
InternalDNSDomain=local

; FixedDomain and FixedNameservers can be used to define a DNS
domain
; and nameservers, if the DHCP/DNS server on system network
; is disabled.
;FixedDomain=local
;FixedNameservers=192.168.214.1

; If DisableNATRouting is set to 1, the NCU won't forward
```

```
; from TCU or IBN net
; (X120 anad X127, resp.) to external net (X130).
; If set to "X120" or "X127", forwarding will only be
; disabled from that interface.
;DisableNATRouting=1

[IBNInterface]
; If DisableIBNNet is set to 1, the whole interface will be
disabled
;DisableIBNNet=1

; Setting EnableDHCPD_IBNNet to 0 suppresses that a DHCP server
; is started on that interface
;EnableDHCPD_IBNNet=0

; If DisableIBNForwarding is set to 1, the NCU won't do NAT
routing
; from IBN network (X127) to system network (X120).
;DisableIBNForwarding=1

; If EnableSysNetToIBNForwarding is set to 1, the NCU forwards
; packets from X120 to X127. (Please note that the sending host
must be
; able to route the packets to the NCU by its own means, the NCU
just
; accepts and forwards packets to 192.168.215.x with this
setting.)
;EnableSysNetToIBNForwarding=1

[SNMP]
```

```
; The following variables define strings which are delivered on
; certain SNMP requests.
; SNMPLocation and SNMPContact are reported for the standard OIDs
; SNMPv2-MIB::sysLocation and SNMPv2-MIB::sysContact, resp.
; SNMPAutLocation and SNMPFunction are used in the Siemens A&D
specific
; AUTOMATION-SYSTEM-MIB as
automationSystemIdent.automationLocationTag
; and automationSystemIdent.automationFunctionTag, resp.
SNMPLocation="not specified"
SNMPContact="not specified"
SNMPAutLocation="not specified"
SNMPFunction="not specified"
SNMPStation=""

[DCP]
; These variables en/disable the DCP protocol on X120/X127
; and X130, resp.
;InternalDcpEnabled=0
;ExternalDcpEnabled=1

[LLDP]
; These variables en/disable the LLDP protocol on X120/X127
; and X130, resp.
;InternalLldpEnabled=0
;ExternalLldpEnabled=1

[LinuxBase]
; Keep size of /var/log/messages around this value
```

```
; (not followed exactly for performance reasons)
LogfileLimit=102400

; Protocol/Port pairs to open in the firewall
; (e.g. TCP/5900, UDP/514, ...)
;FirewallOpenPorts=TCP/5900 TCP/102 TCP/22

; If there is only one Ethernet interface, it's used by
; default as an external (company) network. Alternatively,
; usage as automation net (TCU boot support etc.) is possible
; by setting NetworkModel to "automation". (NCU only, Service
; and PCU20A variants!)
;NetworkModel=automation

; DisableSubsystems can be used to skip certain CFSes (=
subsystems)
; a list of multiple names (separated by spaces) is possible
; a simple name means all CFSes with this name, an absolute path
; (e.g.,/siemens/sinumerik/nck) exactly this one CFS with
; the special value "ALL", all subsystems can be disabled
;DisableSubsystems=nck

; Directory for OEM netboot files (served by tftp to diskless
clients)
NetbootDirectory=/oem/common/netboot

; Default security type for all CIFS mounts, if not given
explicitly
; by sc connect -sec=TYPE. If not defined, NTLMv1 is used or
NTLMv2
```

```
; if the server allows.
;DefaultCIFSSecurity=ntlmssp

; Properties of local time zone: names, offset, start and end day
; Some examples:
;  Europe: WET0WEST,M3.5.0,M10.5.0
;          CET-1CEST,M3.5.0,M10.5.0
;          EET-2EEST,M3.5.0,M10.5.0
;   USA:   EST5EDT,M4.1.0,M10.5.0
;          CST6CDT,M4.1.0,M10.5.0
;          MST7MDT,M4.1.0,M10.5.0
;          PST8PDT,M4.1.0,M10.5.0
;   China: CST-8
;   Japan: JST-9
Timezone=UTC
; TimezoneName(for-into-only-do-not-change)=Etc/Universal

; If DisablePLCTimeSync is set, no time synchronization with
; PLC will happen
;DisablePLCTimeSync=1

; PLCTimeType informs time synchronization what kind of time
; is kept in PLC:
; local: PLC uses "wall clock time", with switches between DST and
;   standard time, if DST exists in the selected Timezone
;  If Linux synchronizes to an external server and passes time on
;   to PLC, this means the PLC clock is changed twice a year.
;   For PLC time-of-day interrupts this means they can be skipped
```

```
;   (begin of DST) or executed twice (end of DST). If this might
be
;   an issue,try to avoid such interrupts between 2am and 3am.
;   If Linux receives time from PLC, it assumes that this clock
will
;   switch between DST and standard time somehow (manually or
;   automatically). If this doesn't happen (or is delayed) Linux
time
;   will be 1h behind the correct value. NCK and drives will still
;   follow PLC clock.
;
; local-nodst: PLC uses local time as above, but without DST
;   (summertime). All components (PLC, NCK, HMI, drives) always
;   use standard time, even if the selected timezone has a DST.
;   If Linux receives time from the PLC, it assumes standard
;   (non-DST) time, too. If this is not the case, the Linux clock
;   will be 1h ahead of the correct value.
;   This time type has been used implicitly up to SW 4.7.3 and is
;   still the default to maintain compatibility.
;
; utc: PLC clock contains universal time (= GMT+0)
;   All components (PLC, NCK, HMI, drives) will use UTC.
;   However, S7-300 engineering and HMI Operate 4.7 are not
prepared
;   for displaying UTC times as local times yet.
;   This type makes synchronization simple, as no changes for
;   timezone or DST are needed. It also avoids changes to PLC clock
;   and supports applications across different timezones.
;   If Linux receives time from the PLC, it assumes UTC there.
;   If this is not the case, the Linux clock will have a permanent
```

```
;   offset according to local timezone.
;PLCTimeType=local

; Set to 1 to enable coredumps in /var/tmp
EnableCoreDumps=0
```

2.4 服务命令

2.4.1 使用服务命令

概述

维修命令‘sc’是执行 SINUMERIK NCU 上不同维护任务的工具。将所需要的动作写在命令行的‘sc’后面，例如：sc help

该动作会输出所有动作的列表，并附有简要描述。在该动作之后，需要时可能有其他参数或者选项。

不管是在 NCU 基础系统中，还是在维修系统（通常为 USB 存储器形式）中，都可以使用‘sc’。尽管如此，有些动作仅在一个或者另一个系统（NCU/维修）中有意义，这要根据各个动作的说明来确定。

调用维修命令

维修命令'sc'既可以由维修系统调用，也可以由 TCU 的维修控制台调用：

- 维修系统：在主菜单中选择菜单项“Service Shell”。
- 如要在 TCU 上调用维修控制台，需要按下以下按键：



权限等级

‘sc’命令可以执行带有更多权限的动作，比通常情况下提供给所调用用户的权限要多。例如：在启动或者停止子系统时需要引导权，尽管如此 'sc' 允许每个属于‘service’组的用户拥有该权限。

每个‘sc’动作分配了一个权限等级。这是一个用户组，用户必须属于这个组，以便能够执行该动作。因为这些组按等级来划分，所以当然也允许“较高”组的成员来使用有关的动作。这样‘manufact’组的权限就高于‘service’组，也就是说，‘manufact’组的成员允许调用所有‘service’权限等级要求的动作。

使用单个动作时会分别注明，该动作要求调用者具有哪个权限等级。按升序排列的可能的等级为：

- 无
- operator

- user
- service
- manufact

示例：

一个标识为'user'权限等级的动作总是可以由'service'和'manufact'组的成员来执行。每个用户都可以调用权限等级为“无”的动作。

如果某个用户不具有所需权限，则会输出下列故障信息：

```
Action 'ACTION' needs at least GROUP privilege level.
```

2.4.2 动作的句法

说明

'sc'的命令行原则上不区分大小写。

下列输入是完全相同的：

sc help show

SC help SHOW

sc HeLp sHoW

但大小写在有些情况下也是有区别的，例如：文件名或者用户名。 但这些情况要避免。

对此，使用的惯例是：

- 全部大写的名称可以替换为那些需要根据情况来使用的动作。
  示例： `sc help ACTION`
  此处 ACTION 可以用一个动作来替换，以便获取该动作的描述。 小写的部分只要按照规定那样输入即可。
- 方括号标识出可选择的参数。
  示例： `sc help [ACTION]`
  某个动作的说明是可选的，可以选择，但不是必须选择。 方括号有时也可以嵌套：
  ... [USERNAME[/DOMAIN]] ...
  此处 USERNAME 和 DOMAIN 两者都是可选的，但只有当 USERNAME 存在时才可以规定 DOMAIN。

- 有多个参数可以选择时中间用'|'隔开。
  示例： `sc start all|system|SUBSYSTEM`
  也就是说，下列命令中的任意一个都是有效的：
  `sc start all`
  `sc start system`
  `sc start SUBSYSTEM`
  最后一种情况大写的 SUBSYSTEM 还可以由一个具体的子系统名称来替换。
- 简写方式可能将供选择的参数写在方括号内。
  `sc save [-full|-user] ...`
  可以选择使用选项 "-full" 或者 "-user" 或者都不选。
- 用'-'开头的选项的排列顺序可以是任意的。
  因此书写方式可以这样解释，"-force" 必须在 "-full" 或者 "-user" 的后面，但这不是必然的情况：
  `sc save [-full|-user] [-force] FILENAME`

2.4.3 类单元

概述

在这一段中会描述由多个动作所使用的句法单元。

允许的接口名称

网络接口的名称可以由例如"sc show ip"或"sc enable DHCPSv"等指令使用。

因为接口说明是可选的，所以总是用'-'来引导名称。 在'-'之后才是真正的名称，多个名称可能指同一个接口。

可接受的名称是：

设备网络端口：	"X120", "eth2", "tcu", "intern"
公司网络端口：	"X130", "eth1", "factory", "extern
开机调试端口：	"X127", "ibn0", "pg"

允许的子系统名称

子系统名称例如可以在使用 "sc enable" 和 "sc start" 时给出。 在大多数情况下，子系统名称就是所属的 CFS 的名称，没有路径说明和扩展名 ".cfs"。 如果仔细观察一个文件名，例如：CFS /siemens/sinumerik/nck.cfs，会发现子系统名称就是 "nck"。

除此之外也可以在一个子系统名称中使用绝对路径说明（以 / 开头）。 在上面的例子中也可以使用 "/siemens/sinumerik/nck" 作为子系统名称。 带和不带路径说明的名称之间的区别是，在没有路径说明的情况下，会认为是所有该名称的 CFS 文件，在有路径说明时，则仅仅只针对确定的那一个文件。

子系统

子系统是一个 CFS, 其不仅包含多个文件、而且运行时会执行某个程序。 对此， CFS 包含一个脚本，用该脚本可以控制该程序的启动和停止。

出于这个原因，只有管理员才允许对 NFS 文件系统进行设置，并且 NFS 大多设置为统一的管理环境。 服务器输出的文件系统直接通过其路径在服务器上进行响应。

CFS (compressed file system)

CFS （文件结尾 ".cfs"）是一个压缩文件系统，和 zip 文件基本类似。 它包含文件和子目录，这些文件和子目录在控制系统上运行时和普通文件几乎一样。 CFS 中的文件和目录不可以进行更改。 它们会在运行时根据需要进行解压缩。

NFS (network file system)

NFS 是在 Unix 环境下最常用的远程文件系统协议，也可用于 Windows 操作系统。 Unix 计算模型在每次存取时都会同时提供一个 UID 和 GID，服务器根据这些数据决定是否允许进行操作，因此计算模型很大程度上决定了 NFS。 服务器默认用户提供的是正确的 ID。

Remote File System

远程文件系统是通过网络进行响应的文件系统。 这些文件物理上位于网络中的另一台计算机上（“服务器”），但在本地也可以像其他所有文件一样显示。 对此类文件的操作可以通过网络传到服务器上，而不是直接在本地存储媒介（硬盘，CF 卡）上进行。

因为一个服务器通常输出多个文件系统，所以除了规定服务器的名称之外，还要规定所需文件系统的名称。

SMB (server message block)

SMB 是一个协议，MS Windows 的远程文件系统是以该协议为基础的（也可以作为网络驱动器、使能、Share 等）。 SMB 连接在特定用户的联系中总是有效的，该用户在服务器上必须是已知的。 输出的文件系统有一个名称（使能名称），以该名称响应文件系统，用户无需知道服务器上具体的路径。

2.5 动作描述

2.5.1 Help

描述

句法： sc help [Action]

可选择的名称： -h, - -help

权限等级： 无

调用“sc help”时，如后面没有其他动作，则会输出一个列表，上面包含所有允许的动作，并带有简要描述。如果另外规定一个动作，则会输出该动作的详细描述。

示例：

```
sc help
All actions:
help [ACTION]
Print help about a specific or list all actions
restart
Reboot the machine
enable hmi|nck|SUBSYSTEM...
enable DHCPSvr -INTERFACE
Enable HMI, NCK, or any other subsystem
[...]
sc help enable
enable hmi|nck|SUBSYSTEM...
DHCPSvr -INTERFACE


Enable subsystem(s), like 'hmi', 'nck', and so on. A subsystem
name
is the name of the CFS containing it, without the '.cfs'
extension.
```

```
This enables all CFSes with that name, but you can also use a full
path (e.g., /siemens/sinumerik/nck) to enable just a specific CFS.
Another form is to enable the DHCP server on a network interface,
for example 'enable DHCPSvr -X120'.
```

2.5.2 Check-cf

描述

句法：	sc check-cf
可选择的名称：	checkcf
权限等级：	user

用该动作可以检查 CF 卡是否含有坏扇区。如果出现故障，会记录在 /var/log/messages 文件中。

2.5.3 Clear

描述

用 “sc clear” 指令可以执行不同的指令。

sc clear dhcp

句法：	sc clear dhcp [-INTERFACE]
可选择的名称：	---
权限等级：	service

用该指令在指定的接口（预设置为设备网络）上删除 DHCP 服务器的任何状态并将其设置到初始状态。表示，删除了租借数据：网络中的所有 IP 地址将重新分配，服务器忘记一次性检查主机服务器。

该指令仅在具有有效 DHCP 服务器的机床上运行。

说明

删除租借数据并不是删除了全部文件，而只删除了所包含的数据。版本号也升级，则现有的待机 DHCP 服务器能够接受该删除。

sc clear 预装键

句法：	sc clear preinstalled-keys
可选择的名称：	---
权限等级：	service

该指令可删除控制系统上的由西门子预装的所有 SSH 钥匙。调用服务系统时，此时只涉及 CF 卡上的钥匙，而不涉及服务系统上的 SSH 钥匙本身。

注意
访问控制系统 删除西门子预装的 SSH 钥匙可排除扰动风险。 为了确保有足够的权限访问系统，建议删除前先指定和安装新的 SSH 钥匙。

2.5.4 Closeport

描述

句法：	sc closeport ID
可选择的名称：	---
权限等级：	service

工作原理

该指令再次关闭防火墙中通过 “sc openport” 打开的端口。当 “sc openport” 设置的时间用尽前， 应当再次关闭该端口时，该指令才有意义。否则，该端口生命周期用尽时将自行再次关闭。

参数是防火墙规则的 ID 编号，由 “sc openport” 输出。

2.5.5 Connect

说明

句法：	sc connect [-ro] [-sec] SERVER:/PATH [MOUNTPOINT] sc connect [-ro] [-public] [-sec] //[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE [MOUNTPOINT]
可选择的名称：	mount
权限等级：	无

该动作可以允许在控制系统上使用某个服务器的远程文件系统。为此要连接一个本地目录，即所谓的 “MOUNTPOINT”。在该目录下可以看见服务器提供的文件。

注意
口令保护 请注意，在输入符合规定句法的指令时，口令在屏幕上显示为明码文本。 并且在输入路径时请使用斜杠“/”，而不是反斜杠“\”。

支持的文件系统

支持两种类型的远程文件系统：Windows SMB 和 Unix NFS。这两种类型的特性完全不同，特别是在用户管理方面。

- 使用 Windows SMB 时，要作为服务器能够识别的特定用户进行连接。通过建立的连接，作为该用户可以存取文件，本地用户是否存取文件对其不会产生影响。
  需要使用该特性必须在 SMB 进行连接时给出一个用户名，需要的时候还要给出域名和口令。
- 使用 NFS 时连接本身不需要给出特定的用户名，而是在每次进行文件操作时都要报告给服务器是谁要执行该操作。然后由服务器决定是否允许该操作。用户说明要有用户 ID 和组 ID，不需要名称。服务器必须识别相应的 ID （或者允许任何人进行存取操作。）
  所支持的远程文件系统的另一种类型是由 TCU 输出的 USB 存储媒介（USB 闪存驱动器）。该存储媒介在 NFS 的帮助下进行连接，服务器和路径的说明看起来和 NFS 的类似。当然， TCU 名称的管理会有些不同，并且 USB 存储媒介有特殊的，物理上不存在的路径。

借助 “security type” 认证

不同的 Windows 服务器会针对所用认证程序提出要求。如果服务器只接受特定类型且缺省设置不匹配，在 CIFS 和 SMB 上可通过选件 -sec 指定一个熔断器类型。

允许的值有：

- ntlm（= NTLMv1；仅出于兼容性原因存在）
- ntlmv2
- ntlmv2i
- ntlmssp
- ntlmsspi

结尾处的 "i" 要求对发送通讯信号。

如果没有选件 -sec，则使用文件 basesys.ini (页 27) 中的变量 "DefaultCIFSSecurity" 的值。如果该值也未指定，则默认使用 NTLMv1 或 NTLMv2。

注意
认证 建议不再使用 NTLMv1，因为安全性无法再得到保障。

远程文件系统的标记

远程文件系统的说明对于 SMB 和 NFS/TCU 会使用不同的标记：

SMB: //[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE

固定部分为：//SERVER/SHARE

服务器名称当然也可以是一个数字的 IP 地址。SHARE 是服务器上释放份额的名称。请注意，在此类名称中经常出现的'$'字符，在命令行中前面必须加一个反斜杠 ('\')，否则会尝试扩展一个变量。

在服务器名称前还可以有一个用户名，用'@'字符隔开。如果必要，还可以用'/'和 Windows 域名来扩展该用户名，该用户名从属于该域名。属于该用户的口令通常情况下会进行互动式的询问，这样在屏幕上就看不见口令了。

对于特定的应用程序可能仍然需要将口令同时写入命令行。（例如：从键盘不能读取由 WinSCP 启动的程序）。在这种情况下可以在口令后再附上一个'%'字符。如果密码包含任意一个由 Shell 解释的特殊字符 (<, >, &, ;, ", ', $, (,), |)，则应该用一个前置的反斜杠将其屏蔽。逗号在 SMB 密码中不能解释。

NFS:SERVER:/PATH

使用 NFS 时取消用户说明，仅保留两部分，SERVER 和 PATH。SERVER 和前面一样是文件服务器的名称。和 SMB 不同的是 PATH 是一个在服务器上已经存在的路径名称，而不是一个任意分配的名称。

命令 “sc show drives SERVER” 已经以正确的标记显示由某个特定文件服务器提供的文件系统。对于 SMB，在需要时还是必须补充用户名等。

在远程文件系统说明后面还可以规定所需的目录 ('MOUNTPOINT')，该目录下在本地可以看见远程文件。这应该是一个空的目录，对于该目录有写入权限。如果删除 MOUNTPOINT，则 'sc' 会自动生成一个合适的目录。名称是 /tmp/drvNN（连续编号），成功连接时会显示出来。

选项 “-ro” 以只读模式连接远程文件系统，也就是说，在那里只能读取，但不能写入。如果要禁止从控制系统进行更改，或者服务器仅允许进行只读连接，则该选项是有用的。

如上所述，一个 SMB 文件系统会作为规定用户进行连接。为了避免第三方用户在控制系统上以连接用户的名义在服务器上执行文件操作，通常情况下，只有最后一个用户（和所有其组中的成员）可以对本地目录进行存取。

但如果要让其他本地用户也能使用远程目录，则可以用选项 “-public” 来要求 MOUNTPOINT 从属于组 “operator”，这样就可以进行存取。

2.5.6 Disable

说明

句法：	sc disable hmi|nck|SUBSYSTEM ...
	sc disable DHCPSvr -INTERFACE
	sc disable DHCPSync [-X120]
	sc disable IPAlias [-X120]
	sc disable usb [-MINUTES] all | HOSTS
可选择的名称：	---
权限等级：	service

“sc disable” 指令关闭了指定的子系统，在下次重新启动时不再加载。也可以使用一个“disable” 的指令来关闭多个子系统。可以使用 “hmi”、“nck” 或者任意其他子系统名称作为子系统名。

说明

指令 “sc disable” 对位于以下目录中的 basesys.ini 文件中的参数生效：user/system/etc/basesys.ini

使用 “sc disable IPAlias” 取消激活网络接口 X120 上的第二个假 IP 地址。

另见

文件 “basesys.ini” 中的参数 “InternalIP_Alias” 和 “InternalNetMask_Alias”。

特殊子系统

“DHCPSvr” 名称是一个特殊情况，因为它不能启动或关闭一个正常的子系统，而是会影响 DHCP 服务器是否在网络接口上启动。该接口会以标准形式给出。

使用 “DHCPSync” 可以启用并关闭 DHCP 服务器的同步。该同步只能在工程网络上实现。

指令 “sc enable” 和 “sc disable” 可以更改文件 /user/system/etc/basesys.ini 中的特定变量。子系统的相关变量为 “DisableSubsystems”，DHCPSvr 的相关变量为 “DisableDHCPD<INTERFACE>”。

通过 “sc disable usb” 禁用 USB 存储设备：具体涉及 USB 类型为 “storage” 的设备，即：输入设备也可通过 USB 接口运行。

- 如不给定时间 [-MINUTES]，则设为永久禁用。在其中设定了时间时，系统按照该时间禁用 USB 存储卡。
- 工程网络中的所有设备都可以被禁用，或者输入需要禁用的一系列主机名称或 IP 地址禁用设备。

2.5.7 Disconnect

说明

句法：	sc disconnect MOUNTPOINT
	sc disconnect all
可选择的名称：	umount, unmount
权限等级：	user

该指令 “sc disconnect” 会断开之前用 “sc connect” 进行连接的远程文件系统。如果 ‘sc’ 自动创建 MOUNTPOINT，则相应的目录也会再次被删除。

一种形式是 “sc disconnect -all”，这会删除所有存在的 SMB 和 NFS 文件系统。

2.5.8 Distribute

描述

句法：	sc distribute [parameter]
可选择的名称：	dist
权限等级：	service

用该指令将 TCU 数据分配到设备网络的其他机床上。这里只涉及参数 “tcudata”。

distribute tcudata

句法：	sc distribute tcudata
可选择的名称：	---
权限等级：	service

该指令向有效的 DHCP 服务器发送提示，在 /user/common/tcu 下手动更改 TCU 数据。由此，该服务器能够向待机服务器继续发送这种数据。

不推荐在 /user/common/tcu 下手动更改版本文件，因为 DHCP 服务器只将该版本文件读入一次。升级后的版本只在重启时生效。

说明

不仅在有效的服务器上，还可以在设备网络上的任意机床上执行该指令。在每种情况下，发送消息到有效的服务器上。

2.5.9 Enable

说明

句法：	sc enable hmi|nck|SUBSYSTEM ...
	sc enable DHCPSvr -INTERFACE
	sc enable DHCPSync [-X120] [-PRIORITY]
	sc enable IPAlias [-X120] IPADDR[/NETMASK]
	sc enable usb [-MINUTES] all | HOSTS
可选择的名称：	---
权限等级：	service

指令 “sc enable” 可以启用指定的子系统，在启动某个子系统它一同被激活。可以只用一个 “enable” 指令来启用多个子系统。可以使用 “hmi”、“nck” 或者任意其他子系统名称作为子系统名。

说明

指令 “sc enable” 对位于以下目录中的 basesys.ini 文件中的参数生效：user/system/etc/basesys.ini

使用 “sc enable IPAlias” 来定义用于网络接口的第二个假 IP 地址。该指令只用于 X120。IP 地址的普遍书写方式是 4 个由点隔开的数值。如果没有给定子网掩码，它可由 IP 的等级(A/B/C)来确定或者清楚确定。此时也允许使用 CIDR 符号（网络部分的位数）。

另见

文件 “basesys.ini” 中的参数 “InternalIP_Alias” 和 “InternalNetMask_Alias”。

特殊子系统

“DHCPSvr” 名称是一个特殊情况，因为它不能启动或关闭一个正常的子系统，而是会影响 DHCP 服务器是否在网络接口上启动。该接口会以标准形式给出。

指令 “sc enable” 和 “sc disable” 可以更改文件 /user/system/etc/basesys.ini 中的特定变量。子系统的相关变量为 “DisableSubsystems”，DHCPSvr 的相关变量为 “DisableDHCPD<INTERFACE>”。

使用 “sc enable usb” 再次使能永久禁用的 USB 接口上的或者特定主机或 IP 地址上的 USB 存储设备。

DHCP 同步

使用 “DHCPSync” 可以启用并关闭 DHCP 服务器的同步。只允许在工程网络(X120)中进行此同步。另外还可以给定优先级：LOW、HIGH 或 MASTER。

对于 DHCP 服务器的同步，优先级的作用如下：

- **MASTER:** 计算机节点是激活的 DHCP 服务器。如果在工程网络中设计了多个服务器，则该计算机具有最高优先级。
- **HIGH:** 该计算机节点属于带有较高优先级的服务器成员，即：当没有带有 “MASTER” 标识的服务器被激活时，可以将带有 “HIGH” 标记的计算机当作激活的服务器。
- **LOW:** 该计算机节点属于带有较低优先级的服务器成员，即：当没有带有 “MASTER” 或 “HIGH” 标记的服务器被激活时，可以将带有 “LOW” 标记的计算机当作激活的服务器。

说明

推荐的设置：
- 在工程网络中启用 DHCP 运行和 DHCP 同步。
- **只有一个** NCU 被设置为激活的 DHCP 服务器。
- **最多 2 个**计算机是带有 “HIGH” 标记的成员。
- 所有其他组件都设置为 DHCP 客户端或者带有 “LOW” 标记的成员。

另见

文件 “basesys.ini” 中的参数 “SyncModeDHCPD_SysNet”。

2.5.10 ntpdiag（诊断）

描述

句法：	sc ntpdiag
可选择的名称：	---
权限等级：	无

通过指令 “sc ntpdiag” 可以输出一个已配置的时间服务器及其状态的列表。

此外还会输出使用 NCU 的时间服务器的信息，如 IP 地址、层（与时间信号接收器的距离，如原子钟）及 NTP 服务器与其客户端之间的偏移。如果与服务器的连接不良，则删除 Ping 指令，以检查连接性能。

2.5.11 Openport

说明

句法：	sc openport [-MINUTES] PROTO/PORT SOURCE ...
可选择的名称：	---
权限等级：	service

功能方式

该指令在一定时间内打开用于公司网络（X130）防火墙中的端口。标准时间为 15 分钟，也可以更改选项 -MINUTES 。最大可能为 60 分钟。

在格式 “PROTO/PORTNR” 中指定了要打开的端口。该协议为 “tcp” 或 “udp”。随后是一个主程序从端口要达到的数据。多个格式可能：

- 单个 IP 地址：“128.128.12.12”
- 主程序名称（如果通过 DNS 可以释放），例如：“server”
- 带网络掩码数据的 IP 范围，例如 "128.128.12.0/255.255.255.0"
- 有效位数据的 IP 范围，例如 "128.128.12.0/24"

如果完全缺少源主机的数据，通常故障消息来应答。当 “sc openport” 指令在一个通过 SSH 打开的 SHELL 中输入，为一个特例。从环境变量 $SSH_CLIENT 中取到允许的主程序，包含 SSH 客户的 IP。

如果成功打开，在状态消息中输出一个 ID 编号：

```
sc: Port tcp/25 is open, rule ID is 6620
```

该 ID 可与 “sc closeport” 一起使用来手动关闭上一个端口。

注意
安全风险 请注意，打开防火墙的端口潜在有安全隐患。请只有在确实需要的情况下才打开防火墙端口。 请注意，每个可访问的服务也可能有安全漏洞。

另见

文件 “basesys.ini” 中的参数 “FirewallOpenPorts”。

2.5.12 Portstatus

说明

句法：	sc portstatus [-x127] PORTSPEC SOURCE
可选择的名称：	---
权限等级：	service

生效方式

通过指令 “sc portstatus” 可以查看防火墙中的某个端口是否打开。如果指的是 X127 防火墙（非 X130 防火墙），则须指定选项 “-x127”。端口规格参见 “sc openport”，例如：“tcp/102”。但是在该位置上不允许名称，必须输入唯一的“Source-IP”。

该指令会输出以下状态：

- 退出状态为 1 的 “Port not open”
- “Port is open (rule ID is <N>)”和退出状态 0。
  此时会一并提供在 “sc closeport” 时给定的 ID。

2.5.13 Restart

说明

句法：	sc restart all|system|network| [!]SUBSYSTEM
可选择的名称：	reboot
权限等级：	service

通过动作 “sc restart” 先停止指定的子系统，然后通过以下边界条件重新启动：重启时会同时重新启动相关子系统。

句法

作为子系统，可以使用 ‘hmi’、‘nck’ 或者其他任意的子系统名称。此处也可以依次指定多个子系统，这些子系统的关联子系统会一同暂停然后重新启动。您也可以在其中指定不重启的系统，为此需要在清单前面加上感叹号“!”。

作为子系统的特殊值可以有 “all”、“system” 和 “network”：

- “all” 停止并启动所有存在的子系统。
- “sc restart system” 触发系统重启（重新引导启动）。
- 数据 “network” 只可与 “重启” 组合使用。“sc restart network” 重新初始化网络，例如：在修改了 basesys.ini 中的设置后。

示例

停止所有子系统：

$ sc stop all

只停止 HMI：

$ sc stop hmi

只启动 HMI 和 NCK：

$ sc start hmi nck

重新启动所有子系统，除了 NCK

$ sc start ! nck

2.5.14 Restore

说明

句法：	sc restore [-full|-addon|-addon+|-oem|-oem+|-user] [-force] [-nodelete] [-update] [-restart] FILENAME
可选择的名称：	---
权限等级：	user

一个用 “sc save” 创建的备份可以用 “sc restore” 再次录入到控制系统中。

作用方式

和 “sc save” 一样，只能从服务系统来进行修复或者先停止所有的子系统再进行修复。如果 “sc restore” 报告一个故障，例如：在子系统正在运行时修复异常中断，可以和使用 “sc save” 时一样用选项 “-force” 强制要求修复继续进行。

通常情况下 “sc restore” 在修复备份之前会删除整个目标区域（完整备份时删除全部 CF 卡，用户数据备份时删除 /user 目录下内容）。如此备份文件中不包含的文件随后被删除。

存档和备份文件的大小超过 4 GB

如要使用 “sc restore” 修复由多个存档文件组成的备份，则在参数 FILENAME 中只需给定备份的第一个文件。该文件结束时会自动查找下一部份。

在查找下一个文件时起作用的是调用时使用的文件名的格式，即在调用“sc restore /tmp/backup.01of02.tgz”时，当“/tmp/backup.01of02.tgz”结束时会查找名为“/tmp/backup.02of02.tgz”的下一个文件。当调用“sc restore /tmp/backup.1of2.tgz”时，会相应地查找名为“”/tmp/backup.2of2.tgz”的文件作为备份的第二部分。

选项

如无其它选项，系统会默认存档是一个完整备份文件，进行完全复制。因此全部文件的状态在 “restore” 后与备份时相同。

- 此外，选项 -full 会强制要求在 CF 卡上重新创建分区和文件系统。但该选项只能在服务系统上操作。当分区表和/或文件系统不存在和损坏时，“-full” 是必要的。
- 如果虽然需要恢复备份文件，但是不想丢失在此期间所创建的文件，则可以用选项 “-nodelete” 来阻止删除文件。“-nodelete” 不可以和 “-full” 一起执行，因为在重新创建文件系统时通常要删除所有数据。
- 使用选项 “-addon, -oem, -user, -addon+ ” 和 “ -oem+” 也可以只打开部分存档。
- 选项 “ -update” 用于录入由西门子提供的更新软件。使用 “-update” 时还可以使用选项 “-restart”。如果在录入控制系统中的存档时，自行改动了某个系统文件，则必须进行重启或重新初始化。此时，“sc restore” 不会在结束时输出相应信息。利用选项 “-restart” 可以同时执行要求的动作。

2.5.15 Save

说明

句法：	sc save [-full|-addon|-addon+|-oem|-oem+|-user] [-force] [-update] FILENAME
可选择的名称：	backup
权限等级：	user

操作 “save” 或者 “backup” 可将 CF 卡备份在 FILENAME 中，用于指定保存路径和文件名。如由服务系统出发使用该动作，则备份包含基础控制系统的文件，不包含服务系统的文件。

示例：

```
sc save -user /tmp/drv01/backup.tgz
```

选项

下列选项用于选择哪些文件系统目录应当写入存档：

-full:	完全备份（默认设置）：控制系统的所有文件，连同 Boot Loader 程序
-addon:	仅 /addon 目录
-oem:	仅 /oem 目录
-user:	仅 /user 目录（用户数据）
-addon+:	/addon, /oem 与 /user
-oem+:	/oem 与 /user

如果在控制系统上直接使用 “sc save” 且子系统正在运行，则可能会导致保存的文件互相间不一致，因为在备份过程中这些文件可能会有更改。所以在这种情况下，“sc save” 会以一个故障显示信息结束，报告仍有子系统在运行中。如果仍要创建备份，则可以使用选项 -force。系统会继续输出子系统信息并继续执行备份过程。

如果在服务系统上启动 “sc save”，则一定不会有控制系统的子系统在运行，所以 “-force” 是没有必要的。

存档和备份文件的大小超过 4 GB

如要将备份写入 FAT32 目标文件系统，该系统不支持超过 4 GB 的文件，备份会被分割为多个小于该限值的文件。

此时会在调用 “sc save” 时给定的目标文件名中添加说明，注明是总共多少个文件中的第几个文件：例如在调用 “sc save /tmp/backup.tgz” 时会生成文件 “/tmp/backup.01of02.tgz” 和 “/tmp/backup.02of02.tgz”。

2.5.16 save-nvram（备份数据）

描述

句法：	sc save-nvram [-drive|-plc|-tco] [DIRECTORY]
可选择的名称：	---
权限等级：	无

通过指令 “sc set-nvram” 可以备份非易失性存储器的内容 (non-volatile memory)。根据机床类型和设备，有集成驱动、PLC 和 TCO-EEPROM 中的 NVRAM。

通过选件 -drive、-plc 和 -tco 也可选择备份单个 NVRAM 类型。默认情况下会备份所有现有类型。

内容被写入名为 “nvram-TYP.bin” 的文件中，同时 TYP 由 “drive”、“plc” 等代替。如未指定目录，文件则创建在当前目录中。

2.5.17 set-svc-route（网关）

描述

句法：	sc set-svc-route GATEWAY DESTNET
可选择的名称：	---
权限等级：	无

通过指令 “sc set-svc-route” 可输入一个特殊网络路由器。使用范围须达到通过调制解调器连接的服务网络，以进行远程诊断。

参数为 GATEWAY，通过该网关可对 DESTNET 的包进行路由。指定网关时须指定其 IP 地址，且该地址必须位于**直接**连接的网络中。没有其他中间路由器的情况下网关必须可达。

注意
网络路由 适用于目标网络的 GATEWAY 线路，可按常规方式与网络掩码 ("128.128.12.0/255.255.255.0") 或 CIDR ("128.128.12.0/24") 一并指定。不允许全部或部分与其中一个可直接连接的 NCU 网络重叠。

2.5.18 Show

说明

在 “sc show” 的指令中概括了系统状态的各种显示。

show ip

句法： sc show ip [-INTERFACE]

可选择的名称： ----

权限等级： 无

该命令显示网络接口的 IP 地址数据。也可以给定一个特定的接口。如果没有给定接口，则会显示所有存在接口的数据，并显示缺省网关。

示例：

```
sc show ip
X120 (system network, eth0):
   configured: (default)
   current : IP=192.168.214.1 Netmask=255.255.255.0
MAC=08:00:06:73:55:fd
   DNS Name  : ncu1.local
   Nameserver: 127.0.0.1
   DNS Suffix: local
   DHCP      : synced server, prio=high, active
   Statistics: RX=0.0MB (0.00% errors), TX=0.2MB (0.00% errors)
X130 (company network, eth1):
   configured: DHCP
   current   : IP=111.222.333.64 Netmask=255.255.248.0
MAC=08:00:06:73:55:fe
   DNS Name  : name.test.siemens.de
   Nameserver: 111.222.333.12 111.222.333.13 111.222.333.14
   DNS Suffix: test.siemens.de
   DHCP      : client (server: 111.222.333.221)
   Statistics: RX=1.2MB (0.00% errors), TX=0.0MB (0.00% errors)
X127 (engineering network, ibn0):
   current   : IP=192.168.215.1 Netmask=255.255.255.224
MAC=08:00:06:73:55:ff
   DNS Name : ncu-ibn
   DHCP      : server
   Statistics: RX=0.0MB (0.00% errors), TX=0.0MB (0.00% errors)
Default gateway: 111.222.333.1 (via eth1)
Used nameserver: 127.0.0.1
Used DNS suffix: test.siemens.de local
```

对于每个接口都会显示下列数据：

- **名称：**连接插口的名称 (X1_ _)，括号内是操作系统中所用的名称（ethN 或者 ibnN）。
- **“已配置的”：**在 basesys.ini 文件中配置的 IP 地址（用于 X130 的变量外部 IP/ 外部网络掩码，用于 X120 的内部 IP/内部网络掩码），或者“（缺省）”（如果在 basesys.ini 文件中没有配置地址），或者 “DHCP”（如果地址要通过 DHCP 获取）。
- **"current":**当前设置的 IP 地址以及网络掩码和接口的 MAC 地址
- **“DNS 名称”：**在当前 IP 地址上查找 DNS 反转结果。
- **“名称服务器”：**此处可以输出一个和该接口相关的 DNS 服务器的列表（例如：在该接口上接收 DHCP）。
- **“DNS 后缀”：**DNS 查找后缀，与该接口相关。
- **“DHCP”：**此处显示该接口上是否有 DHCP 客户端或者服务器在运行。如果有客户端，则显示 IP 地址所在的服务器。X120 上的 DHCP 服务器可以与其它工程网络同步运行。然后显示它是否是激活的服务器或者处于待机状态。
- **“统计”：**通过该接口接收或者发送的数据总量以及出错数据包的百分数程序段。

显示所有接口时，还另外输出缺省网关，即一个路由器的地址，所有不能直接通过本地接口到达其目标的数据包都发送到该路由器。

缺省网关是跨接口的数据，且仅存在一个。而且还有一个通过缺省网关必须达到的接口，显示在地址后的括号中。

show drives

句法： sc show drives SERVER

权限等级： 无

“sc show drives” 显示可由特定服务器使用的远程文件系统。服务器名称 SERVER 可以是 NFS 服务器或者 TCU（代表连接到 TCU 的 USB 设备）。允许的服务器名称的详细信息、SMB 时经常需要的某个用户名说明可以参见 “sc connect” 指令的说明。

示例

符号：

```
sc show drives someuser/somedomain@somepc # Windows-Server
Password: *******
//somepc/C$
```

```
//somepc/D$
//somepc/images
sc show drives someserver # NFS-Server
someserver:/export/home1
someserver:/export/home2
sc show drives TCU # TCU USB-Medien
TCU1:/dev0-0
TCU2:/dev0-0
```

各个服务器可使用的远程文件系统会分别列出，写法和 “sc connect” 的要求一致。

- 使用 NFS 文件系统时，服务器名称后面有冒号，之后是输出路径。
- TCU 上的 USB 存储器是 NFS 的特殊情况，和 NFS 文件系统一样进行标注。但给出的路径在 TCU 上实际是不存在的，而是由 NFS 服务器在那里转换至 USB。

show net

句法：	sc show net [-xml] [-hw|-tco|-sw|-swfull|-loc|-panel|-dhcp|-switch|-all] [HOSTS...]
权限等级：	无

指令 “sc show net” 可以显示工程网络中的设备，并给出这些设备的详细信息。

这两个功能都需要借助 SNMP (Simple Network Management Protocol) 实现，从而只查找出支持 SNMP 的设备。在基于 Linux 的设备上具有从版本 2.6 起的 SNMP；在其他设备上如 MCP 则取决于使用的软件版本。这是一种简单的调用方式，无需使用其他选项即可输出一张查找到的设备的列表，列表中给出了相应的 IP 地址、DNS 名称（能够识别时）以及一些简单说明（组件名称）。

选项

通过各个相互组合的开关可以输出查找出的设备的详细信息。

- 出于屏幕显示的考虑采用了 “sc show net” 的输出格式，它不带其他选项。而可选的 XML 格式更适合于自动化的后续处理，通过选项” -xml”可以选择此格式（该选项必须在第一位！）。
- 如果在指令行的此开关后没有其他指令，“sc show net” 会在整个工程网络中通过广播查找支持 SNMP 的设备。搜索过程一般会占用一些时间，而且在大型的设备工程中会输出大量信息。因此，为简化信息的显示，可以同时给定一张 IP 地址或 (DNS-)名称的列表。从而可以限制搜索过程，只查找指定的设备。

-hw	硬件信息，如：MLFB、序列号、硬件 ID、硬件更新、SRM 版本、CF 卡的制造商和序列号。某些设备会具有两个此类硬件信息块，其中第二个针对集成组件。 例如，NCU 中的 PLC 以及 TCU 上 OP 中的 TACO。
-tco	运行状态数据：接通次数计数器、运行时间计数器、CF 卡上写入的数据量、超出最高温度的次数。
-loc	地点数据：设备的地点、功能和联系方式（如果其中配置了这些数据）。
-sw	安装的软件的简明说明，仅整个系统的版本；只安装了基础系统时，显示基础系统的版本。
-swfull	显示所有安装的软件组件。各个组件按照结构尺寸分级排列；显示每个组件的版本、内部版本（不一致时）、给定版本（不一致时）、版本的详细说明和安装路径。
-panel:	显示面板专有的数据，即：显示屏的尺寸和颜色深度、MCP 索引、TCU 索引、DCK 索引和 EKS 索引、当前显示的 VNC 服务器。
-dhcp	显示正在同步的 DHCP 服务器的数据：运行状态（关闭/待机/激活）、优先级、Lease 文件和 TCU 数据的版本、动态分配的 IP 地址的区域。
-switch	显示设备上旋转开关的位置，如：NCU 的 NC 开关或 PLC 开关或者 HT 8 的 DIPFIX。
-all	相当于所有上述开关的功能。

说明

此处读出的数据和信息会自动录入 EUNA 文件(End User Notification Administration)并保存在相应的设备上。

2.5.19 Start, Stop

说明

句法：	sc start all|system|SUBSYSTEM...
	sc stop all|system|SUBSYSTEM...
可选择的名称：	---
权限等级：	service

这两个指令会启动或者停止单个或所有子系统。由于这两个动作会对运行性能产生影响，所以只能在控制系统上直接使用，而不能在维修系统中使用（例外：“sc stop system”）。在那里基础控制系统的子系统不运行，也不可以对其进行控制。

说明

仅 “all” 和 “system” 起作用，而不是任意的子系统。这两个动作只能实现部分功能！

子系统名称

作为子系统的名称，可以使用 “hmi”、“nck” 或者其他任意名称。也可以连续指定多个按顺序启动或者停止的子系统。

作为子系统的特殊值可以有 “all” 和 “system”：

- “all” 启动或者停止所有存在的子系统。
- “sc stop system” 同样停止所有子系统，但也停止带控制系统的基础系统。
- “sc start system” 和 “sc start all” 是相同的，主要是出于对称的原因设置的。

2.5.20 usb（连接设备）

描述

前提条件：

- 硬件：30.3
- 软件：如要使用 USB 设备，则需额外的驱动器或软件且必须安装在 NCU 上。

句法：	sc usb attach|export|unexport SERVER BUSID
	sc usb detach PORT
	sc usb showdevs SERVER
	sc usb status
可选择的名称：	---
权限等级：	无

通过指令 “sc usb” 可使连接在 TCU 上的任意 USB 设备（集线器和存储媒介除外）也可在 NCU 上使用。

对于参数 SERVER（应使用 TCU 及其设备），可指定符号主机名称（如 TCU1）或 IP 地址（如 192.168.214.10）。

示例

列出 TCU1 上的设备：

```
# sc usb showdevs TCU1
                0bf8:100c
```

导出 TCU1 的设备：

```
# sc usb export TCU1 1-2.1
```

列举时，该设备标记为已导出：

```
# sc usb showdevs TCU1
                0bf8:100c
                [exported]
```

将设备连接到 NCU 上：

```
# sc usb attach TCU1 1-2.1
```

设备以 NCU 本地状态列出：

```
# sc usb status
Port 00: Port in Use
TCU1:1-2.1
```

设备借助端口号与 “sc usb status” 隔开：

```
# sc usb detach 00
```

不导出 TCU 上的设备：
```
# sc usb unexport TCU1 1-2.1
```

设备不再标记为已导出：
```
# sc usb showdevs TCU1
                0bf8:100c
```

2.5.21 vncpwd（密码）

描述

通过指令 “sc vncpwd” 执行下列动作：

- sc vncpwd set，以给定密码。
- sc vncpwd reset，以删除密码。

句法：	sc vncpwd set companynetwork|systemnetwork
	sc vncpwd reset companynetwork|systemnetwork
可选择的名称：	---
权限等级：	无

通过指令 “"sc vncpwd set” 可在服务器 (NCU) 上通过公司网络或设备网络为 VNC 连接设置一个密码。给定密码后，只有在密码输入正确的情况下才能建立与 VNC 客户端（TCU、Ultr@VNC 等）的连接。

指令 `sc vncpwd set` 始终用于输入密码。

或者例如：sc vncpwd set systemnetwork secreT7+

指令 `sc vncpwd reset` 用于删除指定网络的密码。这样便可在网络中建立连接，无需认证。

说明

网络中的密码

为了能在 TCU（客户端）的网络（设备网络或公司网络）中通过 VNC 连接访问网络中任意一个服务器 (NCU)，所有服务器都应使用相同的密码。

诊断与维修 3

3.1 生成服务系统

概述

执行以下步骤**备份**用户数据或整张 CF 卡：

- 创建服务系统。
- 备份授权
- Backup：将数据备份至 USB 存储介质或网络驱动器。
- 系统诊断
  - 输出网络设置
  - 修改网络设置
  - 启动 VNC 浏览器

执行以下步骤**恢复**用户数据或整张 CF 卡：

- Restore/Recover：从 USB 存储介质或网络驱动器上恢复数据。
- 执行软件更新。
- 执行固件更新。

3.1.1 这样创建 NCU 的服务系统

使用目的

可以在 USB 存储器上创建一个维修情况下使用的便携式“Emergency Boot System”（EBS:紧急启动系统）。 这样可通过服务系统开始 NCU 的引导启动，从而可在一个 Service Shell 下执行不同的维护任务，例如：数据备份或者更新。

在服务系统上划分为两个分区：

- 一个 Linux 分区，在 Windows 下不可见。
- 一个 FAT 分区，用于保存备份文件或软件更新程序。
  FAT 分区既可通过 Linux，也可通过 Windows 系统读取和写入。 在 Linux 下的 Command Shell 中可通过路径 /data 访问 FAT 分区。

供货范围

在光盘上随附了以下文件，用于在 USB 存储器上创建服务系统：

- 可执行文件 installdisk.exe
- USB 设备用镜像文件
- 含最新信息的文件 siemensd.txt / siemense.txt

建议：

最好使用 SIMATIC PC USB 设备。

说明

创建服务系统需要有管理员权限。

创建服务系统将会删除 USB 存储器上所有已经存在的数据。

使用 USB 2.0 进行传输是最佳的；而使用 USB 1.1 传输到 USB 存储器会比使用 USB 2.0 的时间长。

步骤

在 USB 存储器上创建一个服务系统步骤如下：

1. 将服务系统复制到 PG/PC 的本地硬盘上。
2. 将 USB 存储器连接到 PG/PC 的 USB 接口上。
3. 可以在 Windows 浏览器窗口中看到给 USB 存储器分配了哪一个驱动器字符，例如 H：
4. 打开一个 DOS Shell，并且切换到保存服务系统文件的目录。
5. 在 DOS Shell 中输入下列命令：
   ```
   installdisk --verbose --blocksize 1m linuxbase.img h:
   ```

结果：该镜像被传输至 USB 存储器。

3.1.2 这样操作维修系统

连接服务系统

步骤：

1. 将 USB 存储器连接到 NCU 的 USB 接口 X125 或 X135 上。
2. 重新启动系统。
   或
3. 按下"Reset"键。
   在首次通过 USB 存储器启动 NCU 时，服务系统被拆包：此后 USB 存储器才能作为服务系统使用，且整个存储容量都可用。

操作服务系统

用于在服务系统中导航的按键和软键：

软键	OP 上的按键	外部键盘	说明
↓	HSK1	<F1>	光标，下一行
↑	HSK2	<F2>	光标，上一行
Page↓	HSK3	<F3>	光标，下一页
Page↑	HSK4	<F4>	光标，上一页
Char↓	HSK5	<F5>	插入文本或数字
Char↑	HSK6	<F6>	插入文本或数字
Cancel	VSK7	<Shift> + <F7>	取消/返回
Ok	VSK8	<Shift> + <F8>	OK/确认
---	NEXT WINDOW	Pos1	光标，至最上行
---	END	Ende	光标，至最下行

说明

触摸屏上的操作

维修系统支持触摸操作输入栏是可触控的且内置键盘会自动显示。

例如：SINUMERIK OP 019 black、SIMATIC HMI Industrial Thin Client

3.1.3 诊断功能

“服务”菜单的功能

服务系统上提供以下用于服务和诊断的功能：

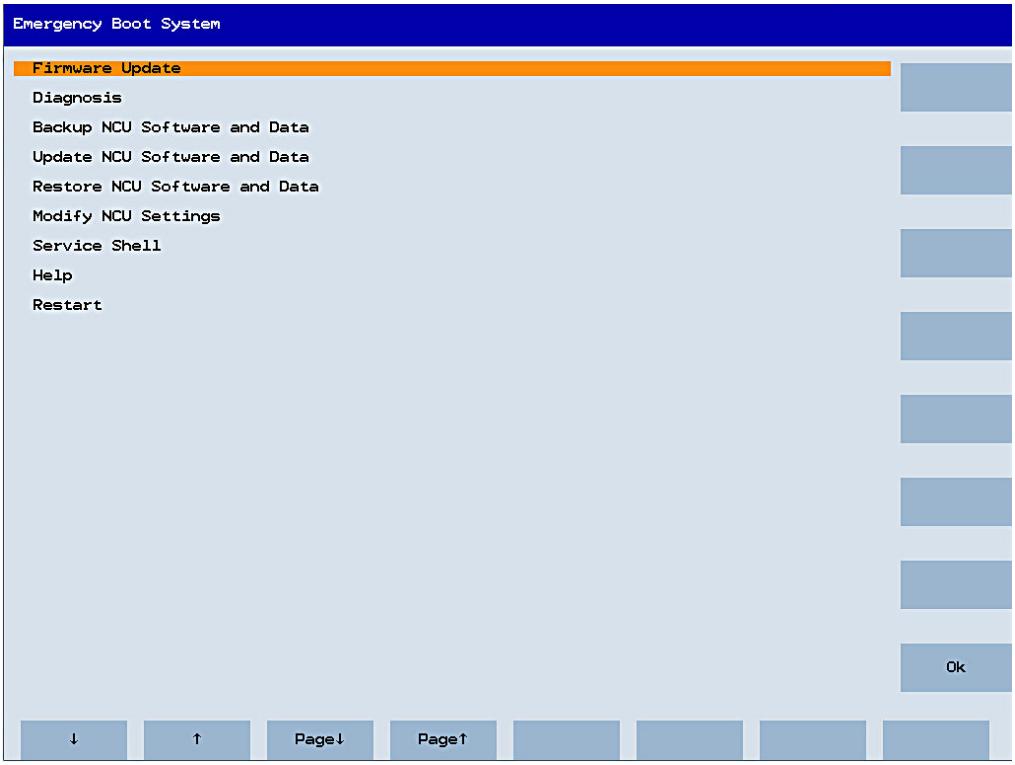


图 3-1 “服务”菜单

主菜单概览：

主菜单	说明	参考
Firmware Update	执行 BIOS 更新： • PLC BIOS • FPGA • CBE30 引导加载程序	这样执行固件更新 (页 92)
VNC Viewer	启动 VNC 浏览器	调用 VNC 浏览器 (页 102)
Diagnosis	诊断功能，用于显示网络连接数据和系统 CF 卡上的数据。	执行系统诊断 (页 96)
Backup NCU Software and Data	将系统数据和用户数据备份到 USB 设备或网络驱动器上。	通过“服务”菜单备份数据 (页 80)

主菜单	说明	参考
Update NCU Software and Data	从 USB 设备或网络驱动器升级系统。	这样执行软件更新 (页 94)
Restore NCU Software and Data	从 USB 存储介质或网络驱动器恢复系统数据和用户数据。	通过“服务”菜单备份数据 (页 80)
Modify NCU Settings	• 修改网络设置 • 激活/取消 HMI • 恢复以下组件的出厂设置： – HMI (SINUMERIK Operate) – NC – PLC – 驱动装置	这样修改系统设置 (页 98)
Service Shell	打开 Service Shell ，输入服务命令。	通过 Service Shell 备份数据 (页 88) 服务命令 (页 42)
Help	用于操作维修菜单的帮助文本（外部键盘没有功能键时）：	
	ALT 1 ... 8	用于操作水平软键的功能键 <F1> ... <F8>
	SHIFT ALT 1 ... 8	用于操作垂直软键的组合键：VSK1 ... VSK8
Restart	重启系统	

3.1.4 这样连接网络驱动器

连接网络驱动器

1. 选择“Connect to network drive”。
2. 输入网络驱动器的完整路径名。

3. 使用具有访问权限的用户名登录此驱动器。
   请注意给出示例中的句法：

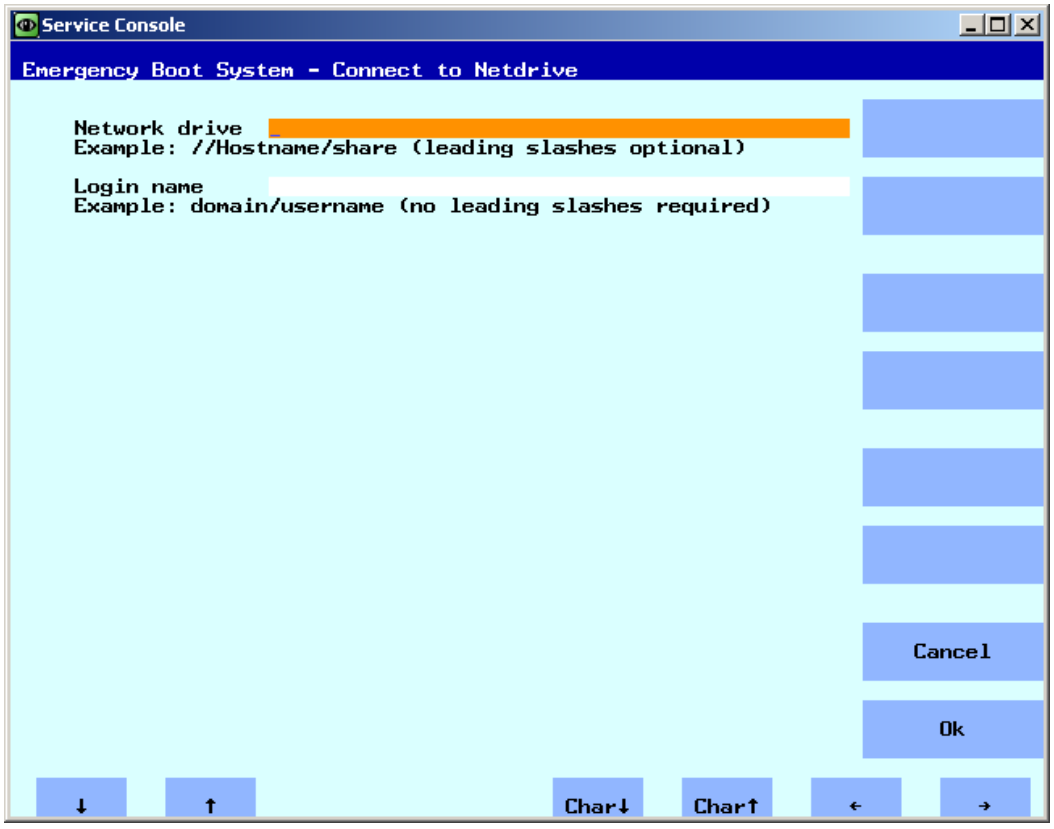


图 3-2　连接网络驱动器

4. 按下“Ok”确认。 之后会被要求输入密码。

3.2 备份授权

授权密钥的效力

授权密钥与 CF 卡的序列号相对应。 如果在 SINUMERIK 840D sl 上更换了 CF 卡，则授权密钥失效，设备不再处于运行就绪状态。

该情况发生在系统 CF 卡出现硬件损坏时。

应用场合

备份 CF 卡的授权密钥，用于 USB 存储介质或网络驱动器上的以下情况：

- 软件更新，无授权密钥
- 系统软件故障，无硬件损坏

说明

CF 卡备件

为了在更换系统 CF 卡后获取有效的授权密钥，需要以下数据： **损坏的和新的 CF 卡的序列号**。

3.2.1 这样备份授权密钥

备份授权密钥至 USB 存储介质

步骤：

1. 在主菜单中选择菜单项“Backup NCU Software and Data”。
   显示以下菜单：

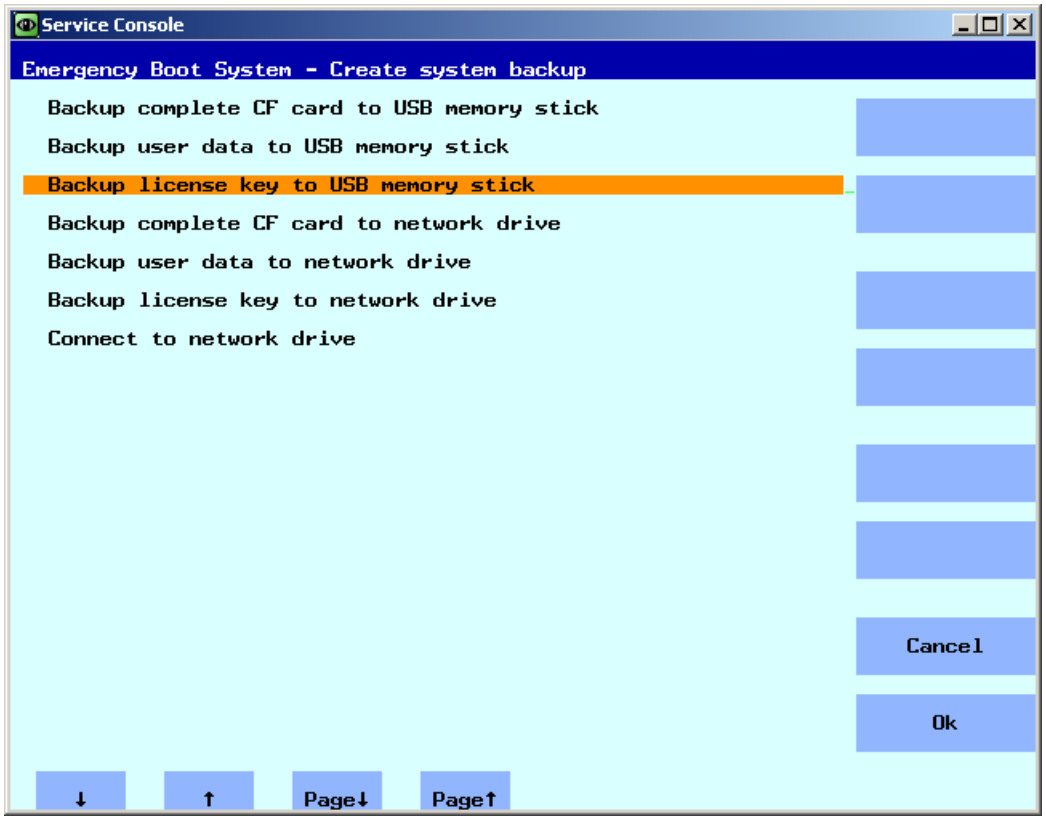


图 3-3 备份授权

2. 选择菜单项“Backup license key to USB memory stick”。
   显示以下菜单：

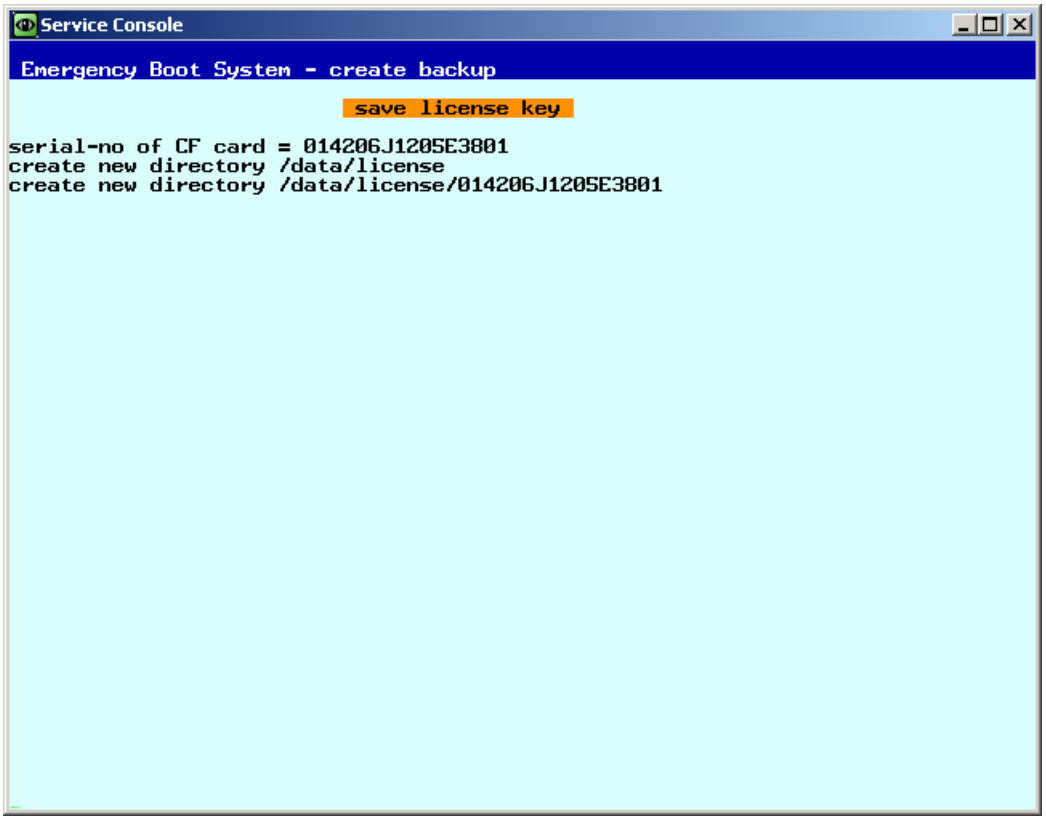


图 3-4　Save license key

授权密钥和 CF 卡序列号一同备份至指定的目录下。

另见

“Backup license key to network drive”的步骤与此相似。 连接网络驱动器的信息另见：

这样连接网络驱动器 (页 75)

3.3 通过维修菜单备份数据

3.3.1 这样备份用户数据

备份和恢复用户数据

步骤：

1. 在主菜单中选择菜单项“Backup NCU Software and Data”。
2. 选择“Backup user data to USB memory stick”，并按下“Ok”确认。

3. 输入完整的文件名（如示例所示）：

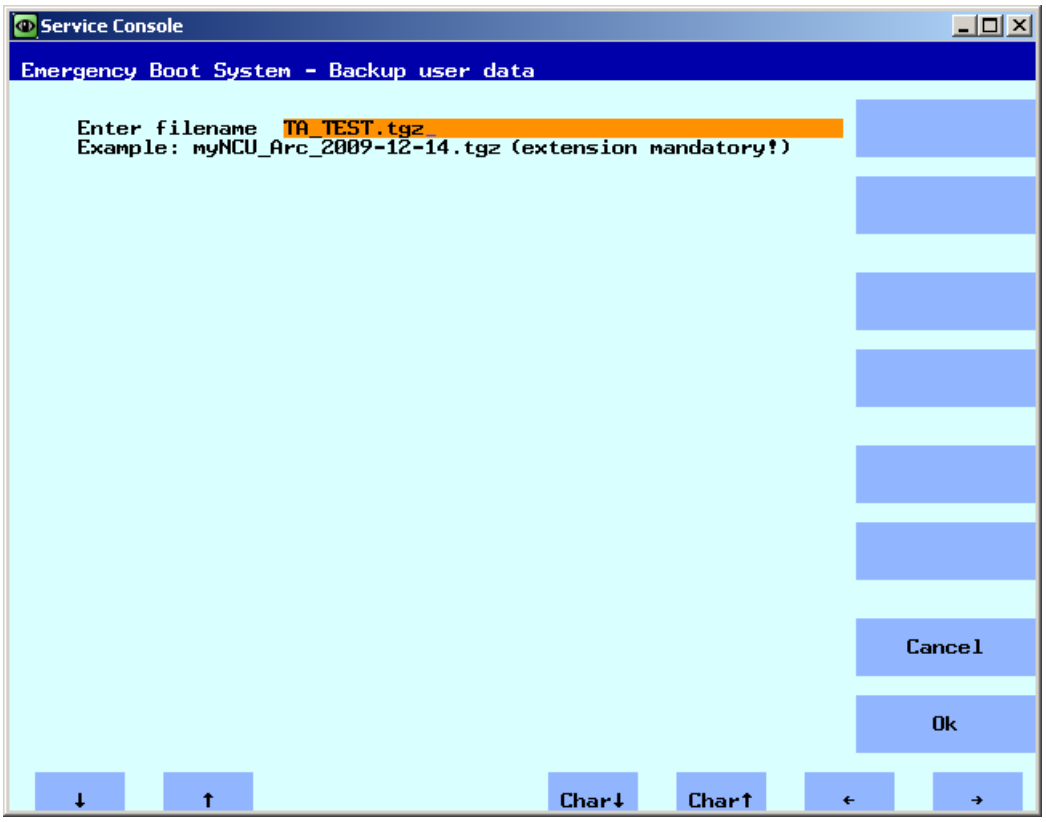


图 3-5　输入文件名

4. 按下"Ok"确认。在此步骤成功执行后，会输出以下信息：

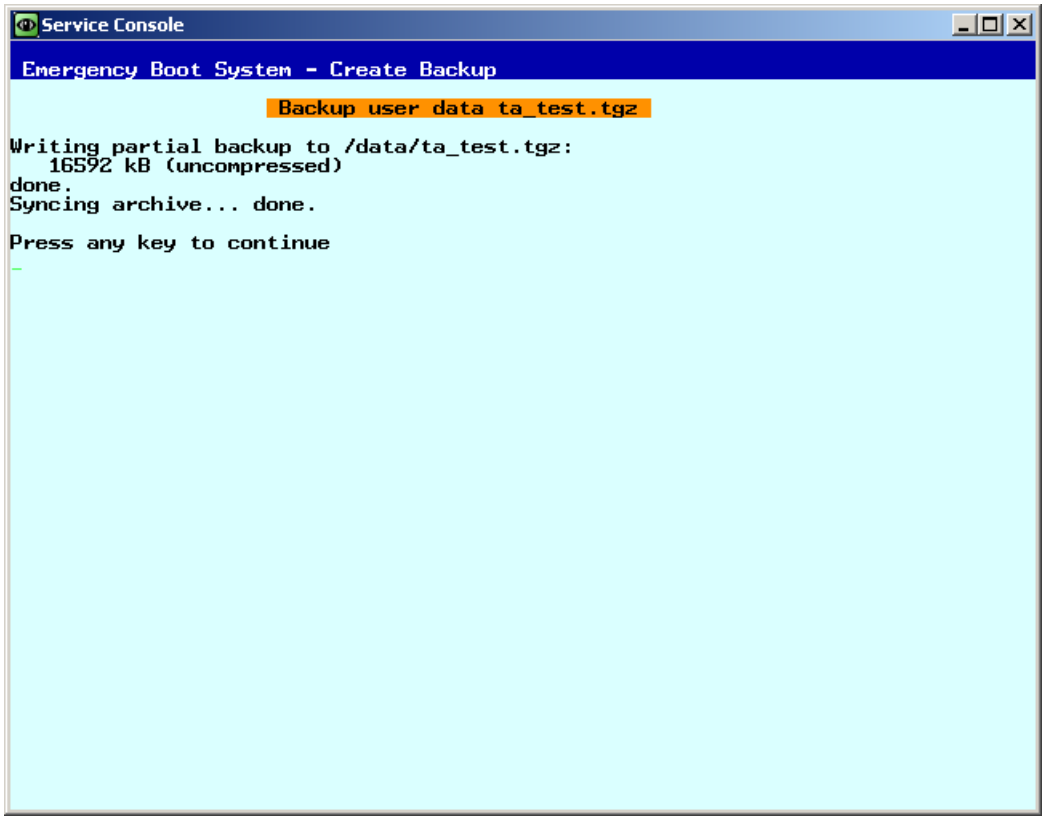


图 3-6　步骤完成

说明

超过 4 GB 的备份文件

备份文件大小超过 4 GB 时，服务系统 FAT 分区（8 GB）的容量将会不足。这样将会分割该大小的备份文件（另见：sc save/sc restore）或者您可将其保存在网络驱动器上。

另见

“Backup user data to network drive”的步骤与此相似。此外还应连接网络驱动器：

这样连接网络驱动器 (页 75)

3.3.2 这样恢复用户数据

恢复用户数据

步骤：

1. 在主菜单中选择菜单项“Restore NCU Software and Data”。
   显示以下菜单：

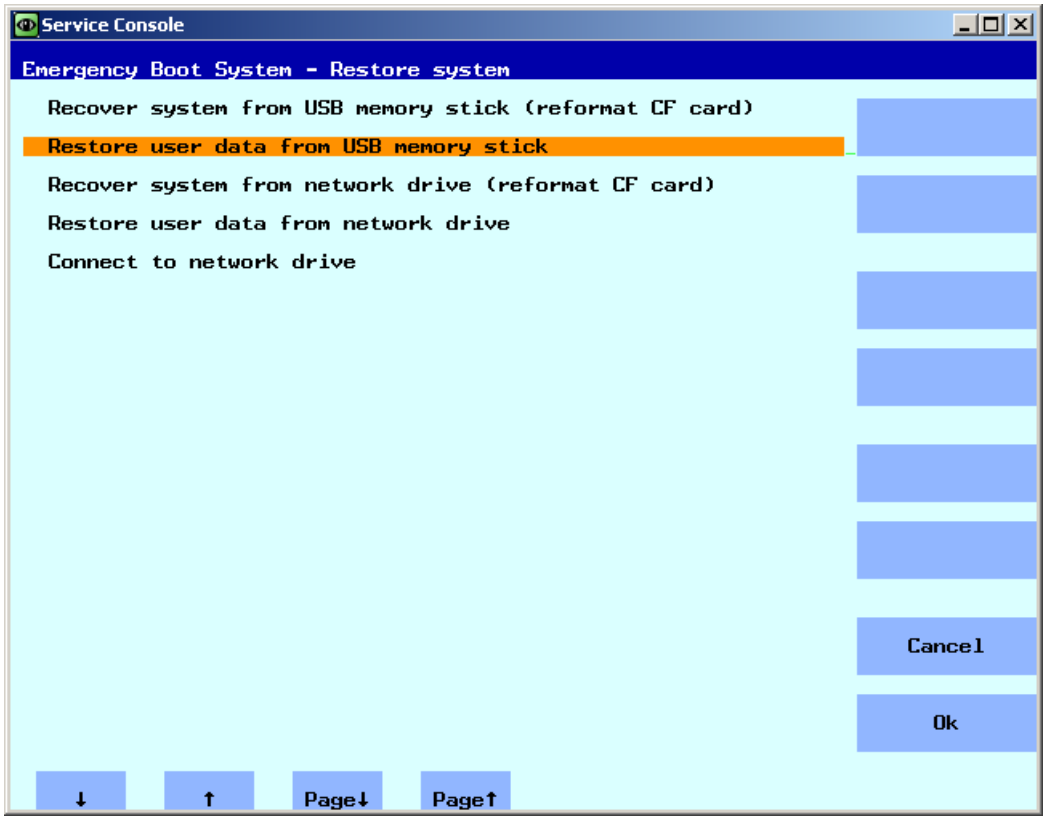


图 3-7 恢复用户数据

2. 选择“Restore user data from USB memory stick”，并按下“Ok”。
   显示 USB 存储媒介上可用的 tgz 文件的列表。
3. 选择相应的 tgz 文件，并按下“Ok”确认。
   在此步骤成功执行后，会输出一条信息。

另见

“Restore user data from network drive”的步骤与此相似。此外还应连接网络驱动器：

这样连接网络驱动器 (页 75)

3.3.3 这样创建 CF 卡的完整备份

创建完整备份

步骤：

1. 在主菜单中选择菜单项“Backup NCU Software and Data”。
   显示以下菜单：

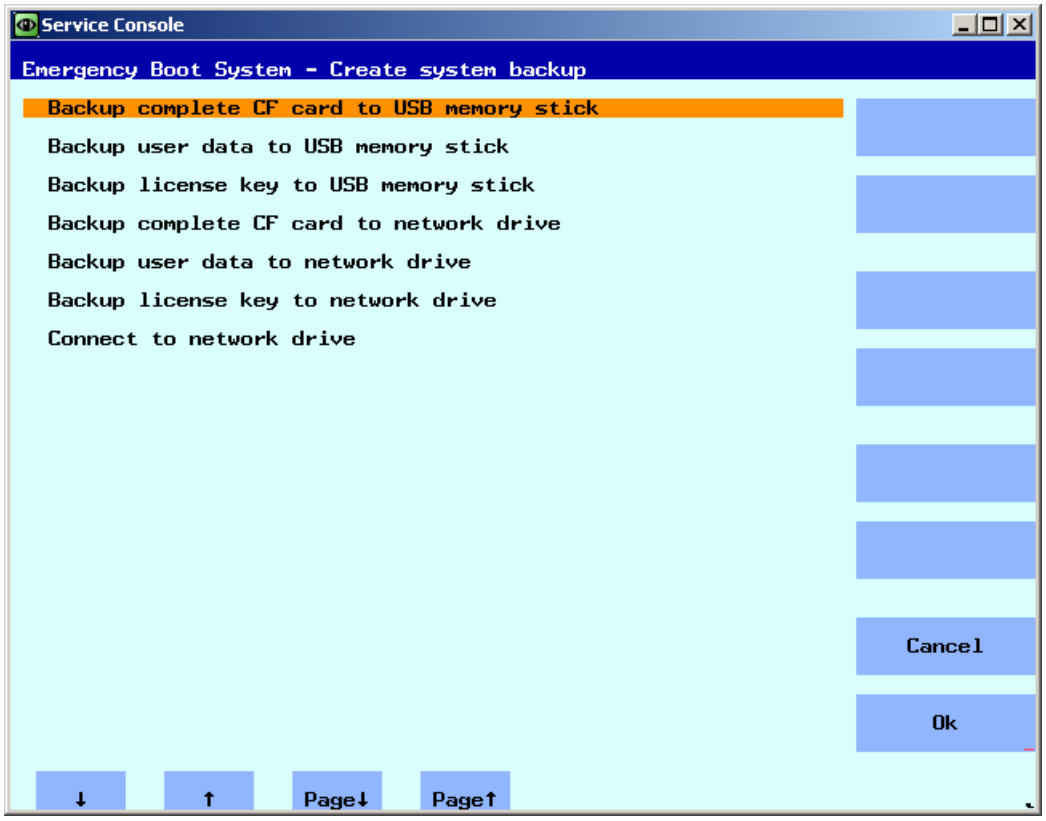


图 3-8 备份

2. 选择“ackup complete CF card to USB memory stick”，并按下“Ok”确认。

3. 输入完整的文件名（如示例所示）。
4. 按下"OK"确认。该过程可能要持续几分钟。
   过程显示如下：

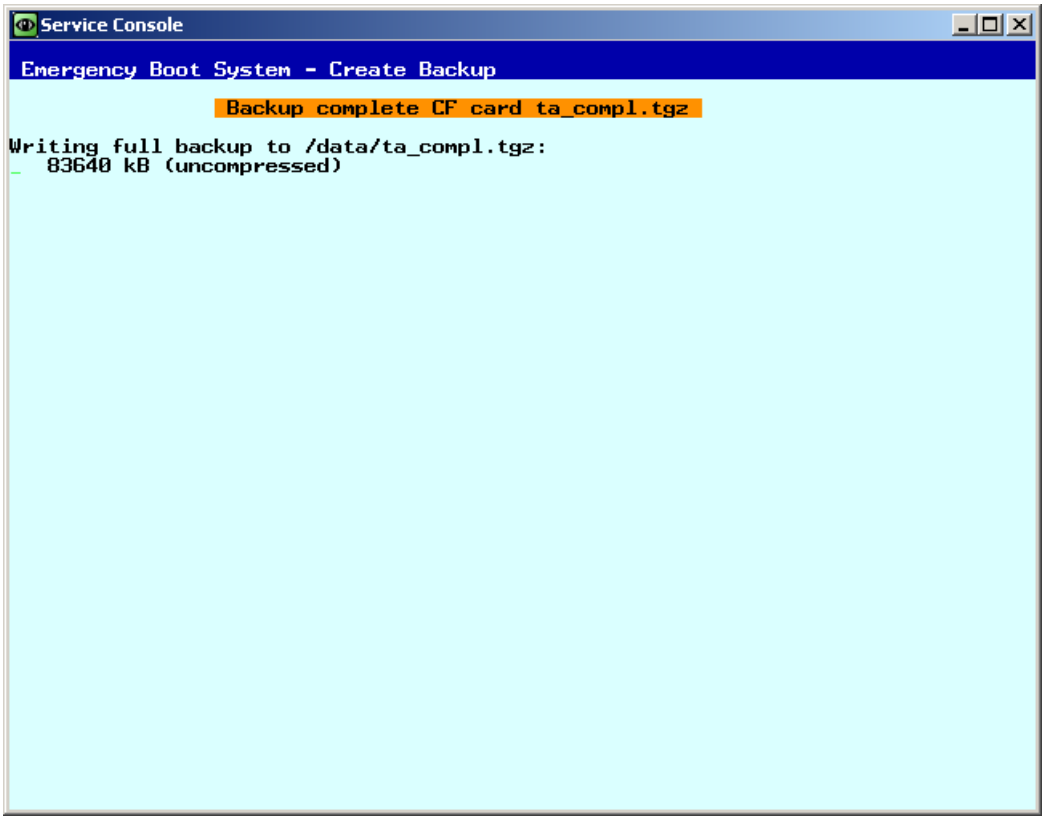


图 3-9　备份中...

在此步骤成功执行后，会输出一条信息。

另见

"Backup complete CF card to network drive"的步骤与此相似。此外还应连接网络驱动器：

这样连接网络驱动器 (页 75)

3.3.4 这样安装完整的系统备份

安装完整备份

通过此菜单项从服务系统或网络驱动器在 CF 卡安装完整数据备份，完整备份既包含系统软件也包含用户数据。

注意
格式化 此步骤相当于对 CF 卡进行格式化，全部现有数据都会被覆盖。

安装完整备份

步骤：

1. 在主菜单中选择菜单项“Restore NCU Software and Data”。
   显示以下菜单：

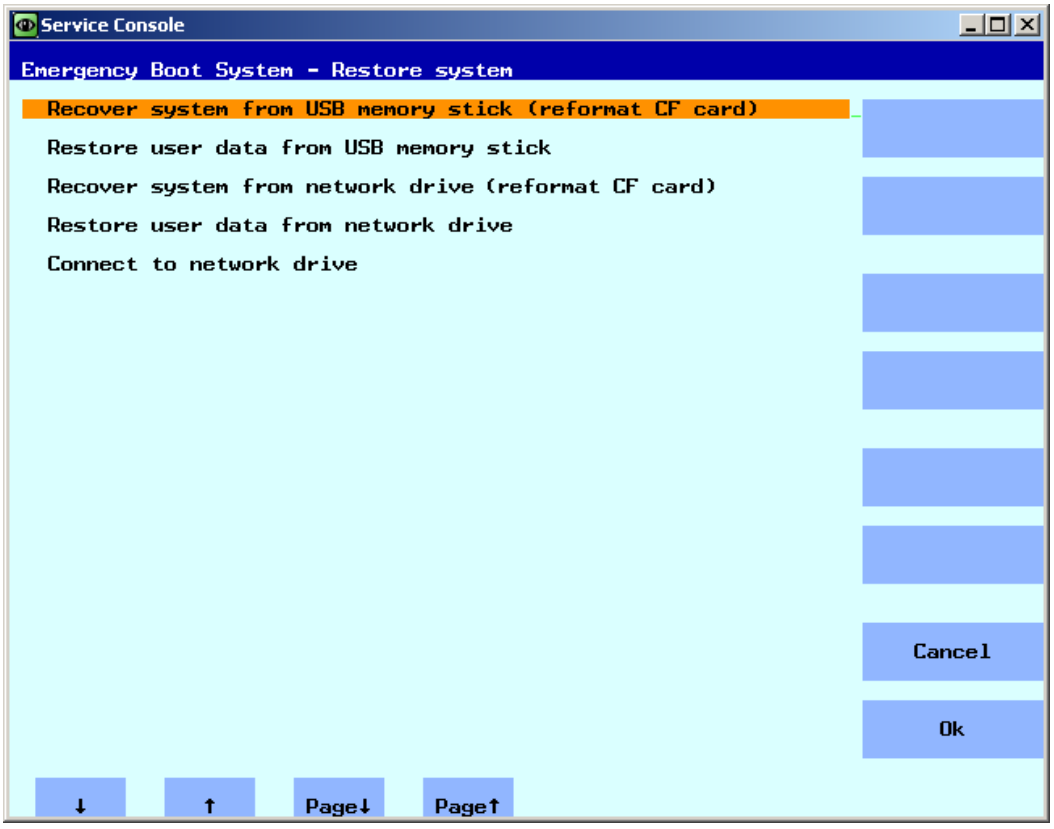


图 3-10 Restore system

2. 选择“Recover system from USB memory stick (reformat CF card)”，并按下“Ok”确认。
   显示 USB 存储介质上可用的 tgz 文件的列表。
3. 选择相应的 tgz 文件，并按下“Ok”确认。
   在此步骤成功执行后，会输出一条信息。

完成

重新启动系统后，新的数据才会生效。

另见

“Recover system from network drive (reformat CF card)”的步骤与此相似。此外还应连接网络驱动器：

这样连接网络驱动器 (页 75)

3.4 通过 Service Shell 备份数据

使用 Service Shell 备份和恢复

根据配置，采用以下方法调用 Service Shell：

(I) 带 TCU 的 NCU 配置：在 Linux 下调用 Service Shell。

(II) 带 PCU 50 或编程器（PG）的 NCU 配置：

可通过以下方式调用 Service Shell：

- 在 NCU 上的 Linux 下
- 在 PCU/PG 上通过 VNC 浏览器（设备网络或公司网络）
- 在 PCU/PG 上通过 WinSCP（设备网络或公司网络）

3.4.1 这样将数据备份至服务系统

步骤

备份完整的系统步骤如下：

1. 将该维修系统连接至 NCU 的一个 USB 接口（X125 或者 X135），并按下复位键。
   也可以关闭 NCU，连接维修系统，然后再次接通 NCU。
   结果: 从维修系统引导启动 NCU。
2. 在主菜单中选择菜单项"Service Shell"。
3. 请作为维修技术员以用户名"manufact"和密码"SUNRISE"登录（Login）。
4. 用命令"sc backup"来创建备份文件"backup01.tgz"。
   维修系统上的目录 /data 为备份文件预留。 必须给出完整的路径。
5. 可以选择 –full，备份 CF 卡上的所有数据；或者选择 –user，仅备份用户数据。
   示例： `sc backup -full /data/backup01.tgz`
   结果：
   在维修系统的 USB 存储器上的 /data 路径下生成一个完整 CF 卡的备份文件。

3.4.2 这样从服务系统恢复数据

步骤

需要恢复完整的系统，步骤如下：

1. 将该维修系统连接至 NCU 的一个 USB 接口（X125 或者 X135），并按下复位键。
   也可以关闭 NCU，连接维修系统，然后再次接通 NCU。
   结果: NCU 从维修系统引导启动，并显示主菜单。
2. 在主菜单中选择菜单项“Service Shell”。
3. 请作为维修技术员以用户名“manufact”和密码“SUNRISE”登录（Login）。
4. 用命令“sc restore”将备份文件“backup01”从维修系统复制到 NCU 的 CF 卡上。 必须给出完整的路径。
   示例： `sc restore /data/backup01.tgz`
   结果：
   在 NCU 上恢复了文件“backup01”中保存的系统状态。

说明

如果不能存取 CF 卡上的系统数据（因为 CF 卡损坏或者为空），则只能以用户“admin”和密码“SUNRISE”登录，而不能以用户“manufact”登录。

3.4.3 这样将数据备份至网络驱动器

步骤

请按照下列顺序操作：

- 连接服务系统。
- "打开“Service Shell”。
- 创建与网络驱动器的连接。
- 创建备份文件（Backup）。

情况 I： 在 Linux 下启动 Command Shell

步骤：

1. 将服务系统连接到 NCU 的 USB 接口 X125 或 X135 上。
2. 在主菜单中选择菜单项“Service Shell”。
3. 请作为服务人员以用户名“manufact”和密码“SUNRISE”登录（Login）。

4. 用指令“sc connect”连接网络驱动器：
   `sc connect //username%password@server/share /tmp/backup`
   在此输入待连接网络驱动器的用户名和密码。
5. 通过指令“sc save”创建备份文件“backup01”。
   此时可选择 –full，备份 CF 卡上的所有数据；或者选择 –user，仅备份用户数据。
   示例： `sc save -full /tmp/backup/backup01`
   结果：
   在网络驱动器的指定路径下生成整张 CF 卡的备份文件。

情况 II： 用编程器上的 WinSCP 来启动 Command Shell

步骤：

1. 启动 WinSCP，并在登录窗口中输入下列数据：
   – NCU 的 IP 地址（或者可能的主机名称）
   – 用户名“manufact”和密码“SUNRISE”。
2. 选择菜单中的“Commands” → “Open Terminal”。
3. 根据情况 I 中的步骤 4 到 6 的指令进行数据备份。

情况 II： 用编程器上的 VNC 浏览器来启动 Command Shell

步骤：

1. 启动 VNC 浏览器，并通过 IP 地址（或者必要时通过主机名）与 NCU 连接
2. 请以用户名“manufact”和密码“SUNRISE”登录。
3. 根据情况 I 步骤 3 到 5 的描述输入相应指令，以恢复数据。

3.4.4 这样从网络驱动器恢复数据

步骤

请按照下列顺序操作：

- 连接维修系统。
- "打开“Service Shell”。
- 建立与网络驱动器的连接。
- 恢复数据（还原）。

情况 I：在 Linux 下启动 Command Shell

1. 将维修系统连接到 NCU 的 USB 接口 X125 或 X135 上。
2. 在主菜单中选择菜单项“Service Shell”。
3. 请作为维修技术员以用户名“manufact”和密码“SUNRISE”登录（Login）。
4. 用指令“sc connect”连接网络驱动器：
   `sc connect //username%password@server/share /tmp/backup`
5. 在此输入待连接网络驱动器的用户名和密码。
   为了完整地恢复系统（系统数据和用户数据），请输入：`sc restore –full backup01`
   结果： 备份数据覆盖整个系统。
6. 如仅要恢复用户数据，请输入下列命令：
   `sc restore –user backup01`
   结果： 恢复用户数据。
   之后重新启动所有子系统：`sc start all`
   结果：
   在 NCU 上恢复了文件“backup01.tgz”中保存的系统状态。

情况 II：用编程器上的 WinSCP 来启动 Command Shell

步骤：

1. 启动 WinSCP，并在登录窗口中输入下列数据：
   – NCU 的 IP 地址（或者可能的主机名称）
   – 用户名“manufact”和密码“SUNRISE”。
2. 选择菜单中的“Commands” → “Open Terminal”。
3. 根据情况 I 步骤 3 到 5 的描述输入相应指令，以恢复数据。

情况 II： 用编程器上的 VNC 浏览器来启动 Command Shell

步骤：

1. 启动 VNC 浏览器，并通过 IP 地址（或者必要时通过主机名）与 NCU 连接
2. 请以用户名“manufact”和密码“SUNRISE”登录。
3. 根据情况 I 步骤 3 到 5 的描述输入相应指令，以恢复数据。

3.5 执行软件更新

3.5.1 这样执行固件更新

Firmware Update

在进行服务时，服务人员可能需要安装 BIOS 更新程序。相应的更新文件由西门子提供。可能的格式有：*.img 或*.rom 或*.bin.更新文件也可能随服务系统发货，此时可省略步骤 1。

步骤：

1. 将更新文件复制到服务系统的 FAT 分区中。
2. 将服务系统连接到 NCU 的接口 X125 或 X135 上。
3. 连接系统。
4. 在主菜单中选择菜单项“Firmware Update”。
   显示以下菜单：

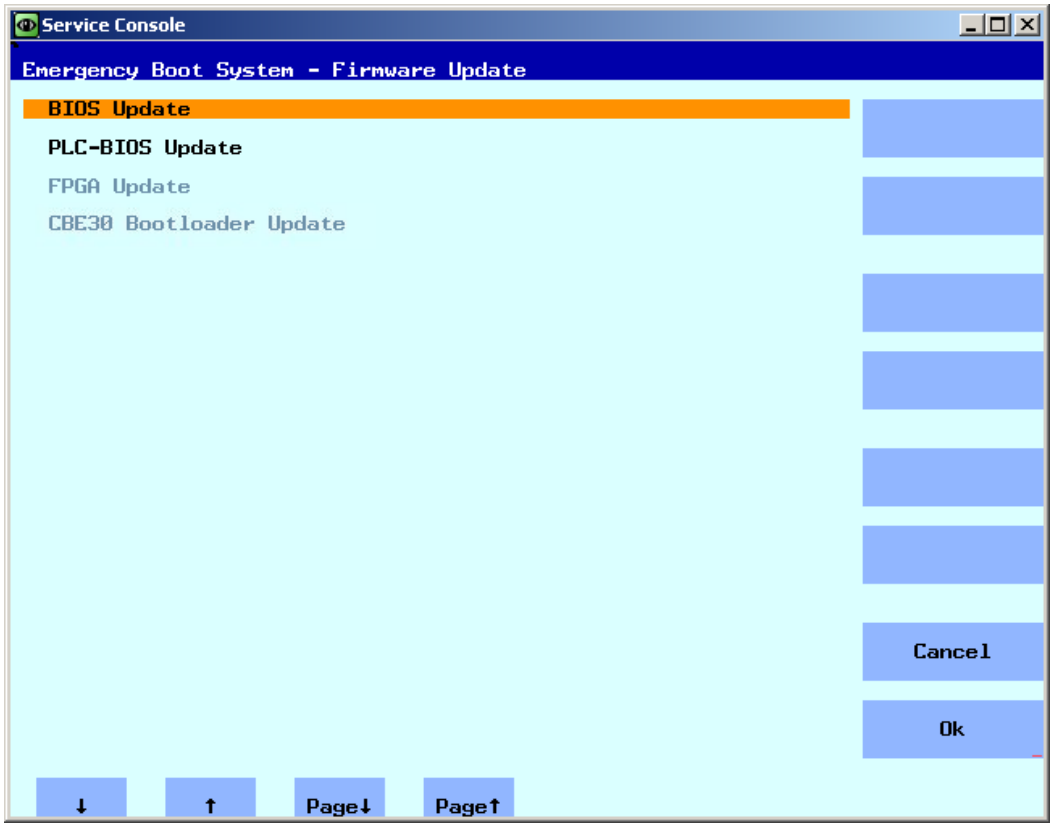


图 3-11 固件更新

5. 选择"BIOS Update"，按照后续指令操作。
输出 NCU 上安装的固件版本，以及维修系统上现有的更新文件：

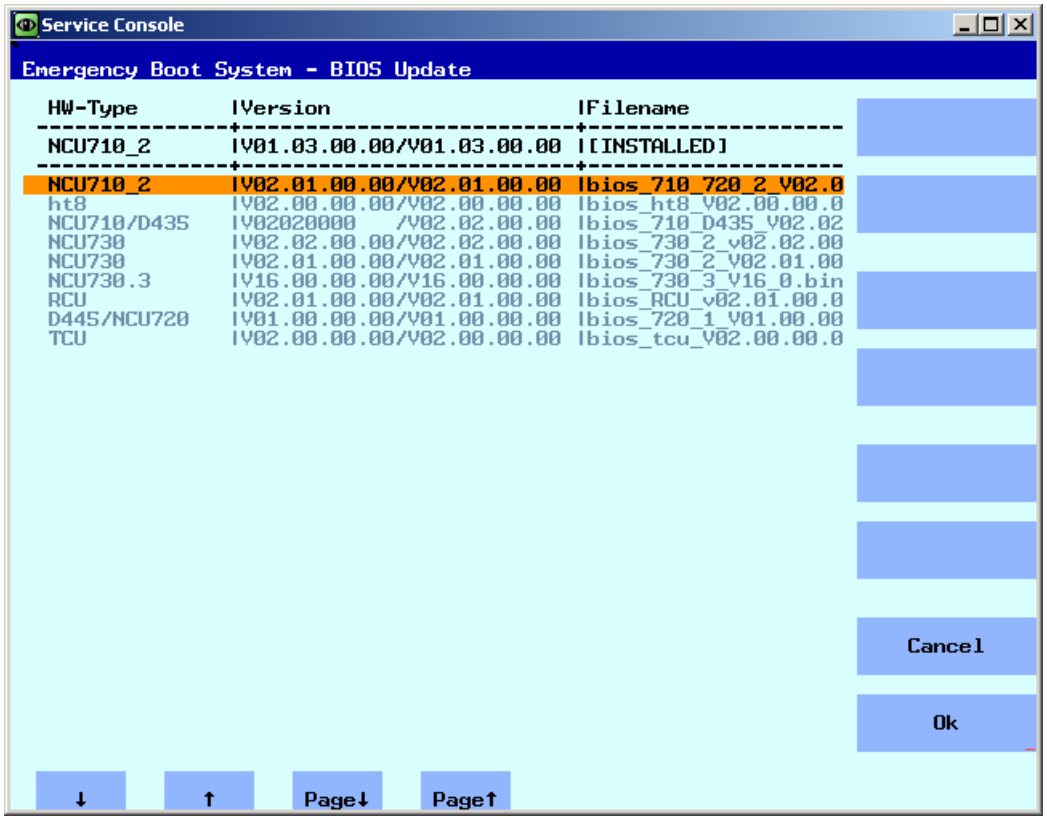


图 3-12　选择更新

6. 选择更新文件，并按下"Ok"确认。
之后需要"Restart"。

说明

恢复方案

在安装固件更新前，会在维修系统上保存安装的固件版本的备份。

由于安全原因，仅可选择与 NCU 相匹配的文件。

如果在维修系统的 FAT 分区中创建了一个名为 "BIOS" 的目录，则会在此目录中找到固件更新且已安装的固件版本也会保存在此目录中。

更新其他组件

在更新其他组件时注意：

- PLC-BIOS Update 更新时的步骤与固件更新的步骤相似。
- FPGA Update：针对 NCU7x0.3
- CBE30 Bootloader Update：针对 CBE30 和 CBE30-2（选件模块）

3.5.2 这样执行软件更新

Update / Recover

可进行以下操作：

- 通过维修系统或网络驱动器执行软件更新。
- 通过维修系统或网络驱动器完整恢复含系统软件的 CF 卡（Recover system ...）。

执行软件更新

步骤：

1. 在主菜单中选择菜单项“Update NCU Software and Data”。
   显示以下菜单：

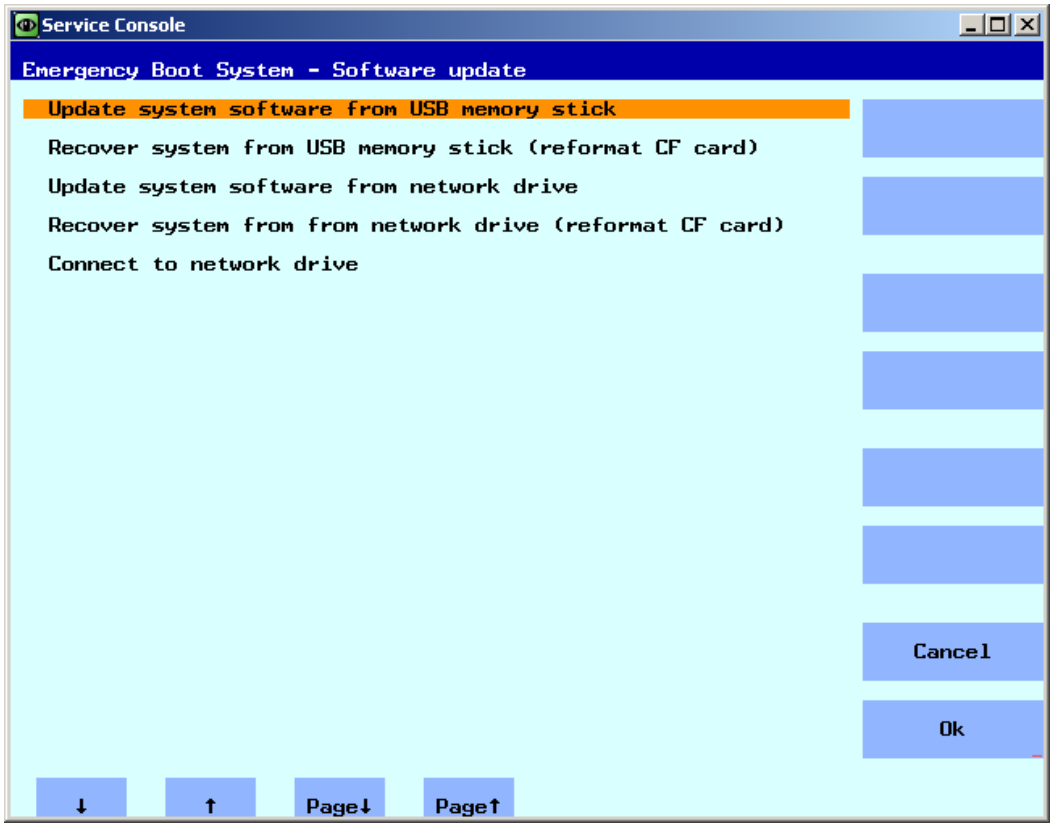


图 3-13 软件更新

2. 选择菜单项“Update system software from USB memory stick”。
   显示 USB 存储介质上可用的 tgz 文件的列表。

3. 为了检查 tgz 文件的软件版本，先选中 tgz 文件，然后按下“Show version”。
   会输出下列信息：

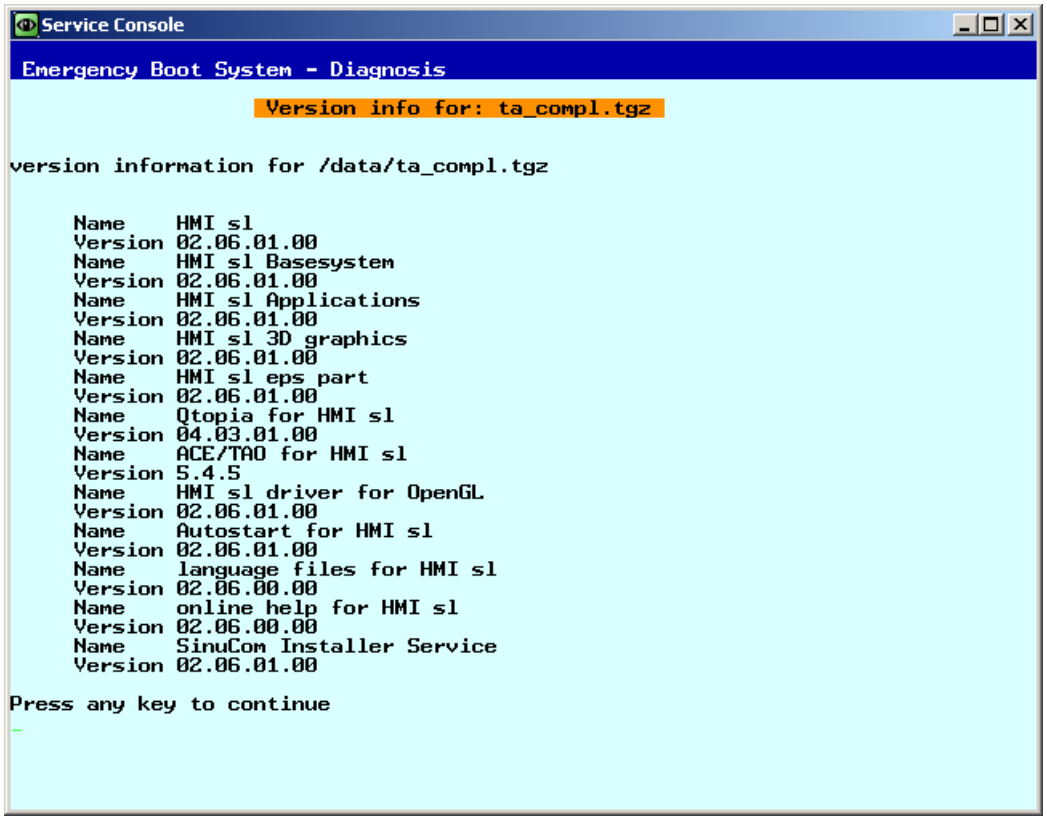


图 3-14　版本信息

4. 选择相应的 tgz 文件，并按下“Ok”确认。
   更新步骤成功执行后，会输出一条信息。

完成

重新启动系统后，新的数据才会生效。

另见

“Update system software from network drive”的步骤与此相似。 连接网络驱动器的信息另见：这样连接网络驱动器 (页 75)

系统的完整备份参见：这样安装完整的系统备份 (页 86)

3.6 系统诊断

3.6.1 执行系统诊断

诊断功能概览

在此菜单中，以下诊断功能用于显示网络设置：

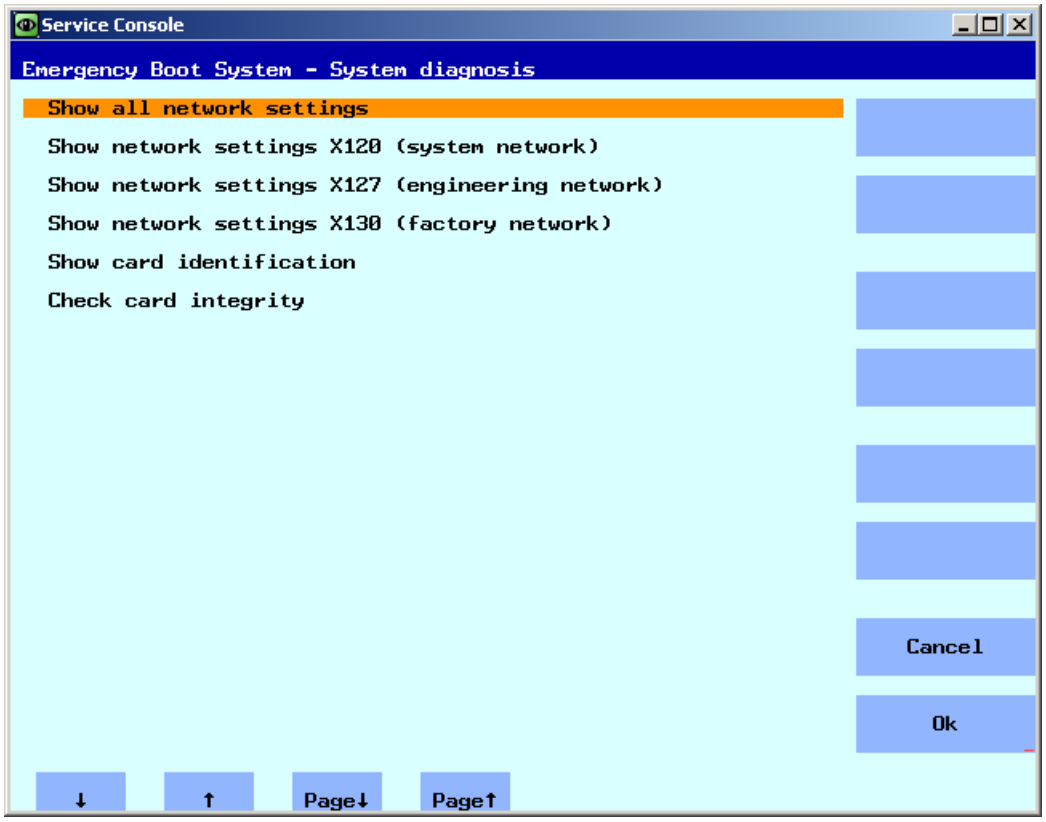


图 3-15 诊断功能

System diagnosis	说明
Show all network settings	显示所有 Etherne 接口的连接数据。
Show network settings X120 (system network)	显示 X120 接口的连接数据。
Show network settings X127 (engineering network)	显示 X127 接口的连接数据。

System diagnosis	说明
Show network settings X130 (factory network)	显示 X130 接口的连接数据。
Show card identification	输出系统 CF 卡的下列数据： • 序列号 • 系统软件版本 • 硬件（NCU）版本
Check card integrity	检查系统 CF 卡

示例： NCU 的所有以太网接口的连接数据：

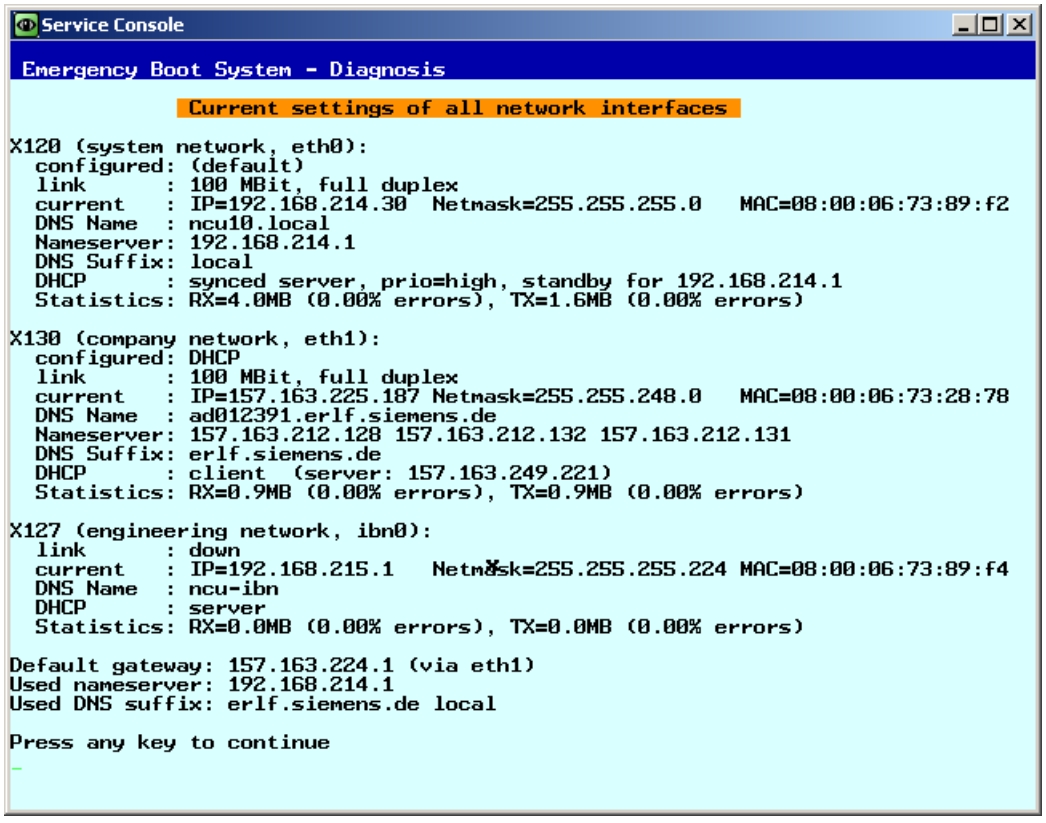


图 3-16 NCU：接口

另见

这样检查 CF 卡 (页 100)

3.6.2 这样修改系统设置

Modify NCU Settings

在此菜单中修改以下系统设置：

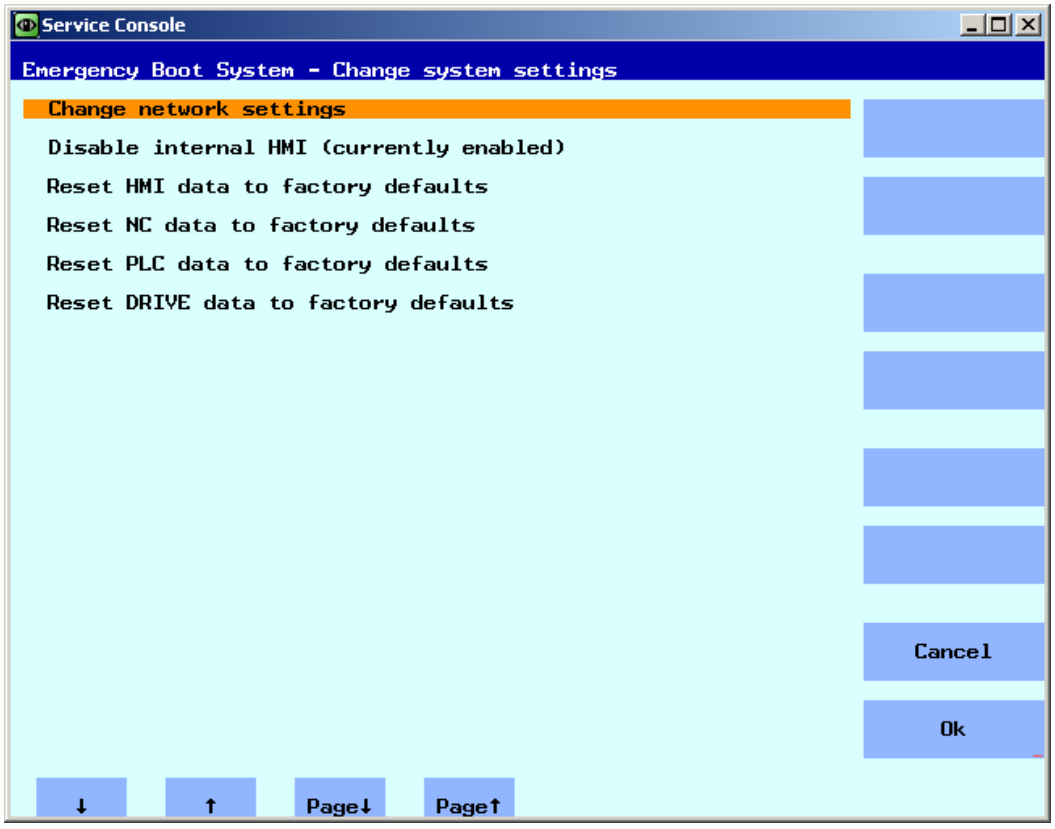


图 3-17 修改系统设置

菜单项	说明
Change network settings	修改 NCU 的网络设置。
Disable internal HMI (currently enabled)	激活/关闭 HMI 任务
Reset HMI data to factory defaults	恢复 HMI 数据的出厂设置。
Reset NC data to factory defaults	恢复 NC 数据的出厂设置。
Reset PLC data to factory defaults	恢复 PLC 数据的出厂设置。
Reset DRIVE data to factory defaults	恢复驱动数据的出厂设置。

注意
将 ... 数据恢复为出厂设置 选中的数据被恢复到出厂状态，所有之前完成的调试设置被清除。

3.6.3 这样检查 CF 卡

显示标识数据

步骤：

1. 在主菜单中选择菜单项“Diagnosis”。
   显示以下菜单：

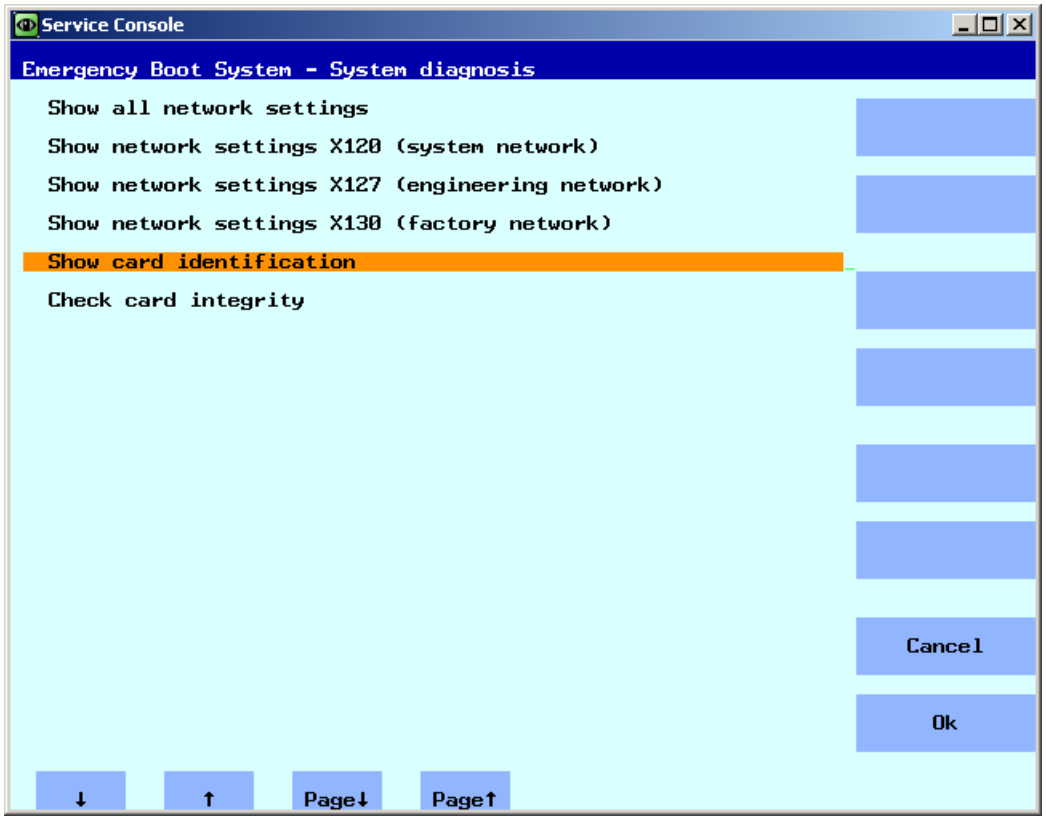


图 3-18 系统诊断

2. 选择菜单项“Show card identification”。
   输出以下标识数据：

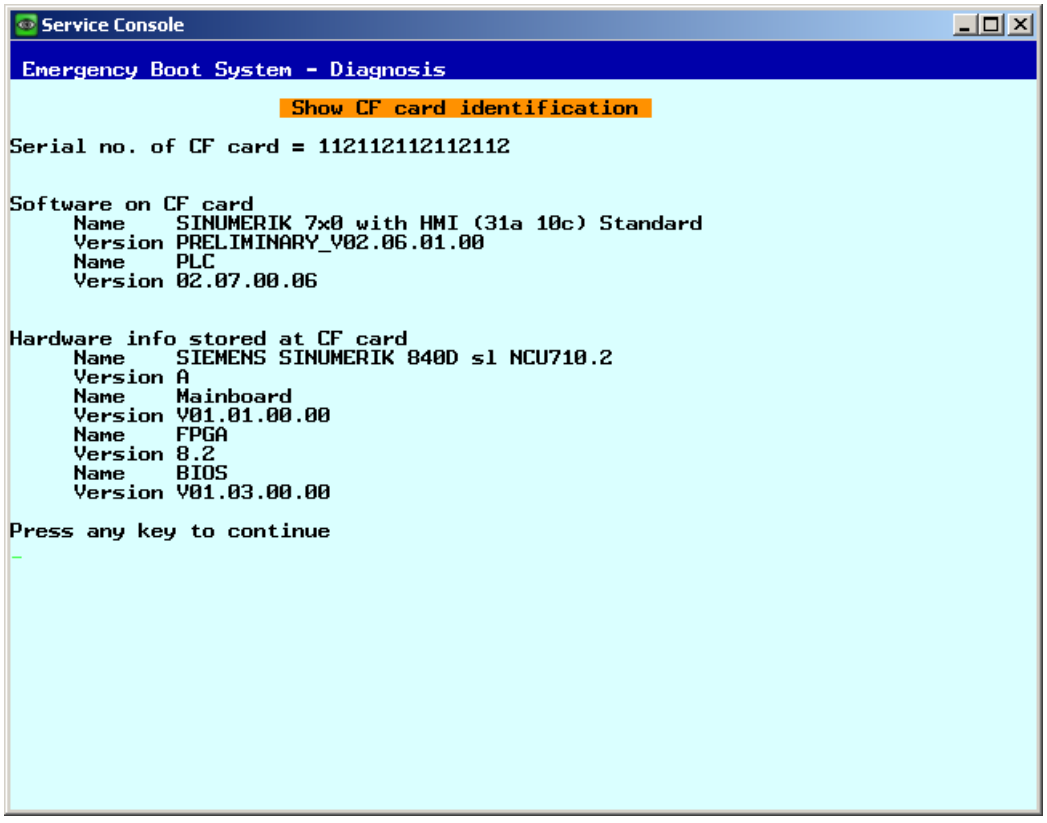


图 3-19 标识

3. 选择菜单项“Check card integritiy”。
检查 CF 卡上是否有错误：

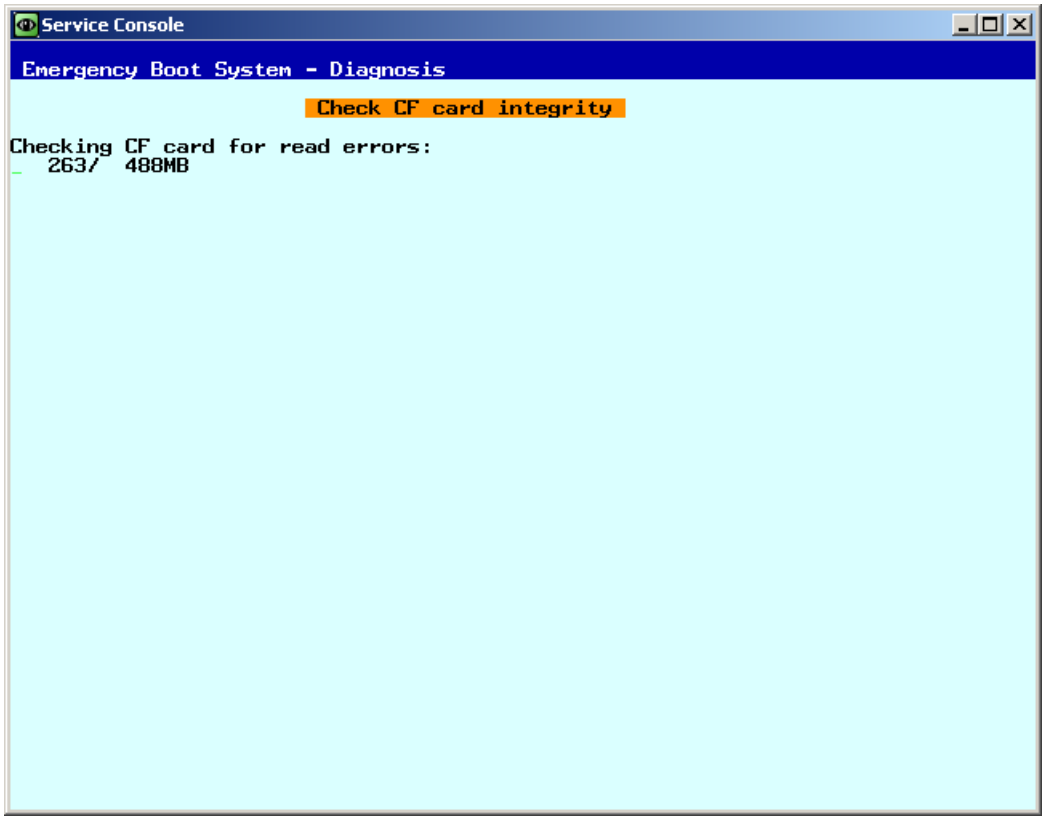


图 3-20 检查

说明

此过程可能会持续好几分钟。

3.6.4 调用 VNC 浏览器

使用目的

使用 VNC 浏览器可直接与 VNC 服务器连接，并操作 HMI 应用程序。

此外还可以列出全部的网络设备，从而获取设备网络概览。

启动 VNC 浏览器

步骤：

1. 在主菜单中选择菜单项“VNC Viewer”。
   显示以下菜单：

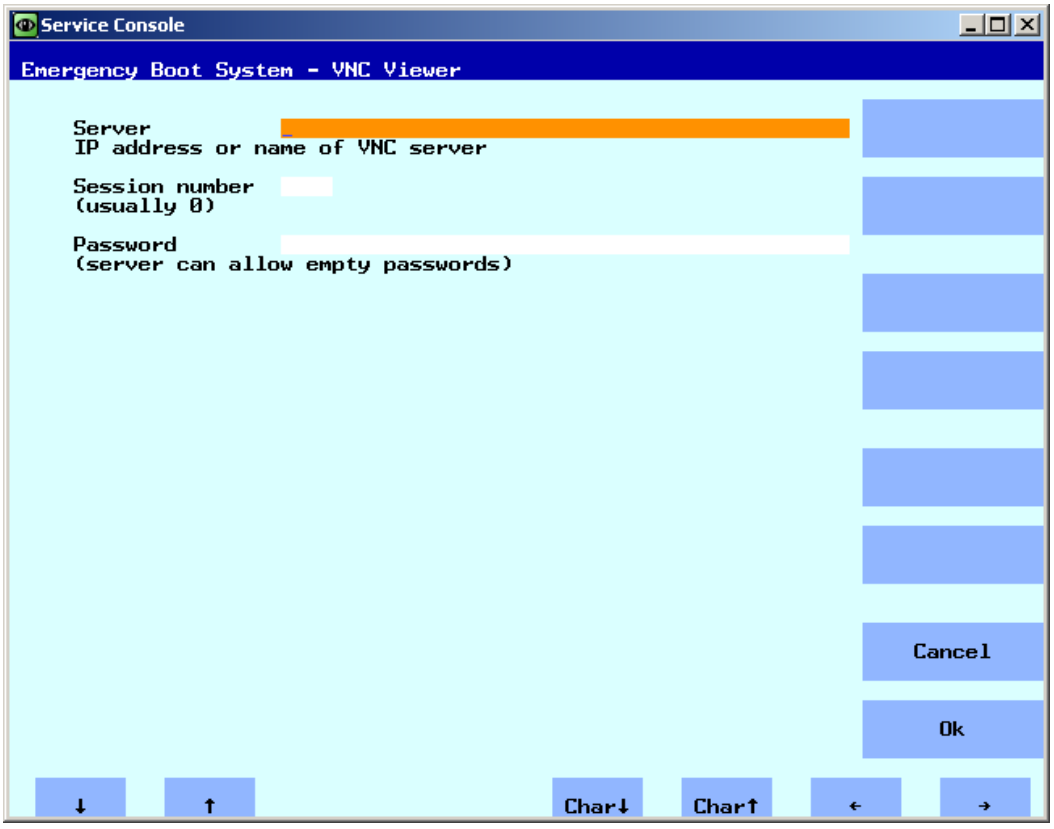


图 3-21 VNC Viewer

2. 输入以下连接数据，用于启动 VNC 浏览器：
   - 服务器名称，例如 IP 地址或 DNS。
   - 密码为可选设置，可为空。
   - 以下分配适用于“Session number”：

序号	含义
Session 0	HMI-Applikation
Session 4	Command Shell
Session 5	System logfile

3.6.5 WinSCP 和 PuTTY

使用

WinSCP 和 PuTTY 是可以自由使用的 Windows 开源程序。 WinSCP 主要用于文件传输、从 Linux 系统导出和导入文件，PuTTY 用于 Command Shell 的远程操作。

- WinSCP 可以通过下面的链接进行下载： http://winscp.net/eng/download.php (http://winscp.net/eng/download.php) (Installation Package)。
  WinSCP 也提供一个“Command Shell”，但是会有如下限制：仅可以取消命令，但不能应答查询。
- 相比之下，PuTTY 提供了一个完整功能的“Command Shell”。
  PuTTY 的网页是： http://www.chiark.greenend.org.uk/~sgtatham/putty (http://www.chiark.greenend.org.uk/~sgtatham/putty)

用这两个程序，维修技术员可以在 NCU 上登录并且处理维修任务。 用户名“manufact”和口令“SUNRISE”用于西门子维修人员。

启动 WinSCP

当维修技术员（例如：作为用户“manufact”以口令“SUNRISE”登录）在他所连接的 NCU 上用相应权限登录后，从 Window 启动 WinSCP 。

在菜单“Commands”中请选择“Open Terminal”，用来打开一个“Command Shell”。 那里可以用常规方式执行维修命令。

附录 A

A.1 缩写

CF	CompactFlash Card:存储卡
CFS	Cluster File System： 集群文件系统
CIFS	Common Internet File System：通用网络文件系统
DCK	Direct Control Keys:直接控制按键
DCP	Discovery and Basic Configuration Protocol： 发现与基本配置协议
DHCP	Dynamic Host Configuration Protocol：动态主机配置协议，向网络内的计算机动态分配 IP 地址和其他配置参数
DNS	Domain Name System：域名系统，将域名转换为 IP 地址
EBS	Emergency Boot System： 紧急启动系统
EKS	Electronic Key System：电子钥匙系统，用于检验并识别用户的系统（授权系统）
EUNA	End User Notification Administration： 终端用户通知管理
HMI	Human Machine Interface:操作界面
IRT	Isochronous Real Time (Ethernet)： 同步实时
LLDP	Link Layer Discovery Protocol：链路层发现协议，不受制造商影响的 Layer 2 协议，它根据标准 IEEE-802.1AB 定义，实现不同设备间的信息交换。
MAC	Media Access Control：媒体存取控制，MAC 地址是一个 48 位的以太网 ID。
MCP	Machine Control Panel：机床控制面板
MPI	Multi Point Interface：多点接口
MUI	Multilingual User Interface：多语言用户接口
NAT	Network Address Translation：网络地址翻译
NCK	Numerical Control Kernel：带有程序段处理、运行范围等的数控核心
NCU	Numerical Control Unit：NCK 硬件单元
NFS	Network File Service（网络文件服务）是一个网络协议。
NRT	Non Real Time (Ethernet)： 非实时
NTFS	New Technology File System： 新技术文件系统
NTLMSSP	NT LAN Manager (NTLM) Security Support Provider：NT LAN 管理器 (NTLM) 安全支持供应商
NTP	Network Time Protocol：网络时间协议，用于整个网络中时间同步的标准

NTPD	NTP Daemon： 在后台运行，无需由用户启动的服务程序。
PCU	PC Unit：计算机单元
PDEV	Physical Device： 物理设备
PG	编程器
PLC	Programmable Logic Control：可编程的逻辑控制器
RAM	Random Access Memory：随机存储器，可读写的程序存储器
RDY	Ready：系统就绪。
RFC	Remote Function Call：远程功能调用
SMB	Server Message Block：服务器信息块
SNMP	Simple Network Management Protocol：一种网络协议，由中央工作站来监控和调节网络组件，如：路由器、服务器、开关和打印机等。
SSD	Solid State Drive：固态硬盘
SSH	Secure Shell:与远程设备进行加密网络连接的协议
TCU	Thin Client Unit： 薄型客户单元
TFTP	Trivial File Transfer Protocol：非常简单的数据传输协议
UDP	User Datagram Protocol：用户数据报协议，一般通过 UDP 处理 NTP。
USB	Universal Serial Bus： 通用串行总线
USV	不间断电源
UTC	Universal Time, Coordinated：协调世界时（之前：Greenwich Mean Time：格林威治标准时间）
VNC	Virtual Network Computing： 虚拟网络计算

术语表

CFS (compressed file system)

CFS （文件结尾 ".cfs"）是一个压缩文件系统，和 zip 文件基本类似。 它包含文件和子目录，这些文件和子目录在控制系统上运行时和普通文件几乎一样。 CFS 中的文件和目录不可以进行更改。 它们会在运行时根据需要进行解压缩。

NFS (network file system)

NFS 是在 Unix 环境下最常用的远程文件系统协议，也可用于 Windows 操作系统。 Unix 计算模型在每次存取时都会同时提供一个 UID 和 GID，服务器根据这些数据决定是否允许进行操作，因此计算模型很大程度上决定了 NFS。 服务器默认用户提供的是正确的 ID。

Remote File System

远程文件系统是通过网络进行响应的文件系统。 这些文件物理上位于网络中的另一台计算机上（“服务器”），但在本地也可以像其他所有文件一样显示。 对此类文件的操作可以通过网络传到服务器上，而不是直接在本地存储媒介（硬盘，CF 卡）上进行。

因为一个服务器通常输出多个文件系统，所以除了规定服务器的名称之外，还要规定所需文件系统的名称。

SMB (server message block)

SMB 是一个协议，MS Windows 的远程文件系统是以该协议为基础的（也可以作为网络驱动器、使能、Share 等）。 SMB 连接在特定用户的联系中总是有效的，该用户在服务器上必须是已知的。 输出的文件系统有一个名称（使能名称），以该名称响应文件系统，用户无需知道服务器上具体的路径。

VNC（virtual network computing：虚拟网络计算）

虚拟网络处理是一种软件，它可以在本地计算机（VNC 浏览器运行中）上显示远程计算机（VNC 服务器运行中）的屏幕内容并且可以通过本地计算机的键盘和鼠标来操作远程计算机。

网络接口

网络接口是实现网络通讯的接口。 在 NCU 上，网络接口是以太网接口。

子系统

子系统是一个 CFS, 其不仅包含多个文件、而且运行时会执行某个程序。 对此， CFS 包含一个脚本，用该脚本可以控制该程序的启动和停止。

出于这个原因，只有管理员才允许对 NFS 文件系统进行设置，并且 NFS 大多设置为统一的管理环境。 服务器输出的文件系统直接通过其路径在服务器上进行响应。

索引

"

"服务"菜单, 74

7

7 段显示, 10

B

basesys.ini, 15
  CheckTCUforUpdatePackage, 31
  DefaultCIFSSecurity, 30
  DHCPClientID, 18
  DHCPDNoMasterWait, 21
  DisableCompanyNet, 16
  DisableIBNForwarding, 23
  DisableIBNNet, 22
  DisableNATRouting, 22
  DisableNTPTimeSync, 31
  DisablePLCTimeSync, 31
  DisableSubsystems, 30
  DisableSysNet, 19
  Domain, 18
  EnableCoreDumps, 31
  EnableDHCP_IBNNet, 22
  EnableDHCPD_SysNet, 20
  EnableSysNetToIBNForwarding, 22
  EnableUsbShares, 32
  ExternalDcpEnabled, 25
  ExternalIP, 16
  ExternalLldpEnabled, 26
  ExternalLldpTLVsTxEnabled, 26
  ExternalNetMask, 16
  FirewallOpenPort, 30
  Gateway, 16
  Hostname, 17
  InternalDcpEnabled, 25
  InternalDNSDomain, 21
  InternalDynRangeEnd, 21
  InternalDynRangeStart, 21
  InternalIP, 19
  InternalIP_Alias, 19
  InternalLldpEnabled, 25
  InternalLldpTLVsTxEnabled, 26
  InternalNetMask, 19
  InternalNetMask_Alias, 20
  LogFileLimit, 29
  Nameservers, 17
  PLCTimeType, 29
  SNMPAutLocation, 24
  SNMPContact, 24
  SNMPFunction, 24
  SNMPLocation, 23
  SNMPStation, 24
  SyncModeDHCPD_SysNet, 20
  Timeservers, 17
  Timezone, 27

C

CF 卡
  存储器划分, 6
  分区, 5
CFS (compressed file system), 45

D

DHCP 同步, 56

E

Emergency Boot System, 71
Ethernet 接口, 13

I

IP 地址
  显示, 14

N

NCK 调试开关, 8
NCU 启动, 9
NFS (network file system), 45

P

PLC 运行方式开关, 8

R

Remote File System, 45

S

SMB (server message block), 46

V

VNC（virtual network computing：虚拟网络计算）, 13

W

WinSCP, 104

备

备件, 77

段

段落
    [DCP], 25
    [ExternalInterface], 16
    [IBNInterface], 22
    [InternalInterface], 19
    [LinuxBase], 27
    [LLDP], 25
    [SNMP], 23

服

服务命令
    句法, 43
服务系统
    为 NCU 创建, 72

复

复位键, 9

更

更新
    BIOS, 92
    固件, 92

接

接口
    Ethernet, 44
    NCU, 13

权

权限（特权）, 42

上

上电复位, 9

时

时区, 27

数

数据
    恢复, 90

同

同步
    DHCP, 56

网

网络接口, 13
网络驱动器, 75

维

维修工具 WinSCP, 104
维修命令
    权限, 42
    应用, 42
维修系统
    备份数据, 88
    恢复数据, 89

系

系统
    日志文件, 5, 31
    诊断, 10
系统故障, 12

应

应用情况, 71

用

用户（缺省设置）, 6
用户组, 42

指

指令
    Backup, 60
    Check-cf, 48
    Clear（清除）, 48
    Closeport, 49
    Connect, 50
    Disconnect, 53
    Enable, 55
    ntpdiag, 56
    Portstatus, 58
    Reboot, 58
    Restart, 58
    Restore, 59
    Save, 60
    save-nvram, 62
    set-svc-route, 62
    Show, 63
    Start, 67
    Stop, 67
    usb, 67
    vncpwd, 69
    帮助, 47
    打开端口, 57
    取消使能, 52

装

装载操作系统, 10

子

子网掩码, 14
子系统, 45
    名称, 45

附录 A

A.1 文档一览

有关 SINUMERIK 840D sl 功能的详细文档，自版本 4.8 SP4 起，请参见 840D sl 文档一览 (https://support.industry.siemens.com/cs/ww/en/view/109766213)。



您可以直接打开文档或者下载 PDF 和 HTML5 格式。

文档分为以下几个类别：

- 用户：操作
- 用户：编程
- 制造商/服务：功能
- 制造商/服务：硬件
- 制造商/服务：配置/调试
- 制造商/服务：Safety Integrated
- 制造商/服务：SINUMERIK Integrate / MindApp
- 介绍和培训
- 制造商/服务：SINAMICS

A.1 文档一览