SIEMENS

# SINUMERIK 840D sl

# Base software and HMI-Embedded

Commissioning Manual



Valid for
SINUMERIK 840D sl/840DE sl control

Software Version
NCU System Software 1.5
with HMI Embedded 7.5


01/2008
6FC5397-8CP10-1BA0

## Safety Guidelines

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

| ⚠DANGER |
|---|
| indicates that death or severe personal injury **will** result if proper precautions are not taken. |

| ⚠WARNING |
|---|
| indicates that death or severe personal injury **may** result if proper precautions are not taken. |

| ⚠CAUTION |
|---|
| with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken. |

| CAUTION |
|---|
| without a safety alert symbol, indicates that property damage can result if proper precautions are not taken. |

| NOTICE |
|---|
| indicates that an unintended result or situation can occur if the corresponding information is not taken into account. |

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

## Qualified Personnel

The device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by **qualified personnel**. Within the context of the safety notes in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

## Prescribed Usage

Note the following:

| ⚠WARNING |
|---|
| This device may only be used for the applications described in the catalog or the technical description and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens. Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance. |

## Trademarks

All names identified by ® are registered trademarks of the Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

## Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Automation and Drives
Postfach 48 48
90327 NÜRNBERG
GERMANY

Ordernumber: 6FC5397-8CP10-1BA0
℗ 02/2008

# Preface 1

## 1.1 Preface

### SINUMERIK Documentation

The SINUMERIK documentation is organized in three parts:

- General Documentation
- User Documentation
- Manufacturer/Service Documentation

An overview of publications, which is updated on a monthly basis and provides information about the language versions available, can be found on the Internet at:

http://www.siemens.com/motioncontrol

Select the menu items "Support" → "Technical Documentation" → "Overview of Publications".

The Internet version of DOConCD (DOConWEB) is available at:

http://www.automation.siemens.com/doconweb

Information about training courses and FAQs (Frequently Asked Questions) can be found at the following website:

http://www.siemens.com/motioncontrol under menu item "Support".

### Target group

This documentation is intended for commissioning personnel.

The plant/product is installed, connected, and ready to start. The Commissioning Manual should contain all necessary information about or at least references to subsequent procedures such as testing the cabling, power on and functional testing.

### Benefits

The intended target group can use the Commissioning Manual to test and commission the product/system correctly and in total safety.

Utilization phase: Setup and commissioning phase

## Standard scope

This documentation only describes the functionality of the standard scope. Extensions or changes made by the machine manufacturer are documented by the machine manufacturer.

Other functions not described in this documentation might be executable in the control. However, no claim can be made regarding the availability of these functions when the equipment is first supplied or in the event of servicing.

Further, for the sake of simplicity, this documentation does not contain all detailed information about all types of the product and cannot cover every conceivable case of installation, operation or maintenance.

## Hotline

If you have any technical questions, please contact our hotline:

| | Europe / Africa |
|---|---|
| Phone | +49 180 5050 222 |
| Fax | +49 180 5050 223 |
| Internet | http://www.siemens.com/automation/support-request |

| | America |
|---|---|
| Phone | +1 423 262 2522 |
| Fax | +1 423 262 2200 |
| E-mail | mailto:techsupport.sea@siemens.com |

| | Asia / Pacific |
|---|---|
| Phone | +86 1064 719 990 |
| Fax | +86 1064 747 474 |
| E-mail | mailto:adsupport.asia@siemens.com |

### Note

National telephone numbers for technical support are provided under the following Internet address:

http://www.siemens.com/automation/service&support

## Questions about this documentation

If you have any queries (suggestions, corrections) in relation to this documentation, please send a fax or e-mail to the following address:

| | |
|---|---|
| Fax | +49 9131- 98 63315 |
| E-mail | mailto:docu.motioncontrol@siemens.com |

A fax form is available at the end of this document.

## SINUMERIK Internet address

http://www.siemens.com/sinumerik

## EC Declaration of Conformity

The EC Declaration of Conformity for the EMC Directive can be viewed/downloaded from the Internet at:

http://support.automation.siemens.com

under the Product Order No. 15257461or at the relevant branch office of the A&D MC Division of Siemens AG.

SIEMENS



# SINUMERIK 840D sl

# Commissioning HMI-Embedded (IM2)

Commissioning Manual

Valid for

SINUMERIK 840D sl/840DE sl control system

| Software | Version |
|---|---|
| NCU System Software | 1.5 |
| with HMI-Embedded sl | 7.5 |


01/2008
6FC5397-8CP10-1BA0

## Safety Guidelines

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

| ⚠DANGER |
| --- |
| indicates that death or severe personal injury **will** result if proper precautions are not taken. |

| ⚠WARNING |
| --- |
| indicates that death or severe personal injury **may** result if proper precautions are not taken. |

| ⚠CAUTION |
| --- |
| with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken. |

| CAUTION |
| --- |
| without a safety alert symbol, indicates that property damage can result if proper precautions are not taken. |

| NOTICE |
| --- |
| indicates that an unintended result or situation can occur if the corresponding information is not taken into account. |

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

## Qualified Personnel

The device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by **qualified personnel**. Within the context of the safety notes in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

## Prescribed Usage

Note the following:

| ⚠WARNING |
| --- |
| This device may only be used for the applications described in the catalog or the technical description and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens. Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance. |

## Trademarks

All names identified by ® are registered trademarks of the Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

## Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Automation and Drives
Postfach 48 48
90327 NÜRNBERG
GERMANY

Ordernumber: 6FC5397-8CP10-1BA0
Ⓟ 02/2008

# Table of contents

# 1 Introduction

This Commissioning Guide describes the procedures for installing the HMI-Embedded software.

For further reading on special NCK, PLC, or drive functions, please consult the following Manual.

### References

CNC Commissioning Manual: NCK, PLC, Drive

## 1.1 Hardware / software requirements

### 1.1.1 Hardware

It is a requirement that all system components be fully mounted, installed and configured.

At least the following system components are needed in order to run the HMI-Embedded (Human Machine Interface Embedded) software:

- Numerical Control Unit (NCU sl)
- Operator panel (OP)
- Machine control panel (MCP)
- Thin Client Unit (TCU)
- SINAMICS 120 drive system

For this Sinumerik 84D sl system, one or more TCUs (Thin Client Units) are used to operate and monitor the control. The actual HMI-Embedded operator software runs on the NCU, while the TCUs make up the user interface.

The TCUs are equipped with USB interfaces, to which you can connect USB devices (e.g., USB-FlashDrive, USB floppy disk drive, card reader...) to download data onto the control or to back up the control data.

A description of the individual components and how they can be combined, connected and installed can be found in various manuals:

See Documentation (Page 9)

### 1.1.2 Software

A CompactFlash card containing the complete HMI-Embedded software package is supplied with the NCU.

#### License release

You need a license release (authorization to use the software) in order to use various options of HMI-Embedded. Activation is required before the protected software can be used.

See Licensing (Page 13)

## 1.2 Accessories

#### Memory media

**Memory expansion with the CompactFlash Card**

You can use the CompactFlash card as a storage location for backing up data.

To obtain additional memory expansion, you must purchase the “256 MB user memory” option with order number: 6FC5800-0AP12-0YB0.

This option is then released via the license manager (addit. 256 MB HMI user memory on CF card of NC,...).

**USB FlashDrive**

You can use a 512 MByte USB FlashDrive as an additional memory medium:
Order Number: 6ES57648-0DC20-0AA0

However, do not use the USB FlashDrive to directly execute programs as there is no way of ensuring that it cannot fall out or be accidentally removed.

#### Keyboard

With HMI-Embedded you have to distinguish between upper and lower case. As it is not possible to switch to lower case mode on the keyboard of the machine control panel or operator panel front, you will need an external keyboard (USB connector) for commissioning.

Plug in the cable connection of the keyboard in the USB interface on the front of the operator panel. The system immediately detects the keyboard.

## 1.3 Documentation

You require the following documentation to install and start up the SINUMERIK 840D sl:

### References

Equipment Manual and Networking

SINUMERIK 840D sl Equipment Manual

CNC Commissioning Manual: NCK, PLC, Drive

Function Manual Basic Functions

List Manual, Lists (Book1)

Diagnostics Manual

Operating Manual HMI-Embedded

CNC Commissioning Manual: ShopTurn

CNC Commissioning Manual: ShopMill

## 1.4 Startup

**Note**

Before a cold restart all USB devices must be removed from the TCU and not connected again until commissioning has been completed!

1. Insert the CompactFlash Card containing the HMI-Embedded software into the slot marked "CF" (interface X109) before switching on the NCU.
2. Now switch on the power supply. The NCU starts up automatically.
   By default, you will be in the "Machine" operating area, "Jog" mode after commissioning and the "Reference point approach" window will be active.

For additional information about booting is provided in the following documentation:

### References

CNC Commissioning Manual: NCK, PLC, Drive

# Configuring the HMI system 2

## 2.1 Protection levels

### 2.1.1 Overview

#### Overview

Access to programs, data and functions is useroriented and controlled via 8 hierarchical protection levels. These are divided into

- 4 password levels for Siemens, machine manufacturer, start-up personnel, and end user
- 4 keyswitch positions for end user

There are protection levels 0 to 7 (see table below); where

- 0 is the highest and
- 7 is the lowest level.

Table 2-1 Protection levels concept

| Protection level | Protected by | Area |
|---|---|---|
| 0 | Password | Siemens |
| 1 | Password: SUNRISE (default) | Vendor |
| 2 | Password: EVENING (default) | Service |
| 3 | Password: CUSTOMER (default) | User |
| 4 | Keylock switch position 3 | Programmer, machine setter |
| 5 | Keylock switch position 2 | Qualified operator |
| 6 | Keylock switch position 1 | Trained operator |
| 7 | Keylock switch position 0 | Semi-skilled operator |

### 2.1.2 Editing the password of the protection levels

You can activate the protection levels used with passwords via the user interface.

You can edit the password in the following ways:

- Set new password.
- Change password.
- Delete password.

#### Password Set

1. Select the “Commissioning” operating area.
2. Press the "Set password" softkey.
3. Enter the appropriate password in the text field "Please enter password" for the particular area (manufacturer, service, user).
4. Press the "OK" softkey to confirm your entry. A permissible password is set and the valid protection level is displayed. Invalid passwords will be rejected.

#### Change password

You must change the standard passwords to obtain a secure access protection.

1. Press the "Change password" softkey.
2. Using the vertical softkeys, select the area (e.g. user) for which you want to assign a new password.
3. Enter the new password in the input fields "Please enter new password".
4. Press the "OK" softkey to save your entry.

#### Delete password

Press the "Delete password" softkey. The access authorization is reset.

Access authorization is not automatically deleted at POWER ON!

| NOTICE |
| --- |
| When a general reset is performed on the NCK, the default passwords are restored. |

## 2.2 Licensing

A suitable license is needed in order to use the options.

In the commissioning operating area, HMI-Embedded has three menus, which contain information about the status of the options and their licenses.

### Prerequisite

You need at least the access authorization for the manufacturer area in order to set or reset options.

### Procedure

1. In the "Commissioning" operating area press the ">" ETC key.
2. Press the "License" softkey on the new softkey bar. The following three menus are then displayed:
   – Overview
   – All options
   – Missing licenses

### Overview

1. The following information appears when you press the “Overview” softkey:

| | |
|---|---|
| Hardware serial number | Shows the unique number of the SINUMERIK sl |
| Type of hardware, e.g.: | "SINUMERIK 840D sl" |

Status message of the license key, e.g., “License key is NOT sufficient!”

1. Enter the new license key in the input field that appears after the prompt “Please enter a license key:”. Hyphens are inserted automatically after every fourth character. Their only purpose is to improve legibility.
2. Press the “Apply” softkey to send the data to the NCU and store them there.
3. Press the “NCK Reset” softkey to activate the licenses.

### All options

1. When you press the “All Options” softkey, all NCU and HMI-Embedded options are displayed. A description and order number are displayed for each option. Options shown red are activated, however, not yet licensed or not licensed in adequate number. You can set or reset the availability in the adjacent “Set” field.
2. After changing the option, press the "Apply" softkey.

### Missing licenses

1. When you press the “Missing Licenses” softkey, any insufficiently licensed options will be listed.

You can set and reset the option in the adjacent field, as in the “All Options” menu.

---

**Note**

**Use of non-licensed options**

Before licensing options, you can also activate them temporarily without the license keys and use them for test purposes. In this case, the control displays alarm 8080 periodically, indicating that the corresponding license has not been allocated to the hardware.

---

A detailed description of the license management is provided in the following documents:

### References

CNC Commissioning Manual: NCK, PLC, Drive

## 2.3 Setting-up network connections

The NCU hardware has its own network card and can be integrated in an existing network.

If the appropriate option is set, additional horizontal softkeys can be integrated in HMI-Embedded with icons for a mapped drive or a local drive.

With the appropriate settings in the Program, Service and Commissioning operating areas, you can access various memory locations or save data.

### Possible connections

The number of connections between the HMI-Embedded system and different servers or local drives is limited to eight.

These connections are also called "logical drives" and can be configured in three different ways:

- from the HMI user interface
- using machine data
- using the "logdrive.ini" file

## 2.3.1 Setting up the drive connection from the HMI user interface

The drive connections (local, network and USB drives) and their softkeys are set up in the settings window of the commissioning operating area.

Network connections can be set up in the screen displayed below. These connections are also called "logical drives". A logical drive may be any of the following:

- Network connection,
- Internal drive, CompactFlash card (PCMCIA memory card interface)
- USB interfaces, e.g., diskette drive can be connected via USB

### Procedure

1. In the "commissioning" operating area press the "Connections" sofkey.
2. Press the "Logical drives" softkey. The "Network Connections" dialog box opens where you can set up the connections as follows:

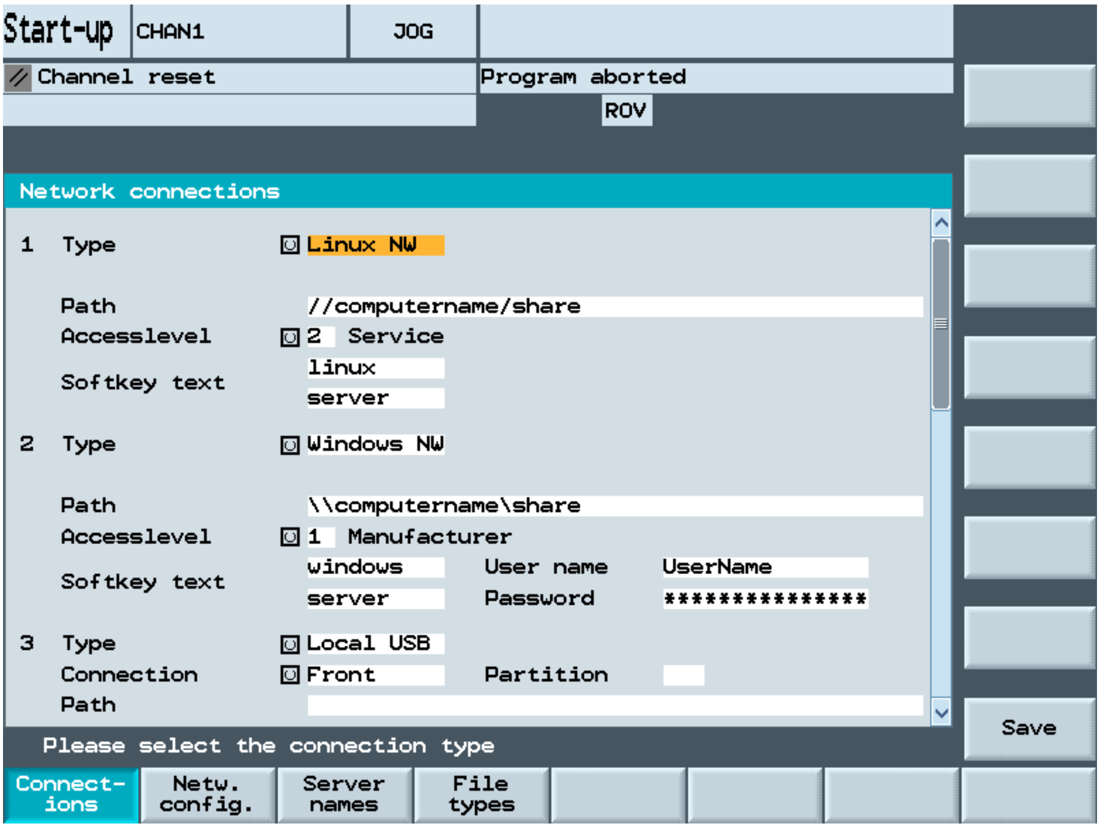



Figure 2-1 Drive connections

3. Use the "Select" key to select entries for the following fields:
   - "Type"
   - "Connection"
   - "Access Level"

| | |
|---|---|
| Type: | No drive, USB local, USB global |
| Connection: | Front (preassigned), X203, X204 |
| Access level: | 1 Manufacturer |
| | 2 Service |
| | 3 User |
| | 4 Keyswitch setting 3 |
| | 5 Keyswitch setting 2 |
| | 6 Keyswitch setting 1 |
| | 7 Keyswitch setting 0 |

**Note**

You need the memory expansion (optional) to access the CompactFlash card. When the option is set, the softkey is displayed: Memory card (CF) (supplied CompactFlash card with Software HMI-Embedded)

see Accessories (Page 8)

1. Enter the other fields manually. You can choose any softkey text.
   The following options are available for establishing connections:

| Type name | Terminal | Path (path definition) | Connection to |
|---|---|---|---|
| No drive | - | - | |
| Local USB | Front (preassigned)<br>X203<br>X204 | acttcu://0,2<br>acttcu://2<br>acttcu://1 | TCU, to which the device is connected via USB. |
| USB global | X203<br>X204 | | Additional TCUs |
| Flash card | - | .../user/sinumerik/data/... | CompactFlash card in the NCU |
| NW Linux | - | // Linux ComputerName/Linux ShareName | |
| NW Windows | - | // WindowsComputername/Windows ReleaseName | A released directory in a network drive. The path name must start with "//". |

- A numerical setting for DOS (Windows) is made in the "Partition" field.
- Two lines, with a maximum of nine characters each, are available for the softkey text.
- You must enter a user name and a password for Windows shares.
  If you do not define a user, a standard user (PCU20_USER) will be entered and a standard password used.
- No user names or passwords are needed for Linux (NFS) shares and local/global network paths (USB).

In the "Program" and "Services" operating areas, the first four softkeys of the established connections are displayed in the horizontal bar. The other four softkeys appear in the second horizontal bar, which can be viewed by pressing the ">" ETC key.

The selection of the drive can be prevented by setting the appropriate access level.

## 2.3.2 Configuring the network

### Settings

The data required for networking the components are entered in the "Network Configuration" window.

Only the TCP/IP protocol is supported and can be configured within the network setup screens.

1. In the "Startup" operating area press the "Network Config." softkey. The following window appears:

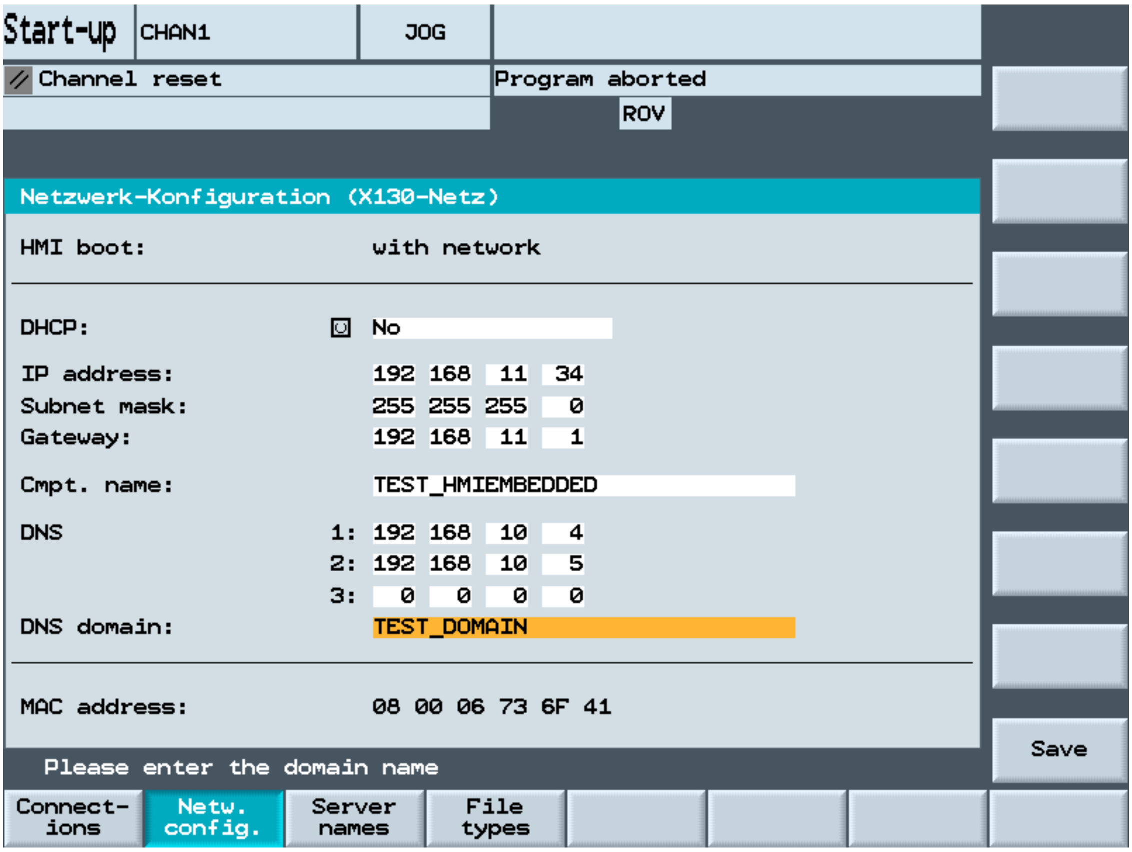



Figure 2-2 Configuring the network

- DHCP:
  Use the "Select" key to select the "Yes" or "No" setting, depending on whether or not a DHCP server is available.
- IP addresses:
  A unique IP address must be selected for each computer.
- Subnet mask:
  A unique subnet mask must be entered for the entire network.
- Gateway:
  A gateway is not needed for a local area network.
  Please ask your administrator for the settings of a company network.
- Computer name:
  The name is optional, as it is only a symbolic identifier and can be replaced by the IP address at any time.
- DNS:
  If the DNS (Domain Name Server) has the appropriate assignment of computer and IP address, no entry is required here.
  MAC address: This is unique (on the NCU 710 after X130). If the NCU is already known, it will be displayed automatically.

2. Press the "Save" softkey to save the connection settings.

## 2.3.3 Examples: Network configuration

This section describes two possible configurations:

1. A stand-alone local area network
2. A large company network.

### Local area network

A small stand-alone network is the easiest option.

| NOTICE |
| --- |
| As no "hub" is used, you will need a special twisted-pair cable for the point-to-point connection. |

1. Select for DHCP: "Yes", the IP address, subnet mask and gateway data are transmitted and displayed automatically via a DHCP server. These data cannot be overwritten.
   You have to enter the computer name and DNS domain manually.
2. Select for DHCP: "No", because no server exists, for example, you have to determine and enter the data yourself.

3. For stand-alone local area networks, RFC 1597 provides for three address ranges so that there are no conflicts while using the Internet at the same time:

| | | | |
|---|---|---|---|
| 10.0.0.0 | > | 10.255.255.255 | (Class A network) |
| 172.16.0.0 | > | 172.31.255.255 | (Class B network) |
| 192.168.0.0 | > | 192.168.255.255 | (Class C network) |

The IP addresses for small networks should be selected from these ranges.

## Company network

Since the company network already exists, you will need to ask the administrator for the settings and addresses.

A fixed IP address is used to address the company network. You will need to ask your administrator for an IP address for each NCU.

The computer name is also provided by the network administrator and corresponds to the IP address that is set.

Further, it is necessary to have the correct subnet mask for the network and a default gateway (router).

If you intend to use a DNS server, you will also need its IP address and the DNS domain.

## Server name (setting not mandatory)

In order to assign symbolic names to the computers, the relevant nodes to be addressed within the network can be entered so that their names can be resolved:

| | |
|---|---|
| Server name: | |
| 192.168.0.20 | R2345 |
| 192.168.0.21 | R2346 |

With these settings computers R2345 and R2346 can be addressed by means of their symbolic names.

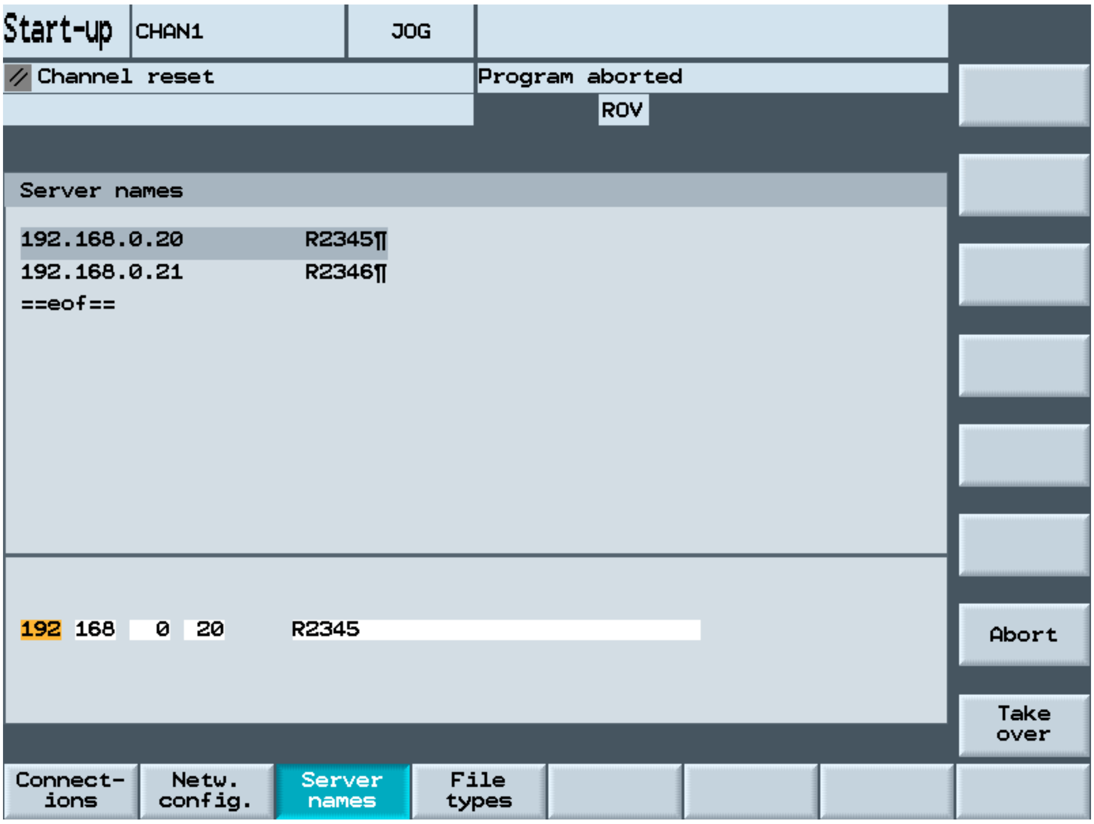



Figure 2-3 Server name

Operation:

The first entry in the line is the IP address.

The second entry is the computer name. The data already specified appears in the upper section of the screen.

By selecting the "New" and "Edit" softkeys you can go to the lower half of the screen where you can edit entries.

You can change back to the upper section by pressing the "Cancel" softkey (data is not saved) or the "Save" softkey (data is saved).

With external drives (network connections) you need to specify the full network path (\\computer name\release name\path data), whereby the path name is optional:

\\R4711\workpieces\template

\\R4812\\workpieces

These names must be available on the network computer. The network computer, for example, is called "\\R4812". The share name WORKPIECES must be specified on any hard disk drive on this computer.

### File Types

The file types are set up as needed and are evaluated and displayed by the NCU.

Using the file extension, the system filters which files are visible and which are not. This filter function can be set as follows:

If the "New" or "Edit" softkeys are selected, a switch is made to the lower half of the window where a new extension can be entered or an existing extension edited. To exit, press "Cancel" (without changes) or "Save" (with changes).

## 2.3.4 Setting up the drive connection via MD

If you do not require more than four drive connections you can set them up via machine data.

For the authorization of the input for configuration data, the following display machine data must be set up:

MD9509 $MM_USER_CLASS_DIRECTORY_CHG (weighting 0 ... 7)

### Display machine data

The following display machine data is available for authorizing the use of the created drives:

For the “Program” and “Services” areas

MD9510 $MM_USER_CLASS_DIRECTORY1_P
MD9511 $MM_USER_CLASS_DIRECTORY2_P
MD9512 $MM_USER_CLASS_DIRECTORY3_P
MD9513 $MM_USER_CLASS_DIRECTORY4_P
(weighting 0...7)

For the “Machine” area

MD9516 $MM_USER_CLASS_DIRECTORY1_M

MD9516 $MM_USER_CLASS_DIRECTORY2_M

MD9516 $MM_USER_CLASS_DIRECTORY3_M

MD9516 $MM_USER_CLASS_DIRECTORY4_M

The network drives are configured in the display machine data:

MD9676 $MM_DIRECTORY_SOFTKEY_PATH1

MD9676 $MM_DIRECTORY_SOFTKEY_PATH2

MD9676 $MM_DIRECTORY_SOFTKEY_PATH3

MD9676 $MM_DIRECTORY_SOFTKEY_PATH4

The display machine data for the access rights are effective immediately, i.e., after switching to the relevant application. No restart is necessary.

### 2.3.5 Setting up the drive connection via "logdrive.ini"

You can also use the "logdrive.ini" file to set up all drives (e.g. NW Linux, USB interface) that do not require a password.

The settings in the display machine data are ignored.

**Note**

A Windows drive (prefix: smb) cannot be changed or set up in the "logdrive.ini" file, because an encrypted password is required. It is not possible to enter encrypted data in the file.

#### Structure of the "logdrive.ini" file

You define all logical drives in section [CONNECTIONS].

ConnectionNum: Number of logical drives

Settings for each logical drive (replace "X" with the relevant drive letter).

| | |
|---|---|
| ConnectionX | Drive path, e.g., C:/Temp. |
| SK_ConnectionX | Softkey text<br>• Text ID (only for language-specific softkey texts)<br>• Text |
| UsernameX | User name (for network drives) |
| PasswordX | Password belonging to UsernameX |
| SoftkeyIndexX | Softkey position on the horizontal softkey bar. Index 1-8 is the first ETC level, 9-16 second level, etc. |
| SoftkeyTextContextX: | Text context for the text ID (only for language-specific softkey texts) |
| SoftkeyTextFileX | Text file that contains the softkey text identified by the text ID (cf. SK_ConnectionX) (only for language-specific softkey texts) |
| SoftkeyPictureX | File name of the icon |
| AccessProgramX | Access level for the program area |
| AccessMachineX | Access level for the Machine area |
| AccessServicesX | Access level for the Services area |

## Drive paths

Use the syntax described in the table below for entering the drive paths.

| Logical drive | |
|---|---|
| //NC/MPF.DIR | Current NC |
| //NCU_1/MPF.DIR | Specific NC |
| /SIEMENS/SINUMERIK/MPF.DIR/ | Local drive on CF |
| /siemens/sinumerik/mpf.dir/ | |
| /card/user/data | |
| C:/ MPF.DIR/ | External drives WinXP hard disk (Win version) |
| c:\mpf.dir | |
| //tcu1/X204/mpf.DIR | Global TCU-USB drive |
| //tcu1/X203,1/test/mpf.dir | -\|\|- with partition data |
| //usb/X204/mpf.DIR | Global NCU-USB drive |
| //ACTTCU/FRONT<br>//ACTTCU/FRONT,1 | Local TCU-USB drive (=USB device on the active TCU, also has a matching global path)<br>-\|\|- with partition data |
| //ef3307/MPF.DIR | External network drives (WinXP) |
| //ef3307/MPF.DIR | External network drives (Linux) |
| //NC/NC_CARD/MPF.DIR | CF Card via SSH |
| //PLC/XXXXXXX | PLC |

## 1. Example

You must make the following entries to display the content of the first partition of the USB device connected to the front of the actual TCU:

```
[CONNECTIONS]
ConnectionNum=1
Connection1=//ACTTCU/FRONT,1
SK_Connection1=Front-USB
SoftkeyIndex1=3
SoftkeyTextContext1=SlPmLogicalDrives
SoftkeyTextFile1=slpmdialog
SoftkeyPicture1=sk_usb_front.png
AccessProgram1=7
AccessMachine1=7
AccessServices1=7
```

A new softkey then appears at position HSK3 (position 3 on the horizontal softkey bar) on the user interface with the text "Front USB". The softkey also shows the icon sk_usb_front.png an and is visible in the operating areas "Program", "Services" from access level 7.

### 2. Example

```
[CONNECTIONS]
ConnectionNum=1
Connection1=//ACTTCU/FRONT,1
SK_Connection1=SL_PM_SK_LOCAL_TCU_USB
SoftkeyIndex1=3
SoftkeyTextContext1=SlPmLogicalDrives
SoftkeyTextFile1=slpmdialog
SoftkeyPicture1=sk_usb_front.png
AccessProgram1=7
AccessMachine1=7
AccessServices1=7
```

A new softkey then appears on the user interface at position HSK3 (position 3 on the horizontal softkey bar) displaying the text with text ID SL_PM_SK_LOCAL_TCU_USB which it reads from file slpmdialog_XXX.qm. The softkey also shows the icon sk_usb_front.png an and is visible in the operating areas "Program", "Services" from access level 7.

## 2.3.6 Open file "logdrive.ini"

### Prerequisite

You need access level 0 to call up the file.

### Procedure

Open the file as follows:

1. Select the "Commissioning" operating area.
2. Press the "Editor" softkey (HMI only).
3. Press the "Memory Card (CF)" vertical softkey .
   The "logdrive.ini" file is in directory: /user/sinumerik/hmi/cfg
4. Press the "Input" key to open the file.

# 2.4 Configuring a channel

## 2.4.1 Configuring a channel menu

### Introduction

You can establish a connection between HMI-Embedded and the PLC via the user interface of HMI-Embedded.

Access to the channel menu configuration is granted by extending the file "netnames.ini". The file is stored on the CompactFlash Card at the following path: user/sinumerik/hmi/cfg.

### Procedure

Proceed as follows to modify the file for your application:

1. In the "Commissioning" operating area press the "HMI" sofkey.
2. Press the "Editor (HMI only)" softkey and the vertical softkey "Edit.net-names.ini". A menu with the following selection options opens:
   – "Do you want to edit your changes that have not yet been activated?"
   – "Do you want to edit the current data?"
3. Press the "Inactive file" vertical softkey. This opens an ini file where you can make your settings.
4. Press the "Close editor" softkey. Your entries are checked and the "Test completed" window pane is displayed together with the number of errors and messages.
5. Press the "Data active" softkey, this initiates a reset and configuring is completed.

## 2.4.2 "netnames.ini" file

### General

- The groups are defined in "logChanSetList" (max. 8)
- The unique names of the groups and channels (NCK) are defined in "log-ChanList"
- A unique NC name for the channels (max. 8/group) is defined in "logNCName".

## Example

Table 2-2 Operator panel front as main operator panel

```
; NETNAMES.INI, Version 04, 03.11 30.01.05
[own]
owner=              MMC_1
; Description of possible connections
[conn MMC_1]
conn_1=             NCU_1
; Description of significant net parameters
[param network]
bus=                mpi
[param MMC_1]
mmc_address = 1     ;only the first letter N or H will be used for evaluation
name=VNC            ;NCU -> connection using IP to 710s, ip=must be specified
                    ;HT8 -> connection using SPC2 to PCU with Linux, ip is ignored
                    ;VNC -> or no name, connection to 710s with internal HMI, ip is
                    ignored
[param NCU_1]
nck_address = 3     ,ip=127.0.0.1
plc_address = 2     ,ip=127.0.0.1
name = NCU710
; Description of a standard channel configuration
[chan MMC_1]
ShowChanMenu            = TRUE
DEFAULT_logChanSet =    ChannelArea1
DEFAULT_logChan =       Channel11
logChanSetList =        ChannelArea1
[ChannelArea1]
logChanList             = Channel11,Channel12,Channel13
[Channel11]
logNCName               = NCU_1
ChanNum                 = 1
[Channel12]
logNCName               = NCU_1
ChanNum                 = 2

[Channel13]
logNCName               = NCU_1
ChanNum                 = 3
```

### 2.4.3 Inserting foreign-language texts

#### General

In particular for the channel menu, the "chan.txt" file is available in several languages and stored on the CompactFlash Card in the following directories depending on the language:

- for German texts at: card/user/sinumerik/hmi/lng/deu",
- for English texts at: card/user/sinumerik/hmi/lng/eng, etc.

"chan.txt" files that do not yet contain any texts look like this:

#### Example:

```
// CP = 1252
// IDEO = NO
// ANSI = YES
// UNICODE = NO
// VERSION = $VERSION
[SWITCHES]/NATIVE
File with OEM name specifications for channel-areas and channels
//* Lengths of softkey text 2 * 9 characters */
//* For next line use %n in the string */
//* You must not change any number but only insert the texts!
*/
*/
//* Text-definitions for softkeys in channel-switch-menu */
//* Name of channel area 1 and names of channels of channel
area 1 */
3080193      "NCU_1"          //2x9 T_CHAN_AREA_1
3080194      "CHANNEL_1"      //2x9 T_CHAN_AREA_1_CHANNEL_1
3080195      "CHANNEL_2"      //2x9 T_CHAN_AREA_1_CHANNEL_2
3080196      "CHANNEL_3"      //2x9 T_CHAN_AREA_1_CHANNEL_3
3080197      ""               //2x9 T_CHAN_AREA_1_CHANNEL_4
3080198      ""               //2x9 T_CHAN_AREA_1_CHANNEL_5
3080199      ""               //2x9 T_CHAN_AREA_1_CHANNEL_6
3080200      ""               //2x9 T_CHAN_AREA_1_CHANNEL_7
3080201      ""               //2x9 T_CHAN_AREA_1_CHANNEL_8
//* Name of channel area 2 and names of channels of channel area 2 */
3080202      ""               /2x9 T_CHAN_AREA_2
3080203      ""               /2x9 T_CHAN_AREA_2_CHANNEL_1
3080204      ""               /2x9 T_CHAN_AREA_2_CHANNEL_2
3080205      ""               /2x9 T_CHAN_AREA_2_CHANNEL_3
3080206      ""               /2x9 T_CHAN_AREA_2_CHANNEL_4
3080207      ""               /2x9 T_CHAN_AREA_2_CHANNEL_6
3080208      ""               /2x9 T_CHAN_AREA_2_CHANNEL_7
```

```
3080209      ""                 /2x9 T_CHAN_AREA_2_CHANNEL_8
//* Name of channel area 3 and names of channels of channel area 3 */
3080210      ""                 /2x9 T_CHAN_AREA_3
3080211      ""                 /2x9 T_CHAN_AREA_3_CHANNEL_1
3080212      ""                 /2x9 T_CHAN_AREA_3_CHANNEL_2
3080213      ""                 /2x9 T_CHAN_AREA_3_CHANNEL_3
3080214      ""                 /2x9 T_CHAN_AREA_3_CHANNEL_4
3080215      ""                 /2x9 T_CHAN_AREA_3_CHANNEL_5
3080216      ""                 /2x9 T_CHAN_AREA_3_CHANNEL_6
3080217      ""                 /2x9 T_CHAN_AREA_3_CHANNEL_7
3080218      ""                 /2x9 T_CHAN_AREA_3_CHANNEL_8
//* Name of channel area 4 and names of channels of channel area 4 */
3080219      ""                 /2x9 T_CHAN_AREA_4
3080220      ""                 /2x9 T_CHAN_AREA_4_CHANNEL_1
3080221      ""                 /2x9 T_CHAN_AREA_4_CHANNEL_2
3080222      ""                 /2x9 T_CHAN_AREA_4_CHANNEL_3
3080223      ""                 /2x9 T_CHAN_AREA_4_CHANNEL_4
3080224      ""                 /2x9 T_CHAN_AREA_4_CHANNEL_5
3080225      ""                 /2x9 T_CHAN_AREA_4_CHANNEL_6
3080226      ""                 /2x9 T_CHAN_AREA_4_CHANNEL_7
3080227      ""                 /2x9 T_CHAN_AREA_4_CHANNEL_8
//* Name of channel area 5 and names of channels of channel area 5 */
3080228      ""                 /2x9 T_CHAN_AREA_5
3080229      ""                 /2x9 T_CHAN_AREA_5_CHANNEL_1
3080230      ""                 /2x9 T_CHAN_AREA_5_CHANNEL_2
3080231      ""                 /2x9 T_CHAN_AREA_5_CHANNEL_3
3080232      ""                 /2x9 T_CHAN_AREA_5_CHANNEL_4
3080233      ""                 /2x9 T_CHAN_AREA_5_CHANNEL_5
3080234      ""                 /2x9 T_CHAN_AREA_5_CHANNEL_6
3080235      ""                 /2x9 T_CHAN_AREA_5_CHANNEL_7
3080236      ""                 /2x9 T_CHAN_AREA_5_CHANNEL_8
//* Name of channel area 6 and names of channels of channel area 6 */
3080237      ""                 /2x9 T_CHAN_AREA_6
3080238      ""                 /2x9 T_CHAN_AREA_6_CHANNEL_1
3080239      ""                 /2x9 T_CHAN_AREA_6_CHANNEL_2
3080240      ""                 /2x9 T_CHAN_AREA_6_CHANNEL_3
3080241      ""                 /2x9 T_CHAN_AREA_6_CHANNEL_4
3080242      ""                 /2x9 T_CHAN_AREA_6_CHANNEL_5
3080243      ""                 /2x9 T_CHAN_AREA_6_CHANNEL_6
3080244      ""                 /2x9 T_CHAN_AREA_6_CHANNEL_7
3080245      ""                 /2x9 T_CHAN_AREA_6_CHANNEL_8
//* Name of channel area 7 and names of channels of channel area 7 */
3080246      ""                 /2x9 T_CHAN_AREA_7
3080247      ""                 /2x9 T_CHAN_AREA_7_CHANNEL_1
3080248      ""                 /2x9 T_CHAN_AREA_7_CHANNEL_2
```

```
3080249     ""              /2x9 T_CHAN_AREA_7_CHANNEL_3
3080250     ""              /2x9 T_CHAN_AREA_7_CHANNEL_4
3080251     ""              /2x9 T_CHAN_AREA_7_CHANNEL_5
3080252     ""              /2x9 T_CHAN_AREA_7_CHANNEL_6
3080253     ""              /2x9 T_CHAN_AREA_7_CHANNEL_7
3080254     ""              /2x9 T_CHAN_AREA_7_CHANNEL_8
//* Name of channel area 8 and names of channels of channel area 8 */
3080255     ""              /2x9 T_CHAN_AREA_8
3080256     ""              /2x9 T_CHAN_AREA_8CHANNEL_1
3080257     ""              /2x9 T_CHAN_AREA_8CHANNEL_2
3080258     ""              /2x9 T_CHAN_AREA_8CHANNEL_3
3080259     ""              /2x9 T_CHAN_AREA_8CHANNEL_4
3080260     ""              /2x9 T_CHAN_AREA_8CHANNEL_5
3080261     ""              /2x9 T_CHAN_AREA_8CHANNEL_6
3080262     ""              /2x9 T_CHAN_AREA_8CHANNEL_7
3080263     ""              /2x9 T_CHAN_AREA_8CHANNEL_8
```

You can enter your own user texts in the blank strings (s.a. ""), e.g. "NCU_1". 8 characters are possible per softkey bar; the carriage return is created with characters "%n": e.g., "NC3Turn%nChann1".

## 2.5 Setting the time and date

The following options are available:

- Setting the date and time of the PLC manually.
- To synchronize the date and time of the PLC and HMI-Embedded.

### Procedure

1. In the “Commissioning” operating area press the "PLC" → "Date/Time" softkeys.

   The date (weekday, date, year) and time are displayed in the "Current" window area of HMI-Embedded and the PLC.

   Enter the following new date in the window area "new":

   – Date: Day, Month, Year

   – Time of day: Hour, Min., Sec.

   – Synchronization time (0-99 minutes), 10 minutes is the default setting.
2. Press the "Accept" softkey to transfer the date and time of HMI-Embedded to the PLC.
3. Press softkey "Synchronous AUTO/ON" or "Synchronous AUTO/OFF" to activate/deactivate the cyclic synchronization time.
4. If you have not set automatic synchronization, you can synchronize the time immediately with softkey "Synchronous manual". The new data are displayed in the output field "current".

**Note**

Changes can only be made to the PLC operands with the appropriate password.

The set values are retained when the control is rebooted.

## 2.6 Installing/selecting language

The HMI-Embedded software is available in 6 languages as standard.

### Selecting the user interface language

1. The selection of the first and second language, set in the operating area "Commissioning" via the softkeys "HMI" -> "Language selection."
2. Select the first and second languages from the “Language Selection” menu. The following languages are available:
   - Simplified Chinese
   - German
   - English
   - Spanish
   - French
   - Italian

### Changing the language

During operation, the switchover between the set languages is performed via the "Change Language" softkey in the “Commissioning” operating area.

### Setting the boot language

The language displayed when the control boots is set via the MD9003 $MM_FIRST_LANGUAGE display machine data

## 2.7 Defining the standard directory

You can select one of the set up drives as the connection for the standard directory.

You can define the drive as follows:

- In the display machine data MD9005 $MM_PRG_DEFAULT_DIR.
- Via the user interface of HMI-Embedded.

### Setting in the display machine data.

Use the Select key to select all created drive connections (1 to max. 8).

### Setting via the user interface of HMI-Embedded

1. Press the vertical softkey "Settings".
2. Press the vertical softkey "Program Overview". All created drives will be shown.
3. Select a drive with the Select key.
4. Press the "OK" softkey. The drive is saved.

# Functions 3

## 3.1 Configuring Caps Lock

The CAPSLOCK function ensures that text entered via an external keyboard is always entered in upper case and not in lower case.

The key behavior is set using the display machine data MD9009 $MM_KEYBOARD_STATE:

- 0: CAPSLOCK off
- 1: Setting not relevant!
- 2: CAPSLOCK on (default setting)

If the display machine data is set to “CAPSLOCK” you can switch between upper and lower case with key combination “Ctrl” and the Shift key.

If the machine data is changed, the system must be rebooted.

## 3.2 Activating screen darkening

If a screen with high contrast is displayed unchanged for longer than an hour, the screen must be switched dark (screensaver) in order to protect the TFT display against so-called "freezing" of the last displayed screen.

### Activation via PLC

You can activate the screensaver via the PLC using DB19 DBX0.1. The PLC has priority over the entry in the display machine data.

**References:** Function Manual Basic Functions, Chapter: Various Interface Signals

### Activating via the machine data

The duration (max. 60 minutes), after which the screen is automatically darkened if no key is pressed on the keyboard, can be specified using the display machine data MD9006 $MM_DISPLAY_BLACK_TIME.
In this display machine data, the "Screensaver" function can be started with the following entry:

1: < time in minutes, e.g. 1 >
60: < time in minutes, e.g., 60 >

0: Deactivating the function (standard setting)

# 3.3 Switching the calculator help on/off

## Procedure

You can activate/deactivate the calculator help function using the display machine data MD9991 $MM_HMI_HELP_SYSTEMS.

- Bit 0 = 0 help display not active
- Bit 0 = 1 help display active (default setting)

# 3.4 Setting the Editor

The editor is available in the program and commissioning operating areas.

You can define editor settings as follows:

- In the display machine data MD9460 $MM_PROGRAM_SETTINGS.
- Via the user interface of HMI-Embedded.

## Settings in the display machine data

| | | |
|---|---|---|
| Bit 2 | = 0 | No automatic enabling for programs |
| | = 1 | Automatic enabling for programs created using the "New" softkey. |
| Bit 4 | = 0 | Line feed will be displayed in an opened file in the edit field. |
| | = 1 | The LF (line feed) icon will be hidden in an opened file in the edit field. |
| Bit 5 | = 0 | Hidden lines marked with "*HD" (hidden) will be displayed. |
| | = 1 | Hidden lines marked with "*HD" (hidden) remain hidden. |
| Bit 6 | = 0 | Active program disabled for editing. |
| | = 1 | Active program enabled for editing. |
| Bit 7 | = 0 | Perform line test |
| | = 1 | Disable the line test of the cycle support for programs with a file size set in the display machine data MD9464 $MM_MAX_PROGRAMM_SIZE_CHECK. The size is entered in KBytes. New lines can also be added within the G-Code created by the "Wizard" or the geometry processor. |

The settings in the Program/Commissioning operating areas are retained after a reset.

### Setting via the user interface of HMI-Embedded

Please proceed as follows:

1. Open a program for editing in the "Program" operating area.
2. Press the "Settings" horizontal softkey and the "Set Editor" vertical softkey to obtain the setting screen form.
   You can mark the following functions in the field with "yes".
   – Release new programs
   – Hide LF in the program
   – Display hidden lines in the program
   – Enable active programs for editing.
   – Special handling for moldmaking file size specification in kBytes

## 3.5 Outputting the acknowledgement icon for PLC alarm

Using the display machine data MD9055 $MM_PLC_ALARM_PICTURE activate or deactivate the following icons:

- Value: -1 no icon display
- Value: 0 Icon labeled "PLC" is displayed.
- Value: 1 "Cancel" icon is displayed.

This functionality refers only to PLC alarms, not to PLC messages

## 3.6 Alarm display

If several alarms are pending at the same time, only the last alarm is displayed in the alarm and message line. If you want to display the pending NCK, PLC or HMI alarms alternatively, you can set this in the display machine data MD9056 $MM_ALARM_ROTATION_CYCLE.

The following settings can be made:

| | |
|---|---|
| < 500: | No change of the alarm display, only the last alarm is displayed |
| 500 – 10000: | Display duration of an alarm in milliseconds |

If a valid display duration is set, each alarm is displayed for this time before being replaced by the next alarm.

Messages will not be displayed alternatively.

## 3.7 Inch/metric switchover

The availability of the softkey, and thus the associated functionality, can be configured using the general NCK machine data MD10260.

In the "Machine" operating area, you can switch between the inches measuring system and the metric measuring system via a softkey. If you want to display the "Switch to inch" or "Switch to mm" softkey, set the machine data as follows:

MD10260 $MN_CONVERT_SCALING_SYSTEM ≠ 0

### References

Function Manual, Basic Functions; Axis Monitoring, Protection Zones (A3)

Function Manual, Basic Functions; Velocities, Setpoint-Actual Value Systems, Closed-Loop Control (G2)

## 3.8 Geometry processor

You can define text output at the end of a contour and representation of the contour elements in one of two ways:

- Via the user interface of HMI-Embedded.
- In the display machine data MD9460 $MM_PROGRAM_SETTINGS.

### Setting via the interface

1. Press the "Contour settings" vertical softkey. The window pane: "Free contour programming settings" will be opened.
2. Enter after "Text output at the end of the contour programming" the text that is to appear after each contour, e.g. "Contour end".
3. You can select/deselect other settings:
   – Technology specifications: "Rotate"
   – Graphically show softkey for geometry elements.

### Settings in the display machine data

Bit 0 = 1 Turning technology (contour support).

Bit 3 = 0 The softkeys for selecting the contour elements are assigned text.

= 1 The softkeys for selecting the contour elements are assigned icons.

## 3.9 Tool fine offset

The limit values of the tool fine correction are contained in the display machine data MD9450 $MM_WRITE_TOA_FINE_LIMIT.

When entering the tool wear fine, the difference between the previous value and the new value may not exceed the limit entered here.

The MD9450 can only be changed if a permissible protection level
has been entered in the display machine data MD9202 $MM_USER_CLASS_TOA_WEAR.

### References

Function Manual Basic Functions; various NC/PLC interface signals and functions (A2)

Function Manual Tool Manager

CNC Commissioning Manual: ShopMill

CNC Commissioning Manual: ShopTurn

## 3.10 Set tool offset to active immediately

You can specify that the tool offset is to take effect immediately in display machine data MD9440 $MM_ACTIVATE_SEL_USER_DATA.

The part program is in the "Reset" or "Stop" state.

To ensure that the tool offset is not reset on "Reset", the channel-specific machine data MD20110 $MC_RESET_MODE_MASK bit 0 = reset mode must be appropriately set.

## 3.11 Selecting tools with D number

You activate the type of D number programming using the general machine data MD18102 $MN_TYPE_OF_CUTTING_EDGE.

- Value = 0 NCK manages the T and D numbers = default setting
- Value = 1 Only tools with the D number structure can be selected.

The D number can be assigned just once per tool, i.e., each D number stands for one and no more than one offset data set.

### References

Function Manual Tool Manager: Tool offset

## 3.12 Analog spindles

The spindle load from 0 to 100% can be displayed in the spindle window for up to two analog spindles.

Set the following bytes in DB 19:

- Analog spindle: DB19.DBB 6
- Analog spindle: DB19.DBB 7

## 3.13 Use a workpiece template

You can use workpiece templates that will be used as templates for similar programs. Create a workpiece called _TEMPL_, copy it, and then give it a new name.

### Example

Suppose you want to create a workpiece template with the content "Daten.ini" and "TEST.MPF": Workpiece: _TEMPL_

_TEMPL_.MPF

DATEN.INI

TEST.MPF

### Procedure

1. Press the softkey "New".
2. Assign a new name, e.g. "AXIS"
3. The new workpiece "ACHSE" is created with the following files:
   ACHSE.MPF
   DATEN.INI
   TEST.MPF

# 3.14 Zero point shift

## 3.14.1 Access protection for basic softkey offset

### "Parameters operating area

Using the display machine data MD9248 $MM_USER_CLASS_BASE_ZERO_OFF_MA you can set from which access level the "WO base" softkey is displayed in the "Scratching" screen in the Machine operating area and/or G500 can be entered in the work offset field.

At the same time, the base frames are also shown/hidden in the "Work offset" window and in the "Active WO + offsets" window.

---

**Note**

The display machine data MD9210 $MM USER_CLASS_WRITE_ZOA (write coarse offset) has no effect with “active zero offset” in the "Parameters" operating area.

---

### Machine operating area

Using the display machine data MD9247 $MM_USER_CLASS_BASE_ZERO_OFF_PA you can set from which access level the "Base WO" softkey is displayed in the "Work offset" window in the "Parameters" operating area.

## 3.14.2 Fine zero offset and base offset

### Zero point shift

This function refers to the offset in the screen form "Settable work offset" and "Base work offset". Another column for the fine offset is inserted alongside the column for entering the offsets.

The inputs will be validated against the following display machine data:

MD9203 $MM_USER_CLASS_WRITE_FINE (defines the access level)

MD9451 $MM_WRITE_ZOA_FINE_LIMIT (possible input values)

The base work offset (base WO) is displayed like a settable work offset and can be selected and modified using the "Base WO" softkey in the screen form "Overview of work offsets".

### Fine offset

The fine offset for all settable FRAMES and the basic frame is activated via the following general machine data:

MD18600 $MM_FRAME_FINE_TRANS = 1

### References

Function Manual, Basic Functions; Axes, Coordinate Systems, Frames (K2)

## 3.14.3 Preset actual value memory, preset, scratching

### General

The behavior of the scratching and preset actual value memory, PRESET functions is influenced by a number of machine data. The selected settings affect operation, e.g. display of softkeys and values, and how they are stored.

With System Frames two variants are now available for the functions. The variants are differentiated using the following channel-specific machine data
MD28082 $MC_SYSTEM_FRAME_MASK:

| Bit | |
|---|---|
| 0 | Preset actual value memory, scratching |
| 1 | External zero offset |
| 2 | TCARR, PAROT |
| 3 | TOROT, TOFRAME |
| 4 | Workpiece reference points |
| 5 | Cycles |
| 6 | Frame transformation |

Variant 1 **without**system frame:

The machine data does not exist (older NCK software versions) or bit 0 for system frame for preset actual value memory, scratching is not set.

Variant 2 **with**system frame:

Bit 0 for system frame for preset actual value memory, scratching is set in the machine data.

## PRESET function

In the "JOG" mode you specify the following in the display machine data MD9422 $MM_MA_PRESET_MODE, function PRESET/basis offset:

| Bit | |
|---|---|
| 0 | No softkey |
| 1 | The old PRESET in the "Machine" operating area (default). |
| 2 | Preset actual value memory<br>NCK **without** system frame: This setting is only possible when G500 is active in the base offset, otherwise an error message appears.<br>NCK **with** system frame: Setting in the system frame possible.<br>The basic frame is not used with G500 in the system frame. |
| 3 | Preset actual value memory<br>NCK with/without system frame. Setting possible in the currently active frame. |

**Note**

To hide the PRESET, actual value setting and scratching softkeys, the display machine data MD9220 $MM_USER_CLASS_PRESET can also be correspondingly set.

## Preset actual value memory

Preset actual value memory **with** system frame

| | |
|---|---|
| G500 active | The values are written to the system frame. |
| G500 not active | An error message is output. |

Preset actual value memory **without** system frame

| | |
|---|---|
| G500 active | (settable frames reset)<br>The values are written to the set basic frame depending on the entry in the display machine data.<br>MD9245 $MM_ MA_PRESET_FRAMEIDX = Index of the basic frame |

### Scratching

Scratching **with** system frame

| | |
|---|---|
| G500 active | |
| | The values are written to the system frame |
| G500 not active | |
| | The values are written in the currently active frame. |

Only select the tool located in the spindle.

If there is no tool in the spindle, you can choose any tool you want.

Scratching **without** system frame

| | |
|---|---|
| G500 active | (settable frames reset) |
| | The values are written to the set basic frame depending on the entry in the display machine data. |
| | MD9245 $MM_ MA_PRESET_FRAMEIDX = Index of the basic frame |
| G500 not active | |
| | The values are written in the currently active frame. |

You can select any tool you wish.

## 3.14.4 Hiding machine axes

### General

You can hide axes in the "Machine" area using the channel-specific machine data MD20098 $MC_DISPLAY_AXIS. All axes are displayed by default. The following settings can be made:

| Bit | |
|---|---|
| 16 = 1 | Display the machine axis in the actual-value windows (default setting). |
| 16 = 0 | Hide the machine axis in the actual-value windows. |
| 17 | n.c. |
| 18 = 1 | Display the machine axis in the base offset window (default). |
| 18 = 0 | Hide the machine axis in the base offset window. |
| 19 = 1 | Display the machine axis in the handwheel selection window (default). |
| 19 = 0 | Hide the machine axis in the handwheel selection window. |

### Special cases

Reference point approach and Safety Integrated

The channel-specific machine data MD20098 $MC_DISPLAY_AXIS is not evaluated in the displays for the "REF" referencing mode, i.e. the display, "Axis reference point and acknowledge safe position". All machine axes are always displayed.

You cannot change the values for the base work offset without the appropriate access right. Enter access right 7 as a standard value in the display machine data MD9247 and MD9248.

### 3.14.5 Displaying geometry axes

To display geometry axes at the first position (e.g. before the auxiliary axes), set the display machine data MD9421 $MM_MA_AXES_SHOW_GEO_FIRST as follows:

1: Display geometry axes at the first position

0: Do not display geometry axes at the first position (default setting)

## 3.15 Specifying the machine and rotary axis position

### Singularities with special kinematics and robots

When the cartesian point-to-point method (PTP) in workpiece coordinates is used, the position of machine axes is ambiguous at certain points. In order to define these singularities without ambiguity, the position of the machine STAT and the position of the axes TU are defined in addition to the workpiece coordinates for this type of transformation (e.g., 5-axis transformation).

### Setting the numerical basis

The STAT and TU values are displayed in the axis screens (actual-value window, broad actual-value window with feed and zoom display) and entered in the MDA editor during the teach-in procedure. Whether the values in STAT and TU are displayed in binary, decimal or hexadecimal format is set via the following display machine data:

| | |
|---|---|
| MD9242 $MM_MA_STAT_DISPLAY_BAS E | Numerical basis for display of moving joint STAT |
| MD9243 $MM_MA_TU_DISPLAY_BASE | Numerical basis for display of rotary axis position TU |

Possible settings:

| | |
|---|---|
| 02 | Display as binary value |
| 10 | Display as decimal value |
| 16 | Display as hexadecimal value |

These settings apply to both the actual-value window and the editor window.

## Example

In the NC program the movement of axes from X120 Y20 Z-50 to X200 Y200 Z-120 looks like this in binary display:

X=120 Y=20 Z=-50 STAT='B010' TU'B011101'

X=200 Y200 Z=-120 STAT'B110' TU='B011111'

## Showing/hiding STAT and TU

The STAT and TU values are displayed in the axis screens (actual-value windows) and in teach mode (insert block).

You can choose whether or not to show the STAT and TU values by setting the following display machine data:

STAT and TU are not displayed:

| | | |
|---|---|---|
| MD9242 $MM_MA_STAT_DISPLAY_BASE | = 0 | of articulated position STAT |
| MD9243 $MM_MA_TU_DISPLAY_BASE | = 0 | of rotary axis position TU |

STAT and TU are displayed:

| | | |
|---|---|---|
| MD9242 $MM_MA_STAT_DISPLAY_BASE | = 1 | of articulated position STAT |
| MD9243 $MM_MA_TU_DISPLAY_BASE | = 1 | of rotary axis position TU |

# 3.16 Program simulation

## 3.16.1 Turning simulation and program test

### Prerequisite

PLC signal DB19.DBB20.6 is set to 1 when simulation is activated. This signal can be evaluated for the purpose of switching to program test mode or canceling axis or controller enabling signals (to prevent axes from moving during simulation runs). The signal is reset to 0 when simulation is terminated.

In order to simulate certain operations during a dry run even though the axis/controller pulses have been disabled, the transfer of status "Ext. pulse disable active, terminal 663 open" to the NC has to be prevented. The machine manufacturer can achieve this by evaluating the above-mentioned PLC signal and activating the dry run simultaneously with NC Start.

If the drive machine data MD1012 $MN_FUNC_SWITCH, Bit 2 is set to 0, the status "Ext. pulse disable active, terminal 663 open" is not transferred to the NC.

### References

Function Manual Drive Functions: Operational messages/alarm responses

## 3.16.2 Turning/milling simulation

Using the display-machine data MD9020 $MM_TECHNOLOGY, you define the basic configuration for the free contour programming of turning/milling.

1: Turning machine configuration

≠ 1: Milling machine configuration

## 3.16.3 Milling simulation 3D graphic

This function is a software option named "Milling Simulation", with order number: 6FC5253-0AE02-0AA0.

If the option is set, the "3D Simulation" softkey will appear and can be used to select the function.

Operation of milling machine simulation is described in:

**References:** Operator Manual HMI-Embedded

# 3.17 Measuring cycle support in the G code editor

## General

In HMI-Embedded, you can integrate screen forms in the G code editor that provide support when programming measuring cycles. These cycles can then be recompiled automatically.

The "measuring cycles" function is a software option with order no. 6FC520-0BX00-0AB0.

For further information about measuring cycle support, please refer to:

**References:** Program Manual Measuring Cycles

## Procedure

Proceed as follows when installing the function:

1. Import file "common.com" to the following directory on the CompactFlash card:
   /user/sinumerik/hmi/cfg
   .
2. Open the file in the editor.
3. Remove the semicolon ";" in front of the following lines:
   – ;SC326 = AEDITOR.COM ;Turning measuring cycles (horizontal softkey 6 on the extended softkey bar in the G code editor)
   – ;SC327 = AEDITOR.COM ;Milling measuring cycles (horizontal softkey 7 on the extended softkey bar in the G code editor).
   This is how you establish the connection between the softkeys used to call measuring-cycle support and the configuration file for these support screens.
4. Remove the semicolon ";" in front of the following line:
   – ;SC617 = STARTUP.COM ;Commissioning operating area (horizontal softkey 7 on the extended softkey bar in the G code editor).
   This is how you change the properties of the measuring-cycle support in the Commissioning operating area.

You can change other properties of the measuring cycle support from the user interface of HMI-Embedded:
In the "Commissioning" operating area, press the expansion key ">" and the softkey "Measuring cycles".

## 3.18 Expanding user displays from the operator interface

"Expand user interface" is implemented with an interpreter and configuration files that describe the user interfaces. The "Expand user interface" tools can be used to create user interfaces, which display functional expansions designed by the machine manufacturer or end user, or simply to implement your own screen form layout.

User interfaces configured by Siemens or non-Siemens machine manufacturers can be modified or replaced. The interpreter is available for HMI-Embedded, ShopMill and ShopTurn on the NCU.

Part programs, for example, can be edited on user interfaces created by users. Screen forms can be created directly on the control system.

The application diskette and PaintShopPro (http://www.jasc.com) are required to create graphics and images.

The tool box supplied contains configuration examples for new screen forms. You can also use these examples as a template for creating your own forms.

You cannot call user-defined dialog windows simultaneously in different channels, i.e., you can only use the MMC command with a 1:1 link.

The exact procedure is described in the following submanual:

### References

Expand User Interface (BE1)

## 3.19 Set Wide Display

### Introduction

The HMI-Embedded program is displayed on a large screen as "Wide Display" in such a way that an area for OEM applications remains free in the upper section.

### Prerequisite

To activate the "Wide Display" function, the following prerequisites apply:

- In conjunction with JobShop on PCU 50.
- The screen resolution must be at least 1280*1024 pixels.
- The "UpperScreen" function must be set to TRUE in the configuration file REGIE.INI.
- The sequence control stores the start position of the HMI program in the HMI_Start entry in section [CONTROL] in the MMC.INI file.
- OEM applications must be regenerated with the current OEM package and the expansion of procedural control.

### Activating the function

REGIE.INI configuration file section:

```
 [UpperScreen]
; Upper screen area above HMI-Wide-Screen.
; The upper screen area is managed by an independent native windows
; application independently of the REGIE management.
; The upper screen area has an independent softkey area.
; This upper screen softkey area is available only
; on special OEM operator panels.
; Activate this function here
UpperScreen = TRUE
; Announce main window of upper screen,
; to transfer the softkeys from Regie.
; WindowName
UpperScreenWindowName = ""
; Window ClassName
UpperScreenClassName = ""
```

### Operating the HMI program and OEM application

When switching the operation between HMI program and OEM application, use the window shift key <NEXT WINDOW> or simply click.

The OEM application is either operated by pressing an OEM key (refer below) or with a mouse click in the main window. All other keyboard entries will now appear in this active window until one of the function keys (F1-F10, etc.) recognized by the HMI program is detected. The HMI program is then active again.

The following key combinations are reserved:

<Ctrl+1>, <Ctrl+2>, <Ctrl+3>, <Ctrl+6>, <Ctrl+7>, <Ctrl+8>

## 3.20 Coordinate systems

### 3.20.1 Actual value display

Set the following in the display machine data MD9424 $MM_MA_COORDINATE_SYSTEM:

- The position of the workpiece coordinate system (programmed position, corresponds to the default setting) or
- show the holding position of the active tool relative to the workpiece zero in the actual-value display.

| | |
|---|---|
| MD9424 = 0 | Display in workpiece coordinate system, WCS (default) |
| MD9424 = 1 | Display in the settable zero system, SZS (fixture position of the active tool) |

**References:**

Function Manual, Basic Functions; Axes, Coordinate Systems, Frames (K2)

## 3.20.2 Position of coordinate system

### Milling

The position of the coordinate system for milling is set in display machine data MD9650 $MM_CMM_POS_COORDINATE_SYSTEM.

Table 3-1 Machine data 9650

| 9650 MD number | CMM_POS_COORDINATE_SYSTEM<br>Position of coordinate system | |
|---|---|---|
| Default setting 0 | Min. input limit: 0 | Units: - |
| Data type BYTE | | Valid as of SW release:<br>SW 05.01.13 |
| Meaning | With this machine data you adapt the coordinate system of the user interface to the coordinate system of the machine.<br>The coordinate system can assume the following positions. | |

### Technology

The position of the coordinate system for the technology is set in display machine data MD9610 $MM_CTM_POS_COORDINATE_SYSTEM.

### References

Function Manual, Basic Functions, Mode Group, Channel, Program Operation, Reset Behavior (K1)

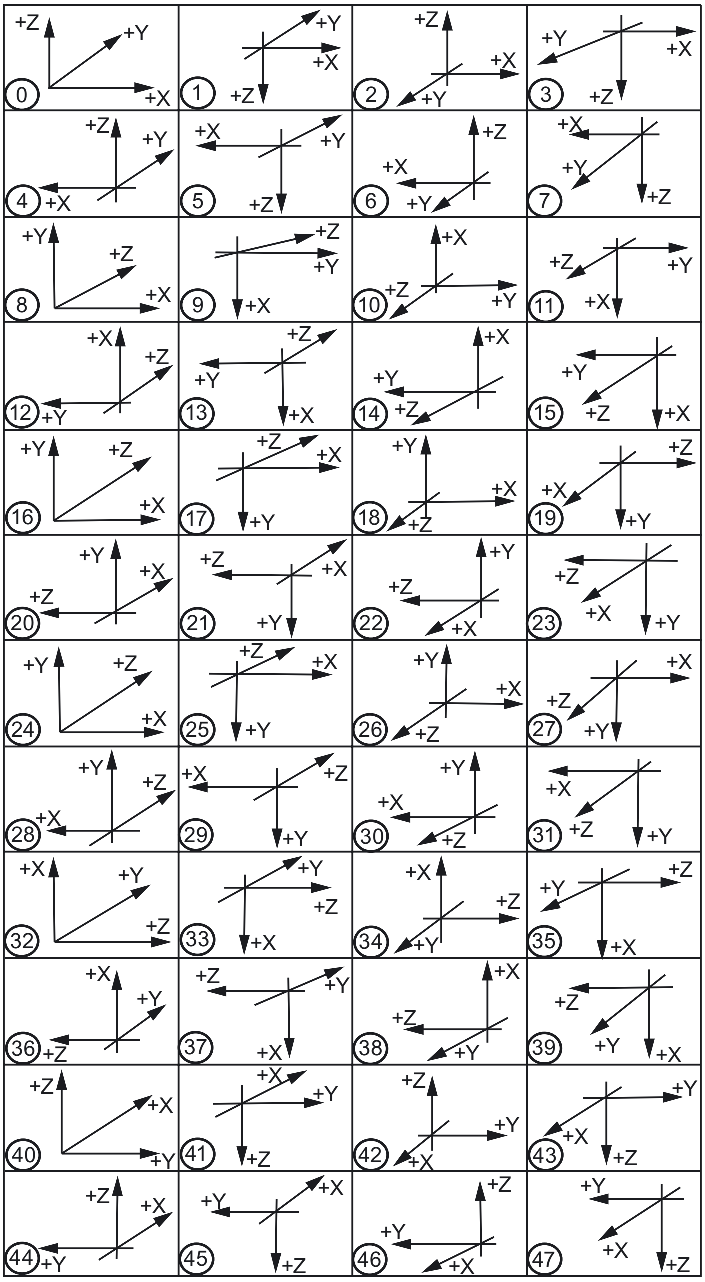



Figure 3-1 Coordinate systems

# 3.21 Available display machine data

Please refer to the following documentation, available on DOCONCD, for descriptions of all machine-data displays:

**References:** Detailed Machine Data Description

## Legend

**Target systems:**

Adv: HMI Advanced

Emb: HMI-Embedded

HT6: Handheld terminal 6

OP30: Operator panel OP 030

MT: ManualTurn

SM: ShopMill

ST: ShopTurn

**Filter:**

H01: ShopMill

H02: ShopTurn

H03: ManualTurn

H04: Access levels

H05: Standard machine

| **MD number** | **Name** | | | **Filter** | **Cross reference:** |
|---|---|---|---|---|---|
| Unit | Name | | | Data type | Read/<br>write protection |
| PLC | Default value | Min. value | Max. value | Activation | SW version |

| | | | | | |
|---|---|---|---|---|---|
| **9000** | **LCD_CONTRAST** | | | H05 | **Cross reference:**<br>A2 |
| - | Contrast | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7,<br>Emb: 7 | 0 | 15 | Power On | - |
| **9001** | **DISPLAY_TYPE** | | | H05 | **Cross reference:**<br>A2 |
| - | Type of operator panel<br>9002 external monitor | | | BYTE | 0/0 |
| OP30, Adv, Emb | OP30: 1, Adv: 1,<br>Emb: 1 | 0 | 2 | Power On | |
| **9002** | **DISPLAY_MODE** | | | | **Cross reference:**<br>A2 |
| - | (1: black and white, 2: color) | | | BYTE | 3/4 |
| OP30, | OP30: 0, | 0 | 2 | Power On | 1 |

| | | | | | |
|---|---|---|---|---|---|
| 9003 | FIRST_LANGUAGE | | | H05 | Cross reference: A2 |
| - | Foreground language | | | BYTE | 3/4 |
| OP30, Emb | OP30: 1, Emb: 1 | 1 | 2 | Power On | 1.1 |
| 9004 | DISPLAY_RESOLUTION | | | H05 | Cross reference: A2 |
| - | Display resolution | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 3, Adv: 3, Emb: 3 | 0 | 5 | Power On | - |
| 9005 | PRG_DEFAULT_DIR | | | H05 | Cross reference: A2 |
| - | Basic setting Program directory | | | BYTE | 3/4 |
| OP30, Emb | OP30: 1, Emb: 1 | 1 | 5 | IMMEDIATELY | - |
| 9006 | DISPLAY_BLACK_TIME | | | H05 | Cross reference: A2 |
| - | Time for screen darkening | | | BYTE | 3/4 |
| OP30, Emb | OP30: 15, Emb: 15 | 0 | 60 | Power On | SW 2 |
| 9007 | TABULATOR_SIZE | | | H05 | Cross reference: A2 |
| - | Tabulator length | | | BYTE | 3/4 |
| OP30, Emb | OP30: 4, Emb: 4 | 0 | 30 | IMMEDIATELY | SW 2 |
| 9008 | KEYBOARD_TYPE | | | H05 | Cross reference: A2 |
| - | Keyboard type<br>(0: OP, 1: MFII/QWERTY) | | | BYTE | 3/4 |
| Adv, Emb | Adv: 0, Emb: 0 | 0 | 1 | Power On | SW3.6 |
| 9009 | KEYBOARD_STATE | | | H05 | Cross reference: A2 |
| - | Keyboard shift behavior during booting<br>(0: single, 2: CAPSLOCK) | | | BYTE | 3/4 |
| Adv, Emb | Adv: 0, Emb: 2 | 0 | 2 | Power On | SW3.6 |

| | | | | | |
|---|---|---|---|---|---|
| 9011 | DISPLAY_RESOLUTION_INCH | | | H05 | Cross reference: A2 |
| - | Display resolution for INCH system of measurement | | | BYTE | 3/4 |
| Adv, Emb | Adv: 4, Emb: 4 | 0 | 6 | Power On | SW5.1 |
| 9012 | ACTION_LOG_MODE | | | H05 | Cross reference: IM2, IM4 |
| - | Set action mode for action log | | | INTEGER | 1/1 |
| Adv, Emb | Adv: 0xFE, Emb: 254 | 0 | 0xFFFF | Power On | SW5.2 |
| 9013 | SYS_CLOCK_SYNC_TIME | | | H05 | Cross reference: IM1 |
| - | Synchronization time MMC/HMI time with PLC<br>Synchr. time for HMI/PLC time | | | REAL | 0/0 |
| Emb | Emb: 0 | 0 | 199 | Power On | SW5.3 |

| | | | | | |
|---|---|---|---|---|---|
| 9014 | USE_CHANNEL_DISPLAY_DATA | | | H05 | Cross reference: FBT, FBSP, EMB, ADV |
| - | Use channel-specific display MDs | | | INTEGER | 3/4 |
| Adv, Emb | Adv: 0, Emb: 0 | 0 | 1 | IMMEDIATELY | SW6.3 |
| 9016 | SWITCH_TO_AREA | | | H05 | Cross reference: IAM, BE1 |
| - | Default boot menu can be selected | | | INTEGER | 3/4 |
| OP30, Emb | OP30: 20, Emb: -1 | -1 | 10000 | Power On | SW6.3 |
| 9020 | TECHNOLOGY | | | H05 | Cross reference: A2, FBT |
| - | Technology for NC prog. and simulation<br>0: No specific assignment<br>1: Turning machine configuration<br><>1: Milling machine configuration | | | BYTE | 3/4 |
| Adv, Emb | Adv: 0, Emb: 1 | 0 | 2 | Power On | SW6, ST SW6.1 |
| 9021 | LAYOUT_MODE | | | H05 | Cross reference: |
| - | Design of the user interface | | | BYTE | 3/4 |
| Adv, Emb, HT 6 | Adv: 1, Emb: 0,<br>HT 6:1 | 0 | 1 | Power On | SW6.3 |
| 9025 | DISPLAY_BACKLIGHT | | | | Cross reference: IM2 |
| - | Display backlight brightness level | | | BYTE | 3/4 |
| | HT6: 15 | 0 | 31 | Power On | SW5.3 |
| 9026 | TEACH_MODE | | | | Cross reference: IM2 |
| - | Teach mode to be activated | | | REAL | 3/4 |
| | HT6: 1 | *** | *** | Power On | SW5.3 |
| 9027 | NUM_AX_SEL | | | | Cross reference: IM2 |
| - | Number of axis groups for traversing keys | | | REAL | 3/4 |
| | HT6: 0 | 0 | 4 | Power On | SW5.3 |
| 9030 | EXPONENT_LIMIT | | | H05 | Cross reference: A2 |
| - | Number of digits for display without exponent | | | BYTE | 3/4 |
| Emb | Emb: 6 | 0 | 20 | Power On | SW5.1 |
| 9031 | EXPONENT_SCIENCE | | | H05 | Cross reference: A2 |
| - | Exponent in the technical representation | | | BYTE | 3/4 |
| Emb | Emb: 1 | 0 | 1 | Power On | SW5.1 |
| 9032 | HMI_MONITOR | | | H05 | Cross reference: FBT, FBSP, EMB, ADV |
| - | Determine PLC data for HMI monitor information | | | STRING | 2/4 |
| Adv, Emb | Adv: '', Emb: 0 | *** | *** | Power On | SW6.2 |

| | | | | | |
|---|---|---|---|---|---|
| 9050 | STARTUP_LOGO | | | H05 | Cross reference: FBT, FBSP, EMB, ADV |
| - | Activate OEM boot screen | | | BYTE | 1/4 |
| Adv, Emb | Adv: 0, Emb: 0 | 0 | 1 | Power On | SW6.2 |
| 9052 | SHOW_CHANNEL_SPANNING_STATE | | | H05 | Cross reference: FBT, FBSP |
| - | Change cross-channel status display | | | BYTE | 2/4 |
| Adv, Emb | Adv: 0, Emb: 0 | 0 | 1 | Power On | SW6.3 |
| 9053 | PLC_SYMBOL_SORT | | | H05 | Cross reference: |
| - | Sorting algorithm for PLC icons | | | INTEGER | 3/4 |
| Adv, Emb | Adv: 0, Emb: 0 | 0 | 4 | IMMEDIATELY | SW6.3 |
| 9054 | PLC_SYMBOL_FILTER | | | H05 | Cross reference: |
| - | Filter settings for PLC icons | | | REAL | 3/4 |
| Adv, Emb | Adv: 0, Emb: 0 | 0 | 0xFFFF | IMMEDIATELY | SW6.3 |
| 9055 | PLC_ALARM_PICTURE | | | H05 | Cross reference: |
| - | Acknowledgement icon selection for PLC alarms | | | INTEGER | 3/4 |
| Adv, Emb | Adv: 1, Emb: 1 | -1 | 1 | Power On | SW6.3 |
| 9056 | ALARM_ROTATION_CYCLE | | | H05 | Cross reference: |
| - | Rotation cycle time for the alarm display | | | INTEGER | 3/4 |
| Emb | Emb: 0 | 0 | 10000 | IMMEDIATELY | SW6.4 |
| 9180 | USER_CLASS_READ_TCARR | | | H04, H05 | Cross reference: A2 |
| - | Protection level read toolholder offsets | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | IMMEDIATELY | SW6.1 |
| 9181 | USER_CLASS_WRITE_TCARR | | | H04, H05 | Cross reference: A2 |
| - | Protection level write toolholder offsets | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | IMMEDIATELY | SW6.1 |
| 9182 | USER_CLASS_INCH_METRIC | | | H04, H05 | Cross reference: EMB |
| - | Protection level inch/metric switchover | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | IMMEDIATELY | SW6.2 |
| 9183 | USER_WRITE_TOOLFRAME | | | H04, H05 | Cross reference: |
| - | Protection level write toolholder | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 0 | 0 | 7 | IMMEDIATELY | SW6.4 |
| 9184 | USER_WRITE_PARTFRAME | | | H04, H05 | Cross reference: |
| - | Protection level write tool reference point | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 0 | 0 | 7 | IMMEDIATELY | SW6.4 |
| 9185 | USER_WRITE_WPFRAME | | | H04, H05 | Cross reference: |
| - | Protection level write workpiece reference point | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 0 | 0 | 7 | IMMEDIATELY | SW6.4 |

| | | | | | |
|---|---|---|---|---|---|
| 9186 | USER_WRITE_CYCFRAME | | | H04, H05 | Cross reference: |
| - | Protection level write cycle frame | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 0 | 0 | 7 | IMMEDIATELY | SW6.4 |
| 9187 | USER_WRITE_TRAFRAME | | | H04, H05 | Cross reference: |
| - | Protection level write transformation frame | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 0 | 0 | 7 | IMMEDIATELY | SW6.4 |
| 9188 | USER_WRITE_EXTFRAME | | | H04, H05 | Cross reference: |
| - | Protection level Protection level write external WO | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 0 | 0 | 7 | IMMEDIATELY | SW6.4 |
| 9200 | USER_CLASS_READ_TOA | | | H04, H05 | Cross reference: A2 |
| - | Protection level read tool offsets | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| 9201 | USER_CLASS_WRITE_TOA_GEO | | | H04, H05 | Cross reference: A2 |
| - | Protection level write tool geometry | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| 9202 | USER_CLASS_WRITE_TOA_WEAR | | | H04, H05 | Cross reference: A2 |
| - | Protection level write tool wear data | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| 9203 | USER_CLASS_WRITE_FINE | | | H04, H05 | Cross reference: A2 |
| - | Protection level fine | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| 9206 | USER_CLASS_WRITE_TOA_SUPVIS | | | H04, H05 | Cross reference: A2 |
| - | Protection level tool monitoring Change limit values fine | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 0 | 0 | 7 | IMMEDIATELY | SW5 |
| 9209 | USER_CLASS_WRITE_TOA_ADAPT | | | H04, H05 | Cross reference: A2 |
| - | Protection level write tool adapter data | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | SW5 |
| 9210 | USER_CLASS_WRITE_ZOA | | | H04, H05 | Cross reference: A2 |
| - | Protection level write settable WO | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| 9211 | USER_CLASS_READ_GUD_LUD | | | H04, H05 | Cross reference: A2 |
| - | Protection level read user variables | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | SW6.1 |

| | | | | | |
|---|---|---|---|---|---|
| **9213** | **USER_CLASS_OVERSTORE_HIGH** | | | H04, H05 | **Cross reference:** A2 |
| - | Protection level extended overstore | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| **9214** | **USER_CLASS_WRITE_PRG_CONDIT** | | | H04, H05 | **Cross reference:** A2 |
| - | Protection level program control | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| **9215** | **USER_CLASS_WRITE_SEA** | | | H04, H05 | **Cross reference:** A2 |
| - | Protection level write setting data | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| **9216** | **USER_CLASS_READ_PROGRAM** | | | H04, H05 | **Cross reference:** A2 |
| - | Protection level read part program | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| **9217** | **USER_CLASS_WRITE_PROGRAM** | | | H04, H05 | **Cross reference:** A2 |
| - | Protection level write parts program | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| **9218** | **USER_CLASS_SELECT_PROGRAM** | | | H04, H05 | **Cross reference:** A2 |
| - | Protection level 'Program selection' | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| **9219** | **USER_CLASS_TEACH_IN** | | | H04, H05 | **Cross reference:** A2 |
| - | Protection level TEACH IN | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| **9220** | **USER_CLASS_PRESET** | | | H04, H05 | **Cross reference:** A2 |
| - | Protection level PRESET | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| **9221** | **USER_CLASS_CLEAR_RPA** | | | H04, H05 | **Cross reference:** A2 |
| - | Protection level delete R parameters | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |

| | | | | | |
|---|---|---|---|---|---|
| 9222 | USER_CLASS_WRITE_RPA | | | H04, H05 | Cross reference: A2 |
| - | Protection level write R parameters | | | BYTE | 3/4 |
| OP30, Adv, Emb | OP30: 7, Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| 9223 | USER_CLASS_SET_V24 | | | H04, H05 | Cross reference: A2 |
| - | Protection level V24 interface parameterization | | | BYTE | 3/4 |
| OP30, Emb | OP30: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| 9224 | USER_CLASS_READ_IN | | | H04, H05 | Cross reference: A2 |
| - | Protection level read in data | | | BYTE | 3/4 |
| OP30, Emb | OP30: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | - |
| 9225 | USER_CLASS_READ_CST | | | H04, H05 | Cross reference: A2 |
| - | Protection level standard cycles | | | BYTE | 3/4 |
| OP30, Emb | OP30: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | SW 2 |
| 9226 | USER_CLASS_READ_CUS | | | H04, H05 | Cross reference: A2 |
| - | Protection level user cycles | | | BYTE | 3/4 |
| OP30, Emb | OP30: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | SW 2 |
| 9227 | USER_CLASS_SHOW_SBL2 | | | H04, H05 | Cross reference: A2 |
| - | Skip single block2 (SBL2) | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | IMMEDIATELY | SW3.5 |
| 9228 | USER_CLASS_READ_SYF | | | H04, H05 | Cross reference: A2 |
| - | Access level for selecting the directory SYF | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | IMMEDIATELY | SW4.2 |
| 9229 | USER_CLASS_READ_DEF | | | H04, H05 | Cross reference: A2 |
| - | Access level for selecting the directory DEF | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | IMMEDIATELY | SW4.2 |
| 9230 | USER_CLASS_READ_BD | | | H04, H05 | Cross reference: A2 |
| - | Access level for selecting the directory BD | | | BYTE | 3/4 |
| Emb | Emb: 3 | 0 | 7 | IMMEDIATELY | SW4.2 |
| 9242 | MA_STAT_DISPLAY_BASE | | | H05 | Cross reference: K2 |
| - | Numerical basis for display of articulated position STAT | | | WORD | 3/4 |
| Adv, Emb | Adv: 10, Emb: 0 | 0 | 16 | IMMEDIATELY | SW6.1 |
| 9243 | MA_TU_DISPLAY_BASE | | | H05 | Cross reference: K2 |
| - | Numerical basis for display of rotary axis position TU | | | WORD | 3/4 |
| Adv, Emb | Adv: 10, Emb: 0 | 0 | 16 | IMMEDIATELY | SW6.1 |

| | | | | | |
|---|---|---|---|---|---|
| 9246 | USER_CLASS_SYS_ZERO_OFF | | | H04, H05 | Cross reference: |
| - | Access level for writing system frames | | | BYTE | 2/2 |
| Adv, Emb | Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | Adv. SW6.3, Emb. SW6.2 |
| 9247 | USER_CLASS_BASE_ZERO_OFF_PA | | | H04, H05 | Cross reference: K2 |
| - | Access level for basic offset PA | | | BYTE | 2/2 |
| Adv, Emb | Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | SW5.3 |
| 9248 | USER_CLASS_BASE_ZERO_OFF_MA | | | H04, H05 | Cross reference: IAM, IM1 |
| - | Access level for basic offset MA | | | BYTE | 2/2 |
| Adv, Emb | Adv: 7, Emb: 7 | 0 | 7 | IMMEDIATELY | SW5.3 |
| 9249 | USER_CLASS_VERT_MODE_SK | | | H04, H05 | Cross reference: K2 |
| - | Protection level vertical softkeys of the area softkeys<br>Protection for vertical SKs | | | DOUBLE | 3/4 |
| Emb | Emb: 2004318071 | 0 | 0x77777777 | IMMEDIATELY | SW6.1 |
| 9251 | USER_CLASS_TM_SKTLLIST | | | H04, H05 | Cross reference: FBW |
| - | Display tool list | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | S 4.1 |
| 9252 | USER_CLASS_TM_SKTOOLLOAD | | | H04, H05 | Cross reference: FBW |
| - | Protection level for loading tools | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9253 | USER_CLASS_TM_SKTOOLUNLOAD | | | H04, H05 | Cross reference: FBW |
| - | Protection level for unloading tools | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9254 | USER_CLASS_TM_SKTOOLMOVE | | | H04, H05 | Cross reference: FBW |
| - | Protection level for relocating tools | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9256 | USER_CLASS_TM_SKMGLREPR2 | | | H04, H05 | Cross reference: FBW |
| - | Protection level for displaying 2nd magazine list | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9257 | USER_CLASS_TM_SKMGLREPR3 | | | H04, H05 | Cross reference: FBW |
| - | Protection level for displaying 3rd magazine list | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9258 | USER_CLASS_TM_SKNCNEWTOOLE | | | H04, H05 | Cross reference: FBW |
| - | Protection level for creating new tool edges | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |

| | | | | | |
|---|---|---|---|---|---|
| 9259 | USER_CLASS_TM_SKNCDELTOOL | | | H04, H05 | Cross reference: FBW |
| - | Protection level for deleting tools | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9260 | USER_CLASS_TM_SKMGBUFFER | | | H04, H05 | Cross reference: FBW |
| - | Protection level for buffer ON/OFF | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9261 | USER_CLASS_TM_SKMGFIND | | | H04, H05 | Cross reference: FBW |
| - | Protection level for search | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9262 | USER_CLASS_TM_SKMGLISTPOS | | | H04, H05 | Cross reference: FBW |
| - | Protection level for positioning | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9263 | USER_CLASS_TM_SKMGNEXT | | | H04, H05 | Cross reference: FBW |
| - | Protection level for paging to next magazine | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9264 | USER_CLASS_TM_SKTLNEWTOOL | | | H04, H05 | Cross reference: FBW |
| - | Protection level for creating tools | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9265 | USER_CLASS_TM_SKTLLREPR1 | | | H04, H05 | Cross reference: FBW |
| - | Protection level for displaying 1st tool list | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9266 | USER_CLASS_TM_SKTLLREPR2 | | | H04, H05 | Cross reference: FBW |
| - | Protection level for displaying 2nd tool list | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9267 | USER_CLASS_TM_SKTLLREPR3 | | | H04, H05 | Cross reference: FBW |
| - | Protection level for displaying 3rd tool list | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9269 | USER_CLASS_TM_SKFINDPLACE | | | H04, H05 | Cross reference: FBW |
| - | Softkey empty location, displ. Tool list | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9270 | USER_CLASS_TM_SKACTPLACE | | | H04, H05 | Cross reference: FBW |
| - | Protection level for loading to curr. location | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9271 | USER_CLASS_TM_SKLDTOOLDAT | | | H04, H05 | Cross reference: FBW |

| | | | | | |
|---|---|---|---|---|---|
| - | Viewing and editing tool data | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | Power On | SW4.1 |
| 9272 | USER_CLASS_APPLICATION | | | H04, H05 | Cross reference: |
| - | Protection level for selecting the operating area 1-16 | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | IMMEDIATELY | SW6.4 |
| 9273 | USER_CLASS_APP_PARAMETER | | | H04, H05 | Cross reference: |
| - | Protection level for softkeys in parameter 1-16 | | | BYTE | 3/4 |
| Emb | Emb: 7 | 0 | 7 | IMMEDIATELY | SW7.1 |
| 9300 | V24_USER_XON | | | H05 | Cross reference: K4 |
| - | User: X on character | | | REAL | 3/4 |
| OP30, Emb | OP30: 17, Emb: 17 | 0 | 0xFF | IMMEDIATELY | - |
| 9301 | V24_USER_XOFF | | | H05 | Cross reference: K4 |
| - | User: X off character | | | REAL | 3/4 |
| OP30, Emb | OP30: 19, Emb: 19 | 0 | 0xFF | IMMEDIATELY | - |
| 9302 | V24_USER_EOF | | | H05 | Cross reference: K4 |
| - | User: End of transmission character | | | REAL | 3/4 |
| OP30, Emb | OP30: 26, Emb: 26 | 0 | 0xFF | IMMEDIATELY | - |
| 9303 | V24_USER_CONTROLS | | | H05 | Cross reference: K4 |
| - | User: Special bits | | | REAL | 3/4 |
| OP30, Emb | OP30: 76, Emb: 76 | 0 | 0x3FF | IMMEDIATELY | - |
| 9304 | V24_USER_RTS | | | H05 | Cross reference: K4 |
| - | User: Line-controlled | | | BYTE | 3/4 |
| OP30, Emb | OP30: 1, Emb: 0 | 0 | 1 | IMMEDIATELY | - |
| 9305 | V24_USER_BAUD | | | H05 | Cross reference: K4 |
| - | User: Baud rate<br>(300, 600, 1200, 2400, 4800, 9600, 19200)<br>0 1 ... | | | BYTE | 3/4 |
| OP30, Emb | OP30: 4, Emb: 5 | 0 | 8 | IMMEDIATELY | - |
| 9306 | V24_USER_DATABITS | | | H05 | Cross reference: K4 |
| - | User: Data bits | | | BYTE | 3/4 |
| OP30, Emb | OP30: 1, Emb: 1 | 0 | 1 | IMMEDIATELY | - |
| 9307 | V24_USER_PARITY | | | H05 | Cross reference: K4 |
| - | User: Parity bits | | | BYTE | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | 0 | 2 | IMMEDIATELY | - |

| | | | | | |
|---|---|---|---|---|---|
| 9308 | V24_USER_STOPBIT | | | H05 | Cross reference: K4 |
| - | User: Stop bits | | | BYTE | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | 0 | 1 | IMMEDIATELY | - |
| 9309 | V24_USER_LINE | | | H05 | Cross reference: K4 |
| - | User: V24 interface (COM1/COM2)<br>(COM1/COM2) | | | BYTE | 3/4 |
| Emb | Emb: 1 | 1 | 2 | IMMEDIATELY | SW5 |
| 9310 | V24_PRINTER_XON | | | H05 | Cross reference: K4 |
| - | Printer: X on character | | | REAL | 3/4 |
| OP30, Emb | OP30: 17, Emb: 17 | 0 | 0xFF | IMMEDIATELY | - |
| 9311 | V24_PRINTER_XOFF | | | H05 | Cross reference: K4 |
| - | Printer: X off character | | | REAL | 3/4 |
| OP30, Emb | OP30: 19, Emb: 19 | 0 | 0xFF | IMMEDIATELY | - |
| 9312 | V24_PRINTER_EOF | | | H05 | Cross reference: K4 |
| - | Printer: End of transmission character | | | REAL | 3/4 |
| OP30, Emb | OP30: 12, Emb: 12 | 0 | 0xFF | IMMEDIATELY | - |
| 9313 | V24_PRINTER_CONTROLS | | | H05 | Cross reference: K4 |
| - | Printer: Special bits | | | REAL | 3/4 |
| OP30, Emb | OP30: 76, Emb: 76 | 0 | 0x3FF | IMMEDIATELY | - |
| 9314 | V24_PRINTER_RTS | | | H05 | Cross reference: K4 |
| - | Printer: Line-controlled | | | BYTE | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | 0 | 1 | IMMEDIATELY | - |
| 9315 | V24_PRINTER_BAUD | | | H05 | Cross reference: K4 |
| - | Printer: Baud rate<br>(300, 600, 1200, 2400, 4800, 9600, 19200)<br>0 1 ... | | | BYTE | 3/4 |
| OP30, Emb | OP30: 5, Emb: 5 | 0 | 8 | IMMEDIATELY | - |
| 9316 | V24_PRINTER_DATABITS | | | H05 | Cross reference: K4 |
| - | Printer: Data bits | | | BYTE | 3/4 |
| OP30, Emb | OP30: 1, Emb: 1 | 0 | 1 | IMMEDIATELY | - |
| 9317 | V24_PRINTER_PARITY | | | H05 | Cross reference: K4 |
| - | Printer: Parity bits | | | BYTE | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | 0 | 2 | IMMEDIATELY | - |

| | | | | | |
|---|---|---|---|---|---|
| 9318 | V24_PRINTER_STOPBIT | | | H05 | Cross reference: K4 |
| - | Printer: Stop bits | | | BYTE | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | 0 | 1 | IMMEDIATELY | - |
| 9319 | V24_PRINTER_LINE | | | H05 | Cross reference: K4 |
| - | Printer: V24 interface (COM1/COM2)<br>(COM1/COM2) | | | BYTE | 3/4 |
| Emb | Emb: 1 | 1 | 2 | IMMEDIATELY | - |
| 9320 | V24_PG_PC_XON | | | H05 | Cross reference: K4 |
| - | PG: X on character<br>9321 PG: X off character | | | REAL | 3/4 |
| OP30, Emb | OP30: 17, Emb: 17 | 0 | 0xFF | IMMEDIATELY | - |
| 9321 | V24_PG_PC_XOFF | | | H05 | Cross reference: K4 |
| - | | | | REAL | 3/4 |
| OP30, Emb | OP30: 19, Emb: 19 | 0 | 0xFF | IMMEDIATELY | - |
| 9322 | V24_PG_PC_EOF | | | H05 | Cross reference: K4 |
| - | PG: End of transmission character | | | REAL | 3/4 |
| OP30, Emb | OP30: 26, Emb: 26 | 0 | 0xFF | IMMEDIATELY | - |
| 9323 | V24_PG_PC_CONTROLS | | | H05 | Cross reference: K4 |
| - | PG: Special bits<br>9324 PG: Line-controlled | | | REAL | 3/4 |
| OP30, Emb | OP30: 144,<br>Emb: 144 | 0 | 0x3FF | IMMEDIATELY | - |
| 9324 | V24_PG_PC_RTS | | | H05 | Cross reference: K4 |
| - | | | | BYTE | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | 0 | 1 | IMMEDIATELY | - |
| 9325 | V24_PG_PC_BAUD | | | H05 | Cross reference: K4 |
| - | PG: Baud rate<br>(300, 600, 1200, 2400, 4800, 9600)<br>0 1 ... | | | BYTE | 3/4 |
| OP30, Emb | OP30: 5, Emb: 5 | 0 | 8 | IMMEDIATELY | - |
| 9326 | V24_PG_PC_DATABITS | | | H05 | Cross reference: K4 |
| - | PG: Data bits | | | BYTE | 3/4 |
| OP30, Emb | OP30: 1, Emb: 1 | 0 | 1 | IMMEDIATELY | - |
| 9327 | V24_PG_PC_PARITY | | | H05 | Cross reference: K4 |
| - | PG: Parity bits | | | BYTE | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | 0 | 2 | IMMEDIATELY | - |

| | | | | | |
|---|---|---|---|---|---|
| **9328** | **V24_PG_PC_STOPBIT** | | | H05 | **Cross reference:** K4 |
| - | PG: Stop bits | | | BYTE | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | 0 | 1 | IMMEDIATELY | - |
| **9329** | **V24_PG_PC_LINE** | | | H05 | **Cross reference:** K4 |
| - | PG: V24 interface (COM1/COM2)<br>(COM1/COM2) | | | BYTE | 3/4 |
| Emb | Emb: 1 | 1 | 2 | IMMEDIATELY | - |
| **9400** | **TOOL_REF_GEO_AXIS1** | | | H05 | **Cross reference:** BA |
| - | Absolute dimension tool length compensation GEOaxis.1 | | | DOUBLE | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | *** | *** | IMMEDIATELY | - |
| **9401** | **TOOL_REF_GEO_AXIS2** | | | H05 | **Cross reference:** BA |
| - | Absolute dimension tool length compensation GEOaxis.2 | | | DOUBLE | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | *** | *** | IMMEDIATELY | - |
| **9402** | **TOOL_REF_GEO_AXIS3** | | | H05 | **Cross reference:** BA |
| - | Absolute dimension tool length compensation GEOaxis.3 | | | DOUBLE | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | *** | *** | IMMEDIATELY | - |
| **9410** | **TM_LOAD_PLACE** | | | H05 | **Cross reference:** BA |
| - | Number of loading station | | | INTEGER | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | *** | *** | Power On | - |
| **9411** | **TM_NUM_MAG** | | | H05 | **Cross reference:** BA |
| - | Number of work magazine | | | INTEGER | 3/4 |
| OP30, Emb | OP30: 0, Emb: 0 | *** | *** | Power On | - |
| **9412** | **TM_DEFAULT_TOOLSIZE** | | | H05 | **Cross reference:** FBW |
| - | Default for tool size | | | REAL | 3/4 |
| Emb | Emb: 1111 | 1111 | 7777 | IMMEDIATELY | SW4.1 |
| **9414** | **TM_KIND_OF_TOOLMANAGEMENT** | | | H01, H02, H05 | **Cross reference:** FBW |
| - | Type of representation of tool management<br>0: Old, 1: New (SW 5.2 and higher) | | | BYTE | 3/4 |
| SM, ST, Emb | SM: 1, ST: 1, Emb: 0 | 0 | 1 | Power On | SW5 |
| **9415** | **TM_DEFAULT_TOOLPLACESPEC** | | | H05 | **Cross reference:** FBW |
| - | Default value for location type | | | BYTE | 3/4 |
| Emb | Emb: 1 | 1 | 99 | IMMEDIATELY | SW4.2 |

| | | | | | |
|---|---|---|---|---|---|
| 9416 | TM_DEFAULT_TOOLTYPE | | | H05 | Cross reference: FBW |
| - | Default for location type | | | REAL | 3/4 |
| Emb | Emb: 120 | 100 | 900 | IMMEDIATELY | SW4.1 |
| 9417 | TM_DEFAULT_TOOLSTATE | | | H05 | Cross reference: FBW |
| - | Default for tool status load | | | INTEGER | 3/4 |
| Emb | Emb: 2 | 0 | 255 | IMMEDIATELY | SW4.1 |
| 9419 | TM_DEFAULT_DELETE_TOOL | | | H05 | Cross reference: FBW |
| - | Default for tool data auto. Deleting | | | BYTE | 3/4 |
| Emb | Emb: 0 | 0 | 1 | IMMEDIATELY | SW4.1 |
| 9420 | MA_ONLY_MKS_DIST_TO_GO | | | H05 | Cross reference: FBW |
| - | Distance-to-go display in the WCS window<br>0: Work value<br>1: Machine value | | | BYTE | 3/4 |
| Emb | Emb: 0 | 0 | 1 | IMMEDIATELY | SW4.1 |
| 9421 | MA_AXES_SHOW_GEO_FIRST | | | H05 | Cross reference: K1 |
| - | Actual-value display with leading geo axes | | | BYTE | 3/4 |
| Adv, Emb | Adv: 1, Emb: 1 | 0 | 1 | IMMEDIATELY | SW 2 |
| 9422 | MA_PRESET_MODE | | | H05 | Cross reference: K1 |
| - | Selection PRESET/basic offset in JOG<br>0 no preset, no preset actual value memory<br>1 PRESET<br>2 Preset actual value memory<br>3 Preset actual value memory s. Online help | | | BYTE | 3/4 |
| Adv, Emb | Adv: 1, Emb: 1 | 0 | 3 | IMMEDIATELY | SW5 |
| 9423 | MA_MAX_SKP_LEVEL | | | H05 | Cross reference: K1 |
| - | Max. skip levels in NC program | | | BYTE | 3/4 |
| Adv, Emb | Adv: 1, Emb: 1 | 1 | 8 | Power On | SW5 |
| 9424 | MA_COORDINATE_SYSTEM | | | H05 | Cross reference: K2 |
| - | Coord. System for actual-value display<br>0: WCS<br>1: SZS (settable zero system) | | | BYTE | 3/4 |
| Adv, Emb | Adv: 0, Emb: 0 | 0 | 1 | Power On | SW5 |
| 9425 | MA_SCRATCH_DEFAULT_MODE | | | H05 | Cross reference: K2 |
| - | Tool offset calculation scratching | | | DOUBLE | 3/4 |
| Emb | Emb: 0 | 0 | 2236962 | IMMEDIATELY | 5.3 |

| | | | | | |
|---|---|---|---|---|---|
| 9426 | MA_AX_DRIVELOAD_FROM_PLC1 | | | H01, H02, H05 | Cross reference: |
| - | Machine-axis index, analog spindle power-rating display | | | BYTE | 3/4 |
| Emb | Emb: 0 | 0 | 31 | Power On | Emb 6.5 |
| 9427 | MA_AX_DRIVELOAD_FROM_PLC2 | | | H01, H02, H05 | Cross reference: |
| - | Machine-axis index, analog spindle power-rating display | | | BYTE | 3/4 |
| Emb | Emb: 0 | 0 | 31 | Power On | Emb 6.5 |
| 9428 | MA_SPIND_MAX_POWER | | | H01, H02, H05 | Cross reference: IAM4 |
| - | Maximum value of spindle display | | | REAL | 3/4 |
| Emb | Emb: 100 | 100 | 0xFFFF | Power On | Emb 6.5 |
| 9429 | MA_SPIND_POWER_RANGE | | | H01, H02, H05 | Cross reference: IAM4 |
| - | Maximum value of spindle display | | | REAL | 3/4 |
| Emb | Emb: 100 | 100 | 0xFFFF | Power On | Emb 6.5 |
| 9440 | ACTIVATE_SEL_USER_DATA | | | H05 | Cross reference: K2 |
| - | Activating the active wear immediately | | | BYTE | 3/4 |
| Adv, Emb | Adv: 1, Emb: 0 | 0 | 1 | IMMEDIATELY | SW4.3 |
| 9442 | MA_AUXFU_GROUPS | | | H01, H02, H05 | Cross reference: |
| - | Displayed auxiliary function groups | | | STRING | 3/7 |
| Emb, | Emb: 0 | *** | *** | Power On | SW4.2 |
| 9450 | WRITE_TOA_FINE_LIMIT | | | H05 | Cross reference: K2 |
| mm | Limit value for wear fine | | | DOUBLE | 3/4 |
| Adv, Emb | Adv: 0, Emb: 0.999 | *** | *** | IMMEDIATELY | SW4.2 |
| 9451 | WRITE_ZOA_FINE_LIMIT | | | H05 | Cross reference: K2 |
| mm | Limit value for offset fine | | | DOUBLE | 3/4 |
| Adv, Emb | Adv: 0, Emb: 0.999 | *** | *** | IMMEDIATELY | SW4.2 |
| 9459 | PA_ZOA_MODE | | | H05 | Cross reference: K2, IM2 |
| - | Display mode of zero offset | | | BYTE | 3/4 |
| Emb | Emb: 1 | 0 | 1 | IMMEDIATELY | SW6.1 |
| 9460 | PROGRAM_SETTINGS | | | H05 | Cross reference: A2 |
| - | Settings in the Program area | | | INTEGER | 3/4 |
| Adv, Emb | Adv: 0, Emb: 5 | *** | *** | IMMEDIATELY | SW5.1 |
| 9461 | CONTOUR_END_TEXT | | | H05 | Cross reference: A2 |
| - | String to be added at the end of the contour | | | STRING | 3/4 |
| Emb | Emb: '' | *** | *** | IMMEDIATELY | SW5.1 |
| 9464 | MAX_PROGRAMM_SIZE_CHECK | | | H05 | Cross reference: |
| - | File size, as of which no check is made | | | INTEGER | 3/4 |
| Emb | Emb: 0 | *** | *** | IMMEDIATELY | SW6.4 |

| | | | | | |
|---|---|---|---|---|---|
| 9477 | TO_TRACE | | | H01, H02, H05 | Cross reference: |
| - | For internal testing purposes | | | REAL | 3/4 |
| SM, ST, Emb | SM: 0, ST: 0, Emb: 0 | 0 | 0xFFFF | Power On | |
| 9478 | TO_OPTION_MASK | | | H01, H02, H05 | Cross reference: |
| - | For internal purposes | | | INTEGER | 2/2 |
| SM, ST, Emb | SM: 1, ST: 1, Emb: 0 | 0 | 0xFFFF | Power On | |
| 9479 | TO_MAG_PLACE_DISTANCE | | | H02, H05 | Cross reference: |
| mm | Distance between individual magazine locations | | | DOUBLE | 3/4 |
| ST, Emb | ST: Emb: 0 | 0 | 10000 | Power On | SW6.3 |
| 9500 | NC_PROPERTIES | | | H05 | Cross reference: A2 |
| - | NC properties<br>Bit 0: Digital drives<br>Bit 1: Software commissioning switch<br>Bits 2...4: Reserved | | | BYTE | 3/4 |
| OP30, Emb | OP30: 255, Emb: 255 | 0 | 0xFF | IMMEDIATELY | SW 2 |
| 9509 | USER_CLASS_DIRECTORY_CHG | | | H04, H05 | Cross reference: FBT, FBSP, EMB |
| - | Protection level for network configuration | | | BYTE | 3/4 |
| Emb | Emb: 1 | 0 | 7 | IMMEDIATELY | SW6.2 |
| 9510 | USER_CLASS_DIRECTORY1_P | | | H04, H05 | Cross reference: A2 |
| - | Protection level for network drive 1 prog. | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 1 | 0 | 7 | IMMEDIATELY | SW6.1 |
| 9511 | USER_CLASS_DIRECTORY2_P | | | H04, H05 | Cross reference: A2 |
| - | Protection level for network drive 2 prog. | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 1 | 0 | 7 | IMMEDIATELY | SW6.1 |
| 9512 | USER_CLASS_DIRECTORY3_P | | | H04, H05 | Cross reference: A2 |
| - | Protection level for network drive 3 prog. | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 1 | 0 | 7 | IMMEDIATELY | SW6.1 |
| 9513 | USER_CLASS_DIRECTORY4_P | | | H04, H05 | Cross reference: A2 |
| - | Protection level for network drive 4 prog. | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 1 | 0 | 7 | IMMEDIATELY | 6.1 |
| 9516 | USER_CLASS_DIRECTORY1_M | | | H04, H05 | Cross reference: A2 |
| - | Protection level for network drive 1 mach. | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 0 | 0 | 7 | IMMEDIATELY | 6.1 |

| | | | | | |
|---|---|---|---|---|---|
| 9517 | USER_CLASS_DIRECTORY2_M | | | H04, H05 | Cross reference: A2 |
| - | Protection level for network drive 2 mach. | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 0 | 0 | 7 | IMMEDIATELY | 6.1 |
| 9518 | USER_CLASS_DIRECTORY3_M | | | H04, H05 | Cross reference: A2 |
| - | Protection level for network drive 3 mach. | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 0 | 0 | 7 | IMMEDIATELY | 6.1 |
| 9519 | USER_CLASS_DIRECTORY4_M | | | H04, H05 | Cross reference: A2 |
| - | Protection level for network drive 4 mach. | | | BYTE | 3/4 |
| Adv, Emb | Adv: 7, Emb: 0 | 0 | 7 | IMMEDIATELY | 6.1 |
| 9600 | CTM_SIMULATION_DEF_X | | | H01, H02, H03, H05 | Cross reference: FBMA, FBSP |
| - | Simulation default value for X | | | INTEGER | 3/4 |
| SM, ST, MT, Emb | SM: 0, ST: 0, MT: 0, Emb: 0 | -10000 | 10000 | Power On | SW2.1 (810D), 4.3 (840D) |
| 9601 | CTM_SIMULATION_DEF_Y | | | H01, H02, H03, H05 | Cross reference: FBMA, FBSP |
| - | Simulation default value for Z | | | INTEGER | 3/4 |
| SM, ST, MT, Emb | SM: 0, ST: 0, MT: 0, Emb: 0 | -10000 | 10000 | Power On | SW2.1 (810D), 4.3 (840D) |
| 9602 | CTM_SIMULATION_DEF_VIS_AREA | | | H01, H02, H03, H05 | Cross reference: FBMA, FBSP |
| - | Simulation default value for display area | | | INTEGER | 3/4 |
| SM, ST, MT, Emb | SM: 100, ST: 100, MT: 100, Emb: 100 | -10000 | 10000 | Power On | SW2.1 (810D), 4.3 (840D) |
| 9603 | CTM_SIMULATION_MAX_X | | | H01, H02, H03, H05 | Cross reference: FBMA, FBSP |
| - | Simulation of maximum display X | | | INTEGER | 3/4 |
| SM, ST, MT, Emb | SM: 0, ST: 0, MT: 0, Emb: 0 | -10000 | 10000 | Power On | SW2.1 (810D), 4.3 (840D) |
| 9604 | CTM_SIMULATION_MAX_Y | | | H01, H02, H03, H05 | Cross reference: FBMA, FBSP |
| - | Simulation maximum display Z | | | INTEGER | 3/4 |
| SM, ST, MT, Emb | SM: 0, ST: 0, MT: 0, Emb: 0 | -10000 | 10000 | Power On | 840D SW4.3, 810D SW2.1 |

| | | | | | |
|---|---|---|---|---|---|
| **9605** | **CTM_SIMULATION_MAX_VIS_AREA** | | | H01, H02, H03, H05 | **Cross reference:** FBMA, FBSP |
| - | Simulation of maximum display area | | | INTEGER | 3/4 |
| SM, ST, MT, Emb | SM: 1000, ST: 1000, MT: 1000, Emb: 1000 | -10000 | 10000 | Power On | 840D SW4.3, 810D SW2.1 |
| **9606** | **CTM_SIMULATION_TIME_NEW_POS** | | | H01, H02, H03, H05 | **Cross reference:** FBMA, FBT |
| - | Simulation updating rate of actual value | | | INTEGER | 3/4 |
| SM, ST, MT, Emb | SM: 250, ST: 350, MT: 250, Emb: 100 | 0 | 4000 | Power On | 840D SW4.3, 810D SW2.1, ST SW6.1 |
| **9610** | **CTM_POS_COORDINATE_SYSTEM** | | | H03, H05 | **Cross reference:** FBMA |
| - | Position of coordinate system for turning | | | BYTE | 3/4 |
| MT, Adv, Emb | MT: 2, Adv: 2, Emb: 2 | 0 | 7 | IMMEDIATELY | 840D SW4.3, 810D SW2.1 |
| **9611** | **CTM_CROSS_AX_DIAMETER_ON** | | | H02, H03, H05 | **Cross reference:** FBMA, FBT |
| - | Diameter display for active transverse axes | | | BYTE | 3/4 |
| ST, MT, Emb | ST: 1, MT: 1, Emb: 1 | 0 | 1 | IMMEDIATELY | 840D SW4.3, 810D SW2.1, ST 6.1 |
| **9619** | **CTM_G91_DIAMETER_ON** | | | H02, H03, H05 | **Cross reference:** FBMA, FBT |
| - | Incremental infeed | | | BYTE | 3/4 |
| ST, MT, Emb | ST: 0, MT: 0, Emb: 1 | 0 | 1 | IMMEDIATELY | 840D SW4.3, 810D SW2.1, ST SW6.1 |
| **9632** | **CTM_ANGLE_REFERENCE_AXIS** | | | H03, H05 | **Cross reference:** FBMA |
| - | Angle reference axis<br>1: 1. Axis<br>2: 2. Axis | | | REAL | 3/4 |
| MT, Emb | MT: 1, Emb: 1 | 0 | 1 | IMMEDIATEL Y | 840D SW4.4, 810D SW2.4 |
| **9639** | **CTM_MAX_TOOL_WEAR** | | | H03, H05 | **Cross reference:** FBMA |
| - | Upper limit tool wear input | | | DOUBLE | 3/4 |
| MT, Emb | MT: 1, Emb: 1 | 0 | 10 | Power On | 840D SW4.4, 810D SW2.4 |

| | | | | | |
|---|---|---|---|---|---|
| 9650 | CMM_POS_COORDINATE_SYSTEM | | | H01, H02, H05 | Cross reference: FBSP, FBT |
| - | Position of coordinate system | | | BYTE | 3/4 |
| SM, ST, Adv, Emb | SM: 0, ST: 34, Adv: 0, Emb: 0 | 0 | 47 | IMMEDIATELY | SW4.3, ST SW6.1 |
| 9651 | CMM_TOOL_MANAGEMENT | | | H01, H02, H05 | Cross reference: FBSP, FBT |
| - | Tool management strategy | | | BYTE | 3/4 |
| SM, ST, Adv, Emb | SM: 4, ST: 4, Adv: 4, Emb: 4 | 1 | 4 | Power On | SW6.1, ST SW6.1 |
| 9652 | CMM_TOOL_LIFE_CONTROL | | | H01, H02, H05 | Cross reference: FBSP, FBT |
| - | Tool monitoring | | | BYTE | 3/4 |
| SM, ST, Adv, Emb | SM: 1, ST: 1, Adv: 1, Emb: 1 | 0 | 1 | Power On | SW6.1 |
| 9663 | CMM_TOOL_DISPLAY_IN_DIAM | | | H01, H02, H05 | Cross reference: FBSP, FBT |
| - | Display radius/diameter for tool | | | BYTE | 3/4 |
| SM, ST, Adv, Emb | SM: 1, ST: 1, Adv: 1, Emb: 1 | 0 | 1 | Power On | 840D SW4.3, 810D SW2.3, ST SW6.1 |
| 9671 | CMM_TOOL_LOAD_DEFAULT_MAG | | | H01, H02, | Cross reference: FBSP |
| - | Load tool in default magazine | | | BYTE | 3/4 |
| SM, ST, Emb | SM: 0, ST: 0 | 0 | 30 | Power On | 840D SW6.4 |
| 9672 | CMM_FIXED_TOOL_PLACE | | | H01, H02, H05 | Cross reference: FBSP, FBT |
| - | Fixed location coding | | | BYTE | 3/4 |
| SM, ST, Adv, Emb | SM: 0, ST: 1, Adv: 0, Emb: 0 | 0 | 1 | Power On | 840D SW4.4, 810D SW2.4, ST SW6.1 |
| 9673 | CMM_TOOL_LOAD_STATION | | | H01, H02, H05 | Cross reference: FBSP, FBT |
| - | Number of loading station | | | BYTE | 3/4 |
| SM, ST, Adv, Emb | SM: 1, ST: 1, Adv: 1, Emb: 1 | 1 | 2 | Power On | 840D SW4.4, 810D SW2.4, ST SW6.1 |
| 9674 | CMM_ENABLE_TOOL_MAGAZINE | | | H01, H02, H05 | Cross reference: FBSP, FBT |
| - | Display of magazine list | | | BYTE | 3/4 |
| SM, ST, Adv, Emb | SM: 1, ST: 1, Adv: 1, Emb: 1 | 0 | 1 | Power On | 840D SW4.4, 810D SW2.4, ST SW6.1 |

| | | | | | |
|---|---|---|---|---|---|
| 9676 | CMM_DIRECTORY_SOFTKEY_PATH1 | | | H01, H02, H05 | Cross reference: FBSP, FBT |
| - | Path to the drive names in the directory man. | | | STRING | 3/4 |
| SM, ST, Adv, Emb | SM: ", ST: ", Adv: ", Emb: 0 | *** | *** | Power On | 840D SW4.4, 810D SW2.4 with SM |
| 9677 | CMM_DIRECTORY_SOFTKEY_PATH2 | | | H01, H02, H05 | Cross reference: FBSP, FBT |
| - | Path to the drive names in the directory man. | | | STRING | 3/4 |
| SM, ST, Adv, Emb | SM: ", ST: ", Adv: ", Emb: 0 | *** | *** | Power On | 840D SW4.4, 810D SW2.4 with SM |
| 9678 | CMM_DIRECTORY_SOFTKEY_PATH3 | | | H01, H02, H05 | Cross reference: FBSP, FBT |
| - | Path to the drive names in the directory man. | | | STRING | 3/4 |
| SM, ST, Adv, Emb | SM: ", ST: ", Adv: ", Emb: 0 | *** | *** | Power On | 840D SW4.4, 810D SW2.4 with SM |
| 9679 | CMM_DIRECTORY_SOFTKEY_PATH4 | | | H01, H02, H05 | Cross reference: FBSP, FBT |
| - | Path to the drive names in the directory man. | | | STRING | 3/4 |
| SM, ST, Adv, Emb | SM: ", ST: ", Adv: ", Emb: 0 | *** | *** | Power On | 840D SW4.4, 810D SW2.4 with SM |
| 9687 | CMM_TOOL_MOVE_DEFAULT_MAG | | | H01, H02 | Cross reference: FBSP |
| - | Reload tool in default magazine | | | BYTE | 3/4 |
| SM, ST, | SM: 0, ST: 0 | 0 | 30 | Power On | SW6.4 |
| 9688 | CMM_COUNT_GEAR_STEPS_S2 | | | H01 | Cross reference: - |
| - | Number of gear stages for the 2nd spindle | | | BYTE | 3/4 |
| SM | SM:1 | 0 | 5 | IMMEDIATELY | SW6.4 |
| 9679 | CMM_DIRECTORY_SOFTKEY_PATH4 | | | H01, H02, H05 | Cross reference: FBSP, FBT |
| - | Path to the drive names in the directory man. | | | STRING | 3/4 |
| SM, ST, Adv, Emb | SM: ", ST: ", Adv: ", Emb: 0 | *** | *** | Power On | 840D SW4.4, 810D SW2.4 with SM |
| 9900 | MD_TEXT_SWITCH | | | H05 | Cross reference: - |
| - | Plaintext instead of MD identifier | | | BOOL | 3/4 |
| OP30, Adv, Emb | OP30: 0, Adv: 0, Emb: 0 | *** | *** | IMMEDIATELY | SW 2 |
| 9950 | MD_NC_TEA_FILTER | | | H05 | Cross reference: |
| - | Display options general machine data | | | INTEGER | 0/0 |
| Emb | Emb: 67108865 | *** | *** | Power On | |
| 9951 | MD_NC_TEA_IDX_LIMIT | | | H05 | Cross reference: |
| - | Index filter for general machine data | | | INTEGER | 0/0 |
| Emb | Emb: 0 | *** | *** | Power On | |

| | | | | | |
|---|---|---|---|---|---|
| 9952 | MD_AX_TEA_FILTER | | | H05 | Cross reference: |
| - | Display options axis machine data | | | INTEGER | 0/0 |
| Emb | Emb: 67108865 | *** | *** | Power On | |
| 9953 | MD_AX_TEA_IDX_LIMIT | | | H05 | Cross reference: |
| - | Index filter for axis machine data | | | INTEGER | 0/0 |
| Emb | Emb: 0 | *** | *** | Power On | |
| 9954 | MD_CH_TEA_FILTER | | | H05 | Cross reference: |
| - | Display options channel machine data | | | INTEGER | 0/0 |
| Emb | Emb: 33554433 | *** | *** | Power On | |
| 9955 | MD_CH_TEA_IDX_LIMIT | | | H05 | Cross reference: |
| - | Index filter for channel machine data | | | INTEGER | 0/0 |
| Emb | Emb: 0 | *** | *** | Power On | |
| 9956 | MD_DRV_TEA_FILTER | | | H05 | Cross reference: |
| - | Display options drive machine data | | | INTEGER | 0/0 |
| Emb | Emb: 8388609 | *** | *** | Power On | |
| 9957 | MD_DRV_TEA_IDX_LIMIT | | | H05 | Cross reference: |
| - | Index filter for drive machine data | | | INTEGER | 0/0 |
| Emb | Emb: 0 | *** | *** | Power On | |
| 9958 | MD_SNX_FILTER | | | H05 | Cross reference: |
| | Display options Sinamics parameters | | | INTEGER | |
| Emb | Emb: 0 | *** | *** | Power On | |
| 9959 | MD_SNX_IDX_LIMIT | | | H05 | Cross reference: |
| | Index filter for Sinamics parameters | | | INTEGER | |
| Emb | Emb: 0 | *** | *** | Power On | |
| 9980 | LANGUAGE_SETTINGS | | | H05 | Cross reference: |
| - | Internal language settings | | | INTEGER | 0/0 |
| Emb | Emb: 513 | *** | *** | Power On | |
| 9990 | SW_OPTIONS | | | H05 | Cross reference: FBSP, FBT |
| - | Enable MMC/HMI SW options | | | INTEGER | 2/2 |
| Adv, Emb | Adv: 0, Emb: 0 | *** | *** | Power On | SW5.3 |
| 9991 | HMI_HELP_SYSTEMS | | | H05 | Cross reference: FBSP, FBT |
| - | Enable MMC/HMI help systems<br>Bit 0 = 1 Help for calculator enabled (standard) | | | INTEGER | 2/2 |
| Emb | Emb: 1 | *** | *** | Power On | SW6.1 |
| 9992 | HMI_TESTAUTOMAT_OPTION | | | H05 | Cross reference: FBT, FBSP, EMB |
| - | Options for HMI automatic test machine | | | INTEGER | 2/2 |
| Emb | Emb: 0 | *** | *** | Power On | SW6.3 |
| 9993 | HMI_WIZARD_OPTION | | | H05 | Cross reference: FBT, FBSP, EMB |
| - | Options for the wizard | | | INTEGER | 2/2 |
| Emb | Emb: 0 | *** | *** | Power On | SW6.3 |

| 9999 | TRACE | | | H05 | Cross reference: - |
|---|---|---|---|---|---|
| - | Test flags for internal diagnosis | | | INTEGER | 2/2 |
| OP30, Adv, Emb | OP30: 0, Adv: 0, Emb: 0 | 0 | 0xFFFF | Power On | - |

# Creating user-specific alarm texts 4

## 4.1 Alarm, user, message text files

### 4.1.1 Introduction

#### General

This chapter describes the basics and the procedure to follow when in-house text files, e.g., for in-house cycle alarm texts or in-house PLC alarm texts and messages, are to be inserted. With Sinumerik 840D sl, all alarm text files are stored in the CompactFlash card's file system.

The file system is subdivided into the directories (subtrees) “siemens”, “addon”, “oem”, and “user”. Each directory is structured in the same way.

---

**Note**

None of the files in the “siemens” area may be changed!

---

New/modified text files can be inserted into “oem” (for the machine manufacturer) or “user” (for end user).

#### Access to CompactFlash card (optional)

You can access the CompactFlash card’s file system in order to insert/edit text files in the "Commissioning“ operating area via the "HMI" →"Editor (only HMI)" → "Memory Card (CF)" softkeys and the file functions provided there.

See Editing user-specific text files (Page 83).

---

**Note**

The CompactFlash card can only be accessed while the NCU is running.

---

### 4.1.2 Storing the text files

#### Directory structure

The alarm and message text files are stored in the corresponding subdirectories in the following areas:

| | |
|---|---|
| "siemens" | (/**siemens**/sinumerik/**hmi**/lng/) |
| "addon" | (/**addon/**sinumerik/**hmi**/lng/) |
| "oem" | (/**oem**/sinumerik/**hmi**/lng/) |
| "user" | (/**user**/sinumerik/**hmi**/lng/) |

The text files in the “siemens” area represent the original status and cannot be modified. The text files are sorted according to language and stored in separate subdirectories in each “lng” directory. The subdirectory names correspond to the language abbreviations from Table "Supported languages", column: Language abbreviation, e.g. you will find German text files under the following path: /siemens/sinumerik/hmi/lng/**deu**

The text files for cycles are stored in a directory named "cycles", which is located parallel to "hmi" in each case:

| | |
|---|---|
| "siemens" | (/**siemens**/sinumerik/**cycles**/lng/) |
| "addon" | (/**addon**/sinumerik/**cycles**/lng/) |
| "oem" | (/**oem**/sinumerik/**cycles**/lng/) |
| "user" | (/**user**/sinumerik/**cycles**/lng/) |

The language-dependent subdirectories for the installed languages are already set up in the "siemens" directory.

They will have to be created in the “oem” and “user” directories, if required.

#### Creating a directory

You can save the language-dependent directories, e.g. deu, eng, ... under the directories "oem" and "user".

Use the language abbreviation from the table:

See Supported languages (Page 80)

Press softkeys “Commissioning” → "HMI" → "Editor (only HMI)" → "File Function" → "New".

---

**Note**

Directory names must be written in lower case letters, e.g., deu.

An external keyboard is required to write in lower case!

---

### Modifiable alarm text files

You can insert you own texts in the following files:

| | |
|---|---|
| alpu.txt | PLC alarm/message texts |
| alsi.txt | Safety Integrated alarm texts |
| alc.txt | Compile cycle alarm text |

In the “cycles” subdirectory (/oem/sinumerik/cycles/lng/deu):

| | |
|---|---|
| aluc.txt | User-cycle and tool-management texts<br>Amend texts for user interface (wizard) |
| alzu.txt | User cycle alarm texts |

## 4.2 Number ranges of alarms and messages

### Alarm numbers

The following alarm number ranges are available for cycle, compile cycle and PLC alarms/messages:

Table 4-1 Alarm number ranges

| Number range | Name | Effect | Deleting | File name |
|---|---|---|---|---|
| 000000 - 000320 | Safety alarms | | | alsi.txt |
| 085000 - 089990 | User-cycle and tool-management texts, supplement texts for user interface (Wizard) | | | aluc.txt |
| 065000 - 065999 | Cycle alarm (user) | Display, interlocking NC start | Reset | alzu.txt |
| 066000 - 066999 | | Display, interlock NC start, motion standstill after executing the pre-decoded blocks | Reset | |
| 067000 – 067999 | | Display | Cancel | |
| 068000 - 069000 | Reserved | | | |
| 070000 - 079999 | Compile cycle alarms | | | alc.txt |
| 400000 - 499999 | PLC alarms general | | | alpu.txt |
| 500000 - 599999 | PLC messages for axis/spindle and channels | | | |
| 700000 - 709999 | PLC messages for users | | | |
| 800000 – 899999 | PLC alarms for sequence cascades/graphs | | | |
| 810000 - 810009 | System messages in the PLC | | | |

The number range in the list is not available with every number.

**References:** Diagnostics Manual

# 4.3 Syntax of the text file

## 4.3.1 File name

Only those file names listed under “Editable alarm text files” in the previous section may be used.

## 4.3.2 Example of an alpu.txt file

```
// CP = 1252
// IDEO = NO
// ANSI = YES
// UNICODE = NO
// VERSION = $VERSION
// NCKOEM.P7_61
// NCK.P7_61
[SWITCHES]/ALARM
700000 0 0 "Alarm Sample 700000"
800001 0 0 "Safety door faulty, error analysis required"
//*******************************************************************
****/
//* PLC user alarm file */
//*******************************************************************
****/
//* User entries: */
//* p.e. */
//* 800001 0 0 "Safety door faulty, error analysis required" */
//* max. 62 characters! */
//* Area for Application: 700000-799999,800000-809999,811000-819999
*/
//*******************************************************************
*/
```

**Note**

A minimum of 2 alarm texts must be configured in the alarm text file so that the alarm/message text is displayed.

## Header

The text files begin with several comments, which contain information for the runtime system.

You can enter texts that can be assigned freely after “[SWITCHES]/ALARM”.

| | |
|---|---|
| // CP = XXXX | Number of the code page used to create the file (ANSI table, see Table 5-5) |
| // IDEO = | Use of an Asian text:<br>Yes: Logographic language<br>No: Single-byte language (default: No) |
| // ANSI = | File format of the text file<br>Yes: ANSI code page<br>No: OEM code page (default: No) |
| // UNICODE = | File format of the text file<br>Yes: Unicode<br>No: Single-byte (default: No) |
| // VERSION = | Version ID |

## Comment

You can insert comments before every line in the alarm text file. These comments must begin with "//".

## Format of the text file for PLC alarm/message texts

The ASCII file for PLC alarm texts has the following structure:

Table 4-2 Structure of text file for PLC alarm texts

| Alarm number | Display | Help ID | Text or alarm number | Displayed text |
|---|---|---|---|---|
| 510000 | 1 | 0 | "Channel %K FDDIS all" | Channel 1 FDDIS all |
| 600124 | 1 | 0 | "Feed disable axis %A" | Feed disable axis 1 |
| 600224 | 0 | 0 | 600124 | Feed disable axis 2 |
| 600324 | 1 | 0 | 600224 | Feed disable axis 3 |
| 702463 | 0 | 0 | "Group index:%A Number:%N" | Group index:24 Number:63 |
| // Alarm text file for PLC alarm | | | | |

### Alarm number

List of alarm numbers

### Display

This number defines the alarm display type:

0: Display in alarm line

1: Display in a dialog box

### Help ID

Since help is not displayed with HMI-Embedded, a "0" always has to be entered.

### Text or alarm number

Characters " and # must not be used in alarm texts. The % character is reserved for displaying parameters.

If an existing text is to be used, this can be done with a reference to the corresponding alarm. 6-digit alarm number instead of "text".

- The maximum length of the alarm/message text is 110 characters in total for a 2-line display. If the text is too long, it is truncated and the symbol "/*/" added to indicate missing text. The entries should be separated from one another by blanks.

| | |
|---|---|
| Parameter "%K": | = 0, e.g., channel number with FC10<br>(As substitute for %K, the 2nd digit of the 6-digit alarm number is inserted in the alarm text). Digit count from left to right. |
| Parameter "%A": | = 24, the parameter is replaced by the signal group no. (e.g., axis no., user area no., sequence cascade no.)<br>(As substitute for %A, the 3rd and 4th digits of the alarm number are inserted in the alarm text). |
| Parameter "%N": | = 63, e.g., signal number<br>(As substitute for %N, the 5th and 6th digits of the alarm number are inserted in the alarm text). |
| Parameter "%Z": | e.g., status number for Higraph<br>(As substitute for %Z, the supplementary information for the alarm number is inserted in the alarm text. This parameter is only relevant in conjunction with HIGRAPH programming 800000...899999). |

**Note**

The alarm number is output by the PLC via basic program module FC 10. (This uses PLC system function SFC 52 to send alarms to HMI-Embedded). Supplementary information is not supported.

However, the alarm number can also be transferred to HMI-Embedded by means of PLC system function SFC 17, SFC 18 (ALARM S, ALARM SQ). The mechanisms are used, for example, by the PDIAG configuring tool.

The alarm number (displayed by HMI-Embedded) is transferred via SFC parameter EV_ID. The supplementary information for the alarm number is transferred via SFC parameter SD.

## Cycle alarm texts

Area of freely-assignable texts e.g. for cycle alarms

Table 4-3 Example, cycle alarms

| Alarm number | Display | Help ID | Text or alarm number |
|---|---|---|---|
| 065100 | 1 | 0 | "No D number %1 is programmed" |
| 065101 | 1 | 0 | 600100 |
| 065202 | 0 | 0 | "Axis %2 in channel %1 is still moving" |

The alarm number, display and help ID are as described in the previous example.

### Text or alarm number

The structure of the alarm text/number is the same as that described in the previous example, apart from the parameter assignment:

Parameter "%1": Channel number

Parameter "%2": Block number

## Indexed alarm texts

Indexed alarms can be used to display additional comments in the alarm line, e.g., an explanation for "action =%"

Table 4-4 Example indexed alarms

| Alarm number | Display | Help ID | Text |
|---|---|---|---|
| 010203 | 0 | 0 | "Channel %1 NC Start without reference point (Action=%2<ALNX>)" |
| 016903 | 1 | 0 | "Channel %1 Action=%2<ALNX> not permitted in current status" |
| 016912 | 0 | 0 | "Channel %1 Action=%2<ALNX> only possible in reset state" |

**References:** Diagnostics Manual

# 4.4 Supported languages

Table 4-5 Supported languages

| Language | Language codes | Standard languages | Code page ANSI table (Windows) |
|---|---|---|---|
| Chinese (simplified) | chs | X | 1252 |
| Chinese (traditional) | cht | | 1252 |
| Danish | dan | | 1252 |
| German | deu | X | 1252 |
| English | eng | X | 1252 |
| Finnish | fin | | 1252 |
| French | fra | X | 1252 |
| Italian | ita | X | 1252 |
| Japanese | jpn | | 1252 |
| Korean | kor | | 1252 |
| Dutch | nld | | 1252 |
| Polish | plk | | 1250 |
| Portuguese | ptb | | 1252 |
| Russian | rus | | 1251 |
| Swedish | sve | | 1252 |
| Spanish | esp | X | 1252 |
| Czech | csy | | 1250 |
| Turkish | trk | | 1254 |
| Hungarian | hun | | 1250 |

# 4.5 Bitmaps and icons

## 4.5.1 Storing bitmaps and icons

### Storage structure

Bitmaps and icons are stored in the following folders on the CompactFlash Card, depending on their resolution and the operator panel being used:

| Storage folder | Formats | Resolution | OP (operator panel) |
|---|---|---|---|
| ico640 | *.ico, *.png, *.bmp | 640 x 480 | OP10 |
| ico800 | *.ico, *.png, *.bmp | 800 x 600 | OP12 |
| ico1024 | *.ico, *.png, *.bmp | 1024 x 768 | OP15 |

They are still to be found under different paths, divided into write-protected (standard) and user-specific areas.

**Standard (write-protected)**

The write-protected standard icons are stored in folders (icoxxx) using the following path:

/siemens/sinumerik/hmi/ico/icoxxx/

xxx stands for the relevant resolution, “640”, “800”, “1024”, see table above.

**User**

The user-specific icons are stored in folders (icoxxx) in the following three subdirectories:

| | |
|---|---|
| Standard expansion (AddOn projects) | /addon/sinumerik/hmi/ico/icoxxx |
| Vendor | /oem/sinumerik/hmi/ico/icoxxx |
| User | /user/sinumerik/hmi/ico/icoxxx |

### In-house configuration

Files with the configured screen definition "Expanding the user interface" (wizard "com files" including icons) are located in folders:

/oem/sinumerik/hmi/proj

/user/sinumerik/hmi/proj

Please refer to the sub-book for a detailed description on how to configure bitmaps and icons:

### References

Supplement user interface (BE1)

## 4.5.2 Configuring the user status display

### General

Machine states controlled by the PLC can be displayed in the program status bar via user icons.

The display of user symbols (icons) must be activated using the display machine data MD9052 $MM_SHOW_CHANNEL_SPANNING_STATE spanning state.

The program path of the currently selected program is then displayed in the bar underneath, together with the program name.

16 display positions are defined in the program status bar.

The user icons have to meet the following requirements:

- Colors: 16-color mode
- Size: OP 010/ OP 010C/ OP 010S: 16 x 16 pixels (height x width)
  OP 012: 20 x 20 pixels
  OP 015: 27 x 26 pixels
- File name: 8 characters
- Format: BMP

## Procedure

The cross-channel status display is allocated and configured with user symbols in the HEADER.INI file.

HEADER.INI is located on the CompactFlash card in the following directory:

siemens/sinumerik/hmi/cfg/

1. Copy the HEADER.INI file into the directory oem/sinumerik/hmi/cfg.
2. Use the editor to open the file and assign the user icons to the desired positions.
3. Enter the names of the user icons and the signal for controlling symbol selection in the HEADER.INI file, section "UserIcons".

[UserIcons]

UI_0= <Ikone_00.bmp>, <position>

| | |
|---|---|
| UI_0: | Name |
| Ikone_00.bmp: | Name of user icon |
| Position: | Display position (1 to 16) |

…

UI_31= <Ikone_31.bmp>, <position>

USER_ICON_BASE = DBx.DBBy

| | |
|---|---|
| DBx.DBBy: | Signal for controlling icon selection, defined by the user. |

User icons are addressed bit-by-bit, i.e., if bit n is set in signal DBx DBBy, the user icon with identifier UI_n is displayed.

If the bit is reset by the PLC, the assigned user icon in the program status display is deleted.

If several user icons are assigned to the same position, the user icon with the highest identifier number is displayed. Empty positions do not have to be entered.

## 4.6 Editing user-specific text files

There are two ways to create or edit alarm text files.

1. You can copy the default files on the CompactFlash card from the “siemens” area, edit them as required and store them in your own "oem" or "user" areas.
2. You can copy in-house text files, e.g., files that have already been modified, from a USB FlashDrive and insert them in your own “oem” or “user" areas on the CompactFlash card.

---

**Note**

User-specific text files may only be inserted under the "oem" or "user" directories.

---

| NOTICE |
| --- |
| The entire files must always be stored to prevent the contents of files in different subtrees being compared. |

---

**Note**

If you try to save a copied file to the CompactFlash card by overwriting an existing file of the same name, the message “Internal error” is output.

Delete the previous file of the same name and repeat the action.

---

### Copying files, for example, from a USB FlashDrive

1. If you want to copy data from a USB FlashDrive, first insert the USB FlashDrive in the USB interface on the front of the OP.
2. Press the "Commissioning" softkey.
3. Press the "HMI" softkey.
4. Select the connection, e.g., to the USB FlashDrive using the vertical softkeys.
5. Select the text file using the arrow keys to select the directories and opening them with the “Input” key.
6. Press the "File Function" softkey.
7. Press the "Copy" soft key.
8. Press the "Back" softkey and then select the storage path.

### Inserting files, e.g., onto the CompactFlash card

1. Press e.g. the "Memory Card (CF)" softkey.
2. Select the corresponding directory using the arrow keys to select the directories and opening them with the "Input" key.
3. Press the "File Function" softkey.
4. Press the "Paste" softkey.

### Editing files

1. To edit the selected file, press the “Input” key to open the file.
2. Press the "Close editor" softkey to close the file.

## 4.7 Plaintext for PLC machine data

User-specific and language-dependent plain texts can be configured for each index for so-called PLC machine data.

| | |
|---|---|
| 14510[i] User data (INT) | i = 0 ... 255 |
| 14512[i] User data (HEX) | i = 0 ... 255 |
| 14514[i] User data (FLOAT) | i = 0 ... 31 |

### Storage

The text file must be called "oemtea.txt".

Place the files in the following directory:

user/sinumerik/hmi/lng/eng

<language abbreviation, deu or eng>

### Example of a file

```
//CP=1252
//IDEO=NO
//ANSI=YES
//UNICODE=NO
//VERSION=$VERSION
[Switches]/NATIVE
//--------------
//Texts for user machine data
//--------------
14510 plc 14510
14510[0] md14510_0
14510[1] md14510_1
14510[2] md14510_2
14512 xxxxx 1412
14514 MD14514
```

**Note**

If a machine data item has different indices and you do not specify an index for the plaintext, the same plaintext appears for all the indices of the machine data item.

### Display

The text for the selected machine data item is displayed below the machine data list in the Commissioning operating area.

# Tool management 5

## 5.1 Introduction

For the magazine and PLC configuration, you must create the commissioning file yourself, which is then executed once by the NCK.

The commissioning file is a part program e.g. _N_MAGKONF_MPF.

Two program examples are provided in the toolbox CD, which you can use as template.

---

**Note**

Graphic support for commissioning is not available for tool management with HMI-Embedded.

---

HMI-Embedded supports up to 4 real tool magazines.

### Creating a commissioning file

There are several ways to create the part program:

- Enter at the operator panel (OP 030) of HMI-Embedded.
- Enter at an external PC, using an ASCII editor without formatting.
- Load the example from the toolbox CD and modify at HMI-Embedded or at the PC.
- Create using HMI-Advanced commissioning tool and load into the NCK.

## 5.2 Structure of the commissioning file

The commissioning file is a part program e.g. _N_MAGKONF_MPF.

### Structure of the part program

- Delete data example.
- Define the type of search strategy.
- Define a tool magazine.
- Define a buffer magazine.
- Define a load magazine.
- Define locations of the real tool magazine.
- Define locations of the buffer magazine.
- Define spindle assignment, (which buffer belongs to the spindle).
- Define the locations for the load magazine.
- Define clearances (offset) to the tool magazine. Which spindle, gripper, loading point belong to which tool magazine.

## 5.3 Brief description of the most important variables

An overview of the most important variables is subsequently provided.

### Brief description

$TC_MAP3 Magazine description data
$TC_MAP3[MagazineNo]=status of magazine
Default = 17 means: Active magazine, enabled for loading

$TC_MAMP2 Search strategy
$TC_MAP3[MagazineNo]=status of magazine
This mask is divided into right and left bytes. A value must be specified for both strategies.
The right byte describes the tool search.
The left byte describes the empty location search for the spindle tool.

$TC_MPP1 Location type
$TC_MPP1[MagazineNo, LocNo]=Type of location:
Default: Value corresponding to location type

$TC_MPP2 Location type

$TC_MPP2[MagazineNo, LocNo]=Type of location

Any values can be entered. The values must match the tools to be loaded at the location. Buffers and loading points have the value 0.

$TC_MPP3 Consider adjacent location

$TC_MPP3[MagazineNo, LocNo]= Consider adjacent location ON/OFF

$TC_MPP4 Location state

$TC_MPP4[MagazineNo, LocNo]= Location status (bit mask)

Default=2 Location free

$TC_MPP5 Location type index

$TC_MPP5[MagazineNo, LocNo]= Location type index

For $TC_MPP1[Magazine No., Location No.]=1 (location type is the magazine location), the location number is entered. For other location types, the type index is appropriately incremented:

Example with 2 grippers with location type 3:

- The first gripper has location index 1.
- The second gripper has location index 2.

Clearance between a change location, loading point and the zero points, clearances (offsets) to the magazine.

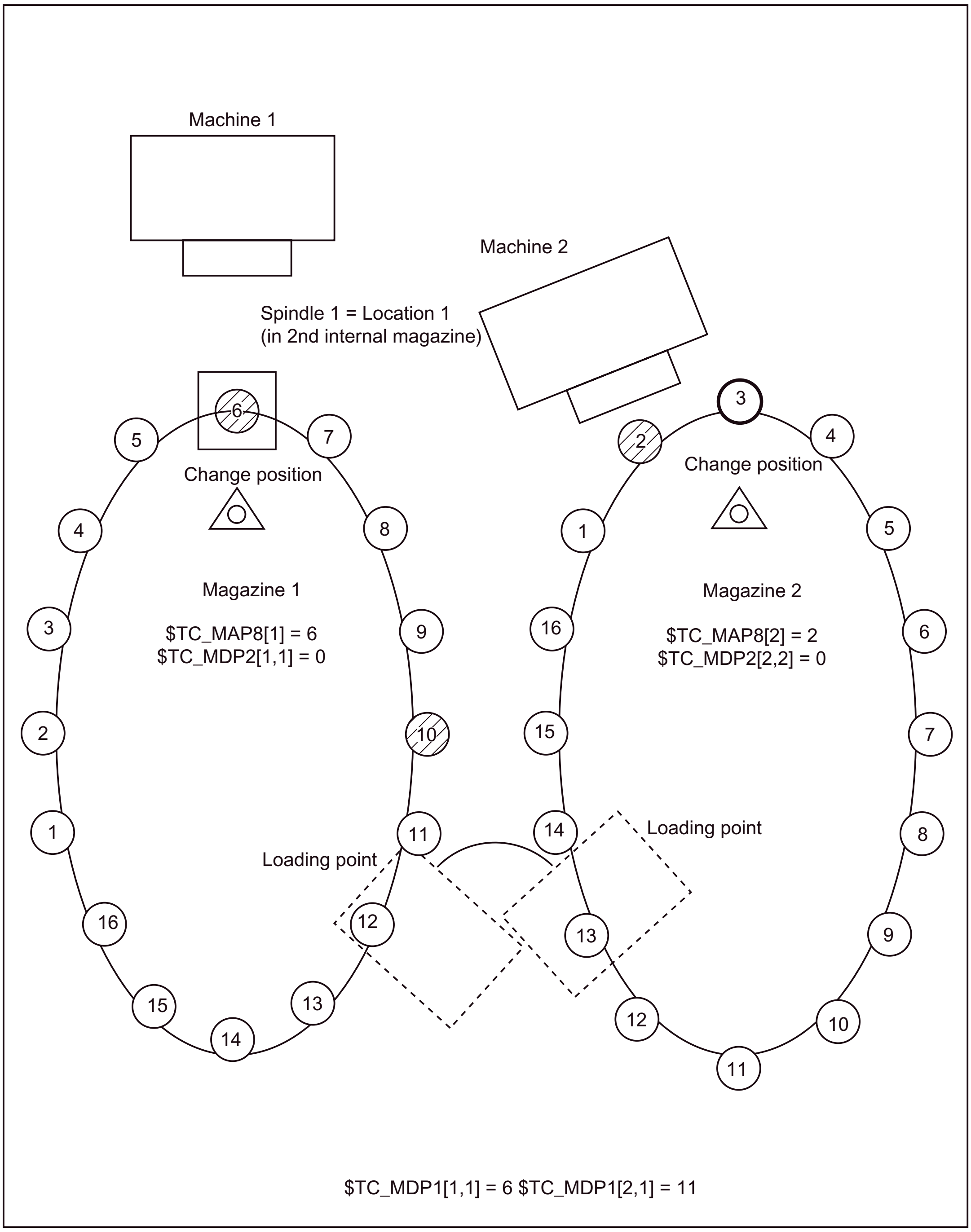
$TC_MDP2 Clearances between buffer location and magazine

$TC_MDP2[magazine No, buffer location No.]

A value must be entered for each buffer, at least a zero. The value is not evaluated at this point, it is only for assignment.

$TC_MDP1 Clearances between loading points and the magazine

$TC_MDP1[magazine No., loading point No.]


Machine 1
Machine 2
Spindle 1 = Location 1
(in 2nd internal magazine)
Change position
Change position
Magazine 1
$TC_MAP8[1] = 6
$TC_MDP2[1,1] = 0
Magazine 2
$TC_MAP8[2] = 2
$TC_MDP2[2,2] = 0
Loading point
Loading point
$TC_MDP1[1,1] = 6 $TC_MDP1[2,1] = 11

The zero position is at the change location of the spindles, therefore the following applies:

If location 1 is at the change location, the current magazine position = 1 = $TC_MAP8[x]

$TC_MDP1[1,1] = 6 Clearance between location 1 of the loading point and the zero position of the magazine.

$TC_MDP1[2,1] = 11 Clearance of the same location from the zero position of magazine 2.

$TC_MDP2[1,1] =0 Clearance of location 1 of the 2nd internal magazine (spindle 1) from the zero position of magazine 1.

$TC_MDP2[2,2] = 0 Clearance between the same location and the zero position of magazine 2

Assignment of magazine locations to spindles.

$TC_MLSR Assignment of magazine locations to spindles

$TC_MLSR [location No. of the buffer, location No. of the spindle in the buffer magazine]

This assigns buffers that have a link between a spindle and the magazines assigned to the spindle. This can be used to define which buffer – e.g. gripper – may execute the tool change in the spindle.

For example, in the diagram, gripper 2 in location 3 can change the tool in the spindle in location 1 ($TC_MLSR[3,1]).

## 5.4 Example of a commissioning file

The following example refers to the following constellation:

1 chain magazine with 50 locations

3 buffer locations

2 loading points

### System configuration

```
%_N_MAGKONF_MPF
;$PATH=/_N_MPF_DIR
N10 ;
N20 ;
N30;
N40 ;
----------------------------------
N50 ; Magazine configuration: MMC100
----------------------------------
N60 ;
N70 ;
N80 ; Delete old data
N90 ;
N100 $TC_MAP1[0]=0
N110 $TC_DP1[0,0]=0
N120 ;
```

```
N130 ; Configuration
N140 ;
N160 $TC_MAMP2=4097 ; Type of search strategy
N170 ;
N180 ; Magazines
N190 ; Real magazine  with number [1]
N200 $TC_MAP1[1]=1 ; Magazine type (1: Chain, 3: Turret,
; 5: Flat magazine)
N220 $TC_MAP3[1]=17 ; Magazine status
N230 $TC_MAP6[1]=1 ; Number of magazine tiers
N240 $TC_MAP7[1]=50 ; Number of magazine locations
N250 ;
N260 ;  Definition of buffer magazine (always number 9998)
N270 $TC_MAP1[9998]=7 ; Magazine type: 7: Buffer
N280 $TC_MAP3[9998]=17 ; Magazine status
N290 $TC_MAP6[9998]=1 ; Number of tiers
N300 $TC_MAP7[9998]=3 ; Number of locations
N310 ;
N320 ; Definition of loading magazine (always number 9999)
N330 $TC_MAP1[9999]=9 ; Magazine type: 9: Loading magazine
N340 $TC_MAP3[9999]=17 ; Magazine status
N350 $TC_MAP6[9999]=1 ; Number of tiers
N360 $TC_MAP7[9999]=2 ; Number of locations
N370 ;
N380 ; Locations of chain magazine
N390 ;
N400 $TC_MPP1[1,1]=1 ; Location type
N410 $TC_MPP2[1,1]=2 ; Location type
N420 $TC_MPP3[1,1]=1 ; Consider adjacent location on (off is 0)
N430 $TC_MPP4[1,1]=2 ; Location state
N440 $TC_MPP5[1,1]=1 ; Location type index
N450 ;
N460 $TC_MPP1[1,2]=1
N470 $TC_MPP2[1,2]=2
N480 $TC_MPP3[1,2]=1
N490 $TC_MPP4[1,2]=2
N500 $TC_MPP5[1,2]=2
N510 ;
N520 $TC_MPP1[1,3]=1
N530 $TC_MPP2[1,3]=2
N540 $TC_MPP3[1,3]=1
N550 $TC_MPP4[1,3]=2
N560 $TC_MPP5[1,3]=3
N570 ;
N580 $TC_MPP1[1,4]=1
N590 $TC_MPP2[1,4]=2
N600 $TC_MPP3[1,4]=1
N610 $TC_MPP4[1,4]=2
N620 $TC_MPP5[1,4]=4
N630 ;
N640 $TC_MPP1[1,5]=1
N650 $TC_MPP2[1,5]=2
N660 $TC_MPP3[1,5]=1
N670 $TC_MPP4[1,5]=2
N680 $TC_MPP5[1,5]=5
```

```
N690 ;
......
......
N3160 $TC_MPP1[1,47]=1
N3170 $TC_MPP2[1,47]=2
N3180 $TC_MPP3[1,47]=1
N3190 $TC_MPP4[1,47]=2
N3200 $TC_MPP5[1,47]=47
N3210 ;
N3220 $TC_MPP1[1,48]=1
N3230 $TC_MPP2[1,48]=2
N3240 $TC_MPP3[1,48]=1
N3250 $TC_MPP4[1,48]=2
N3260 $TC_MPP5[1,48]=4
8N3270 ;
N3280 $TC_MPP1[1,49]=1
N3290 $TC_MPP2[1,49]=2
N3300 $TC_MPP3[1,49]=1
N3310 $TC_MPP4[1,49]=2
N3320 $TC_MPP5[1,49]=49
N3330 ;
N3340 $TC_MPP1[1,50]=1
N3350 $TC_MPP2[1,50]=2
N3360 $TC_MPP3[1,50]=1
N3370 $TC_MPP4[1,50]=2
N3380 $TC_MPP5[1,50]=50
N3390 ; locations of the buffer
N3400 $TC_MPP1[9998,1]=2 ; Location type (here spindle)
N3410 $TC_MPP2[9998,1]=0 ; Location type: as the buffer is 0 here
N3420 $TC_MPP3[9998,1]=0 ; Consider adjacent location off
N3430 $TC_MPP4[9998,1]=2 ; Location state
N3440 $TC_MPP5[9998,1]=1 ; Location type index
N3450 ;
N3460 $TC_MPP1[9998,2]=3 ; Gripper 1
N3470 $TC_MPP2[9998,2]=0
N3480 $TC_MPP3[9998,2]=0
N3490 $TC_MPP4[9998,2]=2
N3500 $TC_MPP5[9998,2]=1
N3510 ;
N3520 $TC_MPP1[9998,3]=3 ; Gripper 2
N3530 $TC_MPP2[9998,3]=0
N3540 $TC_MPP3[9998,3]=0
N3550 $TC_MPP4[9998,3]=2
N3560 $TC_MPP5[9998,3]=2
N3870 ;
N3880 ; Spindle assignment ; Spindle assignment
```

| | |
|---|---|
| N3890 $TC_MLSR[2,1]=0; | 1. Gripper (location 2) belongs to spindle (location 1) |
| N3900 $TC_MLSR[3,11]=0; | 2. Gripper (location 3) belongs to spindle (location 1) |

```
N3920 ; Loading magazine locations
N3930 $TC_MPP1[9999,1]=7 ; Location type loading point (for
spindle!)
N3940 $TC_MPP2[9999,1]=0; Location type (here always 0)
```

```
N3950 $TC_MPP3[9999,1]=0; Consider adjacent location off! N3960
$TC_MPP4[9999,1]=2; Location status: Unassigned
iN3970 $TC_MPP5[9999,1]=1; Location type index
N3980 ;
N3990 $TC_MPP1[9999,2]=7
N4000 $TC_MPP2[9999,2]=0
N4010 $TC_MPP3[9999,2]=0
N4020 $TC_MPP4[9999,2]=2
N4030 $TC_MPP5[9999,2]=2
N4040 ;
N4650 ; Offsets (clearances) ; Clearances to magazine
N4660 ;
N4670 $TC_MDP2[1,1]=0 ; Spindle
N4680 $TC_MDP2[1,2]=0 ; Gripper 1
N4690 $TC_MDP2[1,3]=0 ; Gripper 2
N4700 $TC_MDP1[1,1]=0 ; 1st loading point
N4710 $TC_MDP1[1,2]=25 ; 2nd loading point (clearance 25 to actual
position)
N4720 ;
N4730 ; End
N4740 ;
N4750 M30
```

## 5.5 Loading and activating commissioning file

### Commissioning file created on an external PC

Place the externally created commissioning file into directory _N_MPF_DIR.

The file must be started as part program to activate the commissioning file in the NC:

- Select the part program, e.g. _N_MAGKONF_MPF.MPF
- Execute the program with NC Start.

### Create PLC data with HMI-Embedded

The data relevant for booting are located in DB 4 from data word 64 onwards

This data must be described by the PLC user program.

**References:** Function Manual, Tool Manager: Signal description of PLC data

The number of magazines, loading points, spindles and turrets is determined from this data and used to automatically set up the tool management data blocks (DB 71 to DB 74).

Booting is part of the basic program.

## 5.6 Selecting tool management

The tool manager ShopTurn, ShopMill is activated using display machine data MD9414 $MN_TM_KIND_OF_TOOLMANAGEMENT to display ShopMill/ ShopTurn.

1 = ShopMill/ ShopTurn tool management

0 = Standard tool management (default setting)

All of the information required for commissioning can be found in the following documents:

### References

Function Manual, Tool Manager

CNC Commissioning Manual: ShopMill

CNC Commissioning Manual: ShopTurn

# 6 Series start-up

## 6.1 Introduction

In order to transfer a particular configuration to further controls of the same software version as simply as possible (e.g., those that are operated on the same machine type), you can create what are called series startup files.

1. Series startup with or without compensation data
2. Area—specific archiving:
   - PLC data
   - Drive data

This type of file cannot be modified externally using an ASCII editor.

## 6.2 Output of data

### Procedure

1. Select the "Services" operating area.
2. Press the “ETC” key.
3. Press the horizontal softkey "Series commissioning".
   A window opens in which you can make the following selection:
   - NCK data – with compensation data
   - Drive data
   - PLC data
4. Use the arrow keys to select an area (.arc) and mark it with the "Select" key.
5. Start the “Output data” operation by pressing the “Generate Archive” softkey.
6. The lower part of the window contains the "Save as..." field and the drives that have been set-up are displayed as storage location.
7. Using the arrow keys, mark the storage location, e.g. "memory card (CF)" (CompactFlash card).
8. Press the "OK" softkey.

9. The storage directory, e.g.: on the card: user/sinumerik/data/archive and the file, e.g. "PLC.ARC" are displayed. Press the "OK" softkey.
10. Now, you can assign a different name.
11. Press "OK" again to start the data transmission.

    **Note**

    When the drive data is selected, it takes 1 to 2 minutes after pressing the "Create archive" softkey until the drive has saved the data on the CompactFlash Card.

**Note**

When you save very large files to the CompactFlash card, the message "Please wait - file being flashed" appears. Acknowledge the message with the "Recall" button.

## 6.3 Importing data

To import the backed up data to another control, proceed as follows:

### Procedure

1. In the “Services” area, select the created archive (arc.) and press the “Read in archive” vertical softkey. The "Read in update archive" window pane opens.
2. Press the "OK" softkey to start importing.
3. Press the " ˆ " key to return to the previous window.
4. After importing an NC series commissioning archive you must restart HMI-Embedded.
5. Press the "NCK Reset" vertical softkey in the "Commissioning" operating area. NCK and drive are restarted.
   Or press the CRT and Q keys and then Enter.

### Protocol

Press the "Log" softkey to view the transmission report on the screen.

# Data Backup 7

## 7.1 Introduction

### Execution

You should backup your data

- after commissioning,
- after changing machine-specific settings,
- after service (e.g., after replacing hardware or software), in order to resume operation quickly
- during commissioning, whenever you change the memory configuration.

The modified data can be stored over all furnished drive connections.

### Required accessories

You will require the following accessories in order to save data:

- Keyboard, for upper and lower case.

### Prerequisite

- To store data to the CompactFlash card you need the license release for the additional 256 MB HMI user memory.
- All configured network drives must be permanently accessible while HMI-Embedded is running.

### Save area

- The area indicates which data are to be backed up or retrieved (general, channel-specific, or axis-specific).
- The unit defines the channel, the axis or the TOA area. The unit does not have to be specified if the entire area is selected.
- The type determines the data type. During a data backup, the file names are created and output automatically.

Areas

| | |
|---|---|
| NC | General NC-specific data |
| CH | Channel-specific data (unit corresponds to the channel number) |
| AX | Axis-specific data (unit corresponds to the number of the machine axis) |
| TO | Tool data |
| COMPLETE | All data of an area |
| INITIAL | Data for all areas (_N_INITIAL_INI) |

Types

| | |
|---|---|
| TEA | Machine data |
| SEA | Setting data |
| OPT | Option data |
| TOA | Tool data |
| UFR | User input frames: Settable Work offset, rotations, etc. |
| EEC | Measuring system error compensation |
| CEC | Sag/angularity compensation |
| QEC | Quadrant error compensation |
| PRO | Protection zone |
| RPA | R parameters |
| GUD | Global user data |
| INI | General initialization program (all data in the active file system) |

| | |
|---|---|
| _N_COMPLETE_TEA | Archiving of all machine data |
| _N_AX_TEA | Archiving of all axis machine data |
| _N_CH1_TEA | Archiving of the machine data for channel 1 |
| _N_CH1_GUD | Archiving of the global user data for channel 1 |
| _N_INITIAL_INI | Archiving of all data in the active file system |

References: CNC Commissioning Manual: NCK, PLC, Drive

## 7.2 Data backup via user interface

### Save archive

For SINUMERIK 840D sl, the data backup of various components is divided as follows:

1. Data backup for NCK with/without compensation data
2. Data backup for PLC
3. Drive data

The procedure for this data backup is described in the previous chapter.

See Output of data (Page 97)

### Save files

- In the "Services" and "Commissioning" operating areas, you can save individual directories or files/programs on the configured drives using the "Copy" and "Paste" softkeys.
- If you edit files or programs in the operating area commissioning, softkey "Commissioning", then you can save the changes also using the "USB front" softkey directly on the USB FlashDrive that you have inserted in the front USB interface.
- Further, in the operating areas program, services and commissioning, you can save data on all of the configured drives (logical drives).

# 7.3 Data backup using machine data

## Backing-up modified values

When backing-up machine and setting data, the general machine data MD11210 $MN_UPLOAD_MD_CHANGES_ONLY (only modified MD data are backed-up) can be used to define whether all data are output or only data that deviate from the default setting.

If a value of a data item that is stored as an array is changed, the entire MD array is always output (e.g MD10000 $MN_AXCONF_MA-CHAX_NAME_TAB).

| 11210<br>MD number | UPLOAD_MD_CHANGES_ONLY<br>MD backup of changed MD only | |
|---|---|---|
| Default setting: 0 | Min. input limit: 0 | Max. input limit: 255 |
| Change effective: Immediately | Protection level: 2/4 | Units: – |
| Data type: BYTE | | |
| Significance | Selection of differential MD upload:<br>Bit 0 (LSB) Scope of the differential upload with TEA files (area-by-area archiving)<br>• 0: All data are output<br>• 1: Only machine data are output that deviate from the standard (this does not apply to INITIAL_INI)<br>If a value is changed for a piece of data that is saved as array, then the complete MD array is always output<br>(e.g. MD10000 $MN_AXCONF_MACHAX_NAME_TAB). | |
| | Bit 1 Scope of the differential upload with INI files<br>• 0: All data are output<br>• 1: Only data that deviate from the standard are output<br>(e.g. INITIAL_INI) | |
| | Bit 2 Change to an array element<br>• 0: Complete array is output<br>• 1: Only modified array elements are output | |
| | Bit 3 R parameters (only for INITIAL_INI)<br>• 0: All R parameters are output<br>1: Only R parameters not equal to zero are output. | |
| | Bit 4 Frames (only for INITIAL_INI)<br>• 0: All frames are output.<br>• 1: Only frames not equal to zero are output | |
| | Bit 5 Tool data, cutting edge parameters (only for INITIAL_INI)<br>• 0: All tool data are output<br>• 1: Only tool data not equal to zero are output | |
| | Bit 6 Retentive system variables<br>($AC_MARKER [ ]; $AC_PARAM [ ] only for INITIAL_INI)<br>• 0: All system variables are output<br>• 1: Only system variables not equal to 0 are output | |

| 11210<br>MD number | UPLOAD_MD_CHANGES_ONLY<br>MD backup of changed MD only |
|---|---|
| | Bit 7 Synchronized action GUD (only for INITIAL_INI)<br>• 0: All synchronized action GUD are output<br>• 1: Only synchronized action GUD not equal to zero are output |
| | Effective: Changes to the data become effective when the upload for the next area is started. |

**Note**

It might be sensible to back up only altered machine data before a software update if changes have been made to the default machine data settings in the new software version. This applies particularly to machine data that are assigned SIEMENS protection level 0.

**Note**

MD11210 $MN_UPLOAD_MD_CHANGES_ONLY should be set to "1" or the corresponding bits set to "1". With this setting, the transferred files contain only those data, which deviate from the default.

# Diagnostics/Service 8

## 8.1 Software version display

The version data of the installed system software are output in a version display.

### Proceed as follows

1. Select the "Diagnosis" operating area.
2. Press the softkeys "Service Displays" → "Version".
3. You can display the version data of the following areas via the horizontal softkeys:
   – Version data of the NCU
   – Version data of the HMI
   – Version data of the cycles (user, manufacturer, standard cycles)
   – Definitions
   – Compile cycles

## 8.2 Displaying and Editing System Resources

For the NCK and HMI-Embedded areas, you can display the following currently used system resources (utilization display) - and you can also change these.

### Prerequisite

As different protection levels are assigned to machine data per default, the set access authorization for the editing of machine data must be sufficiently high. Depending on the authorization level, it will be possible to edit data such as cycles and machine data.

### Procedure

1. In the "Commissioning" operating area press the ">" ETC key
2. Press the "NC memory" softkey. The screenform for the memory overview with the current memory allocation of the NC user memory for programs and data.
   The following areas are available to you for viewing and editing memory-configuring machine data in detail.
   – Static user memory (SRAM)
   – Dynamic user memory (DRAM)

### 8.2.1 Static user memory (SRAM)

### SRAM allocation

When you select the "SRAM" softkey, the total memory allocation is displayed in the upper part of the window. The content of the data register can be modified directly in the display.

The memory-configuring machine data are divided into the following groups:

- Tool management
- Global user data
- Curve tables
- Compensations
- File system / program memory
- Protection zones

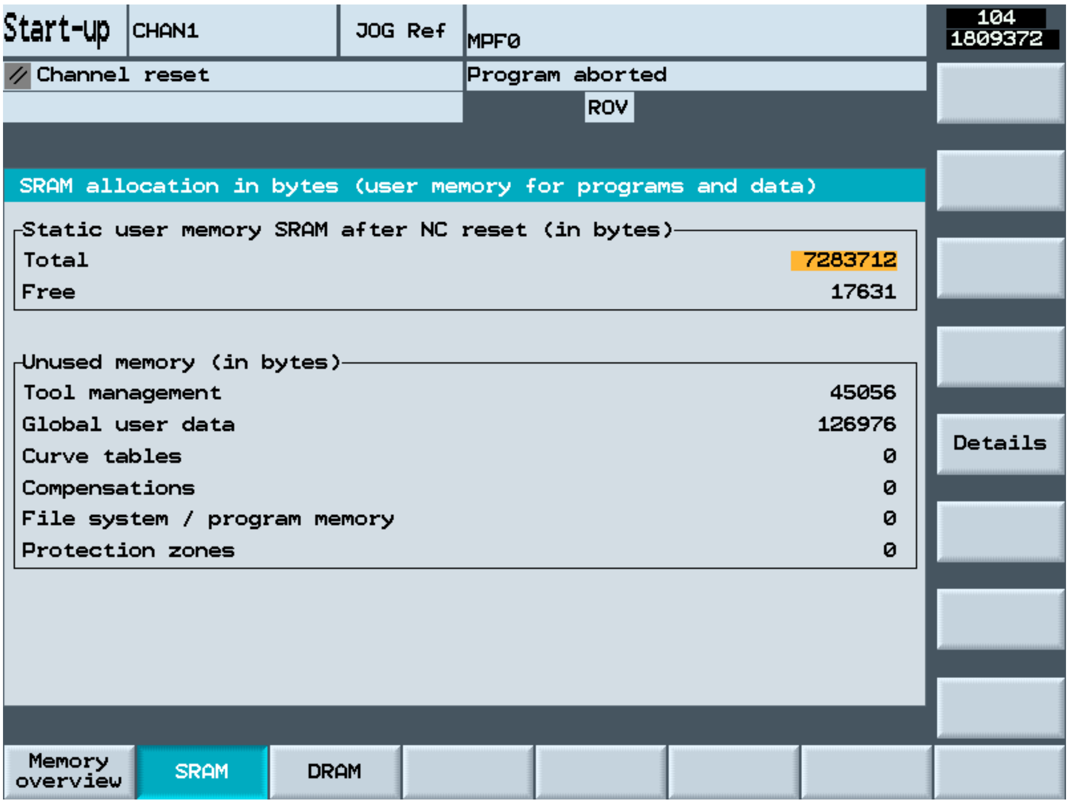



Figure 8-1 Static user memory (SRAM)

## Detailed view

To display the current values, use the cursor to select the required area and press the "Details" softkey.

- The block header contains the number and name of the machine data.
- The currently set values are output on the lines below.
- You can change the memory setting in the gray field, after "New value" or "New number", to the right of the field containing the actual value. This new setting is entered temporarily. Your entry is automatically checked for limits and an appropriate message output in the dialog line if you make a mistake.
- The total available memory, taking into account any modified values, is updated in the lower part of the screen.
  - The "Axis+" and "Axis-" softkeys are displayed for axis-specific machine data.
  - The "Channel+" and "Channel-" are softkeys displayed for channel-specific machine data.

## Data backup

Press the "Accept" softkey to transfer the temporary values of the displayed machine data to the NC.

Press the "Cancel" softkey or the Recall key "^" to exit the detailed view without transferring the change.

---

**Note**

Once you have pressed the "Accept" softkey, the "Cancel" softkey is no longer operative, i.e., any changes you have made cannot be undone.

---

When you change the allocation of a memory, alarm 4400 "Machine data alteration will cause reorganization of buffered memory (data loss)" appears.

Data are not backed up automatically, so to avoid data loss you must back up the data yourself by performing an NCK reset before you transfer the MD change.

## Memory allocation: Tool management

The following window is displayed by selecting the "Tool management" area and pressing the "Details" vertical softkey:

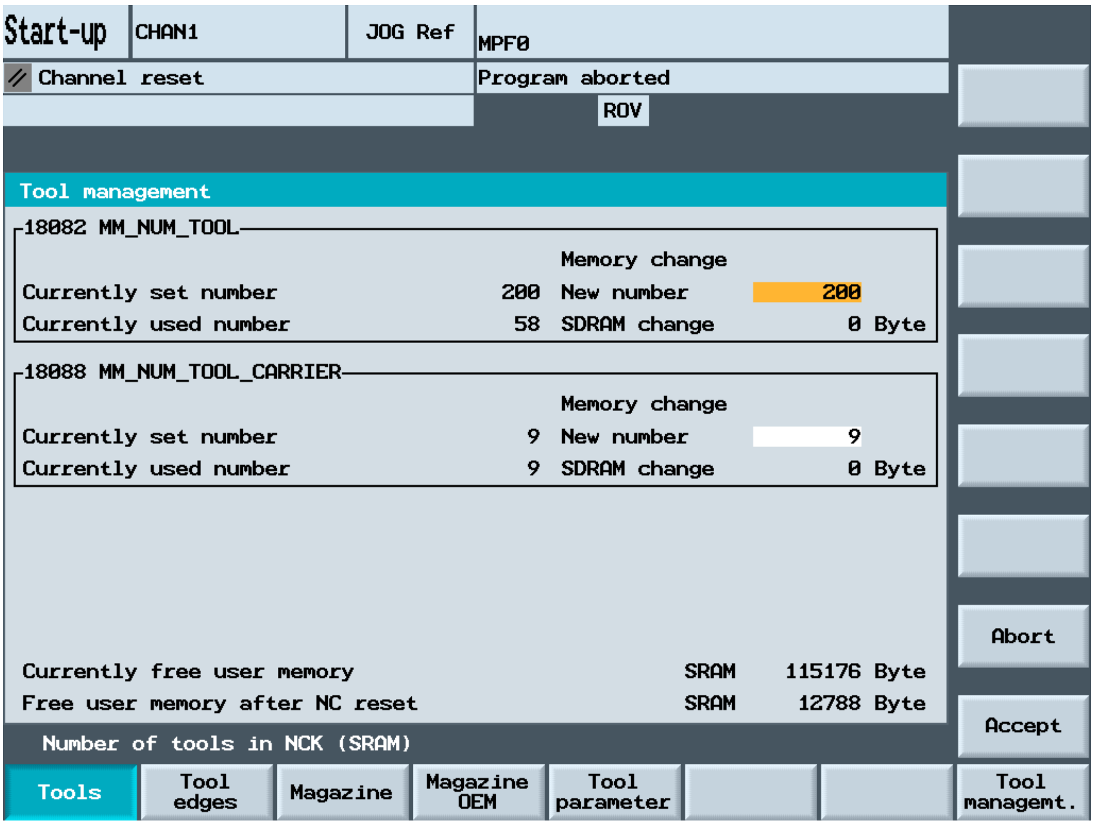



Figure 8-2 SRAM tools

By selecting the horizontal softkeys, you can view or change the memory values of general machine data of other subgroups.

| | |
|---|---|
| "Tools": | MD 18082: MM_NUM_TOOL<br>MD 18088: MM_NUM_TOOL_CARRIER |
| "Tool edges": | MD 18100: MM_NUM_CUTTING_EDGES_IN_TOA<br>MD 18110: MM_MAX_SUMCORR_PER_CUTTEDGE<br>MD 18104: MM_NUM_TOOL_ADAPTER |
| "Magazines": | MD 18084: MM_NUM_MAGAZINE<br>MD 18086: MM_NUM_MAGAZINE_LOCATION |
| "Magazine OEM": | MD 18090: MM_NUM_CC_MAGAZINE_PARAM<br>MD 18092: MM_NUM_CC_MAGLOC_PARAM |
| "Tool parameter": | MD 18094: MM_NUM_CC_TOA_PARAM<br>MD 18096: MM_ NUM_CC_TDA_PARAM<br>MD 18098: MM_ NUM_CC_MON_PARAM |
| "Tool management": | MD 18080: MM_TOOL_MANAGEMENT_MASK |

## Memory allocation: Global user data

| | |
|---|---|
| "GUD files": | MD 18118: MM_NUM_GUD_MODULES<br>MD 18150: MM_GUD_VALUES_MEM |
| "Number of variables": | MD 18120: MM_NUM_GUD_NAMES_NCK<br>MD 18130: MM_NUM_GUD_NAMES_CHAN |

## Memory allocation: Curve tables

MD 18400: MM_NUM_CURVE_TABS
MD 18402: MM_NUM_CURVE_SEGMENTS
MD 18404: MM_NUM_CURVE_POLYNOMS

## Memory allocation: Compensations

| | |
|---|---|
| "Sag compens.": | MD 18342: MM_NUM_CEC_MAX_POINTS[ ] |

These configuring machine data are indexed machine data. The index is set in the top half of the screen.

| | |
|---|---|
| "E. error compens.": | MD 38000: MM_ENC_COMP_MAX_ POINTS[0] |
| | MD 38000: MM_ENC_COMP_MAX_ POINTS[1] |

There are only two tables for the encoder/spindle compensation interpolation points. There is no need to select an index.

| | |
|---|---|
| "Q. error compens.": | MD 38010: MM_QEC_MAX_POINTS[0] |

There is only one table for quadrant error compensation. There is no need to select an index.

## Memory allocation: File system / program memory

| | |
|---|---|
| "Directories": | MD 18310: MM_NUM_DIR_IN_FILESYSTEM |
| | MD 18270: MM_NUM_SUBDIR_PER_DIR |
| "Files": | MD 18320: MM_NUM_FILES_IN_FILESYSTEM |
| | MD 18280: MM_NUM_FILES_PER_DIR |

## Memory allocation: Protection zones

| | |
|---|---|
| "Global prot. zone": | MD 18190: MM_NUM_PROTECT_AREA_NCK |
| "Chann-spec. prot. zone": | MD 28200: MM_NUM_PROTECT_AREA_CHAN |
| | MD 28210: MM_NUM_PROTECT_AREA_ACTIVE |

## 8.2.2 Dynamic user memory (DRAM)

### DRAM allocation

When you press the "DRAM" softkey, the total memory allocation is displayed in the upper part of the window. The content of the data register can be modified directly in the display.

To help you optimize memory utilization effectively, the memory-configuring machine data are divided into the following groups:

- Local user data
- REORG
- Cycles
- Interpolation buffer
- Execution from external
- Synchronized actions
- Macros
- Tool management
- Protection zones

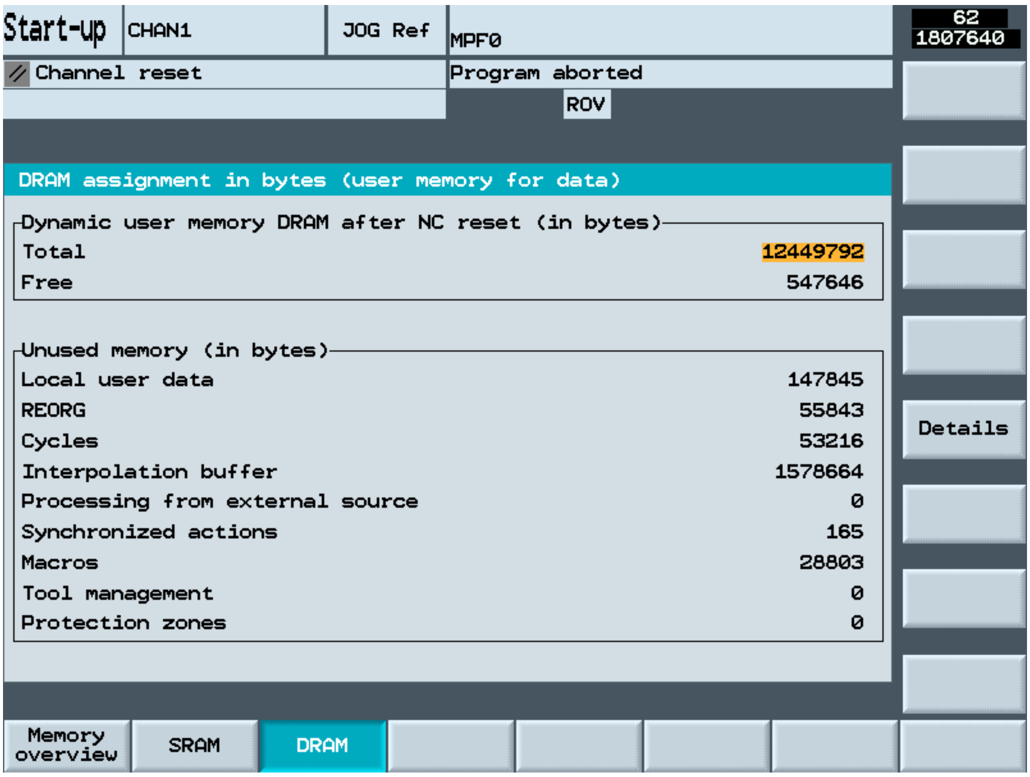



Figure 8-3 Dynamic user memory DRAM

## Detailed view

To display the current values, use the cursor to select the required area and press the "Details" softkey.

## Peak values

Additional elements in DRAM memory allocation screen:

Since the memory of some data areas is allocated to dynamic processes, their memory requirements vary. The values of the displayed machine data are set to zero using the "Peak Values=0" softkey.

## Memory allocation: Local user data

MD 28020: MM_NUM_LUD_NAMES_TOTAL

MD 28040: MM_LUD_VALUES_MEM

## Memory allocation: Reorganization

MD 28000: MM_REORG_LOG_FILE_MEM

MD 28010: MM_NUM_REORG_LUD_MODULES

## Memory allocation: Cycles

MD 18170: MM_NUM_MAX_FUNC_NAMES

MD 18170: MM_NUM_MAX_FUNC_PARAM

## Memory allocation: Interpolation buffer

MD 18360: MM_EXT_PROG_BUFFER_SIZE

MD 28070: MM_NUM_BLOCKS_IN_PREP

## Memory allocation: Execution from external

MD 18360: MM_EXT_PROG_BUFFER_SIZE

MD 18362: MM_EXT_PROG_NUM

### Memory allocation: Synchronized Actions

"Synchronized action 1": MD 28250: MM_NUM_SYNC_ELEMENTS
MD 28252: MM_NUM_FCTDEF_ELEMENTS
MD 28258: MM_NUM_AC_TIMER

"Synchronized action 2": MD 28254: MM_NUM_AC_PARAM
MD 28256: MM_NUM_AC_MARKER

### Memory allocation: Macros

MD 18160: MM_NUM_USER_MACROS

### Memory allocation: Tool management

MD 18105: MM_MAX_CUTTING_EDGE_NO
MD 18106: MM_MAX_CUTTING_EDGE_PERTOOL

### Memory allocation: Protection zones

See Chapter: Static user memory (SRAM), memory allocation: Protection zones.

# 8.3 Action log

## 8.3.1 Set action log

### General

The function "action log" is available for reconstructing operation sequences at a later date.

You can set which functions and events are to be logged via the HMI user interface.

### Prerequisite

The action log is password-encrypted (protection level ≥ equal to 3).

Operating area "Start-up" must be installed.

## Switching functions on and off via the HMI interface

You reach the "Action Log Settings" menu via the "HMI" → "Action Log" softkeys in the "Commissioning" operating area. The following information and selection options are available:

- Logging on
- Path of the log file: /card/user/sinumerik/hmi/action.com (not modifiable)
- Size of log file: 5000000 (default setting, in bytes)
- Write delay for file:
  - 1: (Default setting) Log entries are only written to the CompactFlash card if the internal buffer is full. If the control system is switched off, entries in the internal buffer may be lost.
    If a PLC crash signal is output or the action log is activated/deactivated, the entries are always saved.
  - 0: These log entries are stored consecutively on the CompactFlash card. This can impair its performance and reduce its service life.
  - >0: Time (in seconds), after which the internal buffer is written to the CompactFlash card.
- Program status when an alarm occurs: Input of alarms. These alarms directly trigger writing of the current action log to the CompactFlash card. If several alarm numbers are entered, they must be separated by commas.

## Log events

If the action log is switched on, the following log events are activated:

| | |
|---|---|
| Interrupts | Logging all incoming and outgoing alarms/messages of the NC/PLC and HMI-Embedded |
| Keys | Logging of all keyboard actions |
| Channel status | NC/PLC states are logged via the information channel status. At least in parts, if they can be recorded by time, these statuses are used to verify the operation of the MCP. |
| Window switchover | Logging of all movements between windows (current operating area, ID number) |
| Writing NCK/PLC data | Logging of all modified NCK and PLC values,<br>for example, modified access to geometric data, such as tool offset and zero offset |
| PI Services (program invocation) | Logging of all PI services, which affect the NC-program workflow, for example, Program selection, Delete file, Delete tool, etc. |

The following current program statuses are logged under "Curr. program status":

| | |
|---|---|
| Critical alarms (acknowledgment >= NC reset),<br><br>PLC crash signal: | Curr. channel status<br>Curr. program layer<br>Curr. actual values of axes<br>Curr. tool<br>Curr. G function<br>Curr. M function<br>Curr. zero offset |
| NC channel reset | Only subsets of the above events are logged. |
| Tool change | Only subsets of the above events are logged. |

**Note**

Changes to the override are only logged if the PLC crashes or on a critical alarm with the IPO trace.

**Note**

When operating more than one HMI on an NCU (e.g., HMI-Embedded and HMI-Advanced), the action log must only be activated on one HMI.

## 8.3.2 Structure of the log file

### Overview

When the action log is activated, the following log files are generated:

- Log file "action.com"
- Log file "crash.com"

Both are binary files and are constantly overwritten as ring buffers. When the log files are displayed, these binary files are converted into readable "action.log" and "crash.log" files. The entries appear in English and are non-language-specific. The events follow on from general information, starting with the most recent event.

#### Log file "crash.com"

The "crash.com" file is empty. Data are written to it after the following events:

- Interface signal DB19.DBX0.6 "Save teleprinter log" changes from 0 to 1.
- The alarm entered in the "Program status on alarm" field is output.
- Even when the action log is deactivated, alarms are still logged in the "alarm.com" log file.

## Content of log files

The log files contain the following information:

- HMI version and NCK version
- Action-log version and log-file version
- Types of logged entry. The following data are recorded for every logged event:
  - Date and time of day
  - User name
  - Entry level
  - Entry-type designation
- Event data are written in plain language.

## Example of an "action.log" file:

```
HMI Version: V07.20.01.00         NCK Version: 660000
ActionLog-Version: x              Logfile-Version: y
Activated Entries: HMI_START HMI_EXIT PLC_CRASH PLC_CRASH_ALARM
KEY_PRESSED KEY_PRESSED KEY_PRESSED KEY_RELEASED KEY_RELEASED
KEY_RELEASED ALARM ALARM_QUIT OPEN_WINDOW OPEN_WINDOW
OPEN_WINDOW CLOSE_WINDOW CLOSE_WINDOW CLOSE_WINDOW
CH_STATE_CHANGED OPMODE_CHANGED TOOL_CHANGED OVERRIDE PI_CMD
DOM_CMD DOM_CMD DOM_CMD WRITE_VAR WRITE_VAR WRITE_VAR FINDBL_CMD
OVERSTORE FILE_ACCESS AREA_CHANGED PROG_CONTROL_CHANGED ALARM
USER ACTIVATED DEACTIVATED SUSPEND RESUME
--- Date         Time             User             L Entry-Id
Entry
-------------------------------------------------------------------
-----
```

```
--- 04.10.2006 09:35:40 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Overview (program / work piece)" (Id 35100, Appl.
3)
--- 04.10.2006 09:35:40 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Dummy-Window" (Id 1000, Appl. 3)
--- 04.10.2006 09:35:40 HMI-Emb 0 KEY_PRESSED
Key pressed: KEY_F5 (RELEASED) (e0/3f)
--- 04.10.2006 09:35:39 HMI-Emb 0 KEY_PRESSED
Key pressed: KEY_F5 (0/3f)
--- 04.10.2006 09:35:38 <default> 0 ALARM
NCK Alarm: 8080, cleartype Alarm-Cancel
7 option(s) is/are activated without setting the license key
--- 04.10.2006 09:35:38 HMI-Emb 0 OPEN_WINDOW
Window opened: "Dummy-Window" (Id 1000, Appl. 3)
--- 04.10.2006 09:35:38 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Notebooks screen" (Id 5421, Appl. 3)
--- 04.10.2006 09:35:38 HMI-Emb 0 KEY_PRESSED
Key pressed: KEY_SELECT_MODE (0/44)
--- 04.10.2006 09:35:32 HMI-Emb 0 PI_CMD
PI-Command executed: _N_F_XFER "/_N_MPF_DIR"
--- 04.10.2006 09:35:31 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Program editor" (Id 35200, Appl. 3)
--- 04.10.2006 09:35:31 HMI-Emb 0 KEY_PRESSED
Key pressed: KEY_F8_V (0/5b)
--- 04.10.2006 09:35:27 HMI-Emb 0 PI_CMD
PI-Command executed: _N_F_OPEN "/_N_MPF_DIR/_N_NEW_FILE_MPF",
"_N_1_EDI"
--- 04.10.2006 09:35:26 HMI-Emb 0 OPEN_WINDOW
Window opened: "Program editor" (Id 35200, Appl. 3)
--- 04.10.2006 09:35:26 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Overview (program / work piece)" (Id 35100, Appl.
3)
--- 04.10.2006 09:35:26 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Overview window" (Id 36070, Appl. 3)
--- 04.10.2006 09:35:25 HMI-Emb 0 WRITE_VAR
NCK/PLC Var. wrote: PLC:/DB19.DBX20.1 = 0
--- 04.10.2006 09:25:01 HMI-Emb 0 TOOL_CHANGED
Active tool in channel 1 changed to T0, D1
Active tool in channel 1 changed to T0, D1
Mode: AUTO Program: running Channel: active
Program-Level information:
Level Program Invoc Offset
running:
1 /_N_MPF_DIR/_N_RALF_MPF 1 5
stopped:
1 /_N_MPF_DIR/_N_RALF_MPF 1 #

Actual Block:
t0

x1

t1
```

```
Number of Machine axis: 5
Increment: continuous Increment
MCS-Name Position DistToGo
X1 1.000 0.000
Y1 0.000 0.000
Z1 0.000 0.000
A1 0.000 0.000
B1 0.000 0.000
WCS-Name Position DistToGo Position ENS DistToGo ENS
X 1.000 0.000 1.000 0.000
Y 0.000 0.000 0.000 0.000
Z 0.000 0.000 0.000 0.000
A 0.000 0.000 0.000 0.000
B 0.000 0.000 0.000 0.000
Feed Rate: 0.000 Set: 0.000
Spindle Rate: 0.000 Set: 0.000
Actual Tool T0, D1, Type 0
Geometry -- tool type 0
Geometry -- tool point direction 0.000
Geometry -- length 1 0.000
Geometry -- length 2 0.000
Geometry -- length 3 0.000
Geometry -- radius 0.000
Geometry -- corner radius 0.000
Geometry -- length 4 0.000
Geometry -- length 5 0.000
Geometry -- angle 1 0.000
Geometry -- angle 2 0.000
Wear -- length 1 0.000
Wear -- length 2 0.000
Wear -- length 3 0.000
Wear -- radius 0.000
Wear -- slot width b 0.000
Wear -- proj. length k 0.000
Wear -- length 5 0.000
Wear -- angle 1 0.000
Wear -- angle 2 0.000
Adapter -- length 1 0.000
Adapter -- length 2 0.000
Adapter -- length 3 0.000
Relief angle 0.000
Manual Turn: Cutting rate 0.000
Transformation active:
Active toolholder 0
Active G-Codes: G01 STARTFIFO G17 G40 G500 G60 G601 G71 G90 G94
 CFC NORM G450 BNAT ENAT BRISK CUT2D CDOF FFWOF ORIWKS RMI ORIC
 WALIMON DIAMOF COMPOF G810 G820 FTOCOF OSOF SPOF PDELAYON
 FNORM SPIF1 CPRECOF CUTCONOF LFOF TCOABS G140 G340 SPATH LFTXT
 G290 G462 CP ORIEULER ORIVECT PAROTOF TOROTOF ORIROTA RTLION
 TOWSTD FENDNORM RELIEVEON DYNNORM WALCS0
```

```
Active M-Codes:
--- 04.10.2006 09:24:57 HMI-Emb 0 CH_STATE_CHANGED
Channel State of Channel 1 changed to active
--- 04.10.2006 09:24:57 HMI-Emb 0 OPEN_WINDOW
Window opened: "Machine configuration" (Id 13392, Appl. 6)
--- 04.10.2006 09:24:57 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Dummy-Window" (Id 20005, Appl. 1)
--- 04.10.2006 09:24:57 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Position - Work" (Id 20040, Appl. 1)
--- 04.10.2006 09:24:57 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Transformation/G functions" (Id 20220, Appl. 1)
--- 04.10.2006 09:24:57 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Feedrate" (Id 20280, Appl. 1)
--- 04.10.2006 09:24:56 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Tool" (Id 20300, Appl. 1)
--- 04.10.2006 09:24:56 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Actual block" (Id 20190, Appl. 1)
--- 04.10.2006 09:24:56 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Dummy-Window" (Id 21010, Appl. 1)
--- 04.10.2006 09:24:56 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Dummy-Window" (Id 1000, Appl. 1)
--- 04.10.2006 09:24:56 HMI-Emb 0 KEY_PRESSED
Key pressed: KEY_F6 (RELEASED) (e0/40)
--- 04.10.2006 09:24:56 HMI-Emb 0 KEY_PRESSED
Key pressed: KEY_F6 (0/40)
--- 04.10.2006 09:24:53 <default> 0 ALARM_QUIT
Alarm quit: 100014
--- 04.10.2006 09:24:53 HMI-Emb 0 OPEN_WINDOW
Window opened: "Dummy-Window" (Id 1000, Appl. 1)
--- 04.10.2006 09:24:53 HMI-Emb 0 CLOSE_WINDOW
Window closed: "Notebooks screen" (Id 5421, Appl. 1)
--- 04.10.2006 09:24:53 HMI-Emb 0 KEY_PRESSED
Key pressed: KEY_SELECT_MODE (0/44)
--- 04.10.2006 09:24:51 <default> 0 ALARM
NCK Alarm: 8080, cleartype Alarm-Cancel
7 option(s) is/are activated without setting the license key
--- 04.10.2006 09:24:50 HMI-Emb 0 KEY_PRESSED
Key pressed: KEY_F5 (RELEASED) (e0/3f)
```

### 8.3.3 Saving and outputting the log file

The log file display is protected using a password (access stage: Manufacturer).

£The log files (both the binary and readable variants) can be saved to an installed drive, e.g., Front (USB FlashDrive). The setting is made on the HMI Embedded user interface.

#### Reading-out the "action.log"

1. Press the "Service Displays" softkey in the "Diagnosis" operating area".
2. Press the "Action log” softkey.
3. Press the "Current Data" softkey in the vertical softkey bar. The "Action Log" window opens. The data are in ASCII format.
4. Press the "Save as..." softkey.
   If you want to save the log in binary format, press the "Save as binary..." softkey.
5. £Select an installed logical drive, e.g., "USB Front"/"Local drive".

#### Reading-out the "crash.log" log file

1. Press the "Service Displays" softkey in the "Diagnosis" operating area".
2. Press the "Action log” softkey.
3. Press the "Crash Data" softkey in the vertical softkey bar. The "Action Log Crash Data" window opens. The data are in ASCII format.
4. Press the "Save as..." softkey.
   If you want to save the log in binary format, press the "Save as binary..." softkey.
5. £Select an installed logical drive, e.g., "USB Front"/"Local drive".

#### Log file from the CompactFlash card

If the NCU crashes and the system does not reboot you can download the log files directly from the CompactFlash card using the "WinSCP" program.

1. The log files are on the CompactFlash card under the following directory:
   /user/sinumerik/hmi/

## 8.4 "Blue screen" error status screen

### Introduction

If the system crashes, an error status screen ("Blue Screen") showing current system information is displayed.

The error status screen displays a brief description of the error, which has occurred, plus all relevant processor registers.

You can transfer the complete system data image to the CompactFlash card (optional) or a USB FlashDrive.

### Displaying system data

In the event of a system crash, the crash address, the register allocation and any additional information needed for crash analysis are displayed on the screen.

```
Sorry, but HMI-Embedded has caused an exception in task XXXXX
Function at XXXXXXXX with code XXXXX
The type of exception is : XXXXXXXXXXXXXXXXXXXXX
The exception has occurred at cs : XX eip : XXXXXXXX
EAX= XXXXXXXX EBX= XXXXXXXX ECX= XXXXXXXX EDX= XXXXXXXX
EDI= XXXXXXXX ESI= XXXXXXXX FLAGS= XXXXXXXX DS= XXXX ES= XXXX
SS= XXXX ESP= XXXXXXXX EBP= XXXXXXXX FS= XXXX GS= XXXX
Additional information:
XXXXXXXX loaded at : XXXXXXXX XXXXXXXX loaded at : XXXXXXXX
XXXXXXXX loaded at : XXXXXXXX XXXXXXXX loaded at : XXXXXXXX
XXXXXXXX loaded at : XXXXXXXX XXXXXXXX loaded at : XXXXXXXX
CODE = XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 EIP: XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
 XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
STACK= XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX
 XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX
 XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX
 XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX XXXXXXXX
Please send the above or saved information to the SINUMERIK
Hotline!
E-mail : ad.support@siemens.com
Fill in subject : "HMI-Embedded exception (SW XX.XX.XX; date)"
Press 'S' to save data or press 'R' to reboot immediately
```

Press the following keys to:

Press "S"(Save) to save the file on the CompactFlash card.

Press "R" to initiate an NCU restart

## Save file on CompactFlash card

```
Press ‘L’ to save exception data to the local compact flash card.
You can also press ‘R’ to reboot immediately but we recommend to
first save the data.
```

Press "L", the following additional status message is displayed:

```
Please wait - data transmission in progress.
 If you have problems you can
 press key 'S' to stop transmission
```

Once all the data have been transferred successfully, the following message appears:

```
Transmission finished, XXXXXX bytes saved
The exception data was saved in folder /user/sinumerik/hmi
Please send the saved information to the SINUMERIK Hotline!
email : ad.support@siemens.com
Fill in subject : "HMI-Embedded exception (SW XX.XX.XX; date)"
```

## Saving a file on a USB FlashDrive

If you wish to save the error log file on a USB FlashDrive, then copy the "HMI-EM-LX-EX-*.TRC" file from the CompactFlash card from the folder /user/sinumerik/hmi.

## Restarting

If you initiate a restart, the system attempts an automatic restart.

If the NCU does not respond after one minute, perform restart manually by switching the system off and then on again.

```
Try rebooting - in the event of no reaction after 1 minute, please
switch off/on
```

## 8.5 Remote diagnosis "RCS Host Embedded / RCS Viewer Embedded"

It is possible for a service engineer to monitor and influence a control from a remote PC (remote PC = viewer), i.e. the same display of HMI-Embedded is shown on the remote PC as on the screen of the operator panel front.

Remote diagnosis has the following services:

- Direct access to the HMI-Embedded via network
- Data exchange (file transfer)

You can order the remote diagnosis software "RCS Viewer Embedded V1.1" for PC (Windows), available on CD "RCS Viewer Embedded", with
Order No.: 6FC6000-6DC81-1BA0.

### Prerequisite

In order to use the remote diagnosis software "RCS Host" on the NCU you require the license release (authorization to use the software) available under
Order No. 6FC5800-0AP30-0YB0.

See Licensing (Page 13)

The following hardware and software is required:

- Establish a tunnel connection between NCU 7x0 (Ethernet interface X130) (machine) and the remote PC (viewer), e.g. via hardware router with so-called VPN (Virtual Private Network) functionality, e.g. Cisco 803
- Windows PC (XP/ NT4/ 95/ 98) with network connection
- "Viewer" application (mmcR.exe) from the remote diagnosis CD (remote PC)
- Ethernet cable
- Logic drive for data exchange (file transfer)

### Connecting

The example shows the connection via the ISDN network (64 kbit/s). This telephone network provides the capability of connecting to remote LANs via a tunnel using the point-to-point protocol (PPP). The NCU 7x0 is connected to the ISDN network via the Ethernet interface with a hardware router (with VPN functionality).

Structure example of an ISDN connection:

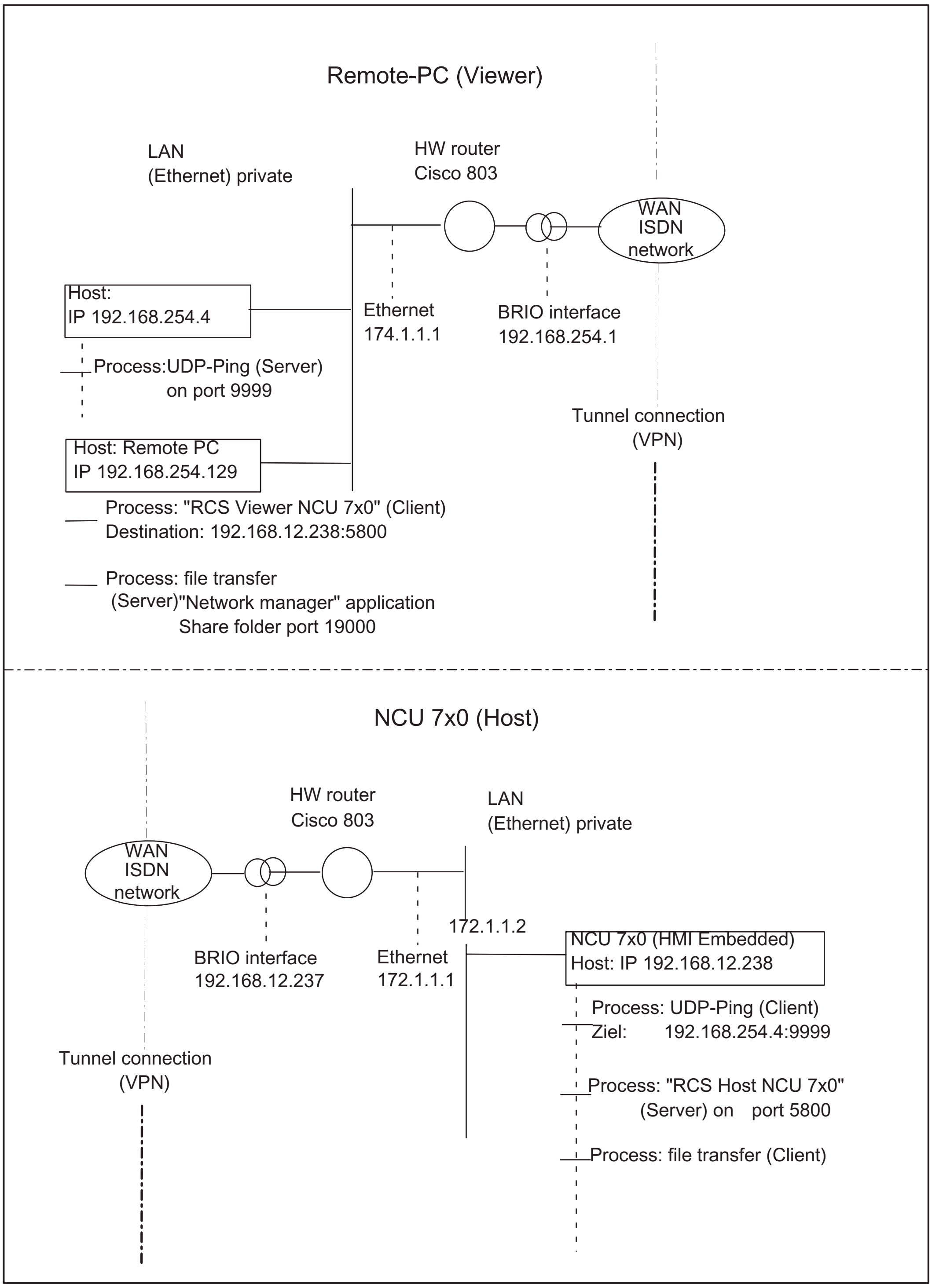



Figure 8-4 Example: Connection structure of the remote diagnosis

### How remote diagnosis works

A machine manufacturer who provides this service has selected an appropriate topology for a tunnel connection and will provide support when setting up the remote diagnosis.

The relevant “Ping server” must be activated on the remote PC for this purpose. This server has the task of accepting the incoming "Ping" data (e.g. serial number) of the requesting machine and registering the IP address of the requesting machine. If this server application is not provided by the service provider, it is possible to configure the RCS viewer application so that a simple “Ping” server process can be started on the RCS Viewer.

The IP address, the port, and the protocol used for this Ping server are specified by the service provider and must be set on the machine.

The remote diagnosis is initiated by the machine. The NCU 7x0 is connected to the appropriately configured router via an Ethernet cable and starts the remote diagnosis via the user interface. The NCU 7x0 then sends cyclical preset "Ping" data to the remote PC and waits for its response. The transmission of the "Ping" causes the router to establish the connection (e.g. switched ISDN connection).

When the response is received from the remote PC, the NCU 7x0 is ready for the data communication for the remote diagnosis. For this, a server process is started on the NCU 7x0, which is responsible for the exchange of graphic information to the remote PC and the keyboard inputs from the remote PC.

The exchange is performed via the TCP/IP protocol and a "permanently" agreed port. A so-called "Viewer" application (mmcR.exe) is installed on a remote PC as communication partner. From now on, all further activities will be initiated by the remote PC.

With the arrival of the "Ping" on the PC, the IP address of the requesting NCU 7x0 is known and the "Viewer" application (mmcR.exe) with this destination IP address can be started on a PC. This application runs as a client. The machine can be monitored and controlled via this remote PC.

The transfer of files between the remote PC and the machine is handled with local drives.

### See also

Setting-up network connections (Page 14)

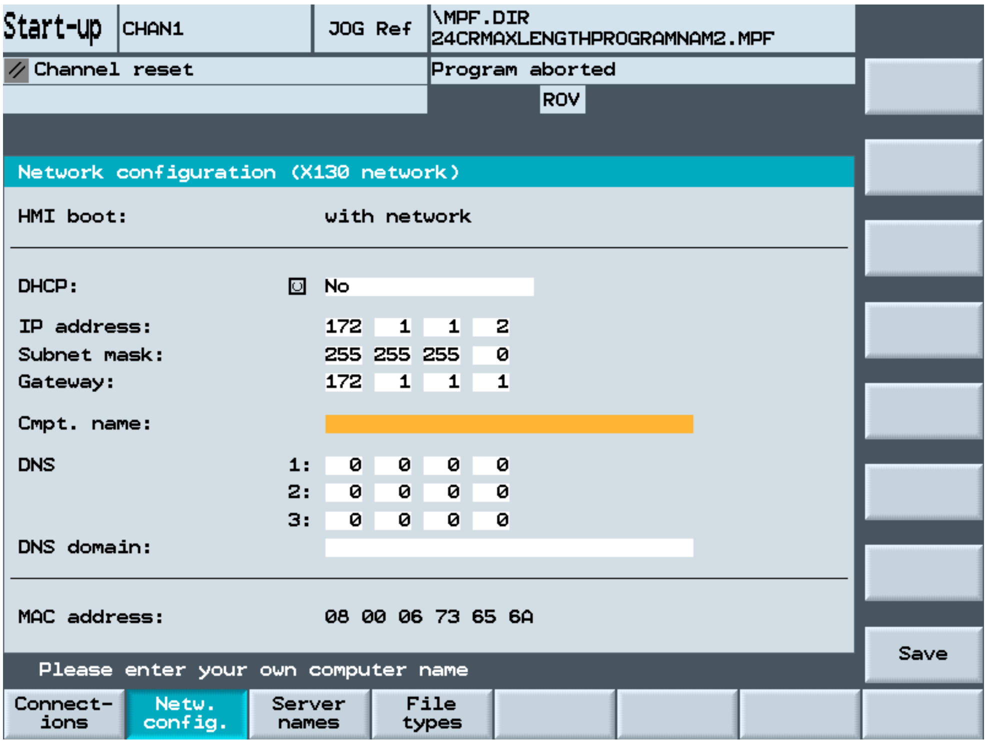
### 8.5.1 Configuring "RCS Host Embedded"

#### Adapting NCU 7x0 to router (VPN network)

NCU 7x0 (Ethernet interface X130) and router are connected via an Ethernet cable.

Adapt the network configuration of the NCU 7x0 to the configuration of the Ethernet interface of the router.

1. If the NCU 7x0 is already located on the network, save the current network configuration.
2. Start the machine.
3. Press the "Startup" -> "HMI" -> "Network Config." softkeys.
4. Entries:




5. Configuration of the Ethernet interface "Router NCU 7x0"
6. The following entries are predefined when the router is configured and are only used as examples in the above illustration.
   – IP address
   – Subnet mask
   – Gateway
7. Press the "Save" softkey. The NCU 7x0 restarts.

## Adapting the machine to the remote PC

1. In the "Start-up" area press the "ETC" key (">").
2. Press the "Remote Diagnosis” softkey. The following window appears:

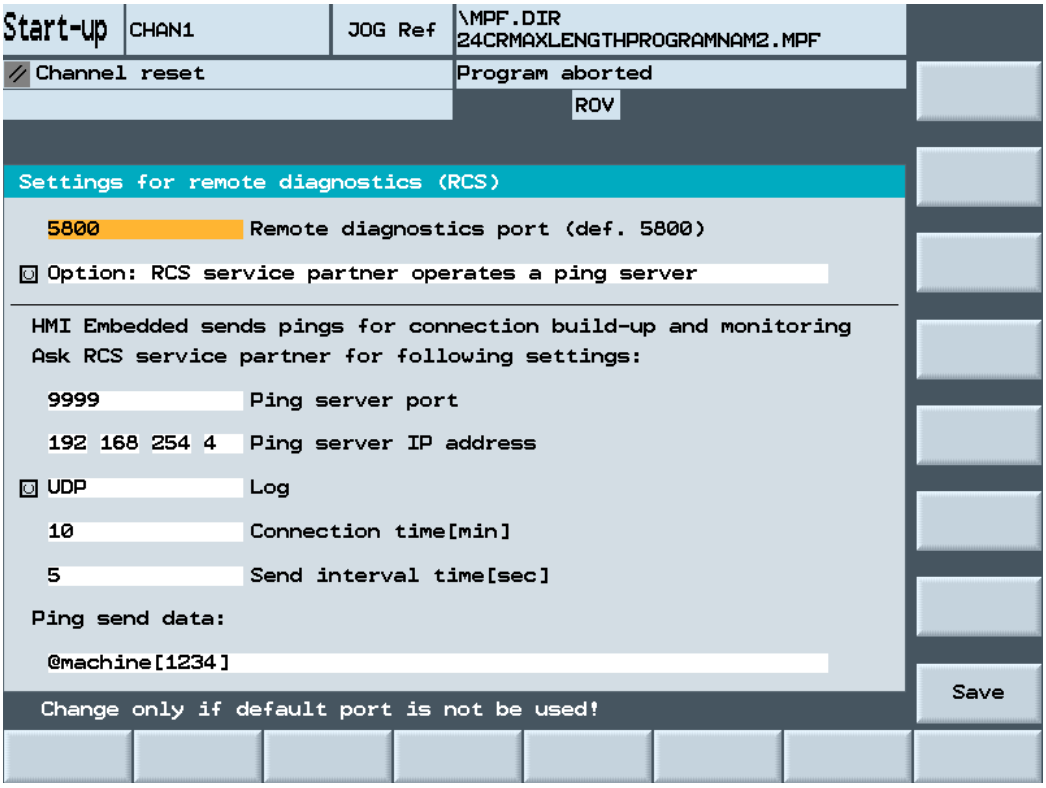



Figure 8-5 Setting up logon server

- The port number, default 5800, is entered in the "Port for remote diagnosis (default 5800)" input field for the server process (remote diagnosis). The same port number must be entered in the "Viewer" application on the remote PC.

## Option: "RCS service partner operates a Ping server" selected

By selecting the option "RCS service partner operates a Ping server", the remote PC is triggered to operate a Ping server. The following settings must be obtained from the service partner operating a ping server:

- Ping server port
- Png server IP address of the “Ping” server is running
- Protocol selection, UDP or TCP/IP
- Connection duration (min): Entry for the maximum time for the duration of the remote diagnosis. After this time expires, the connection is interrupted.
- Transmission interval (sec): The Ping data is transmitted cyclically in this interval from the NCU 7x0 to the remote PC. This is used for the monitoring of the connection.
- Ping transmission data: Entry of an agreed user data string

### Option: "RCS service partner operates a Ping server" deactivated

This option should only be deselected if the initiative for remote diagnosis is to come from the remote PC (e.g., if the remote PC and the NCU 7x0 are on the same intranet and an individual NCU 7x0 on the LAN is accessed from a PC).

When this option is deactivated, no other entries are possible.

1. Press the "Save" softkey to complete the configuration of the server.

### Connections for data exchange (file transfer)

In order to perform a file transfer, connection entries (\\<remote_ip>\<share folder name>) are made automatically by the system for the remote diagnosis. These connection entries are used by the "logical drives".

If none of the eight possible "logical drive" connections are free, switch to the "Start-up" area, press the softkey "Connections" and delete a connection entry.

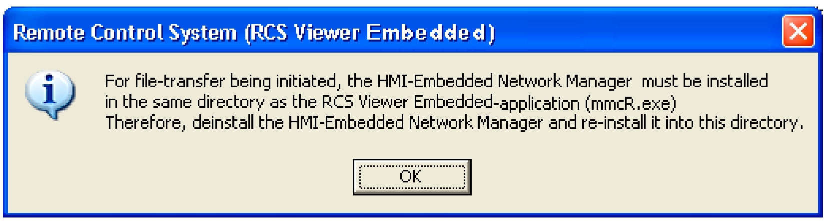
## 8.5.2 Configuring "RCS Viewer Embedded"

### Requirement

- Remote diagnosis CD with the "RCS Viewer Embedded" software
- Logic drive for data exchange (file transfer)

### Installation

1. Start installation of the remote diagnosis with "Setup.exe".
2. The following message appears:




3. Acknowledge the message.

**Note**

The Network Manager is not required with SINUMERIK 840D sl.

## Configuration

1. Start the "Viewer" application "mmcR.exe" on the remote PC.
2. If the default for the remote diagnosis port is changed, for example, due to firewall settings when the port is activated for incoming packages only, select the “Configuration” switch in the "RCS Viewer Embedded (Remote Control System)" window (see Figure 4-6). The following window appears:

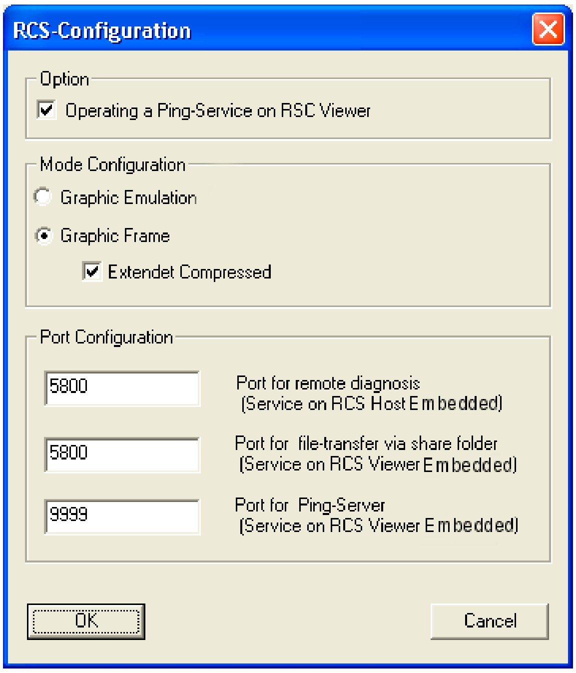



Figure 8-6 Port setting

- In the “Option” area, you can specify that a Ping server process is set up on the RCS Viewer, which will wait for the incoming “Pings”.
- Enter the port numbers in the “Port Configuration” area.

## Additional settings

### Setting graphic transmission mode

- Set the type of transmission for the screenshots in the “Mode Configuration” area. “Graphic Frame” and “Extended Compressed” are the default settings and feature the highest transmission rate for this remote diagnosis version.

**Note**

The field “Graphic Emulation” is not supported by SINUMERIK 840D sl.

1. Click "OK" to save the settings. The next window contains the box:
   "The port number has been stored. Please restart your application.”
2. Click "OK" to restart the system.

**Note**

Every time you change the configuration on the remote PC you must also restart the NCU.

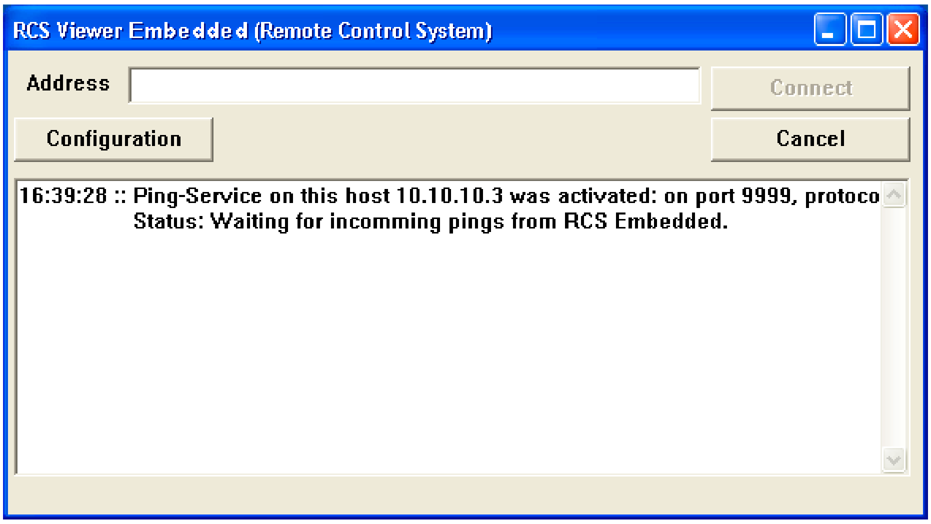
## 8.5.3 Starting remote diagnostics

### NCU 7x0 (Host)

1. Activate the "RCS Host Embedded" in the "Diagnosis" operating area via the "Remote diagnosis" and "Start" softkeys. Status messages provide information on the current state.
2. When the connection to the remote PC is established, the NCU 7x0 waits for the "Viewer“ application (mmcR.exe) to be started on the remote PC and the communication for the data exchange to be activated.
   The message: "Ping server responds" appears.
3. This communication is performed via remote diagnosis which, per default, is applied to port number 5800.
4. When the message "Communication running" appears, operation of the machine from the remote PC is possible.

### Remote PC (viewer)

1. Either start the RCS Viewer directly from the Start menu or by double-clicking on file "mmcR.exe".
   – If a "Ping" server is operated in the system, the IP address of the NCU 7x0 (Ethernet interface X130) is made known to the remote PC on arrival of the next "Ping". (e.g. 147.54.235.46).
   – If no Ping server is available, enter the address manually.




2. RCS Viewer waiting for incoming “Pings”
3. Click the button "Connect". The connection with the machine is activated via the port for remote diagnosis. The corresponding status messages are output in the field below. When these are completed, the dialog is minimized and the interface of HMI-Embedded sl appears.
4. Press the “Configuration” softkey to display the “RCS Configuration” window. See Fig: Port setting

## File transfer

1. The file transfer is performed in the "Program" operating area.
2. Press the softkey specified under "Logical drives". The content of the selected folder is displayed and transfer can be executed via the vertical softkeys.

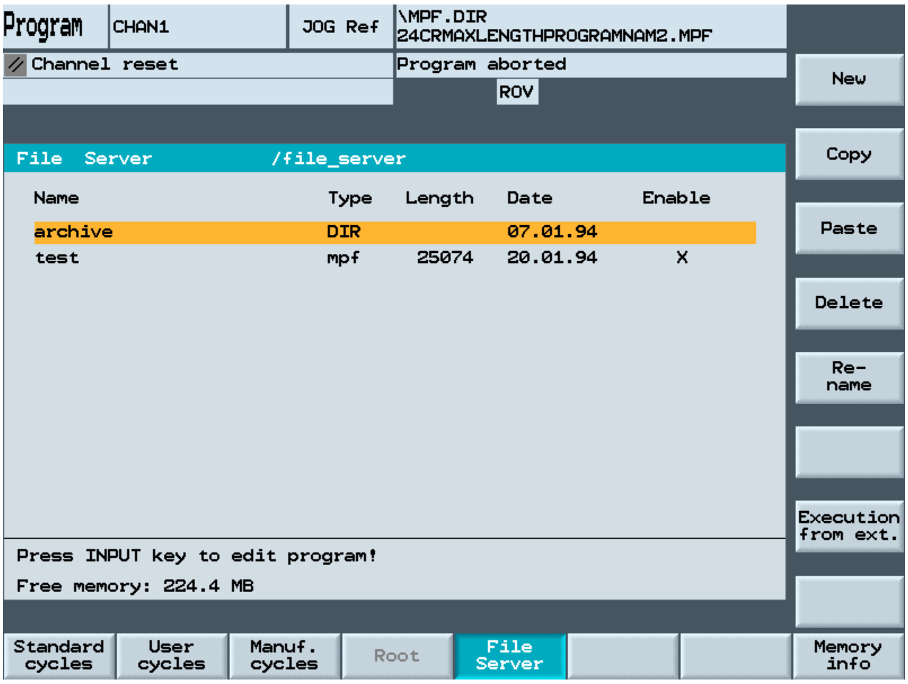



Figure 8-7 Share folder for file transfer

# Index

## D



## E



## F



## H



## I

## K



## L



## M

## T



## U

## V

## W

SIEMENS

SINUMERIK 840D sl

# Expand User Interface

Commissioning Manual



Valid for
SINUMERIK 840D sl/840DE sl control system

| Software | Version |
|---|---|
| NCU System Software | 1.4 |
| with HMI-Embedded sl | 7.2 |


11/2006
6FC5397-8CP10-0BA0

## Safety Guidelines

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

**Danger**

indicates that death or severe personal injury **will** result if proper precautions are not taken.

**Warning**

indicates that death or severe personal injury **may** result if proper precautions are not taken.

**Caution**

with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken.

**Caution**

without a safety alert symbol, indicates that property damage can result if proper precautions are not taken.

**Notice**

indicates that an unintended result or situation can occur if the corresponding information is not taken into account.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

## Qualified Personnel

The device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by **qualified personnel**. Within the context of the safety notes in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

## Prescribed Usage

Note the following:

**Warning**

This device may only be used for the applications described in the catalog or the technical description and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens. Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance.

## Trademarks

All names identified by ® are registered trademarks of the Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

## Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency.
However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Automation and Drives
Postfach 48 48
90437 NÜRNBERG
GERMANY

Order No.: 6FC5397-8CP10-0BA0
℗ 02/2007

# Table of contents

# 1 Scope of performance

## Overview

"Expand user interface" is implemented with an interpreter and configuration files that contain the descriptions of the user interfaces. "Expand user interface" is configured using ASCII files: These configuration files contain the description of the user interface. The syntax that must be applied in creating these files is described in the following chapters.

The "Expand user interface" tools can be used to create user interfaces that display functional expansions designed by the machine manufacturer or end user, or simply to implement your own dialog layout. Preconfigured user interfaces supplied by Siemens or the machine manufacturer can be modified or replaced.

The interpreter is available for HMI Embedded sl, ShopMill and ShopTurn on NCU as well as HMI Advanced.

Parts programs, for example, can be edited on user interfaces created by users. Dialogs can be created directly on the control system.

## Prerequisites

An additional graphics program is needed to produce graphics/display images.
For HMI Embedded sl you need the application disk and the Paint Shop Pro tool (http://www.jasc.com). The tool box supplied contains configuration examples for new dialogs. You can also use these examples as a template for creating your own dialogs.

## Use

You can implement the following functions:

1. Display dialogs containing the following elements:
   - Softkeys
   - Variables, tables
   - Texts and Help texts
   - Graphics and Help displays
2. Open dialogs by:
   - Pressing the (start) softkeys
   - Selection on the PLC
3. Restructure dialogs dynamically:
   - Edit and delete softkeys
   - Define and design variable fields
   - Insert, exchange and delete display texts (language-dependent or independent)
   - Insert, exchange and delete graphics

4. Initiate operations in response to the following actions:
   – Displaying dialogs
   – Input values (variables)
   – Select a softkey
   – Exiting dialogs
5. Data exchange between dialogs
6. Variables
   – Read (NC, PLC and user variables)
   – Write (NC, PLC and user variables)
   – Combine with mathematical, comparison or logic operators
7. Execute functions:
   – Subroutines
   – File functions
   – PI services
   – External functions (HMI Advanced)
8. Apply protection levels according to user classes

## Supplementary Conditions

The following conditions must be met:

- It is only possible to switch between dialogs within one HMI operating area.
- In the case of HMI Advanced, user, setting and machine data are initialized on request.
- User variables may not have the same names as system or PLC variables.
- The dialogs activated by the PLC form a separate operating area for HMI Advanced (similar to measuring cycle displays).

---

**Note**

The programming support functions described in the chapter of the same name and the user interfaces for Siemens cycles have been created with the system tools for Expand user interface. As a result, they can be modified as required by the machine manufacturer or end user within the scope described in this chapter.

---

## See also:

You can find details of the configuration files in the chapter "Configuring environment".

# 2 Programming

## 2.1 Getting started

### 2.1.1 Fundamentals of Configuration

#### Configuration files

The defining data for new user interfaces are stored in configuration files. These files are automatically interpreted and the result displayed on the screen. Configuration files are not stored in the software supplied and must be set up by the user.

An ASCII editor (e.g., Notepad or the HMI editor) is used to create configuration files.

#### Menu tree principle

Several interlinked dialogs create a menu tree. A link exists if you can switch from one dialog to another. You can use the newly defined horizontal/vertical softkeys in this dialog to call the preceding or any other dialog.

A menu tree can be created behind each start softkey:

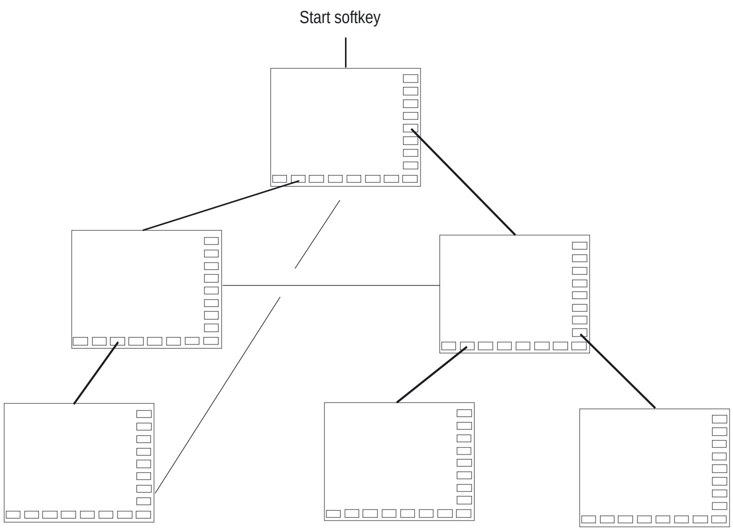



Figure 2-1 Menu tree

### Start softkeys

One or more softkeys (start softkeys), which are used to initiate your own operating sequences, are defined in one of the specified configuration files.

The loading of a dedicated dialog is associated with a softkey definition or another softkey menu. These are then used to perform the subsequent actions.

Pressing the start softkey loads the assigned dialog. This will also activate the softkeys associated with the dialog. Variables will be output to the standard positions unless specific positions have been configured.

### Reverting to the standard application

You can exit the newly created user interfaces and return to the standard application.

You can use the <RECALL> key to close new user interfaces if you have not configured this key for any other task.

### Calling your own dialogs from PLC

Dialogs can be selected via the PLC as well as via softkeys. An interface (in DB19) is available for signal exchange between the PLC and HMI.

### See also

Structure of the interface (Page 153)

Configuring start softkeys (Page 179)

## 2.1.2 Configuration and start files

### Overview

Every application uses permanent (HMI Embedded sl, ShopMill, and ShopTurn on NCU) or preset (HMI Advanced) start softkeys, which can be be used to access newly generated dialogs. Additional start softkeys can be configured with HMI Advanced.

**Other files:**

In the event of "Load a screen form" (LM) or a "Load softkey menu" (LS) call in a configuration file, a new file name containing the object called can be specified. This makes it possible to structure the configuration, e.g., all functions in one operation mode in a separate configuration file.

### Creating configuration file as ASCII file

Dialogs can contain, for example, the following elements:

- Input/output fields (variables) with
  – Short text
  – Graphic text
  – Text for units
- Images
- Tables
- Softkey menus

### Search sequence for configuration files

- HMI Embedded sl on NCU

  For HMI Embedded sl as well as ShopMill and ShopTurn on NCU, the system searches for the configuration files by accessing the relevant directories of the CF card.

  A setting can be made in file COMMON.COM in the standard cycle directory/user cycle directory of HMI Embedded sl to specify whether a configuration file search should be carried out each time access is attempted (only relevant when the dialogs are being set up directly on the control) or if the file already located and buffered should be reused (corresponds to the standard operating scenario).
- HMI Advanced

  With HMI Advanced, the system starts by searching for the configuration files in the user cycle directory and then looks in the manufacturer cycle directory followed by the standard cycle directory.

### See also

Search function principle (Page 184)

## 2.1.3 Structure of configuration file

### Overview

A configuration file consists of the following elements:

- Description of the start softkeys
- Definition of dialogs
- Definition of variables
- Description of the blocks
- Definition of a softkey menu

Example

```
//S (START)                ; Definition of the start softkey (optional)
....
//END
//M (.....)                ; Definition of the dialog
DEF .....                  ; Definition of variables
LOAD                       ; Description of the blocks
...
END_LOAD
UNLOAD
...
END_UNLOAD
ACTIVATE
...
END_ACTIVATE
...
//END
//S (...)                  ; Definition of a softkey menu
//END
```

### 2.1.4 Troubleshooting (log book)

#### Overview

The log book is a file (Error.com) to which error messages generated by syntax interpretation are written. The operator himself must set up the file in the comment directory (HMI Advanced).

Example

```
DEF VAR1 = (R)
DEF VAR2 = (R)
LOAD
VAR1 = VAR2 + 1            ; Error message in log book, as VAR2 has no value.
```

## Syntax

The system does not start to interpret syntax until the start softkey has been defined and a dialog with start and end identifiers as well as a definition line has been configured.

```
//S(Start)
HS6=("1st screen form")
PRESS(HS6)
  LM("Screen form1")
END_PRESS
//END

//M(Screen form1)
  DEF Var1=(R)
//END
```

## Content of ERROR.COM

If "Expand user interface" detects errors when interpreting the configuration files, these errors will be written to the ERROR.COM ASCII file.

The file indicates:

- The action during which an error occurred
- The line and column number of the first faulty character
- The entire faulty line of the configuration file

If the dialog was created using the PC test environment, then the error file will be stored in the folder referenced by environment variable RAMDISK (HMI Embedded sl).

The ERROR.COM file is created only when errors actually occur during interpretation of the configuration files.

Storage path of the ERROR.COM file:

- With HMI Embedded sl: In the /tmp/hmiemb folder on the CF card
- With HMI Advanced: In the \DH\COM.DIR\ folder

This file is deleted each time HMI Embedded sl / HMI Advanced is restarted.

## Displaying the ERROR.COM file

HMI Advanced:

- Call the editor in the "Services" or "Startup" operating area.

HMI Embedded sl:

- "Setup" operating area → "HMI" → "Editor" → "Temp drive"
  (4th softkey on vertical softkey bar; although this softkey is not displayed with the manufacturer password, it will respond). (It is also displayed with the system password.)
- Select ERROR.COM.

- Press the <INPUT> key.
- With the “File functions” softkey, the file can also be copied to a PC by means of a Windows network drive.

## 2.2 Structure and elements of a dialog

### 2.2.1 Defining a dialog

#### Definition

A dialog is part of a user interface consisting of a display line, dialog elements and/or graphics, an output line for messages and 8 horizontal and 8 vertical softkeys.

Dialog elements are:

- Variables
  - Limits
  - Default setting of variables
- Help display
- Texts
- Attributes
- System or user variable
- Position of short text
- Position of input/output field
- Colors
- Help (HMI Advanced only)

Dialog properties:

- Header
- Graphic
- Dimension
- System or user variable
- Graphic position
- Attributes

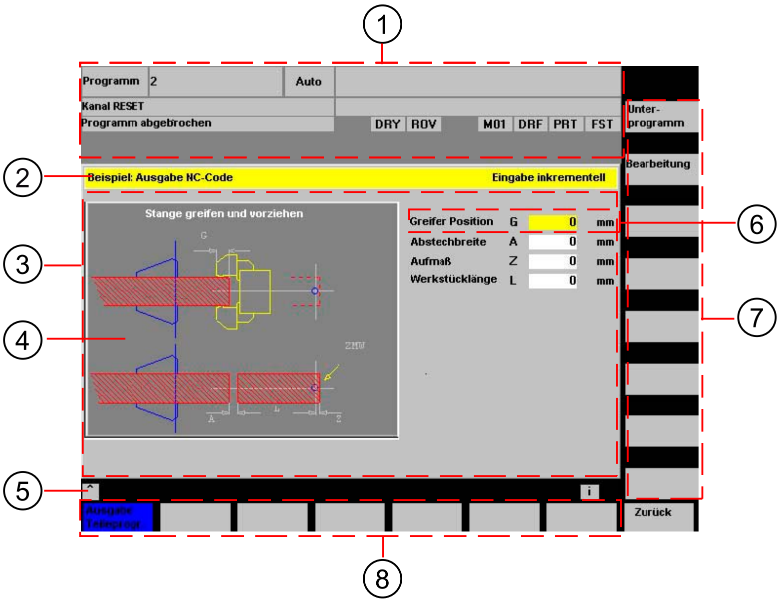



Figure 2-2 Structure of the dialog

1 Machine status display ("header")
2 Header line of the dialog with header and long text
3 Dialog
4 Graphic
5 Displaying messages
6 Dialog element
7 8 vertical softkeys
8 8 horizontal softkeys

## Overview

The definition of a dialog (definition block) is basically structured as follows:

| Definition block | Comment | Chapter reference |
|---|---|---|
| `//M...` | ;Dialog start identifier | |
| `DEF Var1=...`<br>`...` | ;Variables | See chapter "Variables" |
| `HS1=(...)`<br>`...` | ;Softkeys | See chapter "Softkey menus" |
| `PRESS(HS1)`<br>`  LM...`<br>`END_PRESS` | ;Method start identifier<br>  ;Actions<br>;Method end identifier | See chapter "Methods" |
| `//END` | ;Dialog end identifier | |

Within the dialog definition block, various variables that appear as dialog elements in the dialog, as well as horizontal and vertical softkeys, are defined first. Different types of actions are then configured in methods.

## 2.2.2 Example Opening the Dialog

### Programming

A new dialog is called via start softkey "Example" from the "Parameters" operating area.

```
//S(Start)
HS7=("Example", ac7, se1)

PRESS(HS7)
  LM("Screen form1")
END_PRESS

//END
//M(Screen form1/"Cycles")
HS1=("")
HS2=("")
HS3=("")
HS4=("")
HS5=("")
HS6=("")
HS7=("")
HS8=("")
VS1=("")
VS2=("")
VS3=("")
VS4=("")
VS5=("")
VS6=("")
VS7=("")
VS8=("")
...                                          ; Methods
//END
```

## Result

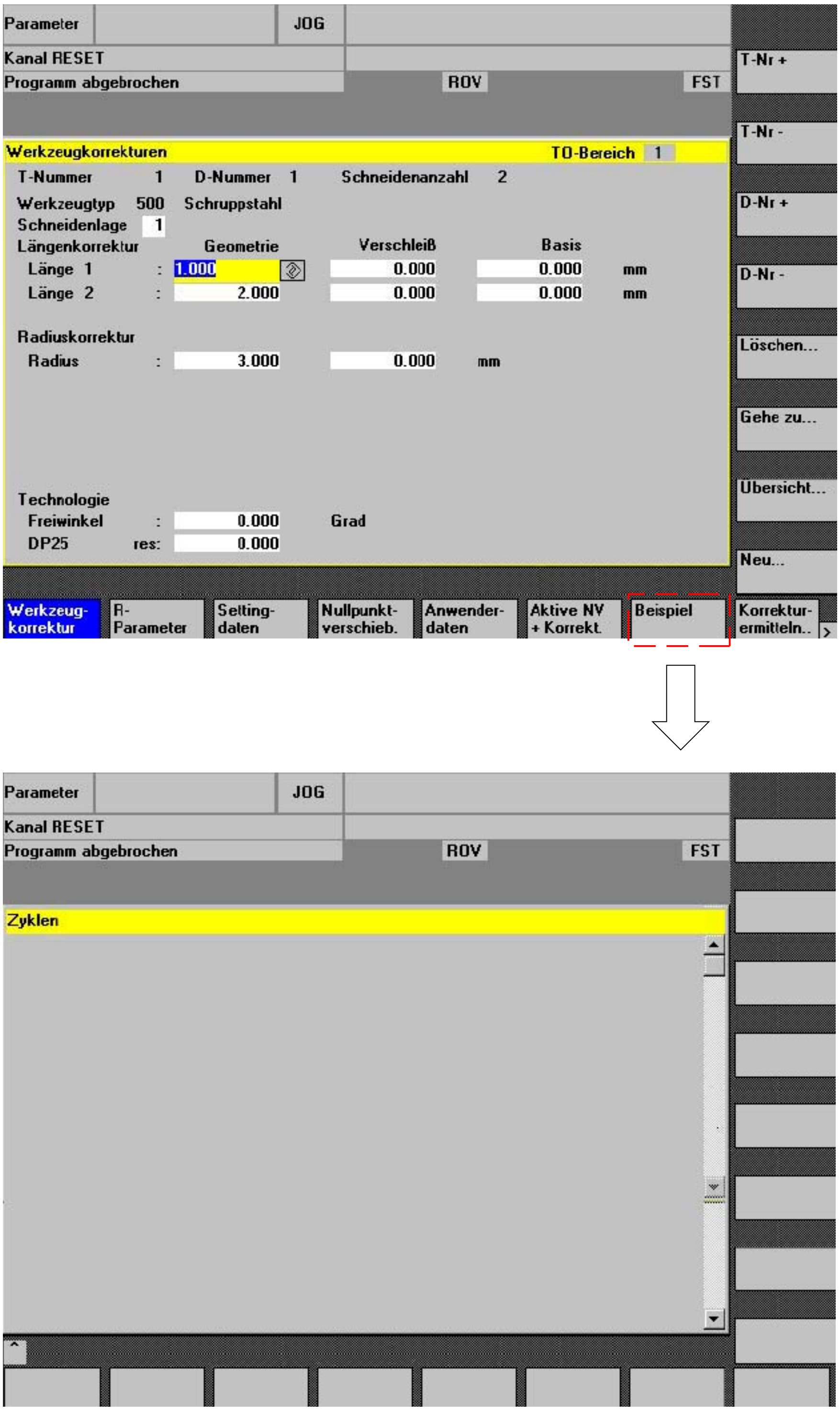



Figure 2-3 Example Call "Cycles" dialog with the start softkey "Example"

## 2.2.3 Dialog properties

### Description

The properties of the dialog are defined in the start identifier line of the dialog.

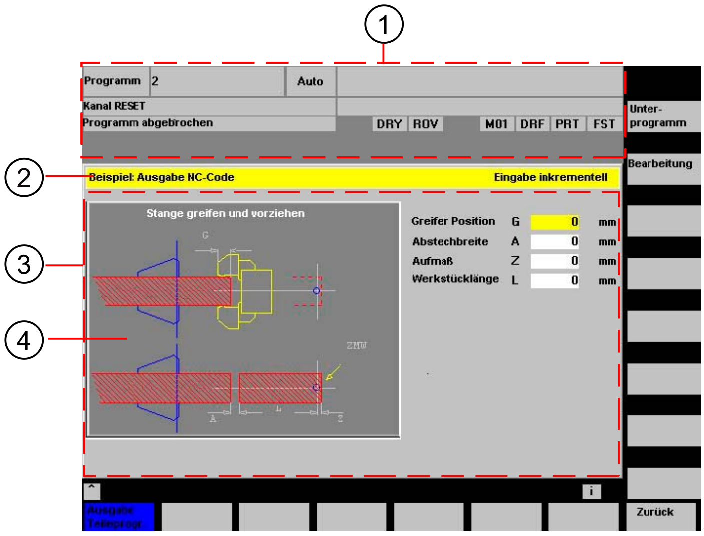



Figure 2-4 Dialog properties

| | |
|---|---|
| 1 | Machine status display ("header") |
| 2 | Header line of the dialog with header and long text |
| 3 | Dialog |
| 4 | Graphic |

### Programming

| | | |
|---|---|---|
| Syntax: | //M(Identifier/[Header]/[Graphic]/[Dimension]/[System or user variable]/[Graphic position]/[Attributes]) | |
| Description: | Defines a dialog | |
| Parameters: | Identifier | Name of the dialog |
| | Header | Dialog header as text or call for text (e.g., $85011) from a language-specific text file |
| | Graphic | Graphics file with path in double quotation marks |
| | Dimension | Position and size of the dialog in pixels (distance from left-hand side, distance from right-hand side, width, height), in relation to the upper left-hand corner of the screen. The entries are separated by a comma. |

| | | |
|---|---|---|
| System or user variable | | System or user variable to which the current cursor position is assigned. The cursor position can be assigned to the NC or PLC via the system or user variable. The first variable has the index 1. The order corresponds to the configuration order of the variables. |
| Graphic position | | Position of the graphic in pixels (distance from left-hand side, distance from right-hand side), in relation to the upper left-hand corner of the dialog.<br>The minimum distance from the top is 18 pixels.<br>The entries are separated by a comma. |
| Attributes | | The specifications of the attributes are separated by a comma. |
| | | Possible attributes are: |
| CMx | | Column mode: Column alignment |
| | CM0 | Default: The column distribution is carried out separately for each line. |
| | CM1 | The column distribution of the line with the most columns applies to all lines. |
| CB | | CHANGE block: Response when dialog is opened: cb attributes specified for a variable in a variables definition take priority over the default setting in the dialog definition. |
| | CB0 | Default: All CHANGE blocks associated with the dialog are processed when it is opened. |
| | CB1 | CHANGE blocks are then only processed if the relevant value changes. |
| System | | The "System" property can be read during runtime: |
| | 0: | HMI_Embedded |
| | 1: | HMI_Advanced |

### Accessing the dialog properties

Read and write access is provided to the following dialog properties within Methods (e.g., PRESS block)

- Hd = Header
- Hlp = Help display
- Var = System or user variable

## Example

```
//S(Start)
HS7=("Example", se1, ac7)

PRESS(HS7)
  LM("Screen form1")
END_PRESS

//END
//M(Screen form1/"Example 2 :
Display graphics"/"MCP.BMP")
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
```

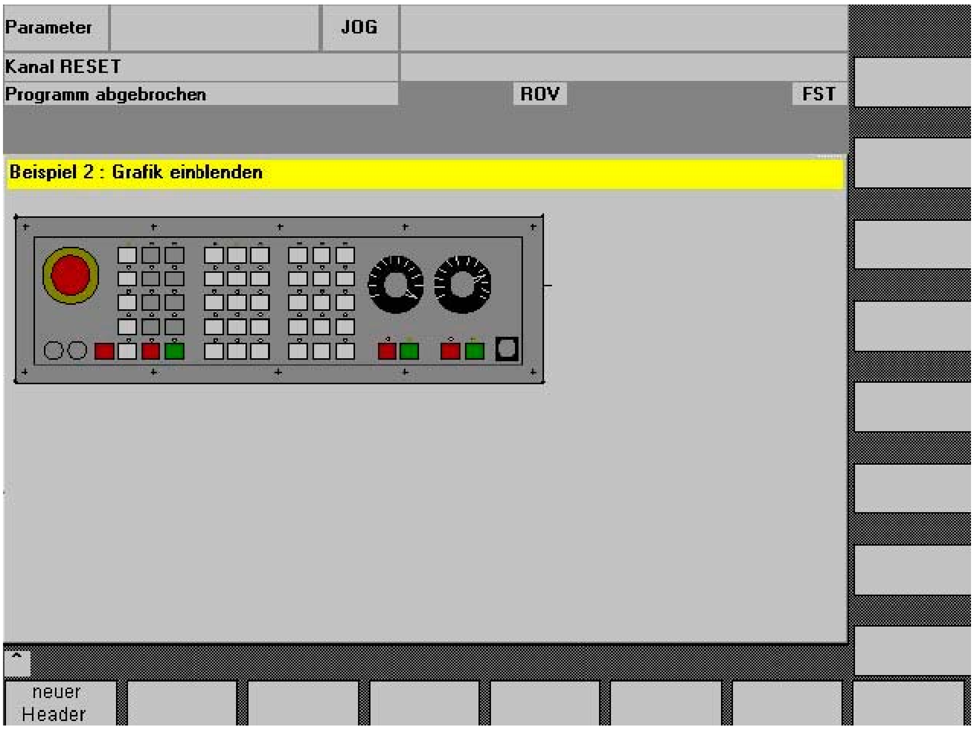



Figure 2-5 "Example 2:Display graphics"

### See also

Using display images/graphics (Page 25)

Language-dependent text (Page 181)

## 2.2.4 Dialog elements

### Dialog element

The term "dialog element" refers to the visible part of a variable, i.e., short text, graphics text, input/output field and unit text. Dialog elements fill lines in the main body of the dialog. One or more dialog elements can be defined for each line.

### Variable properties

All variables are valid only in the active dialog. Properties are assigned to a variable when it is defined. The values of dialog properties can be accessed within Methods (e.g., a PRESS block).

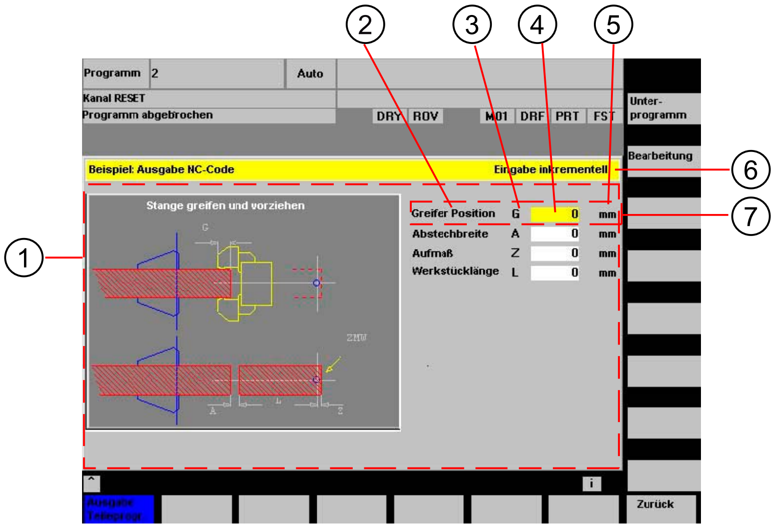



Figure 2-6 Elements of a dialog

| | |
|---|---|
| 1 | Main body of the dialog |
| 2 | Short text |
| 3 | Graphic text |
| 4 | Input/output field |
| 5 | Text for units |
| 6 | Header line of the dialog with header and long text |
| 7 | Dialog element |

## Programming - Overview

The single parameters to be separated by commas are enclosed in round parentheses:

| `DEF` *Identifier* = | **Identifier = Name of variable** | | |
|---|---|---|---|
| | Variable type | → | 1 |
| | /[Limits or toggle field or table identifier] | → | 2 |
| | /[Default] | → | 3 |
| | /[Texts (Long text, Short text\|Image, Graphic text, Units text)] | → | 4 |
| | /[Attributes] | → | 5 |
| | /[Help display] | → | 6 |
| | /[System or user variable] | → | 7 |
| | /[Position of short text] | → | 8 |
| | /[Position of input/output field(Left, Top, Width, Height)] | → | 9 |
| | /[Colors] | → | 10 |
| | /[Help] (HMI Advanced only) | → | 11 |

### See also

Dialogs with multiple columns (Page 23)
Variable properties (Page 27)

## 2.2.5 Dialogs with multiple columns

### Overview

Multiple variables can also be represented in a dialog on one line. In this case, the variables are all defined in the configuration file on a single definition line.

```
DEF VAR11 = (S///"Var11"), VAR12 = (I///"Var12")
```

To make individual variables in the configuration file more legible, the definition lines can be wrapped after every variables definition and following comma.

The key word "DEF" always indicates the beginning of a new line:

```
DEF Tnr1=(I//1/"","T ",""/wr1///,,10/20,,50),
  TOP1=(I///,"Type="/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)
...
```

When creating dialogs with multiple columns, the options of the hardware being used should be taken into consideration, e.g., HMI Embedded sl supports up to 10 columns and 60 DEF instructions.

## 2.2.6 Unifying the dialog appearance

### Boundary condition

If HMI Advanced is installed together with ShopMill or ShopTurn on one operator panel, then the two systems have different font types.

HMI Advanced has "proportional fonts", while JobShop products and HMI Embedded sl have a "fixed font".

If "Expand user interface" is used on HMI Embedded sl and HMI Advanced, dialogs that are defined identically will appear differently by default.

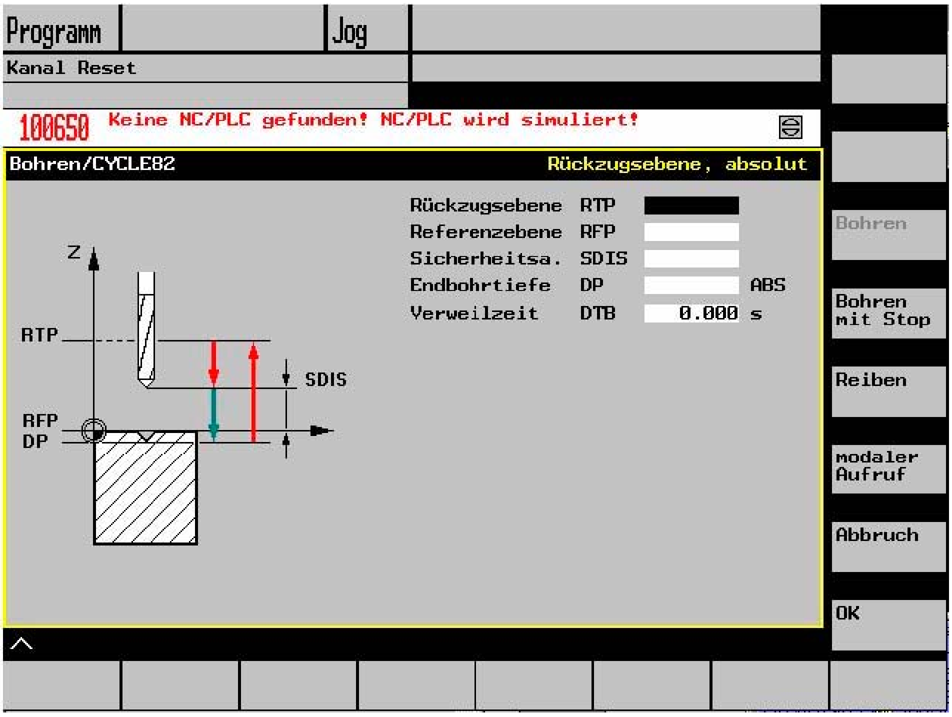



Figure 2-7 Appearance in HMI Embedded sl

Below you can see how, despite being configured identically, the same screen appears differently under HMI Advanced.

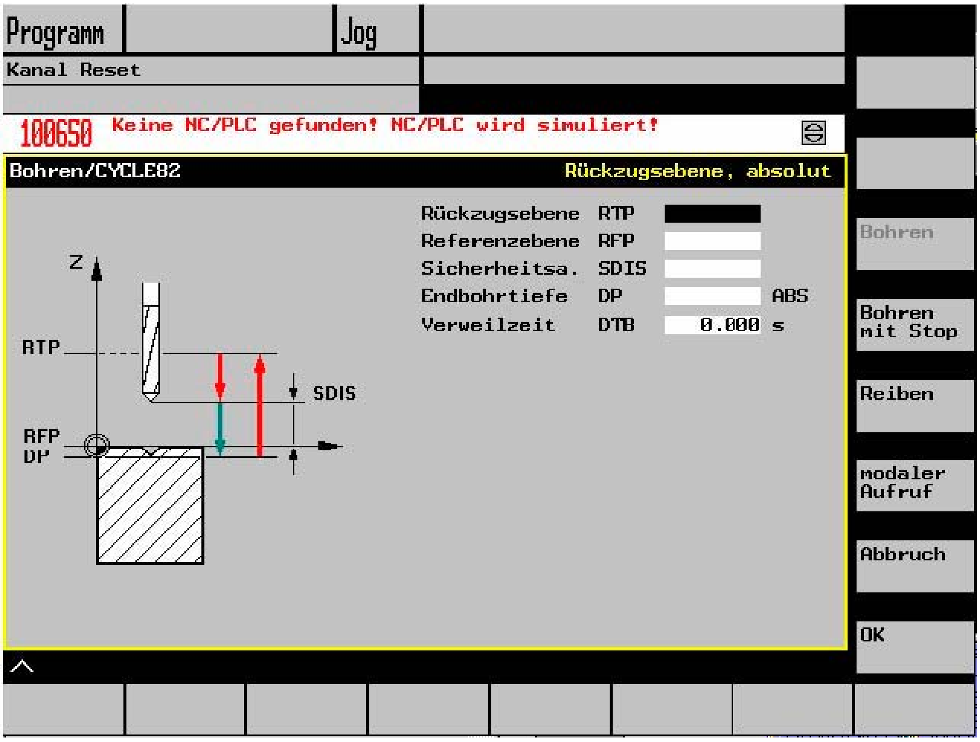



Figure 2-8 Appearance in HMI Advanced

### Setting for the same screen appearance

In order for the appearance in HMI Advanced to tally with that in HMI Embedded sl, the font is changed to a fixed font in the configuration file WIZARD.INI:

```
[FONT]
FixedFont=1
```

The text will be displayed in the current language in fixed font format. The only exception is text for the softkey labels. Unlike the HMI Embedded sl input field, the HMI Advanced input field contains an input mode pictogram. This means it is smaller than the HMI Embedded sl field.

As you can see above, the combination of short text, graphic text, input field, and unit text is approximately the same length and contains the same information in both screens.

However, individual fields can be positioned differently in the two HMI versions depending on the length of the text to be displayed and the text field within which it will be contained.

To enable a common configuration file to be produced for both systems, the SYSTEM dialog property has been created. The runtime environment can be scanned in the SYSTEM dialog property. This enables a common configuration to be used for both target systems.

## 2.2.7 Using display images/graphics

### Use of graphics

There are two display categories:

- Display images/graphics in the graphic area
- Help displays illustrating, for example, individual variables, which are superimposed in the graphic area.
- More Help displays can be configured instead of short text or an input/output field, which you position where you like.

| Maximum size | System |
|---|---|
| 560 * 326 pixels | HMI Advanced/HMI Embedded sl |
| 688 * 376 pixels | PCU 50 with OP 012 |

### Creating screen images for HMI Embedded sl

Create the screen images using, e. g., the MS Paint application.

HMI Embedded sl can handle converted graphics as well as BMP files. Graphics created using the conversion program BMP2BIN can continue to be used. For BMP files, the colors are converted online. For this purpose, the files containing the color table (syscol.col, sysbw.col, etc.) have been extended to include the [BMP] section. This section allocates a color of a BMP pixel to a color in HMI Embedded.

### Storage locations

HMI Embedded sl automatically finds the resolution of the connected monitor and searches for the desired display image first in the associated resolution directory. If it is not found there, HMI Embedded sl searches for the display image in the next smaller resolution directory until – if it doesn't find the display image earlier – it reaches directory ico640.

### Procedure

Procedure when integrating the bitmaps (customer displays):

1. Integration on HMI Embedded sl
2. Generate BMP files using Microsoft Paint, Version 4.0 or higher
3. Use the arj.exe supplied (Version 2.41) to archive the BMP files into archive CUS.ARJ or, alternatively, archive each BMP file in a separate archive with the file name of the BMP file and extension ".BM_"

   For example:

   – Several files per archive:

     arj a cus.arj my_file1.bmp my_file2.bmp my_file3.bmp my_file4.bmp

   – One file per archive:

     arj a my_file1.bm_  my_file1.bmp

### Limits

Archiving is optional. However, it must be noted that the CF card has space for at least 10 MB.

A bitmap can be assigned to any softkey if the softkey text is preceded by 2 backslashes. The text behind the backslashes is then interpreted as the name of the file that contains the bitmap.

For example, in ALUC.TXT  85000  0 0  "\\mybitmap.bmp"

---

**Note**

The display colors might not appear in the same format in HMI Embedded sl and in Paint.

---

### Display building, HMI Advanced

Graphic programs can be freely selected provided that they are capable of generating the prescribed formats. Graphics, display images and Help displays can have the following format:

- Bitmap (BMP)
- Windows Metafile (WMF)
- Icon (ICO)

### "Display in display"

You can insert further displays in the background (Help display), even as a function of the values of variables.Like for softkeys you can assign a graphics file to a display field.

Examples:

- Short text field is overlaid with graphics:

  **DEF VAR1**=(S///,"\\image1.bmp" ////160,40,50,50)   ;Type S irrelevant

- Toggle field overlaid with graphics, the graphic being selected with a PLC flag byte:

  **DEFVAR1**=(IDB/*1="\\image1.bmp",2="\\image2.bmp"//,$85000/wr1//"MB[0]" //160,40,50,50)

The height of the displayed graphic (of the I/O field) is specified as the fourth parameter in respect of the positions. You can also define bitmaps in fields (short text, I/O field).

### See also

Search function for images (Page 188)

List of colors (Page 193)

# 2.3 Variables

## 2.3.1 Variable properties

### Variable value

The most basic property of a variable is its value.

The value of variables can be assigned by means of:

- Default settings when defining variables
- Assignment to a system or user variable
- A method

### Programming

| | | |
|---|---|---|
| Syntax: | Identifier **val** = Variable value | |
| | Identifier = Variable value | |
| Description: | Variable value val (value) | |
| Parameters: | Identifier: | Name of the variable |
| | Variable value: | Value of variable |
| Example | `VAR3 = VAR4 + SIN(VAR5)`<br>`VAR3.VAL = VAR4 + SIN(VAR5)` | |

## Variable status

The "Variable status" property can be scanned during runtime to find out whether a variable has a valid value. This property can be read and written with the value FALSE = 0.

## Programming

| | | |
|---|---|---|
| Syntax: | Identifier.**vld** | |
| Description: | Variable status vld (validation) | |
| Parameters: | Identifier: | Name of the variable |
| | | The result of the scan can be: |
| | FALSE = | invalid value |
| | TRUE = | valid value |

Example

```
IF VAR1.VLD == FALSE
  VAR1 = 84
ENDIF
```

## Variable: Changing properties

When a change is made, a new value is assigned to the variables in the notation *Identifier.Property = Value* . The expression to the right of the equality sign is evaluated and assigned to the variable or variable property.

Examples:

| | |
|---|---|
| Identifier.**ac** = Access level | (ac: access level) |
| Identifier.**al** = Text alignment | (al: alignment) |
| Identifier.**bc** = Background color | (bc: background color) |
| Identifier.**fc** = Foreground color | (fc: foreground color) |
| Identifier.**fs** = Font size | (fs: font size) |
| Identifier.**gt** = Graphic text | (gt: graphic text) |
| Identifier.**hlp** = Help display | (hlp: help) |
| Identifier.**htx** = Help text | (htx: help text) |
| Identifier.**li** = Limit | (li: limit) |
| Identifier.**lt** = long text | (lt: long text) |
| Identifier.**max** = MAX limits | (max: maximum) |
| Identifier.**min** = MIN limits | (min: minimum) |
| Identifier.**st** = short text | (st: short text) |
| Identifier .**typ** = Variable type | (typ: type) |
| Identifier.**ut** = Unit text | (ut: unit text) |
| Identifier.**val** = Variable value | (val: value) |
| Identifier.**var** = System or user variable | (var: variable) |
| Identifier.**vld** = Variable status | (vld: validation) |
| Identifier .**wr** = Input mode | (wr: write) |

## 2.3.2 Programming variables

### Programming

The parameters of the variables are explained briefly in the following overview.
A comprehensive description can be found in subsequent chapters.

| Parameter | Description | |
|---|---|---|
| 1 Variable type | The variable type must be specified. | |
| | R[x]: | REAL (+ digit for the decimal place) |
| | I: | INTEGER |
| | S[x]: | STRING (+ digit for string length) |
| | C: | CHARACTER (individual character) |
| | B: | BOOL |
| | V: | VARIANT |
| 2 Limits | Limit value MIN, limit value MAX<br>Default: Empty<br>The limit values are separated by a comma. Limits can be specified for types I, C and R in decimal formats or as characters in the form "A", "F". | |
| Toggle field | List with predefined entries in the I/O field: The list starts with a * and the entries are separated by commas. The entries can be assigned a value.<br>For the toggle field, the entry for the limit is interpreted as a list. If only one * is entered, a variable toggle field is created.<br>Default: none | |
| Table identifiers | Identifiers of a table containing NCK/PLC values of the same type, which can be addressed via a channel block. The table identifier is differentiated from limits or toggle fields by the addition of a % sign in front of it.The file containing the table description can be specified by adding a comma after the identifier and then inserting the name of the file. | |
| 3 Default setting | If a default setting has not been configured and no system or user variable has been assigned to the variable, the first element of the toggle field is assigned. If no toggle field has been defined, there is no default setting, i.e., the status of the variable is "not calculated".<br>Default: No default | |
| 4 Texts | The sequence is specified. Instead of a short text, an image can also be displayed.<br>Default: Empty | |
| | Long text: | Text in the display line |
| | Short text: | Name of the dialog element |
| | Graphic text: | Text refers to the terms in the graphics |
| | Unit text: | Unit of the dialog element |

<table>
<tr><th>Parameter</th><th colspan="2">Description</th></tr>
<tr><td rowspan="7">5 Attributes</td><td colspan="2">The attributes influence the following properties:<br>• Input mode<br>• Access level<br>• Alignment of short text<br>• Font size<br>• Limits<br>• Response when dialog is opened in terms of CHANGE block<br>The attributes are separated by commas and appear in any order. The attributes are not valid for toggle fields. A definition can be made for each component.</td></tr>
<tr><td>Input mode</td><td>wr0: Input/output field invisible, short text visible<br>wr1: Read (no focus possible for input)<br>wr2: Read and write (line appears in white)<br>wr3: wr1 with focus<br>wr4: All variable elements invisible, no focus possible<br>wr5: The value entered is saved immediately on every keystroke (in contrast to wr2, where it is only saved when the field is exited or RETURN is pressed).<br>Default: wr2</td></tr>
<tr><td>Access level</td><td>empty: Can always be written<br>ac0...ac7: Protection levels<br>If the access level is not adequate, then the first line is displayed in gray, default setting: ac7</td></tr>
<tr><td>Alignment of short text</td><td>al0: Left-justified<br>al1: Right-justified<br>al2: centered<br>Default: al0</td></tr>
<tr><td>Font size</td><td>fs1: Default font size (8 pt.)<br>fs2: Double font size<br>Default: fs1<br>The clearances between the lines is defined. With the default font size, 16 lines can fit into the dialog. Graphics and unit text can only be configured in the default font size.</td></tr>
<tr><td>Limits</td><td>Consequently, it is possible to check whether the values of the variable are within the MIN and MAX limits specified.<br>Default: Determined by specified limits<br>li0: No check<br>li1: Check with respect to min.<br>li2: Check with respect to max.<br>li3: Check with respect to min. and max.</td></tr>
<tr><td>Behavior when opening</td><td>cb attributes specified for a variable in a variables definition take priority over the cb default setting in the dialog definition. Multiple attributes are separated by commas.</td></tr>
</table>

| Parameter | Description | |
|---|---|---|
| | cb0: | The CHANGE block defined for this variable is processed when the dialog is opened (default setting). Multiple attributes are separated by commas. |
| | cb1: | The CHANGE block defined for this variable is then only processed if the value of the variable changes. |
| 6 Help display | Help display file: | Name of the PDF-file<br>Default: Empty |
| | The name of the Help display file appears in double quotation marks. The display appears automatically (instead of the previous graphic) if the cursor is positioned on this variable. | |
| 7 System or user variable | An item of system or user data on the NC/PLC can be assigned to the variable.<br>The system or user variable appears in double quotation marks.<br>References: Parameter Manual, "List of System Variables" /PGA1/ | |
| 8 Position of short text | Position of short text (distance from left, distance from top, width)<br>The positions are entered in pixels and relate to the upper left-hand corner of the main body of the dialog. The entries are separated by commas. | |
| 9 Position of input/output field | Position of input/output field (distance from left, distance from top, width, height)<br>The positions are entered in pixels and relate to the upper left-hand corner of the main body of the dialog. The entries are separated by commas. If this position changes, the positions of the short text, graphic text and unit text also change. | |
| 10 Colors | Foreground color, background color: The colors are separated by a comma. Color settings are only relevant to the input/output field; colors cannot be specified for the other texts.<br>Range of values: 1...10<br>Default:<br>Foreground color: Black, background color: white<br>The default colors of the input/output field are determined by the Write mode: | |
| | wr0: | Foreground and background color: Window background color |
| | wr1: | Text color: Black, background color for window |
| | wr2: | Text color: Black, background color: white |
| | wr3: | As wr0 |
| | wr4: | As wr1 |
| | wr5: | As wr2 |
| 11 Help<br>(HMI Advanced only) | Help file: | Path to the PDF-file |
| | Index: | Index in the Help text file to the Help text |
| | Help text: | Help text for display in the Help text file |

| Parameter | Description |
|---|---|
| | The data is separated by commas, the sequence is defined. The Help file and Help text appear in double quotation marks.<br>The PDF files must be filed in folder CUS.DIR\hlp.dir or CST.DIR\hlp.dir. PDF and text files corresponding to one another must have the same name. The name of the PDF file must be written in capital letters in the txt file.<br>Several help references can be configured in a loop (Help loop) for each dialog element, i.e., the references are called up in succession and when the last reference closes, the first is displayed again.<br>If a second or subsequent Help call is linked to the same file/index/Help text, the entries do not have to be made.<br>Help is displayed when the cursor is positioned on this field and the information button is pressed. |

### 2.3.3 Detailed information about the variable type [1]

#### Variable type INTEGER

The following extensions for determining the display in the input/output field and the memory utilization are possible for the "INTEGER" type:

- **2nd character** in the extension data type

| Display format | |
|---|---|
| B | Binary |
| D | Decimal signed |
| H | hexadecimal |
| No data | Decimal signed |

- **3rd and/or 4th character** in the extension data type

| Memory utilization | |
|---|---|
| B | Byte |
| W | Word |
| D | Double Word |
| BU | Byte, Unsigned |
| WU | Word, Unsigned |
| DU | Double word, Unsigned |

### Sequence of characters in the INTEGER data type

1. "I" Basic INTEGER designation
2. Display format
3. Memory utilization
4. "U" Unsigned

| Valid INTEGER type specifications: | |
|---|---|
| IB | Integer variable 32 bits in binary notation |
| IBD | Integer variable 32 bits in binary notation |
| IBW | Integer variable 16 bits in binary notation |
| IBB | Integer variable 8 bits in binary notation |
| I | Integer variable 32 bits in decimal notation signed |
| IDD | Integer variable 32 bits in decimal notation signed |
| IDW | Integer variable 16 bits in decimal notation signed |
| IDB | Integer variable 8 bits in decimal notation signed |
| IDDU | Integer variable 32 bits in decimal notation unsigned |
| IDWU | Integer variable 16 bits in decimal notation unsigned |
| IDBU | Integer variable 8 bits in decimal notation unsigned |
| IH | Integer variable 32 bits in hexadecimal notation |
| IHDU | Integer variable 32 bits in hexadecimal notation |
| IHWU | Integer variable 16 bits in hexadecimal notation |
| IHBU | Integer variable 8 bits in hexadecimal notation |

### VARIANT variable type

The VARIANT variable type is determined by the data type of the last value assignment.
It can be scanned using the ISNUM or ISSTR functions. The VARIANT type is mainly suited to write either variable names or numerical values to the NC code.

### Programming

The data type of variables can be checked:

| | | |
|---|---|---|
| Syntax: | ISNUM *(VAR)* | |
| Parameters: | VAR | Name of the variable whose data type is to be checked. |
| | | The result of the scan can be: |
| | FALSE = | not a numerical variable (data type = STRING) |
| | TRUE = | numerical variable (data type = REAL) |

| | | |
|---|---|---|
| Syntax: | **ISSTR** *(VAR)* | |
| Parameters: | VAR | Name of the variable whose data type is to be checked. |
| | | The result of the scan can be: |
| | FALSE = | numerical variable (data type = REAL) |
| | TRUE = | not a numerical variable (data type = STRING) |
| Example | | |

```
IF ISNUM(VAR1) == TRUE
IF ISSTR(REG[4]+2) == TRUE
```

The display mode of variables can be changed:

- For INTEGER, the display type can be changed.

| | |
|---|---|
| B | Binary |
| D | Decimal signed |
| H | hexadecimal |

unsigned

With the addition of U for Unsigned

- For REAL data types, only the number of places after the decimal point can be changed.

  Changing the type is illegal and generates an error message in the ERROR.COM file.

  Example

  ```
  Var1.typ = "IBW"
  Var2.typ = "R3"
  ```

## Number representation

Numbers can be represented in either binary, decimal, hexadecimal or exponential notation. Numerical values in binary, hexadecimal and exponential notation must be enclosed in single quotation marks:

| | |
|---|---|
| Binary | 'B01110110' |
| decimal | 123.45 |
| hexadecimal | 'HF1A9' |
| exponential | '-1.23EX-3' |

Examples:

```
VAR1 = 'HF1A9'
REG[0]= 'B01110110'
DEF VAR7 = (R//'-1.23EX-3')
```

---

**Note**

When codes are generated with the "GC" function, only numerical values in decimal or exponential notation are evaluated, but **not** those in binary or hexadecimal notation.

---

## 2.3.4 Detailed information about the toggle field [2]

### Description

The toggle field extension function can be used to display texts (entries in toggle field) as a function of NC/PLC variables. A variable, which makes use of a toggle field extension, is read-only.

### Programming

| | | |
|---|---|---|
| Syntax: | **DEF Identifier =(Variable type /+ $Text number \| * value="\\display"[,value="\\image2.bmp"][, ...]**<br>/[Default]<br>/[Texts(Long text, Short text, Graphic text, Units text)]<br>/[Attributes]<br>/[Help display]<br>/[System or user variable]<br>/[Position of short text]<br>/[Position input/output field(Left, Top, Width, Height)]<br>/[Colors]<br>/[Help]) | |
| Description: | When the dialog is opened, the content of text number $85015 is displayed in the input/output field. Default value 15 is entered in system variable DB90.DBB5. If the value saved in system variable DB90.DBB5 changes, then the displayed text number $(85000 + <DB90.DBB5>) is recalculated in response to every change. | |
| Parameters: | Variable type | Type of variables specified in the system or user variable |
| | Text number | Number (basis) of the language-specific text valid as the basis number. |
| | System or user variable | System or user variable (offset) via which the final text number (basis + offset) is displayed. |
| Example | `DEF VAR1=(IB/+ $85000/15////"DB90.DBB5")` | |

### Variable toggle field

It is possible to assign a variable toggle field to a dialog element, i.e., when the toggle key is pressed, a value configured in a CHANGE method is assigned to the variable.

An asterisk * is entered in the Limits or Toggle Field property to identify a variable toggle field when a variable is defined.

Example: `DEF VAR1=(S/*)`

### Toggle-field-dependent displays

The toggle field is overlaid with alternating graphics: If the value of the flag byte is 1, "image1.bmp" will appear. If it is 2, "image2.bmp" will appear.

```
DEF VAR1=(IDB/*1="\\image1.bmp",
               2="\\image2.bmp"//,$85000/wr1//"MB[0]"//160,40,50,50)
```

The size and position of the image is defined under "Position of input/output field (left, top, width, height)".

## 2.3.5 Detailed information about the default setting [3]

### Overview

A variable can assume various states depending on whether a default value, or a system or user variable, or both, has been assigned to the variable field (I/O field or toggle field). (Not calculated: Toggling is not possible until a valid value is assigned to the variable).

### Scope of the default settings

| If... | | | Then... |
|---|---|---|---|
| Field type | Default setting | System or user variable | Reaction of field type |
| I/O field | yes | yes | Write default value to system or user variable |
| | No | yes | Use system or user variable as default value |
| | Error | yes | Not calculated, system or user variable is not written into/used. |
| | yes | No | Default setting |
| | No | No | Not calculated |
| | Error | No | Not calculated |
| | yes | Error | Not calculated |
| | No | Error | Not calculated |
| | Error | Error | Not calculated |
| Toggle | yes | yes | Write default value to system or user variable |
| | No | yes | Use system or user variable as default value |
| | Error | yes | Not calculated,<br>system or user variable not written/used |
| | yes | No | Default setting |

| If... | | | Then... |
|---|---|---|---|
| | No | No | Default = first toggle field element |
| | Error | No | Not calculated |
| | yes | Error | Not calculated |
| | No | Error | Not calculated |
| | Error | Error | Not calculated |

## 2.3.6 Detailed information about the position of the short text [8] and position of the input/output field [9]

### Overview

The short text and graphic text, as well as the input/output field and unit text, are each treated like a unit, i.e., position settings for short text apply to the graphic text and settings for the input/output field and to unit text.

### Programming

The configured position entry overwrites the default value, i.e., only one value can be changed. If no position settings have been configured for subsequent screen form elements, then the position settings for the preceding screen form element are applied.

If no positions have been specified for any dialog elements, the default setting is applied. By default, the column width for the short text and input/output field is calculated for each line based on the number of columns and maximum line width, i.e.,column width = maximum line width/number of columns.

The width of the graphics and unit text is predefined and optimized to suit the requirements of programming support. If graphics or unit text has been configured, the width of the short text or I/O field is reduced accordingly.

The order of short text and I/O field can be reversed by position settings.

## 2.3.7 Detailed information about on Help [11] (HMI-Advanced only)

### Description

The Help loop for a dialog element can be extended or deleted during runtime. The Help loop can be extended as required by calling the function repeatedly.

### Programming

| | | | |
|---|---|---|---|
| Syntax: | **ADDHTX** (Identifier, Help file, Index, Help text) | | |
| Description: | Extends the Help loop | | |
| Parameters: | Identifier | Name of the variable whose Help loop is to be expanded. | |
| | Help file: | Path specification of the file (PDF format) | |
| | Index: | Index in the Help text file to the Help text | |
| | Help text: | Help text for display in the Help text file | |
| Example | `ADDHTX(VAR1,"C:\OEM\HLP\MYHLP.PDF",15,"Machine data")` | | |

| | | | |
|---|---|---|---|
| Syntax: | **CLRHTX** (Identifier) | | |
| Description: | Deletes the Help loop | | |
| Parameters: | Identifier | Name of the variable whose Help loop is to be deleted | |
| | Help file: | Path specification of the file (PDF format) | |
| | Index: | Index in the Help text file to the Help text | |
| | Help text: | Help text for display in the Help text file | |
| Example | `CLRHTX(VAR1)` | | |

## 2.3.8 Application examples

### Help variables

Help variables are internal arithmetic variables. Arithmetic variables are defined like other variables, but have no other properties apart from variable value and status,
i.e., Help variables are not visible in the dialog. Help variables are of the VARIANT type.

## Programming

| | |
|---|---|
| Syntax: | DEF *Identifier* |
| Description: | Internal arithmetic variables of the VARIANT type |
| Parameters: | Identifier: Name of Help variables |
| Example | `DEF OTTO ;Definition of a Help variable` |

| | | |
|---|---|---|
| Syntax: | Identifier.val = *Help variable value* | |
| | Identifier = *Help variable value* | |
| Description: | A value is assigned to a Help variable in a method. | |
| Parameters: | Identifier: | Name of Help variables |
| | Help variable value: | Content of the Help variables |

**Example**

```
LOAD
  OTTO = "Test"            ; Assign the value "Test" to the Otto Help variable
END_LOAD
LOAD
  OTTO = REG[9].VAL        ; Assign value of register to the Help variable
END_LOAD
```

## Calculation with variables

Variables are calculated every time you exit an input/output field (by pressing the ENTER or TOGGLE key). The calculation is configured in a CHANGE method that is processed every time the value changes.

You can scan the variable status to ascertain the validity of the value of the variable, e.g.,

```
Var1 = Var5 + SIN(Var2)
Otto = PI * Var4
```

## Addressing system variables indirectly

A system variable can also be addressed indirectly, i.e., as a function of another variable:

```
PRESS(HS1)
  AXIS=AXIS+1
  WEG.VAR="$AA_DTBW["<<AXIS<<"]"   ;Address axis address via variable
END_PRESS
```

### Changing the softkey label

Example

```
HS3.st = "New Text"   ;Change softkey label
```

## 2.3.9 Example 1: Assigning the Variable Type, Texts, Help, Colors properties

### Example 1

Assigning the Variable Type, Texts, Help, Colors properties

**DEF Var1 = (R///,"Actual value",,"mm"//"Var1.bmp"////8,2)**

| | |
|---|---|
| Variable Type: | REAL |
| Limits or toggle field entries: | none |
| Default: | none |
| Texts: | |
| Long text: | None |
| Short text: | Actual value |
| Graphic text: | none |
| Unit text: | mm |
| Attributes: | none |
| Help display: | Var1.bmp |
| System or user variable: | none |
| Position of short text: | No data, i.e., default position |
| Position of input/output field: | No data, i.e., default position |
| Colors: | |
| Foreground color: | 8 |
| Background color: | 2 |
| Help: | none |

## 2.3.10 Example 2: Assigning the Variable Type, Limits, Attributes, Short Text Position properties

### Example 2

Assigning the Variable Type, Limits, Attributes, Short Text Position properties

**DEF Var2 = (I/0,10///wr1,al1///,,300)**

| | |
|---|---|
| Variable Type: | INTEGER |
| Limits or toggle field entries: | MIN: 0 |
| | MAX: 10 |
| Default: | none |
| Texts: | none |
| Attributes: | |
| Input mode | read-only |
| Alignment of short text | Right-justified |
| Help display: | none |
| System or user variable: | none |
| Position of short text: | |
| Distance from left | None |
| Distance from top | None, i.e., default distance from top left |
| Width: | 300 |
| Position of input/output field: | No data, i.e., default position |
| Colors: | No data, i.e., default |
| Help: | none |

## 2.3.11 Example 3: Assigning the Variable Type, Default, System or User Variable, Input/Output Field Position properties

### Example 3

Assigning the Variable Type, Default, System or User Variable, Input/Output Field Position properties

**DEF Var3 =(R//10////"$R[1]"//300,10,200//"Help.pdf",1,"Help1")**

| | |
|---|---|
| Variable Type: | String |
| Limits or toggle field entries: | none |
| Default: | 10 |
| Texts: | none |
| Attributes: | none |

**DEF Var3 =(R//10////"$R[1]"//300,10,200//"Help.pdf",1,"Help1")**

| | |
|---|---|
| Help display: | none |
| System or user variable: | $R[1] (R-Parameter 1) |
| Position of short text: | Default position in relation to input/output field |
| Position of input/output field: | |
| Distance from left | 300 |
| Distance from top | 10 |
| Width: | 200 |
| Colors: | No data, i.e., default |
| Help: | In file Help.pdf, Help with the Help text "Help1" is called on the page with the index 1 when the `<i>` button is pressed. |

## 2.3.12 Examples relating to toggle field, Help call and image display

### Example 4

Various entries in the toggle field:

Limits or toggle field entries:

```
DEF Var1 = (I/* 0,1,2,3)
DEF Var2 = (S/* "In", "Out")
DEF Var3 = (B/* 1="In", 0="Out")   ;1 and 0 are values, "In" and "Out" are displayed.
DEF Var4 = (R/* ARR1)              ;ARR1 is the name of an array.
```

### Example 5 (HMI Advanced only)

Several Help calls per dialog element:

**DEF Var5 = (R//////////"Help1.pdf",1,"Help1",,2,"Help2","Help3.pdf",3,)**

| | |
|---|---|
| Variable Type: | REAL |
| Limits or toggle field entries: | none |
| Default: | none |
| Texts: | none |
| Attributes: | none |
| Help display: | none |
| System or user variable: | none |
| Position of short text: | None |
| Position of input/output field: | none |
| Colors: | No data, i.e., default |

| DEF Var5 = (R//////////"Help1.pdf",1,"Help1",,2,"Help2","Help3.pdf",3,) | | |
|---|---|---|
| Help: | 1. Entry in the Help loop | |
| | Help file: | HELP1.PDF |
| | Index: | 1 |
| | Help text: | Help1 |
| | 2. Entry in the Help loop | |
| | Help file: | HELP2.PDF |
| | Index: | 2 |
| | Help text: | Help2 |
| | 3. Entry in the Help loop | |
| | Help file: | HELP3.PDF |
| | Index: | 3 |
| | Help text: | Help3 |

## Example 6

Displaying an image instead of a short text: the size and position of the image is defined under "Position of input/output field (Left, Top, Width, Height)".

| DEF VAR6= (V///,"\\image1.bmp" ////160,40,50,50) | |
|---|---|
| Variable Type: | VARIANT |
| Limits or toggle field entries: | none |
| Default: | none |
| Texts: | None |
| Attributes: | none |
| Help display: | none |
| System or user variable: | none |
| Position of short text: | image1.bmp |
| Position of input/output field | |
| Distance from left: | 160 |
| Distance from the top: | 40 |
| Width: | 50 |
| Height: | 50 |
| Colors: | No data, i.e., default |
| Help: | none |

### 2.3.13 Use of strings

#### Strings

Strings can be used as part of the configuration. These allow text to be displayed dynamically or different texts to be chained for the purpose of code generation.

#### Rules

The following rules must be observed with regard to string variables:

- Logic operations are processed from left to right.
- Nested expressions are solved from the inside outwards.
- No distinction is made between uppercase and lowercase type.

Strings can be deleted simply by assigning a blank string.

Strings can be appended after the equality sign using the operator "<<". Quotation marks (") in the string are represented by two successive quotation mark symbols. Strings can be checked for equality in IF instructions.

#### Example

Default settings for the following examples:

```
VAR1.VAL = "This is an"
VAR8.VAL = 4
VAR14.VAL = 15
VAR2.VAL = "Error"
$85001 = "This is an"
$85002 = "Alarm text"
```

Editing strings:

- Chaining of strings:
  ```
  VAR12.VAL = VAR1 << " Error."   ;Result: "This is an error"
  ```
- Deleting a variable:
  ```
  VAR10.VAL = ""   ;Result: Blank string
  ```
- Setting a variable with a text variable:
  ```
  VAR11.VAL = VAR1.VAL   ;Result: "This is an"
  ```
- Data type matching:
  ```
  VAR13.VAL ="This is the " << (VAR14 - VAR8) << "th error"
              ;Result: "This is the 11th error"
  ```

- Treatment of numerical values:

```
VAR13.VAL = "Error " << VAR14.VAL << ": " << $T80001 << $T80002
         ;Result: "Error 15: This is an alarm text"
IF VAR15 == "Error"   ;Strings in IF statement
  VAR16 = 18.1234
         ;Result: VAR16 equals 18.1234,
         ;if VAR15 equals "Error".
ENDIF
```

- Quotation marks within a string:

```
VAR2="Hello, this is a "" Test"
   ;Result: Hello, this is a " Test
```

- System or user-variable strings dependent on variable content:

```
VAR2.Var = "$R[" << VAR8 << "]"   ;Result: $R[4]
```

## 2.3.14 CURPOS variable

### Description

Using the CURPOS variable, it is possible to display or manipulate the position of the cursor in the active input field of the current dialog. The variable indicates how many characters are located in front of the cursor. If the cursor is located at the start of the input field, then CURPOS assumes the value of 0. If the value of CURPOS is changed, then the cursor is positioned at the appropriate location in the input field.

In order to be able to respond to changes in the variable value, it is possible to monitor for changes using a CHANGE block. If the value of CURPOS changes, then a jump is made to the CHANGE block and the instructions contained there are executed.

## 2.3.15 CURVER variable

### Description

The CURVER (CURrent VERsion) property allows the programming to be adapted in order to handle different versions. The CURVER variable is read-only.

---

**Note**

Even if previously recompiled with an older version, the code is automatically generated with the most recent version. The "GC" command always generates the most recent version. An additional identifier indicating the generated version is inserted in the user comment of the generated code in versions > 0.

---

### Rules

The most recent dialog with all its variables is always displayed.

- Variables used previously may not be changed.
- New variables are inserted in the existing (cycle) programming in arbitrary order.
- It is not permissible to delete variables from a dialog from one version to the next.
- The dialog must contain all variables of all versions.

### Example

| | |
|---|---|
| `(IF CURVER==1 ...)` | ; When the code is recompiled, CURVER is automatically assigned the version of the recompiled code. |

## 2.3.16 ENTRY variable

### Description

The ENTRY variable can be used to check by what method a dialog has been called.

### Programming

| | | |
|---|---|---|
| Syntax: | ENTRY | |
| Description: | The ENTRY variable is a read only variable. | |
| Return Value: | | The result of the scan can be: |
| | 0 = | No programming support |
| | 1 = | Programming support (the dialog was called by programming support) |
| | 2 = | Programming support + default setting from the previous dialog (sub-dialog) |
| | 3 = | Programming support + recompilation |
| | 4 = | Programming support + recompilation with generated comments, with # sign |
| | 5 = | Programming support + recompilation with generated comments, without # sign |

### Example

```
IF ENTRY == 0
  DLGL("The dialog was not called during programming")
ELSE
  DLGL("The dialog was called during programming")
ENDIF
```

## 2.3.17 ERR variable

### description.

Variable ERR can be used to check whether the preceding lines have been executed correctly.

### Programming

| | | |
|---|---|---|
| Syntax: | ERR | |
| Description: | The ERR variable is read-only. | |
| Return Value: | | The result of the scan can be: |
| | FALSE = | previous line was executed error-free |
| | TRUE = | previous line was not executed error-free |

### Example

| | | |
|---|---|---|
| `VAR4 = Thread[VAR1,"CDM",3]` | ; | Output value from array |
| `IF ERR == TRUE` | ; | Scan to check whether value has been found in array |
| `  VAR5 = "Error accessing array"` | | |
| | ; | If the value has not been found in the array, the value "Error accessing array" is assigned to the variables. |
| `ELSE` | | |
| `  VAR5 = "All OK"` | ; | ;If the value has been found in the array, the value "All OK" is assigned to the variables. |
| `ENDIF` | | |

## 2.3.18 FILE_ERR variable

### description.

Variable FILE_ERR can be used to check whether the preceding GC or CP command has been executed correctly.

### Programming

| | |
|---|---|
| Syntax: | FILE_ERR |
| Description: | The FILE_ERR variable is read-only. |
| Return Value: | Possible results are:<br>0 = Operation okay<br>1 = Drive/path not available<br>2 = Path/file access error<br>3 = Drive not ready<br>4 = Incorrect file name<br>5 = File is already open<br>6 = Access denied<br>7 = Target path not available or not permitted<br>8 = Copy source same as target<br>10 = Internal error: FILE_ERR = 10 means that the error cannot be classified in the other categories. |

### Example

```
CP("D:\source.mpf","E:\target.mpf")
                                   ;   Copy from source.mpf to E:\target.mpf
IF FILE_ERR > 0                    ;   Scan to ascertain whether error has occurred
   IF FILE_ERR == 1                ;   Scan specific error numbers and output associated
                                       error text
      VAR5 = "Drive/path not available"
   ELSE
      IF FILE_ERR == 2
         VAR5 = "Path/file access error"
      ELSE
         IF FILE_ERR == 3
            VAR5 = "Wrong file name"
         ENDIF
      ENDIF
   ENDIF
ELSE
   VAR5 = "All OK"                 ;   If no errors have occurred in CP (or GC), "All OK"
                                       is output
ENDIF
```

## 2.3.19 FOC variable

### description.

The FOC variable can be used to control the input focus (the current active input/output field) in a dialog. Responses to cursor left, right, up and down movements, as well as PGUP, PGDN, are predefined and cannot be modified.

---

**Note**

The FOC function may not be initiated as a result of a navigation event. The cursor position may only be changed in softkey PRESS blocks, CHANGE blocks, etc.

The FOC function cannot be applied to variables with input mode wr = 0 and wr = 4 or to Help variables.

---

### Programming

| | | |
|---|---|---|
| Syntax: | **FOC** | |
| Description: | The variable can be read and written. | |
| Return Value: | Read | The result is the name of the variable to which the FOC function has been applied. |
| | Write | It is possible to assign either a string or a numerical value.A string is interpreted as a variable name and a numerical value as a variable index. |

### Example

```
IF FOC == "Var1"                ; Read focus
  REG[1] = Var1
ELSE
  REG[1] = Var2
ENDIF

FOC = "Var1"                    ; The input focus will be assigned to Variable 1.
FOC = 3                         ; The input focus will be assigned to the 3rd dialog element
                                  with WR ≥ 2.
```

### 2.3.20 S_CHAN variable

#### Description

The S_CHAN variable can be used to determine the number of the current channel for display or evaluation purposes.

## 2.4 Complex dialog elements

### 2.4.1 Array

#### Definition

An array can be used to organize data of the same data type stored in the memory in such a way that it is possible to access the data via an index.

#### Description

Arrays can be one- or two-dimensional. A one-dimensional array is treated like a two-dimensional array with just one line or column.

Arrays have start identifier //A and end identifier //END. The number of lines and columns is optional. An array is structured in the following way:

#### Programming

| | | |
|---|---|---|
| Syntax: | //A*(Identifier)* | |
| | (a/b...) | |
| | (c/d...) | |
| | ... | |
| | //END | |
| Description: | Defines array | |
| Parameters: | Identifier | Name of array |
| | a, b, c, d | Values of array |
| | | Values of the STRING type must be enclosed in double quotation marks. |

### Example

```
//A(Thread)                          ;  Size/lead/core diameter
(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
```

## 2.4.2 Accessing the value of an array element

### Description

The value of an array access operation can be transferred with property Value (identifier.val).

The line index (line number of the array) and the column index (column number of the array) each begin at 0. If a line index or column index is outside the array, the value 0 or a blank string is output and the ERR variable is set to TRUE. The ERR variable is also set to TRUE if a search concept cannot be found.

### Programming

| | | | |
|---|---|---|---|
| Syntax: | Identifier [Z,[M[,C]]].val or<br>Identifier [Z,[M[,C]]] | | |
| Description: | Access to one-dimensional array with only one column | | |
| Syntax: | Identifier [S,[M[,C]]].val] or<br>Identifier [S,[M[,C]]] or | | |
| Description: | Access to one-dimensional array with only one line | | |
| Syntax: | Identifier [Z,S,[M[,C]]].val or<br>Identifier [Z,S,[M[,C]]] | | |
| Description: | Access to two-dimensional array | | |
| Parameters: | Identifier: | | Name of array |
| | | Z: | Line value (line index or search concept) |
| | | S: | Column value (column index or search concept) |

M: Access mode

- 0 Direct
- 1 Searches the line, column directly
- 2 Searches the column, line directly
- 3 Searches
- 4 Searches line index
- 5 Searches column index

C: Compare mode

- 0 Search concept must be located in the range of values of the line or column.
- 1 Search concept must be located exactly.

Example

```
VAR1 = MET_G[REG[3],1,0].VAL  ;Assign Var1 a value from
                                array MET_G
```

## Access mode

- **"Direct" access mode**

  With "Direct" access mode (M = 0), the array is accessed with the line index in Z and the column index in S. Compare mode C is not evaluated.

- **"Search" access mode**

  In the case of access mode M = 1, 2 or 3, the search always commences in line 0 or column 0.

| Mode M | Line value Z | Column value S | Output value |
|---|---|---|---|
| 0 | Line index | Column index | Value from line Z and column S |
| 1 | Search concept:<br>Search in column 0 | Column index of column from which value is read | Value from line found and column S |
| 2 | Line index of line from which return value is read | Search concept:<br>Search in line 0 | Value from line Z and column found |
| 3 | Search concept:<br>Search in column 0 | Search concept:<br>Search in line 0 | Value from line and column found |
| 4 | Search concept:<br>Search in column S | Column index of search column | Line index |
| 5 | Line index of search line. | Search concept:<br>Search in line Z | Column index |

### Compare mode

When compare mode C = 0 is used, the content of the search line or search column must be sorted in ascending order. If the search concept is smaller than the first element or larger than the last, the value 0 or a blank string is output and the error variable ERR is set to TRUE.

When compare mode C = 1 is used, the search concept must be found in the search line or search column. If the search concept cannot be found, the value 0 or an empty string is output and the error variable ERR is set to TRUE.

## 2.4.3 Example Access to an array element

### Prerequisite

Two arrays are defined below. These are the basis for the following examples.

```
//A(Thread
                (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)
                ("DES" /     "PTCH" /    "CDM" )
                (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
```

## Examples

- Access mode example 1:

  The search concept is in Z. This key is always sought in column 0. The value from column S is output with the line index of the concept found.

  ```
  VAR1 = Thread[0.5,1,1]   ;VAR1 has the value 0.125
  ```

  Explanation:

  Search for value 0.5 in column 0 of "Thread" array and output the value found in column 1 of the same line.

- Access mode example 2:

  The search concept is in S. This concept is always searched for in line 0. The value from line Z is output with the column index of the concept found:

  ```
  VAR1 = ARRAY2[3,"PTCH",2]   ;VAR1 has the value 0.125
  ```

  Explanation:

  Search for column containing "PTCH" in line 0 of array "Array2". Output the value from the column found and the line with index 3.

- Access mode example 3:

  A search concept is in each of Z and S. The line index is searched for in column 0 with the concept in Z and the column index in line 0 with the concept in S. The value from the array is output with the line index and column index found:

  ```
  VAR1 = ARRAY2[0.6,"PTCH",3]   ;VAR1 has the value 0.15
  ```

  Explanation:

  Search for the line with the content 0.6 in column 0 of array "Array2", search for the column with the content "STG" in line 0. Transfer the value from the line and column found to VAR1.

- Access mode example 4:

  The search concept is in Z. S contains the column index of the column in which concept is being searched for. The line index of the concept found is output:

  ```
  VAR1 = Thread[0.125,1,4]   ;VAR1 has the value 2
  ```

  Explanation:

  Search for value 0.125 in column 1 of array "Thread" and transfer the line index of the value found to VAR1.

- Access mode example 5:

  Z contains the line index of line in which concept is being searched for. The search concept is in S. The column index of the concept found is output:

  ```
  VAR1 = Thread[4,0.2,5,1]  ;VAR1 has the value 1
  ```

  Explanation:

  Search in line 4 of the "Thread" array for the value 0.2 and transfer the column index of the value found to VAR1. Comparison mode 1 was selected because the values of line 4 are not sorted in ascending order.

### 2.4.4 Scanning the status of an array element

#### Description

The Status property can be used to run a scan to find out whether an array access operation is supplying a valid value.

#### Programming

| | | |
|---|---|---|
| Syntax: | *Identifier [Z, S, [M[,C]]]*.vld | |
| Description: | Status is a read-only property. | |
| Parameters: | Identifier | Name of array |
| Return Value: | FALSE = | invalid value |
| | TRUE = | valid value |

#### Example

```
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
```

## 2.4.5 Table grid (grid)

### Definition

In contrast to the array, the values of a table grid (grid) are continually updated. This involves a tabular representation of the values of system variables that can be addressed using one block in one channel.

### Assignment

A variables definition is assigned to the table-elements definition via a table identifier:

- The variables definition determines the values to be displayed and the table-elements definition determines the on-screen appearance and assignment. The table grid takes the properties of the input/output fields from the variables definition line.
- The visible area of the grid is determined by the width and height of the I/O field. Any lines or columns than cannot be seen can be displayed by scrolling horizontally and vertically.

### Description

The variables definition will contain a reference to a table description:

| `DEF` *Identifier* = | Identifier = Name of variable | | |
|---|---|---|---|
| | Variable type | → | 1 |
| | /[Limits or toggle field or table identifier] | → | 2 |
| | /[Default] | → | 3 |
| | /[Texts (Long text, Short text\|Image, Graphics text, Units text)] | → | 4 |
| | /[Attributes] | → | 5 |
| | /[Help display] | → | 6 |
| | /[System or user variable] | → | 7 |
| | /[Position of short text] | → | 8 |
| | /[Position input/output field(Left, Top, Width, Height)] | → | 9 |
| | /[Colors] | → | 10 |
| | /[Help] (HMI Advanced only) | → | 11 |

### Table identifier [2]

Identifiers of a table containing NCK/PLC values of the same type, which can be addressed via a channel block. The table identifier is differentiated from limits or toggle fields by the addition of a % sign in front of it. The file containing the table description can be specified by adding a comma after the identifier and then inserting the name of the file.

### System or user variable [7]

This parameter remains empty for table grids, because the column definition lines contain detailed information about the variables to be displayed. The table description can be provided in a dynamic format.

## 2.4.6 Defining table grids

### Description

The table block comprises:

- Header
- 1 to n column descriptions

### Programming

| | | | |
|---|---|---|---|
| Syntax: | //G*(Table identifier/Table type/Number of lines/ [Fixed line attribute],[Fixed column attribute])* | | |
| Description: | Defines table grids | | |
| Parameters: | Table identifiers | The table identifier is used without a leading % sign. It can only be used once in a dialog. | |
| | Table type | 0 (default) | Table for PLC or user data (NCK- and channel-specific data) |
| | | 1 | and others, reserved |
| | No. of lines | Number of lines including header<br>The fixed line or fixed column is not scrolled. The number of columns is the number of columns configured. | |
| | Fixed line attribute | 1: Active<br>0: Not active | |
| | Fixed column attribute | 1: Active<br>0: Not active | |

## 2.4.7 Defining columns

### Description

For table grids, it is advisable to use variables with an index. For PLC or NC variables, the index number is of significance with one or more indices.

The values displayed in a grid can be modified directly by the end user within the restrictions of the rights granted by the attributes and within any limits defined.

### Programming

| | | |
|---|---|---|
| Syntax: | *(Type/Limits/Empty/Long text,column header/Attributes/Help display/ System or user variable/Column width/Offset1, Offset2, Offset3)* | |
| Description: | Defines columns | |
| Parameters: | Similar to variables | |
| | Type | Data type |
| | Limits | Limit value MIN, limit value MAX |
| | Long text, column header | |
| | Attributes | |
| | Help display | |
| | System or user variable | The PLC or NC variables should be entered in double quotation marks. |
| | Column width | Entry in pixels. |
| | Offset | The increments to be used to accumulate each index in order to fill the column are specified in the assigned offset parameter:<br>• Offset1: Step width for the 1st index<br>• Offset2: Step width for the 2nd index<br>• Offset3: Step width for the 3rd index |

### Variable of type STRING

If the variable is a STRING type variable, the length must be specified in the type, e.g., `DEF CHAN STRING [16] TEXT[41]`.

The column definition for the CHAN variable, therefore, starts, e.g., (S16/... ).

### Column header from text file

The column header can be entered as text or text numbers ($8xxxx) and is not scrolled.

### Modifying column properties

The column properties, which can be modified dynamically (written) are:

- Limits (min,max),
- Column header (st),
- Attributes (wr, ac and li),
- Help display (hlp) and
- OPI-Variable (var).

Column properties are modified via the variable identifier in the definition line and the column index (starting at 1).

Example: `VAR1[1].st="Column 1"`

Column properties cannot be read in the LOAD block.

The wr, ac and li attributes can be specified for column definitions.

### See also

Dialog elements (Page 21)
List of accessible system variables (Page 194)

## 2.4.8 Focus control in the table grid

### Description

The Row and Col properties can be used to set and calculate the focus within a table:

- Identifier.Row
- Identifier.Col

### Programming

Each cell in a table has the Val and Vld properties.

In order to read and write cell properties, a line and column index must be specified in addition to the variable identifiers from the definition list.

| | |
|---|---|
| Syntax: | Identifier[Line index, column index].val or |
| | Identifier[Line index, column index] |
| Description: | Val properties |
| Syntax: | Identifier[Line index, column index].vld |
| Description: | Vld properties |

### Example

```
Var1[2,3].val=1.203
```

If the line and column indices are not specified, the indices of the focused cell apply. This corresponds to:

```
Var1.Row =2
Var1.Col=3
Var1.val=1.203
```

## 2.4.9 Example Defining columns

### Overview

The next three examples show the assignment between cells and PLC variables in a standard table (table type=0).

### Example 1:

The first line shows the column headers:

```
//G(MB_TAB/0/4/,1)
(I///,"MB 1 to MB 3"///"MB1"/100/1)
(I///,"MB 4 to MB 6"///"MB4"/100/1)
```

Result:

| MB 1 to MB 3 | MB 4 to MB 6 |
|---|---|
| Value(MB1) | Value(MB4) |
| Value(MB2) | Value(MB5) |
| Value(MB3) | Value(MB6) |

## Example 2:

Offset > 1 in the column definition; this results in the following lines and column assignments:

```
//G(MB_TAB/0/4/,1)
(I///," MB1, MB3, MB5"///"MB1"/100/2)
(I///," MB2, MB4, MB6"/// "MB2"/100/2)
```

**Result:** In each line, the index of the variable is increased by the offset (=2).

| MB1, MB3, MB5 | MB2, MB4, MB6 |
|---|---|
| Value(MB1) | Value(MB2) |
| Value(MB3) | Value(MB4) |
| Value(MB5) | Value(MB6) |

## Example 3:

Offset and index number in the column definition:

- In the first column, the first variable index for each line is increased by 1: Offset 1 = 1
- In the first column, the second variable index for each line is increased by 1: Offset 2 = 1

```
//G(MB_TAB/0/4/,1)
(IB///,"M1.1, M2.1, M3.1"/// "M1.1"/100/1)
(IB///,"M1.1, M1.2, M1.3"/// "M1.1"/100/,1)
```

**Result:**

| M1.1, M2.1, M3.1 | M1.1, M1.2, M1.3 |
|---|---|
| Value(M1.1) | Value(M1.1) |
| Value(M2.1) | Value(M1.2) |
| Value(M3.1) | Value(M1.3) |

**Other options:**

- Ascending numbers can be entered in the first column:
  Example: `(I///,"Line"///"0"/60/1)`
- Consecutive texts from the language files can be entered in the first column:
  Example: `(S///,"Line"///"$80000"/60/1)`

## 2.4.10 Example Loading different table grids

### Description

In this example the "dummygrid" table is first assigned to variable "VAR1". Depending on the content of the R-parameter R[0], either the "grid1" or "grid2" table will be loaded in the LOAD block. The tables are defined in the same file as the variable "VAR1".

```
//M(SCREEN FORM1/"GRID")
DEF VAR1=(R/% dummygrid//////200,75,300,85)
HS1=("")
HS2=("")
HS3=("")
HS4=("")
HS5=("")
HS6=("")
HS7=("")
HS8=("")
VS1=("")
VS2=("")
VS3=("")
VS4=("")
VS5=("")
VS6=("")
VS7=("EXIT",ac7,se1)
VS8=("")

LOAD
  REG[0] = RNP ("$R[0]")
    IF (REG[0] == 0)
       LG ("grid1", "var1")
   ELSE
       LG ("grid2", "var1")
   ENDIF
END_LOAD

PRESS(VS7)
   EXIT
END_PRESS

//END

//G(grid1/0/5/1,1)                                  ;  (Name/Type/Lines...)
```

| | |
|---|---|
| `(R///"Long text1","R1 to R4"/wr2//"$R[1]"/80/1)` | ; 1. Column, header "R1 to R4", from R1 with Offset1 |
| `(R///"Long text2","R5 to R8"/wr2//"$R[5]"/80/1)` | ; 2. Column from R5 |
| `(R///"Long text3","R9 to R15"/wr2//"$R[9]"/80/2)` | ; 3. Column from R9 with Offset2, R9 11 13 15 |
| `//END` | |
| | |
| **`//G(grid2/0/5/1,1)`** | ; (Name/Type/Lines...) |
| `(R///"Long text1","R1 to R4"/wr2//"$R[1]"/60/1)` | ; 1. Column, header "R1 to R4", from R1 with Offset1 |
| `(R///"Long text2","R5 to R8"/wr2//"$R[5]"/60/1)` | ; 2. Column from R5 |
| `(R///"Long text3","R9 to R15"/wr2//"$R[9]"/60/2)` | ; 3. Column from R9 with Offset2, R9 11 13 15 |
| `(R///"Long text4","R9 to R15"/wr2//"$R[9]"/60/2)` | ; 3. Column from R9 with Offset2, R9 11 13 15 |
| `//END` | |
| | |
| **`//G(dummygrid/0/5/1,1)`** | ; (Name/Type/Lines...) |
| `(R///"Long text1","R1 to R4"/wr2//"$R[1]"/80/1)` | ; 1. Column, header "R1 to R4", from R1 with Offset1 |
| `(R///"Long text2","R5 to R8"/wr2//"$R[5]"/80/1)` | ; 2. Column from R5 |
| `//END` | |

# 2.5 Softkey menus

## 2.5.1 Softkey description

### Overview

The names of the soft keys are predefined. Not all softkeys need to be assigned.

HSx x 1 - 8, horizontal softkeys 1 to 8

VSy y 1 - 8, vertical softkeys 1 to 8

The term softkey menu is used to refer to all the horizontal and vertical softkeys displayed on a screen form. In addition to the existing softkey menus, it is possible to define other menus, which partially or completely overwrite the existing menus.

The definition of a softkey menu (softkey menu definition block) is basically structured as follows:

| Definition block | Comment | Chapter reference |
|---|---|---|
| `//S...` | ;Start identifier of softkey menu | |
| `HSx=...` | ;Define softkeys | |
| `PRESS(HSx)`<br>`  LM...`<br>`END_PRESS` | ;Method start identifier<br>  ;Actions<br>;Method end identifier | See chapter "Methods" |
| `//END` | ;End identifier of softkey menu | |

## 2.5.2 Define softkey menu

### Description

Properties are assigned to softkeys during definition of the softkey menu.

### Programming

| | | |
|---|---|---|
| Syntax: | //S*(Identifier)*<br>...<br>//END | ;Start identifier of softkey menu<br><br>;End identifier of softkey menu |
| Description: | Defines softkey menu | |
| Parameters: | Identifier | Name of softkey menu |

| | | |
|---|---|---|
| Syntax: | SK = *(Text[, Access level][, Status])* | |
| Description: | Define softkey | |
| Parameters: | SK | softkey, e.g., HS1 to HS8, VS1 to VS8 |
| | Text | Enter text |
| | Display file name | "\\my_pic.bmp"<br>or via separate text file $85199, e.g., with the following text in the (language-specific) text file: 85100 0 0 "\\c:\pic\my_pic.bmp".<br>Size of image, which can be displayed on a softkey:<br>max. 80 x 34 pixels |
| | Access level | ac0 to ac7 (ac7: default) |
| | Status | se1: visible (default)<br>se2: disabled (gray text)<br>se3: displayed (last softkey used) |

---

**Note**

Enter %n in the softkey text to create a line break.

A maximum of 2 lines of 10 characters each are available in HMI Advanced and 2 lines of 9 characters each in HMI Embedded sl.

---

## Assigning the security level

The operator can only access information on the level for which he is authorized and all lower levels.

The meanings of the different protection levels are as follows: ac0 is the highest protection level and ac7 the lowest.

| Security level | Locked by | range |
|---|---|---|
| ac0 | Password | Siemens |
| ac1 | Password | Machine manufacturer |
| ac2 | Password | Service |
| ac3 | Password | User |
| ac4 | Keylock switch position 3 | Programmer, machine setter |
| ac5 | Keylock switch position 2 | Qualified operator |
| ac6 | Keylock switch position 1 | Trained operator |
| ac7 | Keylock switch position 0 | Semi-skilled operator |

## Example

| | | |
|---|---|---|
| `//S(Menu1)` | ; | Start identifier of softkey menu |
| `HS1=("NEW",ac6,se2)` | ; | Define softkey HS1, assign the label "OK", security level 6 and the status "disabled" |
| `HS3=("\\bild1.bmp")` | ; | Assign a graphic to the softkey |
| `HS5=("Exit")` | | |
| `VS2=("Subscreen")` | | |
| `VS3=($85011, ac7, se2)` | | |
| `VS7=("Abort", ac1, se3)` | ; | Define softkey HS1, assign the label "Cancel", security level 1 and the status "displayed". |
| `VS8=("OK", ac6, se1)` | | Define softkey VS8, assign the label "OK", security level 6 and the status "visible" |
| | | |
| `PRESS(HS1)` | ; | Method start identifier |
| `  HS1.st="Calculate"` | ; | Assign a label text to the softkey |
| `...` | | |
| `END_PRESS` | ; | Method end identifier |

```
PRESS(RECALL)              ;  Method start identifier
  LM("Screen form21")      ;  Load dialog
END_PRESS                  ;  Method end identifier
//END                      ;  Softkey menu end identifier
```

### <RECALL> key

In addition to the horizontal and vertical softkeys, a <RECALL> key is also available.

In contrast to the softkeys, the RECALL key does not need to be defined. The "Status" and "Access Level" properties can be assigned to the key during runtime.

If you do not assign any action (function, calculation of variables, property changes) to <RECALL>, you can use it to exit newly configured user interfaces and return to the standard application.

### Example

```
PRESS(RECALL)
  RECALL.ac = 1
  LM("Screen form5")
END_PRESS
```

## 2.5.3 Changing softkey properties during runtime

### Description

The softkey properties Text, Access Level and Status can be changed in the methods during runtime.

### Programming

| | | |
|---|---|---|
| Syntax: | SK.st = "Text"" | ;Softkey with label |
| | SK.ac = Access level | ;Softkey with security level |
| | SK.se = Status | ; Softkey with status |
| Description: | Assigns properties | |
| Parameters: | Text | Label text in inverted commas |
| | Access level | Range of values: 0...7 |

| | |
|---|---|
| Status | 1: visible and operator-controllable |
| | 2: disabled (gray text) |
| | 3: displayed (last softkey used) |

## Example

```
//S(Start)
HS7=("Example", ac7, se1)

PRESS(HS7)
  LM("Screen form3")
END_PRESS

//END

//M(Screen form3/"Example 3 : Graphics and
softkeys"/"MST.BMP")
HS1=("")
HS2=("")
HS3=("")
HS4=("")
HS5=("")
HS6=("")
HS7=("")
HS8=("")
VS1=("")
VS2=("")
VS3=("")
VS4=("")
VS5=("")
VS6=("")
VS7=("")
VS8=("OK", AC7, SE1)
PRESS(VS8)
  EXIT
END_PRESS
//END
```

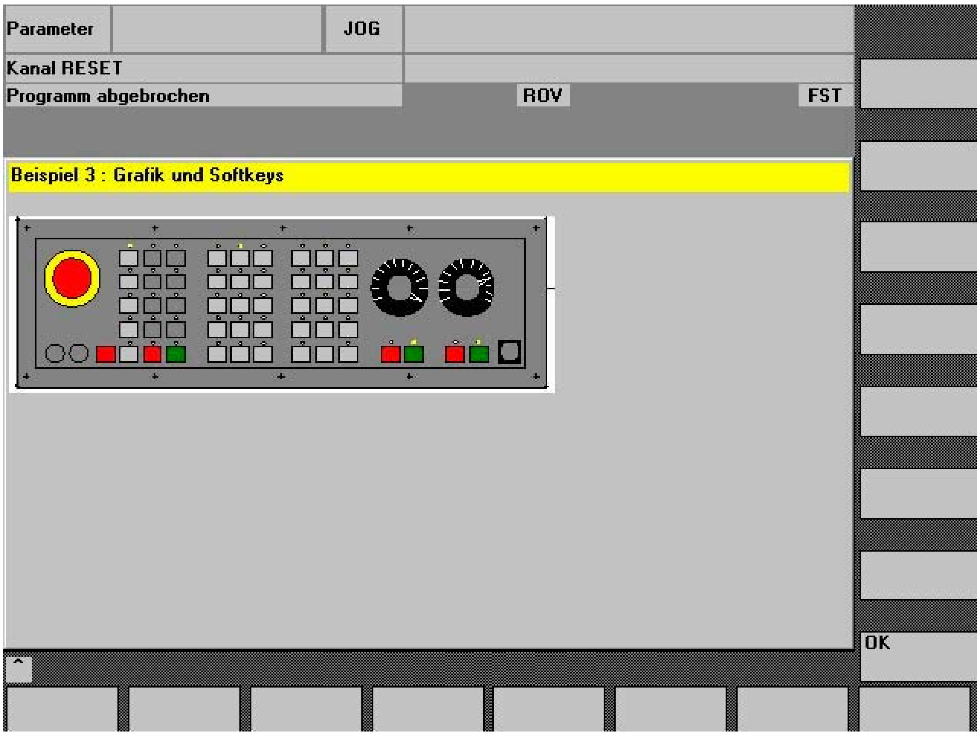



Figure 2-9 Example 3: Graphics and softkeys

## 2.5.4 Defining the start softkey

### Dialog-independent softkey

Start softkeys are dialog-independent softkeys that, rather than being called from a dialog, have been configured **before** the first new dialog. In order to access the start screen or a start softkey menu, the start softkey must be defined.

### Programming

The definition block for a start softkey is structured as follows:

| | |
|---|---|
| `//S(Start)` | ; Start identifier of start softkey |
| `HS1=(...)` | ; Define the start softkey: horizontal SK 1 |
| `PRESS(HS1)` | ; Method |
| `  LM...` | ; LM or LS function |
| `END_PRESS` | ; End of method |
| `//END` | ; End identifier of start softkey |

## 2.5.5 Functions for start softkeys

### Functions for dialog-independent softkeys

Only certain functions can be initiated with start softkeys.

The following functions are permitted:

- The **LM function** can be used to load another dialog: **LM***("Identifier"[,"File"])*
- The **LS function** can be used to display another softkey menu: **LS***("Identifier"[, "File"][, Merge])*
- You can use the **"EXIT" function** to exit newly configured user interfaces and return to the standard application.
- You can use the **"EXITLS" function** to exit the current user interface and load a defined softkey menu.
- The **"EXE" function** can be used on HMI Advanced to call a program created as an application with the HMI Advanced OEM package or to start the Free Contour Programming application. With HMI Embedded sl, "EXE" can only be used to start the Free Contour Programming application.

### PRESS method

The softkey is defined within the definition block and the "LM" or "LS" function is assigned in the PRESS method.

If the start softkey definition is designated as a comment (semicolon (;) at beginning of line) or the configuration file removed, the start softkey will not function.

| | |
|---|---|
| `//S(Start)` | ; Start identifier |
| `HS6=("1st screen form")` | ; horizontal SK 6 labeled "1st screen form" |
| `PRESS(HS6)` | ; PRESS method for horizontal SK 6 |
| `  LM("Screen form1")` | ; Load screen form1 function, where screen form 1 must be defined within the same file. |
| `END_PRESS` | ; End of PRESS method |
| `HS7=("2nd screen form")` | ; horizontal SK 7 labeled "2nd screen form" |
| `PRESS(HS7)` | ; PRESS method for horizontal SK 7 |
| `  LM("Screen form2")` | ; Load screen form2 function, where screen form 2 must be defined within the same file. |
| `END_PRESS` | ; End of PRESS method |
| `//END` | ; End identifier of entry block |

### Example

```
HS1 = ("new softkey menu")
HS2=("no function")
PRESS(HS1)
  LS("Menu1")                          ; load new softkey menu
END_PRESS
PRESS (HS2)                            ; empty PRESS method
END_PRESS
```

### See also

Load Softkey (LS) (Page 97)
Load Mask (LM) (Page 96)

## 2.6 Methods

### Overview

Various types of event (exit input field, actuate softkey) can initiate specific actions in dialogs and dialog-dependent softkey menus (softkey menus that are called from a newly configured dialog). These actions are configured in methods.

The following table shows the basic principle used to program a method:

| Definition block | Comment | Chapter reference |
|---|---|---|
| `PRESS(HS1)` | ;Method start identifier | |
| `LM...`<br>`  LS...` | ;Functions | See chapter "Functions" |
| `Var1.st = ...` | ;Changing properties | see chapter "Softkey menu"<br>and chapter "Dialog elements" |
| `Var2 = Var3 + Var4`<br>`...`<br>`EXIT` | ;Calculation with variables | See chapter "Defining variables" |
| `END_PRESS` | ;Method end identifier | |

## 2.6.1 CHANGE

### Description

CHANGE methods are executed if a variable value changes, i.e., variable calculations that are performed as soon as a variable value changes are configured within a CHANGE method.

There are two types of CHANGE method, i.e., element-specific and global:

- The **element-specific CHANGE method** is executed if the value of a specified variable changes. If a system or user variable is assigned to a variable, cyclic updating of the variable value can be configured in a CHANGE method.
- The **global CHANGE method** is executed if the value of any variable changes and no element-specific CHANGE method has been configured.

### "Element-specific" programming

| | | |
|---|---|---|
| Syntax: | CHANGE*(Identifier)*<br>...<br>END_CHANGE | |
| Description: | Changes the value of a specific variable | |
| Parameters: | Identifier | Name of the variable |

### Example

| | | |
|---|---|---|
| `DEF VAR1=(S//////"DB20.DBB1")` | ; | A system variable is assigned to Var1 |
| `CHANGE(VAR1)` | | |
| `  IF VAR1.Val <> 1` | | |
| `  VAR1.st="Tool OK!"` | ; | If the value of the system variable ≠ 1, the short text of the variable states: Tool OK! |
| `    otto=1` | | |
| `  ELSE` | | |
| `    VAR1.st="Attention: Error!"` | ; | If the value of the system variable = 1, the short text of the variable states: Attention: Error! |
| `    otto=2` | | |
| `  ENDIF` | | |
| `  VAR2.Var=2` | | |
| `END_CHANGE` | | |

### "Global" programming

| | |
|---|---|
| Syntax: | CHANGE()<br>...<br>END_CHANGE |
| Description: | Changes any variable value |
| Parameters: | - None - |

### Example

| | |
|---|---|
| `CHANGE()` | |
| `EXIT` | ; If any of the variable values change, the dialog will be terminated. |
| `END_CHANGE` | |

## 2.6.2 FOCUS

### Description

The FOCUS method is executed if the focus (cursor) is positioned on another field in the dialog.

The FOCUS method must not be initiated as a result of a navigation event. The cursor may only be moved in softkey PRESS blocks, CHANGE blocks, etc. Responses to cursor movements are are predefined and cannot be modified.

---

**Note**

Within the FOCUS block, it is not possible to select a different variable, nor can a new dialog be loaded.

---

### Programming

| | |
|---|---|
| Syntax: | FOCUS<br>...<br>END_FOCUS |
| Description: | Positions the cursor |
| Parameters: | - None - |

### Example

```
FOCUS
  DLGL("The focus has been placed on variable" << FOC << ".)
END_FOCUS
```

## 2.6.3 LOAD GRID

### Description

The table description can be made available dynamically within the LOAD block using the LG method.

In order to assign a table using the LG method, the variable must have already been defined as a grid variable and cross-referenced to an existing, valid table.

### Programming

| | | |
|---|---|---|
| Syntax: | LG *(Grid name, Variable name [,File name])* | |
| Description: | Loads a table | |
| Parameters: | Grid name | Name of the table (grid) in inverted commas |
| | Variable name | Name of the variable to which the table is to be assigned, in inverted commas |
| | File name | Name of the file in which the table (grid) is defined, in inverted commas. Only needs to be specified if the table is not defined within the file that also contains the definition of the variable. |

## 2.6.4 LOAD

### Description

The LOAD method is executed after the variable and softkey definitions (DEF Var1= ..., HS1= ...) have been interpreted. At this time, the dialog is not yet displayed.

### Programming

| | |
|---|---|
| Syntax: | LOAD<br>...<br>END_LOAD |
| Description: | Download |
| Parameters: | - None - |

### Example

```
LOAD                          ; Start identifier
  Screen form1.Hd =  $85111   ; Assign text for dialog header from language file
  VAR1.Min = 0                ; Assign MIN variable limit
  VAR1.Max = 1000             ; Assign MAX variable limit
END_LOAD                      ; End code
```

## 2.6.5 UNLOAD

### Description

The UNLOAD method is executed before a dialog is unloaded.

### Programming

| | |
|---|---|
| Syntax: | UNLOAD<br>...<br>END_UNLOAD |
| Description: | Unload |
| Parameters: | - None - |

### Example

```
UNLOAD
  REG[1] = VAR1                    ; Save variable in register
END_UNLOAD
```

## 2.6.6 OUTPUT

### Description

The OUTPUT method is executed if the "GC" function is called.Variables and Help variables are configured as an NC code in an OUTPUT method. The individual elements in a code line are linked by means of blanks.

---

**Note**

The NC code can be generated in an extra file by means of file functions and transferred to the NC.

---

### Programming

| | | |
|---|---|---|
| Syntax: | OUTPUT *(Identifier)*<br>...<br>END_OUTPUT | |
| Description: | Outputs variables in the NC program. | |
| Parameters: | Identifier | Name of OUTPUT method |

### Block numbers and skip identifiers

The OUTPUT block must not contain line numbers or skip identifiers if you wish to keep the line numbers and hide markings directly set with active program support in the parts program in case of recompilations.

Editor changes in the parts program produce the following response:

| Condition | Response |
|---|---|
| Number of blocks remains unchanged. | Block numbers are retained. |
| Number of blocks is reduced. | The highest block numbers are canceled. |
| Number of blocks is increased. | New blocks are not numbered. |

### Example

```
OUTPUT(CODE1)
  "CYCLE82(" Var1.val "," Var2.val "," Var3.val ","Var4.val "," Var5.val
"," Var6.val ")"
END_OUTPUT
```

## 2.6.7 PRESS

### Description

The PRESS method is executed when the corresponding softkey is pressed.

### Programming

| | | | |
|---|---|---|---|
| Syntax: | PRESS*(softkey)* | | |
| | ... | | |
| | END_PRESS | | |
| Identifiers: | Pressing a softkey | | |
| Parameters: | Softkey | Name of softkey: HS1 - HS8 and VS1 - VS8 | |
| | RECALL | <RECALL> key | |
| | PU | Page Up | Screen up |
| | PD | Page Down | Screen down |
| | SL | Scroll left | Cursor left |
| | SR | Scroll right | Cursor right |
| | SU | Scroll up | Cursor up |
| | SD | Scroll down | Cursor down |

### Example

```
HS1 = ("another softkey menu")
HS2=("no function")
PRESS(HS1)
  LS("Menu1")                                 ;   load another softkey menu
  Var2 = Var3 + Var1
END_PRESS
PRESS (HS2)
END_PRESS
```

```
PRESS(PU)
  INDEX = INDEX -7
  CALL("UP1")
END_PRESS
```

### 2.6.8 Example Version management with OUTPUT blocks

#### Overview

Additional variables can be added to existing dialogs when expanding the user interface. A version identifier in parentheses is appended to the additional variables in the definition following the variable name: (0 = Original, is not written), 1 = Version 1, 2 = Version 2, etc.

**Example**

```
DEF var100=(R//1)            ; Original, corresponds to Version 0
DEF var101(1)=(S//"Hello")   ; Expansion with effect from Version 1
```

When writing the OUTPUT block, you can specify which variables are written, with reference to a particular version identifier.

**Example**

```
OUTPUT(NC1)      ; Only the variables of the original version are made available
                 in the OUTPUT block.
OUTPUT(NC1,1)    ; The variables of the original version and the expansions
                 with version identifier 1 are made available in the OUTPUT
                 block
```

The OUTPUT block for the original version does not need a version identifier, however you can specify it with 0. OUTPUT(NC1) is equivalent to OUTPUT(NC1,0). Version identifier n in the OUTPUT block includes all variables of the originals 0, 1, 2, ... up to and including n.

#### Programming with version identifier

```
//M(XXX)                    ; Version 0 (default)
DEF var100=(R//1)
DEF var101=(S//"Hello")
DEF TMP
VS8=("GC")
PRESS(VS8)
   GC("NC1")
```

```
END_PRESS

OUTPUT(NC1)
var100",,"var101
END_OUTPUT

; ************ Version 1, extended definition ***************
//M(XXX)
DEF var100=(R//1)
DEF var101=(S//"Hello")
DEF var102(1)=(V//"HUGO")
DEF TMP
VS8=("GC")
PRESS(VS8)
   GC("NC1")
END_PRESS
...

OUTPUT(NC1)                        ; Original and the new version in addition
var100","var101
END_OUTPUT
...

OUTPUT(NC1,1)                      ; Version 1
var100","var101"," var102
END_OUTPUT
```

## 2.7 Functions

### Overview

A variety of functions are available in dialogs and dialog-dependent softkey menus. These can be activated by specific events (exit input field, actuate softkey) and configured in methods.

### Subroutines

Repeatedly used configuring instructions or others, which define the process for a particular operation can be configured in subprograms. Subprograms can be loaded into the main program or other subprograms at any time and executed as often as necessary, i.e., the instructions they contain do not need to be configured repeatedly. The definition blocks of the dialogs/softkey menu constitute a main program.

### PI services

The PI_SERVICE function can be used to start PI Services (Program Invocation Services) from the PLC in the NC area.

### External functions (only HMI Advanced)

Additional, user-specific functions can be integrated by means of external functions. External functions are stored in a DLL file and identified by an entry in the definition lines of the configuration file.

### See also

List of PI services (Page 205)

External functions (only HMI Advanced) (Page 114)

## 2.7.1 Activate Program (AP)

### Description

The AP (Activate Program) function transfers a file from the passive HMI file system to the active NC file system. The file is loaded into the NC and enabled and then deleted in the HMI file system. With HMI Embedded sl, this function has the same effect as Set enable.

### Programming

| | | |
|---|---|---|
| Syntax: | AP("*File*") | |
| Description: | Transfers a file from the passive HMI file system to the active NC file system | |
| Parameters: | File | Complete path name of HMI file to be transferred |

### Example

```
//M(TestGC/"Code generation:")
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)                    ; Write code from the OUTPUT method to file
                                          \MPF.DIR\MESSEN.MPF
END_PRESS
PRESS(HS8)
  D_NAME = "\MPF.DIR\MESSEN.MPF"
  AP(D_NAME)                            ; Load file into NC
END_PRESS
```

## 2.7.2 Define block (//B)

### Description

In the program file, subprograms are identified by the block identifier //B and terminated with //END. Several subprograms can be defined under each block identifier.

---

**Note**

The variables used in the subprogram must be defined in the dialog in which the subprogram is called.

---

### Programming

A block is structured in the following way:

| | | |
|---|---|---|
| Syntax: | //B(*Block name*)<br>SUB*(Identifier)*<br>END_SUB<br>[SUB(*Identifier*)<br>...<br>END_SUB]<br>...<br>//END | |
| Description: | Defines a subprogram | |
| Parameters: | Block name | Name of block identifier |
| | Identifier | Name of subprogram |

### Example

```
//B(PROG1)                    ; Block start
SUB(UP1)                      ; Start of subprogram
  ...
  REG[0] = 5                  ; Assign value 5 to register 0
  ...
END_SUB                       ; End of subprogram
SUB(UP2)                      ; Start of subprogram
  IF VAR1.val=="Otto"
    VAR1.val="Hans"
    RETURN
```

```
  ENDIF
  VAR1.val="Otto"
END_SUB                              ;    End of subprogram
//END                                ;    Block end
```

## 2.7.3 Subprogram call (CALL)

### Description

The CALL function can be used to call a loaded subprogram from any point in a method. Subprogram nesting is supported, i.e., you can call a subprogram from another subprogram.

### Programming

| | | |
|---|---|---|
| Syntax: | CALL*("Identifier")* | |
| Description: | Subroutine call | |
| Parameters: | Identifier | Name of subprogram |

### Example

```
//M(SCREEN FORM1)
VAR1 = ...
VAR2 = ...
LOAD
  ...
  LB("PROG1")                            ;    Load block
  ...
END_LOAD
CHANGE()
  ...
  CALL("UP1")                            ;    Call subroutine and execute
  ...
END_CHANGE
...
//END
```

## 2.7.4 Check Variable (CVAR)

### Description

You can use the CVAR (CheckVariable) function to run a scan to ascertain whether all or only certain variables or Help variables in a screen form are error-free.

It may be useful to check if variables contain a valid value before an NC code with the GC function.

A variable is error-free if the state of the variable Identifier.vld = 1.

### Programming

| | | | |
|---|---|---|---|
| Syntax: | CVAR(*VarN*) | | |
| Description: | Checks variables for valid content | | |
| Parameters: | VarN | | List of variables to be checked.<br>Up to 29 variables, each separated by a comma, can be checked. A character length of 500 must not be exceeded.<br>The result of the scan can be: |
| | | 1 = | TRUE (all variables have valid content) |
| | | 0 = | FALSE (at least one variable has invalid content) |

### Example

| | | |
|---|---|---|
| `IF CVAR == TRUE` | ; | Check all variables |
| `  VS8.SE = 1` | ; | If all variables are error-free, softkey VS8 is visible |
| `ELSE` | | |
| `  VS8.SE = 2` | ; | If a variable has an invalid value, softkey VS8 is disabled |
| `ENDIF` | | |
| | | |
| `IF CVAR(”VAR1”, ”VAR2”) == TRUE` | | |
| | ; | Check variables VAR1 and VAR2 |
| `  DLGL ("VAR1 and VAR2 are OK")` | | |
| | ; | If the values of VAR1 and VAR2 are error-free, "VAR1 and VAR2 are OK" appears in the dialog line |
| `ELSE` | | |
| `  DLGL ("VAR1 and VAR2 are not OK")` | | |
| | ; | If the values of VAR1 and VAR2 are invalid, "VAR1 and VAR2 are not OK" appears in the dialog line |
| `ENDIF` | | |

## 2.7.5 Copy Program (CP)

### Description

The CP (Copy Program) function copies files within the HMI file system or NC file system.

---

**Note**

With HMI Embedded on NCU, copying is only possible within the NC file system.

---

### Programming

| | | |
|---|---|---|
| Syntax: | CP*("Source file", "Target file")* | |
| Description: | Copies a file | |
| Parameters: | Source file | Complete path to the source file |
| | Target file | Complete path data of the target file |

### Example

```
CP("\MPF.DIR\CFI.MPF ", "\spf.dir\cfi.nc")
```

## 2.7.6 Dialog line (DLGL)

### Description

It is possible to configure short texts (messages or input tips) for output in the dialog line of the dialog in response to certain situations.

Possible number of characters in the default font size:

- HMI Embedded sl: approx. 50
- HMI Advanced: approx. 100

### Programming

| | | |
|---|---|---|
| Syntax: | DLGL*("String*") | |
| Description: | Outputs text in the dialog line | |
| Parameters: | String | Text, which is displayed in the dialog line |

### Example

```
IF Var1 > Var2
  DLGL("Value too large!")       ; The text "Value too large!" appears in the dialog line
                                   if variable1 > variable2.
ENDIF
```

## 2.7.7 Delete Program (DP)

### Description

The DP (Delete Program) function deletes a file from the passive HMI or active NC file system.

### Programming

| | | |
|---|---|---|
| Syntax: | DP*("File*") | |
| Description: | Delete file | |
| Parameters: | File | Complete path name of file to be deleted |

### Example

```
DP("\MPF.DIR\CFI.MPF")
```

## 2.7.8 Evaluate (EVAL)

### Description

The EVAL function evaluates a transferred expression and then executes it. With this function, expressions can be programmed during runtime. This can be useful, for example, for indexed access operations to variables.

### Programming

| | | |
|---|---|---|
| Syntax: | EVAL*(exp)* | |
| Description: | Evaluates an expression | |
| Parameters: | exp | Logic expression |

### Example

```
VAR1=(S)
VAR2=(S)
VAR3=(S)
VAR4=(S)
CHANGE()
  REG[7] = EVAL("VAR"<<REG[5])        ; The expression in parentheses produces VAR3
                                        if the value of REG[5] is equal to 3. The value
                                        of VAR3 is, therefore, assigned to REG[7].

  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
```

## 2.7.9 Execute (EXE)

### Description

The EXE function can be used on HMI Advanced to call a program created as an application with the HMI Advanced OEM package or to start the Free Contour Programming application.

With HMI Embedded sl, EXE can only be used to start the Free Contour Programming application.

---

**Note**

The EXE function is only available within the parts program editor. In order to start the program, its task index must be entered in the application file under [CHILDS] as it appears in the REGIE.INI file.

---

### Programming

| | | |
|---|---|---|
| Syntax: | EXE*(Program name)* ; HMI Advanced | |
| | EXE*(GPROC)* ; HMI Embedded sl | |
| Description: | Executes program | |
| Parameters: | Program name | Name of the program that is to be executed |

### Example

```
PRESS(VS3)
  EXE(GPROC)                ;    Start GPROC.EXE (free contour programming)
END_PRESS
```

## 2.7.10 Exist Program (EP)

### Description

The EP (Exist Program) function checks whether a particular NC program is stored on the specified path in the NC or HMI file system.

### Programming

| | | |
|---|---|---|
| Syntax: | EP*("File*") | |
| Description: | Checks the existence of the NC program | |
| Parameters: | File | Complete path to the file in the NC or HMI file system |
| Return Value: | | Name of a variable to which the result of the scan should be assigned. The result of the scan can be:<br>• M = File is stored on HMI<br>• N = file is stored on NC<br>• Blank string = The file neither exists on the HMI nor on the NC |

### Example

| | |
|---|---|
| `EP("\MPF.DIR\CFI.MPF", VAR1)` | ; Check whether file CFI.MPF exists in the HMI file system. |

```
IF VAR1 == "M"
  DLGL("File is located in the HMI file system")
ELSE
  IF VAR1 == "N"
    DLGL("File is located in the NC file directory")
  ELSE
    DLGL("File is located neither in the HMI nor in the NC file
directory")
  ENDIF
ENDIF
```

## 2.7.11 Exit dialog (EXIT)

### Description

The EXIT function is used to exit a dialog and return to the master dialog. If no master dialog is found, you will exit the newly configured user interfaces and return to the standard application.

### Programming (without parameters)

| | |
|---|---|
| Syntax: | EXIT |
| Description: | Exits a dialog |
| Parameters: | - None - |

### Example

```
PRESS(HS1)
  EXIT
END_PRESS
```

## Description

If the current dialog has been called with a transfer variable, the value of the variables can be changed and transferred to the output dialog.

The variable values are each assigned to the variables transferred from the output dialog to the subsequent dialog using the "LM" function. Up to 20 variable values, each separated by a comma, can be transferred.

---

**Note**

The sequence of variables or variable values must be the same as the sequence of transfer values programmed for the LM function to preclude assignment errors. Any unspecified variable values will not be changed when the transfer is made. The modified transfer variables are immediately valid in the output dialog on execution of the LM function.

---

## Programming with a transfer variable

| | | |
|---|---|---|
| Syntax: | EXIT*[(VARx)]* | |
| Description: | Exits dialog and transfers one or more variables | |
| Parameters: | VARx | Label variables |

## Example

```
//M(Screen form1)
...
PRESS(HS1)
  LM("SCREEN FORM2","CFI.COM",1, POSX, POSY, DIAMETER)
                                   ; Interrupt screen form1 and open screen form2. Transfer
                                     variables POSX, POSY and DIAMETER in doing this.
  DLGL("Screen form2 ended")       ; On returning from screen form2, the following text appears
                                     in the dialog line of screen form 1: Screen form2 ended.
END_PRESS
...
//END

//M(Screen form2)
...
PRESS(HS1)
 EXIT(5, , CALCULATED_DIAMETER)
```

```
                                          ; Exit screen form2 and return to screen form1 in the line
                                            after LM. In doing this, assign the value 5 to the variable
                                            POSX and the value of the CALCULATED_DIAMETER
                                            variable to the DIAMETER variable. The variable POSY
                                            retains its current value.
END_PRESS
...
//END
```

## 2.7.12 Exit Loading Softkey (EXITLS)

### Description

You can use the EXITLS function to exit the current user interface and load a defined softkey menu.

### Programming

| | | |
|---|---|---|
| Syntax: | EXITLS*("Softkey menu"[, "Path*"]) | |
| Description: | Exits dialog and loads a softkey menu | |
| Parameters: | Softkey menu | Name of the softkey menu to be loaded |
| | Path name | Directory path of the softkey menu to be loaded |

### Example

```
PRESS(HS1)
    EXITLS( "Menu1", "AEDITOR.COM" )
END_PRESS
```

### 2.7.13 Generate code (GC)

#### Description

The GC (Generate Code) function generates NC code from the OUTPUT method.

#### Programming

| | | | |
|---|---|---|---|
| Syntax: | GC("*Identifier"[,"Target file"][,Opt],[Append])* | | |
| Description: | Generates an NC code | | |
| Parameters: | Identifier | | Name of OUTPUT block from which code is generated |
| | Target file | | Path name of target file for HMI or NC file system |
| | | | If the target file is not specified (only possible within programming support system), the code will be written to the location of the cursor within the file that is currently open. |
| | Opt | | Option for generating comments |
| | | 0: | (Default setting) Generate code with comment for the purpose of recompilability. |
| | | 1: | Do not create comments in the generated code. |
| | | | **Note:** This code cannot be recompiled. |
| | Append | | This parameter is only relevant if a target file is specified. |
| | | 0: | (Default setting) If the file already exists, the old content is deleted. |
| | | 1: | If the file already exists, the new code is written at the start of the file. |
| | | 2: | If the file already exists, the new code is written at the end of the file. |

#### Example

```
//M(TestGC/"Code generation:")
DEF VAR1 = (R//1)
DEF VAR2 = (R//2)
DEF D_NAME
LOAD
  VAR1 = 123
  VAR2 = -6
END_LOAD
OUTPUT(CODE1)
  "Cycle123(" VAR1 "," VAR2 ")"
  "M30"
```

```
//M(TestGC/"Code generation:")
END_OUTPUT

PRESS(VS1)
  D_NAME = "MPF.DIR\MESSEN.MPF"
  GC("CODE1",D_NAME)            ; Write code from OUTPUT method to file
                                  \MPF.DIR\MESSEN.MPF:
                                  Cycle123(123, -6)
                                  M30
END_PRESS
```

## Recompile

- No entry for target file:

  The GC function can only be used in the Programming Support system and writes the NC code to the file currently open in the Editor. Recompilation of the NC code is possible. If the GC function is configured without specification of a target file under "Expand user interface", an error message is output when it is executed.

- Entry for target file:

  The code generated from the OUTPUT block is transferred to the target file. If the target file does not already exist, it is set up in the NC file system. If the target file is stored in the HMI file system, it is stored on the hard disk (HMI Advanced only). User comment lines (information required to recompile code) are not set up, i.e., the code cannot be recompiled.

## Special considerations for target file specification

In principle, there are three different ways of specifying a target file:

- NC notation: /_N_MPF_DIR/_N_MY_FILE_MPF

  Only possible with HMI Embedded sl.

  The file is created in the MPF directory on the NC.

- DH notation: /MPF.DIR\MY_FILE.MPF

  Possible with both HMI Advanced and HMI Embedded sl

  In the case of HMI Embedded sl, the target file specification is converted into NC notation and the file is created on the NCU.

  In the case of HMI Advanced or HMI Embedded WIN32, the file is created in the data management path.

- DOS notation: d:\abc\my_file.txt or \\RemoteRechner\files\my_file.txt

  Possible with both HMI Advanced and HMI Embedded sl

  The file is written to the specified directory on the hard disk or on the specified PC, provided that the directory is available on the hard disk or on a remote PC.

  In the case of HMI Embedded sl this notation can only be used to write to the RAM drive or to a networked PC provided that a network connection has been configured.

---

**Note**

Invalid variables generate a blank string in generated NC code and an error message in the log book when they are read.

---

### Special features of recompilation

The GC function cannot be called in sub-dialogs because variables originating from master dialogs can be used in sub-dialogs. These variables would not, however, be available in response to a direct call.

When generated code is processed manually with the Editor, the number of characters for values created by the code generation program must not be changed. Changing these values would make it impossible to recompile the code.

Remedy:

1. Recompile
2. Make change using the configured dialog. (e. g., 99 → 101)
3. GC

### See also

Recompile (Page 106)

## 2.7.14 Load Array (LA)

### Description

The LA (Load Array) function can be used to load an array from another file.

### Programming

| | | |
|---|---|---|
| Syntax: | LA*(Identifier [, File])* | |
| Description: | Loads array from file | |
| Parameters: | Identifier | Name of array to be loaded |
| | File | File in which the array is defined |

**Note**

If an array in the current configuration file must be replaced by an array from another configuration file, then both arrays must have the same name.

## Example

| | |
|---|---|
| | ; Extract from file maske.com |
| `DEF VAR2 = (S/*ARR5/"Out"/,"Toggle field")` | |
| `PRESS(HS5)` | |
| `  LA("ARR5","arrayext.com")` | ; Load array ARR5 from file arrayext.com |
| `  VAR2 = ARR5[0]` | ; "Above"/"Below"/"Right"/"Left" appears in the VAR2 toggle field<br>instead of "Out/In" |
| `END_PRESS` | |
| `//A(ARR5)` | |
| `("Out"/"In")` | |
| `//END` | |
| | ; Extract from file arrayext.com |
| `//A(ARR5)` | |
| `("Above"/"Below"/"Right"/"Left")` | |
| `//END` | |

**Note**

Please note that a valid value must be assigned to a variable after the LA function has been used to assign another array to the toggle field of the variable.

## 2.7.15 Load Block (LB)

### Description

The LB (Load Block) function can be used to load blocks containing subprograms during runtime. LB should be configured in a LOAD method so that the loaded subprograms can be called at any time.

---

**Note**

Subprograms can also be defined directly in a dialog so that they do not have to be loaded.

---

### Programming

| | | |
|---|---|---|
| Syntax: | LB*("Block name"[,"File"])* | |
| Description: | Loads subprogram during runtime | |
| Parameters: | Block name | Name of block identifier |
| | File | Path name of configuration file<br>Default setting = Current configuration file |

### Example

| | |
|---|---|
| `LOAD` | |
| `  LB("PROG1")` | ; Block "PROG1" is searched for in the current configuration file and then loaded. |
| `  LB("PROG2","XY.COM")` | ; Block "PROG2" is searched for in the configuration file XY.COM and then loaded. |
| `END_LOAD` | |

## 2.7.16 Load Mask (LM)

### description.

The LM function can be used to load a new dialog.

### Master dialog/Sub-dialog

A dialog, which calls another dialog, but is not ended itself, is referred to as a master dialog. A dialog that is called by a master dialog is referred to as a sub-dialog.

### Programming

| | | | |
|---|---|---|---|
| Syntax: | LM*("Identifier"[,"File"] [,MSx [, VARx] ] )* | | |
| Description: | Loads dialog | | |
| Parameters: | Identifier | | Name of the dialog to be loaded |
| | File | | Path name (HMI file system or NC file system) of the configuration file, default setting:<br>Current configuration file |
| | MSx | | Mode of dialog change |
| | | 0: | (Default setting) The current dialog disappears; the new dialog is loaded and displayed. EXIT will send you back to the standard application. You can use the MSx parameter to determine whether or not the current dialog should be terminated when changing dialogs.<br>If the current dialog is retained, variables can be transferred to the new dialog.<br>The advantage of the MSx parameter is that the dialogs do not always need to be reinitialized when they are changed; instead, the data and layout of the current dialog are retained and data transfer is made easier. |
| | | 1: | The current master dialog is interrupted when the LM function is initiated; the new sub-dialog is loaded and displayed. EXIT will end the sub-dialog and return to the point at which the master dialog was interrupted.<br>In the master dialog, the UNLOAD block is not processed during the interruption. |
| | VARx | | Requirement: MS1<br>List of variables, which can be transferred from the master dialog to the sub-dialog. Up to 20 variables, each separated by a comma, can be transferred. |

**Note**

Parameter VARx transfers only the value of the variable in each case, i.e., variables can be read and written in the sub-dialog, but are not visible in it. Variables can be returned from the sub-dialog to the master dialog by means of the EXIT function.

## Example

```
PRESS(HS1)
  LM("SCREEN FORM2","CFI.COM",1, POSX, POSY, DIAMETER)
                                    ; Interrupt screen form1 and open screen form2: Variables
                                      POSX, POSY and DIAMETER are transferred in doing this.
  DLGL("Screen form2 ended")        ; On returning from screen form2, the following text appears
                                      in the dialog line of screen form 1: Screen form2 ended.
END_PRESS
```

## 2.7.17 Load Softkey (LS)

### Description

The LS function can be used to display another softkey menu.

### Programming

| | | | |
|---|---|---|---|
| Syntax: | LS*("Identifier"[, "File"][, Merge])* | | |
| Description: | Displays softkey menu | | |
| Parameters: | Identifier | | Name of softkey menu |
| | File | | Path (HMI file system or NC file system) to the configuration file<br>Default: Current configuration file |
| | Merge | | |
| | | 0: | All existing softkeys are deleted; the newly configured softkeys are entered. |
| | | 1: | (Default setting) Only the newly configured softkeys overwrite the existing softkeys. The other softkeys (= standard softkeys from HMI or ShopMill/ShopTurn) remain, and retain their functionality and text. |

## Example

| | |
|---|---|
| `PRESS(HS4)` | |
| `  LS("Menu2",,0)` | ; Menu2 overwrites the existing menu. All existing softkeys are deleted |
| `END_PRESS` | |

---

**Notice**

As long as the interpreter has not displayed a dialog, i.e., no LM function has yet been processed, only one LS or one LM command, but no other action, can be configured in the PRESS method of the definition block for the start softkey and the softkey menu.

The LS and LM functions may only be called within a softkey PRESS block and will not react if navigation keys are pressed (PU, PD, SL, SR, SU, SD).

---

## See also

Functions for start softkeys (Page 69)

### 2.7.18 Passivate Program (PP)

#### Description

The PP (Passivate Program) function transfers a file from the active file system on the NC to the passive file system on HMI Advanced. Once the PP function has been executed, the file is no longer present in the active file system of the NC. With HMI Embedded sl, this function has the same effect as Delete enable.

#### Programming

| | | |
|---|---|---|
| Syntax: | **PP***("File*") | |
| Description: | Transfers a file from the active NC file system to the passive HMI Advanced file system. | |
| Parameters: | File | Complete path name of NC file to be transferred |

#### Example

```
PP("\MPF.DIR\MESSEN.MPF")
```

### 2.7.19 Read NC/PLC (RNP), Write NC/PLC (WNP)

#### Description

The RNP (Read NC PLC) command can be used to read NC or PLC variables or machine data.

#### Programming

| | | |
|---|---|---|
| Syntax: | **RNP** ("*System or user variable", value*) | |
| Description: | Reads NC or PLC variable or machine data | |
| Parameters: | System or user variable | Name of NC or PLC variable |
| | Value | Value that is to be written to the system or user variable.<br>If the value is a String type, it must be written in double quotation marks. |

## Example

| | | |
|---|---|---|
| `VAR2=RNP("$AA_IN[2]")` | ; | Read NC variable |

## Description

The WNP (Write NC PLC) command can be used to write NC or PLC variables or machine data.

NC/PLC variables are accessed anew every time the WNP function is executed, i.e., NC/PLC access is always executed in a CHANGE method. It is advisable to use this option in cases where a system or user variable changes value frequently. If an NC/PLC variable is to be accessed only once, then it must be configured in a LOAD or UNLOAD method.

## Programming

| | | |
|---|---|---|
| Syntax: | WNP*("System or user variable", value)* | |
| Description: | Writes NC or PLC variable or machine data | |
| Parameters: | System or user variable | Name of NC or PLC variable |
| | Value | Value that is to be written to the system or user variable.<br>If the value is a String type, it must be written in double quotation marks. |

## Example

| | | |
|---|---|---|
| `WNP("DB20.DBB1",1)` | ; | Write PLC variable |

## 2.7.20 Multiple Read NC PLC (MRNP)

### Description

This MRNP command can be used to transfer several system or OPI variables in a single register access. This access method is significantly faster than reading via individual access attempts. The system or OPI variables must be included on an MRNP command of the same area.

The areas of the system or OPI variables are organized as follows:

- General NC data ($MN..., $SN.., /nck/...)
- Channel-specific NC data ($MC..., $SC.., /channel/...)
- PLC data (DB..., MB.., /plc/...)
- Axis-specific NC data on the same axis ($MA..., $SA..)

### Programming

| | |
|---|---|
| Syntax: | MRNP*(Variable name 1*Variable name 2[* ...], Register index)* |
| Description: | Reads several variables |
| Parameters: | In the variable names, "*" is the separator. The values are transferred to register REG[Register index] and those following in the order that the variable names appear in the command.<br>The following therefore applies:<br>The value of the first variable is located in REG[Register index].<br>The value of the second variable is located in REG [Register index + 1], etc. |

**Notice**

It should be noted that the number of registers is restricted and the list of variables cannot exceed 500 lines.

### Example

| | | |
|---|---|---|
| `MRNP("$R[0]*$R[1]*$R[2]*$R[3]",1)` | ; | The values of variables $R[0] to $R[3] are written to REG[1] to REG[4]. |

**Reading display machine data:**

Display machine data can be read with RNP ($MM...) within the LOAD block.

General read/write access to display machine data is not possible using the "Expand user interface" function.

---

**Note**

User variables may not have the same names as system or PLC variables.

---

## NC variable

All machine data, setting data and R parameters are available, but only certain system variables (see list in Appendix). In HMI Advanced, you can find the accessible system variables under operating area "Parameters"/"System variables"/"Edit view"/"Insert variable".

All global and channel-specific user variables (GUDs) can be accessed. but local and program-global user variables cannot be processed.

| Machine data | |
|---|---|
| Global machine data | $MN_... |
| Axis-specific machine data | $MA_... |
| Channel-specific machine data | $MC_... |

| Setting data | |
|---|---|
| Global setting data | $SN_... |
| Axis-specific setting data | $SA_... |
| Channel-specific setting data | $SC_... |

| System variables | |
|---|---|
| R parameter 1 | $R[1] |

## PLC variable

All PLC data are available.

| PLC data | |
|---|---|
| Byte y bit z of data block x | DBx.DBXy.z |
| Byte y of data block x | DBx.DBBy |
| Word y of data block x | DBx.DBWy |
| Double word y v. of data block x | DBx.DBDy |
| Real y of data block x | DBx.DBRy |

| PLC data | |
|---|---|
| Flag byte x bit y | Mx.y |
| Flag byte x | MBx |
| Flag word x | MWx |
| Flag double word x | MDx |
| Input byte x bit y | Ix.y or Ex.y |
| Input byte x | IBx or EBx |
| Input word x | IWx or EWx |
| Input double word x | IDx or EDx |
| Output byte x bit y | Qx.y or Ax.y |
| Output byte x | QBx or ABx |
| Output word x | QWx or AWx |
| Output double word x | QDx or ADx |
| String y with length z from data block x | DBx.DBSy.z |

## 2.7.21 REFRESH

### Description

The REFRESH function can be called in all blocks. It has no parameters.

Method of operation:

- All active variable content (input/output fields) in the display range are output again with the background and foreground.
- All active and visible short description texts, graphic texts and unit texts are output again without clearing the text background first.

### Programming

| | |
|---|---|
| Syntax: | **REFRESH** |
| Description: | Updates content of input/output fields and output of text |
| Parameters: | - None - |

### 2.7.22 Register (REG)

#### Register description

Registers are needed in order to exchange data between different dialogs. Registers are assigned to each dialog. These are created when the first dialog is loaded and assigned the value 0 or a blank string.

---

**Note**

Registers may not be used directly in OUTPUT blocks for generating NC code.

---

#### Programming

| | | | |
|---|---|---|---|
| Syntax: | REG*[x]* | | |
| Description: | Defines register | | |
| Parameters: | x | Register index with x = 0...19;<br>Type: REAL or STRING = VARIANT<br>Registers with x ≥ 20 have already been assigned by Siemens. | |

#### Description of register value

The assignment of values to registers is configured in a method.

---

**Note**

If a new dialog is generated from an existing dialog by means of the LM function, register content is automatically transferred to the new dialog at the same time and is available for further calculations in the second dialog.

---

#### Programming

| | | |
|---|---|---|
| Syntax: | *Identifier*.val = *Register value*<br>or<br>*Identifier* = *Register value* | |
| Description: | | |
| Parameters: | Identifier | Name of register |
| | Register value | Value of register |

## Example

```
UNLOAD
  REG[0] = VAR1                 ;  Assign value of variable 1 to register 0
END_UNLOAD

UNLOAD
  REG[9].VAL = 84               ;  Assign value 84 to register 9
END_UNLOAD
                                ;  These registers can then be assigned to local
                                   variables again in a method in the next dialog.

LOAD
  VAR2 = REG[0]
END_LOAD
```

## Description of register status

The Status property can be used to scan a register for valid content.

One possible use for the register scan function is to ensure that a value is written to a register only if the relevant dialog is a "master dialog".

## Programming

| | | |
|---|---|---|
| Syntax: | *Identifier*.vld | |
| Description: | Status is a read-only property. | |
| Parameters: | Identifier | Name of register |
| Return Value: | | The result of the scan can be: |
| | FALSE = | invalid value |
| | TRUE = | valid value |

## Example

```
IF REG[15].VLD == FALSE         ;  Scan validity of register value
  REG[15] = 84
ENDIF
VAR1 = REG[9].VLD               ;  Assign the value of the REG[9] status request
                                   to Var1.
```

## 2.7.23 RETURN

### Description

The RETURN function can be used to prematurely terminate execution of the current subprogram and to return to the branch point of the last CALL command.

If no RETURN command is configured in the subprogram, the subprogram will run to the end before returning to the branch point.

### Programming

| | |
|---|---|
| Syntax: | RETURN |
| Description: | Returns to the branch point |
| Parameters: | - None - |

### Example

```
//B(PROG1)                ; Block start
SUB(UP2)                  ; Start of subprogram
  IF VAR1.val=="Otto"
    VAR1.val="Hans"
    RETURN                ; If the variable value = Otto, the value "Hans" is assigned
                          ; to the variable, and the subprogram ends at this point.
  ENDIF
  VAR1.val="Otto"         ; If the variable value ≠ Otto, the value "Otto" is assigned
                          ; to the variable.
END_SUB                   ; End of subroutine
//END                     ; Block end
```

## 2.7.24 Recompile

### Description

In the programming support system, it is possible to recompile NC code that has been generated with the GC function and to display the variable values in the input/output field of the associated entry dialog again.

## Programming

Variables from the NC code are transferred to the dialog. At the same time, the variable values from the NC code are compared with the calculated variable values from the configuration file. If the values do not coincide, an error message is written to the log book because values have been changed during NC code generation.

If the NC code contains the same variable several times, it is evaluated at the point where it last occurs during recompilation. A warning is also written to the log book.

Variables not utilized in NC code during code generation are stored as user comment.
The term "user comment" refers to all information required to recompile codes.
User comment must not be altered.

---

**Note**

The block consisting of NC code and user comment can be recompiled only if it starts at the beginning of a line.

---

**Examples:**

The programm contains the following NC code:

```
DEF VAR1=(I//101)
OUTPUT(CODE1)
  "X" VAR1 " Y200"
  "X" VAR1 " Y0"
END_OUTPUT
```

The following code is then stored in the parts program:

```
;NCG#TestGC#\cus.dir\aeditor.com#CODE1#1#3#
X101 Y200
X101 Y0
;#END#
```

The Editor reads the following during recompilation:

```
X101 Y200
X222 Y0                    ; The value for X has been changed in the parts program
                             (X101 → X222)
```

The following value is displayed for VAR1 in the input dialog: `VAR1 = 222`

## See also

Generate code (GC) (Page 91)

## 2.7.25 Search Forward, Search Backward (SF, SB)

### Description

The **SF, SB (Search Forward, Search Backward)** function is used to search for a string from the current cursor position in the NC program currently selected in the Editor and to output its value.

### Programming

| | | |
|---|---|---|
| Syntax: | SF*("String")* | |
| Identifiers: | Search Forward: Search forward from the current cursor position | |
| Syntax: | SB*("String")* | |
| Identifiers: | Search Backward: Search backward from the current cursor position | |
| Parameters: | String | Text to be found |

### Rules governing text search

- A blank must be inserted before and after the search concept unit, consisting of search string and its value, in the currently selected NC program.
- The system does not search for concepts within comment text or other strings.
- The value to be output must be a numerical expression. Expressions in the form of "X1=4+5" are not recognized.
- The system recognizes hexadecimal constants in the form of X1='HFFFF', binary constants in the form of X1='B10010' and exponential components in the form of X1='-.5EX-4'.
- The value of a string can be output if it contains the following between string and value:
  – Nothing
  – Blanks
  – Equality sign

### Example

The following notations are possible:

```
X100 Y200                    ;   The variable Abc is assigned the value 200
Abc = SB("Y")
X100 Y 200                   ;   The variable Abc is assigned the value 200
Abc = SB("Y")
X100 Y=200                   ;   The variable Abc is assigned the value 200
Abc = SB("Y")
```

### 2.7.26 Select Program (SP)

#### Description

The SP (Select Program) function selects a file in the active NC file system for execution, i.e., the file must be loaded into the NC beforehand.

#### Programming

| | | |
|---|---|---|
| Syntax: | SP*("File")* | |
| Identifiers: | Select Program | |
| Parameters: | "File" | Complete path name of NC file |

#### Example

```
//M(TestGC/"Code generation:")
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)                        ;  Write code from the OUTPUT
                                              method to file
                                              \MPF.DIR\MESSEN.MPF
END_PRESS
PRESS(HS8)
 AP("\MPF.DIR\MESSEN.MPF")                 ;  Load file into NC
 SP("\MPF.DIR\MESSEN.MPF")                 ;  Select file
END_PRESS
```

## 2.7.27 STRING functions

### Overview

The following functions enable strings to be processed:

- Determine length of string
- Find a character in a string
- Extract substring from left
- Extract substring from right
- Extract substring from mid-string
- Replace substring

### LEN function: Length of a string

| | | |
|---|---|---|
| Syntax: | LEN*(string \| varname)* | |
| Description: | Determines the number of characters in a string | |
| Parameters: | string | Every valid string expression. NULL is output if string is blank. |
| | varname | Any valid declared variable name |
| | Only one of the two parameters is allowed. | |

### Example

```
DEF VAR01
DEF VAR02

LOAD
  VAR01="HALLO"
  VAR02=LEN(VAR01)                   ;     Result = 5
END_LOAD
```

### INSTR function: Search for character in string

| | | |
|---|---|---|
| Syntax: | INSTR*(Start, String1, String2 [,Direction])* | |
| Description: | Searches for characters | |
| Parameters: | Start | Starting position for searching from string1 into string2. Enter 0 to start searching at the beginning of string2. |
| | String1 | Character that is being searched for. |
| | String2 | Chain of characters in which the search is being made |

| | |
|---|---|
| Direction (optional) | Direction in which the search is being made<br>0: From left to right (default setting)<br>1: From right to left |

0 is returned if string1 does not occur in string2.

## Example

```
DEF VAR01
DEF VAR02

LOAD
  VAR01="HELLO/WORLD"
  VAR02=INST(1,"/",VAR01)           ;    Result = 6
END_LOAD
```

## LEFT Function: String from left

| | | |
|---|---|---|
| Syntax: | **LEFT***(string, length)* | |
| Description: | LEFT returns a string containing the specified number of characters starting from the left-hand side of a string. | |
| Parameters: | string | Character string or variable with the string to be processed |
| | length | Number of characters that are to be read out |

## Example

```
DEF VAR01
DEF VAR02
LOAD
 VAR01="HELLO/WORLD"
  VAR02=LEFT(VAR01,5)                 ;  Result = "HELLO"
END_LOAD
```

## RIGHT function: String from right

| | | |
|---|---|---|
| Syntax: | **RIGHT***(string, length)* | |
| Description: | RIGHT returns a string containing the specified number of characters starting from the right-hand side of a string. | |
| Parameters: | string | Character string or variable with the string to be processed |
| | length | Number of characters that are to be read out |

## Example

```
DEF VAR01
DEF VAR02
LOAD
 VAR01="HELLO/WORLD"
  VAR02=LEFT(VAR01,4)                    ; Result = "WORLD"
END_LOAD
```

## MIDS function: String from mid-string

| | | |
|---|---|---|
| Syntax: | MIDS*(string, start [, length])* | |
| Description: | MIDS returns a string containing the specified number of characters starting at the specified position in the string. | |
| Parameters: | string | Character string or variable with the string to be processed |
| | start | Start from where characters are to be read in the string |
| | length | Number of characters that are to be read out |

## Example

```
DEF VAR01
DEF VAR02
LOAD
 VAR01="HELLO/WORLD"
  VAR02=LEFT(VAR01,4,4)                  ; Result = "LO/W"
END_LOAD
```

## REPLACE Function: Replacing characters

| | | |
|---|---|---|
| Syntax: | REPLACE*( string, FindString, ReplaceString [, start [, count ] ] )* | |
| Description: | The REPLACE function replaces a character/string in a string with another character/string. | |
| Parameters: | string | String in which FindString is to be replaced with ReplaceString. |
| | FindString | String to be replaced |
| | ReplaceString | Replacement string (is used instead of the FindString) |

| | |
|---|---|
| start | Starting position for search and replace operations |
| count | Number of characters that are to be searched from the starting position after the FindString. |

Return Value:

| | |
|---|---|
| string = Blank string | Copy of string |
| FindString = Blank string | Copy of string |
| ReplaceString = Blank string | Copy of string, in which all occurrences of FindString are deleted |
| start > Len(String) | Blank string |
| count = 0 | Copy of string |

## 2.7.28 PI services

### Description

The PI_SERVICE function can be used to start PI Services (Program Invocation Services) from the PLC in the NC area.

### General programming

| | | |
|---|---|---|
| Syntax: | PI_SERVICE (*service, n parameters*) | |
| Description: | Executes PI service | |
| Parameters: | Service | PI service identifier |
| | n parameters | List of n parameters of PI Service. Individual parameters are separated by commas. |

### Example

```
PRESS (HS2)
  PI_SERVICE("_N_CREATO",55)
END_PRESS
PRESS(VS4)
  PI_SERVICE("_N_CRCEDN",17,3)
END_PRESS
```

### Starting OEM services

The PI_START command executes a PI service based on OEM documentation.

### Programming

| | | |
|---|---|---|
| Syntax: | PI_START("*Transfer string*") | |
| Description: | Executes PI service | |
| Parameters: | "Transfer string" | Unlike the OEM documentation, the transfer string should be entered in inverted commas. |

### Example

```
PI_START("/NC,001,_N_LOGOUT")
```

**Note**

Channel-dependent PI Services always refer to the current channel.

PI services of the tool functions (TO area) always refer to the TO area that is assigned to the current channel.

### See also

List of PI services (Page 205)

## 2.7.29 External functions (only HMI Advanced)

### Description

Additional user-specific functions can be used by means of this call. External functions are stored in a DLL file and identified by an entry in the definition lines of the configuration file.

**Note**

An external function must have at least one return parameter.

## Programming

Syntax: **FCT** *Function name = ("File"/Type of return/Types of permanent call parameters/Types of variable call parameters)*

Description: Declares additional, external functions

Parameters:

| | |
|---|---|
| Function name | Name of external function |
| File | Complete path to DLL file |
| Type of return | Data type of return value for function |
| R, I, S, C, B. | Data types of **permanent** call parameters and return values. The data types are separated by commas. |
| Variable or register | Data types of **variable** call parameters |

Syntax: **FCT** *Function name (call parameter)*

| | |
|---|---|
| Call parameter | List of all call parameters. Individual parameters are separated by commas. |

## Example

```
//M(SCREEN FORM1)
DEF VAR1 = (R)
DEF VAR2 = (I)
DEF RET = (I)
FCT InitConnection = ("c:\user\mydll.dll"/I/R,I,S/I,S)
                              ; The external function "InitConnection" is declared. The data type
                                of what has been declared. The data type of the return value
                                is Integer, the data types of the permanent call parameters are
                                Real, Integer and String; the data types of the variable call
                                parameters are Integer and String.
LOAD
  RET = InitConnection(VAR1+SIN(VAR3),13,"Servus",VAR2,
  REG[2])
                              ; The value of the "InitConnection" external function is assigned
                                to RET along with the call parameters VAR1+SIN(VAR3), 13,
                                Servus, VAR2 and REG[2].
END_LOAD
```

## Extract from DLL file

```
void __export WINAPI InitConnection(ExtFctStructPtr FctRet, ExtFctStructPtr
FctPar, char cNrFctPar)
FctRet->value.i   Return value of function
FctPar[0]->value.r   1st parameter (VAR1+SIN(VAR3))
FctPar[1]->value.i   2nd parameter (13)
```

```
FctPar[2]->value.s   3rd parameter ("Servus")
FctPar[4]->value.i   4th parameter (reference to VAR2)
FctPar[5]->value.s   5th parameter (reference to REG[2])
cNrFctPar   Number of parameters (5)
```

## 2.7.30 Programming example

### Programming

```
//S(Start)
HS7=("Example", se1, ac7)
PRESS(HS7)
LM("Screen form4")
END_PRESS
//END
//M(Screen form4/"Example 4: Machine control panel"/"MST.BMP")
DEF byte=(I/0/0/"Input byte, default=0","Byte
no.:",""/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"/280,180,100/380,180,50/0,11)
DEF spin=(IBB//0/"","Spindle
override",""/wr1//"EB0"/20,210,100/130,210,100),
spinstop=(B//0/"","Spindle
stop",""/wr1//"E2.4"/280,210,100/380,210,50/0,11)
DEF custom1=(IBB//0/"","User key
1",""/wr1//"EB6"/20,240,100/130,240,100)
DEF custom2=(IBB//0/"","User name
2",""/wr1//"EB7"/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=("")
VS8=("OK", SE1, AC7)

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

PRESS(VS8)
  EXIT
END_PRESS
```

## Result

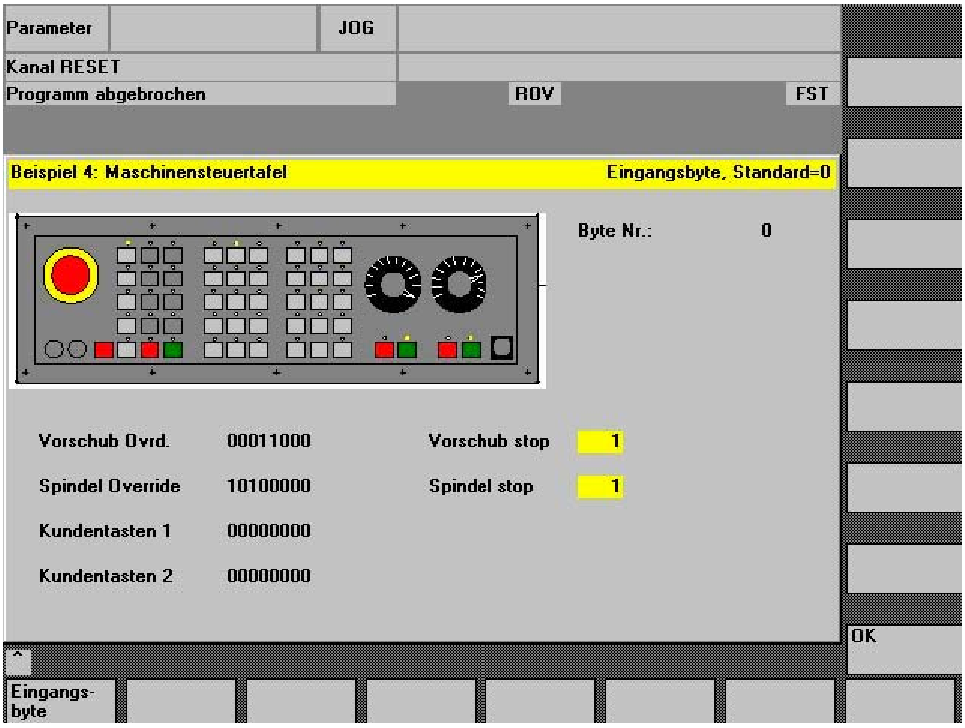



Figure 2-10 Machine control panel

---

**Note**

You will find further examples in the supplied tool box.

---

# 2.8 Operators

## Overview

The following operators can be used when programming:

- Mathematical operators
- Relational operators
- Logic (Boolean) operators
- Bit operators
- Trigonometric functions

## 2.8.1 Mathematical operators

### Overview

| Mathematical operators | Identifier |
|---|---|
| + | Addition |
| - | Subtraction |
| * | Multiplication |
| / | Division |
| MOD | Modulo operation |
| ( ) | Parentheses |
| AND | AND operator |
| OR | OR operator |
| NOT | NOT operator |
| ROUND | Round off numbers with decimal places |

**Example:** `VAR1.VAL = 45 * (4 + 3)`

### ROUND

The ROUND operator is used to round off numbers with up to 12 decimal places during execution of a dialog configuration. The variable fields cannot accept the decimal places in the display.

### Use

ROUND is controlled by the user with two parameters:

```
VAR1 = 5,2328543
VAR2 = ROUND( VAR1, 4 )
```

**Result:** `VAR2 = 5,2339`

VAR1 contains the number to be rounded. The parameter “4” indicates the number of decimal places in the result, which is placed in VAR2.

## Trigonometric functions

| Trigonometric functions | Identifier |
|---|---|
| SIN(x) | Sine of x |
| COS(x) | Cosine of x |
| TAN(x) | Tangent of x |
| ATAN(x, y) | Arc tangent of x/y |
| SQRT(x) | Square root of x |
| ABS(x) | Absolute value of x |
| SDEG(x) | Conversion to degrees |
| SRAD(x) | Conversion to radian |

**Note**

The functions operate with radian measure. The functions SDEG() and SRAD() can be used for conversion.

**Example:** `VAR1.VAL = SQRT(2)`

## Constants

| Constants | |
|---|---|
| PI | 3.14159265358979323846 |
| FALSE | 0 |
| TRUE | 1 |

**Example:** `VAR1.VAL = PI`

## Relational operators

| Relational operators | |
|---|---|
| == | Equal to |
| <> | Not equal to |
| > | Greater than |
| < | Less than |
| >= | Greater than or equal to |
| <= | Less than or equal to |

#### Example

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

### Conditions

The nesting depth is unlimited.

| | |
|---|---|
| Condition with a command: | `IF`<br>`...`<br>`ENDIF` |
| Condition with two commands: | `IF`<br>`...`<br>`ELSE`<br>`...`<br>`ENDIF` |

## 2.8.2 Bit operators

### Overview

| Bit operators | Identifier |
|---|---|
| BOR | Bit-serial OR |
| BXOR | Bit-serial XOR |
| BAND | Bit-serial AND |
| BNOT | Bit-serial NOT |
| SHL | Shift bits to left |
| SHR | Shift bits to right |

### SHL operator

Bits are shifted to the left using the SHL (SHIFT LEFT) operator. You can specify both the value to be shifted and the number of shift increments directly or via a variable. If the limit of the data format is reached, the bits are shifted beyond the limit without displaying an error message.

### Use

| | | |
|---|---|---|
| Syntax: | variable = *value* SHL*increment* | |
| Description: | Shift Left | |
| Parameters: | value | value to be shifted |

| | |
|---|---|
| increment | number of shift increments |

## Example

```
PRESS(VS1)
  VAR01 = 16 SHL 2            ; Result = 64
  VAR02 = VAR02 SHL VAR04     ; Convert content of VAR02 to 32-bit unsigned, and shift
                                content to left by number of bits specified in VAR04.
                                Then convert 32-bit value back to format of variable VAR02.
END_PRESS
```

## SHR operator

Bits are shifted to the RIGHT using the SHR (SHIFT RIGHT) function. You can specify both the value to be shifted and the number of shift increments directly or via a variable. If the limit of the data format is reached, the bits are shifted beyond the limit without displaying an error message.

## Use

| | | |
|---|---|---|
| Syntax: | variable = *value* SHR*increment* | |
| Description: | Shift Right | |
| Parameters: | value | value to be shifted |
| | increment | number of shift increments |

## Example

```
PRESS(VS1)
  VAR01 = 16 SHR 2            ; Result = 4
  VAR02 = VAR02 SHR VAR04     ; Convert content of VAR02 to 32-bit unsigned, and shift
                                content to left by number of bits specified in VAR04.
                                Then convert 32-bit value back to format
                                of variable VAR02.
END_PRESS
```

# 3 Programming support

## 3.1 What does programming support do?

### Overview

The purpose of the programming support system is to assist programmers in writing an NC program using the ASCII Editor. Customized user interfaces can also be created in the Programming support system. These interfaces are configured using the ASCII Editor and the tools provided by the "Expanding the Operator Interface" system.

The system provides the following standard tools for this purpose:

- Cycle support
- Free contour programming
- Contour definition programming
- Recompile
- Simulation

---

**Note**

For the sake of compatibility, the cycles support (//C...) function is still supported by the description language used in earlier versions and not by the syntax of the "Expanding the Operator Interface" system.

---

### Creating new dialogs

New operator interfaces are created with tools provided by the "Expanding the Operator Interface" system. However, there are some differences affecting programming support and these are described in this chapter.

### Configuration file

The definition of new dialogs for programming support is stored in configuration file AEDITOR.COM.

- Newly configured dialogs can be displayed in the Editor main screen through selection of 5 start softkeys (horizontal softkeys 2, 3, 4, 5 and 6).
- The default labels for softkeys 2 to 5 are "Contour", "Drilling", "Milling" and "Turning".

- Horizontal softkeys 14 and 15 (softkeys 6 and 7 in the expanded menu) are assigned the "Gauge turning" and "Gauge milling" functions.

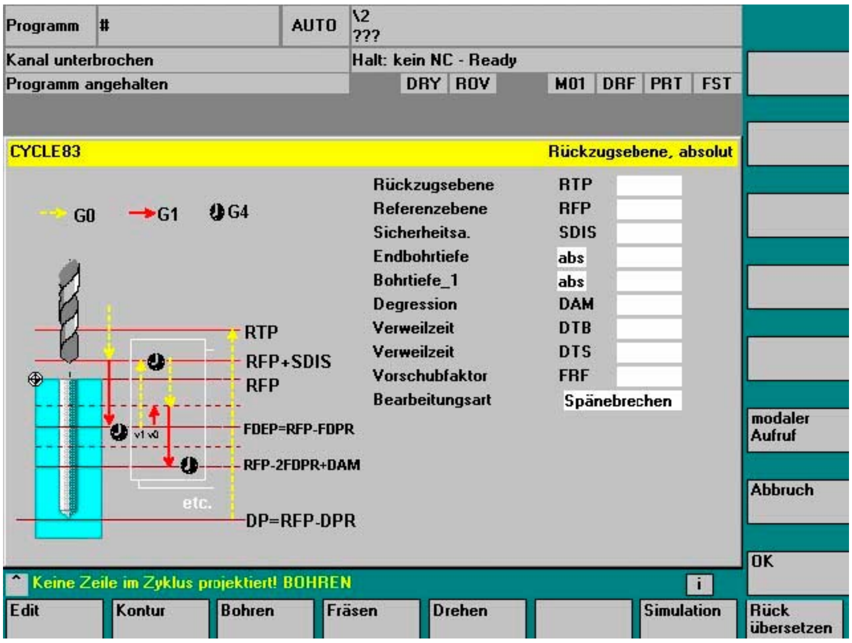


Figure 3-1 View of the default setting

## 3.2 Cycle support

### Use

When using cycle support in the parts programs, additional comment lines are generated in front of and behind the cycle call and used for recompilation. These lines start with ;# ... .

When cycles noted directly using the editor are called (also referred to as "old" cycles), then this information is missing. A description is provided in the following text as to how the information for cycles, contained in the lines in ;# can be provided by configuring INI files. For Siemens cycles, the required configuration files are included in the scope of supply.

This means that after recompilation it will be possible to work with the following cycle calls using the dialog support:

- Directly noted cycle calls
- Siemens cycle calls
- User cycle calls

## Prerequisites

For these cycles, a configuration generated using the "Expand user interface" tools must exist or be generated.

For dialogs generated with the "Expand user interface" function for the purpose of supporting user cycles, the information in the dialog descriptive files can be used for this process.

Cycle-specific settings can be made in the configuration to define whether cycles with ;# lines are to be processed in the parts program or recompiled via the configuration files.

This process can be used to optionally generate the addition of ;# lines required for dialog support in the parts program for directly noted “old" cycles.

## Configuration

The WIZARD.INI file can be stored in the following directories. The directories are browsed in this sequence:

..\user

..\oem

..\hmi_adv

..\mmc2

The following entries refer to cycle packages, e.g.:

```
INI_1=bohren.ini
INI_2=drehen.ini
...
INI_n=paket.ini
```

The information in the [MMC_CycleWizard] section is added so that it is up to date.

Cycle-specific entries in the following format are located in the relevant PAKET.INI files:

```
[cycleName]     ; Name of the cycle as a separate section
Mname=          ;Obligatory   dialog name
Dir=            ;Obligatory   Directory of file with dialog
                              information
Dname=          ;Obligatory   File with dialog information
Output          ;Obligatory   Name of the OUTPUT block
Anzp=           ;Obligatory   Number of parameters (defined
                              variables)
Version=        ; Optional    cycle version, without entry 0
Code_typ=       ; Optional    output with = 0 or without = 1 ;#
                              lines
```

**Example:**

```
[CYCLE83]
Mname=CYC83
Dir=cst.dir
Output=bohren.ini
Anzp=17
Version=3
Code_typ=0
```

### Different cycle versions

A differentiation is made between various versions of cycles with the same name using the version number. Main block/secondary block and block numbers in front of the cycle call are kept unchanged.

Cycle call line in the parts program:

```
/1234 :44 CYCLE94(, , )
```

After expansion by the string and ;# lines:

```
;NCG#CYC94#\CST.DIR\DREHEN.COM#NC1#1#*NCG;*RO*;*HD*
;##########*NCG;*RO*;*HD*
/1234 :44 CYCLE94(,,"",)
;#END#*NCG;*RO*;*HD*
```

### See also

Recompile (Page 106)

Search Forward, Search Backward (SF, SB) (Page 108)

## 3.3 Activating a dialog from the NC program

### Introduction

With HMI Advanced and HMI Embedded sl, dialogs defined by the user can be displayed. The appearance of the dialogs is defined through configuration (modification of COM file in the cycles directory).

The dialog is called and exited by function calls from the parts program. User-configured dialog screens do not modify the HMI system software (user interface). User-defined dialogs cannot be called simultaneously in different channels.

### Command channel

The "Activate dialog from NC Program" function is also referred to as a "command channel".

### Activating the command channel

One possible application of user-defined dialogs is, for example, to assign defined values to particular user variables (GUD) prior to a parts program run.

- Up to 2 channels:

  The "command channel" is activated by default for channels 1 and 2.

- More than 2 channels:

  For HMI Advanced, the "command channel" must be activated (if the Siemens measuring cycles have not yet been installed). This requires a modification to be made to the file F:\MMC2\ COMIC.NSK in the "Startup" operating area:

Select file **F:\MMC2\COMIC.NSK** in the "Startup" operating area using the softkeys "HMI" → "Editor" and attach the following text (after channels 1 and 2):

```
REM CHANNEL
TOPIC(machineswitch) COMIC_START(COMIC001MachineSwitch"...)
[compare the text for channels 1 and 2]
```

When the control is restarted (OFF/ON), the command channel is activated for the corresponding channel.

### Activation with HMI Advanced

Content of the COMIC.NSK file:

```
REM ----- TYPICAL COMIC START
REM CHANNEL 1
TOPIC(machineswitch) COMIC_START("COMIC001MachineSwitch
",/Channel/Configuration/mmcCmd[u1],
/Channel/Configuration/mmcCmdQuit[u1])
REM CHANNEL 2
TOPIC(machineswitch) COMIC_START("COMIC002MachineSwitch
",/Channel/Configuration/mmcCmd[u2],
/Channel/Configuration/mmcCmdQuit[u2])
```

## 3.3.1 Structure of "MMC" instruction

### Programming

| | | | |
|---|---|---|---|
| Syntax | **MMC** ("Operating area, command, Com file, dialog box name, user-data definition file, graphics file, display time or acknowledgment variable, text variables...", "Acknowledgment mode") | | |
| Parameters: | Operating area | Name of softkey used to call the configured user dialogs.<br>Default: CYCLES displayed as "Cycles" on softkey 14 and accessible via the <ETC> key. | |
| | Command | PICTURE_ON | Select screen |
| | | PICTURE_ON | Deselect screen |
| | COM file | Name of the dialog screen file (max. 8 characters, in the user, manufacturer, or standard cycles directory).<br>The dialog display appearance is defined here.<br>The dialog screen is used to display user variables and/or comment texts. | |
| | dialog name | The individual dialogs are selected via the dialog names. | |

| | | |
|---|---|---|
| GUD file | User-data definition file accessed on reading/writing of variables. | |
| Graphics file<br>(HMI Advanced only) | File name of the BMP graphic to be inserted | |
| Acknowledgment variable (HMI Advanced only) | Acknowledgment variable | in acknowledgment mode "A" |
| or display time | Display time | in acknowledgment mode "N" |
| Text variable | Screen header or comment text from a text variable in the COM file. | |
| Acknowledgment mode | "S" for **S**ynchronous<br>"A" for **A**synchronous<br>"N" for **N**o Quit | Acknowledgment via "OK" softkey<br>Acknowledgment via configured softkeys<br>No acknowledgment, but display time |

## Storage structure of graphics

*.bmp files are stored in resolution-specific subdirectories:

- For standard cycles:

| | |
|---|---|
| \CST.DIR\HLP.DIR\640.DIR | For 640 dpi resolution |
| \CST.DIR\HLP.DIR\800.DIR | For 800 dpi resolution |
| \CST.DIR\HLP.DIR\1024.DIR | For 1024 dpi resolution |

- For user cycles:

| | |
|---|---|
| \CUS.DIR\HLP.DIR\640.DIR | For 640 dpi resolution |
| \CUS.DIR\HLP.DIR\800.DIR | For 800 dpi resolution |
| \CUS.DIR\HLP.DIR\1024.DIR | For 1024 dpi resolution |

- For manufacturer cycles:

| | |
|---|---|
| \CMA.DIR\HLP.DIR\640.DIR | For 640 dpi resolution |
| \CMA.DIR\HLP.DIR\800.DIR | For 800 dpi resolution |
| \CMA.DIR\HLP.DIR\1024.DIR | For 1024 dpi resolution |

### 3.3.2 Example of MMC instruction

#### MMC instruction in the parts program

```
MMC("CYCLES,PICTURE_ON,T_SK.COM,PICTURE1,MGUD.DEF,PICTURE3.BMP,TEST_
1,A1","S")
```

| | |
|---|---|
| CYCLES | Operating area |
| PICTURE_ON | Select dialog |
| T_SK.COM | File name in cycle directory |
| PICTURE1 | Name of the dialog |
| MGUD.DEF | User data definition file |
| PICTURE3.BMP | Name of graphics file (HMI Advanced only) |
| TEST_1 | Acknowledgment variable (HMI Advanced only) or display time in "N" mode |
| A1 | Screen header or comment from a text variable (COM file) |
| S | Acknowledgment mode: Synchronized |

#### User variables in definition directory

```
%_N_UGUD_DEF
;$PATH=/_N_DEF_DIR
DEF CHAN REAL TEST_1
```

| | |
|---|---|
| CHAN | Applicable channel |
| REAL | Data type |
| TEST_1 | Name of user variable |

#### Dialog screen file in the cycles directory (*.COM)

```
//C3(Screen2)
R/ 15   75 / 5 /COMMENT, %1 %2 %3/ W,RJ / TEST_1 / ...
```

| | | |
|---|---|---|
| R | Variable Type: Real, Integer or String | |
| 15   75 | Permissible range: 15 to 75 | |
| 5 | Default setting for user variable | |
| COMMENT, %1 %2 %3 | Comment text with optional text variables | |
| W,RJ | Access type: | W = Read and write |
| | | R = Read-only |
| | | W, RJ = Read and write with comment |
| | | J = Right-justified in relation to input/output field |
| | | <without> = Left-justified in relation to input/output field |
| TEST_1 | User variable | |

### Text variable

```
[Text variables]
A1 = Example 2: MMC instruction without acknowledgment
```

| | |
|---|---|
| `A1` | Reference parameters for MMC instruction |
| `Example 2: MMC statement without acknowledgment` | Screen header or comment text |

---

**Note**

For variable names, text variables and cycle names, capital letters must be used.

---

### Configuring softkeys for dialog call

Softkey assignment for MMC command with asynchronous acknowledgment mode.

```
[PICTURE3]
SK1 = END
SK2 = Screen2
```

Softkeys SK0 to SK15 can be configured

## 3.3.3 Example 1: MMC instruction without acknowledgment

### Selecting

```
N10 MMC("CYCLES,PICTURE_ON,T_SK.COM,PICTURE1,GUD4.DEF,,,A1","N")
N20 TEST_1 = 1
N25 G4 F10
N30 MMC("CYCLES,PICTURE_OFF","N")
M30
```

**Parameters:**

| | |
|---|---|
| Dialog file (*.COM) | `//C1(PICTURE1)`<br>`(R///USER VAR TEST_1/W/TEST_1///)` |
| Text variable | `[TEXT VARIABLES]`<br>`A1 =.......Example 1: MMC command without acknowledgment` |

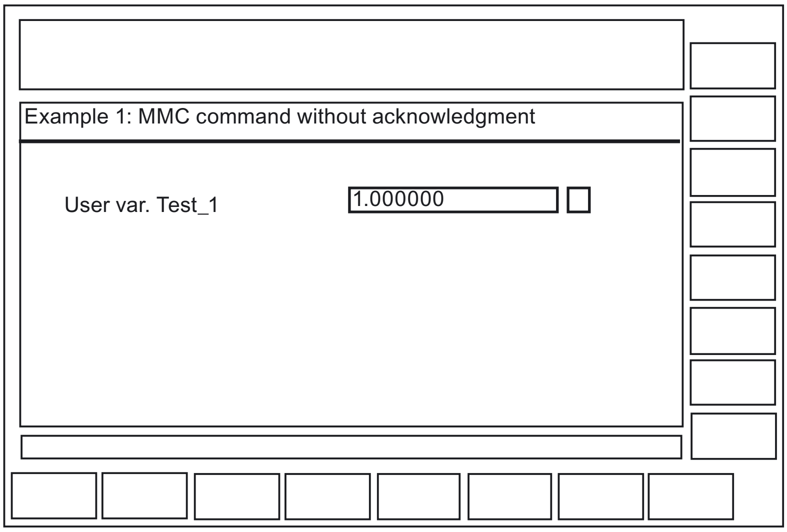



Figure 3-2 Example 1 Without acknowledgment

### Sequence of operations

The user variable TEST_1 from GUD4.DEF with header A1 is displayed for a short time. The hold time comes from parts program block N25.

## 3.3.4 Example 2: Hold time and optional text variables

### Selecting

```
N10
MMC("CYCLES,PICTURE_ON,T_SK.COM,PICTURE6,GUD4.DEF,,10,T1,G1","N")
N15 G4 F15
N30 MMC("CYCLES,PICTURE_OFF","N")
M30
```

**Parameters:**

| | |
|---|---|
| Dialog file (*.COM) | `//C6(screen6)`<br>`(R///USER VAR TEST_1,%1/W/TEST_1///)` |
| Text variable | `[TEXT VARIABLES]`<br>`T1 = Example 2: Dwell time and optional text variables ...`<br>`G1 = Optional text variable` |

The 7th parameter is interpreted as the display time for the mode without acknowledgment (10 seconds). The contents of the table are then deleted. The dialog remains on the screen until PICTURE_OFF is selected. The 8th parameter (T1) is the text variable for the header. If there is no entry, the operating area name, "cycles", is displayed. The parameters 9 to 23 are optional text variables ("G1=optional text variable"). In this COM file, the optional text variables must be preassigned in the [Text variables] section.

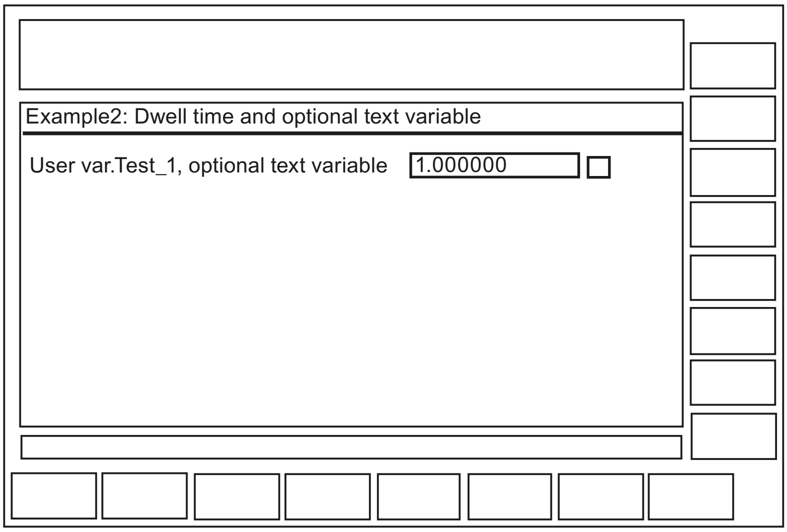



Figure 3-3 Example 2 Dwell time

### Sequence of operations

The comment text from the COM file (USER VAR TEST_1) is expanded in this example at the position of the first wild card (%1) to include the contents of the text variable "G1=optional text variable". By calling the text variables contained in the MMC command (9th to 23rd parameters), messages or names can be "composed" in this way.

## 3.3.5 Example 3: MMC instruction with synchronous acknowledgment mode

### Selecting

```
N15 MMC("CYCLES,PICTURE_ON,T_SK.COM,PICTURE1,GUD4.DEF,,,F1","S")
N18 STOPRE
N20 TEST_1 = 5
N25 MMC("CYCLES,PICTURE_OFF","N")
M30
```

**Parameters:**

| | |
|---|---|
| Dialog file (*.COM) | `//C1(PICTURE1)` `(R///USER VAR TEST_1/W/TEST_1///)` |
| Text variable | `F1 = ...Example 3: MMC command with synchronous acknowledgment mode...` |

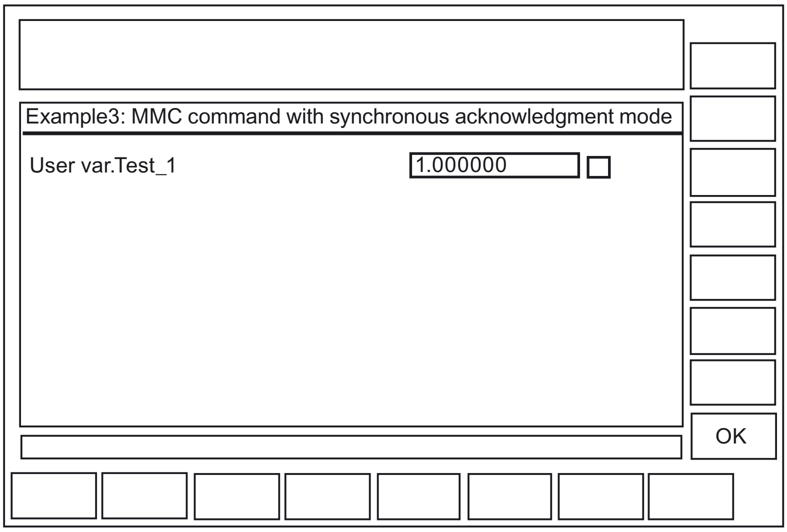



Figure 3-4 Example 3 Synchronous acknowledgment mode

### Sequence of operations

The user variable Test_1 is displayed until the "OK" softkey is pressed. The user variable has also been overwritten with the value 5 here.

Without STOPRE, this assignment would be carried out **before** the keyboard input (after it with STOPRE).

## 3.3.6 Example 4: Positioning of input/output field

### Description

By specifying position parameters in the COM file, you can insert the comment field, or input and output field, at any point in the display area.

### Selecting

```
N15 MMC("CYCLES,PICTURE_ON, T_SK.COM; Screen2,GUD4.DEF,,,C1","S")
N20 TEST_3 = 5
N30 MMC("CYCLES,PICTURE_OFF","N")
N40 M30
```

**Parameters:**

The two parameters each consist of three numerical values; these specify the position and length of the field. The values are specified in Twips, with 15 Twips roughly corresponding to one pixel. The field height is defined as being 250 Twips.

Dialog file (*.COM) `//C2(Screen12)(R///Var.Name/R/TEST_3/6000,2800,8000/200,3000,7500)`

| | | |
|---|---|---|
| `/6000,2800,8000` | | Position of the comment field |
| `/200,3000,7500` | | Position of input/output field |
| First value = 0 | → | Automatic positioning to default setting |
| No details | → | Positioning as on PCU 20 (default setting) |

**Meaning of values:**

`(./6000,2800,8000/....)`

| | |
|---|---|
| `6000` | Distance from left border |
| `2800` | Distance from upper border |
| `8000` | Field length |

## Sequence of operations

It is possible to graphically configure 16 comment and 16 input or output fields. Where there are more than 16 fields, they can be controlled via a scroll bar.

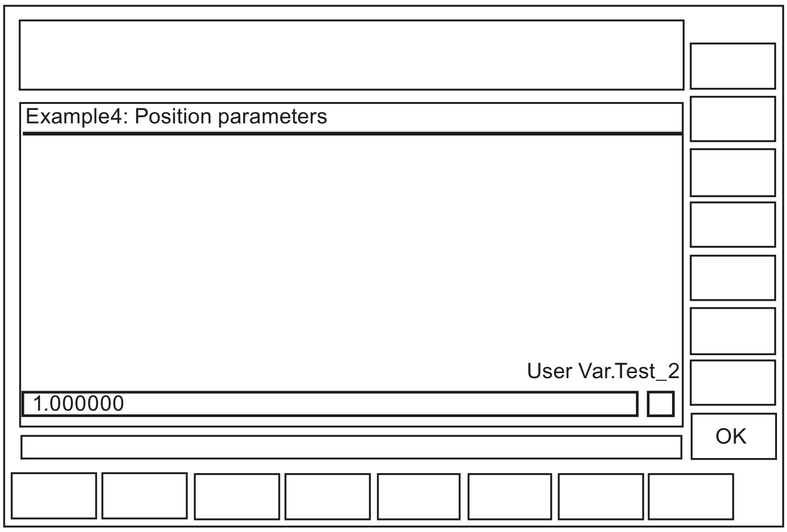



Figure 3-5 Example 4a Position parameters

To ensure that the cursor control works seamlessly, the configured fields must overlap:

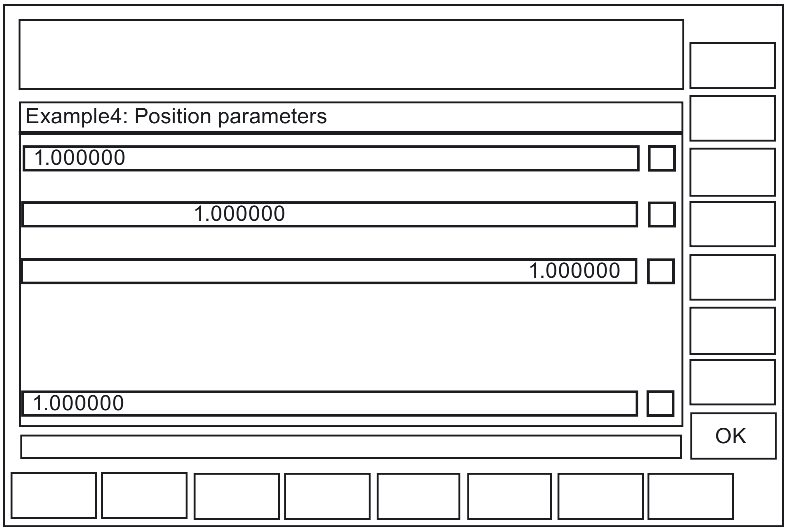



Figure 3-6 Example 4b Position parameters

## 3.3.7 Example 5: Displaying graphics in the dialog screen

### Description

Graphics created with Paintbrush, for example, can be displayed in the dialog by means of a graphics file specification.

A comment text for the graphic can also be preassigned using the COM file. You can position this comment text by specifying position parameters.

**Note**

You can only move the graphic itself by repositioning it in the graphics program.

### Selecting

```
N10
MMC("CYCLES,PICTURE_ON,T_SK.COM,PICTURE8,GUD4.DEF,GRA.BMP,,M1","S")
N20 MMC("CYCLES,PICTURE_OFF","N")
N30 M30
```

**Parameters:**

Dialog file (*.COM)

```
//C8(SCREEN8)
(I///Insert picture no. 2///4000,3000,7500)
(I///Created with Paintbrush ///4000,3250,7500)
```

| | |
|---|---|
| Text variable | `M1 = ......Example5: Insert picture.......`<br>Pictures are created, e.g., with the "Paintbrush" program. Picture size: 300X500 pixels, you can only change the picture size with the graphics program. |

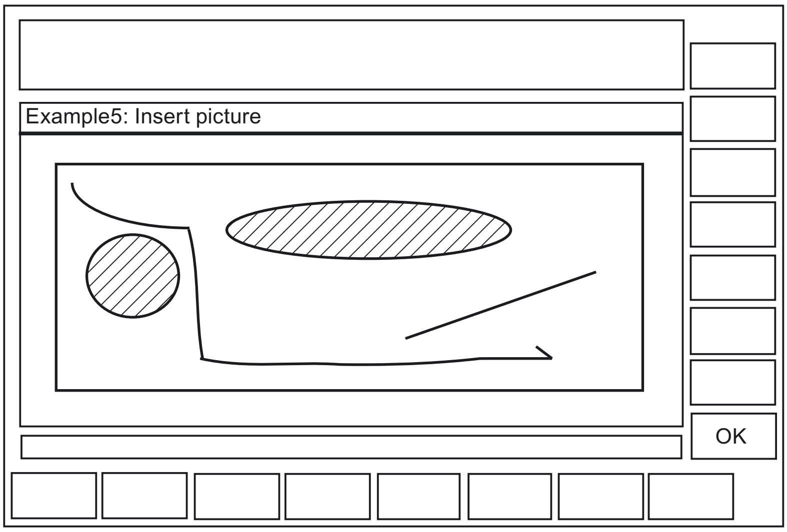



Figure 3-7 Example 5 with graphic

### 3.3.8 Example 6: Displaying OPI variables

#### Description

You can display OPI variables.

#### Selecting

```
MMC("CYCLES,PICTURE_ON,T_SK.COM,SCREEN7,GUD4.DEF,,TEST_1,J1","S")
```

**Parameters:**

Dialog file (*.COM)

```
//C7(Screen7)
(R///Test_1/R/Test_1)
(I///); (Interpreted as space line))
(R///Actual value from axis 1/R/$actual value)
(R//1/R parameter 12/W/$R[12])
```

Text variable

```
J1 = ...Example7: OPI variables

[OPIVar]
$Actual
value=/Channel/machineaxis/actToolbasePos[u1,1]
$R[12]=/Channel/Parameter/rpa[u1,12]
```

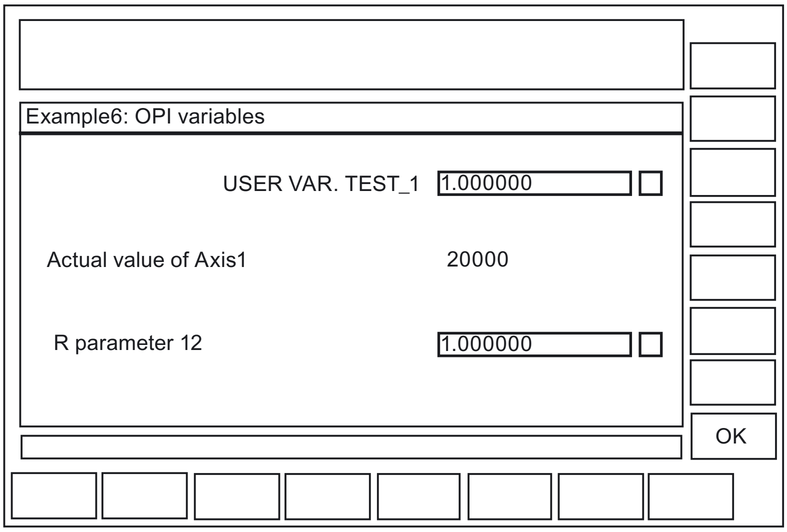



Figure 3-8 Example 6 with OPI variable

### Sequence of operations

A space line is created under the variable TEST_1.

The actual axis value is read-only.

R12 is preset with 1.

## 3.3.9 Example 7: Asynchronous acknowledgment mode with softkeys

### Description

Softkeys can be preprogrammed in the COM file for display in asynchronous mode, linked with the acknowledgment variable and evaluated in the parts program.

### Selecting

```
N10 QUIT_1 = "START"
N20
MMC("CYCLES,PICTURE_ON,T_SK.COM,PICTURE3,GUD4.DEF,"QUIT_1,K1","A")
N30 LABEL0:
N40 STOPRE
N50 IF MATCH (QUIT_1,"SK1") >= 0 GOTOF LABEL1
N60 IF MATCH (QUIT_1,"SK2") >= 0 GOTOF LABEL2
N70 GOTOB LABEL0
N80 LABEL2:
N90 MMC("CYCLES,PICTURE_ON,T_SK.COM,PICTURE1,GUD4.DEF,"N1","N")
N100 G4F10
N110 LABEL1:
```

```
N120 MMC("CYCLES,PICTURE_OFF","N")
N130 M30
```

**Parameters:**

| | |
|---|---|
| Dialog file (*.COM) | `//C3(Screen3)`<br>`(S///USER VAR QUIT_1/W/QUIT_1//)` |
| Text variable | `[TEXT VARIABLES]`<br>`K1 = ..Example8: MMC command with asynchronous acknowledgment mode`<br>`N1 = ..Example8: Picture2` |
| Softkey | `[SCREEN3]`<br>`SK1 = END`<br>`SK2 = Screen2` |

## Program structure

The acknowledgment variable is defined as a string.

```
Length of the string: >= 20
;(Values < 20 are evaluated internally only,  SK0 ... SK15 are
entered at position 17...20 when a softkey is actuated).
```

The string is assigned a value in the parts program and any old softkey information is deleted.

Before the parts program can branch as a function of the acknowledgment variable, the block search must be halted by the STOPRE command.

```
IF Match (Quit_1,"SK1") >= 0 GotoF Label1
;searches for a string within the string.
```

If no softkey has been pressed, the loop is executed again.

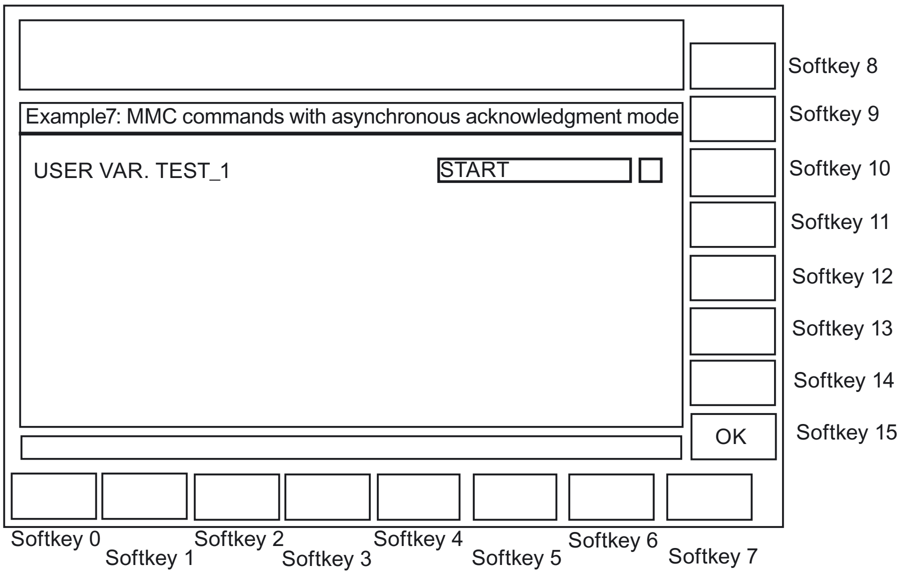



Figure 3-9 Example 7: Asynchronous acknowledgment mode

## Sequence of operations

The screen called via the asynchronous MMC command continues to be displayed until one of the two configured softkeys is actuated:

- With the "END" softkey, the user dialog is immediately exited.
- With the "Picture2" softkey, a further dialog is then displayed for 10 s.

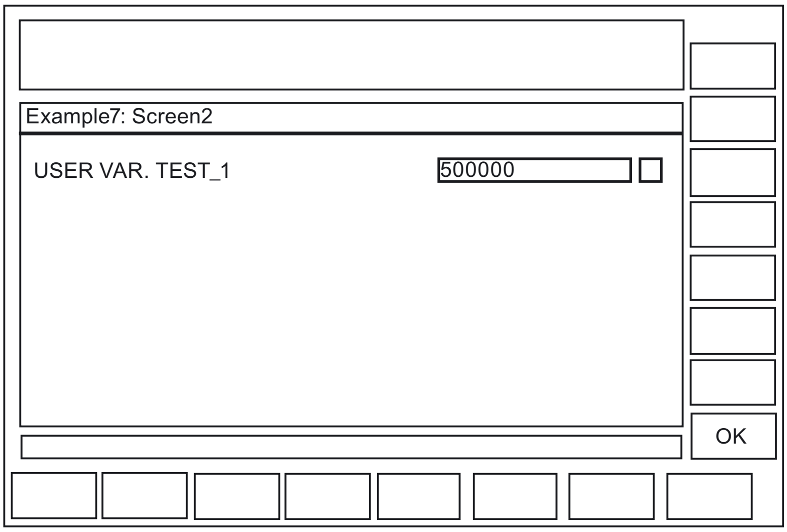



Figure 3-10 Example 7: Fig. 2:

# Configure hotkeys and PLC keys **4**

## 4.1 Introduction

### Overview

This chapter describes the configuration of the following control elements:

- 6 hotkeys on OP 010, OP 010C and SINUMERIK keyboards with hotkey block, as well as the <MACHINE> and <MENU SELECT> keys, whose assignment can be changed.
- Keys evaluated by the PLC, e.g., keys on the machine control panel
- Events that are evaluated by the PLC as PLC keys or "virtual keys", and that can trigger configured operating sequences in the HMI program.

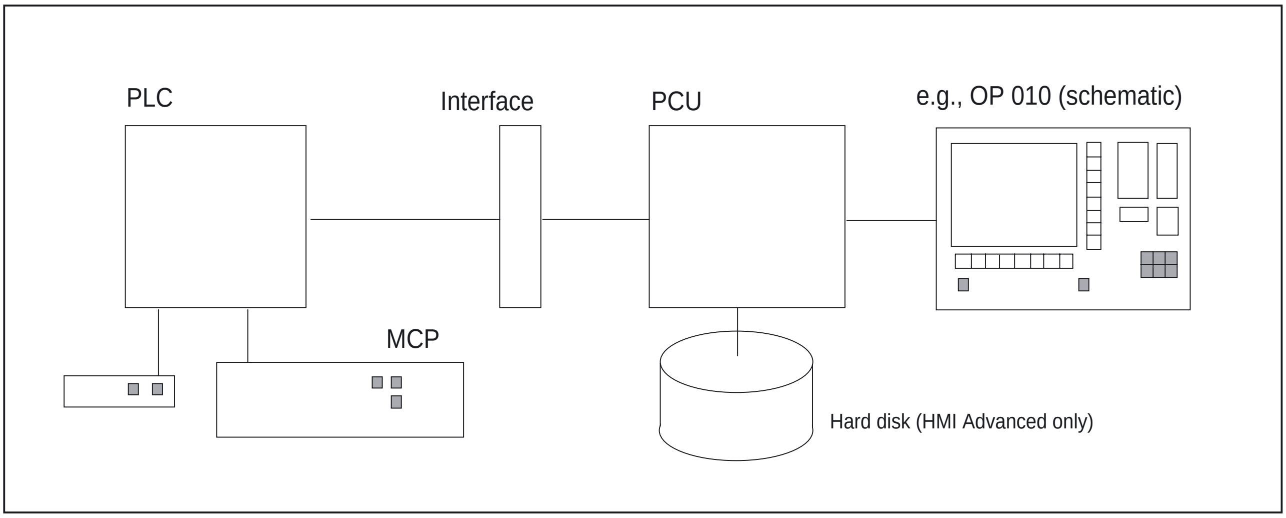



Figure 4-1 Configuration with OP 010

### Application

The hotkeys and keys can be used for, e.g., the following tasks:

- Selection of operating areas (e.g., Machine, Parameters, etc.)
- Selection of specific submenus (e.g., selection of the alarm screen in the Diagnosis operating area)
- Initiation of actions (e.g., selection of the tool list in the Parameters area and pressing of softkey HS3)

- Specific selection of menus created using the "Expanding the Operator Interface" system
- Selection of additional screens, depending on the current operating situation, in screens created using the "Expanding the Operator Interface" system.

### Configuration

Configuration is carried out using the "Expand user interface" functions.

- The 6 OP hotkeys can be used to display any HMI operating area directly. This shortens the usual selection process via the main menu. This changes the default assignment of the 6 hotkeys.
- The interface between the PLC and the HMI enables a PLC key to be transferred from the PLC to the HMI. The operations initiated on the HMI system in this way can be configured. Key numbers 50 to 254 are available for use by the PLC.
- As an option, the <Machine> and <MENU SELECT> keys can be configured in the same way as the 6 OP hotkeys and are assigned to HK7 and HK8.

## 4.1.1 OP hotkeys

### Assignment (default setting)

The 6 OP hotkeys are laid out in 2 lines, each of which contains 3 keys:

| Line 1 Label (no symbol) | | Configured as |
|---|---|---|
| OP-specific: | | |
| OP 10 | Machine | HK1 |
| OP 10C | Machine | HK1 |
| OP 10S | Position | HK1 |
| Program | | HK2 |
| Offset | | HK3 |

| Line 2 Label (no symbol) | Configured as |
|---|---|
| Program Manager | HK4 |
| Alarm | HK5 |
| Custom | HK6 |

**Optional HK7 and HK8:**

The <Machine> and <MENU SELECT> keys can be configured in the same way as HK1 to HK6. This makes it possible to disable the default settings of these keys and to activate new, user-defined functions.

| Label (no symbol) | Configured as |
|---|---|
| Machine | HK7 |
| MENU SELECT | HK8 |

You can find more information about HK7 and HK8 in the chapter "Configuring <M> key and <MENU SELECT> as HK7, HK8".

**Note**

Hotkeys 1 and 7 (<M> key) cannot be distinguished by the hardware on the OP 10S ("Position" labeling). Hot key 7 is always triggered when one of the keys is pressed.
If HK1 is configured, this event can only be triggered by an external (MF2) keyboard.

### Hotkey assignment on the MF2 keyboard

| Hotkey | OP label | Key on MF2 |
|---|---|---|
| HK1 | Position | <SHIFT+F11> |
| HK2 | Program | <END> (NB)* |
| HK3 | Offset | <Page Down> (NB)* |
| HK4 | Prg.Manager | <Home> (NB)* |
| HK5 | Alarm | <Page Up> (NB)* |
| HK6 | Custom | <SHIFT+F12> or Cursor Down (NB)* |
| HK7 | M Machine | <SHIFT+F10> |
| HK8 | Menu Select | <F10> |

*) Key is located on numeric keypad: <NumLock> must be off.

## 4.1.2 Functions of the keys on delivery

### Delivery condition

On delivery of the system, the assignment of hotkeys to functions is entered in the KEYS.INI file.

With HMI Advanced, the file can exist in various directories:

- user
- oem
- add_on *)
- mmc0w32 *)
- mmc2 *)
- hmi_adv *)

*) These directories are reserved for Siemens.

The files are executed in the above sequence. Entries in the directory, which appears at the beginning of the sequence will disable existing entries in directories, which appear later in the sequence.

On delivery, the assignments for HK1 to HK6 are stored in KEYS.INI in the mmc2 directory for the standard system:

| Key | | Function |
|---|---|---|
| HK1 | Position | Machine operating area, last screen |
| HK2 | Program | Program operating area, last screen |
| HK3 | Offset | Parameters operating area, last screen |
| HK4 | Prg. manager | Main screen programming |
| HK5 | Alarm | Diagnosis operating area, alarm screen |
| HK6 | Custom | Default Custom operating area, last screen<br>(user interface configured by the user) |

The assignment for ShopMill/ShopTurn applications is as follows:

| Key | | Function |
|---|---|---|
| HK1 | Position | Machine operating area, last screen |
| HK2 | Program | Program editor, last status |
| HK3 | Offset | Offset area, last status |
| HK4 | Prg. manager | Program directory, last status |
| HK5 | Alarm | Diagnosis operating area, alarm screen |
| HK6 | Custom | Default Custom operating area, last screen<br>(user interface configured by the user) |

# 4.2 Configuration

## 4.2.1 Configuration overview

### Overview

Below is a schematic diagram illustrating the associations between configured OP hotkeys and PLC keys and the "Expand user interface" configuration:

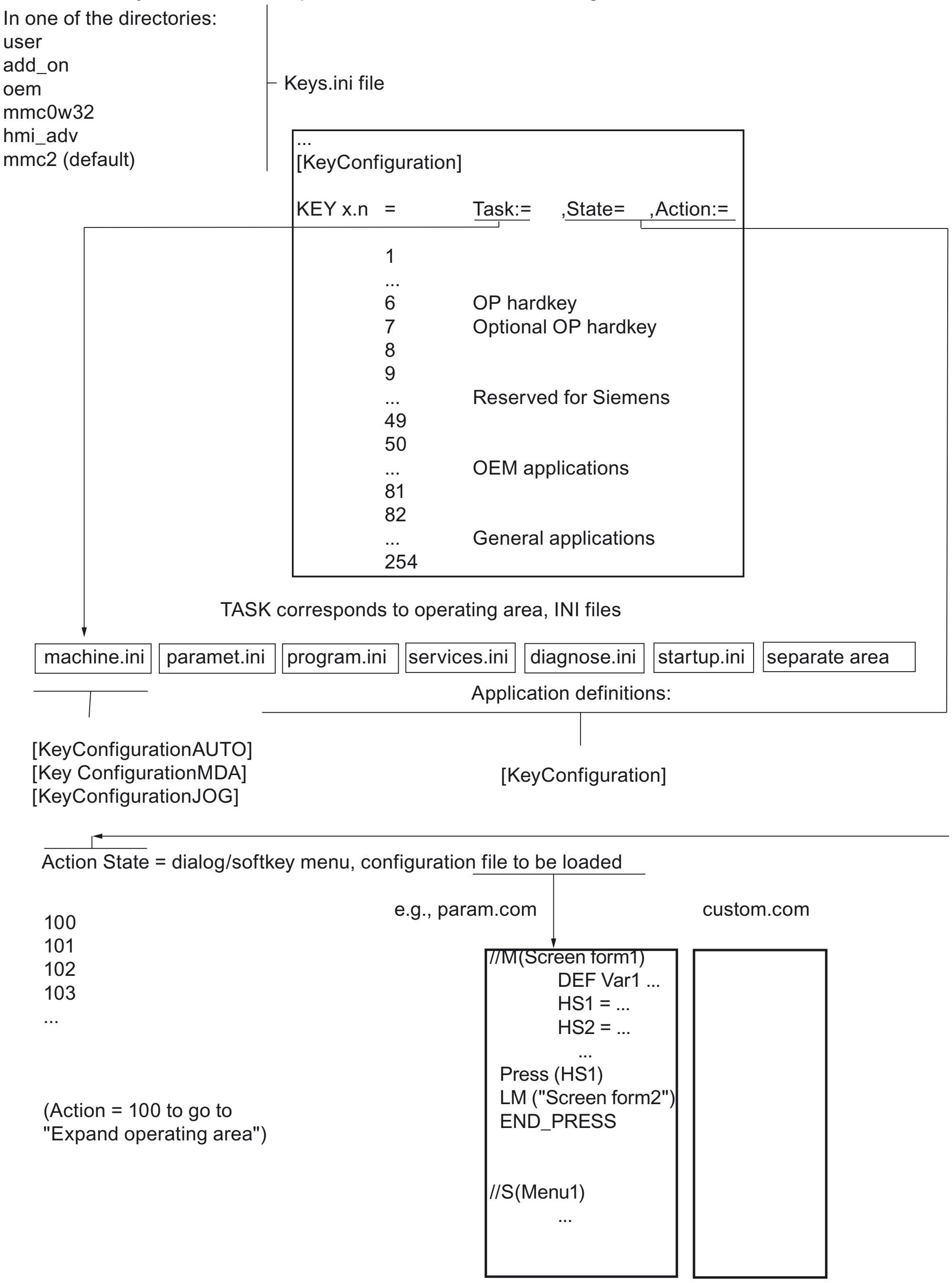



Figure 4-2 Configuration overview

### 4.2.2 Configuration in the KEYS.INI file

#### Overview

Every instance of the KEYS.INI configuration file in the above directories drives the the hotkey responses in a section [KeyConfiguration]. Every entry (line) defines the response of the system to a specific actuation (in the sense of multiple actuation of specific hotkeys). These are referred to as hotkey events below.

#### KEYS.INI configuration file

The KEYS.INI configuration file has a dedicated section for assigning the ini files for PLC keys. Without this entry, the ini files will not be detected.

The default settings are as follows:

```
[HMI_INI_FILES]
Task0 = machine.ini
Task1 = paramet.ini
Task2 = program.ini
Task3 = services.ini
Task4 = diagnose.ini
Task5 = startup.ini
Task6 = shopmill.ini
```

(Alternatively: Task6 = shopturn.ini depending on the system, mandatory! )

```
Task11=custom.ini
```

---

**Note**

A power On is required in order for the changes to KEYS.INI to take effect.

---

#### Entry format

Details about the input format can be found under "Hotkey event". To better understand the repeated actuation of the same hotkey used in that subsection, multiple keypress will be described first.

#### Multiple keypress

A function extension determines the sequence when the hotkey is pressed repeatedly:
The number "n" in the key data in file KEYS.INI represents the number of keypress repeats, thus allowing a separate task/state/action combination to be assigned each time the hotkey is pressed. This means that each time the key is pressed, a change of state can take place, and a screen and a softkey can be selected.

When the user changes to another operating area, the repeat status of the hotkey is cleared. In the new area, the hotkey is considered not to have been pressed. It does not matter how the user changed to the other operating area (operating area menu, etc.).
In addition, each time a function key is pressed (e.g., softkey, area switchover key, channel switchover key, etc.), the status is reset.

Inputting data into fields in the current dialog does not interrupt the repetition sequence. Pressing a softkey in the current dialog interrupts the repetition sequence with the softkey that was used to call the current dialog.

---

**Note**

Multiple keypress is possible for HK1 - HK8 and for keys HK9 - HK49, which are reserved for Siemens.

---

## Example of multiple hotkey keypress

The first time the hotkey is pressed, the associated area is activated and a state/action may be initiated in this area (entry 1). Pressing the hotkey again will execute the subsequent entries for this hotkey and an area switchover will not take place. Once all the entries configured for a hotkey have been executed, the cycle is repeated. The entries are always executed in ascending order (from 0-9). Execution in descending order is not possible. There must be no "event gaps" in the configuration (missing entries). A gap is treated like the end of the chain and processing will restart at Key x.0 the next time a key is pressed.

**Configuration:**

```
[KeyConfiguration]
KEY1.0 = Task:=0, State:=10,        ; Hotkey 1, 1st activation
Action:=2
KEY1.1 = State:=10, Action:=3       ; Hotkey 1, 2nd activation
KEY1.2 = Action:=4                  ; Hotkey 1, 3rd activation
```

Entry in the KEYS.INI file for selecting the machine operating area:

```
KEY1.0 = Task:=0                    ; in the last screen the first
                                      time the key is pressed
KEY1.1 = Task:=0, State:=0          ; in the start screen the second
                                      time the key is pressed
KEY1.2 = State:=0, Action:=3        ; in the start screen the third
                                      time the key is pressed and
                                      selection of the 3rd softkey
KEY1.3 = State:=0, Action:=100        in the start screen the 4th
                                      time the key is pressed and
                                      selection of "Expand user
                                      interface" function with the
                                      configured action ≤ 100 in
                                      MACHINE.INI
```

### Assignment of the softkeys to the action values

| Horizontal softkeys (SKHi) | Action |
|---|---|
| SKH1 | 0 |
| SKH2 | 1 |
| SKH3 | 2 |
| SKH4 | 3 |
| SKH5 | 4 |
| SKH6 | 5 |
| SKH7 | 6 |
| SKH8 | 7 |

| Vertical softkeys (SKVi) | Action |
|---|---|
| SKV1 | 8 |
| SKV2 | 9 |
| SKV3 | 10 |
| SKV4 | 11 |
| SKV5 | 12 |
| SKV6 | 13 |
| SKV7 | 14 |
| SKV8 | 15 |

| Key | Action |
|---|---|
| <RECALL> | 16 |
| <ETC> (HMI Embedded sl only) | 17 |

## 4.2.3 Programming a hotkey event

### Description

The maximum number of attributes, which can be assigned to a hotkey event, is 4. The KEY key attribute identifies the event and must always come first. All other attributes are optional, but at least one other attribute must be specified. These attributes can appear in any sequence in the line.

## Programming a hotkey event

| | | |
|---|---|---|
| Syntax: | KEYx.n = Task:=task, State:=state, Action:=action | |
| Parameters: | **Key** (HotKey) | The value x contains the hotkey number **x** and a discrete event **n** corresponding to the nth activation of the key (when the key is pressed several times in succession).<br>The range of values of the hotkey number goes from 1 to 254.<br>The range of values for the event n goes from 0 to 9 (10 keystrokes)<br>Hotkeys 1 to 8 are assigned to the keys of the OPs. Hotkeys 9 to 49 are reserved for Siemens. PLC-specific keys from 50 to 254 are also available. These keys are not actual hotkeys, but are used for selecting screens via the PLC (virtual keys). Keys 50 to 254 can also be configured without a task. In this case, the event is always assigned to the current task. |
| | **Task** (Operating area) | A hotkey is assigned to an operating area (task) via a task number. The task number also defines the horizontal softkey to which the operating area is assigned in the area switchover menu. This creates a permanent connection between the task number and the corresponding softkey. In the HMI system, there are three softkey menus (two in HMI Embedded sl) reserved for the area switchover function.<br>The default assignments are as follows:<br>Pressing the area switchover key selects softkey menu 1.<br>SK1 → Task0 ...<br>SK8 → Task7<br>Pressing the <ETC> key again accesses softkey menu 2<br>SK1 → Task8 ...<br>SK8 → Task15<br>Pressing the <ETC> key again accesses softkey menu 3 (only possible in HMI Advanced):<br>SK1 → Task16 ...<br>SK8 → Task23 |

Pressing the <ETC> key again returns to softkey menu 1. In HMI Advanced, the range of values for the task numbers is between 0 and 23, in HMI Embedded it is between 0 and 15. If no task is specified, the event is assigned to the current task (current operating area).

The task numbers can be accessed:

- From the REGIE.INI file for HMI Advanced.
- With the SCK (Software Configuration Kit) tool in HMI Embedded sl.

An application-specific assignment between SKs and operating areas (tasks) can be preset during configuration:

- HMI Advanced:
  [TaskConfiguration] section in the REGIE.INI file
- HMI Embedded sl: SCK "Modify configuration"

**State**

The State attribute can be used to select a specific screen within a task. The range of values depends on the individual application and is limited to between 0 and 65534 (65535 is used by the system).

**Action**

The Action attribute can be used to select a specific softkey within a task. In order to be effective, the attribute must be set from a defined state, e.g., the main menu.

The range of values depends on the corresponding application, with a maximum of between 0 and 17. A special role is played by the Action attribute with a value ≥ 100, which initiates a screen selection via "Expand user interface".

Actions in the value range 0 - 99 are not executed for ShopMill/ShopTurn.

### See also:

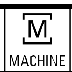
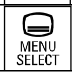
Chapter "List of selectable states"

## 4.2.4 Expansions and special cases

### Configuring the M key and Menu Select as HK7, HK8

The <M> key (Machine) and <MENU SELECT> key can be optionally configured as hotkeys. Doing so causes these keys to lose their original definition and function. The new functionality is defined by the KEYS.INI configuration file. If there is a definition in the configuration file for the key, but no response was saved, then the key does not have a function.

Hotkey assignment:

| | | MF2 keyboard | Hotkey |
|---|---|---|---|
|  | Key <Machine> | SHIFT+F10 | HK7 |
|  | Key <MENU SELECT> | F10 | HK8 |

If there is no entry in the keys.ini configuration file for hotkey 7 or hotkey 8, the key will not be mapped as a hotkey but will retain its original function (compatible mode). A key definition without a response will be assigned the <empty> attribute.

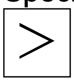
**Example:**

```
[KeyConfiguration]
Key7.0 = Task:= 3, State := 10  ; Activate mapping of the <M> key
                                  to hotkey  7 and define new  key
                                  reaction
                                ; Activate mapping of the <MENU
                                  SELECT> key (F10) to  hotkey 8 ;
                                  key has no function
Key8.0 = <empty>                ; No reaction assigned
```

### Expanded functions in HMI Advanced

The machine manufacturer can overload the entries in KEYS.INI in directory mmc2 with custom settings. These settings can be entered in the /user or /oem directories searched earlier during execution. Only deviations in the settings must be stored, not the complete block.

### Special cases for Action in HMI Advanced:



The <ETC> key, Action 17, has no effect.

### Special cases in HMI Embedded sl

Special cases Task, State, and Action:

- The range of values for the Task is limited to between 0 and 15.
- HMI Embedded sl always selects the last screen combination that was active before the task change (state not configured).
- The range of values for State is limited to 0. Only the main menu of the operating area can be selected.

- The range of values for Action is restricted to between 0 and 17 for:
  - Horizontal SK 1 to 8
  - Vertical SK 1 to 8
  - Recall
  - ETC

## 4.2.5 Expanded functions for PLC keys

### Overview

Changes of task and state can also be configured for PLC keys.

| Range for Action: | |
|---|---|
| For standard applications | 0 – 17 |
| For "Expand user interface" applications | ≥100 |

The PLC-specific keys have numbers between 50 and 254, with numbers 50 to 81 reserved for OEM applications.

**Examples:**

- The configuration shown here does not execute a change of task or state and, therefore, remains in the current task and screen. An Action 100 is forwarded.

```
[KeyConfiguration]
HK50.0 = Action:=100
```

- A change of task and state could be configured here in exactly the same way.

```
HK50.0 = Task:=1, State:=10, Action:=100
```

# 4.3 PLC interface

## 4.3.1 Structure of the interface

### Overview

In the PLC interface, area DB19. Byte10 is provided for key selection. Here, the PLC can directly select one key between 50 and 254.

(Keys 1 to 49 are reserved for Siemens, keys 50 to 254 are specific PLC keys.)

---

**Note**

In M:N mode, the area for the 2nd HMI interface is DB19.Byte60.

---

### Acknowledgment

An HMI system acknowledgment is divided into two steps:

- In the first step, the HMI software transfers the control information 255 to DB19.Byte10.
- In the second step, the actual acknowledgment takes place on the PLC in which DB19.Byte10 is cleared.

This is necessary in order that, despite the lack of synchronism between the HMI and the PLC, the same key code transferred twice in immediate succession can be detected by the HMI. This defined virtual key activation enables the HMI to identify every PLC key sequence uniquely. The control information is of no significance to the PLC program (transparent) and must not be changed.

### Next key input

If the transfer byte is set to 0, the PLC program can specify a new key. The current key request is processed in parallel in the HMI system. The request leads to a switchover to the corresponding task or initiates a state/action command in the current task.

If the task switchover cannot be executed, the operator interface will send a message to indicate this.

### Requirements of the PLC program

A new request can only be set if the HMI system has acknowledged the previous request (0 in the interface). If the PLC program is deriving the key from a machine control panel key or from another source, it must provide sufficient intermediate storage for the key to ensure that no data is lost if the key is pressed very quickly.

### Selecting dialogs from PLC

There is an interface between the PLC and HMI Embedded sl for selecting dialogs.
These PLC dialogs offer the same means of display and functions as for the dialogs selected via the softkeys.

## 4.3.2 Description of the PLC display selection

### Description of the interface

The interface contains the display number, control bits from the PLC to the HMI and control bits from the HMI to the PLC. The interface requires a total of 8 bytes in DB19, 4 bytes for each HMI Embedded sl.

Because each NCU can be addressed by up to two HMI Embedded sls at the same time in systems with "multiple operator panel fronts and multiple NCUs", this interface can exist in duplicate.

**References:** /FB2/, "Multiple operator panel fronts and multiple NCUs" (B3)

### Structure of the interface

The interface between HMI Embedded sl and the PLC uses the following data:

| | | |
|---|---|---|
| HMI 1: | DB19.DBW28: | Screen number |
| | DB19.DBB30: | Control bits PLC → HMI, PLC byte |
| | DB19.DBB31: | Control bits HMI → PLC, HMI byte |
| HMI 2: | DB19.DBW28: | Screen number |
| | DB19.DBB80: | Control bits PLC → HMI, PLC byte |
| | DB19.DBB81: | Control bits HMI → PLC, HMI byte |

| | | |
|---|---|---|
| PLC-Byte | Bit0 | Display selection |
| | Bit1 | Display selection |
| HMI byte | Bit0 | Display selection or display deselection accepted |
| | Bit1 | Display is being selected or deselected |
| | Bit2 | Display is selected |
| | Bit3 | Display is deselected |
| | Bit4 | Error, display selection not possible |
| | Bit7 | Inactive bit |

Two bytes are used for the **display number** to be transferred by the PLC; these are one PLC byte and one HMI byte for **coordinating** the display selection.

## Operating the interface

The interface is operated by the PLC program supplied by the manufacturer, using the following functions (DB 19 extract, first HMI interface):

| | PLC → HMI | Selection | Deselection | HMI → PLC | Selection | Deselection |
|---|---|---|---|---|---|---|
| DBW 28 | Screen number | (1) | | | | |
| Bit | DBB 30 | | | DBB 31 | | |
| 0 | Display selection | 1 (2)<br>0 (4) | | Select/deselect accepted | 1 (3)<br>0 (6) | 1 (2)<br>0 (3) |
| 1 | Display deselection | | 1 (1)<br>0 (4) | Display is being selected<br>Display is being deselected | 0 (3)<br>1 (5) | 0 (3) |
| 2 | | | | Display is selected | 0 (3)<br>1 (7) | 0 (3) |
| 3 | | | | Display is deselected | 0 (3) | 0 (2)<br>1 (3) |
| 4 | | | | Error, display selection not possible | 0 | 0 (2) |
| 5 | | | | – | | |
| 6 | | | | – | | |
| 7 | | | | Inactive | 0 | |

## Display selection

The relations with the individual steps are shown in the above table by the numbers in brackets (step numbers).

- The PLC enters the display number in the number word.
- The PLC sets bit0 in the PLC byte for display selection if bit0 and bit7 are set to zero in the HMI byte.
- HMI Embedded sl acknowledges to the PLC that it has accepted the display selection by setting bit0 in the HMI byte. At the same time, the value zero is written to bit3 and bit4 in the HMI byte.
- The PLC resets bit0 in the PLC byte.
- HMI Embedded sl acknowledges to the PLC that the interpreter has started by setting bit1 in the HMI byte.
- The HMI resets bit0 in the HMI Embedded sl byte.
- Bit2 is set in the HMI byte if the corresponding display appears.

## Display deselection

The relations with the individual steps are shown in the above table by the numbers in brackets (step numbers).

- The PLC sets bit1 for display deselection in the PLC byte if bit1 and bit2 are set in the HMI byte and bit7 is reset.
- HMI Embedded sl acknowledges to the PLC that it has accepted the display deselection by setting bit0 in the HMI byte. At the same time, the value zero is written to bit3 and bit4 in the HMI byte.
- HMI Embedded sl acknowledges the display deselection to the PLC by setting bit3 and resetting bit0, bit1 and bit2 in the HMI byte if the interpreter is terminated.
- The PLC resets bit1 in the PLC byte.
- After display deselection by the PLC or the EXIT command, HMI Embedded sl switches back to the last selected HMI display.

## Error when selecting a display

Bit4 is set in the HMI byte if the interpreter has not responded after 20 seconds or if the required display cannot be opened.

The states of the HMI that do not support PLC display selection are indicated to the PLC via bit7 in the HMI byte; e.g., in HMI Embedded sl during switchover between standard and ShopMill/ShopTurn interface.

## Operating areas

With **HMI Advanced**, the displays are shown in a separate operating area similar to the measuring cycle displays. This operating area can be selected manually by means of the sixth softkey (PLC displays) on the second operating area menu. If no PLC display is active when the area is selected, the text "Currently no PLC display active" is displayed in the header. The PLC is informed when the area is selected/deselected manually via bit2 of the HMI byte.

It is also possible to change to other operating areas via the MMC 103's operating area after a display has been selected by the PLC.

With **HMI Embedded sl**, it is not possible to manually select/deselect displays selected by the PLC via an operating area. The PLC display remains in the foreground even after a change of operating area has occurred.

### 4.3.3 Configuring the dialog selection

#### Description

The interface is activated by means of the section [PLC_SELECT] in the COMMON.COM file during HMI boot. The display number is assigned to the configured display in the section [PLC_SELECT] of the COMMON.COM file.

#### Configuration

| | | |
|---|---|---|
| Syntax: | PC*i* = *Display name, date, comment* | |
| Description: | Assigns display number to configured display | |
| Parameters: | i | Number of the display in the interface |
| | Display name | Dialog identifier |
| | File | File in which the dialog is configured |
| | Comment | Comment on dialog |

#### Example

```
[PLC_SELECT]
PC1= CYC82, drilling.com        ;   Assignment line
PC2= CYCLE90, millthre.com
PC3= ...
```

#### See also

Search function for COMMON.COM (Page 187)

# 4.4 Selecting dialog/softkey menus

## 4.4.1 Assigning INI files to operating areas

### Overview

An Action value ≥ 100 automatically switches the HMI software to "Expand user interface" operational sequences.

If Action ≥ 100, the configuration of the **INI files** determines which softkey menu or screen form should be displayed and in which state.

### HMI Advanced/HMI Embedded sl

The following INI files and sections can be configured for the jump entry to "Expand user interface":

| Operating area | File | Section |
|---|---|---|
| Machine | machine.ini | [KeyConfigurationAuto]<br>[KeyConfigurationMDA]<br>[KeyConfigurationJOG] |
| Parameter | paramet.ini | [KeyConfiguration] |
| Program | program.ini | [KeyConfiguration] |
| Services | services.ini | [KeyConfiguration] |
| Diagnostics | diagnosis.ini | [KeyConfiguration] |
| Start-up | startup.ini | [KeyConfiguration] |
| Custom | custom.ini | [KeyConfiguration] |

### ShopMill on NCU/ ShopTurn on NCU

In the case of ShopMill and ShopTurn on NCU, the files are accordingly named SHOPMILL.INI and SHOPTURN.INI. The following sections, in which "Expand user interface" functions can be configured, are parts of these files:

```
[MachineManual]
...
[MachineAutomatic]
...
[ProgramManager]
...
[Program]
...
[MessagesAlarms]
...
[ToolsZeroOffset]
...
[MachineMDI]
```

### 4.4.2 Configuring "Expand user interface" functions

#### Description

One or more states (according to the State list) can be configured for each action (≥ 100) and you can also specify which "Expand user interface" function is to be initiated. If nothing is configured in these INI files, no screen forms or softkey menus will appear.

#### Programming Action.State

| | | |
|---|---|---|
| Syntax: | Action.State = Dialog/softkey menu to be loaded, configuration file | |
| Parameters: | Action | The action is Action ≥ = 100 configured in KEYS.INI. |
| | State | State in which the application is presently in |
| | Configuration file | File in which the configuration is stored. |
| | LS/LM | "Expand user interface" commands for loading softkeys/dialogs |

#### Examples

```
[KeyConfiguration]
100.10=LS("Softkey1","param.com")
100.30=LM("Form1","param.com")
101.10=LS("Softkey1","param.com")
101.30=LM("Form2","param.com")
102=LM("Form2","param1.com")
```

Explanations:

- For the first line, for example, this means:

  If an Action 100 is initiated when in state (status/screen) 10, the softkey menu configured in the param.com file will be displayed with the name Softkey1.
- For the last line, for example, this means:

  If Action 102 is initiated, the dialog with the name Screen form2 will be called in the current dialog. This dialog is configured in the param1.com file.

# 4.5 List of selectable states

## 4.5.1 Selectable states in HMI Advanced

### Overview

In general, the following conditions apply to all tasks in HMI Advanced:

| | |
|---|---|
| No configuration: | Maintains current state. |
| Configuration with 0: | The basic state of an operating area is activated. |

### "Machine" operating area

In the Machine operating area, the status is always dependent on the status of the machine (AUTO, MDA, JOG, REF). The following can be selected via the PLC key:

- The main screen for the relevant mode

| State | BAG | Machine function | Display |
|---|---|---|---|
| 0 | JOG | REF | JOG/REF main screen |
| 0 | JOG | none | JOG main screen |
| 0 | JOG | REPOS | REPOS main screen |
| 0 | MDI | none | MDA start screen |
| 0 | MDI | TEACH | MDA/Teach main screen |
| 0 | MDI | REF | MDA/REF main screen |
| 0 | AUTO | none | Auto main screen |

- The main actual value screens for each mode (softkey vertical 6)

| State | BAG | Machine function | Display |
|---|---|---|---|
| 10 | JOG | none | Zoom actual values JOG |
| 20 | MDI | none | Zoom actual values MDA |
| 30 | AUTO | none | Zoom actual values Auto |

- Possibly still in JOG / MDA

  Switching between the WCS and MCS is possible via the PLC.

| State | BAG | Machine function | Display |
|---|---|---|---|
| 60 | JOG | none | Handwheel selection JOG |
| 70 | JOG | none | Increment selection JOG |
| 80 | MDI | none | Handwheel selection MDA |

## "Parameters" operating area

It is not possible to move from the current screen in this area.

## "Program" operating area

| State | Function | |
|---|---|---|
| 10 | Data selection | |
| 20 | Program management | |
| 70 | Log | |

## Services operating area

| State | Function | |
|---|---|---|
| 10 | Data in | |
| 20 | Data out | |
| 40 | Manage data | |
| 60 | Data selection | |
| 80 | External drives | |
| 90 | Series startup | |
| 100 | upgrading | |

## Diagnostics operating area

The diagnostics main screen is the alarm overview.

The following horizontal softkeys can be used to access other states from this state:

| State | Function | |
|---|---|---|
| 10 | Alarms | |
| 20 | Alarms | |
| 30 | Alarm log | |
| 40 | Service displays | |
| 50 | PLC status | |

Messages, alarm log and service displays can always be accessed.

## Startup operating area

The startup main screen is an overview of the available NC axes and drives.

| State | Function | |
|---|---|---|
| 0 | NC axes and drives | |
| 10 | Machine data | |
| 40 | PLC status | |

| State | Function | |
|---|---|---|
| 50 | Optimization/test | (V7.1 and later) |
| 60 | HMI | |

### Custom operating area

See chapter "Custom operating area".

## 4.5.2 Selectable states in HMI Embedded sl

### Overview

There are the following configuration options for the state in HMI Embedded sl:

| | |
|---|---|
| A configuration: | Maintains current state. |
| Configuration with 0: | The basic state of an operating area is activated. |

## 4.5.3 Selectable states in ShopMill on NCU

### Machine Manual

Legend:

| | |
|---|---|
| * | If option available (display MD is set) |
| Custom screen | These designated functions can be configured with the "Expanding the Operator Interface" system.<br>If this type of configuration is available, it is activated.<br>Otherwise, the standard ShopMill screen will appear. |

| State | Function |
|---|---|
| 19 | Main screen |
| 2 | T, S, M, etc. |
| 30 | Workpiece zero |
| 5 | Workpiece zero - set up edge |
| 7 | Workpiece zero – /user screen form |
| 31 | Workpiece zero – align edge/user screen form |
| 32 | Workpiece zero – spacing 2 edges/user screen form |
| 33 | Workpiece zero – rectangular edge |
| 8 | Workpiece zero – arbitrary edge/user screen form |

| State | Function |
|---|---|
| 34 | Workpiece zero – rectangular pocket |
| 9 | Workpiece zero – 1 hole/user screen form |
| 35 | Workpiece zero – 2 holes |
| 36 | Workpiece zero – 3 holes |
| 37 | Workpiece zero – 4 holes |
| 38 | Workpiece zero – rectangular spigot |
| 10 | Workpiece zero – 1 circular spigot /user screen form |
| 39 | Workpiece zero – 2 circular spigots |
| 40 | Workpiece zero – 3 circular spigots |
| 41 | Workpiece zero – 4 circular spigots |
| 42 | Workpiece zero - set up plane |
| 11 | Workpiece zero – calibration probe, length*/user screen form |
| 12 | Workpiece zero – calibration probe, radius* |
| 50 | Measure tool |
| 16 | Tool gauging – length manual/user screen form |
| 17 | Tool gauging – diameter/user screen form |
| 13 | Tool gauging – length auto*/user screen form |
| 14 | Tool gauging – diameter auto*/user screen form |
| 51 | Measure tool – /user screen form |
| 15 | Measure tool – calibration probe*/user screen form |
| 52 | Measure tool – calibration fixed point*/user screen form |
| 60 | Swiveling* |
| 4 | Position |
| 18 | Face milling |
| 1 | ShopMill settings |
| 90 | – /user screen form |

## MDI

| State | Function |
|---|---|
| 20 | MDI |

## Machine Auto

| State | Function |
|---|---|
| 200 | Main screen |
| 210 | Program control |
| 220 | Block search |
| 230 | – /user screen form |
| 242 | Simultaneous recording – top view* |
| 243 | Simultaneous recording – 3-plane view* |

| State | Function |
|---|---|
| 244 | Simultaneous recording – volume model* |
| 250 | Settings |

## Program Manager

| State | Function |
|---|---|
| 300 | NC directory |
| 310 | Parts programs* |
| 320 | Subprograms* |
| 330 | User directory 1 * |
| 340 | User directory 2 * |
| 350 | User directory 3 * |
| 360 | User directory 4 * |
| 380 | Standard cycles* |
| 381 | Manufacturer cycles* |
| 382 | User cycles* |
| 383 | User directory 5 * |
| 384 | User directory 6 * |
| 385 | User directory 7 * |
| 386 | User directory 8 * |

## Program

| State | Function |
|---|---|
| 400 | Machining plan / G code editor |
| 412 | Simulation – top view* |
| 413 | Simulation – 3-plane view* |
| 414 | Simulation – volume model* |

## Messages/alarms

| State | Function |
|---|---|
| 500 | Alarms |
| 510 | – /user screen form |
| 520 | – /user screen form |

## Tools/zero offsets

| State | Function |
|---|---|
| 600 | Tool list |

| State | Function |
|---|---|
| 610 | Tool wear |
| 620 | User tools list* |
| 630 | Magazine |
| 640 | Zero point shift |
| 650 | R parameters |
| 660 | – /user screen form |
| 680 | User data |
| 690 | Machine data |

### 4.5.4 Selectable states in ShopTurn on NCU

#### Machine manual (without "Machine manual" option)

Legend:

| | |
|---|---|
| * | If option available (display MD is set) |
| Custom screen | These designated functions can be configured with the "Expanding the Operator Interface" system.<br>If this type of configuration is available, it is activated.<br>Otherwise, the standard ShopMill screen will appear. |

| State | Function |
|---|---|
| 19 | Main screen |
| 2 | T, S, M, etc. |
| 30 | Workpiece zero |
| 31 | Workpiece zero – /user screen form |
| 34 | Workpiece zero – /user screen form |
| 35 | Workpiece zero – /user screen form |
| 36 | Workpiece zero – /user screen form |
| 37 | Workpiece zero – /user screen form |
| 38 | Workpiece zero – /user screen form |
| 40 | Workpiece zero – /user screen form |
| 5 | Workpiece zero – measure edge Z |
| 50 | Measure tool |
| 51 | Measure tool – length manual X/user screen form |
| 52 | Measure tool – length manual Z/user screen form |
| 53 | Measure tool – zoom*/user screen form |
| 54 | Measure tool – /user screen form |
| 55 | Measure tool – /user screen form |

| State | Function |
|---|---|
| 56 | Measure tool – calibration probe*/user screen form |
| 57 | Measure tool – /user screen form |
| 58 | Measure tool – automatic Z * |
| 59 | Measure tool – automatic X * |
| 4 | Position |
| 18 | Face milling* |
| 80 | Cutting* |
| 90 | – /user screen form (tailstock) |
| 1 | ShopTurn settings |

### Machine hand (with "Machine manual" option)

| State | Function |
|---|---|
| 19 | Main screen |
| 50 | Measure tool |
| 51 | Measure tool – length manual X/user screen form |
| 52 | Measure tool – length manual Z/user screen form |
| 53 | Measure tool – zoom*/user screen form |
| 54 | Measure tool – /user screen form |
| 55 | Measure tool – /user screen form |
| 56 | Measure tool – calibration probe*/user screen form |
| 57 | Measure tool – /user screen form |
| 58 | Measure tool – automatic Z * |
| 59 | Measure tool – automatic X * |
| 1300 | Straight |
| 1400 | Bore |
| 1410 | Drilling – centered |
| 1420 | Drilling – thread centered |
| 1433 | Drilling – centering* |
| 1434 | Drilling – drilling* |
| 1435 | Drilling – reaming* |
| 1440 | Drilling – deep-hole drilling* |
| 1453 | Drilling – tapping* |
| 1454 | Drilling – thread milling* |
| 1500 | Turning |
| 1513 | Turning – cutting 1 |
| 1514 | Turning – cutting 2 |
| 1515 | Turning – cutting 3 |
| 1523 | Turning – groove 1 |
| 1524 | Turning – groove 2 |
| 1525 | Turning – groove 3 |
| 1533 | Turning – undercut form E |

| State | Function |
|---|---|
| 1534 | Turning – undercut form F |
| 1535 | Turning – undercut thread DIN |
| 1536 | Turning – undercut thread DIN |
| 1543 | Turning – thread, longitudinal |
| 1544 | Turning – thread, taper |
| 1545 | Turning – thread, facing |
| 1550 | Turning – cut-off |
| 1600 | Milling* |
| 1613 | Milling – rectangular pocket* |
| 1614 | Milling – circular pocket* |
| 1623 | Milling – rectangular spigot* |
| 1624 | Milling – circular spigot* |
| 1633 | Milling – longitudinal groove* |
| 1634 | Milling – circumferential groove* |
| 1640 | Milling – polyhedron* |
| 1670 | Milling – engraving* |
| 1730 | Simulation – 3-window view* |
| 1740 | Simulation – side view* |
| 1750 | Simulation – front view* |
| 90 | – /user screen form (tailstock) |
| 1 | ShopTurn settings |

## MDI

| State | Function |
|---|---|
| 20 | MDI |

## Machine Auto

| State | Function |
|---|---|
| 200 | Current block display |
| 210 | Program control |
| 220 | Block search |
| 230 | – /user screen form |
| 242 | Simultaneous recording – 3-window view* |
| 243 | Simultaneous recording – side view* |
| 244 | Simultaneous recording – front view* |
| 250 | Settings |

## Program Manager

| State | Function |
|---|---|
| 300 | NC directory |
| 310 | Parts programs* |
| 320 | Subprograms* |
| 330 | User directory 1 * |
| 340 | User directory 2 * |
| 350 | User directory 3 * |
| 360 | User directory 4 * |
| 380 | Standard cycles* |
| 381 | Manufacturer cycles* |
| 382 | User cycles* |
| 383 | User directory 5 * |
| 384 | User directory 6 * |
| 385 | User directory 7 * |
| 386 | User directory 8 * |

## Program

| State | Function |
|---|---|
| 400 | Machining plan / G code editor |
| 412 | Simulation – 3-window view* |
| 413 | Simulation – side view* |
| 414 | Simulation – front view* |

## Messages/alarms

| State | Function |
|---|---|
| 500 | Alarms |
| 510 | – /user screen form |
| 520 | – /user screen form |

## Tools/zero offsets

| State | Function |
|---|---|
| 600 | Tool list |
| 610 | Tool wear |
| 620 | OEM tool list* |
| 630 | Magazine |
| 640 | Zero point shift |
| 650 | R parameters |
| 660 | – /user screen form |
| 670 | Spindles |
| 680 | User data |
| 690 | Machine data |

# "Custom" operating area 5

## 5.1 Delivery condition and use

### Overview

The operating areas described thus far can be extended and modified using "Expanding the Operator Interface" tools. Extensions can only be applied to softkeys, which have not been used previously.

Using the tools described below, it is possible to configure a separate operating area in both HMI Embedded sl and HMI Advanced, where all 8 horizontal and 8 vertical softkeys are available for a user-specific user interface.

The default name of this operating area is "Custom".

On OPs with hotkey block (e.g., OP 010, OP 010C), the "Custom" operating area can be directly selected with:

- "Custom" hotkey
- Softkey 4 in the 1st horizontal expanded menu (default setting)

### Condition on delivery

The "Custom" operating area displays an empty window with a configurable header across the whole area of the local menu. The "Custom" operating area also permits a configurable text in the operating area display field in the global menu.

All softkeys are empty and can be assigned as required by the customer using "Expand user interface".

When switching from and back to the "Custom" area, the screen active before exiting the "Custom" operating area is active.

- HMI Advanced

  The "Custom" operating area is available by default on delivery and can be reset via an entry in the REGIE.INI file or assigned to any other horizontal softkey.

- HMI Embedded sl

  The "Custom" operating area is available by default. It is supplied with the application diskette and can be assigned to any horizontal softkey by the customer using the Software Configuration Kit (SCK.exe).

# 5.2 Activating the operating area

## HMI Advanced

The operating area is enabled in the REGIE.INI file and is assigned to a softkey.

### Example

Activation via horizontal softkey 4 in the 1st expanded menu (default setting).

- The entry must be made in the [Miscellaneous] section:

```
[TaskConfiguration]
Task11 = name := custom, Timeout := 12000
```

- In the REGIE.INI file, this operating area can also be selected as the standard startup area. The entry must be made in the [Miscellaneous] section:

```
[Miscellaneous]
PoweronTaskIndex = 11
```

The corresponding lines can be deactivated by adding a ";" comment character at the beginning of the line if the "Custom" operating area is not to be selected or should not be activated on startup. The area can be assigned to a different softkey by changing the task number.

## HMI Embedded sl

Using the display machine data MD 9016: MM_SWITCH_TO_AREA, you can define in which operating area HMI Embedded sl should start up (the "Custom" operating area can be specified here).

The value in the MD provides the number of the softkey for the required operating area.

| | |
|---|---|
| Default: | 12 |
| Horizontal softkeys 1 to 8: | 1 – 8 |
| Softkeys in the expanded menu: | 9 – 16 |

The HMI software for the "Custom" application evaluates the CUSTOM.INI file and decides whether the area should be displayed. In HMI Embedded sl, there is a section [Activate] for this:

```
[Activate]
Activate=True
```

The operating area is assigned to a horizontal softkey by the customer using the Software Configuration Kit (SCK.EXE), menu option "Modify configuration".

In order for the Custom operating area to be activated by default on start-up, the softkey numbers must be entered in display machine data MD 9016: MM_SWITCH_TO_AREA.

If this display MD has the value -1, HMI Embedded sl will start up in the operating area specified as the start-up area with SCK. On delivery, softkey 1 on the main menu is in the "Machine" operating area.

### Activation by keys

- Hotkey

  The "Custom" hotkey on the OP 010 and OP 010C always activates this operating area (default setting). Other additional configurations are possible.

- Horizontal softkey

  Activation via the horizontal softkey 4 in the 1st expanded menu is the default setting. The softkey assignment can be changed in HMI Advanced; in HMI Embedded sl, this is done with the help of the Software Configuration Kit (SCK).

### Behavior during operating area change

When you change from the "Custom" operating area to another operating area and back again, the window that was active when you closed the "Custom" operating area reappears the next time you open it.

## 5.3 Defining the start dialog

### Overview

The inputs in the CUSTOM.INI file are used to define the start dialog and those in the RE_xx.INI file for the operating area name in HMI Advanced.

- Dialog header

  A text with the dialog header can be entered in this section [Header]. This can be in the form of either text or an alarm text number, which will make the header language-dependent:

  ```
  [Header]
  Text="XY special functions"
  Text=$80XXX
  ```

  Default setting: `Text = "Custom"`

- Picture in start dialog

  In the [Picture] section, it is possible to enter a path to a picture that is displayed at the start of the application:

  ```
  [Picture]
  Picture=\directory\bild.bmp
  ```

- Operating area name

| HMI Advanced | HMI Embedded sl |
|---|---|
| The name specified appears on the top left of the start dialog displayed.<br><br>`[HSoftkeyTexts]`<br>`HSK11 = "Custom"` | The name specified appears on the top left of the start dialog displayed.<br><br>`[Task name]`<br>`Notation: Text = $80xxx` |

- Softkey labels

| HMI Advanced | HMI Embedded sl |
|---|---|
| The name entered as the operating area name in the RE_xx.INI file is also displayed on the configured softkey.<br>"xx" stands for the language code. | The text entered is displayed on the softkey assigned. If nothing has been entered, "Custom" will appear as the default setting.<br><br>`[Softkey]`<br>`Text = $80xxx` |

All other elements in the "Custom" operating area, such as softkey menus or input/output fields and the relevant functions, must be configured using the "Expand user interface" tools.

In the "Custom" application all softkeys are available as "Expand user interface" softkeys.

These softkeys are configured in the CUSTOM.COM file (like the standard version of AEDITOR.COM included in the scope of supply) as described below.

In HMI Embedded sl, the COMMON.COM file for customer start softkeys must contain references to CUSTOM.COM. The COMMON.COM file is supplied so that all "Custom" operating area softkeys are linked to CUSTOM.COM.

# Configuring environment 6

## 6.1 Scope of supply

### Overview

The software that interprets the configuration files for user interfaces and provides or activates the functions described is included in the scope of delivery of the HMI software, along with an ASCII editor for creating the content of the configuration files (program editor).

### Products

ShopMill on NCU and ShopTurn on NCU are based on HMI Embedded sl. HMI Embedded WIN32 and HMI Advanced can be combined on a single hardware platform.
Thus, the operating systems of these products are available as alternatives to HMI Advanced on a single hardware platform.

### Differences in the system

Because of the differences in the hardware, the configuration files are created in another way:

- HMI Advanced on PCU 50 has a hard disk.
- HMI Embedded sl only has RAM and user memory on the CF card.
- HMI Embedded WIN32 on PCU 50 or PC with Windows evaluates hard disk configuration files but operates in the same way as HMI Embedded sl in all other respects.

### Creating displays

If the configured screen forms are to include images, then a suitable graphics program (e.g., MS Paint) will be required.

### See also

Search function principle (Page 184)

# 6.2 Creating configuration data

## 6.2.1 Using the COMMON.COM file

### Overview

When using HMI Advanced, no entries are required in the COMMON.COM file.

In HMI Embedded sl, this central control file contains the following information:

- Assignment between start softkeys and configuration files
- Assignment of display numbers to configuration files in PLC interface DB19
- Control entries (size of LOG file, available memory on the temporary directory for configuration files).

Default setting for HMI Advanced/HMI Embedded sl:

| Main screen | Horizontal softkey | Configuration file |
|---|---|---|
| Machine JOG | 1 | MA_JOG.COM |
| Machine MDA | 1 | MA_MDA.COM |
| Machine Automatic | 2 | MA_AUTO.COM |
| Parameter | 7 | PARAM.COM |
| Program | 8 | PROG.COM |
| Services | 7 | SERVICE.COM |
| Diagnostics | 7 | DIAG.COM |
| Start-up<br>expanded softkey menu | 7<br>6, 7 | STARTUP.COM |
| Editor (reserved) | 2, 3, 4, 5 | AEDITOR.COM |
| Editor<br>expanded softkey menu | 6<br>6, 7 | AEDITOR.COM |

### Naming convention and file size

- HMI Embedded sl

  All file names must follow DOS conventions (xxxxxxxx.com).

  The total permissible number of configuration files is 10.

  Color format for the Help displays: 256-color bitmaps in bmp format.

  The memory requirement for the displays and configuration files is determined solely by the size of the CF card used. It is not possible to state what the maximum possible number of files can be.

- HMI Advanced

  Files with the names of the start softkeys are searched for in directories in the order specified above. If files of the same name are stored in different directories, then the first file detected according to the search strategy is interpreted. The sizes of configuration files are not subject to any particular restrictions. It should be noted, however, that large files are processed more slowly.

### 6.2.2 Structure of the COMMON.COM file

#### Overview

The COMMON.COM file is supplied together with the cycles. It contains various sections for specific hardware settings. For HMI Embedded sl, section [MMC_DOS] is relevant for the "Expand user interface" system.

#### Configuring COMMON.COM

| | | | |
|---|---|---|---|
| Syntax | Parameter = *Value* | ; | Any number of blanks can be placed between Parameter, the '=' sign and *Value*. |
| Description | [MMC_DOS] | ; | Start of the DOS section |
| | | ; | Text contained within a line after a semicolon ; is treated as a comment and is not evaluated |
| Parameter | All parameters are optional. | | |
| | SCxxx=*File* | | Softkey connection: Connection between softkey and configuration file<br>"xxx" represents an internal softkey identity of the start softkey. The softkey identity must be entered directly after SC. Softkeys will not be displayed unless a softkey connection is defined for them. |
| | HCyyy = *File* | | Hotkey connection: Connection between hotkey and configuration file.<br>"yyy" represents an internal hotkey identity of the start hotkey. The hotkey identity must be entered directly after HC. Only hotkeys for which a hotkey connection has been defined have an effect. |
| | File | | The configuration file, which contains the softkey and dialog definitions. File names may not exceed a length of 8 characters. The file extension is added with a dot.<br>Example SC101= my_file.com ; (my_file.com on HMI, NC) |

**Note**

Changes to COMMON.COM only take effect following a restart.

## Control entries

| | | | |
|---|---|---|---|
| Syntax: | CHK_FILE_EXIST=*ram* | | |
| Identifiers: | This control flag: specifies whether configuration files must be copied each time by the NC or whether a check must be made as to whether the files are already stored on the RAM drive of the HMI. | | |
| Parameters: | ram | Possible values: | |
| | | 0: | A check whether the file already exists on the temporary drive is not carried out. This mode is only set while the configuration files are being set up online on the NC. Changes to configuration files in the NC then become operative immediately on the HMI Embedded sl, slowing down the display building process. |
| | | 1: | Default setting (if CHK_FILE_EXIST has not been set): The configuration files are read once into the temporary directory and executed thereafter from there. This means Runtime is improved, but there is no reaction to changes in the configuration files in the NC. |

| | | |
|---|---|---|
| Syntax: | LOGSIZE=*kB* | |
| Identifiers: | A LOG file named ERROR.COM is created on the temporary directory of the HMI; its size is determined by this parameter setting. | |
| Parameters: | KB | Size of the LOG file in kilobytes (max. size = 64 KB). |

| | | |
|---|---|---|
| Syntax: | RAMDISK_SIZE = *kBrd* | |
| Identifiers: | RAMDISK_SIZE can be located at any position - within or outside the sections. If the element occurs several times, then the **first** occurrence is relevant.<br>If, **after** copying, the selected size has been exceeded, all of the COM files in the temporary directory are deleted **before** the next copying operation. (Files for dialogs in the background for operating area change are retained.) | |
| Parameters: | kBrd | Size of available RAMDISK in KB.<br>Default: 300 KB<br>(if the RAMDISK_SIZE element was not specified) |

### Example of COMMON.COM

| | | |
|---|---|---|
| [MMC_DOS] | | |
| sc101=tooth.com | ; | MACHINE |
| sc111=mda.com | ; | The files are stored on the flash memory of the HMI. |
| sc122=auto.com | | |
| sc207=param.com | ; | PARAMETER |
| sc314=aeditor.com | ; | PROGRAM, Editor |
| sc315=aeditor.com | | |
| sc316=aeditor.com | | |
| sc407=dienste.com | ; | SERVICES |
| sc507=diagnose.com | ; | DIAGNOSIS |
| sc607=inbetrn.com | ; | STARTUP |
| sc826=cmm.com | ; | ShopMill, Machine, AUTO |
| sc857=cmm.com | ; | Messages / alarms |
| sc858=cmm.com | | |
| sc867=cmm.com | ; | Tools, WO |
| CHK_COMMON.COM=1 | ; | Rapid execution of the HMI |
| LOGSIZE=30 | ; | Size of the error log (LOG file) 30 KB |
| | | |
| [PLC_SELECT] | ; | Notification of displays, which can be called by the PLC |
| PC1= CYC82, drilling.com | ; | Fig. 1: |
| PC2= CYCLE90, millthre.com | ; | Fig. 2: |

## 6.2.3 Configuring start softkeys

### Overview

The start softkeys named here can be used to activate the associated configuration files. The possible start softkeys for dialogs are predefined. Additional start softkeys are not possible. Start softkeys are specific to operating areas.

## Programming

| | | |
|---|---|---|
| Syntax | SCxxx = *File* | |
| Identifier | Softkey connection: Connection between softkey and configuration file<br>"xxx" represents an internal softkey identity of the start softkey. | |
| Parameter | *File* | Name of configuration file |

## Entry points in the operating areas

| Operating area | SCxxx | Output dialog | |
|---|---|---|---|
| MACHINE | SC101 | MACHINE JOG | Horizontal SK 1 |
| | SC111 | MACHINE MDA | Horizontal SK 1 |
| | SC122 | MACHINE AUTO | Horizontal SK 2 |
| PARAMETER | SC207 | Main screen of PARAMETERS | Horizontal SK 7 |
| PROGRAM | SC308 | Main screen of PROGRAM | Horizontal SK 8 |
| | SC312 | 1. softkey line of the editor | Horizontal SK 2 |
| | SC313 | 1. softkey line of the editor | Horizontal SK 3 |
| | SC314 | 1. softkey line of the editor | Horizontal SK 4 |
| | SC315 | 1. softkey line of the editor | Horizontal SK 5 |
| | SC316 | 1. softkey line of the editor | Horizontal SK 6 |
| | SC326 | 2. Softkey menu of the editor | Horizontal SK 6 |
| | SC327 | 2. Softkey menu of the editor | Horizontal SK 7 |
| SERVICES | SC407 | Main screen of SERVICES | Horizontal SK 7 |
| DIAGNOSIS | SC507 | Main screen of DIAGNOSIS | Horizontal SK 7 |
| IBN | SC607 | Main screen of START-UP | Horizontal SK 7 |
| | SC616 | 2. Softkey menu of the editor | Horizontal SK 6 |
| | SC617 | 2. Softkey menu of the editor | Horizontal SK 7 |

The file names listed in chapter "Table of start softkeys" are already stored as default settings. However, the matching files must be created on the NC or HMI by the user.

## See also

List of start softkeys (Page 189)

### 6.2.4 Language-dependent text

#### Overview

The language-dependent texts for dialogs are stored in ASCII text files. The syntax is the same as that of an alarm text file .

Language-dependent texts can be used for:

- Softkey labels
- Headings
- Help texts
- Any other texts

#### Permissible file names

The file names are defined as follows:

| | |
|---|---|
| Alsc.txt | Contains the language-dependent texts for the Siemens standard cycles |
| Almc.txt | Contains the language-dependent texts for the manufacturer cycles |
| Aluc.txt | Language-dependent user texts |

#### Format of a text entry

| | | | |
|---|---|---|---|
| Syntax | 8xxxx 0 0 "Text" | | |
| Description | Assignment between text number and text in the file | | |
| Parameter | xxxx | 5000 to 9899 | Text identification number range reserved for users. You must assign unique numbers. |
| | "text" | | Text that appears in dialog |

Parameters 2 and 3 are separated by blanks and act as control characters for alarm text output. To ensure that the text format is identical to that of the alarm texts, these two parameters must always be set to 0.

The following control characters may be inserted in text:

| | |
|---|---|
| %n | Line break |
| %@x | Axis name of axis x (x is the axis number); HMI Embedded sl only |
| | Show axis names (HMI Embedded sl and HMI Advanced): |
| | NC access to corresponding machine data element containing the relevant axis name; text assembly by means of the string functions it contains. |

**Examples:**

| | |
|---|---|
| 85000 0 0 | "Retraction plane" |
| 85001 0 0 | "Drilling depth" |
| 85002 0 0 | "Pitch" |
| 85003 0 0 | "Pocket radius" |

# 6.3 Storage structure of configuration files

## 6.3.1 HMI Embedded sl

### Storage

In the Linux environment, the user configurations are copied to the **/user/sinumerik/hmi/proj** directory (for "normal" user dialogs, i.e., everything except cycles support) on the CF card. User configurations for user cycles support are copied to the **/user/sinumerik/cycles/proj** directory on the CF card. All files are **unpacked** and copied to the relevant directory.

Similarly, the manufacturer configurations are copied to the **/oem/**sinumerik/.... directories.

| Path | Content |
|---|---|
| /card/oem/sinumerik/hmi/proj | User configurations (com files for "normal" user dialogs, i.e., everything except cycles support) |
| /card/user/sinumerik/cycles/proj | User configurations for user cycles support |
| /card/oem/sinumerik/cycles/proj<br>/card/oem/sinumerik/hmi/proj | Manufacturer configurations |
| /card/user/sinumerik/cycles/ico/icoxxx<br>/card/oem/sinumerik/cycles/ico/icoxxx<br>/card/user/sinumerik/hmi/ico/icoxxx<br>/card/oem/sinumerik/hmi/ico/icoxxx | Bitmaps |
| /card/user/sinumerik/hmi/cfg<br>/card/oem/sinumerik/hmi/cfg | Ini files |
| /card/user/sinumerik/hmi/ico/icoxxx<br>/card/oem/sinumerik/hmi/ico/icoxxx | Header icons |
| /card/user/sinumerik/cycles/lng/xxx<br>/card/oem/sinumerik/cycles/lng/xxx<br>/card/user/sinumerik/hmi/lng/xxx | The relevant texts (aluc.txt or aluctx.s0x), where xxx stands for the language |

On delivery, three USB drives and access to the CF card are set up in the PROGRAM operating area.

The following directory structure is set up on the USB memory:

\cycles

| | |
|---|---|
| \cycles\proj | (com files) |
| \cycles\prog | (customer cycles (.spf)) |
| \cycles\lng | (Language directory – contains only subdirectories) |
| \cycles\lng\xxx | (Language directory, e.g., ger, eng, etc. – one directory for each language. The **aluc.txt** file for the relevant language is stored here.) |
| \cycles\ico | (Display image directory – contains only subdirectories for each resolution) |
| \cycles\ico\ico640 | Directory for the display images of resolution 640*480 as **.bmp** or **.bin** |
| \cycles\ico\ico800 | |
| \cycles\ico\ico1024 | |

For dialogs, which are assigned to other operating areas:

\hmi\proj

\hmi\lng\...

\hmi\ico\...

In the PROGRAM operating area, the entire \cycles or \hmi directory is copied from the USB memory to the /user/sinumerik directory on the CF card.

## 6.3.2 HMI Advanced

### Overview

When using HMI Advanced, no control file entries are required.

The directories are searched for the configuration files in the specified order. If files of the same name are stored in different directories, then the first file detected according to the search strategy is interpreted.

### Test on a PG/standard PC

To test the configured dialogs on a PG/standard PC, the following conditions apply:

- You have installed the PC version of the HMI Advanced software on your PC/PG.
- The directory structure is the same as on HMI Advanced.
- The error log is generated in: \DH\COM.DIR\ERROR.COM

### Storing the alarm text files

Alarm text files are stored in the following directory: \DH\MB.DIR\

File name: ALUC_*xx*.COM

**References:** HMI Advanced Startup Manual

# 6.4 Search function on the HW platform common to both HMI systems

## 6.4.1 Search function principle

### Overview

HMI Embedded WIN32 searches the configuration files for "Expand user interface" on the same paths as HMI Advanced. The starting point is the data management path.

The search sequences described below are relevant for ShopMill/ShopTurn on NCU if HMI Advanced and HMI Embedded sl are both operated in parallel on a single hardware platform. In this case, "Expand user interface" accesses the same configuration files in ShopMill/ShopTurn on NCU running on HMI Embedded sl as on HMI Advanced.

### Supplementary Conditions

HMI Embedded WIN32 evaluates the "mmchome"property in section [DHSTART] in the DH.NI configuration file containing the root directory for the data management path. DH.INI must be stored in the BIN directory path on which MMC0 .EXE is started. The maximum length for the data management root directory path entry is 100 characters.

### Startup

On startup, HMI Embedded WIN32 uses the entries in the registry to ascertain where HMI Advanced is installed. Within the directory indicated in the registry, a search is made for the DH.INI file in the ..\user, ..\oem, ..\add_on, ..\mmc2 subdirectories in the sequence specified. If DH.INI cannot be found there, a search is carried out for the file in the currently selected directory of HMI Embedded WIN32.

#### Example

If HMI Advanced was found on F:\HMI\HMI Advanced, then a search for the DH.INI is made in the following sequence:

- F:\HMI\HMI Advanced\user
- F:\HMI\HMI Advanced\oem
- F:\HMI\HMI Advanced\add_on
- F:\HMI\HMI Advanced\mmc2

## Control variables for search sequences

Search sequence with HMI Advanced:

- CUS directory in the data management path specified in dh.ini
- CMA directory in the data management path specified in dh.ini
- CST directory in the data management path specified in dh.ini
- COM directory in the data management path specified in dh.ini

Directory on the RAMDISK:

*Filename.bin*

*Filename.bmp* .

## Bitmap name without path

In HMI Advanced, files with a .bin extension are not used

Search sequence if no archive is specified and the file name does not contain a path:

*Filename.bin* in the CUS directory on the data management path

*Filename.bmp* in the CUS directory on the data management path

*Filename.bin* in the *Filename.bi_* archive in the CUS directory on the data management path

*Filename.bmp* in the *Filename.bi_* archive in the CUS directory on the data management path

*Filename.bin* in the CUS\resolution directory on the data management path

*Filename.bmp* in the CUS\resolution directory on the data management path

*Filename.bin* in the *Filename.bi_* archive in the CUS\resolution directory on the data management path

*Filename.bmp* in the *Filename.bm_* archive in the CUS\resolution directory on the data management path

*Filename.bin* in the CMA directory on the data management path

*Filename.bmp* in the CMA directory on the data management path

*Filename.bin* in the *Filename.bi_* archive in the CMA directory on the data management path

*Filename.bmp* in the *Filename.bm_* archive in the CMA directory on the data management path

*Filename.bin* in the CUS\resolution directory on the data management path

*Filename.bmp* in the CMA\resolution directory on the data management path

*Filename.bin* in the *Filename.bi_* archive in the CMA\resolution directory on the data management path

*Filename.bmp* in the *Filename.bm_* archive in the CUS\resolution directory on the data management path

*Filename.bin* in the CST directory on the data management path

*Filename.bmp* in the CST directory on the data management path

*Filename.bin* in the *Filename.bi_* archive in the CST directory on the data management path

*Filename.bmp* in the *Filename.bm_* archive in the CST directory on the data management path

*Filename.bin* in the CST\resolution directory on the data management path

*Filename.bmp* in the CST\resolution directory on the data management path

*Filename.bin* in the *Filename.bi_* archive in the CST\resolution directory on the data management path

*Filename.bmp* in the *Filename.bm_* archive in the CST\resolution directory on the data management path

*Filename.bin* in the current directory (Bin directory)

*Filename.bmp* in the current directory (Bin directory)

*Filename.bin* in the *Filename.bi_* archive in the current directory (Bin directory)

*Filename.bmp* in the *Filename.bm_* archive in the current directory (Bin directory)

*Filename.bin* in the CUS.ARJ archive in the CUS directory on the data management path

*Filename.bmp* in the CUS.ARJ archive in the CUS directory on the data management path

*Filename.bin* in the CUS.ARJ archive in the CUS\resolution directory on the data management path

*Filename.bmp* in the CUS.ARJ archive in the CUS\resolution directory on the data management path

*Filename.bin* in the CUS.ARJ archive in the CMA directory on the data management path

*Filename.bmp* in the CUS.ARJ archive in the CMA directory on the data management path

*Filename.bin* in the CUS.ARJ archive in the CMA\resolution directory on the data management path

*Filename.bmp* in the CUS.ARJ archive in the CMA\resolution directory on the data management path

*Filename.bin* in the CUS.ARJ archive in the CST directory on the data management path

*Filename.bmp* in the CUS.ARJ archive in the CST  directory on the data management path

*Filename.bin* in the CUS.ARJ archive in the CST\resolution directory on the data management path

*Filename.bmp* in the CUS.ARJ archive in the CST\resolution directory on the data management path

Steps 29 through 40 are then repeated for CMA.ARJ

Filename.bin in archive CMA.ARJ in the CUS directory on the data management path

....

Filename.bmp in the CMA.ARJ archive in the CST\resolution directory on the data management path

Steps 41 through 52 are then repeated for CST.ARJ

Filename.bin in archive CST ARJ in the CUS directory on the data management path

....

Filename.bmp in archive CST.ARJ in the CST\resolution directory on the data management path

Filename.bin in archive CUS.ARJ in the current directory (Bin directory)

Filename.bmp in archive CUS.ARJ in the current directory (Bin directory)

Filename.bin in archive CMA.ARJ in the current directory (Bin directory)

Filename.bmp in archive CMA.ARJ in the current directory (Bin directory)

Filename.bin in archive CST.ARJ in the current directory (Bin directory)

Filename.bmp in archive CST.ARJ in the current directory (Bin directory)

Filename.bxx

Partial binary files, which have to be processed for reasons of compatibility with Real mode and when starting Protected mode. These files are only searched for in the currently selected directory.

## 6.4.2 Search function for COMMON.COM

### Storage location of COMMON.COM

The central control file COMMON.COM must be stored in one of the following NC directories:

- CUS: User cycles directory
- CMA: Manufacturer cycles directory
- CST: Standard cycles directory
- COM: Comment directory

The system searches through the directories for COMMON.COM in the following order. The first file with this name is evaluated.

In HMI Advanced, the COMMON.COM control file can also be stored on the data management path:

..\dh\cus.dir

..\dh\cma.dir

..\dh\cst.dir

..\dh\com.dir

---

**Note**

In order to activate the changes in this file, HMI Embedded WIN32 must be **restarted** in conjunction with HMI Advanced (ShopMill/ShopTurn).

---

## 6.4.3 Search function for images

### Search sequence

The expanded search function for images (bitmaps) in HMI Embedded sl can also be used for images in the "Expand user interface" system.

If HMI Embedded WIN32 (ShopMill/ShopTurn) is installed together with HMI Advanced on a single hardware platform, a relevant data management path can be preset.

If a data management path is defined, it is prioritized and processed before the currently selected directory in order that HMI Embedded WIN32 and HMI Advanced access the same images.

The new search mechanism also includes the directories of the data management path that are of relevance for "Expand user interface", and the possible image archives (cus.arj, cma.arj, cst.arj).

The search mechanism always searches for the individual files first, before searching possible archives. Consequently, the search sequence for images is as follows:

- Individual image before archive (.bin files are searched for before .bmp files). Once the search of the individual files has been completed, the archives containing a single file (.bi_, then .bm_) are searched.
- Bitmap name with path followed by bitmap name without path

### See also

Search function principle (Page 184)

# Appendix A

## A.1 List of start softkeys

### Start softkeys for ShopMill and ShopTurn

| ShopMill | SCxxx | Output dialog | |
|---|---|---|---|
| | SC818 | Machine Manual operating area (large dialog) | Horizontal SK 8 |
| | SC8181 | Machine Manual operating area (medium dialog) | Horizontal SK 8 |
| | SC8182 | Machine Manual operating area (small dialog) | Horizontal SK 8 |
| | SC8131 | Machine Manual-Workpiece Zero operating area | Vertical SK 1 |
| | SC8132 | Machine Manual-Workpiece Zero operating area | Vertical SK 2 |
| | SC8133 | Machine Manual-Workpiece Zero operating area | Vertical SK 3 |
| | SC8134 | Machine Manual-Workpiece Zero operating area | Vertical SK 4 |
| | SC8135 | Machine Manual-Workpiece Zero operating area | Vertical SK 5 |
| | SC8136 | Machine Manual-Workpiece Zero operating area | Vertical SK 6 |
| | SC8137 | Machine Manual-Workpiece Zero operating area | Vertical SK 7 |
| | SC8141 | Machine Manual-Measure Tool operating area | Vertical SK 1 |
| | SC8142 | Machine Manual-Measure Tool operating area | Vertical SK 2 |
| | SC8143 | Machine Manual-Measure Tool operating area | Vertical SK 3 |
| | SC8144 | Machine Manual-Measure Tool operating area | Vertical SK 4 |
| | SC8145 | Machine Manual-Measure Tool operating area | Vertical SK 5 |
| | SC8146 | Machine Manual-Measure Tool operating area | Vertical SK 6 |
| | SC8147 | Machine Manual-Measure Tool operating area | Vertical SK 7 |

| ShopMill | SCxxx | Output dialog | |
|---|---|---|---|
| | SC826 | Machine Auto operating area (large dialog) | Horizontal SK 6 |
| | SC8261 | Machine Auto operating area (medium dialog) | Horizontal SK 6 |
| | SC8262 | Machine Auto operating area (small dialog) | Horizontal SK 6 |
| | SC8426 | Program-Drilling operating area | Vertical SK 6 |
| | SC8436 | Program-Milling operating area | Vertical SK 6 |
| | SC8454 | Program-Miscellaneous operating area | Vertical SK 4 |
| | SC8951 | Program-Miscellaneous-Workpiece Zero operating area | Vertical SK 1 |
| | SC8952 | Program-Miscellaneous-Workpiece Zero operating area | Vertical SK 2 |
| | SC8953 | Program-Miscellaneous-Workpiece Zero operating area | Vertical SK 3 |
| | SC8954 | Program-Miscellaneous-Workpiece Zero operating area | Vertical SK 4 |
| | SC8955 | Program-Miscellaneous-Workpiece Zero operating area | Vertical SK 5 |
| | SC8956 | Program-Miscellaneous-Workpiece Zero operating area | Vertical SK 6 |
| | SC8957 | Program-Miscellaneous-Workpiece Zero operating area | Vertical SK 7 |
| | SC8961 | Program-Miscellaneous-Measure Tool operating area | Vertical SK 1 |
| | SC8962 | Program-Miscellaneous-Measure Tool operating area | Vertical SK 2 |
| | SC8963 | Program-Miscellaneous-Measure Tool operating area | Vertical SK 3 |
| | SC8964 | Program-Miscellaneous-Measure Tool operating area | Vertical SK 4 |
| | SC8965 | Program-Miscellaneous-Measure Tool operating area | Vertical SK 5 |
| | SC8966 | Program-Miscellaneous-Measure Tool operating area | Vertical SK 6 |
| | SC8967 | Program-Miscellaneous-Measure Tool operating area | Vertical SK 7 |
| | SC857 | Messages/Alarms operating area | Horizontal SK 7 |
| | SC858 | Messages/Alarms operating area | Horizontal SK 8 |
| | SC867 | Tools/Zero offsets operating area | Horizontal SK 7 |
| | SC8492 | Program-G-Code-Editor operating area | Horizontal SK 2 (contour cycles support) * |
| | SC8493 | Program-G-Code-Editor operating area | Horizontal SK 3 (drilling cycles support) * |
| | SC8494 | Program-G-Code-Editor operating area | Horizontal SK 4 (milling cycles support) * |

| ShopMill | SCxxx | Output dialog | |
|---|---|---|---|
| | SC8495 | Program-G-Code-Editor operating area | Horizontal SK 5 (turning cycles support) * |
| | SC8496 | Program-G-Code-Editor operating area | Horizontal SK 6* |
| | SC8406 | Program-G-Code-Editor operating area (expanded area) | Horizontal SK 6 (measuring cycles support) |
| | SC8407 | Program-G-Code-Editor operating area (expanded area) | Horizontal SK 7 (measuring cycles support) |

* These are Siemens dialogs.

| ShopTurn | SCxxx | Output dialog | |
|---|---|---|---|
| | SC818 | Machine Manual operating area (large dialog) | Horizontal SK 8 |
| | SC8181 | Machine Manual operating area (medium dialog) | Horizontal SK 8 |
| | SC8182 | Machine Manual operating area (small dialog) | Horizontal SK 8 |
| | SC8131 | Machine Manual-Workpiece Zero operating area | Vertical SK 1 |
| | SC8132 | Machine Manual-Workpiece Zero operating area | Vertical SK 2 |
| | SC8133 | Machine Manual-Workpiece Zero operating area | Vertical SK 3 |
| | SC8134 | Machine Manual-Workpiece Zero operating area | Vertical SK 4 |
| | SC8135 | Machine Manual-Workpiece Zero operating area | Vertical SK 5 |
| | SC8136 | Machine Manual-Workpiece Zero operating area | Vertical SK 6 |
| | SC8137 | Machine Manual-Workpiece Zero operating area | Vertical SK 7 |
| | SC8141 | Machine Manual-Measure Tool operating area | Vertical SK 1 |
| | SC8142 | Machine Manual-Measure Tool operating area | Vertical SK 2 |
| | SC8143 | Machine Manual-Measure Tool operating area | Vertical SK 3 |
| | SC8144 | Machine Manual-Measure Tool operating area | Vertical SK 4 |
| | SC8145 | Machine Manual-Measure Tool operating area | Vertical SK 5 |
| | SC8146 | Machine Manual-Measure Tool operating area | Vertical SK 6 |
| | SC8147 | Machine Manual-Measure Tool operating area | Vertical SK 7 |
| | SC826 | Machine Auto operating area (large dialog) | Horizontal SK 6 |

| ShopTurn | SCxxx | Output dialog | |
|---|---|---|---|
| | SC8261 | Machine Auto operating area (medium dialog) | Horizontal SK 6 |
| | SC8262 | Machine Auto operating area (small dialog) | Horizontal SK 6 |
| | SC8246 | Program-Drilling operating area | Vertical SK 6 |
| | SC9436 | Program-Turning operating area | Vertical SK 6 |
| | SC9456 | Program-Milling operating area | Vertical SK 6 |
| | SC8454 | Program-Miscellaneous operating area | Vertical SK 4 |
| | SC857 | Messages/Alarms operating area | Horizontal SK 7 |
| | SC858 | Messages/Alarms operating area | Horizontal SK 8 |
| | SC867 | Tools/Zero offsets operating area | Horizontal SK 7 |
| | SC8492 | Program-G-Code-Editor operating area | Horizontal SK 2 (contour cycles support) * |
| | SC8493 | Program-G-Code-Editor operating area | Horizontal SK 3 (drilling cycles support) * |
| | SC8494 | Program-G-Code-Editor operating area | Horizontal SK 4 (milling cycles support) * |
| | SC8495 | Program-G-Code-Editor operating area | Horizontal SK 5 (turning cycles support) * |
| | SC8496 | Program-G-Code-Editor operating area | Horizontal SK 6* |
| | SC8406 | Program-G-Code-Editor operating area (expanded area) | Horizontal SK 6 (turning measuring cycles support)* |
| | SC8407 | Program-G-Code-Editor operating area (expanded area) | Horizontal SK 7 (milling measuring cycles support)* |

* These are Siemens dialogs.

# A.2 List of colors

## Available colors

A standard color table is available for HMI Advanced and HMI Embedded sl for the purpose of configuring dialogs (subset of the respective standard colors):

| Number | Color |
|---|---|
| 1 | black |
| 2 | Red/brown |
| 3 | Dark green |
| 4 | Light gray |
| 5 | Dark gray |
| 6 | Blue |
| 7 | Red |
| 8 | brown |
| 9 | yellow |
| 10 | white |

The appearance of the colors may vary slightly in each HMI program.

## HMI Advanced

For bitmaps in HMI Advanced, the current color table of the shipped software must be used in the character program.

## HMI Embedded sl

For bitmaps in HMI Embedded sl, the current color table of the shipped software must be used in the character program. Color tables depend on the "New Fashion" option.

You will find the color tables in the tool box under
8x0d\examples_tools\wizard.bsp\hmi_emb\...

The file names indicate the application of each table:

- HMI_EMB_NEW_FASHION.PAL:

  This color palette is used for HMI Embedded sl with New Fashion.

  Color indices 160 to 231 are available.

- HMI_EMB_OLD_AND_NEW_FASHION.PAL:

  This color palette is used for HMI Embedded sl with both Old and New Fashion;
  the bitmaps generated with this color palette are identical in appearance for Old and New Fashion. Color indices 160, 163, 184, 187, 196, 199, 204, 205, 207, 217, 219, 220, 221, 223, 226 and 228 are available

The previous color table HMI_EMB.PAL is superseded by the tables specified above.
Only colors 160 through 231 can be used. This is the only way to ensure that images look the same on HMI Embedded sl and HMI Advanced.

**Activating a color table in Paint Shop Pro:**

- File → Open → ...*.bmp
- Colors → Open picture palette → ...*.PAL
- Apply palette by selecting the "Color Indices: Open" option.

### System colors

There is a choice of 10 colors available for the color of an element (text, input field, background, etc.). The system colors have been expanded and there is a difference between the old and new designs (e. g., the header color).

In order to achieve a demarcation between unique and **design-dependent** colors, it has been defined that the unique colors are found between 0 and 128. The design-dependent colors that have been newly introduced as part of this expansion, are defined from index 128.
This means, that also when the unique colors are expanded (up to 128), it is prevented that both color types are mixed.

### Newly defined colors

| Index | Color description | Color | |
|---|---|---|---|
| | | Old design | New design |
| 128 | System color active field | yellow | orange |
| 129 | Background color | Gray | Light gray |
| 130 | Header color (active) | yellow | Blue |
| 131 | Header font color (active) | black | white |

## A.3 List of accessible system variables

| Name | Index | Description |
|---|---|---|
| $A_DBB[x] | x=ByteNo | Data byte from/to PLC |
| $A_DBD[x] | x=Offset | Double data word (32bit) from/to PLC |
| $A_DBR[x] | x=Offset | Real data (32 bit) from/to PLC |
| $A_DBW[x] | x=Offset | Data word (16 bit) from/to PLC |
| $A_DLB[index] | index=Offset | Data byte in link area |
| $A_DLD[index] | index=Offset | Data access to double word in link area |
| $A_DLR[index] | index=Offset | Real data in link area |
| $A_DLW[index] | index=Offset | Data word in link area |
| $A_IN[x] | x=DigitalinputNo | Value of HW digital input |
| $A_INA[x] | x=AnaloginputNo | Value of HW analog input |

| Name | Index | Description |
| --- | --- | --- |
| $A_INCO[x] | x=InputNo | Comparator input NC |
| $A_INSE | | Safe Programmable Logic:<br>External input of the NCK I/O |
| $A_INSED | | Safe Programmable Logic:<br>Image of external NCK input |
| $A_INSEP | | Safe Programmable Logic:<br>External input of the PLC I/O |
| $A_INSEPD | | Safe Programmable Logic:<br>Image of external PLC inputs |
| $A_INSI | | Safe Programmable Logic:<br>Internal NCK input safety |
| $A_INSID | | Safe Programmable Logic:<br>Image of internal NCK input safety |
| $A_INSIP | | Safe Programmable Logic:<br>Internal PLC input safety |
| $A_LINK_TRANS_RATE | | Link transfer rate |
| $A_MARKERSI | | Safe Programmable Logic: NCK markers |
| $A_MARKERSIP | | Safe Programmable Logic:<br>Image of PLC markers |
| $A_OUT[x] | x=DigitaloutputNo | Value of HW digital output |
| $A_OUTA[x] | x=AnalogoutputNo | Value of HW analog output |
| $A_OUTSE | | Safe Programmable Logic:<br>External output of the NCK I/O |
| $A_OUTSED | | Safe Programmable Logic:<br>Image of external NCK output |
| $A_OUTSEP | | Safe Programmable Logic:<br>External output of the PLC I/O |
| $A_OUTSEPD | | Safe Programmable Logic:<br>Image of the external PLC output |
| $A_OUTSI | | Safe Programmable Logic:<br>Internal NCK output safety |
| $A_OUTSID | | Safe Programmable Logic:<br>Image of internal NCK output safety |
| $A_OUTSIP | | Safe Programmable Logic:<br>Internal PLC output 611D safety |
| $A_OUTSIPD | | Safe Programmable Logic:<br>Image of internal PLC output 611D safety |
| $A_TIMERSI | | Safe Programmable Logic: NCK timers |
| $A_PBB_IN[index] | index=Offset | IN data byte |
| $A_PBB_OUT[index] | index=Offset | OUT data byte |
| $A_PBD_IN[index] | index=Offset | IN double data word |
| $A_PBD_OUT[index] | index=Offset | OUT double data word |
| $A_PBR_IN[index] | index=Offset | IN real data |

| Name | Index | Description |
|---|---|---|
| $A_PBR_OUT[index] | index=Offset | OUT real data |
| $A_PBW_IN[index] | index=Offset | IN data word |
| $A_PBW_OUT[index] | index=Offset | OUT data word |
| $A_TC_FCT | | Command number |
| $A_TC_LFN | | Source location number |
| $A_TC_LFO | | Source location number |
| $A_TC_LTN | | Target location number |
| $A_TC_LTO | | Target location number |
| $A_TC_MFN | | Source magazine |
| $A_TC_MFO | | Source magazine number |
| $A_TC_MTN | | Target magazine number |
| $A_TC_MTO | | Target magazine number |
| $A_TC_STATUS | | Command status |
| $A_TC_THNO | | Number of toolholder |
| $A_TC_TNO | | T number |
| $A_TOOLMLN[x] | x=ToolNo T | Current location |
| $A_TOOLMN[x] | x=ToolNo T | Current magazine |
| $AA_COUP_ACT[x] | x=Spindle following | Current coupling status following spindle |
| $AA_COUP_OFFS[x] | x=Axis | Offset to leading axis/leading spindle, setpoint |
| $AA_COUP_OFFS[x] | x=Spindle | Position offset for synchronous spindle (setpoint) |
| $AA_CURR[x] | x=Axis | Actual current value of axis or spindle |
| $AA_DELT[x] | x=Axis | Drive-specific distance to go in WCS |
| $AA_DTBB[x] | x=Axis | Drive-specific path from start of block in basic coordinate system |
| $AA_DTBW[x] | x=Axis | Drive-specific path from start of block in WCS |
| $AA_DTEB[x] | x=Axis | Drive-specific path, end of block in basic coordinate system |
| $AA_DTEPB[x] | x=Axis | Drive-specific dist.-to-go infeed oscillation in basic coordinate system |
| $AA_DTEPW[x] | x=Axis | Drive-specific dist.-to-go infeed oscillation in WCS |
| $AA_DTEW[x] | x=Axis | Drive-specific path, end of block in WCS |
| $AA_EG_ACTIVE [a,b] | a = Following axis<br>b = Leading axis | EG coupling active |
| $AA_EG_AX[n,a] | n = Index leading axis<br>a = Following axis | EG leading axis number |
| $AA_EG_DENOM [a,b] | a = Following axis<br>b = Leading axis | EG coupling factor denominator |
| $AA_EG_NUM_LA[a] | a=Following axis | EG no. of leading axes |
| $AA_EG_NUMERA [a,b] | a = Following axis<br>b = Leading axis | EG coupling factor numerator |
| $AA_EG_SYN[a,b] | a = Following axis<br>b = Leading axis | EG synchronous position leading axis |
| $AA_EG_SYNCDIFF[a] | a=Axis identifier | EG synchronism deviation |

| Name | Index | Description |
|---|---|---|
| $AA_EG_SYNFA[a] | a=Following axis | EG synchronous position following axis |
| $AA_EG_TYPE[a,b] | a = Following axis<br>b = Leading axis | EG coupling type |
| $AA_ESR_ENABLE[a] | a=Axis | ESR axis enable |
| $AA_ESR_ENABLE[a] | a=Axis | ESR enable |
| $AA_ESR_STAT[a] | a=Axis | ESR status |
| $AA_ETRANS[x] | x=FrameNo | Offset of external frames |
| $AA_FXS[x] | x=Axis | Status after travel to fixed stop |
| $AA_IBN[x] | x=Axis | Toolholder actual value |
| $AA_IEN[x] | x=Axis | Holder for active tool relative to workpiece zero point |
| $AA_IM[x] | x=Axis | Toolholder |
| $AA_IW[x] | x=Axis | Toolholder position, setpoint |
| $AA_LEAD_P[x] | x=Axis | Real master value - position |
| $AA_LEAD_SP[x] | x=Axis | Simulated master value - position |
| $AA_LEAD_SV[x] | x=Axis | Simulated master value - velocity |
| $AA_LEAD_TYP[x] | x=Axis | Source of master value |
| $AA_LEAD_V[x] | x=Axis | Real master value - velocity |
| $AA_LOAD[x] | x=Axis | Drive load in % (for 611D only) |
| $AA_MM[x] | x=Axis | Measured value in MCS |
| $AA_MM1[x] | x=Axis | Access to measured value in MCS |
| $AA_MM2[x] | x=Axis | Access to measured value in MCS |
| $AA_MM3[x] | x=Axis | Access to measured value in MCS |
| $AA_MM4[x] | x=Axis | Access to measured value in MCS |
| $AA_MW[x] | x=Axis | Measured value in WCS |
| $AA_OFF[x] | x=Axis | Overlaid motion for programmed axis |
| $AA_OFF_LIMIT[x] | x=Axis | Drive-specific limit value reached, correction for $AA_OFF |
| $AA_OSCILL_REVERSE_POS1[x] | x=Axis | Current reversal position 1 oscill. in synchronous actions |
| $AA_OSCILL_REVERSE_POS2[x] | x=Axis | Current reversal position 2 oscill. in synchronous actions |
| $AA_OVR[x] | x=Axis | Drive-specific override for motion-synchronous actions |
| $AA_POWER[x] | x=Axis | Drive active power in [Watts] |
| $AA_S[x] | x=SpindleNo | Spindle speed, setpoint |
| $AA_SOFTENDN[x] | x=Axis | Software limit position, negative direction |
| $AA_SOFTENDP[x] | x=Axis | Software limit position, positive direction |
| $AA_STAT[x] | x=Axis | Axis status |
| $AA_SYNA_MEM | | Free memory space, motion synchronous actions |
| $AA_SYNC[x] | x=Axis | Coupling of the slave axis for master value coupling |
| $AA_TORQUE[x] | x=Axis | Drive torque setpoint in [Nm] |

| Name | Index | Description |
|---|---|---|
| $AA_TYP[x] | x=Axis | Axis type |
| $AA_VACTB[x] | x=Axis | Drive-specific feed, actual value |
| $AA_VACTM[x] | x=Axis | Drive-specific feed, setpoint |
| $AA_VACTW[x] | x=Axis | Drive-specific feed, actual value |
| $AA_VC[x] | x=Axis | Drive-specific feed,<br>additive path feed correction |
| $AC_ALARM_STAT | | ESR alarm status |
| $AC_AXCTSWA[CTn] | CTn=Axis container no. | Axis container status |
| $AC_DELT | | Distance to go, path WCS |
| $AC_DRF[x] | x=Axis | DRF value |
| $AC_DTBB | | Distance from start of block in basic coordinate system |
| $AC_DTBW | | Distance from block start in WCS |
| $AC_DTEB | | Distance from end of block in basic coordinate system |
| $AC_DTEPB | | Distance to go for oscillating infeed of basic coordinate system |
| $AC_DTEPW | | Distance to go for oscillating infeed in WCS |
| $AC_DTEW | | Distance from block end in WCS |
| $AC_FCT0[x] | x=PolynomNo | a0 coefficient nth polynomial for synchr. action |
| $AC_FCT1[x] | x=PolynomNo | a1 coefficient nth polynomial for synchr. action |
| $AC_FCT2[x] | x=PolynomNo | a2 coefficient nth polynomial for synchr. action |
| $AC_FCT3[x] | x=PolynomNo | a3 coefficient nth polynomial for synchr. action |
| $AC_FCTLL[x] | x=PolynomNo | Lower limit value nth polynomial for synchronized action |
| $AC_FCTUL[x] | x=PolynomNo | Upper limit value nth polynomial for synchronized action |
| $AC_FIFOx[y] | x=FIFONo (1-10)<br>y=ParameterNo | FIFO variables for synchronous actions |
| $AC_MARKER[x] | x=MarkerNo | Counter for motion synchronous actions |
| $AC_MEA | | Probe has responded |
| $AC_OVR | | Path override for synchronous actions |
| $AC_PARAM[x] | x=ParameterNo | Dyn. parameter for motion synchronization |
| $AC_PATHN | | Normalized path parameter |
| $AC_PLTBB | | Distance from start of block in basic coordinate system |
| $AC_PLTEB | | Distance from block end in basic coordinate system |
| $AC_PRESET[x] | x=Axis | PRESET value |
| $AC_PROG | | Program status |
| $AC_RETPOINT[x] | x=Axis | Return position on contour for repositioning |
| $AC_SDIR[x] | x=SpindleNo | Rotat. status |
| $AC_SMODE[x] | x=SpindleNo | Spindle mode |
| $AC_STAT | | Channel status |

| Name | Index | Description |
|---|---|---|
| $AC_TIME | | Time from the start of the block in seconds (including the times for the internally generated intermediate blocks) |
| $AC_TIMES | | Time from the start of the block in seconds (without times for the internally generated intermediate blocks) |
| $AC_TIMEC | | Time from the start of the block in interpolation clock cycles (including clock cycles for the internally generated intermediate blocks) |
| $AC_TIMESC | | Time from the start of the block in interpolation clock cycles (without the clock cycles for internally generated intermediate blocks) |
| $AC_TIMER[x] | x=TimerNo | Timer location in seconds |
| $AC_VACTB | | Interpolation feedrate, setpoint |
| $AC_VACTW | | Path velocity in WCS |
| $AC_VC | | Additive path feed correction for synchronous actions |
| $AN_AXCTAS[n] | | Actual axis container address (rotary position) |
| $AN_AXCTSWA[CTn] | CTn=Axis container no. | Axis container rotation active |
| $AN_ESR_TRIGGER | | ESR start signal |
| $AN_MARKER[x] | x=MarkerNo | Marker variable for motion synchronous actions |
| $MC_DISPLAY_AXIS | Bits 16 -31 | Machine axis identifier |
| $MC_DISPLAY_AXIS | Bits 0 -15 | Identifier for geometry/auxiliary axis |
| $MC_MM_NUM_BASE_FRAMES | | Number of base frames in channel |
| $MN_EXTERN_LANGUAGE<br>$MN_EXTERN_CNC_SYSTEM | and | CNC system language |
| $MN_MAX_CUTTING_EDGE_NO | | Max. value D number |
| $MN_MAX_CUTTING_EDGE_ PER_TOOL | | Max. number of edges per tool |
| $MN_MAX_SUMCORR_ PERCUTTING_EDGE | | Max. number of total offsets per cutting edge |
| $MN_MM_KIND_OF_SUMCORR | | Properties of total offsets in NCK |
| $MN_MM_NUM_CC_MAGAZINE_ PARAM | | Number of parameters for one tool magazine |
| $MN_MM_NUM_CC_MAGLOC_ PARAM | | Number of parameters for one tool magazine location |
| $MN_MM_NUM_CC_MON_ PARAM | | Number of parameters of monitoring user data of one tool edge |
| $MN_MM_NUM_GLOBAL_BASE_ FRAMES | | Number of channel-dependent basic frames |
| $MN_MM_NUM_SUMCORR | | Total number of total offsets in NCK |
| $MN_MM_NUM_TOOL_ADAPTER | | Max. number of traversed tool adapter data blocks |
| $MN_MM_TOOL_MANAGEMENT_ MASK | | Settings NCK tool management |
| $P_UBFR[ x ,MI] | x=FrameNo | Mirroring of settable frame |

| Name | Index | Description |
|---|---|---|
| $P_UBFR[x,RT] | x=FrameNo | Rotation of settable frame |
| $P_UBFR[x,SC] | x=FrameNo | Scaling factor of settable frame |
| $P_UBFR[x,SI] | x=FrameNo | Fine offset for frames |
| $P_UBFR[x,TR] | x=FrameNo | Translation of settable frame |
| $P_CHBFRMASK | | Active channel-specific basic frames |
| $P_EG_BC[a] | | EG block change criterion |
| $P_NCBFRMASK | | Active channel-independent basic frames |
| $P_OFFN | | Offset normal |
| $P_PFRAME[x,TR]<br>$P_ACTFRAME<br>$P_IFRAME | or<br>or | Compilation of active frame |
| $P_TOOL | | Number of active tool edge |
| $P_TOOLL[1] | | Active tool length 1 |
| $P_TOOLL[2] | | Active tool length 2 |
| $P_TOOLL[3] | | Active tool length 3 |
| $P_TOOLND[x] | x=ToolNo | Number of cutting edges |
| $P_TOOLNO | | Number of active tool |
| $P_TOOLR | | Active tool radius |
| $P_UIFR[x,y,MI] | x=FrameNo, y=Axis | Mirroring of settable frame |
| $P_UIFR[x,y,RT] | x=FrameNo, y=Axis | Rotation of settable frame |
| $P_UIFR[x,y,SC] | x=FrameNo, y=Axis | Scaling factor of settable frame |
| $P_UIFR[x,y,SI] | x=FrameNo, y=Axis | Fine offset for frames |
| $P_UIFR[x,y,TR] | x=FrameNo, y=Axis | Translation of settable frame |
| $P_UIFRNUM | | Index of active set frame |
| $R[x] | x=ParameterNo | R parameters |
| $SC_PA_ACTIV_IMMED[x] | x=Number protection zone | Immediately active after referencing |
| $SC_PA_CENT_ABS[x,0] | x=Number protection zone | Abscissa of arc center of 1st contour element |
| $SC_PA_CENT_ABS[x,1] | x=Number protection zone | Abscissa of arc center of 2nd contour element |
| $SC_PA_CENT_ABS[x,2] | x=Number protection zone | Abscissa of arc center of 3rd contour element |
| $SC_PA_CENT_ABS[x,3] | x=Number protection zone | Abscissa of arc center of 4th contour element |
| $SC_PA_CENT_ABS[x,4] | x=Number protection zone | Abscissa of arc center of 5th contour element |
| $SC_PA_CENT_ABS[x,5] | x=Number protection zone | Abscissa of arc center of 6th contour element |
| $SC_PA_CENT_ABS[x,6] | x = Number protection zone | Abscissa of arc center of 7th contour element |
| $SC_PA_CENT_ABS[x,7] | x=Number protection zone | Abscissa of arc center of 8th contour element |
| $SC_PA_CENT_ABS[x,8] | x=Number protection zone | Abscissa of arc center of 9th contour element |
| $SC_PA_CENT_ABS[x,9] | x=Number protection zone | Abscissa of arc center of 10th contour element |
| $SC_PA_CENT_ORD[x,0] | x=Number protection zone | Ordinate of arc center of 1st contour element |
| $SC_PA_CENT_ORD[x,1] | x=Number protection zone | Ordinate of arc center of 2nd contour element |
| $SC_PA_CENT_ORD[x,2] | x=Number protection zone | Ordinate of arc center of 3rd contour element |
| $SC_PA_CENT_ORD [x,3] | x=Number protection zone | Ordinate of arc center of 4th contour element |
| $SC_PA_CENT_ORD[x,4] | x=Number protection zone | Ordinate of arc center of 5th contour element |
| $SC_PA_CENT_ORD[x,5] | x=Number protection zone | Ordinate of arc center of 6th contour element |

| Name | Index | Description |
| --- | --- | --- |
| $SC_PA_CENT_ORD[x,6] | x=Number protection zone | Ordinate of arc center of 7th contour element |
| $SC_PA_CENT_ORD[x,7] | x=Number protection zone | Ordinate of arc center of 8th contour element |
| $SC_PA_CENT_ORD[x,8] | x=Number protection zone | Ordinate of arc center of 9th contour element |
| $SC_PA_CENT_ORD[x,9] | x=Number protection zone | Ordinate of arc center of 10th contour element |
| $SC_PA_CONT_ABS[x,0] | x=Number protection zone | Abscissa of end point of 1st contour element |
| $SC_PA_CONT_ABS[x,1] | x=Number protection zone | Abscissa of end point of 2nd contour element |
| $SC_PA_CONT_ABS[x,2] | x=Number protection zone | Abscissa of end point of 3rd contour element |
| $SC_PA_CONT_ABS[x,3] | x=Number protection zone | Abscissa of end point of 4th contour element |
| $SC_PA_CONT_ABS[x,4] | x=Number protection zone | Abscissa of end point of 5th contour element |
| $SC_PA_CONT_ABS[x,5] | x=Number protection zone | Abscissa of end point of 6th contour element |
| $SC_PA_CONT_ABS[x,6] | x=Number protection zone | Abscissa of end point of 7th contour element |
| $SC_PA_CONT_ABS[x,7] | x=Number protection zone | Abscissa of end point of 8th contour element |
| $SC_PA_CENT_ABS[x,8] | x=Number protection zone | Abscissa of end point of 9th contour element |
| $SC_PA_CONT_ABS[x,9] | x=Number protection zone | Abscissa of end point of 10th contour element |
| $SC_PA_CONT_NUM[x] | x=Number protection zone | Number of valid contour elements |
| $SC_PA_CONT_ORD[x,0] | x=Number protection zone | Ordinate of end point of 1st contour element |
| $SC_PA_CONT_ORD[x,1] | x=Number protection zone | Ordinate of end point of 2nd contour element |
| $SC_PA_CONT_ORD[x,2] | x=Number protection zone | Ordinate of end point of 3rd contour element |
| $SC_PA_CONT_ORD[x,3] | x=Number protection zone | Ordinate of end point of 4th contour element |
| $SC_PA_CONT_ORD[x,4] | x=Number protection zone | Ordinate of end point of 5th contour element |
| $SC_PA_CONT_ORD[x,5] | x=Number protection zone | Ordinate of end point of 6th contour element |
| $SC_PA_CONT_ORD[x,6] | x=Number protection zone | Ordinate of end point of 7th contour element |
| $SC_PA_CONT_ORD[x,7] | x=Number protection zone | Ordinate of end point of 8th contour element |
| $SC_PA_CONT_ORD[x,8] | x=Number protection zone | Ordinate of end point of 9th contour element |
| $SC_PA_CONT_ORD[x,9] | x=Number protection zone | Ordinate of end point of 10th contour element |
| $SC_PA_CONT_TYP[x,0] | x=Number protection zone | Contour type of 1st contour element |
| $SC_PA_CONT_TYP[x,1] | x=Number protection zone | Contour type of 2nd contour element |
| $SC_PA_CONT_TYP[x,2] | x=Number protection zone | Contour type of 3rd contour element |
| $SC_PA_CONT_TYP[x,3] | x=Number protection zone | Contour type of 4th contour element |
| $SC_PA_CONT_TYP[x,4] | x=Number protection zone | Contour type of 5th contour element |
| $SC_PA_CONT_TYP[x,5] | x=Number protection zone | Contour type of 6th contour element |
| $SC_PA_CONT_TYP[x,6] | x=Number protection zone | Contour type of 7th contour element |
| $SC_PA_CONT_TYP[x,7] | x=Number protection zone | Contour type of 8th contour element |
| $SC_PA_CONT_TYP[x,8] | x=Number protection zone | Contour type of 9th contour element |
| $SC_PA_CONT_TYP[x,9] | x=Number protection zone | Contour type of 10th contour element |
| $SC_PA_LIM_3DIM[x] | x=Number protection zone | Limitation of protection zone, applicate |
| $SC_PA_MINUS_LIM[x] | x=Number protection zone | Lower limit of protection zone, applicate |
| $SC_PA_ORI[x] | x=Number protection zone | Plane assignment of protection zone |
| $SC_PA_PLUS_LIM[x] | x=Number protection zone | Upper limit of protection zone, applicate |
| $SC_PA_T_W[x] | x=Number protection zone | Workpiece or tool-related protection zone |
| $SN_PA_ACTIV_ IMMED[x] | x=Number protection zone | Immediately active after referencing |
| $SN_PA_CENT_ABS[x,0] | x=Number protection zone | Abscissa of arc center of 1st contour element |

| Name | Index | Description |
|---|---|---|
| $SN_PA_CENT_ABS[x,1] | x=Number protection zone | Abscissa of arc center of 2nd contour element |
| $SN_PA_CENT_ABS [x,2] | x=Number protection zone | Abscissa of arc center of 3rd contour element |
| $SN_PA_CENT_ABS[x,3] | x=Number protection zone | Abscissa of arc center of 4th contour element |
| $SN_PA_CENT_ABS[x,4] | x=Number protection zone | Abscissa of arc center of 5th contour element |
| $SN_PA_CENT_ABS[x,5] | x=Number protection zone | Abscissa of arc center of 6th contour element |
| $SN_PA_CENT_ABS[x,6] | x=Number protection zone | Abscissa of arc center of 7th contour element |
| $SN_PA_CENT_ABS[x,7] | x=Number protection zone | Abscissa of arc center of 8th contour element |
| $SN_PA_CENT_ABS[x,8] | x=Number protection zone | Abscissa of arc center of 9th contour element |
| $SN_PA_CENT_ABS[x,9] | x=Number protection zone | Abscissa of arc center of 10th contour element |
| $SN_PA_CENT_ORD[x,0] | x=Number protection zone | Ordinate of arc center of 1st contour element |
| $SN_PA_CENT_ORD[x,1] | x=Number protection zone | Ordinate of arc center of 2nd contour element |
| $SN_PA_CENT_ORD [x,2] | x=Number protection zone | Ordinate of arc center of 3rd contour element |
| $SN_PA_CENT_ORD[x,3] | x=Number protection zone | Ordinate of arc center of 4th contour element |
| $SN_PA_CENT_ORD[x,4] | x=Number protection zone | Ordinate of arc center of 5th contour element |
| $SN_PA_CENT_ORD[x,5] | x=Number protection zone | Ordinate of arc center of 6th contour element |
| $SN_PA_CENT_ORD[x,6] | x=Number protection zone | Ordinate of arc center of 7th contour element |
| $SN_PA_CENT_ORD[x,7] | x=Number protection zone | Ordinate of arc center of 8th contour element |
| $SN_PA_CENT_ORD[x,8] | x=Number protection zone | Ordinate of arc center of 9th contour element |
| $SN_PA_CENT_ORD[x,9] | x=Number protection zone | Ordinate of arc center of 10th contour element |
| $SN_PA_CONT_ABS[x,0] | x=Number protection zone | Abscissa of end point of 1st contour element |
| $SN_PA_CONT_ABS[x,1] | x=Number protection zone | Abscissa of end point of 2nd contour element |
| $SN_PA_CONT_ABS[x,2] | x=Number protection zone | Abscissa of end point of 3rd contour element |
| $SN_PA_CONT_ABS [x,3] | x=Number protection zone | Abscissa of end point of 4th contour element |
| $SN_PA_CONT_ABS[x,4] | x=Number protection zone | Abscissa of end point of 5th contour element |
| $SN_PA_CONT_ABS[x,5] | x=Number protection zone | Abscissa of end point of 6th contour element |
| $SN_PA_CONT_ABS[x,6] | x=Number protection zone | Abscissa of end point of 7th contour element |
| $SN_PA_CONT_ABS[x,7] | x=Number protection zone | Abscissa of end point of 8th contour element |
| $SN_PA_CONT_ABS[x,8] | x=Number protection zone | Abscissa of end point of 9th contour element |
| $SN_PA_CONT_ABS[x,9] | x=Number protection zone | Abscissa of end point of 10th contour element |
| $SN_PA_CONT_NUM[x] | x=Number protection zone | Number of valid contour elements |
| $SN_PA_CONT_ORD[x,0] | x=Number protection zone | Ordinate of end point of 1st contour element |
| $SN_PA_CONT_ORD[x,1] | x=Number protection zone | Ordinate of end point of 2nd contour element |
| $SN_PA_CONT_ORD[x,2] | x=Number protection zone | Ordinate of end point of 3rd contour element |
| $SN_PA_CONT_ORD[x,3] | x=Number protection zone | Ordinate of end point of 4th contour element |
| $SN_PA_CONT_ORD[x,4] | x=Number protection zone | Ordinate of end point of 5th contour element |
| $SN_PA_CONT_ORD[x,5] | x=Number protection zone | Ordinate of end point of 6th contour element |
| $SN_PA_CONT_ORD[x,6] | x=Number protection zone | Ordinate of end point of 7th contour element |
| $SN_PA_CONT_ORD[x,7] | x=Number protection zone | Ordinate of end point of 8th contour element |
| $SN_PA_CONT_ORD[x,8] | x=Number protection zone | Ordinate of end point of 9th contour element |
| $SN_PA_CONT_ORD[x,9] | x=Number protection zone | Ordinate of end point of 10th contour element |
| $SN_PA_CONT_TYP[x,0] | x=Number protection zone | Contour type of 1st contour element |
| $SN_PA_CONT_TYP[x,1] | x=Number protection zone | Contour type of 2nd contour element |

| Name | Index | Description |
|---|---|---|
| $SN_PA_CONT_TYP[x,2] | x=Number protection zone | Contour type of 3rd contour element |
| $SN_PA_CONT_TYP[x,3] | x=Number protection zone | Contour type of 4th contour element |
| $SN_PA_CONT_TYP[x,4] | x=Number protection zone | Contour type of 5th contour element |
| $SN_PA_CONT_TYP[x,5] | x=Number protection zone | Contour type of 6th contour element |
| $SN_PA_CONT_TYP[x,6] | x=Number protection zone | Contour type of 7th contour element |
| $SN_PA_CONT_TYP[x,7] | x=Number protection zone | Contour type of 8th contour element |
| $SN_PA_CONT_TYP[x,8] | x=Number protection zone | Contour type of 9th contour element |
| $SN_PA_CONT_TYP[x,9] | x=Number protection zone | Contour type of 10th contour element |
| $SN_PA_LIM_3DIM[x] | x=Number protection zone | Limitation of protection zone, applicate |
| $SN_PA_MINUS_LIM[x] | x=Number protection zone | Lower limit of protection zone, applicate |
| $SN_PA_ORI[x] | x=Number protection zone | Plane assignment of protection zone |
| $SN_PA_PLUS_LIM[x] | x=Number protection zone | Upper limit of protection zone, applicate |
| $SN_PA_T_W[x] | x=Number protection zone | Workpiece or tool-related protection zone |
| $TC_ADPT ... | | Adapter data |
| $TC_ADPTx<br>$TC_ADPTT | x=1 to 3 | Number of parameters per adapter |
| $TC_DPCE | | Transformed edge correction value |
| $TC_DPCx[y,z] | x=ParamNo<br>y=ToolNo, z=EdgeNo | User-defined tool edge parameter |
| $TC_DPx[y,z] | x=ParamNo<br>y=ToolNo, z=EdgeNo | Edge correction value |
| $TC_DPx[y,z] | x=ParamNo<br>y=ToolNo, z=EdgeNo | Transformed edge correction value |
| $TC_ECP ... | | Transformed location-dependent setup correction |
| $TC_MAMP3 | | Wear compound strategy |
| $TC_MAP1 | | Type of magazine |
| $TC_MAP2 | | Identifier of the magazine |
| $TC_MAP3 | | Status of magazine |
| $TC_MAP4 | | Link 1 between magazine and next magazine |
| $TC_MAP5 | | Link 2 between magazine and preceding magazine |
| $TC_MAP6 | | Magazine dimension |
| $TC_MAP9 | | Number of active wear group |
| $TC_MAPCx[y] | x=ParameterNo<br>y=MagazineNo | Magazine user data for a tool magazine |
| $TC_MOP1(x,y) … $TC_MOP15(x,y) | x=ToolNo<br>y=EdgeNo | Monitoring data for each tool edge |
| $TC_MOPCx[y,z] | x=ParamNo,<br>y=T number<br>z=Edge | Magazine user data for a tool edge |
| $TC_MPPCx[y,z] | x= ParamNo<br>y= MagazineNo<br>z= MagPlaceNo | Magazine location user data for a tool magazine |

| Name | Index | Description |
|---|---|---|
| $TC_MPPx | x=1,...7 | No. of parameters per magazine location |
| $TC_SCP... | | Transformed location-dependent wear correction |
| $TC_SCP... | | Location-dependent wear correction |
| $TC_SCPx | x=13,...21,....71 | Number of total offset parameters per total offset block |
| $TC_TP1 | | DUPLO number |
| $TC_TP10 | | Tool search type for replacement tools |
| $TC_TP11 | | Tool information for HMI |
| $TC_TP2 | | Tool designation |
| $TC_TP3 | | Size to left in half locations |
| $TC_TP4 | | Size to right in half locations |
| $TC_TP5 | | Size upwards in half locations |
| $TC_TP6 | | Size downwards in half locations |
| $TC_TP7 | | Magazine location type |
| $TC_TP8 | | Tool status |
| $TC_TP9 | | Tool monitoring method |
| $TC_TPCx[y] | x=ParameterNo<br>y = ToolNo | User-defined tool parameter |
| $TC_TPG1 | | Spindle number |
| $TC_TPG2 | | Chaining rule |
| $TC_TPG3 | | Minimum wheel diameter |
| $TC_TPG4 | | Minimum grinding wheel width |
| $TC_TPG5 | | Current width of grinding wheel |
| $TC_TPG6 | | Maximum speed of grinding wheel |
| $TC_TPG7 | | Maximum peripheral speed of grinding wheel (SUG) |
| $TC_TPG8 | | Angle of inclination of inclined grinding wheel |
| $TC_TPG9 | | Compensation parameter for GWPS |
| $VA_COUP_OFFS[x] | x=Axis | Offset to leading axis / leading spindle, actual value |
| $VA_IS[x] | x=Axis | Safe actual position of axis |
| $VA_VACTM[x] | x=Axis | Actual-value, load-end axis velocity in MCS system |

# A.4 List of PI services

## Programming

| | |
|---|---|
| Syntax | **PI_SERVICE***(service, n parameters)* |
| Service | PI service identifier |
| n parameters | List of PI service parameters.<br>Individual parameters are separated by commas. |

| Service<br>Parameter | Explanation<br>Example |
|---|---|
| _N_ASUP_ | An interrupt number in the specified channel is assigned to a parts program stored in the NCK (identified by path name and program name). This PI service is identical to program instruction 'SETINT'. |
| | *Par1* Interrupt number (0 – 8)<br>*Par2* Priority (0 – 8)<br>*Par3* Liftfast (0, 1)<br>*Par4* Block synchronization (0, 1)<br>*Par5* Path data with max. 32 positions |
| | Assignment of interrupt 5 to program MPF_DIR/TEST_MPF in the current channel. The interrupt has priority 3 and is executed without rapid lift on the<br>*PI_SERVICE("_N_ASUP__",5, 3, 0, 0,"/_N_MPF_DIR/_N_TEST_MPF")* |
| _N_CANCEL | All alarms with the "Cancel Alarm" classification can be acknowledged with this command. Individual acknowledgment for specific alarms is not possible. |
| | --- |
| | All alarms with the "Cancel Alarm" classification are reset.<br>*PI_SERVICE("_N_CANCEL“)* |
| _N_CRCEDN | Create tool edge by specifying the cutting edge number. If the T number of an existing tool is entered in parameter "T Number" of this service, then a tool edge for the existing tool is created (in this case, the parameter "D number", i.e., the number of the edge to be created, has a value range of 1-9). If a positive T number is specified as a parameter and the tool for the T number entered does not exist, then the PI service is aborted.<br>If a value of 0 is entered as the T number (model of absolute D numbers), then the D number values can range from 1 – 31999. The new cutting edge is set up with the specified D number.<br>If the specified cutting edge already exists, then the PI service is aborted in both cases. |
| | *Par1* T number<br>*Par2* D number<br>T number == 0 ==> 1 – 3199<br>T number > 0 ==> 1 – 9 |

| Service | Explanation |
|---|---|
| Parameter | Example |
| | A tool edge with number 3 is created for the tool with number 17 in the current TO area.<br>*PI_SERVICE("_N_CRCEDN",17,3)* |
| _N_CREACE | Create a new tool edge for a specified tool. The next-higher available D number is automatically set up. The following blocks in the active file system are affected by this PI service:<br>Tool offsets TO: The relevant tool edge (with ZERO content) is set up<br>Monitoring data TS: (If available) the relevant tool edge (with ZERO content) is set up<br>User edge data TUE: (If available) the relevant tool edge (with ZERO content) is set up<br>(SW releases NCK < 10.x) |
| | *Par1* Tool number 1 to 31999 |
| | After the tool with number 55 has been set up in T area 1, a further 2 edges are created for it. The tool therefore now possesses a total of 3 edges.<br>*PI_SERVICE("_N_CREATO",55)*<br>*PI_SERVICE("_N_CREACE",55)*<br>*PI_SERVICE("_N_CREACE",55)* |
| _N_CREATO | Create a tool with a specific T number.<br>The following blocks in the active file system are affected by this PI service:<br>Tool directory TV: The tool is entered as an existing tool.<br>Tool offsets TO: The first tool edge D1 (with ZERO content) is set up.<br>User edge data TUE: (If available) the first tool edge D1 (with ZERO content) is set up.<br>User tool data TU: (If available) an empty data block is made available for the tool. |
| | *Par1* Tool number 1 to 31999 |
| | After the tool with number 55 has been set up in T area 1, a further 2 edges are created for it. The tool therefore now possesses a total of 3 edges.<br>*PI_SERVICE("_N_CREATO",55)*<br>*PI_SERVICE("_N_CREACE",55)*<br>*PI_SERVICE("_N_CREACE",55)* |
| _N_DELECE | Delete a cutting edge:<br>If the T number of an existing tool is entered in parameter "T Number" of this PI service, then the tool edge for the relevant tool is deleted (in this case, the parameter "D number, i.e., the number of edge to be deleted, has a value range of 1-9). If a positive T number is specified as a parameter and the tool for the T number entered does not exist, then the PI service is aborted.<br>If a value of 0 is entered as the T number (model of absolute D numbers), then the D number values can range from 1-31999. If the specified cutting edge does not exist, then the PI service is aborted in both cases. |

| Service<br>Parameter | Explanation<br>Example |
|---|---|
| | *Par1* T number of the tool for which the associated tool cutting edge is to be deleted.<br>A setting of 0 indicates that there is no reference to a specific tool (absolute D number).<br>*Par2* Cutting edge number of tool cutting edge that is to be deleted.<br>Range of values:<br>T number == 0 ⇒ 1 – 31999<br>T number > 0 ⇒ 1 – 9 |
| | The tool edge with number 3 for the tool with number 17 in the current TO area is deleted:<br>*PI_SERVICE("_N_DELECE",17,3)* |
| _N_DELETO | Deletes the tool with all tool edges in all data blocks in which it is stored. The tool is also deleted in the following blocks (if available): TO, TU, TUE, TV, TG (type 400), TD, TS. |
| | *Par1* Tool number 1 to 31999 |
| | Deletion of tool with T number 21 in the current T area.<br>*PI_SERVICE("_N_DELETO",21)* |
| | *Par1* Identifier for search mode<br>1: Block search without calculation.<br>2: Block search with calculation<br>3: Search including the main block analysis |
| | Start search with calculation in current channel.<br>To start the PI service in a meaningful manner, the data structure for the block search (block SPRAF ; addressing on HMI Embedded sl with variable services via /Channel/Search/..) must be filled in beforehand.<br>*PI_SERVICE("_N_FINDBL",2)* |
| _N_LOGIN_ | A password, which sets the current access level, is sent to the NCK. |
| | *Par1* Password (precisely 8 characters; if there are fewer than 8 characters, blanks must be added) |
| | Transfers a password to the NCK, thus setting another access level.<br>*PI_SERVICE("_N_LOGIN_","TESTWORD")* |
| _N_LOGOUT | The current access level is reset. |
| | --- |
| | The current access level is reset.<br>*PI_SERVICE("_N_LOGOUT")* |
| _N_SETUFR | The SYSTEM OR USER variables 'linShift', 'mirrorImgActive', 'rotation' and 'scaleFact' in channel-specific data block FU can be used to define up to 8 zero offsets per channel. PI service _N_SETUFR must be called in order to activate these user-defined zero offsets. |
| | --- |
| | Activation of a user frame.<br>*PI_SERVICE("_N_SETUFR")* |

# List of abbreviations

# B

## B.1 Abbreviations

| | |
|---|---|
| A | Output |
| ASCII | American Standard Code for Information Interchange<br>American coding standard for the exchange of information |
| BAG | Mode group |
| BP | Basic program |
| CAD | Computer-Aided Design |
| CNC | Computerized Numerical Control Computerized numerical control |
| CR | Carriage Return |
| DAC | Digital-to-Analog Converter |
| DB | Data Block in the PLC |
| DBB | Data Block Byte in the PLC |
| DBW | Data Block Word in the PLC |
| DBX | Data block bit in the PLC |
| DDE | Dynamic Data Exchange: Dynamic Data Exchange |
| DIN | Deutsche Industrie Norm (German Industry Standard) |
| DIR | DIRectory: List |
| DOS | Disk Operating System |
| DPM | Dual-Port Memory |
| DRAM | Dynamic Random Access Memory |
| DRF | Differential Resolver Function: Differential revolver function (handwheel) |
| DRY | DRY run: Dry run feedrate |
| DW | Data word |
| E | Input |
| EG | Expansion unit |
| ESR | Extended Stop and Retract |
| FD | Feed Drive |
| FIFO | First In - First Out: Method of storing and retrieving data in a memory. |
| FRAME | Data block (FRAME) |
| GUD | Global User Data Global user data |
| GWPS | Grinding wheel surface speed |
| Hardware | Hardware |
| HD | Hard Disk Hard disk |
| HMI | Human Machine Interface: Controller user interface |

| | |
|---|---|
| IBN | Start-up |
| ICA | Interpolatory Compensation Interpolatory compensation |
| INC | Increment: Increment |
| INI | INItializing data Initializing data |
| IPO | Interpolator |
| ISO | International Standard Organization |
| JOG | JOGging: Setup mode |
| K1 .. C4 | Channel 1 to channel 4 |
| $K_v$ | Servo gain factor |
| LED | Light-Emitting Diode: Light-emitting diode |
| LF | Line Feed |
| LUD | Local User Data: Local user data |
| MB | Megabyte |
| MCP | Machine Control Panel Machine control panel (→ MCP) |
| MCP | Machine control panel |
| MCS | Machine coordinate system |
| MD | Machine data |
| MDI | Manual Data Input: Manual input |
| MLFB | Machine-readable product designation |
| MPF | Main Program File: NC part program (main program) |
| MPI | Multi-Point Interface Multiport Interface |
| MSD | Main Spindle Drive |
| NC | Numerical Control: Numerical control |
| NCK | Numerical Control Kernel: NC kernel with block preparation, traversing range, etc. |
| NCU | Numerical Control Unit: NCK hardware unit |
| OEM | Original Equipment Manufacturer |
| OP | Operator Panel: Operating equipment |
| OPI | Operator Panel Interface |
| PCMCIA | Personal Computer Memory Card International Association: Standards body for memory cards |
| PCU | Programmable Control Unit |
| PG | Programming device |
| PLC | Programmable Logic Control: |
| REF | REFerence point approach function |
| REPOS | REPOSition function |
| ROV | Rapid Override: Input correction |
| RPA | R-Parameter Active: Memory area on the NCK for R parameter numbers |
| SBL | Single Block: Single BLock |
| SD | Setting Data |
| SDB | System Data Block |
| SEA | Setting Data Active: Identifier (file type) for setting data |
| SK | Softkey |
| SKP | SKiP: Skip block |
| SPF | SubProgram File: Subprogram |

| | |
|---|---|
| SRAM | Static RAM (non-volatile) |
| SW | Software |
| SYF | SYstem Files System files |
| TC | Tool change |
| TEA | Testing Data Active: Identifier for machine data |
| TO | Tool Offset Tool offset |
| TO | Tool offset |
| TOA | Tool Offset Active: Identifier (file type) for tool offsets |
| UFR | User frame |
| WCS | Workpiece coordinate system |
| ZO | Zero point shift |
| ZOA | Zero Offset Active: Identifier (file type) for work offset data |

# Index

## R



## S



## T



## U



## V



## W

SIEMENS

SINUMERIK 840D sl

# Creating foreign language texts

Commissioning Manual



Valid for
SINUMERIK 840D sl/840DE sl control system

| Software | Version |
| --- | --- |
| NCU System Software | 1.4 |
| with HMI-Embedded sl | 7.2 |


11/2006
6FC5397-8CP10-0BA0

## Safety Guidelines

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

**Danger**

indicates that death or severe personal injury **will** result if proper precautions are not taken.

**Warning**

indicates that death or severe personal injury **may** result if proper precautions are not taken.

**Caution**

with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken.

**Caution**

without a safety alert symbol, indicates that property damage can result if proper precautions are not taken.

**Notice**

indicates that an unintended result or situation can occur if the corresponding information is not taken into account.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

## Qualified Personnel

The device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by **qualified personnel**. Within the context of the safety notes in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

## Prescribed Usage

Note the following:

**Warning**

This device may only be used for the applications described in the catalog or the technical description and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens. Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance.

## Trademarks

All names identified by ® are registered trademarks of the Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

## Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Automation and Drives
Postfach 48 48
90437 NÜRNBERG
GERMANY

Order No.: 6FC5397-8CP10-0BA0
℗ 02/2007

# Table of contents

## Tables

# Introduction 1

## 1.1 General

Since the 1990s, Microsoft have put considerable effort into satisfying global demands for information processing and software development.

In addition to supporting as many languages as possible, one of the aims was to be able to support all languages with the same uniform operating system, as opposed to the language-specific versions of operating systems which were used before.

As part of these efforts, many of the system components and applications were gradually converted to the cross-language platform Unicode.

Although in many cases the user would have been completely unaware of this conversion process, in terms of the subjects discussed in this guide fundamentally different procedures need to be adopted depending on which version of the operating system is used.

The procedure recommended for Windows 2000/XP is much simpler than the one recommended for Windows 95/NT4. Therefore it makes sense to change over to the new systems. The guide for older systems is no longer being updated by the documentation department.

## 1.2 Prerequisites

### Supplementary conditions

This document describes the procedure for generating a text with the following preconditions:

- The text is to be generated on a Microsoft Windows operating system (Windows XP).
- The text is to be generated in a language not identical to the language of the operating system (= default language).
- The present description uses German as the default language for all examples provided herein. However, the described procedure will work just as well with a different language as the default Windows language.
- The text is to be generated in a format known as a "text file", i.e., a file which does not contain any formatting such as font type, font size, underlining, etc. This means that, apart from the text characters, the file will only contain ASCII control characters such as Tab, CR and LF (tab indent, carriage return and line feed).
- The language of the text requires a different Windows code page to that of the default language, i.e., the target language does not belong to the same language family as the default language.

- The text file is to be coded using this code page, not as a Unicode file.
- The text should be generated as simply as possible: in particular, this means that if possible, no additional software should have to be installed.
- For the purposes of this guide, it is assumed that you have installed Microsoft Word 2003 (or a later version) and are familiar with using it.

## Text types for HMI Embedded and HMI Advanced

The following text types can be changed in the individual systems:

| Text types | Systems |
|---|---|
| Alarm texts<br>Texts for "Expand user interface"<br>Application texts for all operating areas incl. softkey texts | HMI Embedded |
| Alarm texts<br>Texts for "Expand user interface"<br>Texts for the "Help" function in the editor<br>Machine data<br>Softkey texts for all operating areas | HMI Advanced |

## 1.3 Language families

A language family is a group of languages for which Microsoft operating systems use the same code page. Microsoft uses the term "language group".

Table 1-1 Language families

| Language family | Windows code page | Languages |
|---|---|---|
| Central European | 1250 | Albanian, Croat, Polish, Romanian, Serbian (Latin), Slovakian, Slovenian, Czech, Hungarian |
| Cyrillic | 1251 | Bulgarian, Macedonian, Russian, Serbian (Cyrillic), Ukrainian, White Russian |
| Western | 1252 | German, English, Finnish, French, Indonesian, Icelandic, Italian, Dutch, Norwegian, Portuguese, Swedish, Spanish |
| Greek | 1253 | Greek |
| Turkish | 1254 | Turkish |
| Baltic | 1257 | Estonian, Latvian, Lithuanian |
| Japanese | 932 | Japanese |
| Chinese | 936 | Simplified Chinese |
| Korean | 949 | Korean |
| Chinese | 950 | Standard Chinese (Traditional Chinese) |

### Operating system and target language from the same language family

If the language of the operating system (e.g., German) belongs to the same language family as the target language (e.g., Portuguese), then some difficulties could possibly occur with the **input** of certain characters. However, there will be no problems **displaying** these characters with a simple text editor.

Remedy:

- Use the Windows "Character Map" tool

  (see Section "Input via the Character Map") or
- or switch the keyboard

  (see Section "Using a different keyboard assignment") or
- use the ALT+digit combination to enter these characterseinzelne Zeichen mit ALT+Ziffernkombination eingeben

  (see Section "Numerical text input")

There should not be any further problems in this case. Use a simple text editor (e.g., Notepad) and save the texts in the usual way.

Section "Files with Word 2003" does not apply.

### Operating system and target language from different language families

If the language of the operating system (e.g., German) does not belong to the same language family as the target language (e.g., Hungarian), difficulties will occur with both **inputting** and **displaying** the characters in a simple text editor (Notepad).

## 1.4 Tools

### Standard version

This document describes the use of Microsoft Word 2003 and other tools which may not necessarily be installed on your PC.

You should have installed the following components:

- Microsoft Word 2003 (required)
- Windows XP: Language Support for East Asian characters (necessary for these languages)
- Windows XP: Character Map (recommended)

### Checking the installation of the Language Support for East Asian languages

To install the Language Support for East Asian languages, please refer to Section "Special considerations when working with East Asian languages".

## 1.5 Terms

### Code page

A list of characters which forms the character set for one or more languages. A code which is unique within this code page is assigned to each of the characters in the list.

All of the code pages used in Windows share the ASCII range (codes 0 to 127).

A distinction is made between single-byte code pages and multiple-byte code pages.

In single-byte code pages (all European languages), each character is represented with a single byte. Accordingly, single-byte code pages comprise a maximum of 256 characters.

Multiple-byte code pages (Asian languages) contain both characters which are stored as a single byte and characters which are stored with two (or more) bytes. The shared ASCII range is contained as a range of single-byte characters.

There is a 1:1 assignment between Windows code page and language family.

The language families are listed in Section "Language families".

### Diacritical characters

(diacritical = distinguishing)

In most cases, a relatively small supplementary character, which is attached to a letter to give the letter a specific stress, a specific pronunciation or even a new meaning.

Section "Special characters in different language families" contains a list of diacritical characters.

If the combination of basic letters and diacritical characters belongs in a language-specific alphabet, this combination is contained in the appropriate Windows character sets (code page, Unicode) as a separate character.

## Use of diacritical characters

A diacritical character is used

- To designate the combined character (e.g., Ģ = G cedilla, Õ = O tilde),
- For the combined input, if there is no separate key that possesses the required combined character

  (see Section "Using a different keyboard assignment"),
- In Vietnamese and Thai, for normal coding in text files,
- For output (only in typographically compound texts): Diacritical characters and basic letter are output separately, meaning that any combinations are possible, e.g., for ancient languages and phonetics.

In all the other cases, the combined letter is always considered as a compound unit.

In some cases, however, the combined letter is incorrectly referred to as the diacritical character.

## Input Method Editor (IME)

An aid for inputting the CJK characters
(CJK: Abbreviation for Chinese, Japanese and Korean).

## Language family

A language family is a group of languages which uses the same code page. This requires the use of the same basic alphabet (e.g., Latin or Cyrillic), but it does not mean that the alphabet of all languages of this group is the same.

There is a 1:1 assignment between Windows code page and language family.

The language families are listed in Section "Language families".

## Font files (TrueType fonts)

Windows XP does not use font files that are stored separated for each code page; instead it uses something known as "Big Fonts", which are fonts that contain the character set for several code pages. saving memory capacity for the common characters.

With Word 2003, the selection of characters from this total character set is performed via the Unicode coding, not via the code page coding. This makes it impossible to mix up characters with the same code in different code pages.

## Default language

The term "default language" is used in this guide for the language of the operating system, i.e., German for a German Windows environment and English for an English Windows environment.

## Default keyboard setting

Under "Control Panel" → "Regional and Language Options", in the tab "Languages", click under the heading "Text Services and Input languages" on the button "Details..." to reach the dialog box "Text Services and Input languages".

Here you can see the default keyboard setting in the "Settings" tab under the heading "Default Language and Region Support".

The default keyboard setting consists of a language property (left) and a keyboard assignment (right).

This language property of the default keyboard setting need not absolutely match the default language.

## Text file

The term "Text file" is used to describe a file which does not contain any formatting (font type, font size, underscore, bold, italics etc.) or embedded objects (pictures, tables, graphics, footnotes etc.).

This means that, apart from the text characters, the file will only contain ASCII control characters like Tab, CR and LF (tab indent, carriage return and line feed).

The text characters are taken from a particular code page.

Occasionally, text files are also incorrectly referred to as ASCII files. This is only correct if a text file only contains ASCII characters (codes 0 to 127) and is therefore code page invariant.

## Unicode

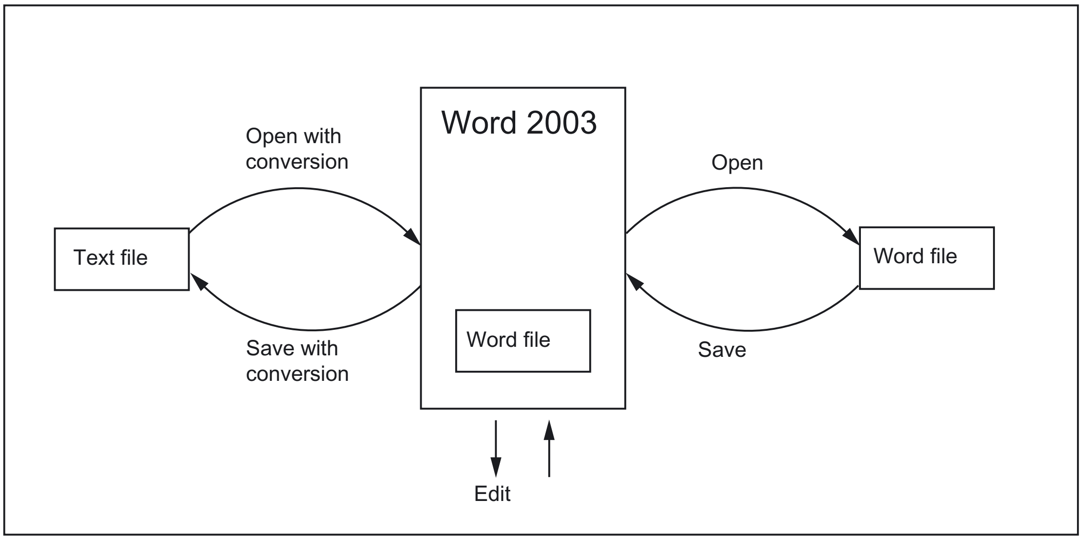
A 16-bit character set for which code pages are no longer required. All character codes are unique, without needing to specify the code page. Unicode programs like Word 2003 do not offer code page-dependent interpreting of characters and thus also cannot display characters differently by assigning a font or language property.

# Editing files with Word 2

## 2.1 Overview

This section describes how text files can be edited using Microsoft Word 2003.

| | |
|---|---|
| Word 2003 does not edit text files directly. Instead, it converts them to Word files upon opening.<br>During saving Word can then convert the edited text back into a text file. | We recommend saving an additional version of the text file as a Word file and using this version as the basis for future editing.<br>It is then no longer necessary to open and convert the file. |



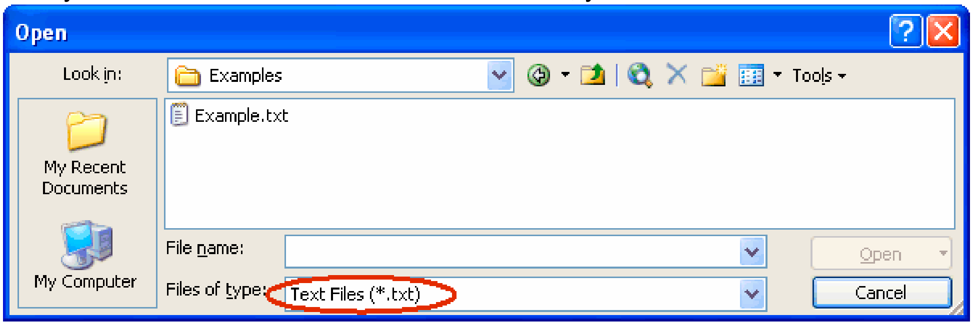

## 2.2 Generating a new text file

1. You can create a new text file as follows:
   – In Windows Explorer, click the command "File" → "New" → "Text Document".
   – Launch Notepad without a file. A new empty file is then opened. Then select the command "File" → "Save As" → "Encoding: ANSI"
   – Make a copy of an existing text file you wish to use as a template for the new text file.
2. Now open the text file with Word 2003 in the way described in the next section.

## 2.3 Opening an existing file

1. In Word 2003, select "Open…" from the "File" menu.
2. From the "Open" dialog box, choose the "Text files" (*.txt)" setting under "Files of type".

   If your file has a different extension from "txt", you must rename it.




3. Choose the file you would like to open and click "Open".

   Instead of steps 1-3, you can also use one of the standard shortcuts:

   – In Word you can use the list of the recently used files.
   – Drag and drop the text file from Explorer to the Word icon on your desktop.
   – If Word has already been launched, drag and drop the text file from Explorer to the title bar of the Word window.
4. Note what happens next.

   Continue with "Convert File" (see point 5.) if this dialog box opens.

   Continue with "File Conversion" (see point 6.) if this dialog box opens.

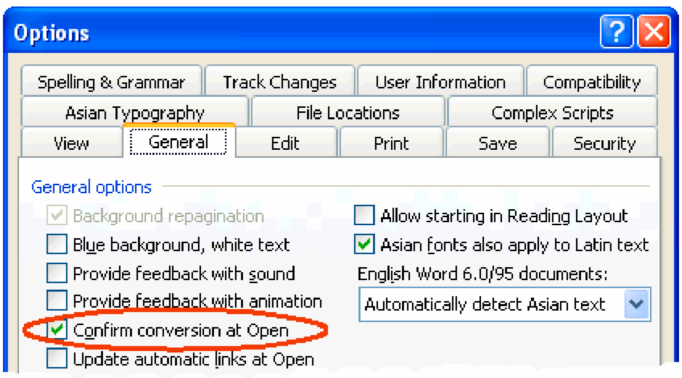
   If, however, the file is opened straightaway with neither the "Convert File" nor the "File Conversion" dialog boxes opening up, then Word has probably opened the file incorrectly under the assumption that the text has been written using the default language.

   Be careful with special characters. The special characters will appear incorrectly if the code page of the text file is not the same as the code page of the default language.

   In this case, go to "Extras" → "Options..." and open the Options dialog box for Word. Click "General" and select the option "Confirm conversion at Open".

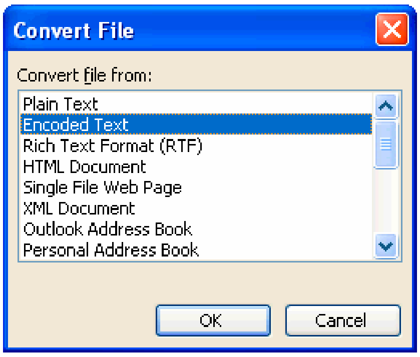
   When this option is active the additional dialog box "Convert File" (see point 5.) will appear every time, but when the option is inactive it is skipped.

   Afterwards close the file and start again from point 1.



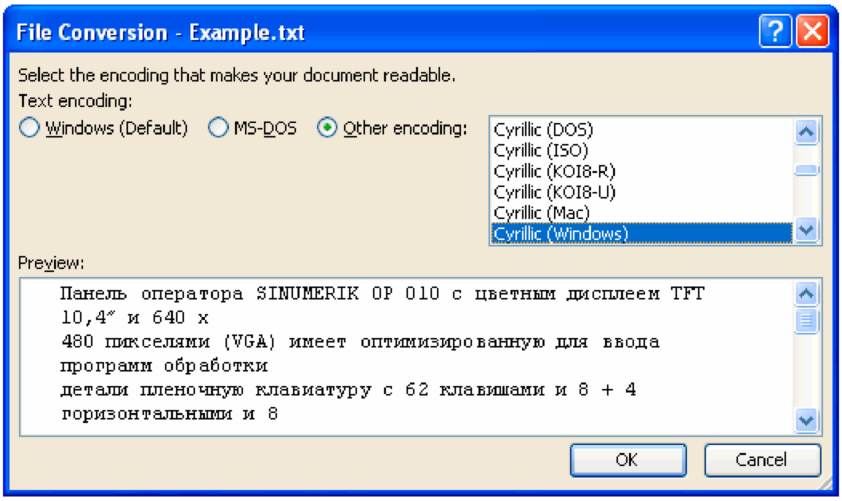
5. The "Convert File" dialog box will appear.

   Select "Encoded Text" and click "OK".




6. The "File Conversion - Example.txt" dialog box will then appear:

   – If "Windows (default)" is selected, activate "Other Encoding". Select the correct encoding (Word will make a suggestion, which will not necessarily be correct).




   – Section "Codes for SINUMERIK HMI", Table 2-1 lists the suitable encoding options. Check the preview area while doing this.

7. Click "OK".

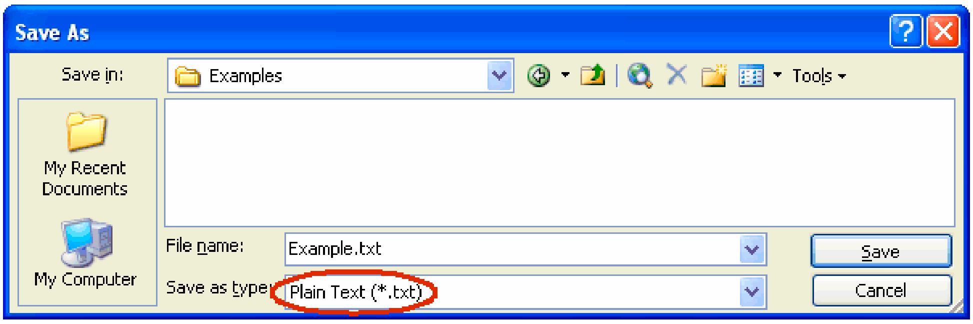
   The file is opened and converted to a Word file at the same time.

   The existing text is formatted using a default font.

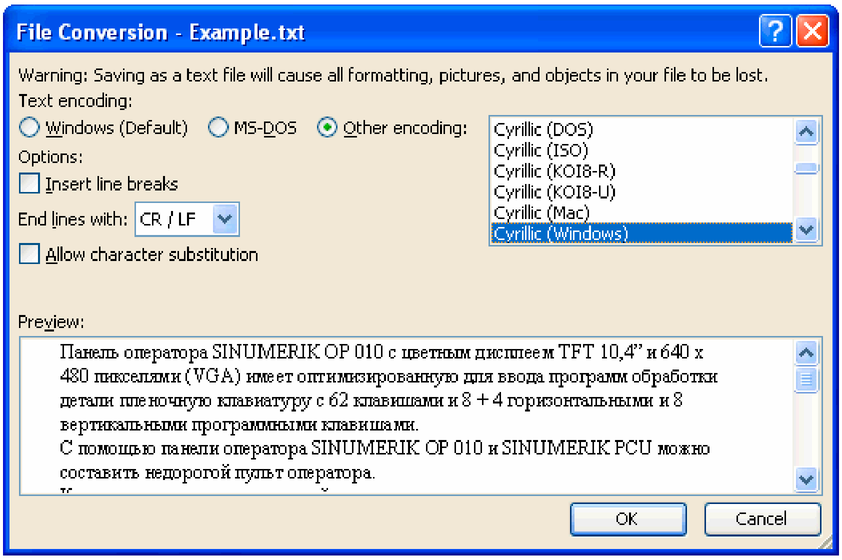
## 2.4 Saving text files

Save the text entered as a "Text File":

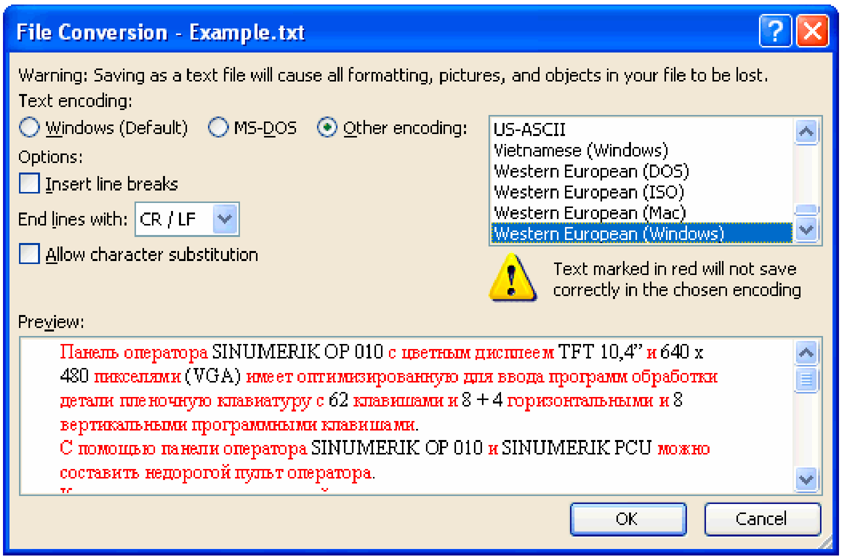
1. In Word, select "Save As…" from the "File" menu.




2. Choose the option "Text Only (*.txt)" under "Save as type".
3. Enter the name of the file in the "File name" box and click "Save".
   – The "File Conversion" dialog box will then appear.




   – Select the encoding which is suitable for your target language and click "OK". Section "Codes for SINUMERIK HMI", Table 2-1 lists the suitable encoding options.
   – Your file is now saved as a text file and encoded using the selected code page in the process.
   – If you have selected a completely unsuitable code then a warning is displayed in the dialog box. This warning will also appear if you have selected the correct code but your file contains characters which are not permitted, such as Latin mutated vowels in a Cyrillic text.




– If you choose to ignore the warning, then all characters which cannot be encoded are replaced with a question mark.

## 2.5 Codes for SINUMERIK HMI

When you open or save a file, code page names are used rather than code page numbers, see Section "Opening an existing file" or "Saving text files".

Depending on the installation of your system, many other codes may be listed, including those for Unix (ISO), Apple (Mac), OEM (DOS) and IBM (EBCDIC) platforms.

Please use the Windows (ANSI) codes for SINUMERIK HMI.

Use the following codes to open and save files (shown in alphabetical sequence in accordance with the Word listing):

Table 2-1 Encoding

| Description | Code page |
|---|---|
| Traditional Chinese (Big5) | 950 |
| Simplified Chinese (GB2312) | 936 |
| Japanese (Shift-JIS) | 932 |
| Korean | 949 |
| Cyrillic (Windows) | 1251 |
| Central European (Windows) | 1250 |
| Turkish (Windows) | 1254 |
| Western European (Windows) | 1252 |

# Input of Text 3

## 3.1 Special characters in different language families

### Latin language family

With the Latin-based language families (Baltic, Central European, Turkish, Western), you can enter most of the characters, i.e., the basic Latin alphabet, directly with your (e.g., Western) keyboard.

Most of the special characters, such as Ąáâåçđéèĺïł ŇñöőÓťŤůū, are made up of the Latin letters A-Z a-z and diacritical characters such as acute ', breve ˘, cedilla, circumflex (caret) ˆ, point ˙, colon (dieresis, trema) ¨, double-acute ˝, grave accent ', háček (caron) ˇ, macron ¯, ogonek (nasal hook, crooked hook) ˛, squiggle ˚, slash /, hyphen – or tilde ˜.

Other characters of the alphabet are special letters (Þþ, Đð, ß) or ligatures (Ææ, Œœ, IJij). These are supplemented by language-specific abbreviations and punctuation marks, e.g., for the endings of the ordinal numbers in Spanish (ª and º) as well as the upside-down exclamation and interrogation marks (¡ and ¿).

The input methods described in the following will apply without restrictions to the Latin-based language families.

### Greek/Cyrillic language family

Certain special considerations need to be taken into account when inputting texts using Greek or Cyrillic fonts.

Read Section "Special considerations when working with Greek and Cyrillic texts" for additional information.

### East Asian languages

Certain special considerations also need to be taken into account when generating texts in Japanese, Chinese or Korean.

Read Section "Special considerations when working with East Asian texts" for additional information.

### Disruptive Word options

Deactivate the following options in Word to prevent unwanted characters from reaching the text:

In the dialog box "Tools" → "AutoCorrect Options" go to the "AutoFormat as you type" tab and under "Replace as you type" check:

- "Straight quotes" with "Smart quotes"
- English ordinals (1st) with superscript
- Fractions (1/2) with fraction character (½)
- Special characters (--) with symbols (—)

"Apply as you type":

- Automatic bulleted lists
- Automatic numbered lists

If you do leave any of the options active then you can always use Ctrl+Z to undo any automatic corrections. As a general rule this will give the same results as if the option was deactivated.

## 3.2 Inputting characters via the Character Map

---

**Note**

The Character Map integrated in Word (menu commands "Insert" → "Symbol...") is not suitable for generating code page-encoded text files, as it cannot be limited to the relevant code page.

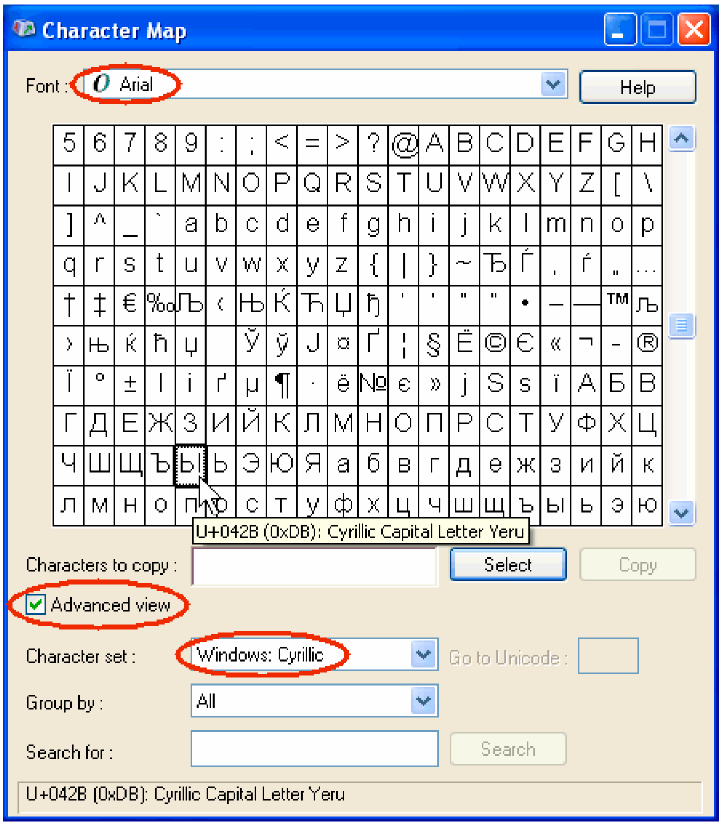
Use the "Character Map" system program as described below.

---

### Selecting the Character Map

1. In the start menu:

   select "Programs" → "Accessories" → "System Tools" → "Character Map" to start the "Character Map" tool.
2. From "Font", choose the same font type as you have selected for the text file.




3. Activate the box "Advanced view".
4. Under "Character set" select the code page for your text file in order to restrict the number of available characters to this code page.

   The code pages are referred to as follows:

Table 3-1 Text file coding

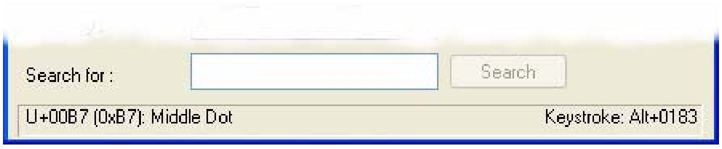
| Description | Code page |
|---|---|
| Windows: Chinese (Taiwan) | 950 |
| Windows: Chinese (PROC) | 936 |
| Windows: Japanese | 932 |
| Windows: Korean | 949 |
| Windows: Cyrillic | 1251 |
| Windows: Eastern European | 1250 |
| Windows: Turkish | 1254 |
| Windows: Western | 1252 |

### Inputting characters

1. Choose a character you want to input.

   The information bar at the bottom will display the Unicode code (U+..., hexadecimal), the code page code (in brackets) and the clear text description of the character.

   If you have set up the keyboard assignment accordingly (see Section "Using a different keyboard assignment"), the keystroke combination for entering the character numerically will appear at the bottom right in the status bar (see Section "Numerical text input").




2. Click "Select".

   The character is copied into the "Characters to copy" field.
3. Repeat steps 1 and 2 for further characters.
4. After you have selected all the characters, click "Copy".

   The characters are copied from the "Characters to copy" field to the clipboard.
5. Switch back to Word and press Ctrl-V.

   The character is copied from the clipboard to the text.

## 3.3 Compound input

With many of the compound special characters it is possible to input the diacritical character first and then the letter. The diacritical character will not appear at first and is then combined with the letter to form a special character.

### Example

With a German keyboard assignment, press, for example, the keys ' (acute) and "a" one after the other to obtain the character á (a-acute).

The keys with the diacritical characters, which are thus dealt with in a special way, are marked with a special color in the descriptions of the keyboard assignments.

### Keyboard assignments

The table below shows some keyboard assignments for the supported diacritical characters:

| Keyboard assignments | Acute | Grave | Circumflex | Háček | Breve | Ogonek | Point | Colon | Cedilla | Double acute | Squiggle | Tilde |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ' | ' | ^ | ˇ | ˘ | ˛ | ˙ | ¨ | ¸ | ˝ | ˚ | ˜ |
| German | X | X | X | | | | | | | | | |
| French | | X | X | | | | | X | | | | X |
| Polish | X | | X | X | X | X | X | X | X | X | X | |
| Spanish | X | X | X | | | | | X | | | | X |
| Czech | X | | X | X | X | X | X | X | X | X | X | |
| Hungarian | X | | X | X | X | X | | X | X | X | X | |
| U.S. International | X | X | X | | | | | X | | | | X |

Obtain a graphical representation of your keyboard assignment to learn which key combinations produce the diacritical characters.

### Keyboard assignment in picture form

Images of the keyboard assignments can be obtained from Microsoft via the Internet at the following address:

http://www.microsoft.com/globaldev/reference/keyboards.mspx

If you press the space bar after the diacritical character, you will see the character itself.

---

**Note**

It can be useful to switch the keyboard assignment to "U.S. International" which corresponds mainly to the American keyboard, but additionally contains 5 diacritical characters and provides many special characters.

---

## 3.4 Using a different keyboard assignment

### Necessity of changing the keyboard assignment

When is a change in the keyboard assignment necessary?

- If you have to input a lot of text.
- Your translator, who is used to a certain language-specific keyboard, wants to use your PC.

However, if you (as a German) only need to make smaller changes to existing files, then the German keyboard assignment may be enough, provided there are only a few special characters which cannot be represented with the German keyboard. You can enter these characters using the Character Map.

## Example

In Hungarian, for example, you would have to input these characters as follows:

- őŐűŰ (O double-acute, U double-acute) via the Character Map
- öÖüÜ directly
- áÁéÉíÍóÓúÚ compound input

## 3.4.1 Concept of the keyboard language

Windows manages pairs for the keyboard consisting of "Language ("Language and Region Support")" and "Keyboard assignment". The set keyboard language determines which language property is assigned to the characters entered. The associated keyboard assignment will determine which key produces which character.

Several such settings can be set up and it is possible to switch between them. One of the settings is defined as the default setting.

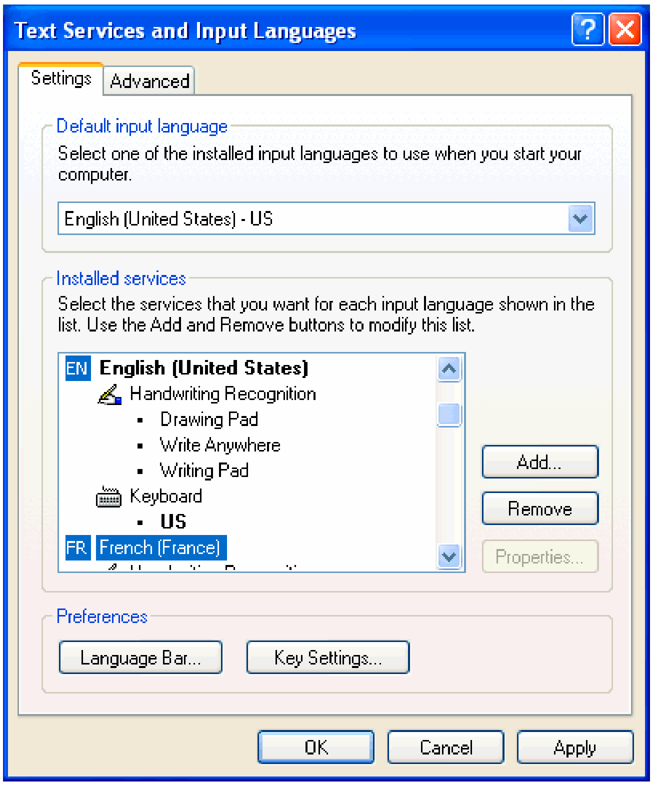
Windows uses the term "Language and Region Support" as this can be used not only to switch the language properties, but also to select settings such as currency, date format etc.

## 3.4.2 Setting up a keyboard assignment in Windows XP

Proceed as follows to select a new keyboard assignment:

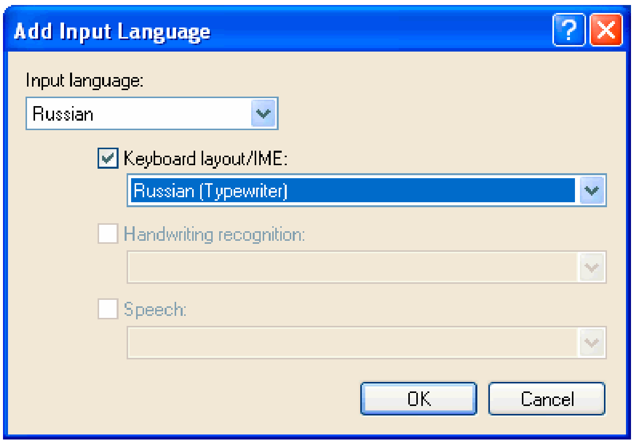
1. From the Start menu, select "Settings" → "Control Panel" and click "Regional and Language Options".
2. Click the "Languages" tab and click the button "Details..." under the heading "Text Services and Input Languages".

   If a symbol such as DE is displayed in the right-hand taskbar, you can also right-click this symbol and select the command "Properties..." to speed up steps 1 and 2.
3. The dialog box "Text Services and Input Languages" is opened. Select the "Settings" tab.




4. Click "Add".

   The dialog box "Add Language and Region Support" is opened.




5. In the dialog box "Add Language and Region Support", select a language under "Language and Region Support" and the associated keyboard assignment under "Keyboard Assignment".

   The language ("Language and Region Support") may also be the same as the default language or as some other language for which you have already set up a keyboard assignment. However, in this case you will not be able to make such a good distinction between the two keyboard assignments, as Windows usually only displays the language ("Language and Region Support").

6. Click "OK". The dialog box "Add Language and Region Support" is closed. You then return to the "Text Services and Input Languages" window.
7. Use "Keyboard…" to select the way in which you want to switch the keyboard assignment. If you select "none", you can only switch using the mouse.
8. Under the "Language and Region Support" bar you can select whether and how the current keyboard assignment is displayed in the taskbar or in a special button bar.
9. Click "OK" to quit the dialog box.

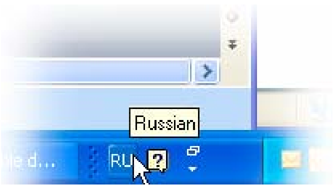
## 3.4.3 Use

You can now create text with a "keyboard assignment" suitable for the target language (for the input of your texts in a foreign language) **or** with your previous default keyboard assignment (for the remaining operation). You can switch between the two assignments as required.

Now, a small blue field with a two-letter language abbreviation should appear in the taskbar for the language you have set:

If you position the mouse pointer on this field (without clicking it), the setting is displayed .

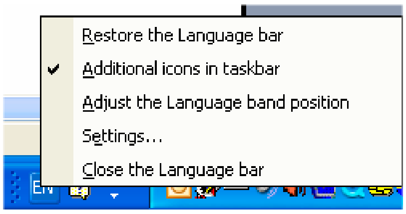
If the selected keyboard assignment is the default assignment for the selected language (e.g., "Turkish" – "Turkish Q"), then only the language ("Turkish") is displayed. Otherwise, the complete settings with language and keyboard assignment will be displayed ("Turkish – Turkish F").




If you click this field with the **left** mouse button, the languages you have set will be offered in a small menu.

The current setting is marked by a check mark.


If you click this field with the **right** mouse button, an abbreviation of the dialog box "Keyboard Properties" is displayed above the "Properties" menu option.




Among further settings that are offered is the ability to arrange this display as an Input language bar independently of the taskbar: at the top edge of the screen, for example.

### Keyboard assignment specific to input window

The Windows operating system will memorize the current keyboard assignment **for each input window**.

- The keyboard assignment is switched automatically if you switch to another window.

  A new window is always started with the default setting. After starting Word, you may have to reselect the keyboard assignment.
- Enable the window in which you want to input texts before you select the keyboard settings for this window.

### Menu operation

All menus in the active program will also use the changed keyboard assignment. You will possibly no longer find certain key combinations with Alt and the letter key or Ctrl with a letter key, or even initiate a wrong command by mistake. You should therefore use the mouse or, in menus, the arrow keys.

---

**Note**

If you do not want to switch back to the other keyboard assignment quite so frequently, you can also declare the new combination with the special keyboard assignment to be the default target language in the "Text Services and Input Languages" dialog box by selecting Start menu "Settings" → "Control Panel" → "Regional and Language Options" → "Languages" tab:

Select the new setting under "Default Language and Region Support".

Please note that all new windows, the Start menu and the desktop will now also use the new keyboard assignment.

---

### Keyboard assignment in picture form

You can obtain the keyboard assignment in picture form.

For appropriate Internet addresses, see Section "References for text generation"

### Checking the keyboard assignment

Compare the keyboard assignment and note a key with two different assignments. If you are not sure, press this key when inputting the text to test which assignment is active.

**Example:** If the German keyboard assignment is active, pressing key "1" (letter group) will produce a "1" and if the Czech keyboard assignment is active, a "+".

---

**Note**

Check whether the automatic option for switching over the keyboard has been activated in Word. Inadvertent switching of the keyboard assignment by Word can cause confusion.

Section "Multi-language Word files" describes situations in which the automatic keyboard switching function is useful and how to activate and deactivate it.

---

## 3.5 Numerical text input

You can only use numerical input with Word if you have selected a suitable keyboard assignment (see Section "Using a different keyboard assignment").

Press the ALT key, and while holding down the key, enter zero and then the three-digit decimal coding of the desired special character on the numerical key group. Only then release the ALT key.

Example (Turkish):

ALT+0222 results in Ş (S with cedilla).

To produce this character, you will only need the Character Map of the appropriate code page (see Section "Language families").

Section "Input via the Character Map" also describes where the key combination is displayed in the "Character Map" tool.

### Overview tables of the code pages

You will find overview tables of the code pages at, e.g., Microsoft under:

http://www.microsoft.com/globaldev/reference/WinCP.mspx

From these tables, combine the headers for the column and row (hexadecimal 00 to FF) and convert this number to form a decimal number (0 to 255).

The 4-digit hexadecimal numbers contained in the individual table fields are the corresponding Unicode codes which can be used for identification of a character, but they cannot be used for numerical input of that character.

## 3.6 Language properties in Word files

### Language property

The language is a property (attribute) that is managed separately in Word for each character, in the same way as other text properties (bold, underline, etc.):

- In the language properties, Word will remember the language to which the character, the character sequence, the word or the whole sentence belongs.
- The language property is independent of the font. For example, a Cyrillic character can possess the language property "German".
- The language property is managed internally as a combination of main language and sublanguage, e.g., "French (Canada)".

Microsoft sometimes uses the designation "Region scheme" for the language property: this term also covers properties such as currency, date format, decimal separators etc.

### Effects of the language property

This property is used (evaluated) for the spell checker and for automatic keyboard switching (see Section "Multi-language Word files").

If you do not wish to use either the spell checker or the automatic keyboard switching function, there is generally no need to worry about the language properties. In this case, do not forget to deactivate the automatic keyboard switching function.

### Saving the language property

Microsoft Word saves this property in Word files (*.doc). If a document is saved as a text file (*.txt) then this property is lost along with the other text properties.

If you wish to use the language property, it therefore makes sense to use a Word file to edit the texts and to keep it for subsequent changes. Whenever you reach a stage where you want to stop editing, you can save the document as a text file following the procedure described in Section "Saving text files".

### Displaying the language property

In Word, the current language property is displayed in the status bar at the bottom edge of the window.

## Setting the language property

Proceed as follows to set the language property for a text:

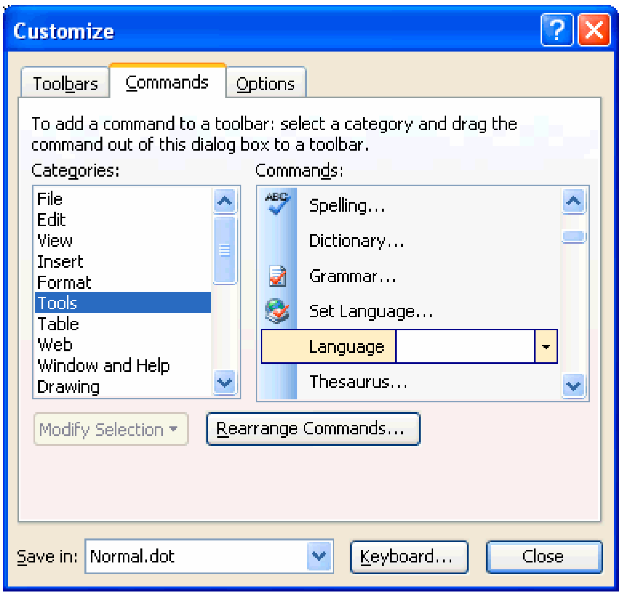
1. Highlight the part of the text for which you wish to set the language property, or press CTRL+A to select the entire text.
2. Then select the command "Extras" → "Language" → "Set Language...".
3. In the "Language" dialog box you can choose the correct language.
   Click "OK" when you are done.

## Quick setting of the language property

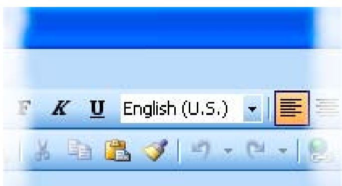
In Word, you can insert a combined display and selection box into a symbol bar. This both displays the current language property and enables you to set the language as well.

Proceed as follows to insert this field into a symbol bar:

1. In the "Extras" menu, select the command "Customize..."




2. In the "Customize" dialog box select the "Commands" tab.
3. Select "Tools" from the list of categories on the left.
4. Then select "Language" from the list of commands on the right (see Fig.).
5. Drag and drop this entry with the left mouse button into a symbol bar.




6. Close the dialog box.

With this selection box, you can then change the selected language for any part of the text.

## Default setting of the language property

Word automatically sets the language property in the following situations:

- When opening a file of type "Encoded Text File" (see Section "Opening an existing text file").
- New text entered with the keyboard is automatically assigned the selected keyboard language.
- If the "Detect language automatically" option is selected, Word will try to assign the words entered to a language once they are complete and then automatically sets the language property.

---

**Note**

The "Detect language automatically" option can often deliver confusing results, particularly in technical texts.

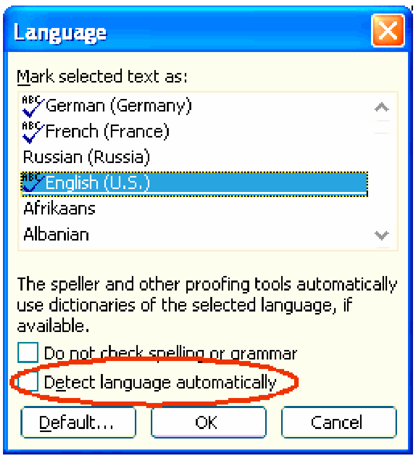
This option is usually activated as a default setting.

**Recommendation**: Deactivate the "Detect language automatically" option.

---

Proceed as follows to deactivate the "Detect language automatically" option:

1. Make sure that no part of the text has been highlighted and select the command "Tools" → "Language" → "Set Language..."
2. In the "Language" dialog box, deactivate the option "Detect language automatically" and click "OK".



## 3.7 Multi-language Word files

In certain cases it may be useful to deliberately set different language properties for different parts of the text in a Word file.

### Supplementary conditions for multi-language Word files

You want to enter certain parts of the text with the German keyboard assignment and then also edit these parts later on with the German keyboard assignment, whereas other parts of the text are to be generated and edited with the keyboard assignment of the target language.

### Table with two language properties

To generate a multi-column text in which, for example, the first column possesses the language property "German" and the second column "Russian", for example, you can proceed as follows:

- Switch to the German keyboard and enter the first column of the first line. Then switch to the Russian keyboard and enter the second column:

  [Text 103][Mode 28] "йцукенгшщзхъфывапролджэячсмить"

- This does not need to be a Word table with borders and fields. You can also use a simple line and separate the different parts with spaces.
- Select the whole line, copy this to the clipboard by using Ctrl-C and paste this line several times by pressing Ctrl-V. This will give you two columns with German on the left and Russian on the right.

| | |
|---|---|
| [Text 103][Mode 28] | "йцукенгшщзхъфывапролджэячсмить" |
| [Text 103][Mode 28] | "йцукенгшщзхъфывапролджэячсмить" |
| [Text 103][Mode 28] | "йцукенгшщзхъфывапролджэячсмить" |
| [Text 103][Mode 28] | "йцукенгшщзхъфывапролджэячсмить" |
| [Text 103][Mode 28] | "йцукенгшщзхъфывапролджэячсмить" |
| [Text 103][Mode 28] | "йцукенгшщзхъфывапролджэячсмить" |

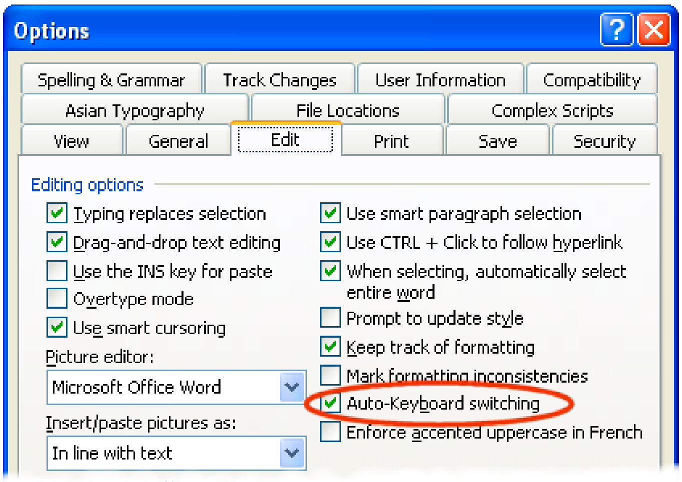
### Examples

This can be necessary in the following cases:

- With Greek and Cyrillic texts, the Latin letters and other ASCII characters are missing in the appropriate keyboard assignments.
- With some keyboard assignments such as Czech, no access to the digits is granted on the standard keyboard level.
- The arrangement of the Latin letters deviates substantially from the arrangement to which you are accustomed (e.g., Turkish-F keyboard assignment).
- You, as a German, want to prepare one part of the text, and a second part is to be generated by your translator.

## Automatic keyboard switching

In order to automatically have the correct keyboard assignment when working on different parts of the text, in Word go to "Tools" → "Options" and activate the "Auto-Keyboard switching" option in the "Edit" tab.




### Note

Accidental use of the automatic keyboard switching function can be confusing. Only use this option when the aforementioned supplementary conditions are met.

## Active keyboard assignment always visible

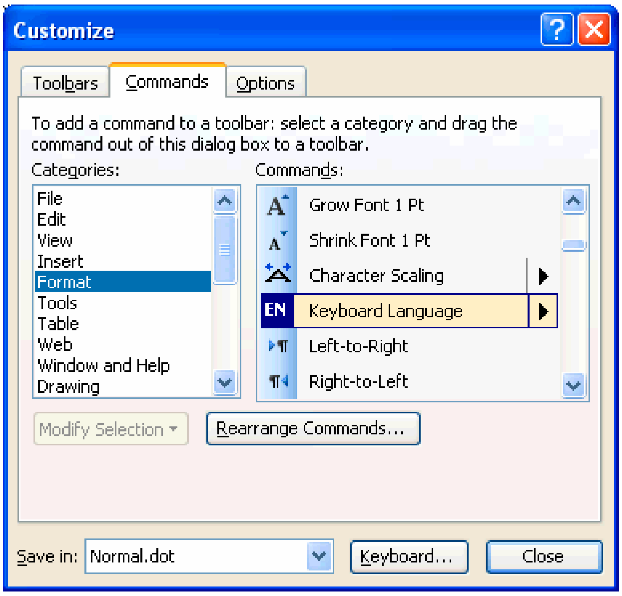
To make sure you know what is going on, you can adjust the taskbar to always display the currently active keyboard assignment.

1. Right-click the taskbar and select "Properties".
2. Activate the "Always keep the taskbar on top" option.
3. Deactivate the "Automatically hide taskbar" option.

You are also free to position the Language and Region Support bar anywhere you want on the screen. To do this, click the button for switching over the keyboard (e.g., **DE**) in the taskbar and select "Restore Language and Region Support Bar".

Another option is to activate keyboard switching in Word:

1. In the "Extras" menu select the command "Customize..."




2. In the "Customize" dialog box select the "Commands" tab.
3. Select "Format" from the list of categories on the left.
4. Then select "DE Keyboard Language" from the list of commands on the right (see Fig.).
5. Drag and drop this entry with the left mouse button into a symbol bar.

## 3.8 Special considerations when working with Greek and Cyrillic texts

**Note**

If you are generating texts in the Greek or Cyrillic language for the first time, then you MUST read this section!

From the point of view of the language sciences, the Greek and Cyrillic alphabets are considered as separate and independent alphabets, and not, for example, as an extension of the Latin alphabet.

This means that Greek and Cyrillic letters which, in their appearance, are identical to Latin letters, are not the same characters and may therefore also not be coded identically.

In some cases, this becomes evident by the fact that although certain letters look the same in upper case, the corresponding lower case letters look different.

| Font | Name | Meaning |
|---|---|---|
| Latin | P, p | Latin letter P |
| Greek | Ρ, ρ | Greek letter RHO |
| Cyrillic | Р, р | Cyrillic letter ER |
| Latin | H, h | Latin letter H |
| Greek | Η, η | Greek letter ETA |
| Cyrillic | Н, н | Cyrillic letter EN |

## Example

Incorrect coding could have the following consequences:

Let us suppose that you wrote the Russian word for "NO" ("HET") using Latin letters instead of "HET" using Cyrillic letters. Initially you would see no difference. However, a search for "нет" in lower case letters with the search option "Ignore uppercase/lowercase letters" would not find the incorrectly coded spelling.

## Keyboard assignment for Greek/Cyrillic font

For this reason, the keyboard assignments for the Greek and Cyrillic fonts have been designed in such a way that the Latin letters are not available at all, not even via AltGr key combinations. These keyboard assignments have normally only two levels (standard and SHIFT).

## Keyboard assignment for the Russian font

In the Russian keyboard assignment, there are also some ASCII special characters which are missing, such as #$[]{}"|~&.

For these languages it makes sense, therefore, to use Word files with two languages, as described in Section "Multi-language Word files".

---

**Note**

- Switch the keyboard to Latin only if this is absolutely necessary! When doing so, do not switch over for individual letters, but only for complete words and sentences.

  Never use words with mixed codings!
- Find out whether and to what extent certain German, English or international abbreviations (e.g. DIN/ISO) may or must be written using Latin fonts, or whether a conversion (ДИН/ИСО) is more appropriate, according to the conventions of the country in question.

  The key assignment for combinations, such as Ctrl-C, Ctrl-V, Ctrl-Z, Alt-A, etc. normally follow the US keyboard assignment.

---

# 3.9 Special considerations when working with East Asian texts.

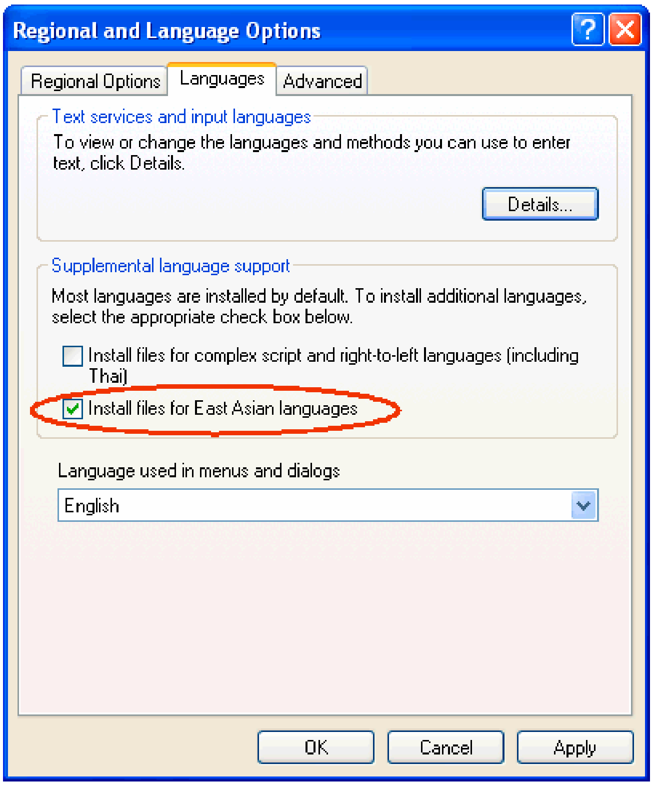
## Requirement

To author or correct texts in Japanese, Chinese or Korean, you will need to have installed the corresponding expansion package for the operating system.

## Checking the system installation

Open the dialog box "Regional and Language Options" via "Start" → "Settings" → "Control Panel".

Check the "Languages" tab to see whether the option "Install files for East Asian languages" has been selected.




You will not usually need the installation CD to reinstall the files for East Asian languages in Windows XP, as the files are already present on the hard drive in compressed form.

You will, however, require Administrator Rights to perform the installation.

## Input

An Input Method Editor (IME) is used for inputting. This is a tool which is used to put together the pictograms on a European keyboard and can be activated analogously to a keyboard assignment.

You will normally need a translator who speaks the target language as his/her mother tongue to operate the IME system.

There is no need to have a Windows operating system in the target language.

## Full-width European characters

In the code pages for the East Asian languages, two versions of many of the European characters (Latin letters, numbers and the characters # $ % & etc.) are included:

- In the normal (narrow, half-width) version.

  This is the set of ASCII characters which is compatible with European code pages.

- In a double-width (full-width) version.

  These versions of the characters are twice as wide and are not compatible with the ASCII characters.

If you are using characters with a functional significance it is important to use the ASCII versions.

**Correct example (SINUMERIK alarm text, Japanese):**

014195 0 0 "Ｄコード と Ｇ４９が同時に指令されています（Ch%1 %2）"

The circled characters are:

| | |
|---|---|
| 014195 0 0 | Identifier and attributes |
| Space character | Syntactical separator |
| Inverted commas | Text delimiters |
| %1, %2 | Parameter variables |

Only the ASCII versions must be used for these characters.

---

**Note**

In the remaining part of the text it is quite acceptable to use full-width characters for better readability (as in the example here for G49).

---

**Notice**

There is even a full-width version of the space character (ideographic space). You can only recognize this character from its width by selecting it.

---

Incorrect example:

Ｏ14195　0 0 “ＤコードとＧ４９が同時に指令されています（Ch％1 %２）”

The circled characters are full-width variants which have been used incorrectly.

---

**Note**

When placing a translation order, tell the translator that only the narrow, ASCII-compatible characters must be used for characters with a functional significance.

---

### Simplified/traditional Chinese

Please make a careful distinction between simplified Chinese and traditional (or standard) Chinese. The former is used in the PR of China, whereas the latter is used in Taiwan. These two variants are coded differently and therefore have to be treated differently. You should therefore also make sure that this is clearly specified in your translation order.

### Generating texts

When you are generating texts in Japanese, Chinese or Korean, it definitely makes sense to initially create a Word file.

There should be no problems if you are using Word 2003 to exchange documents with your external translator.

Afterwards, you can convert this Word file into a text file in the way described in Section "Saving text files".

# 4 Problems and remedies

## Keyboard assignment switches unintentionally

Problem:

The keyboard occasionally switches to a different assignment while text is being entered, without this being explicitly requested.

Remedy:

Check whether the option for automatic keyboard switching is set as described in Section "Multi-language Word files". Deactivate this option.

## Language properties are changed inadvertently

Problem:

After creating a text with certain keyboard assignments (e.g., Polish), you notice that parts of the text have German or English as the language property.

Remedy:

Deactivate the automatic language recognition option as described in Section "Language properties in Word files".

## Hotkeys no longer work

Problem:

Certain commands (Ctrl+C, Alt+T etc.) no longer work.

Remedy:

As a result of a different keyboard assignment, the keyboard shortcuts for certain commands have changed. Use the mouse while you are working with the other keyboard assignment.

## Individual texts not working

Problem:

In some cases, the finished text files do not work in the SINUMERIK Runtime system. Individual texts or lines of text are obviously not being read correctly.

Remedy:

Check the formal text criteria:

Are all of the characters which have a functional significance correctly encoded?

- If inverted commas are a requirement, check whether Word has converted the straight inverted commas (ASCII characters) that were entered into typographical inverted commas. Some disruptive Word options are listed in Section "Special characters in different language families".

- If the texts in question are East Asian texts, check the coding of the non-Asian characters. See Section "Special considerations when working with East Asian texts".

  If full-width characters have been used for functional characters, replace them with the corresponding ASCII variants.

## Yen character (Japanese) or Won character (Korean) in the text

**Problem:**

The translator has obviously used a Yen character ¥ or a Won character W wherever you would expect a backslash “\”.

**Remedy:**

This is not necessarily a mistake. Instead, this has historic reasons. For many years the Yen character has been used in Japan and the Won character in Korea in place of the backslash character.

This representation was kept once the PC users in those countries had got used to using this representation in path names (where you would otherwise expect the backslash character). However, internally this is still the backslash character.

In the fonts "MS Gothic" (Japanese) and "Batang" (Korean) the backslash symbol is represented as a currency character.

However, it is also possible that the Yen character (Unicode U+00A5) or Won character (Unicode U+20A9) was actually entered instead of the backslash character (Unicode U+005C). To check this, proceed as follows:

1. Reformat the text in Word using a different font, e.g., "Courier New". If you see the backslash character in its normal representation, there are no problems.
2. However, if you can still see the currency symbol or a symbol which is not a valid symbol, you should correct the character and replace it with a backslash.
3. Afterwards, revert back to the normal font (MS Gothic or Batang).

# 5 References for text generation

For further information about generating texts, please refer to:

- Book: *Developing International Software, Second Edition*

  Microsoft Press, October 2002, ISBN 0-7356-1583-7

  http://www.microsoft.com/globaldev/getwr/dis_v2/default.mspx
- Overview tables of the Microsoft code pages

  http://www.microsoft.com/globaldev/reference/WinCP.mspx
- Pictures of the keyboard assignments in Microsoft can be found at the following address:

  http://www.microsoft.com/globaldev/reference/keyboards.mspx
- Windows XP FAQs:

  http://www.microsoft.com/globaldev/DrIntl/faqs/winxp.mspx
- Microsoft Global Software Development

  Detailed explanations on various aspects of international software

  http://www.microsoft.com/globaldev
- Unicode

  Tables, definitions, standards and tools

  http://www.unicode.org

# Index

SIEMENS

# SINUMERIK 840D sl

## NCU Operating System (IM7)

Commissioning Manual



Valid for

SINUMERIK 840D sl/840DE sl control system

Software Version
NCU System Software 1.5
with HMI Embedded 7.5


01/2008
6FC5397-8CP10-1BA0

## Safety Guidelines

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

| ⚠DANGER |
| --- |
| indicates that death or severe personal injury **will** result if proper precautions are not taken. |

| ⚠WARNING |
| --- |
| indicates that death or severe personal injury **may** result if proper precautions are not taken. |

| ⚠CAUTION |
| --- |
| with a safety alert symbol, indicates that minor personal injury can result if proper precautions are not taken. |

| CAUTION |
| --- |
| without a safety alert symbol, indicates that property damage can result if proper precautions are not taken. |

| NOTICE |
| --- |
| indicates that an unintended result or situation can occur if the corresponding information is not taken into account. |

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

## Qualified Personnel

The device/system may only be set up and used in conjunction with this documentation. Commissioning and operation of a device/system may only be performed by **qualified personnel**. Within the context of the safety notes in this documentation qualified persons are defined as persons who are authorized to commission, ground and label devices, systems and circuits in accordance with established safety practices and standards.

## Prescribed Usage

Note the following:

| ⚠WARNING |
| --- |
| This device may only be used for the applications described in the catalog or the technical description and only in connection with devices or components from other manufacturers which have been approved or recommended by Siemens. Correct, reliable operation of the product requires proper transport, storage, positioning and assembly as well as careful operation and maintenance. |

## Trademarks

All names identified by ® are registered trademarks of the Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

## Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Automation and Drives
Postfach 48 48
90327 NÜRNBERG
GERMANY

Ordernumber: 6FC5397-8CP10-1BA0
℗ 02/2008

# Table of contents

# 1 Commissioning a System

## 1.1 Delivery condition of the system

### Scope of validity

This document is valid for all systems that are supplied with Linux as operating system on the CF card, such as all NCUs of the type NCU 7x0, PCU 321 etc.

### Structure and contents of the CF card

A Linux partition as well as a FAT partition are available on the CF card. The memory space of the CF card is occupied mainly by the LINUX EXT3 partition containing the system software and the user data. A 2MB large FAT partition still exists but only for internal administration purposes.

When the system is in operation, the following directories (this is a selection and not a complete list) are available in the EXT3 partition on the CF card:

| Directory | Use |
|---|---|
| /siemens | Reserved for Siemens system software |
| /addon | Reserved for Siemens add-on software |
| /oem | Additional software and configurations of the machine manufacturer |
| /user | • Storage of user data<br>• Configurations of the HMI<br>• Data that appear on the user interface when the machine is started up |
| /system | Linux operating system |
| /user/system/etc | File basesys.ini (modifications possible) |
| /user/common/tcu | TCU configuration files |
| /var/log/messages | System log file (same as event.log under Windows) |

Files in the directory under /user always have priority over files with the same name in the directory /oem → /addon → /siemens.

| CAUTION |
|---|
| **Suitable editors for Linux** |
| In most Linux system files, lines may only be ended with LF, and not with CRLF as in Windows. Please take note of this when selecting an editor. The HMI editor under "Commissioning" is suitable. |
| In the Linux operating system, the UNIX Editor vi is available. |
| Please take note that the Linux operating system is case-sensitive. |

### Preset users

The following users have already been set up:

| Users | Password | Target group |
|---|---|---|
| operator | --- | Operators, users |
| user | CUSTOMER | Operators, users |
| service | EVENING | Service personnel |
| manufact | SUNRISE | Machine manufacturer |

## 1.2 System booting

### Sequence

To ensure unproblematic booting of the NCU, the CF card must be inserted.

When the NCU is booting up, visual information on the current operating system is provided using the following displays:

- The RDY-LED flashes slowly yellow when the CF card is accessed.
- During booting, the 7-segment display outputs different codes that indicate, for example, when the BIOS is started, when the CF card is accessed, etc.

When the booting has been completed successfully, the following is displayed:

- The PLC LED lights up green.
- The 7-segment display shows "6." with a flashing dot.
- The RDY-LED and all other LEDs are not illuminated.

### Performing a reset operation

The reset button is located behind the blanking plate of the NCU.

A reset operation resets the entire system and requires a system restart. This is comparable to a "Power On reset" except that the 24 V power supply does not have to be switched off.

### Booting for servicing purposes

For service or diagnosis purposes, the NCU can be booted from a service system, the Emergency Boot System.

### See also

How do you create a service system for the NCU? (Page 12)

## 1.3 Displays during system booting

### States of the RDY LED

Of the LEDs on the front of the NCU, only the RDY LED and its status is important when booting up the NCU.

### BIOS power-up

| | |
|---|---|
| RDY LED: | Yellow |
| Other LEDs: | All yellow, set by PLC/option module |
| Meaning: | No boot device was found: Code 1F or Code FF. |
| Cause: | The CF card is defective or not bootable, or it does not contain any system software. |

### Downloading of operating system

| | |
|---|---|
| RDY LED: | Slowly flashing red (0.5Hz) |
| Other LEDs: | Set by PLC/option module |
| Meaning: | Loading the operating system involves three phases, which are displayed on the 7-segment display. |

## Booting up the kernel

| | |
|---|---|
| RDY LED: | Slowly flashing yellow (0.5Hz) |
| Other LEDs: | Set by PLC/option module |
| 7-segment display: | 1 |
| Meaning: | Phase 2: Driver initialization |

## Booting up the basic system

| | |
|---|---|
| RDY LED: | Slowly flashing yellow/green (0.5Hz) |
| Other LEDs: | Set by PLC/option module |
| Meaning: | Phase 3: Initialization of the basic system |

## NRK/NCK outputs

| | |
|---|---|
| RDY LED: | OFF (if fault status: red) |
| Other LEDs: | Set by PLC/option module |
| Meaning: | After the basic system, the NRK/NCK takes over the LED and 7-segment display. |

## System error

| | |
|---|---|
| RDY LED: | Rapidly flashing red (2Hz) |
| Other LEDs: | Set by PLC/option module |
| Meaning: | An error has occurred.<br>The system is stopped. |

| | |
|---|---|
| RDY LED: | Rapidly flashing red/yellow (2Hz) |
| Other LEDs: | Set by PLC/option module |
| Meaning: | An error has occurred.<br>The system has continued running - although with restricted functions. |

## 1.4 Meaning of switch positions

### Overview

The NCU has two twist buttons in the lower section of the front panel.

- NCK start-up switch with label SIM/NCK
- PLC mode selector switch with label PLC

The switch positions are marked with even numbers or letters; the uneven numbers are represented by dots due to space limitations.

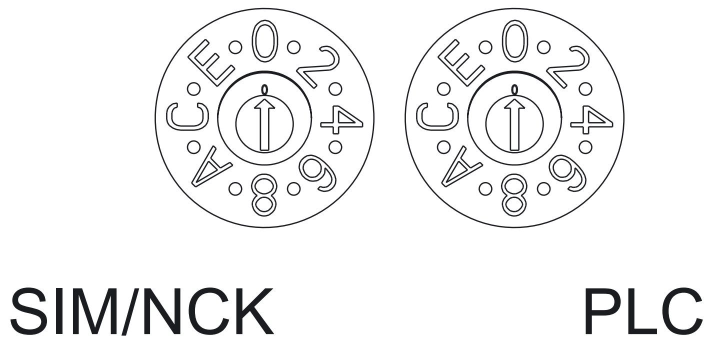



Figure 1-1 Startup and mode selector switch

### SIM/NCK twist button

The switch positions of the SIM/NCK switch have the following meaning:

| Switch position | Operating mode of the NCK |
|---|---|
| 0 | Normal boot-up NCK |
| 1 | NCK boot-up with default values (= memory reset) |
| 2 | The NCK (and PLC) starts up with the data that was saved at the last shutdown. |
| 7 | Debug mode (NCK is not started.) |
| 8 | IP address of the NCU is displayed on the seven-segment display. |
| All others | Not relevant |

### PLC twist button

The switch positions of the PLC switch have the same meaning as in a SIMATIC S7-CPU:

| Switch position | Operating mode of the PLC |
|---|---|
| 0 | RUN |
| 1 | RUN (protected mode) |
| 2 | STOP |
| 3 | Memory reset (MRES) |
| All others | Not relevant |

# 1.5 Ethernet interfaces of the NCU

## Supplementary conditions

For the operation of an NCU:

- No more than one NCU may be operated as a DHCP server on the system network.
- An additional external keyboard is required for uppercase/lowercase letters.

## Configuration of the interfaces

The following connections can be established via the Ethernet interfaces:

| Interface | Labeling | Internal name | Terminal settings |
|---|---|---|---|
| Ethernet (HMI) IE1/OP | X120 | (Eth 2) | Connection to the system network with preset IP address 192.168.214.1 with subnet screen form 255.255.255.0 and active DHCP server for SINUMERIK |
| Ethernet IE2/NET | X130 | (Eth 1) | Connection to company network as standard DHCP client |
| Ethernet (PLC) | X127 | (Ibn 0) | Service terminal with fixed IP address 192.168.215.1 and fixed subnet screen form 255.255.255.224 with active DHCP server |

## Network interface

The network interface is an interface that enables network communication. These are the Ethernet interfaces on the NCU.

## VNC (Virtual Network Computing)

Virtual Network Computing is a software that displays the screen contents of a remote computer, with a running VNC server, on a local computer, with a running VNC viewer, and in return sends keyboard and mouse movements of the local computer to the remote computer.

**Reference:** Operator Components and Networking Manual

# Backing up and Restoring Data 2

## 2.1 Creating a service system

### 2.1.1 Applications

#### Overview

To back up and restore the CF card data, proceed as follows:

- Create a service system
- Back up onto a service system
- Restore from the service system

As an alternative to backing up the data on a service system:

- Back up onto a network drive
- Restore from the network drive

To call a service shell, depending on the configuration, the following possibilities are available:

`(I)` Configuration of NCU with TCU: the service shell is called under Linux.

`(II)` Configuration of NCU with PCU 50.3 or programming device (PG):

the service shell can be called under Linux or alternatively also under Windows via WinSCP.

**Reference:** Operator Components and Networking Manual

### 2.1.2 How do you create a service system for the NCU?

#### Purpose

In case servicing is needed, create a portable "Emergency Boot System" (EBS) on a USB memory. Thus you can start the booting of the NCU from the service system in order to carry out various service tasks, such as data backup or updates, in a service shell.

Two partitions are created on the service system:

- a Linux partition that is invisible under Windows.
- an FAT partition for DOS or Windows applications.

  The FAT partition can be addressed using the path /data, and can be read and written to under Linux and also from a Windows system.

#### Scope of delivery

To create a service system on a USB storage medium with 512 MB storage capacity, the following files are included on CD:

- an executable file installdisk.exe
- an image file linuxbase-512M.img for USB-FlashDrive with 512 MB
- an image file linuxbase-resize.img for USB-FlashDrive > 512 MB
- a file with the newest information siemensd.txt

#### Recommendation:

Preferably, the SIMATIC PC USB-FlashDrive with 512 MB storage capacity should be used.

---

Note

To create the service system, you need administrator rights.

All data already on the USB storage medium will be deleted.

The transfer is optimized for USB 2.0; therefore, the transmission to the USB storage medium takes longer when using USB 1.1 than USB 2.0

---

#### Proceed as follows

To create a service system on a 512 MB USB storage medium:

1. Copy the service system onto a local hard disk of your programming device (PG) or PC.
2. Connect a 512MB USB storage medium to the USB interface of the PG/PC.
3. Determine in Windows Explorer which drive letter the USB storage medium was assigned, e.g. H:
4. Open a DOS shell and change to the directory in which the files for the service system are stored.

5. In the DOS shell, enter the following command:

   ```
   installdisk --verbose --blocksize 1m linuxbase-512M.img h:
   ```

   **Result:**

   The image is transferred to the USB storage medium; a partition for Linux and a FAT partition for Windows systems are created.
6. Disconnect the USB storage medium and connect it again.

   **Result:**

   After this has been completed successfully, you will have a bootable service system on the USB storage medium.

| NOTICE |
| --- |
| **USB storage medium > 512 MB:**<br>If you use a USB storage medium with a storage capacity > 512 MB as service system, there is a further variant, "linuxbase-resize.img", so that the storage capacity of the USB storage medium is retained. |

To create a service system on a USB storage medium > 512 MB:

1. Copy the service system onto a local hard disk of your PG/PC.
2. Connect a 512MB USB storage medium to the USB interface of the PG/PC.
3. Determine in Windows Explorer which drive letter the USB storage medium was assigned, e.g. H:
4. Open a DOS shell and change to the directory in which the files for the service system are stored.
5. In the DOS shell, enter the following command:

   ```
   installdisk --verbose --blocksize 1m linuxbase-resize.img h:
   ```

   **Result:** The image is transferred to the USB storage medium.
6. Connect the USB storage medium to a bootable NCU and boot the NCU from the USB storage medium: Only then can the USB storage medium be used as service system and the entire storage capacity is available.

# 2.2 Data backup on service system

## 2.2.1 This is how you save data on a service system

### Proceed as follows

To back up the complete system:

1. Connect the service system to a USB interface (X125 or X135) of the NCU and press the reset button.

   Alternatively, you can switch the NCU off, connect the service system, and switch the NCU on again.

   **Result:** The NCU boots from the service system.

2. Log on as a service technician using the user name "manufact" and password "SUNRISE".

3. Using the command "sc backup" you can create the backup file "backup01.tgz".

   The directory /data on the service system is provided for backup files. Choose between –full to save all data on the CF card, or –user if you only want to save user data.

   **Example:** `sc backup –full /data/backup01.tgz`

   **Result:**

   A backup file of the complete CF card is created under /data on the service system on the USB storage medium.

## 2.2.2 This is how you restore data from the service system

### Proceed as follows

To restore the complete system:

1. Connect the service system to a USB interface (X125 or X135) of the NCU and press the reset button.

   Alternatively, you can switch the NCU off, connect the service system, and switch the NCU on again.

   **Result:**

   The NCU boots from the service system and the main menu is displayed.
2. Select <F2> (softkey or button on an external keyboard) to open a service shell.
3. Log on as a service technician using the user name "manufact" and password "SUNRISE".
4. With the command "sc restore" you can write the backup file "backup01" from the service system back to the CF card in the NCU.

   Example: `sc restore /data/backup01.tgz`

   **Result:**

   The system state stored in the file "backup01" is restored on the NCU.

---

**Note**

If access to the system data on the CF card is not possible because the CF card is defective or empty, you can only log in as user "admin" with the password "SUNRISE" and no longer as the user "manufact".

---

# 2.3 Data backup on network drive

## 2.3.1 This is how you save data on a network drive

### Sequence

Proceed as follows:

- Establish a connection to a network drive
- Define a MOUNTPOINT
- Create a backup file

### Scenario I: Start Command Shell under Linux

1. Switch to the VNC Starter using the key combination <Recall+MENU SELECT> (area switchover key) or with <F9+F10>:

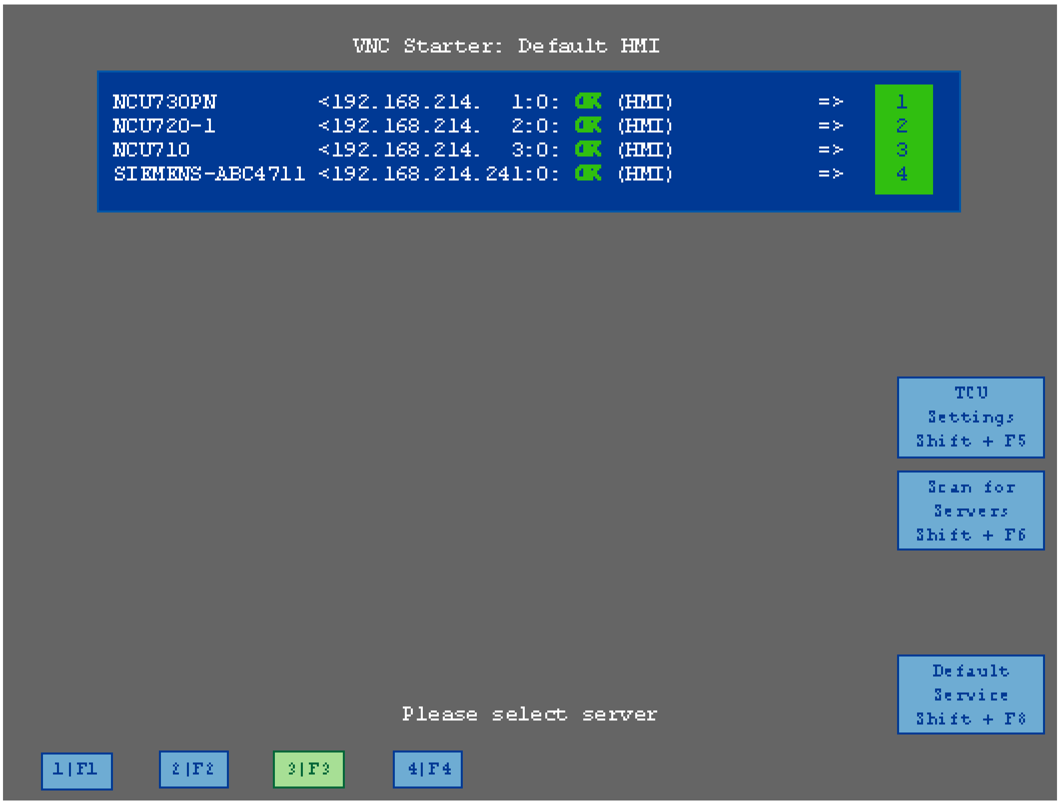



Figure 2-1 VNC connections

2. If only VNC connections to the HMI are displayed, press the "Scan for servers" softkey or <Shift+F6> to display further connections:

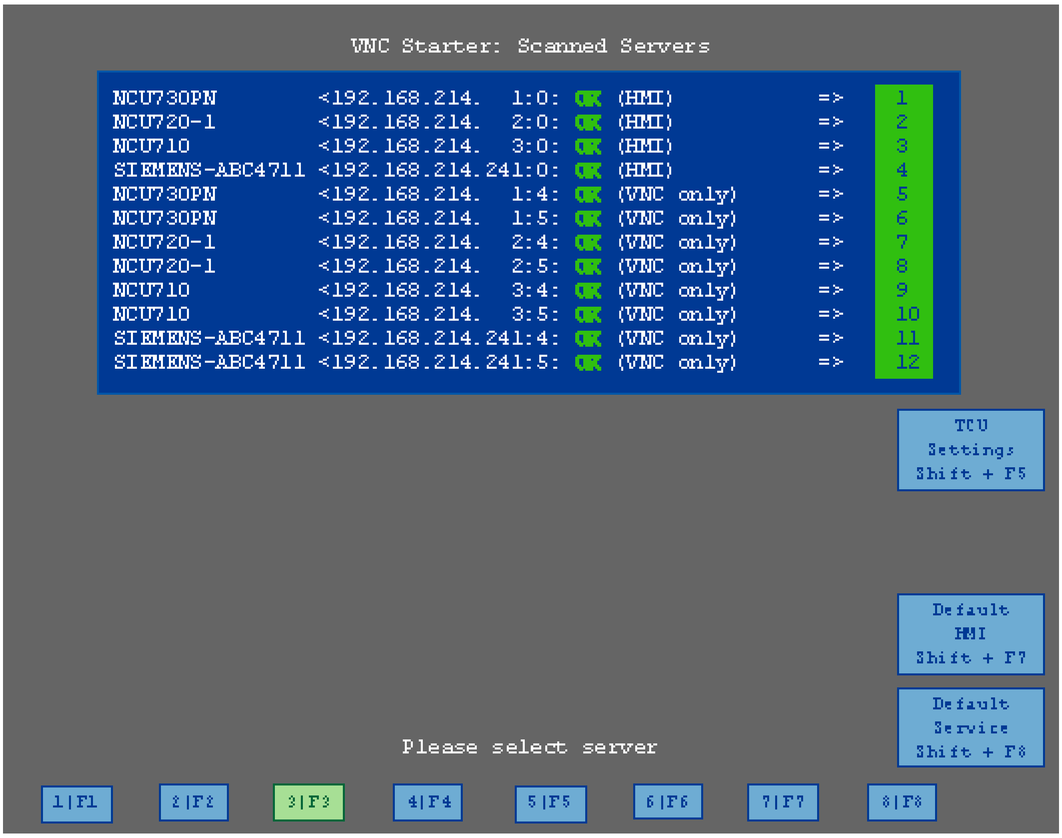



Figure 2-2 Open service shell

3. Start a service shell by using the assigned softkey to select "Session 4". In "Session 5", the log console is output.
4. Log on as a service technician using the user name "manufact" and password "SUNRISE".
5. Create a new "backup" directory under /tmp on the CF card, which you will then use as `MOUNTPOINT`.

   Example: `mkdir /tmp/backup`
6. Use the command "sc connect" to connect the network drive:

   `sc connect //username%password@server/share /tmp/backup`

   Please enter here the user name and the password for the network drive that is to be connected to.
7. Using the command "sc save", create the backup file "backup01".

   Choose between –full to save all data on the CF card, or –user, if you only want to save user data in the directory /user.

   Example: `sc save -full /tmp/backup/backup01`

   **Result:**

   A backup file of the complete CF card is created under the specified path on the network drive.

### Scenario `II`: Start the Command Shell with WinSCP on the PG:

1. Start WinSCP and enter the following data in the log-on window:
   – IP address of the NCU (or, if required, host name)
   – User name "manufact" with password "SUNRISE".
2. Select in the menu "Commands" → "Open Terminal".
3. Execute the data backup using the commands described in steps 4 to 6 from scenario `I`.

### See also

Connect (Page 27)

Save (Page 35)

## 2.3.2 This is how you restore data from the network drive

### Sequence

Proceed as follows:

- Stop all subsystems
- Establish a connection to a network drive
- Define a MOUNTPOINT
- Restore the data
- Start all subsystems again

### Scenario I: Start Command Shell under Linux

1. Before you can restore backup files, you need to stop the subsystems, e.g. NCK.
2. Switch to the VNC Starter using the key combination <Recall+MENU SELECT> (area switchover key) or <F9+F10>: A list of VNC connections that can be selected is displayed.
3. Start a service shell by using the assigned softkey to select "Session 4". In "Session 5", the log console is output.
4. Log on as a service technician using the user name "manufact" and password "SUNRISE".
   ```
   sc stop all
   ```
5. Use the command "sc connect" to connect the network drive:
   ```
   sc connect //username%password@server/share /tmp/backup
   ```
6. Please enter here the user name and the password for the network drive that is to be connected to.

   To completely restore the system (system data and user data), enter the following:
   ```
   sc restore –full backup01
   ```
   **Result:** The whole system is overwritten by the backup data.

7. To restore user data only, enter the following command:

   `sc restore –user backup01`

   **Result:** The user data are written back again.

   All subsystems are then restarted:

   `sc start all`

   **Result:**

   The system state stored in the file "backup01" is restored on the NCU.

## Scenario II: Start the Command Shell with WinSCP on the PG:

1. Start WinSCP and enter the following data in the log-on window:
   - IP address of the NCU (or, if required, host name)
   - User name "manufact" with password "SUNRISE".
2. Select in the menu "Commands" → "Open Terminal".
3. To restore the data, enter the corresponding commands as described under steps 3 to 5 in scenario `I`.

## See also

Connect (Page 27)

# Service Commands 3

## 3.1 Using service commands

### Overview

The service command 'sc' is a tool used for performing a range of service tasks on a SINUMERIK NCU. The required action is written in the command line after the 'sc', e.g.: sc help

This action produces a list of all actions with a short description. After the action, further parameters or options can follow.

'sc' is available in both the NCU basic system and the service system (mostly in the form of a USB memory.) However, some actions are only useful in one of the two systems (NCU/Service). This is noted for each action.

### See also:

The most important terms and abbreviations are explained in the glossary.

### Privilege levels

The 'sc' command can execute its actions using more privileges than are normally granted to the calling user. For example, starting or stopping subsystems requires root privileges, but 'sc' allows every user who belongs to the ‘service’ group to do this.

Each action of 'sc' is assigned a “privilege level”. This is a user group to which the user must belong in order to execute the action. As the groups are hierarchically nested, members of “higher” groups can also use the action in question. For example, the group 'manufact' is above 'service', which means that members of the group 'manufact' can call all actions that require the privilege level 'service'.

The privilege level that the caller requires is noted for each individual action. The possible levels are (in ascending order):

- none
- operator
- user
- service
- manufact

**Example:**

An action with the privilege level 'user' can also always be executed by members of the groups 'service' and 'manufact'. Actions with privilege level ‘none’ can be called by all users.

If a user does not have the required privileges, the following error message is output:

```
Action 'ACTION' needs at least GROUP privilege level.
```

## 3.2 Syntax for the Actions

### Description

'sc' in the command line is essentially not case-sensitive.

The following entries are therefore interpreted as identical:

**sc help show**

**SC help SHOW**

**sc HeLp sHoW**

In some cases, however, upper/lower case can make a difference, for example in file or user names. This is avoided wherever possible.

The conventions used are as follows:

- Names completely in upper case represent objects to be used depending on the situation.

  Example: `sc help ACTION`

  In this case, ACTION is to be replaced by the action for which you want a description. If written in lower case, however, the entry should be entered as specified.
- Square brackets indicate optional entries.

  Example: `sc help [ACTION]`

  In this case the specification of an action is optional, which means that you can enter an action, but do not have to. Square brackets may also be nested:

  ... [USERNAME[/DOMAIN]] ...

  In this case, USERNAME and DOMAIN are both optional, but you can only enter a DOMAIN if you have also entered the USERNAME.
- Alternatives are separated with '|'.

  Example: `sc start all|system|SUBSYSTEM`

  This means that any of the following commands applies:

  ```
  sc start all
  sc start system
  sc start SUBSYSTEM
  ```

  In the latter case, SUBSYSTEM in upper case can be replaced by a concrete subsystem name.

- As a shortened form, alternatives can also be written in square brackets:

  ```
  sc save [-full|-user] ...
  ```

  Here you can use the option "-full" or "-user", or none at all.
- Options that begin with '-' can always be entered in any order.

  For example, the notation could be interpreted to mean that "-force" must come after "-full" or "-user", but this is not necessarily the case:

  ```
  sc save [-full|-user] [-force] FILENAME
  ```

## 3.3 Generic elements

### Overview

This sections describes the syntax elements that are used by several actions.

### Permitted interface designations

The names of network interfaces are used by "sc show ip", "sc set ip" and "sc enable DHCPSvr", for example.

As the input of an interface is optional, it is always introduced by a '−' character. The '−' is followed by the actual name. In most cases, multiple names are possible for the same interface.

Accepted names are:

| | |
|---|---|
| System network port: | "X120", "eth2", "tcu", "internal" |
| Company network port: | "X130", "eth1", "factory", "extern |
| IBN port: | "X127", "ibn0", "pg" |

### Permitted subsystem designations

Subsystem names are entered when using "sc enable" and "sc start", for example. In most cases, the name of the subsystem is simply the name of the corresponding CFS, without the path and without the extension ".cfs". For example, for the CFS /siemens/sinumerik/nck.cfs, the relevant subsystem name is simply "nck".

It is also possible to use absolute paths (beginning with /) in a subsystem name. In the example above, you could also use "/siemens/sinumerik/nck" as a subsystem name. The difference between names with and without a specified path is that without the path, all CFS with this name are included, but if the path is specified, only this exact CFS is meant.

### Subsystem

A subsystem is a CFS that not only contains a collection of files, but also executes a program, for example, at runtime. To do this, the CFS contains a script that is used to control the starting and stopping of this program.

For this reason, only administrators are permitted to set up NFS file systems, and NFS is usually only implemented in uniformly administrated environments. Exported file systems on the server are addressed directly on the server via their path.

### CFS (Compressed File System)

A CFS (file extension ".cfs") is a compressed file system, similar to a zip file. It contains files and subdirectories that look like normal files on the controller at runtime. Files and directories contained in a CFS cannot be changed. They are decompressed at runtime as required.

### NFS (Network File System)

NFS is the most common protocol for remote file systems in the world of Unix, and is also available for Windows. NFS is closely based on the Unix privilege model – each time a file is accessed, a UID and GID are supplied which the server then uses to decide whether the operation is permitted. The server relies on the client to provide the correct IDs.

### Remote File System

A file system that is contacted over the network. The files are physically located on another computer in the network (the "server"), but appear locally the same as all other files. Operations performed on these files are sent via the network to the server, instead of being executed directly on a local storage medium (such as a hard drive or CF card).

As a server usually exports more than one file system, a name for the required file system must also be entered in addition to the name of the server.

### SMB (Server Message Block)

SMB is the underlying protocol of MS Windows file systems (also known as drives, releases, shares, etc.). SMB connections are always active in the context of a specific user, who must be known to the server. Exported file systems have a name (release name), by which they can be addressed. The client does not need to know the concrete path on the server.

# 3.4 Description of the actions

## 3.4.1 Help

### Description

| | |
|---|---|
| Syntax: | sc help [Action] |
| Alternative names: | -h, --help |
| Privilege level: | none |

The call of "sc help" without any additional action outputs a list of possible actions with a short description. If you enter an additional action, you receive a more detailed description for this action.

**Examples:**

```
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 CFSs 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'.
```

## 3.4.2 Check-cf

### Description

| | |
|---|---|
| Syntax: | sc check-df |
| Alternative names: | checkcf |
| Privilege level: | user |

With this action, the CF card is checked reading to see whether it contains defective sectors. If errors occur, this is noted in the file /var/log/messages.

## 3.4.3 Clear

### Description

Various actions can be performed with the "clear" command; only the "clear dhcp" command is relevant here.

### clear dhcp

| | |
|---|---|
| Syntax: | sc clear dhcp [-INTERFACE] |
| Alternative names: | --- |
| Privilege level: | service |

This command clears any state of the DHCP server at the specified interface (default setting is the system network) and resets it to its initial state. This means that the Lease data is deleted: all IP addresses are re-assigned in the network and the server forgets having ever seen a master server.

This action is only appropriate on a machine on which the active DHCP server runs.

---

**Note**

The clearing of the Lease data does not delete the entire file, but only the data contained therein. This action also increments the version number, so that available standby DHCP servers can also perform the deletion.

---

### 3.4.4 Closeport

#### Description

| | |
|---|---|
| Syntax: | sc closeport ID |
| Alternative names: | --- |
| Privilege level: | service |

#### Mode of operation

This command closes a port in the firewall opened previously with the "sc openport" command. This is only necessary when the port is to be closed manually before the time set with "sc openport" expires. Otherwise the port is closed automatically when the port service life expires.

ON parameter is the ID number of the firewall rule output by "sc openport".

### 3.4.5 Connect

#### Description

| | |
|---|---|
| Syntax: | sc connect [-ro] SERVER:/PATH [MOUNTPOINT] |
| | sc connect [-ro] [-public] //[USERNAME[/DOMAIN] [%PASSWORD]@]SERVER/SHARE [MOUNTPOINT] |
| Alternative names: | mount |
| Privilege level: | none |

This action makes a remote file system on a server available on the controller. This is enabled by linking the remote file system to a local directory, known as the "MOUNTPOINT". The files offered by the server are then visible under this directory.

| NOTICE |
|---|
| Note that when entering this command, the password appears on the screen in plain text according to the specified syntax.<br>For path names, use the slash “/” and not the backslash “\”. |

## Supported file systems

Two types of remote file system are supported: Windows SMB and Unix NFS. These two systems have completely different characteristics, particularly in terms of user administration:

- In Windows SMB, you connect to the server as a particular user that the server must recognize. Via this connection, you then access the files as this user, independently of which local user triggers the action.

  This feature means that in SMB systems, you already have to enter a user name, if necessary its domain, and a password at the time of connection.

- In NFS servers, the connection itself does not require a particular user to be entered. Instead, for each file operation, the user who wants to carry out the operation must log on to the server. The server then decides whether or not to permit this. Users are entered using a user ID and group ID, not with names. The server must therefore recognize the corresponding IDs (or permit access for all users.)

  Another type of remote file system that is supported are USB memory devices exported from TCUs (USB Flash Drive). Since these are integrated using NFS, the entries for server and path are similar to those for NFS. However, the TCU names are administrated differently, and the USB memory devices have specific paths that do not physically exist.

## Notation of the remote file system

For SMB and NFS/TCU, the file system is entered using different notations:

## SMB: //[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE

The fixed share is: //SERVER/SHARE

The server name can of course also be a numeric IP address. SHARE is the name of the release on the server. Note that the character '$', which often occurs in this type of name, must be preceded by a backslash ('\') in the command line. Otherwise, the system tries to expand a variable.

A user name can also be entered in front of the server name, separated with a '@' character. If necessary, the user name can also be extended by adding '/' followed by the Windows domain to which it belongs. The password belonging to this user is normally queried interactively, so that it is not visible on the screen. ´

For some applications, however, it may be necessary to write the password on the command line. (For example, programs started from WinSCP cannot read from the keyboard.) In this case, you can append an additional '%' character, followed by the password. If this contains any special characters that are interpreted by the shell (<, >, &, ;, ", ', $, (, ), |), you should shield these by preceding them with a backslash. Commas in SMB passwords cannot be interpreted.

### NFS: SERVER:/PATH

In NFS, entering the user is not required, which just leaves the two components SERVER and PATH. As before, SERVER is the name of the file server. In contrast to SMB, PATH is a path name that exists on the server, and not an arbitrarily assigned name.

The command "sc show drives SERVER" shows the file systems offered by a particular server in the correct notation. For SMB, if required, the user name etc. still need to be added.

After the entry of the remote file system, you can also specify the required directory ('MOUNTPOINT') in which the remote files will become visible locally. This should be an empty directory to which you have write access. If MOUNTPOINT is omitted, 'sc' itself generates a suitable directory. The name is /tmp/drvNN (numbered consecutively), and is displayed when the connection is successful.

The option "-ro" connects the remote file system in read-only mode, which means that you can only read and do not have write access. This can be useful if you want to exclude changes that originate from the controller, or if the server only allows read-only connections.

As explained above, an SMB file system is connected as the specified user. To prevent a third user from executing file operations on the controller in the name of the connected user, the local directory can normally only be accessed by the latter (and all members of the same group).

However, if you do want to make the remote directory available to further local users, you can use the option "-public" to make the MOUNTPOINT belong to the "operator" group, and this access is therefore guaranteed.

## 3.4.6 Disable

### Description

| | |
|---|---|
| Syntax: | sc disable hmi\|nck\|SUBSYSTEM ... |
| | sc disable DHCPSvr -INTERFACE |
| | sc disable DHCPSync [-X120] |
| | sc disable IPAlias [-X120] |
| Alternative names: | --- |
| Privilege level: | service |

The "sc disable" command switches the specified subsystems off. These subsystems are then not loaded again at the next restart. You can also switch several subsystems off with a single "disable" command.

"hmi", "nck" or any other subsystem names can be used for the subsystems.

"sc disable IPAlias" deactivates the second alias IP address for the X120 network interface.

### See also:

Parameters "InternalIP_Alias" and "InternalNetMask_Alias" in the "basesys.ini" file.

### Specific subsystems

The name "DHCPSvr" is a special case. This does not switch a normal subsystem on or off, but specifies whether or not a DHCP server should be started on a network interface. The interface is entered in the standard form.

The synchronization of the DHCP server is switched on and off with "DHCPSync". The synchronization is only possible in the system network.

The commands "sc enable" and "sc disable" function by changing certain variables in the file /user/system/etc/basesys.ini. In subsystems, this variable is "DisableSubsystems", in DHCPSvr "DisableDHCPD<INTERFACE>".

## 3.4.7 Disconnect

### Description

| | |
|---|---|
| Syntax: | sc disconnect MOUNTPOINT |
| | sc disconnect all |
| Alternative names: | umount, unmount |
| Privilege level: | user |

The action "disconnect" terminates the connection to a remote file system previously connected using "connect“. If the MOUNTPOINT has been automatically created by 'sc', the corresponding directory is also deleted again.

The variant "sc disconnect -all" can be used to remove all existing SMB and NFS file systems.

## 3.4.8 Distribute

### Description

| | |
|---|---|
| Syntax: | sc distribute [parameter] |
| Alternative names: | dist |
| Privilege level: | service |

This command distributes TCU data to other machines in the system network. Only the parameter "tcudata" is relevant here.

### distribute tcudata

| | |
|---|---|
| Syntax: | sc distribute tcudata |
| Alternative names: | --- |
| Privilege level: | service |

This command informs the active DHCP server that TCU data in /user/common/tcu has been manually changed. The server can then distribute this data to the standby servers.

It is recommended that the version file in /user/common/tcu be changed manually as this is only read in once by the DHCP server at the start. An incrementing of the version would then only take effect at a restart.

---

**Note**

This command can be performed on any machine in the system network, not only on the active server. A message is always sent to the active server.

---

### 3.4.9 Enable

#### Description

| | |
|---|---|
| Syntax: | sc enable hmi\|nck\|SUBSYSTEM ... |
| | sc enable DHCPSvr -INTERFACE |
| | sc enable DHCPSync [-X120] [-PRIORITY] |
| | sc enable IPAlias [-X120] IPADDR[/NETMASK] |
| Alternative names: | --- |
| Privilege level: | service |

The "sc enable" command switches on the specified subsystems such that these are also activated when a subsystem is started. You can also switch several subsystems on with a single "enable" command.

"hmi", "nck" or any other subsystem names can be used for the subsystems.

"sc enable IPAlias" defines a second alias IP address for a network interface. This is only supported for X120. The IP address itself has the usual notation with four numbers separated by decimal points. If not specified, the subnet screen form can be determined from the class (A/B/C) of the IP or determined explicitly. The CIDR notation (number of bits of the network section) is also possible.

#### See also:

Parameters "InternalIP_Alias" and "InternalNetMask_Alias" in the "basesys.ini" file.

#### Specific subsystems

The name "DHCPSvr" is a special case. This does not switch a normal subsystem on or off, but specifies whether or not a DHCP server should be started on a network interface. The interface is entered in the standard form.

The commands "sc enable" and "sc disable" function by changing certain variables in the file /user/system/etc/basesys.ini. In subsystems, this variable is "DisableSubsystems", in DHCPSvr "DisableDHCPD<INTERFACE>".

## DHCP synchronization

The synchronization of the DHCP server is switched on and off with "DHCPSync". The synchronization is only possible in the system network (X120). A priority can also be specified. LOW, HIGH or MASTER.

The priorities have the following effect for the synchronization of the DHCP server:

- **MASTER:** The computer node is defined as master, i.e. the computer becomes the active DHCP server. If several masters have been configured in the system network, this computer has the highest priority.
- **HIGH:** The computer node belongs to the server candidates with high priority, i.e. if no server with master priority becomes active, then a computer with "HIGH" priority can be the active server.
- **LOW:** The computer node belongs to the server candidates with low priority, i.e. if no server with master priority or "HIGH" priority becomes active, then a computer with "LOW" priority can be the active server.

---

**Note**

Recommended settings are:

- DHCP operation and DHCP synchronization are switched on in the system network.
- **Exactly one** NCU is set as DHCP master.
- **Maximum of two** computers are candidates with "HIGH" priority.
- All other components are set as DHCP clients or candidates with "LOW" priority.

---

## See also:

Parameter "SyncModeDHCPD_SysNet" in the "basesys.ini" file.

### 3.4.10 Openport

#### Description

| | |
|---|---|
| Syntax: | sc openport [-MINUTES] PROTO/PORT SOURCE ... |
| Alternative names: | --- |
| Privilege level: | service |

#### Mode of operation

This command opens a port in the firewall to the company network (X130) for a certain time. The default time is 15 minutes, but this can be changed with the -MINUTES option. The maximum possible time is 60 minutes.

The port to be opened is specified in the form "PROTO/PORTNR". The protocol can be either "tcp" or "udp". This is followed by the specification of the hosts from which the port should be accessible. Several forms are possible here:

- A single IP address: "128.128.12.12"
- A host name (if this can be resolved via DNS), e.g. "server"
- An IP area with network screen form specification, e.g. "128.128.12.0/255.255.255.0"
- An IP area with specification of the valid bits, e.g. "128.128.12.0/24"

If the specification of the source host is completely missing, this is normally acknowledged with an error message. However, an exception is when the "sc openport" command is entered in a shell opened via SSH. The permitted host is taken from the environment variable $SSH_CLIENT that contains the IP of the SSH client.

If the opening was successful, an ID number is output in a status message:

```
sc: Port tcp/25 is open, rule ID is 6620
```

This ID can be used with "sc closeport" to manually close the port earlier.

#### See also:

Parameter "FirewallOpenPorts" in the "basesys.ini" file.

### 3.4.11 Restart

#### Description

| | |
|---|---|
| Syntax: | sc restart |
| Alternative names: | reboot |
| Privilege level: | service |

The action "sc restart" triggers a controlled shut down of the machine (stops all subsystems and the basic system), and then initiates a restart. The effect is the same as the "reboot" command in the basic system, which can only be executed by "root".

### 3.4.12 Restore

#### Description

| | |
|---|---|
| Syntax: | sc restore [-full\|-addon\|-addon+\|-oem\|-oem+\|-user] [-force] [-nodelete] [-update] [-restart] FILENAME |
| Alternative names: | --- |
| Privilege level: | user |

A backup created using "sc save" can be restored to the controller using "sc restore".

#### Mode of operation

As for "sc save", a restore is only possible from the service system or if subsystems are stopped. If “sc restore" reports an error, for example that the restore will cause running subsystems to crash, you can use the option "-force" to force the operation – just as for “sc save”.

Normally, "sc restore" deletes the whole destination area before the backup is restored (complete CF card for full backup, /user in user data backups). This means that no files are subsequently available that were not included in the backup.

### Options

If no further options are specified, the archive should represent a full backup and this should be restored in full. The status of all files is therefore the same after the "restore" as it was at the time of the backup.

- The -full option additionally forces the partition and the file system to be recreated on the CF card. This is only possible from a service system however. -full is necessary if the partition table and/or file system is missing or damaged.
- However, if you want to restore the backup files without losing any files that have been created in the meantime, you can use the option "-nodelete" to prevent these from being deleted. "-nodelete" is not executed together with "-full", as when the file system is recreated, all files are essentially deleted.
- The options -addon, -oem, -user, -addon+ and -oem+ also allow you to unpack only parts of an archive.
- The -update option is used for loading software updates supplied by Siemens. The -restart option can also be useful with -update. If user-defined system files are changed when loading an archive, a restart or reinitialization is necessary. In such cases, "sc restore" outputs a message to that effect at the end of the operation. The -restart option causes the required action to be performed automatically.

## 3.4.13 Save

### Description

| | |
|---|---|
| Syntax: | sc save [-full\|-addon\|-addon+\|-oem\|-oem+\|-user]<br>[-force] [-update] FILENAME |
| Alternative names: | backup |
| Privilege level: | user |

The call of "sc help" without any additional action outputs a list of possible actions with a short description. If you enter an additional action, you receive a more detailed description for this action.

The action "Save" or “Backup” creates a backup of the CF card in FILENAME. If the action is used from a service system, the backup receives the files of the underlying controller, not of the service system itself.

## Options

The options are used for selecting which directories of the file system are to be written to the archive:

- full: complete backup (default): all files of the controller, incl. Boot Loader
- addon: only /addon directory
- oem: only /oem directory
- user: only /user directory (user data)
- addon+: /addon, /oem and /user
- oem+: /oem and /user

If "sc save" is used directly on the controller and the subsystems are running, this may lead to inconsistencies between saved files, because they can still change during the backup process. Therefore, in this case “sc save" normally ends in an error message to inform you that subsystems are still running. If you want to create a backup anyway, you can use the option "-force". "sc save" still issues a warning, but the process continues.

If "sc save" is started from the service system, no subsystems of the controller are running and "-force" is not necessary.

**Example:**

```
sc save -user /tmp/drv01/backup.tgz
```

## 3.4.14 Show

### Description

| | |
|---|---|
| Syntax: | sc show ip [-INTERFACE] |
| Alternative names: | ---- |
| Privilege level: | none |

In the "sc show" command, various displays are grouped together showing the status of the system.

### show ip

| | |
|---|---|
| Syntax: | sc show ip [-INTERFACE] |
| Privilege level: | none |

This command displays the IP address data of network interfaces. You have the option to enter a specific interface. If no particular interface is specified, the data for all existing interfaces is displayed, as well as the Default Gateway.

**Example:**

```
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
```

The following data is displayed for each interface:

- **Name:** the name of the connection socket (X1_ _), together with the name used by the operating system in brackets (ethN or ibnN).
- **"configured":** the IP address configured in the basesys.ini (variables ExternalIP/ExternalNetMask for X130, InternalIP/Internal-NetMask for X120), or "(default)" if nothing is configured in the basesys.ini, or "DHCP" if the address was sourced via DHCP.
- **"current":** the currently set IP address together with the network screen form and MAC address of the interface.
- **"DNS Name":** Result of a DNS reverse lookup on the current IP address.
- **"Nameserver":** Here the list of DNS servers is output which are related to this interface (receive e.g. DHCP via this interface).
- **"DNS Suffix":** DNS search suffix related to this interface.
- **"DHCP":** Indication is given here if a DHCP client or server is running for this interface. In the case of a client, the server where the IP address comes from is displayed, too. A DHCP server on X120 can additionally be synchronized on the system network. Then the information is displayed if this is the active server or if it is in standby mode.

- **"Statistics":** Total amount of data received or sent via this interface, and the percentage of faulty packages.

When all interfaces are displayed, the default gateway is also output, i.e. the address of a router to which all packages are sent that cannot reach their destination directly via a local interface.

The default gateway is therefore a piece of data that applies to all interfaces and of which there is only one instance. However, there is one interface via which it must be possible to address the default gateway. This is displayed in brackets after the address.

## show drives

| | |
|---|---|
| Syntax: | sc show drives SERVER |
| Privilege level: | none |

"sc show drives" shows the available remote file systems for a particular server. The server name SERVER can be an NFS server, an SMB server or "TCU", which represents any USB media that are connected to a TCU. For details of possible server names and entering a user name (often required for SMB), see the description of the "sc connect" command.

## Examples

Notation:

```
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
```

All the available remote file systems of the relevant server are listed in the notation that is expected by "sc connect".

- SMB shared drives on Windows servers always begin with "//", followed by the server name and the name of the shared drive. In NFS file systems, the server name is always first, followed by a colon and then the export path.

  Connection to SMB servers usually requires the entry of a user name (with domain, if applicable) and a password.

- USB memory devices on TCUs are a special form of NFS, and therefore also have the same notation as NFS file systems. The specified path does not exist physically on the TCU, but is converted there to USB by the NFS server.

### 3.4.15 Start, Stop

#### Description

| | |
|---|---|
| Syntax: | sc start all\|system\|SUBSYSTEM |
| | sc start all\|system\|SUBSYSTEM |
| Alternative names: | --- |
| Privilege level: | service |

These two actions start or stop individual or all subsystems. Since these actions have a direct influence on the runtime behavior, they are only available directly on the PLC, but not in the service system (with the exception of "sc stop system"). The subsystems of the underlying PLC do not run here, and can therefore not be controlled.

---

**Note**

Only "all" and "system" function, but not arbitrary subsystems.

Both of these actions have only been implemented to a restricted extent.

---

#### Subsystem names

"hmi", "nck", or any other names, can be used for the subsystems. Multiple subsystems can also be specified in a row, which are then started and stopped in this order.

"all" and "system" are special values for the subsystem:

- "all" starts or stops all available subsystems.
- "sc stop system" similarly stops all subsystems, but also the basic system with the PLC.
- "start system" is identical to "start all" and exists mainly for reasons of symmetry.

# 3.5 Service tools WinSCP and PuTTY

## Use

The programs WinSCP and PuTTY are freely-available open source programs for Windows. WinSCP is intended especially for transferring files from and to Linux systems, PuTTY for the remote operation of a command shell.

WinSCP and PuTTY are included with the package ‘PCU Basesoftware Thin Client for PCU 50 V07.05.00.00 and higher'. With a PCU 50.3, this package is pre-installed. When installing HMI Advanced on PG/PC, WinSCP is automatically installed, too.

- WinSCP can be downloaded via the following link:

  http://winscp.net/eng/download.php (Installation Package).

  WinSCP also offers a "command shell" that is limited so that commands can be issued, but no callbacks can be answered.

- PuTTY, by contrast, offers a complete command shell.

  PuTTY web page: http://www.chiark.greenend.org.uk/~sgtatham/putty

With both programs, a service technician can log onto the NCU and carry out service tasks. The username ‘manufact’ with the password ‘SUNRISE’ is available for the Siemens service technician.

## Starting WinSCP

WinSCP is started from Windows after the service technician has logged onto the NCU with which they are connected with the corresponding privilege (e.g. as user ‘manufact’ with password ‘SUNRISE’).

From the "Commands“ menu, select ”Open terminal” to open a command shell. There you can carry out the service commands in the usual way.

## See also

Applications (Page 11)

# Appendix A

## A.1 Abbreviations

| | |
|---|---|
| CF | Compact Flash: Memory card |
| DCK | Direct Control Keys Direct control key |
| DHCP | Dynamic Host Configuration Protocol: Dynamic assignment of an IP address and other configuration parameters on a computer in a network |
| DNS | Domain Name System: Conversion of domain names into IP addresses |
| EBS | Emergency Boot System |
| EKS | Electronic Key System: System to check the identity of a user (authentication system) |
| HMI | Human Machine Interface: Operator interface |
| IRT | Isochronous Realtime (Ethernet) |
| MAC | Media Access Control: The MAC address is a 48-bit Ethernet ID. |
| MCP | Machine Control Panel Machine control panel |
| MPI | Multi-Point Interface Multiple interface |
| MUI | Multilanguage User Interface |
| NCK | Numerical Control Kernel: NC kernel with block preparation, travel range, etc. |
| NCU | Numerical Control Unit: NCK hardware unit |
| NRT | Non-Realtime (Ethernet) |
| NTFS | New Technology File System |
| NTP | Network Time Protocol: Standard for synchronizing clocks in the entire network |
| NTPD | NTP Daemon:<br>Utility that runs in the background and does not have to be started by the user. |
| PCU | PC Unit: Computer unit |
| PG | Programming device |
| PLC | Programmable Logic Control: PLC |
| PROFIBUS | Process Field Bus: Standard for the fieldbus communication in automation systems |
| RAM | Random Access Memory: Program memory which can be read and written into |
| RDY | Ready Ready |
| TCU | Thin Client Unit |
| TFTP | Trivial File Transfer Protocol: Very simple data transmission protocol |
| UDP | User Datagram Protocol: NTP is mostly processed via UDP. |
| USB | Universal Serial Bus |
| UPS | Uninterruptible power supply |
| UTC | Universal Time, Coordinated Coordinated universal time |
| VNC | Virtual Network Computing |

## A.2 Licenses for WinSCP and PuTTY

### Copyright of license texts

| | |
|---|---|
| WinSCP: | Copyright © 2000-2005 Martin Přikryl.<br>The application is protected by GNU General Public Licence.<br>Part of the code of this software comes from program PuTTY 0.58 © 1997-2005 Simon Tatham. License agreements for using PuTTY are part of WinSCP license agreement. |
| PuTTYgen | and Pageant applications distributed with WinSCP installation package are © 1997-2005 Simon Tatham.<br>PuTTYgen and Pageant applications distributed with WinSCP installation package are © 1997-2005 Simon Tatham. |

### A.2.1 GNU License

### GNU General Public License

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.

51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

### Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether free of charge or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow:

## TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program. You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the program or any portion of it, thus forming a work based on the program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License.

(Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances. It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice. This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

## NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

## END OF TERMS AND CONDITIONS

## How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms. To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found. one line to give the program's name and an idea of what it does.

Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

Also add information on how to contact you by electronic and paper mail. If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands `show w' and 'show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than 'show w' and 'show c'; they could even be mouse-clicks or menu items -- whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision'(which makes passes at compilers) written by James Hacker.

signature of Ty Coon, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.

Updated: $Date: 2005/06/07 16:30:55 $ $Author: novalis $

## A.2.2 PuTTY Licence

### PuTTY Licence

The PuTTY executables and source code are distributed under the MIT licence, which is similar in effect to the BSD licence. (This licence is Open Source certified and complies with the Debian Free Software Guidelines.)

The precise licence text, as given in the file LICENCE in the source distribution, is as follows:

PuTTY is copyright 1997-2005 Simon Tatham.

Portions copyright Robert de Bath, Joris van Rantwijk, Delian Delchev, Andreas Schultz, Jeroen Massar, Wez Furlong, Nicolas Barry, Justin Bradford, Ben Harris, Malcolm Smith, Ahmad Khalifa, Markus Kuhn, and CORE SDI S.A.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL SIMON TATHAM BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

In particular, anybody (even companies) can use PuTTY without restriction (even for commercial purposes) and owe nothing to me or anybody else. Also, apart from having to maintain the copyright notice and the licence text in derivative products, anybody (even companies) can adapt the PuTTY source code into their own programs and products (even commercial products) and owe nothing to me or anybody else. And, of course, there is no warranty and if PuTTY causes you damage you're on your own, so don't use it if you're unhappy with that.

In particular, note that the MIT licence is compatible with the GNU GPL. So if you want to incorporate PuTTY or pieces of PuTTY into a GPL program, there's no problem with that.

# Glossary

### CFS (Compressed File System)

A CFS (file extension ".cfs") is a compressed file system, similar to a zip file. It contains files and subdirectories that look like normal files on the controller at runtime. Files and directories contained in a CFS cannot be changed. They are decompressed at runtime as required.

### Network interface

The network interface is an interface that enables network communication. These are the Ethernet interfaces on the NCU.

### NFS (Network File System)

NFS is the most common protocol for remote file systems in the world of Unix, and is also available for Windows. NFS is closely based on the Unix privilege model – each time a file is accessed, a UID and GID are supplied which the server then uses to decide whether the operation is permitted. The server relies on the client to provide the correct IDs.

### Remote File System

A file system that is contacted over the network. The files are physically located on another computer in the network (the "server"), but appear locally the same as all other files. Operations performed on these files are sent via the network to the server, instead of being executed directly on a local storage medium (such as a hard drive or CF card).

As a server usually exports more than one file system, a name for the required file system must also be entered in addition to the name of the server.

### SMB (Server Message Block)

SMB is the underlying protocol of MS Windows file systems (also known as drives, releases, shares, etc.). SMB connections are always active in the context of a specific user, who must be known to the server. Exported file systems have a name (release name), by which they can be addressed. The client does not need to know the concrete path on the server.

### Subsystem

A subsystem is a CFS that not only contains a collection of files, but also executes a program, for example, at runtime. To do this, the CFS contains a script that is used to control the starting and stopping of this program.

For this reason, only administrators are permitted to set up NFS file systems, and NFS is usually only implemented in uniformly administrated environments. Exported file systems on the server are addressed directly on the server via their path.

## VNC (Virtual Network Computing)

Virtual Network Computing is a software that displays the screen contents of a remote computer, with a running VNC server, on a local computer, with a running VNC viewer, and in return sends keyboard and mouse movements of the local computer to the remote computer.

**Reference:** Operator Components and Networking Manual

# Index

## U



## V



## W

# Appendix A

## A.1 Feedback on the documentation

This document will be continuously improved with regard to its quality and ease of use. Please help us with this task by sending your comments and suggestions for improvement via e-mail or fax to:

| | |
|---|---|
| E-mail: | mailto:docu.motioncontrol@siemens.com |
| Fax: | +49 (0) 9131/98 - 63315<br>Please use the fax form on the back of this page. |

| To: | From |
| --- | --- |
| SIEMENS AG<br>A&D MC MS1<br>Postfach 3180<br><br>91050 ERLANGEN, GERMANY | Name: |
| | Address of your Company/Dept. |
| | Street: |
| | Zip code: City: |
| | Phone: / |
| Fax: +49 (0) 9131/98 - 63315 (documentation) | Fax: / |

Suggestions and/or corrections

# A.2 Overview

## Document Overview SINUMERIK 840D sl / 840Di sl

### General Documentation

SINUMERIK

Brochure

SINUMERIK
840D sl
840Di sl

Catalog NC 61 *)

SINAMICS
S120

Catalog
D21.1 Converter chassis units *)

### User Documentation

SINUMERIK
840D sl

Operating Manual
– HMI sl Universal / Turning / Milling *)
– HMI Embedded *)
– ShopMill
– ShopTurn

SINUMERIK
840D sl
840D
840Di sl
840Di

Operating Manual
– HMI Advanced *)
– Operation (compact)

SINUMERIK
840D sl
840D
840Di sl
840Di

Programming Manual
– Fundamentals *)
– Job planning *)
– Programming (compact)
– System Variable Lists
– ISO Turning/Milling

SINUMERIK
840D sl
840D
840Di sl
840Di

Programming Manual
– Cycles
– Measuring Cycles

SINUMERIK
840D sl
840Di sl

Diagnostics Manual *)

### Manufacturer / Service Documentation

SINUMERIK
840D sl

Manual
– NCU *)
– Machine configuration

SINUMERIK
840D sl
840Di sl

Manual
Operator Components and Networking *)

SINUMERIK
840D sl

Commissioning Manual CNC
– NCK, PLC, Drive *)
– HMI sl *)
– HMI Embedded *)
– HMI Advanced *)
– ShopMill / ShopTurn

SINUMERIK
840Di sl

– Commissioning Manual
– ADI4

SINUMERIK
840D sl
840Di sl

Parameter Manual *)
– Part 1
– Part 2

### Manufacturer / Service Documentation

SINUMERIK
840D sl
840D
840Di sl
840Di

Function Manual
– Basic Functions *)
– Extended Functions
– Special Functions
– Synchronous Actions

SINUMERIK
840D sl
840D
840Di sl
840Di

Function Manual
– Tool Management
– ISO Dialects
– SinuCOM

SINAMICS
S120

Function Manual
Drive Functions

SINUMERIK
840D sl

Function Manual
Safety Integrated

SINUMERIK

EMC Directives

### Electronic Documentation

SINUMERIK
SINAMICS
Motors

DOCONCD *)
DOCONWEB

*) Recommended minimum scope of documentation